2022/10/26, Wed.

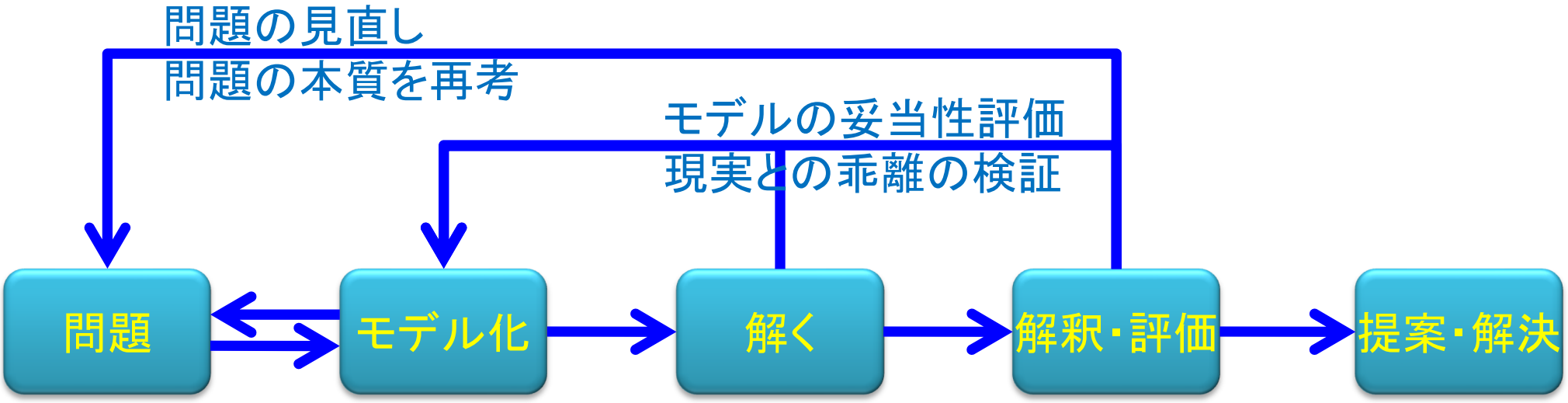
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



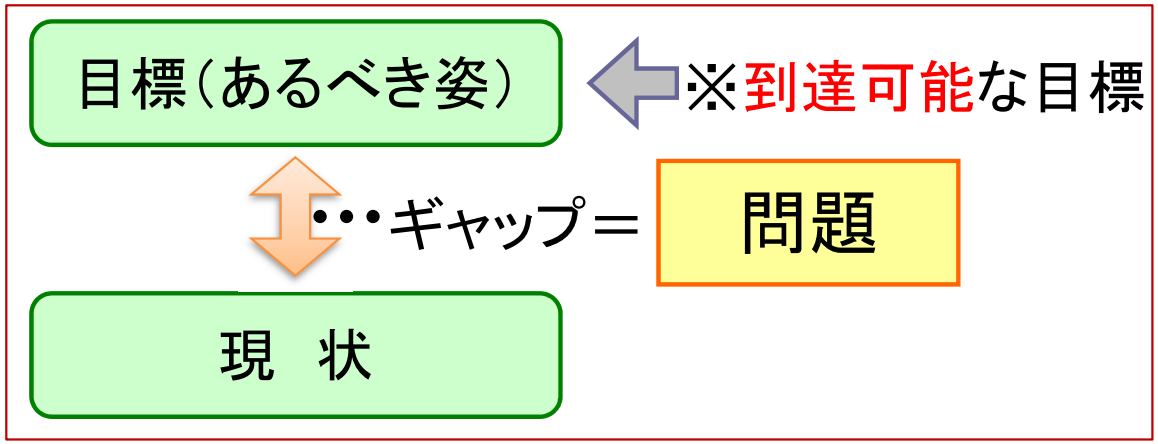
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

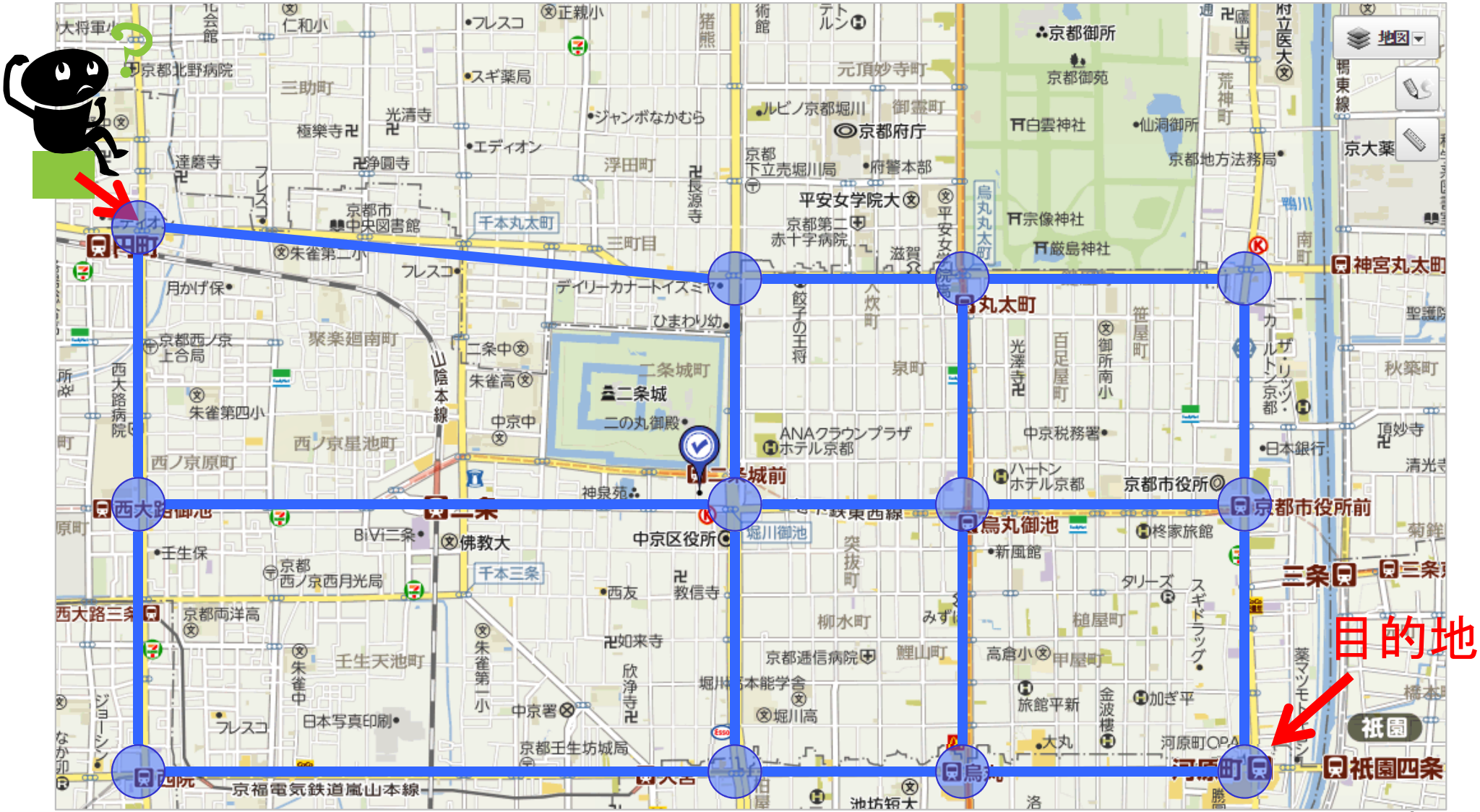
意思決定

問題の定義



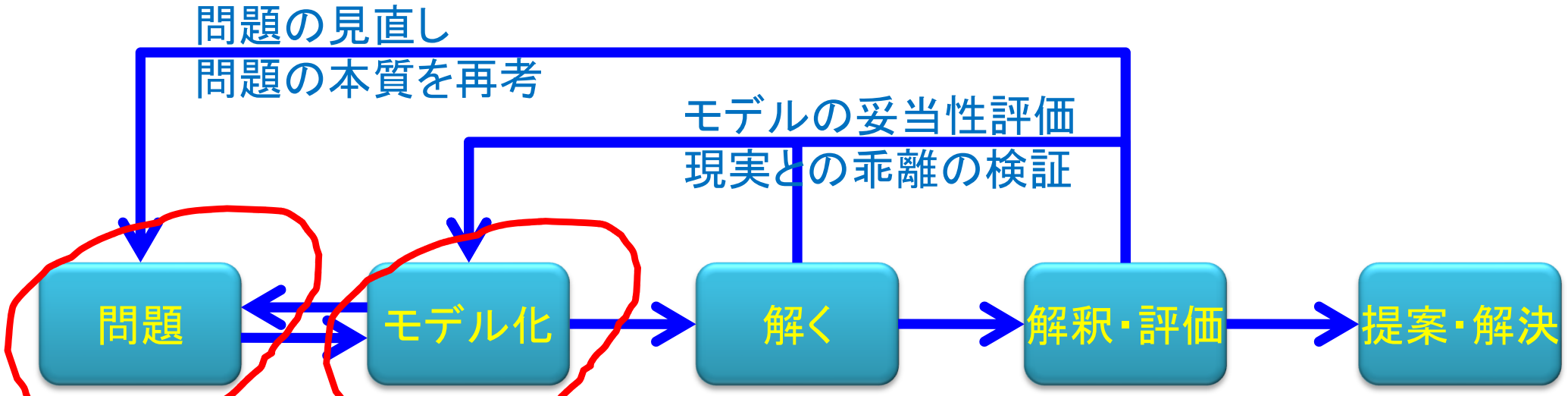
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

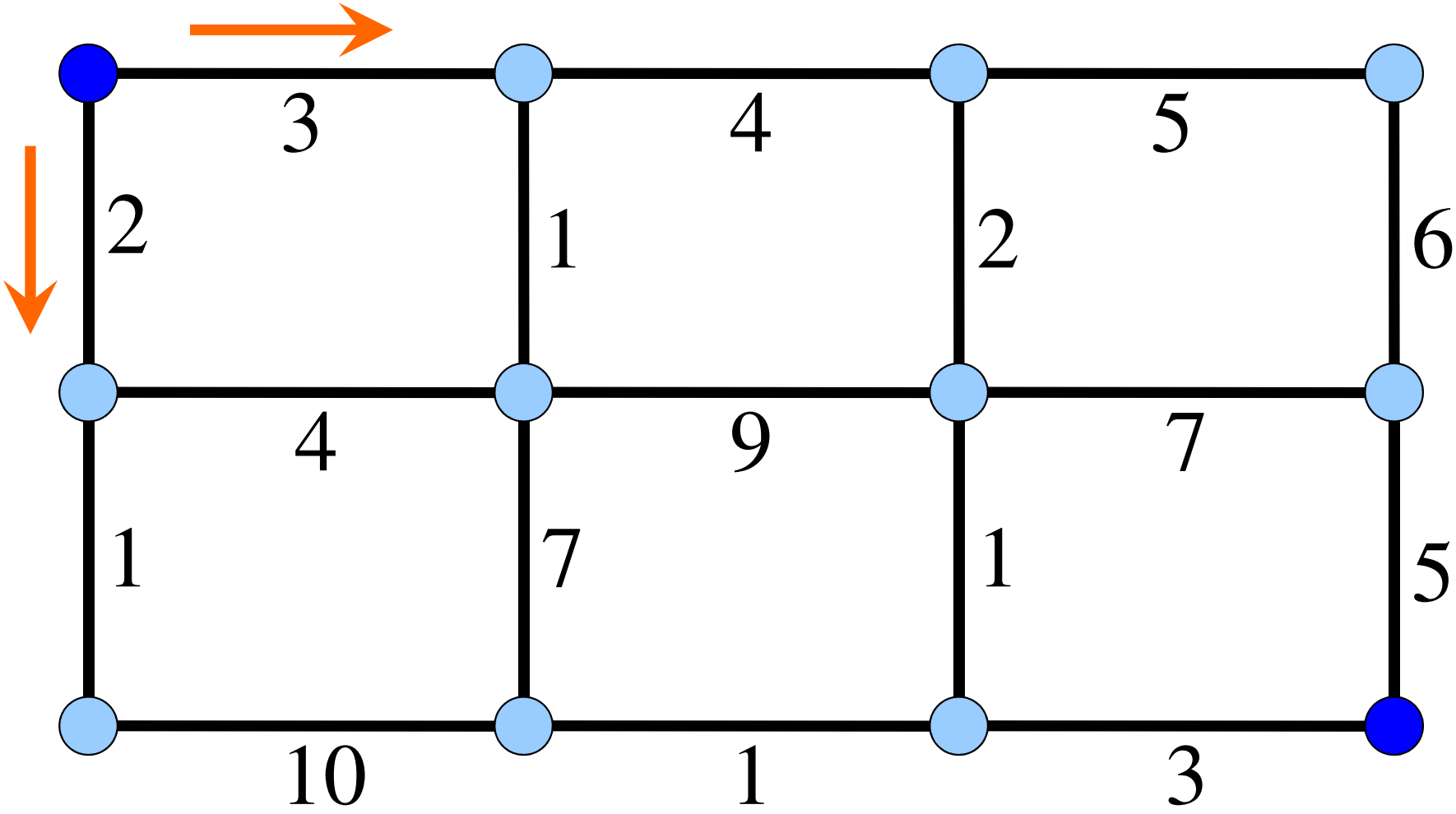
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

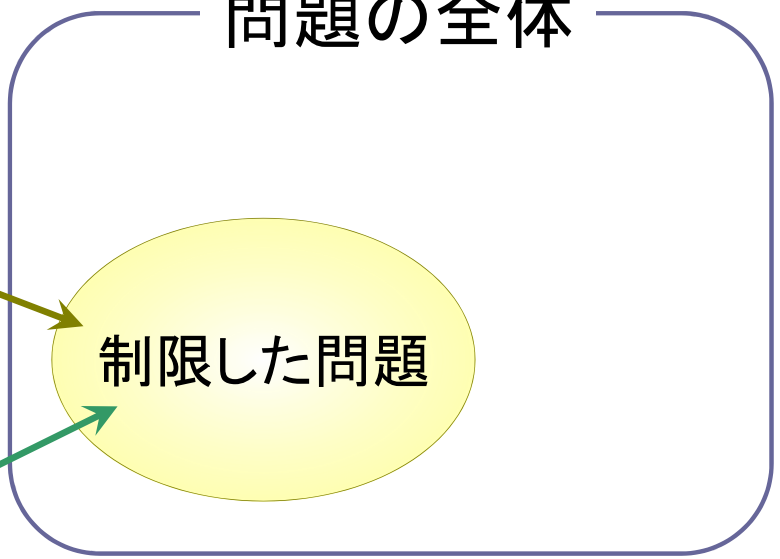


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

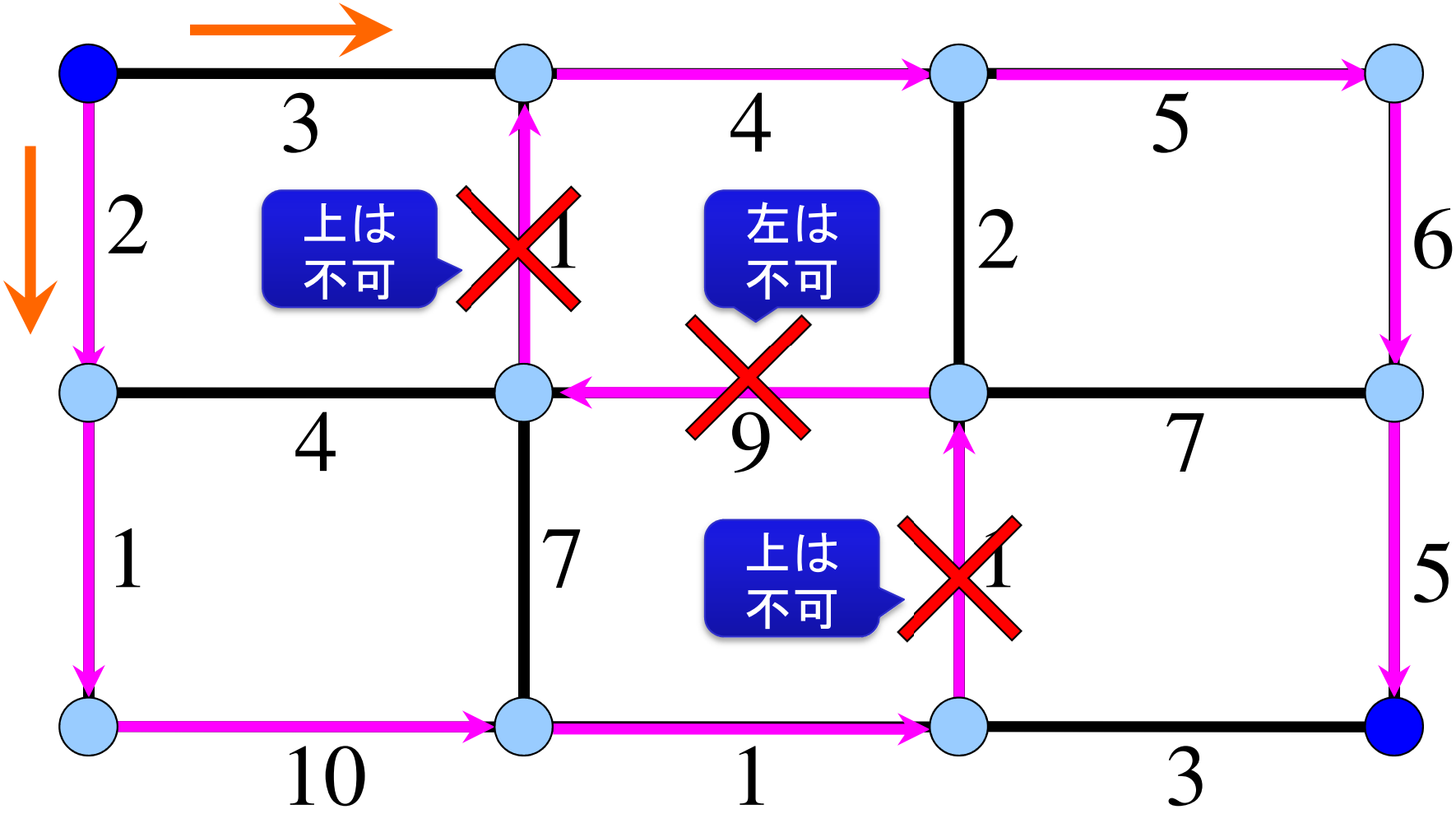
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

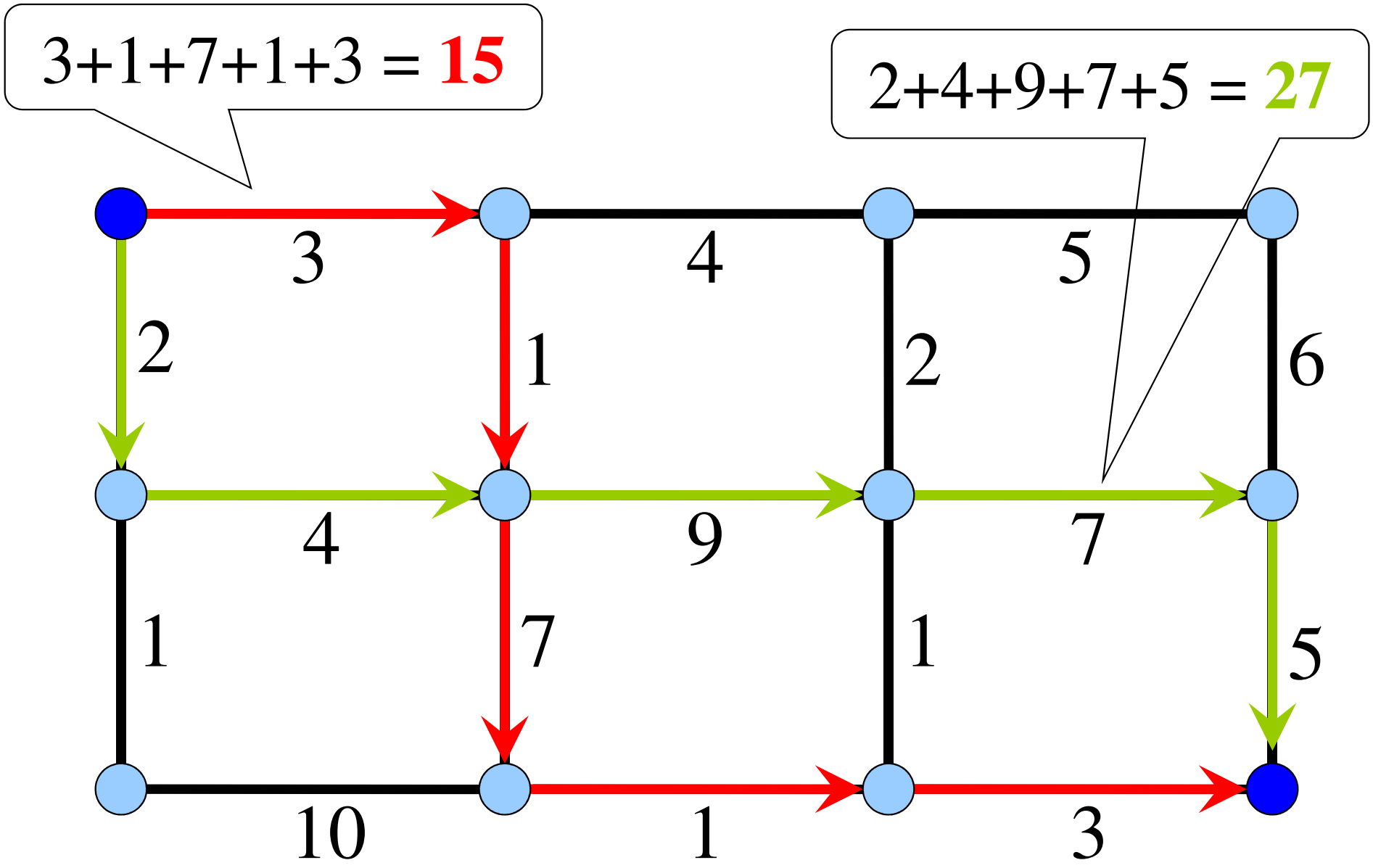
難しいなら易しくすればいいのさ！

制限した問題

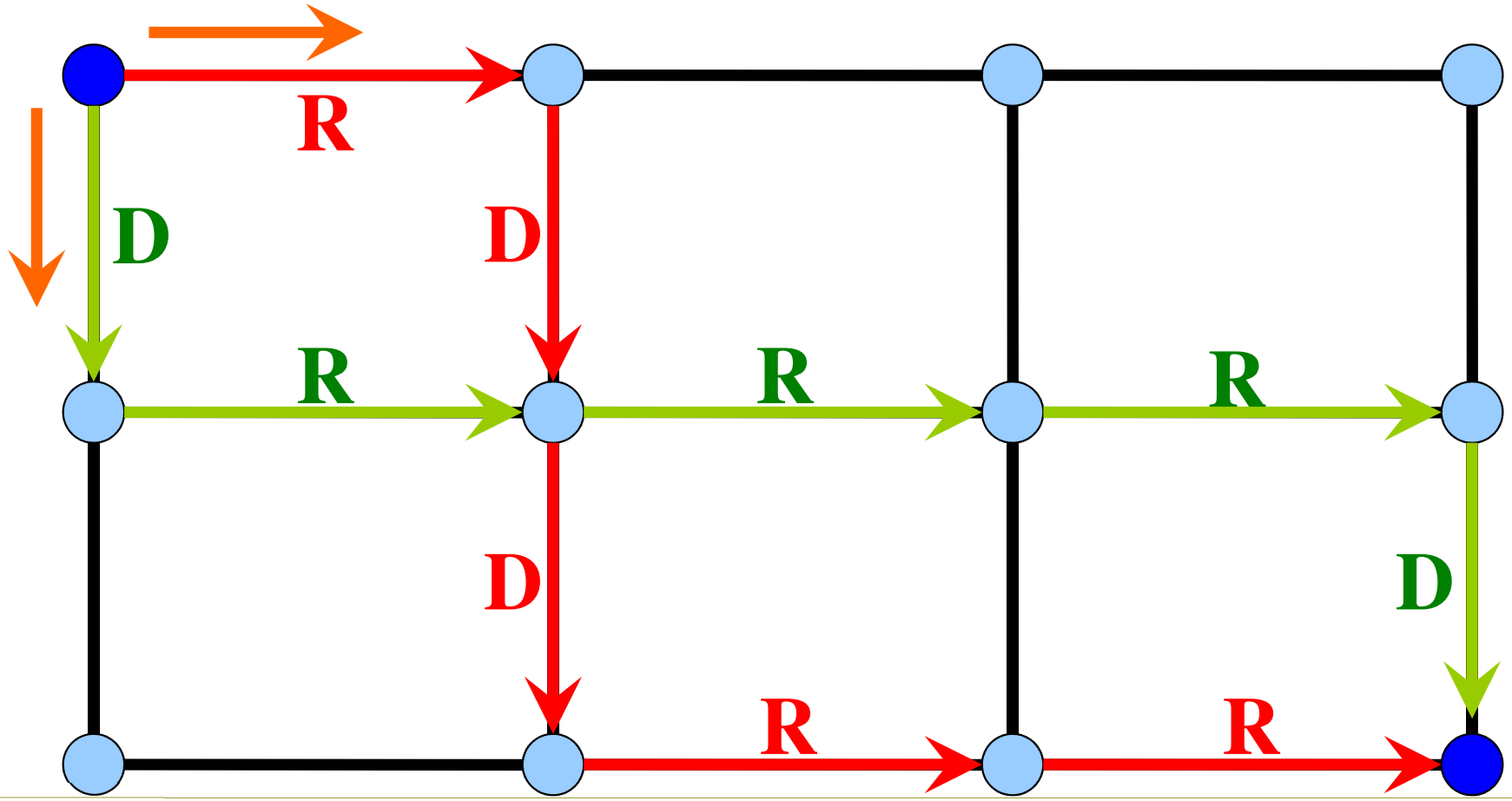
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



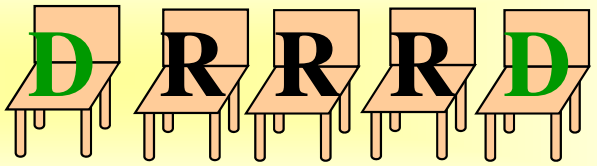
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



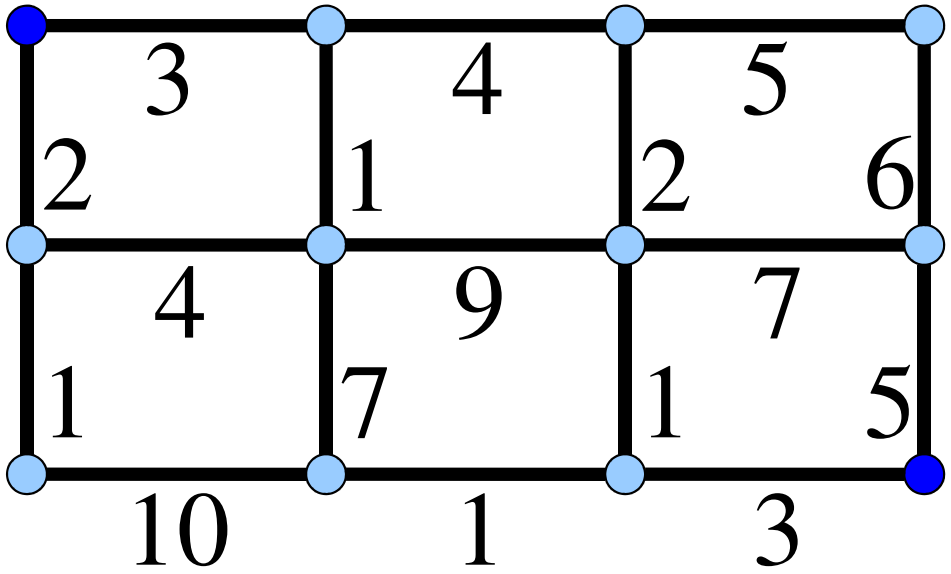
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

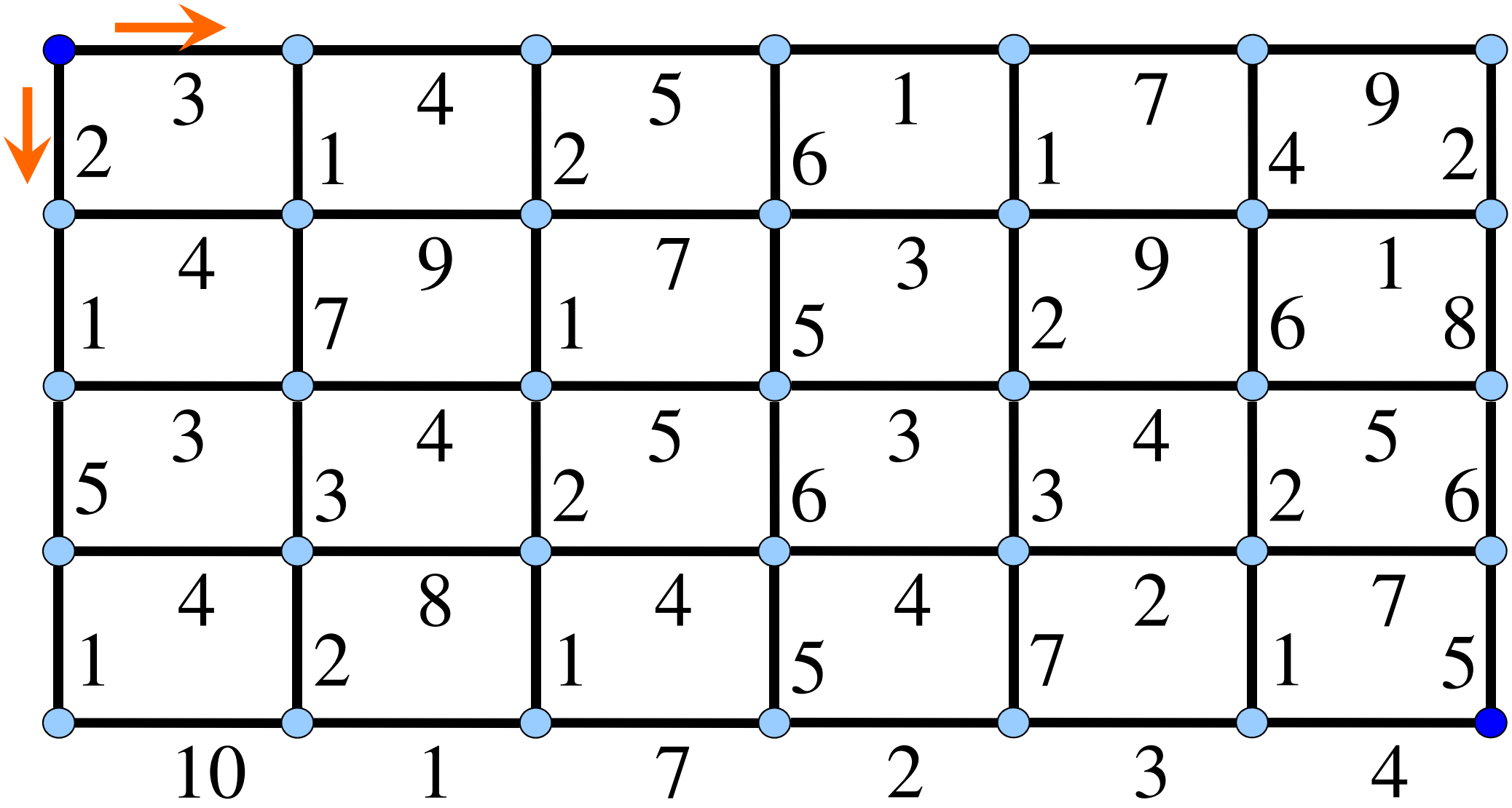
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

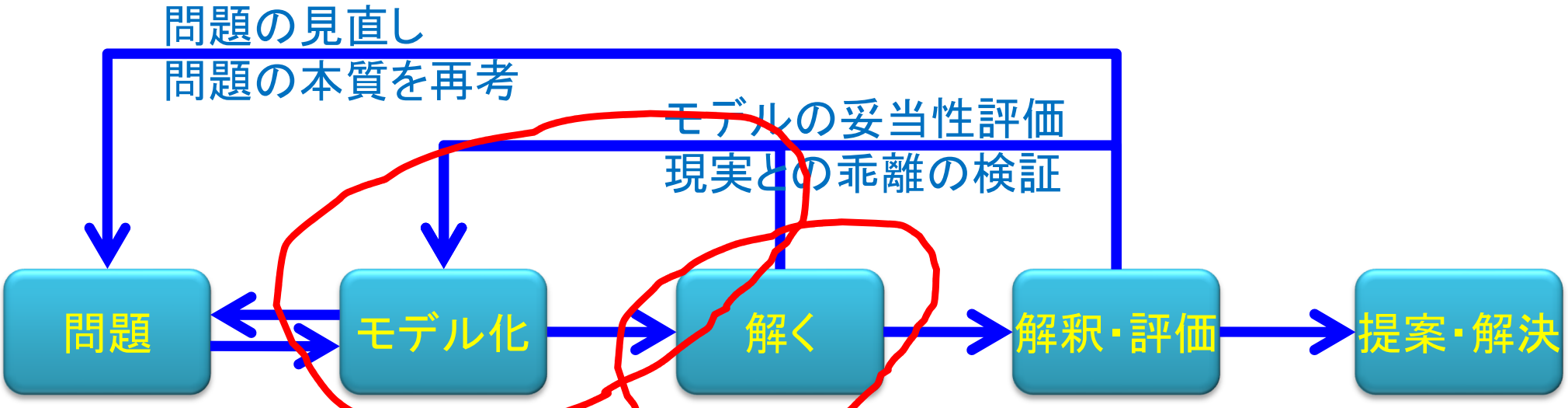
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

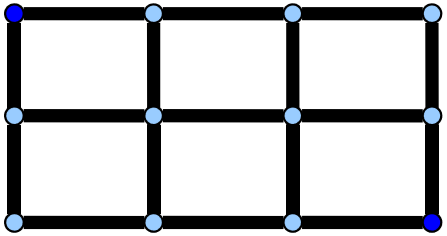
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

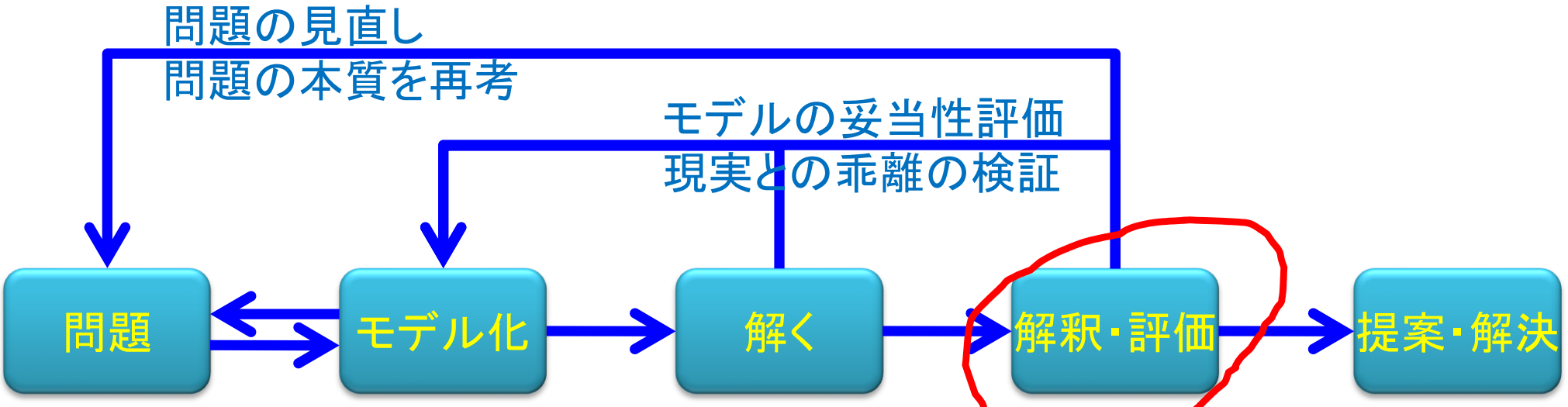
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

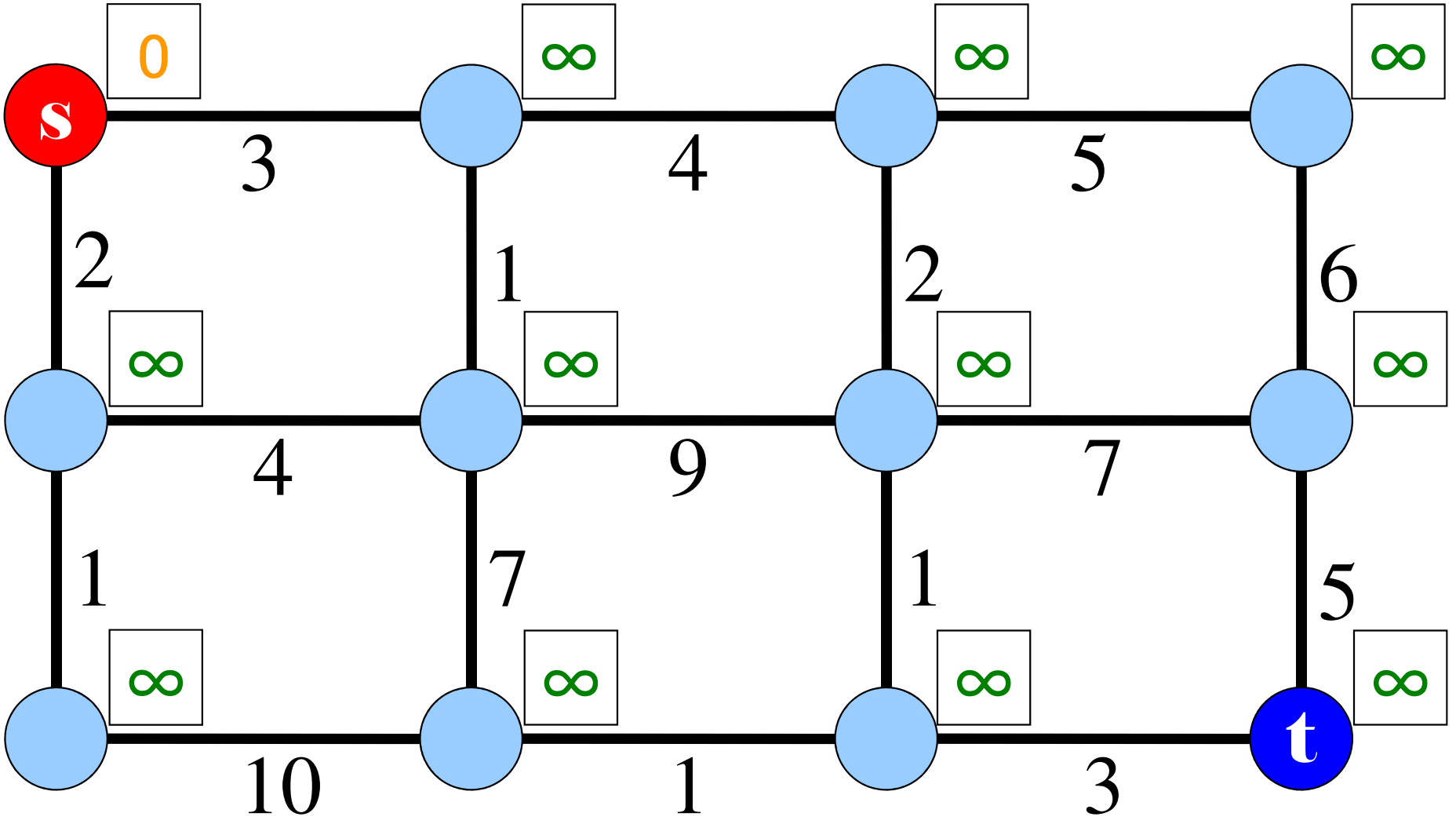
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

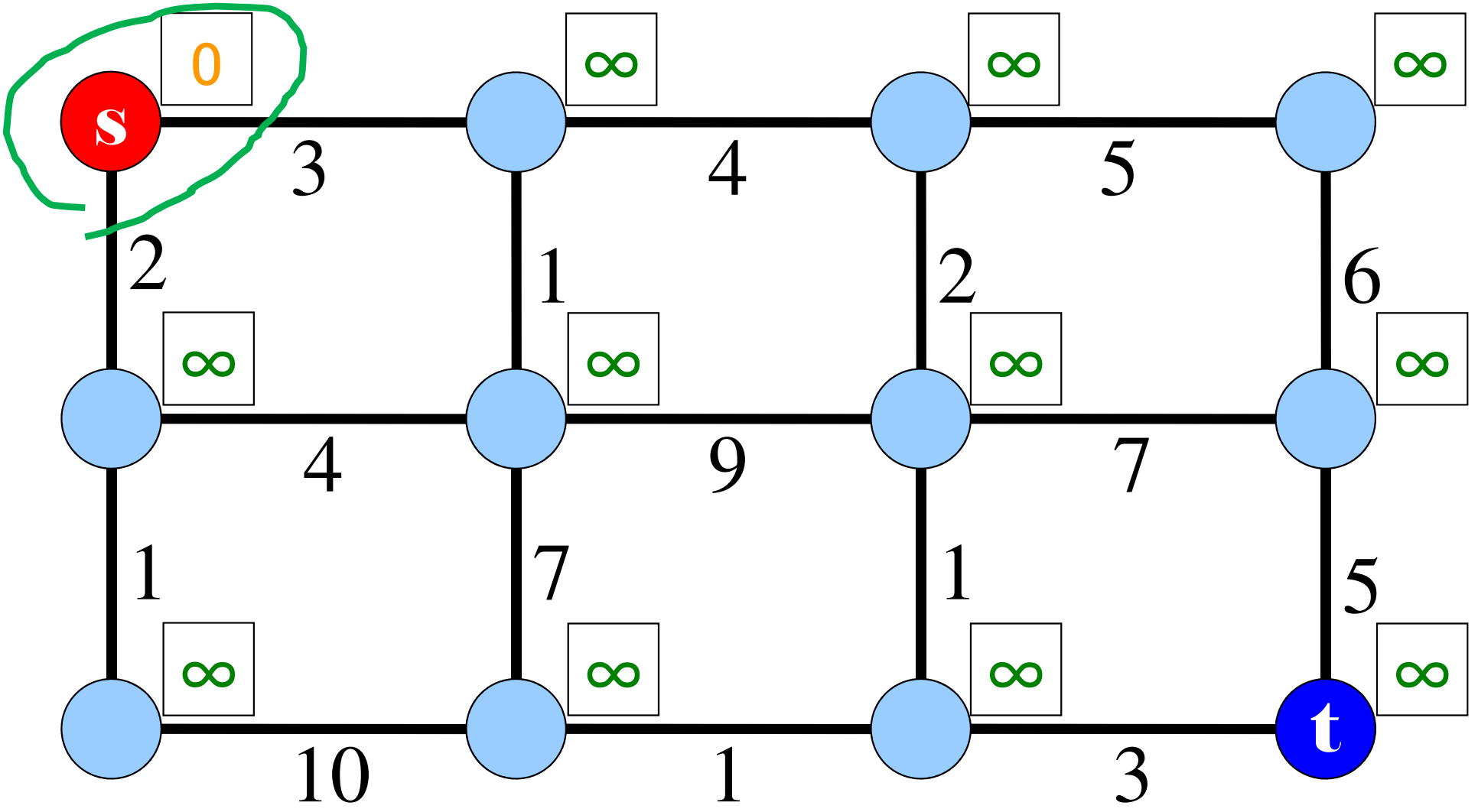
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



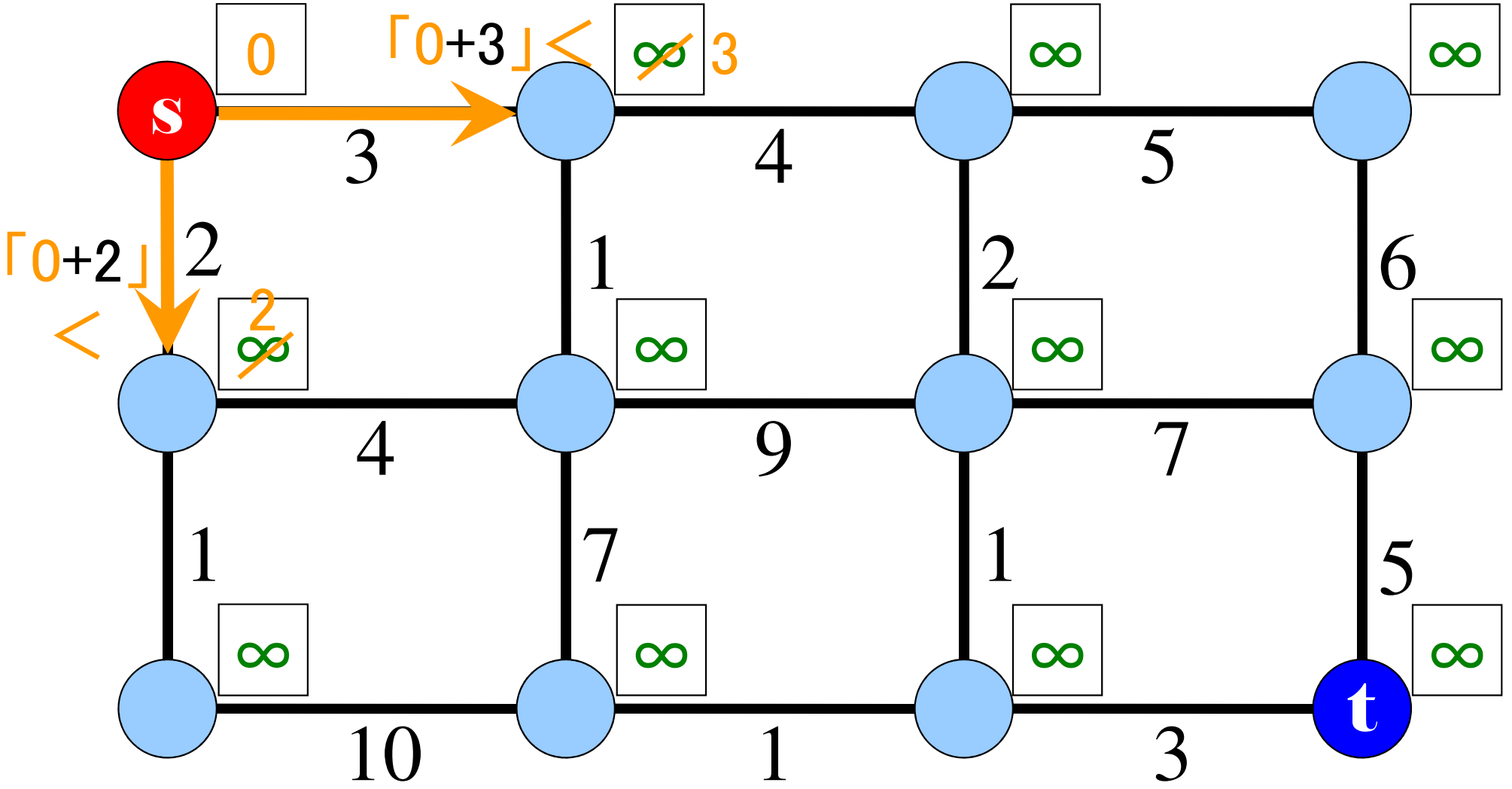
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

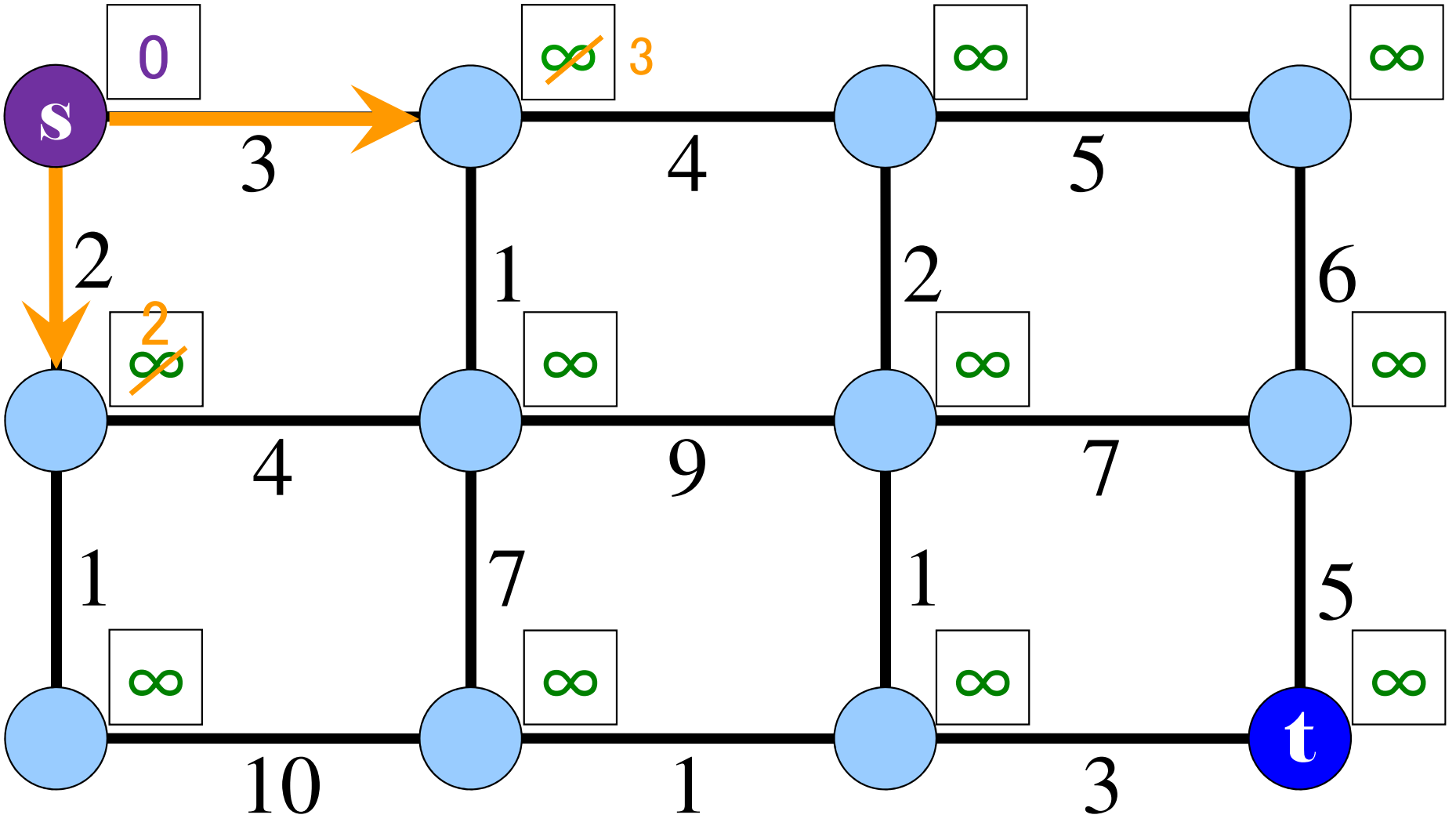


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

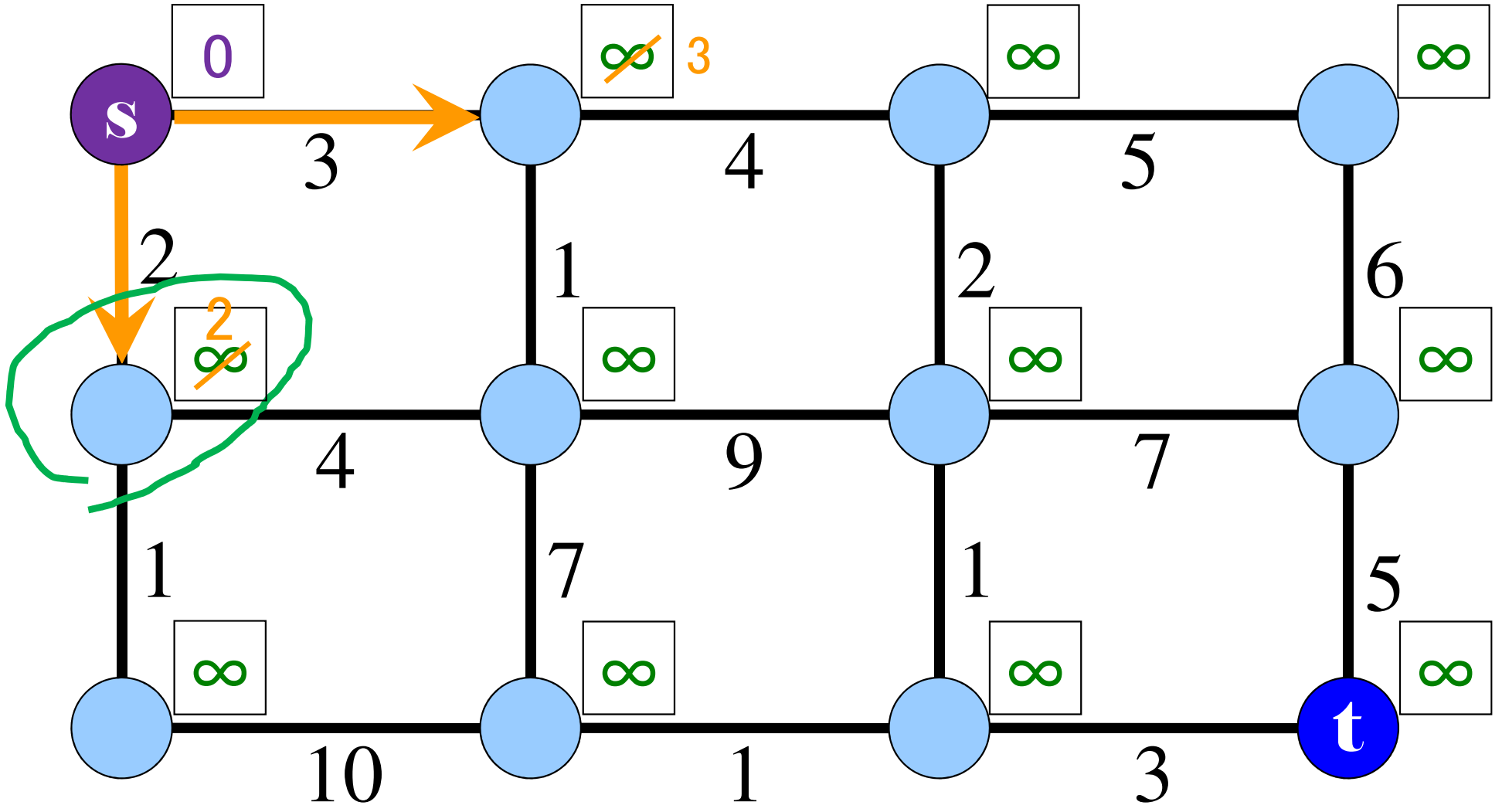
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



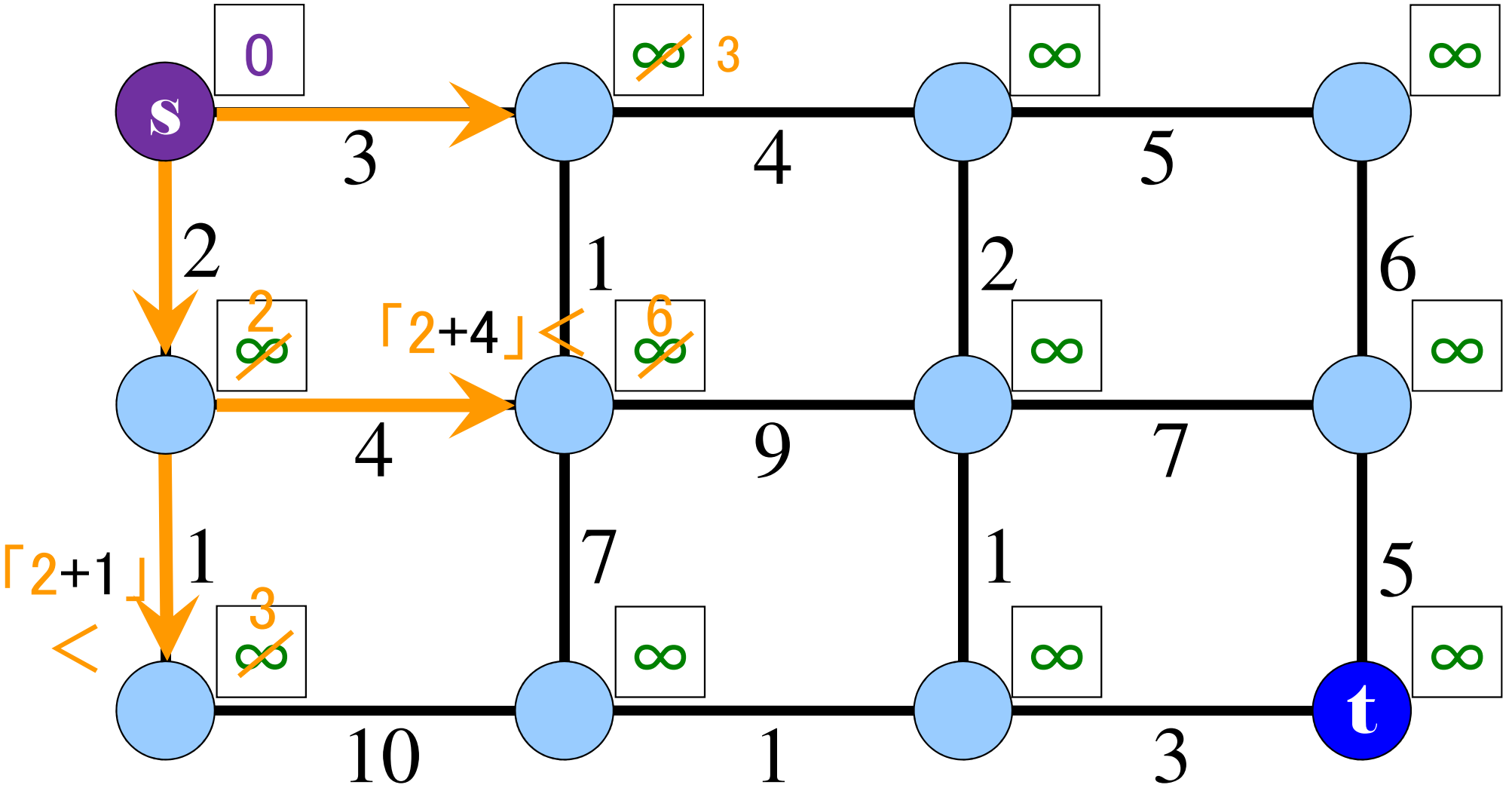
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

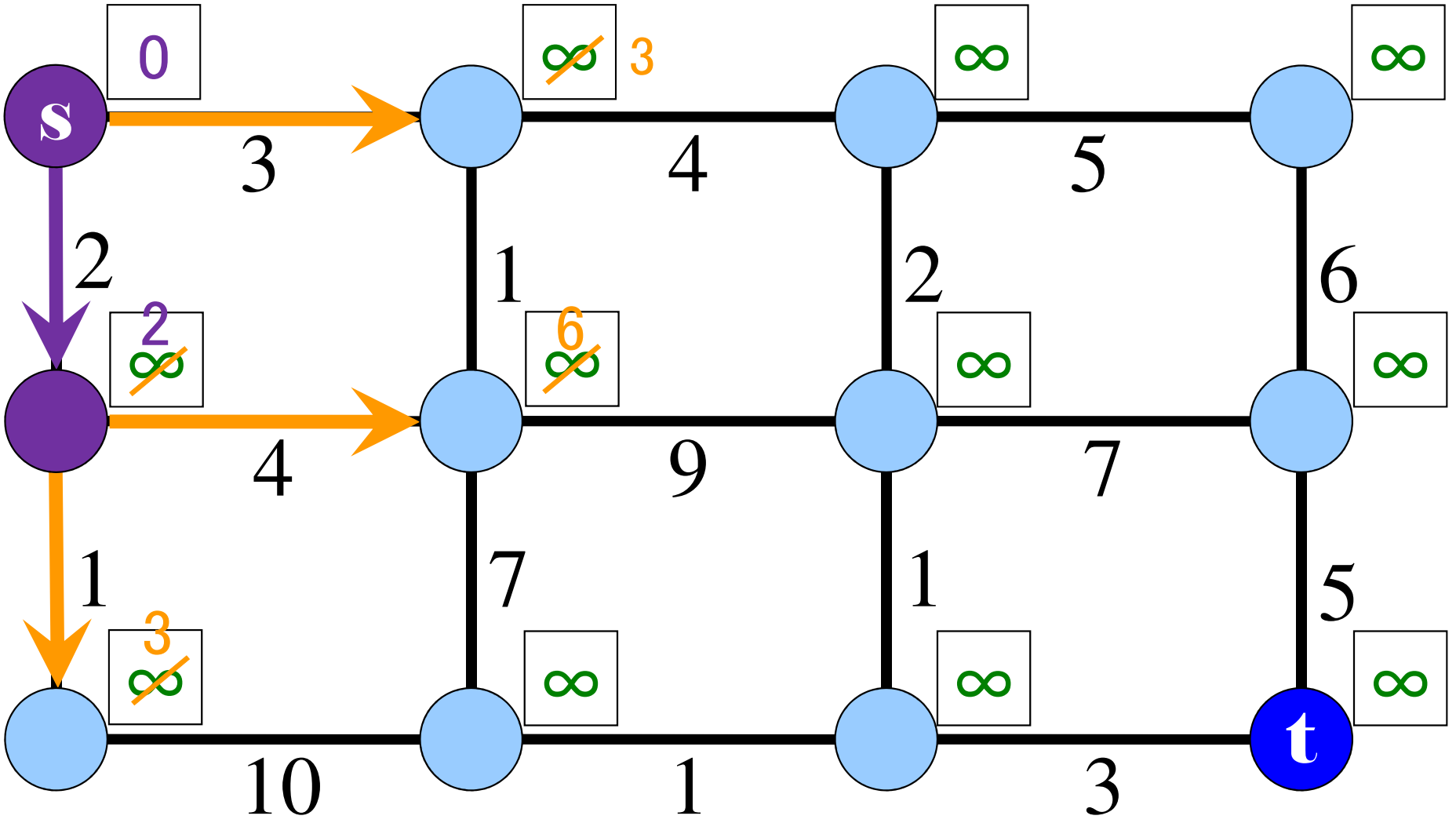


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

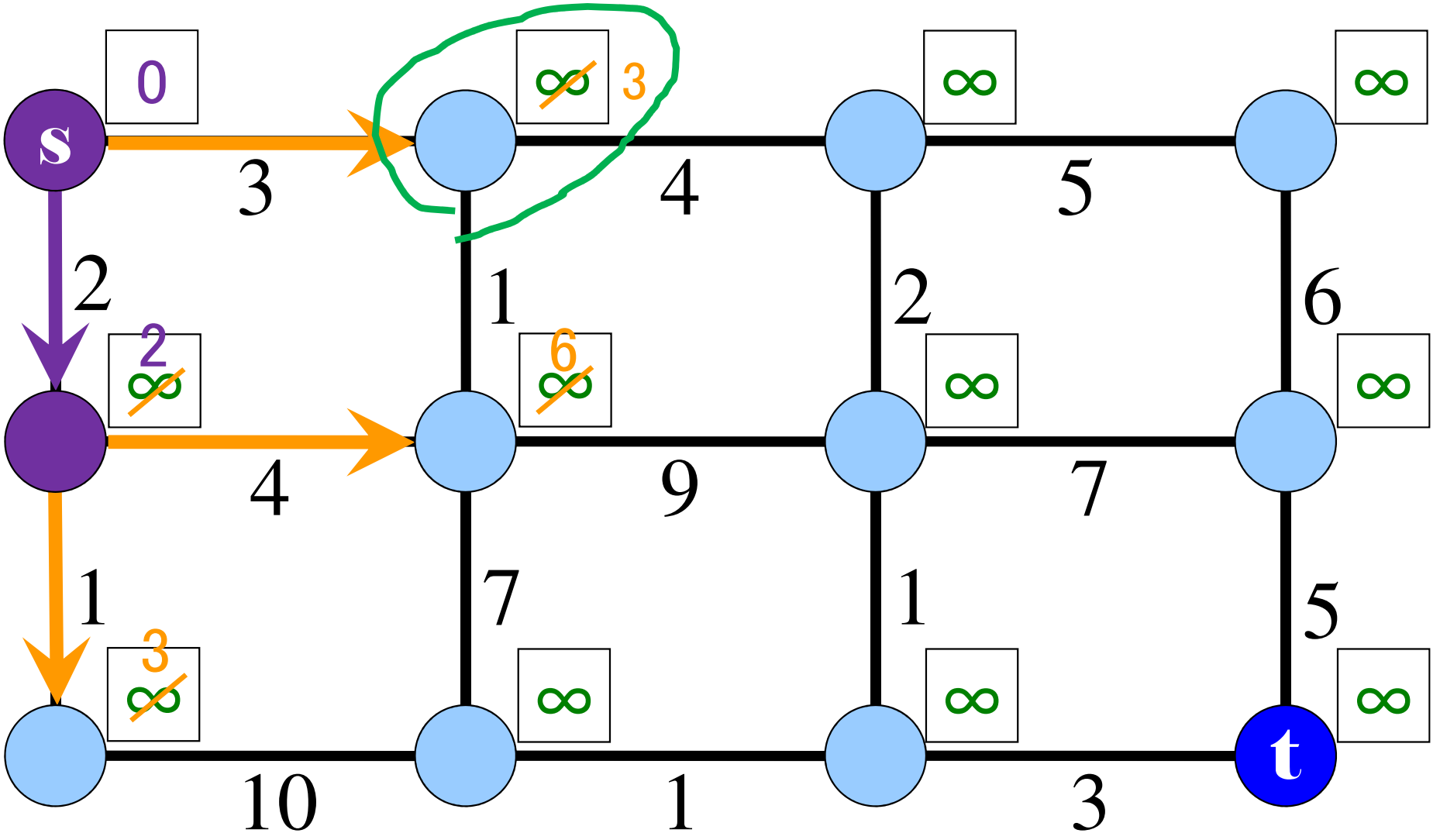
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



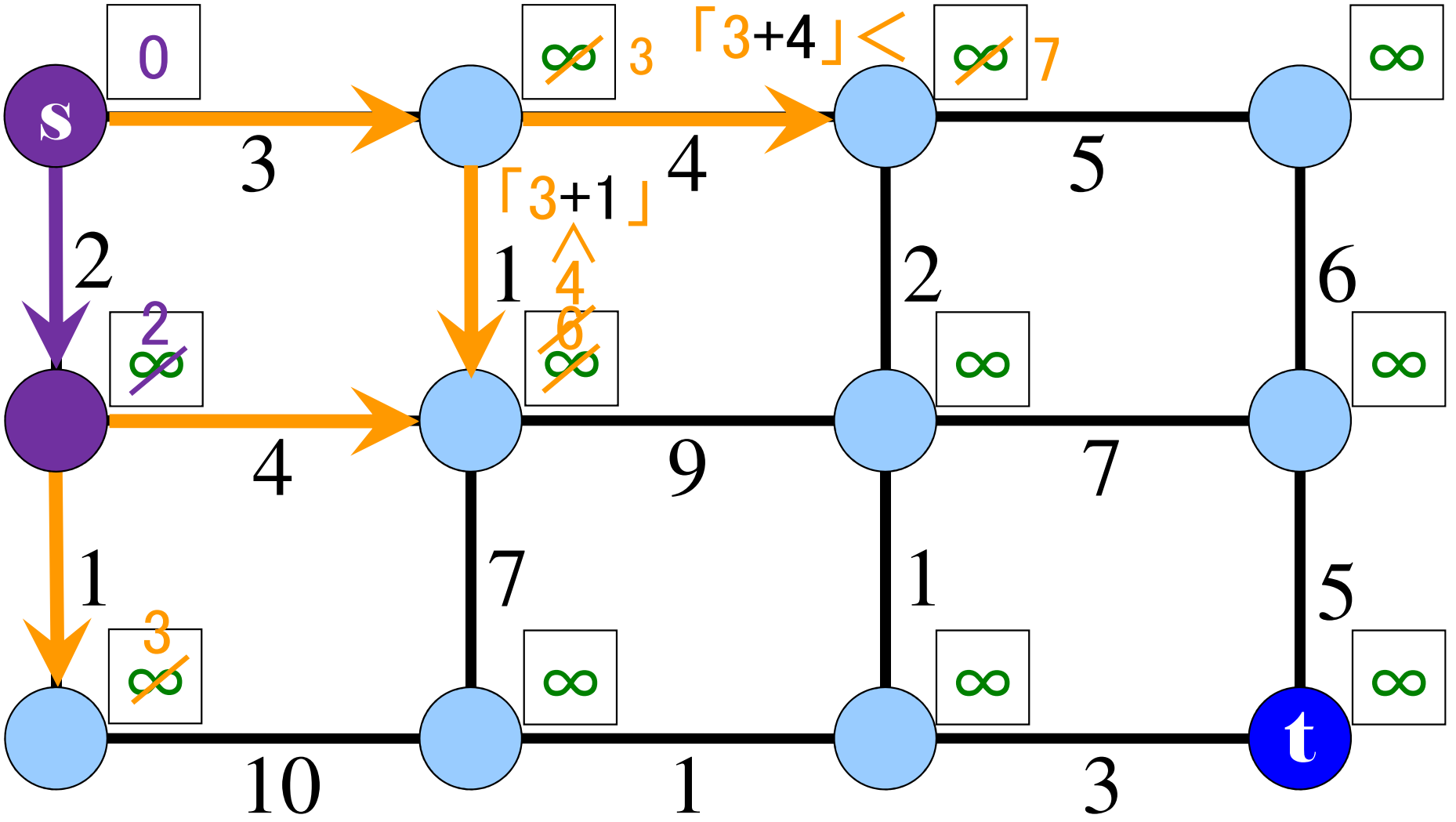
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

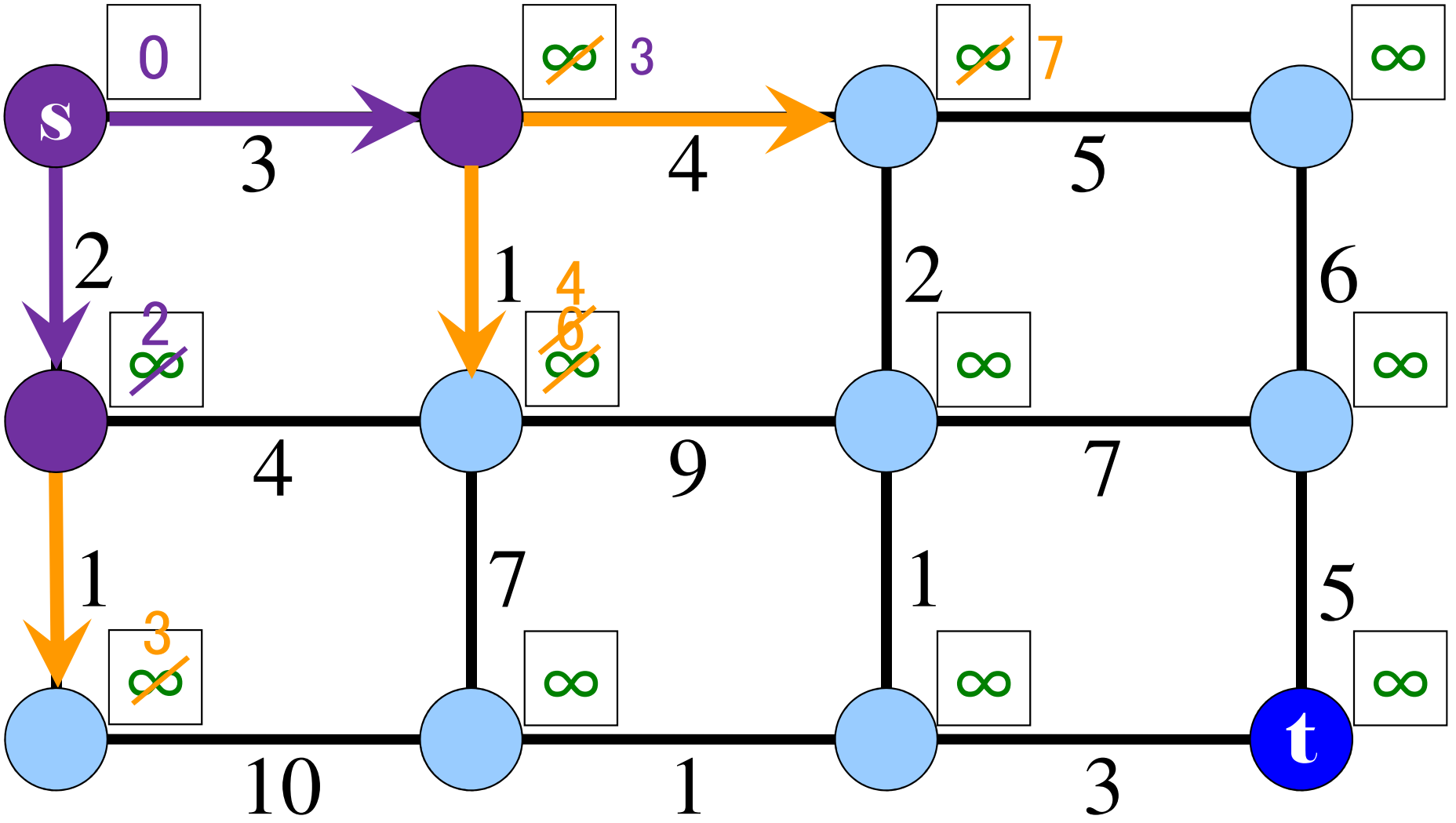


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

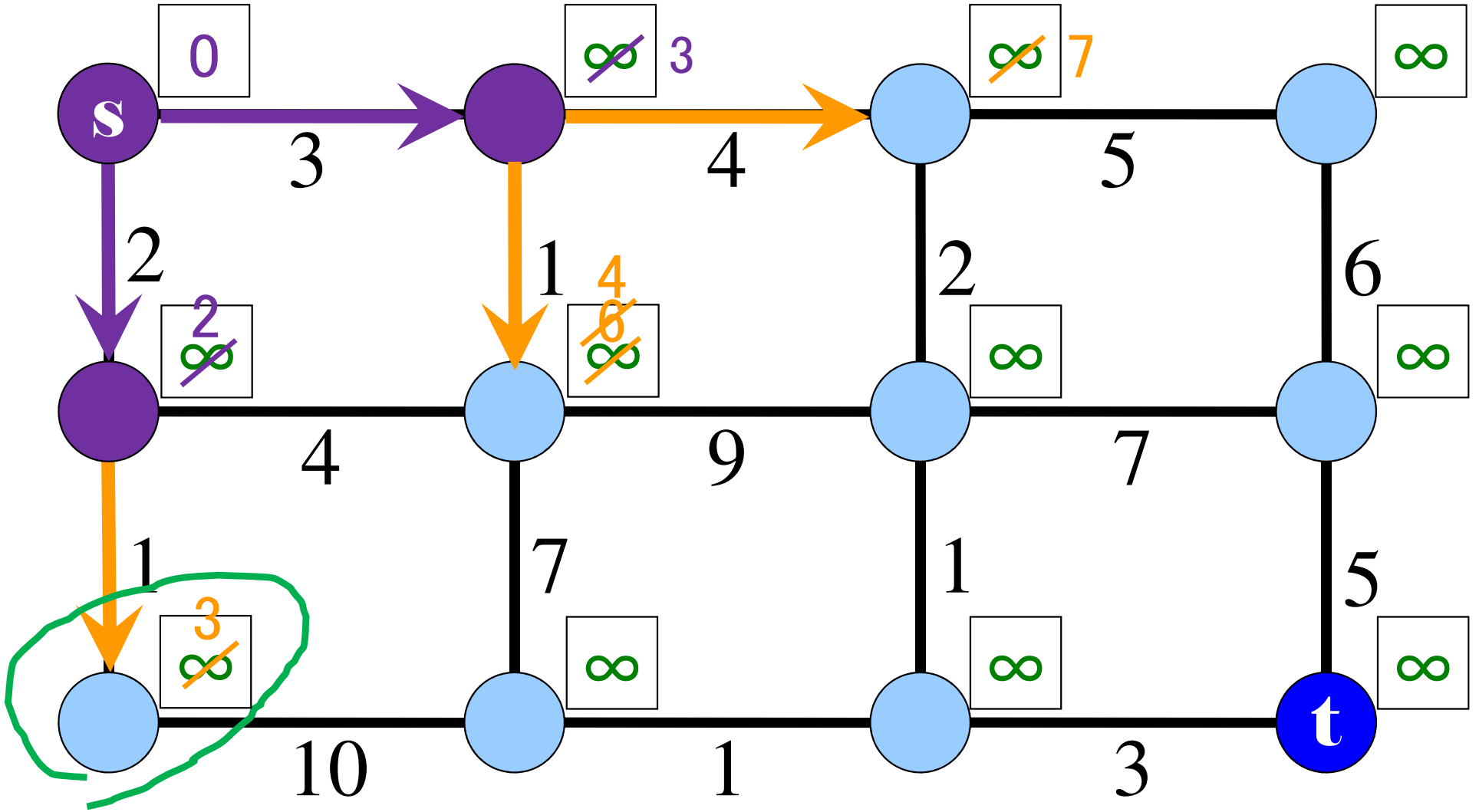
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



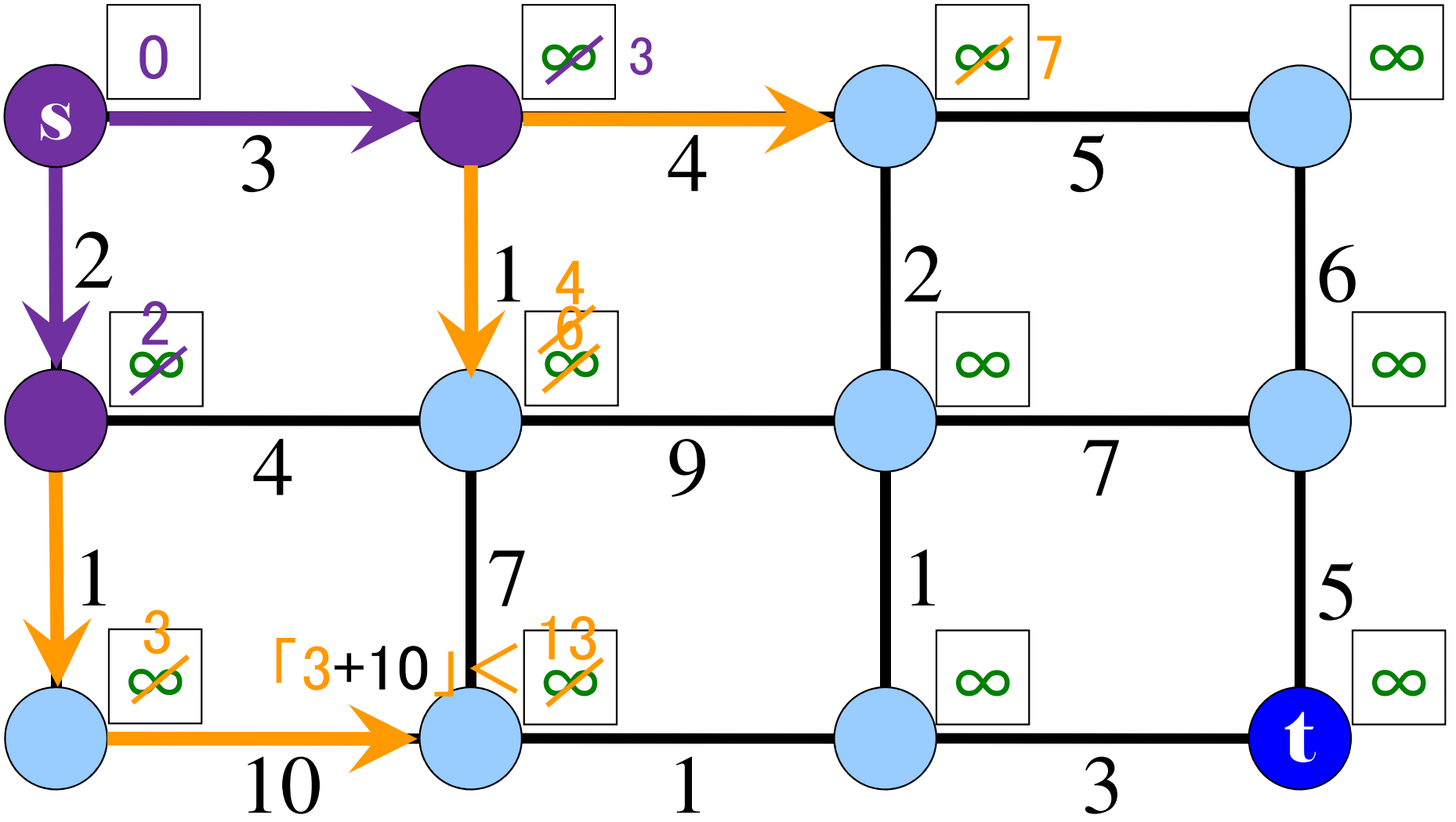
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

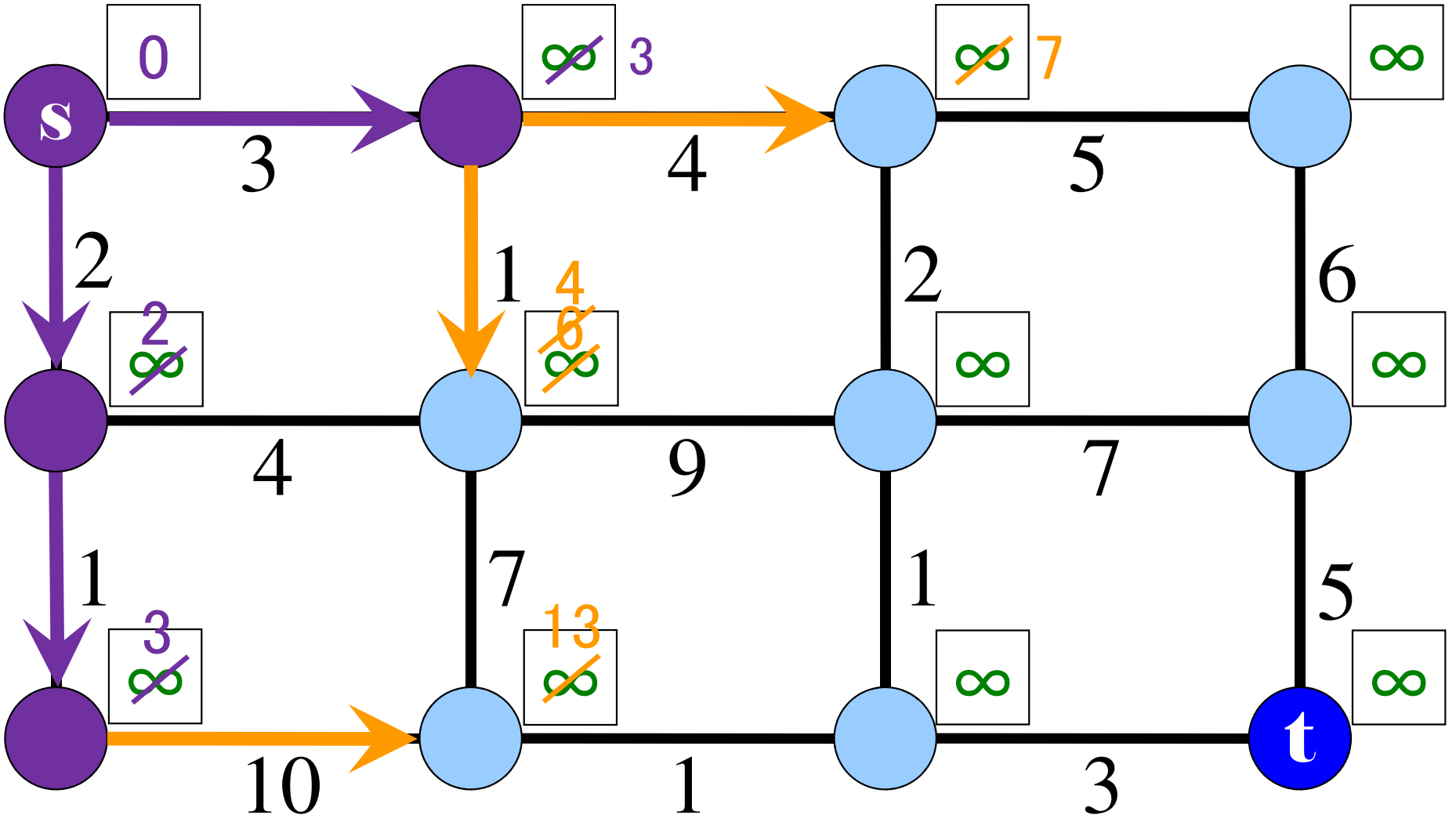


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

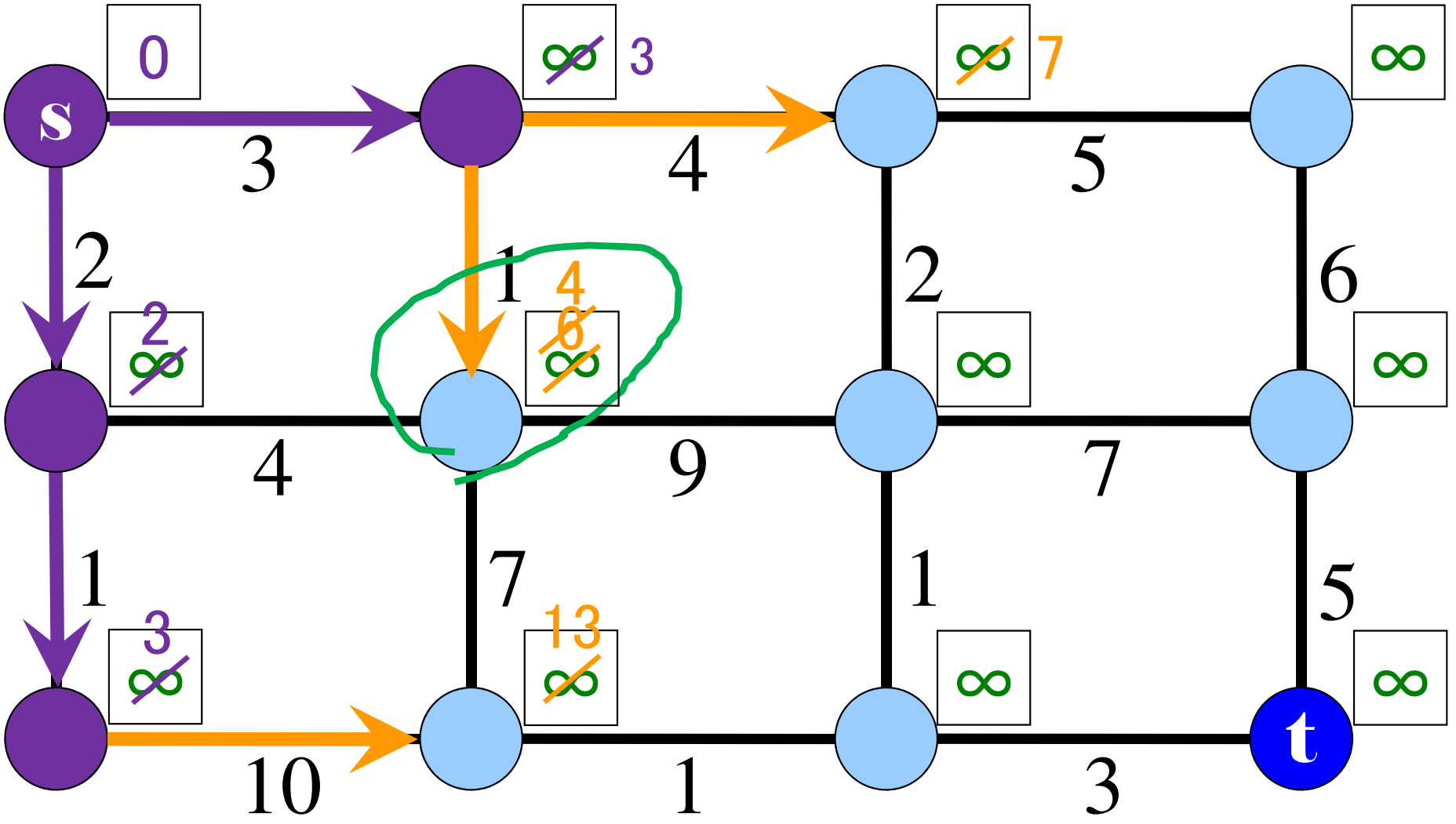
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



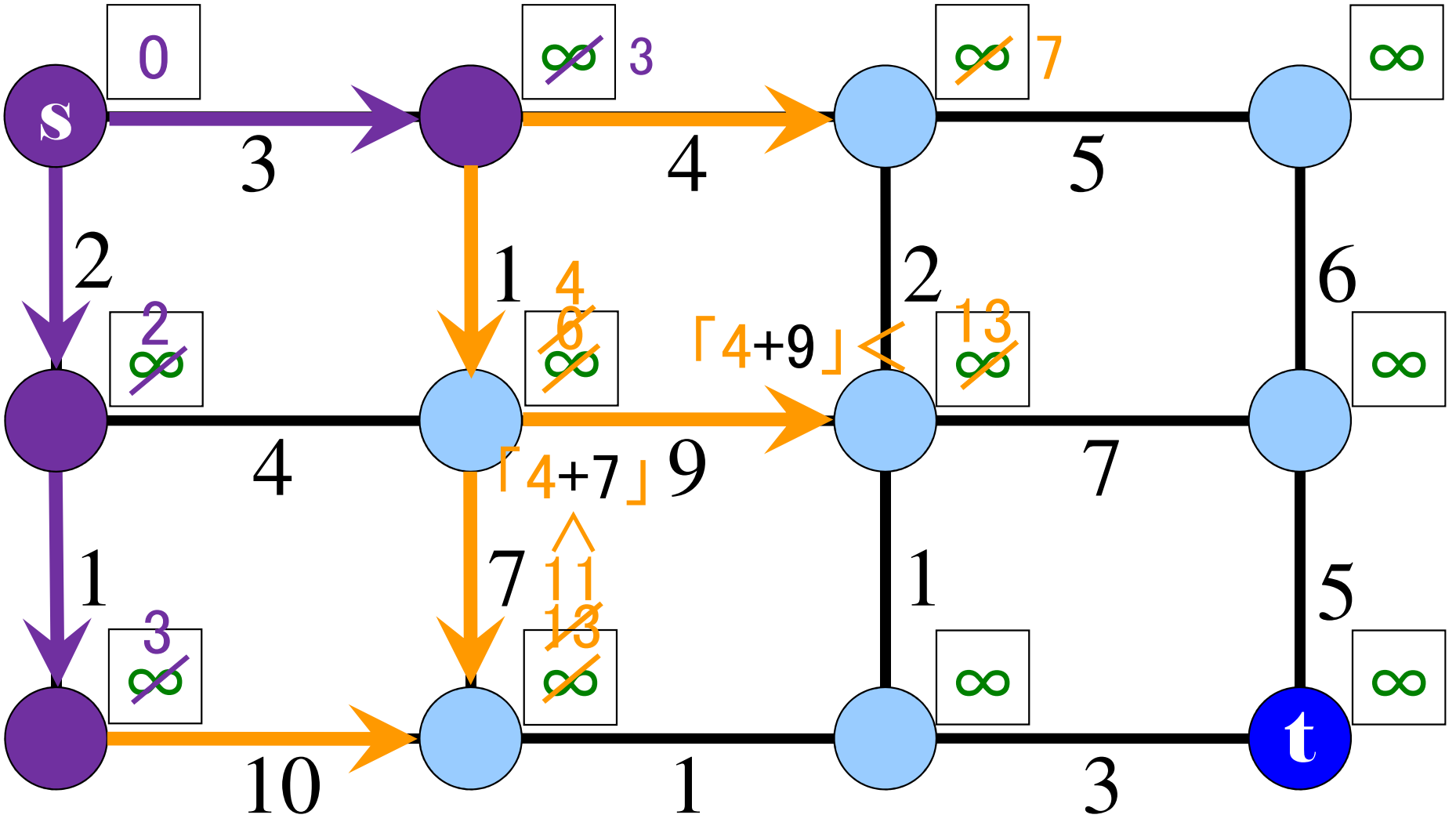
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

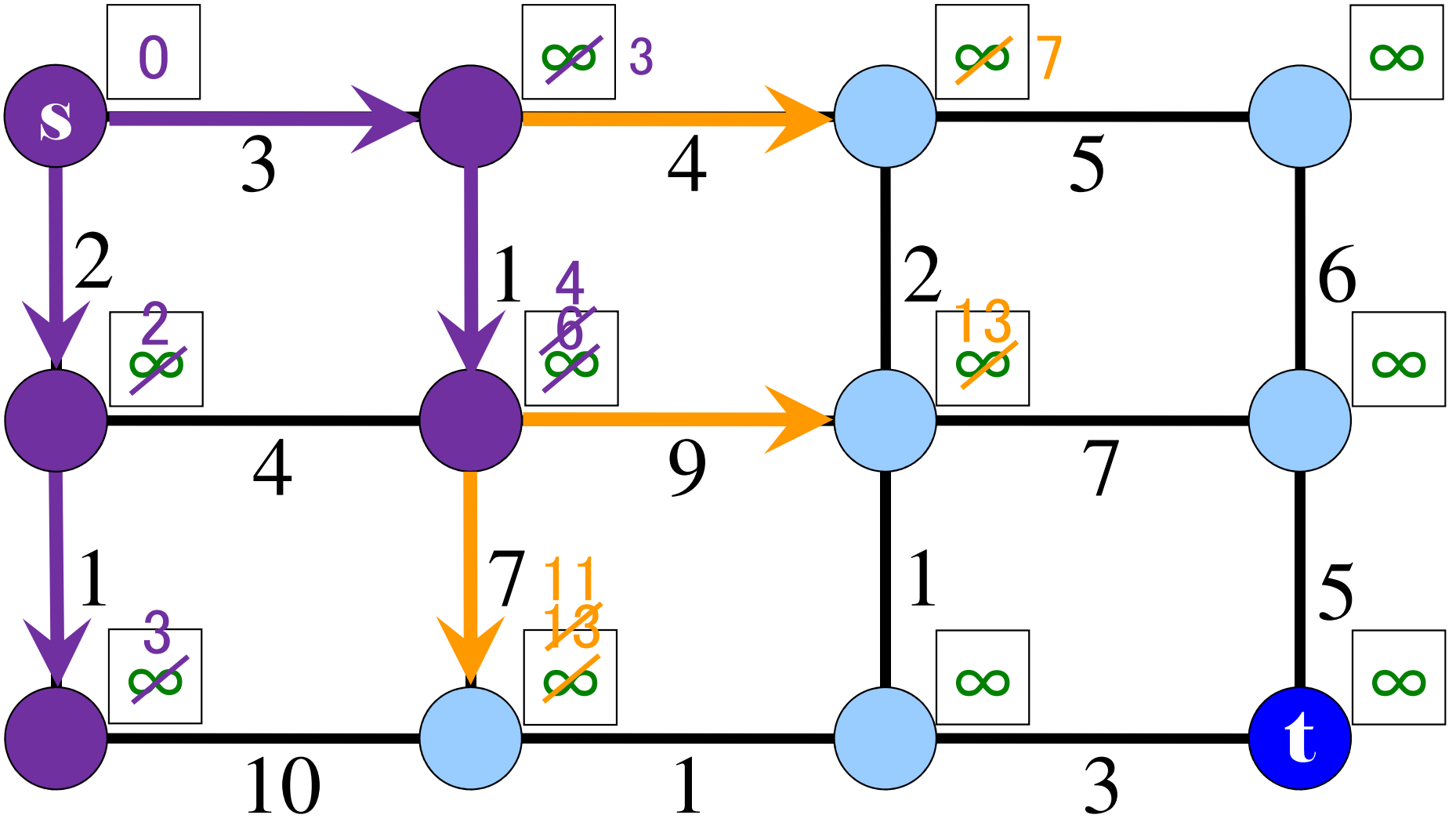


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

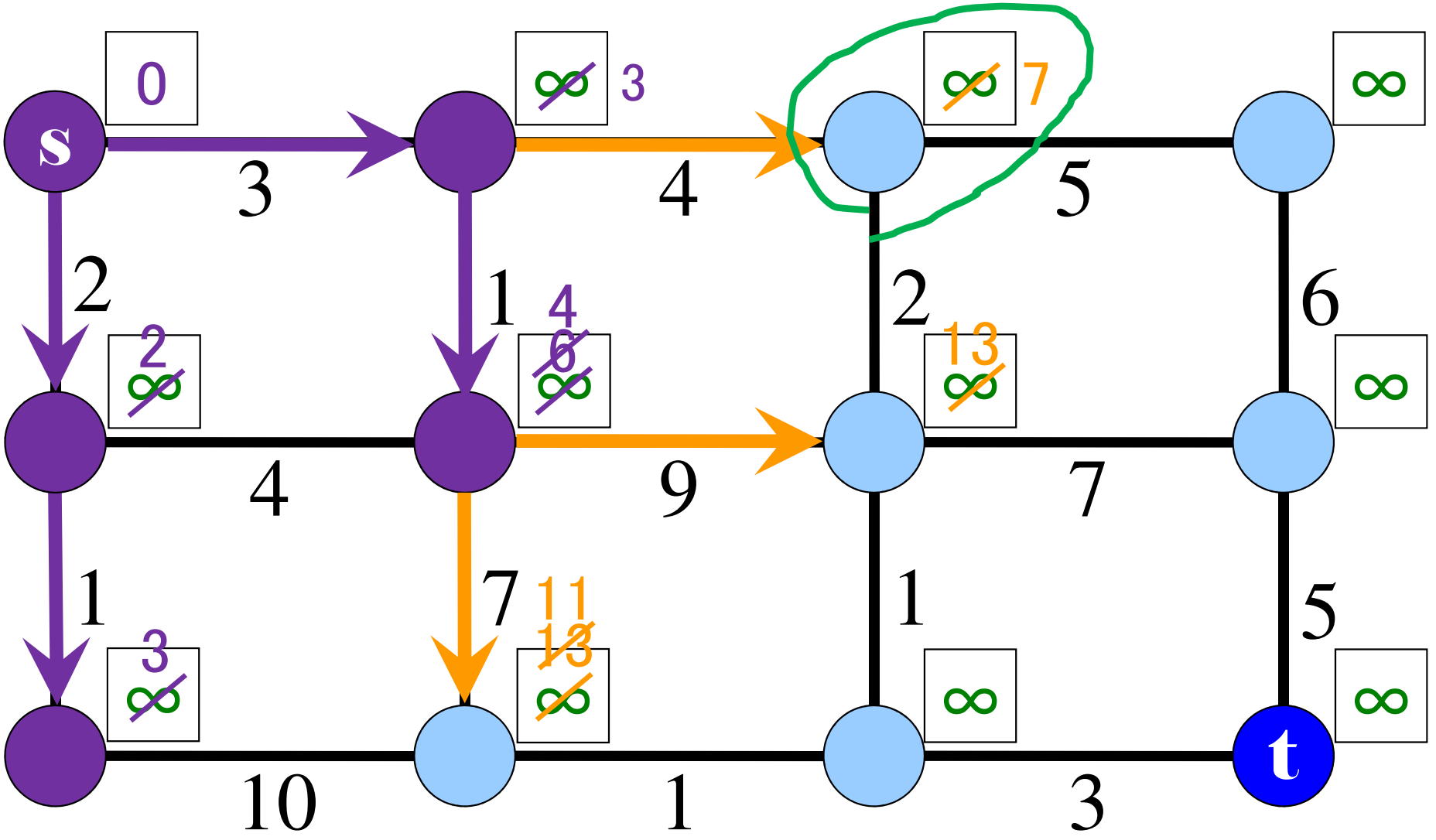
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



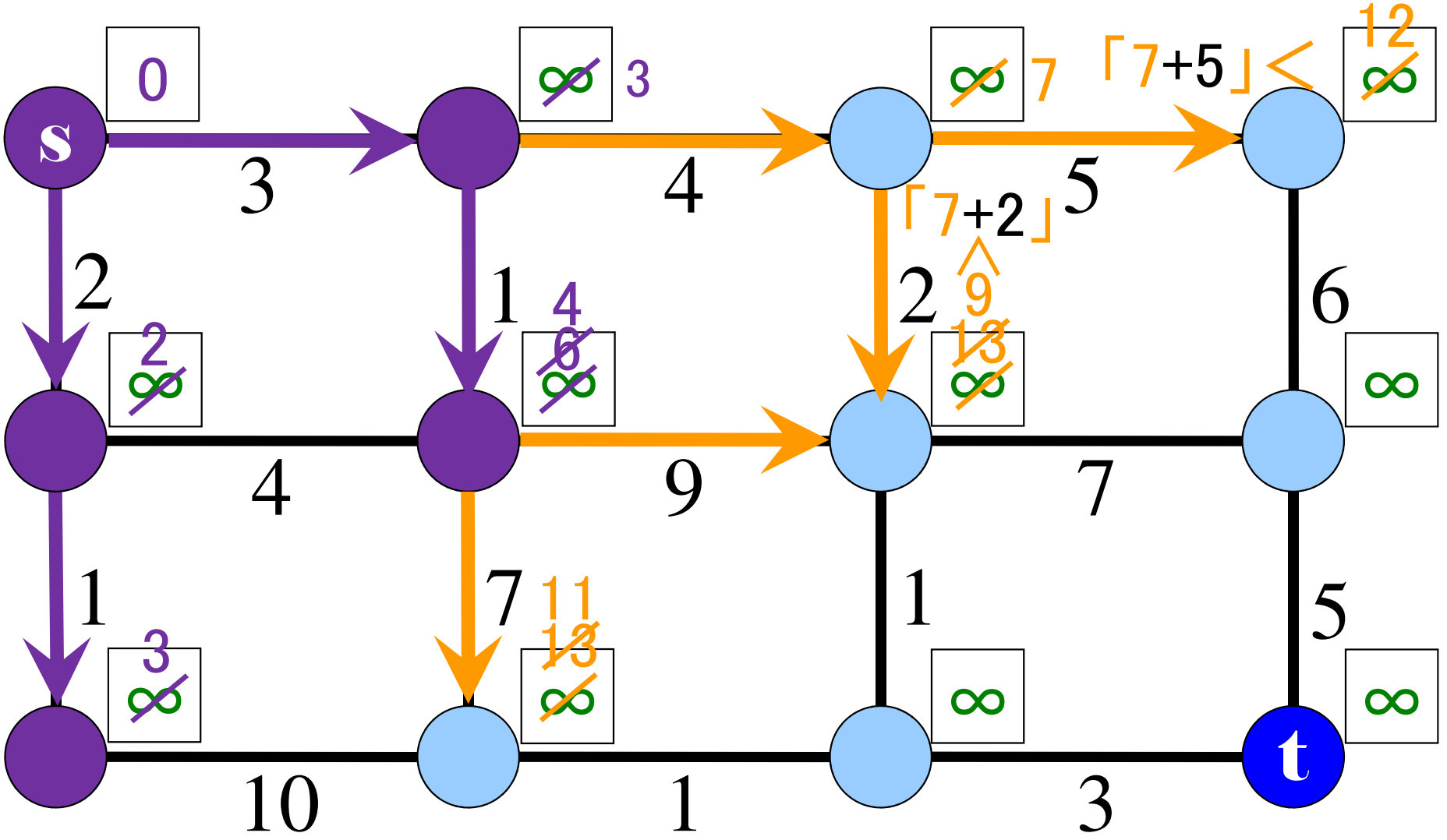
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

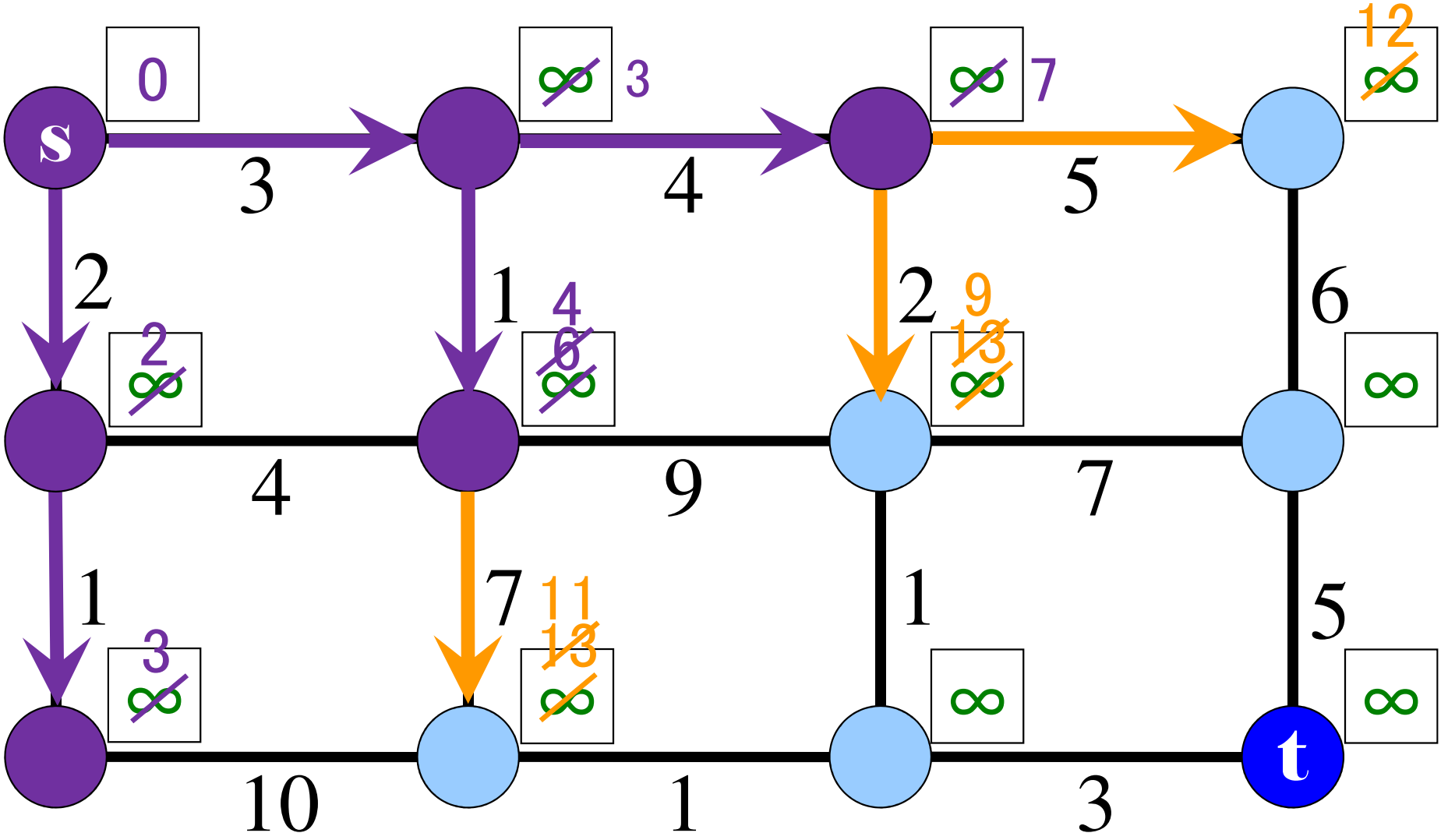


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

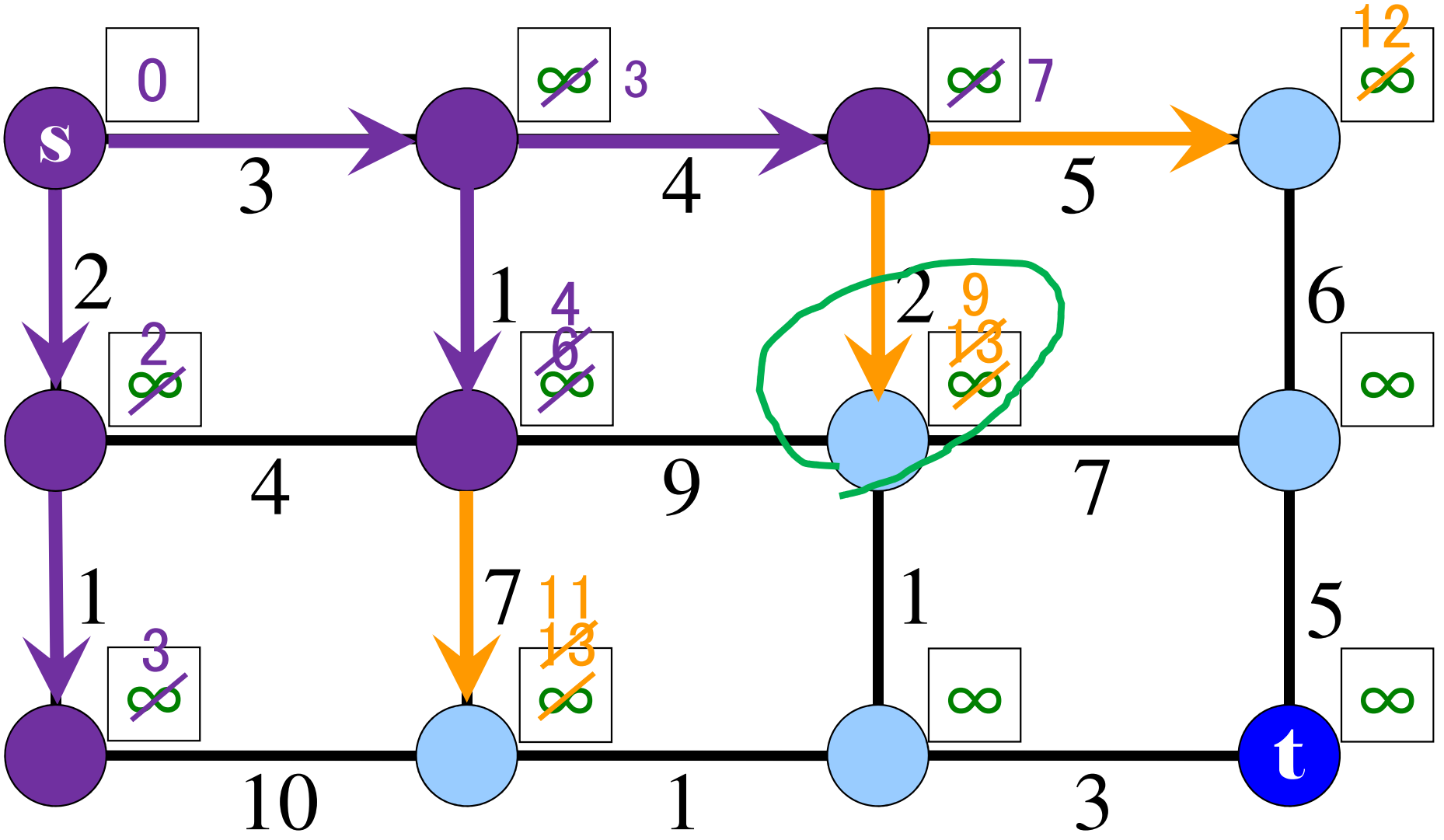
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



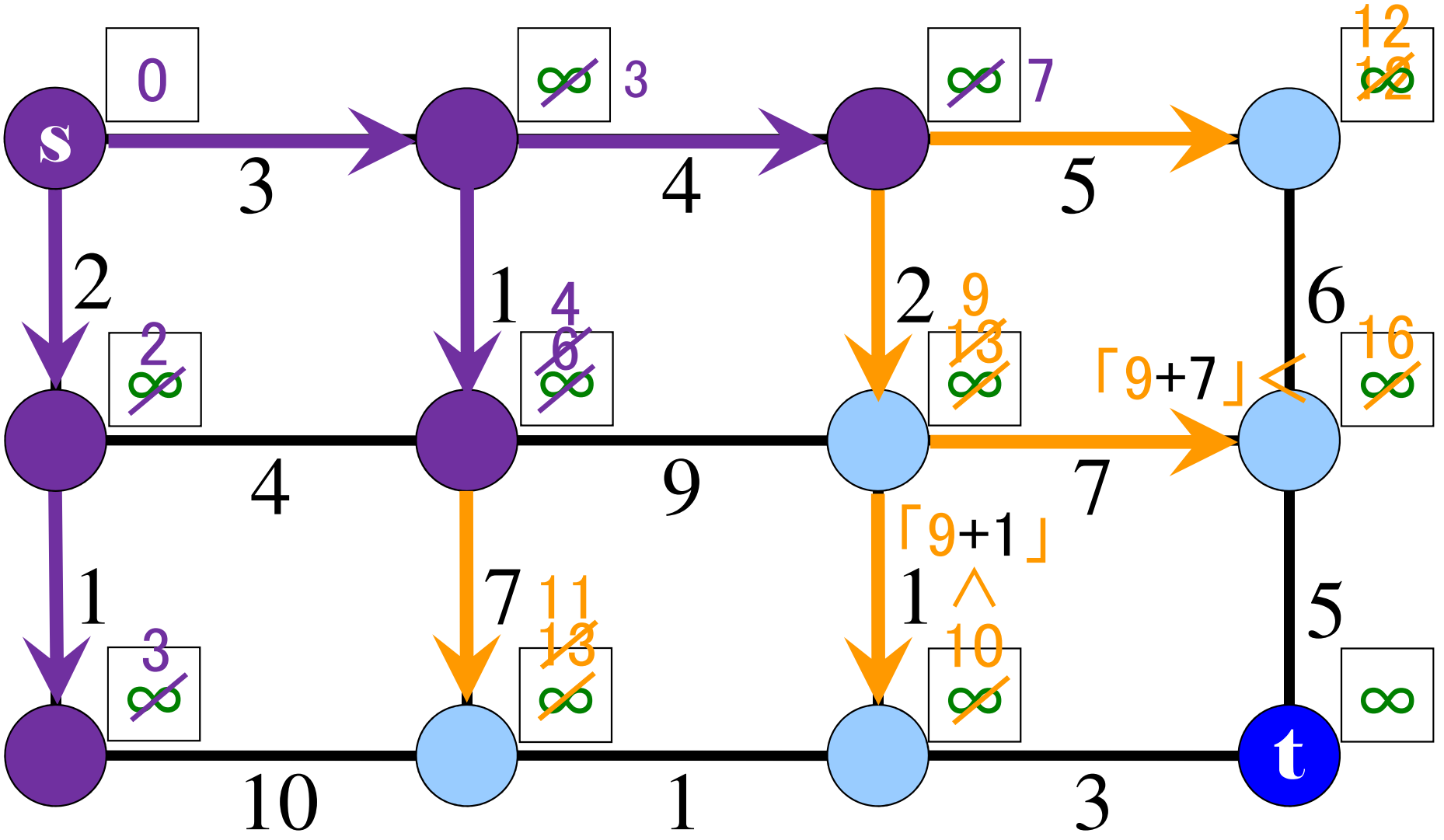
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

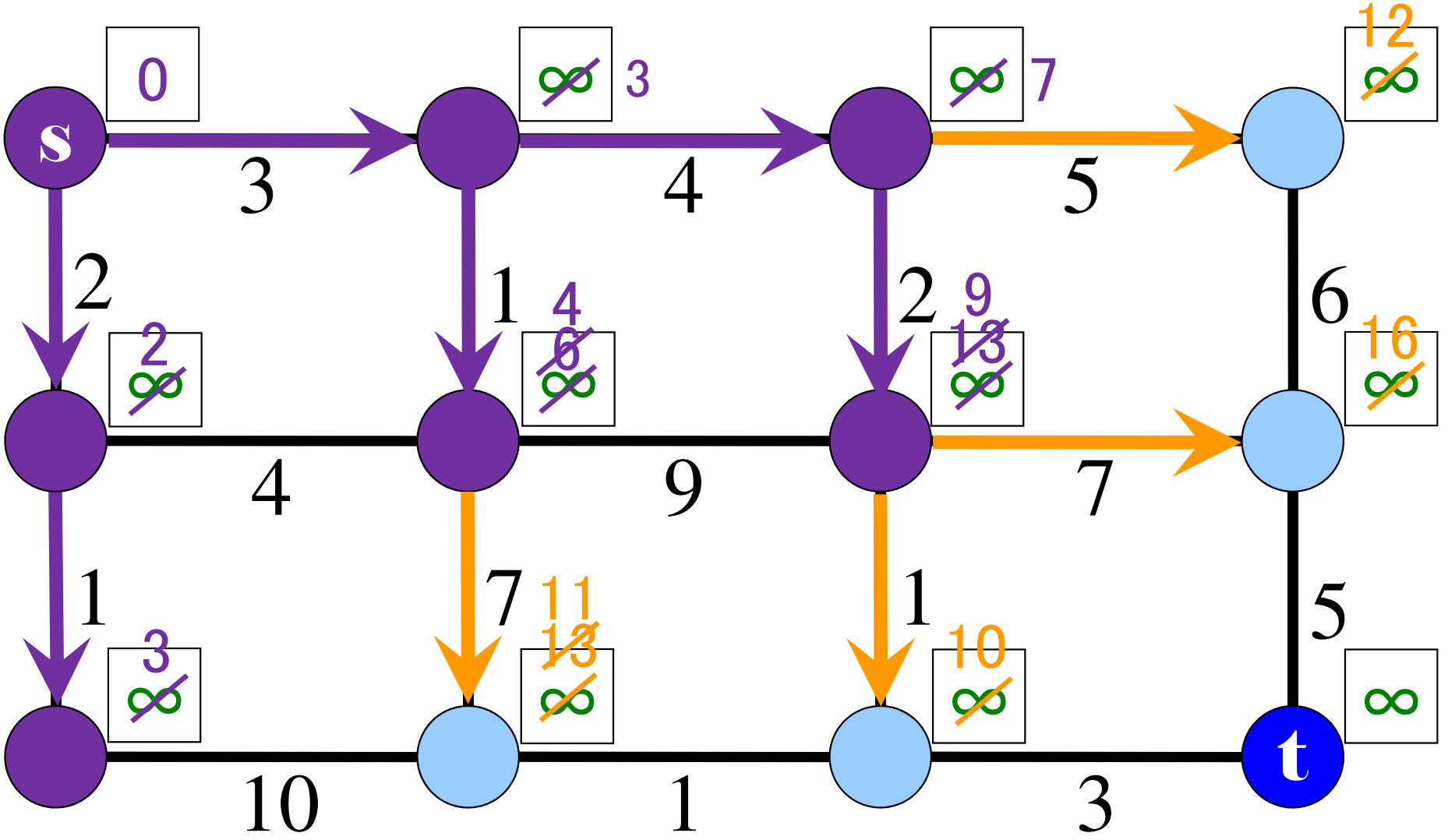


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

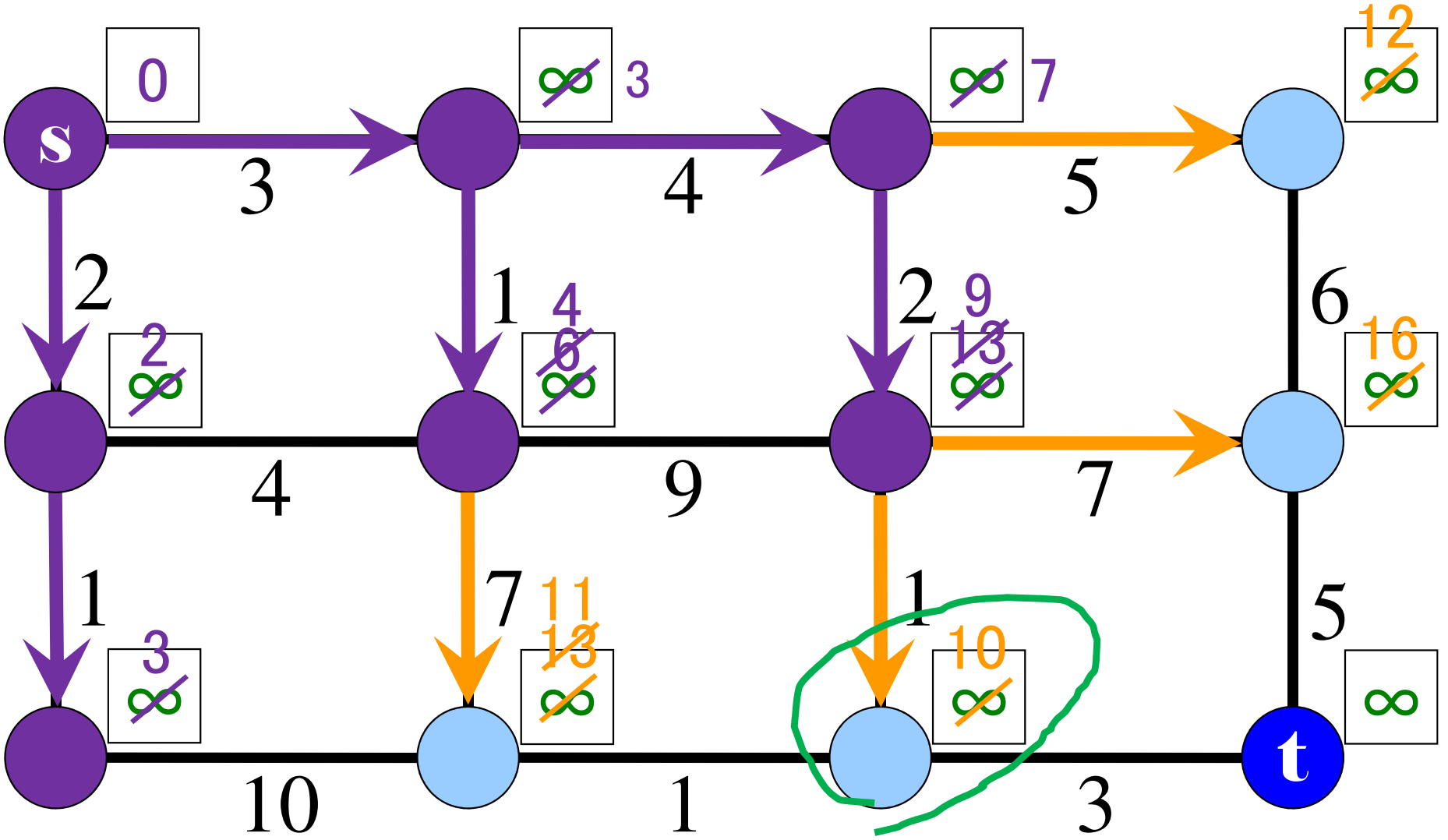
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



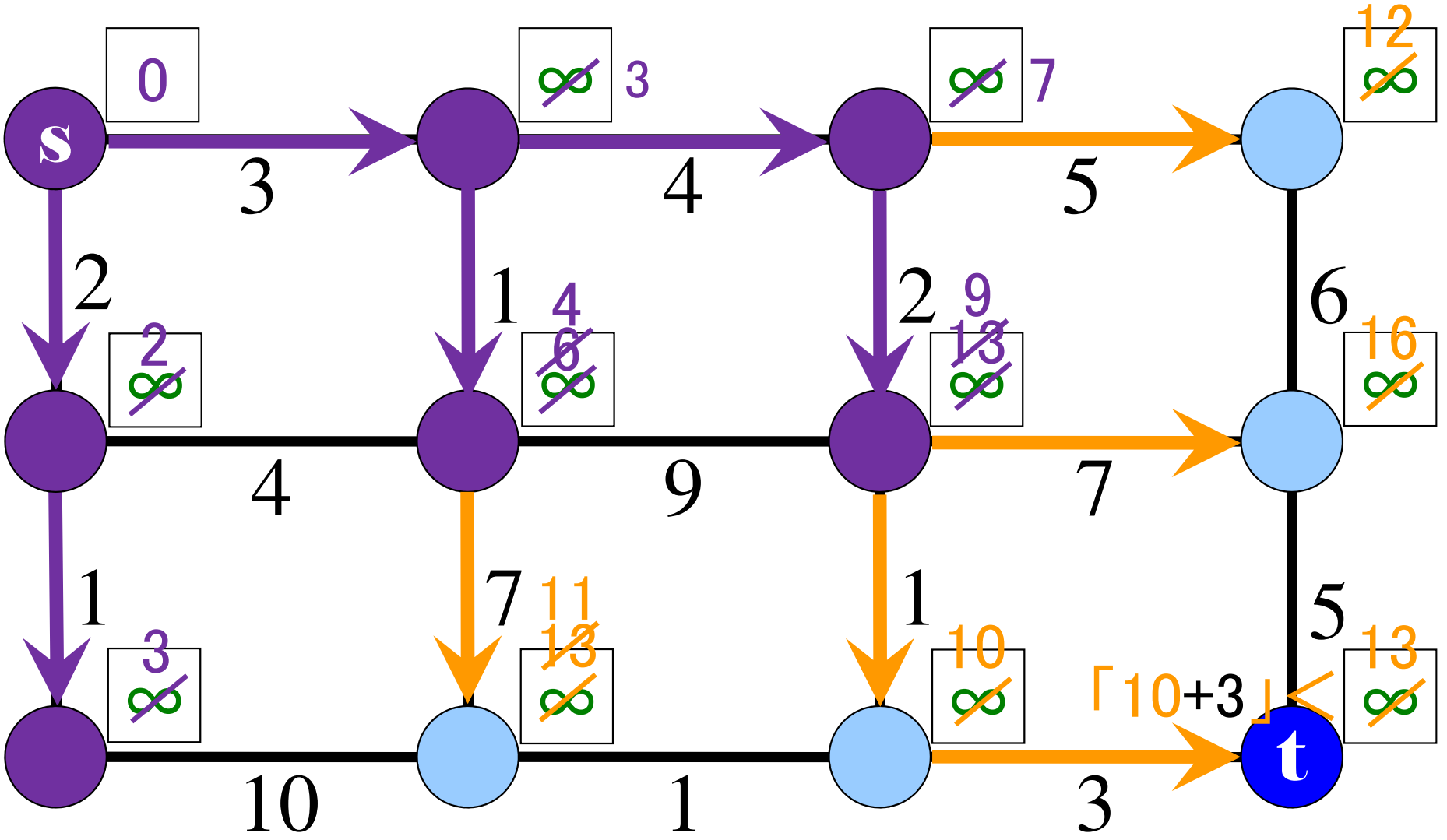
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

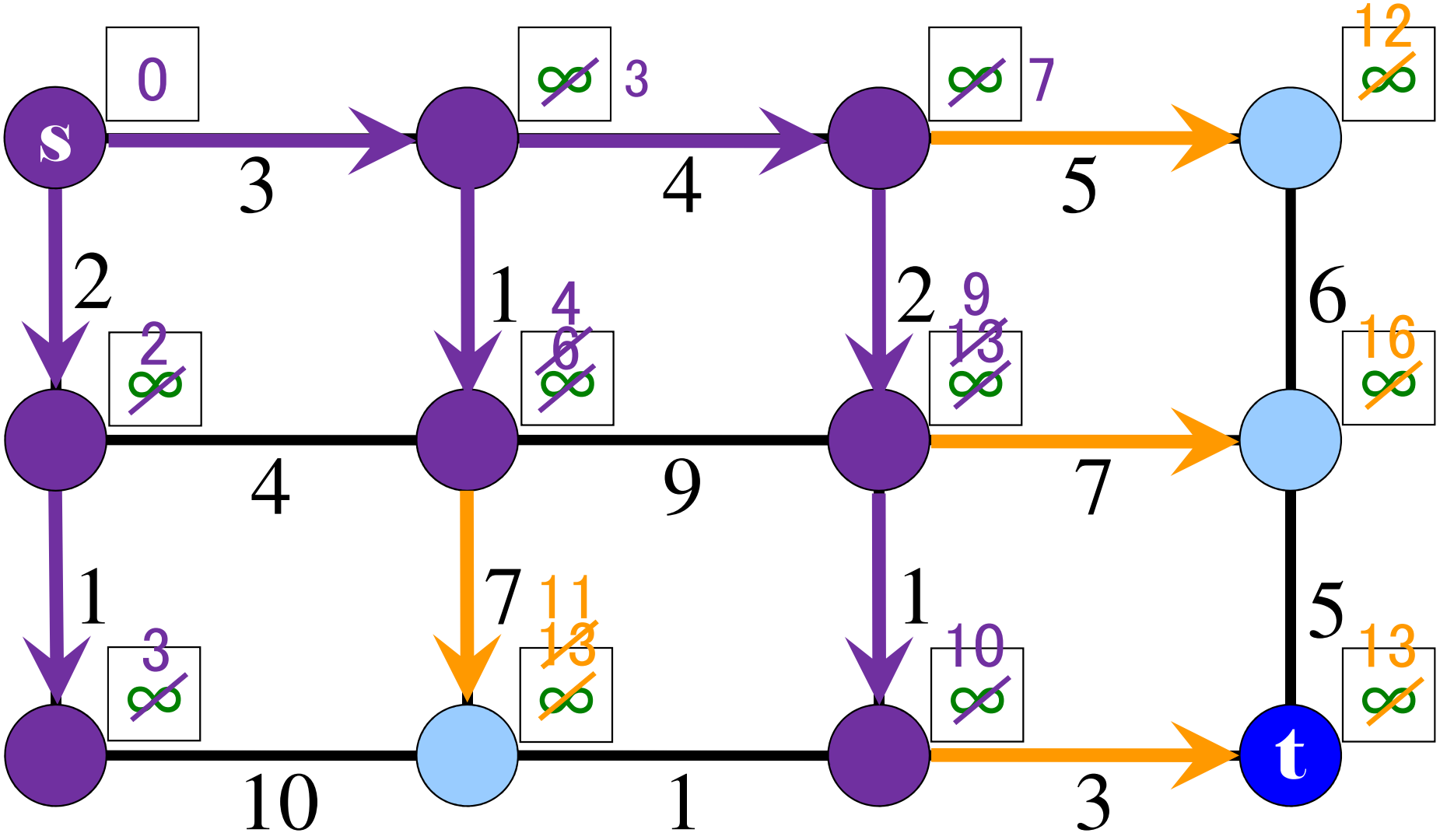


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

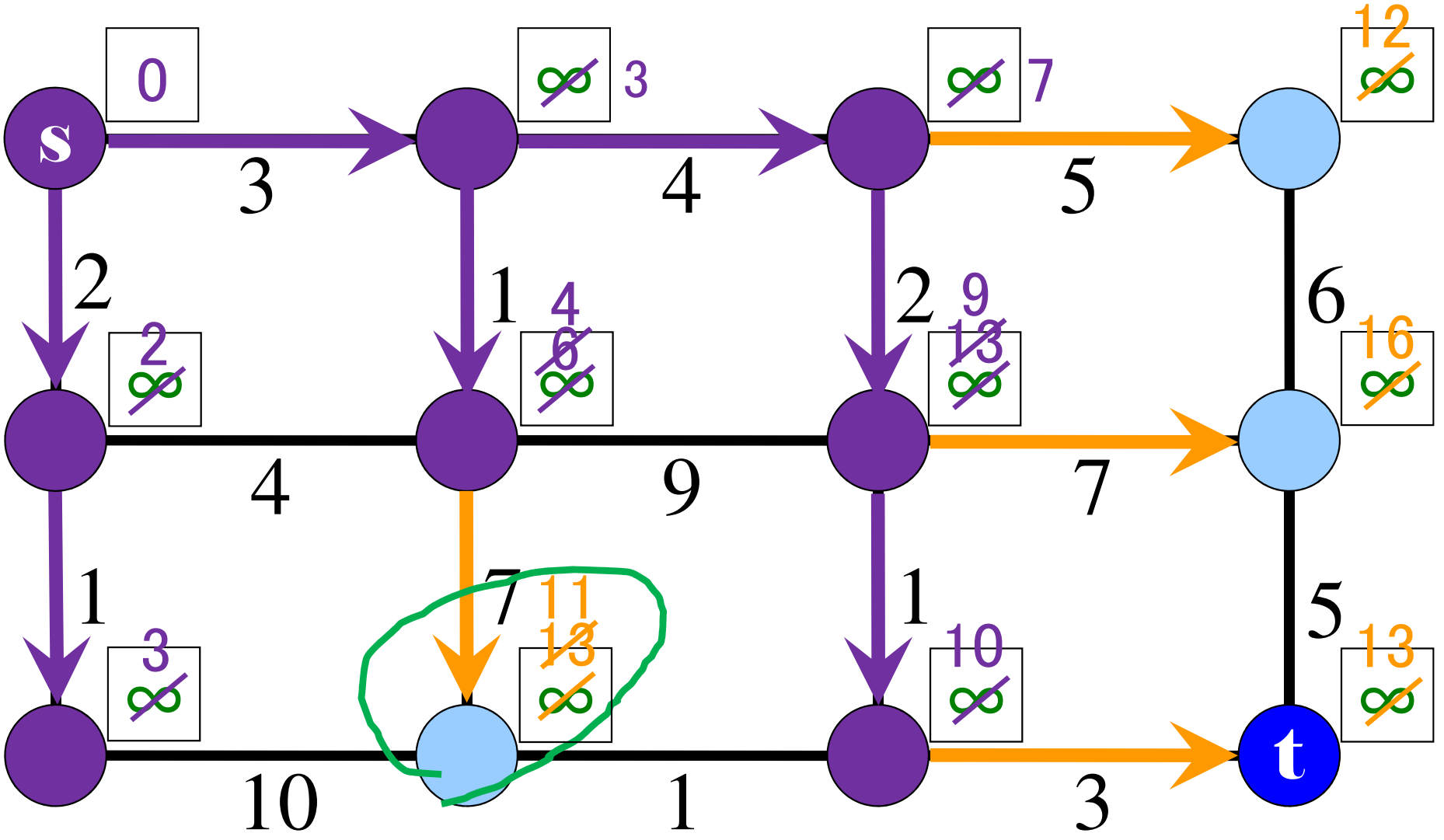
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



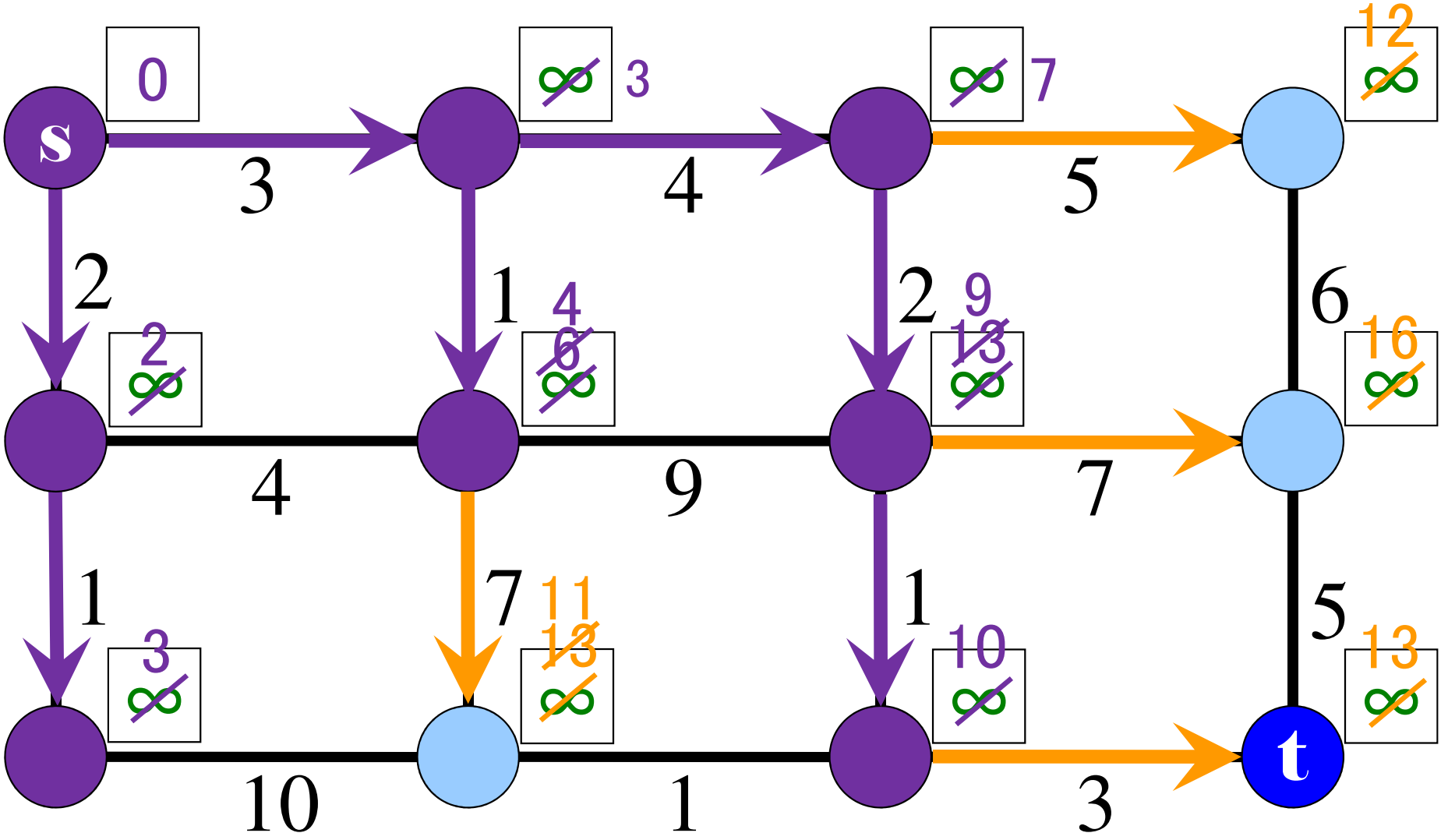
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

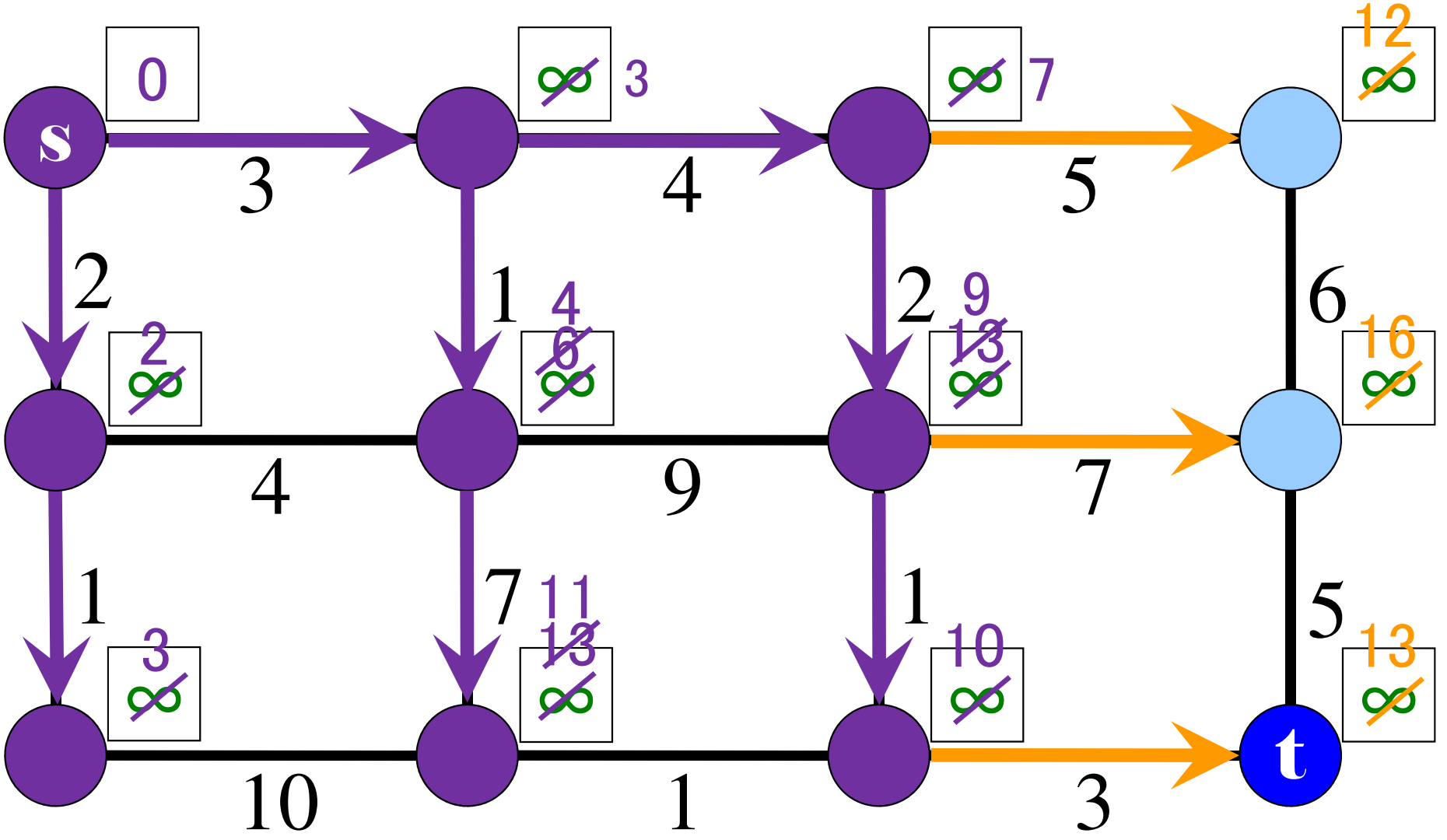


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

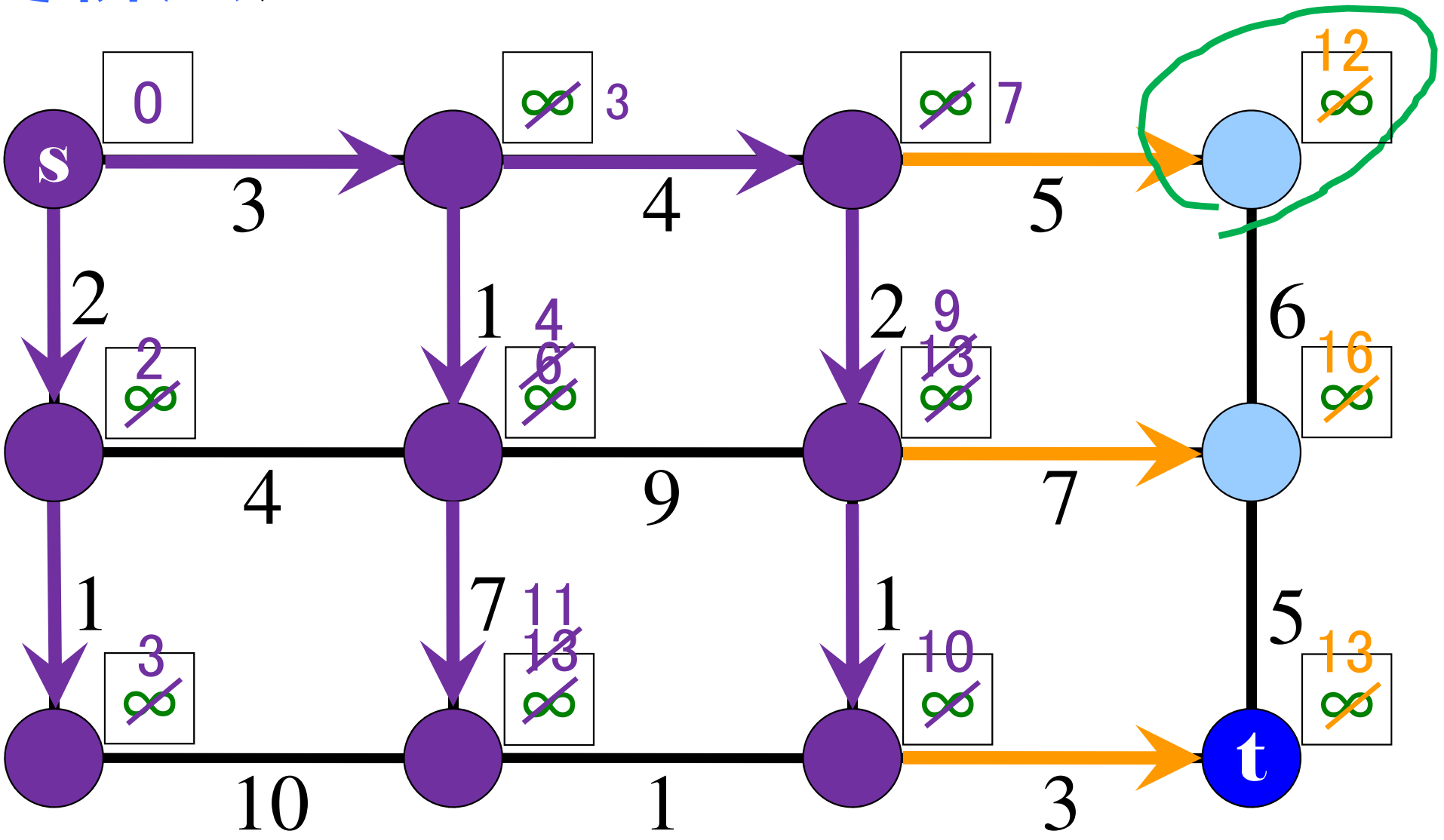
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



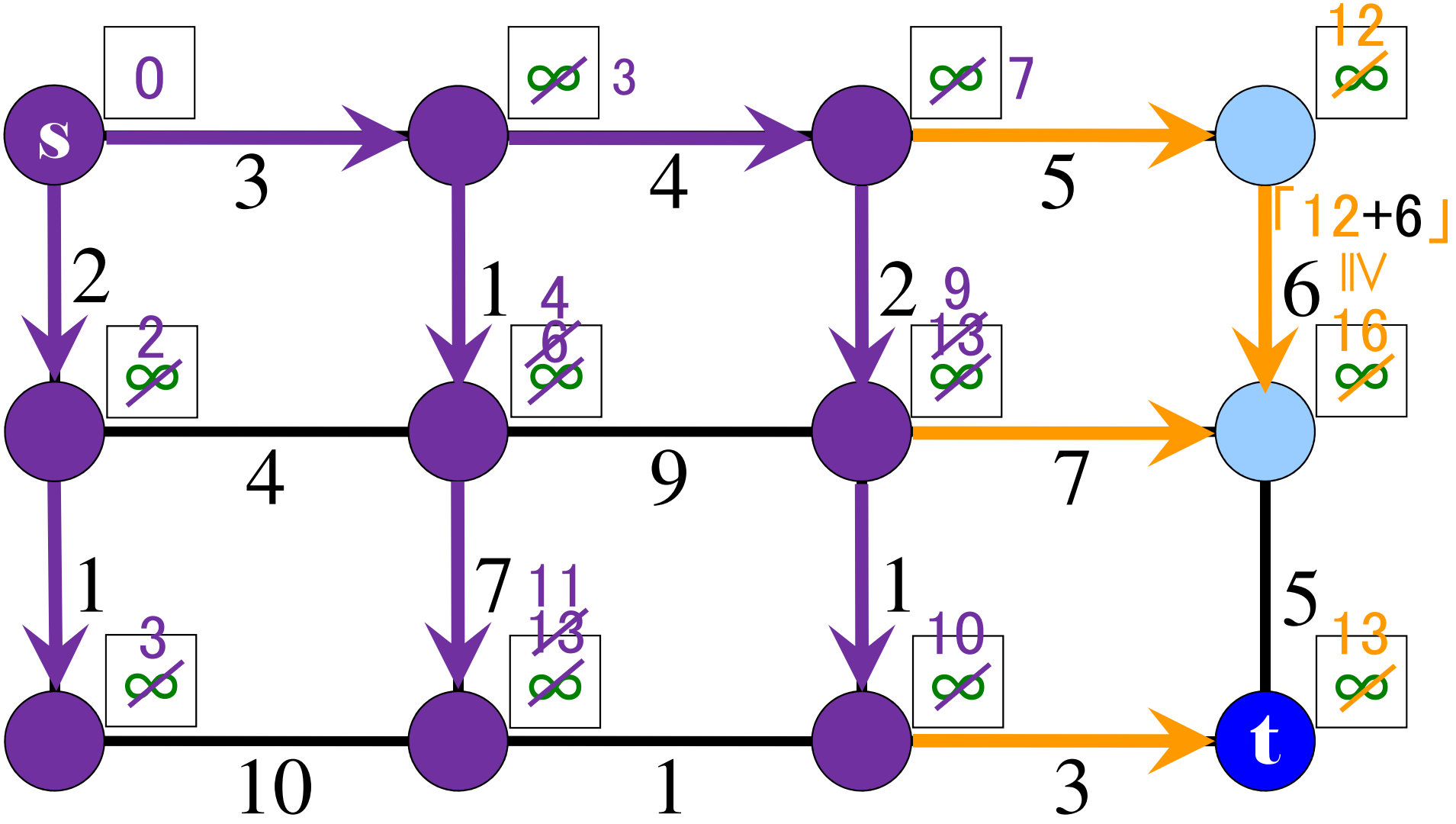
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

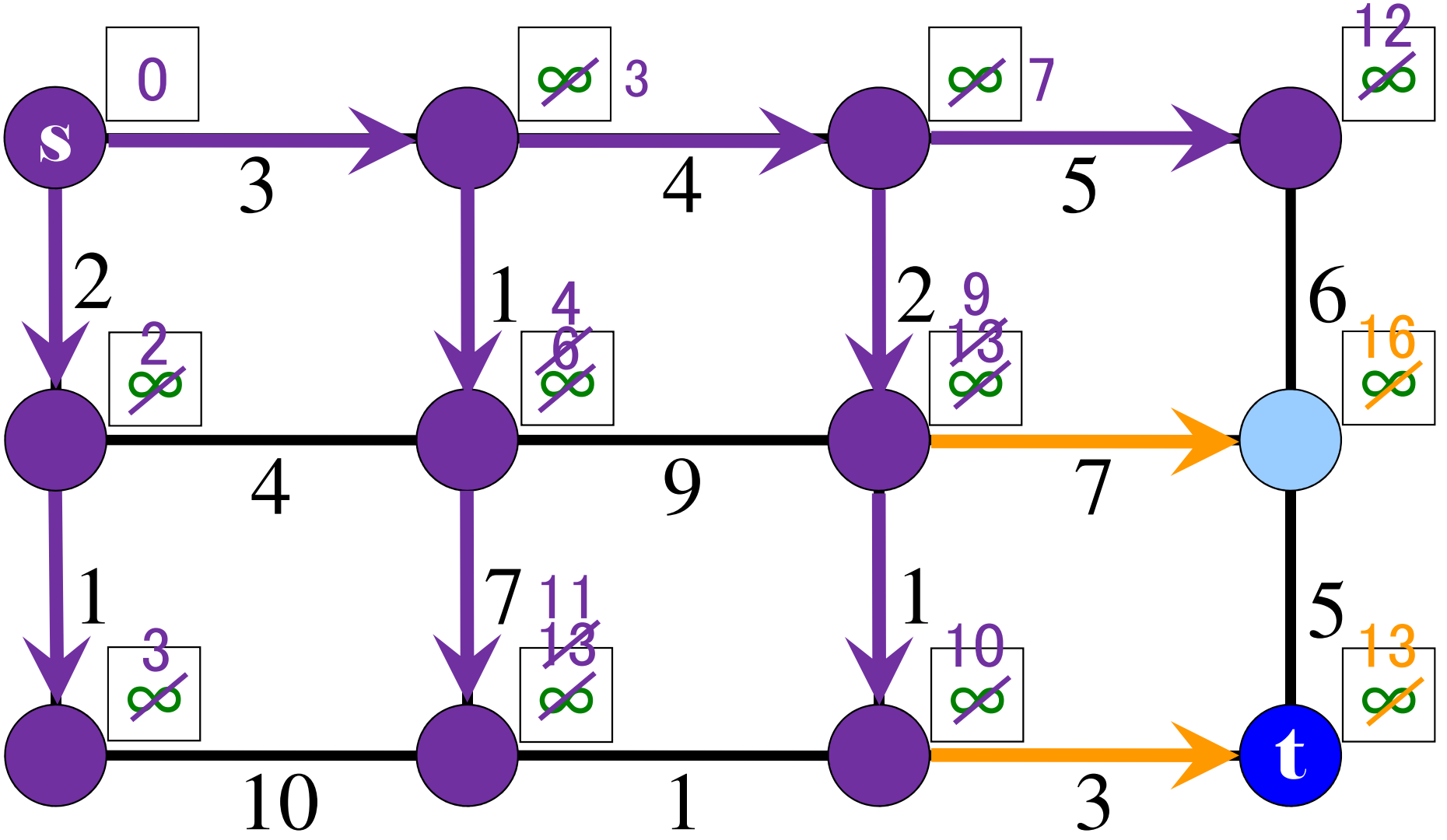


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

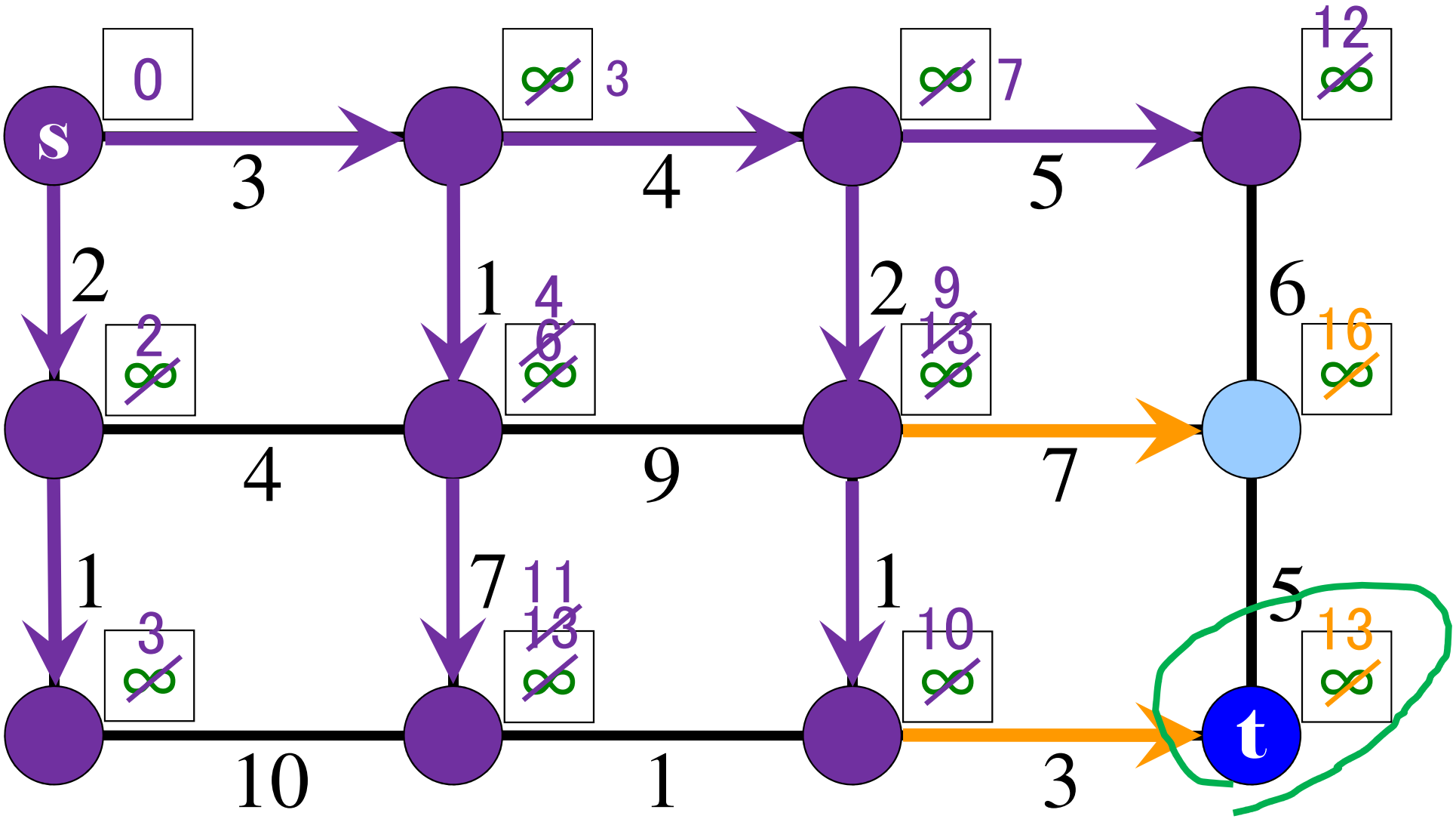
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



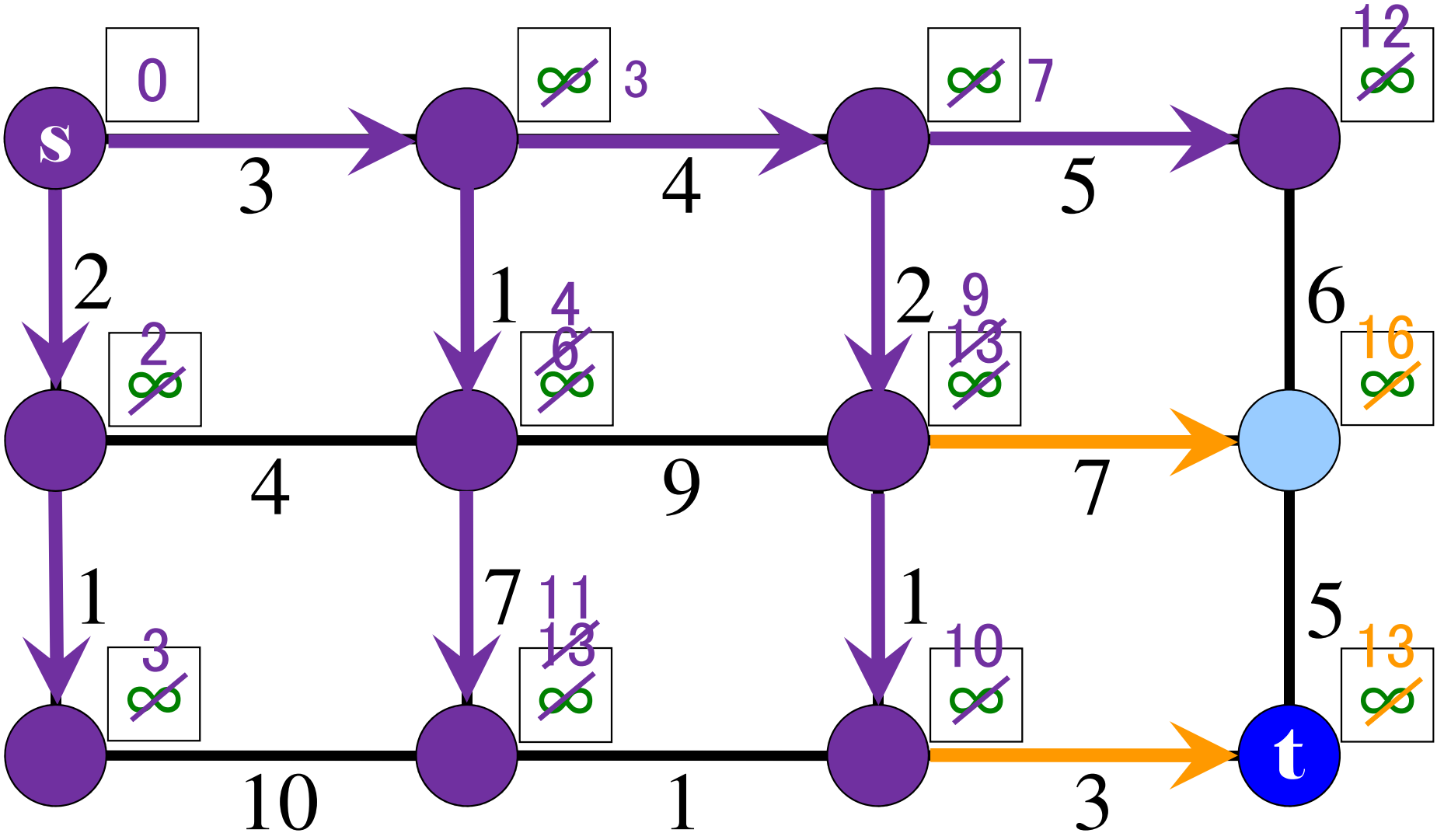
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

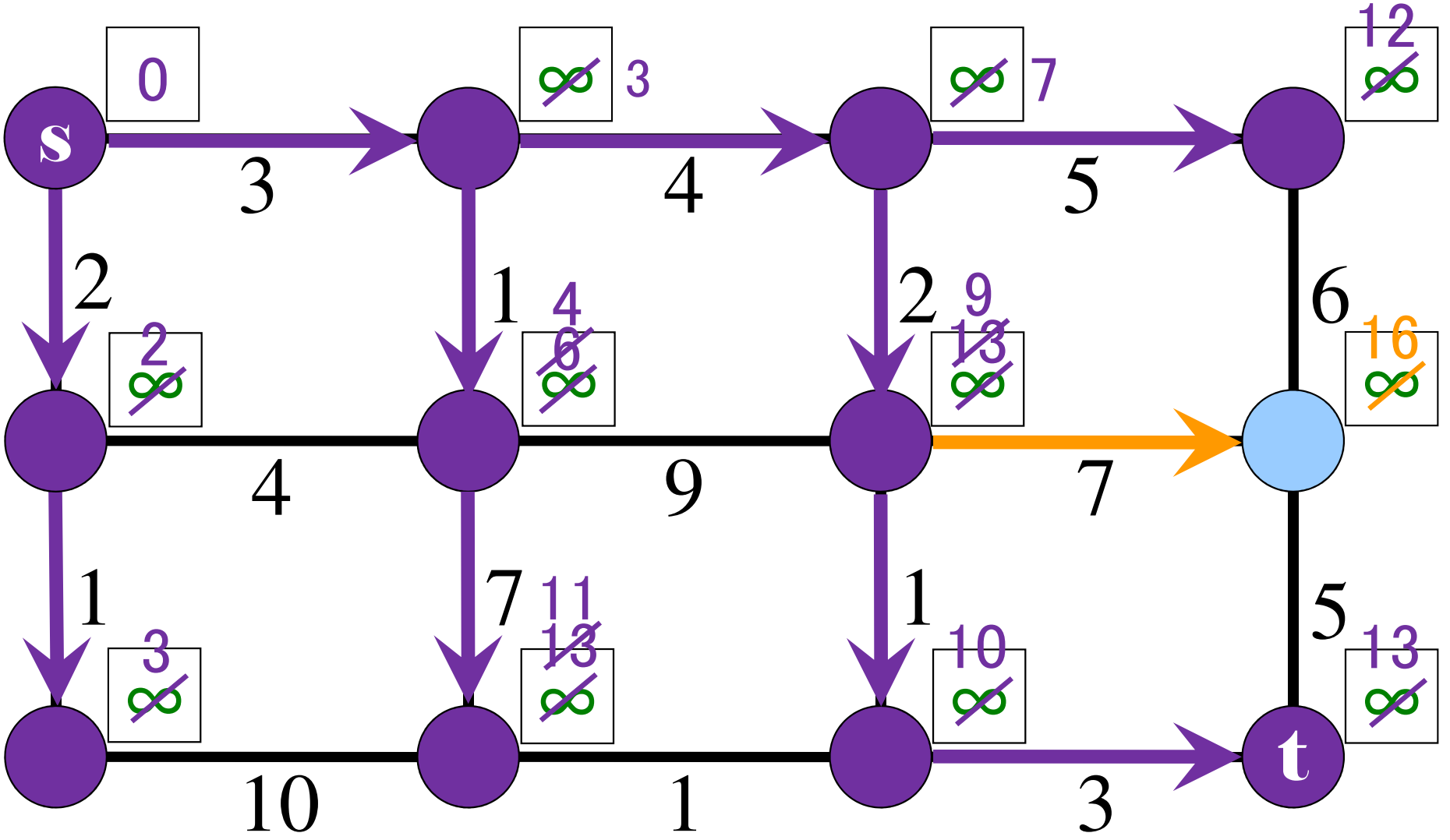


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

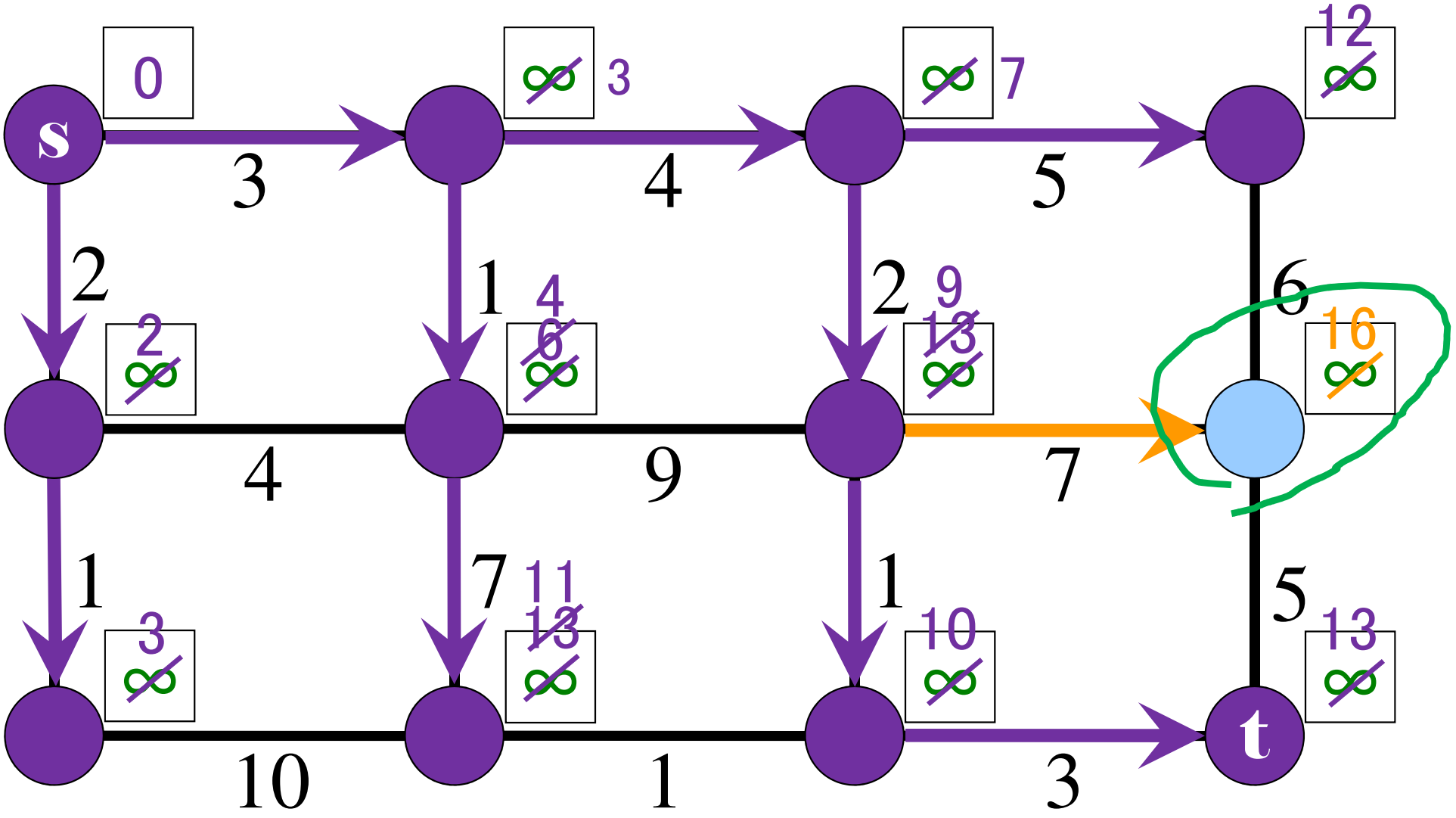
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



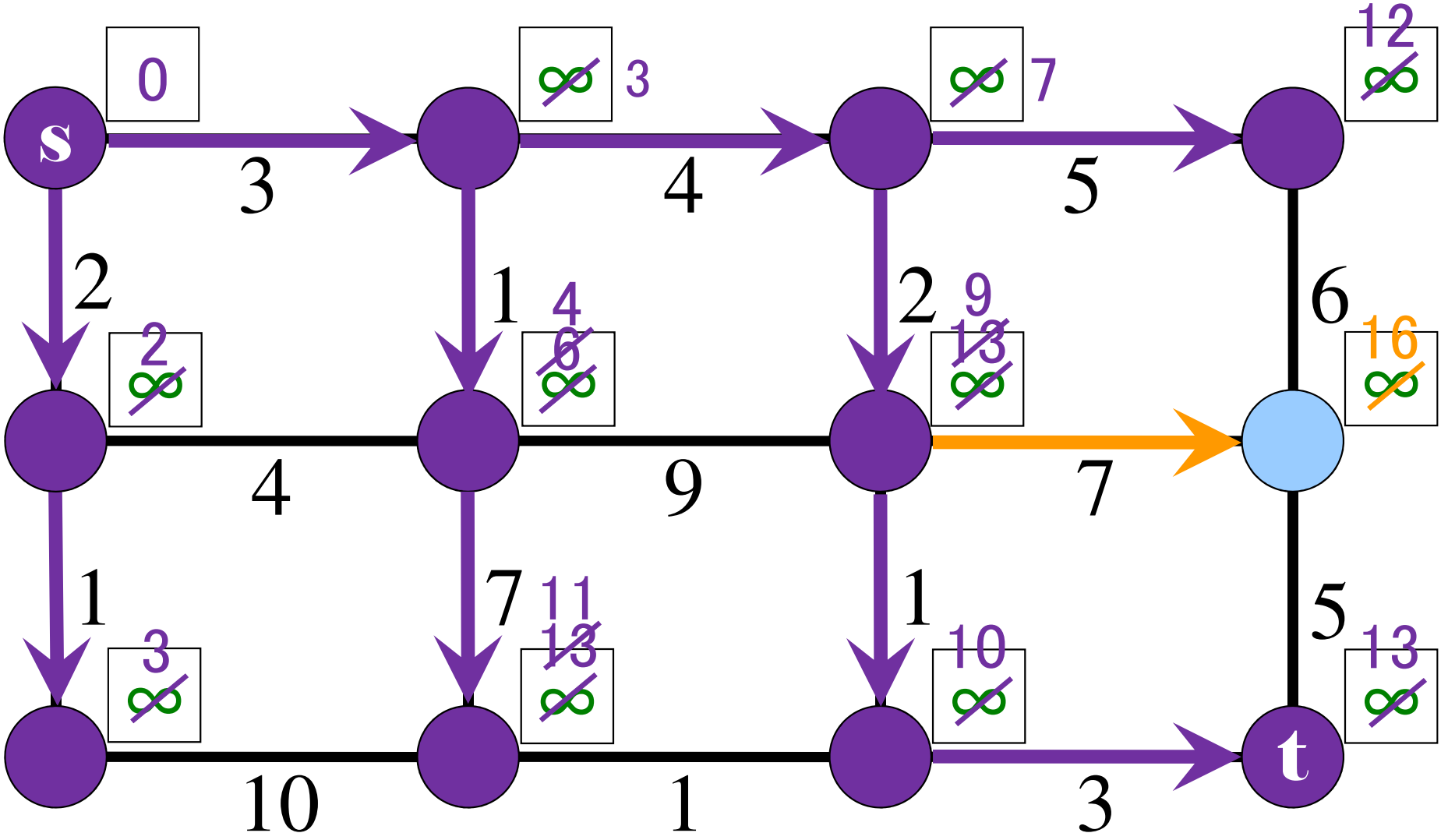
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

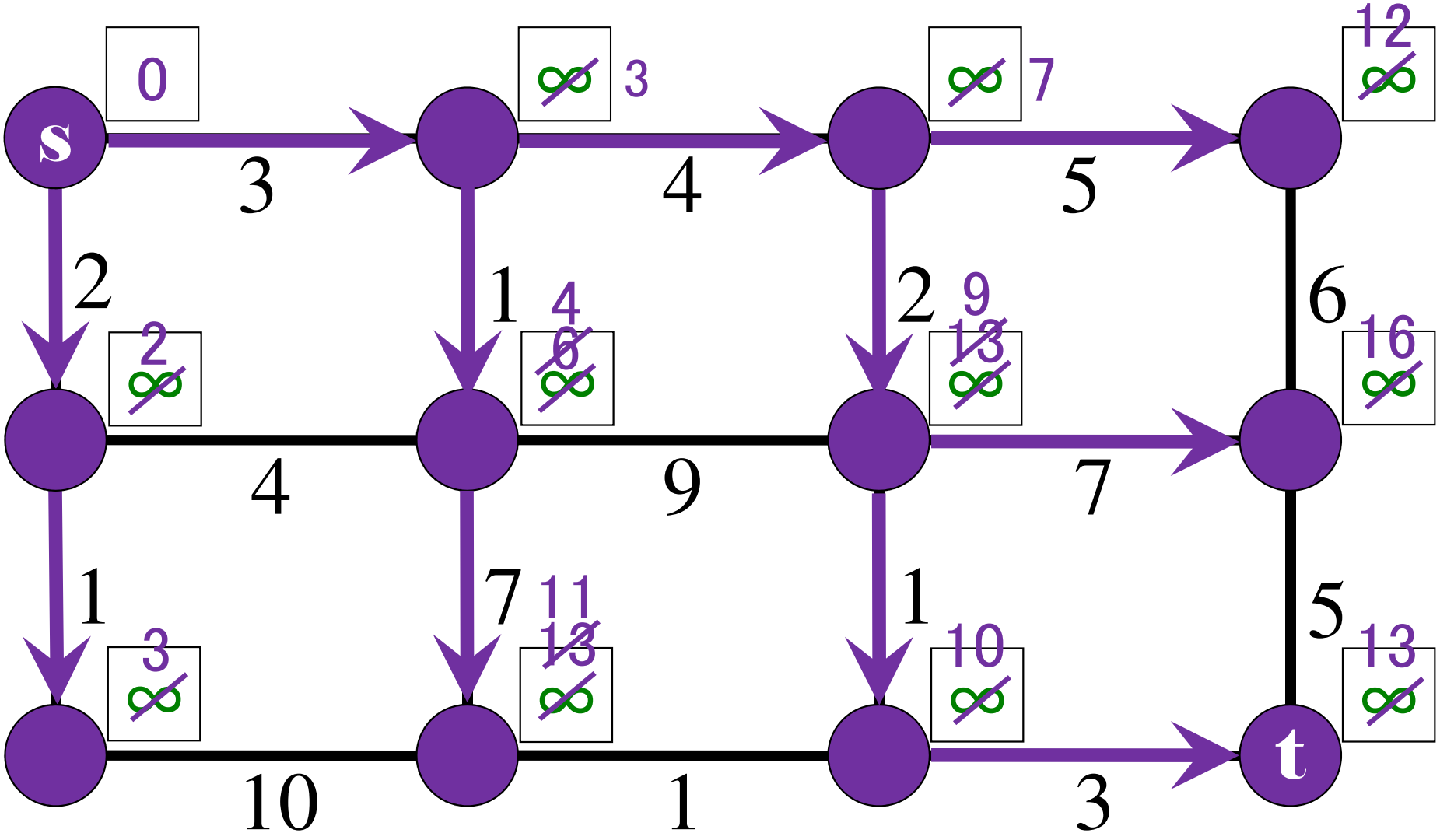


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

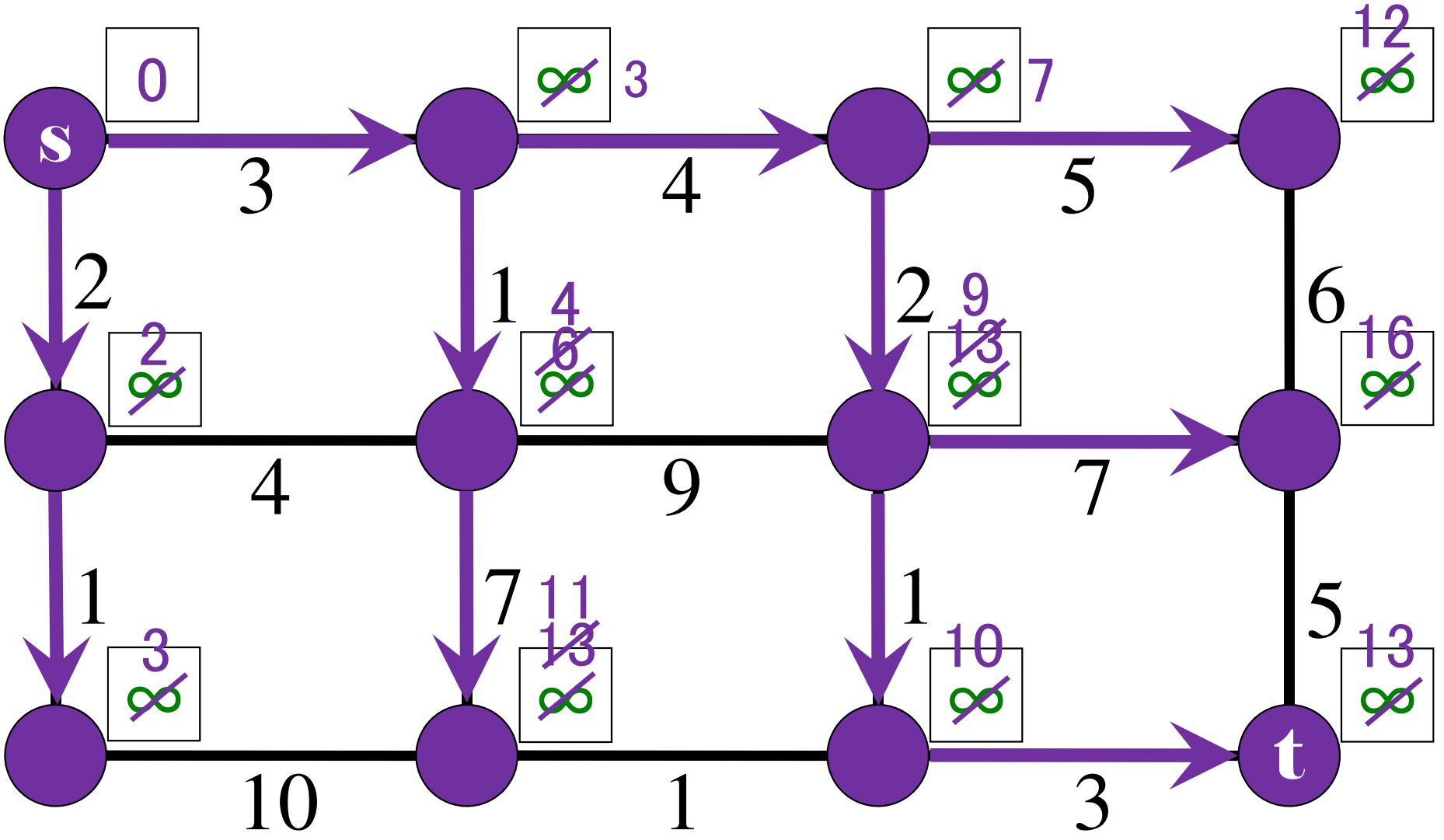
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



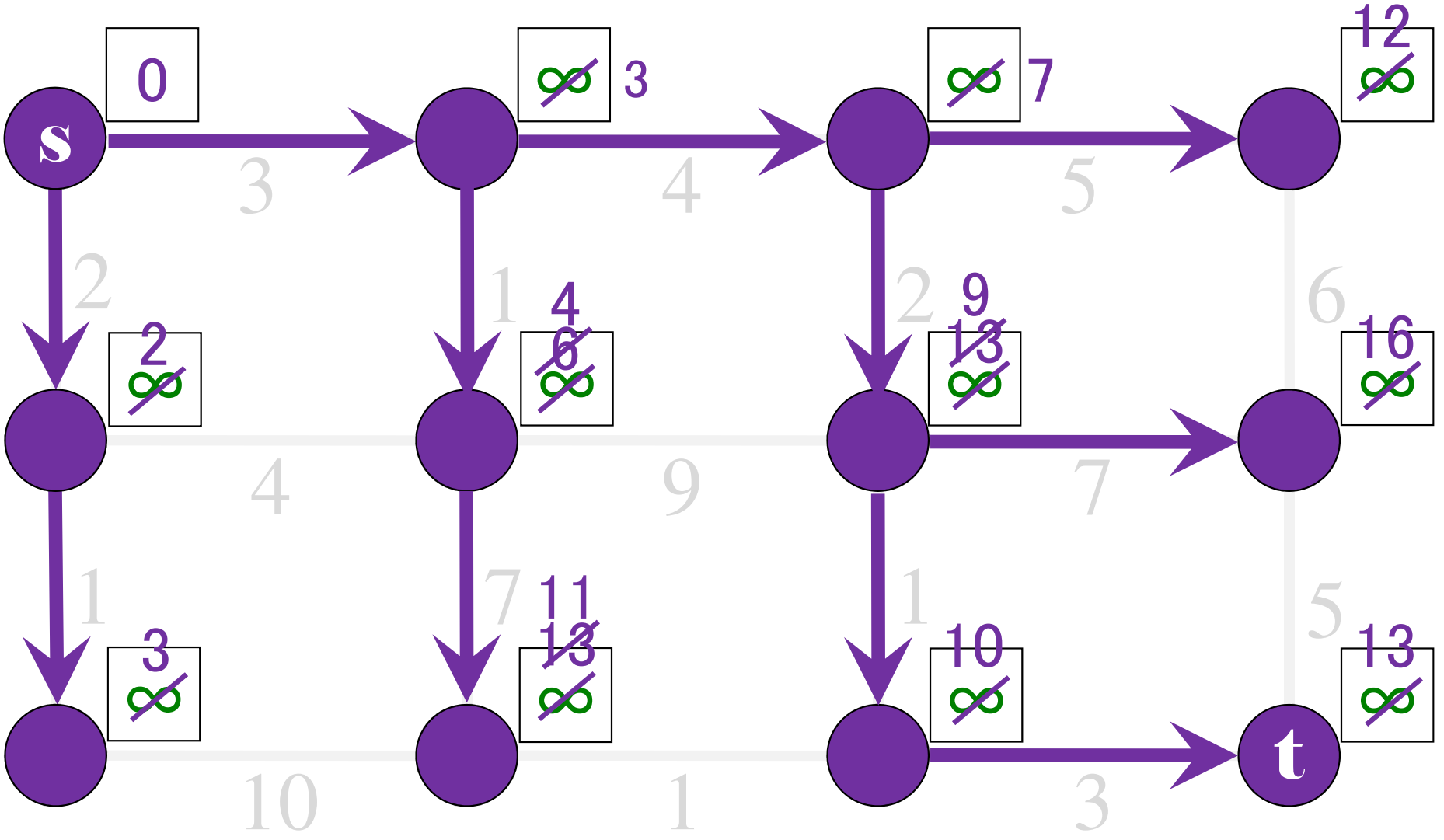
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



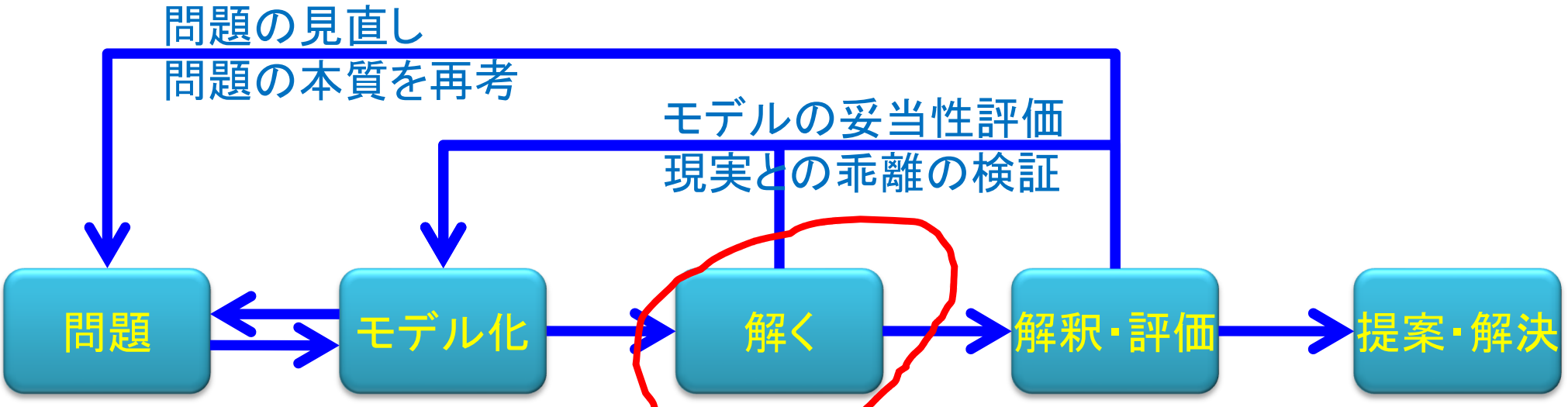
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

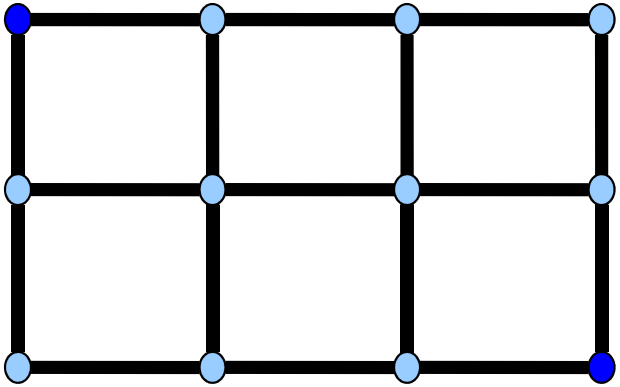
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

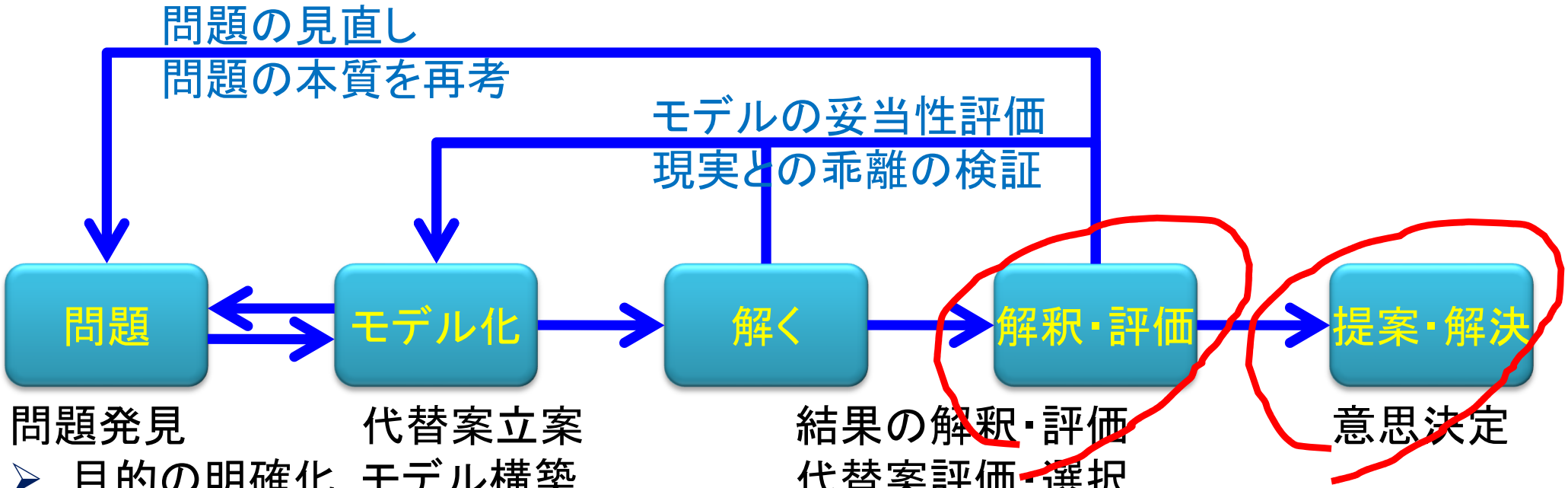
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

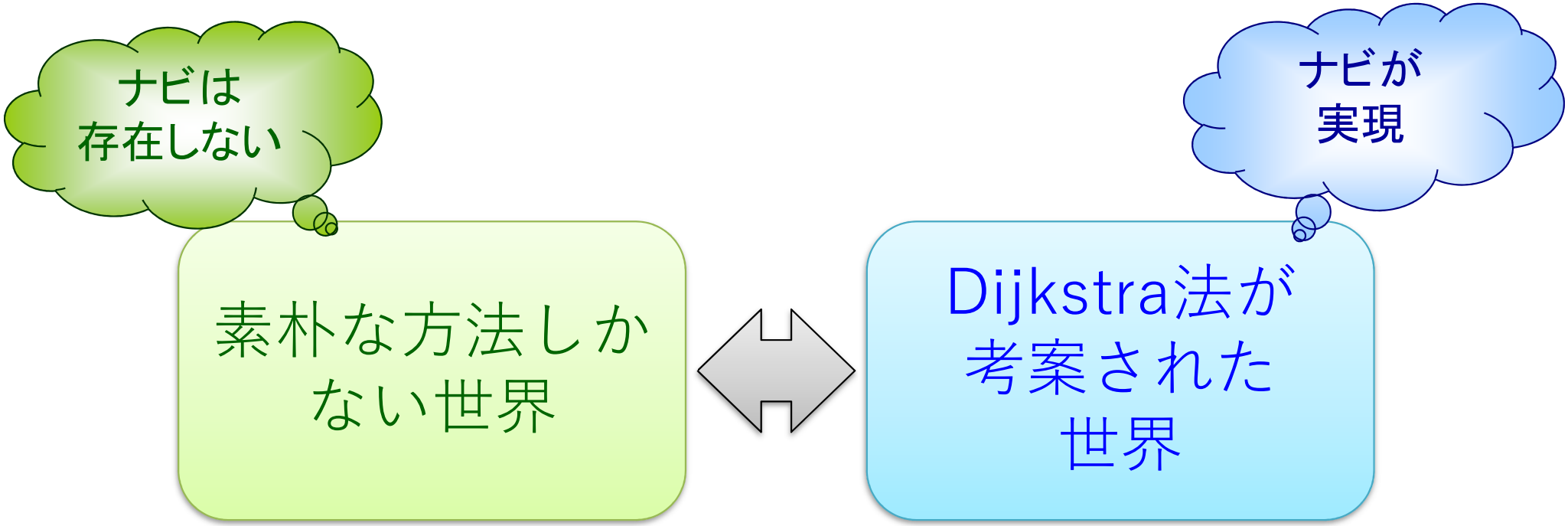
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

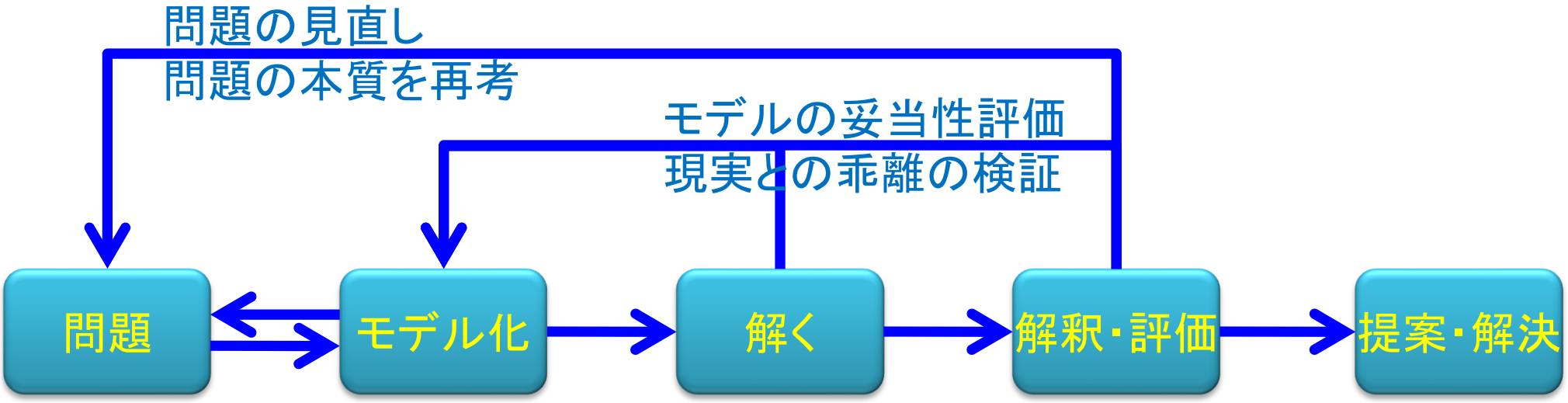
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



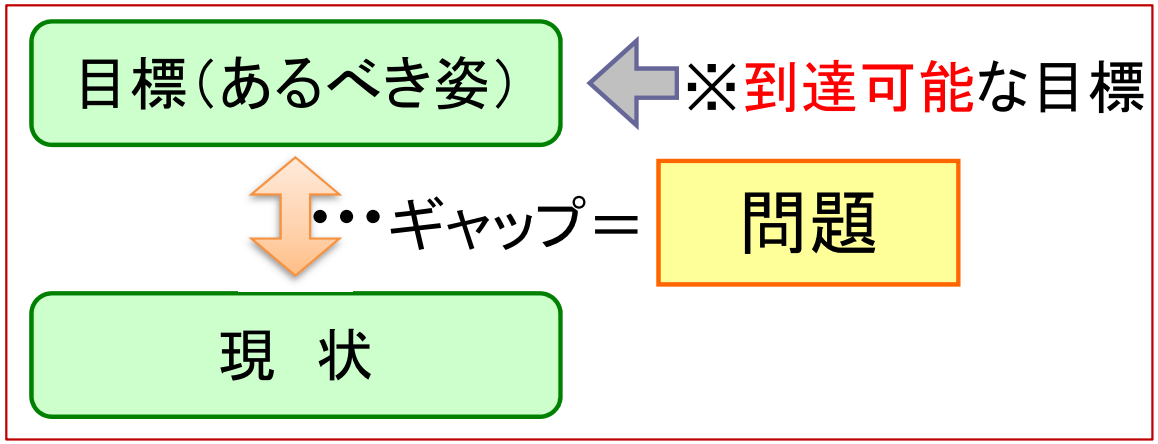
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

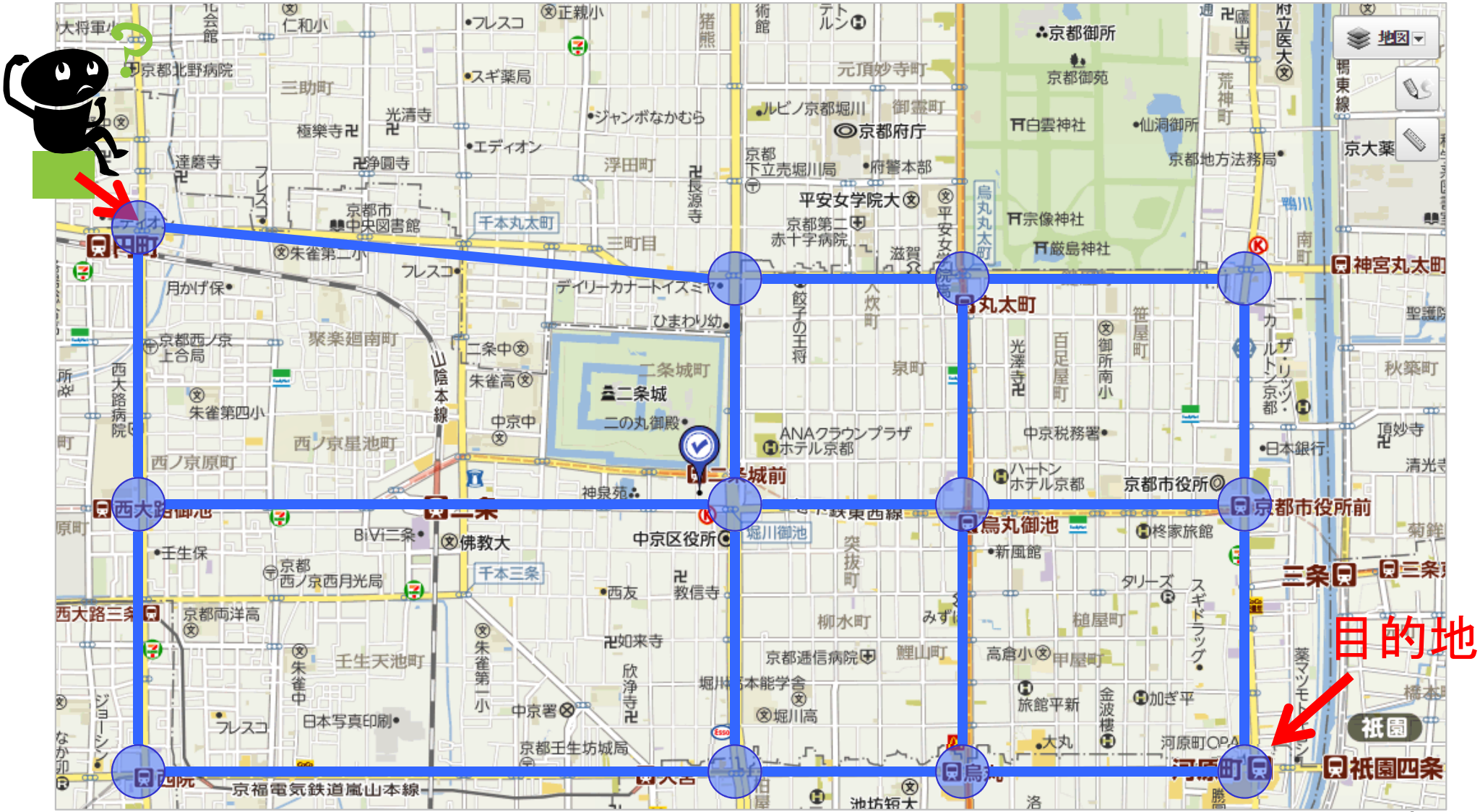
意思決定

問題の定義



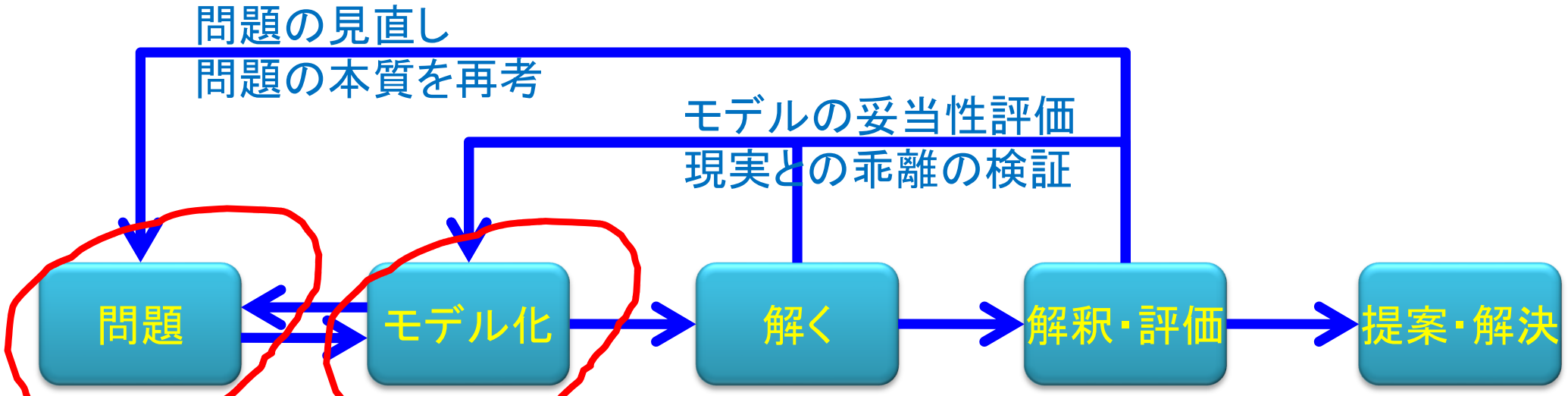
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

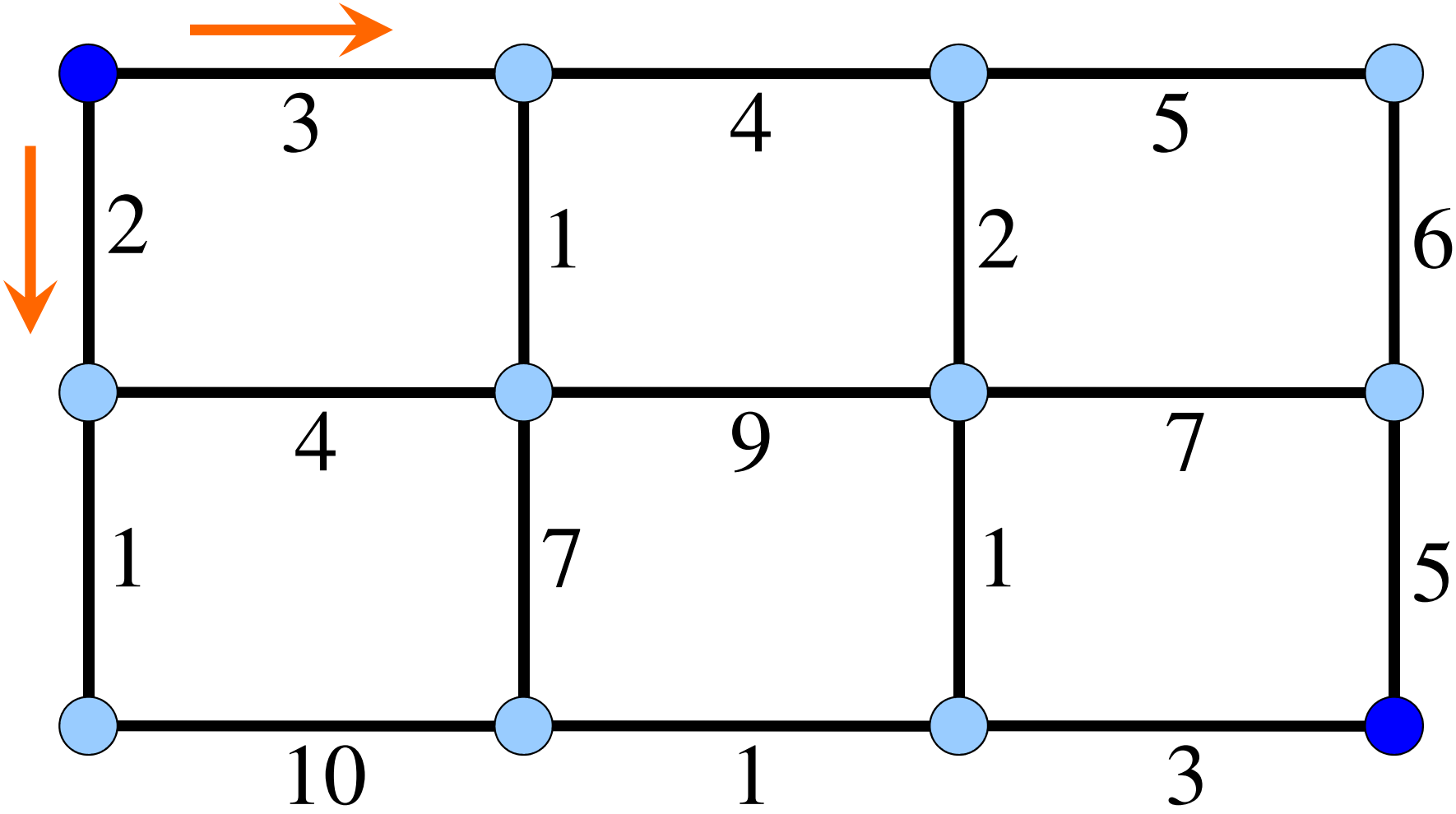
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

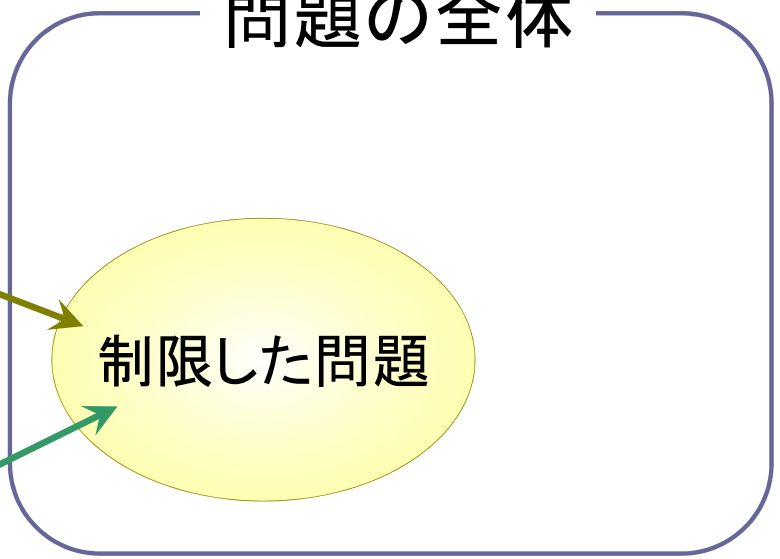


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

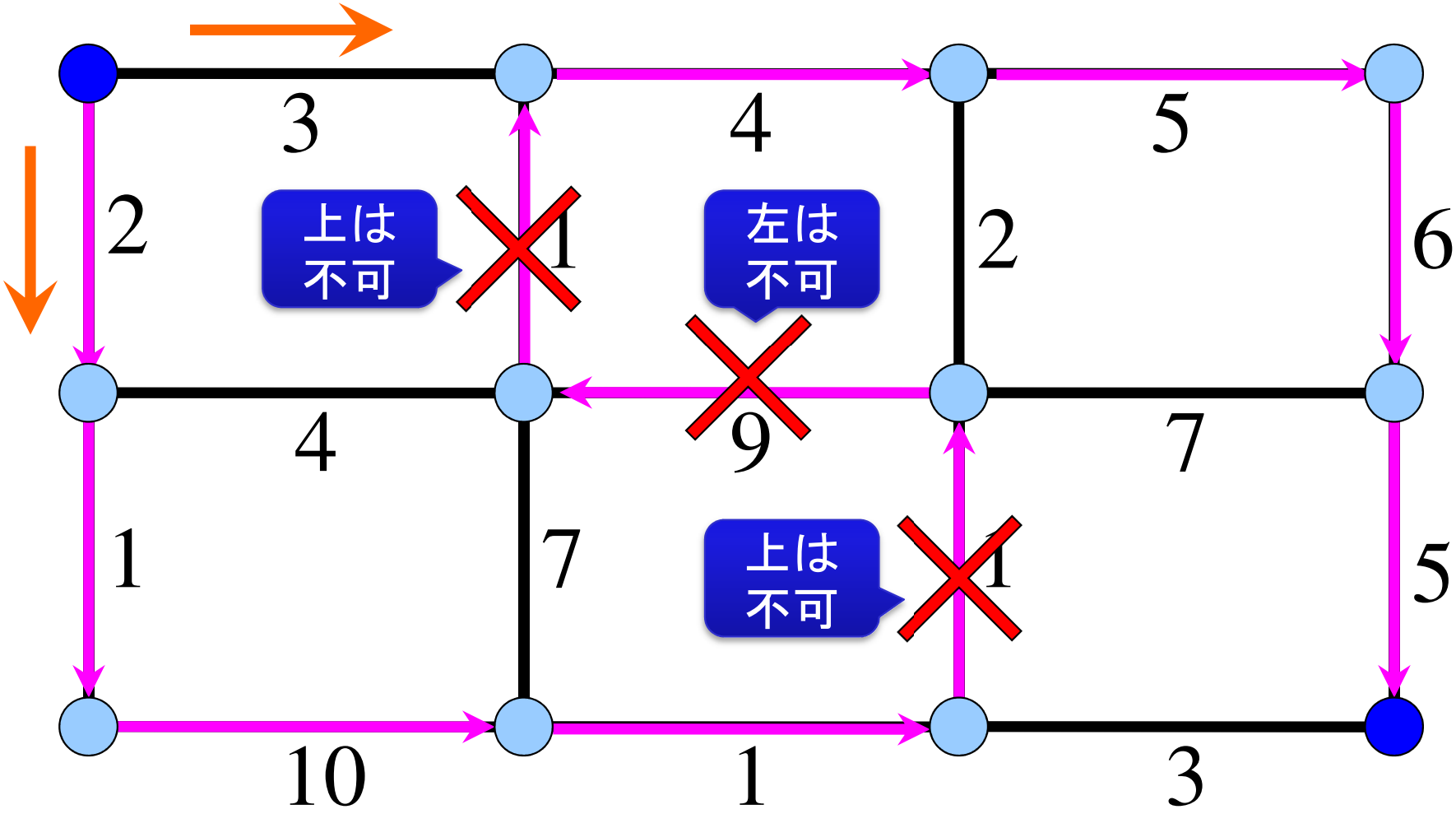
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

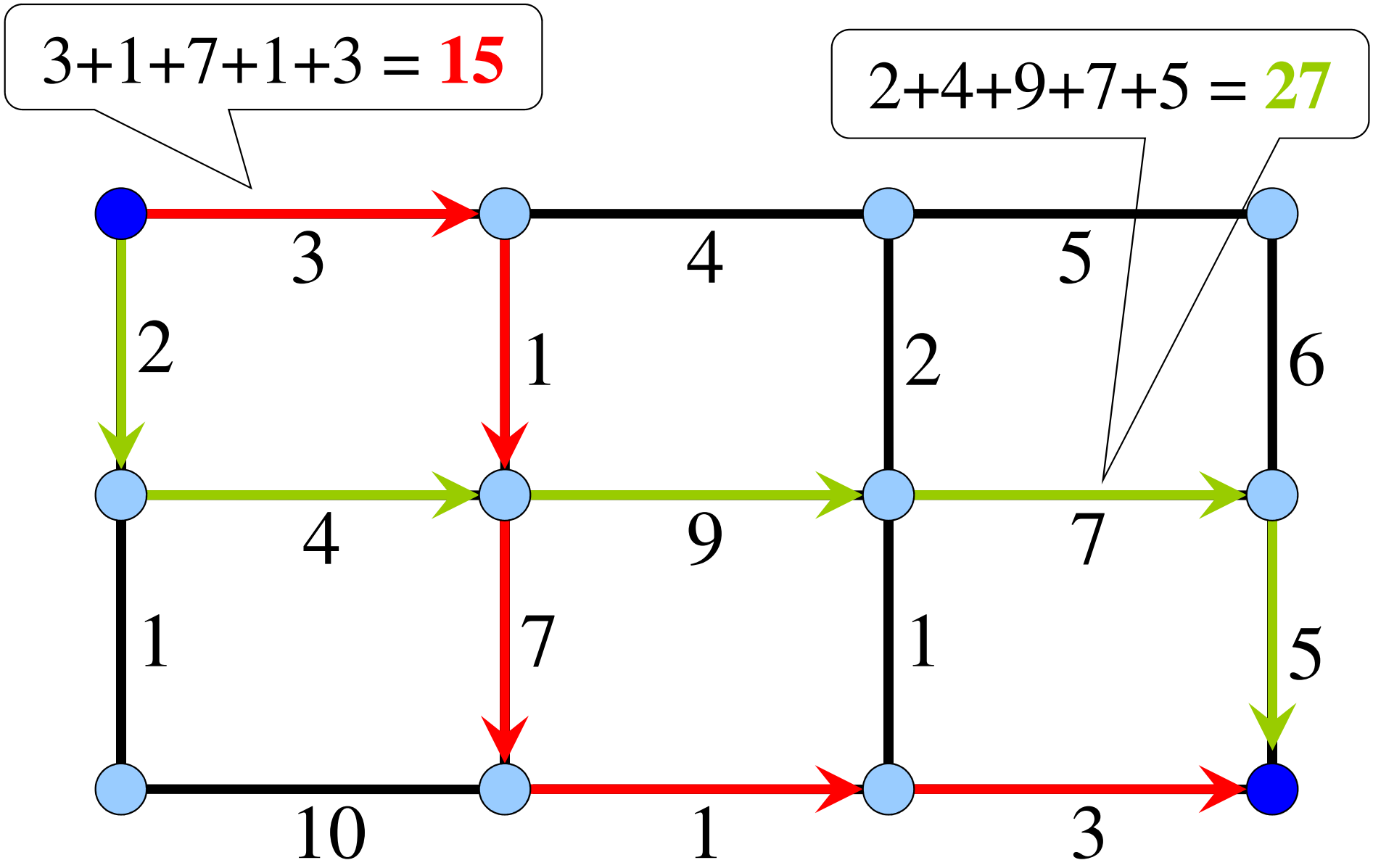
難しいなら易しくすればいいのさ！

制限した問題

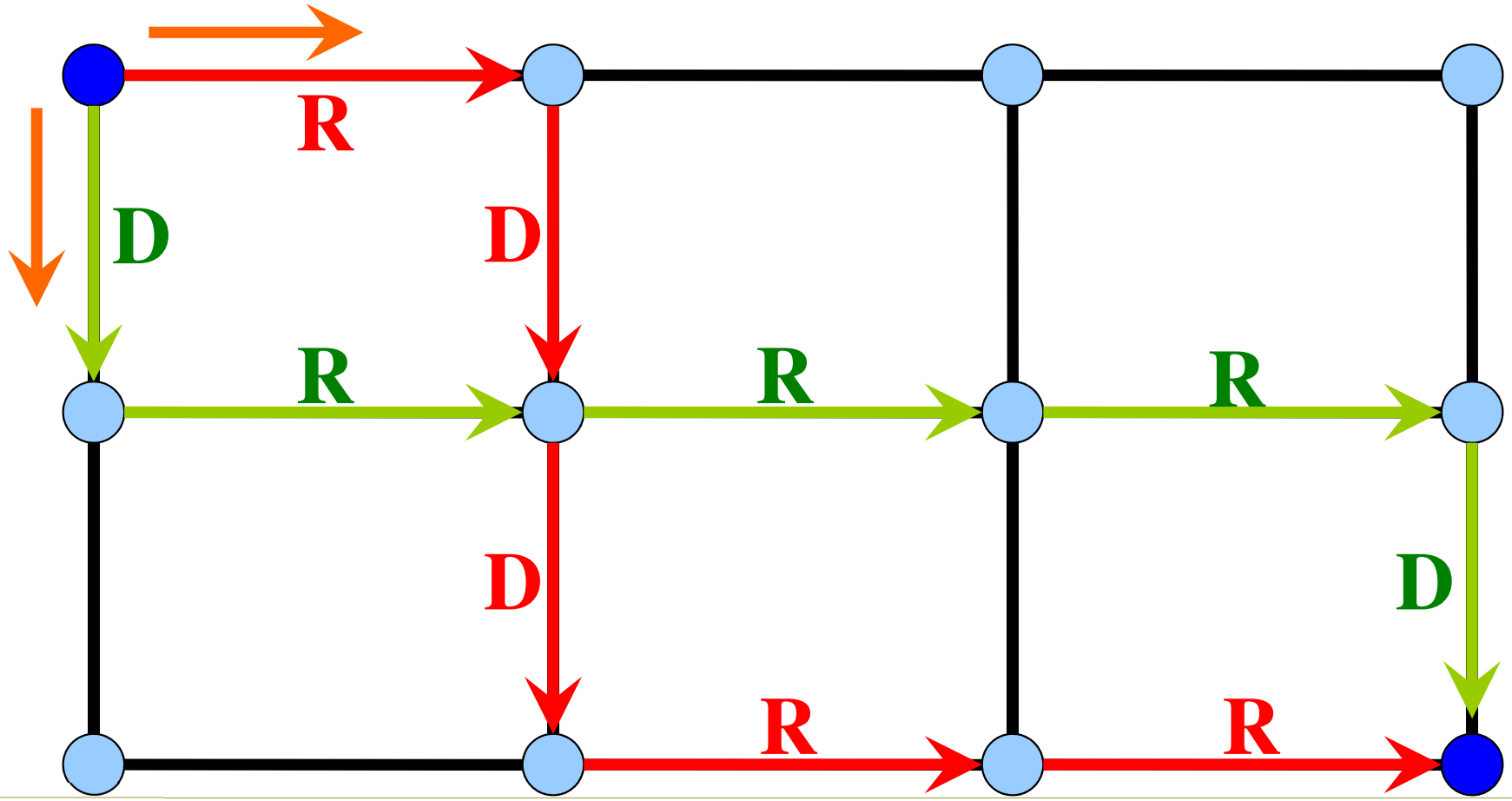
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



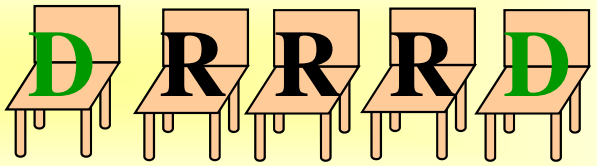
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



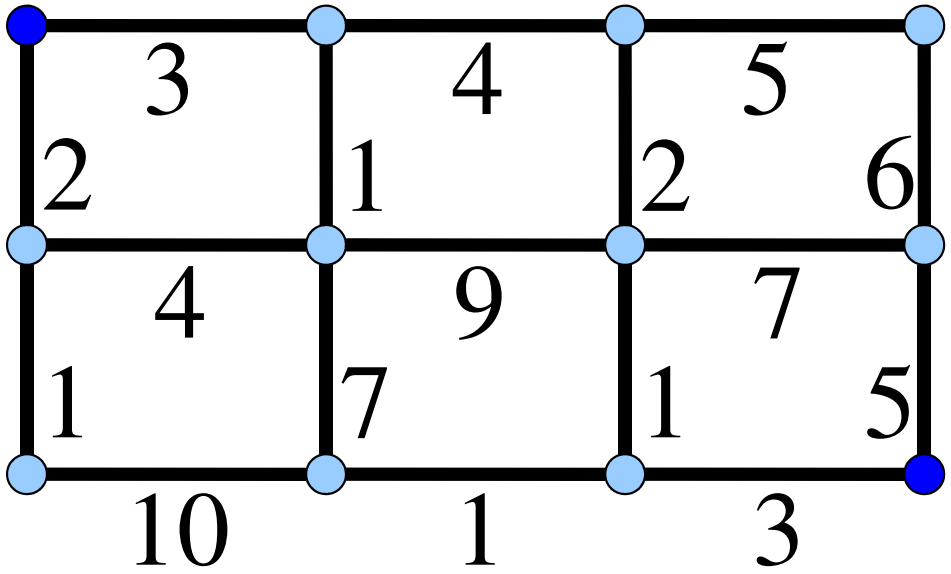
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

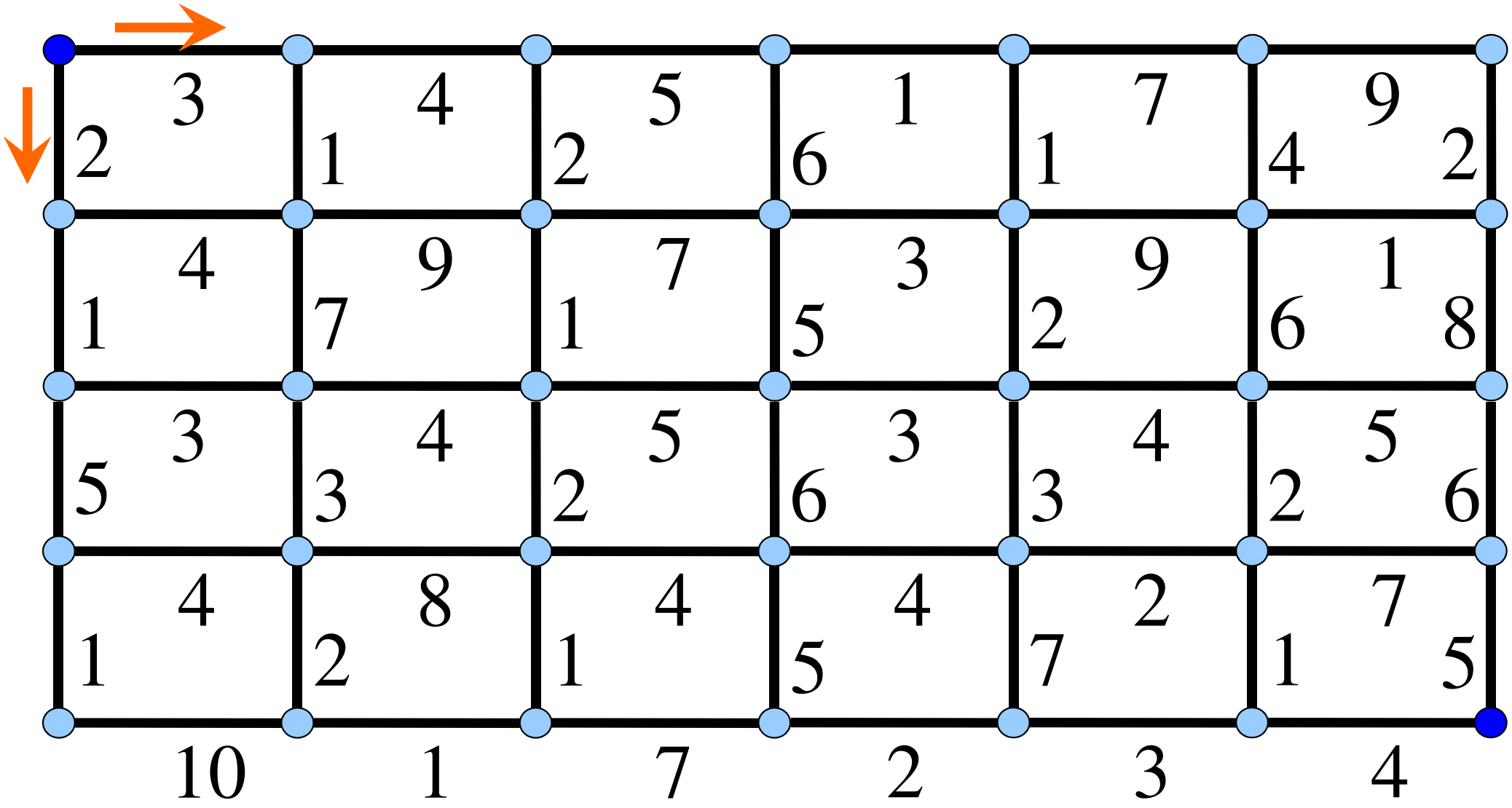
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

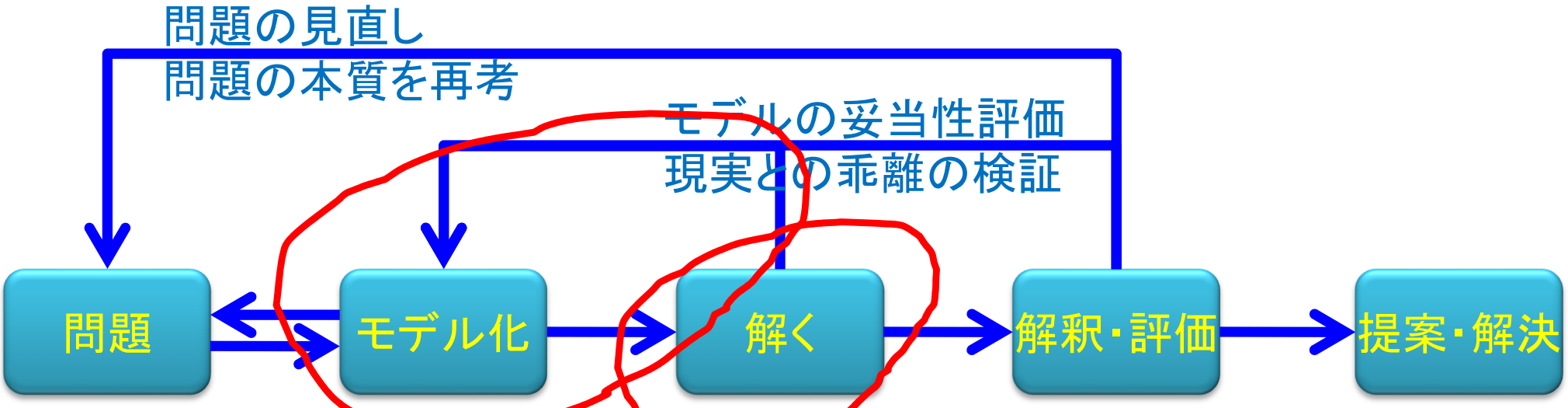
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

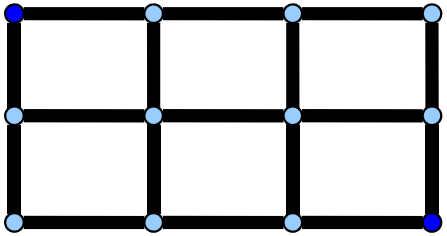
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

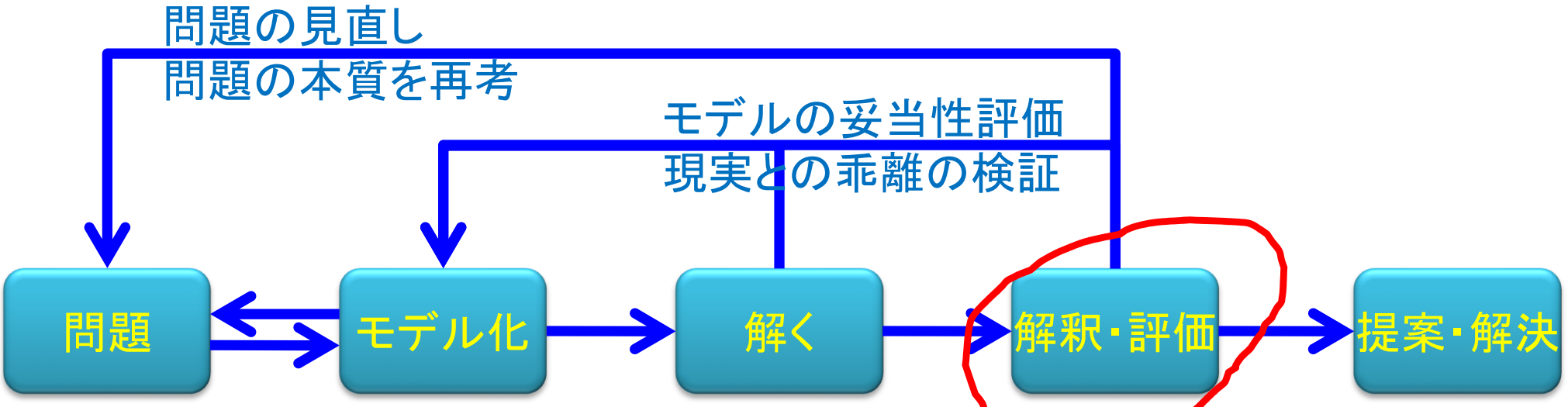
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

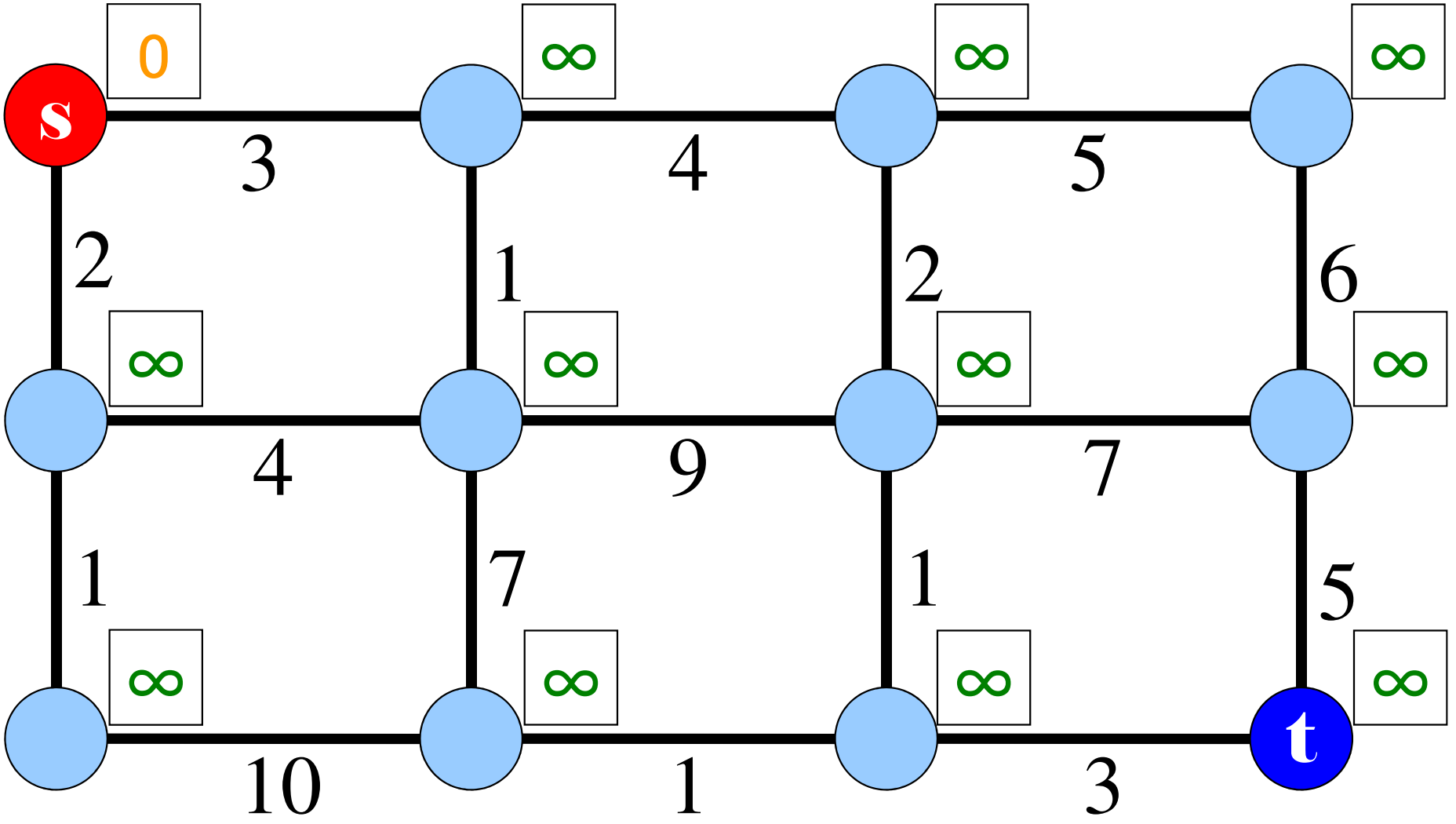
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

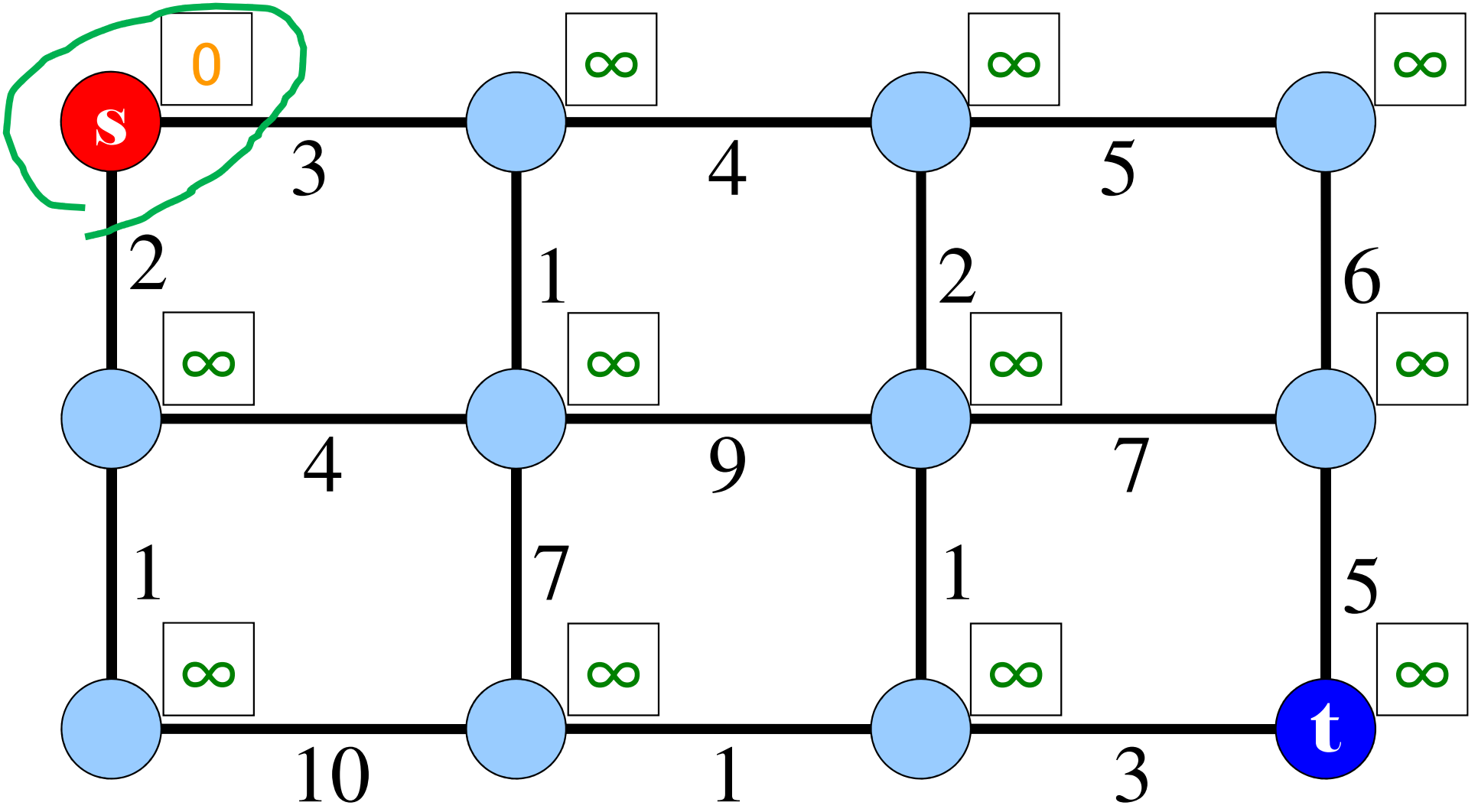
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



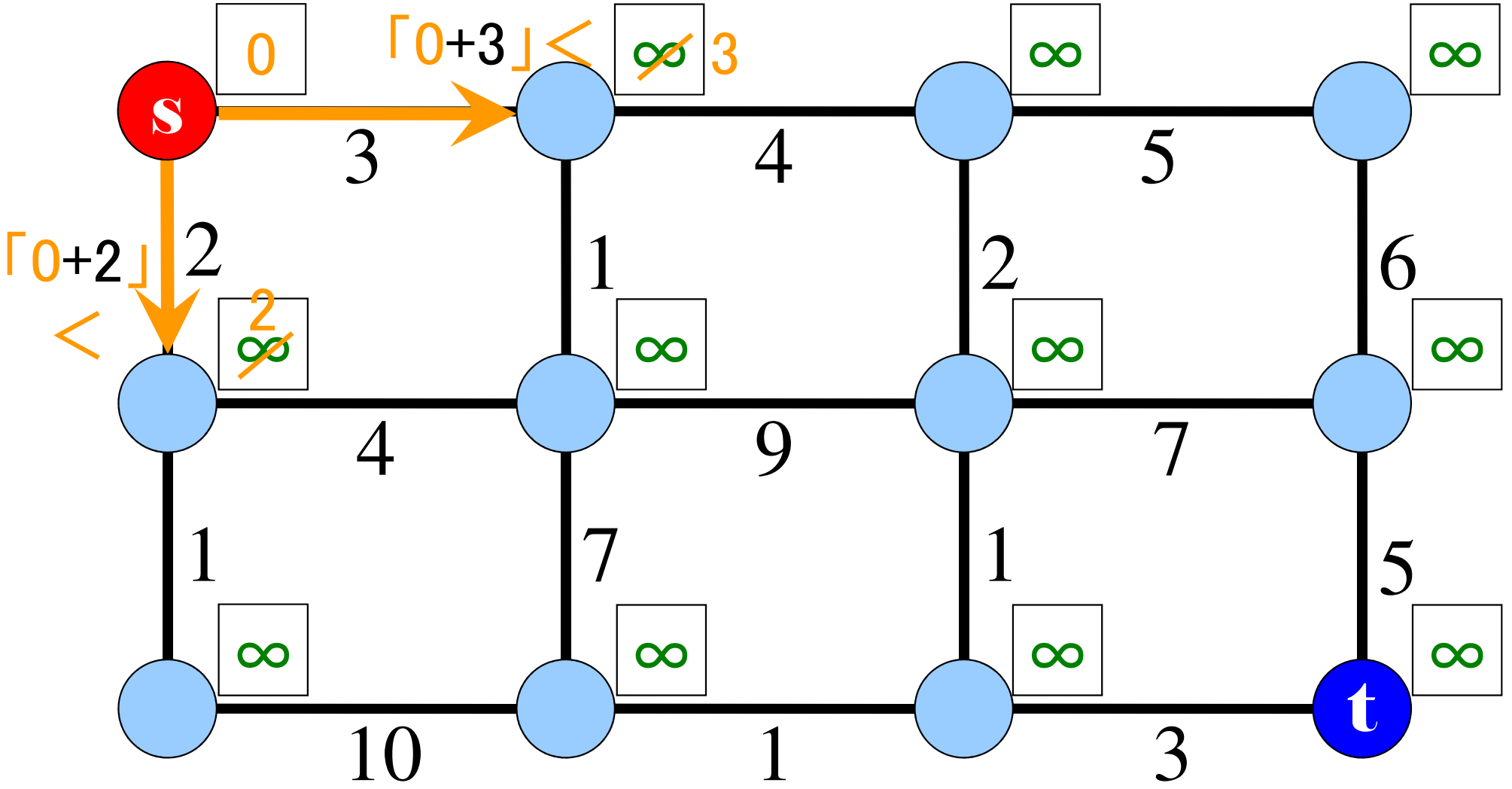
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

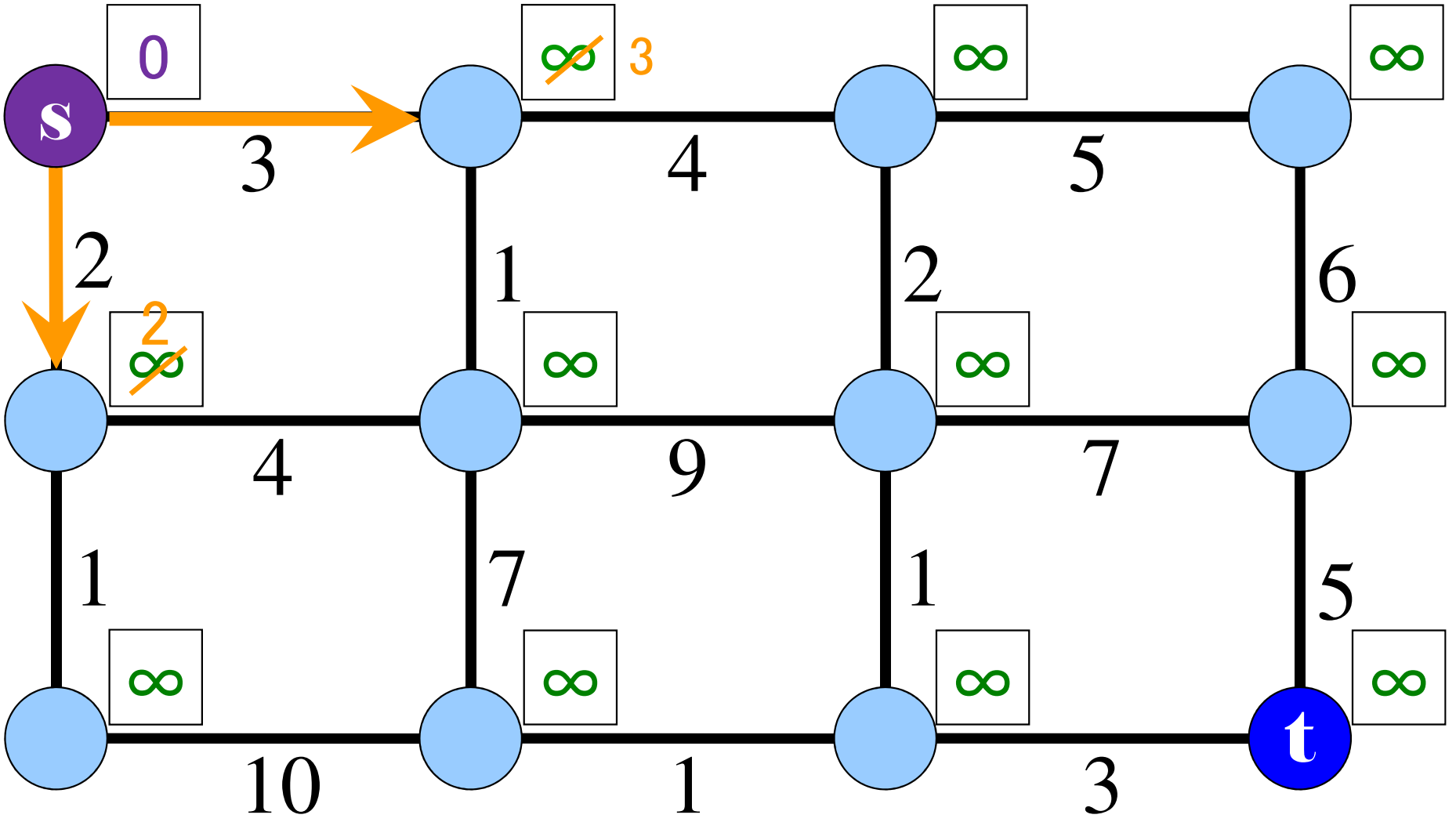


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

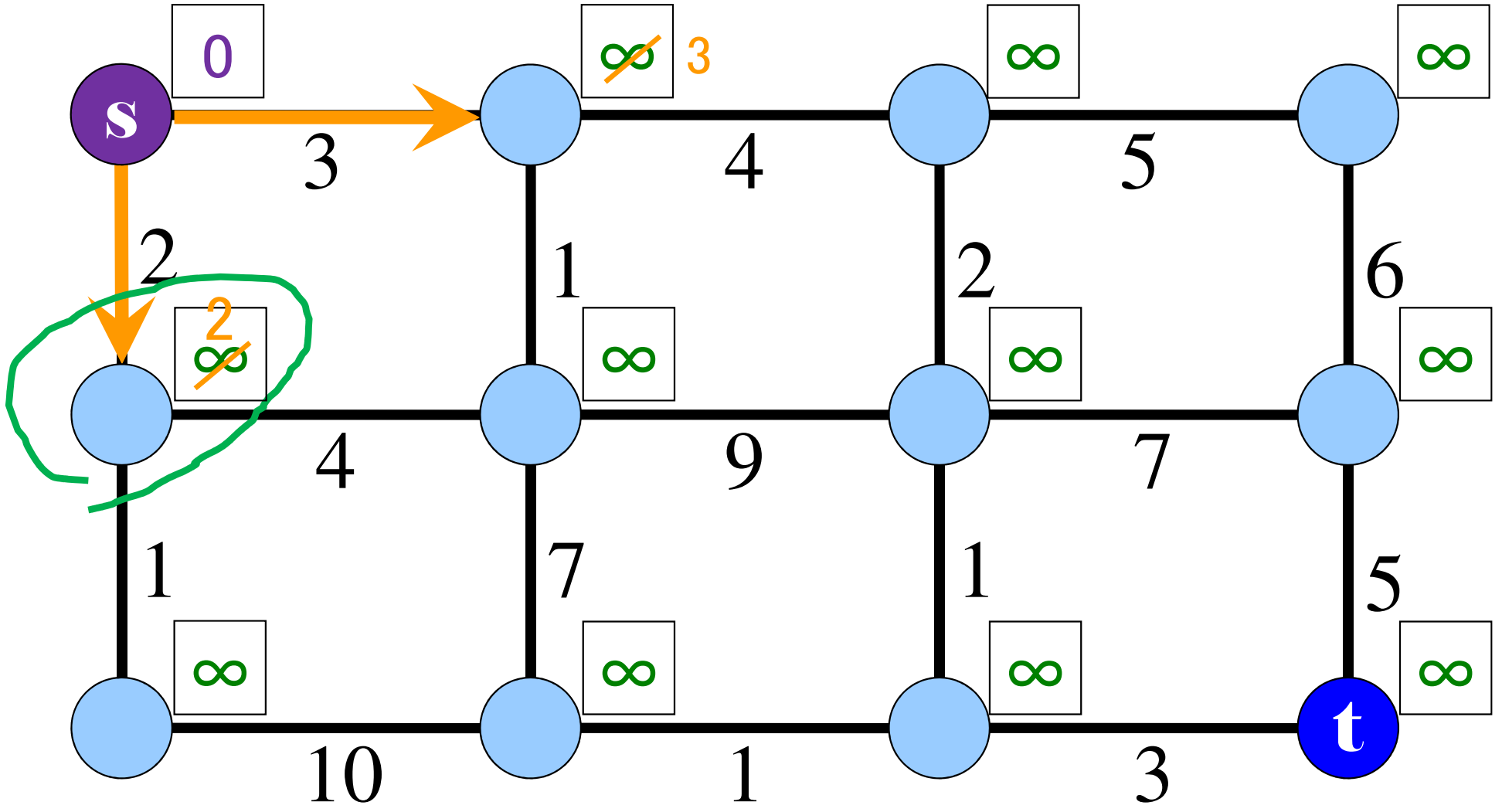
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



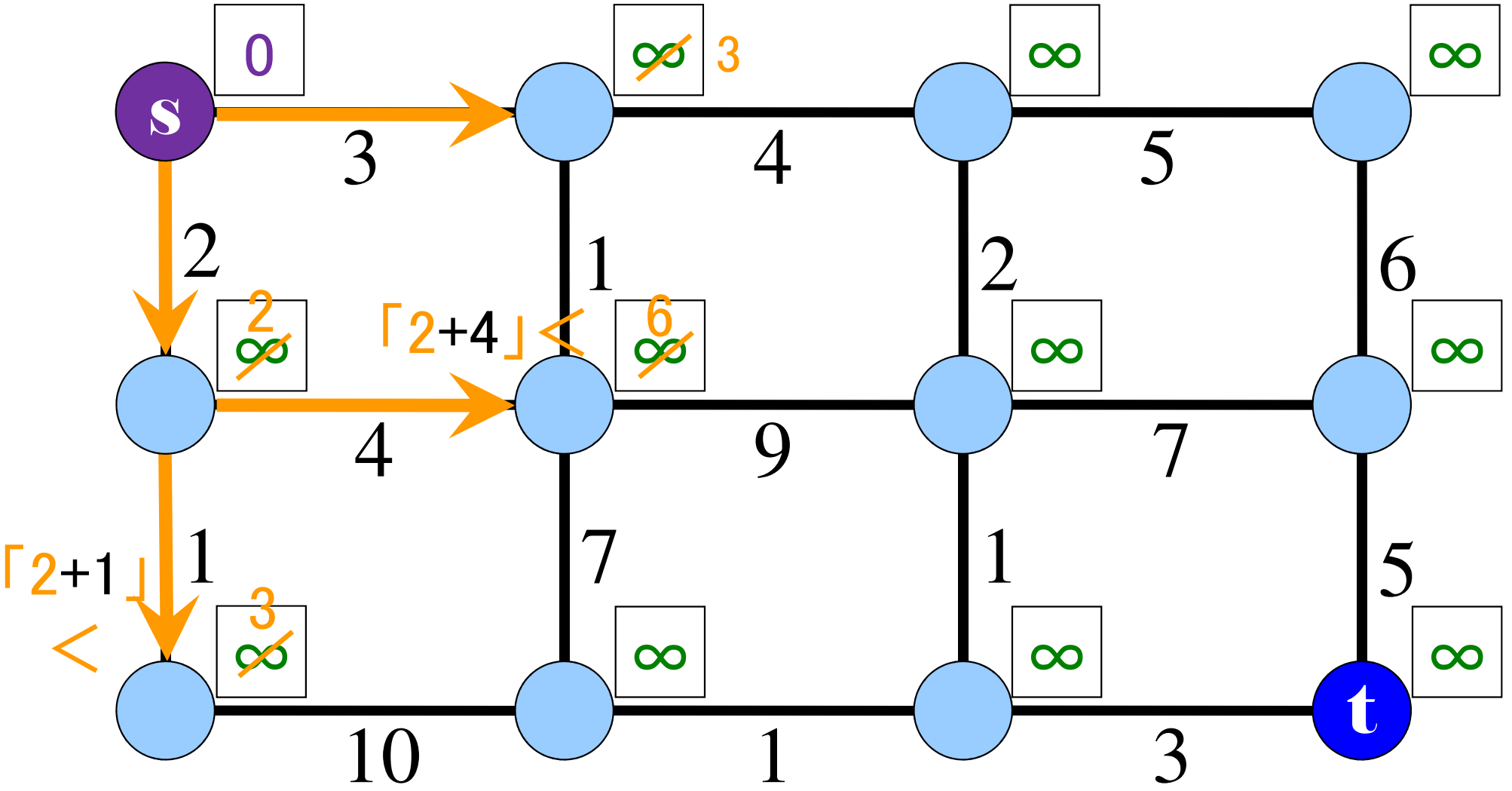
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

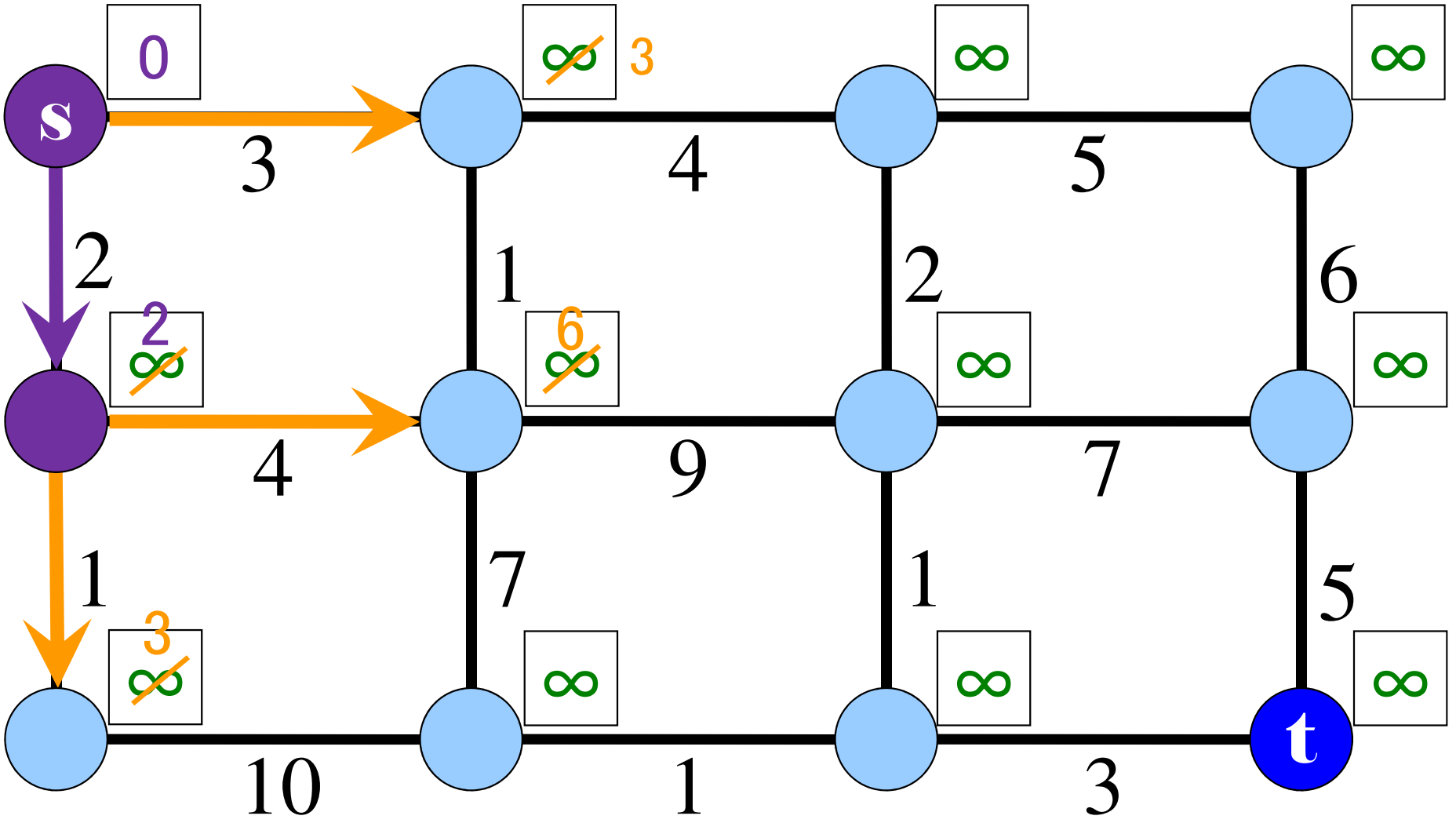


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

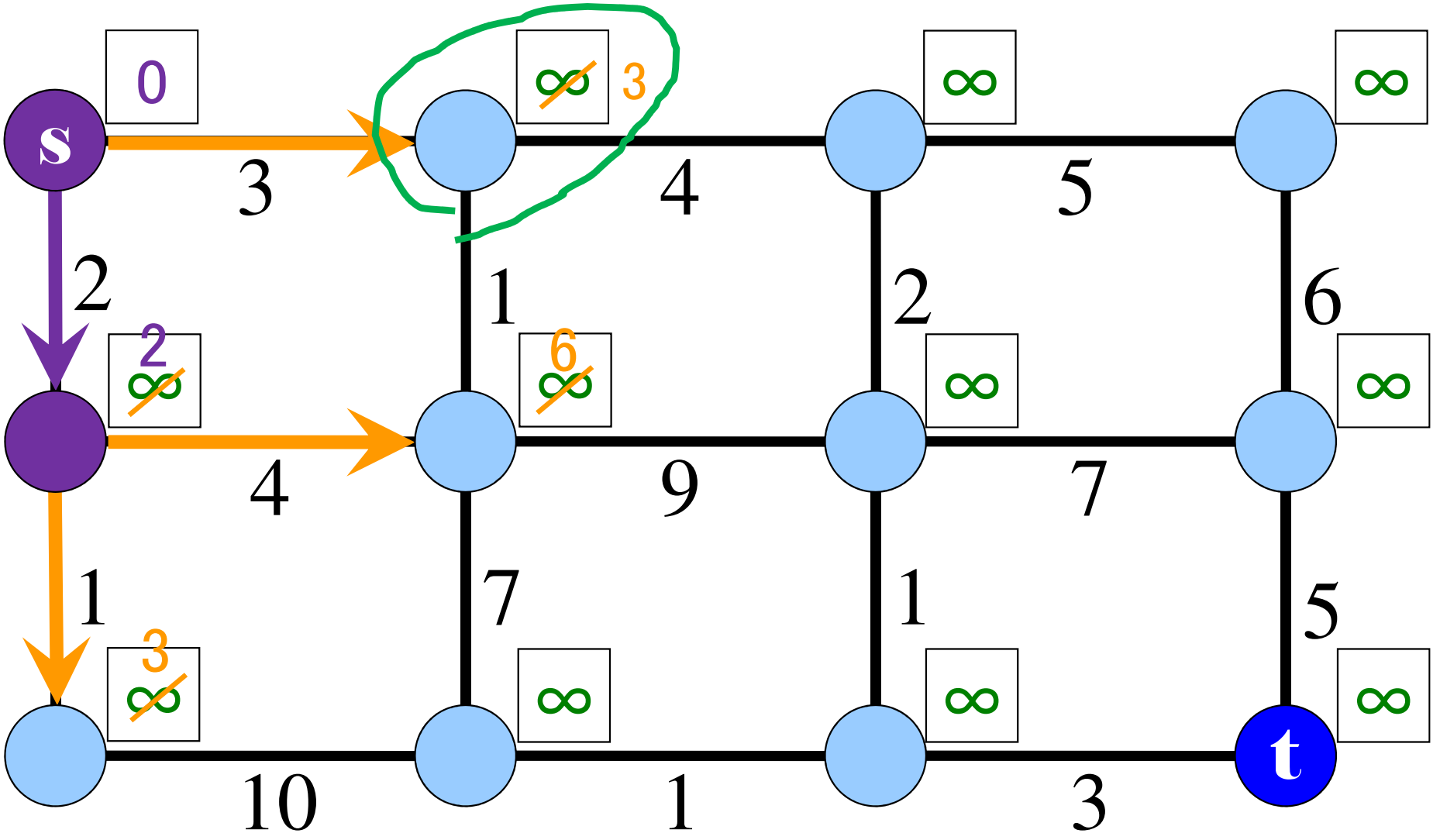
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



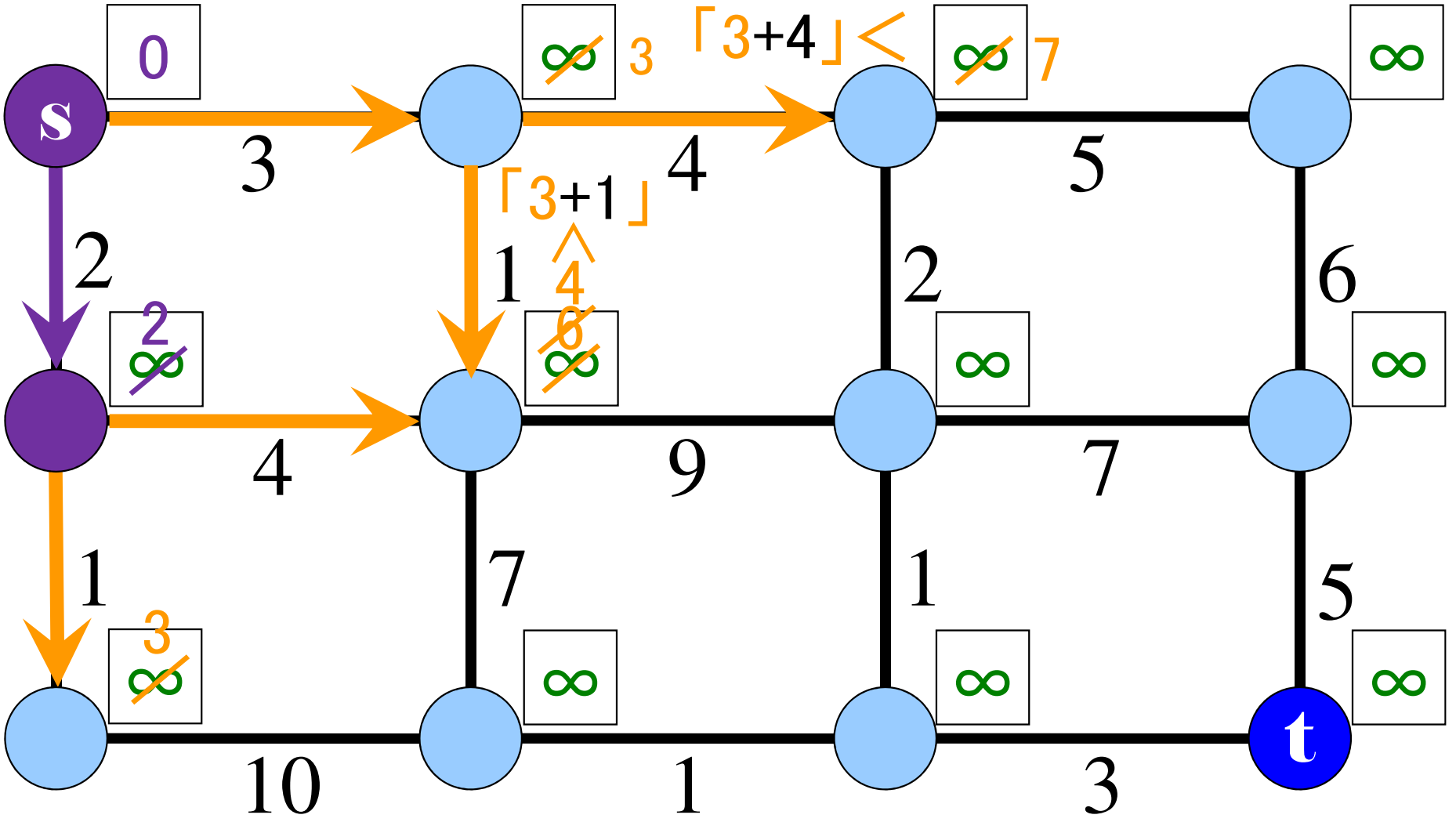
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

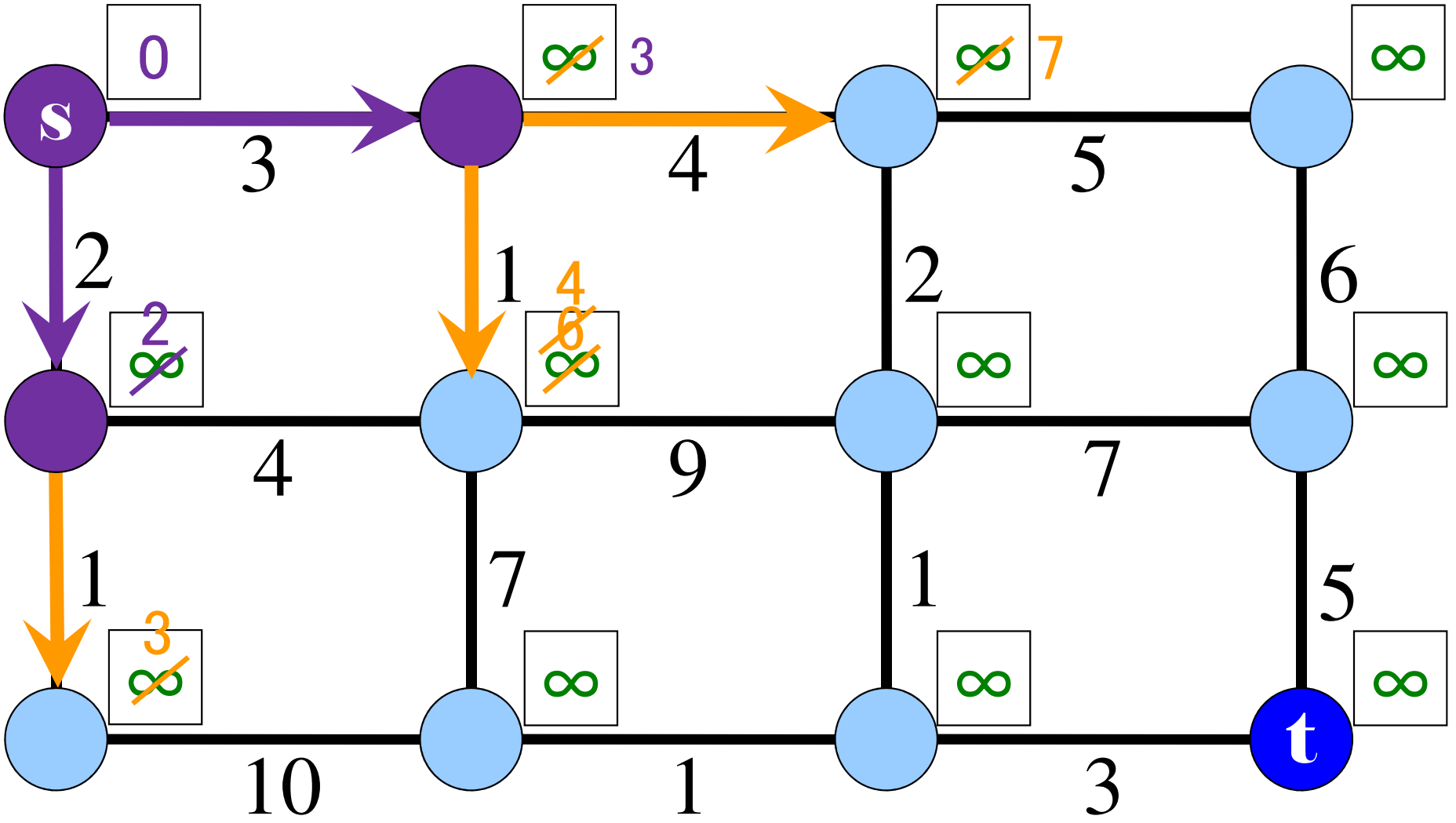


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

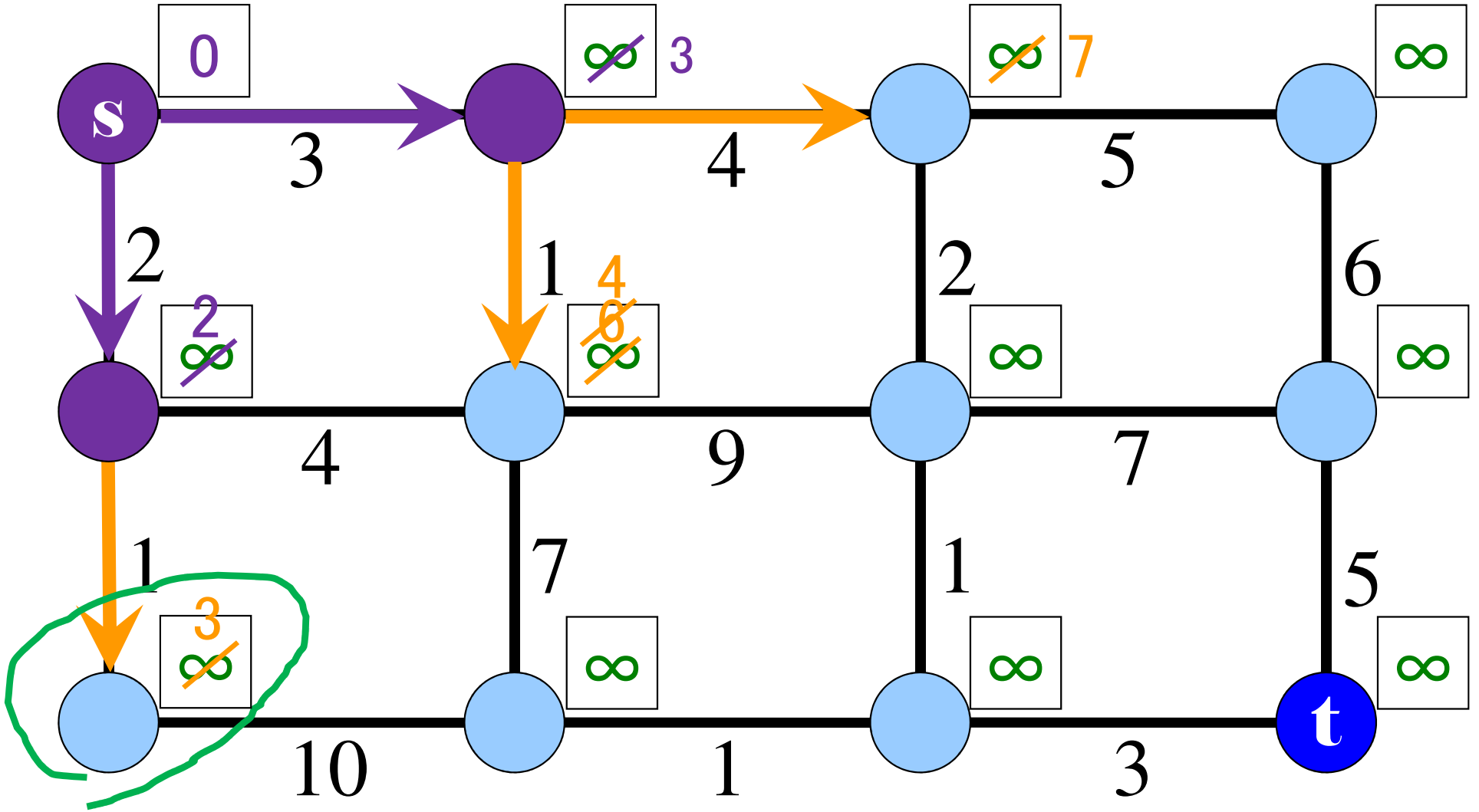
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



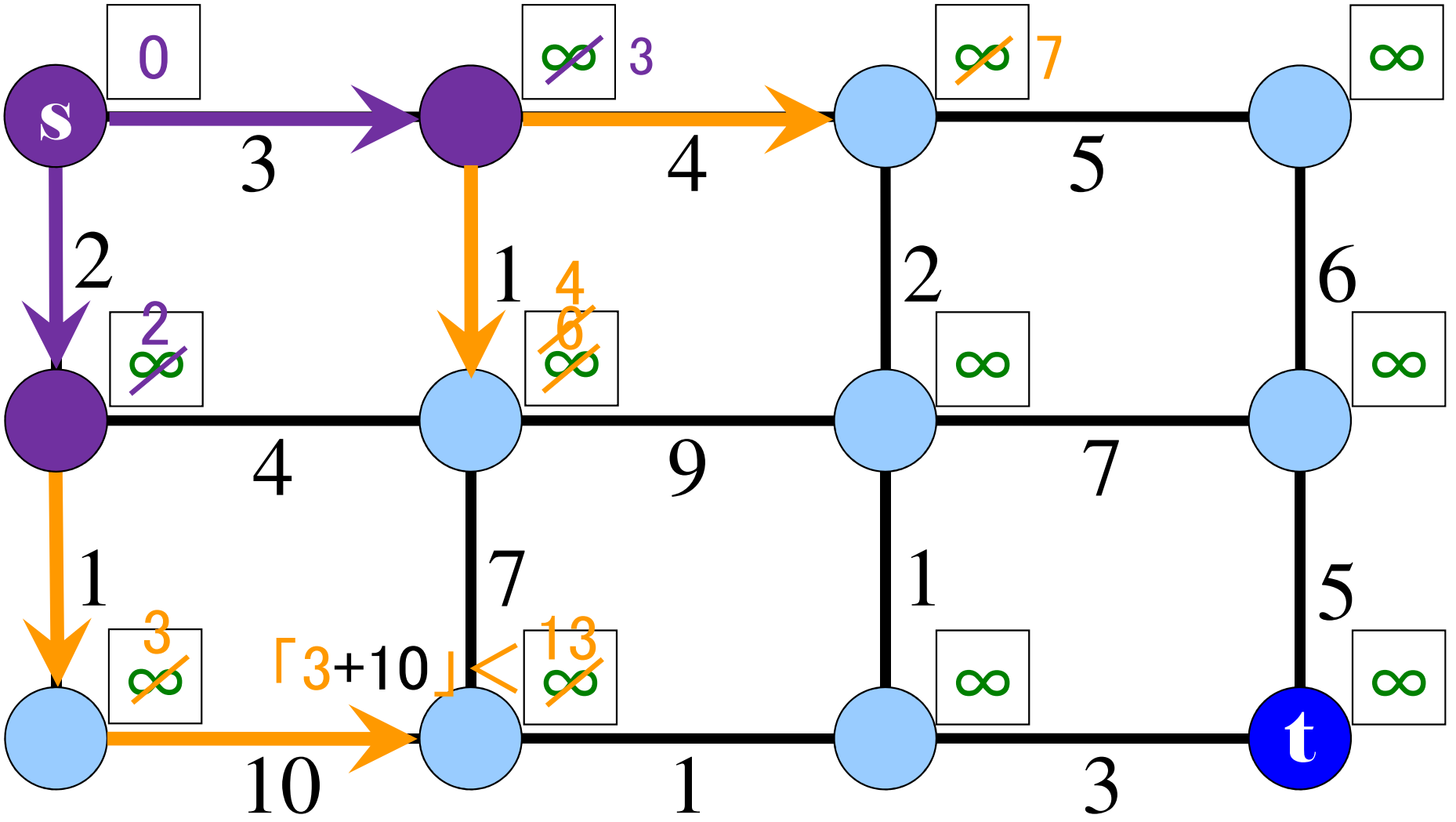
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

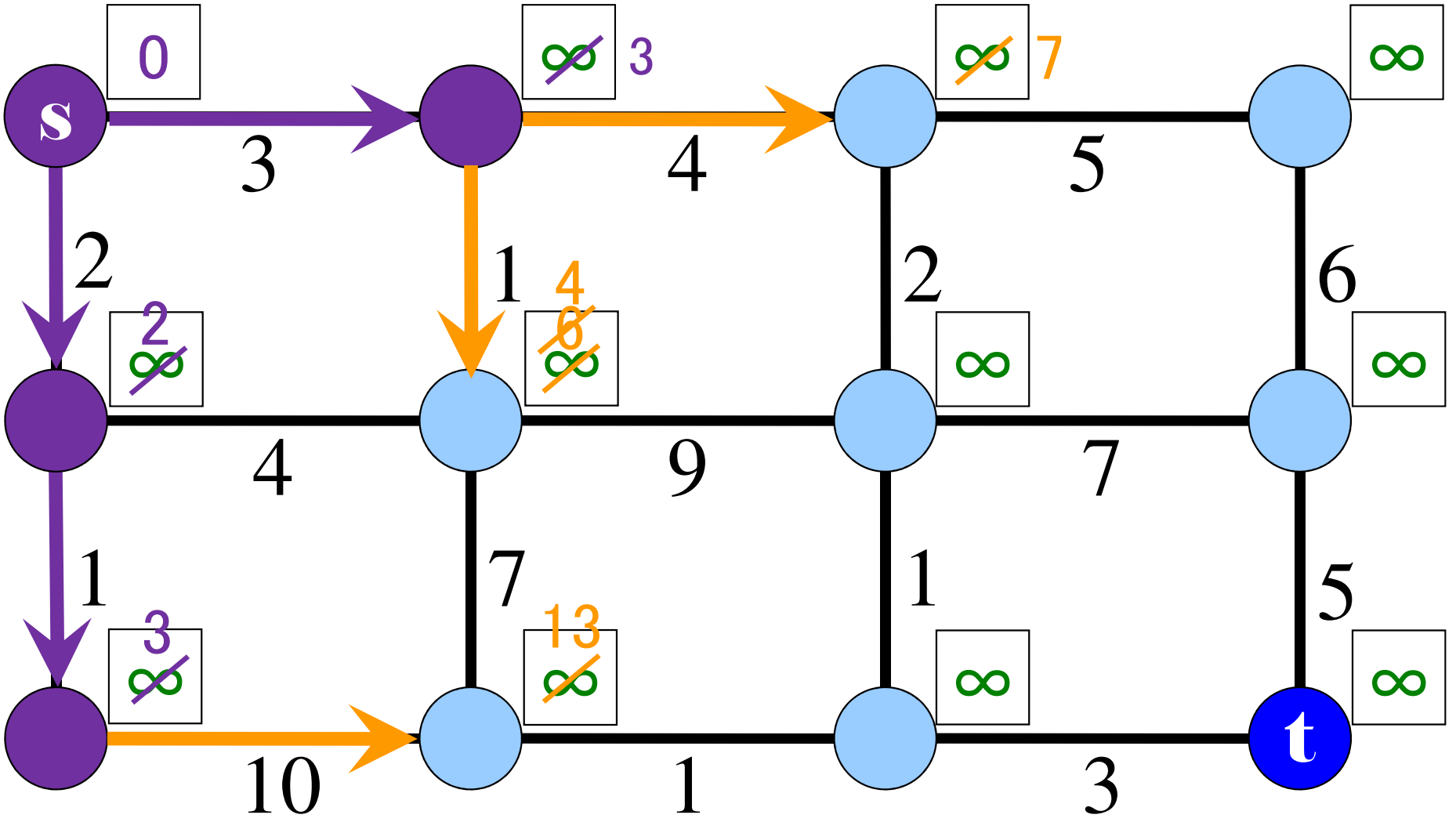


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

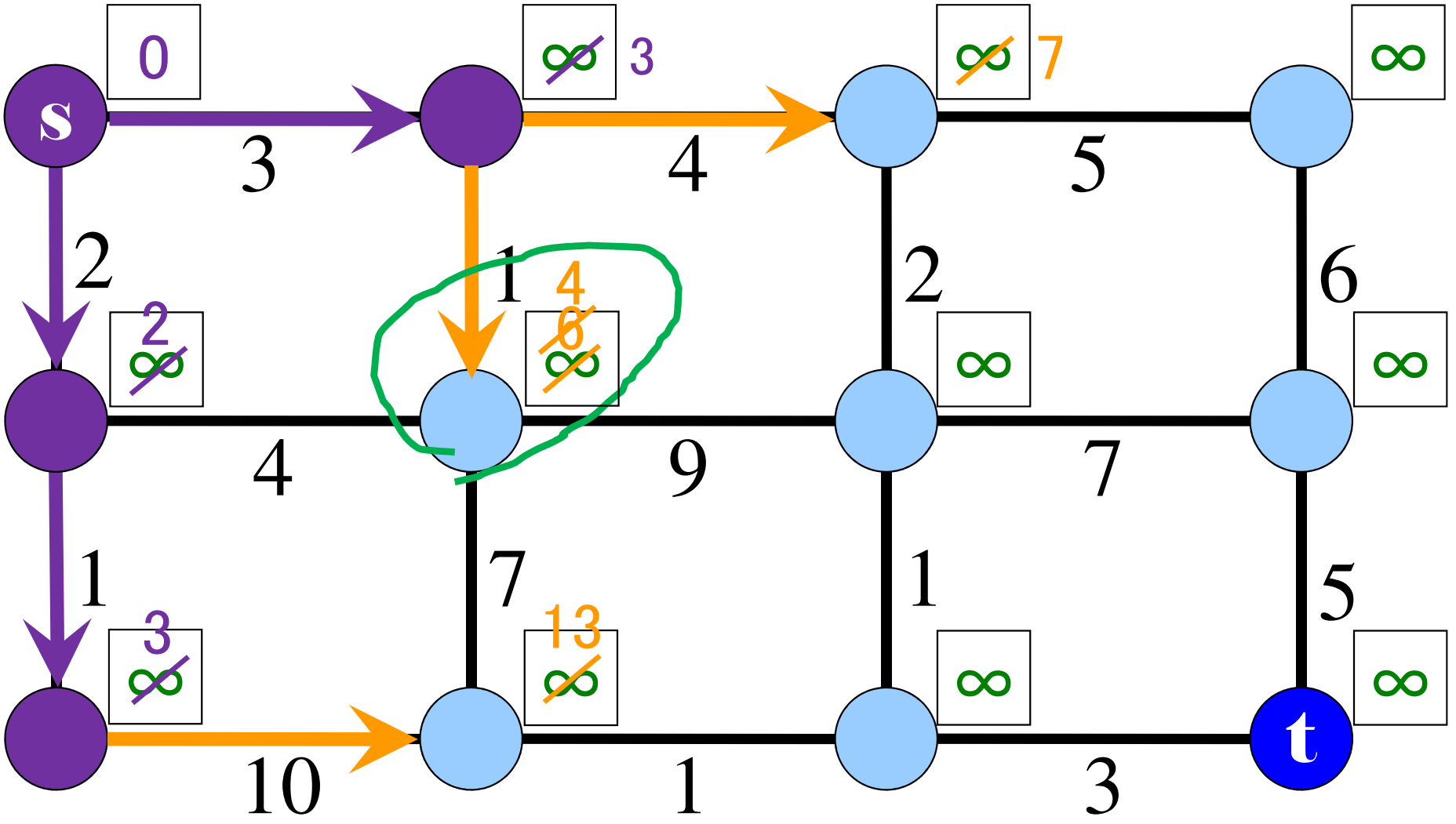
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



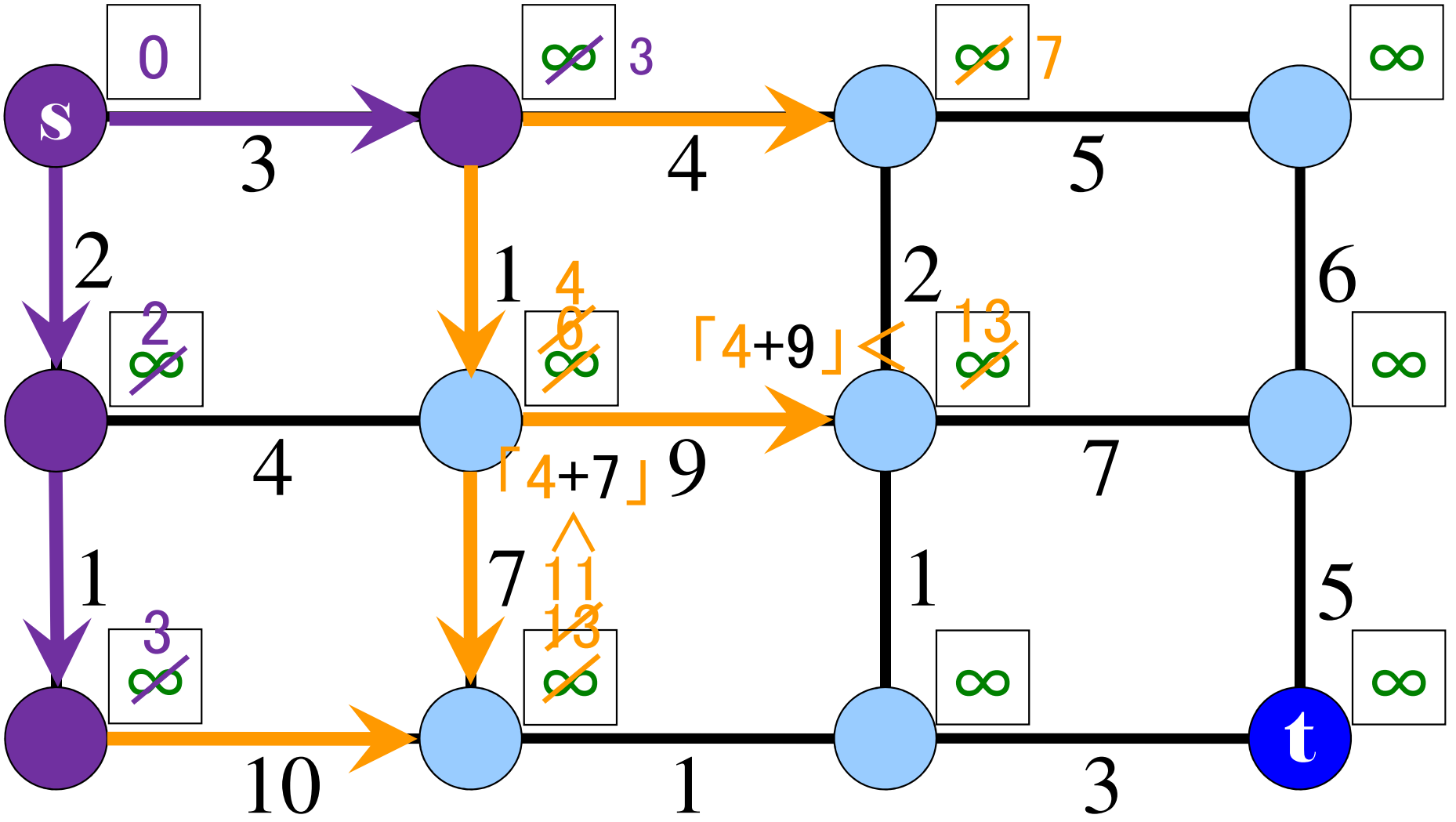
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

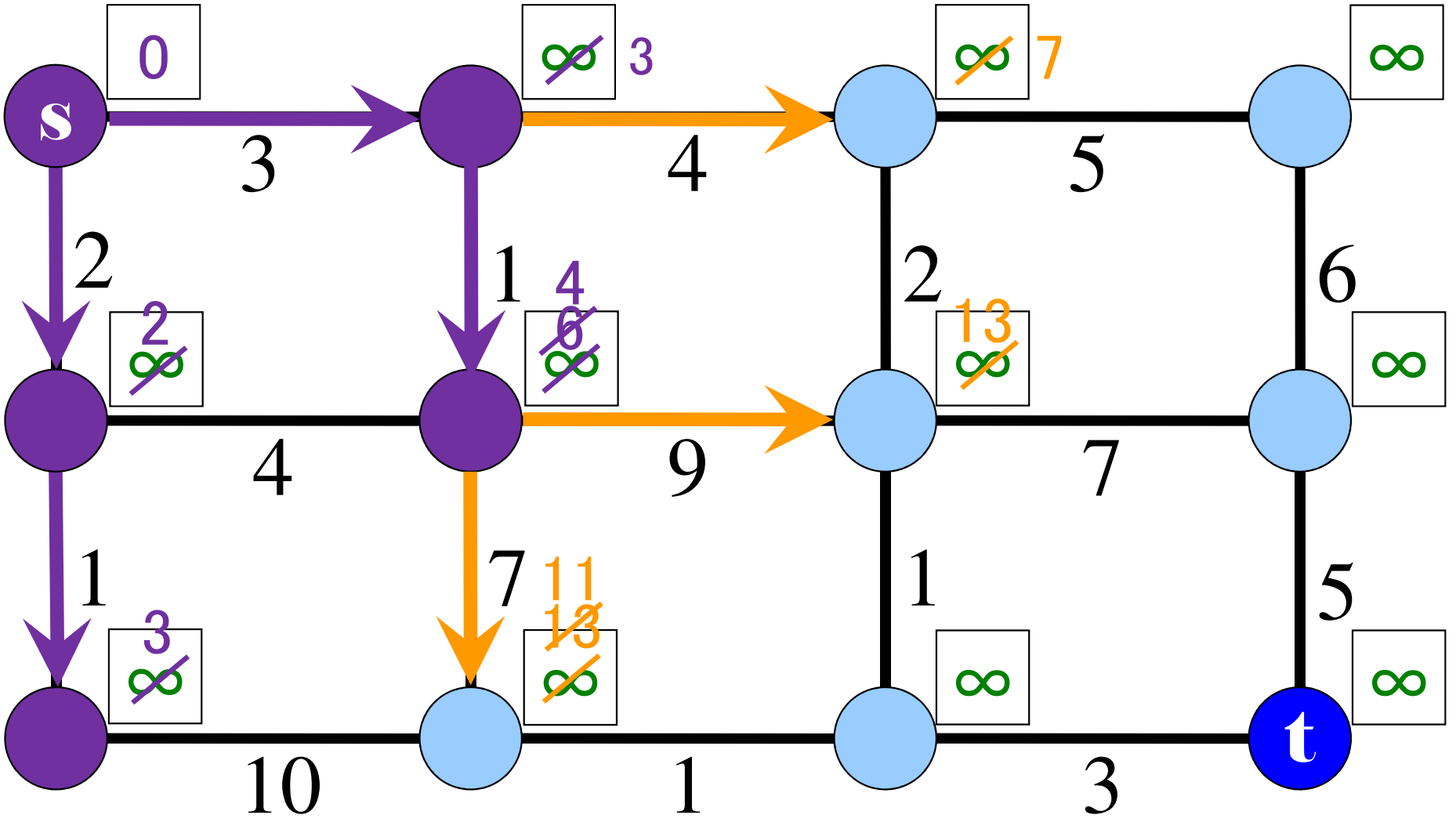


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

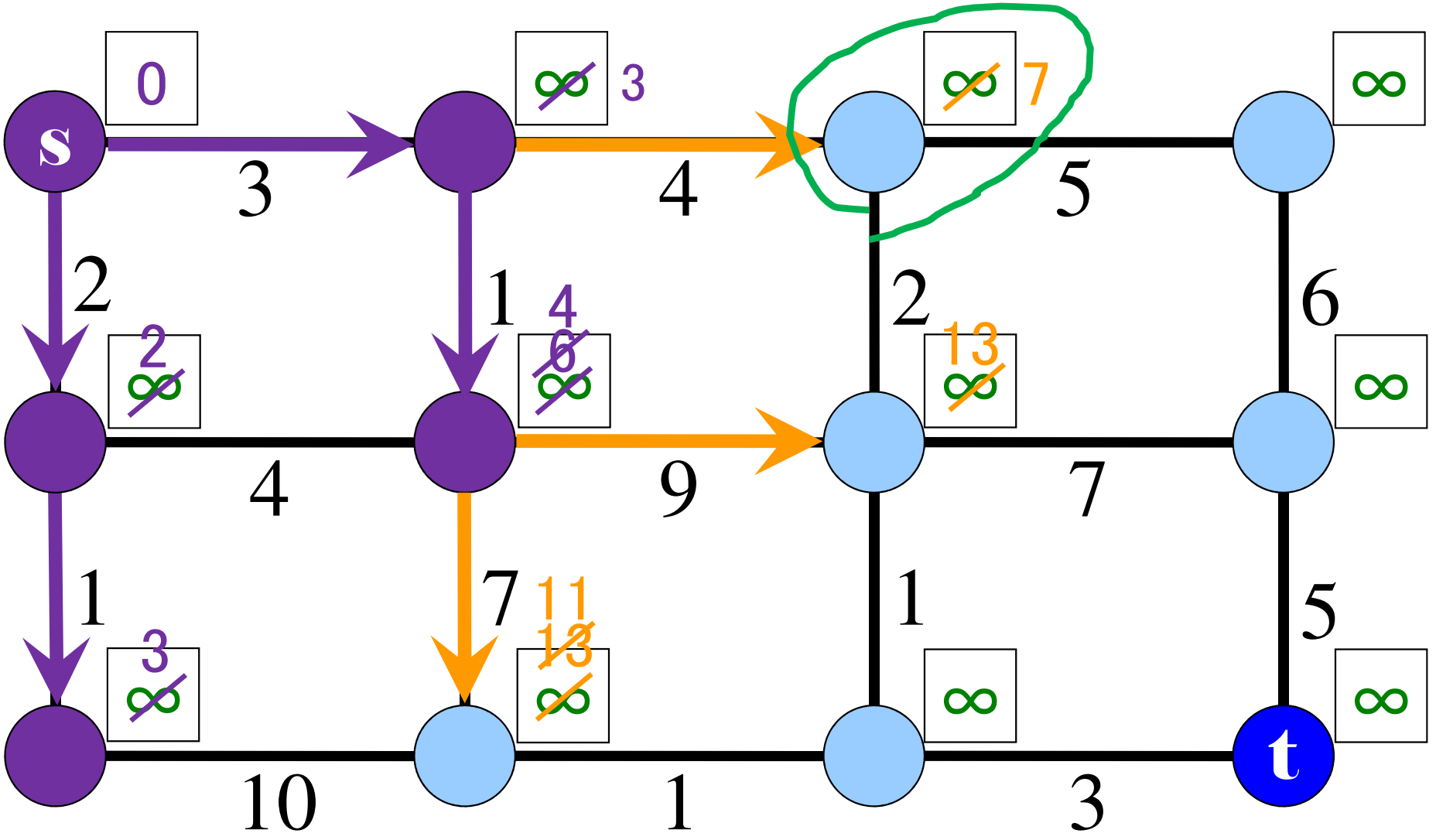
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



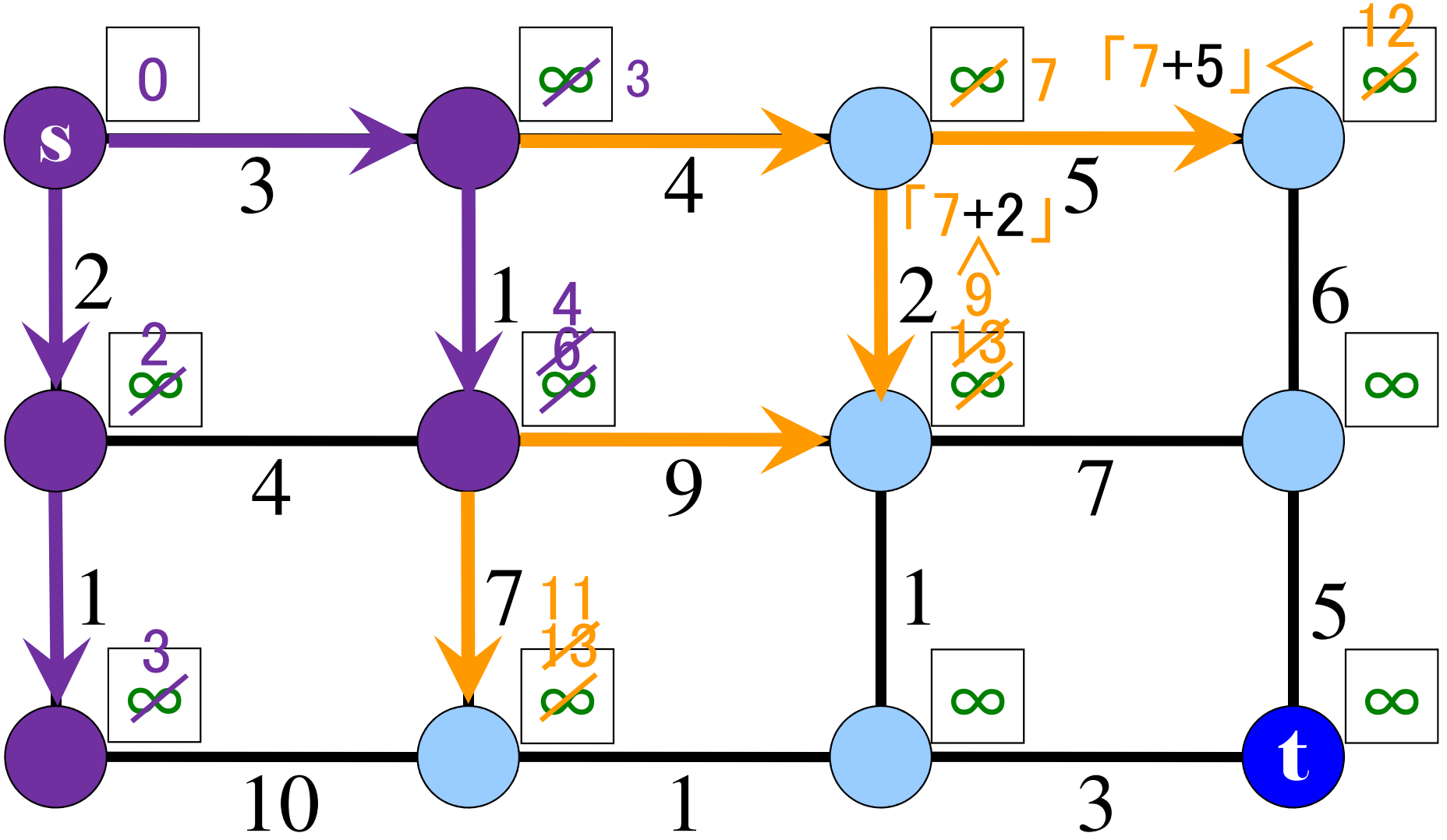
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

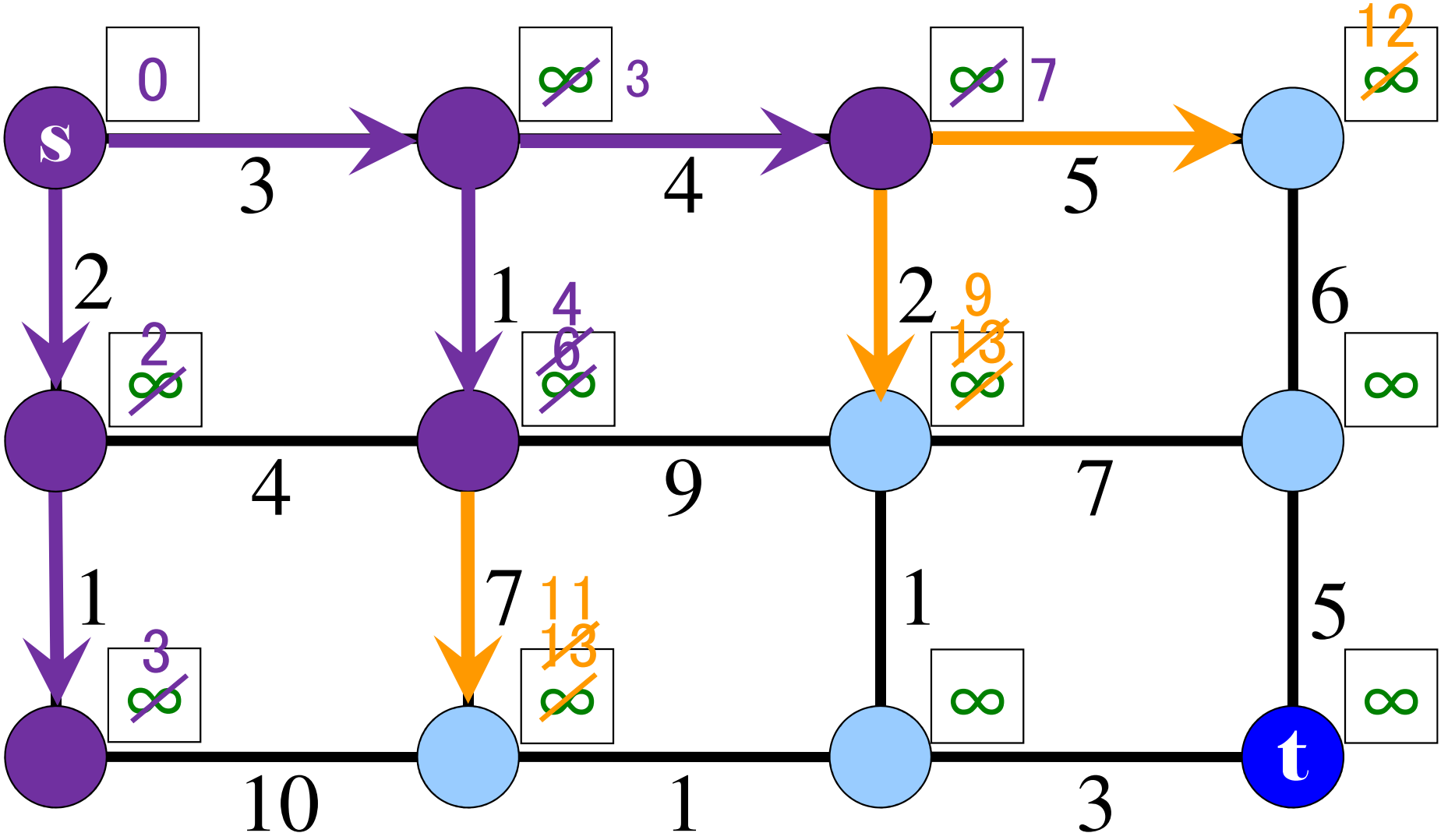


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

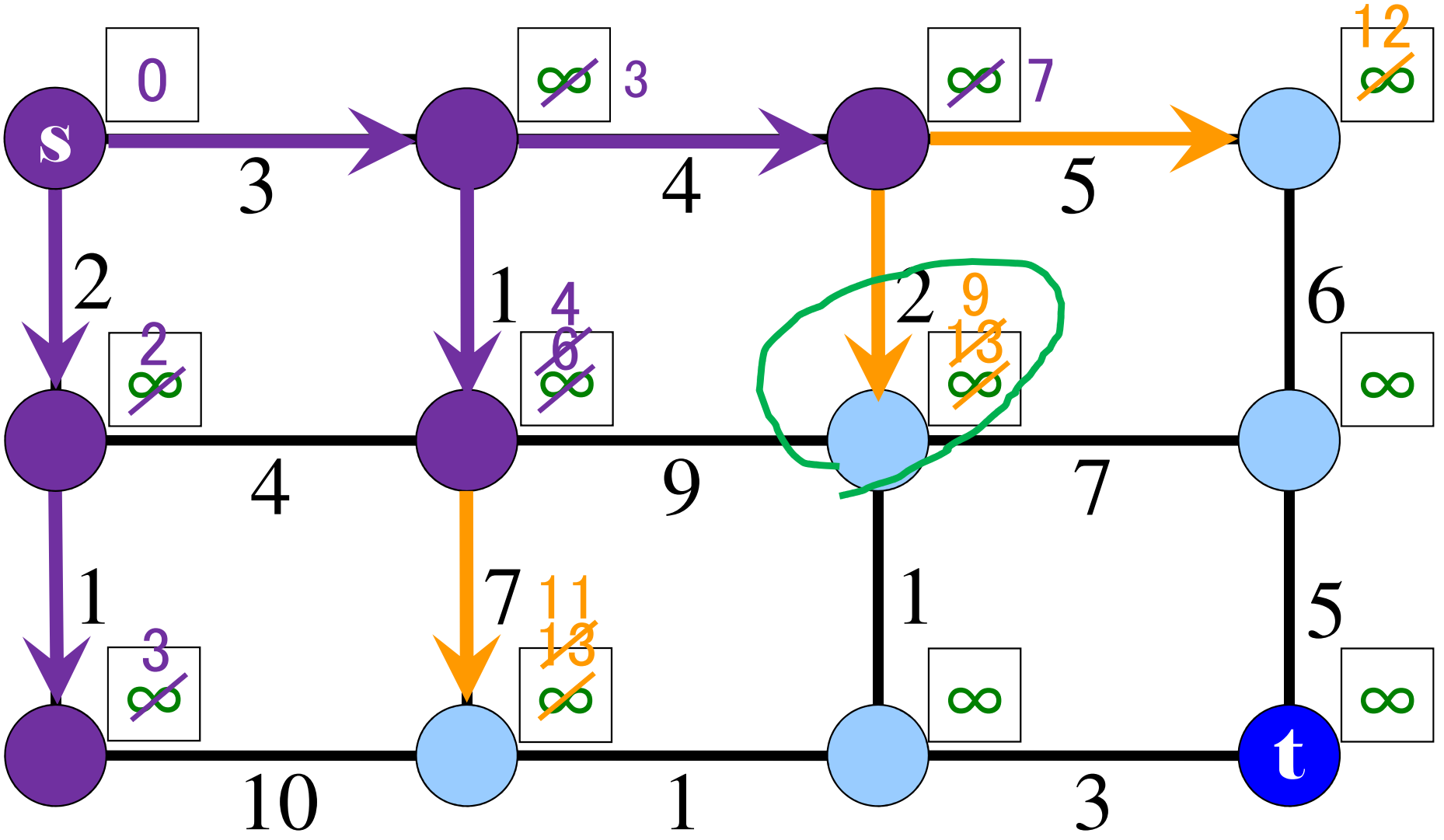
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



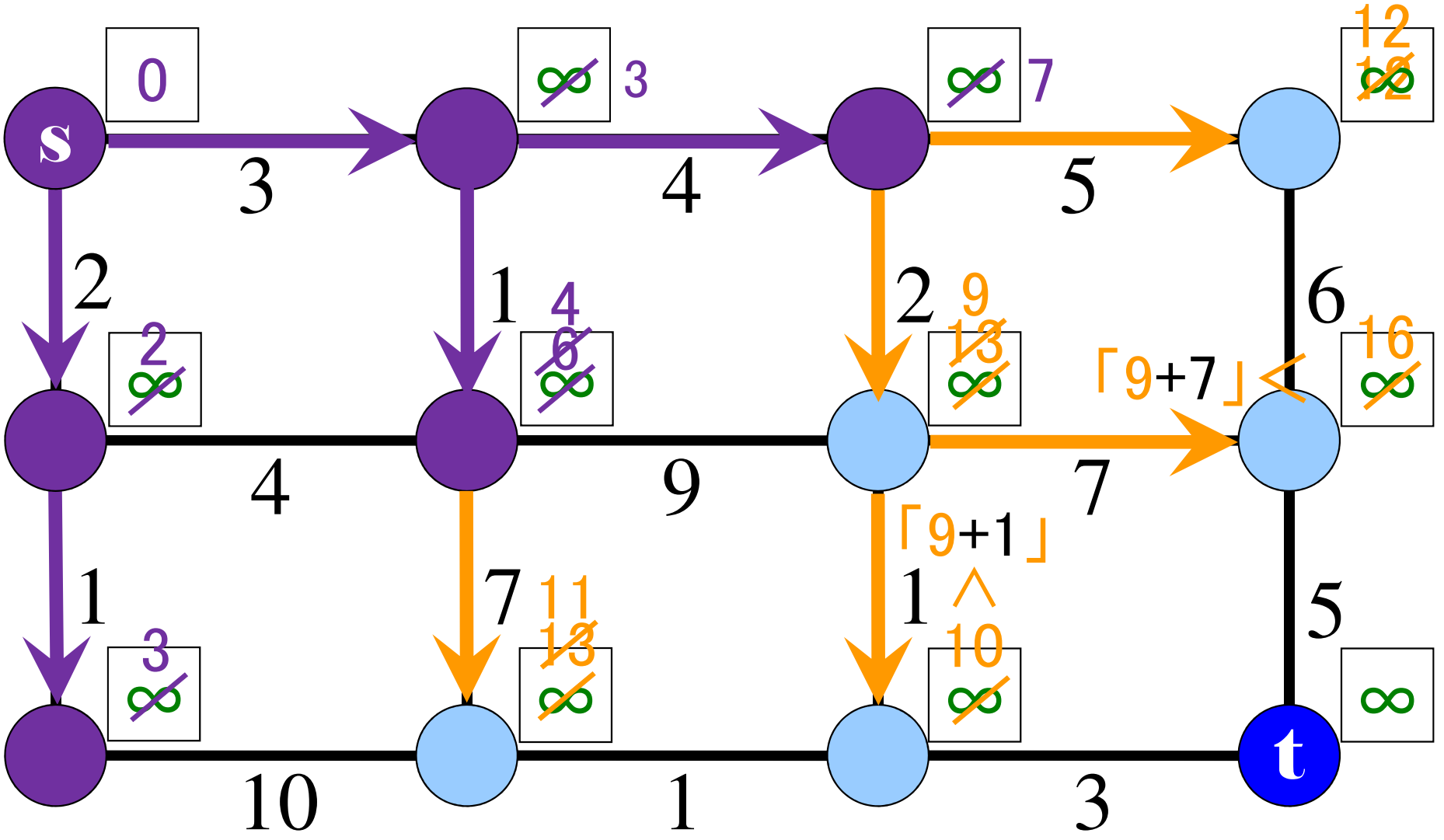
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

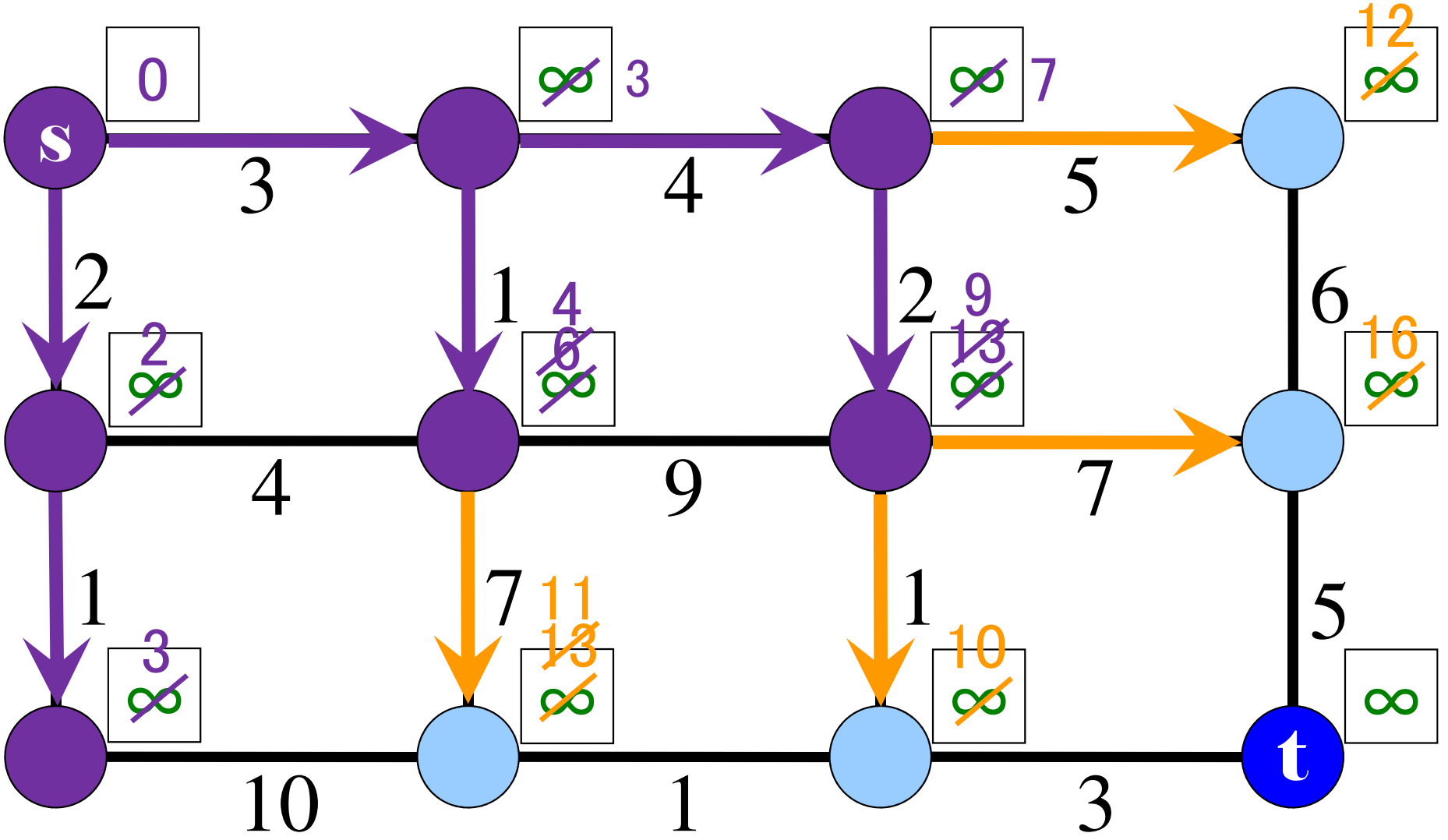


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

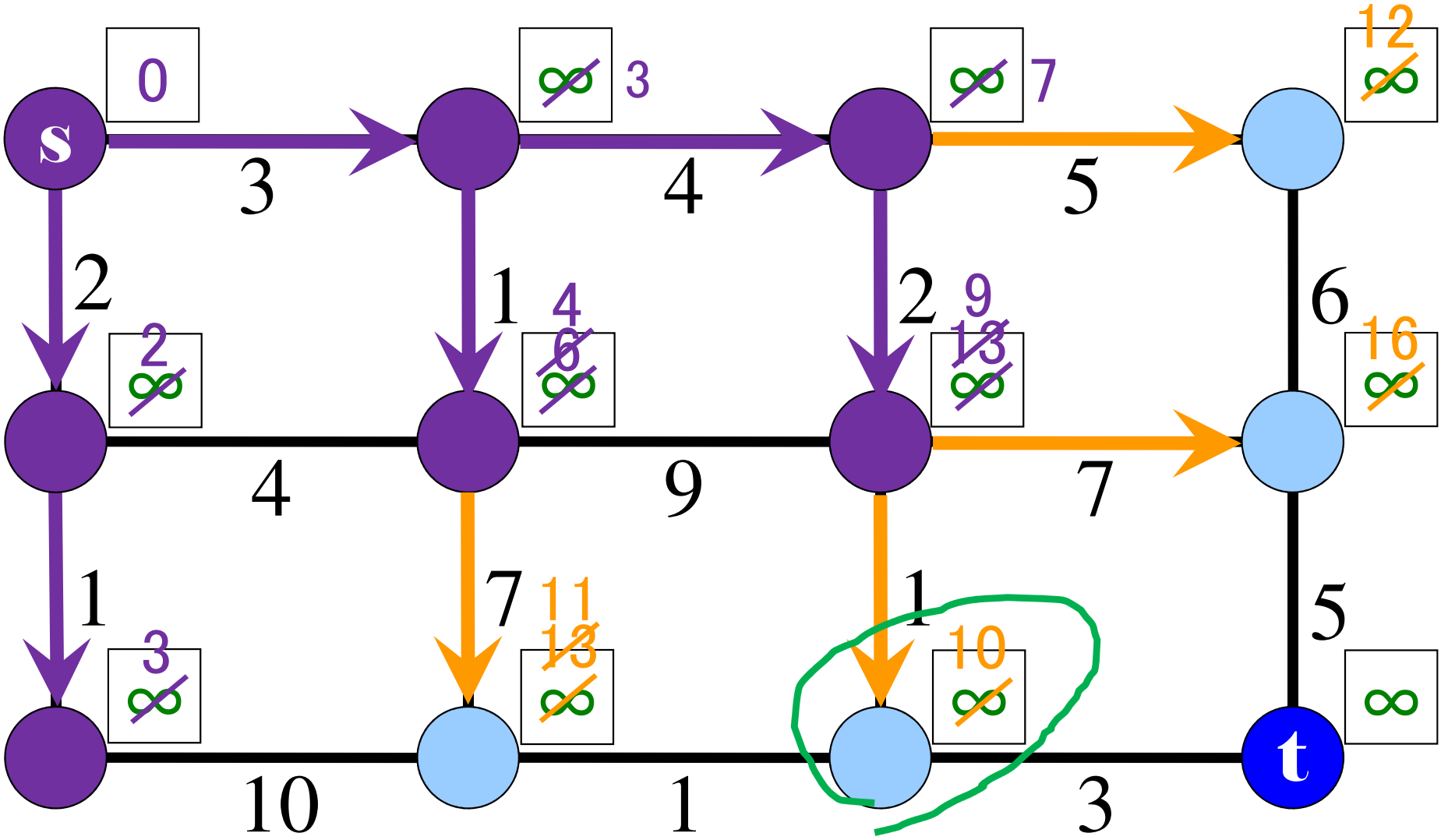
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



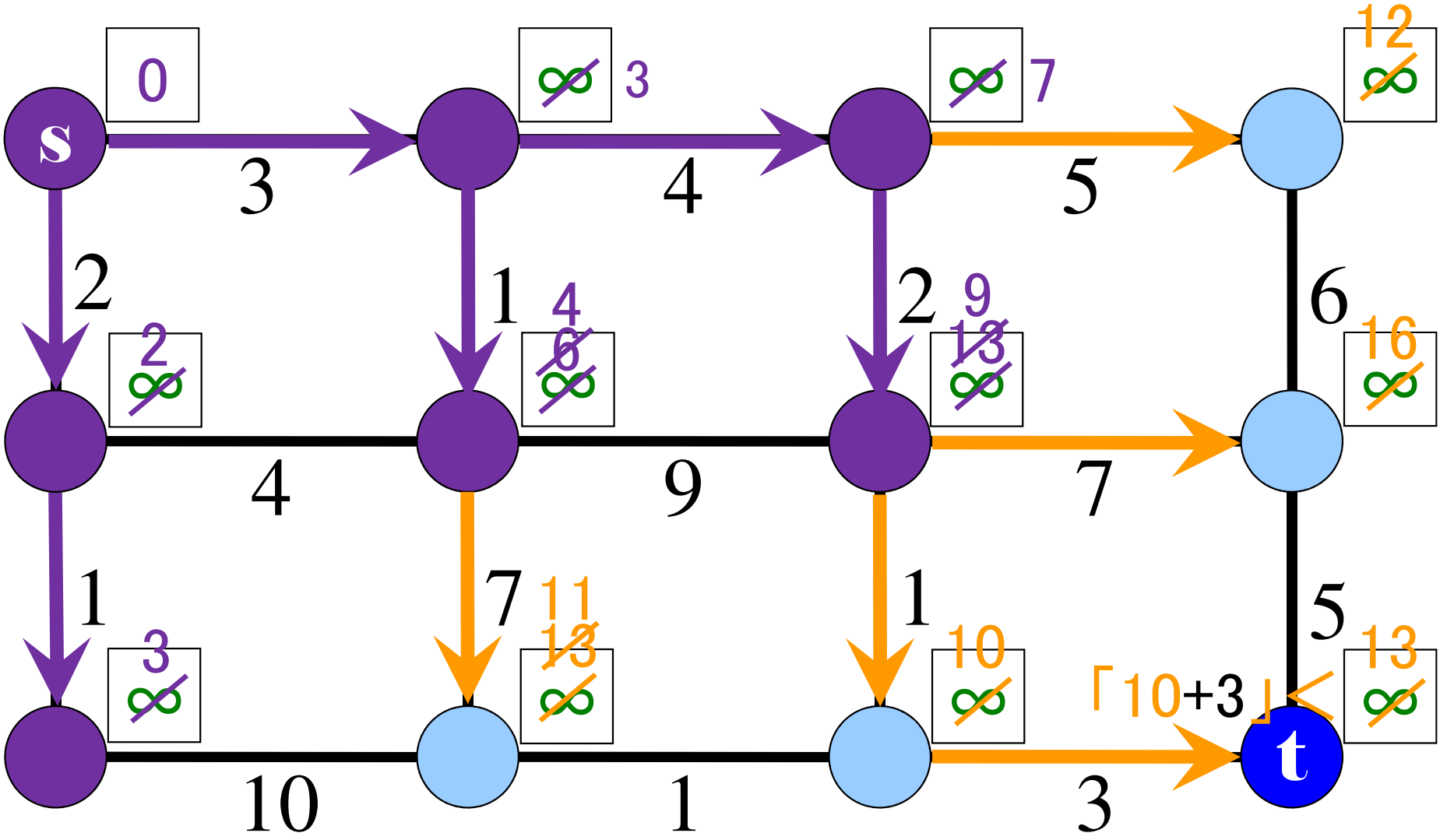
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

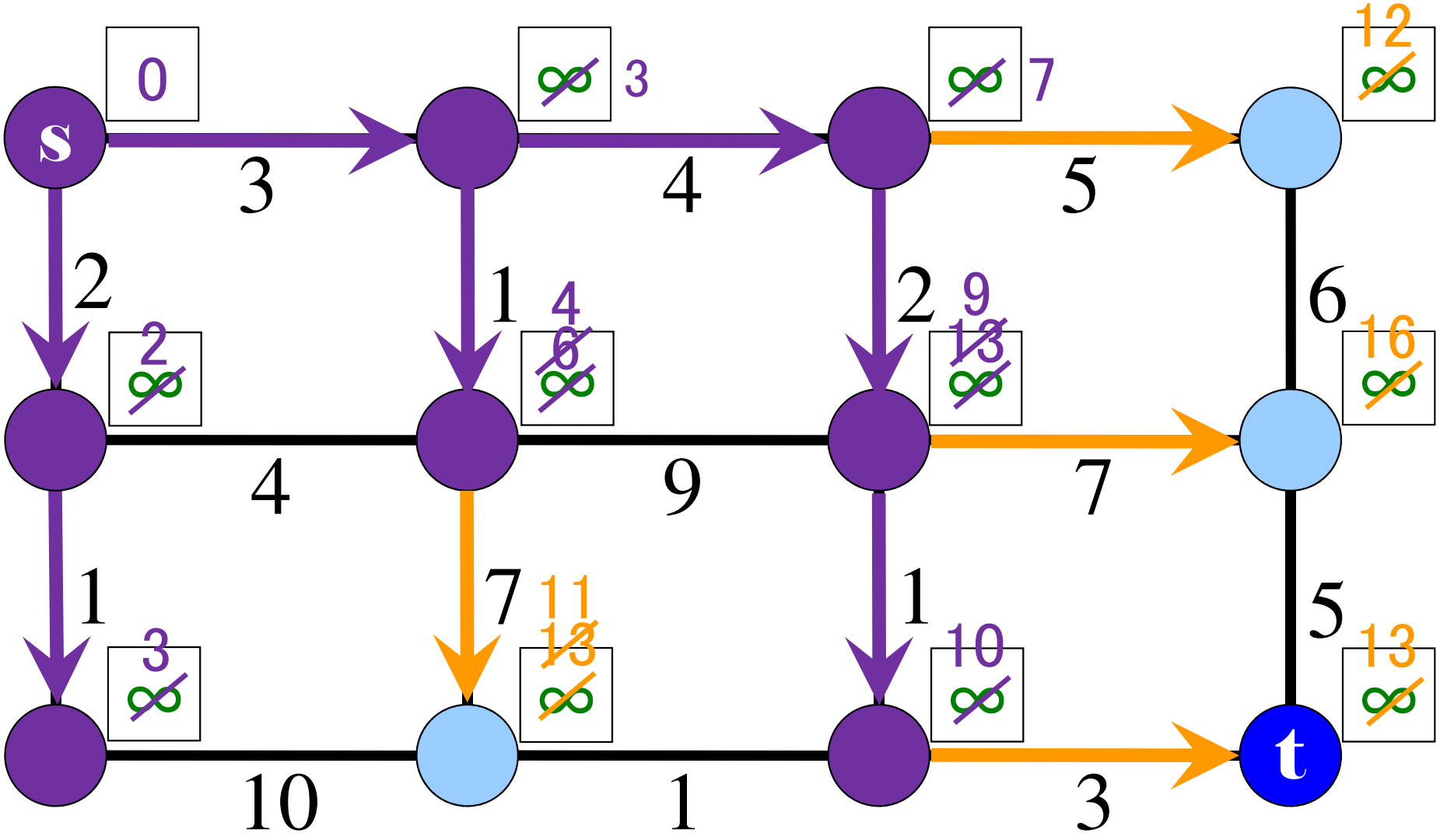


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

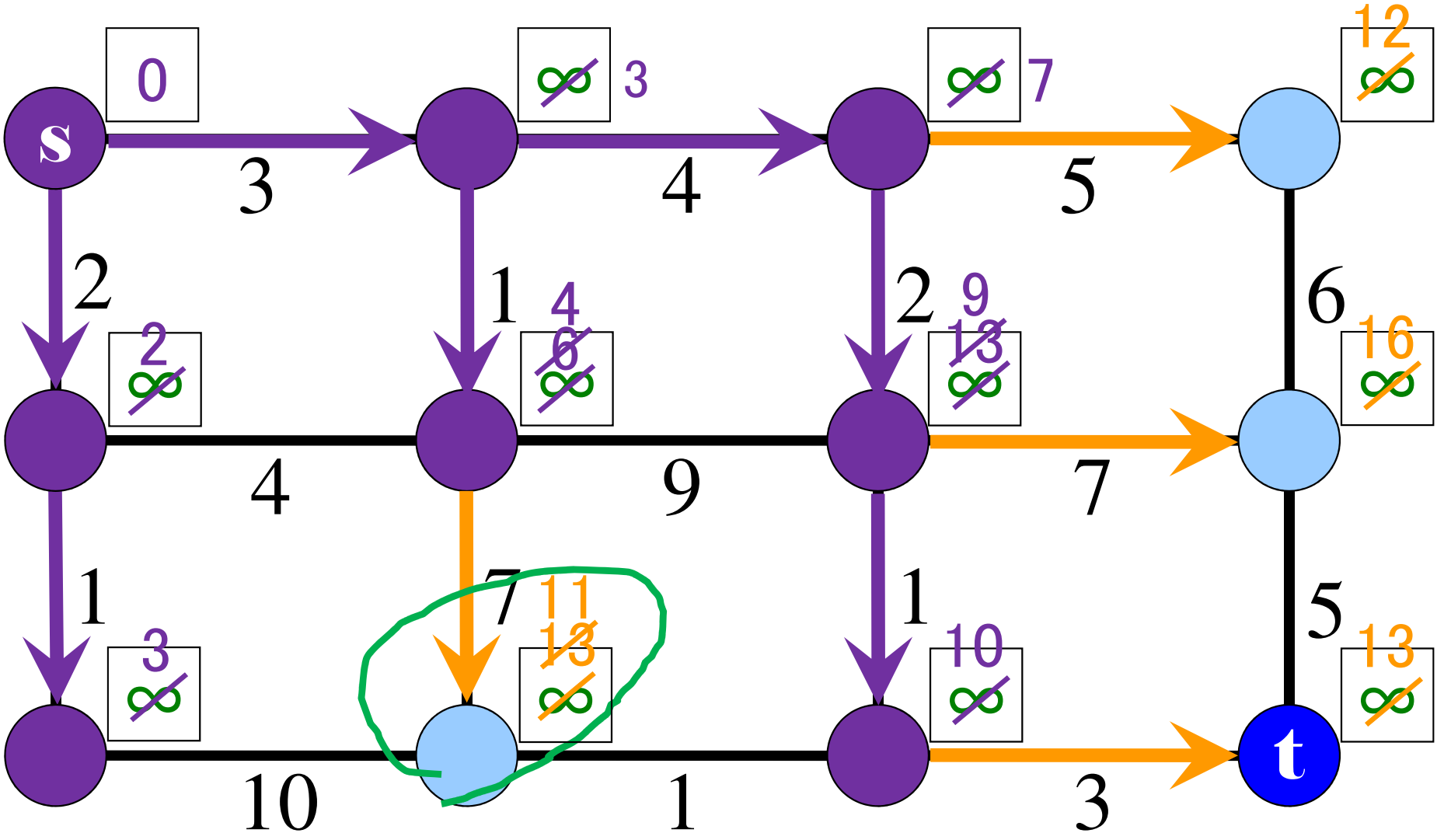
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



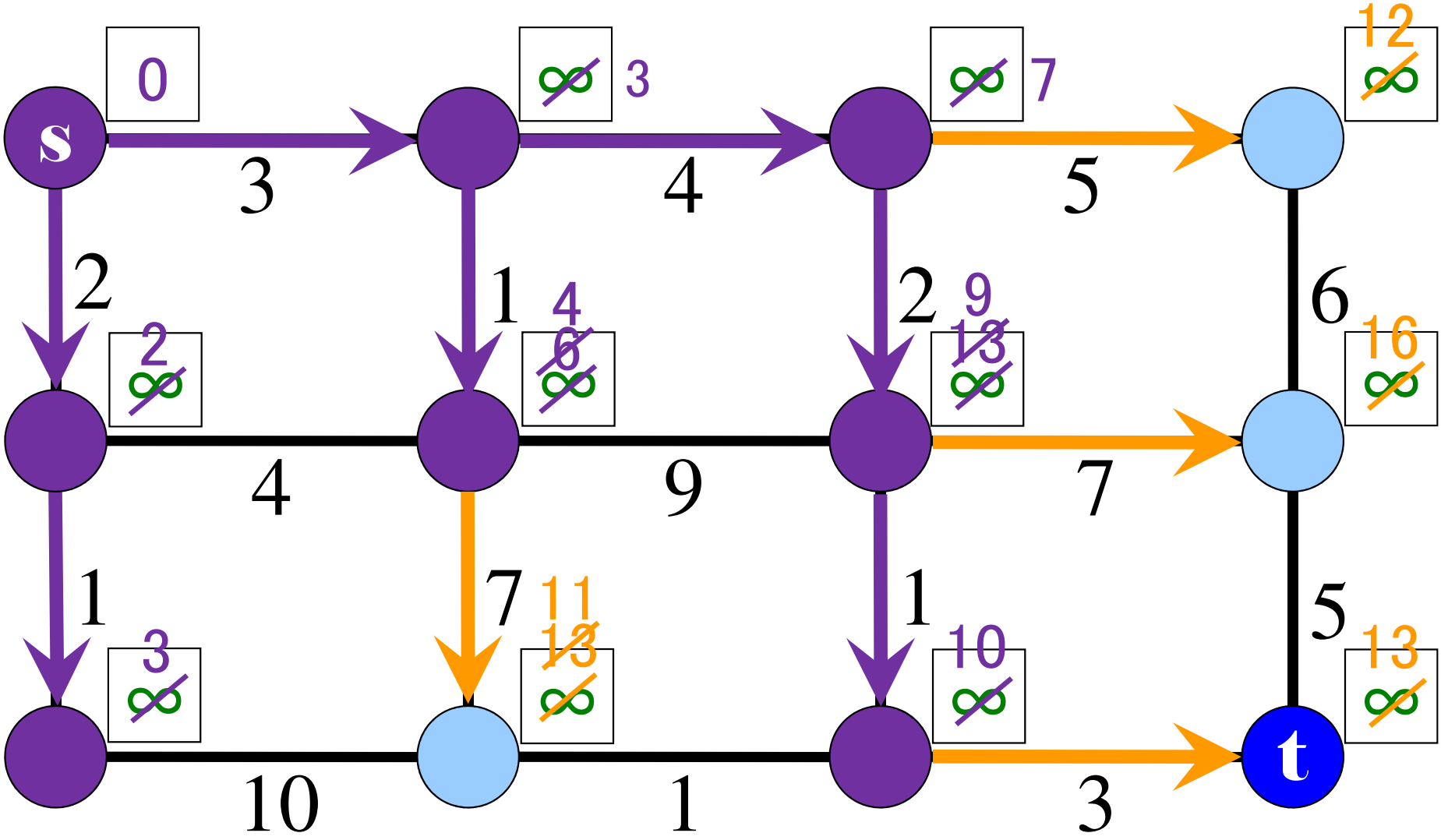
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

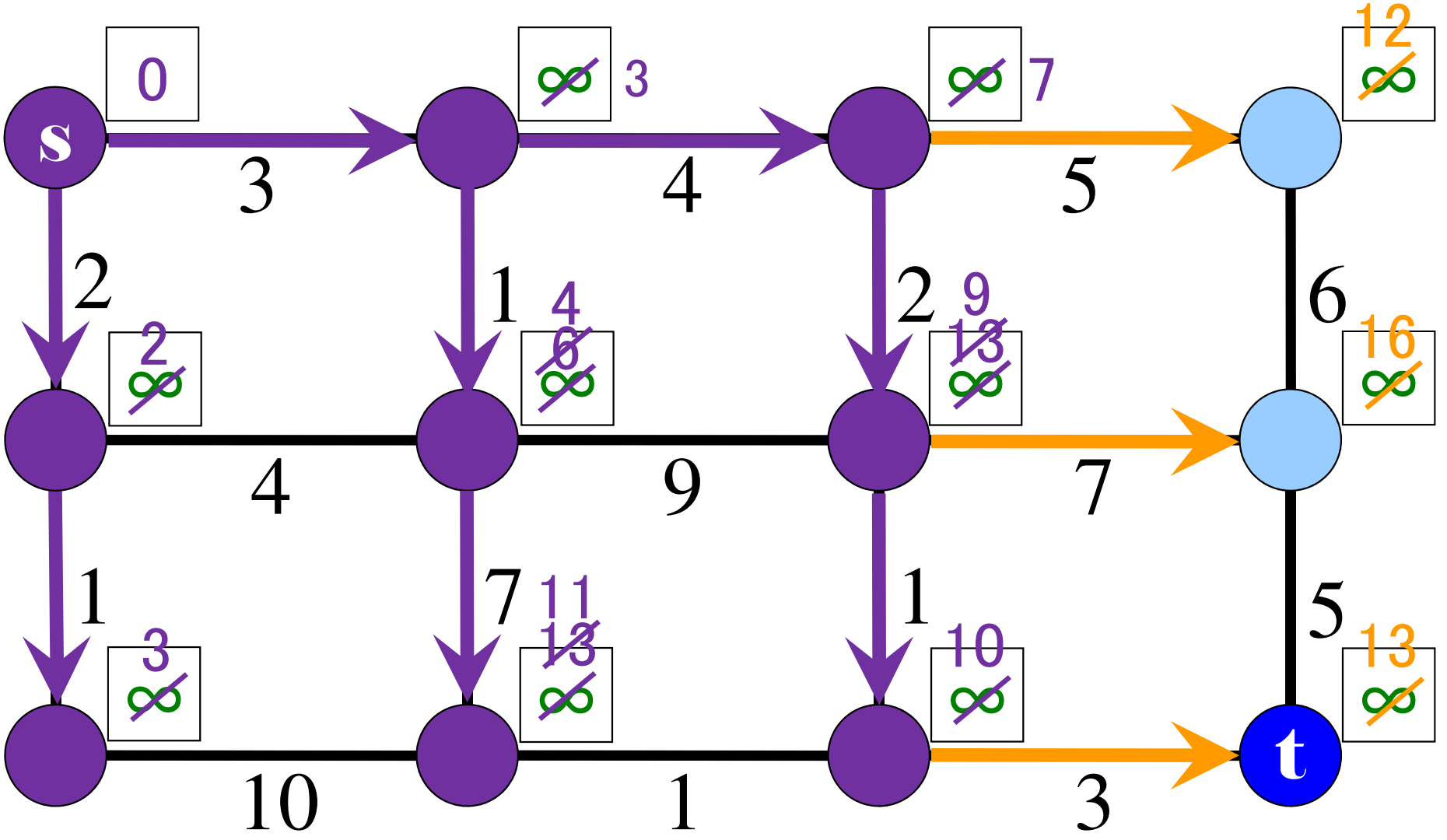


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

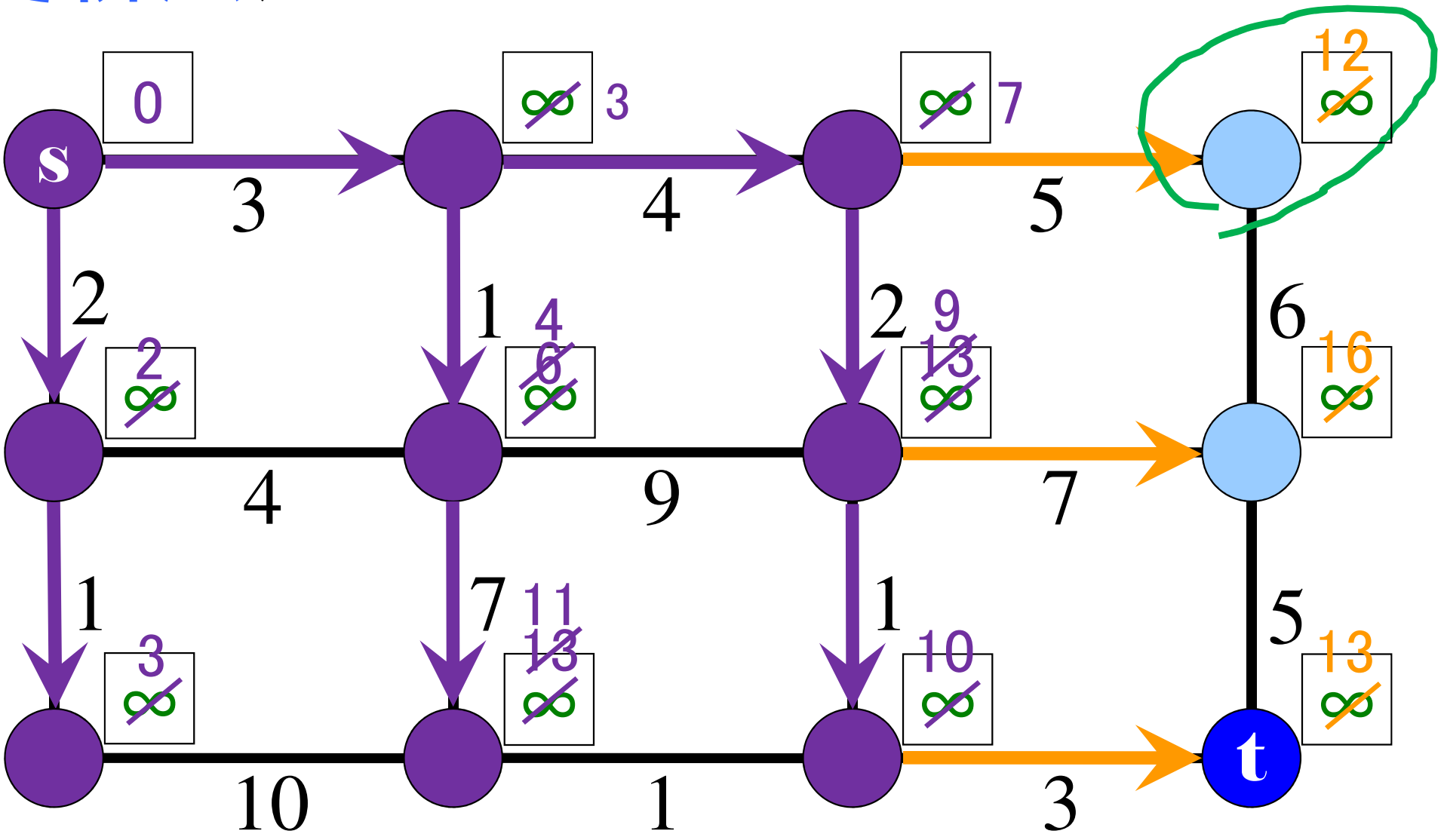
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



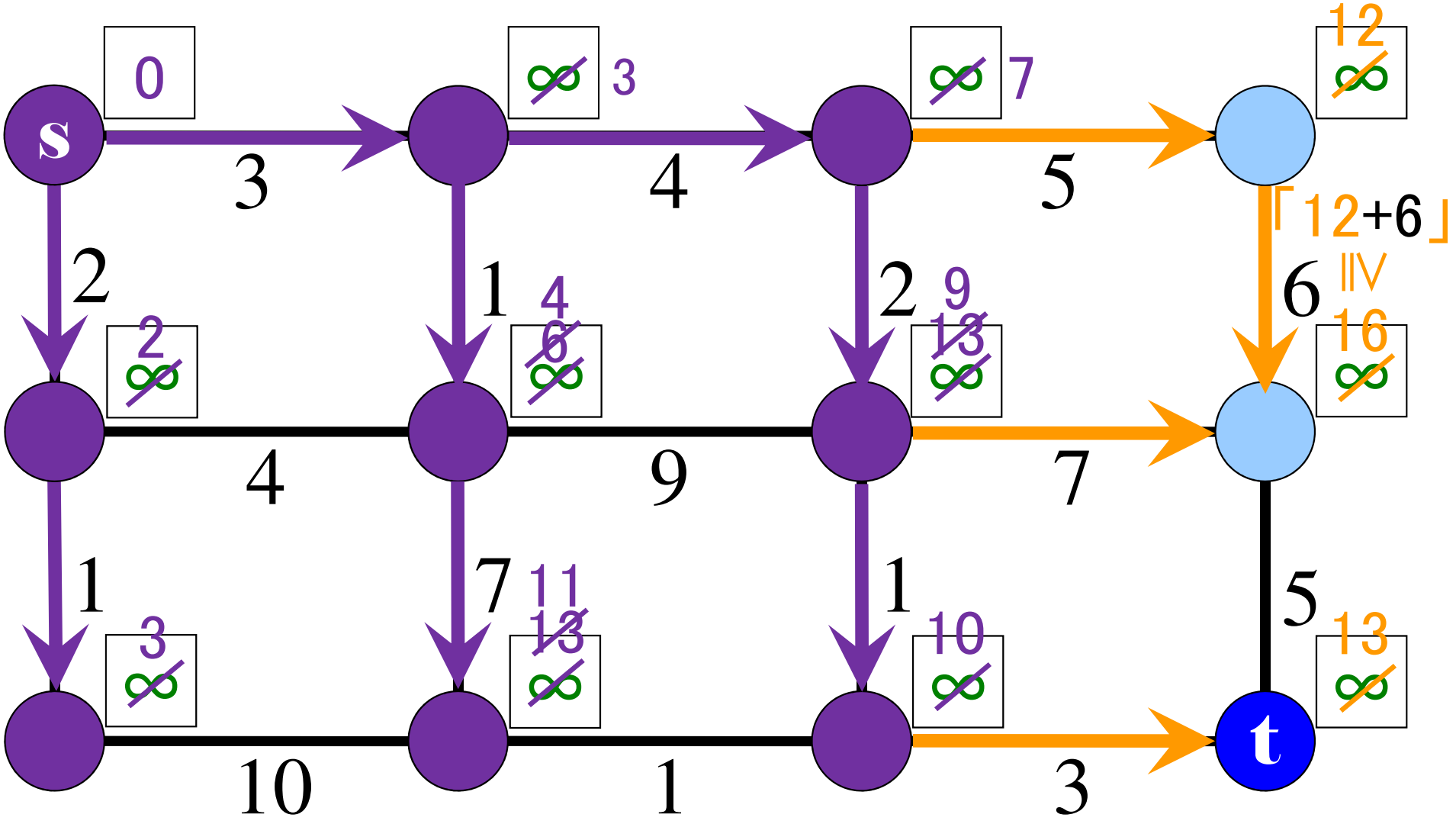
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

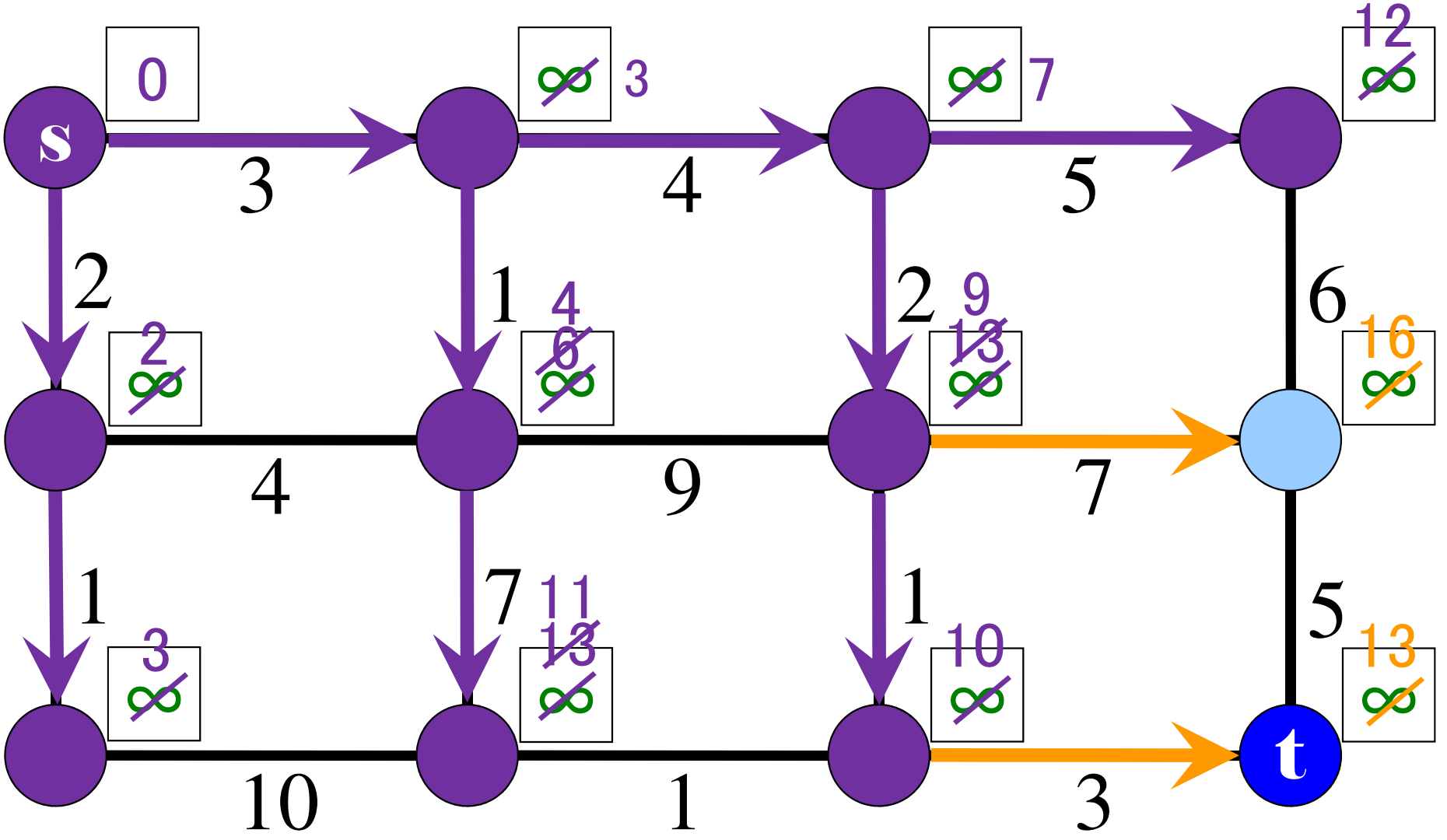


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

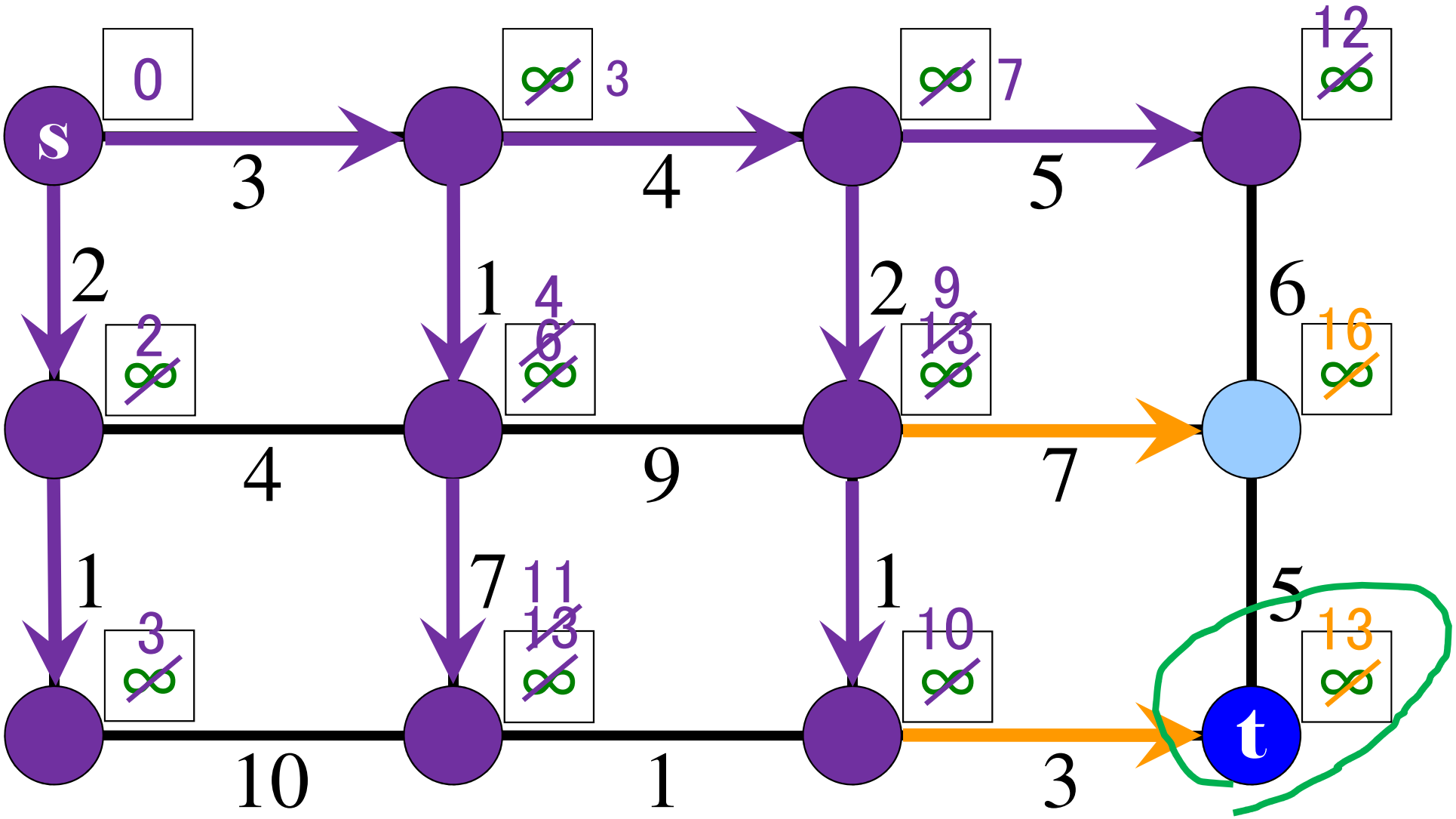
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



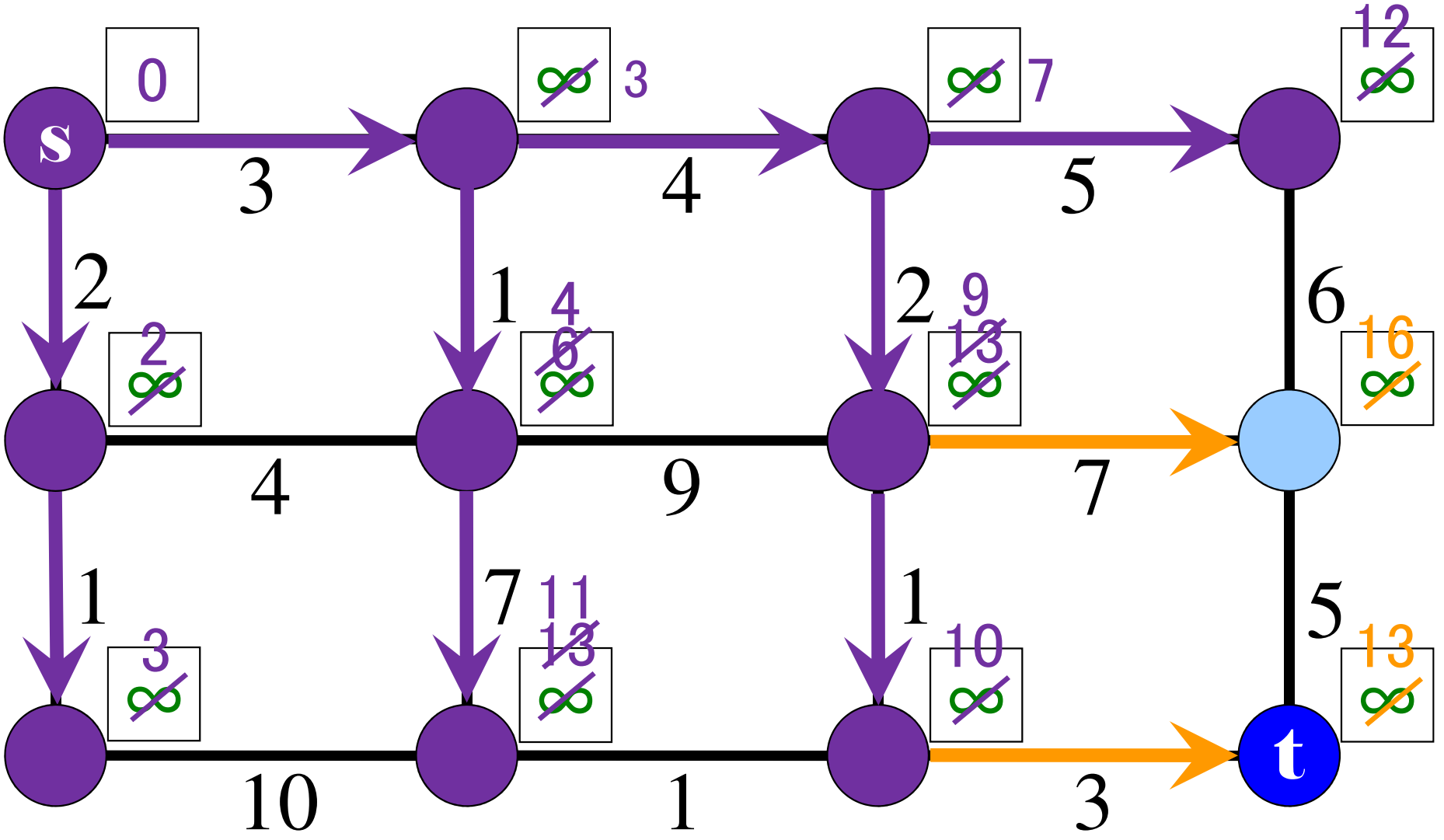
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

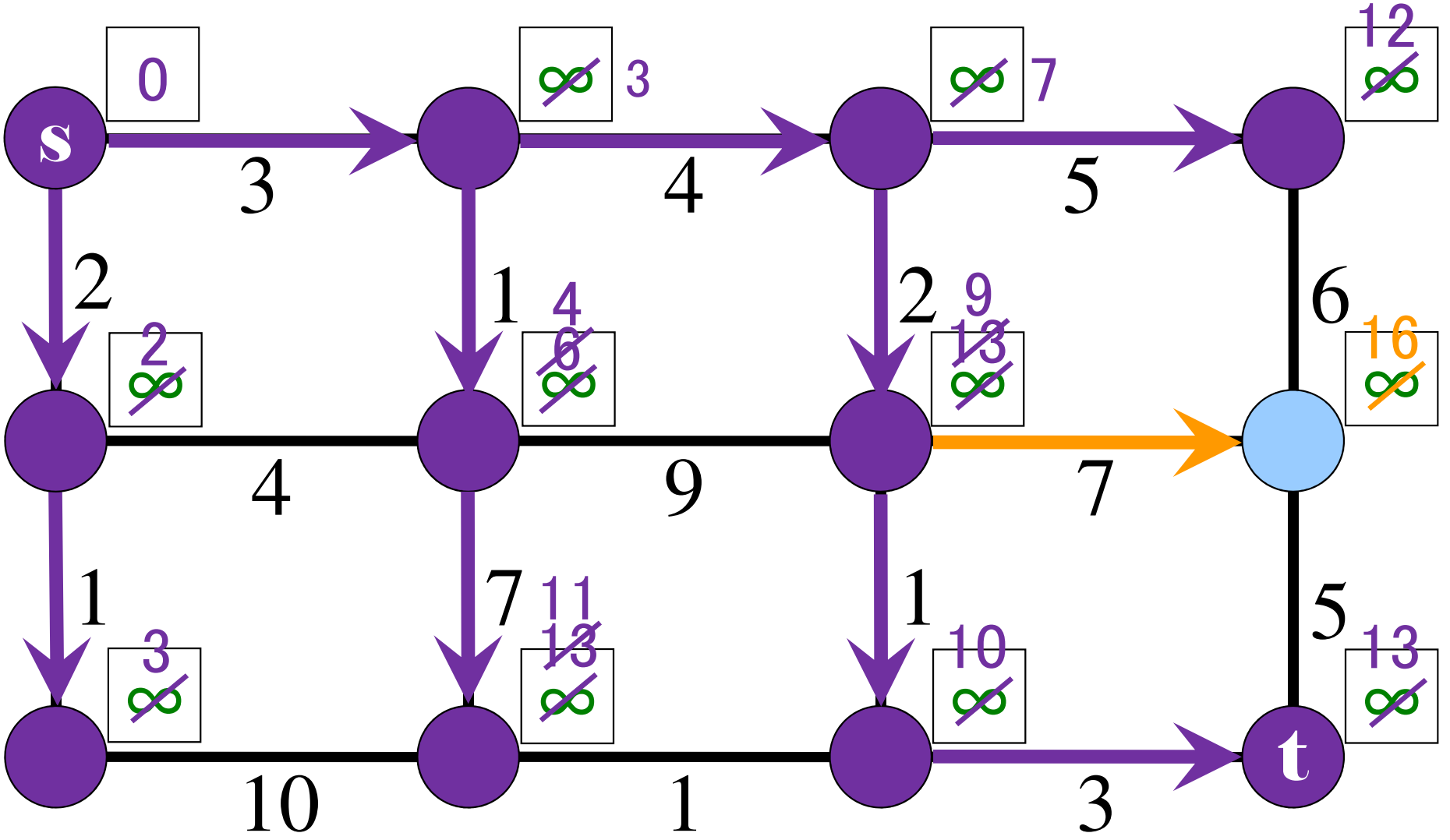


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

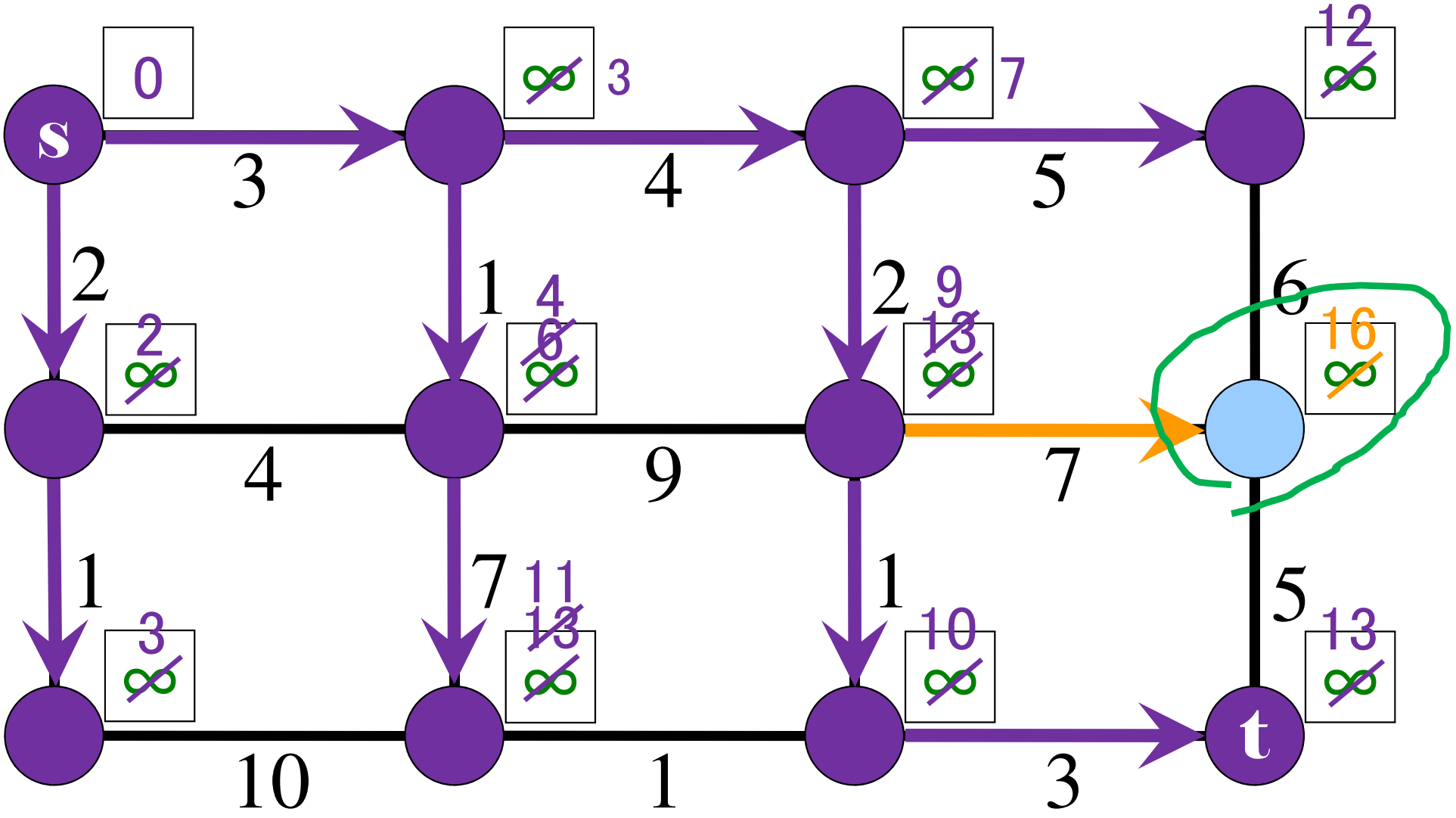
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



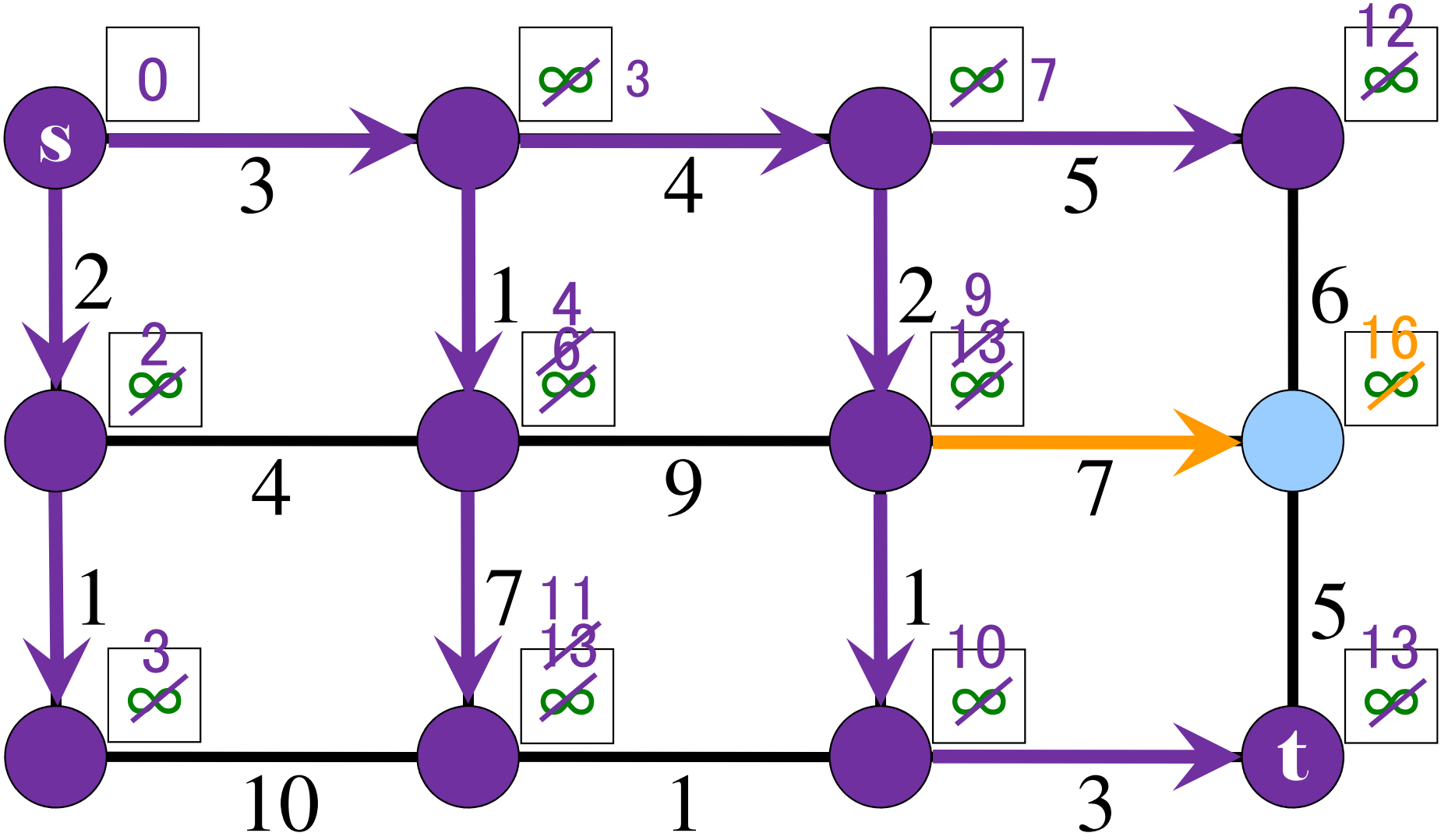
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

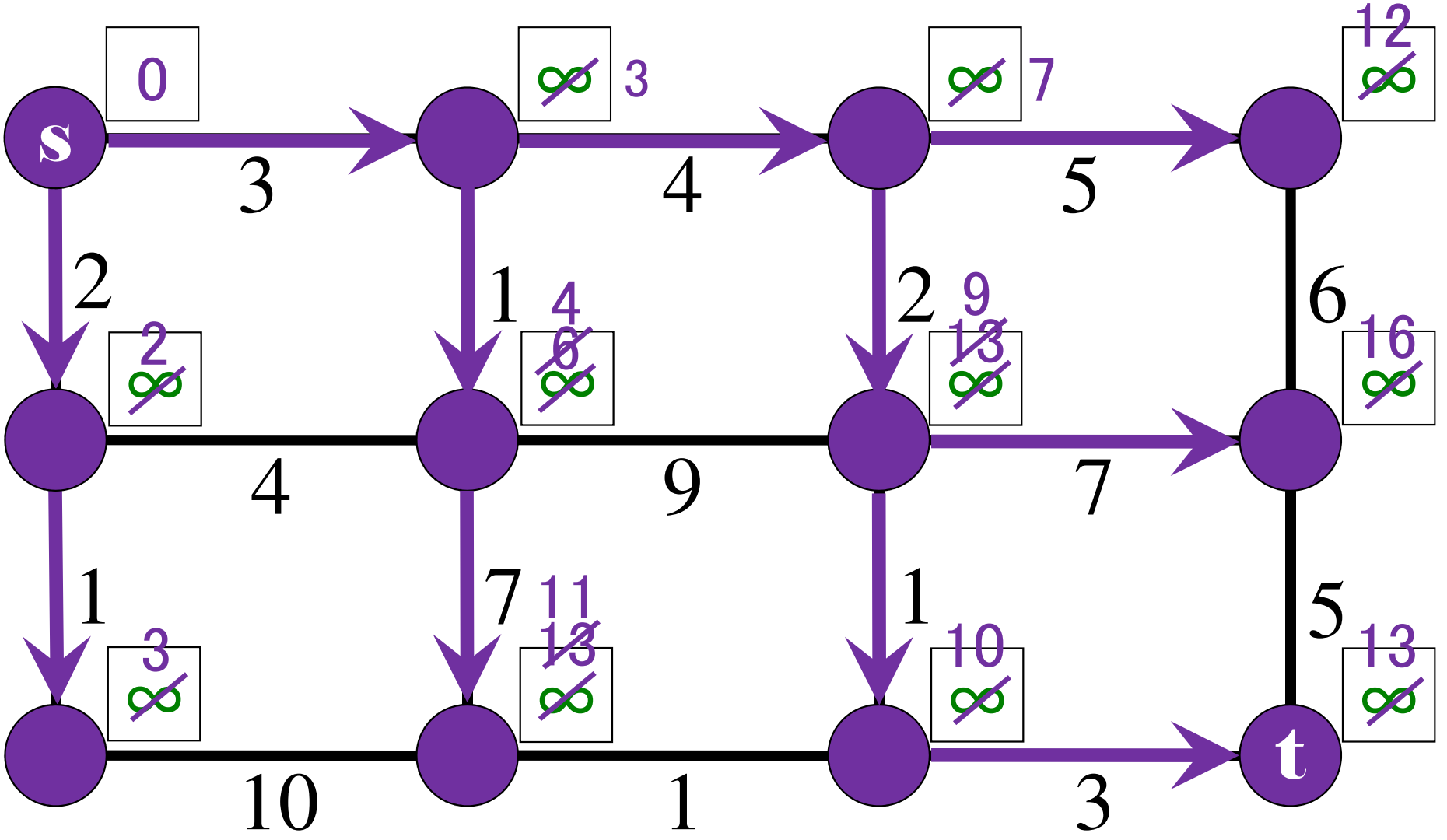


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

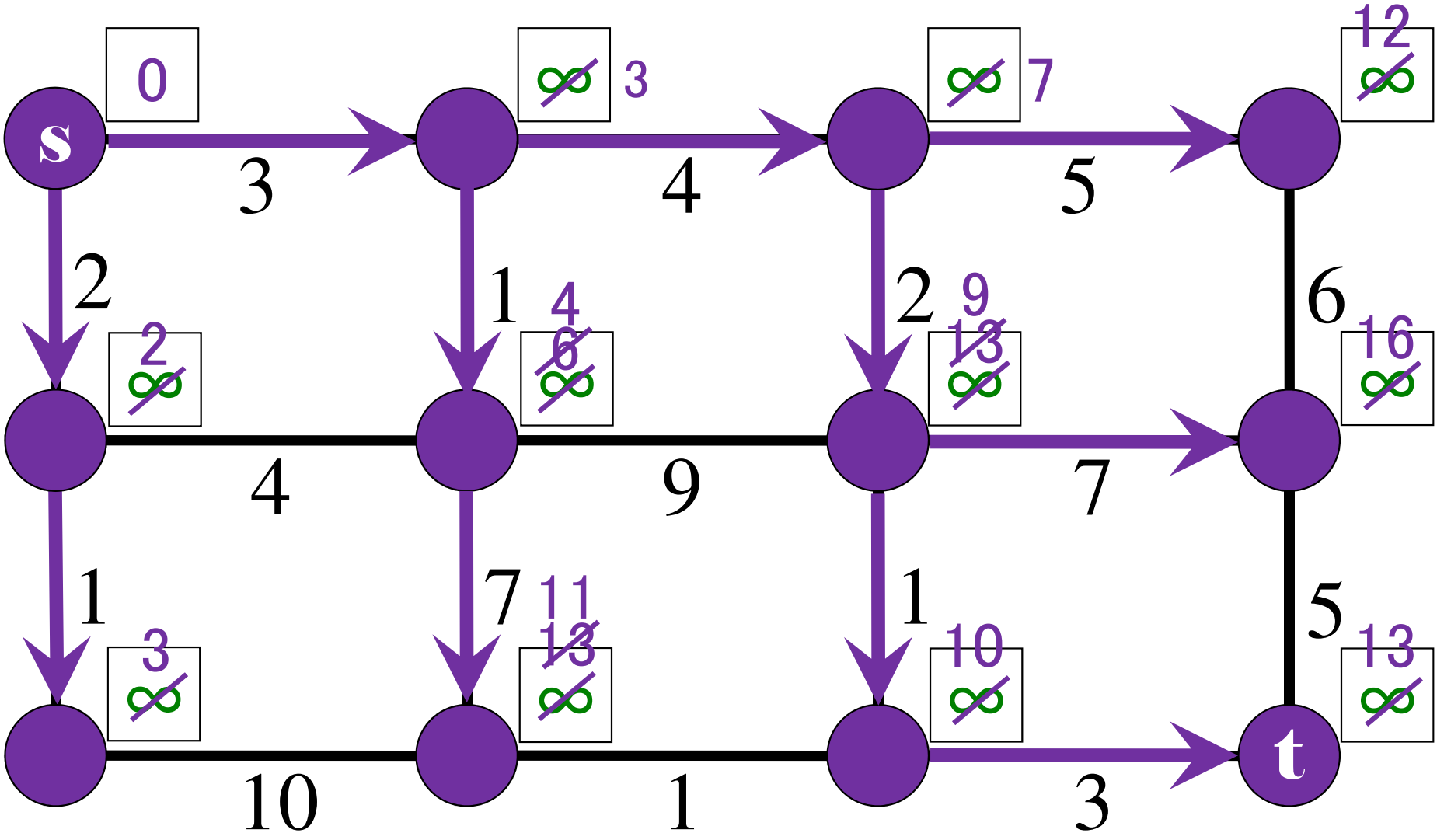
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



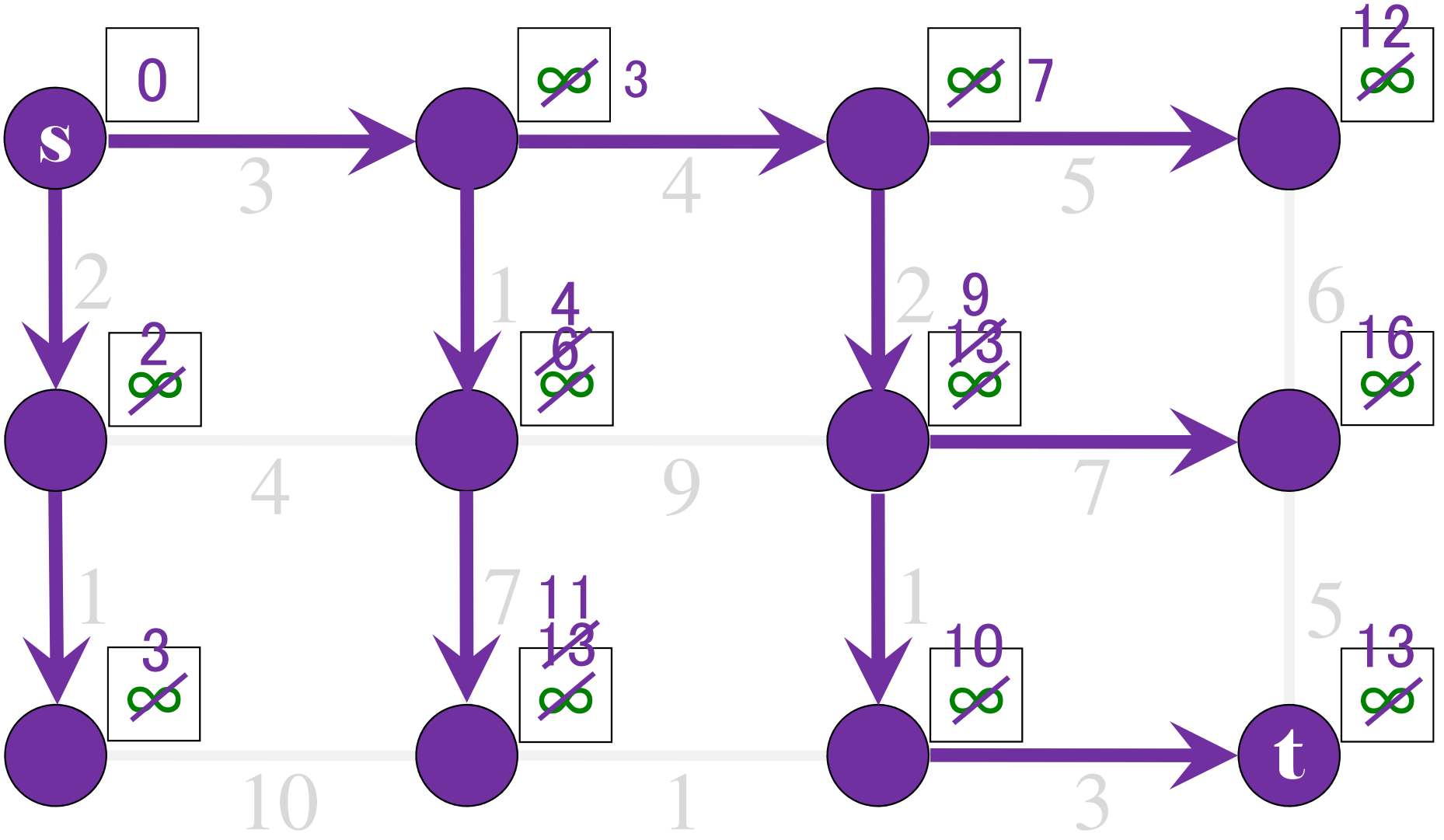
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



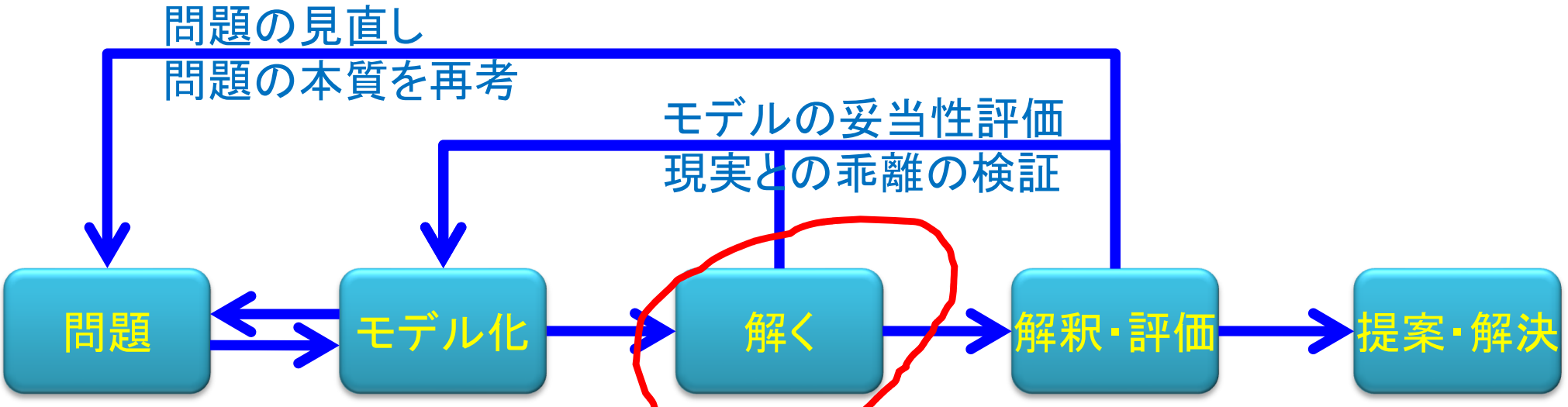
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

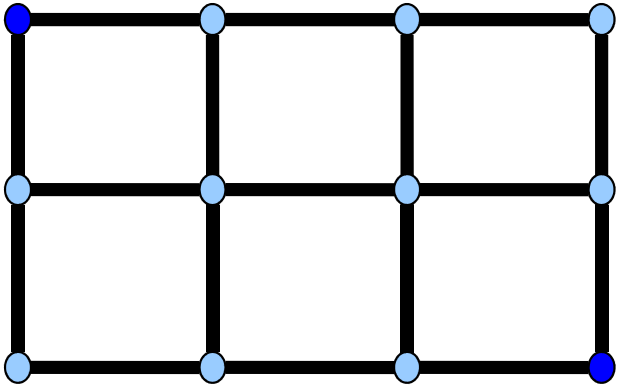
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

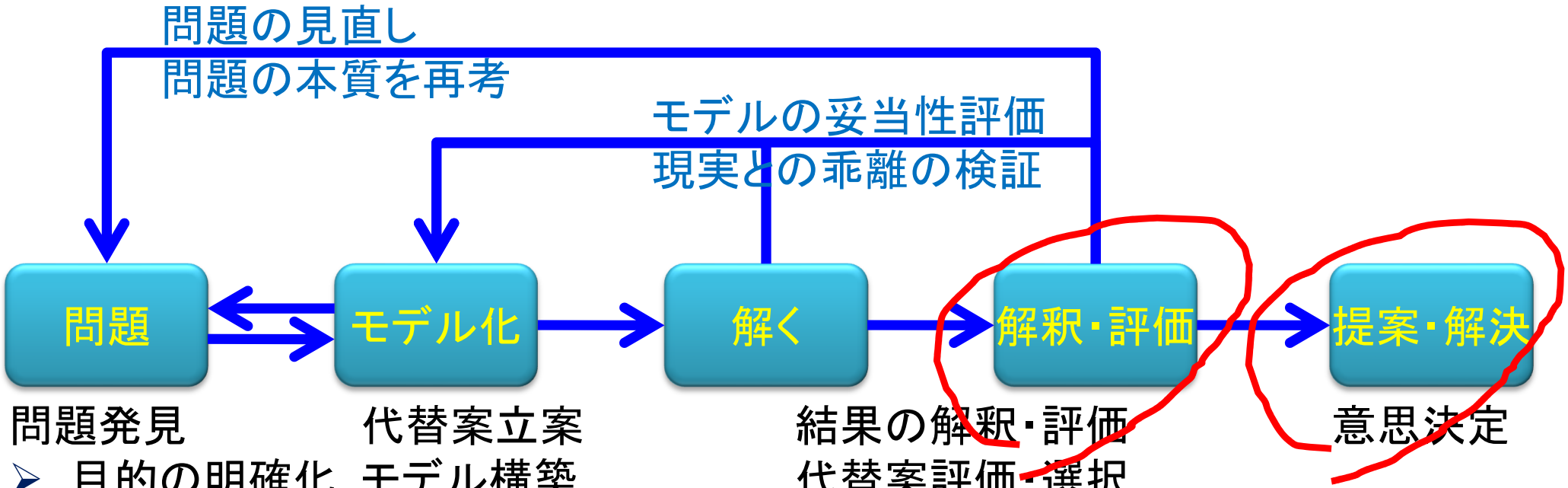
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

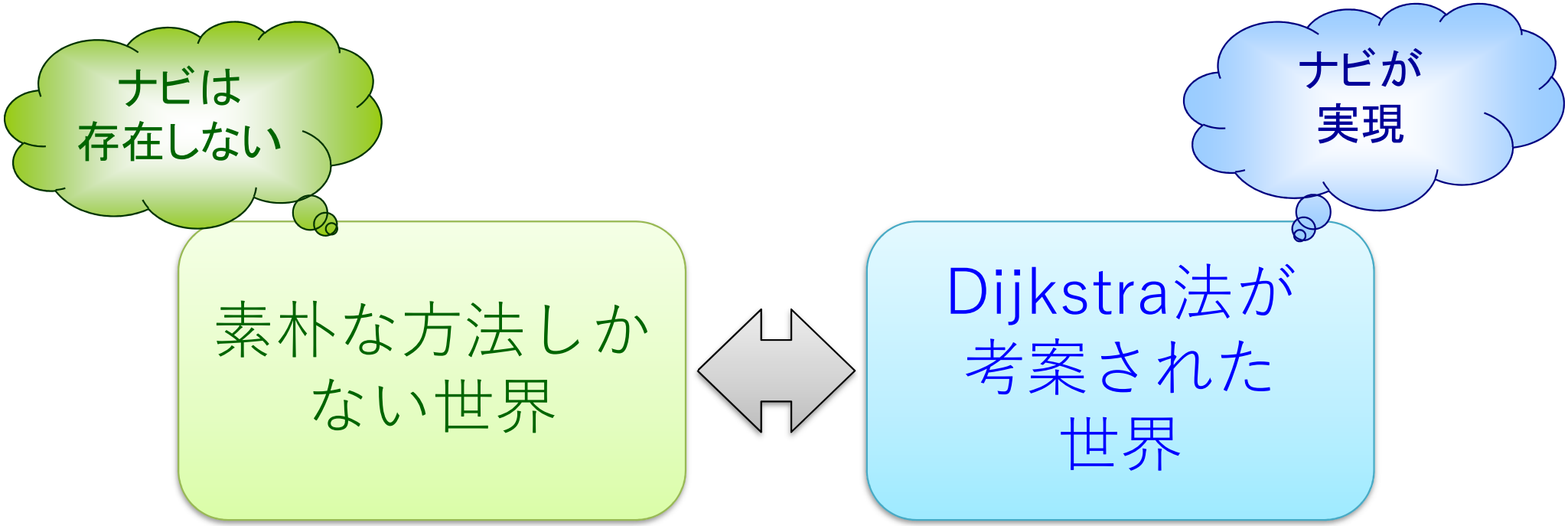
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

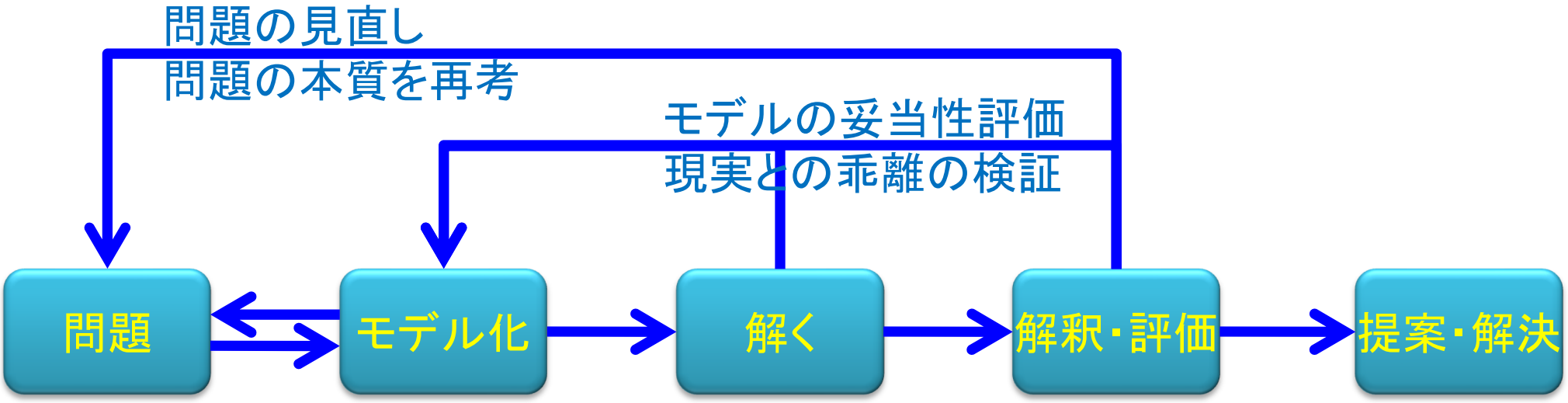
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



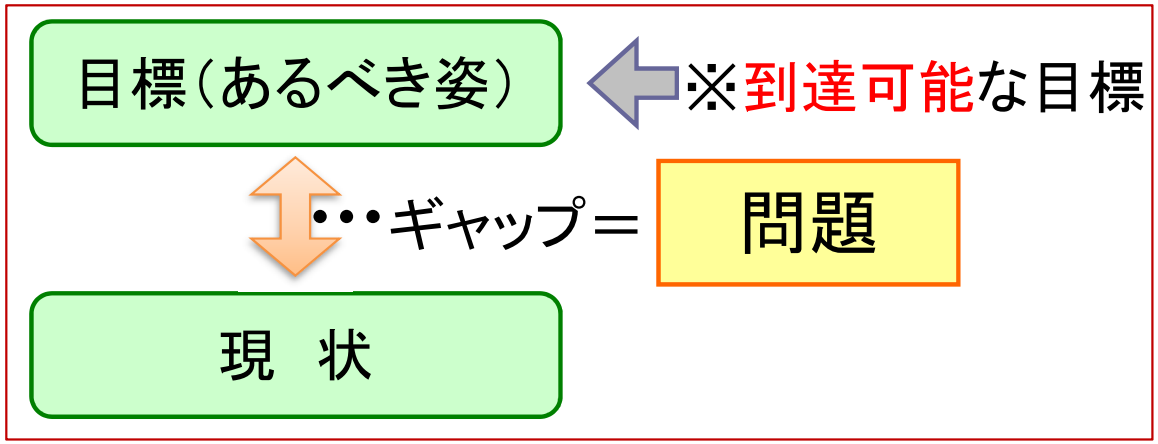
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

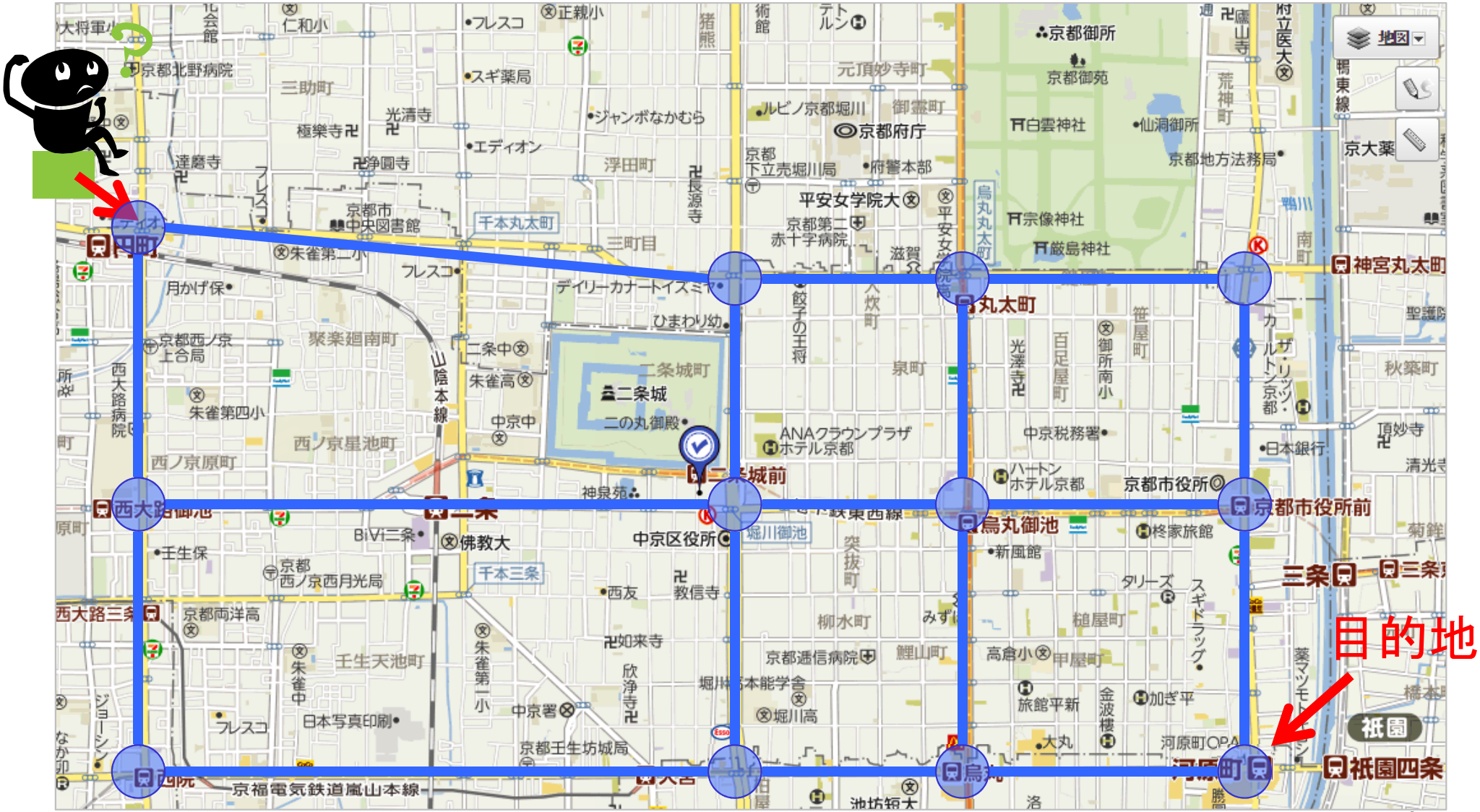
意思決定

問題の定義



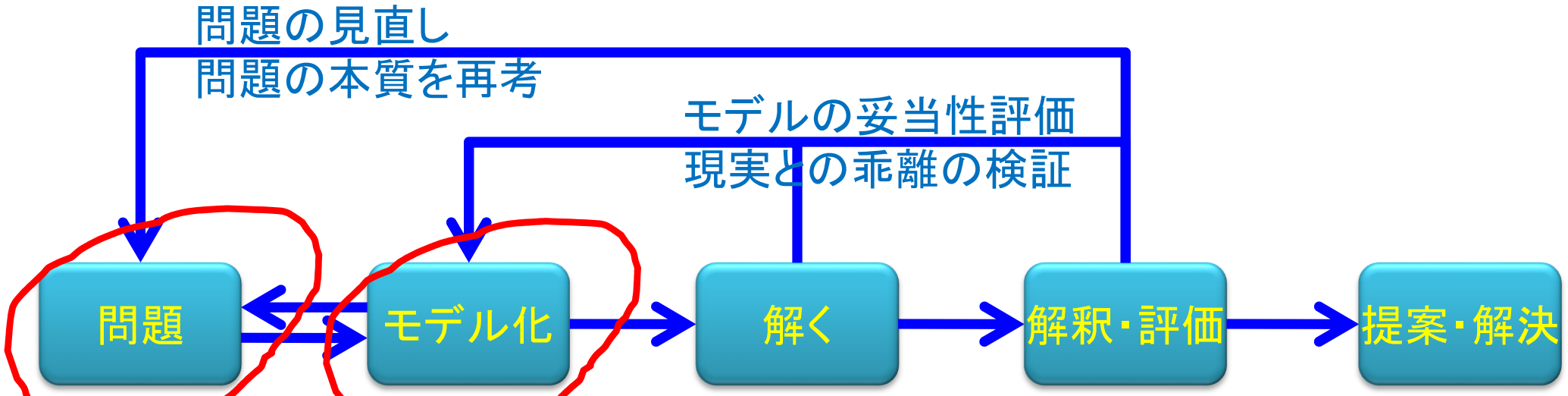
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

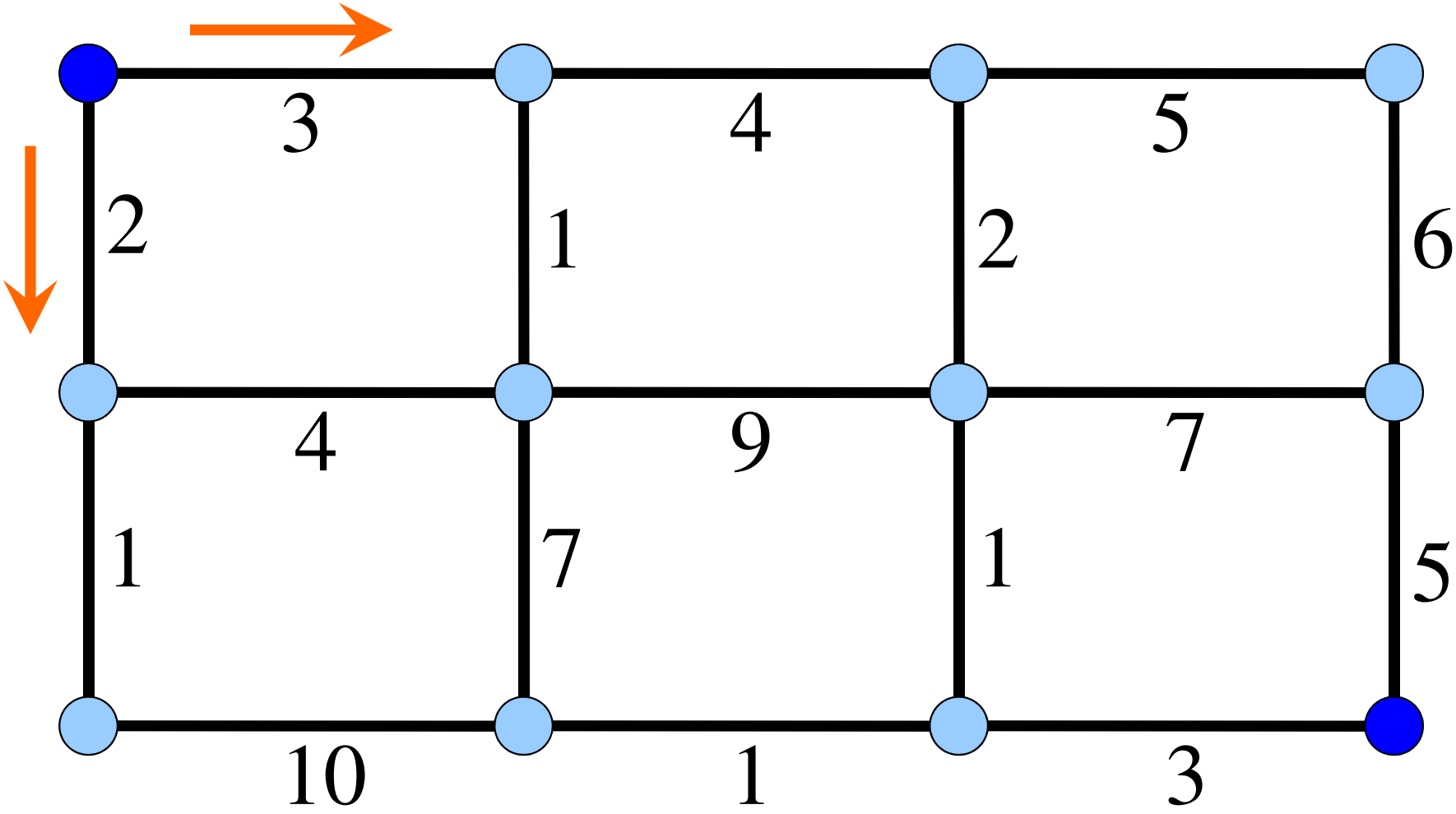
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

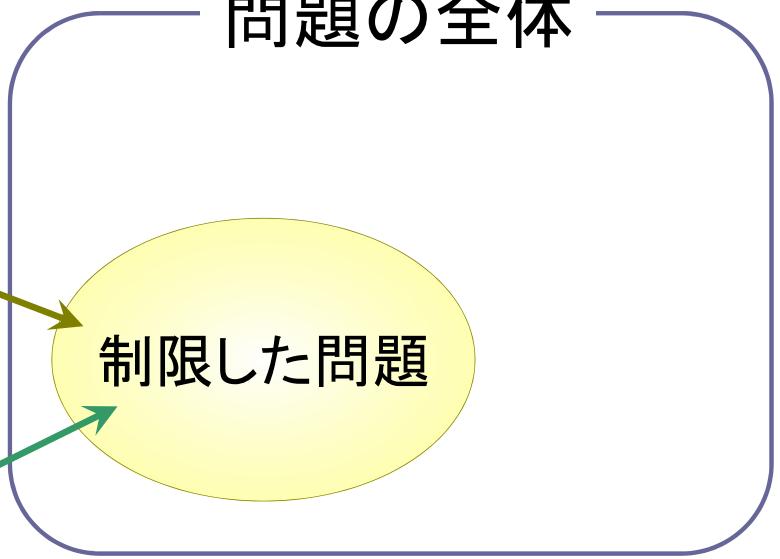


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

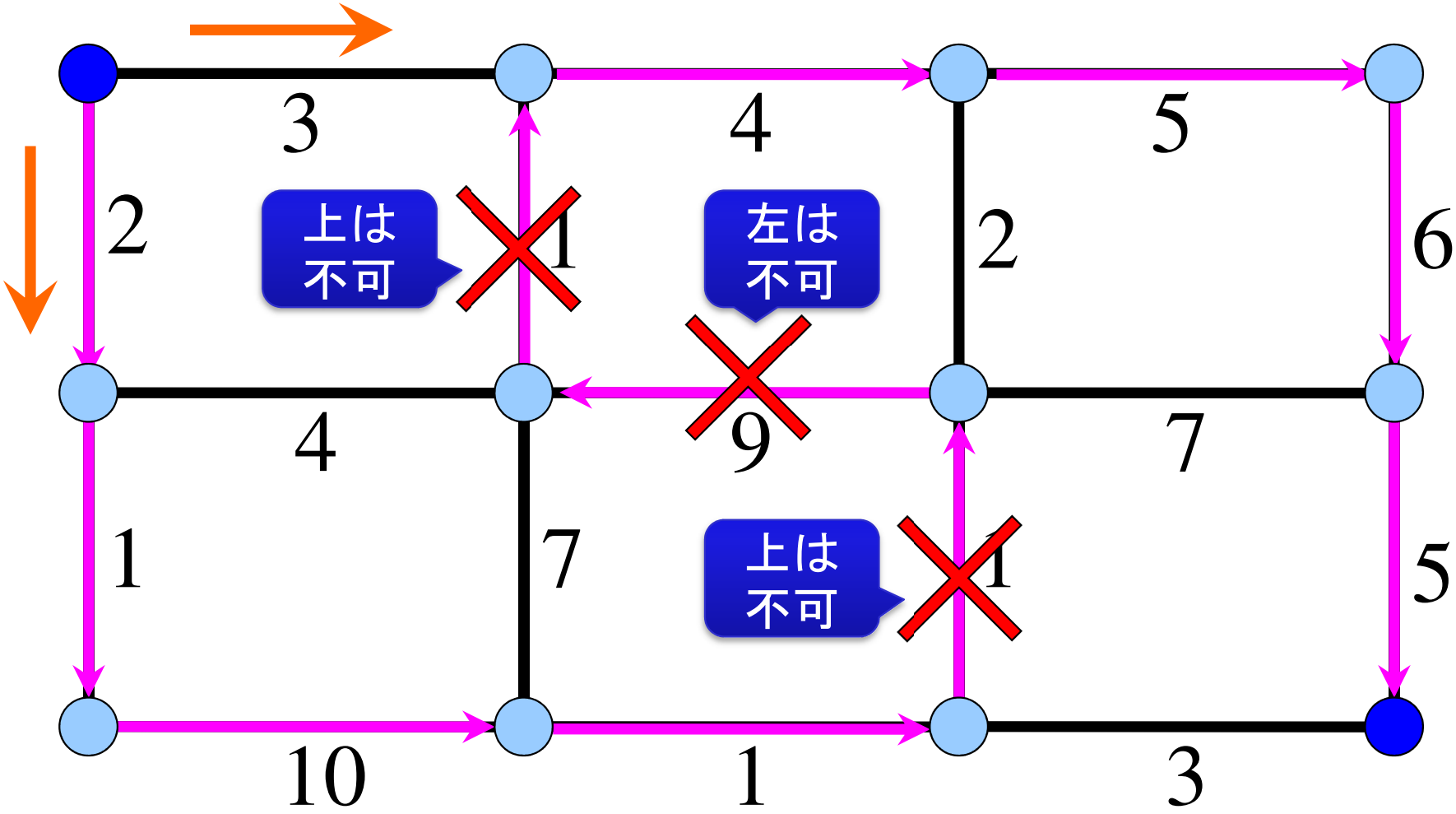
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

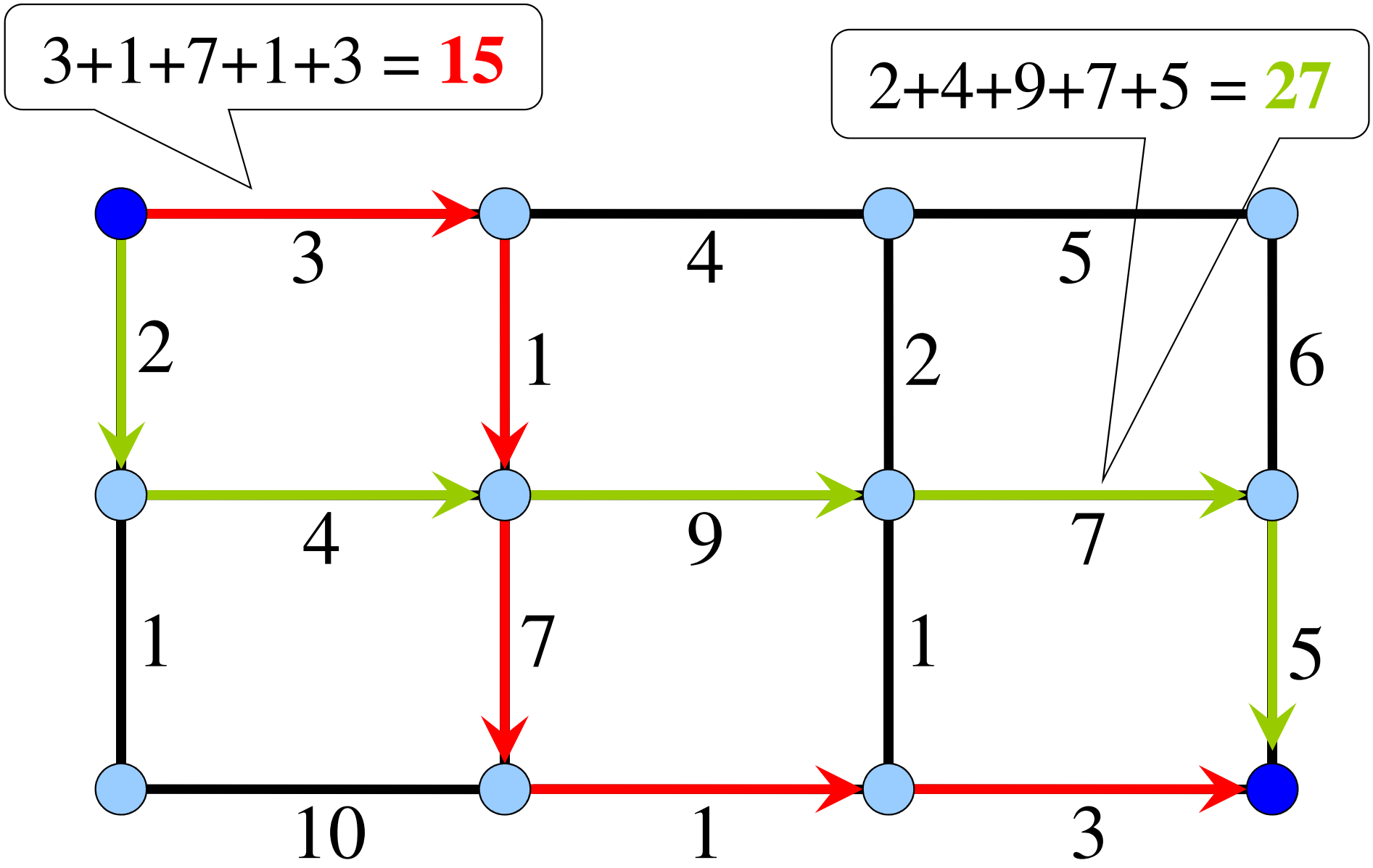
難しいなら易しくすればいいのさ！

制限した問題

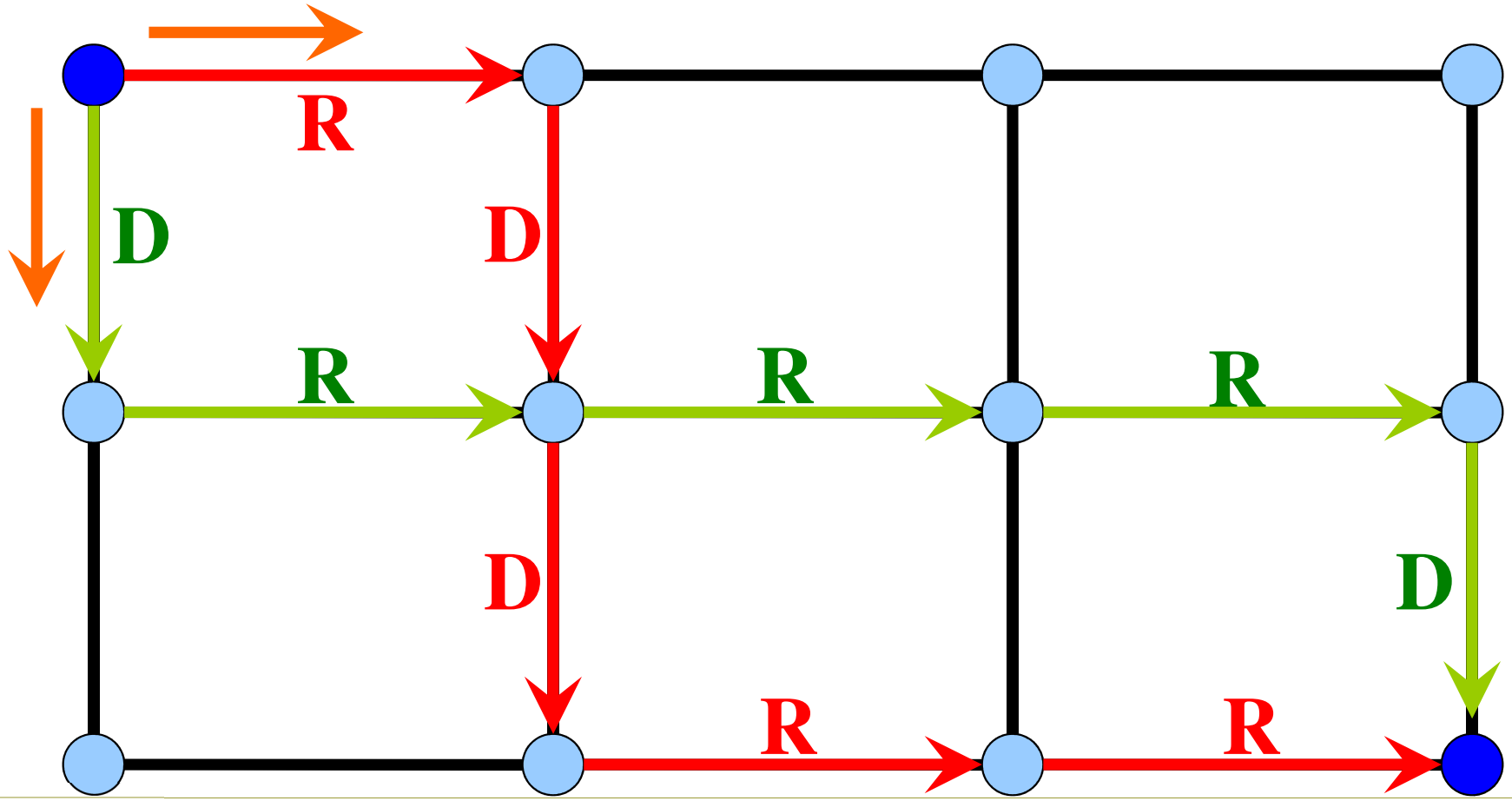
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



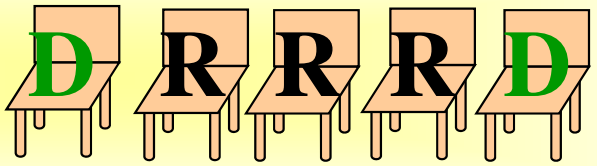
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



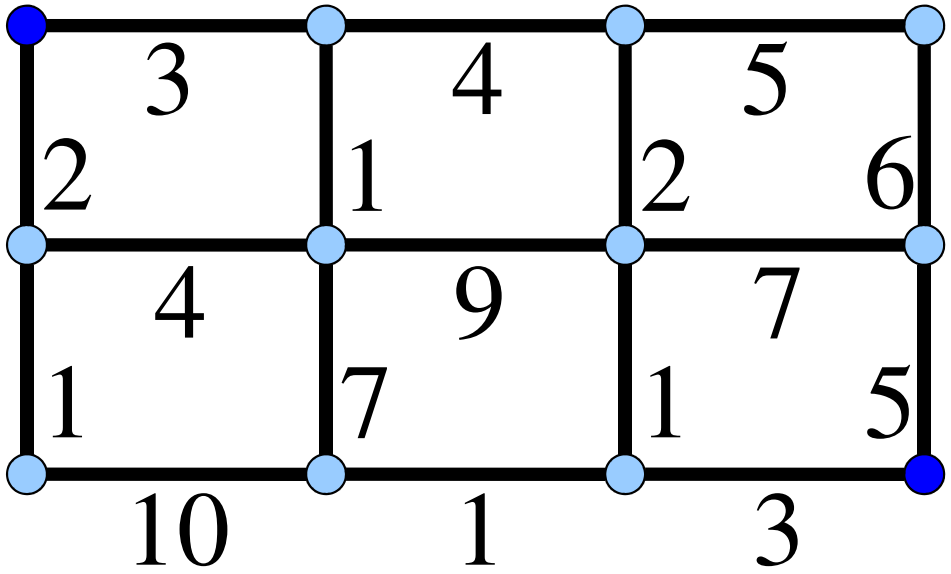
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

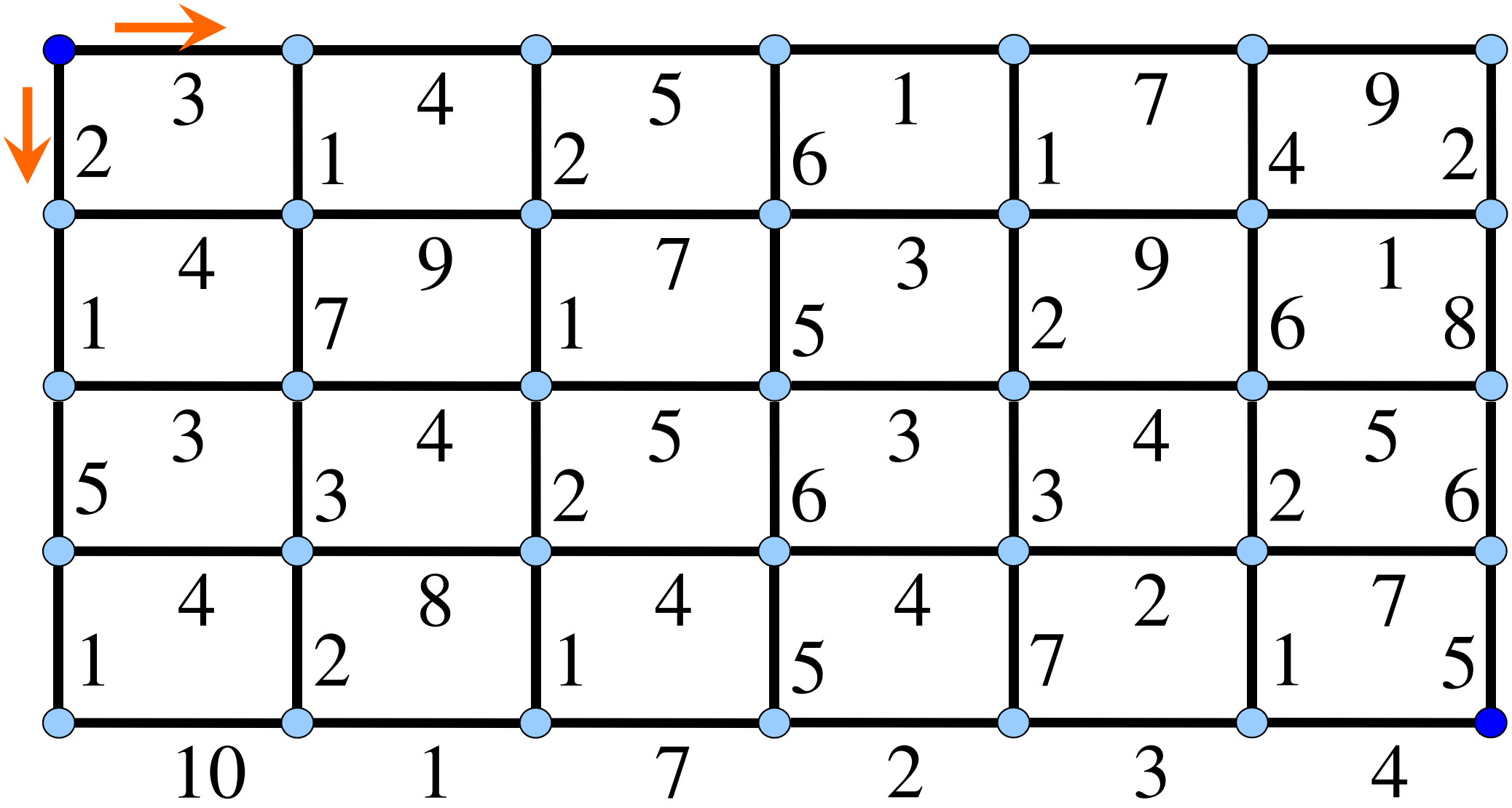
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

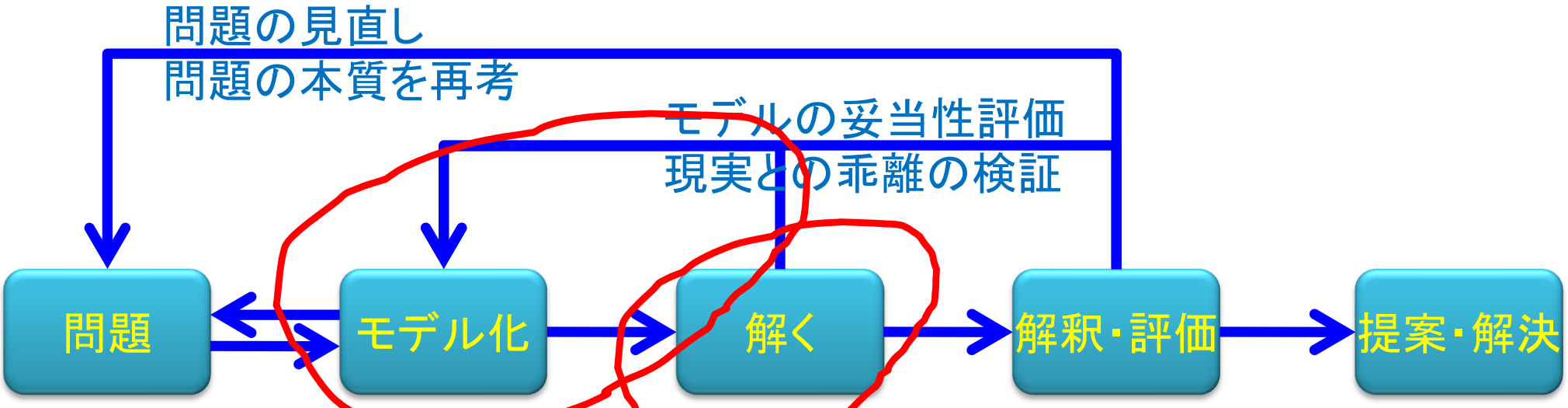
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

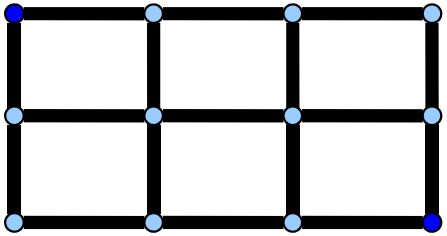
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

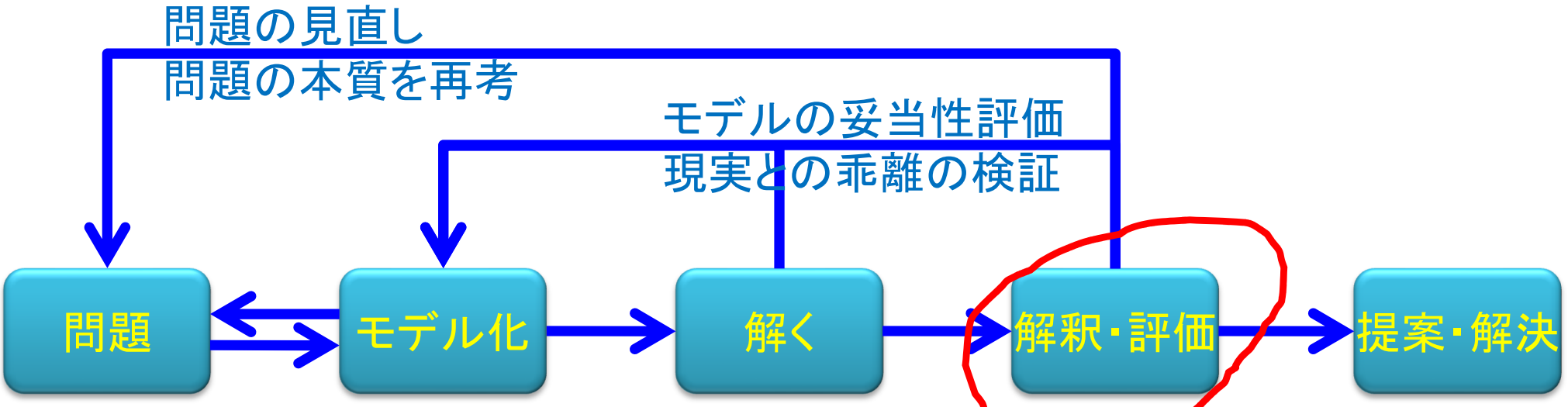
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

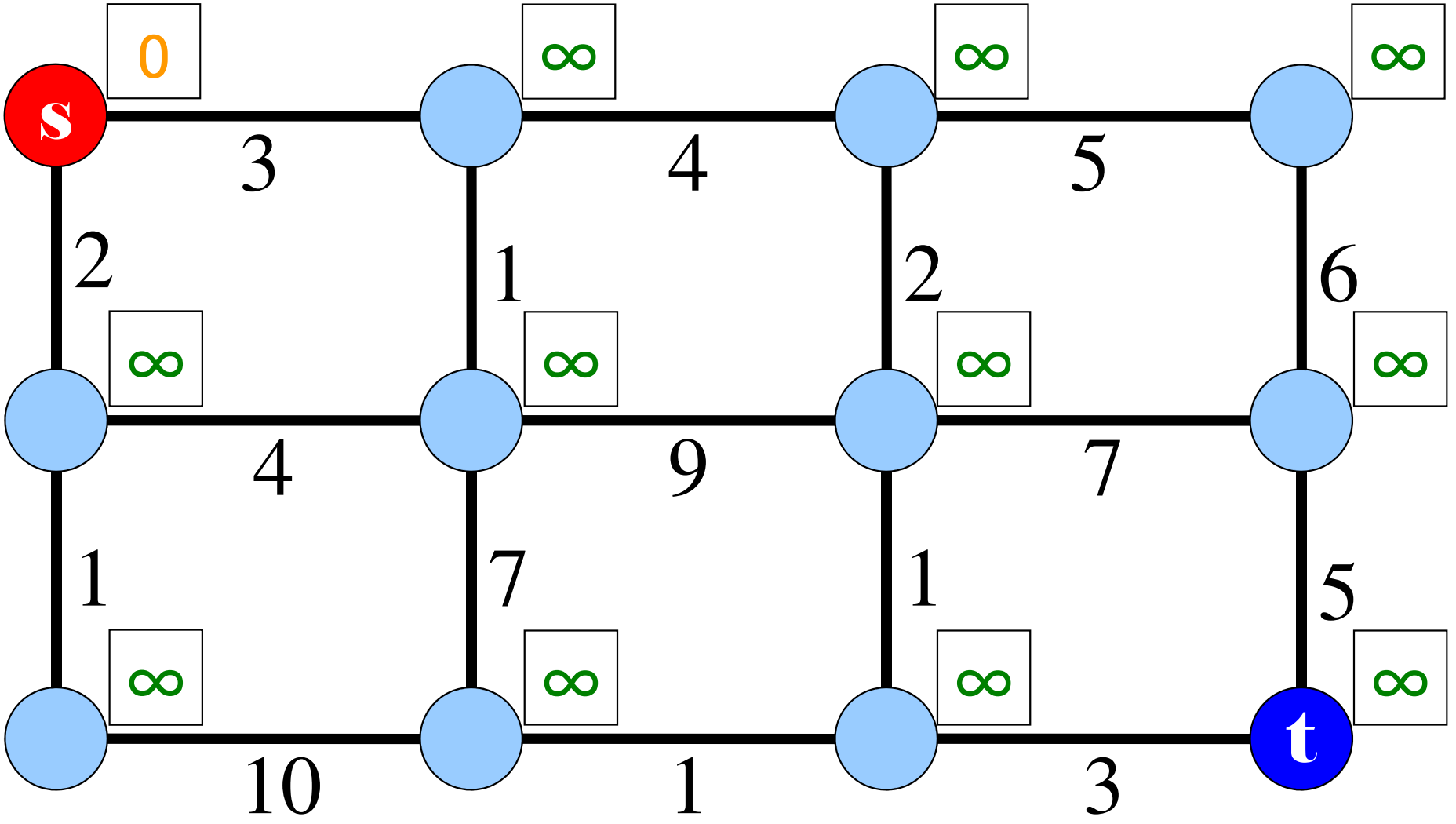
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

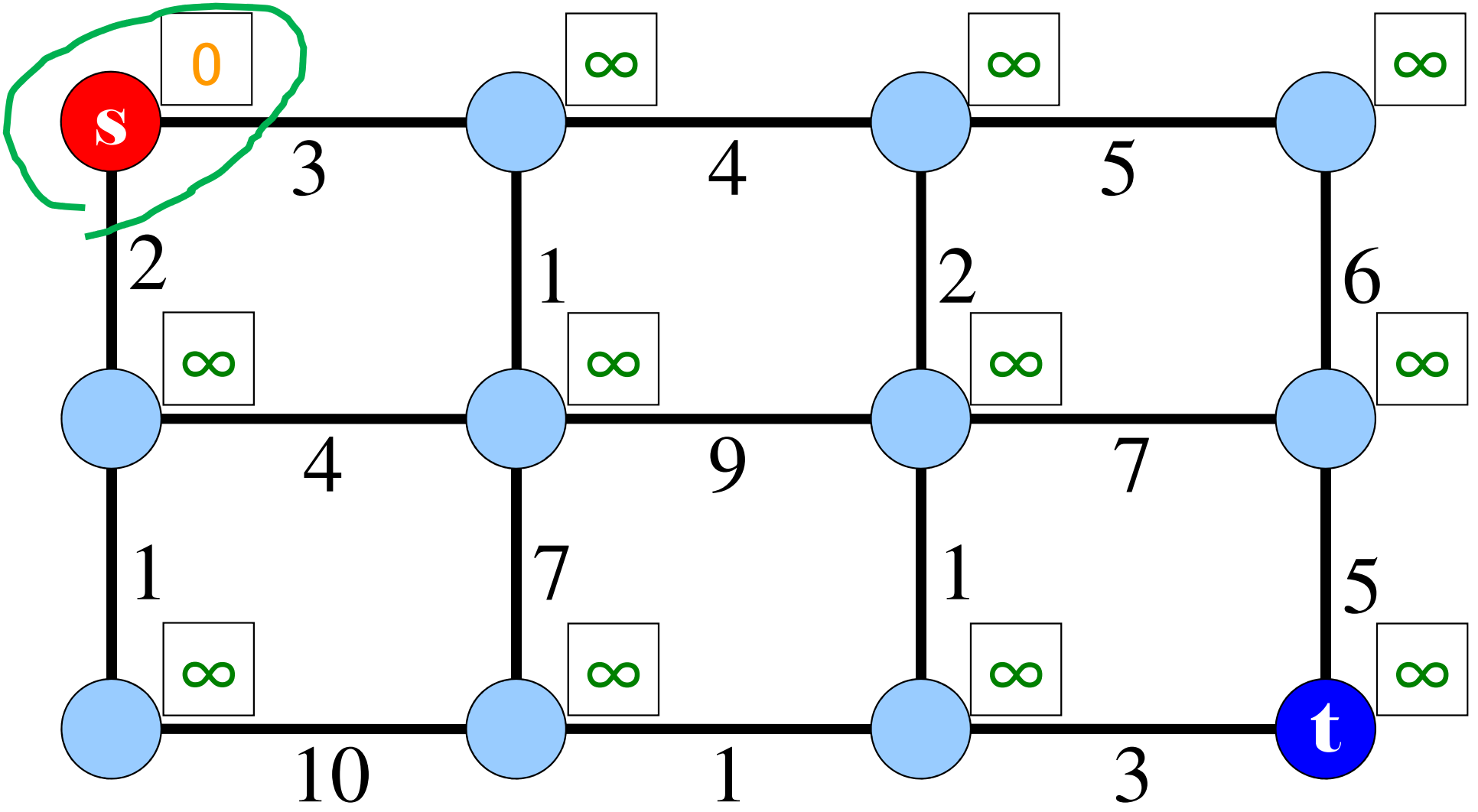
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



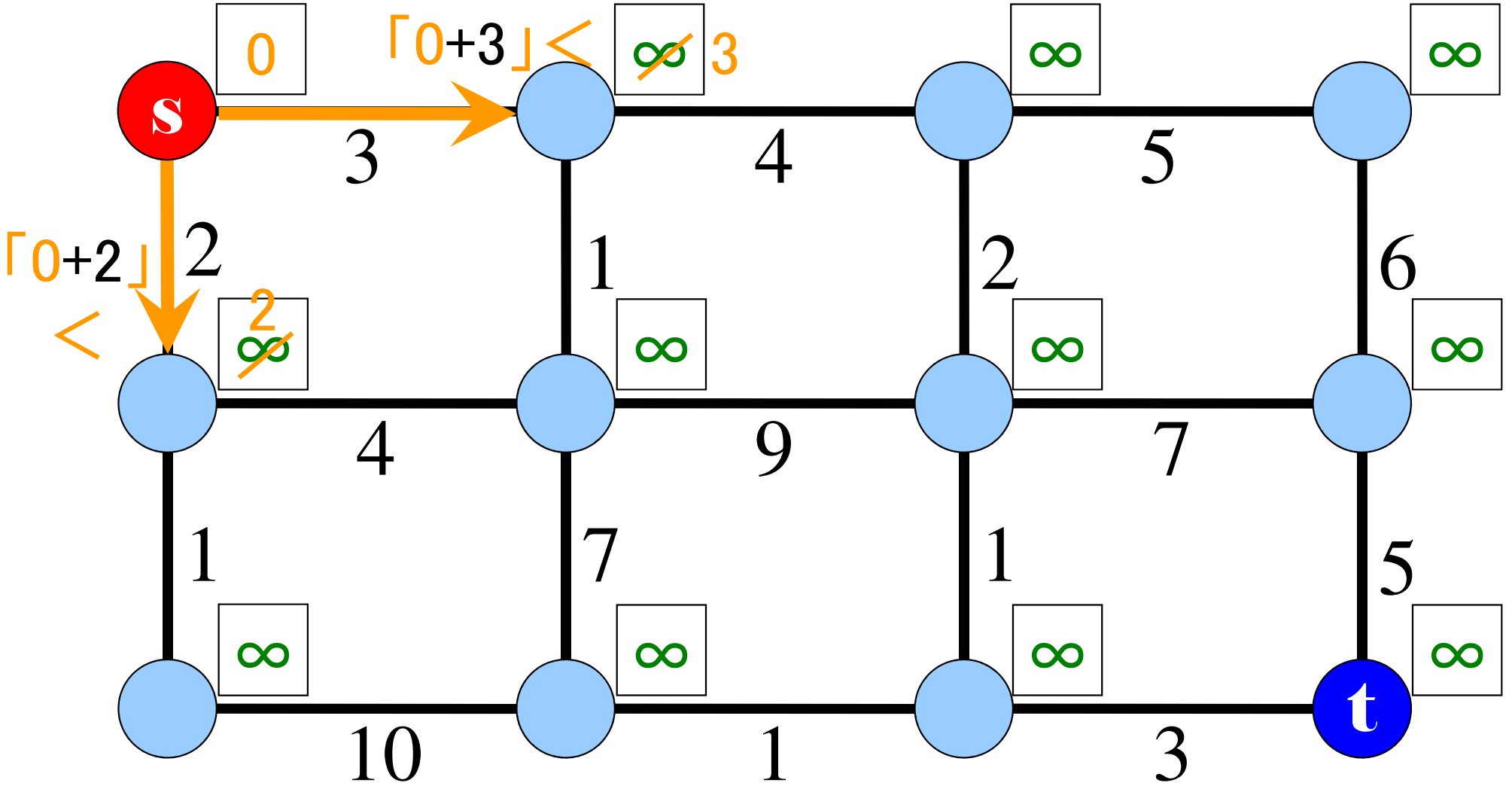
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

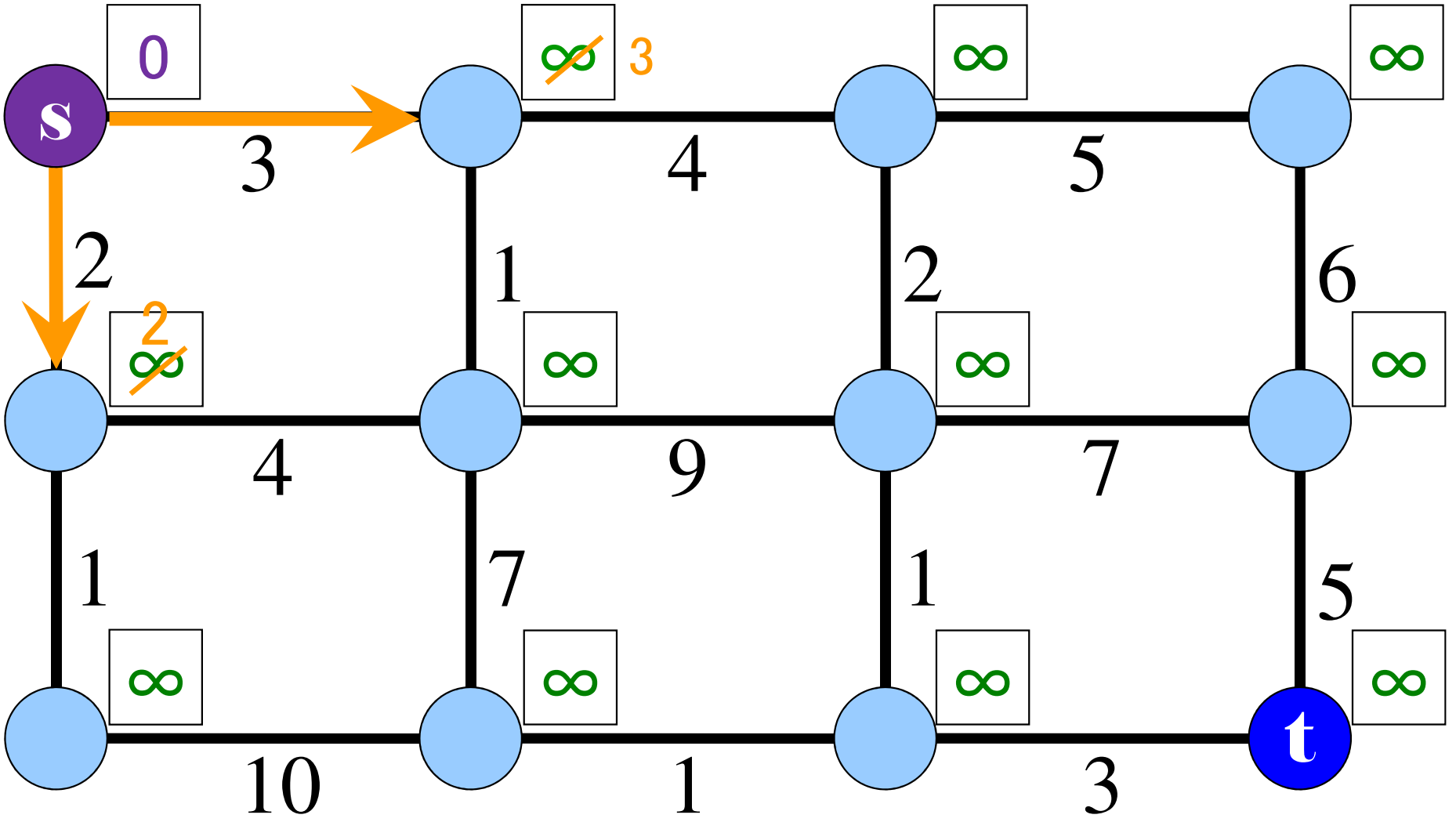


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

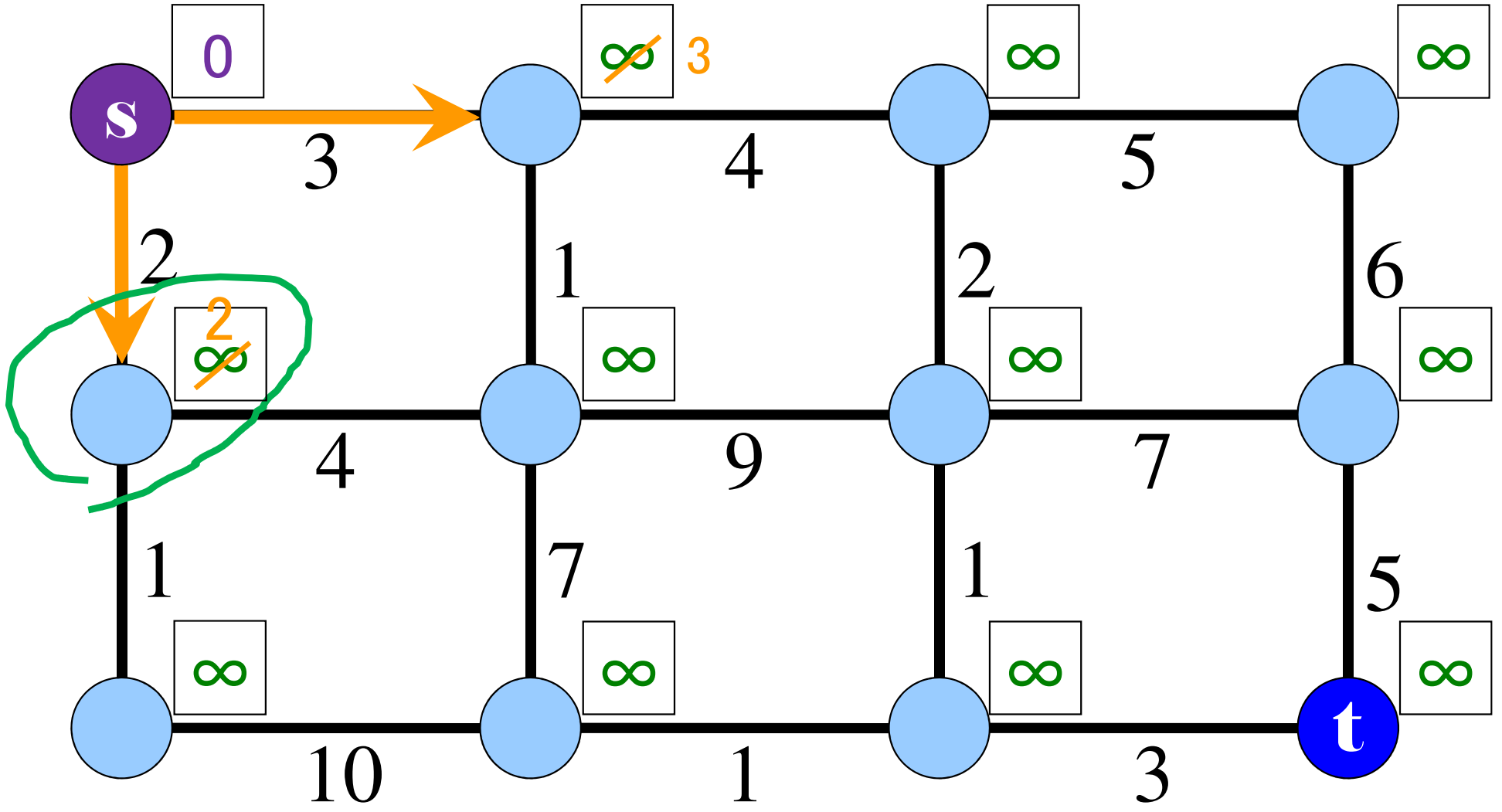
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



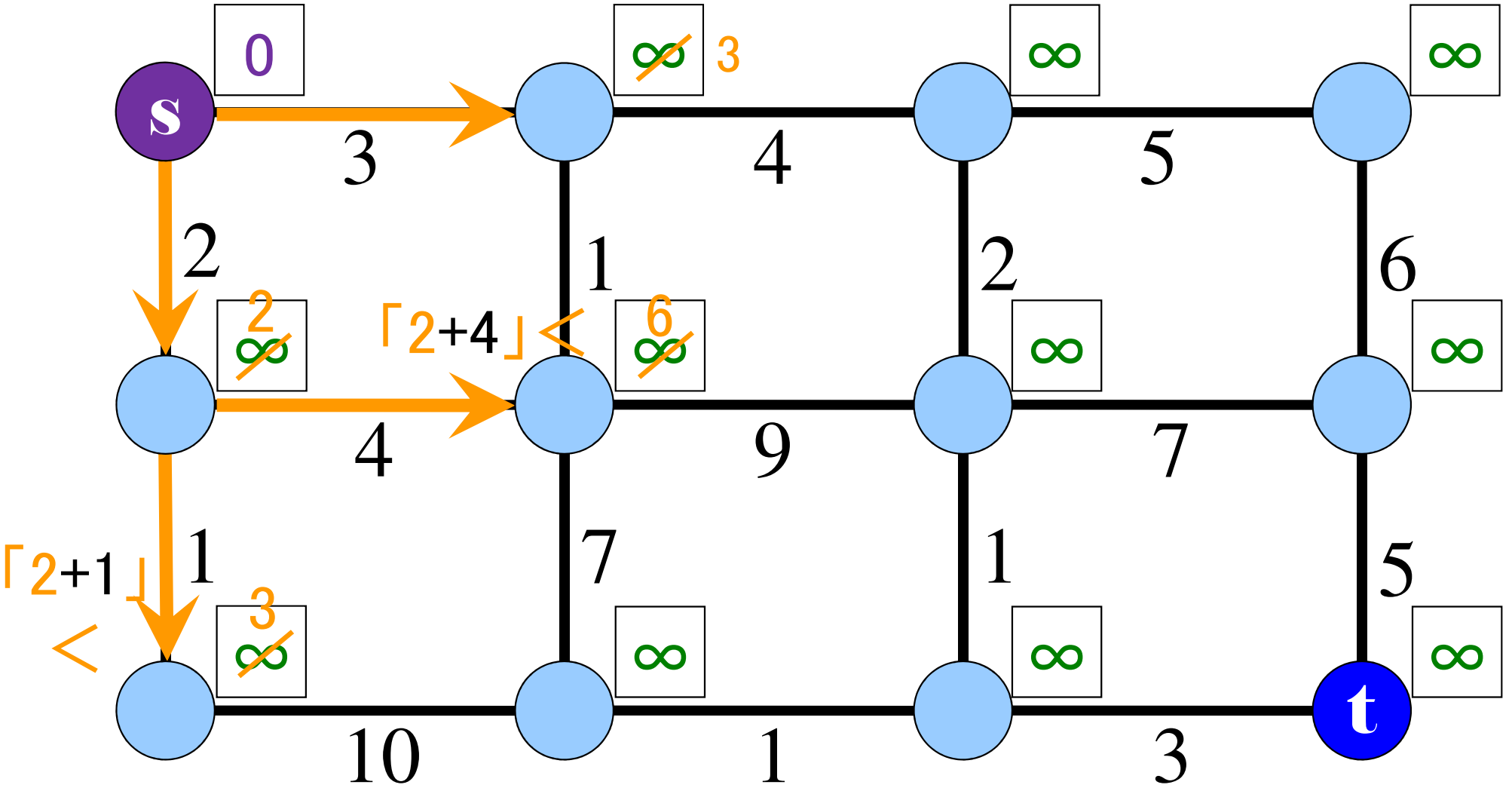
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

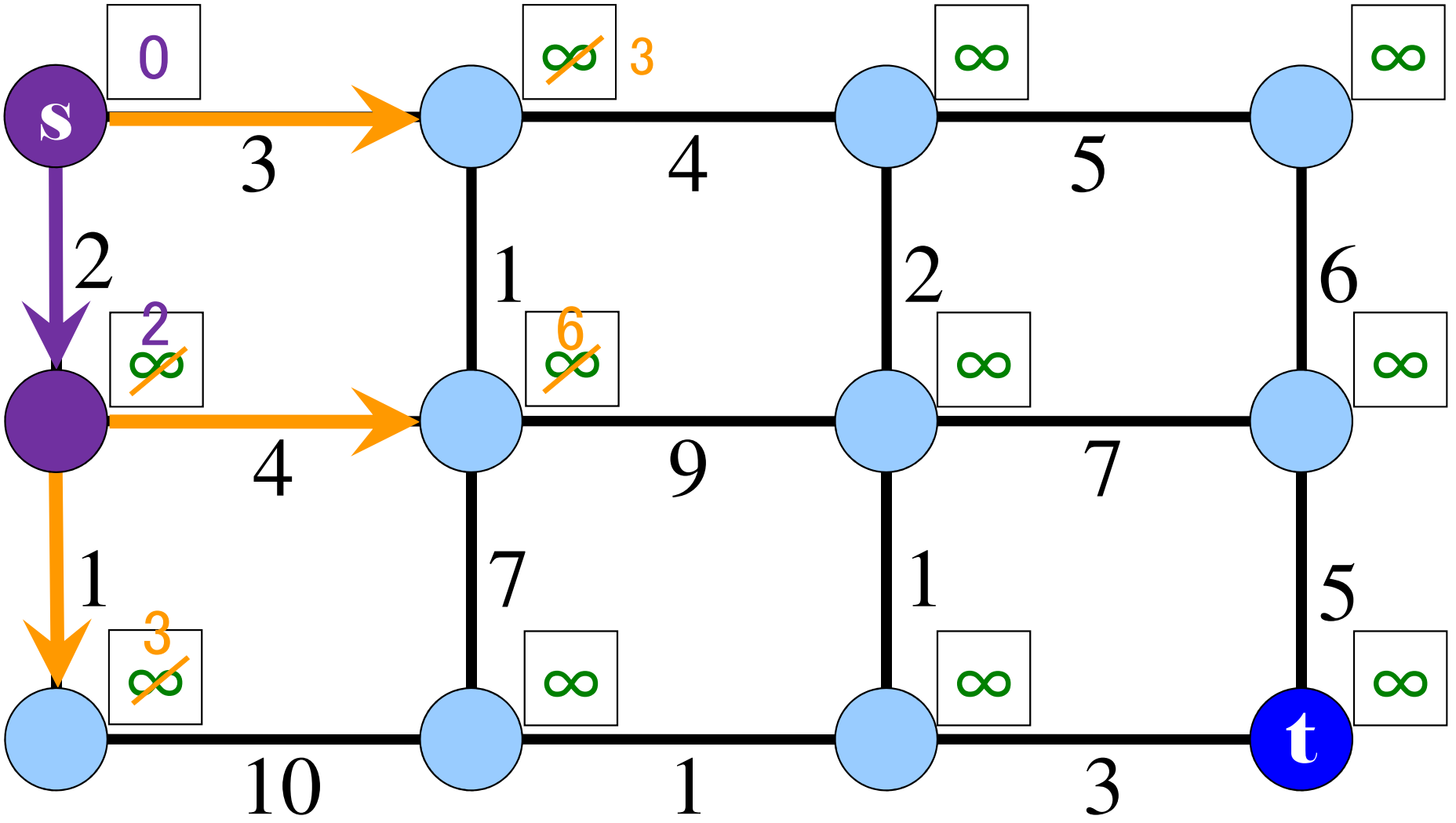


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

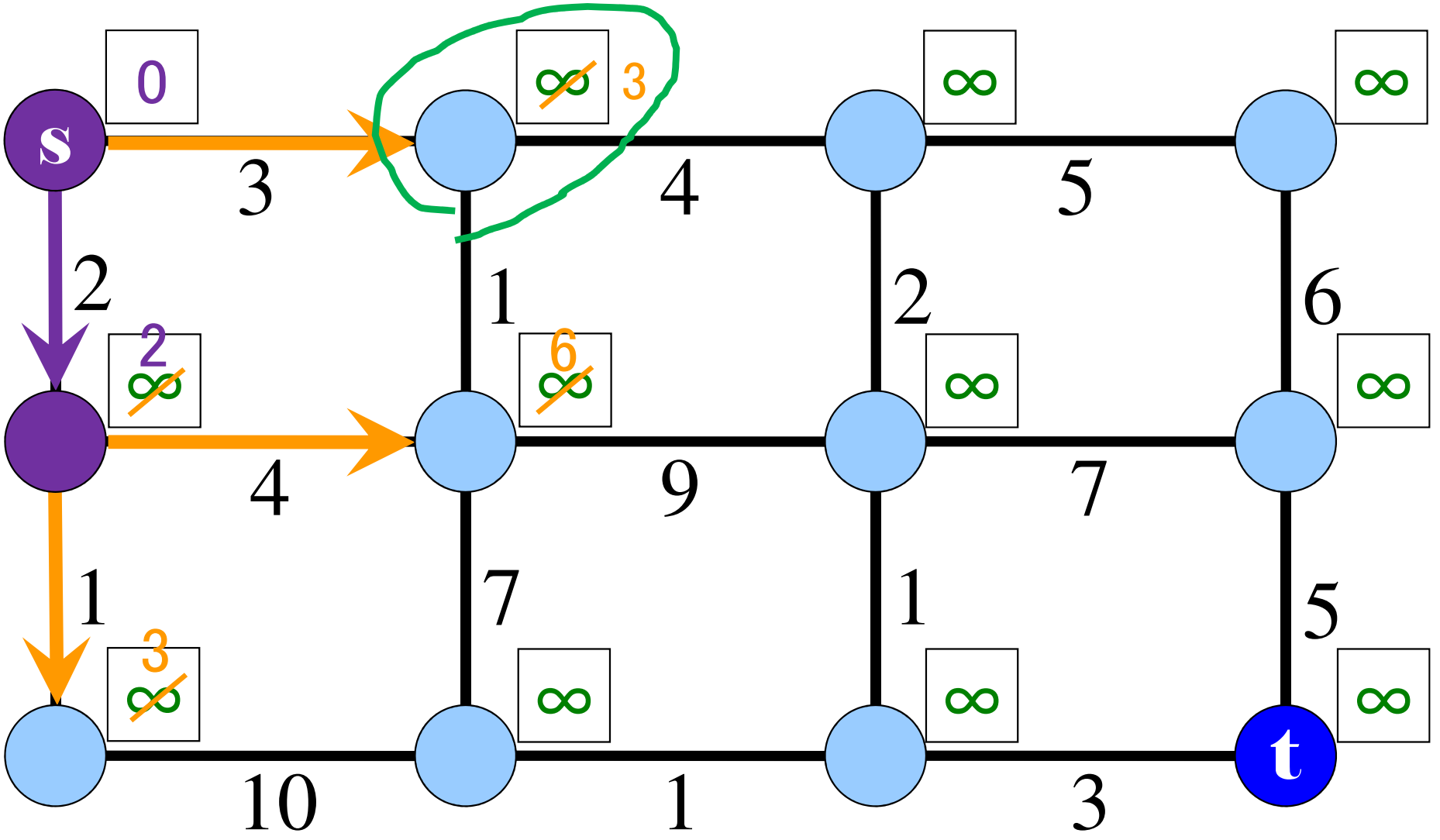
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



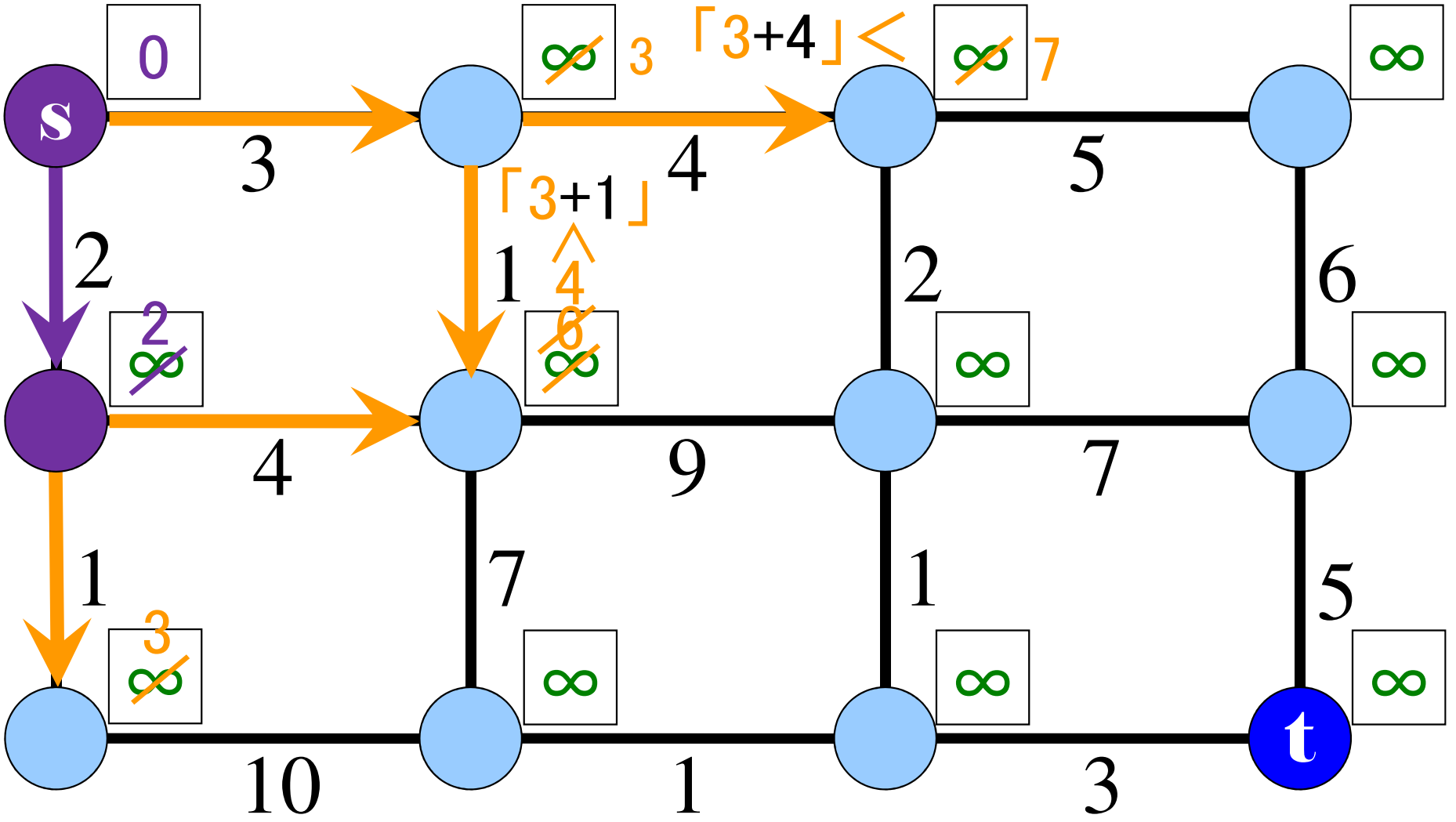
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

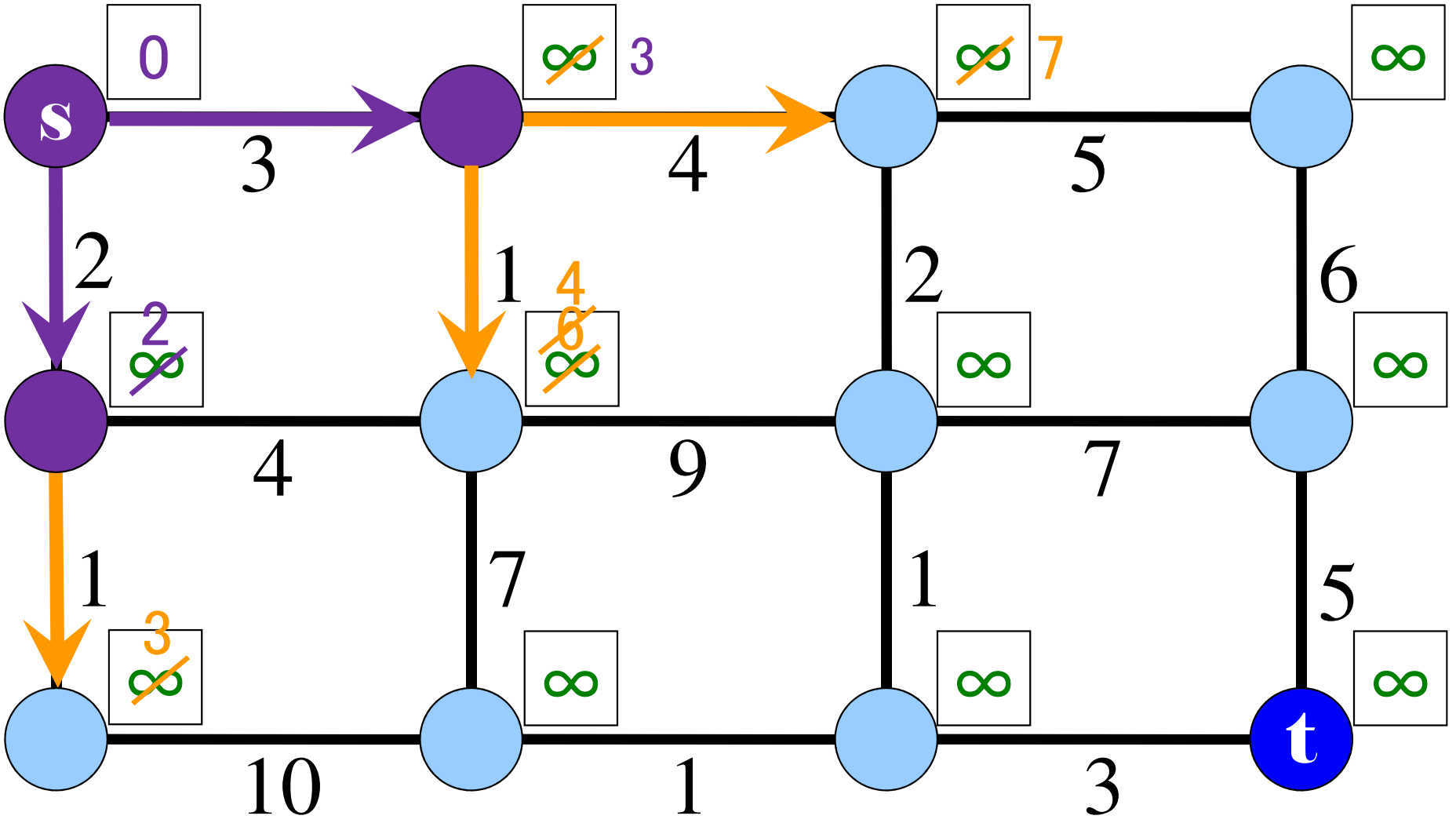


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

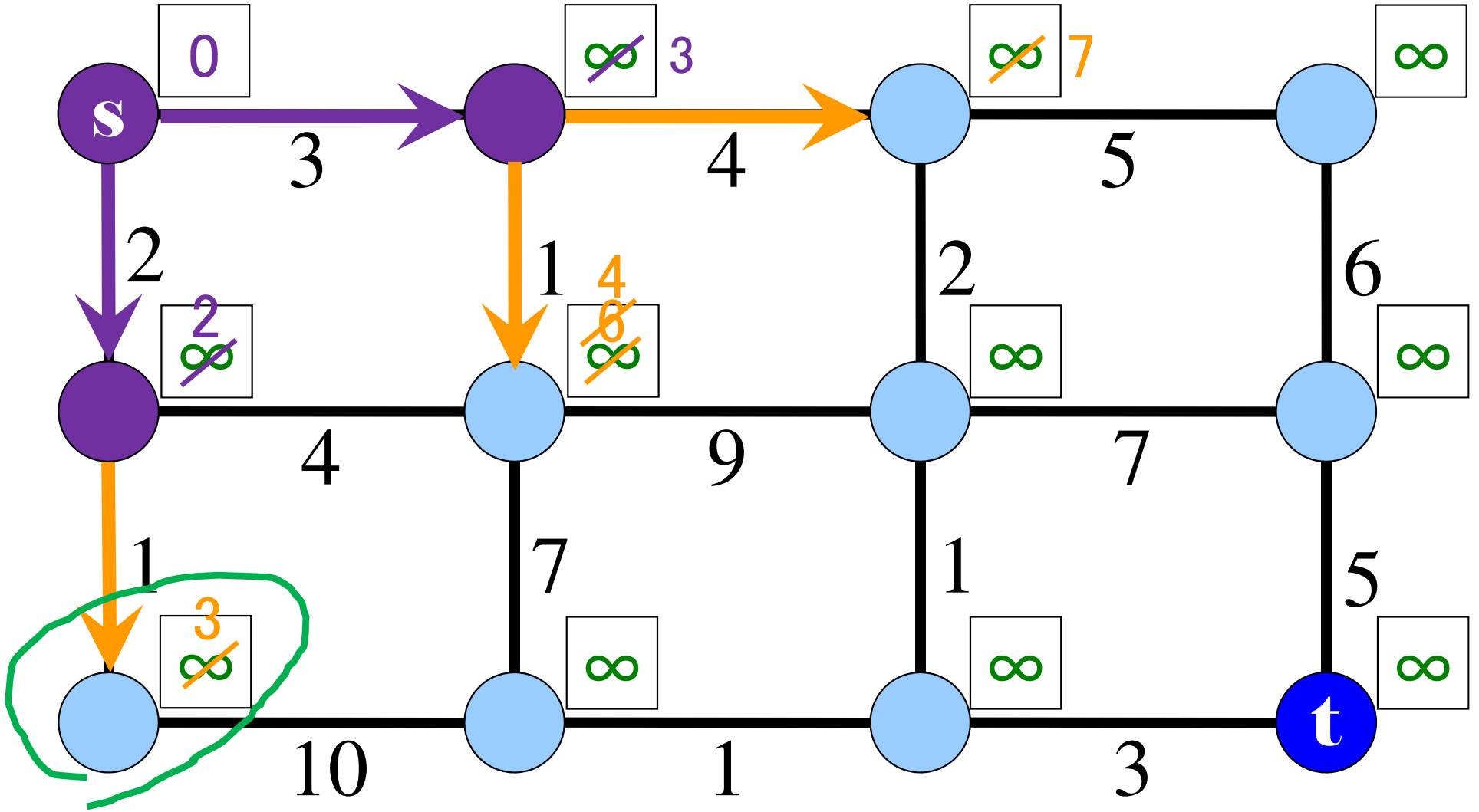
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



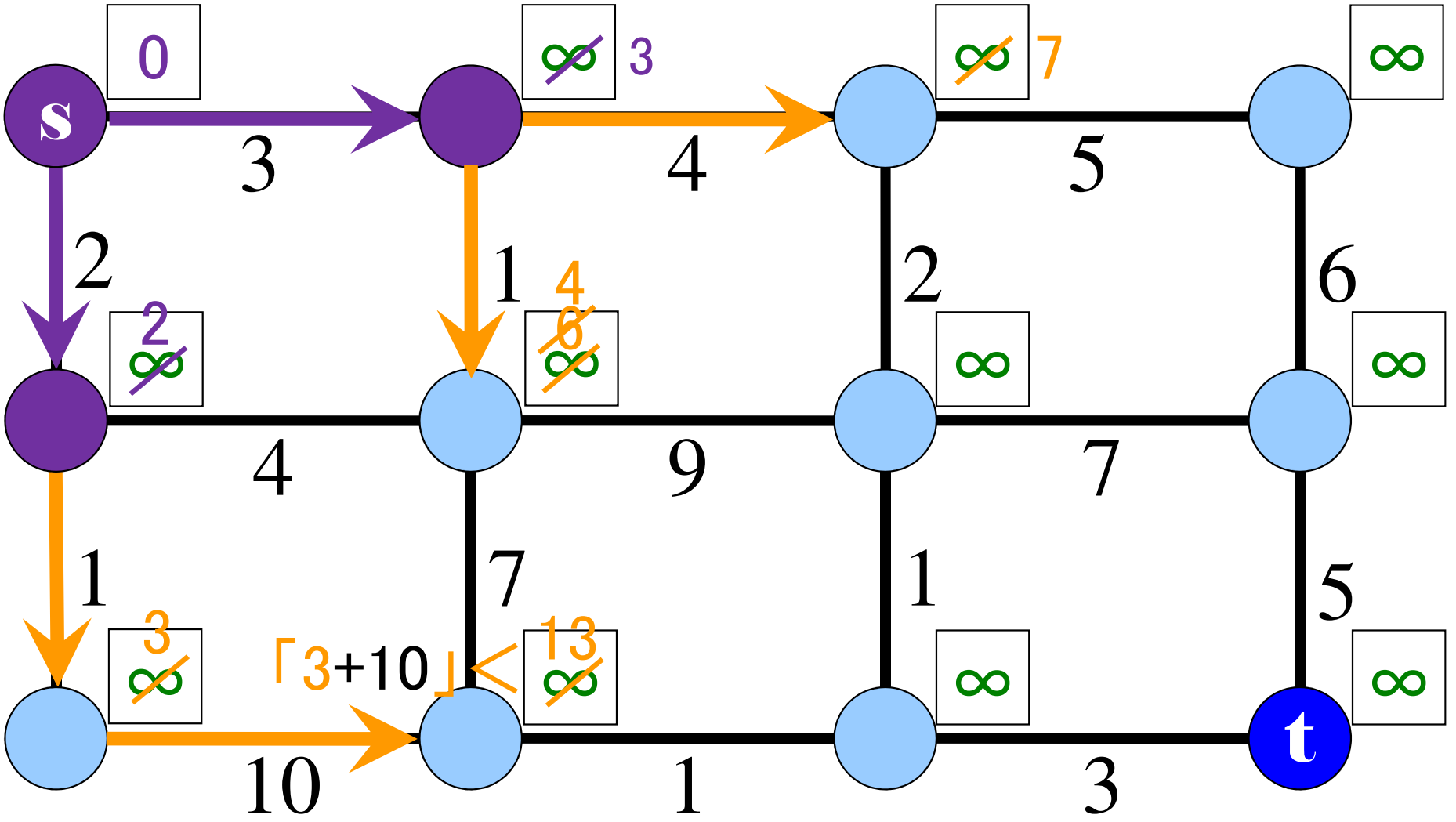
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

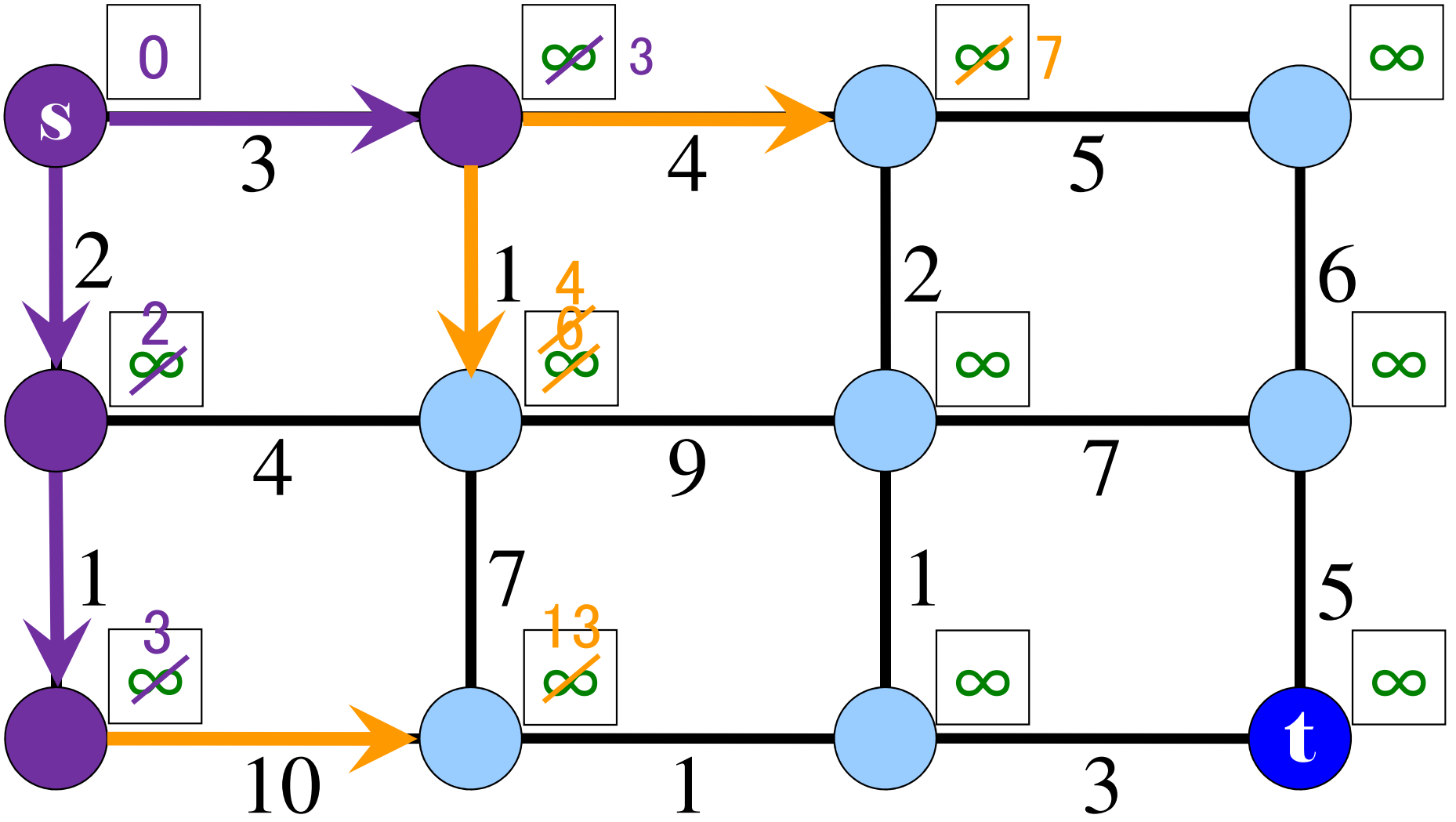


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

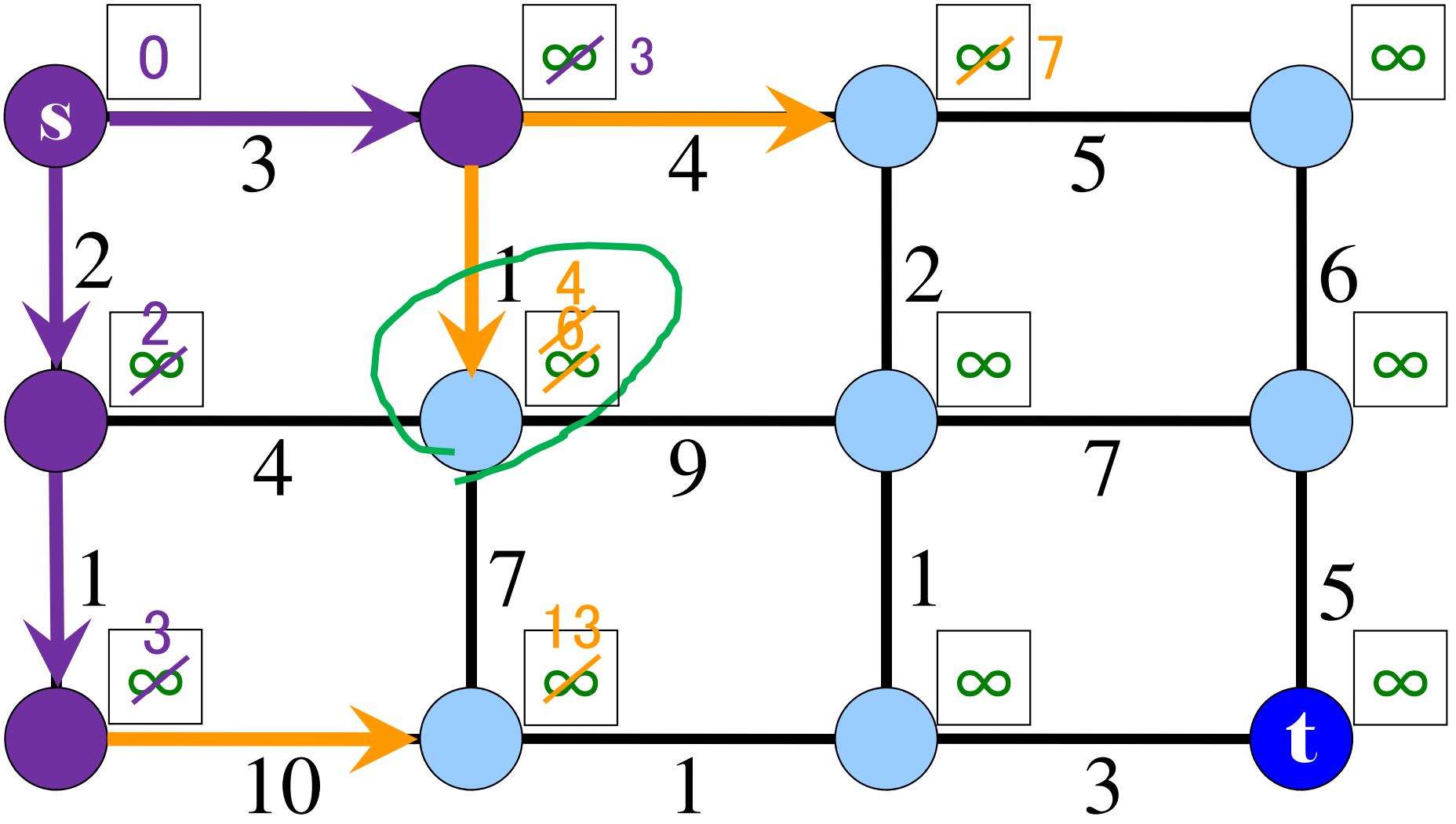
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



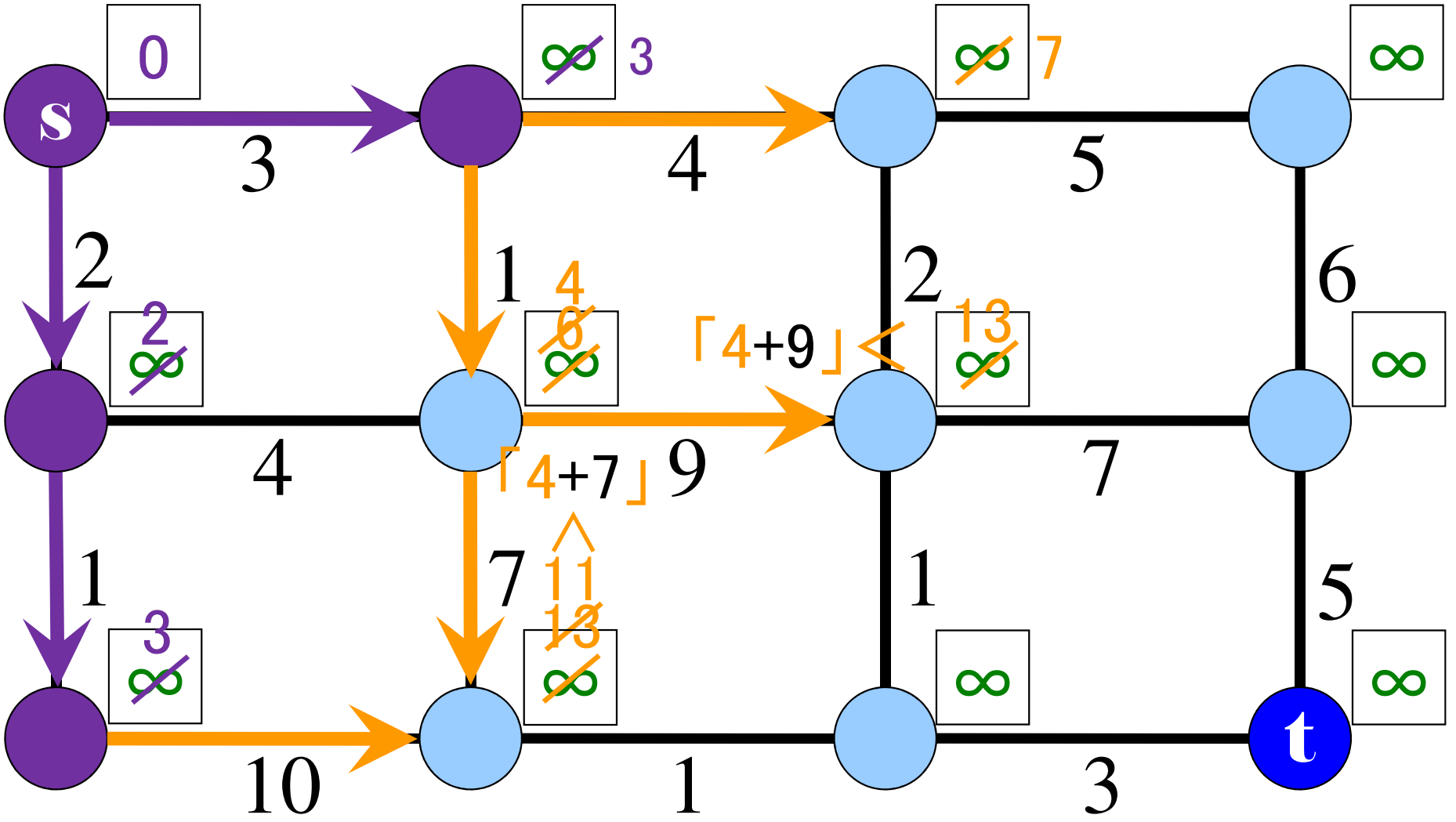
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

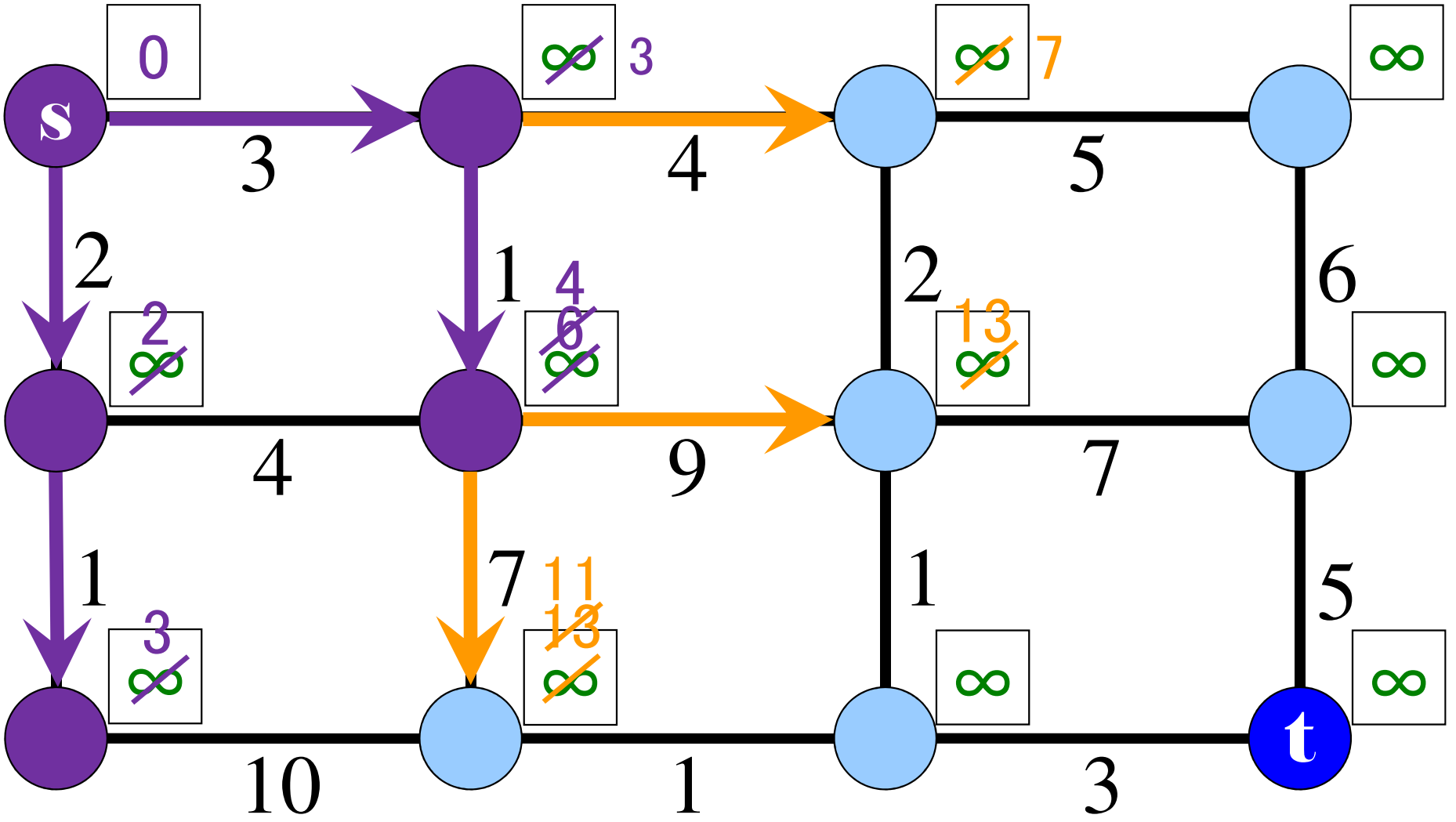


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

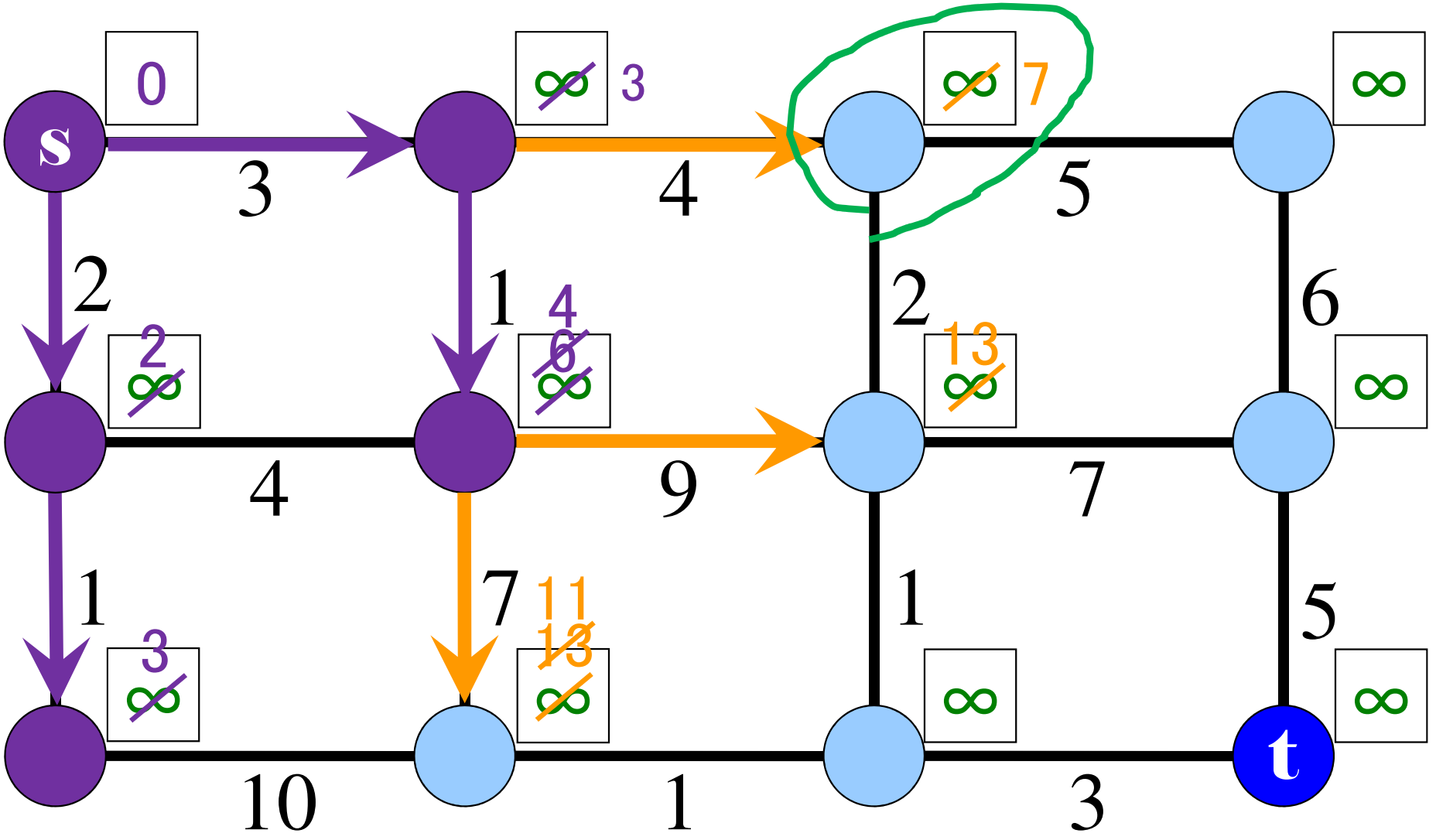
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



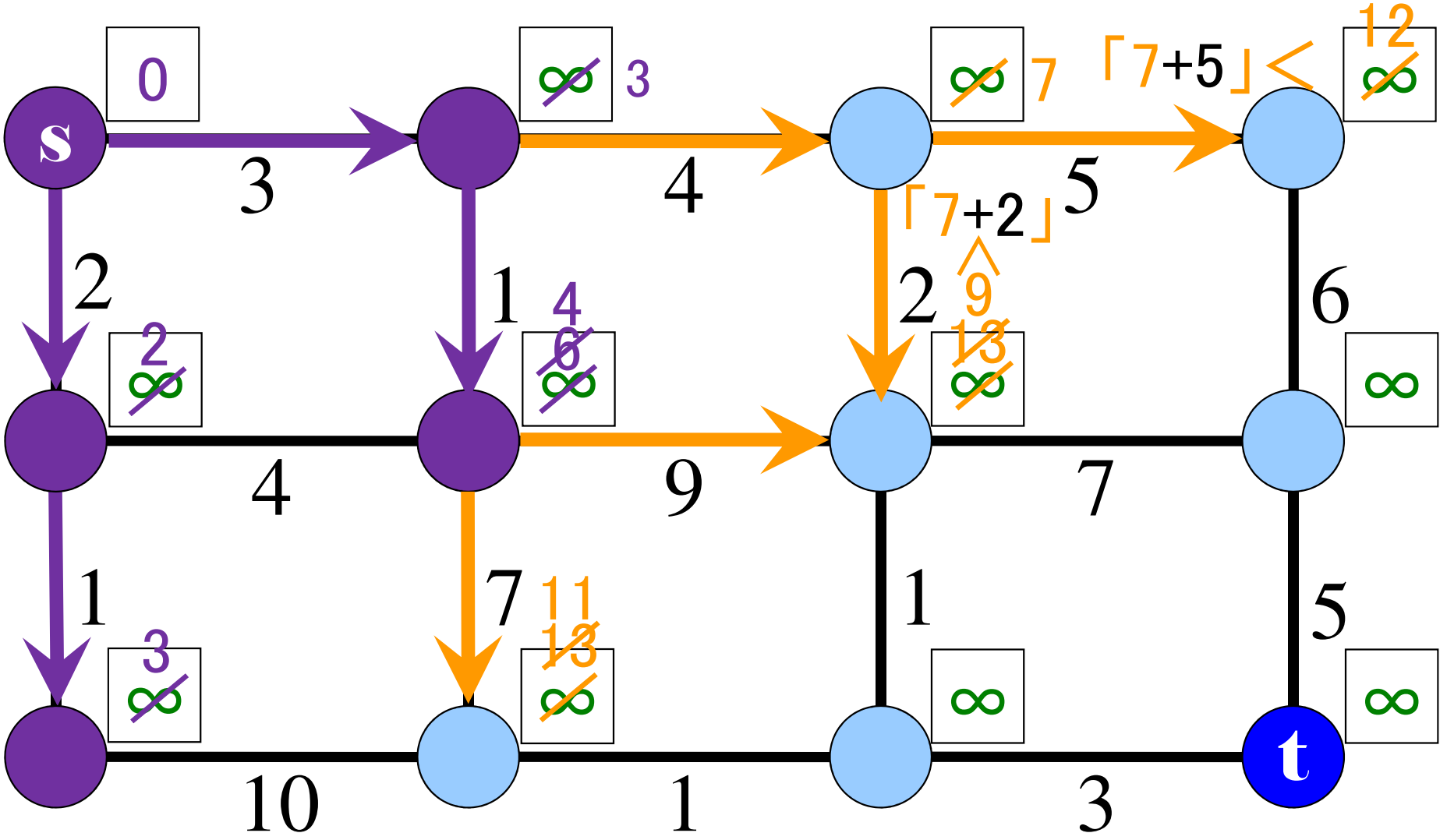
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

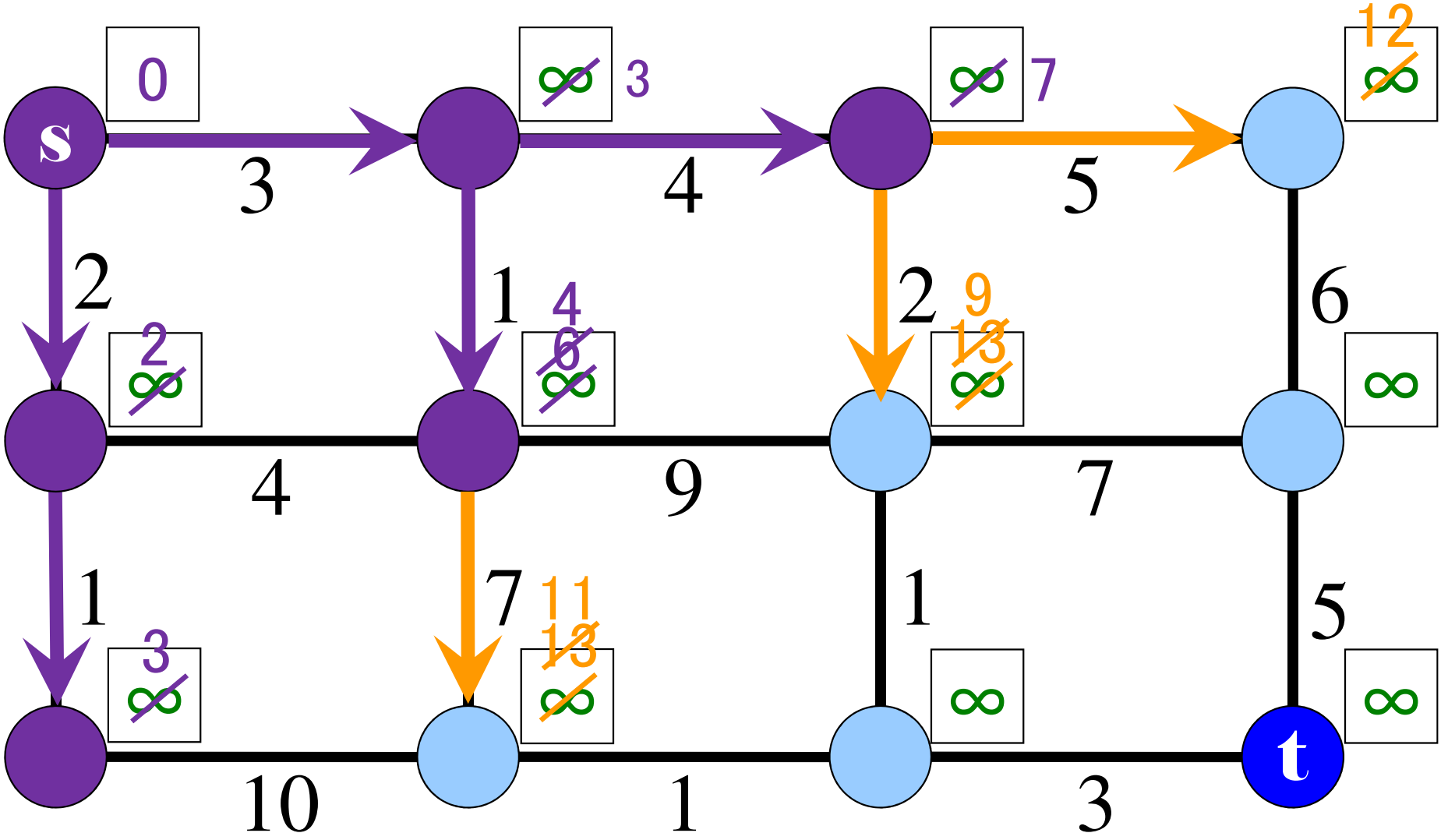


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

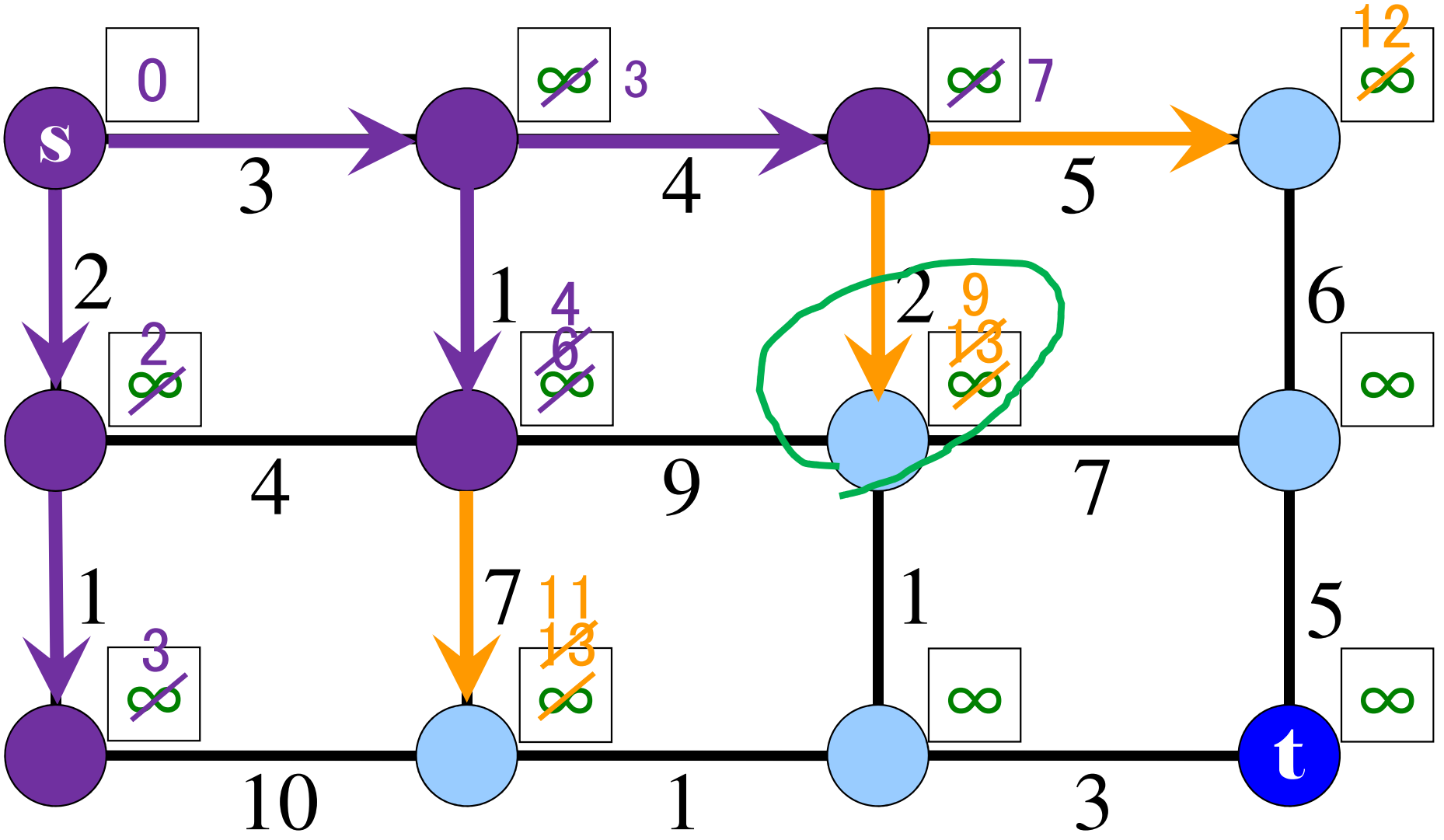
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



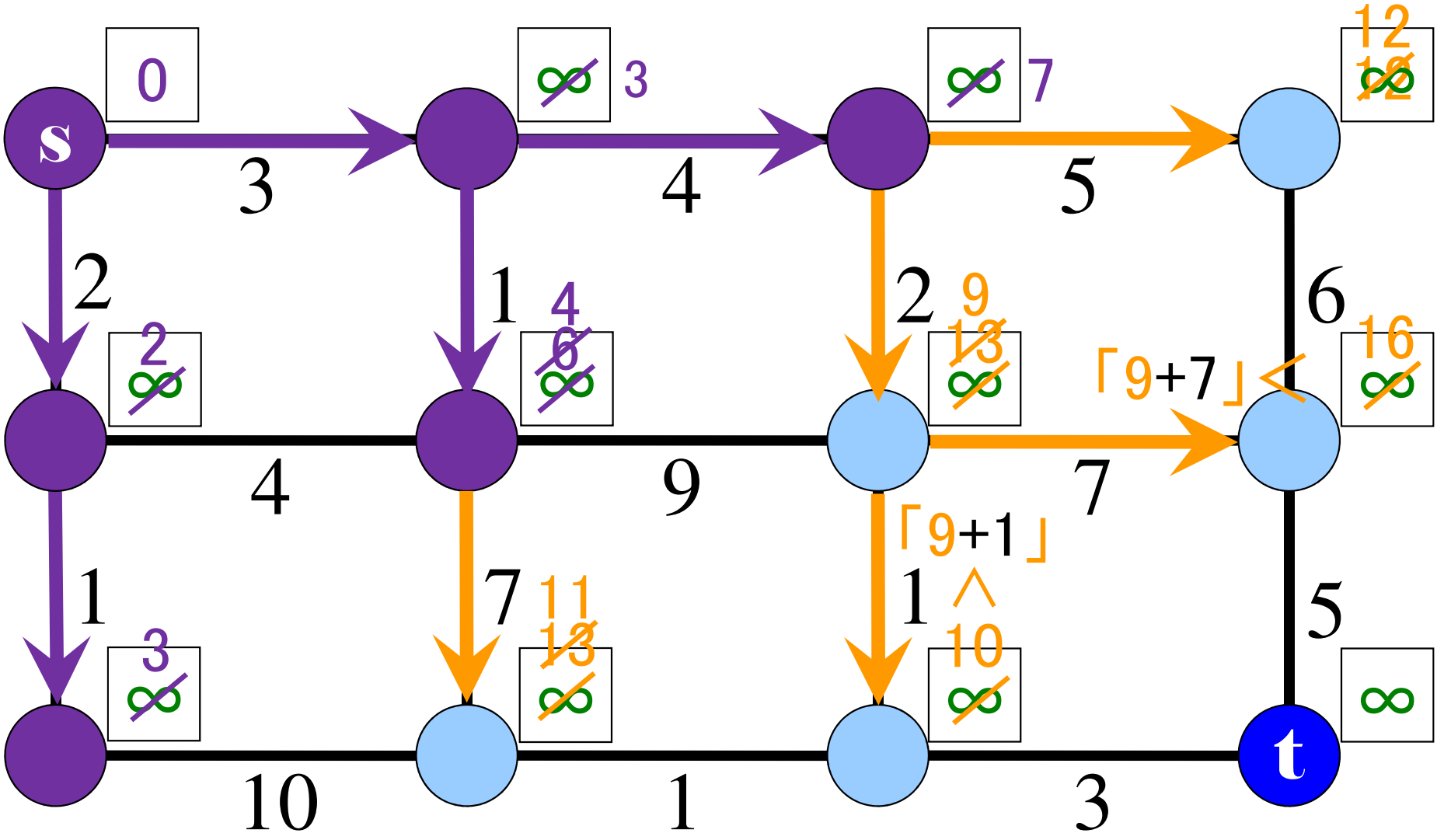
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

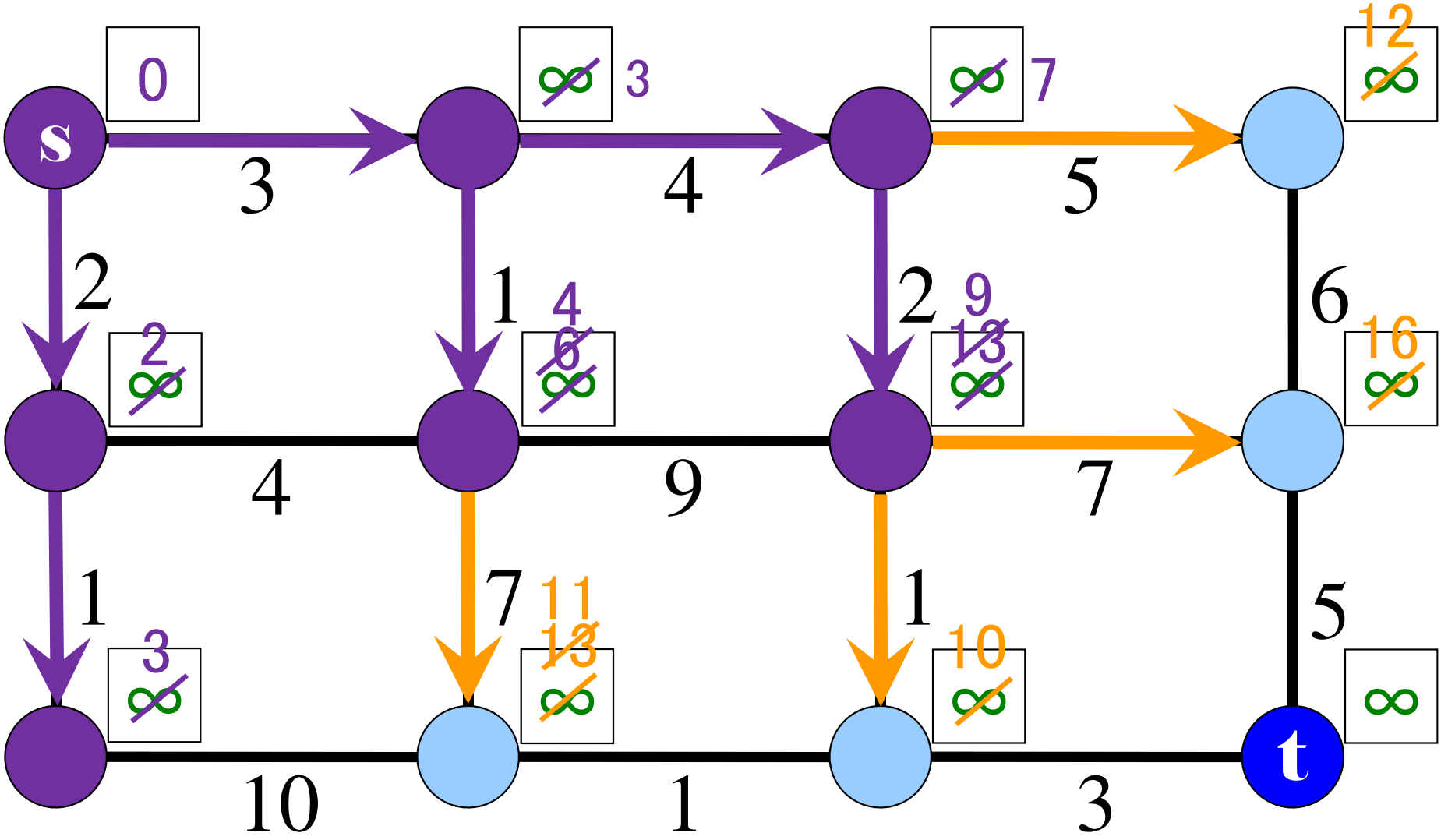


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

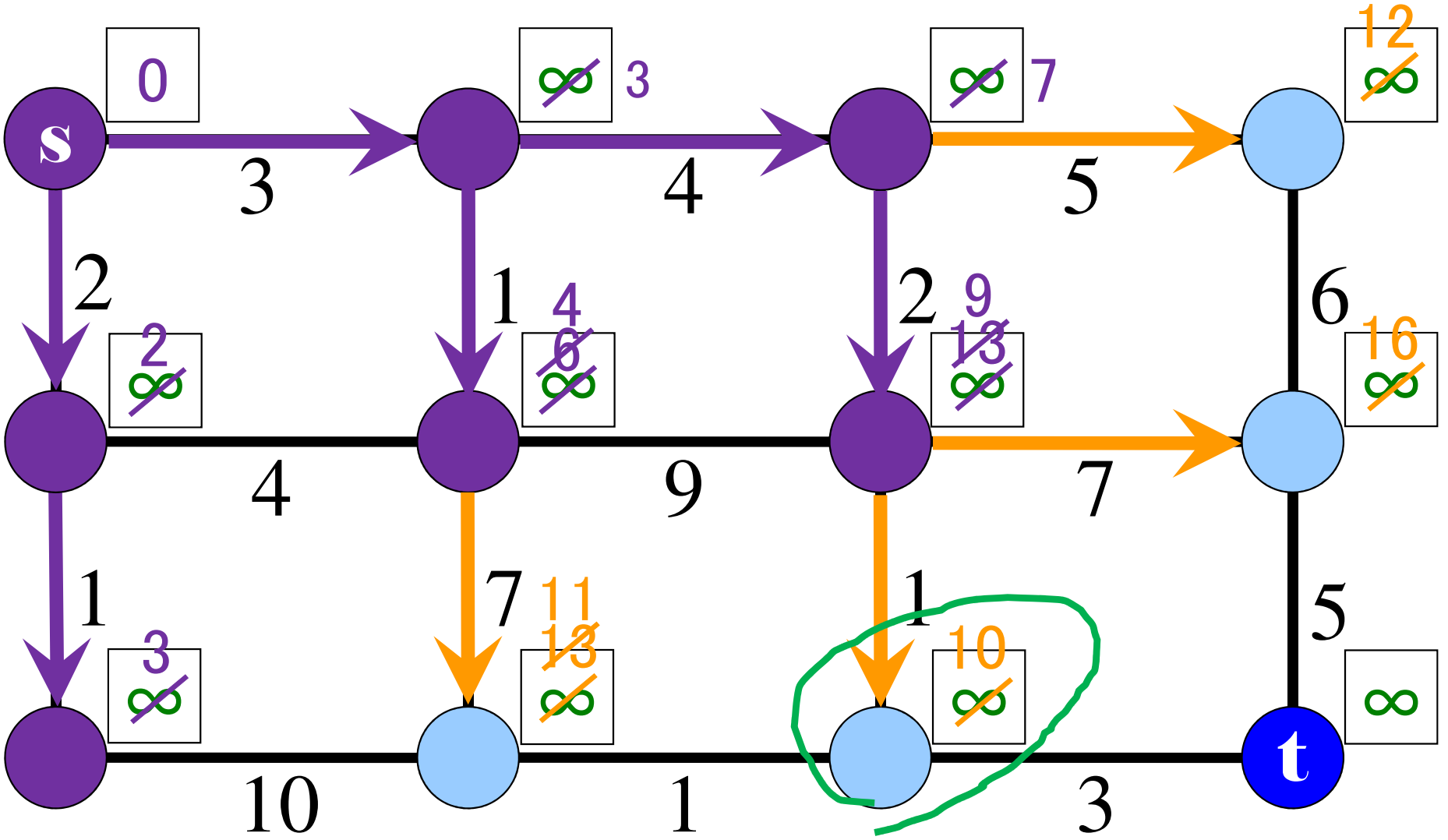
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



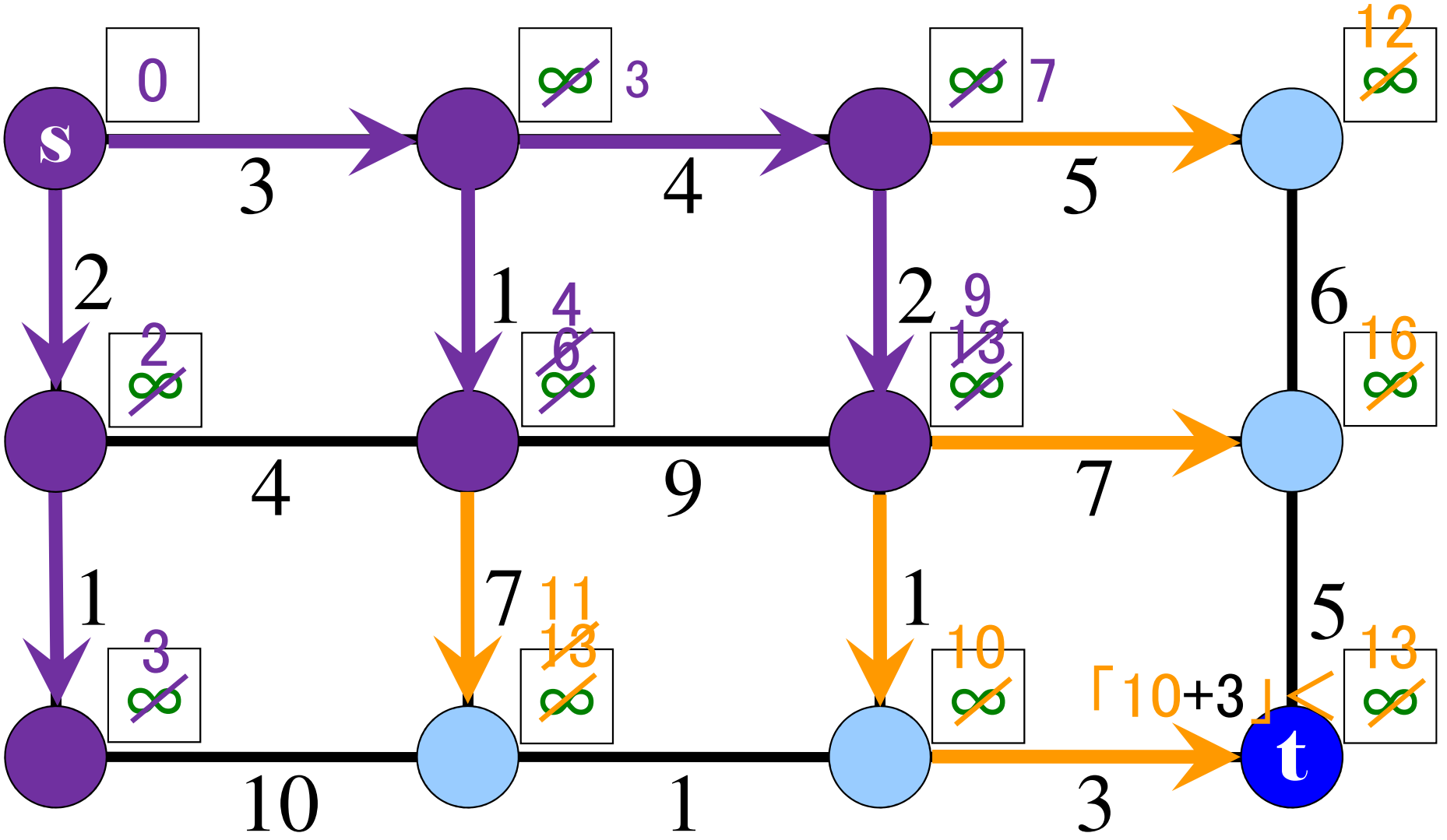
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

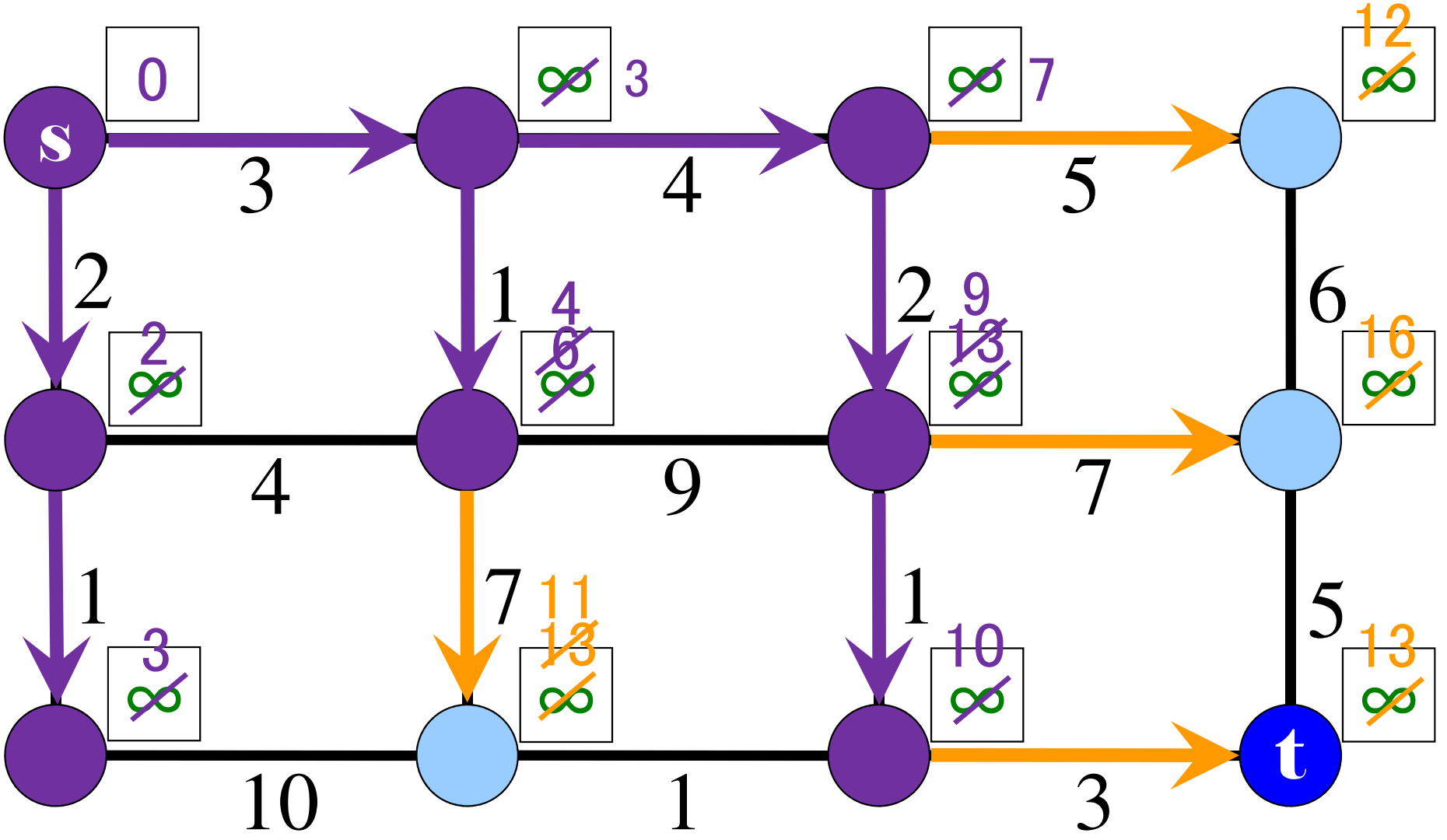


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

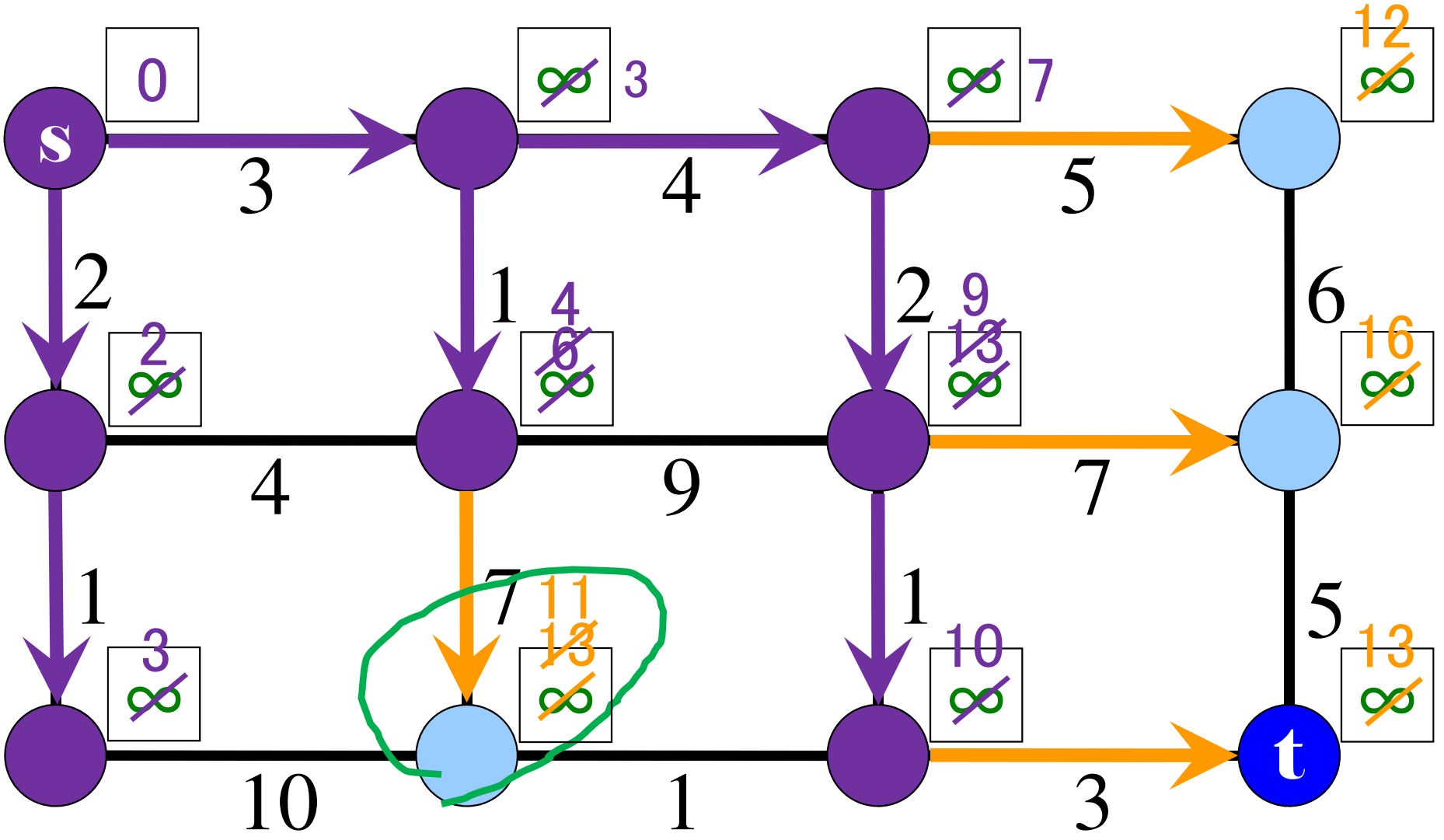
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



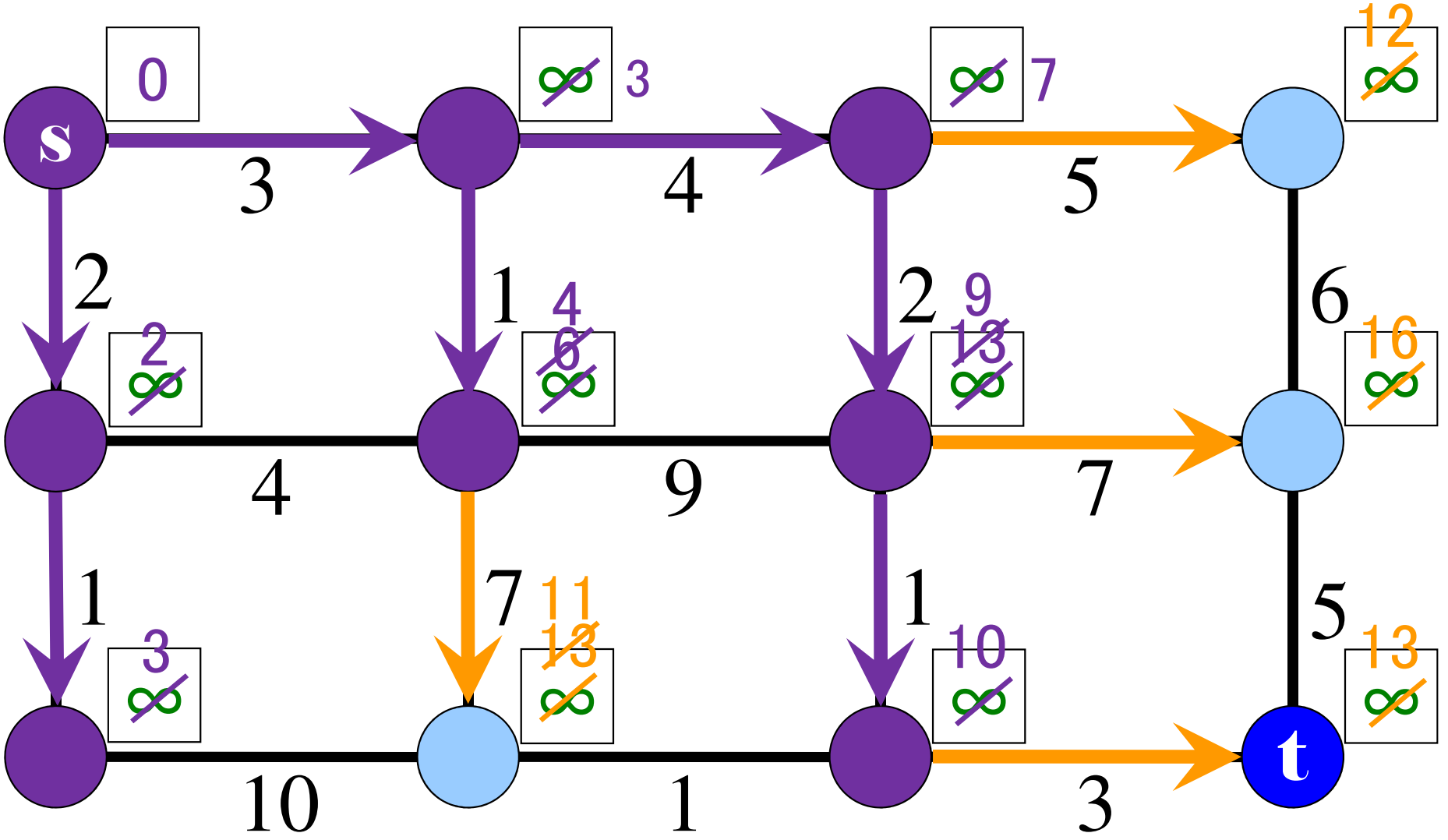
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

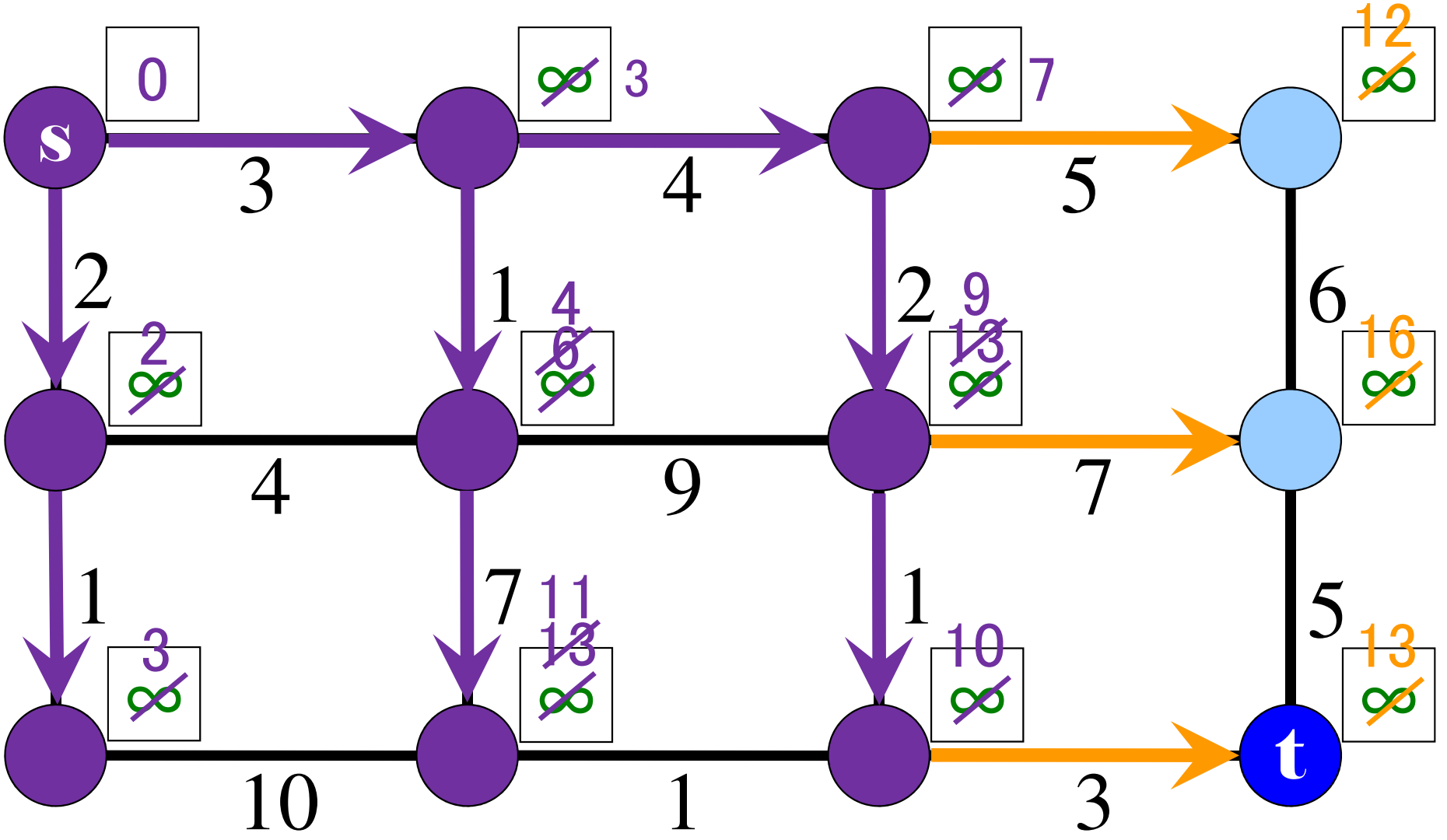


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

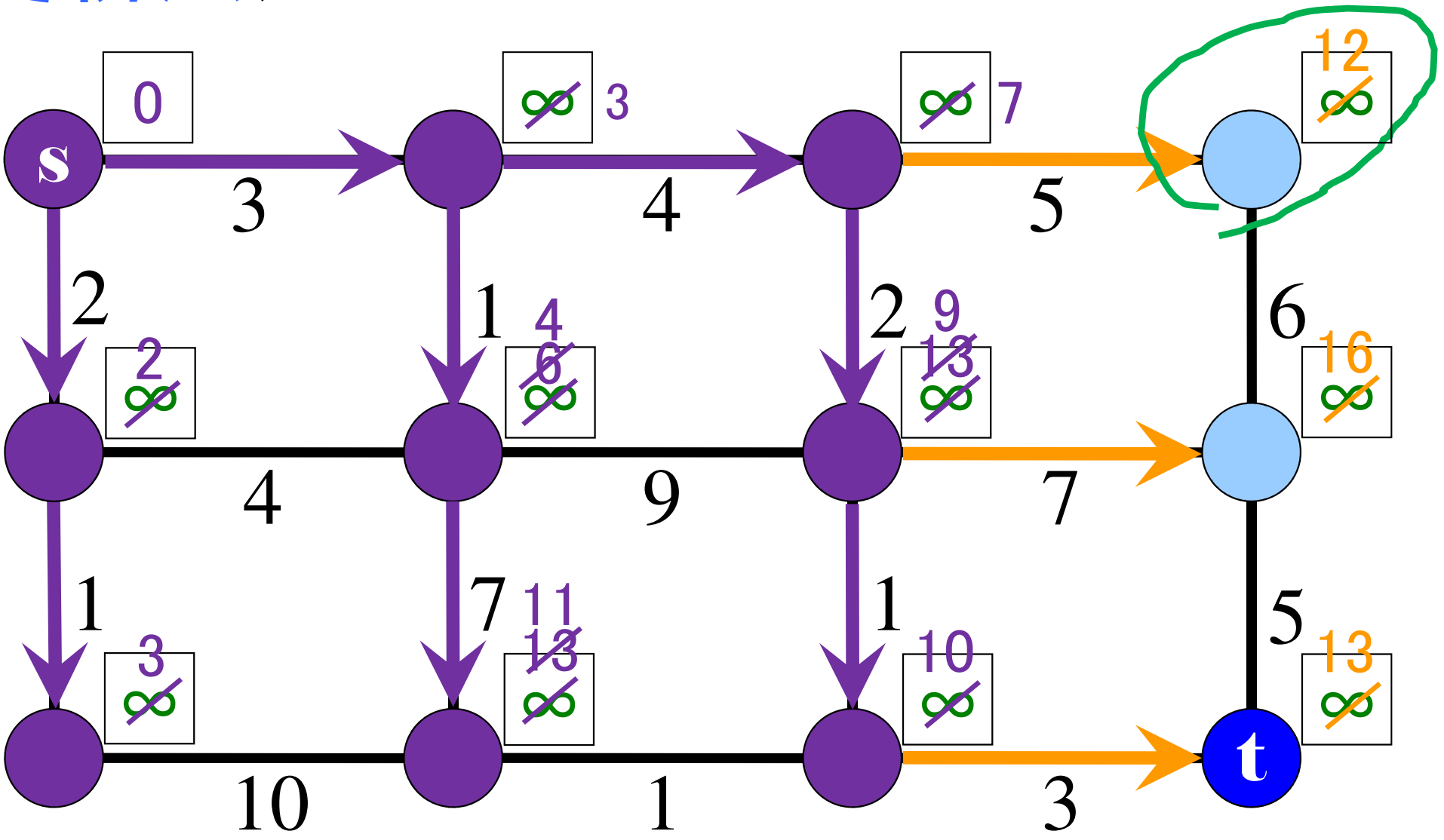
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



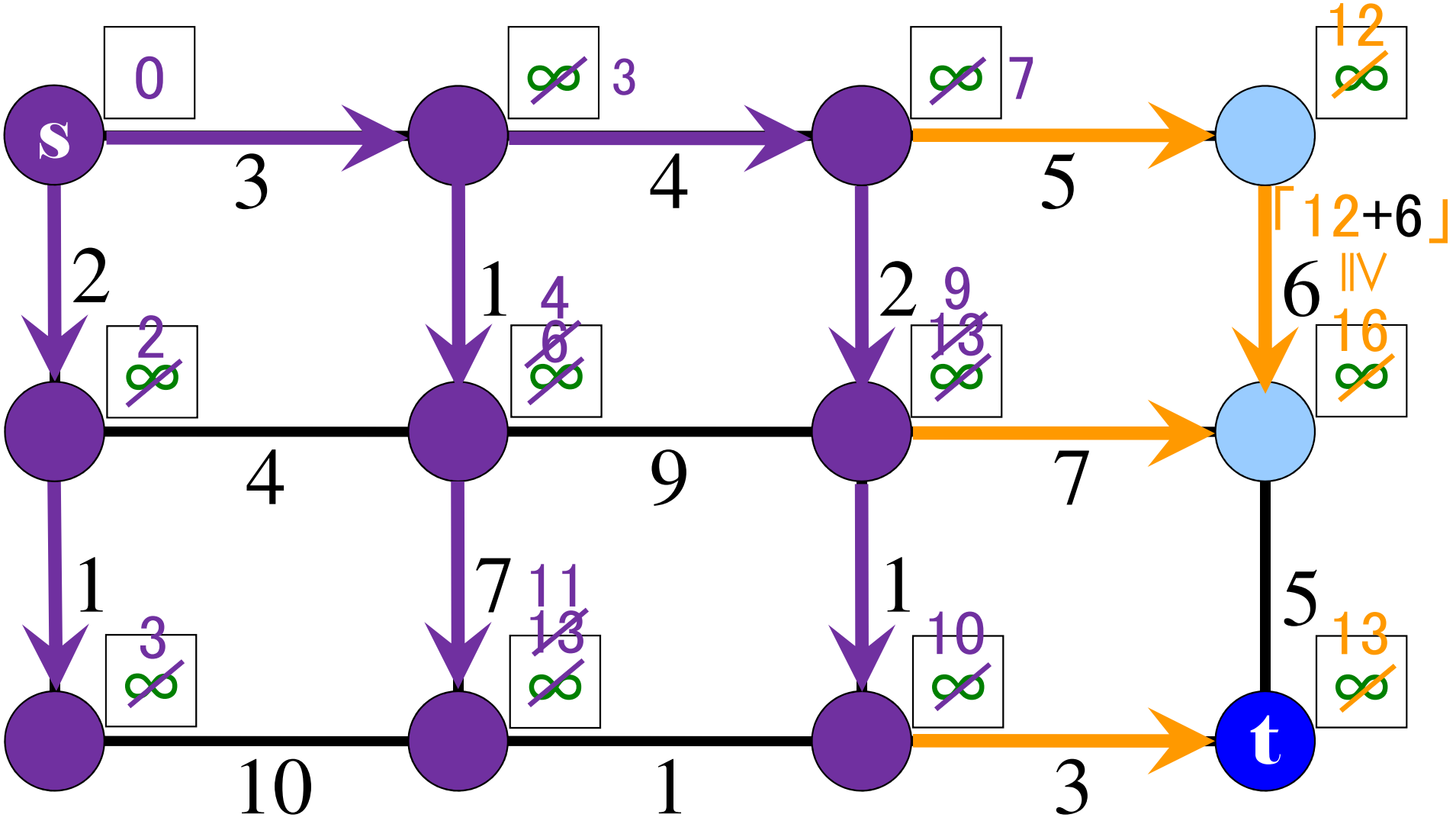
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

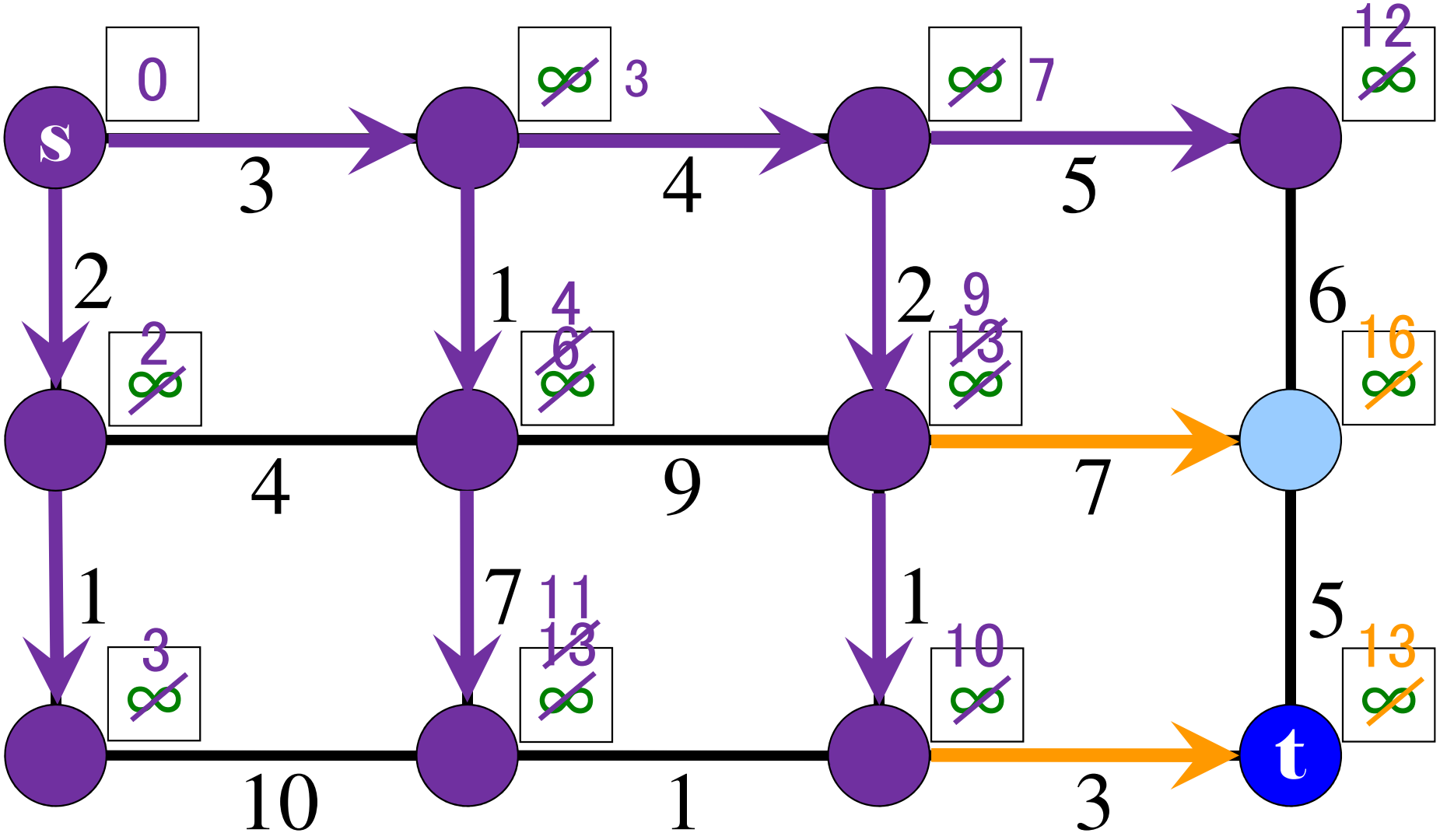


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

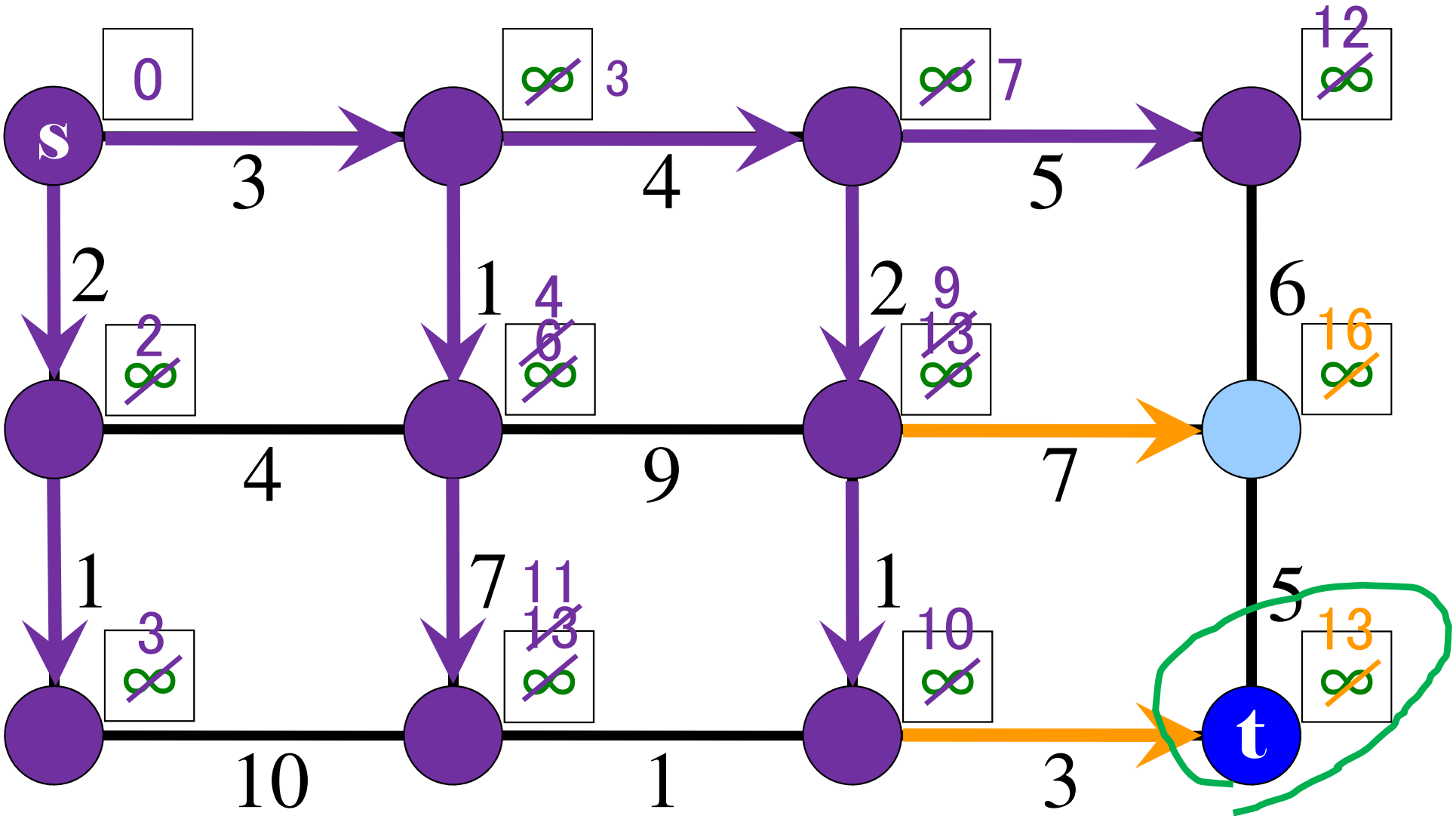
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



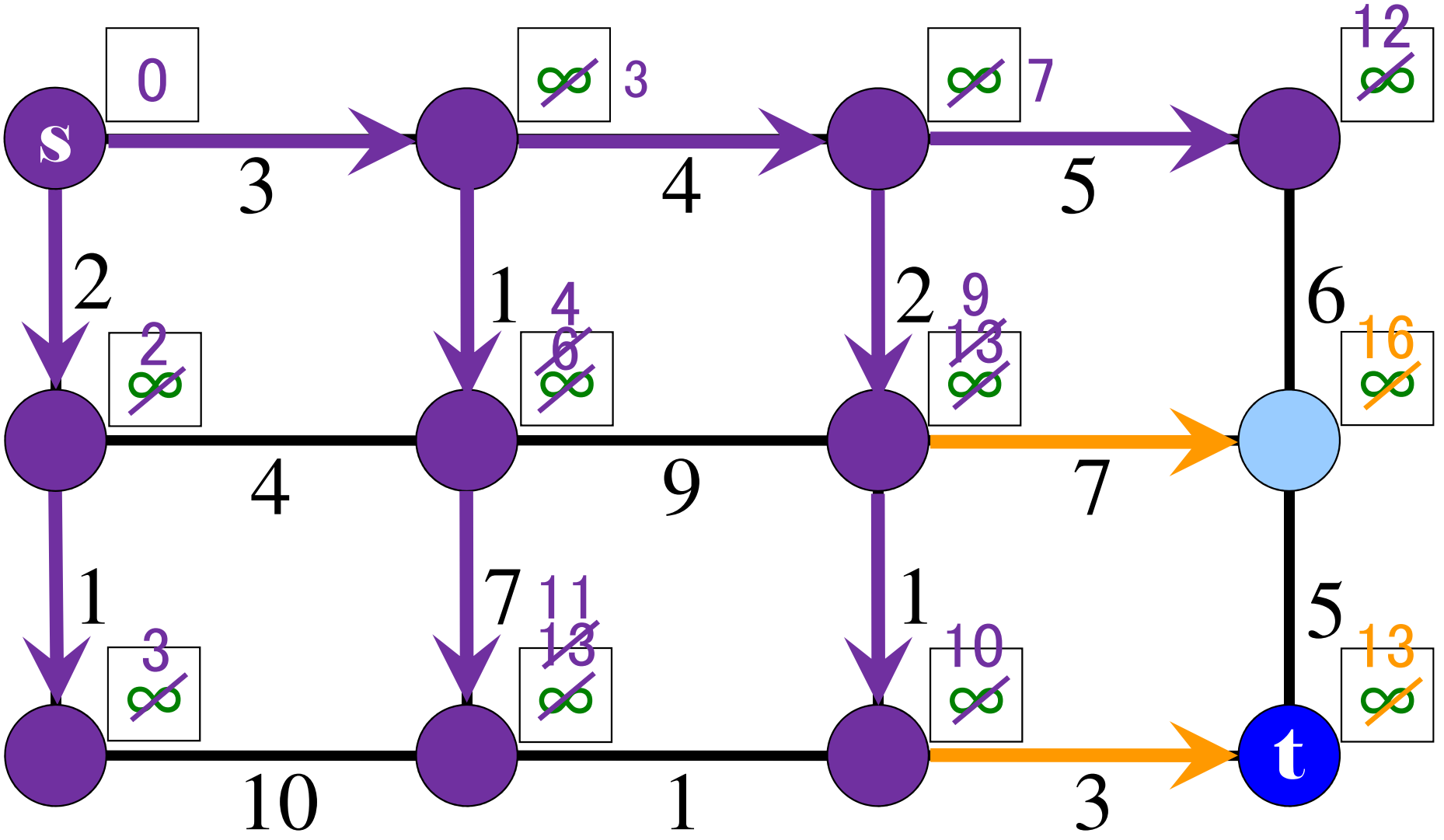
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

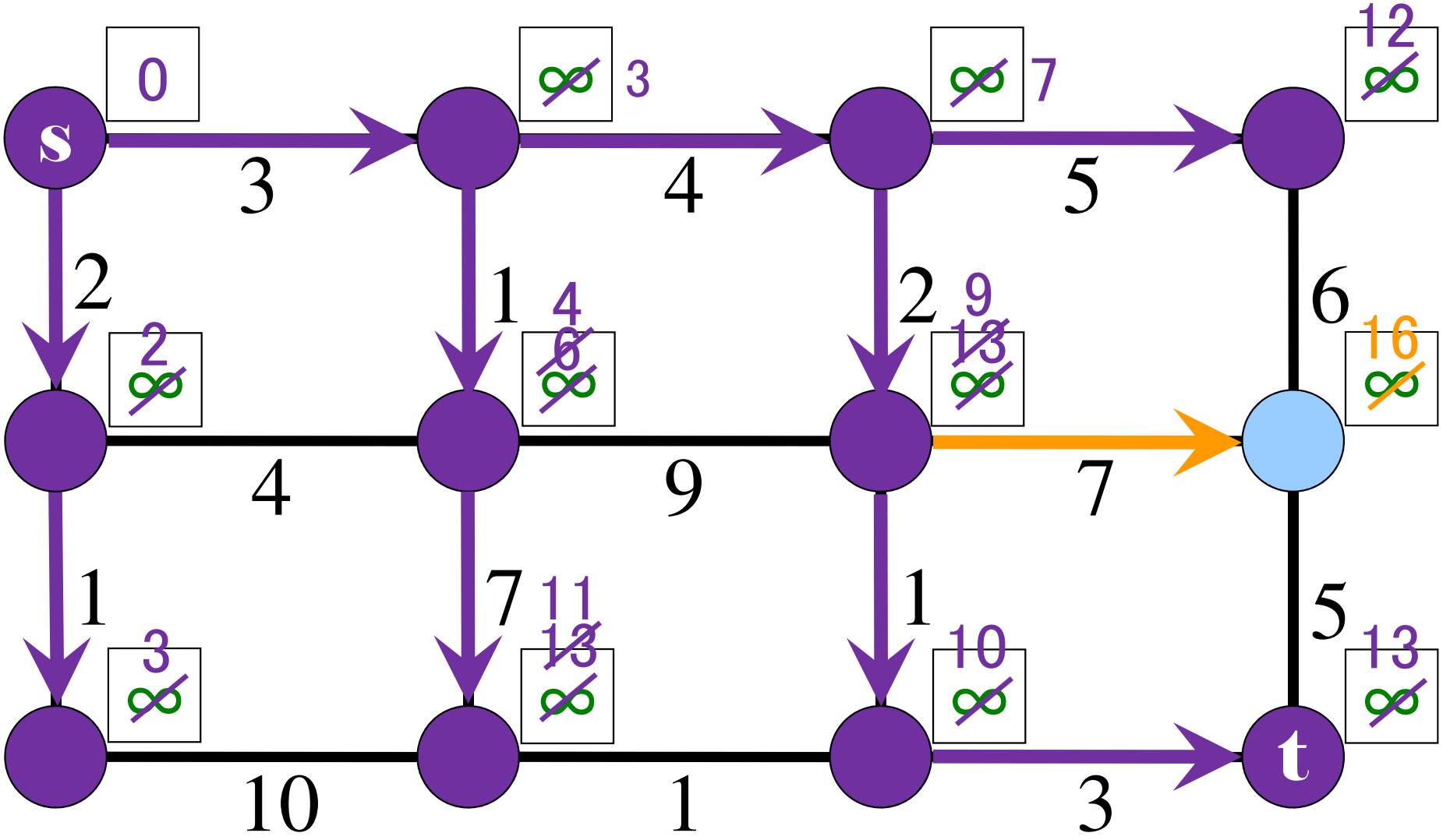


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

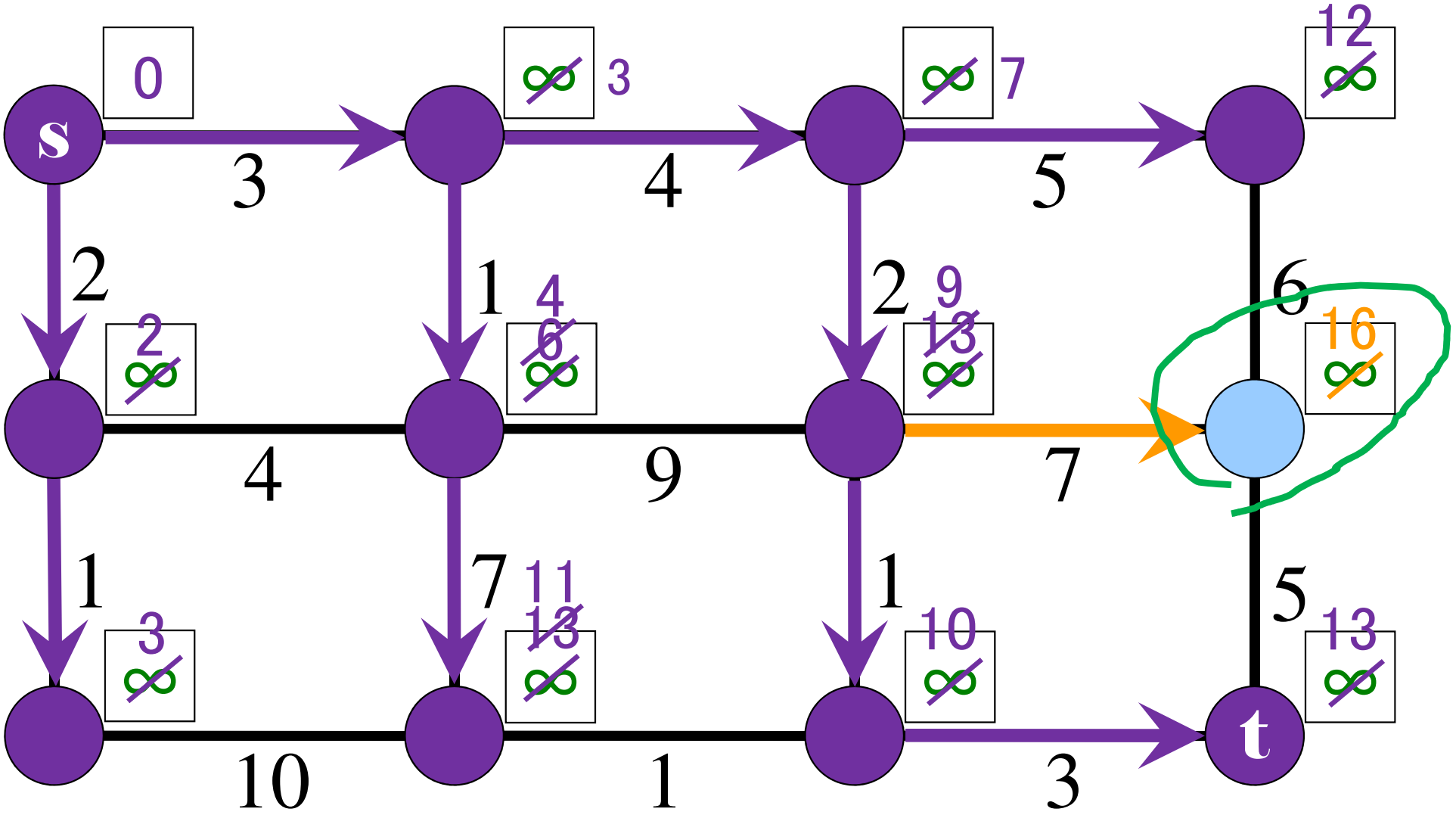
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



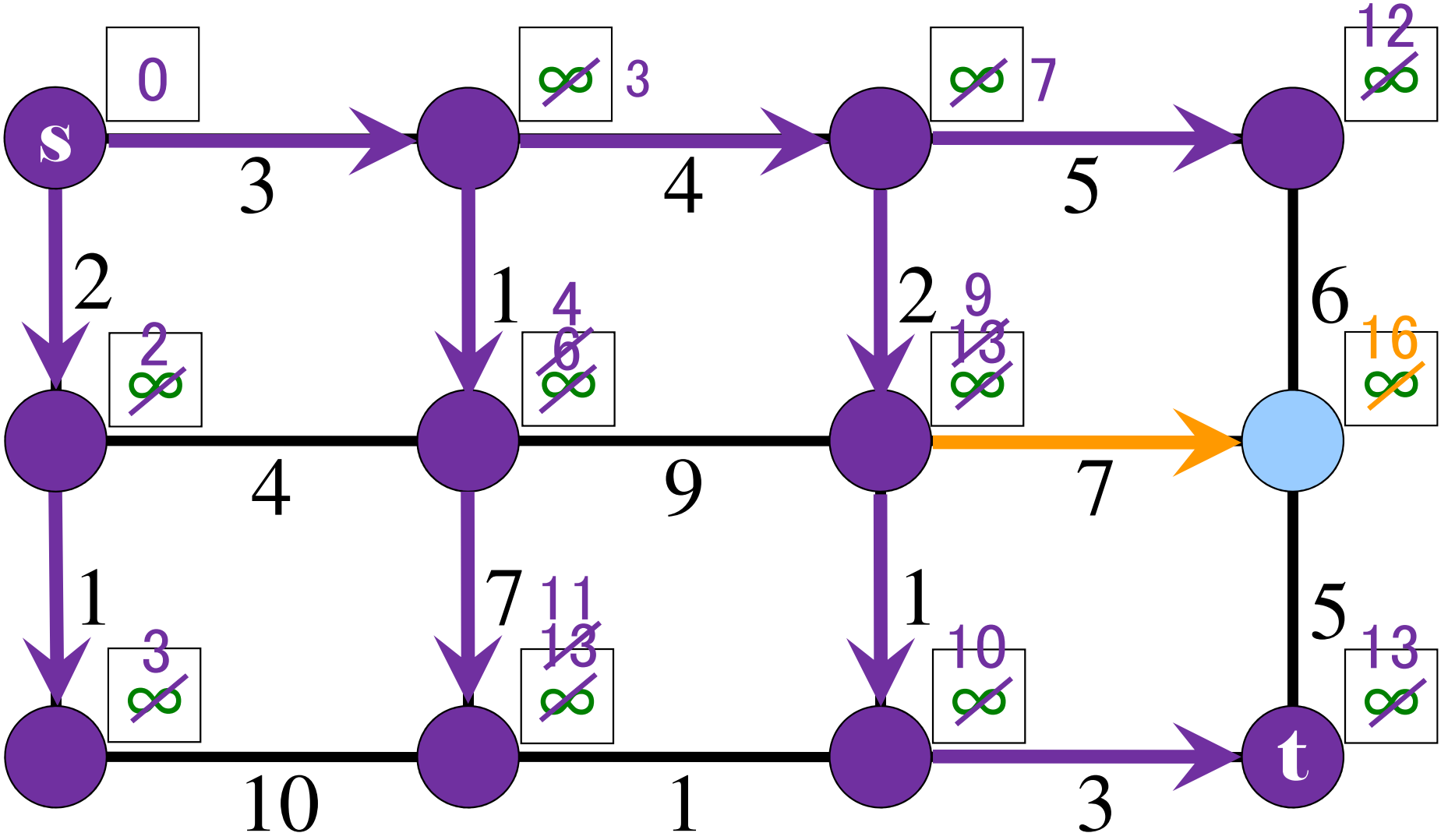
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

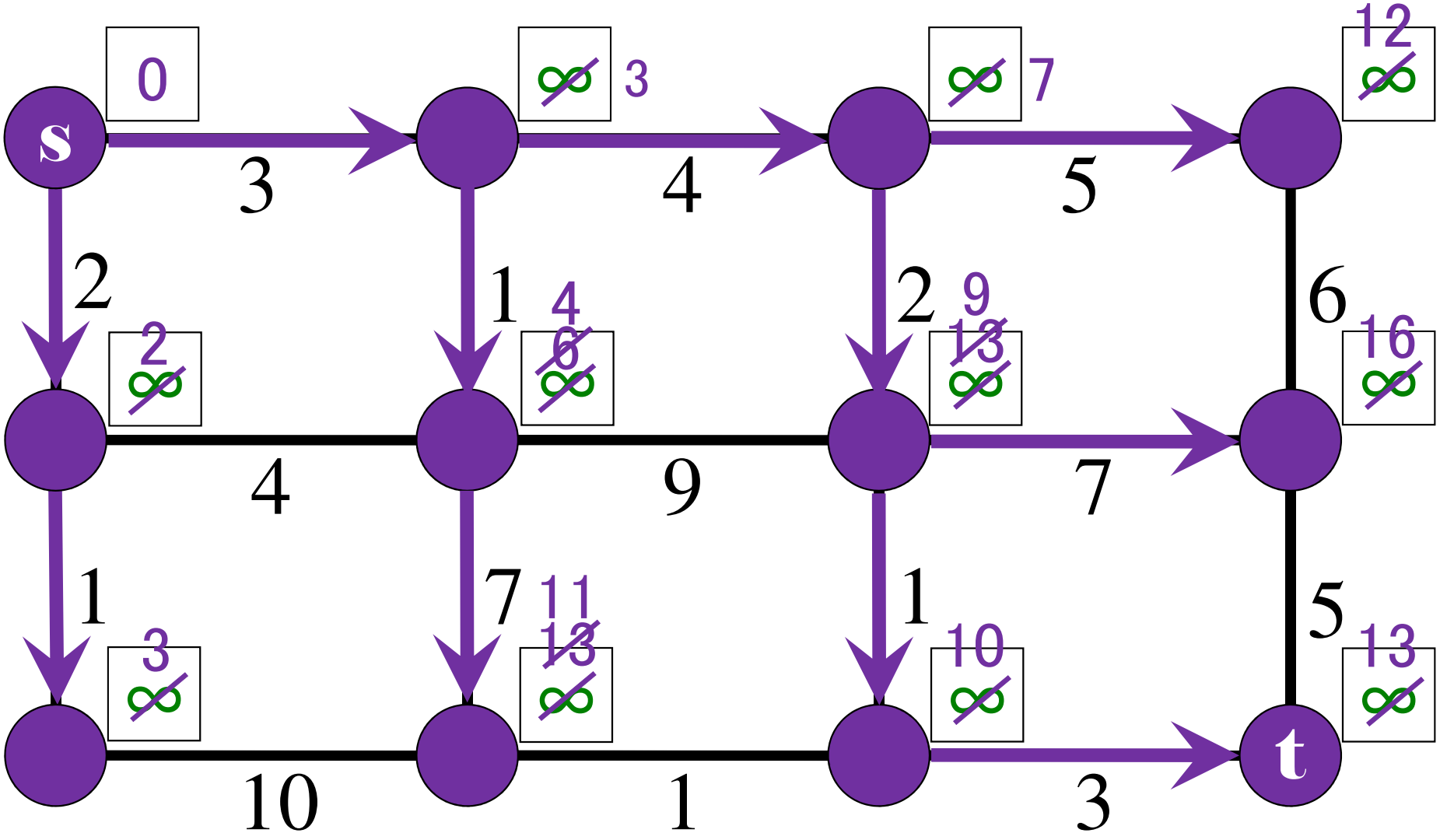


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

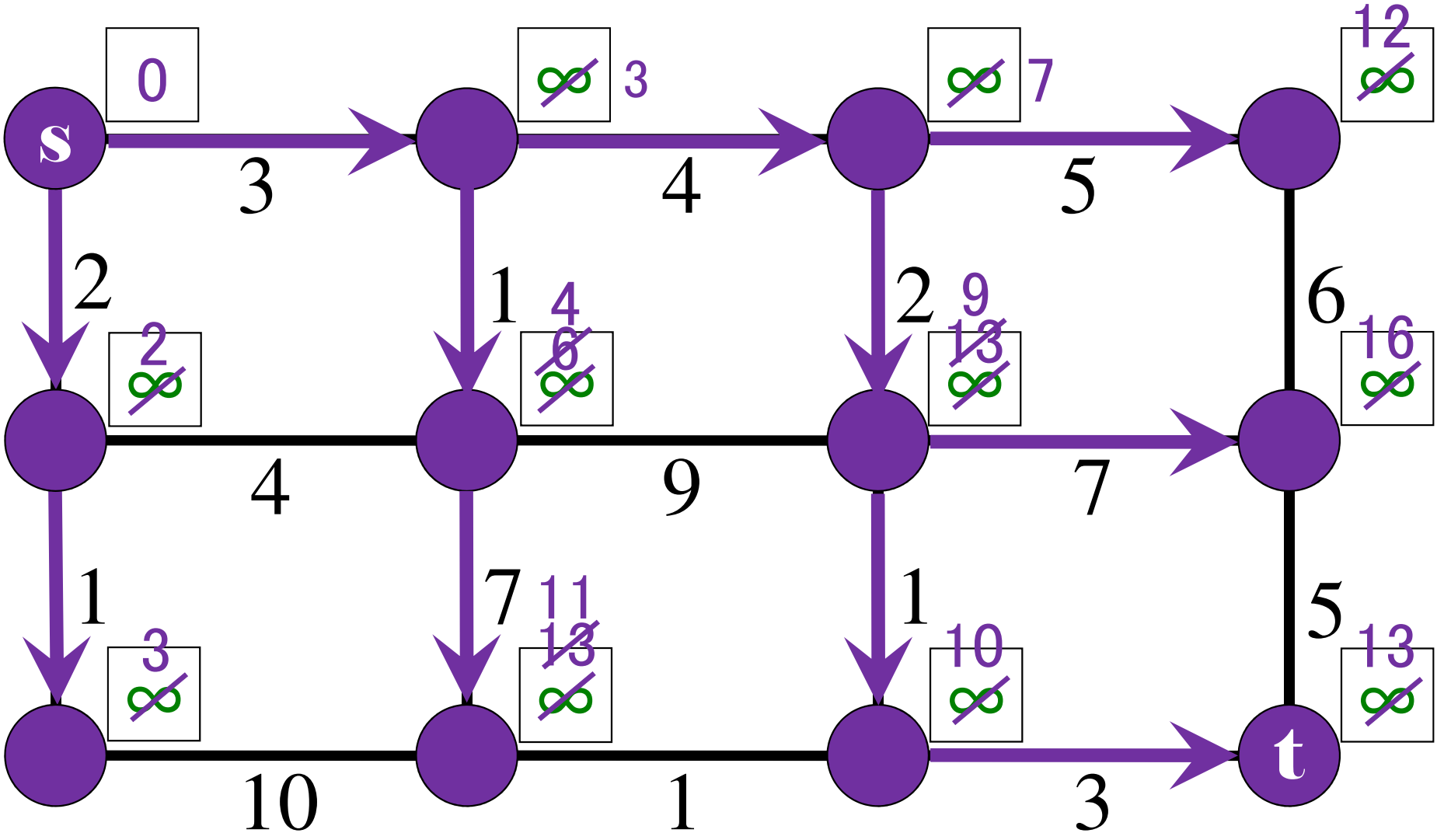
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



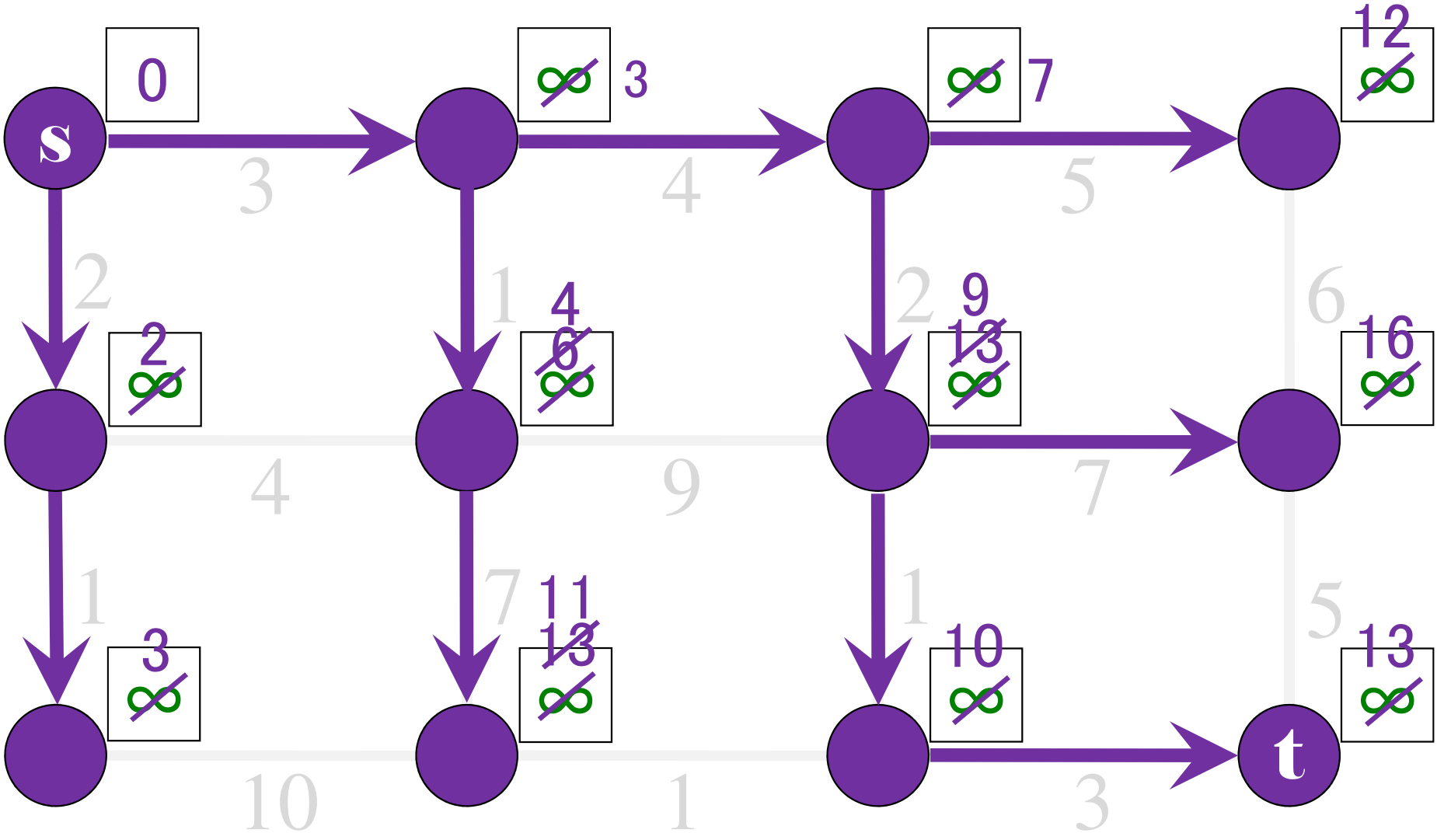
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



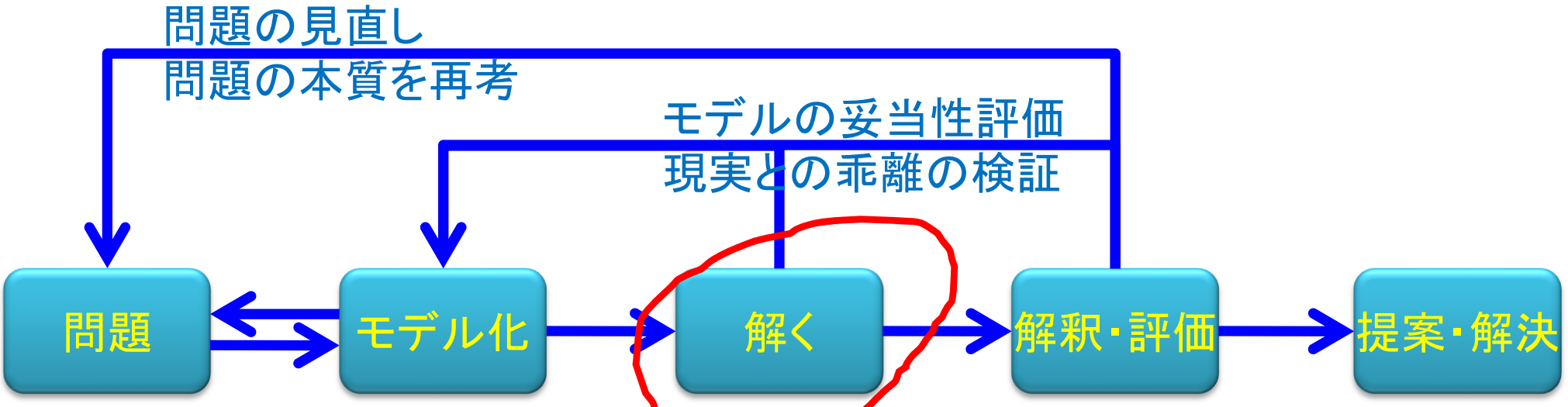
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

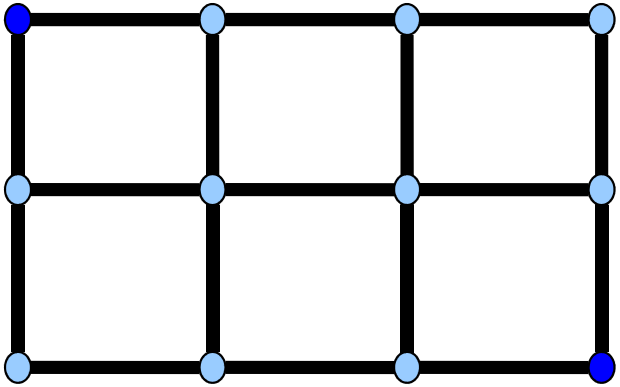
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

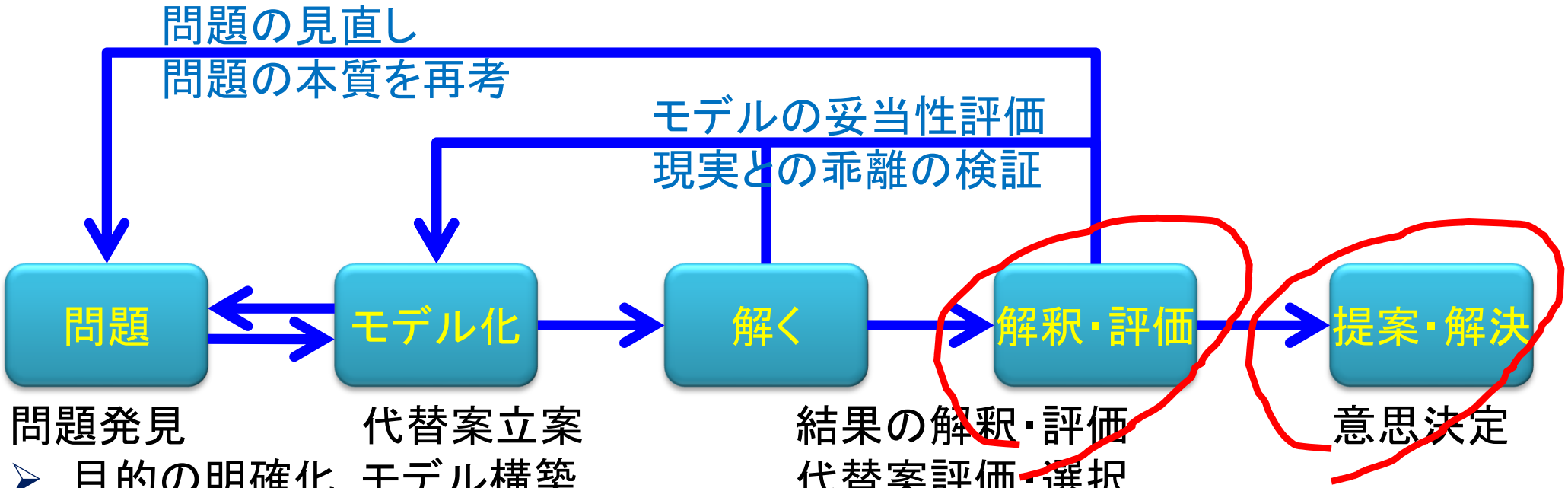
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

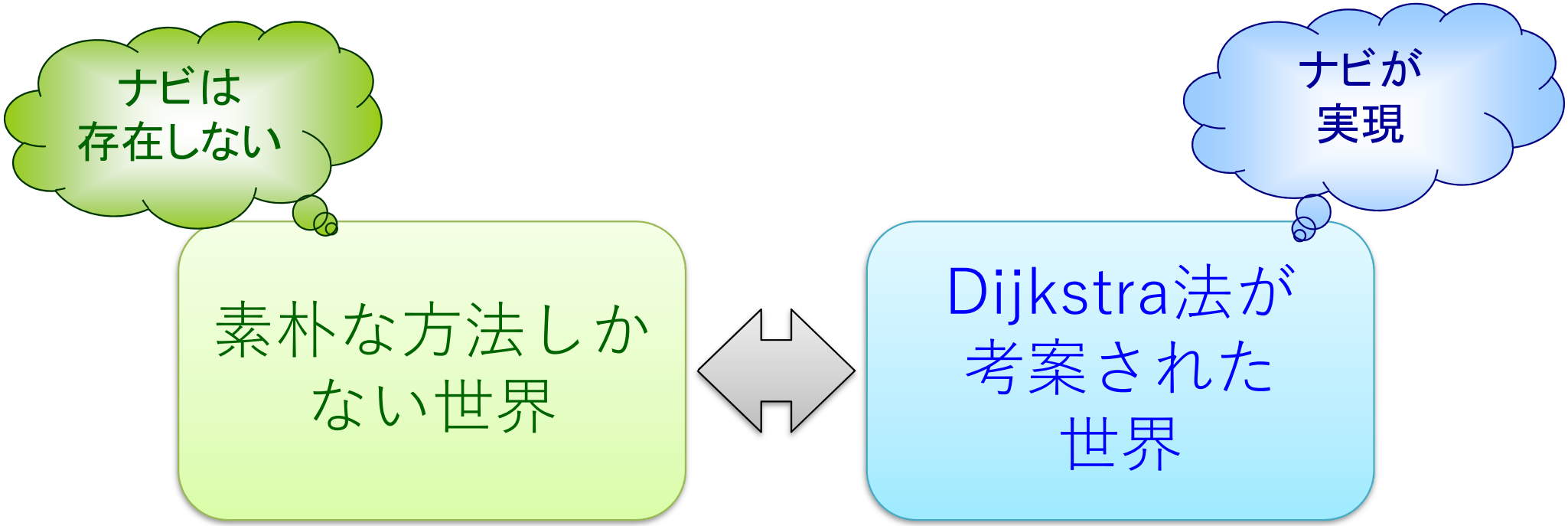
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

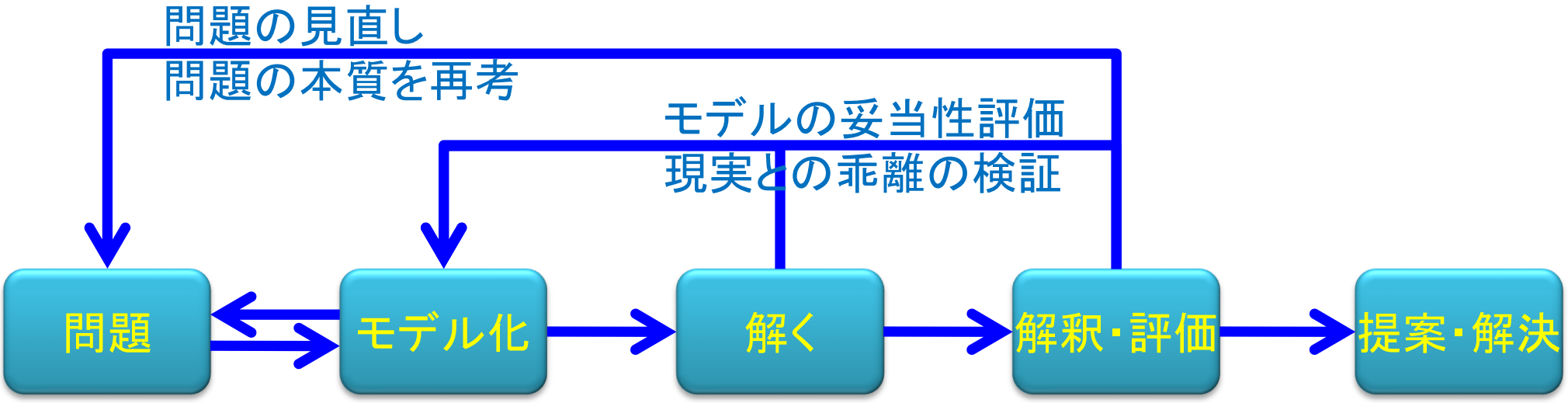
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



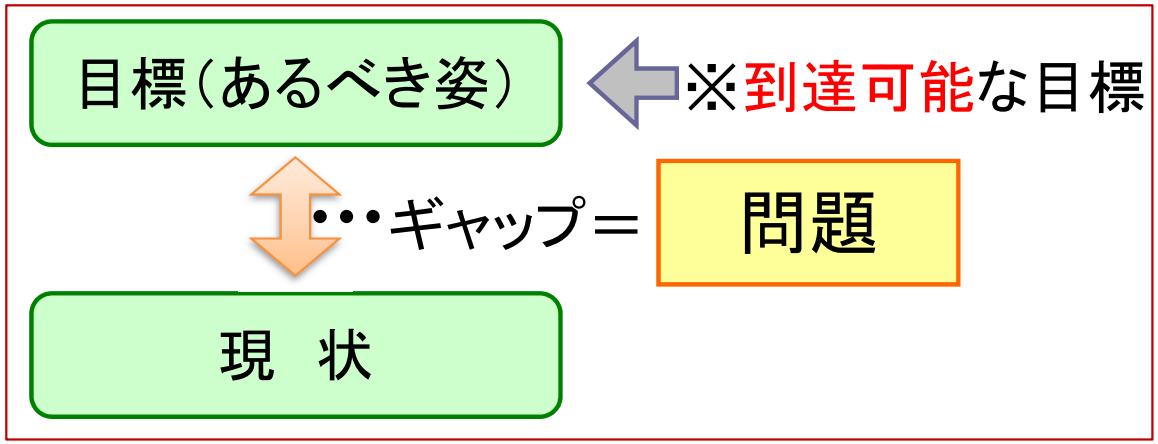
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

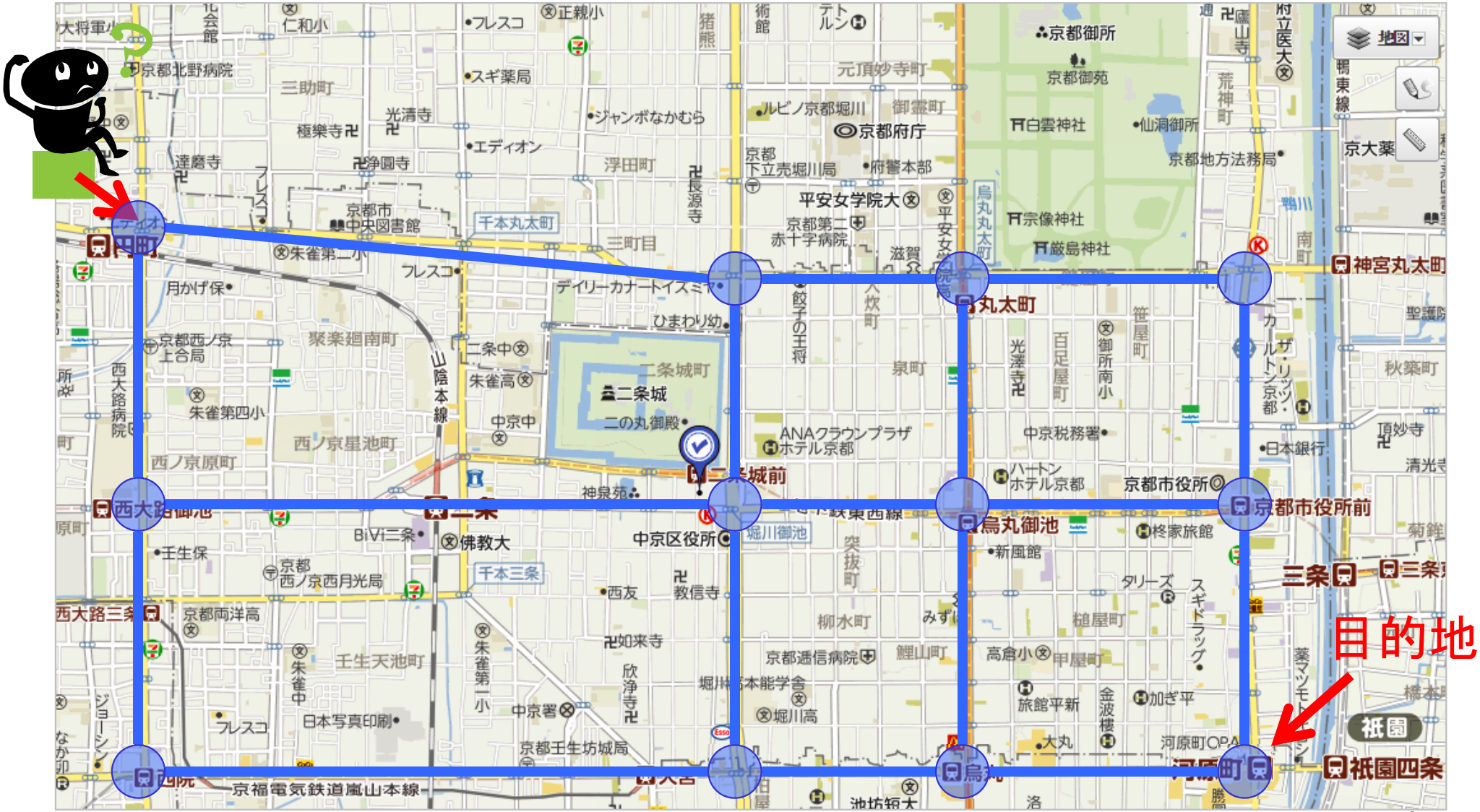
意思決定

問題の定義



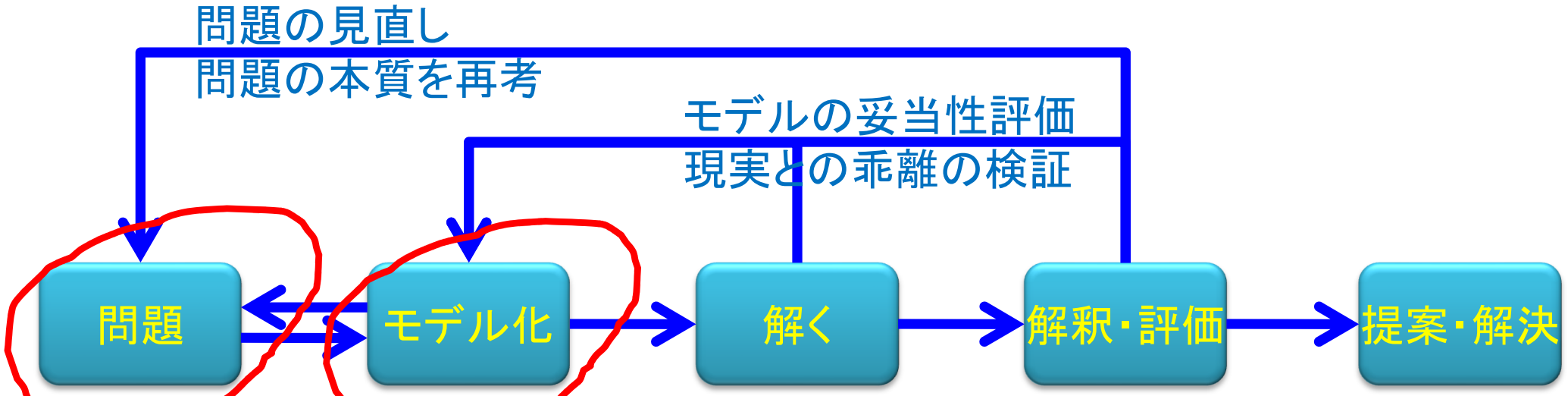
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

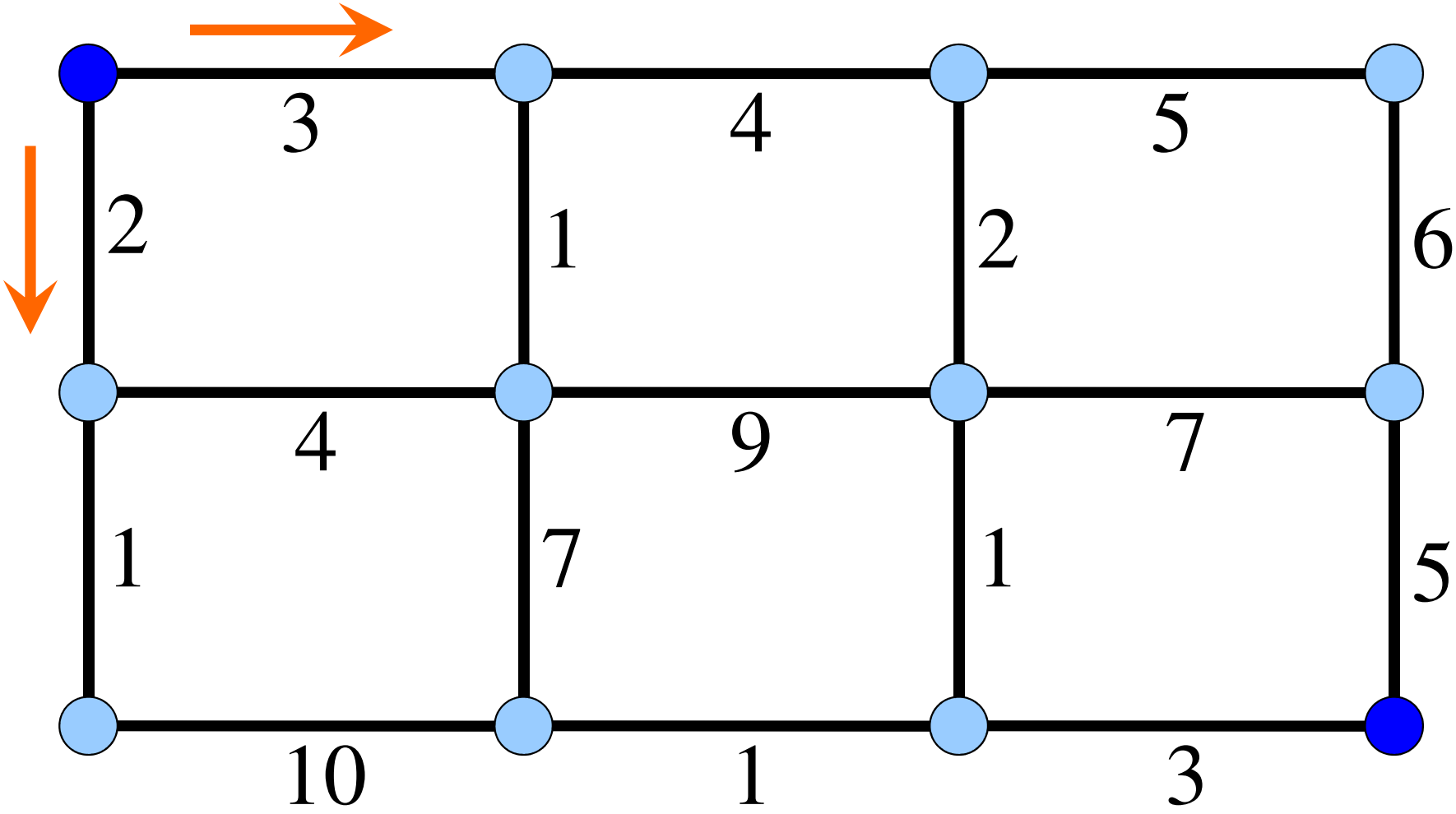
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

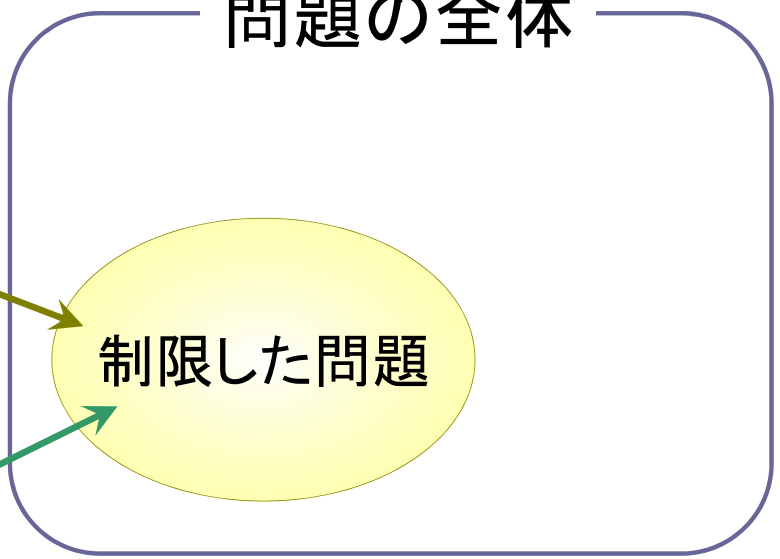


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

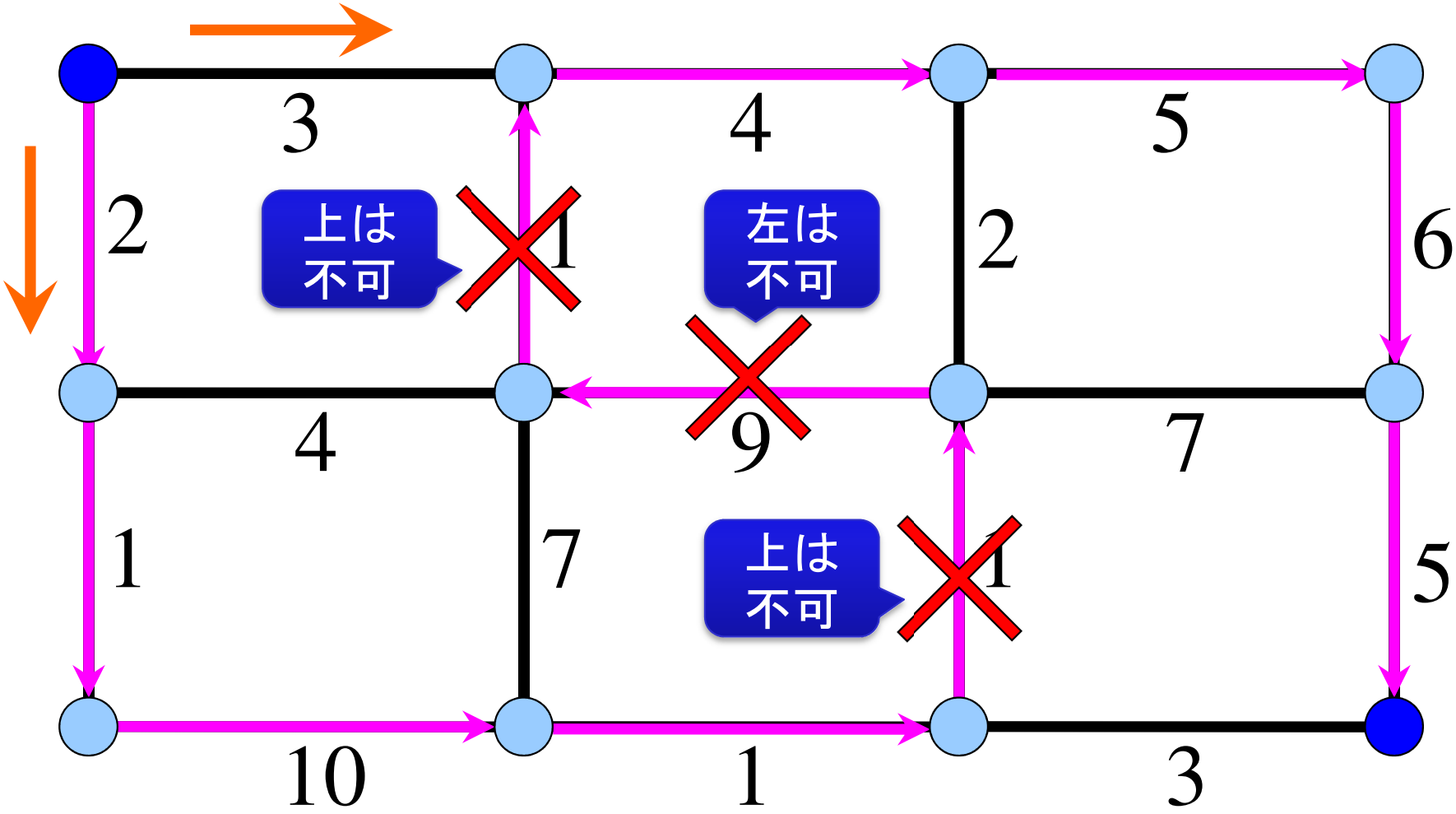
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

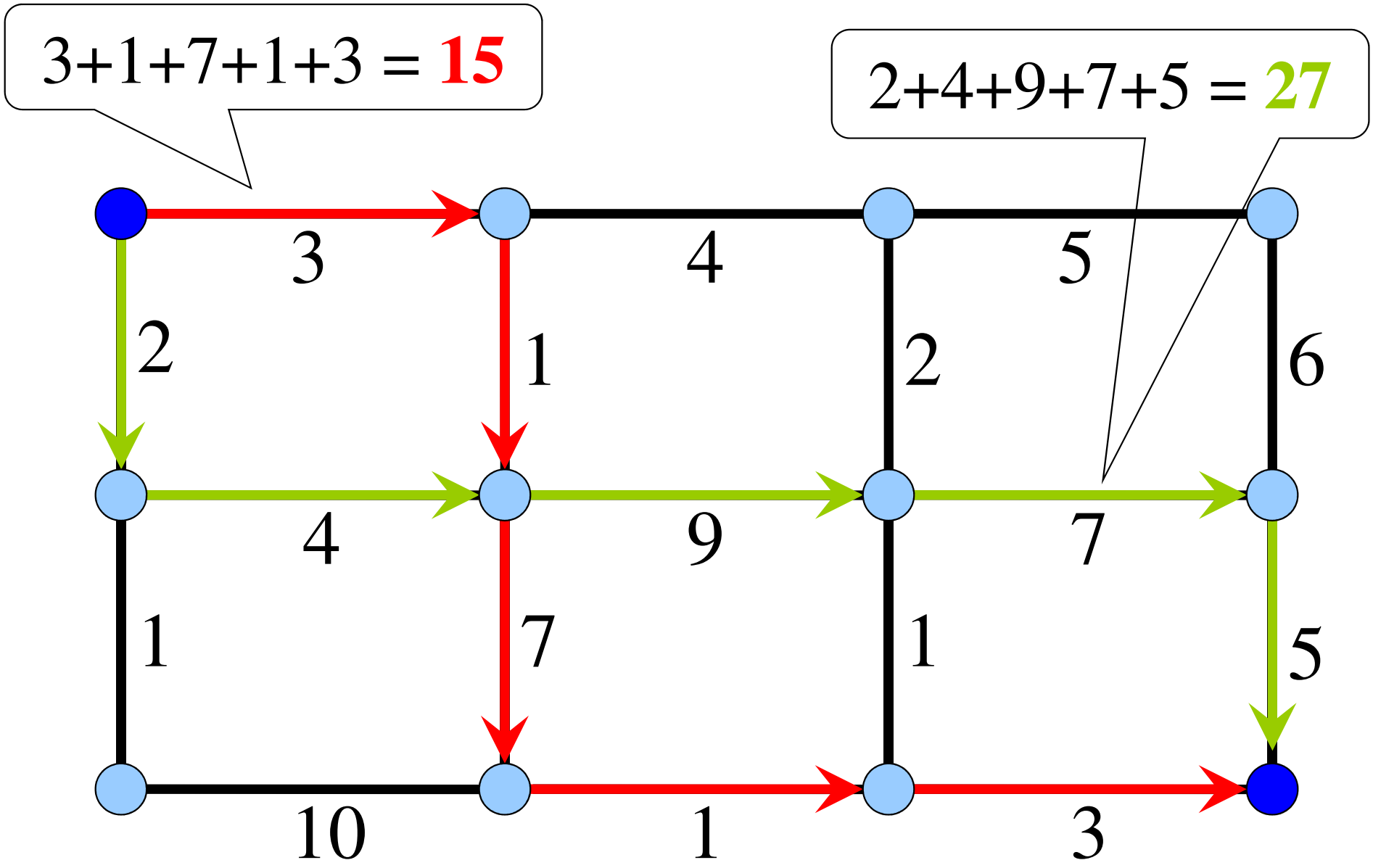
難しいなら易しくすればいいのさ！

制限した問題

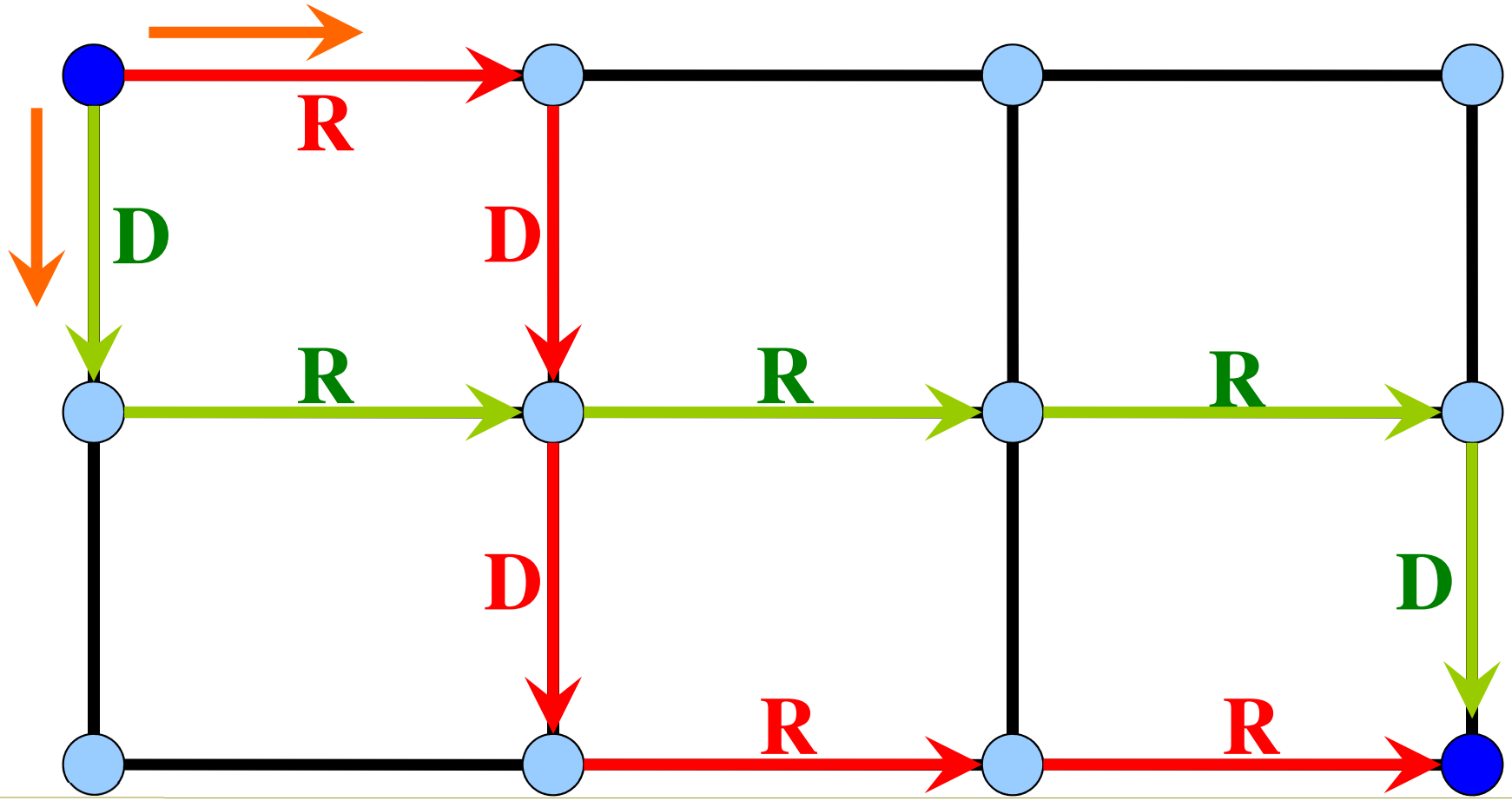
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



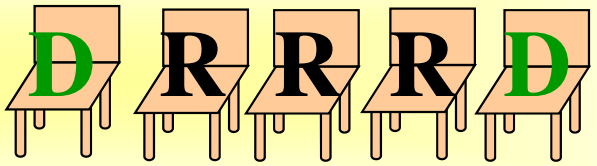
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



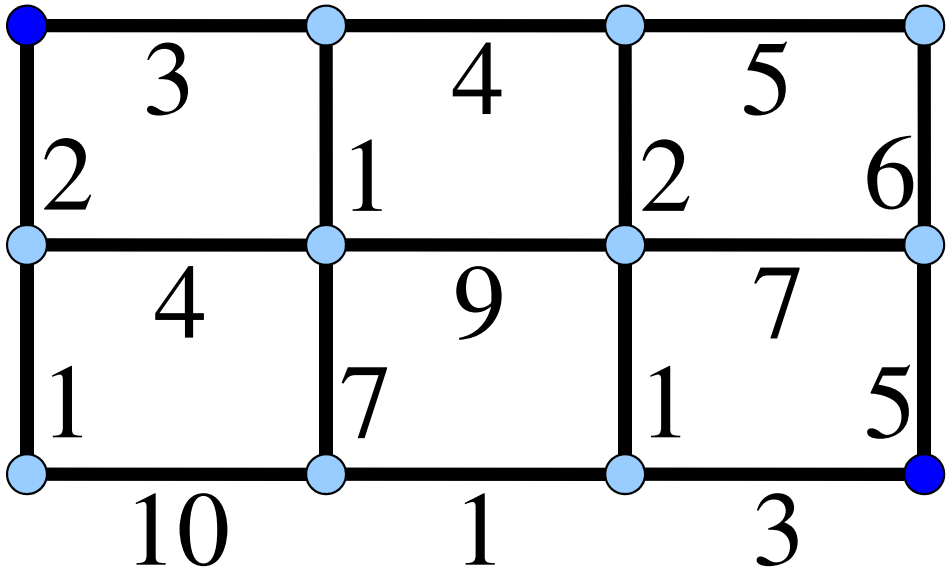
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

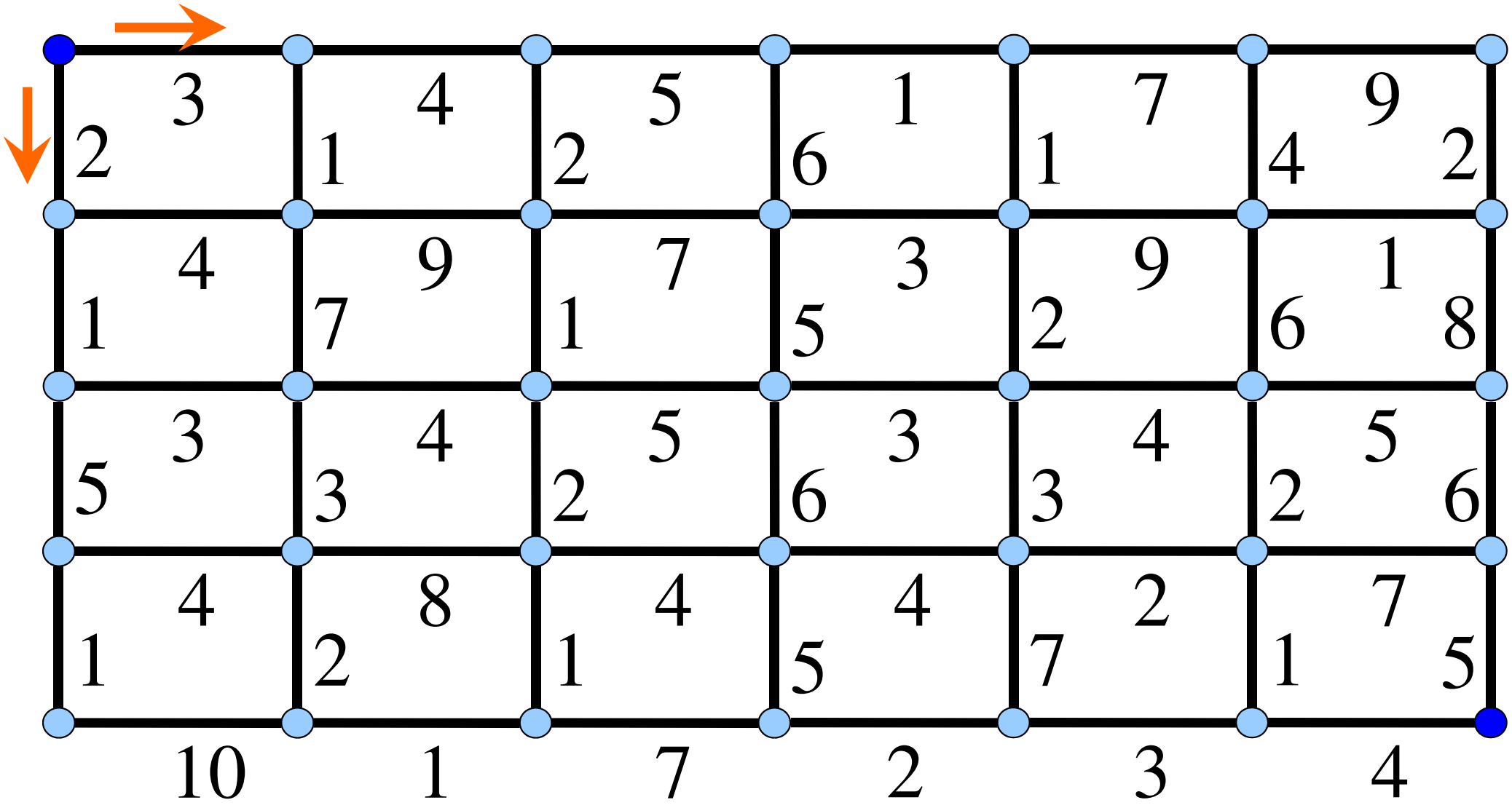
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

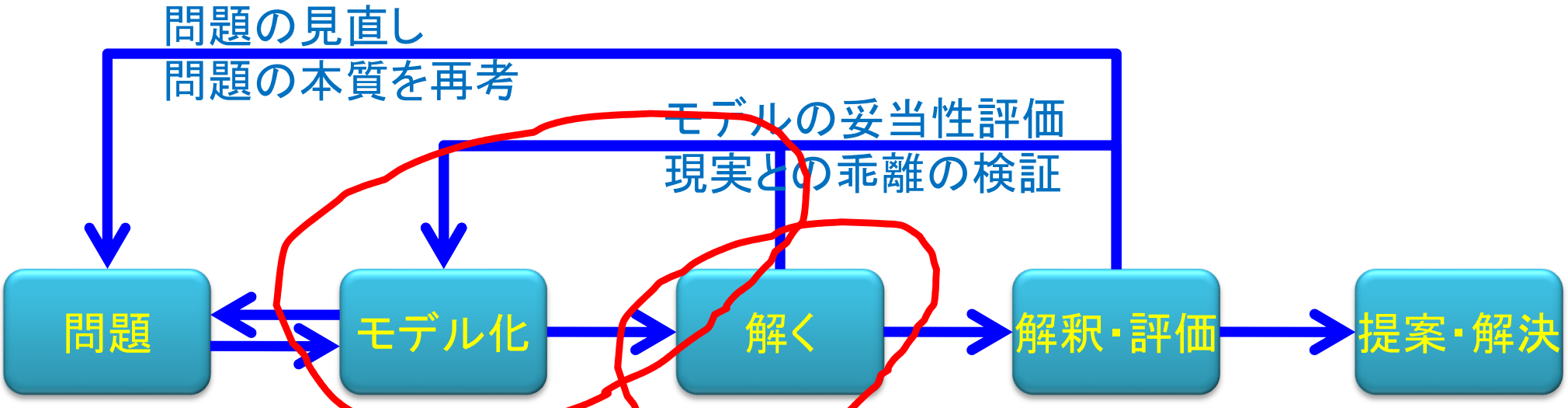
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

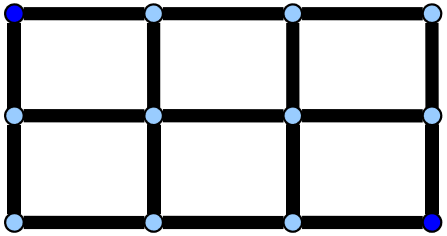
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

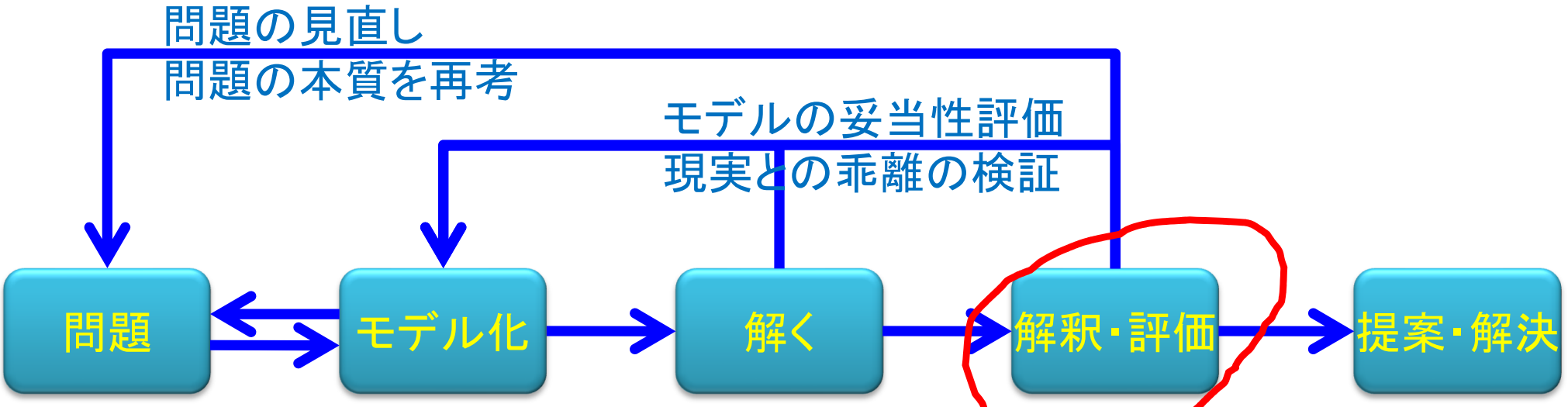
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

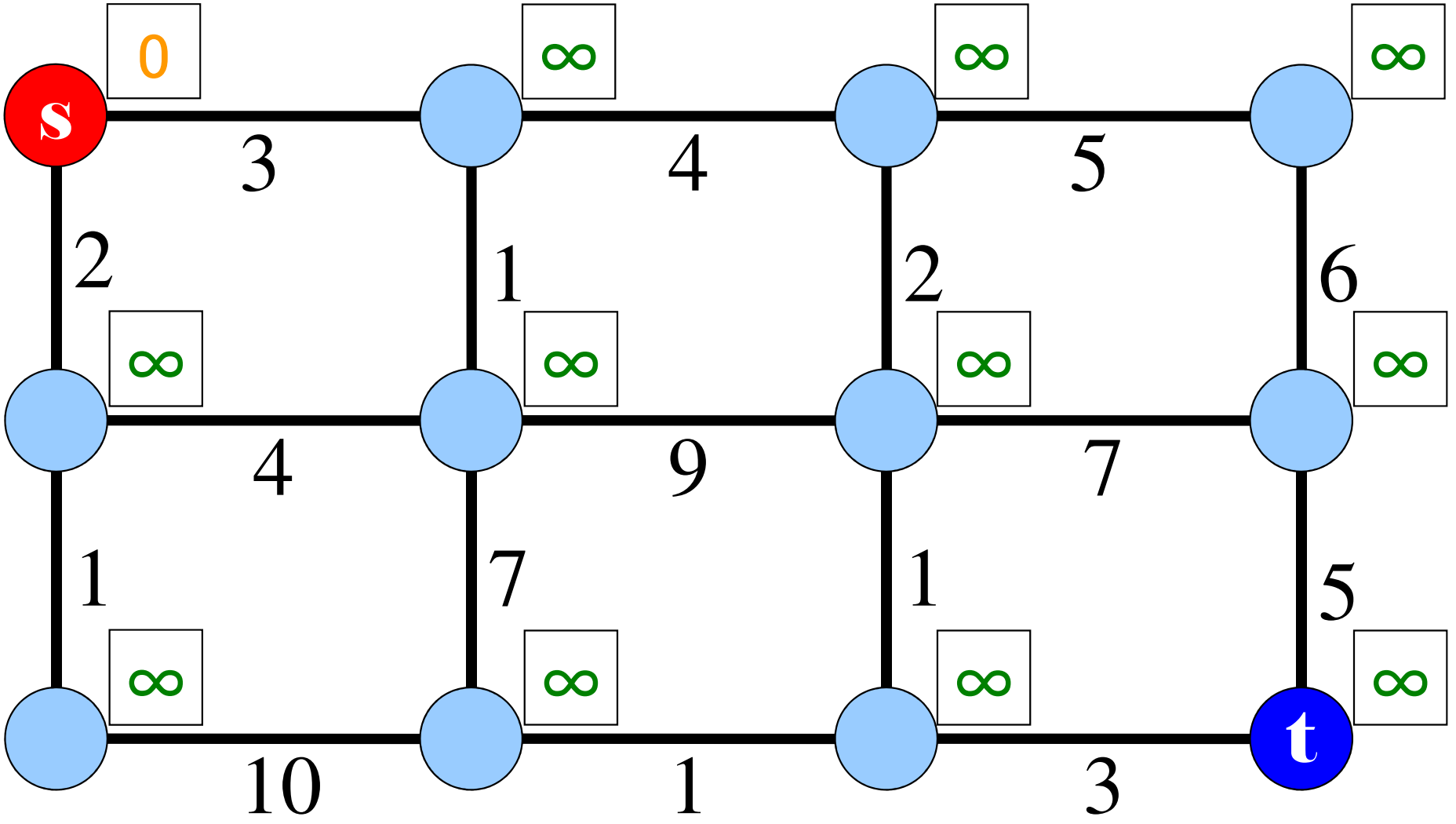
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

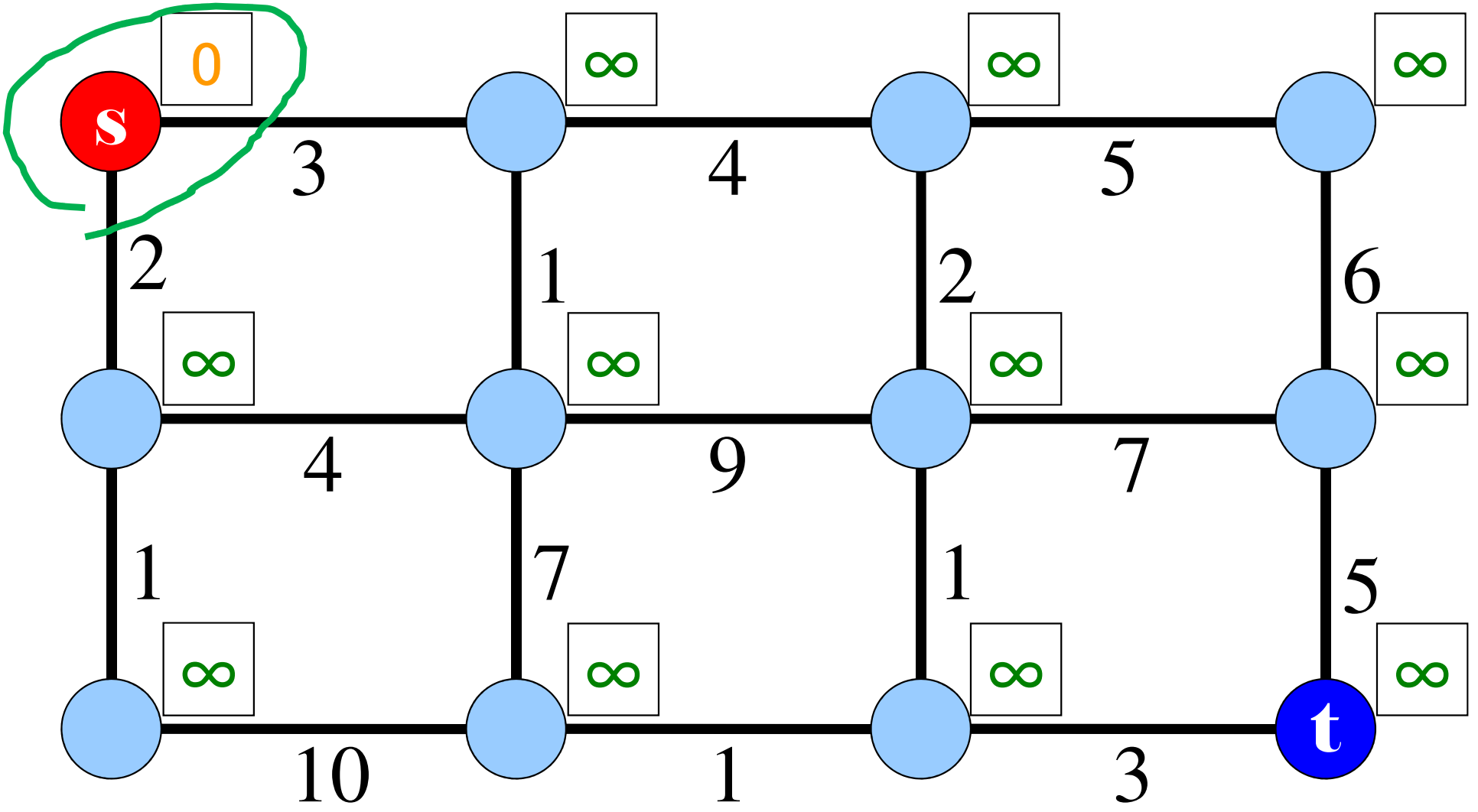
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



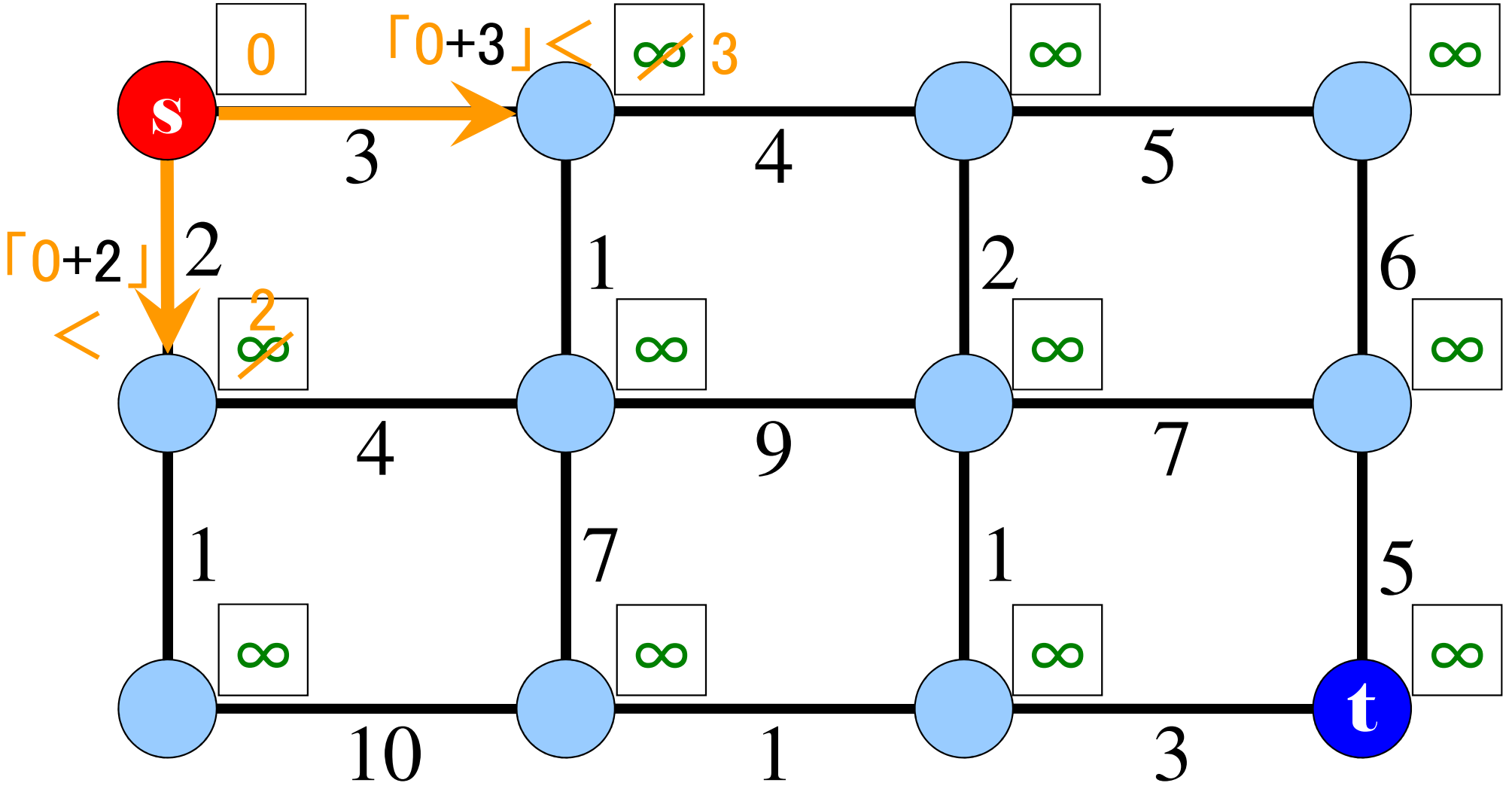
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

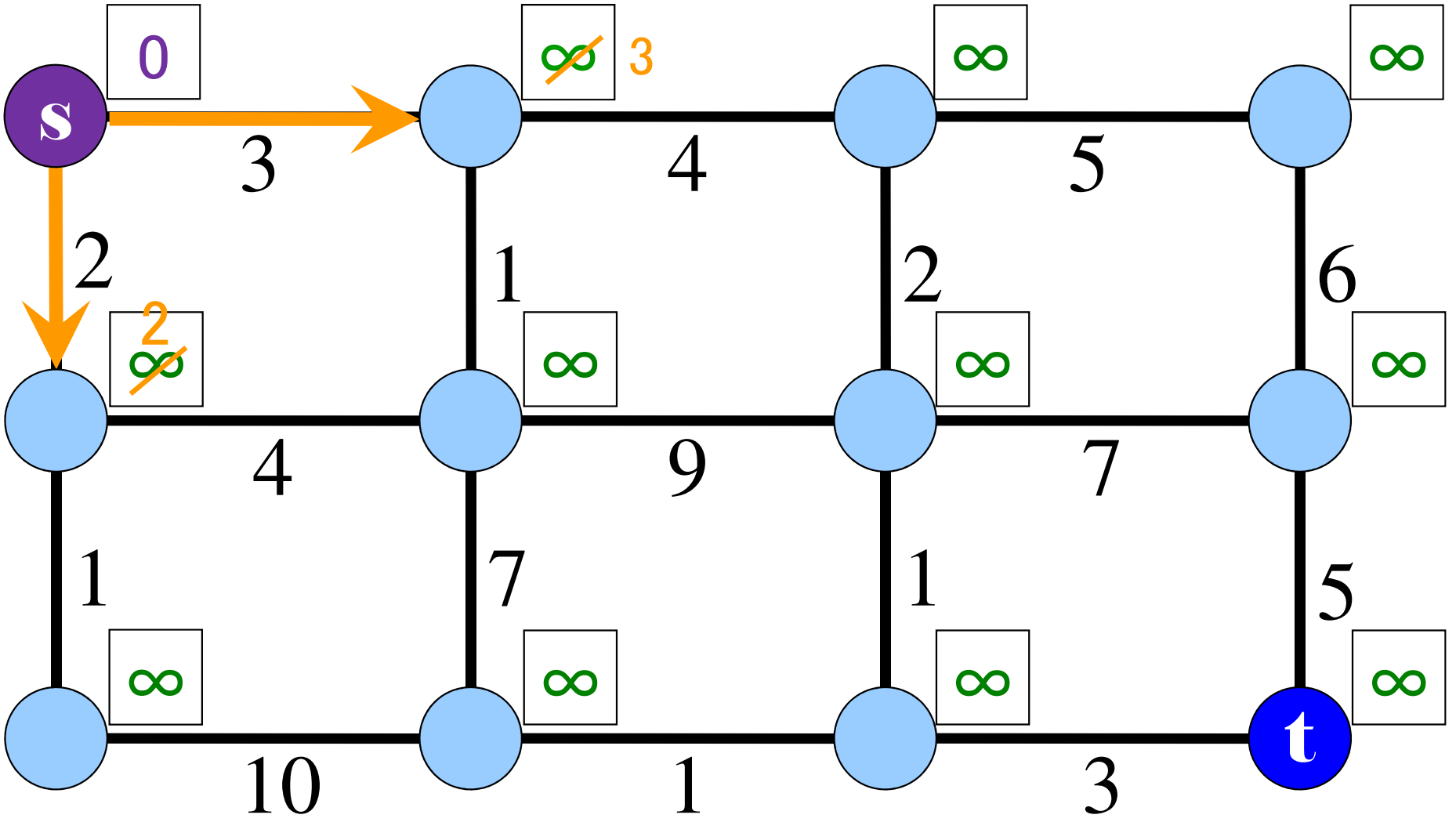


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

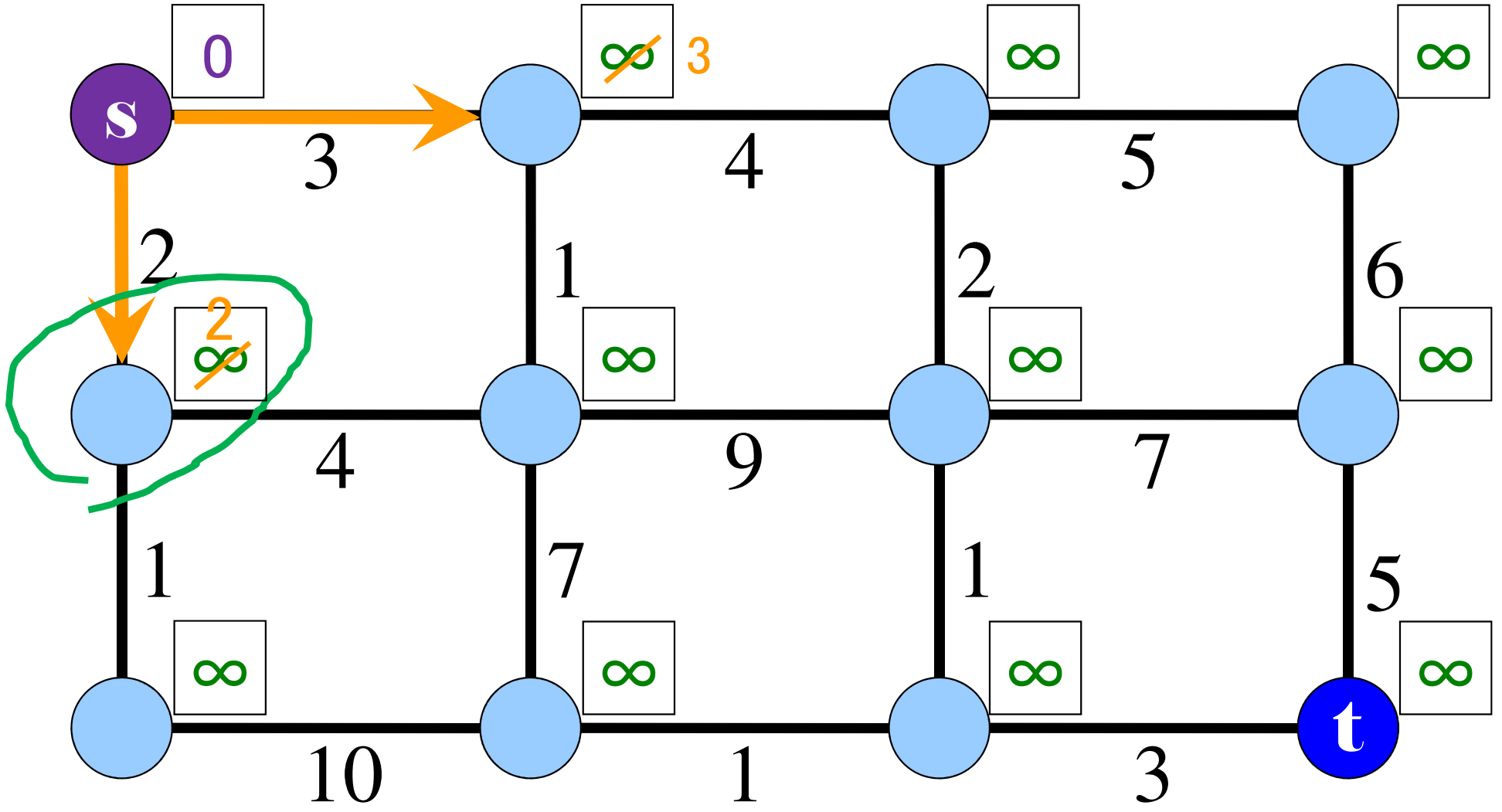
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



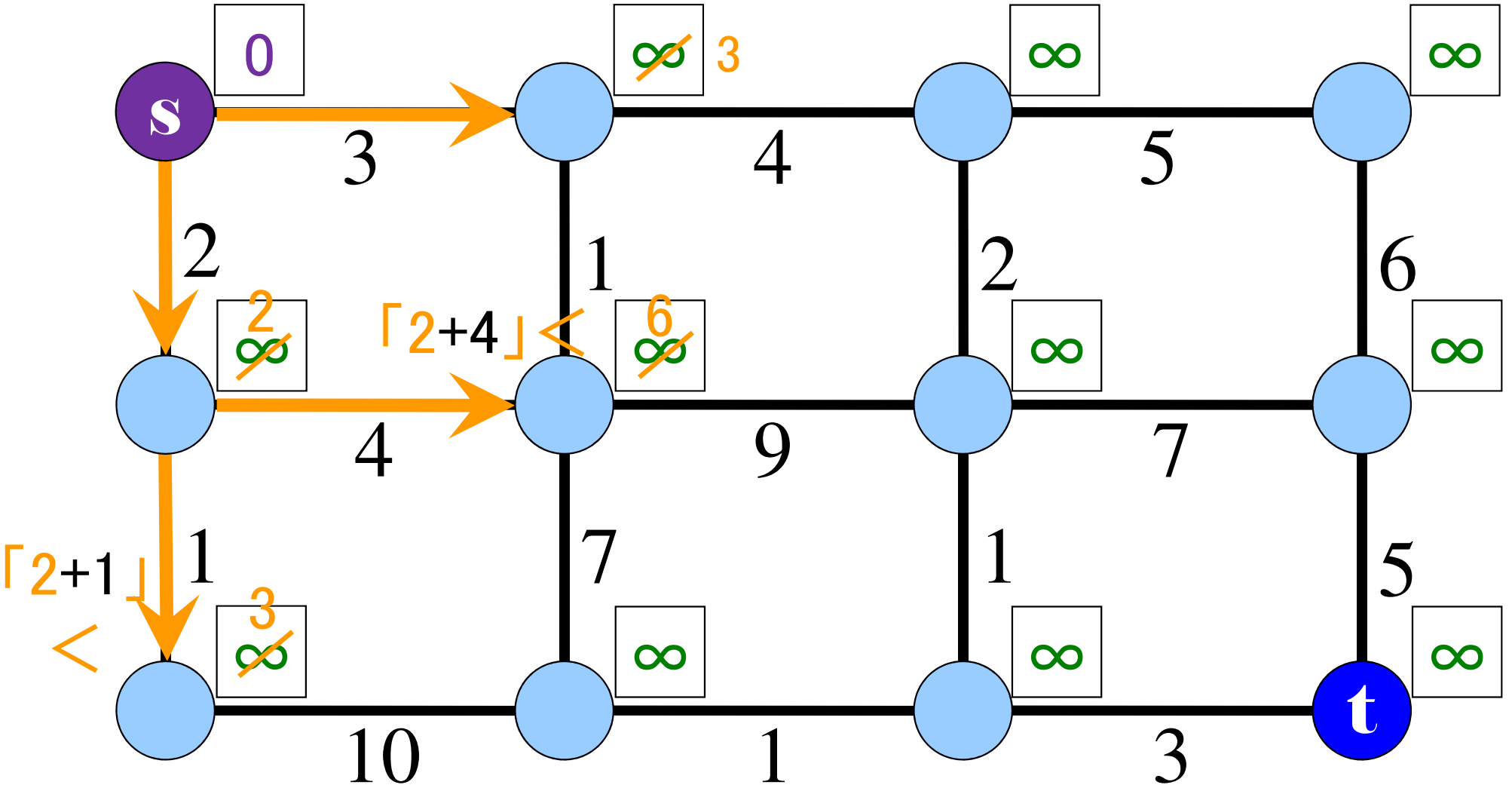
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

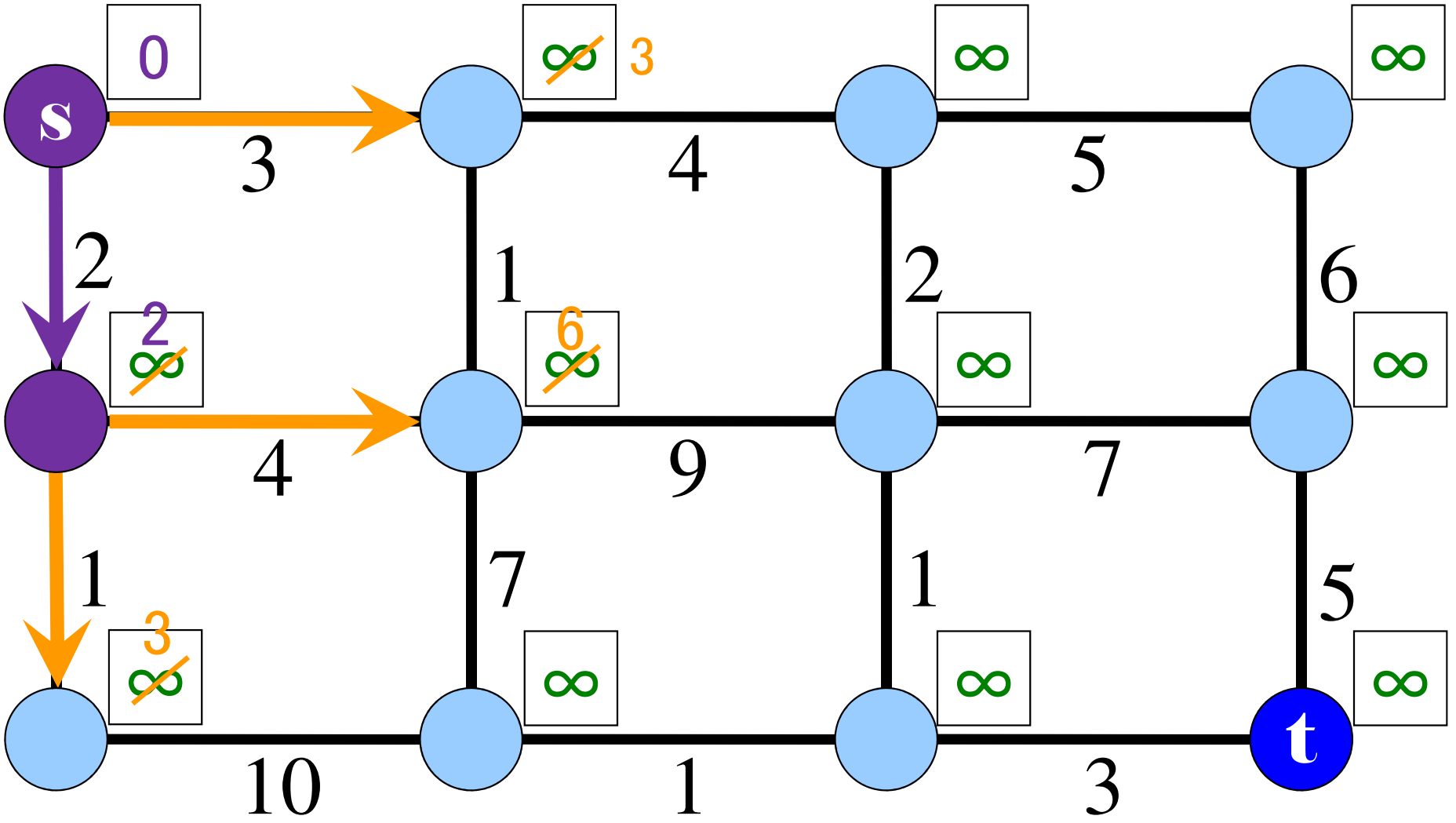


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

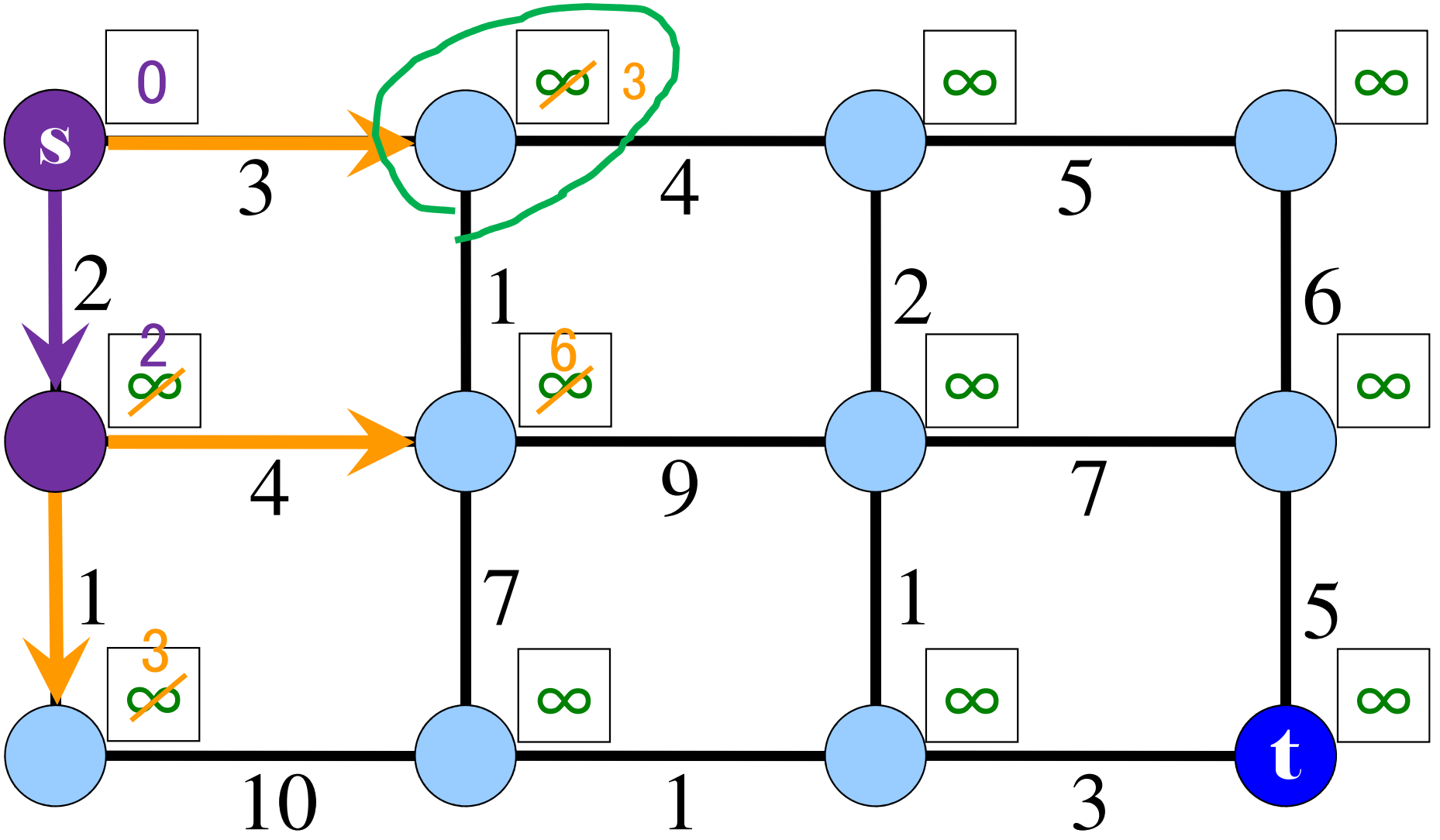
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



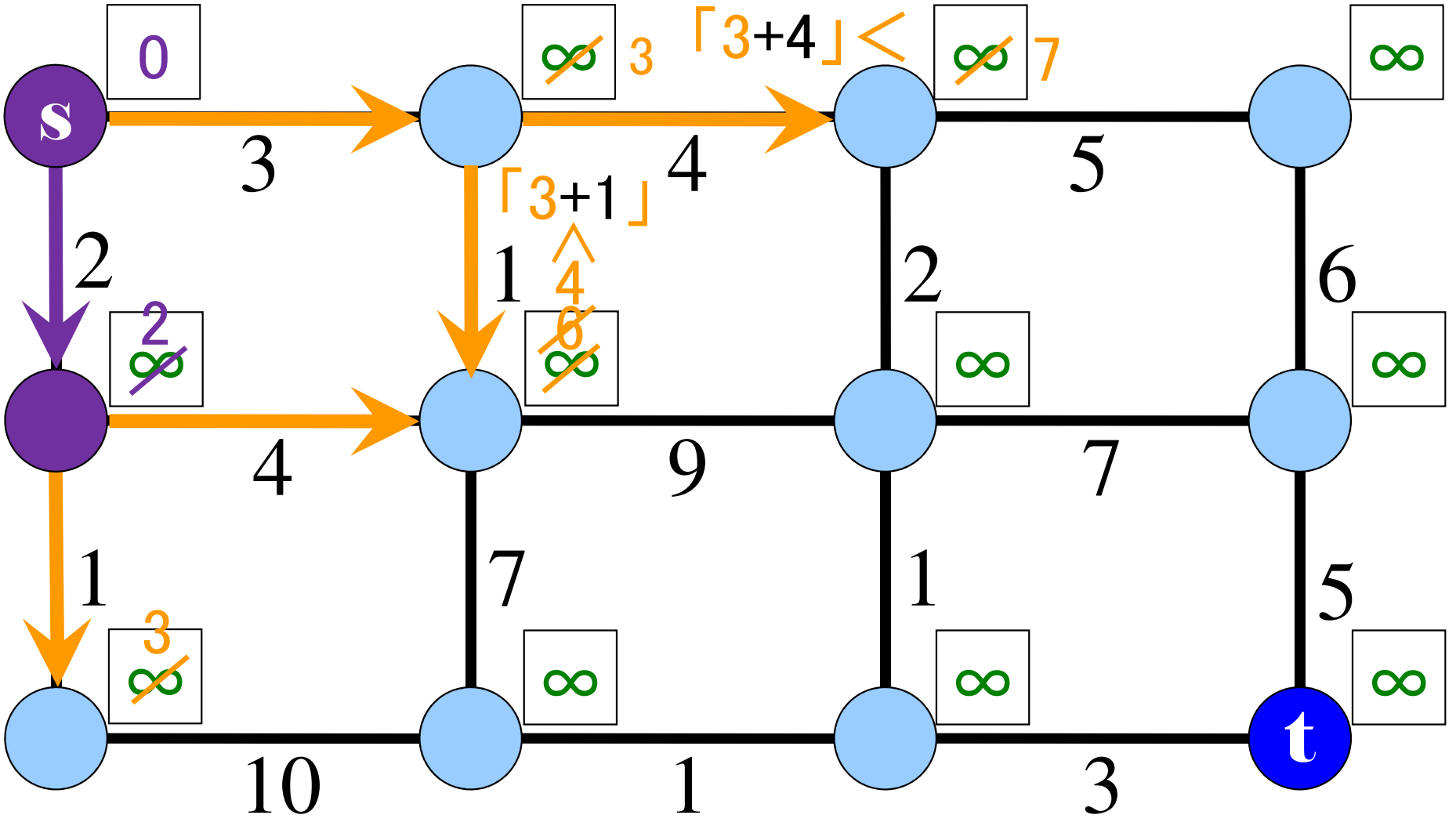
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

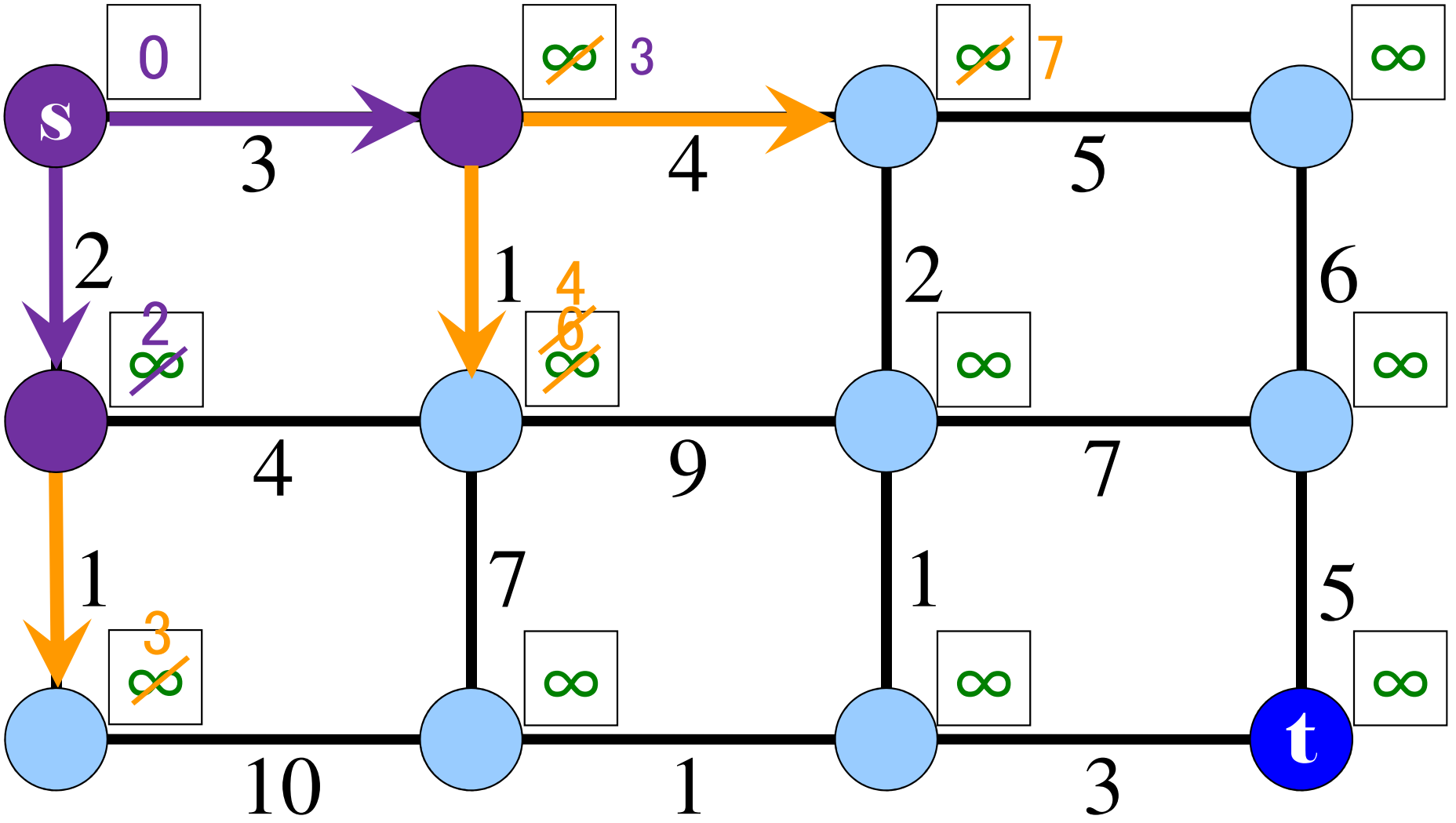


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

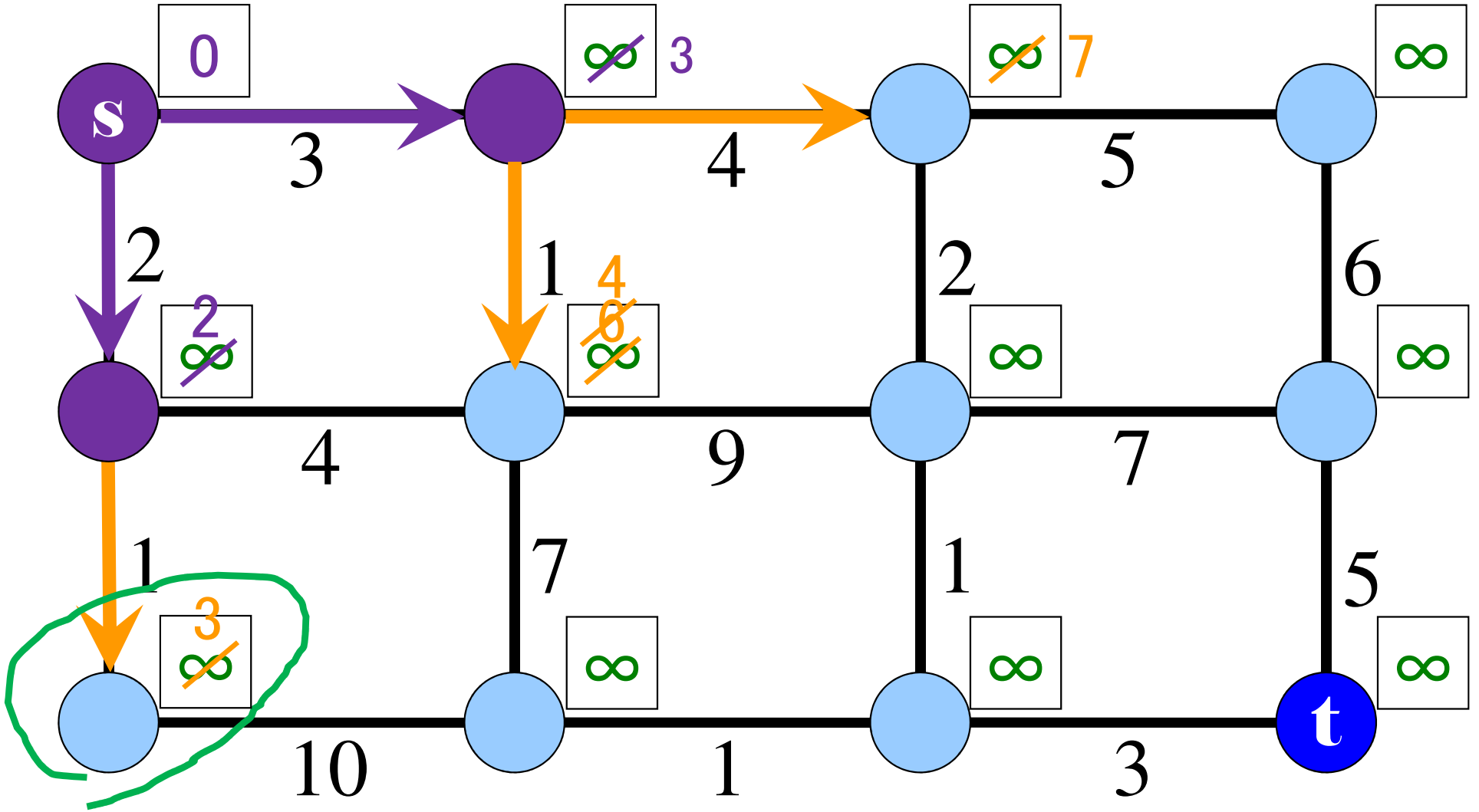
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



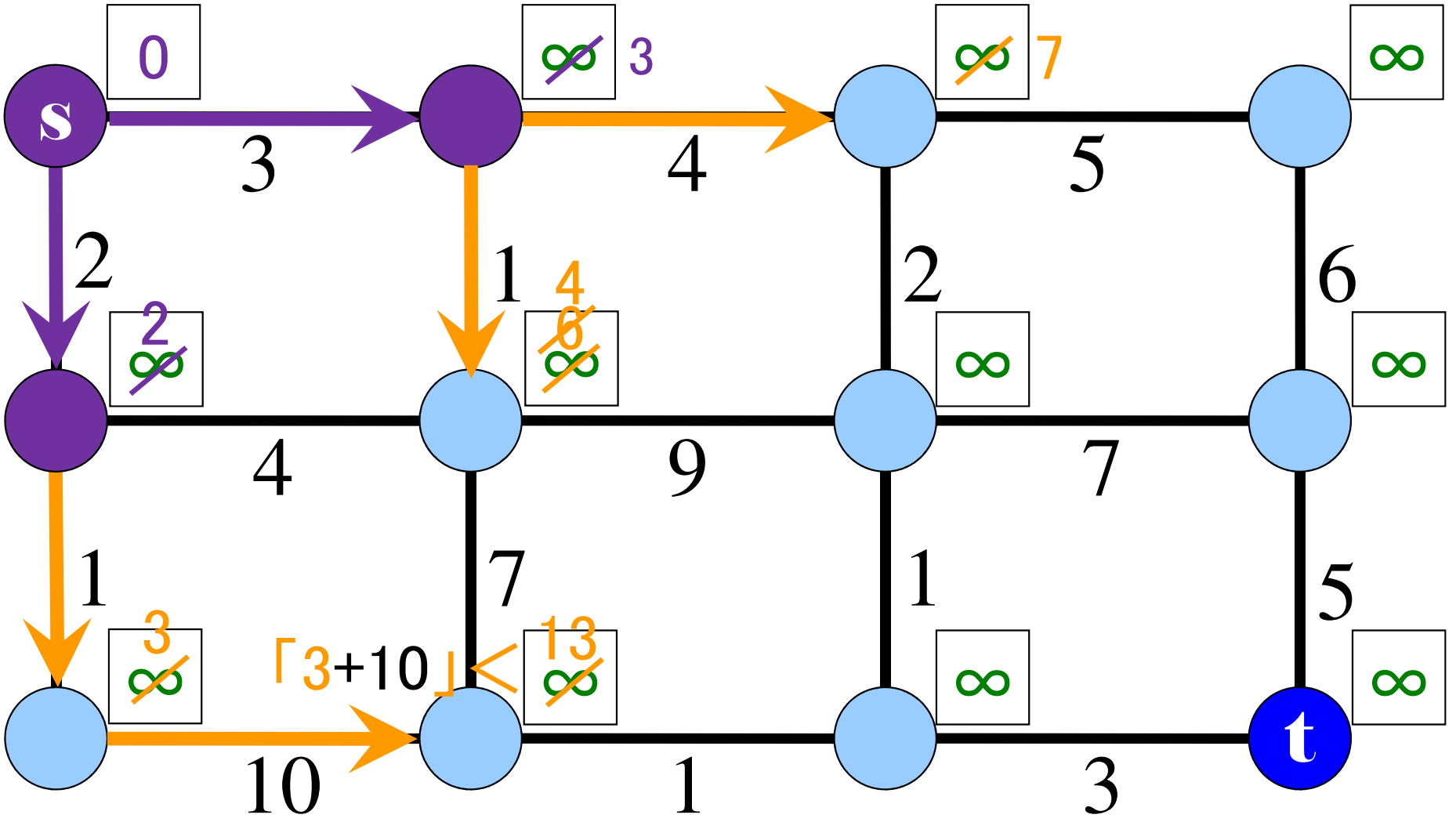
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

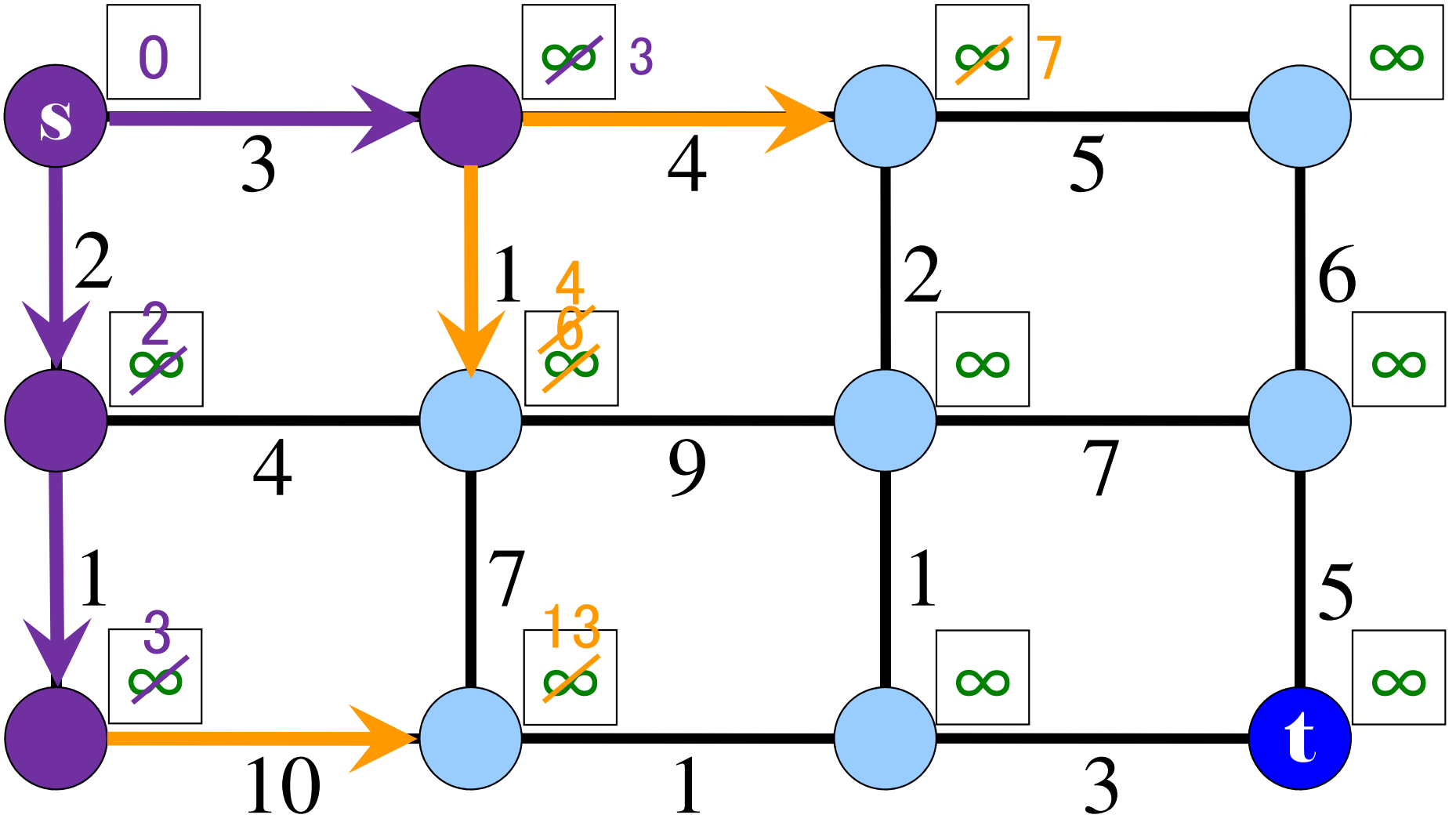


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

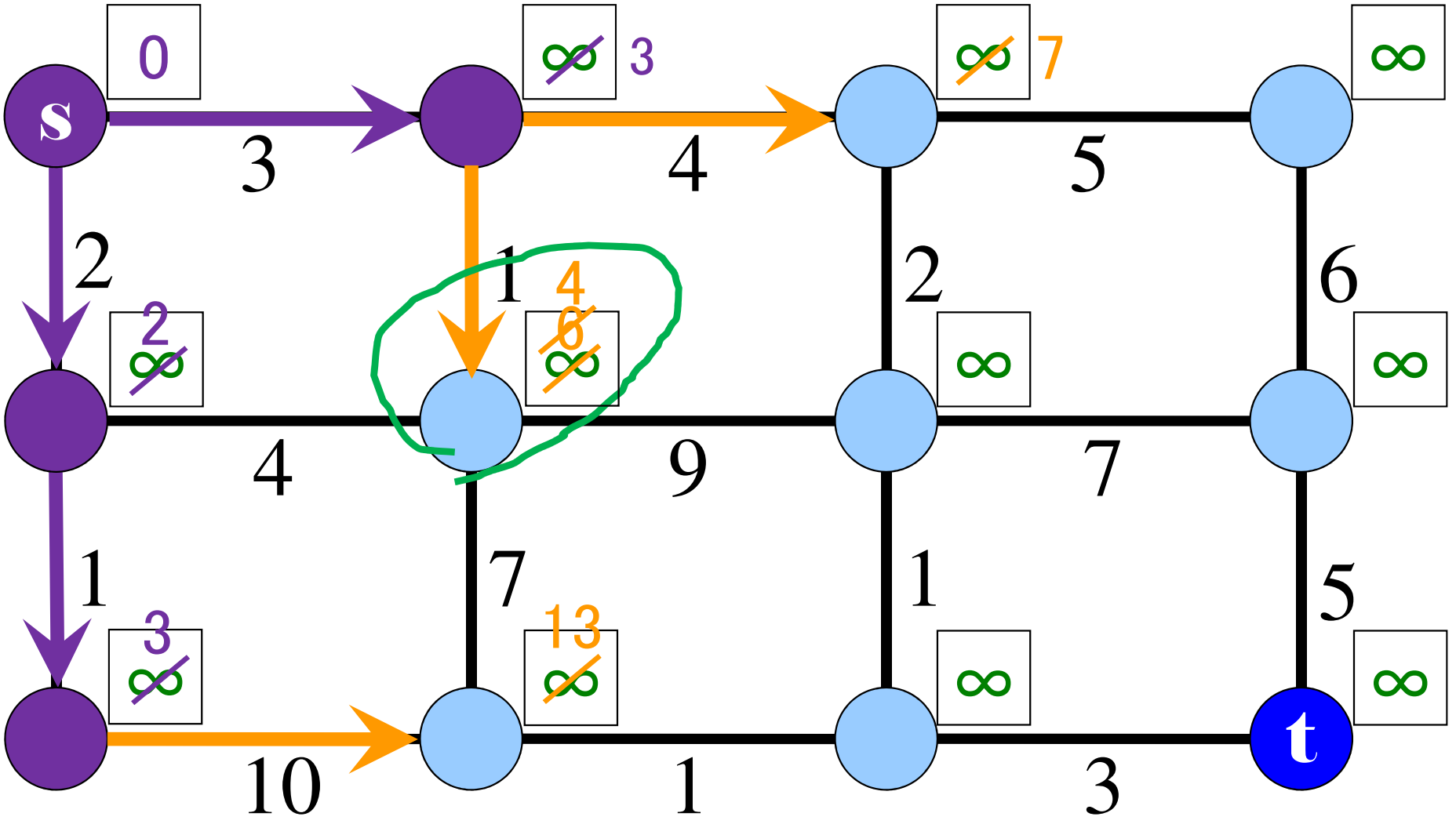
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



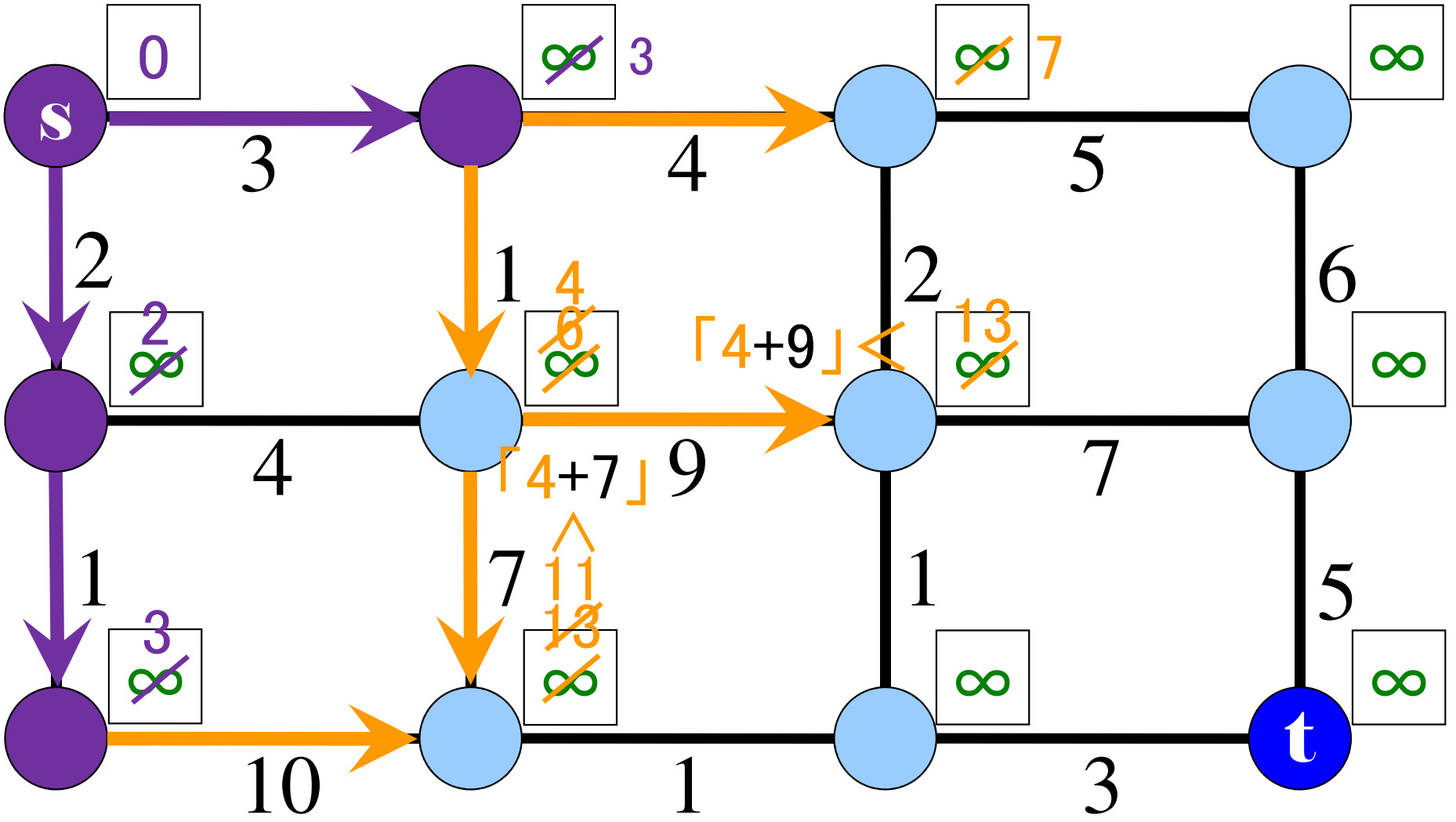
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

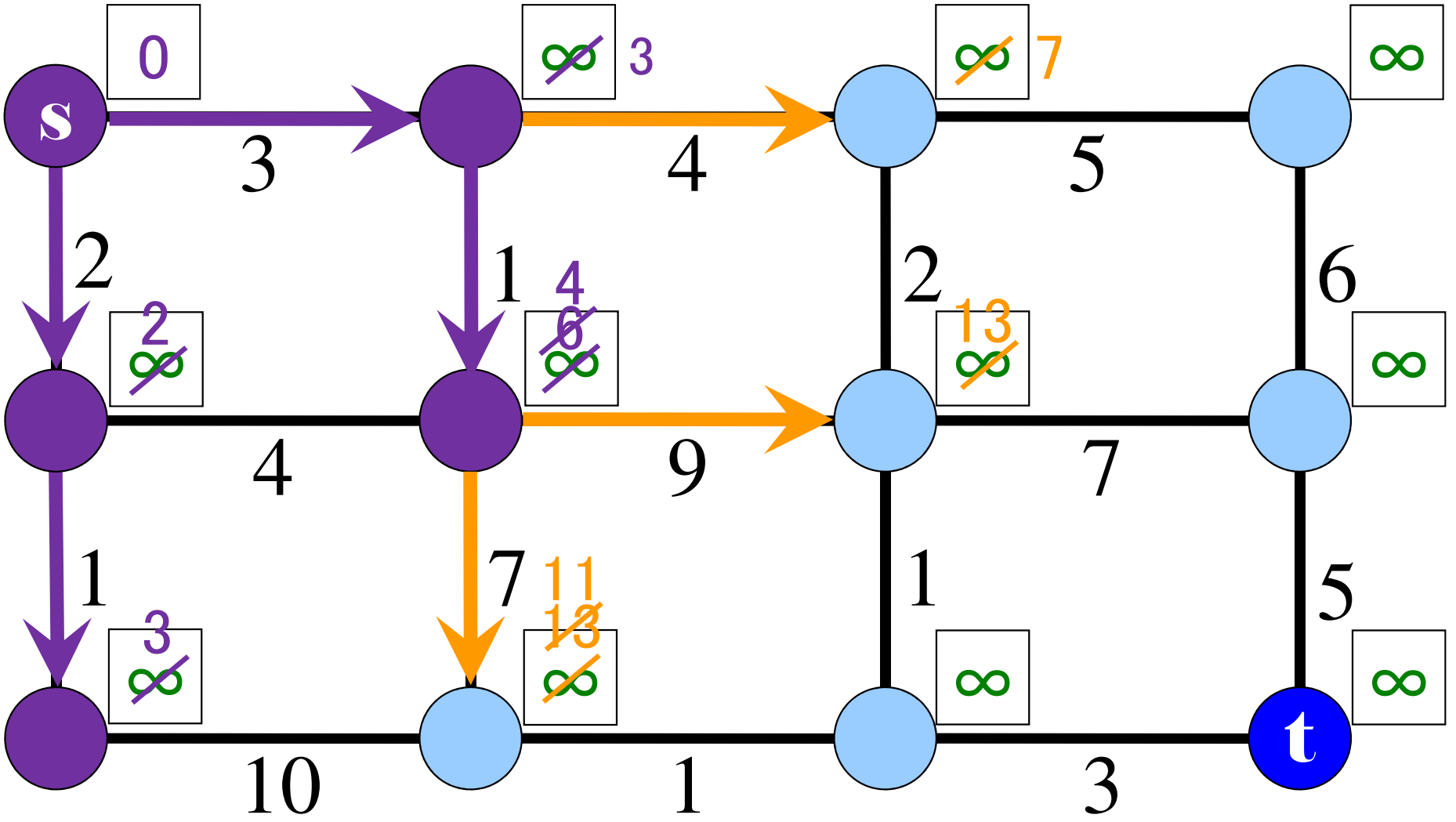


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

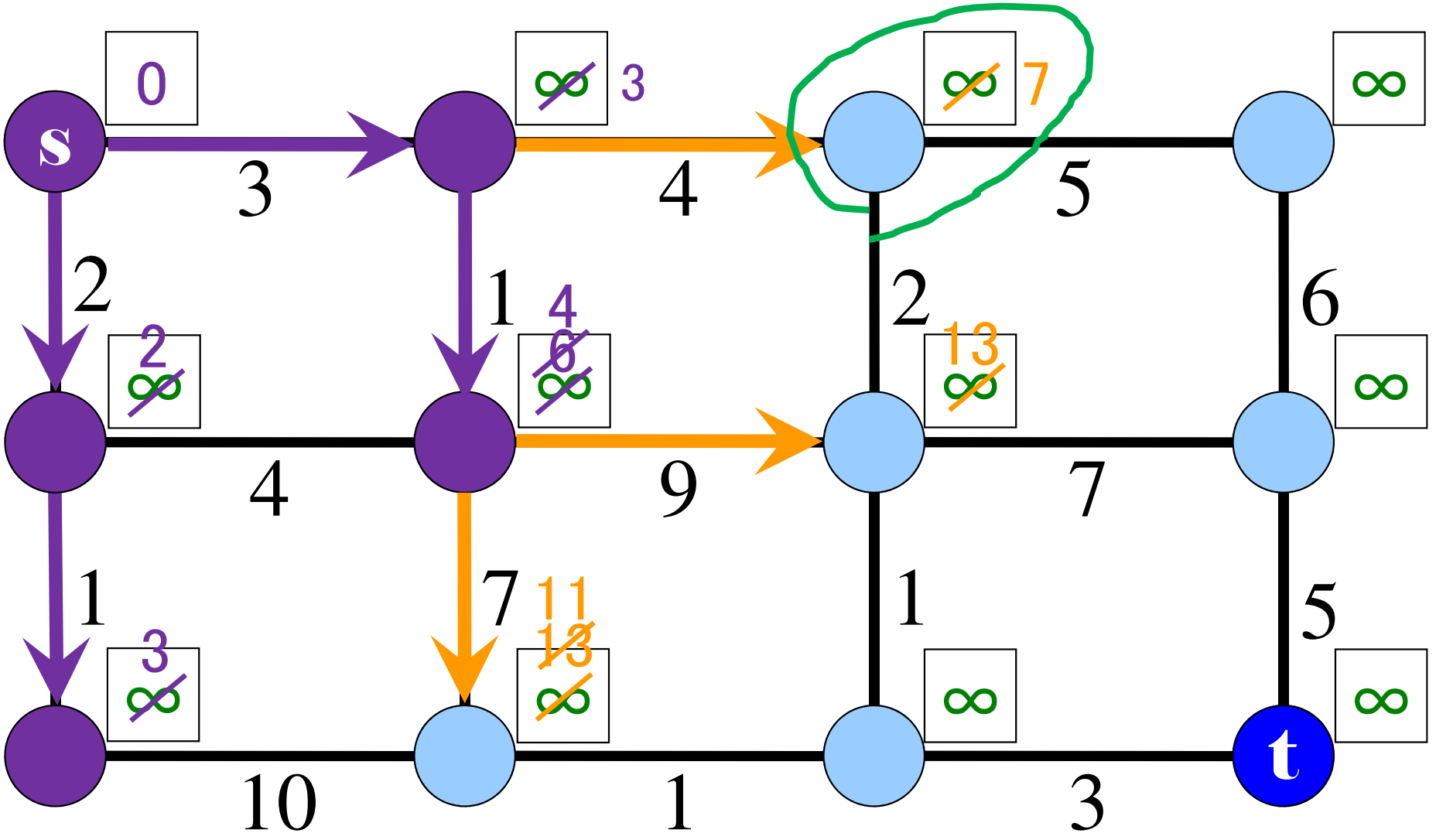
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



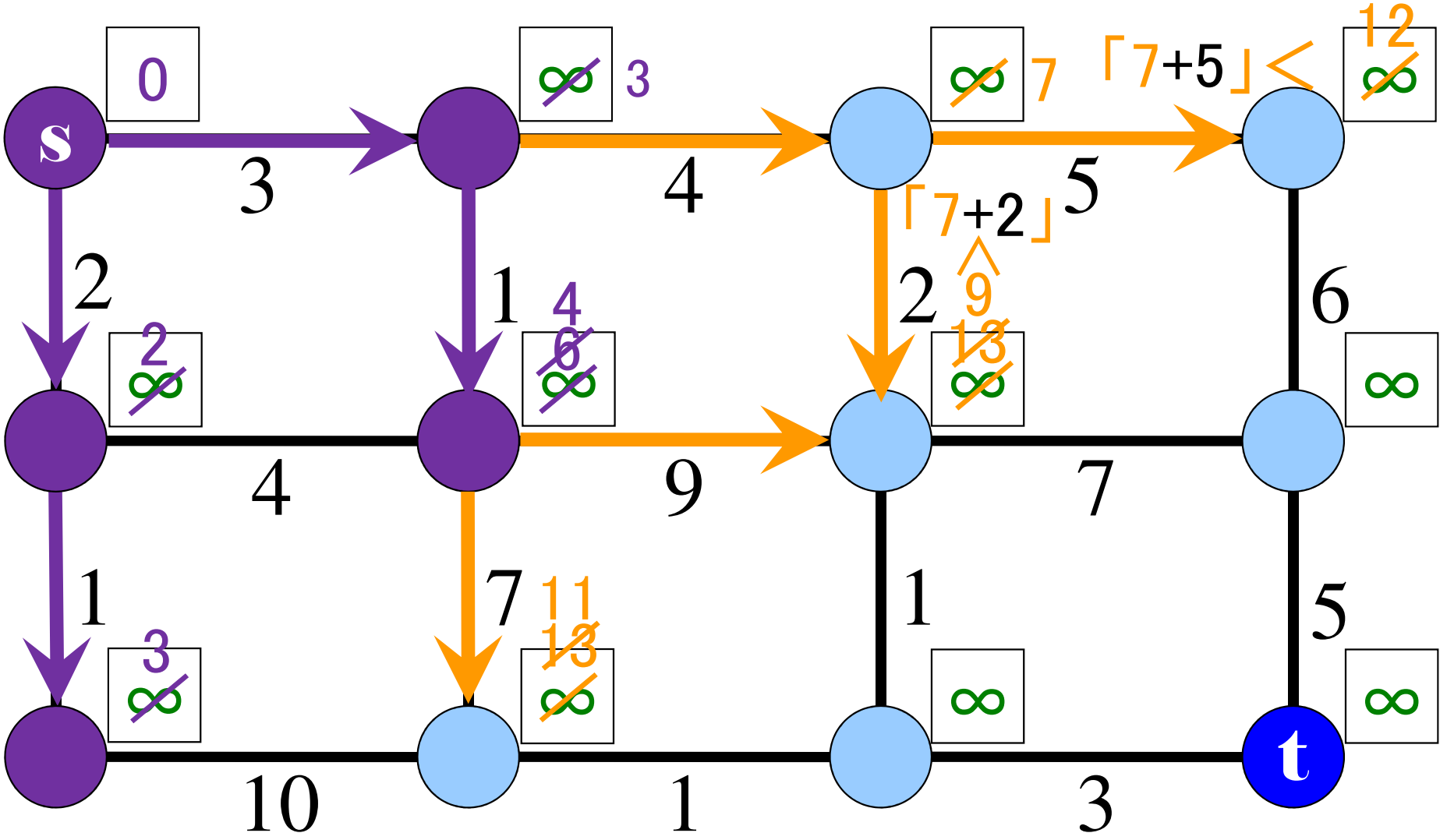
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

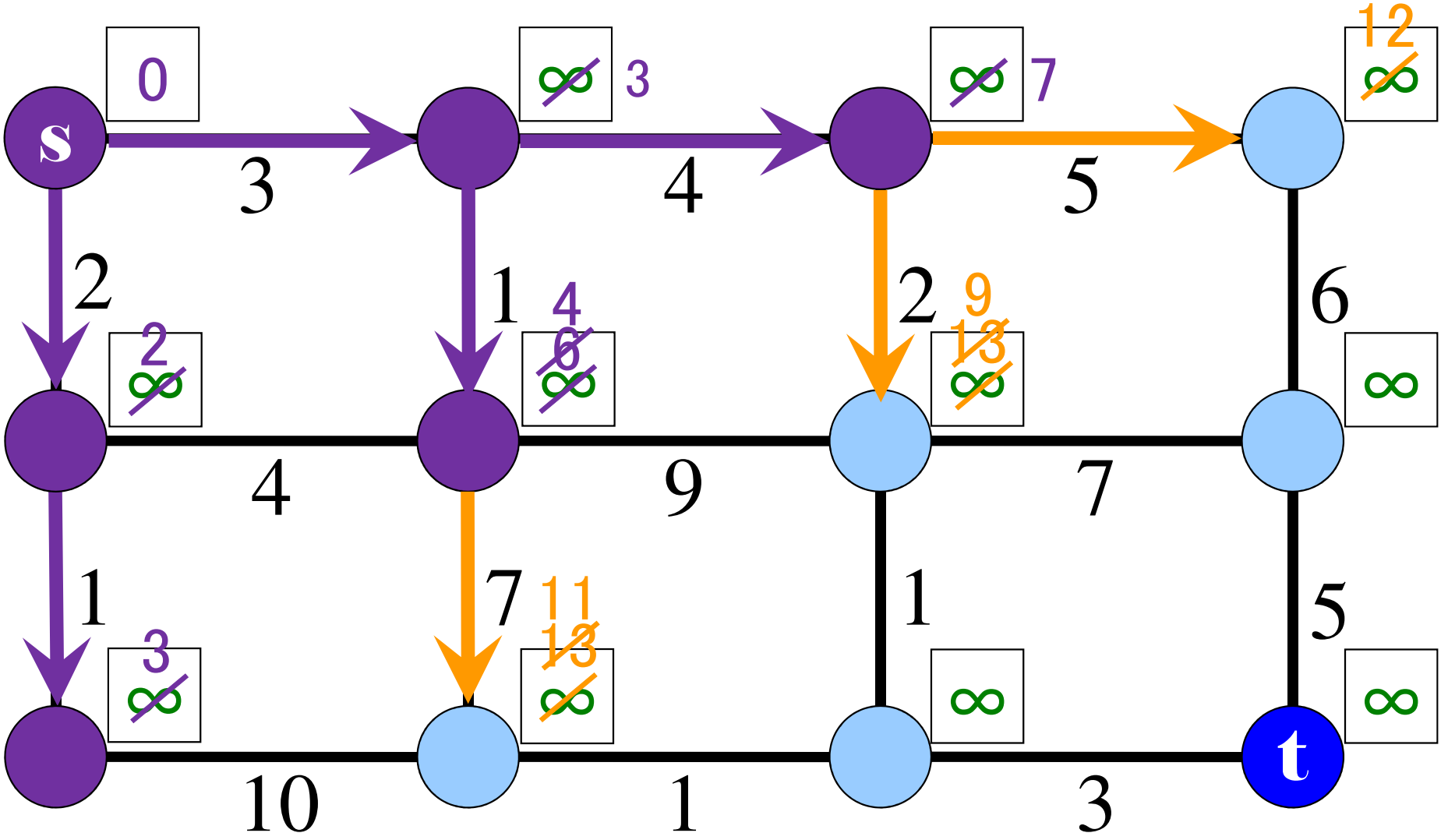


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

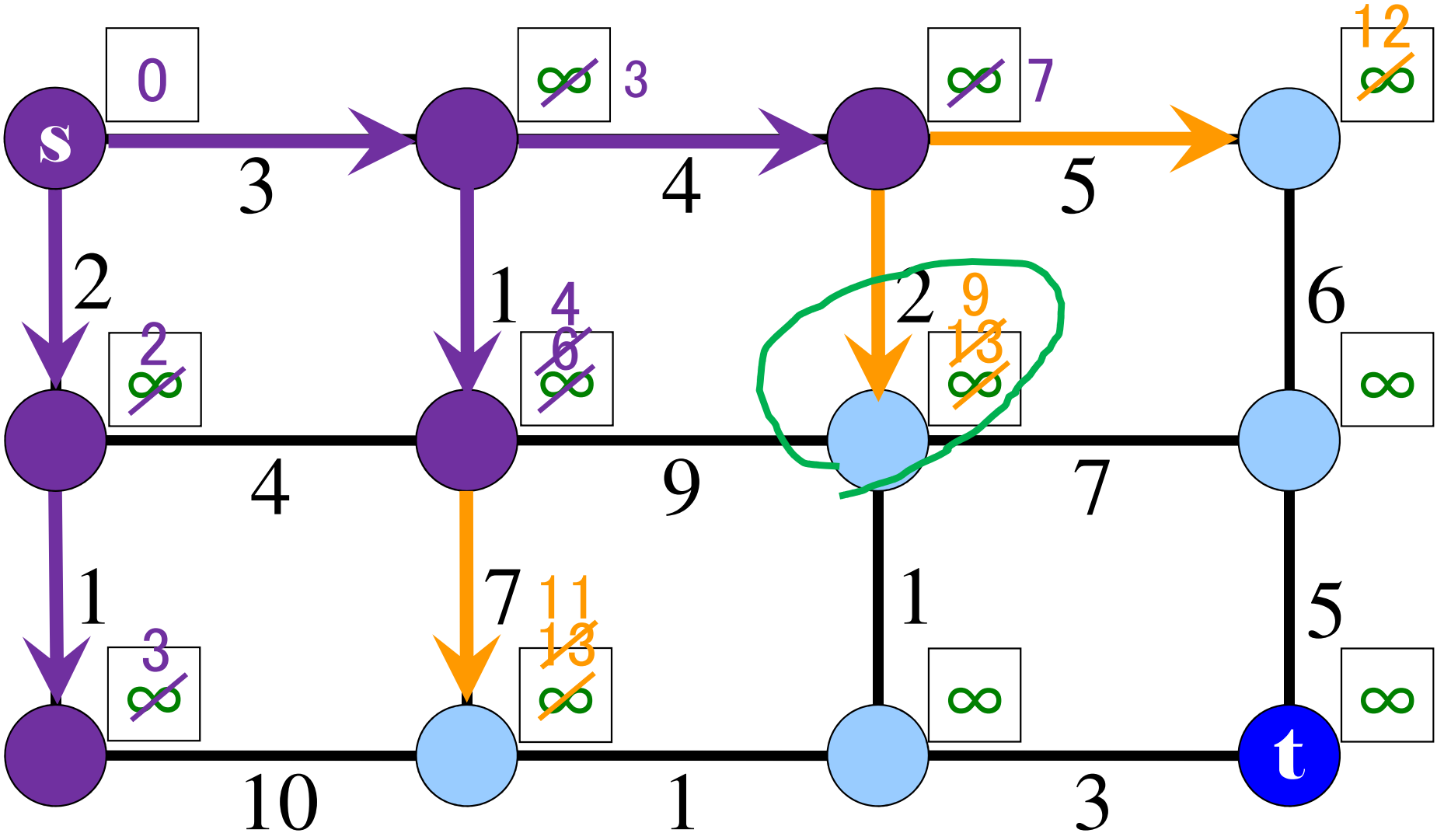
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



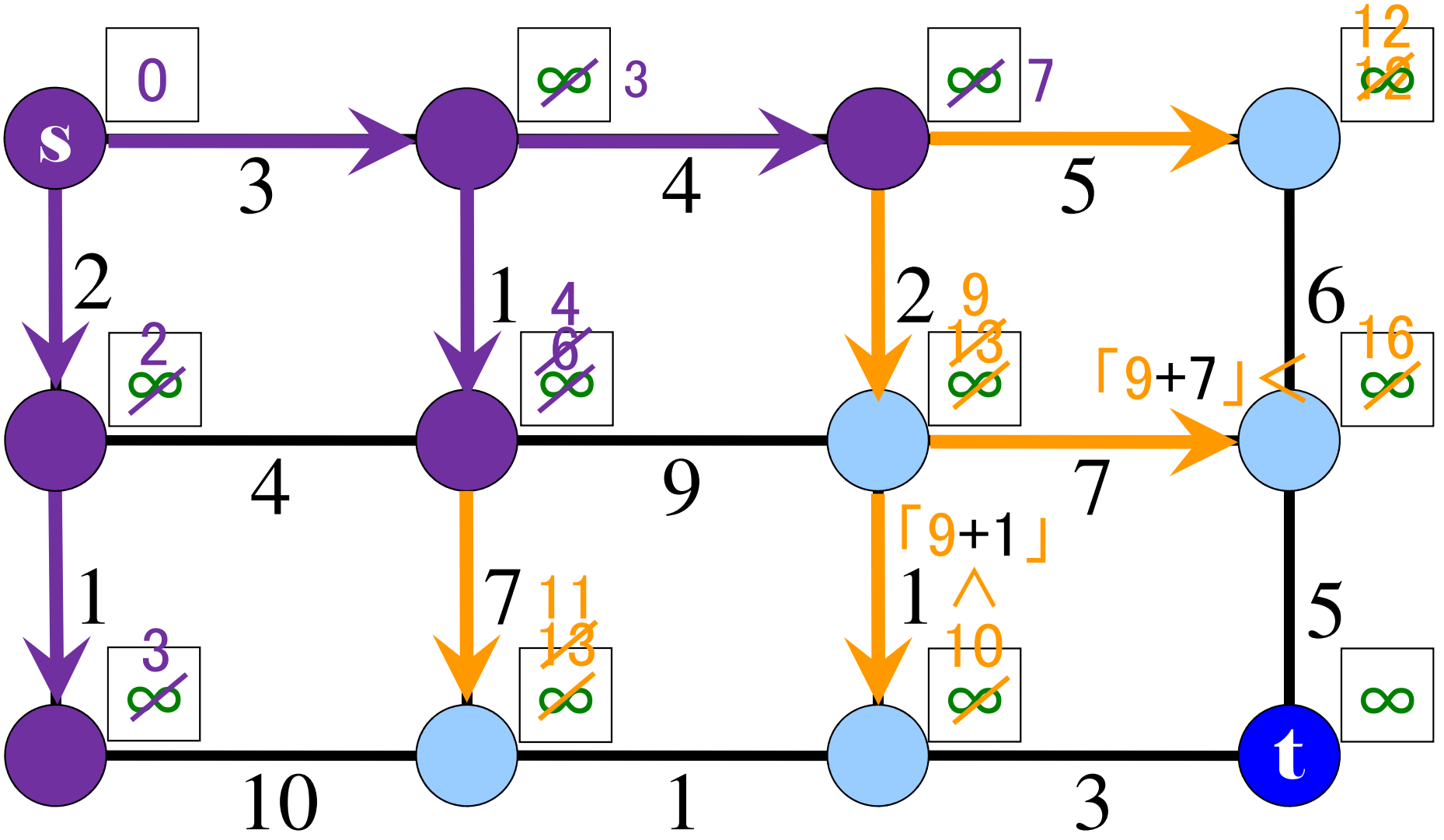
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

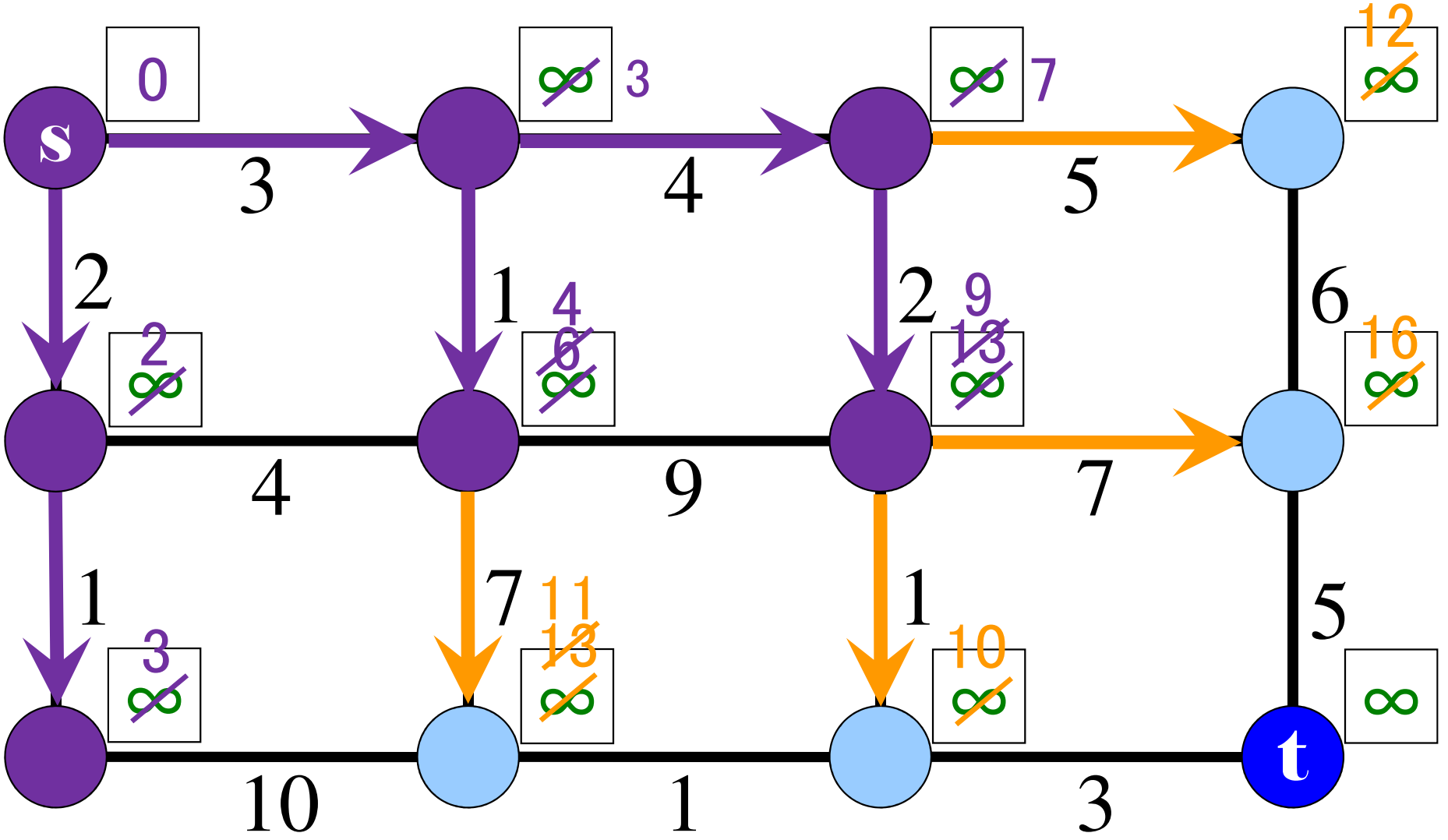


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

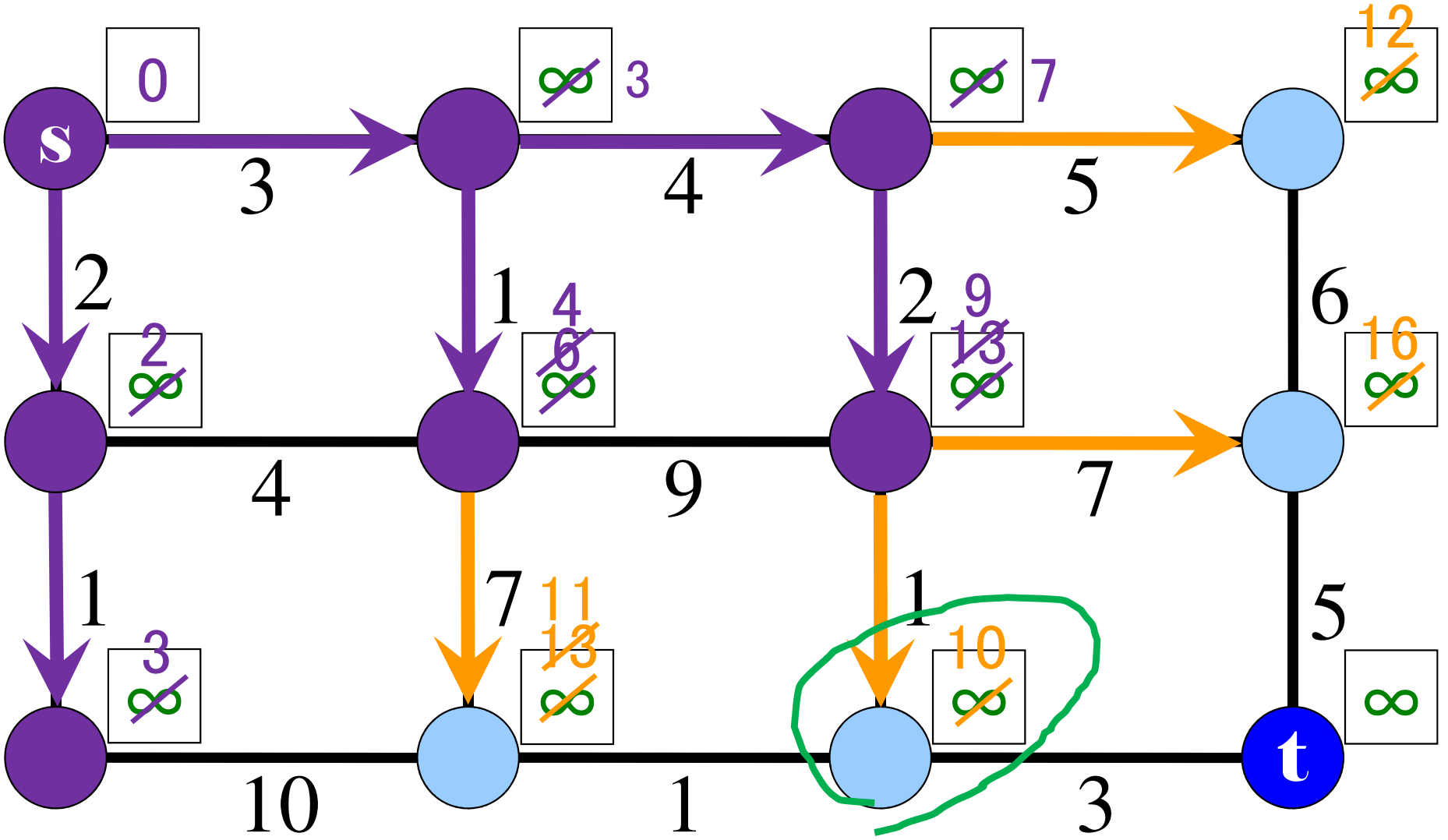
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



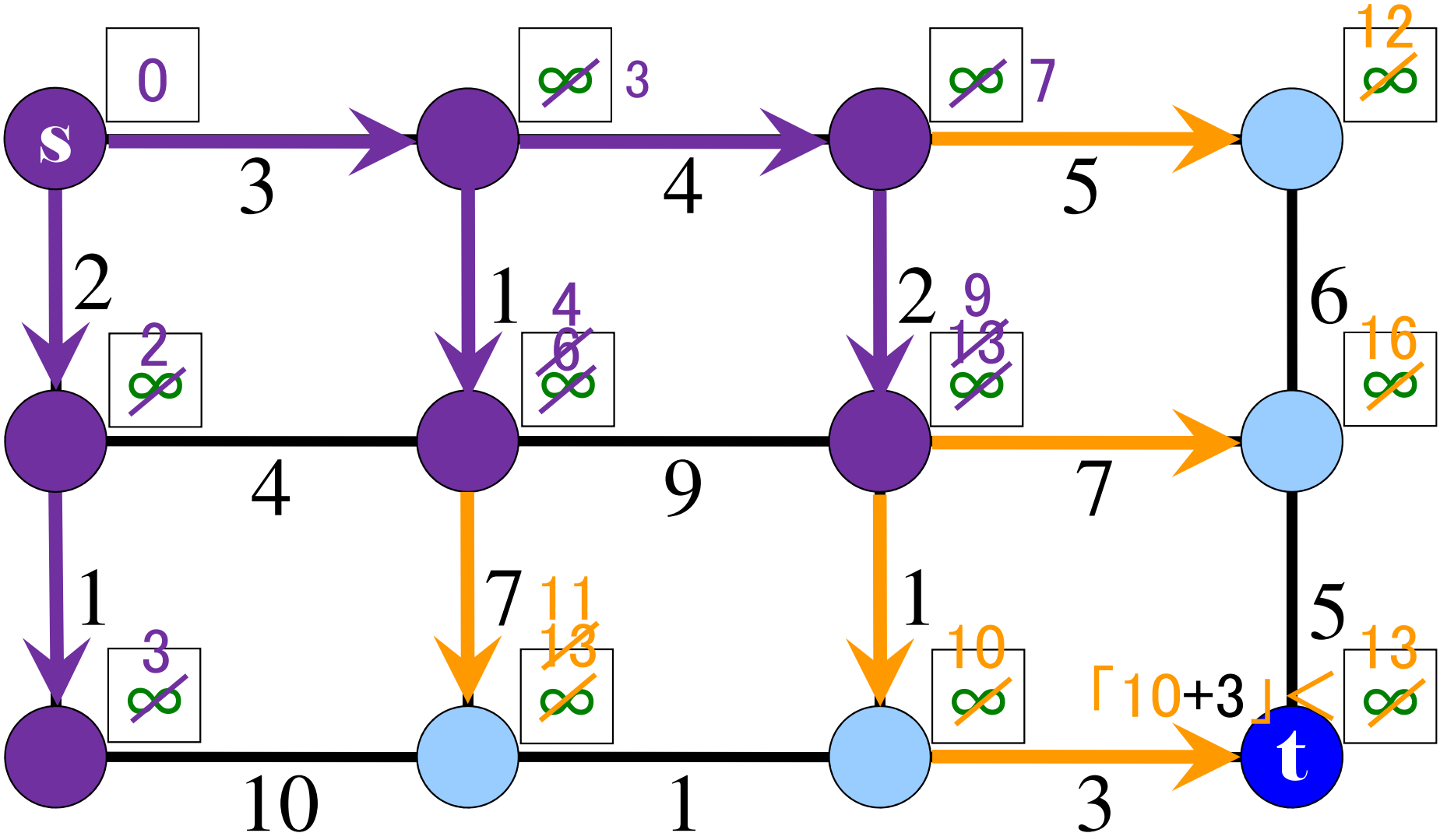
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

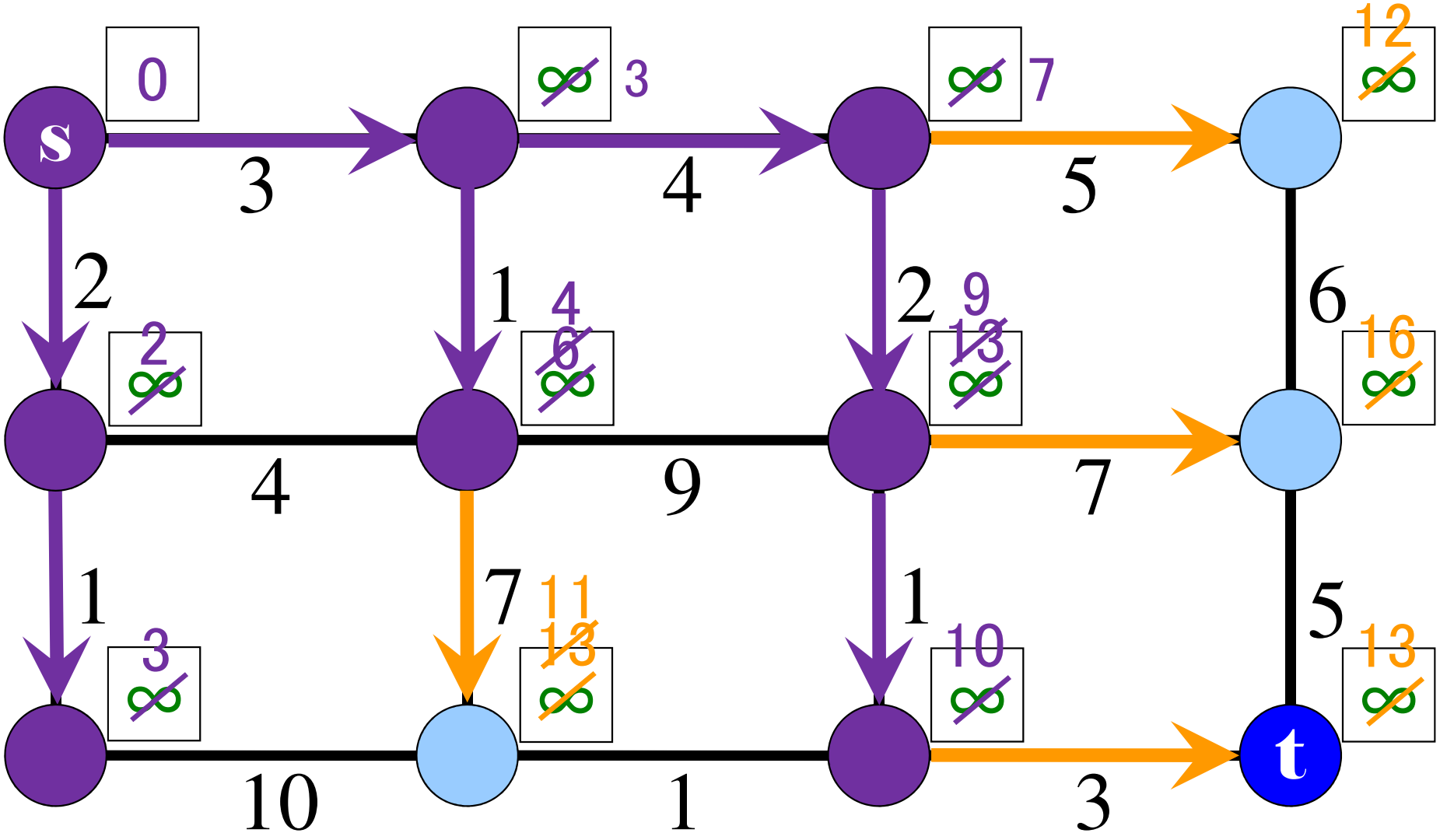


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

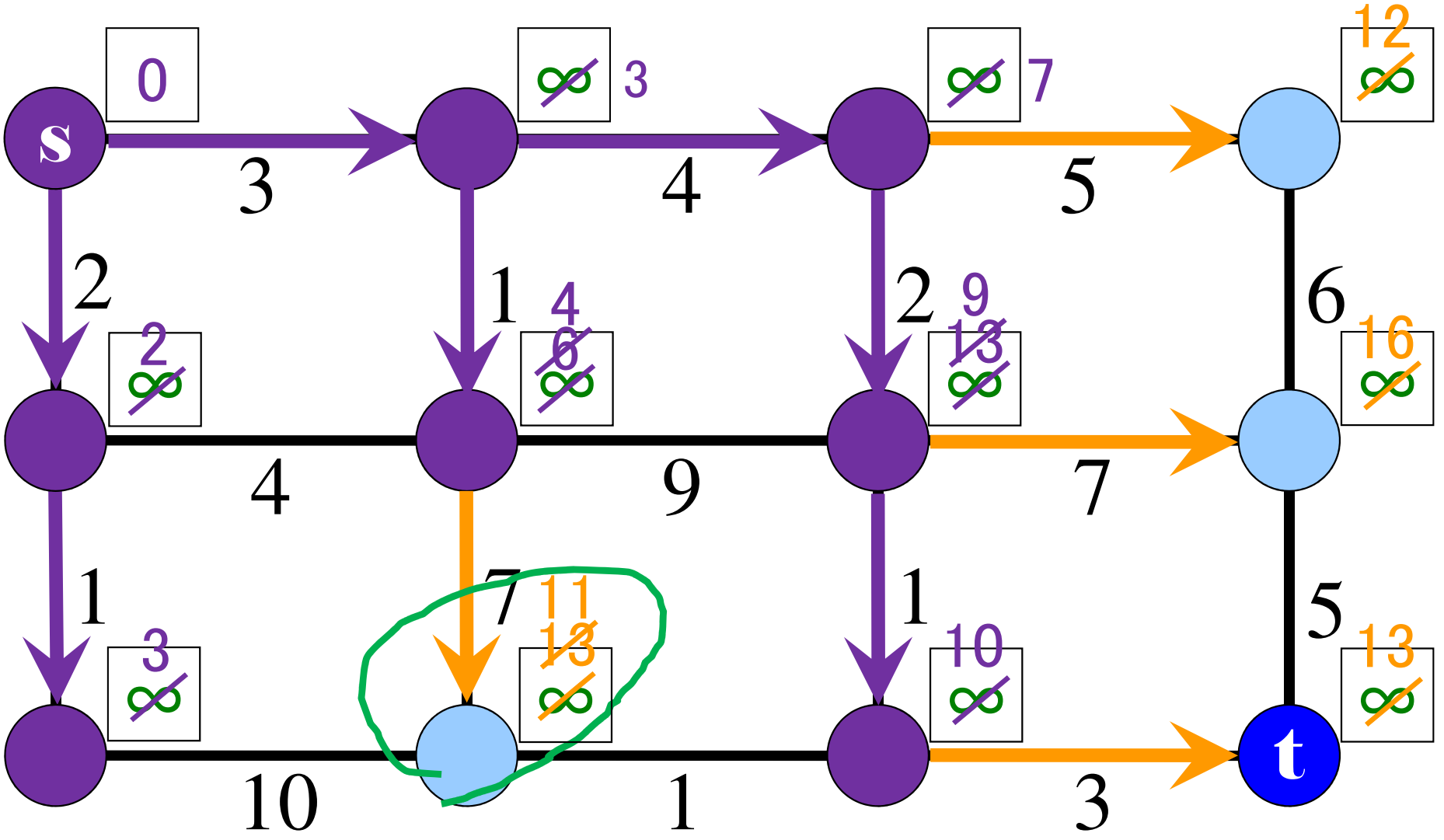
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



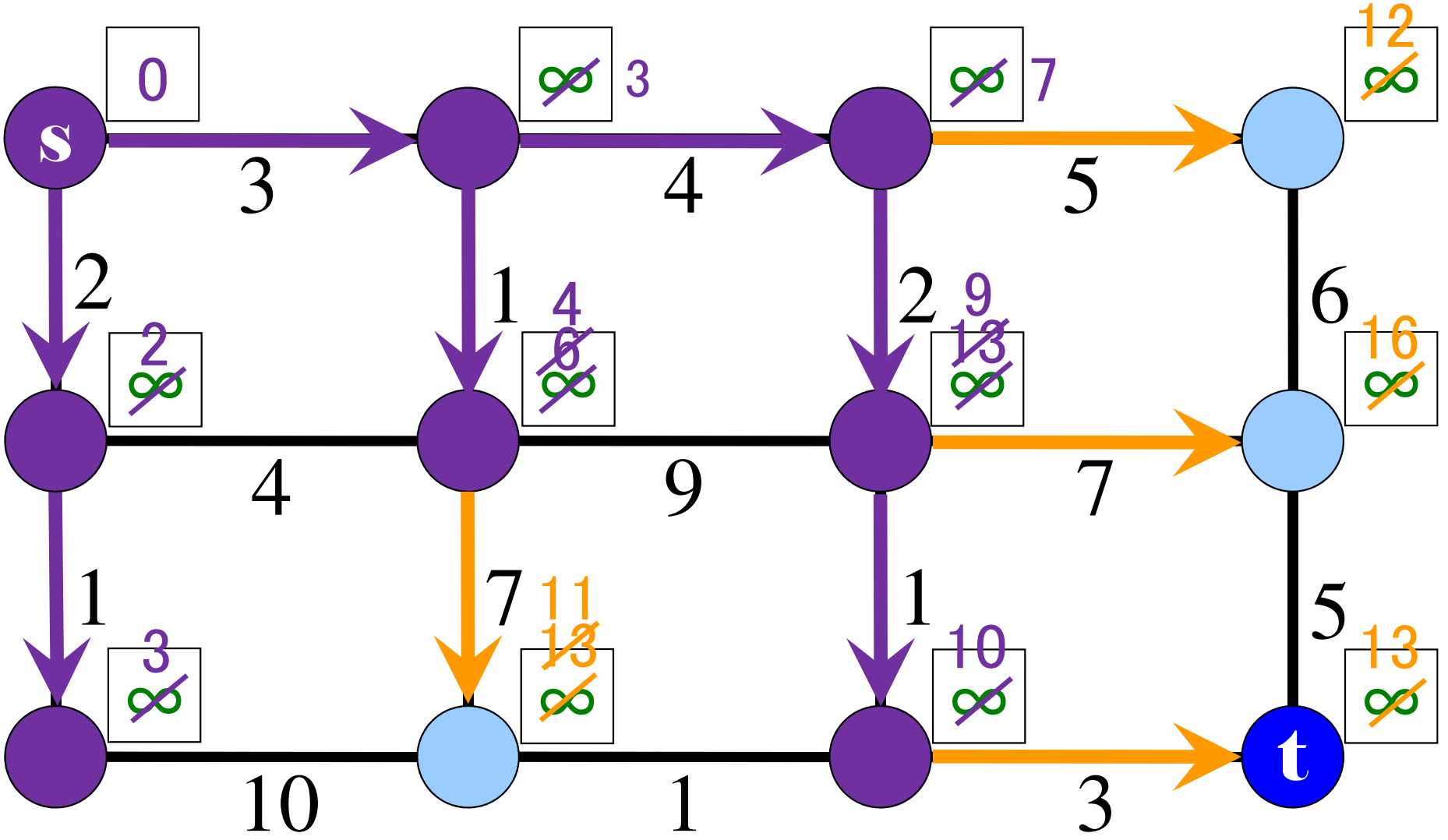
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

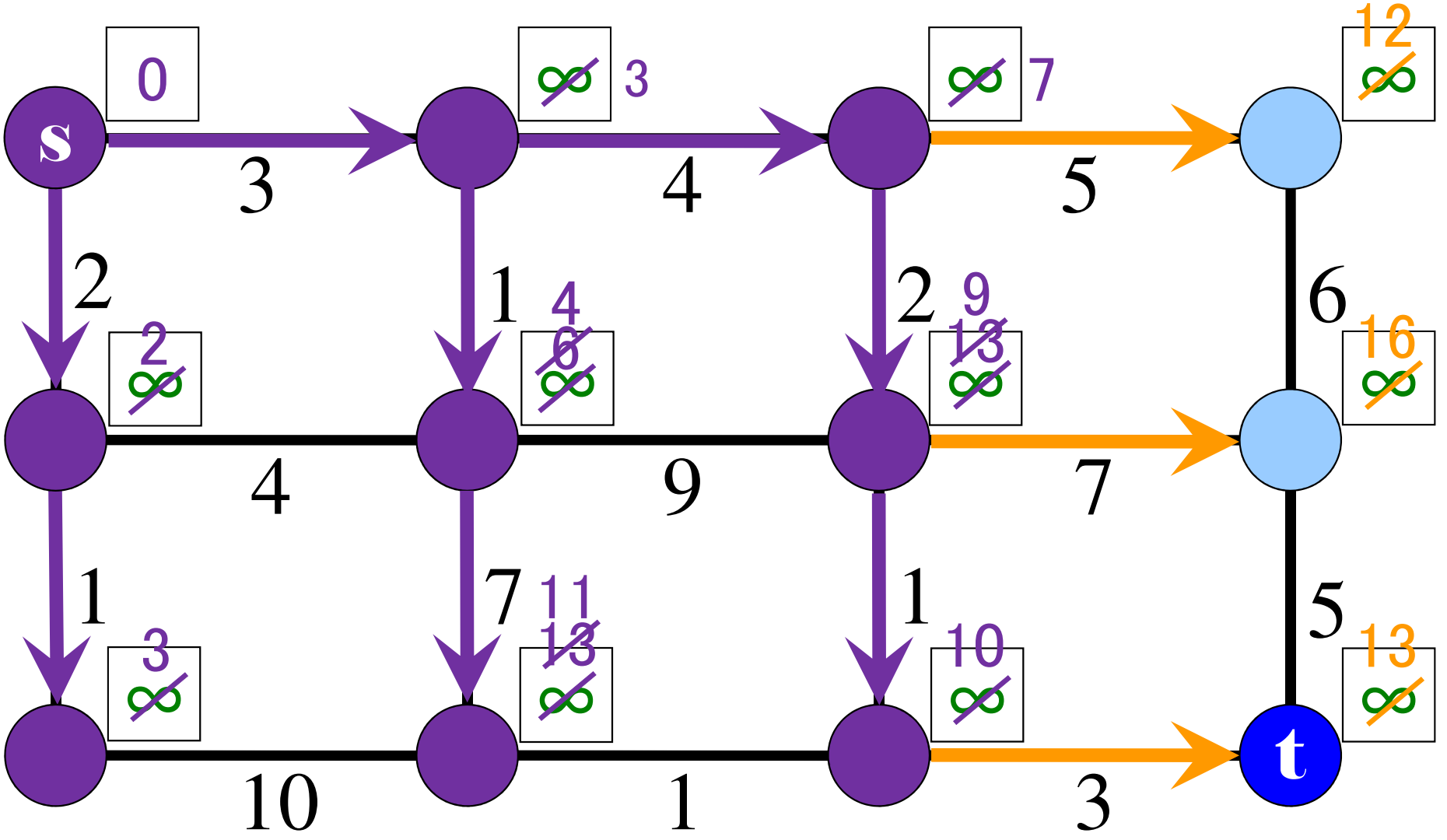


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

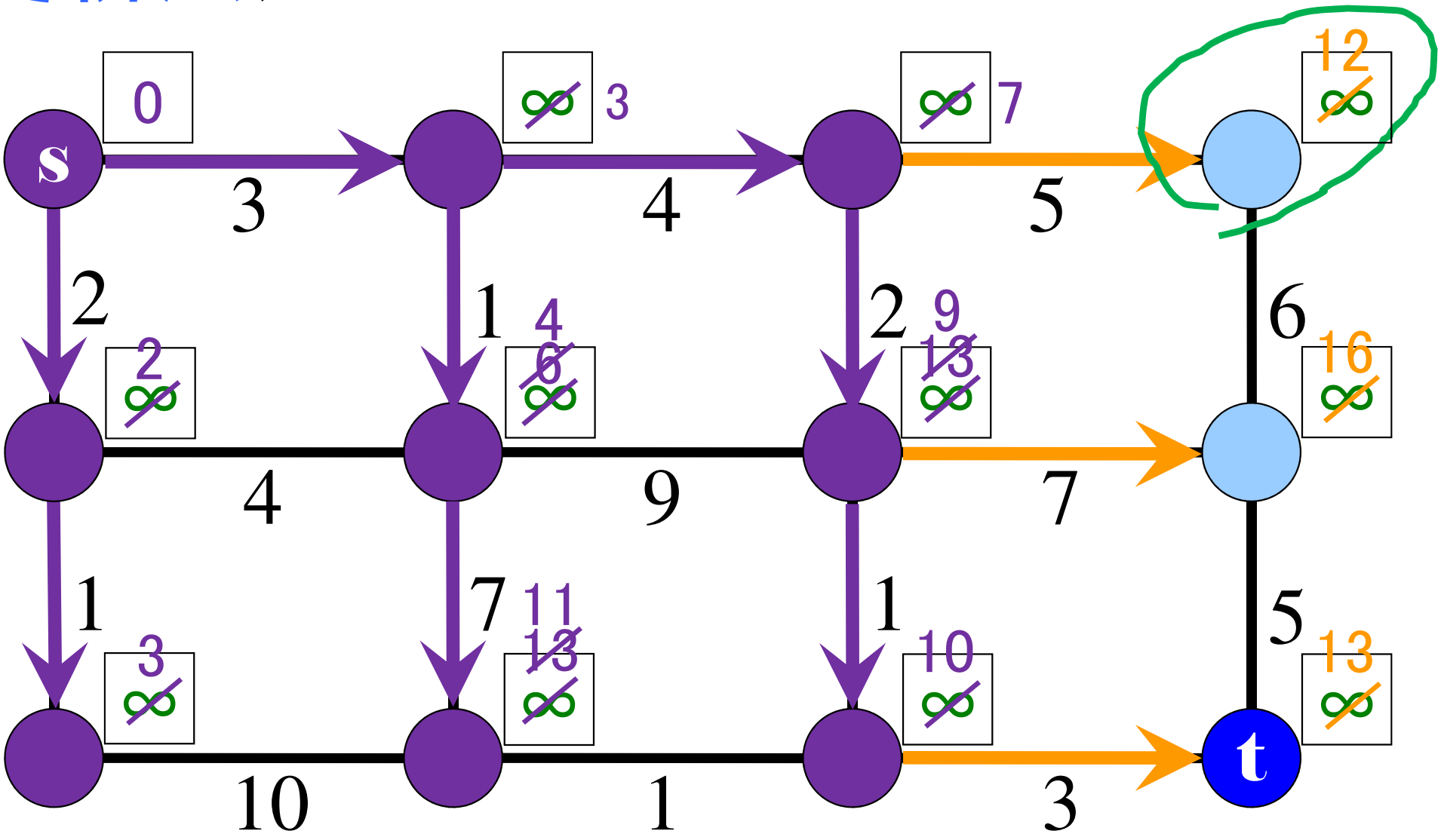
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



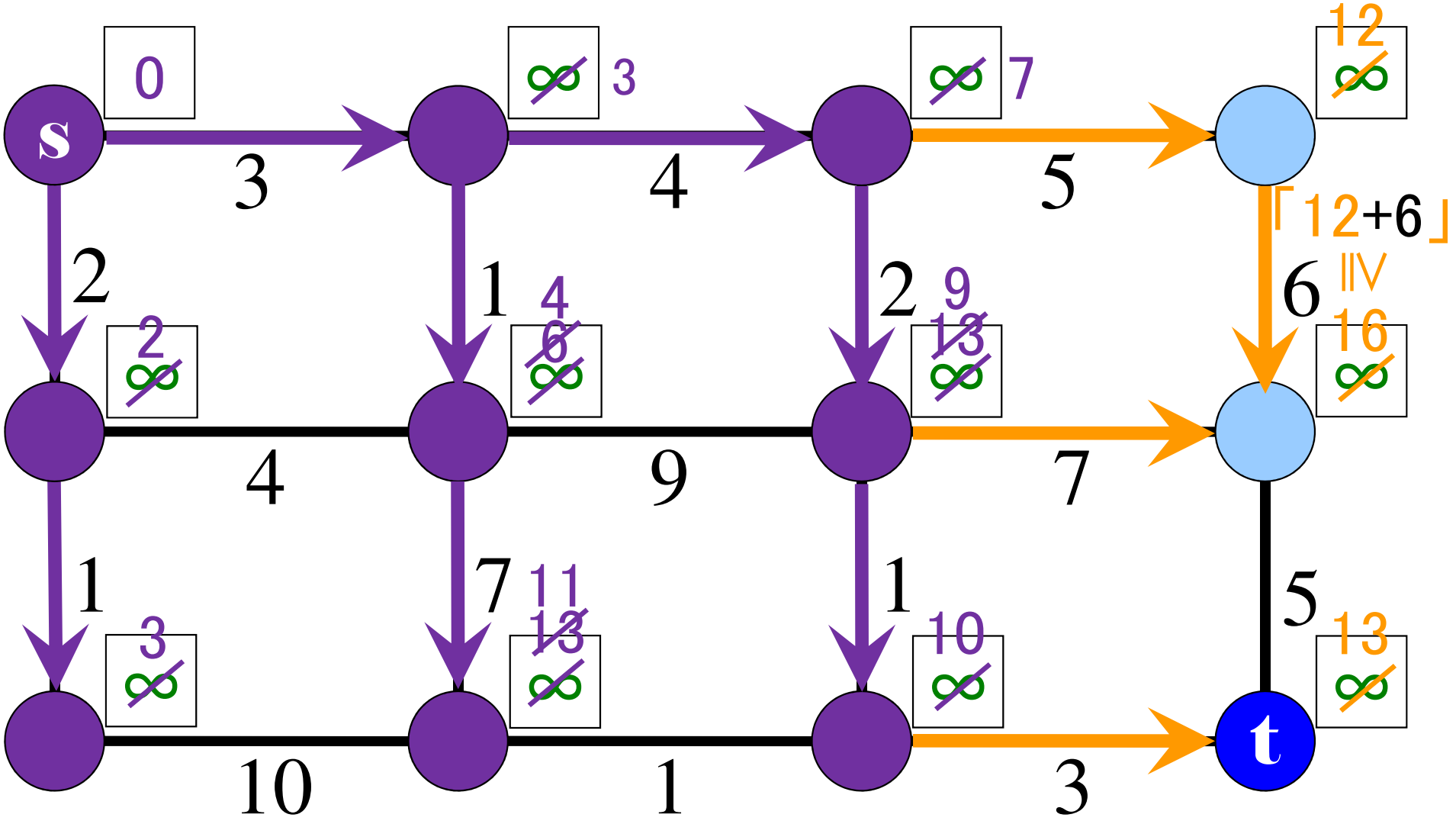
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

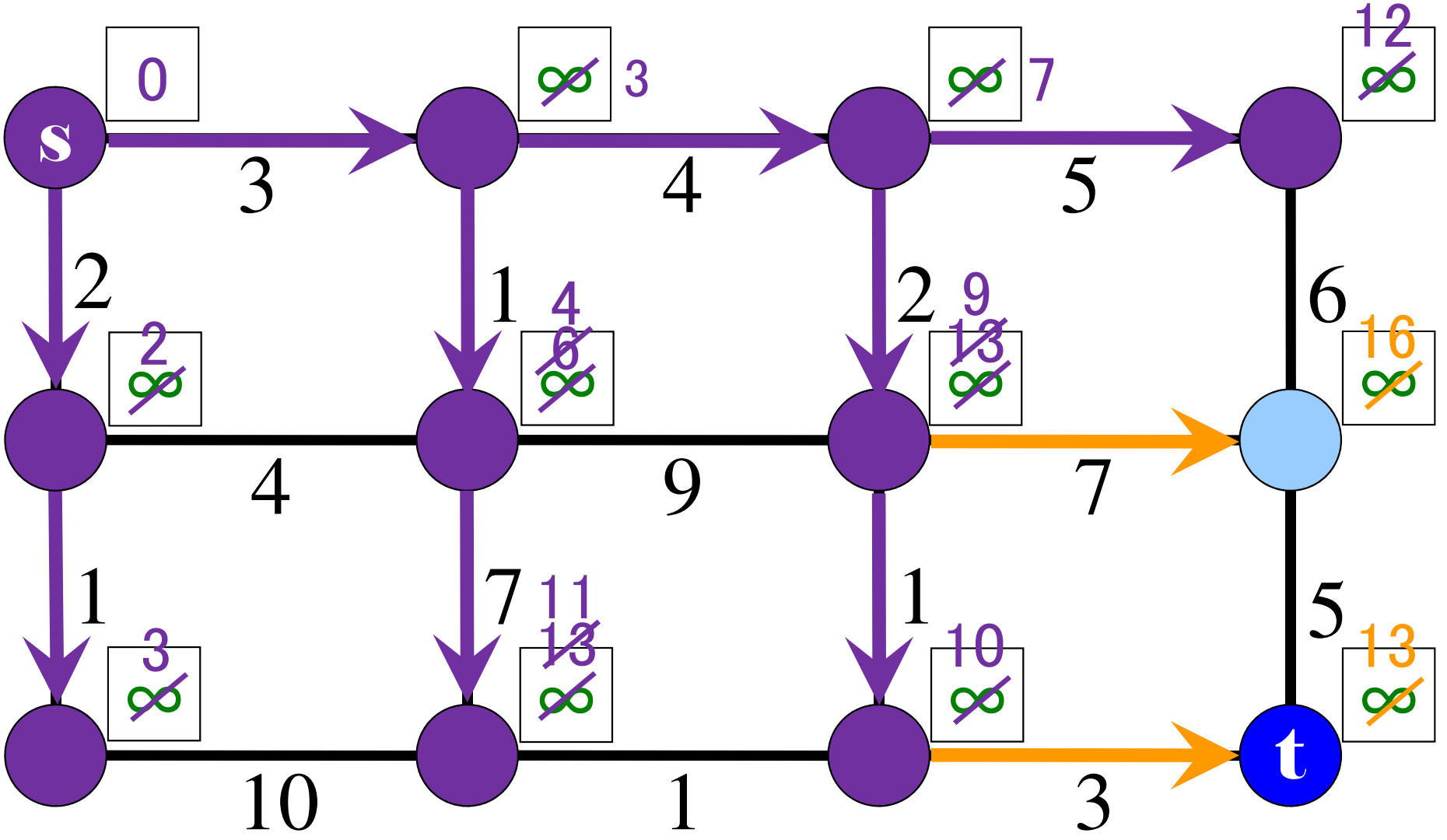


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

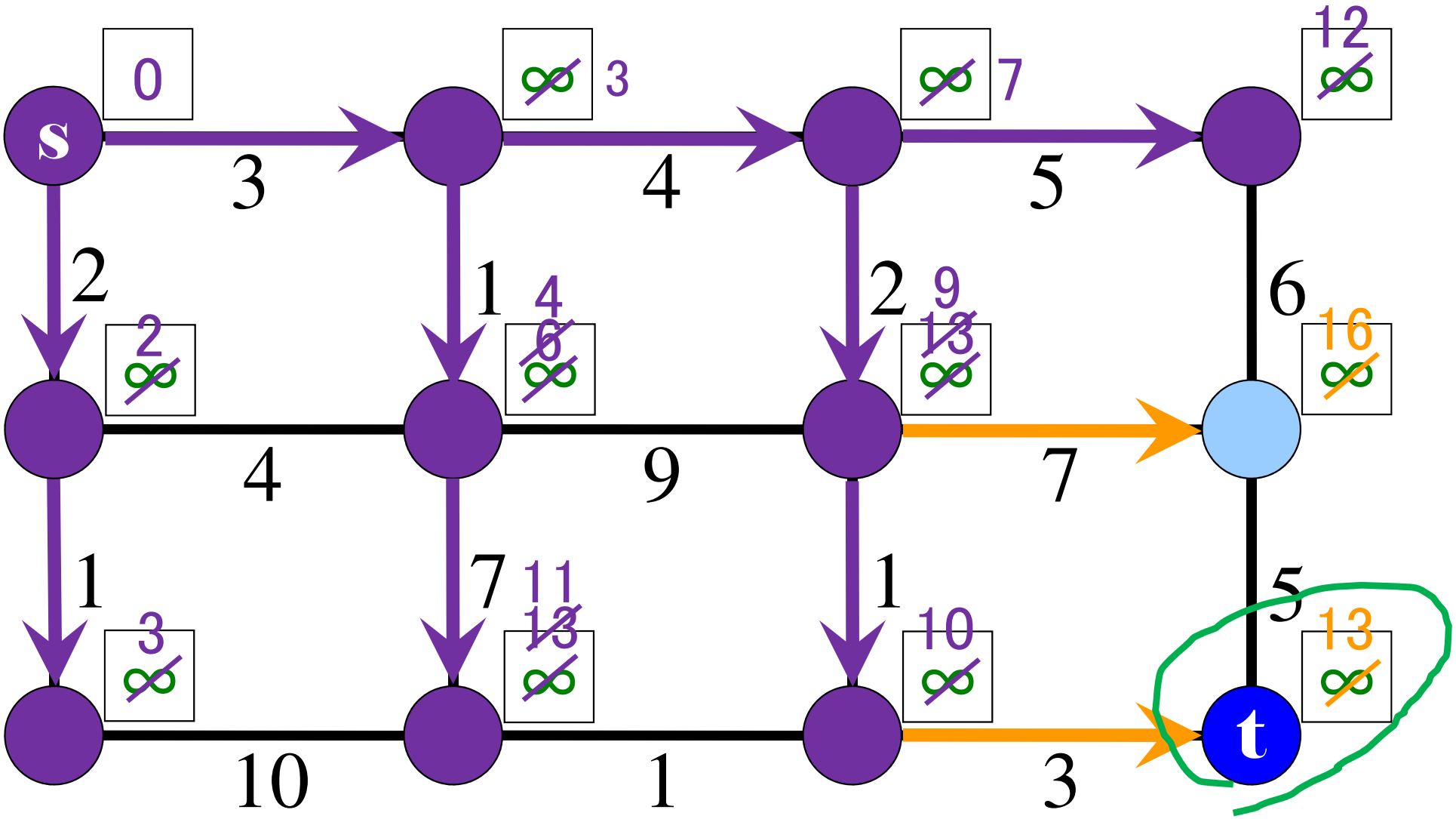
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



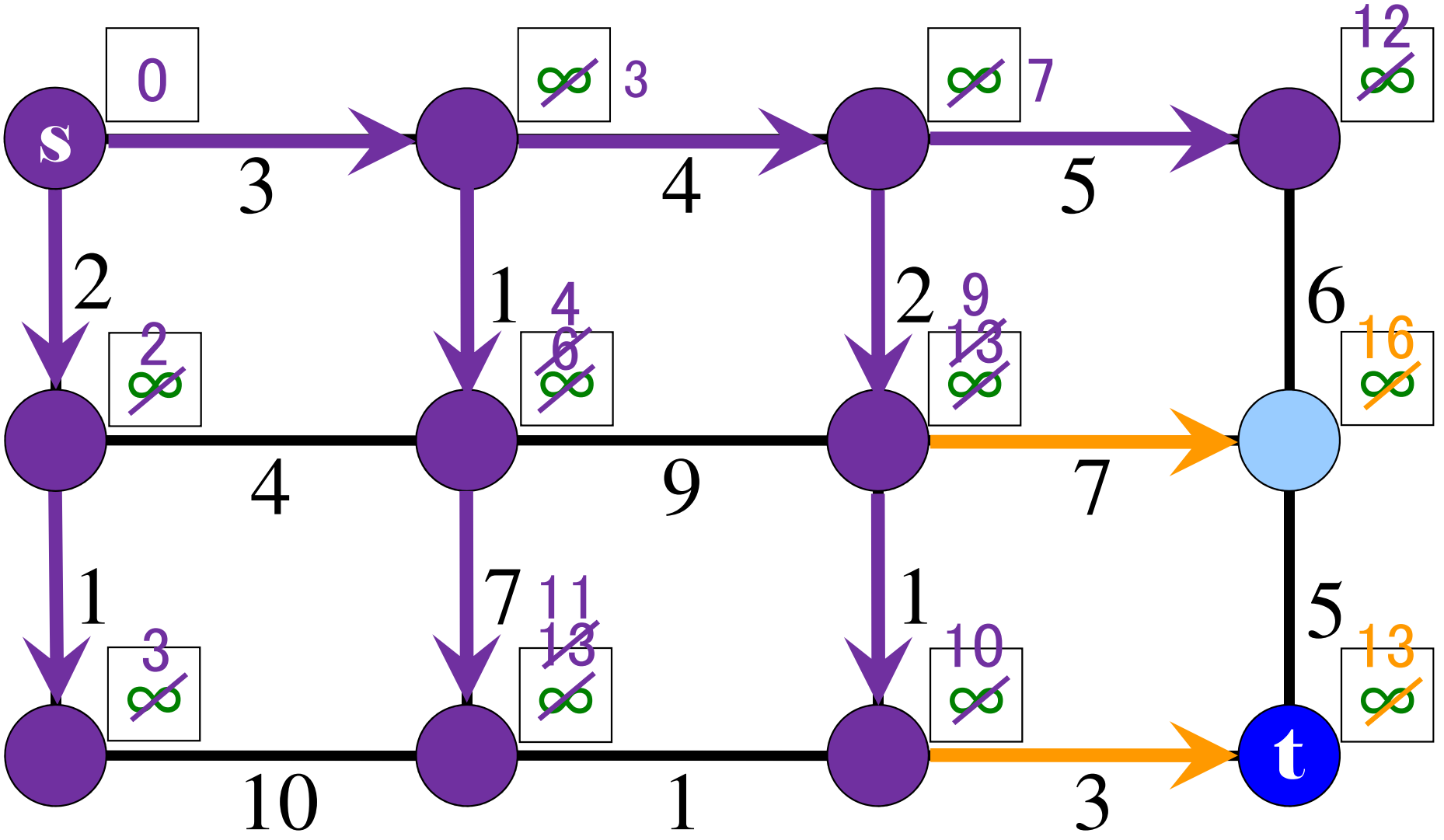
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

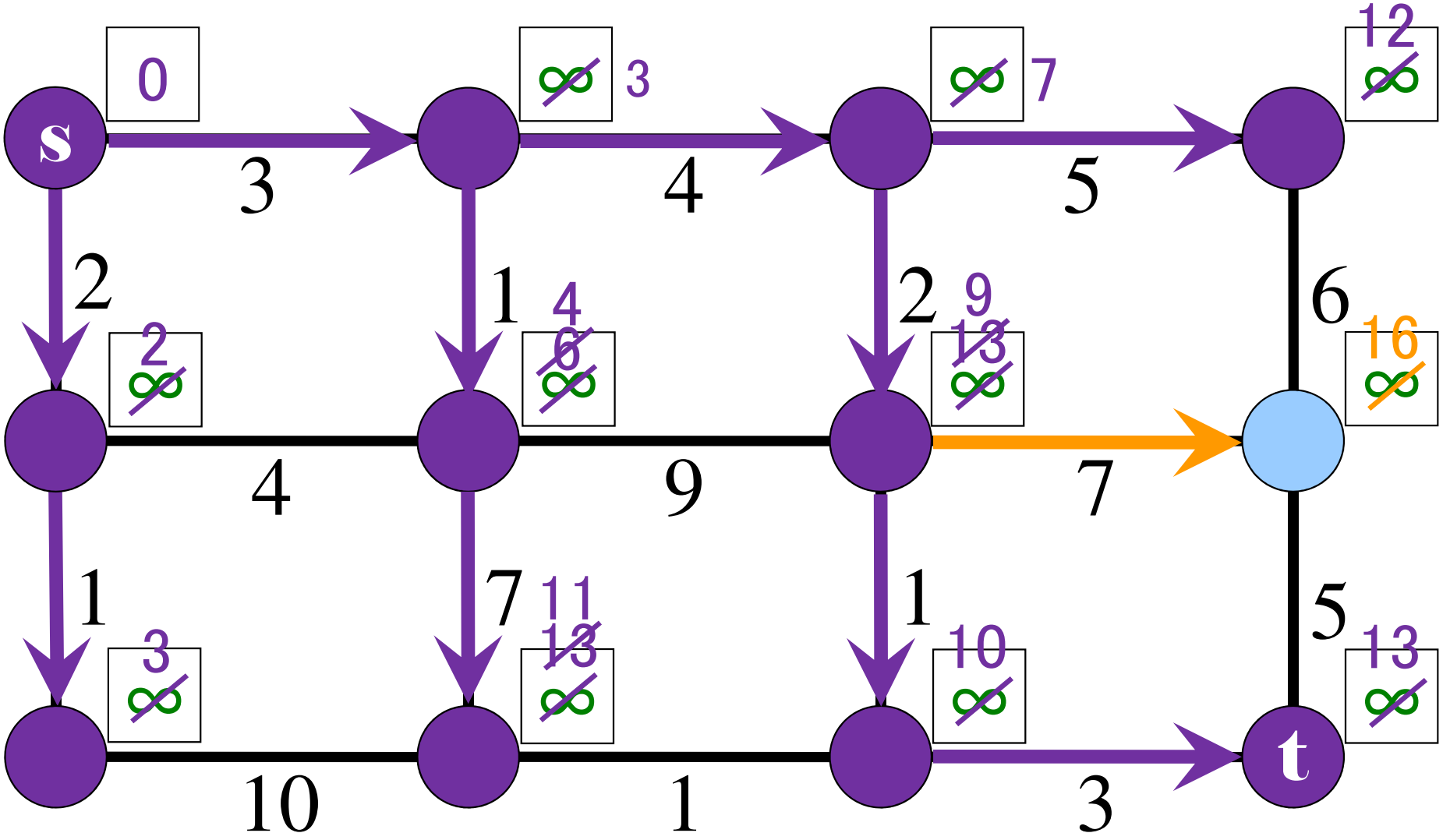


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

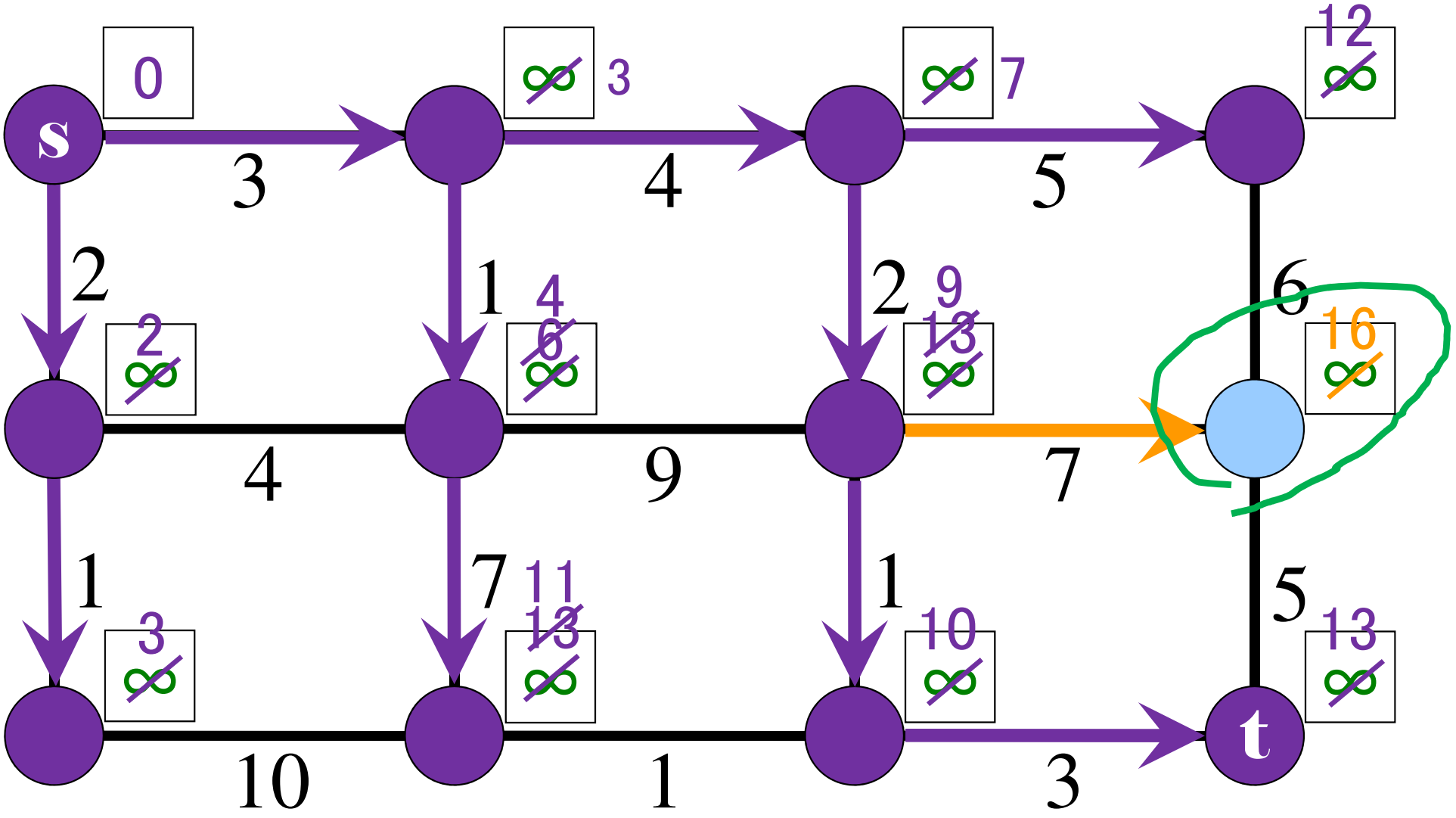
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



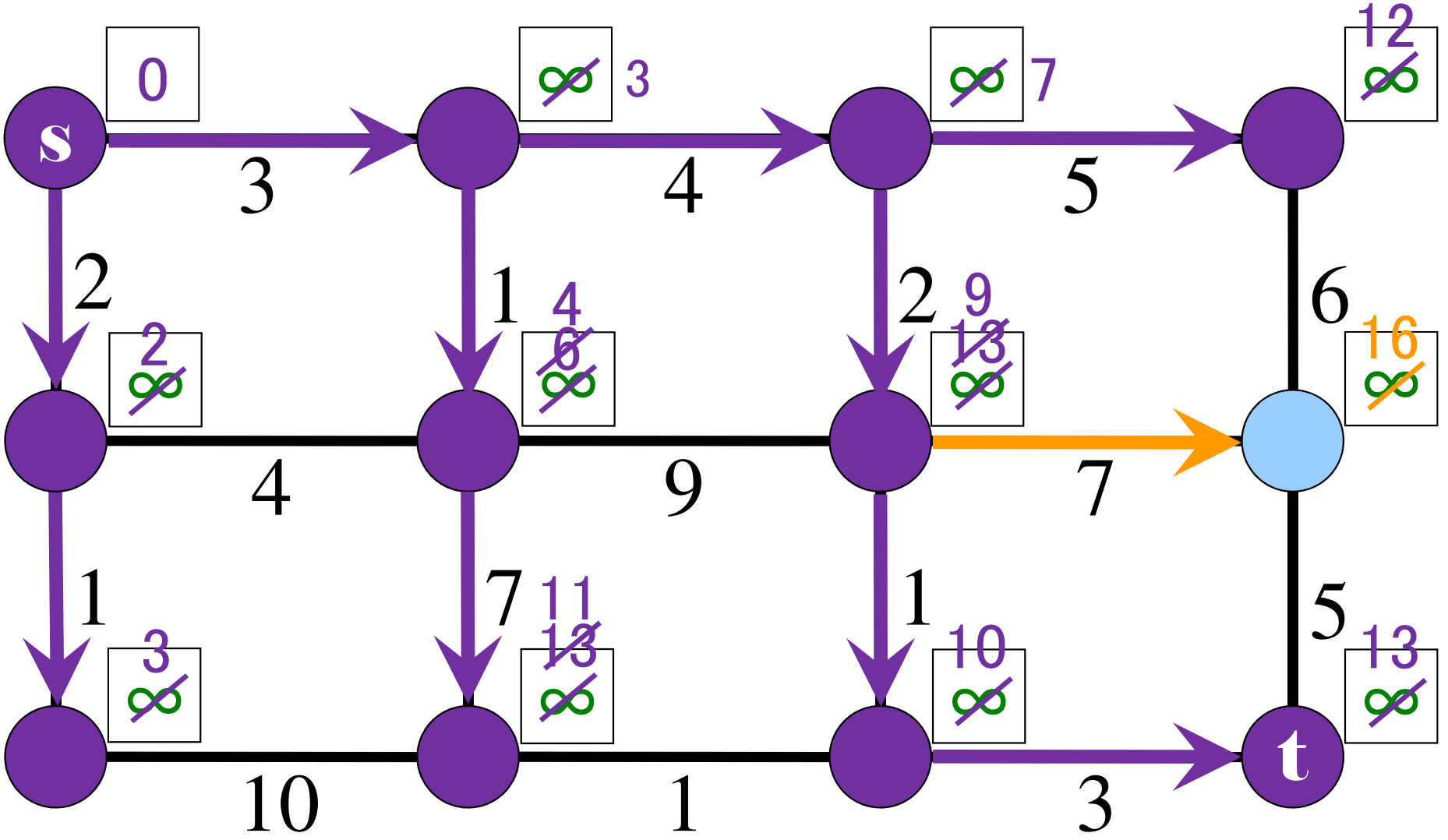
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

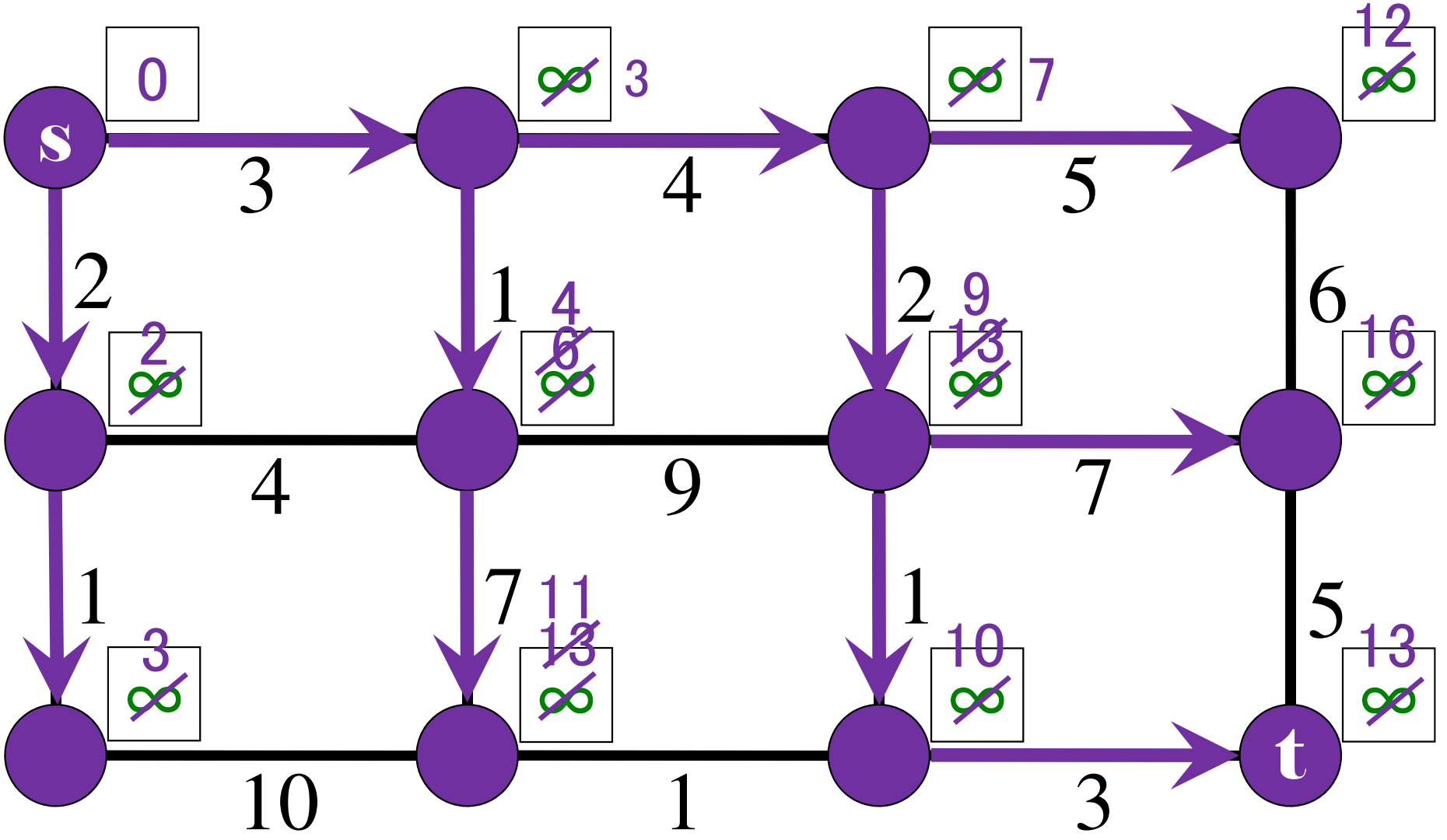


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

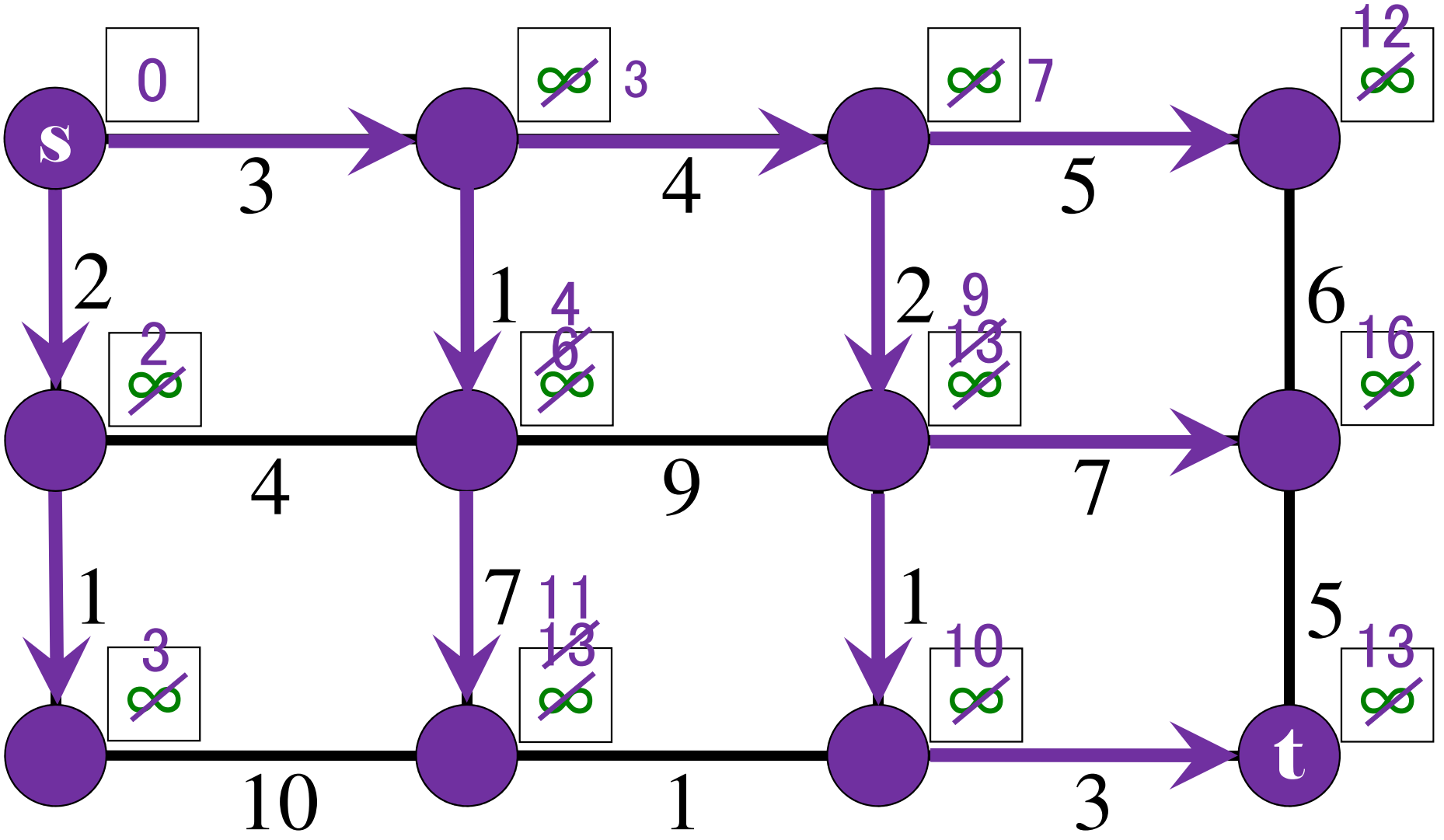
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



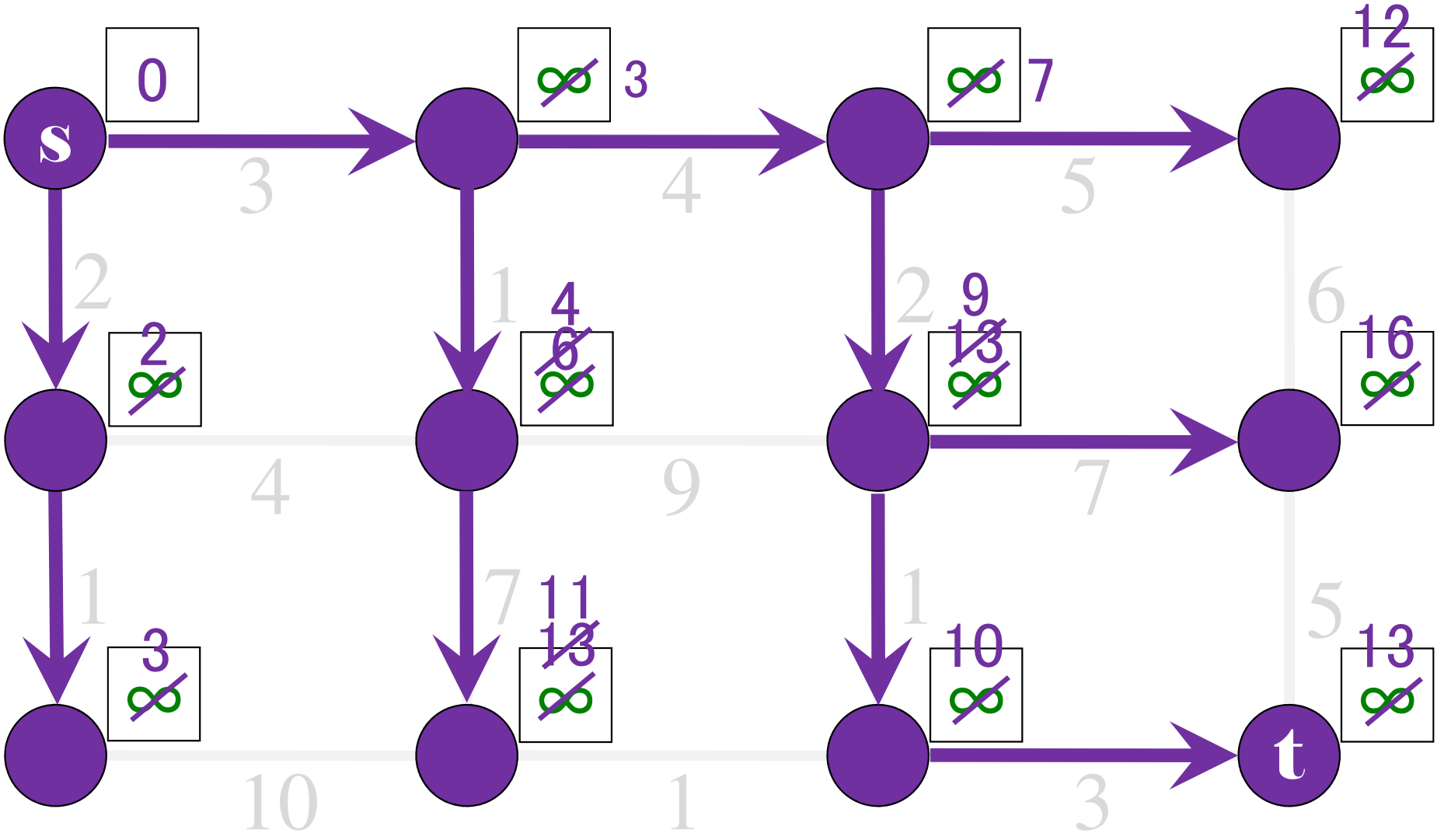
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



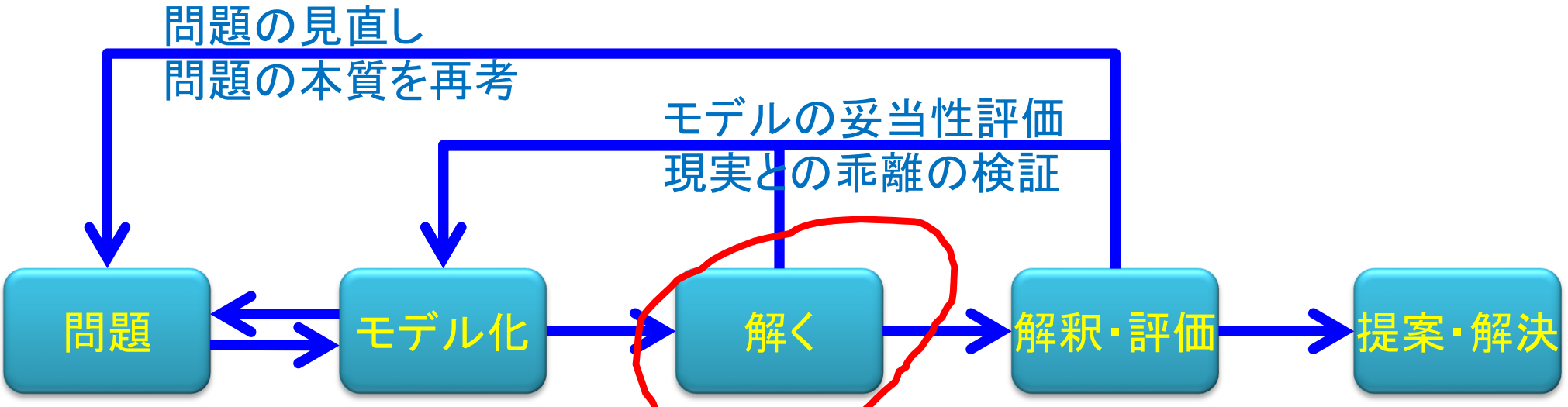
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

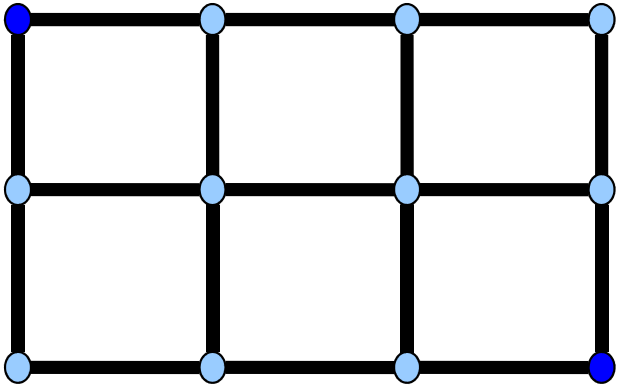
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

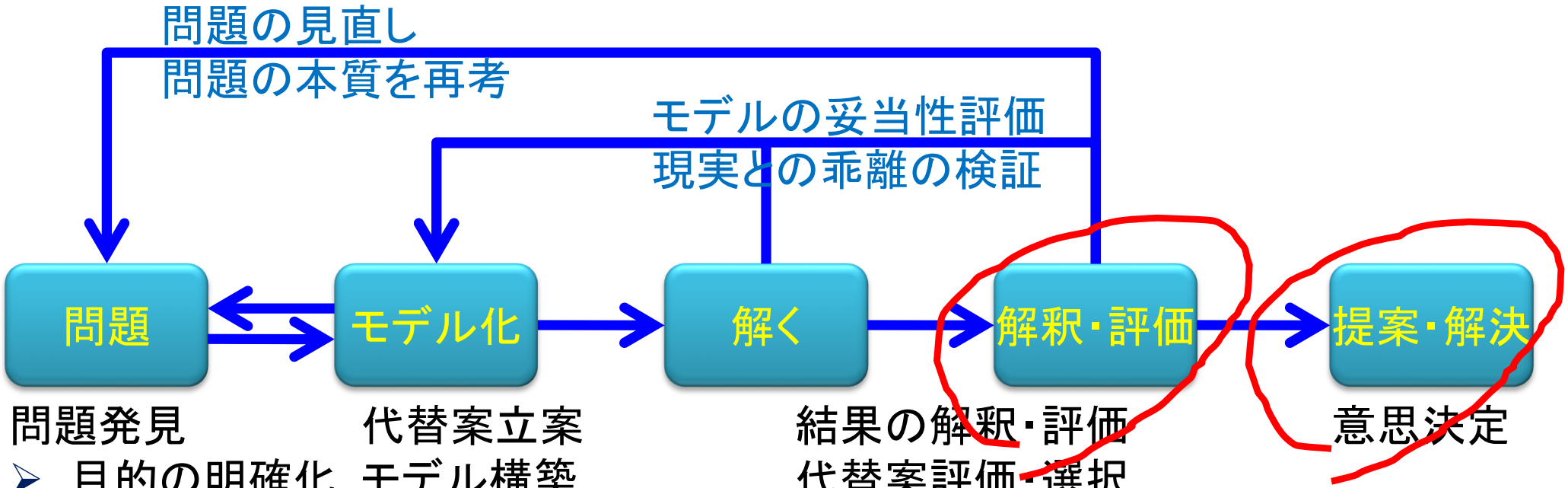
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

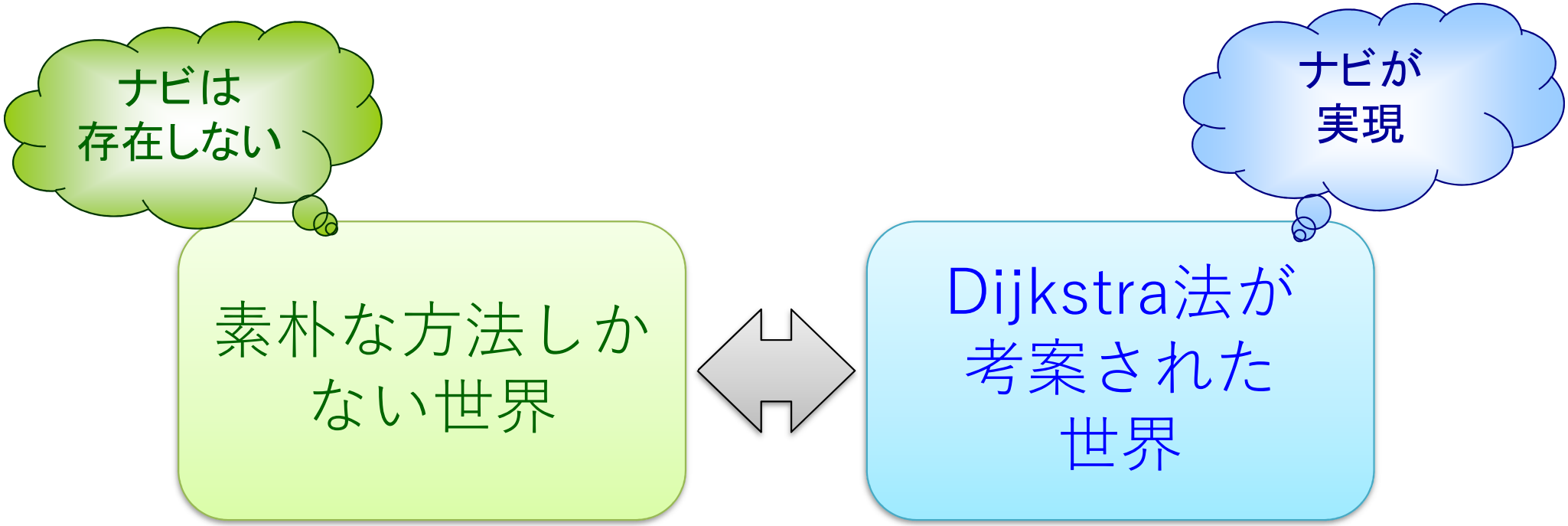
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

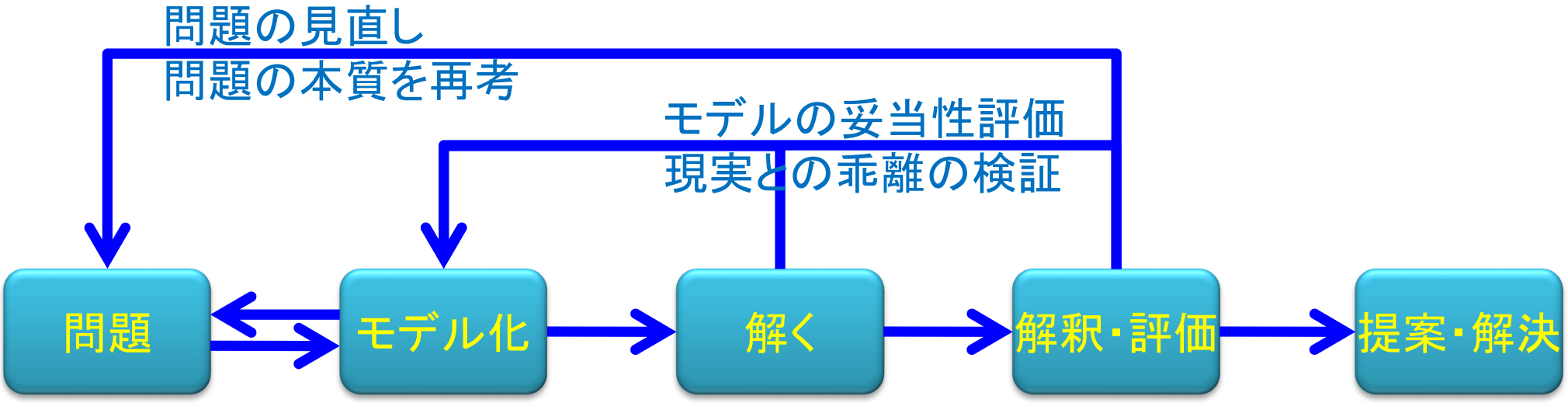
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



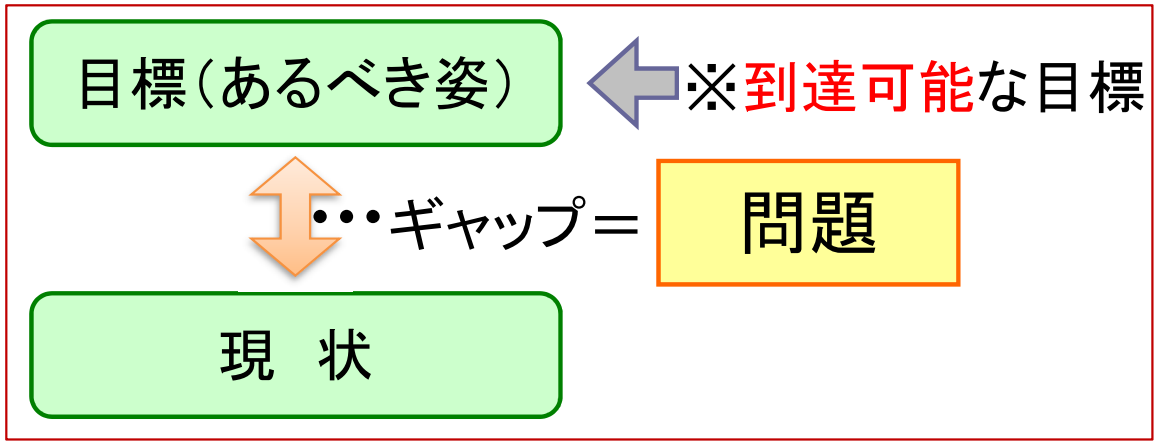
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

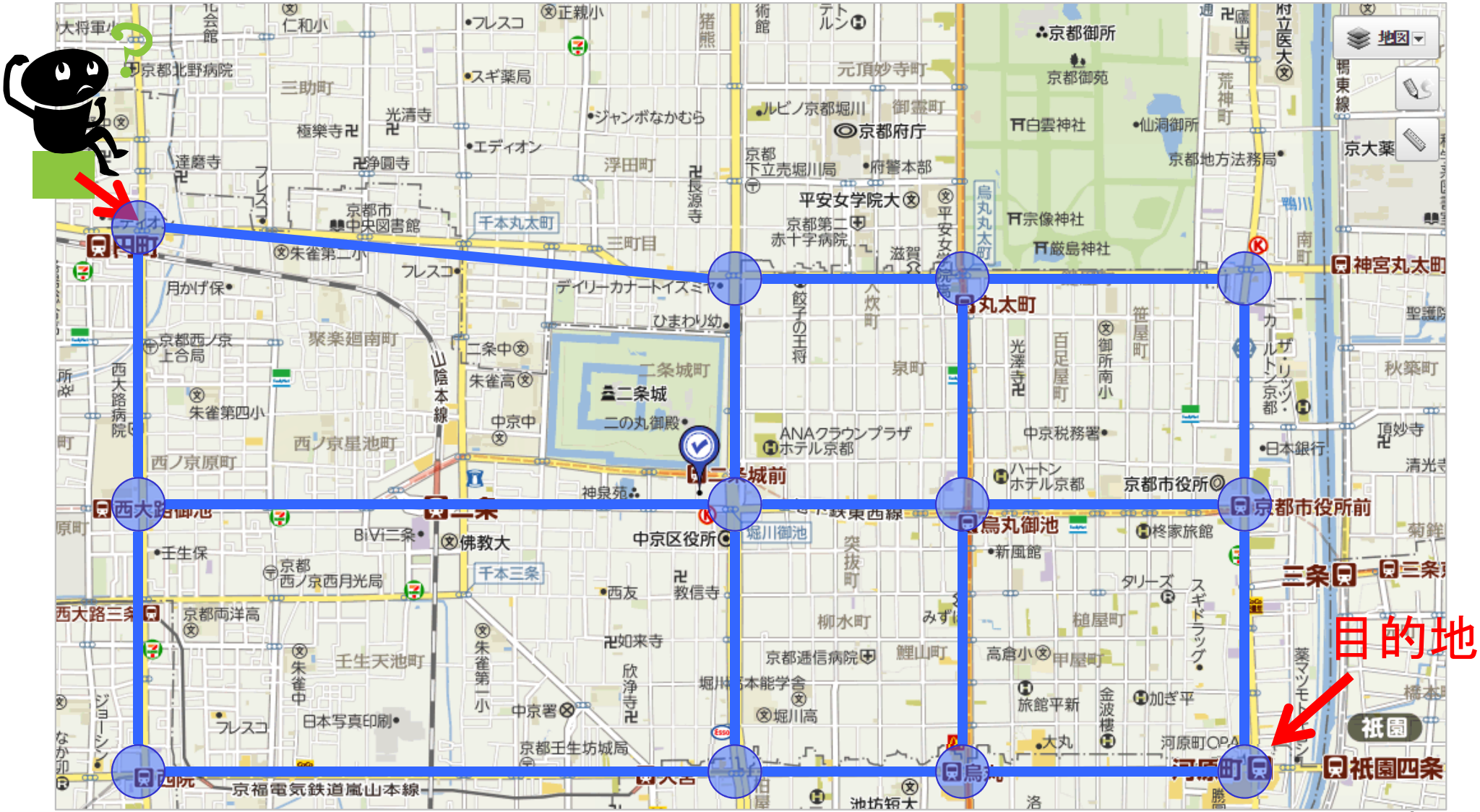
意思決定

問題の定義



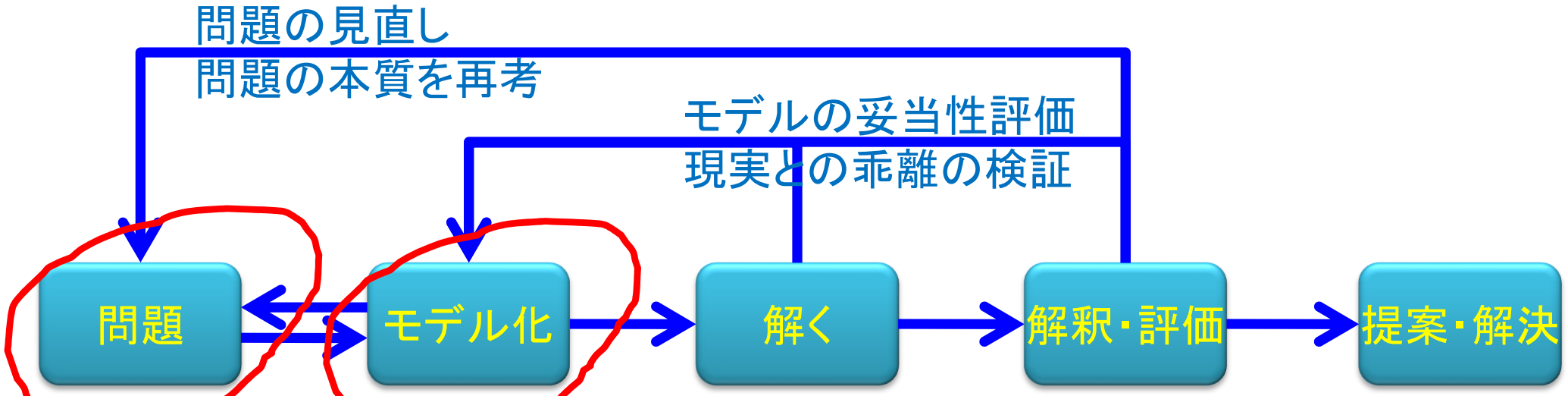
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

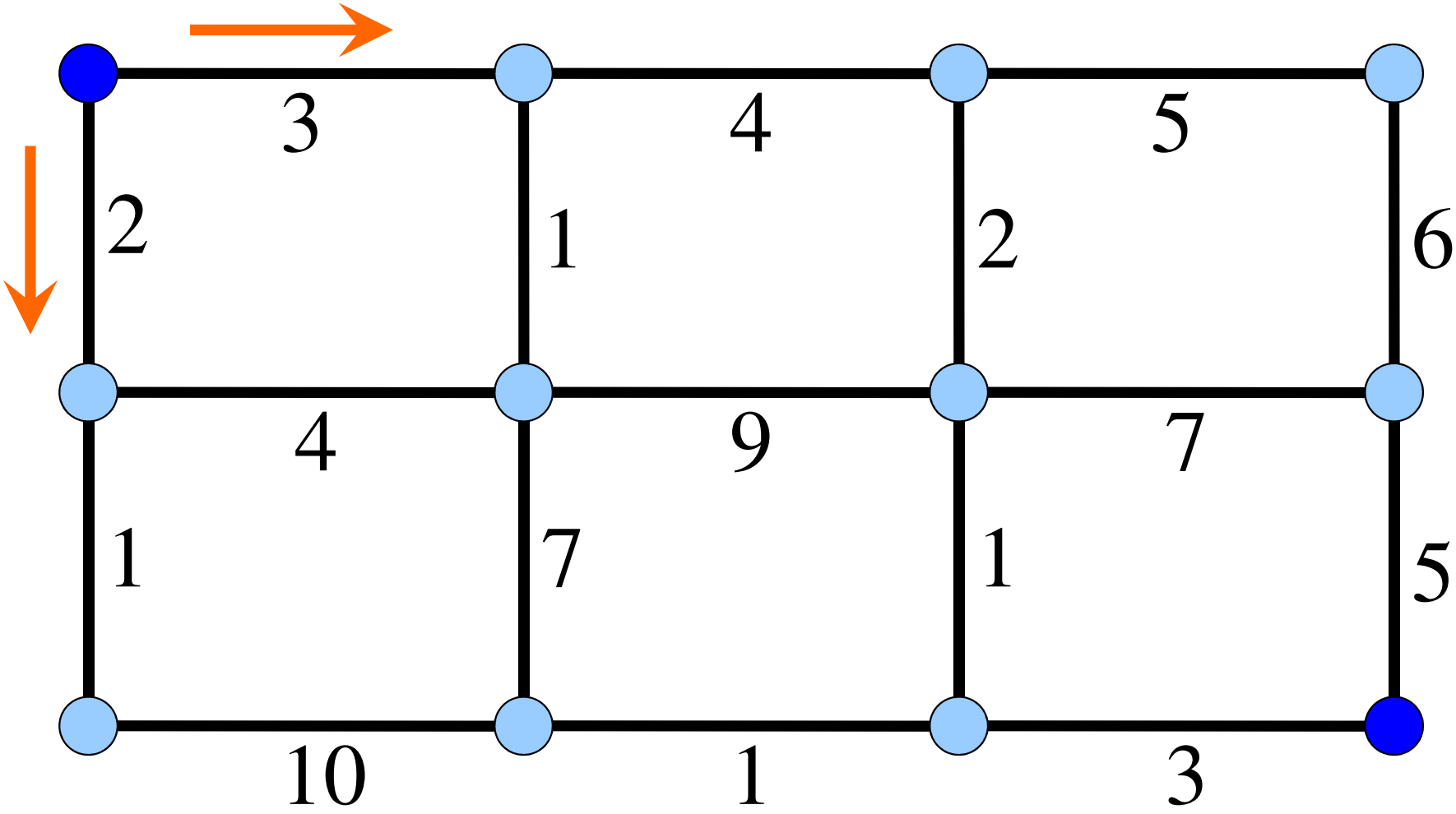
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

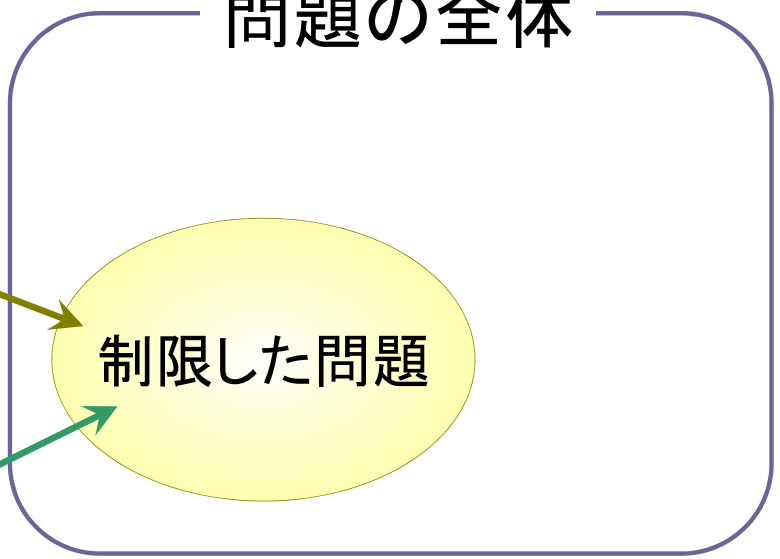


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

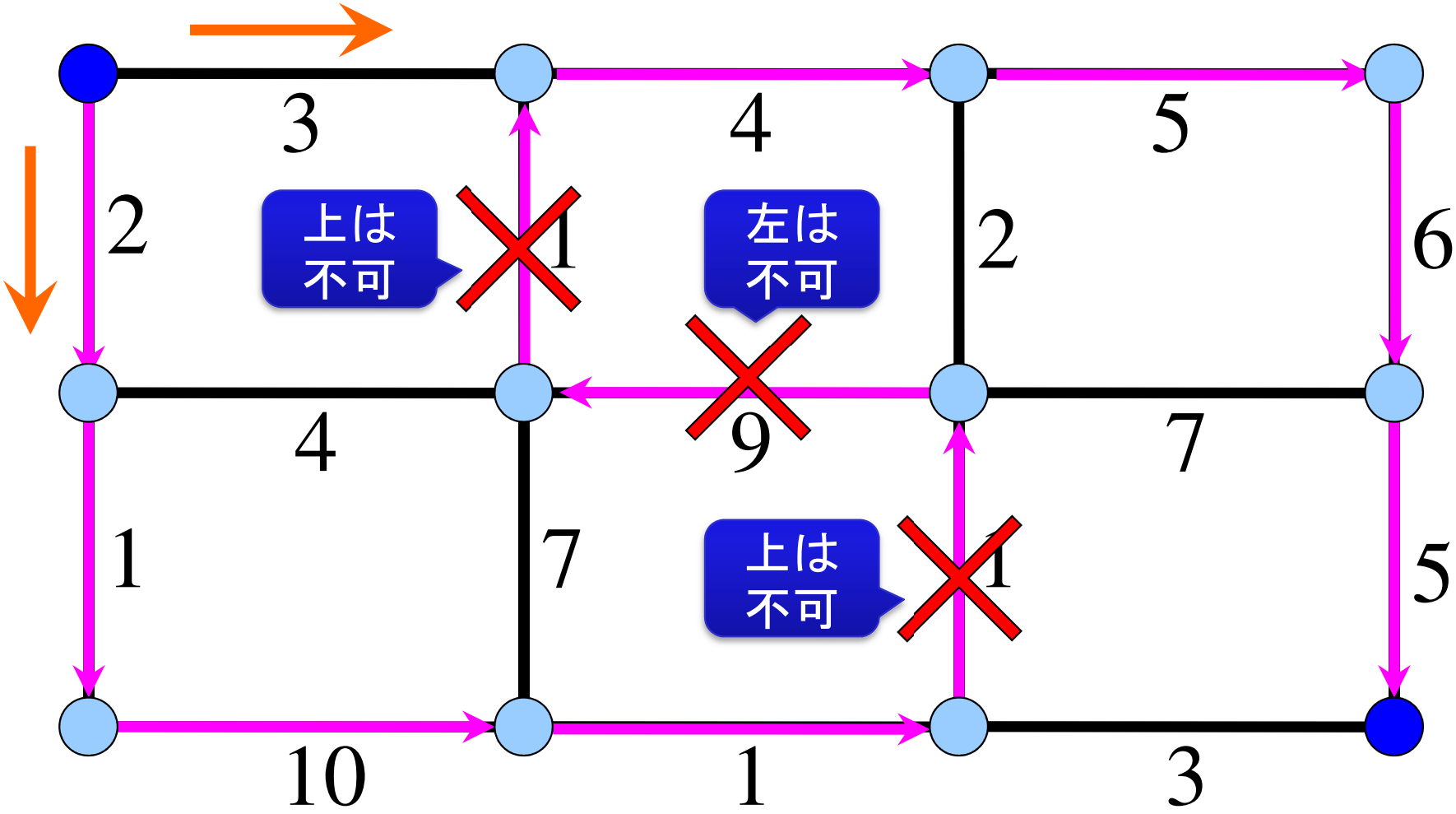
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

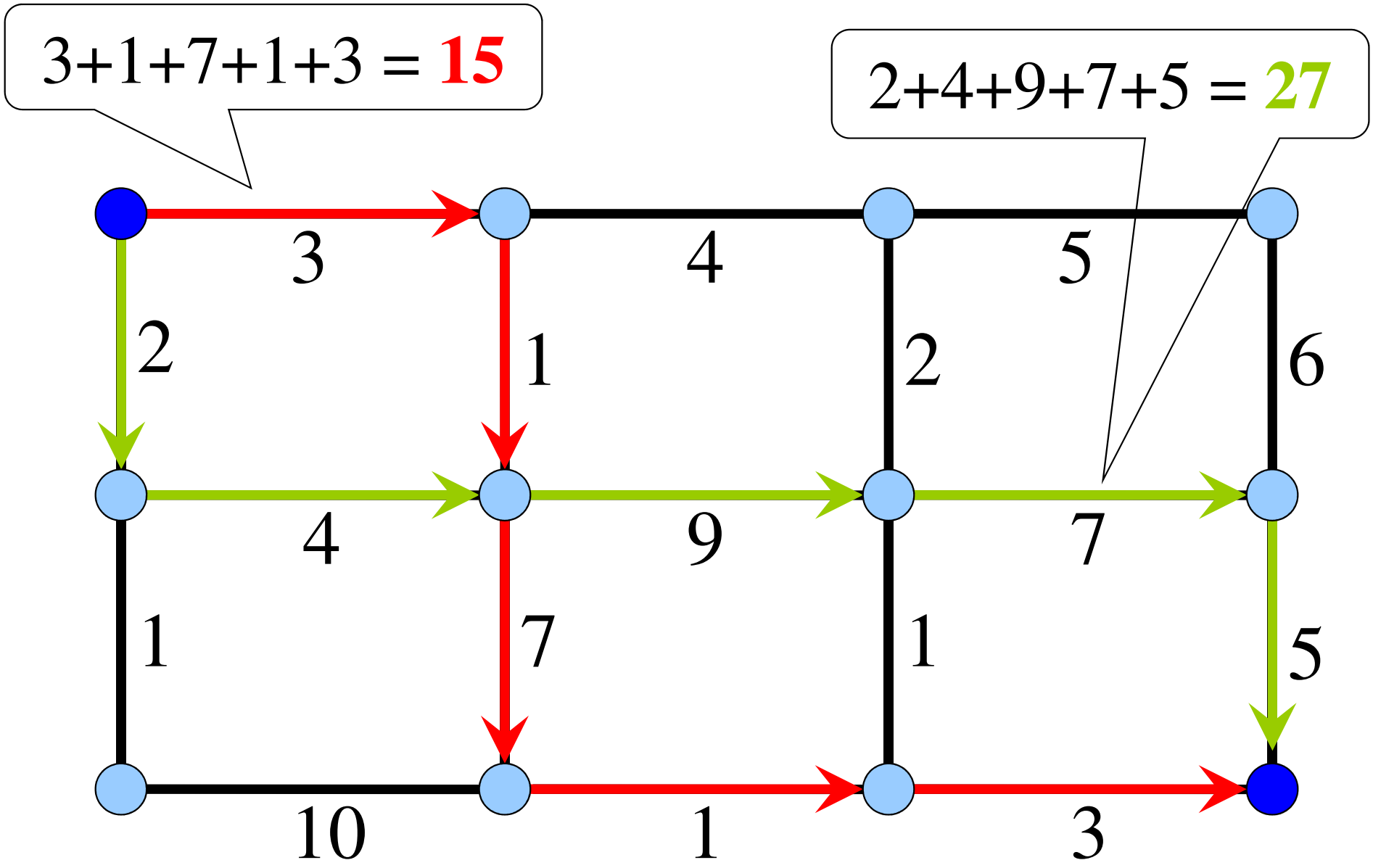
難しいなら易しくすればいいのさ！

制限した問題

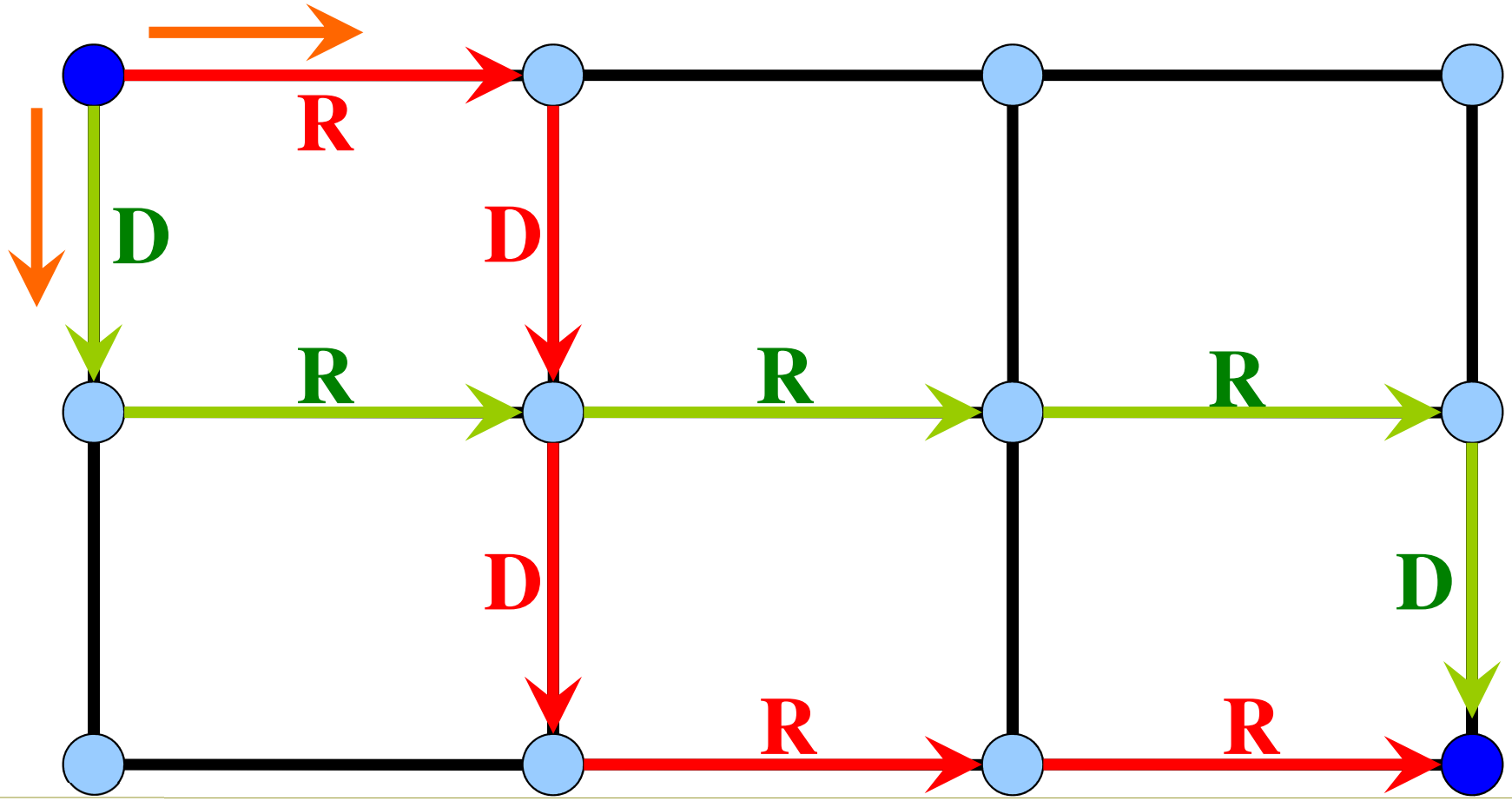
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



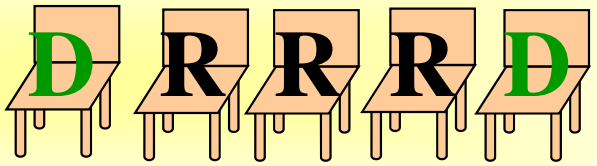
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



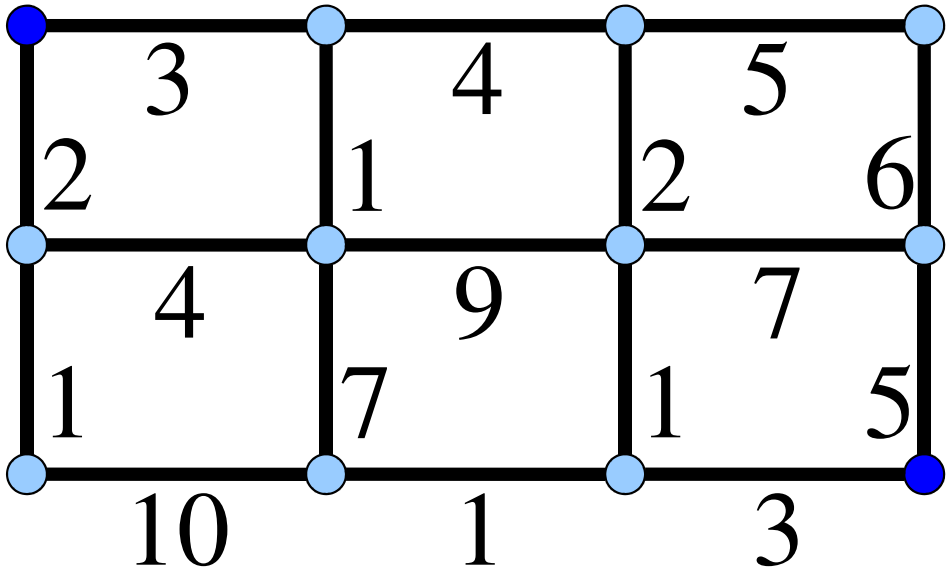
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

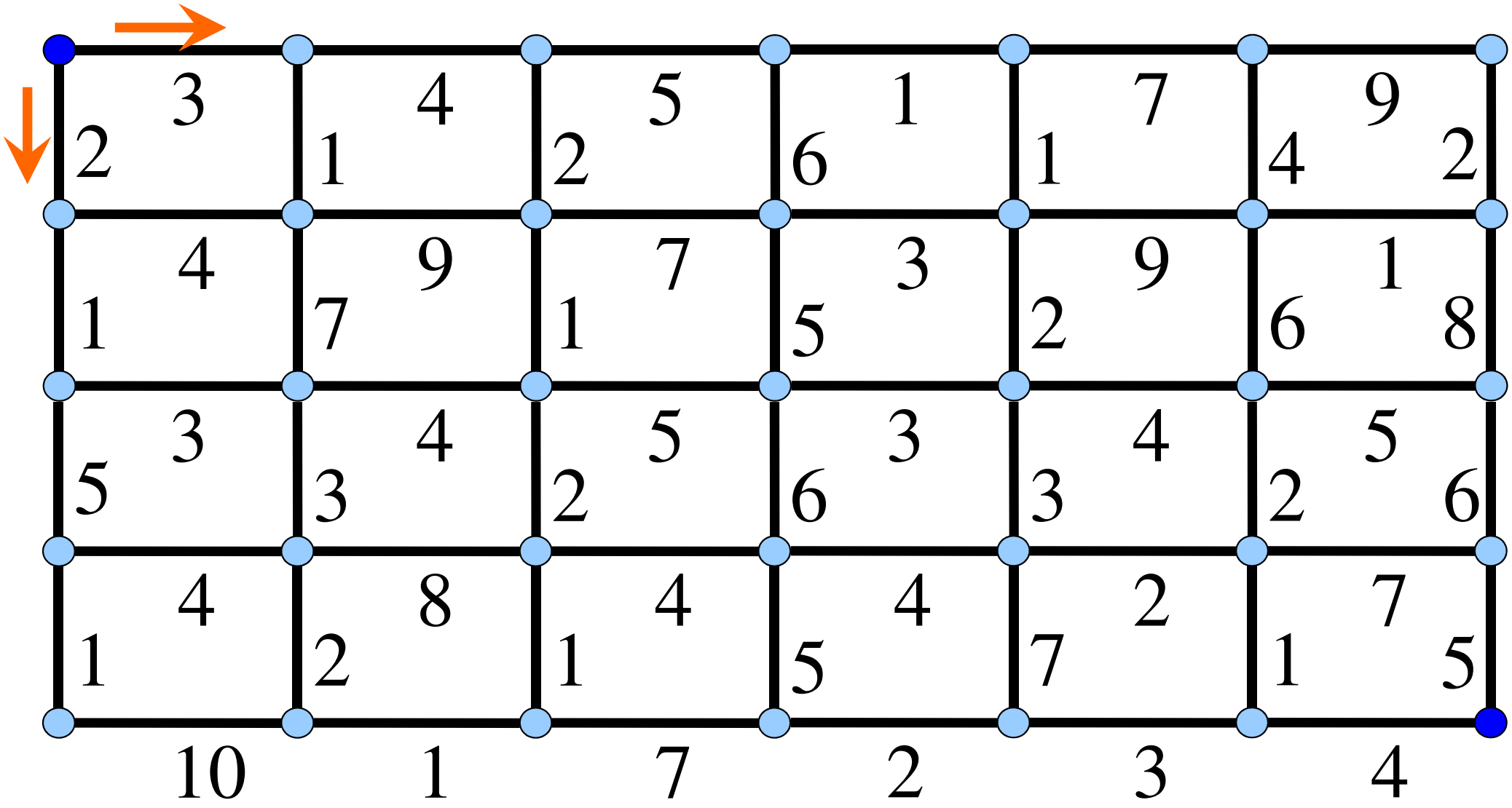
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

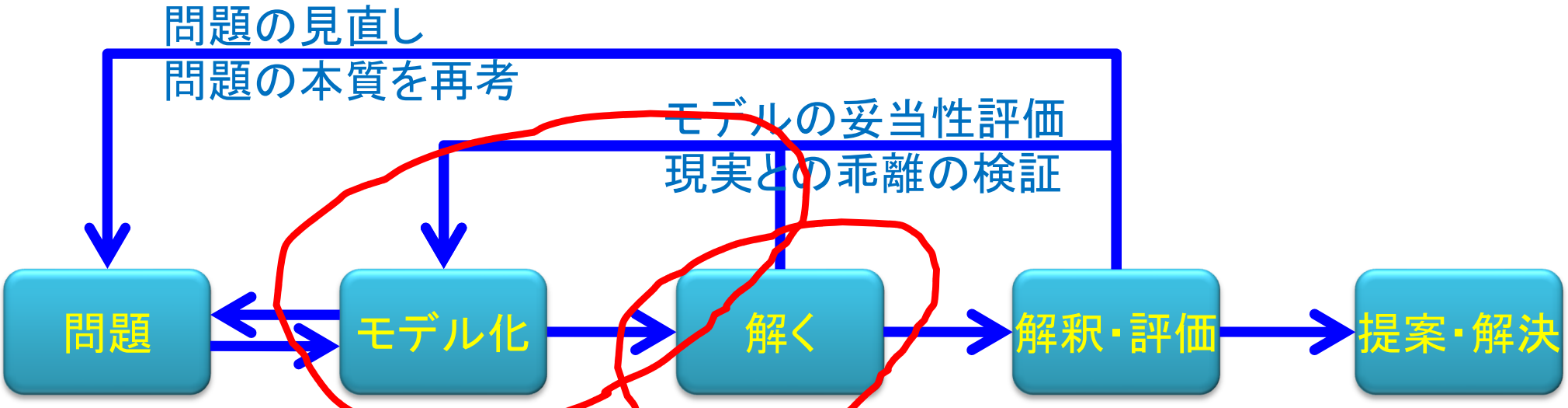
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

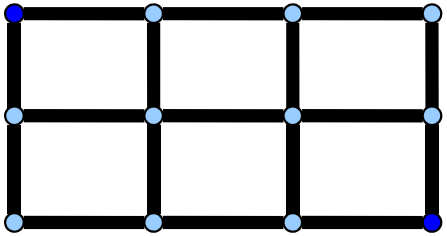
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

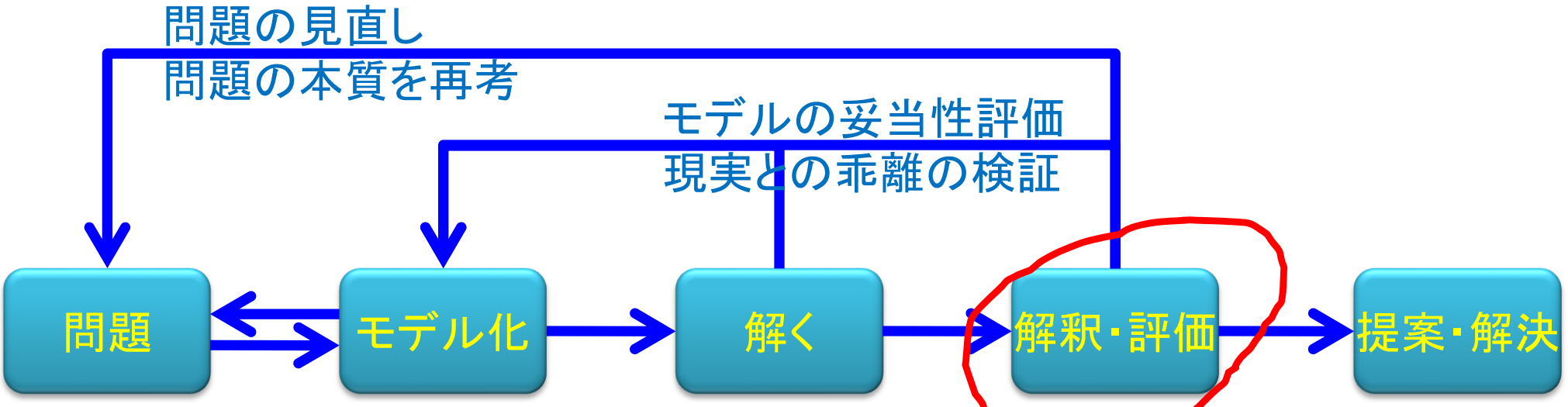
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

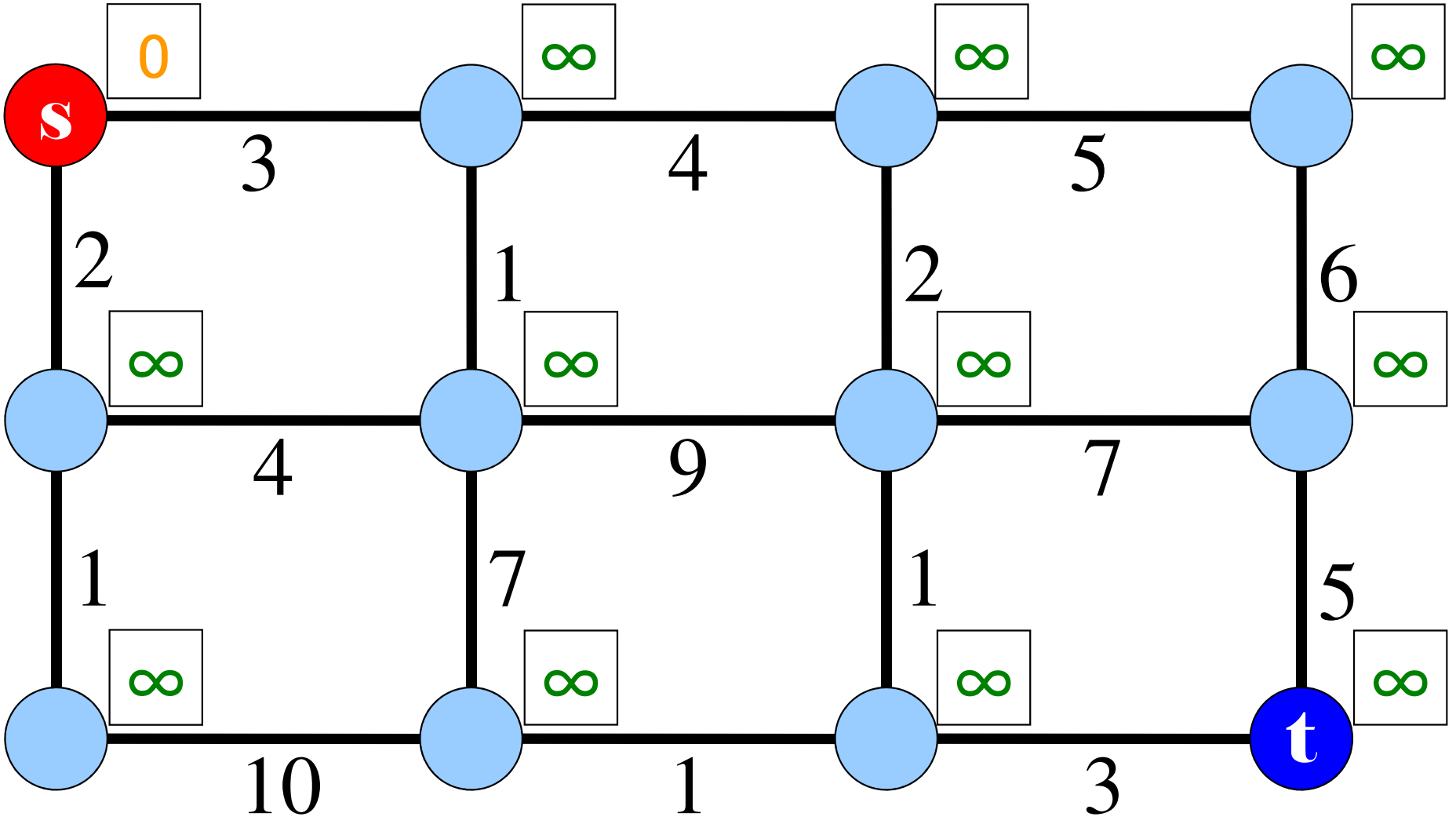
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

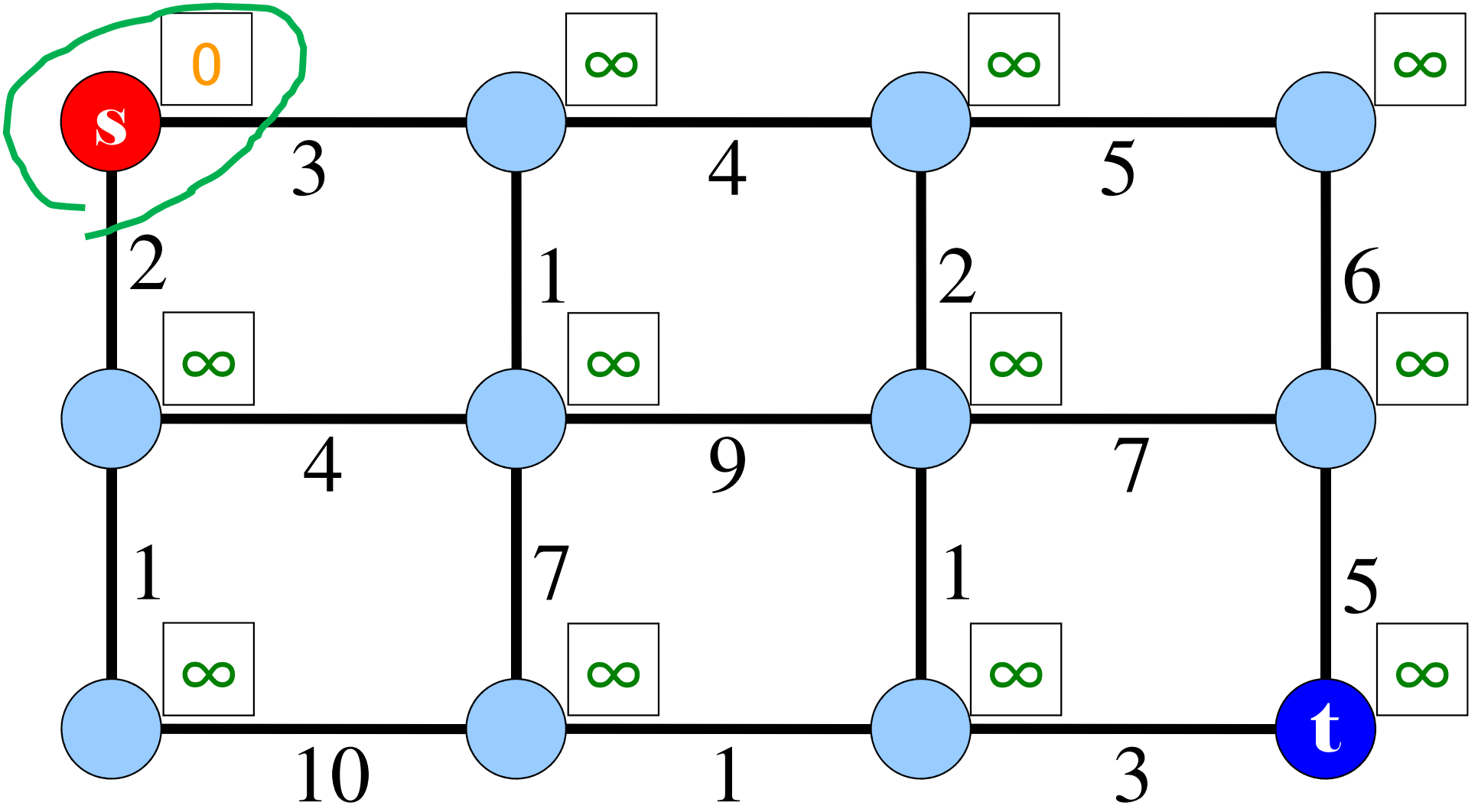
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



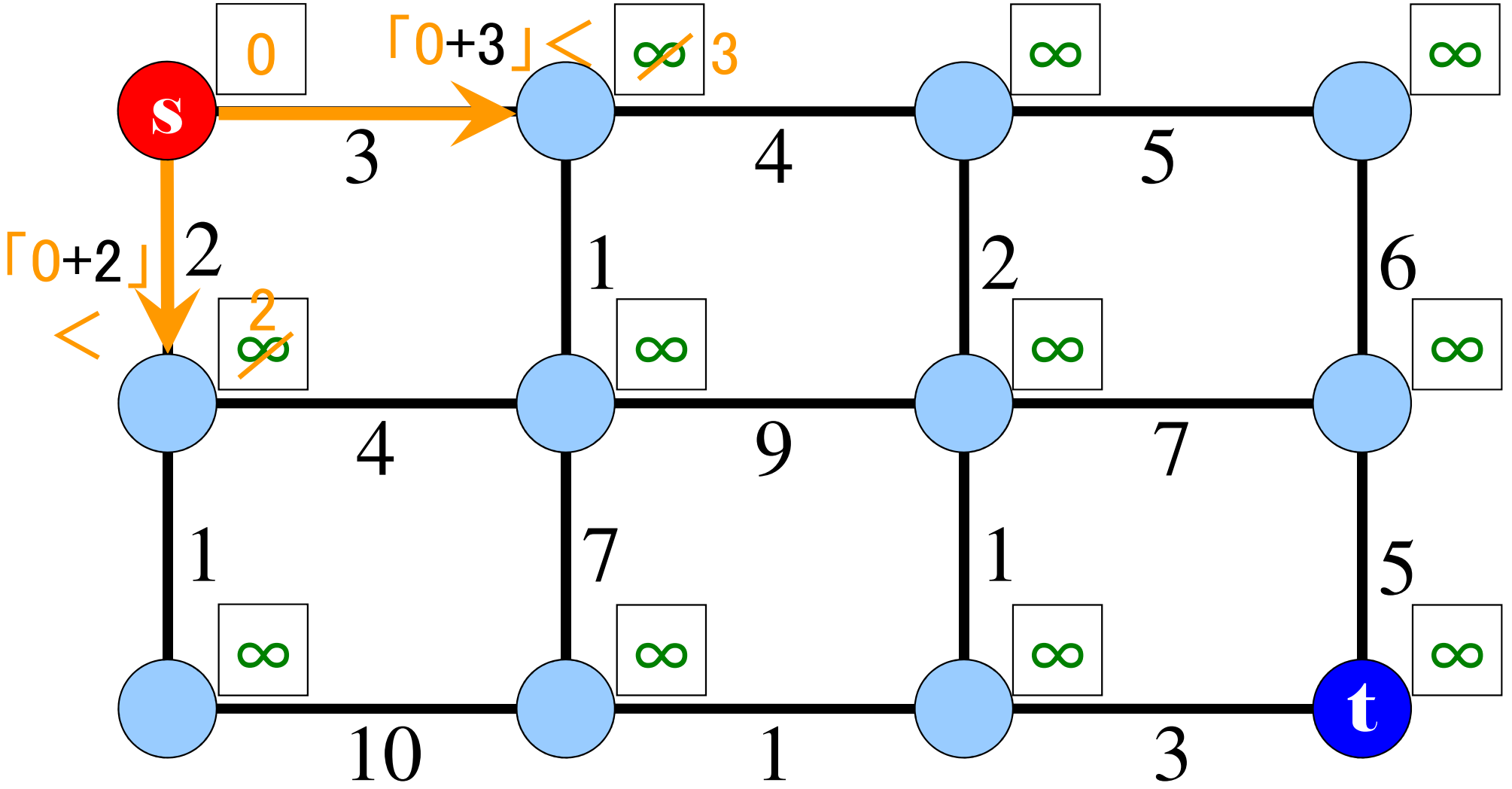
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

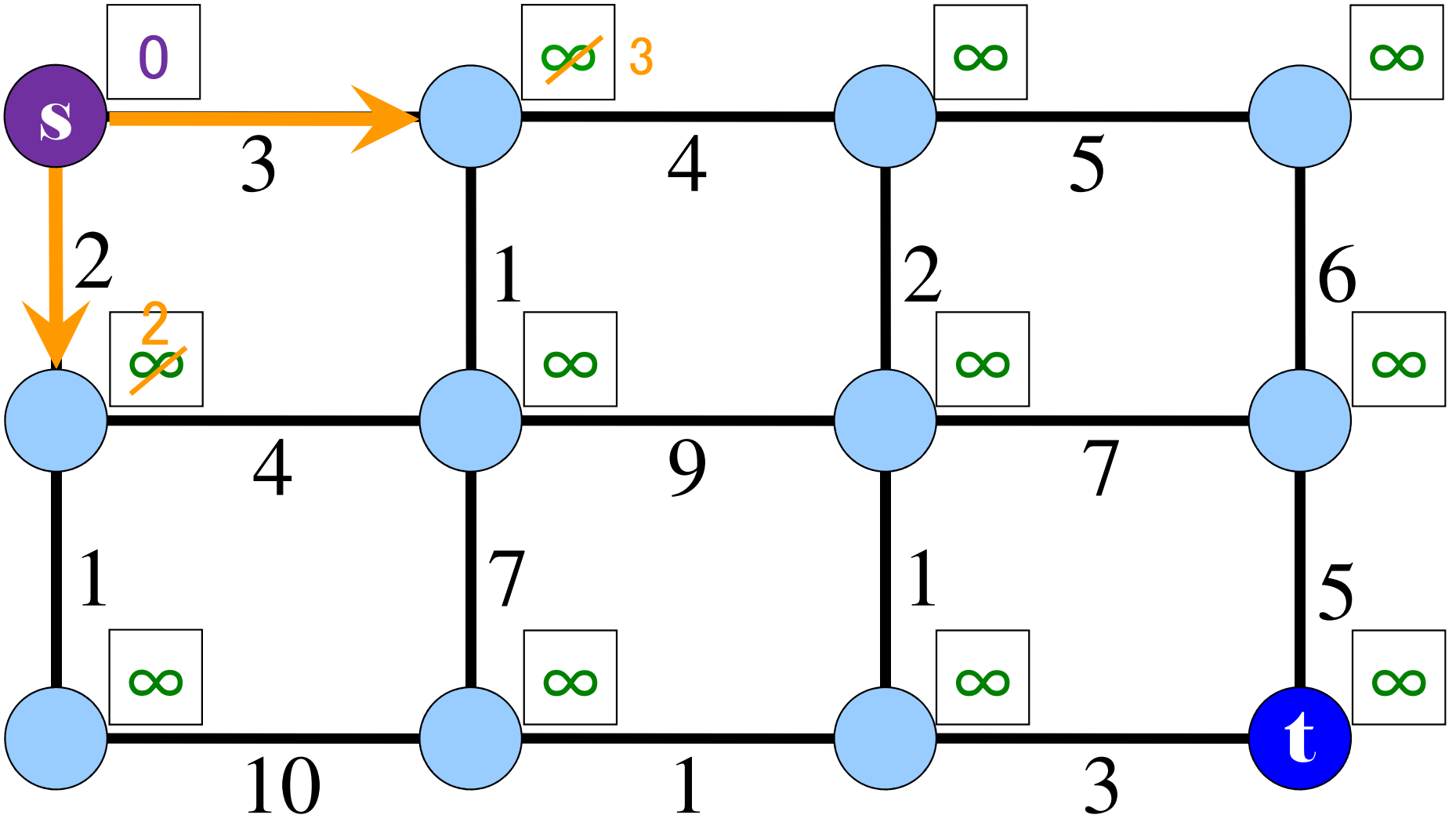


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

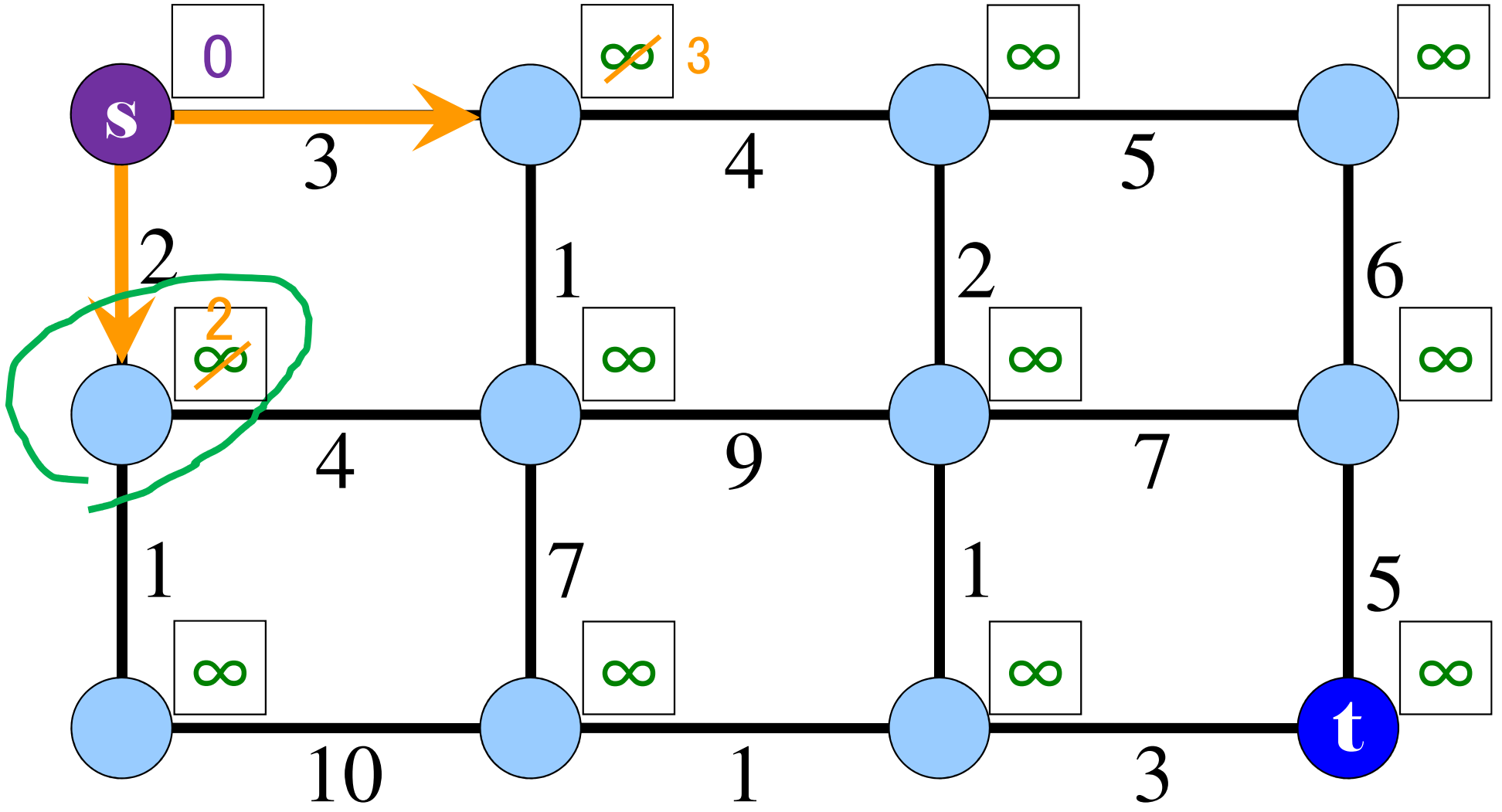
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



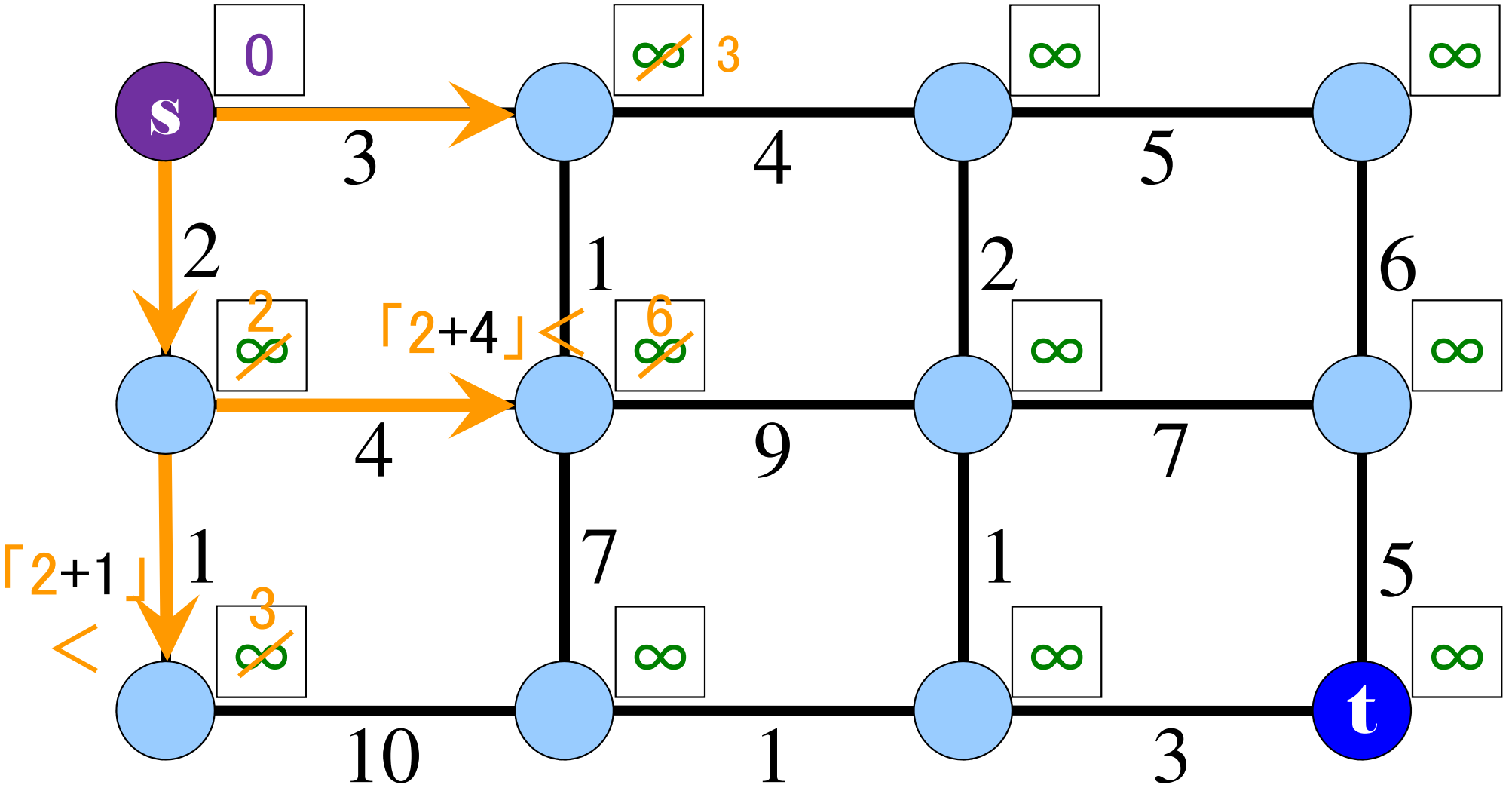
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

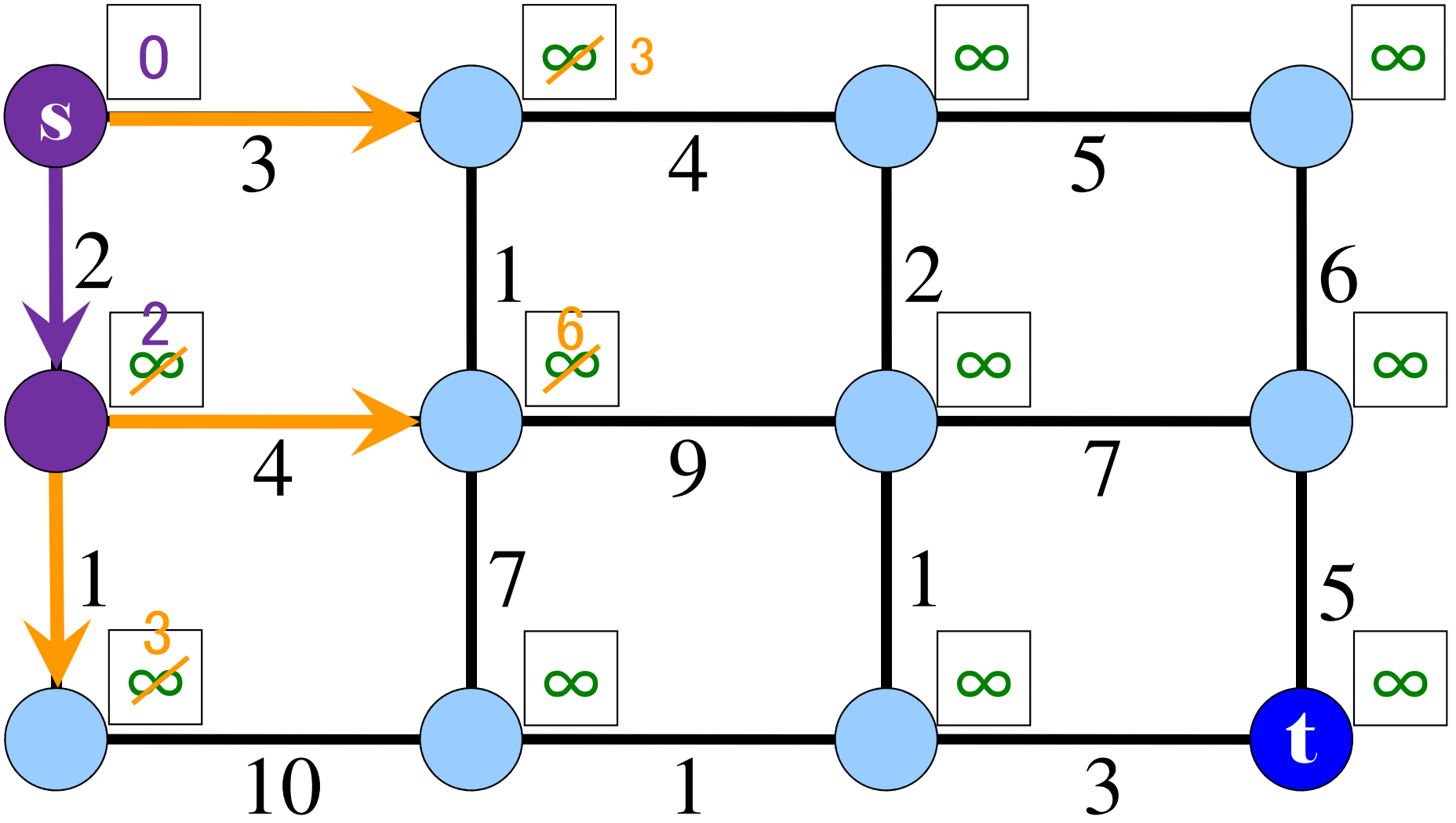


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

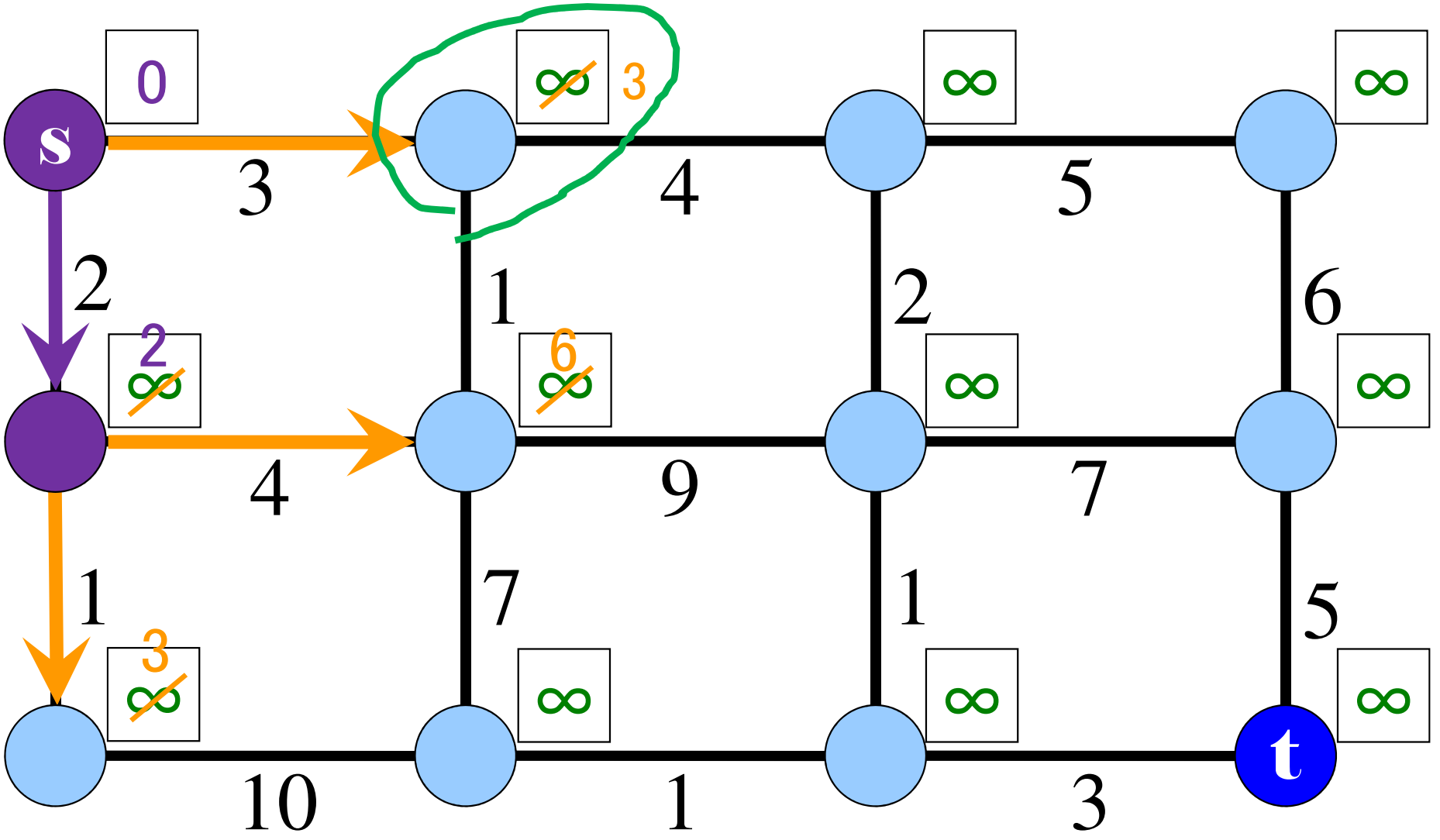
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



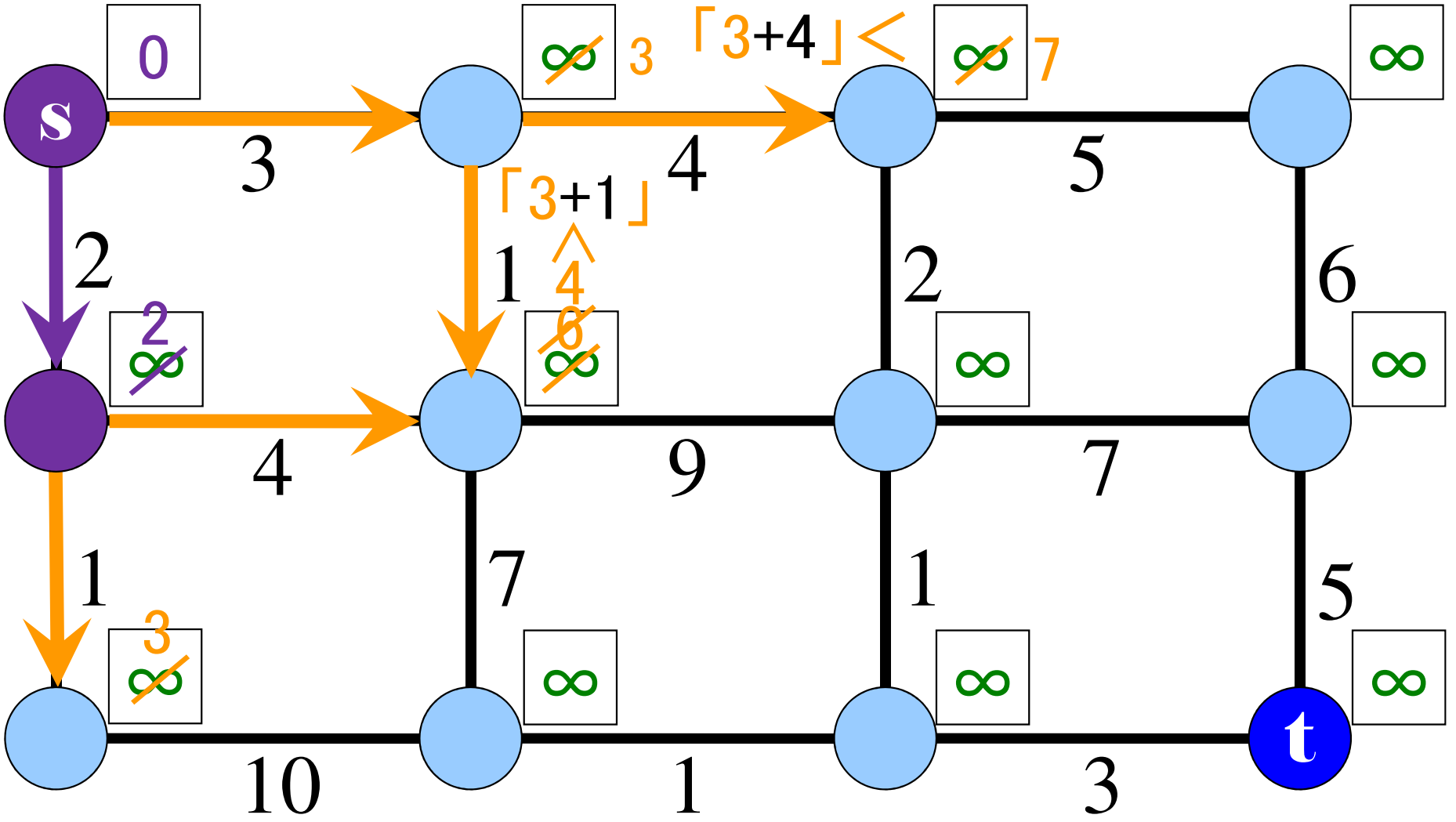
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

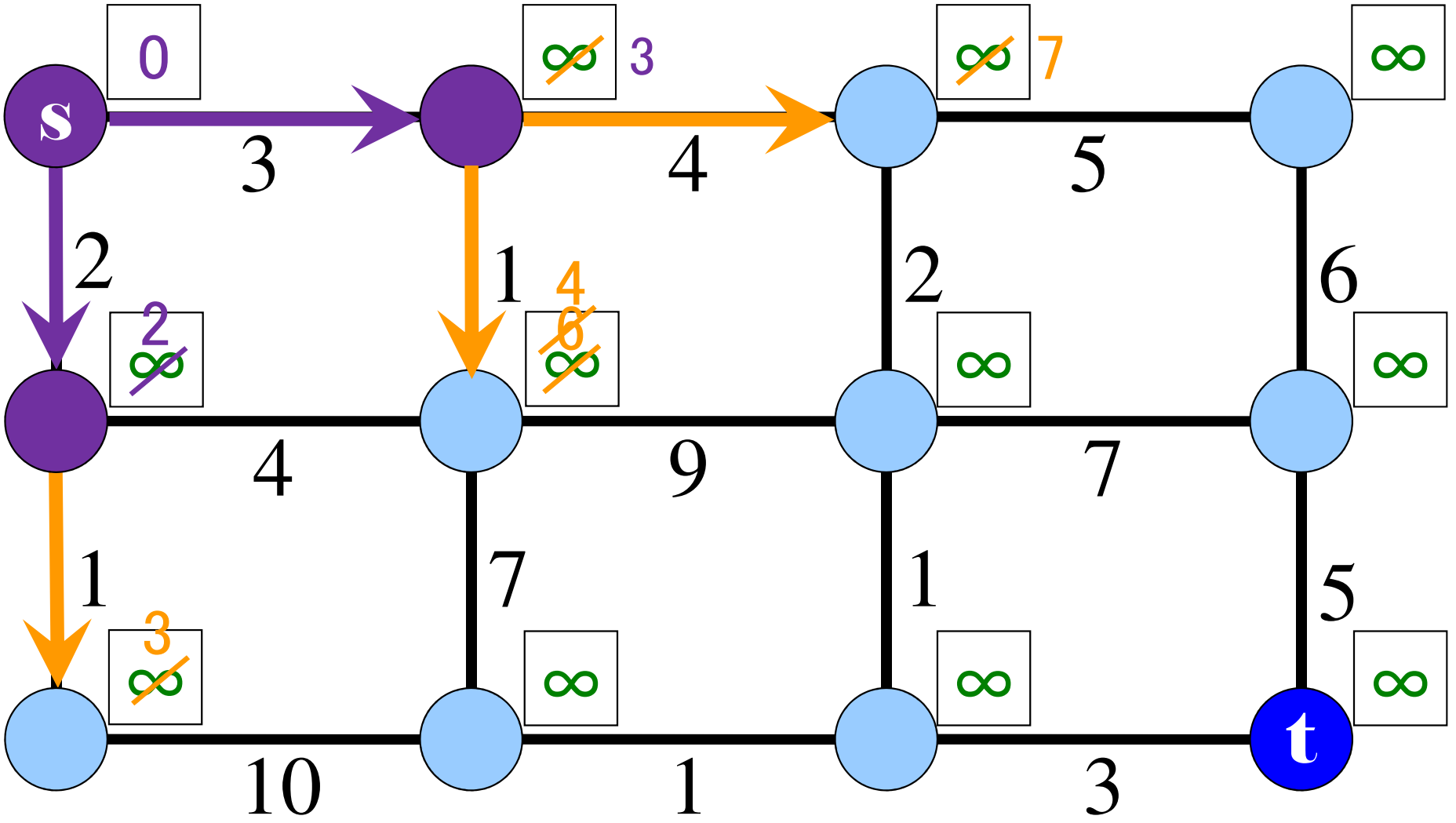


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

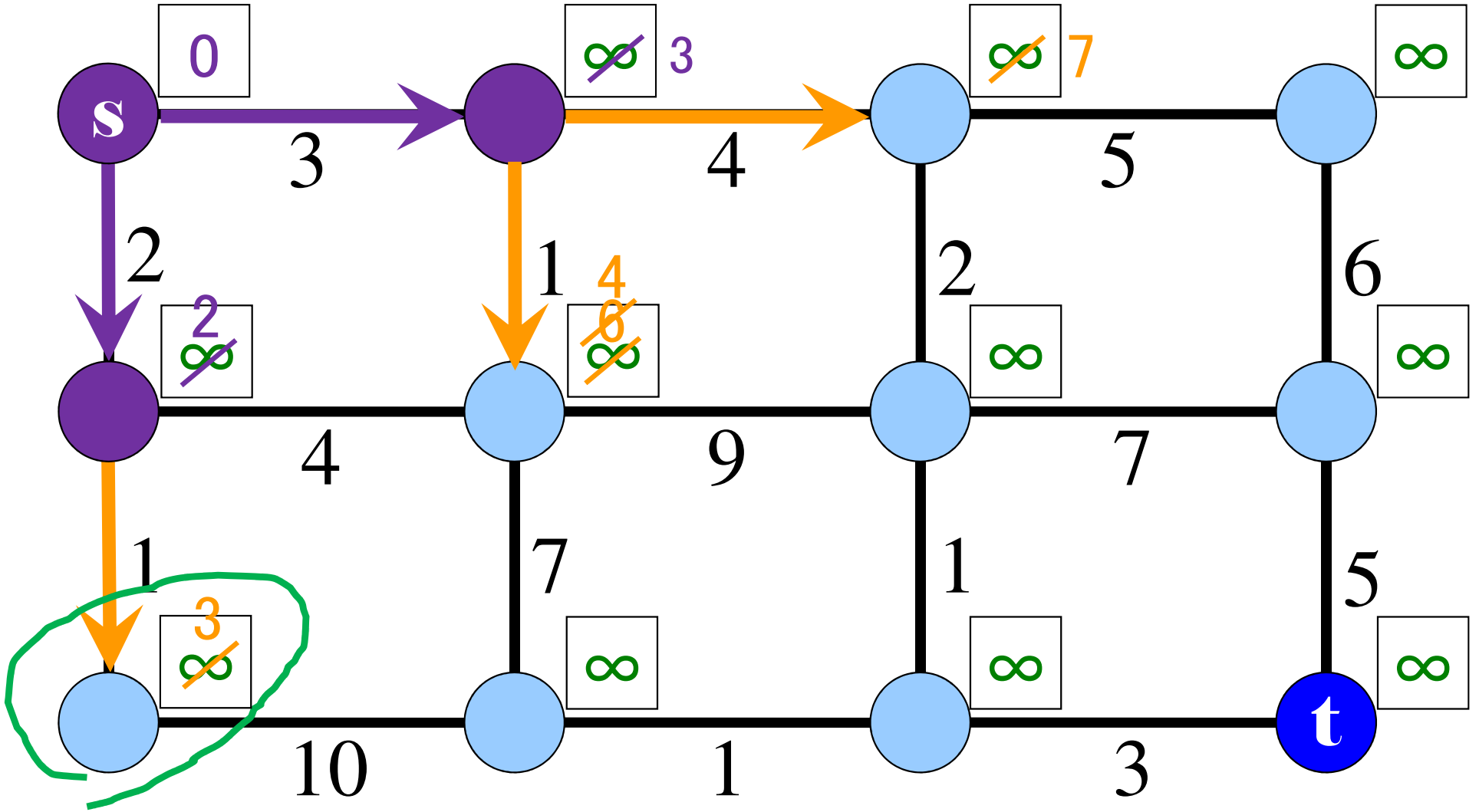
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



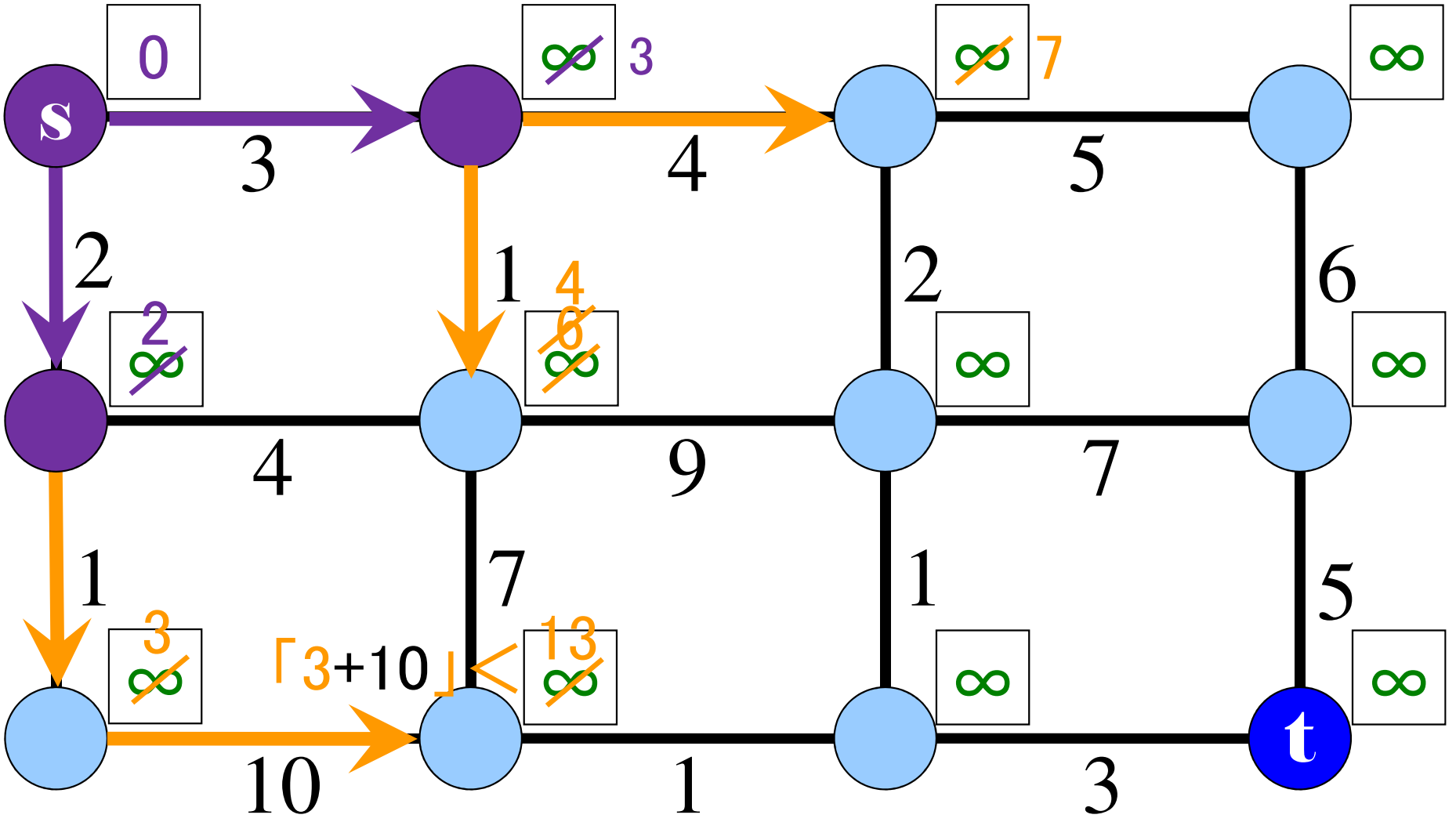
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

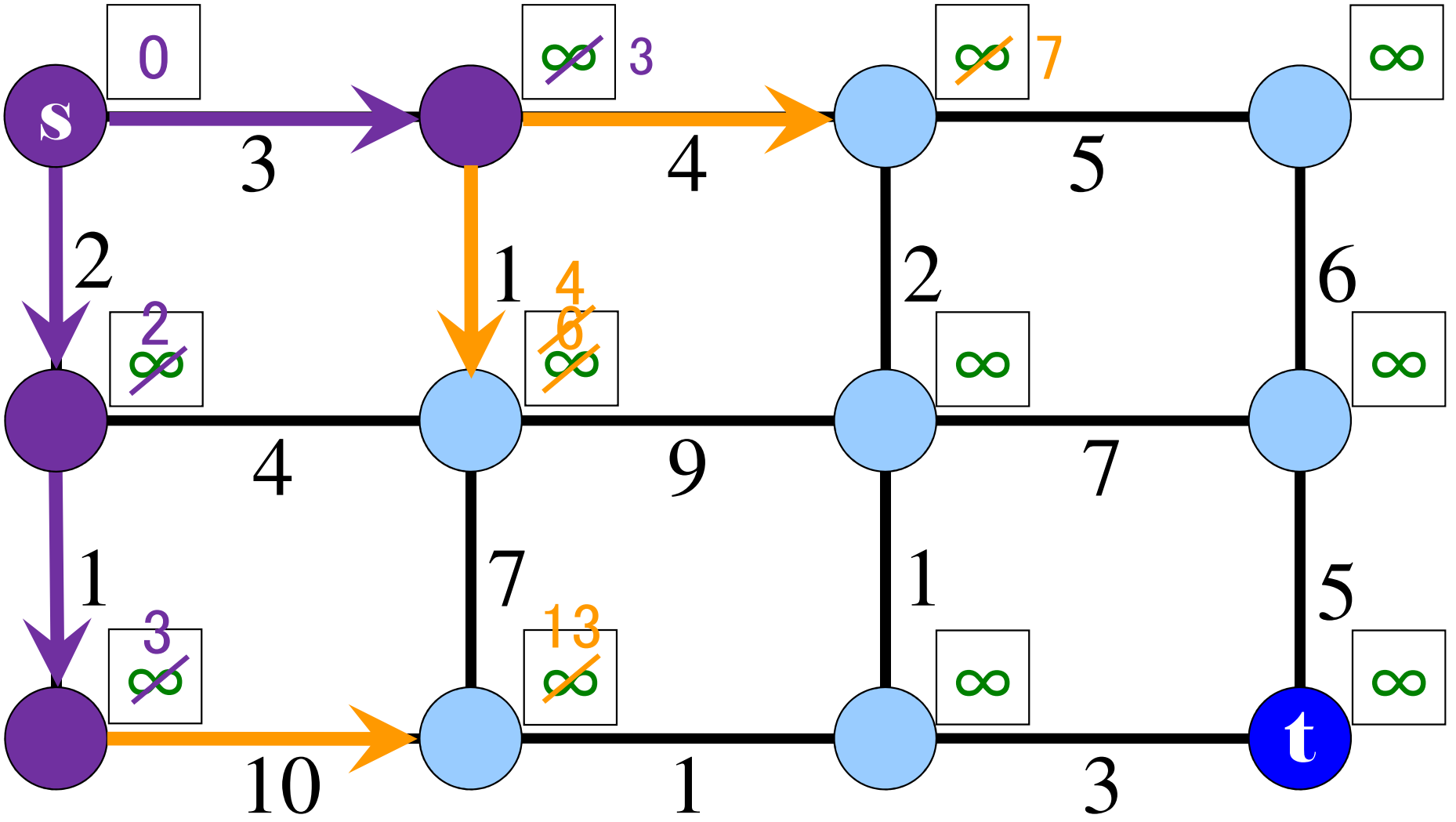


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

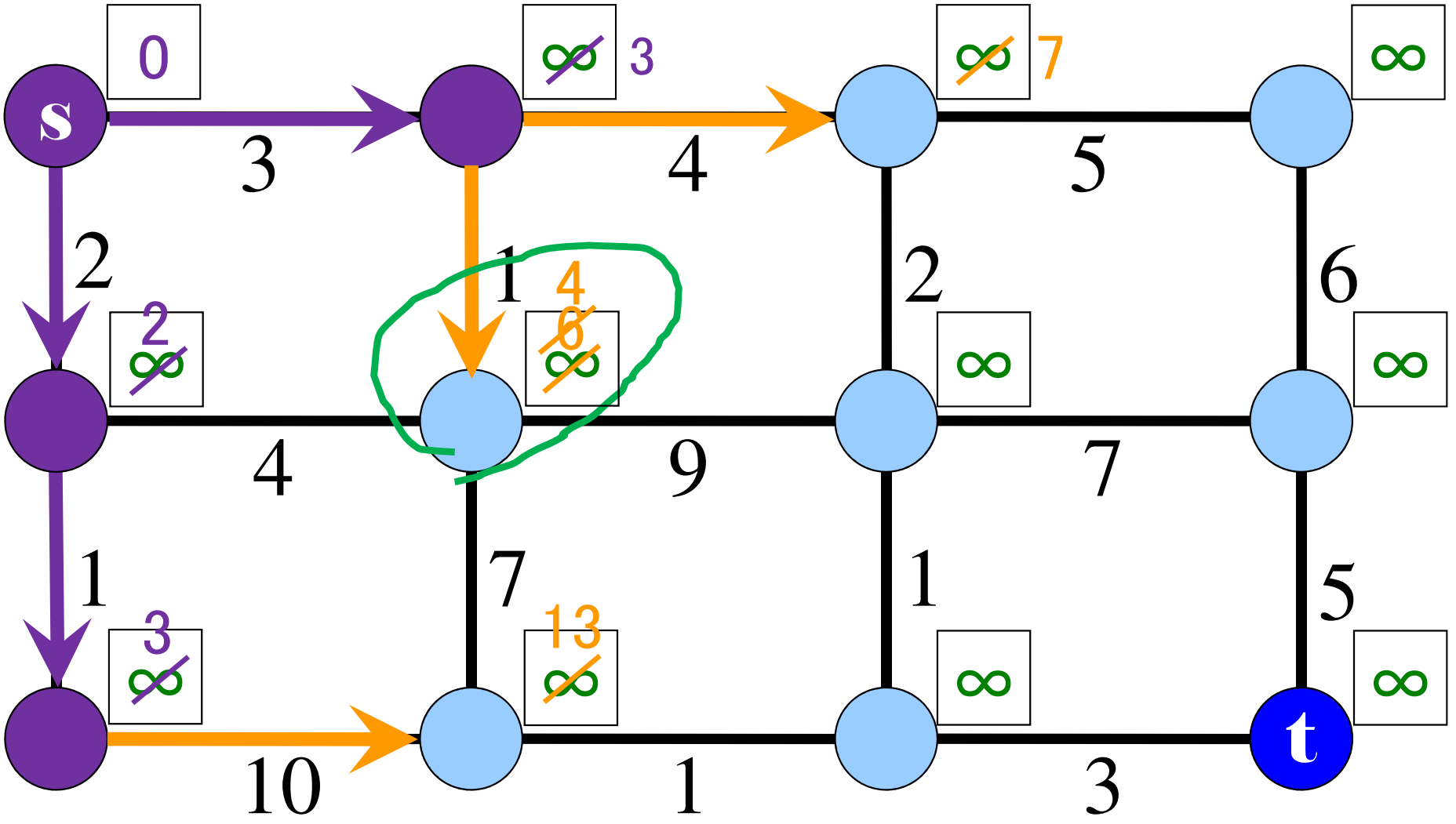
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



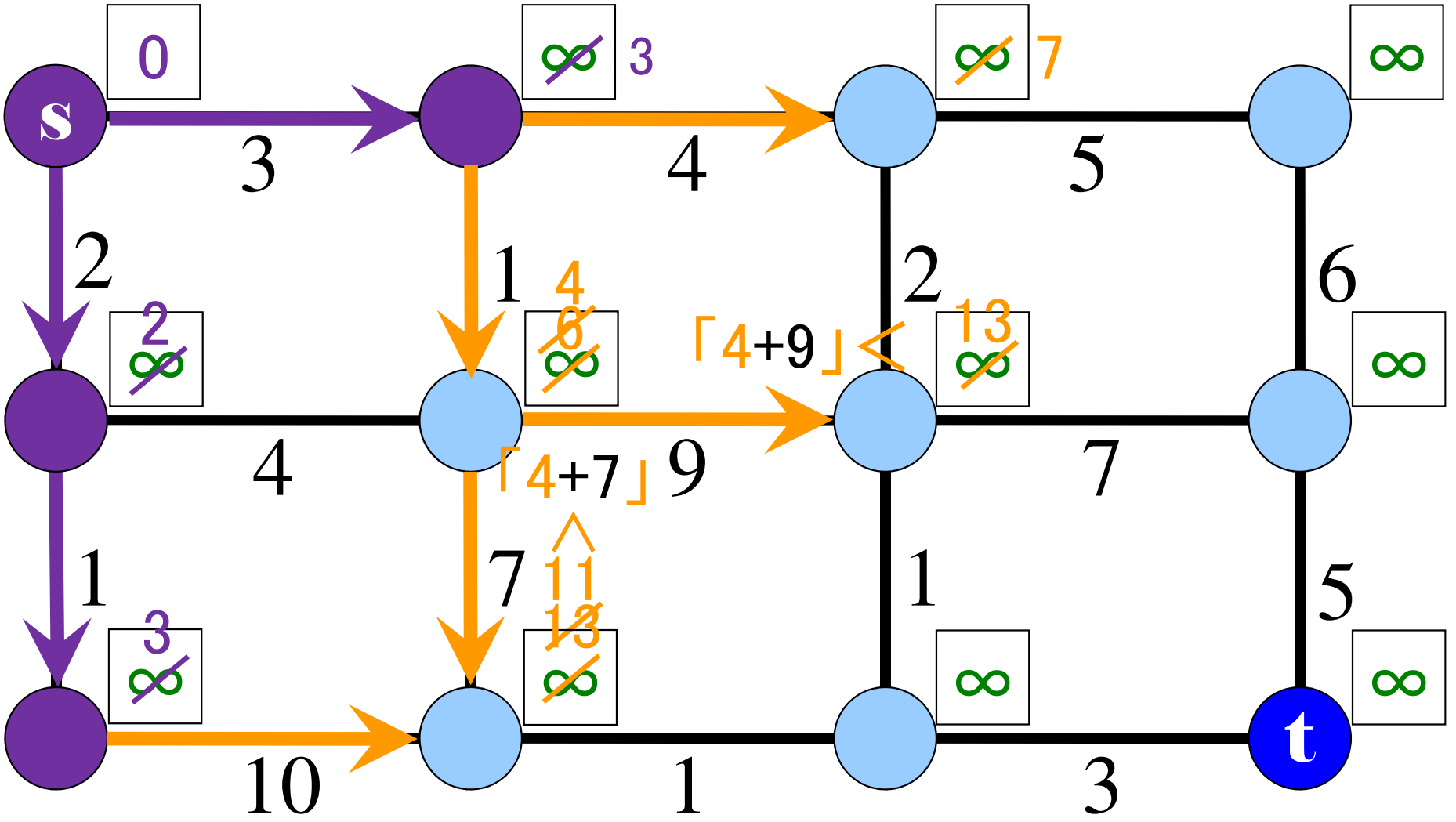
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

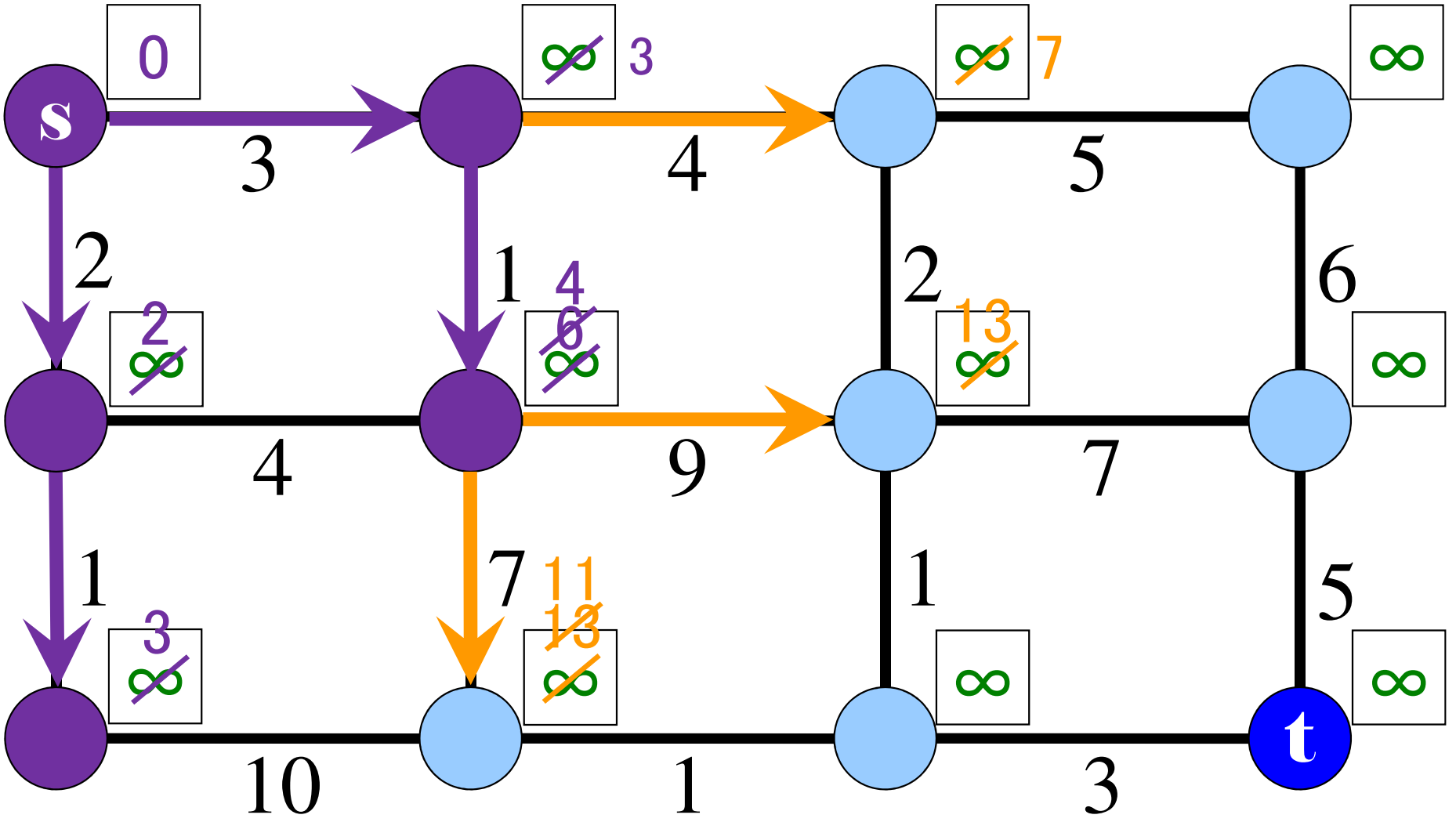


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

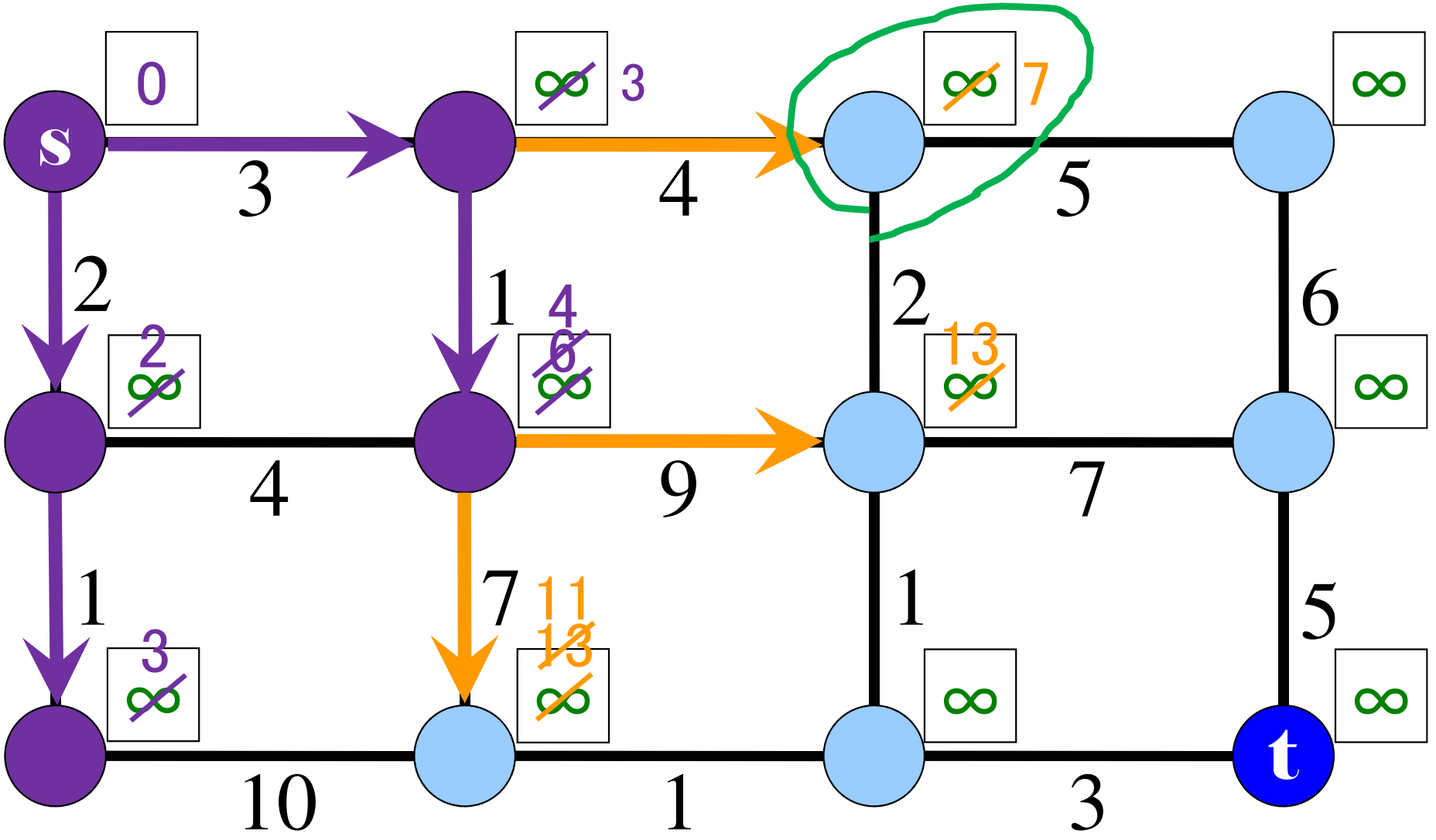
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



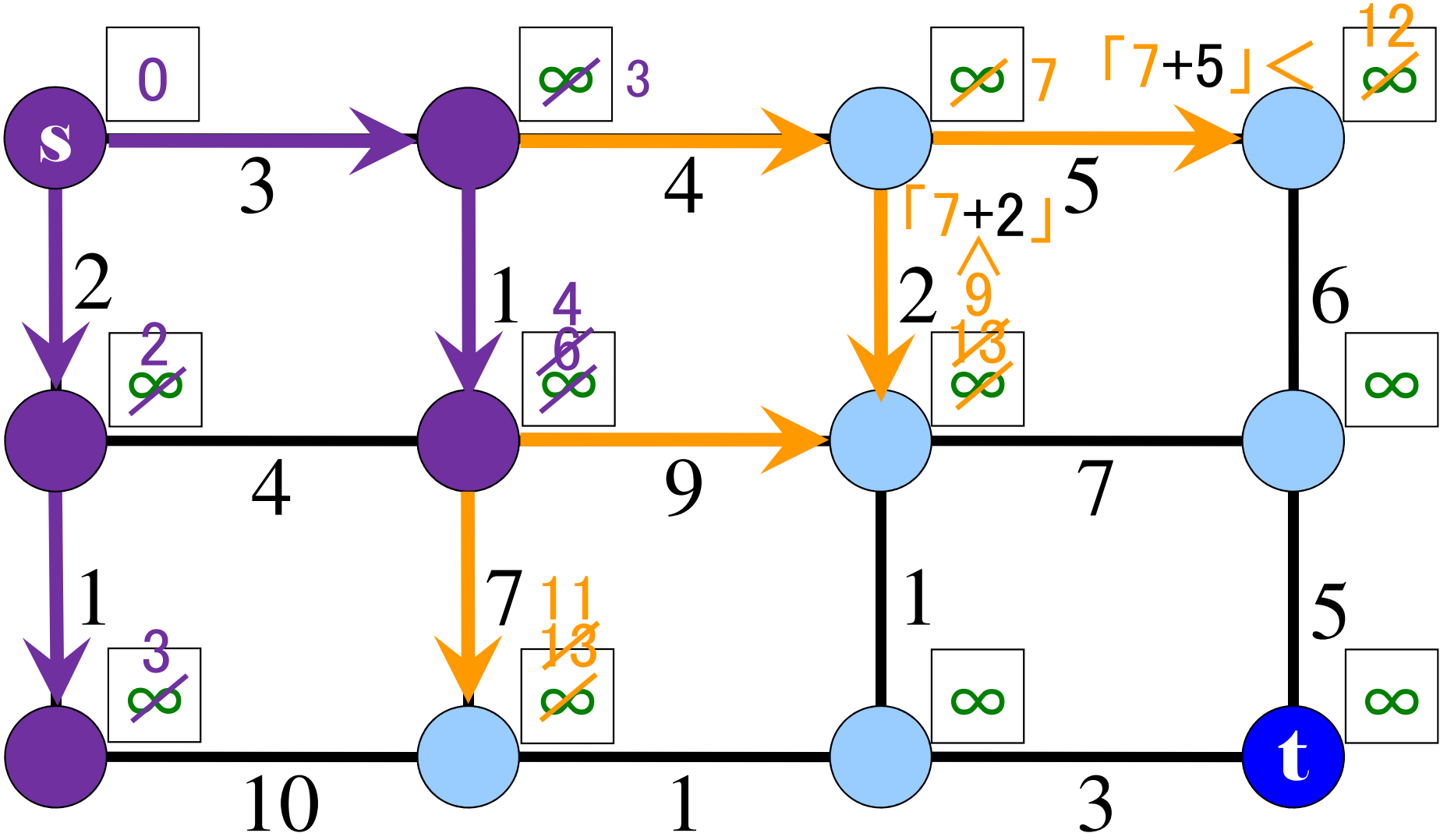
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

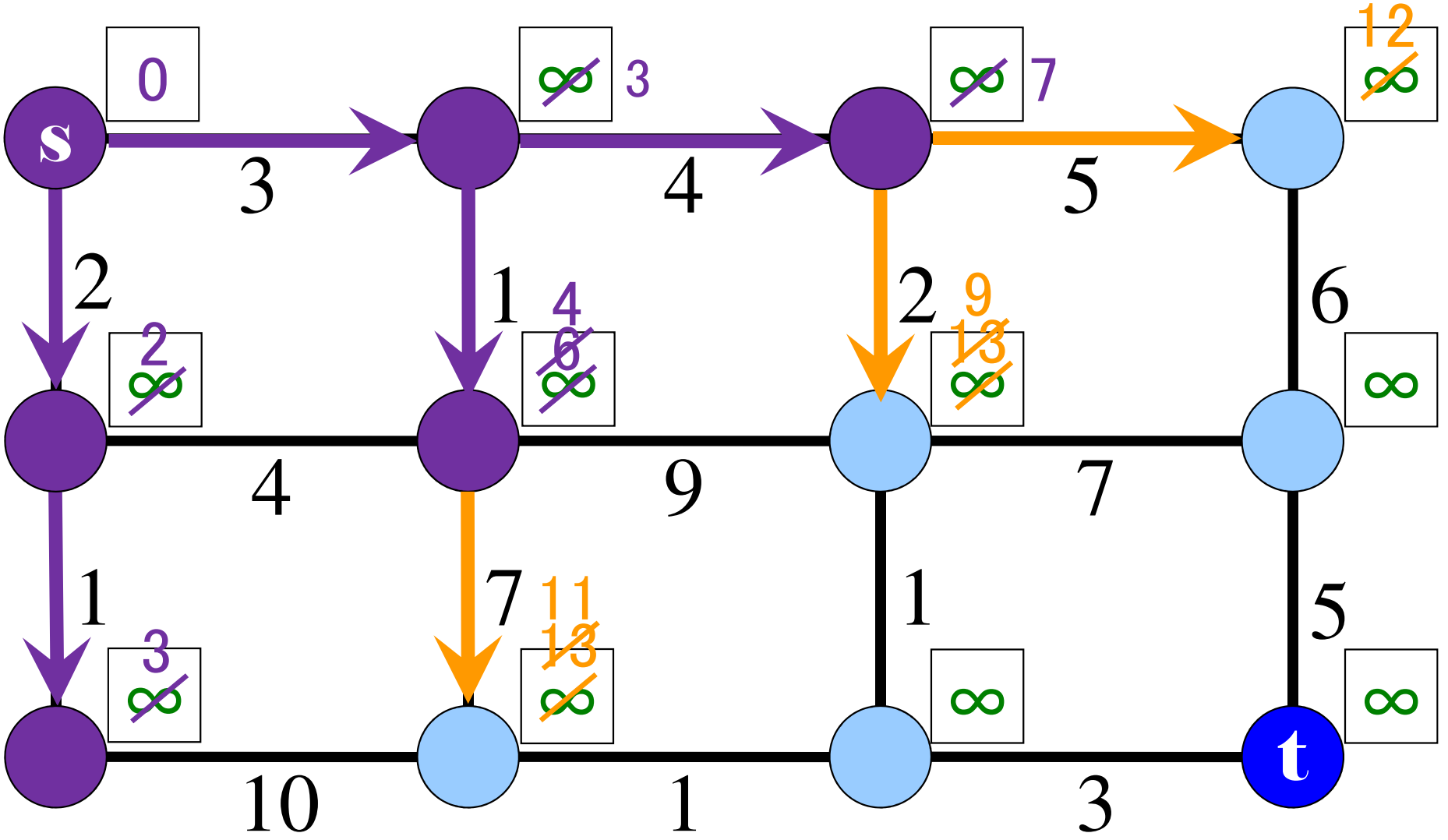


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

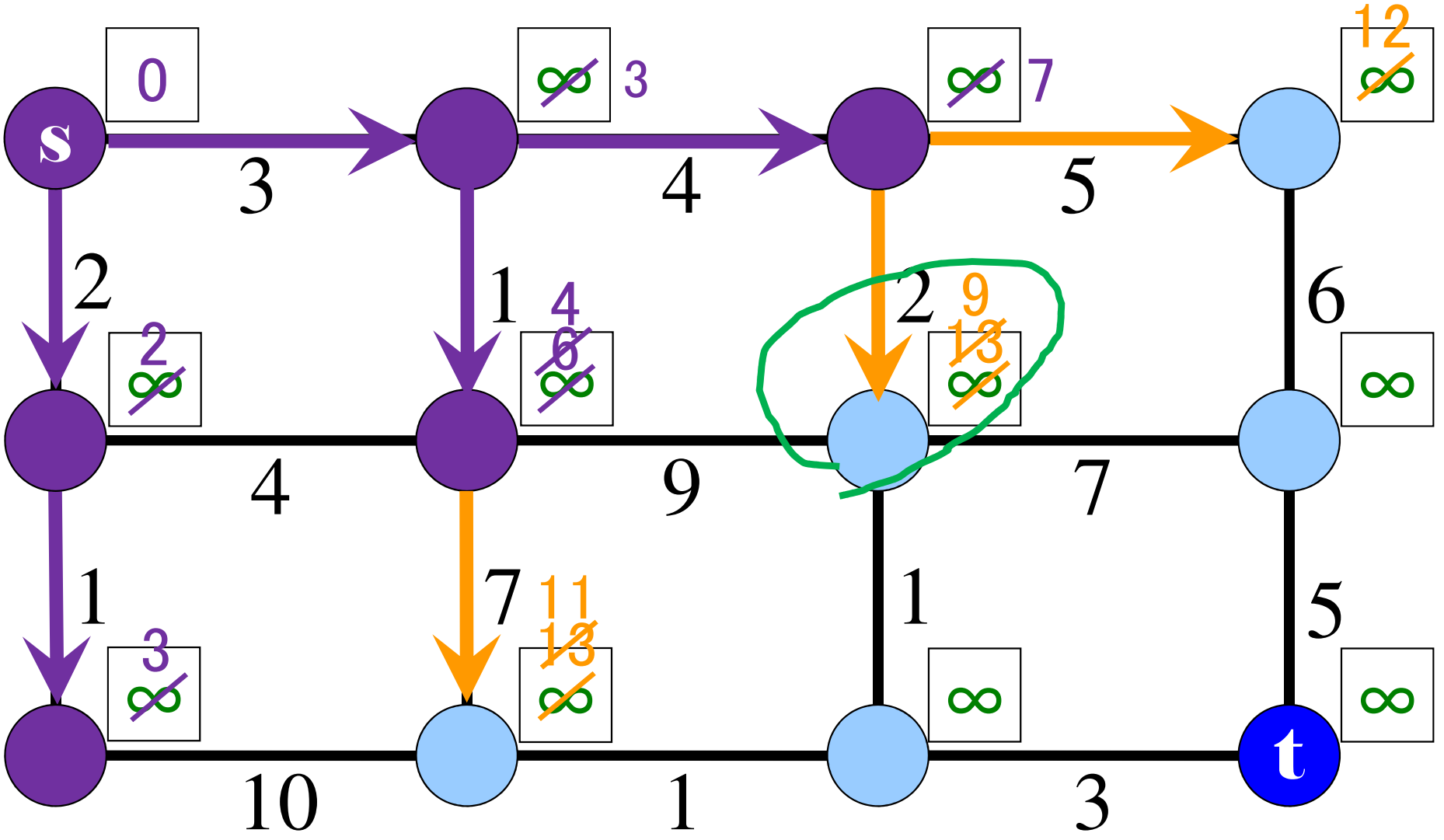
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



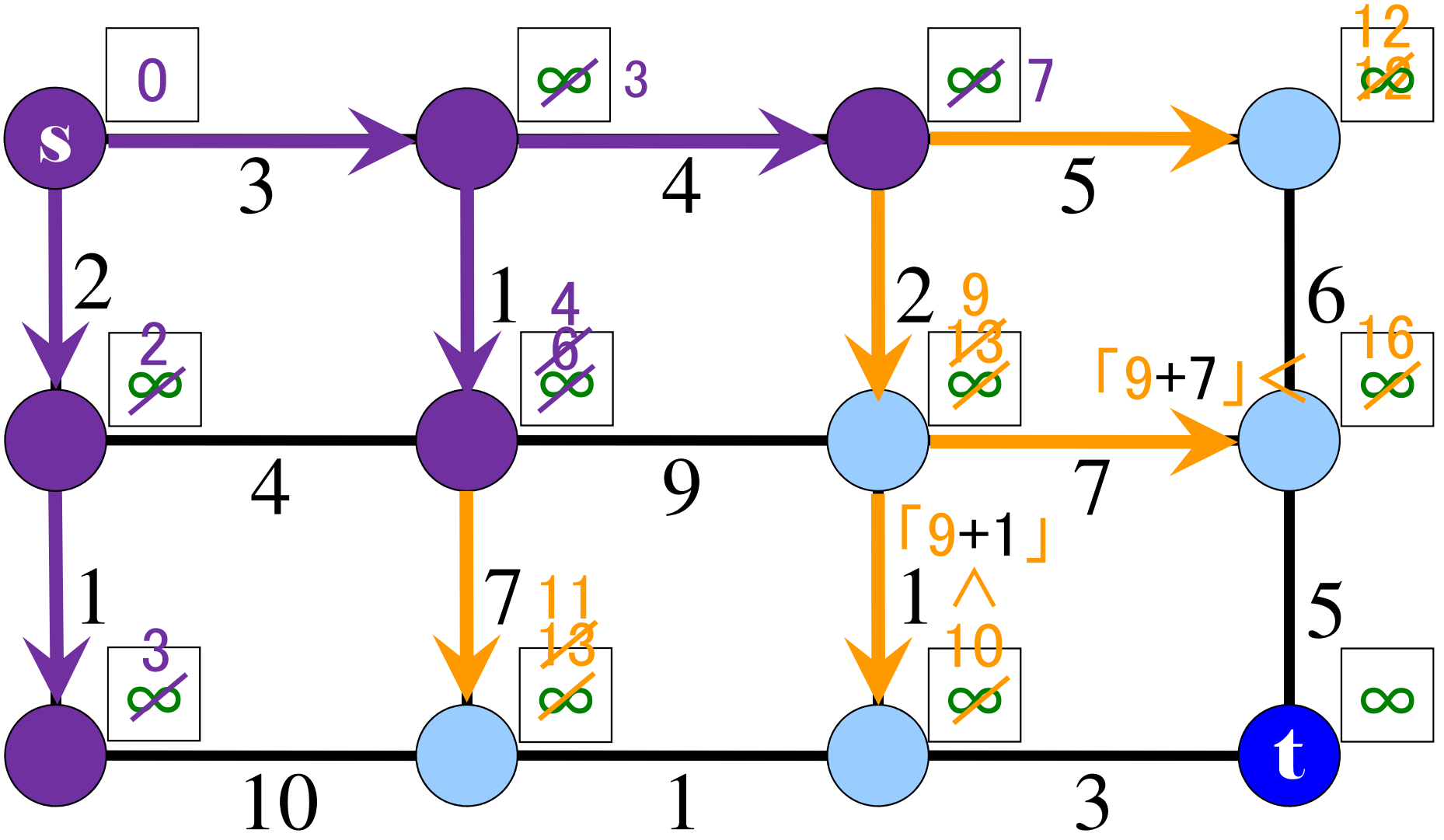
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

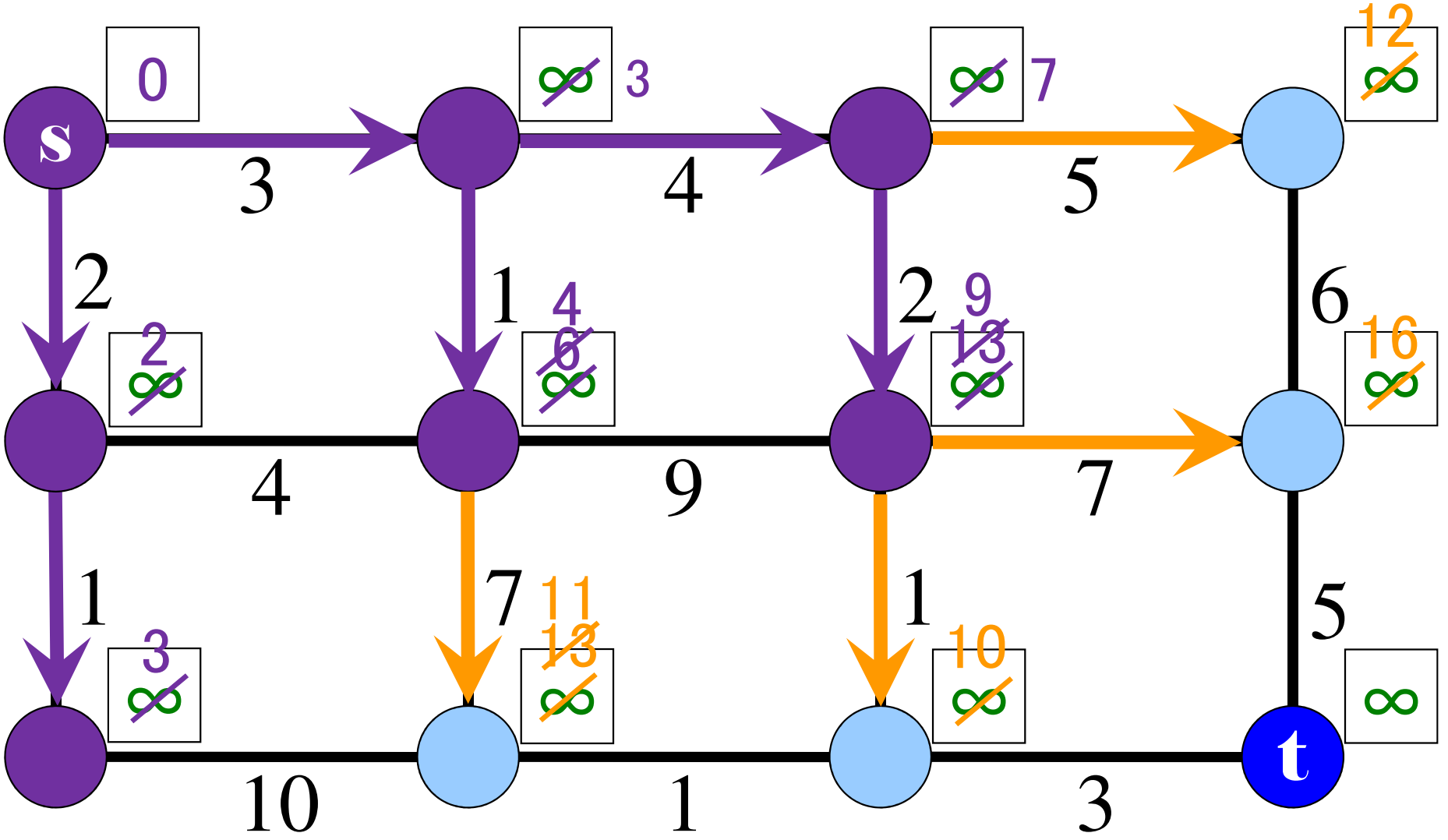


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

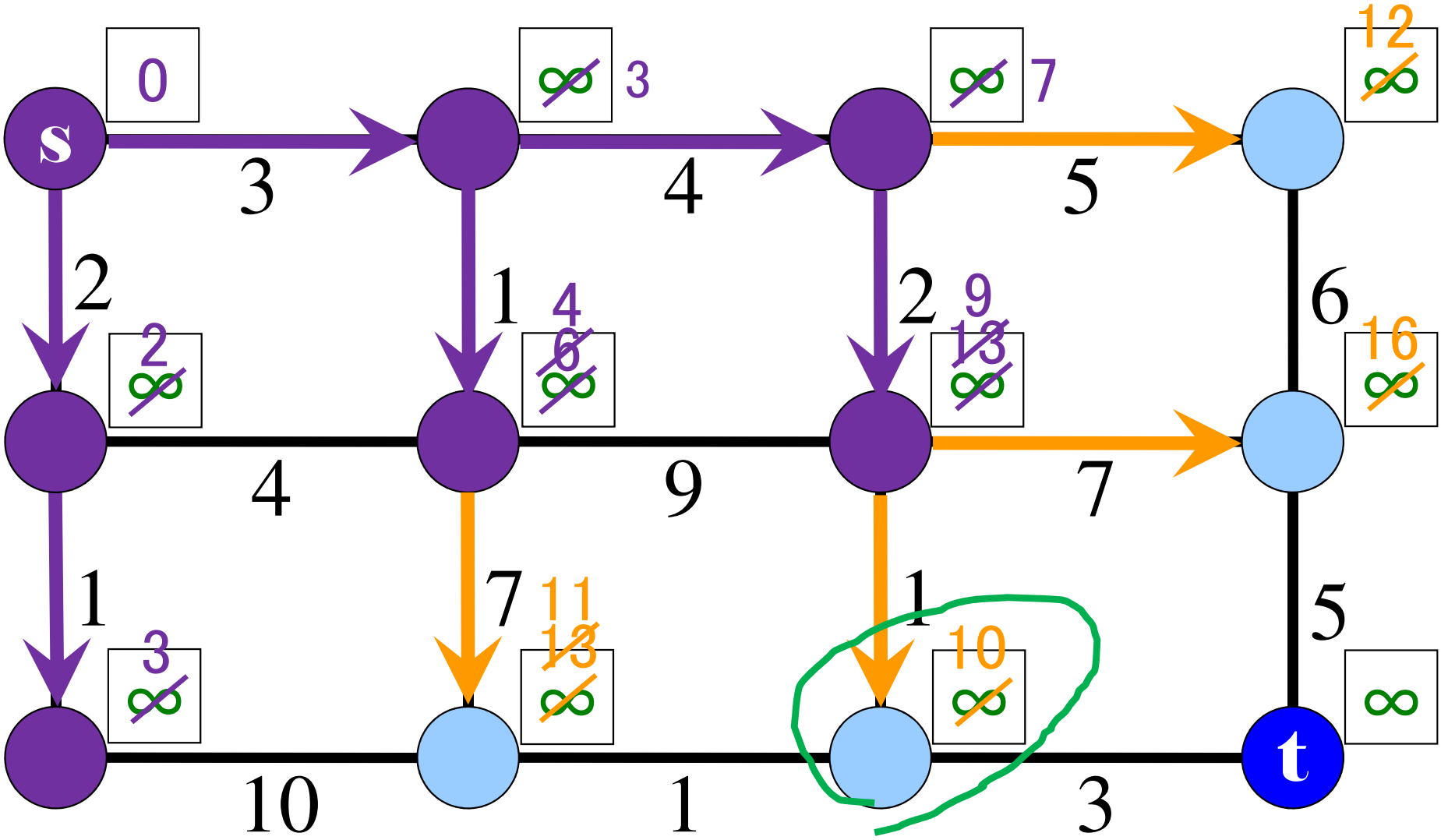
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



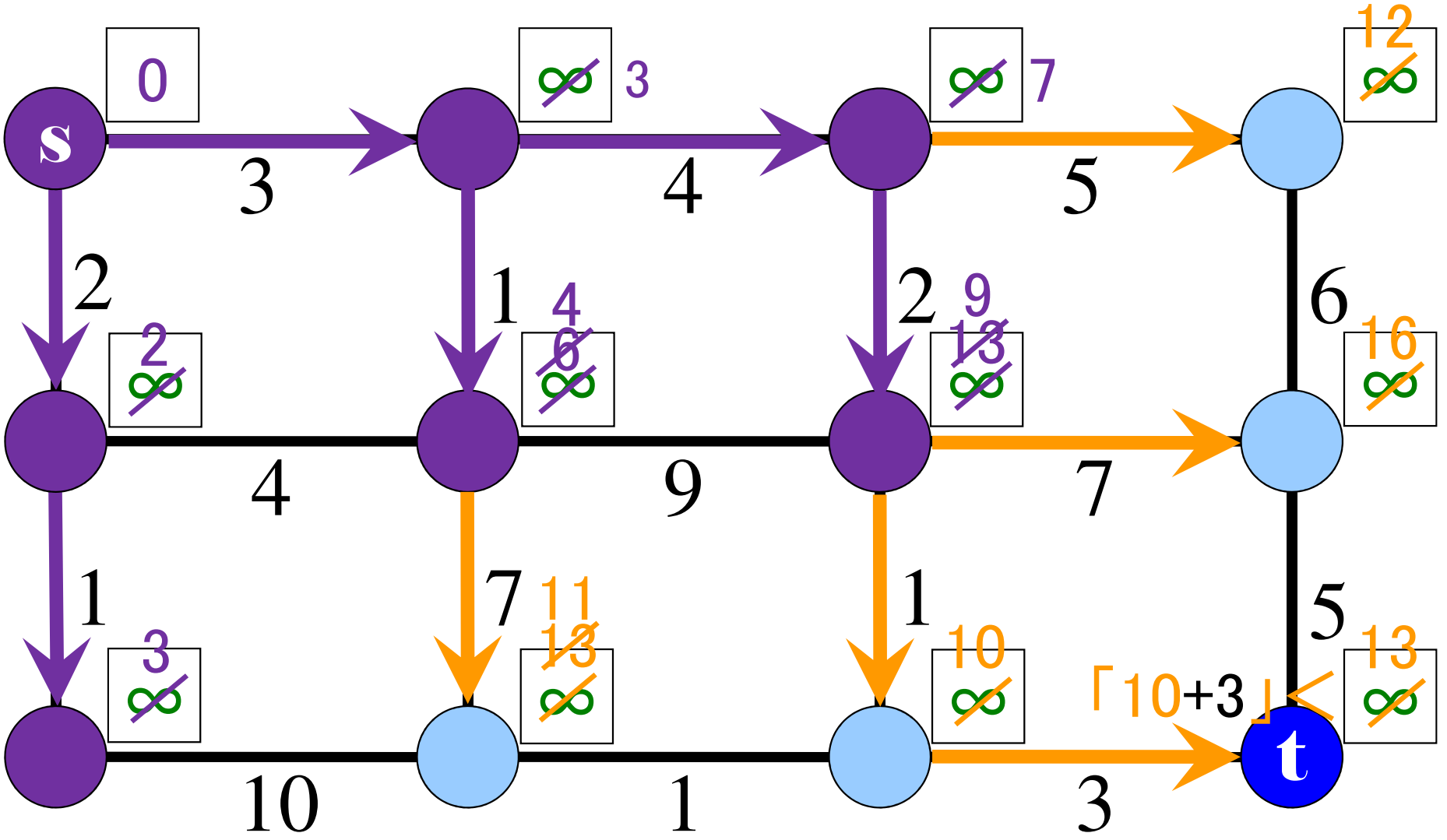
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

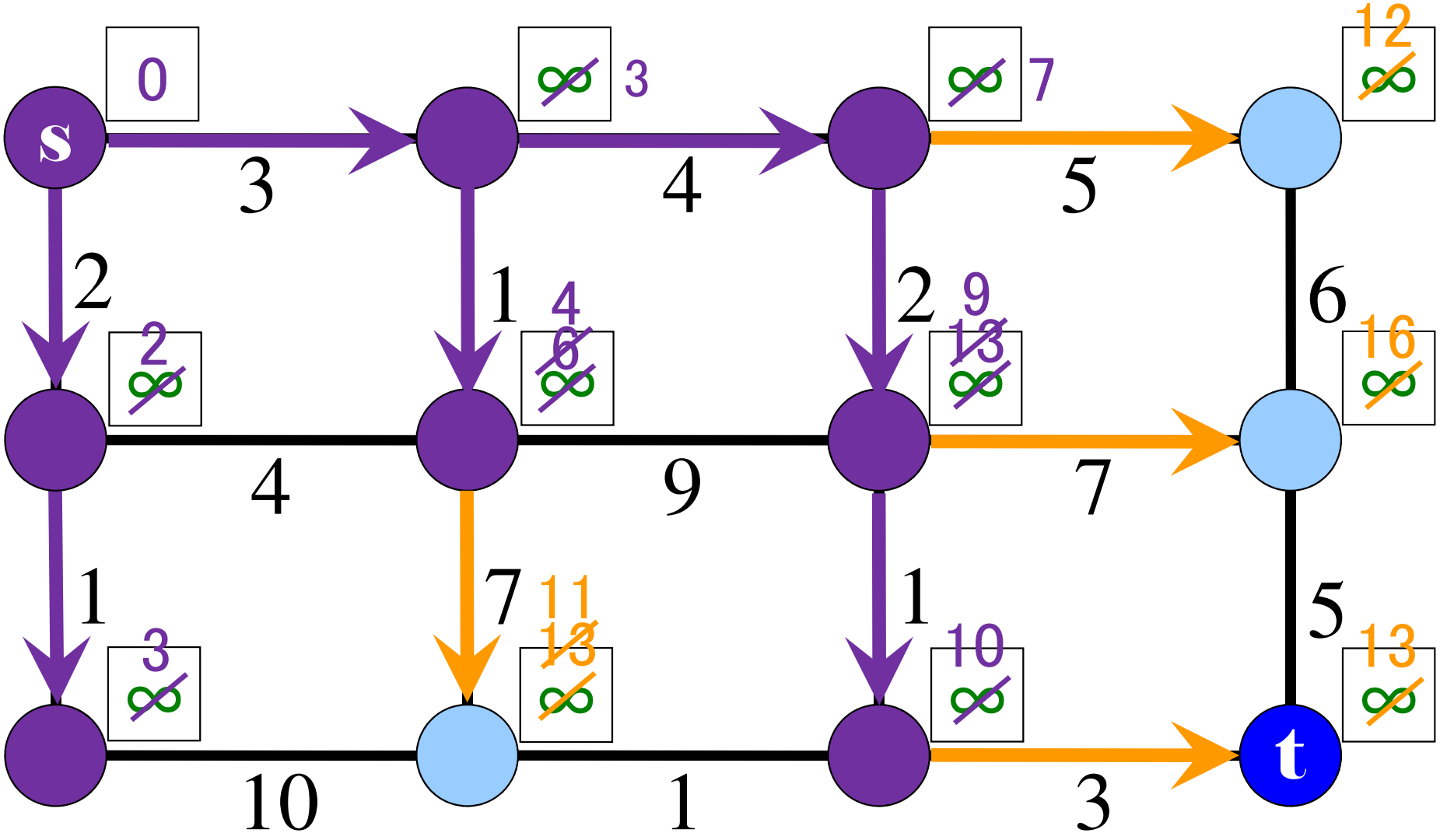


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

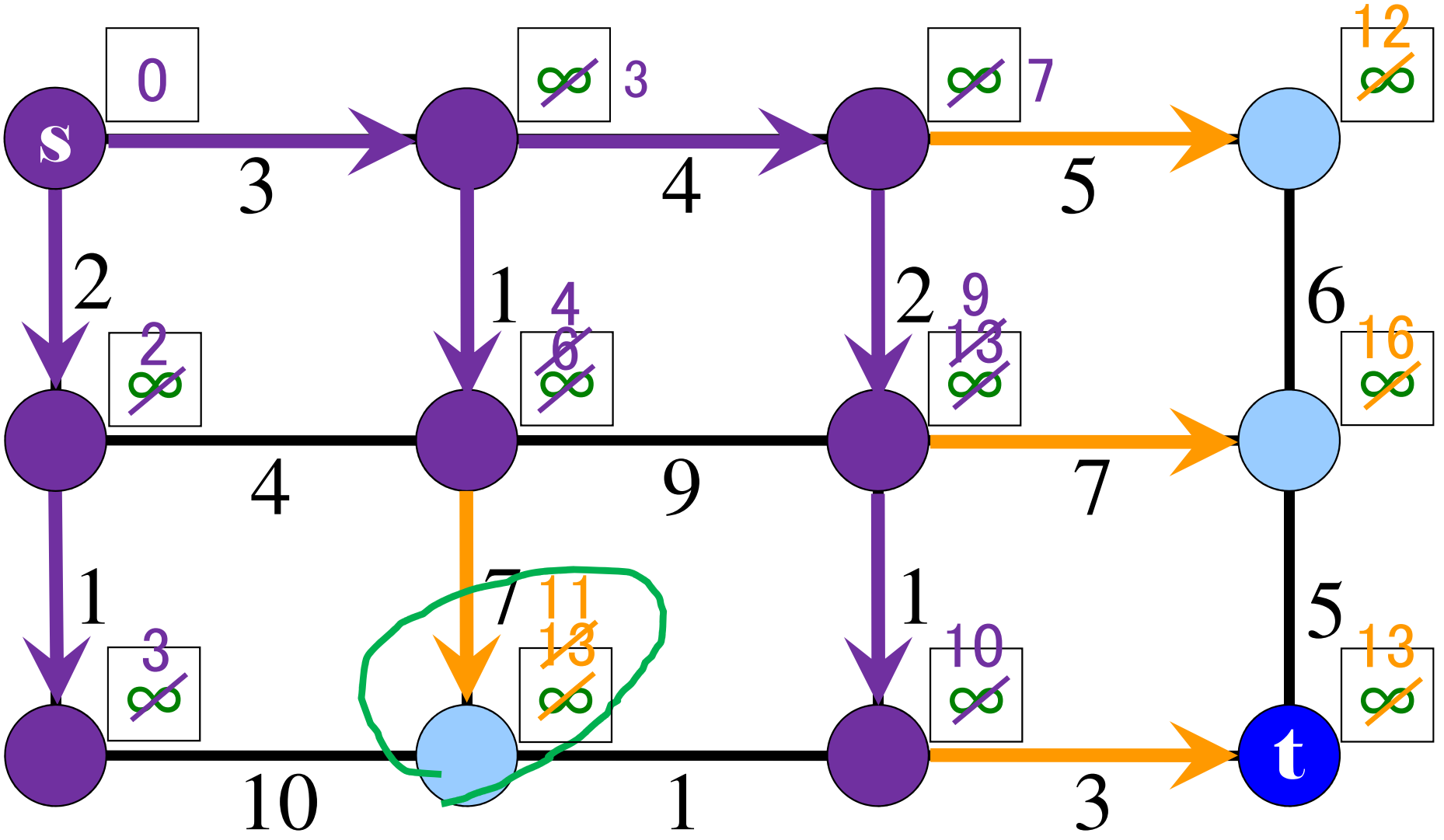
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



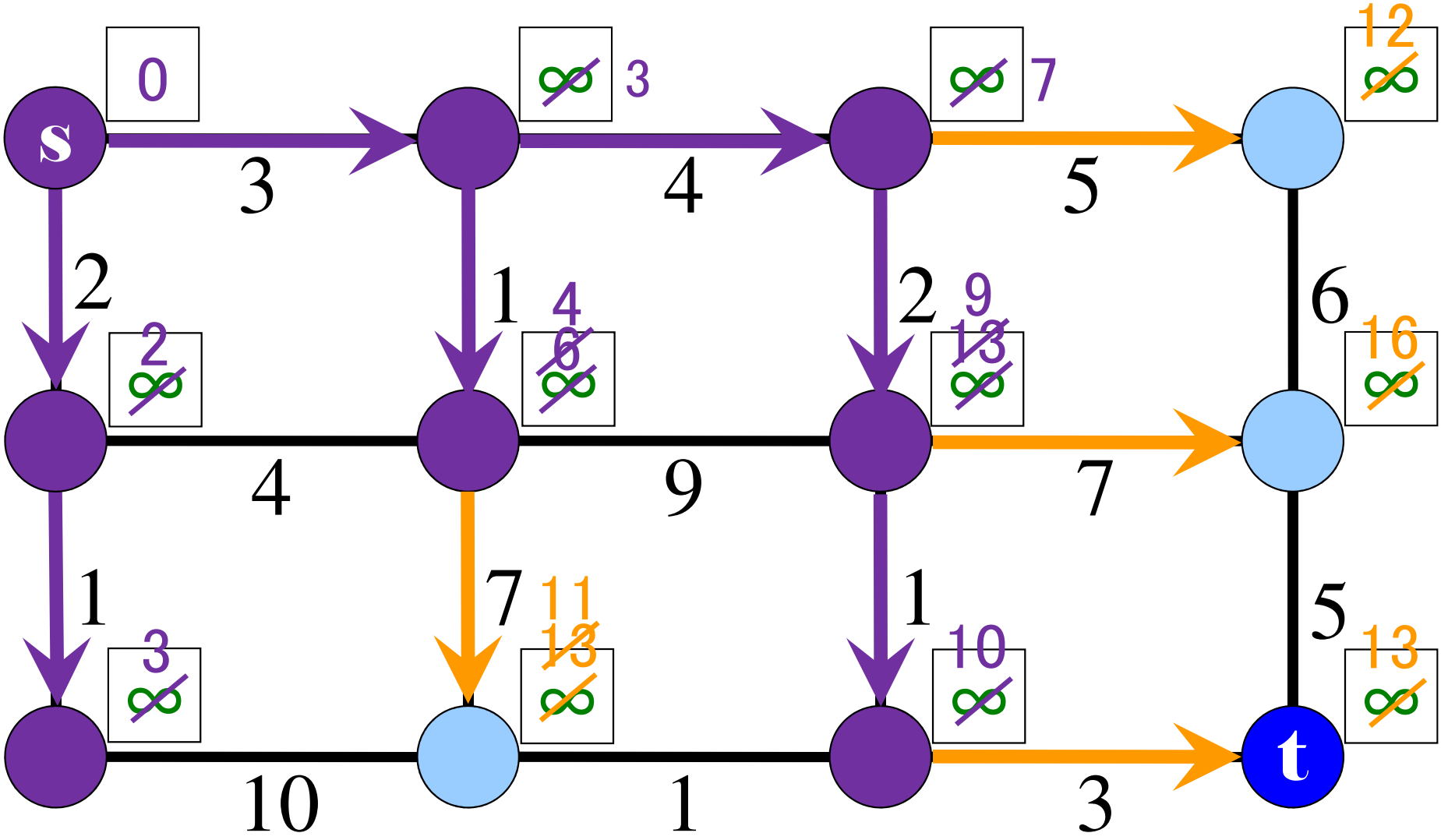
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

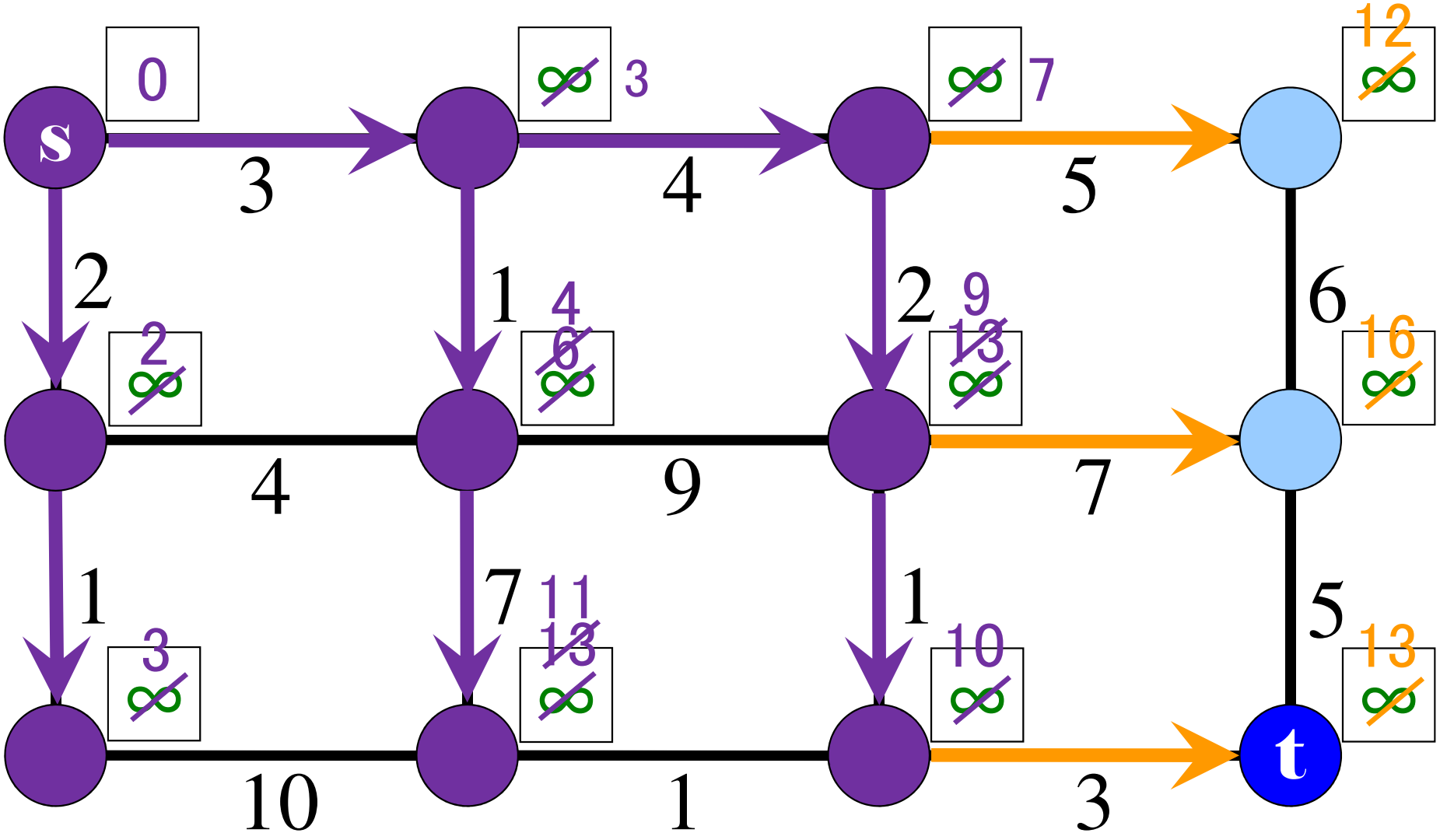


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

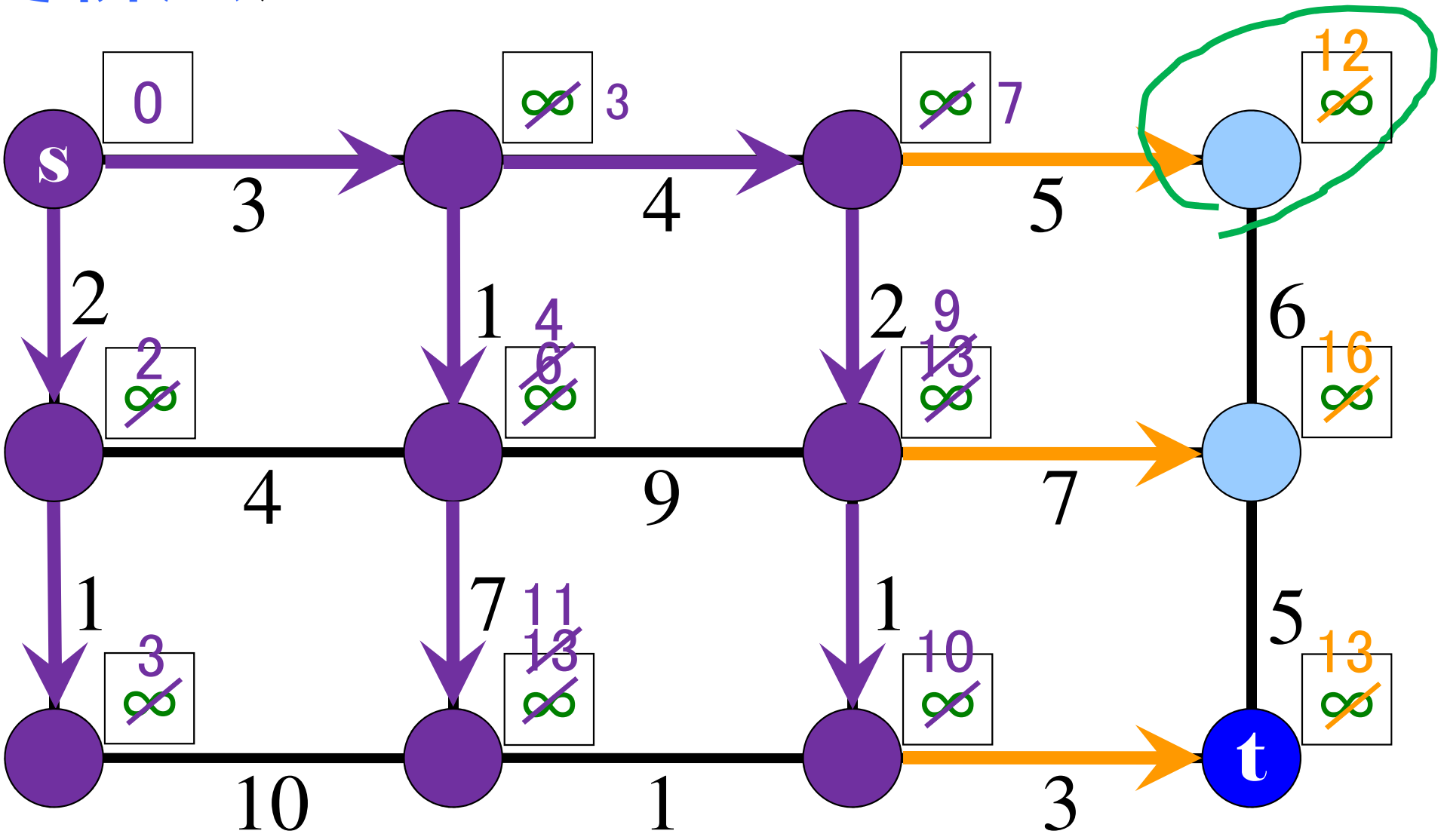
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



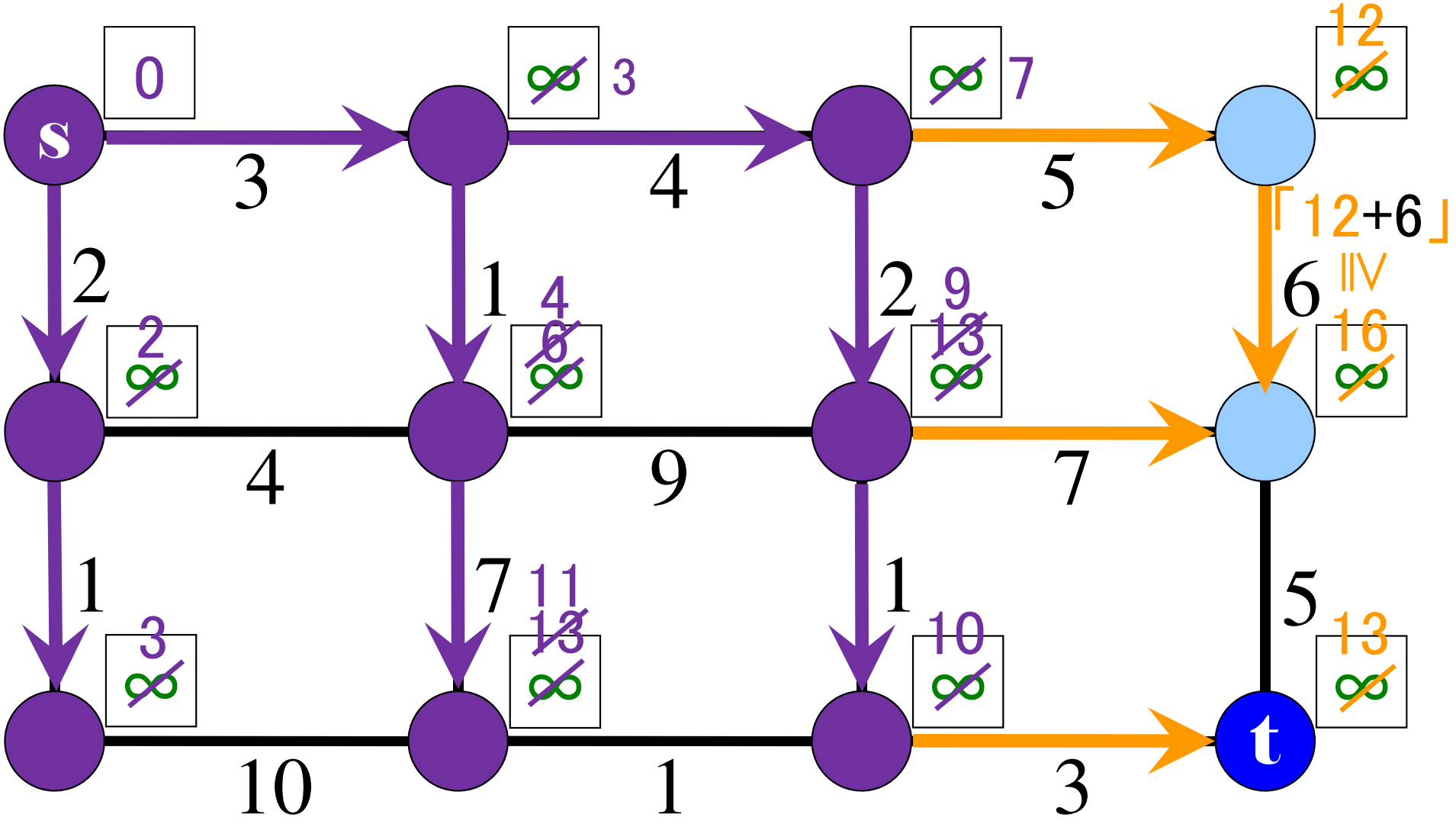
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

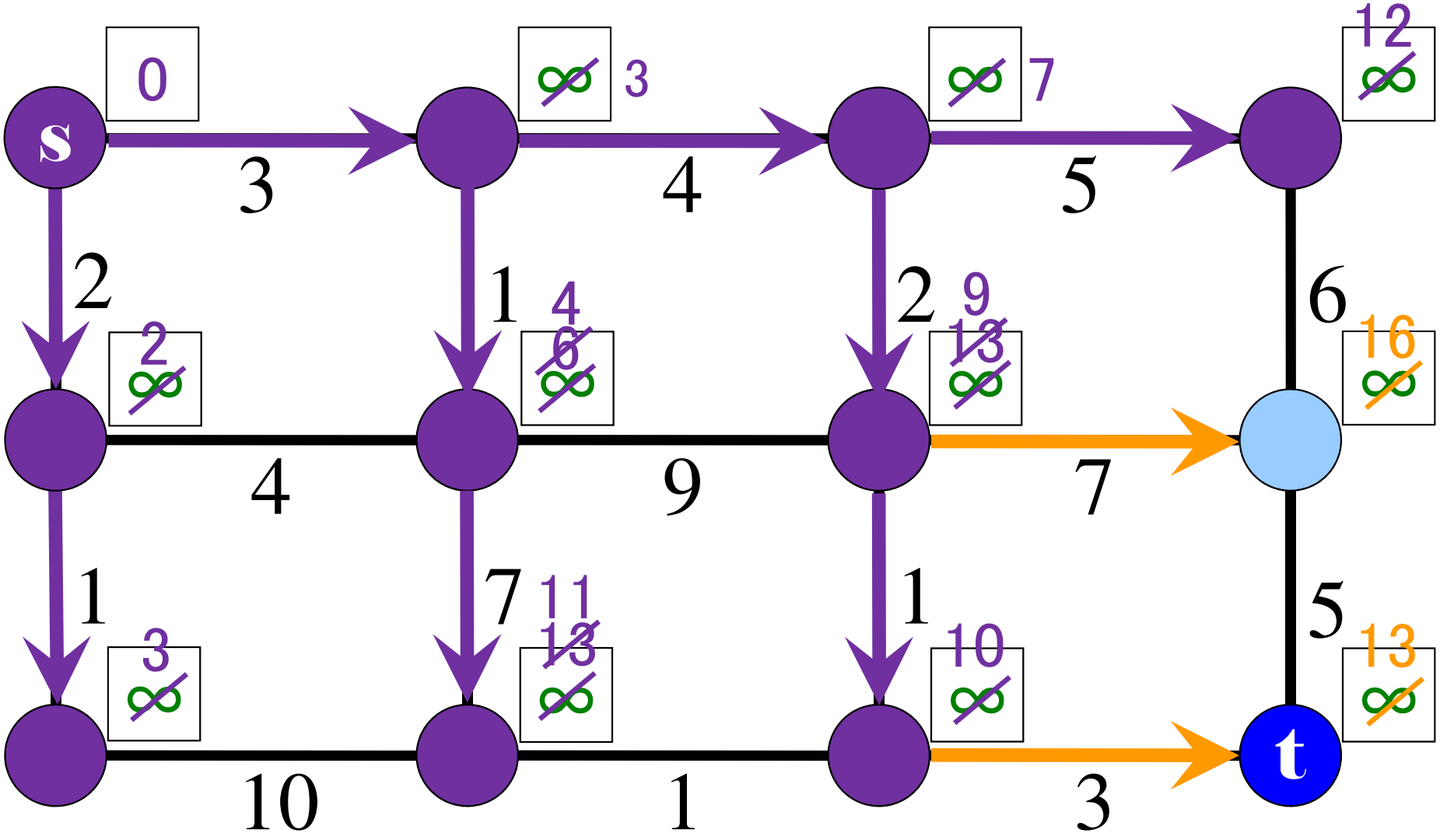


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

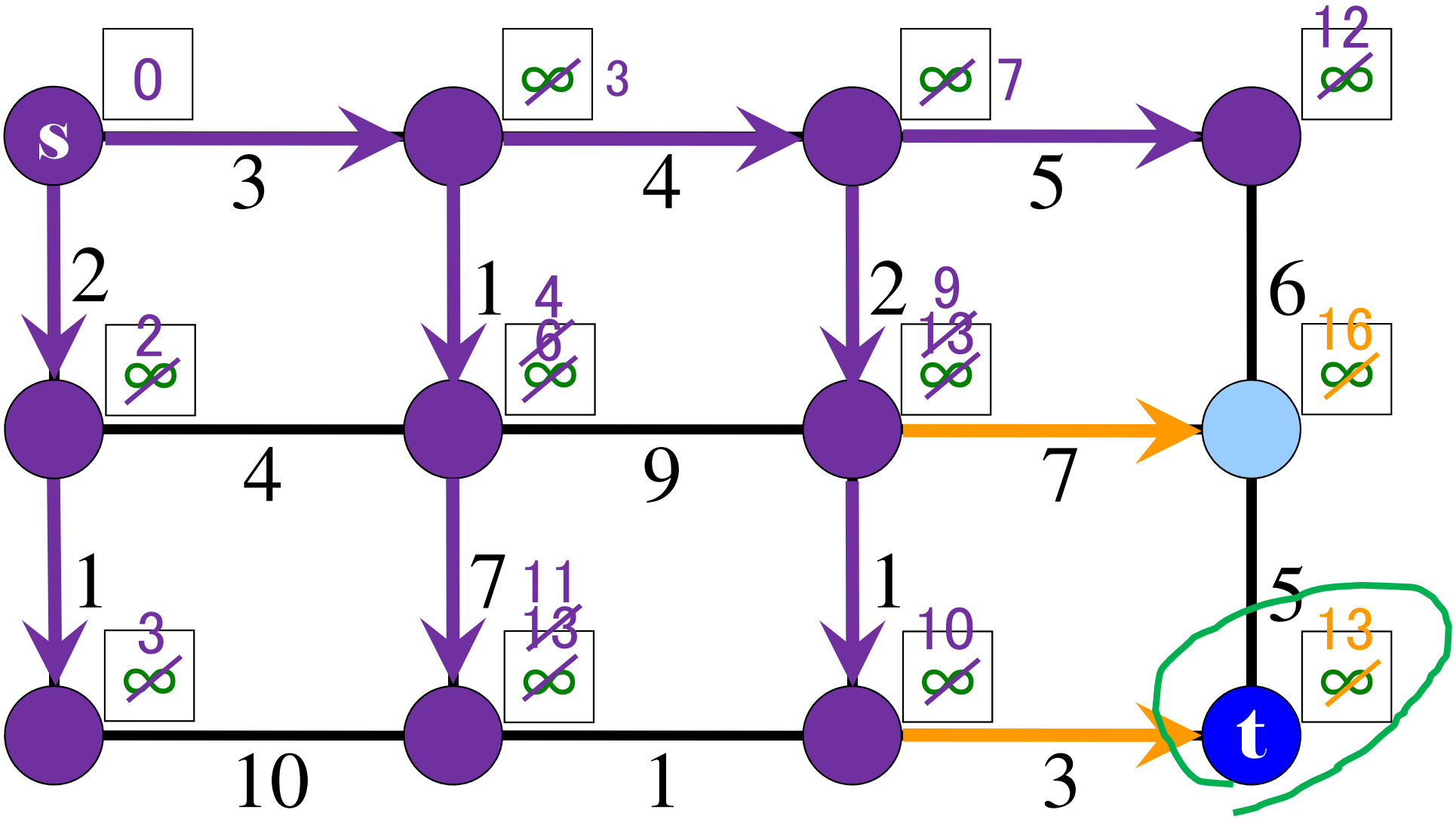
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



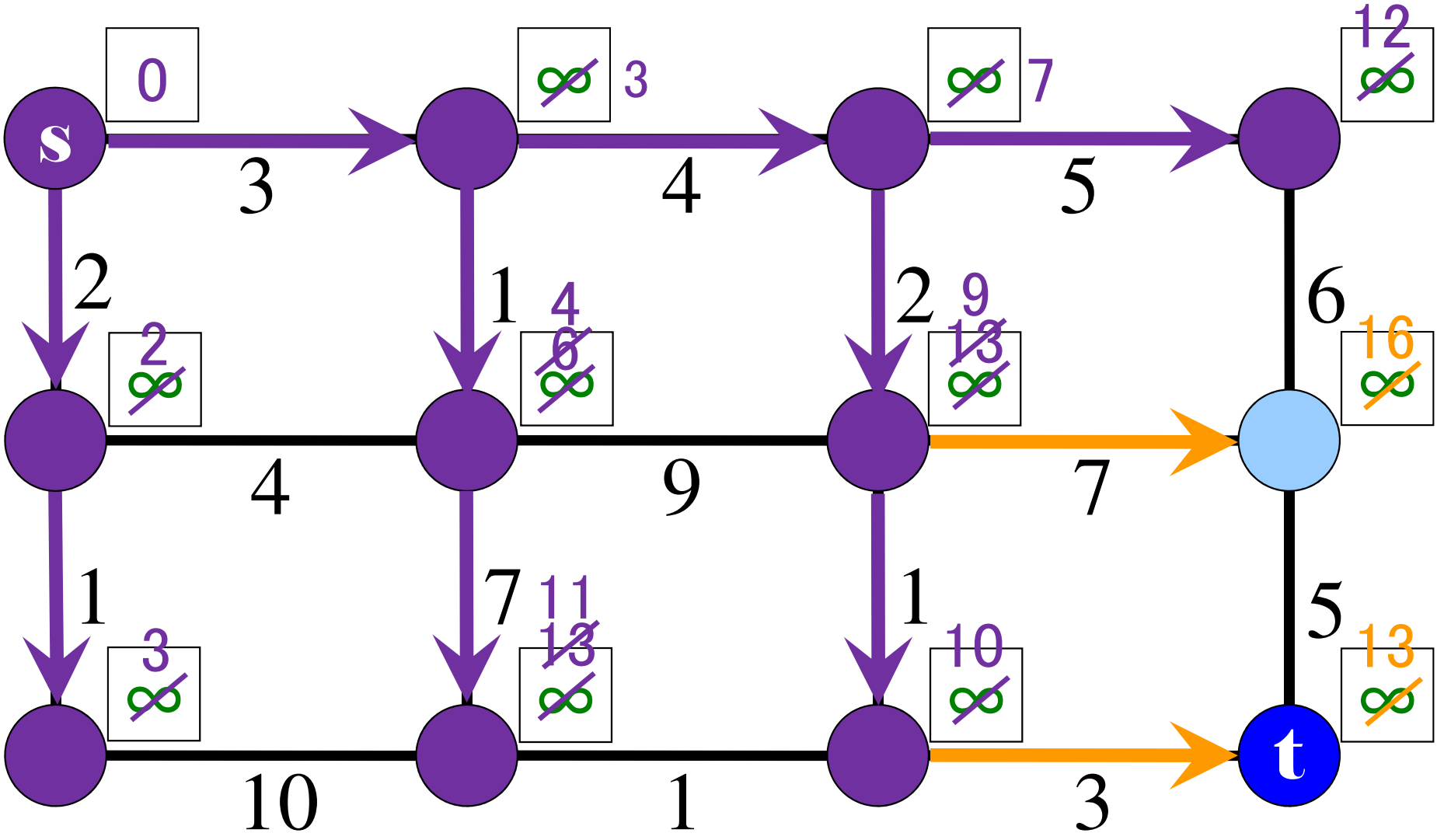
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

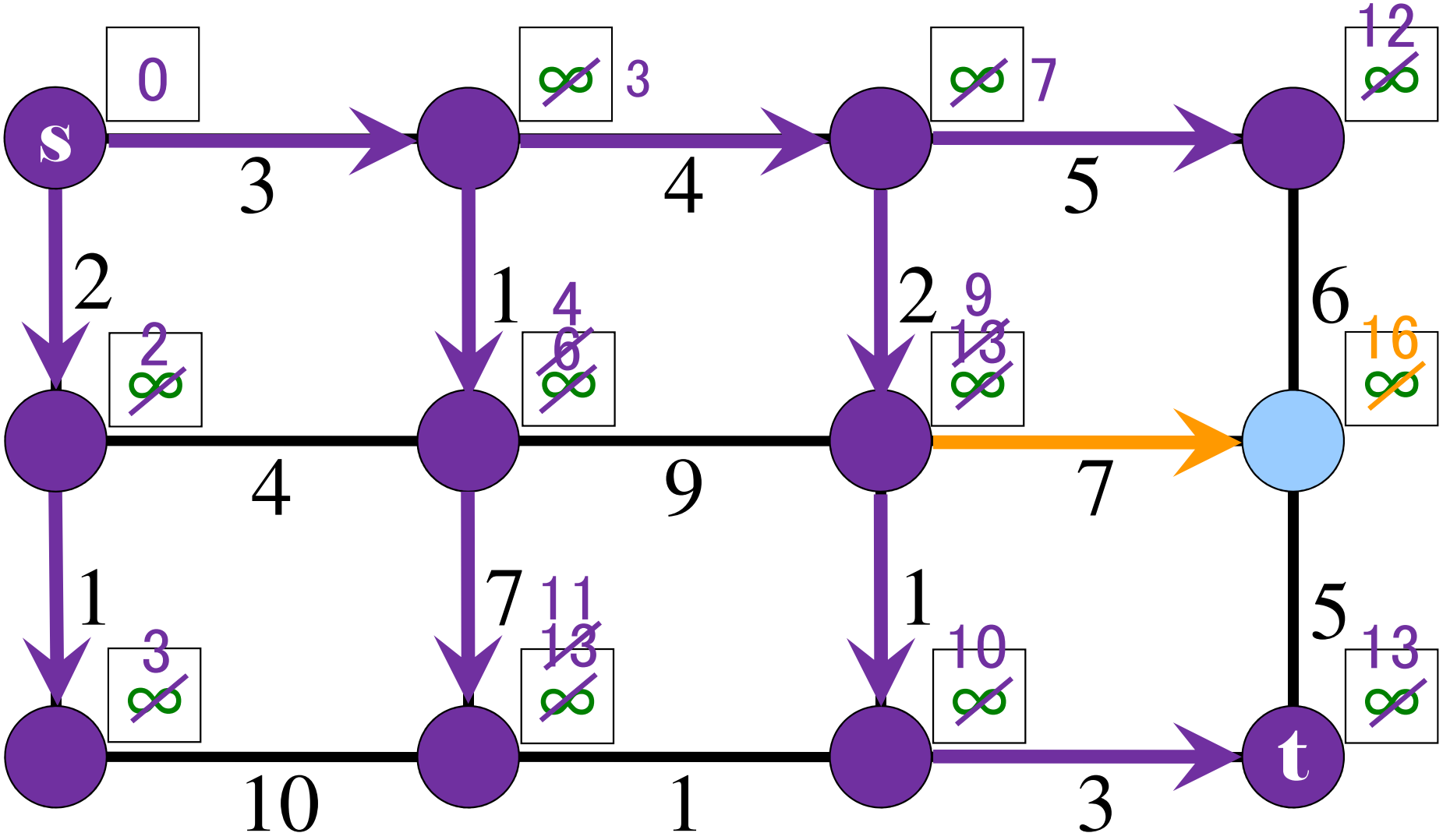


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

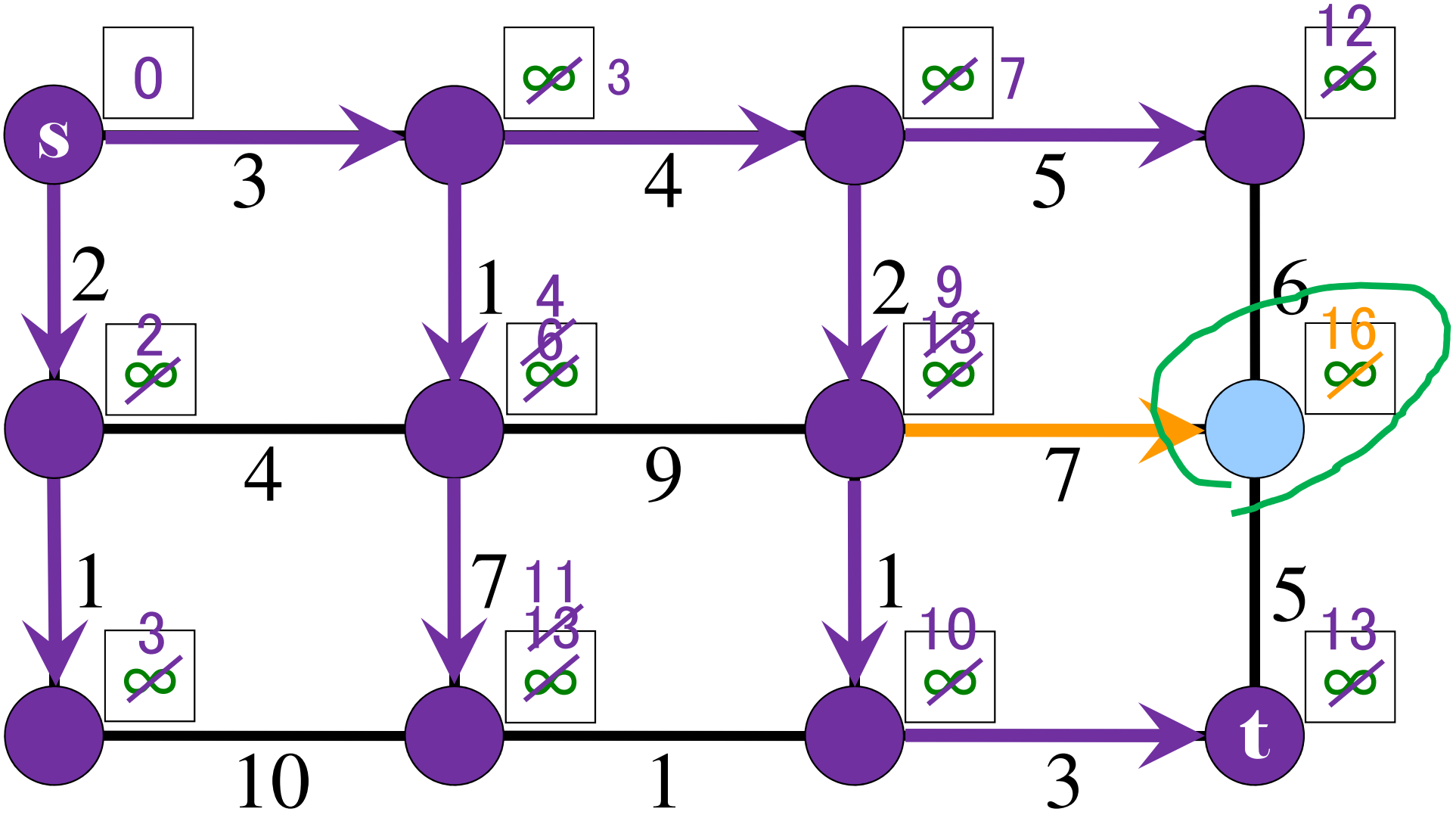
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



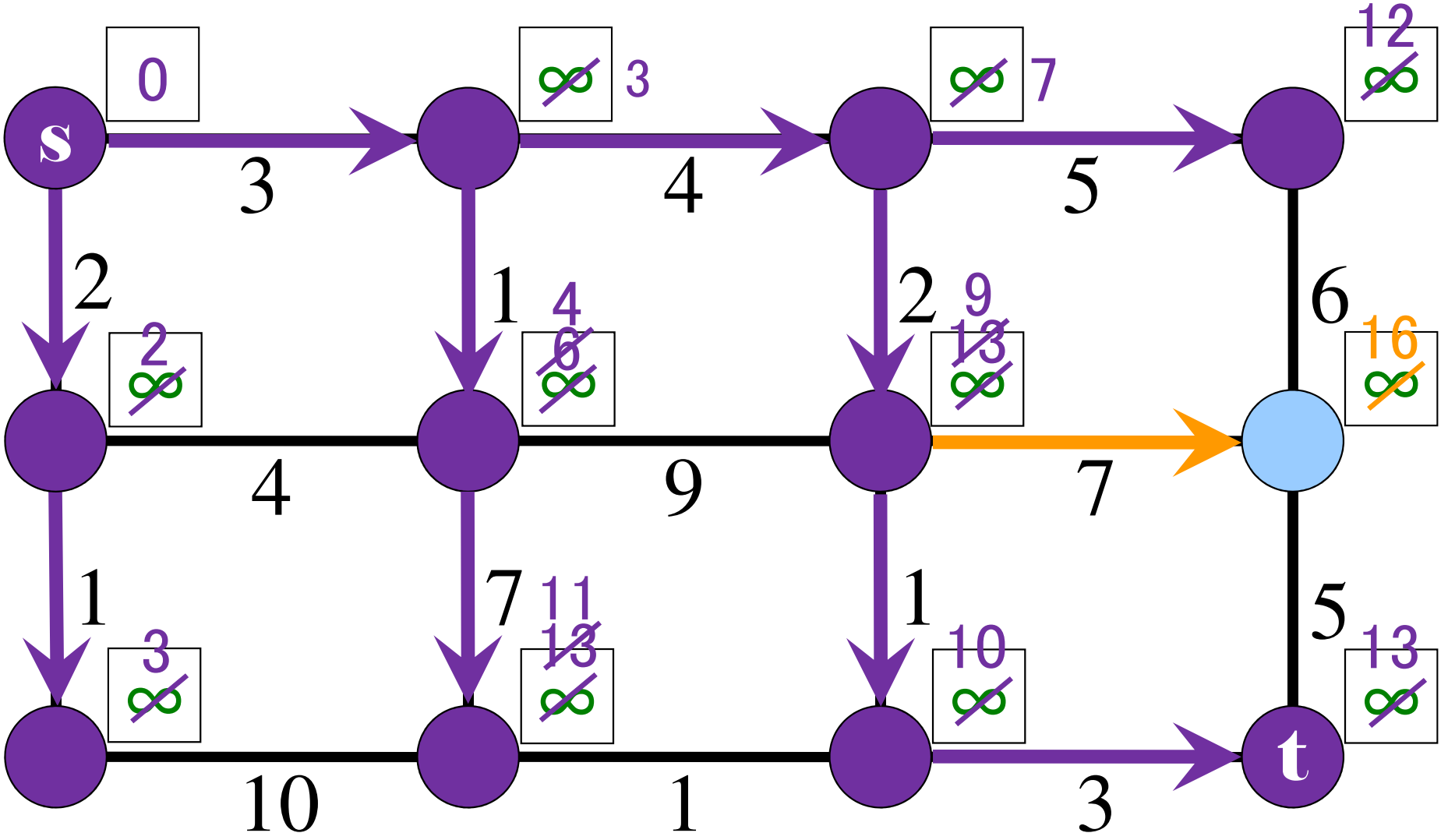
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

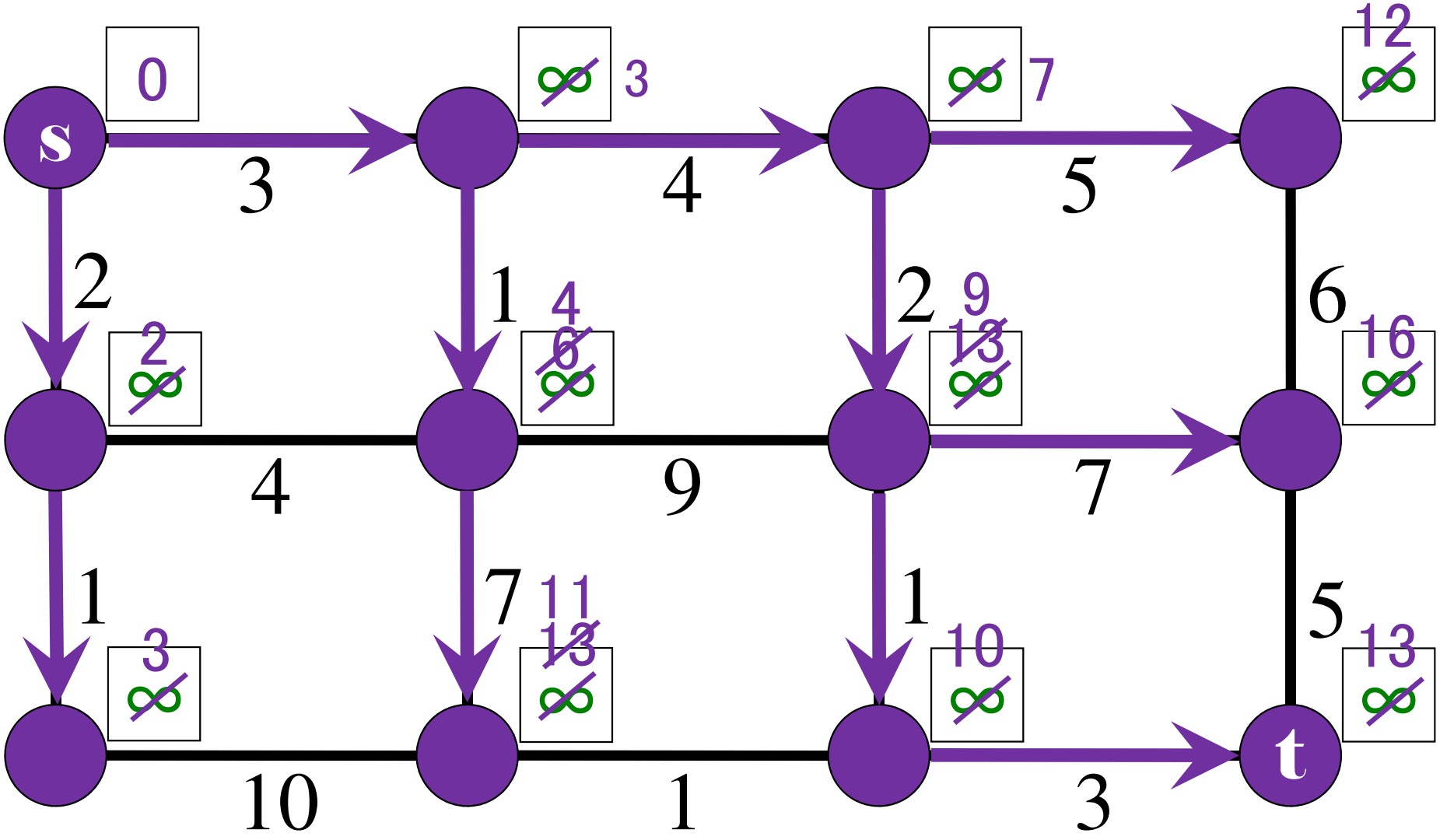


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

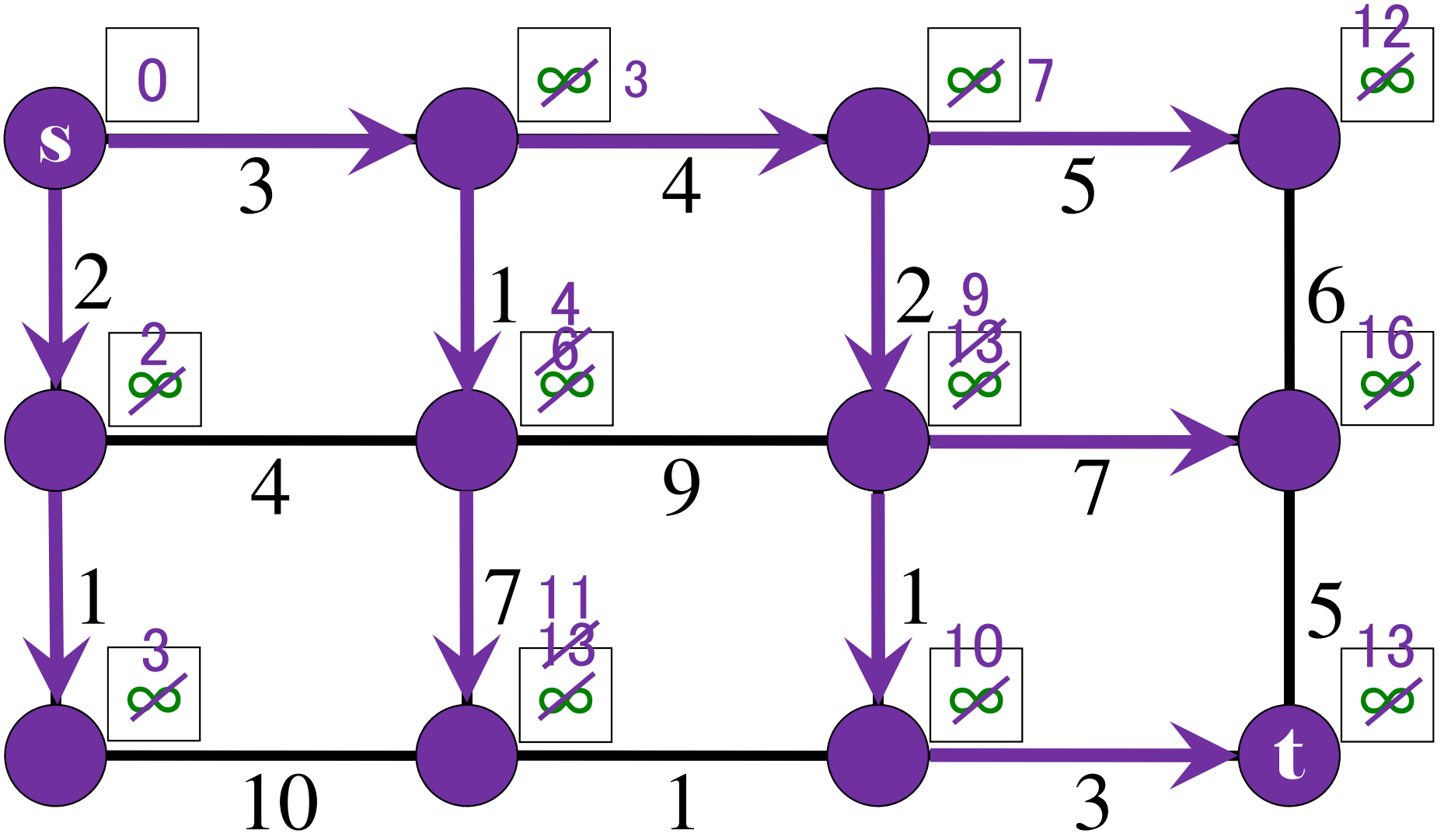
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



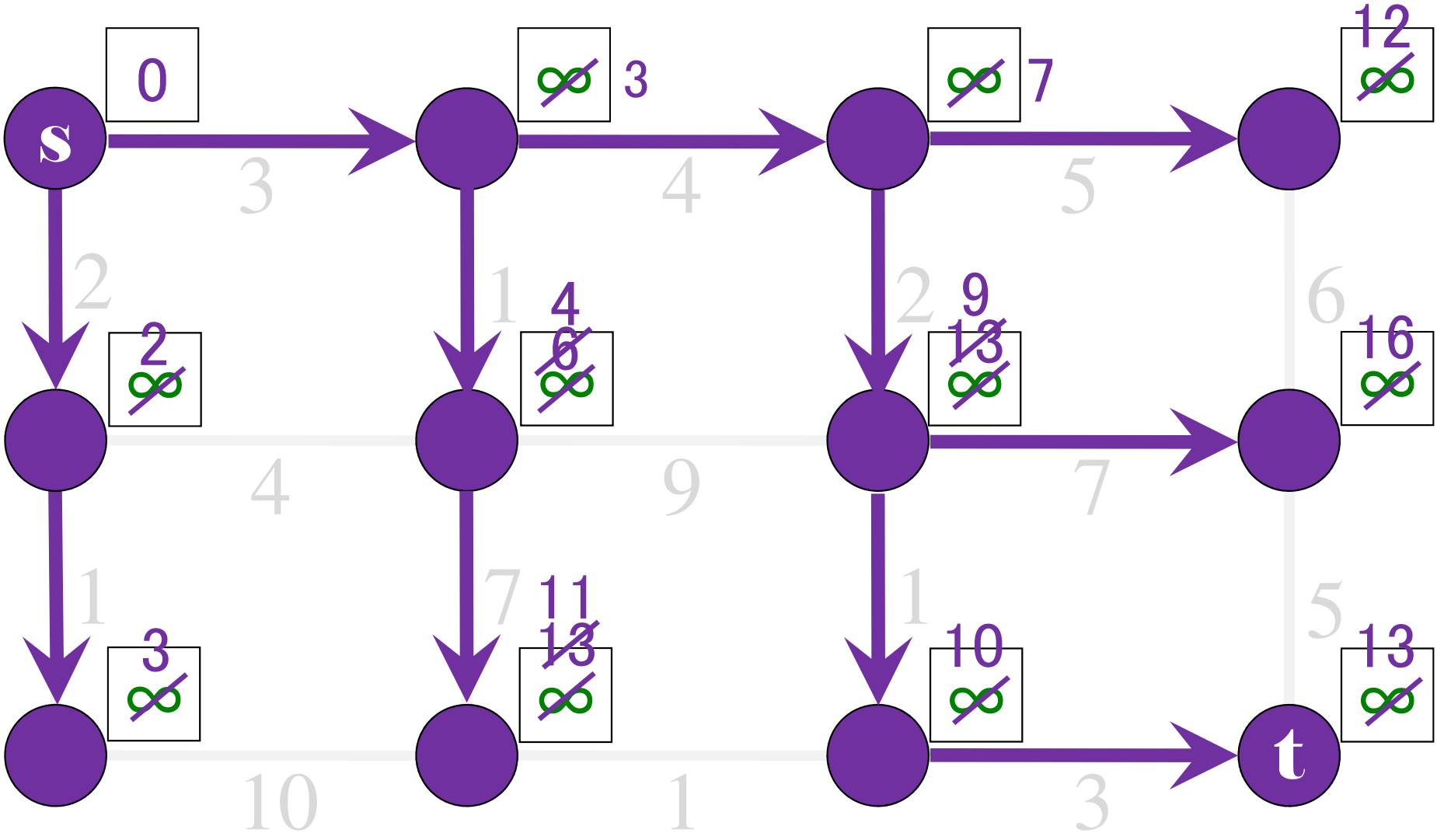
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



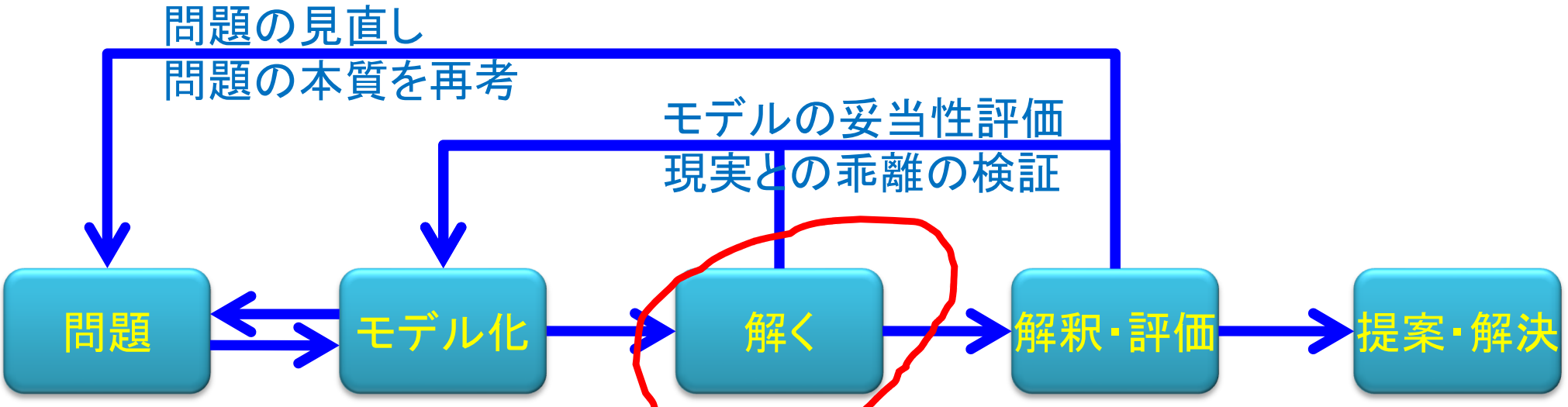
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

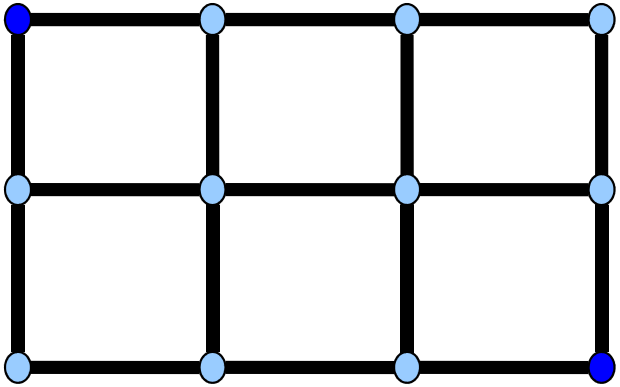
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

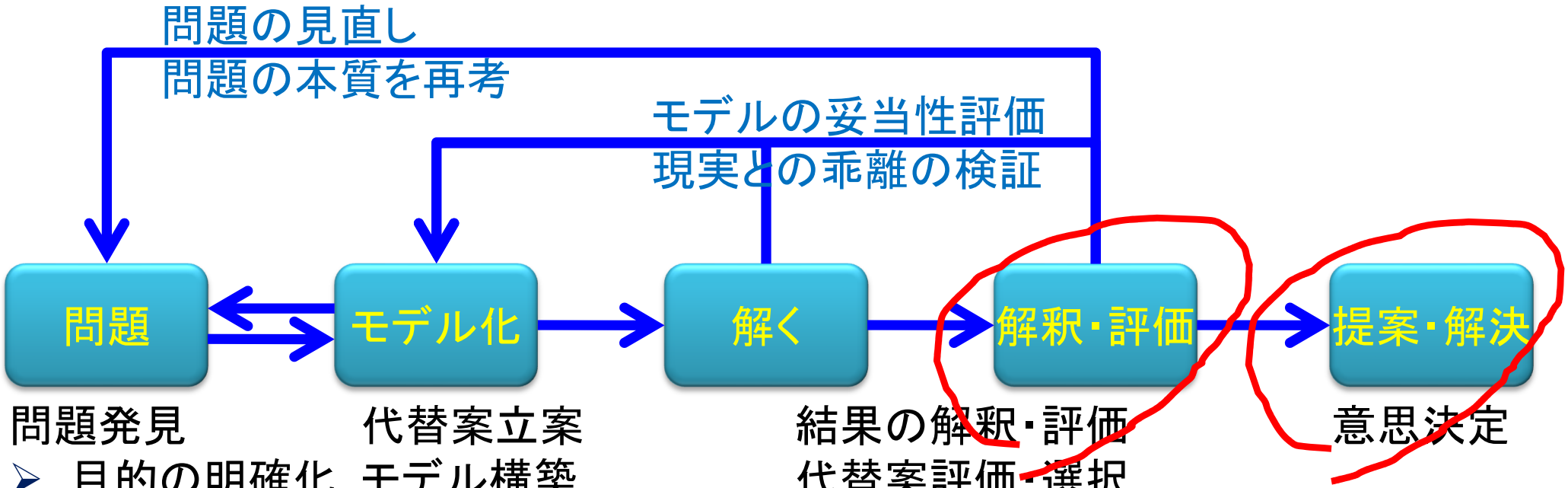
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

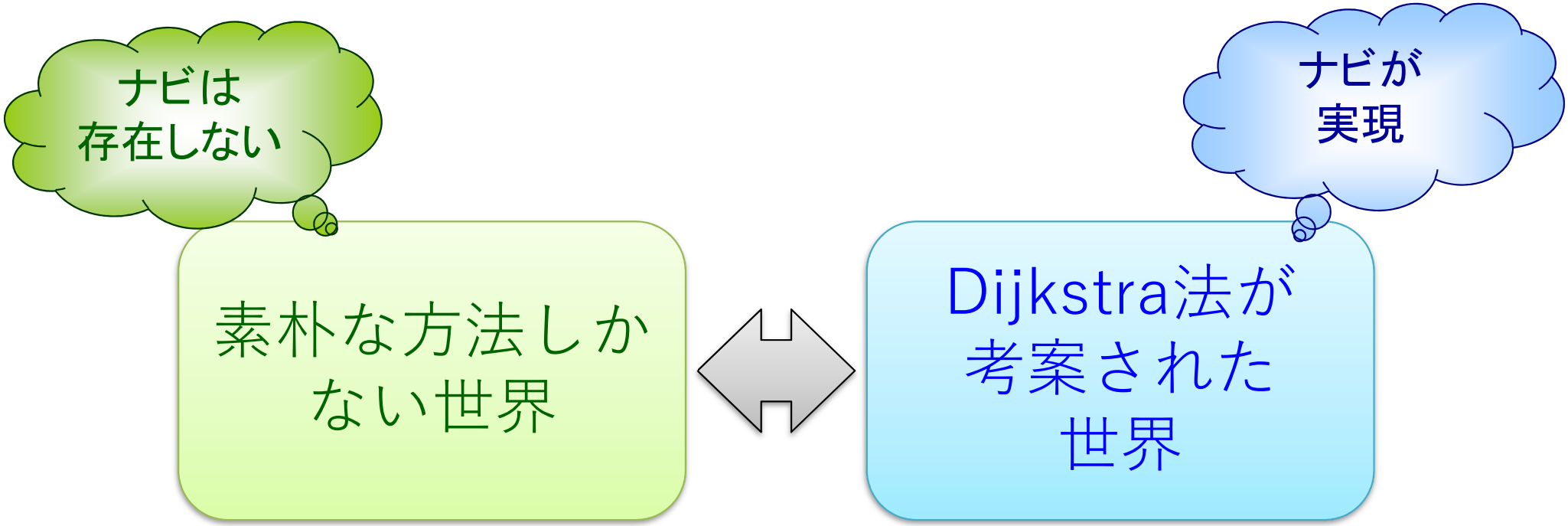
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

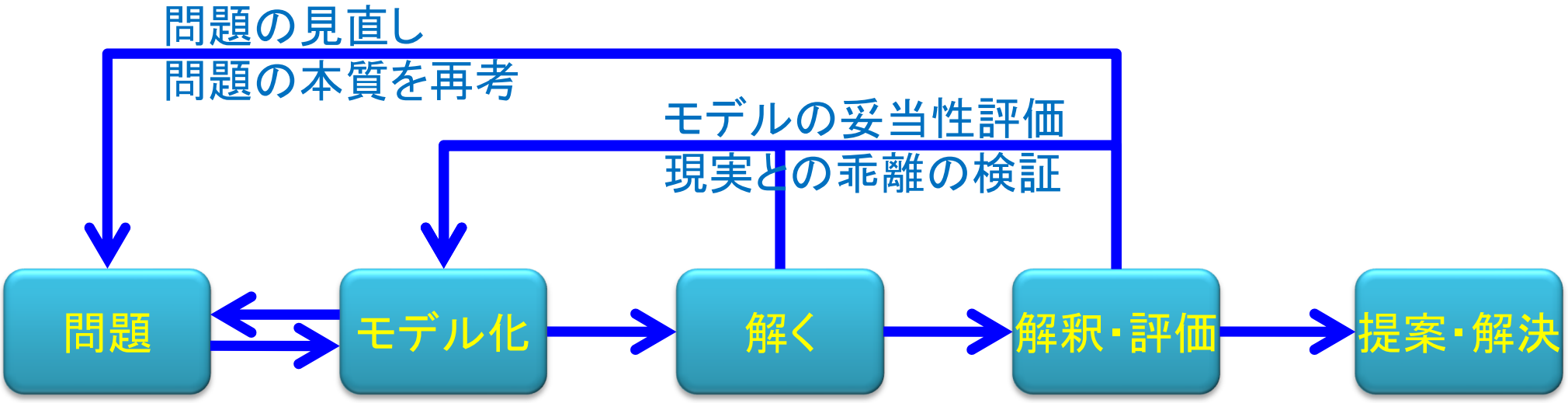
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



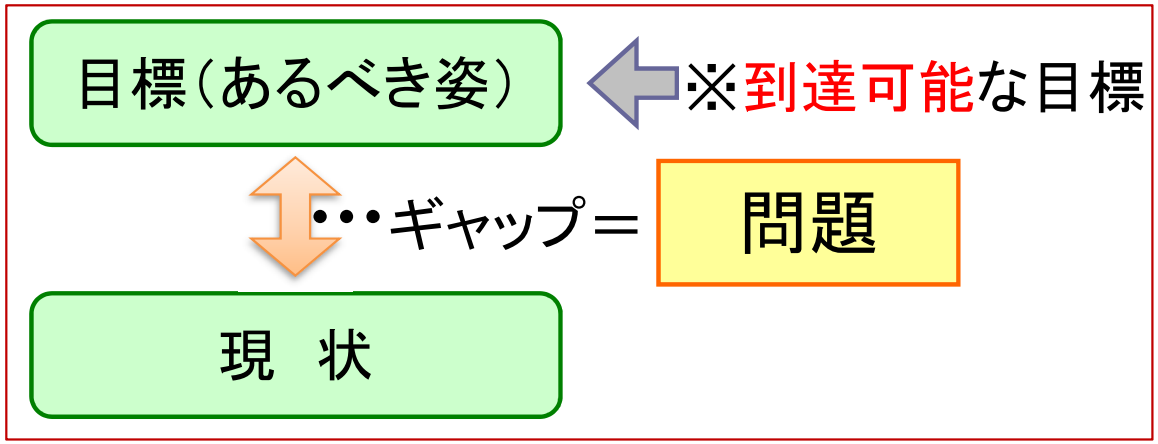
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

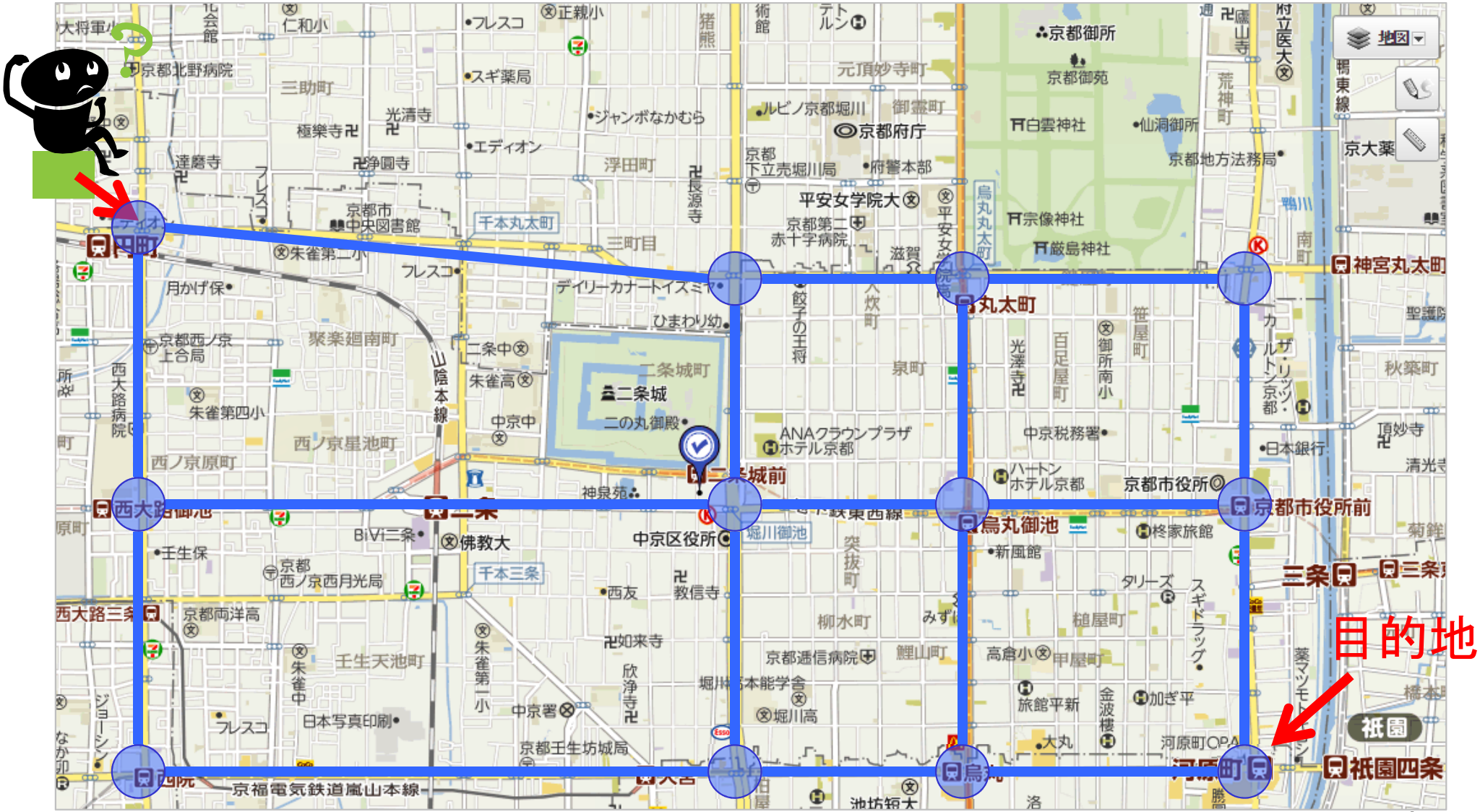
意思決定

問題の定義



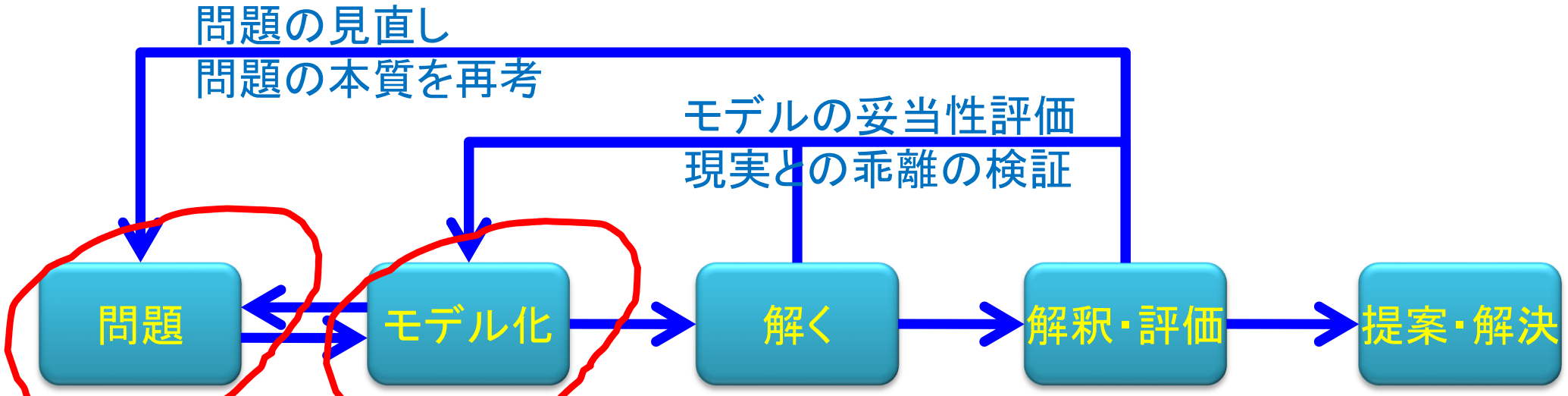
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

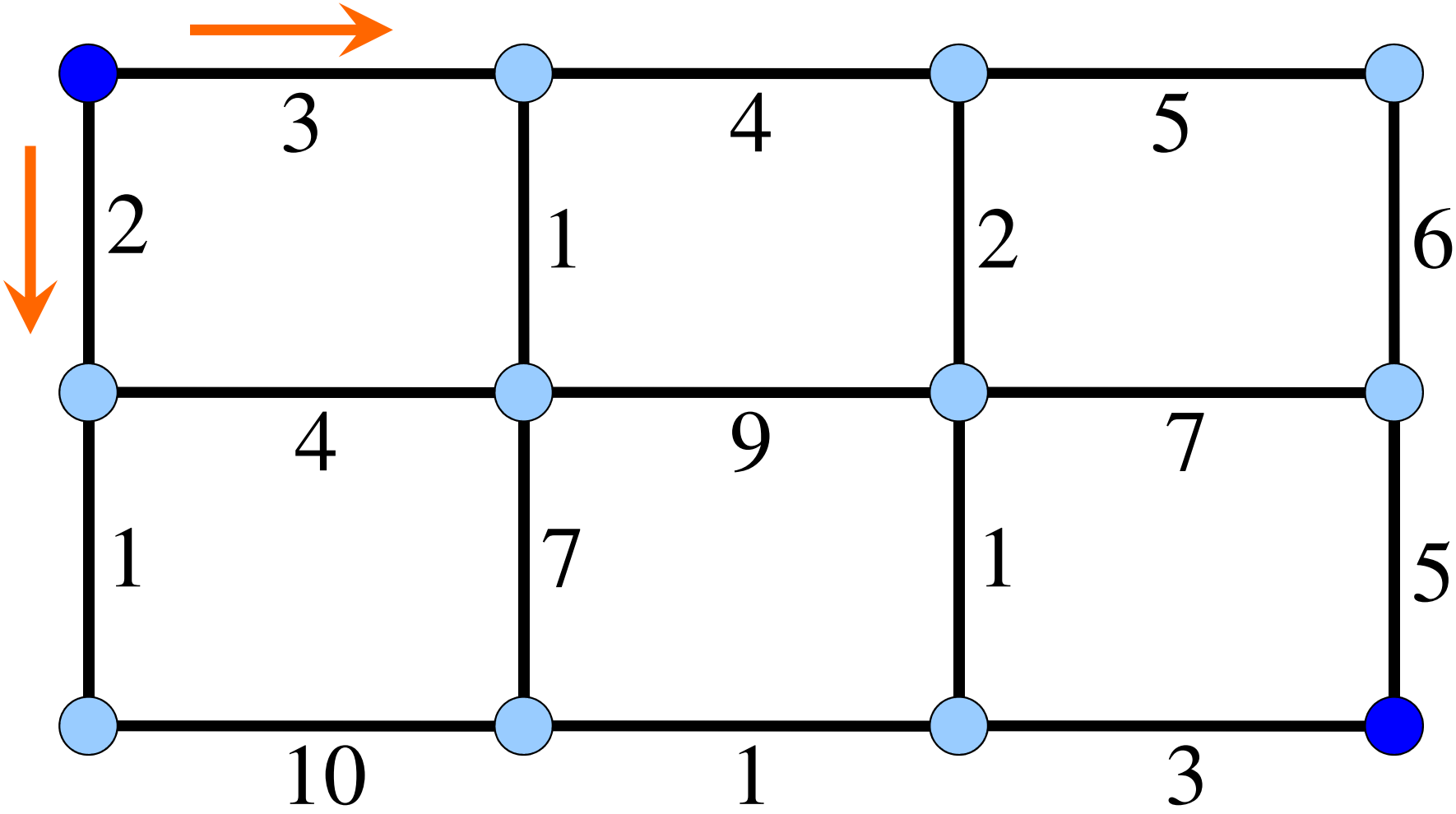
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

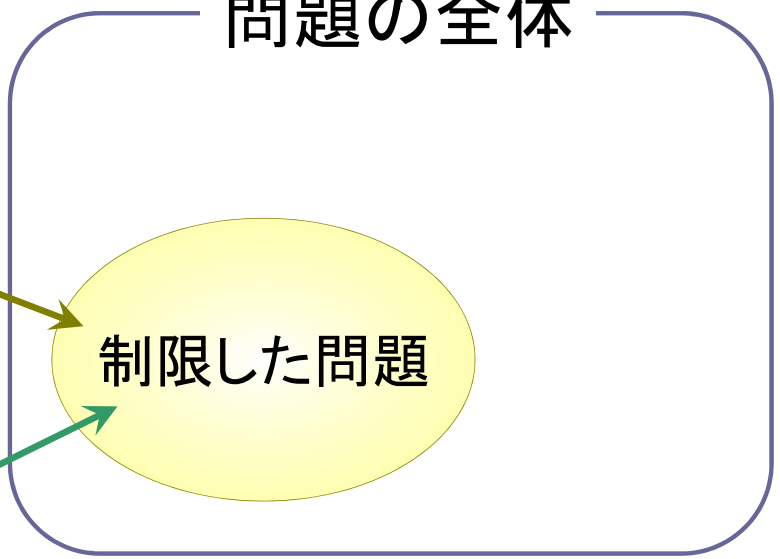


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

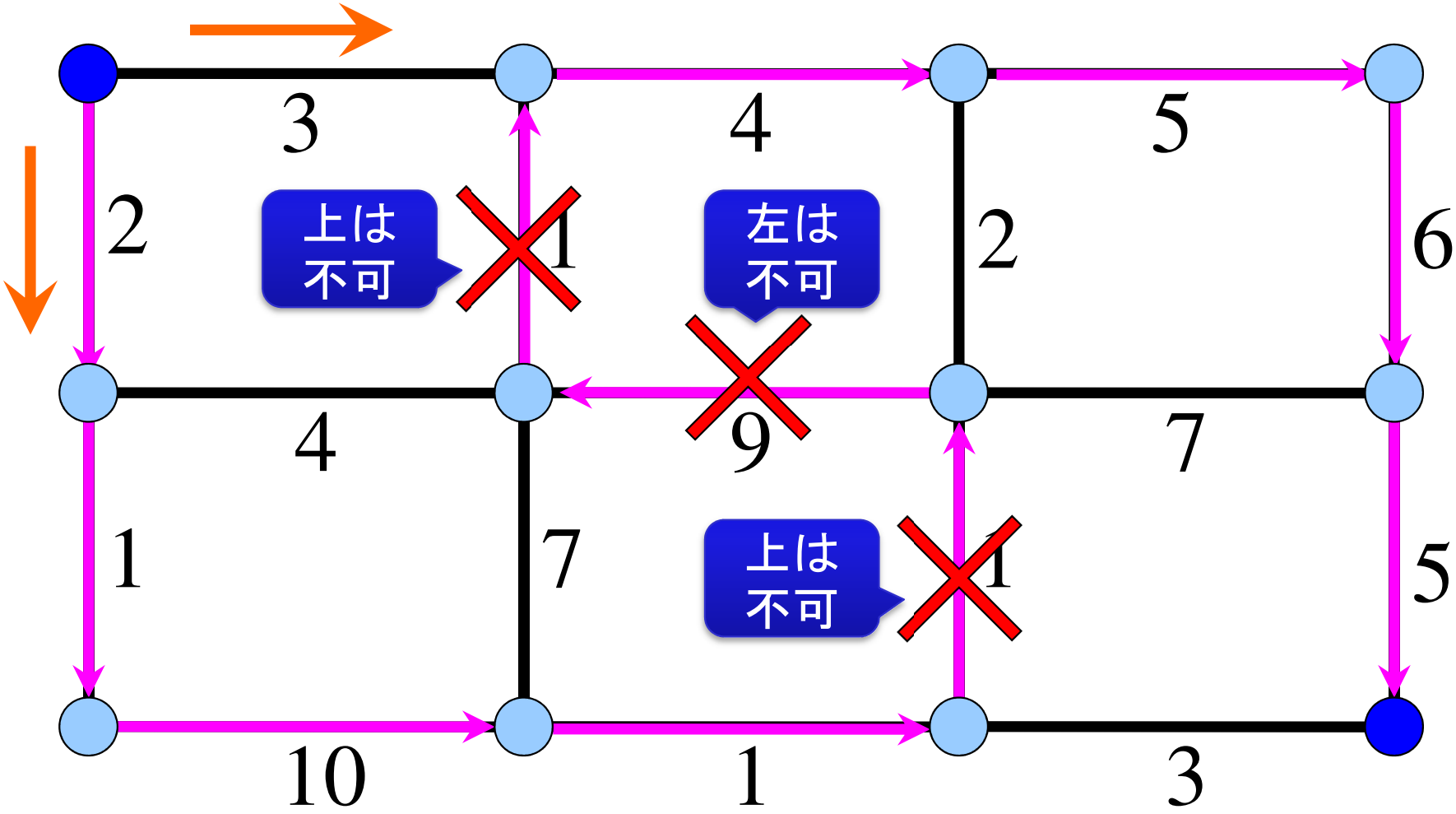
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

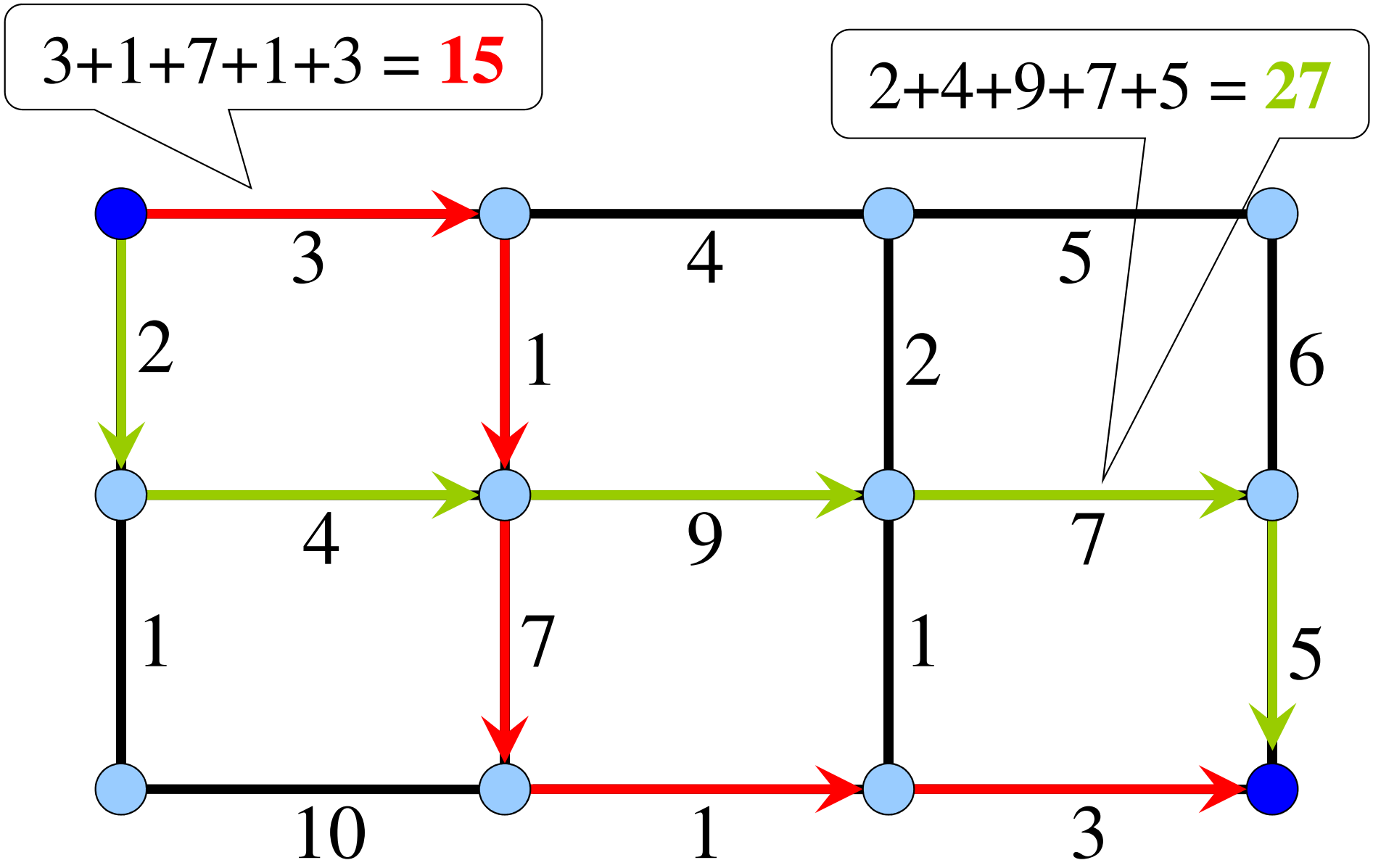
難しいなら易しくすればいいのさ！

制限した問題

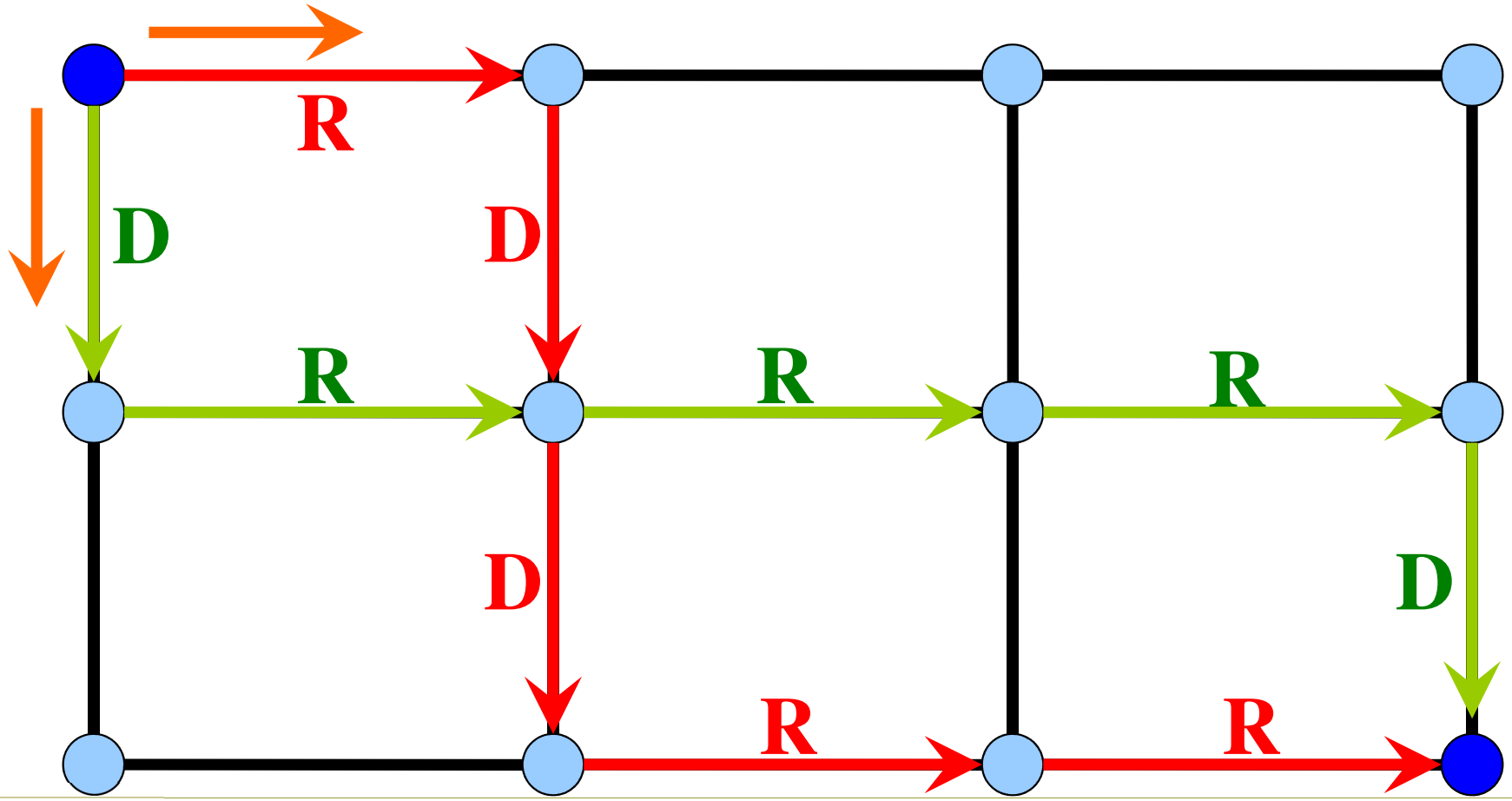
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



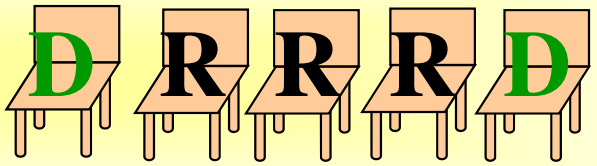
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



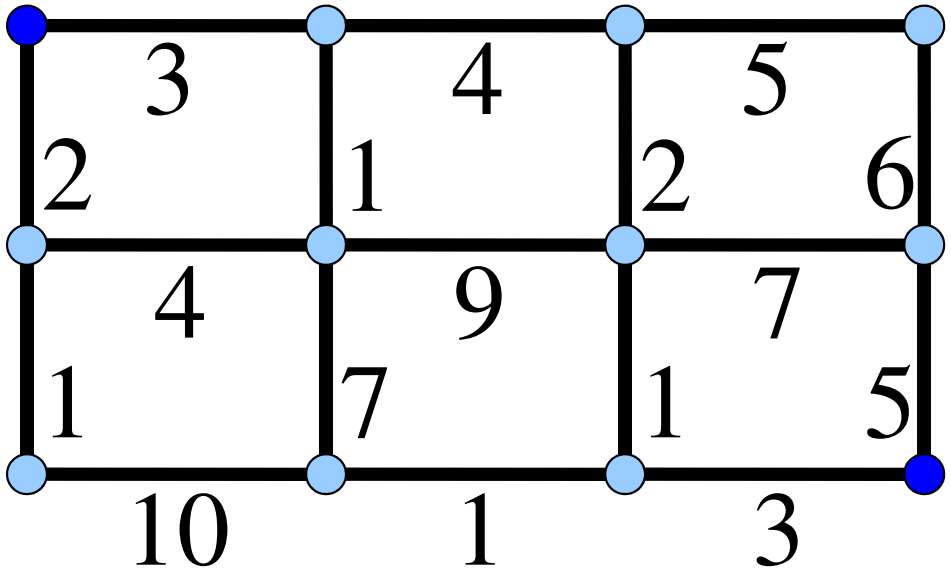
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

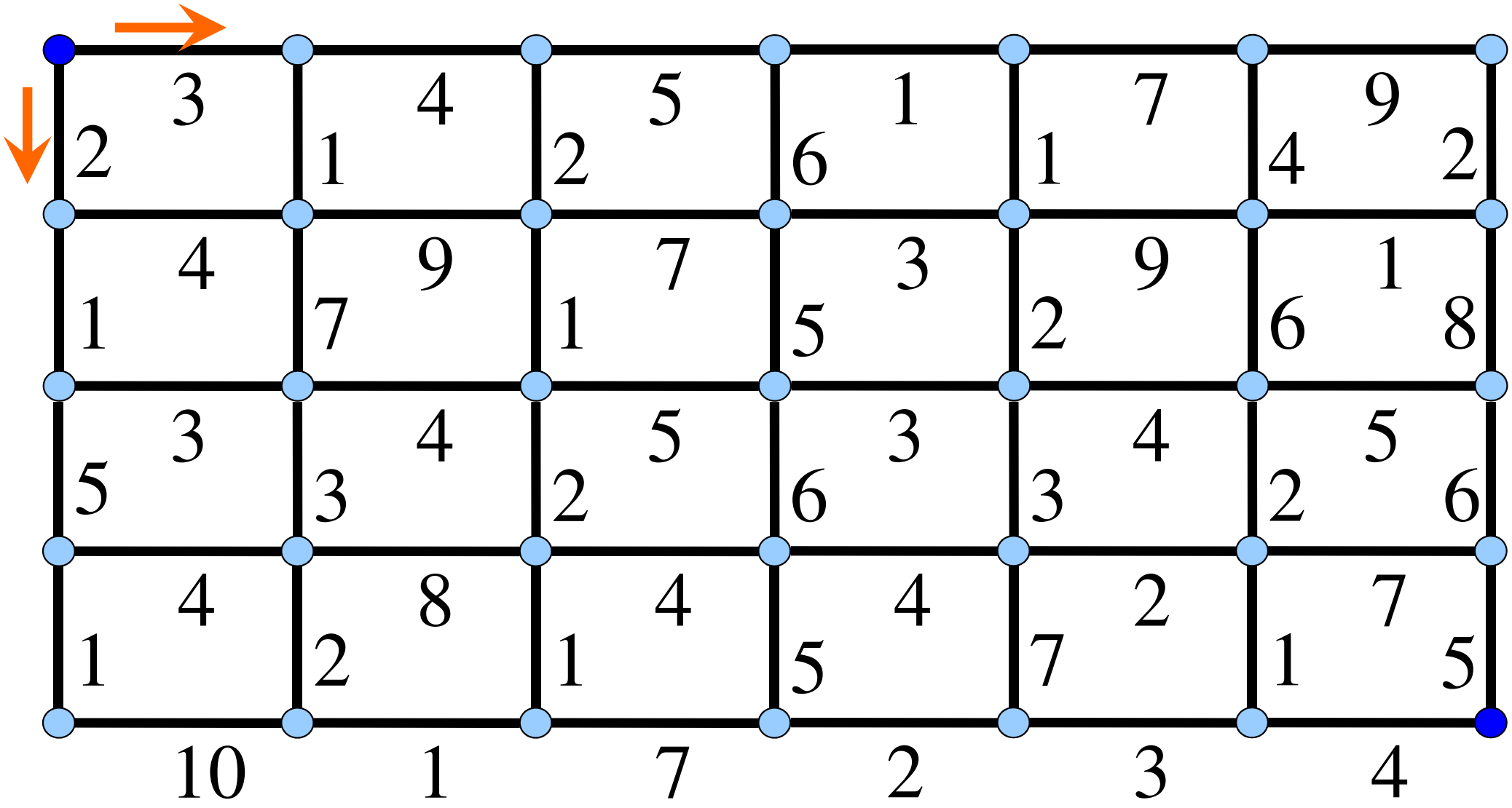
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

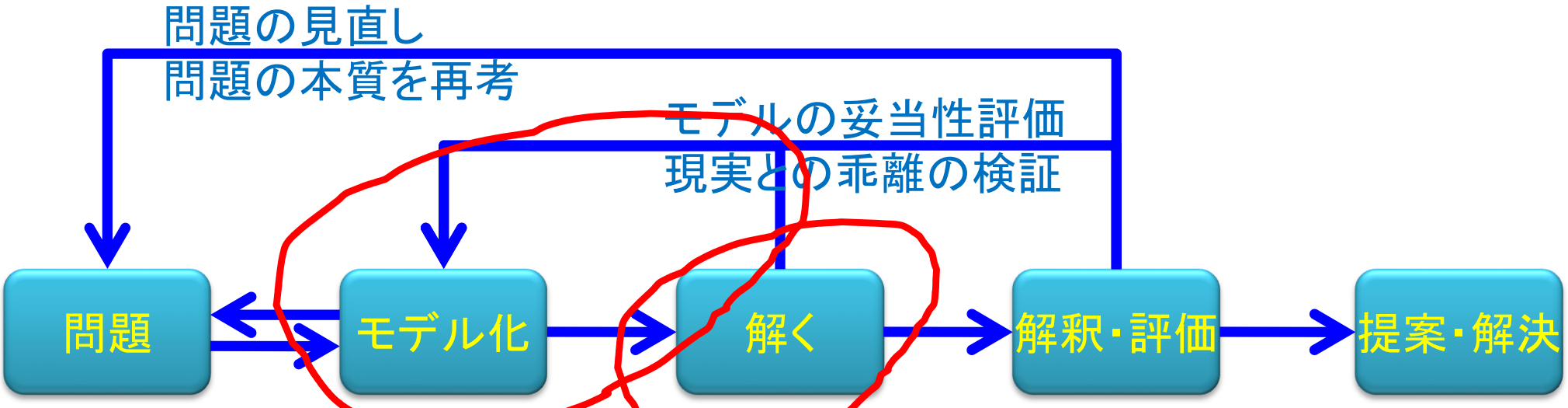
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

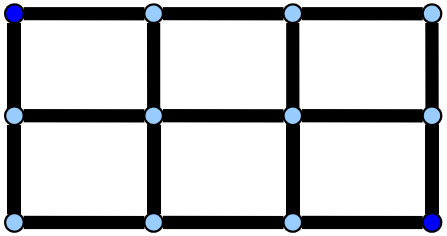
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

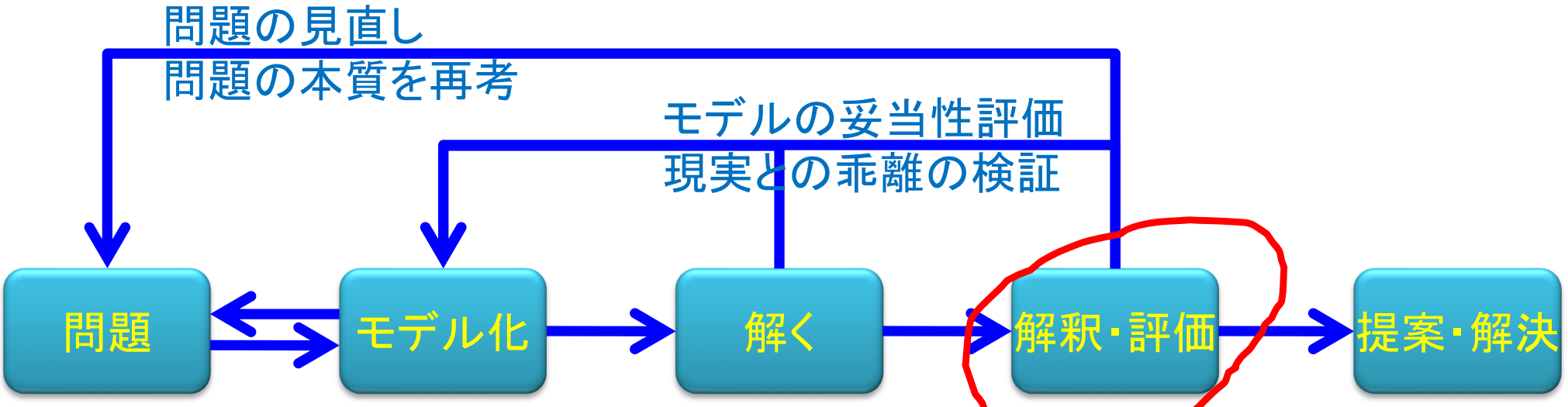
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

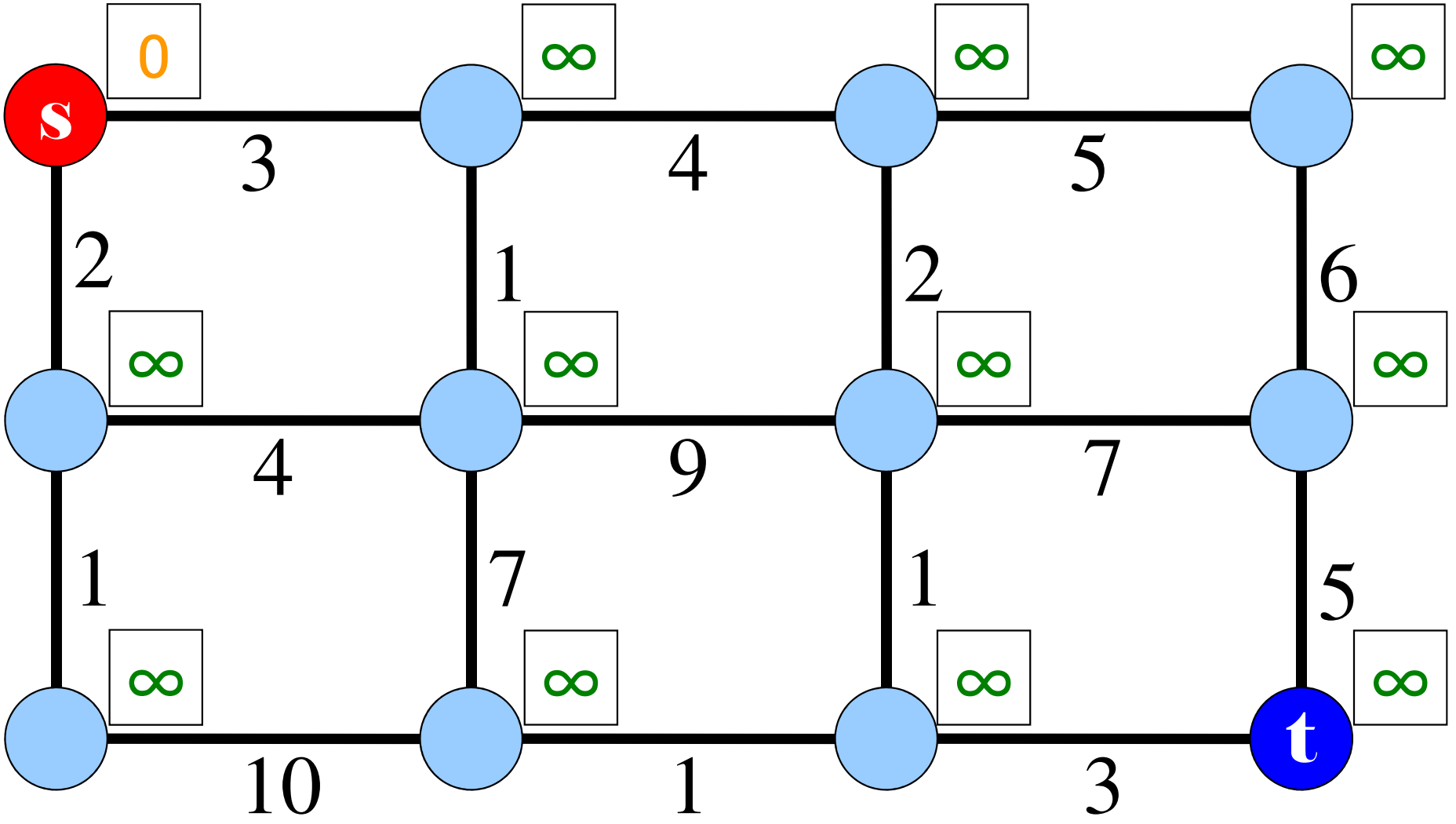
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

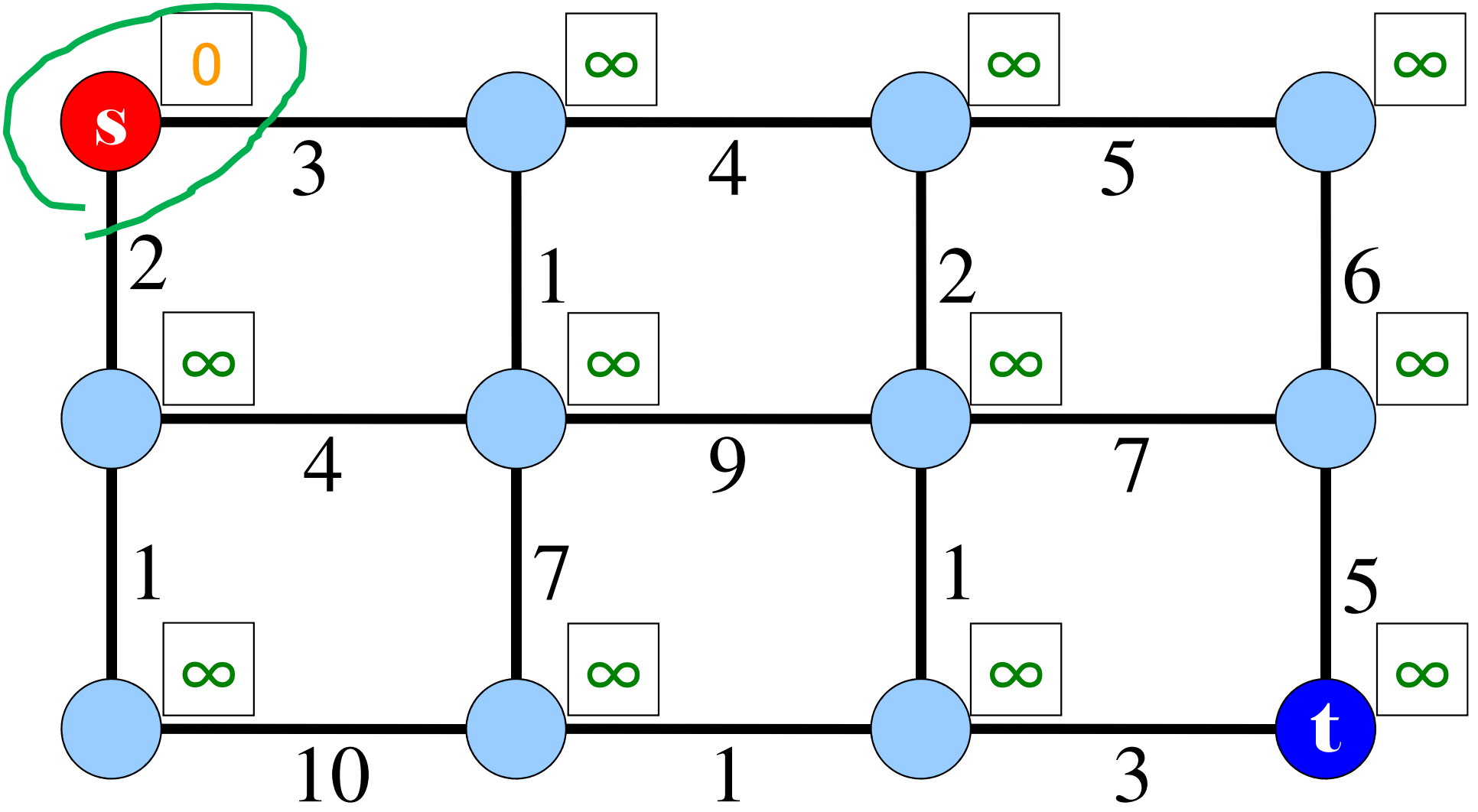
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



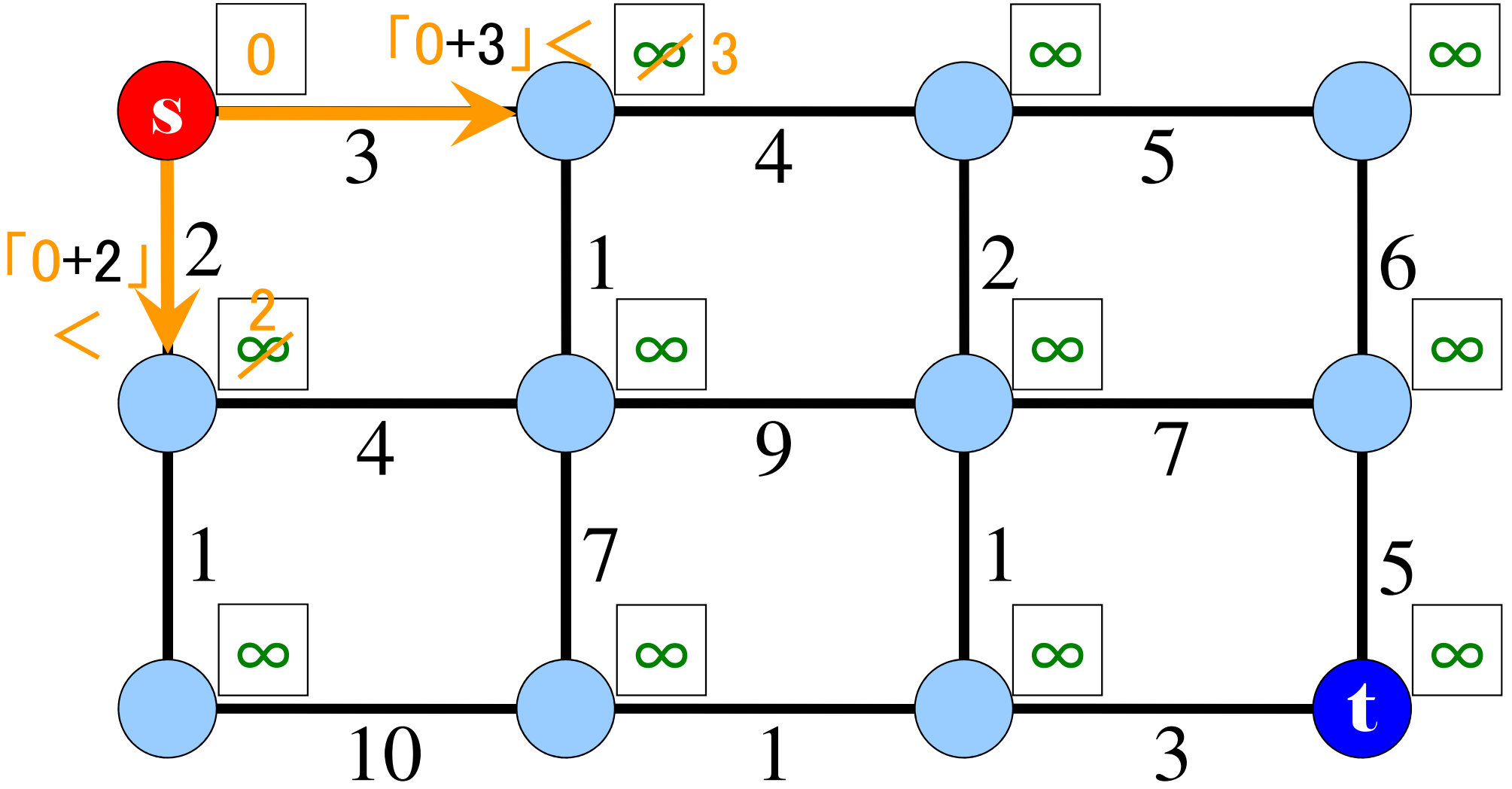
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

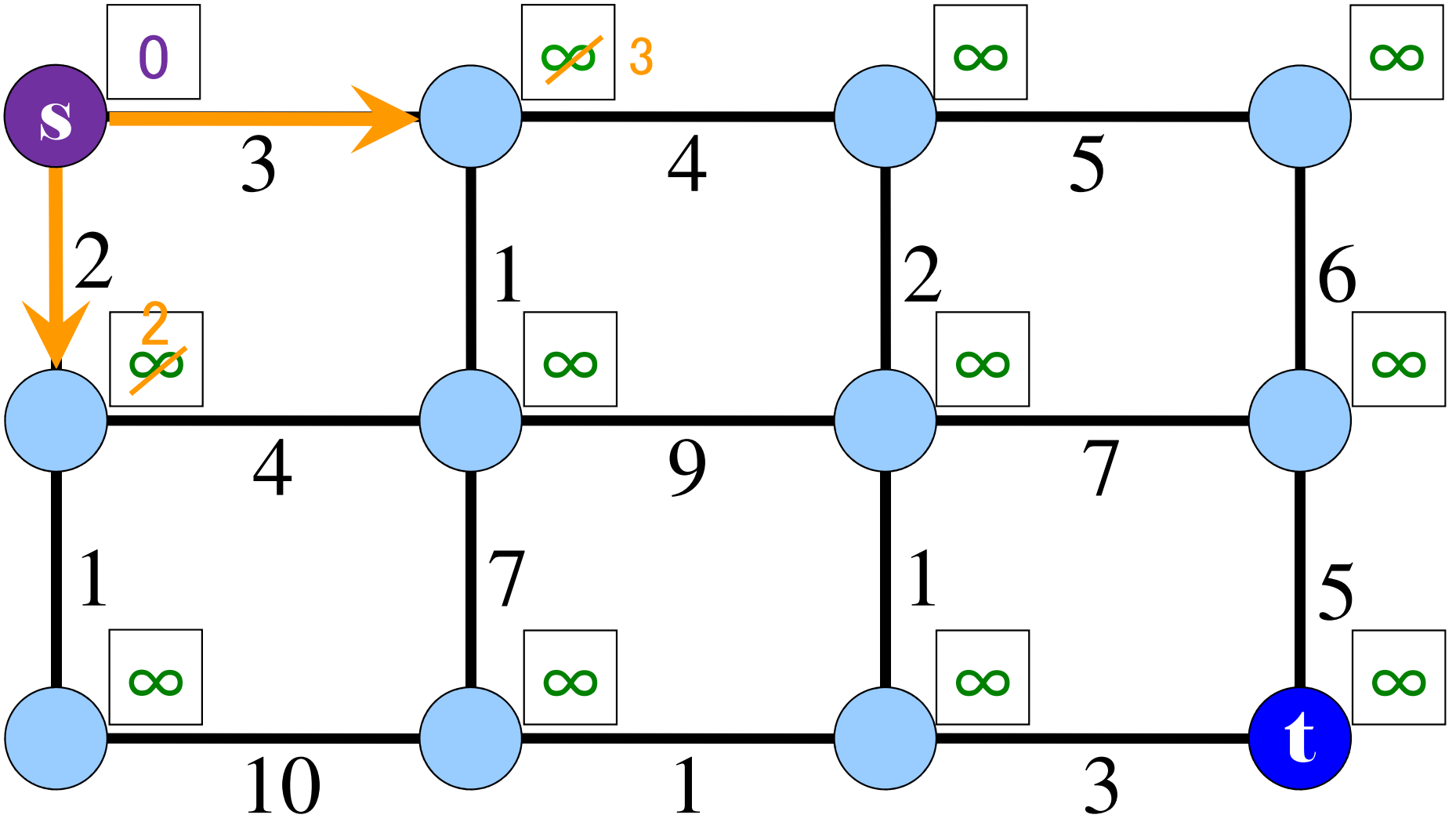


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

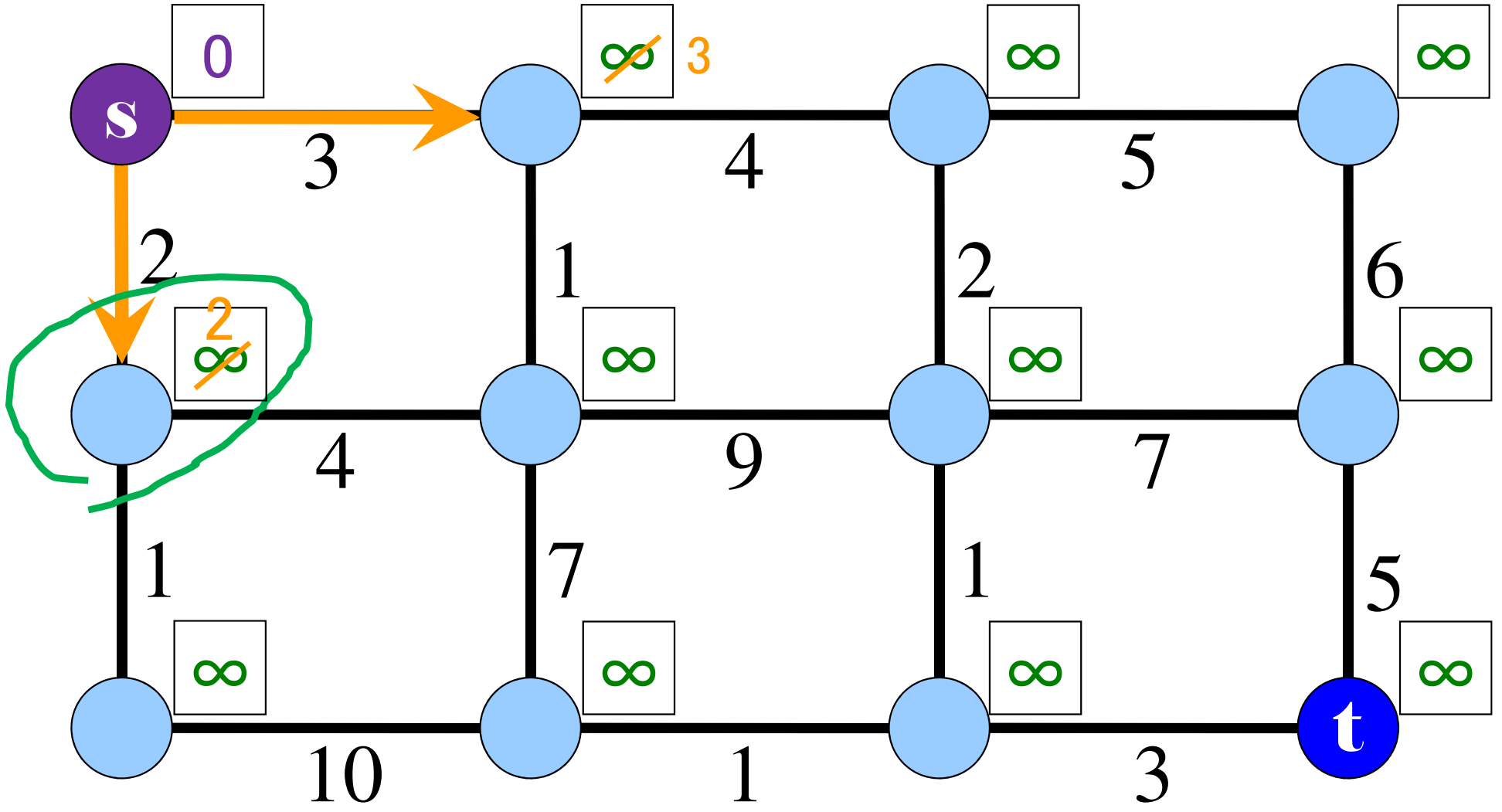
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



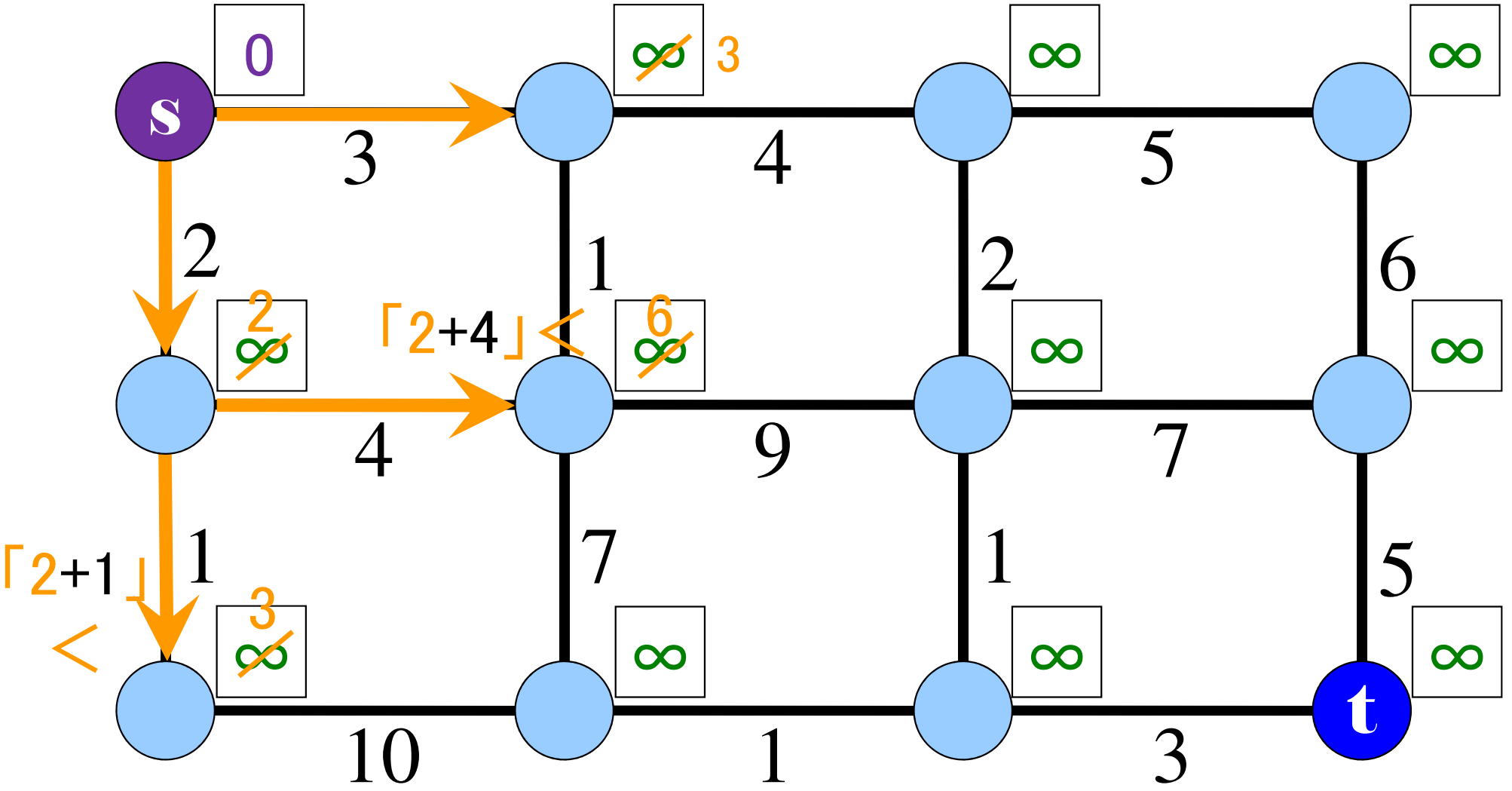
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

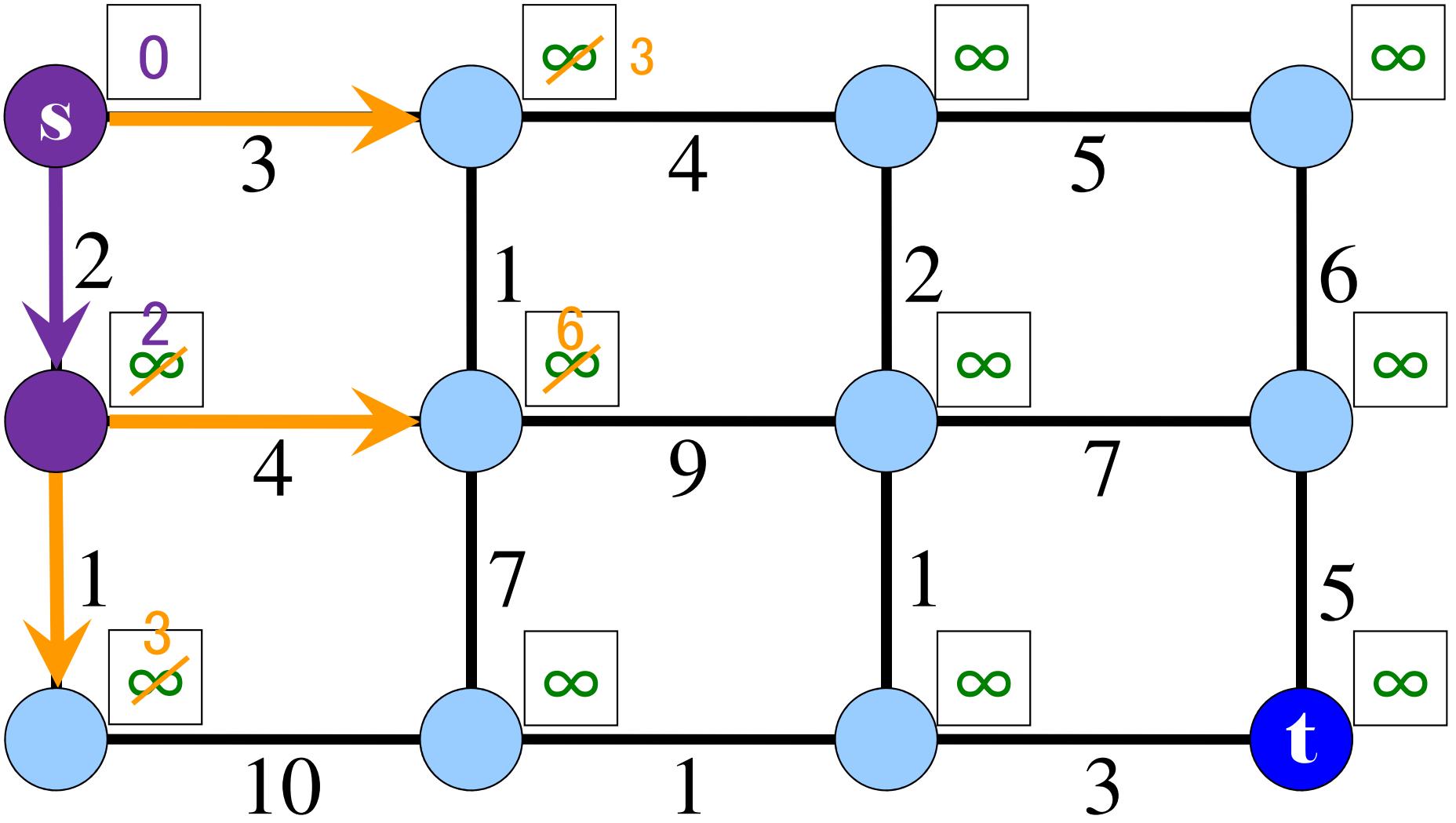


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

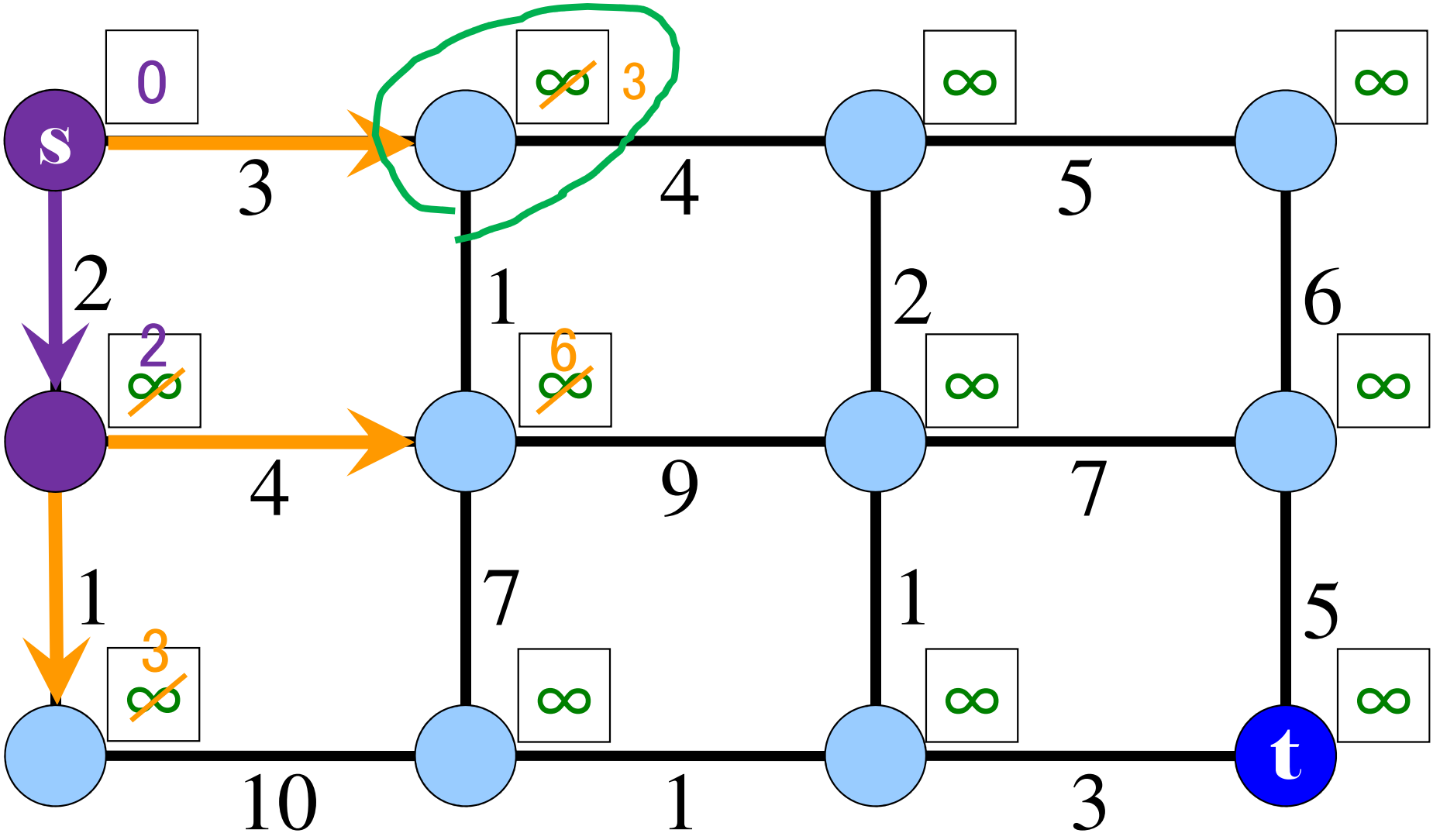
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



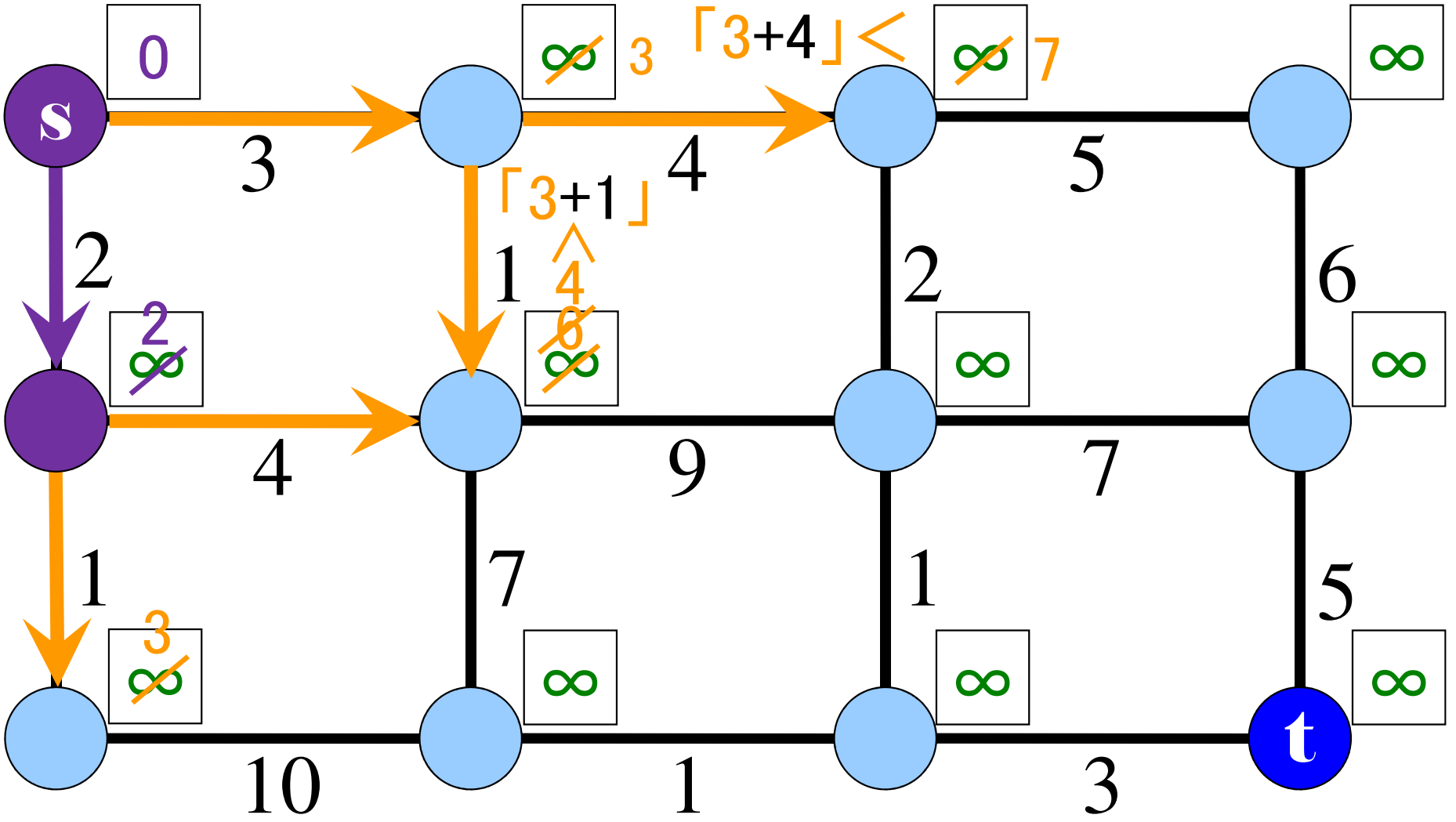
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

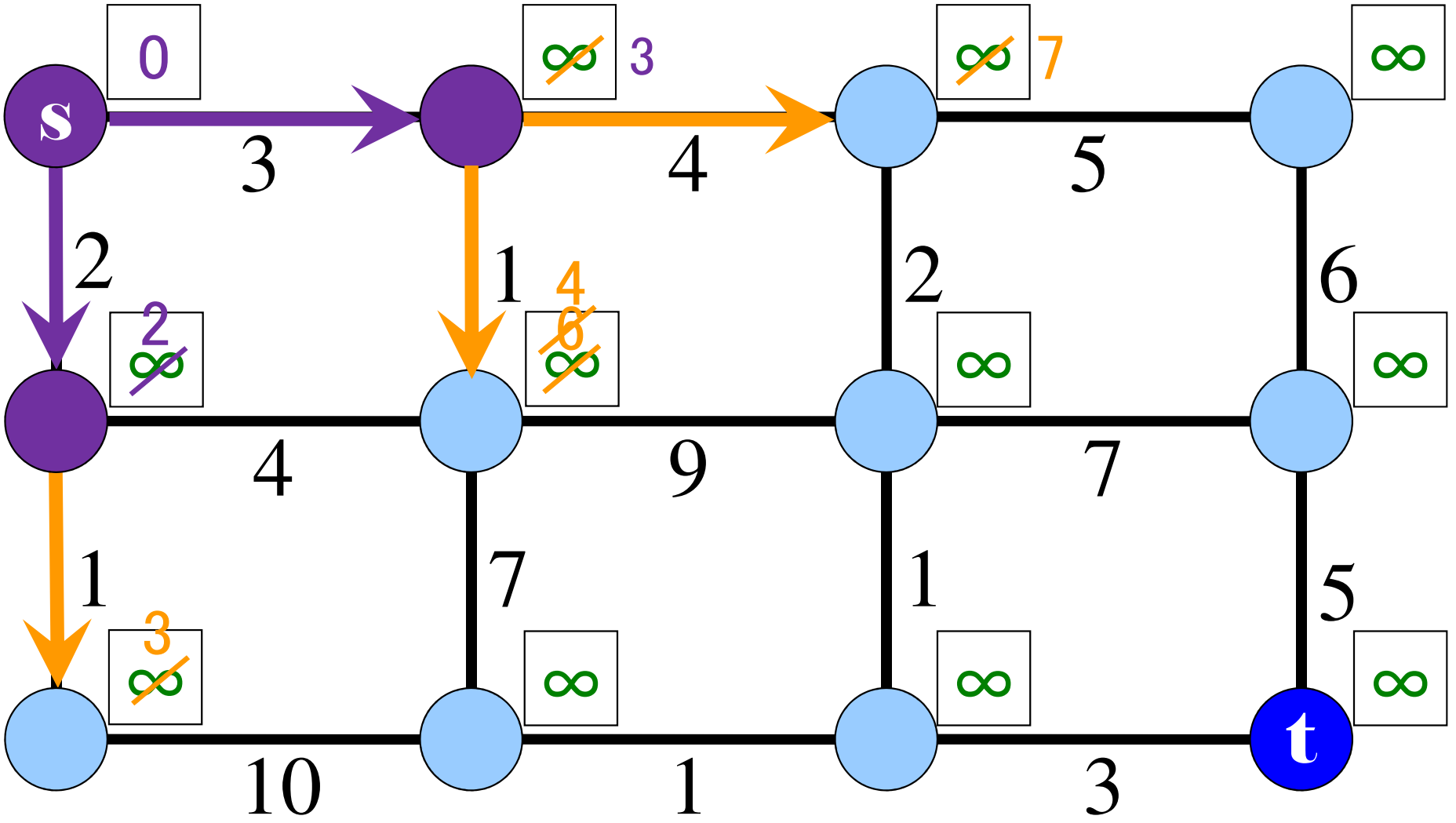


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

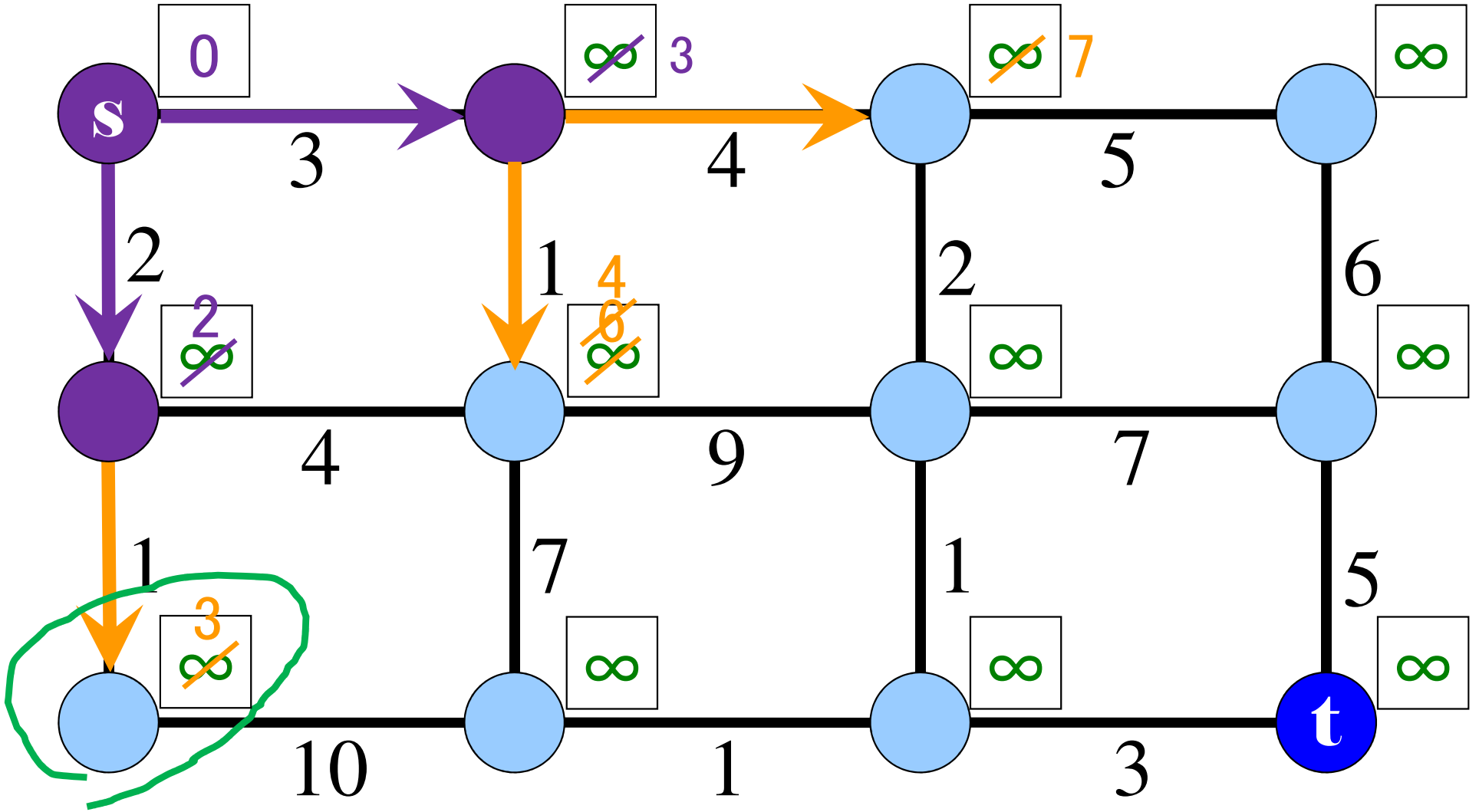
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



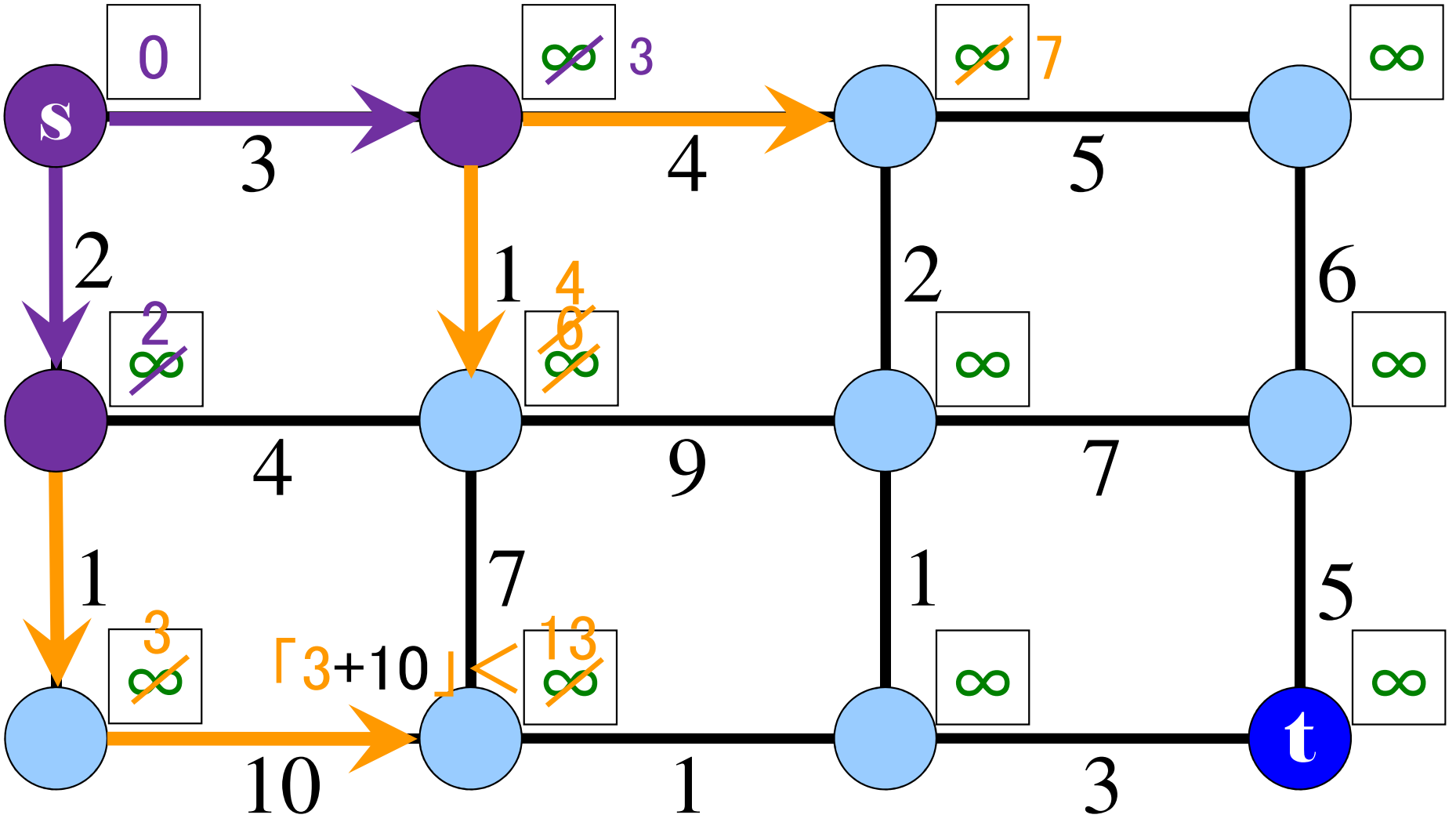
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

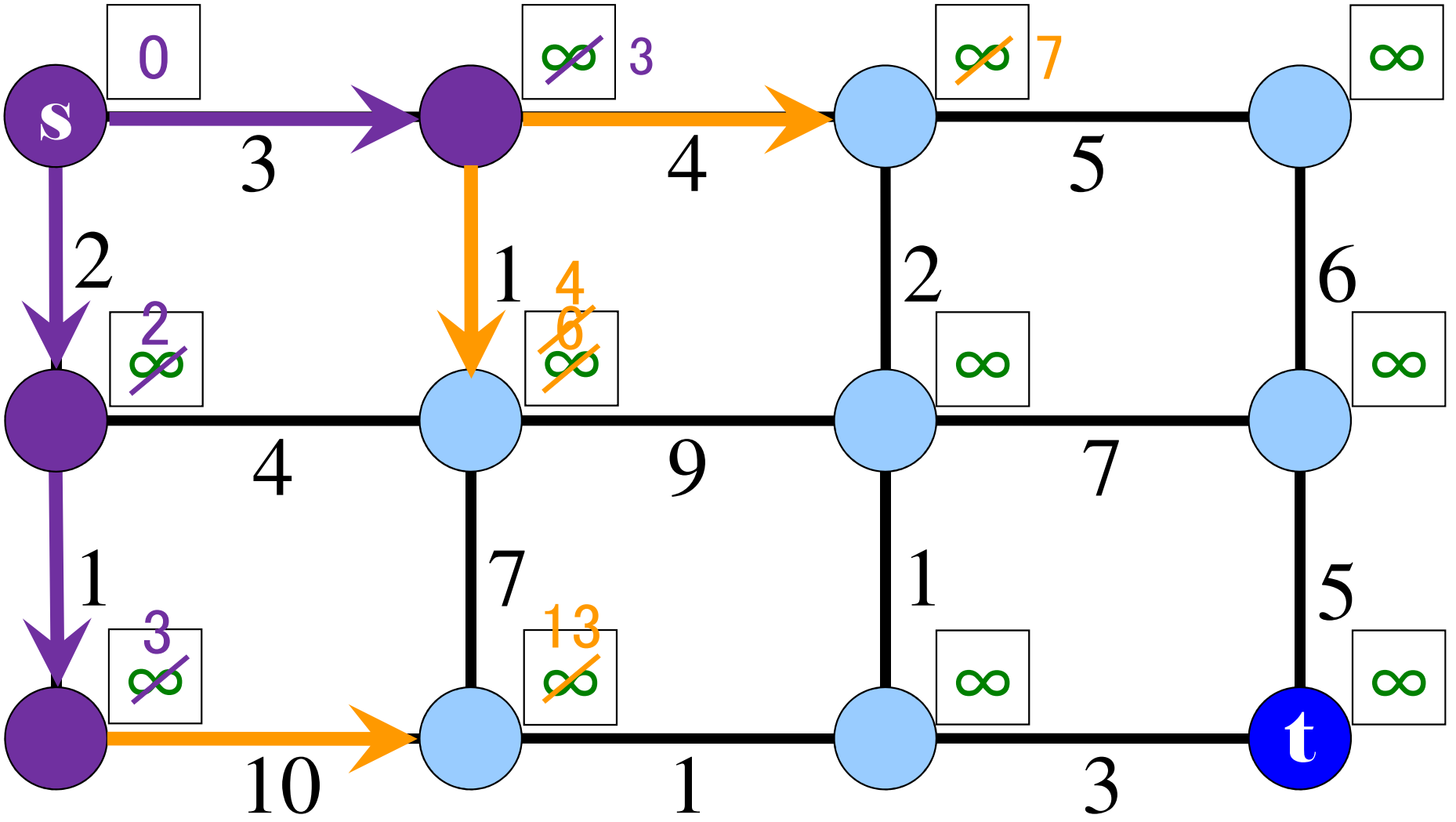


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

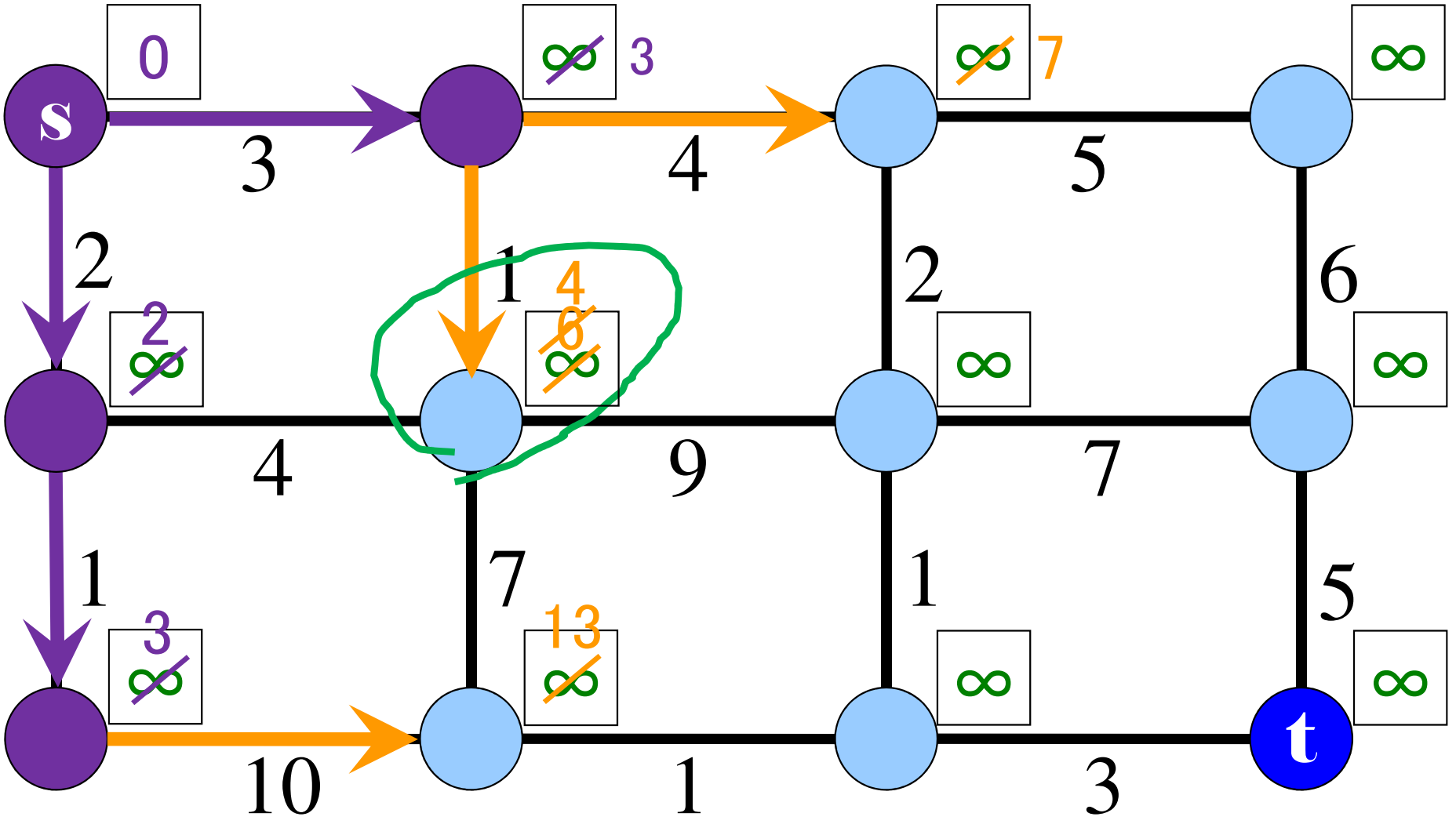
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



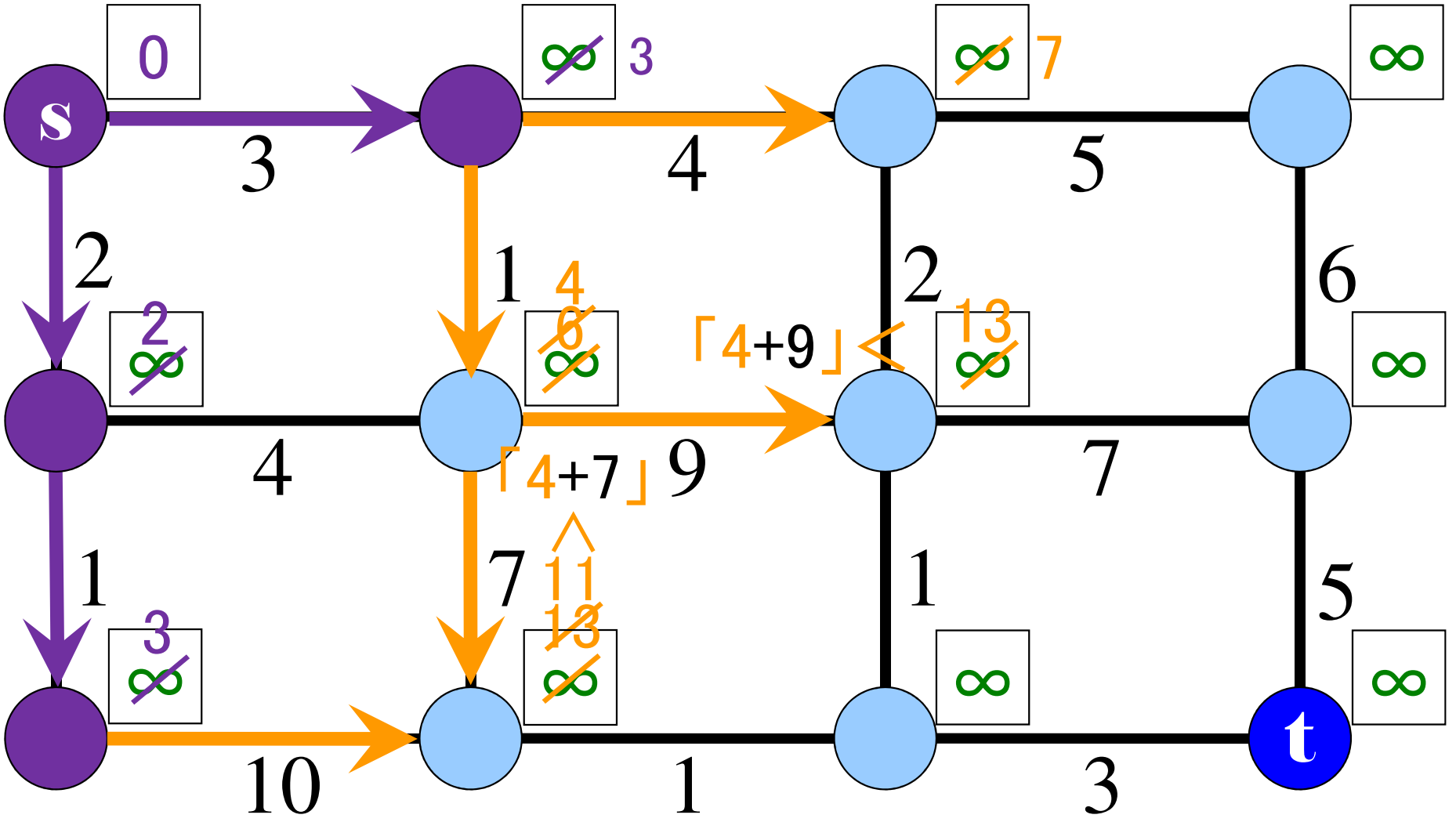
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

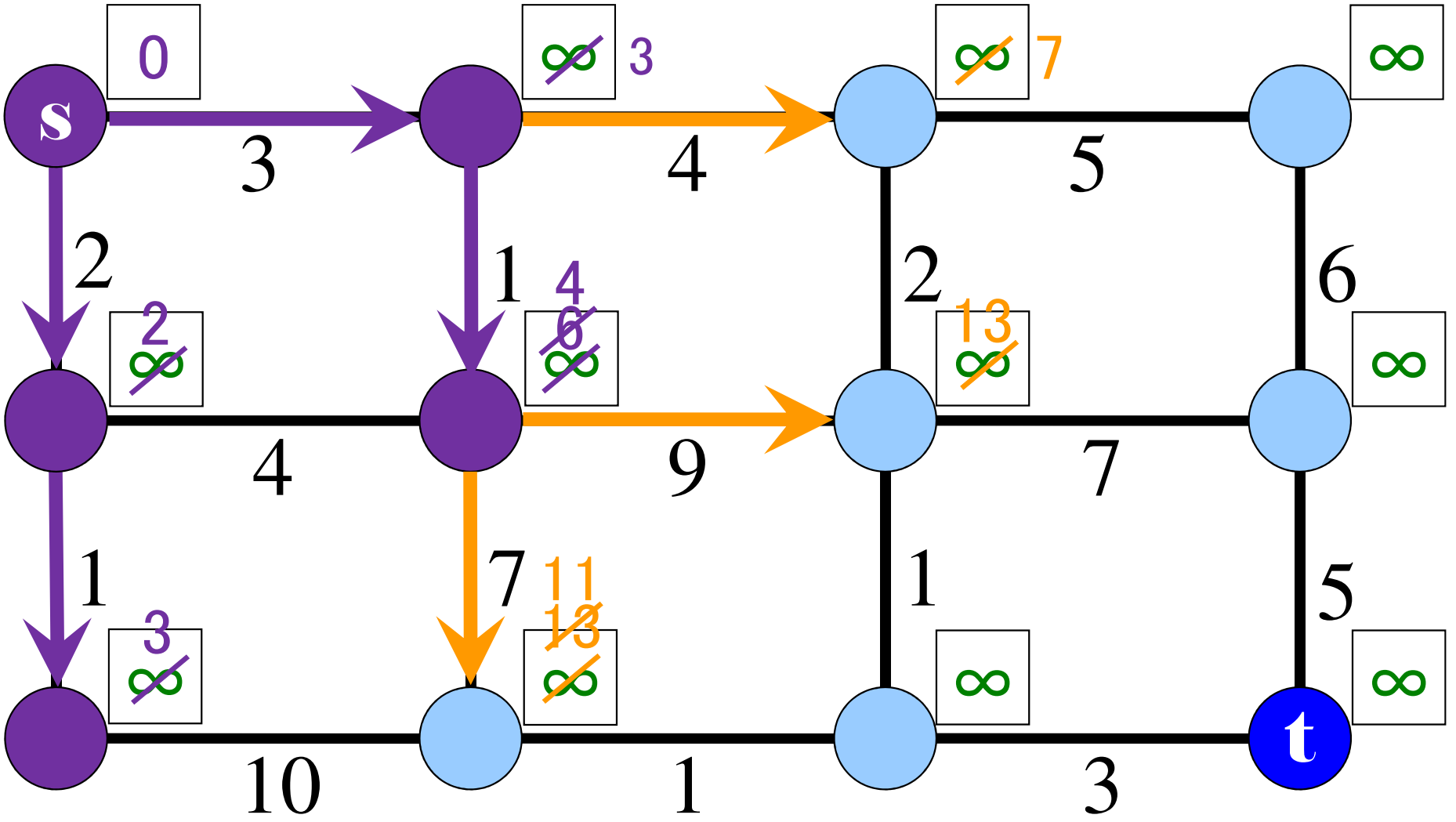


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

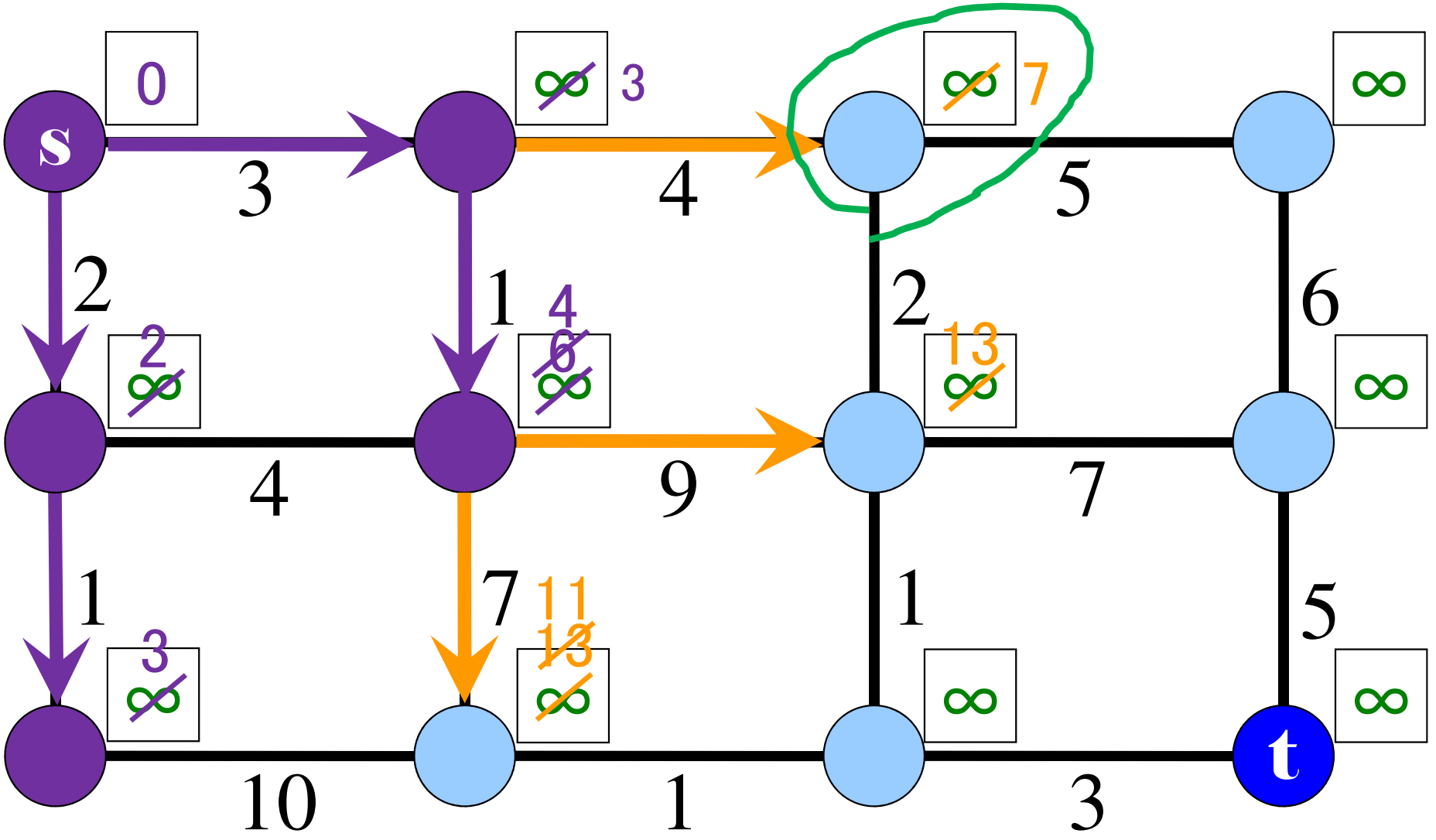
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



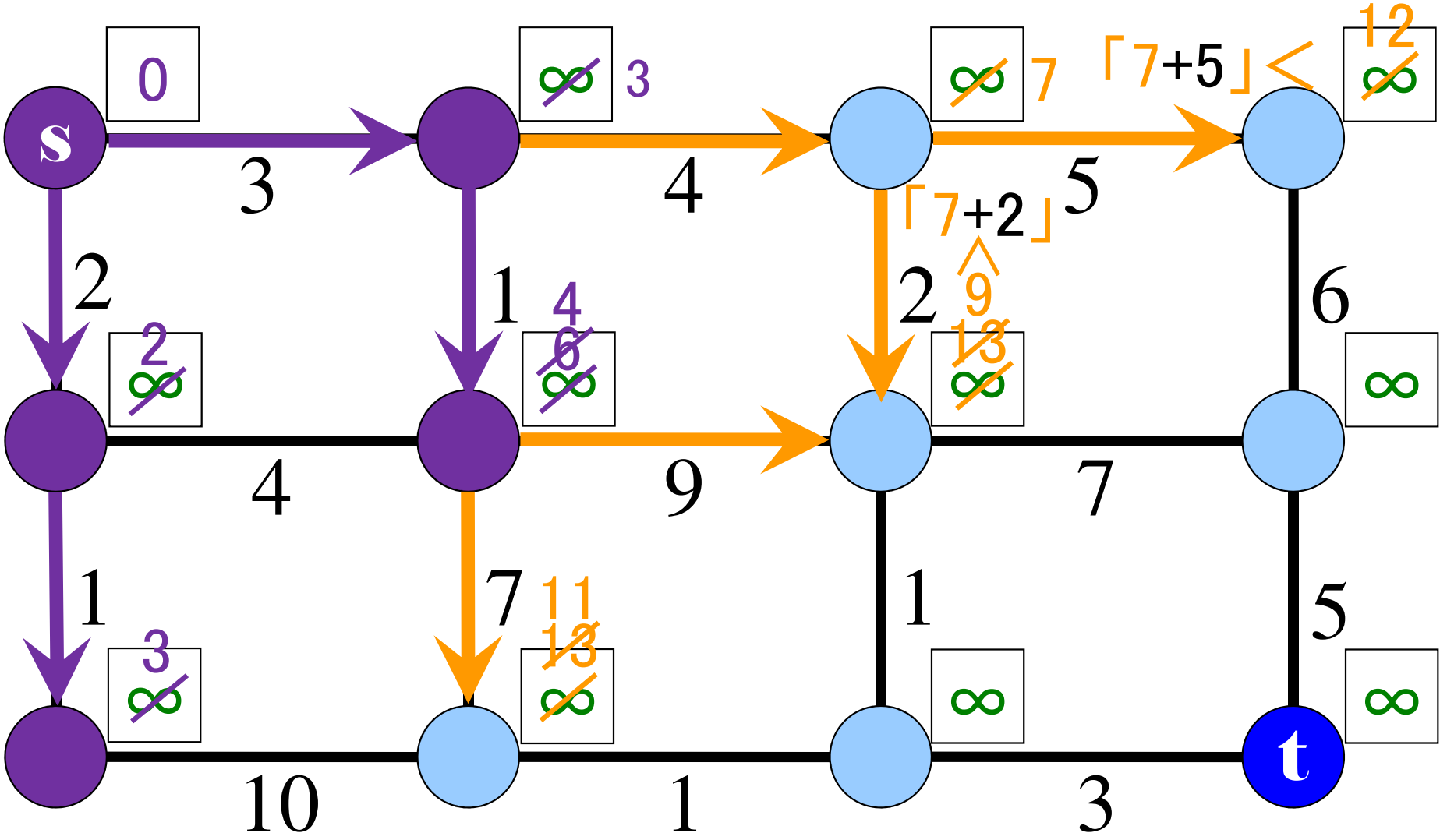
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

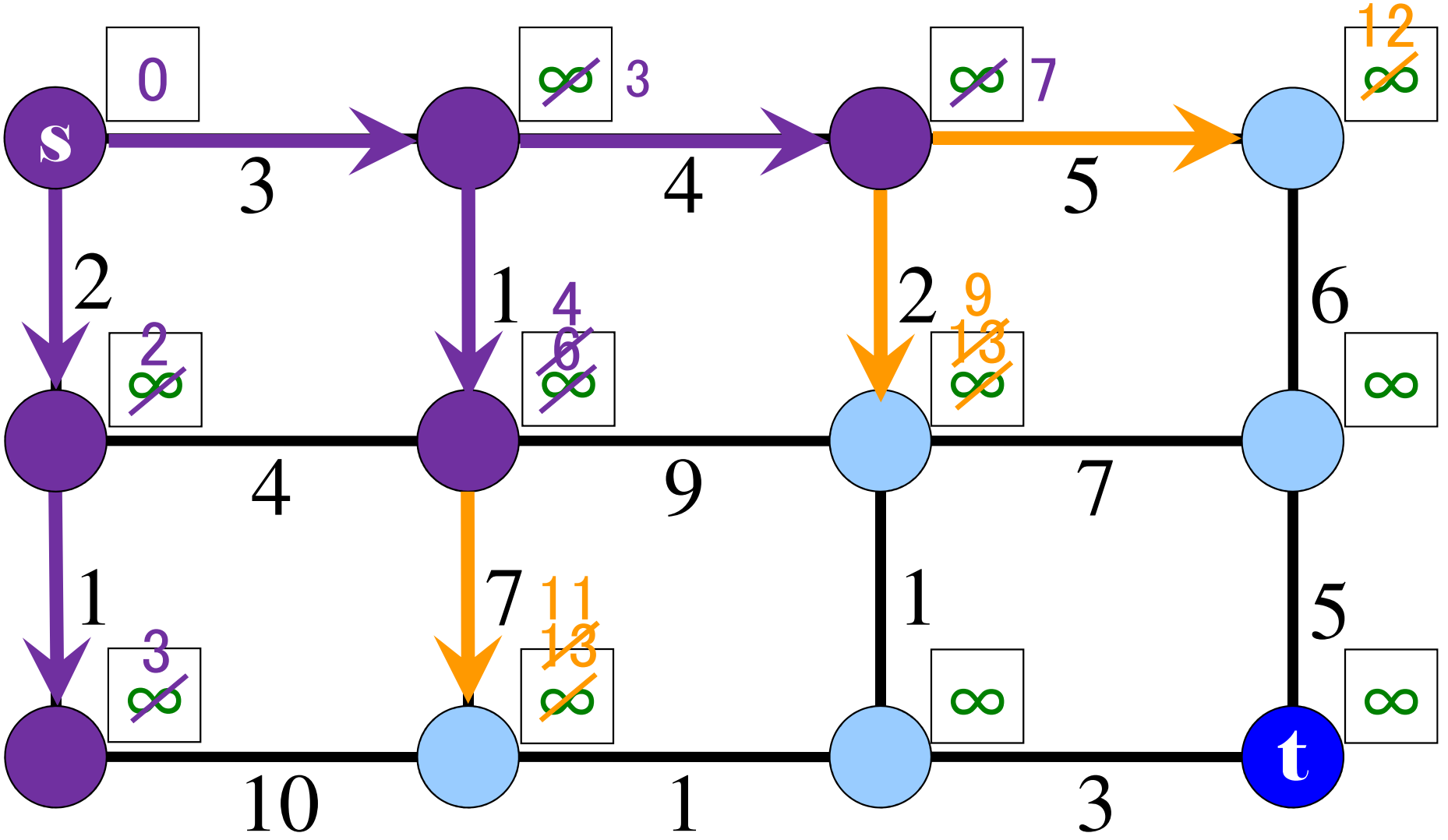


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

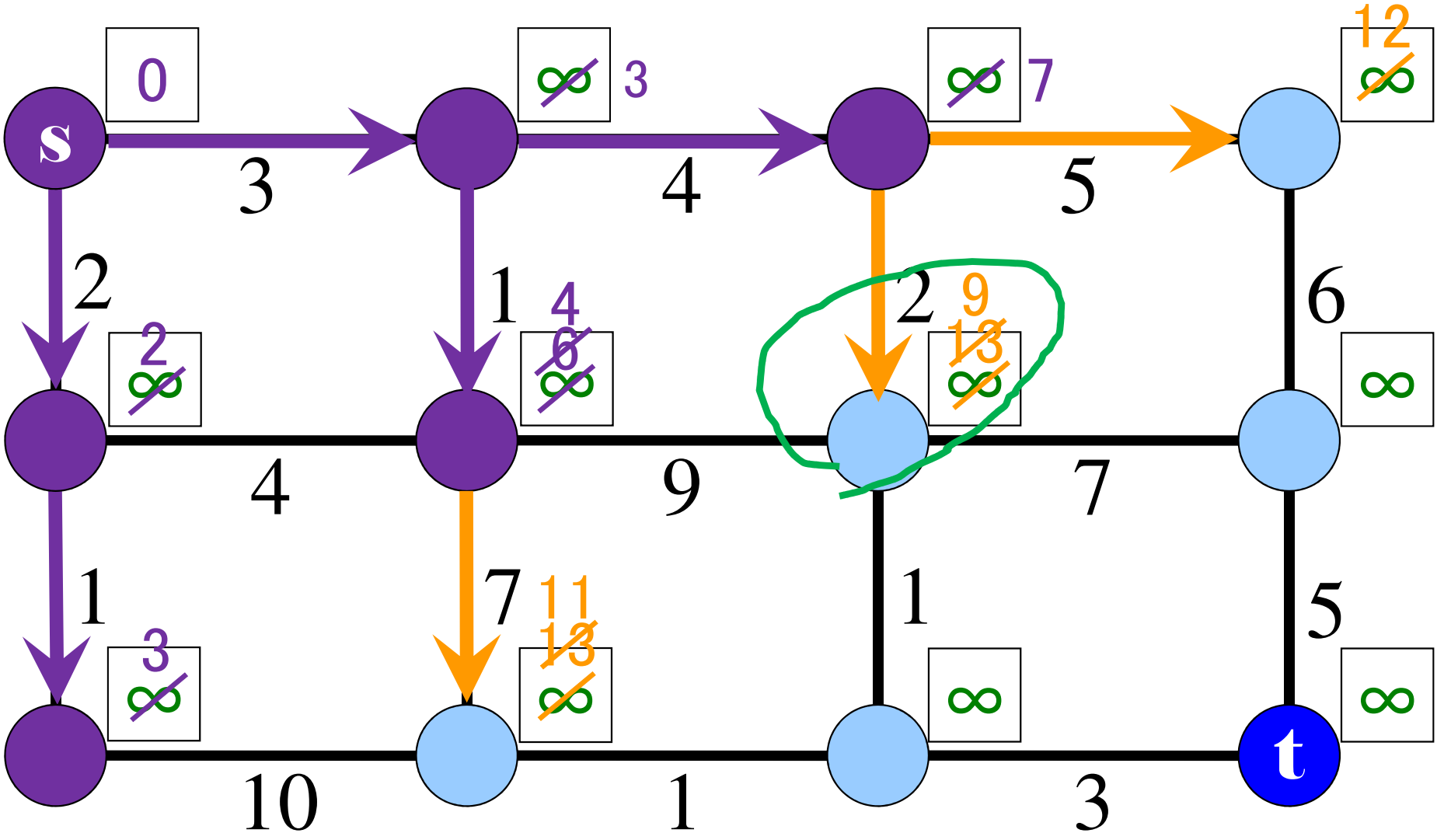
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



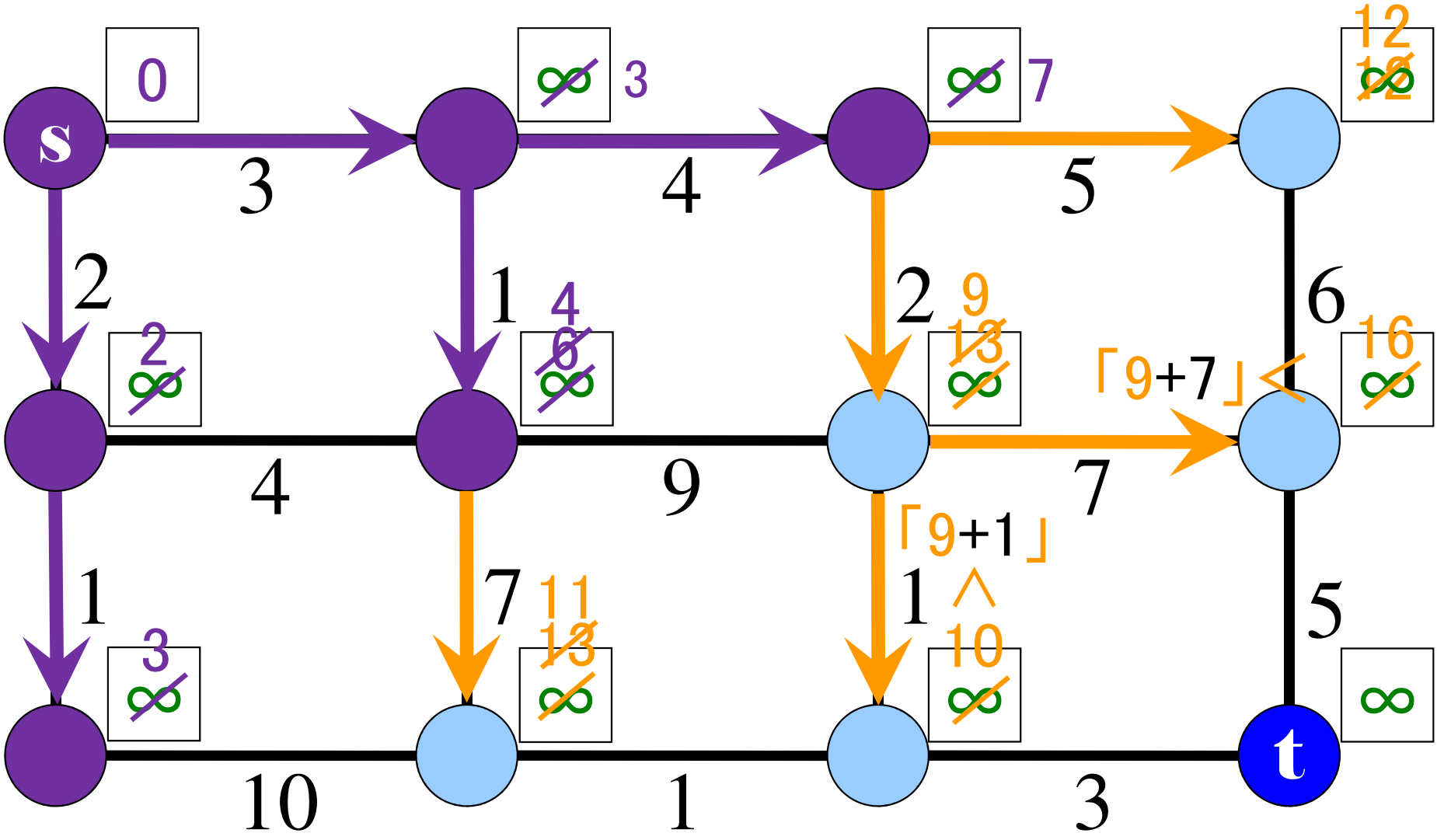
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

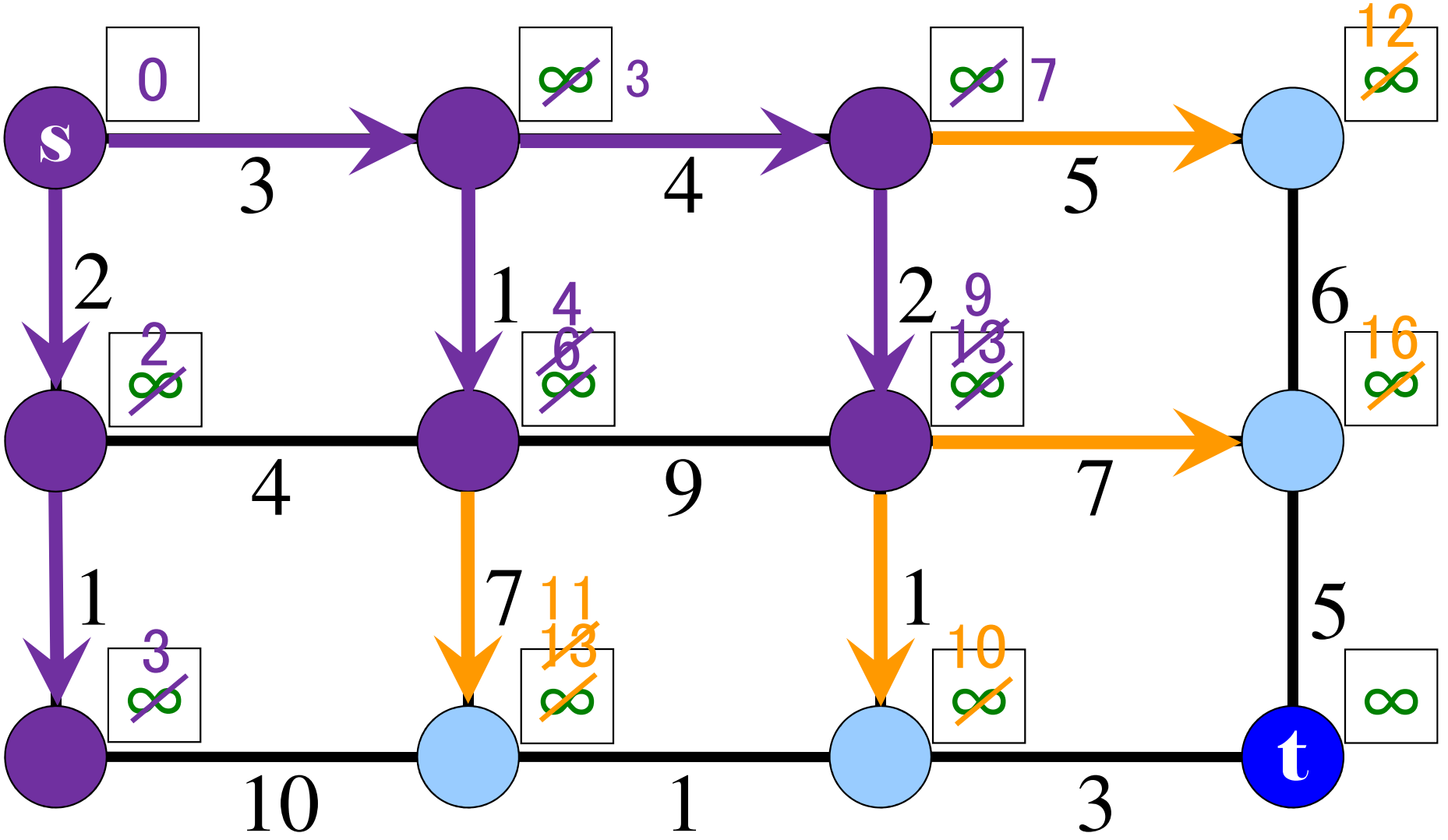


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

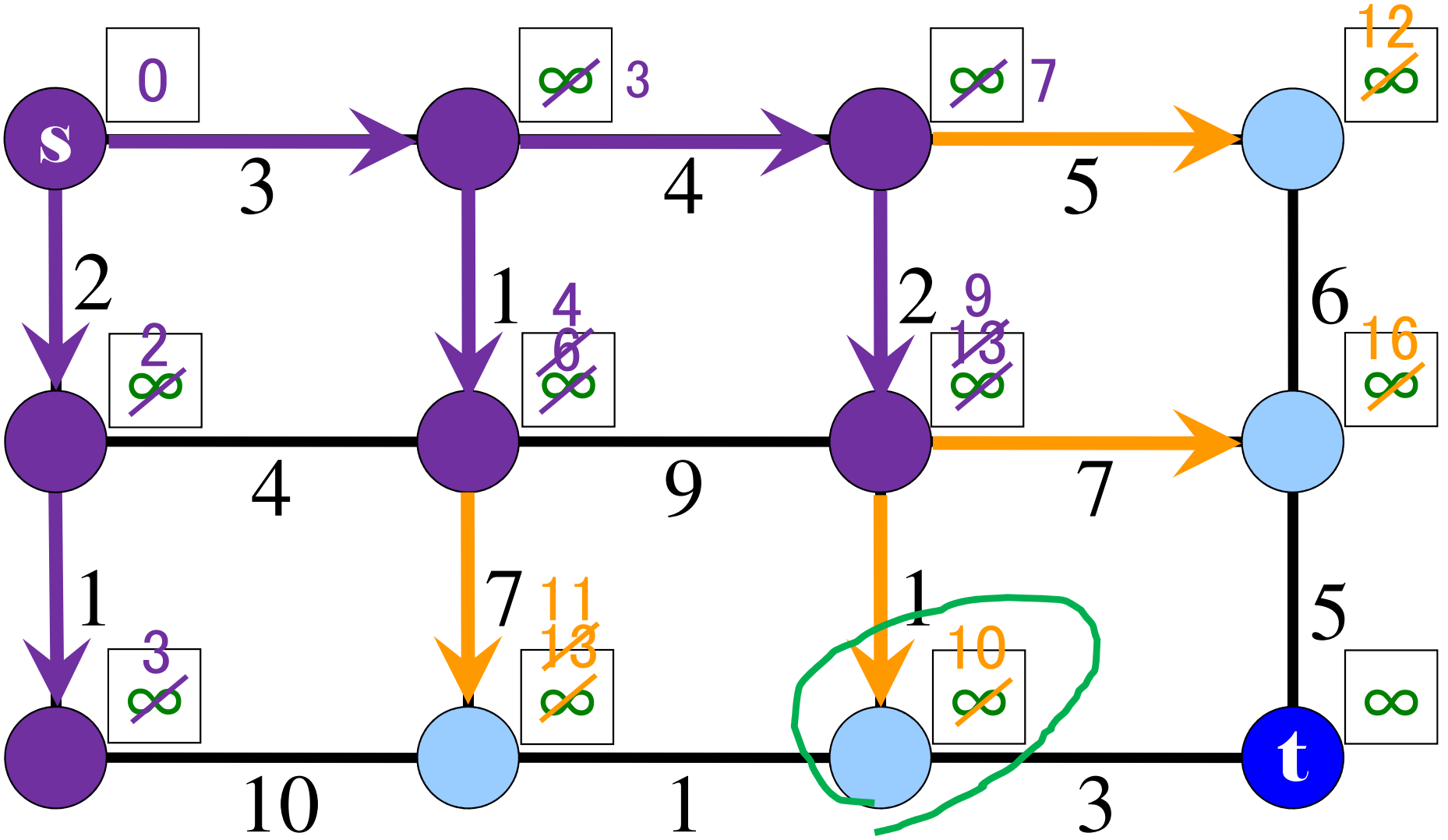
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



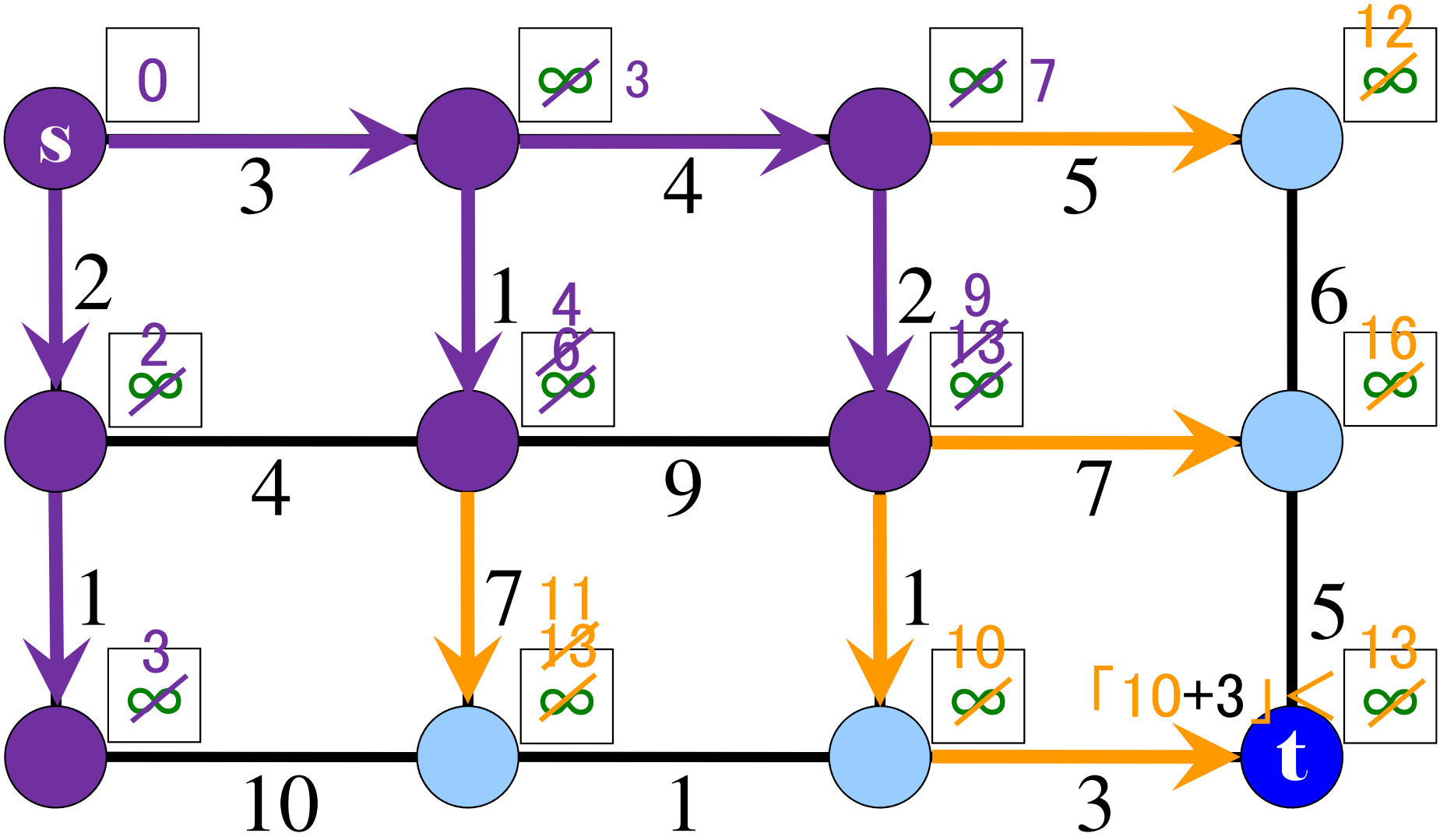
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

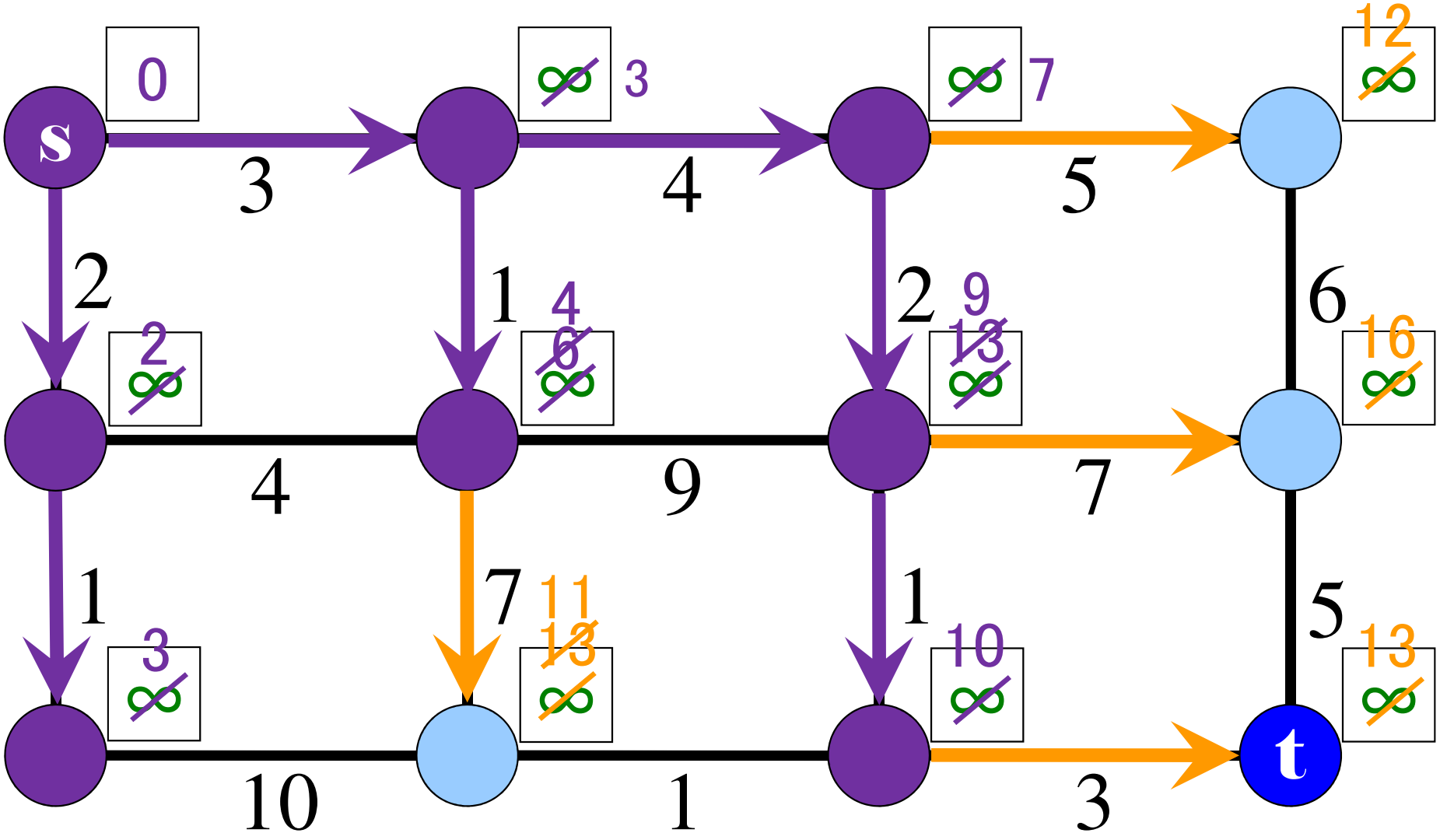


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

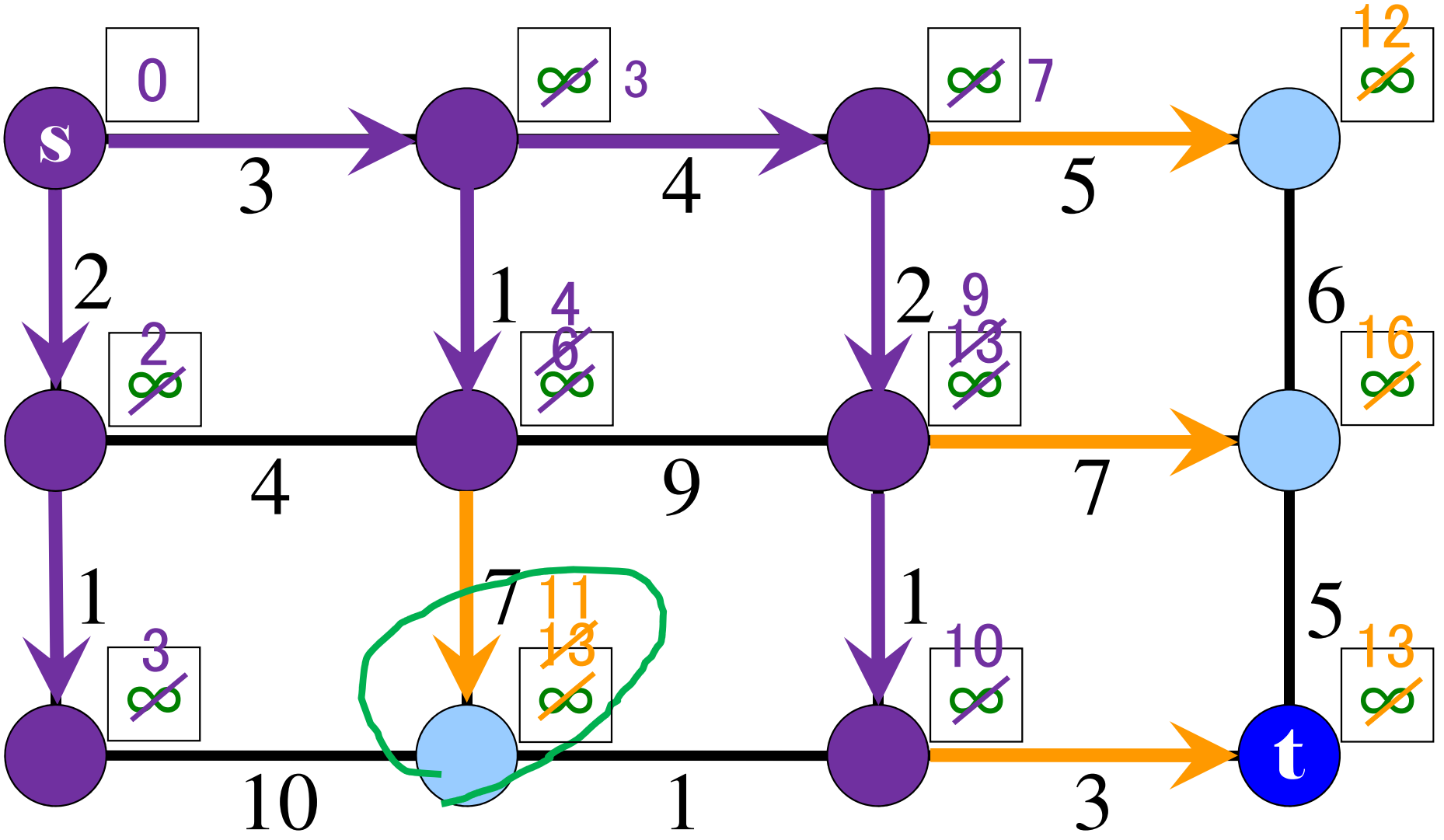
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



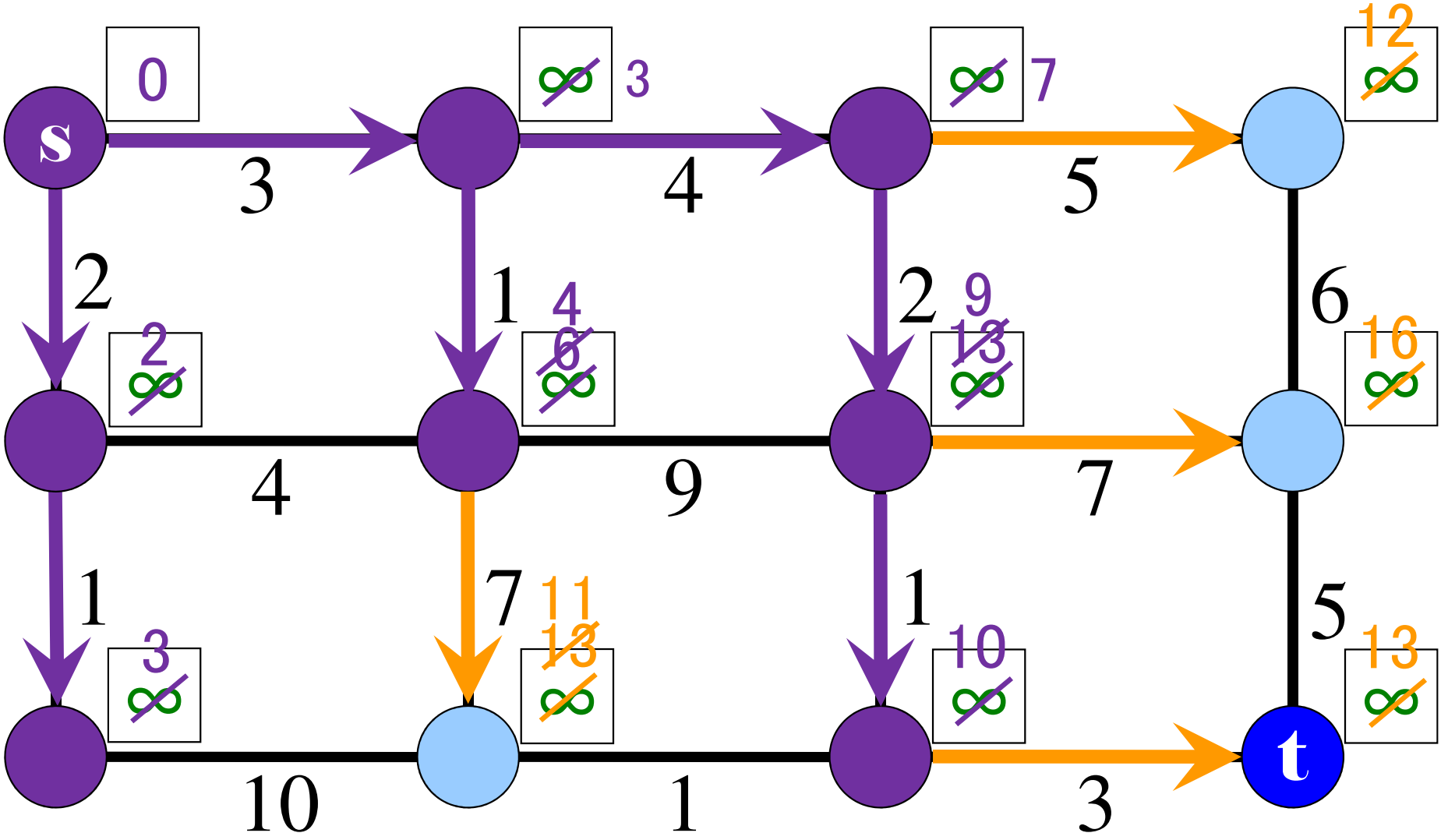
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

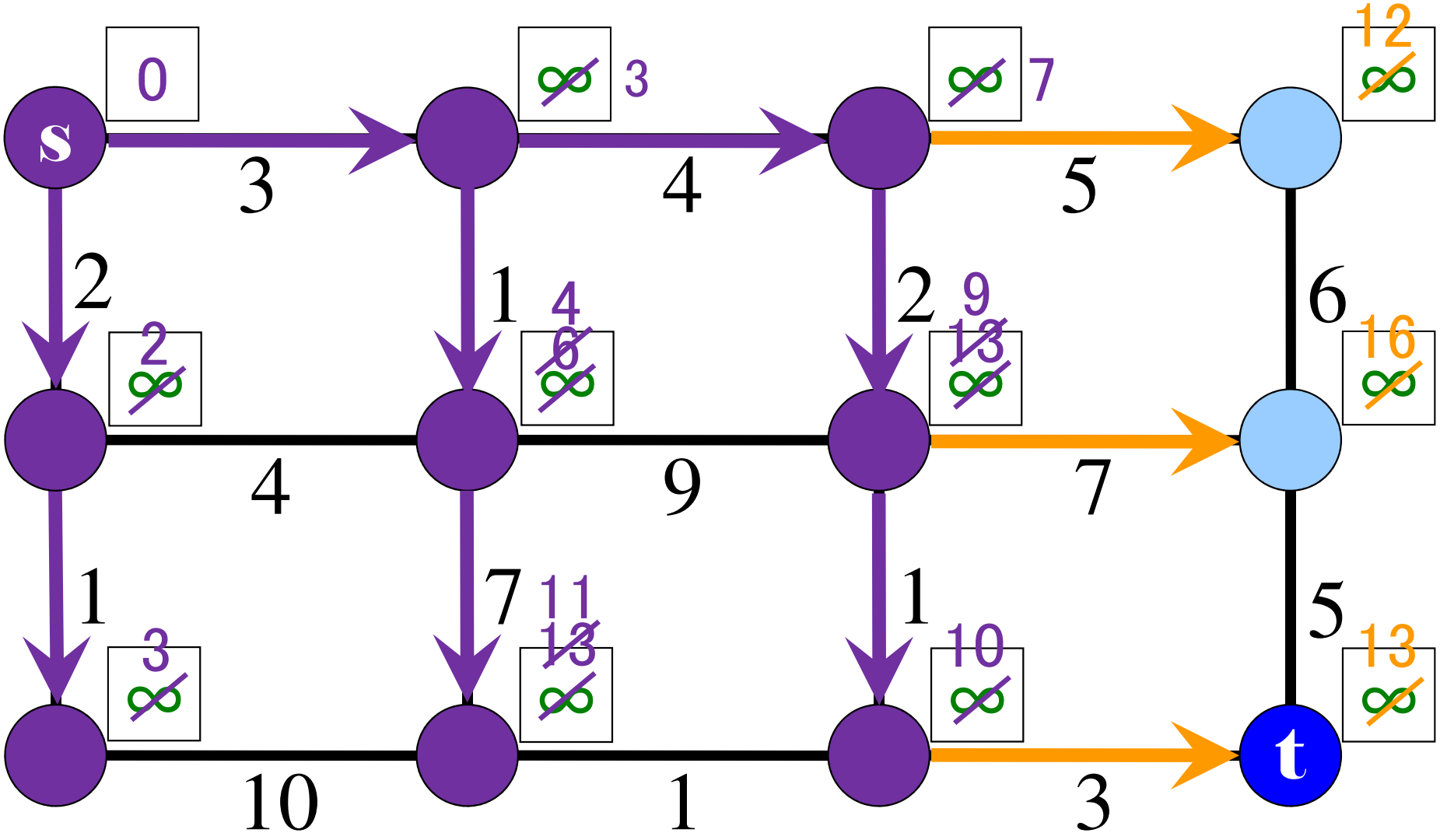


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

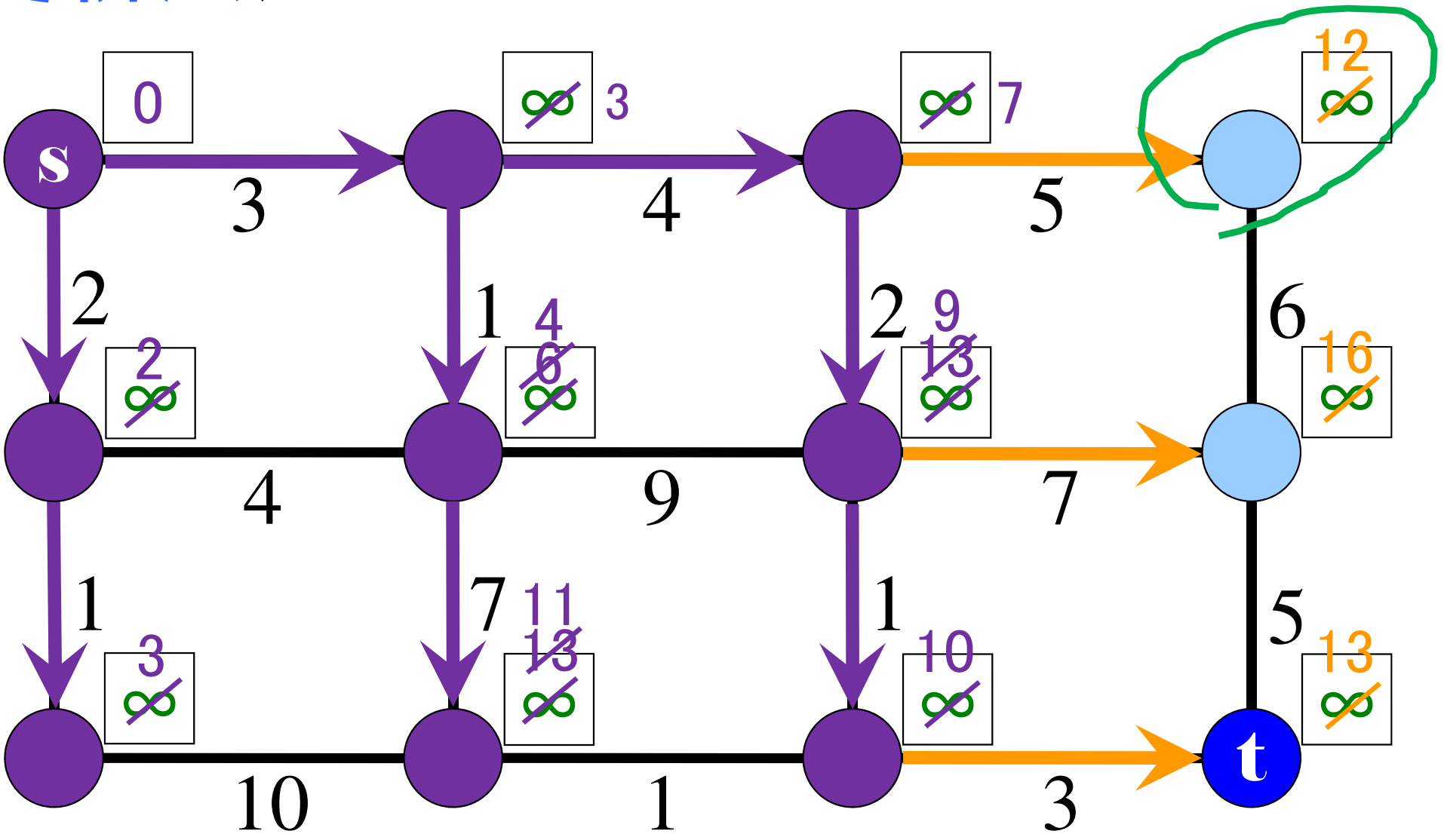
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



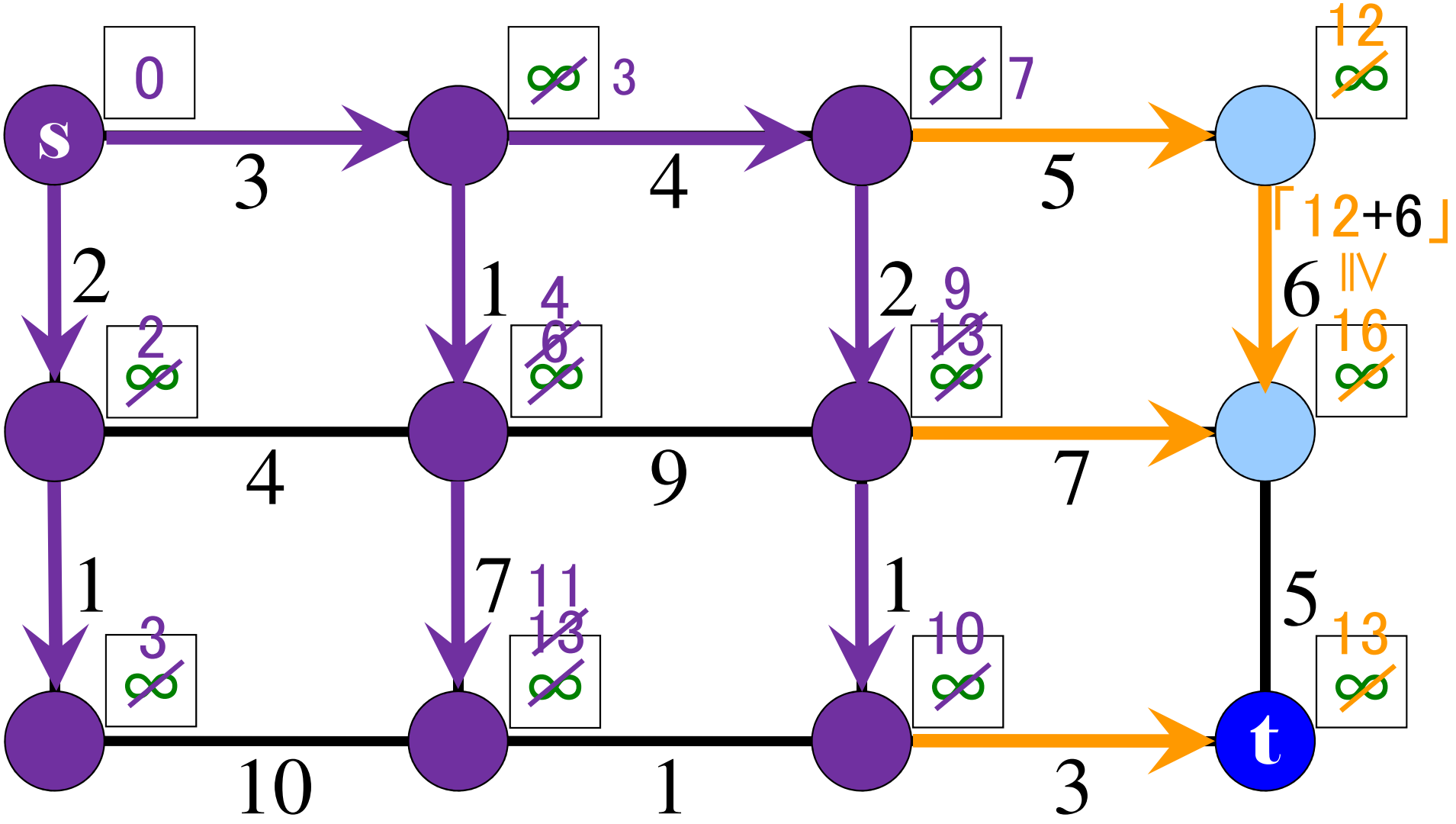
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

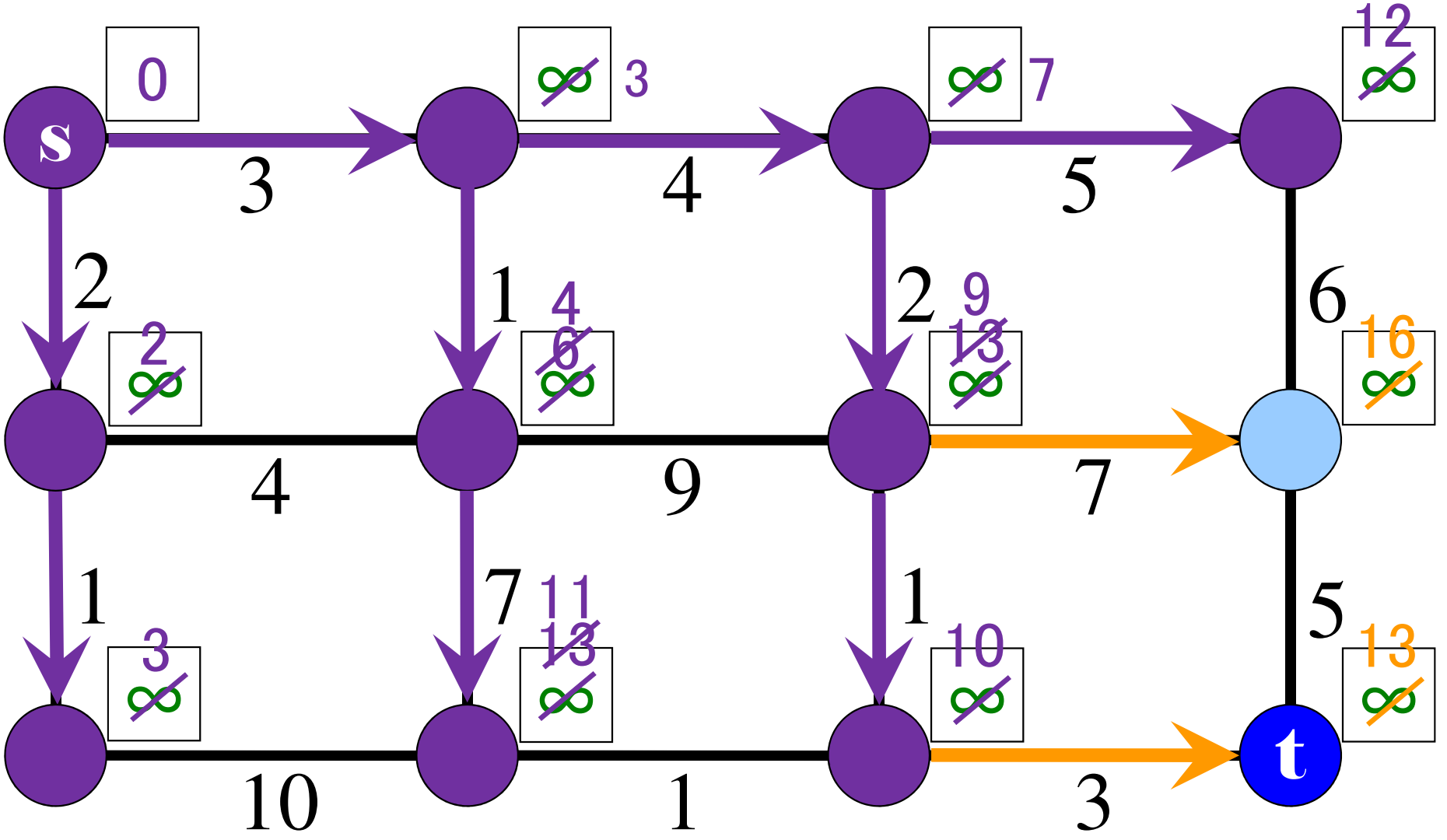


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

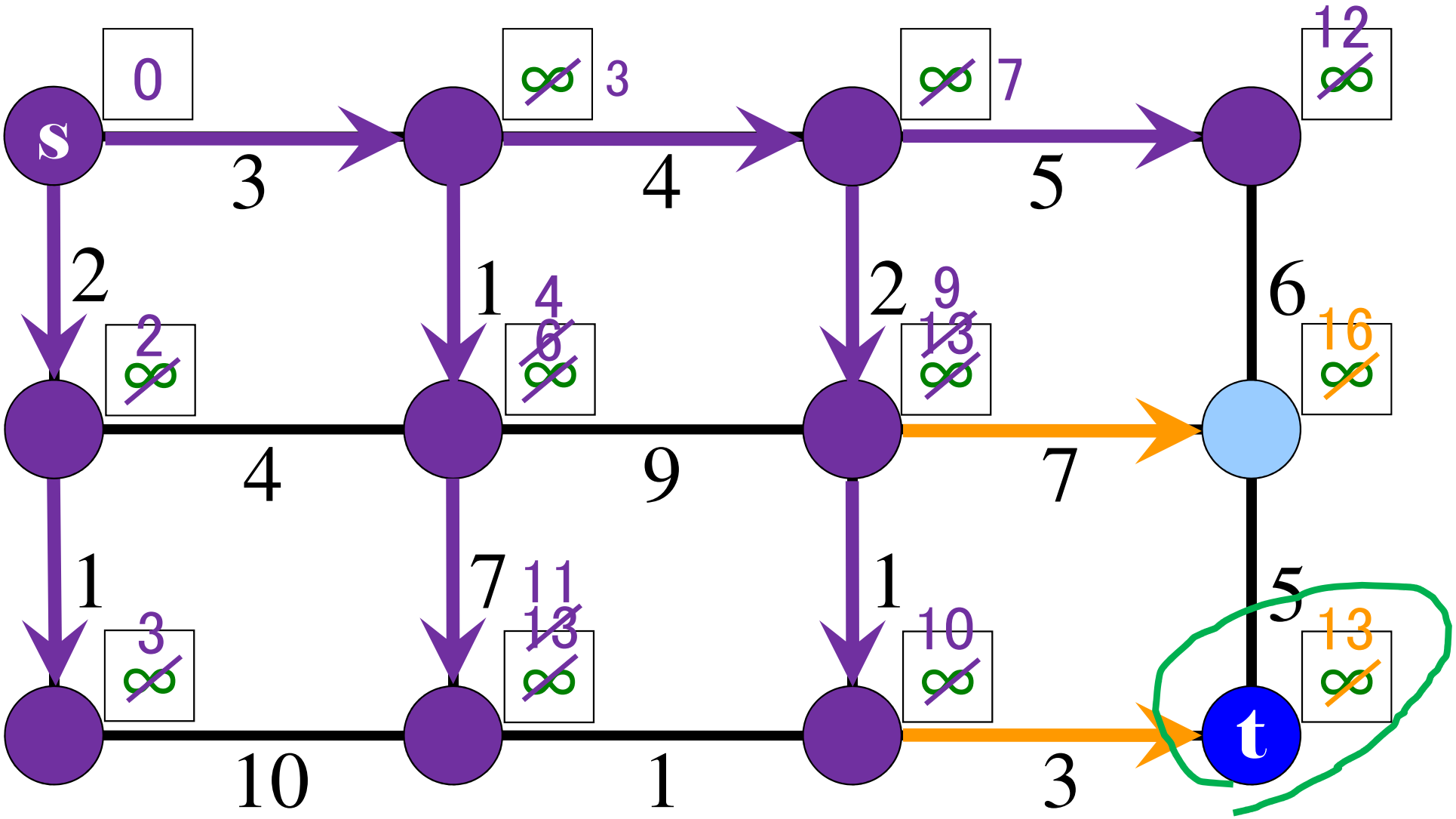
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



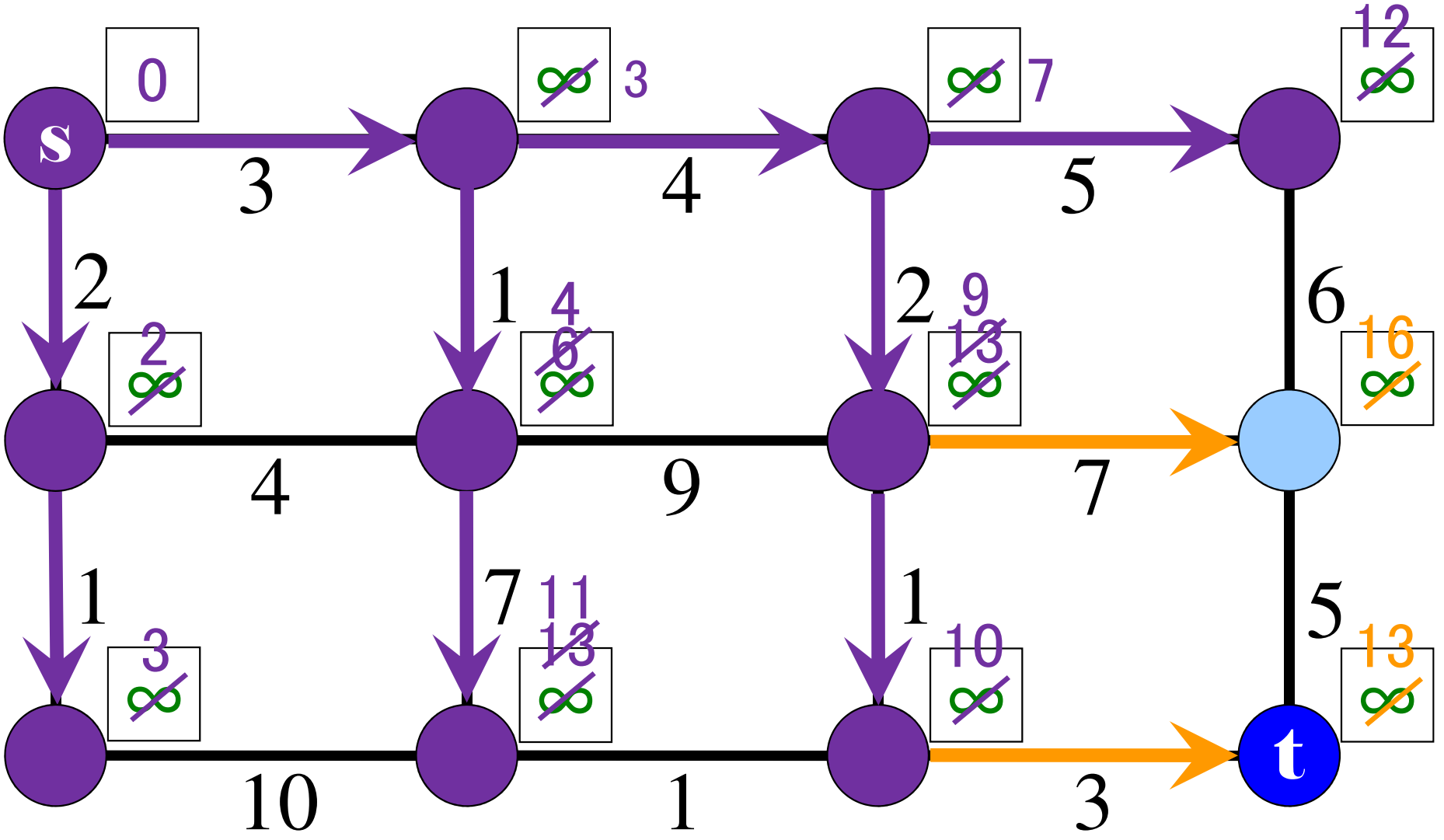
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

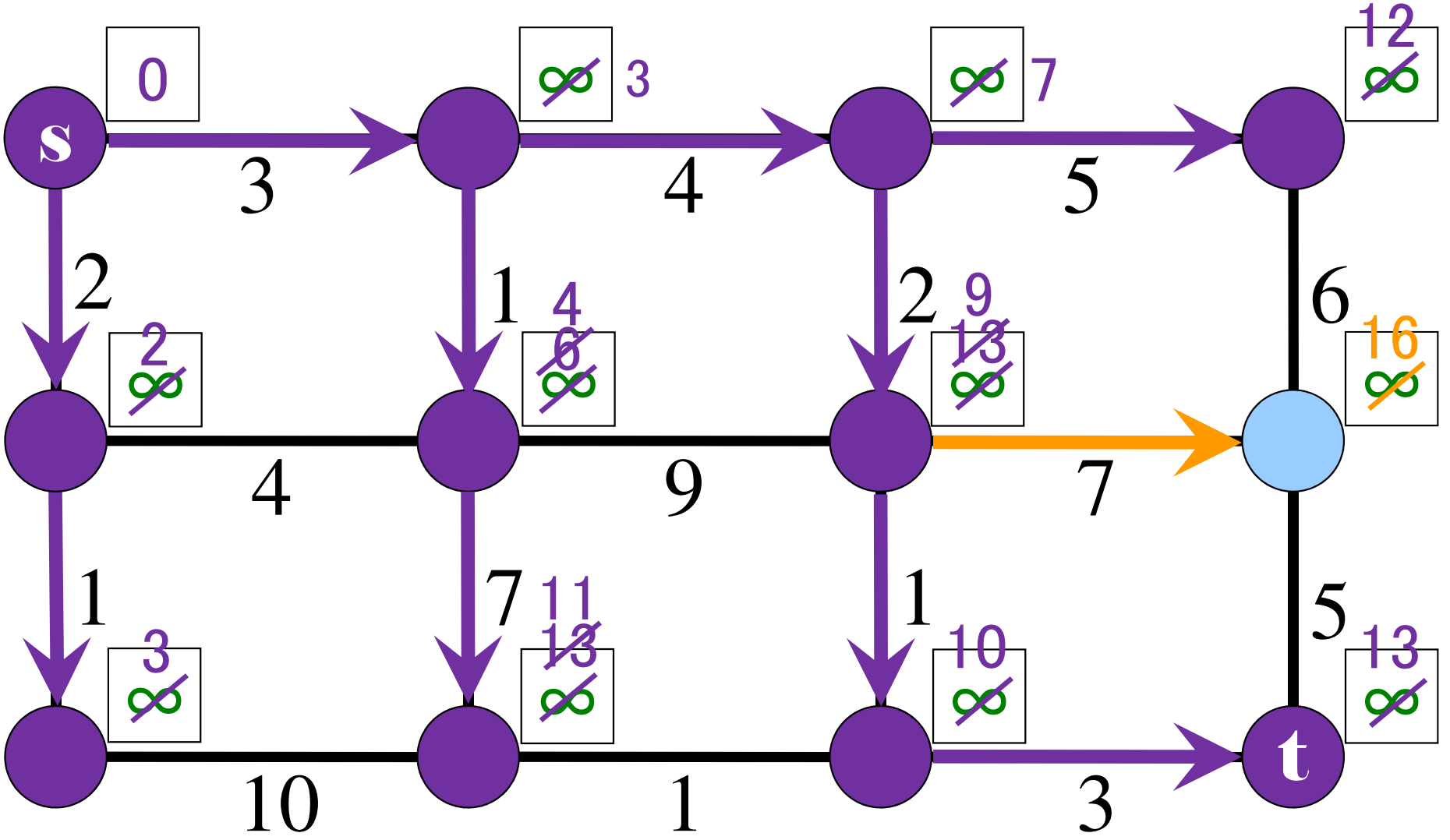


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

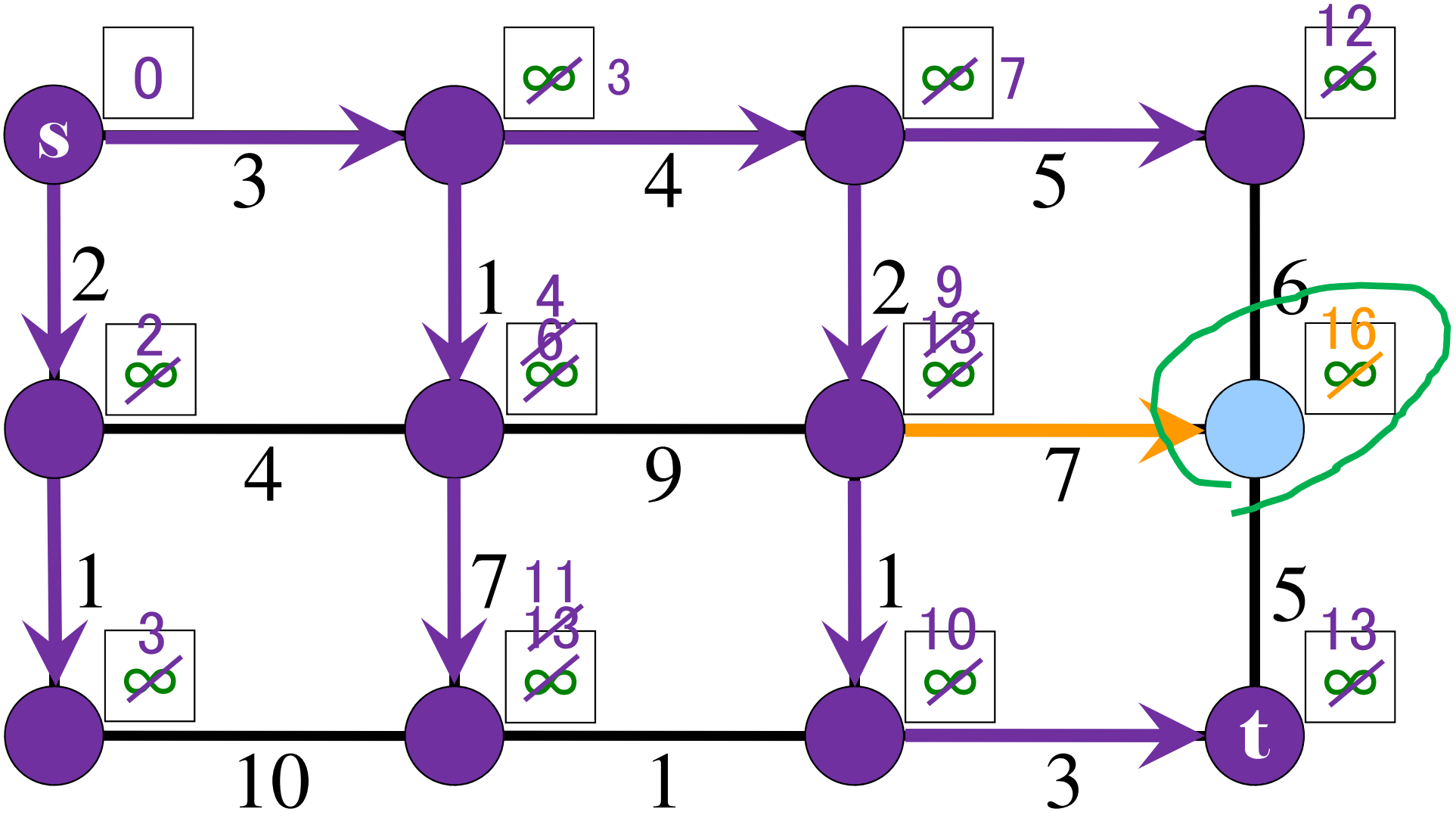
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



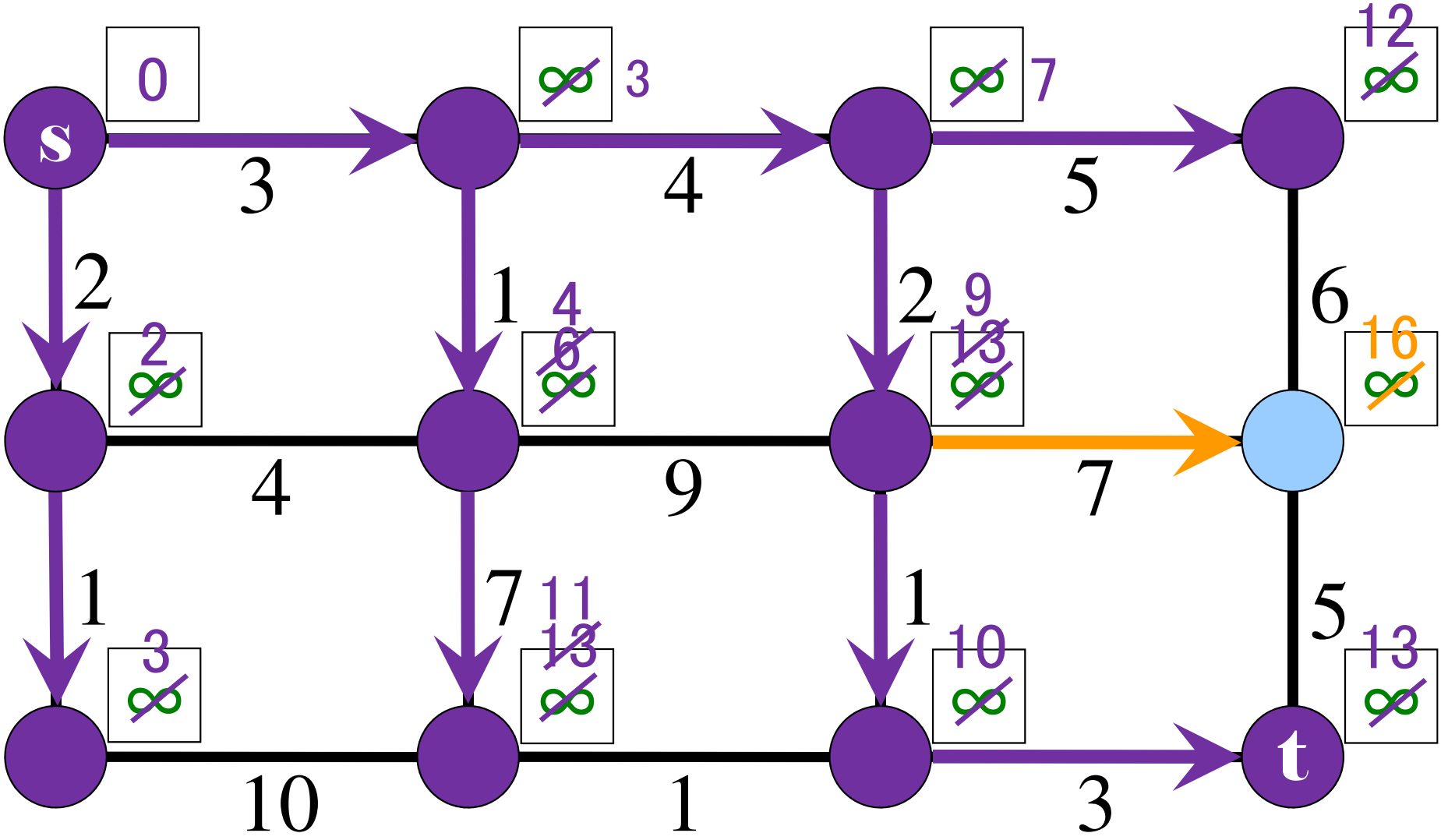
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

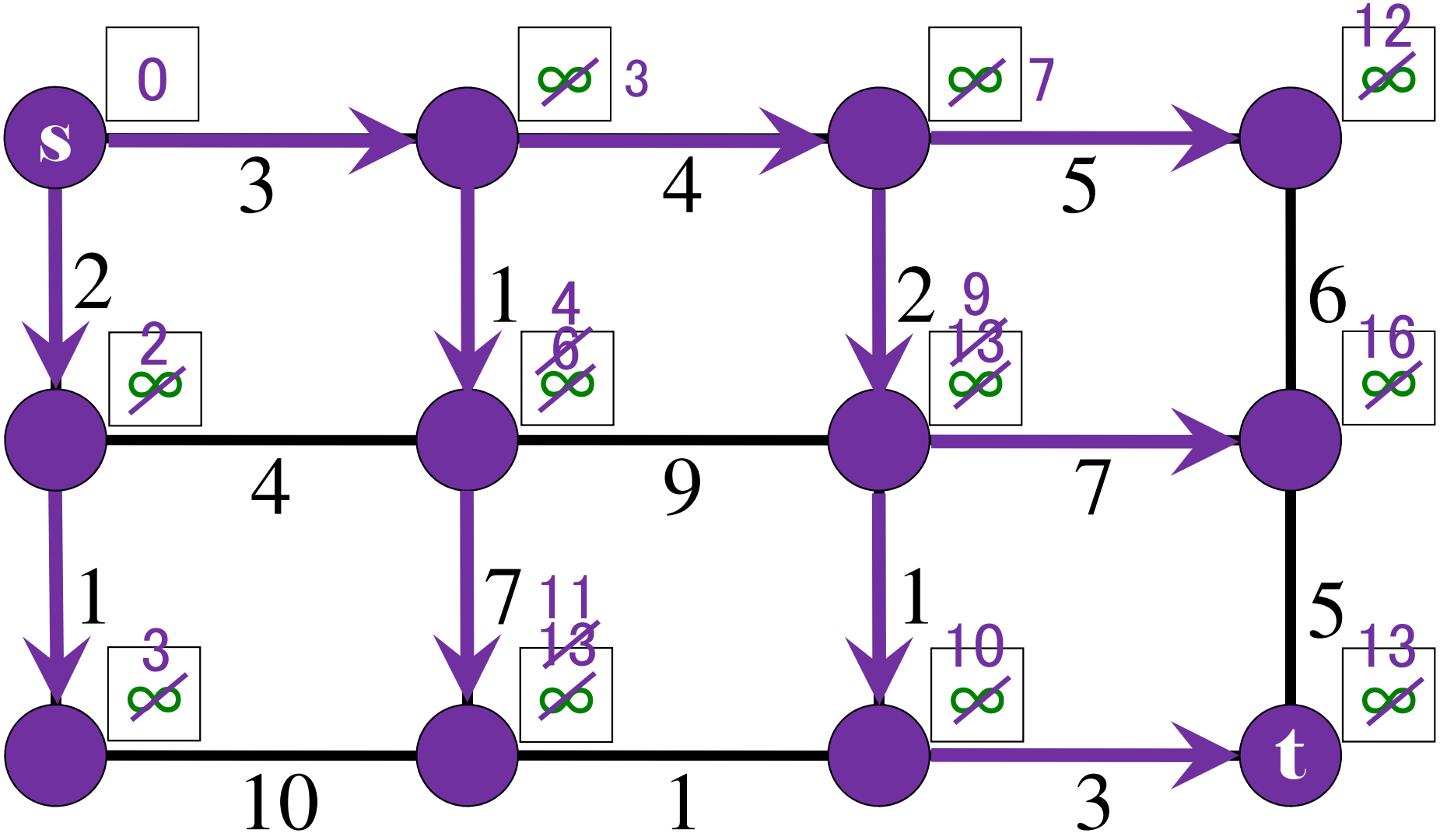


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

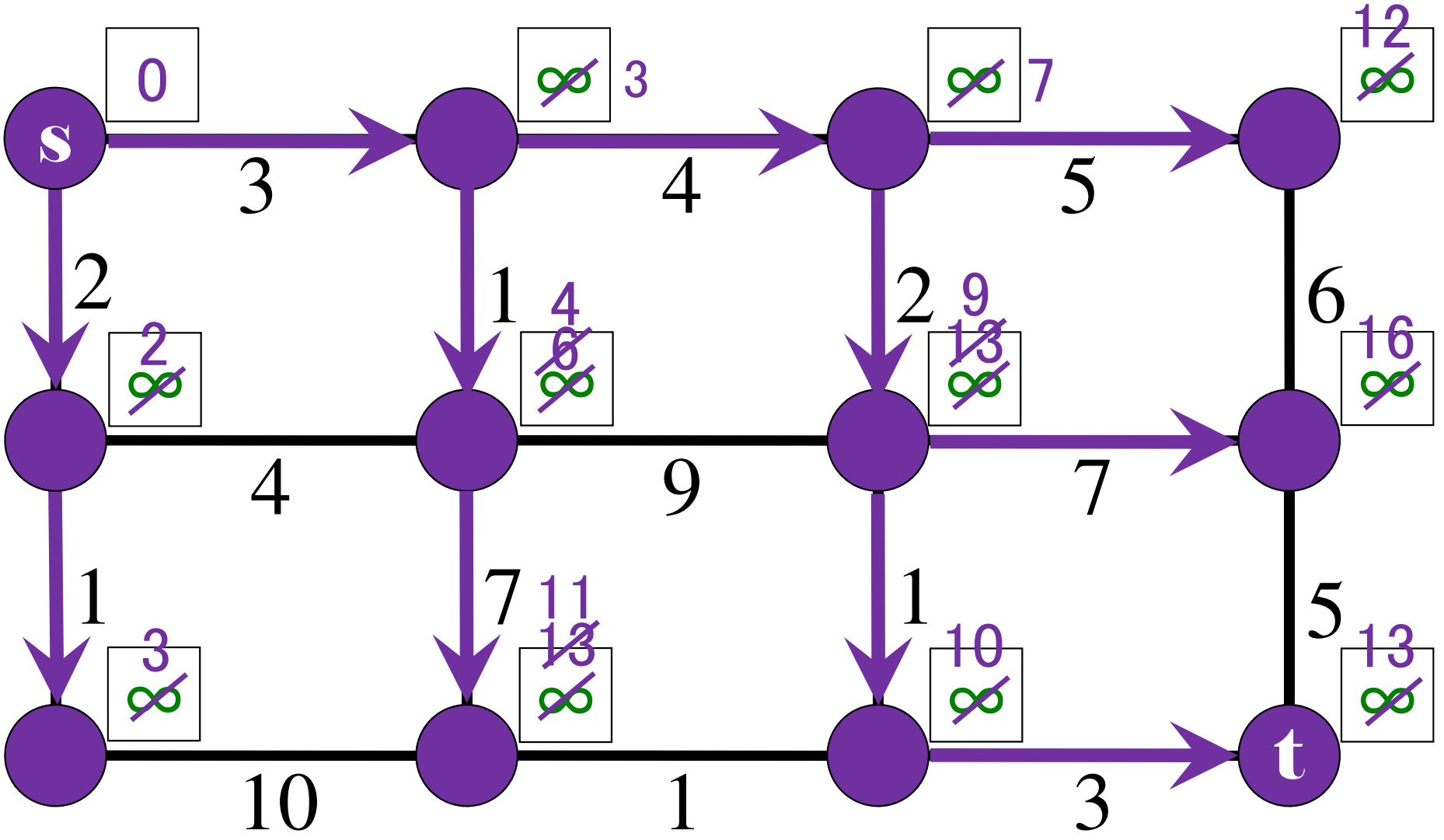
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



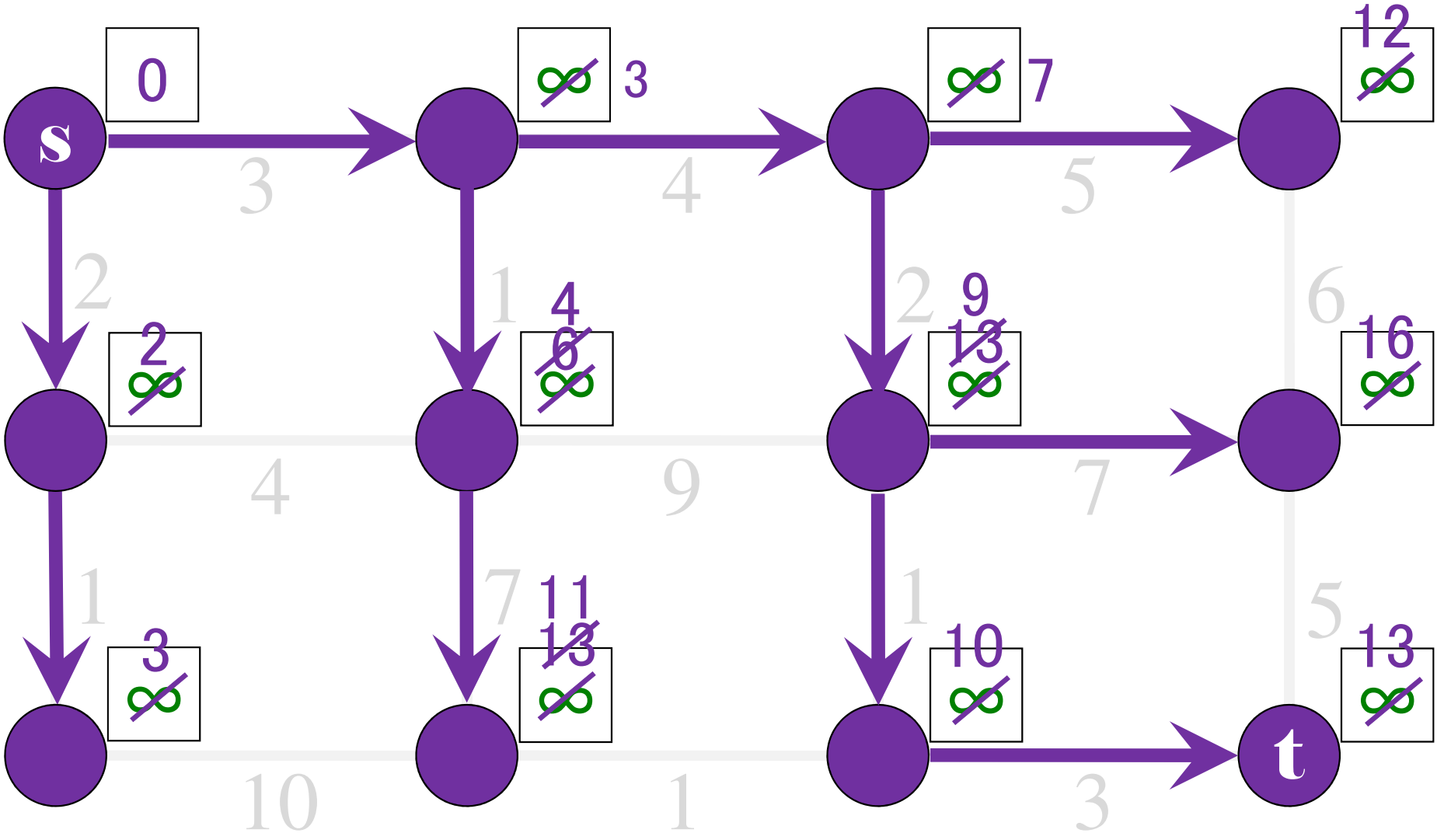
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



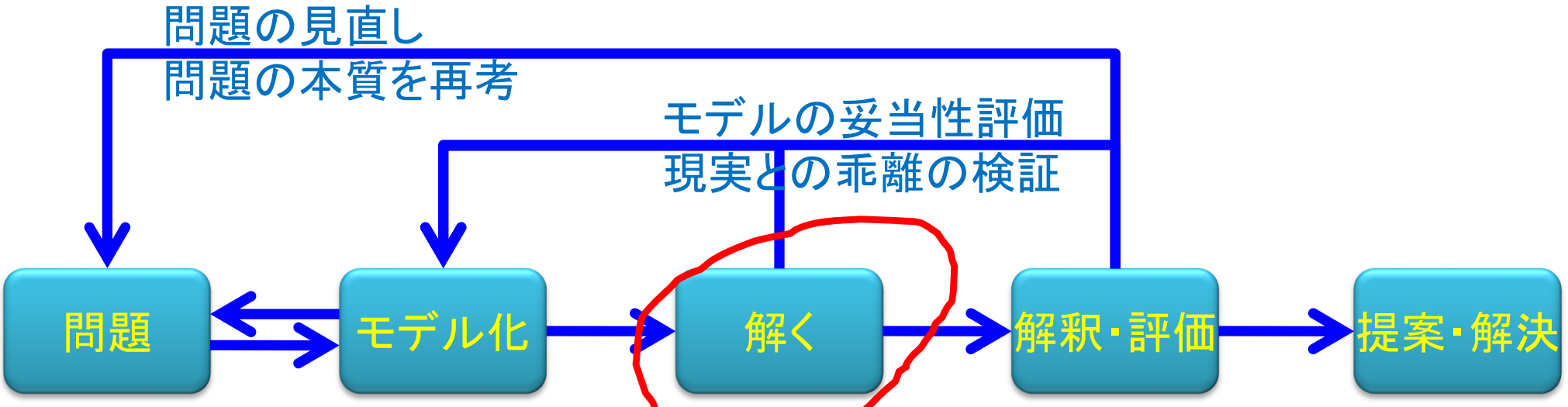
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

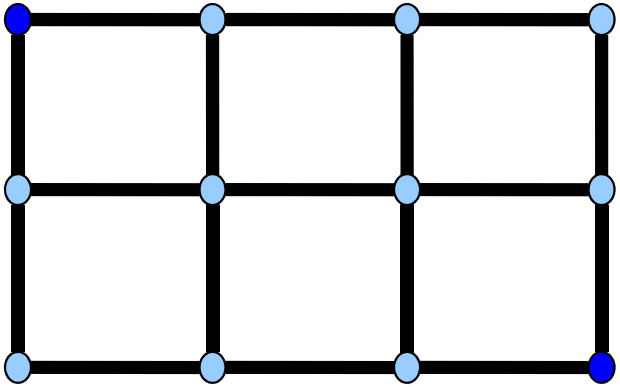
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

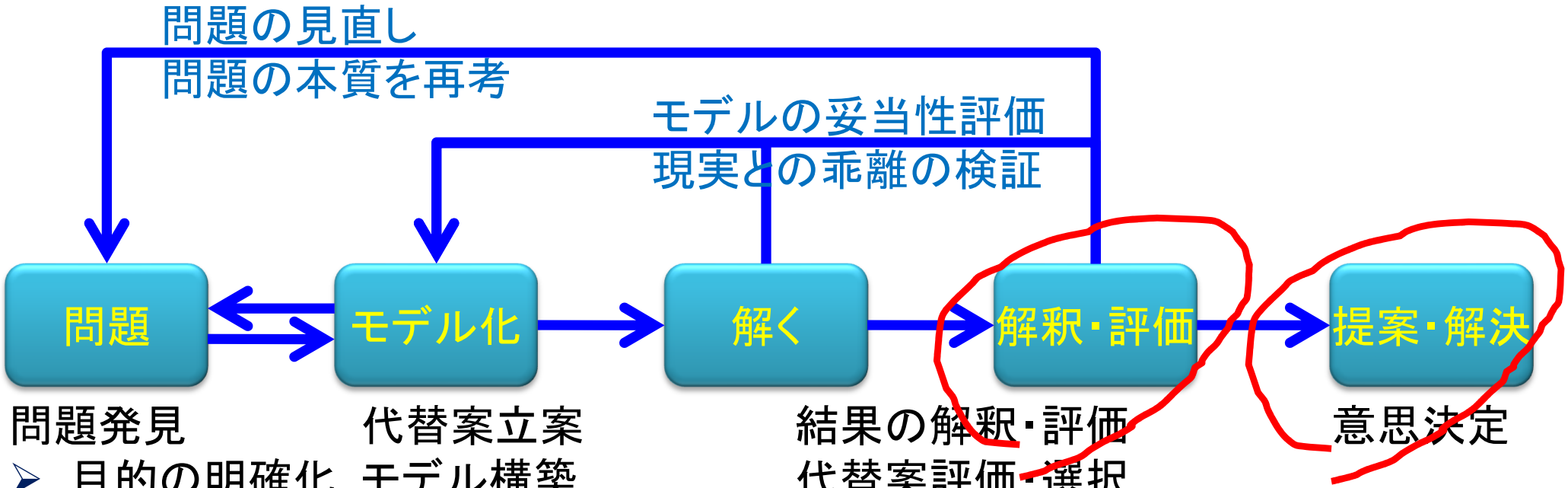
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

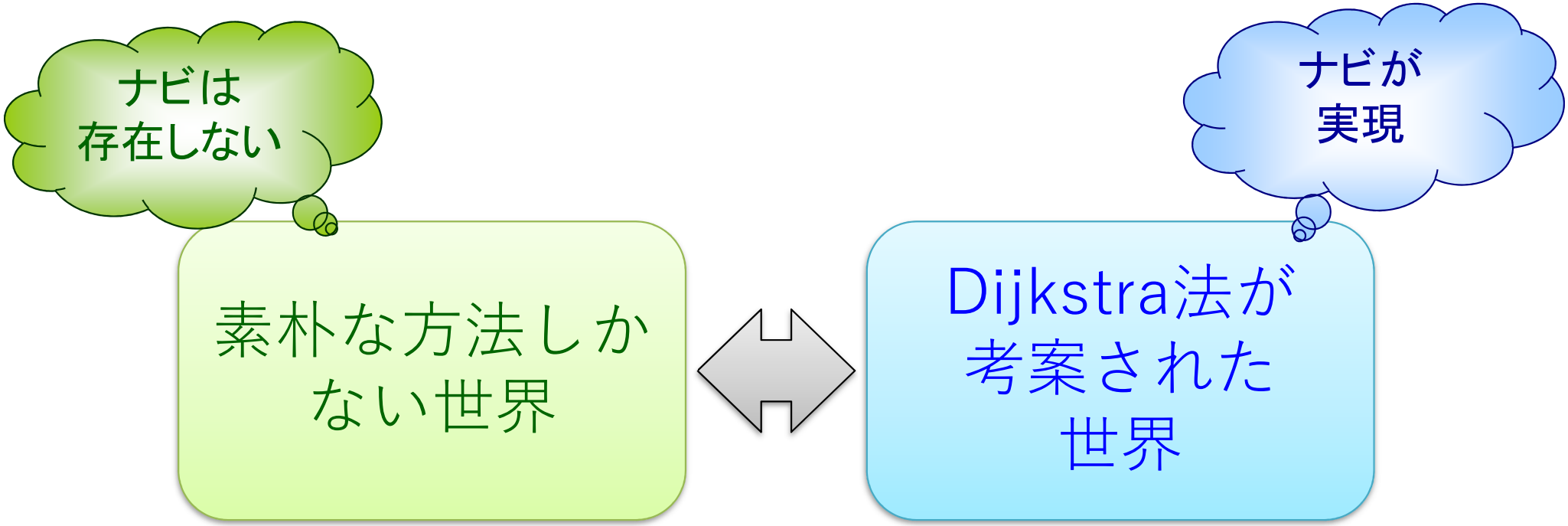
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

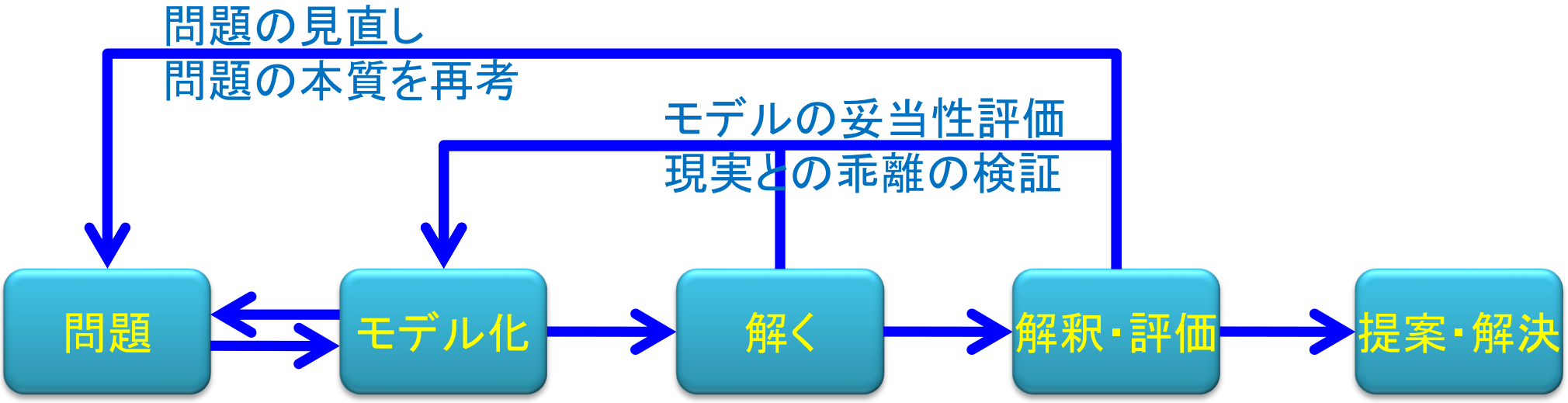
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



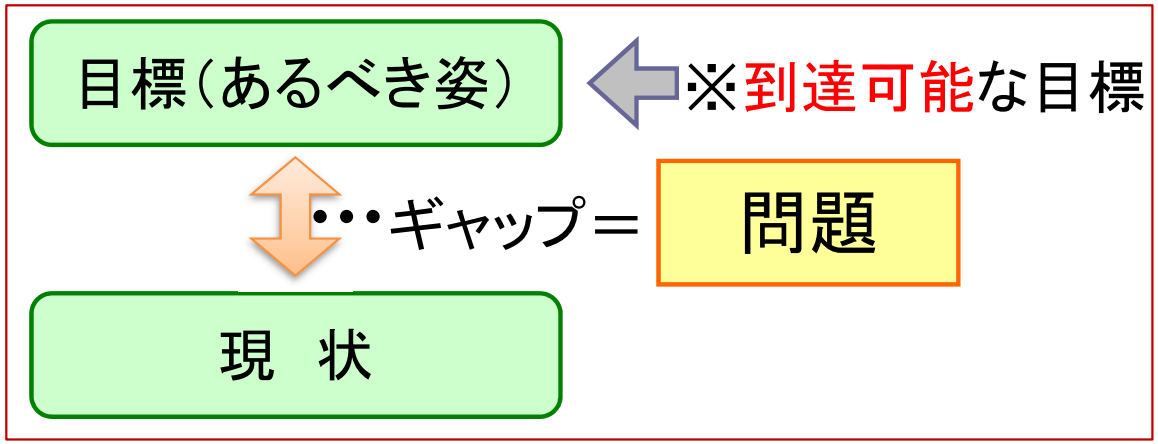
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

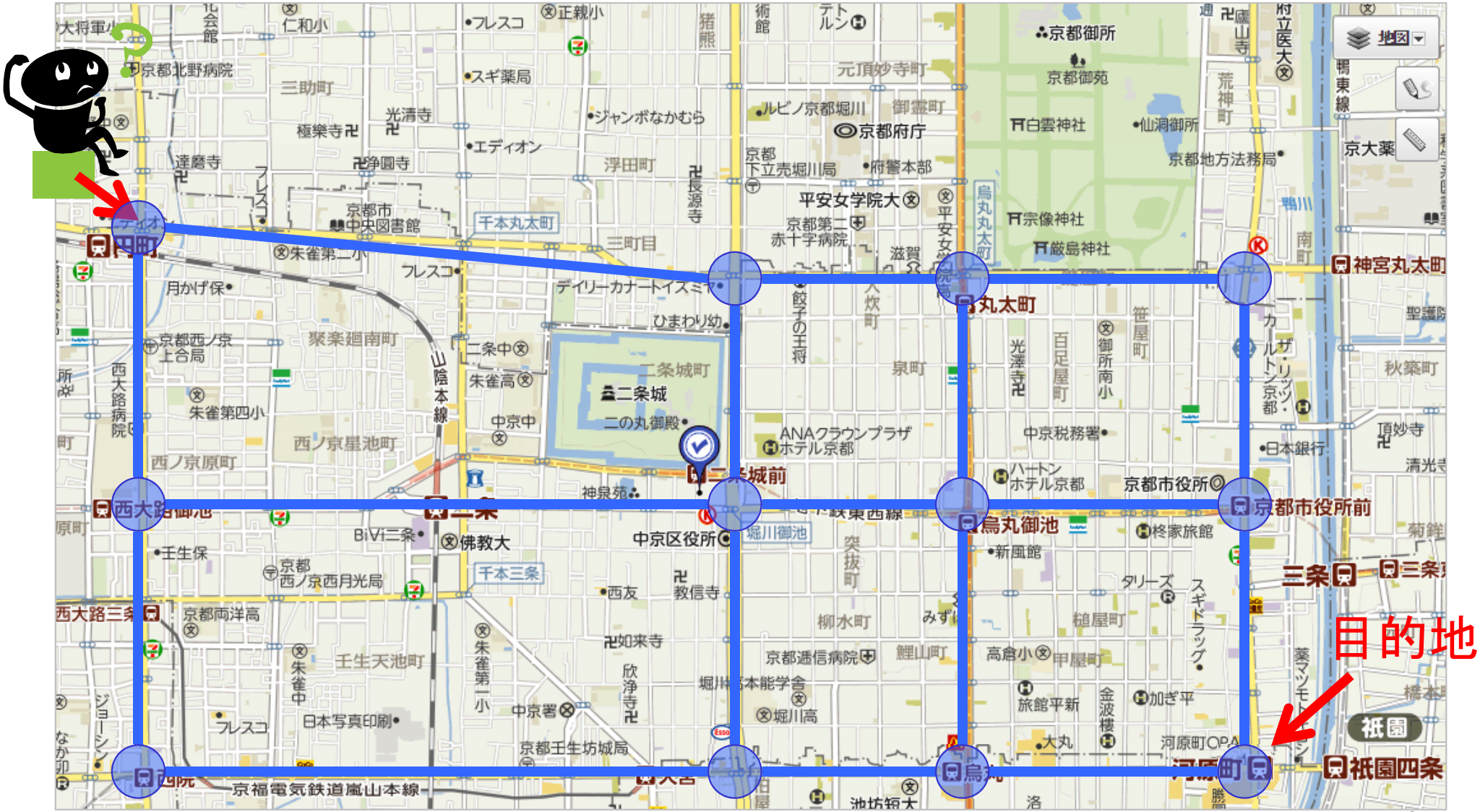
意思決定

問題の定義



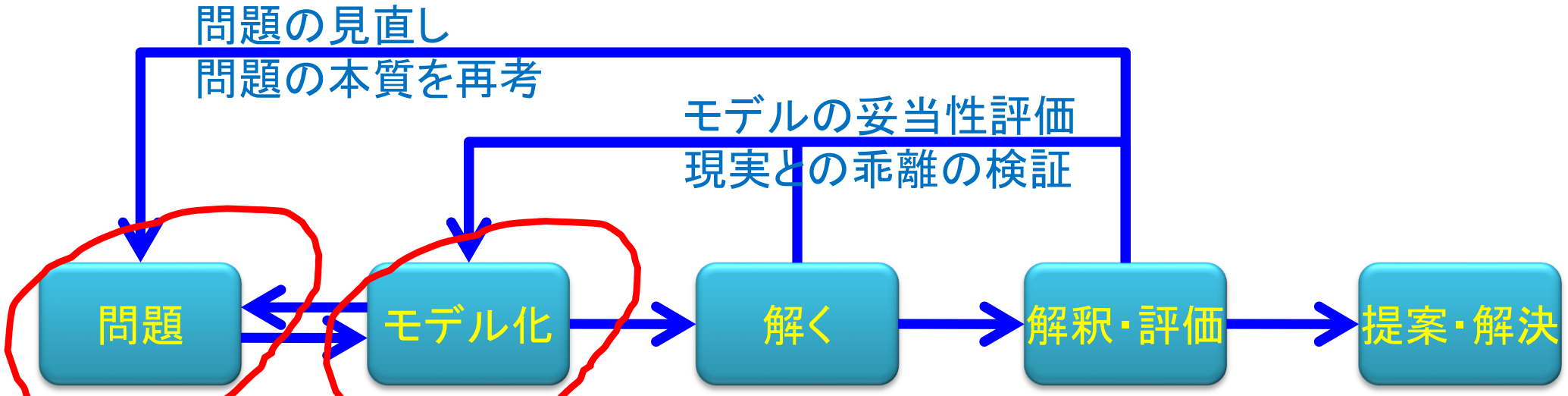
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

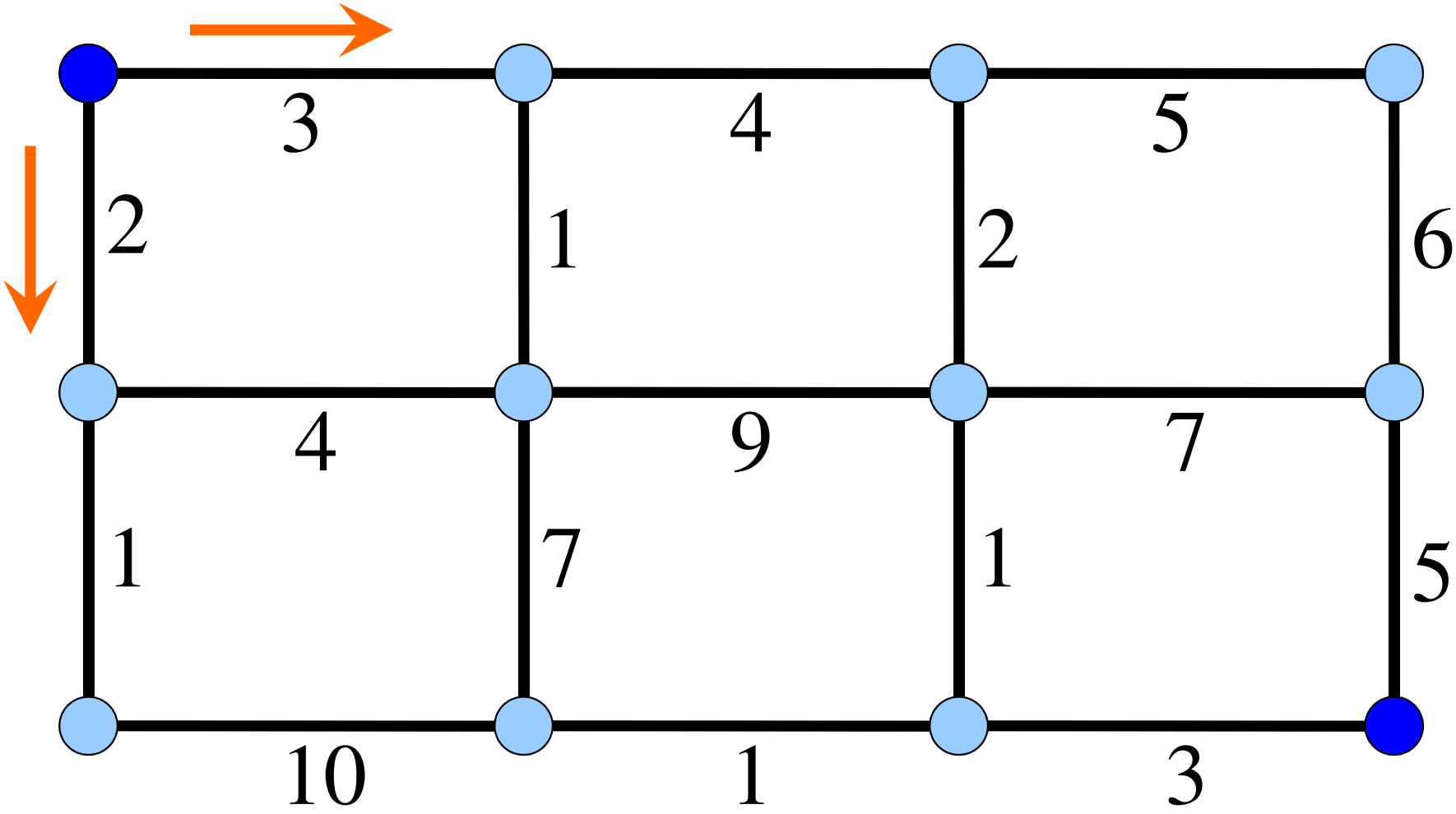
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

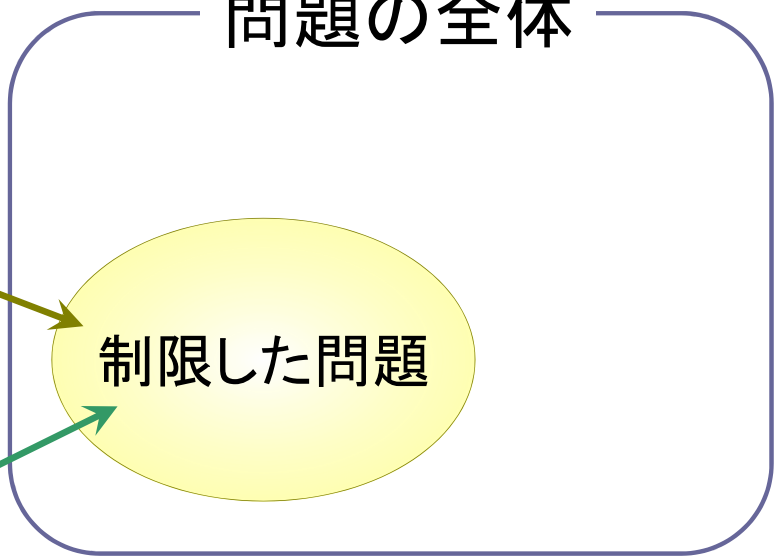


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

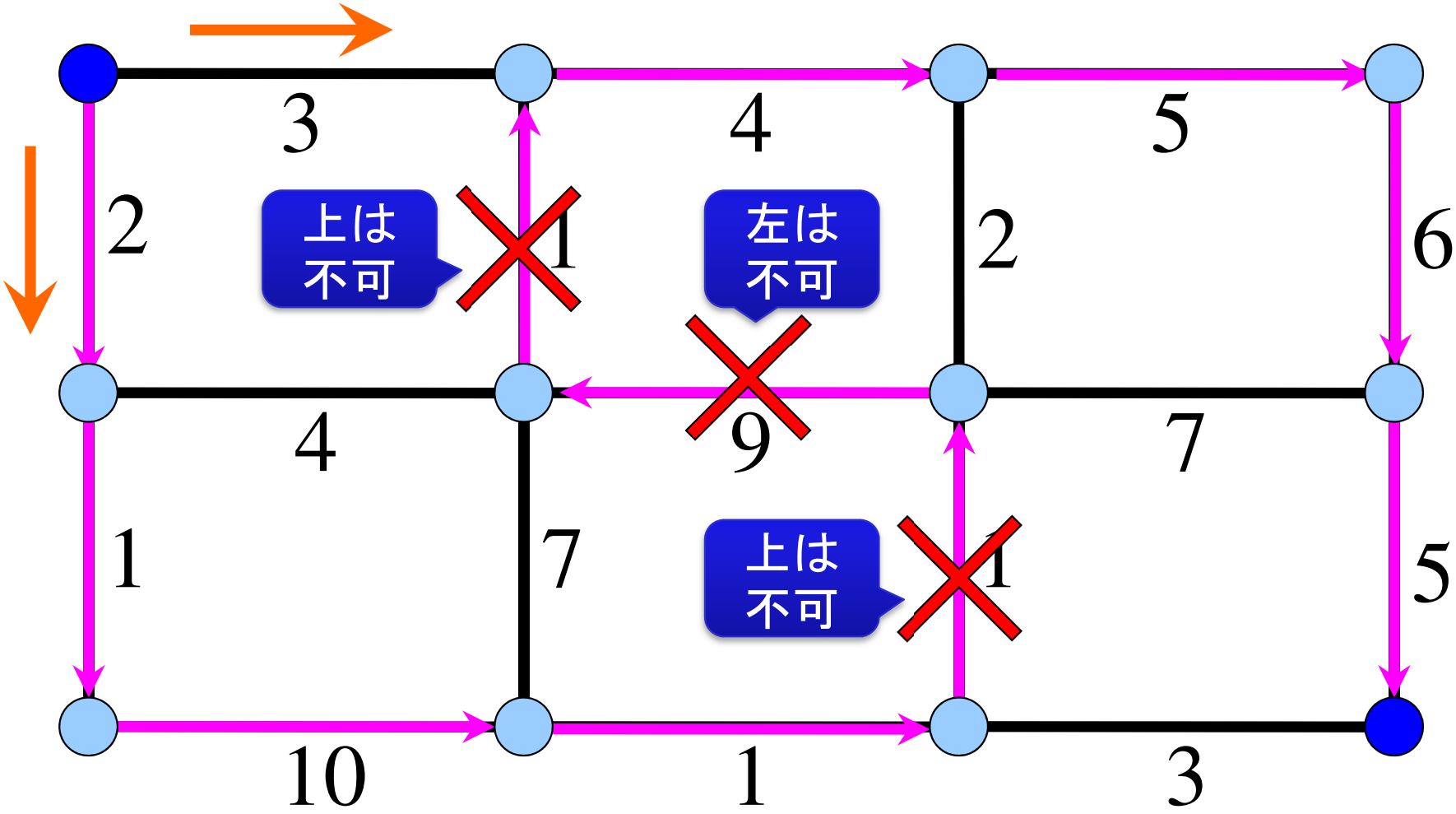
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

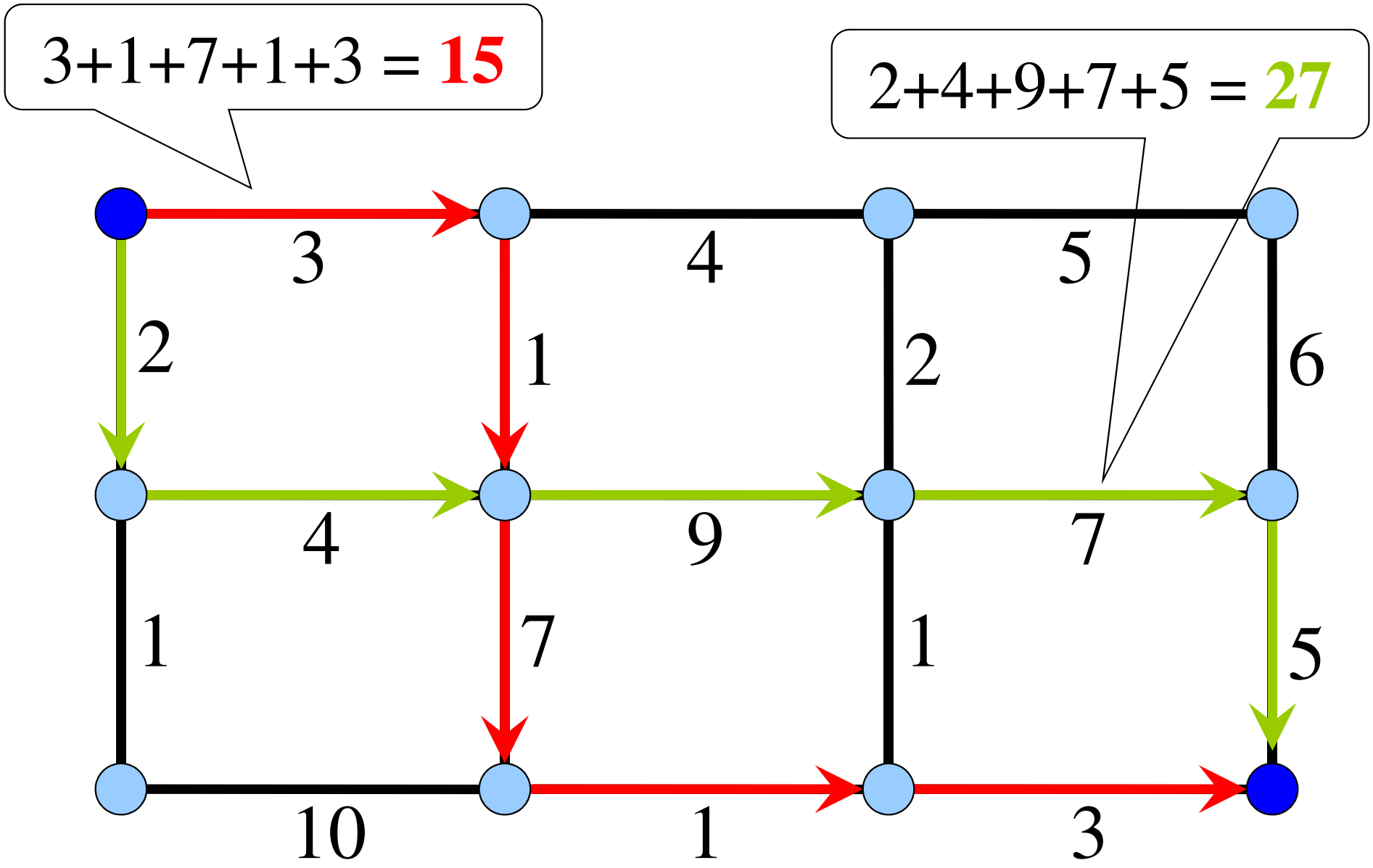
難しいなら易しくすればいいのさ！

制限した問題

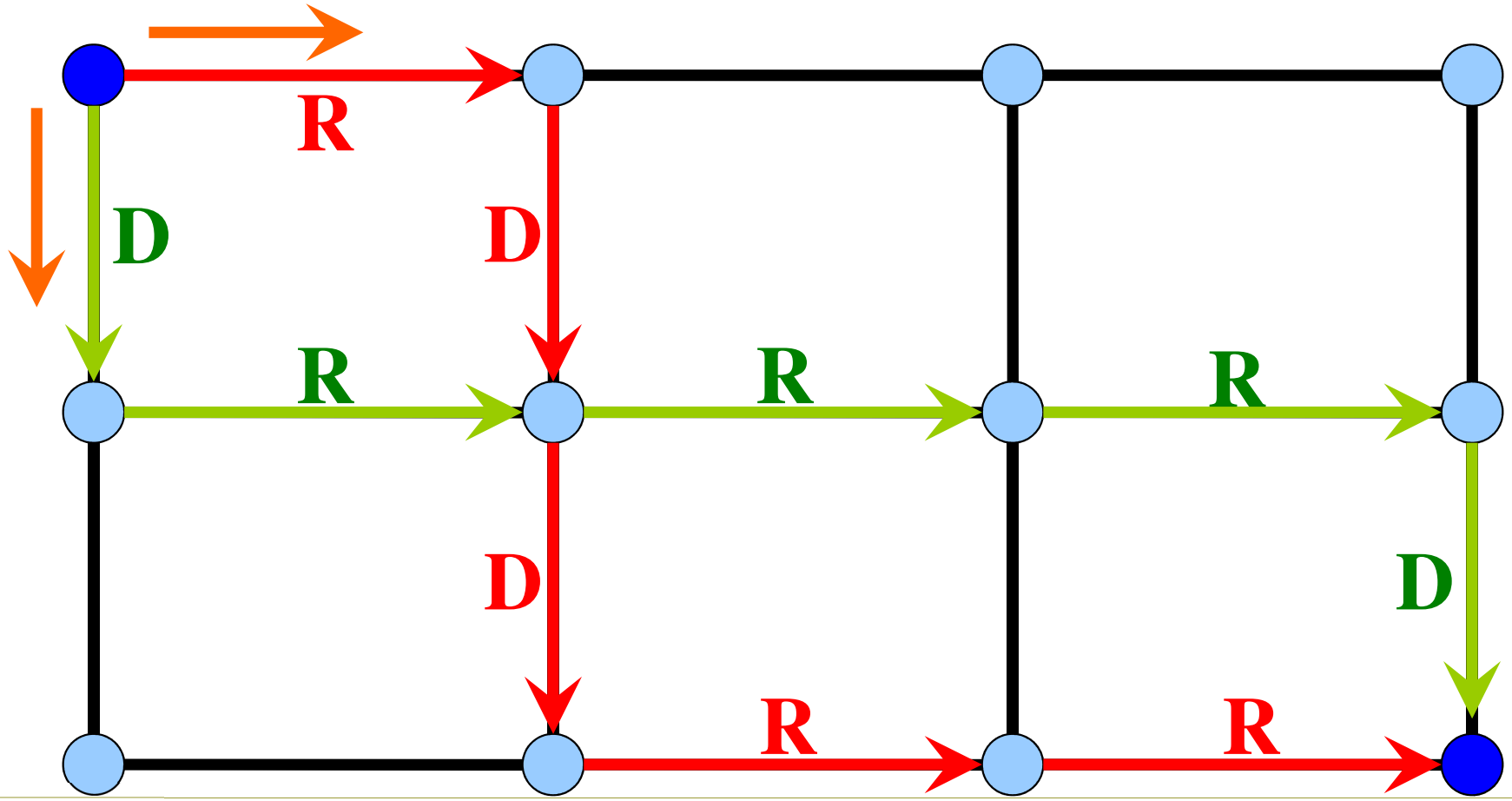
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



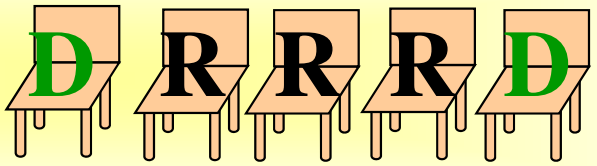
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



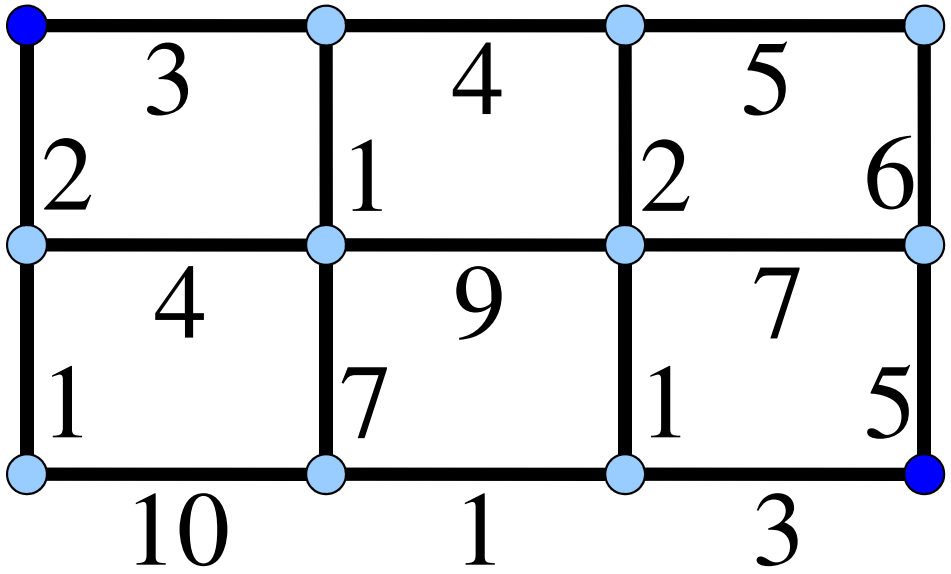
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

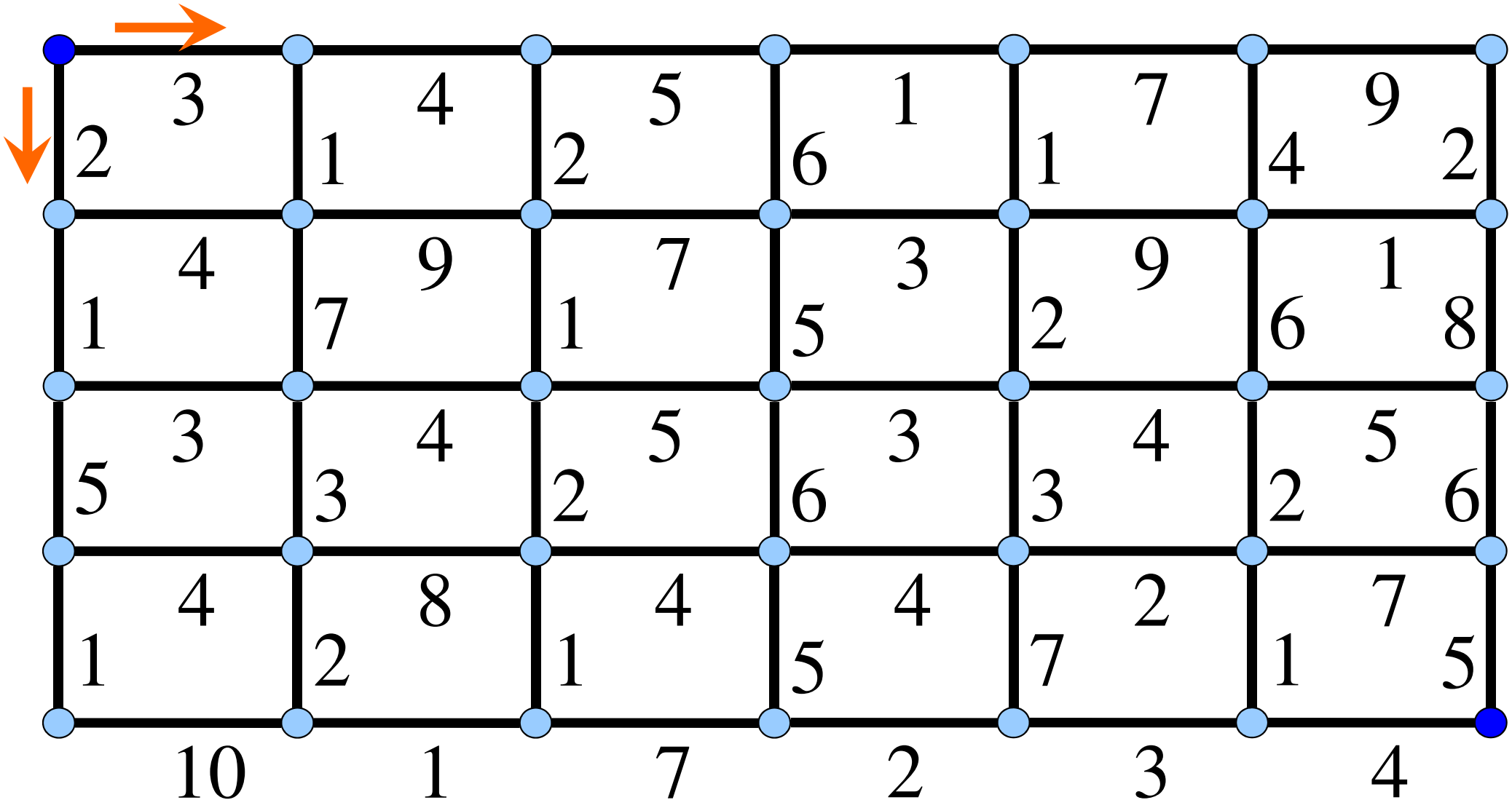
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

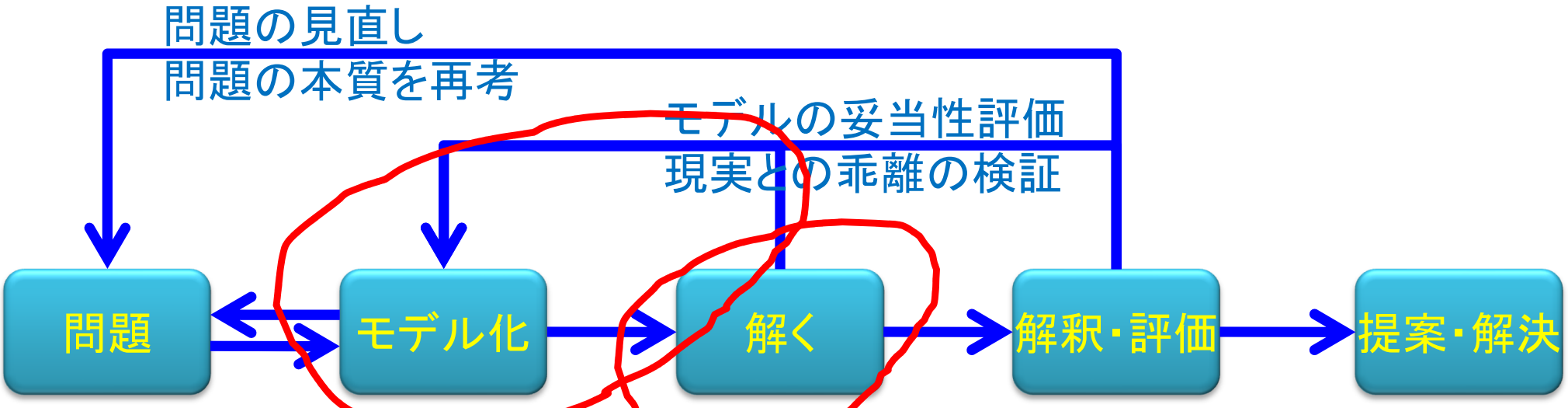
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

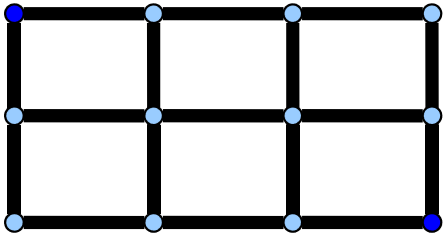
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

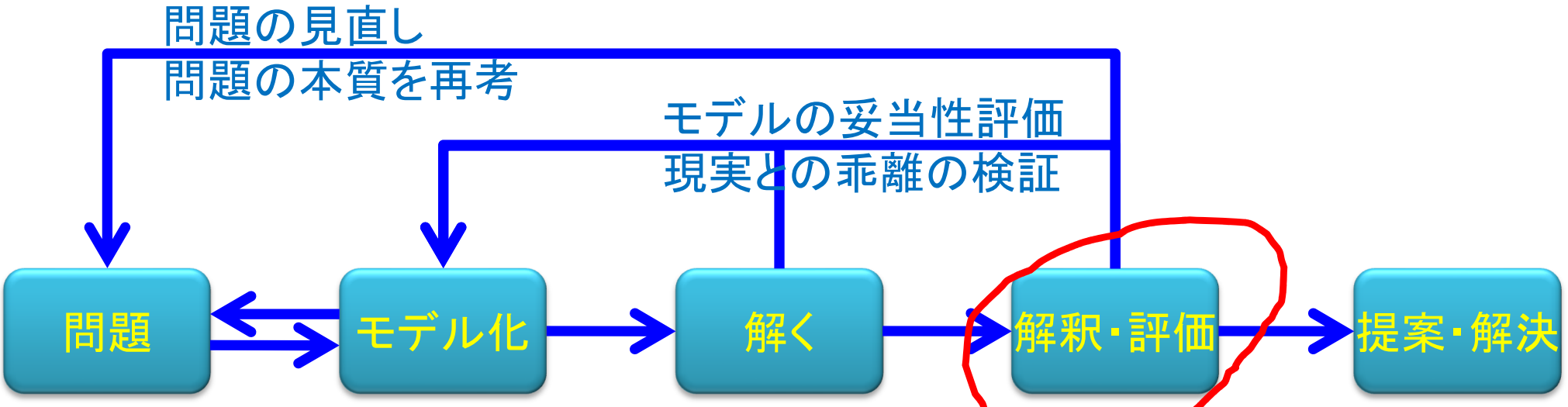
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

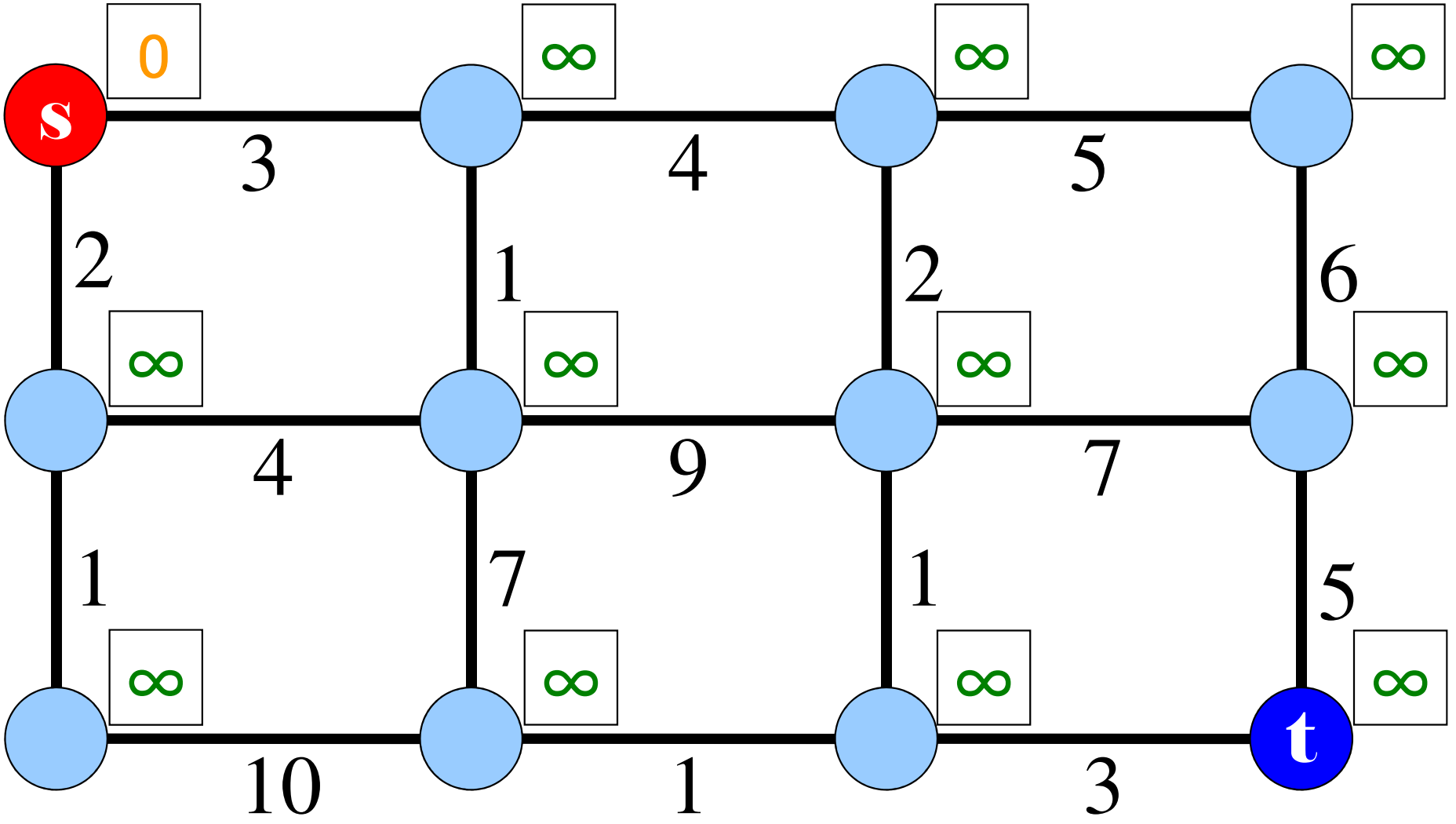
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

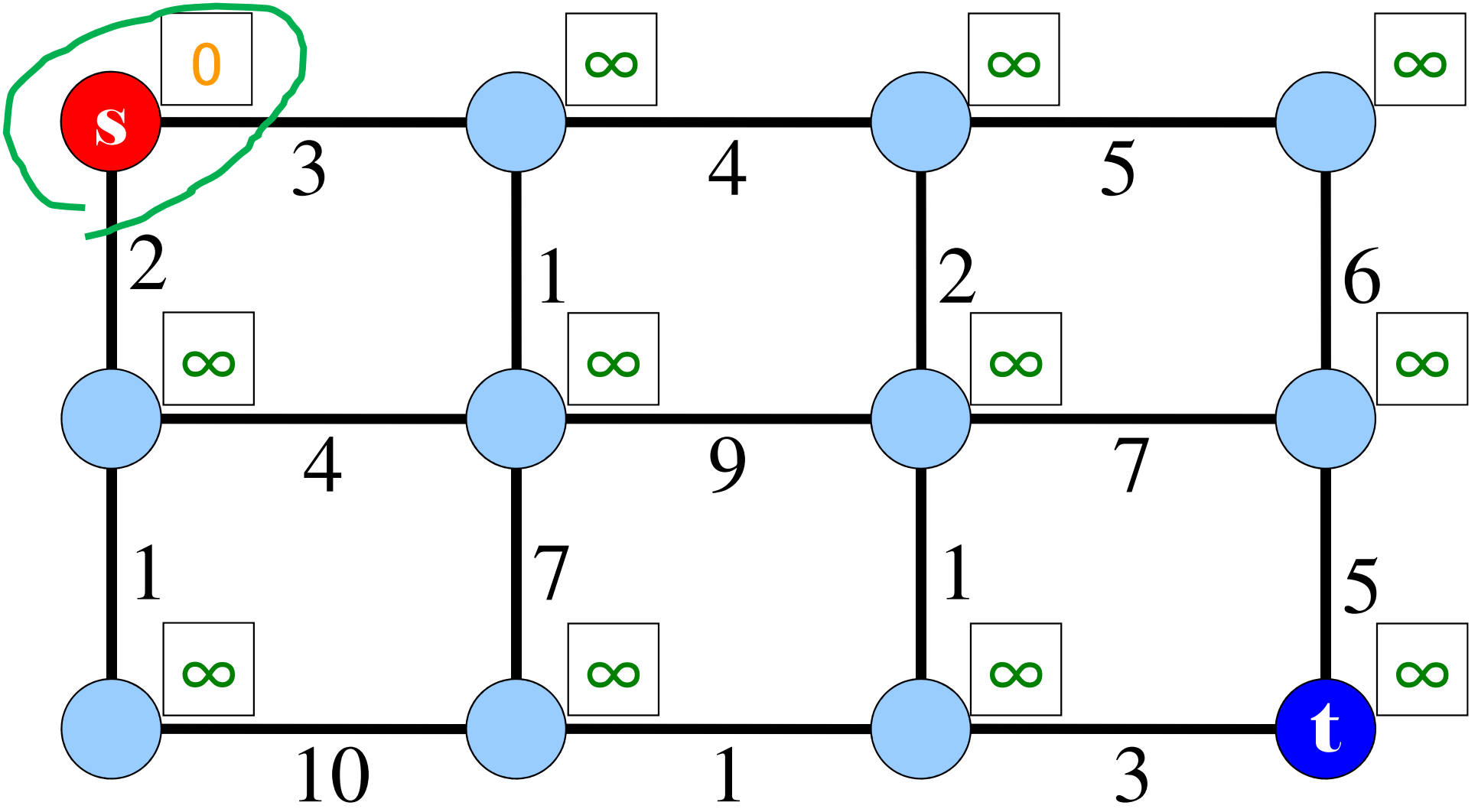
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



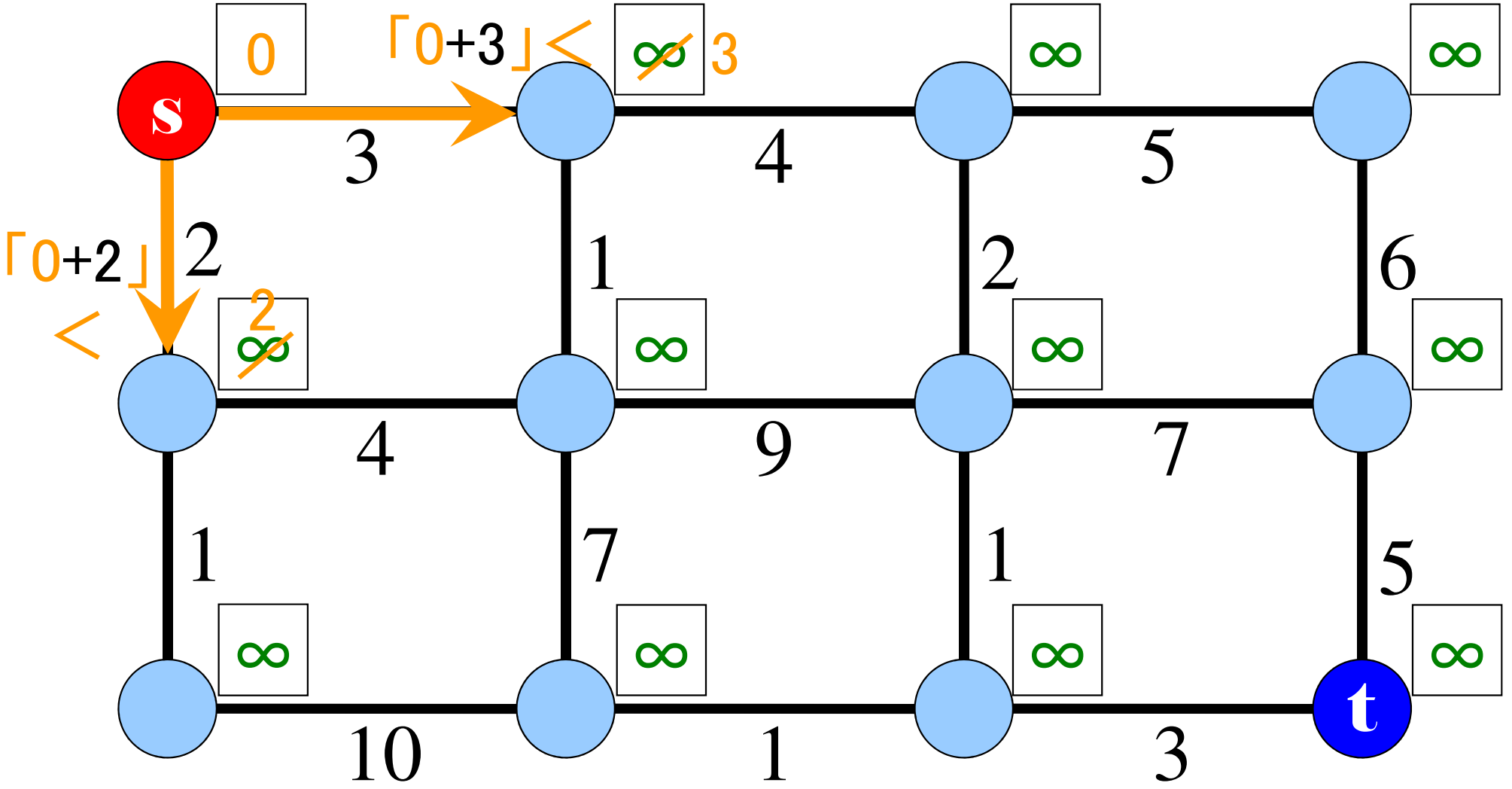
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

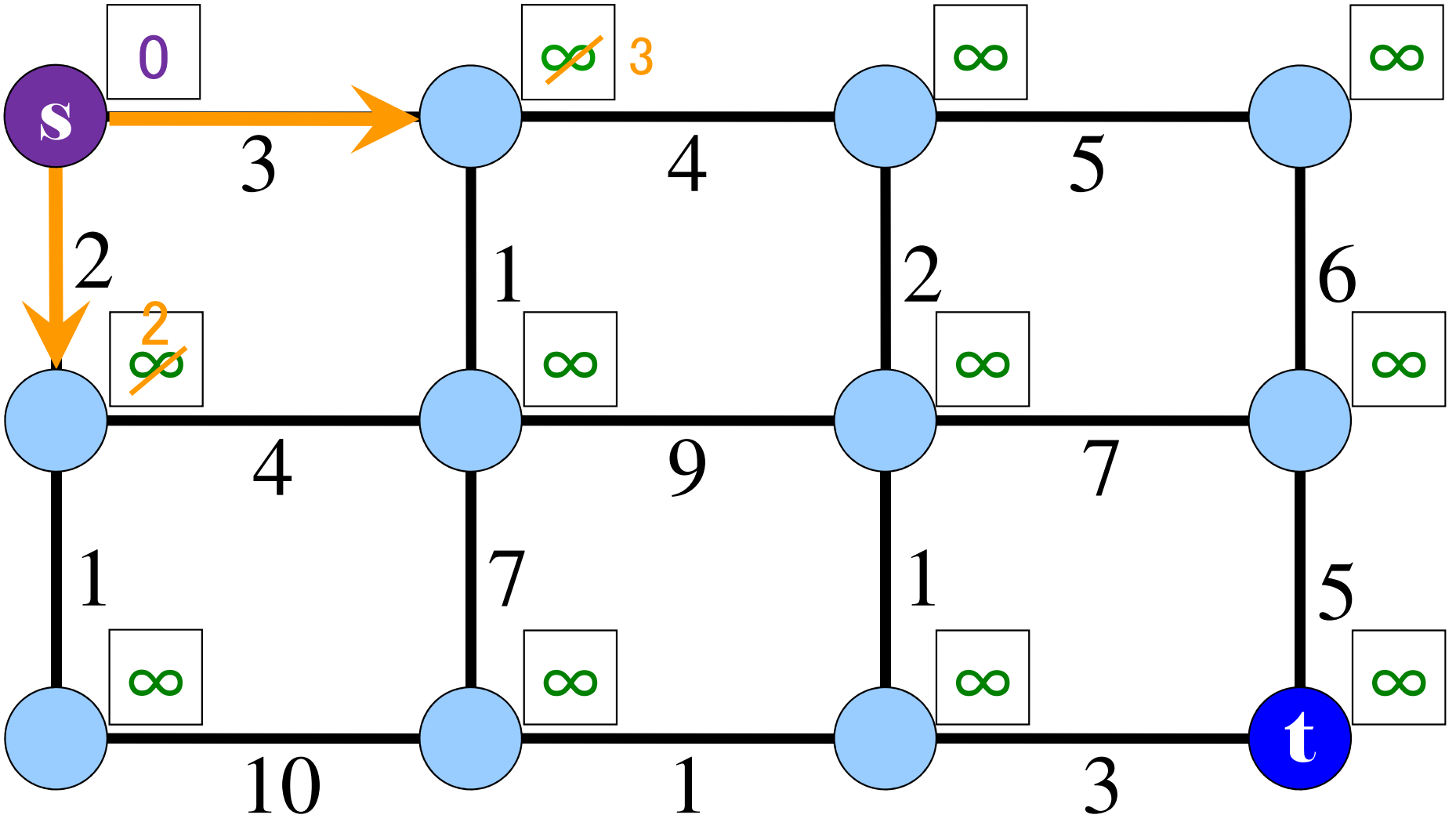


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

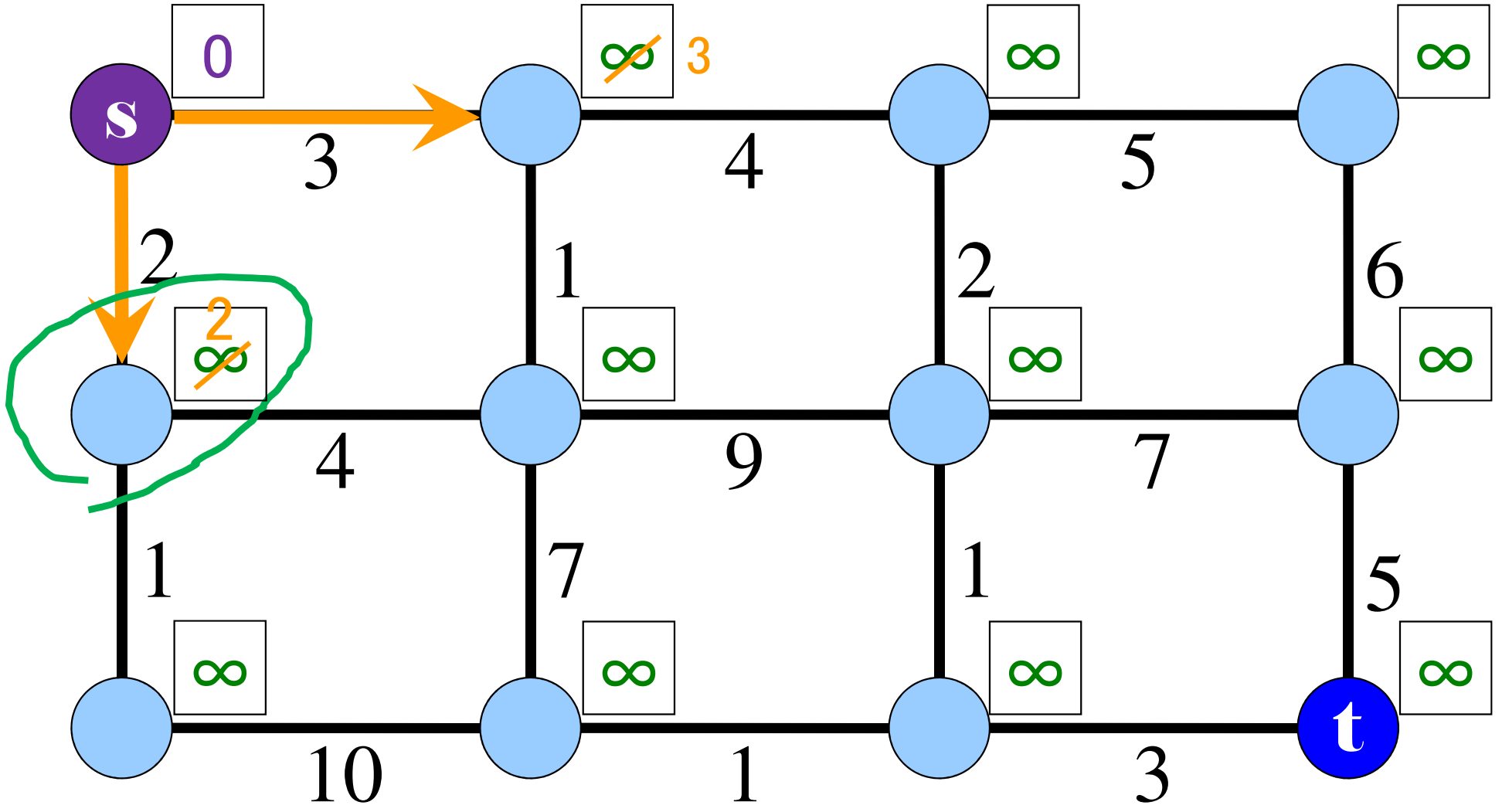
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



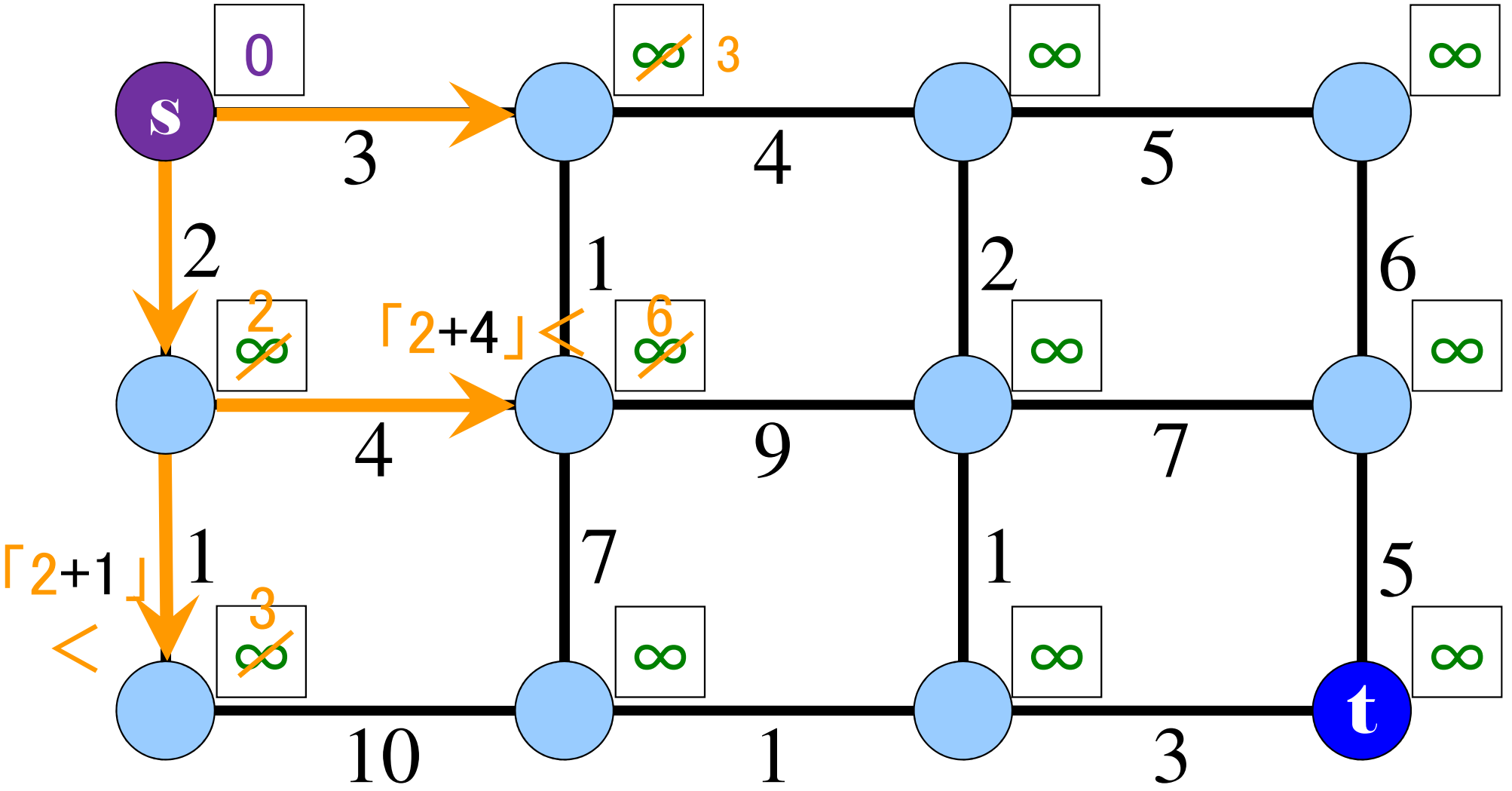
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

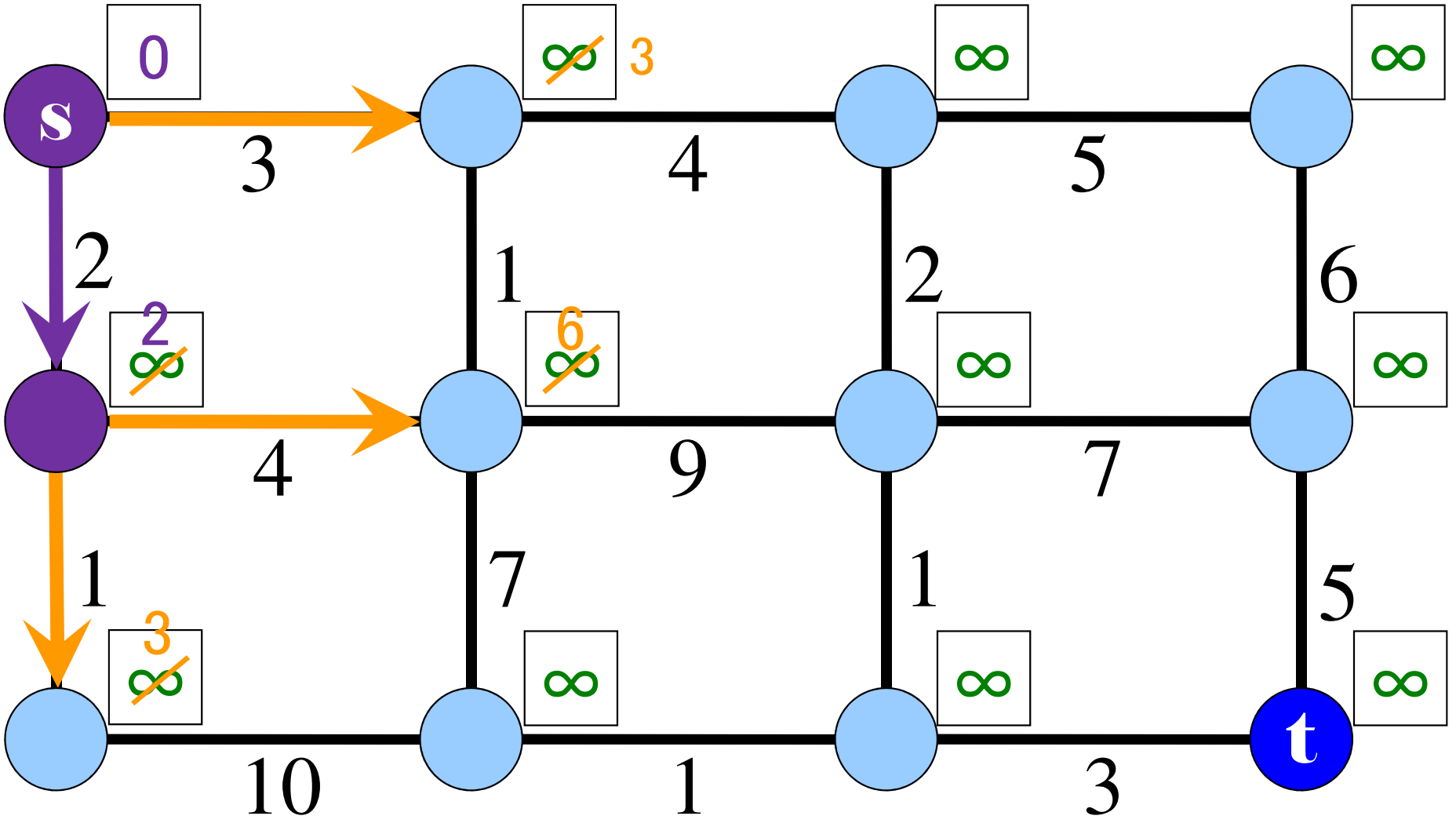


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

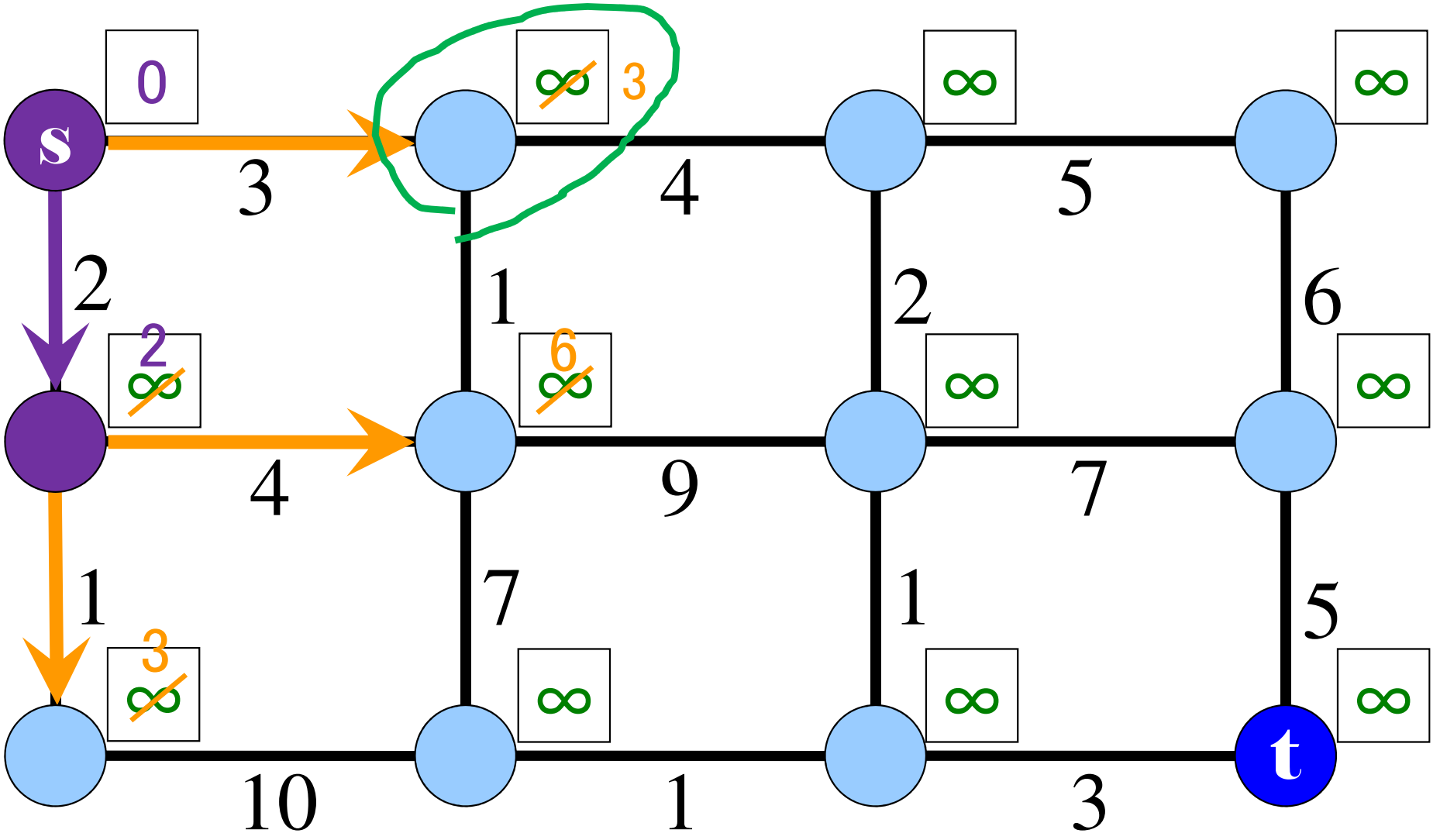
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



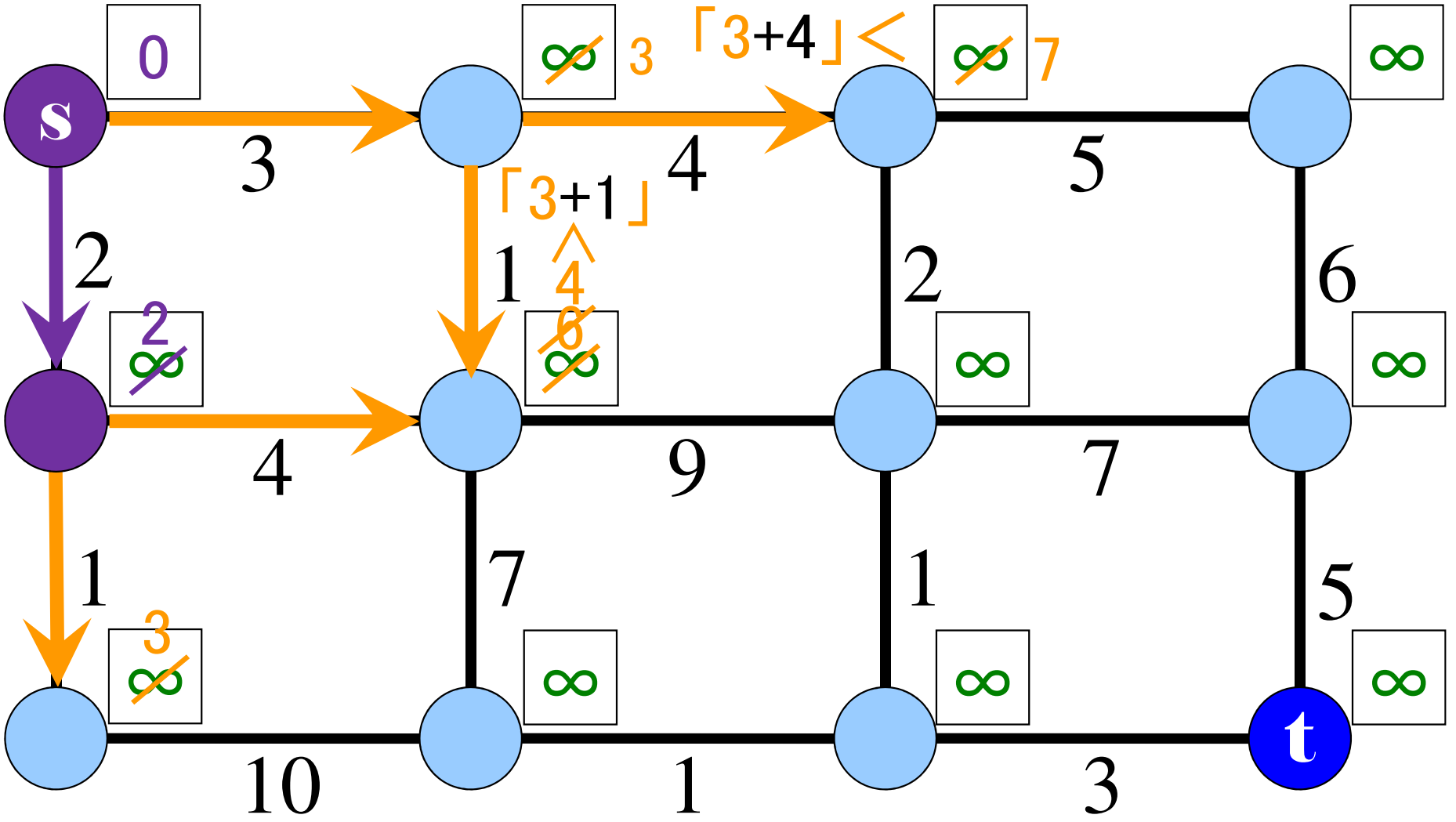
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

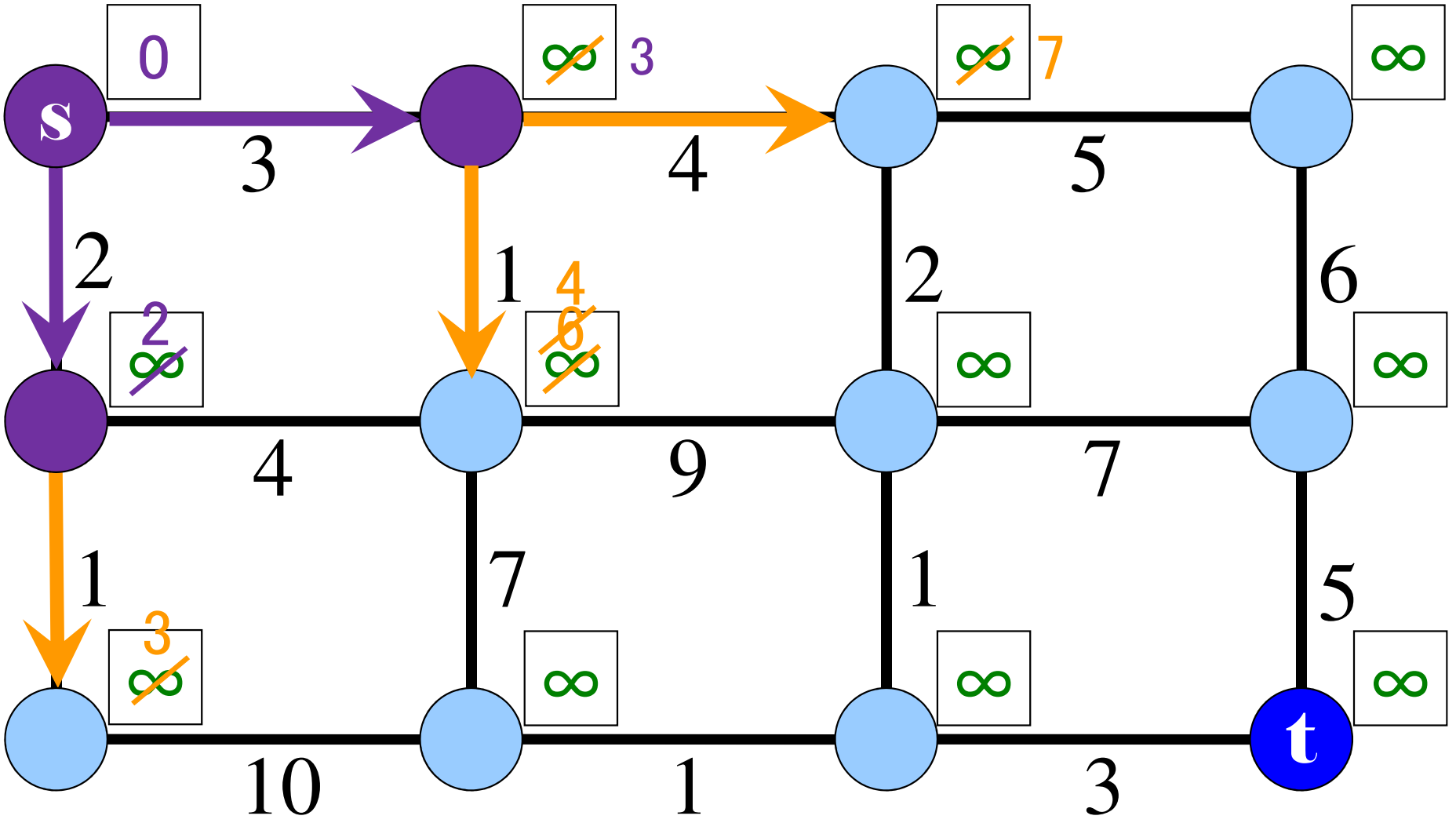


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

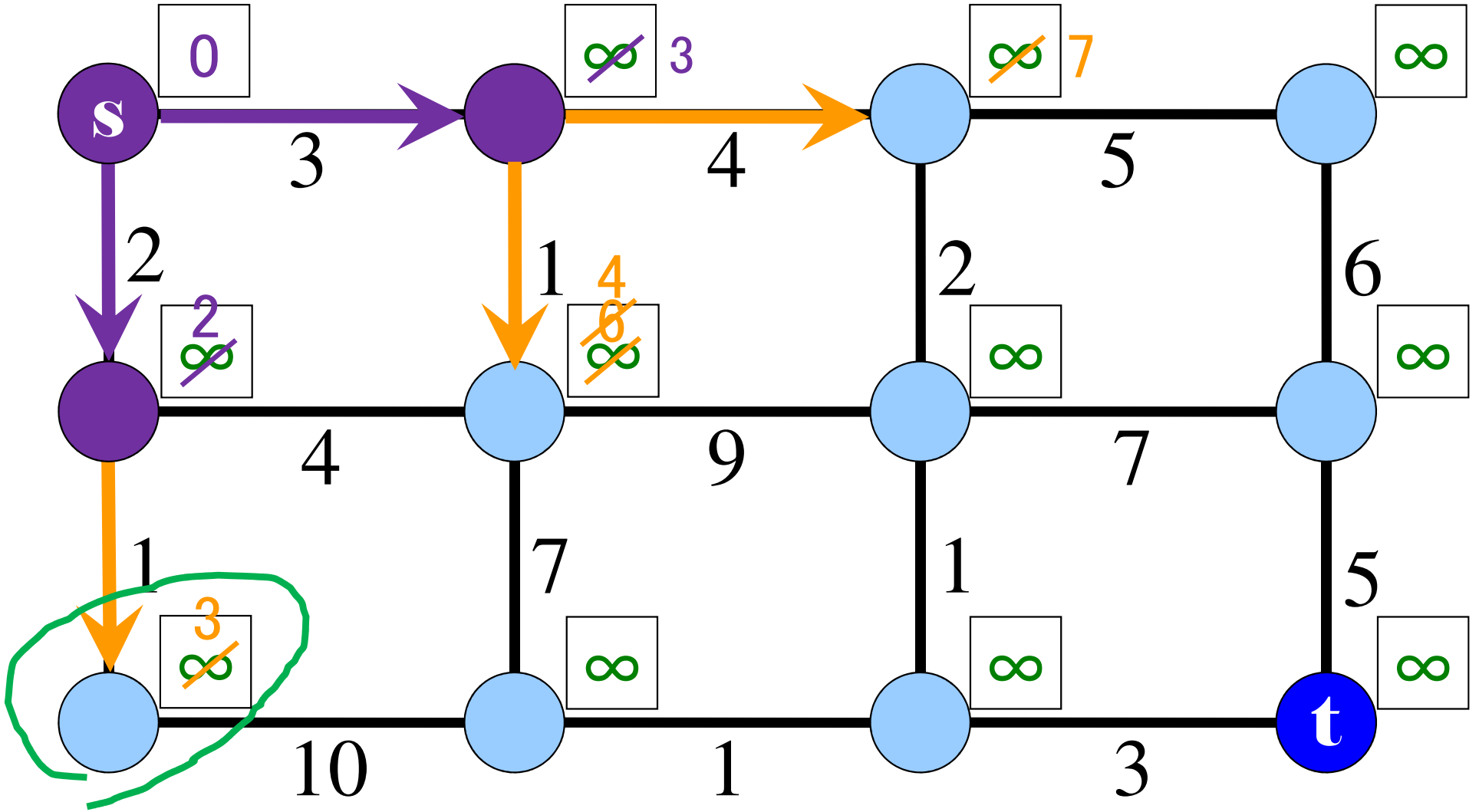
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



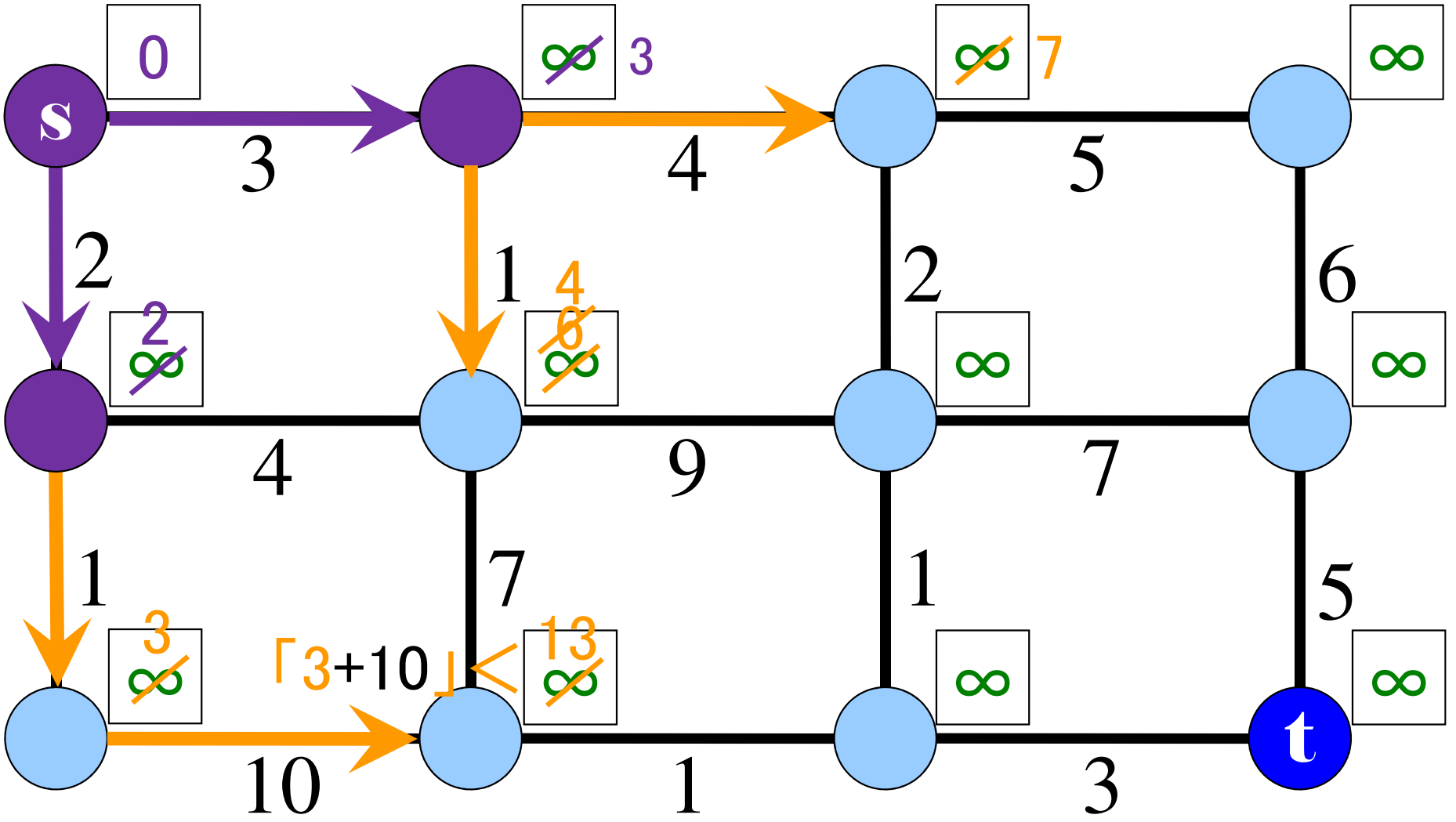
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

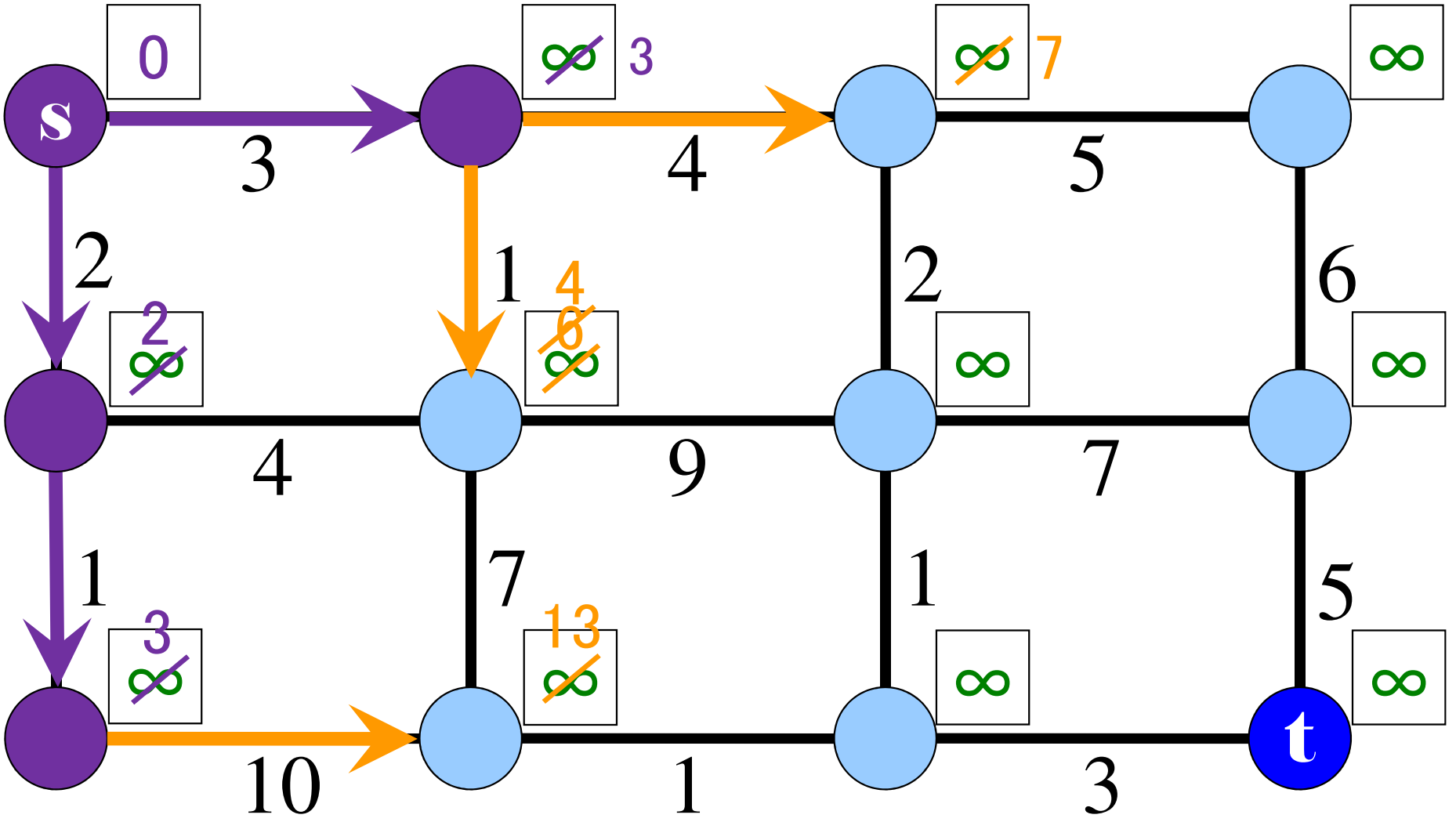


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

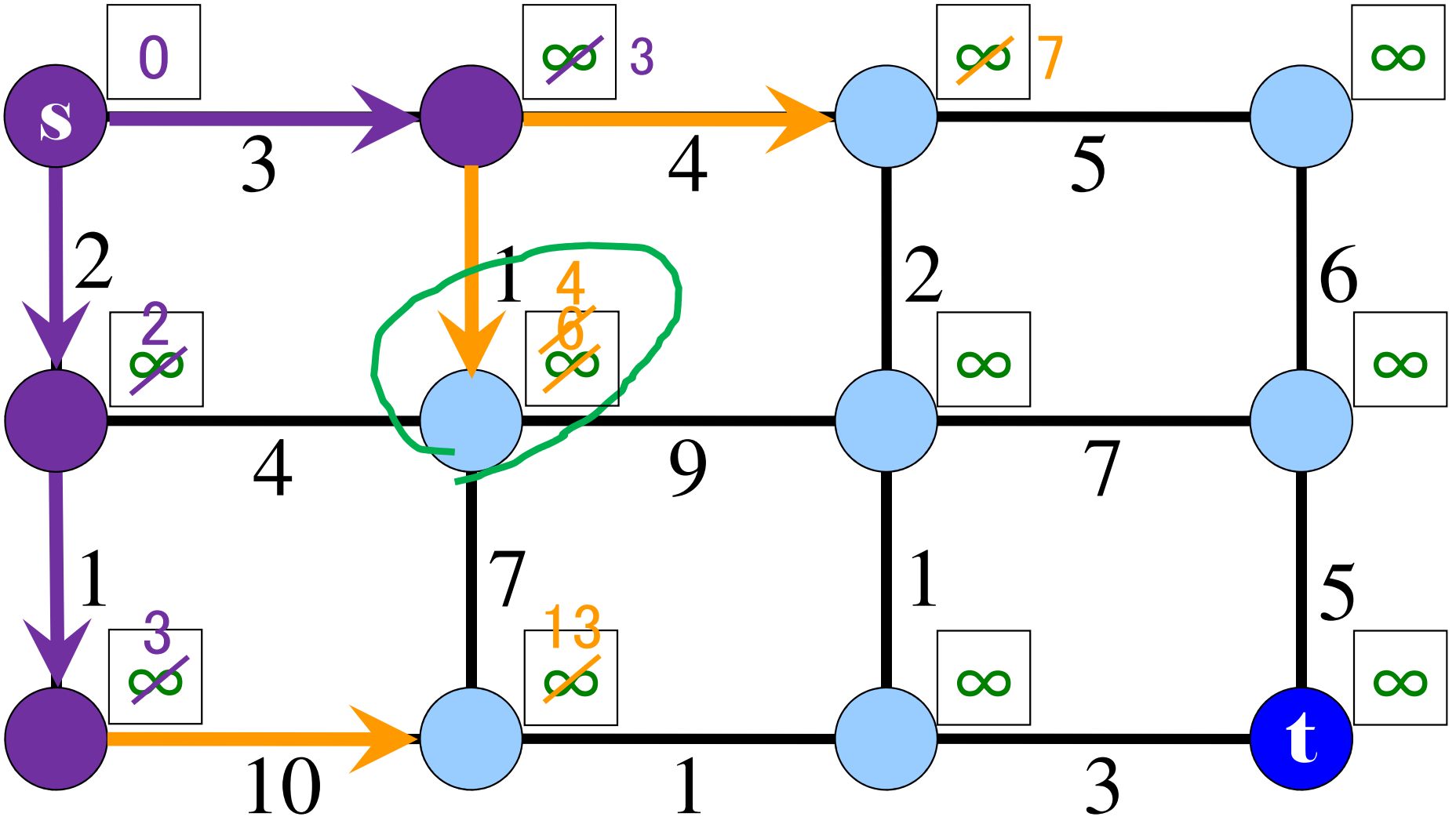
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



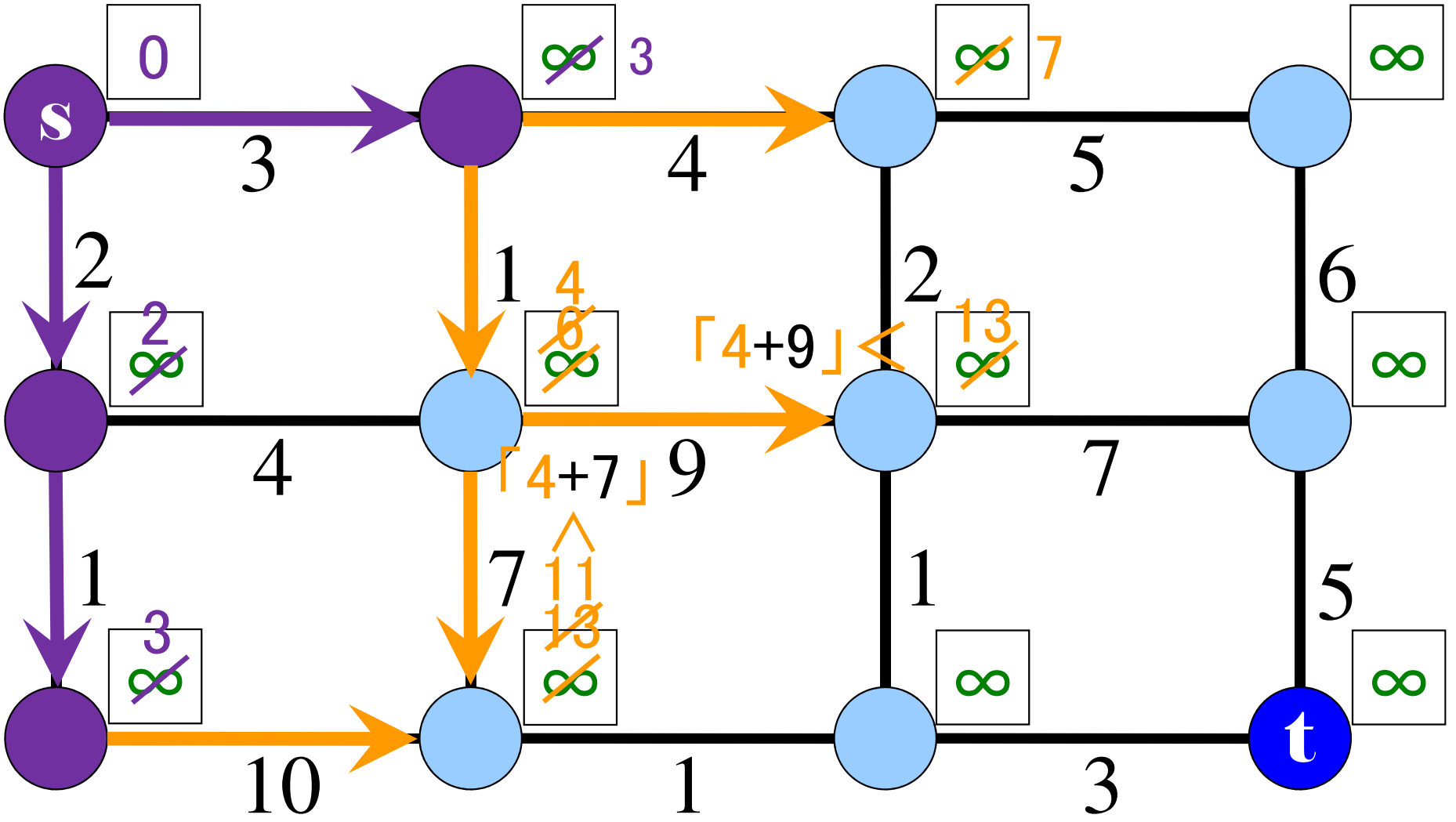
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

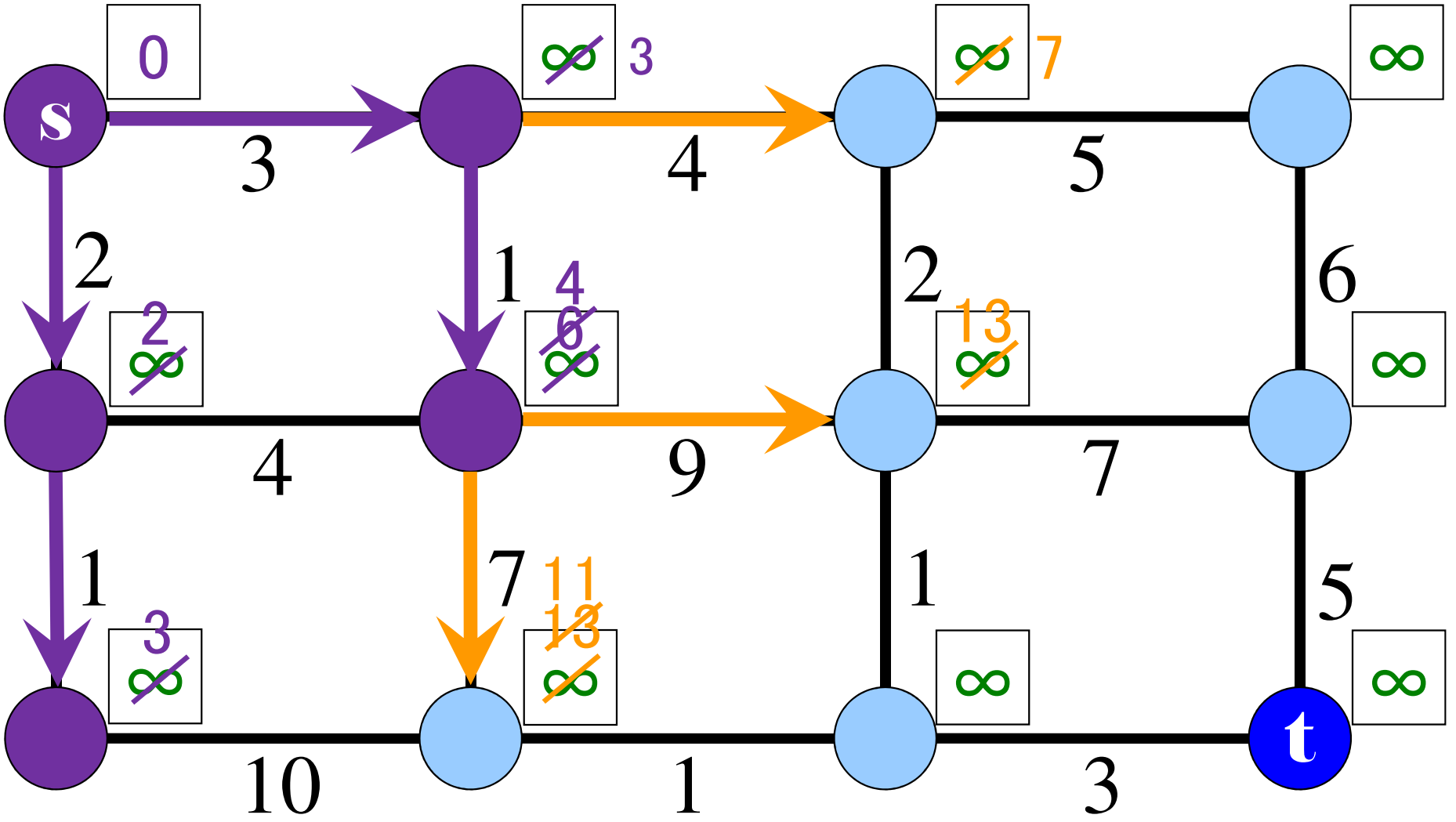


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

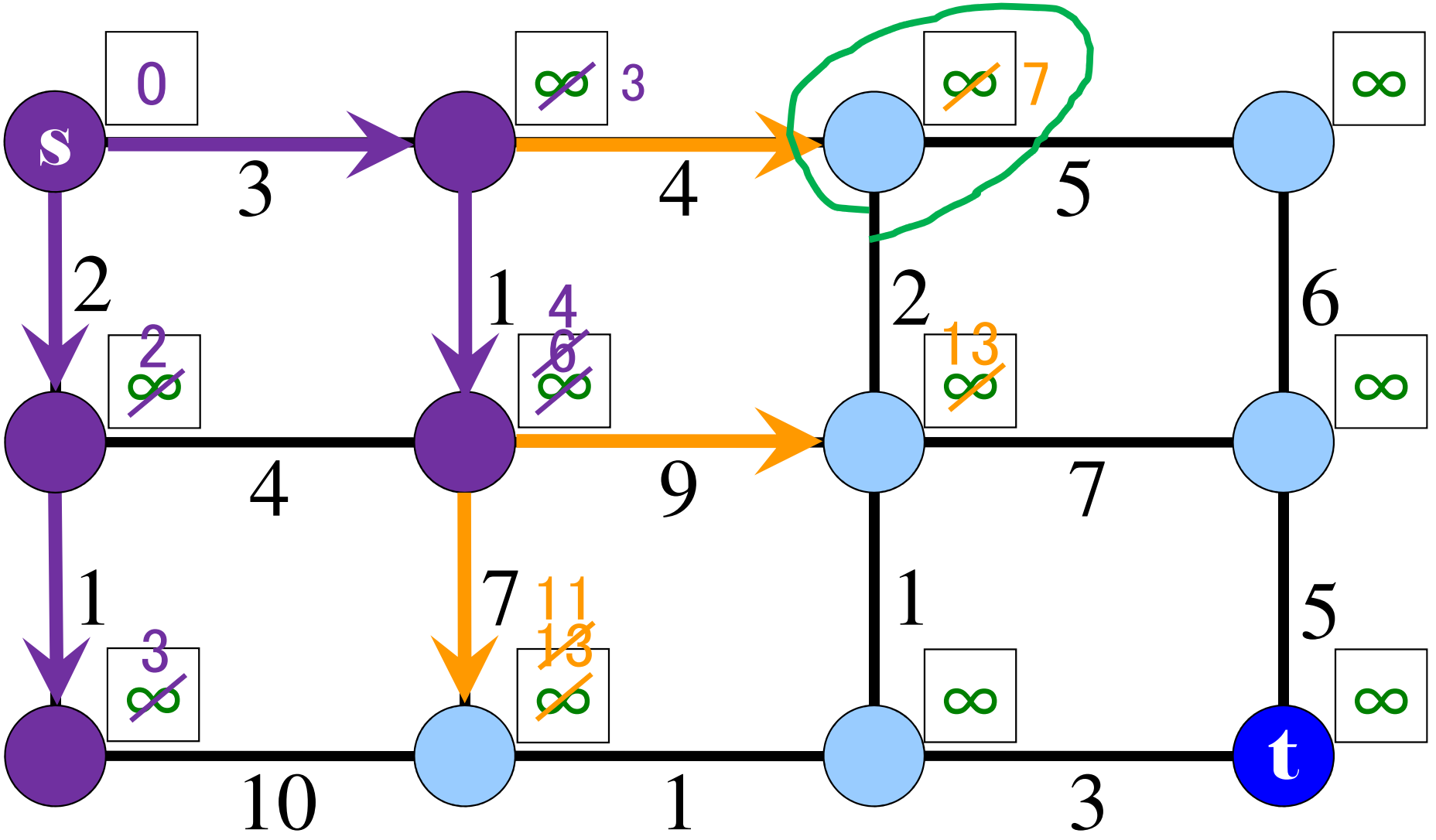
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



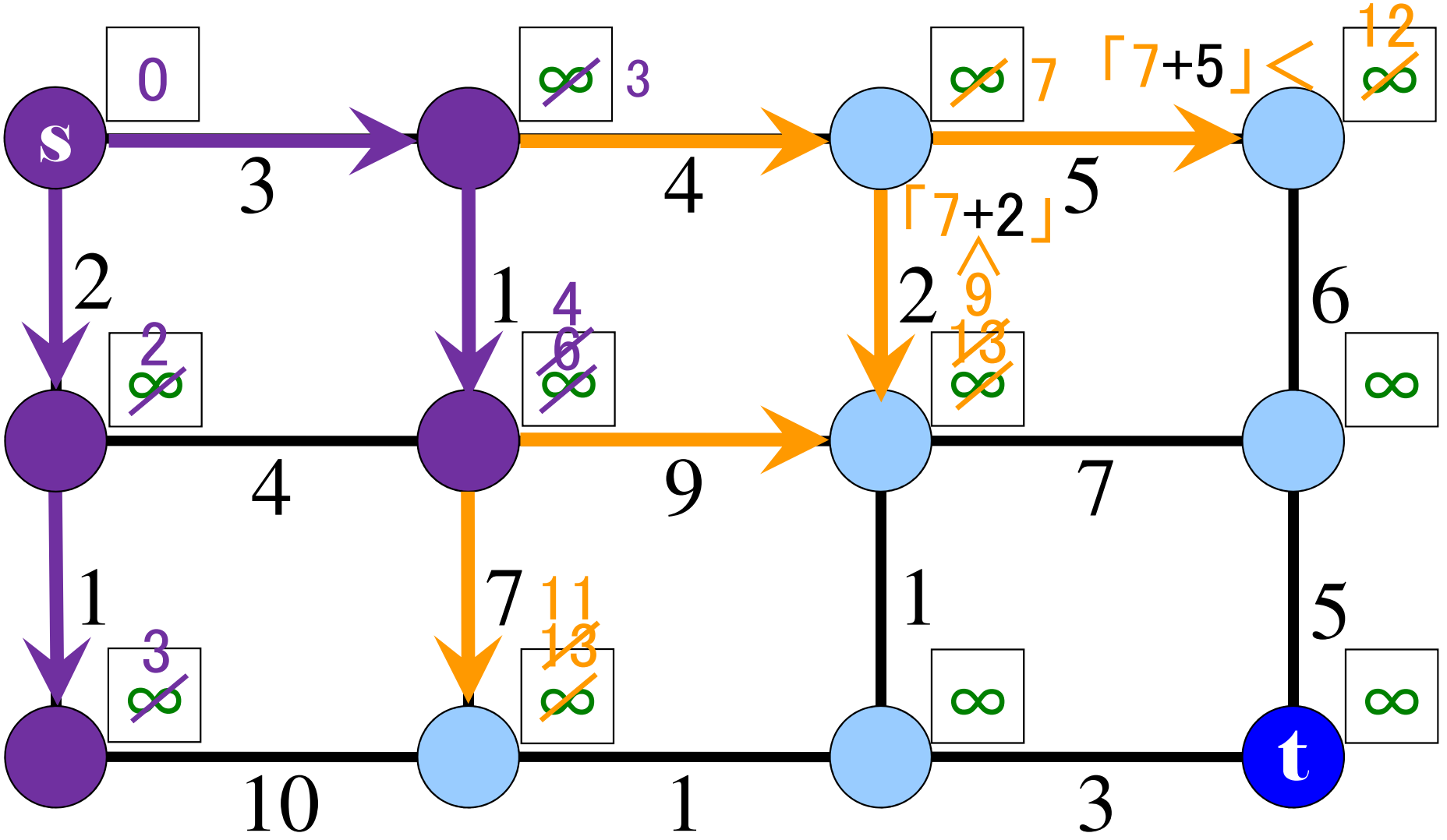
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

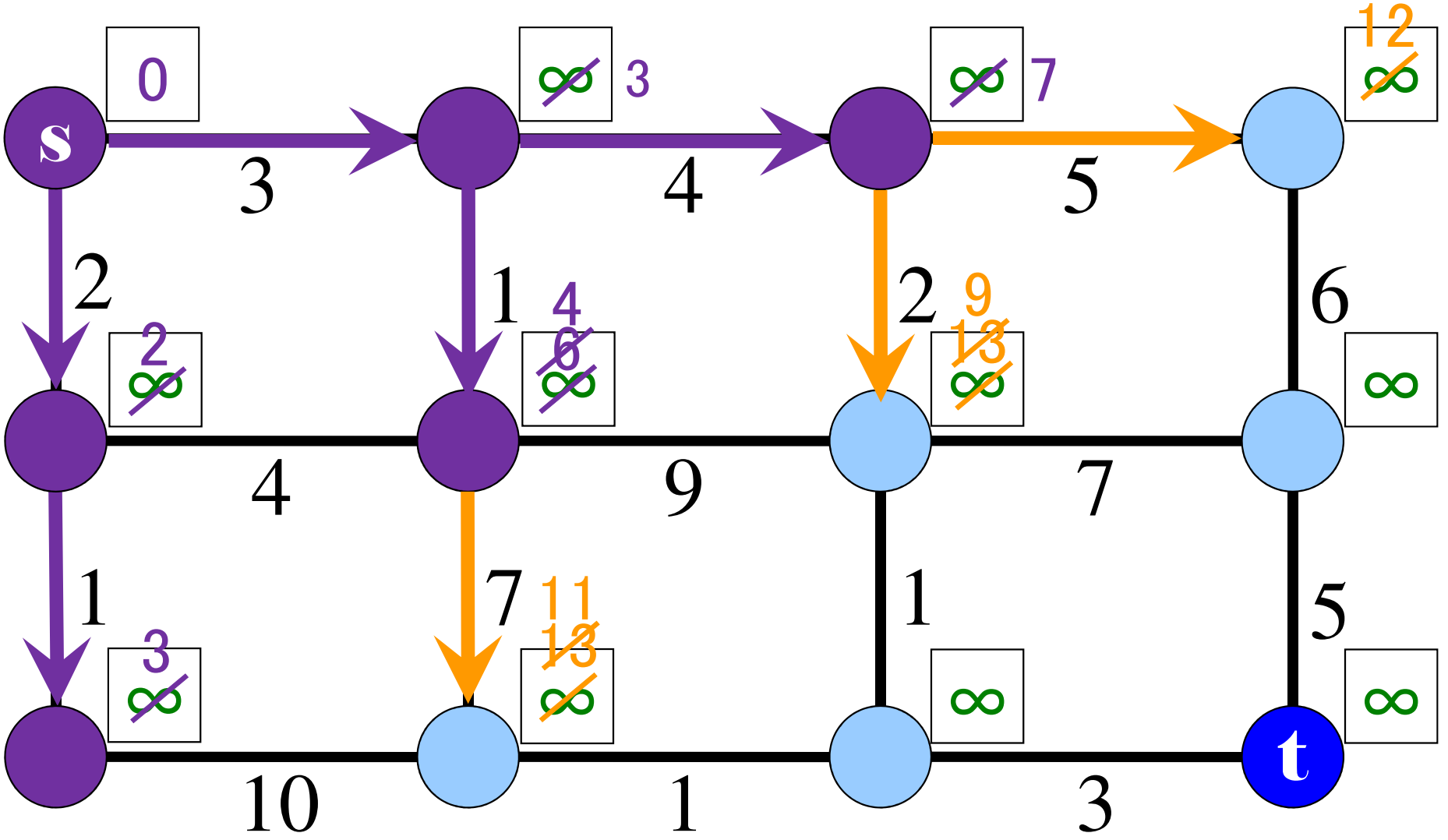


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

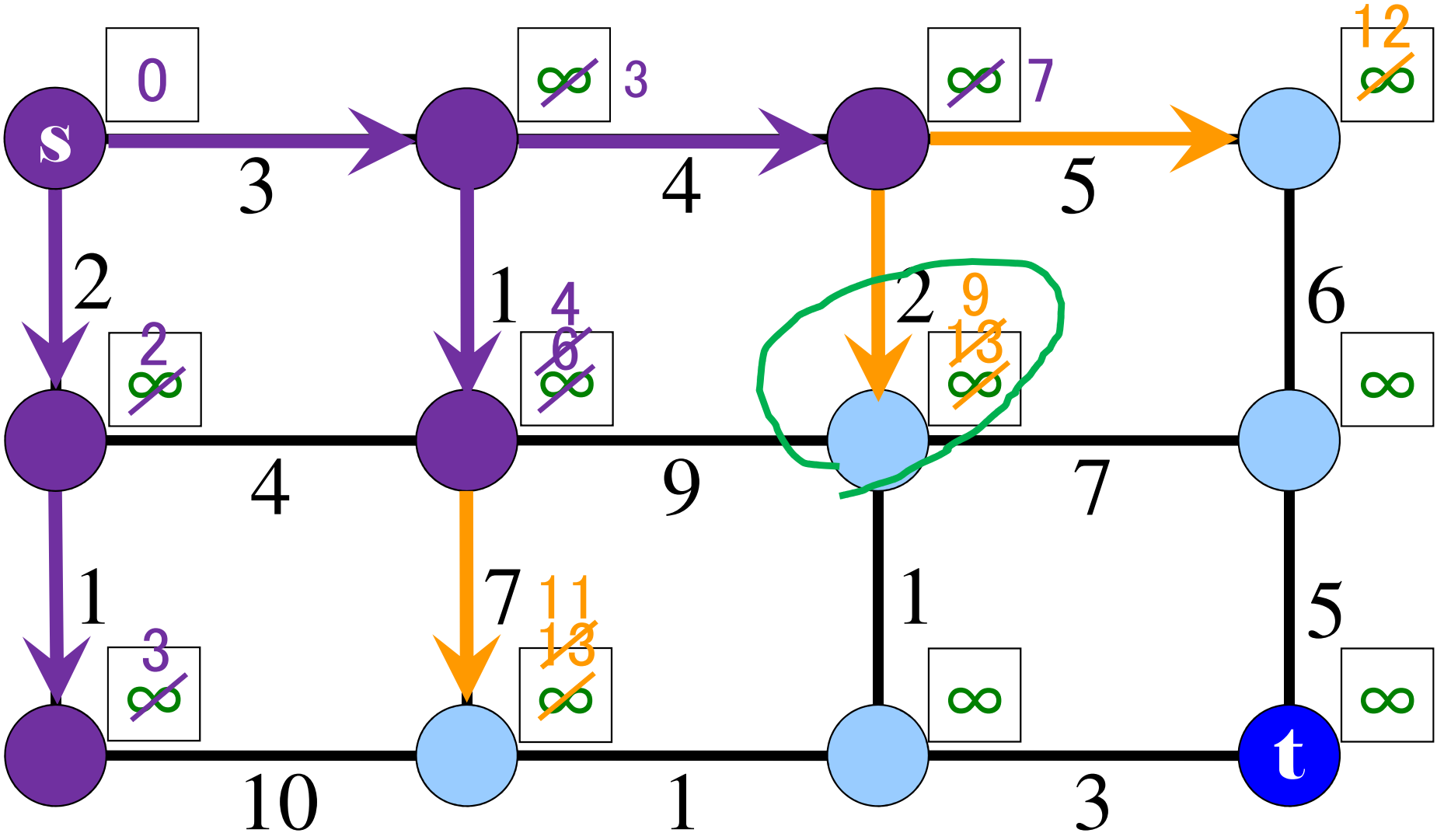
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



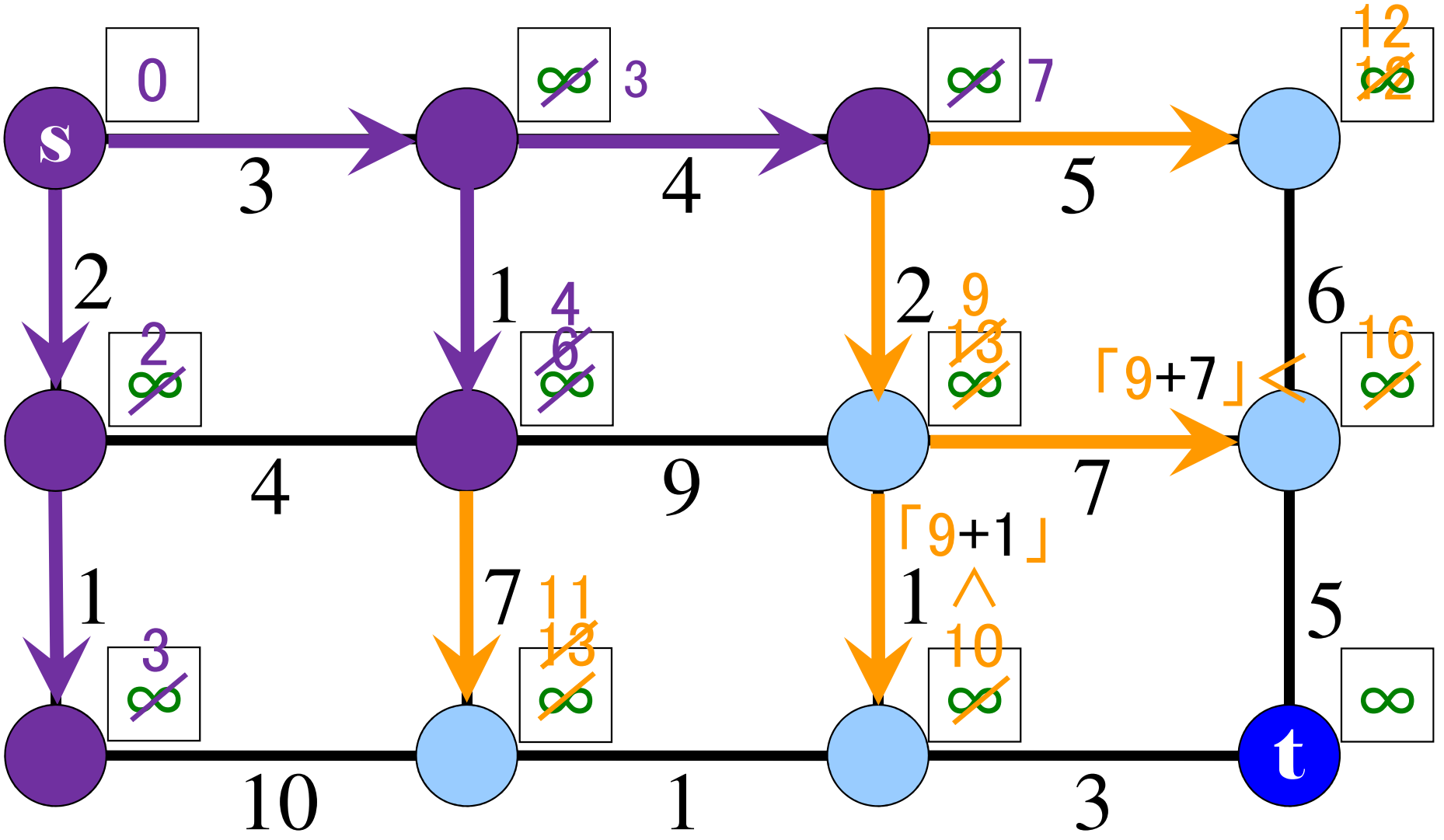
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

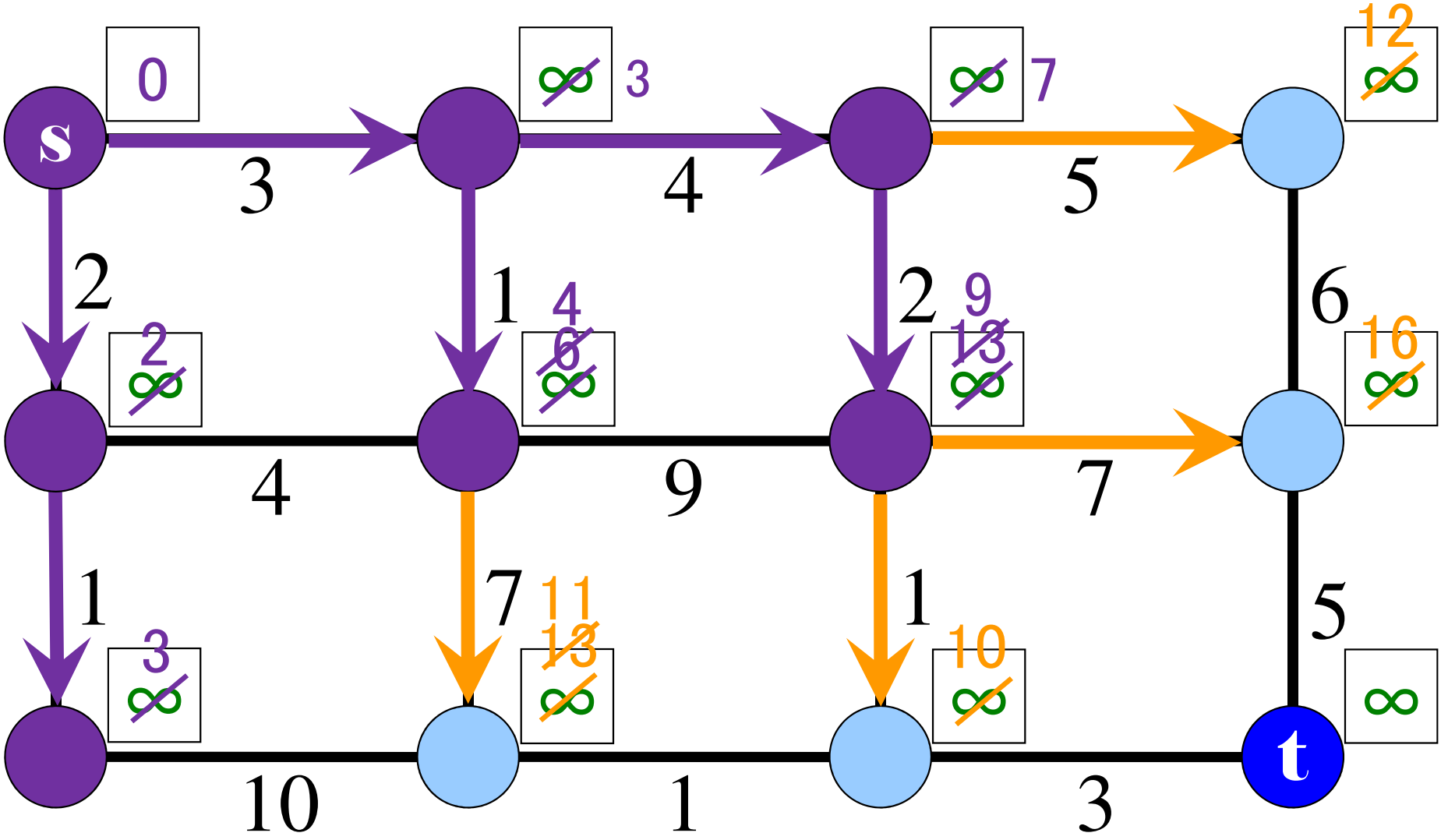


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

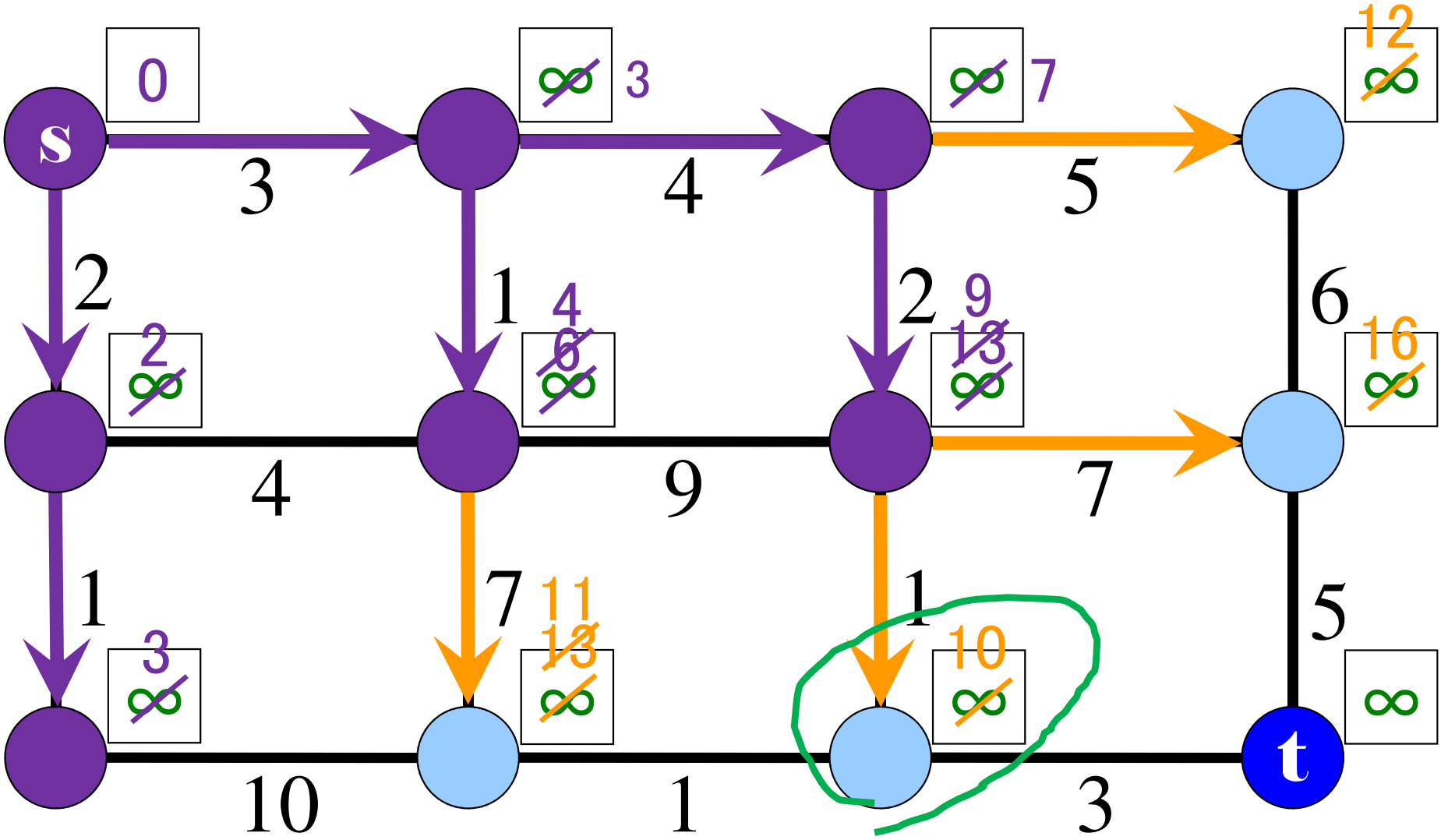
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



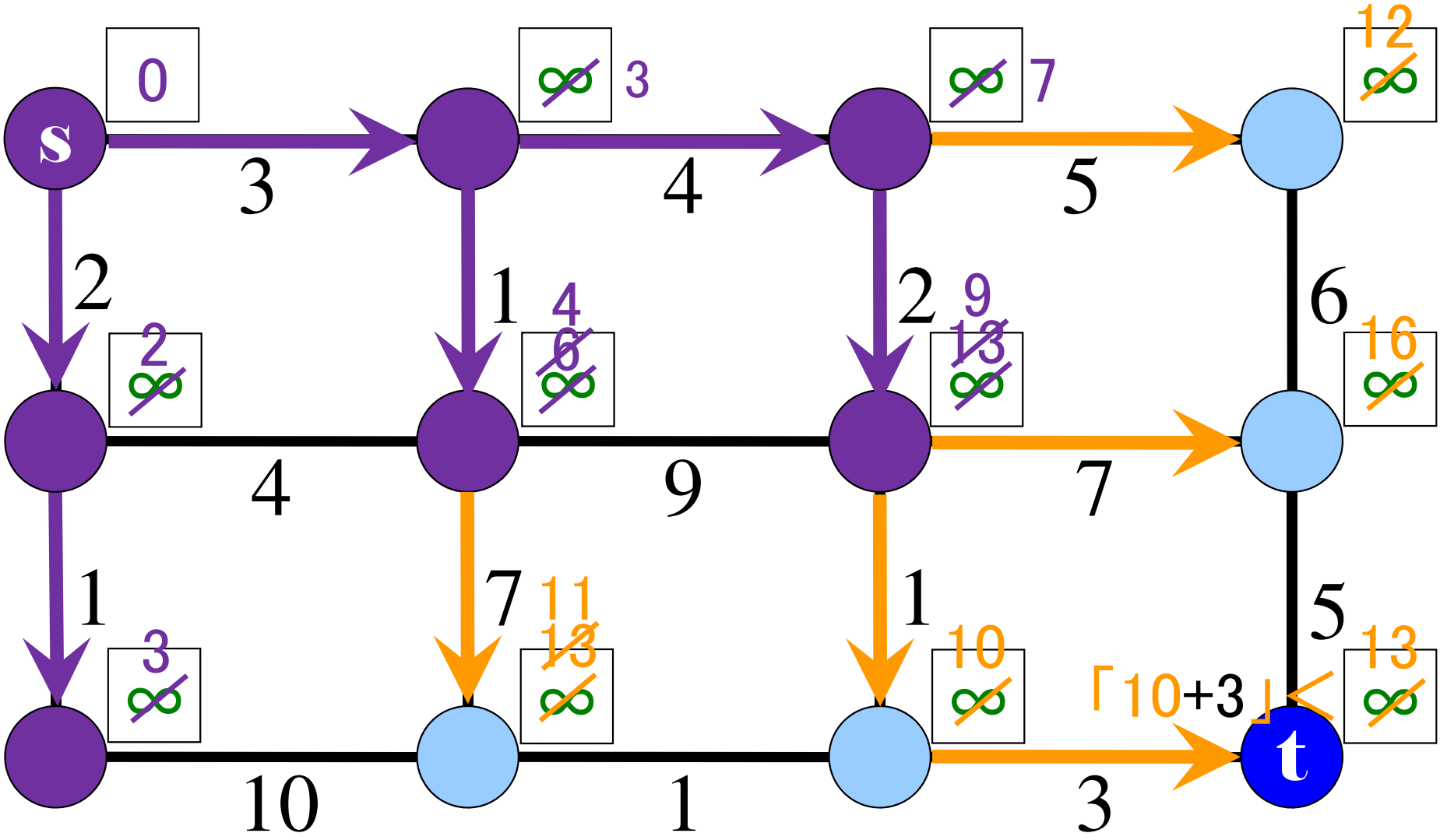
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

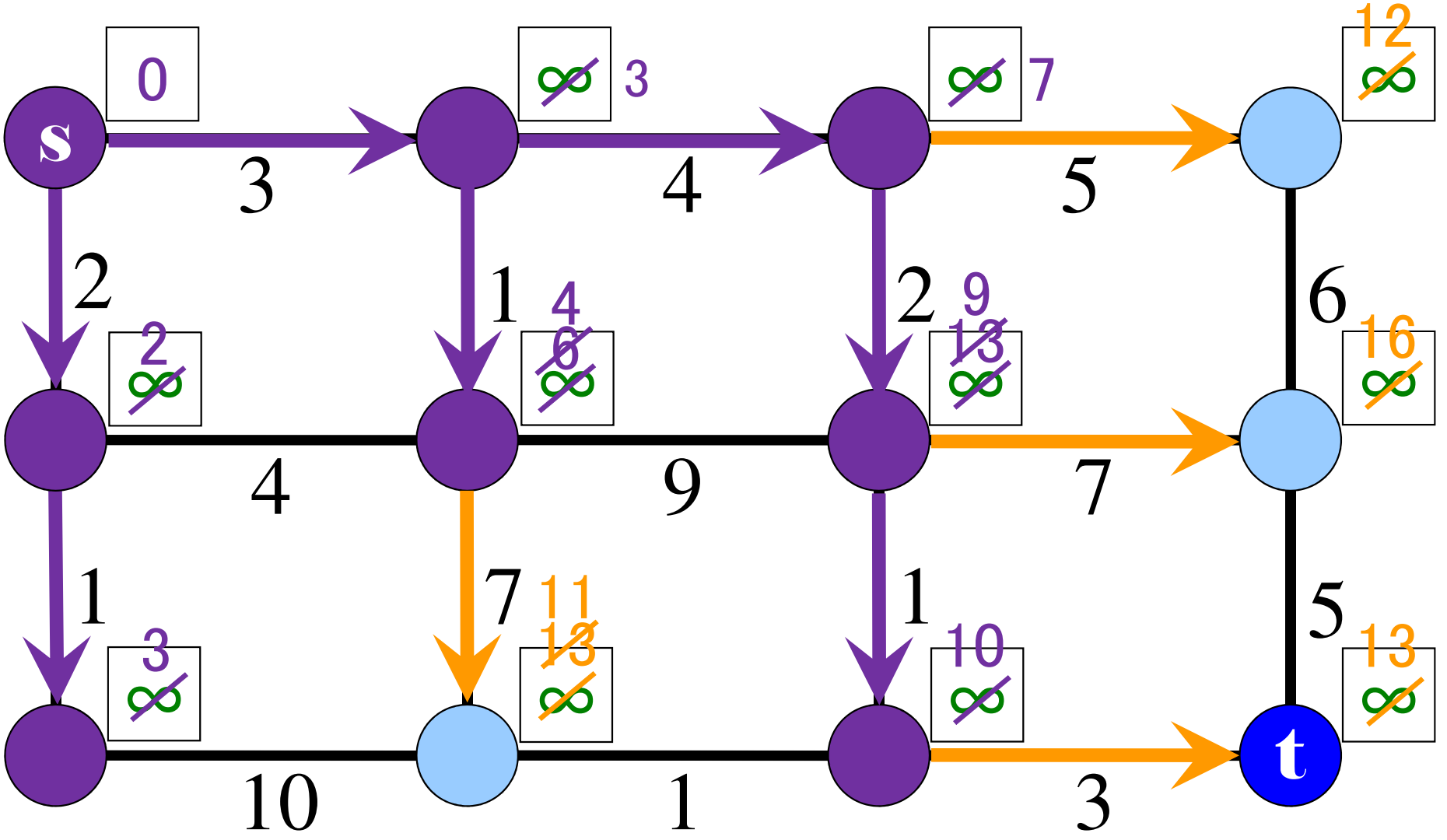


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

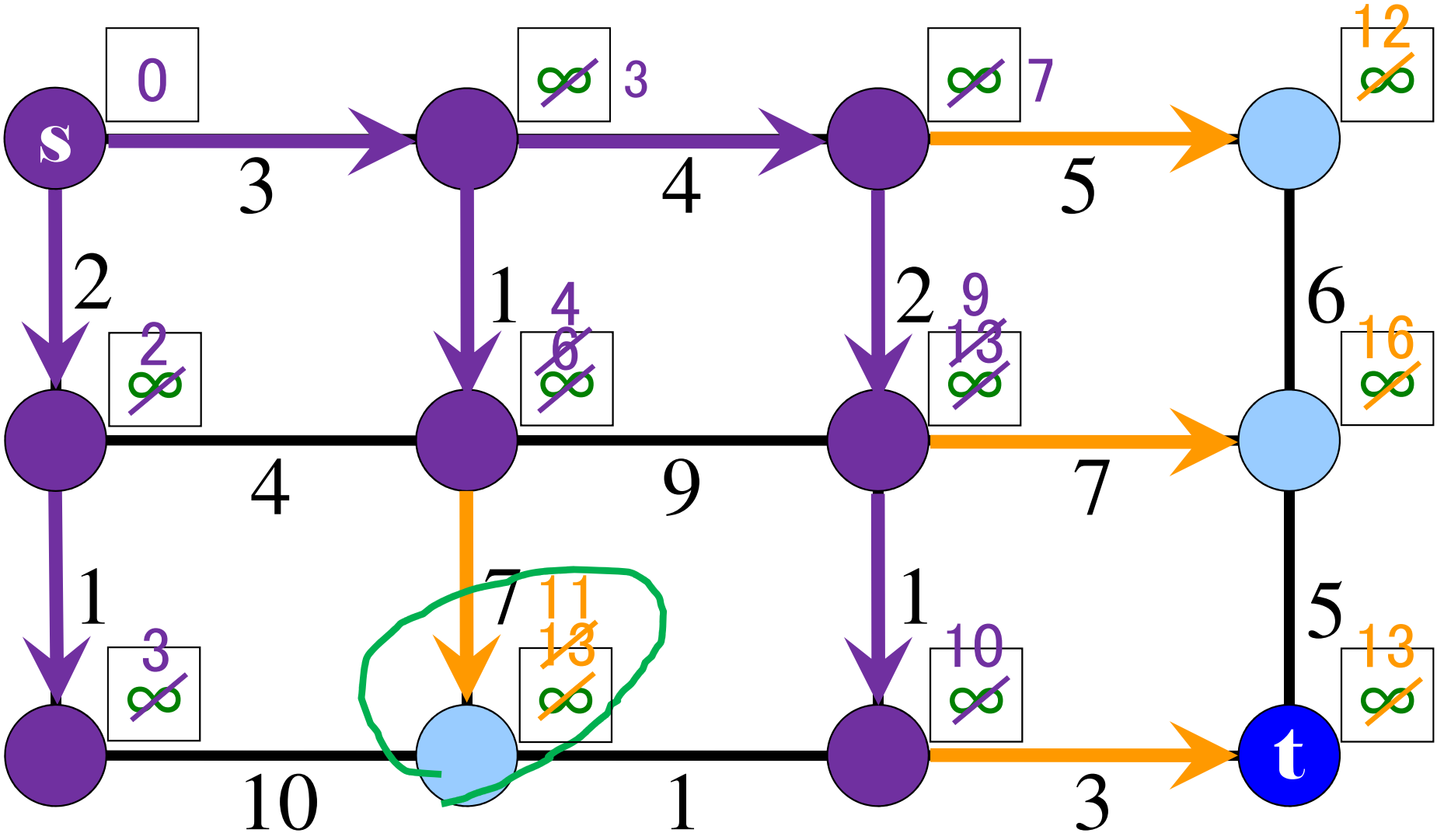
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



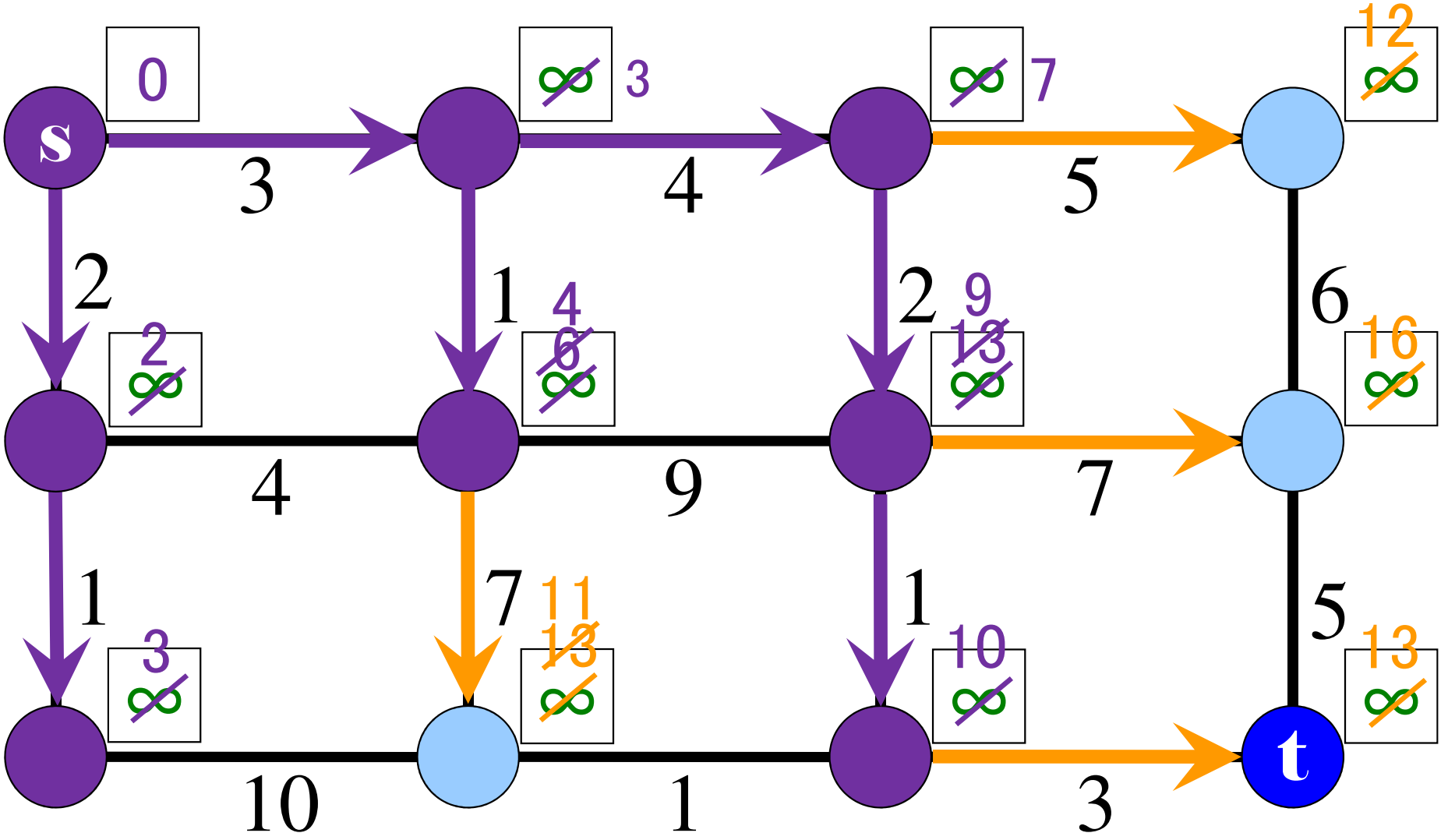
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

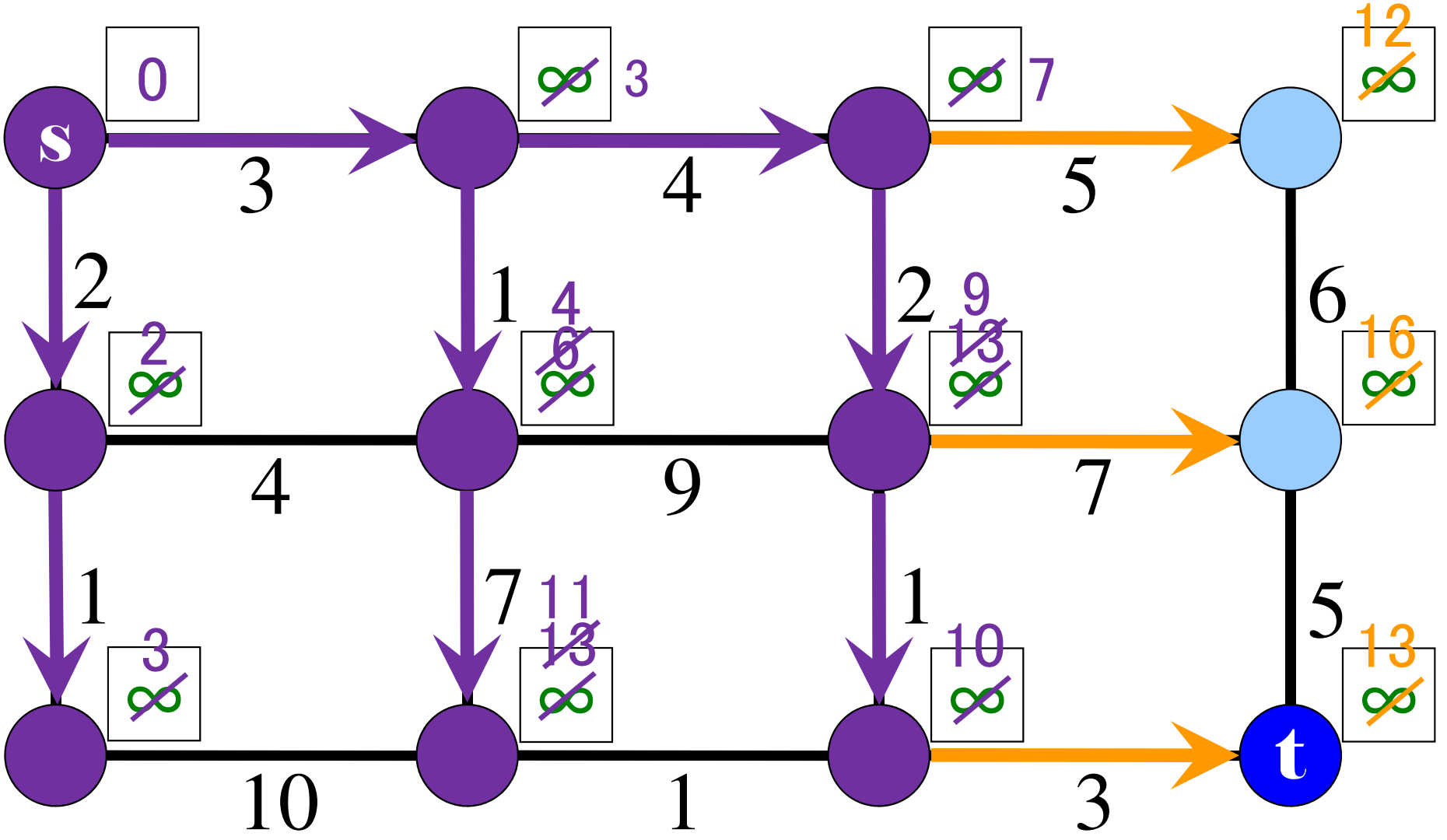


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

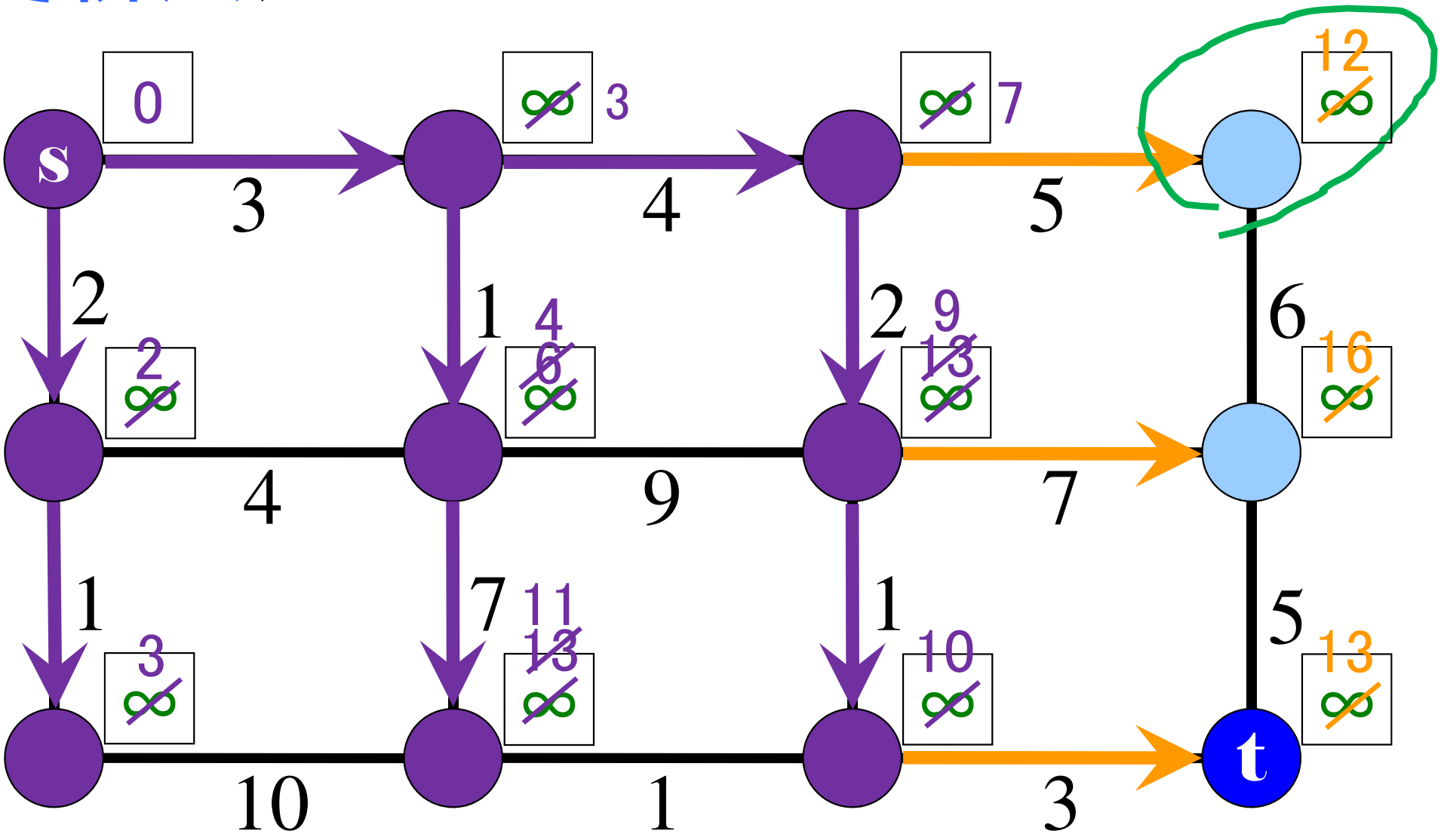
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



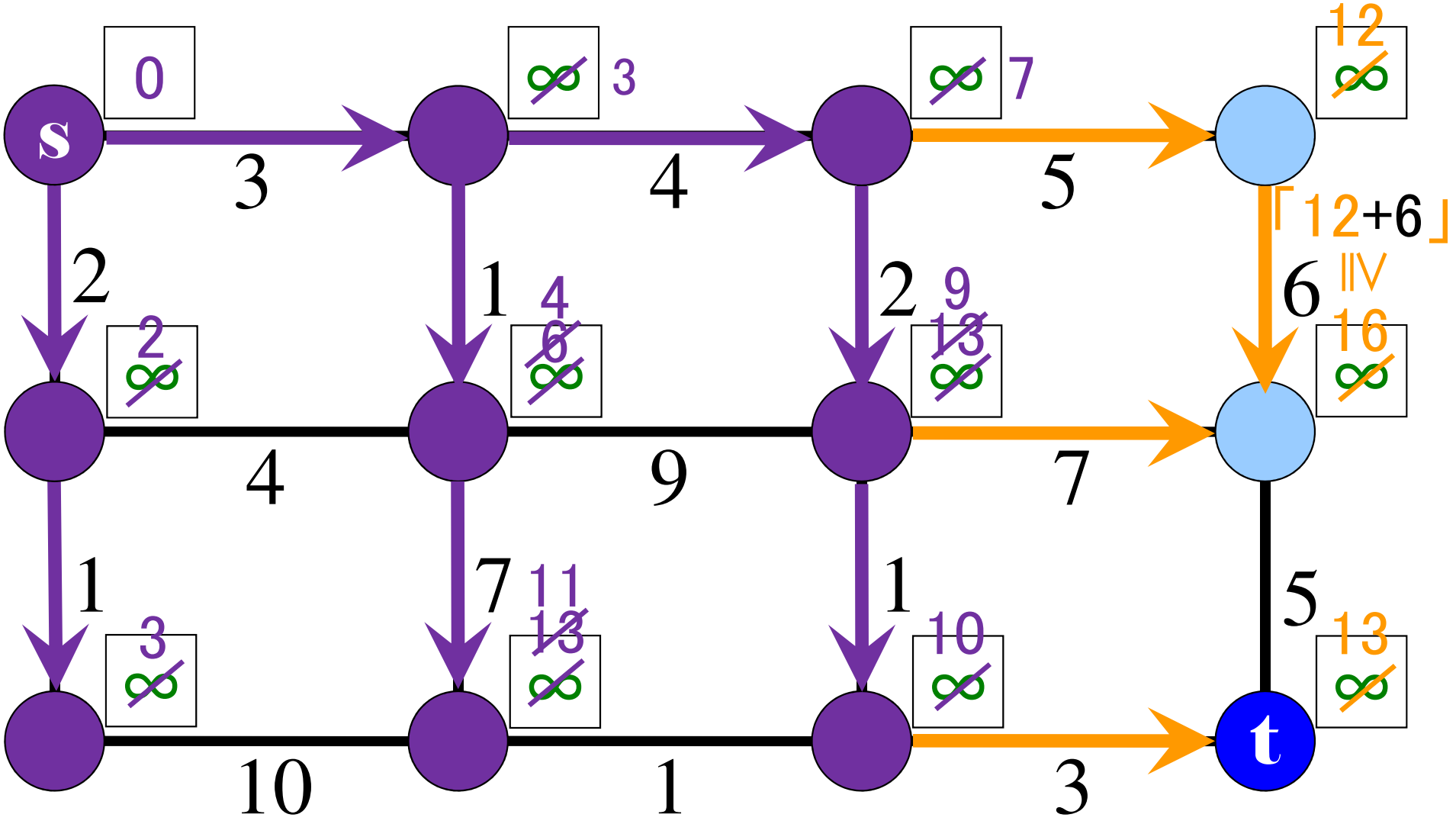
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

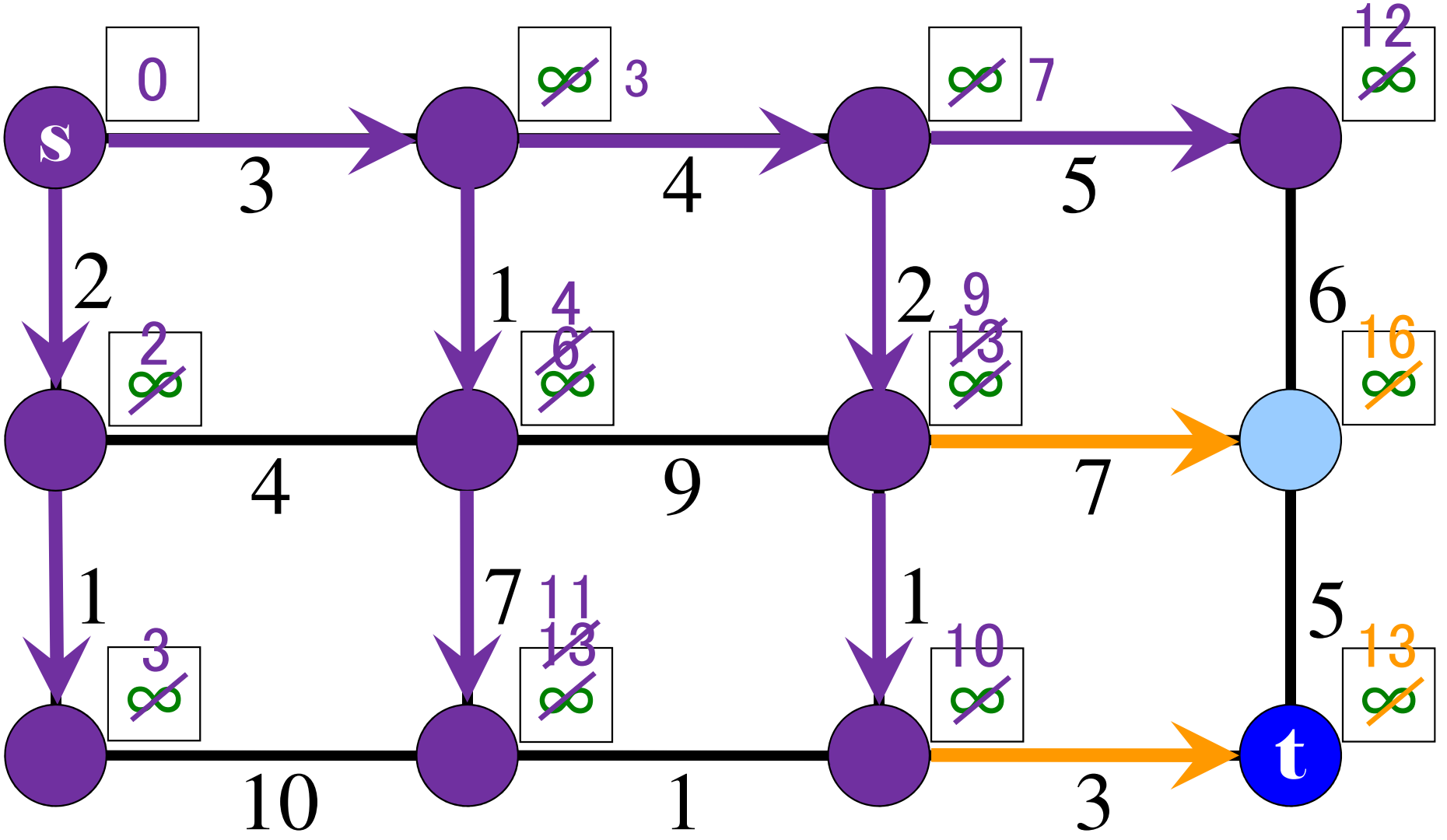


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

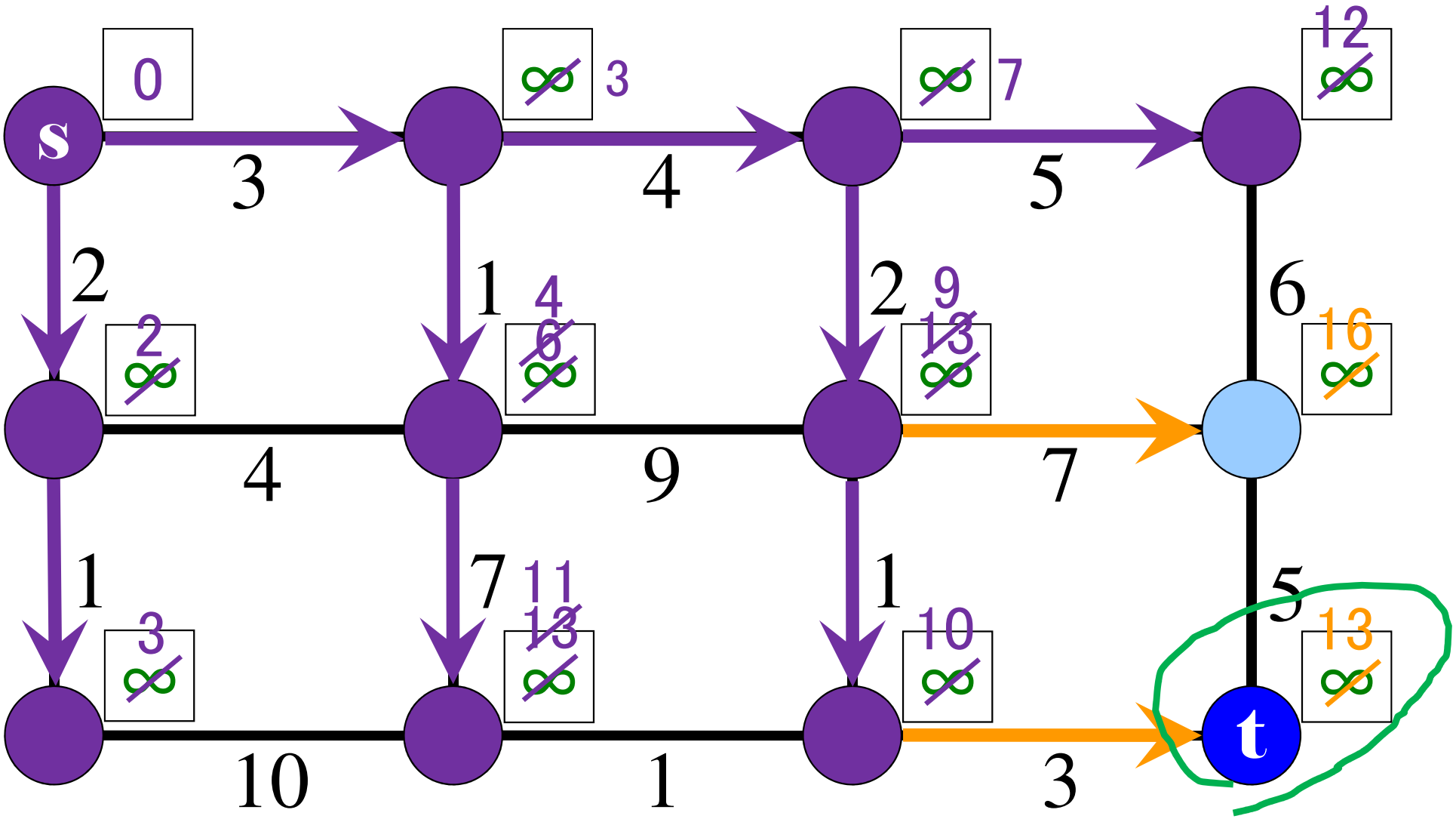
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



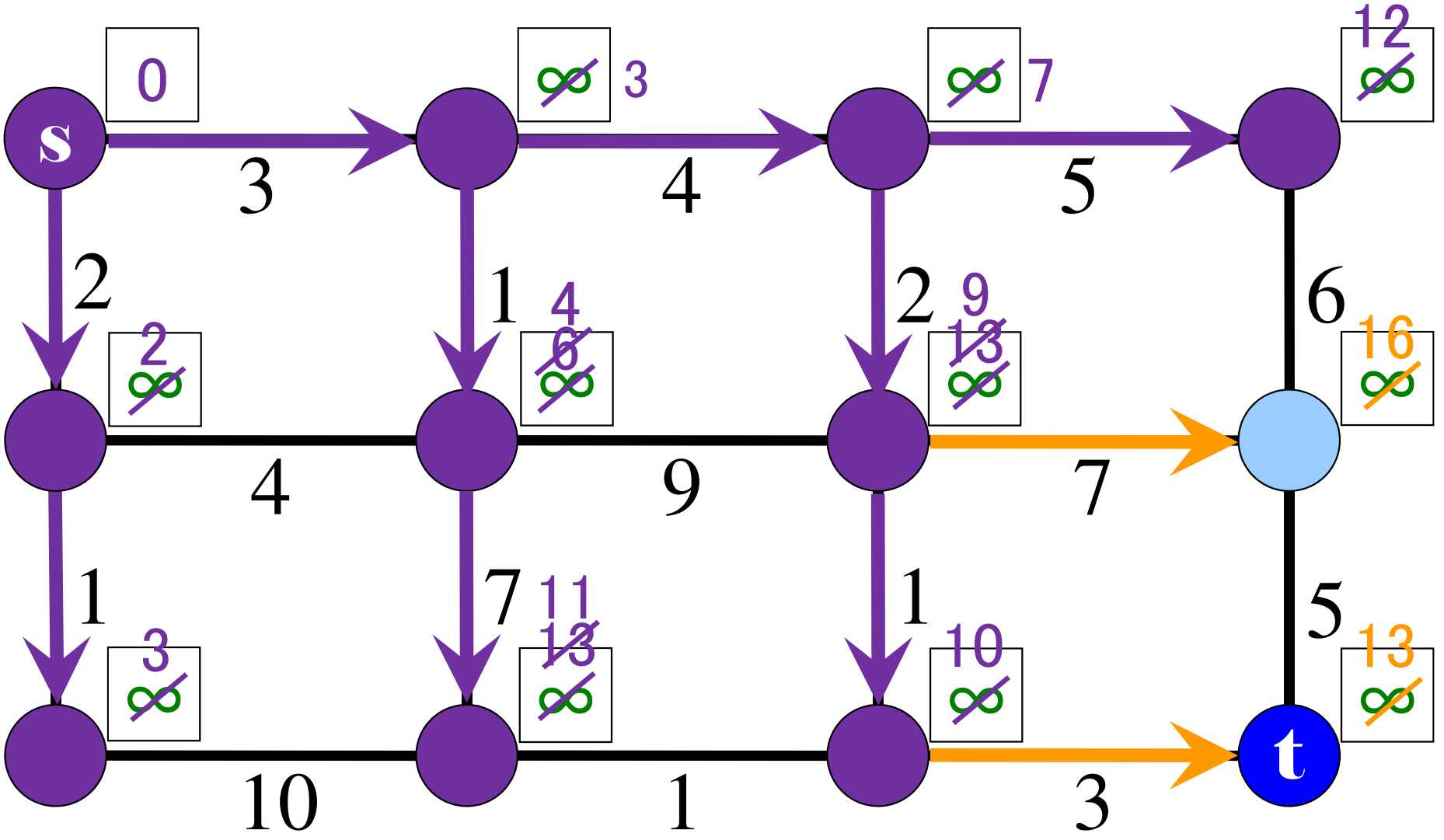
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

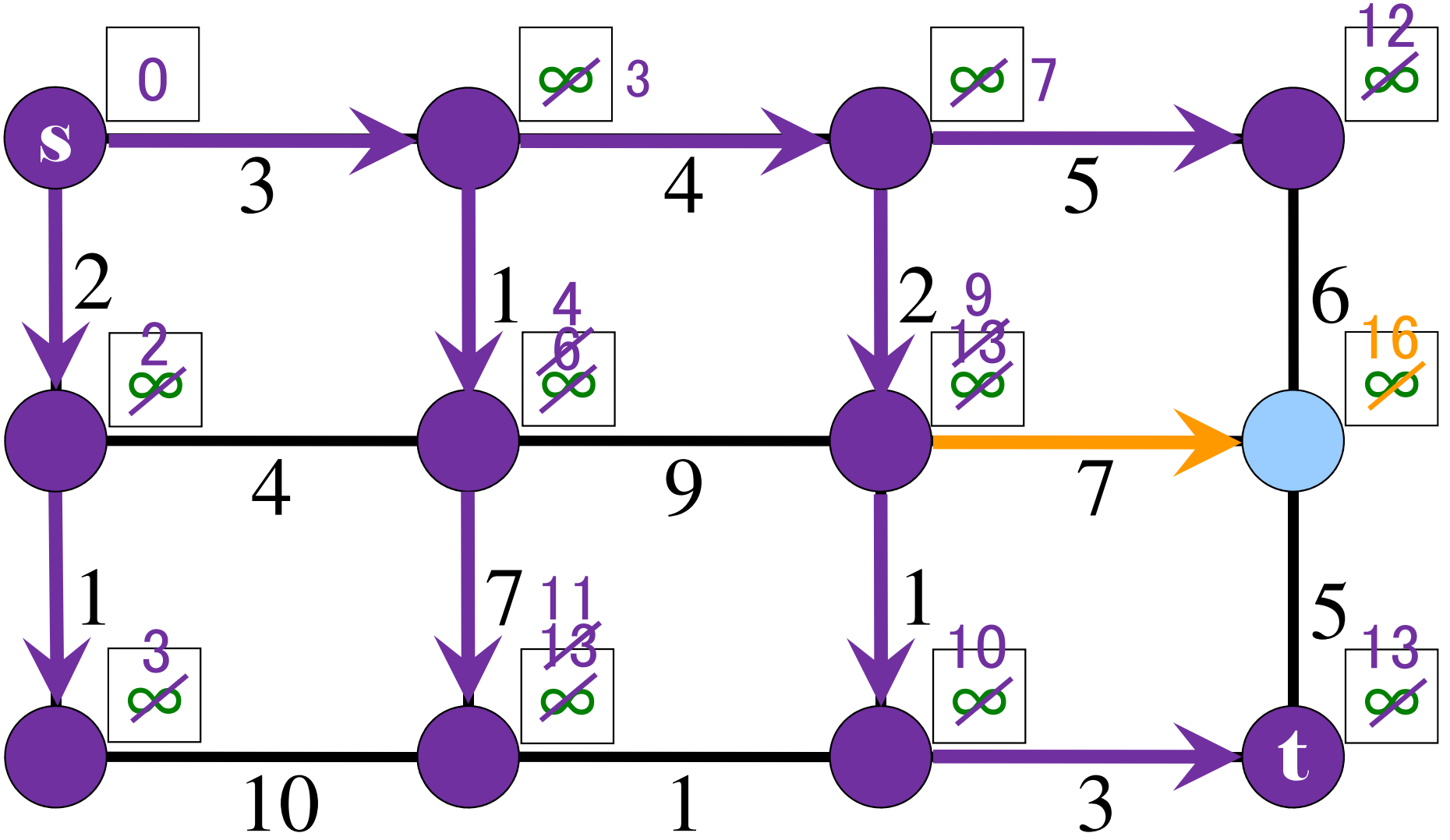


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

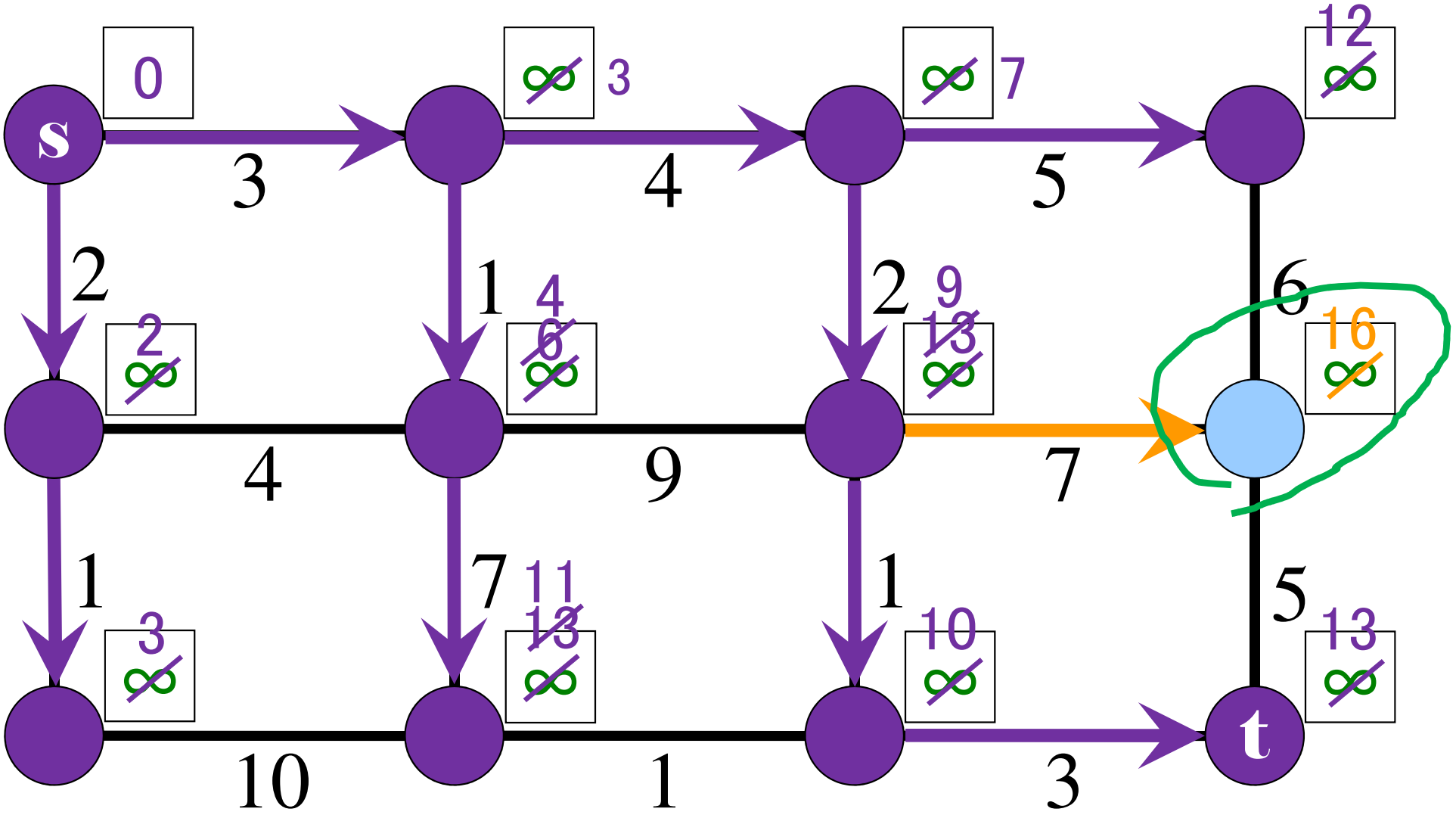
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



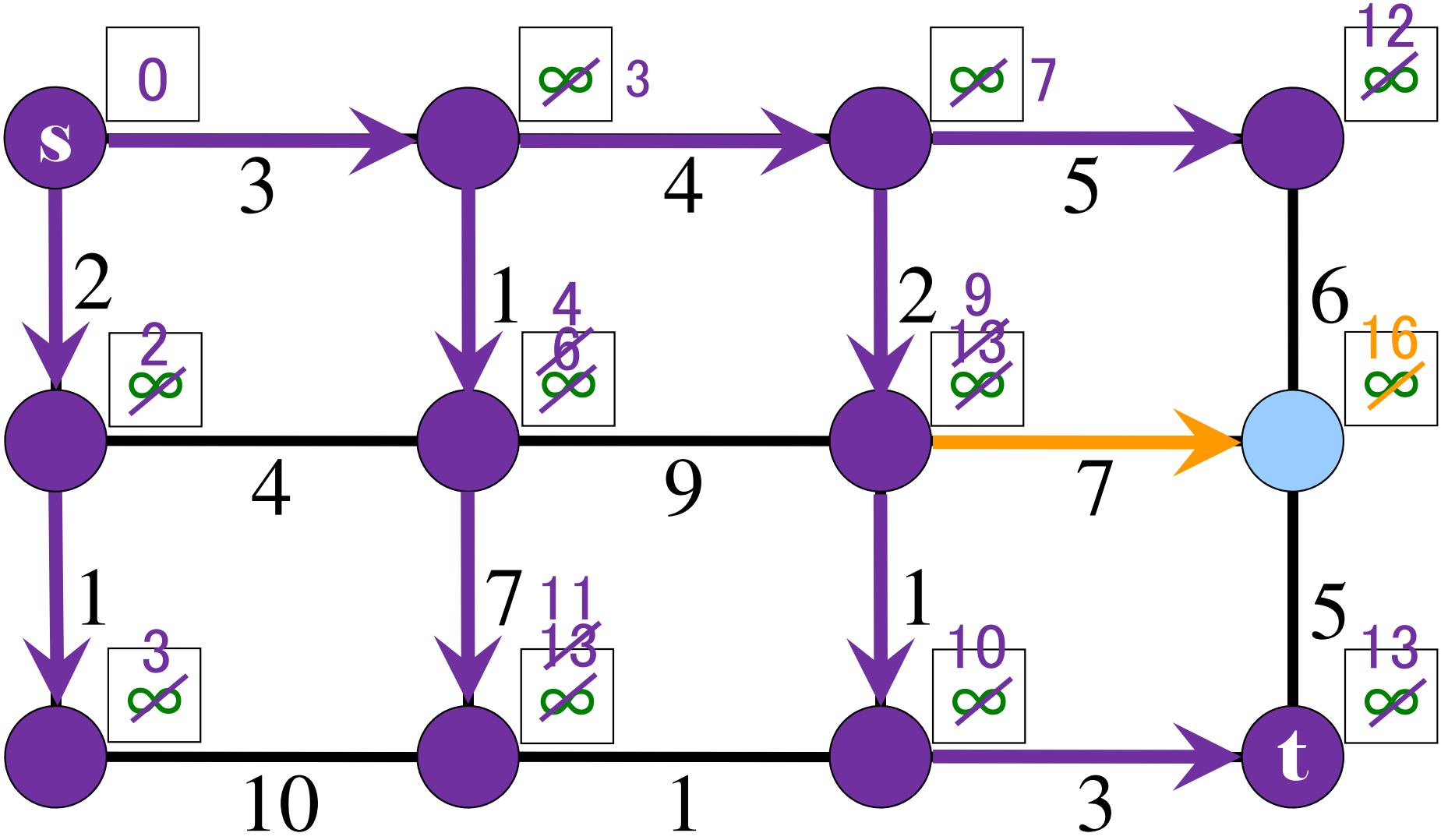
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

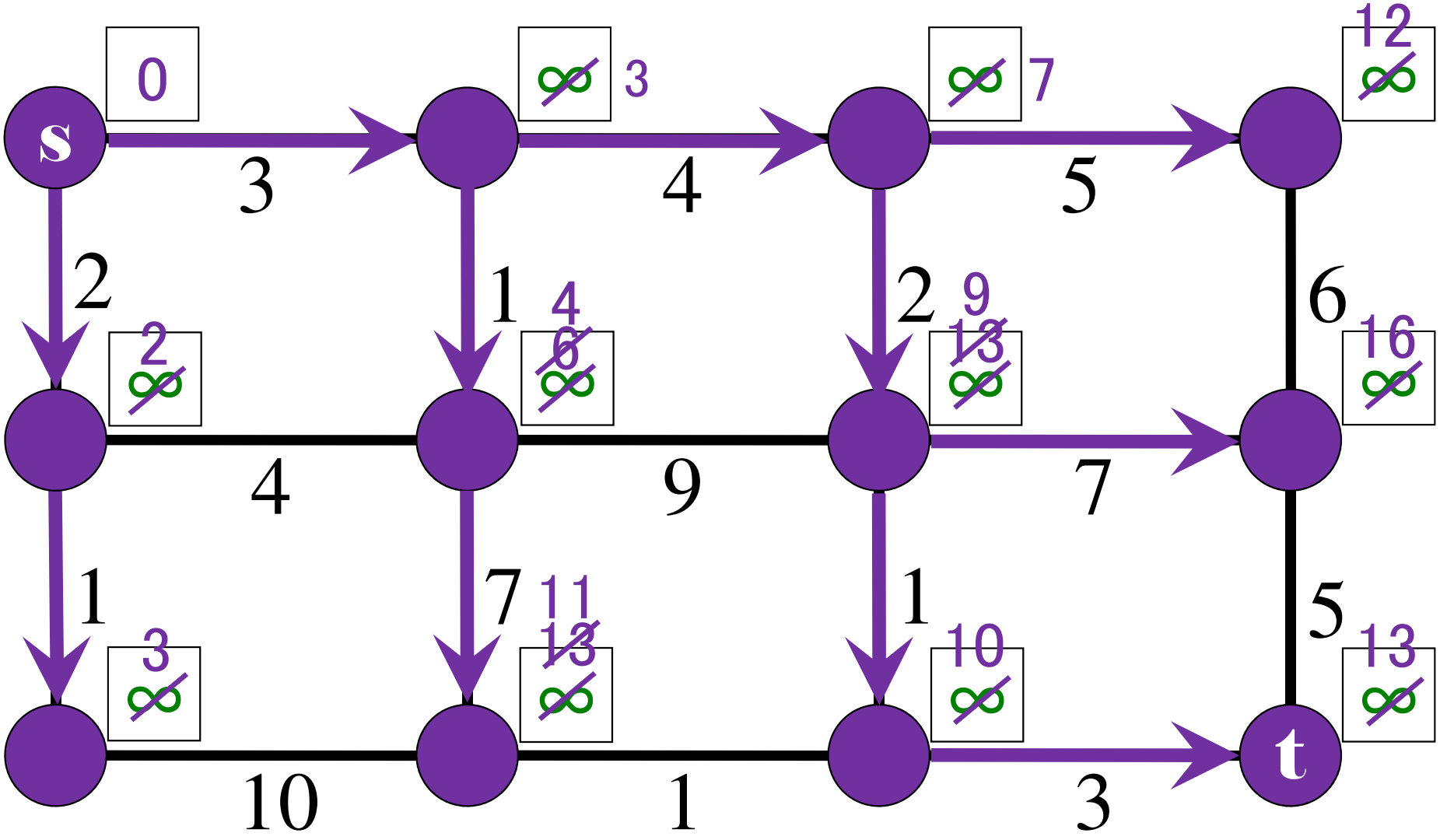


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

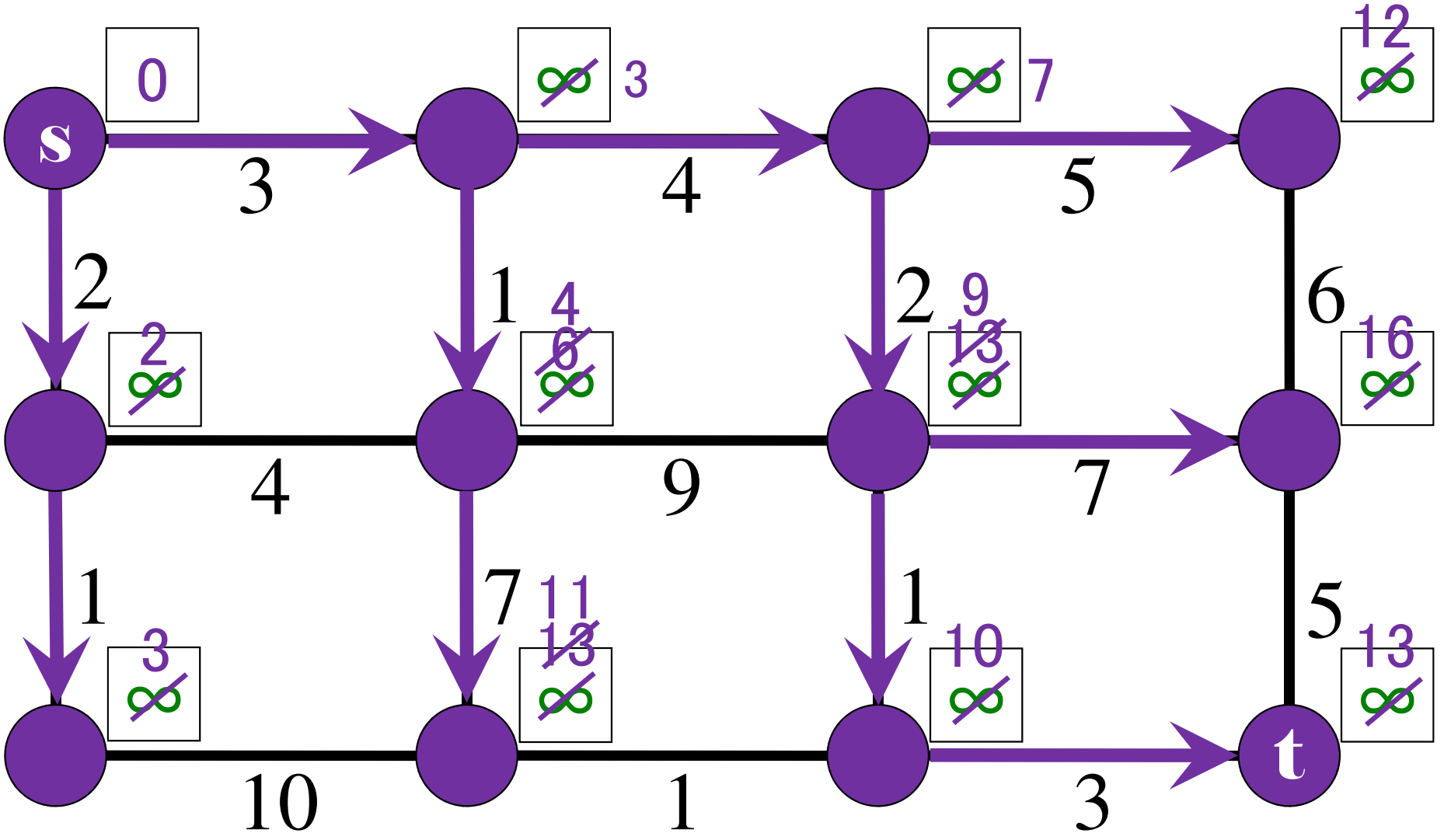
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



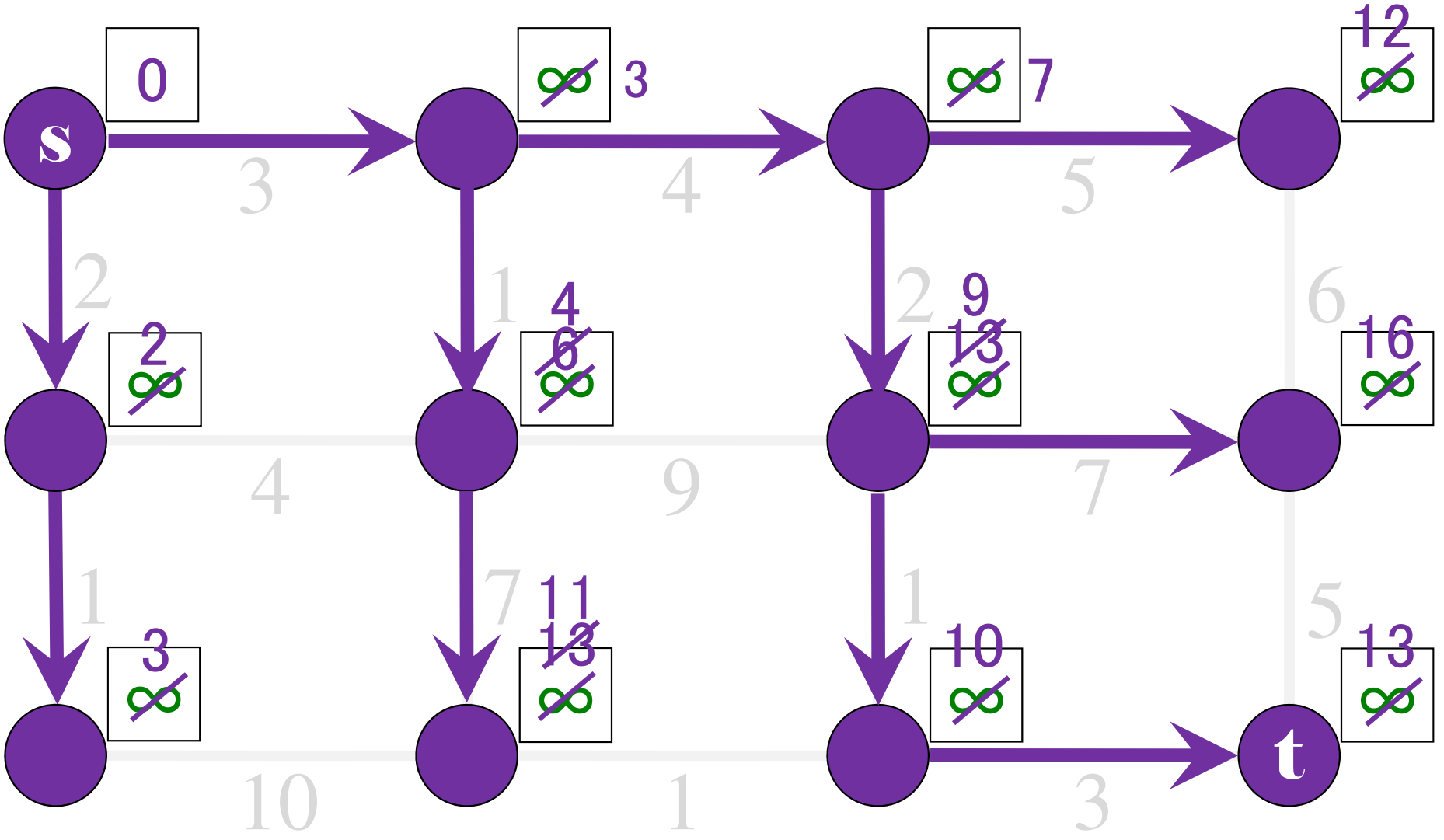
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



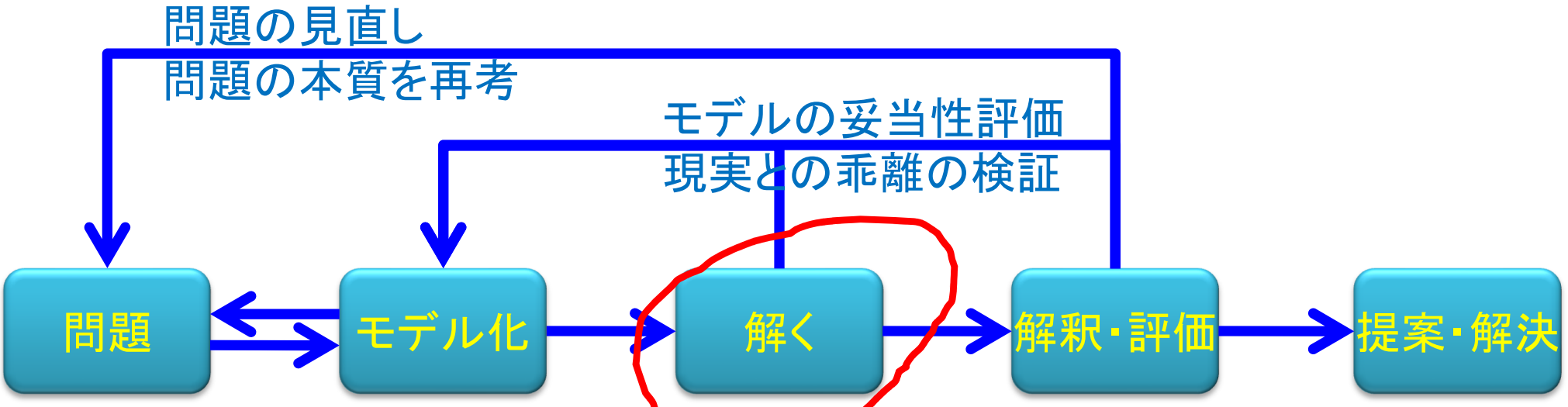
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

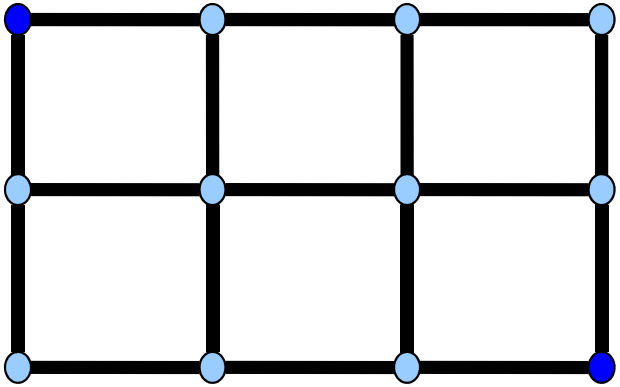
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

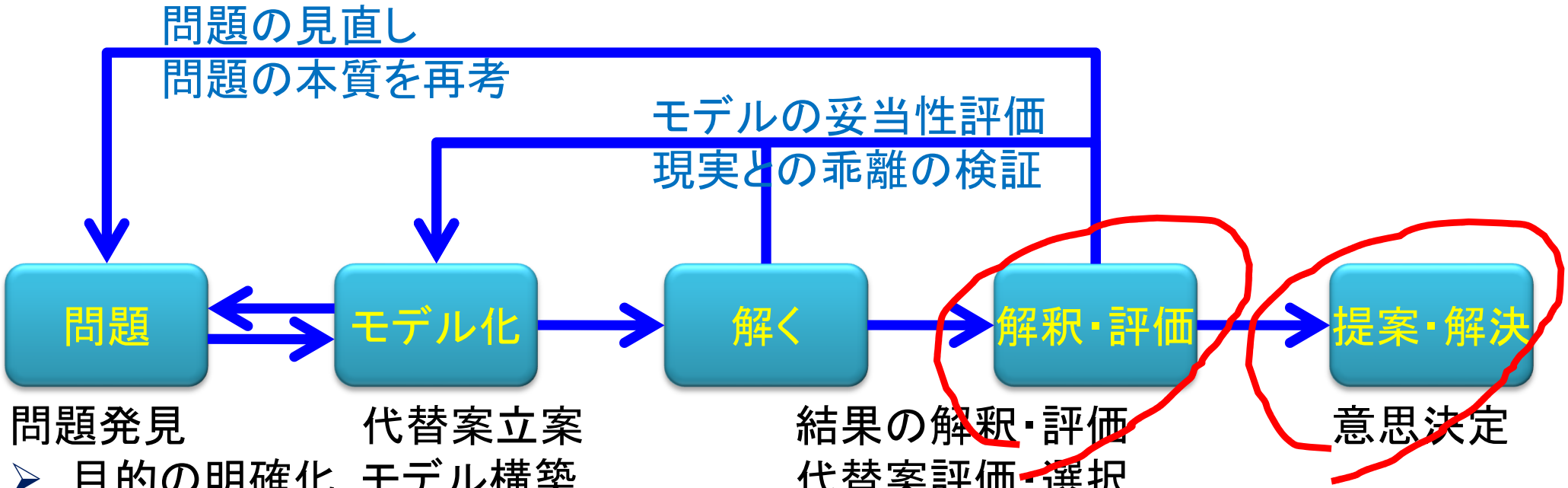
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

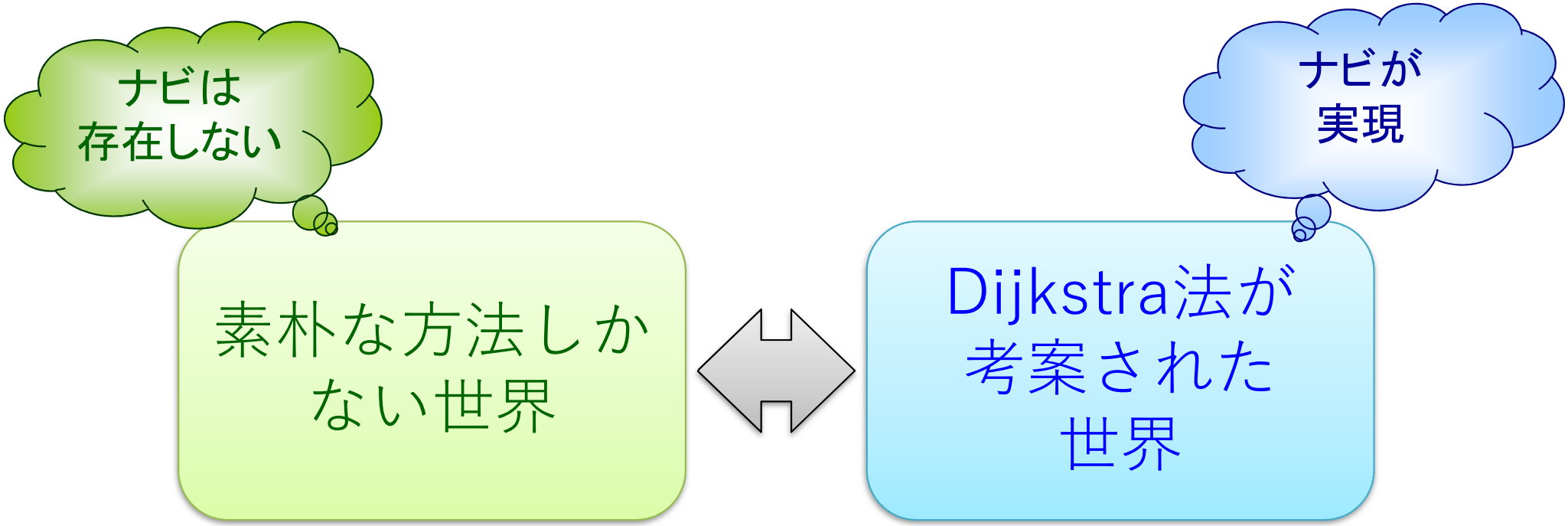
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

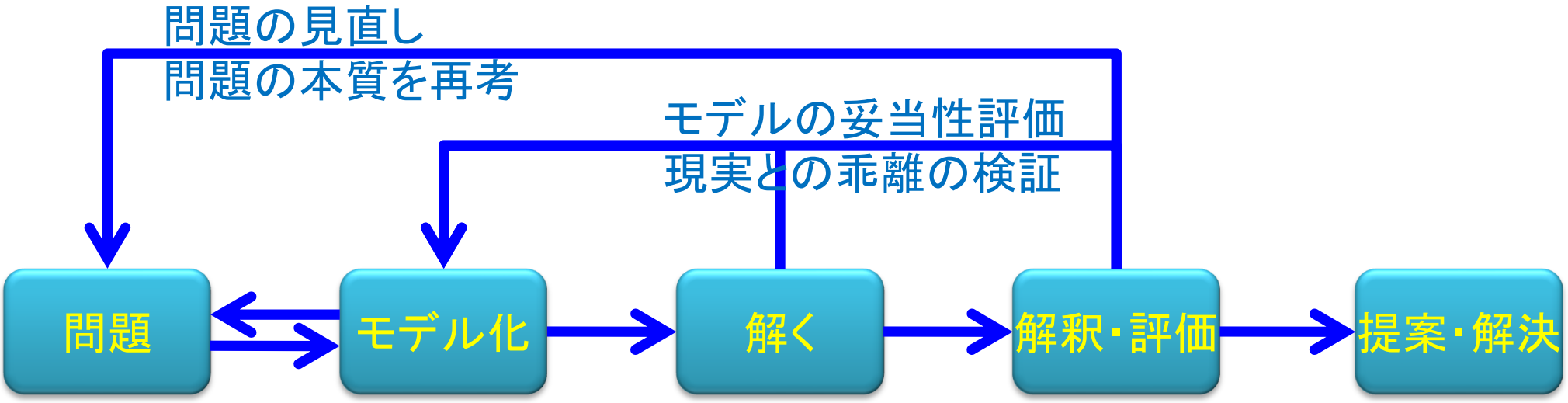
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



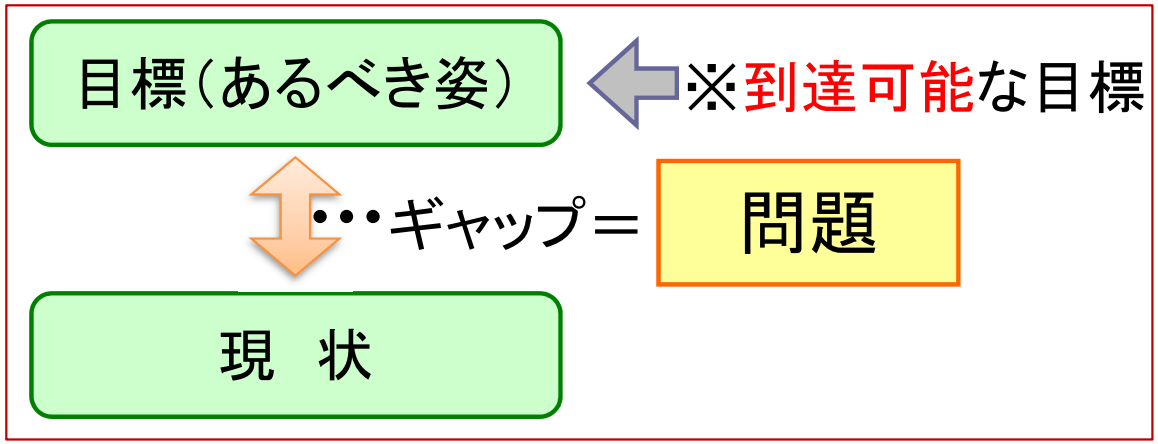
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

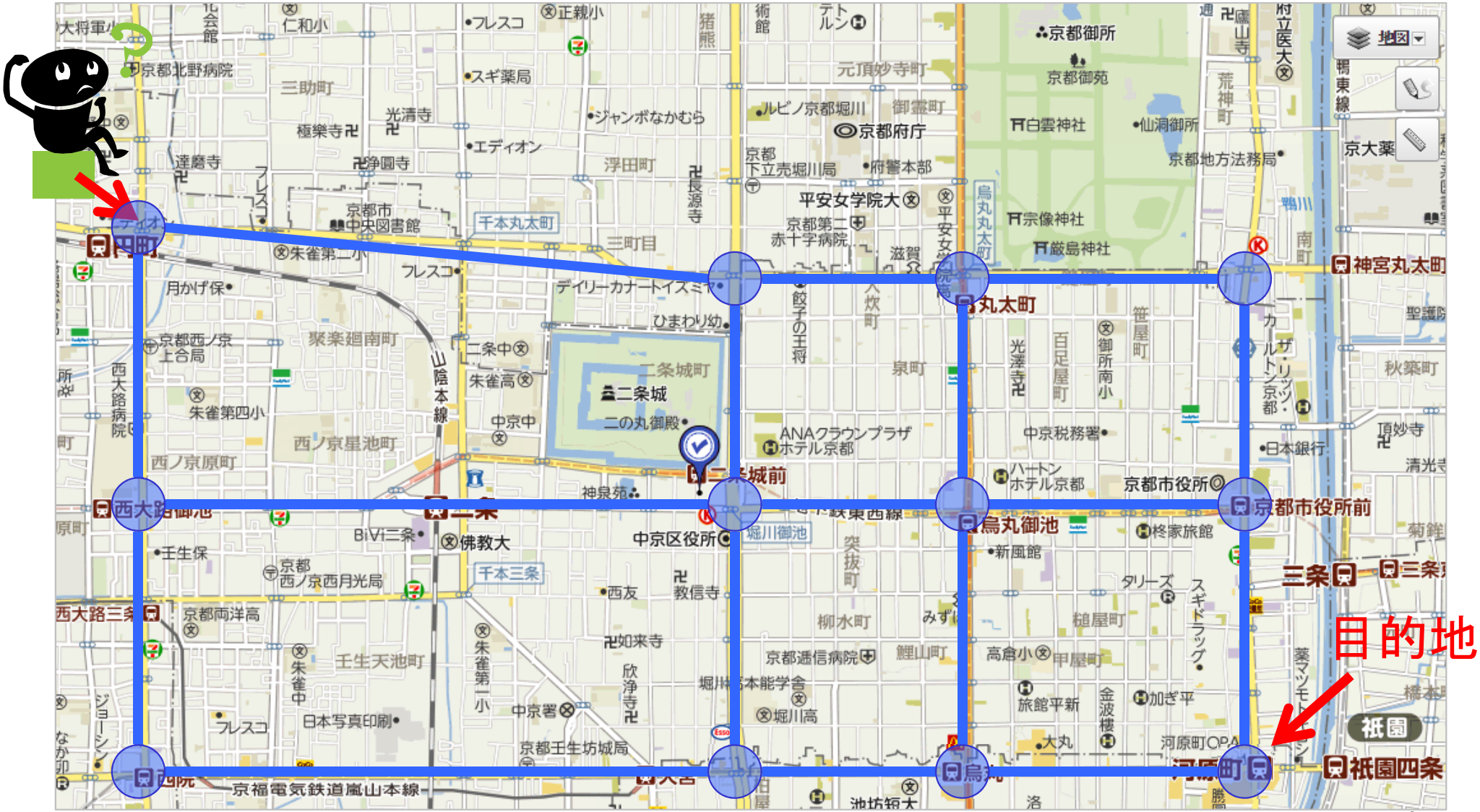
意思決定

問題の定義



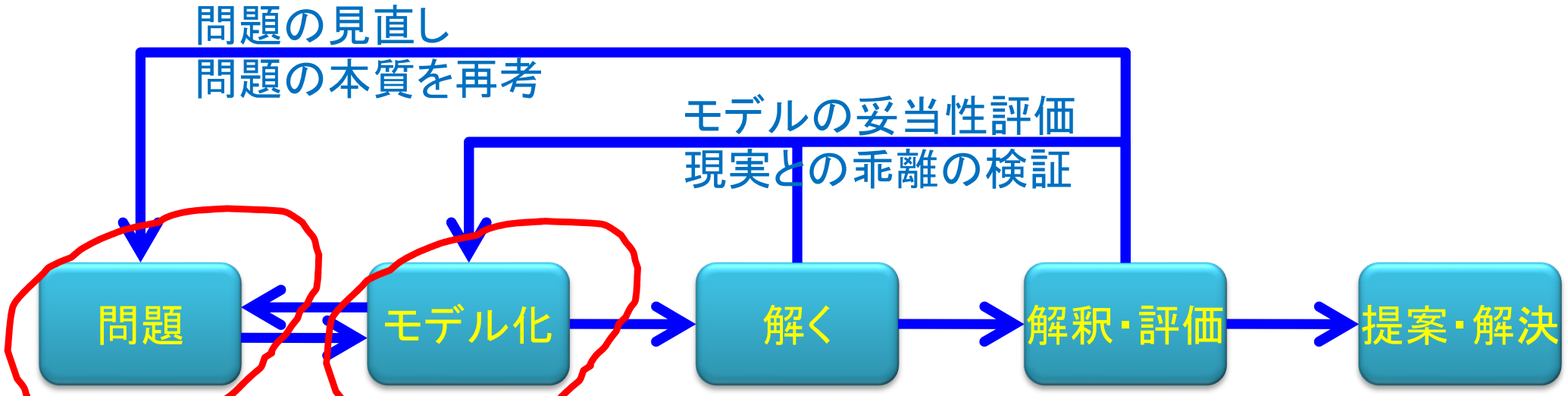
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

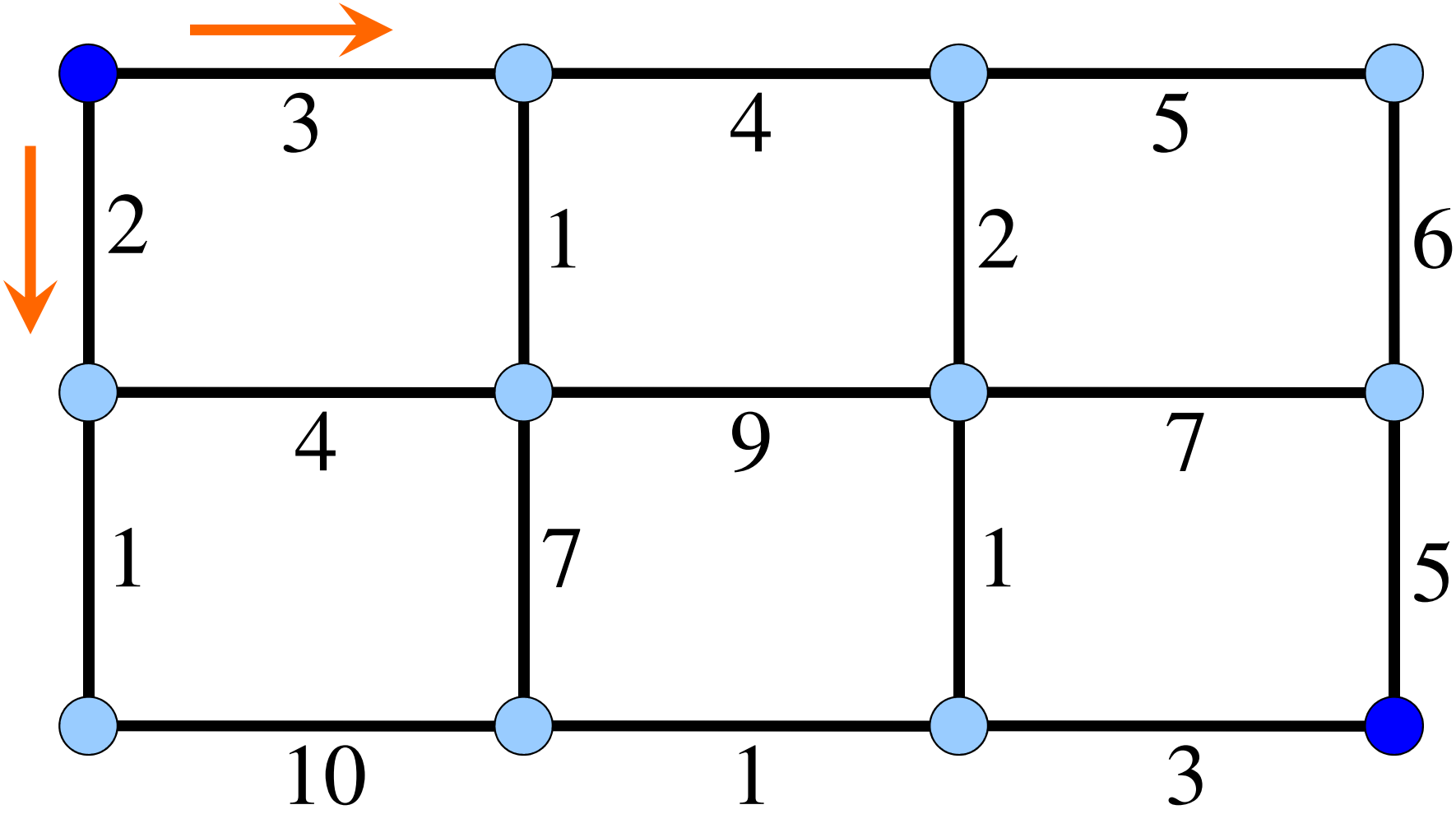
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

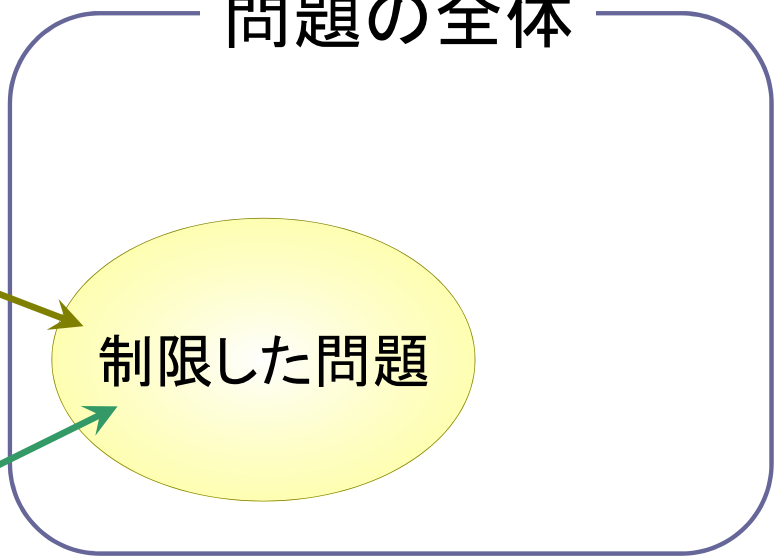


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

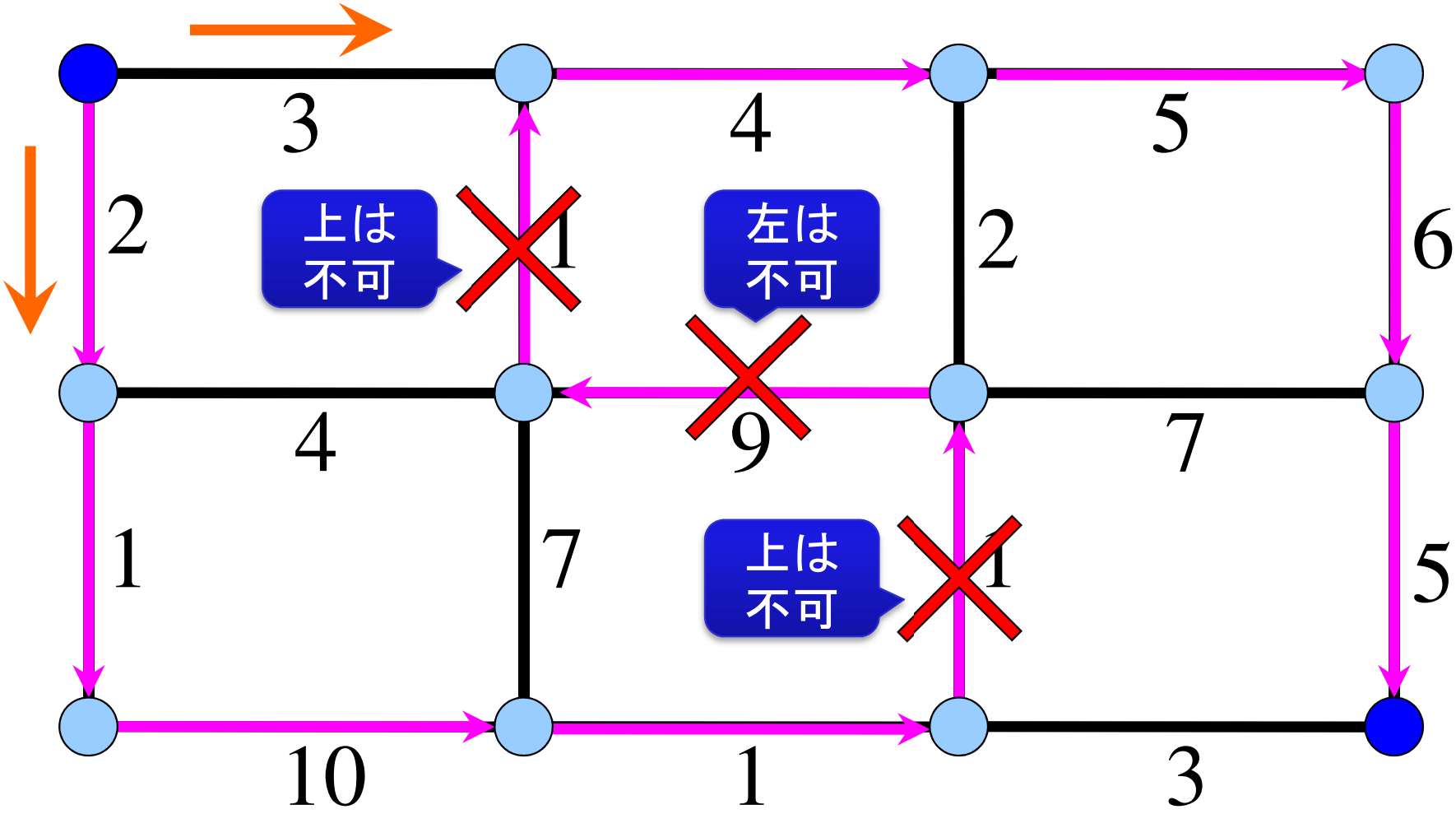
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

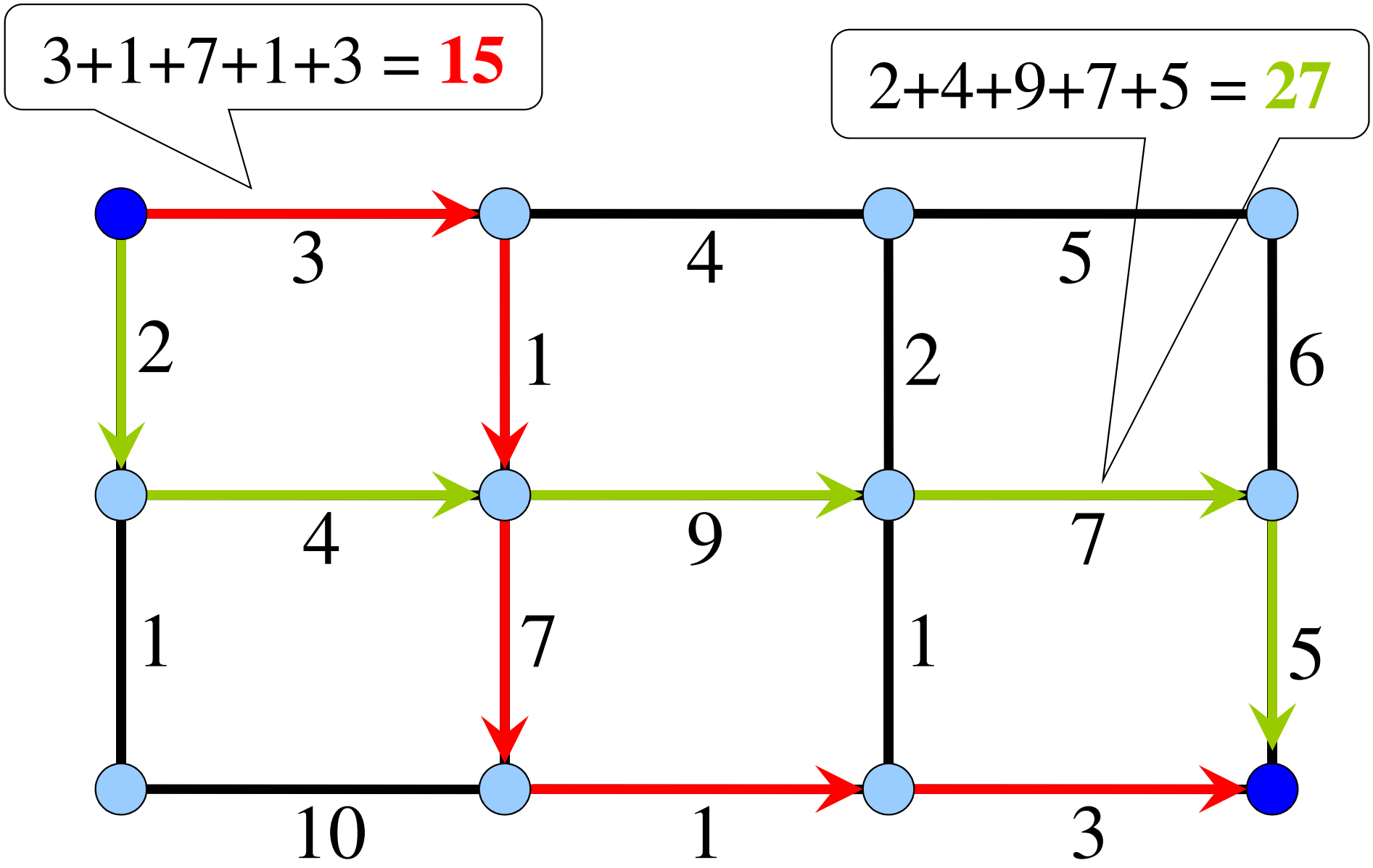
難しいなら易しくすればいいのさ！

制限した問題

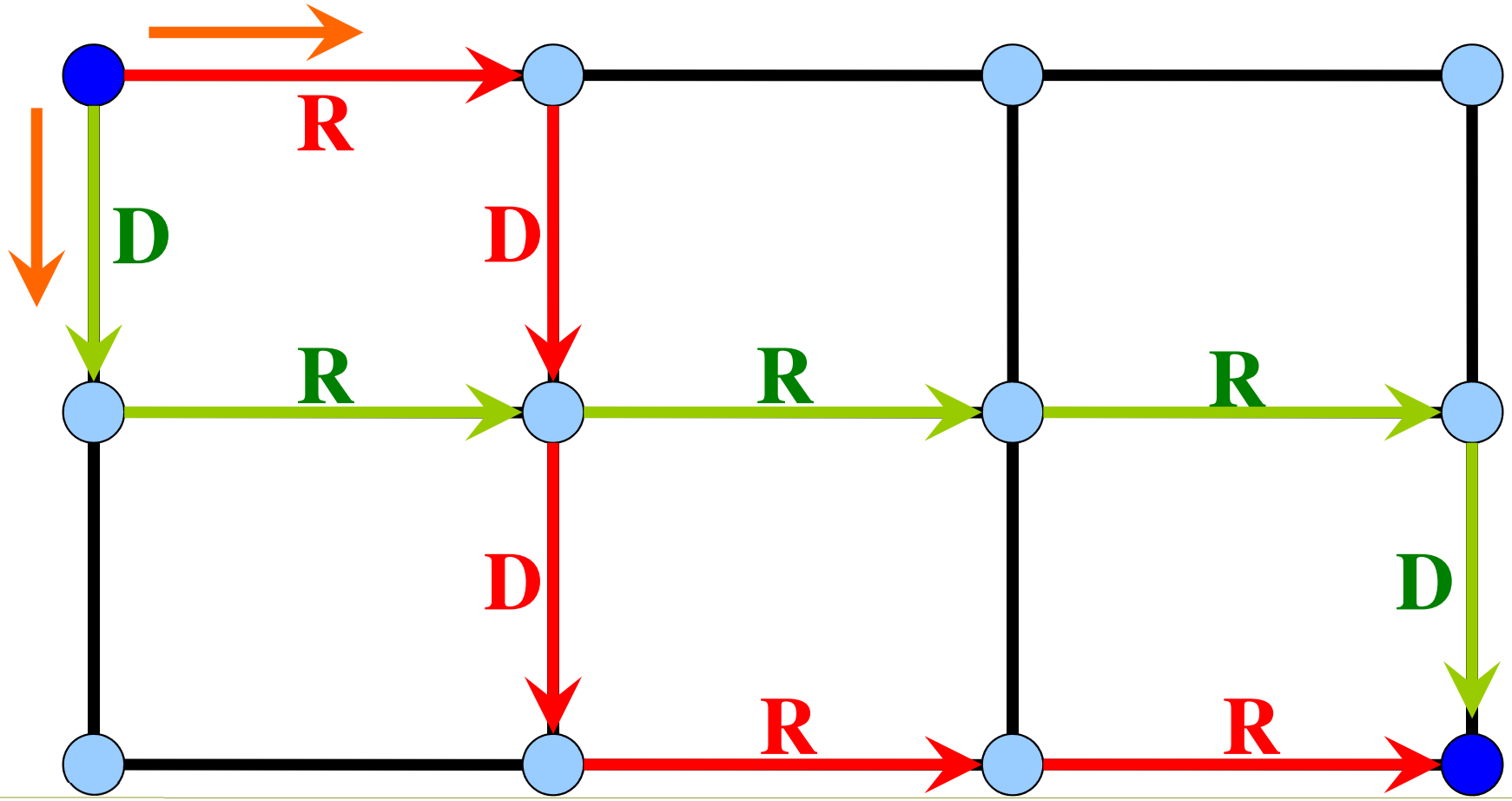
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



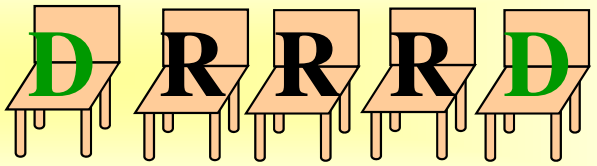
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



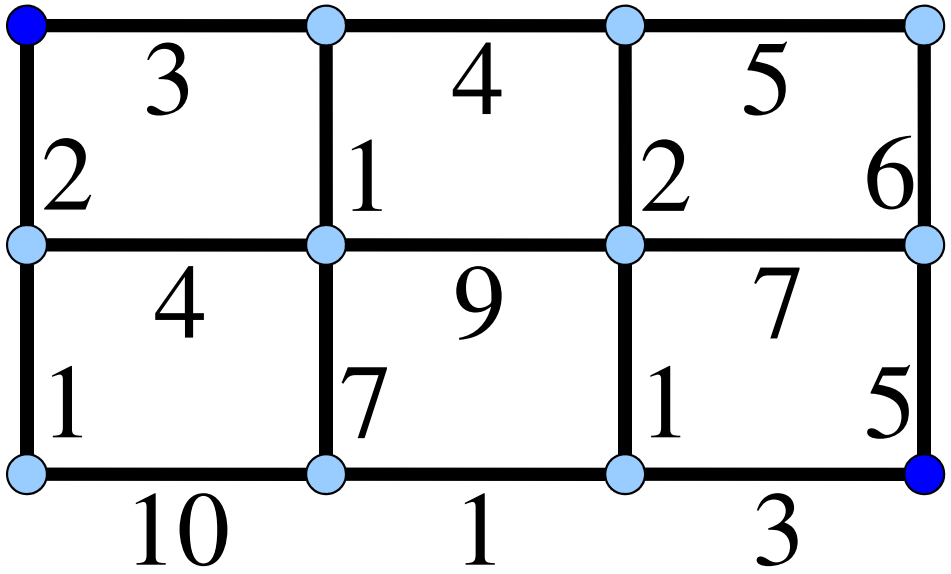
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

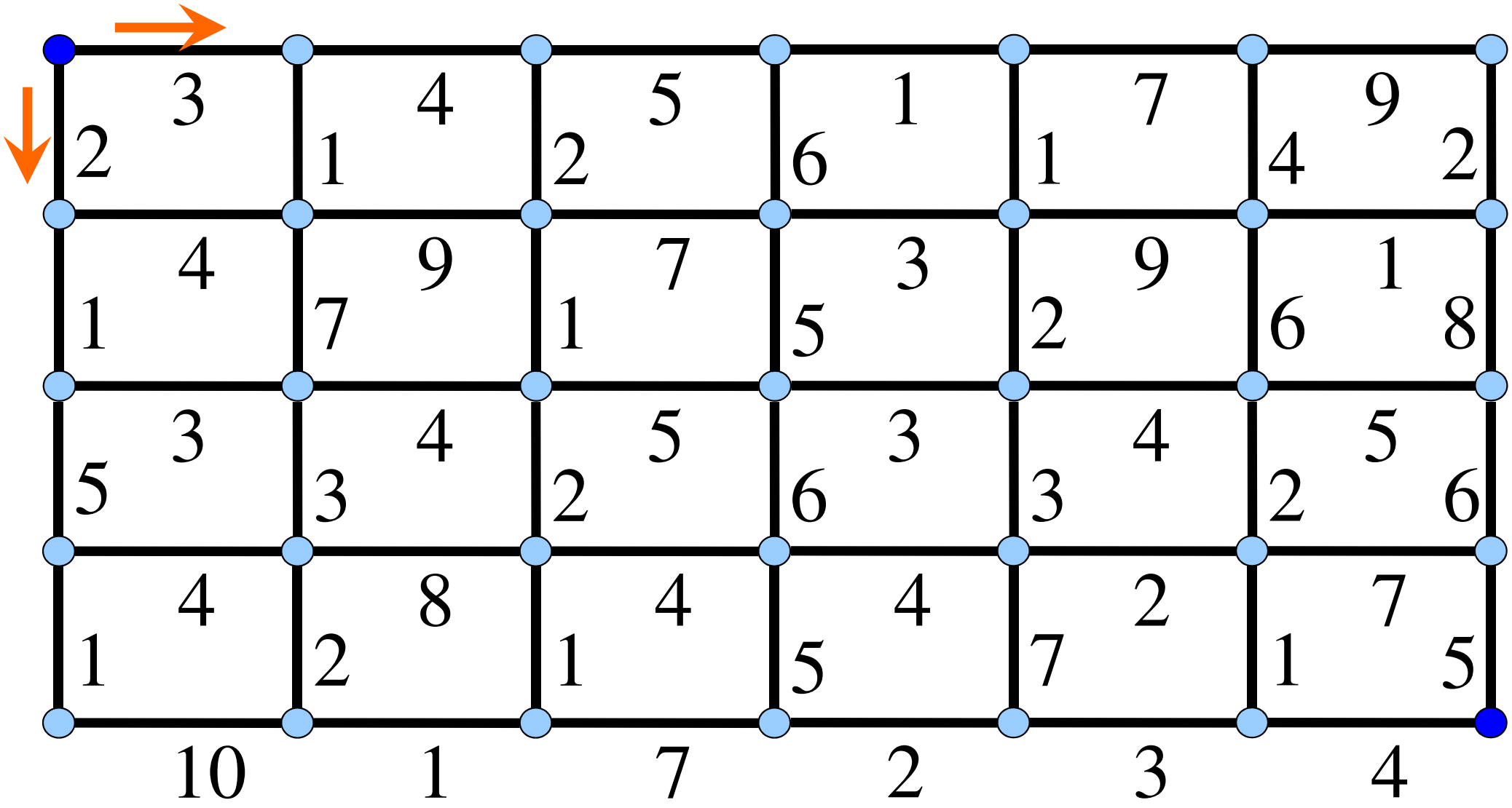
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

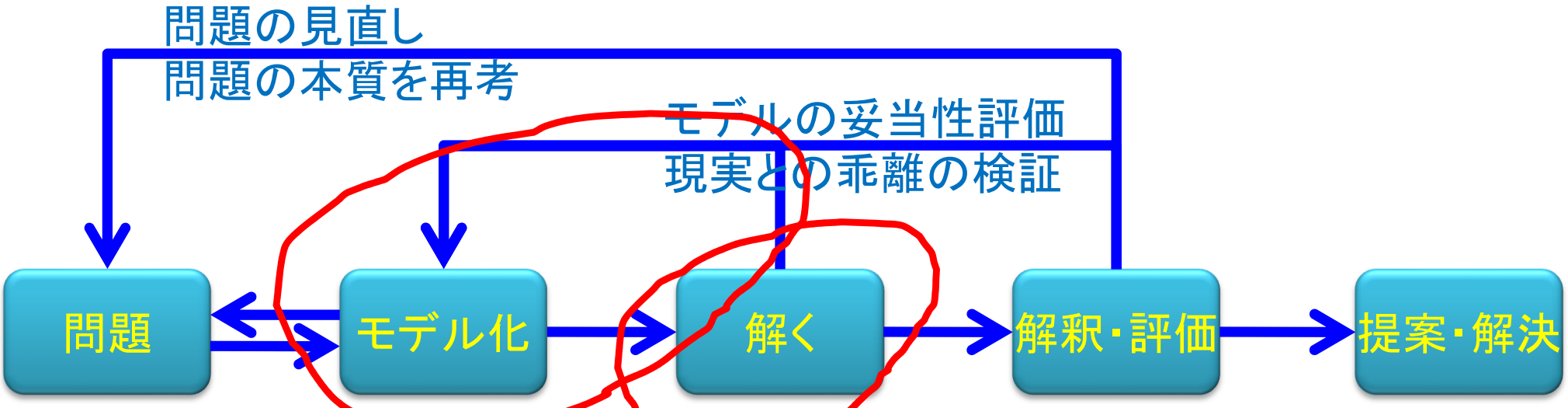
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

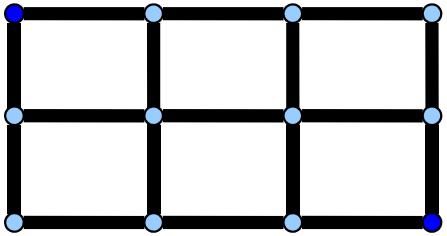
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

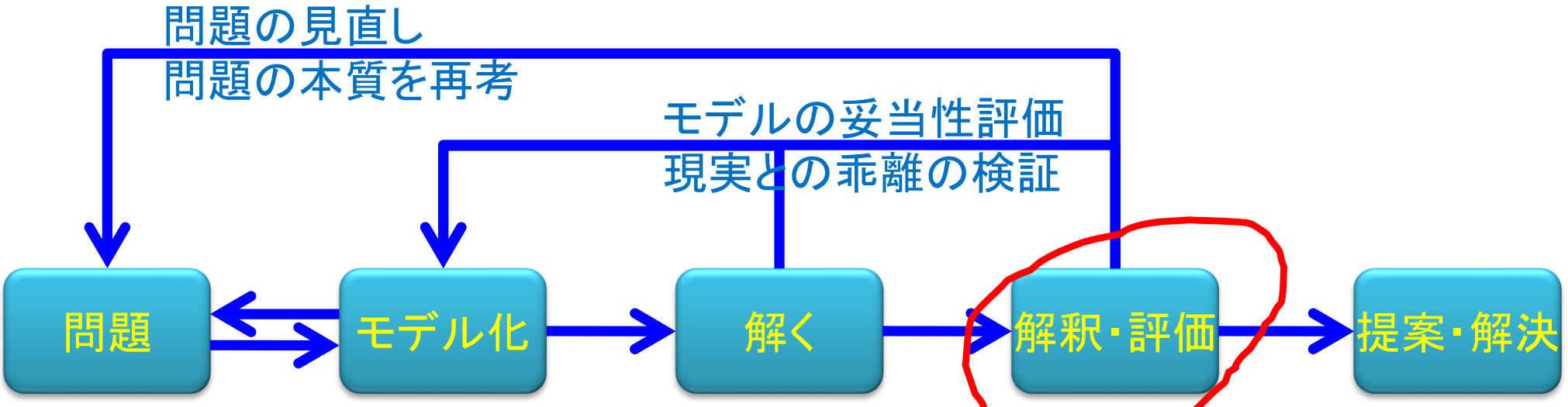
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

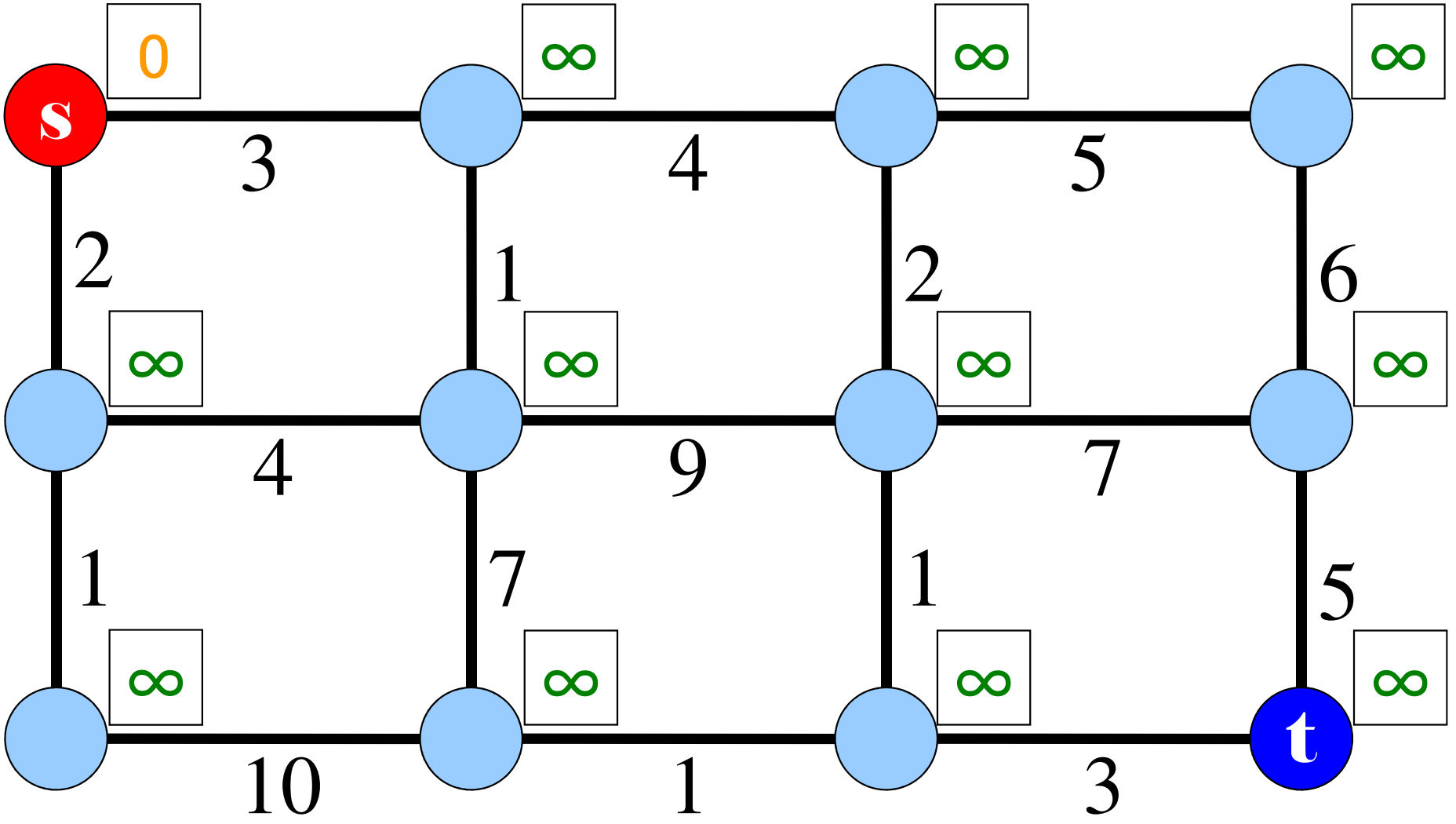
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

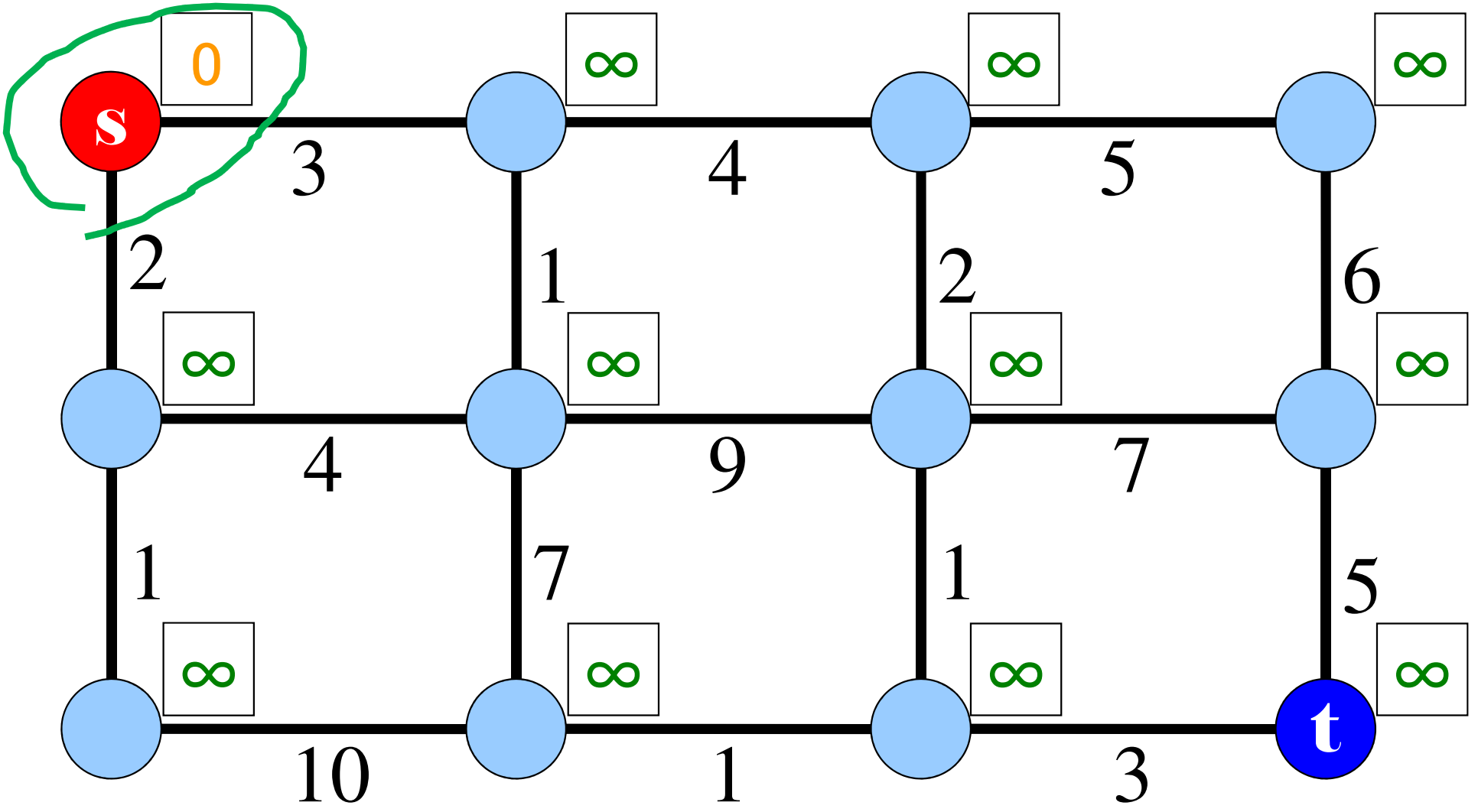
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



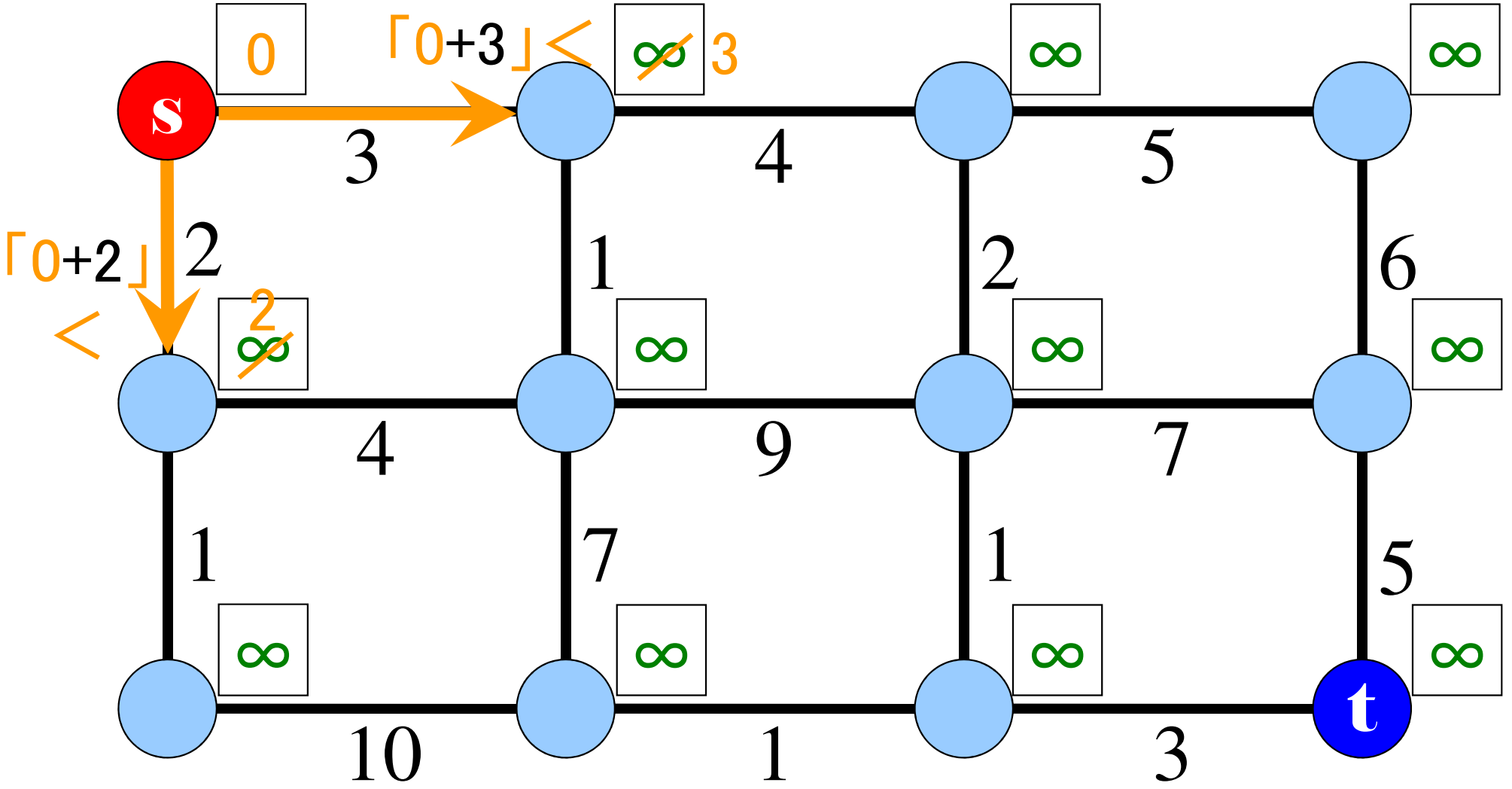
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

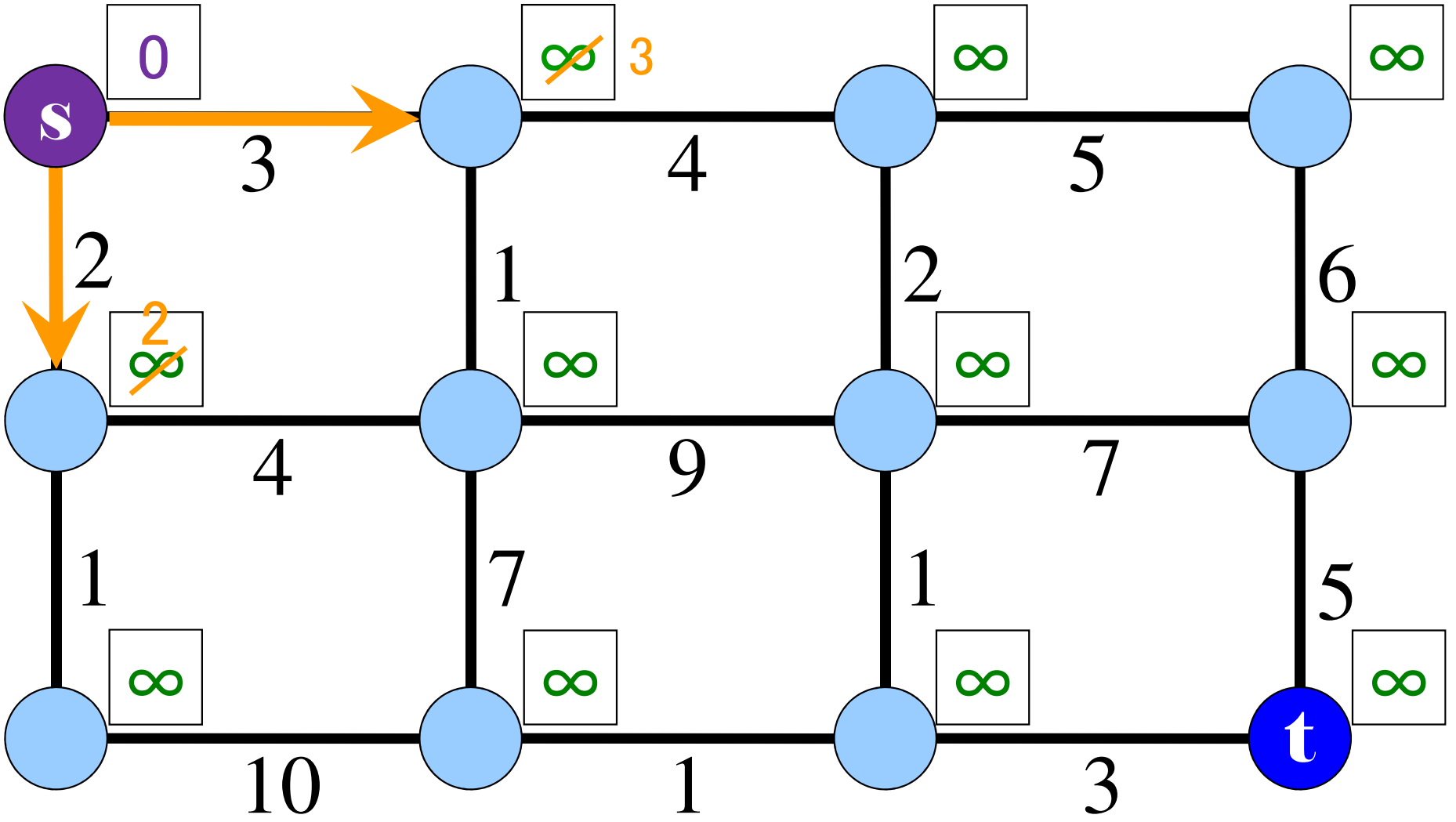


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

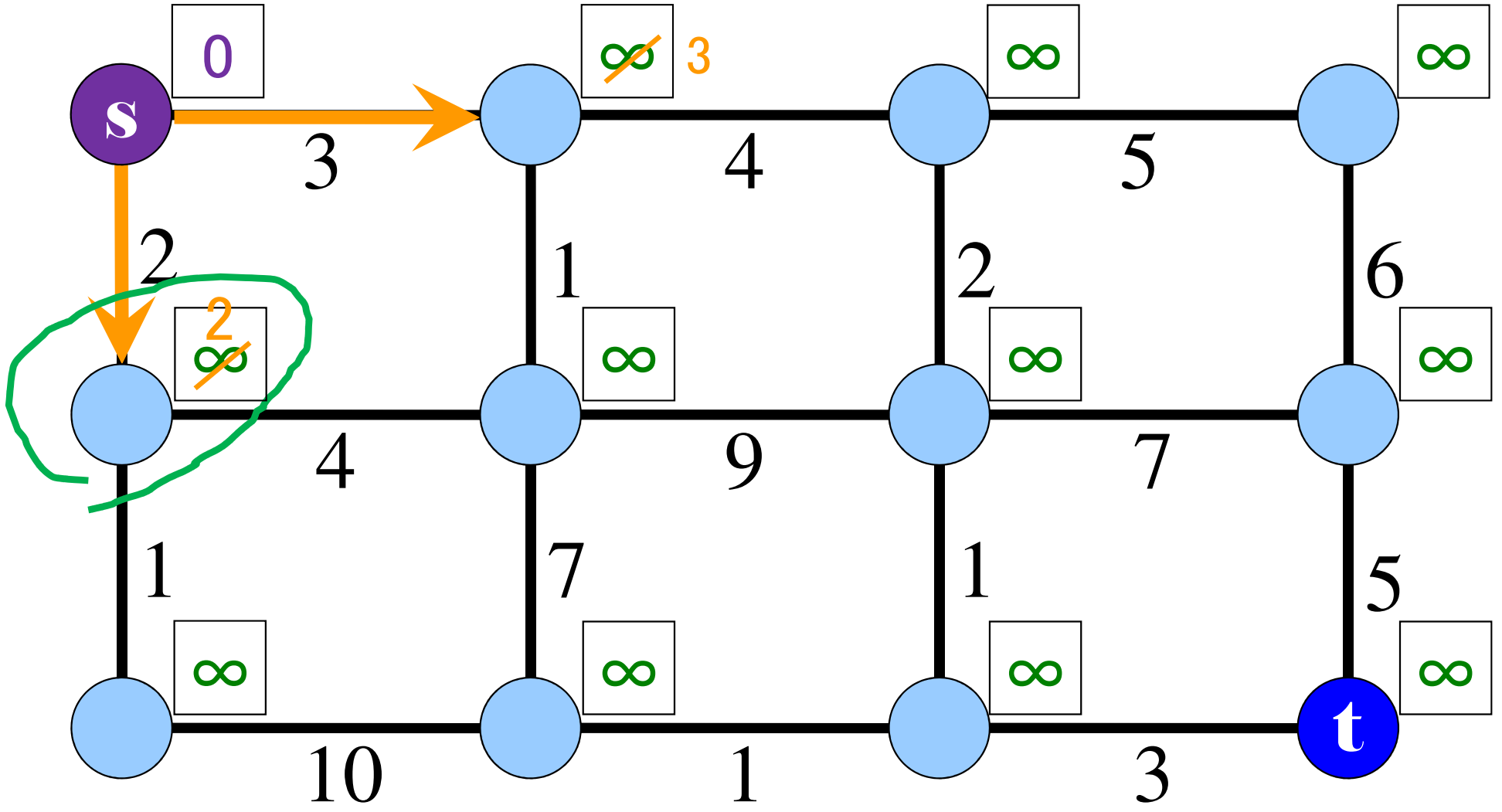
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



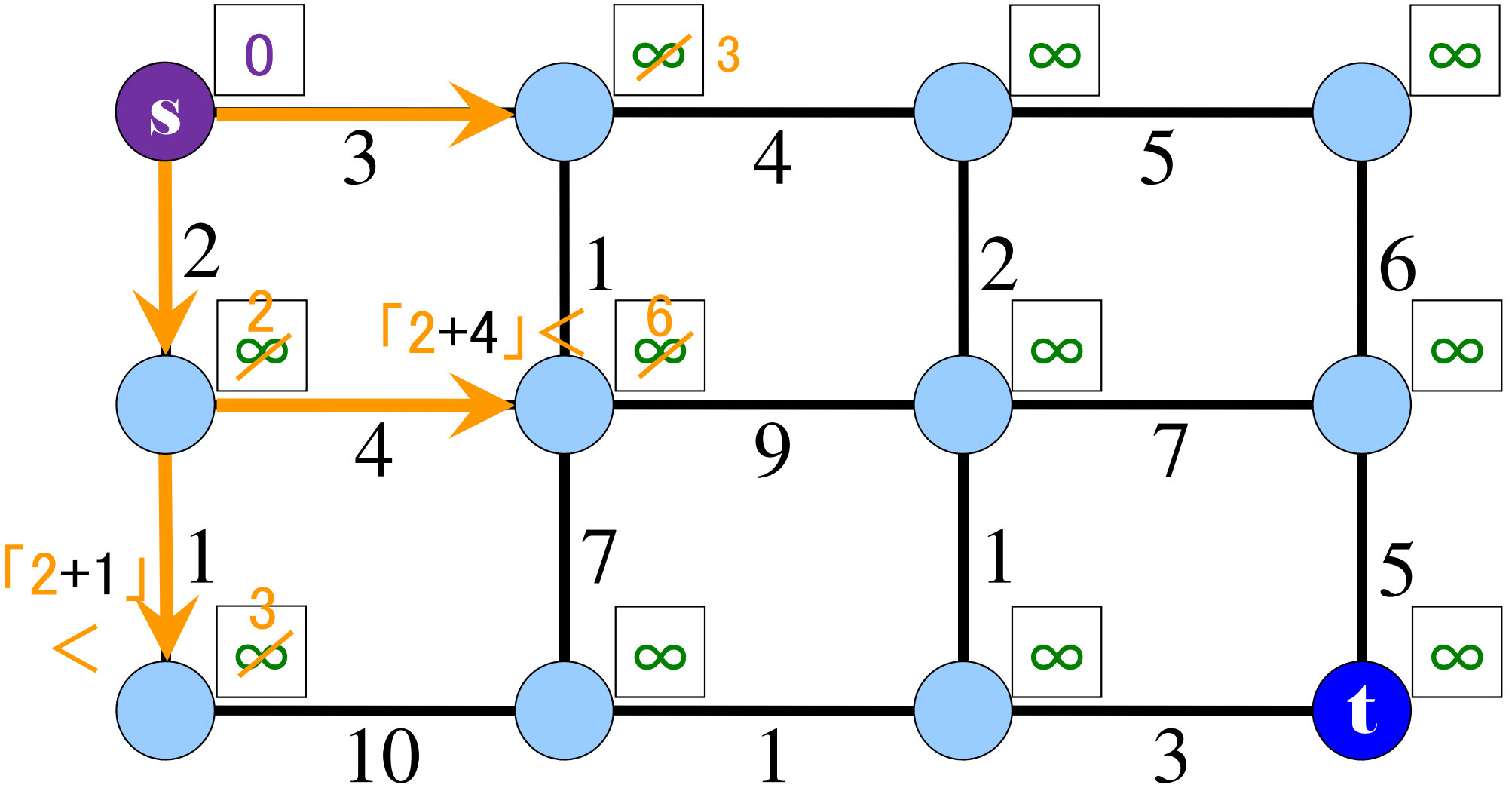
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

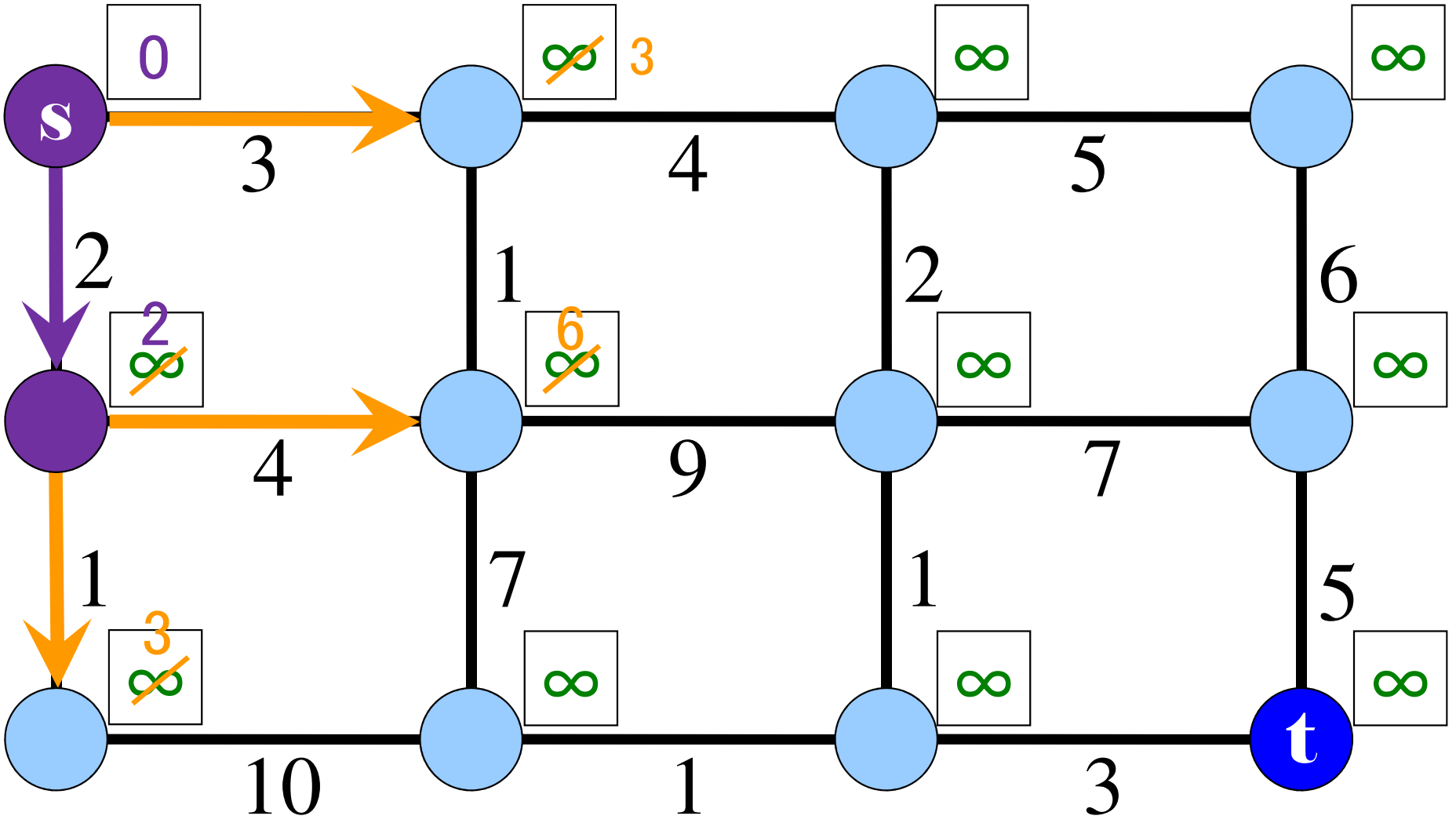


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

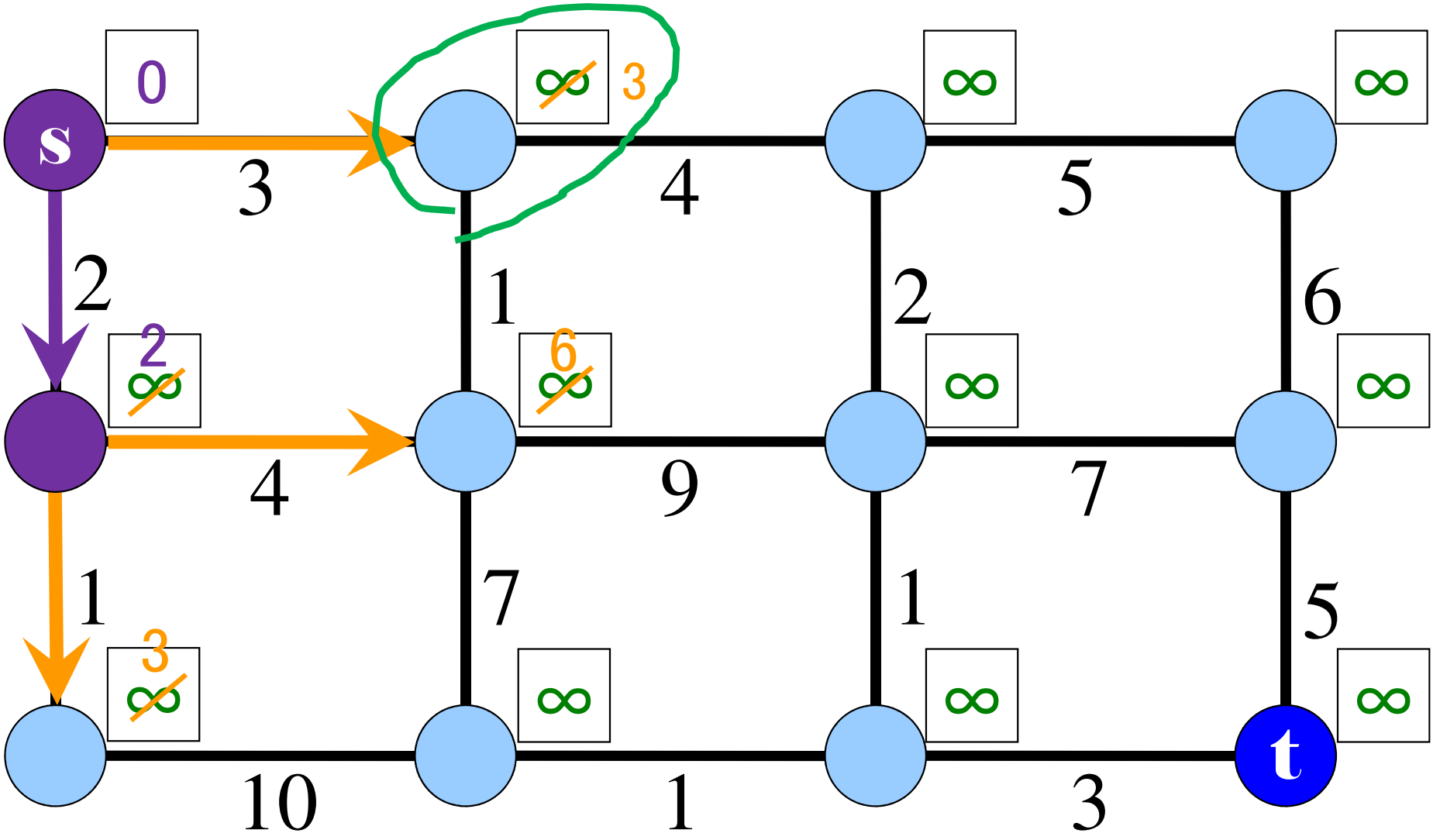
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



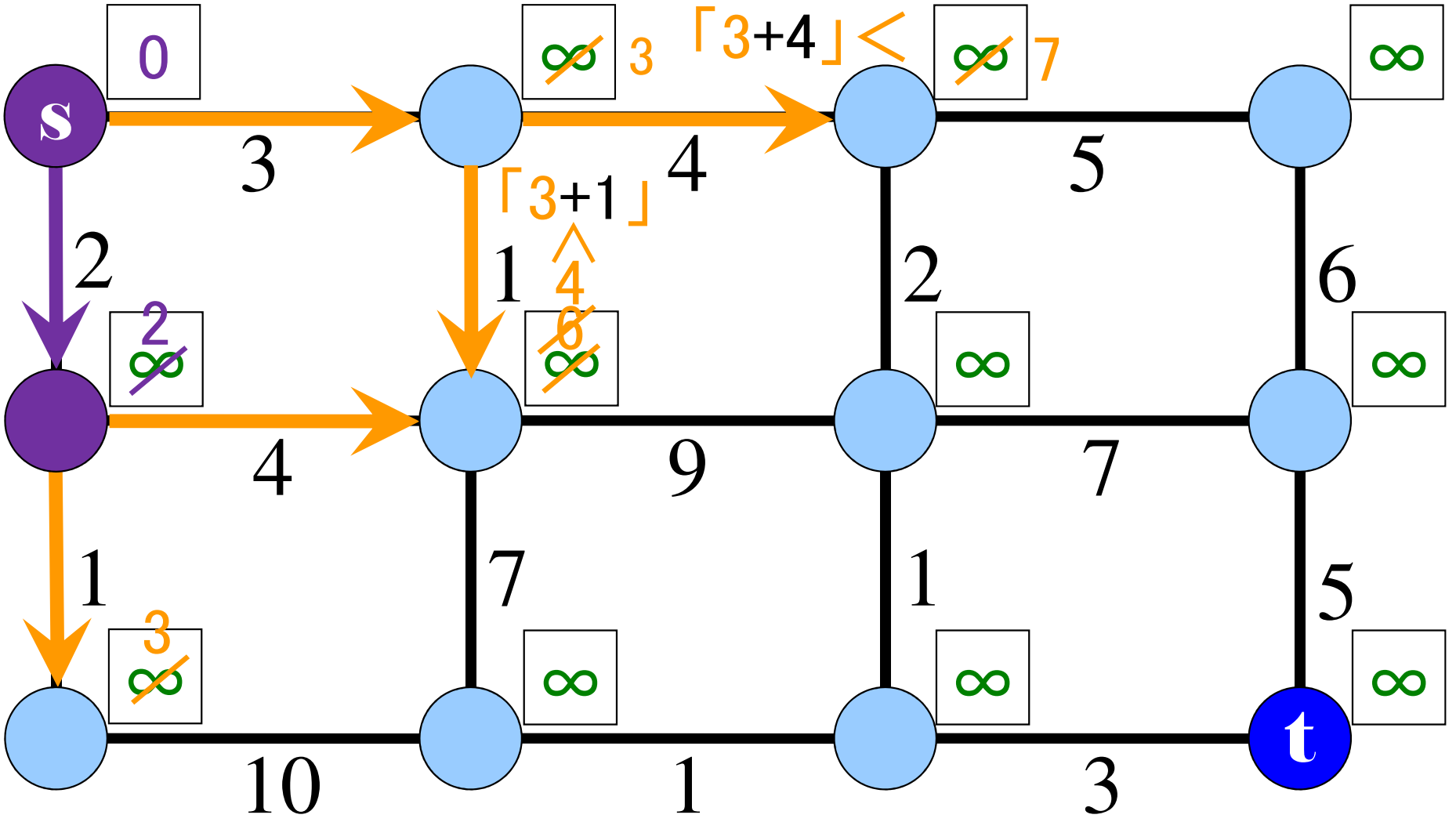
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

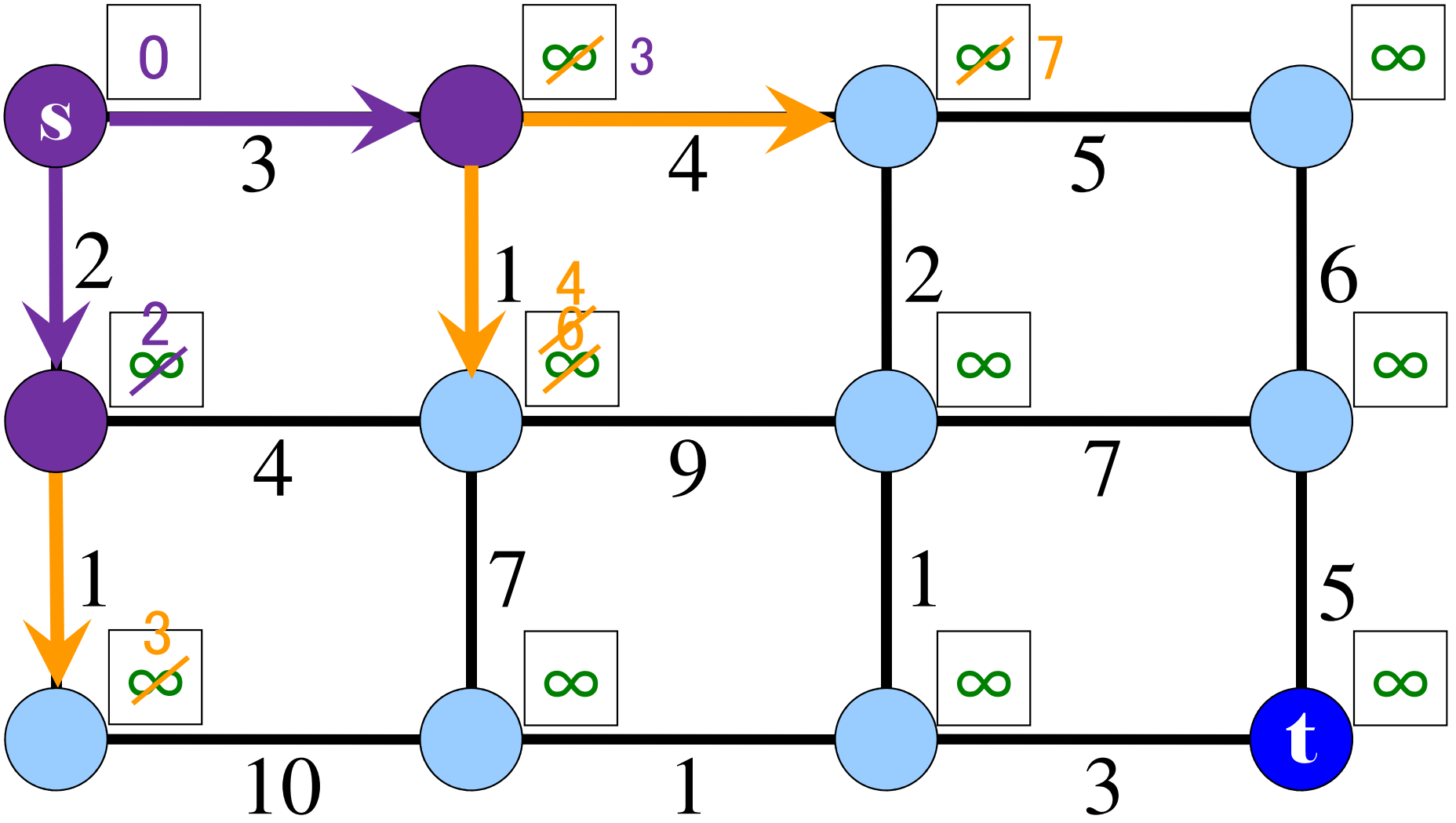


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

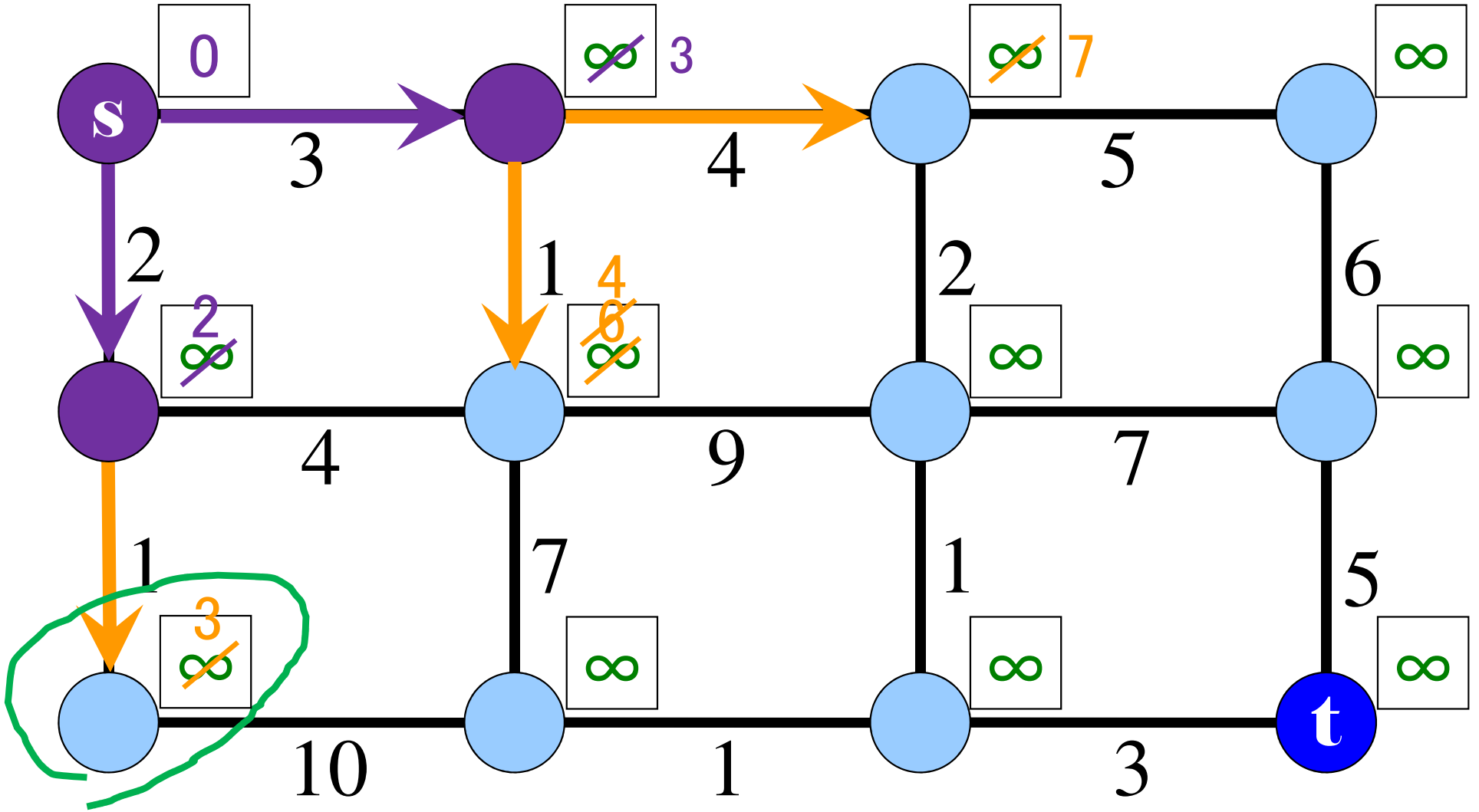
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



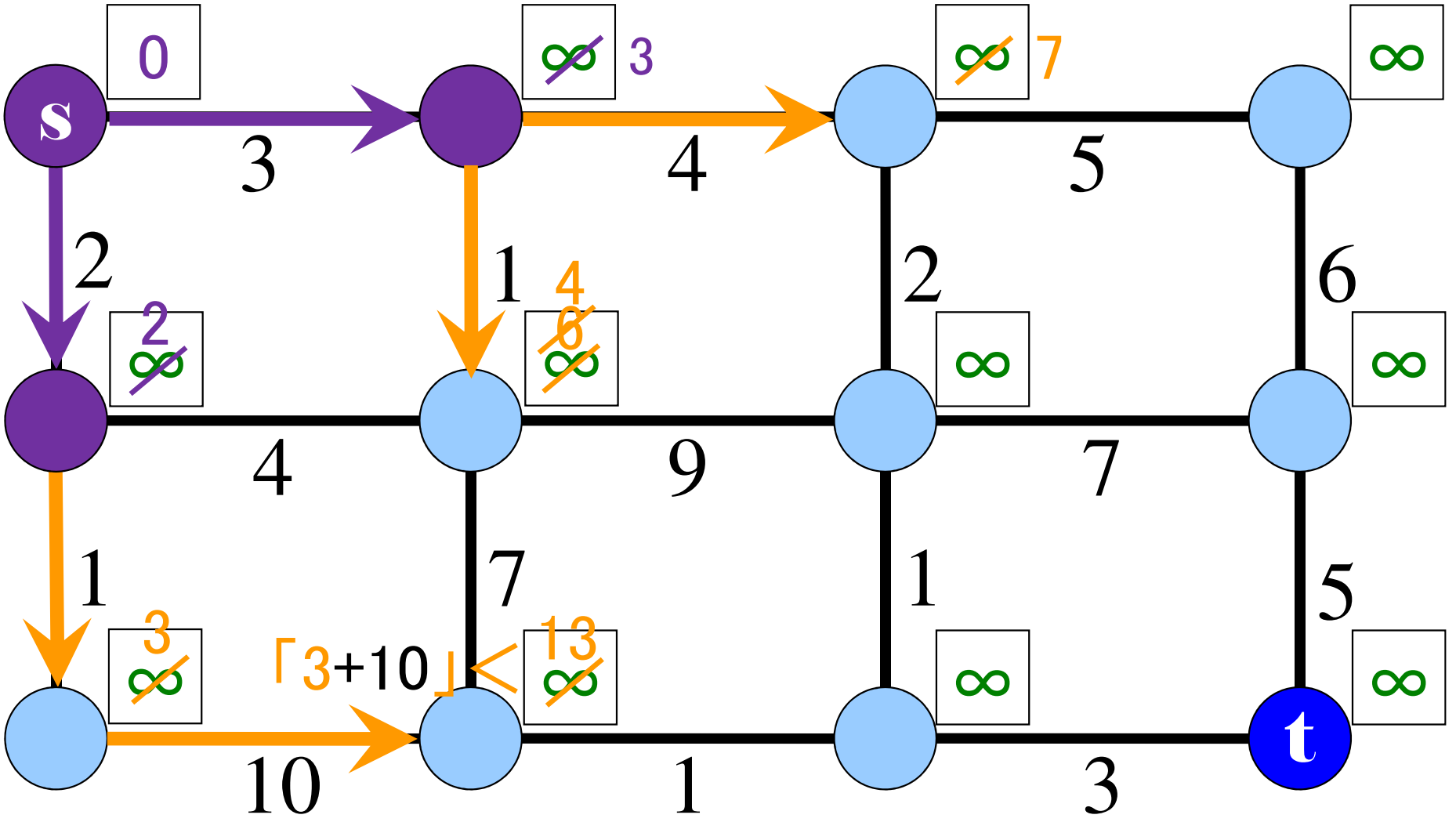
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

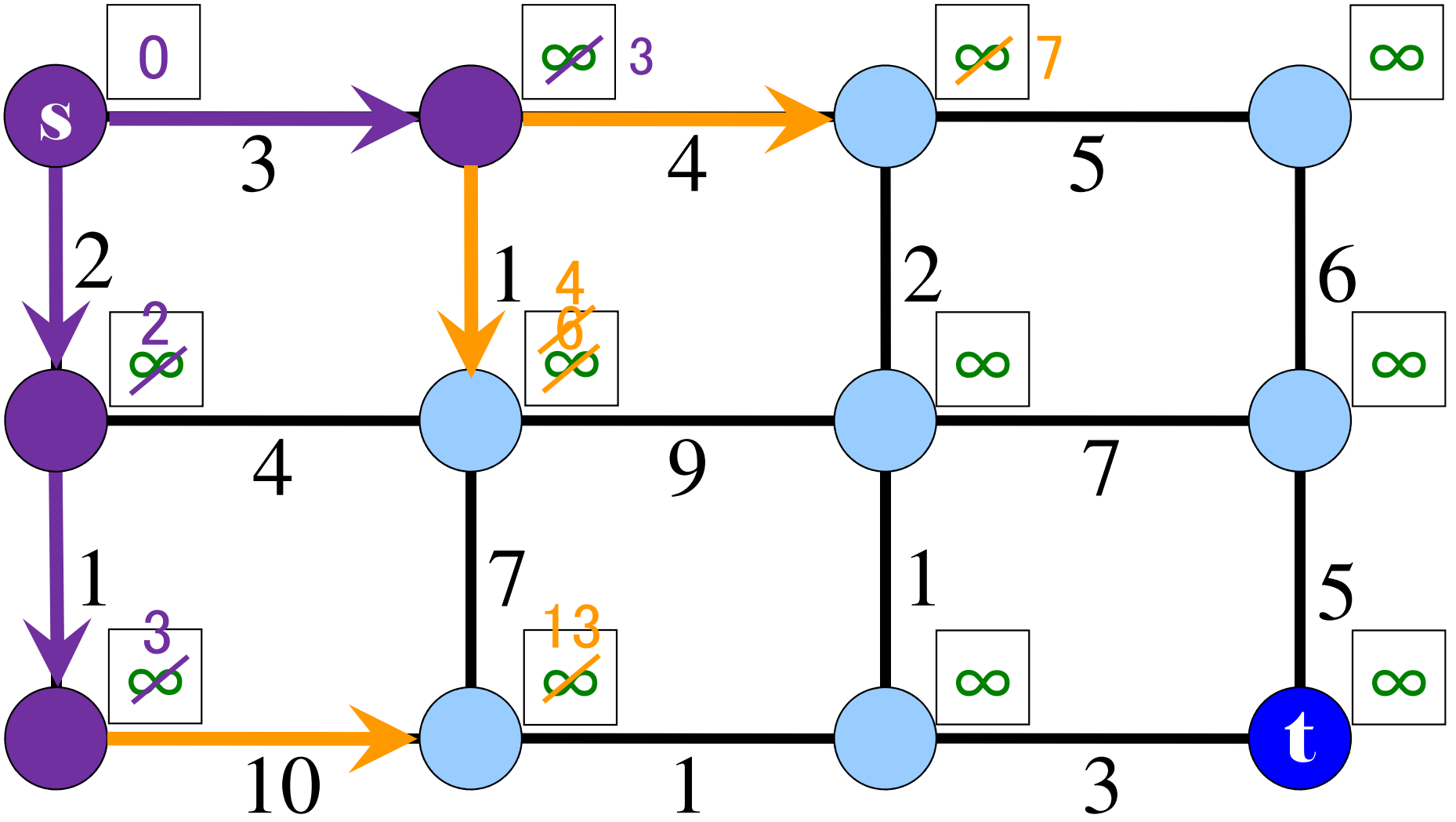


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

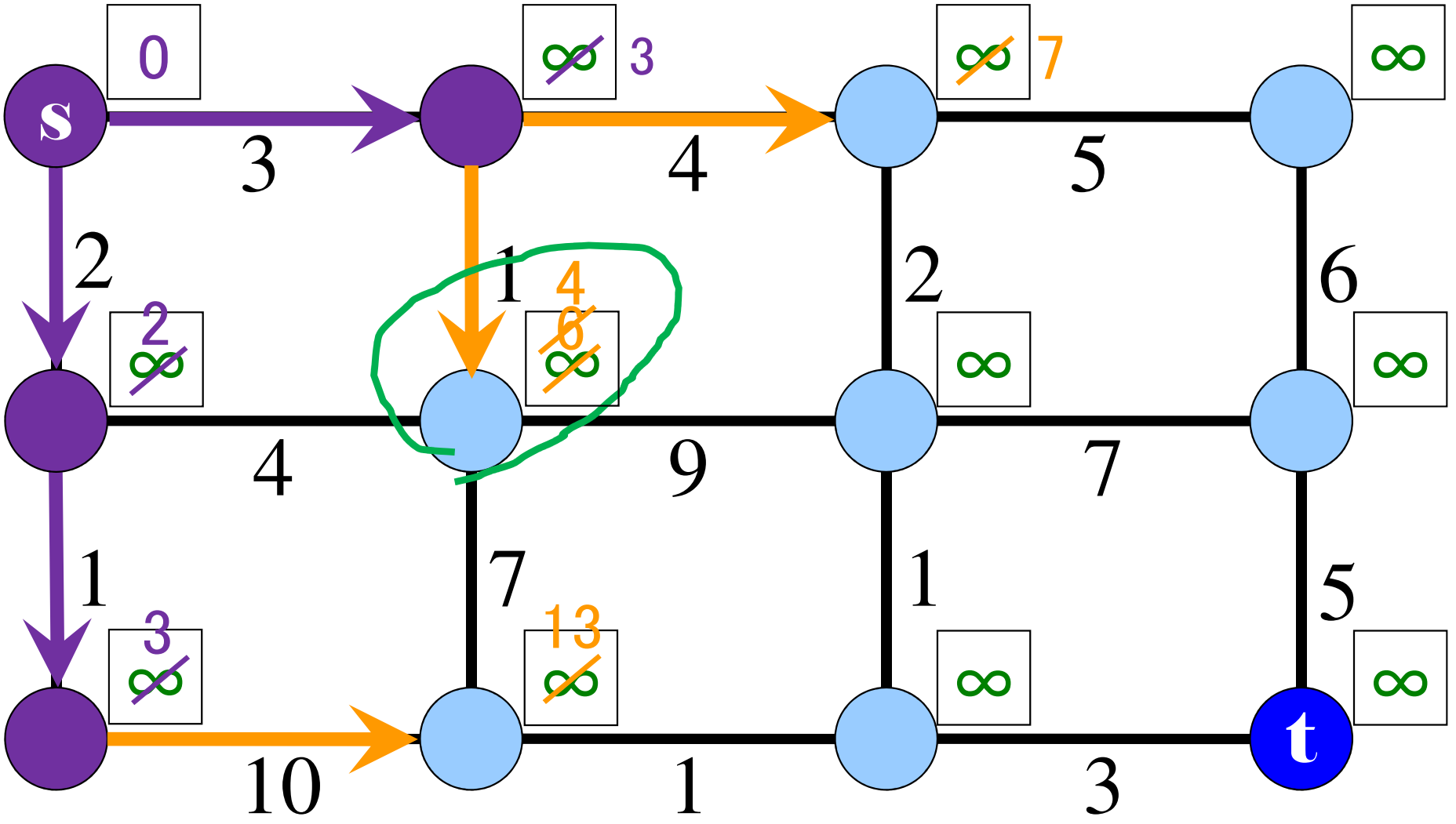
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



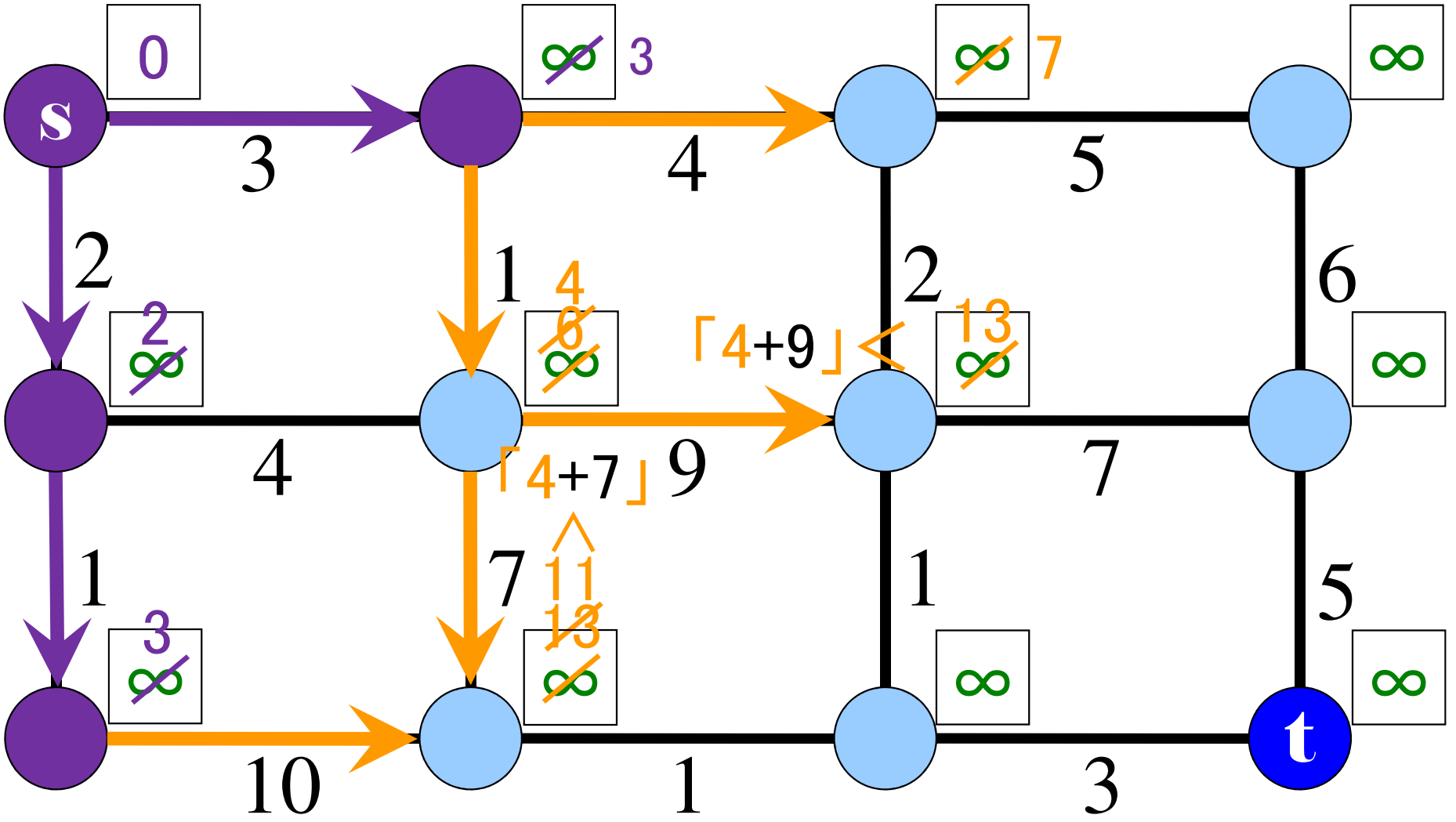
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

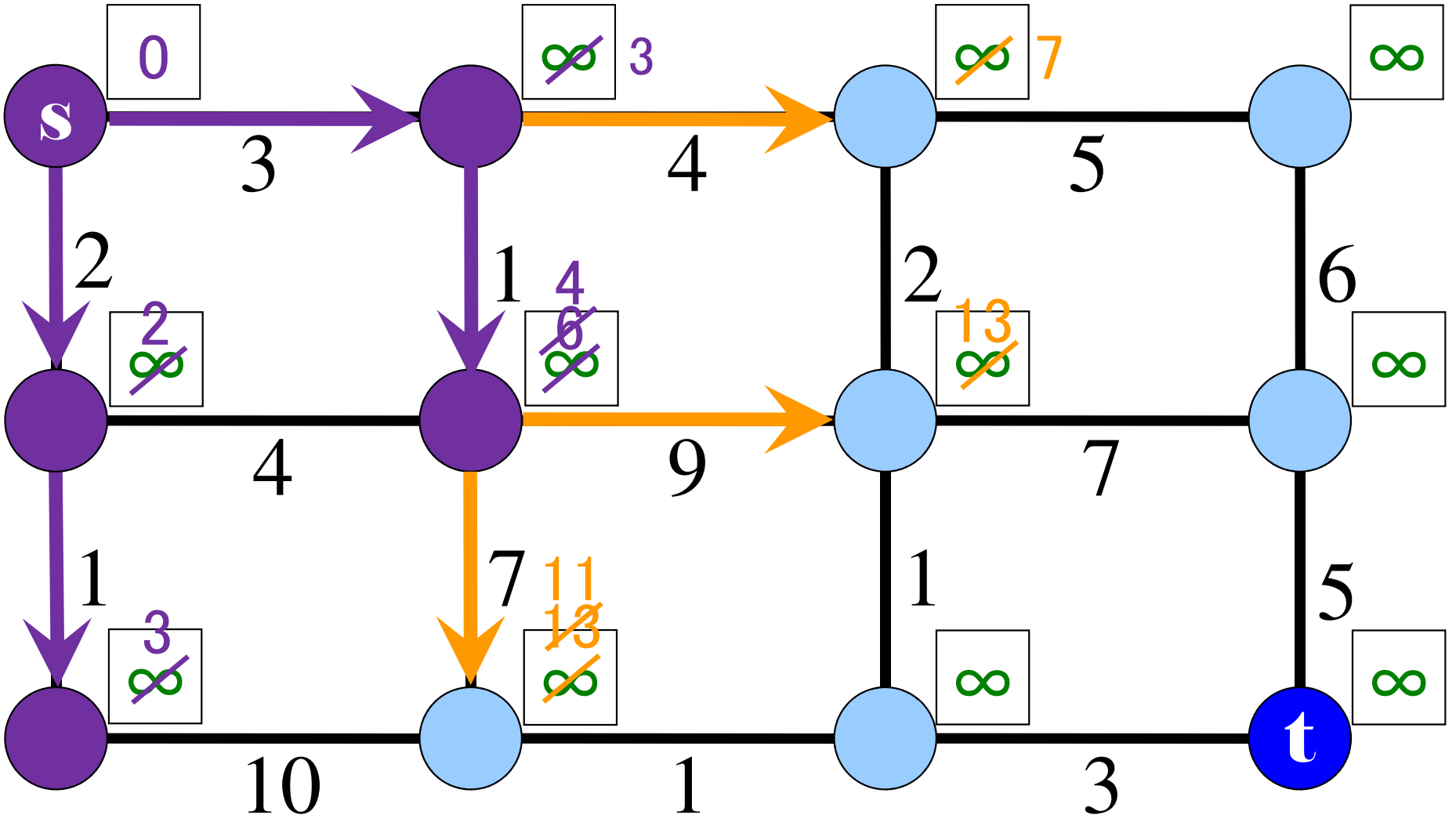


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

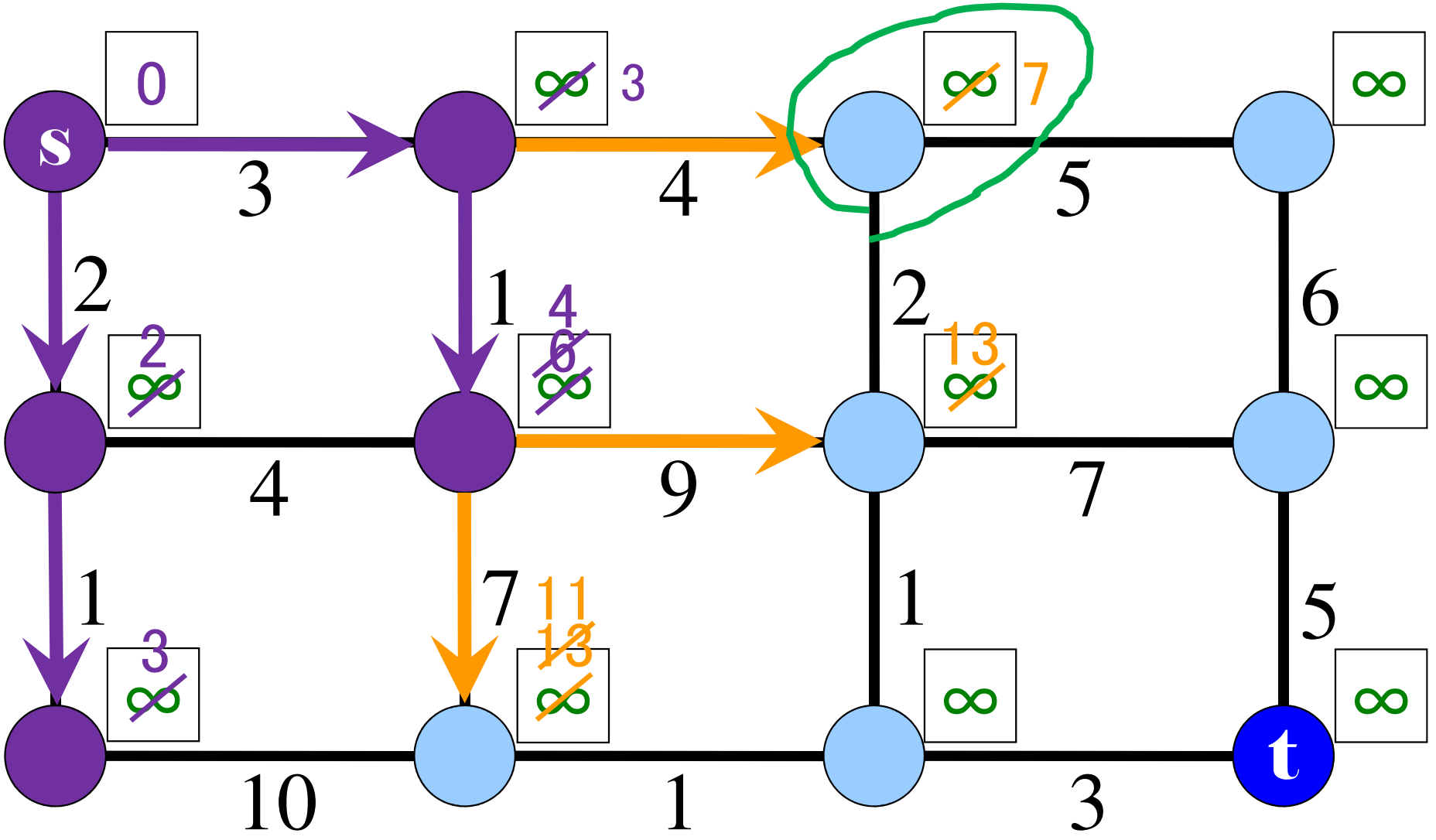
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



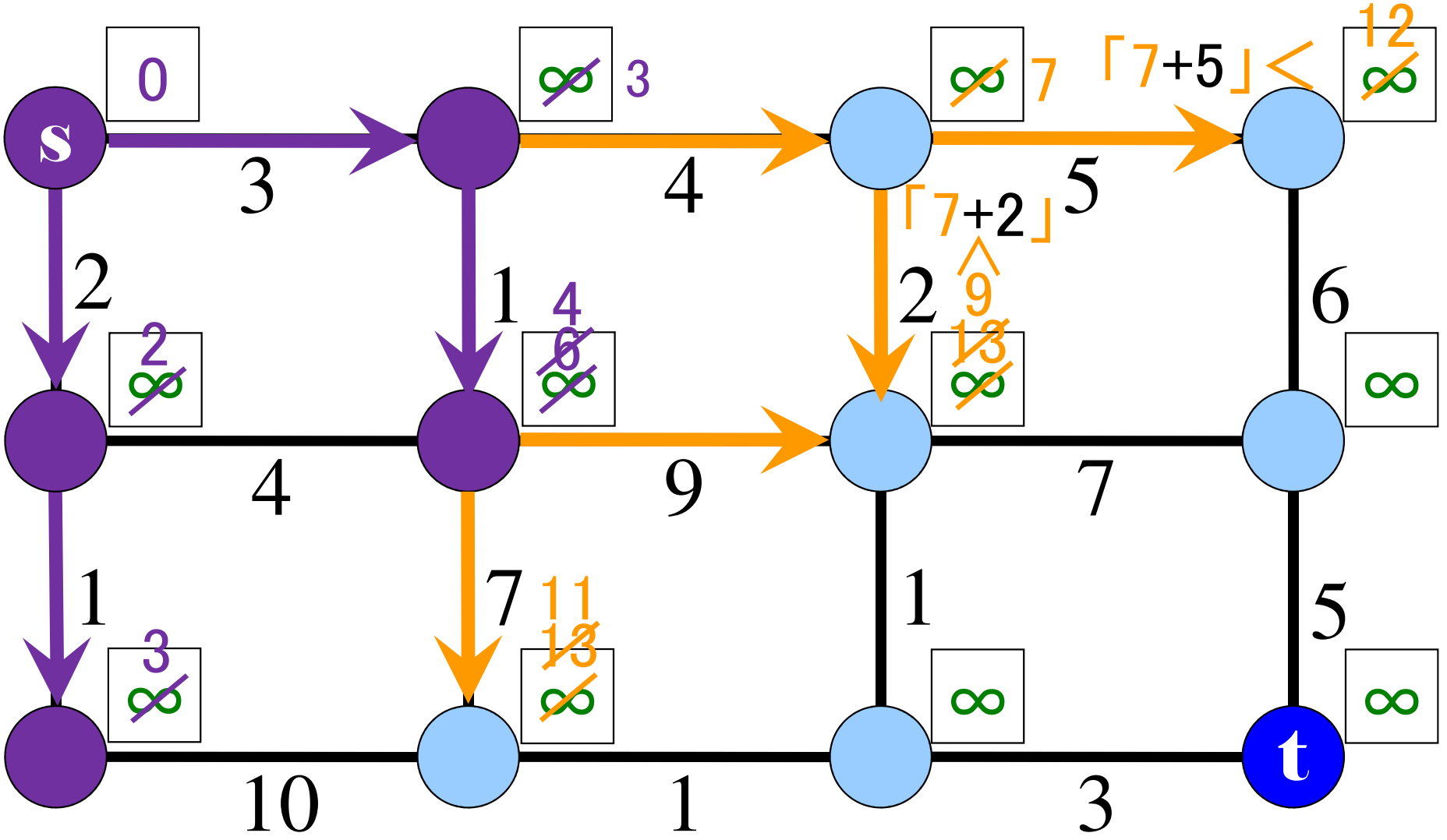
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

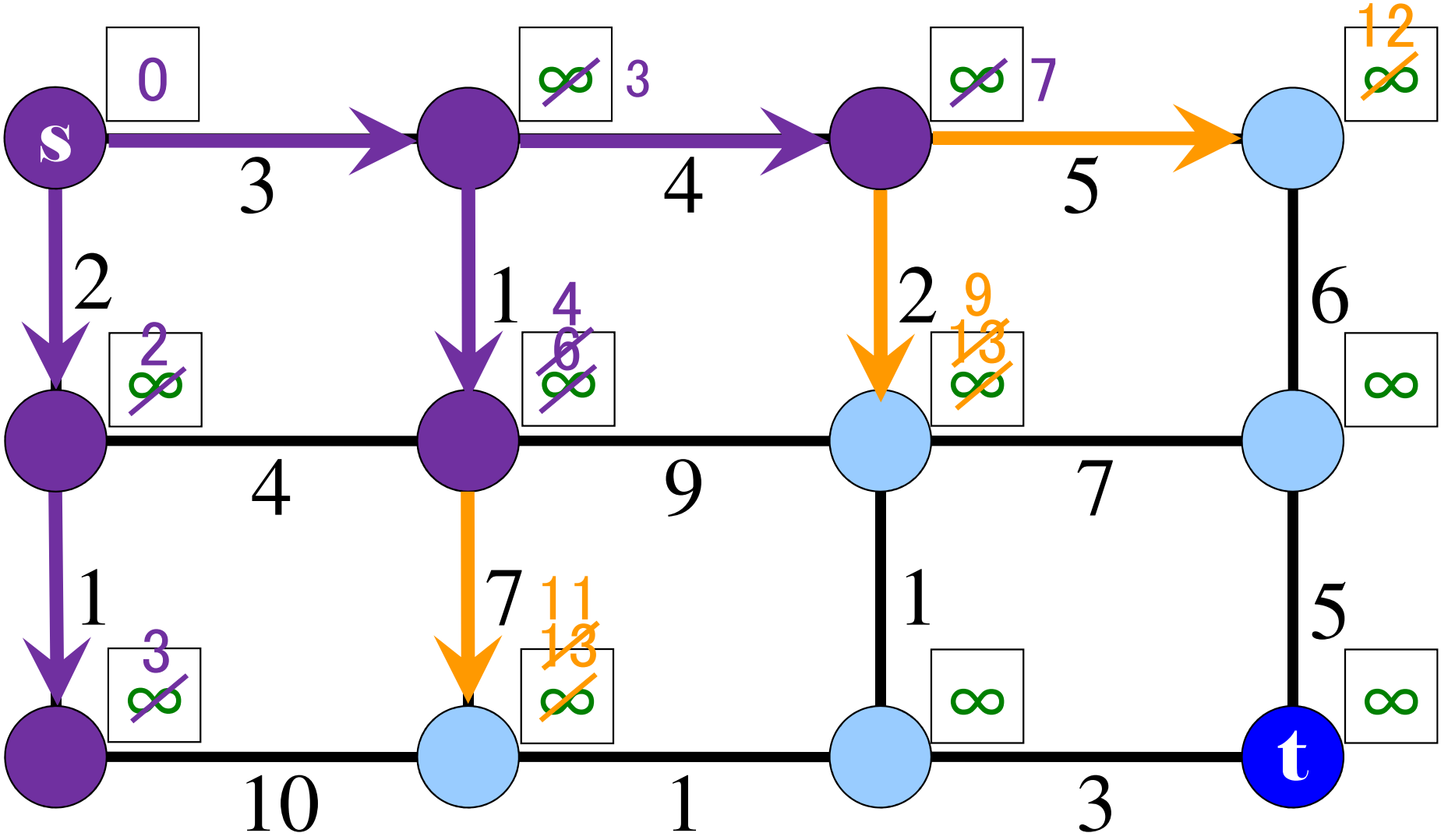


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

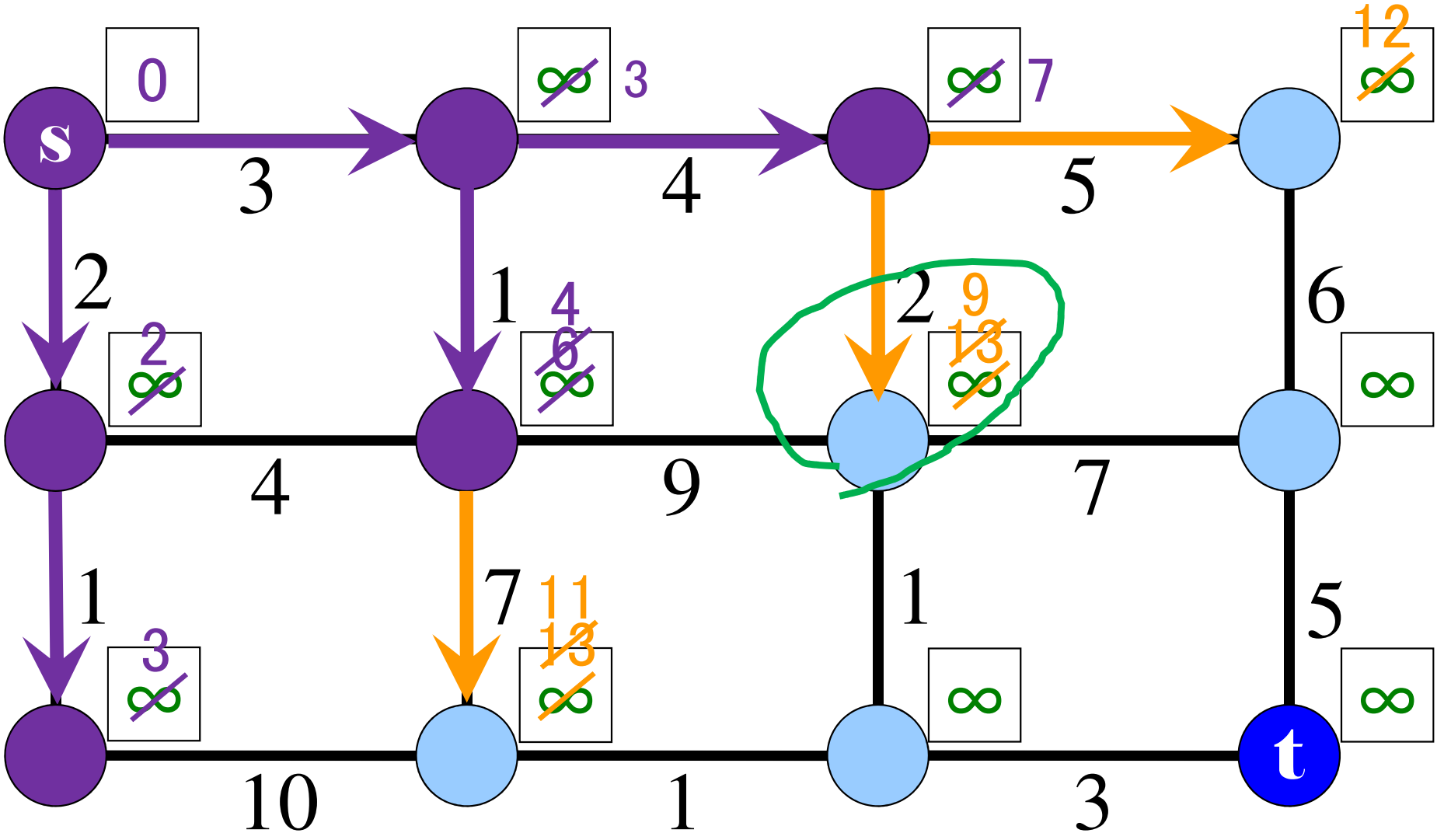
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



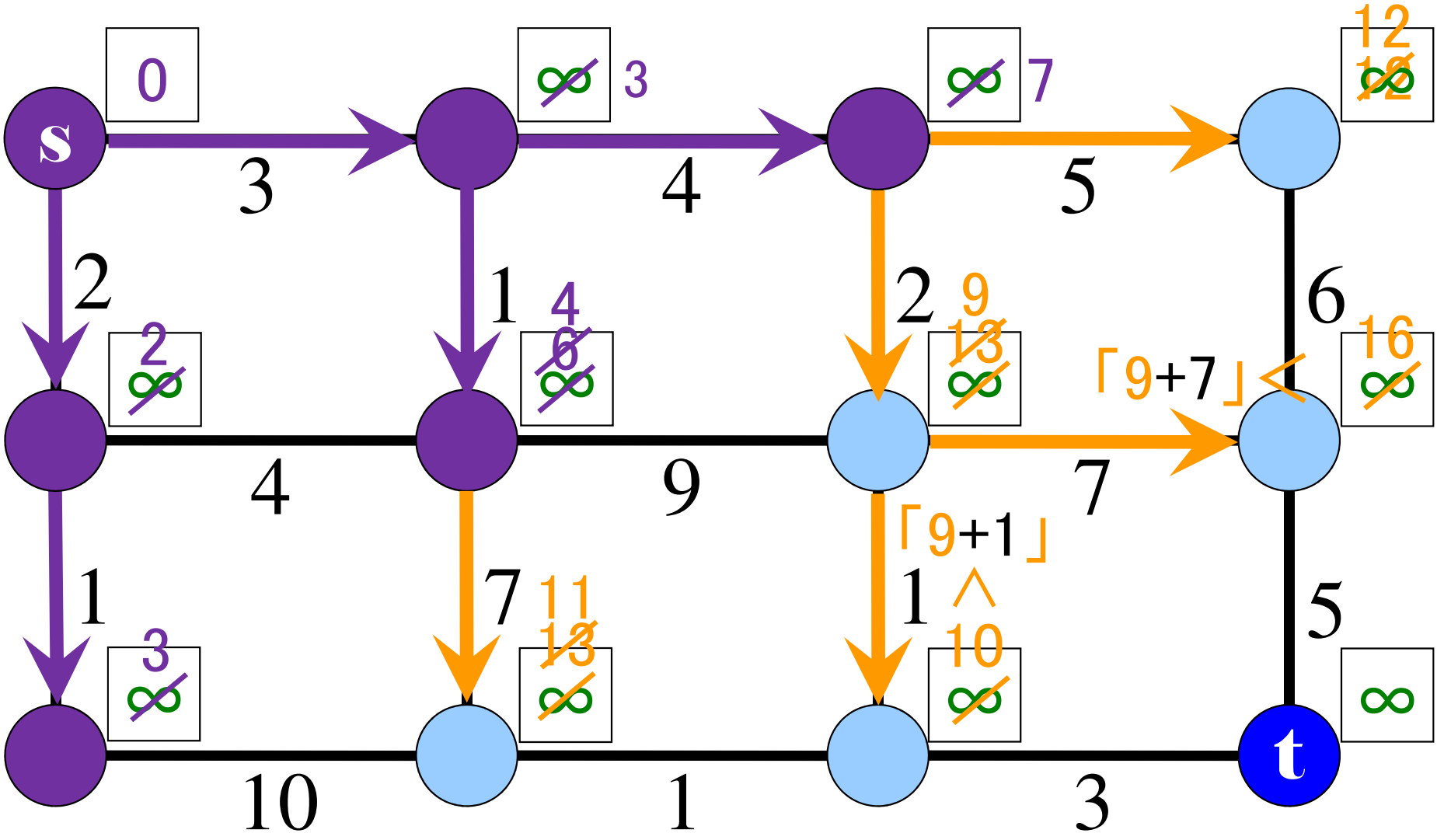
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

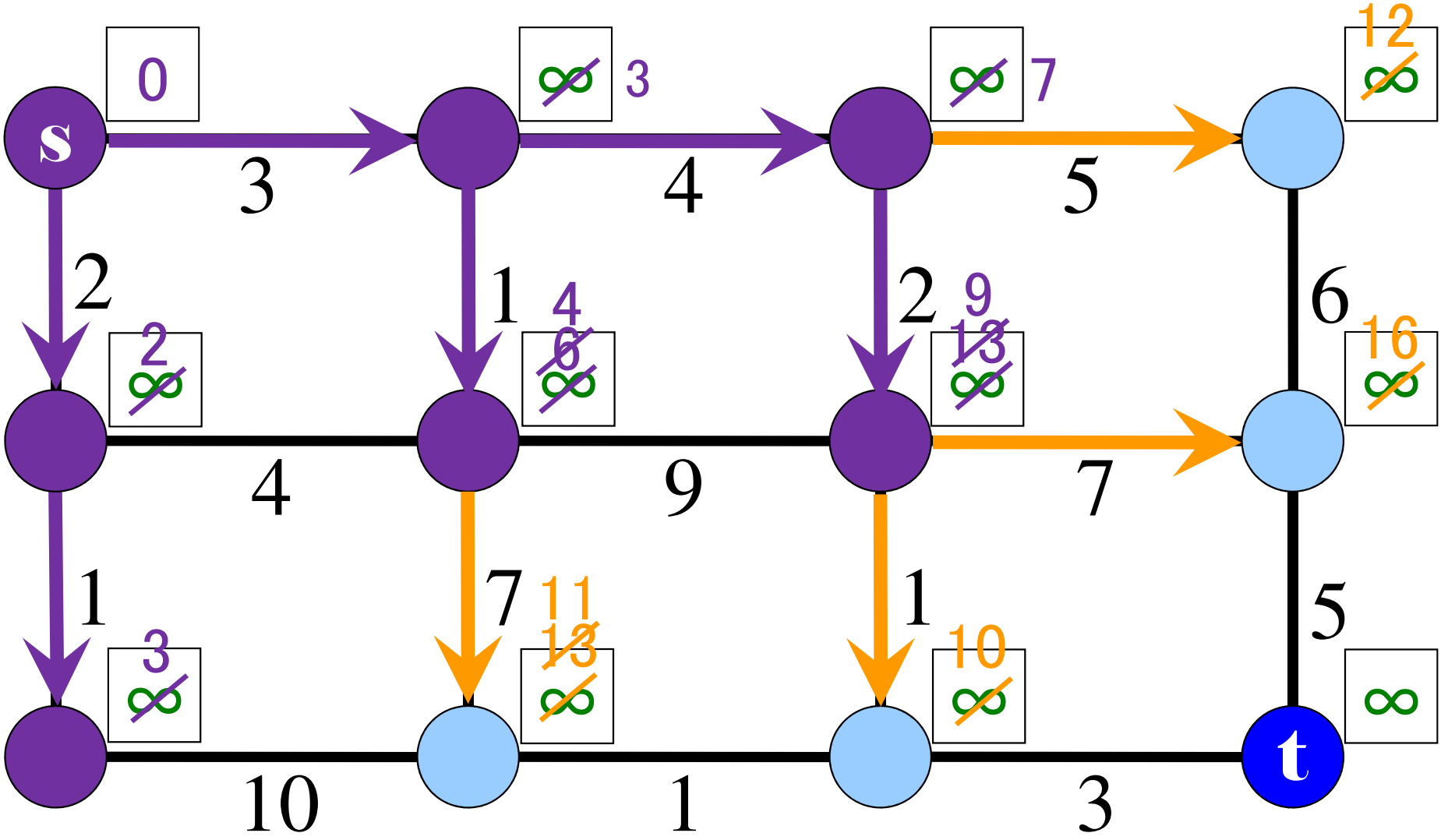


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

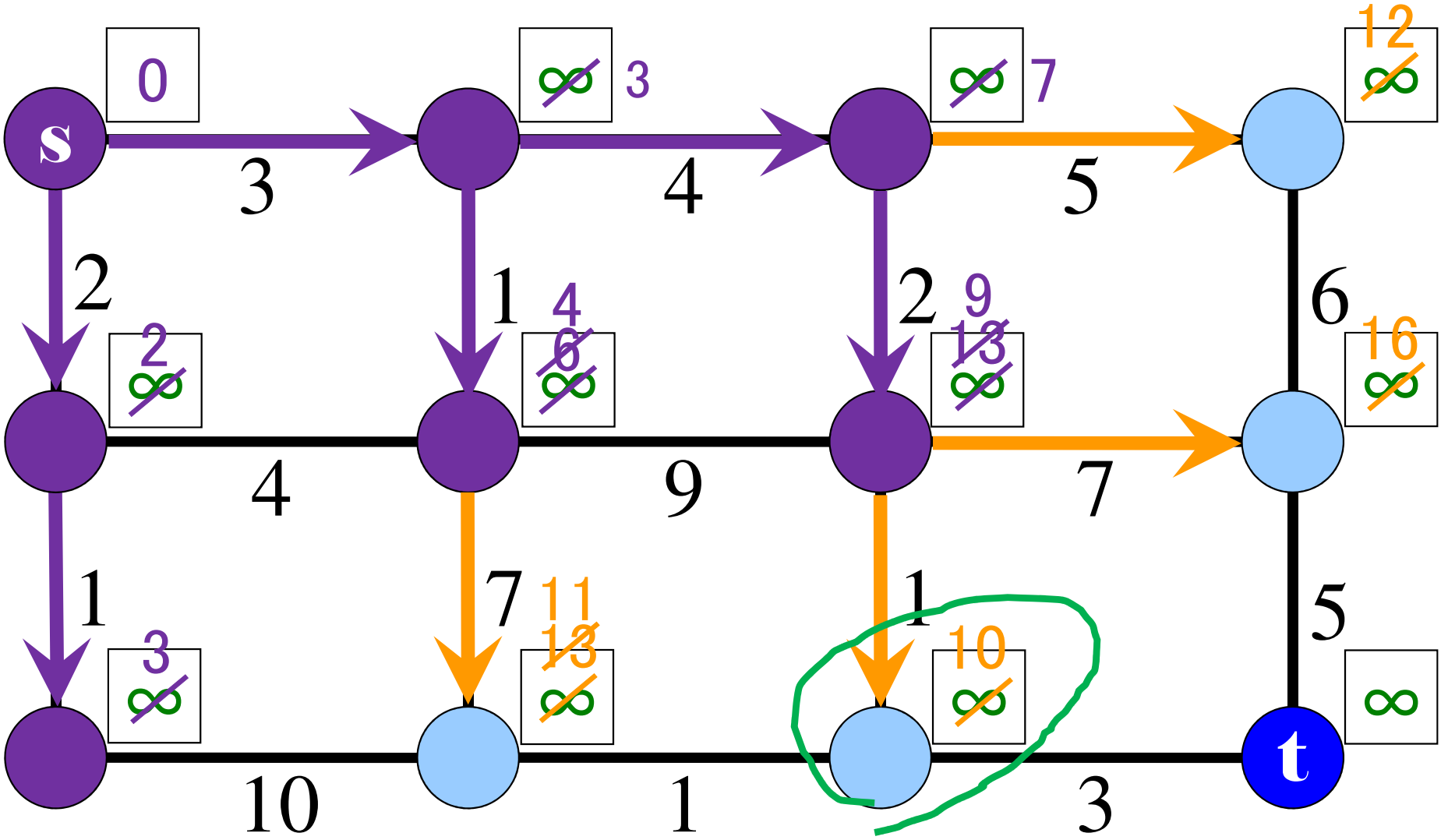
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



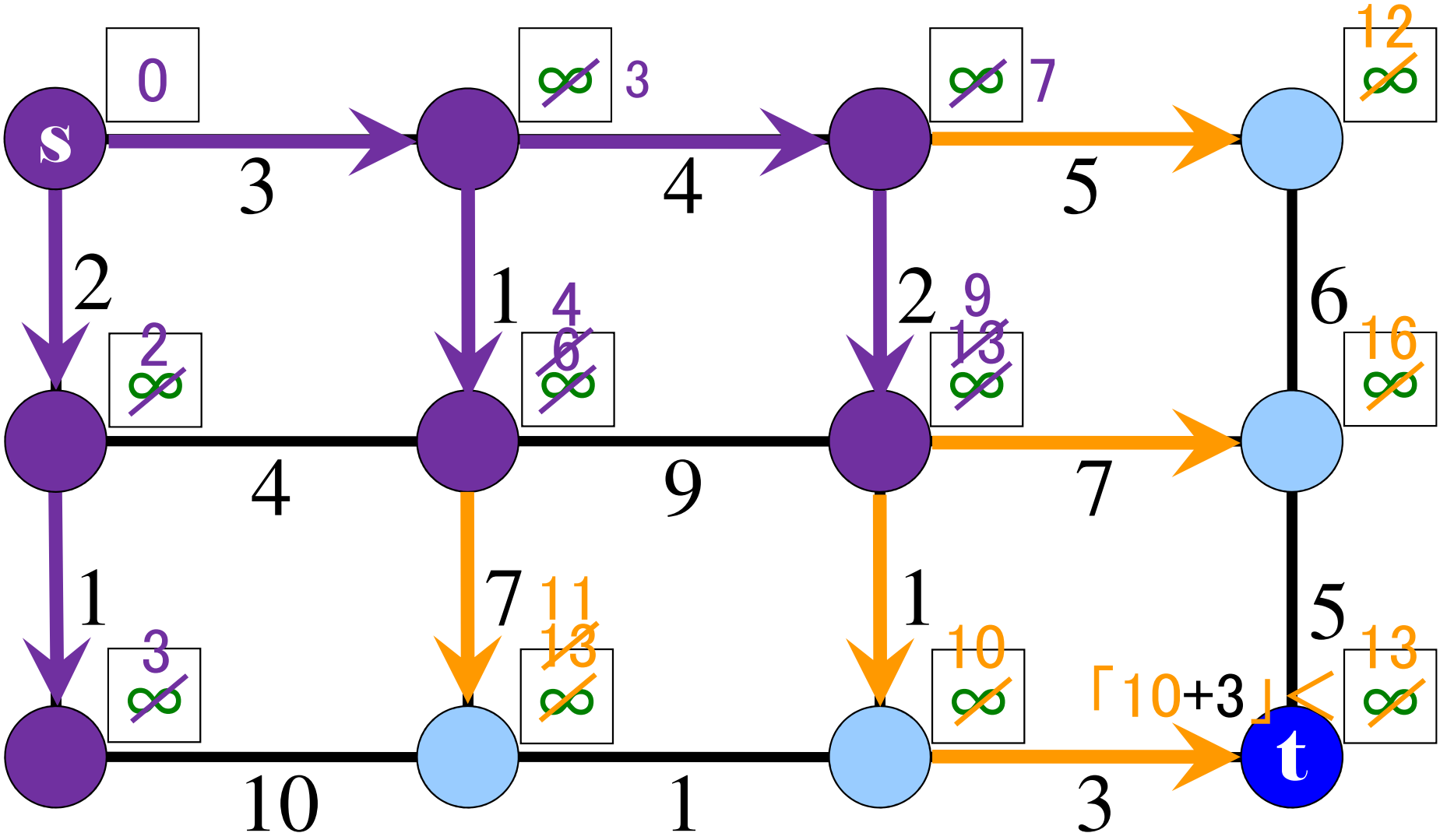
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

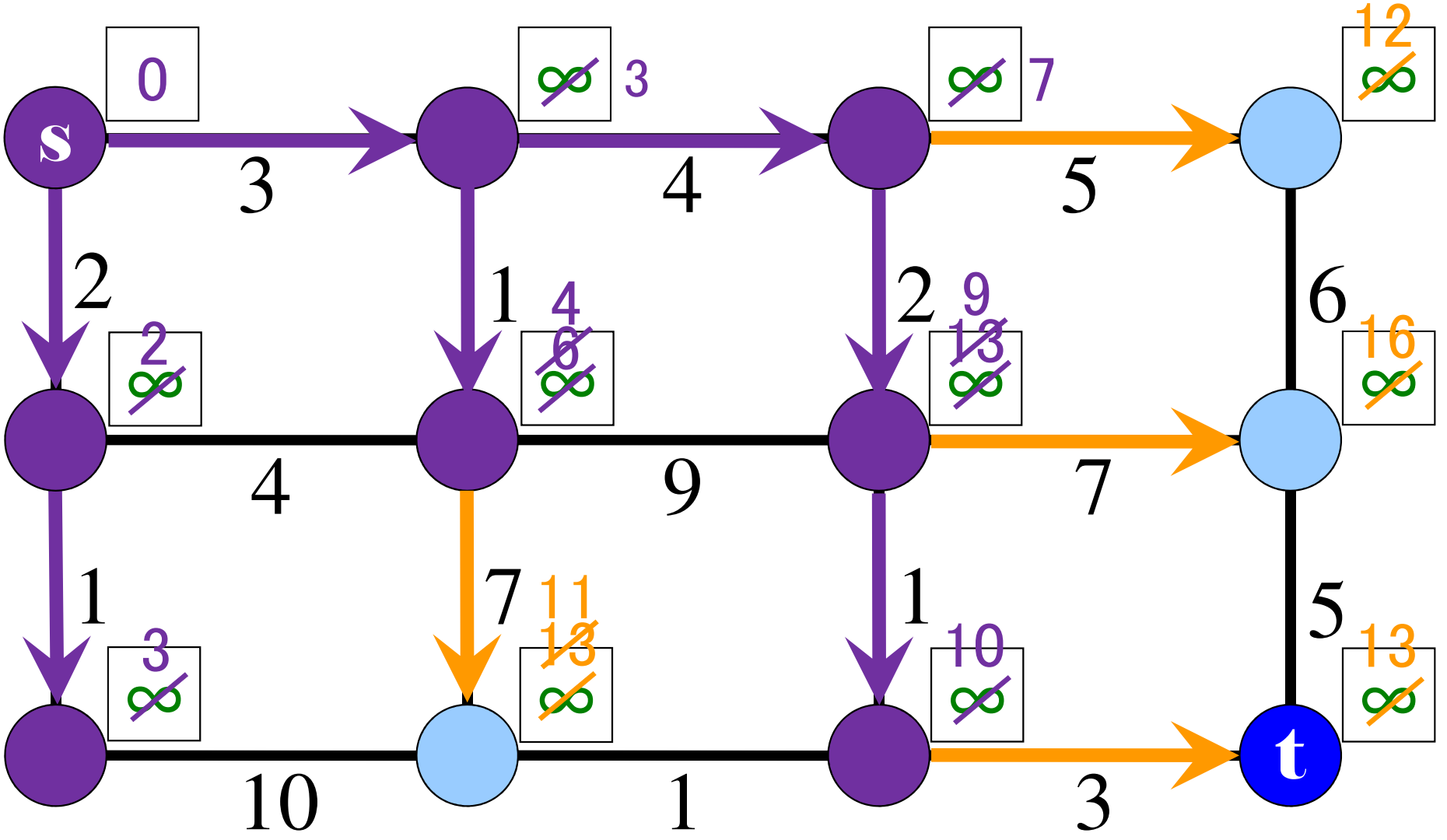


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

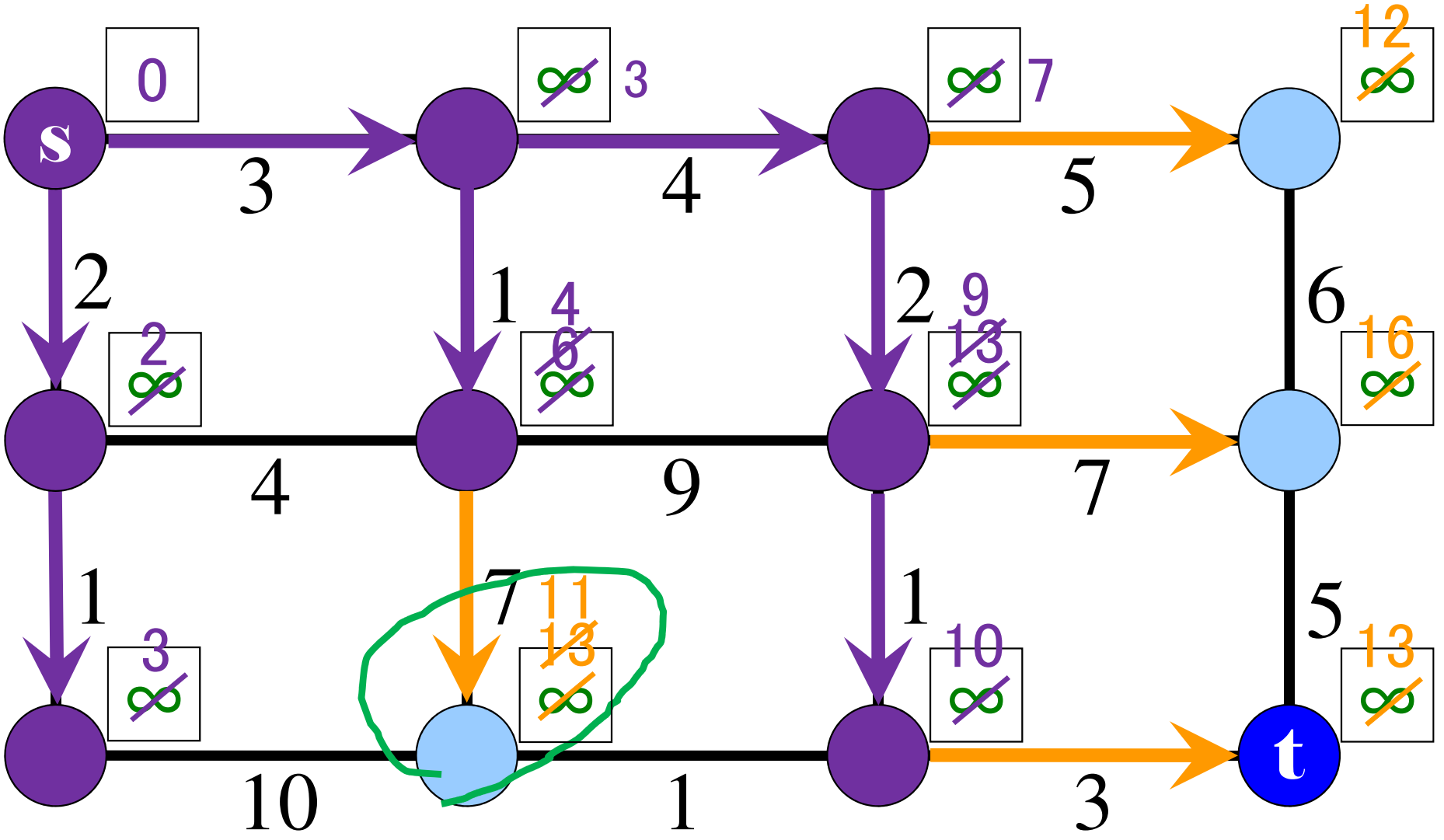
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



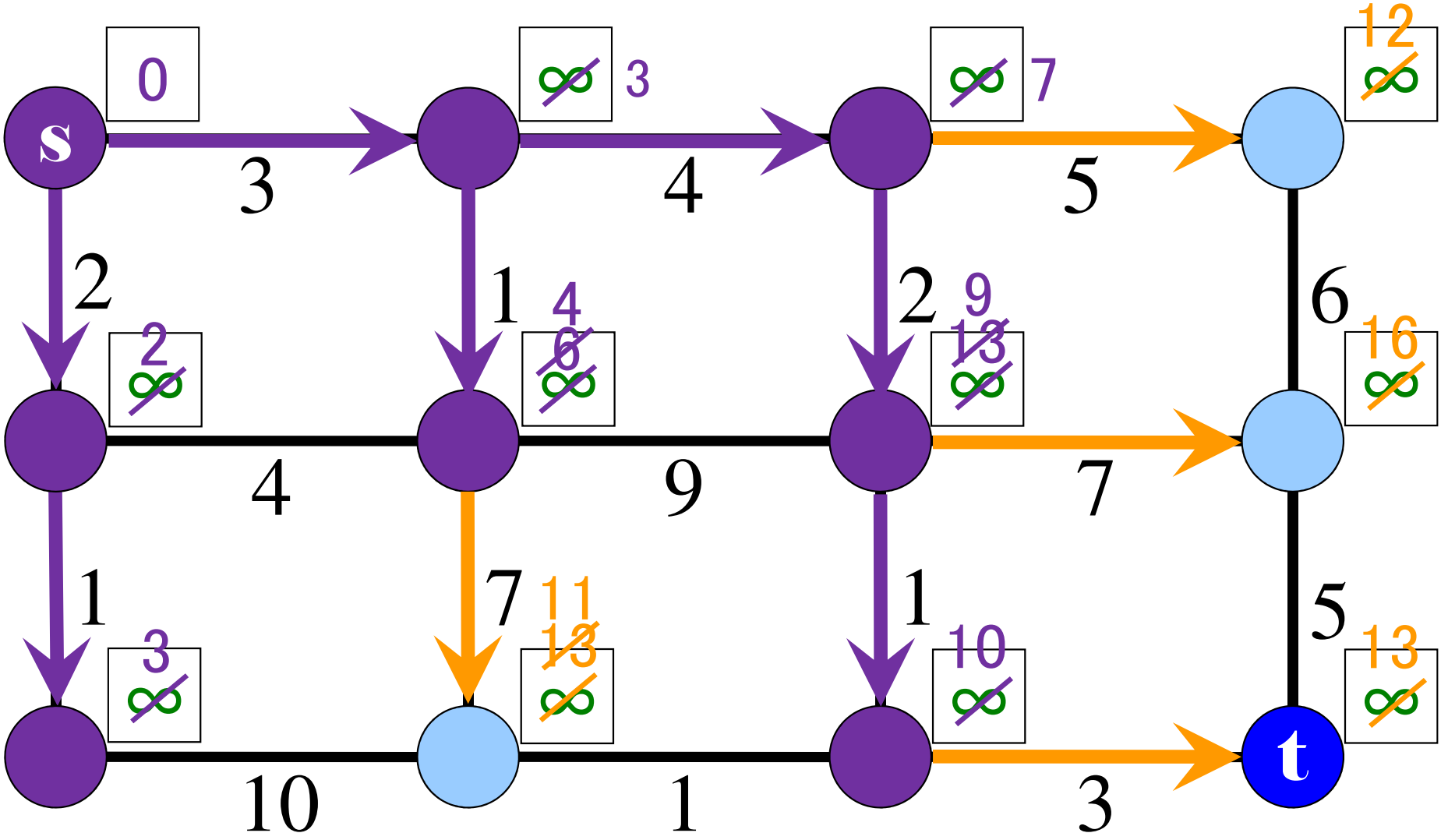
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

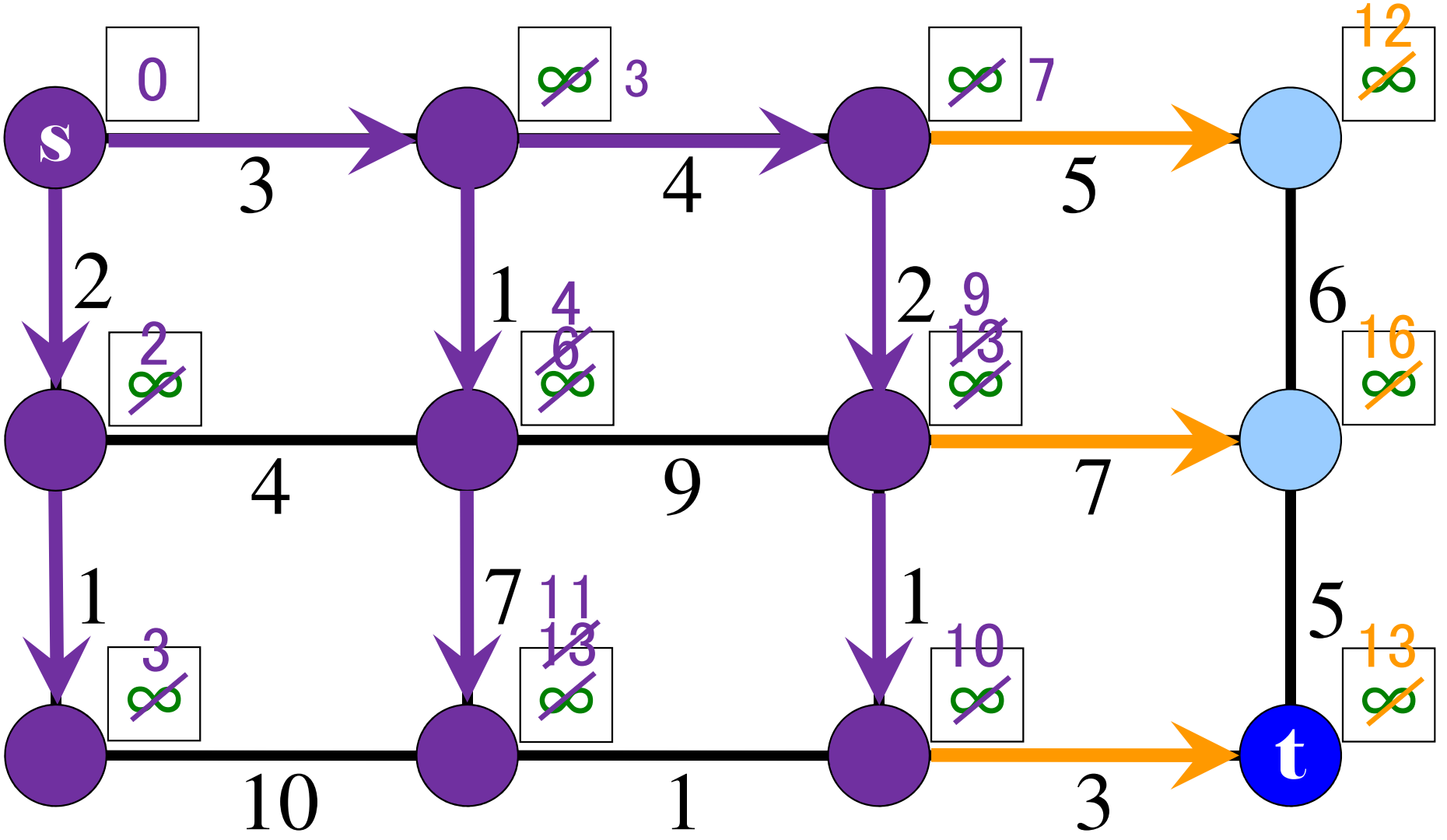


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

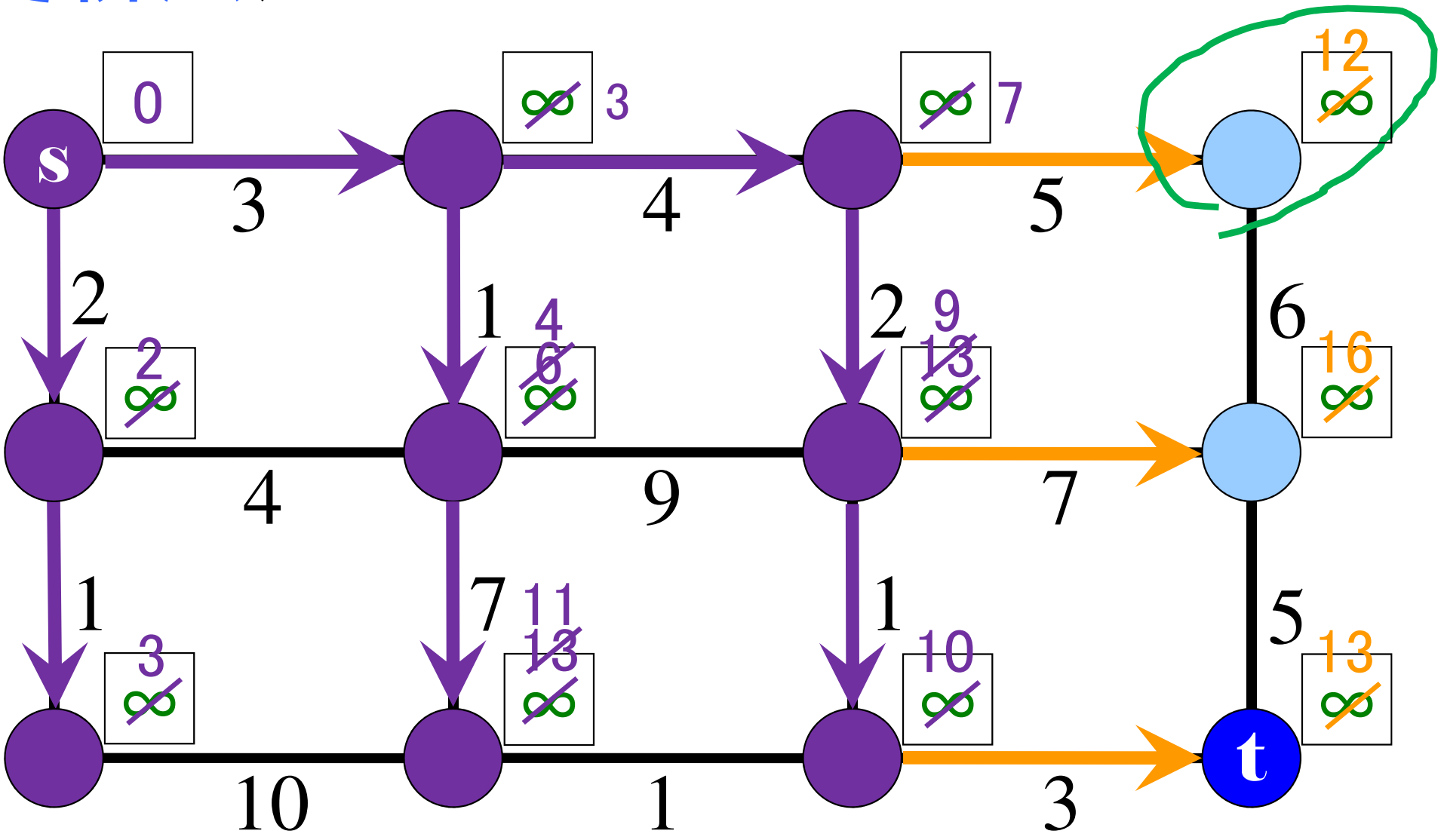
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



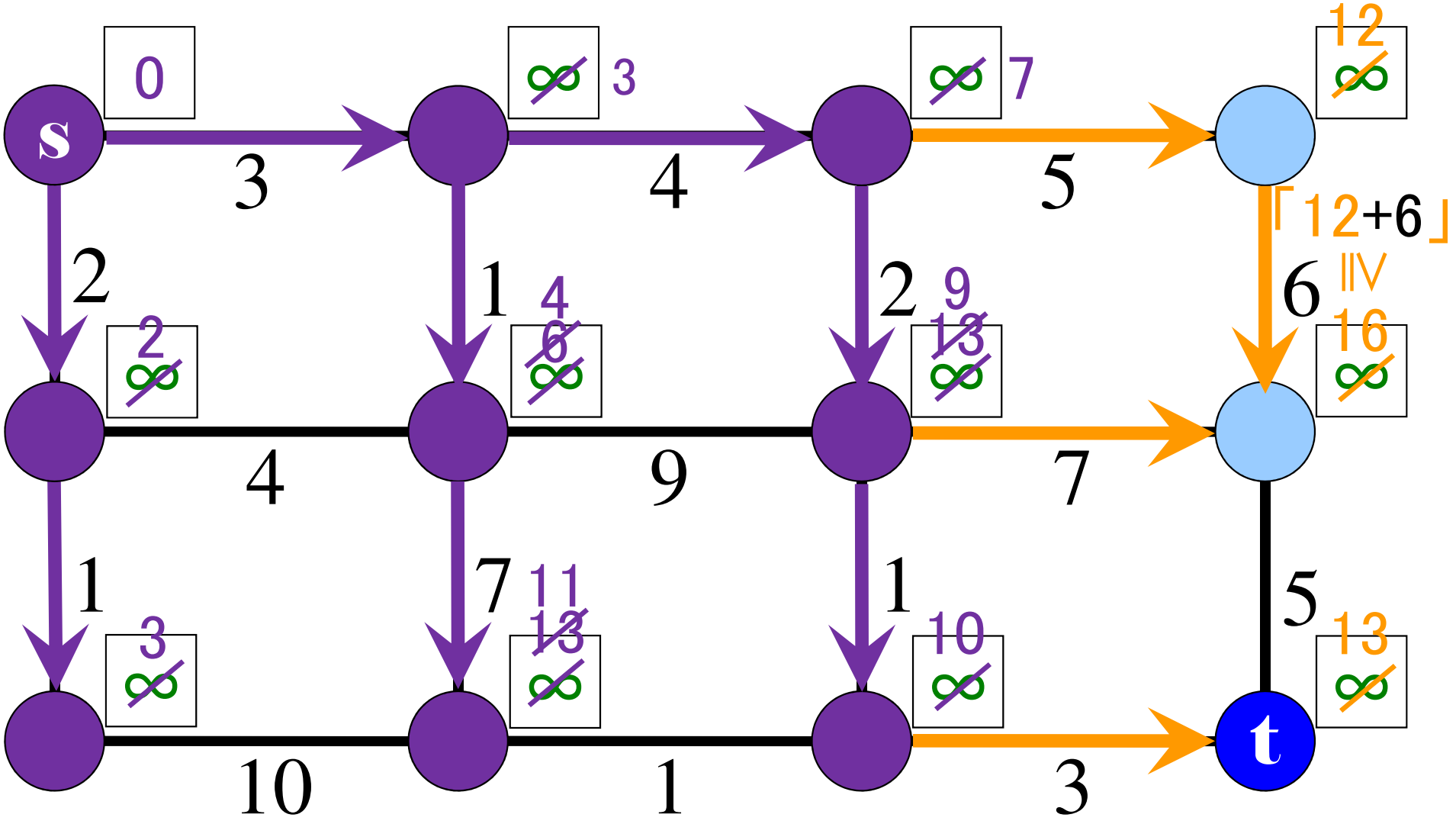
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

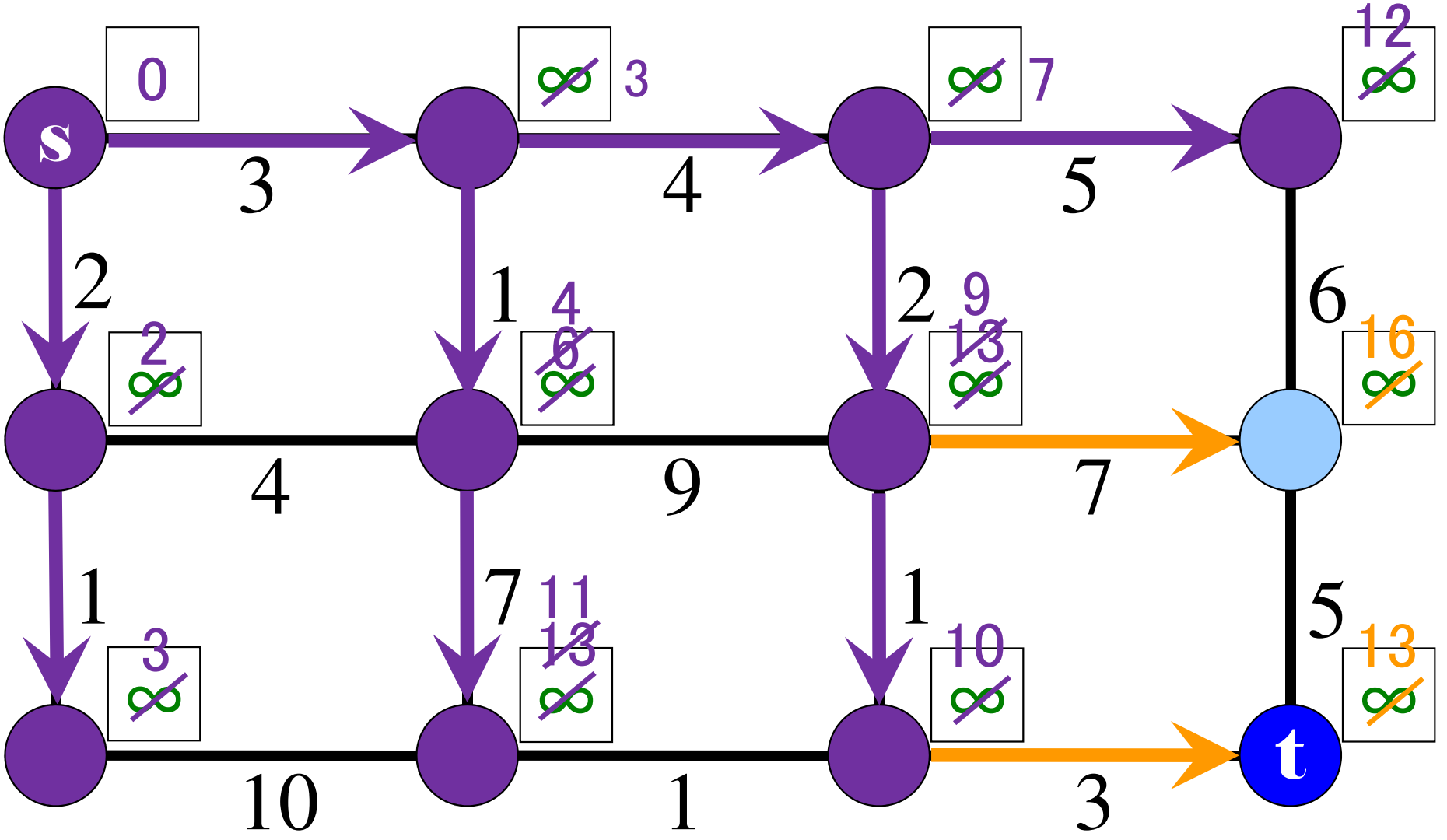


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

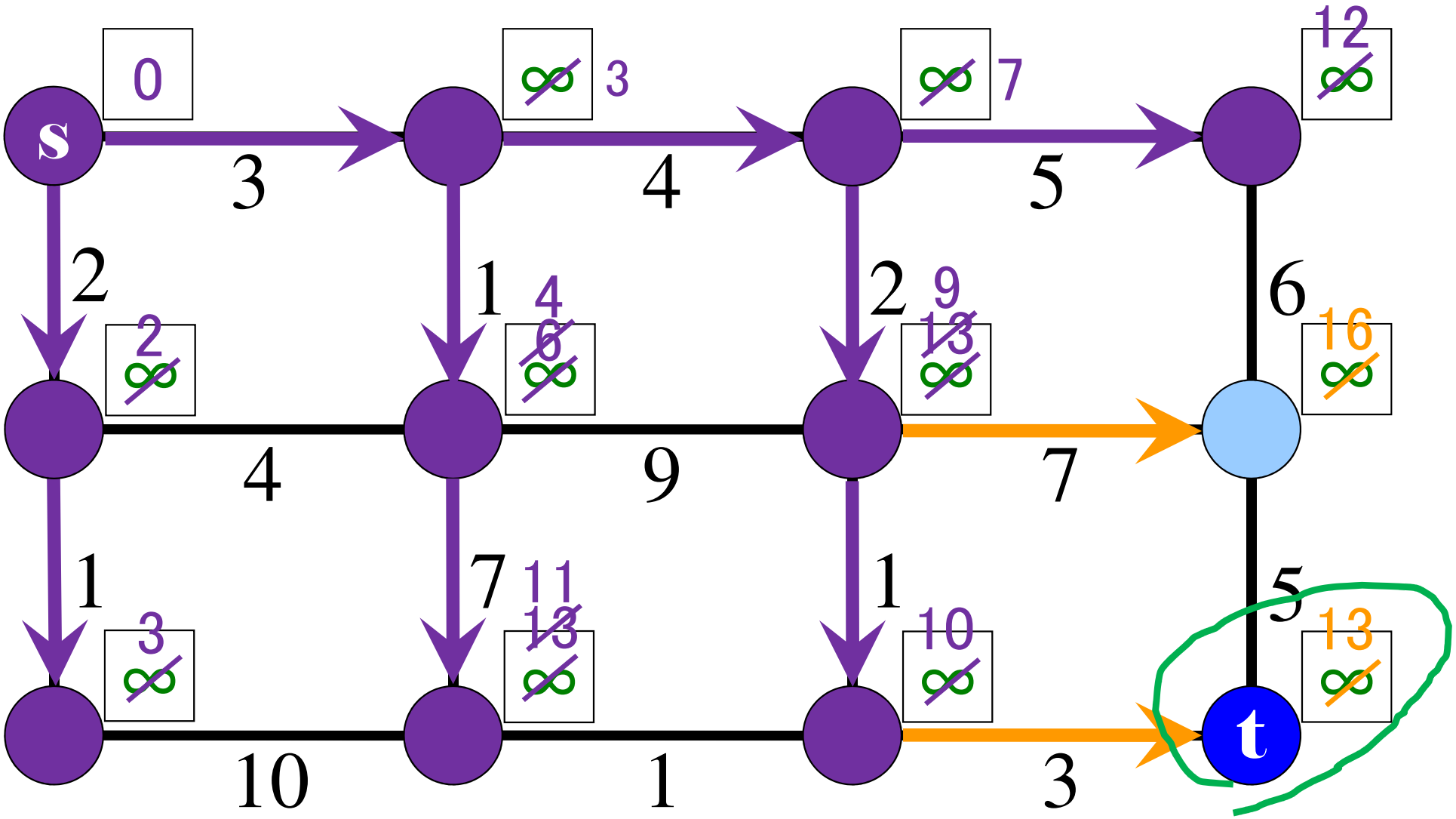
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



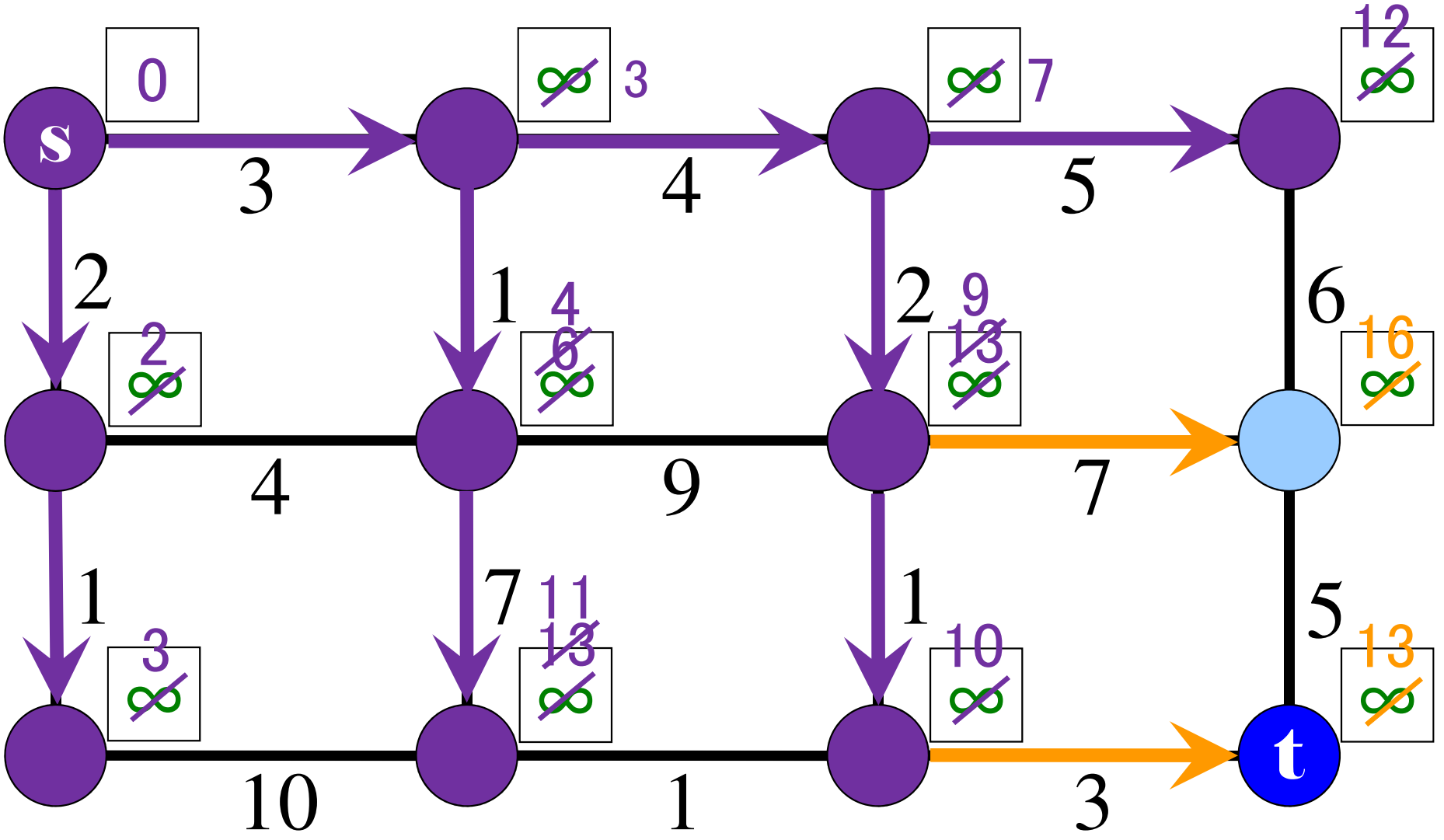
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

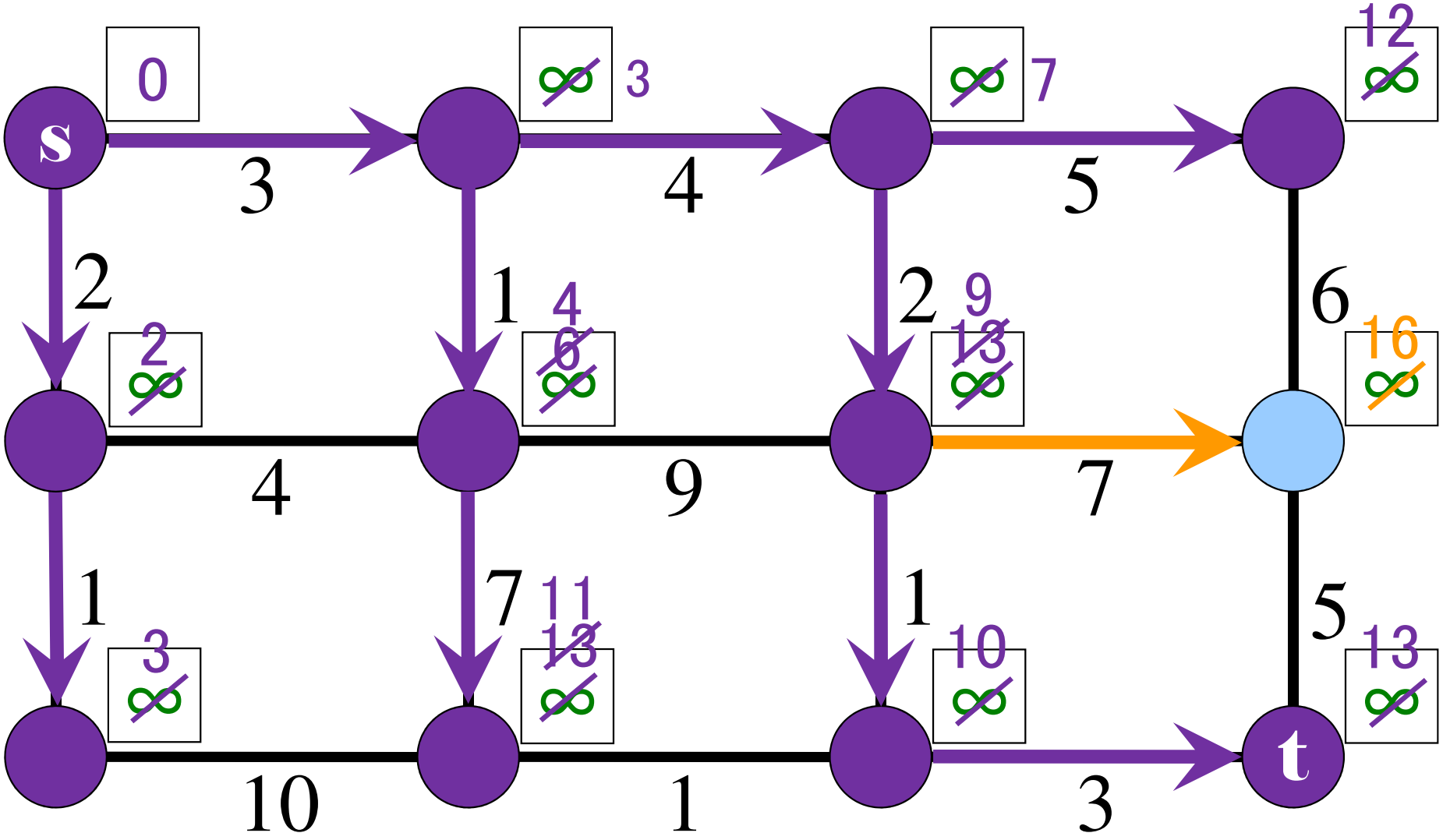


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

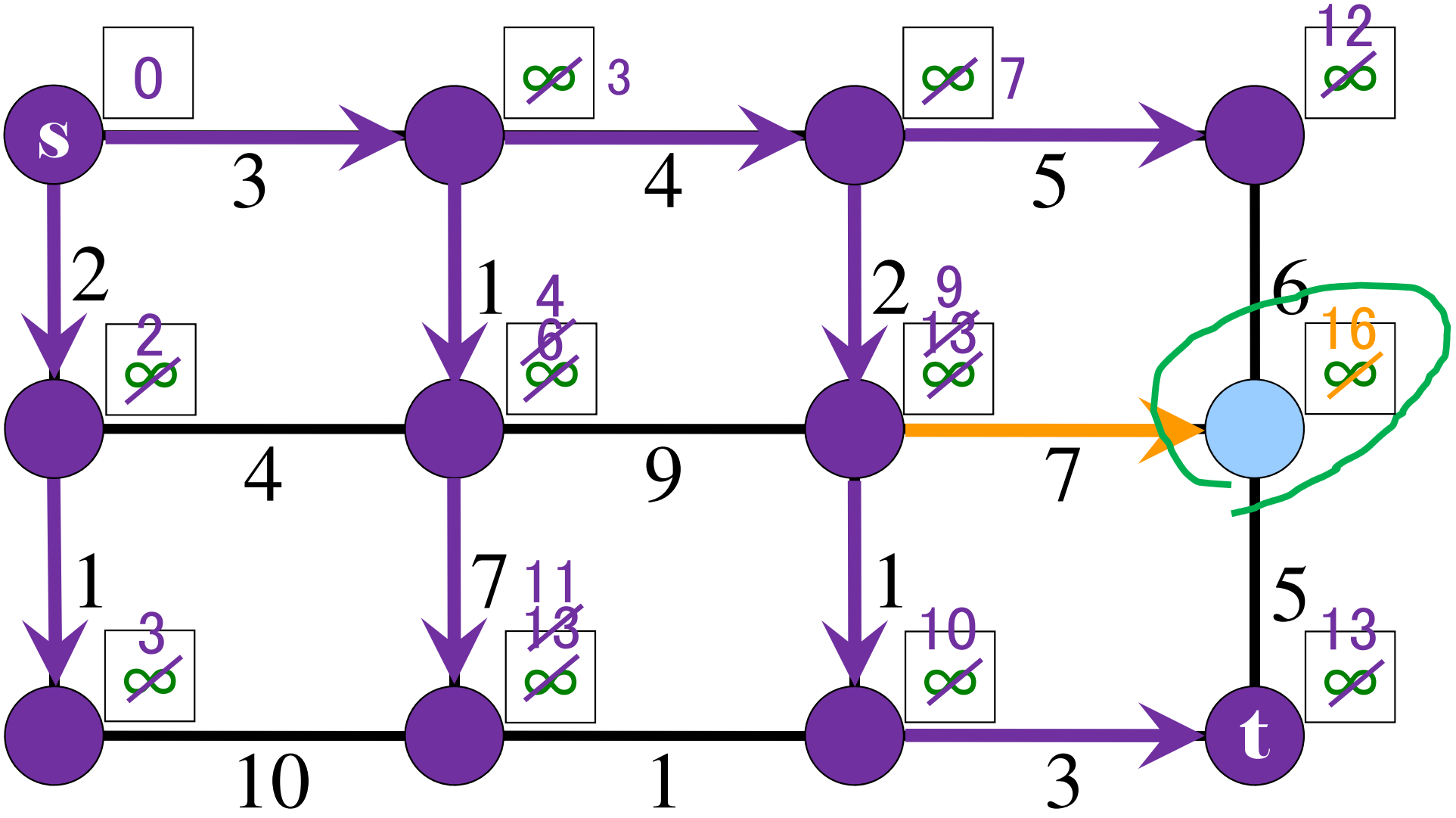
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



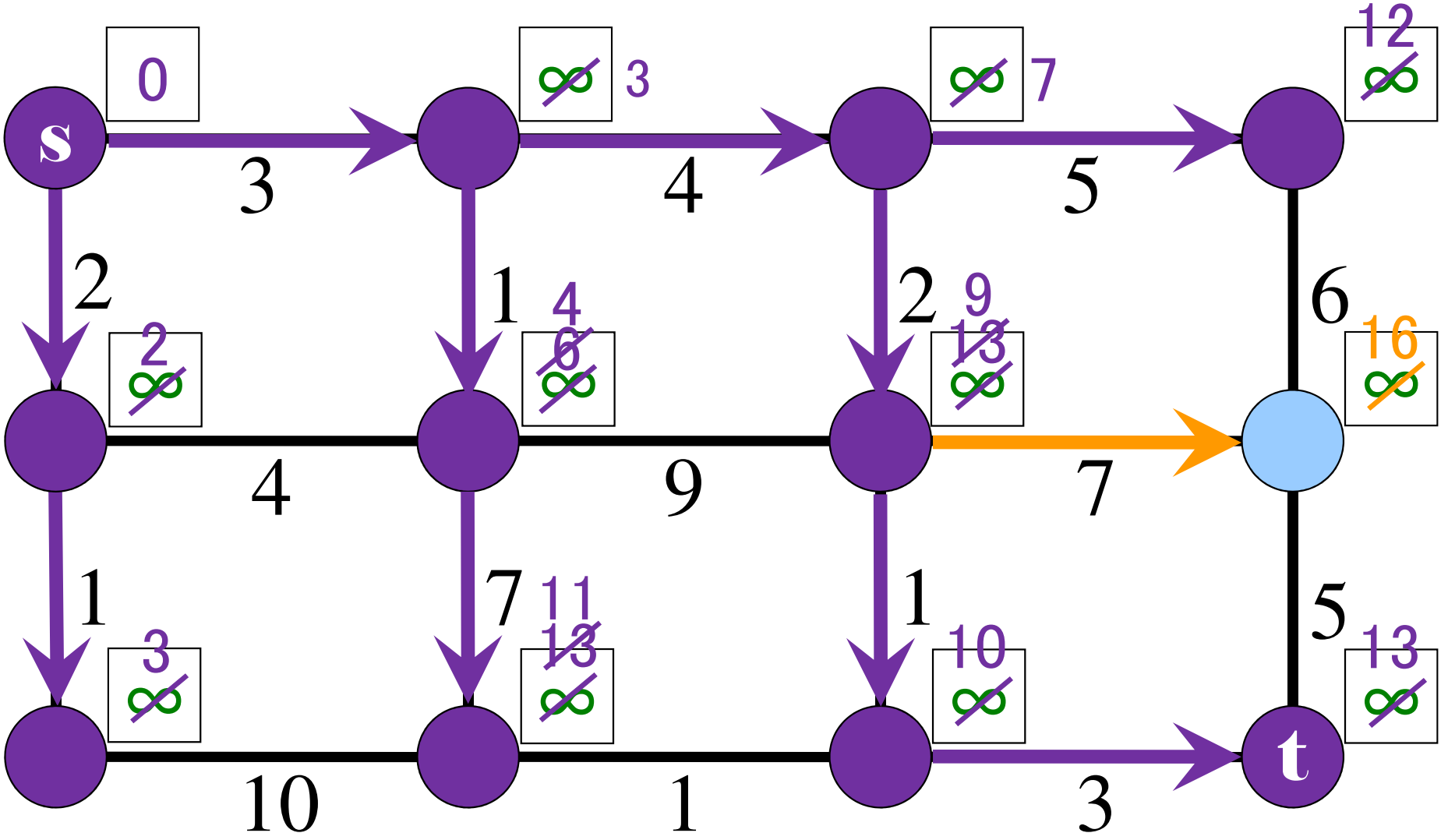
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

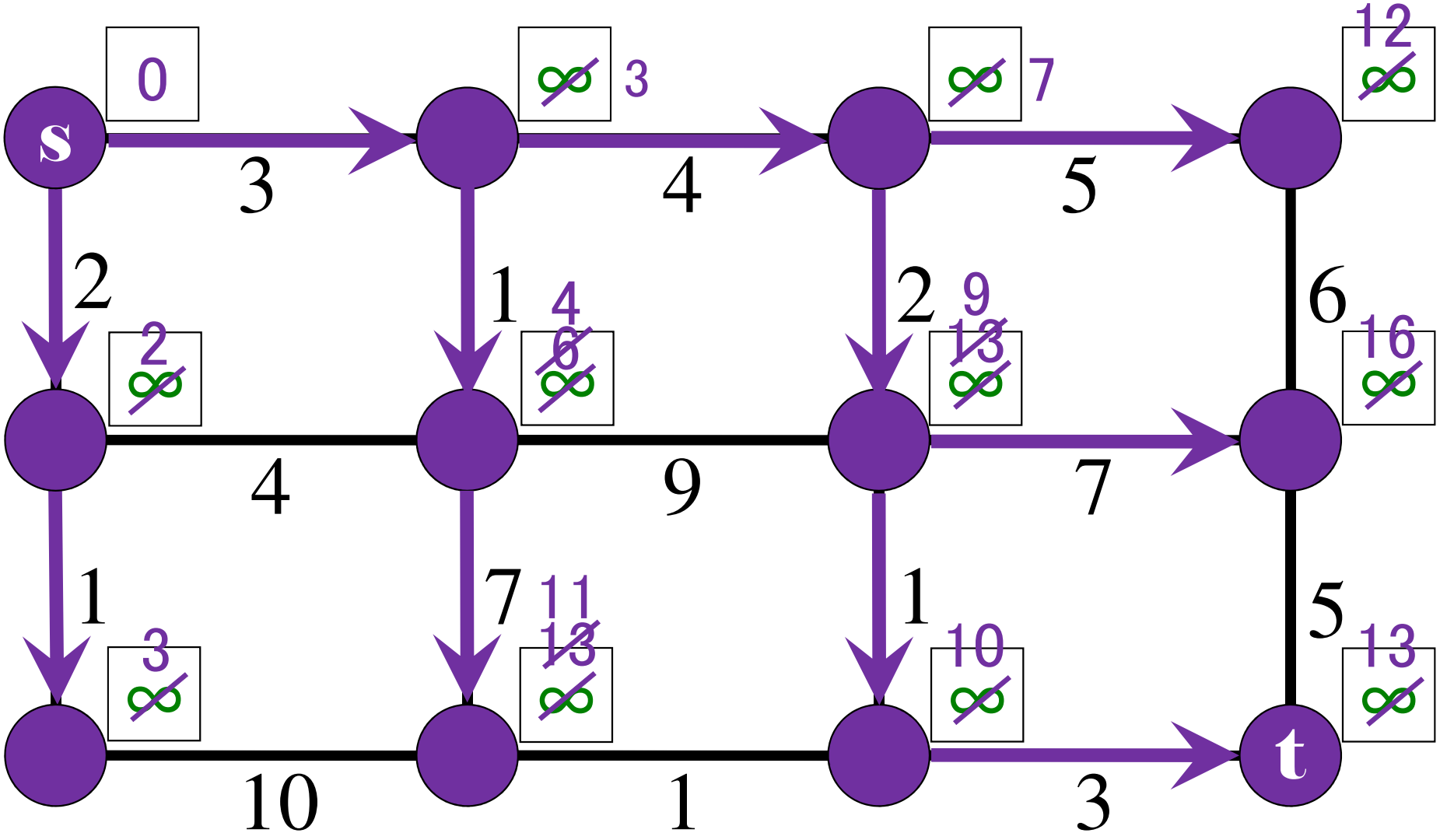


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

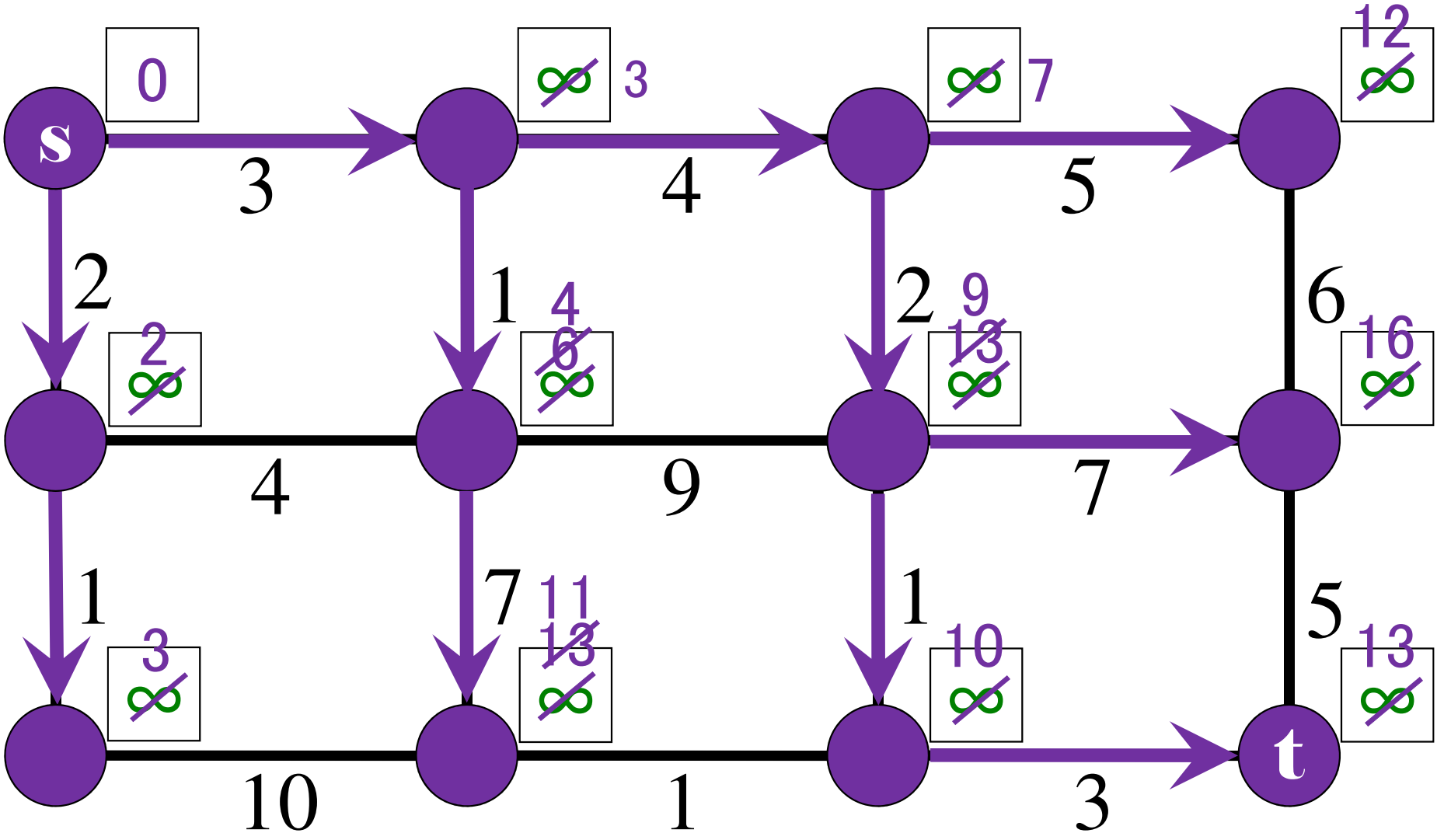
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



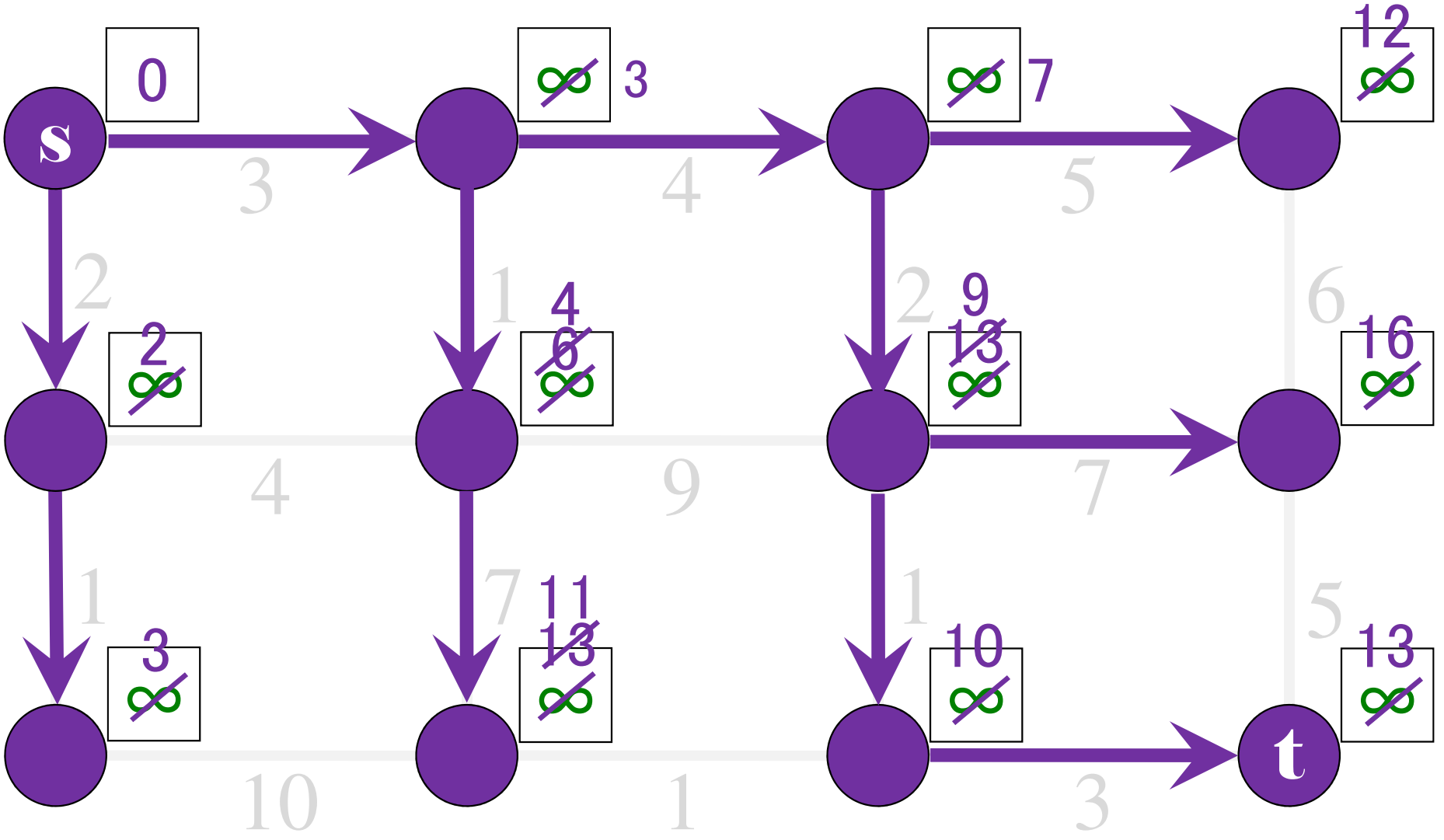
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



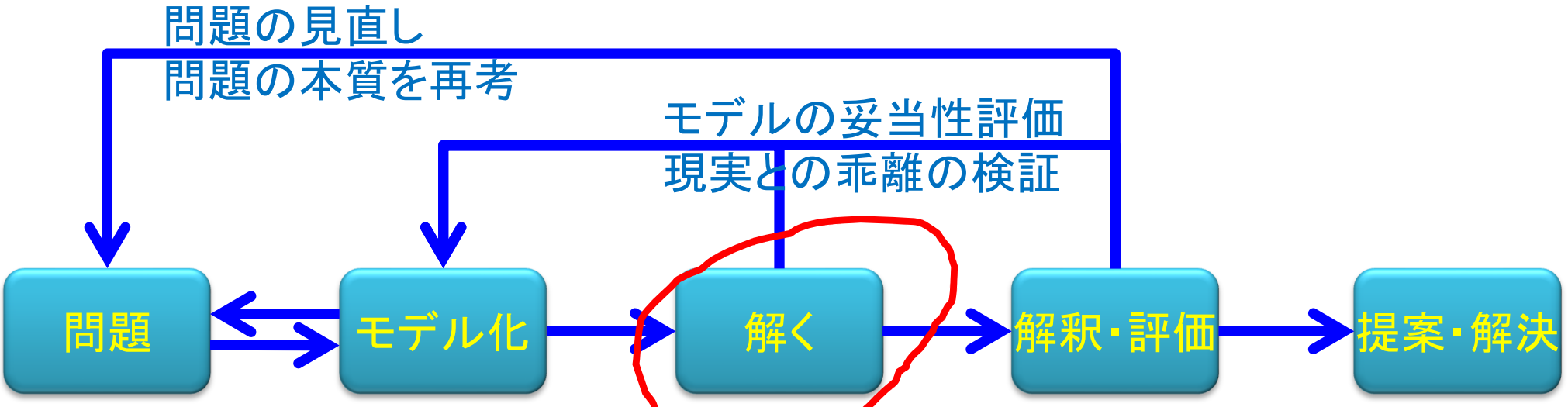
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

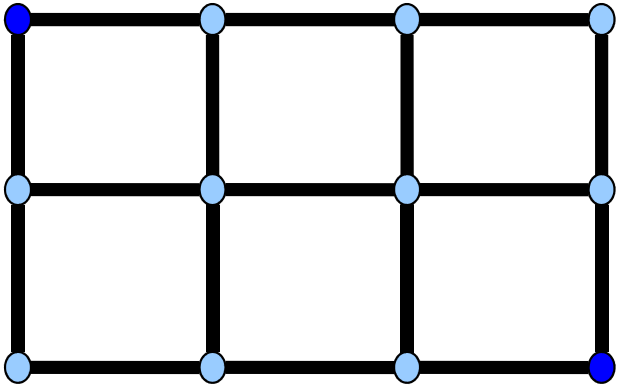
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

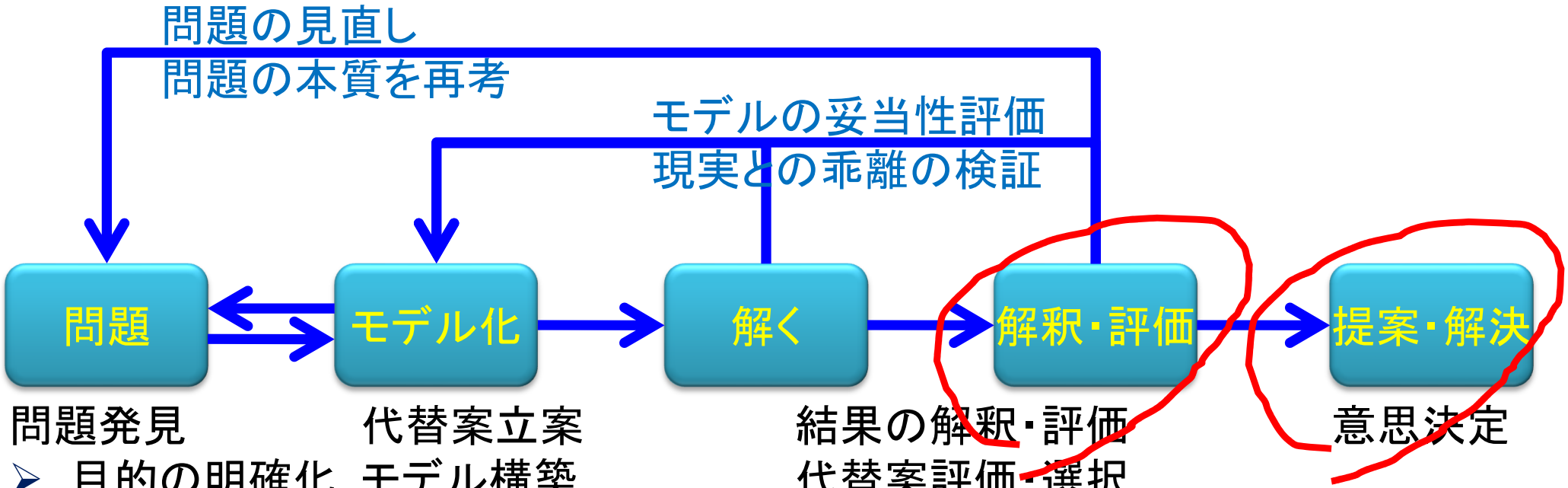
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

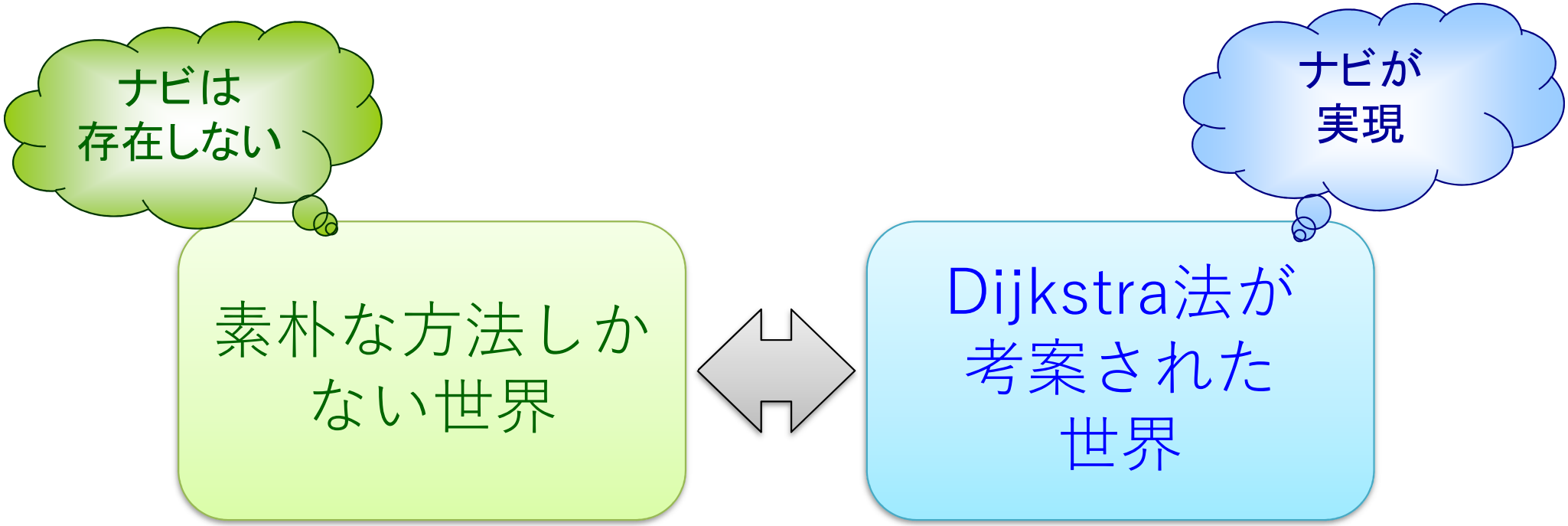
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

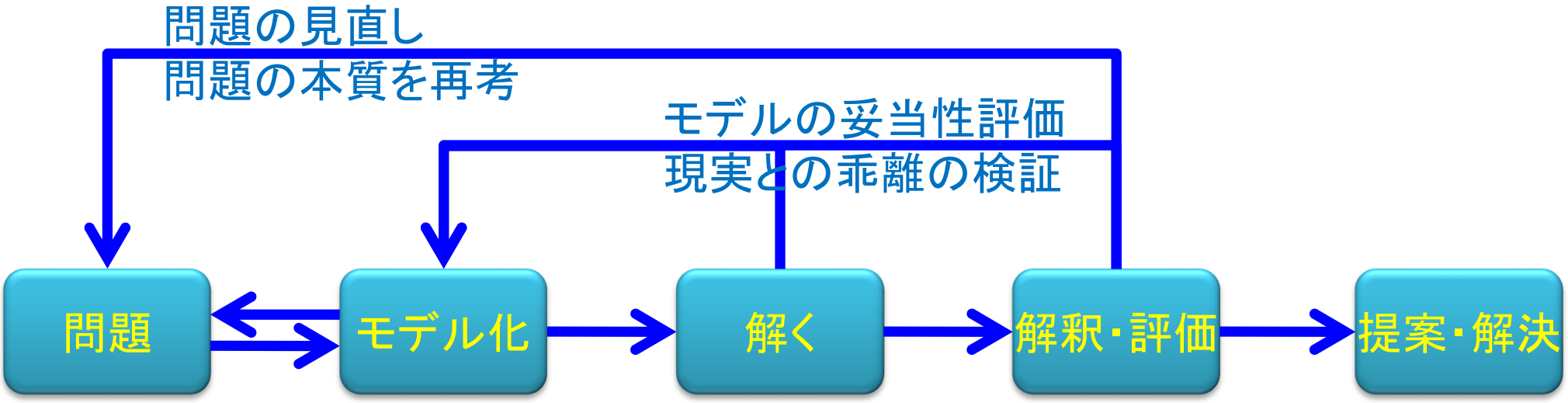
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



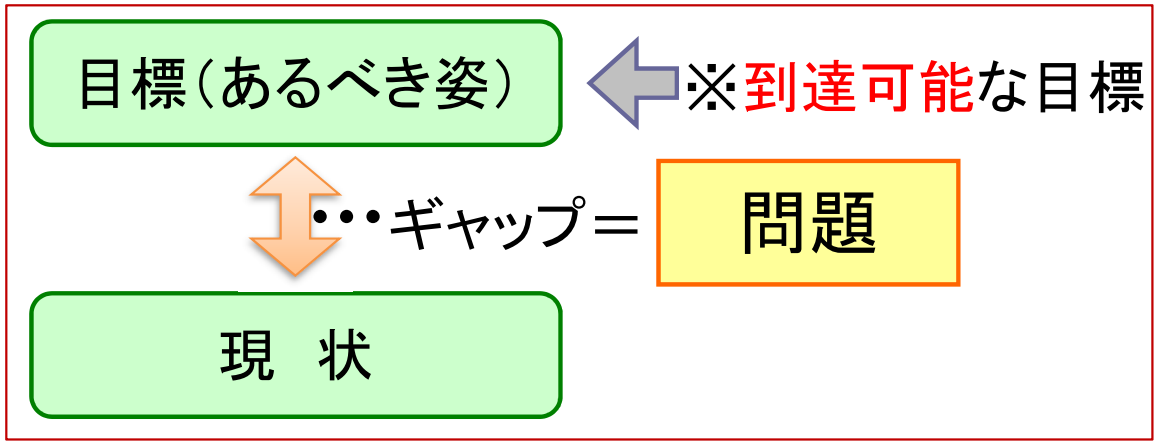
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

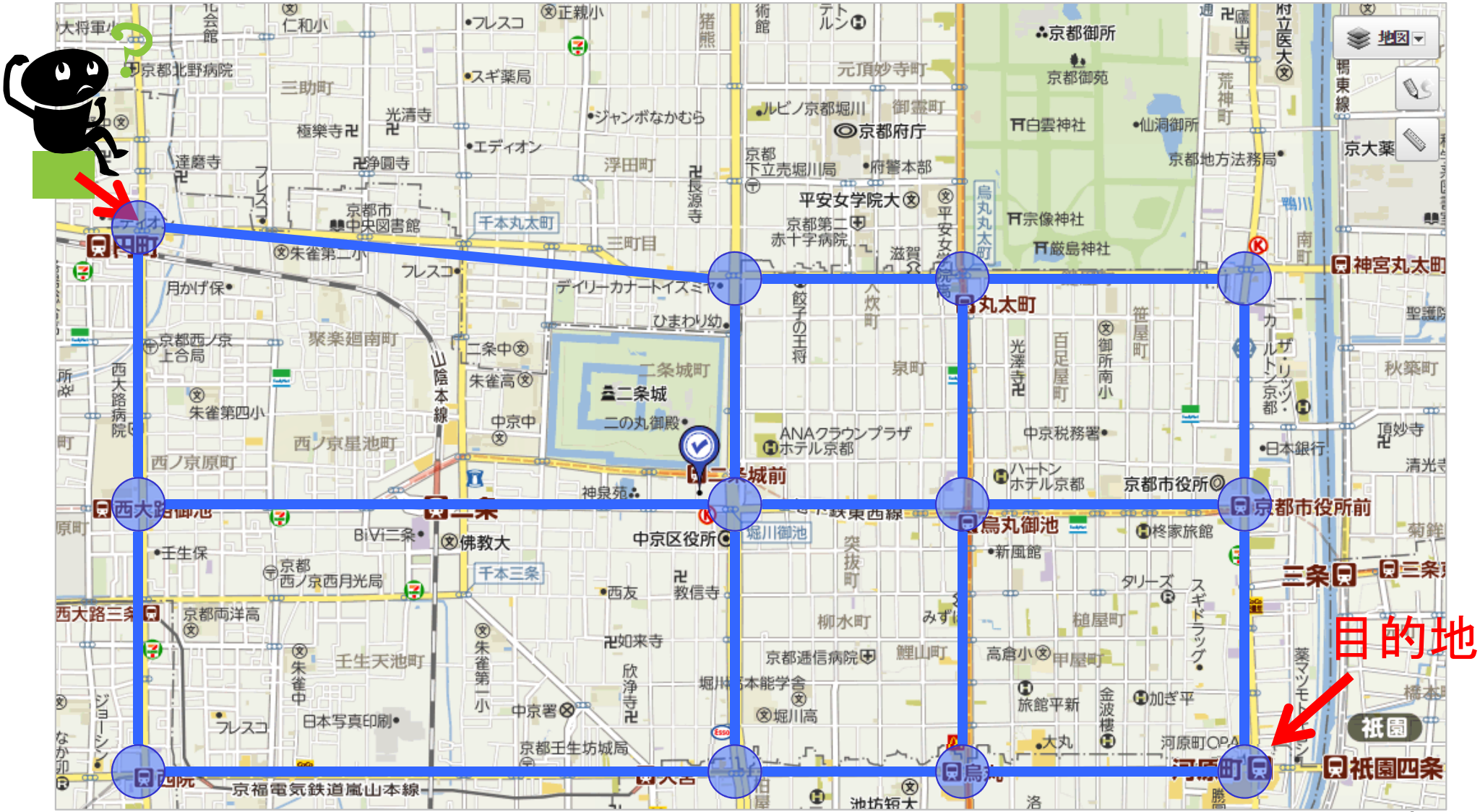
意思決定

問題の定義



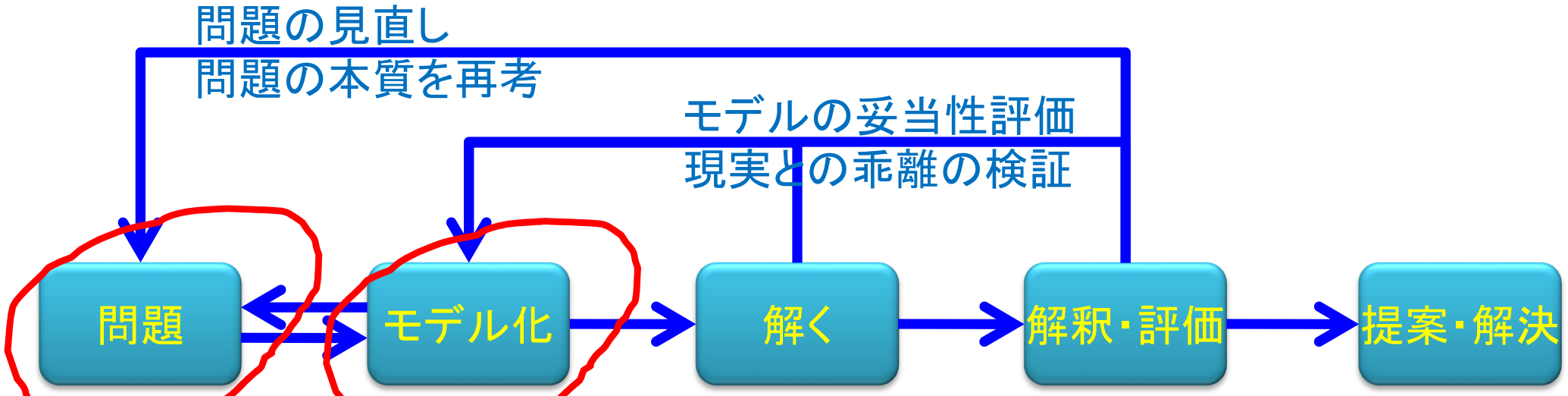
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

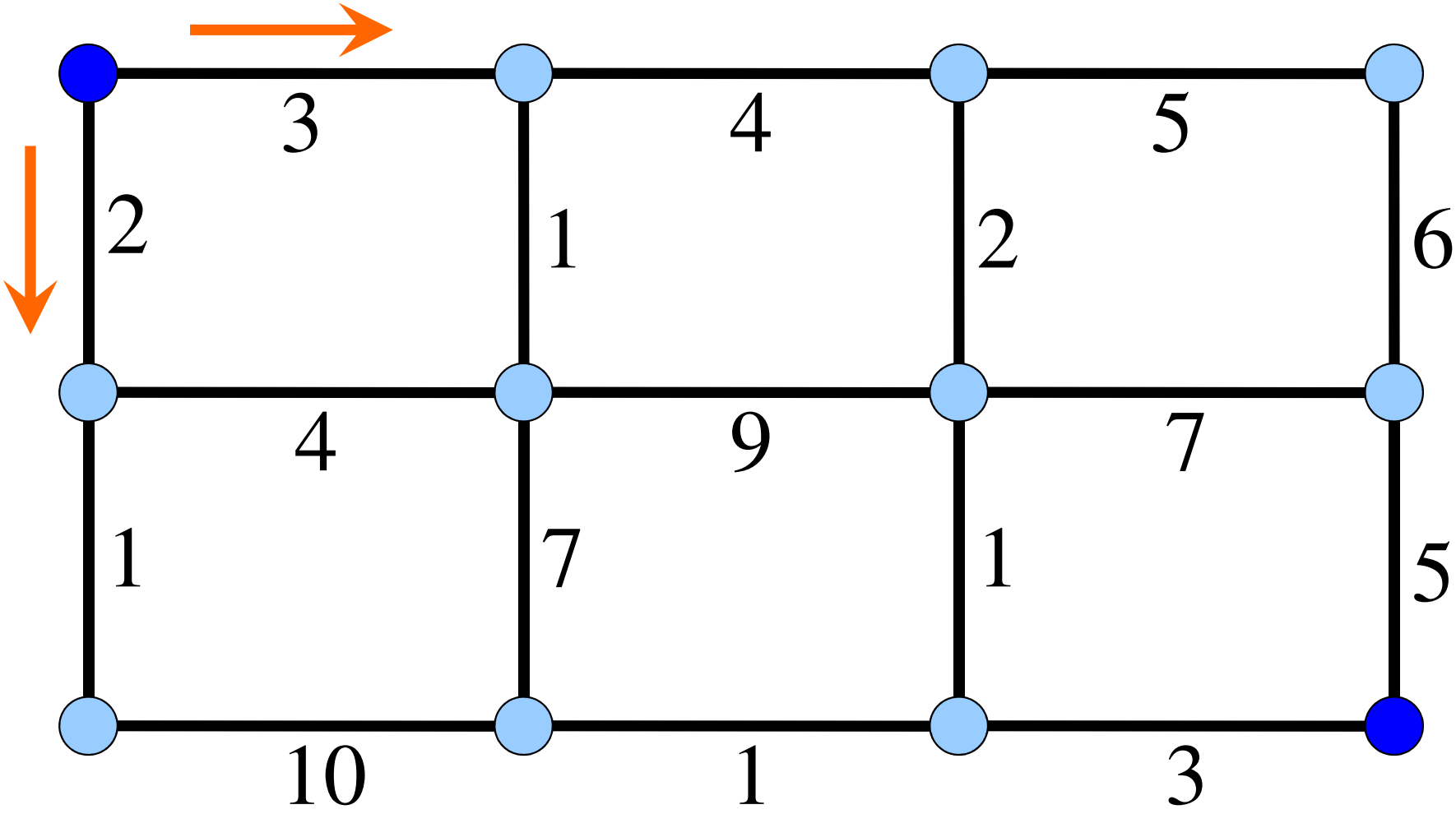
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

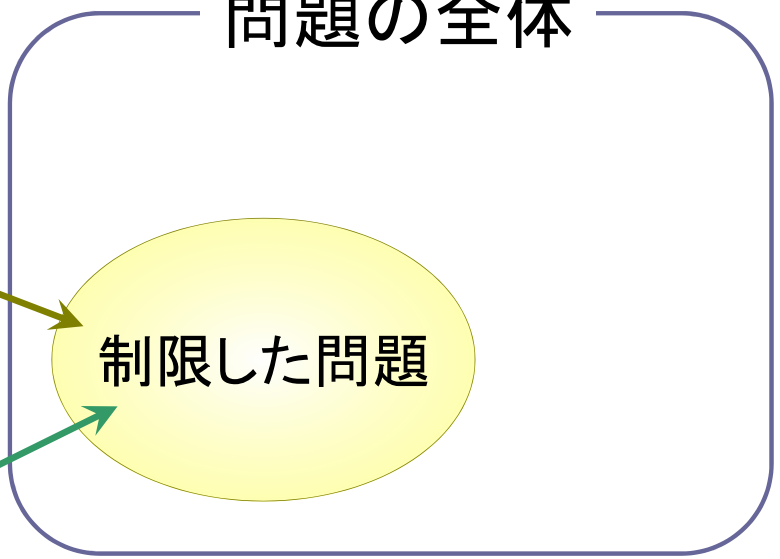


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

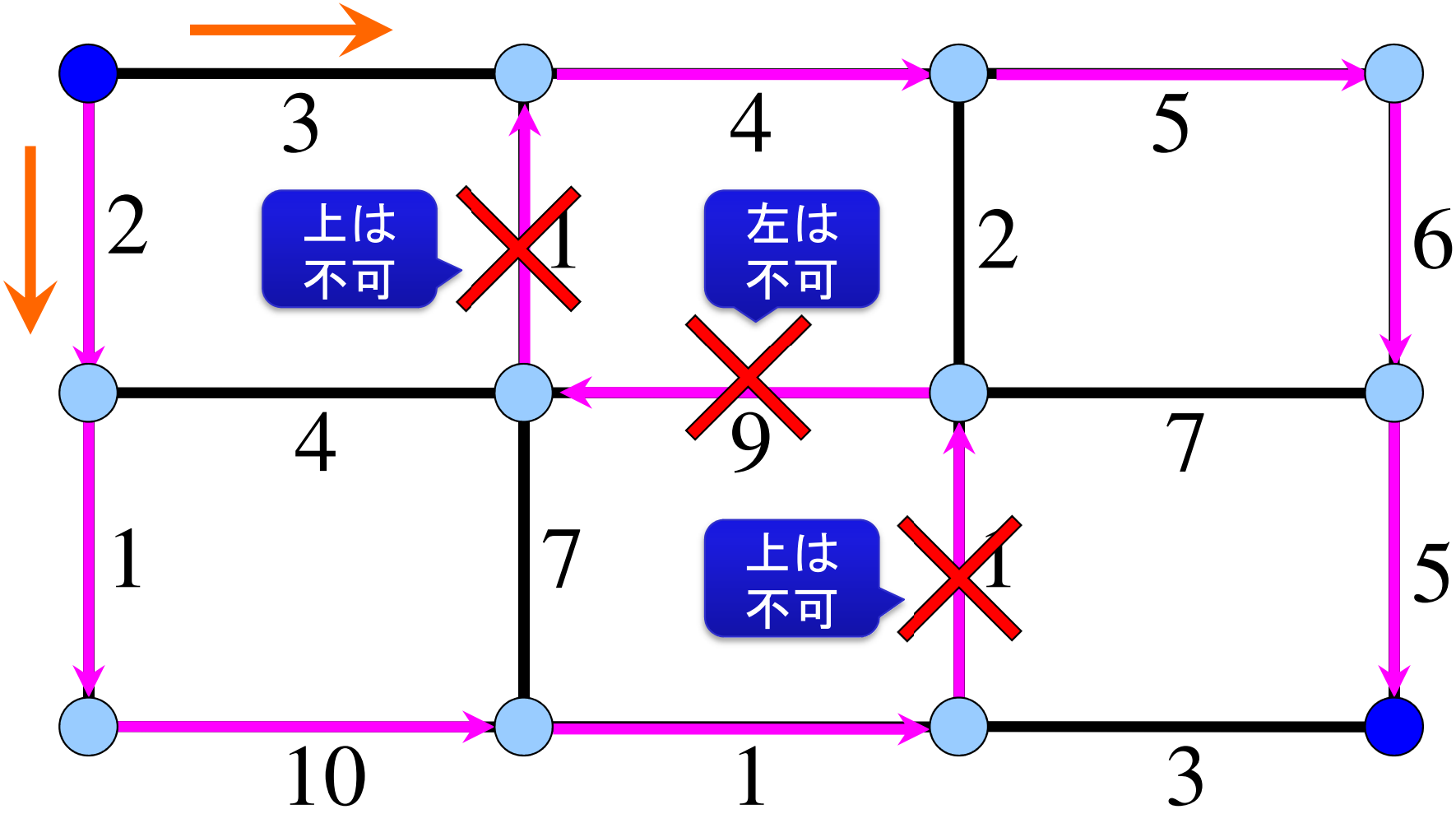
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

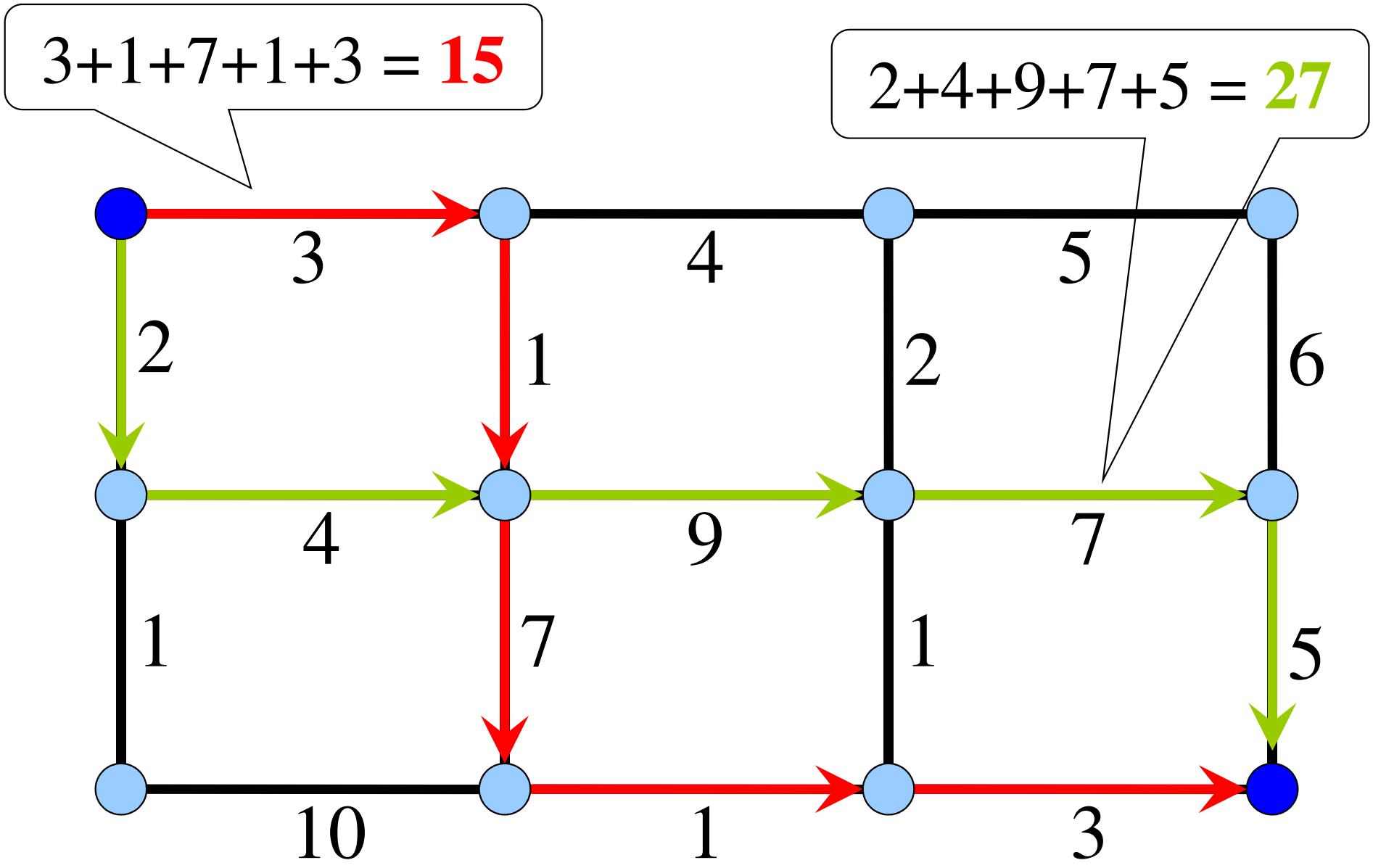
難しいなら易しくすればいいのさ！

制限した問題

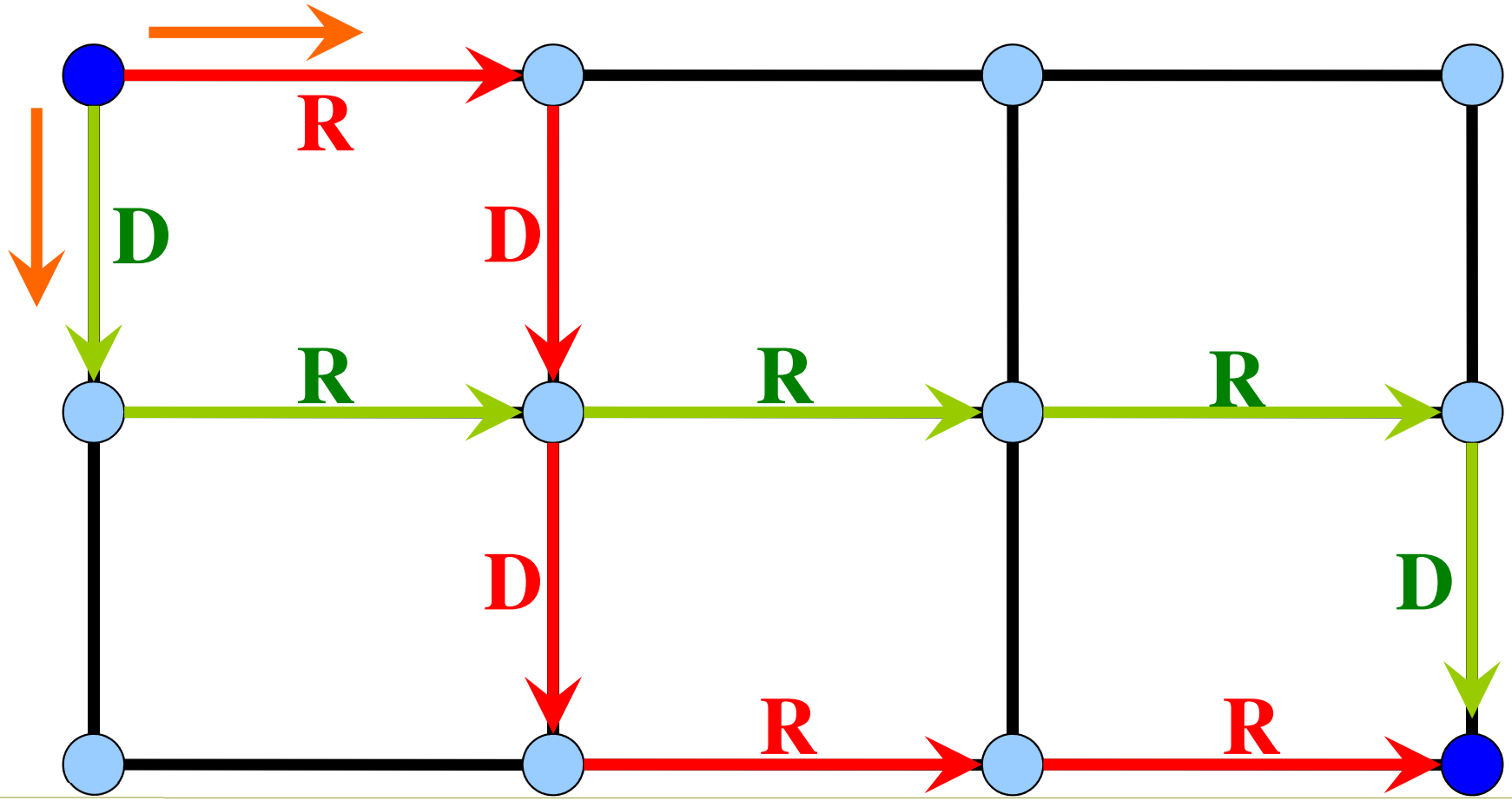
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



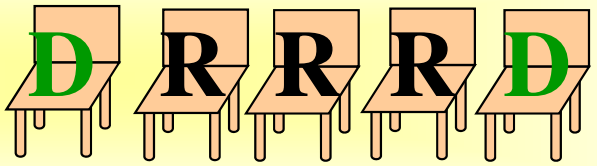
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



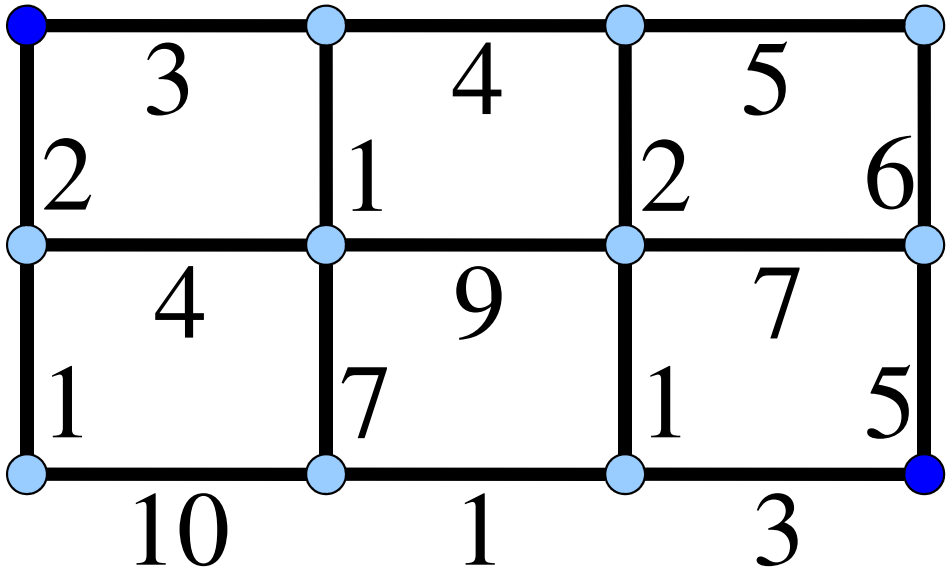
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

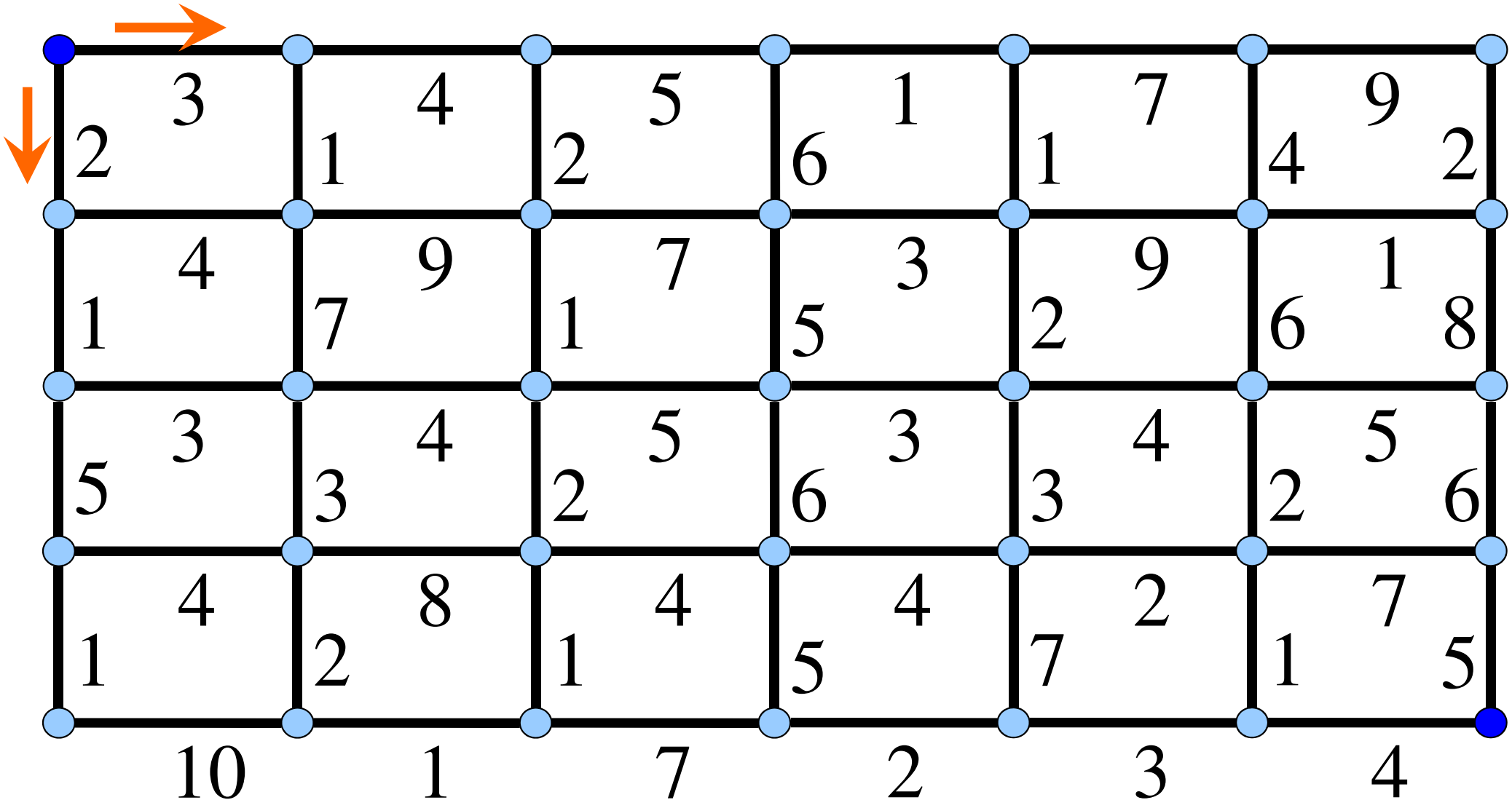
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

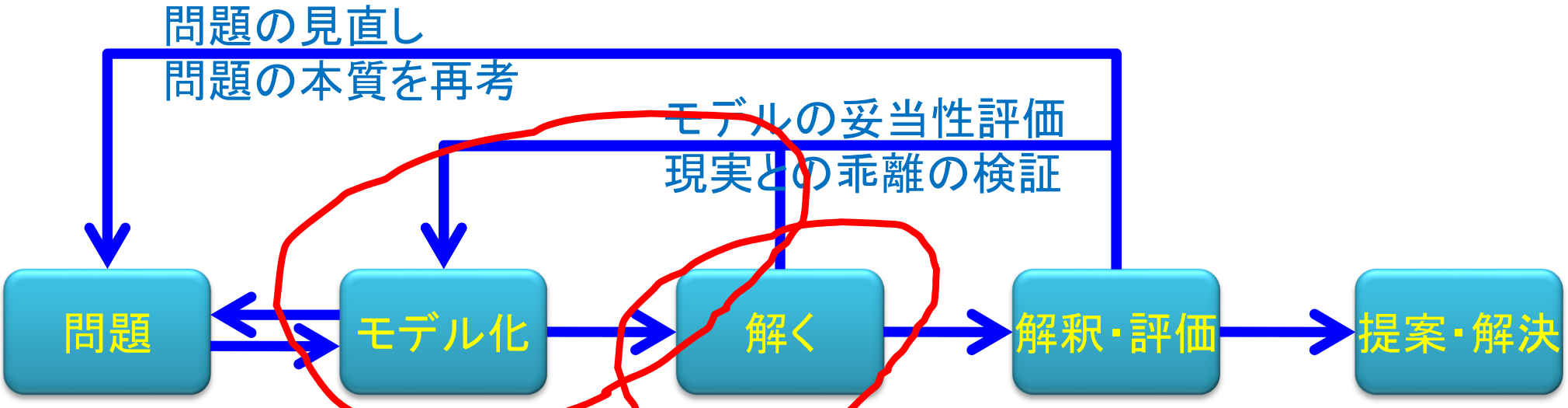
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

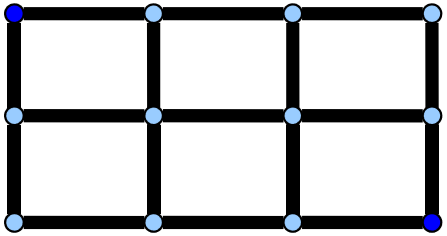
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

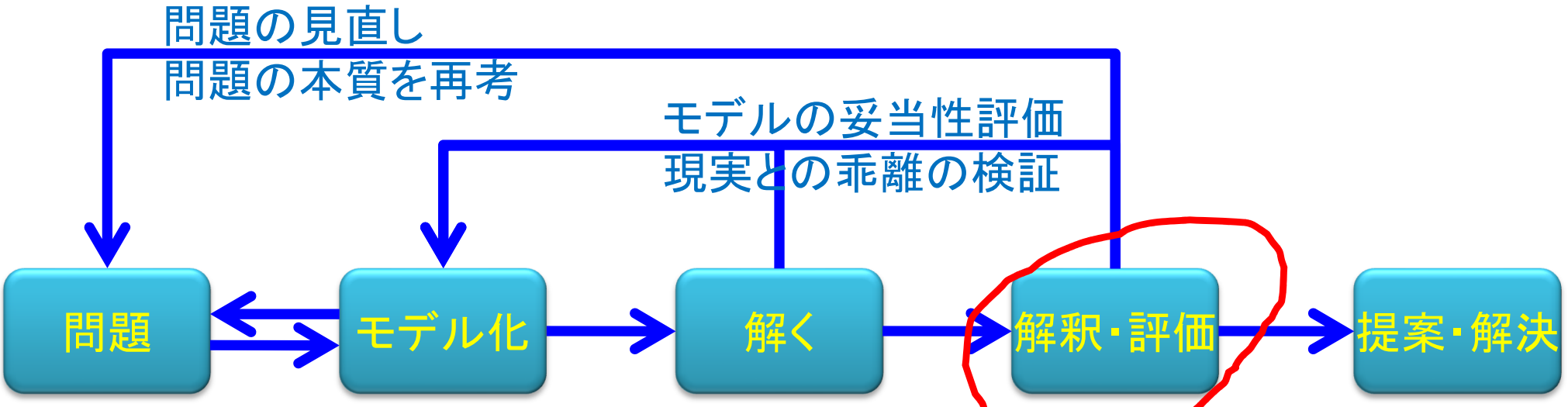
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

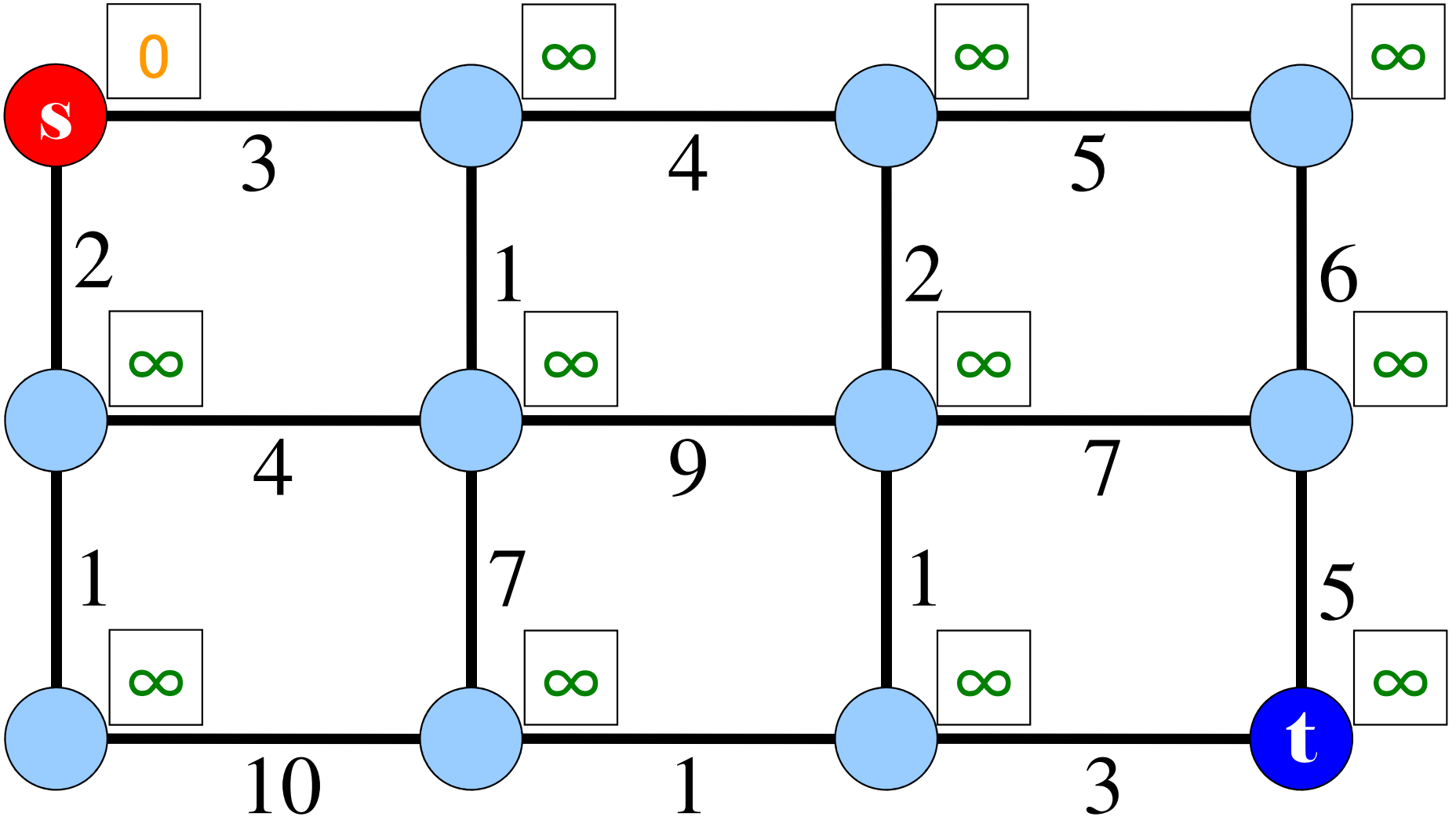
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

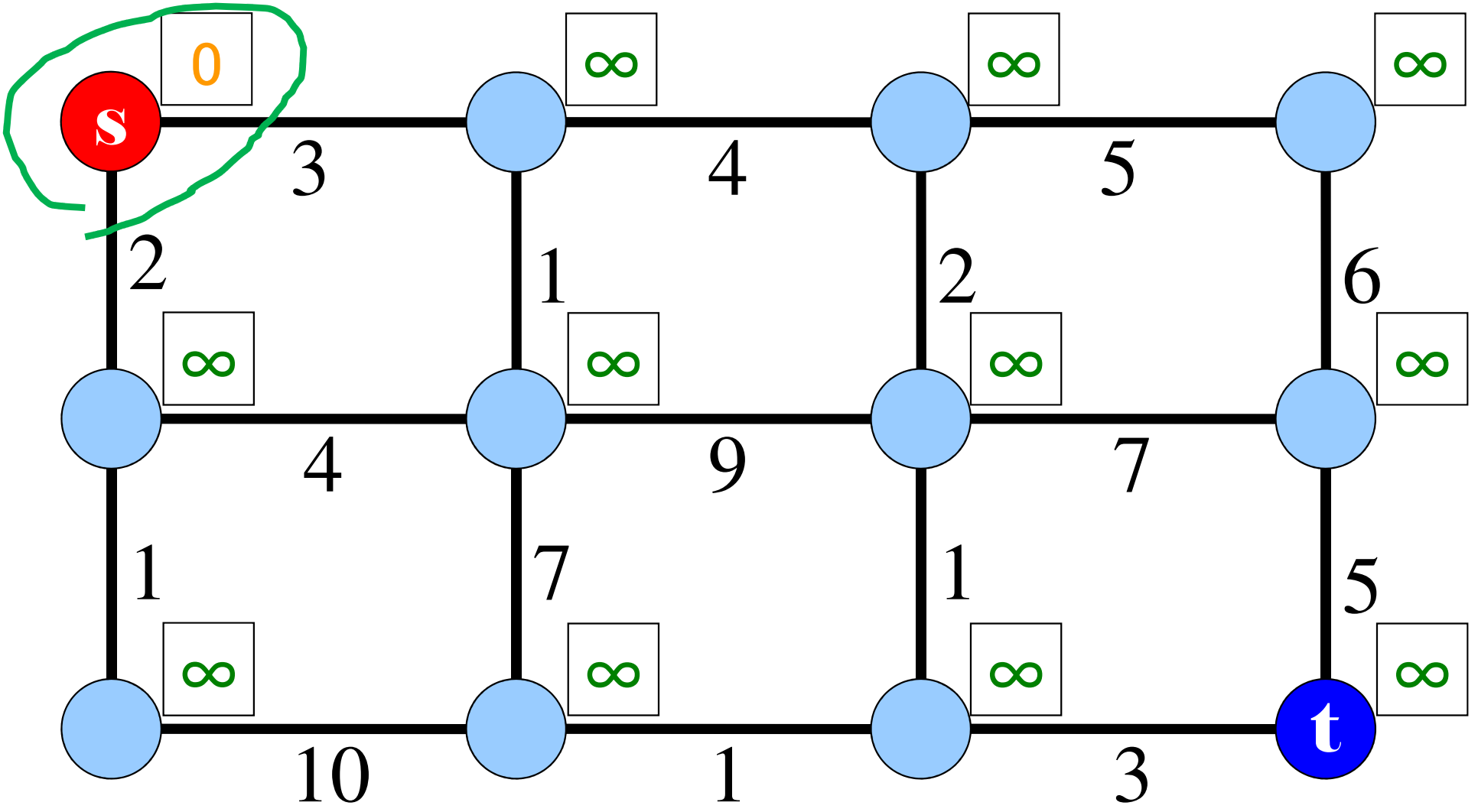
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



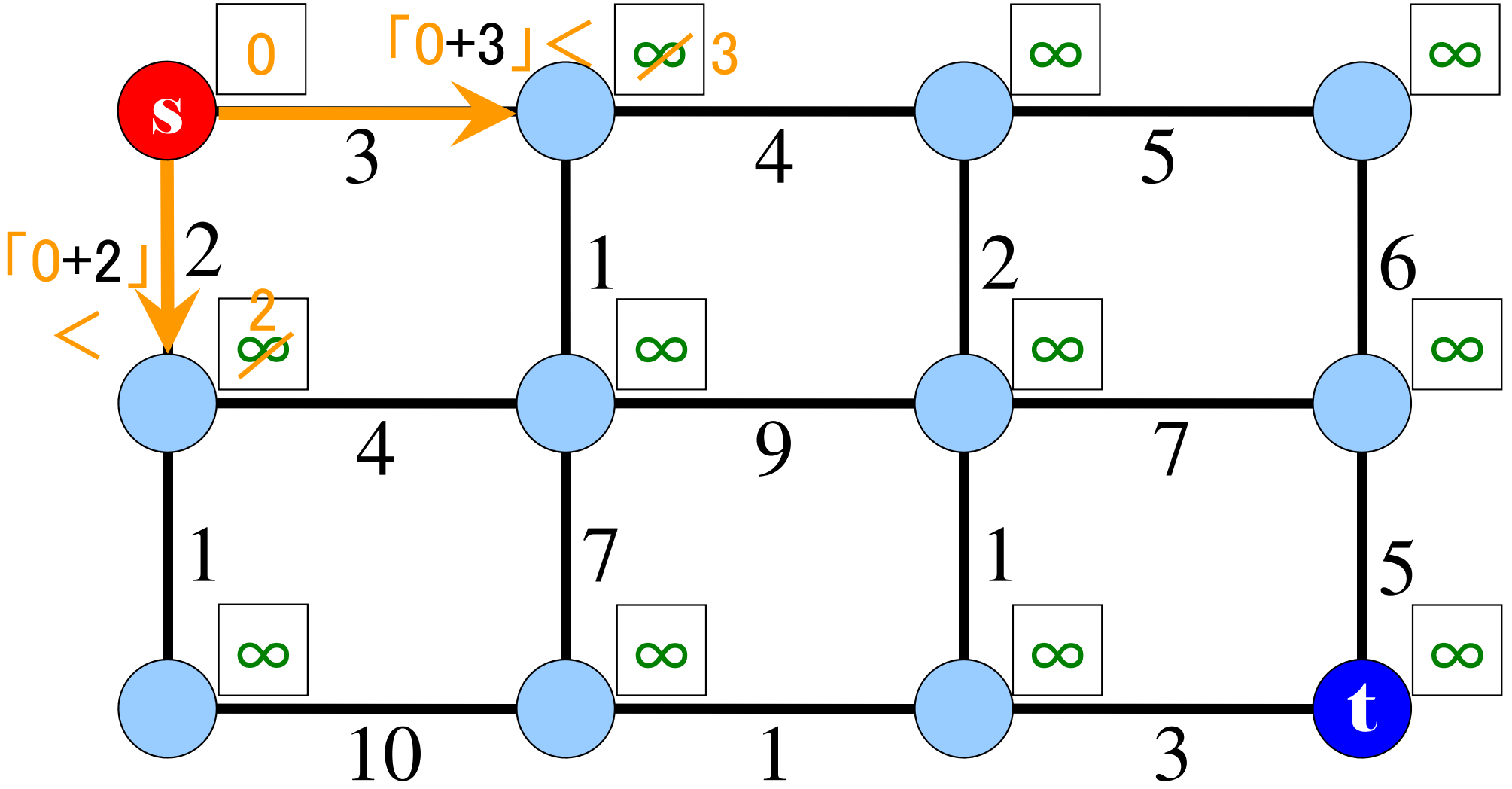
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

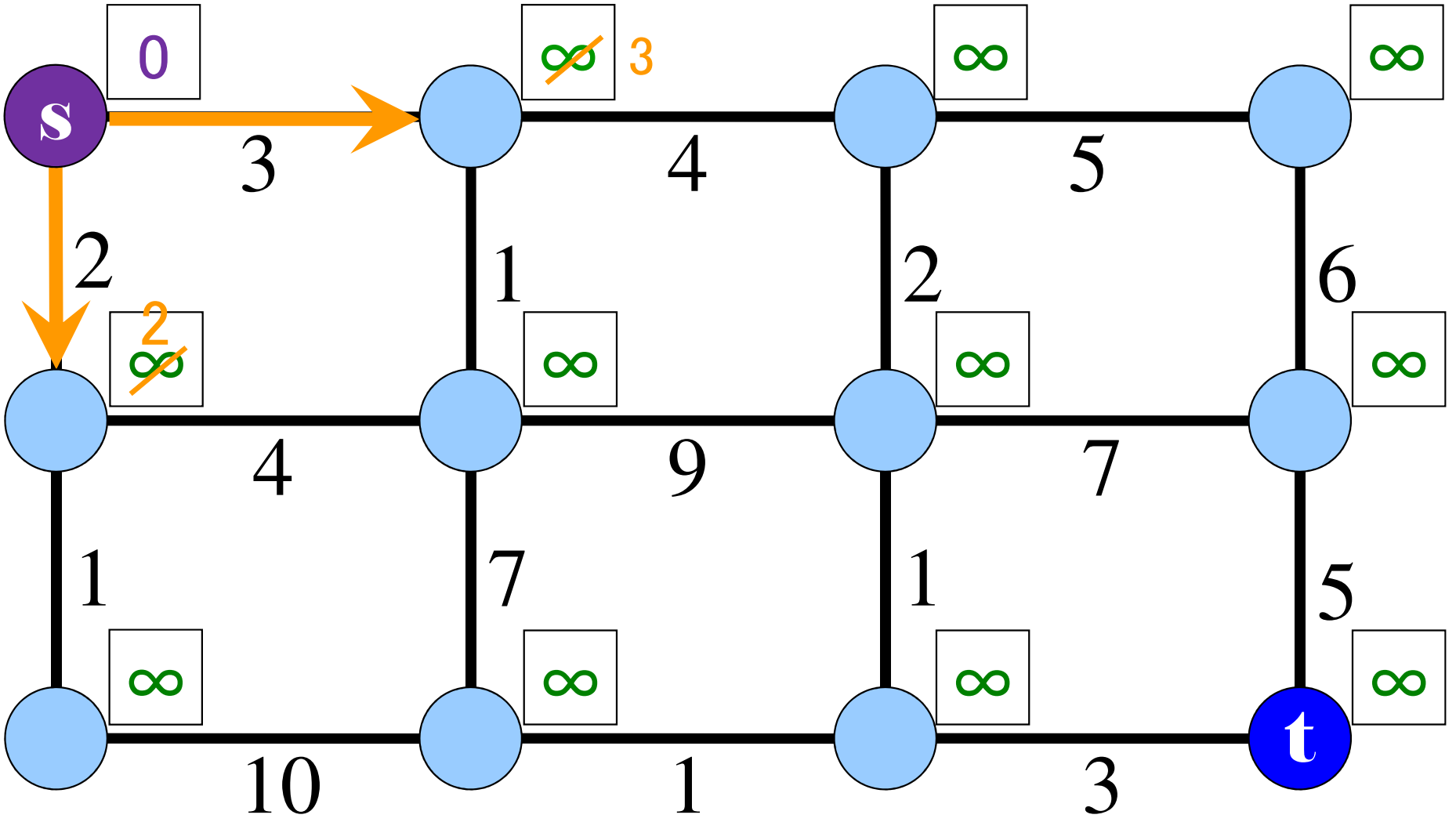


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

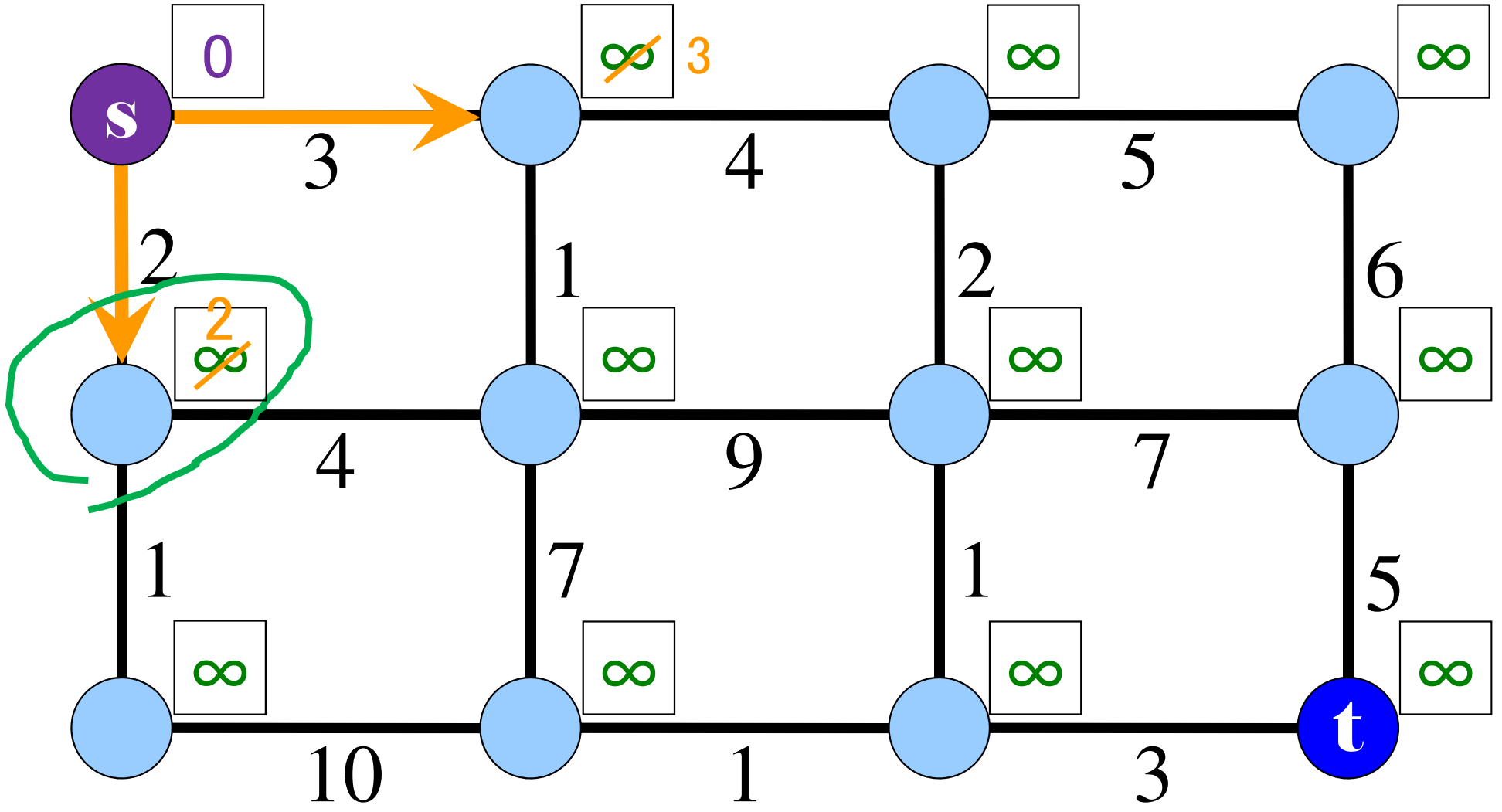
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



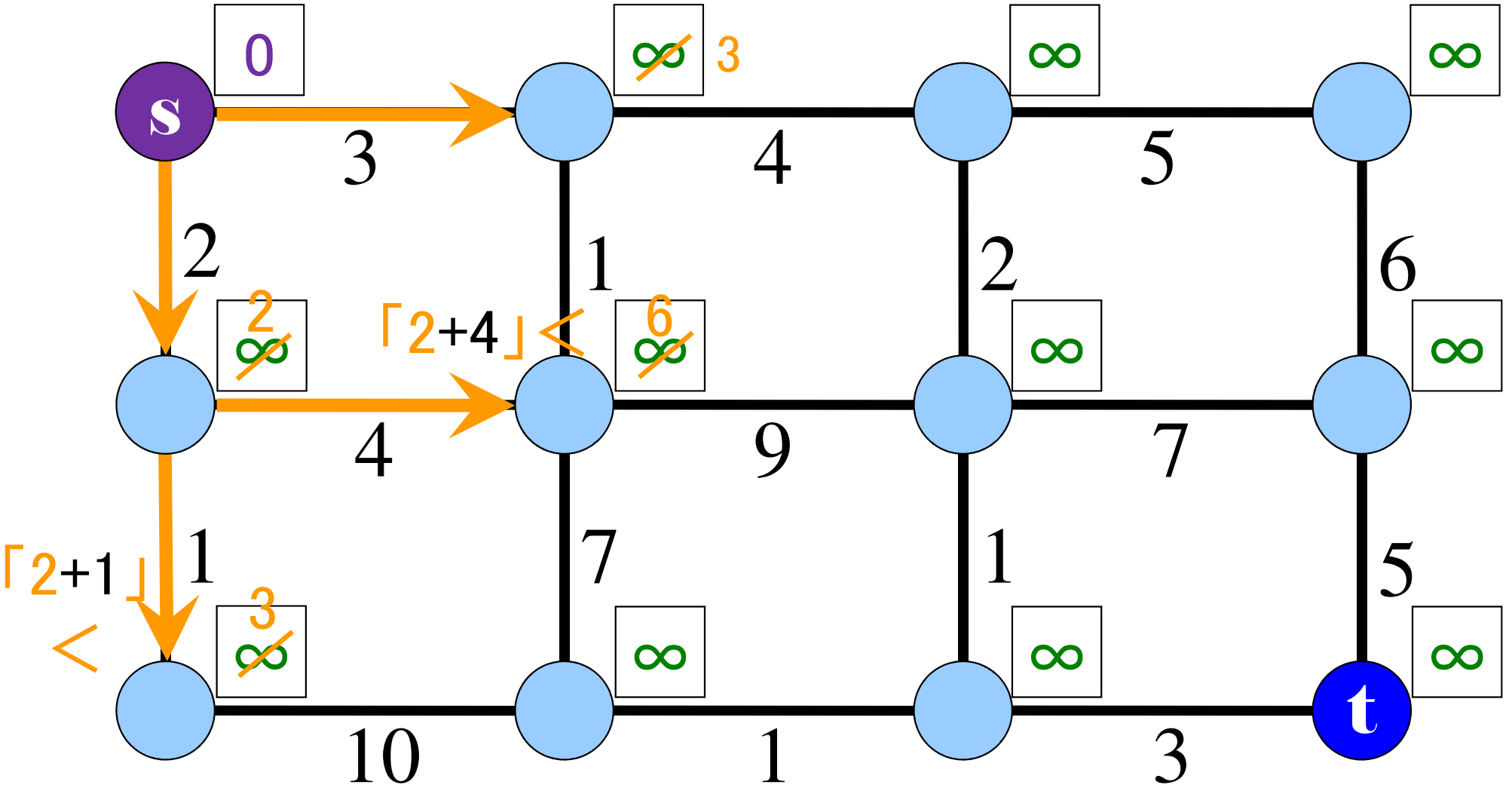
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

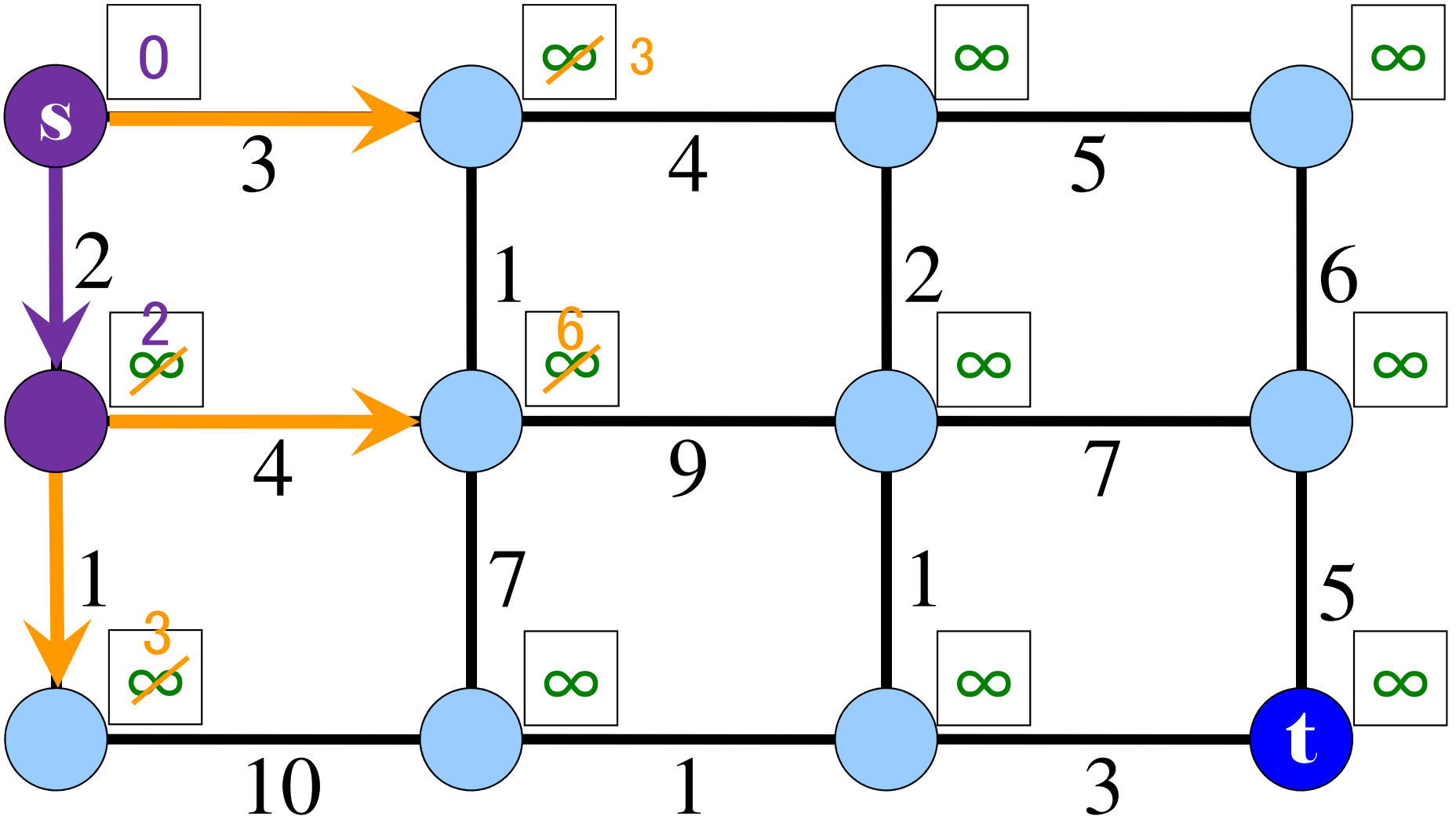


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

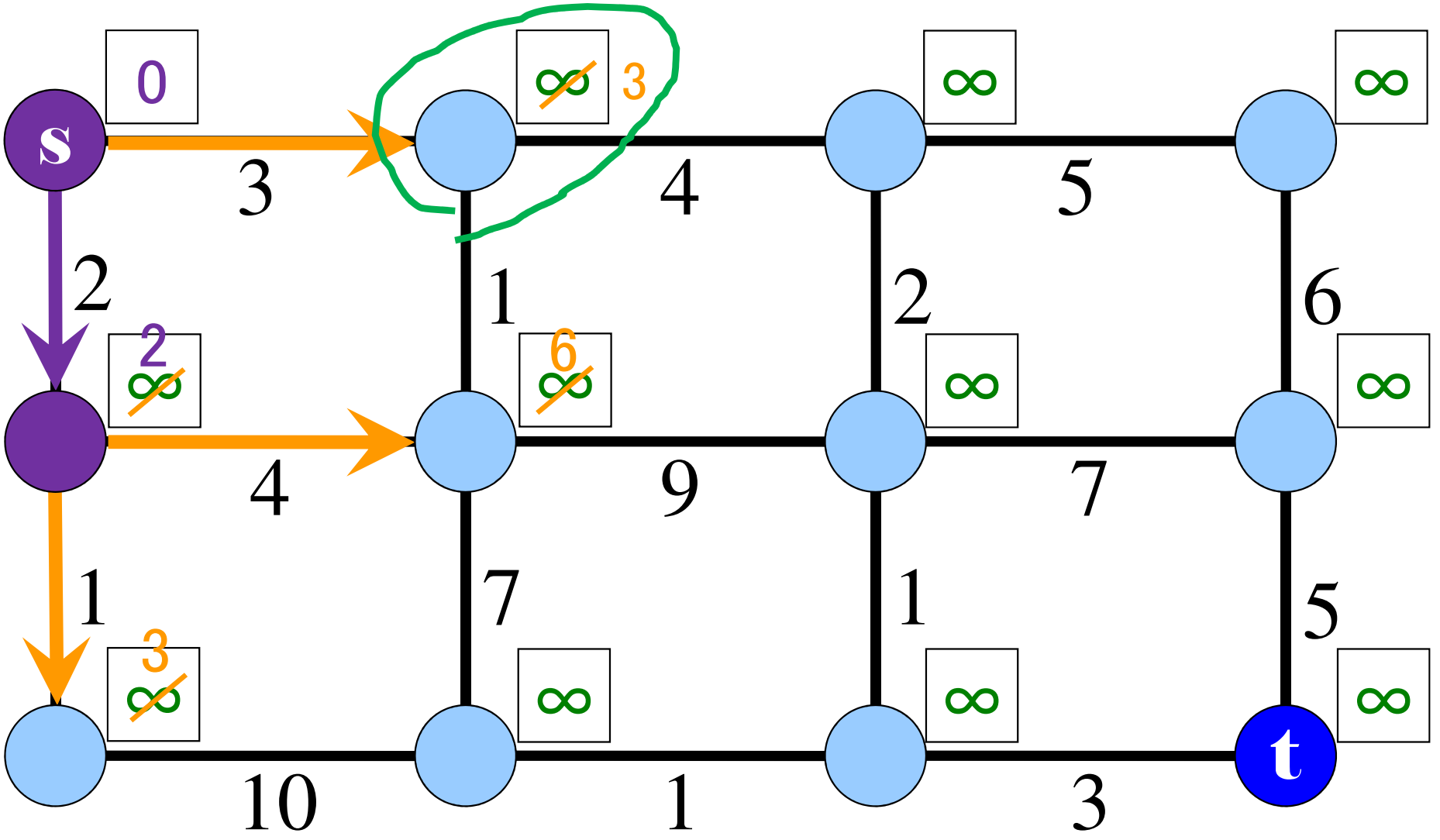
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



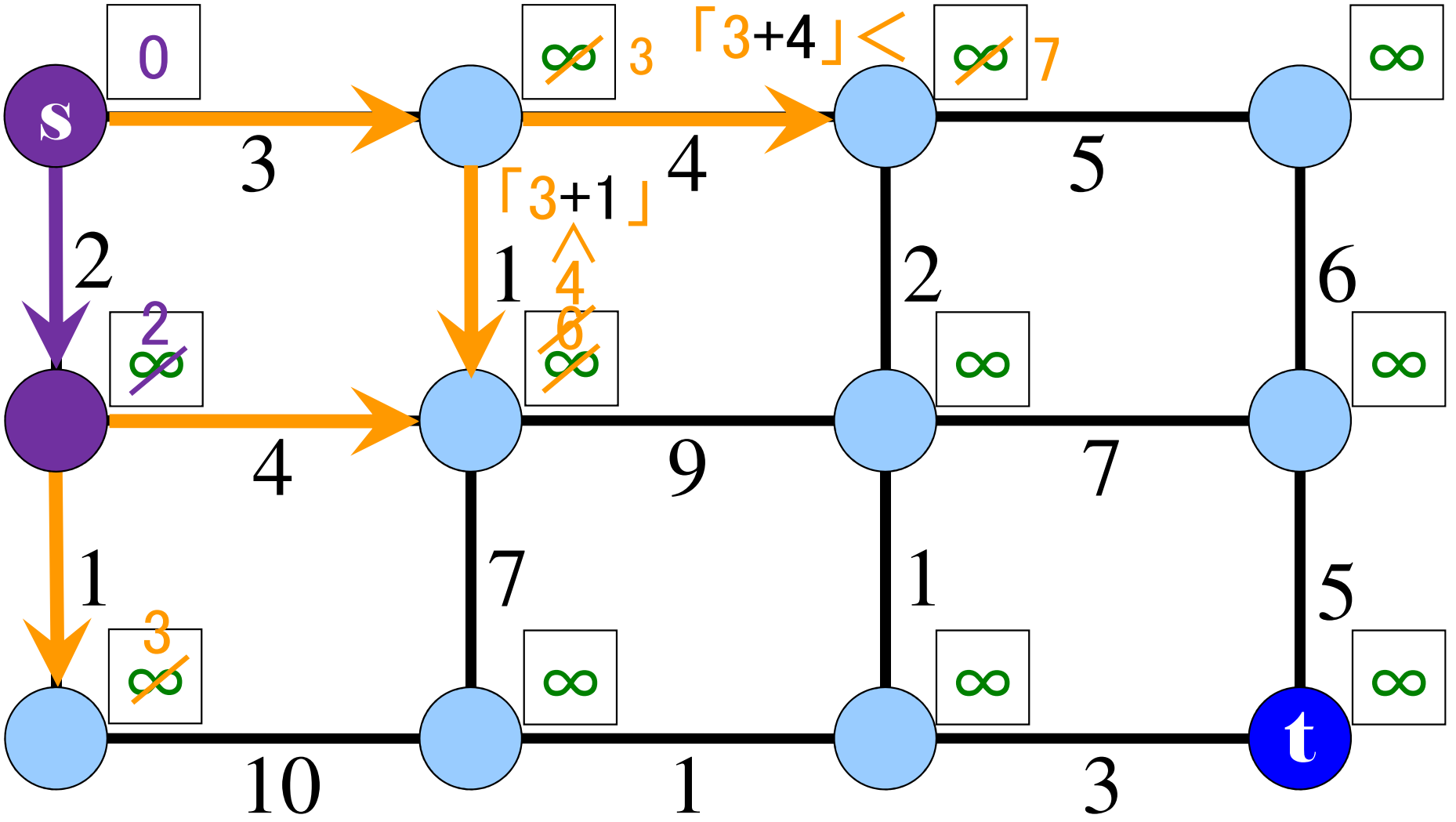
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

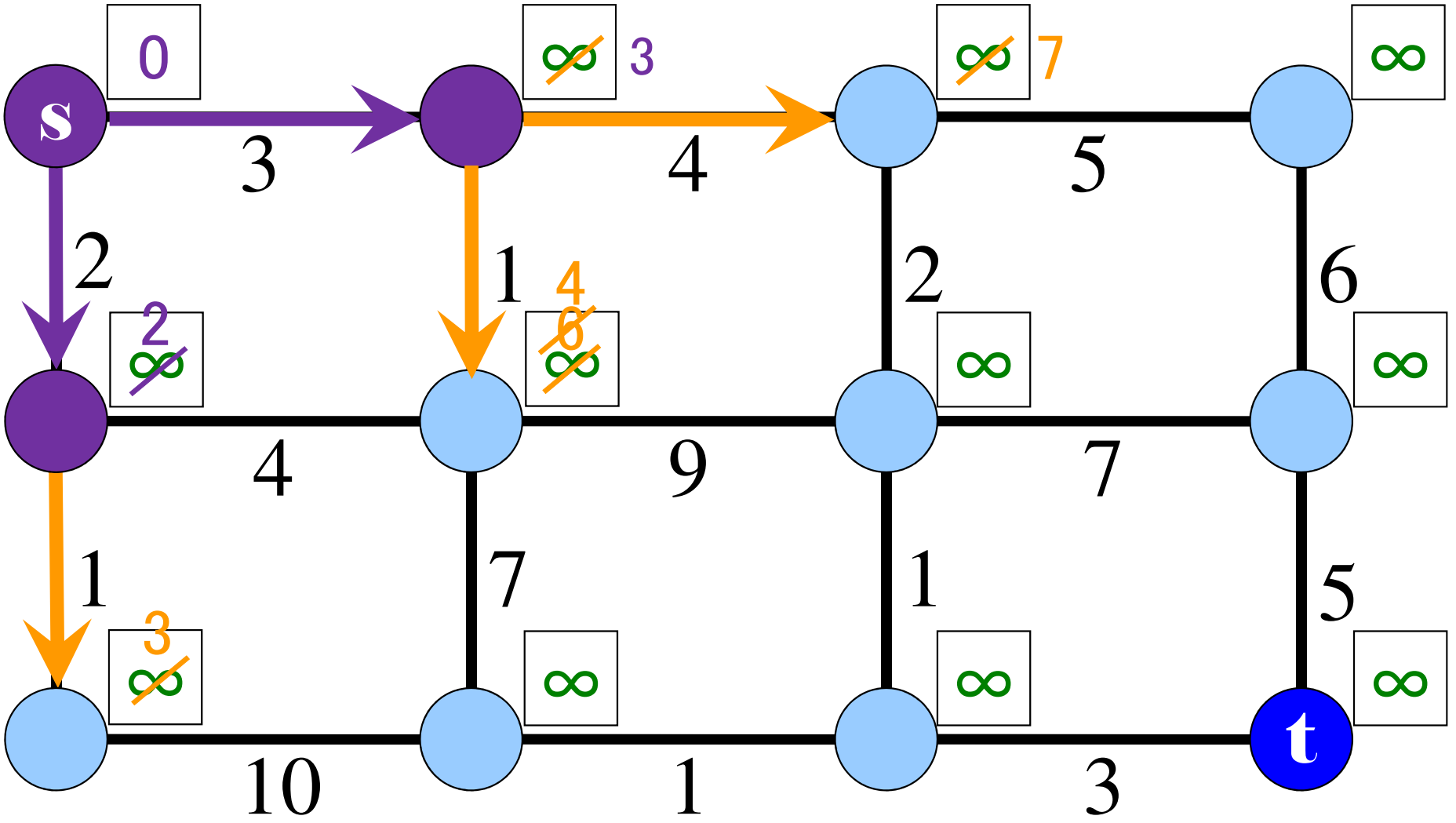


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

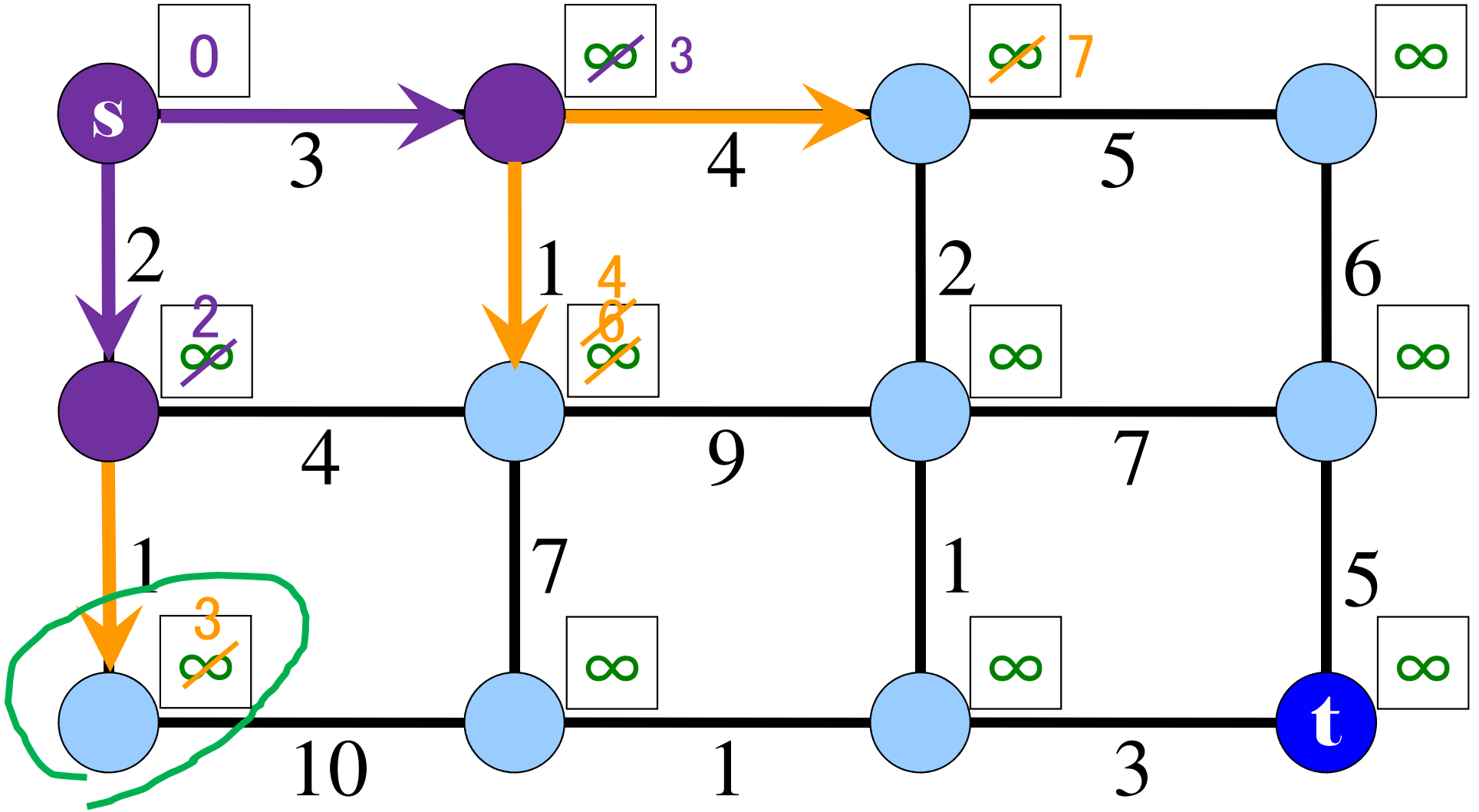
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



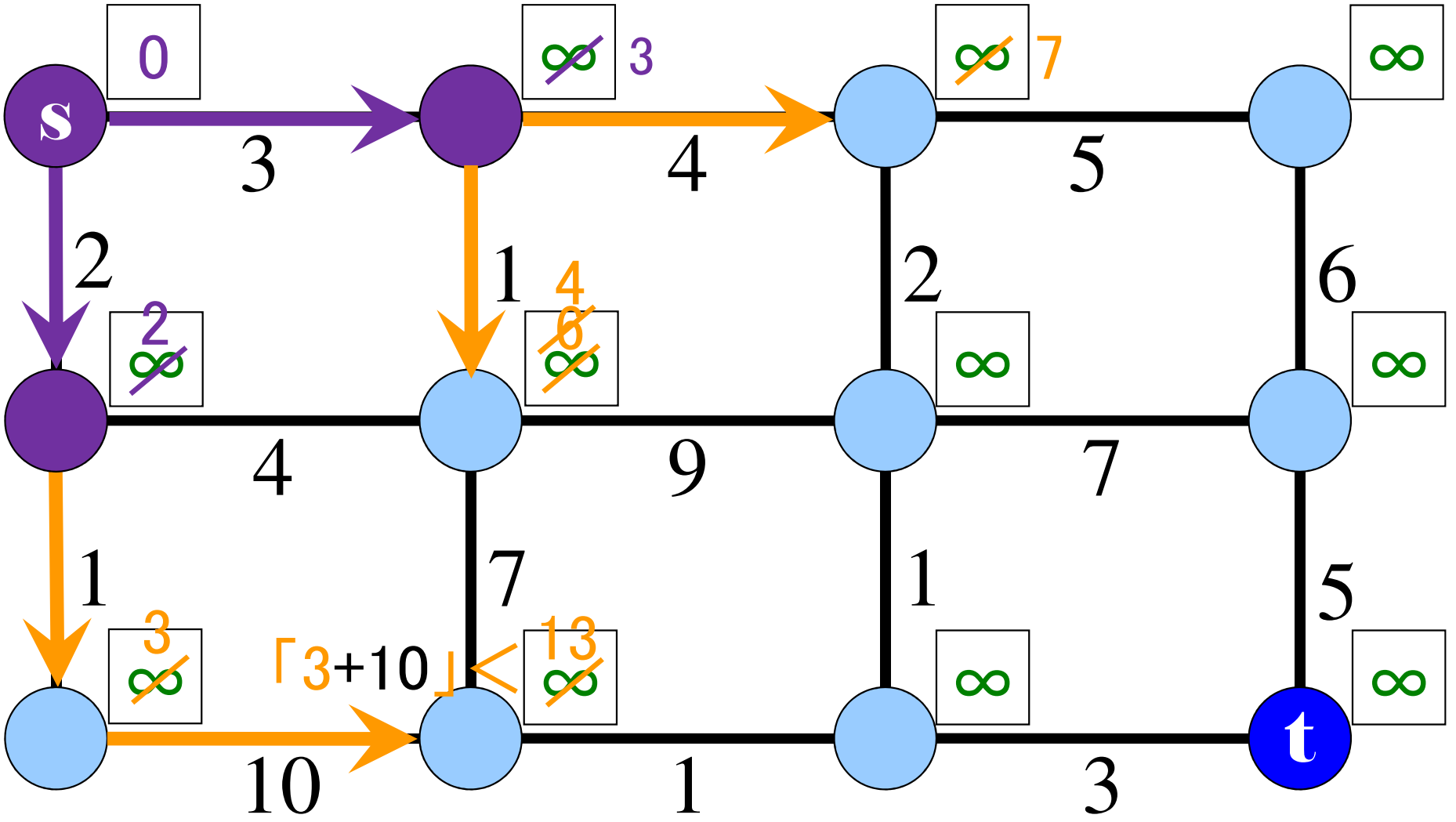
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

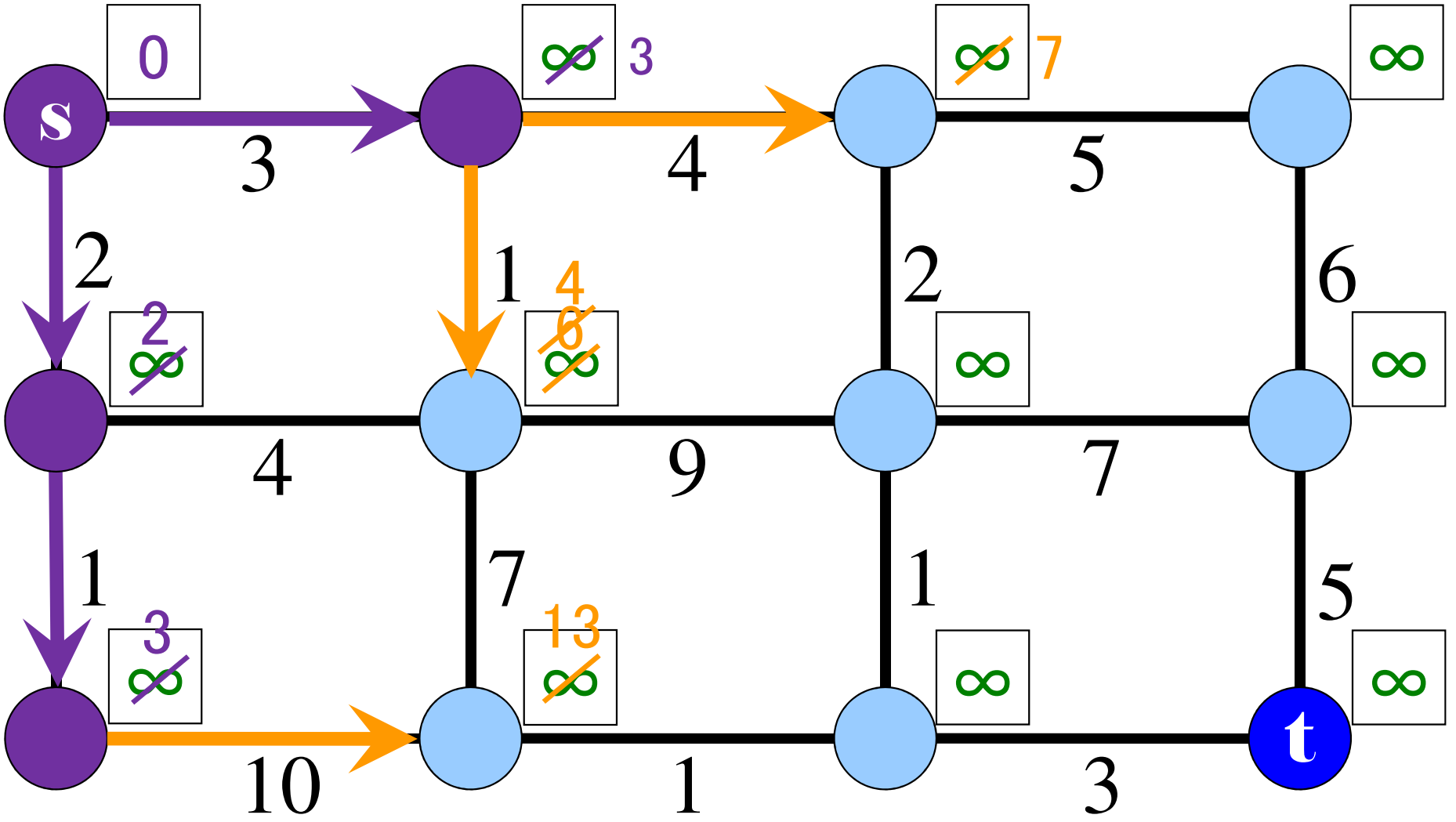


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

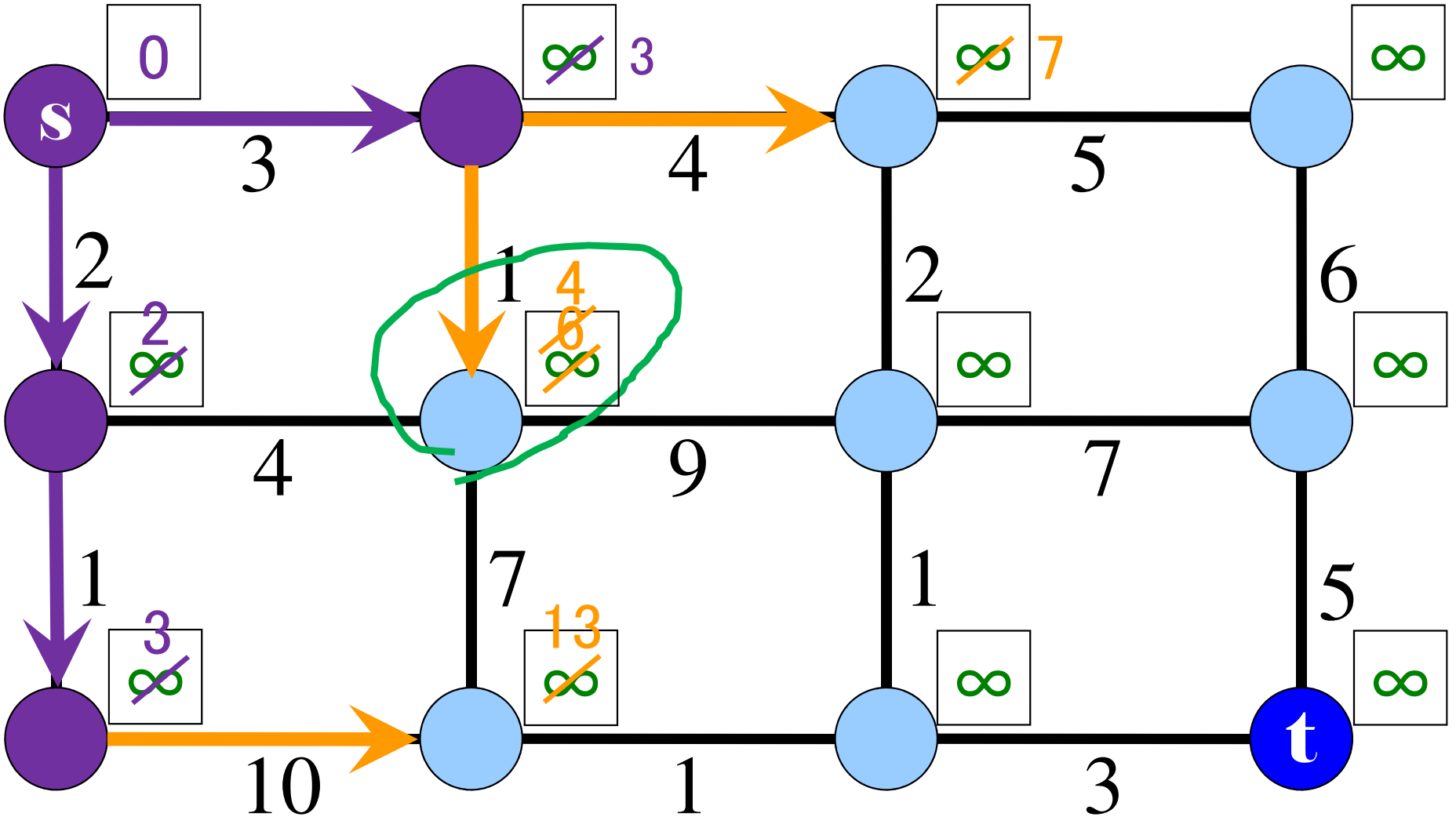
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



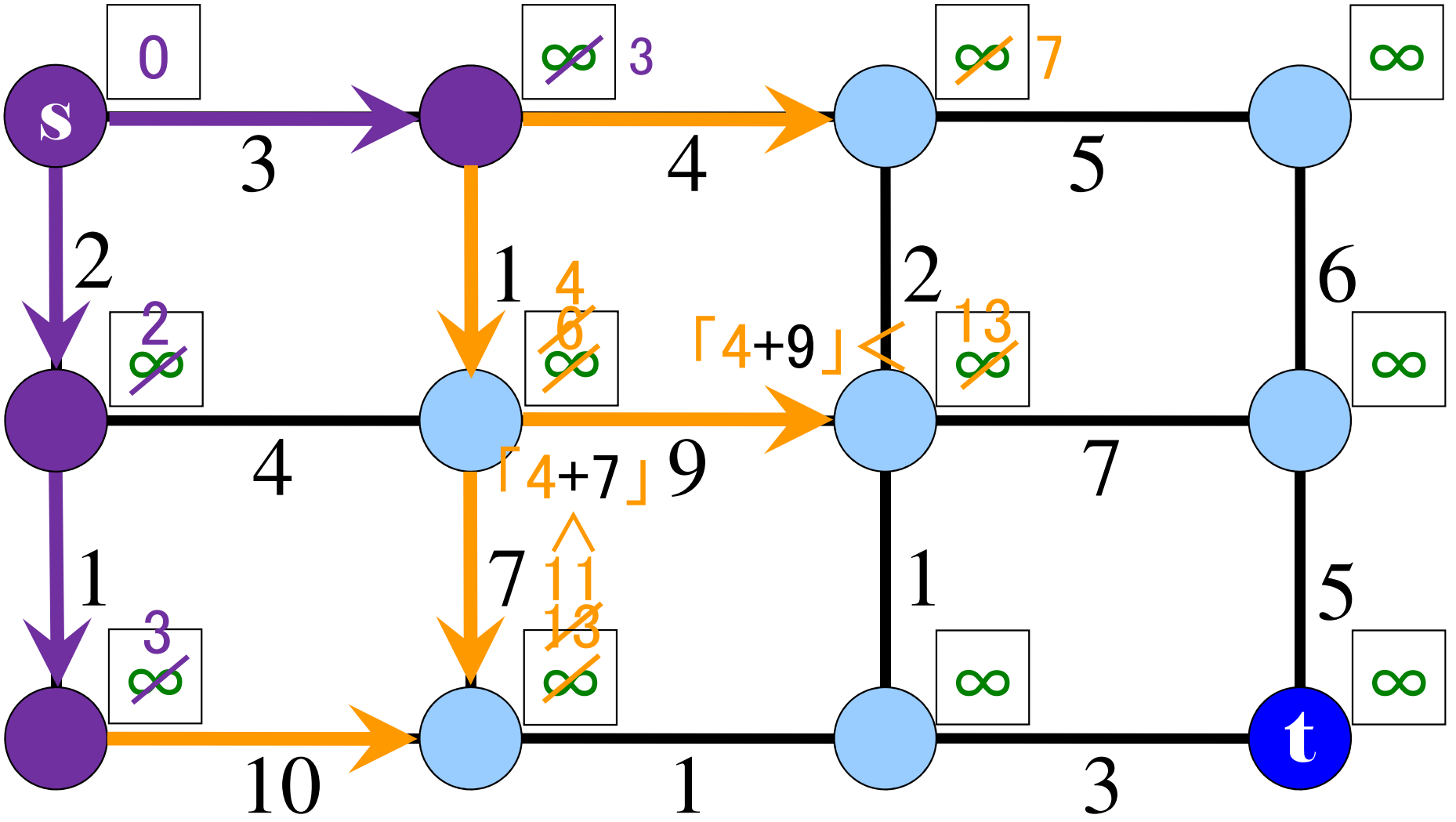
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

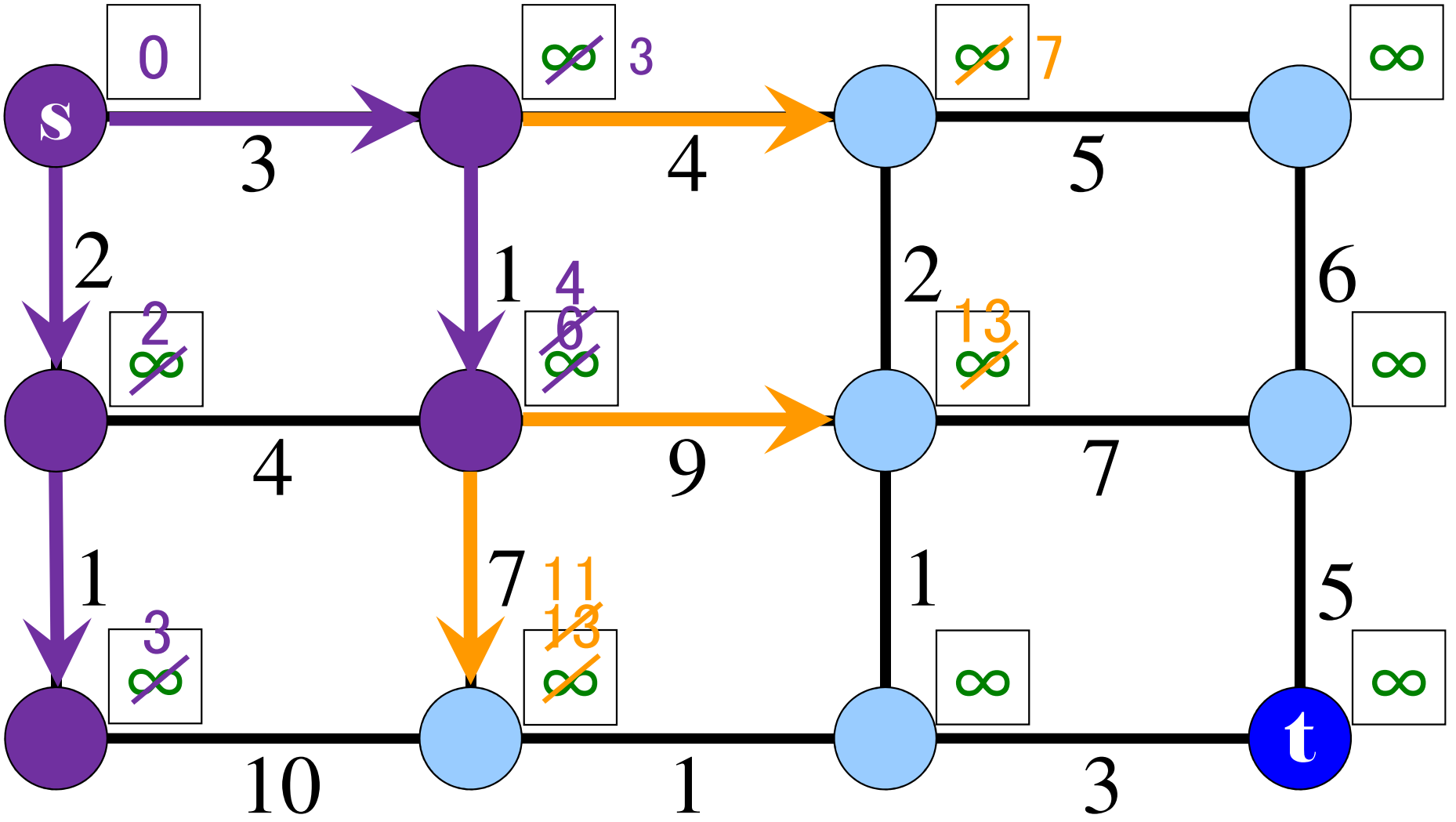


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

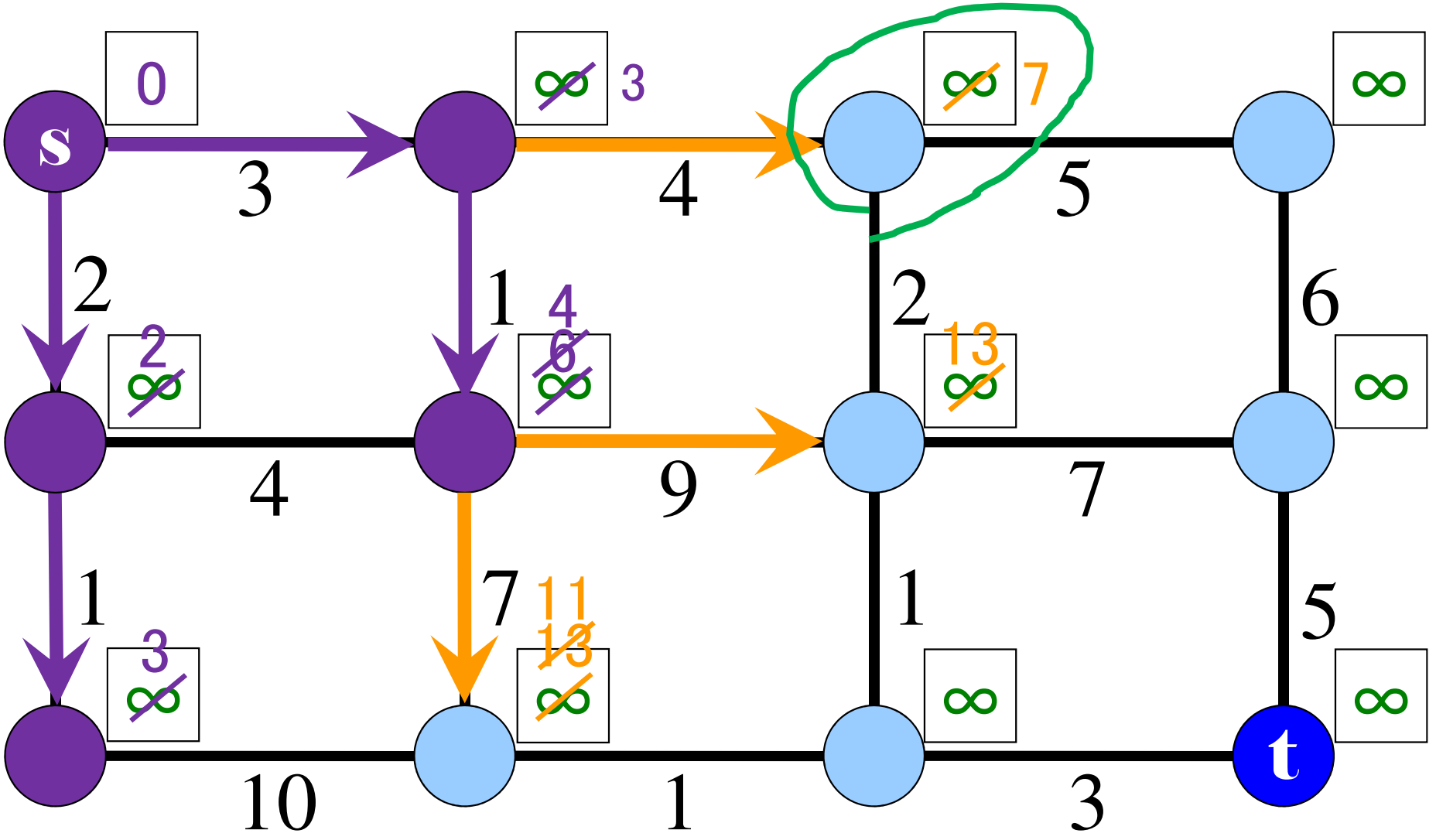
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



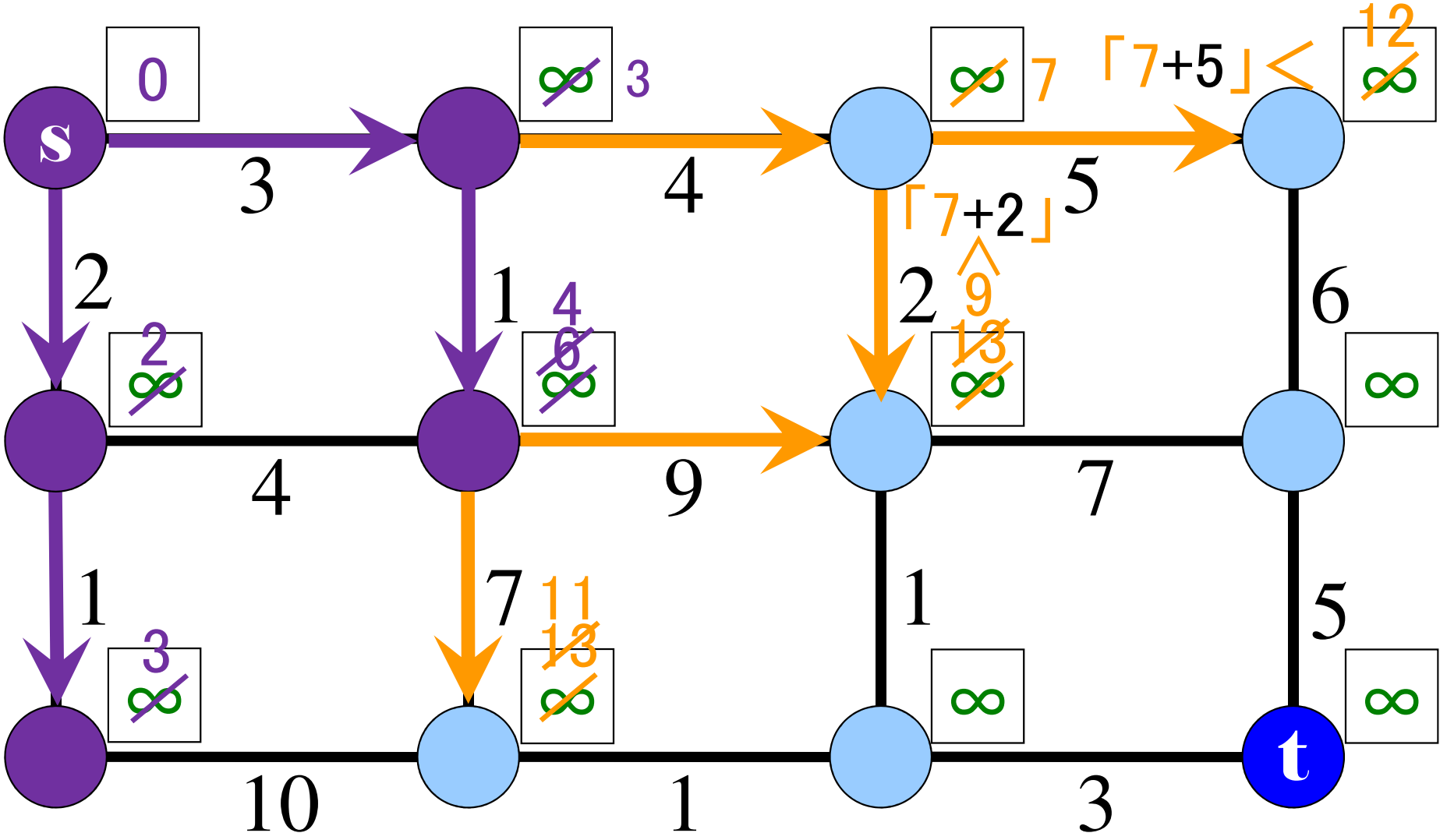
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

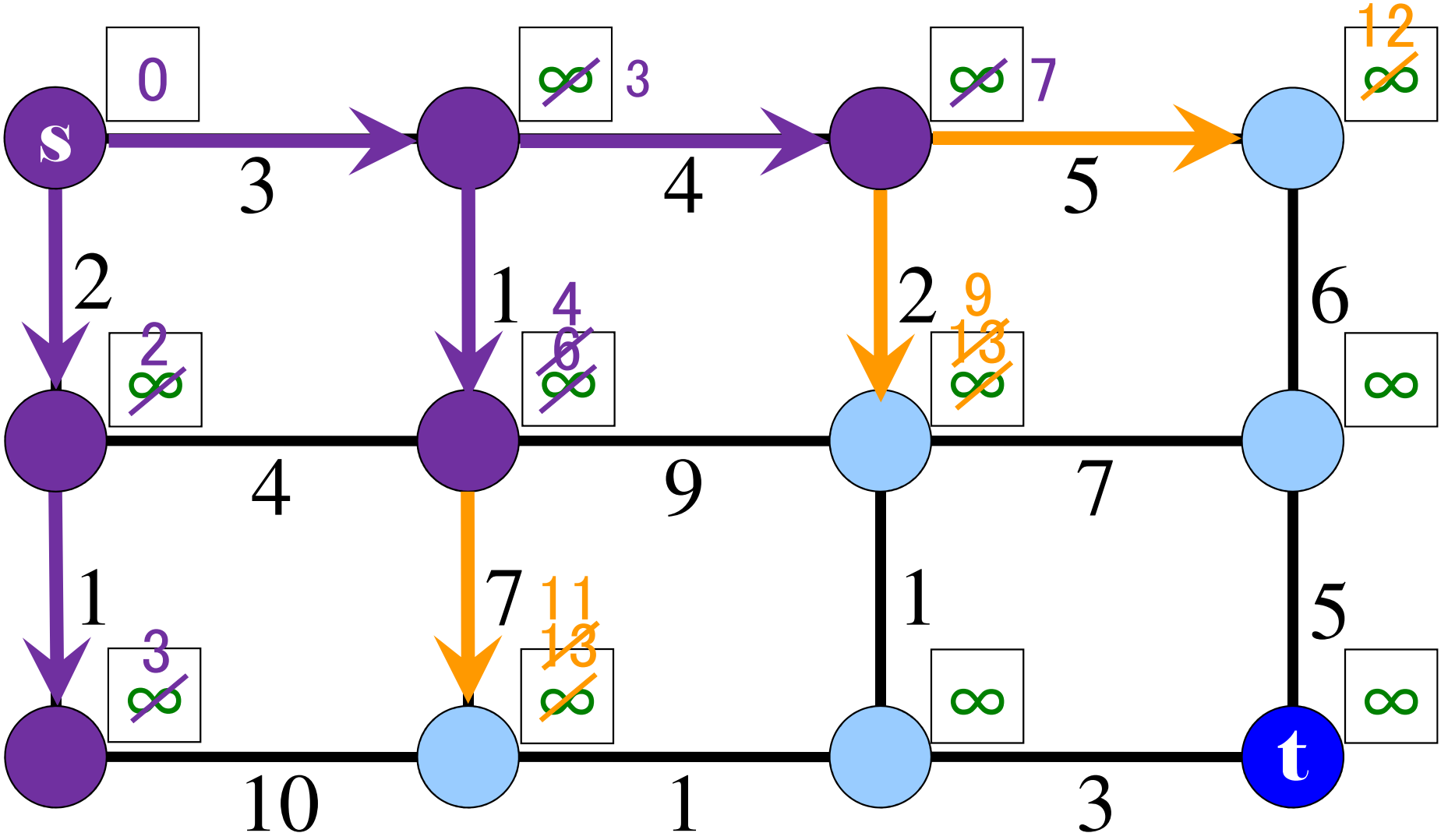


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

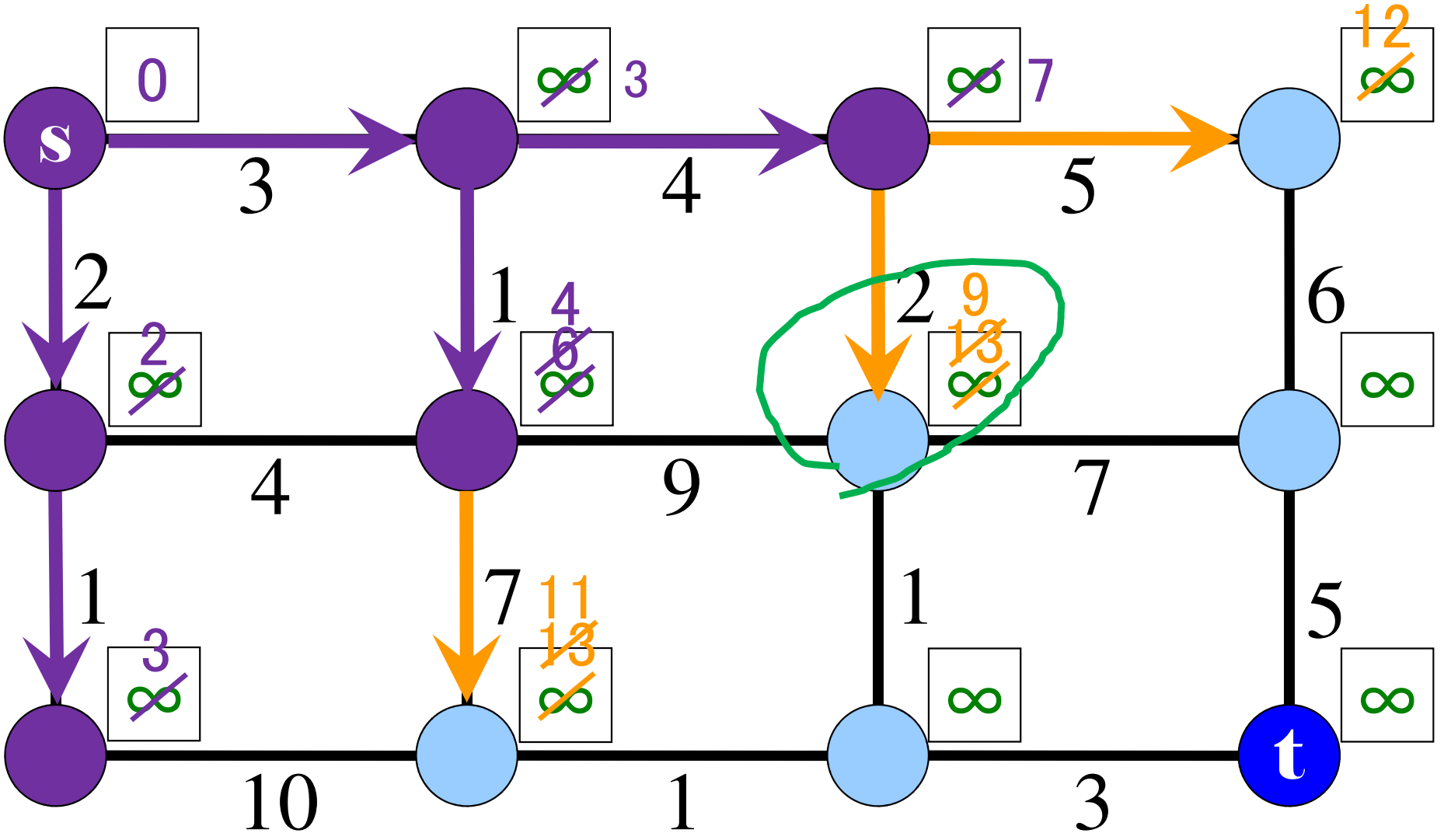
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



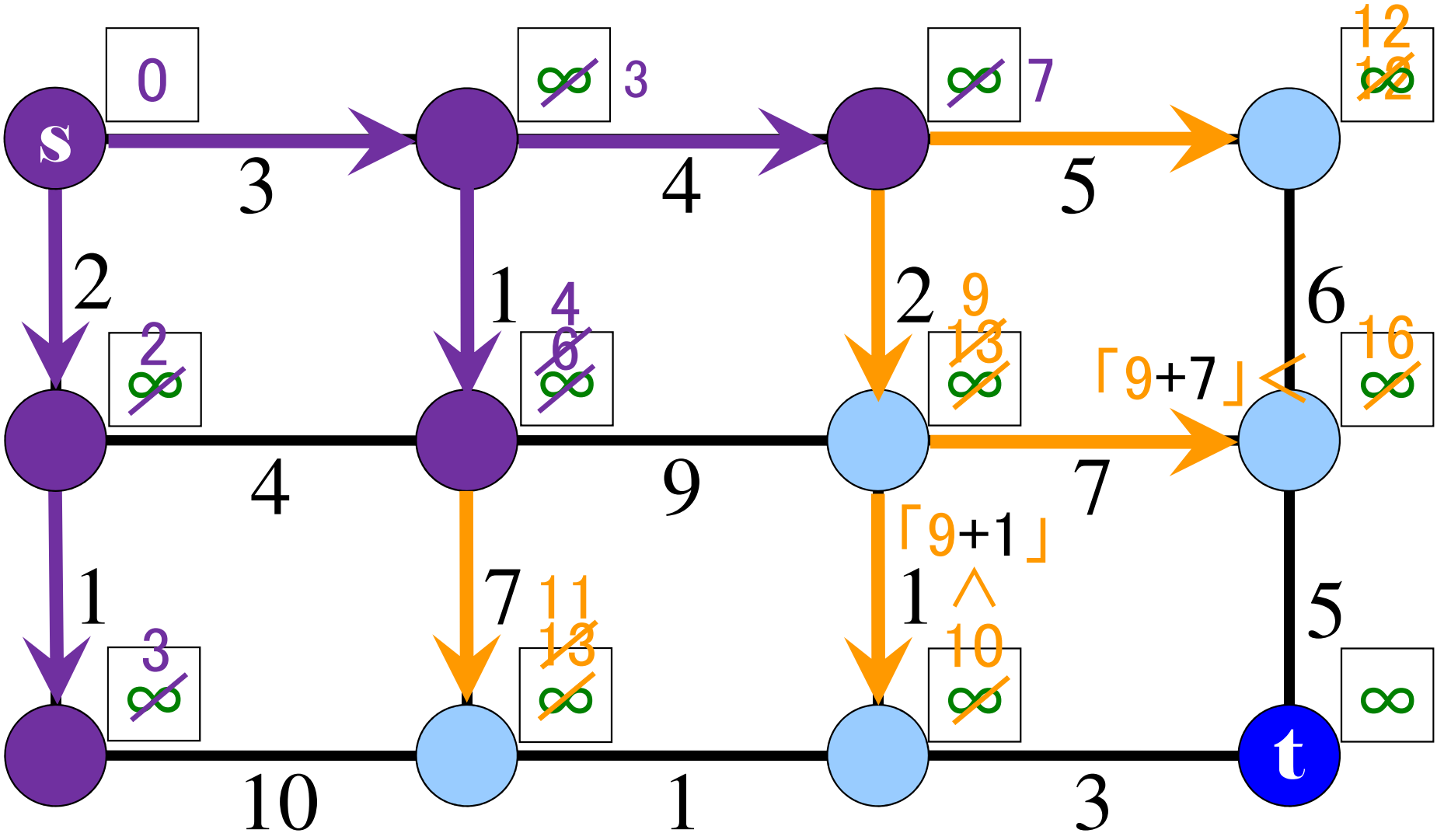
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

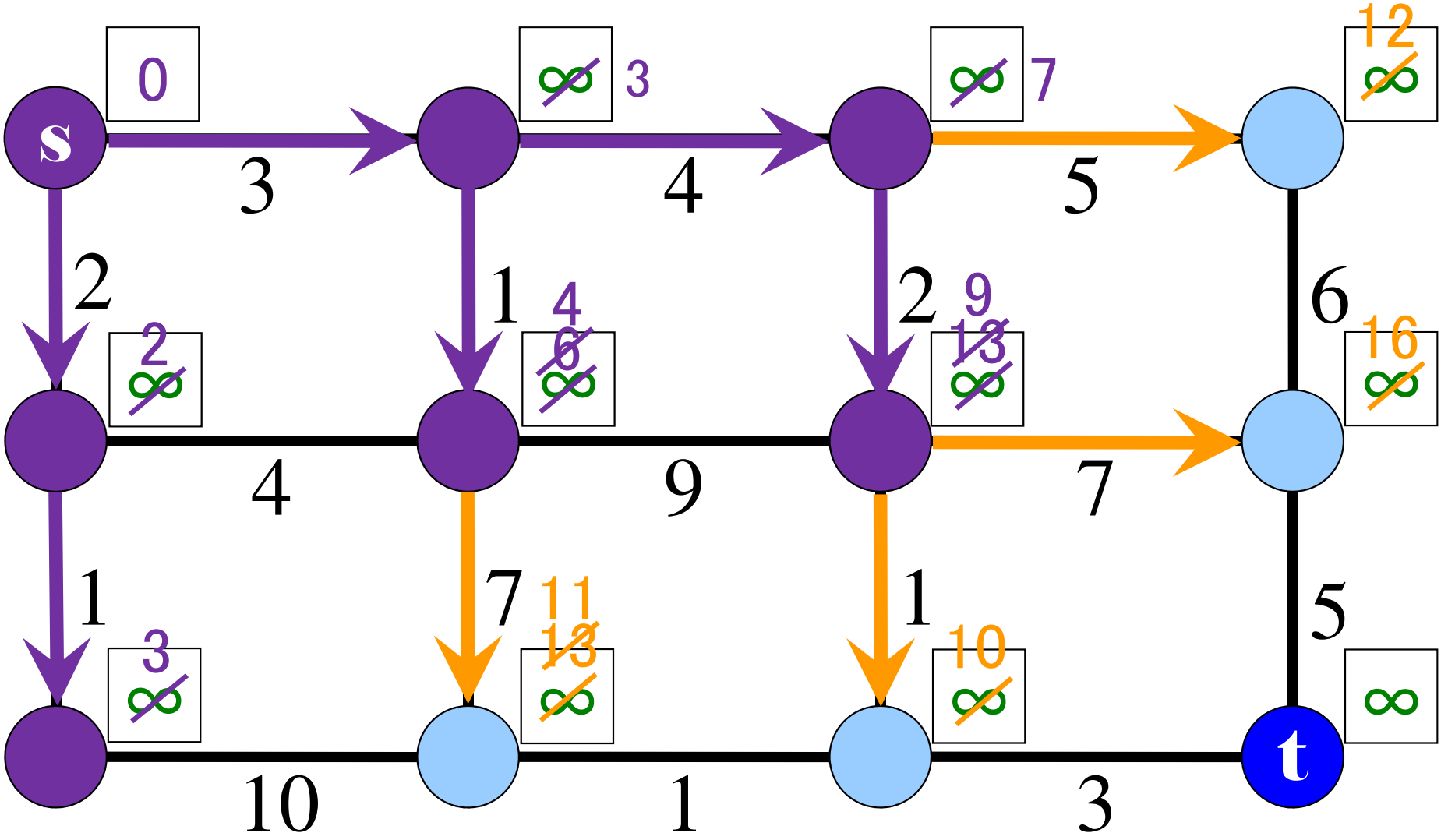


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

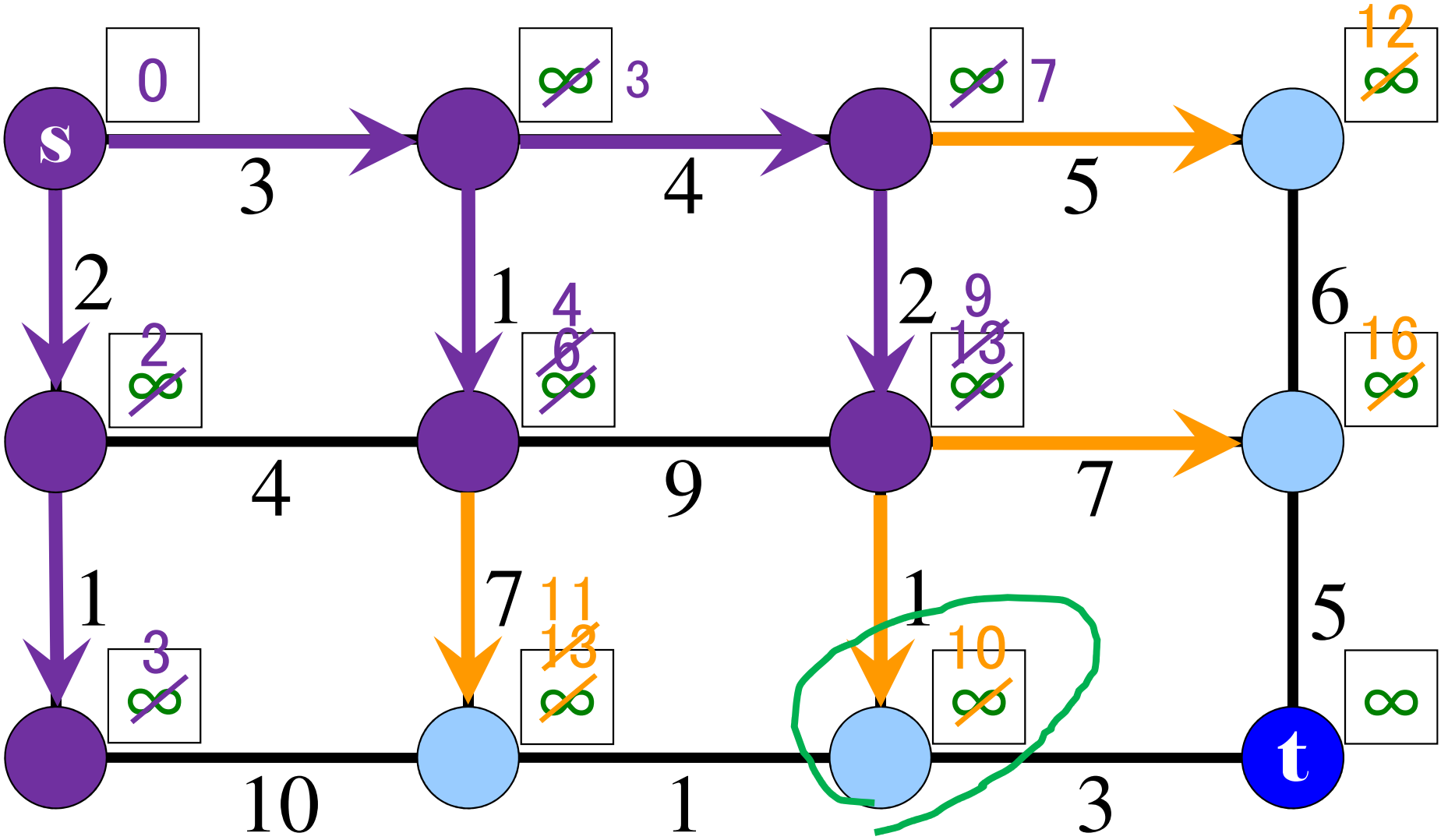
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



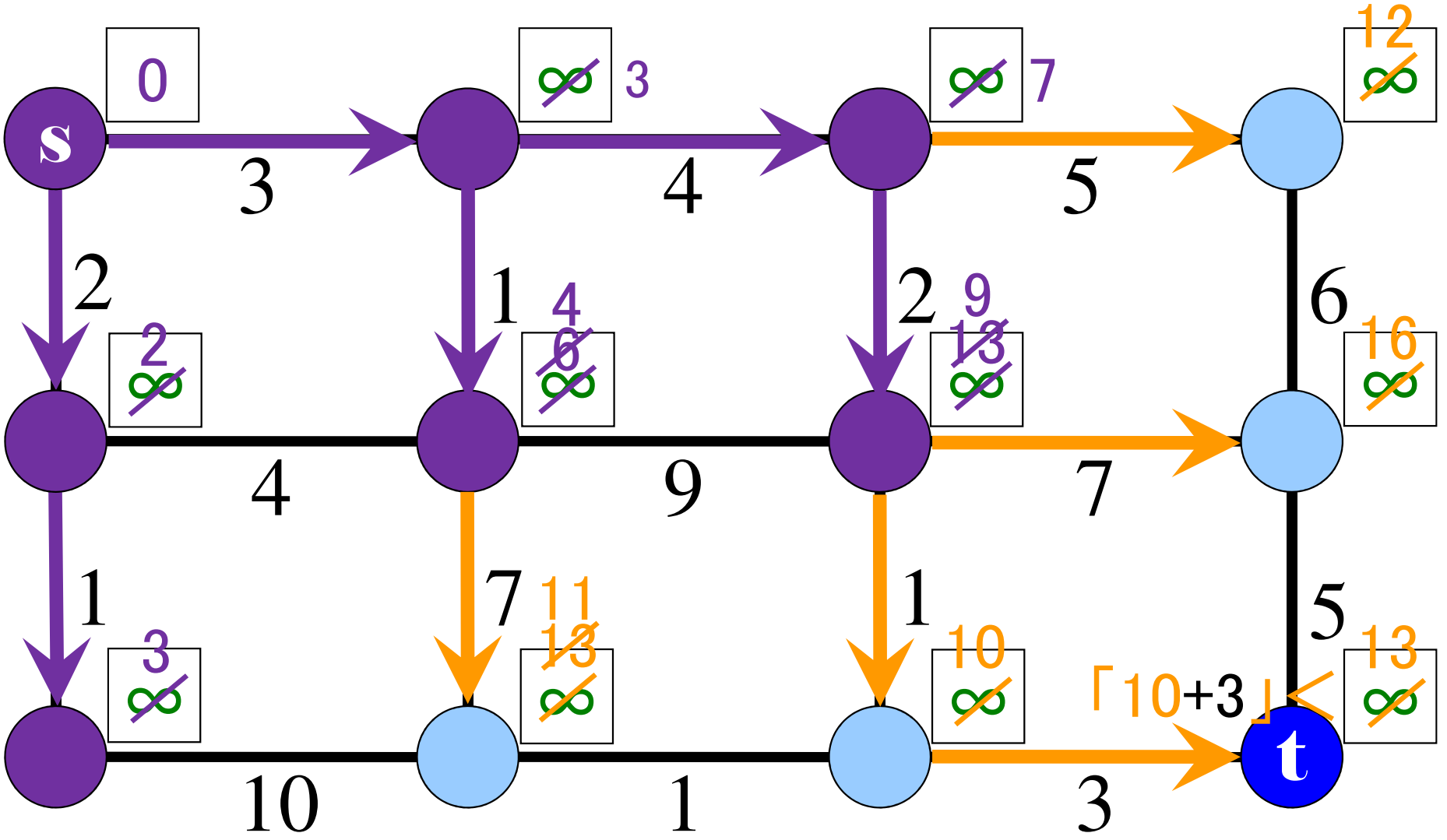
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

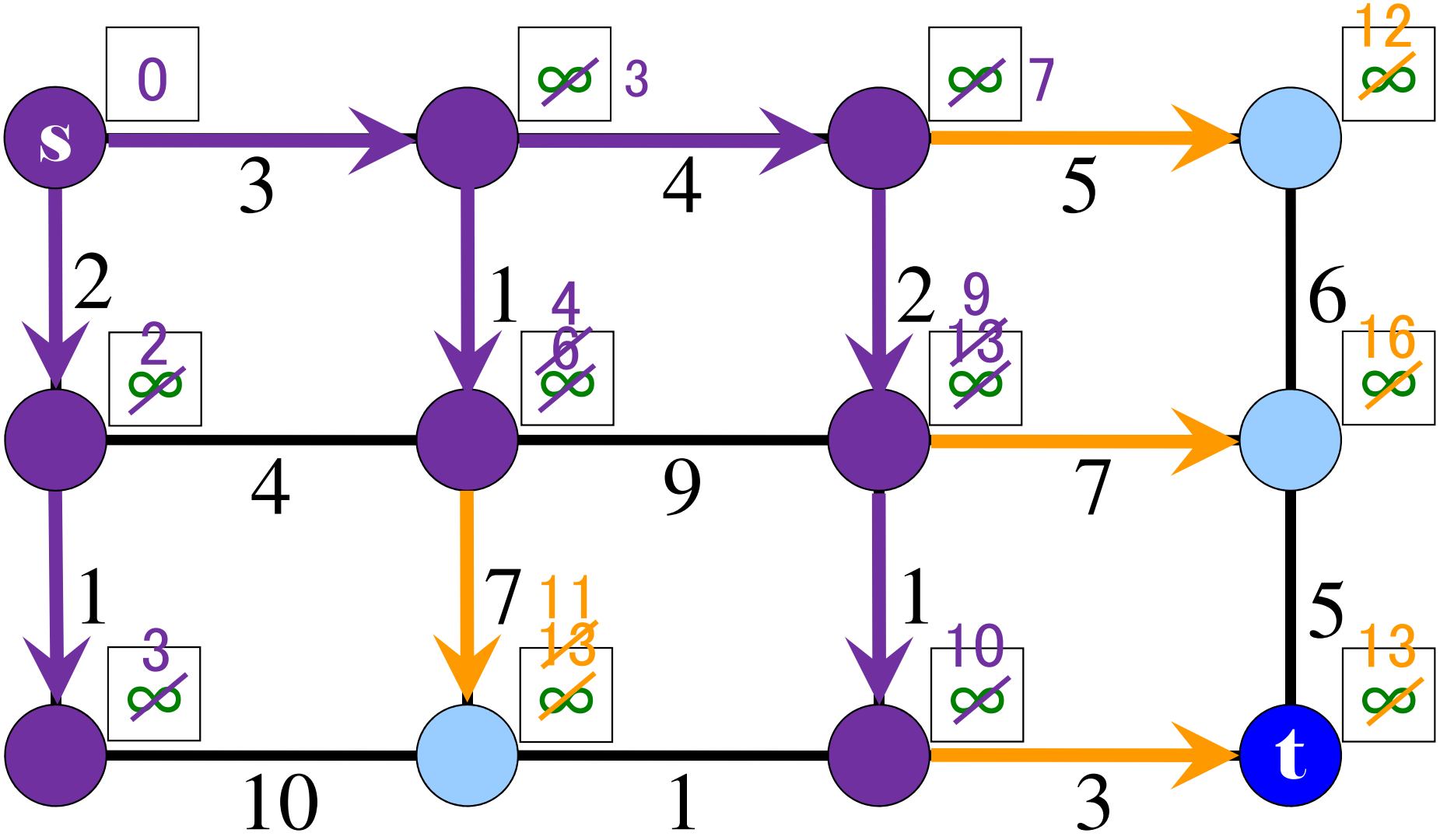


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

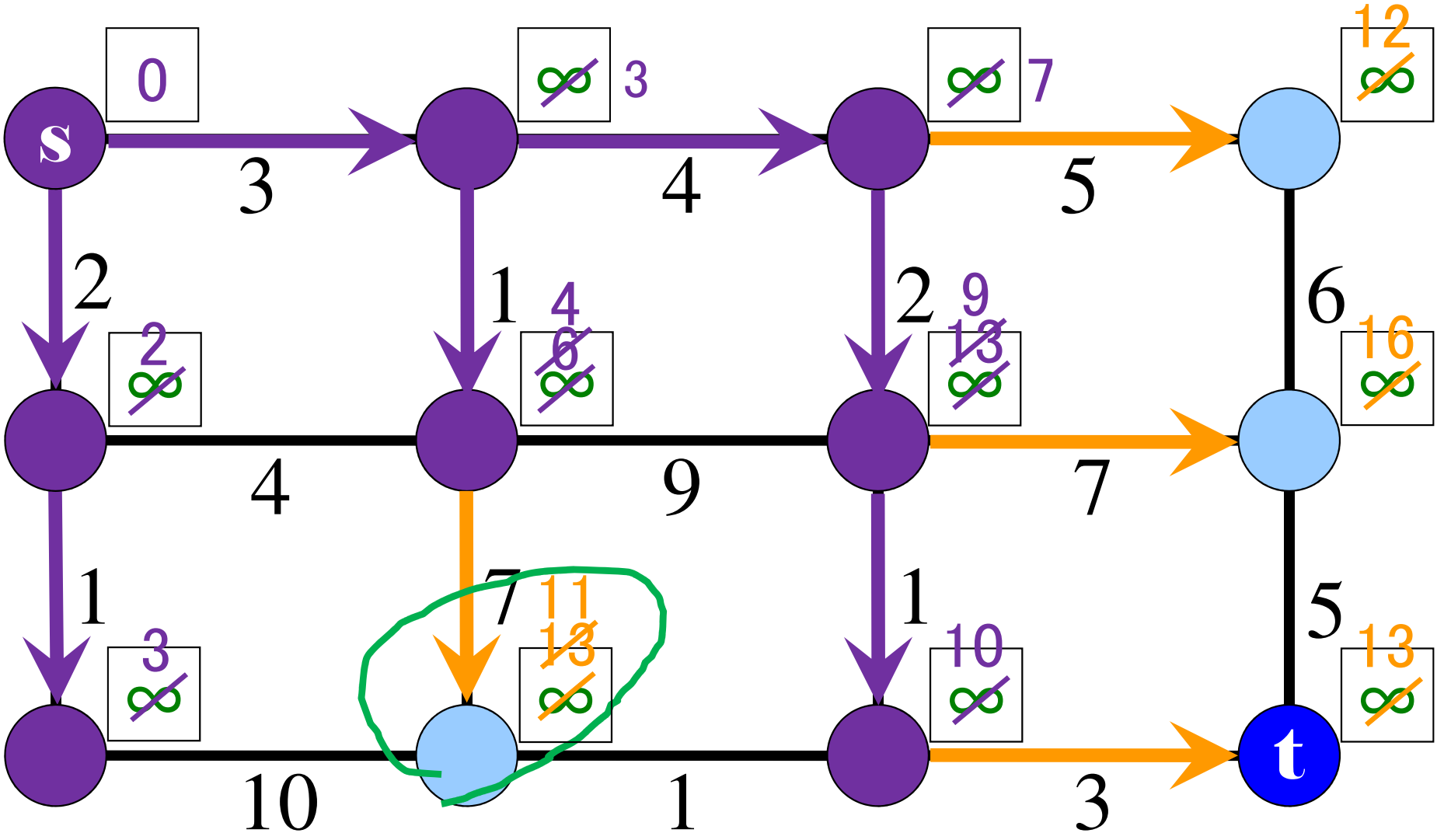
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



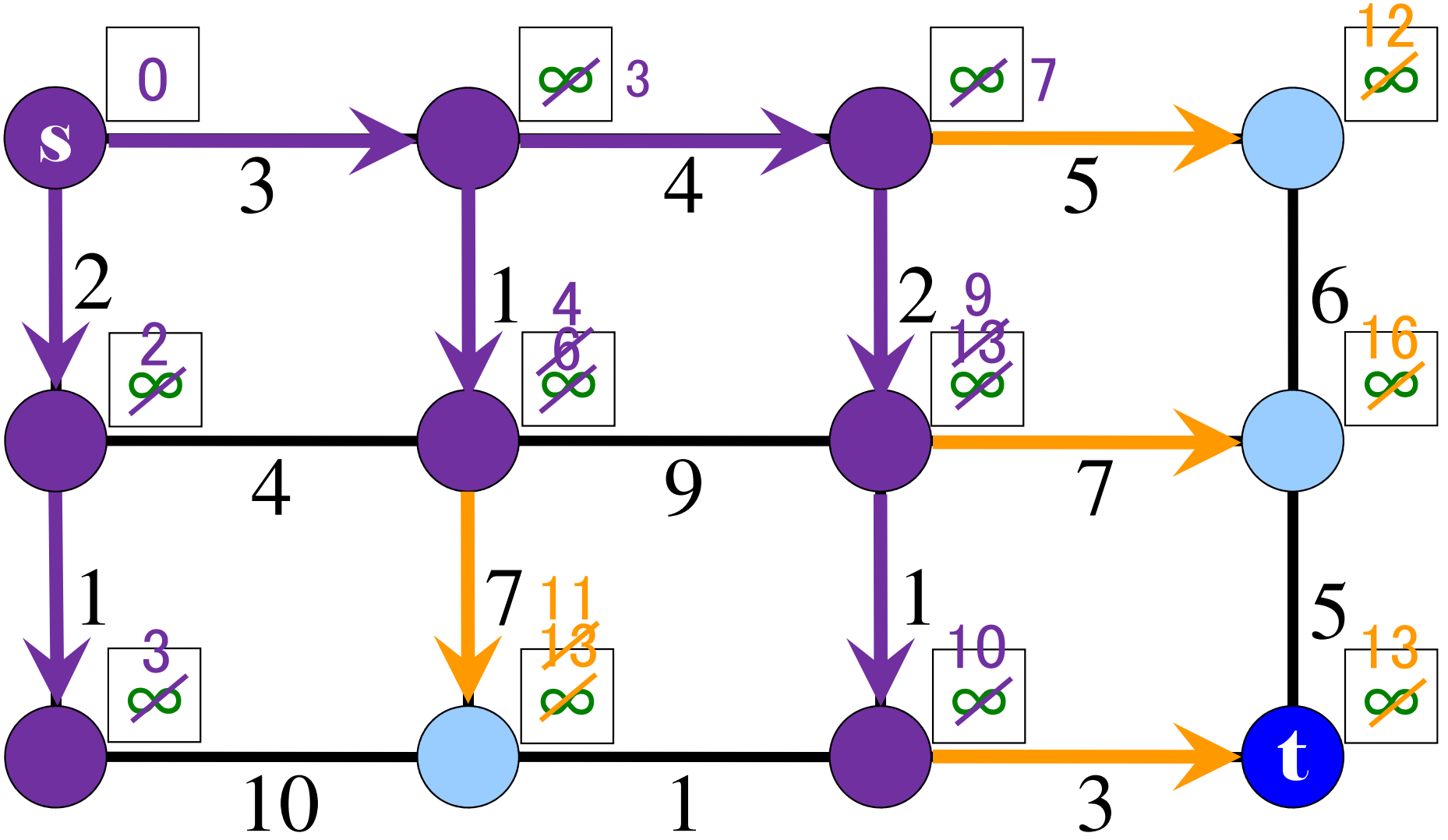
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

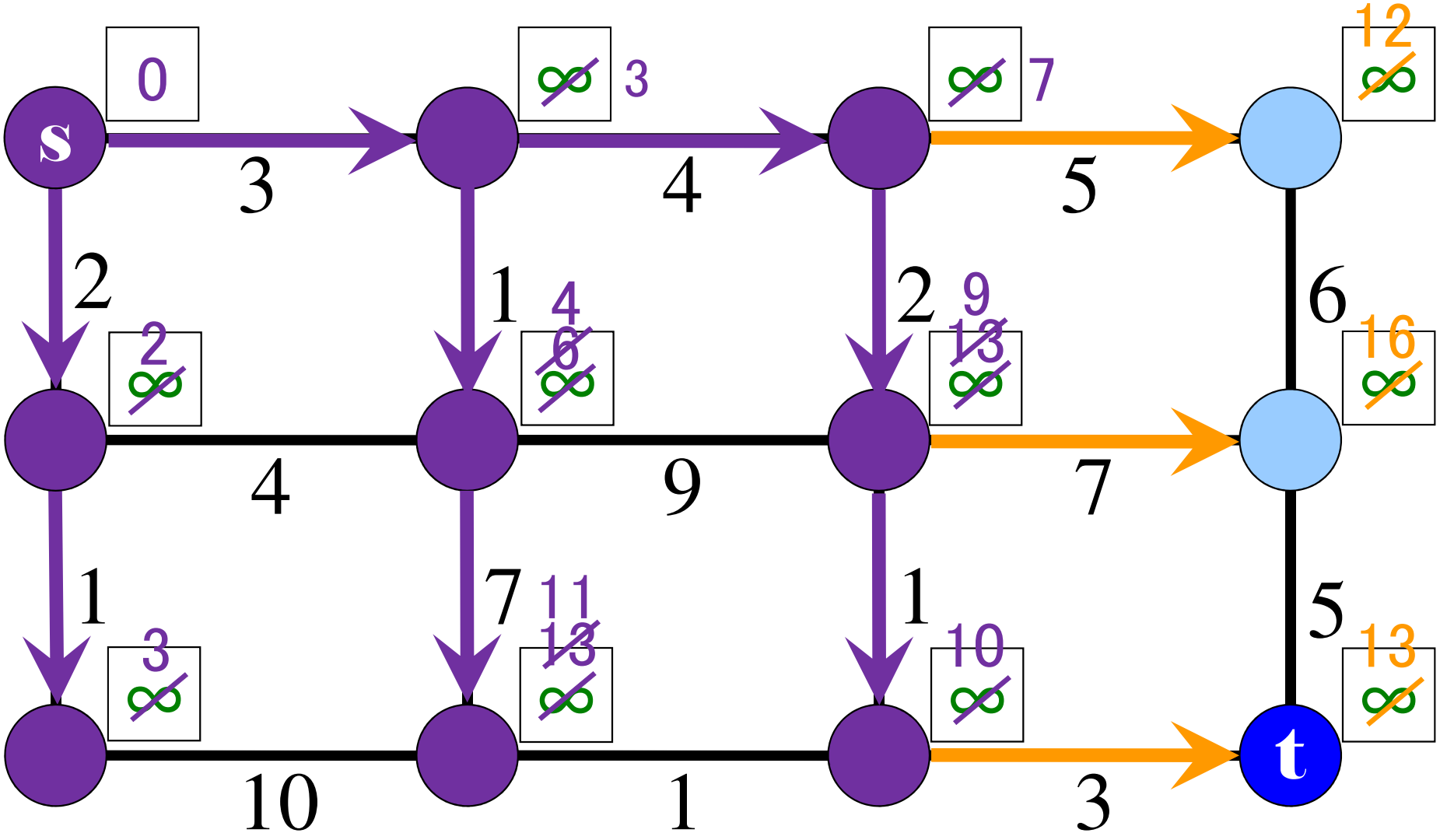


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

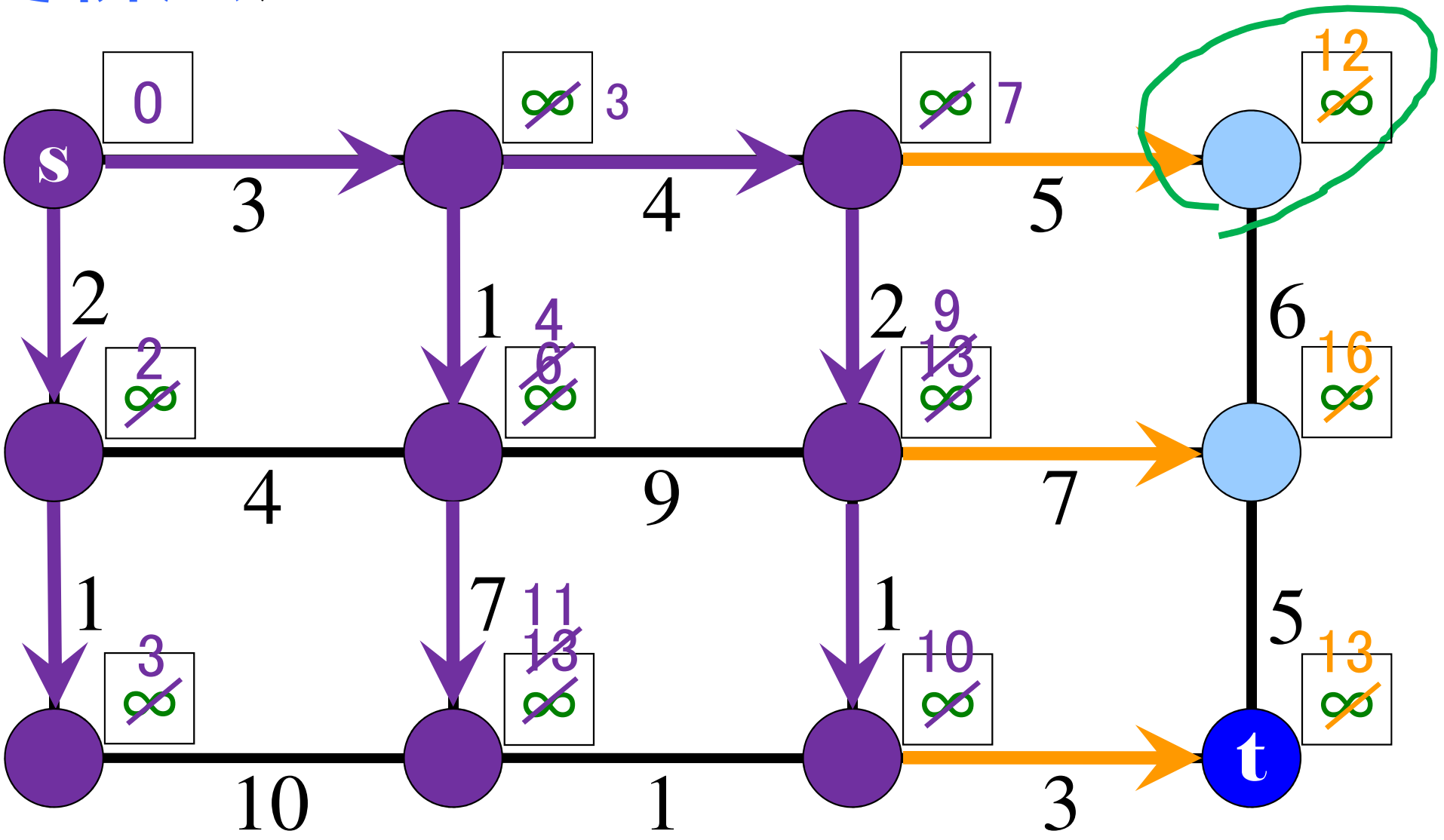
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



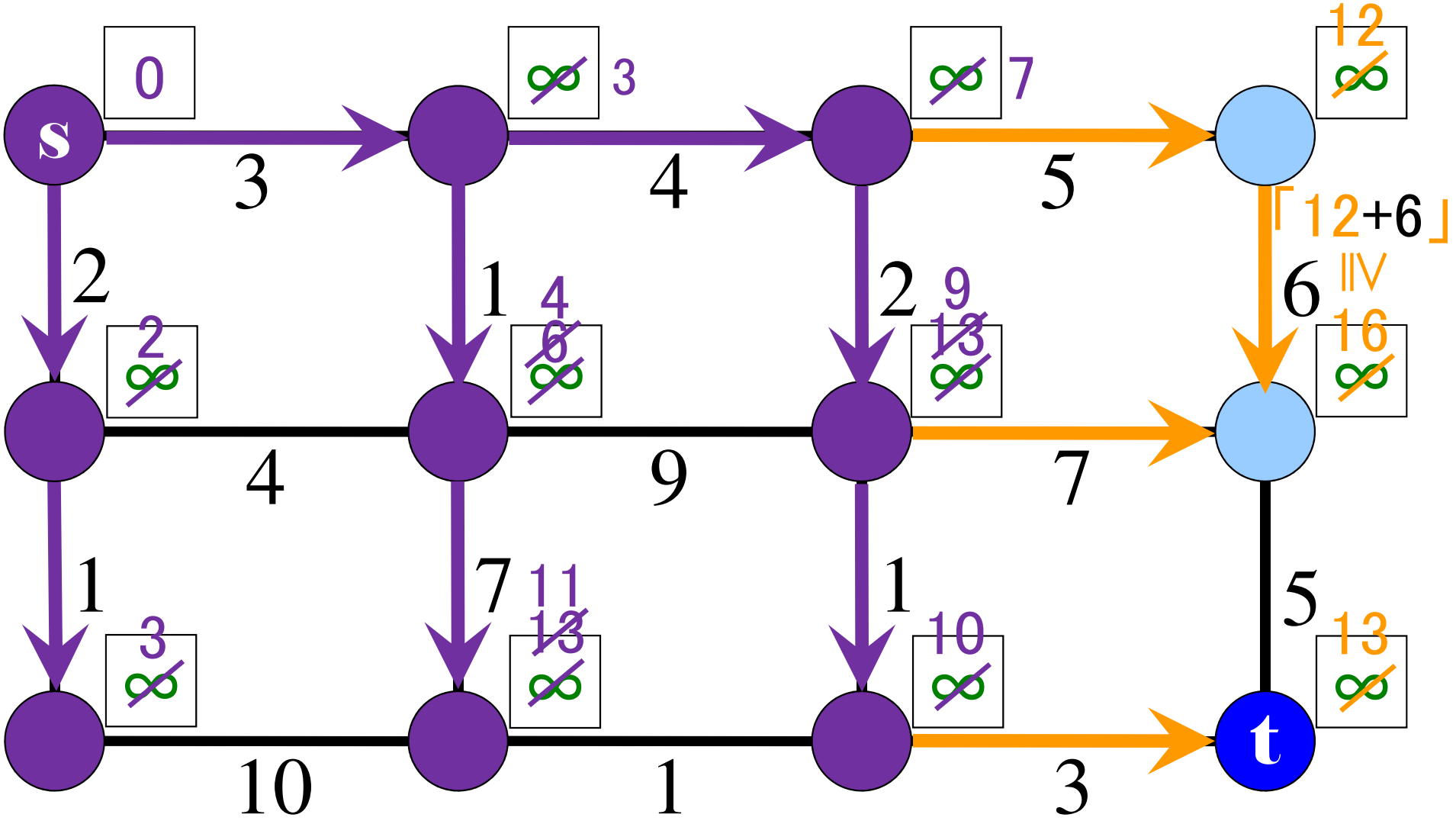
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

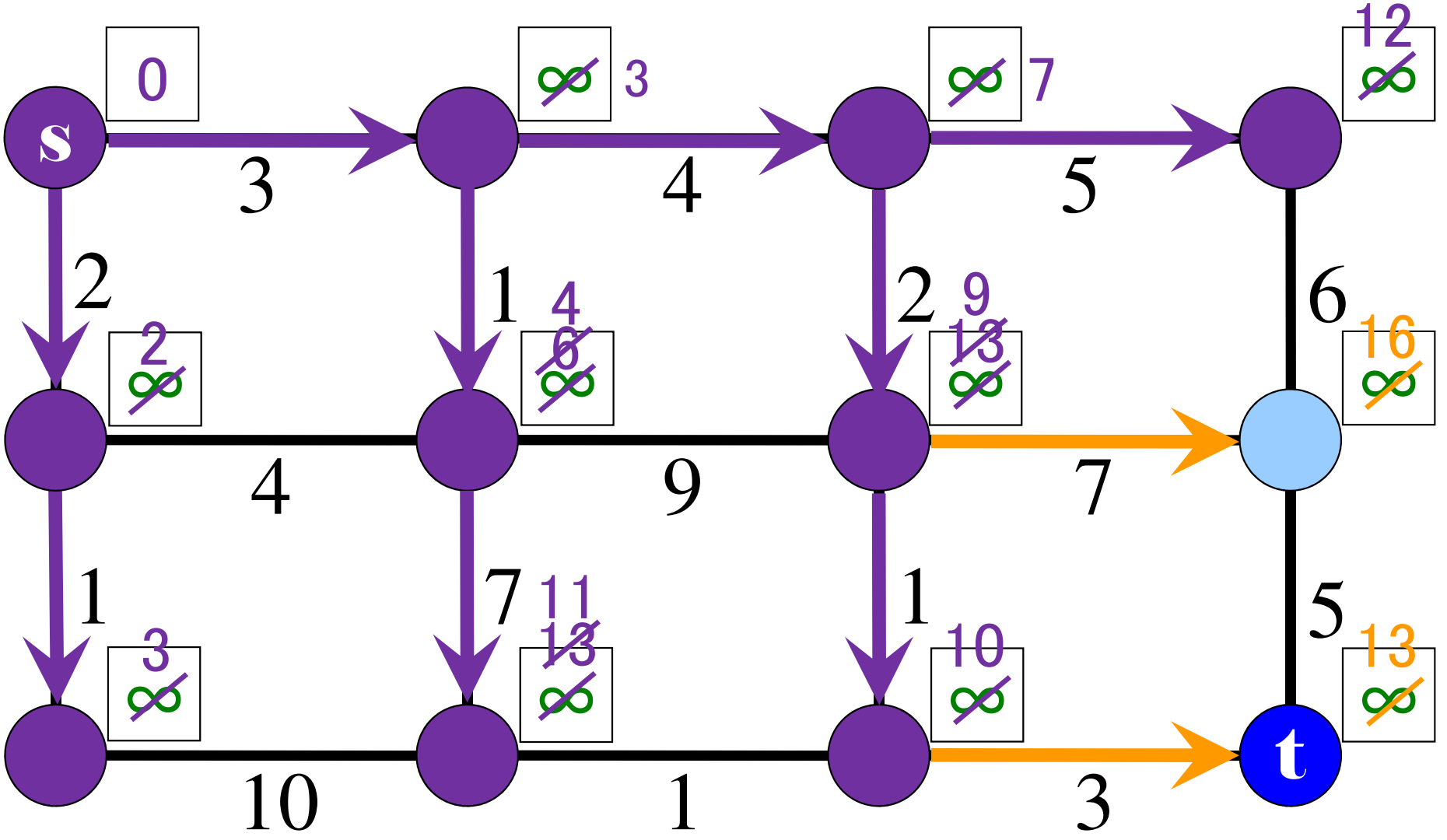


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

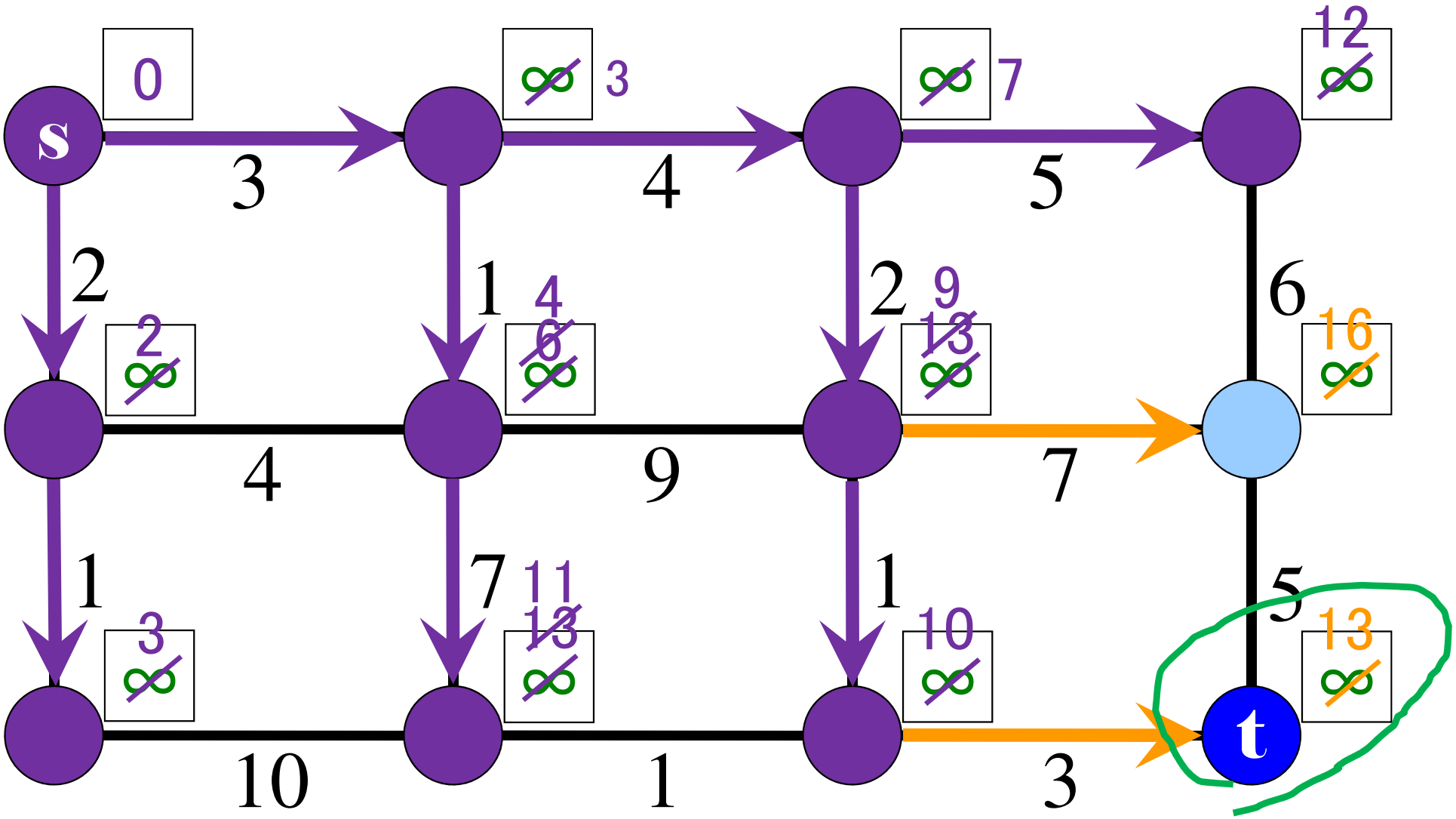
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



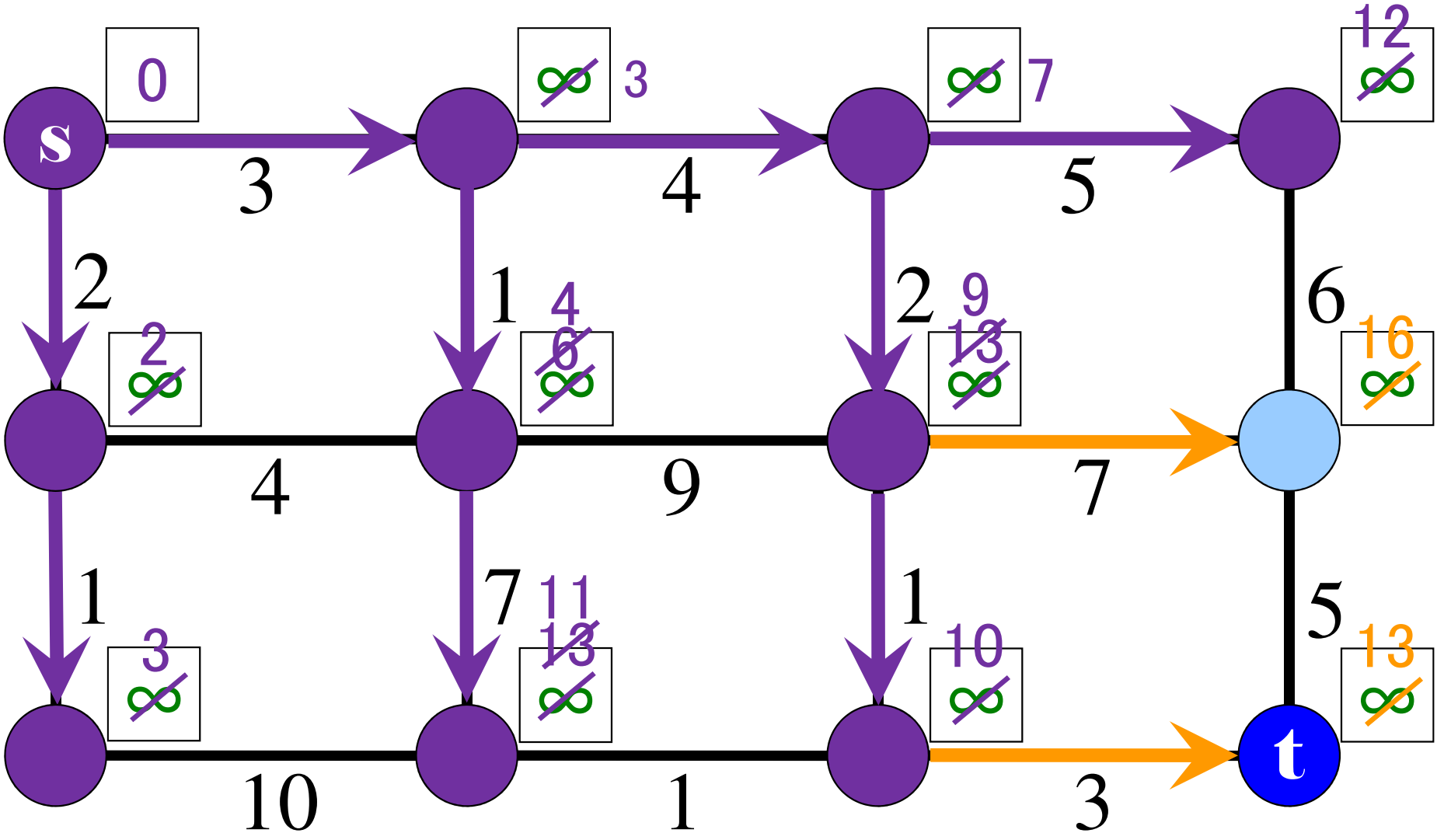
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

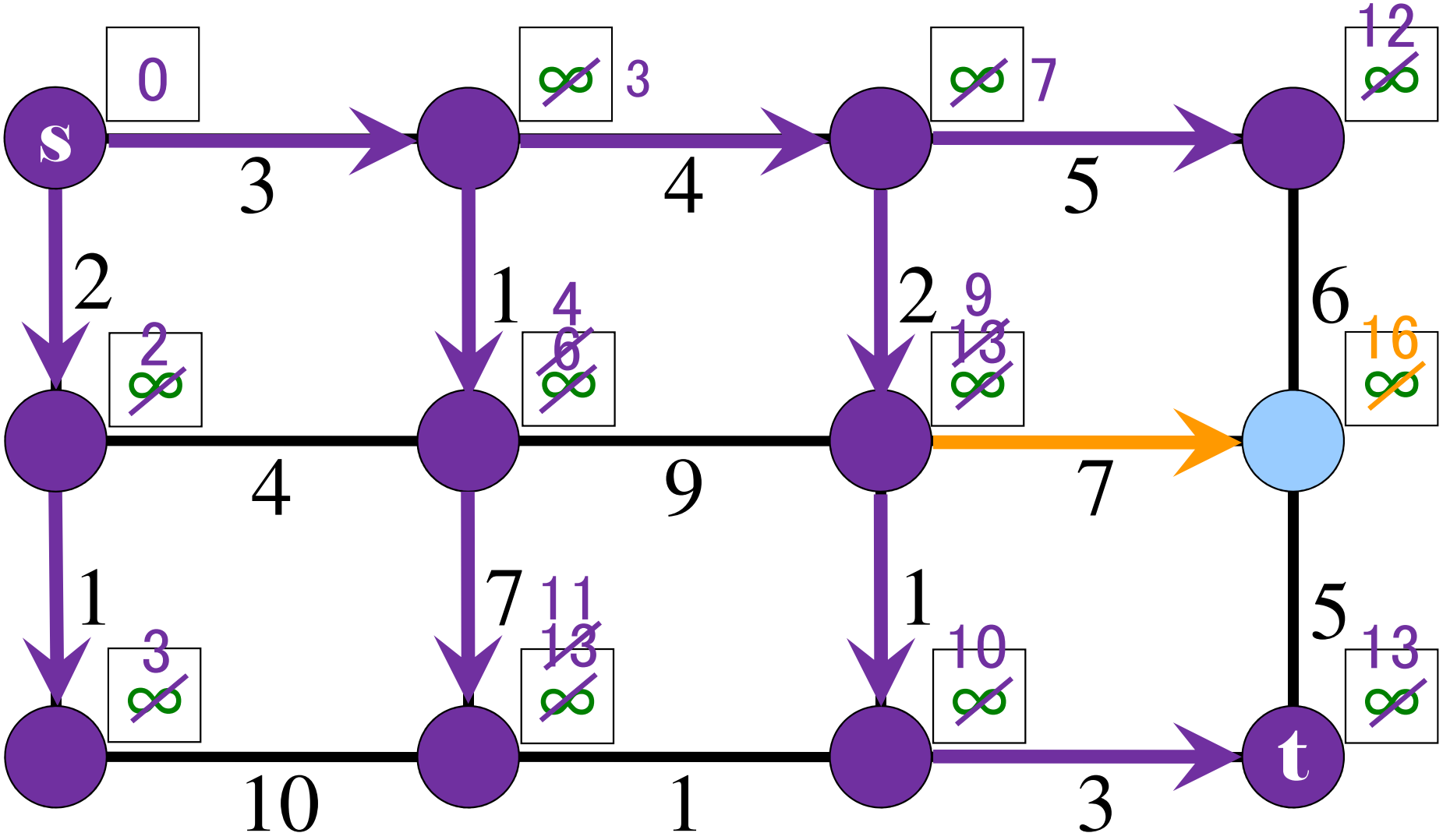


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

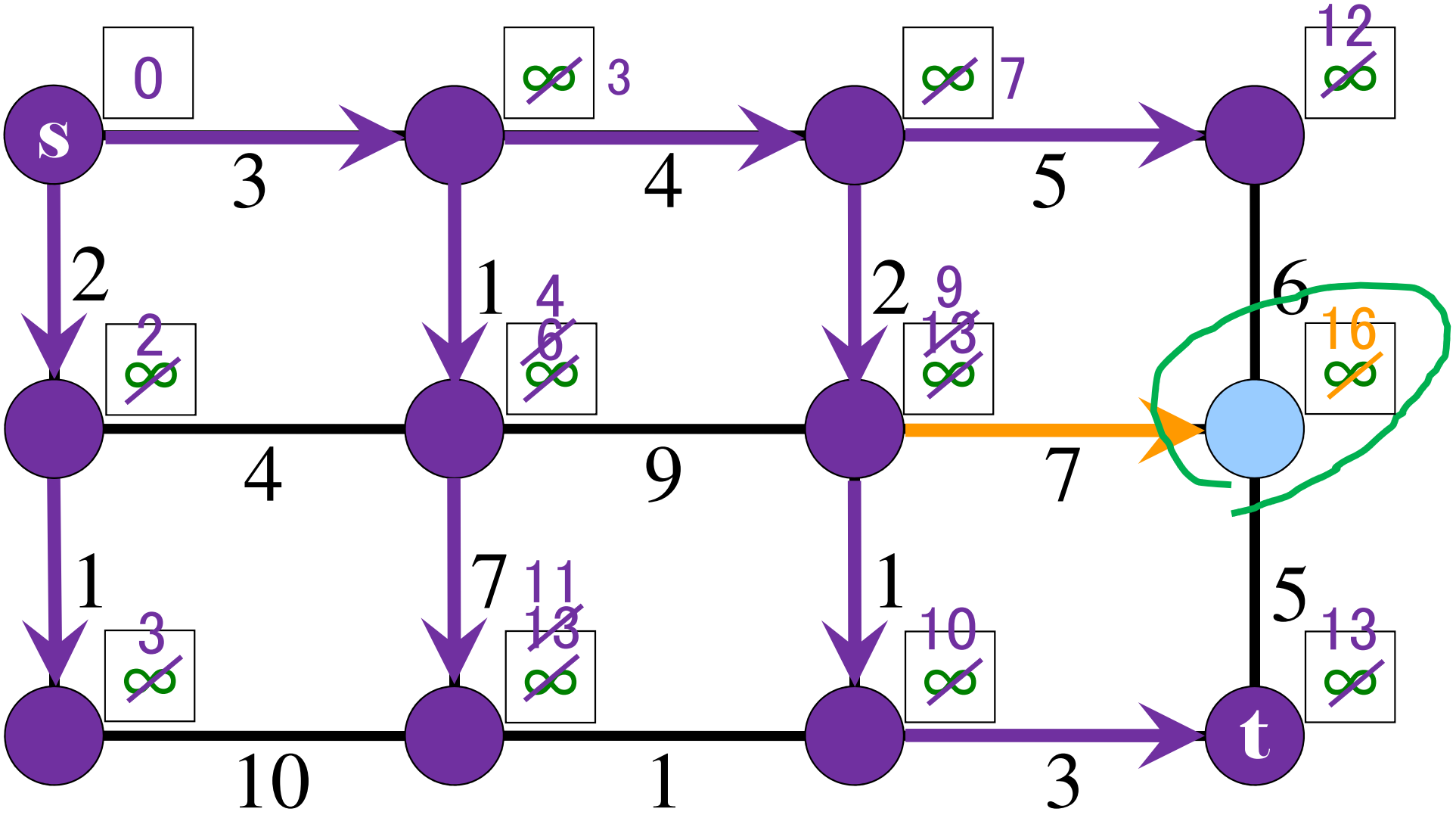
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



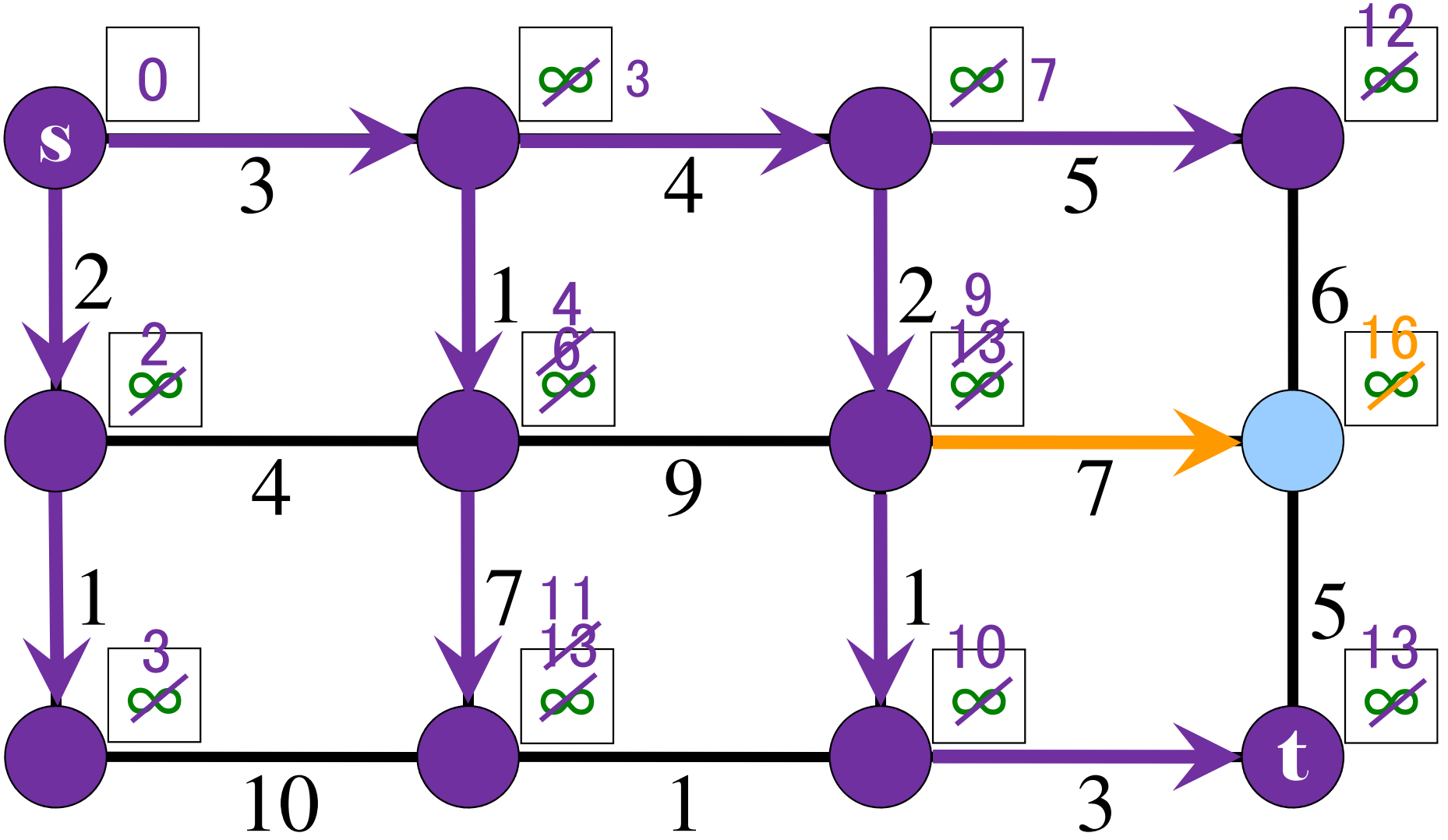
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

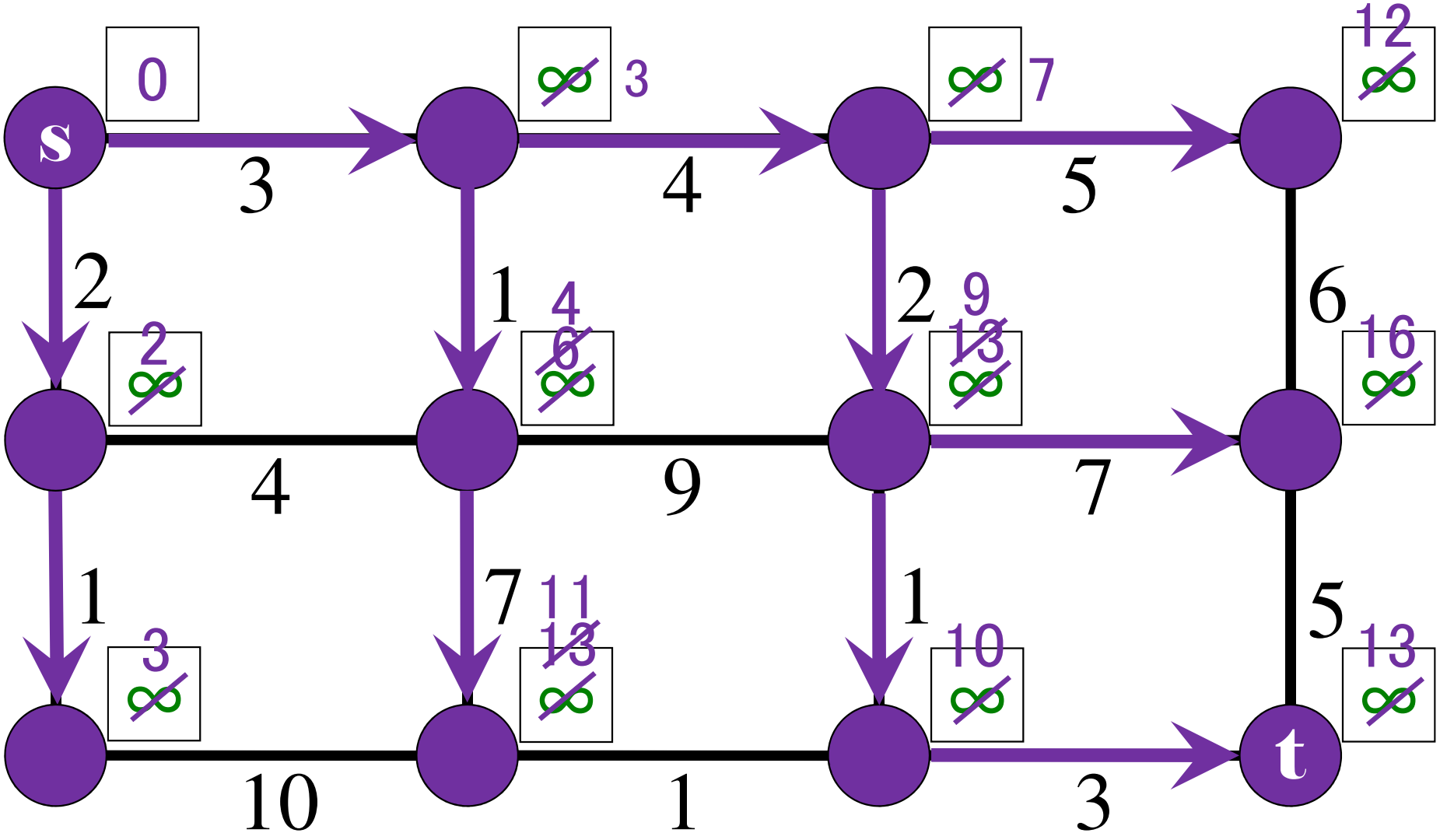


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

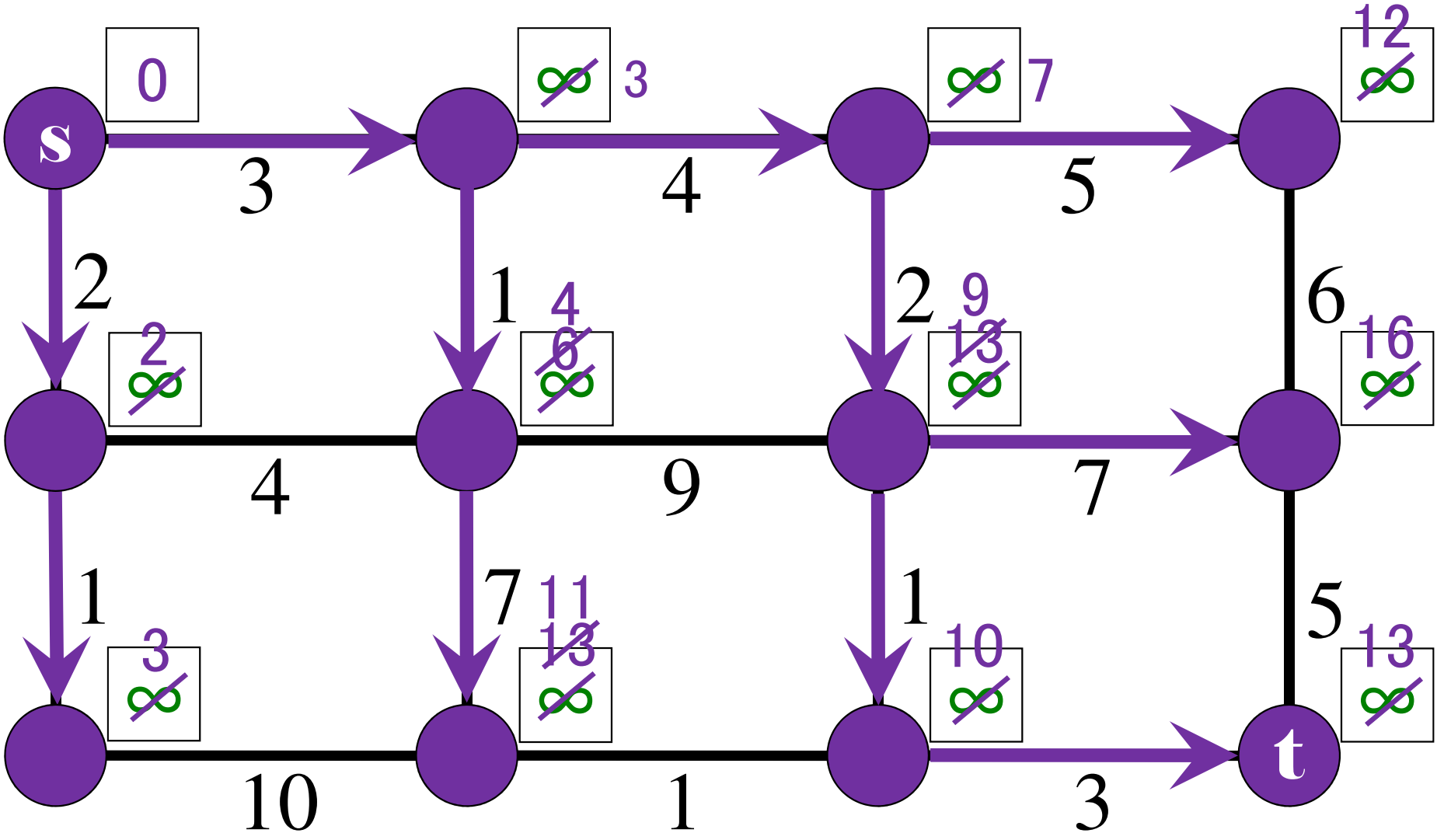
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



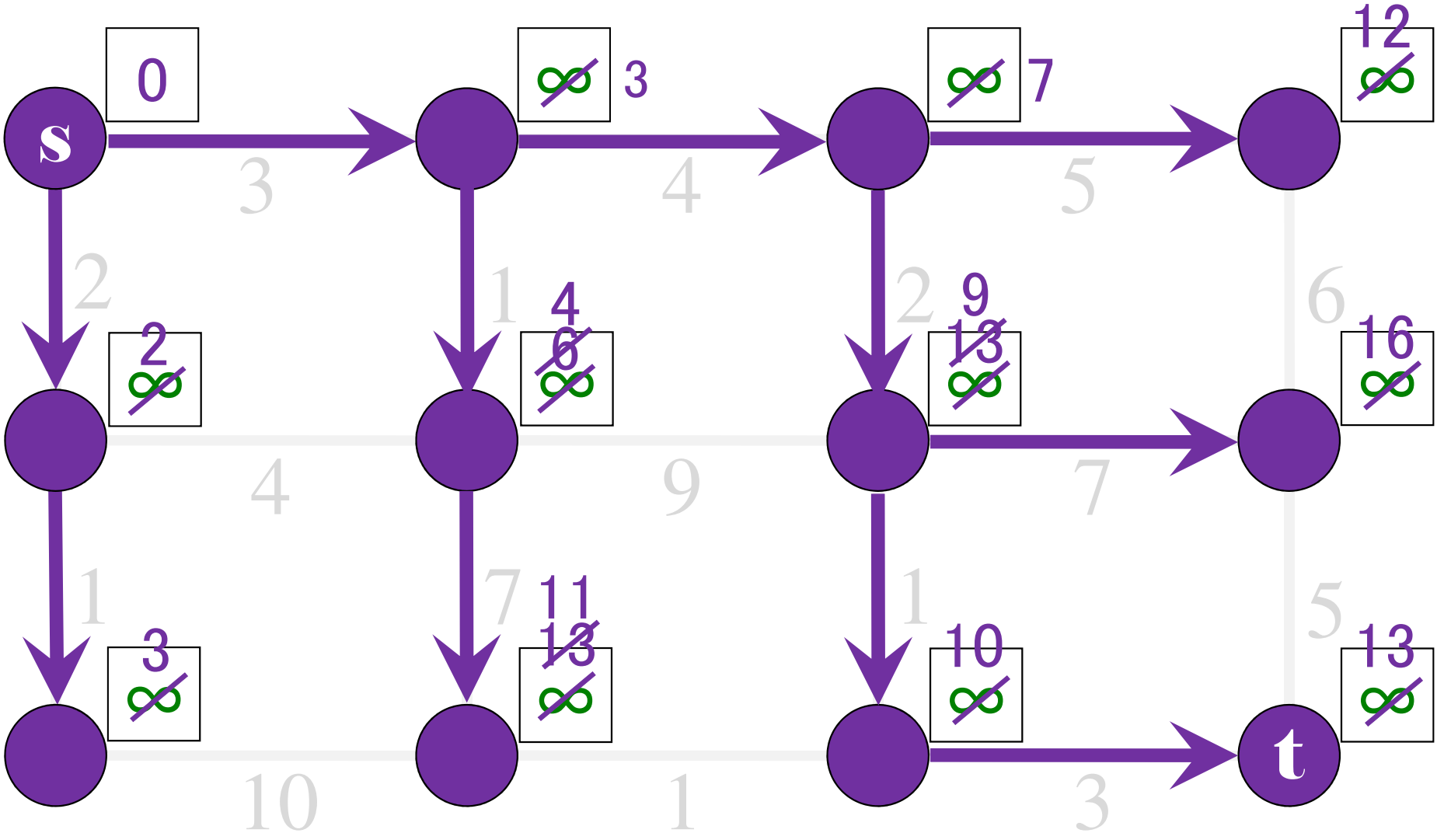
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



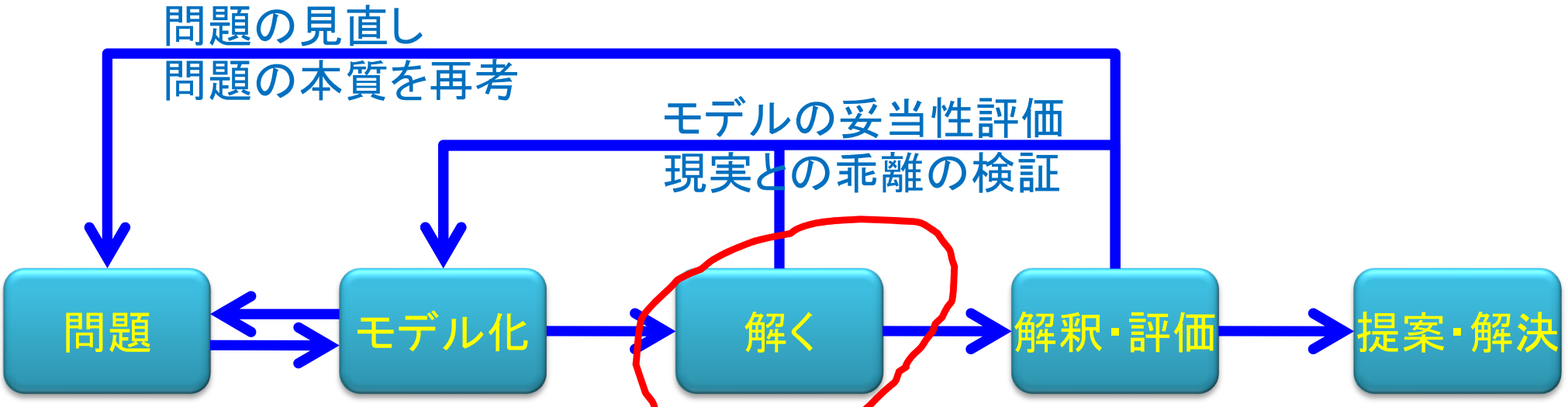
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

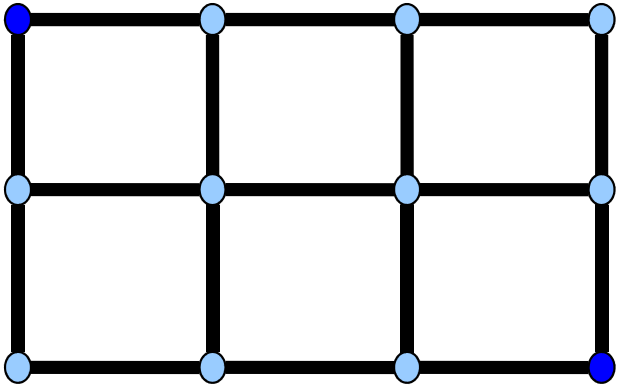
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

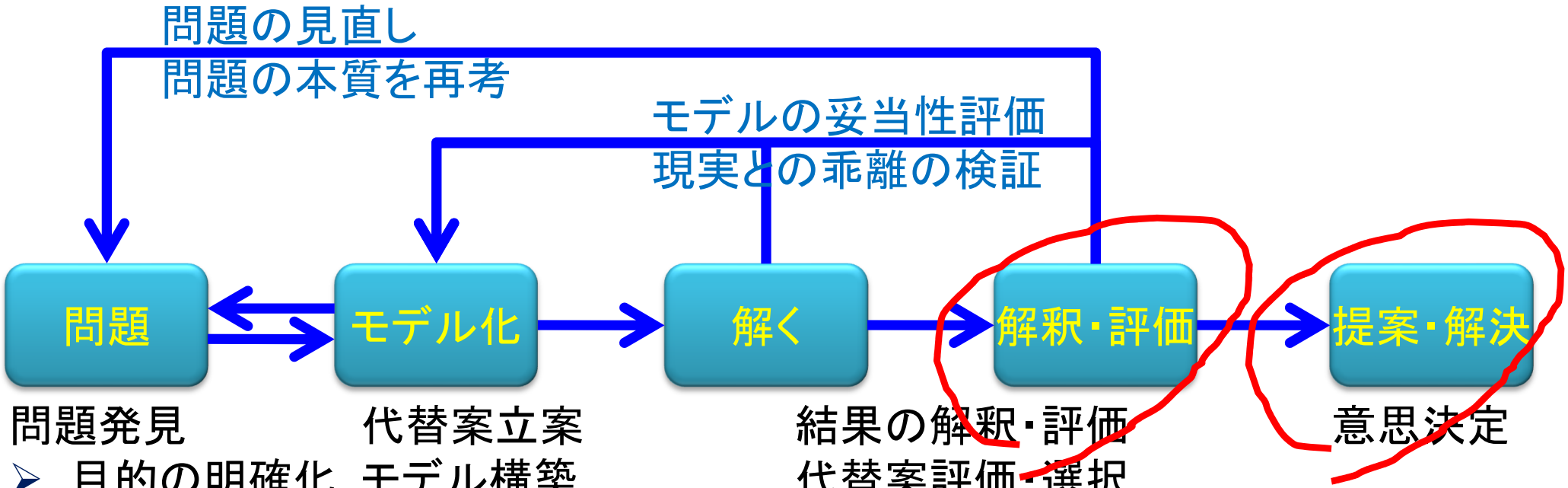
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

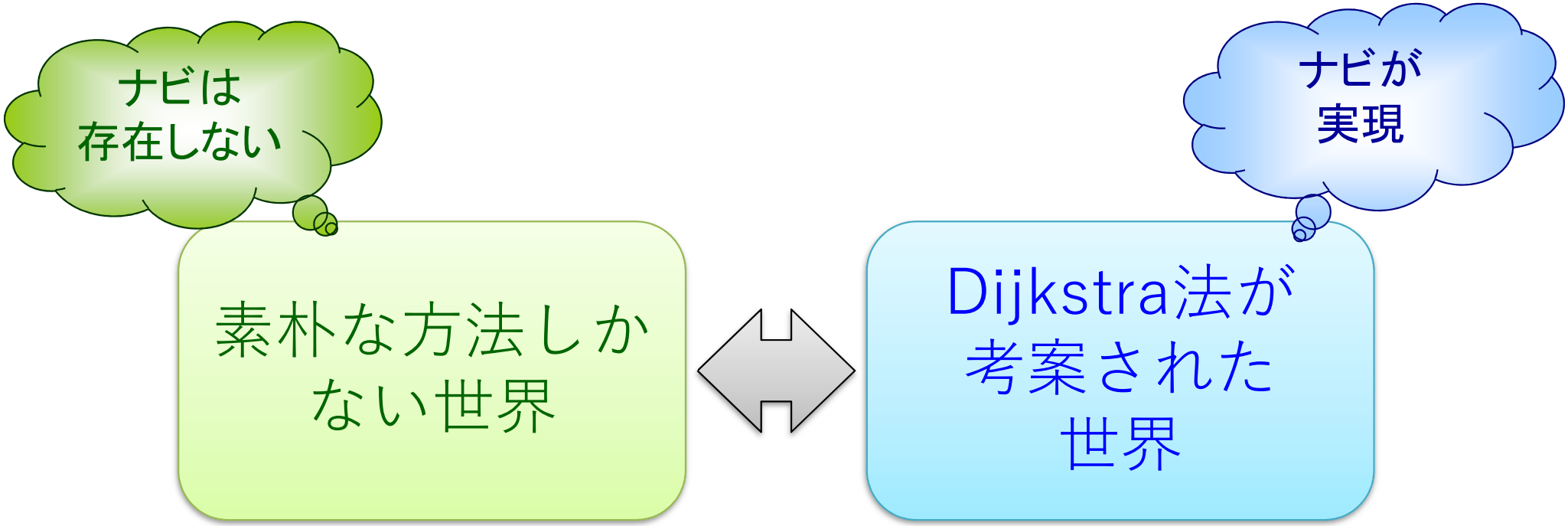
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

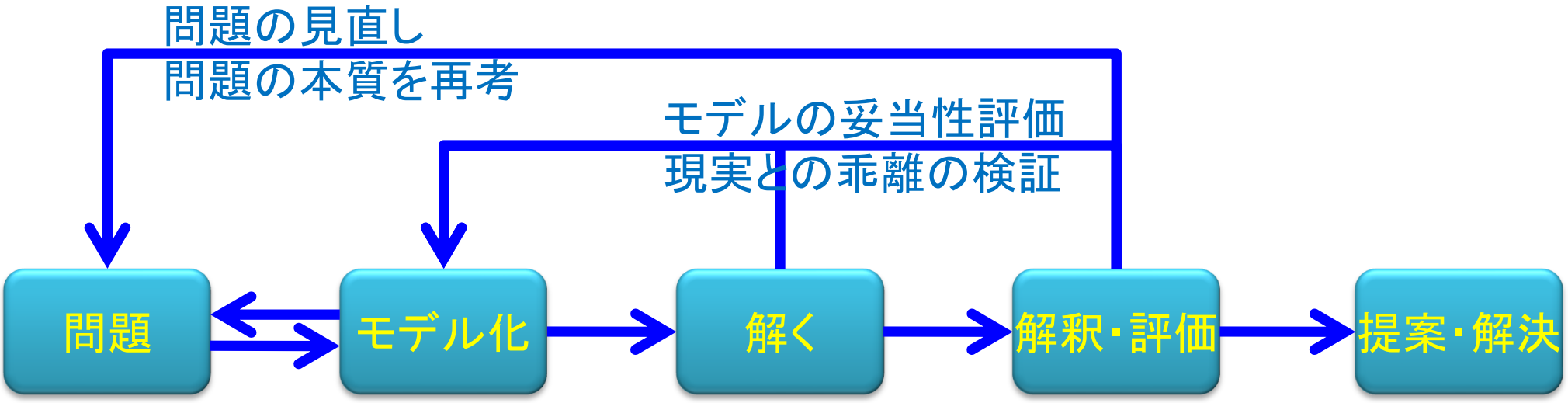
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



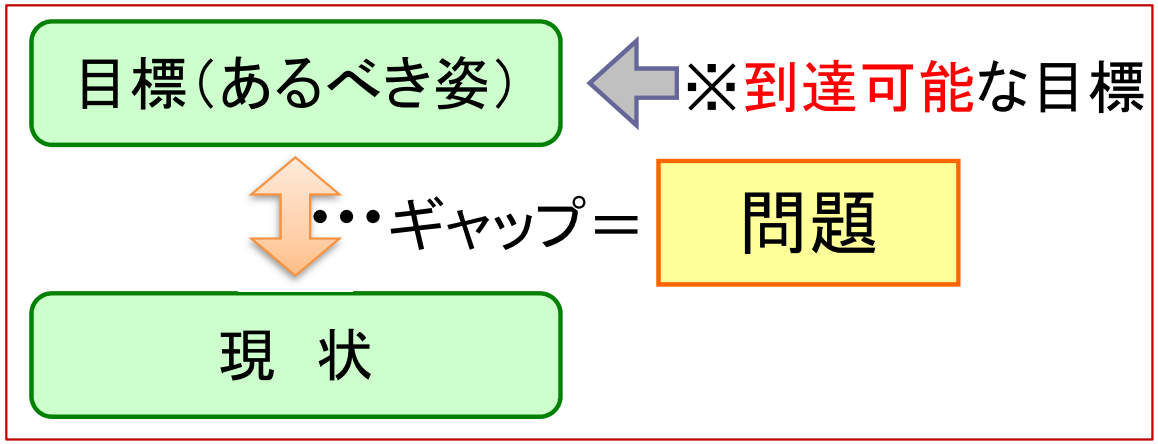
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

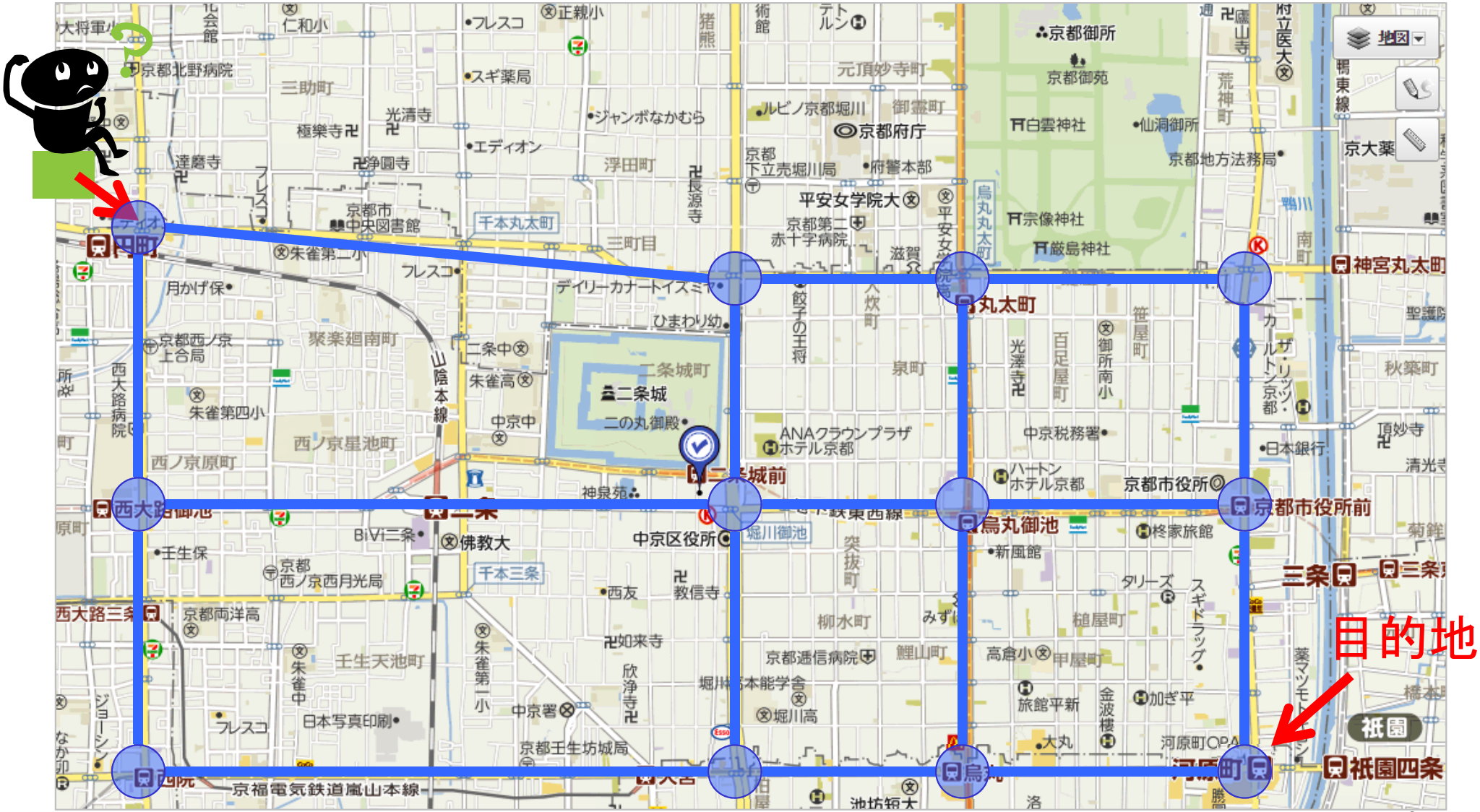
意思決定

問題の定義



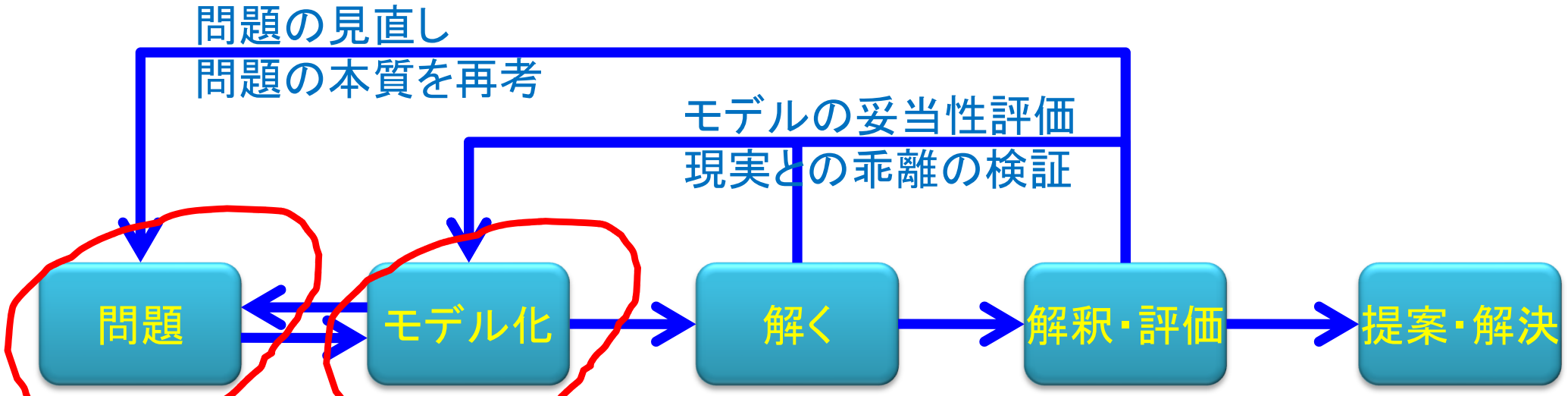
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

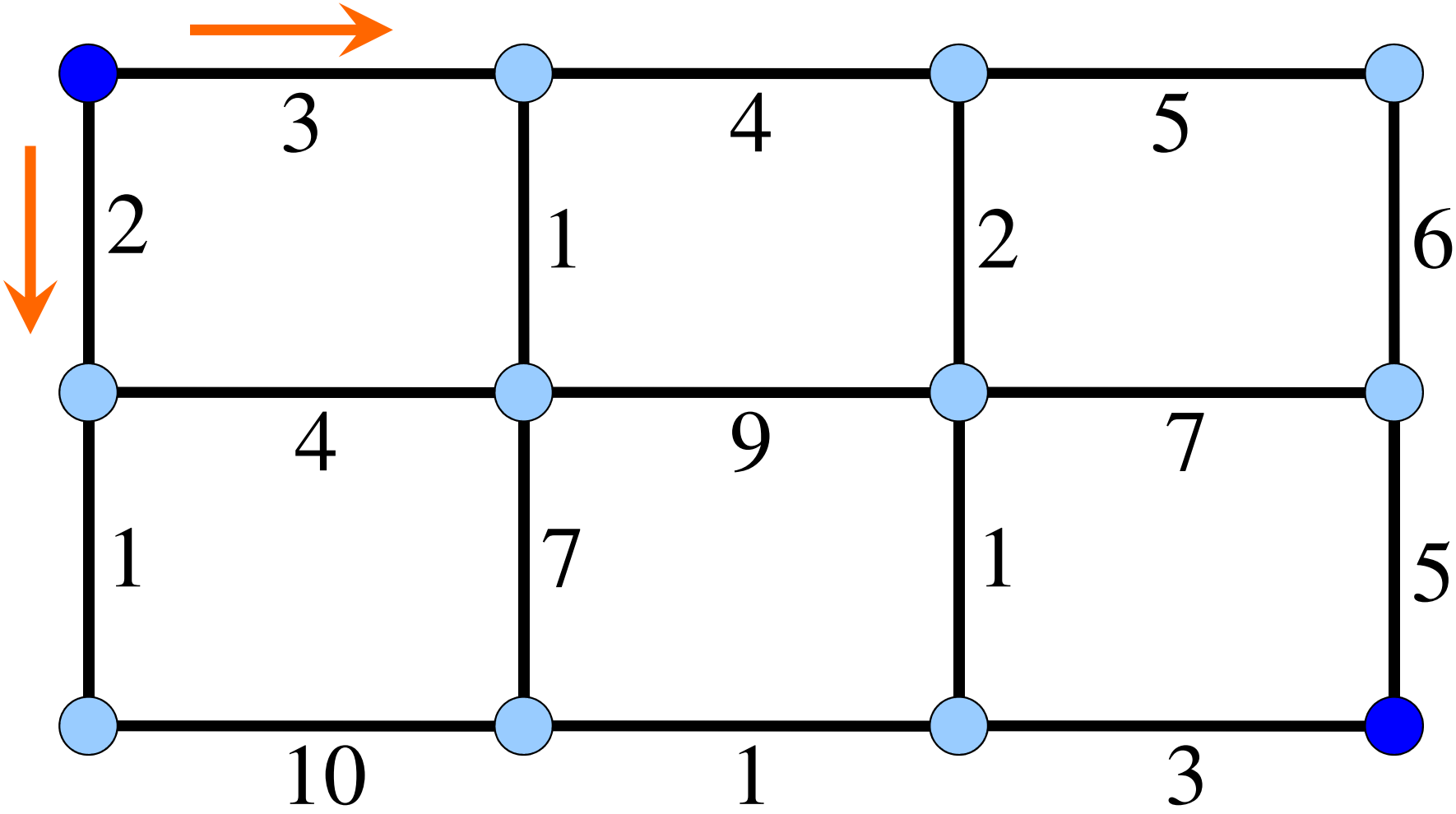
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

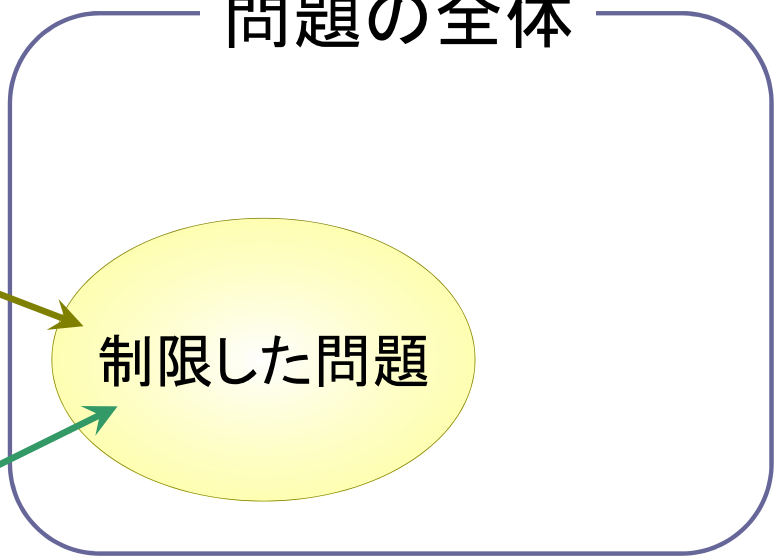


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

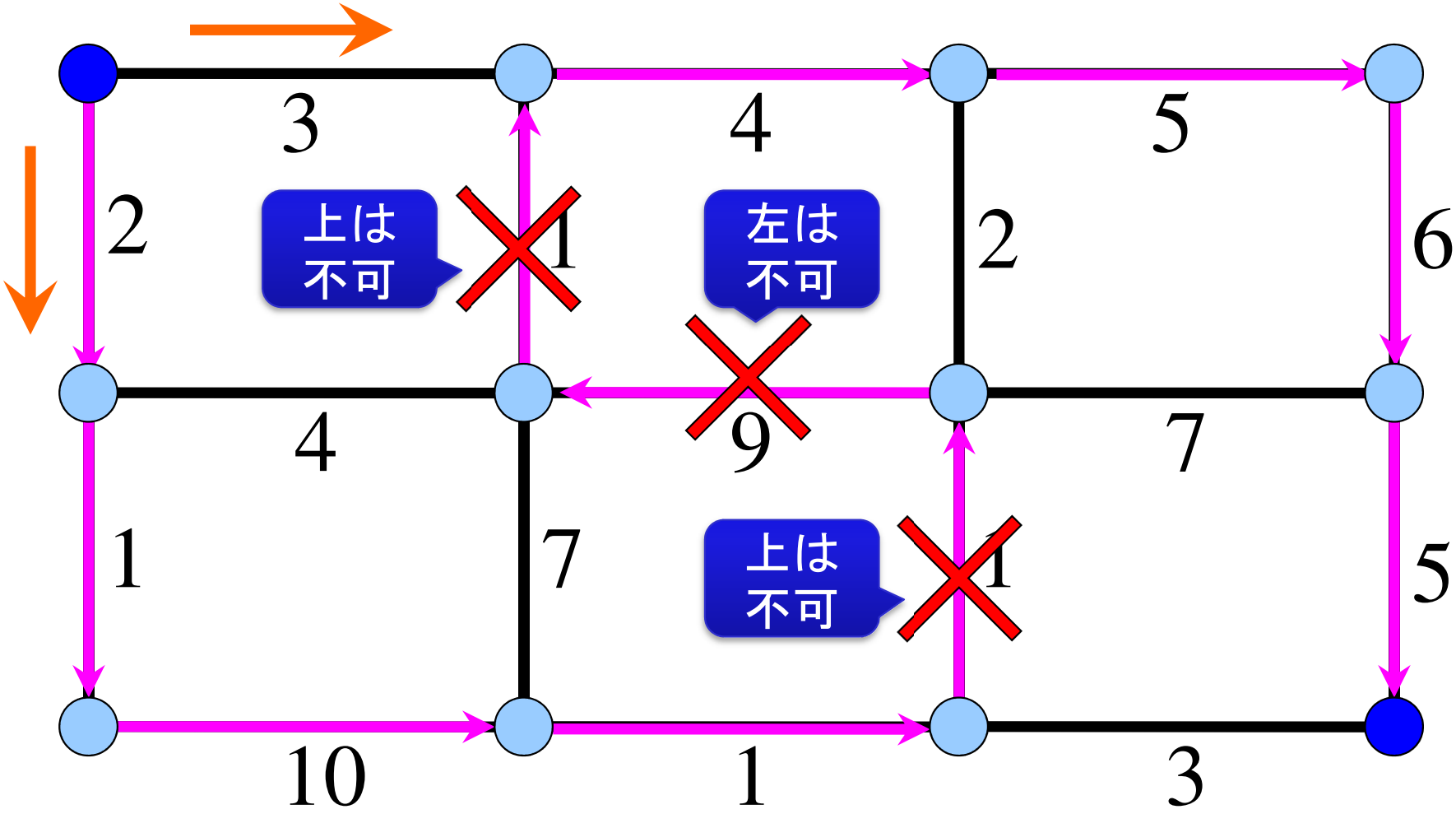
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

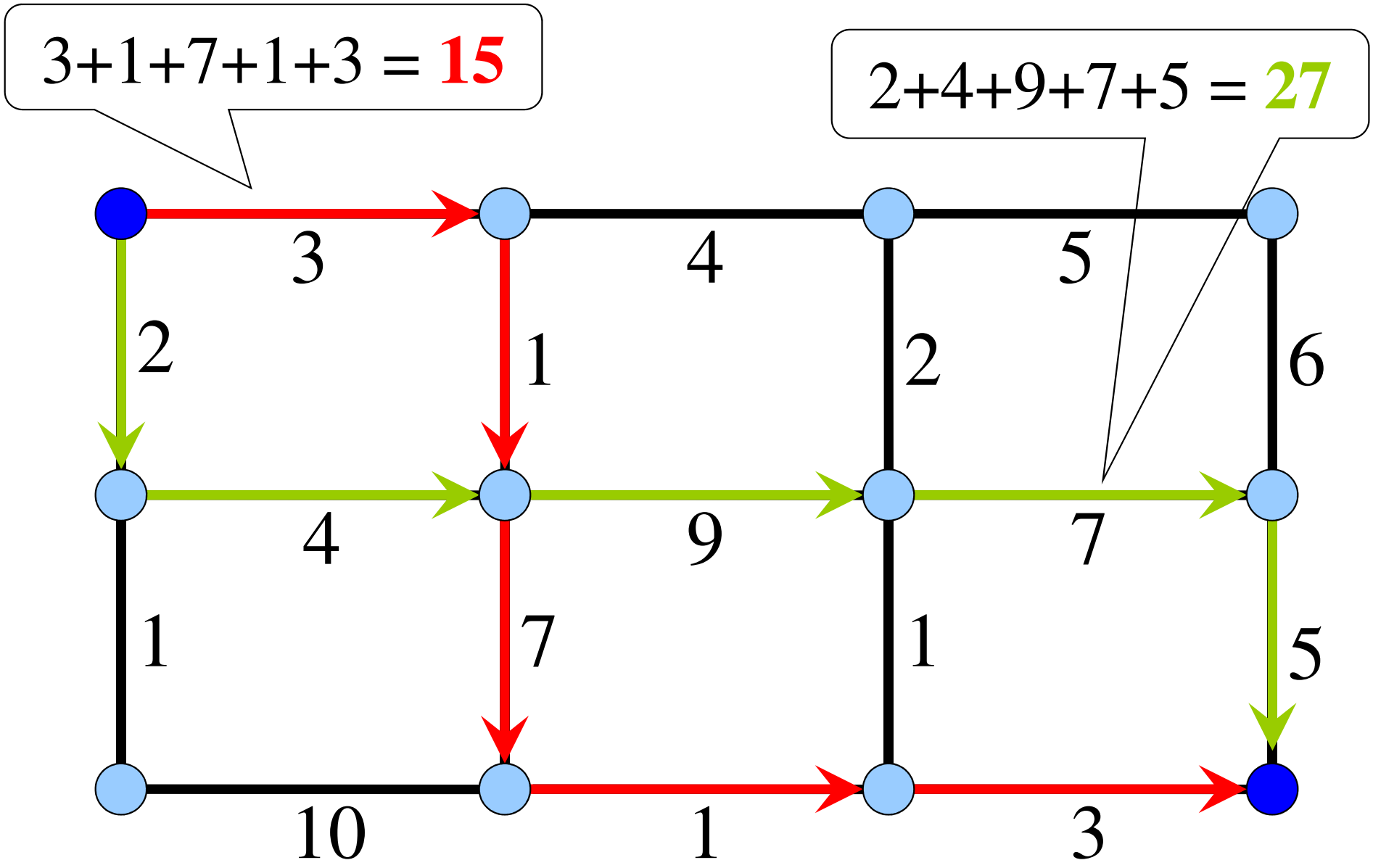
難しいなら易しくすればいいのさ！

制限した問題

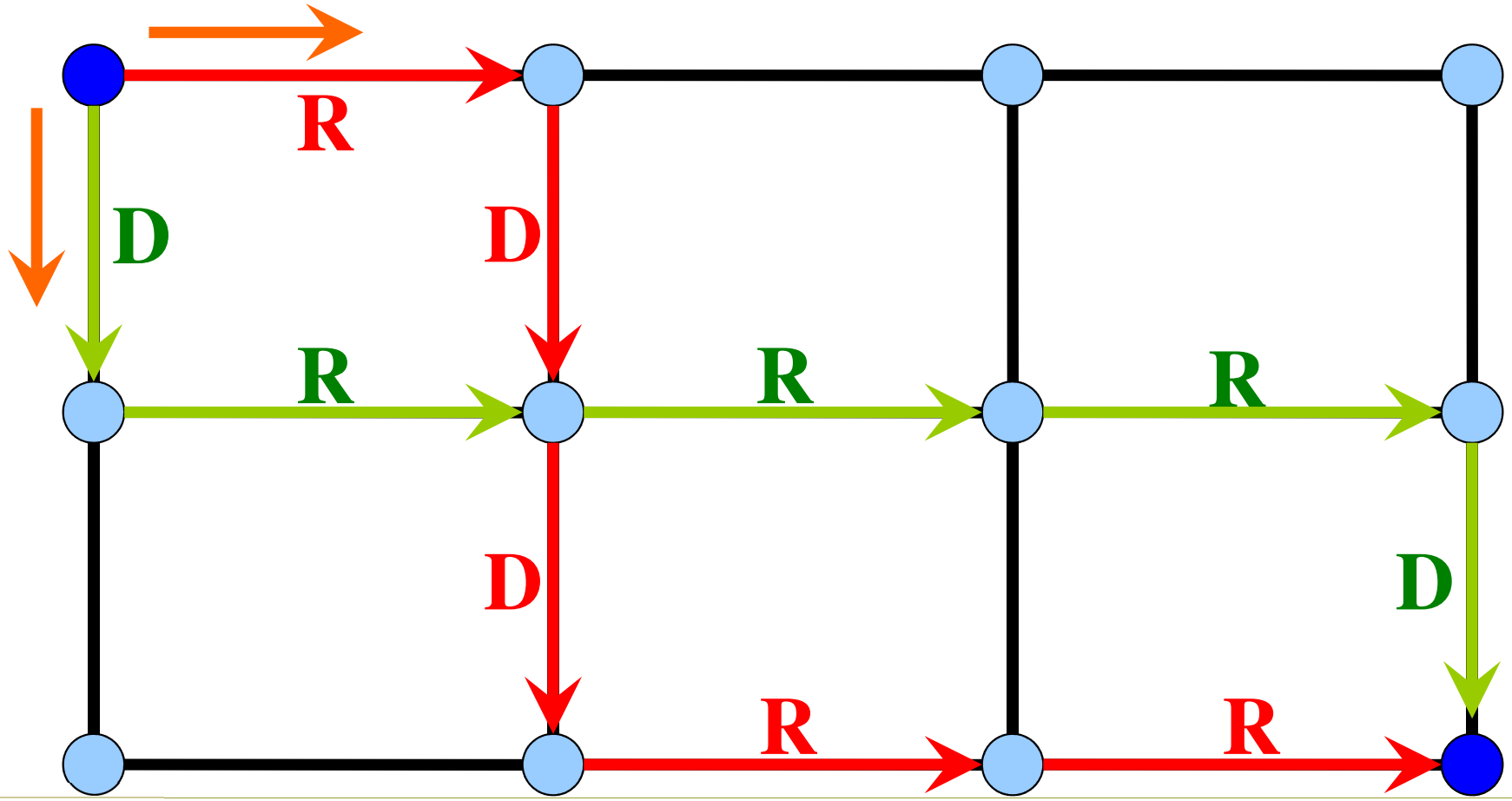
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



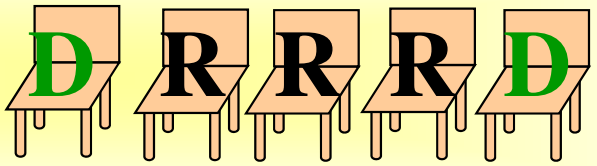
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



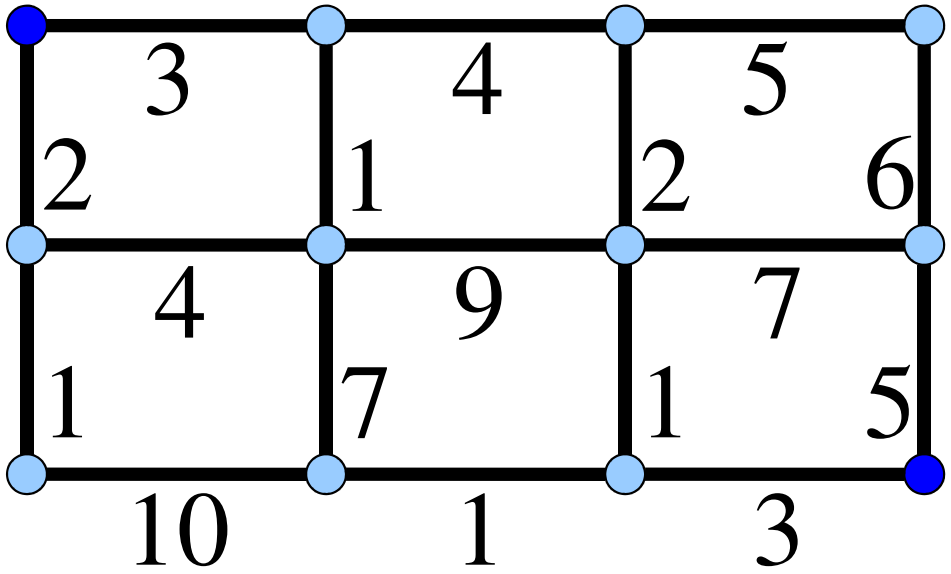
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

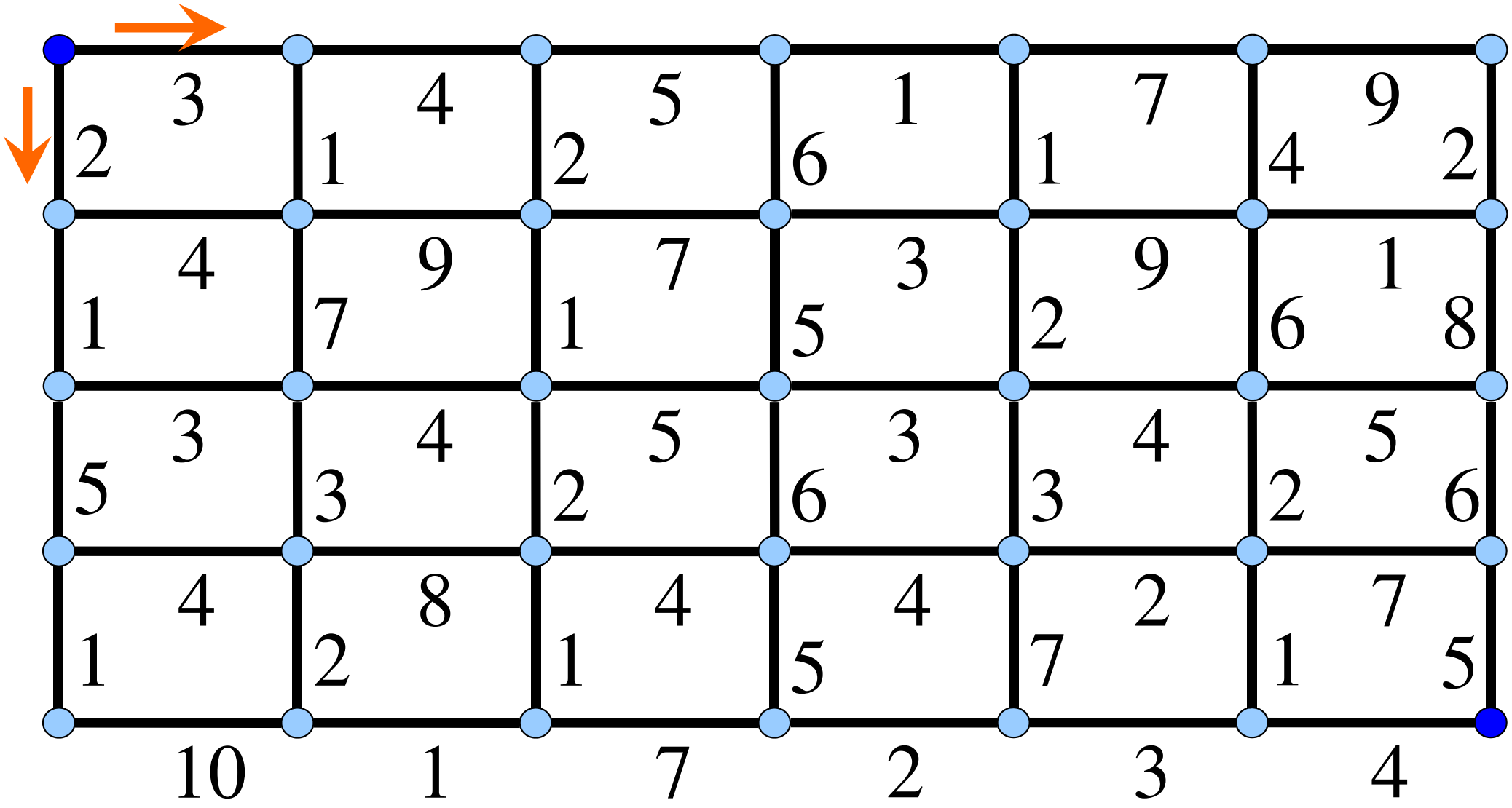
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

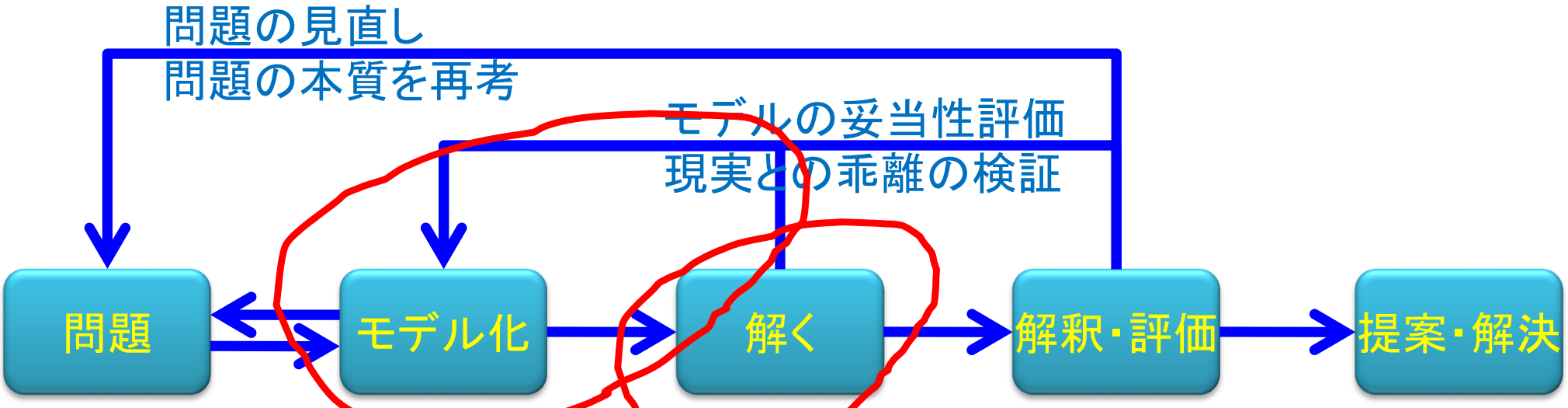
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

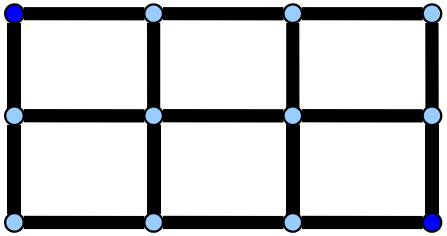
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

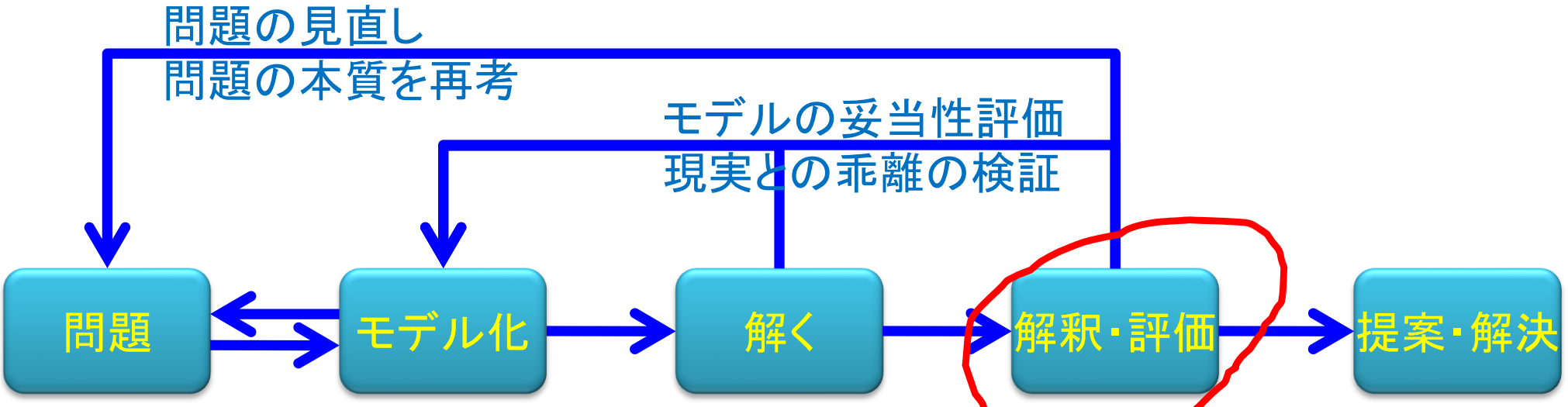
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

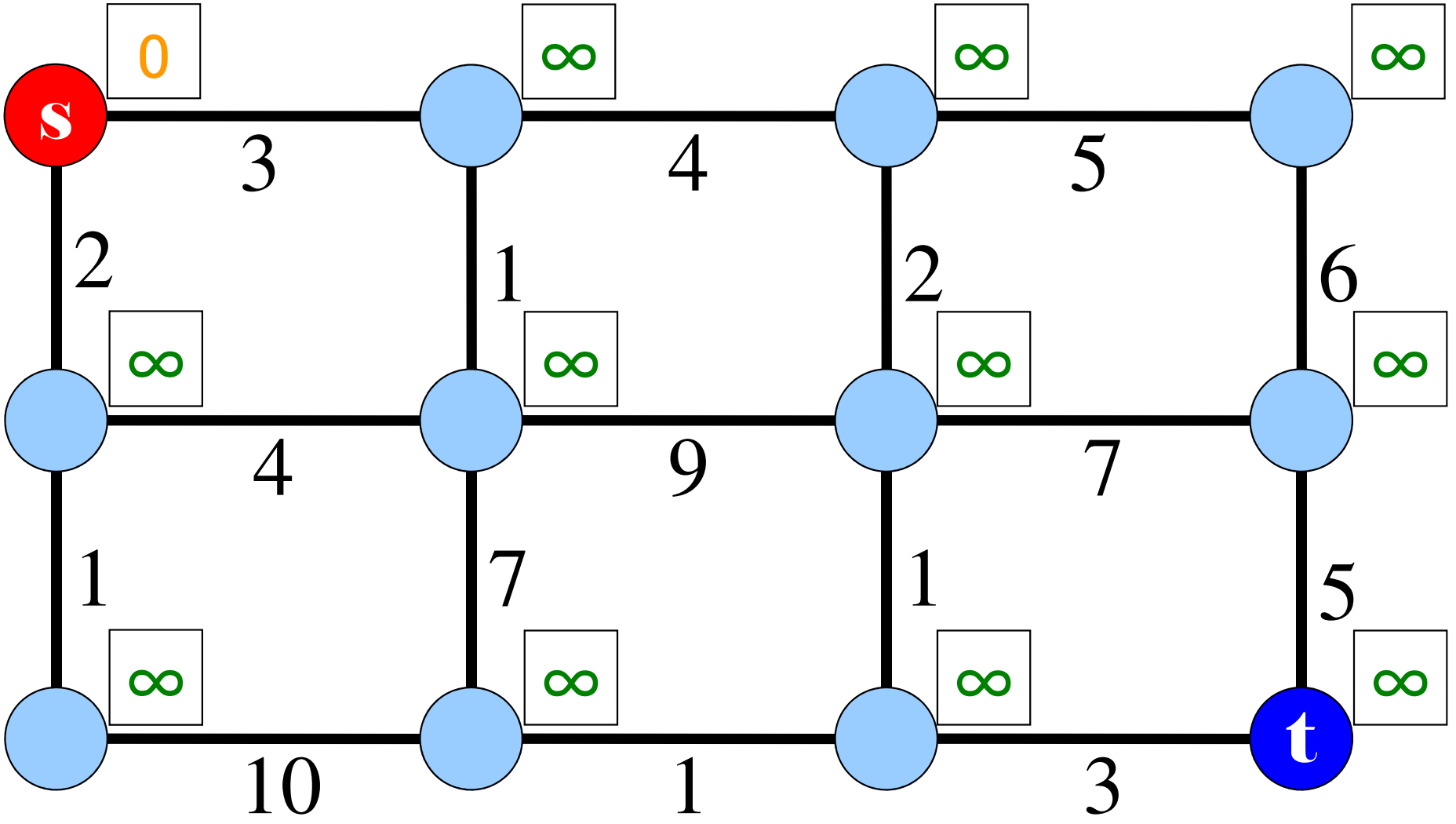
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

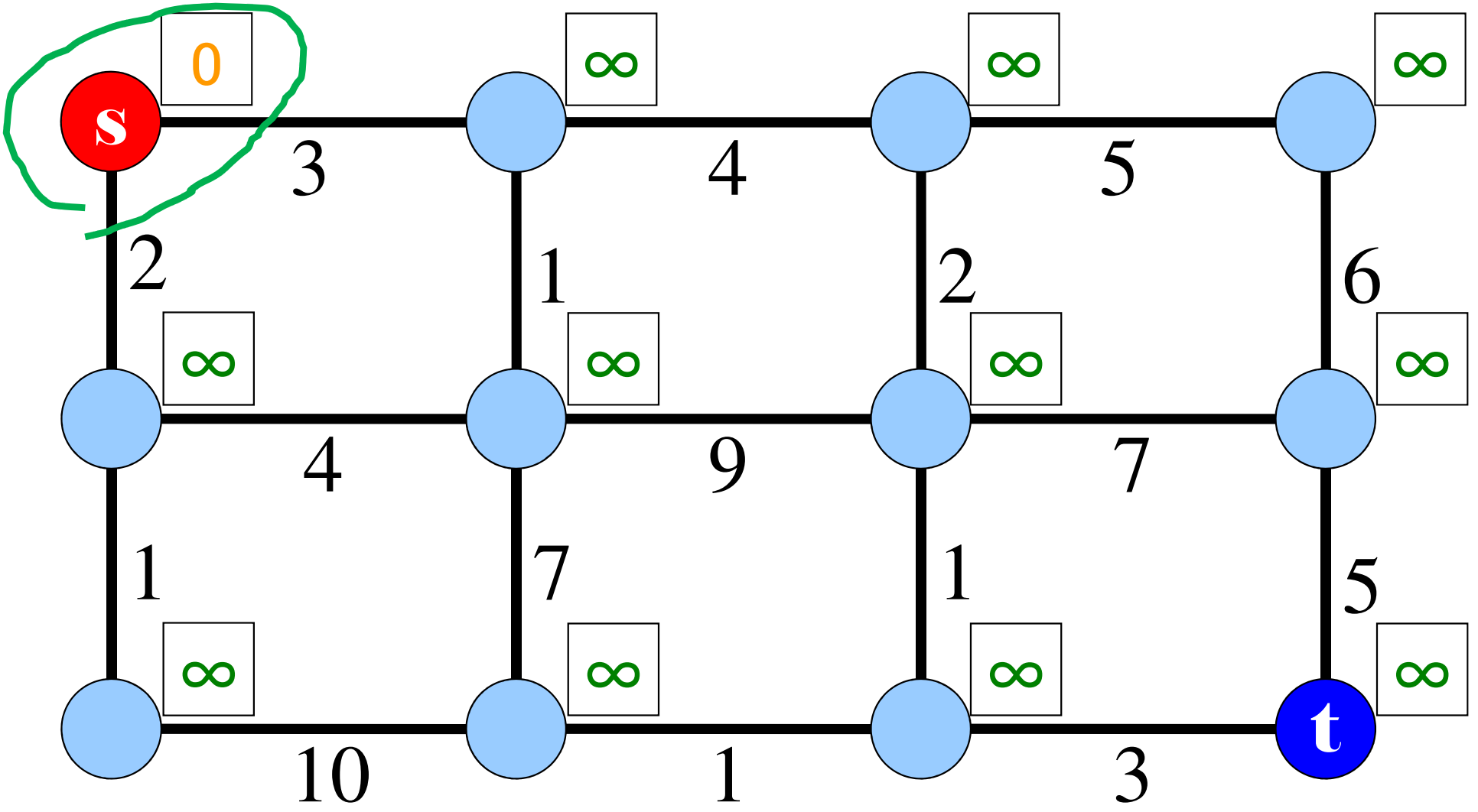
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



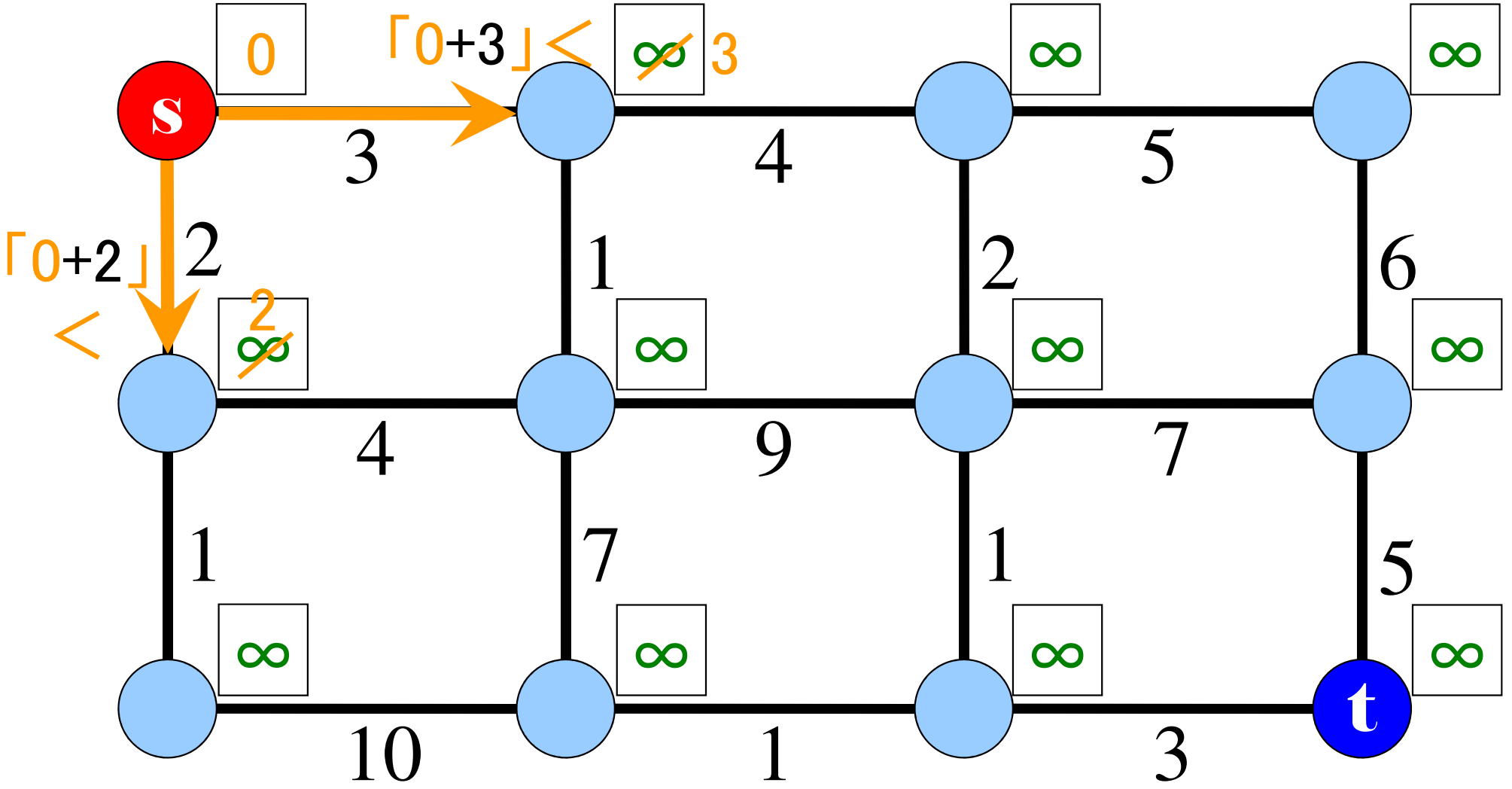
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

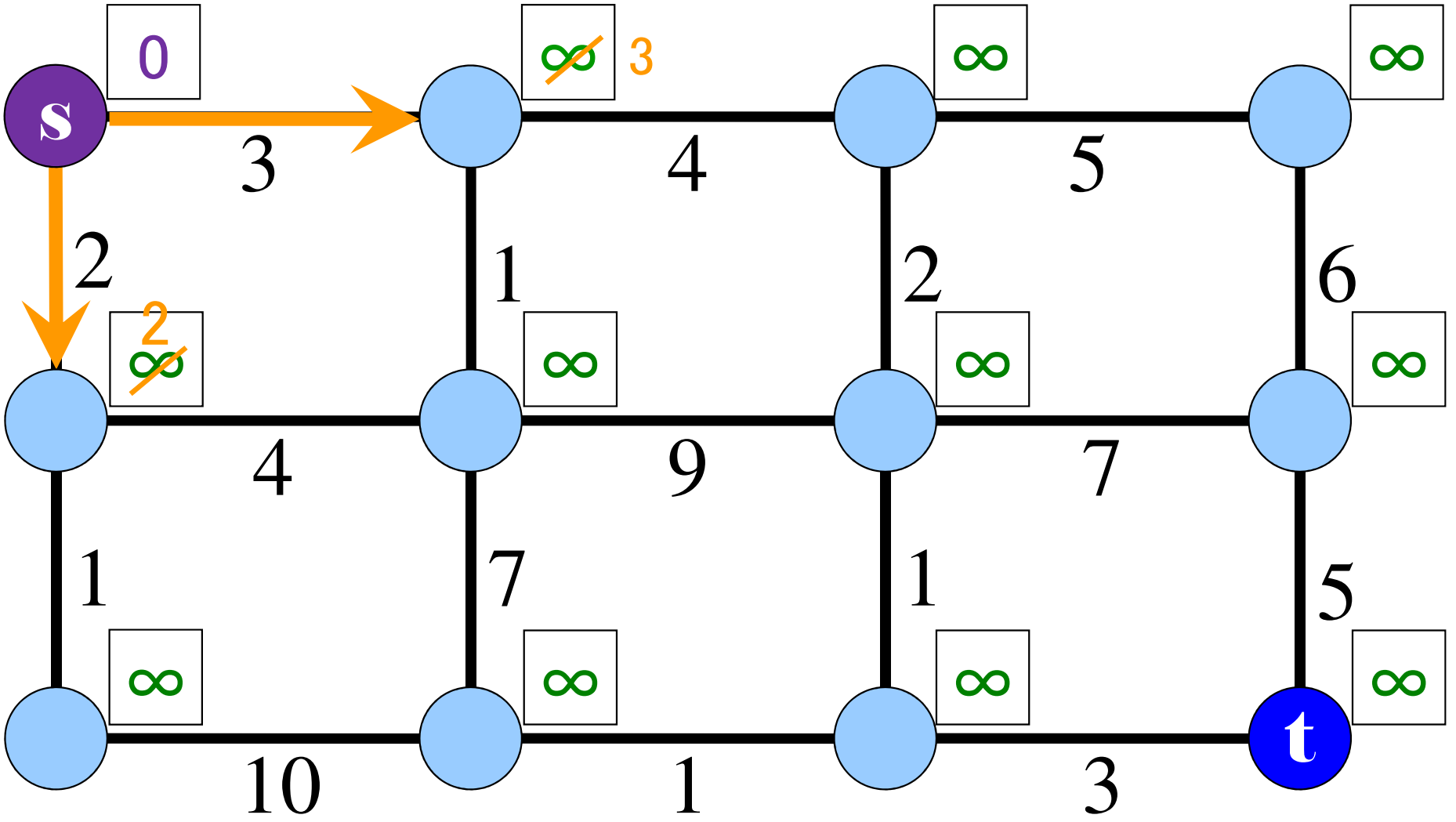


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

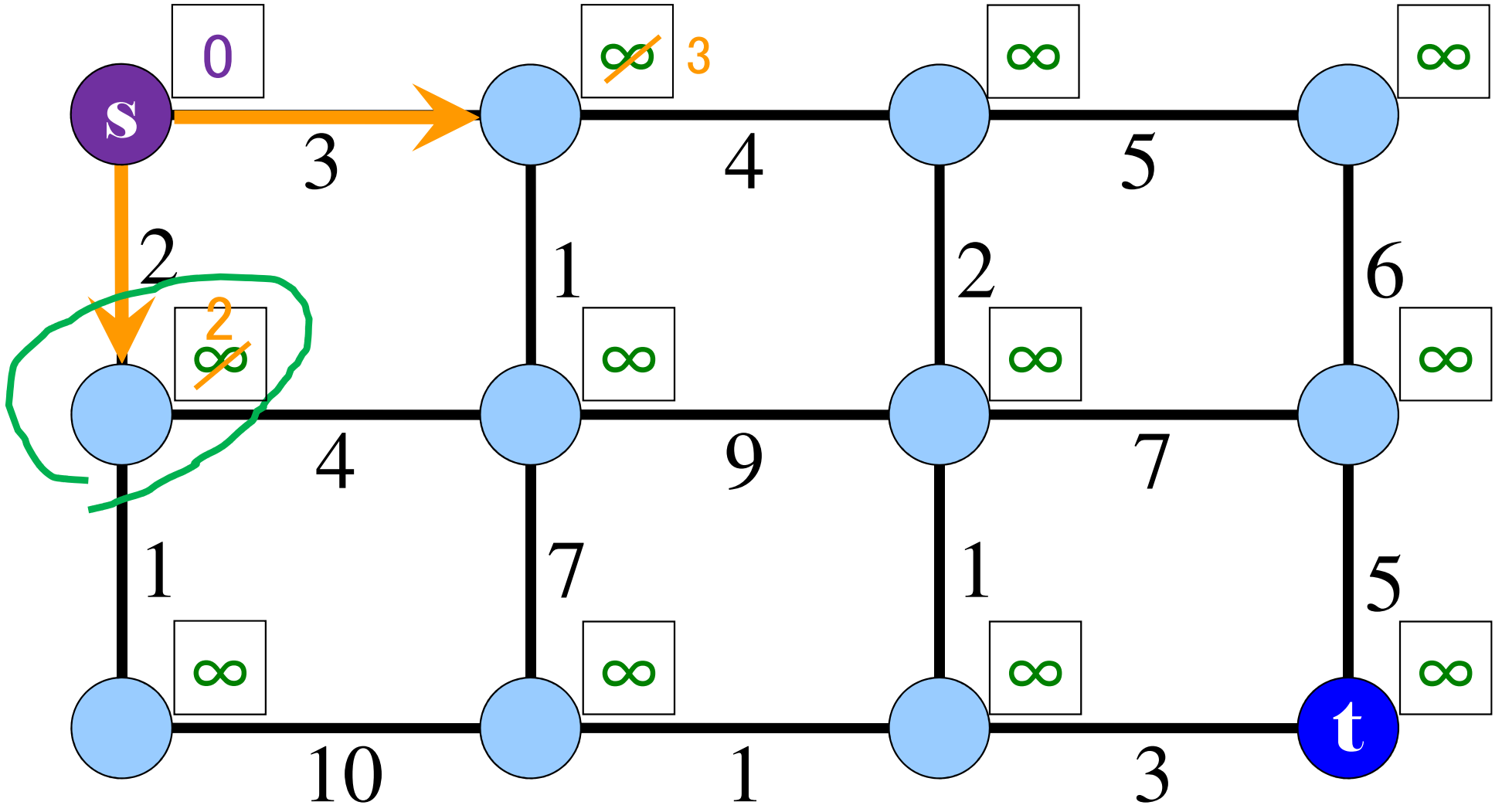
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



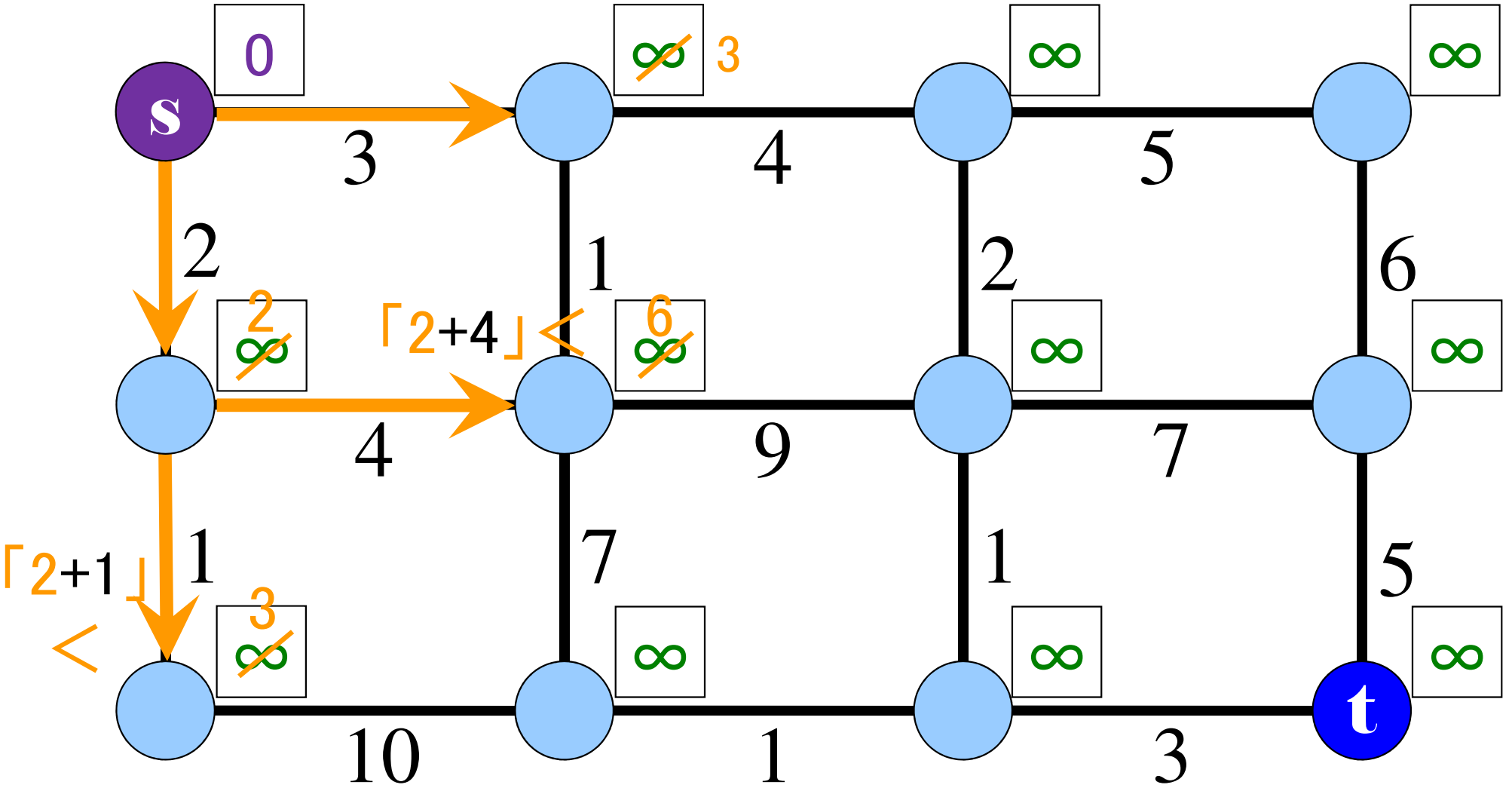
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

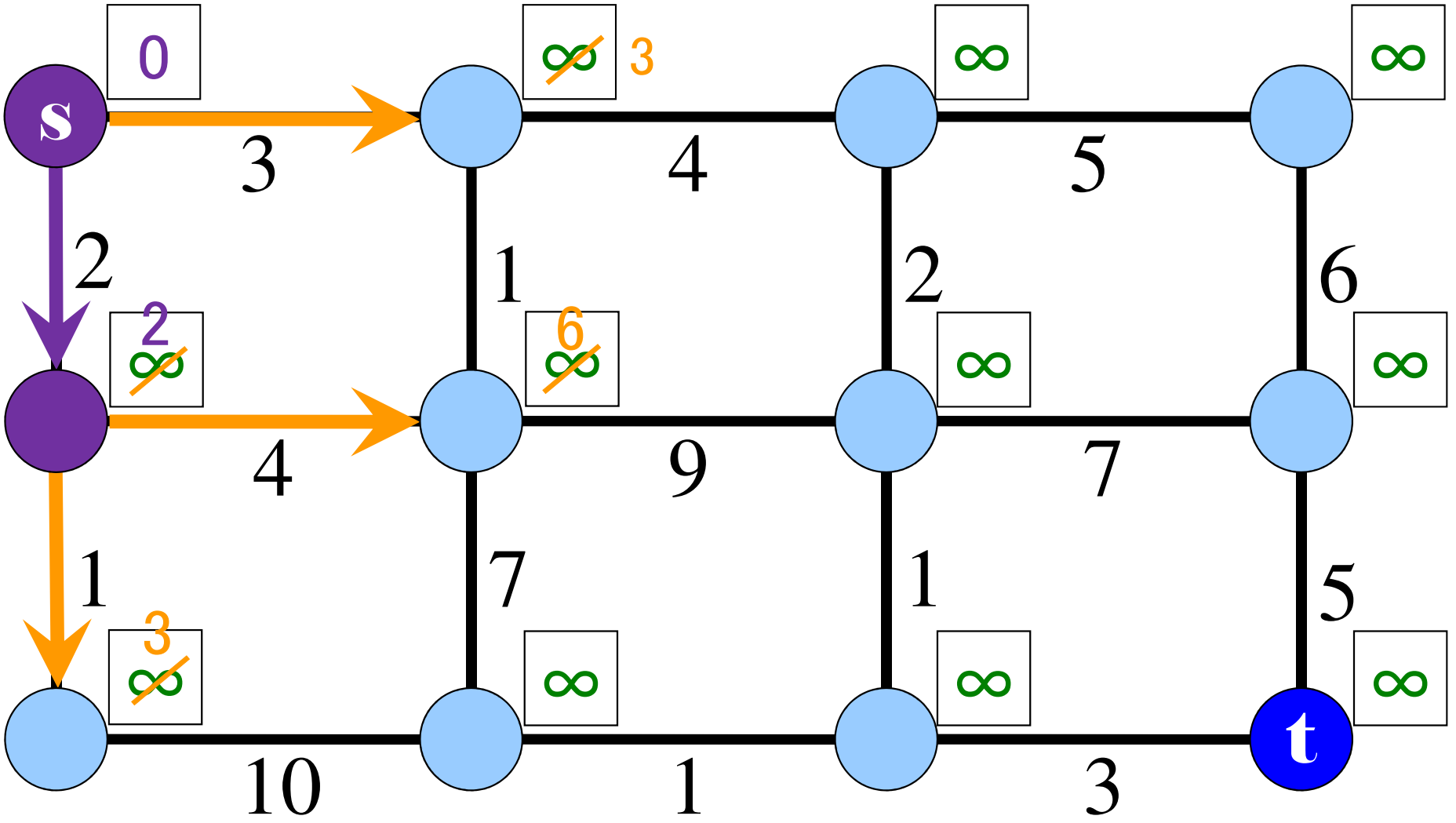


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

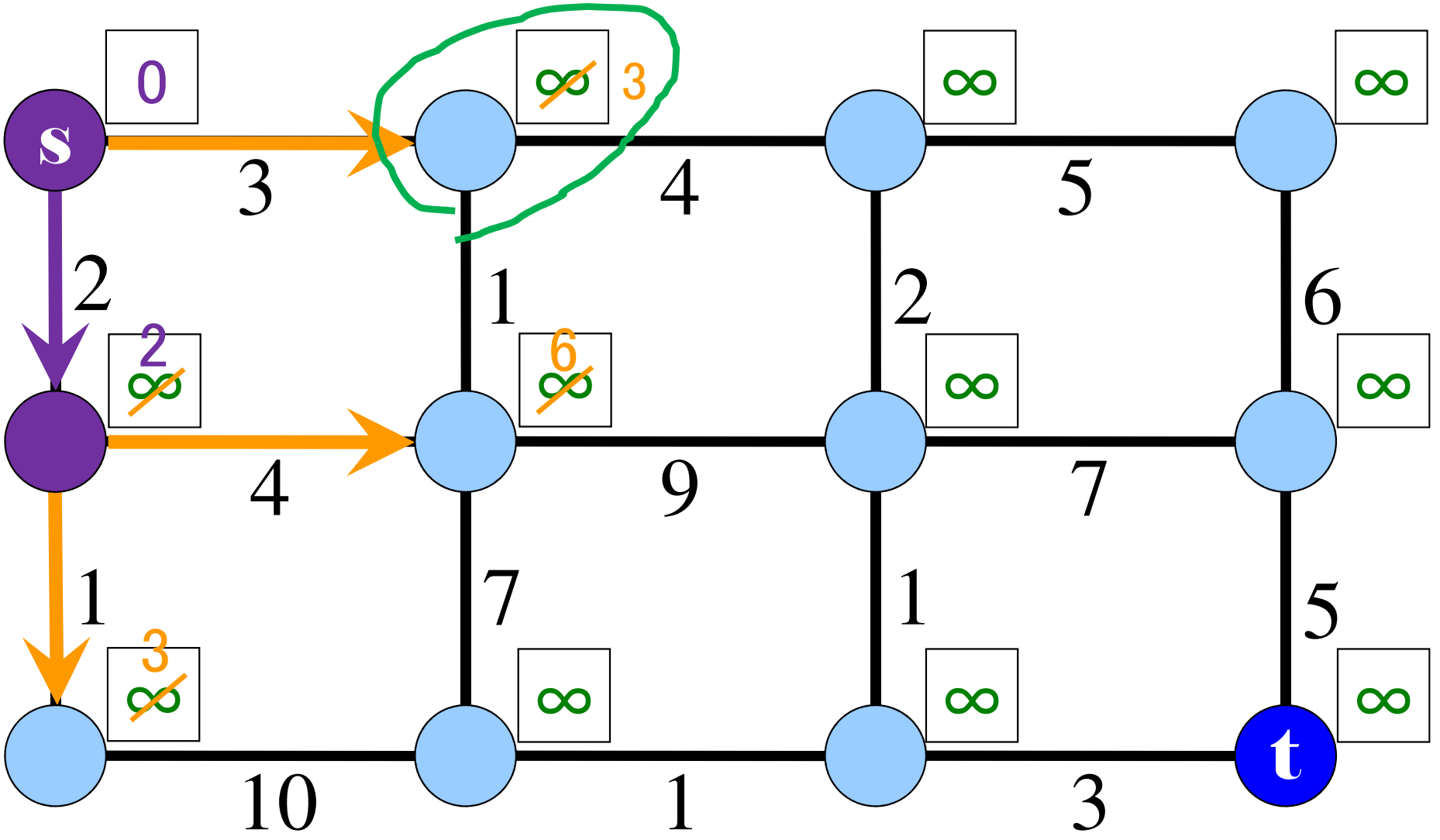
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



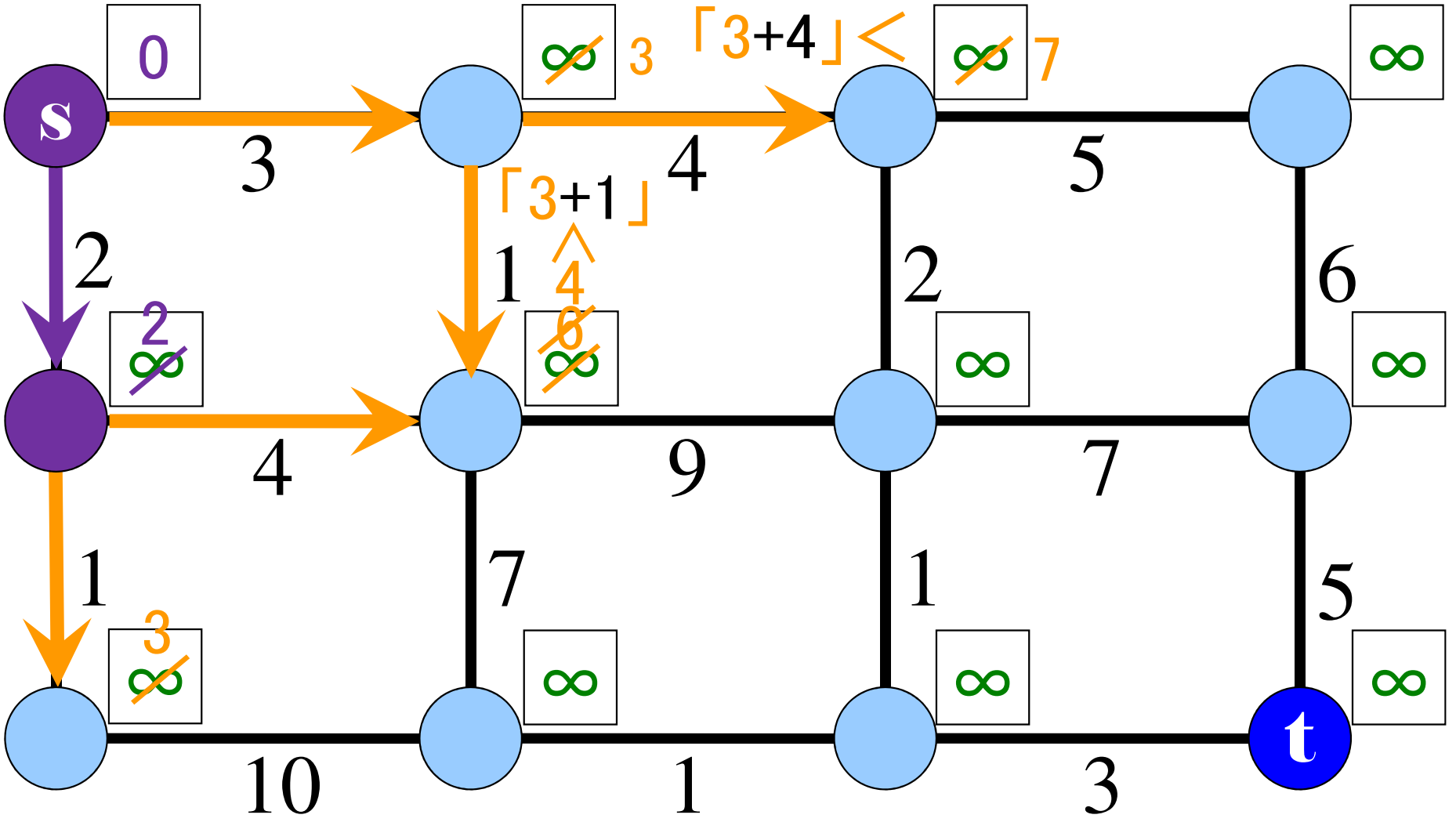
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

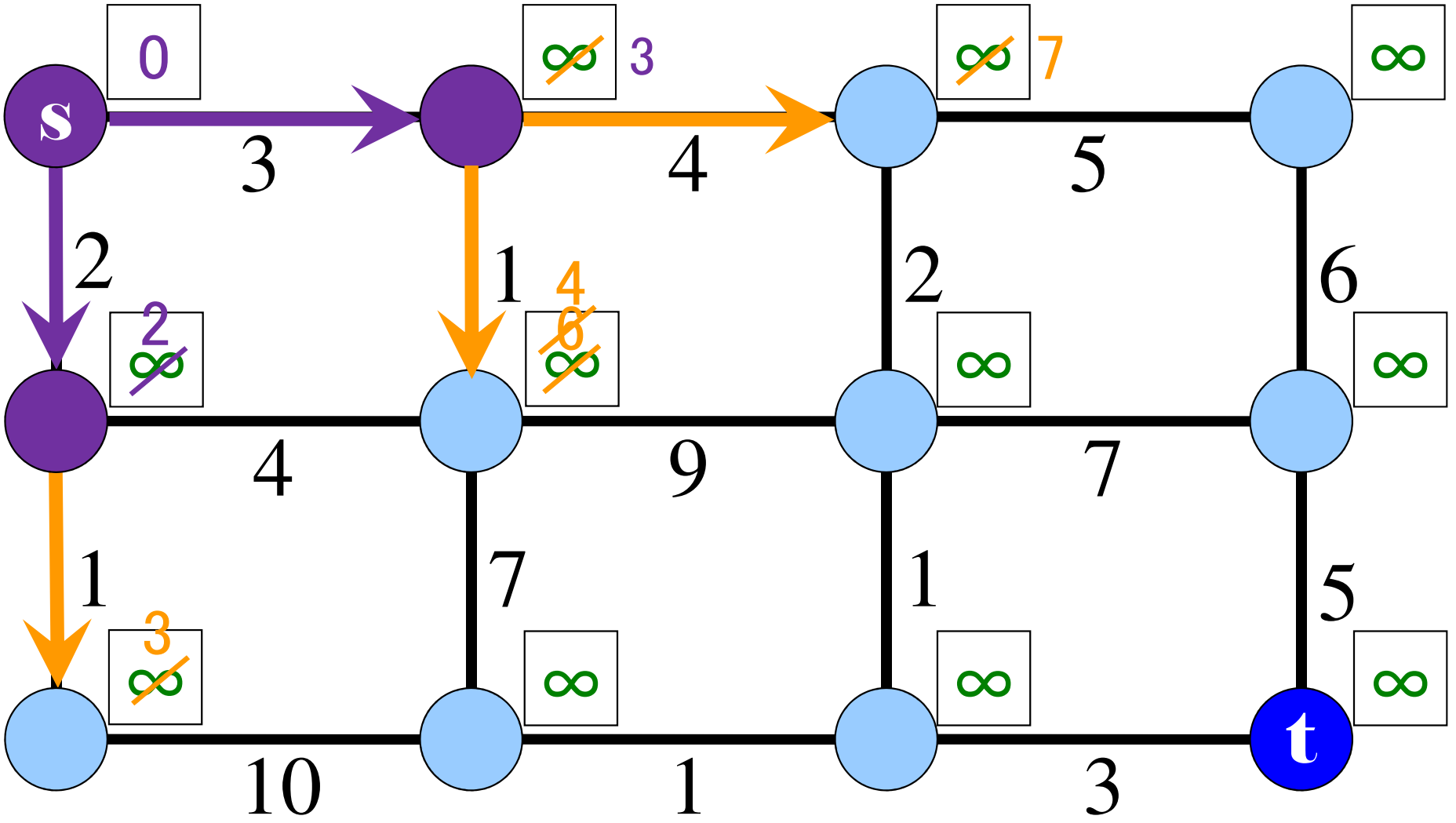


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

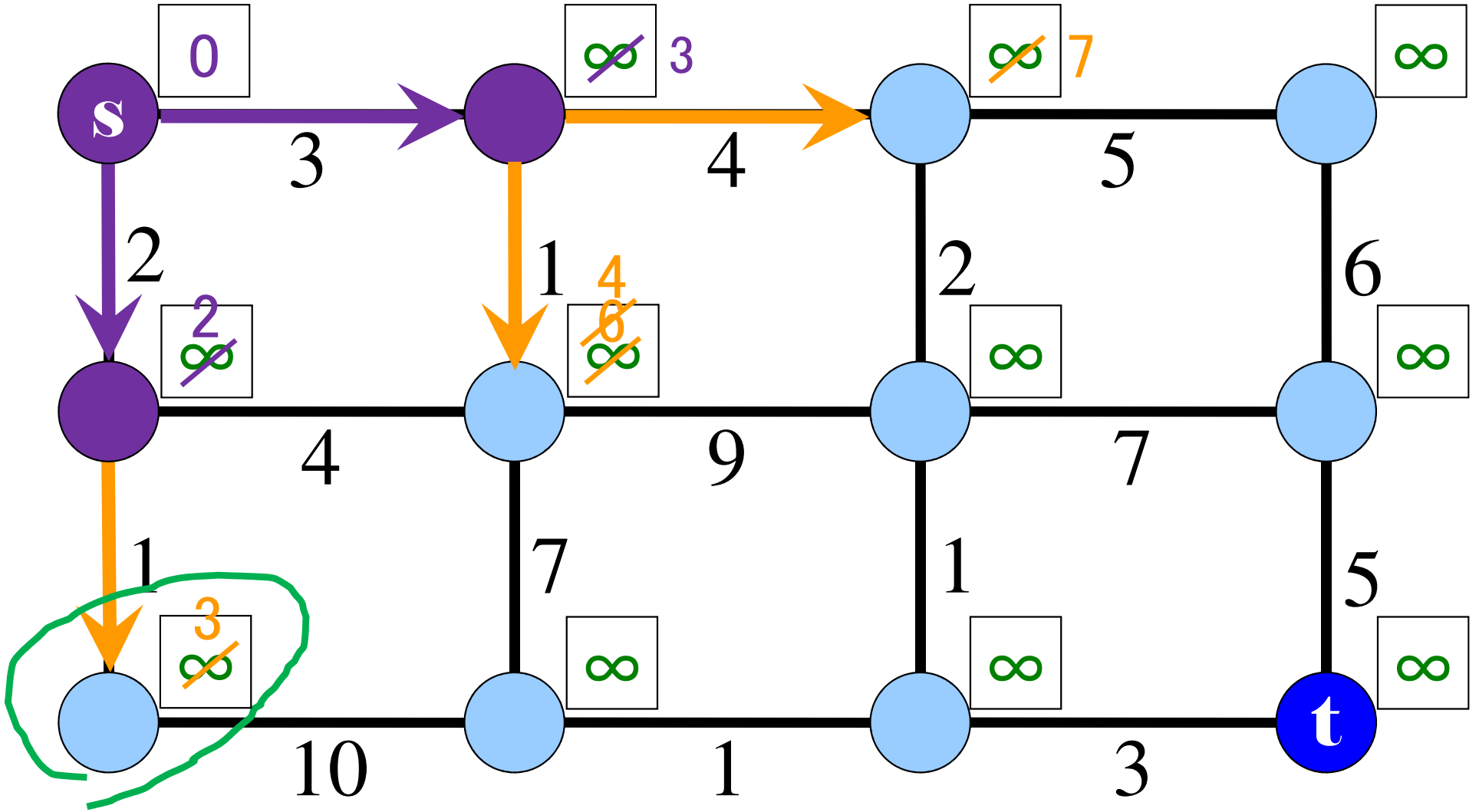
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



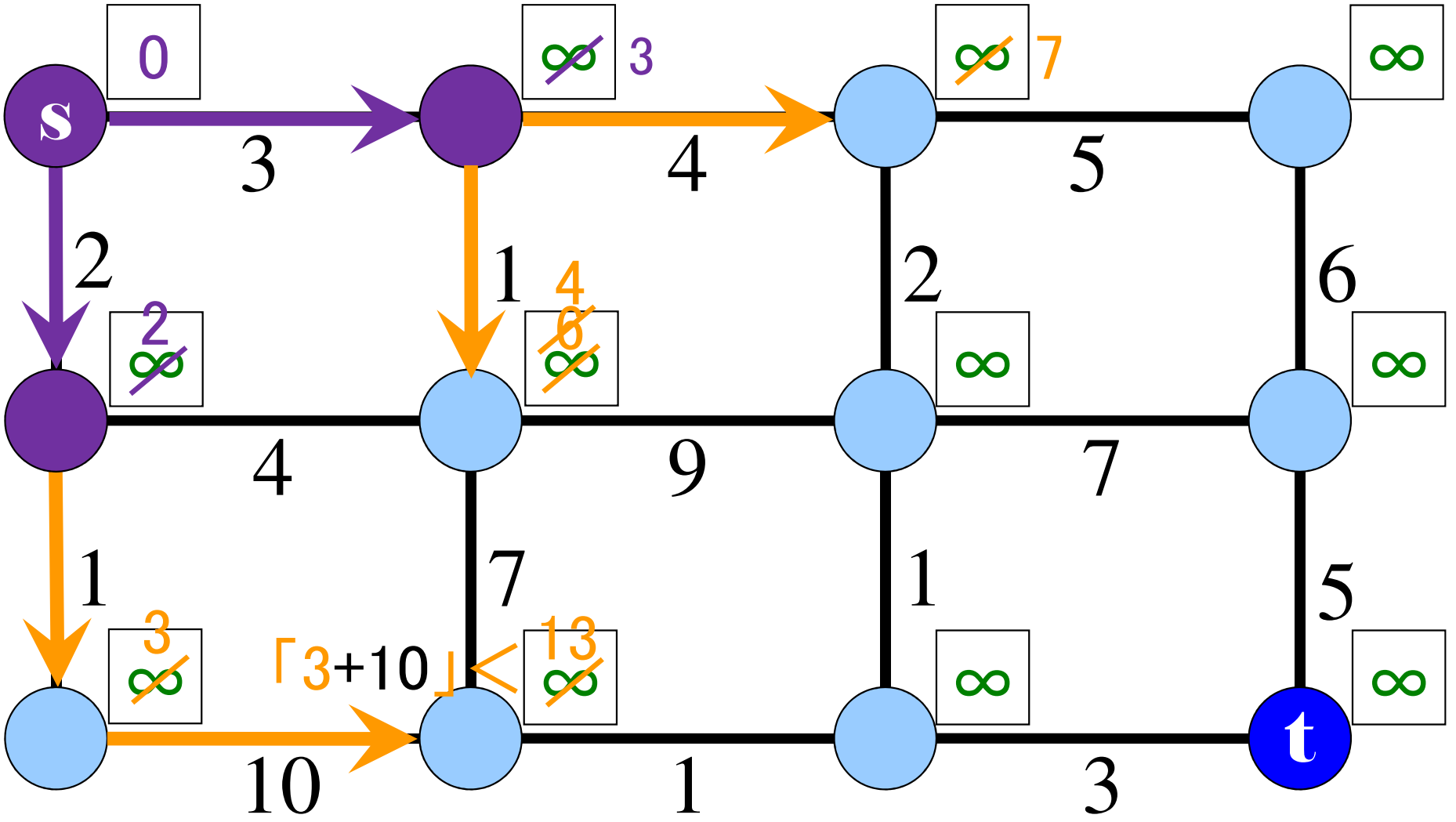
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

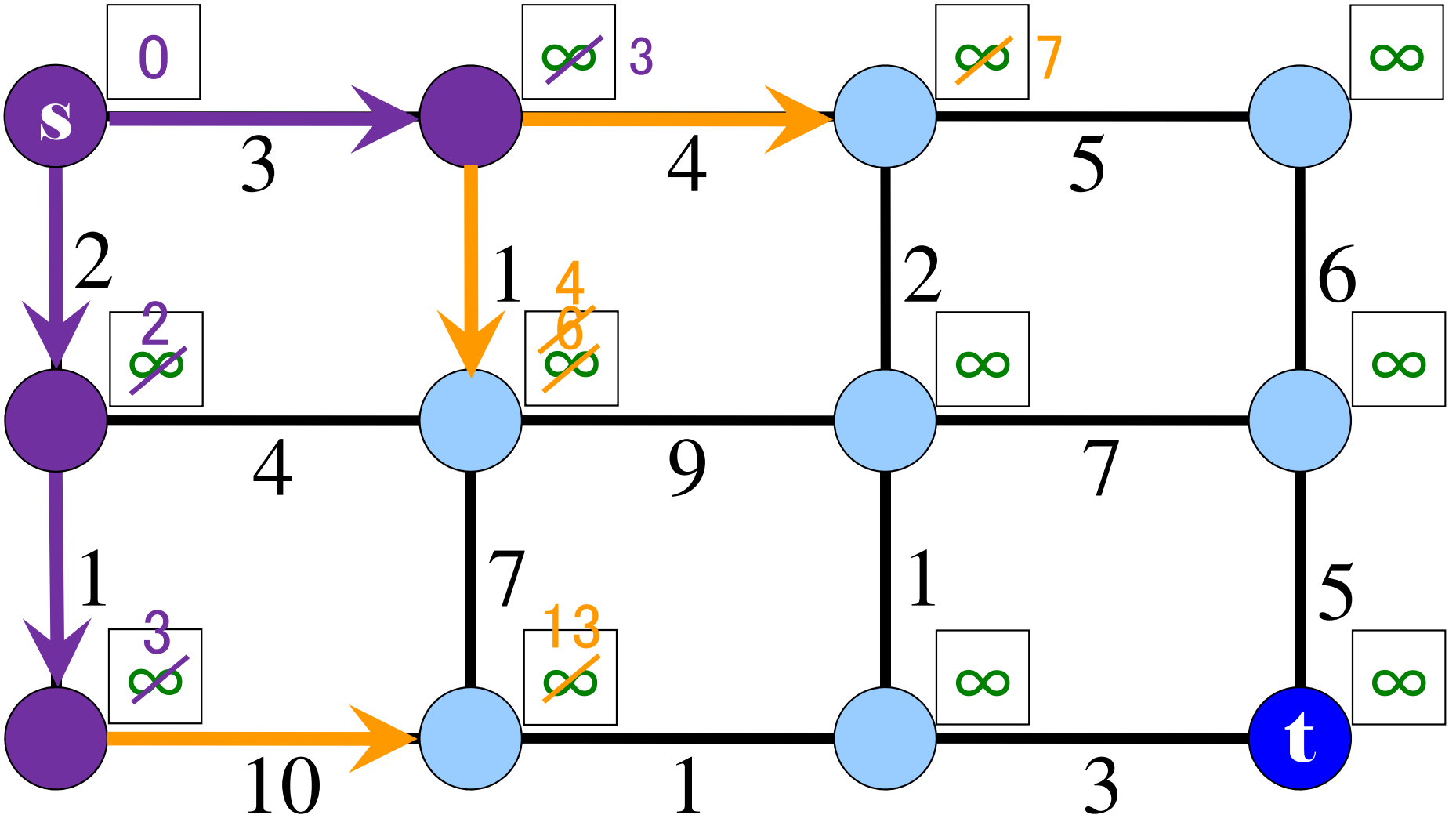


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

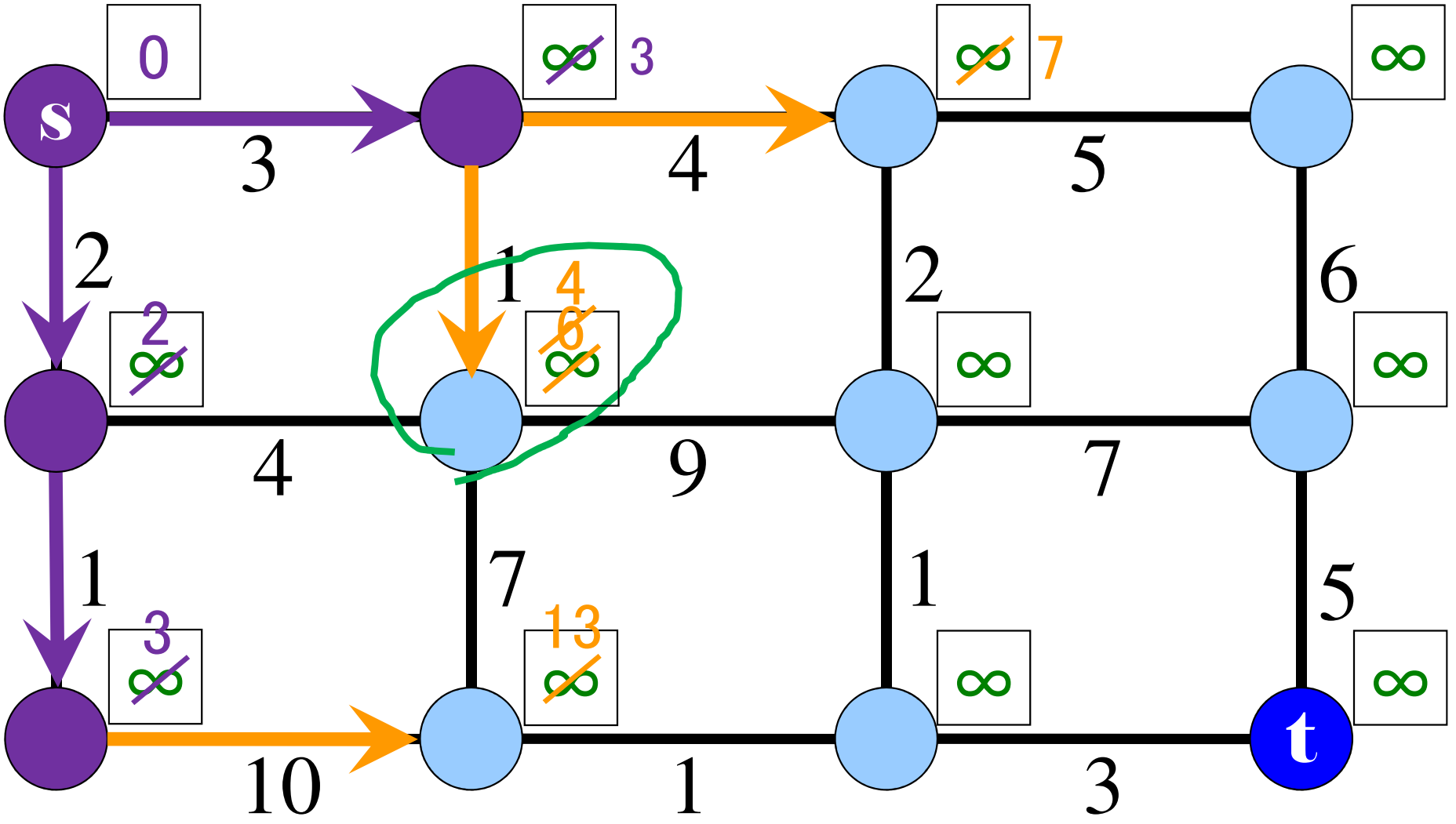
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



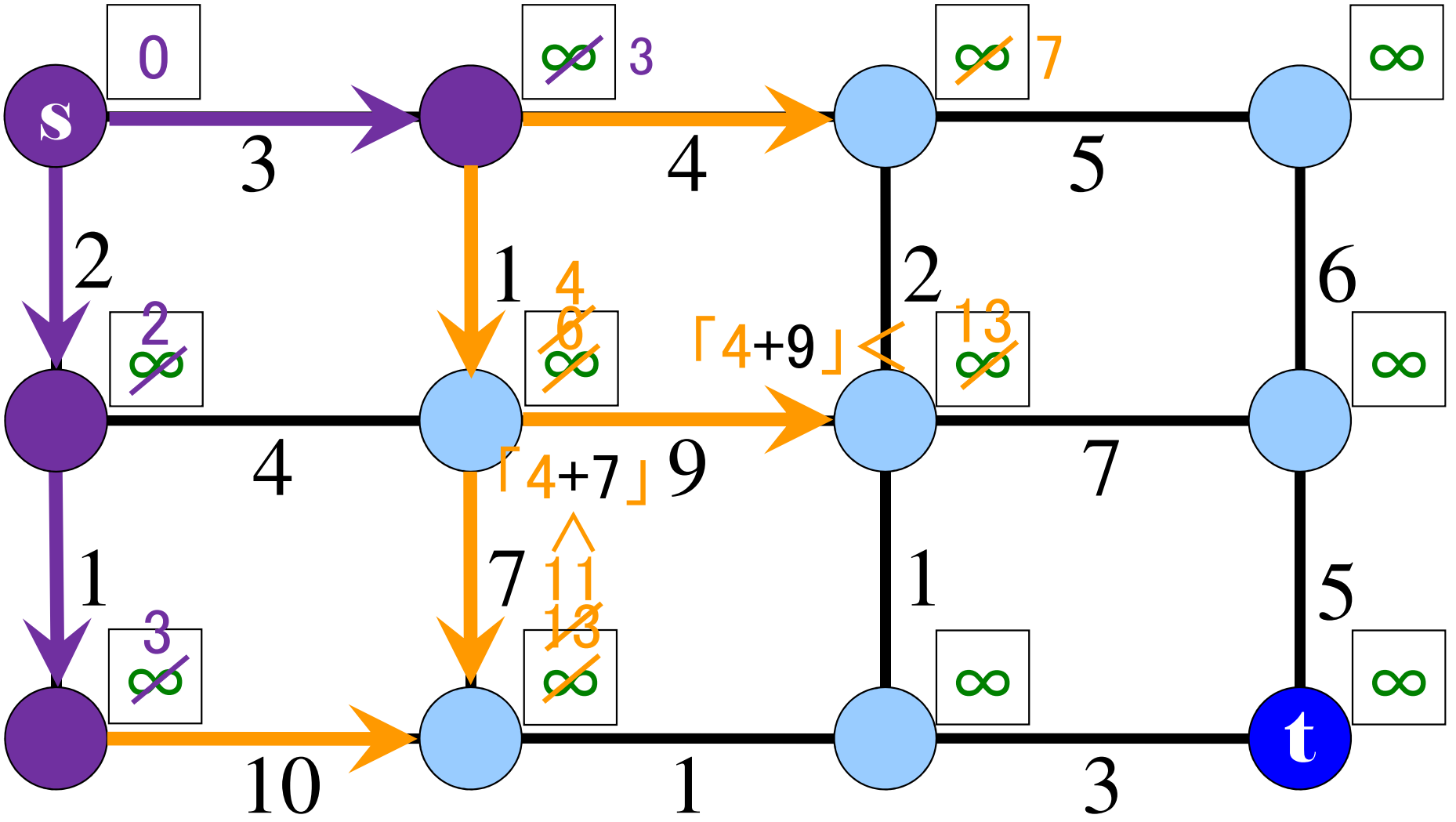
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

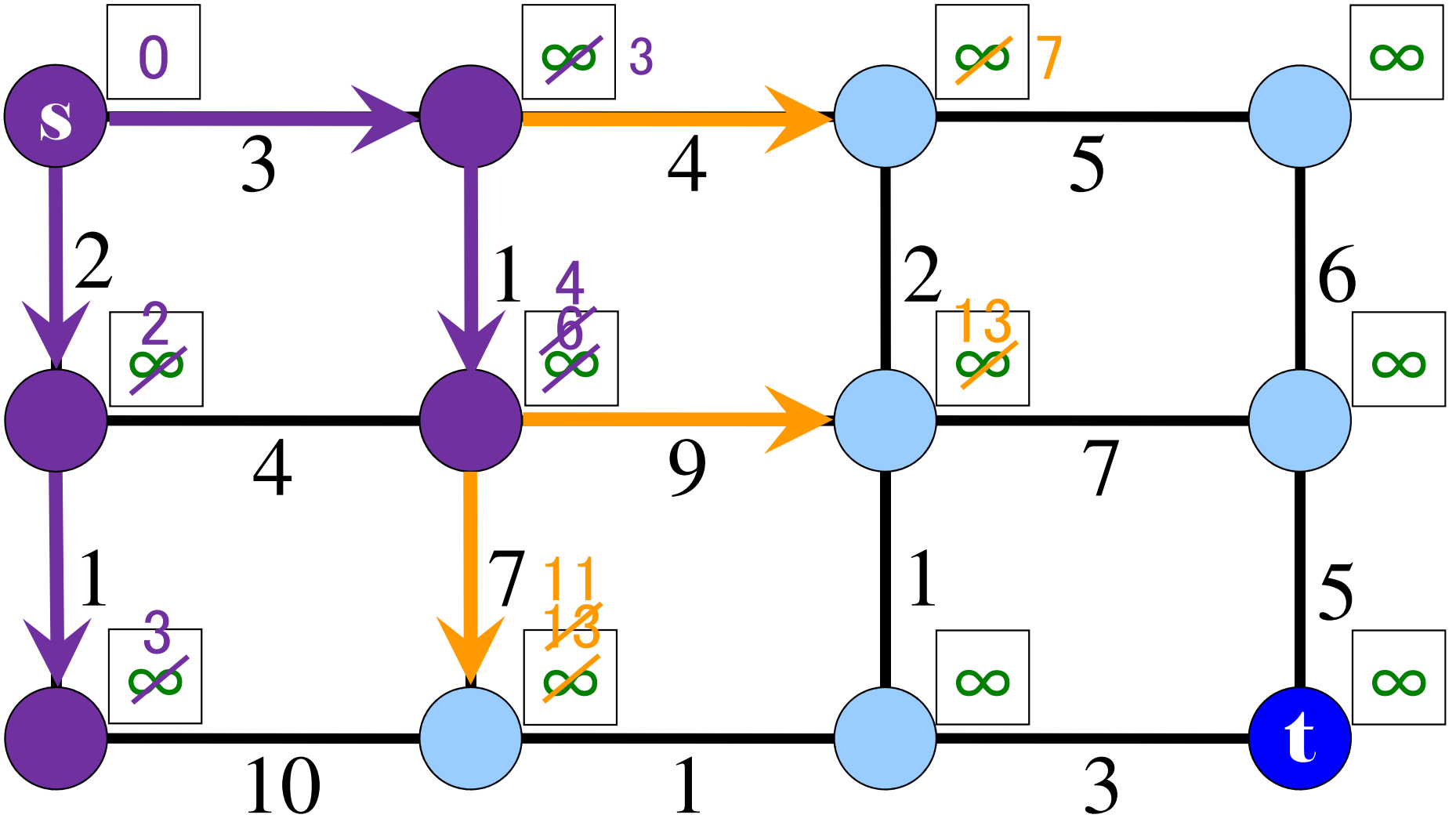


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

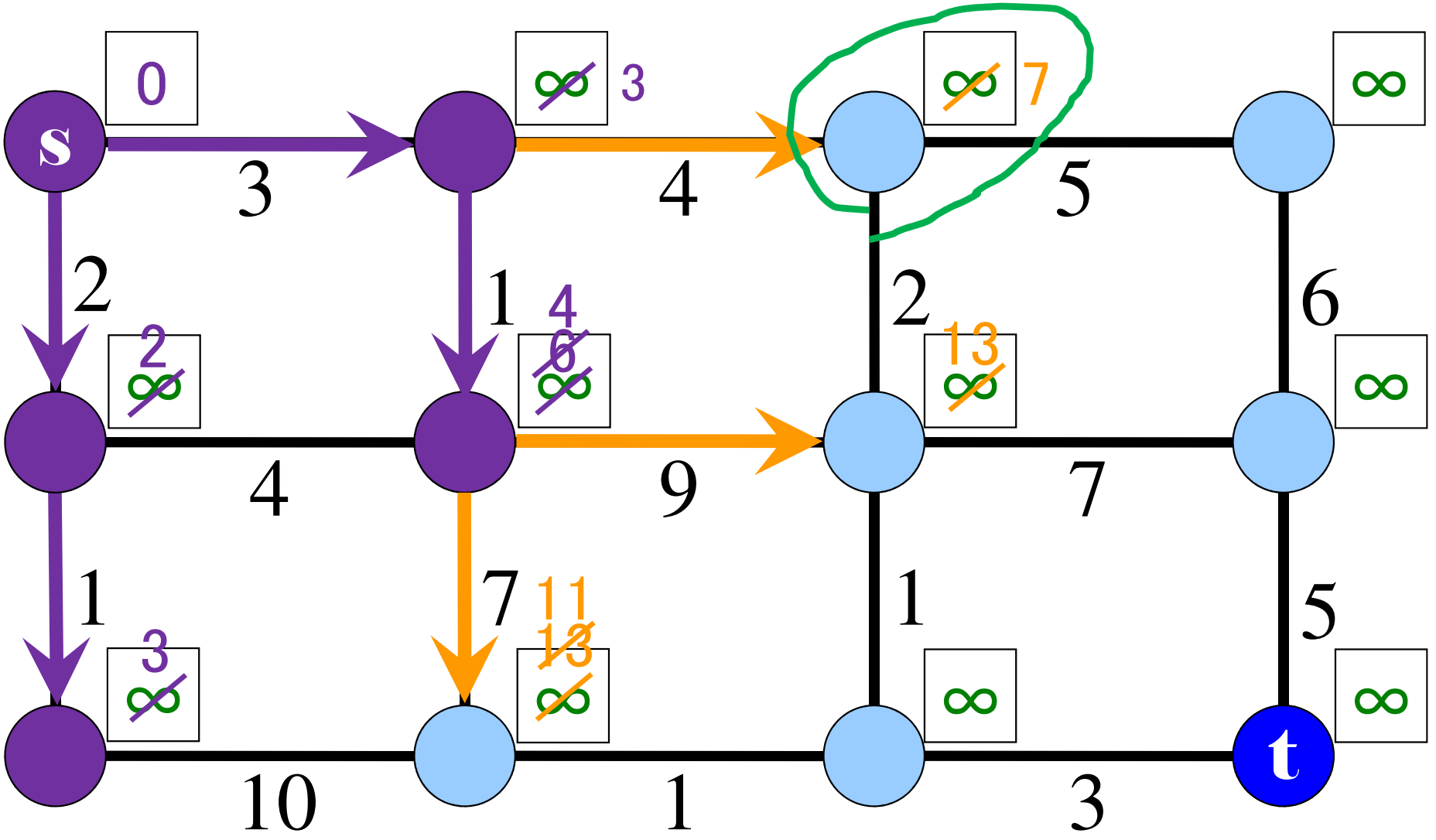
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



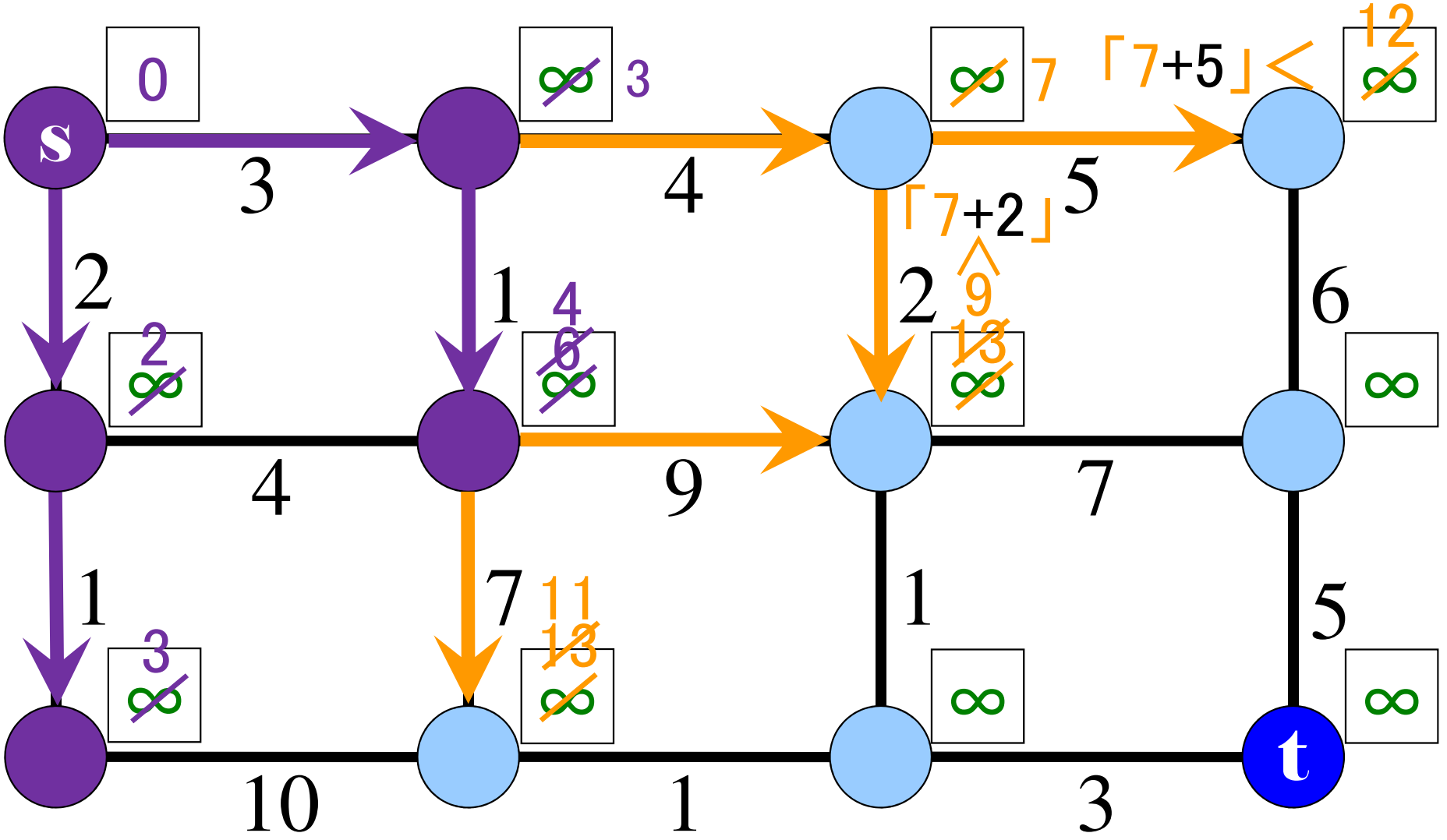
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

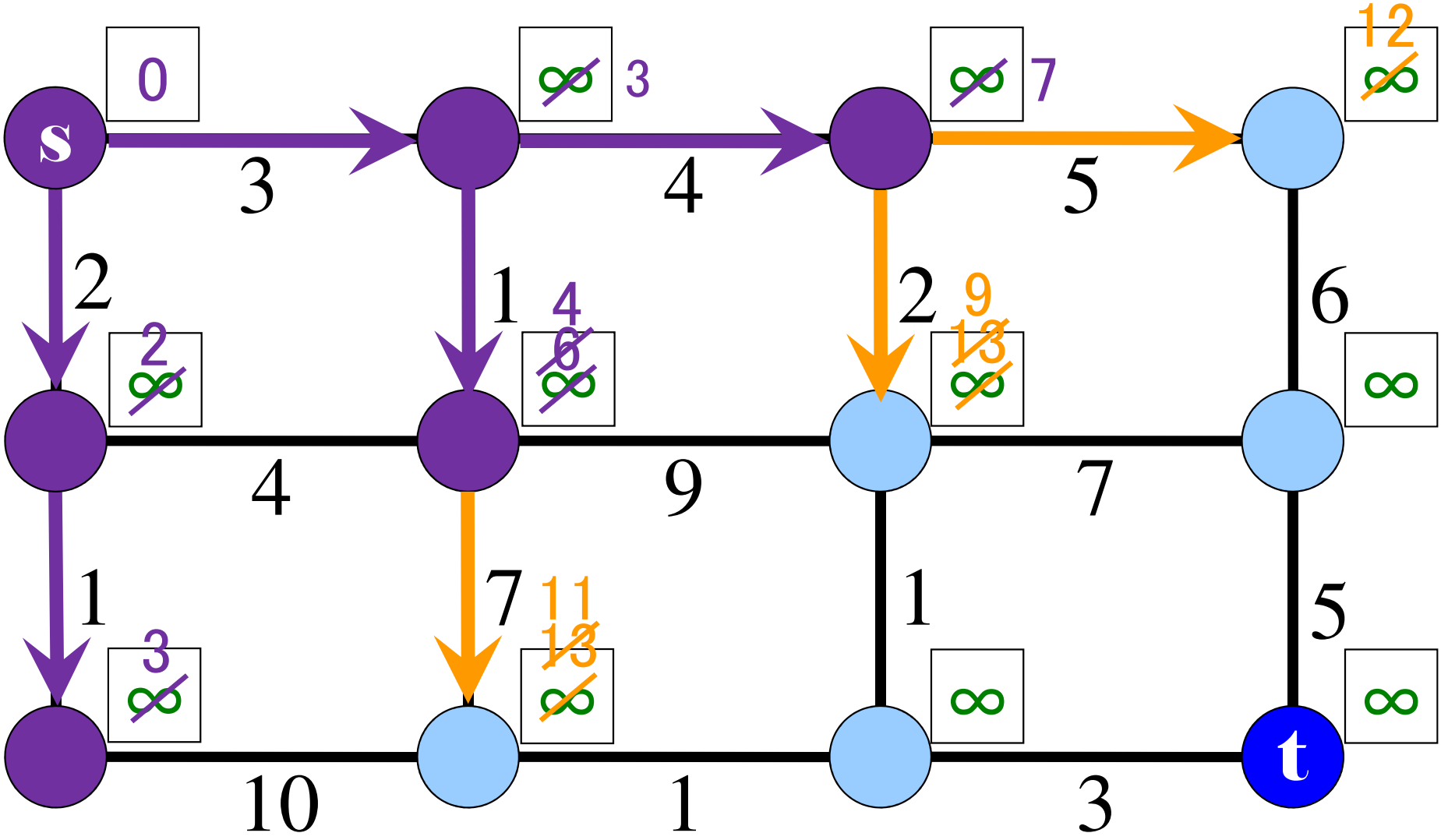


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

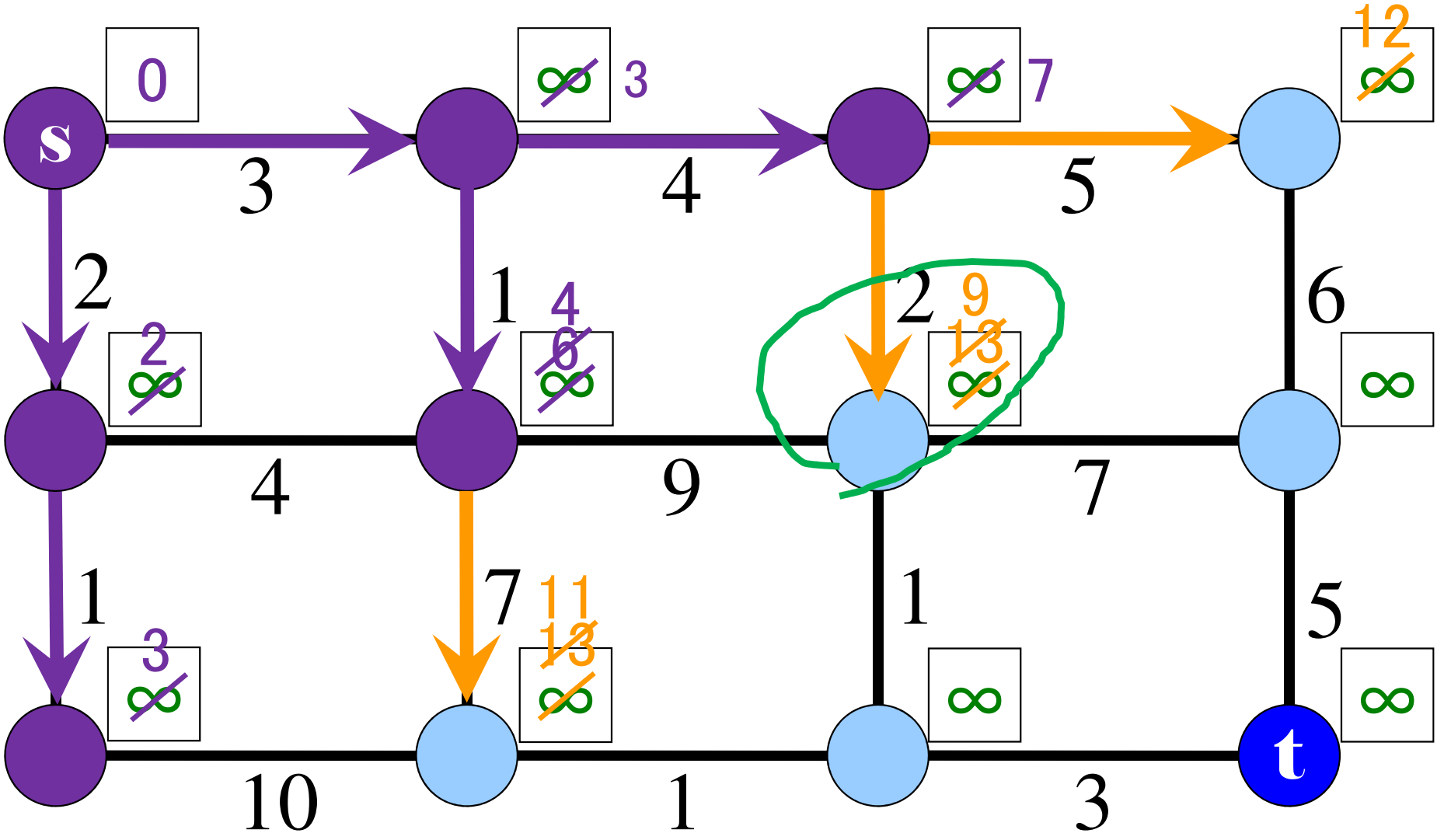
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



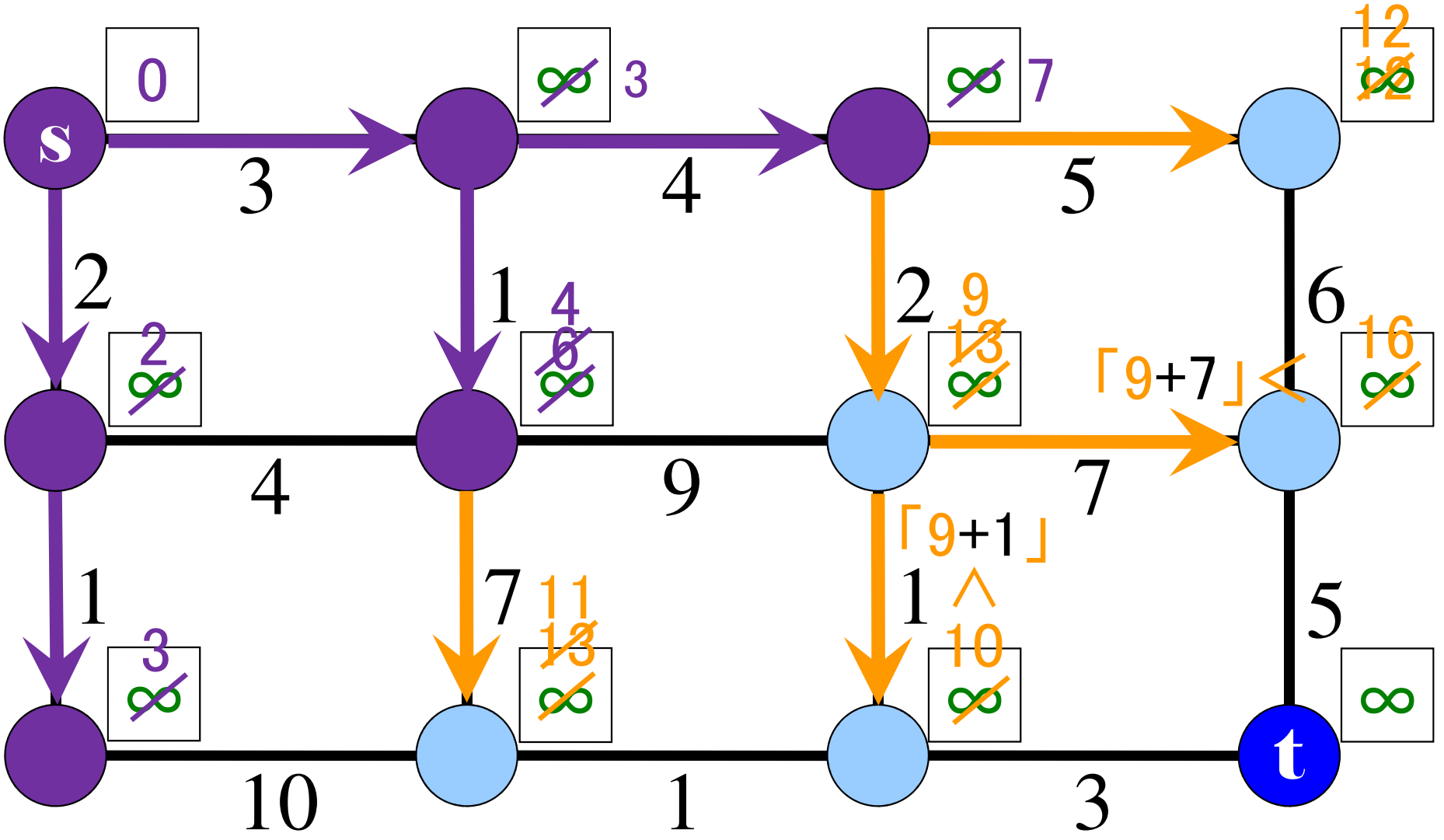
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

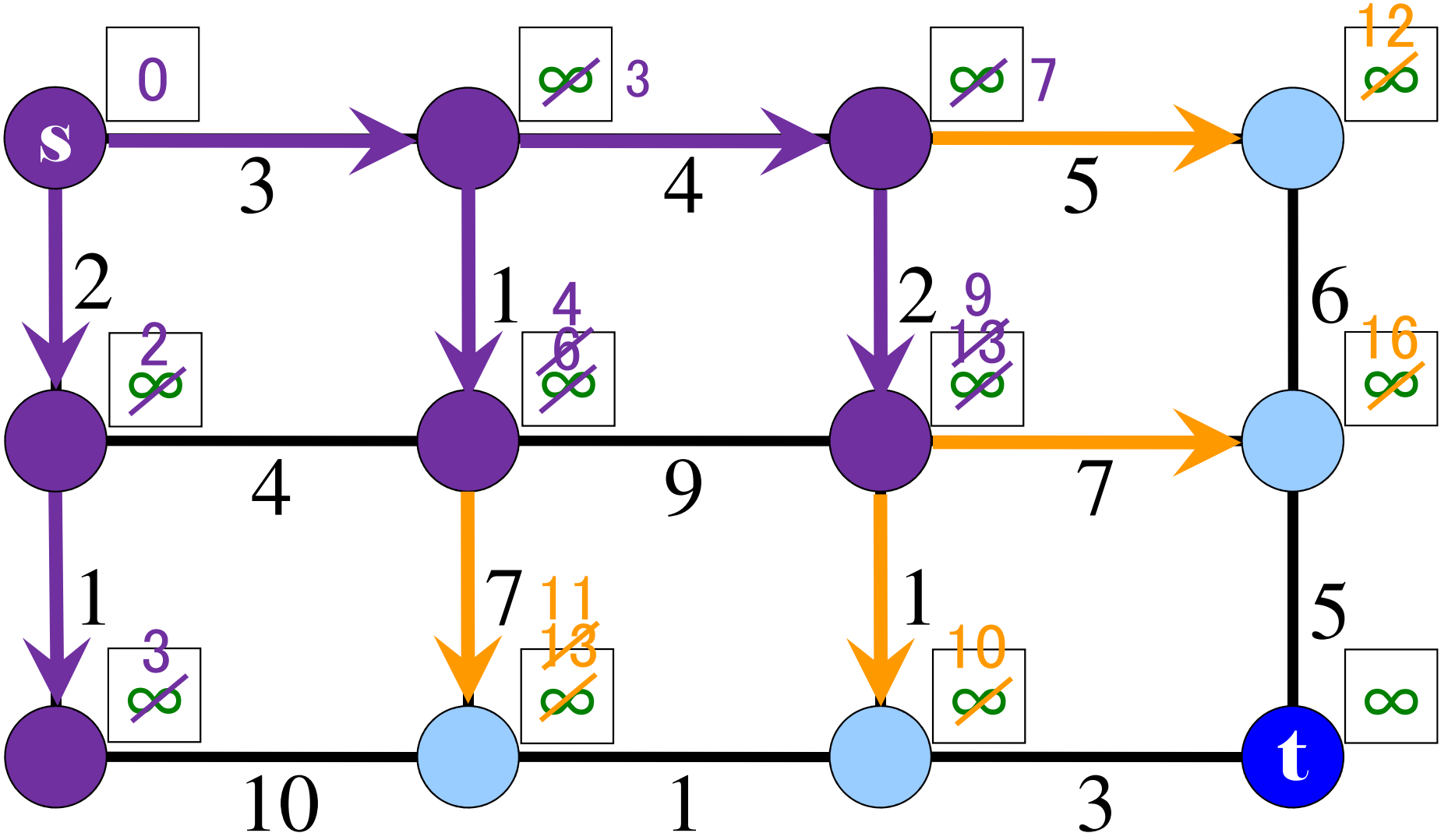


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

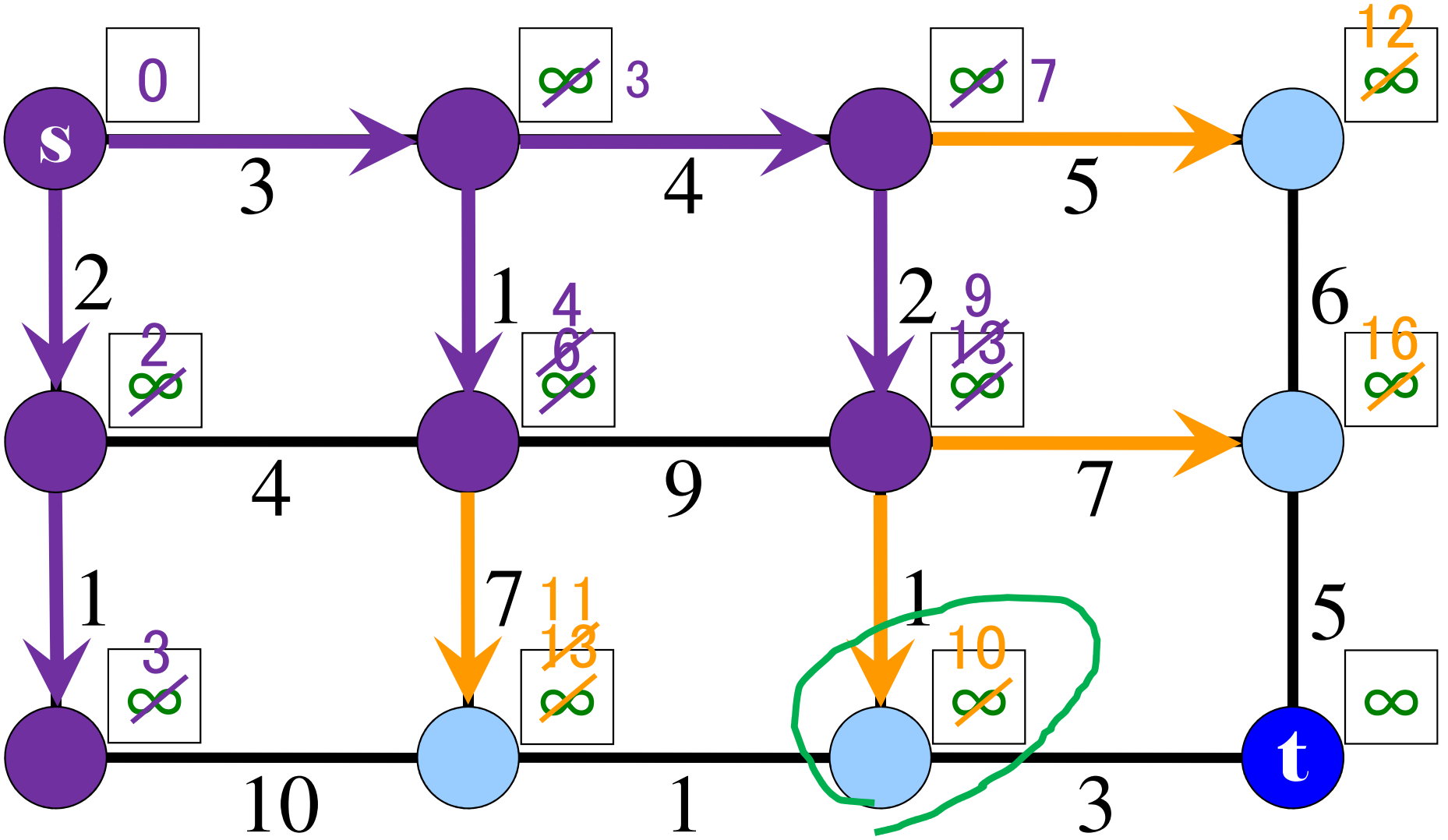
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



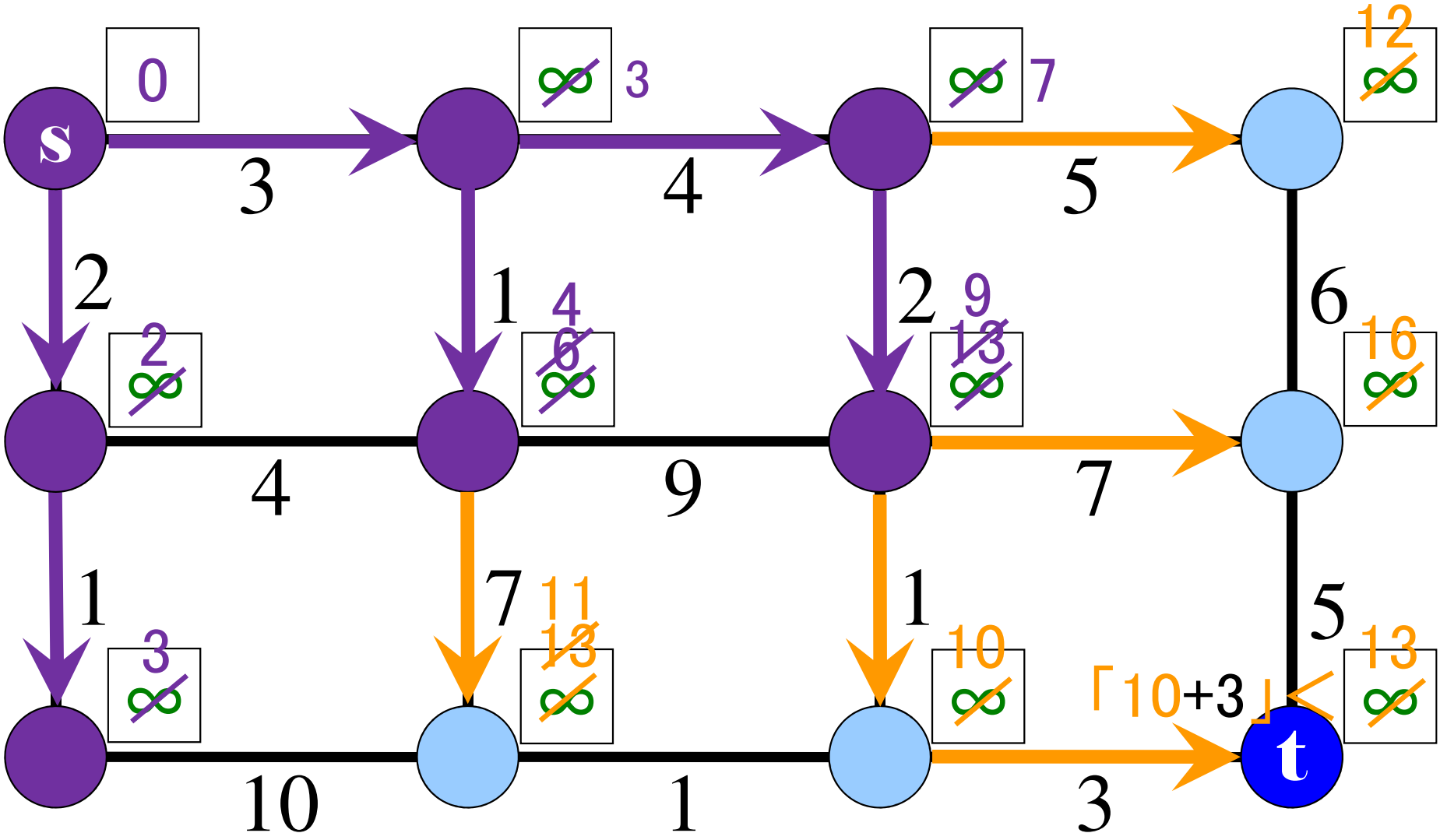
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

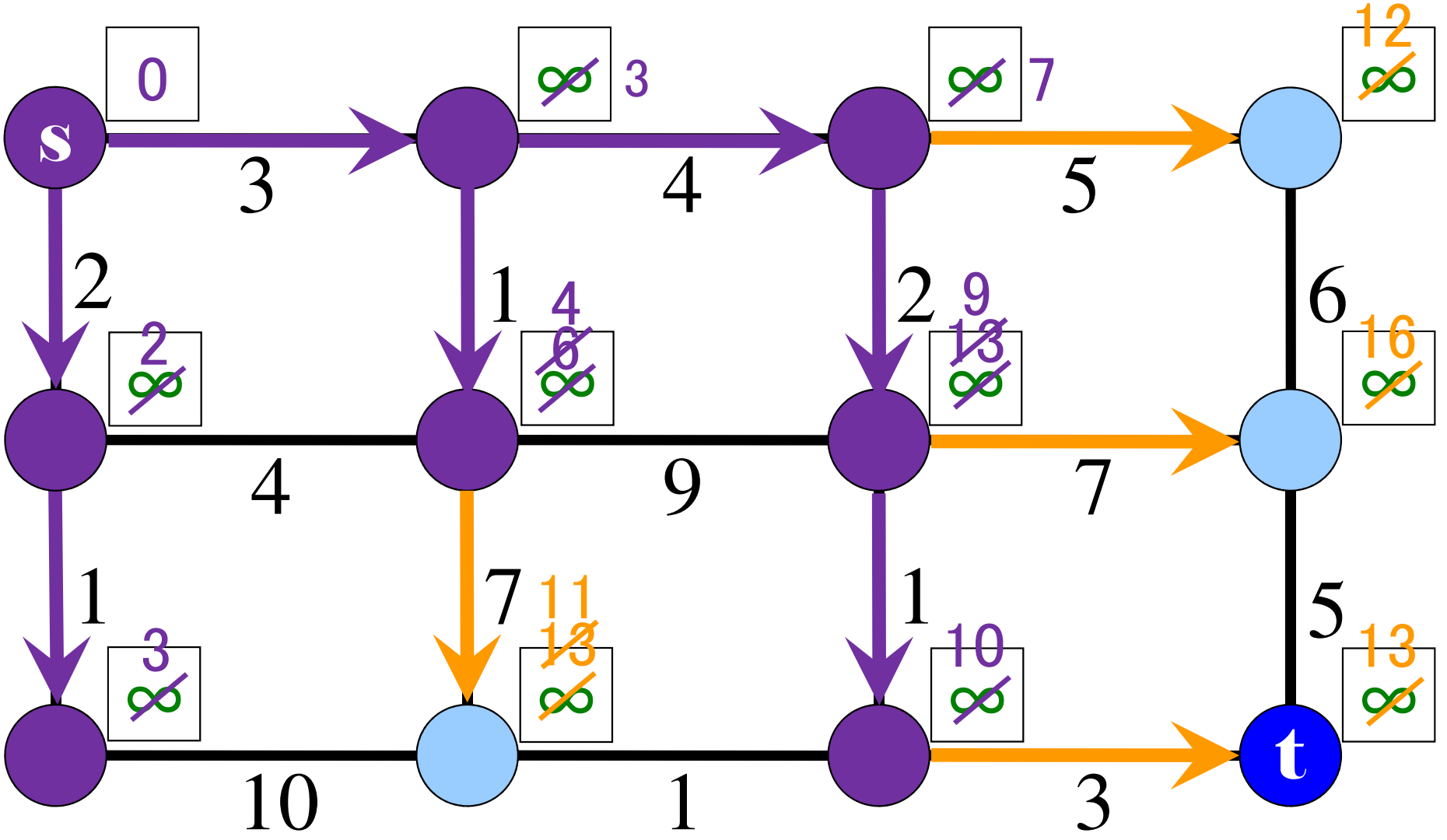


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

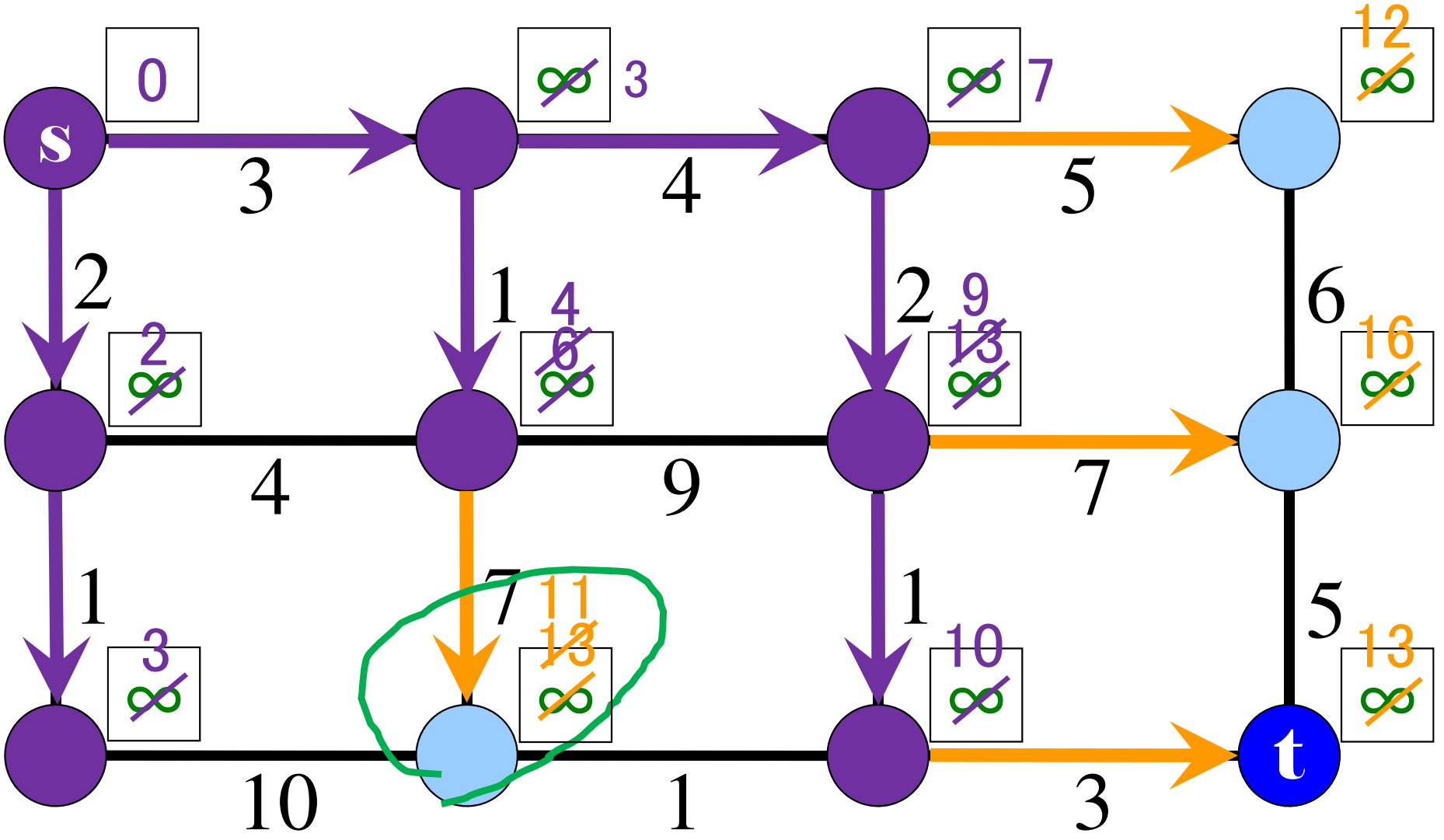
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



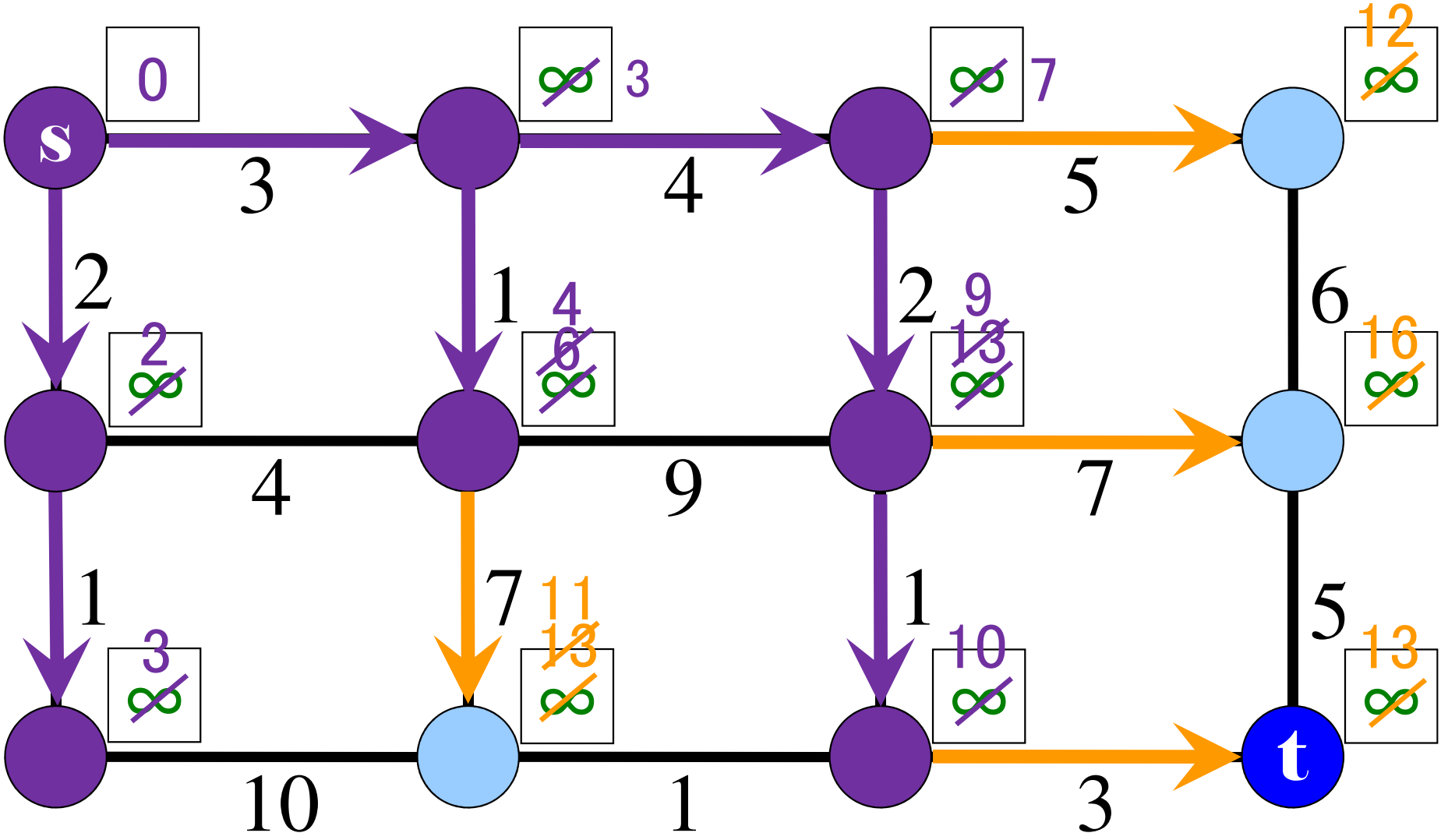
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

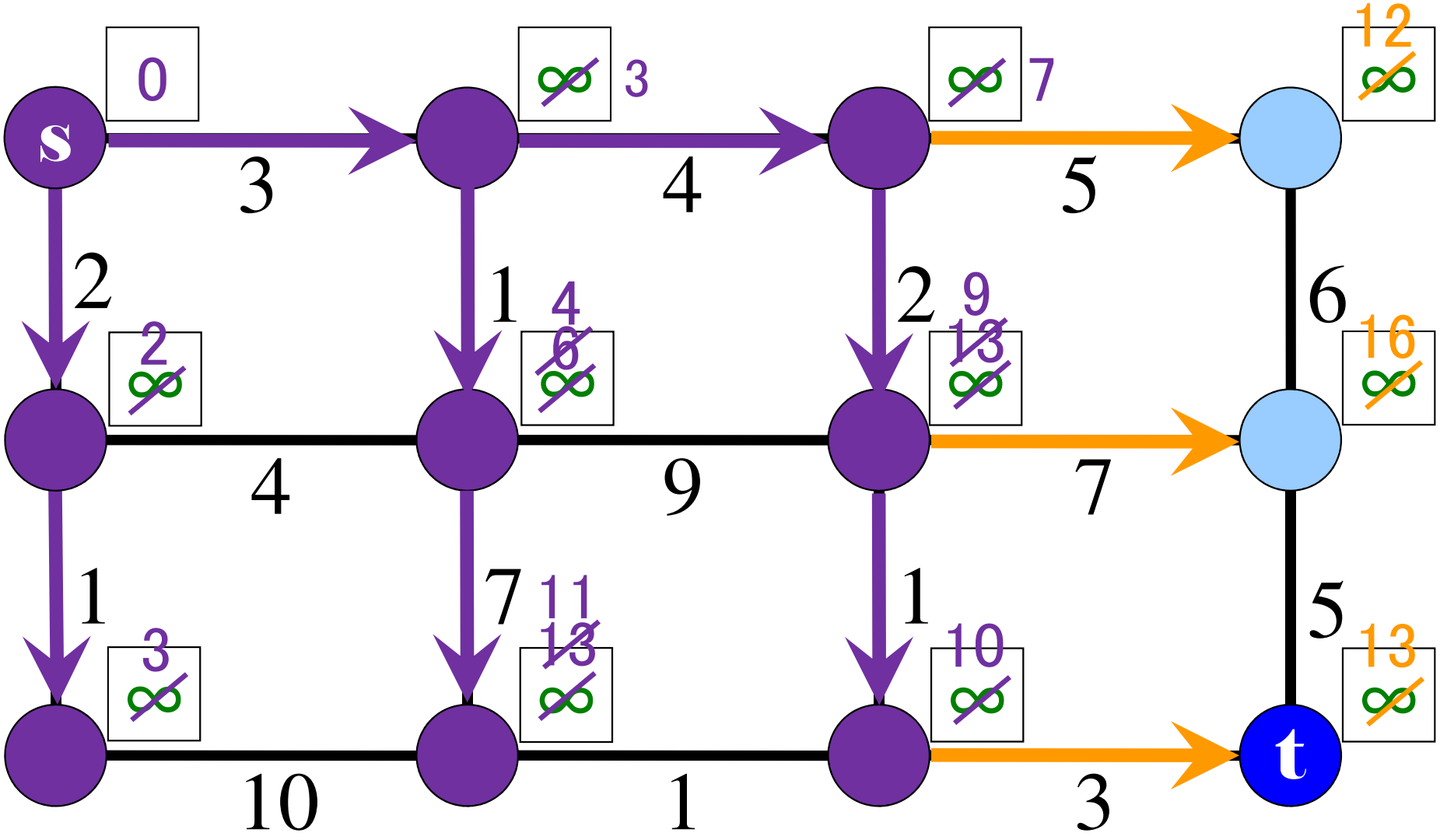


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

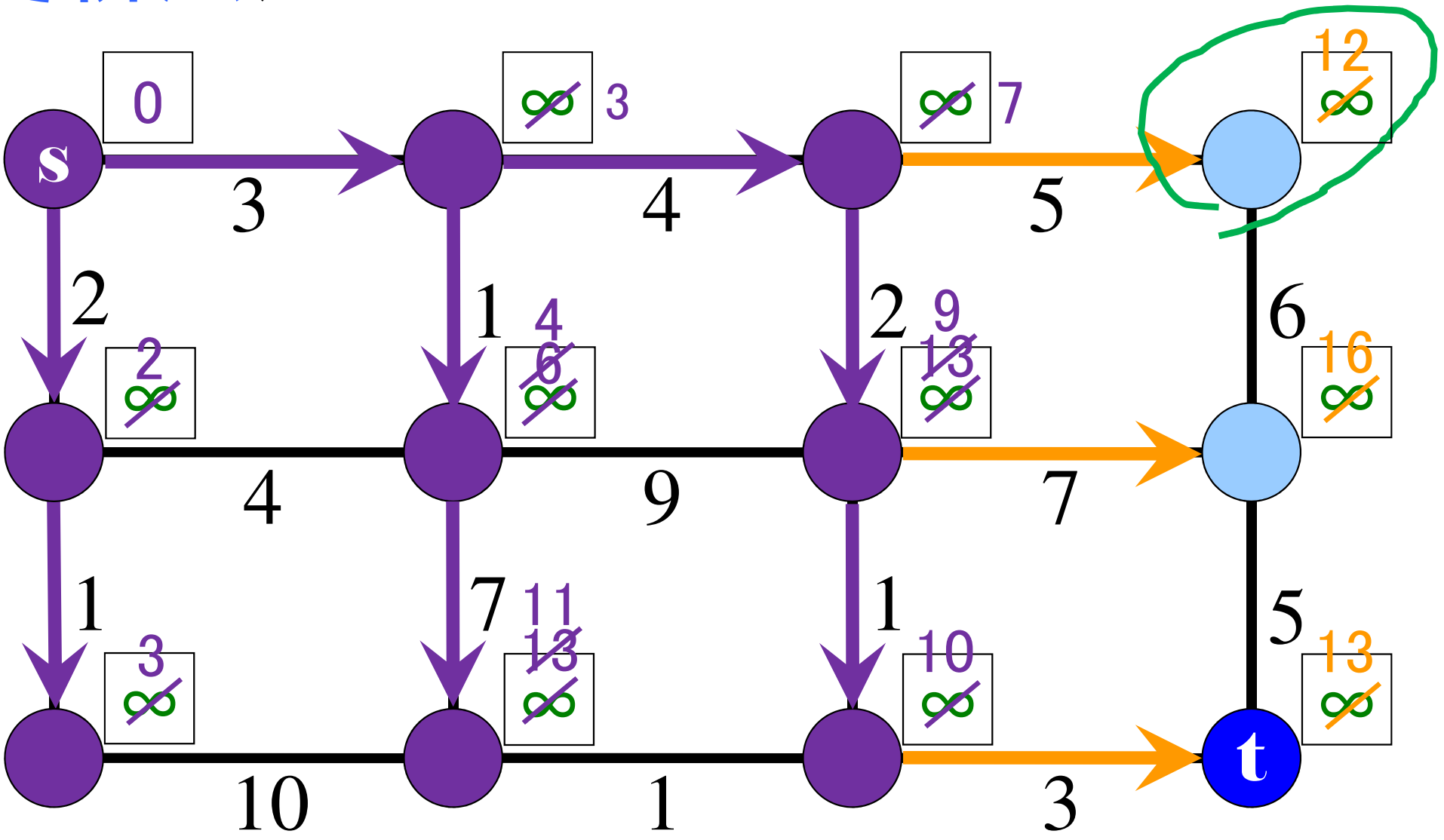
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



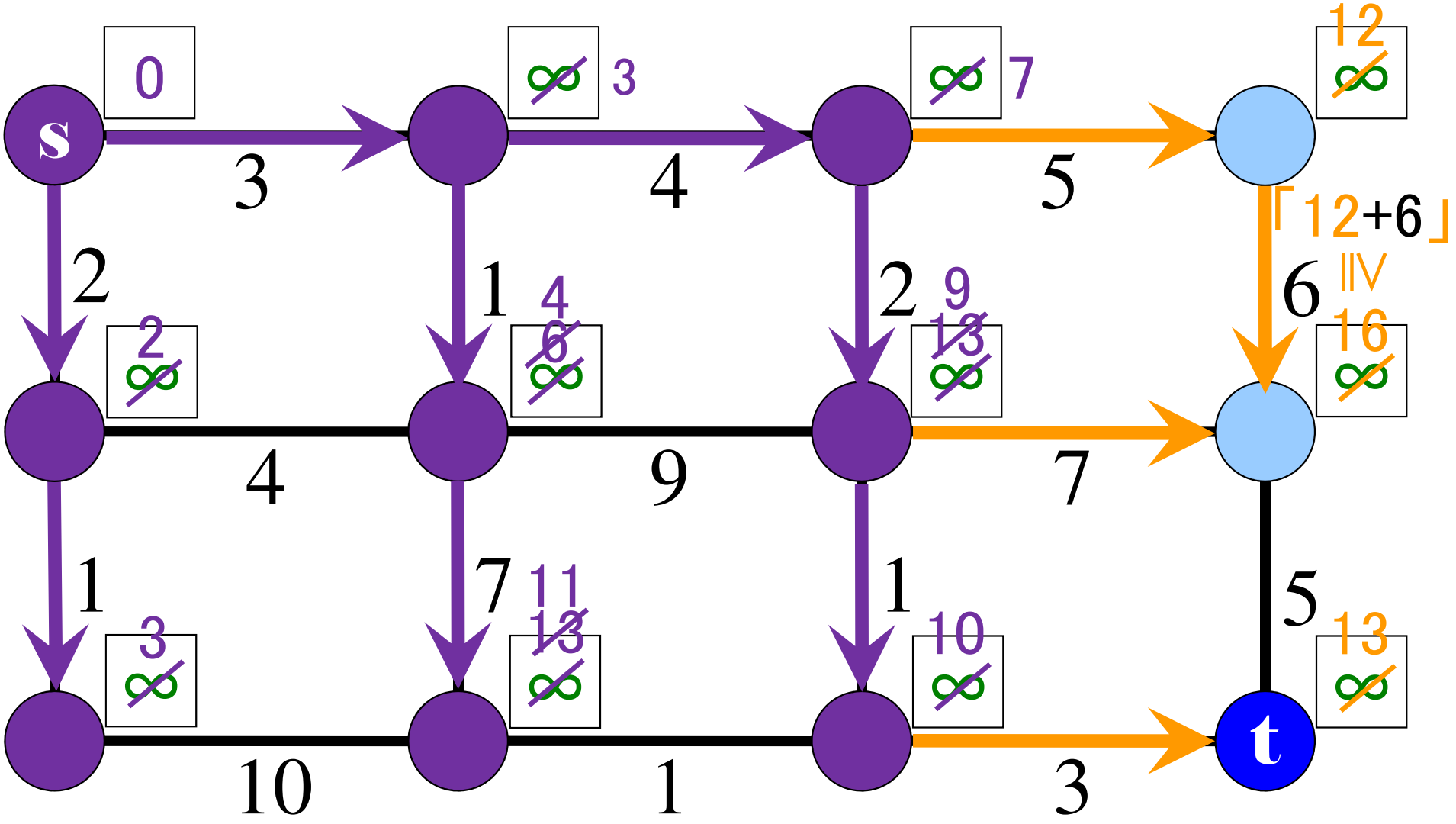
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

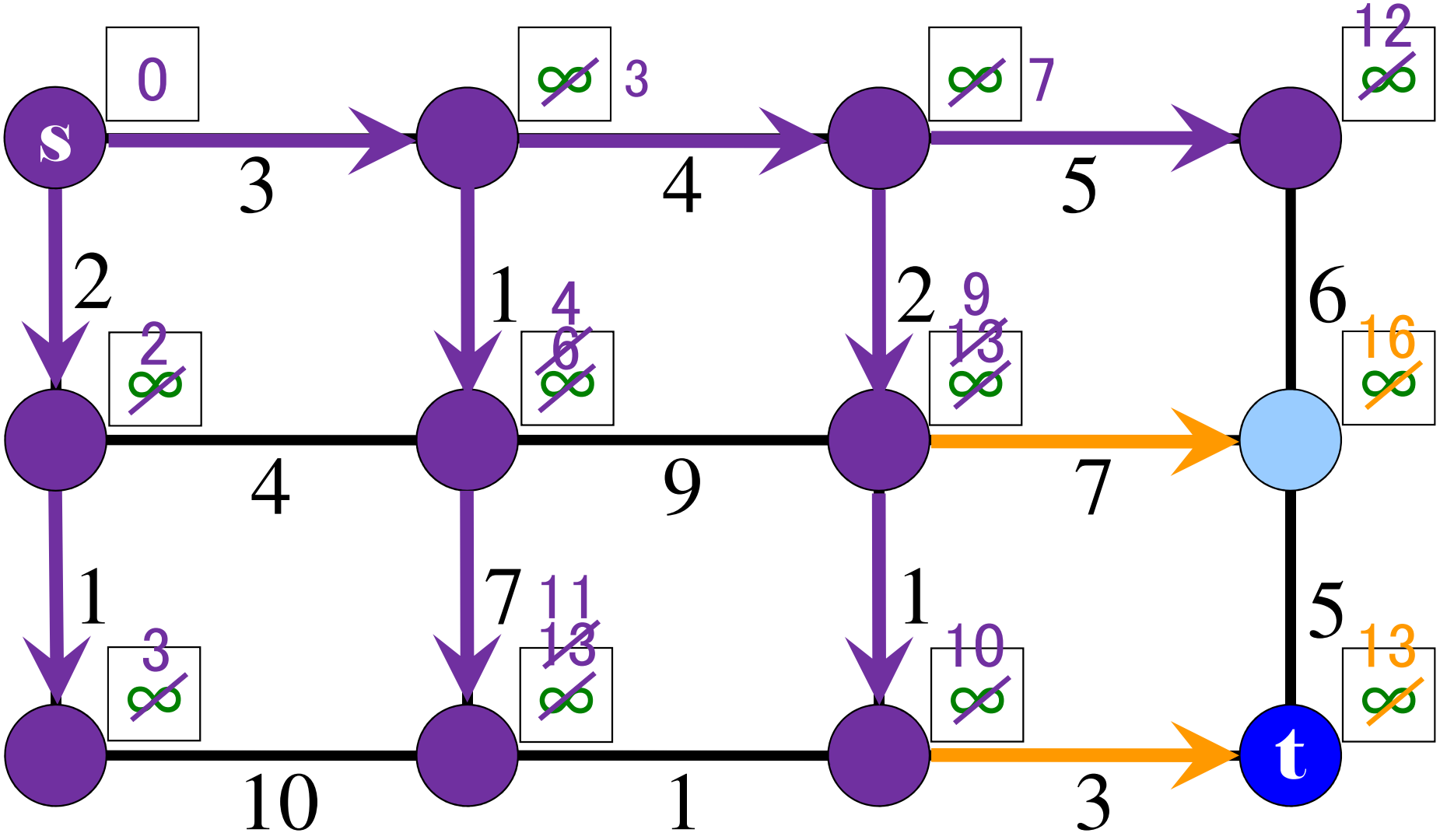


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

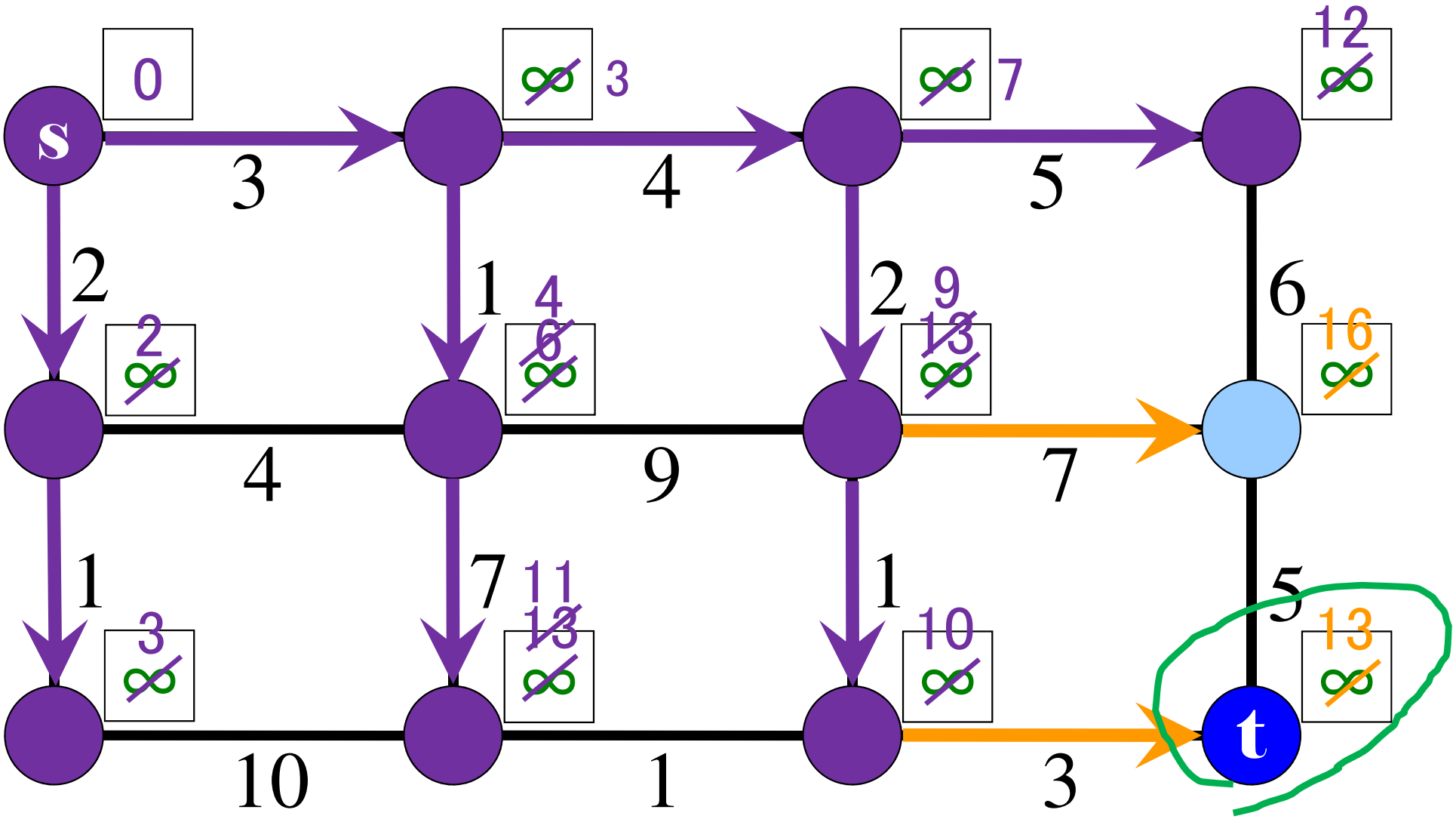
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



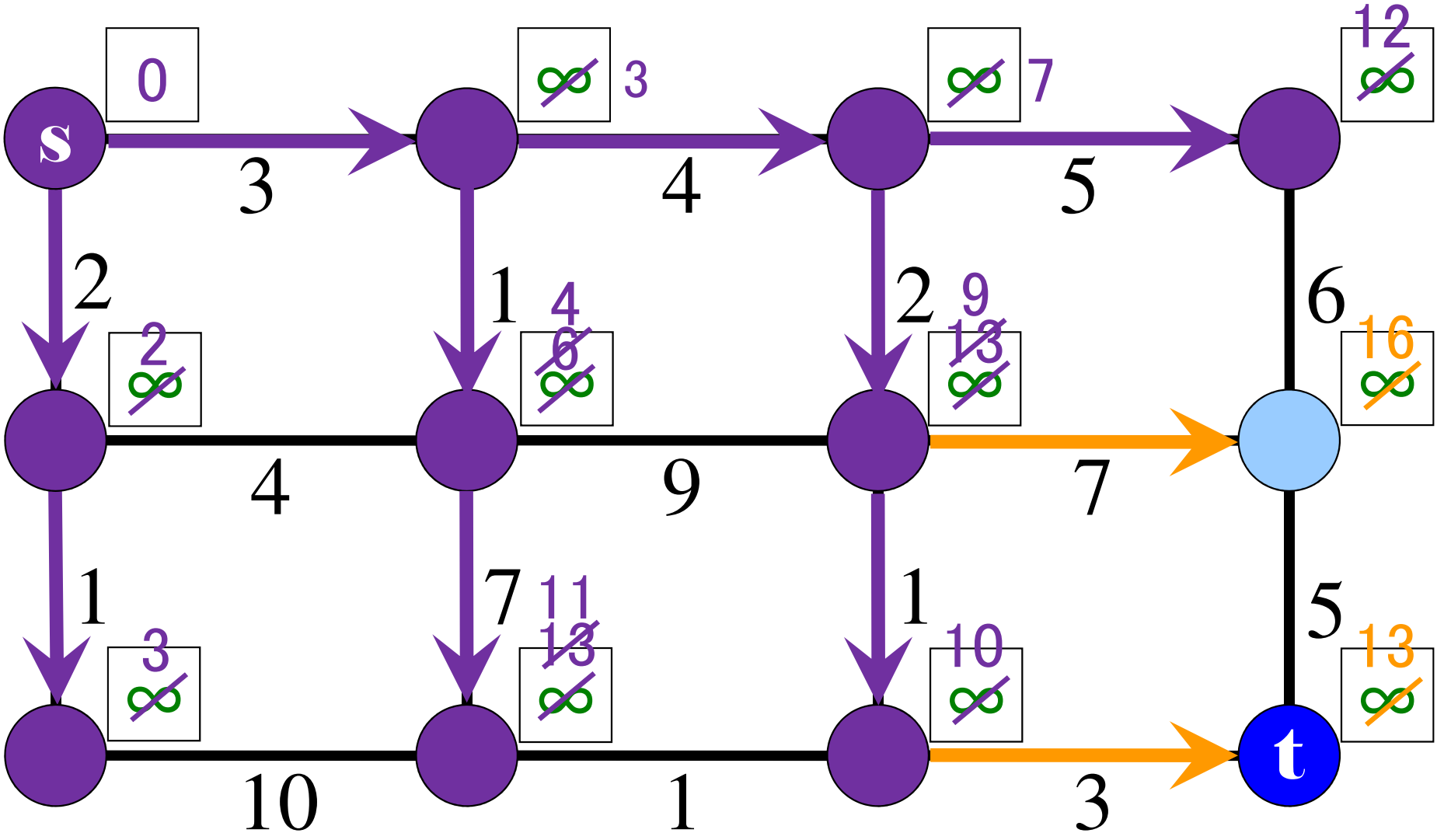
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

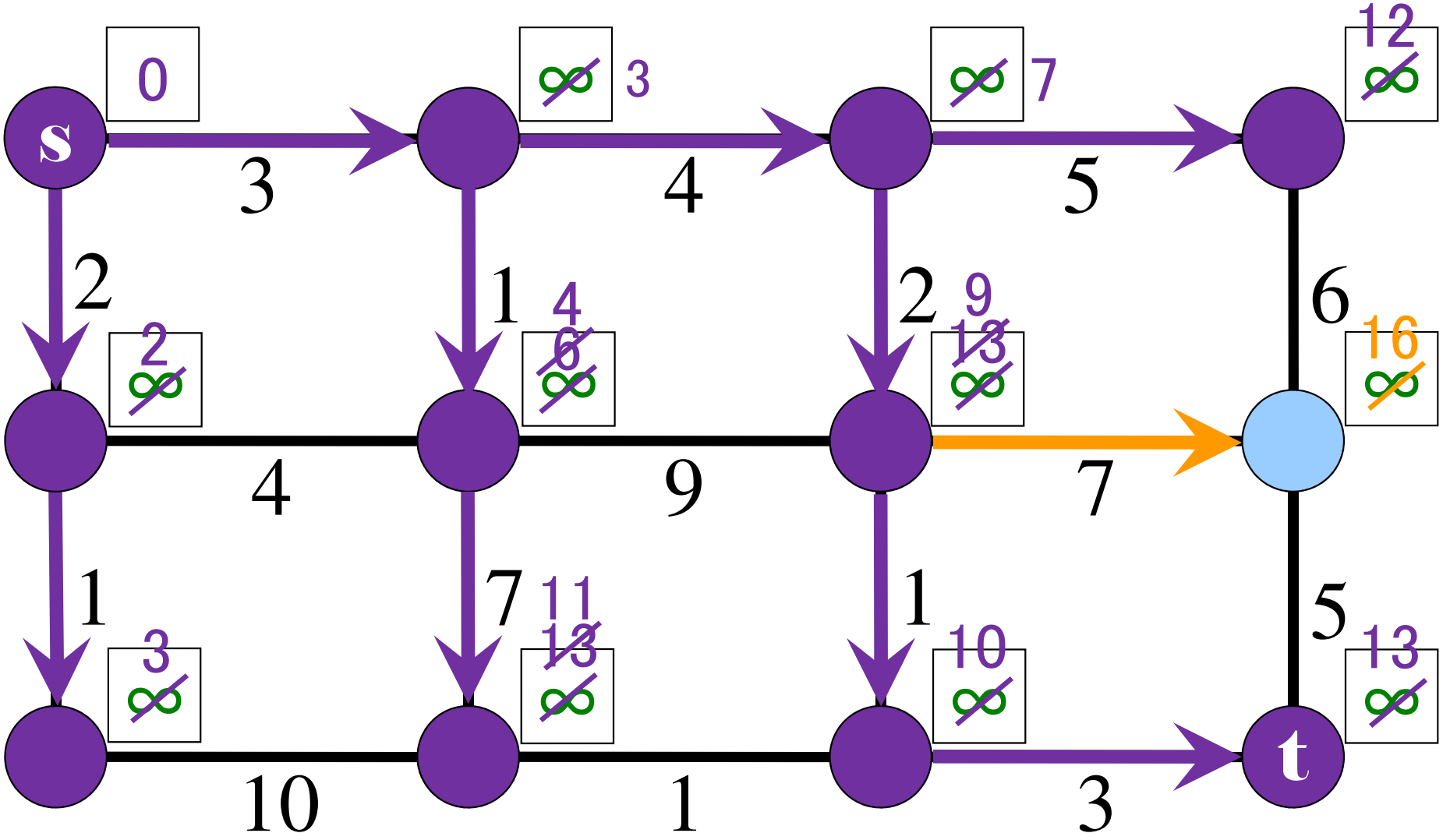


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

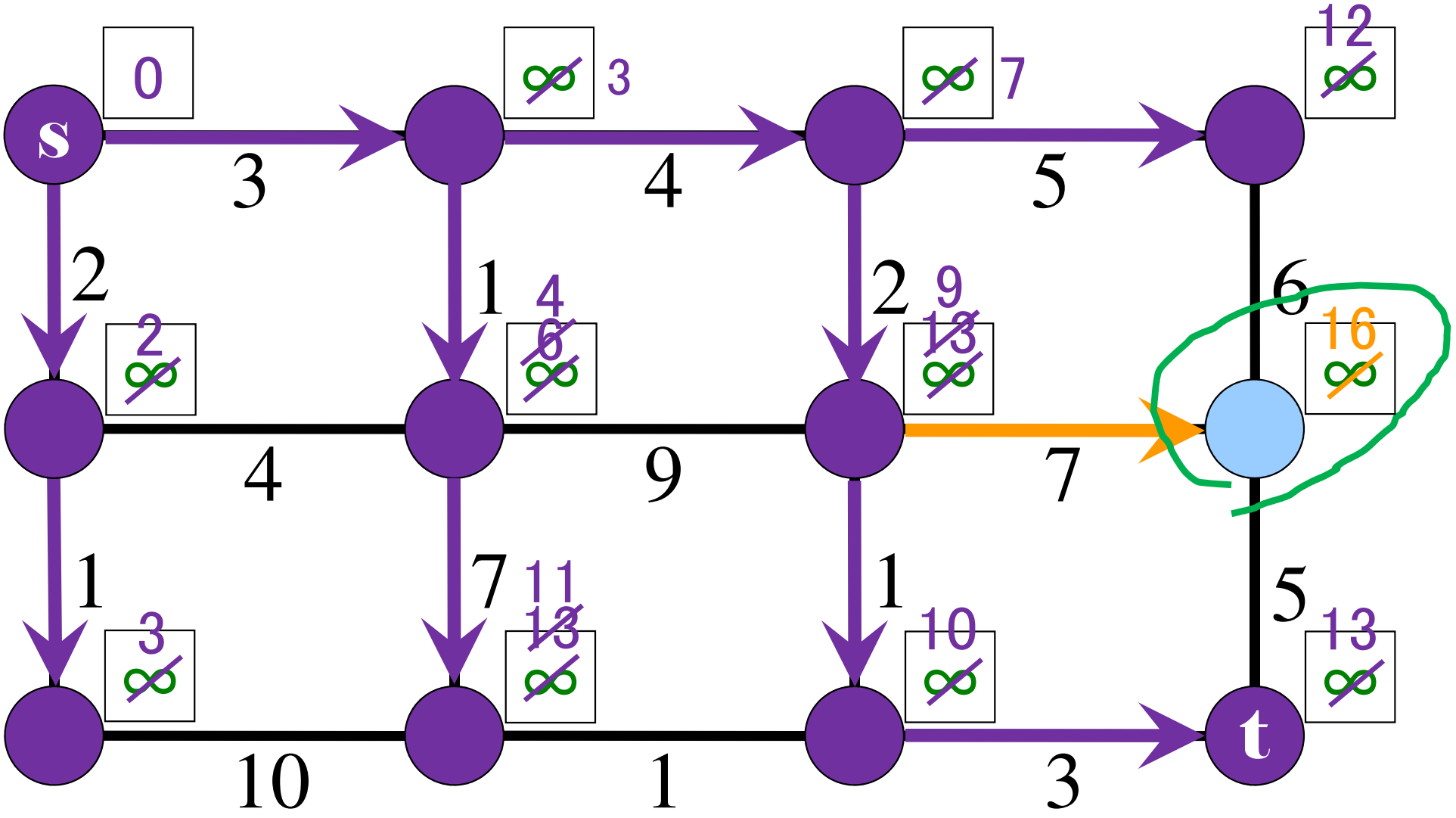
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



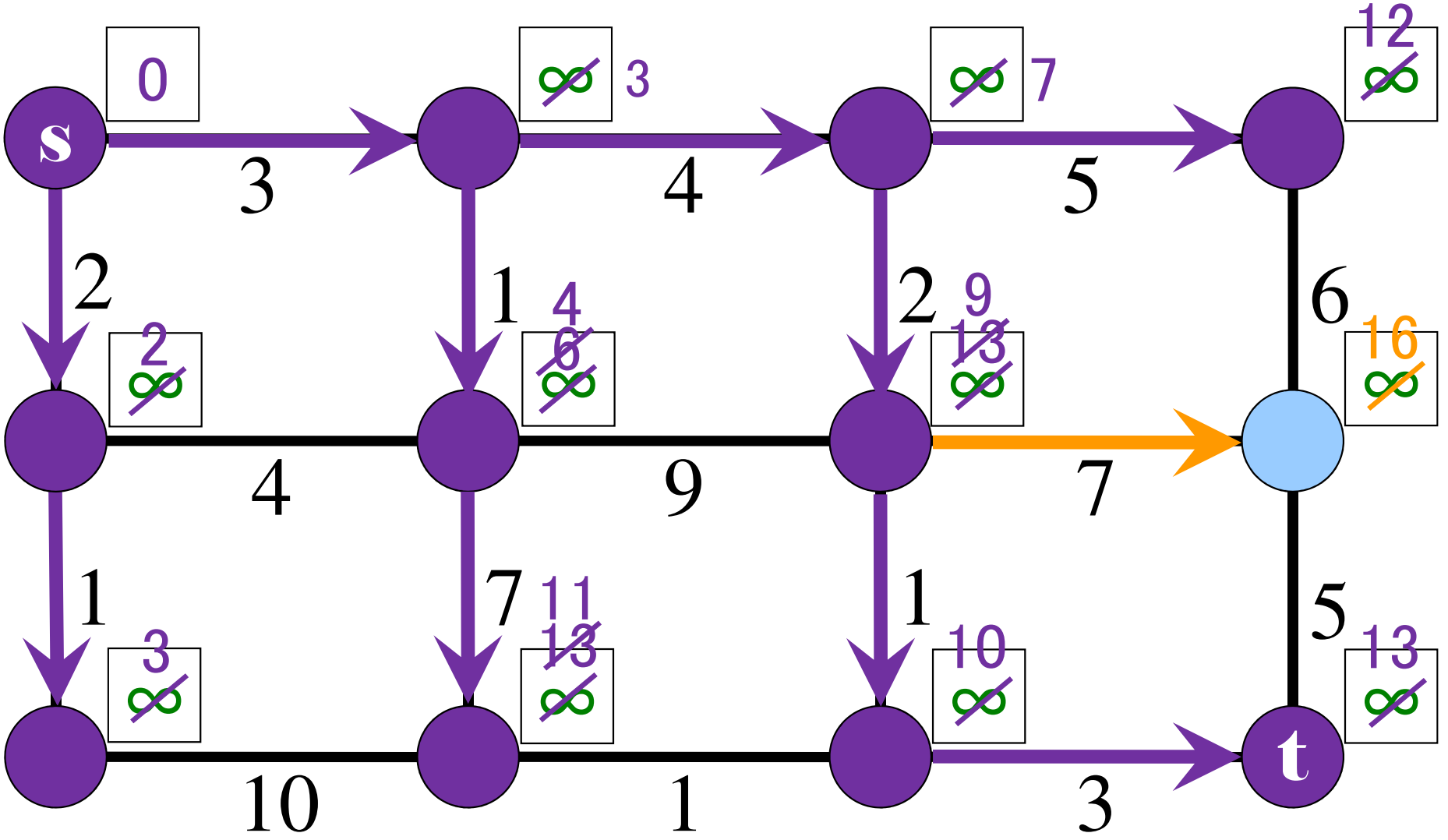
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

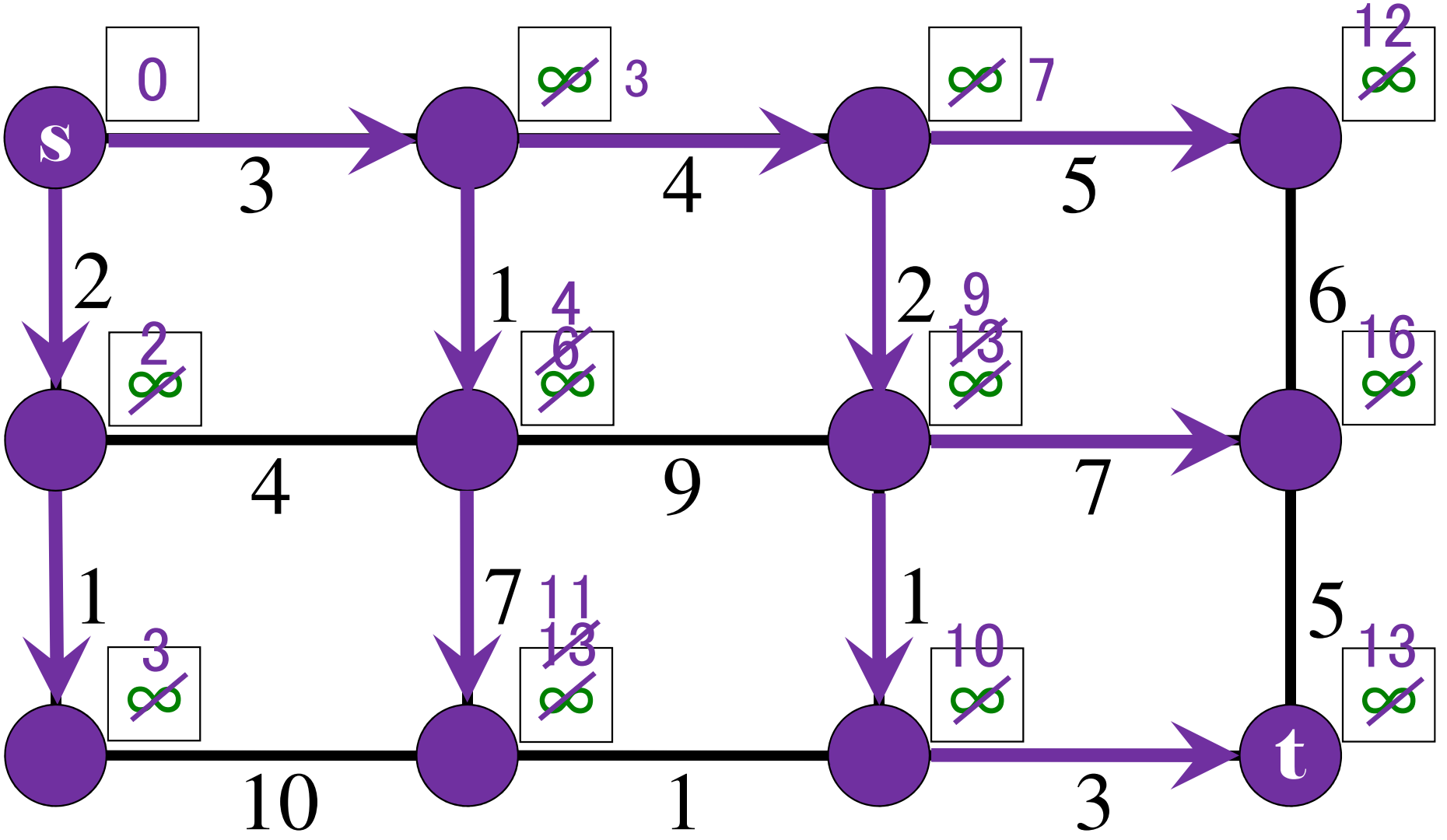


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

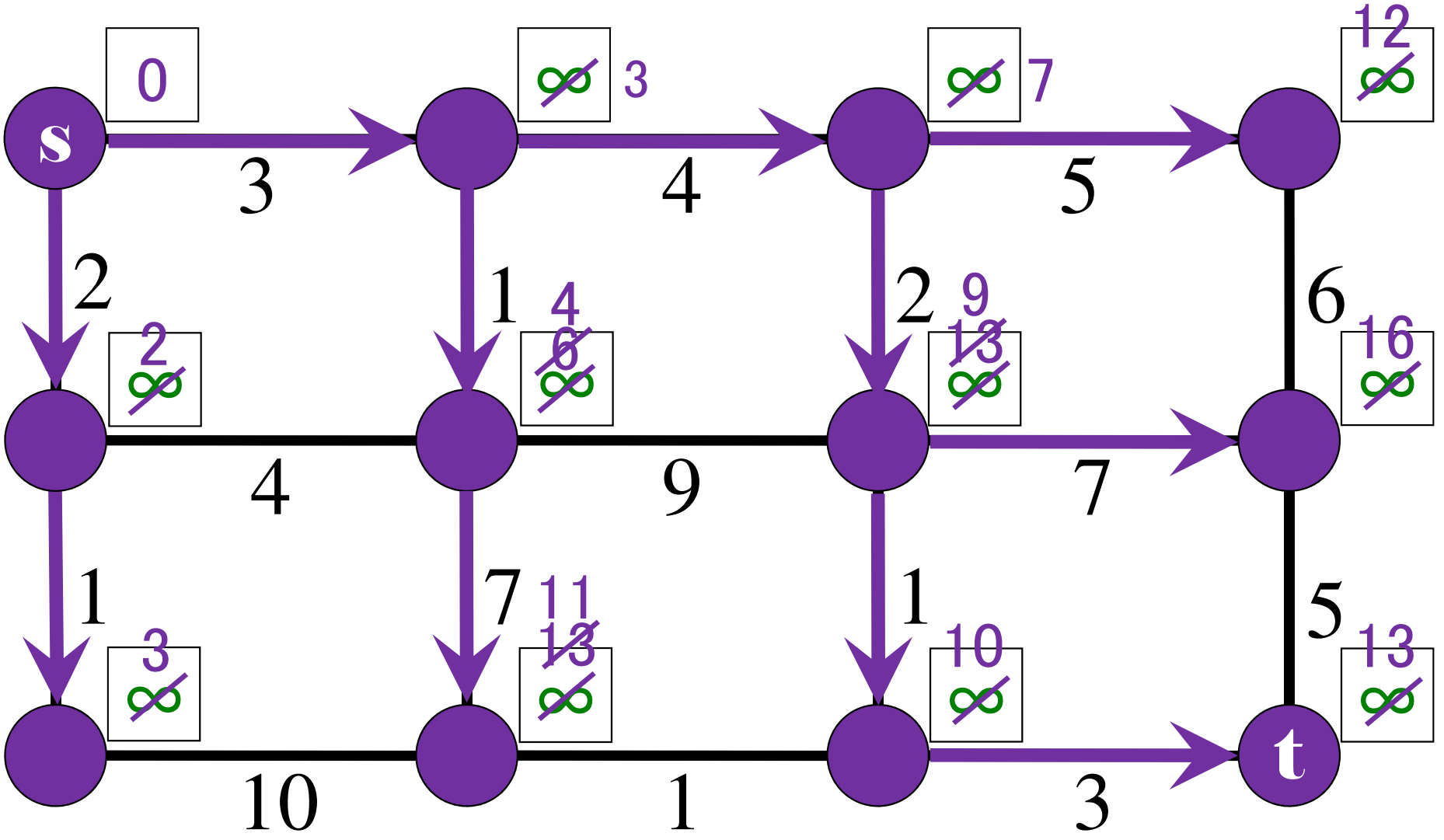
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



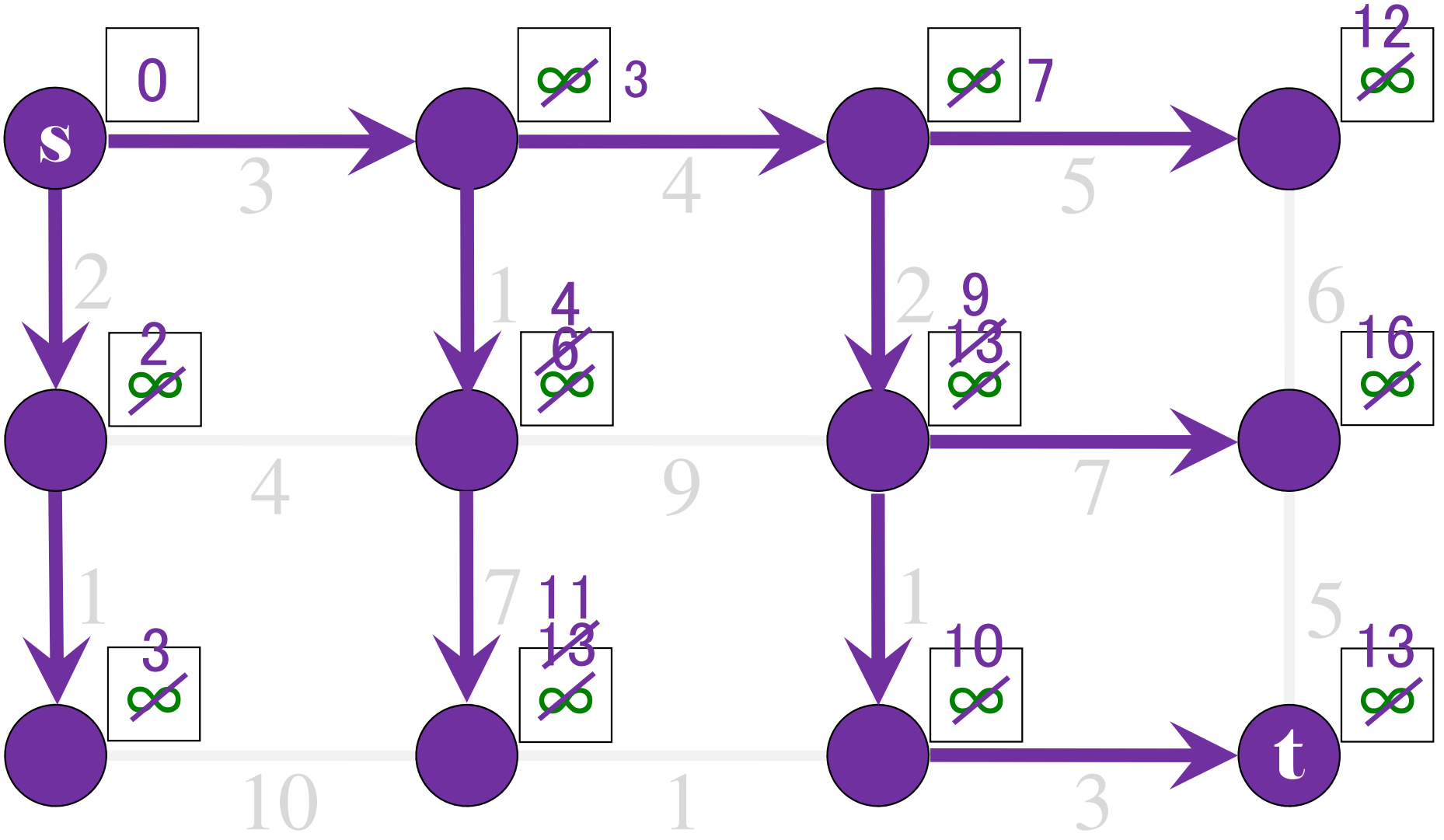
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



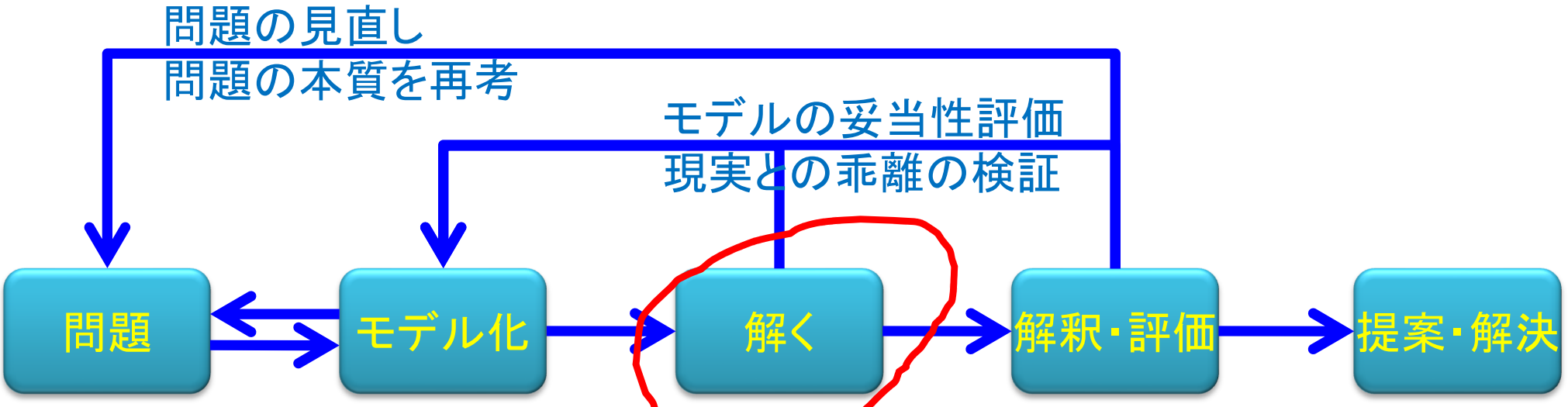
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

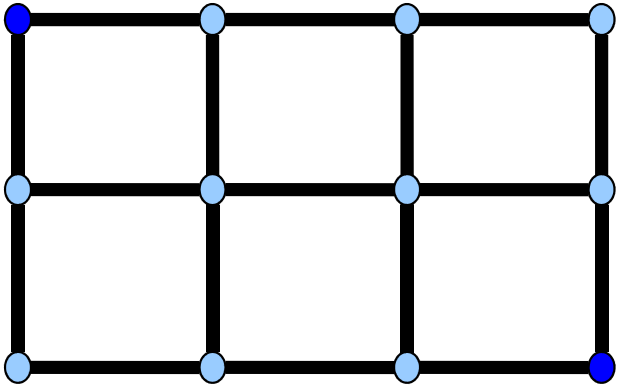
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

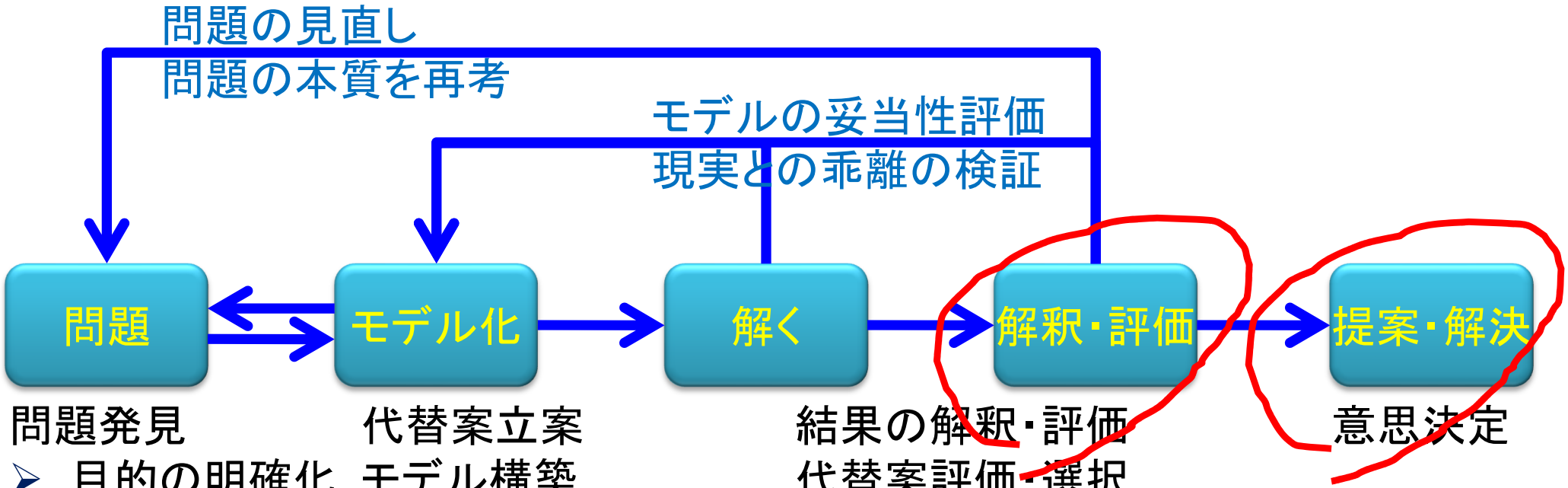
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

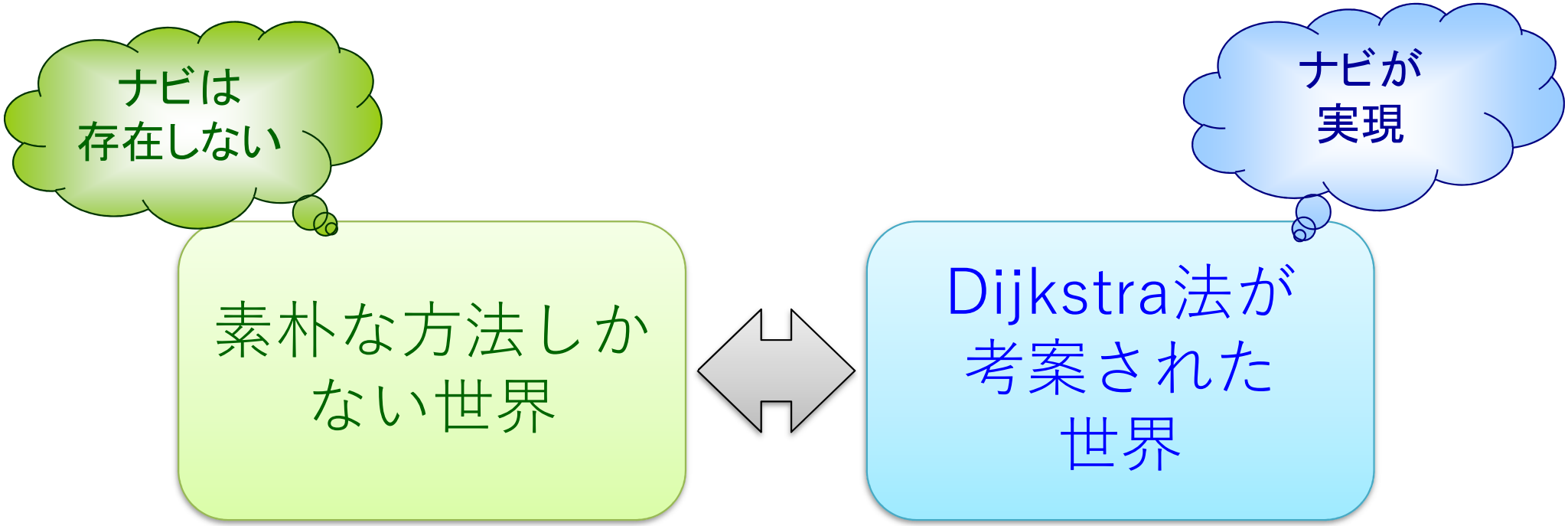
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

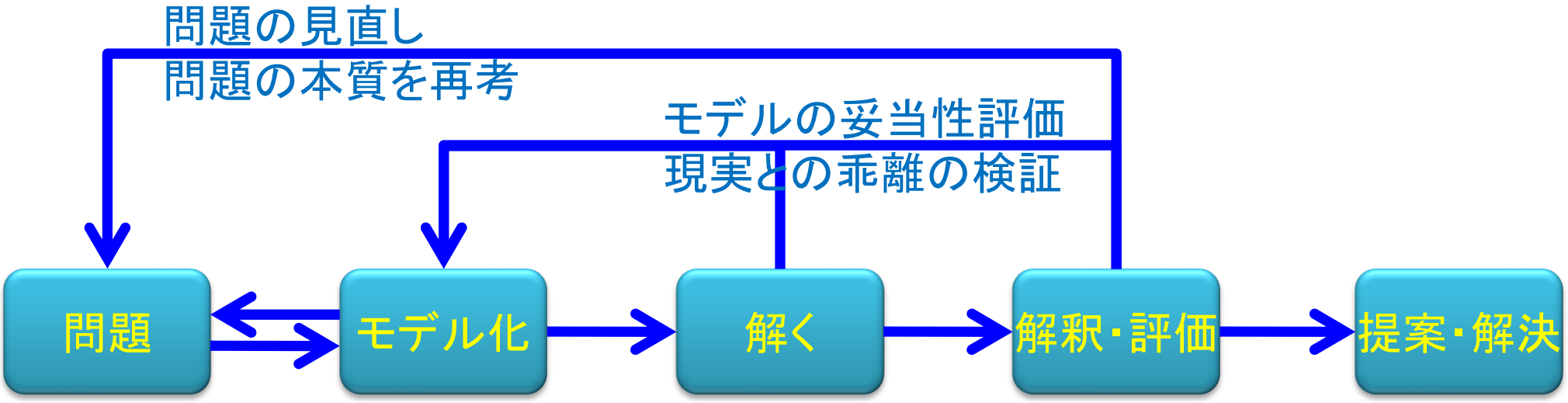
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



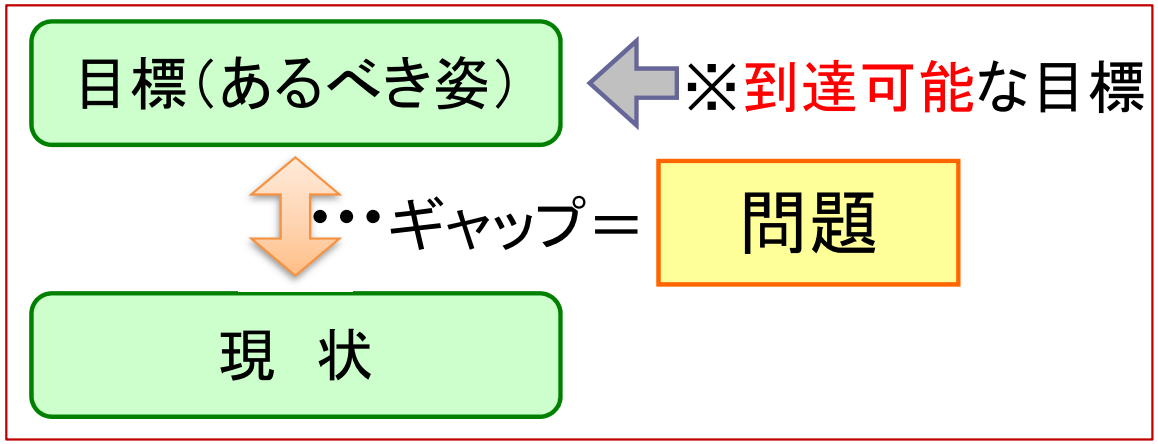
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

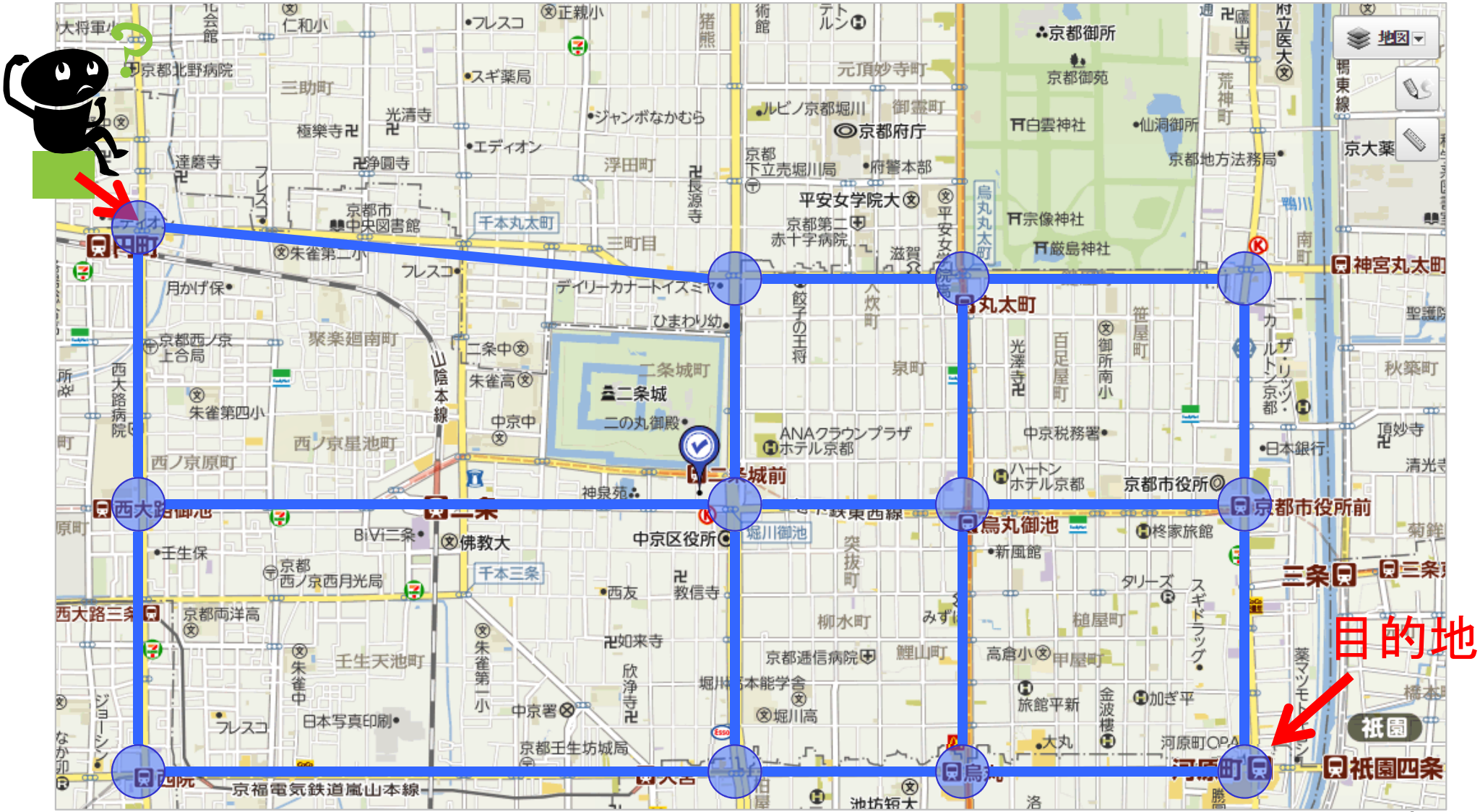
意思決定

問題の定義



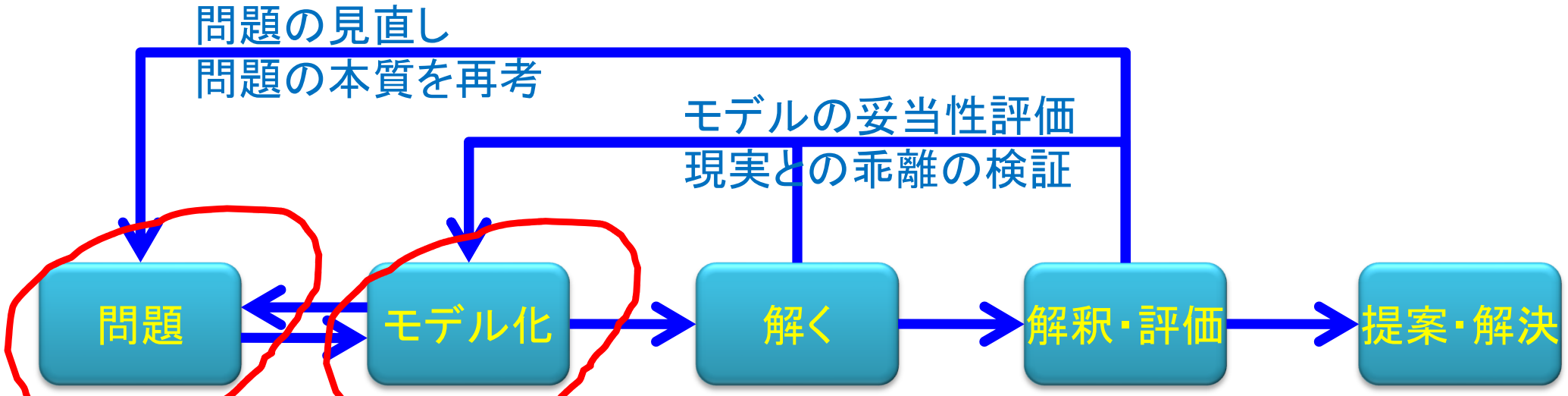
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

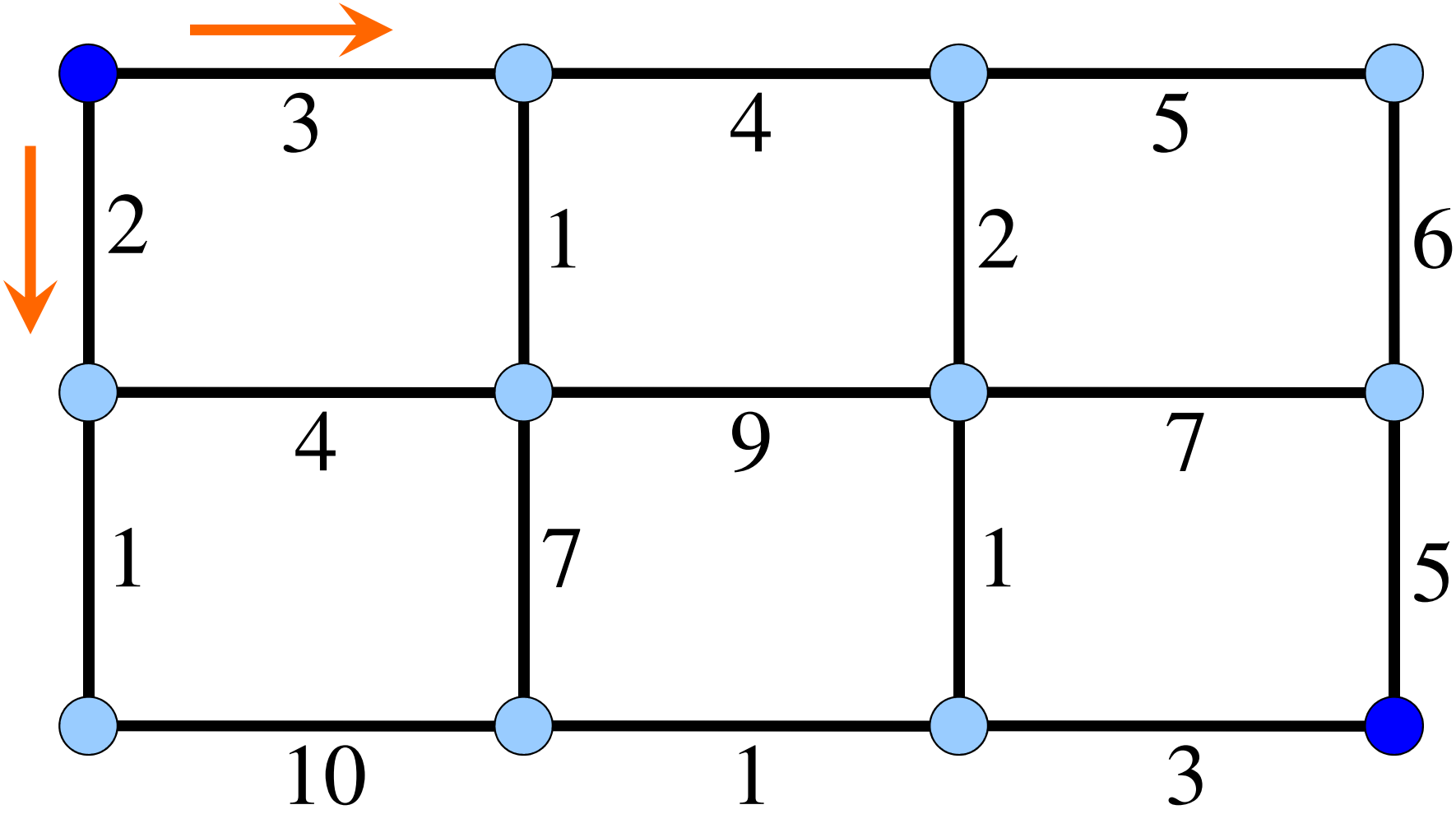
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

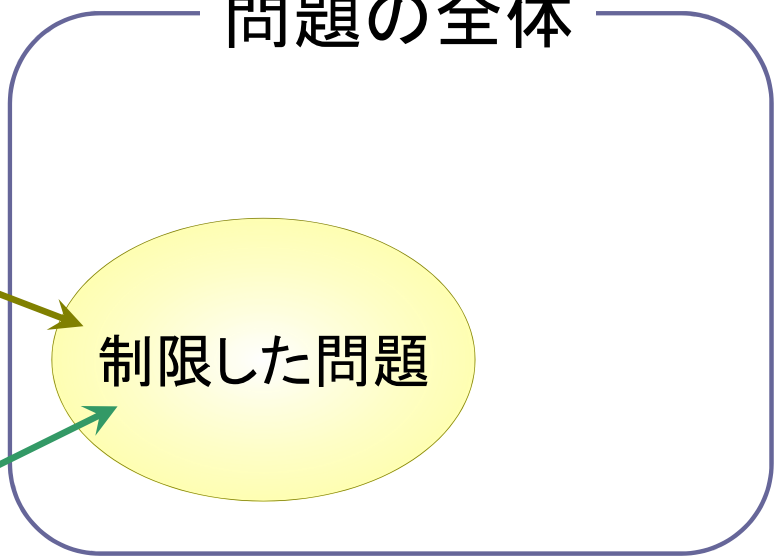


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

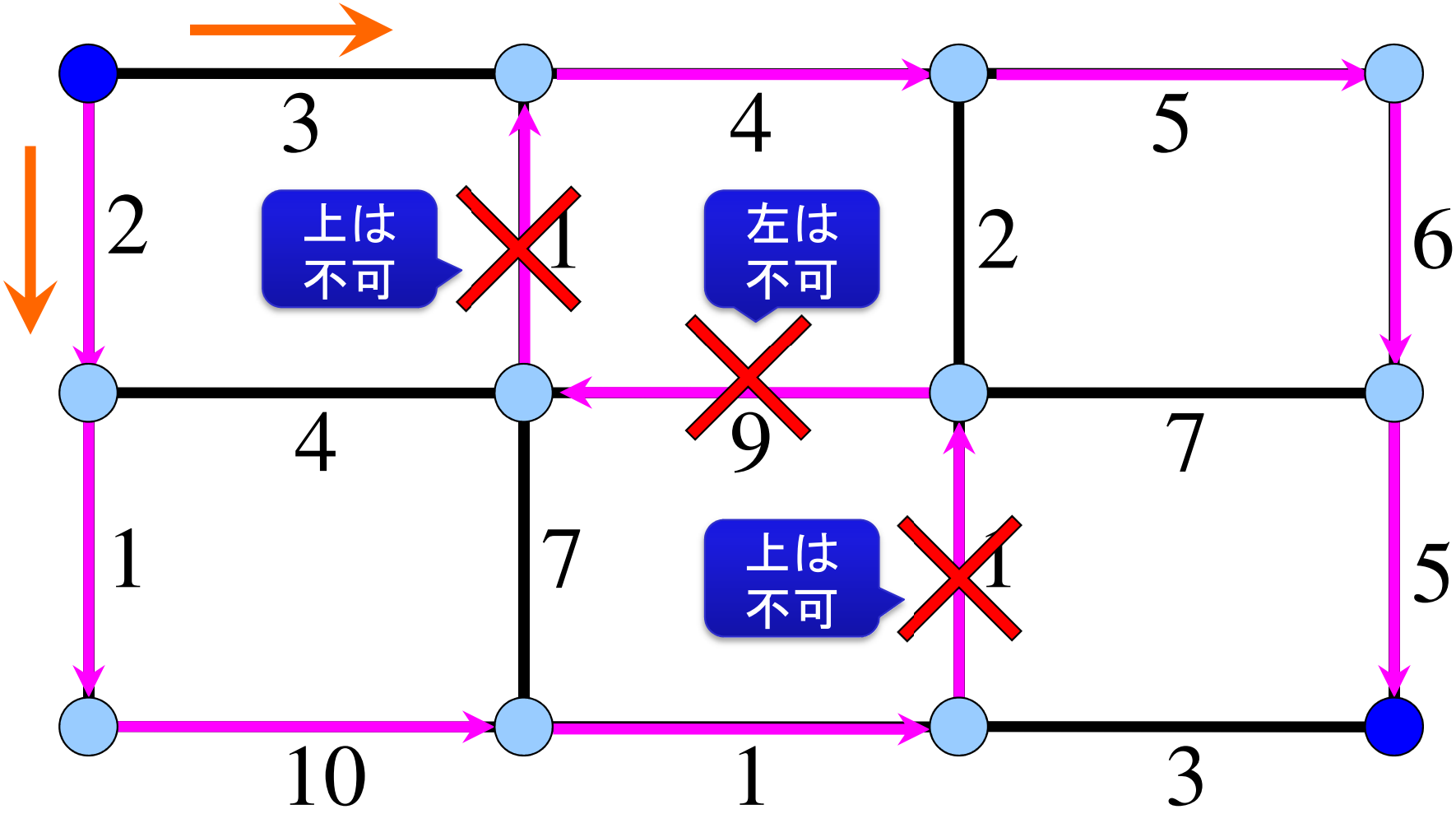
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

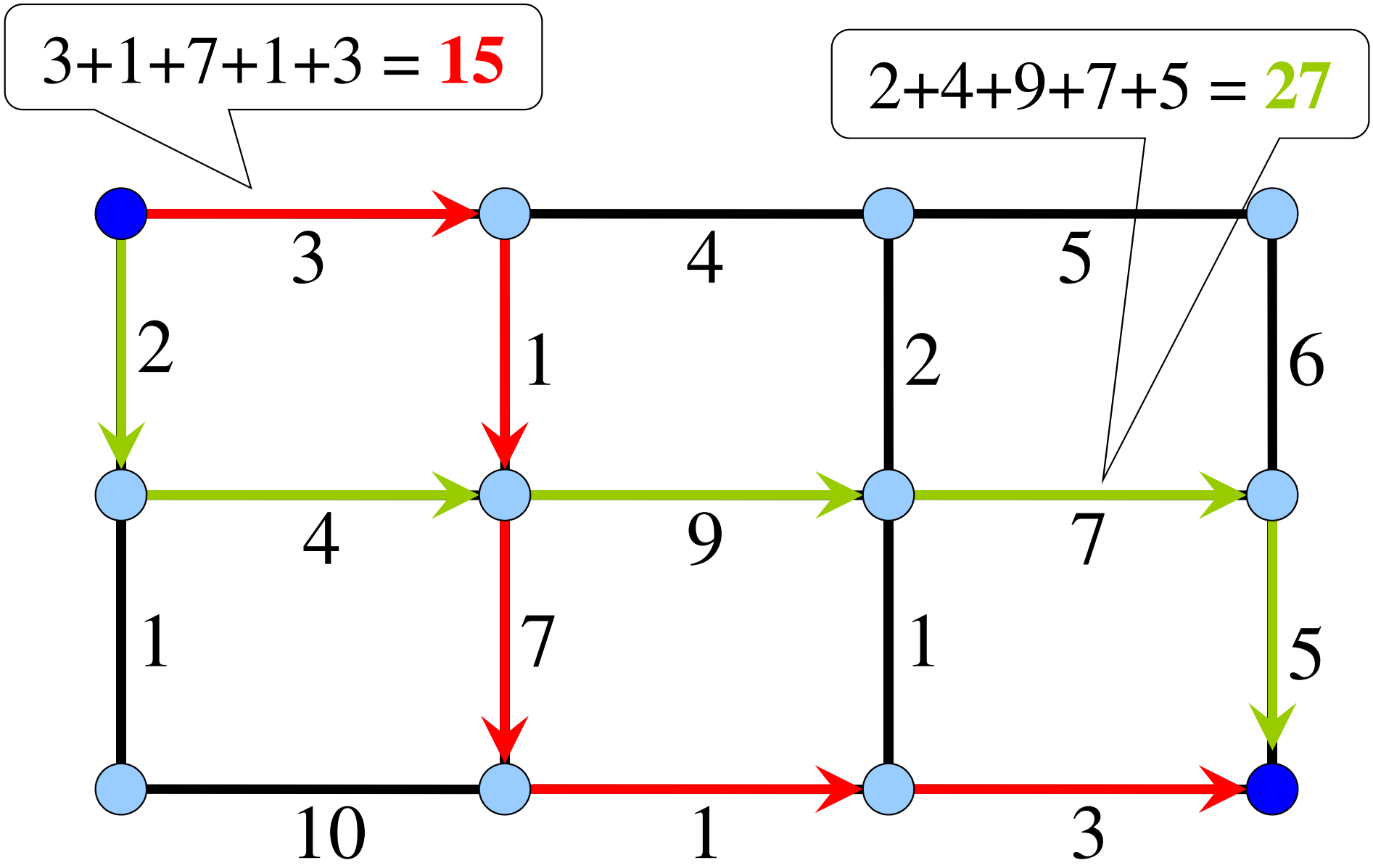
難しいなら易しくすればいいのさ！

制限した問題

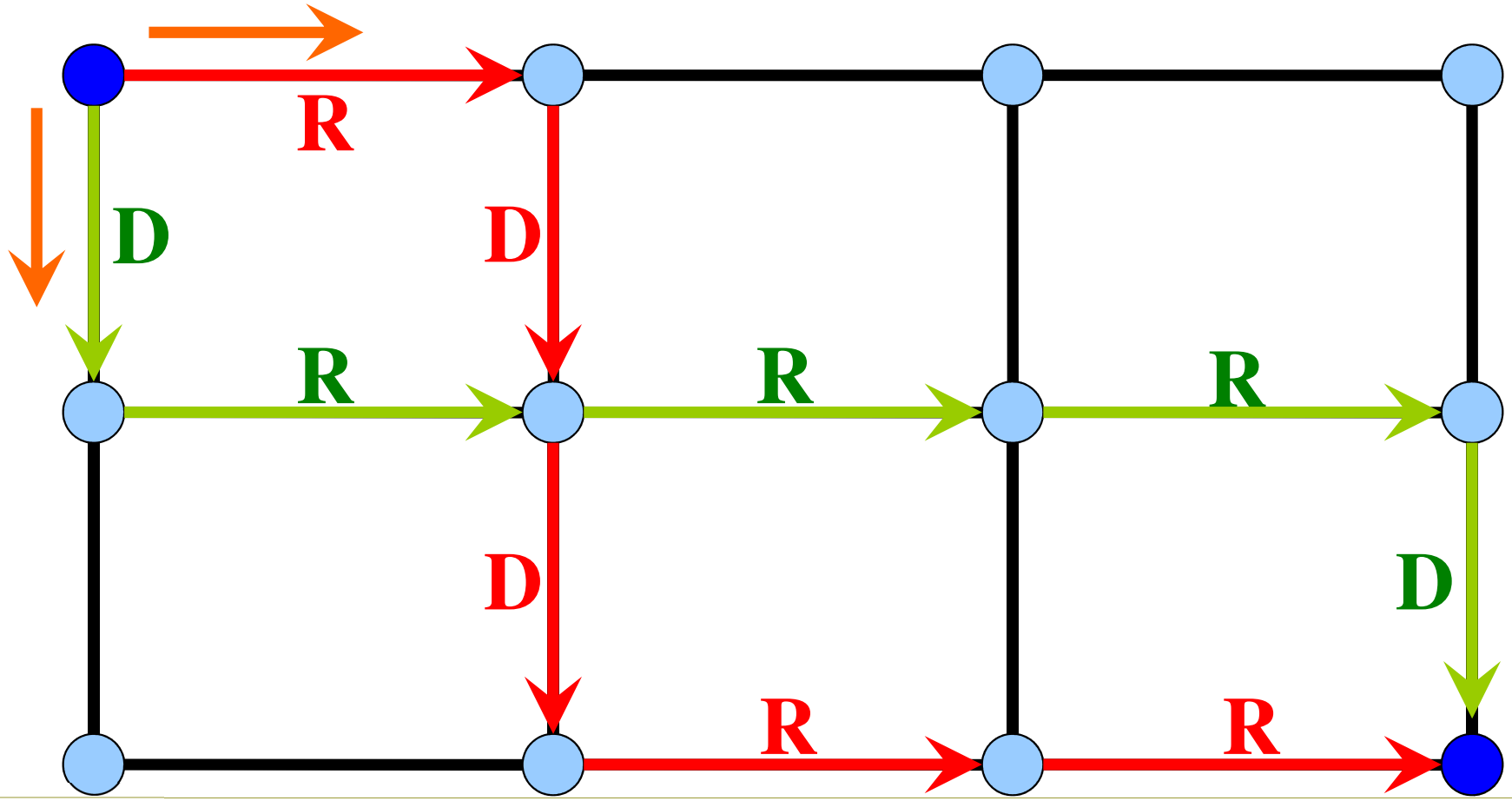
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



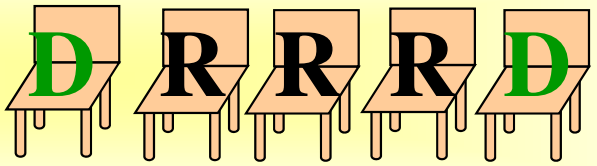
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



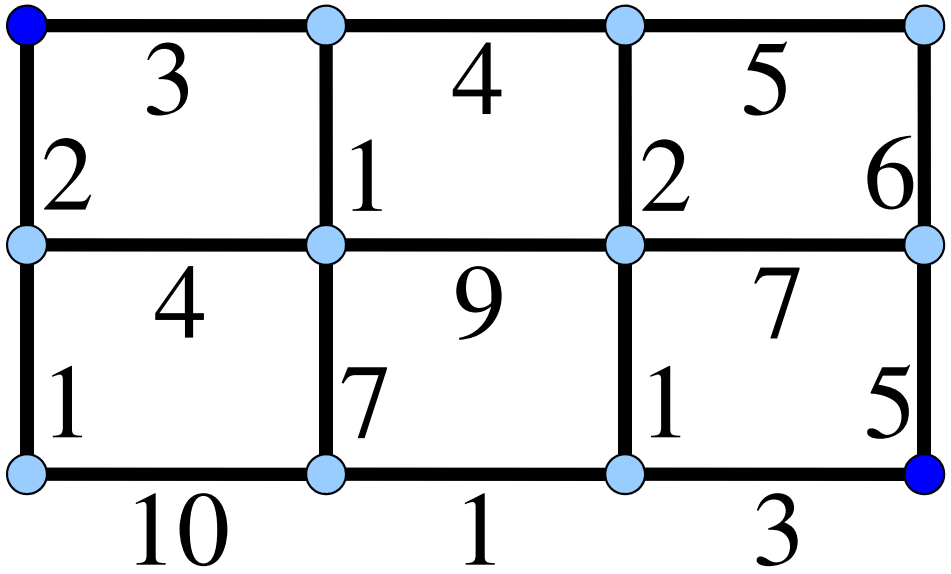
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

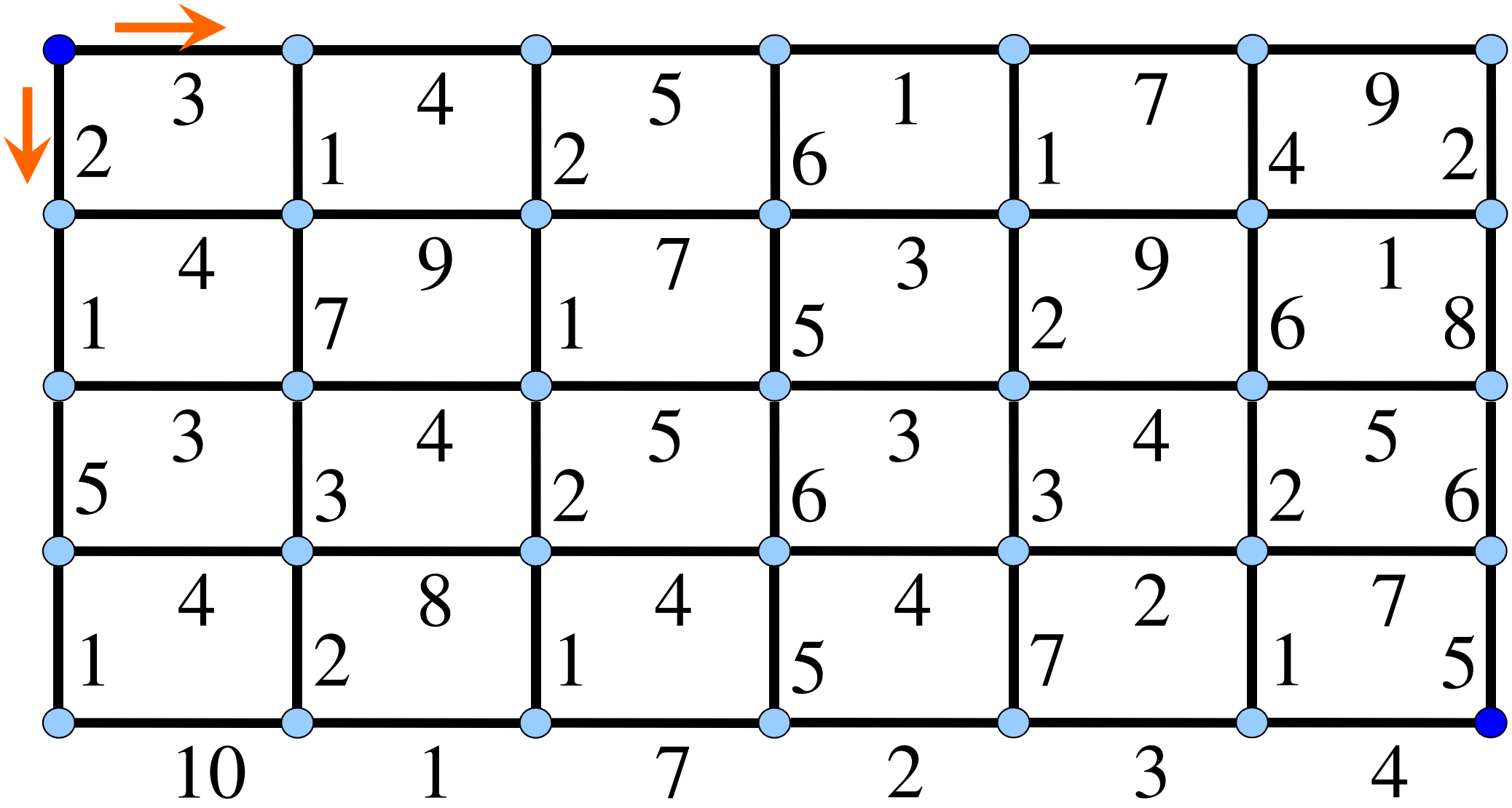
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

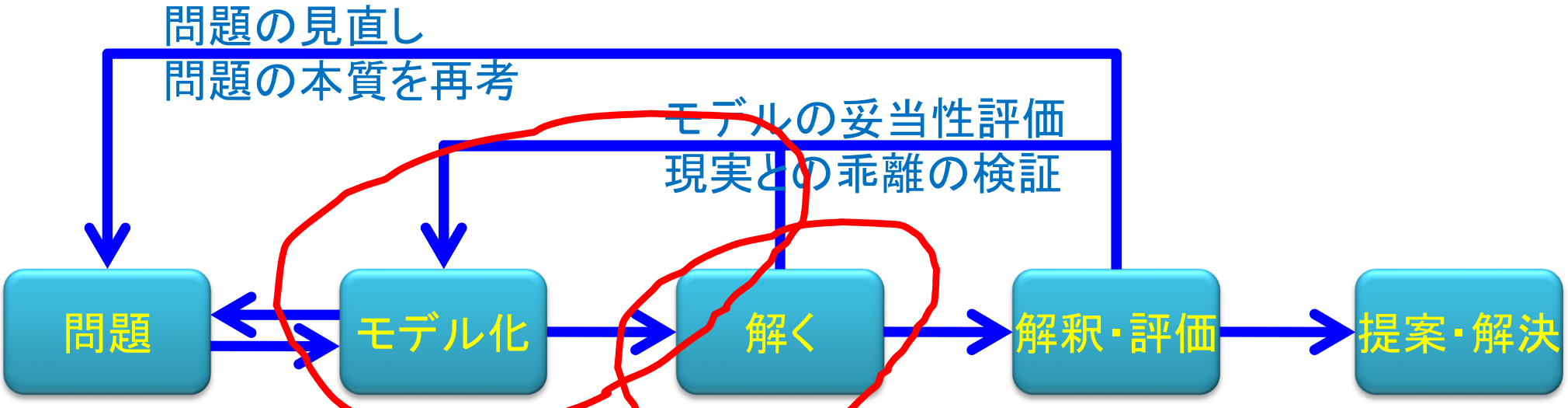
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

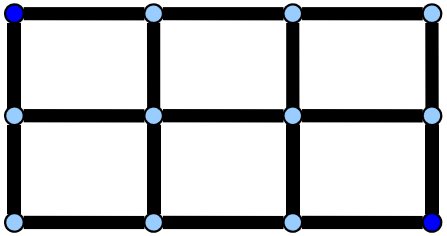
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

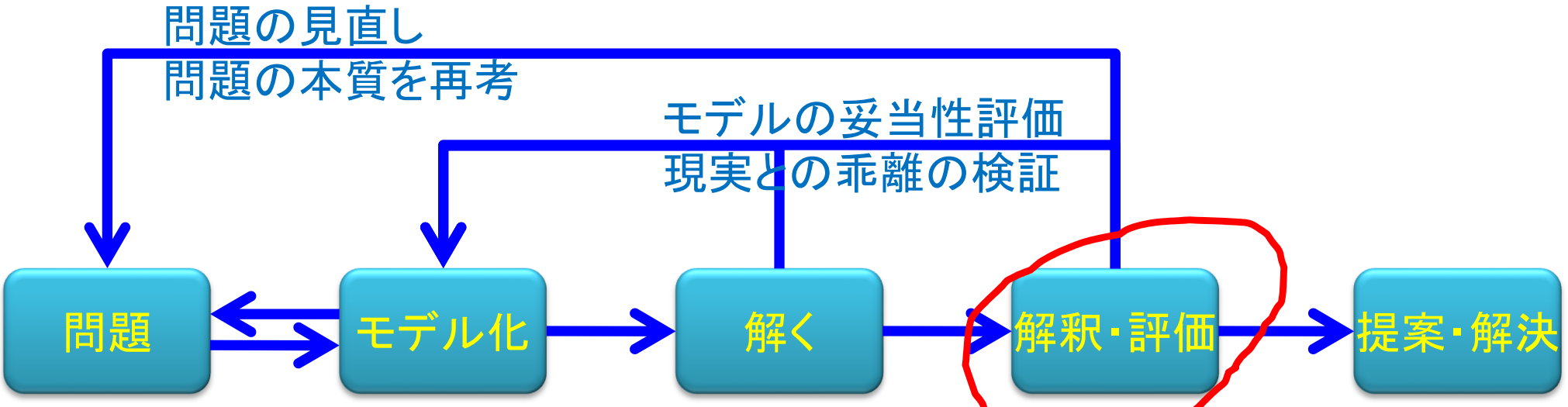
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

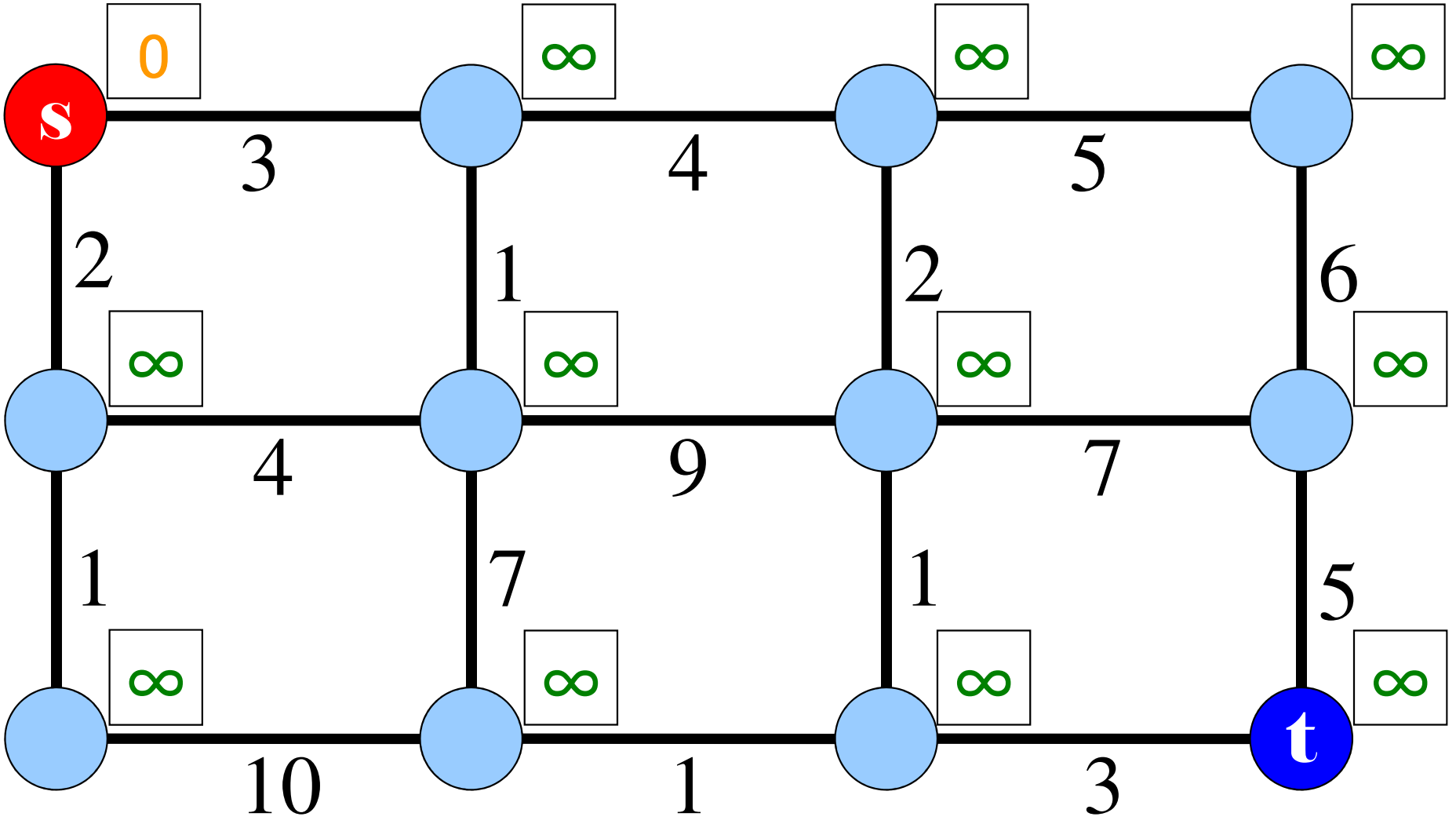
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

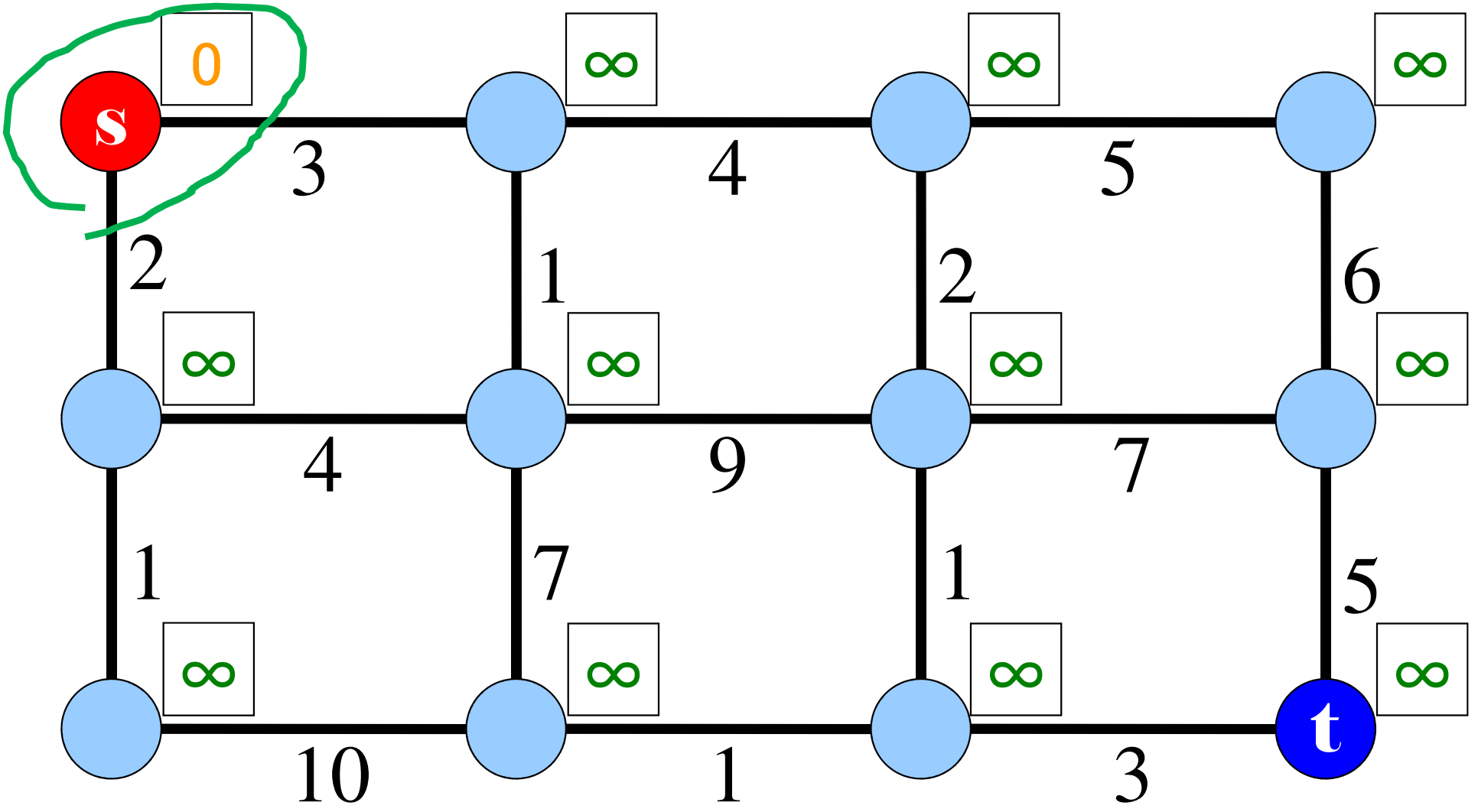
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



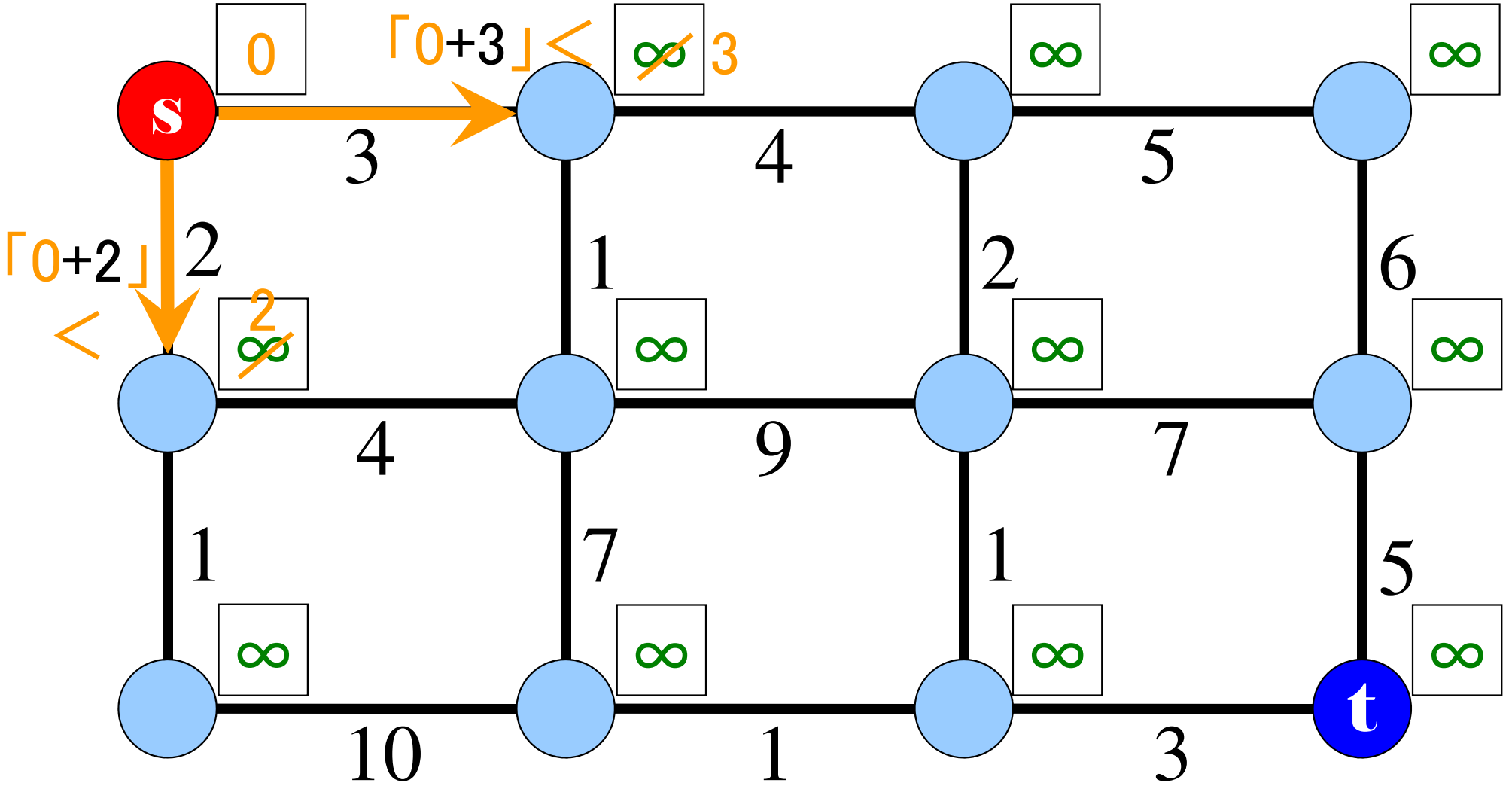
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

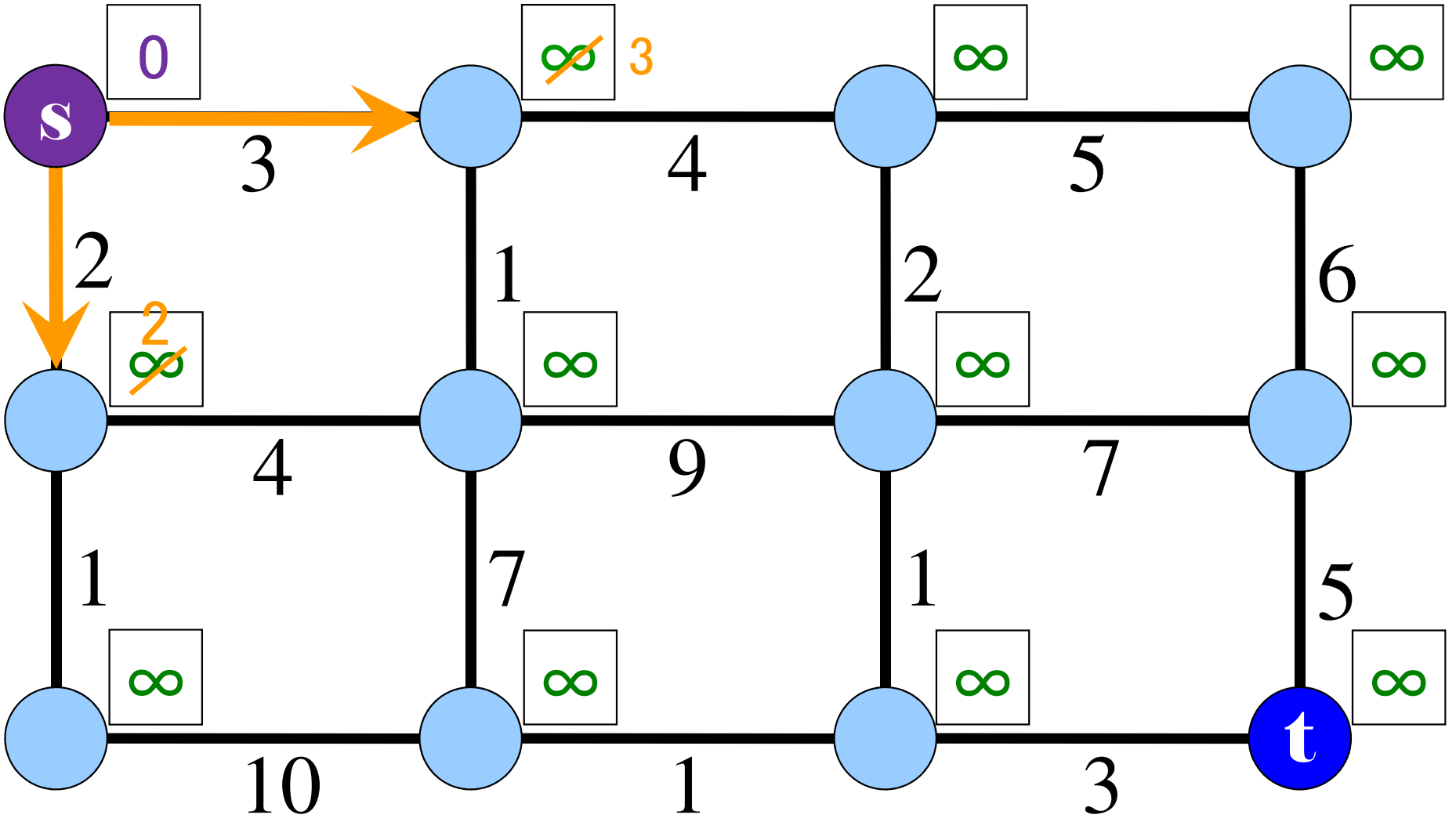


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

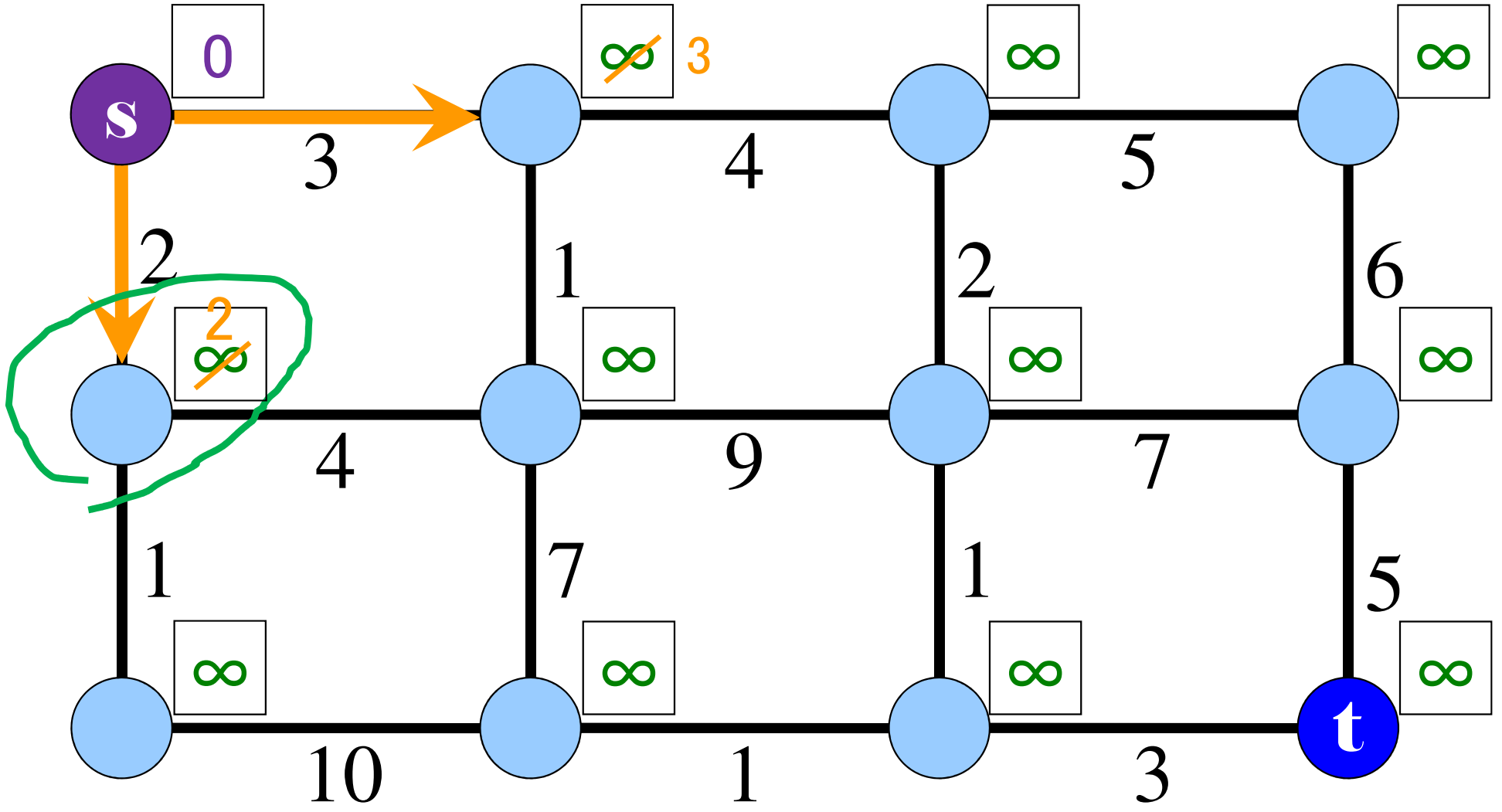
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



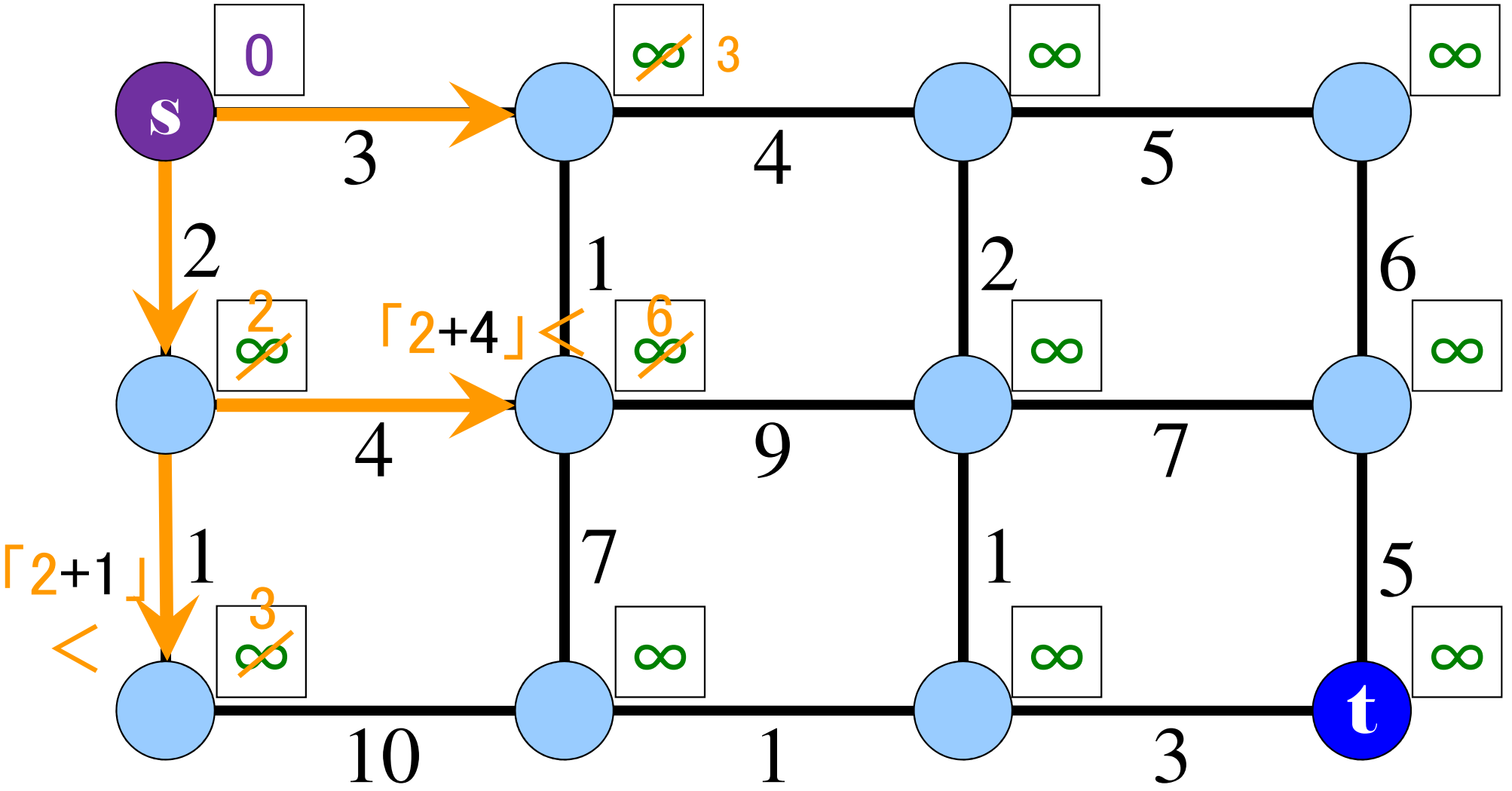
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

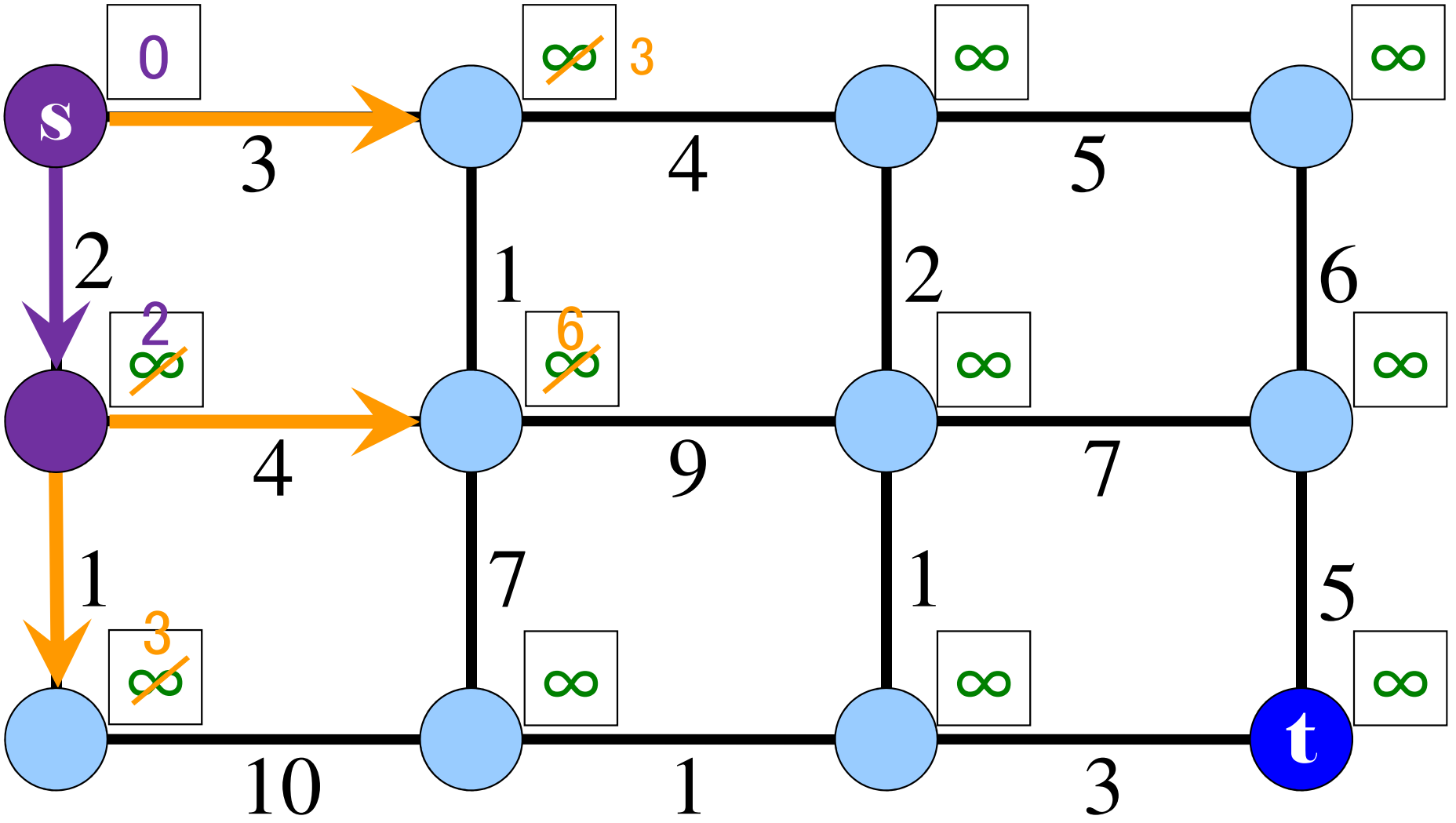


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

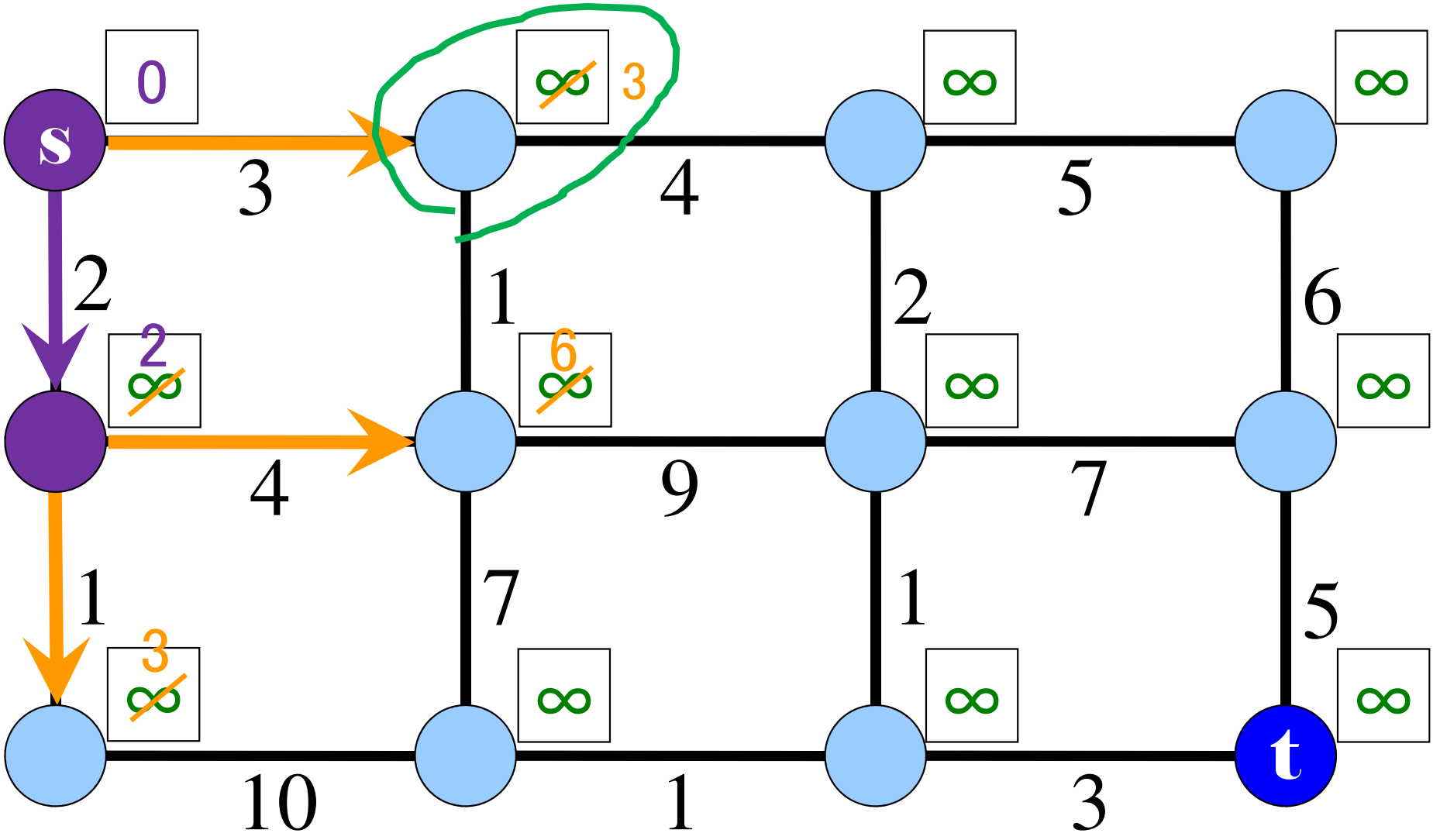
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



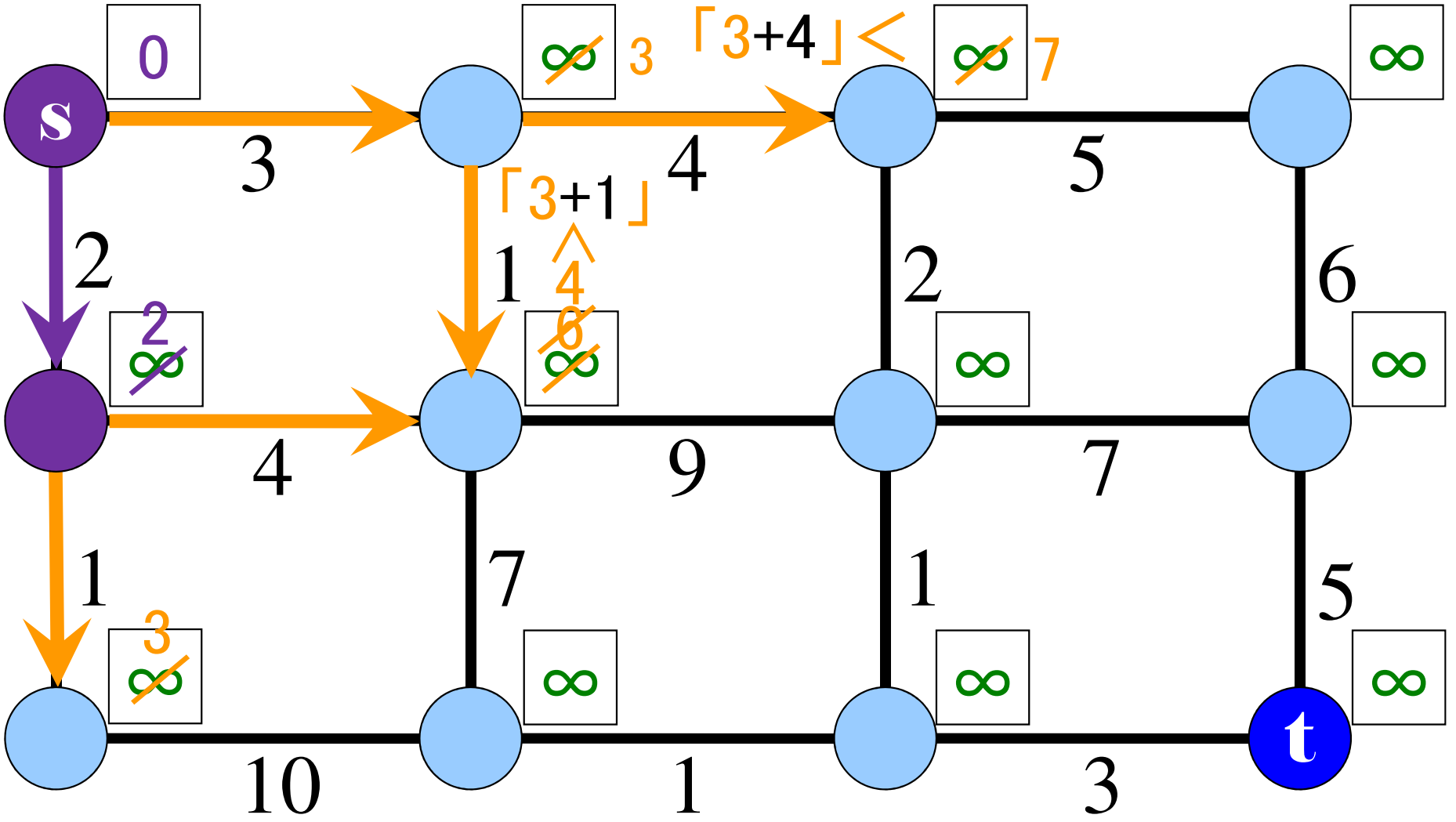
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

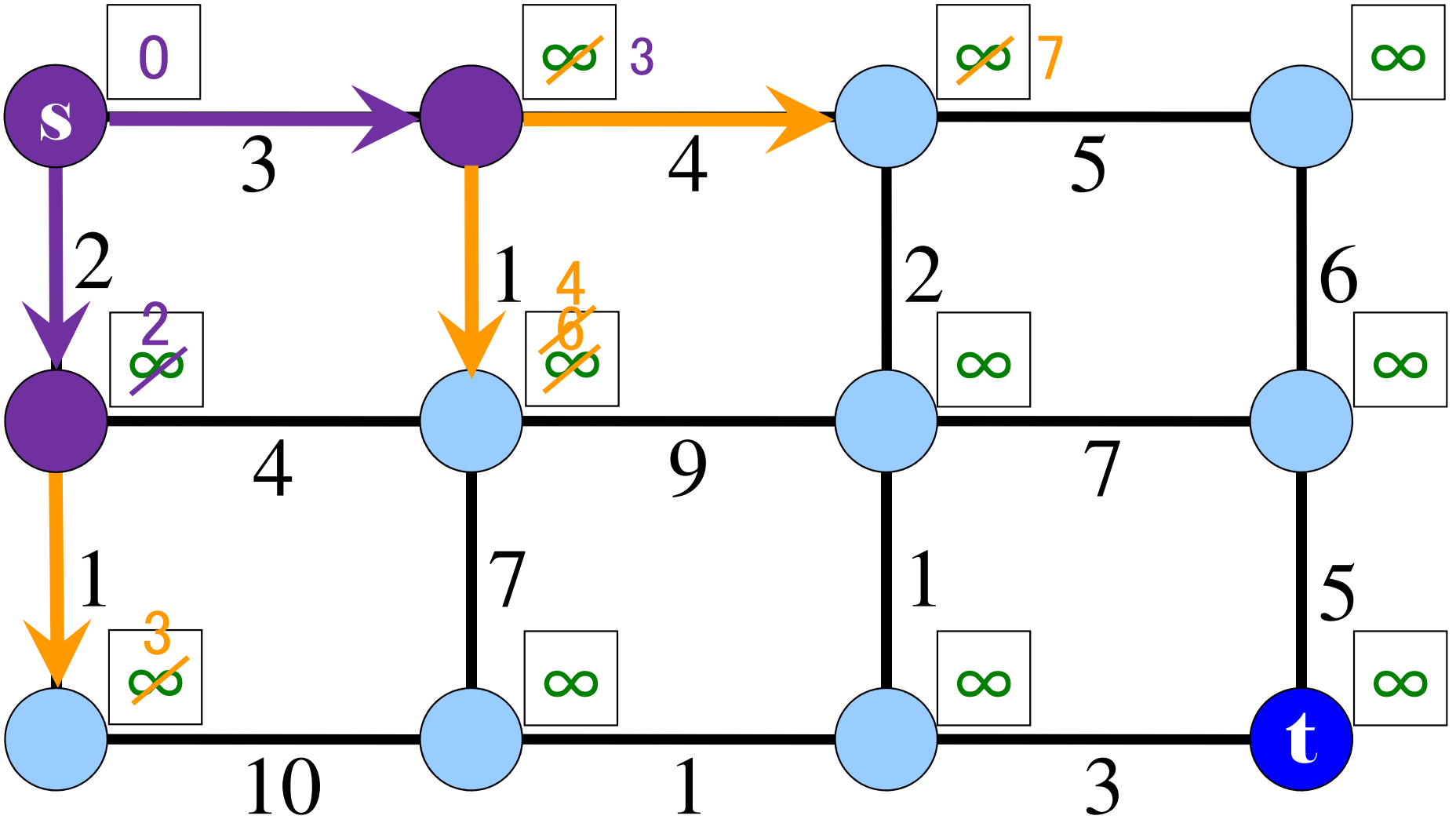


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

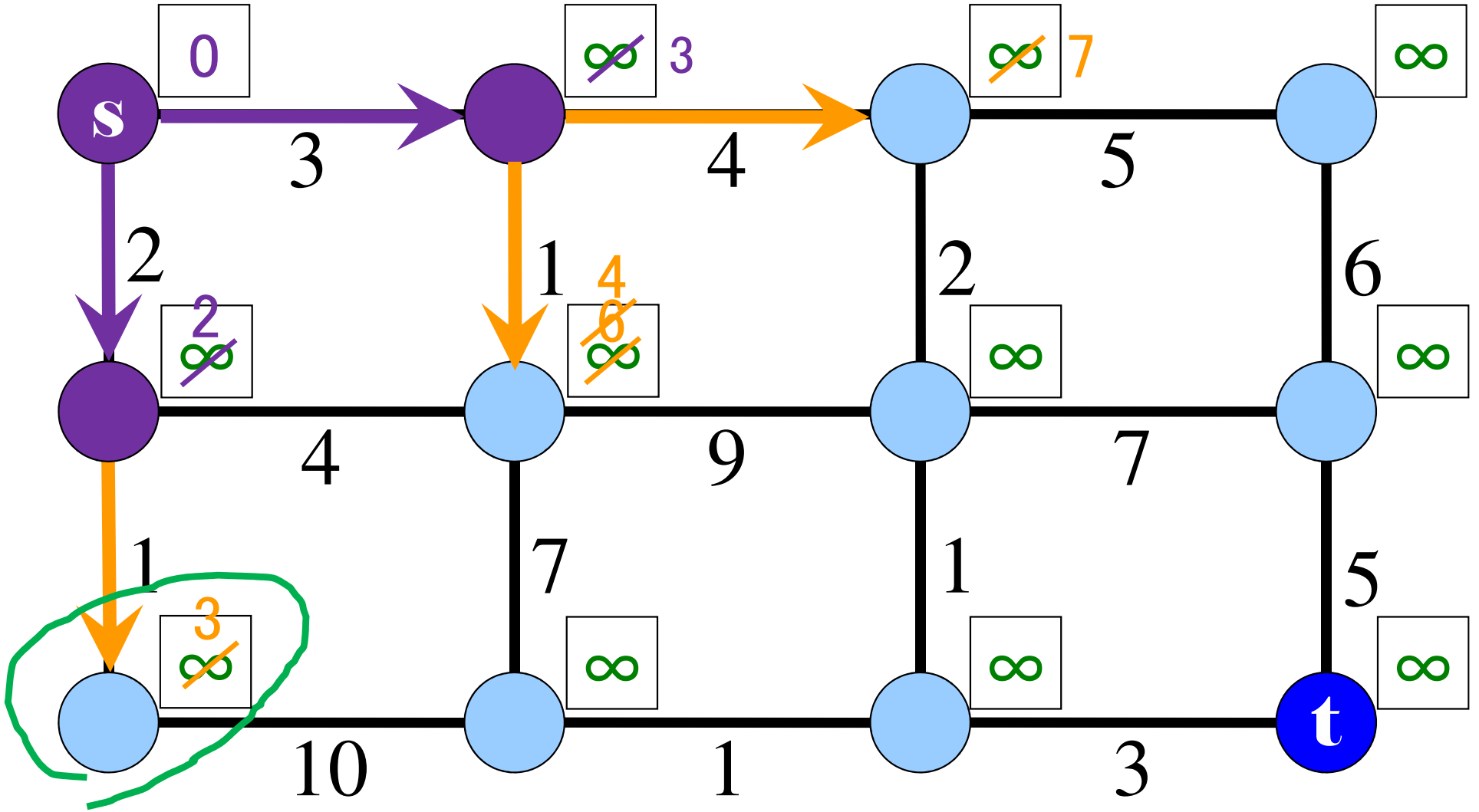
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



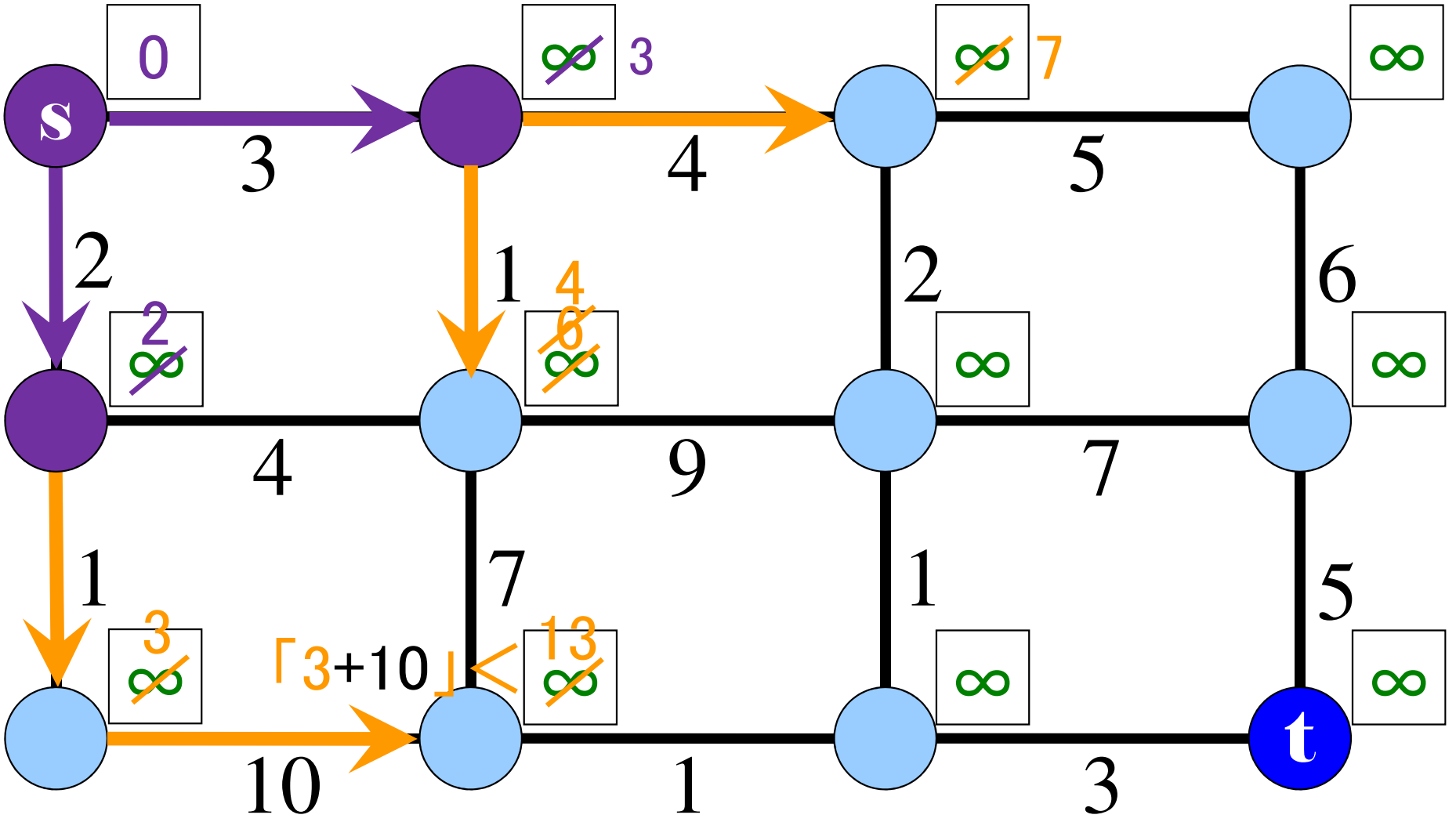
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

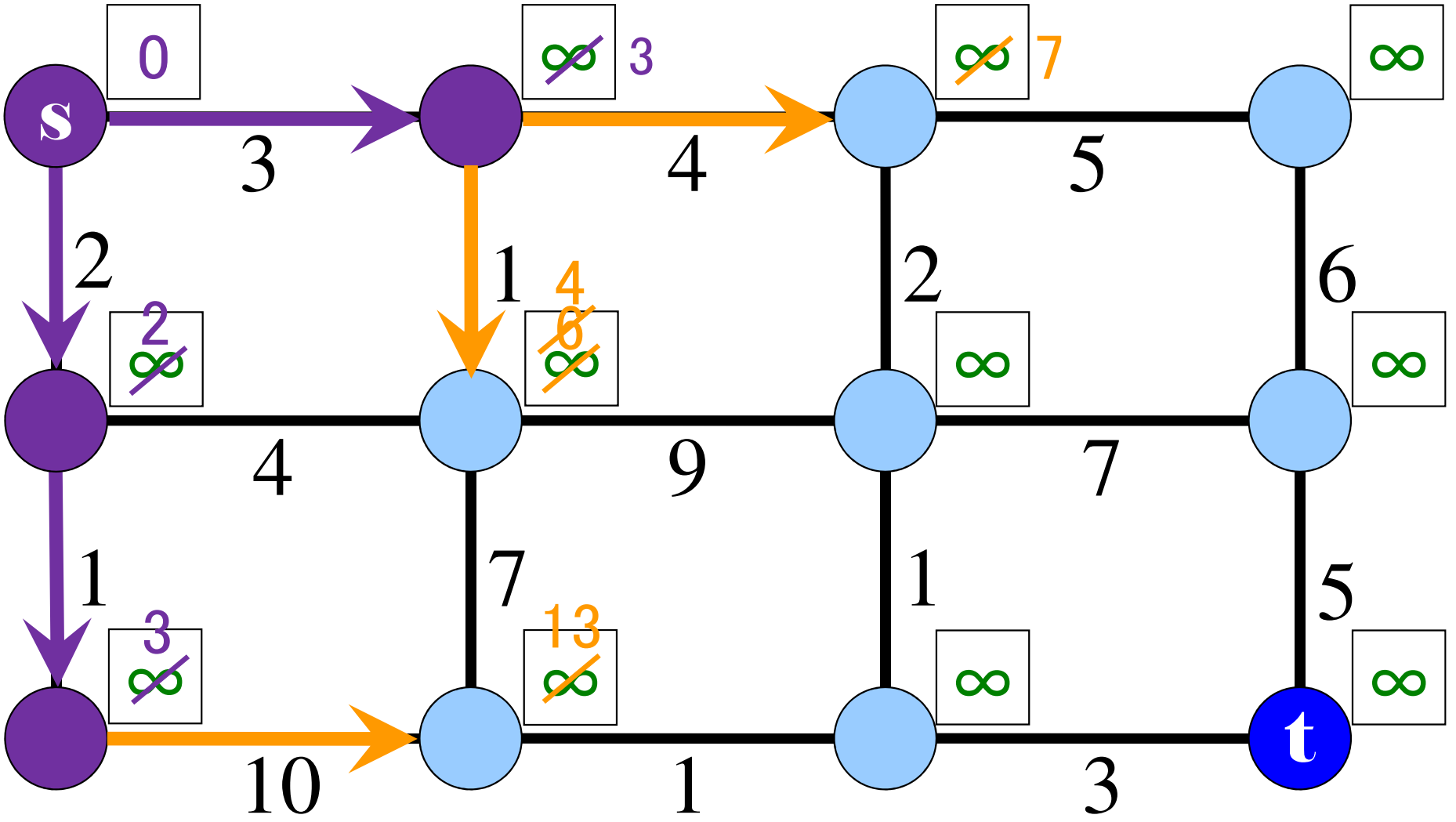


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

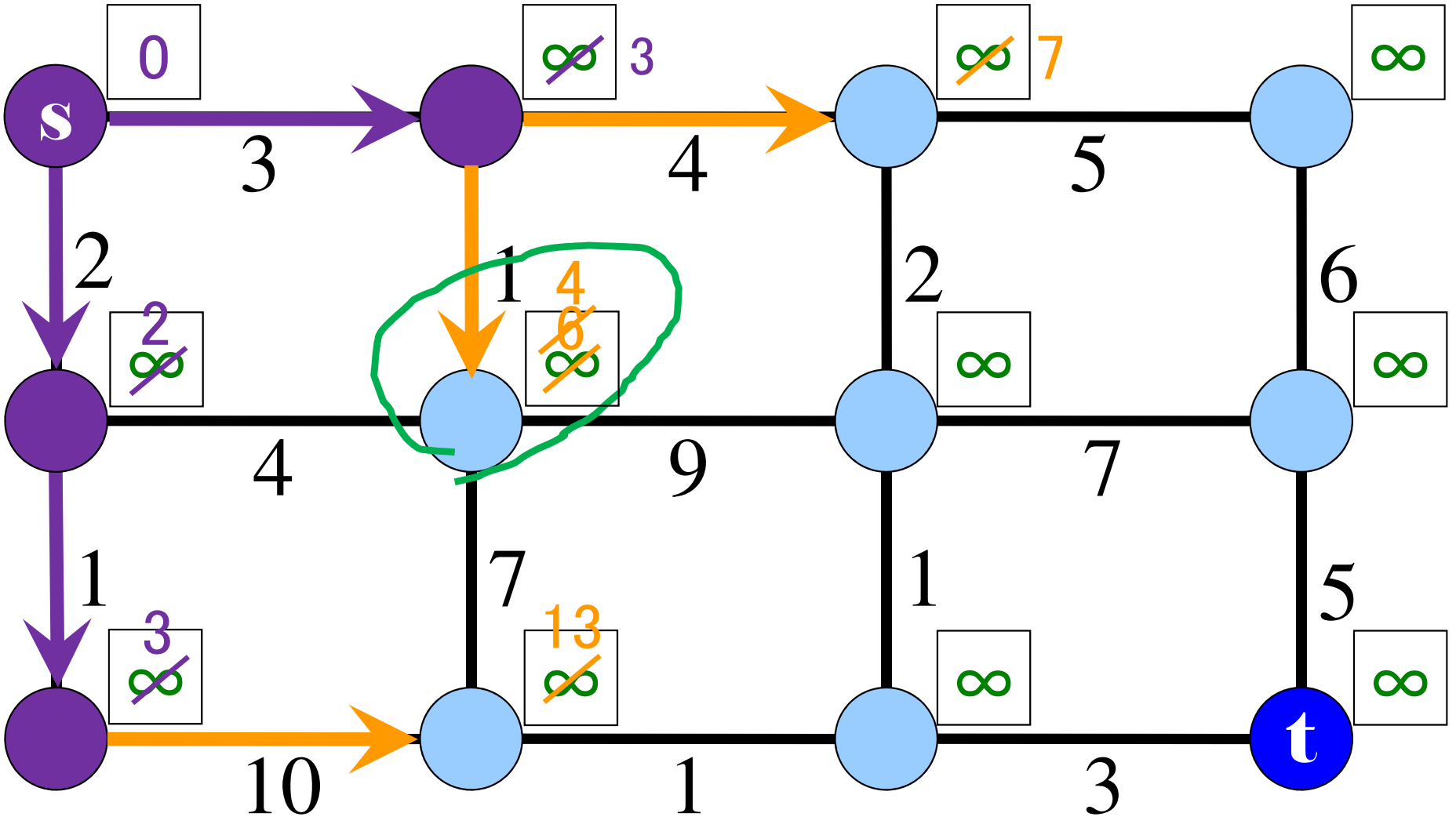
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



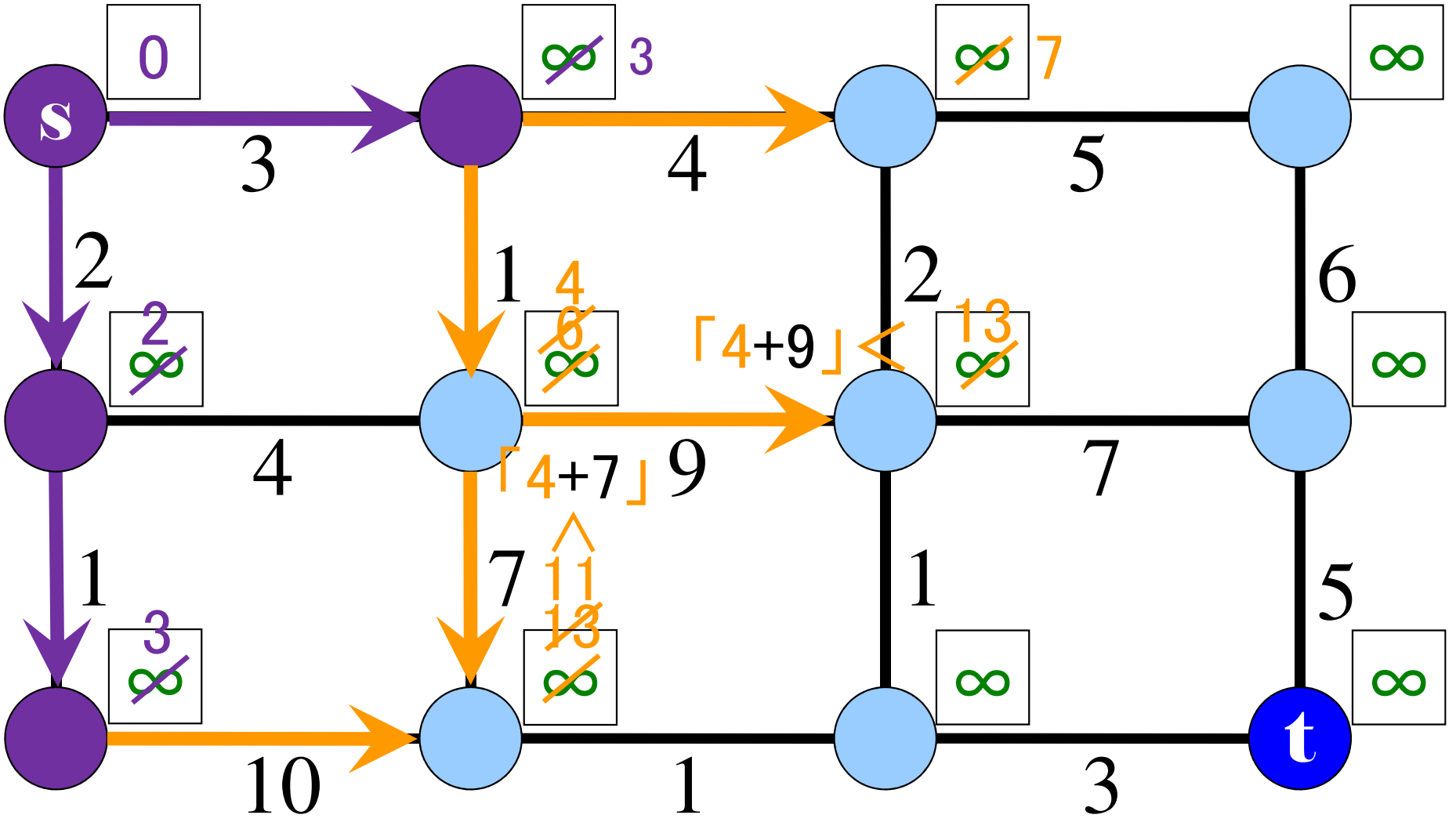
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

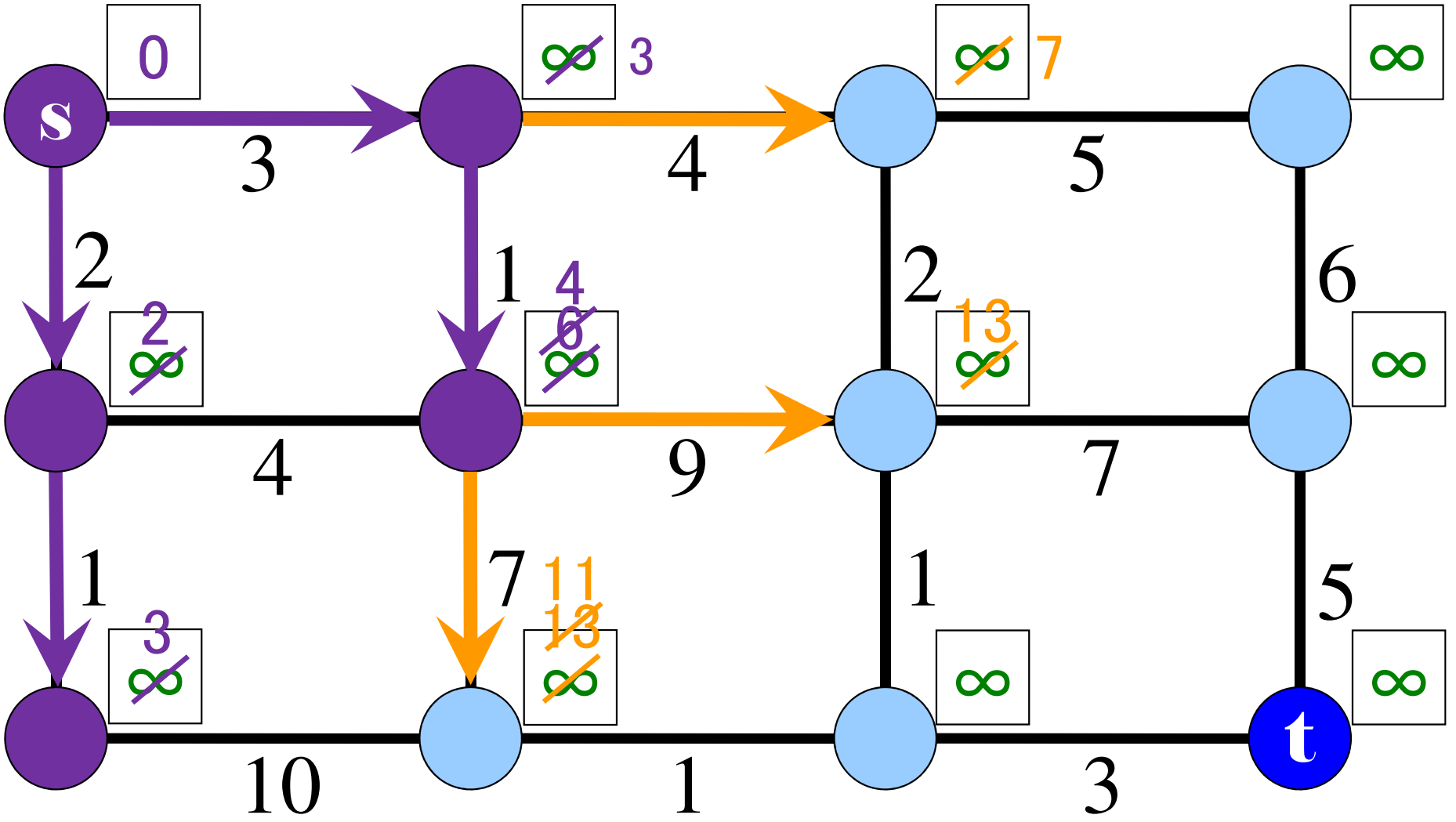


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

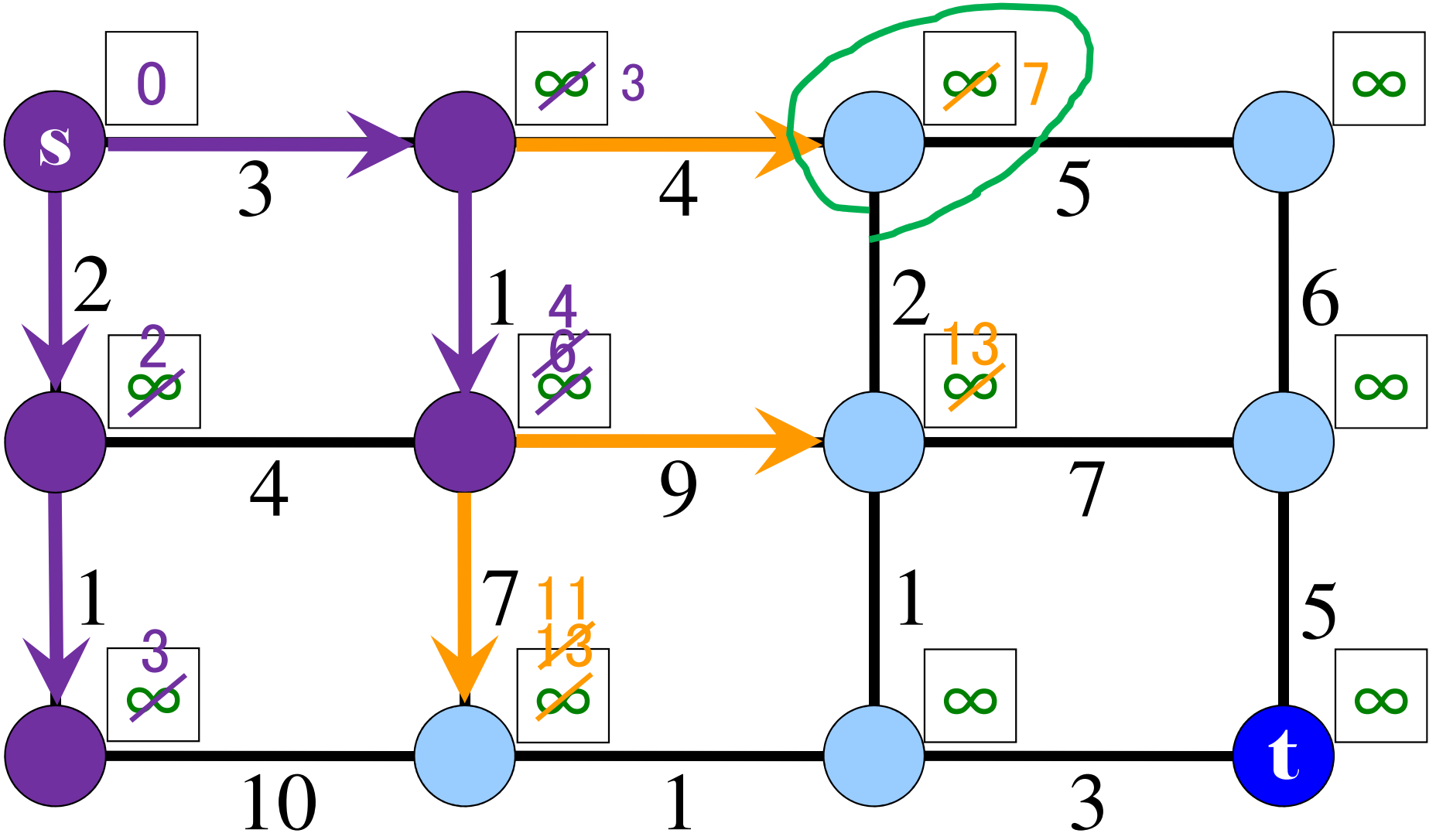
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



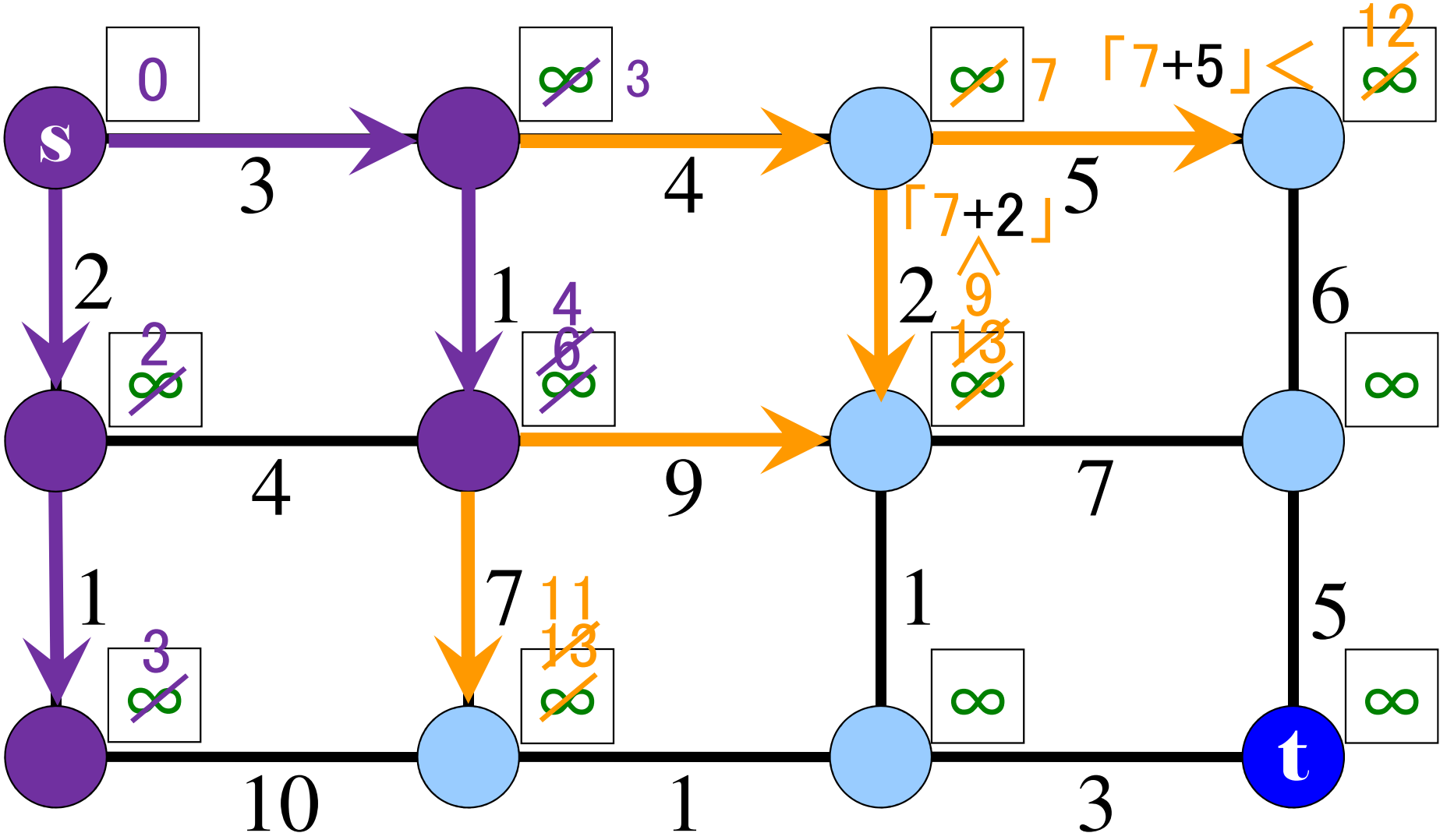
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

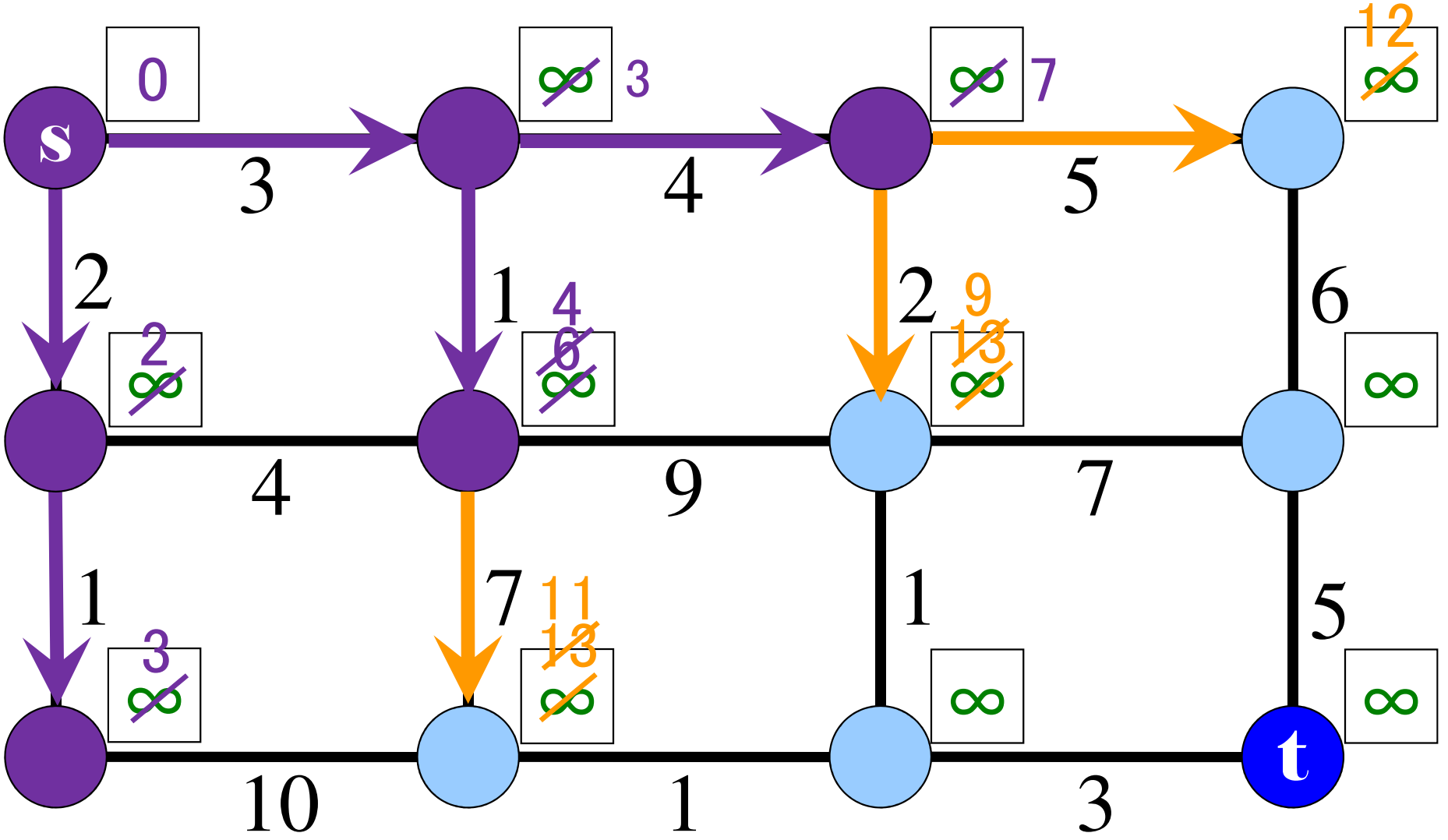


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

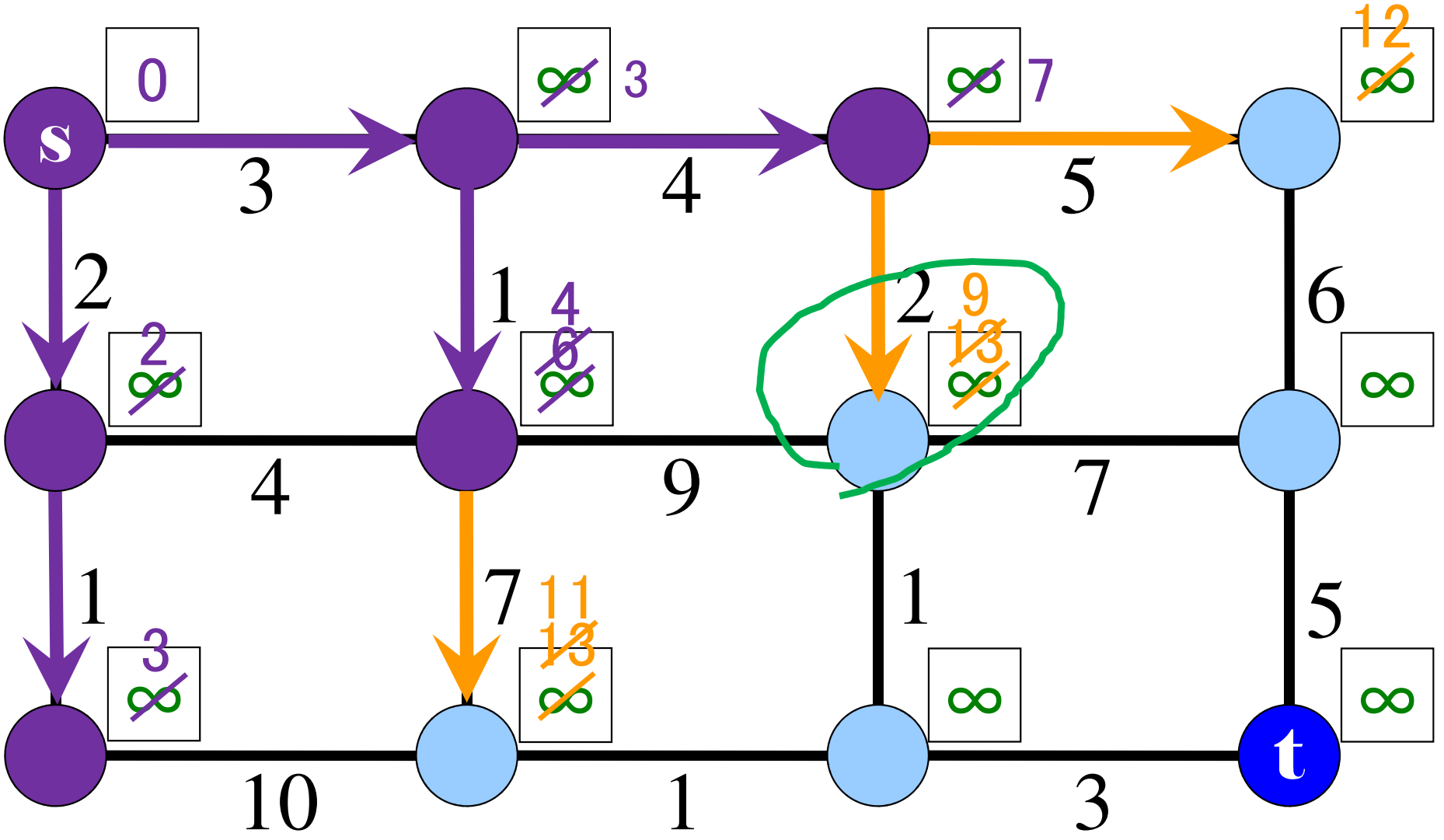
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



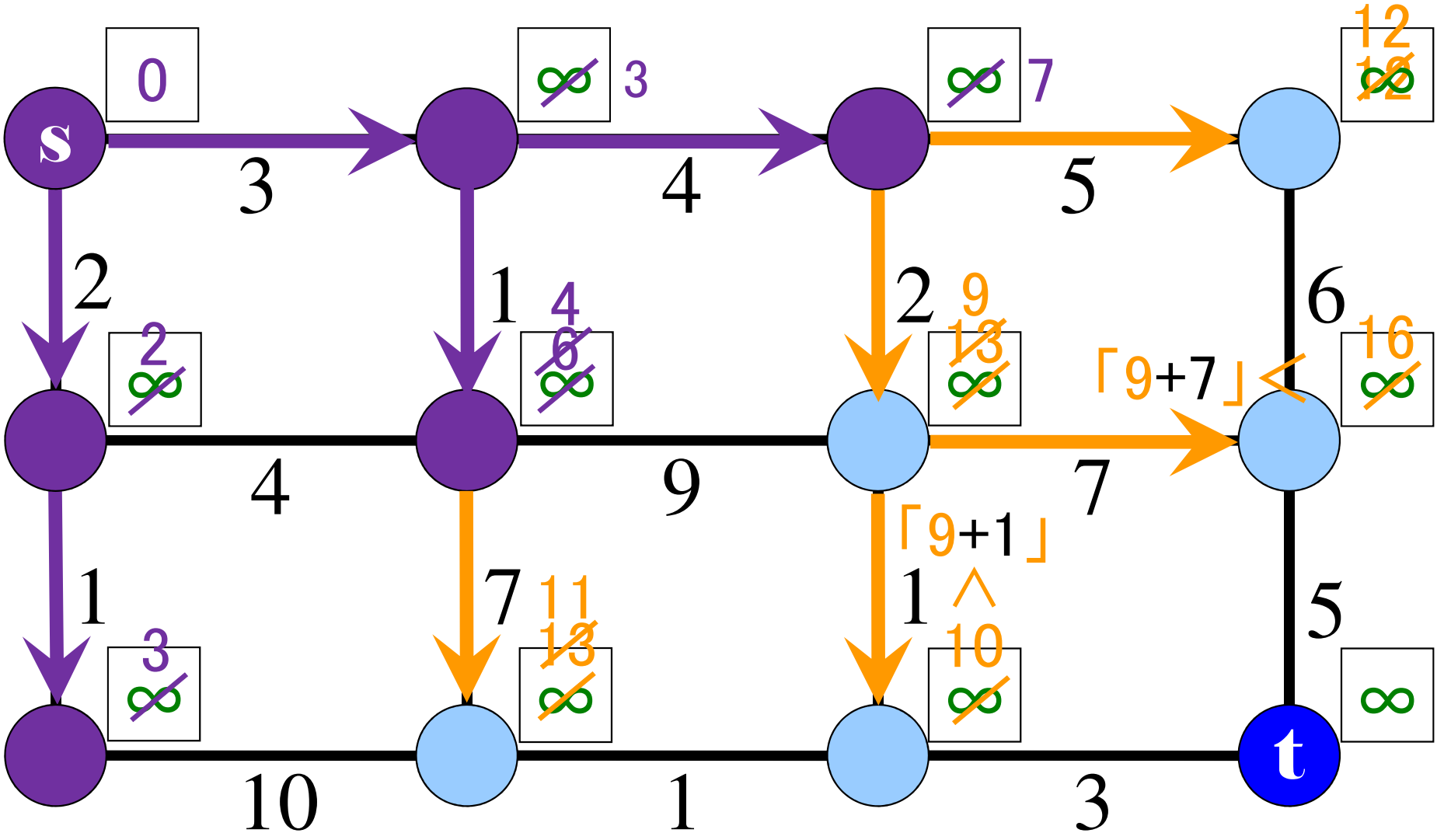
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

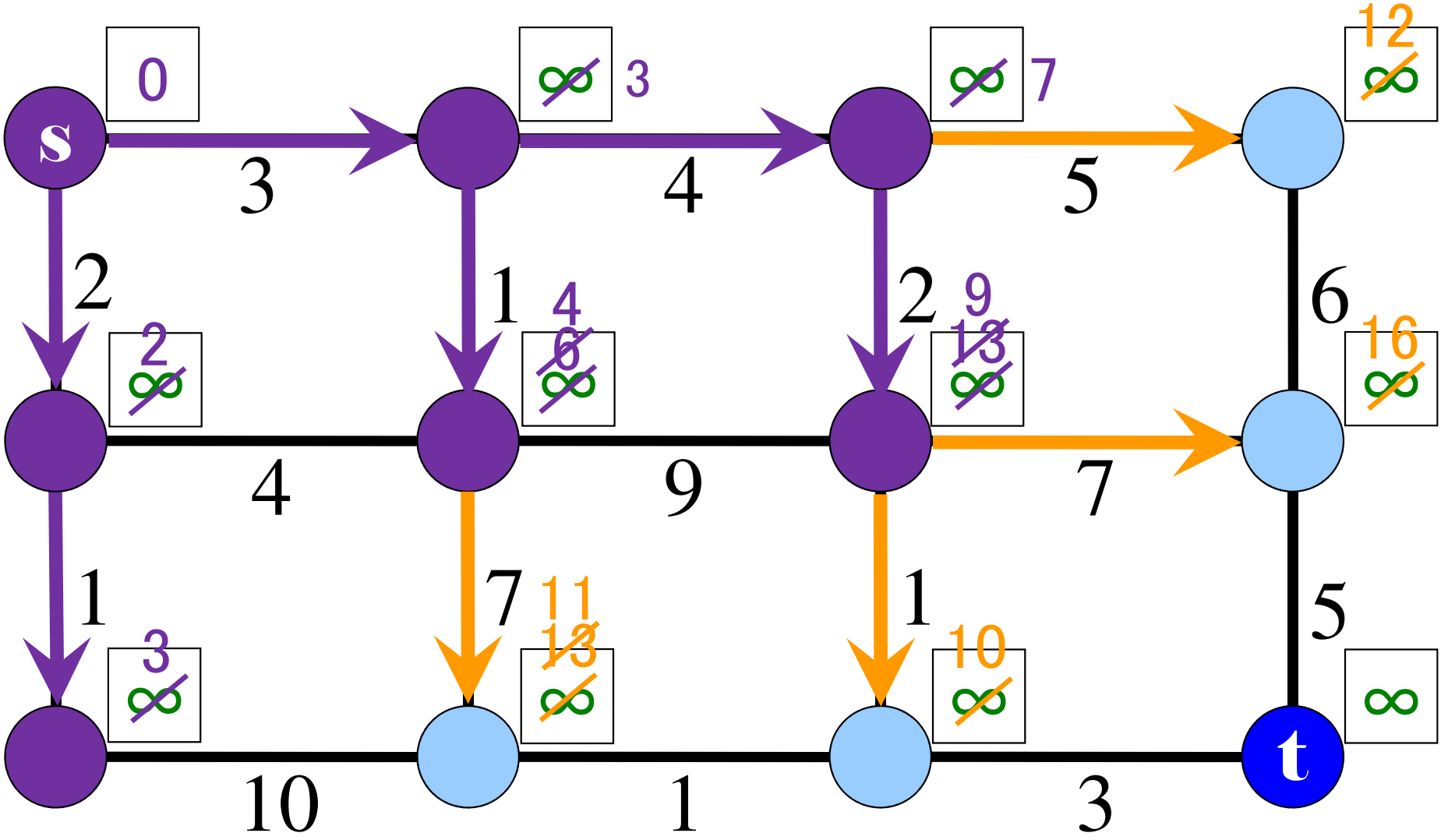


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

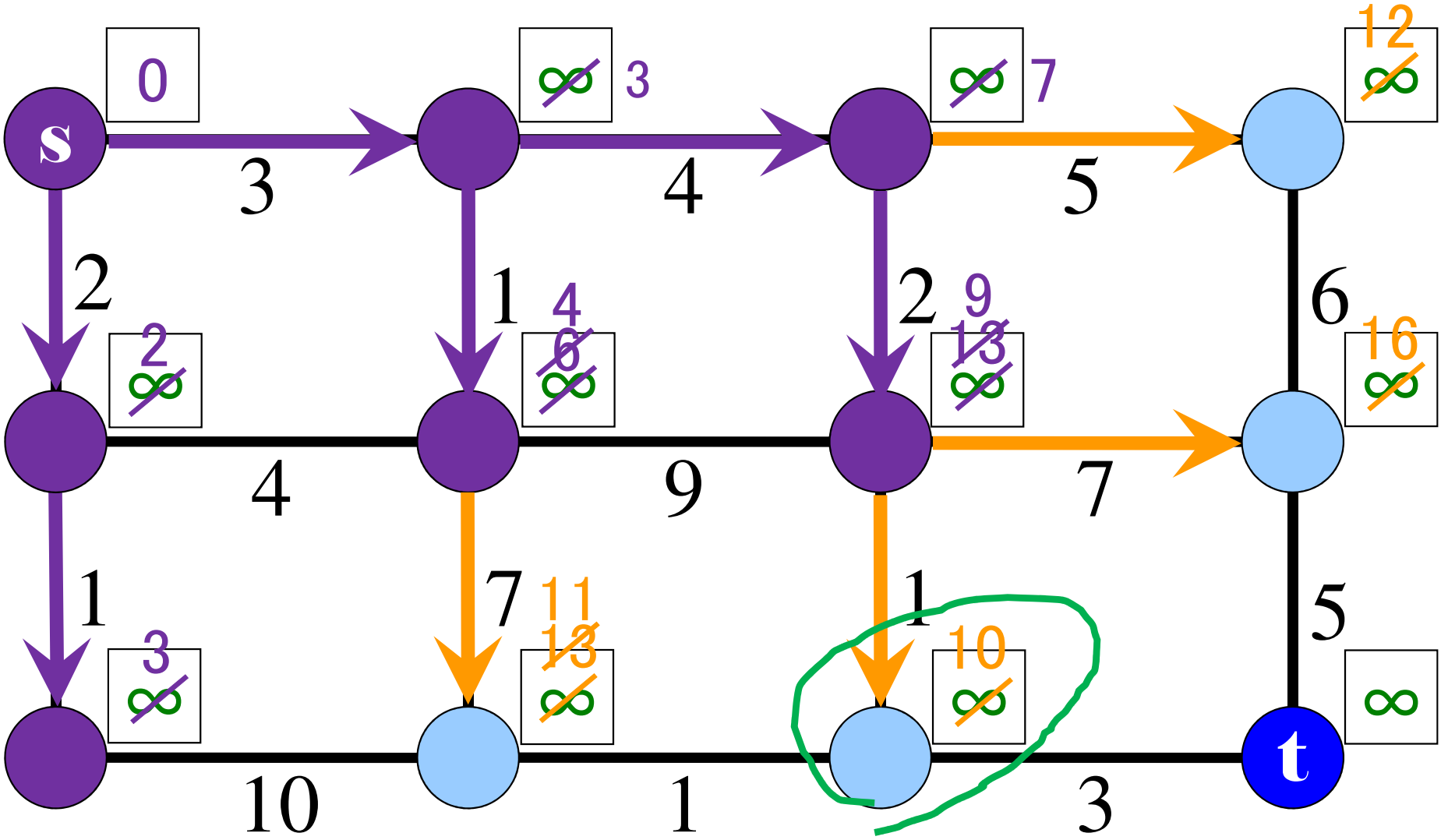
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



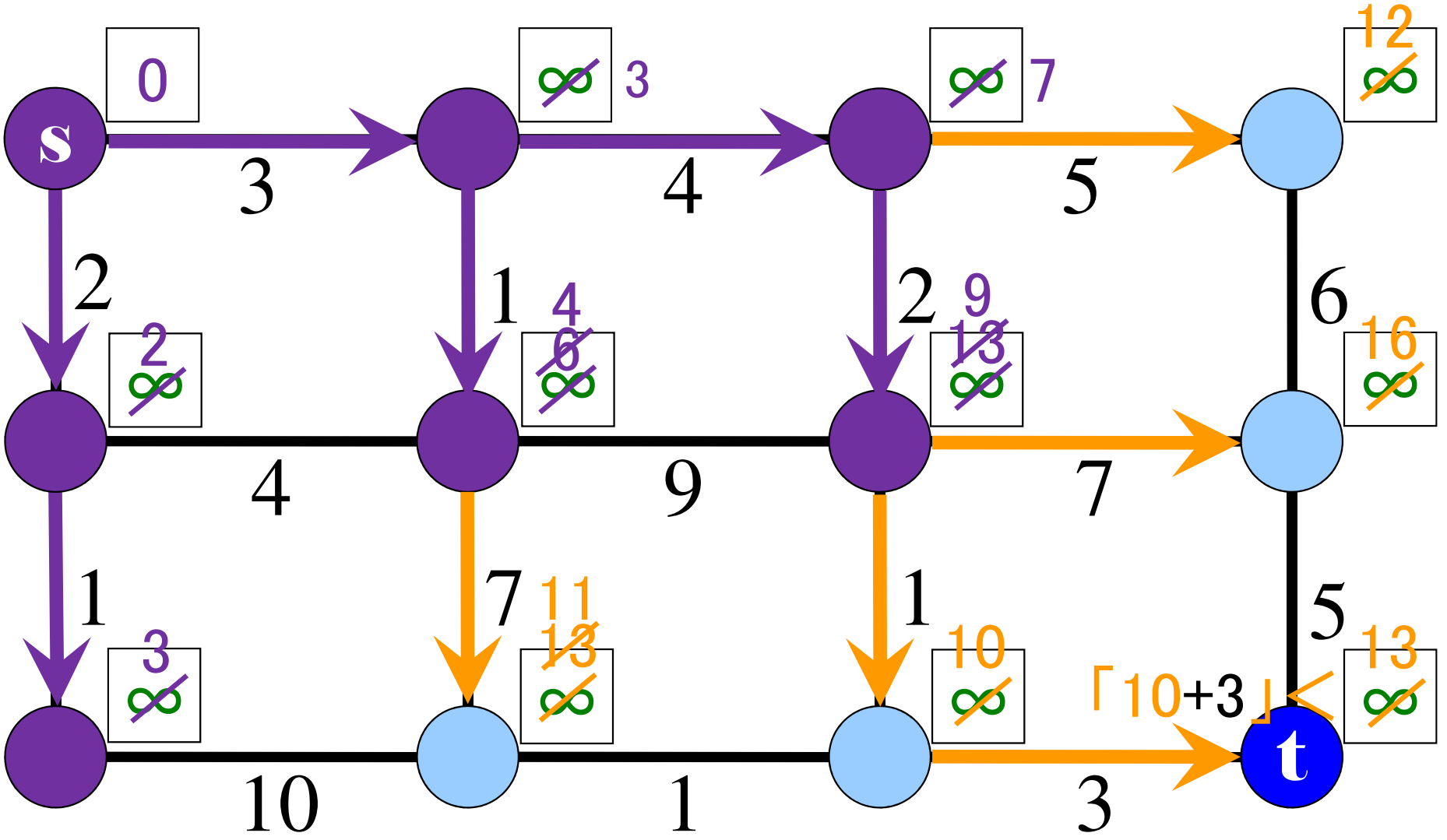
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

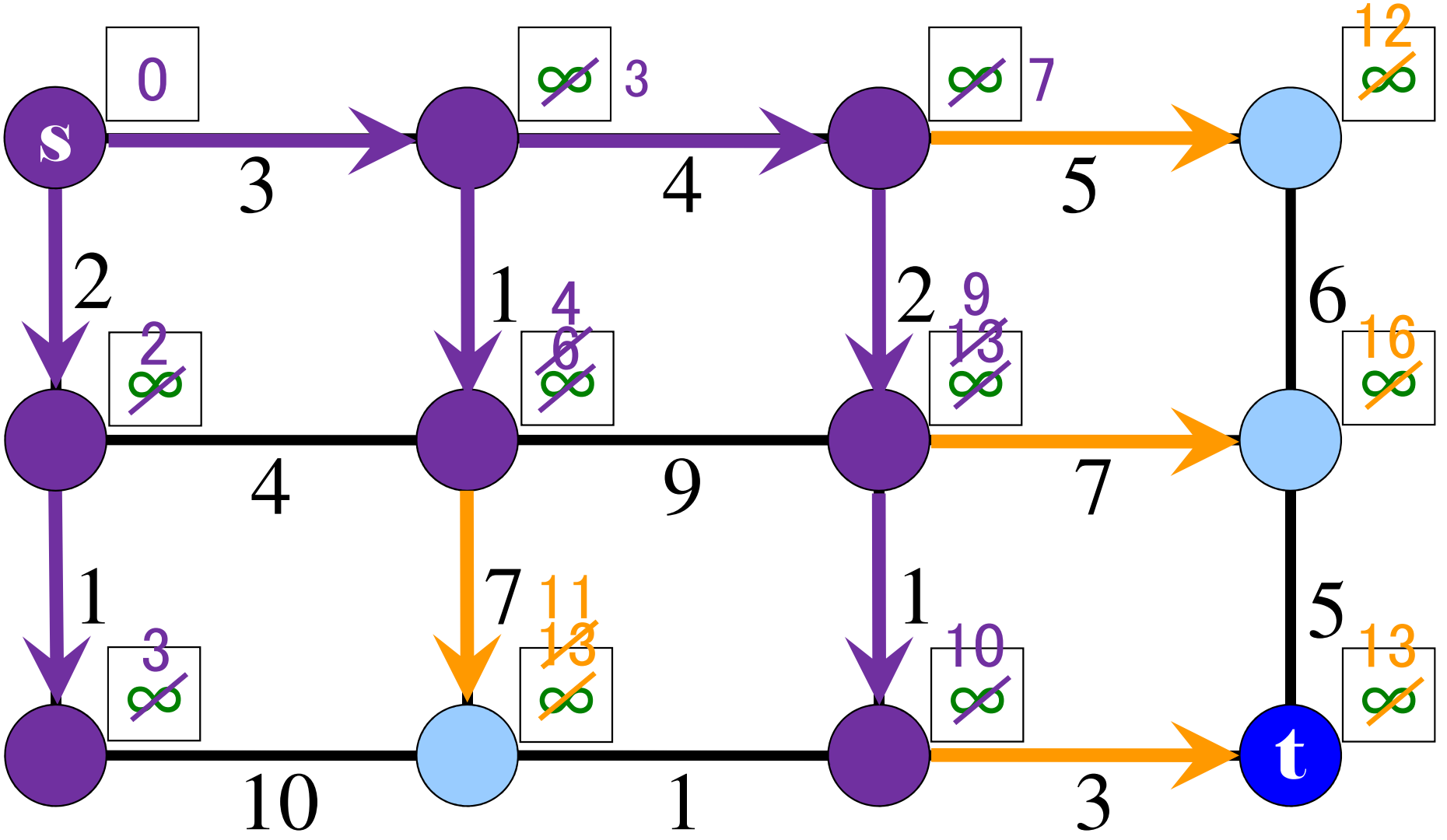


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

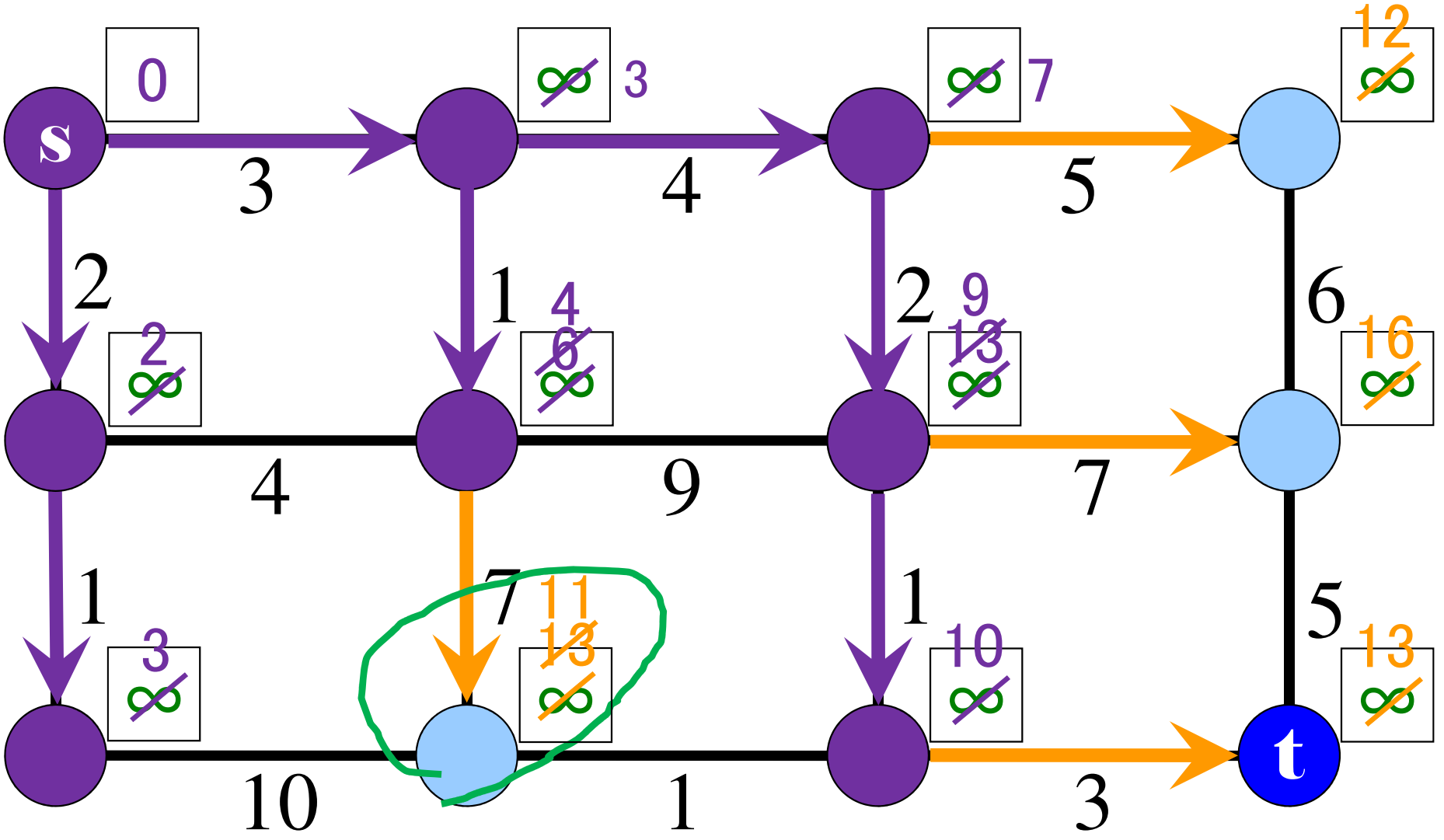
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



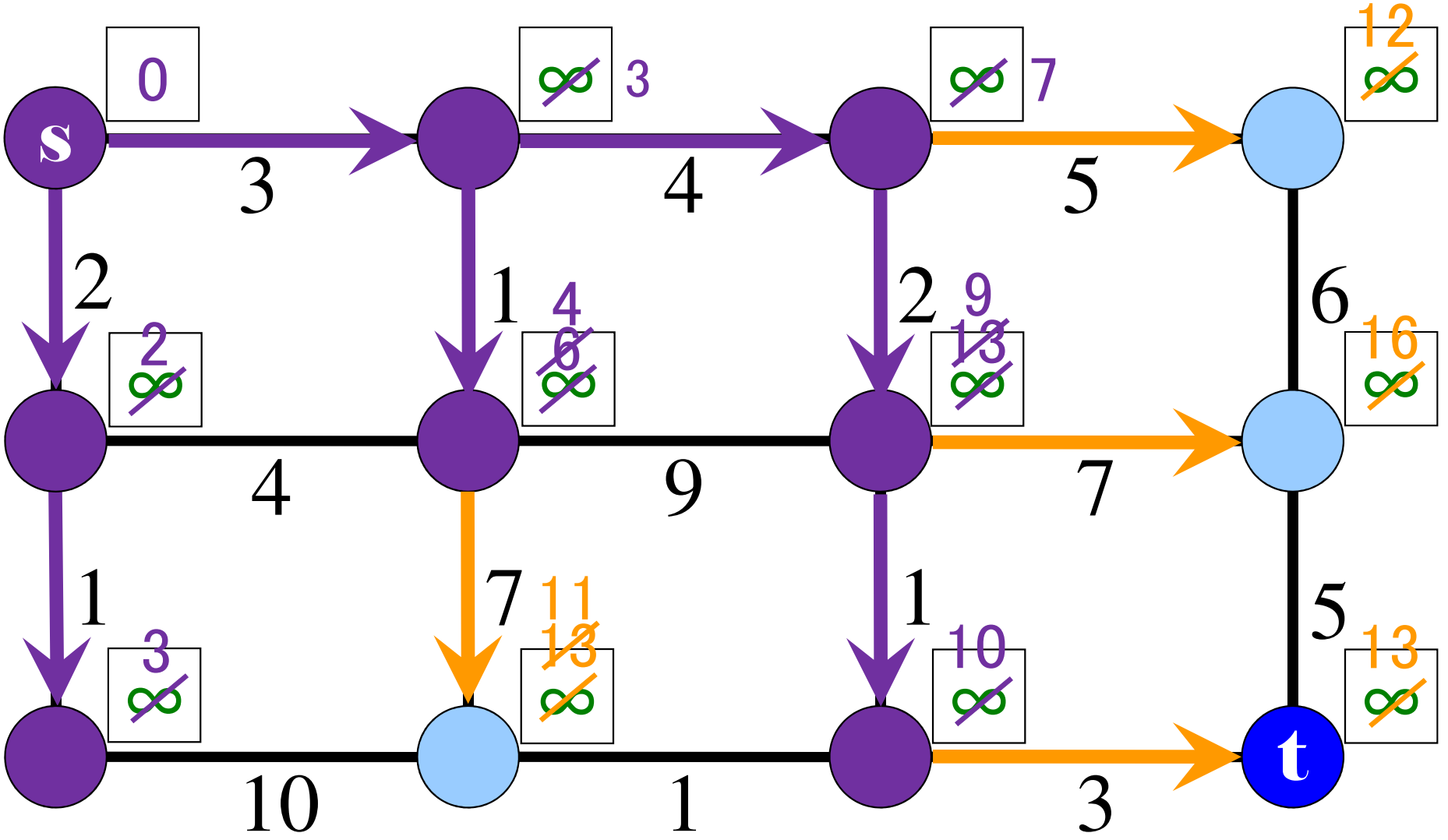
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

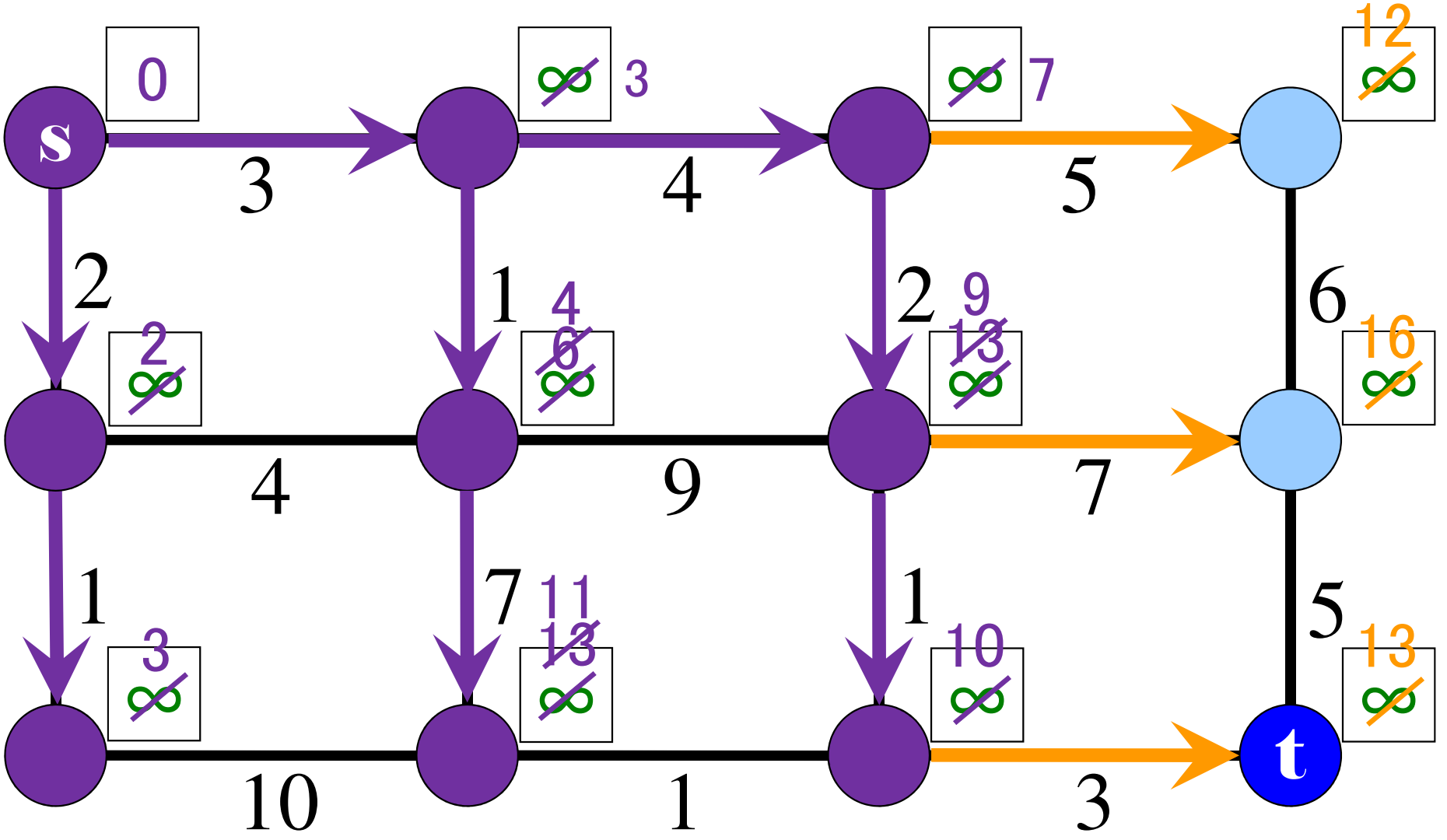


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

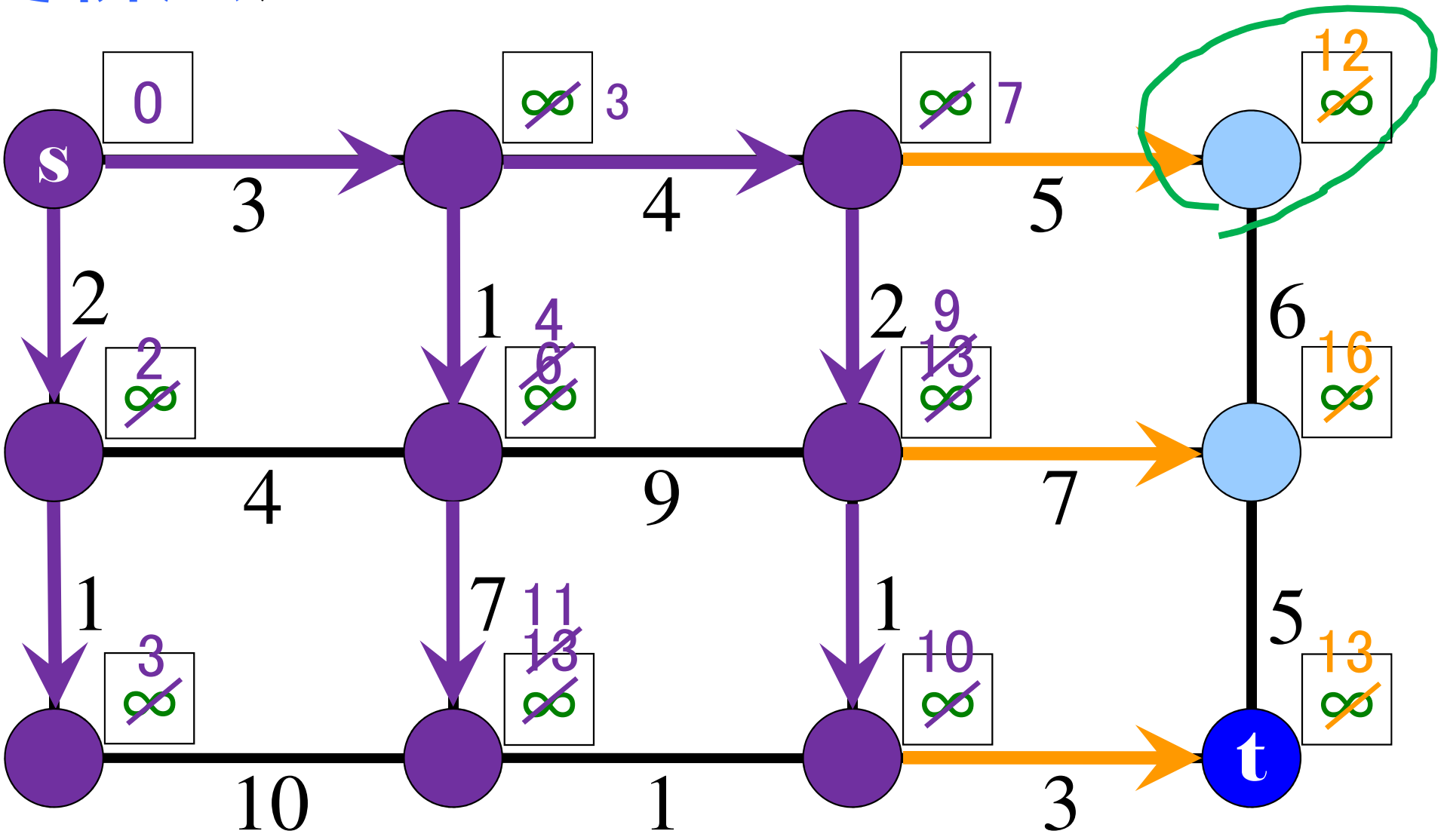
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



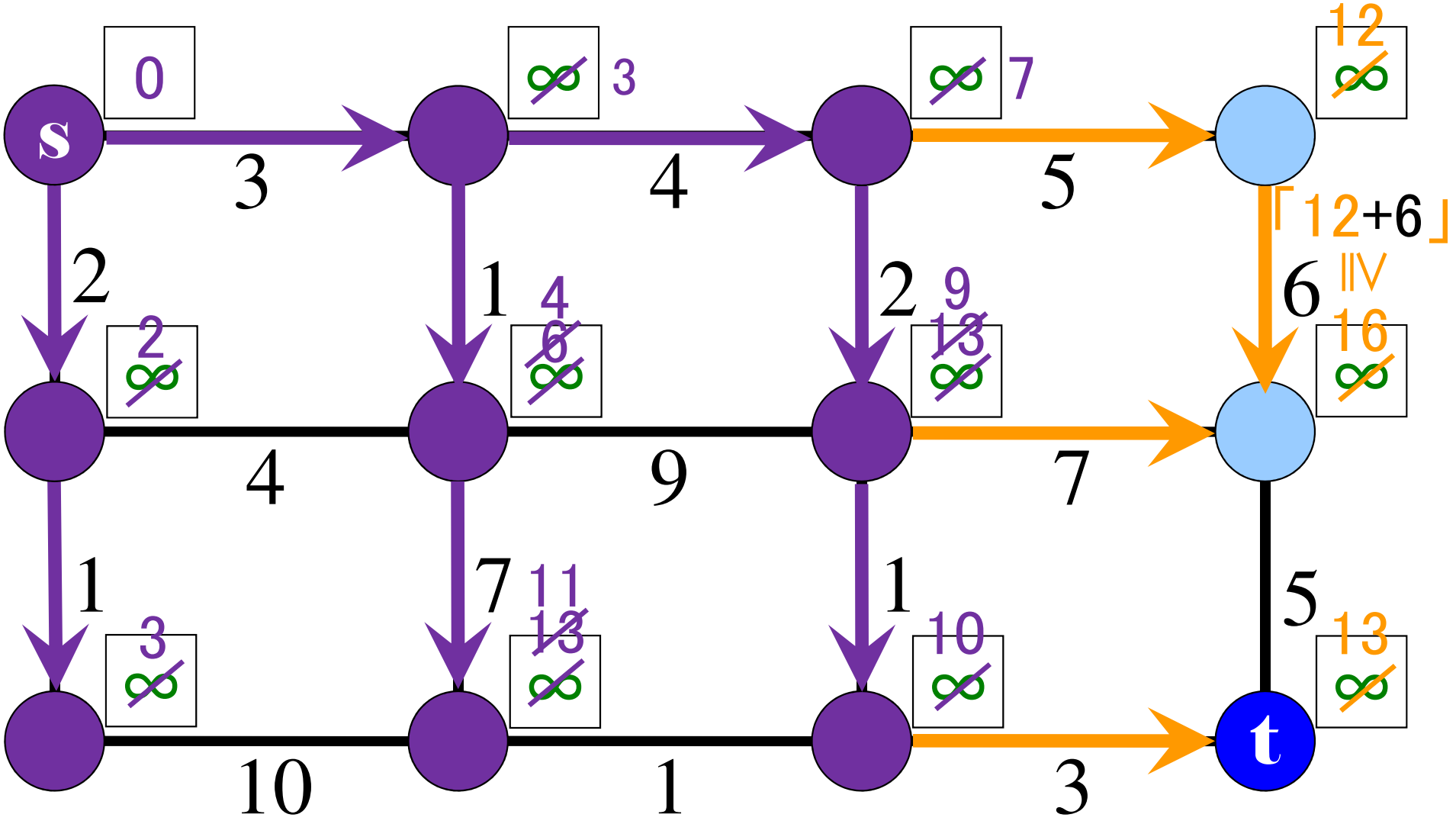
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

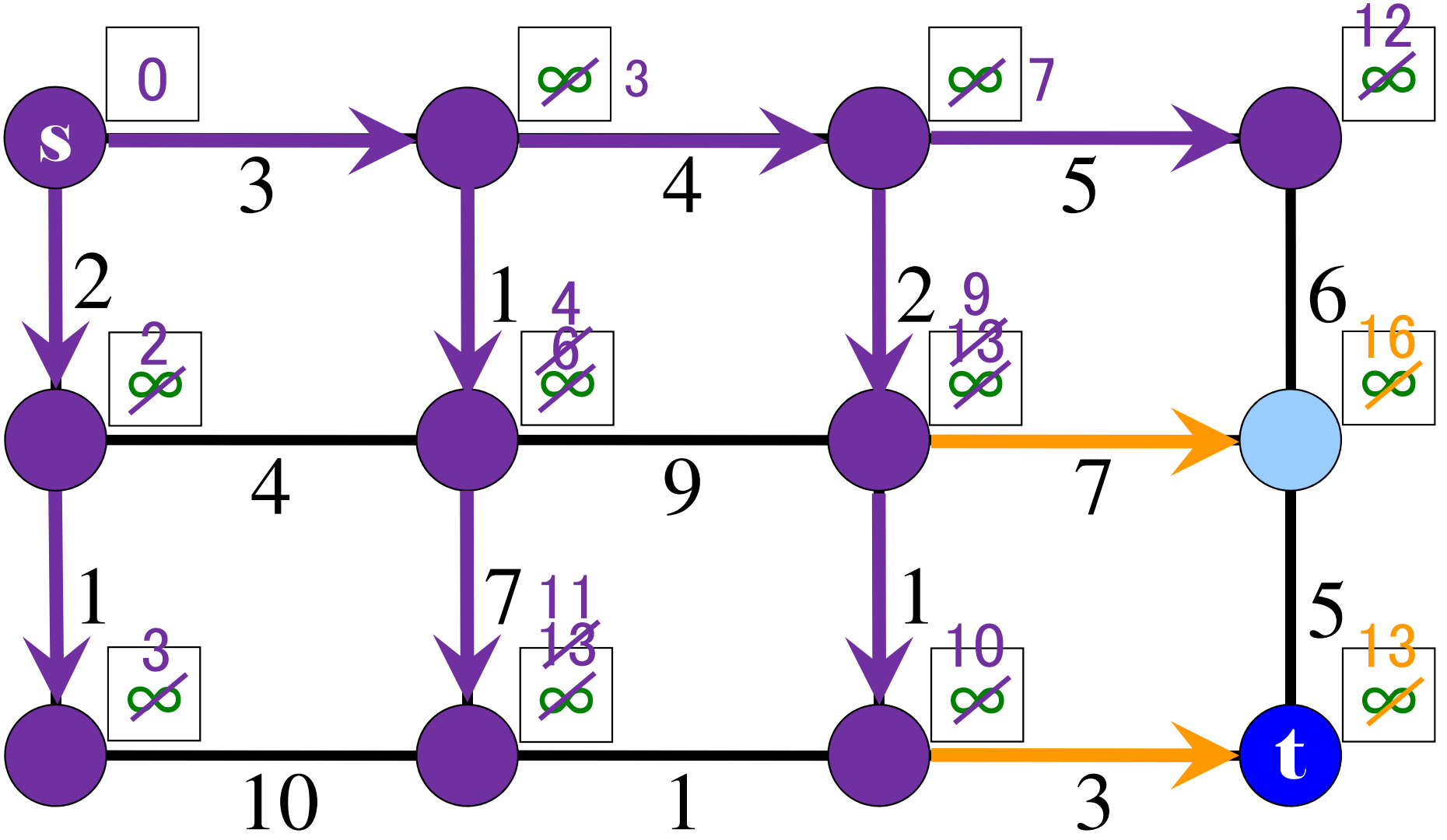


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

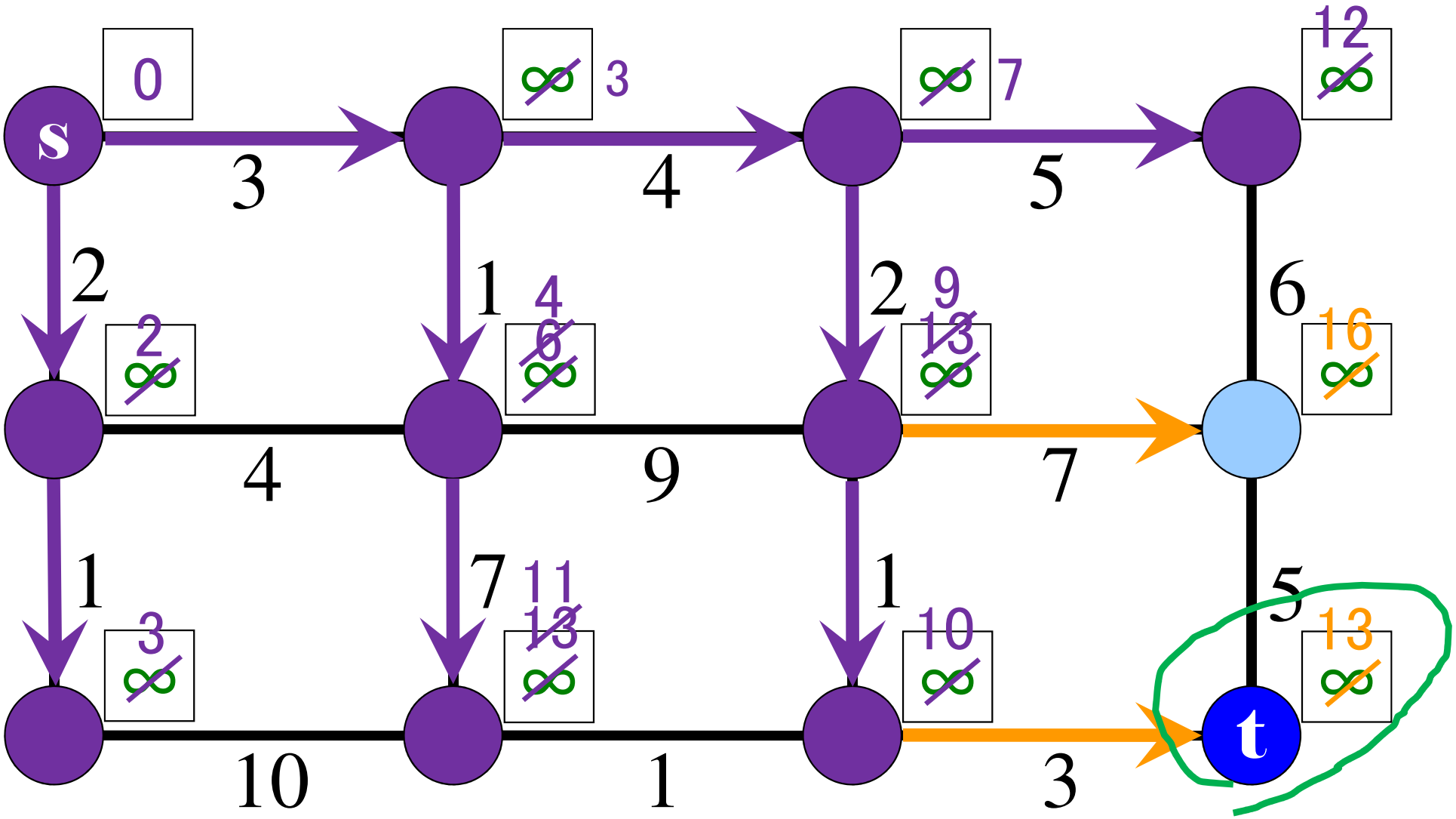
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



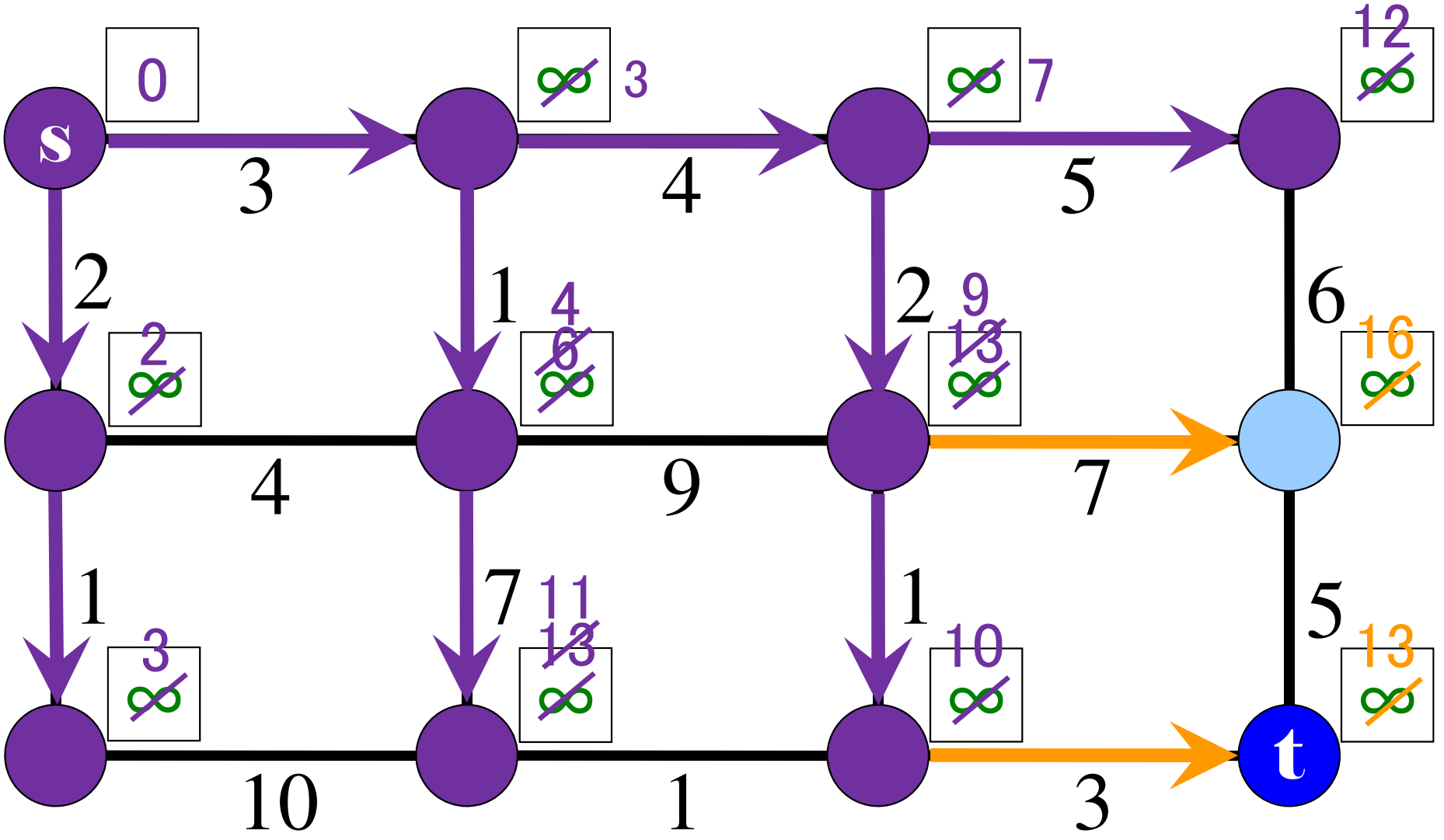
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

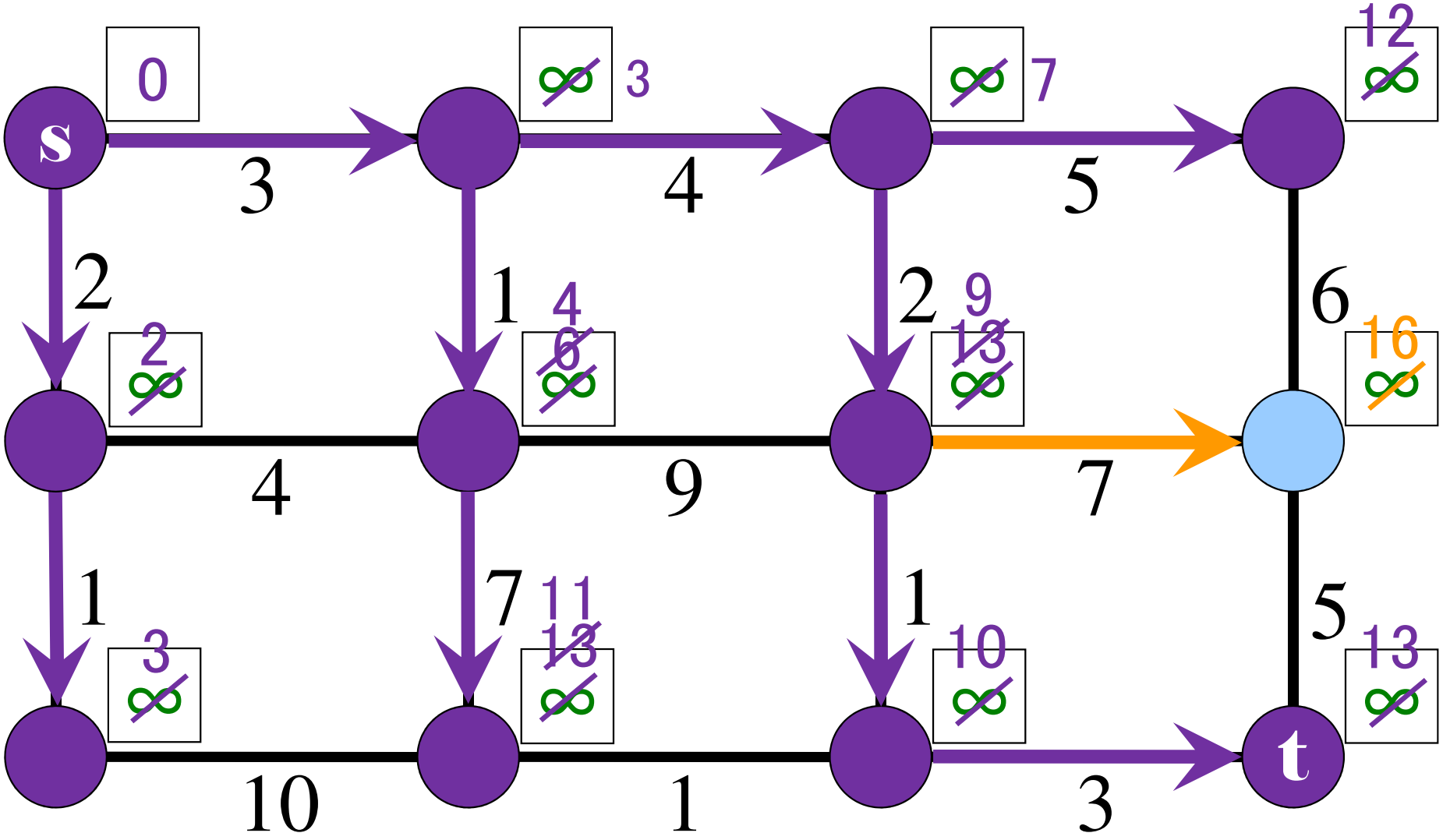


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

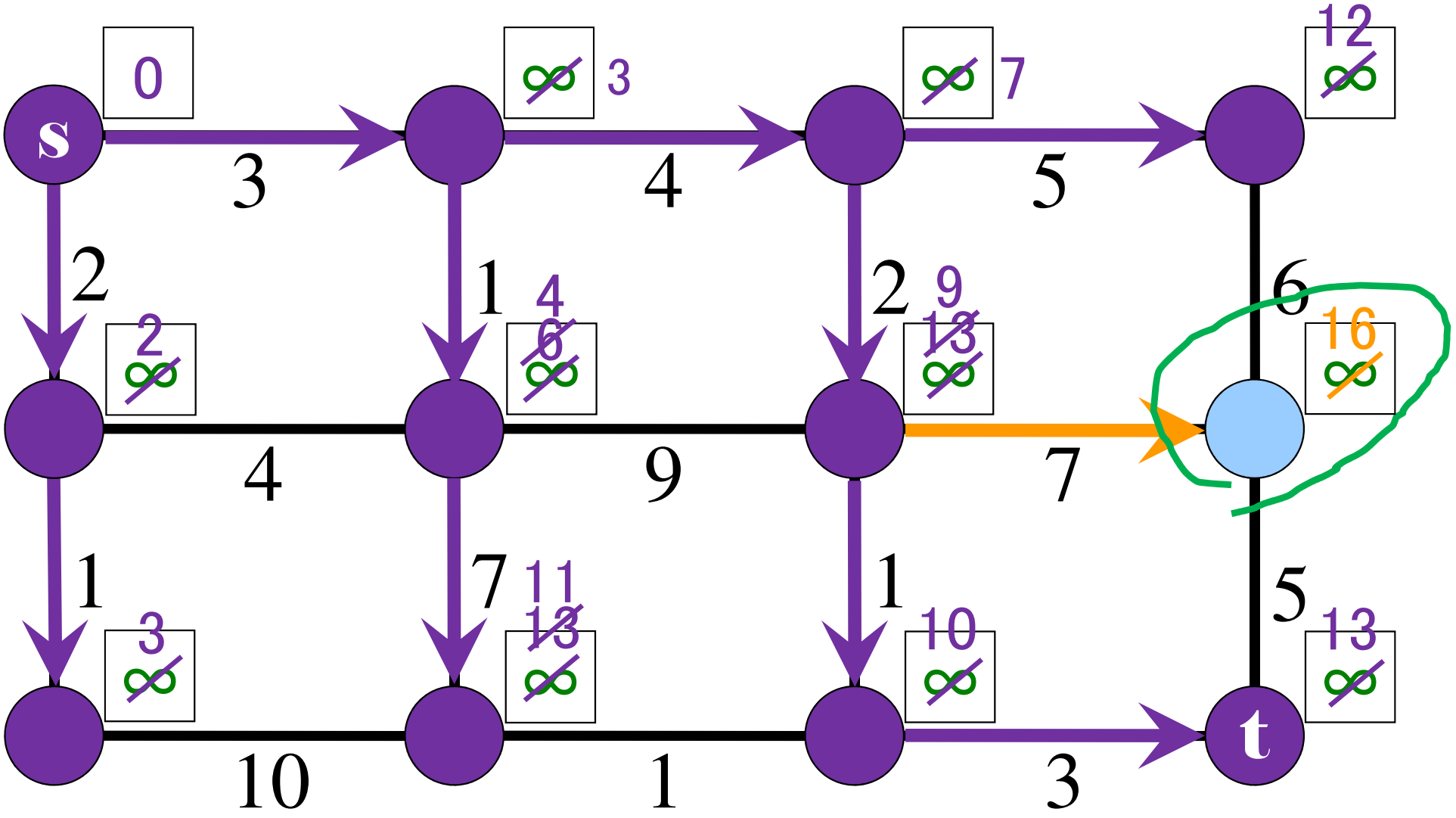
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



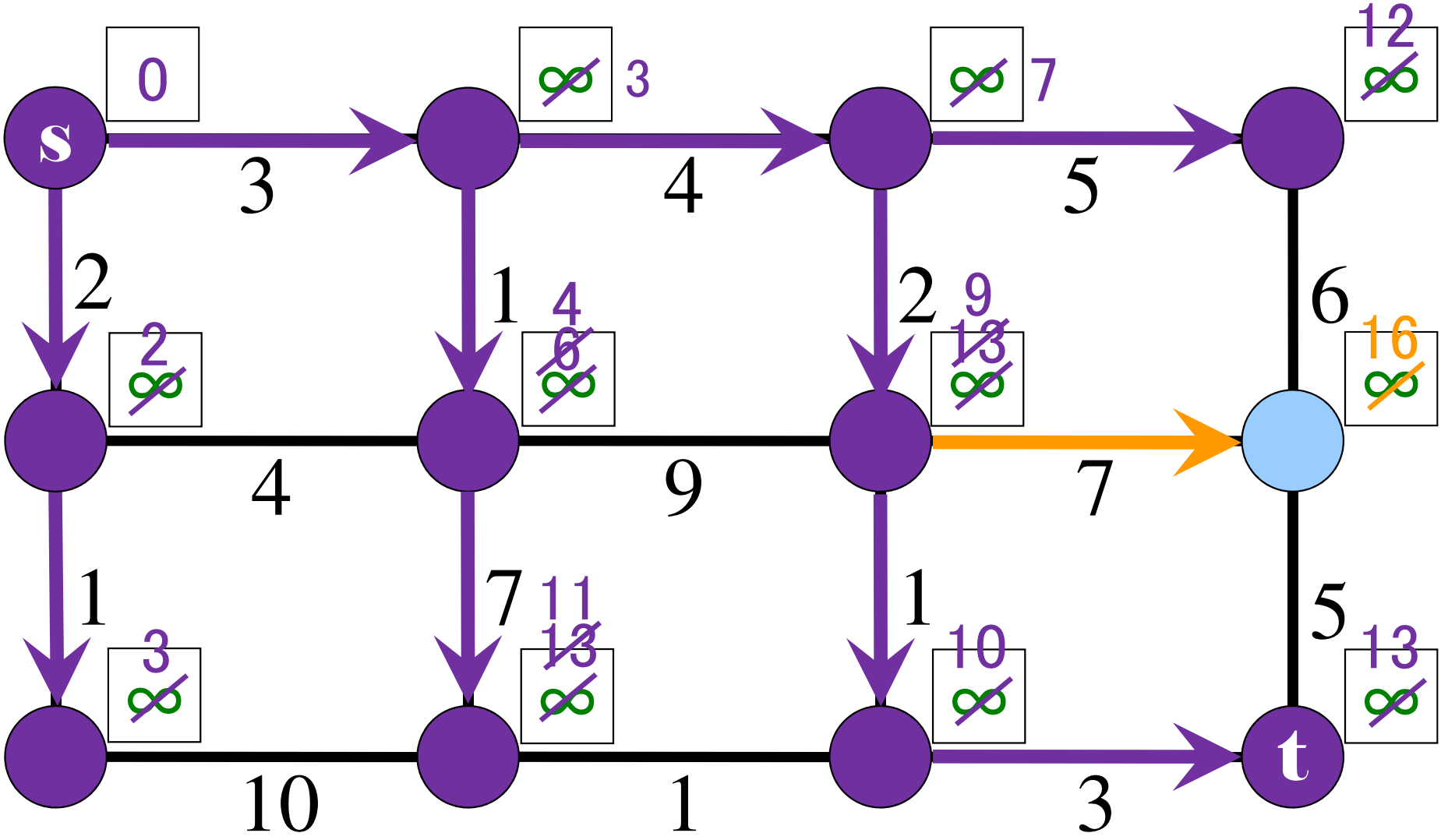
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

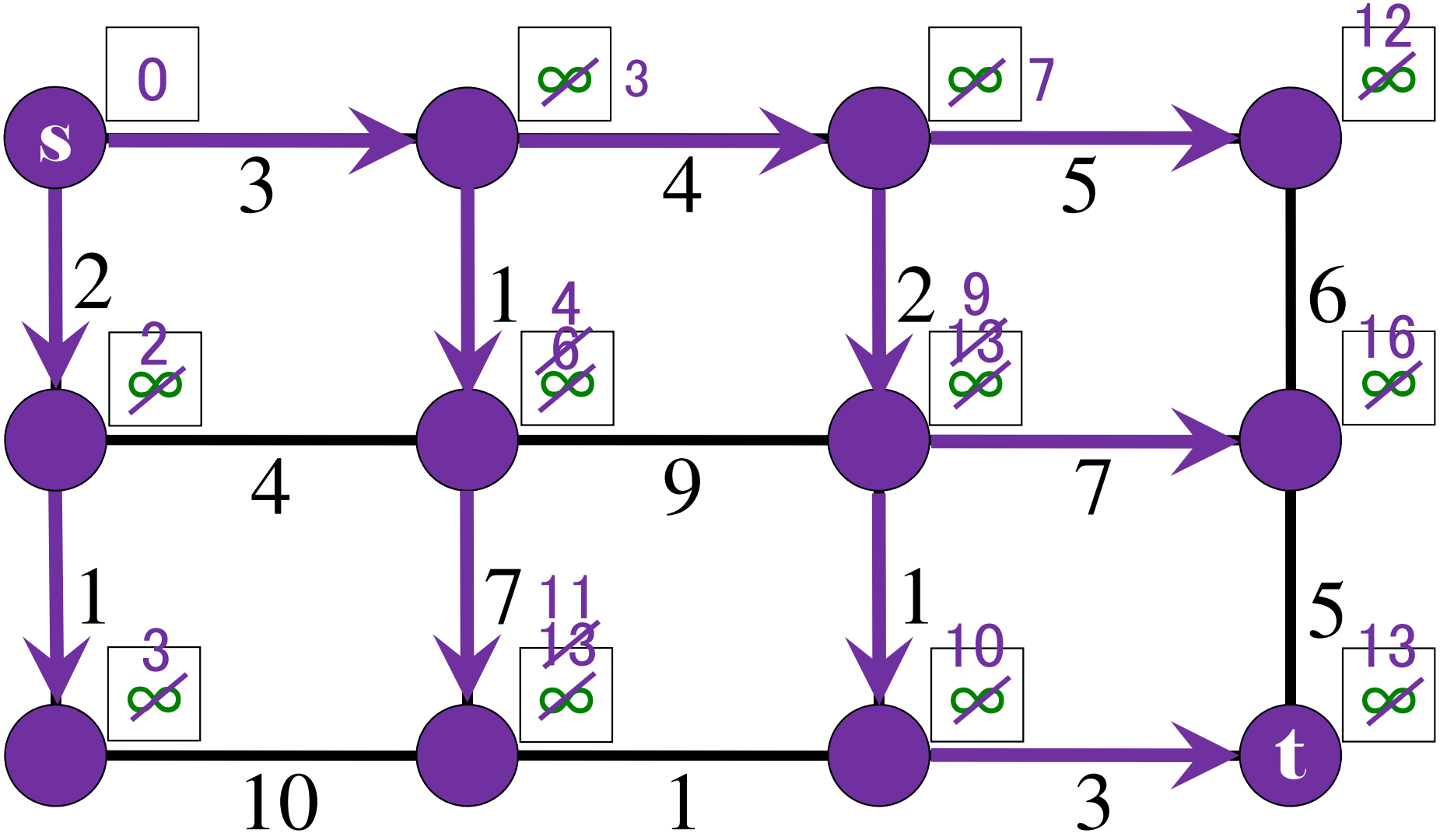


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

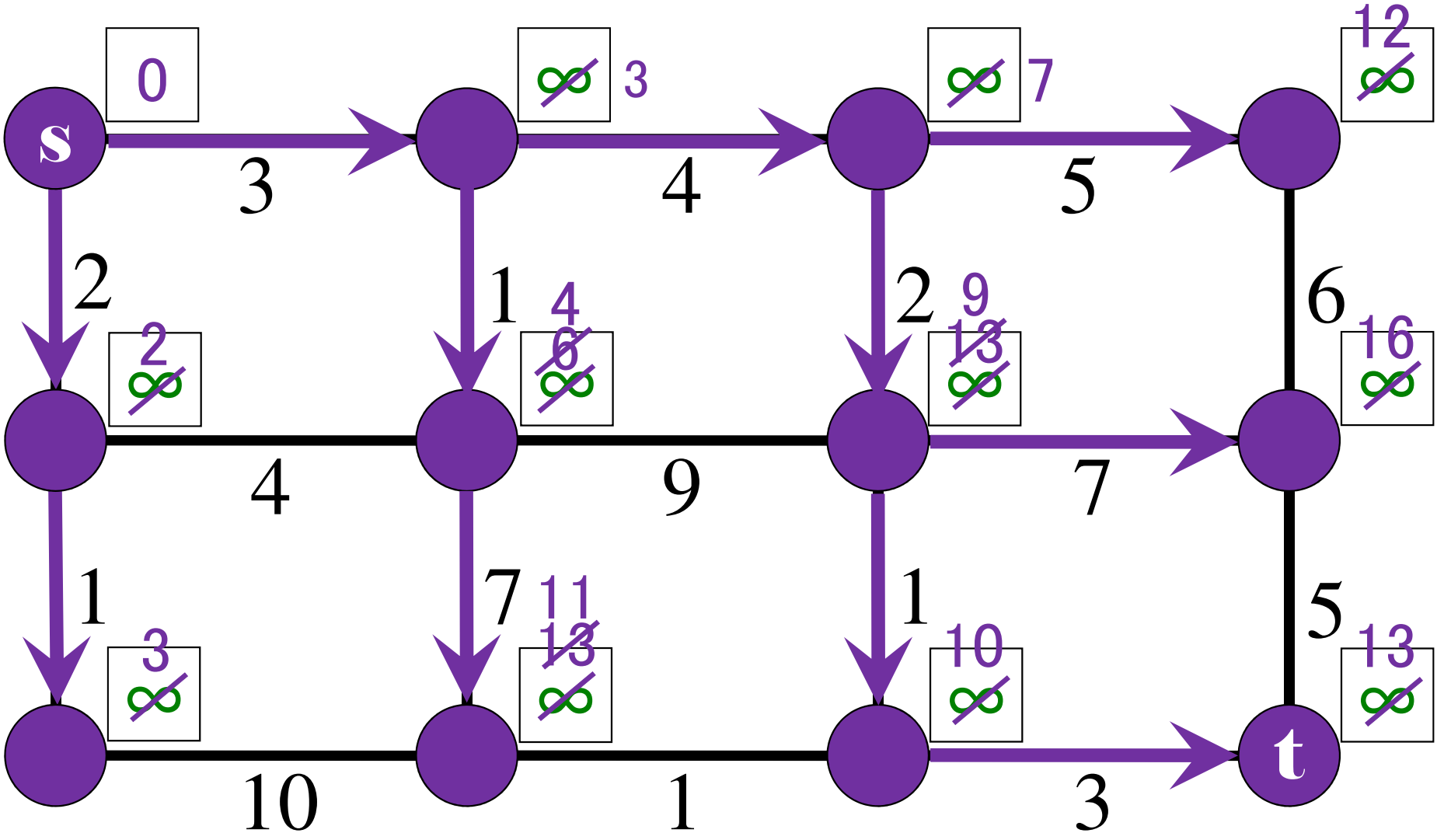
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



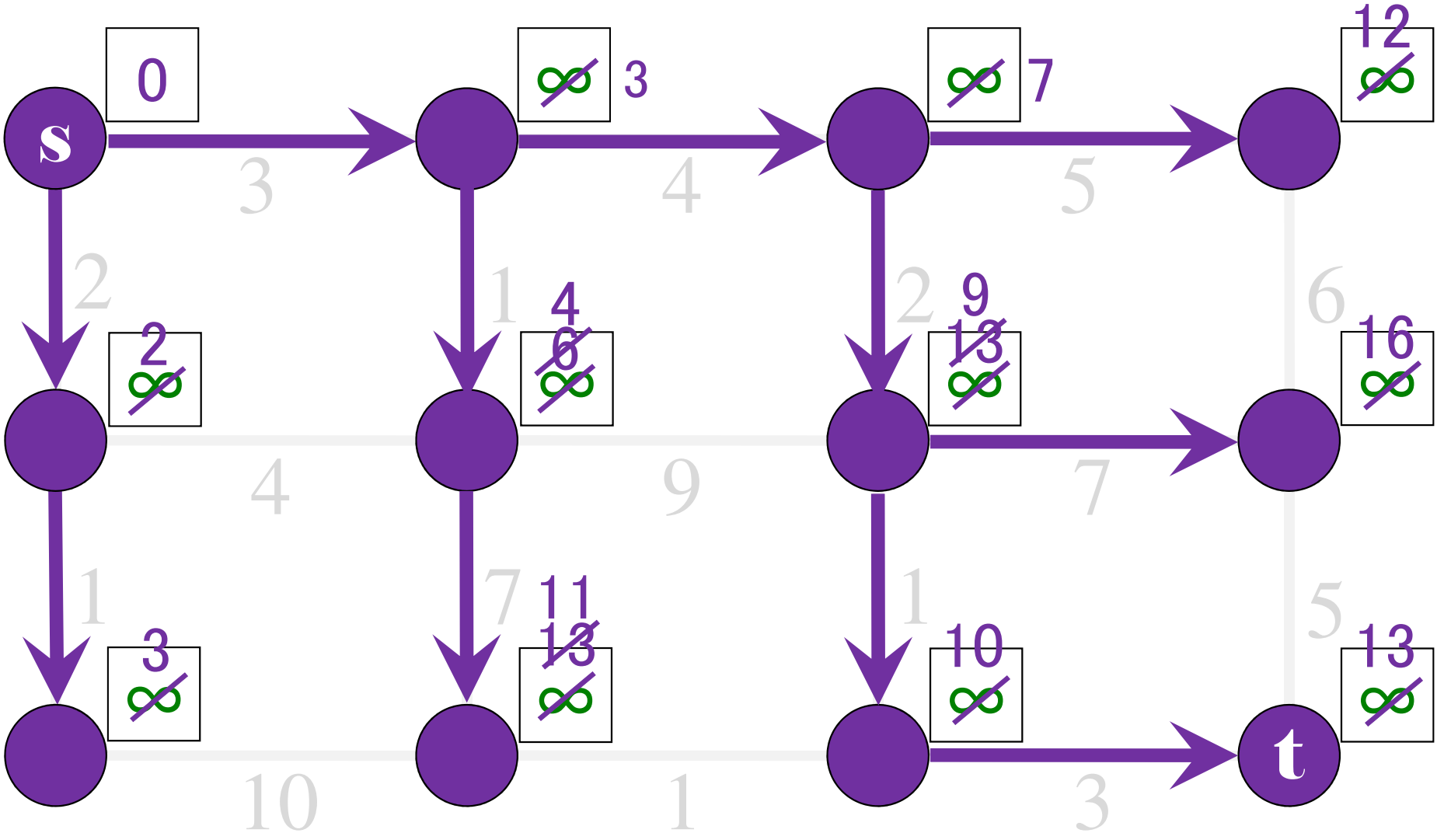
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



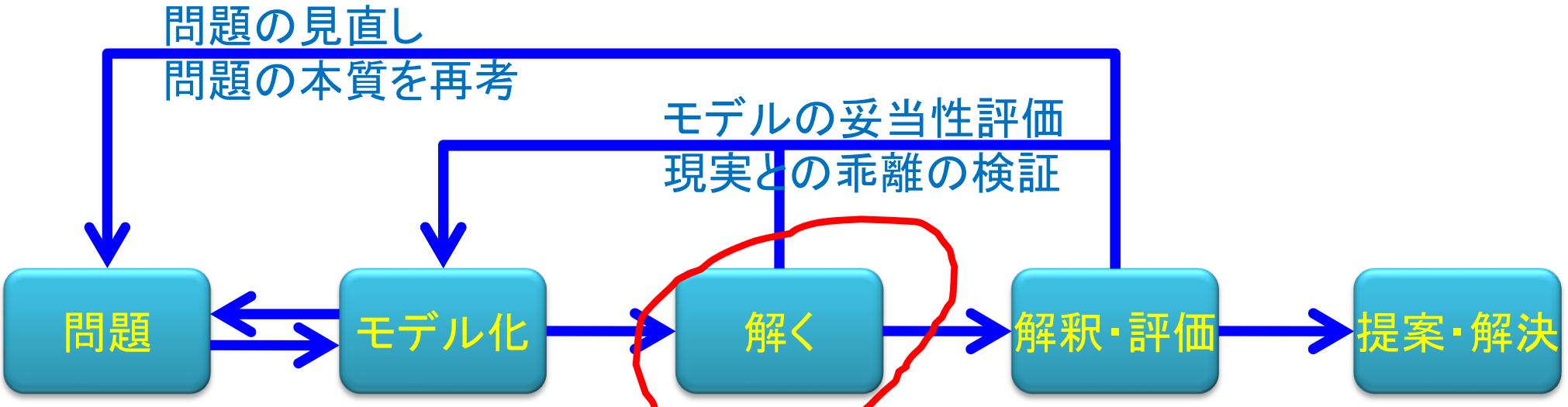
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

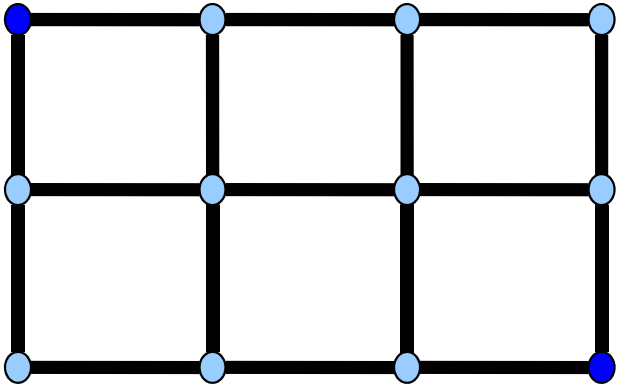
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

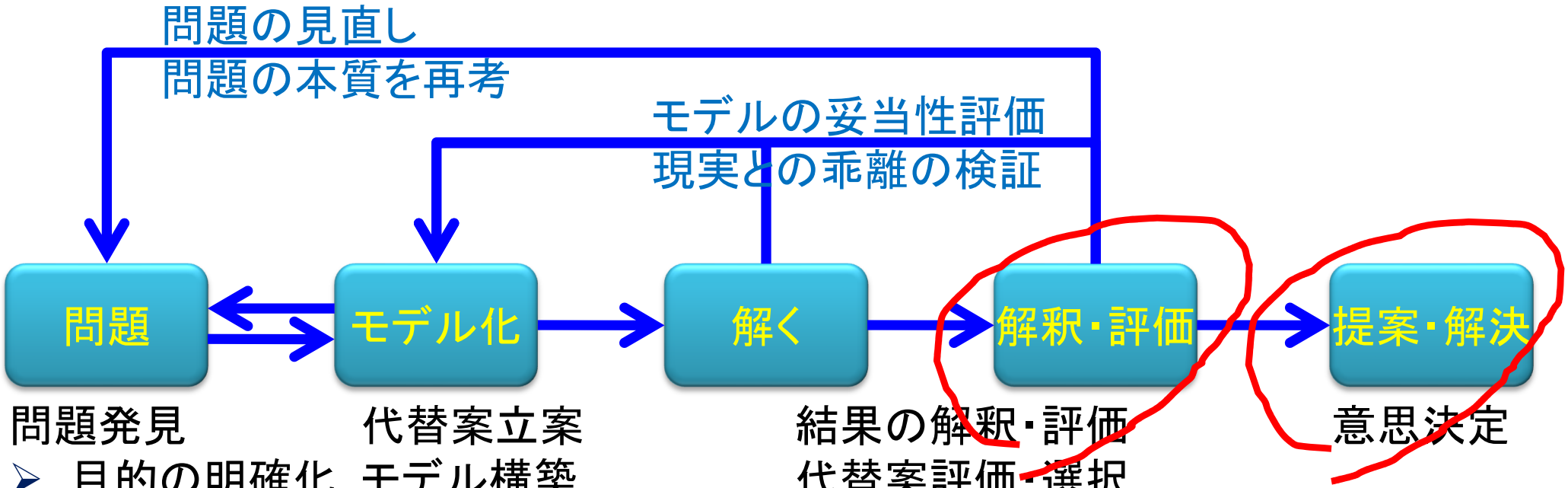
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

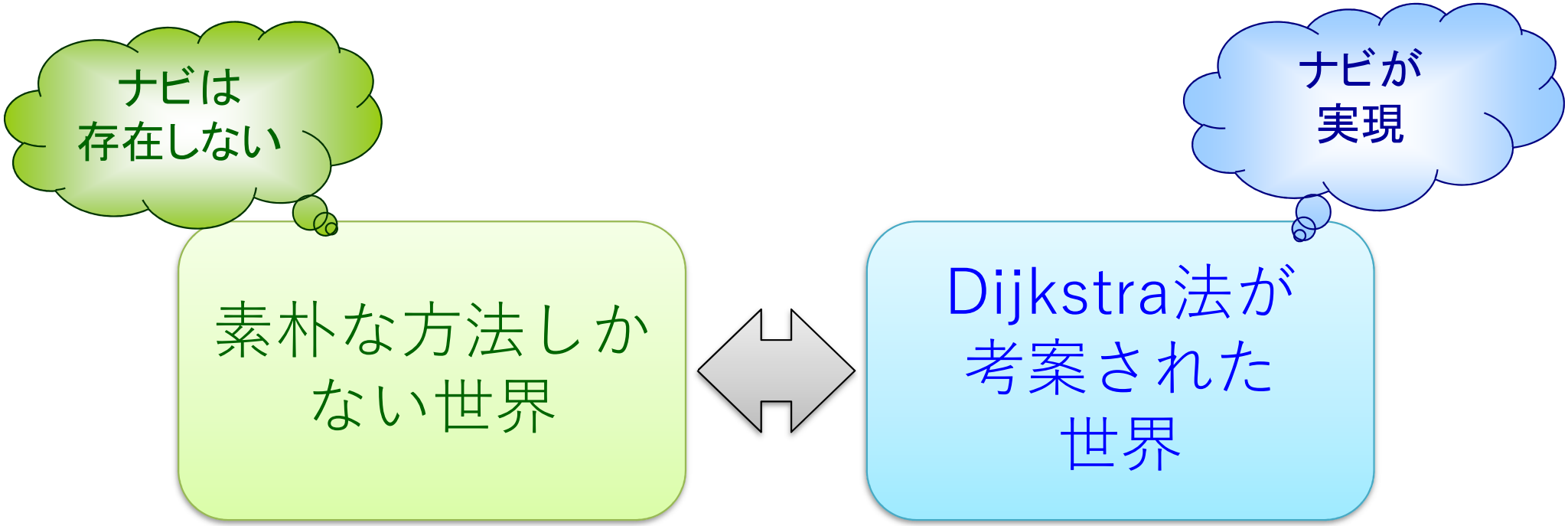
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

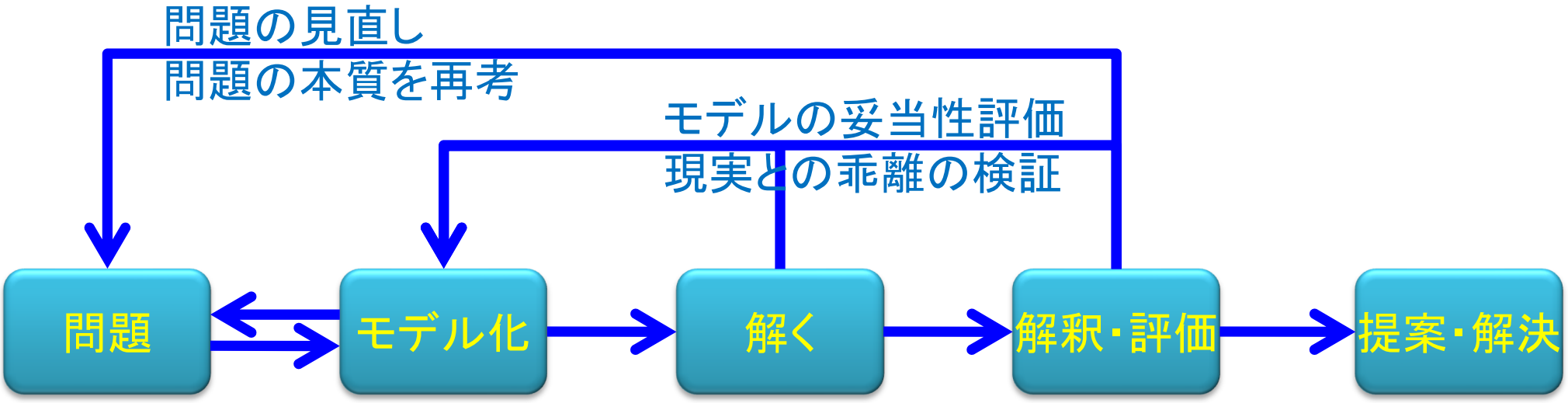
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



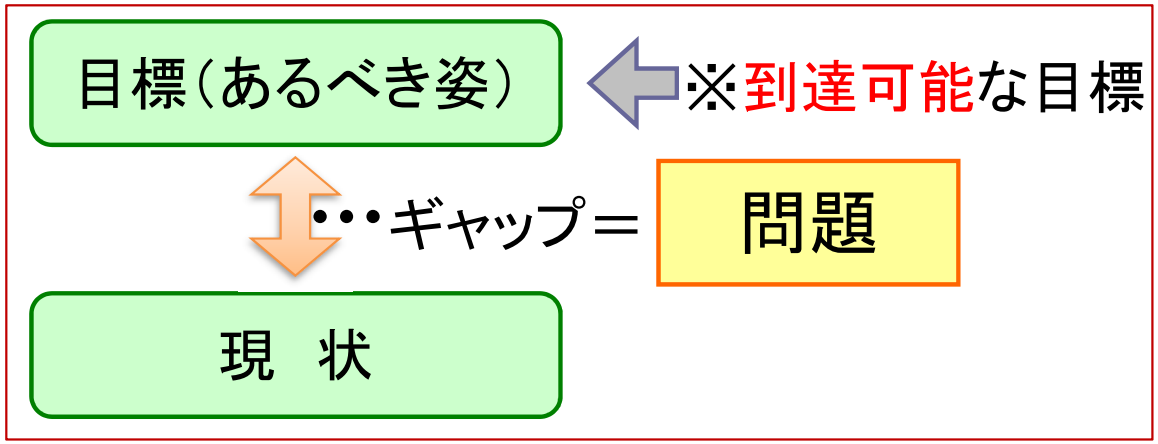
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

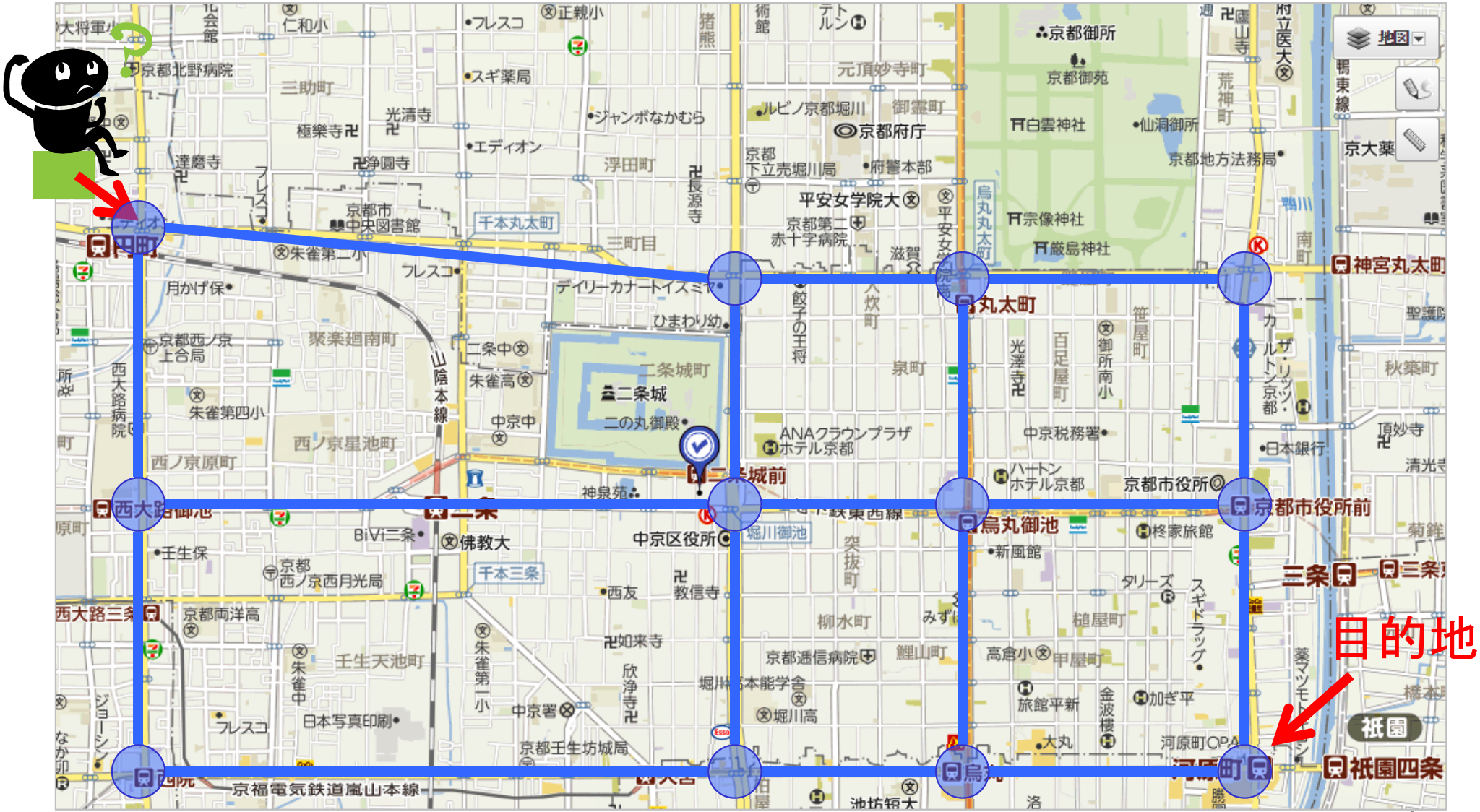
意思決定

問題の定義



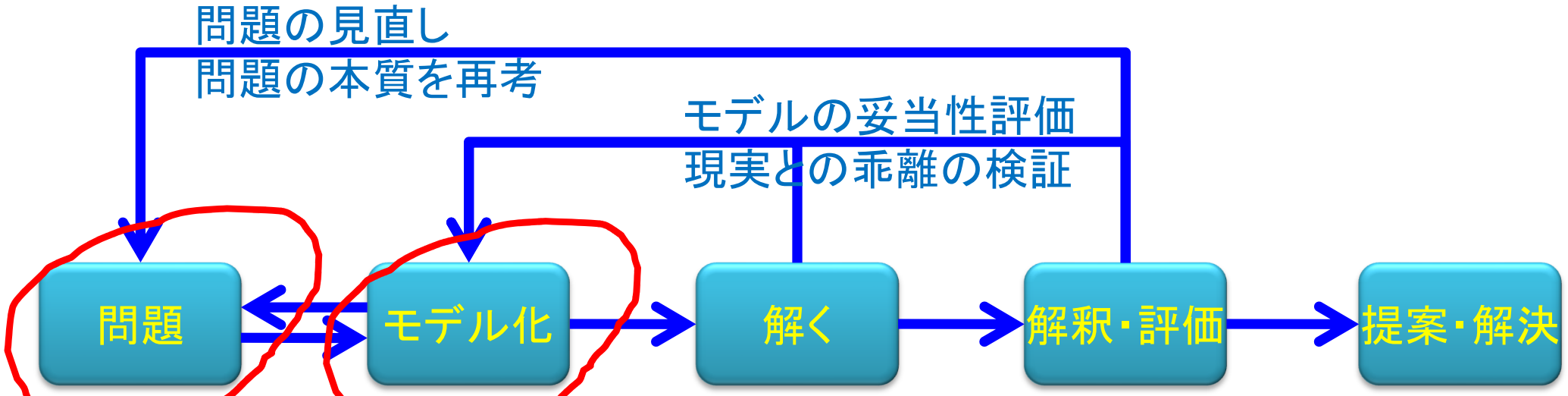
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

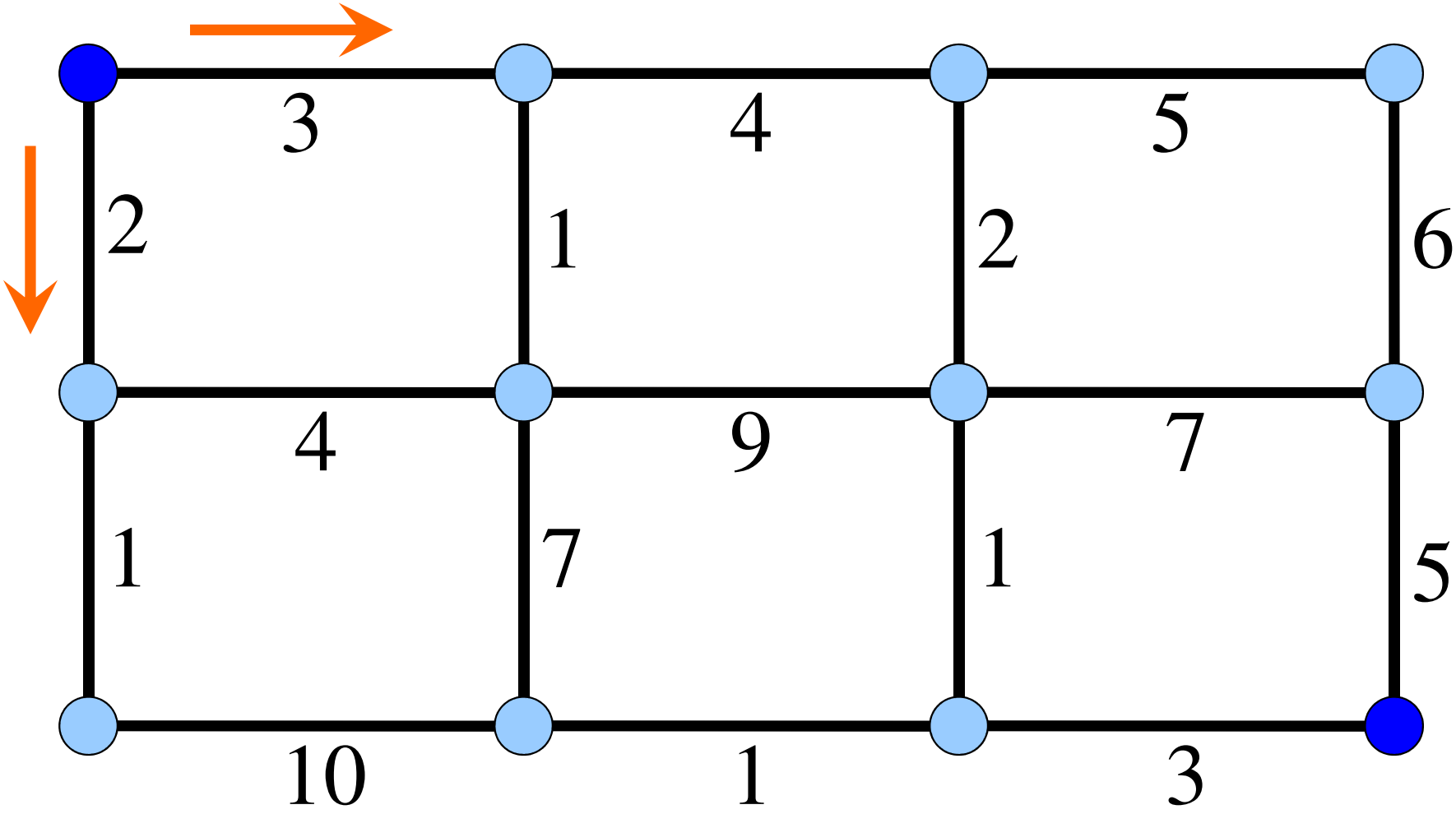
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

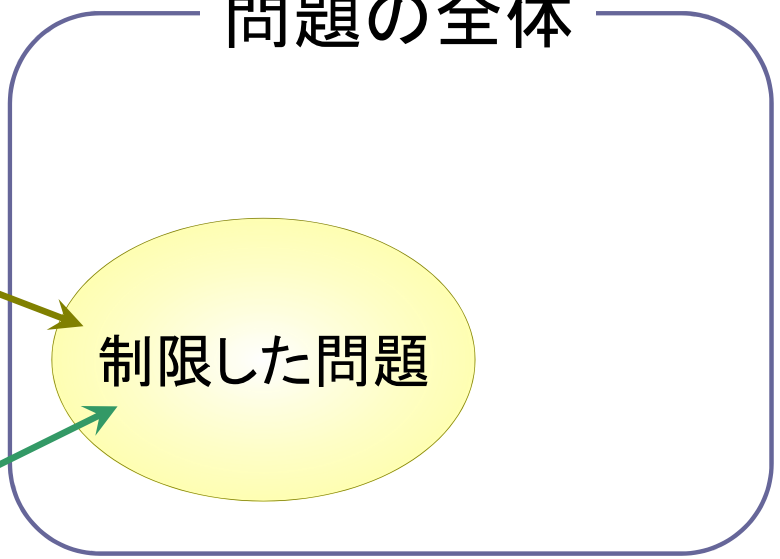


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

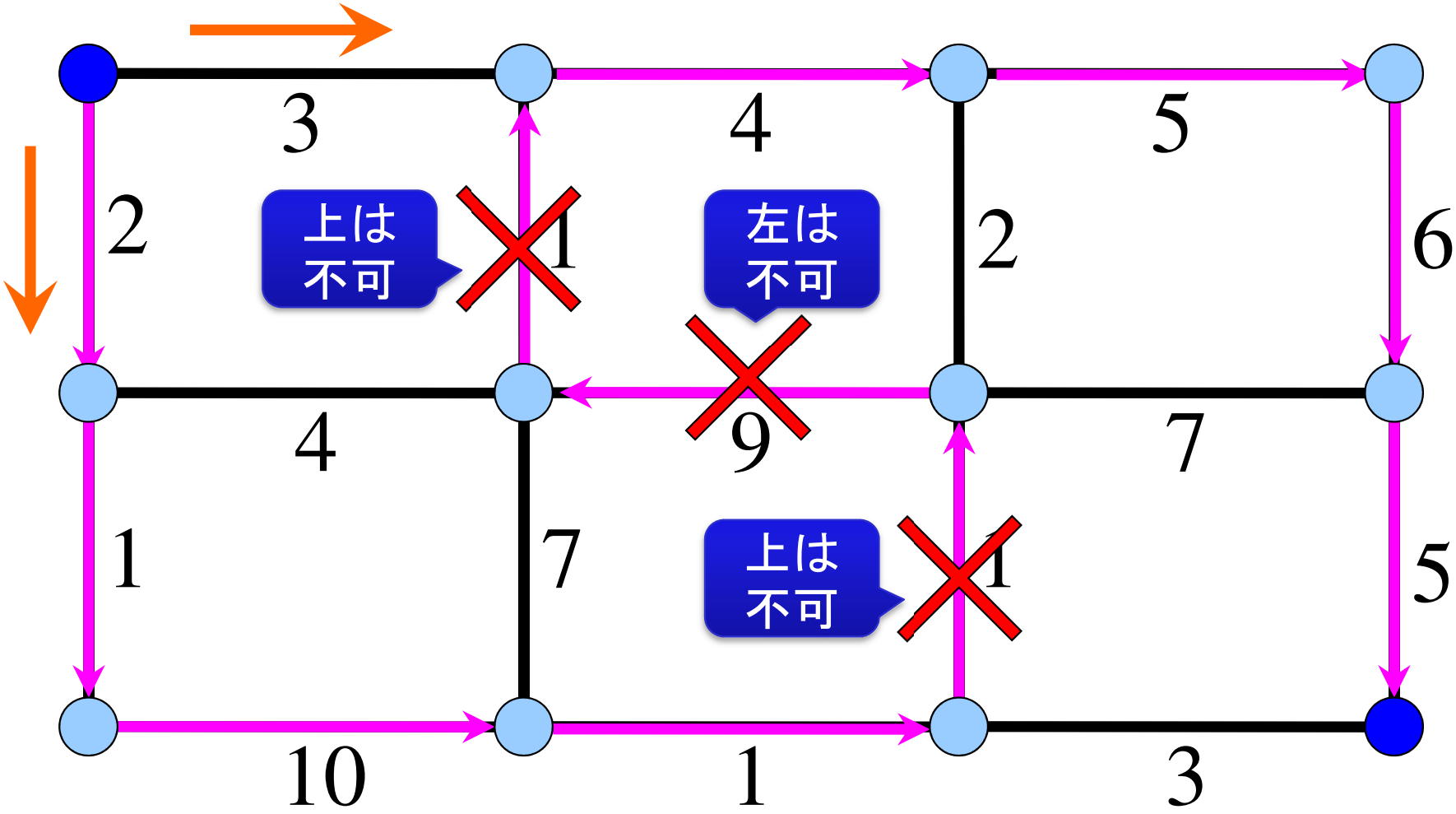
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

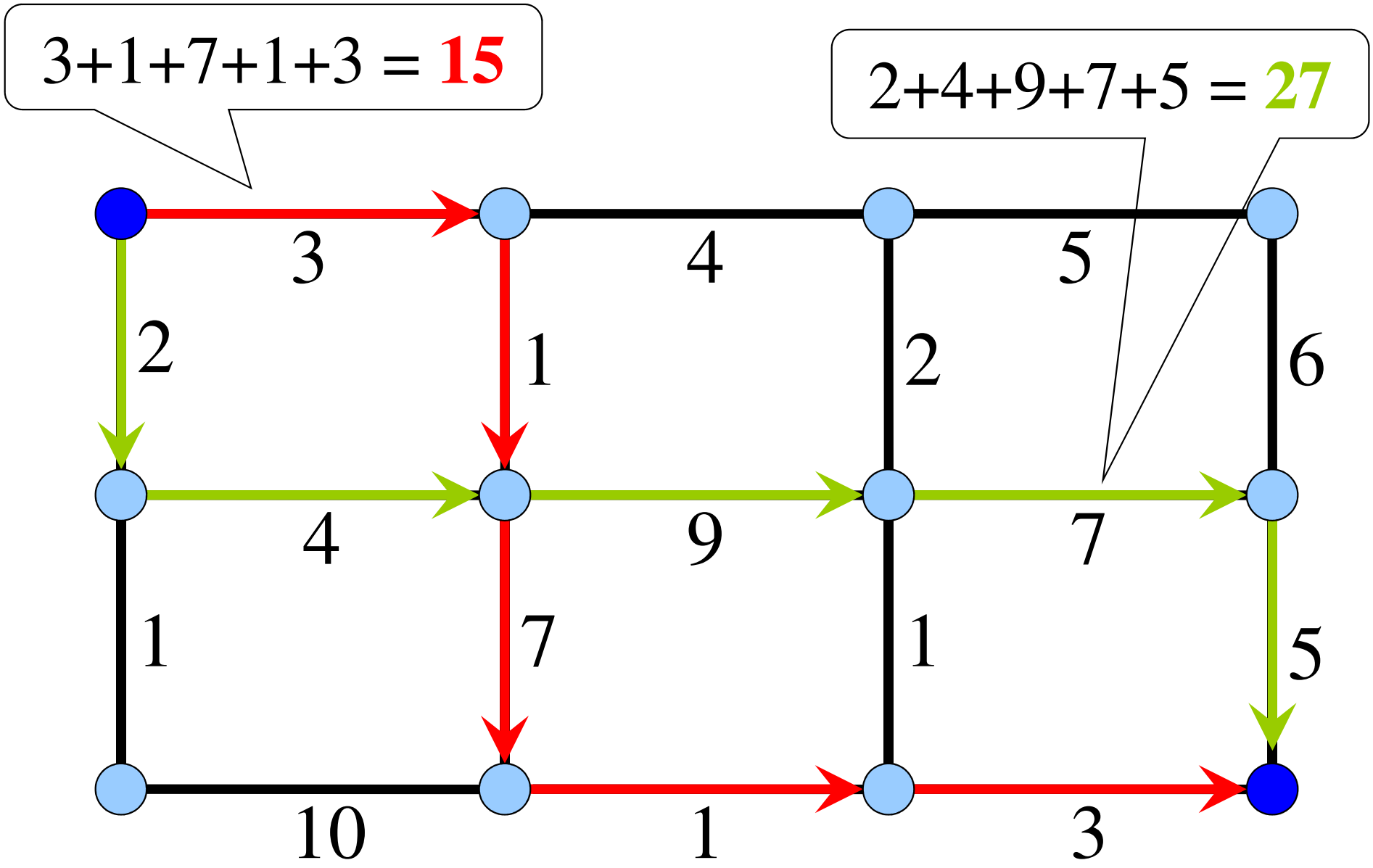
難しいなら易しくすればいいのさ！

制限した問題

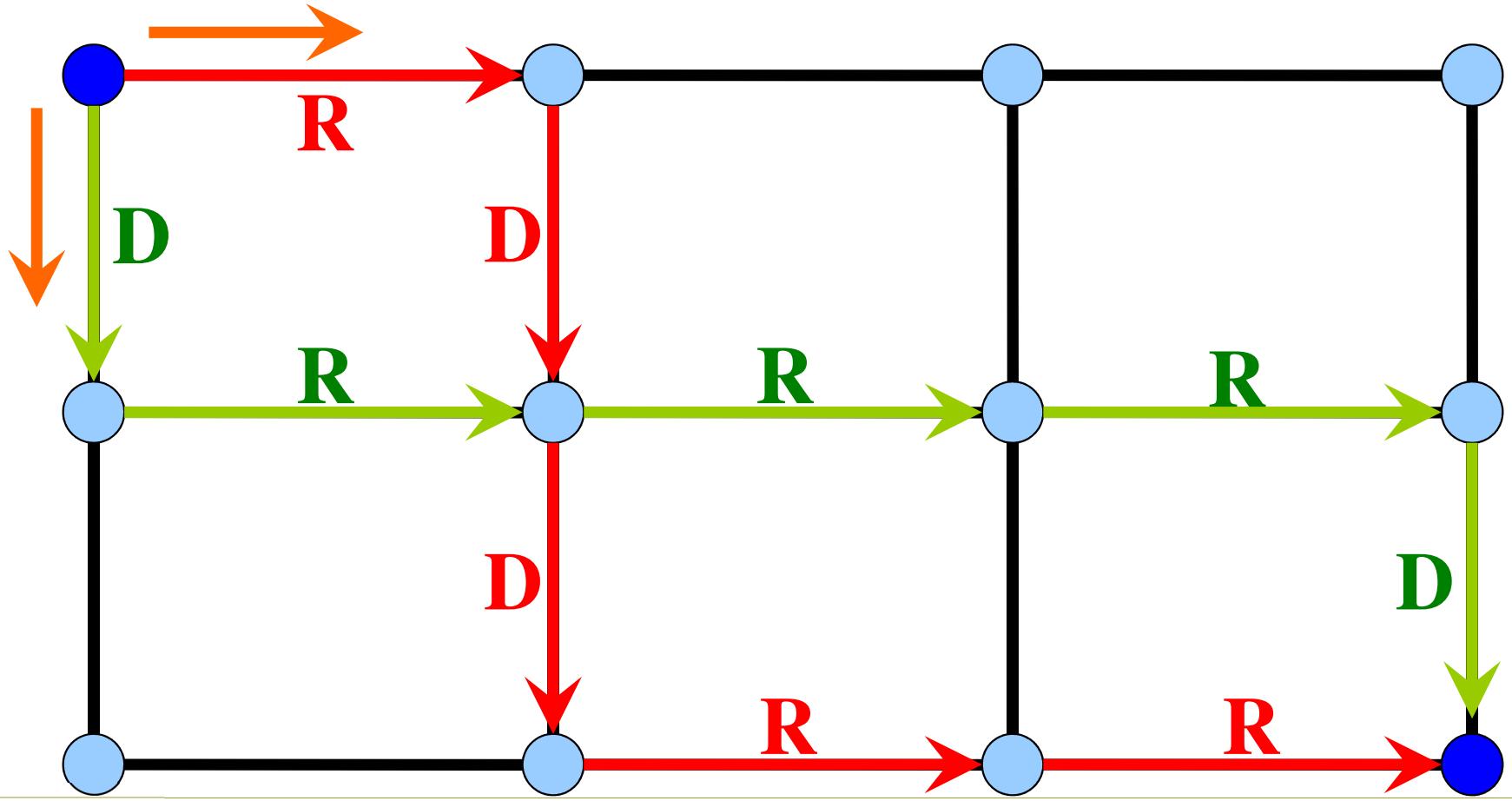
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



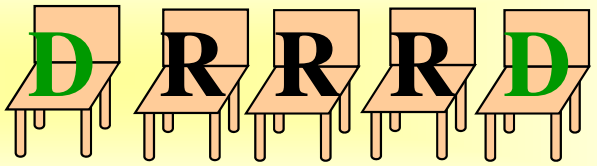
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



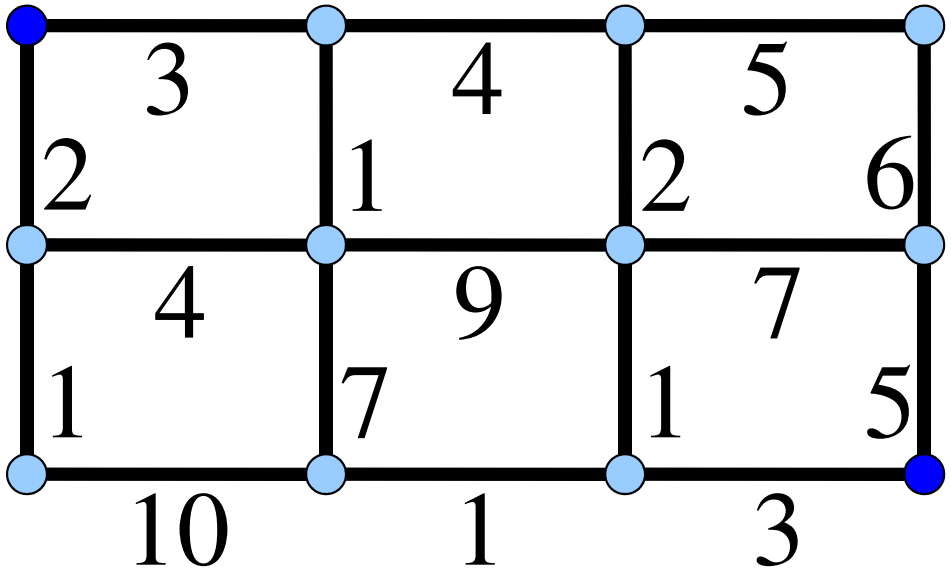
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

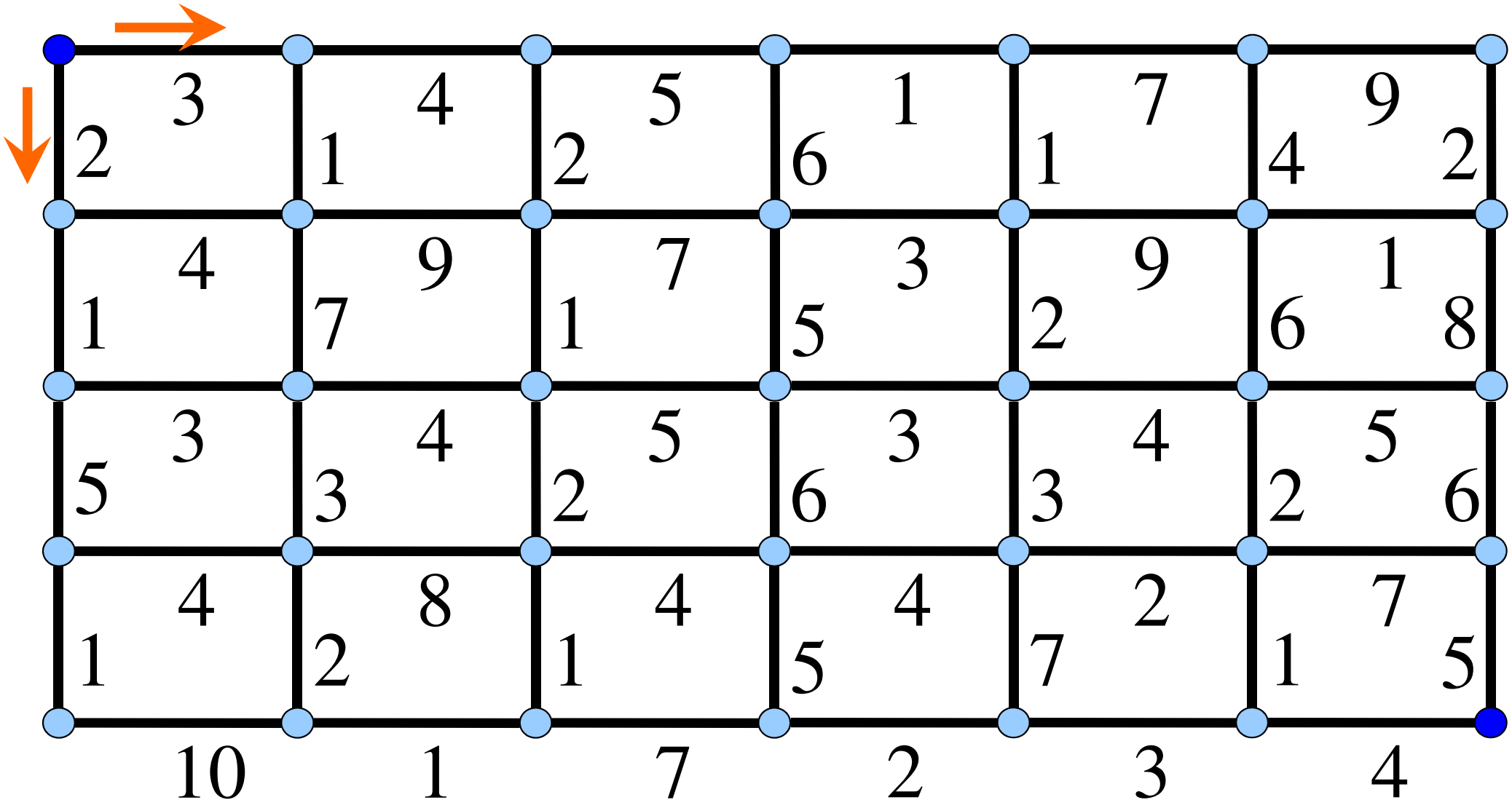
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

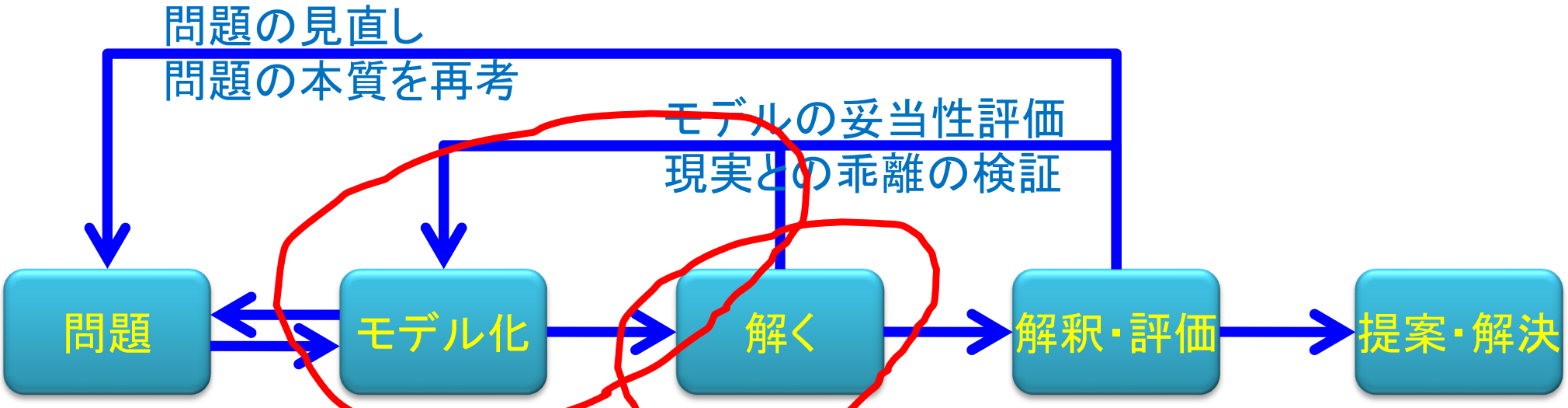
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

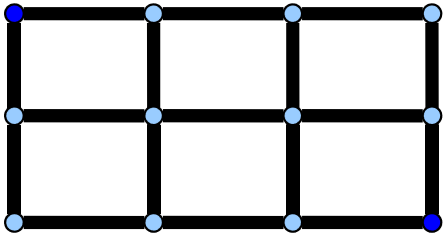
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

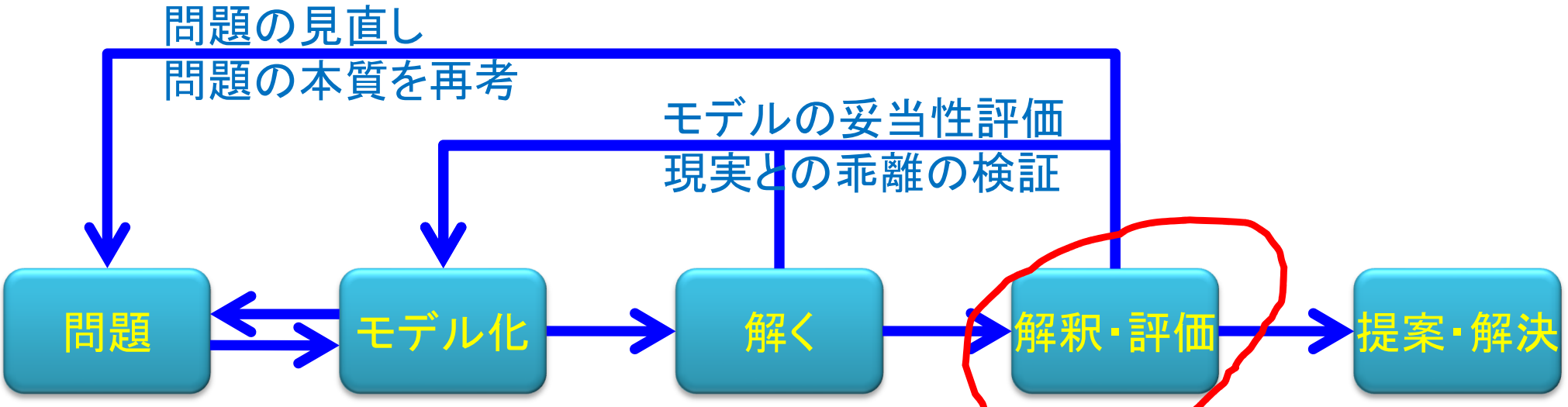
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

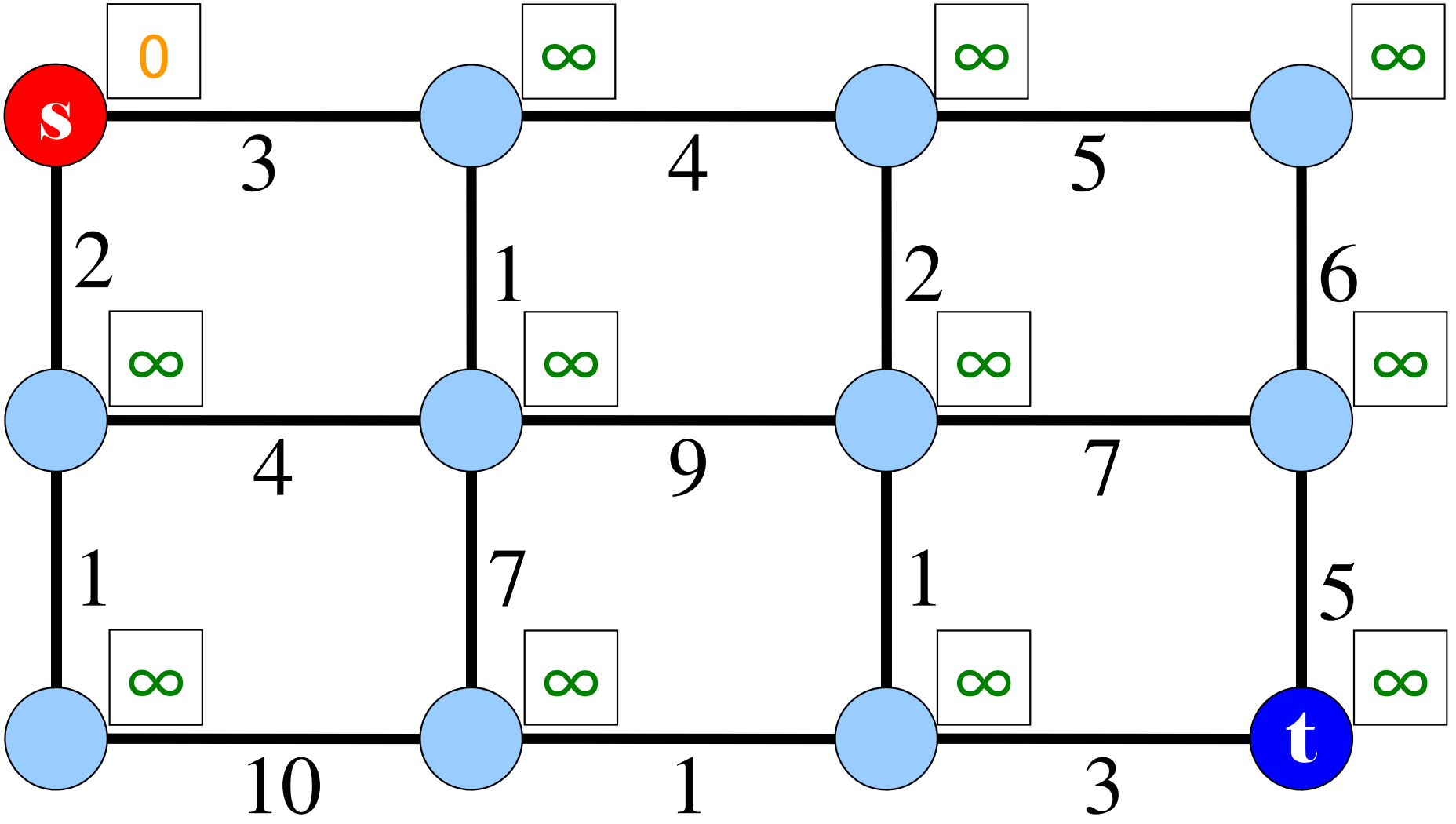
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

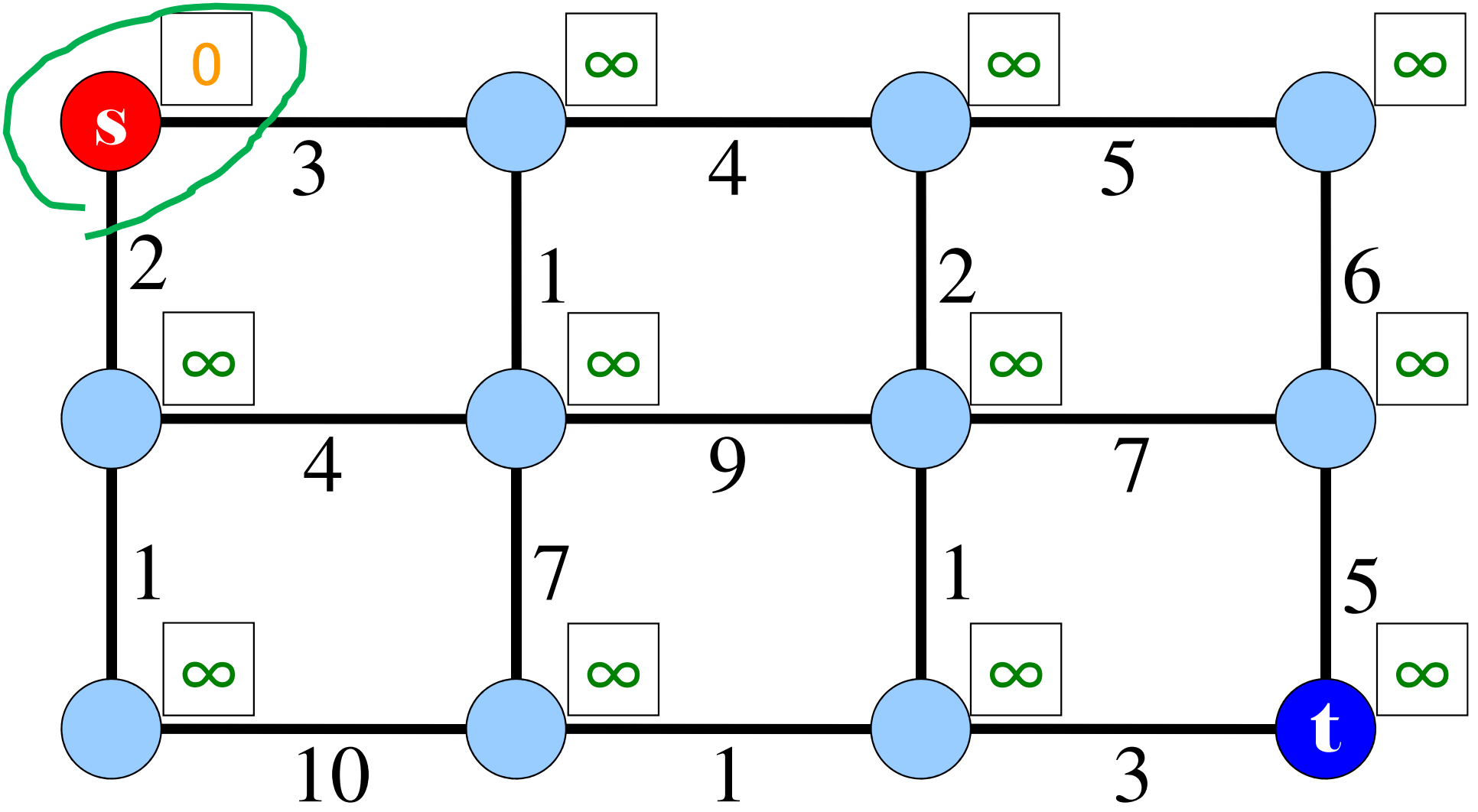
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



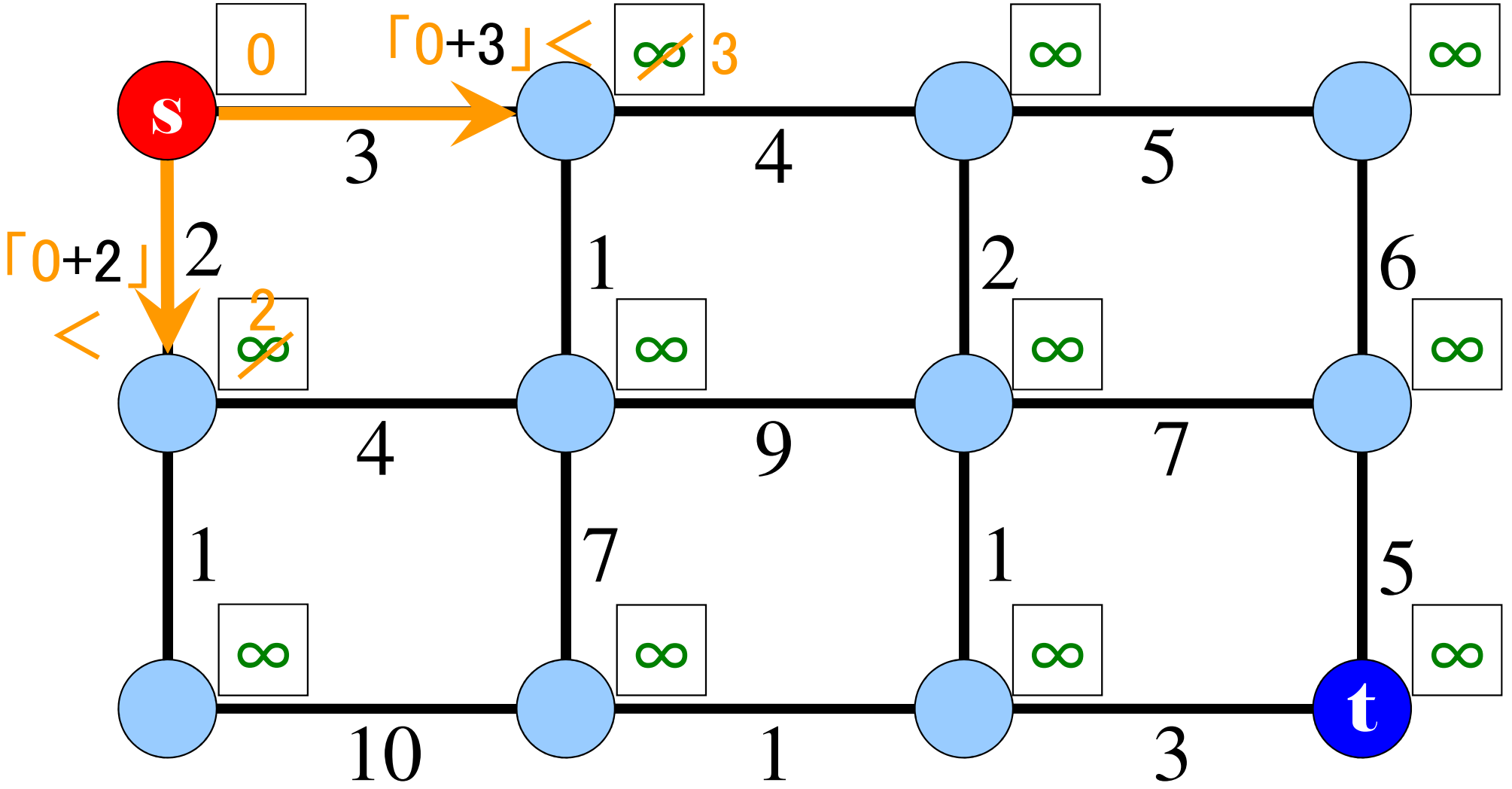
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

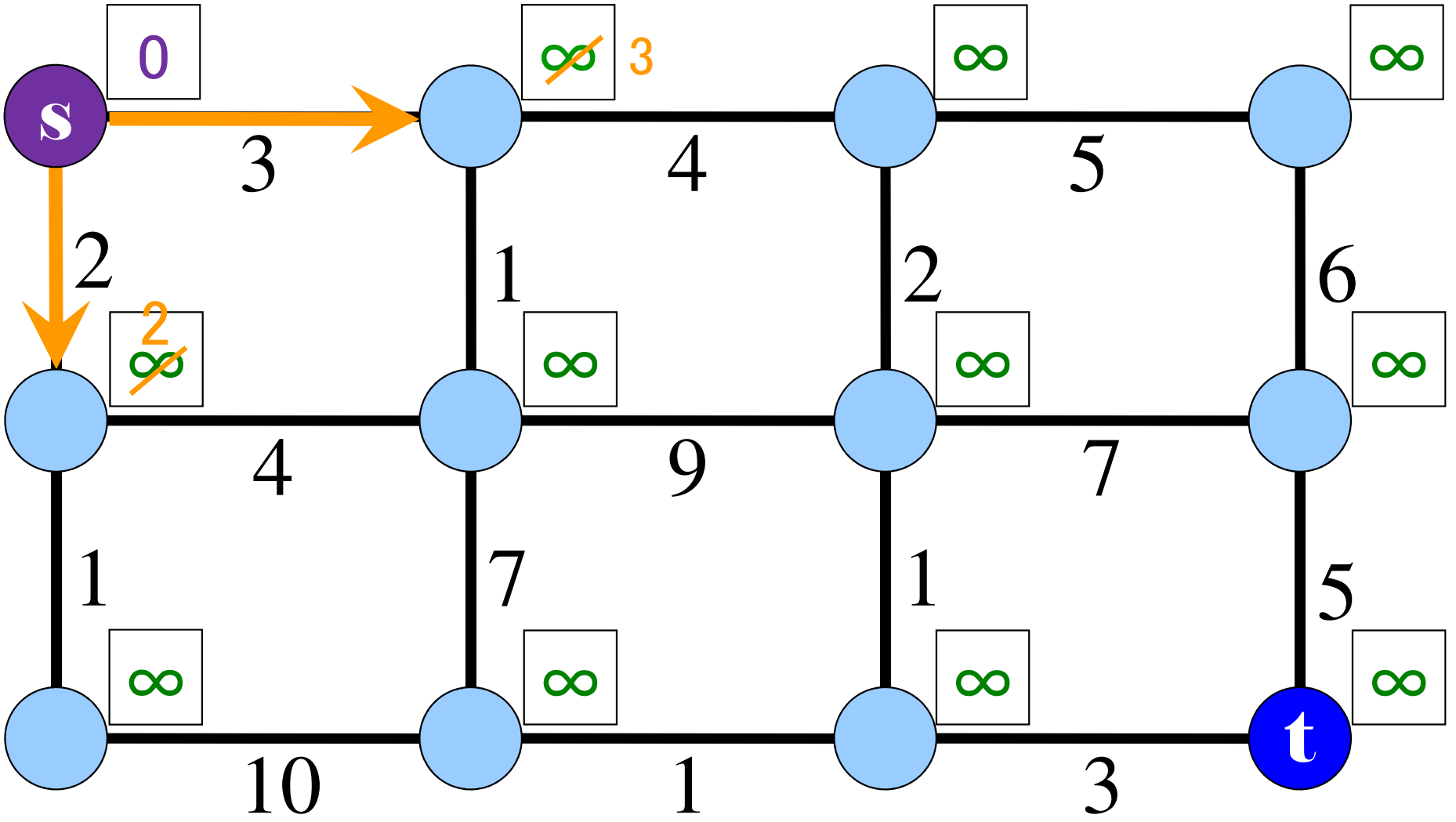


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

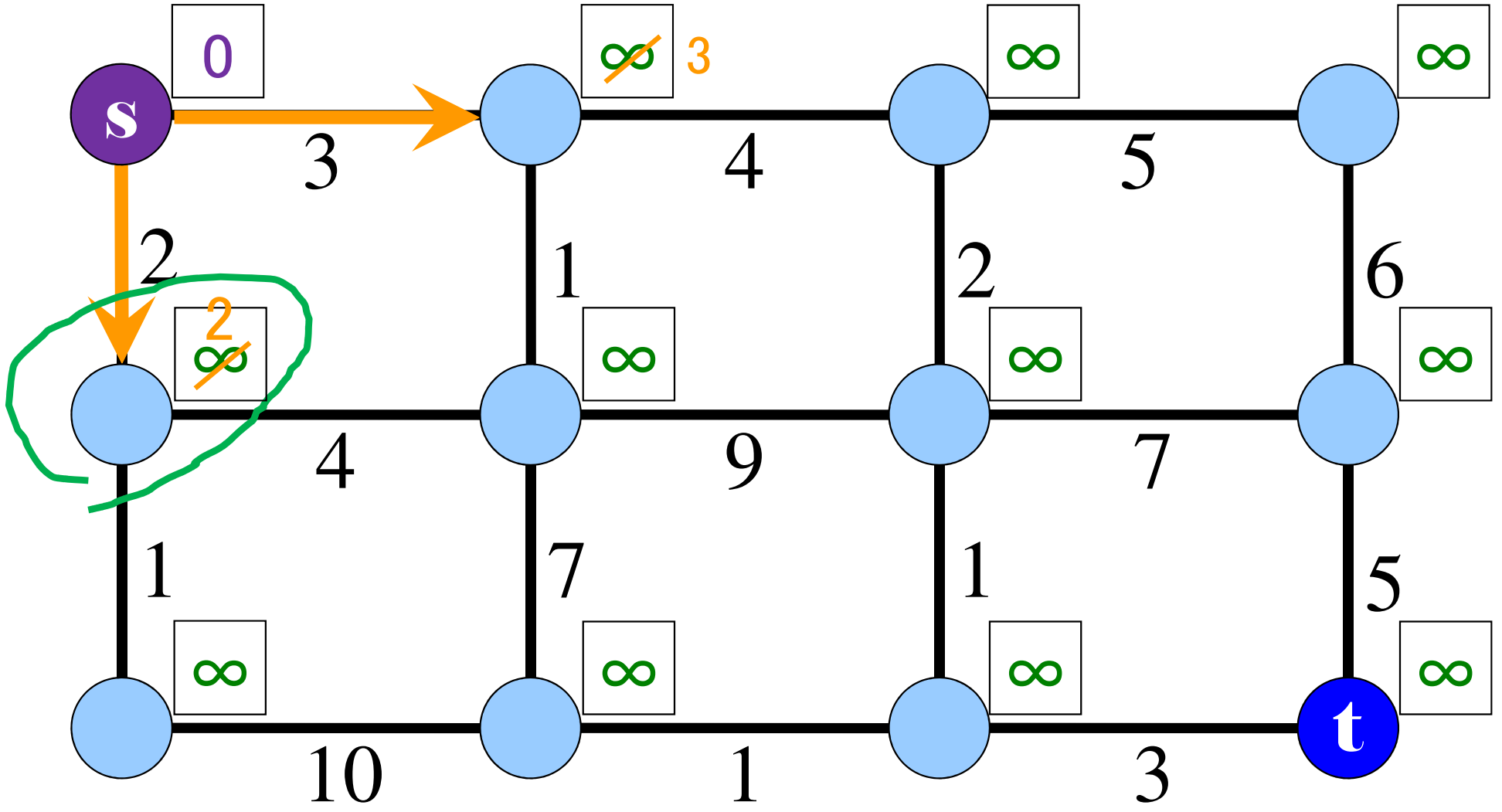
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



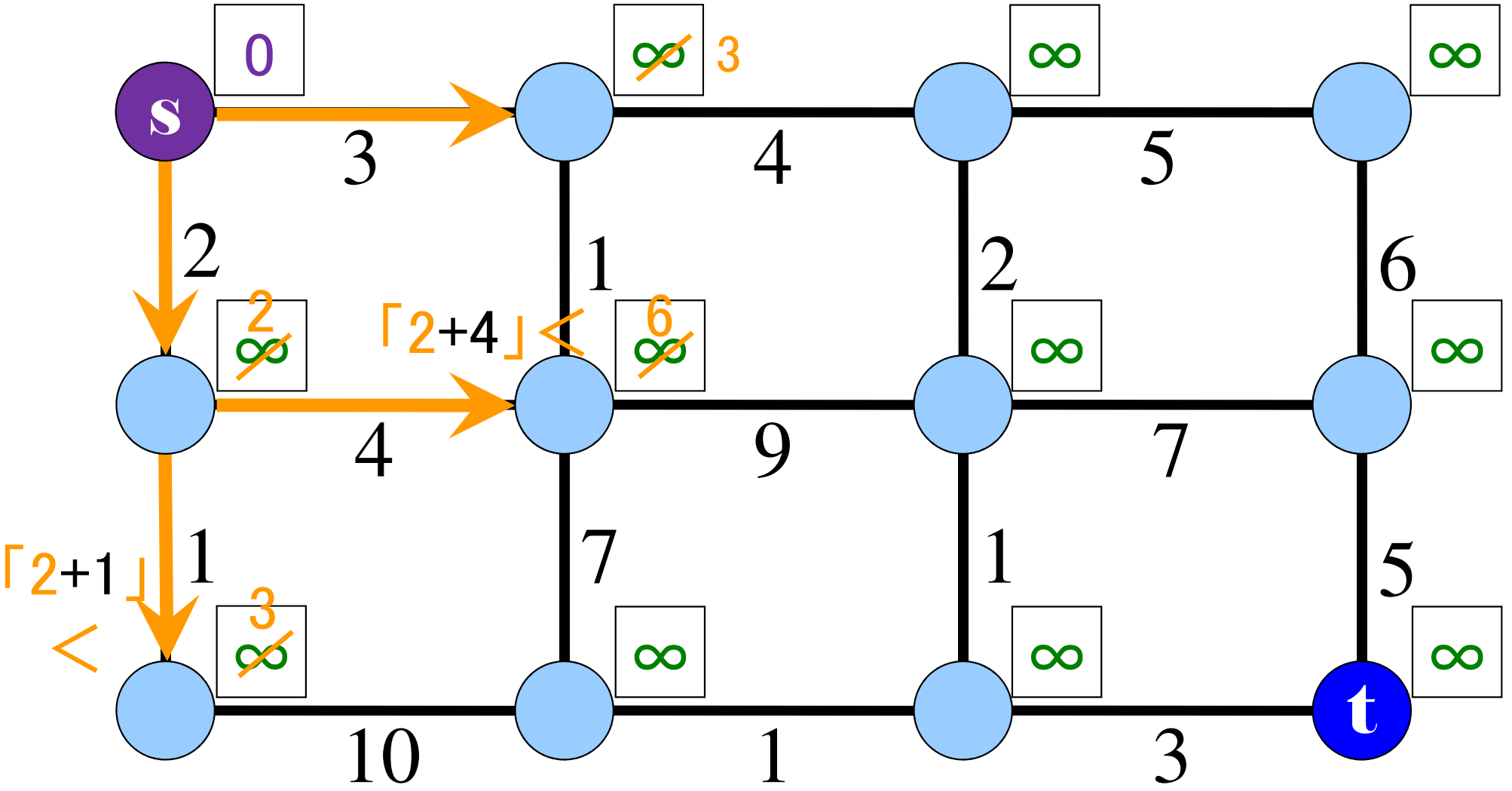
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

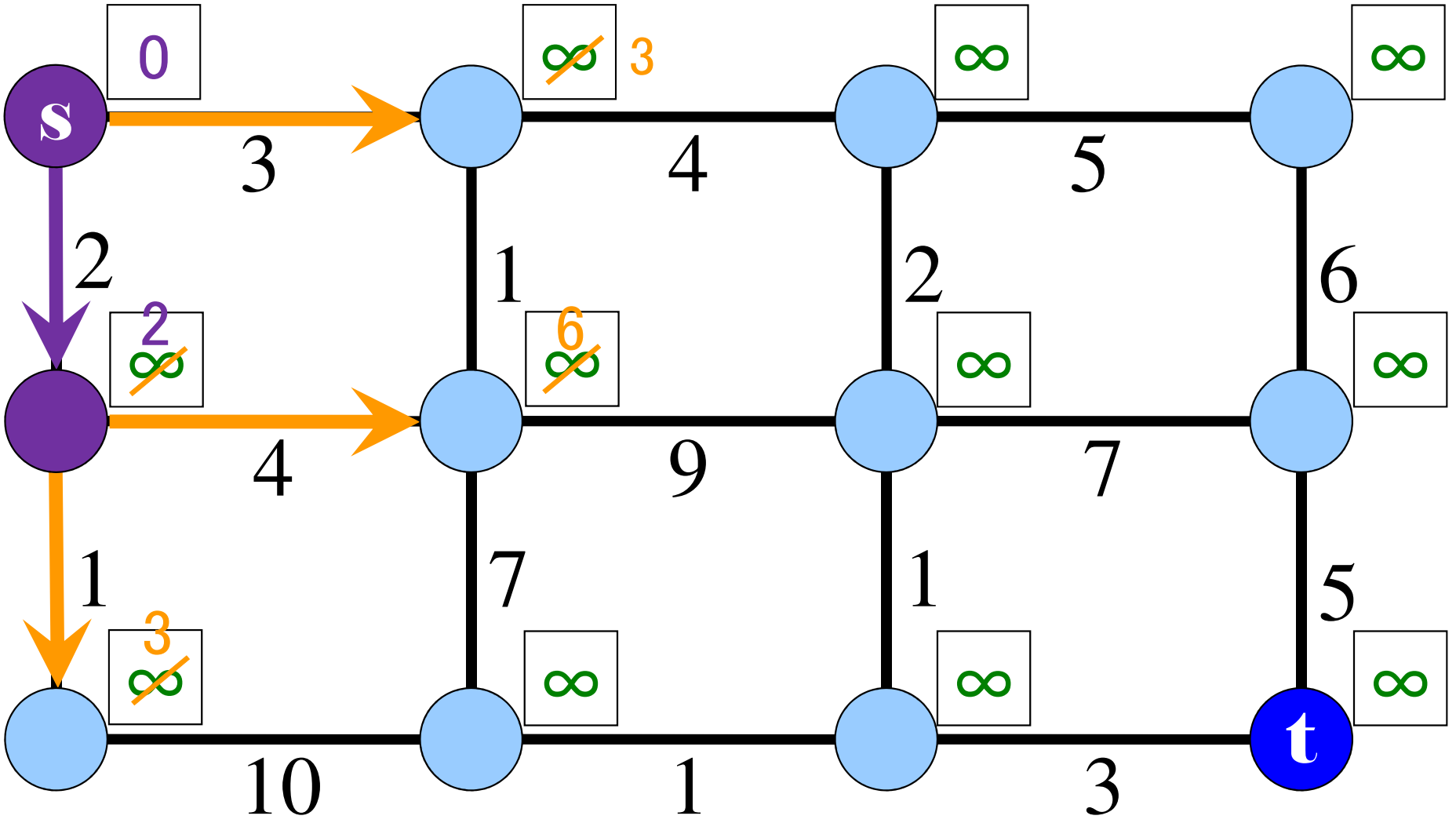


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

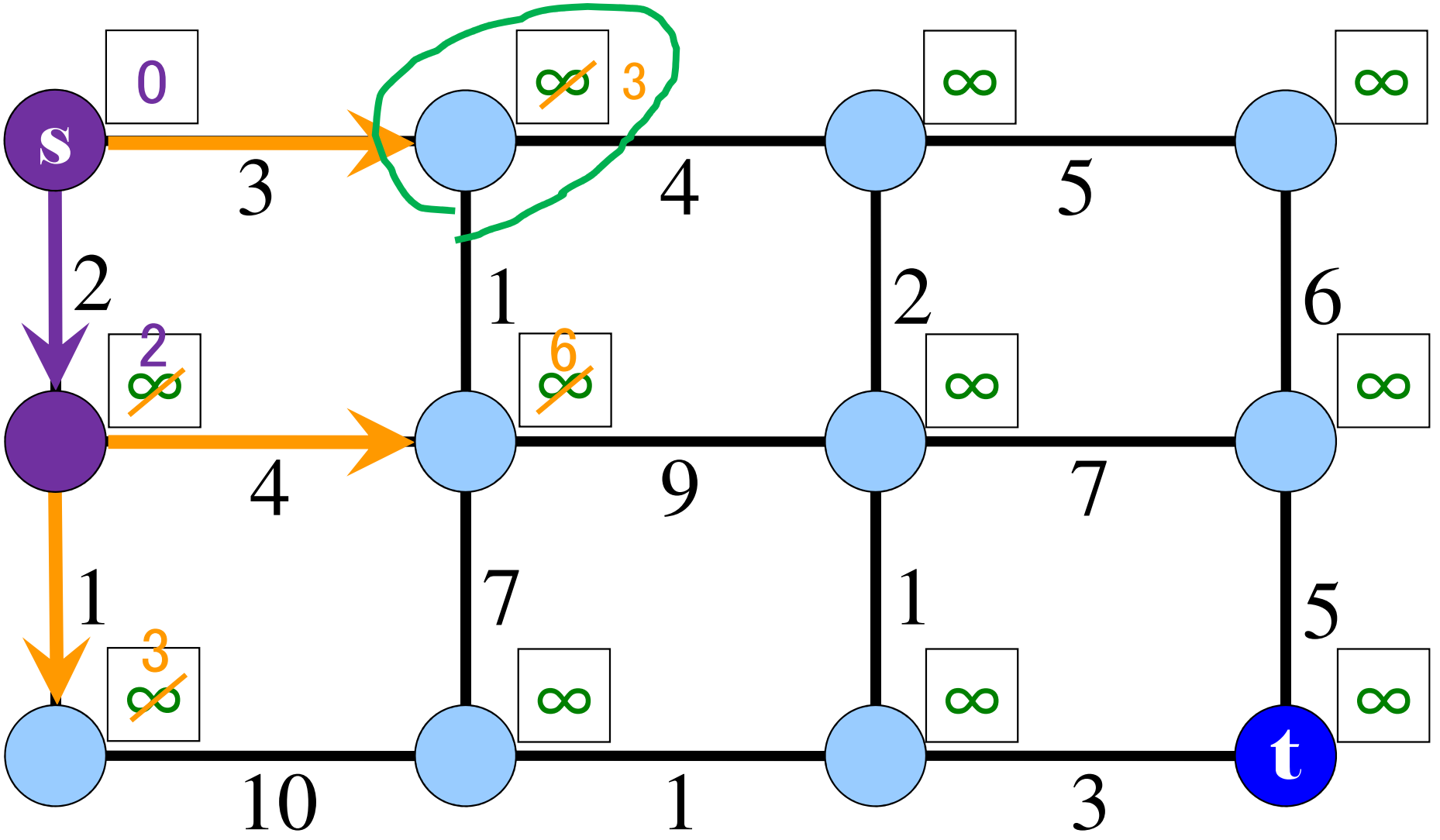
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



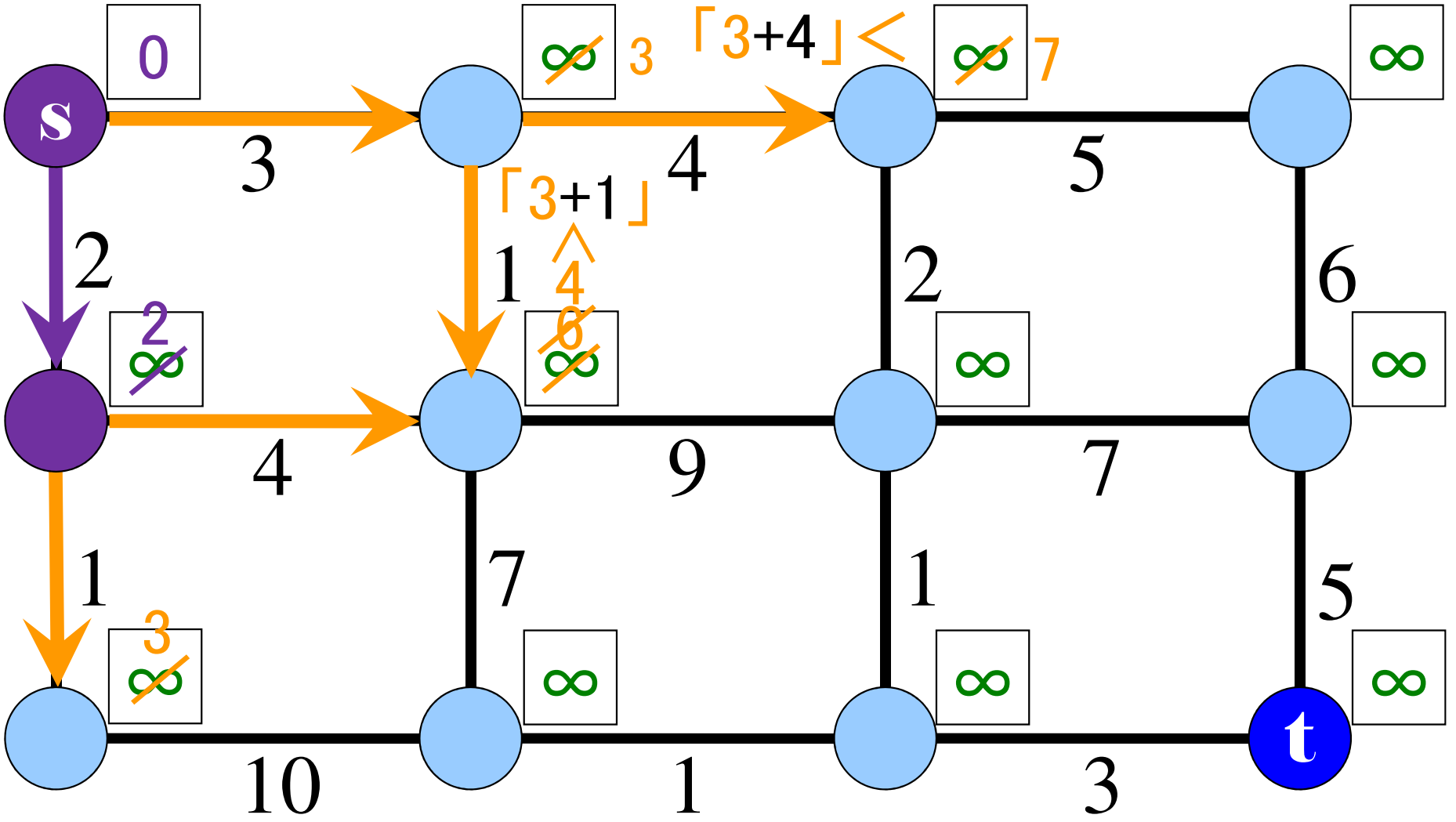
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

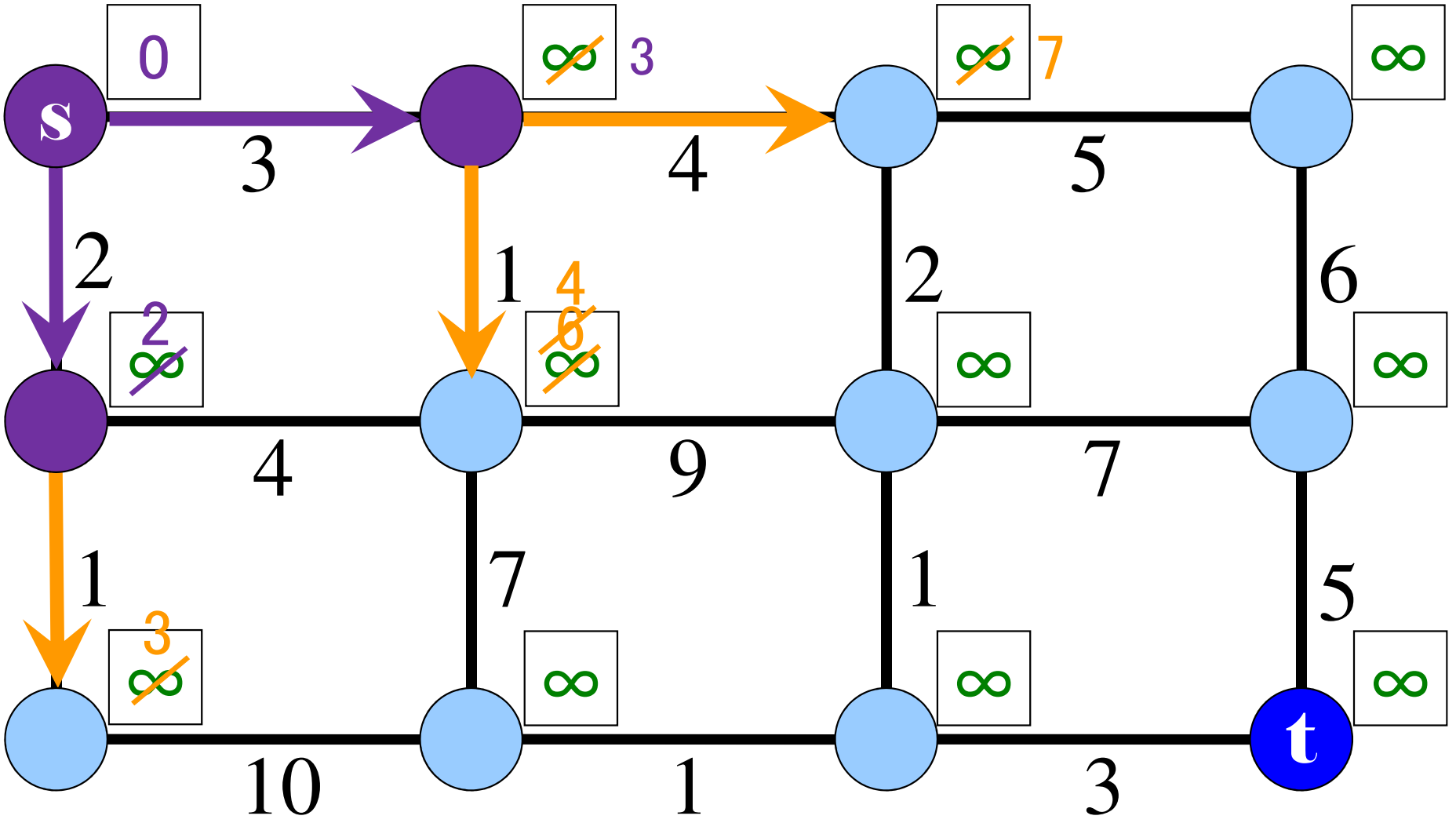


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

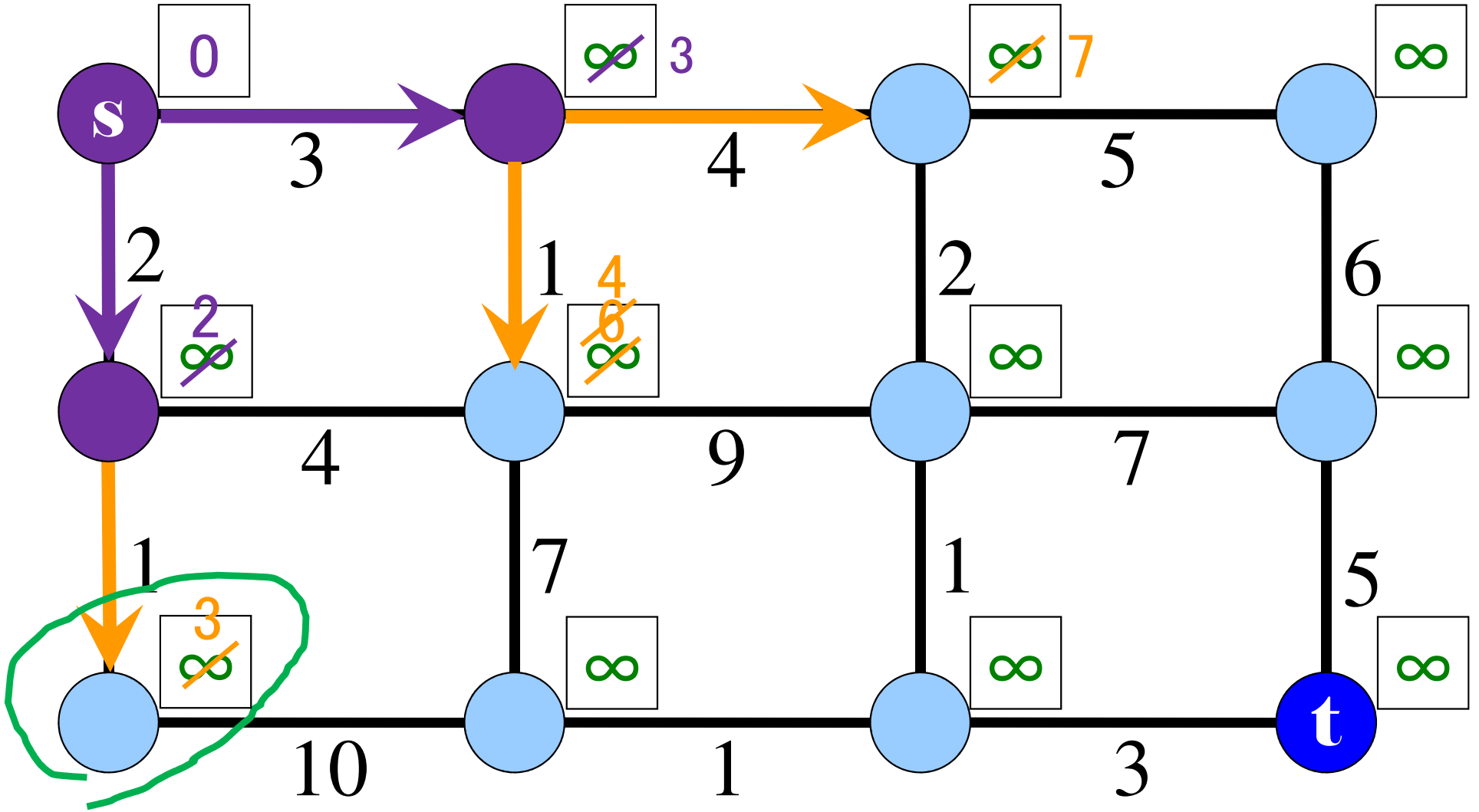
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



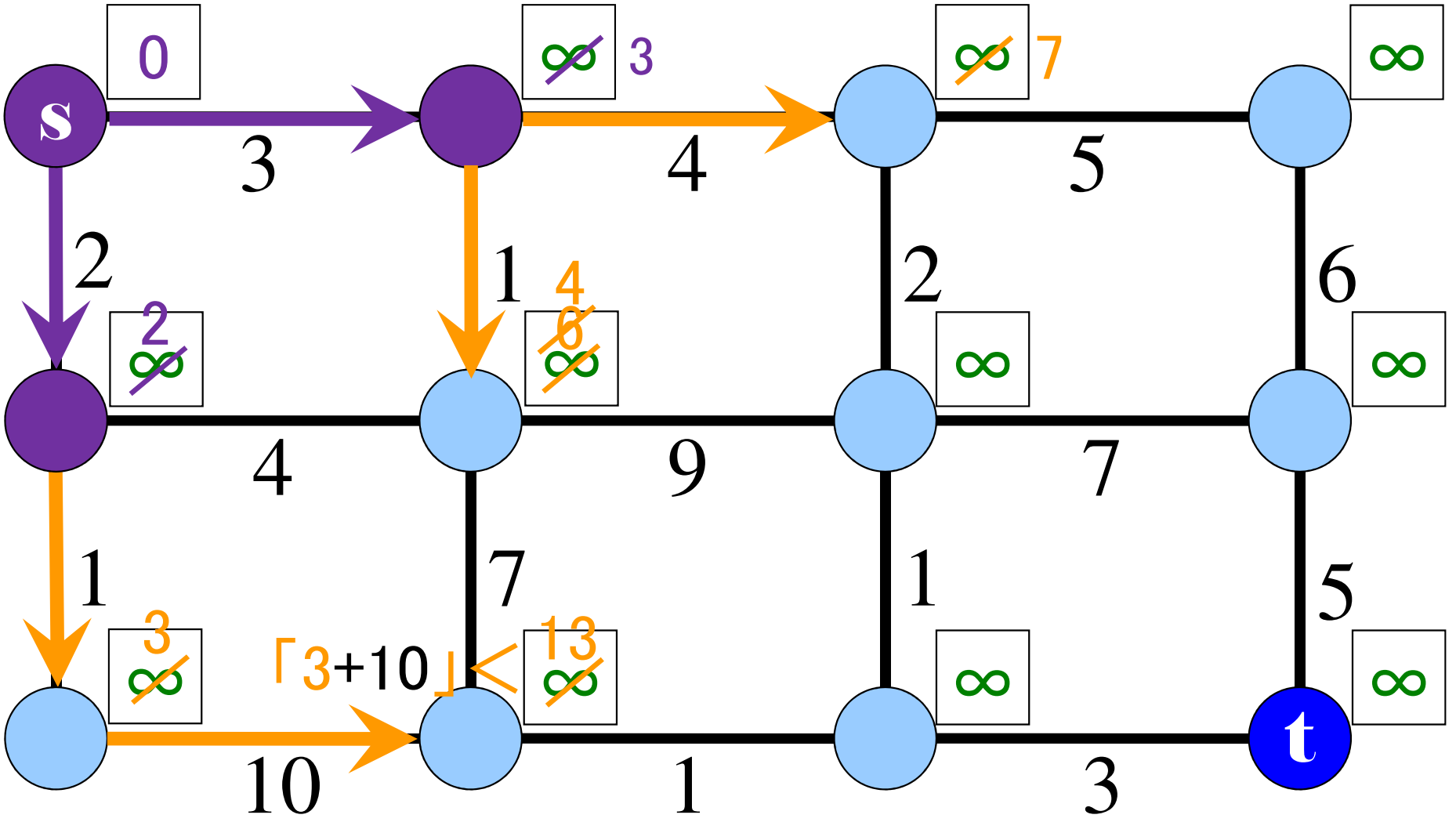
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

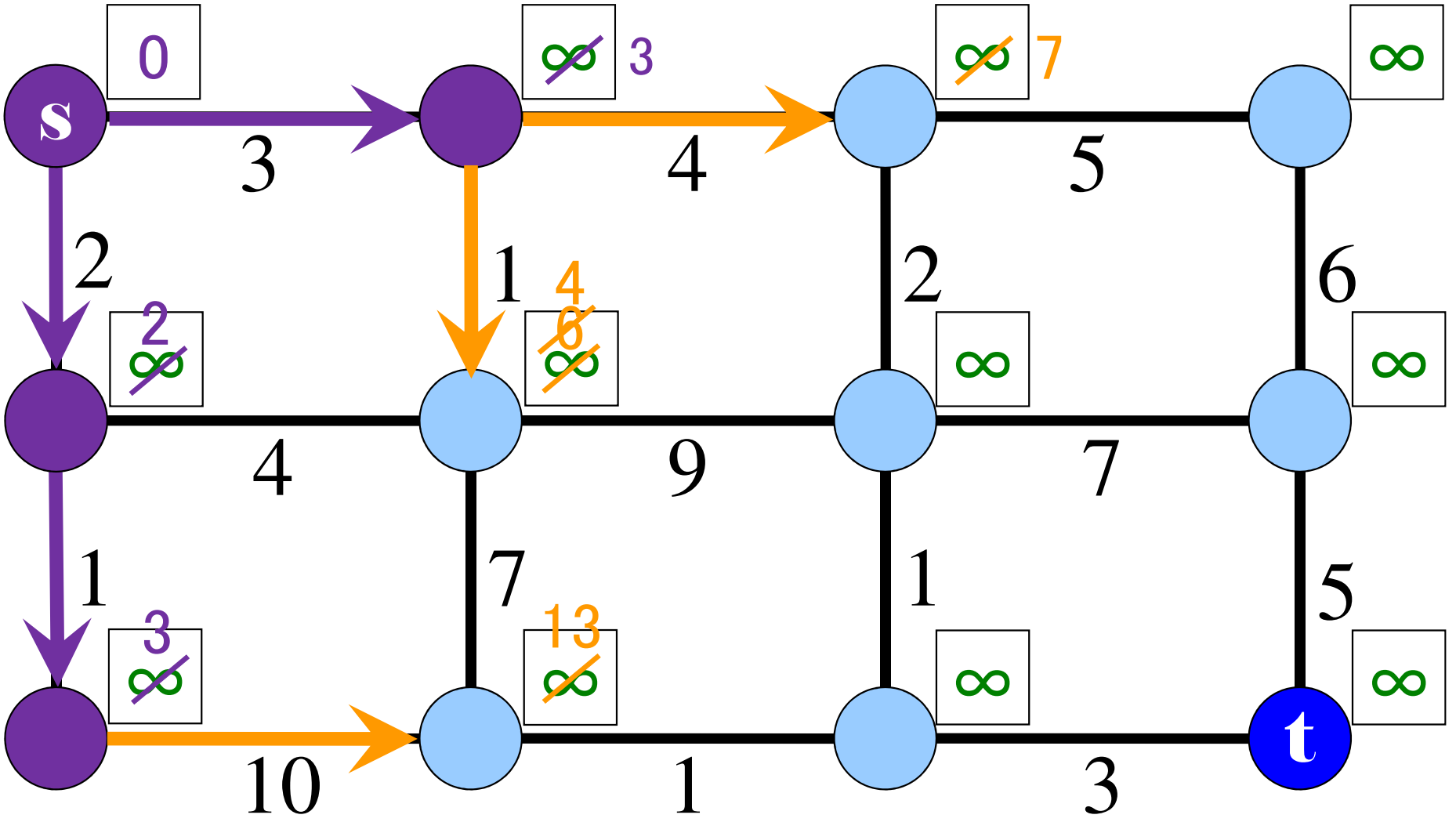


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

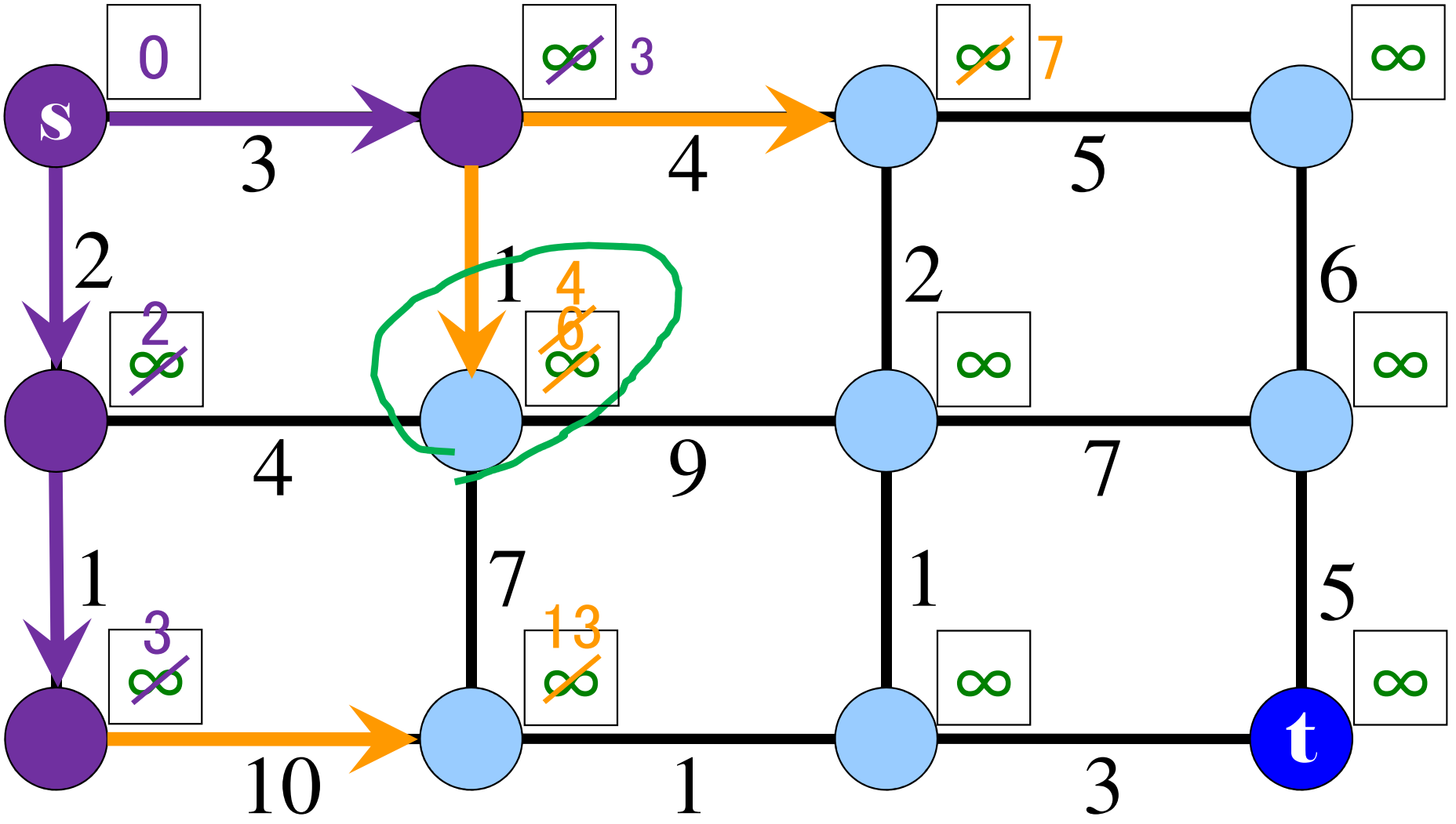
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



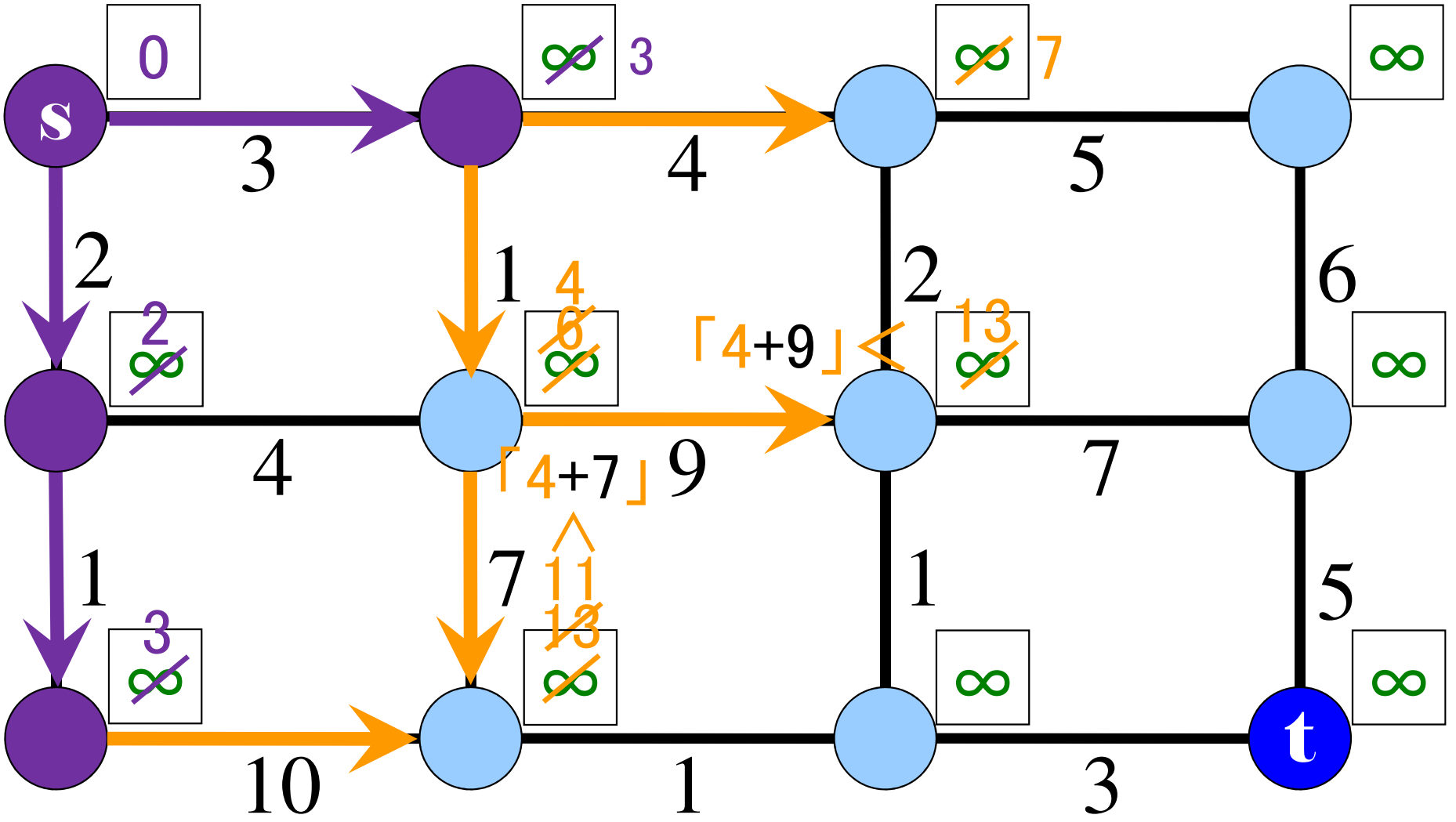
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

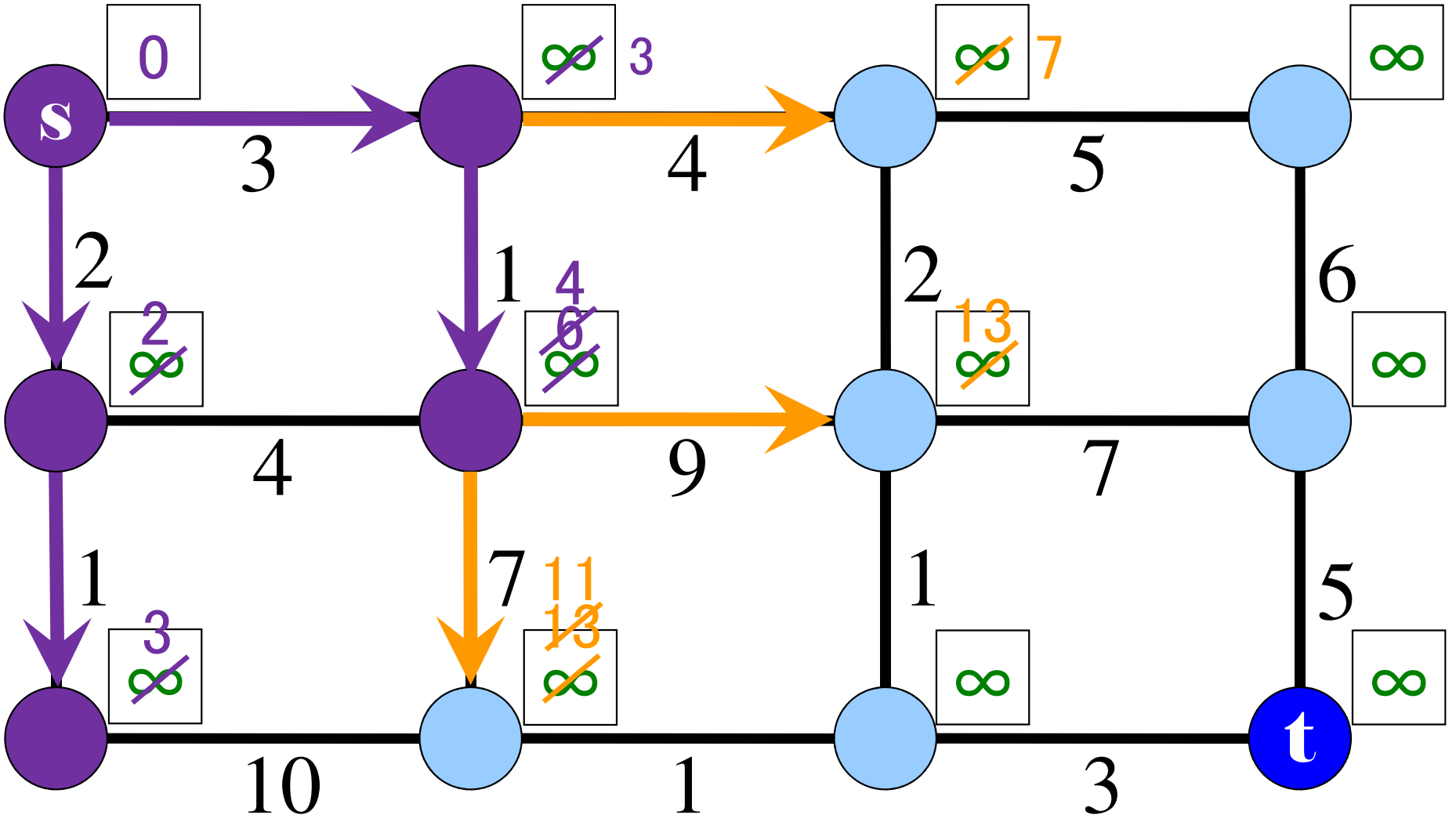


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

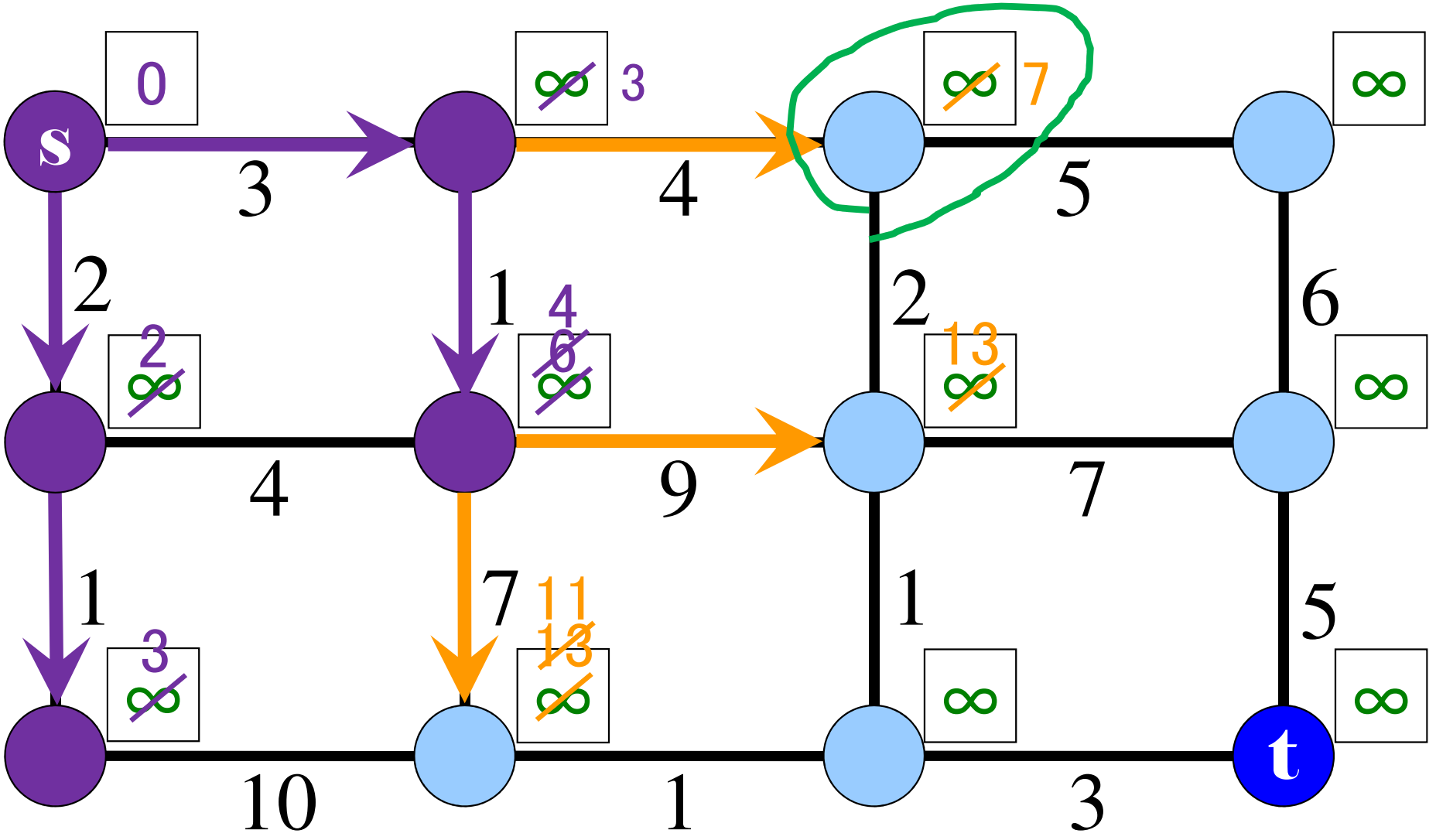
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



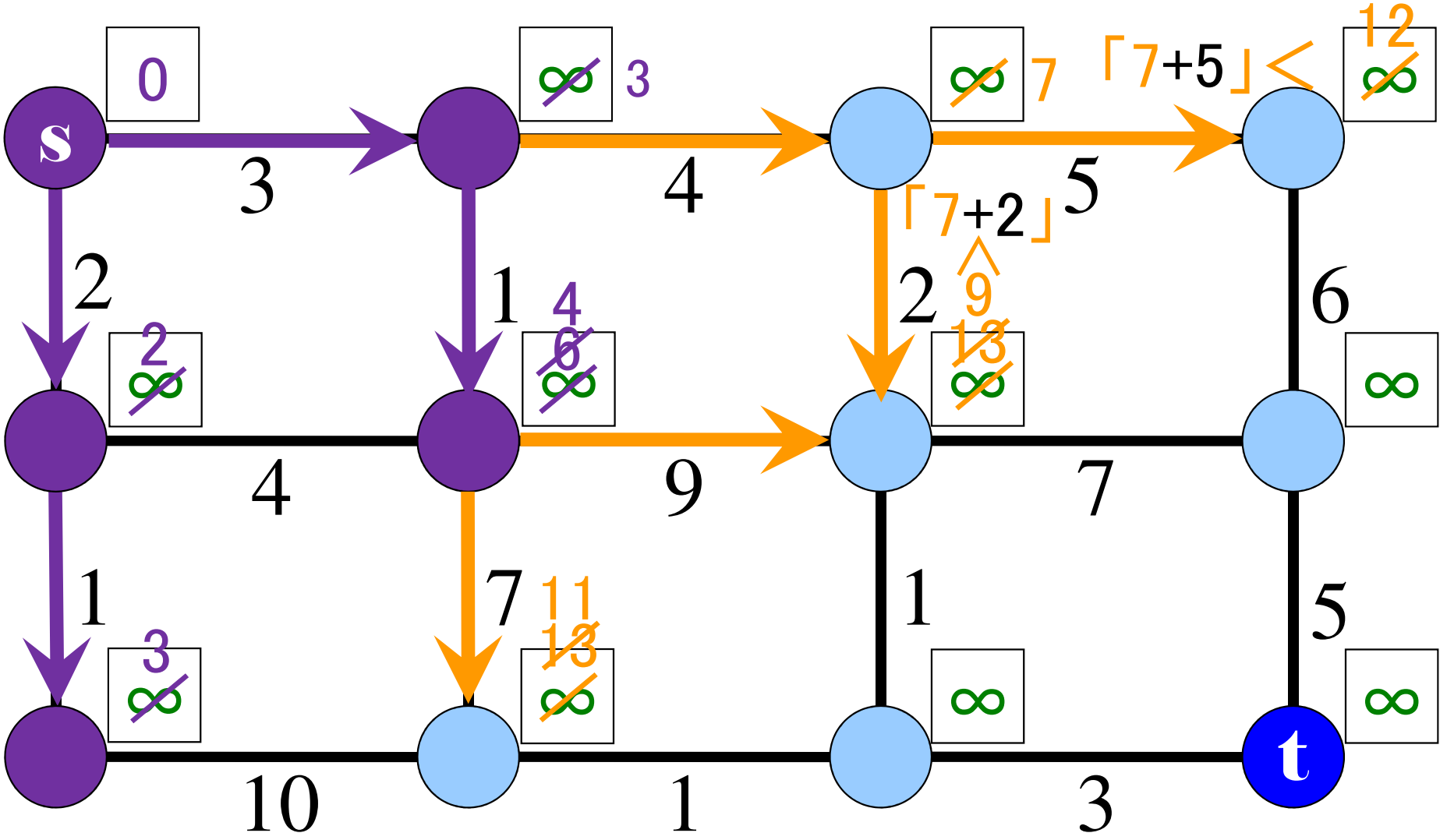
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

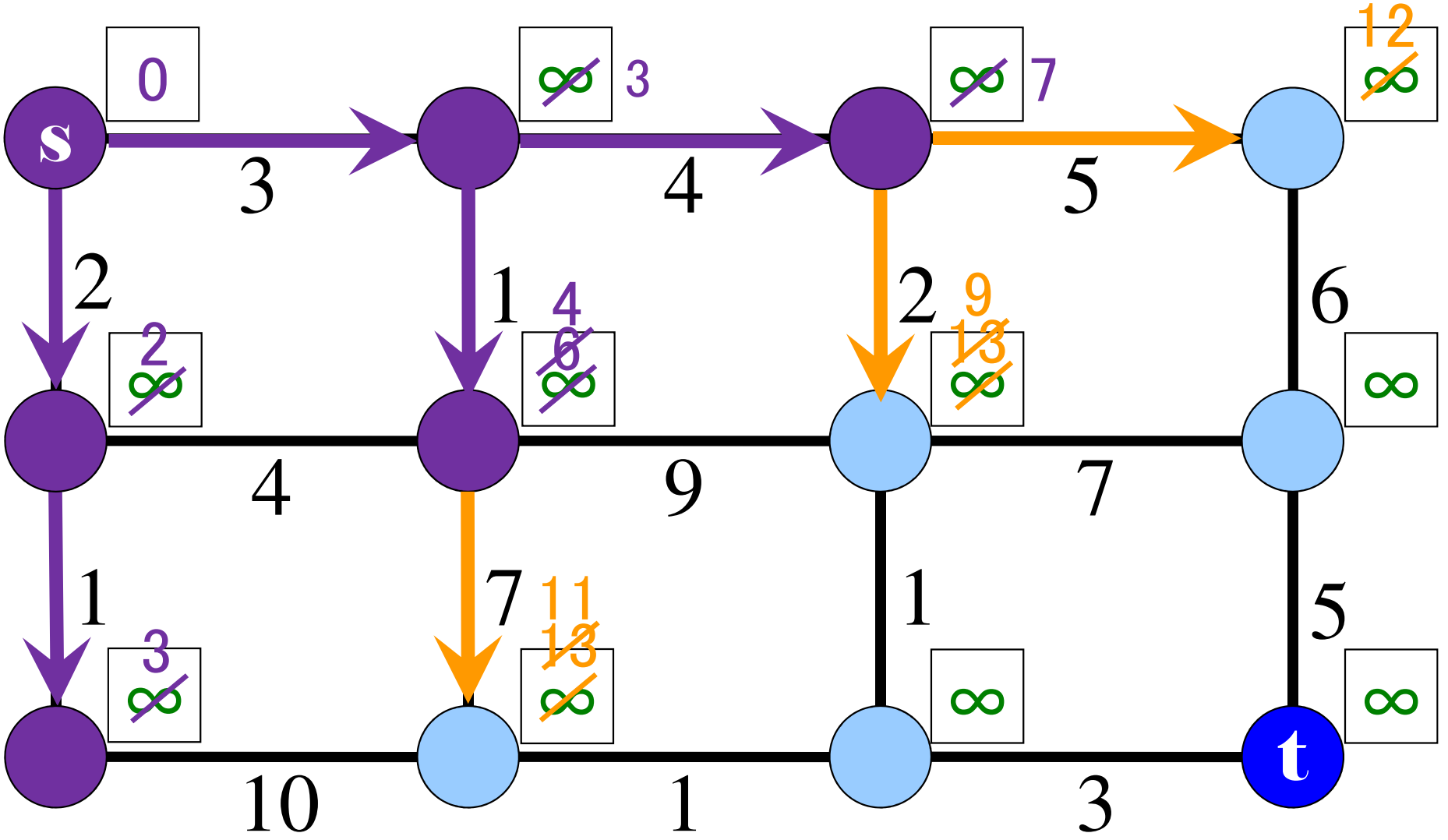


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

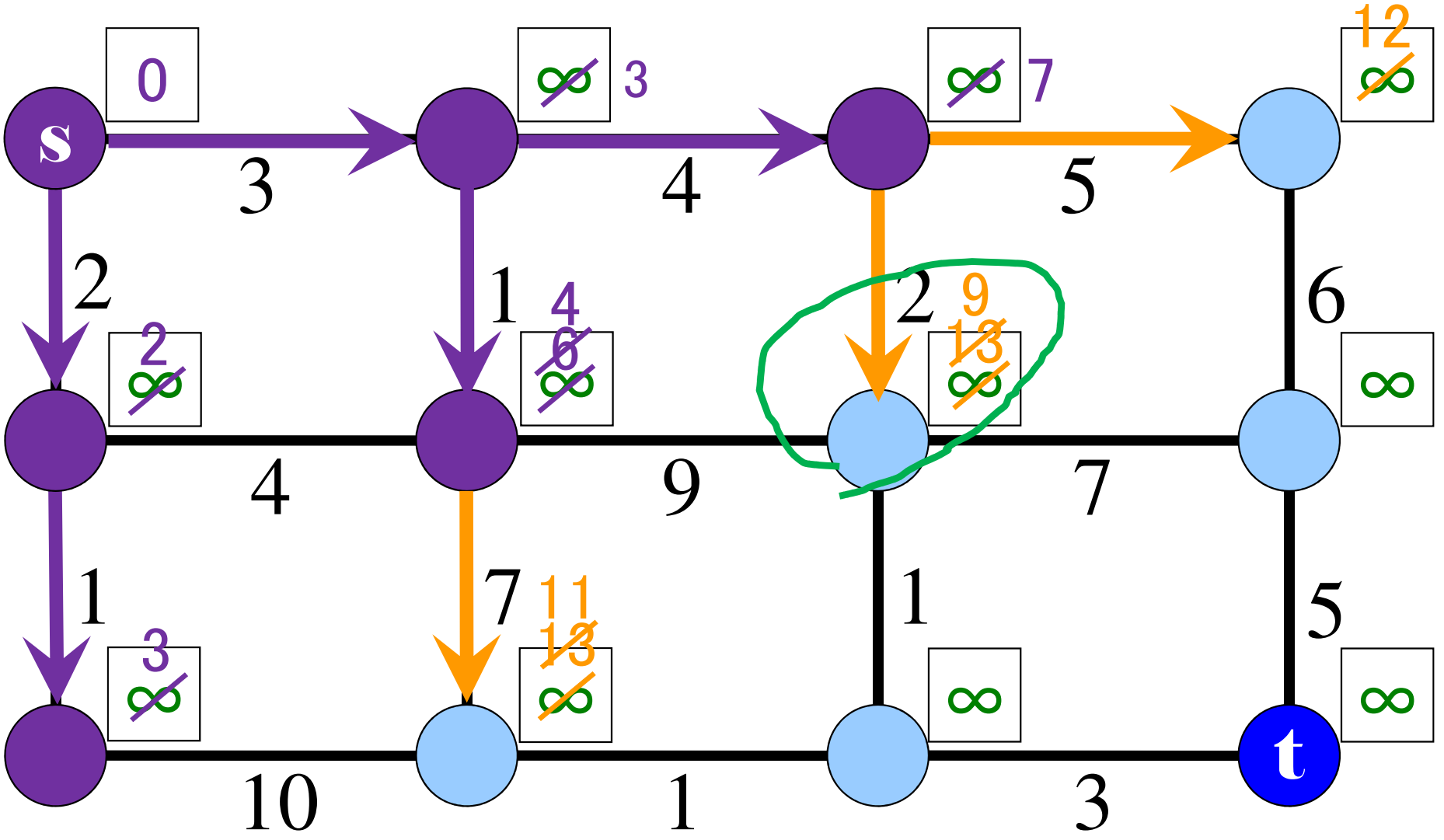
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



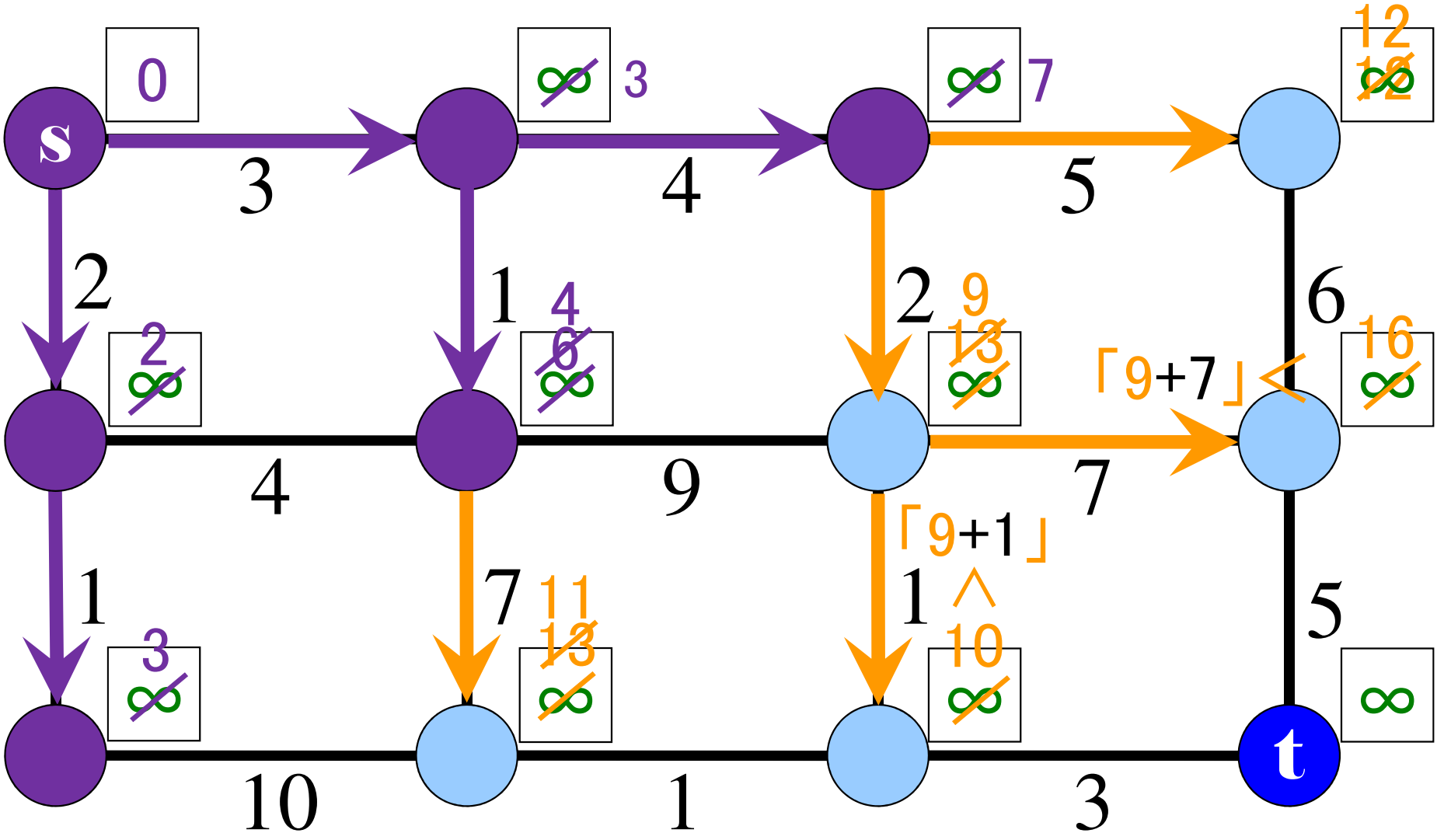
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

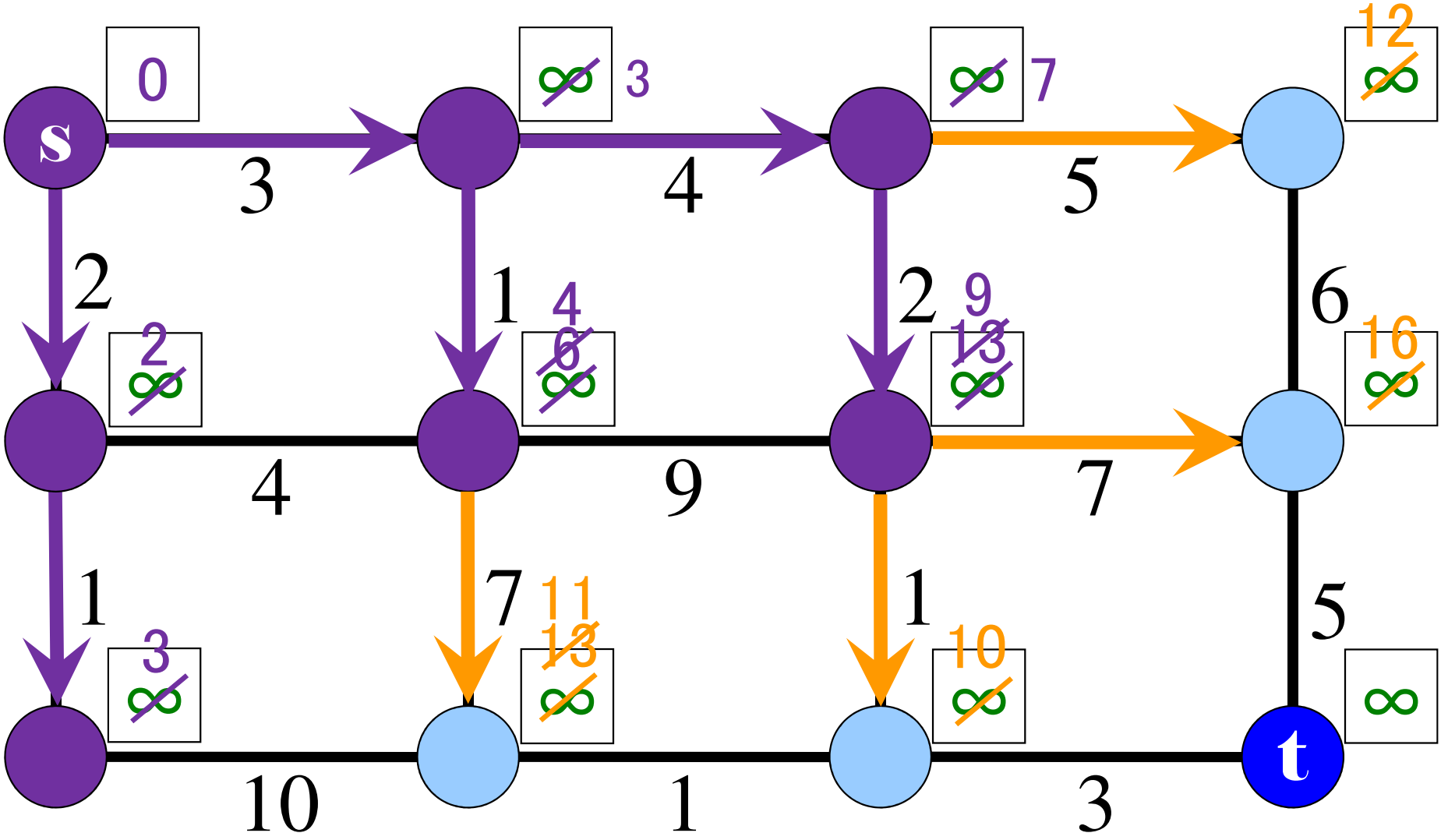


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

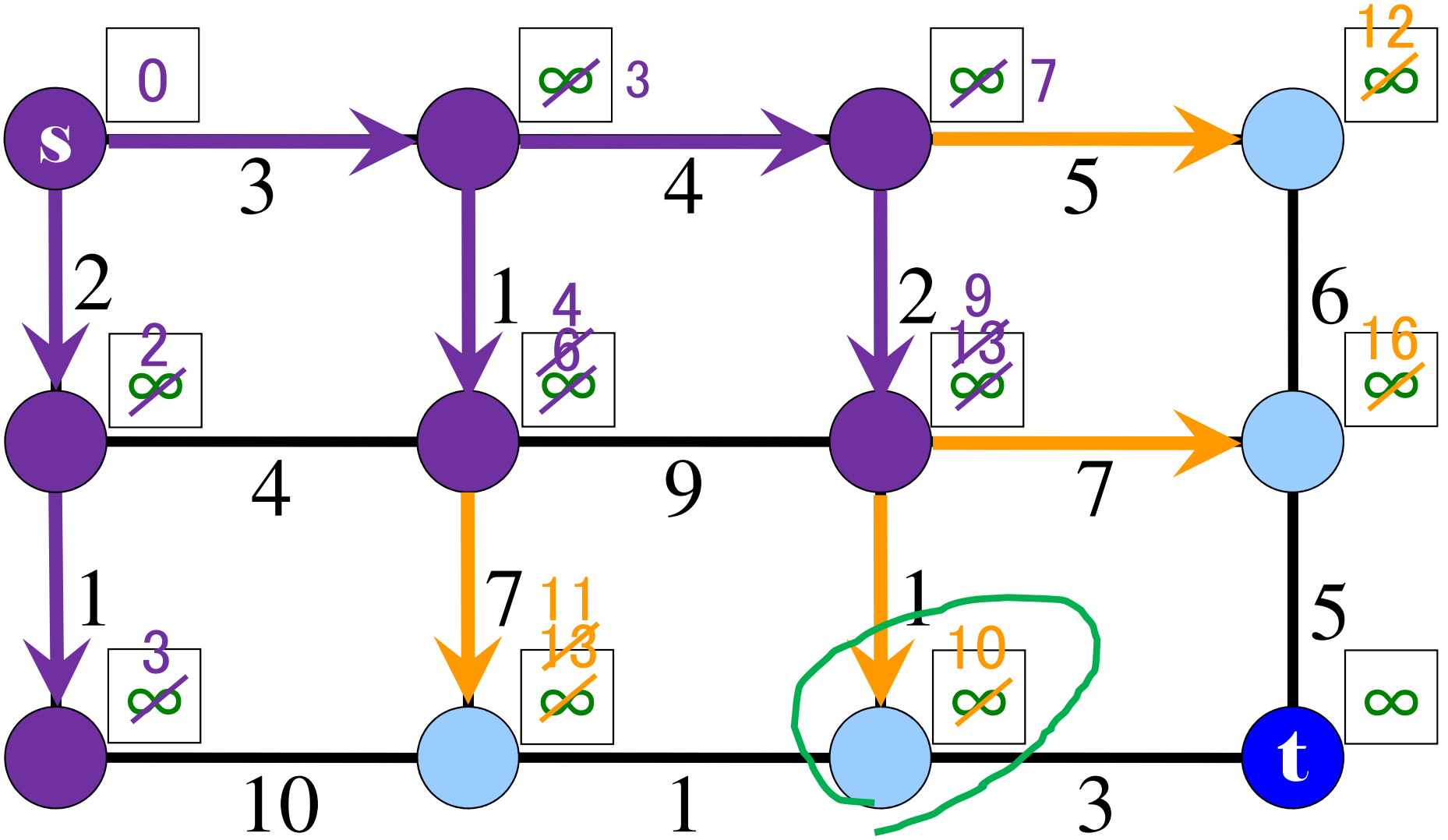
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



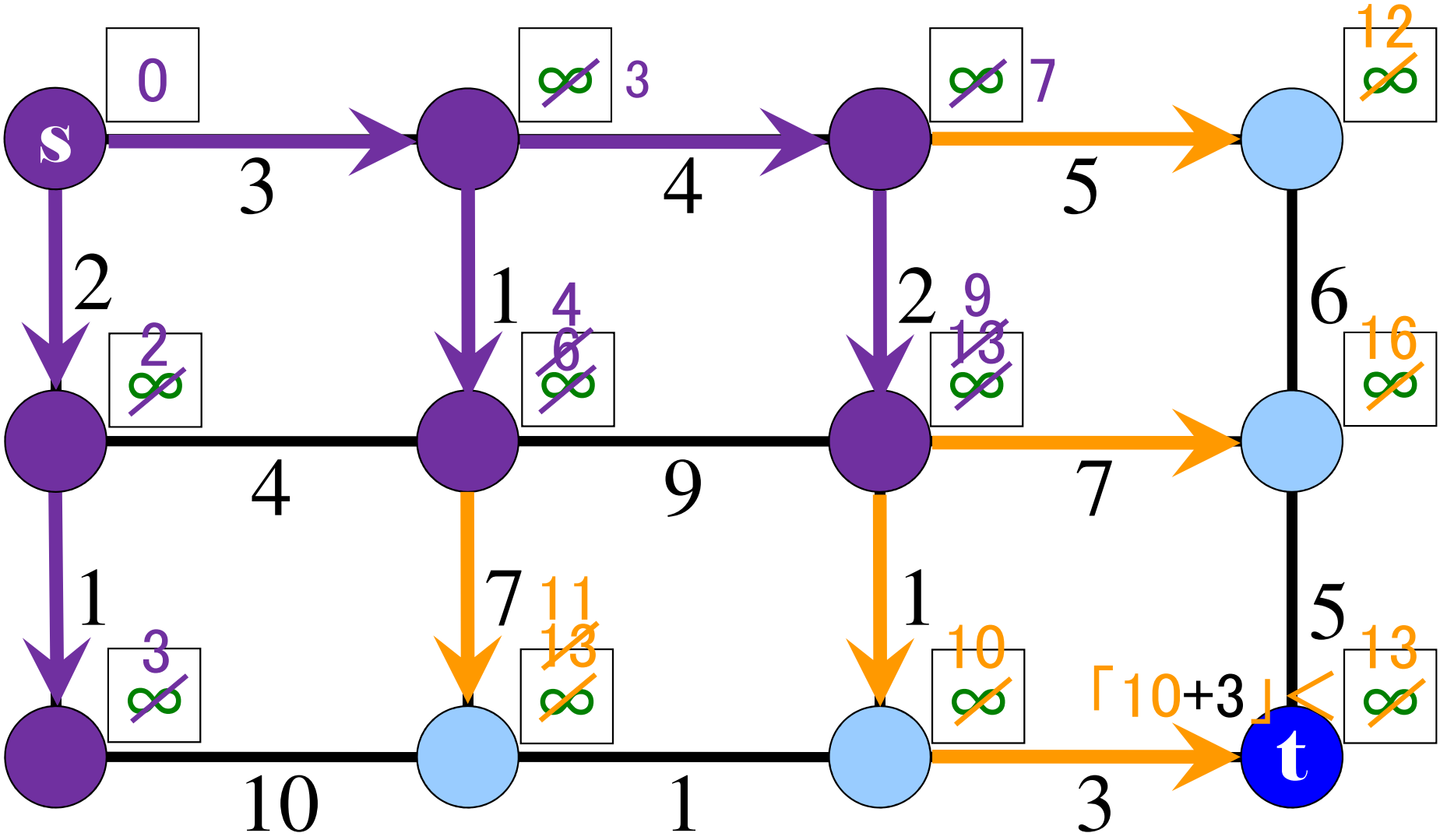
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

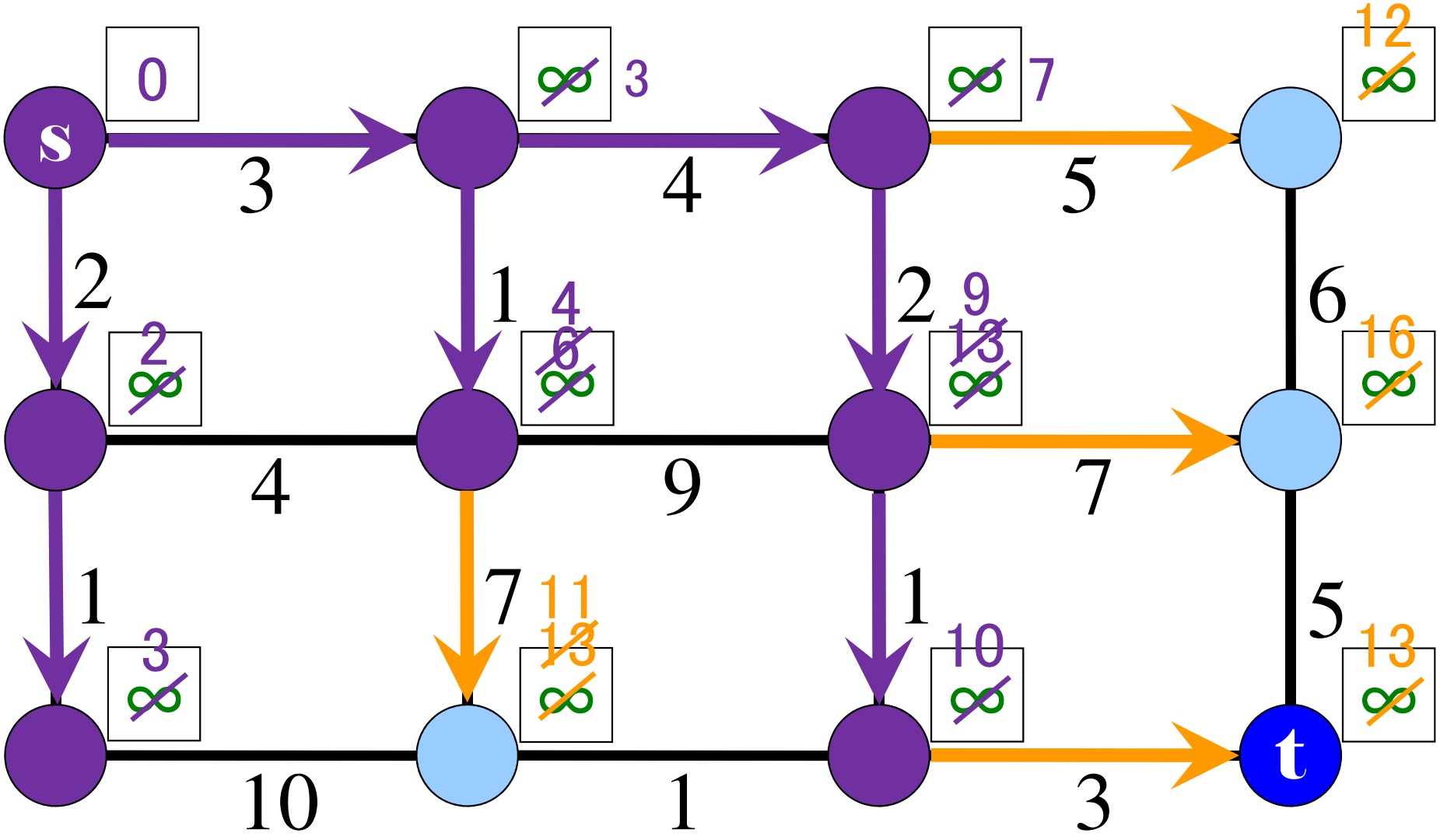


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

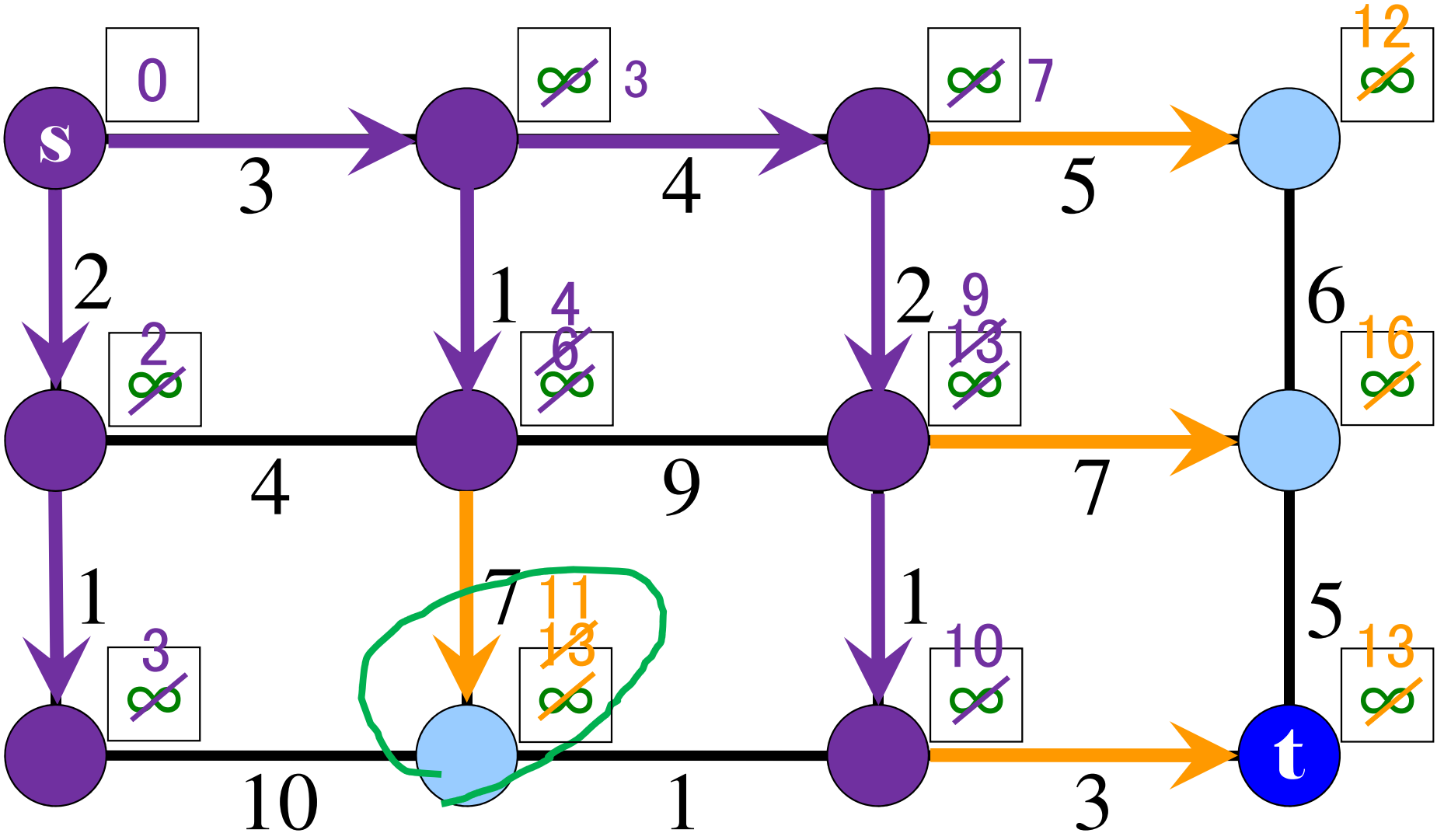
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



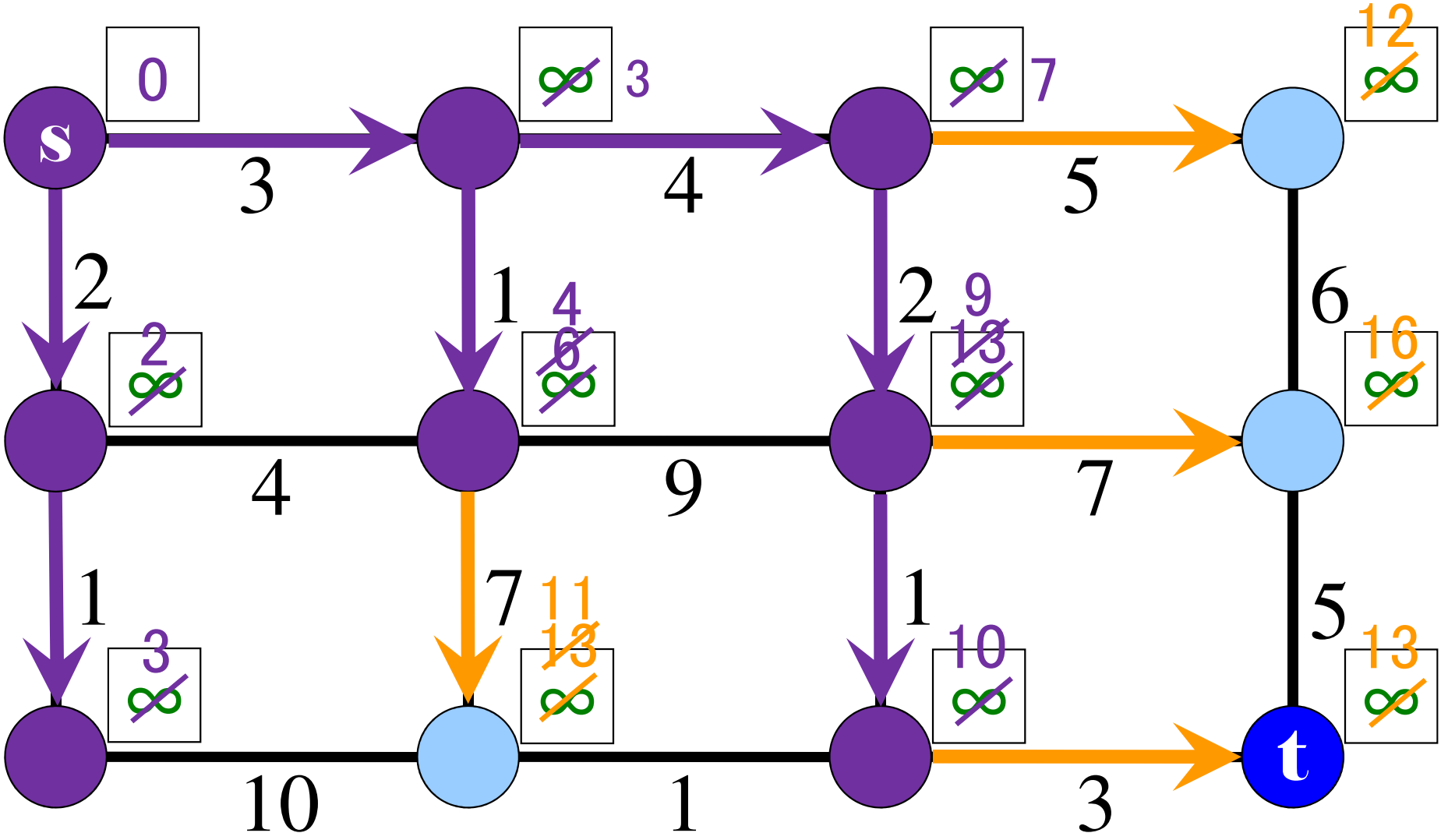
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

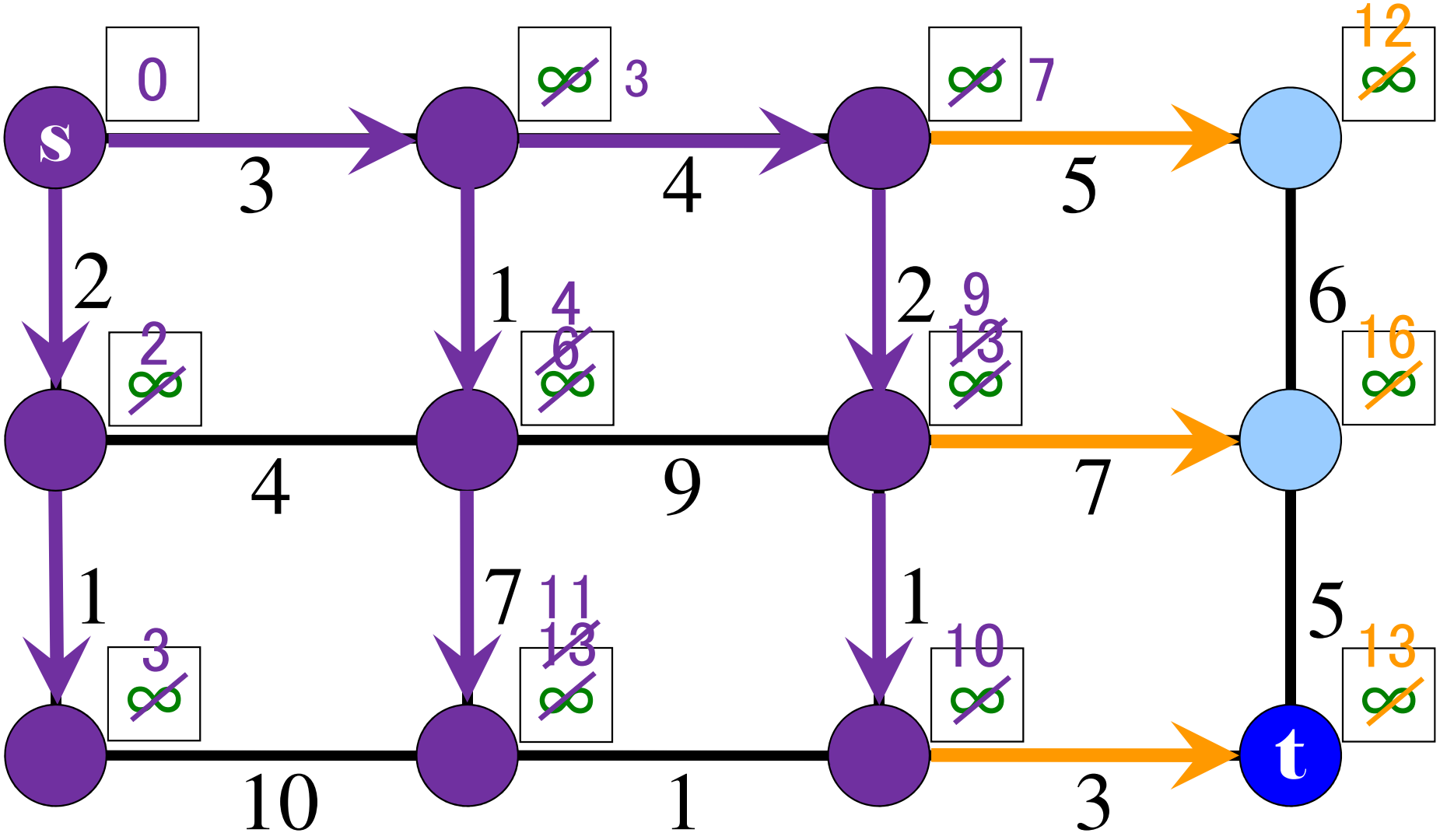


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

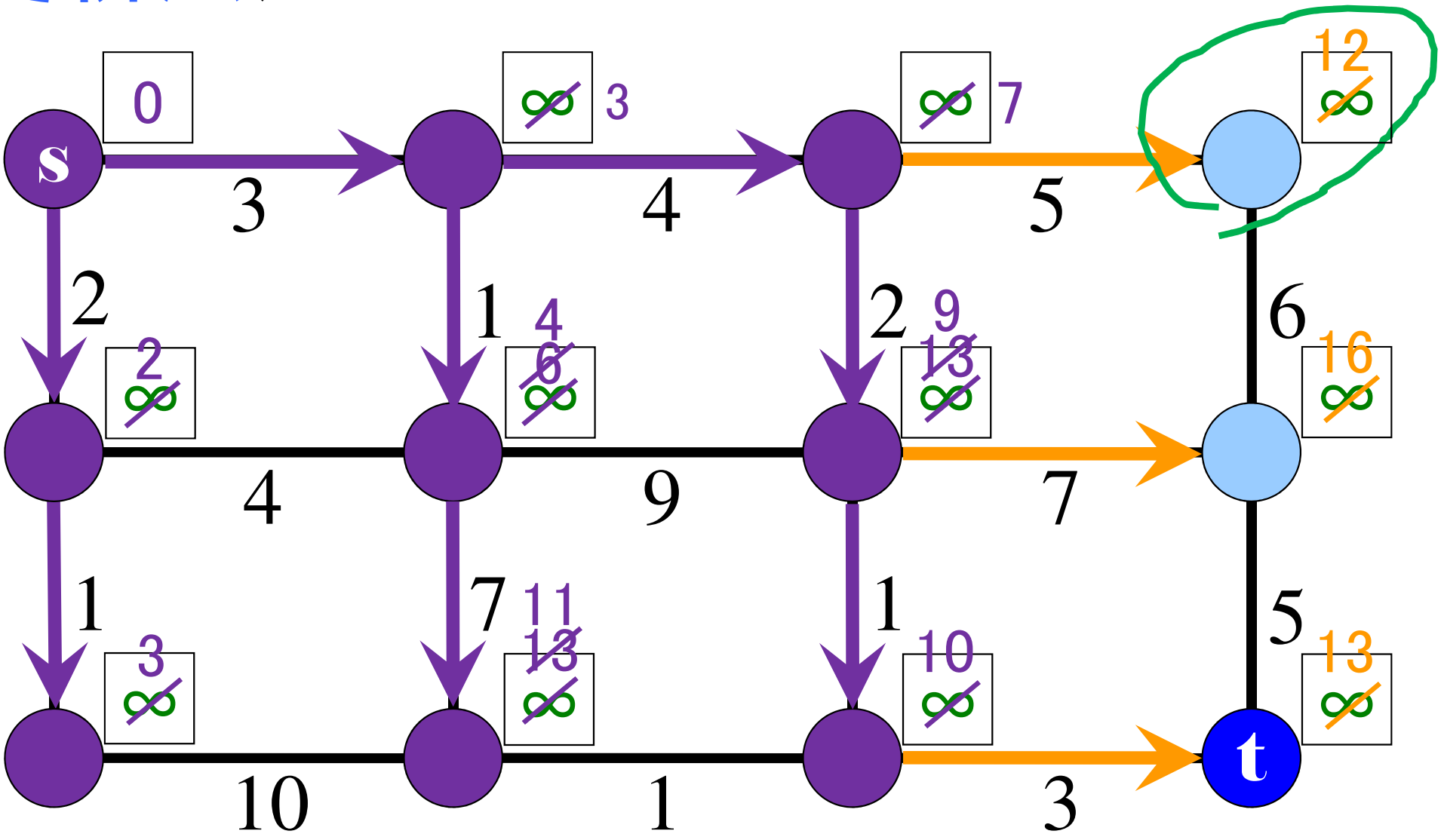
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



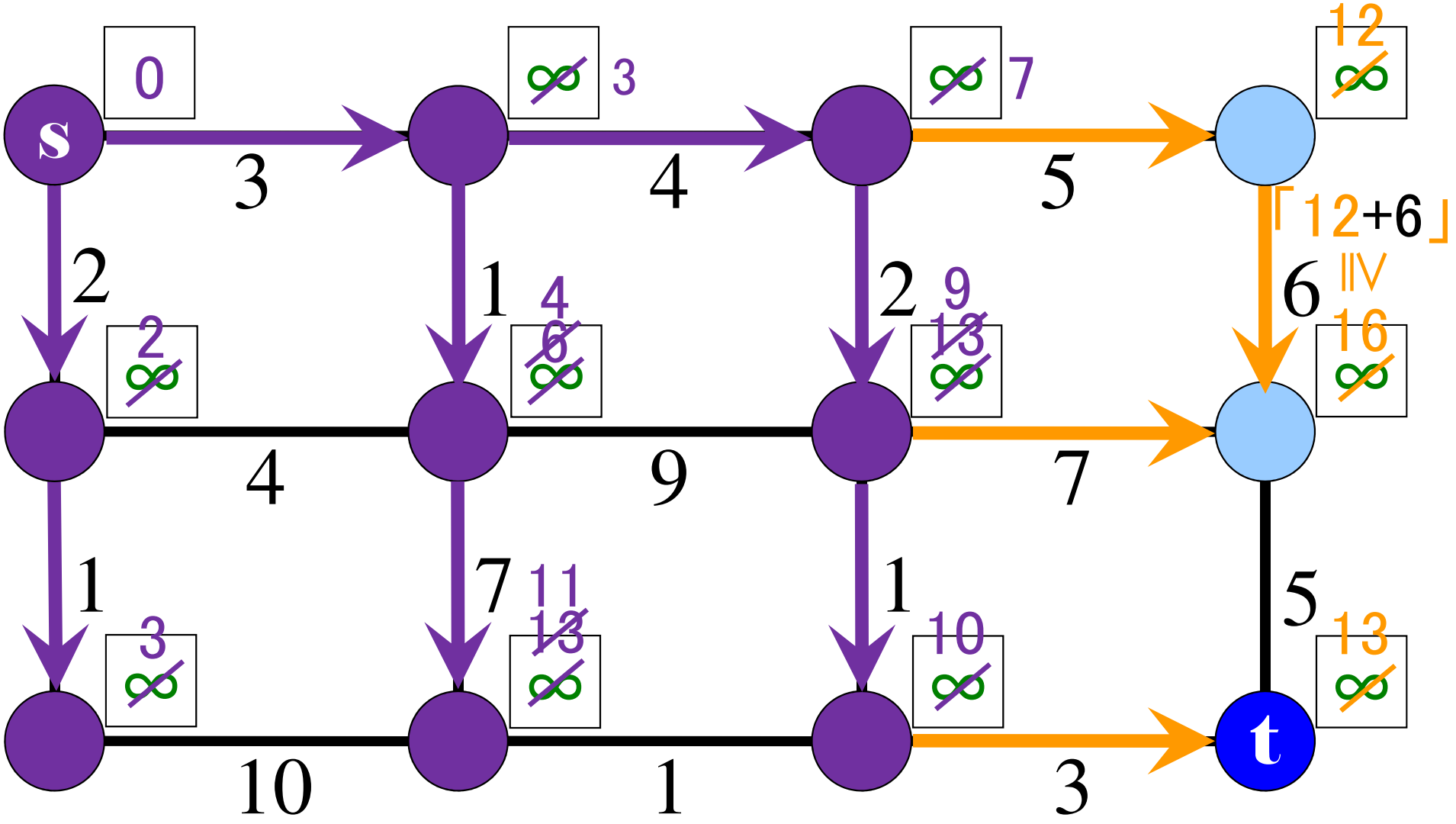
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

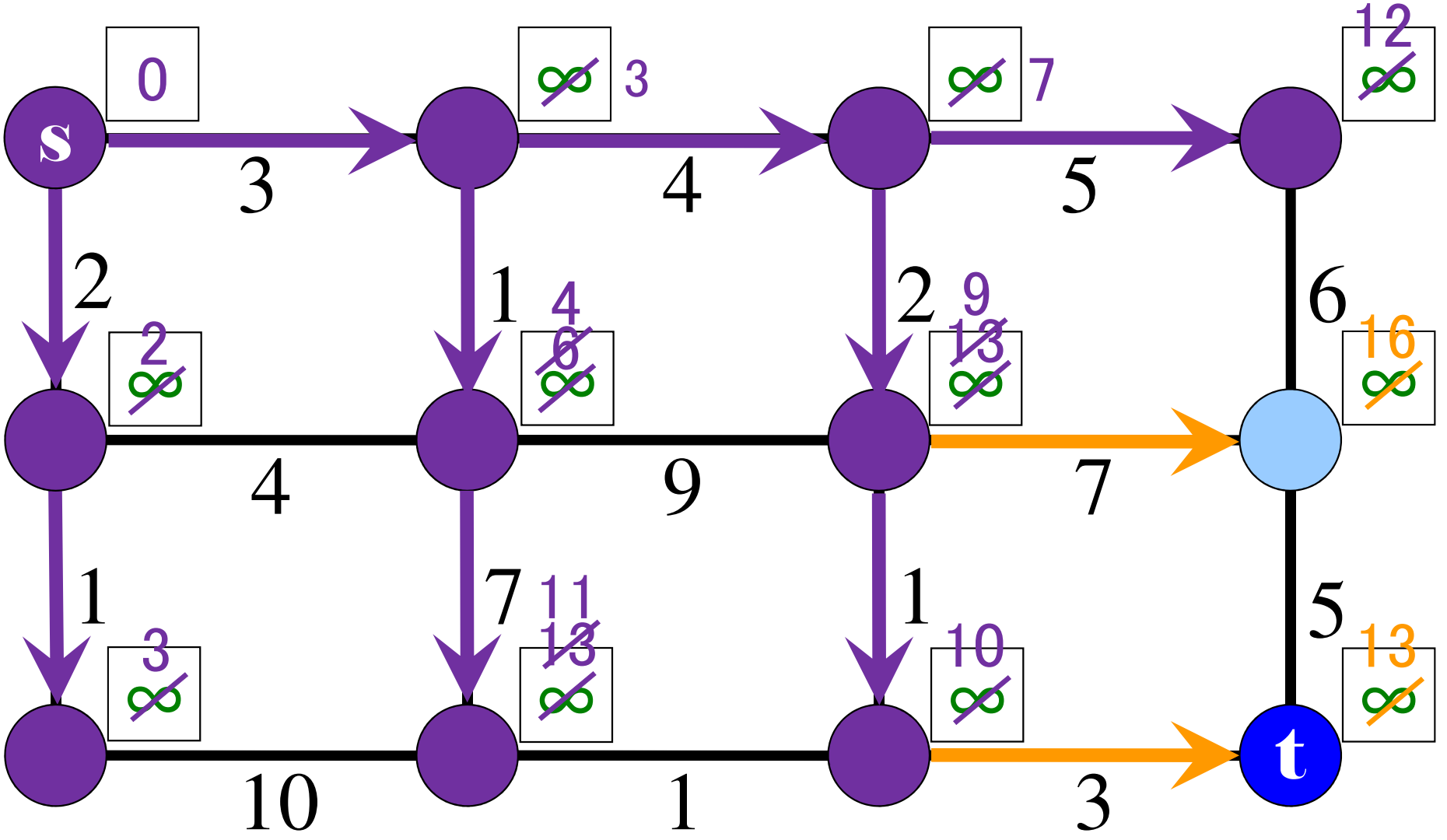


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

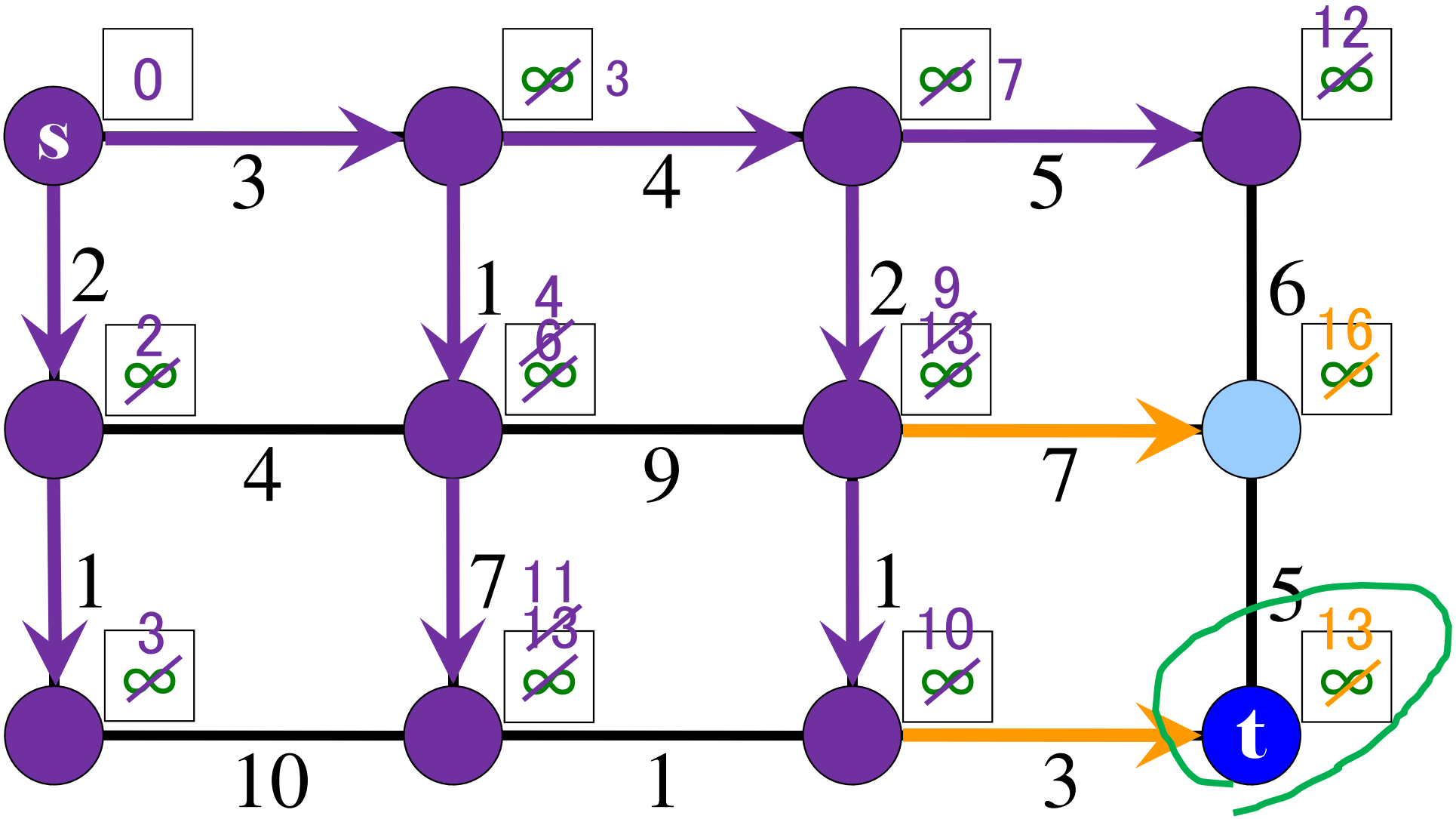
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



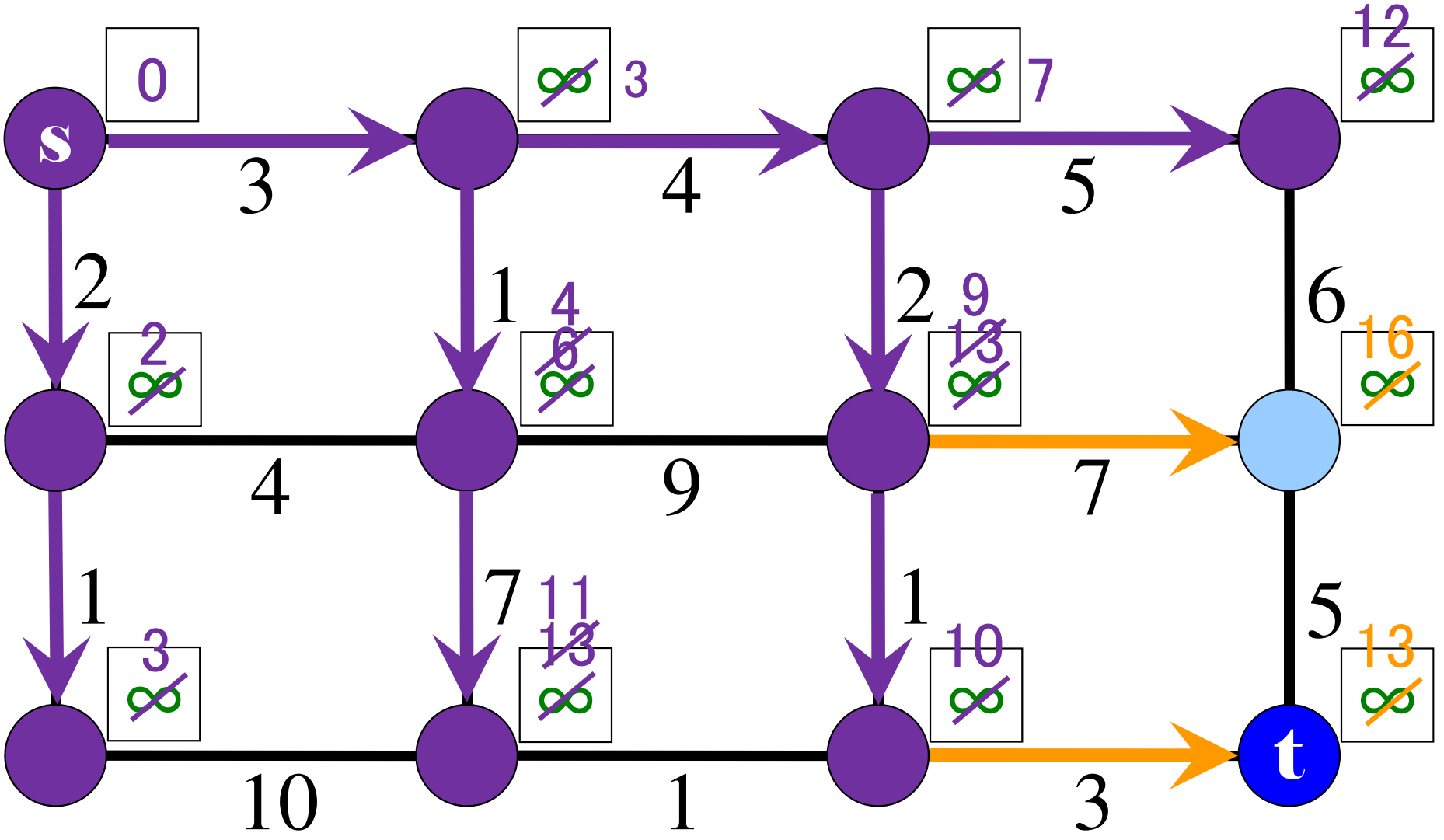
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

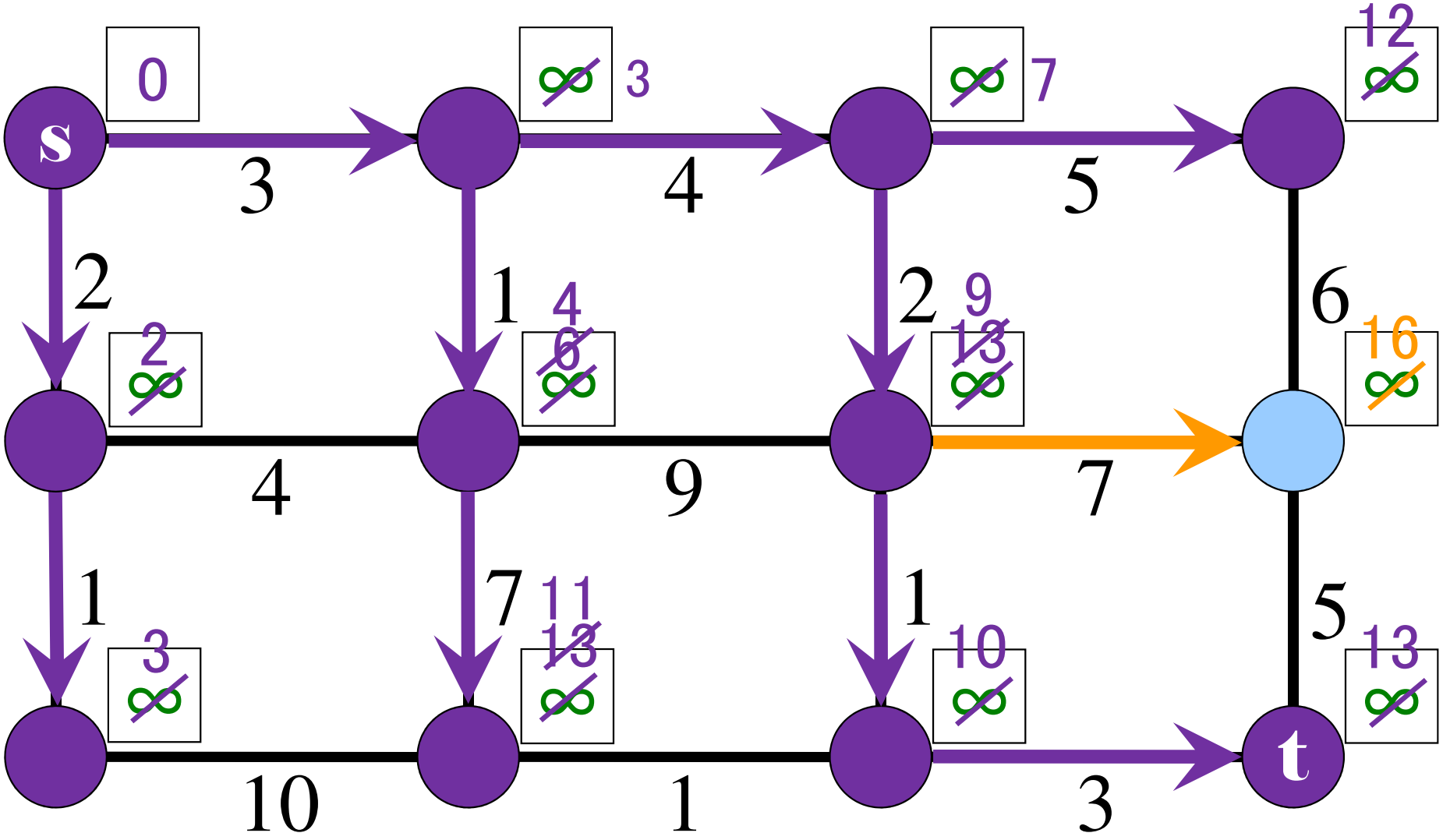


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

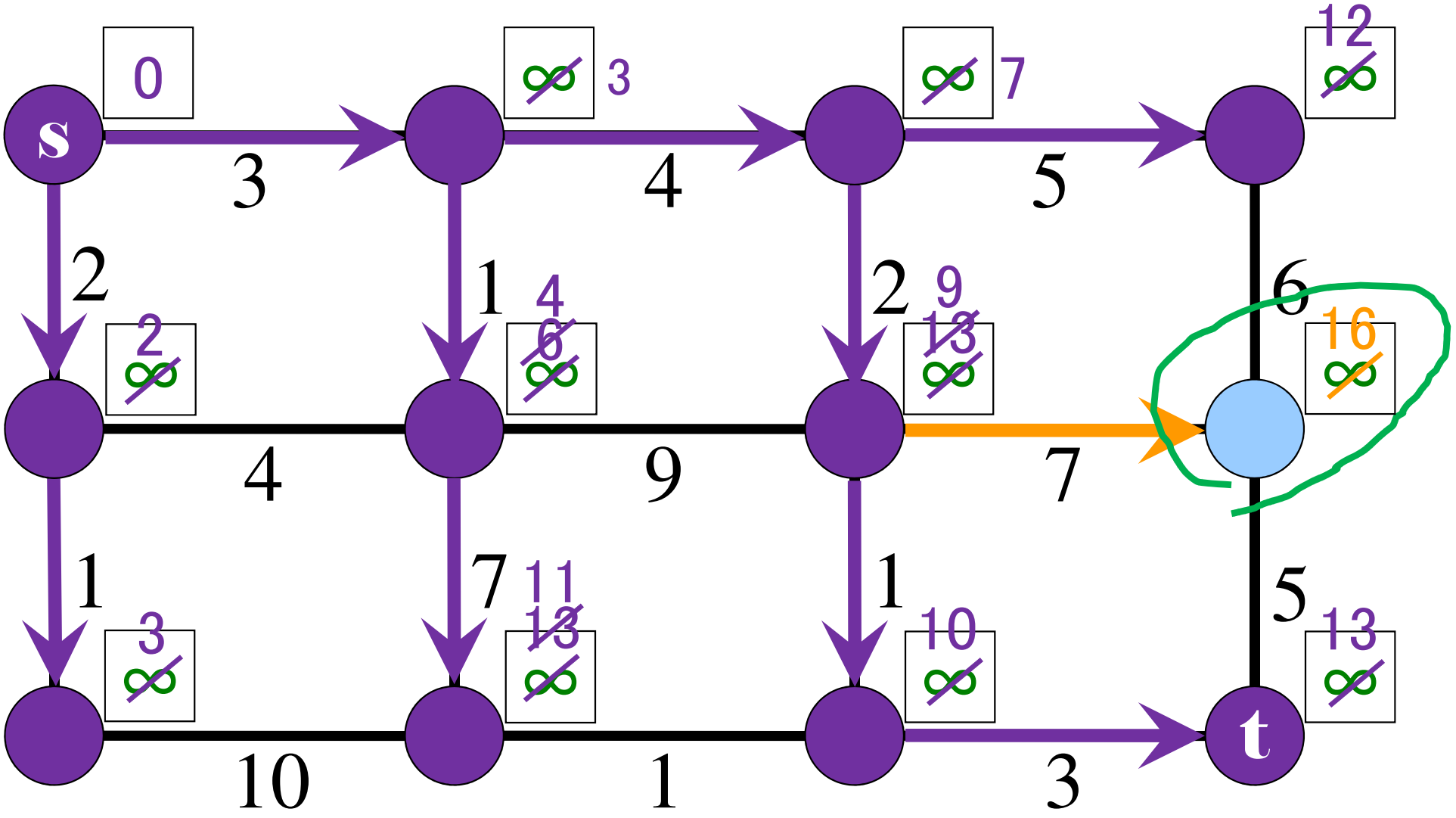
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



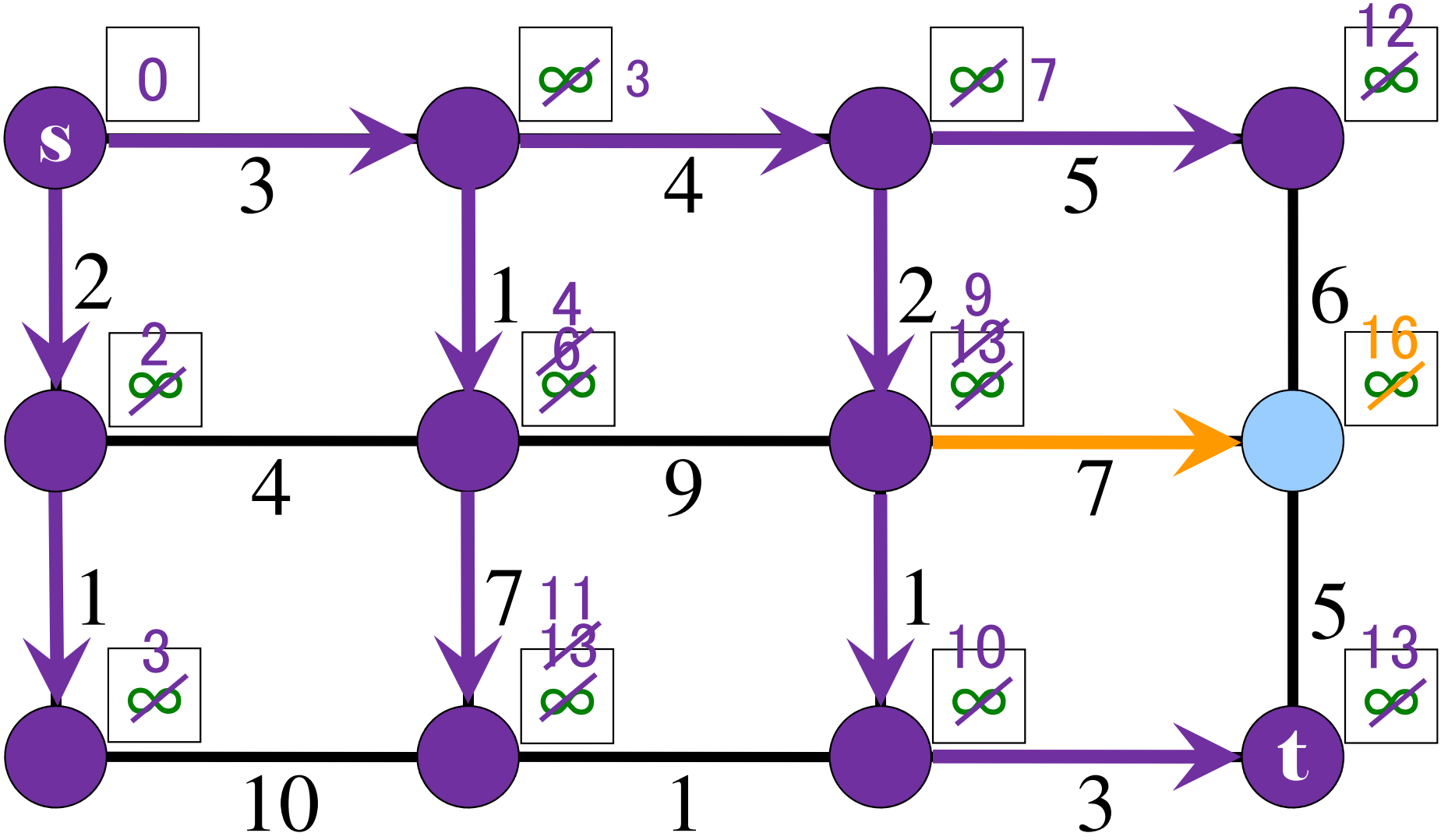
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

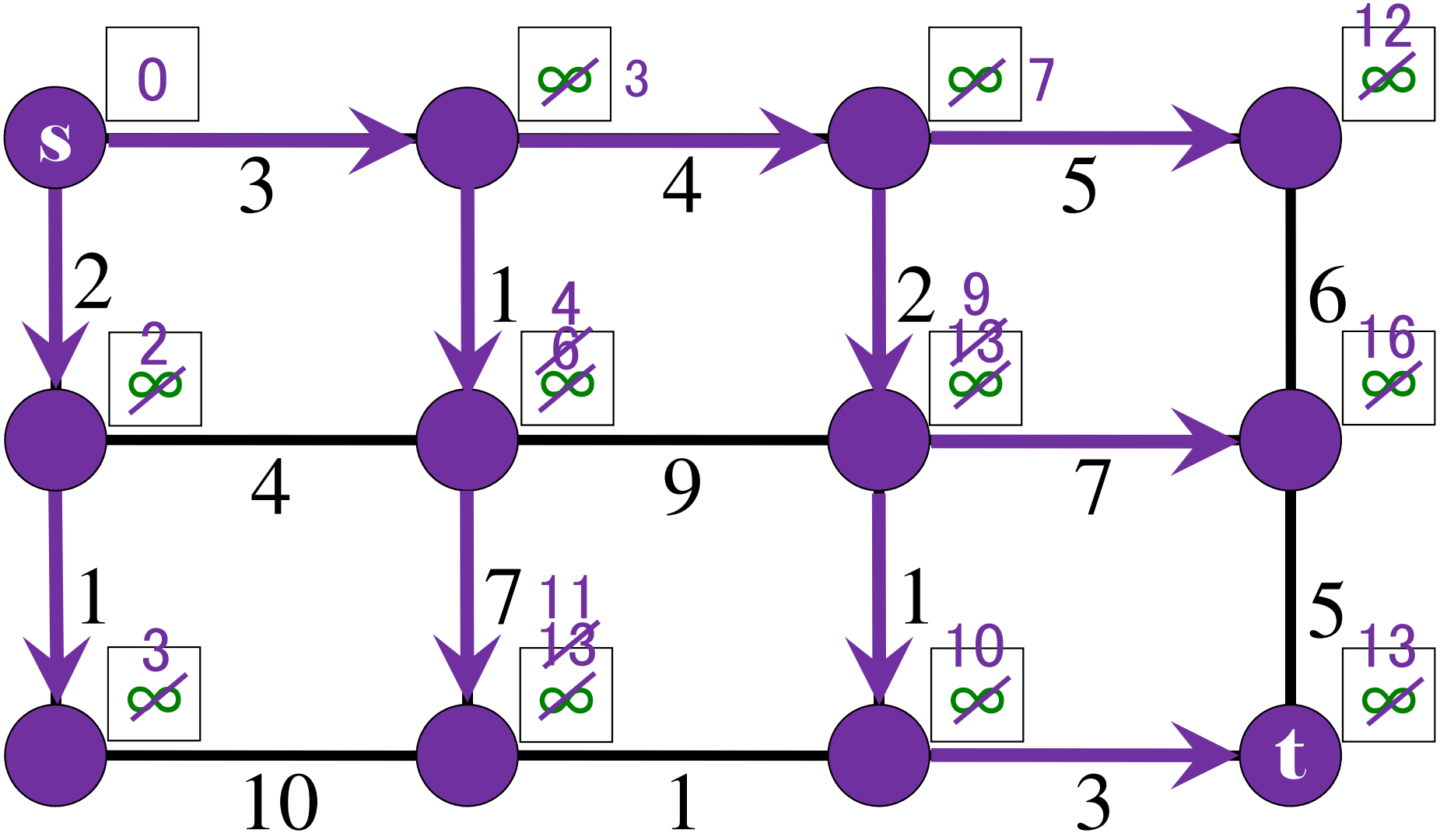


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

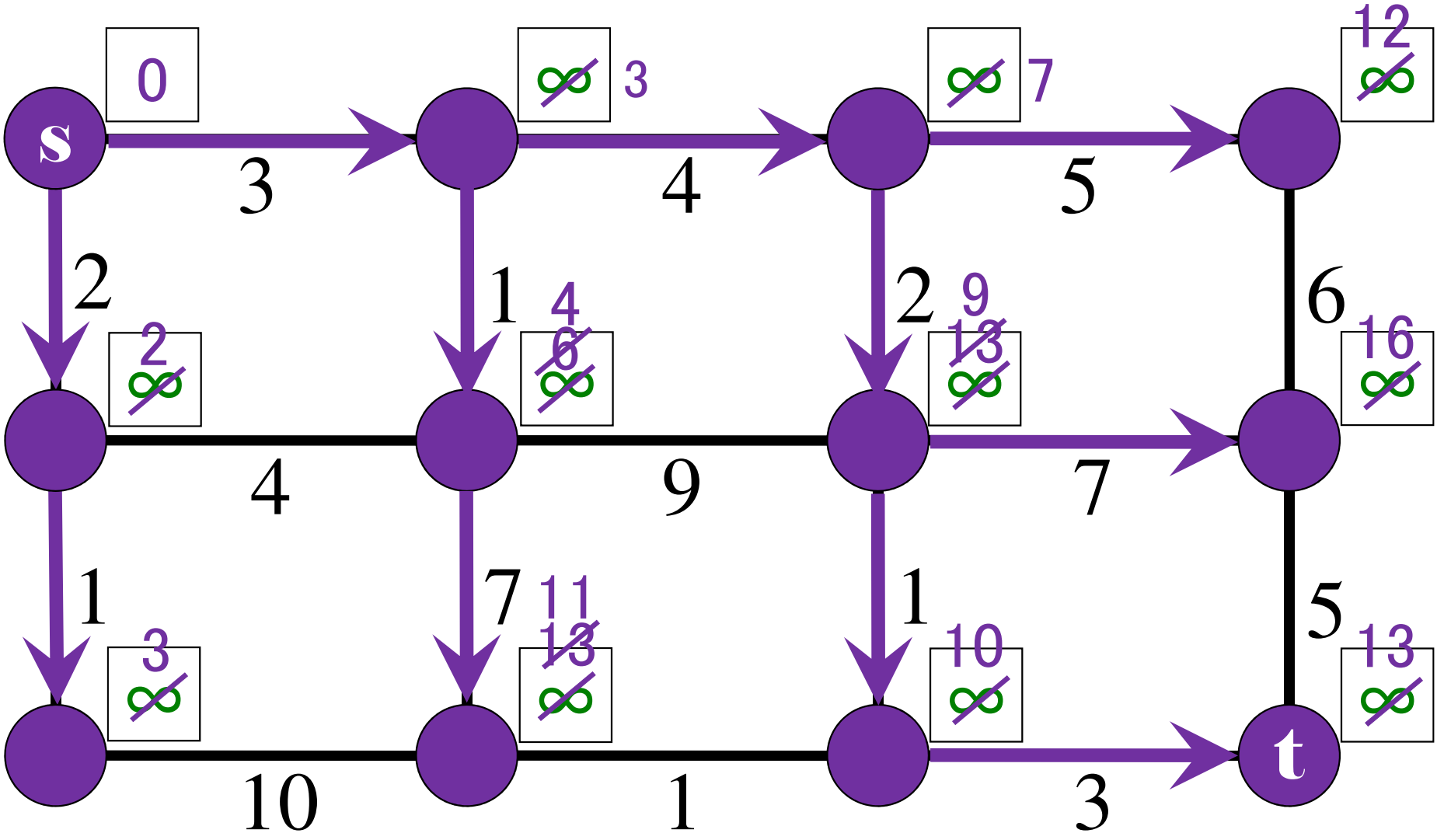
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



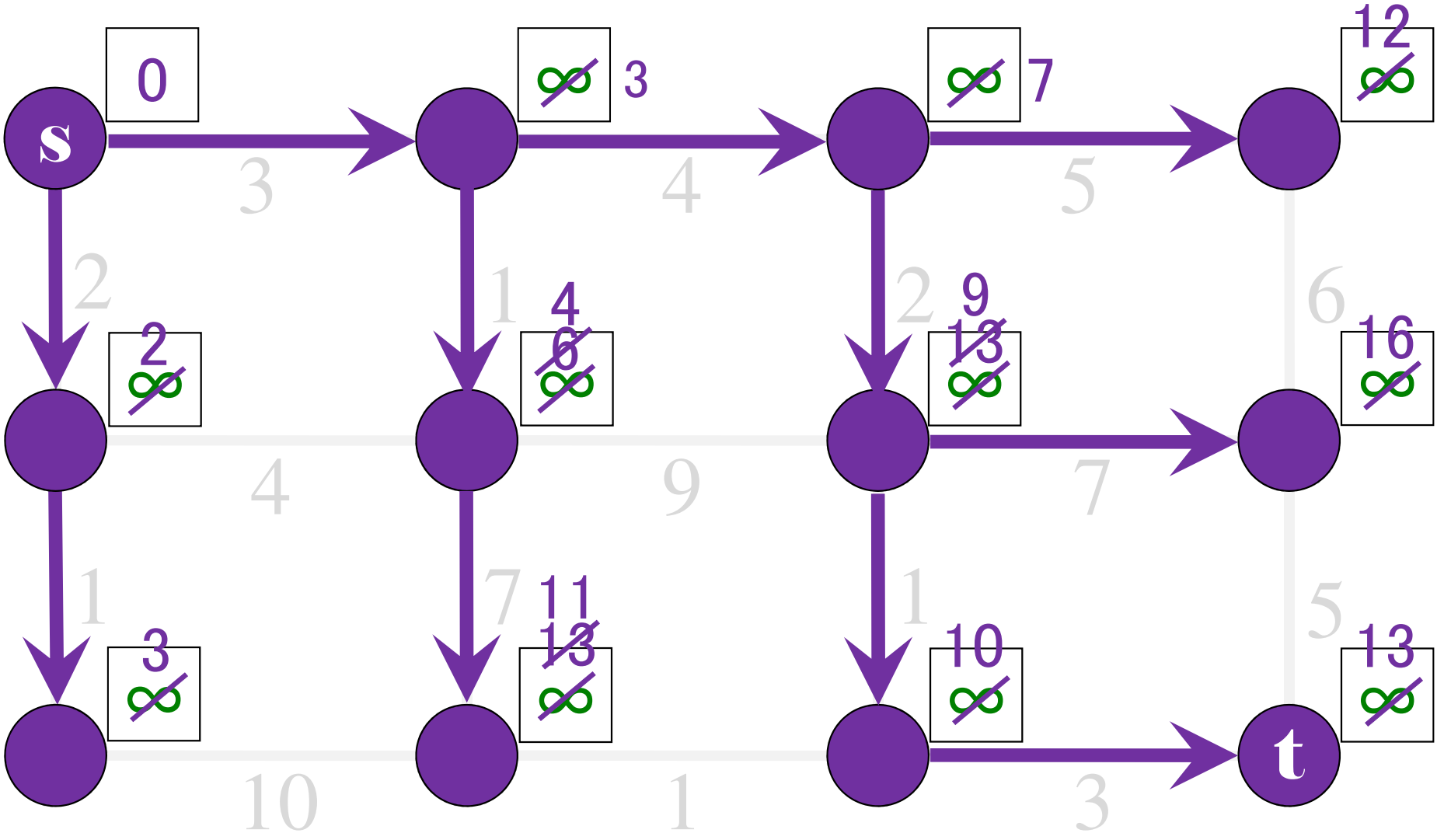
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



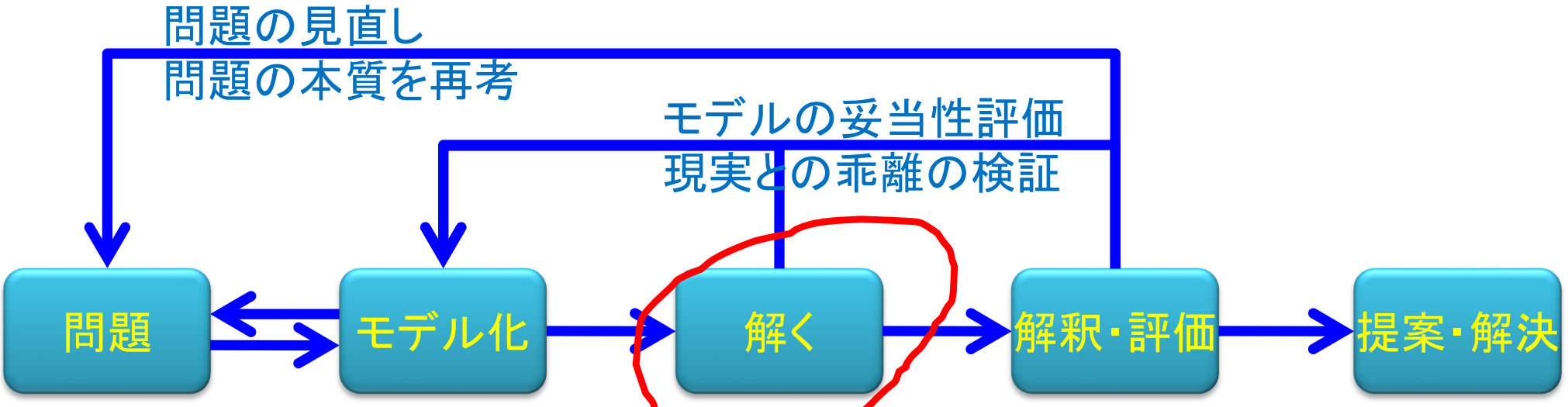
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

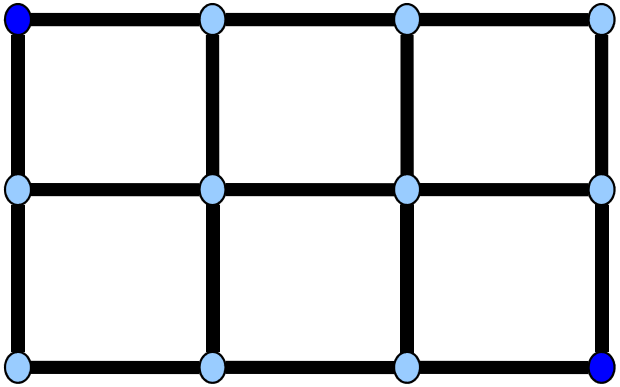
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

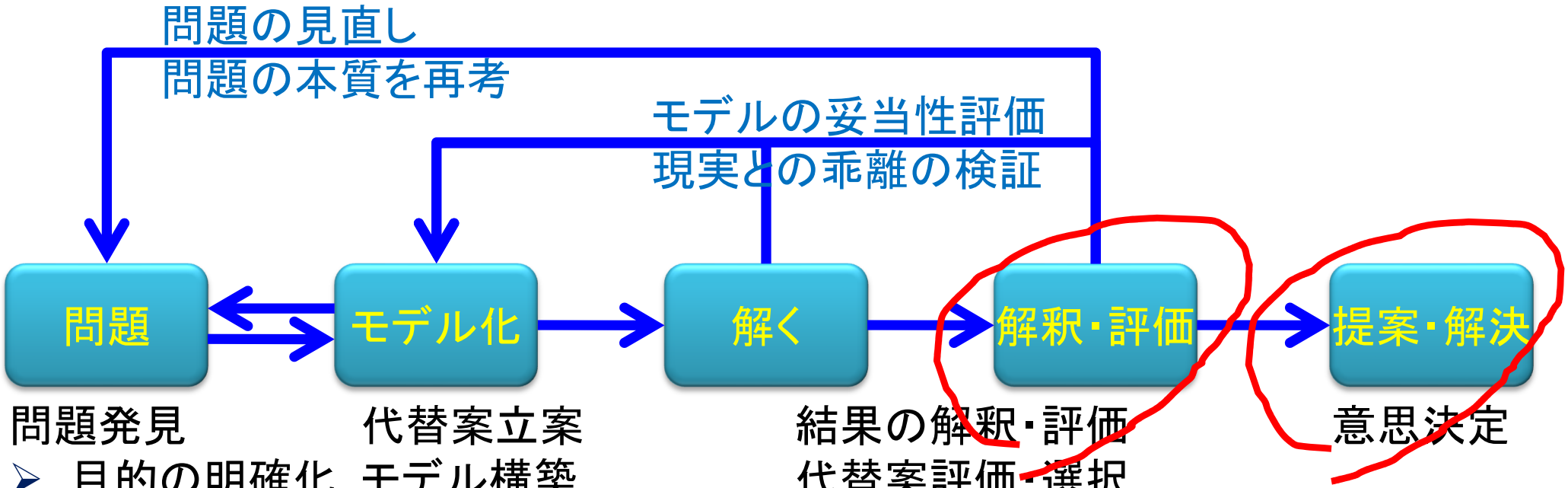
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

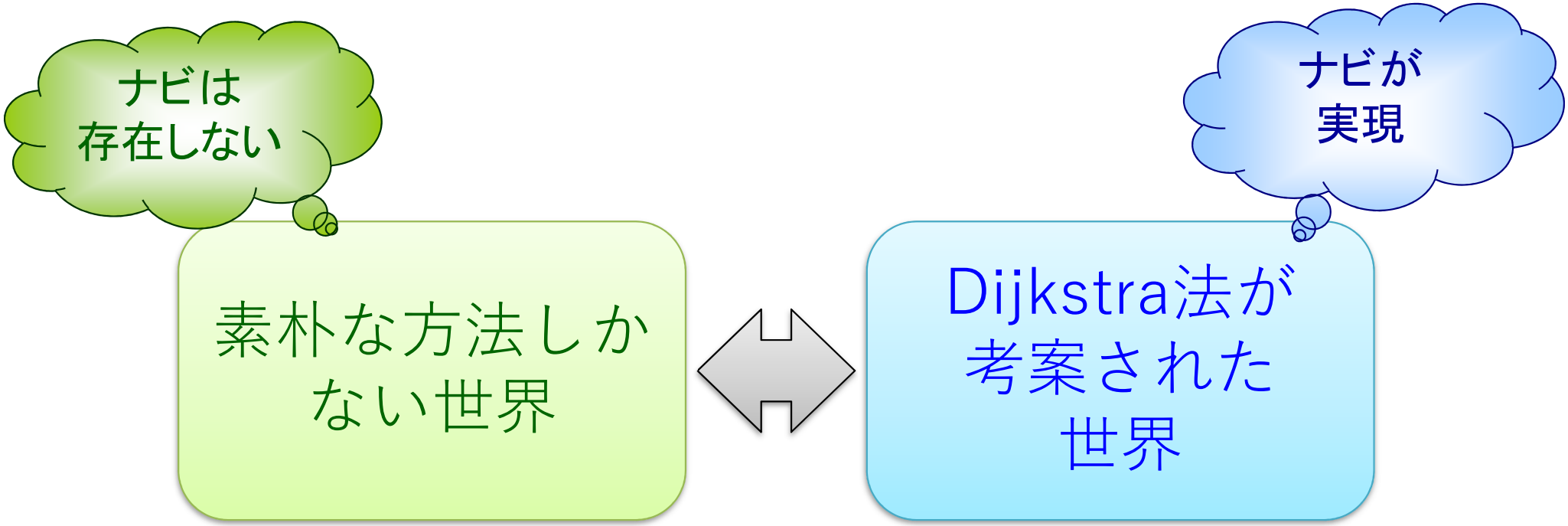
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

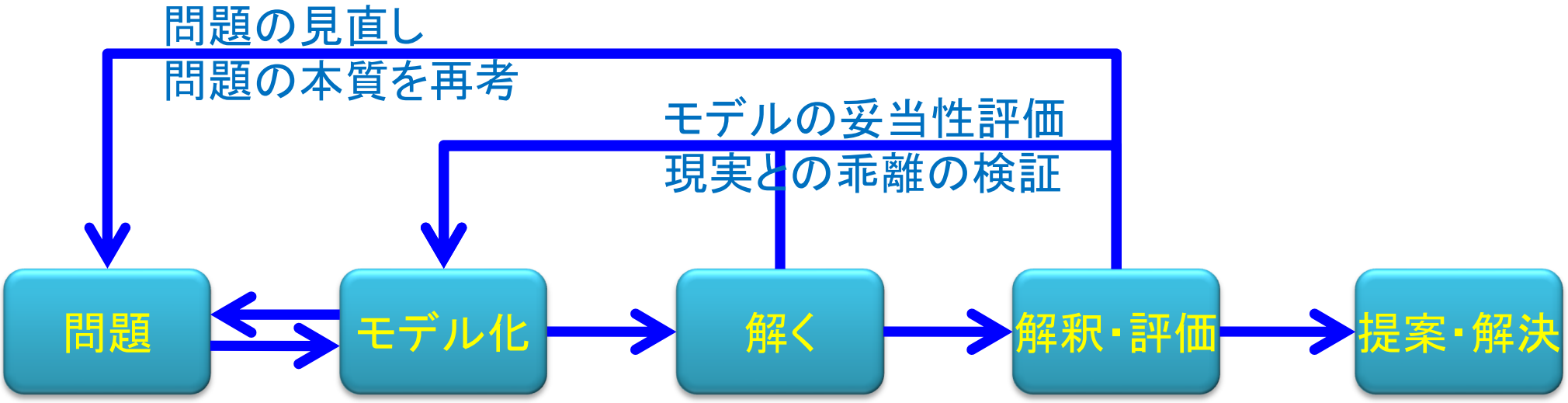
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



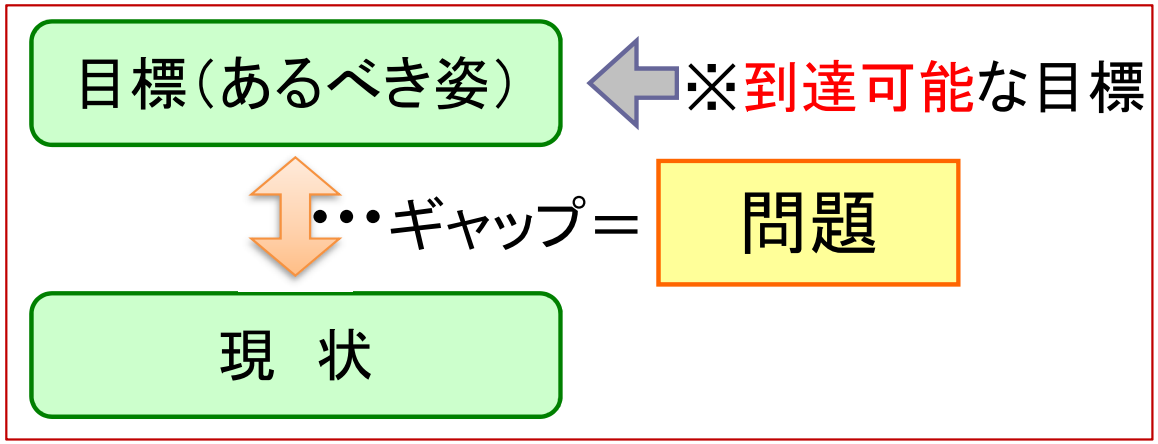
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

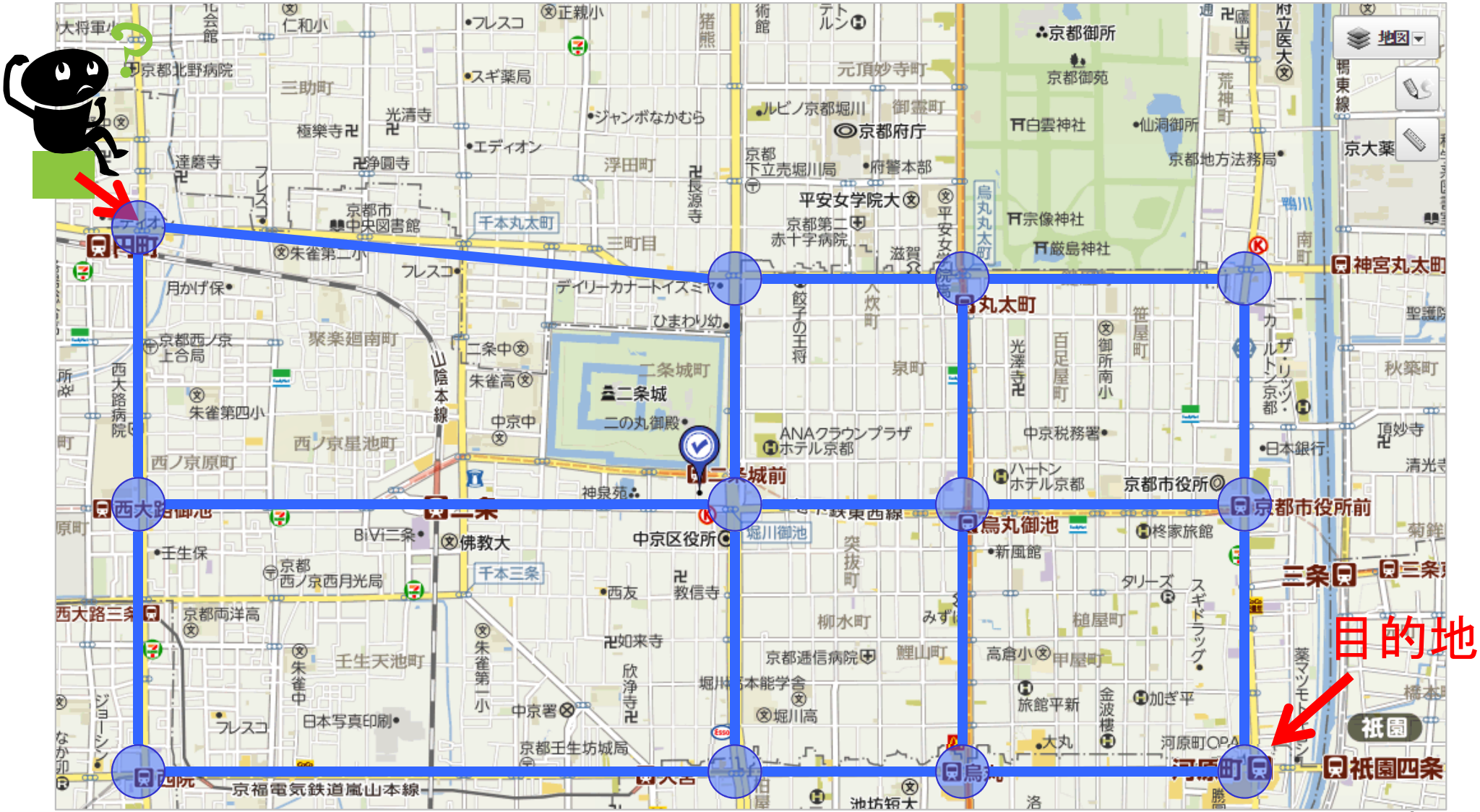
意思決定

問題の定義



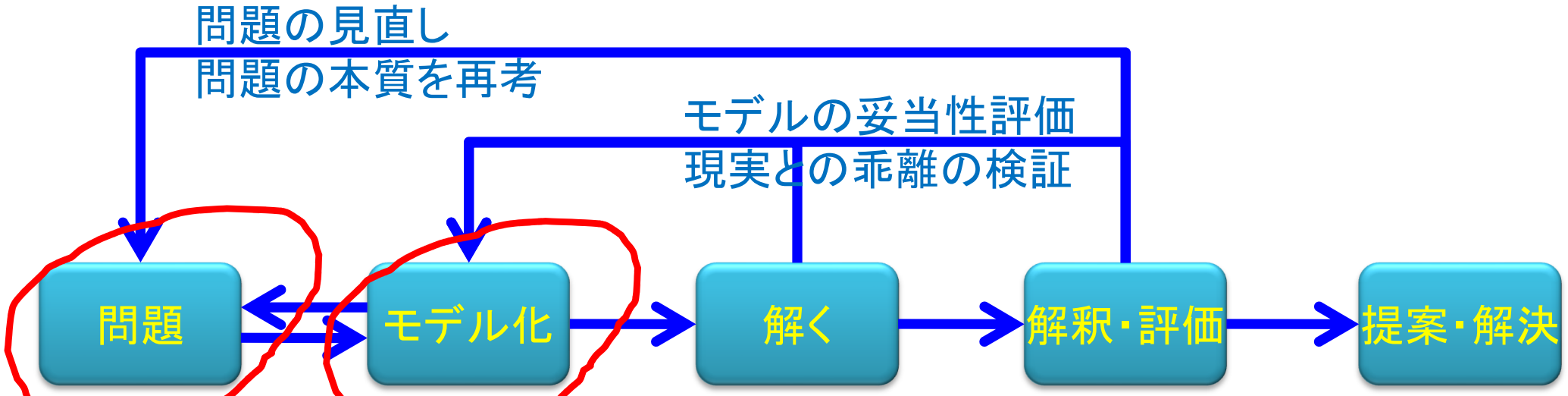
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

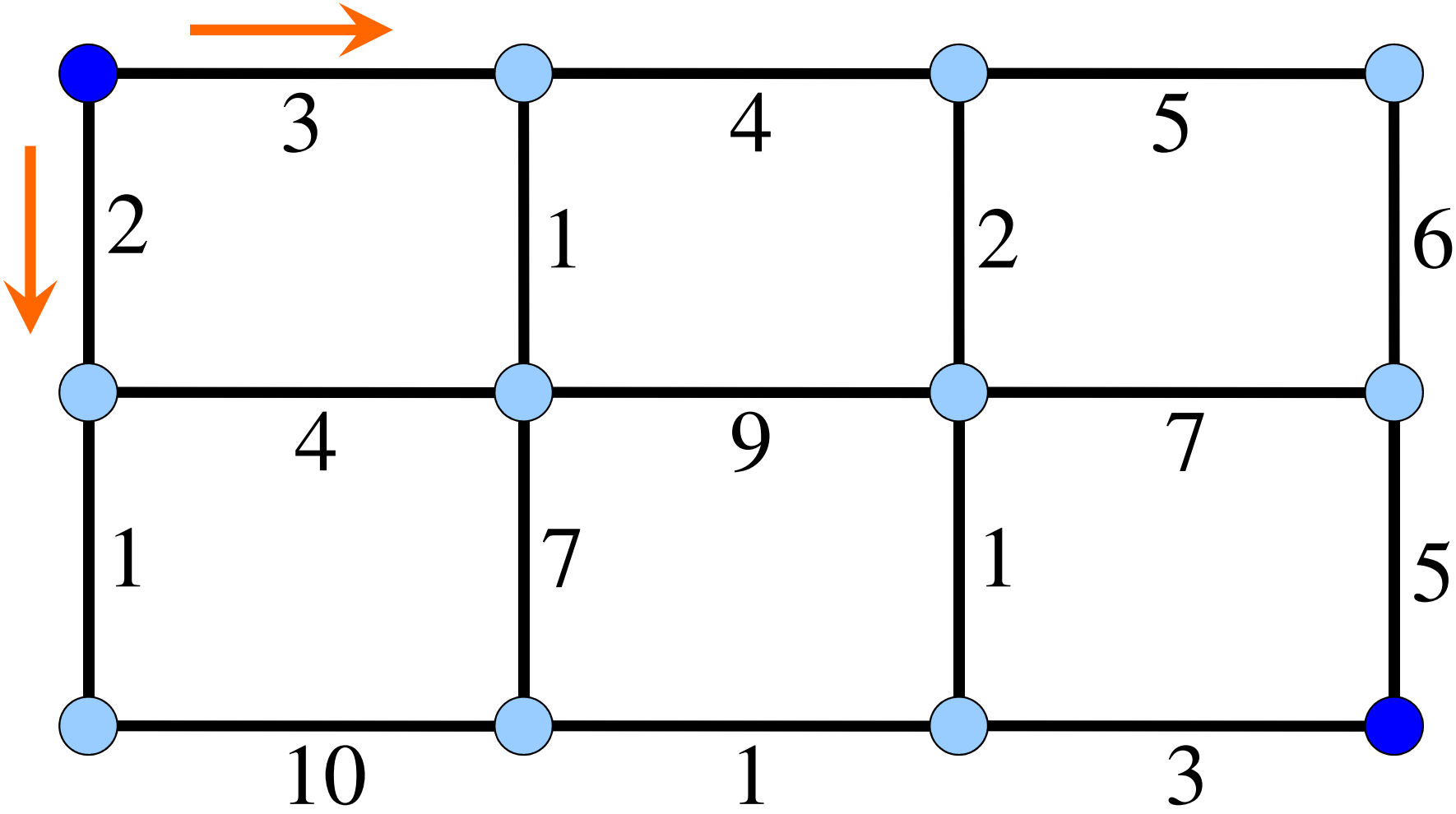
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

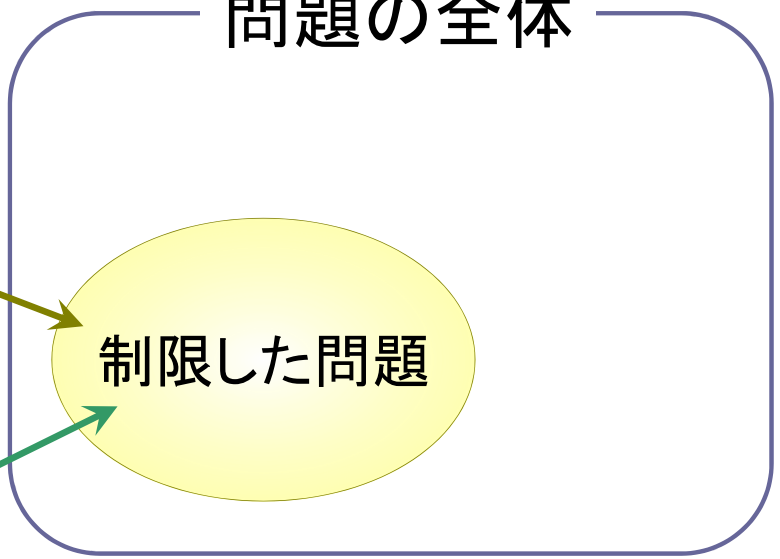


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

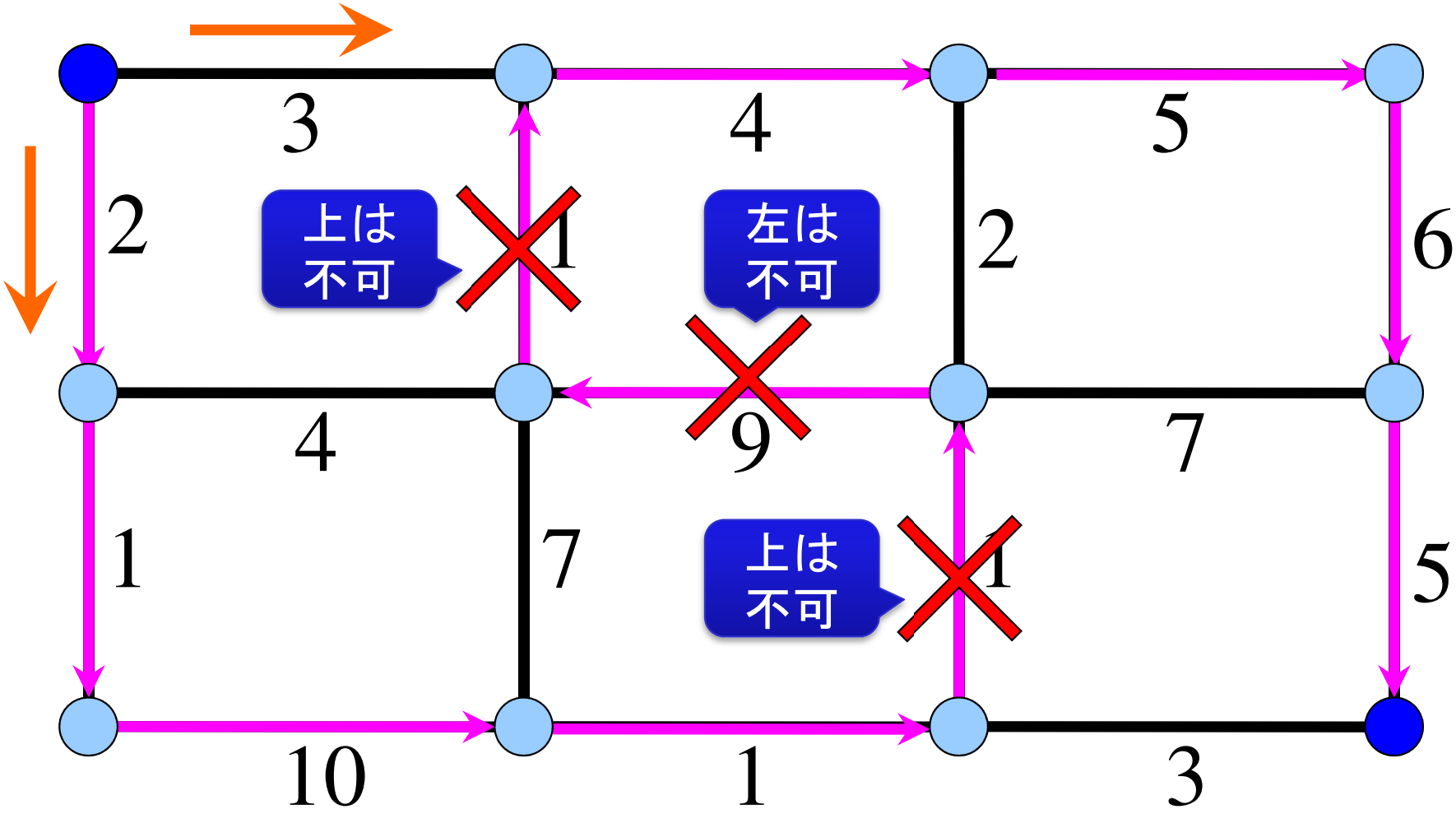
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

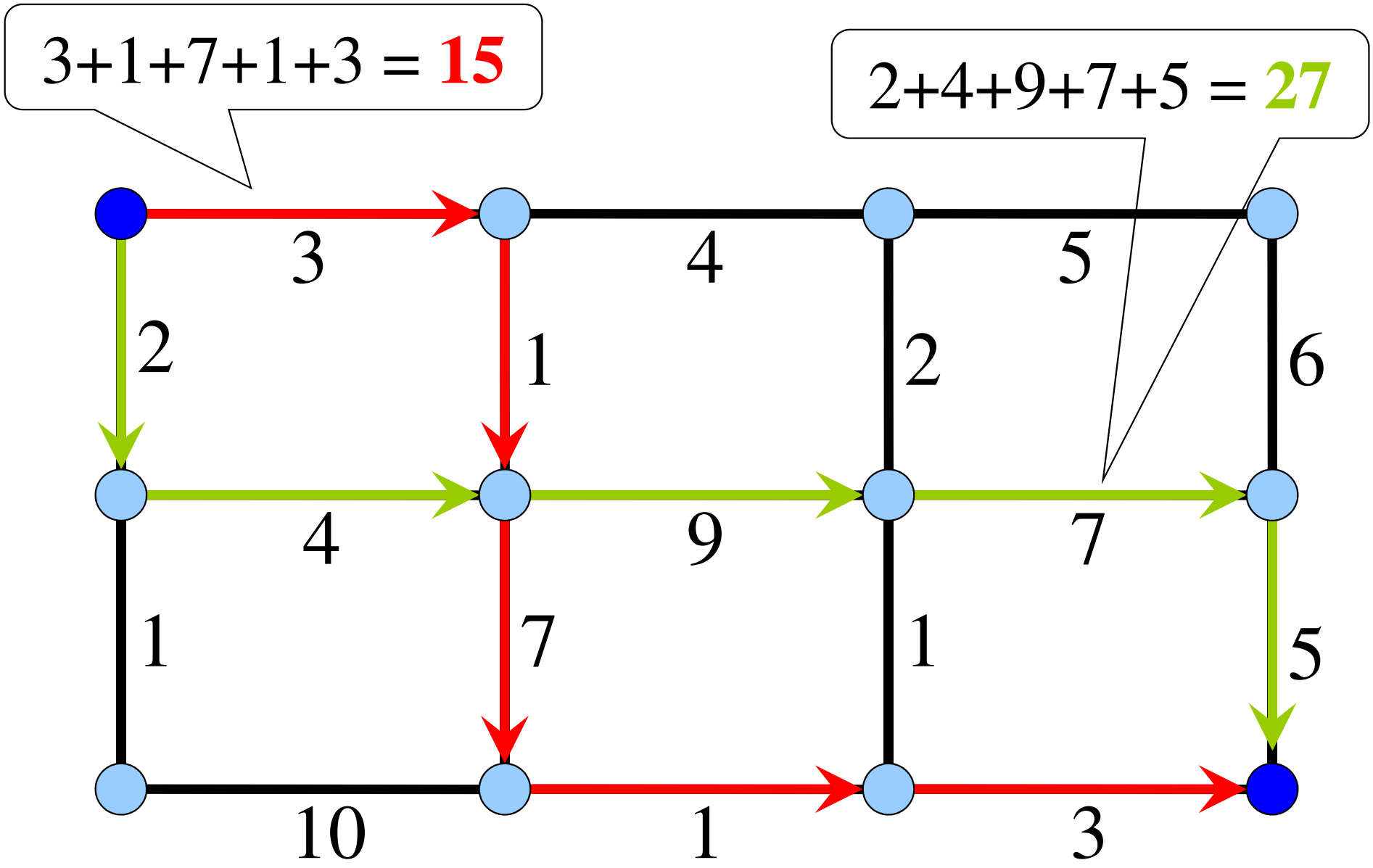
難しいなら易しくすればいいのさ！

制限した問題

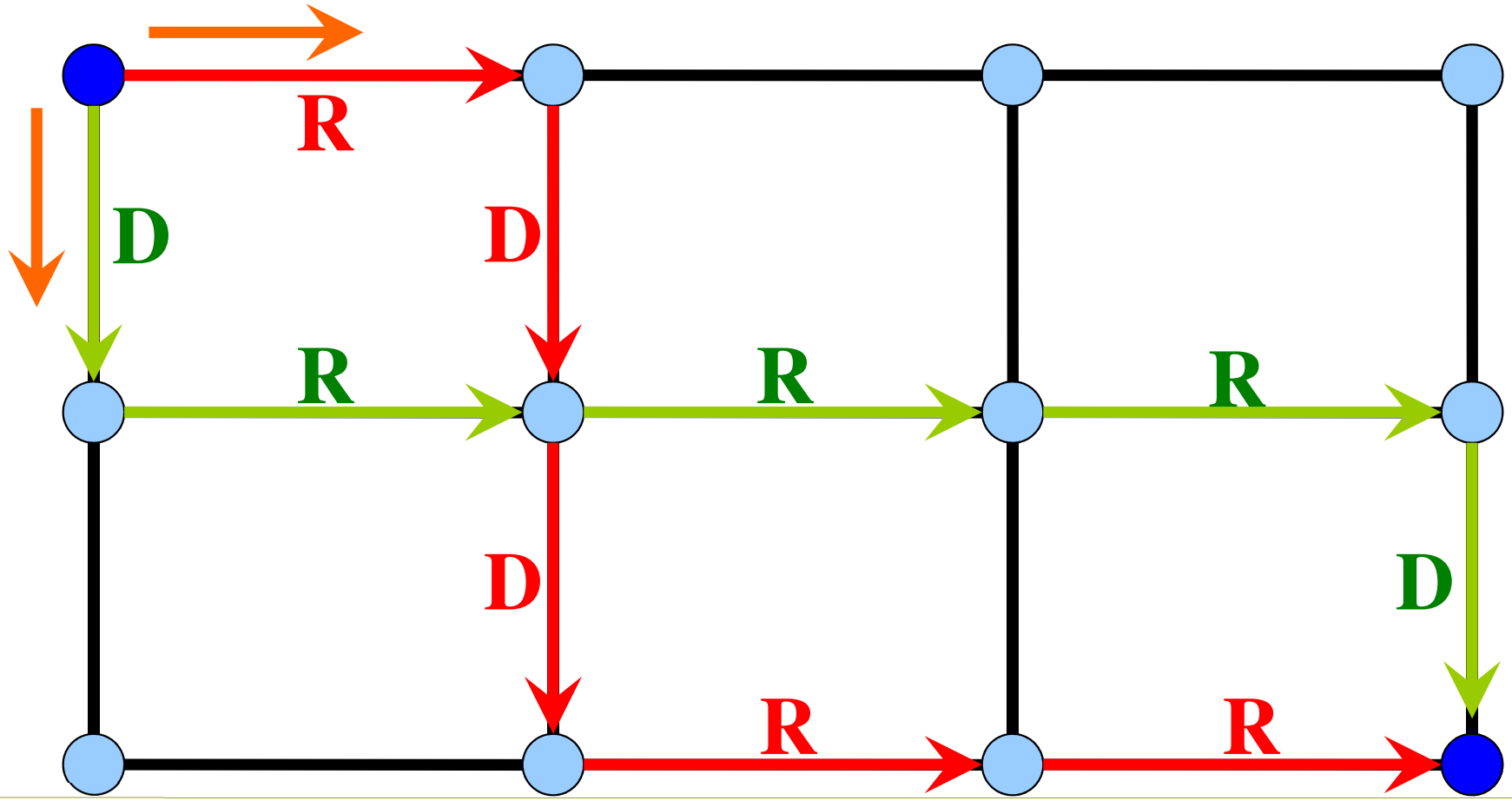
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



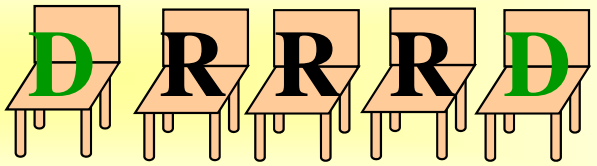
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



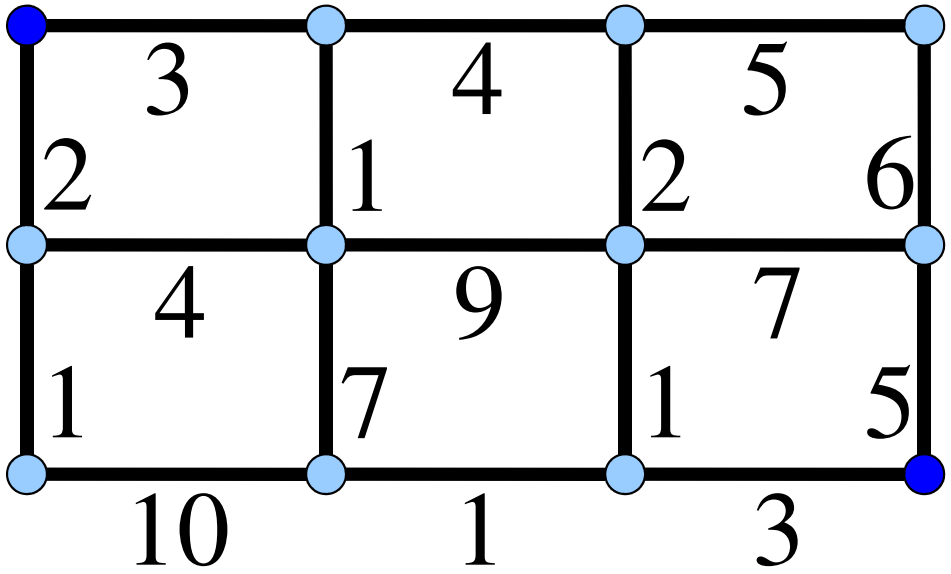
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

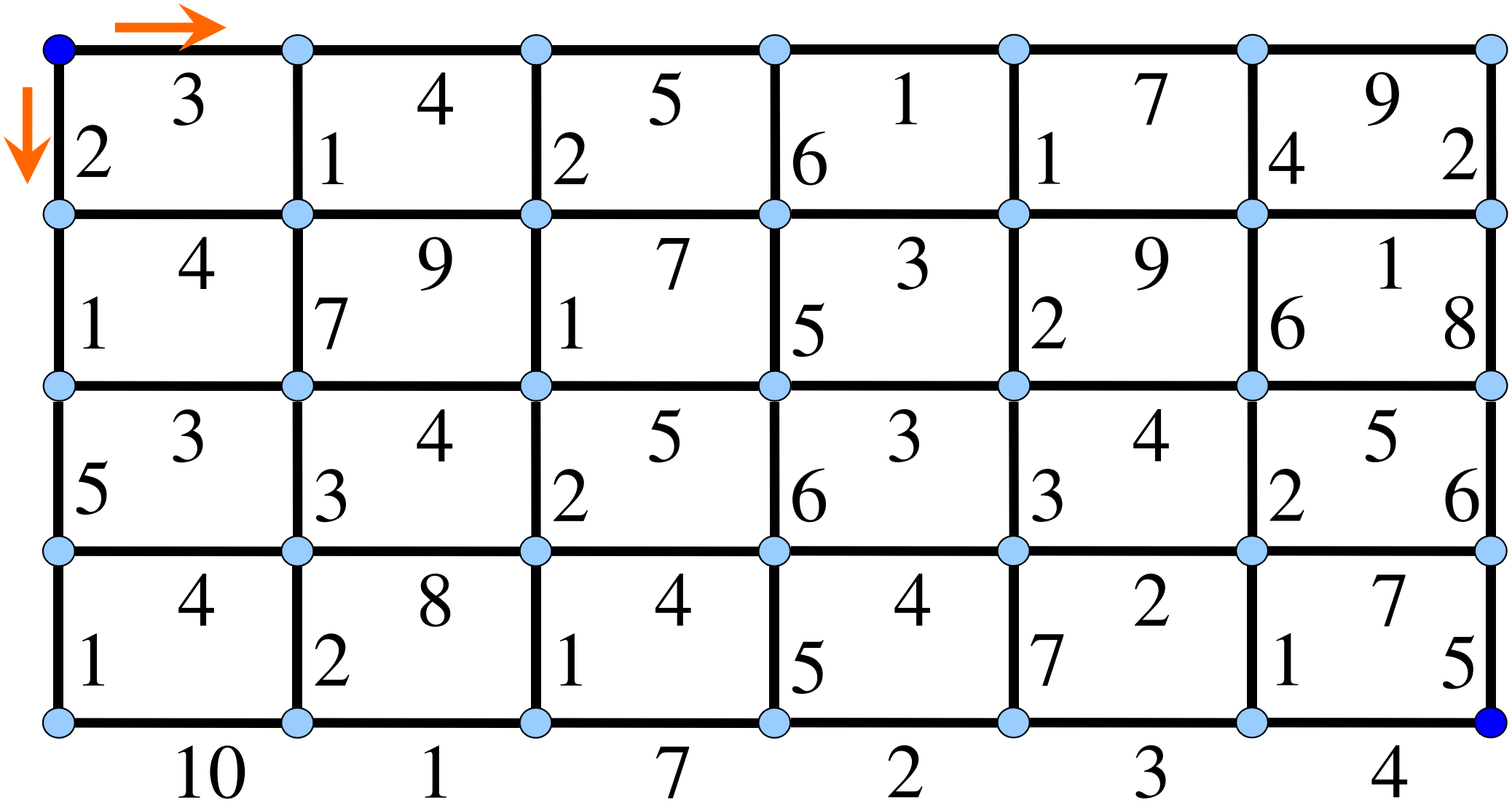
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

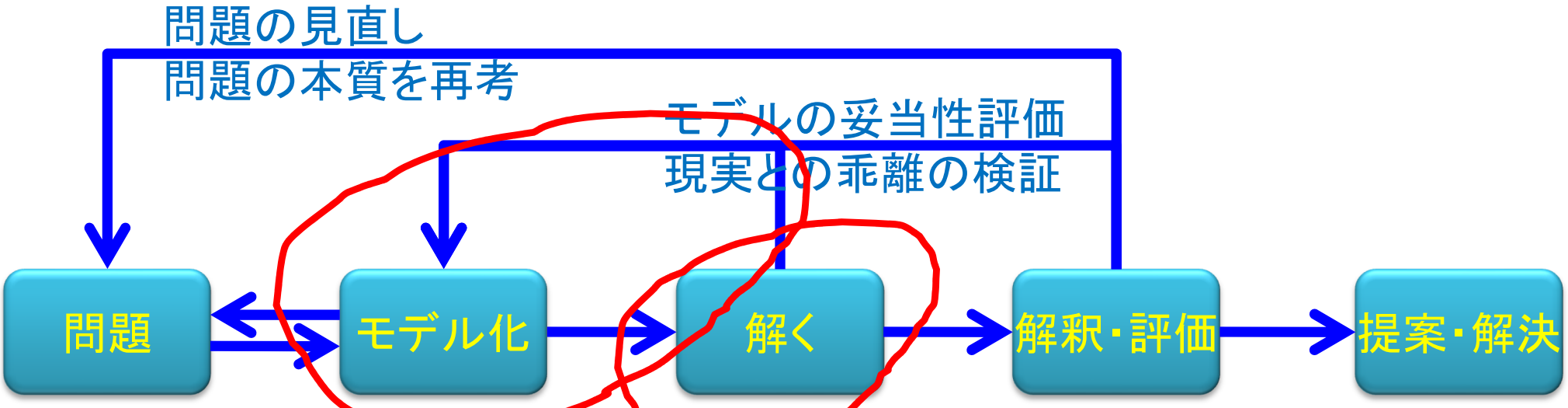
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

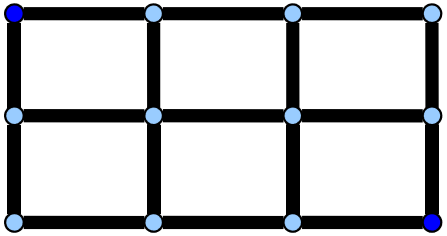
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

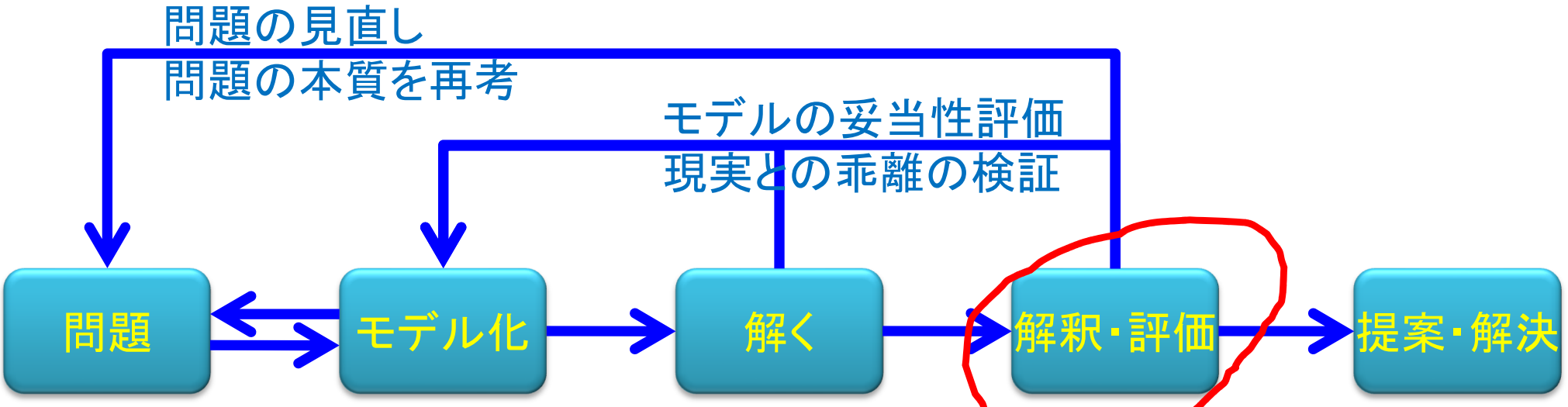
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

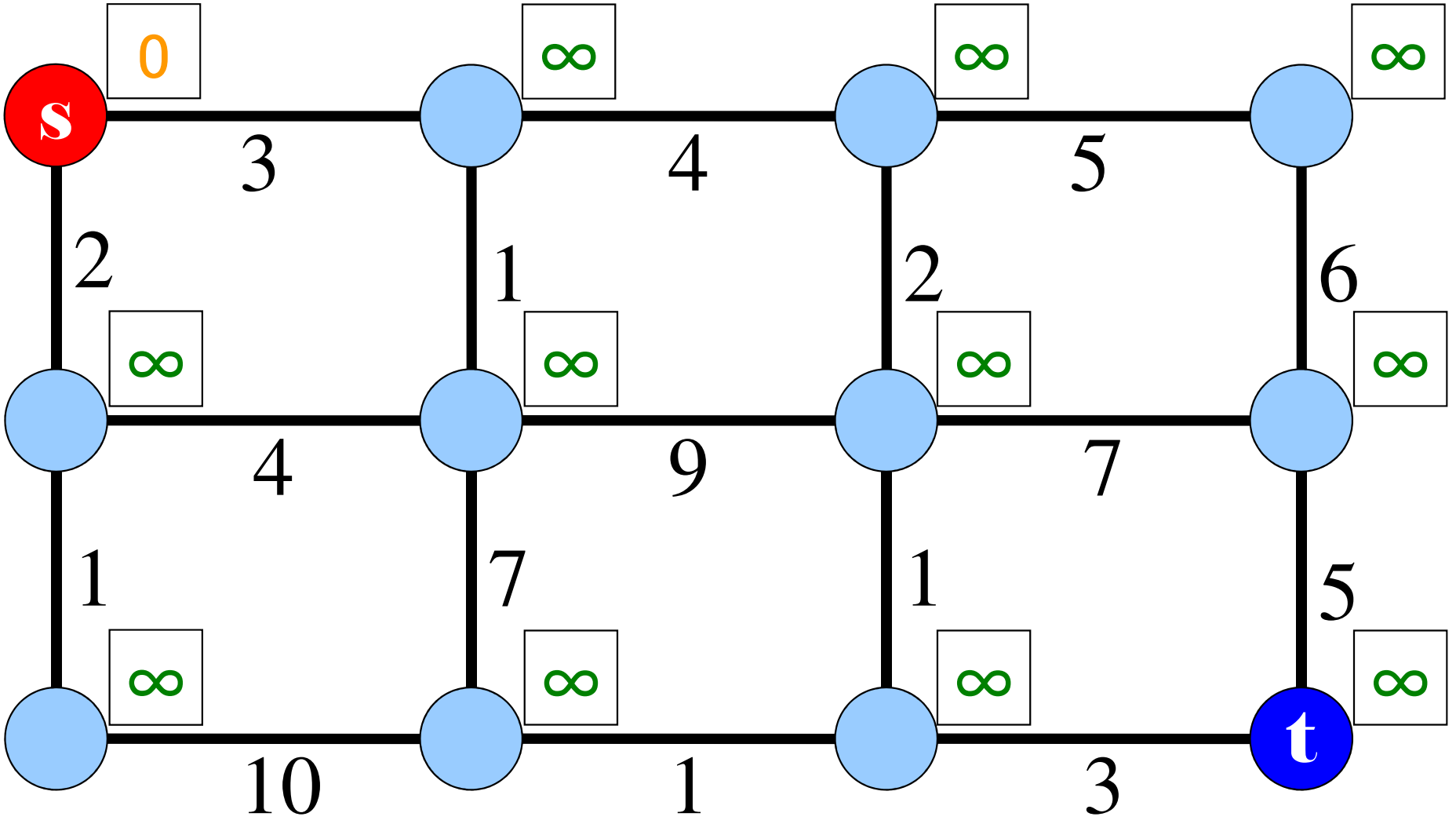
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

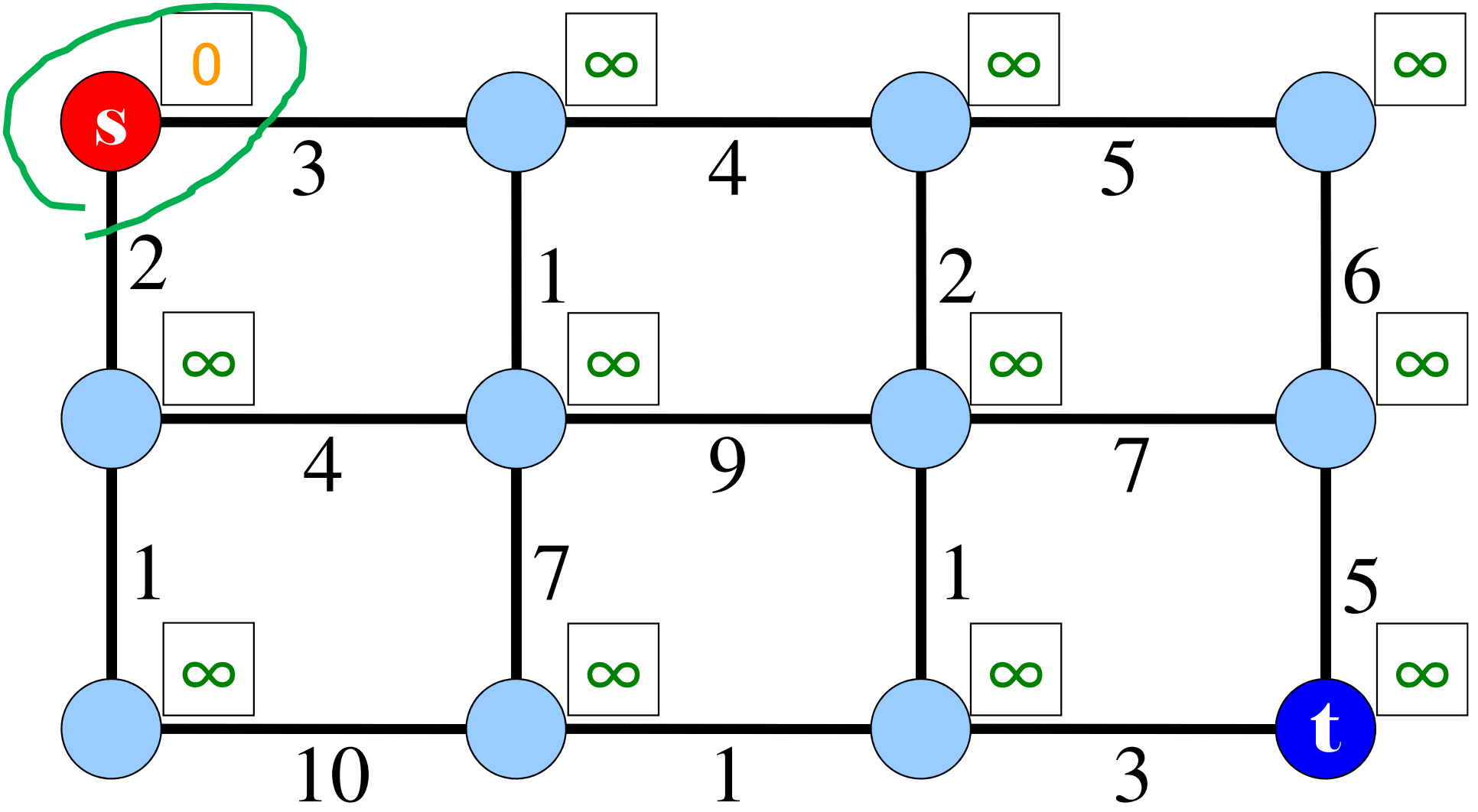
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



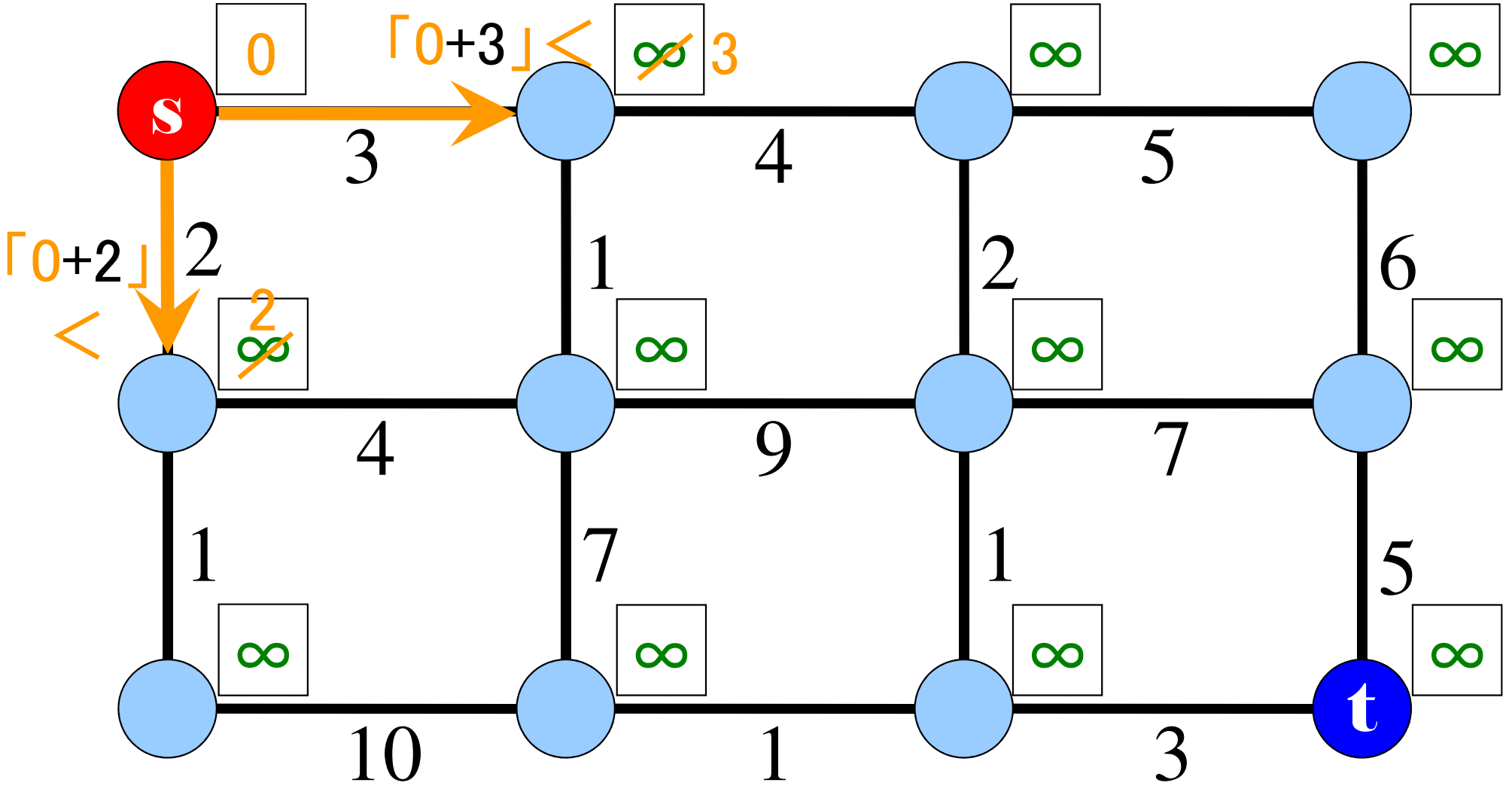
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

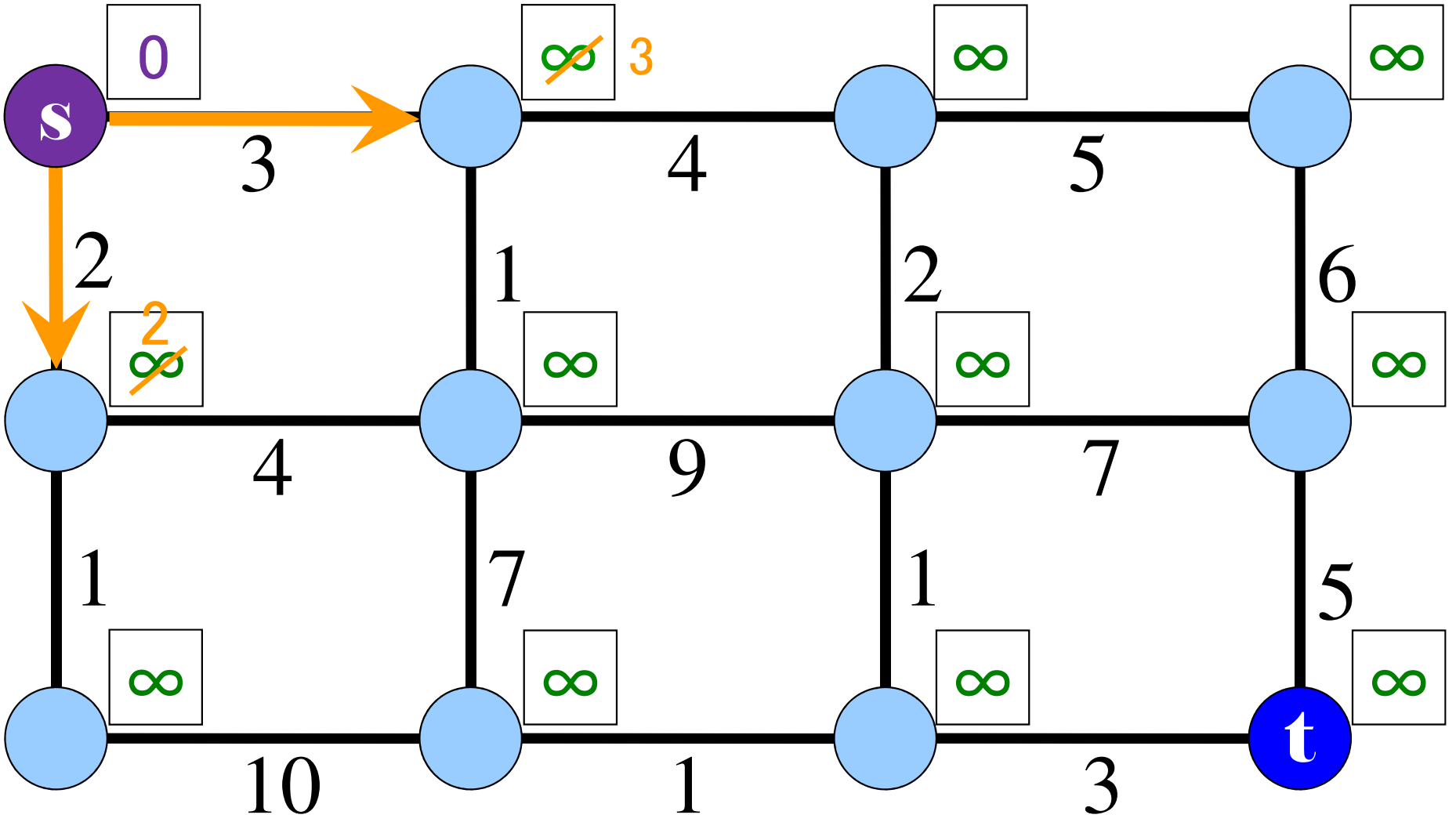


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

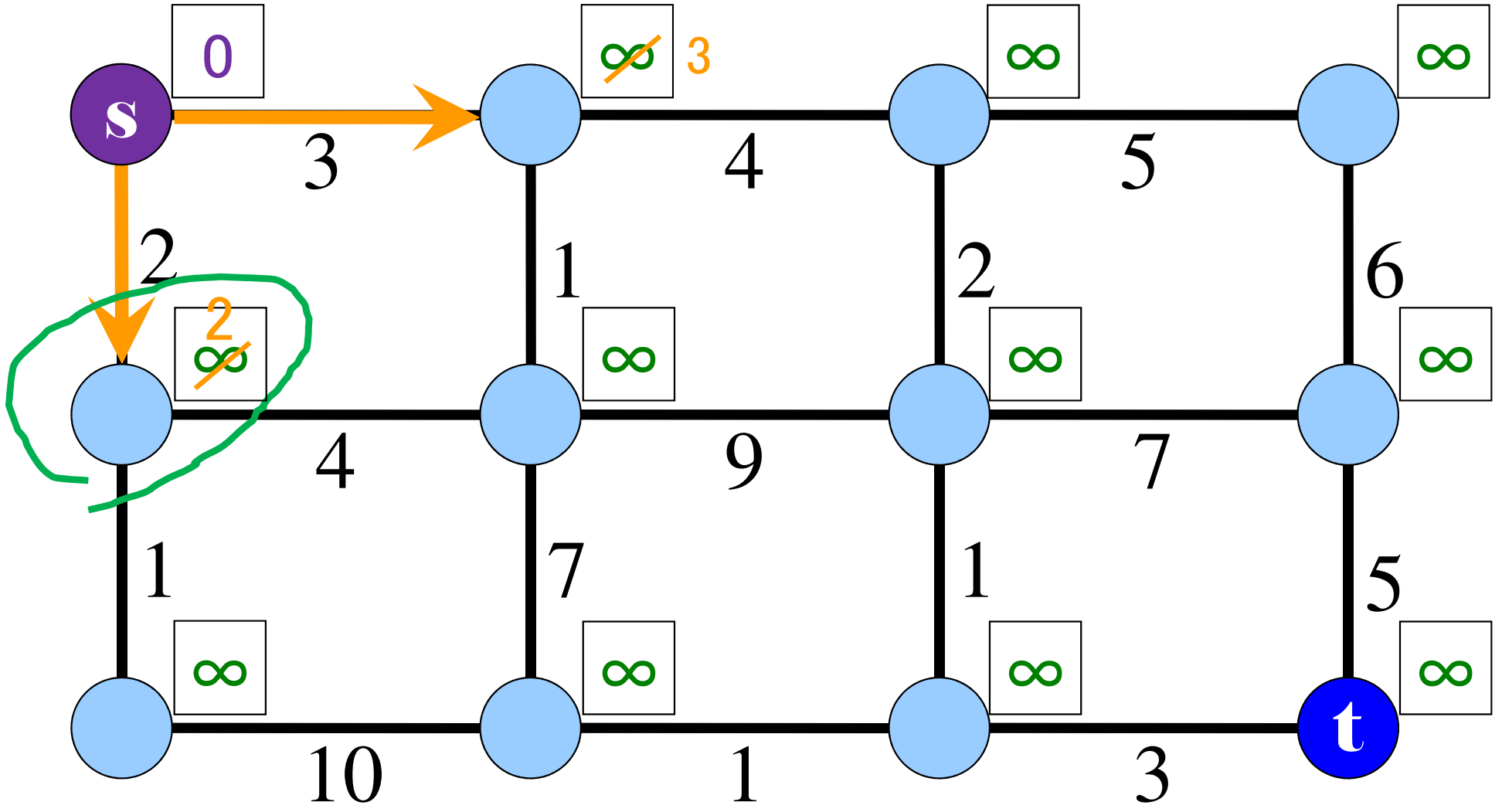
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



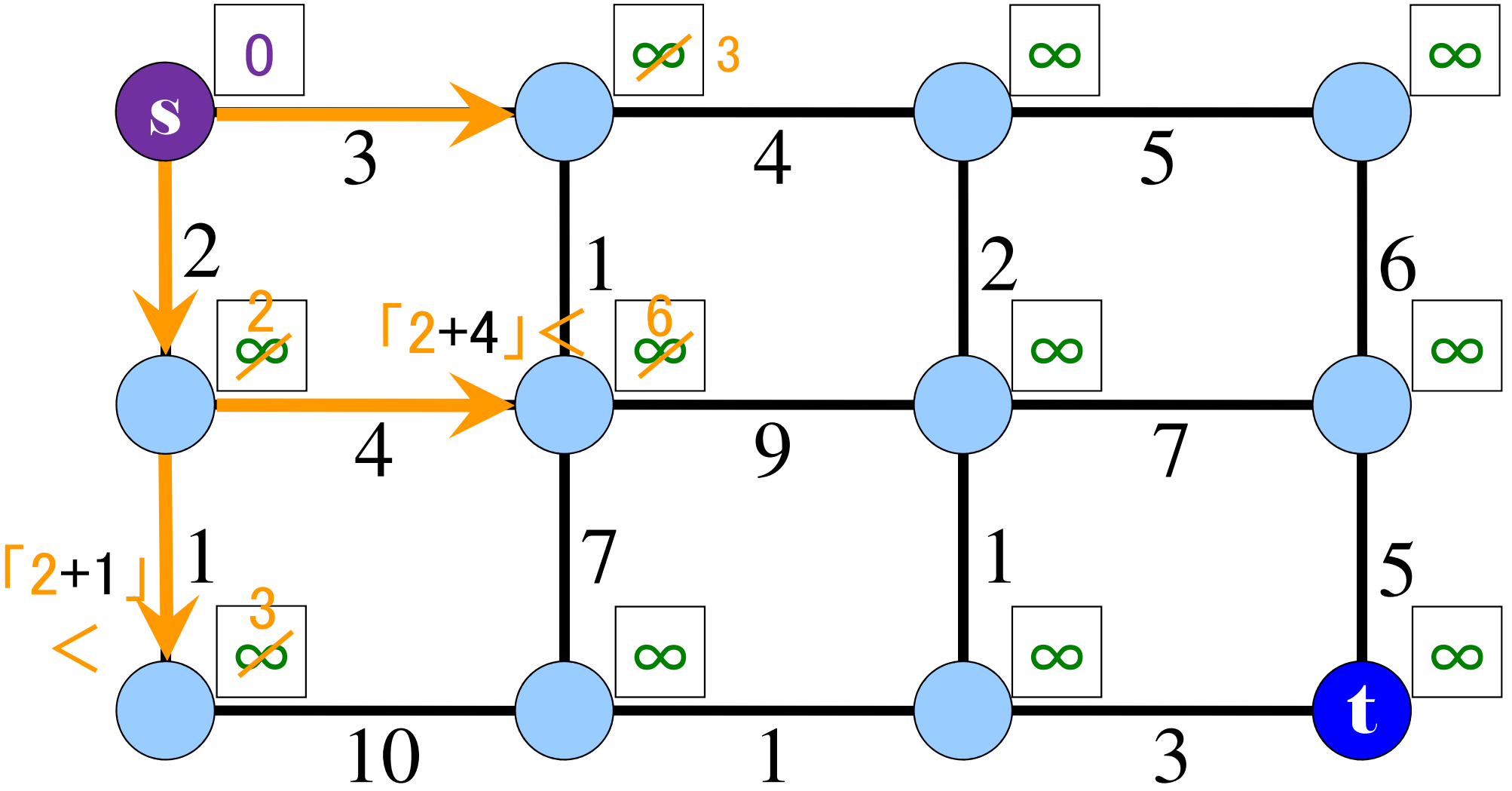
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

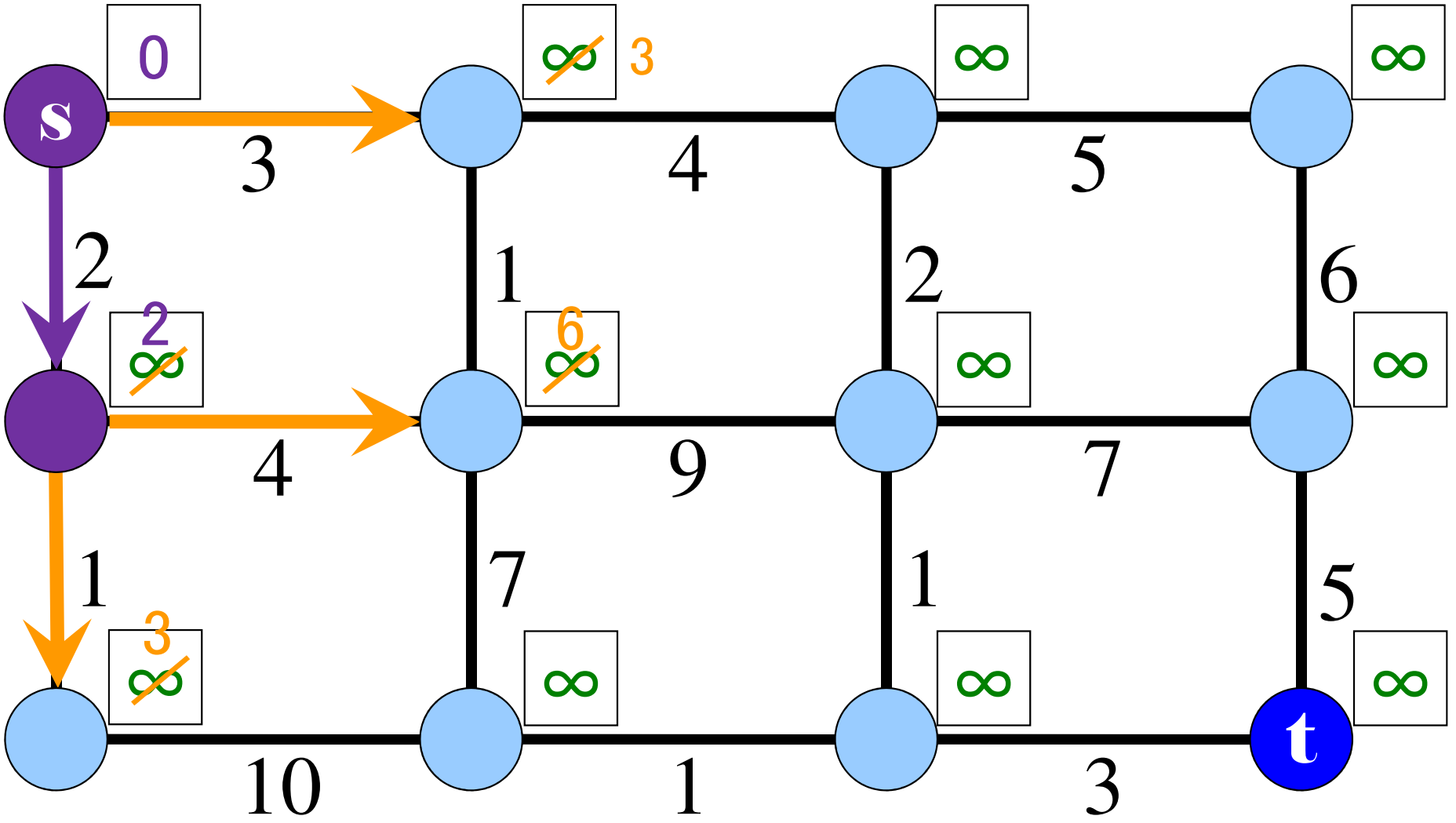


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

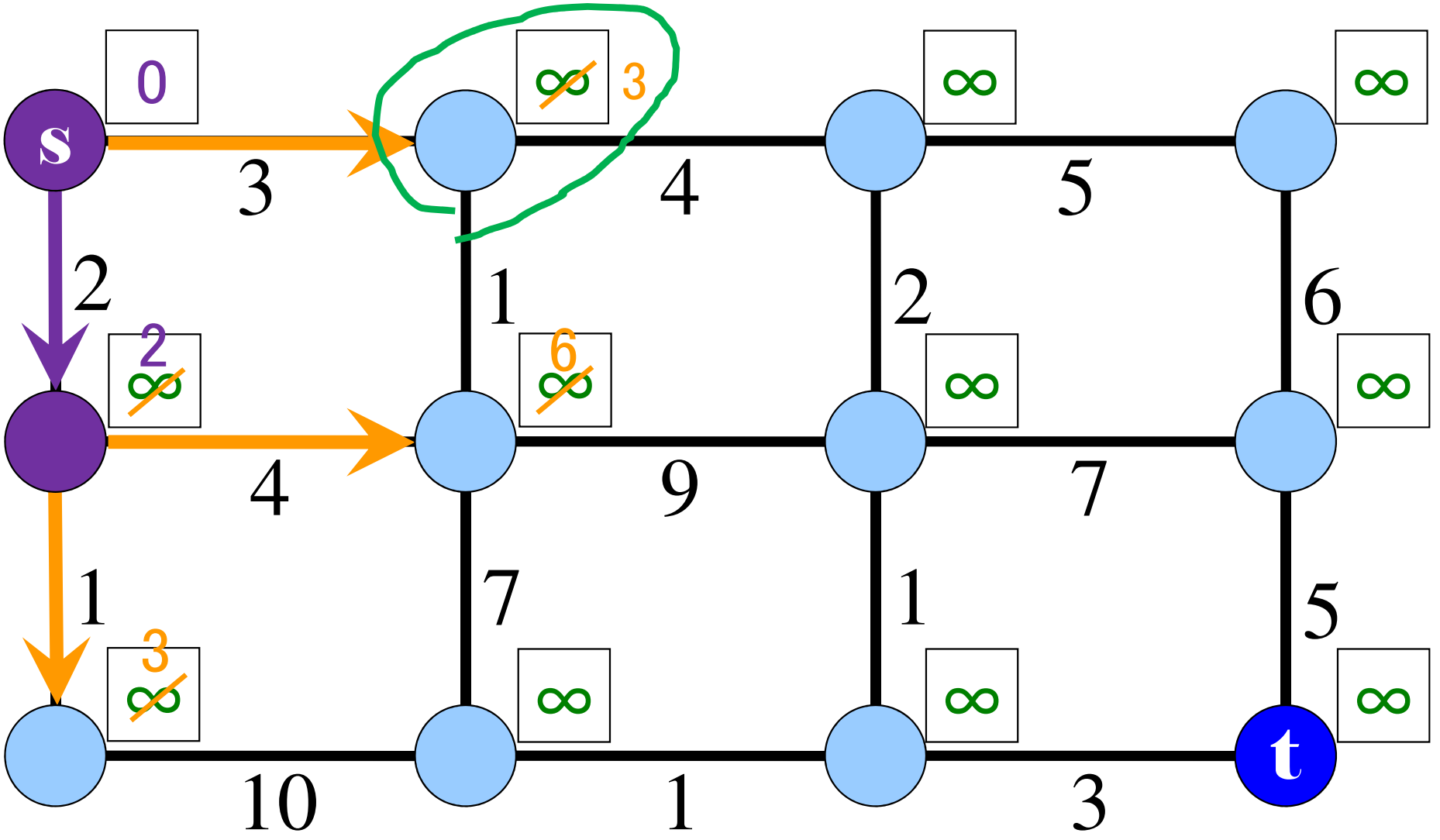
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



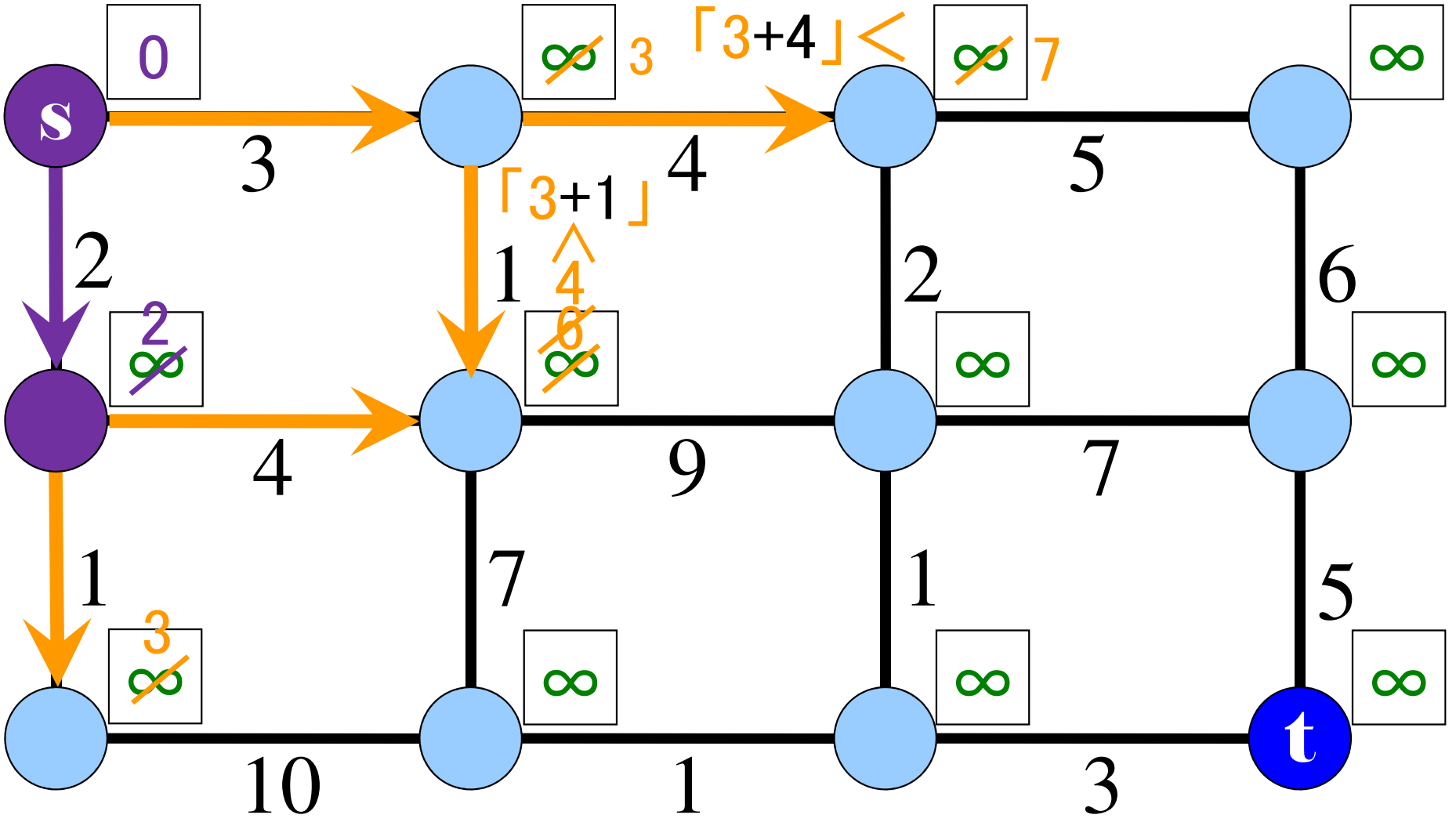
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

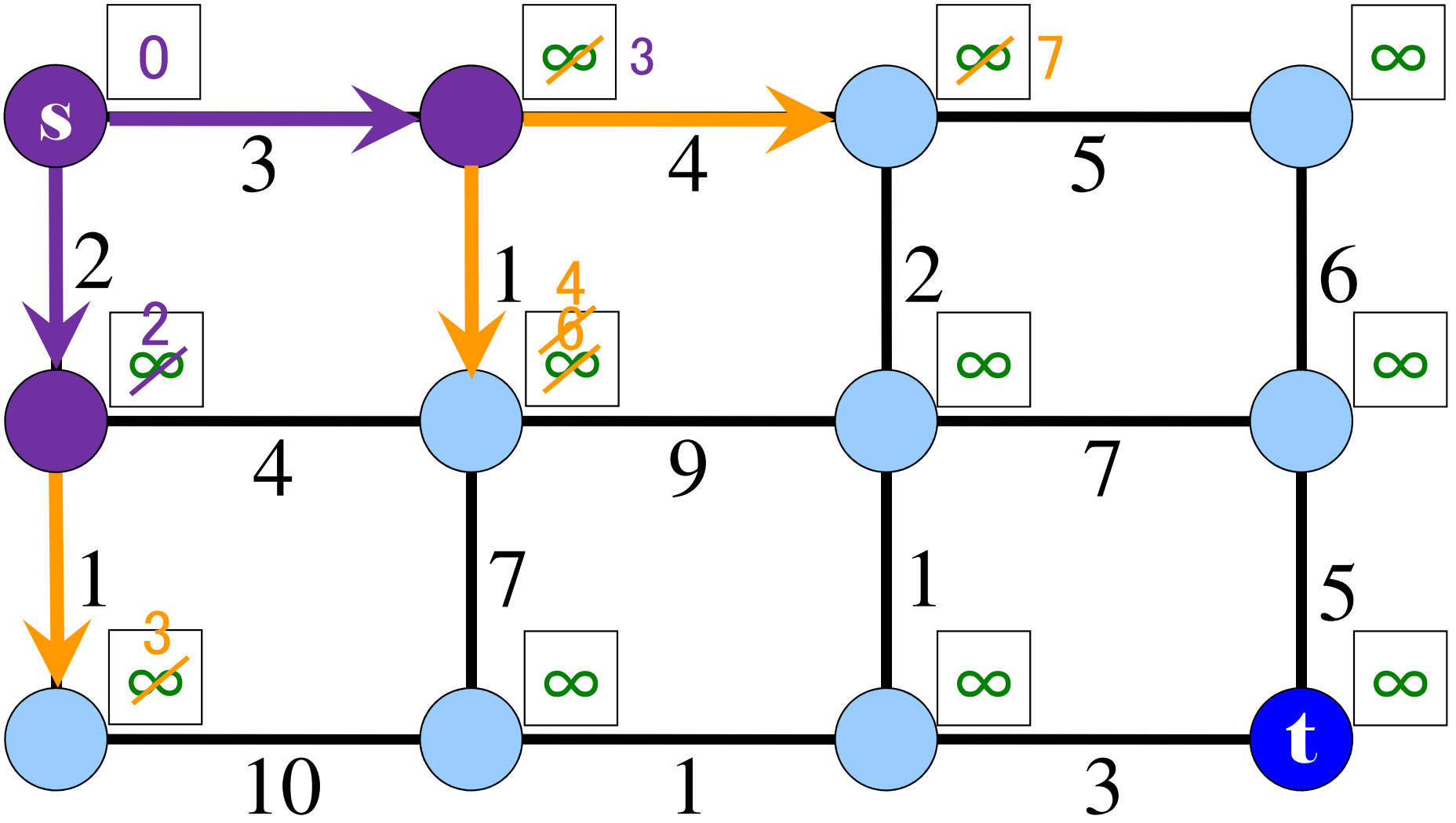


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

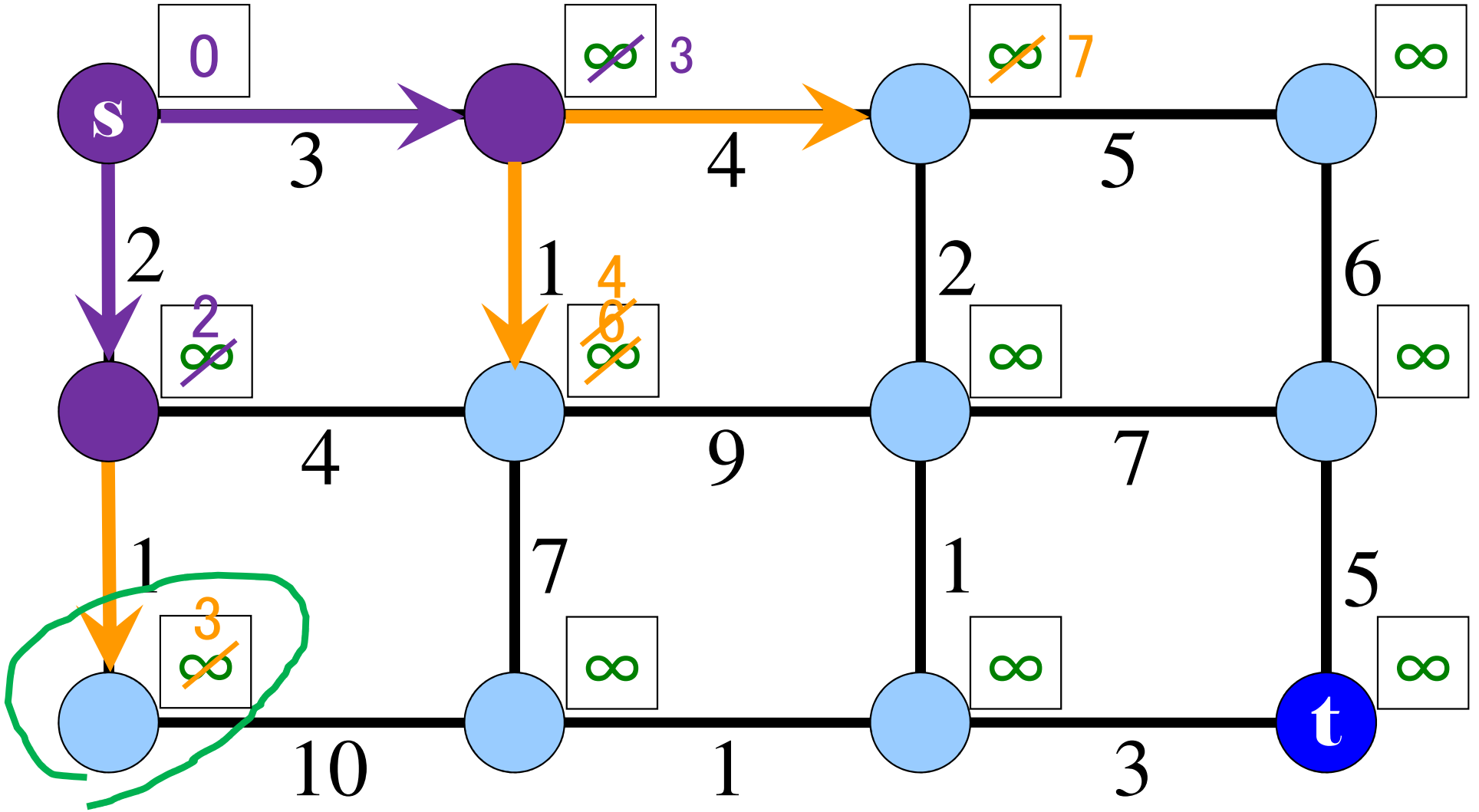
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



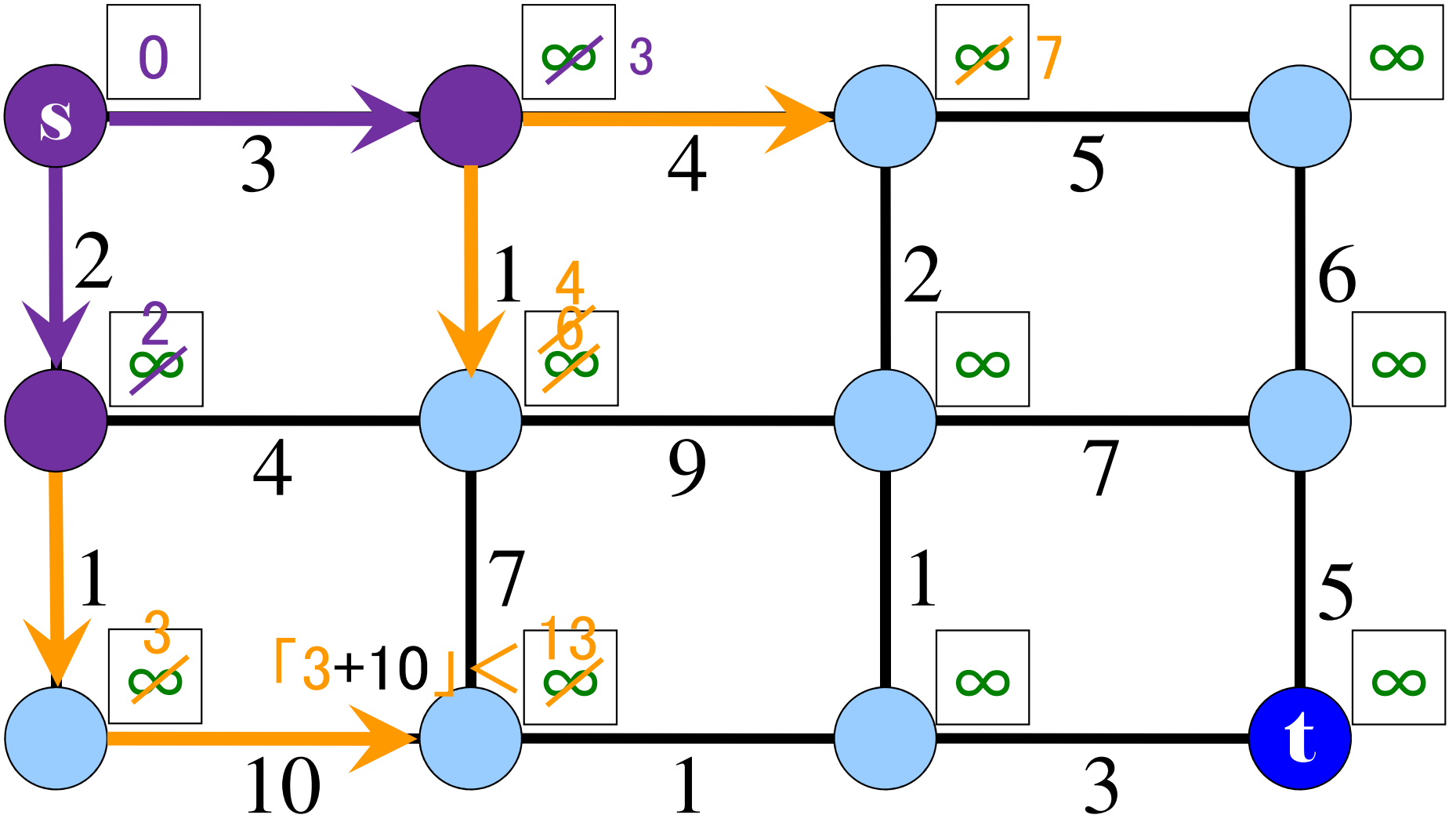
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

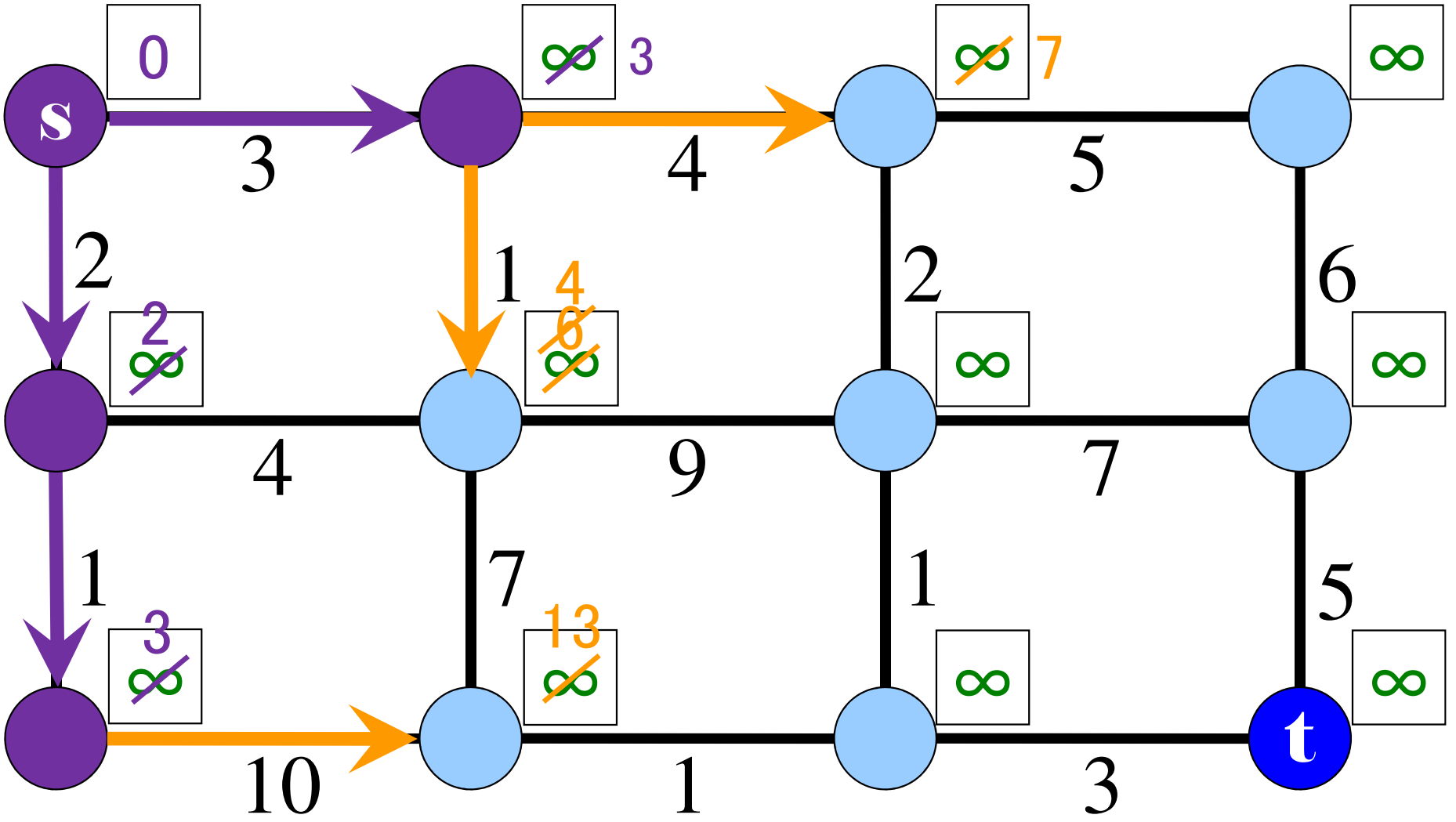


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

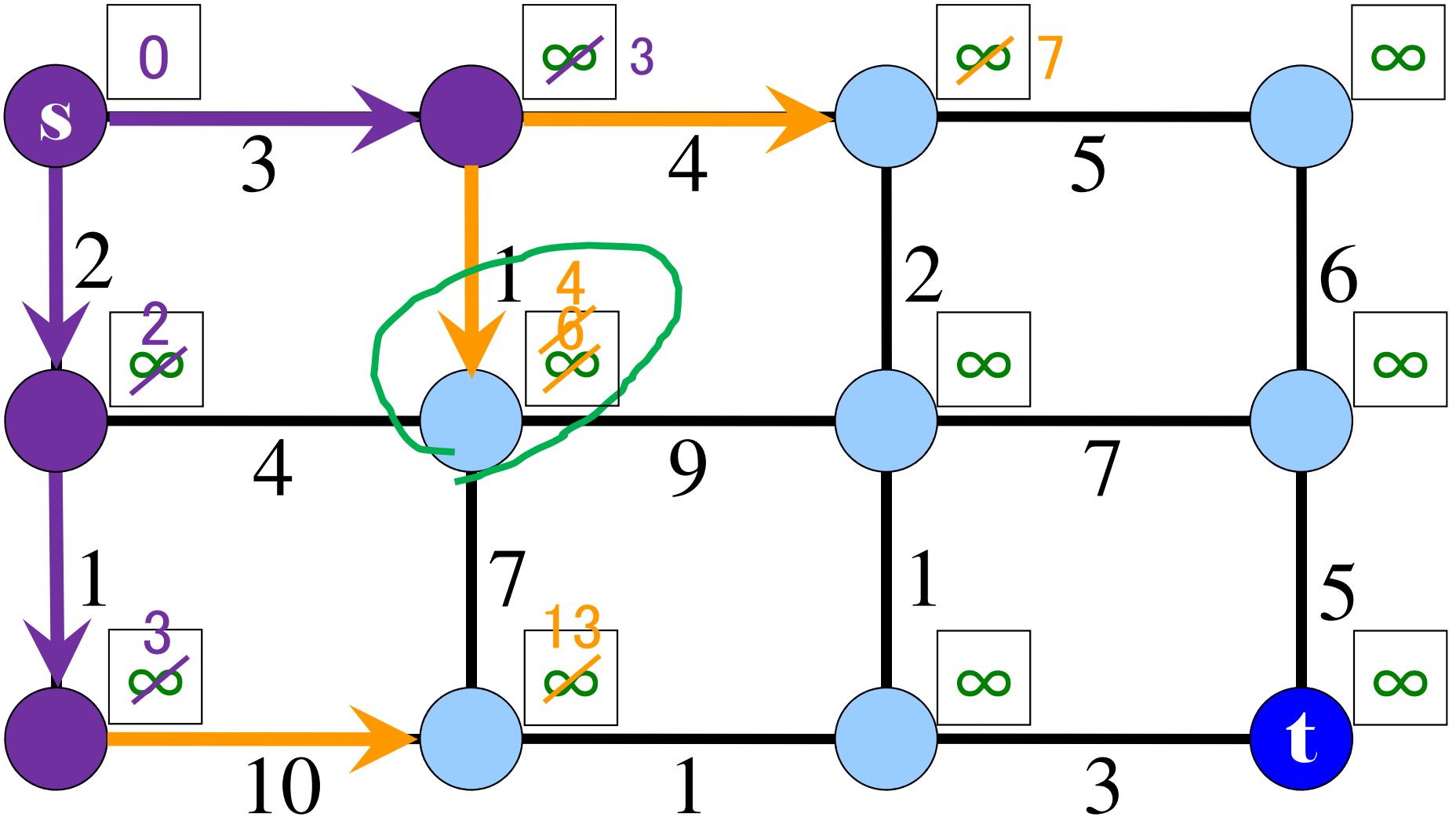
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



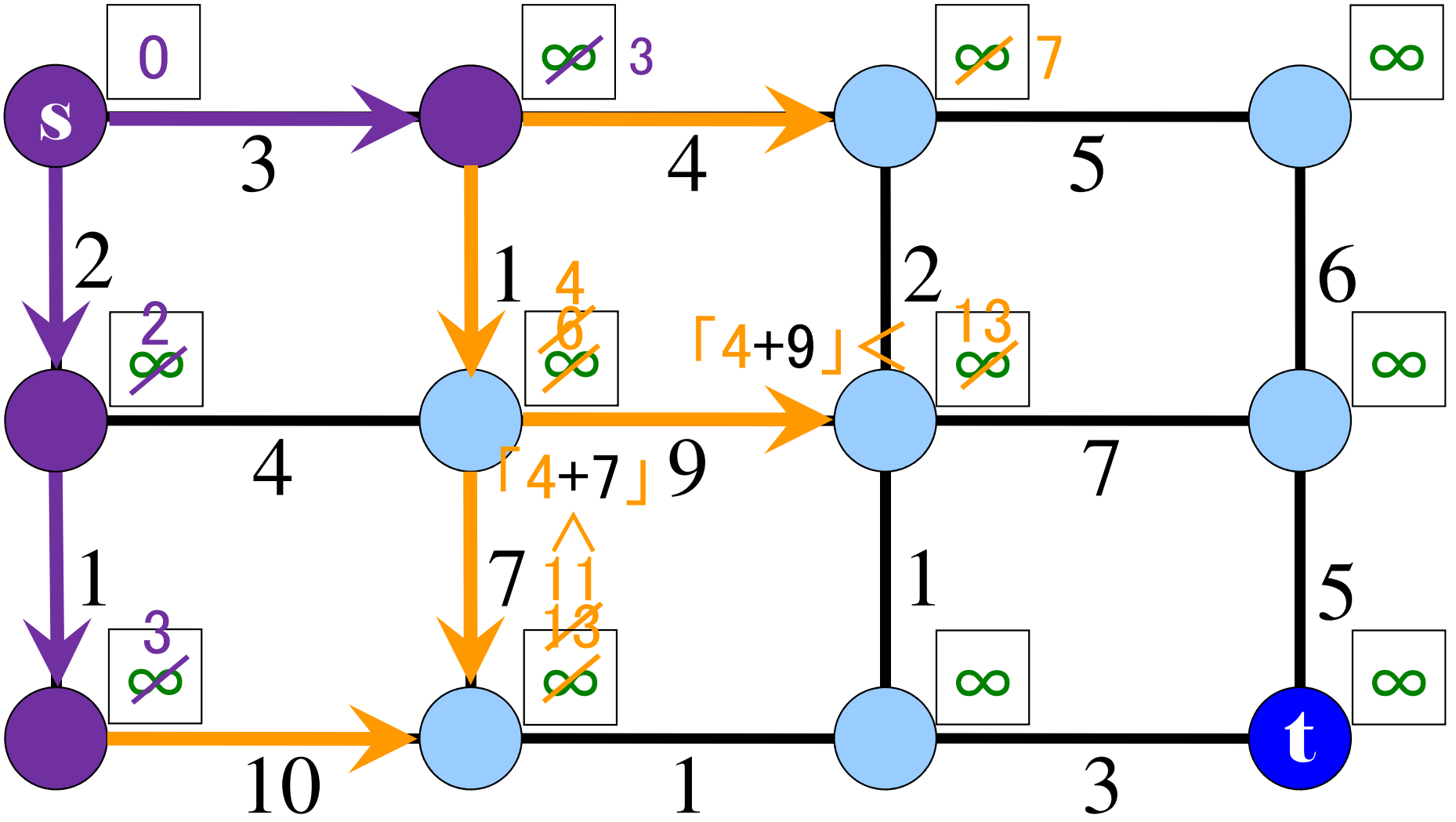
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

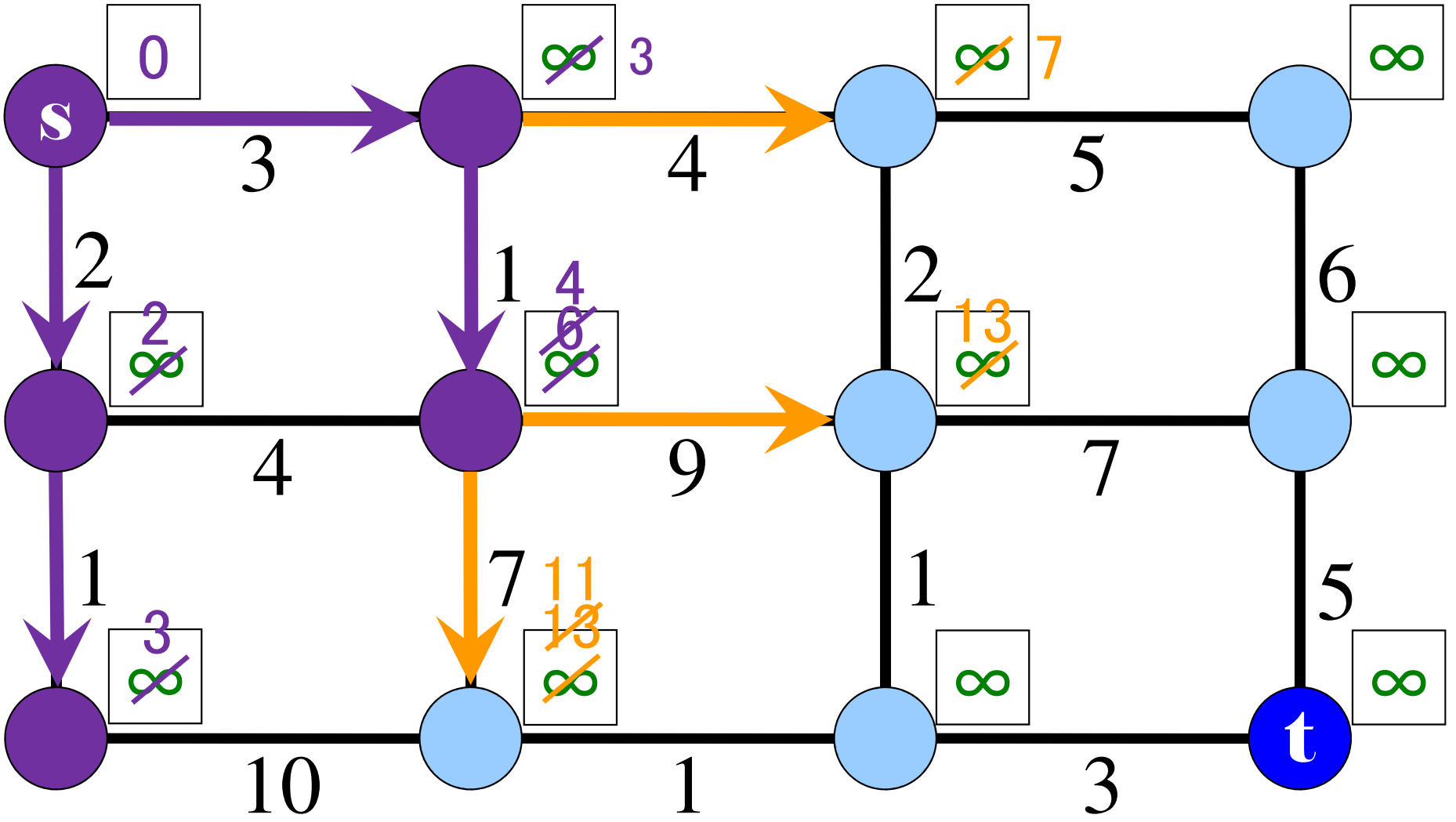


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

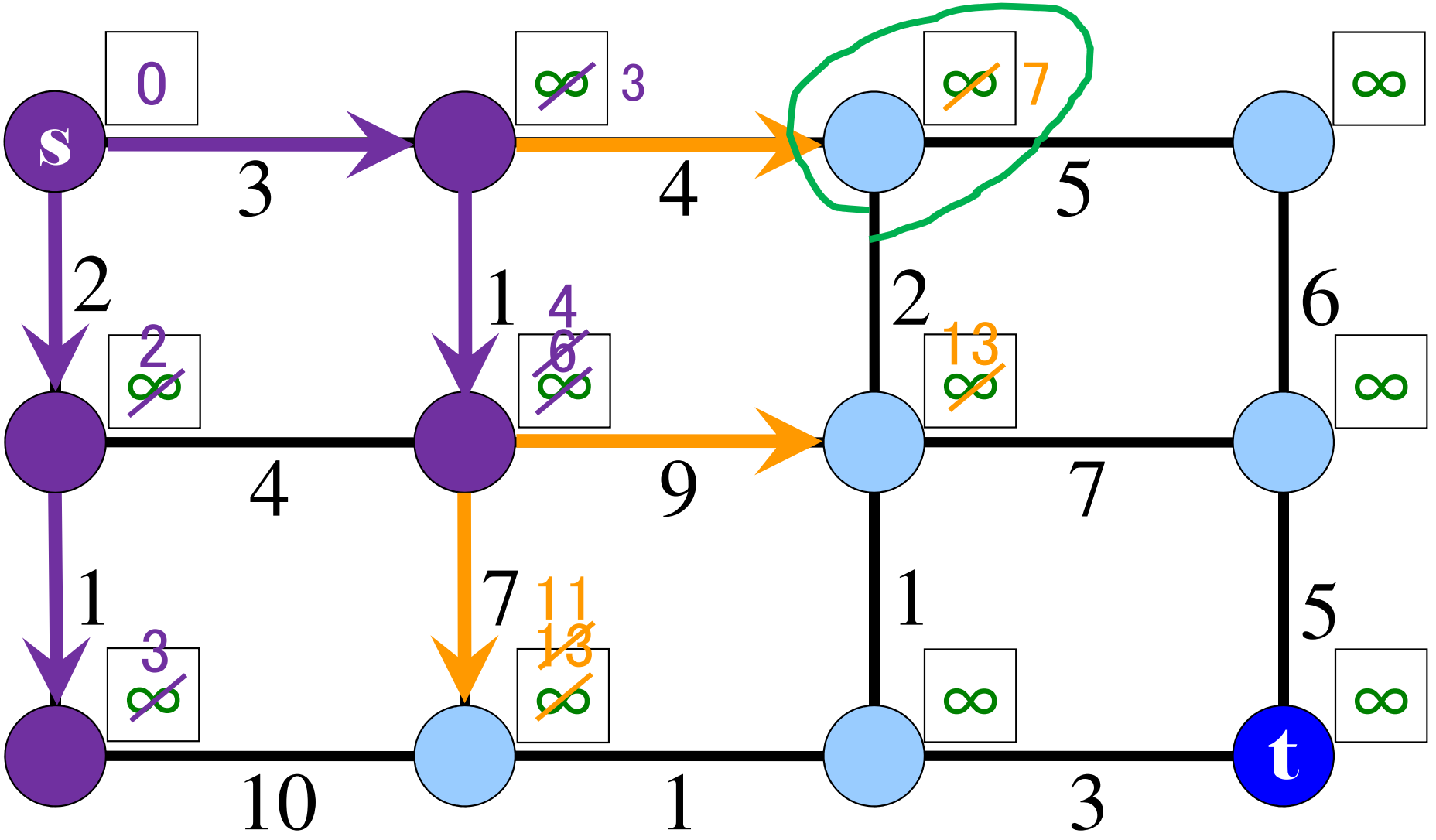
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



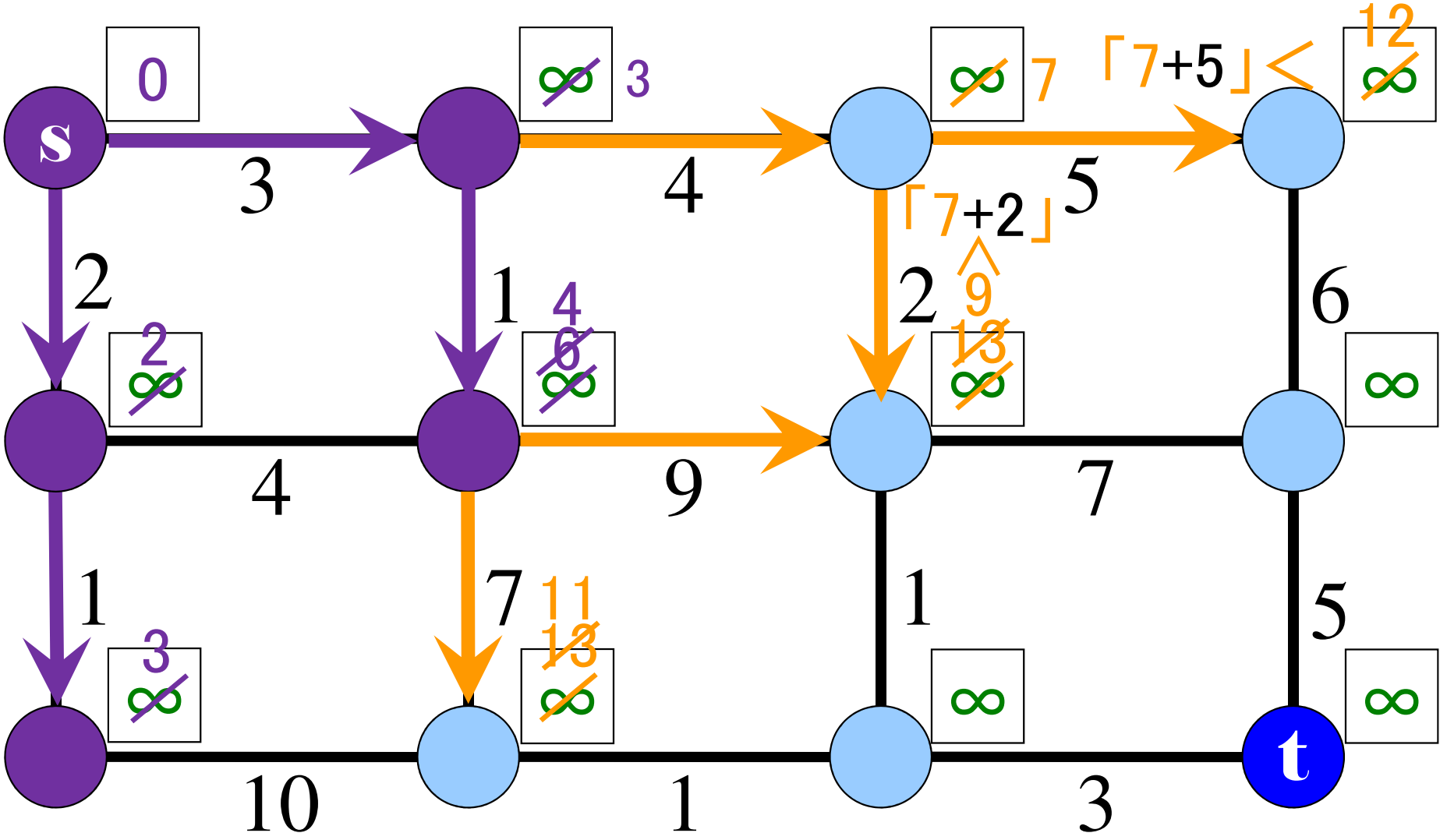
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

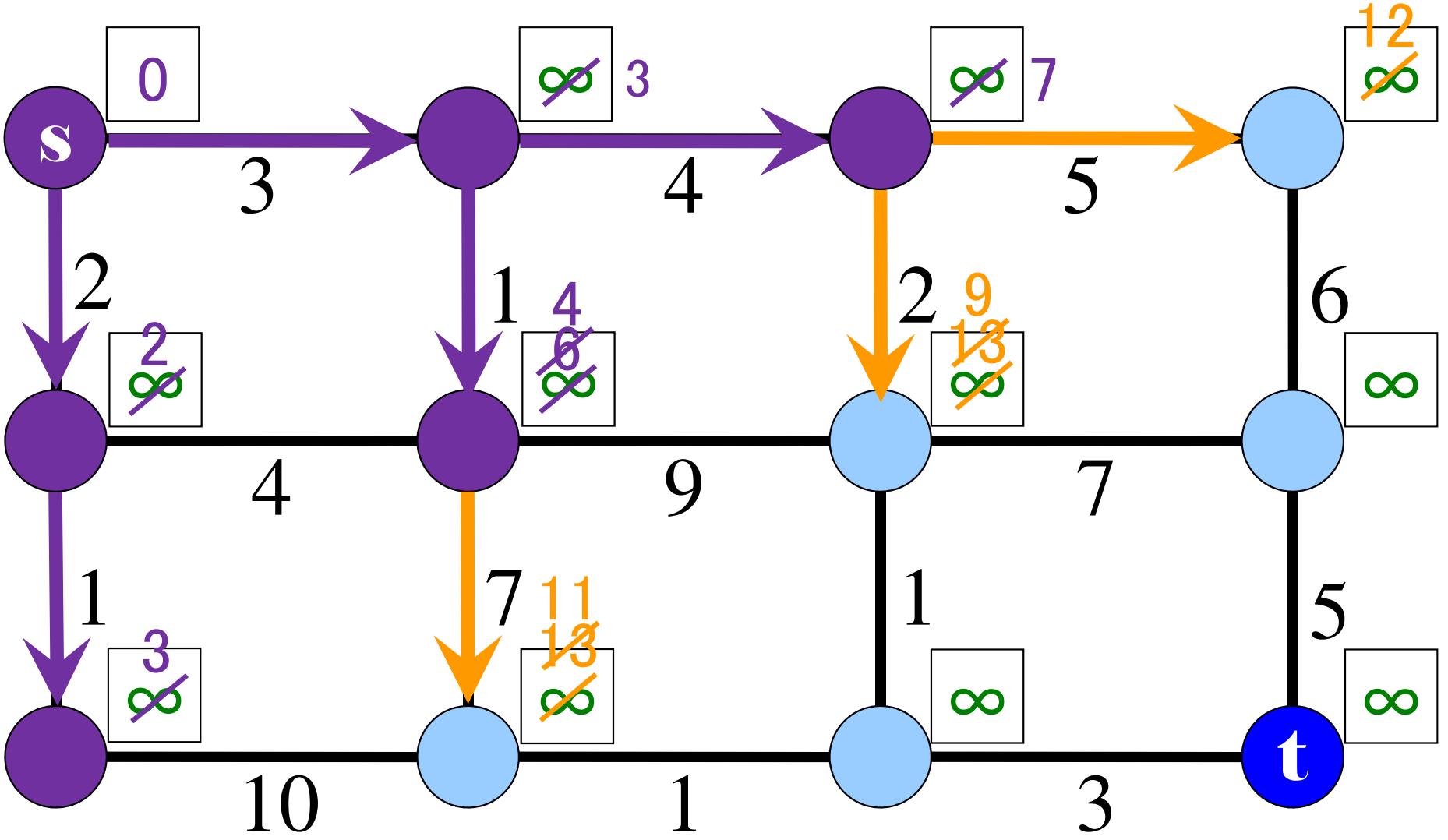


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

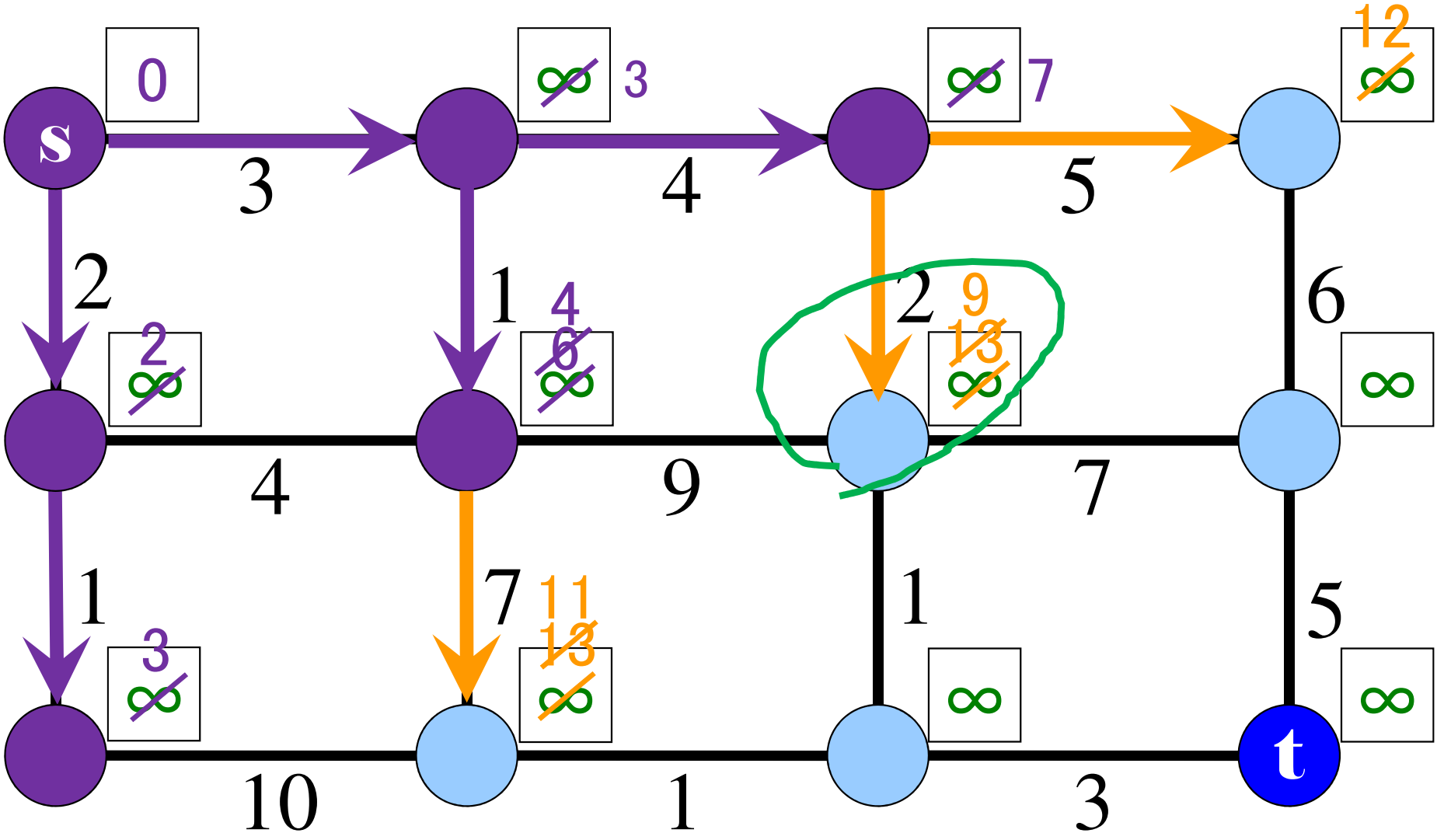
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



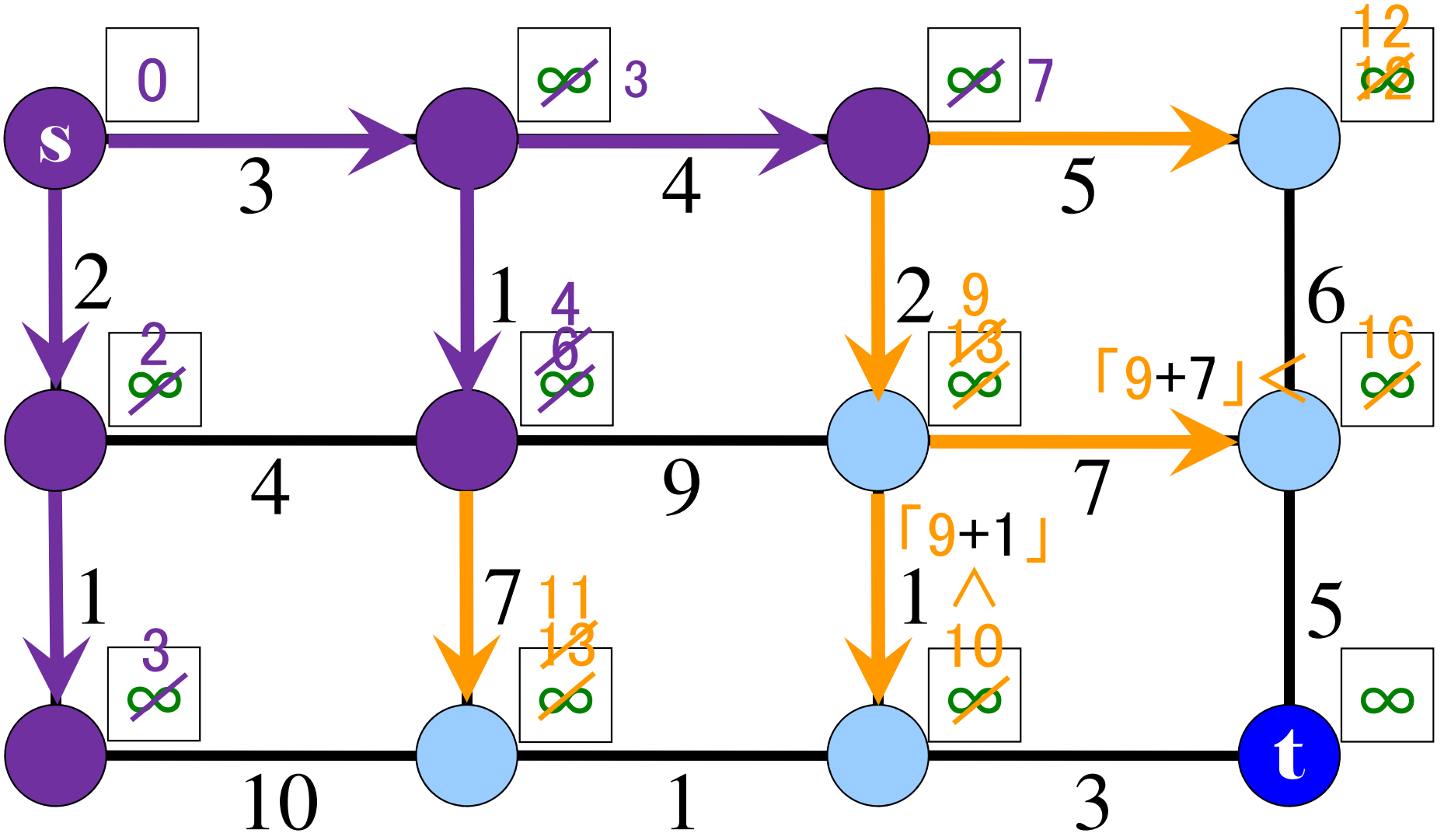
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

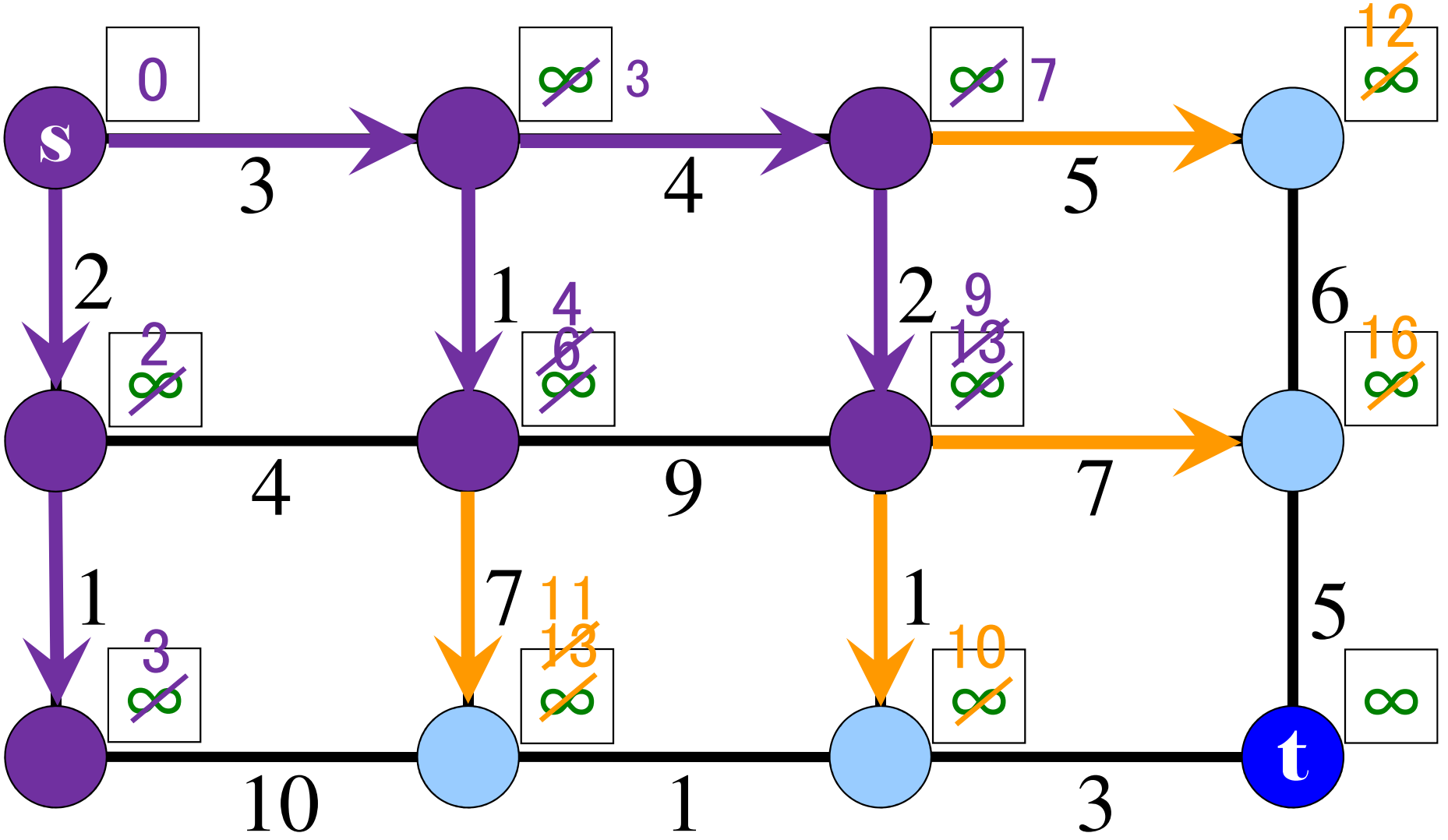


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

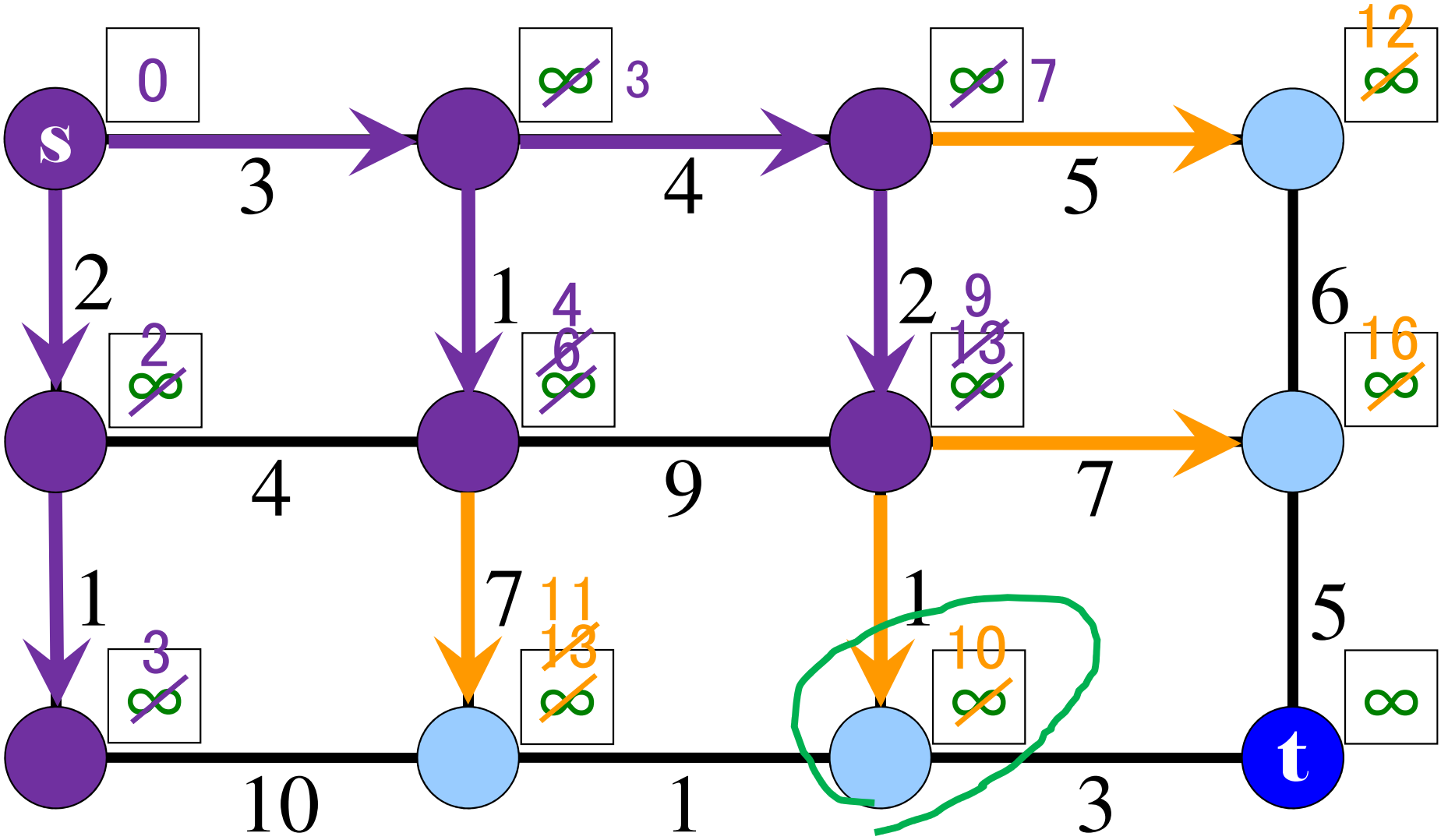
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



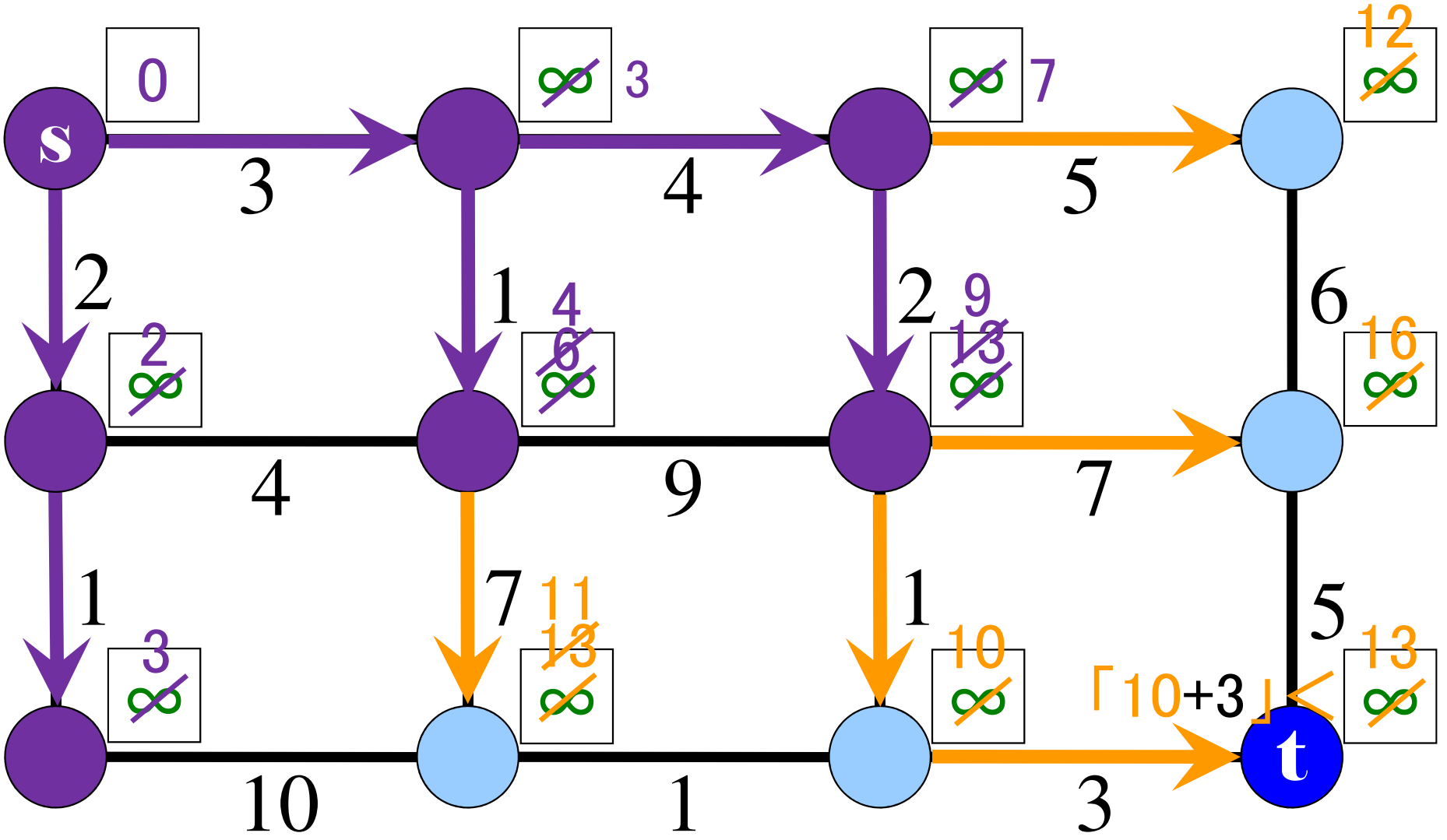
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

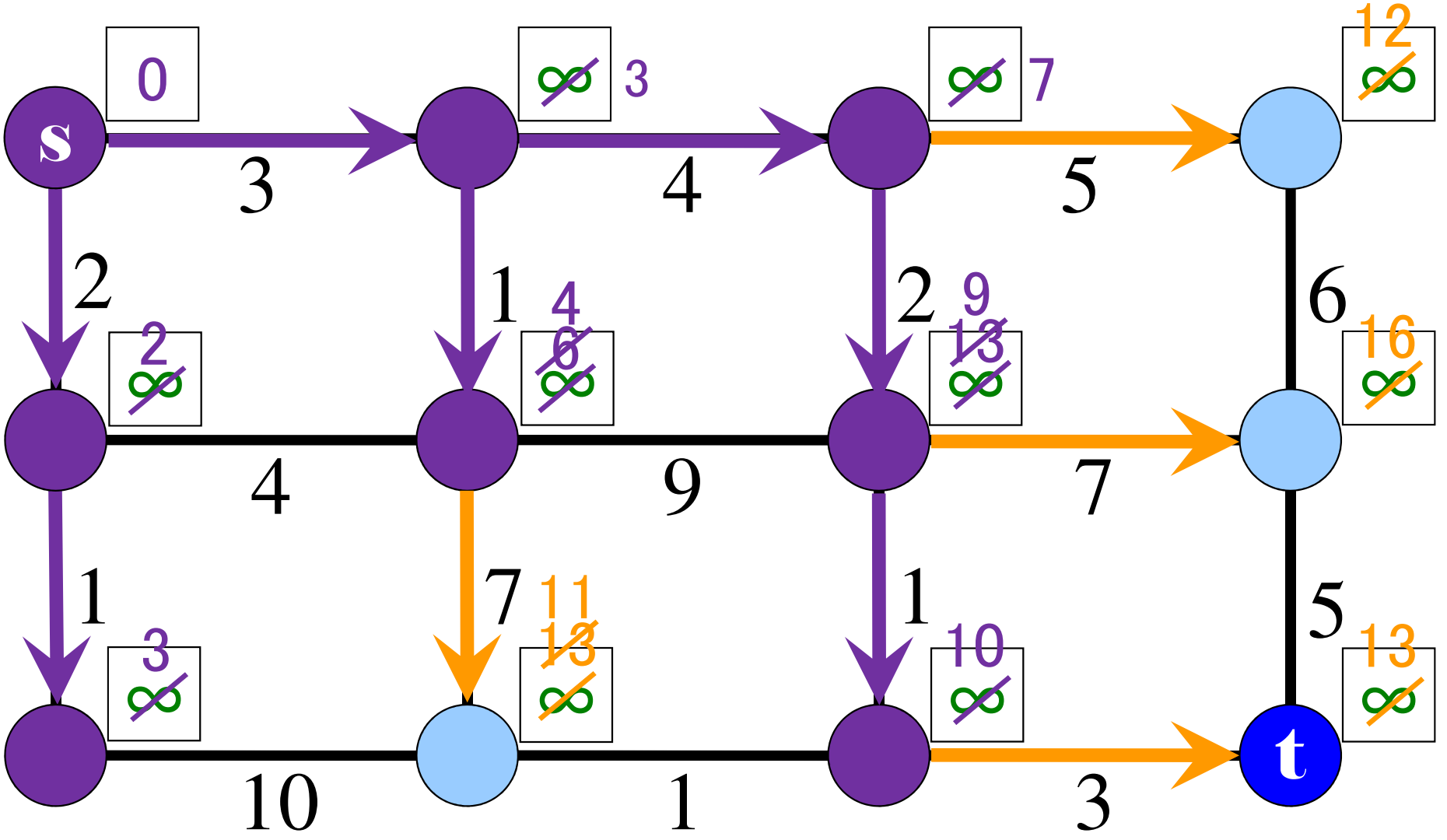


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

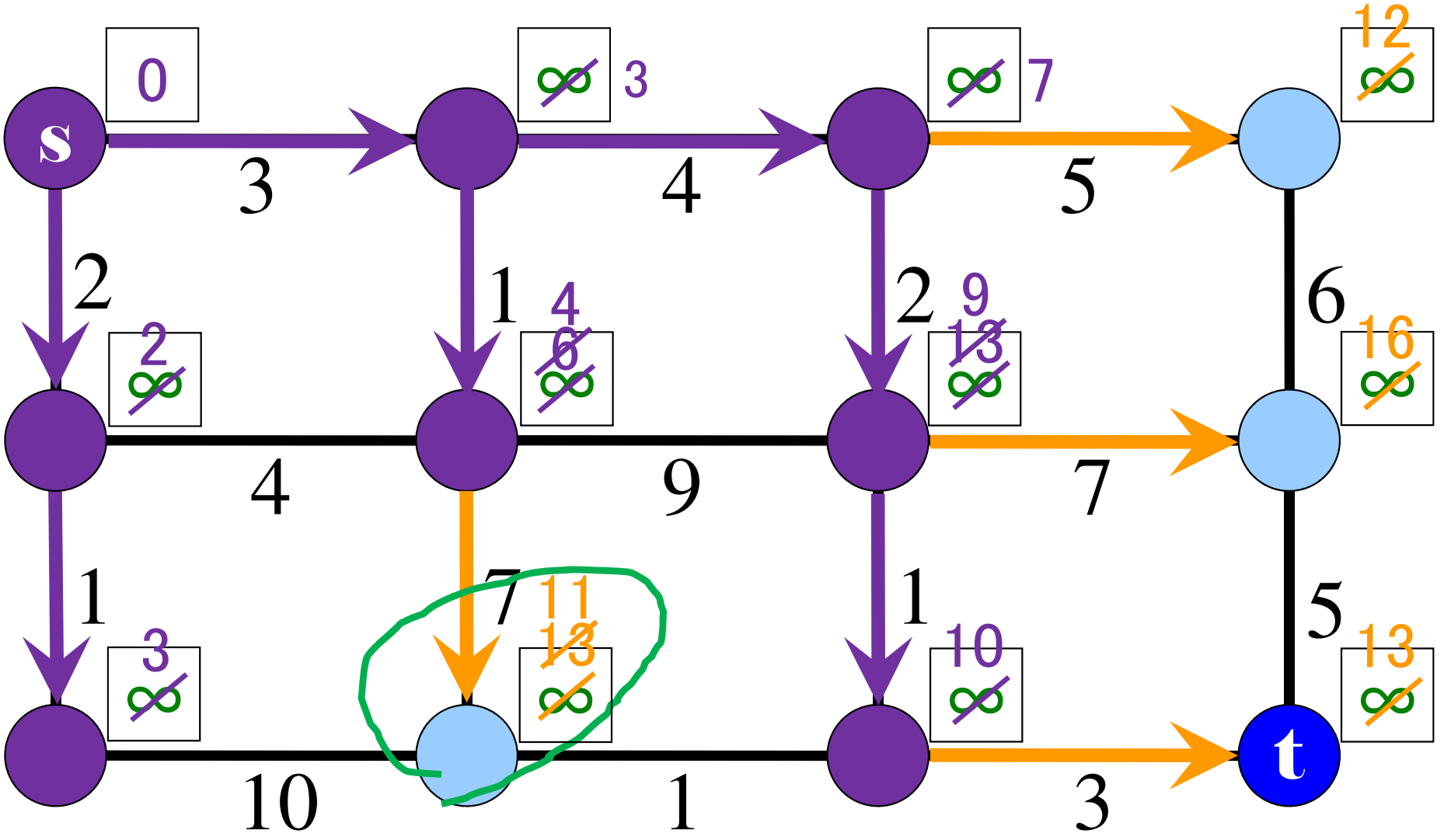
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



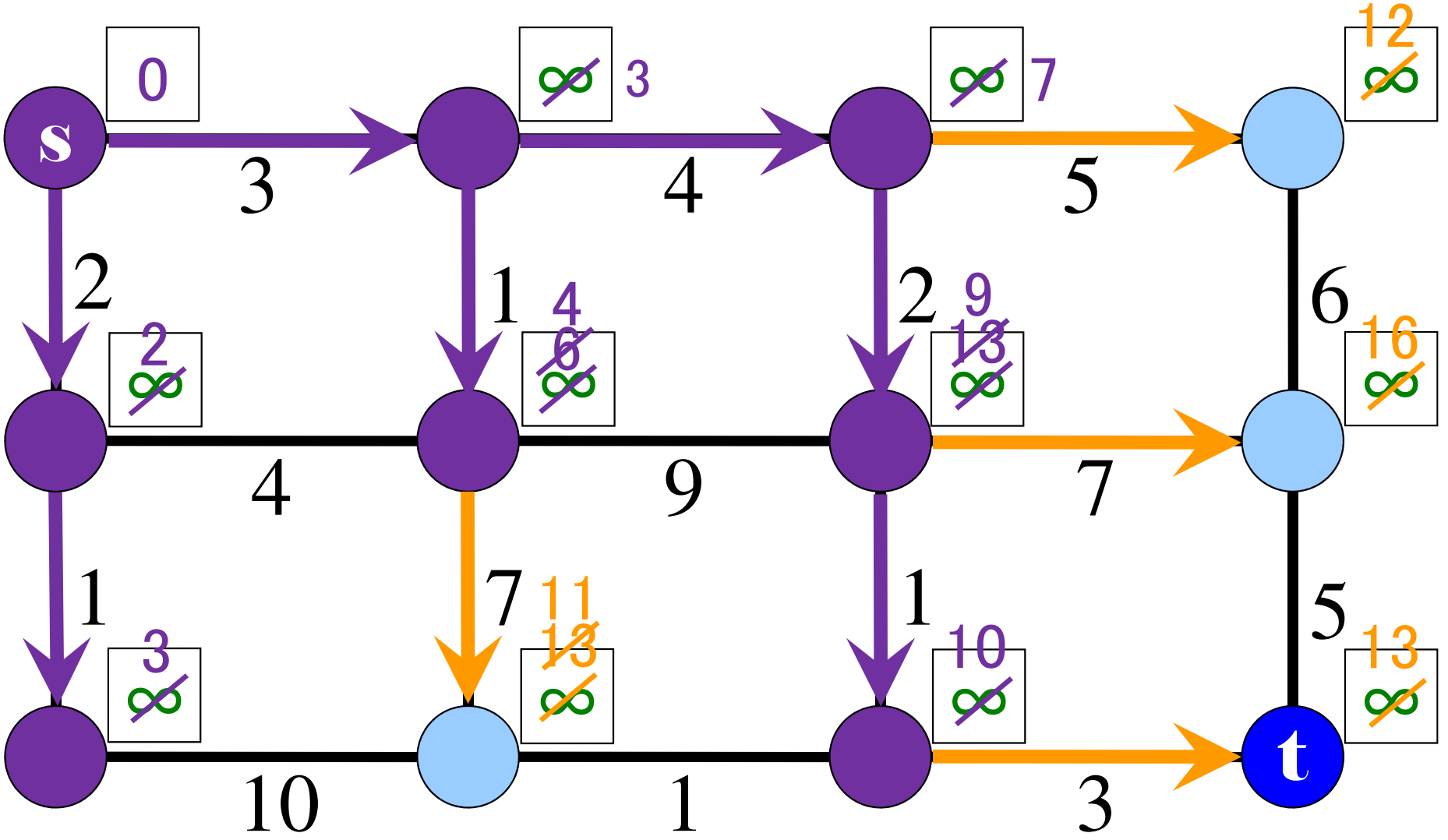
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

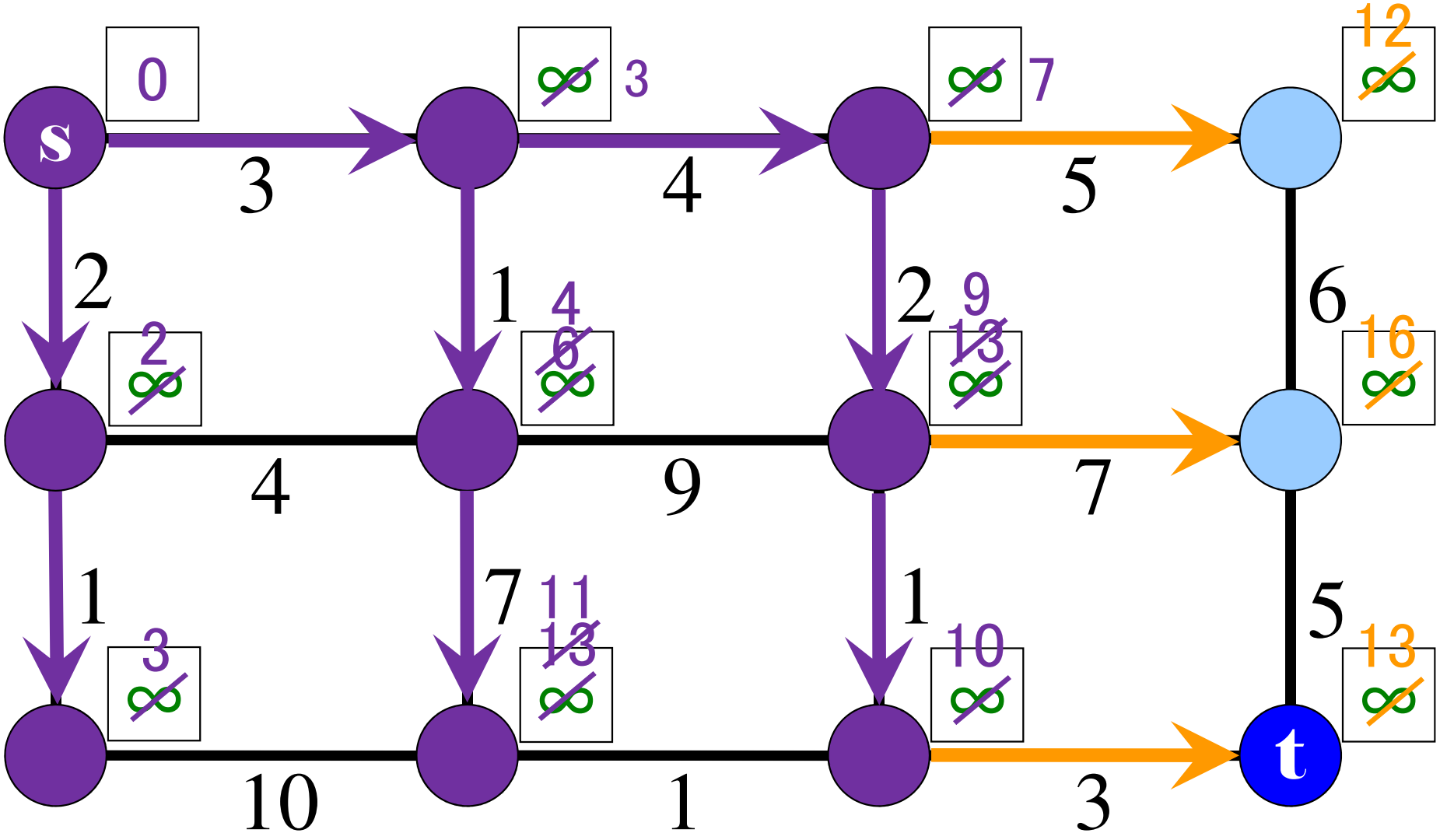


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

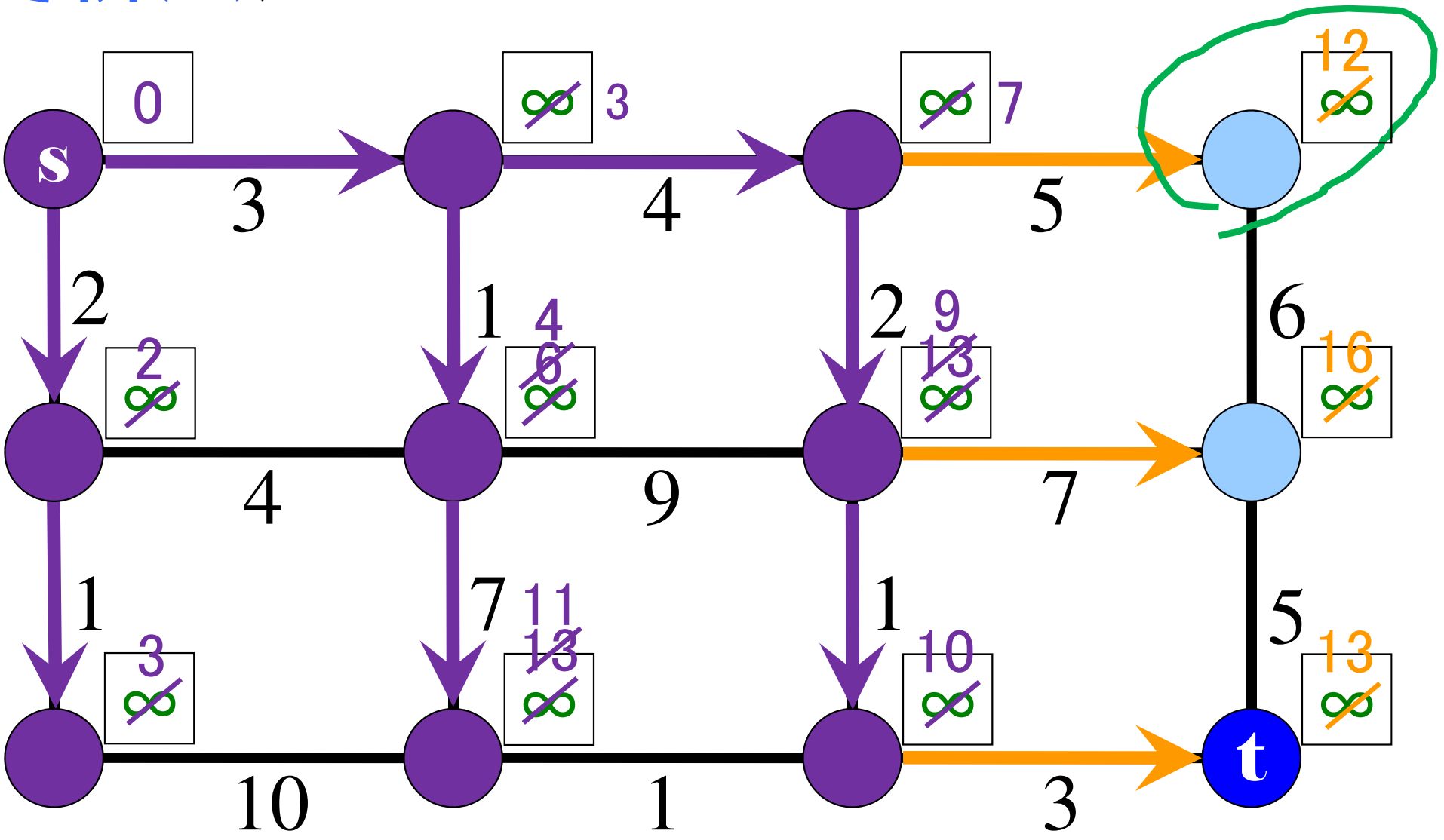
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



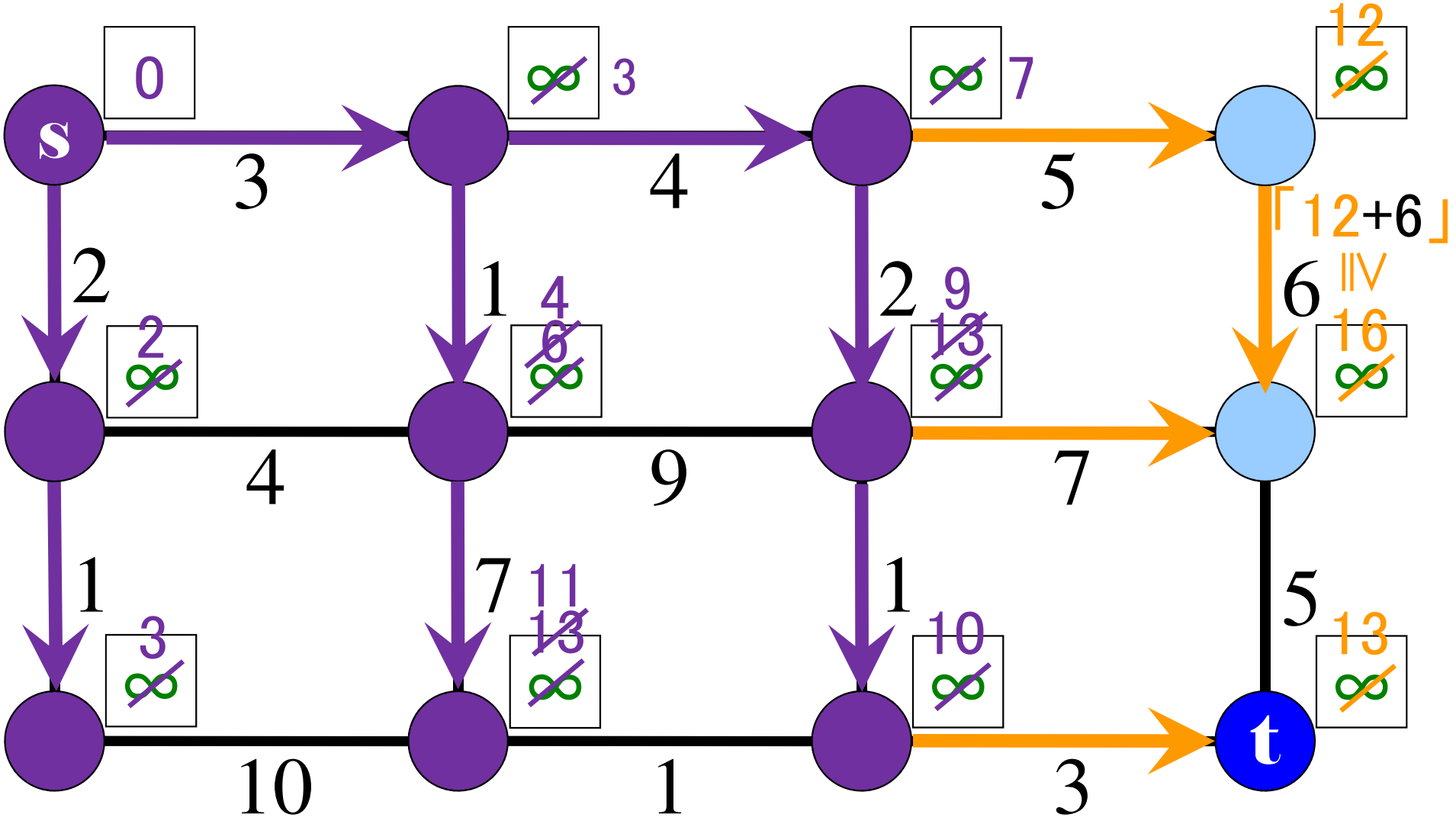
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

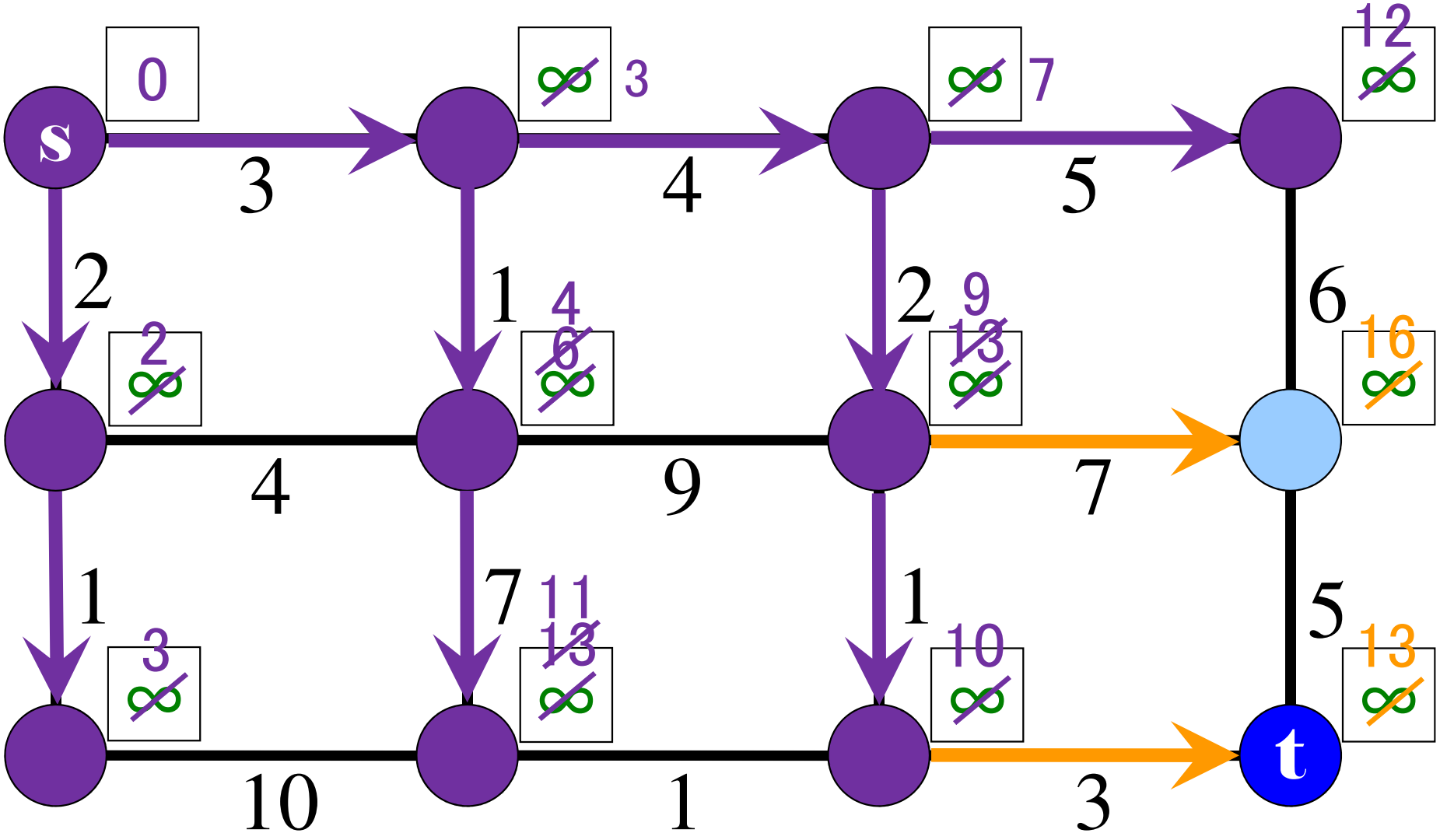


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

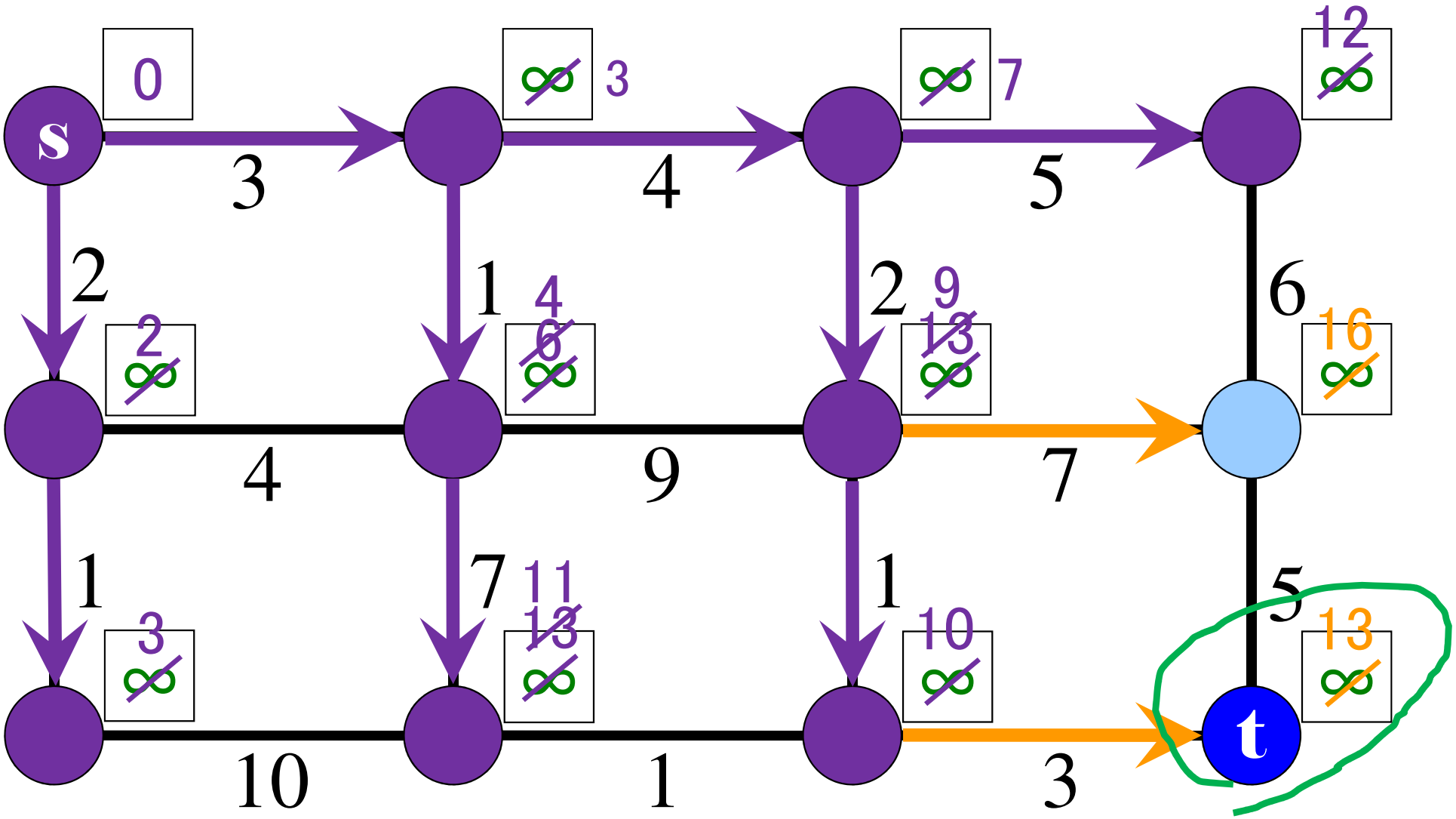
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



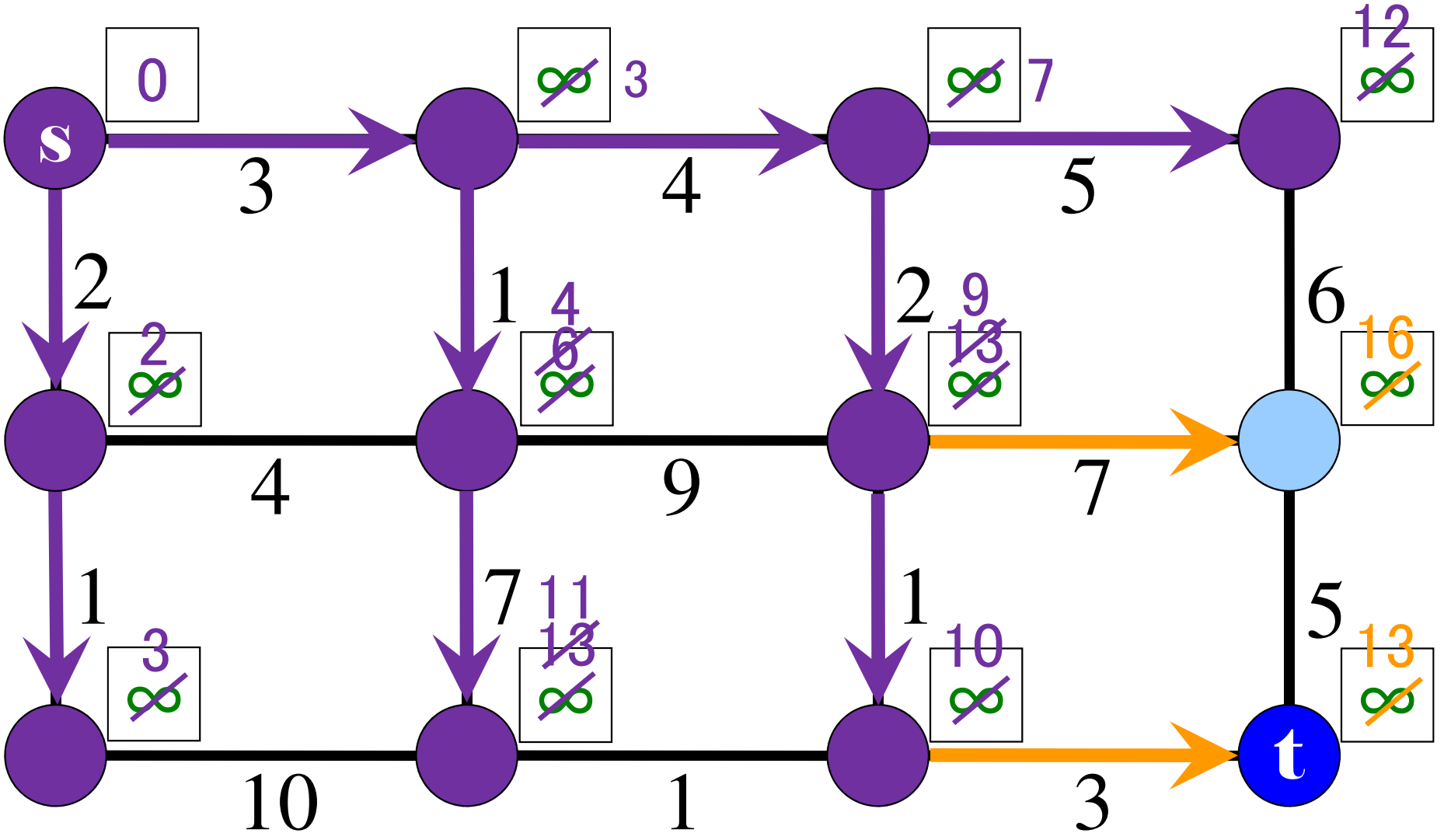
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

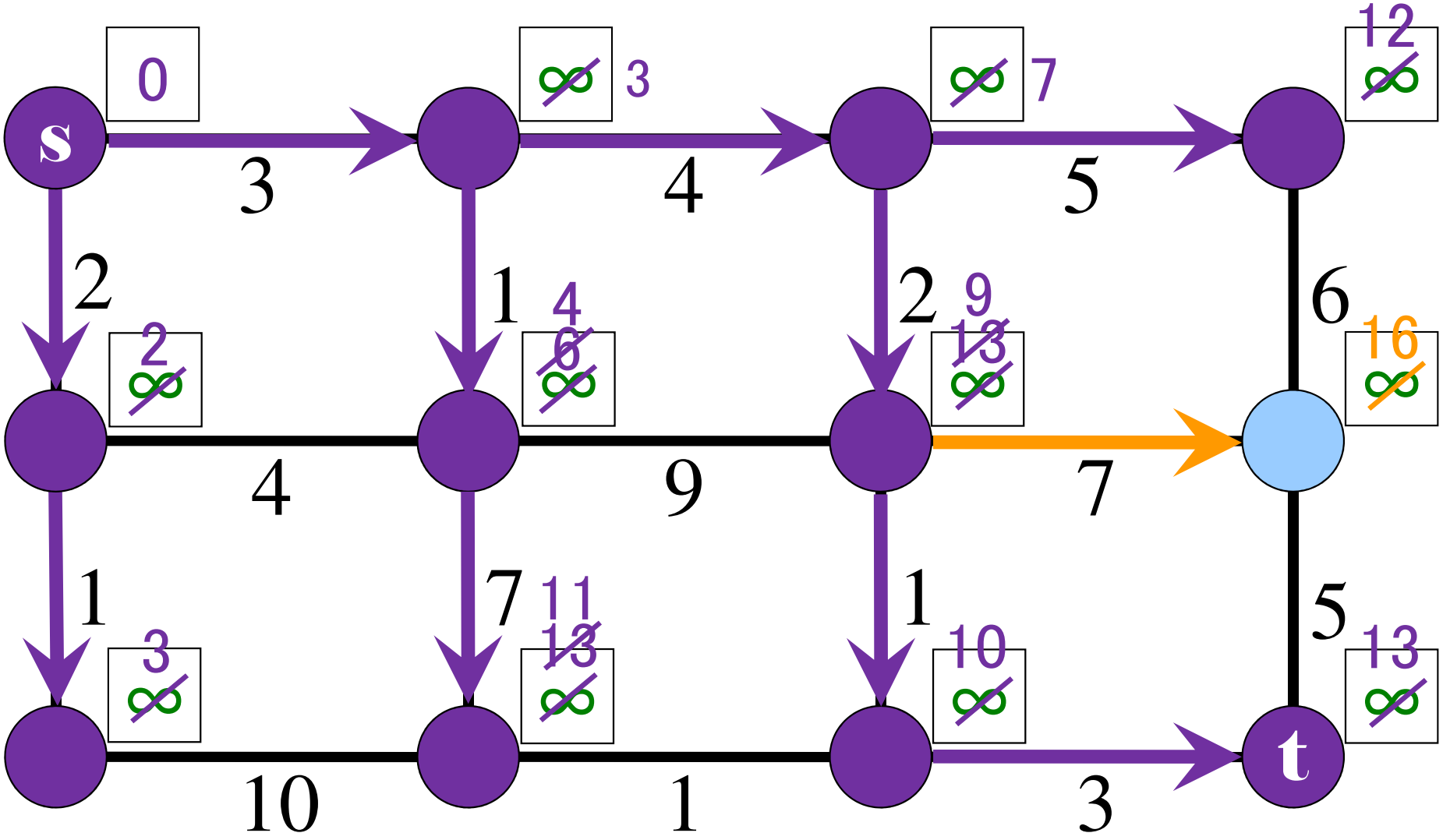


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

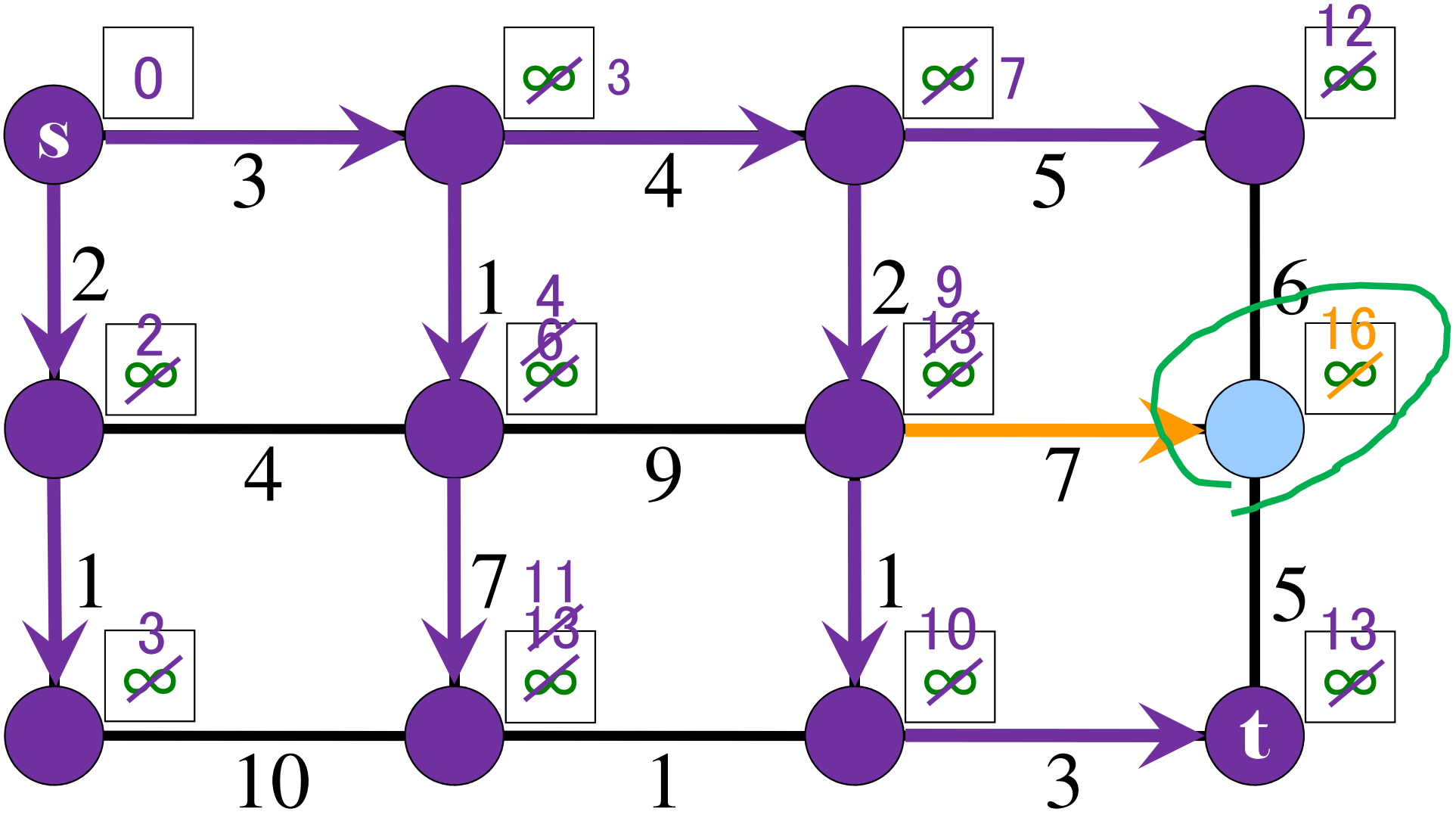
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



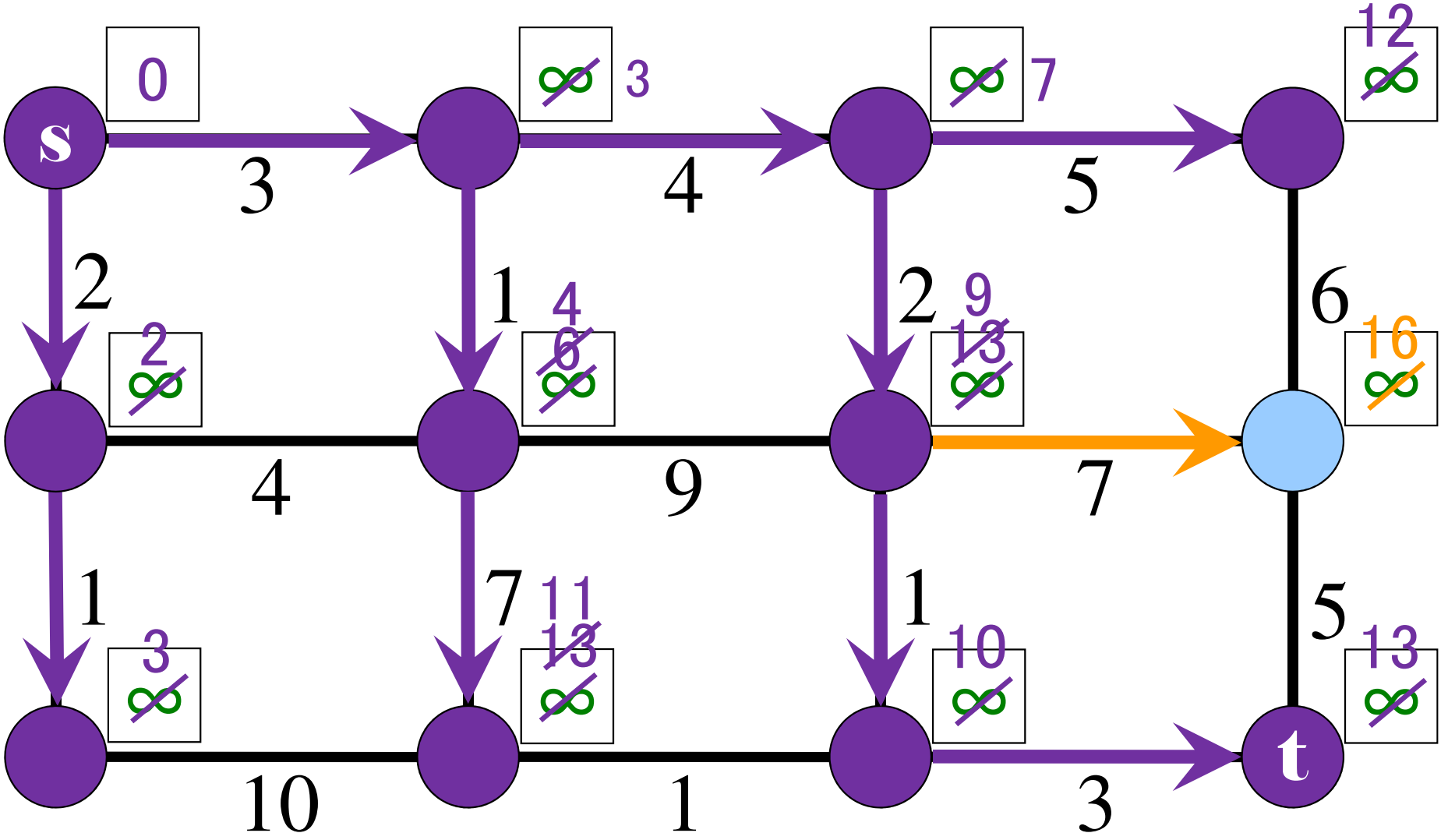
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

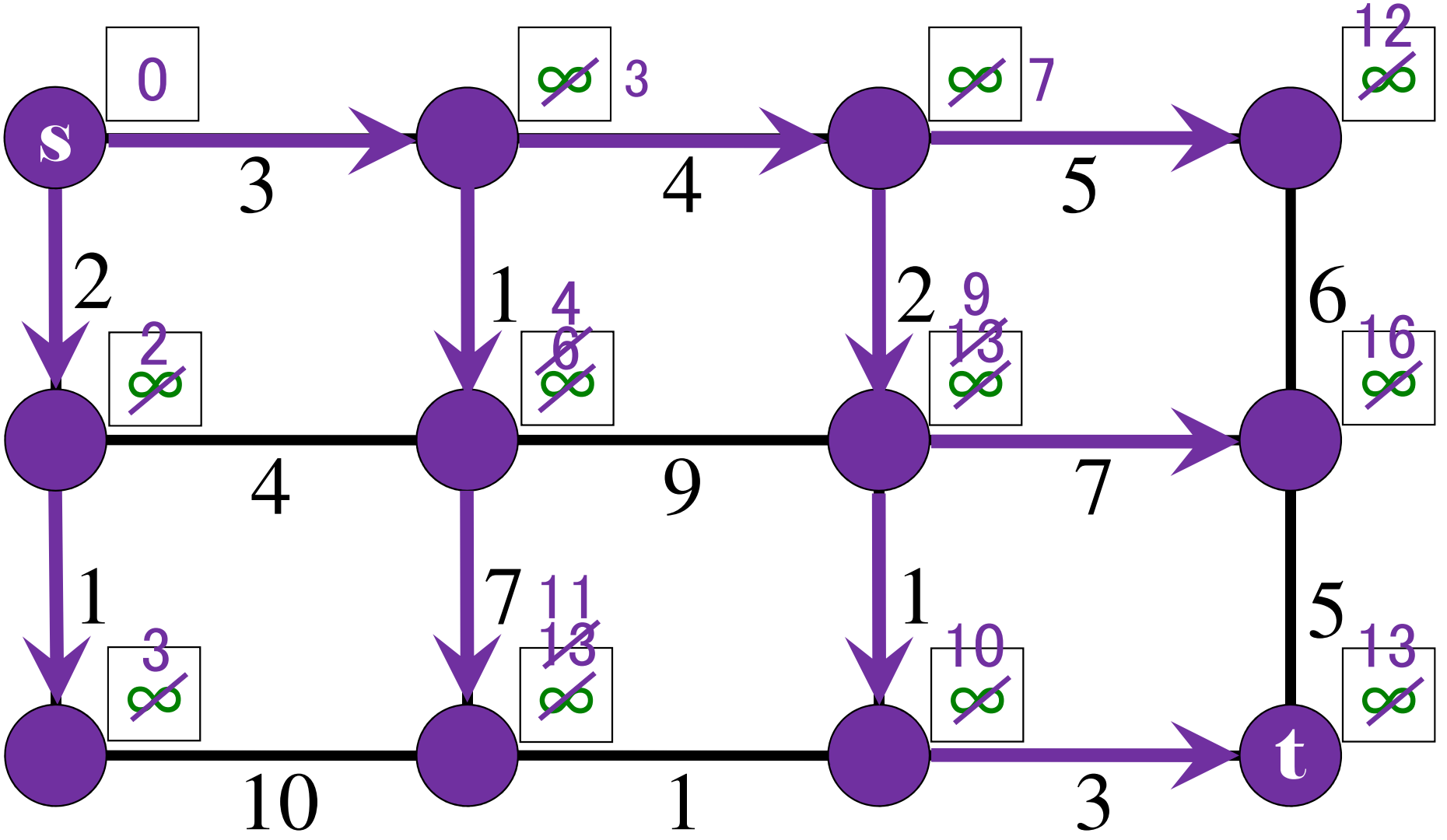


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

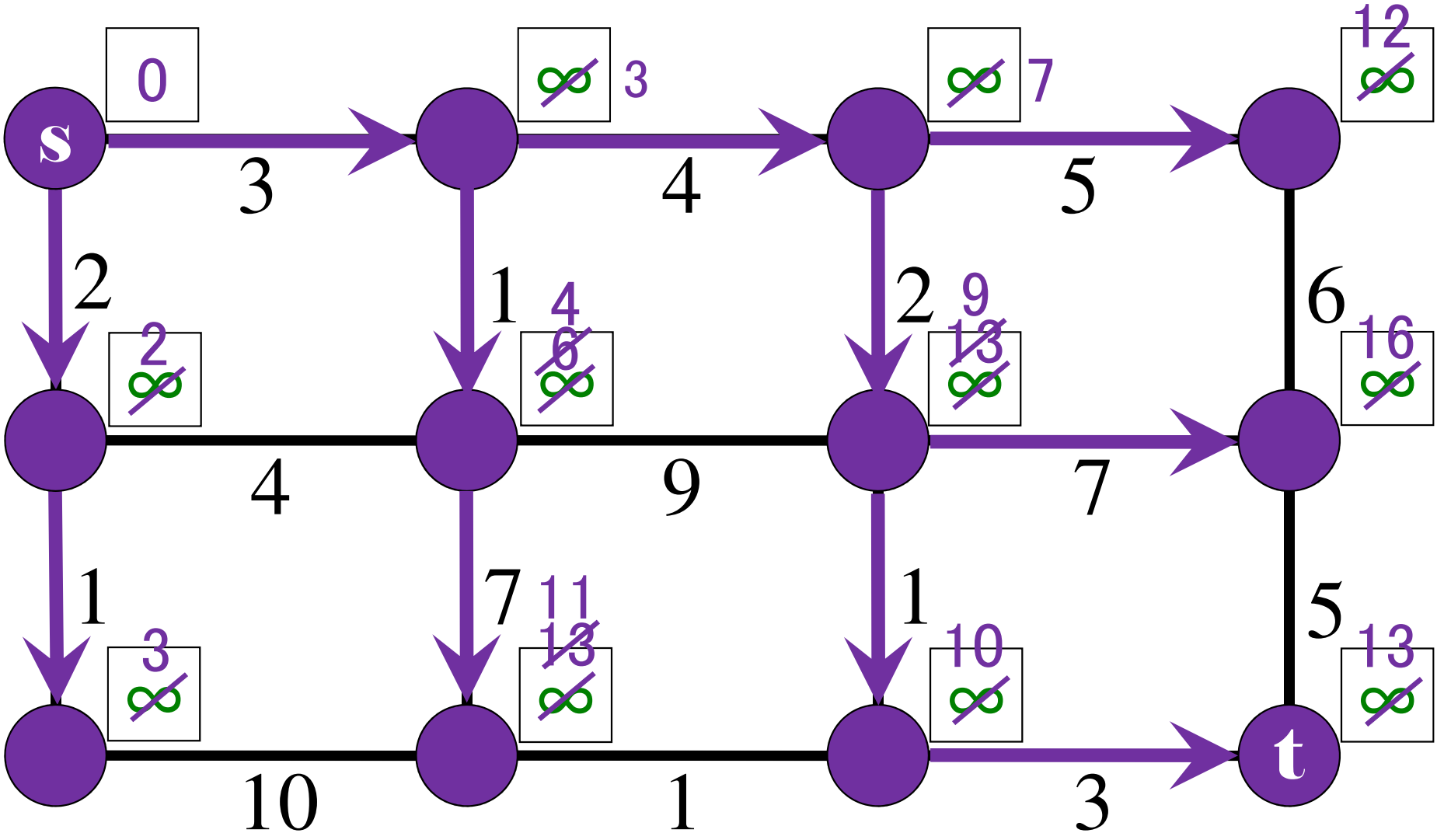
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



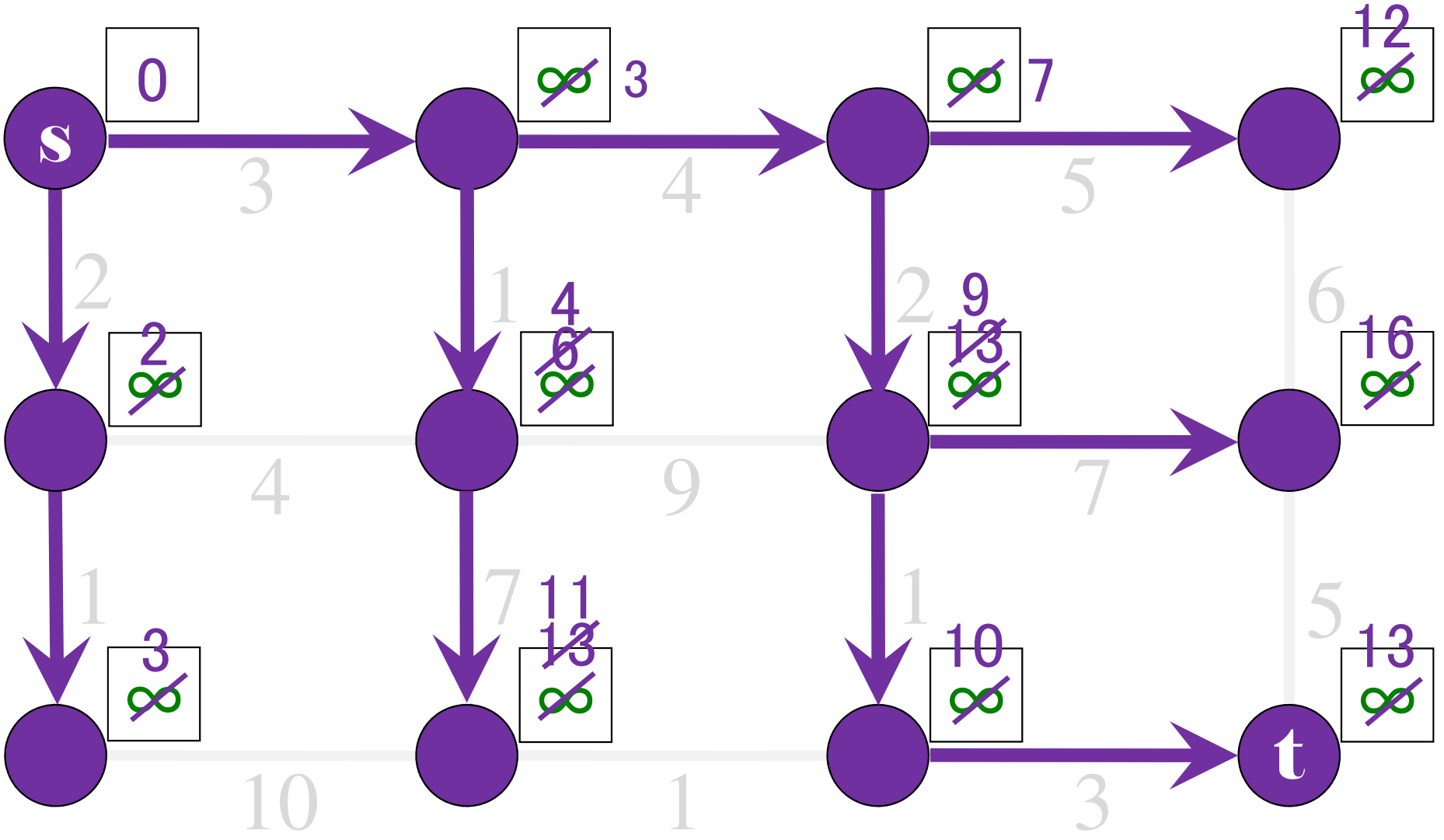
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



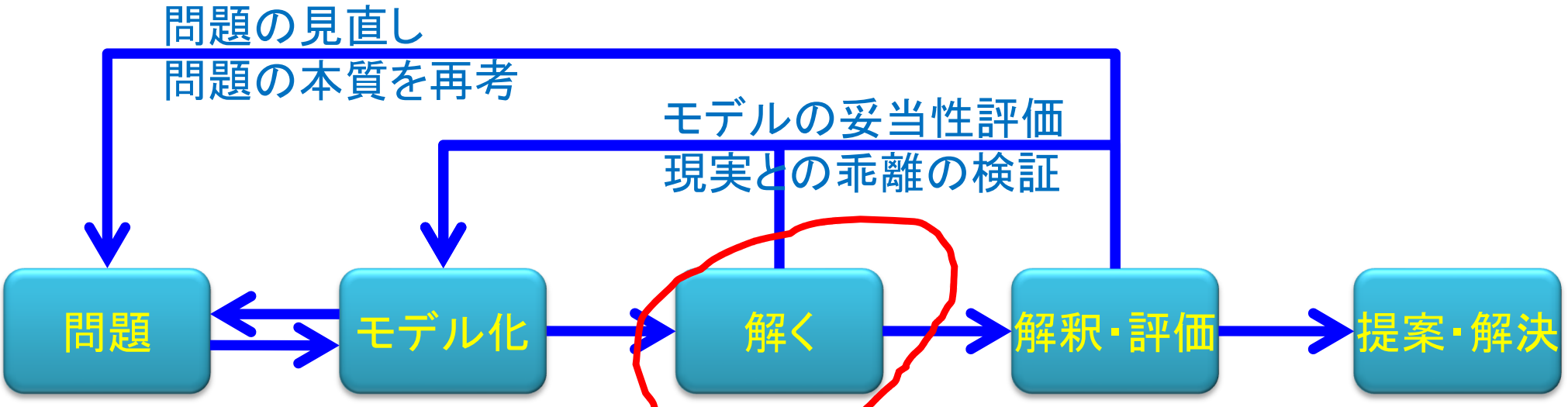
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

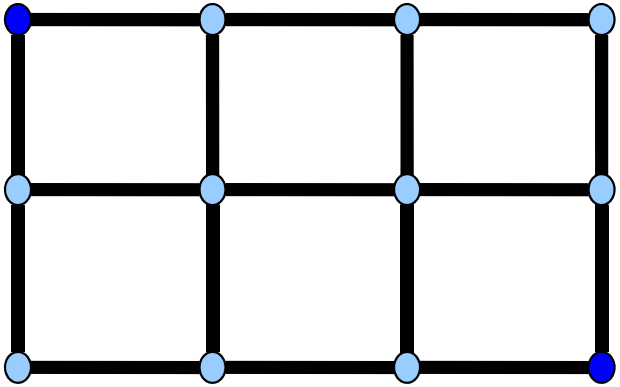
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

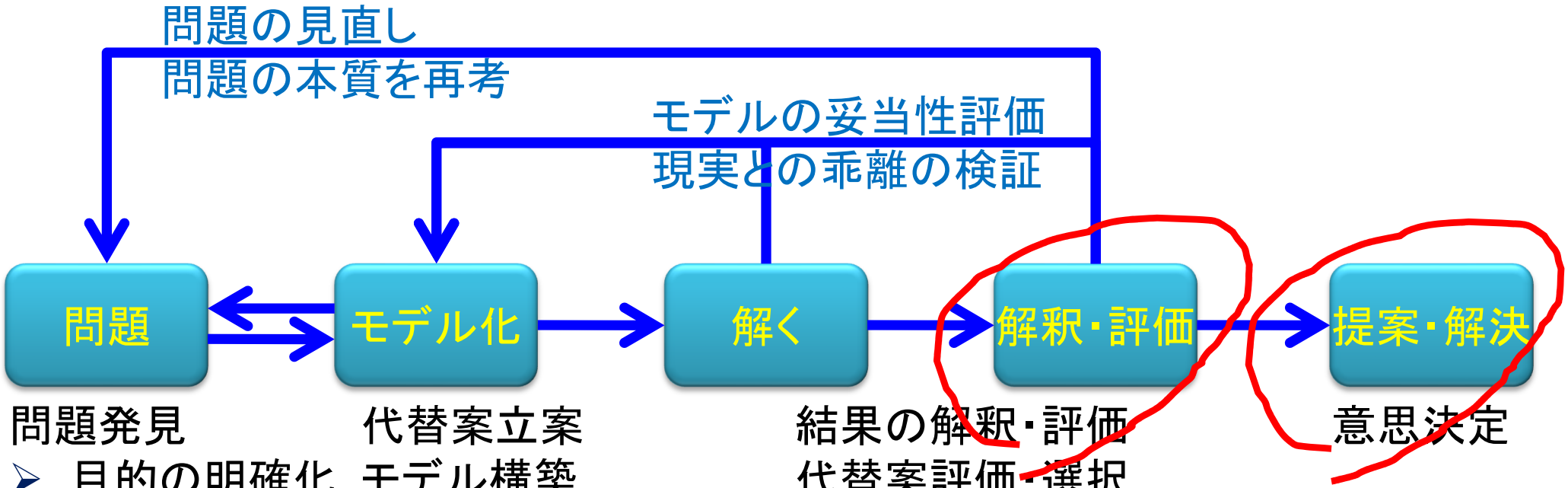
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

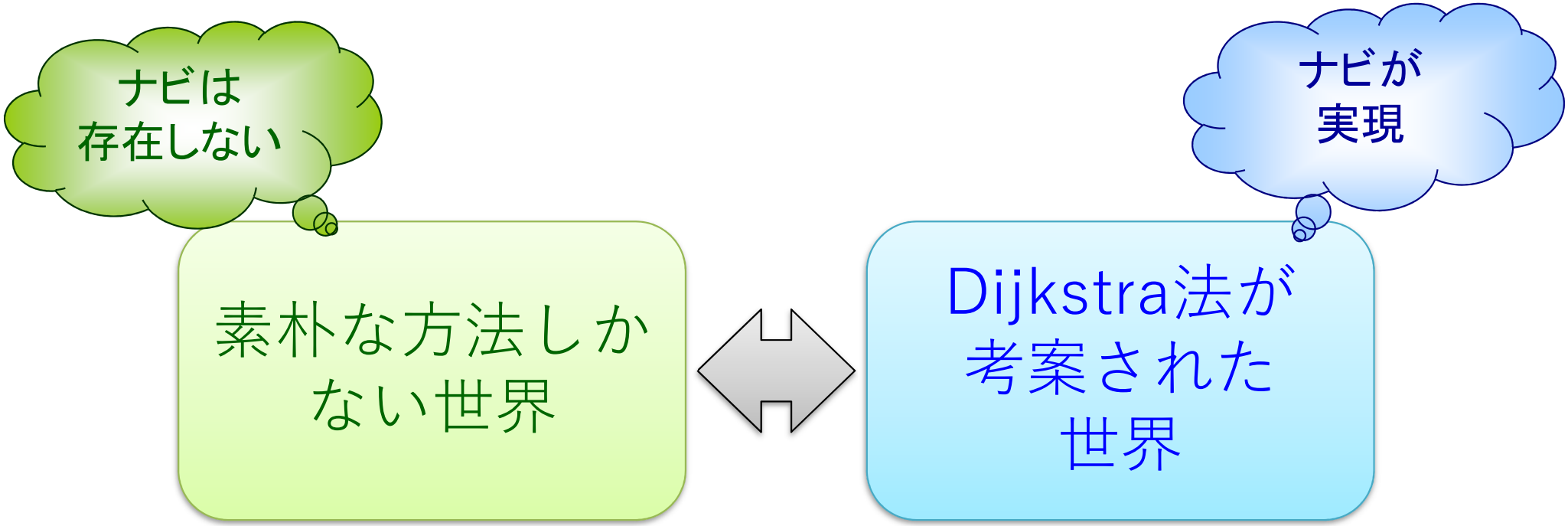
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

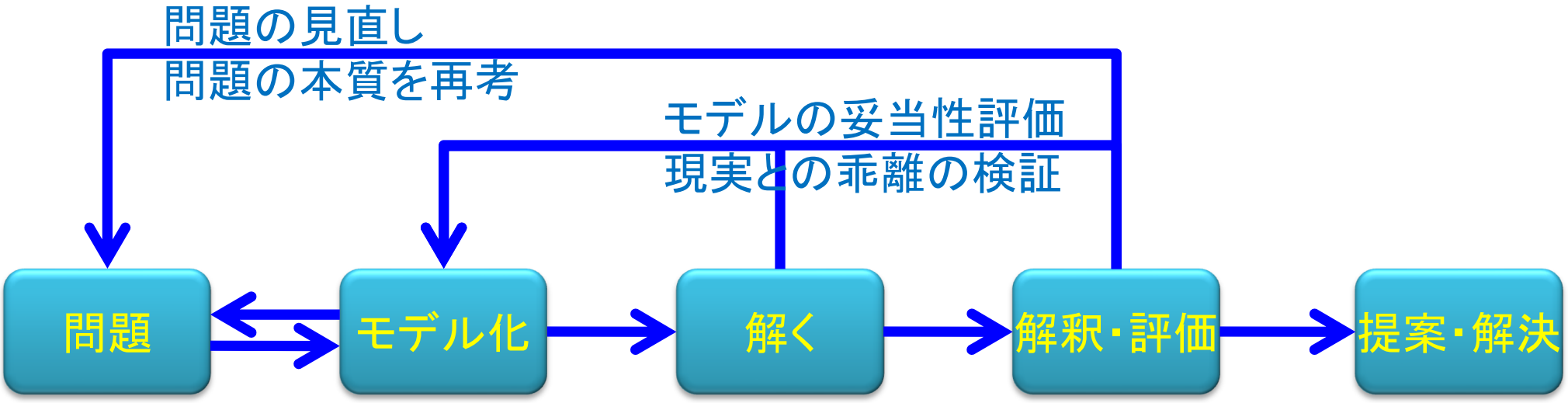
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



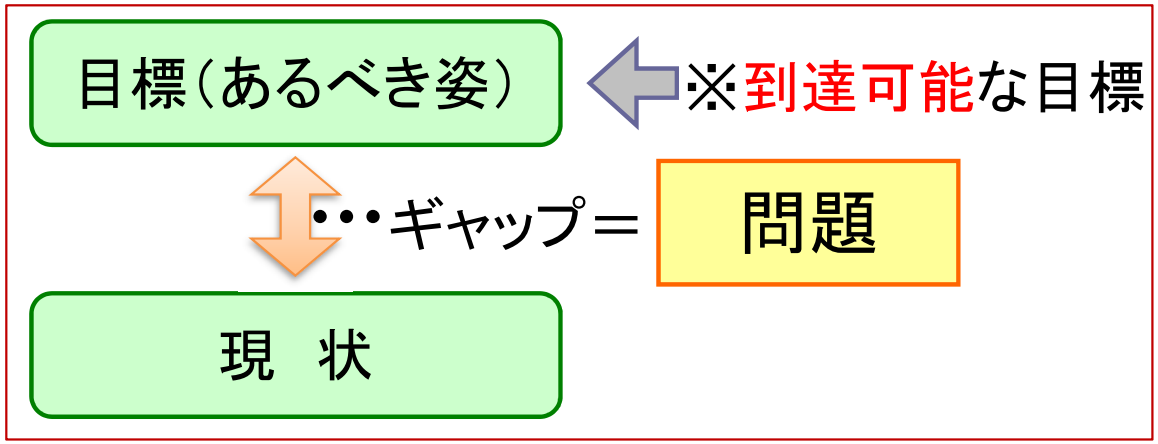
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

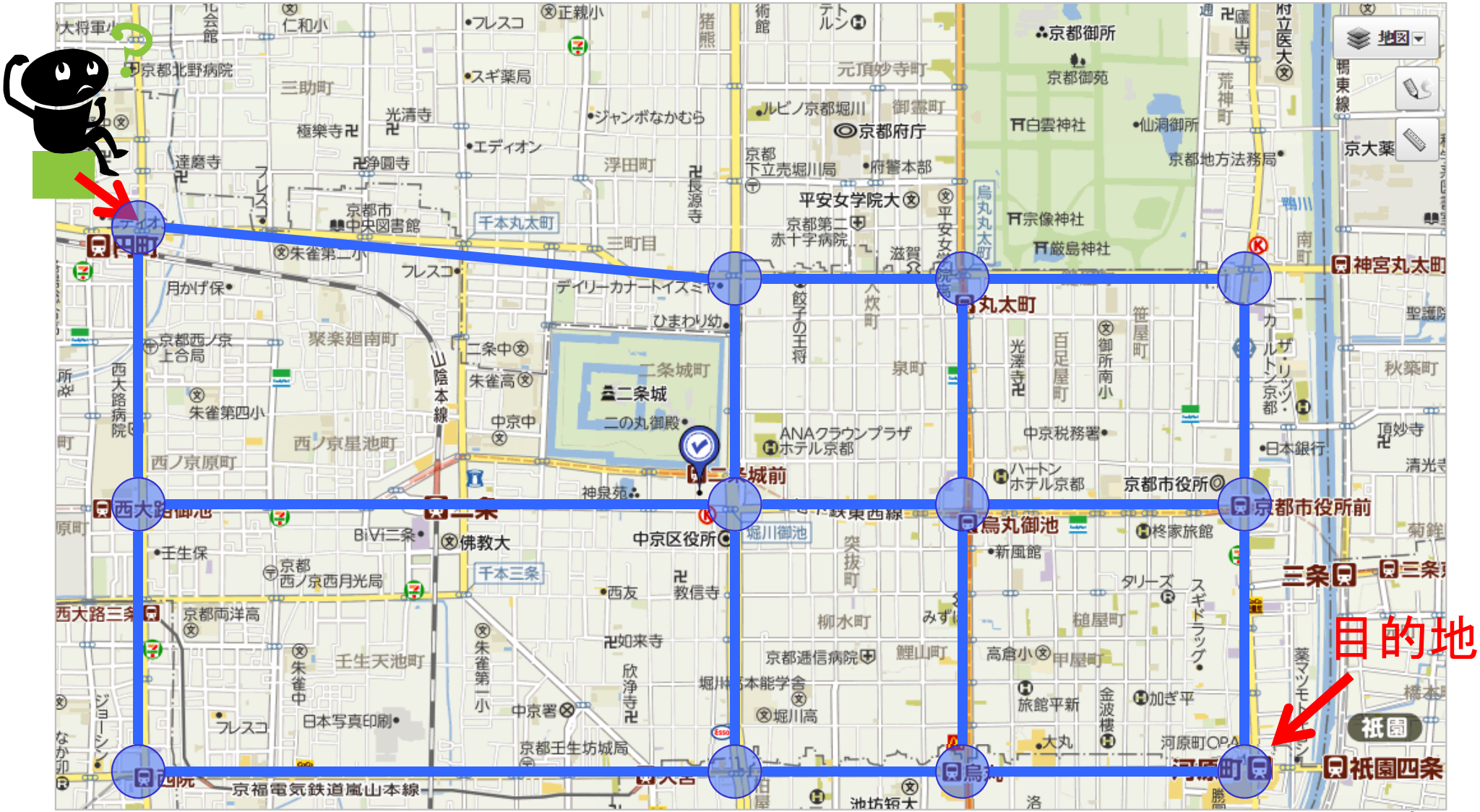
意思決定

問題の定義



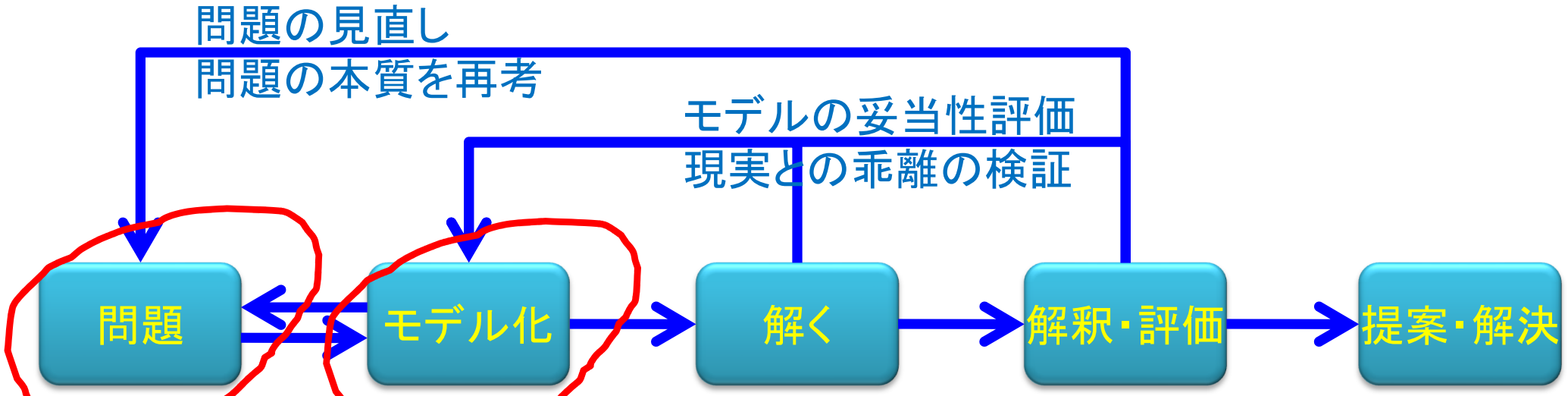
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

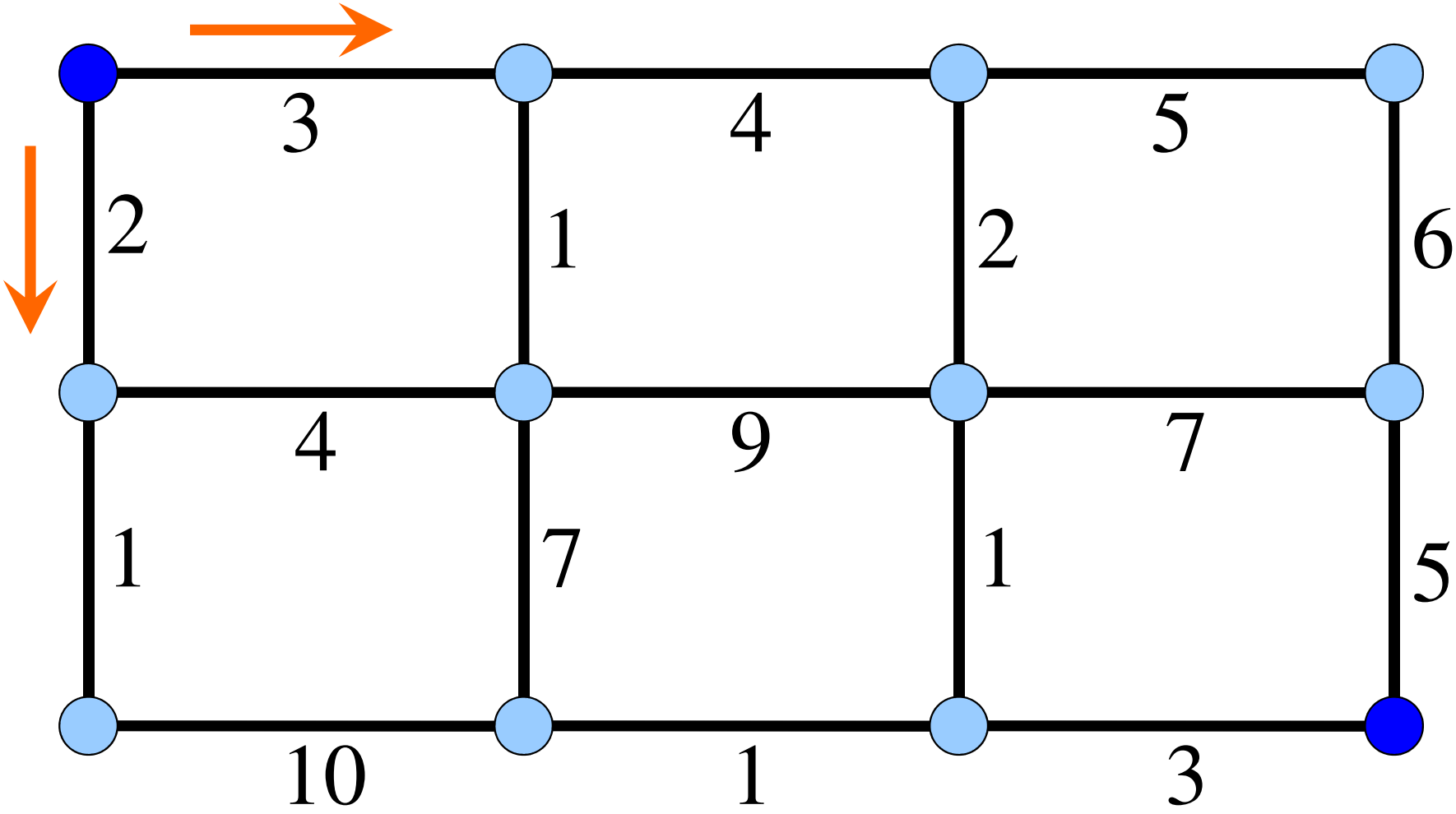
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

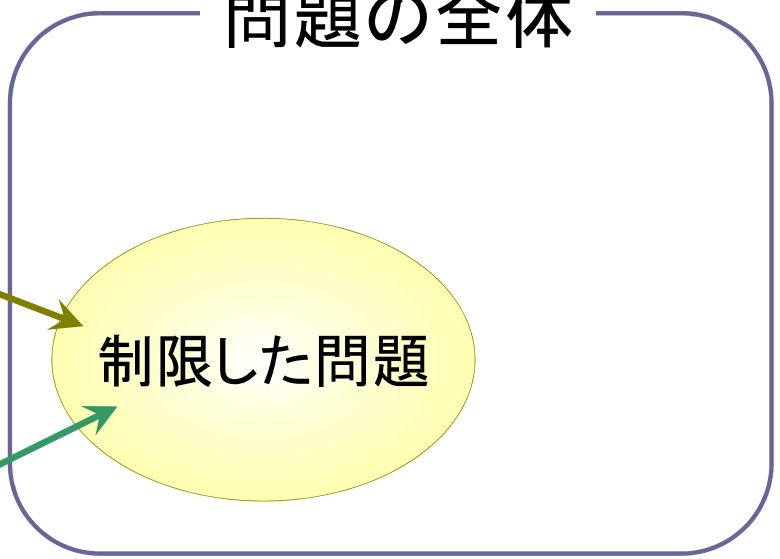


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

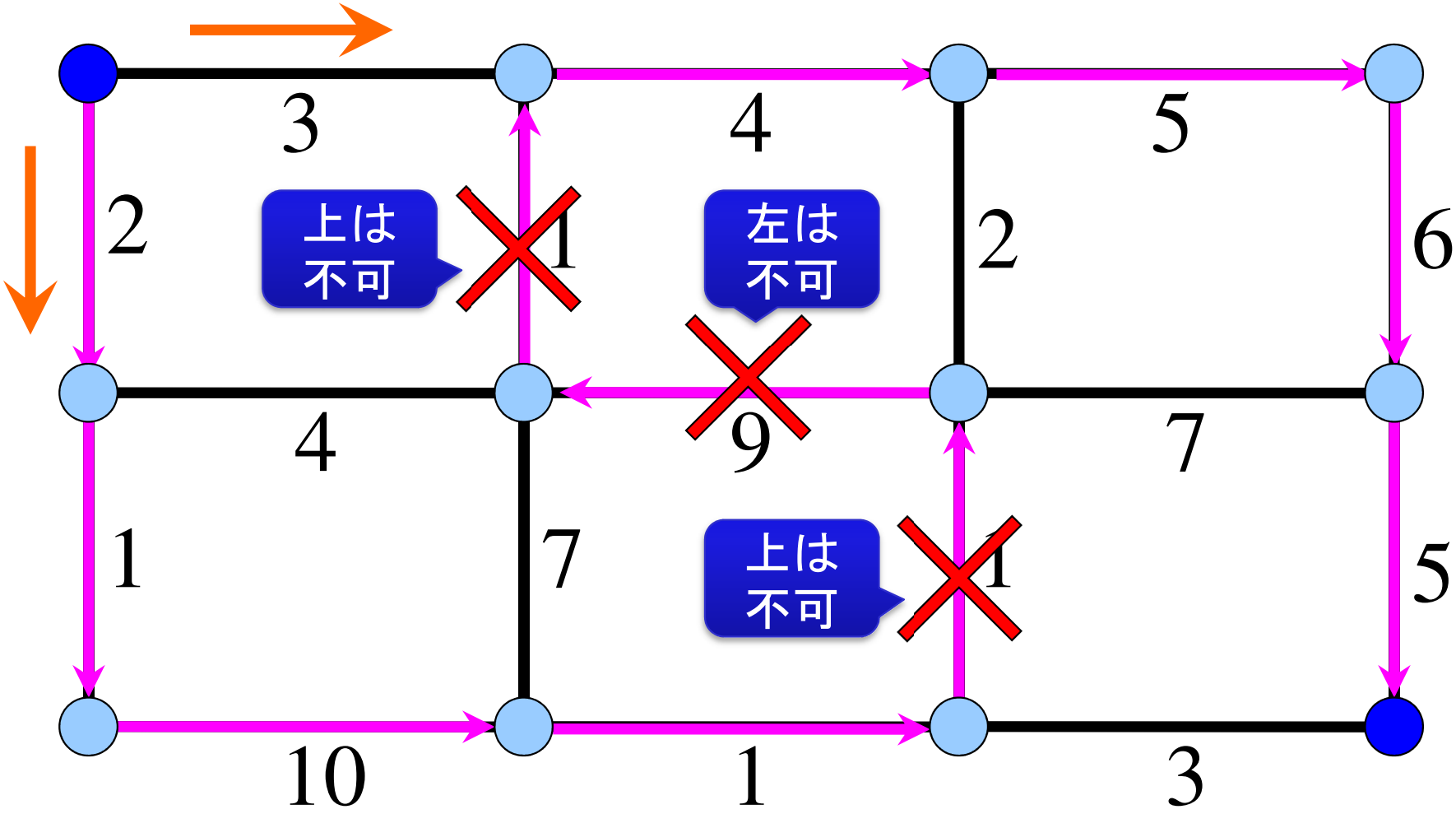
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

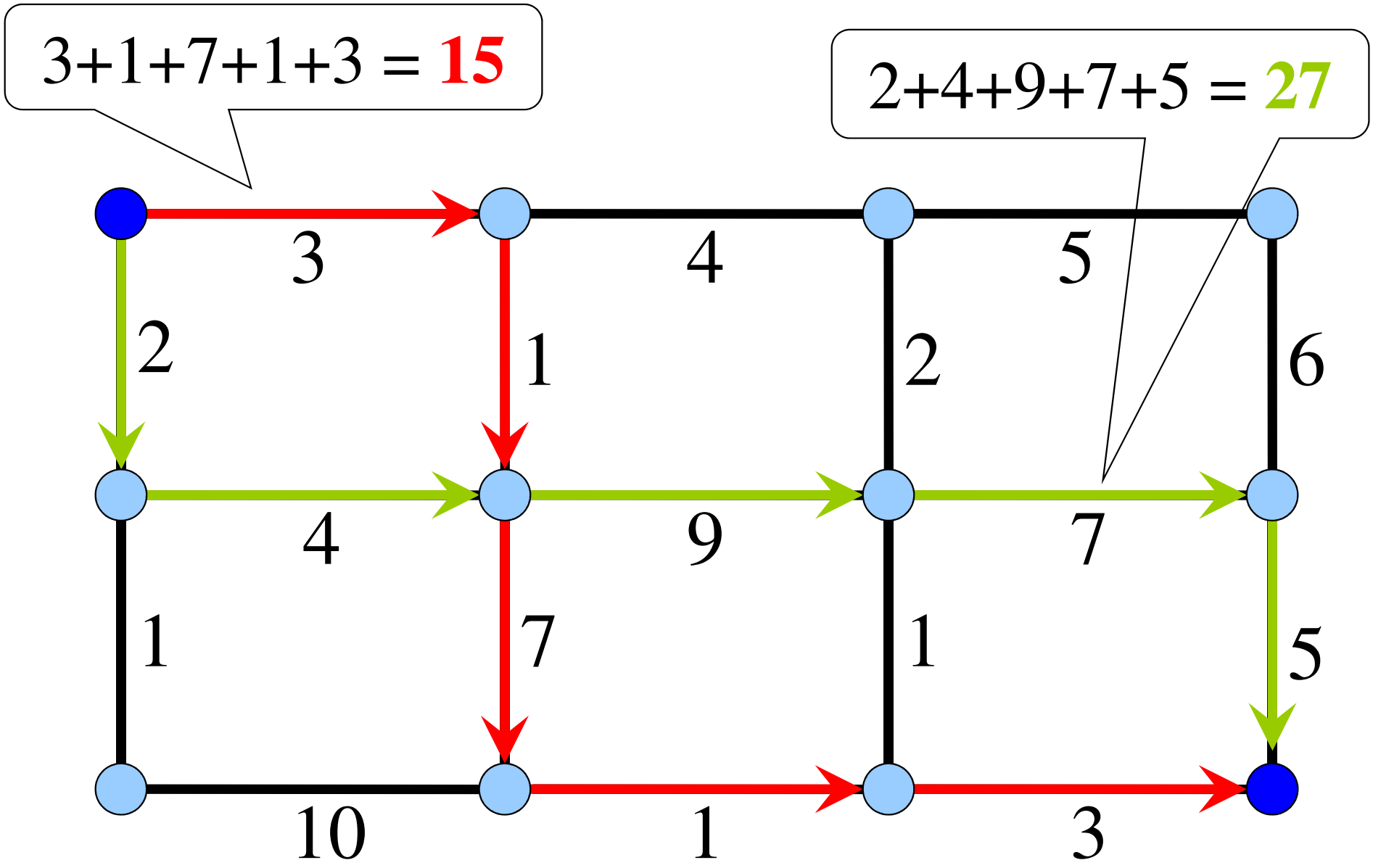
難しいなら易しくすればいいのさ！

制限した問題

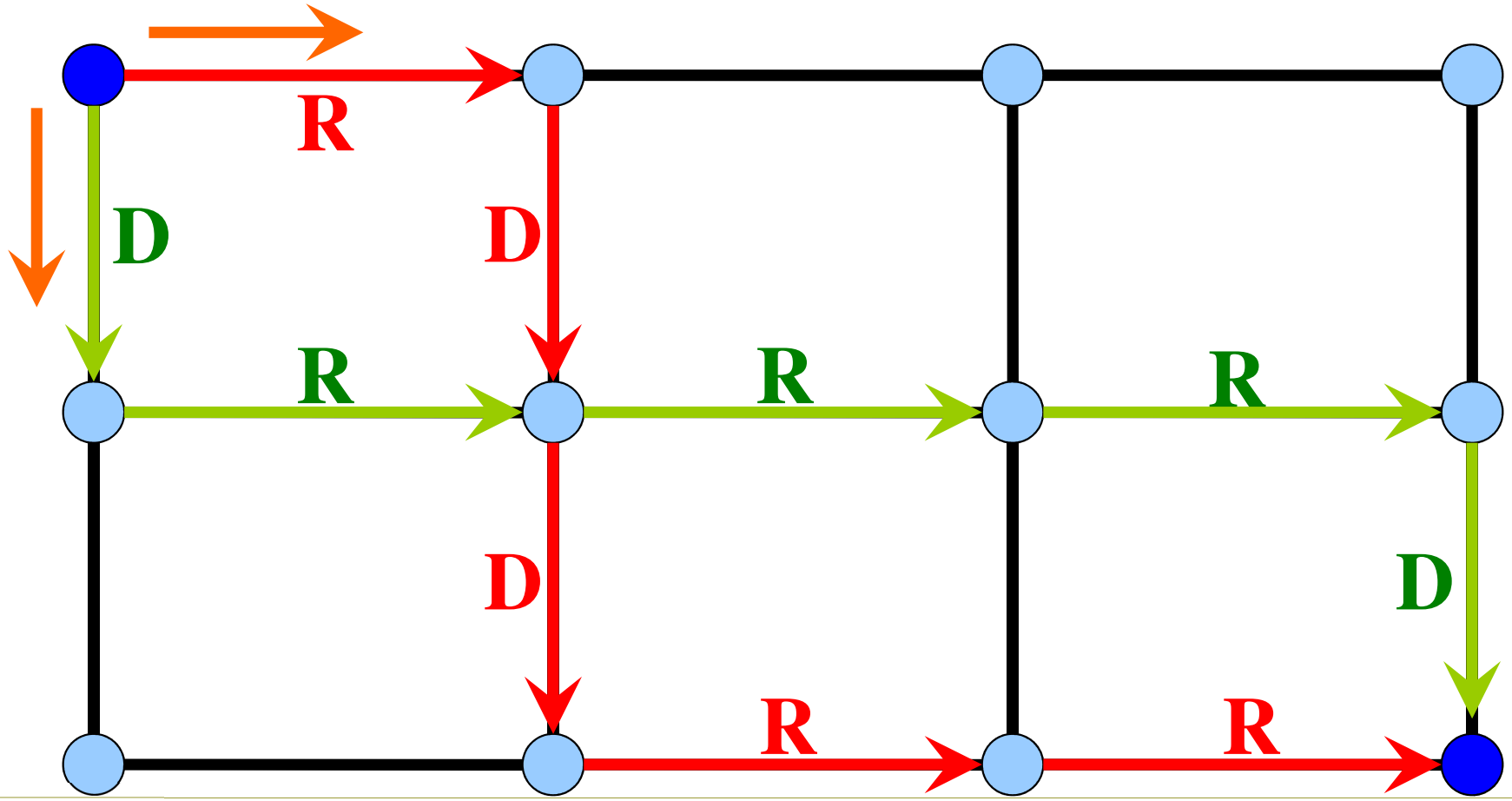
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



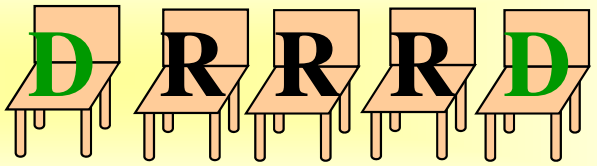
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



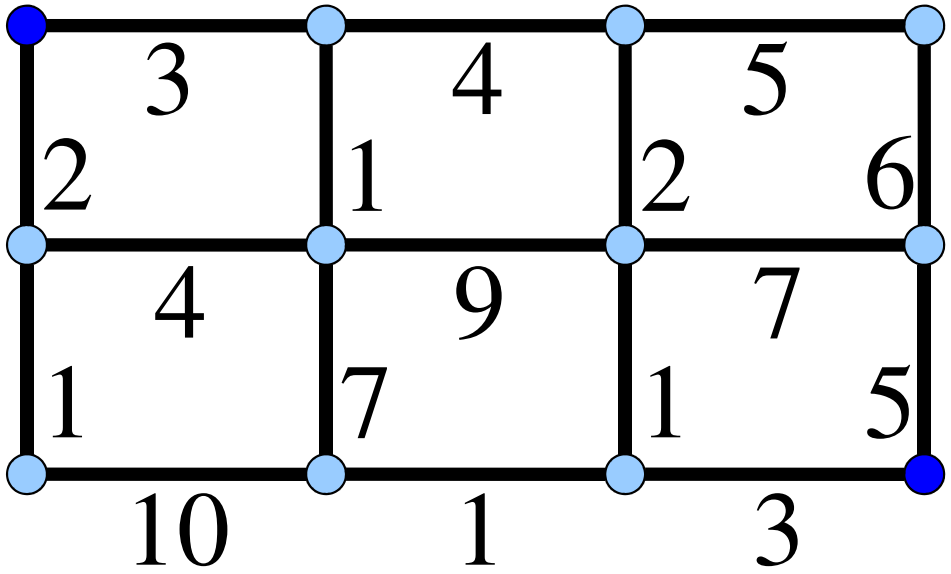
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

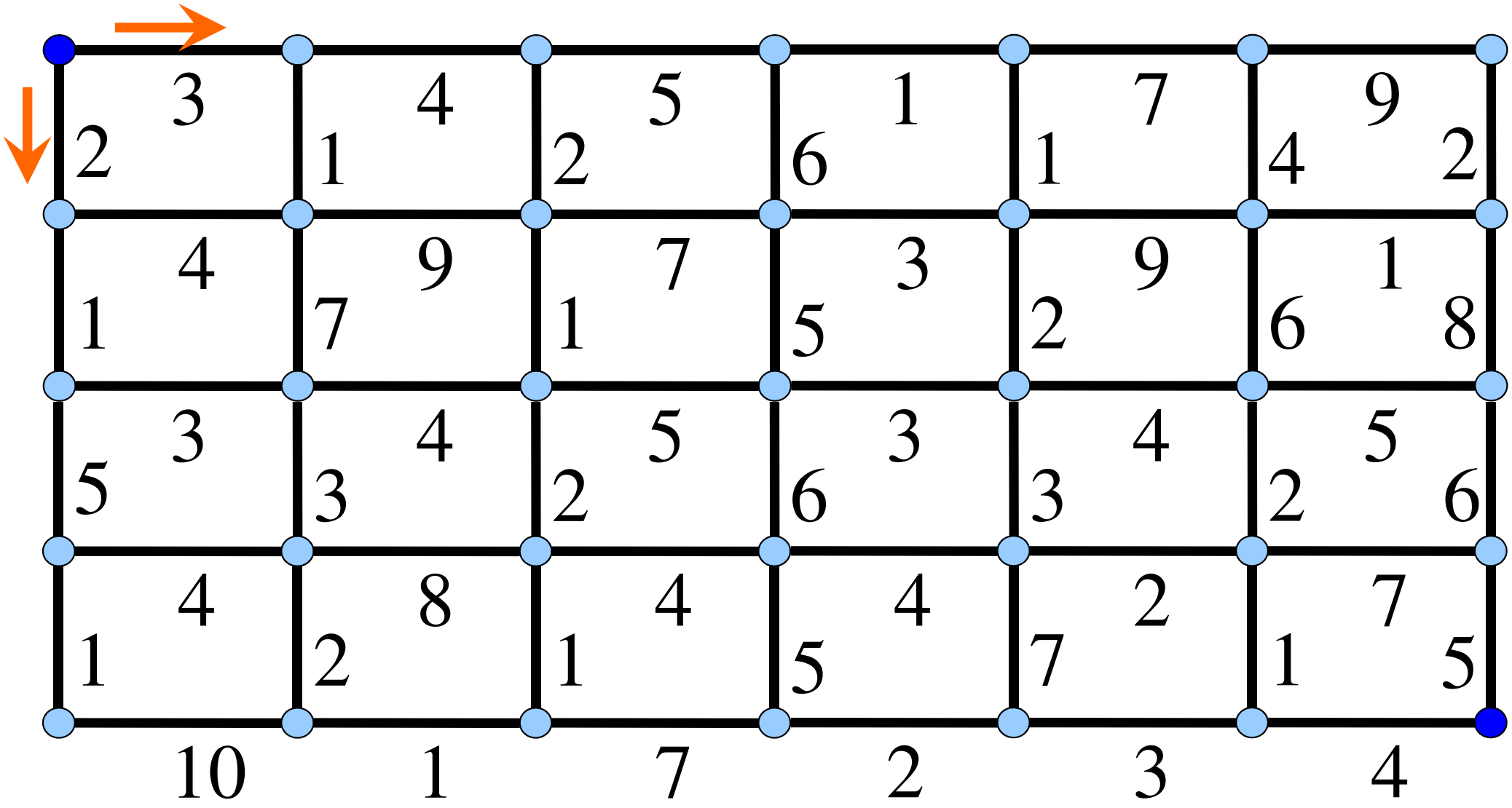
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

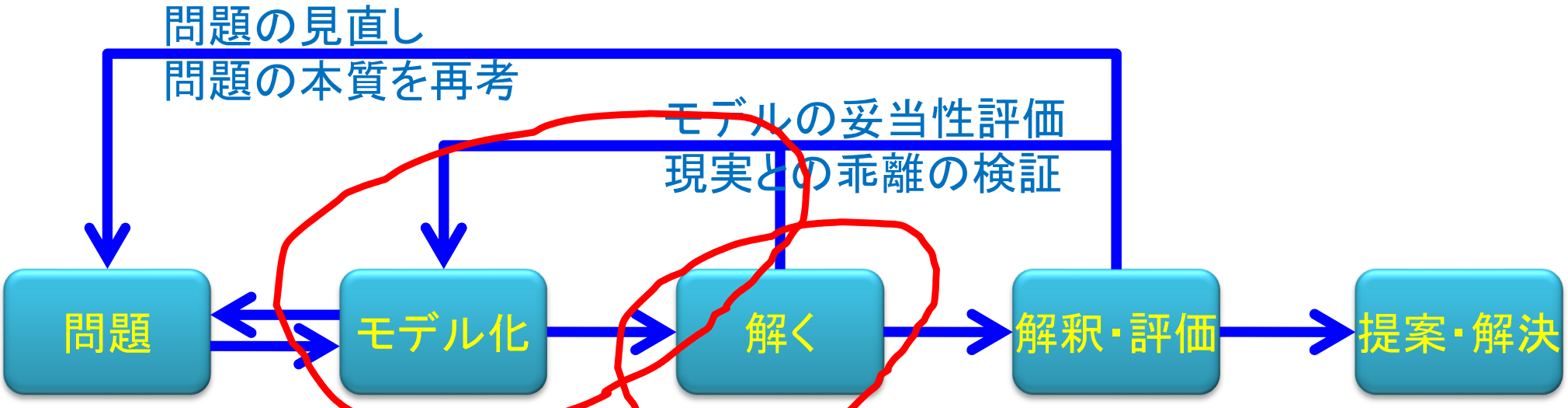
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

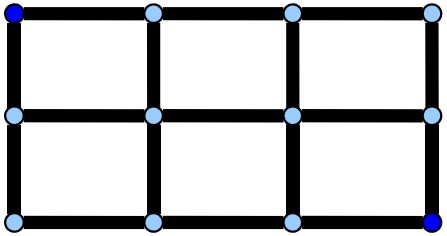
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

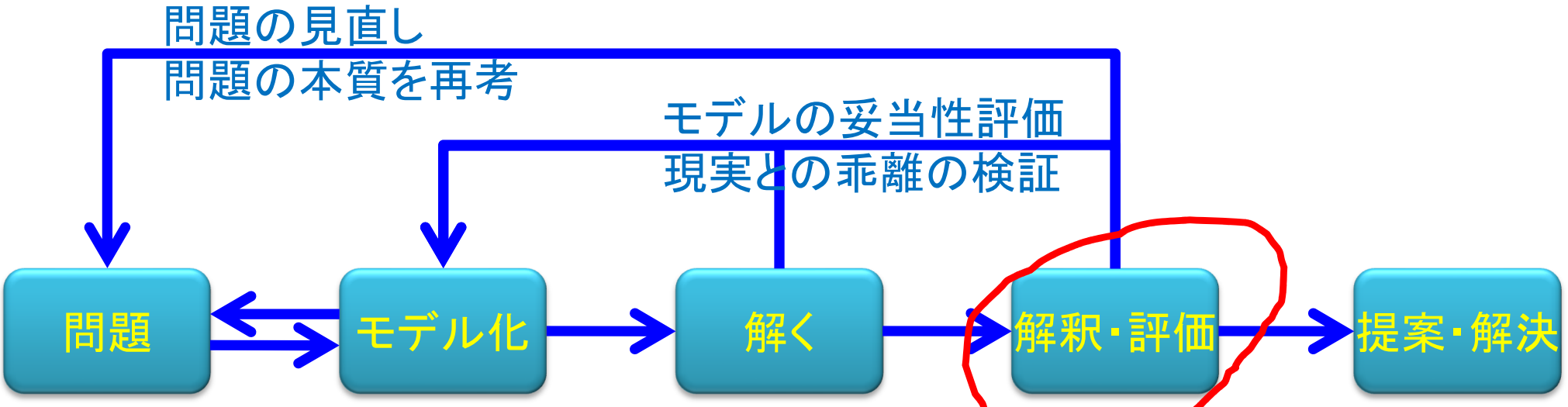
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

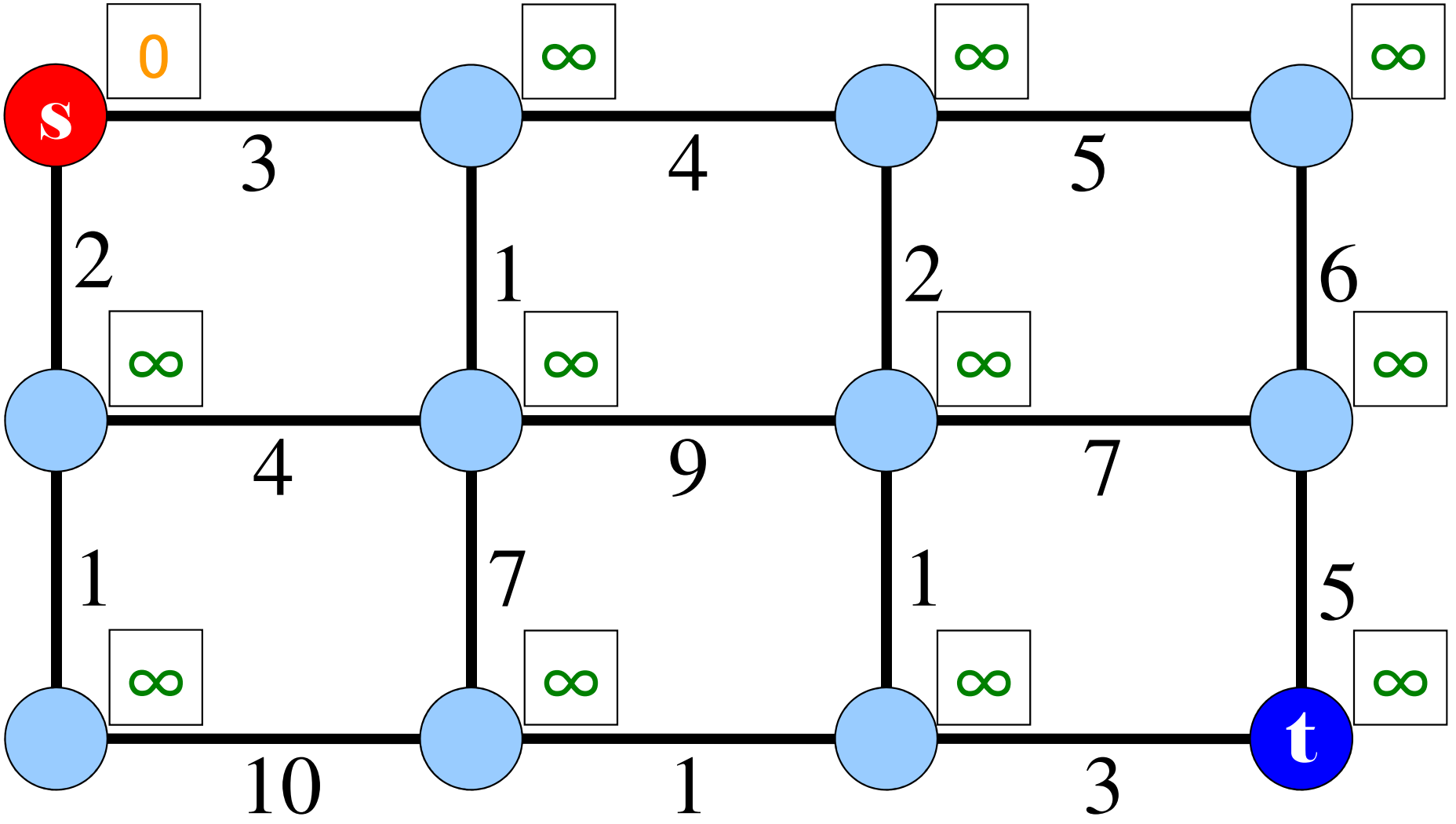
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

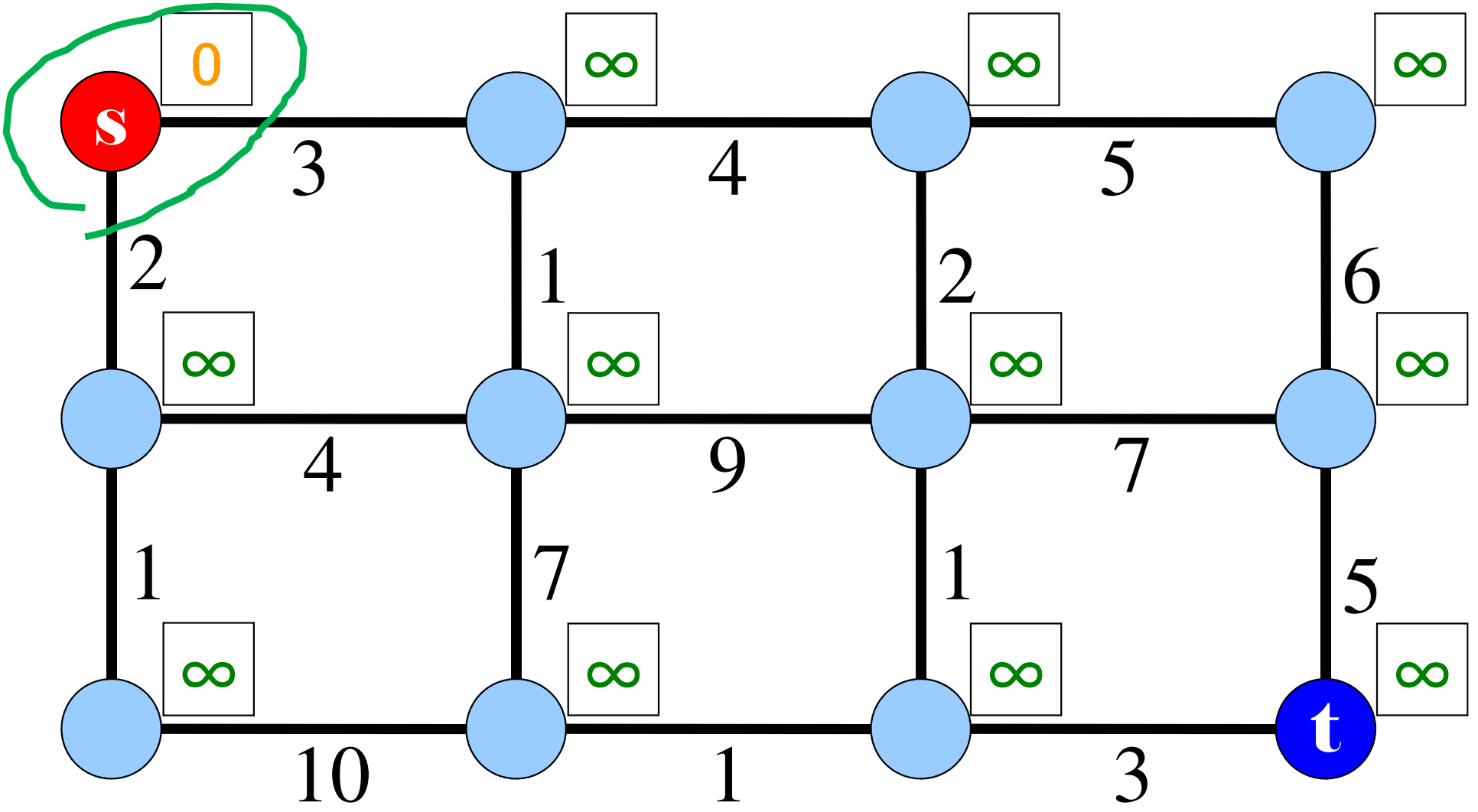
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



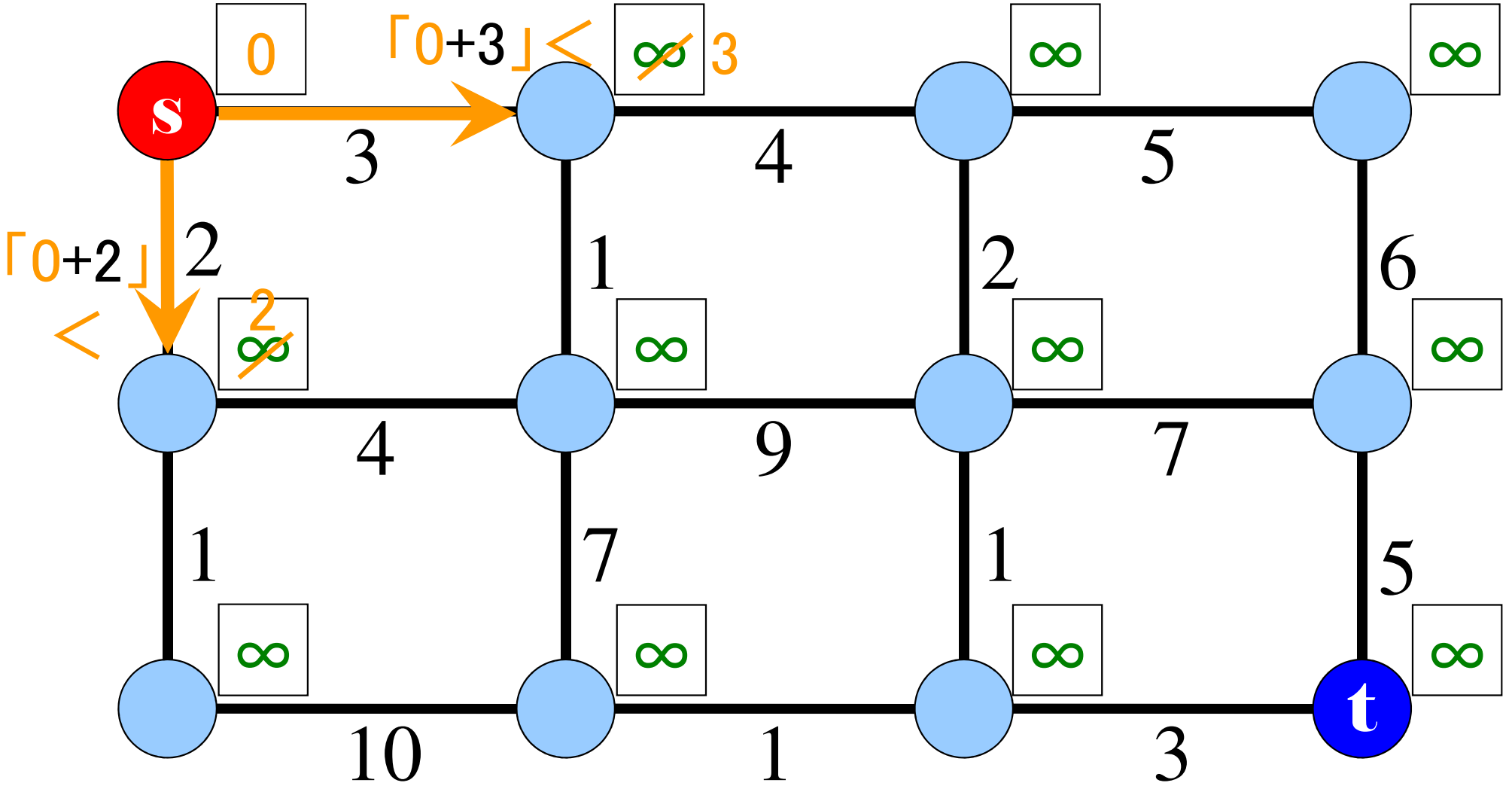
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

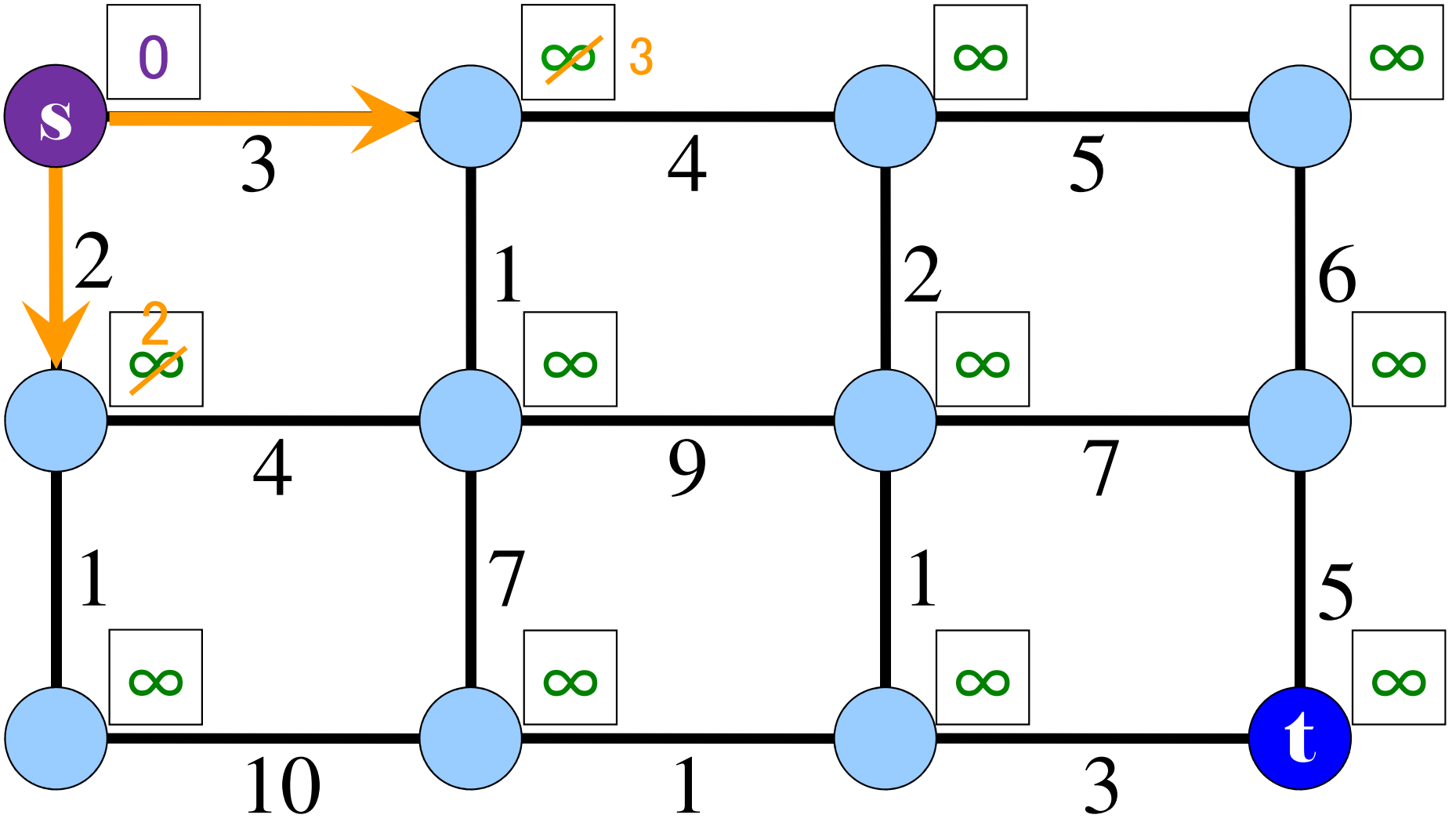


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

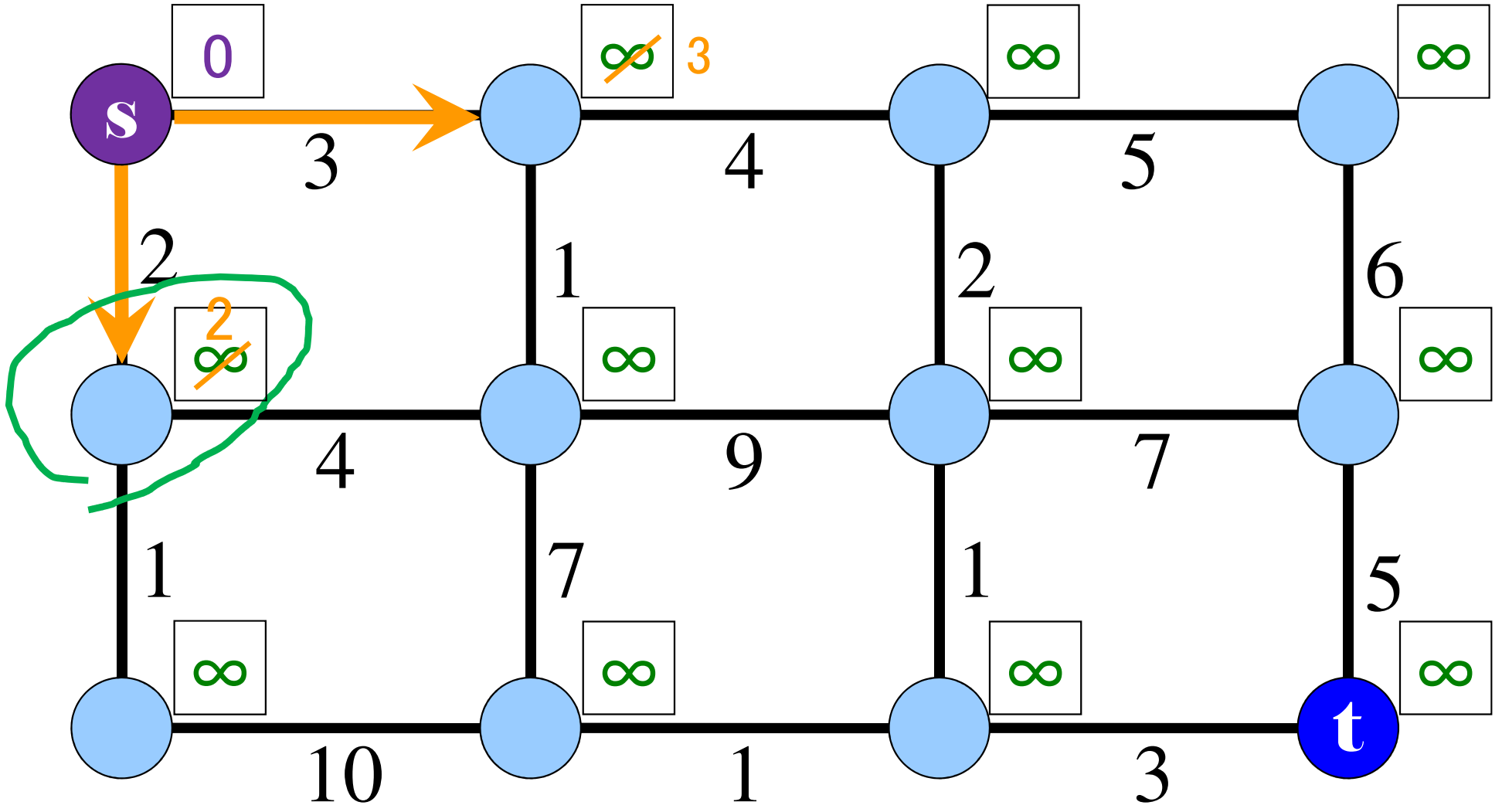
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



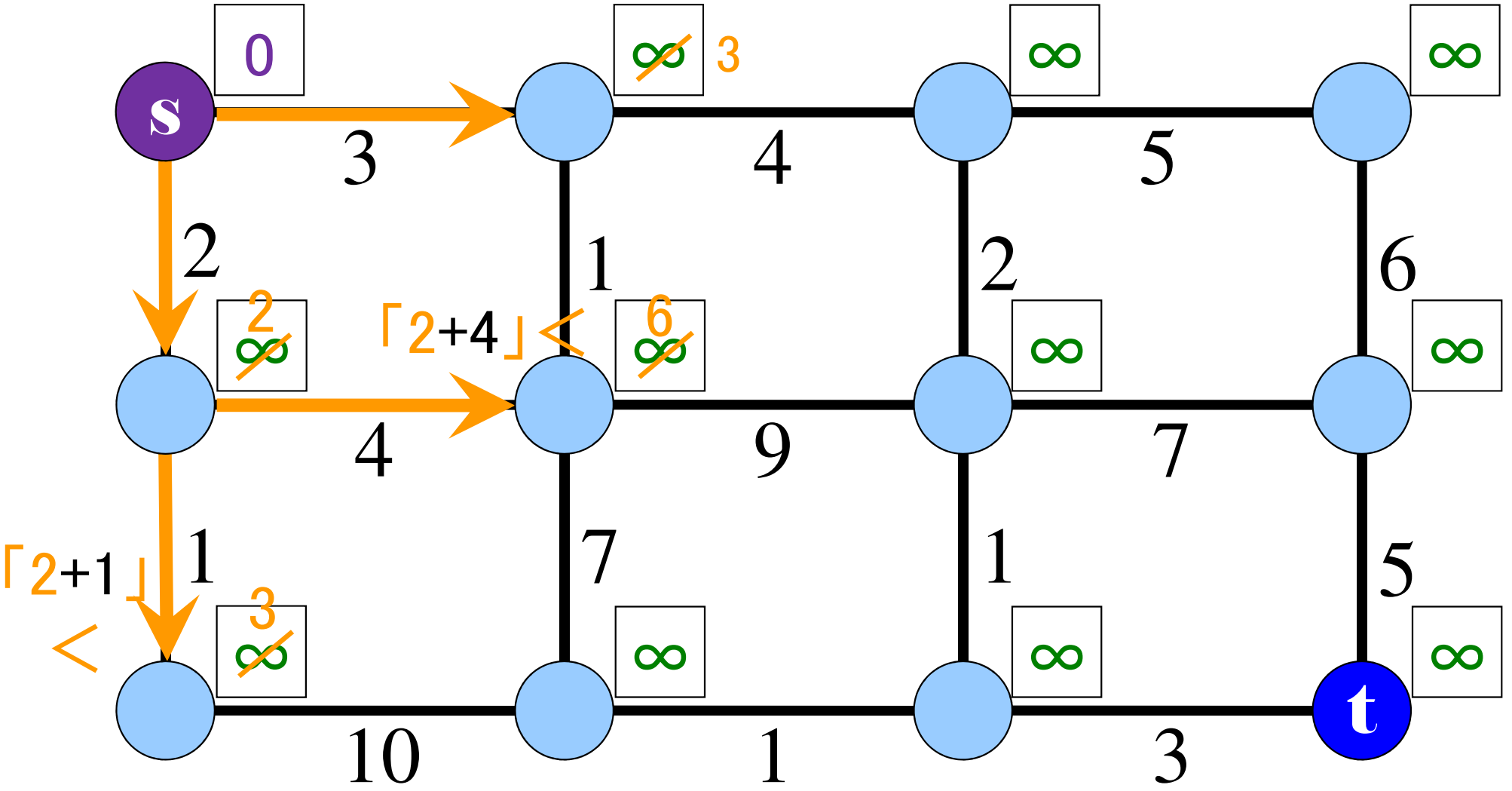
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

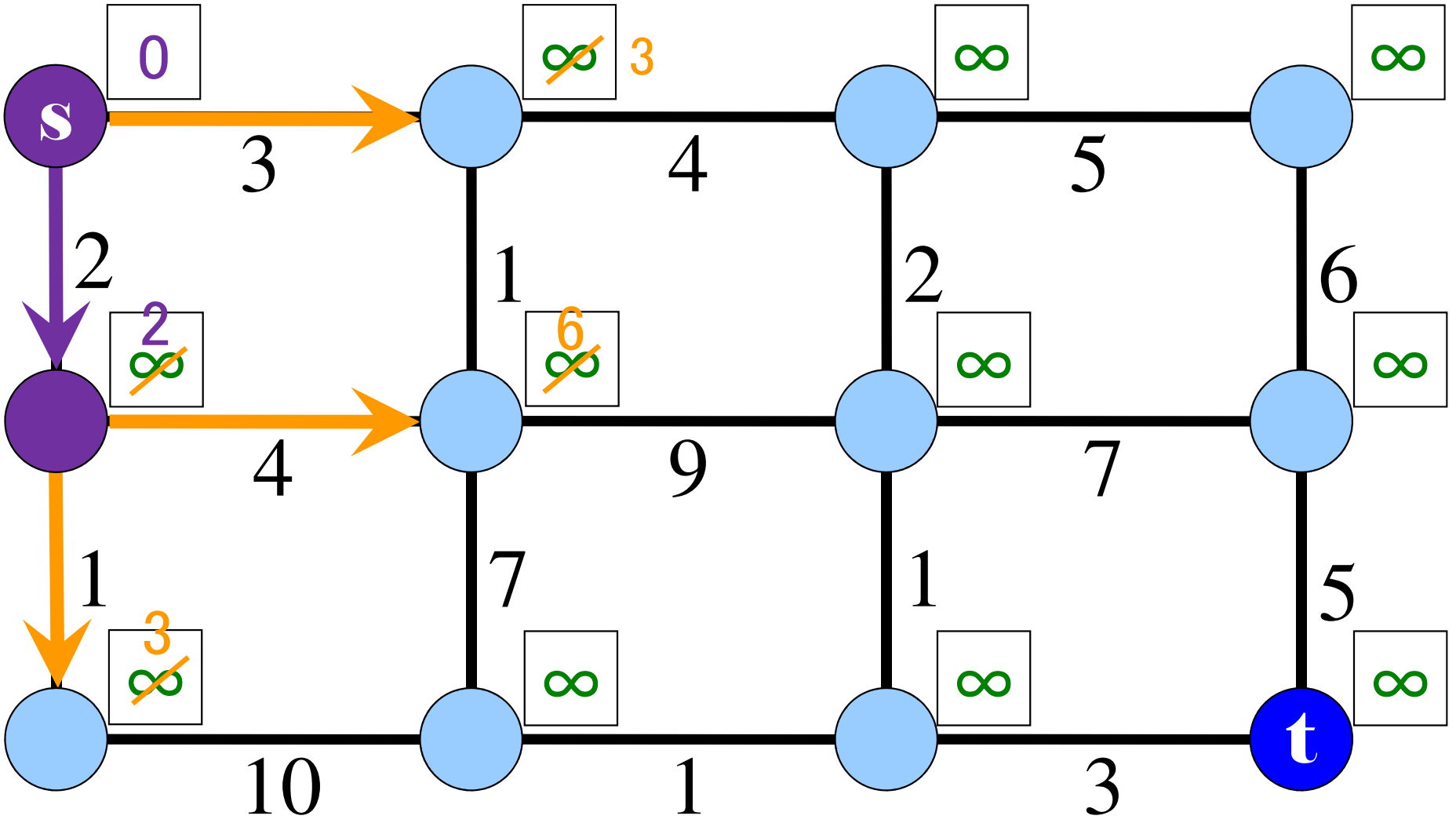


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

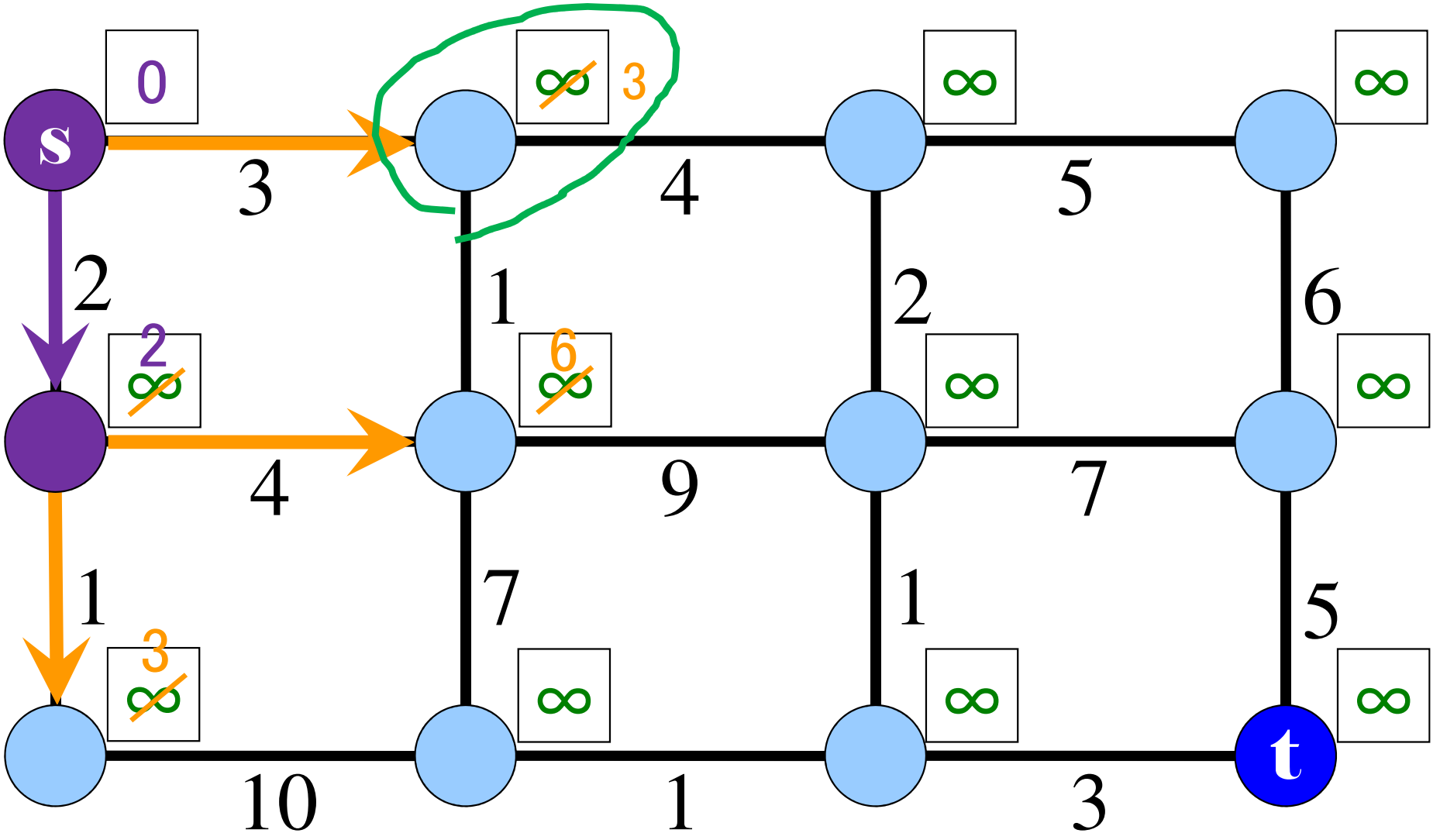
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



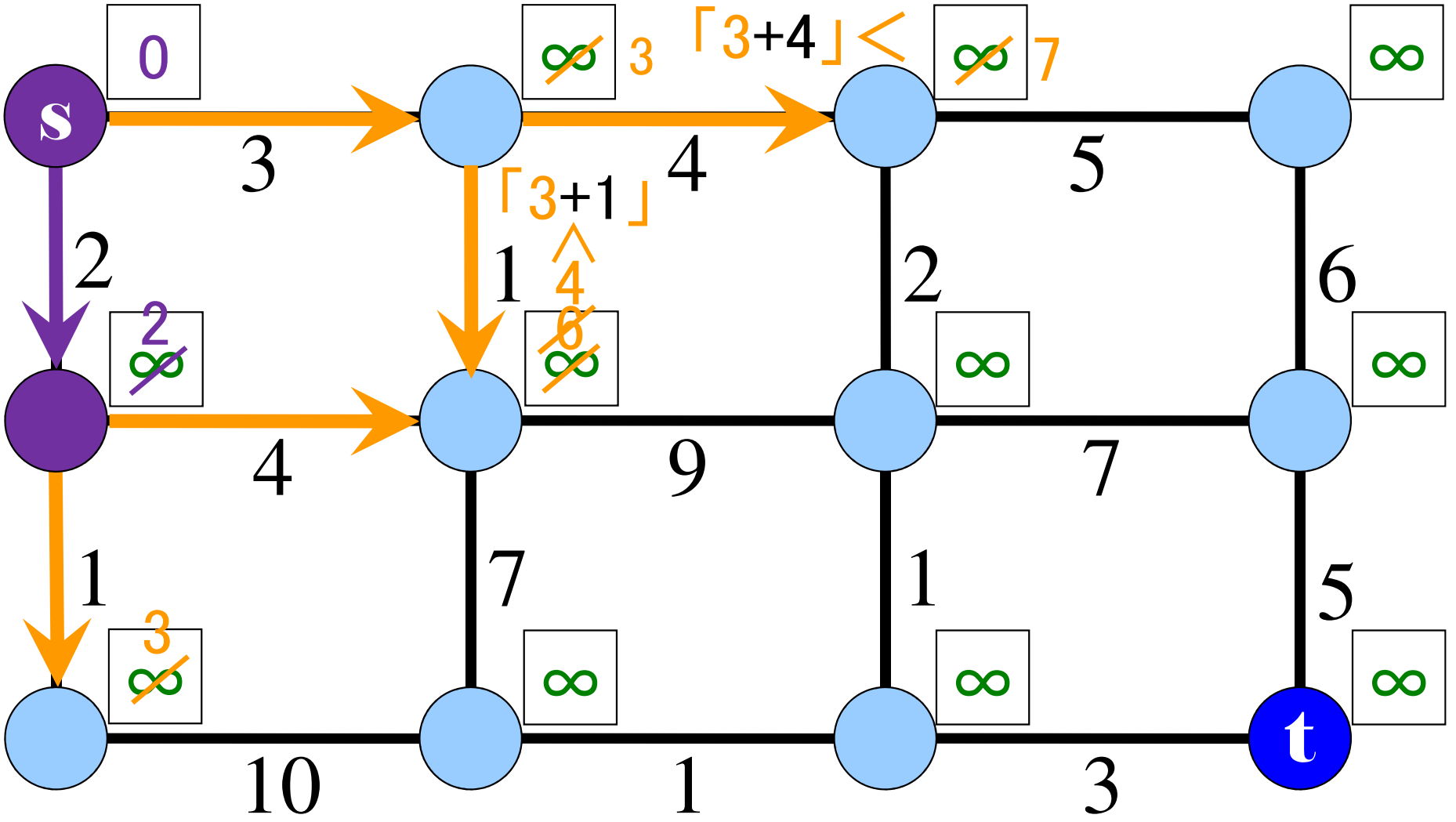
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

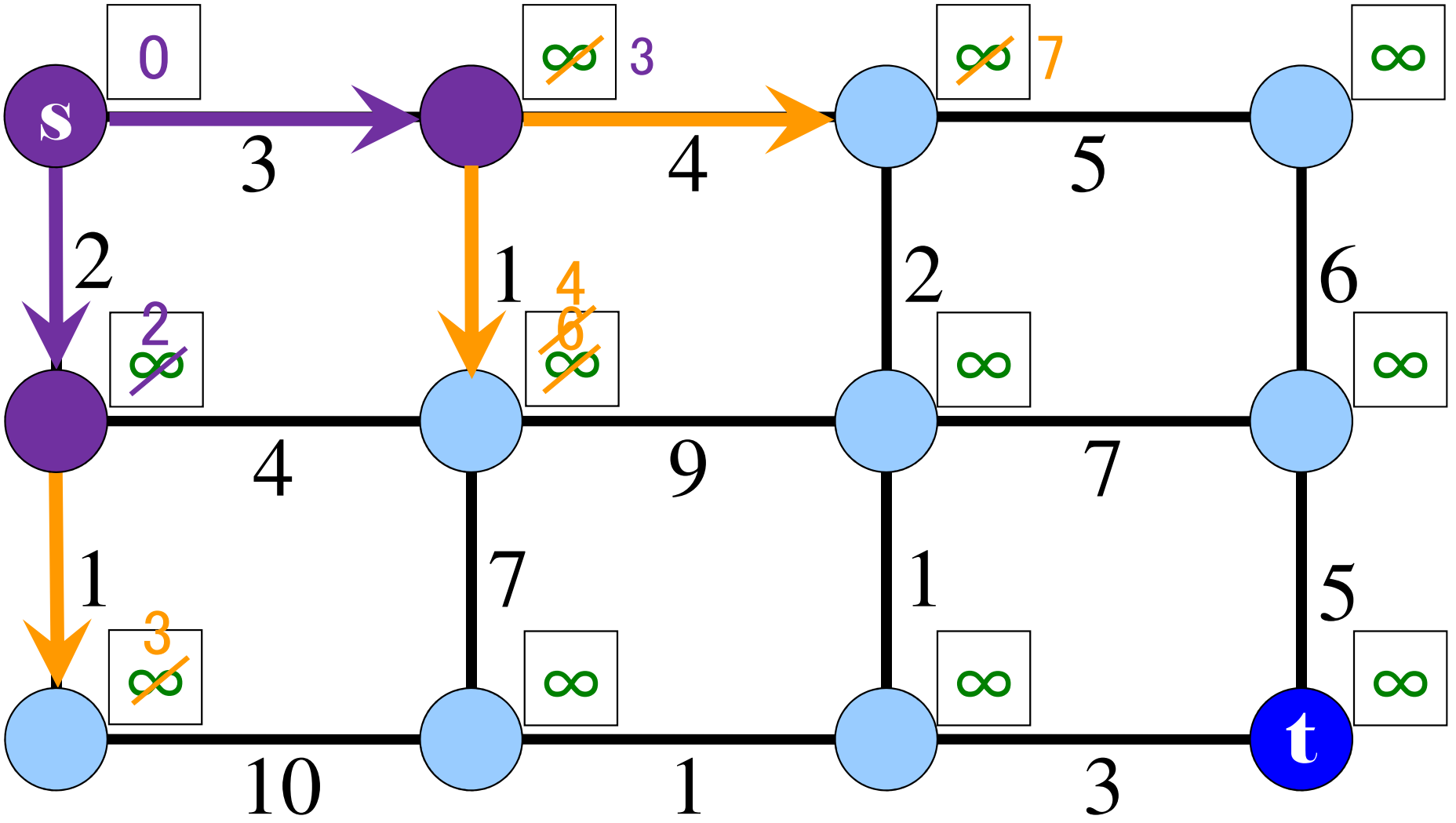


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

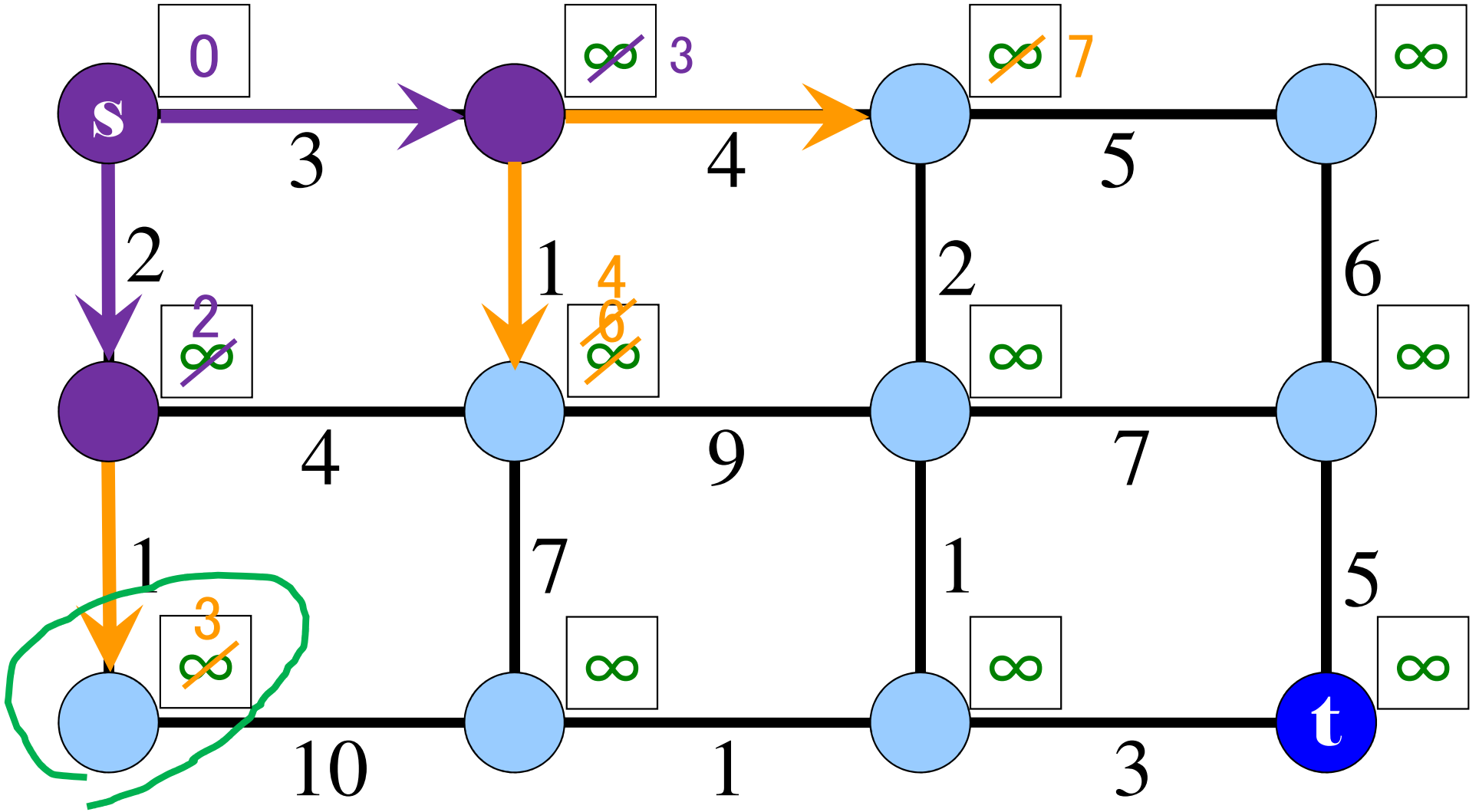
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



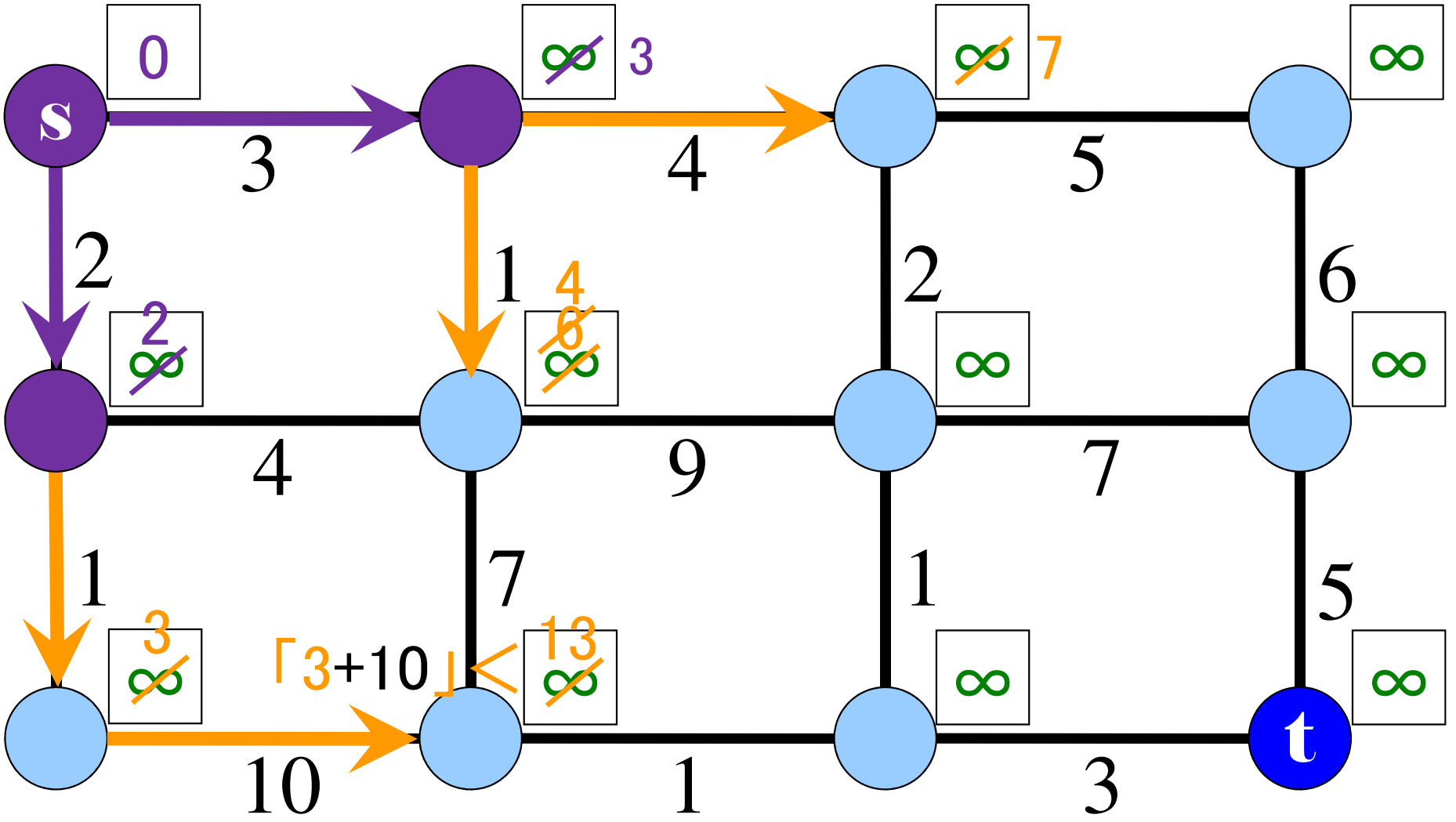
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

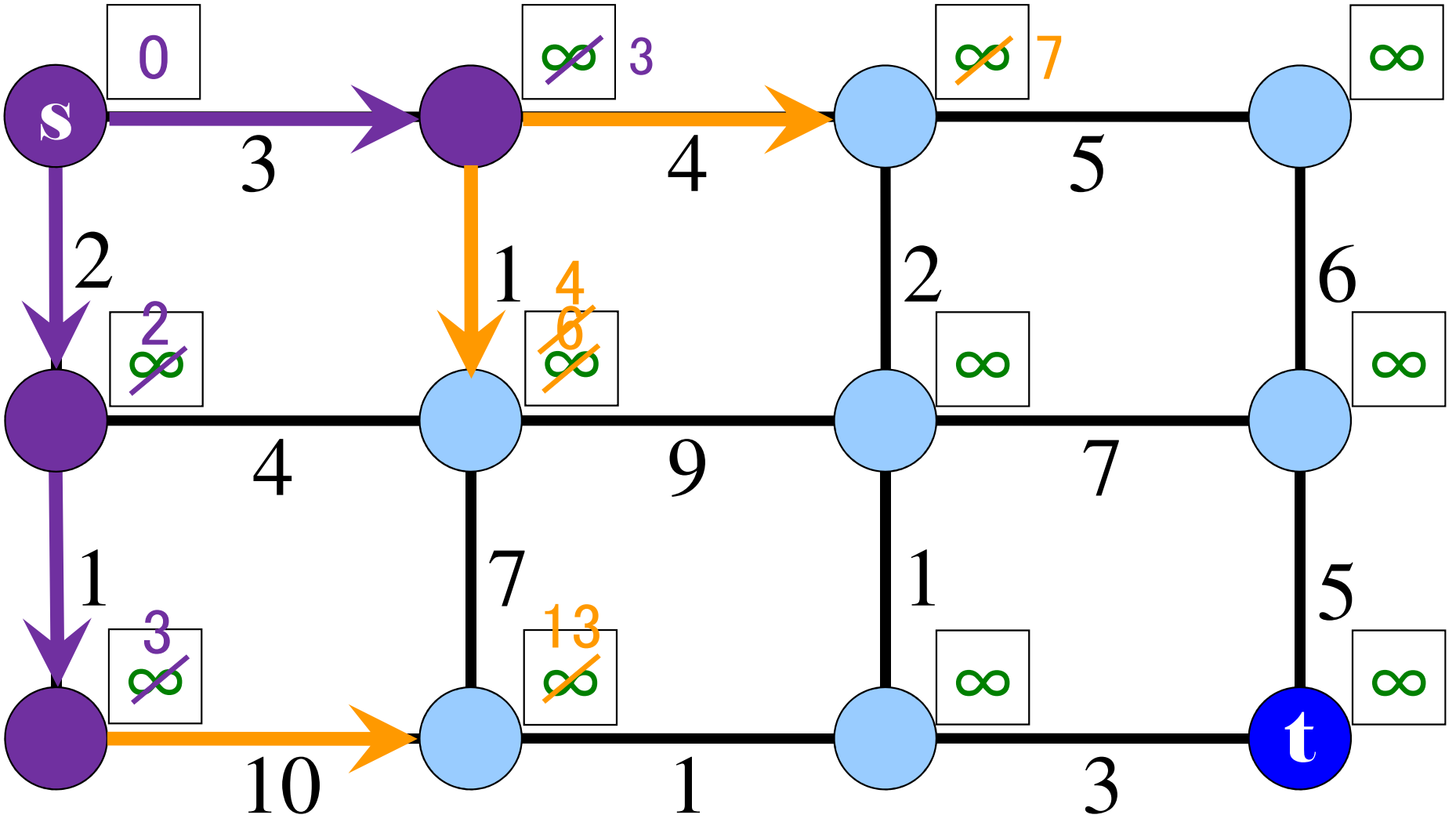


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

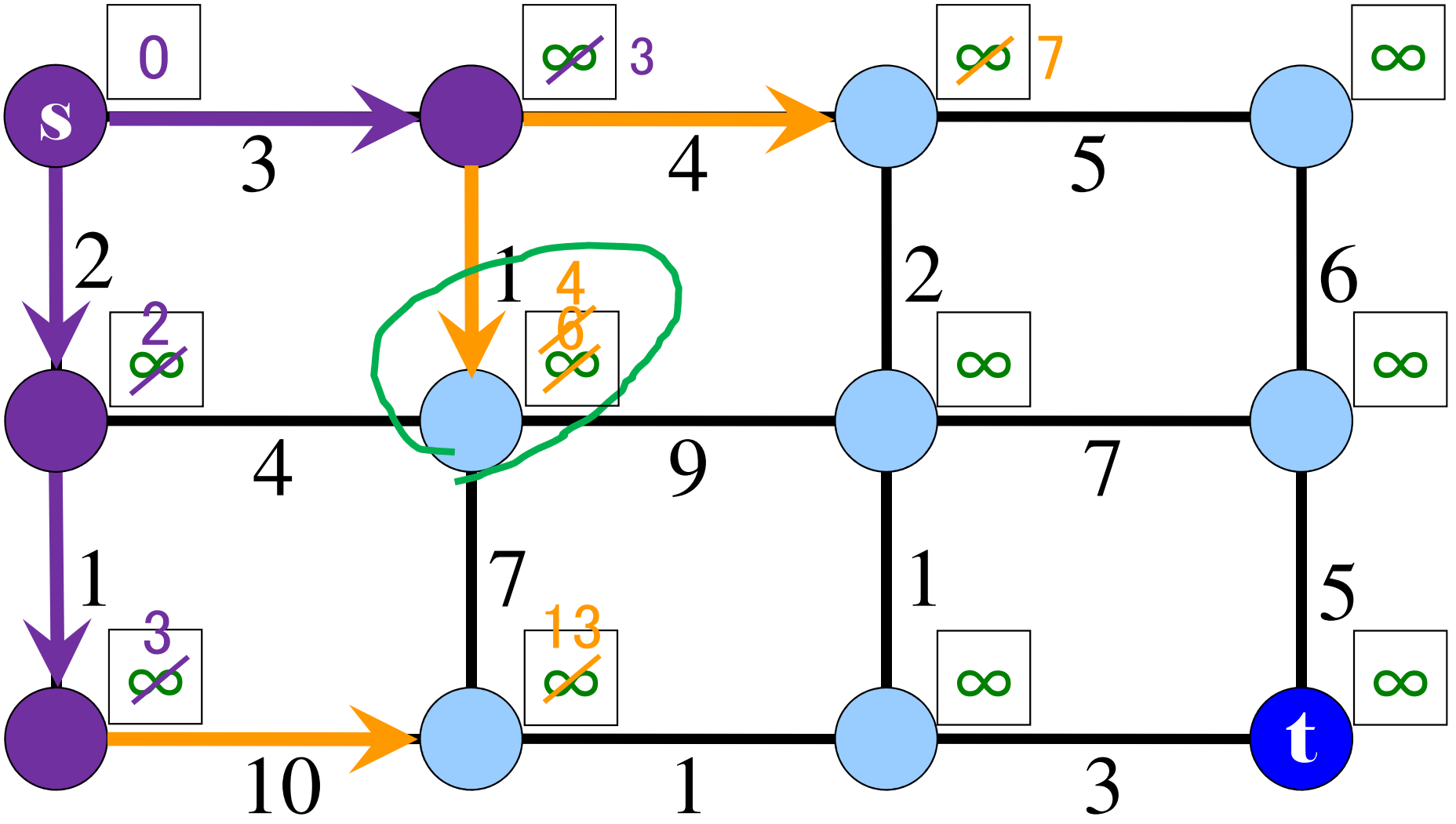
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



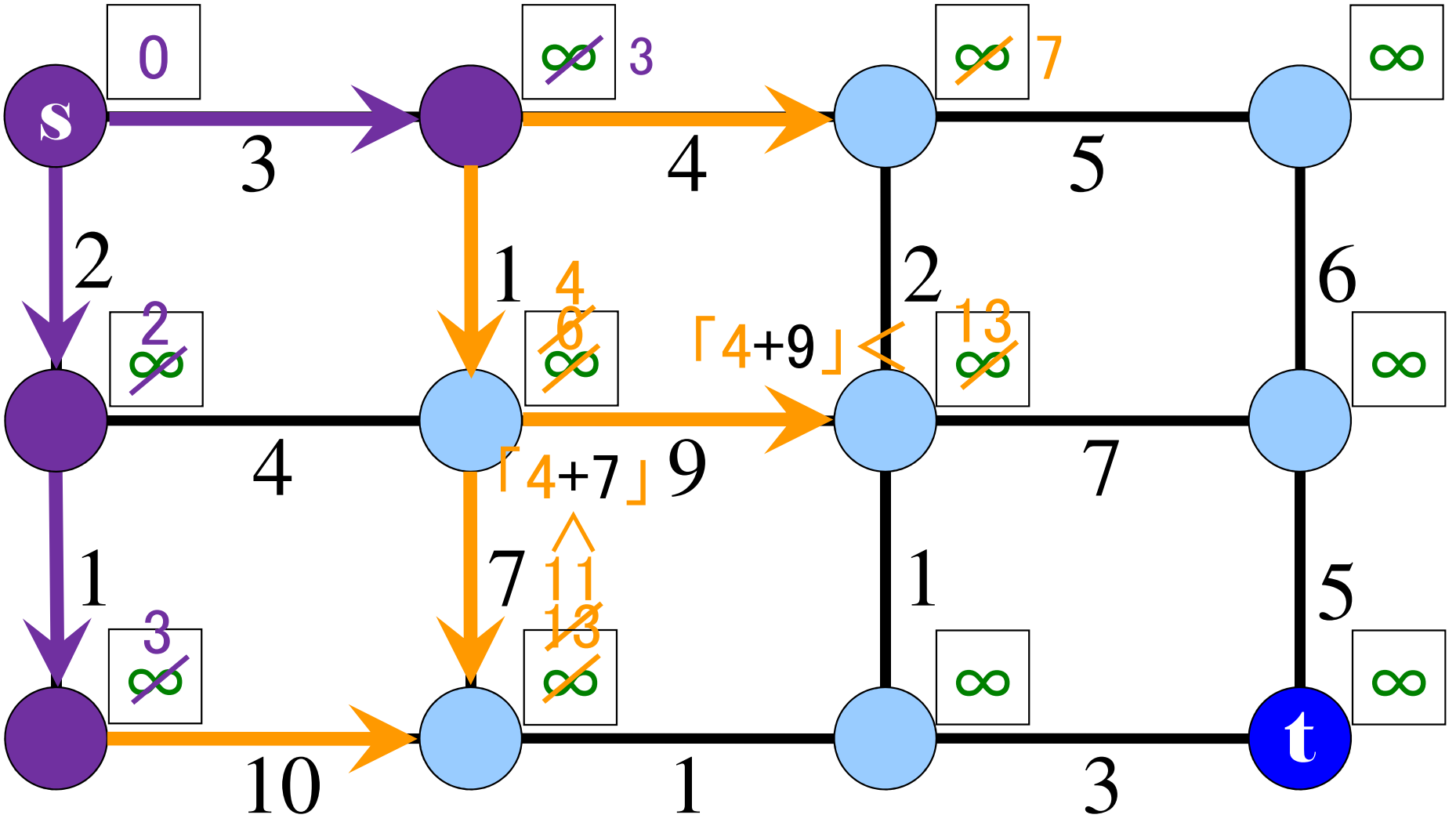
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

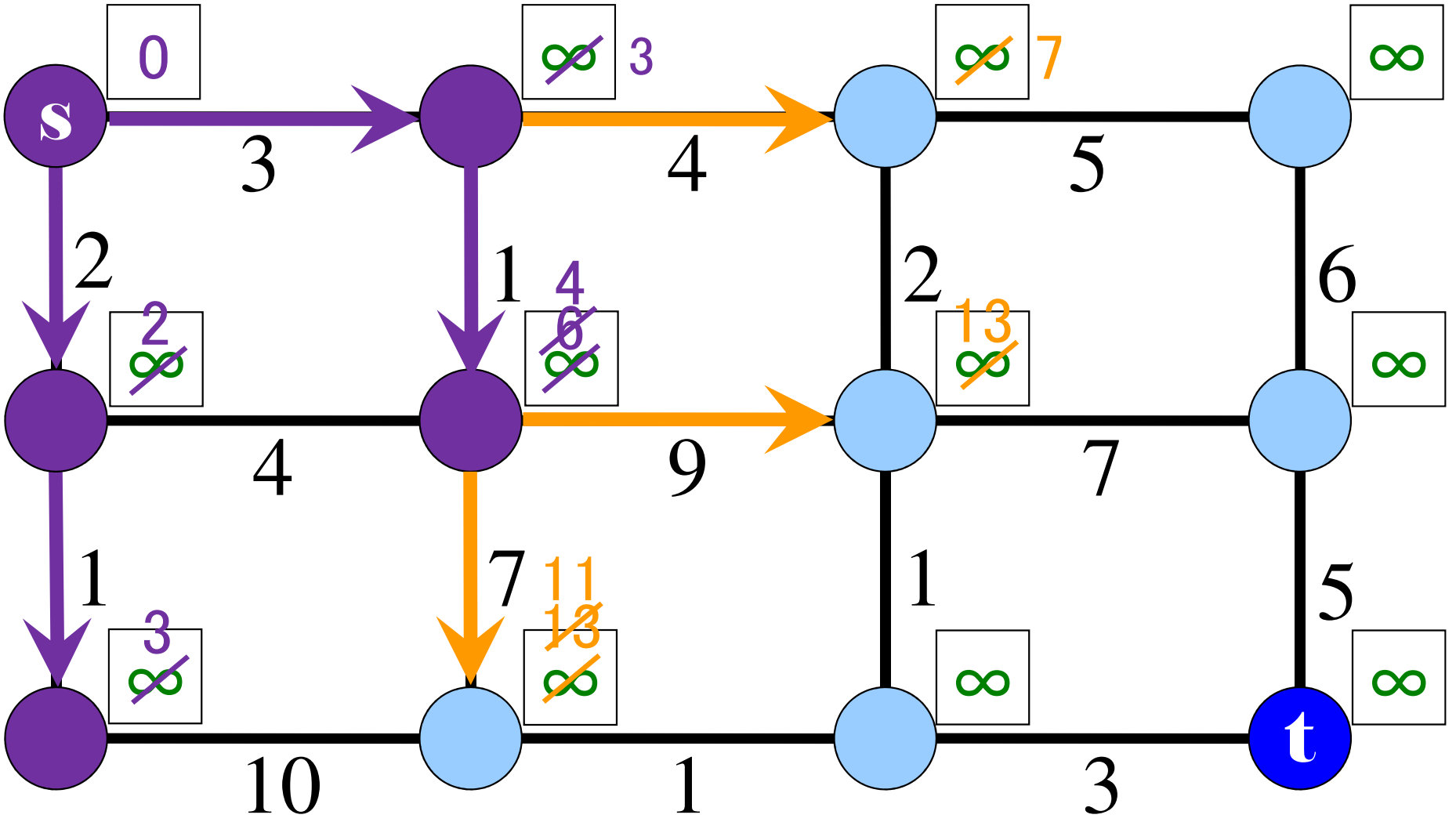


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

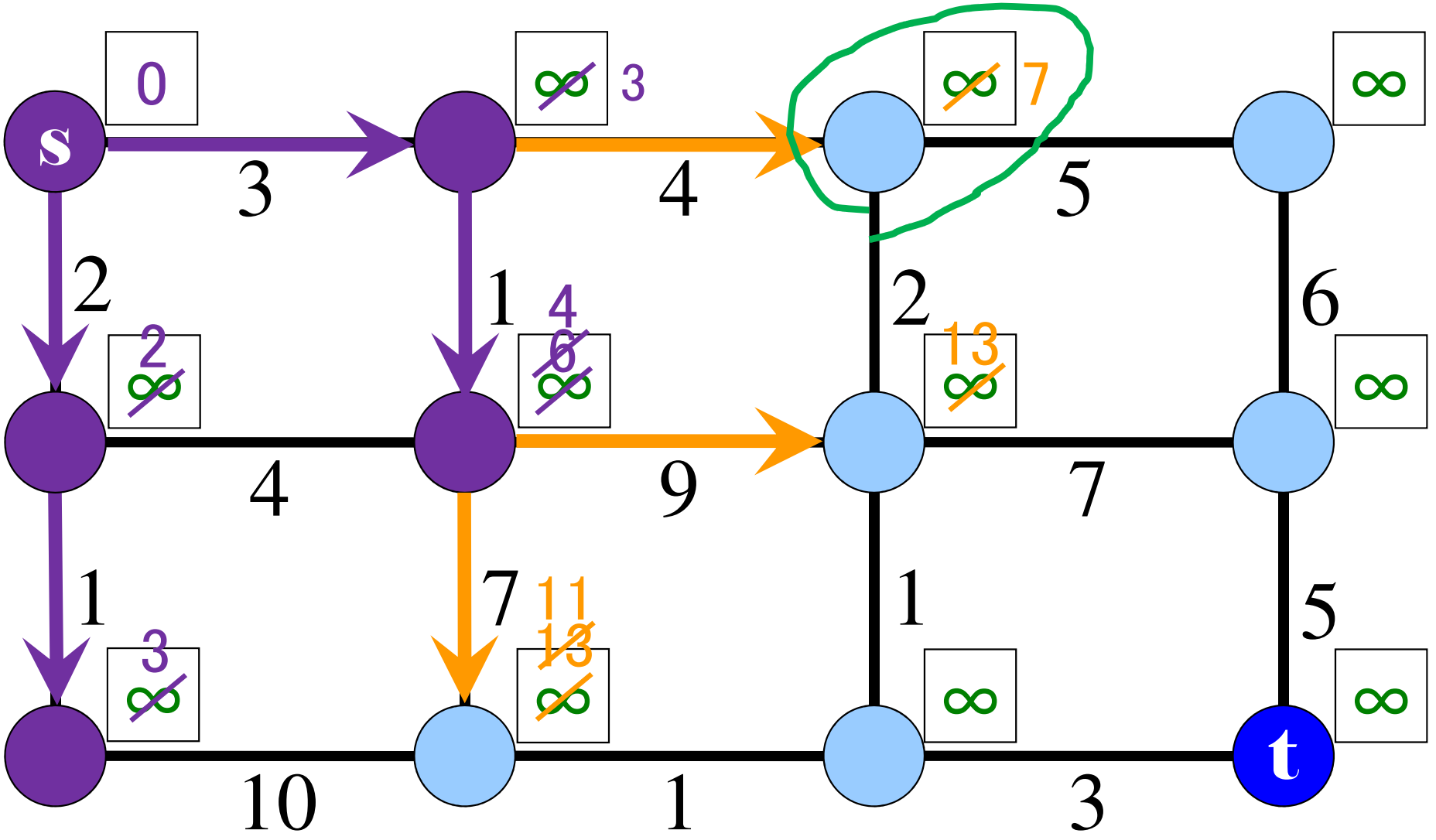
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



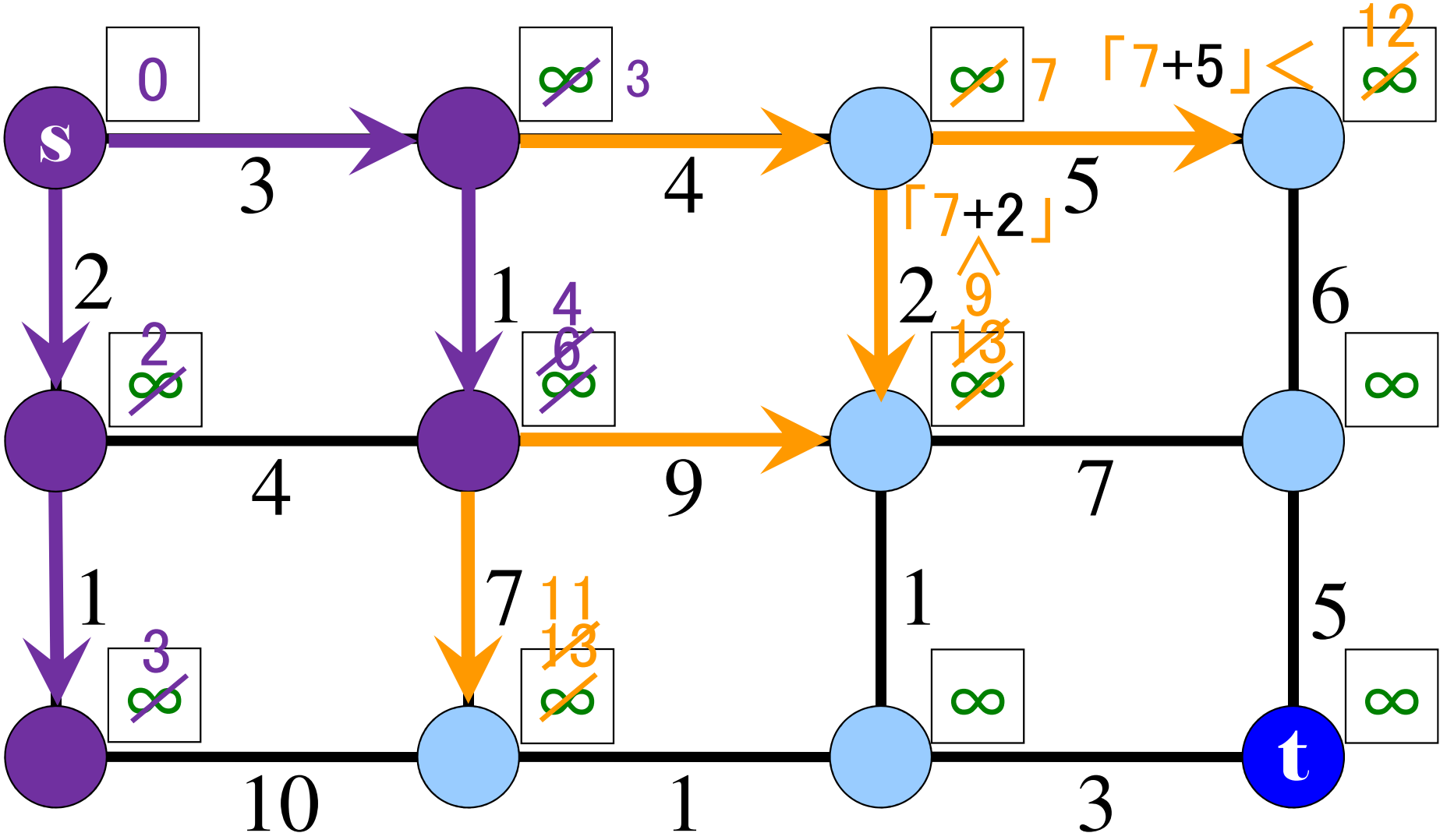
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

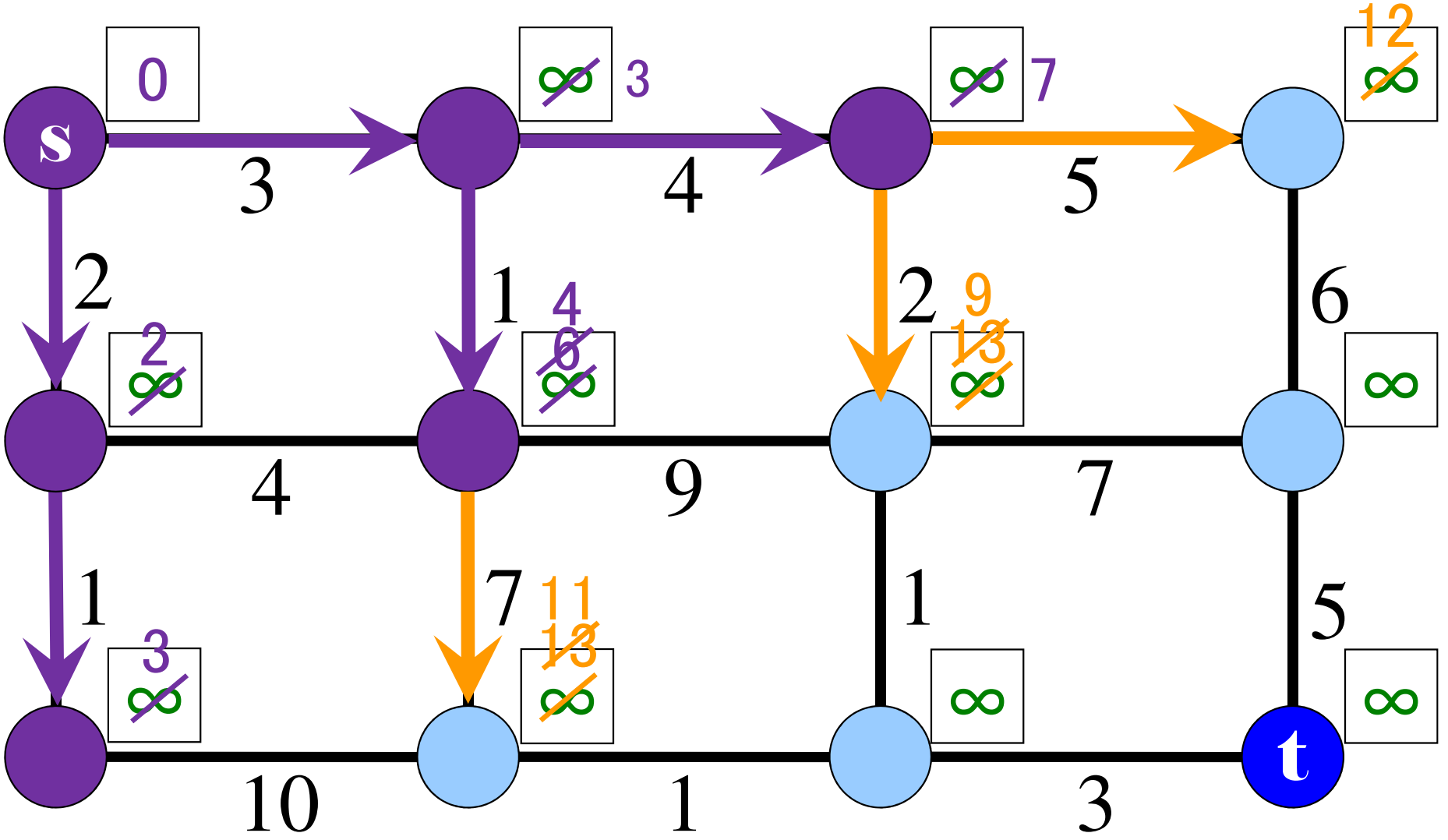


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

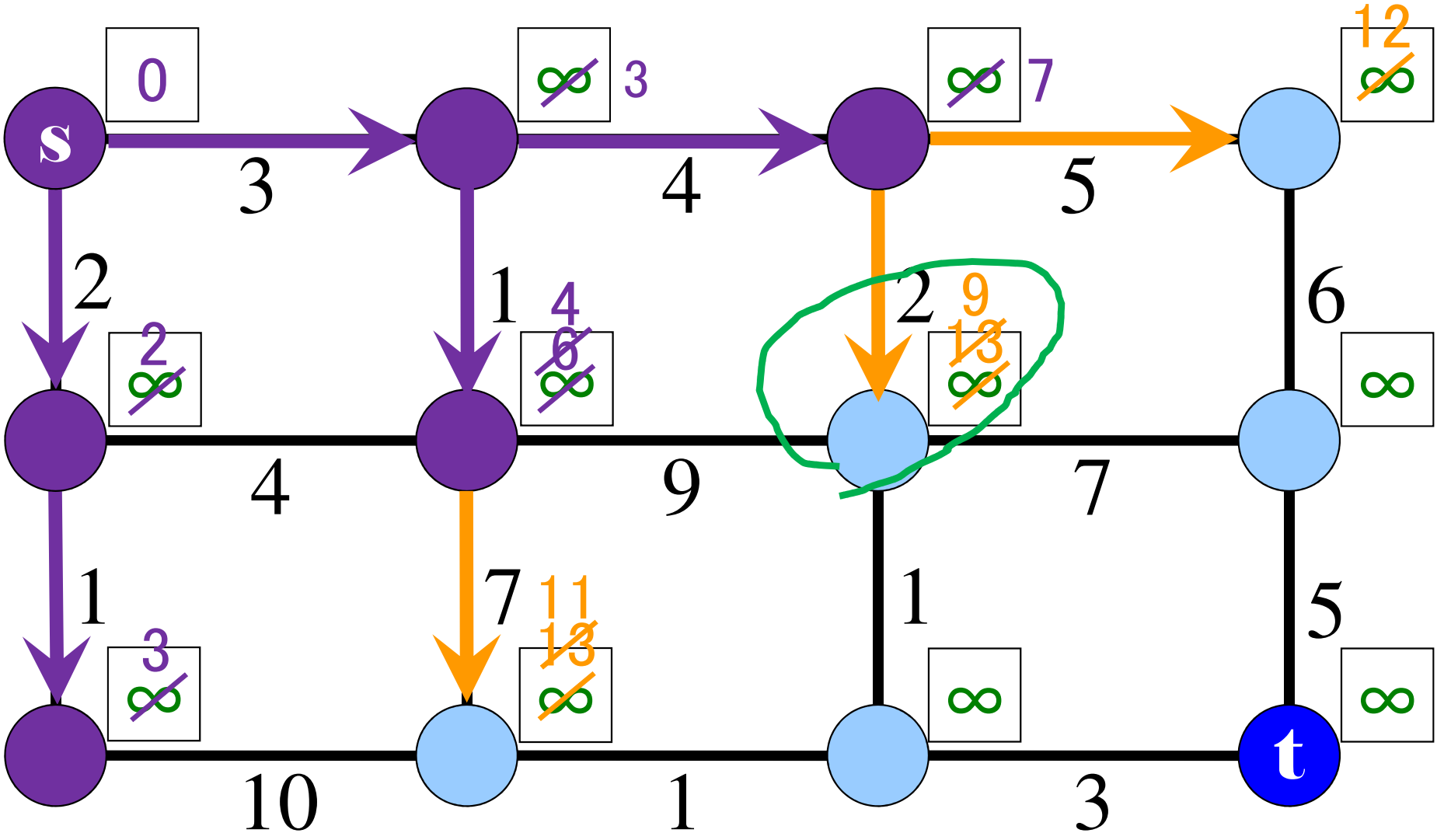
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



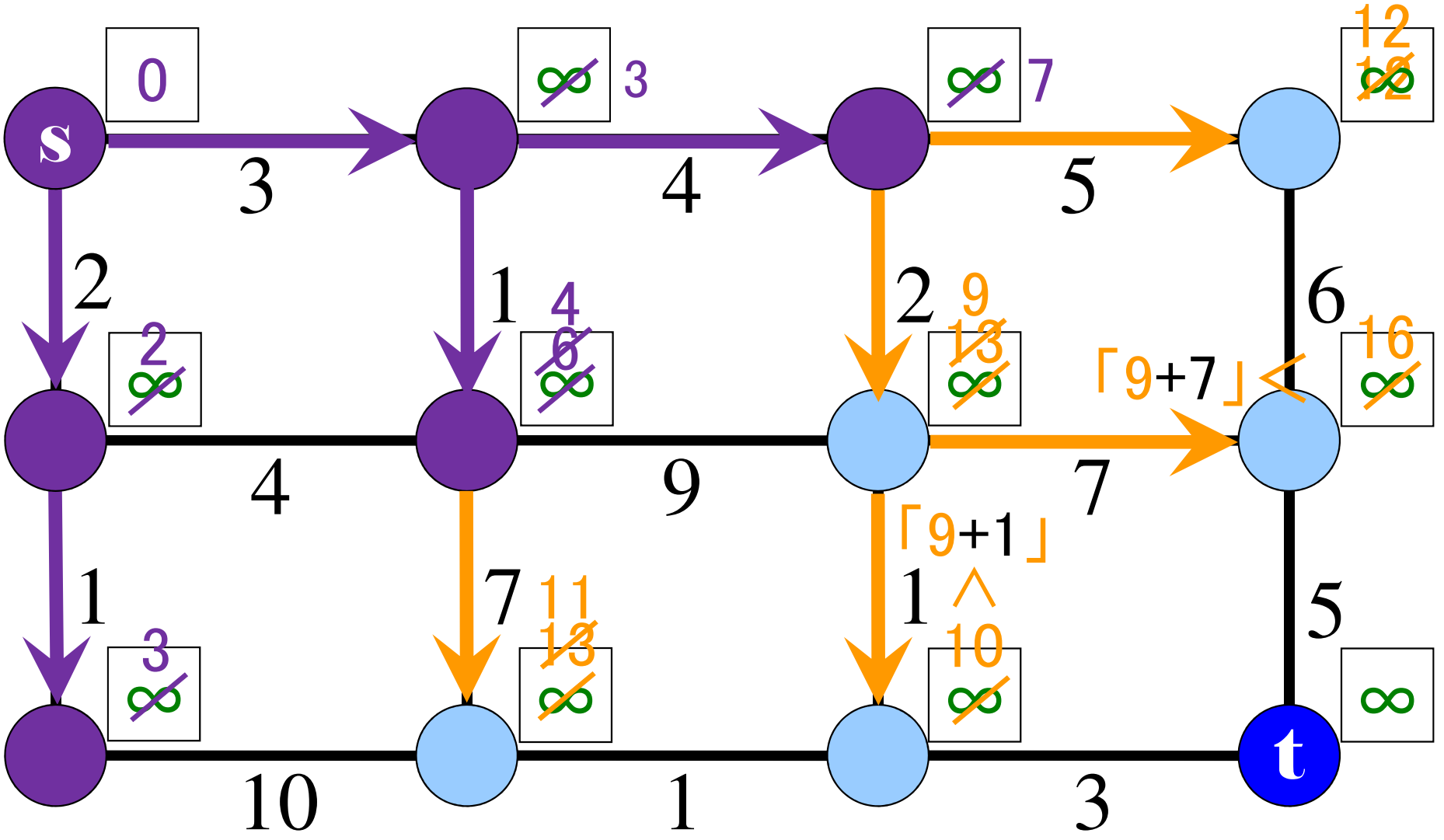
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

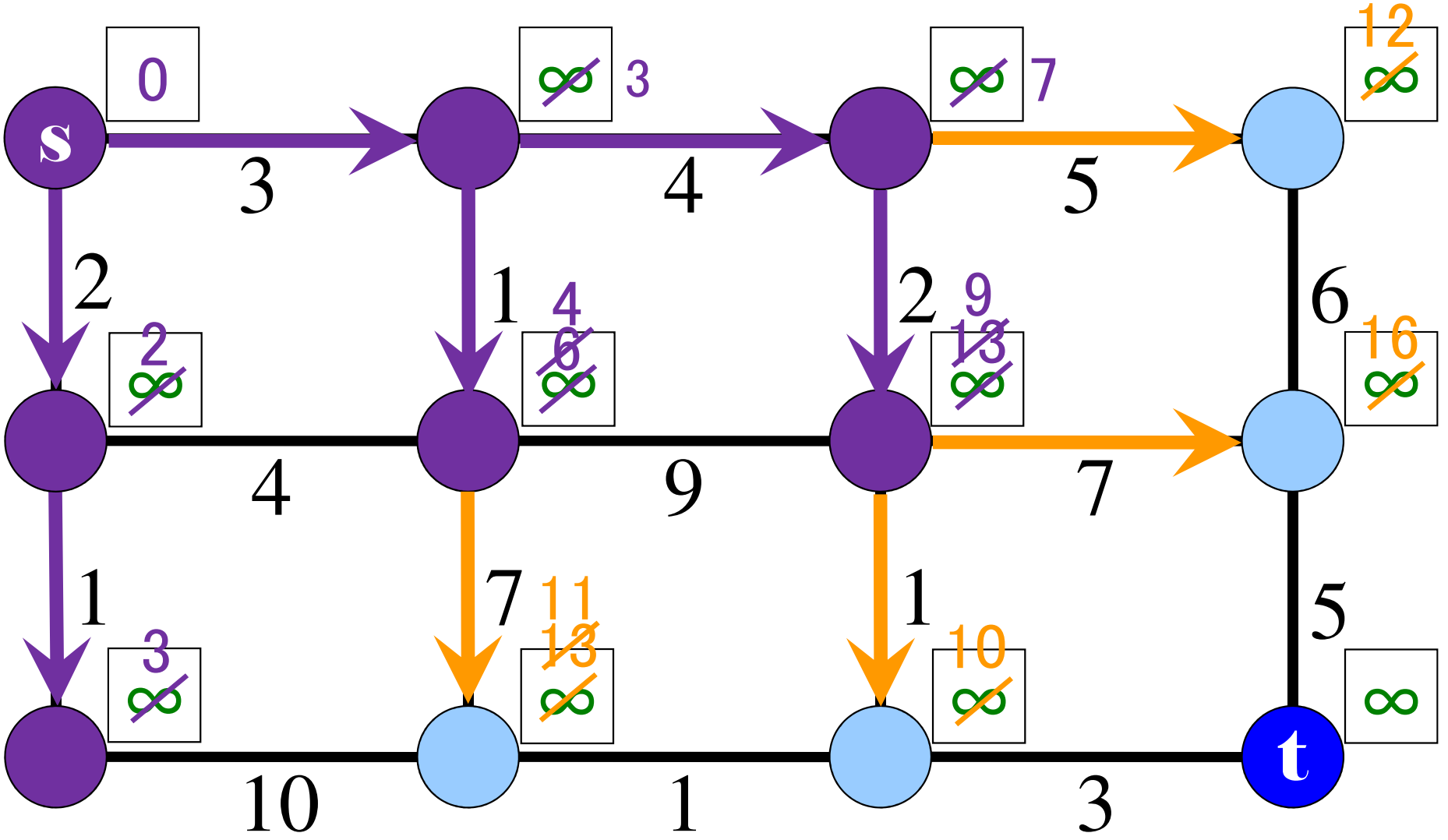


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

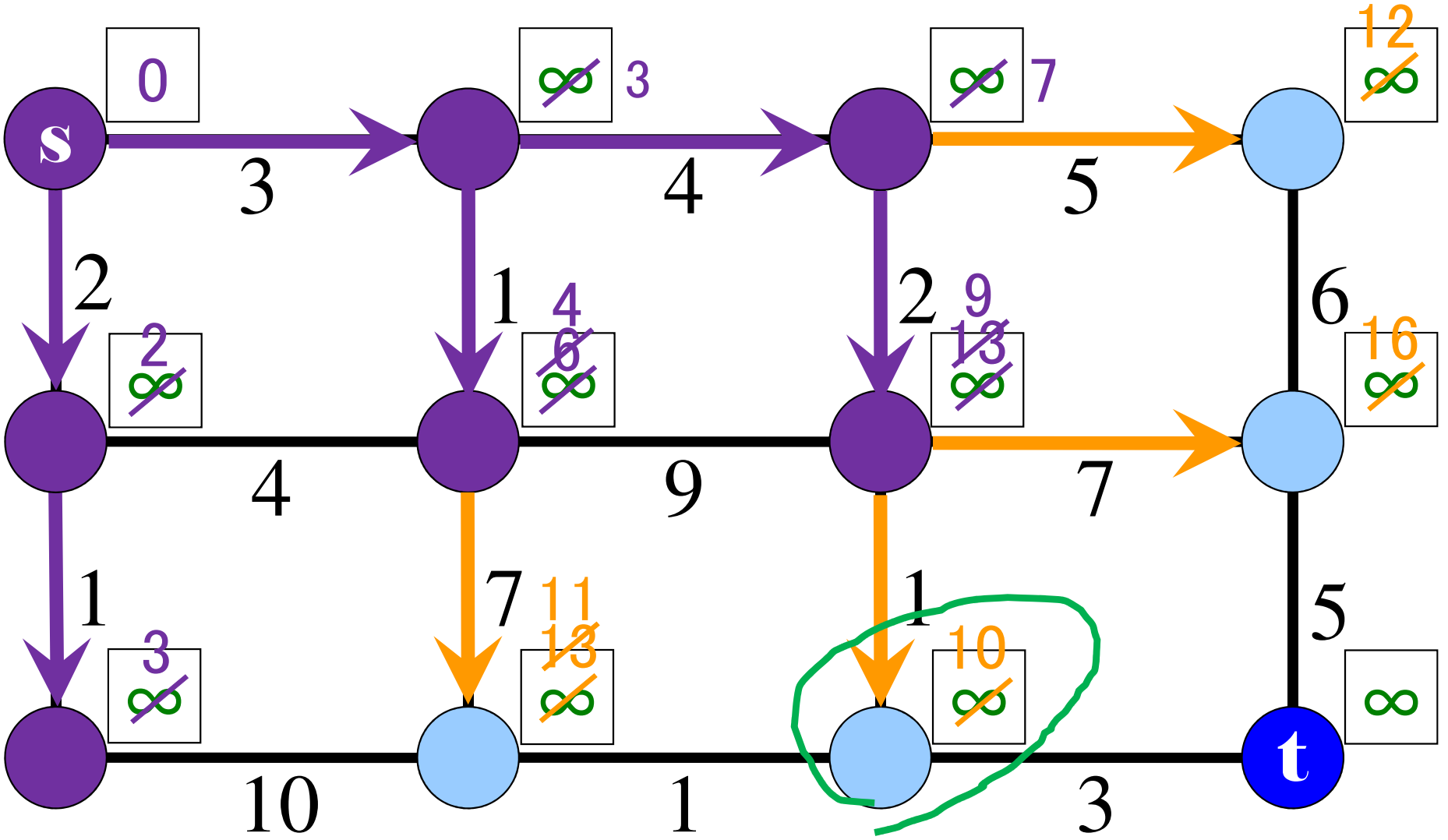
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



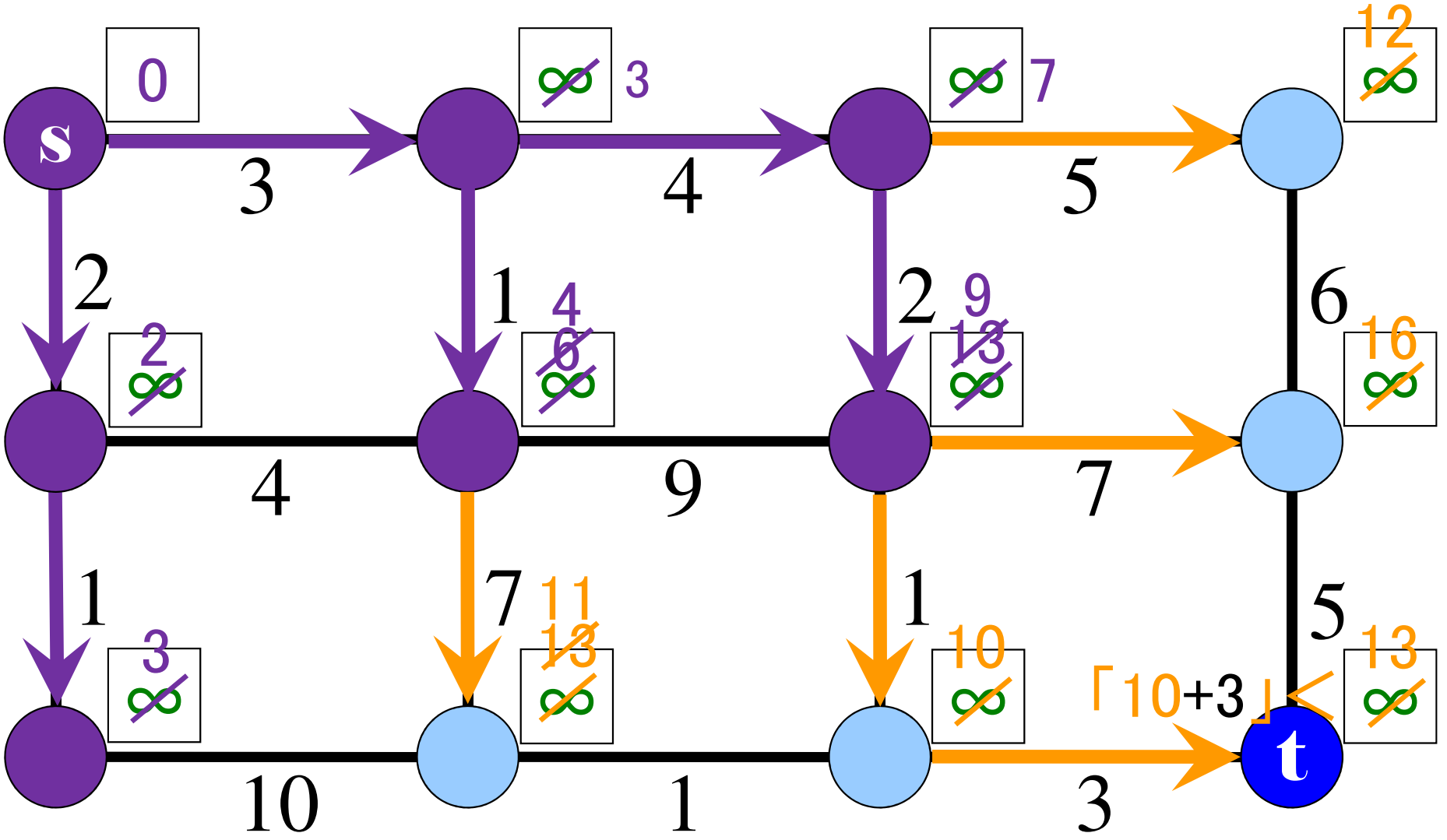
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

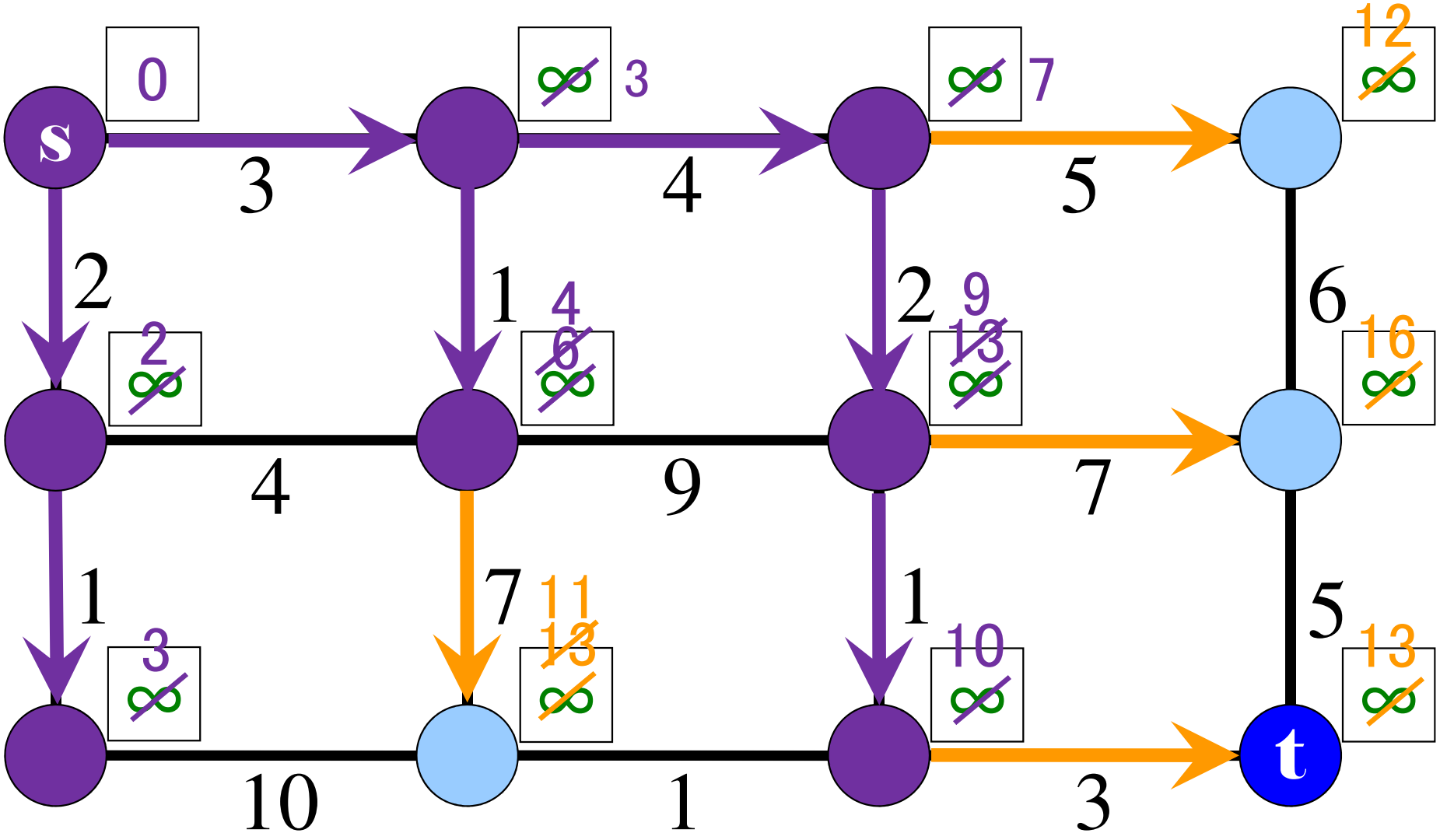


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

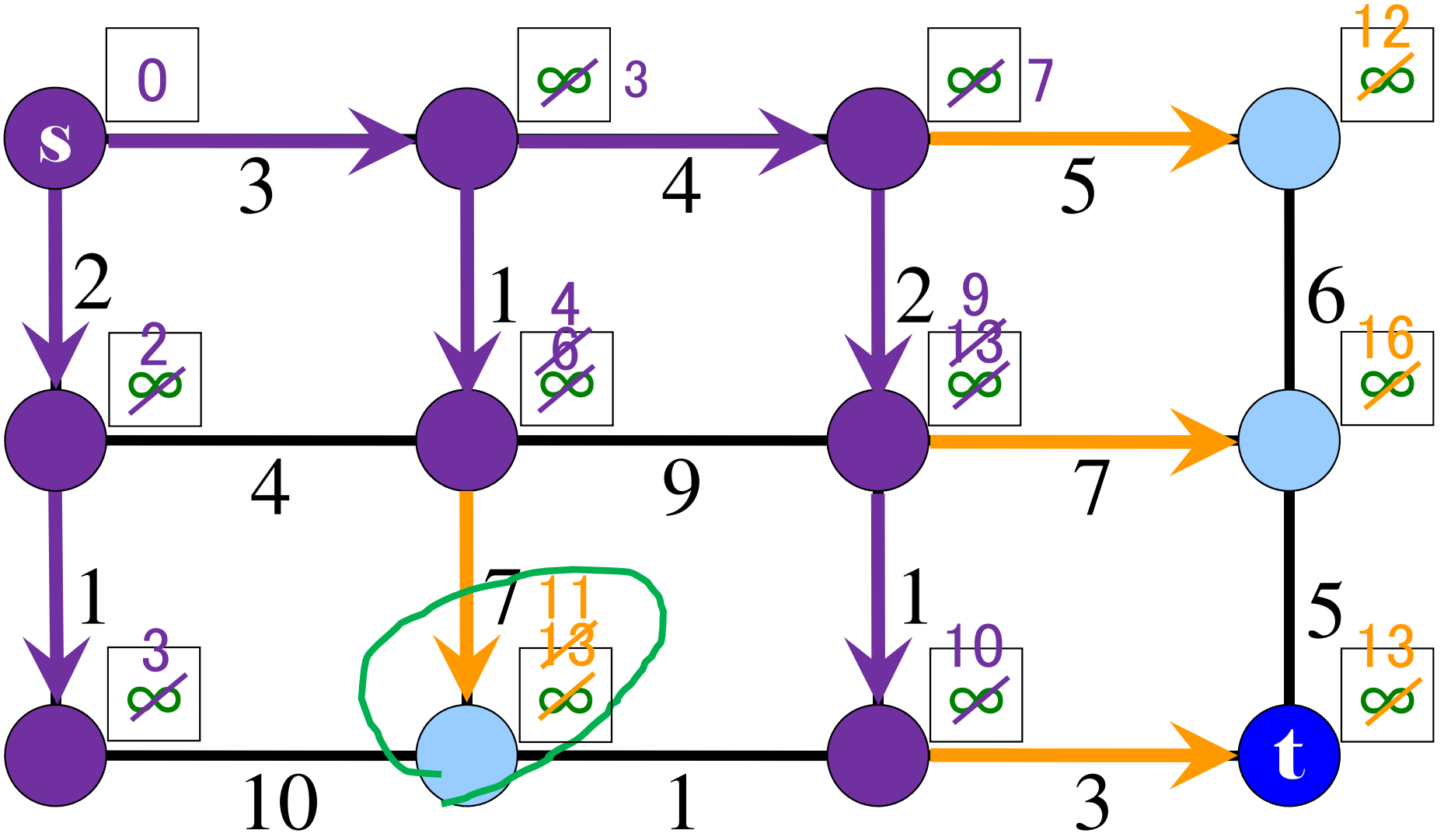
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



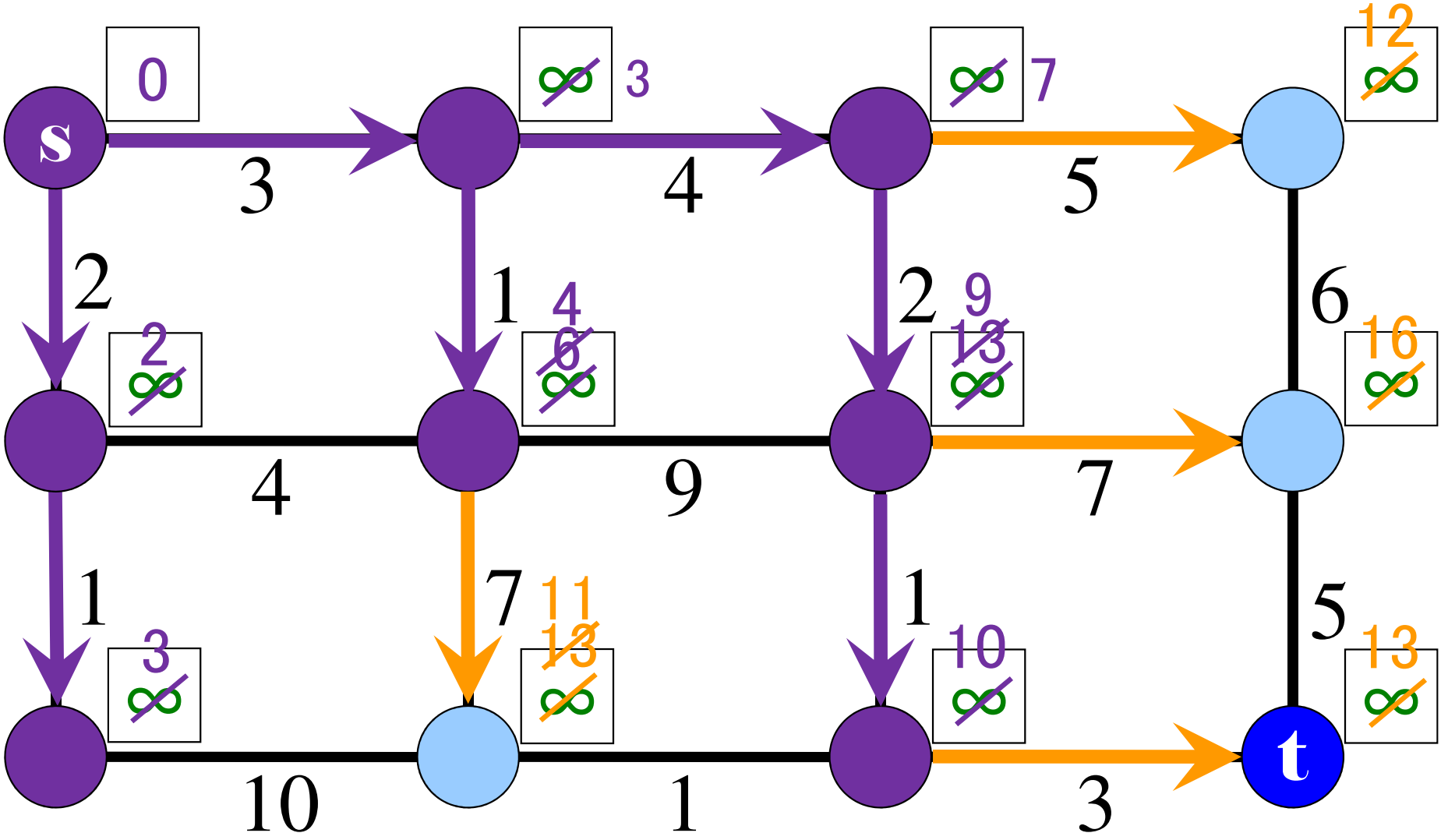
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

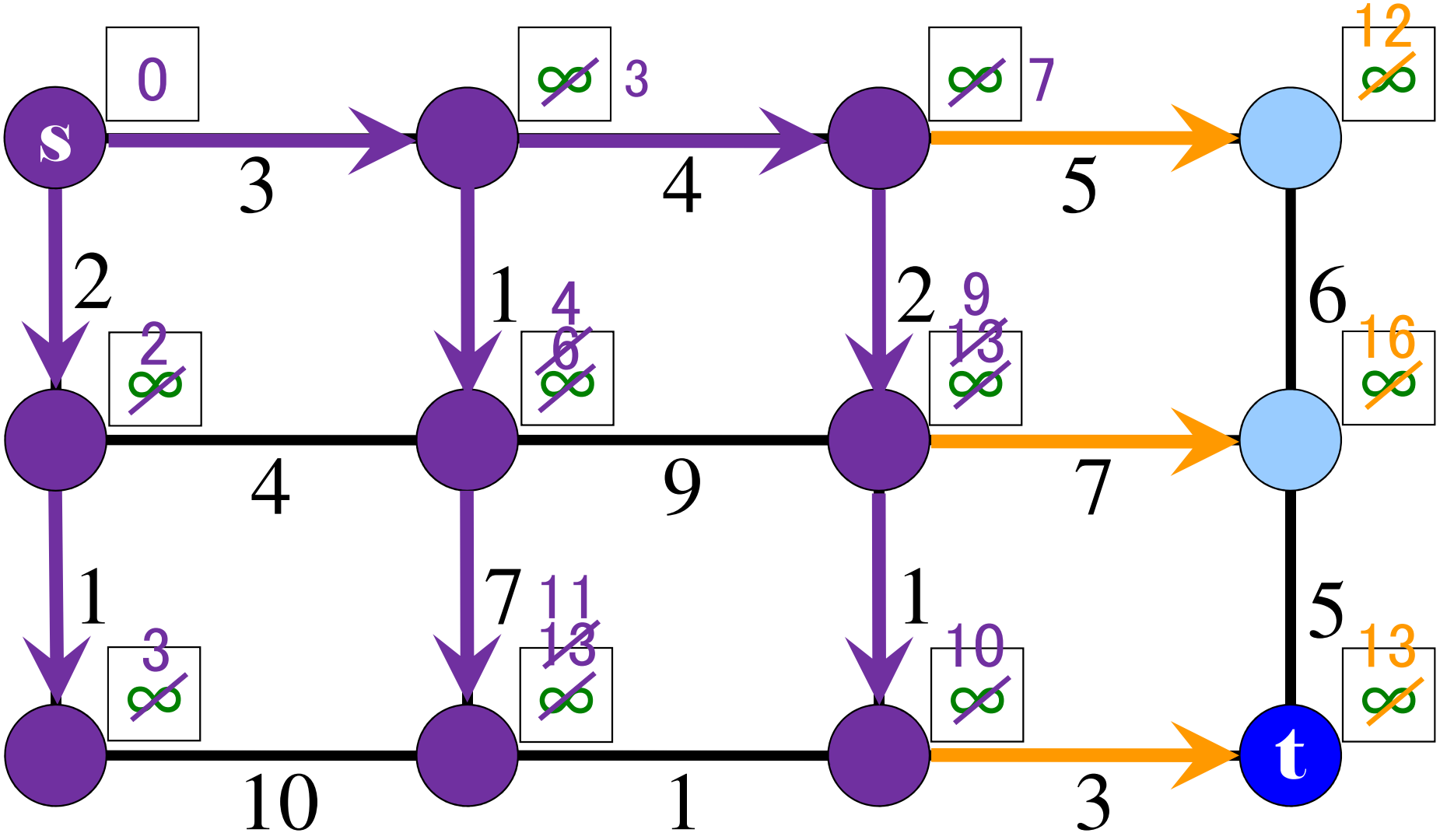


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

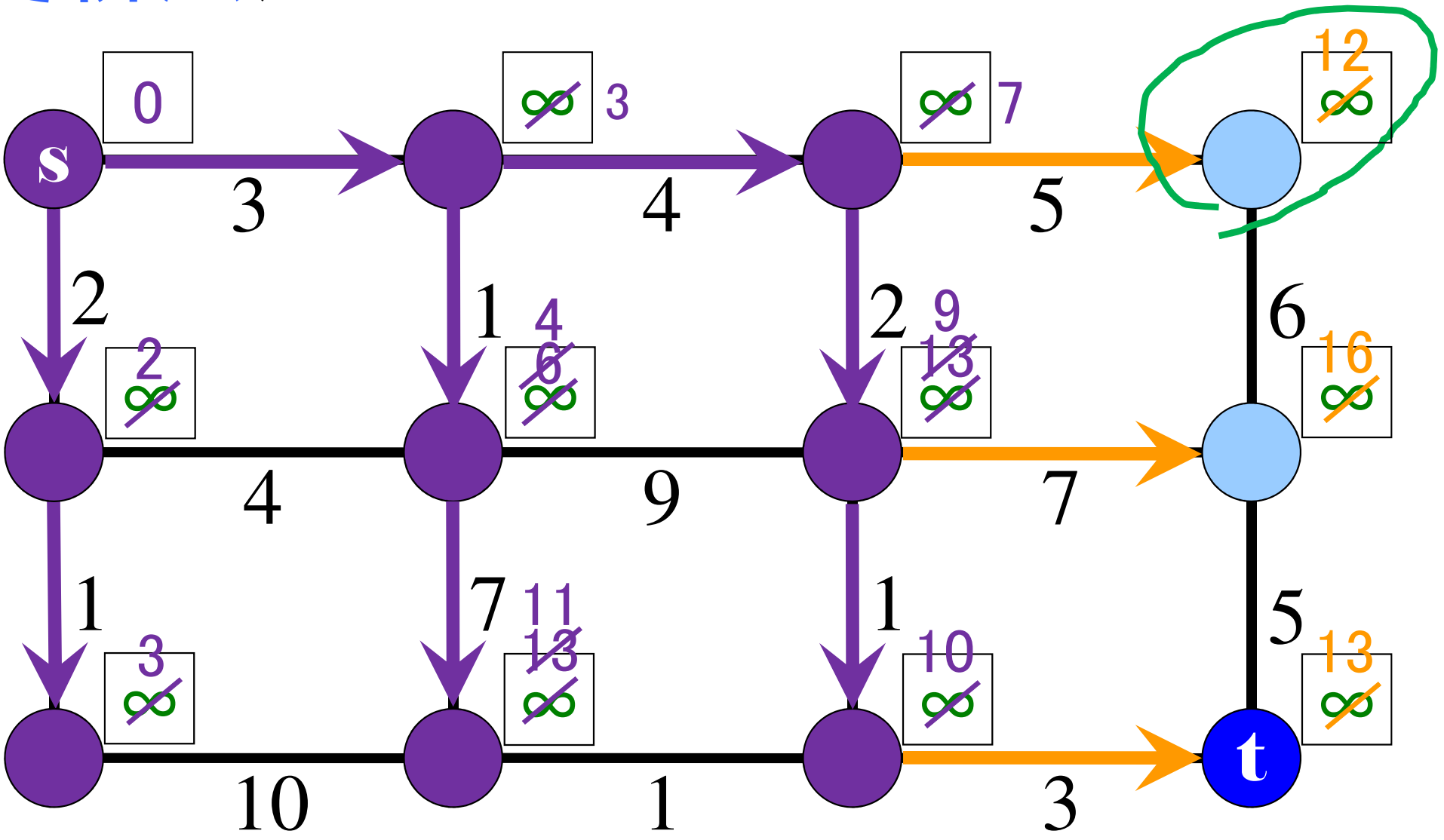
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



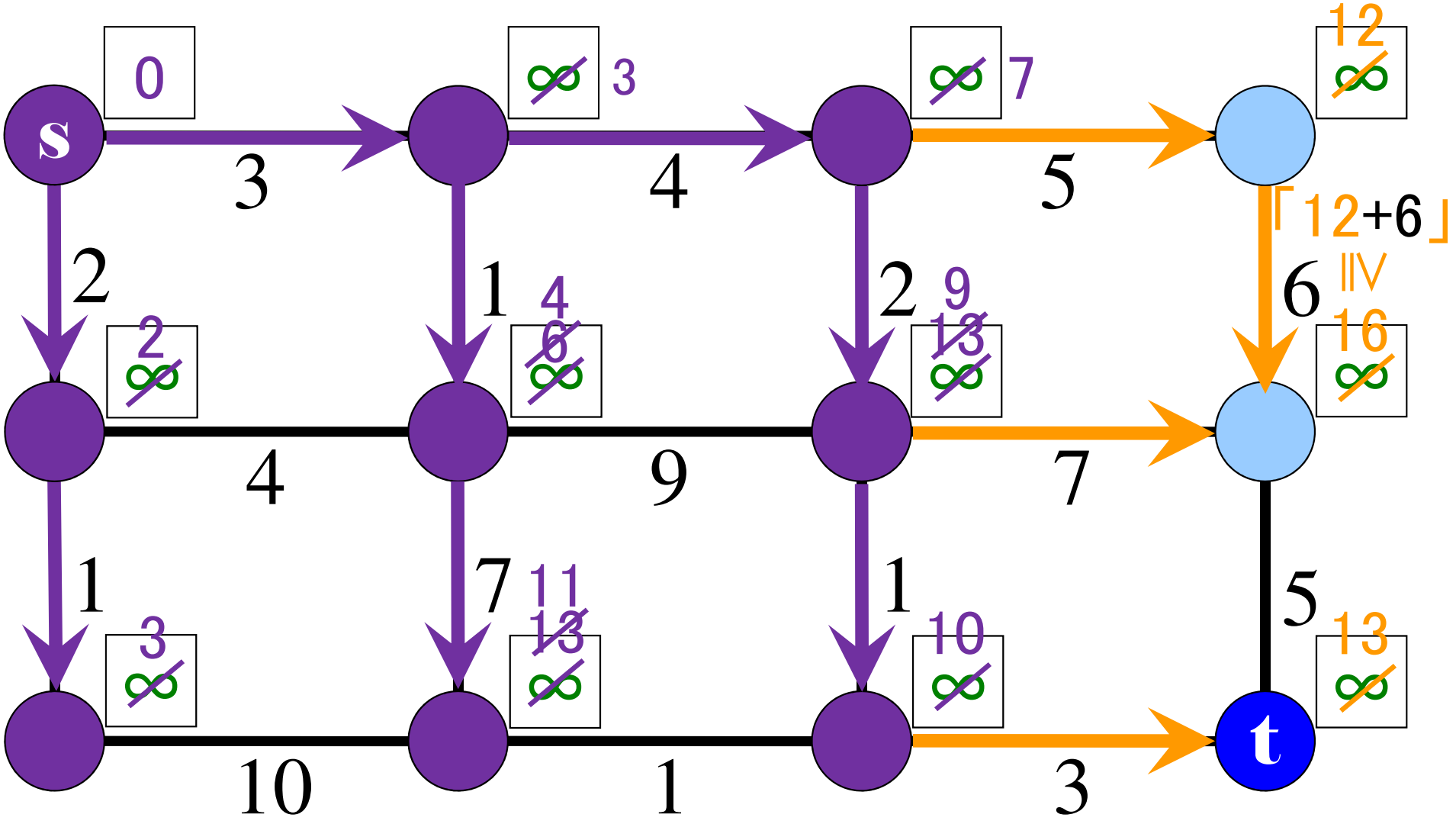
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

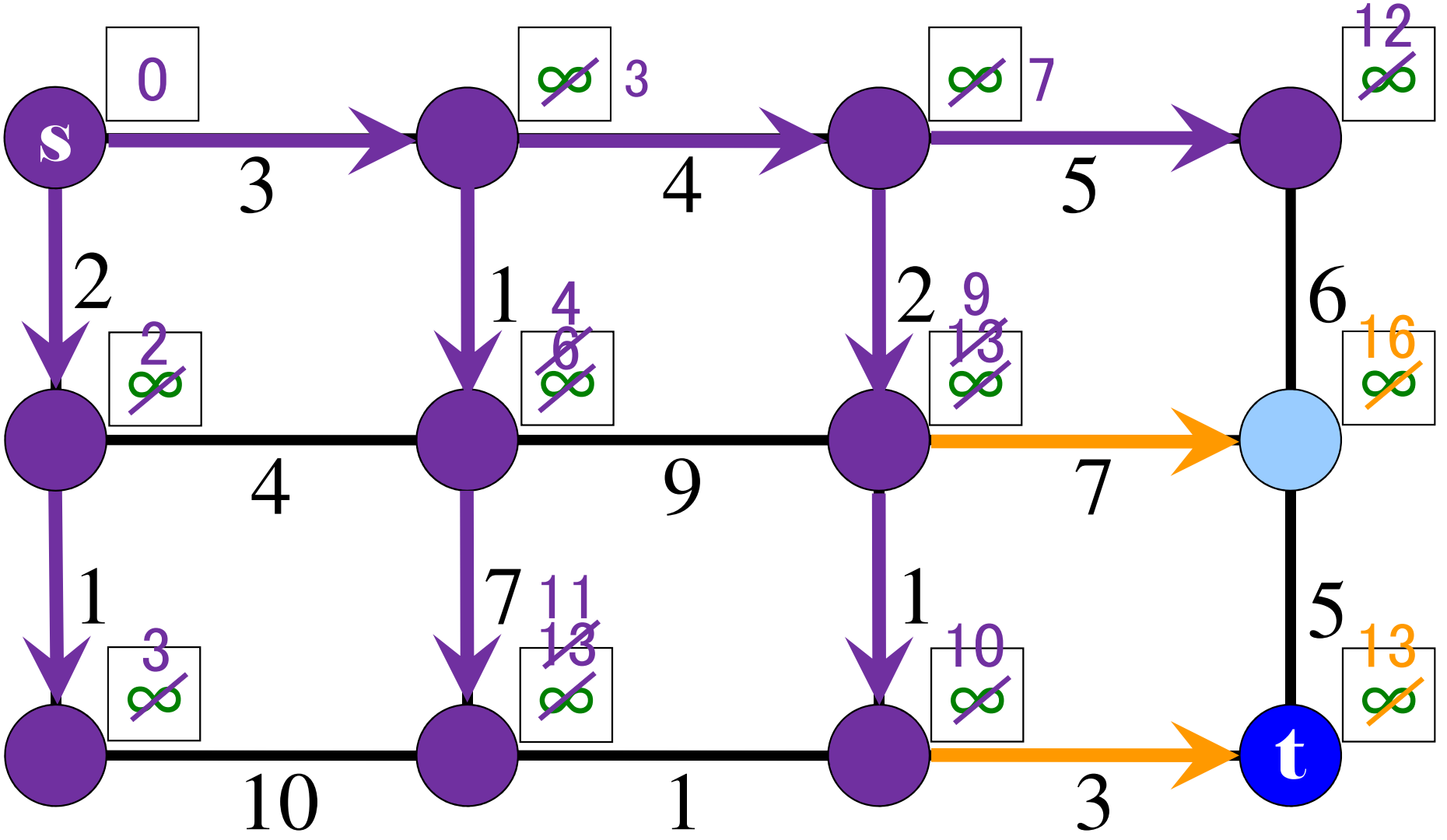


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

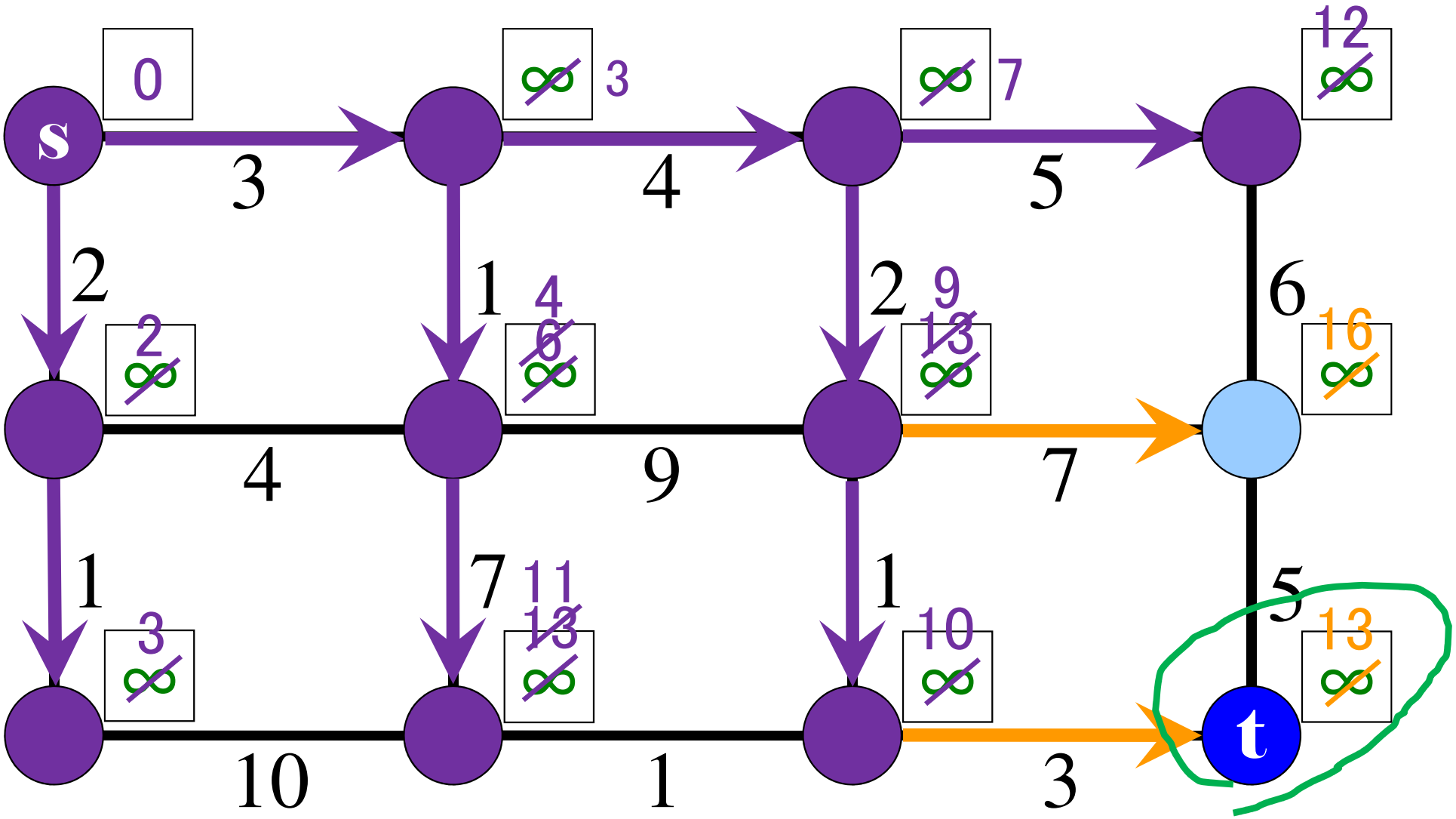
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



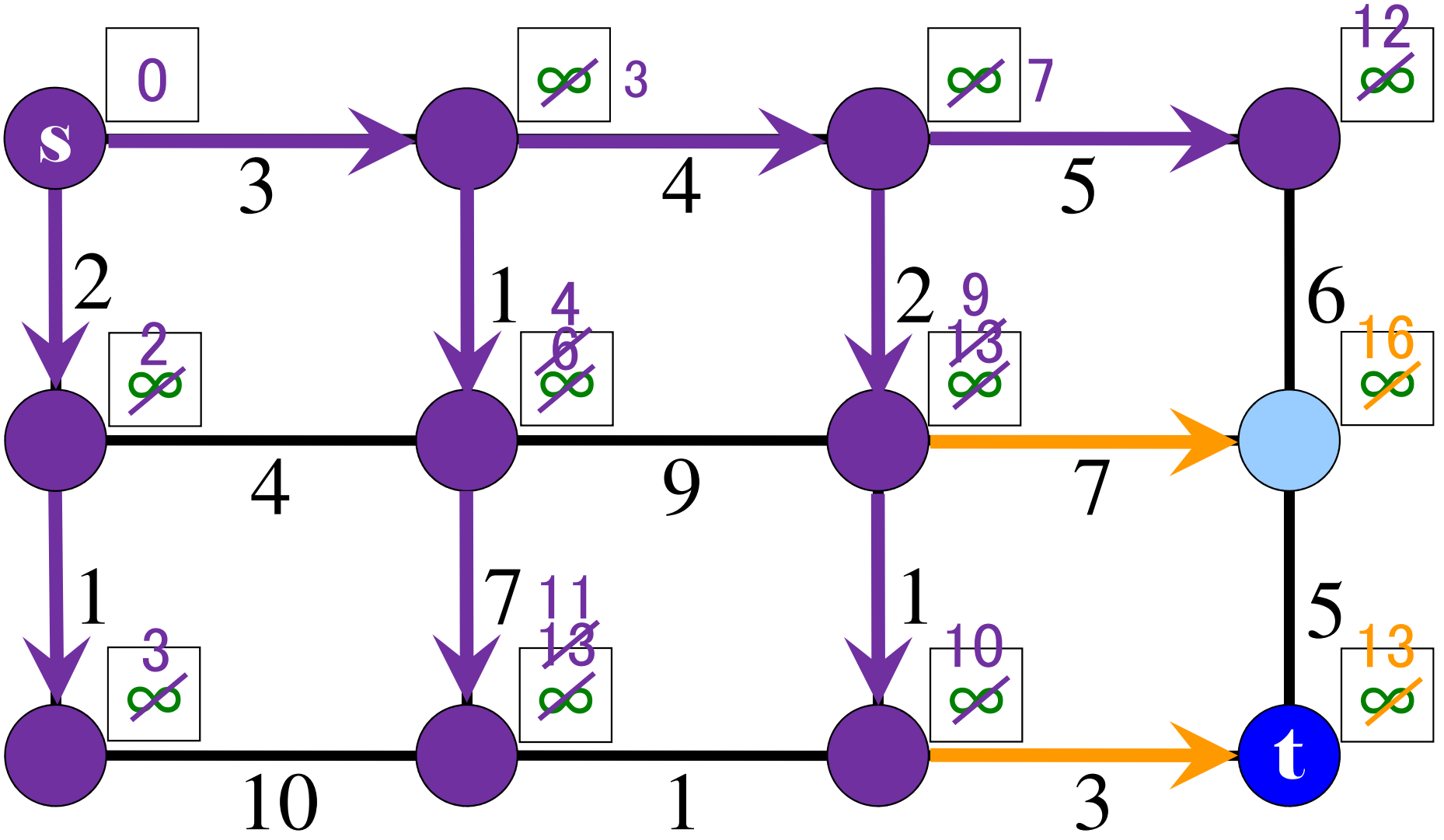
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

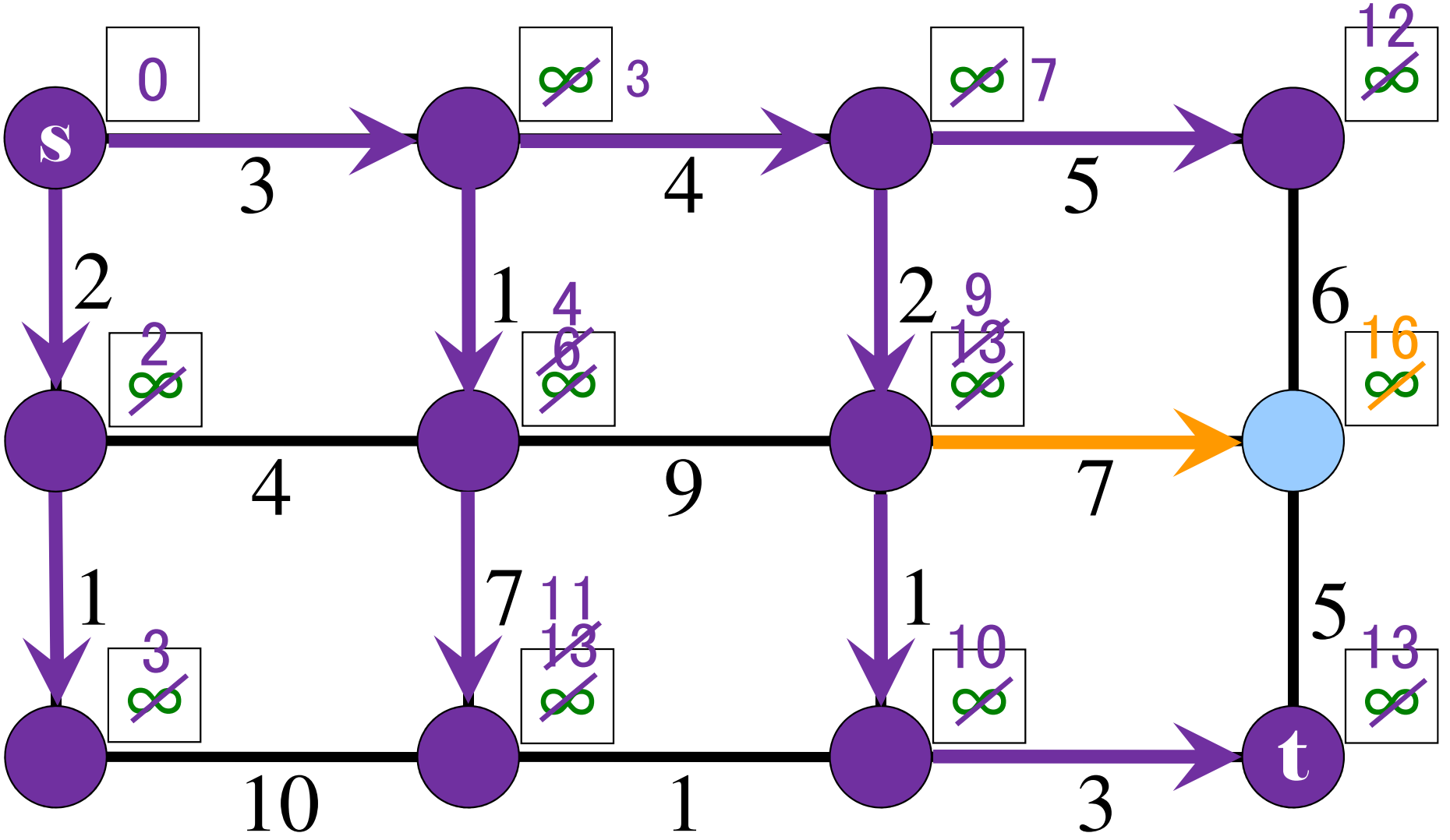


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

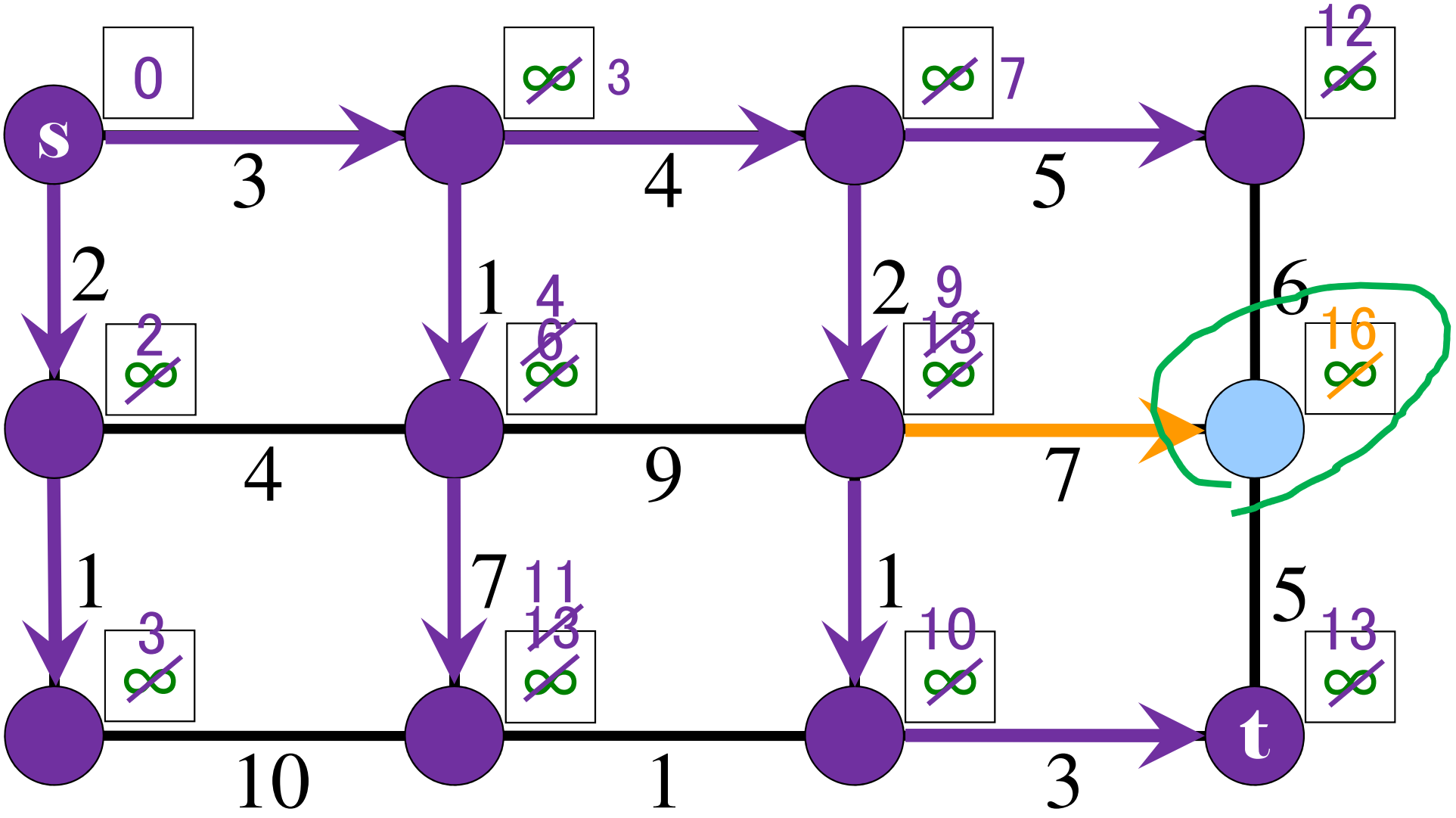
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



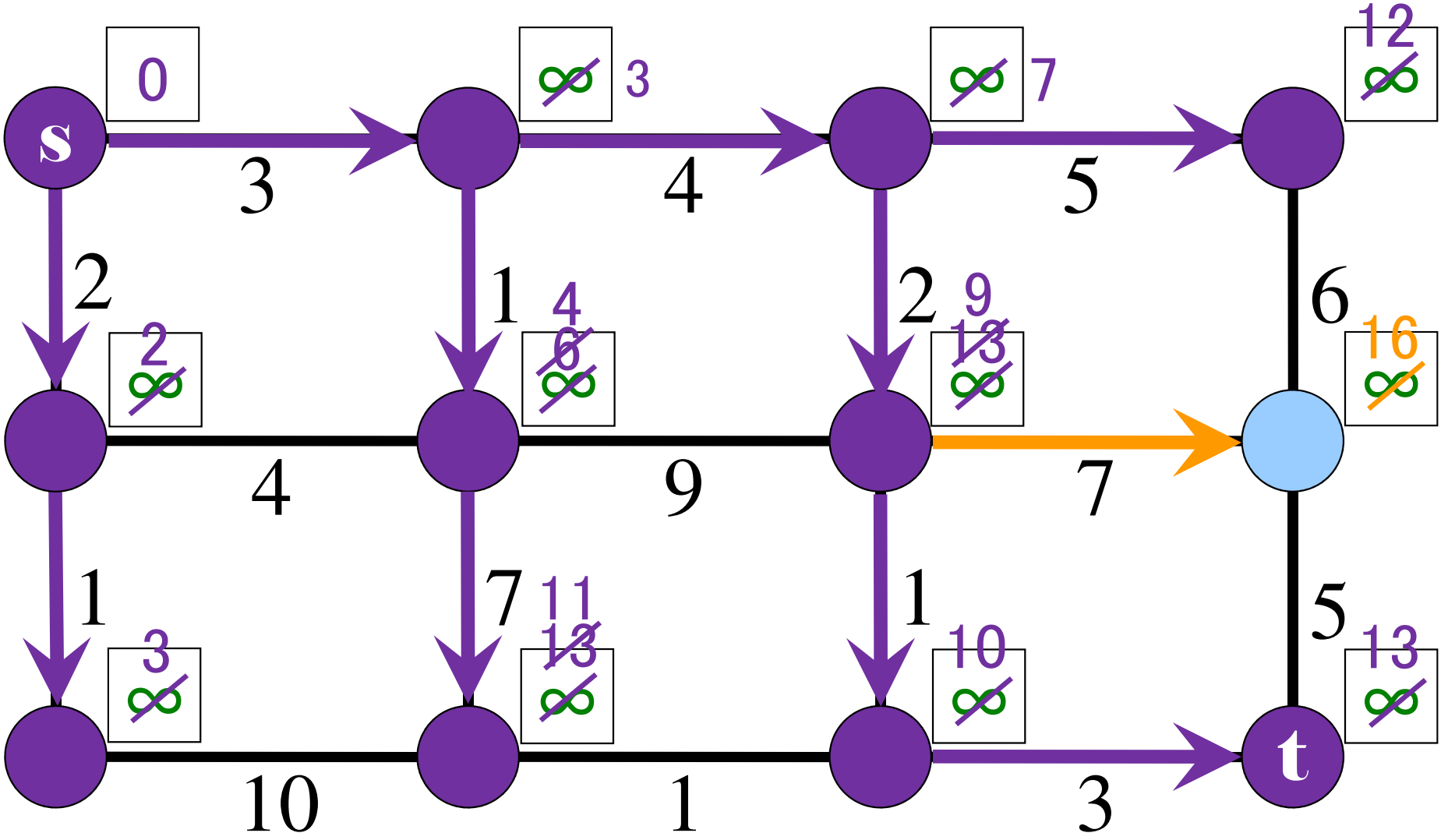
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

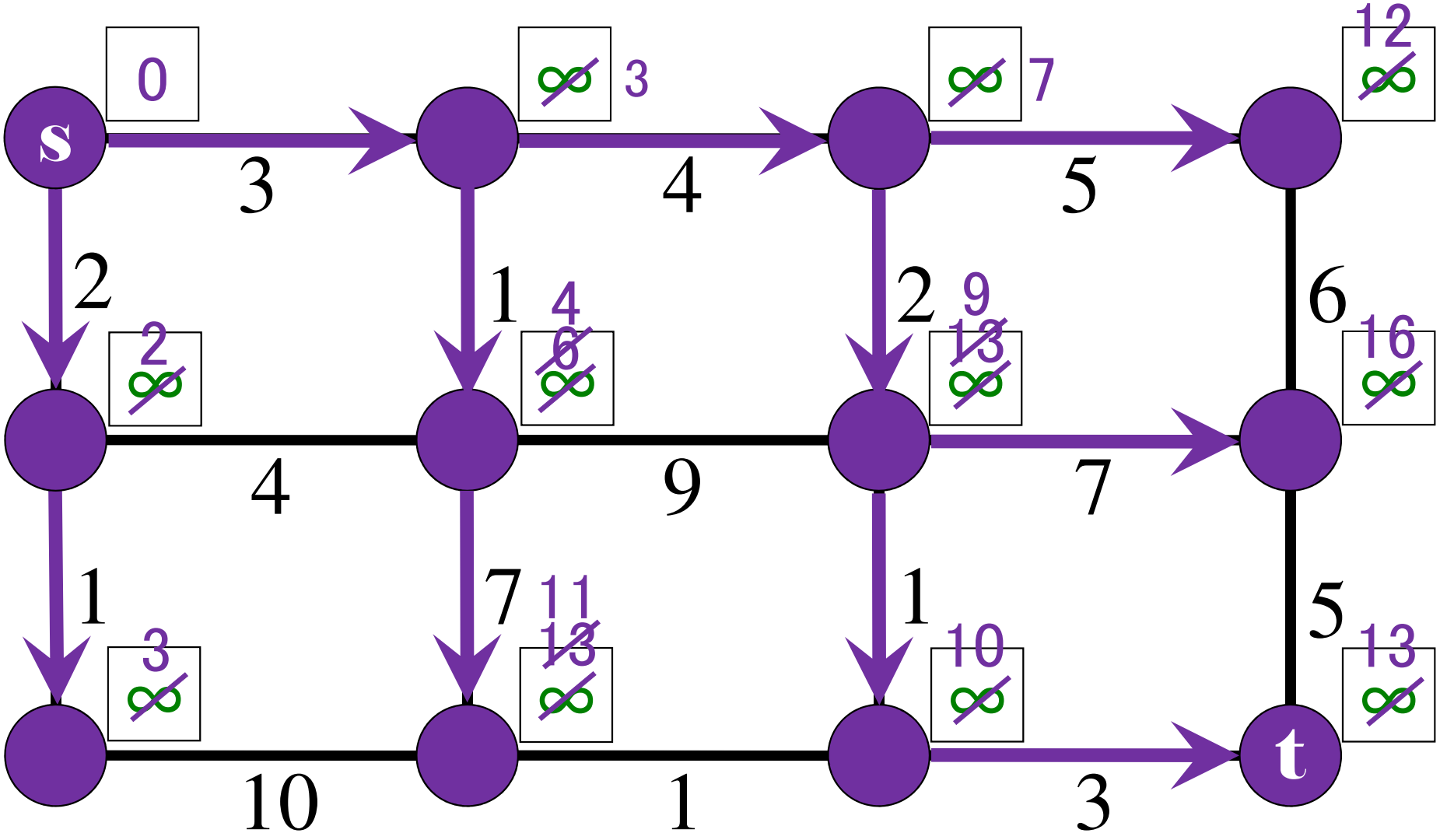


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

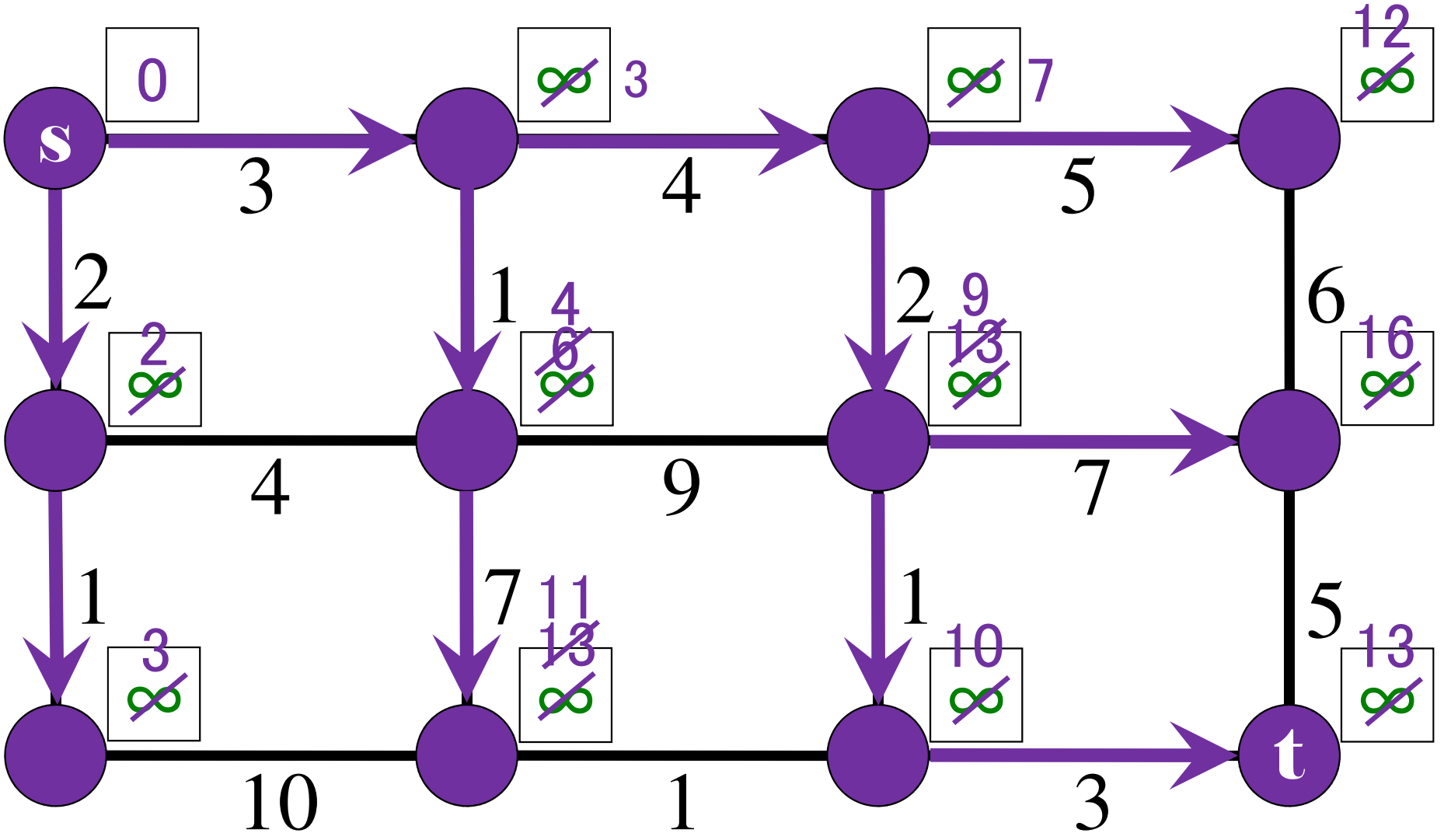
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



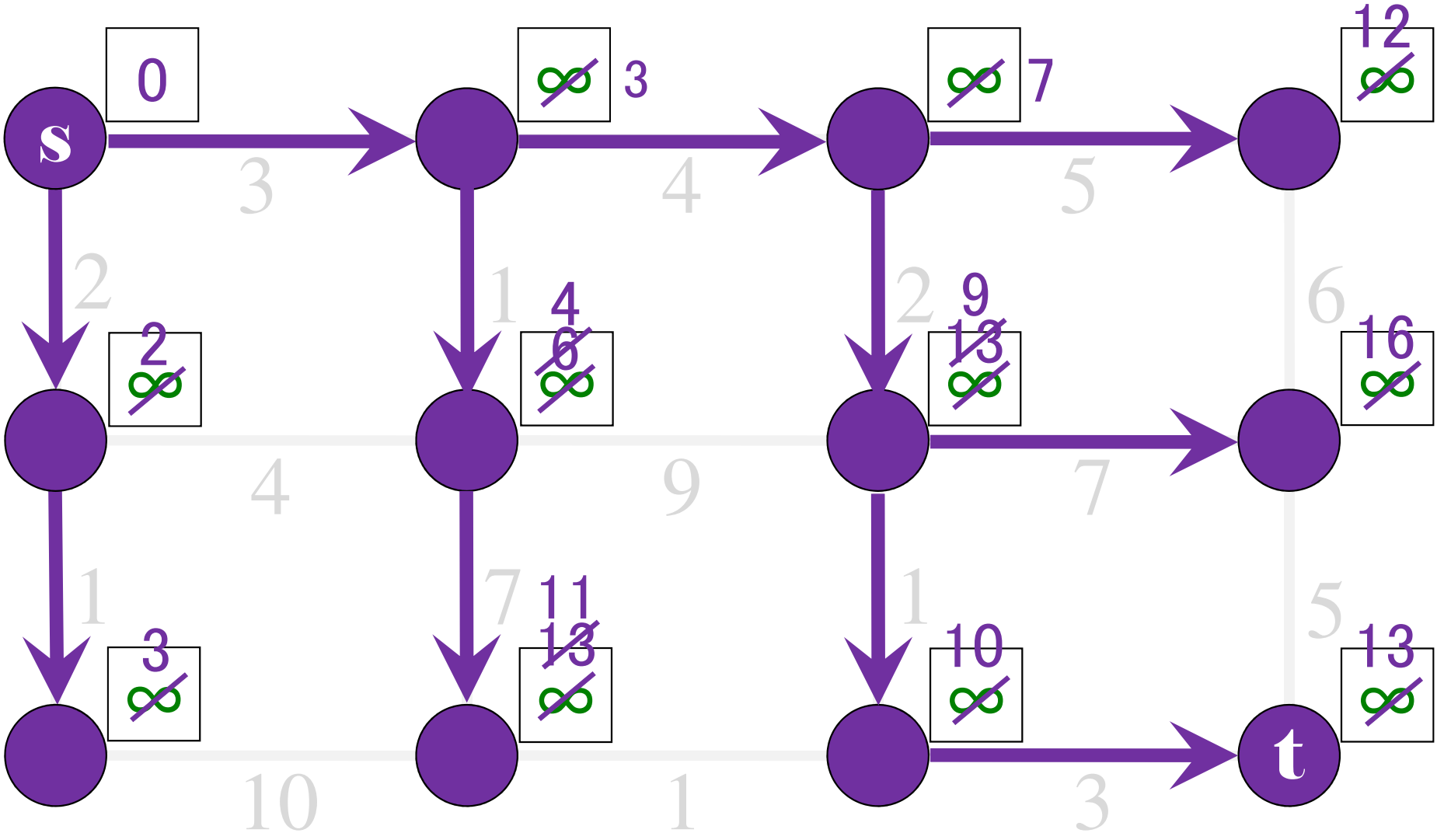
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



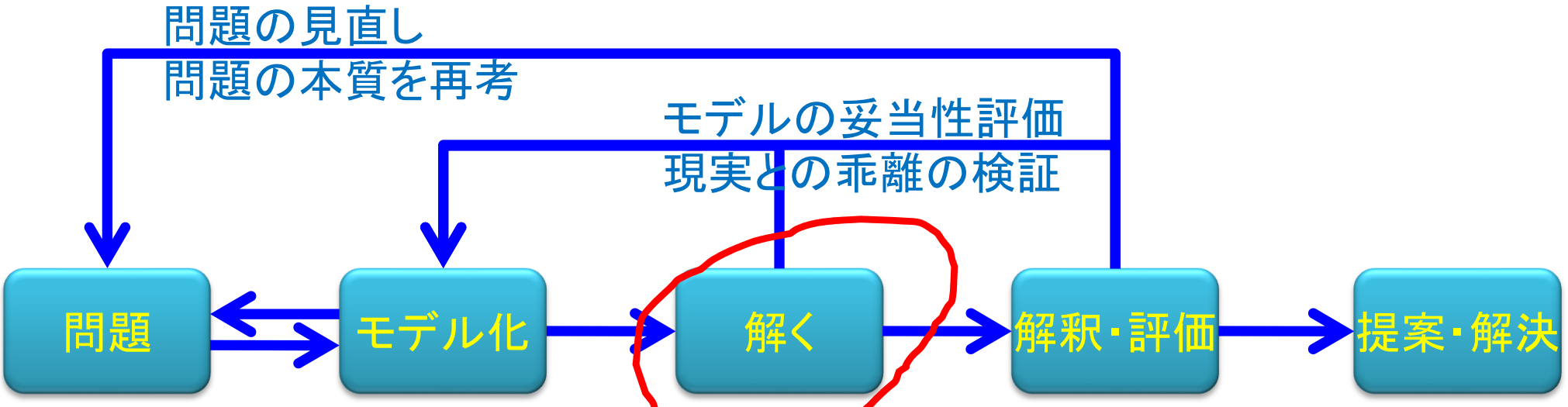
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

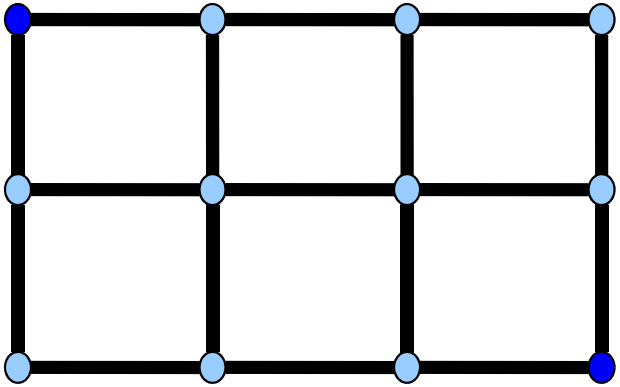
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

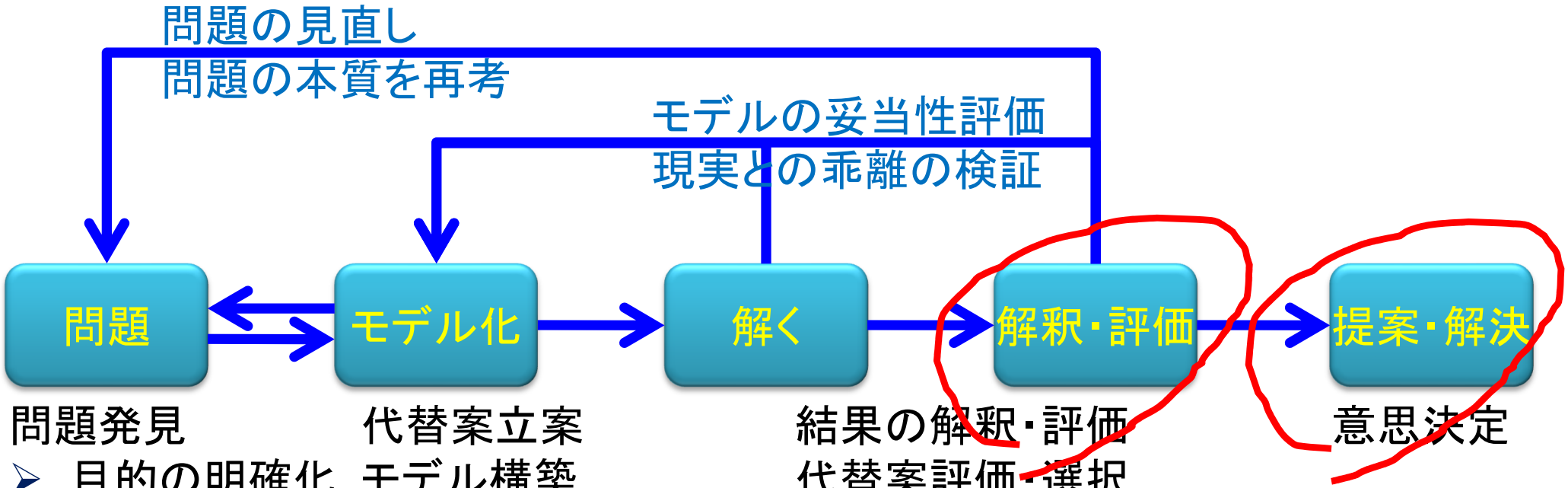
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

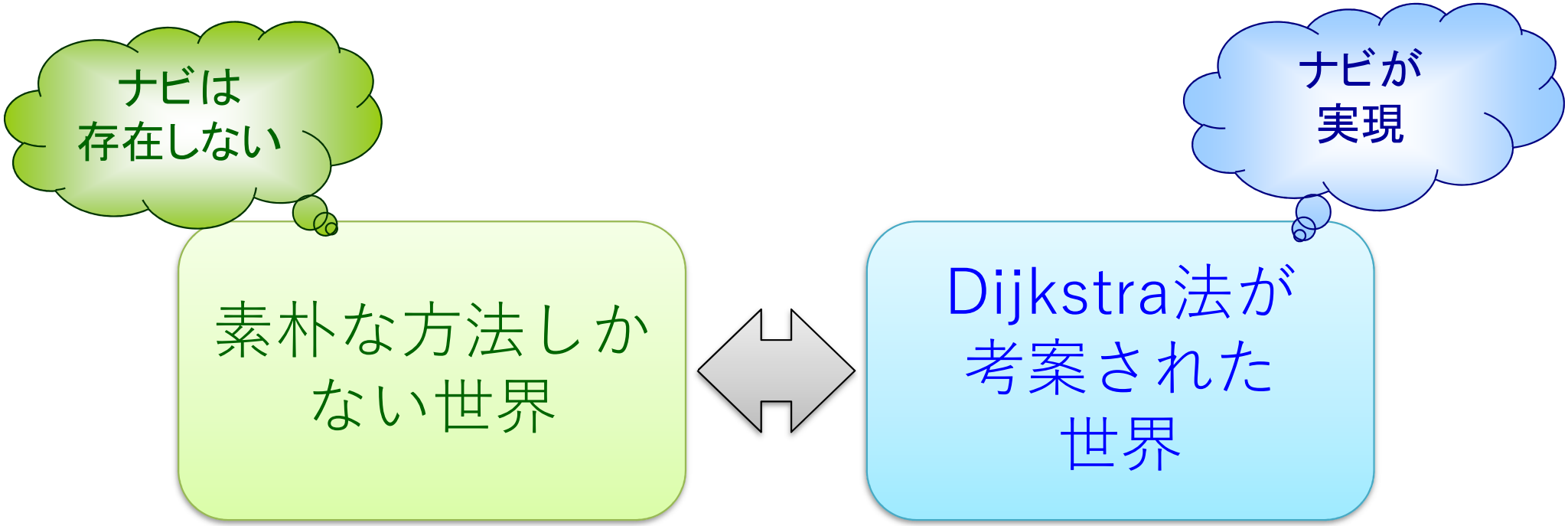
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

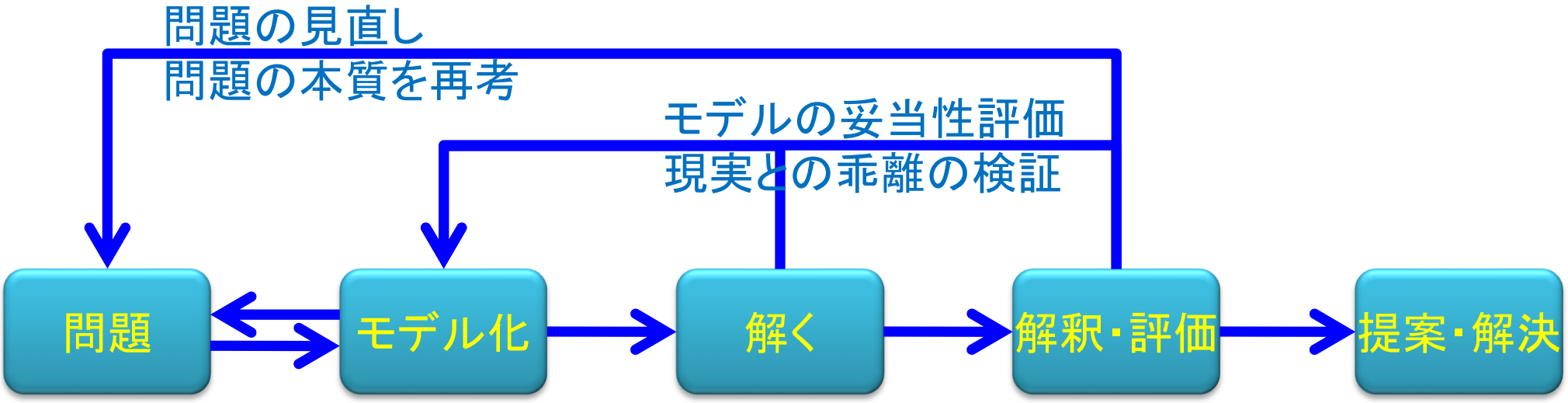
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



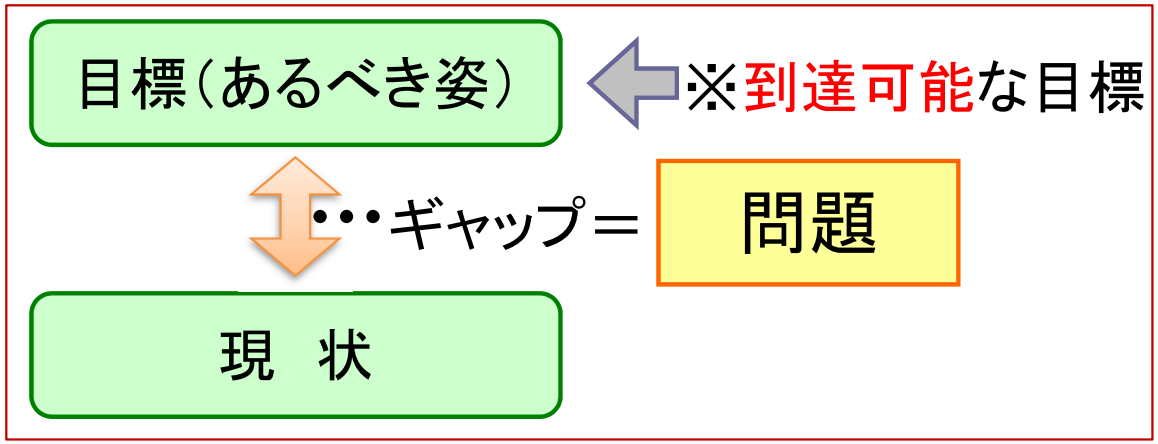
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

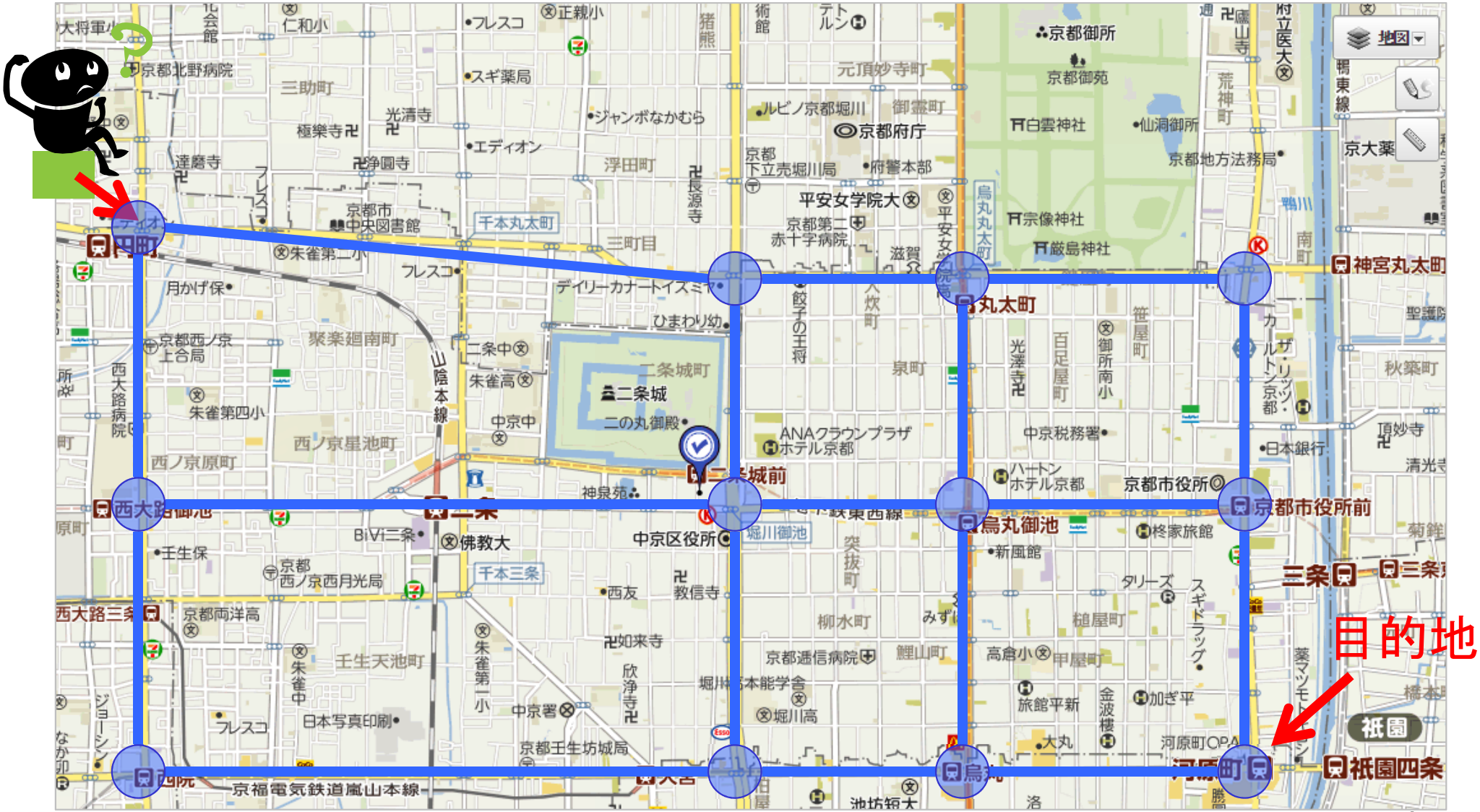
意思決定

問題の定義



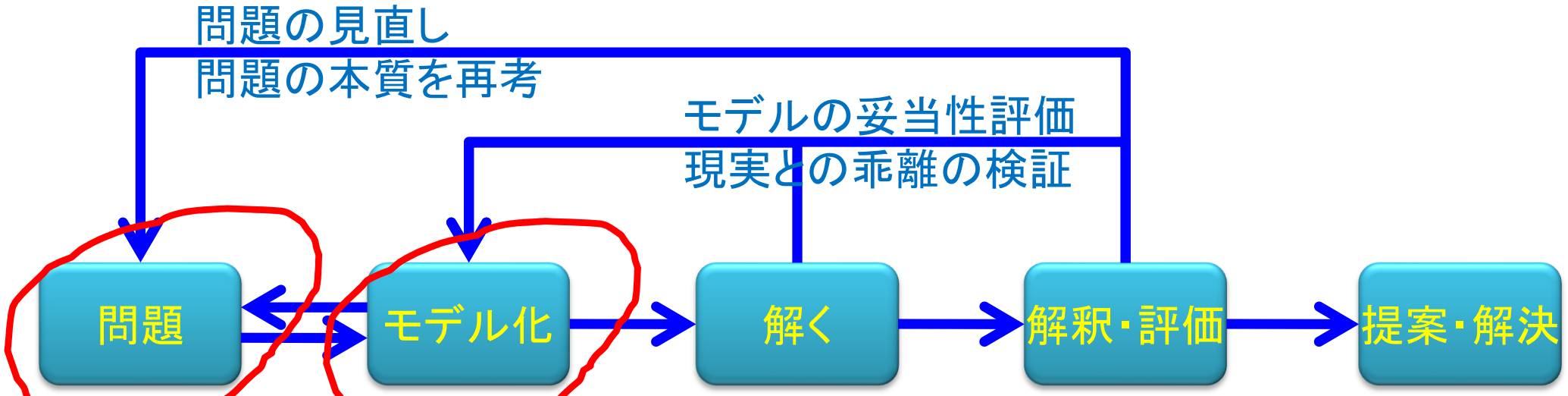
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

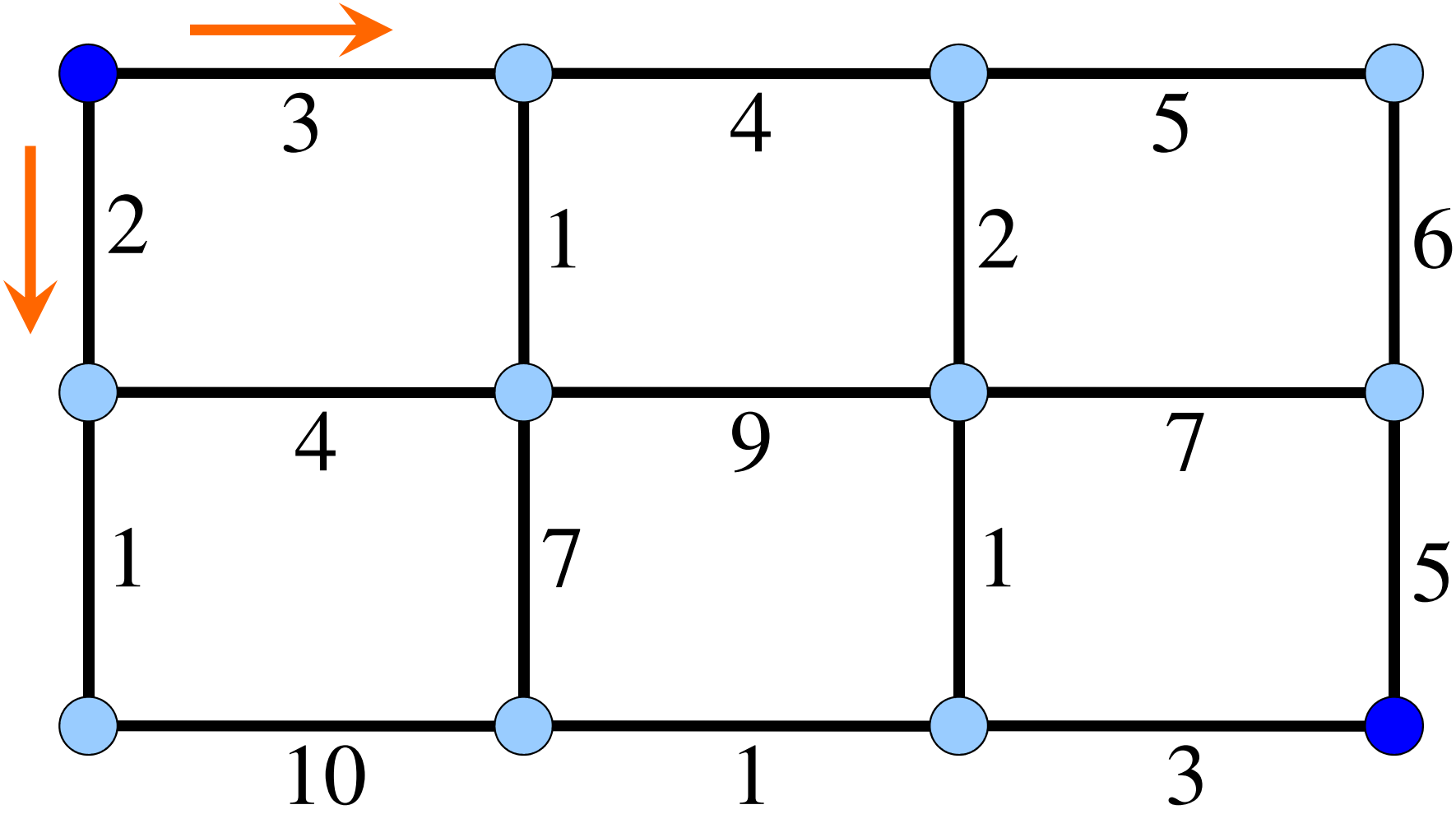
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

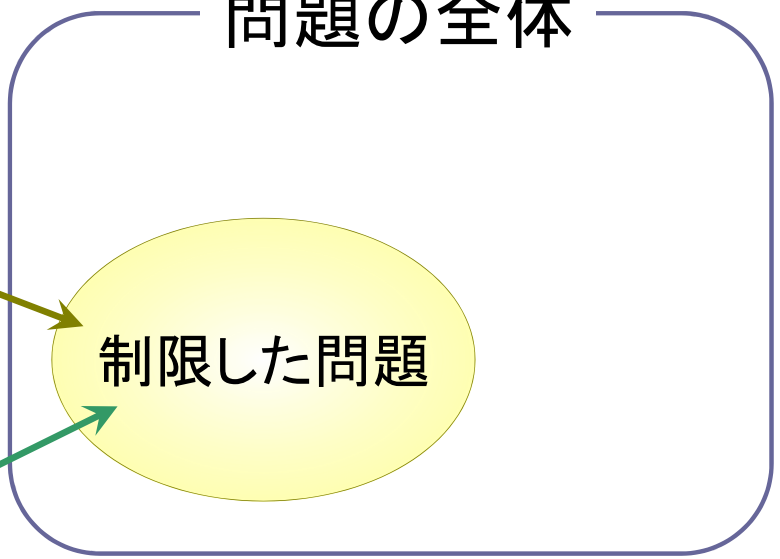


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

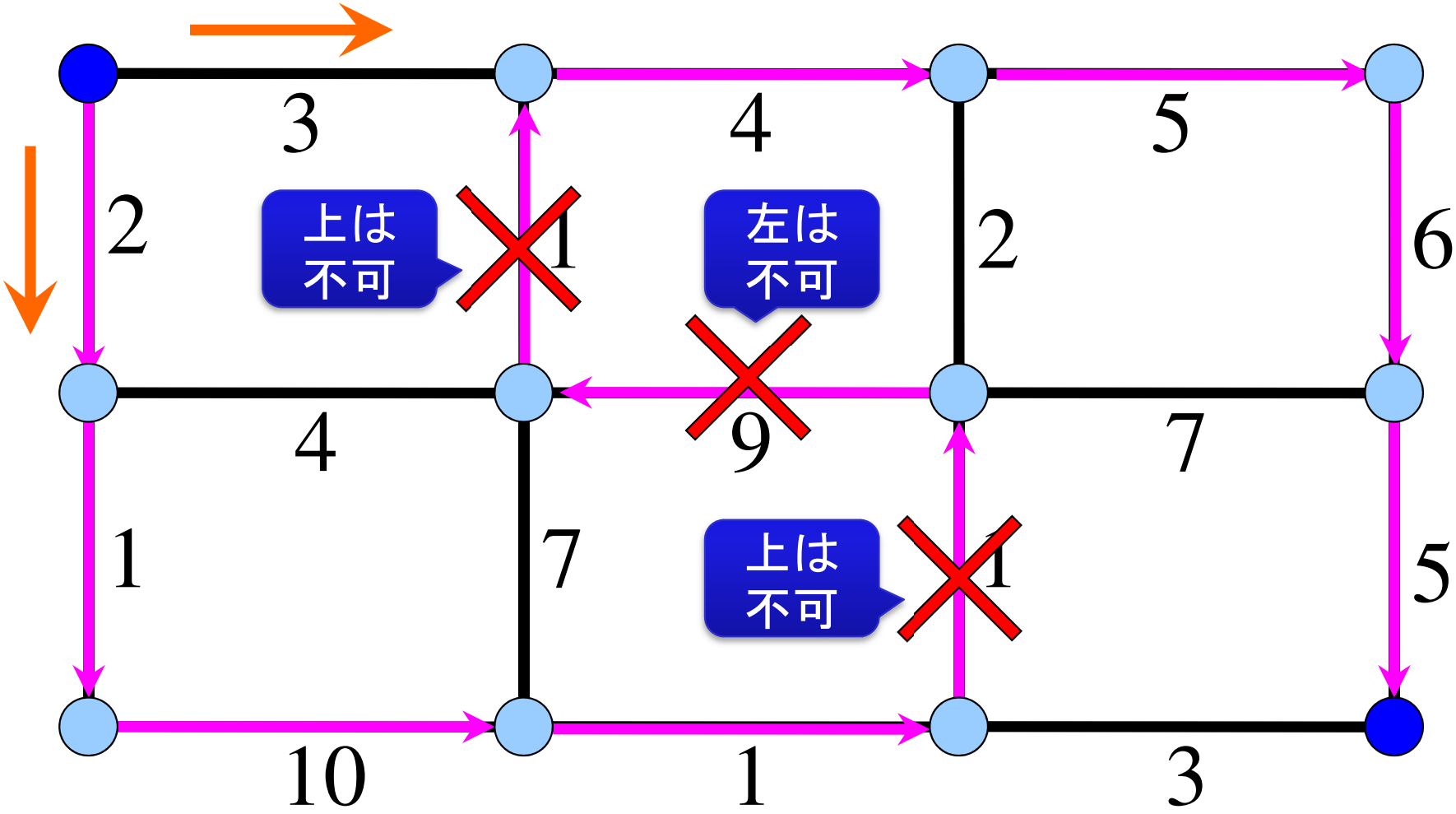
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

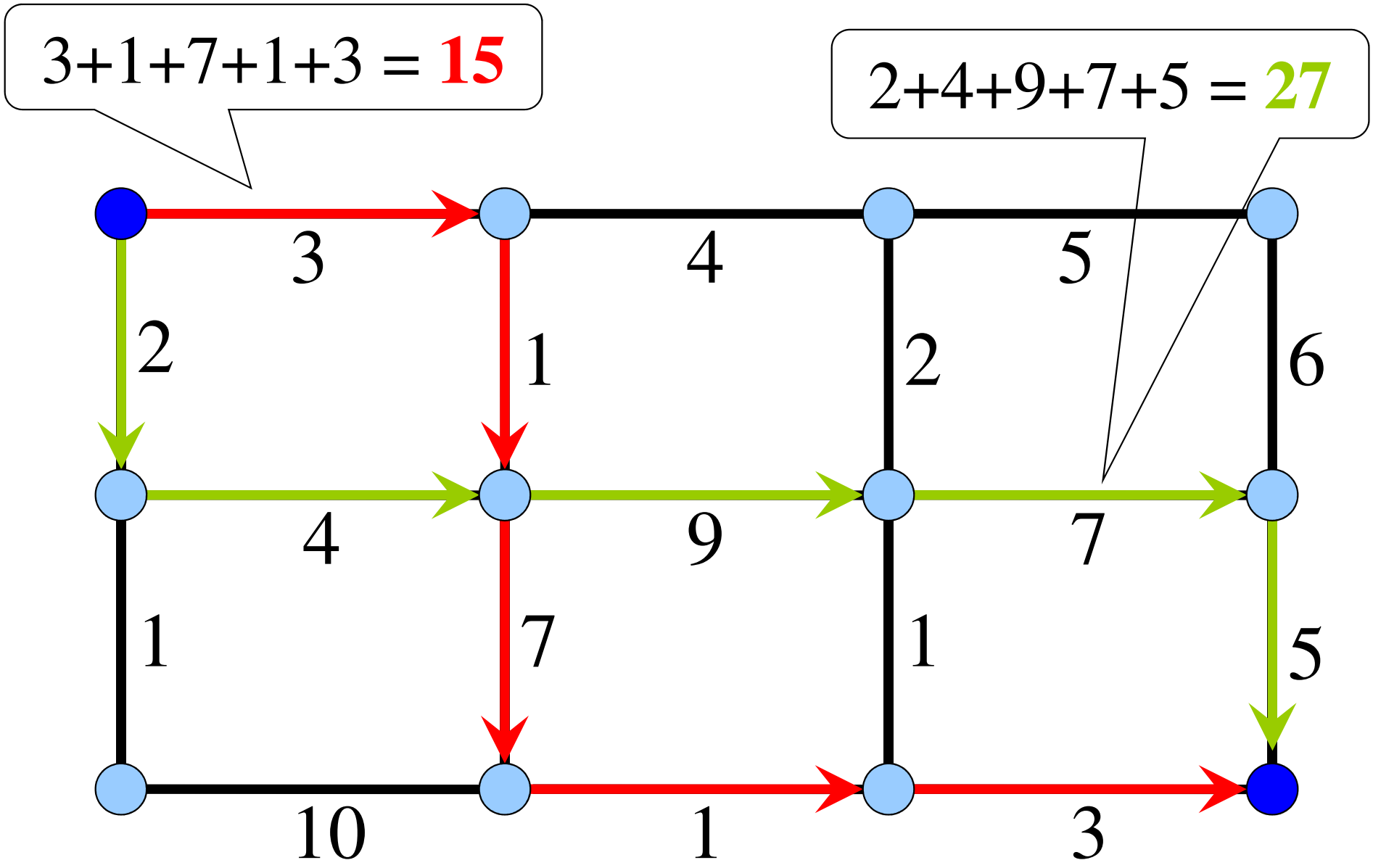
難しいなら易しくすればいいのさ！

制限した問題

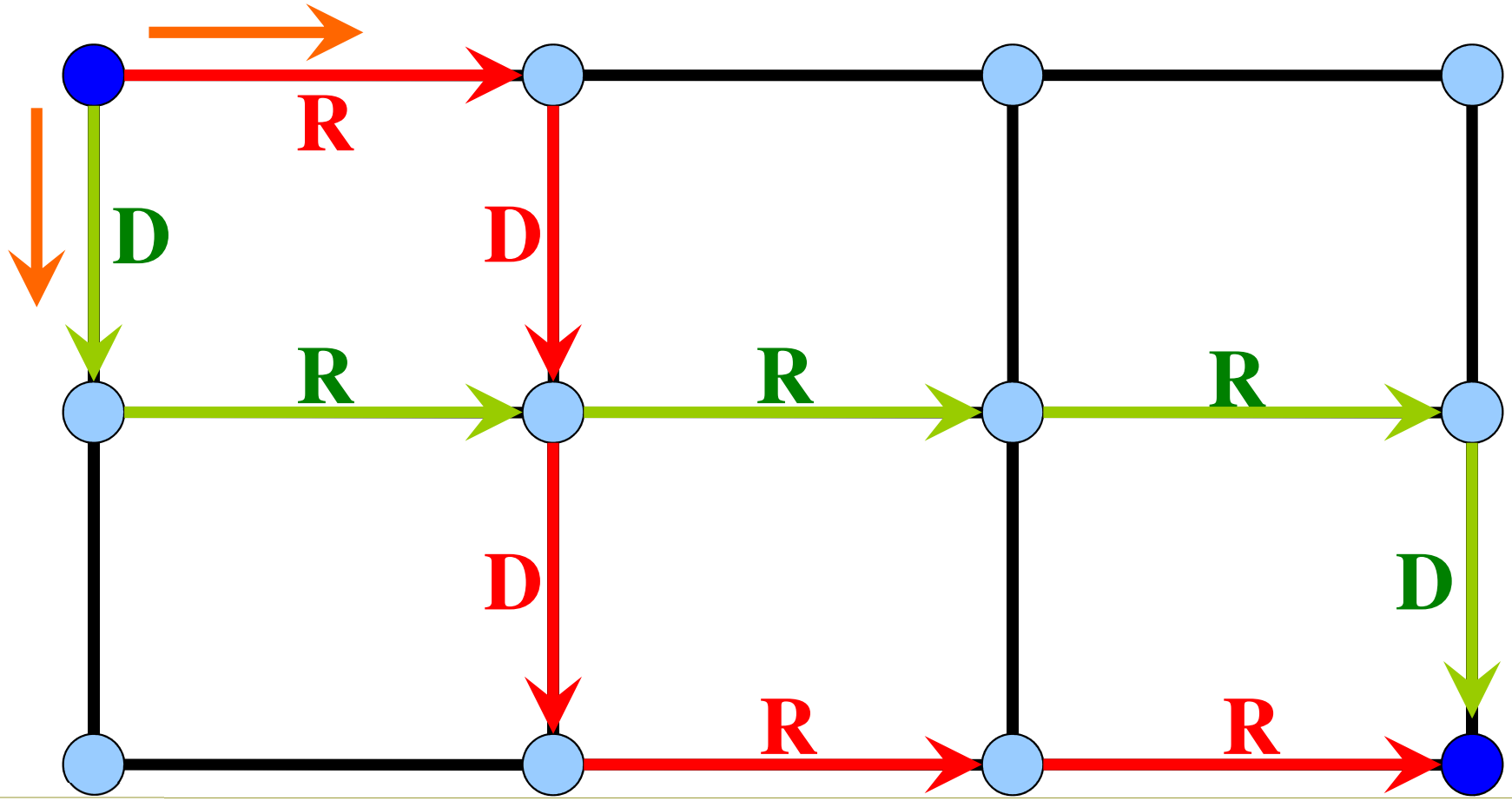
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



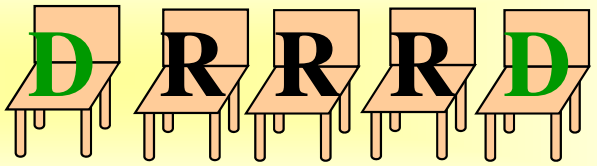
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



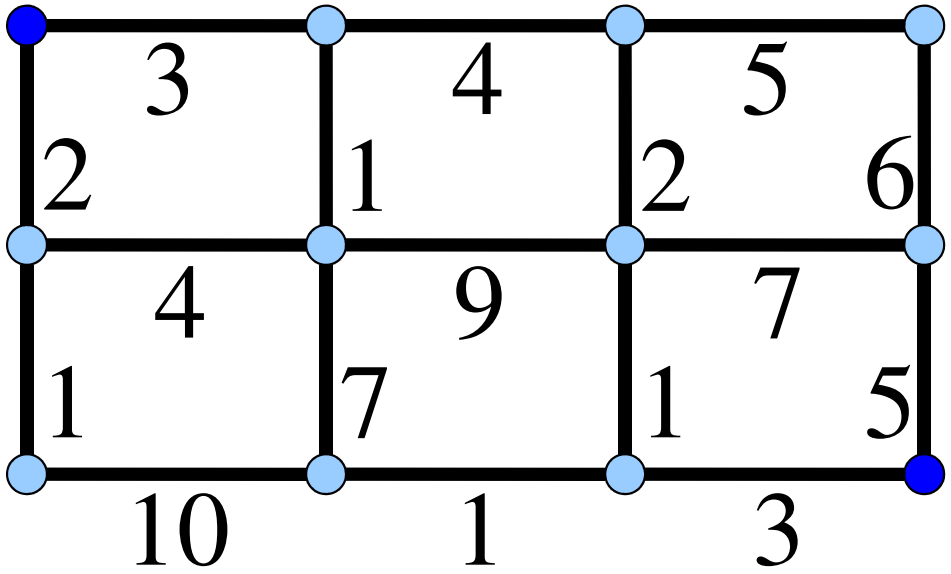
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

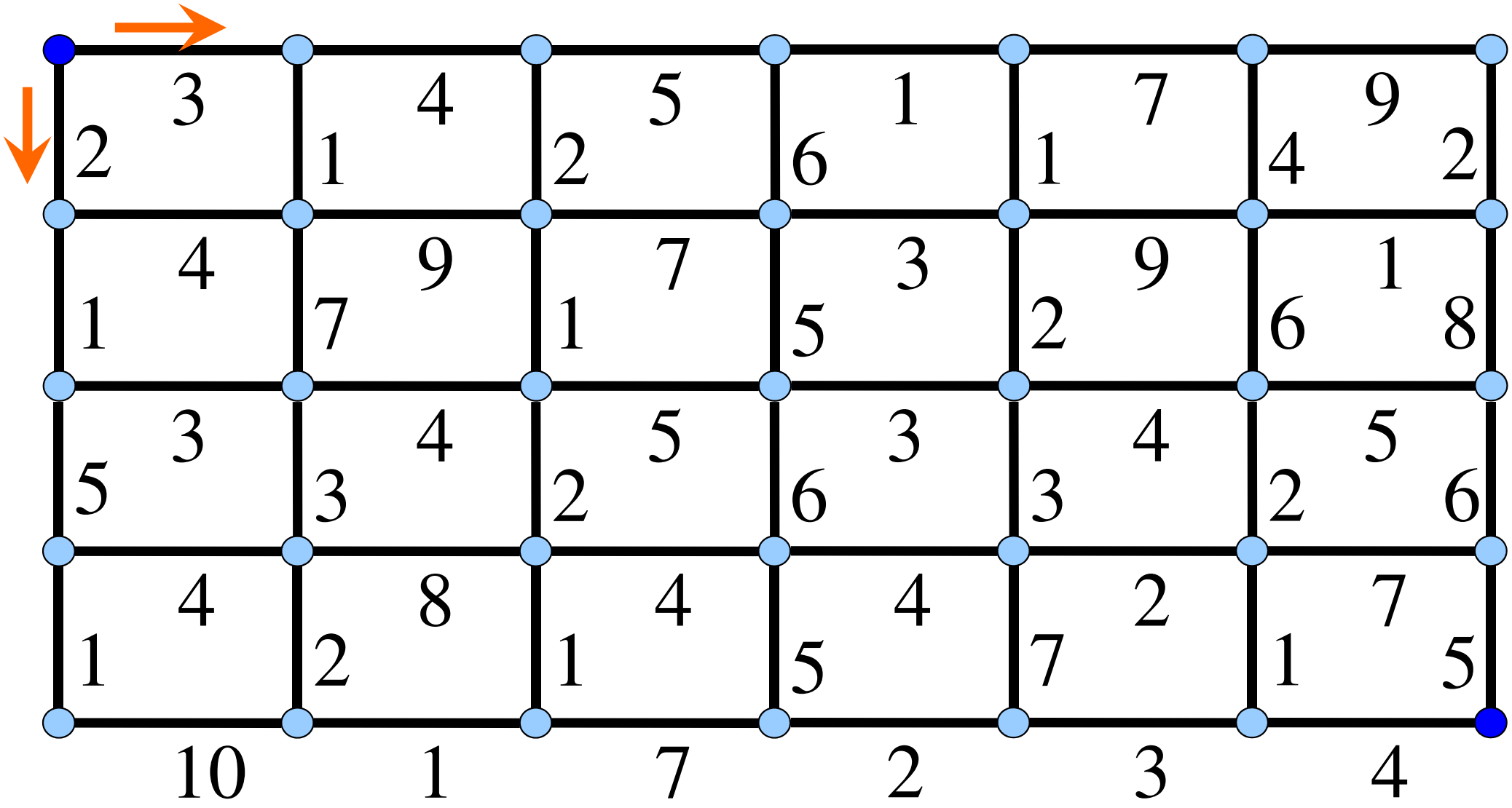
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

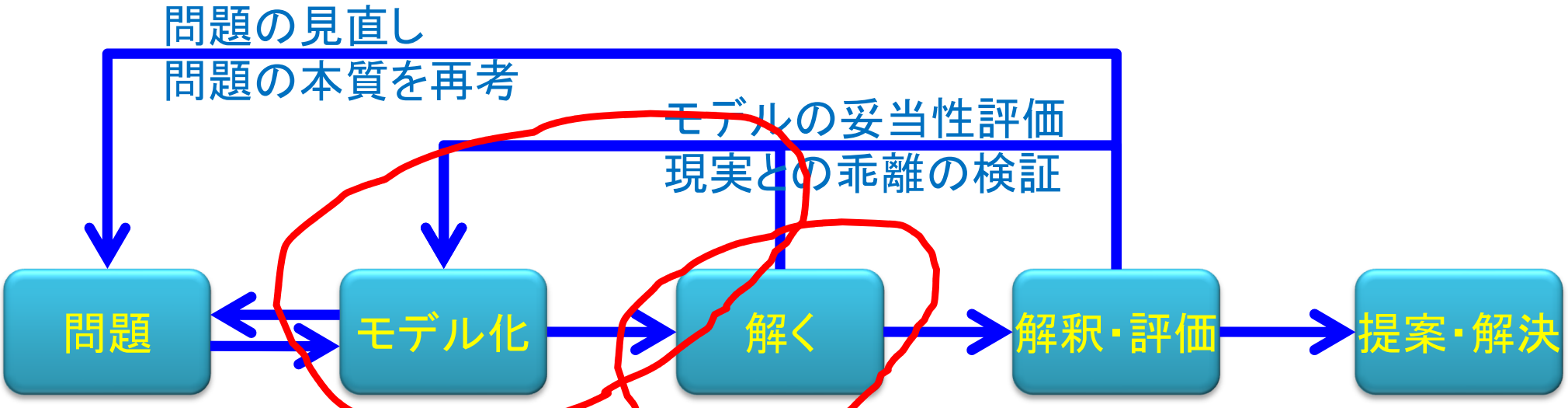
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

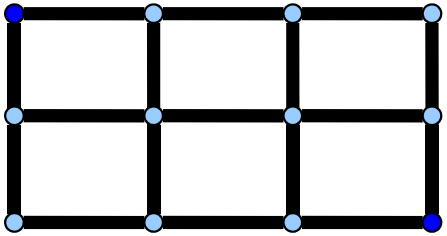
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

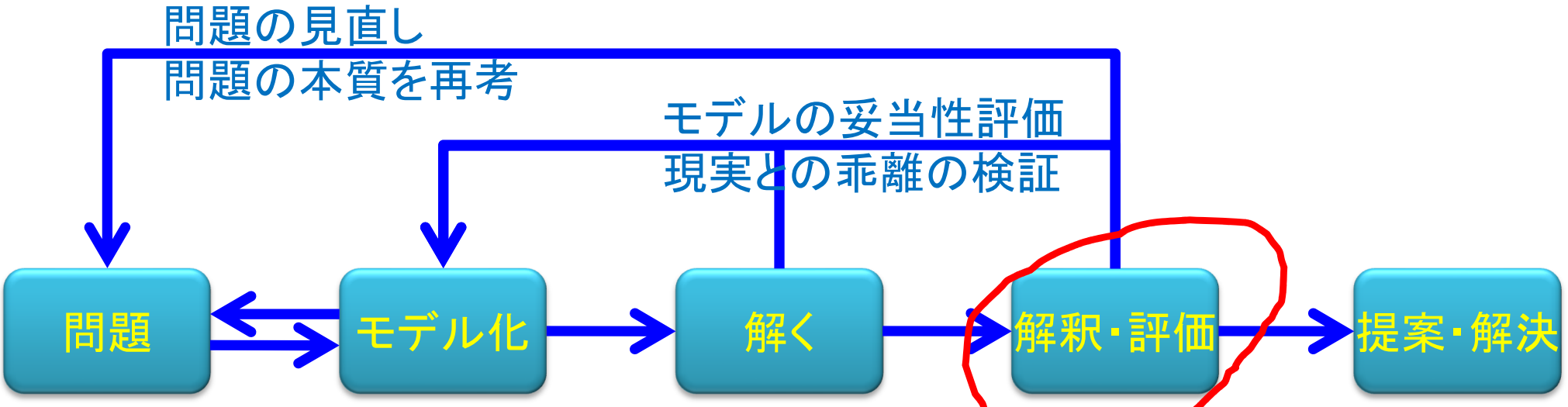
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

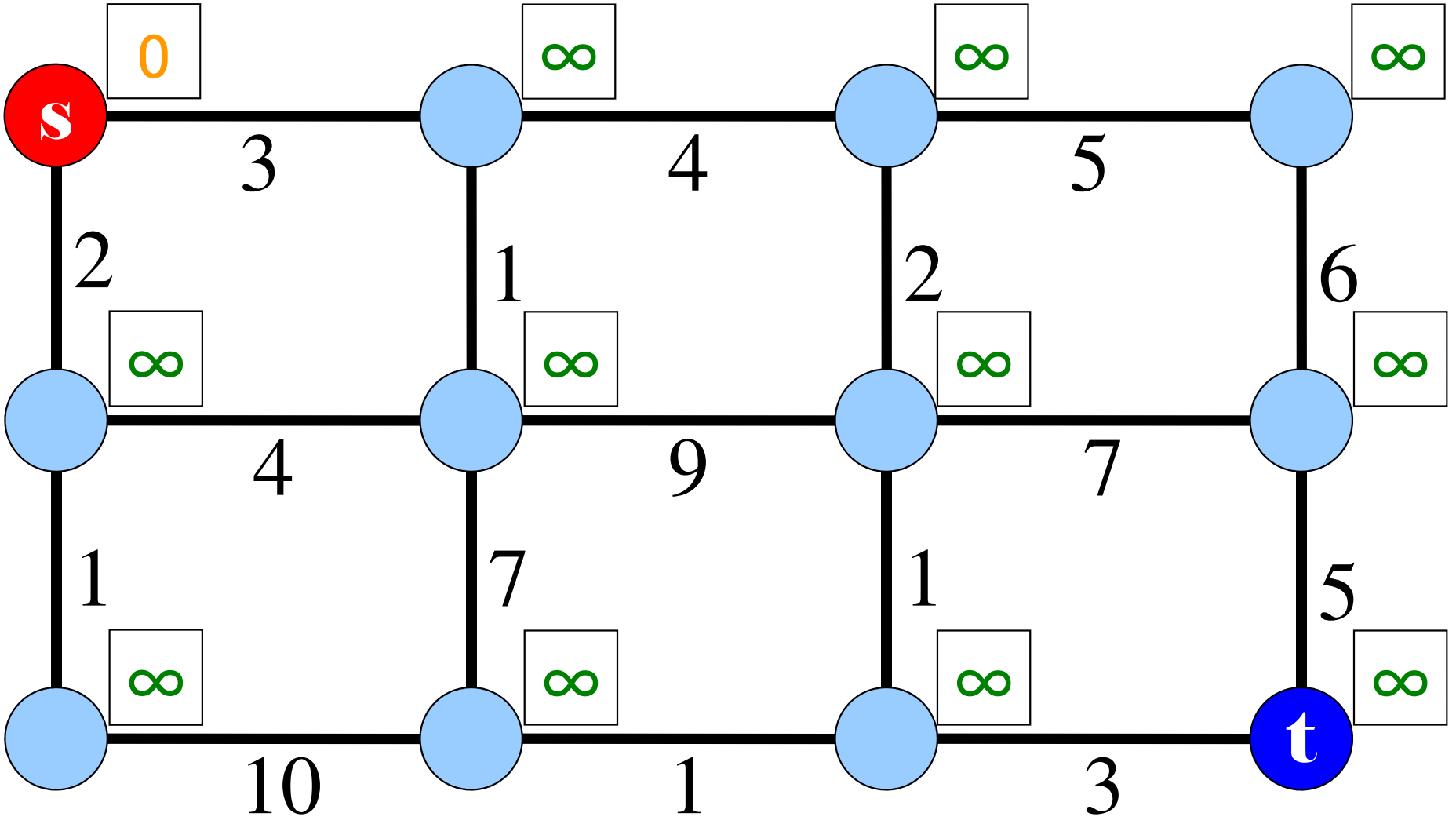
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

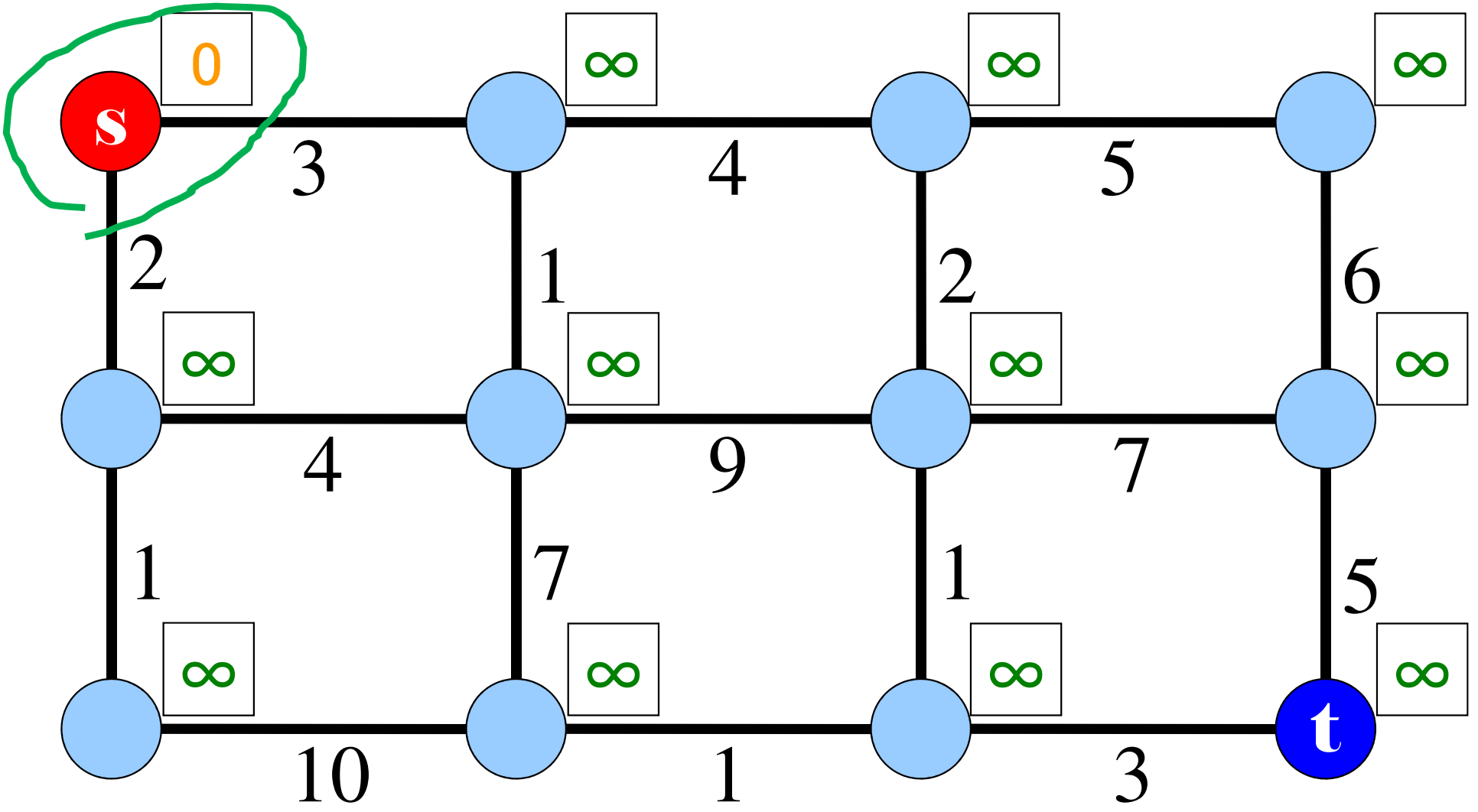
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



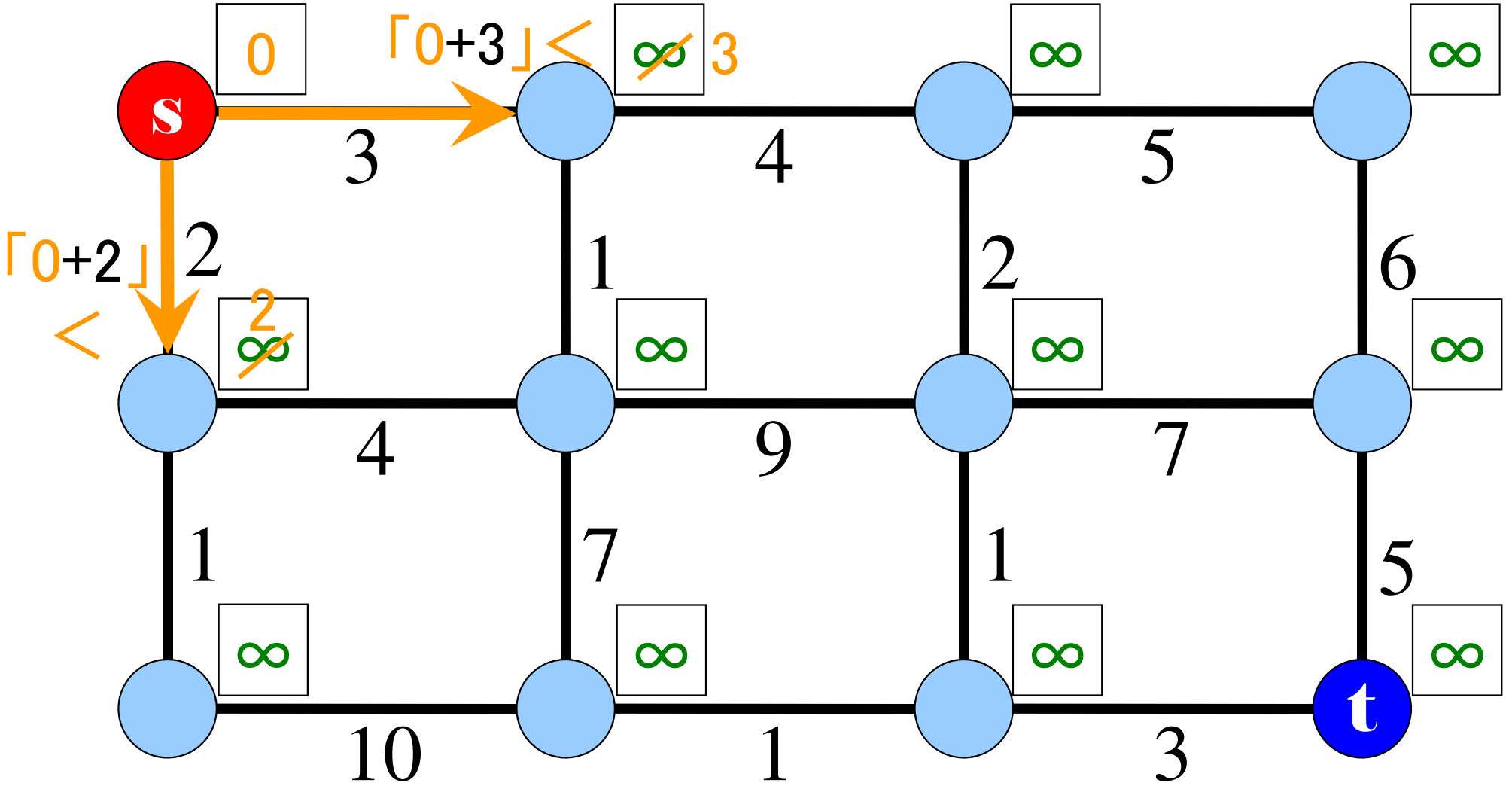
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

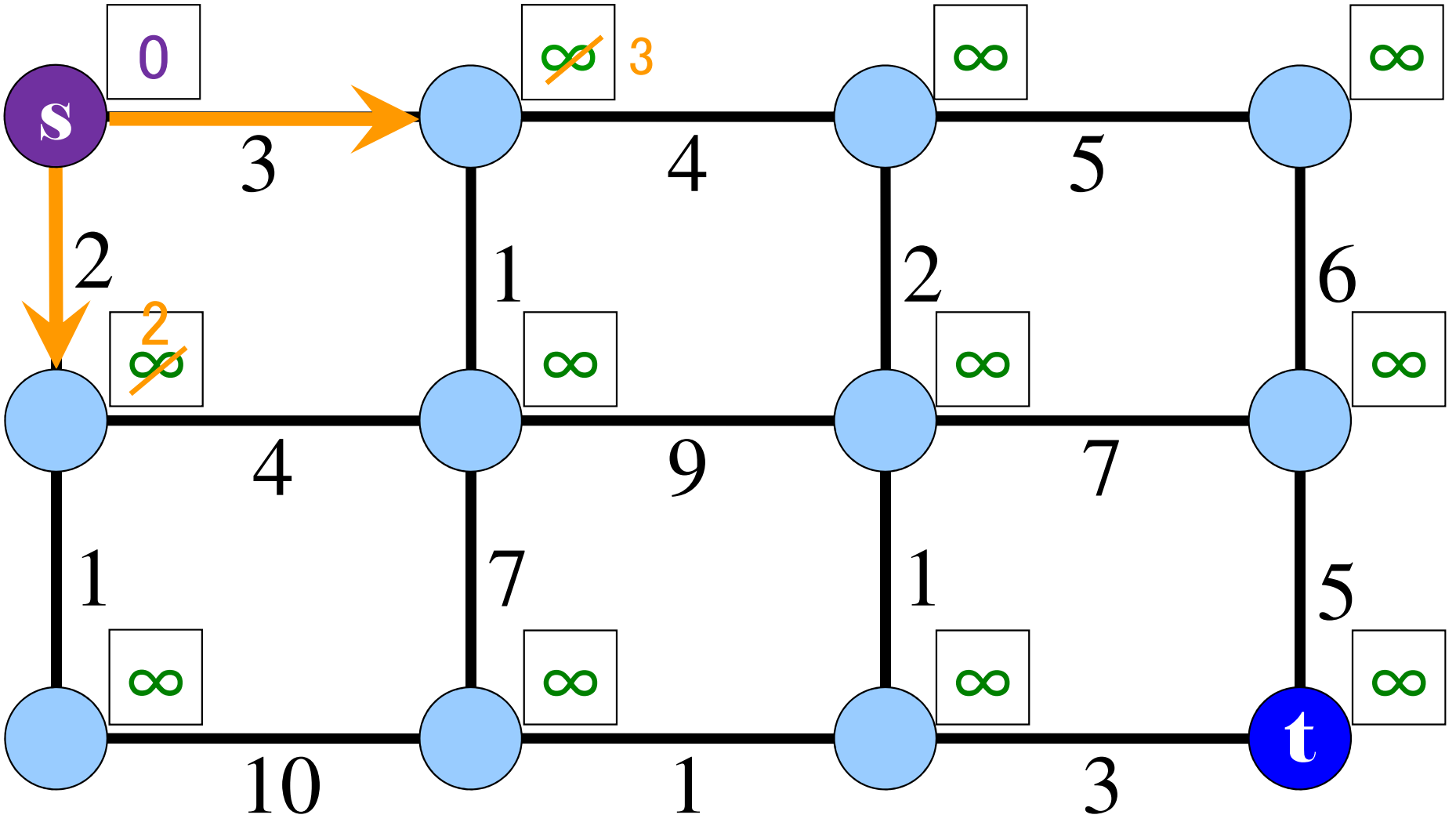


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

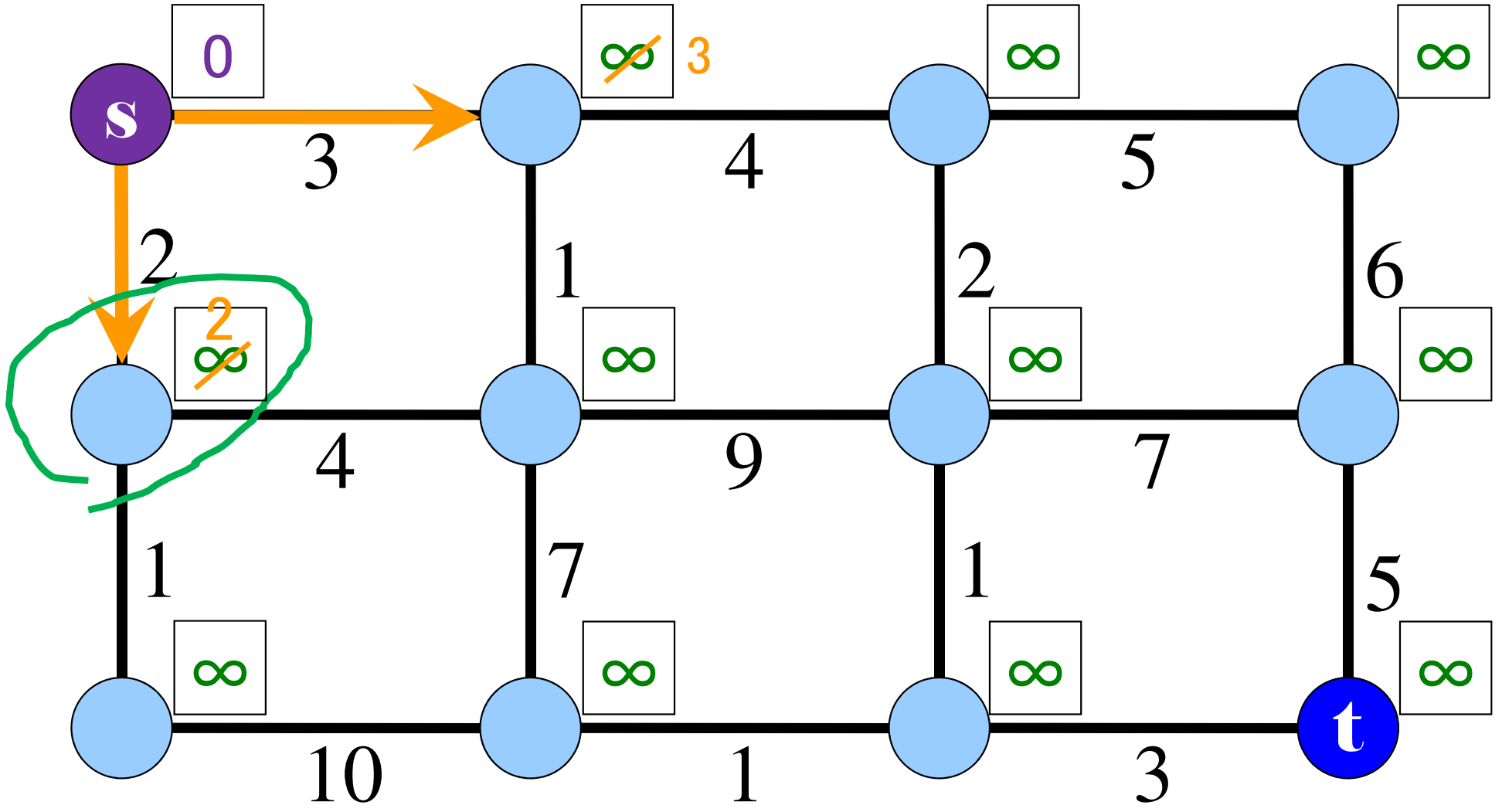
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



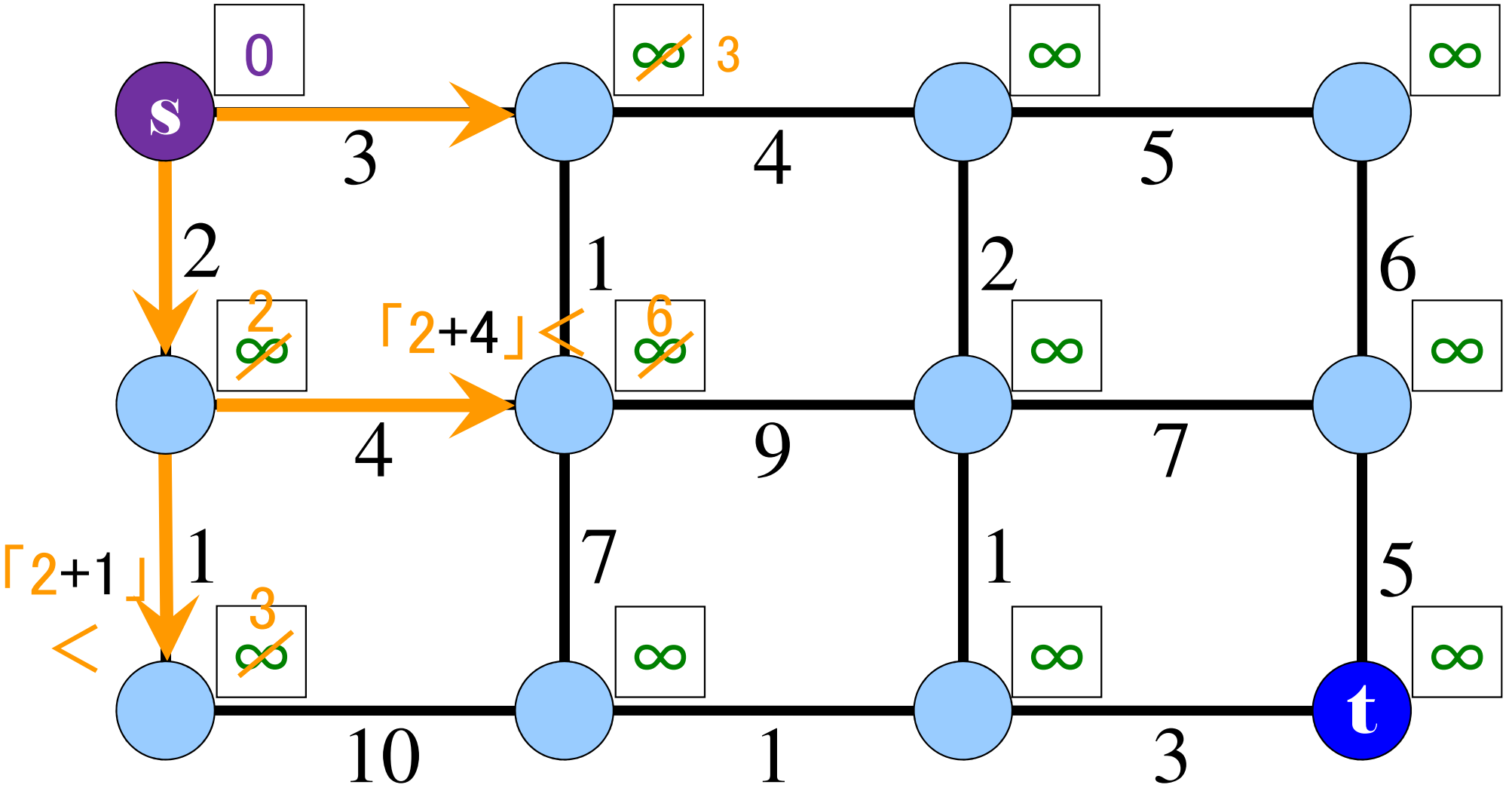
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

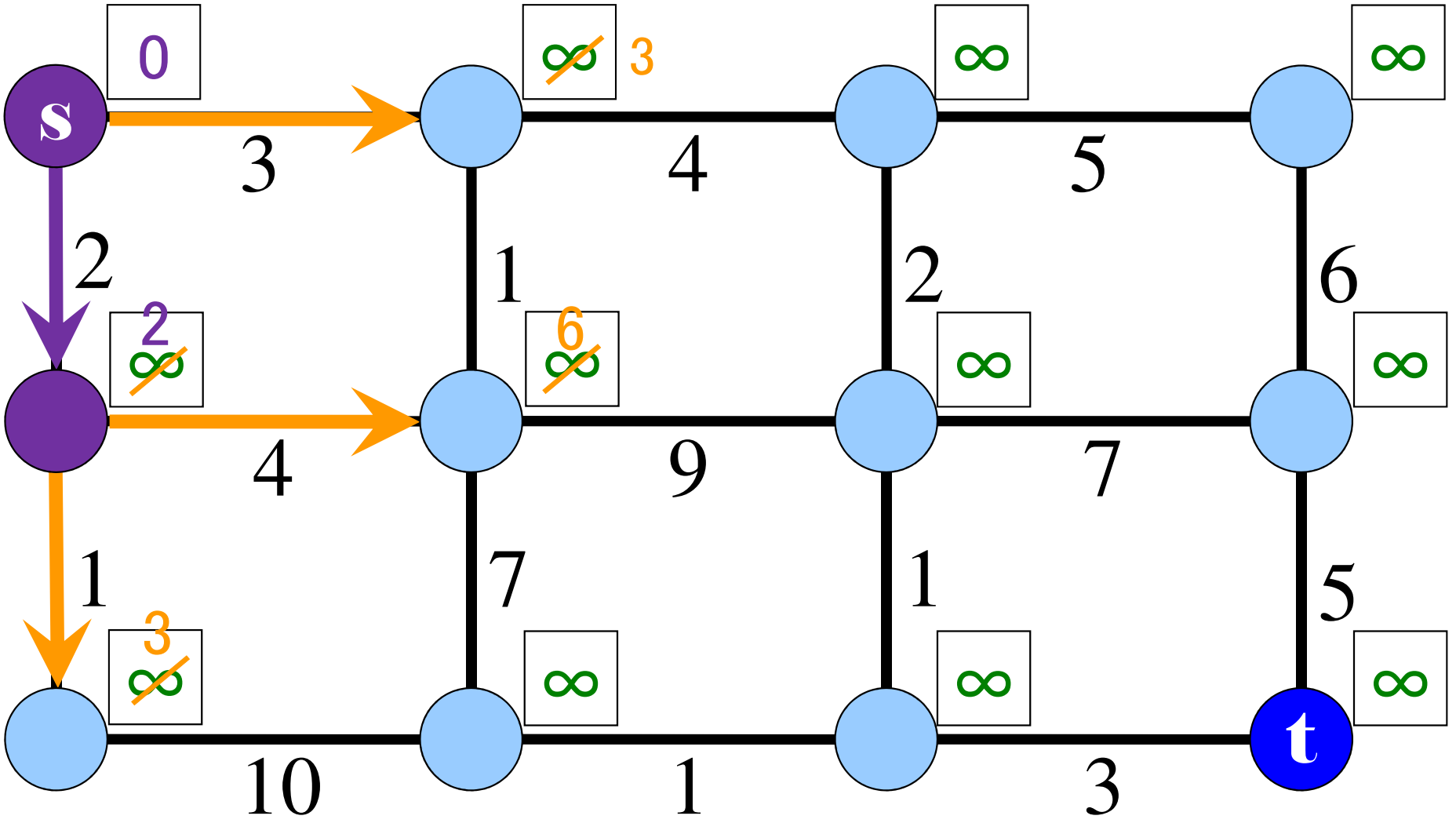


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

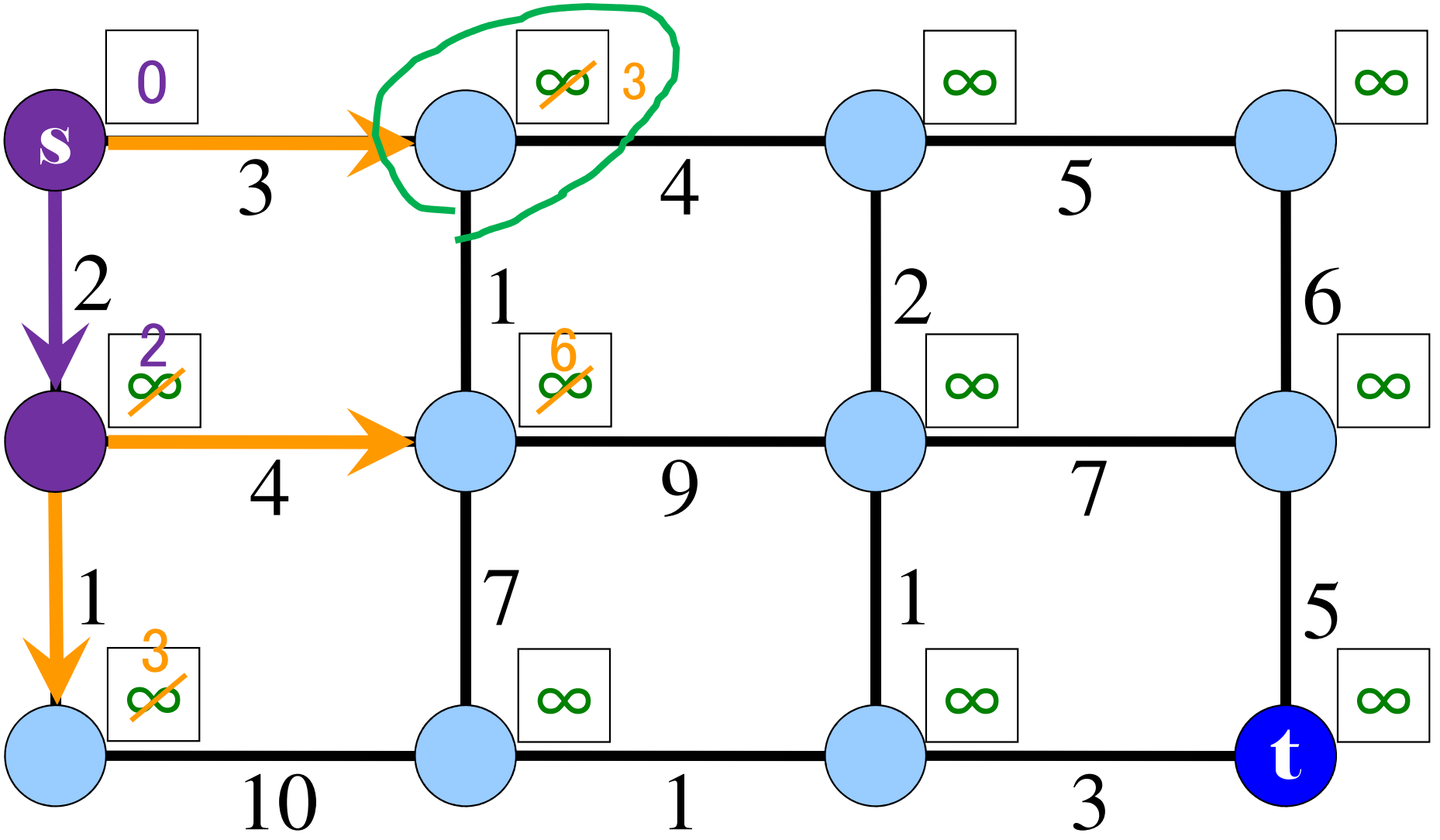
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



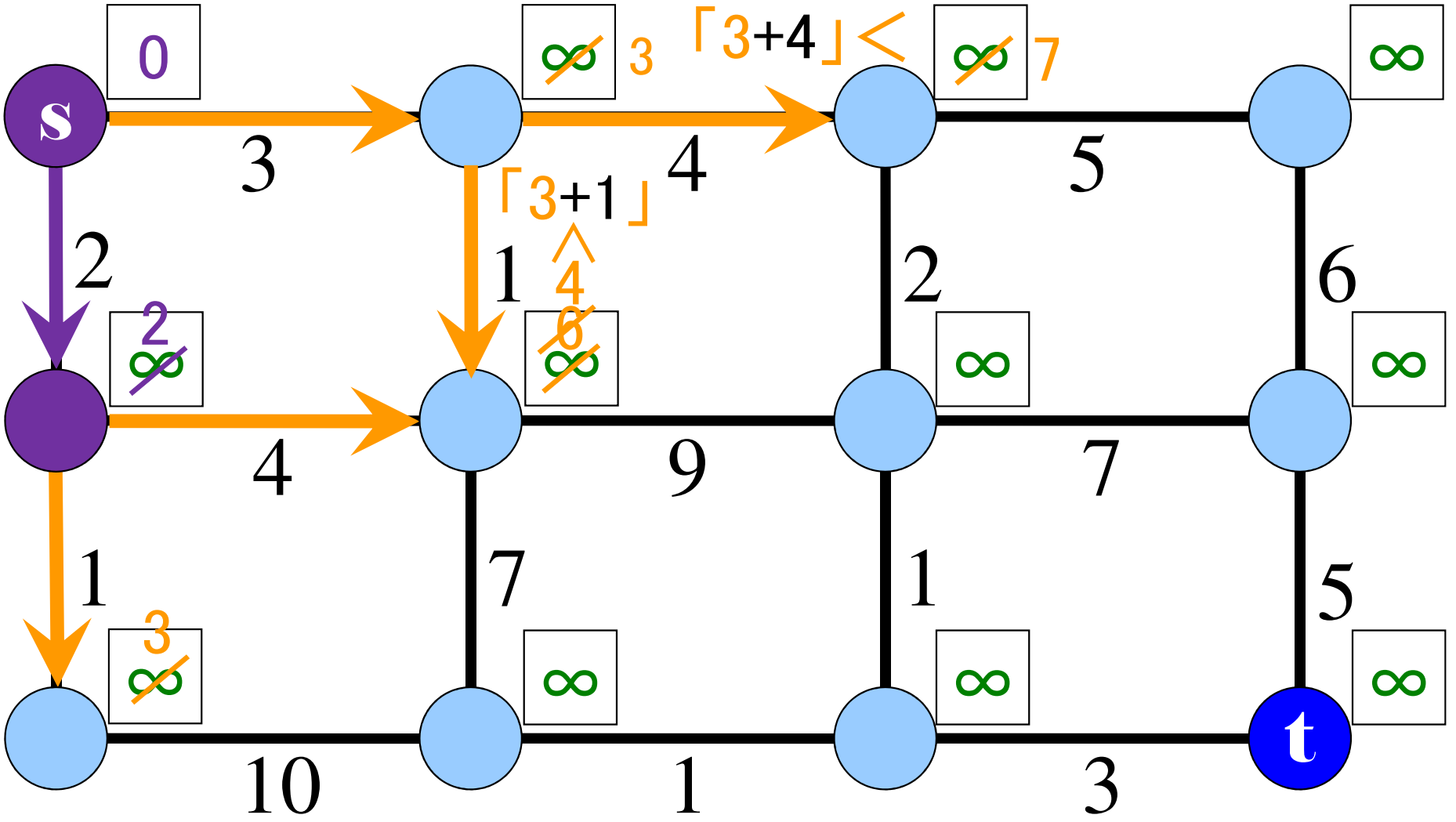
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

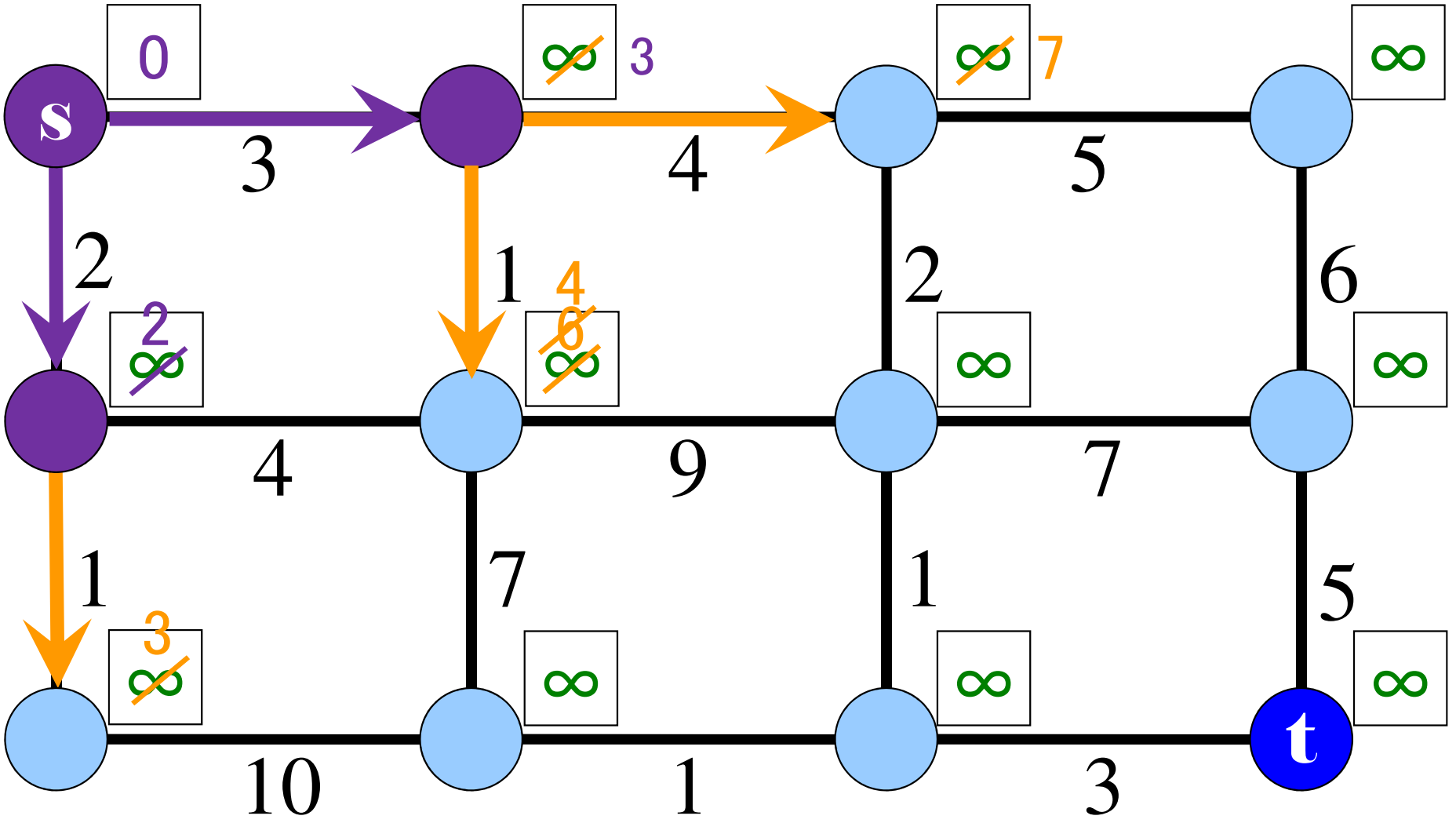


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

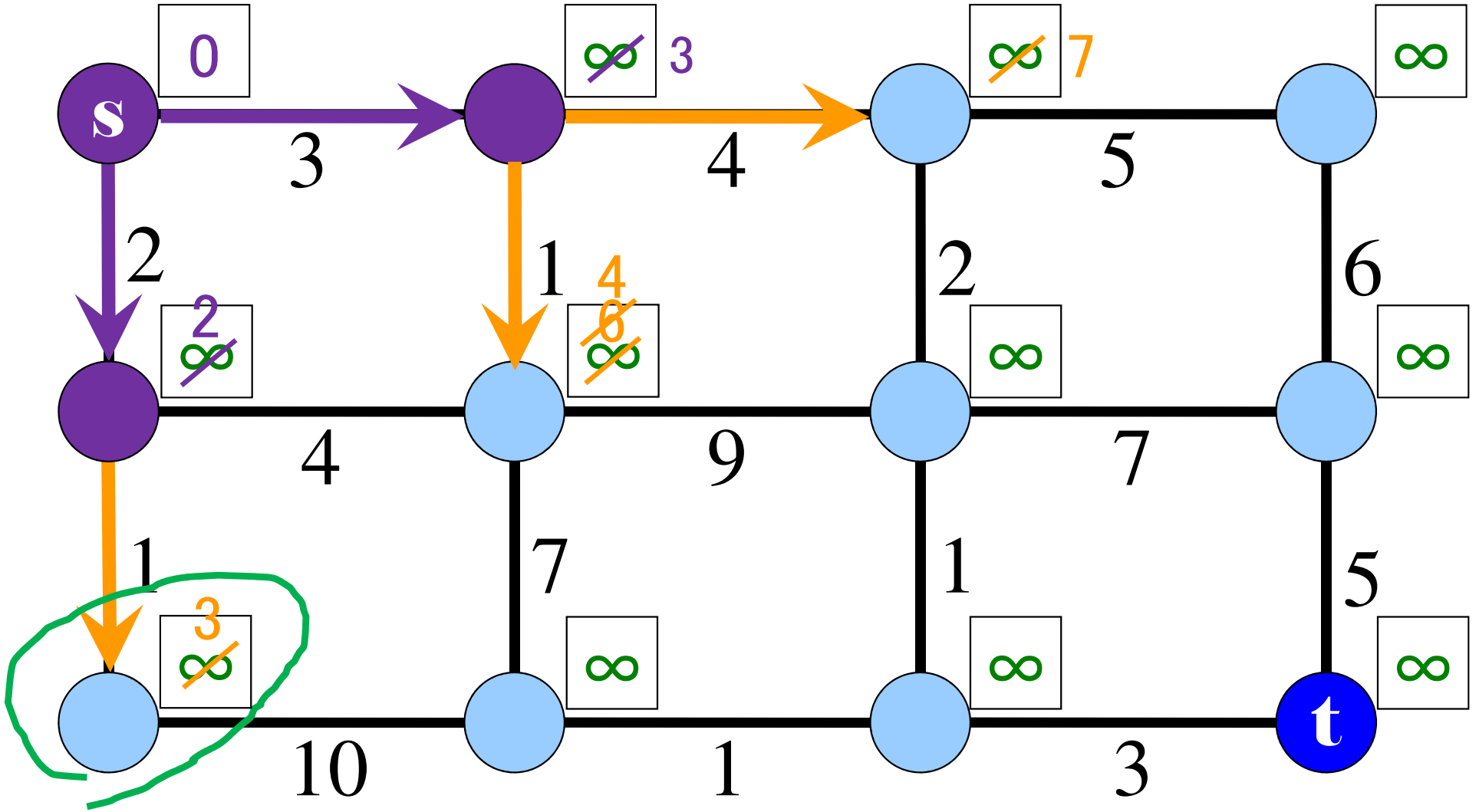
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



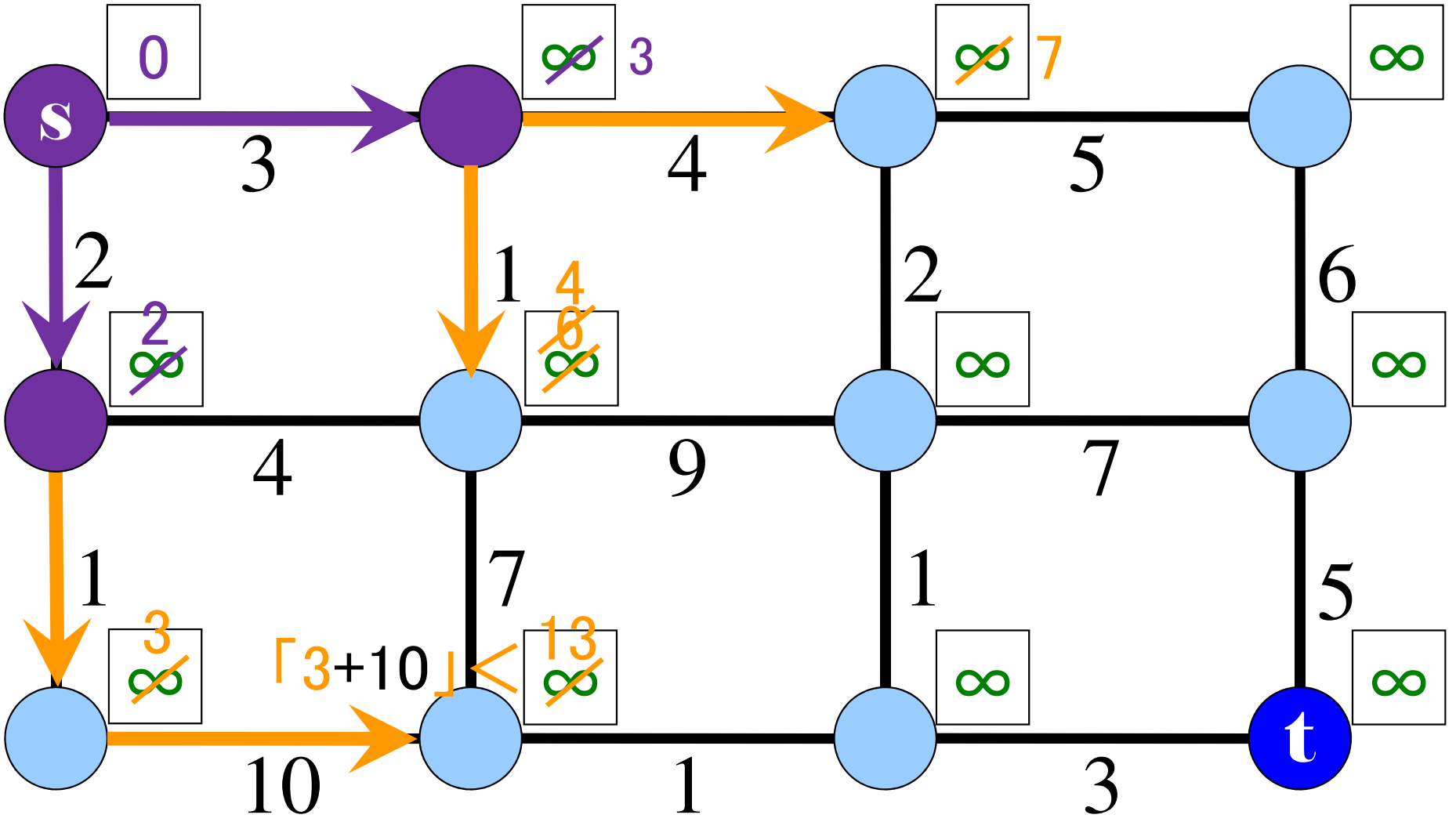
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

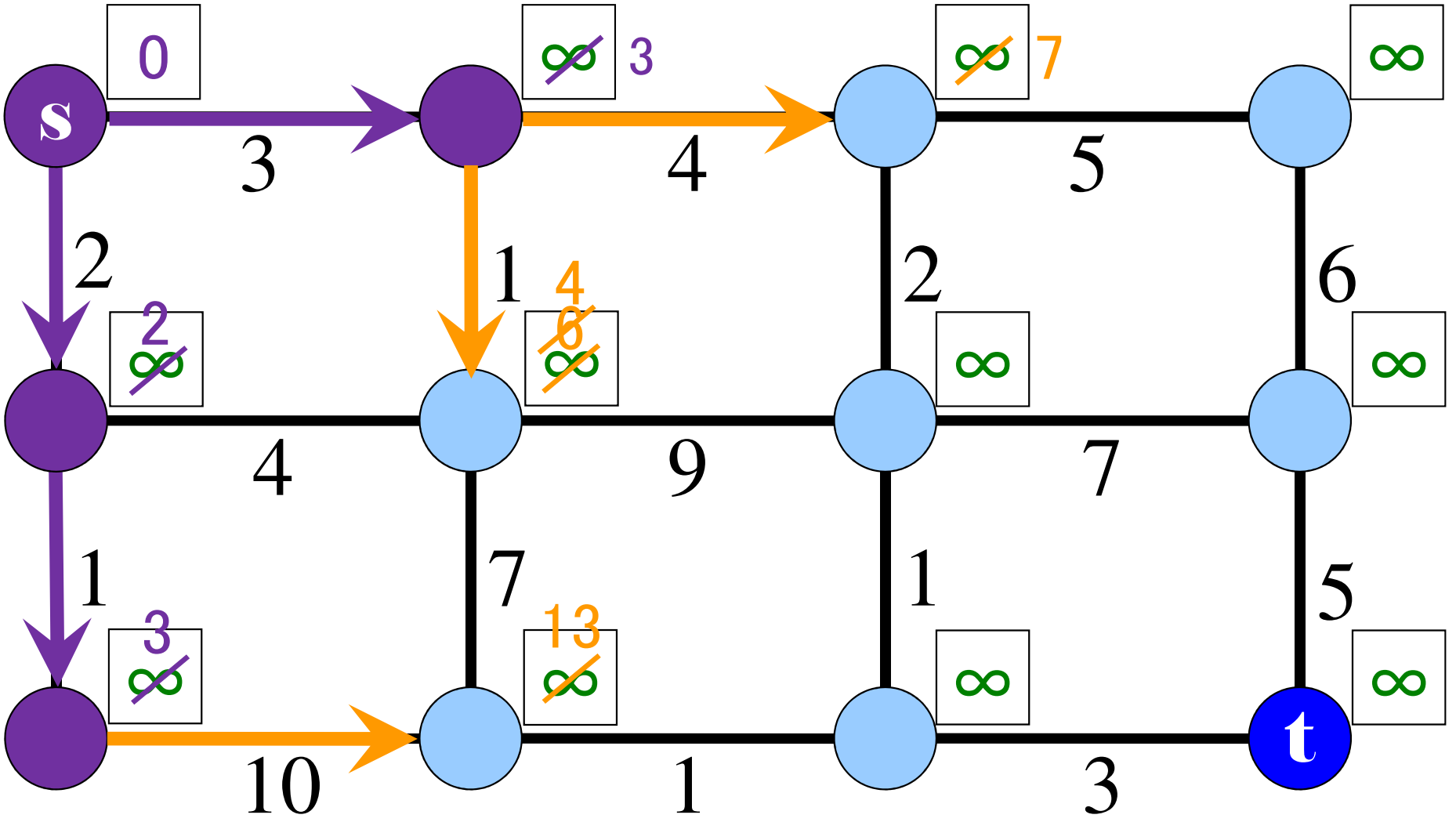


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

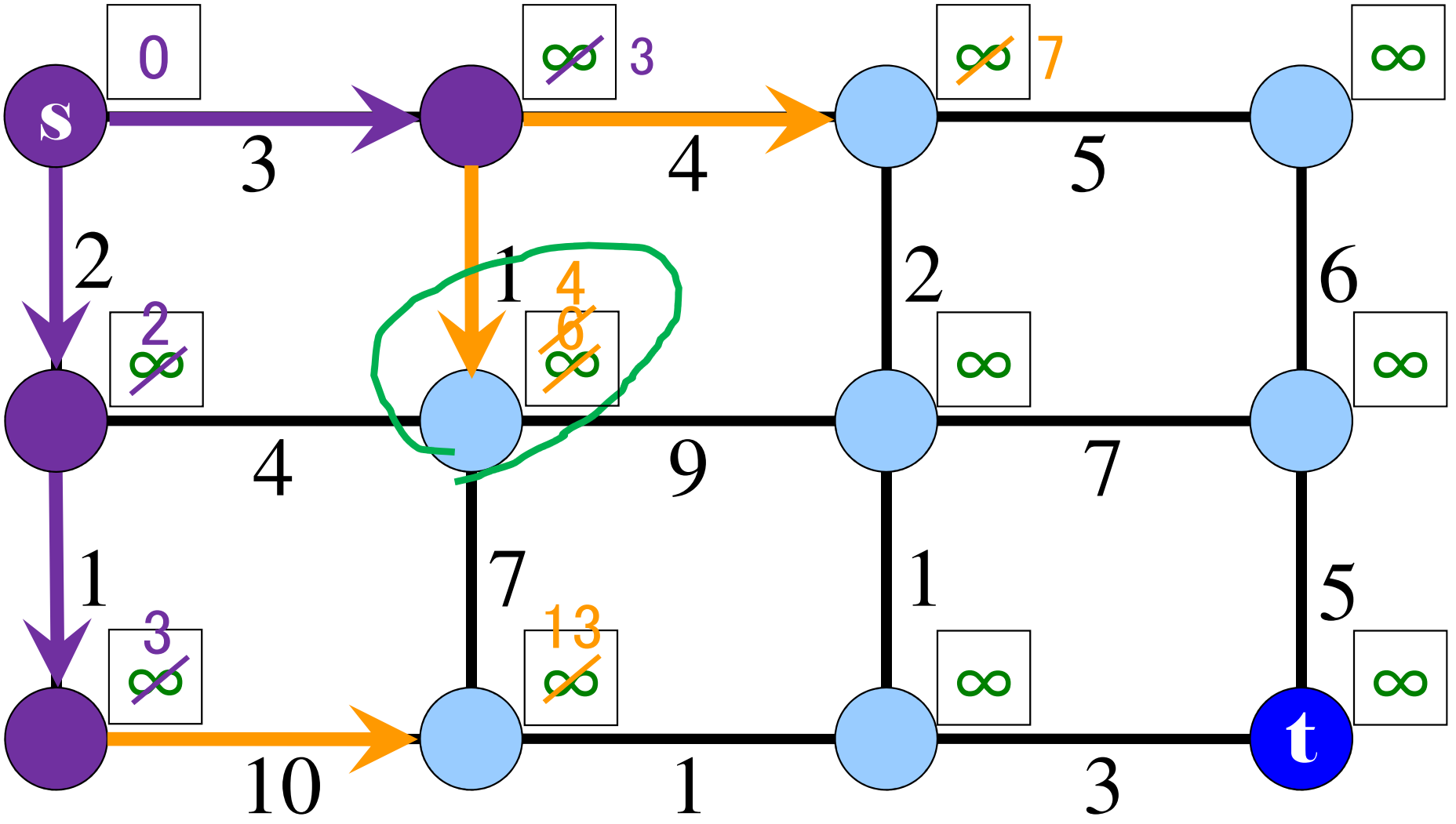
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



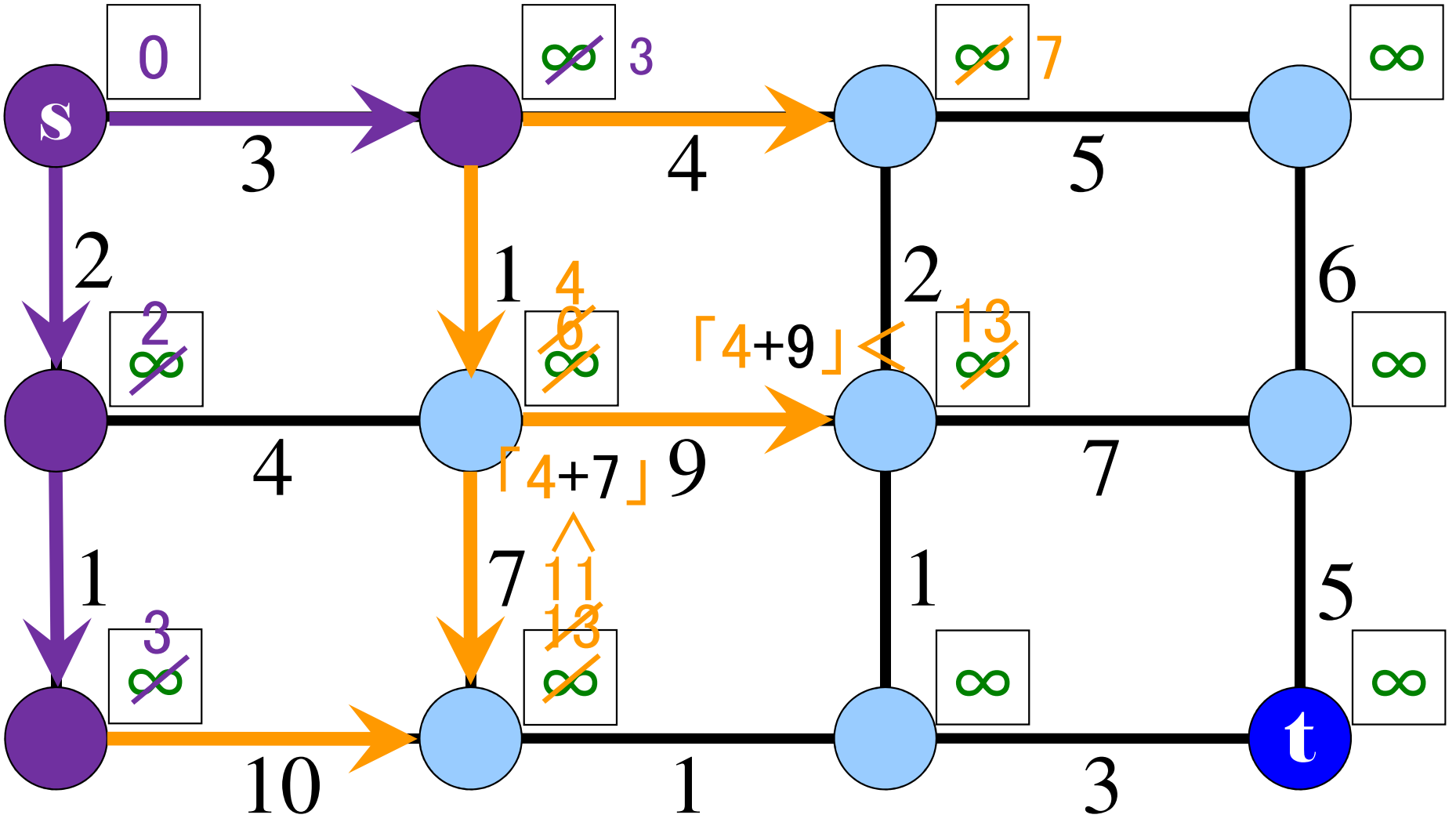
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

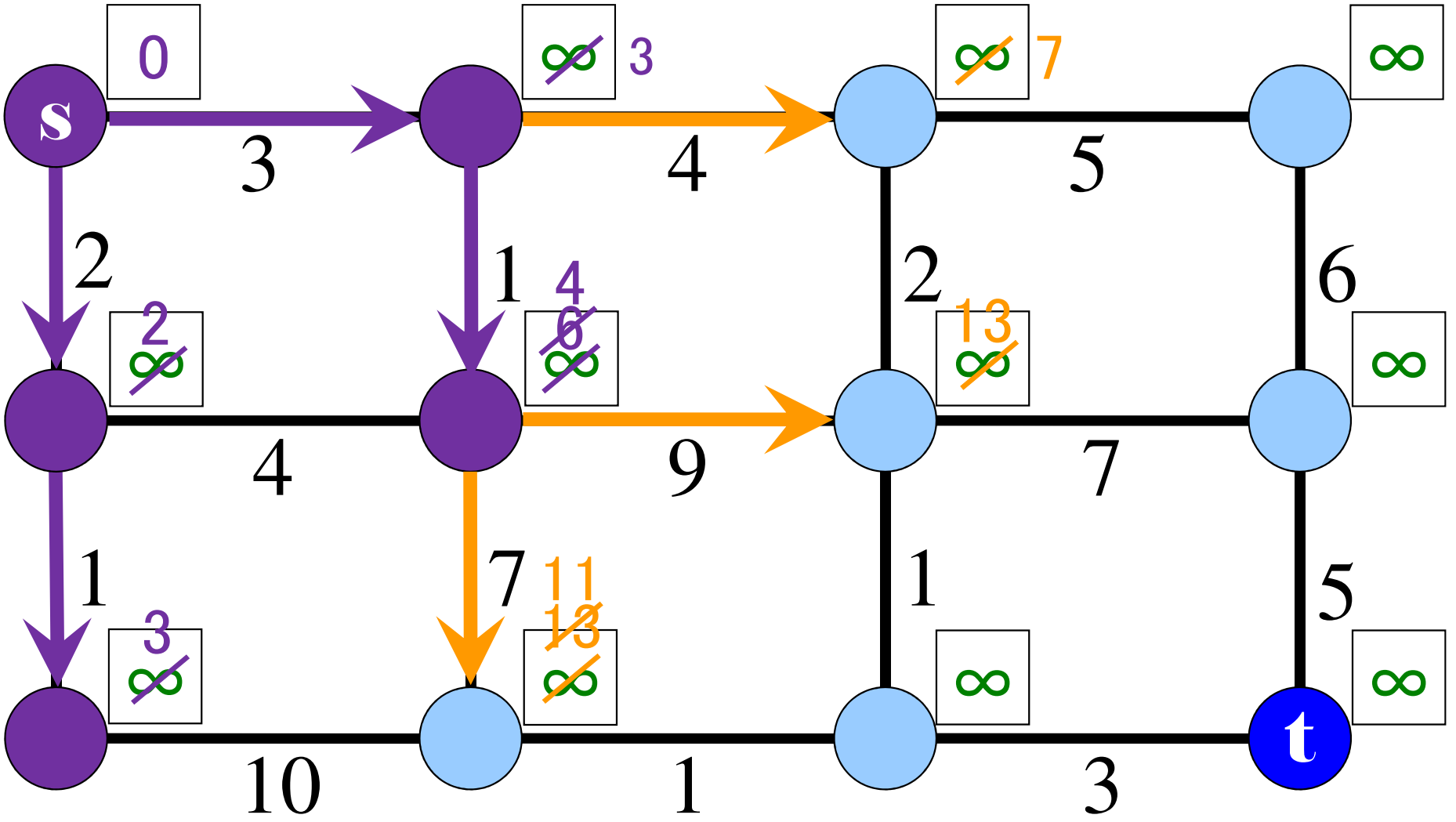


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

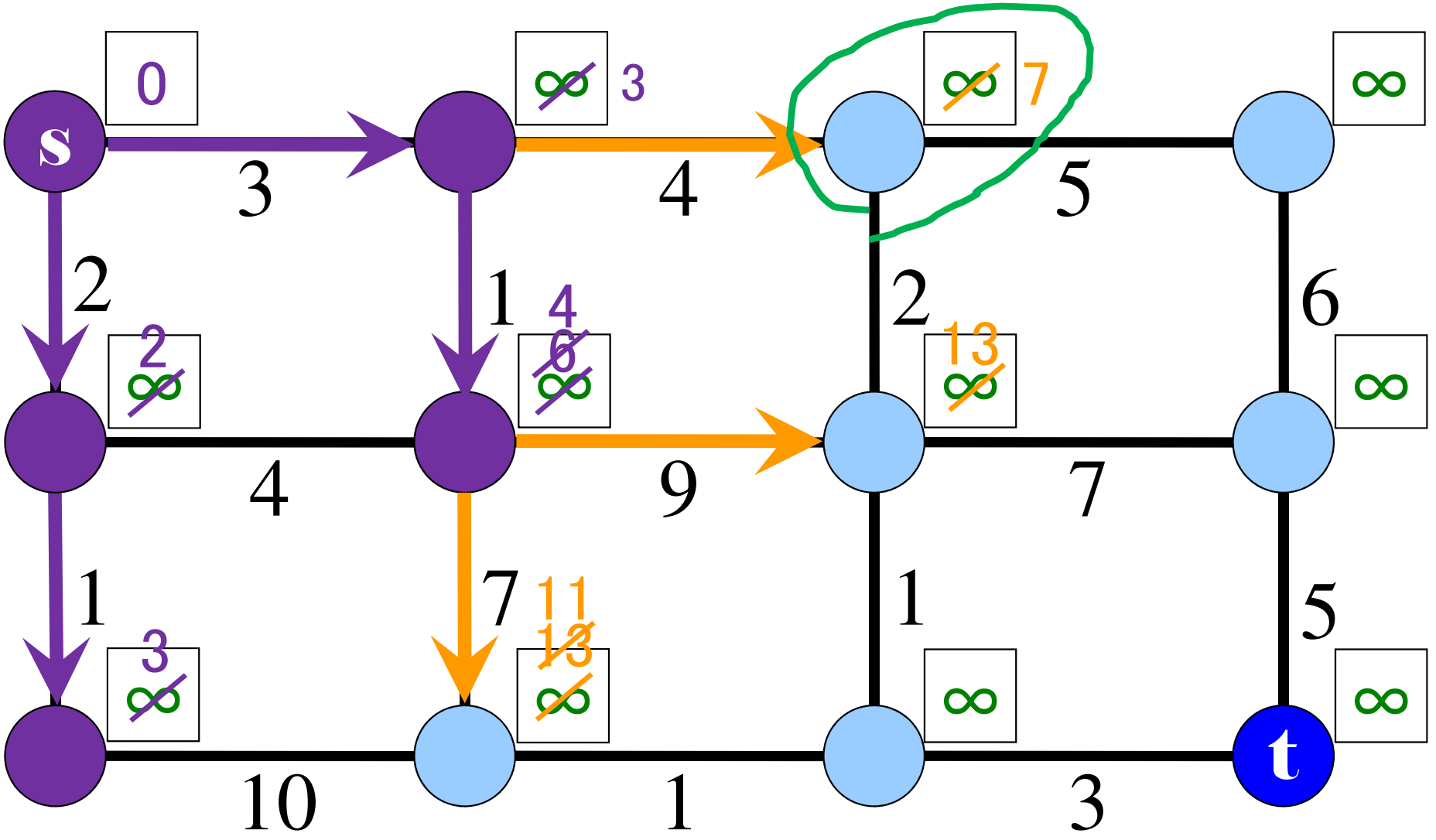
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



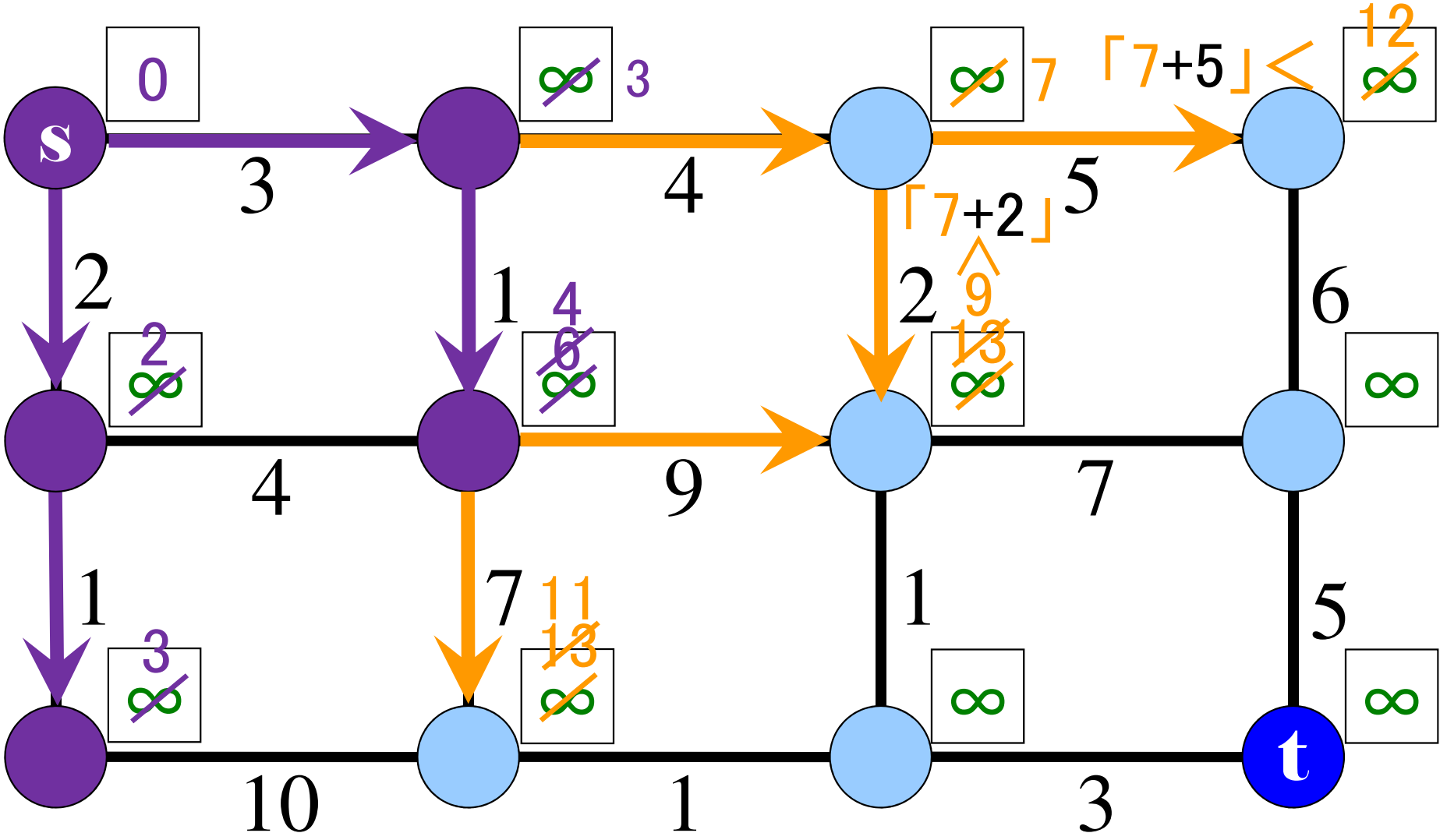
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

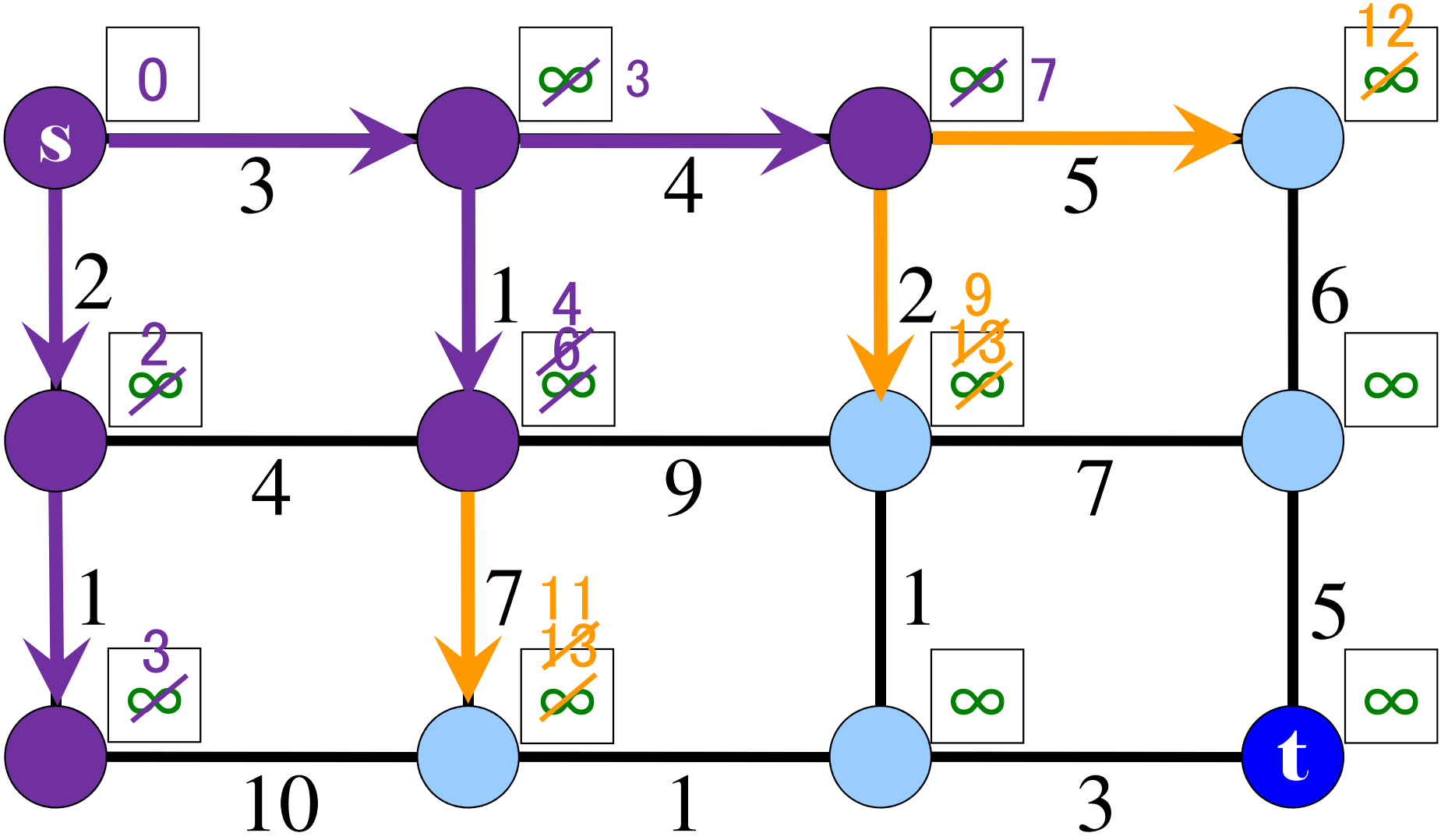


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

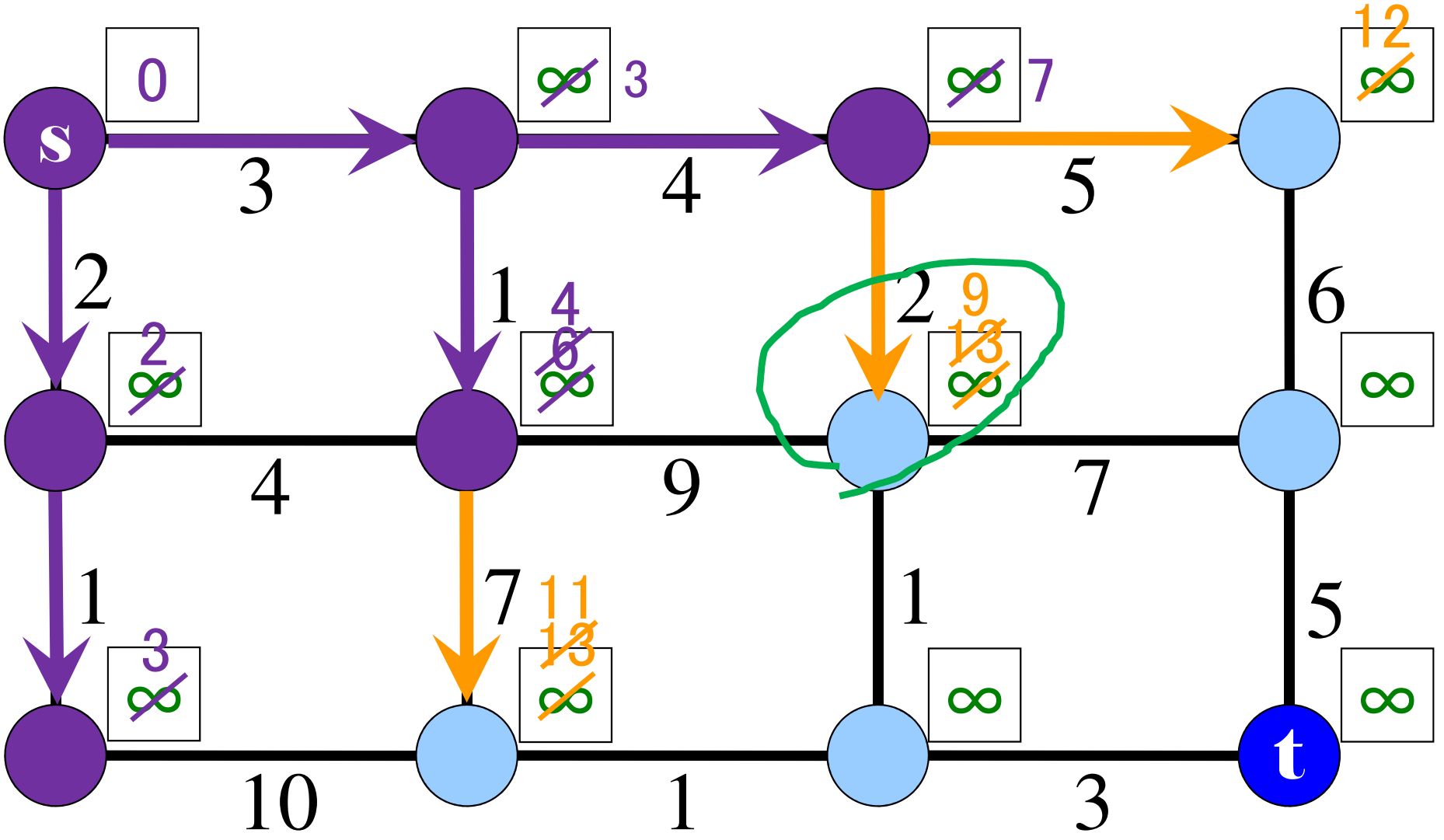
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



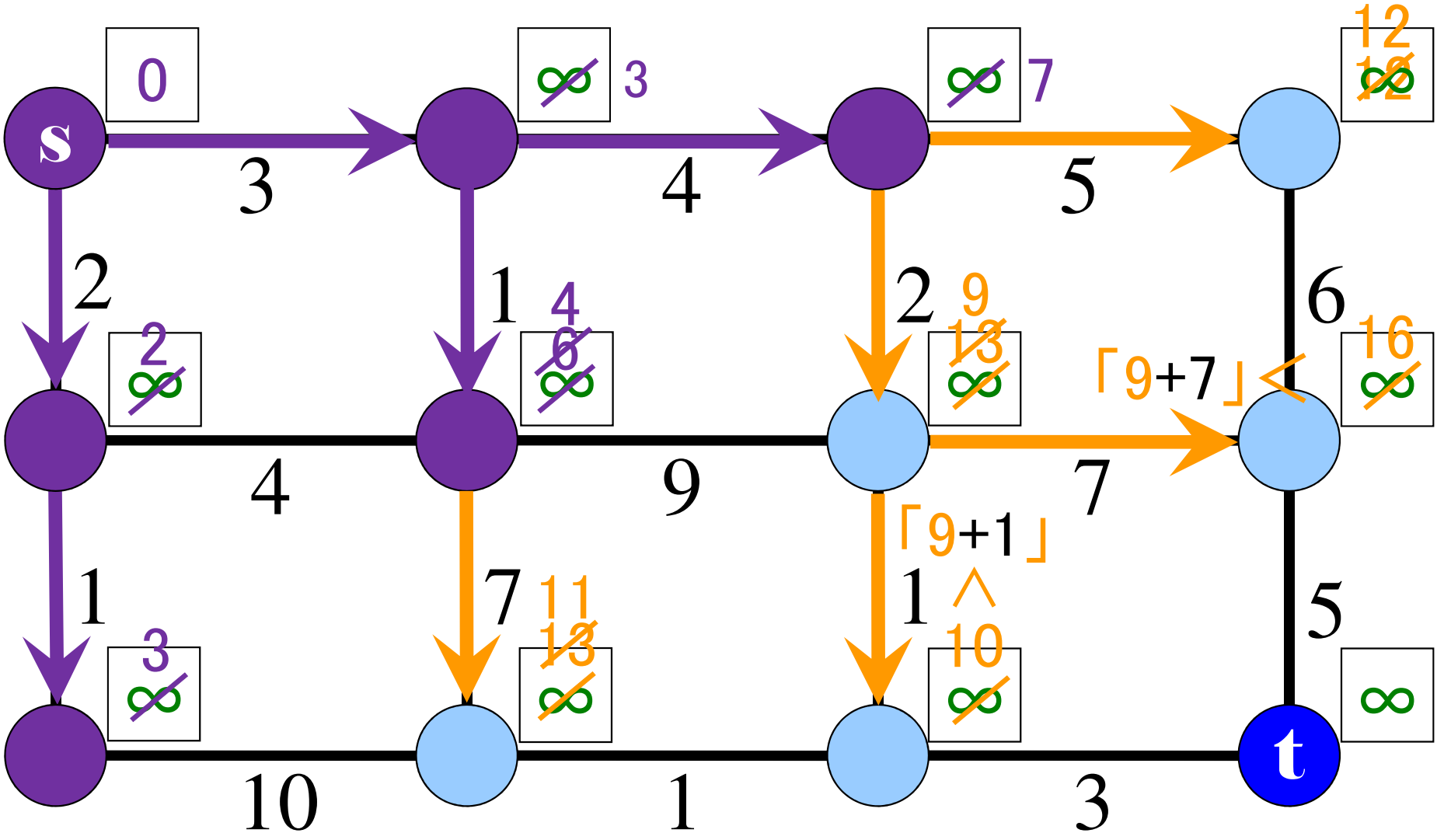
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

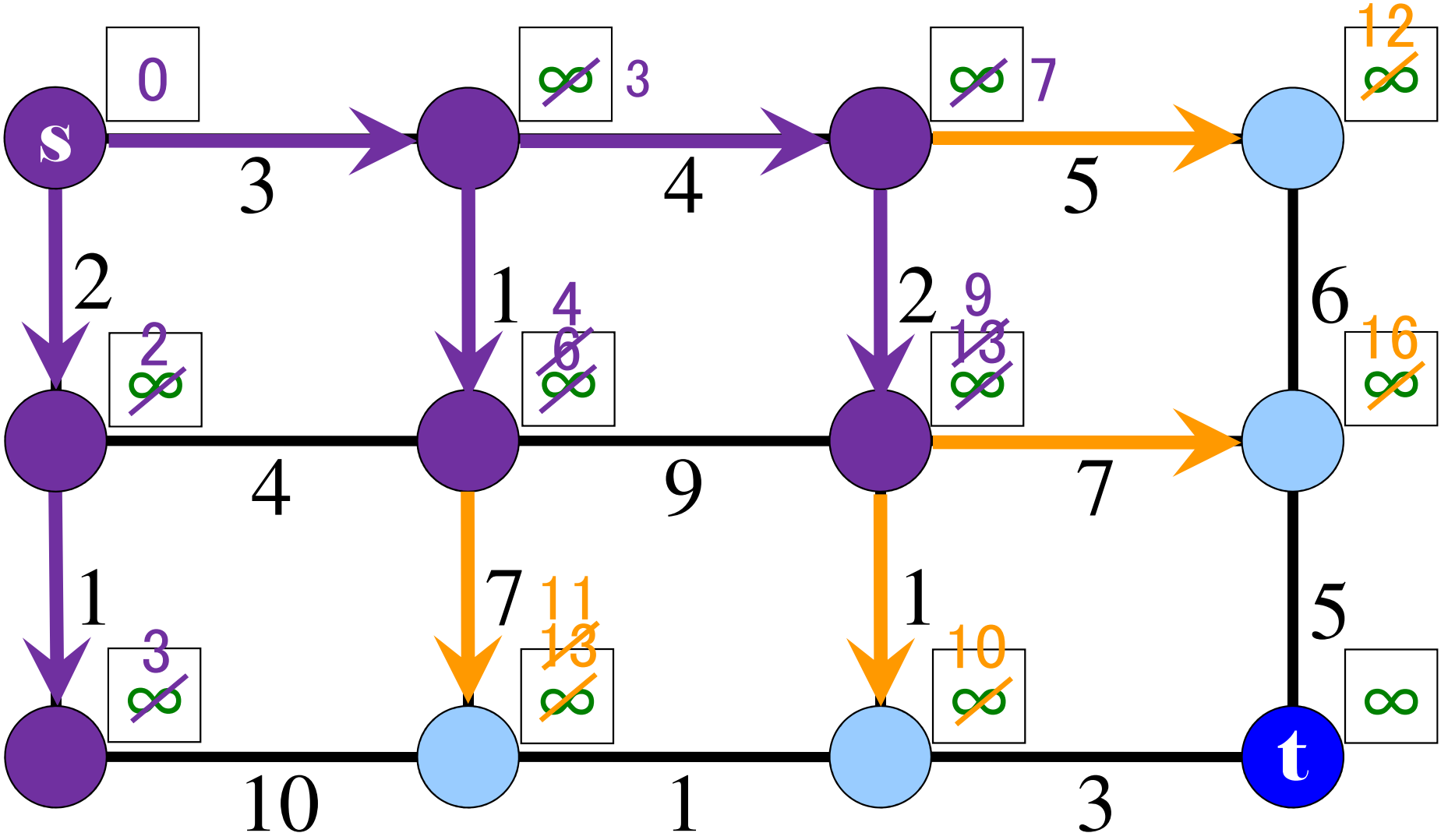


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

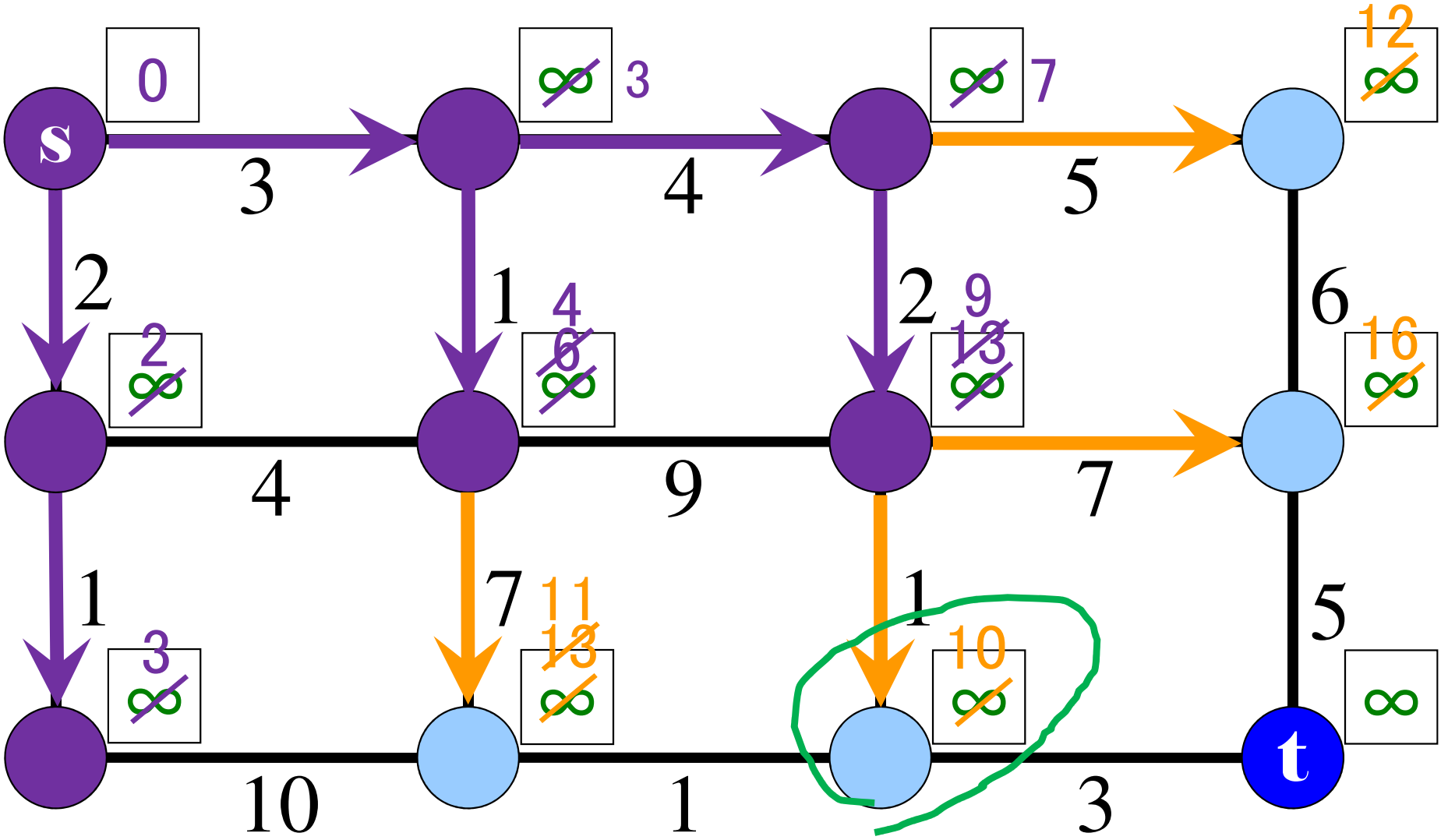
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



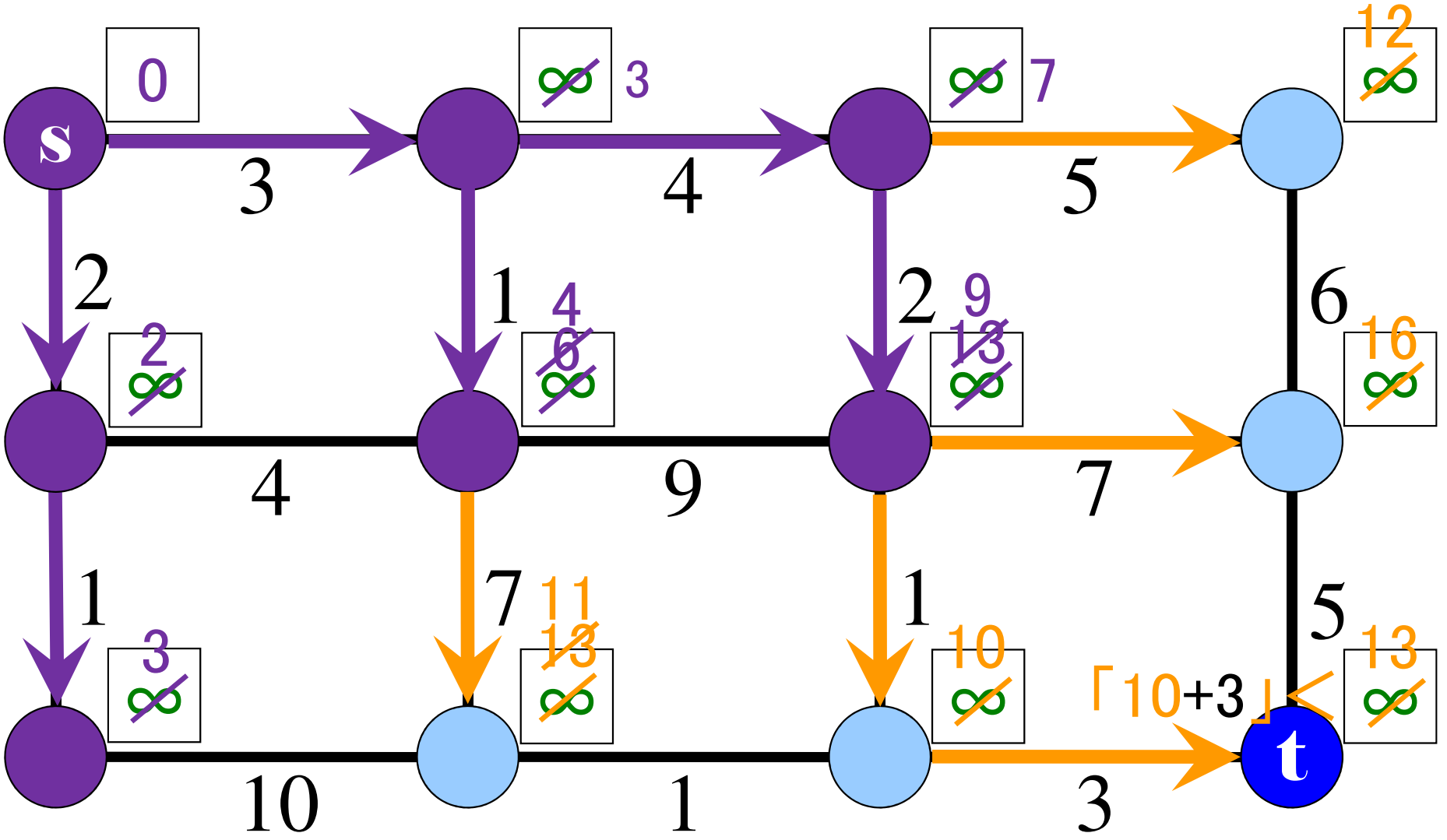
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

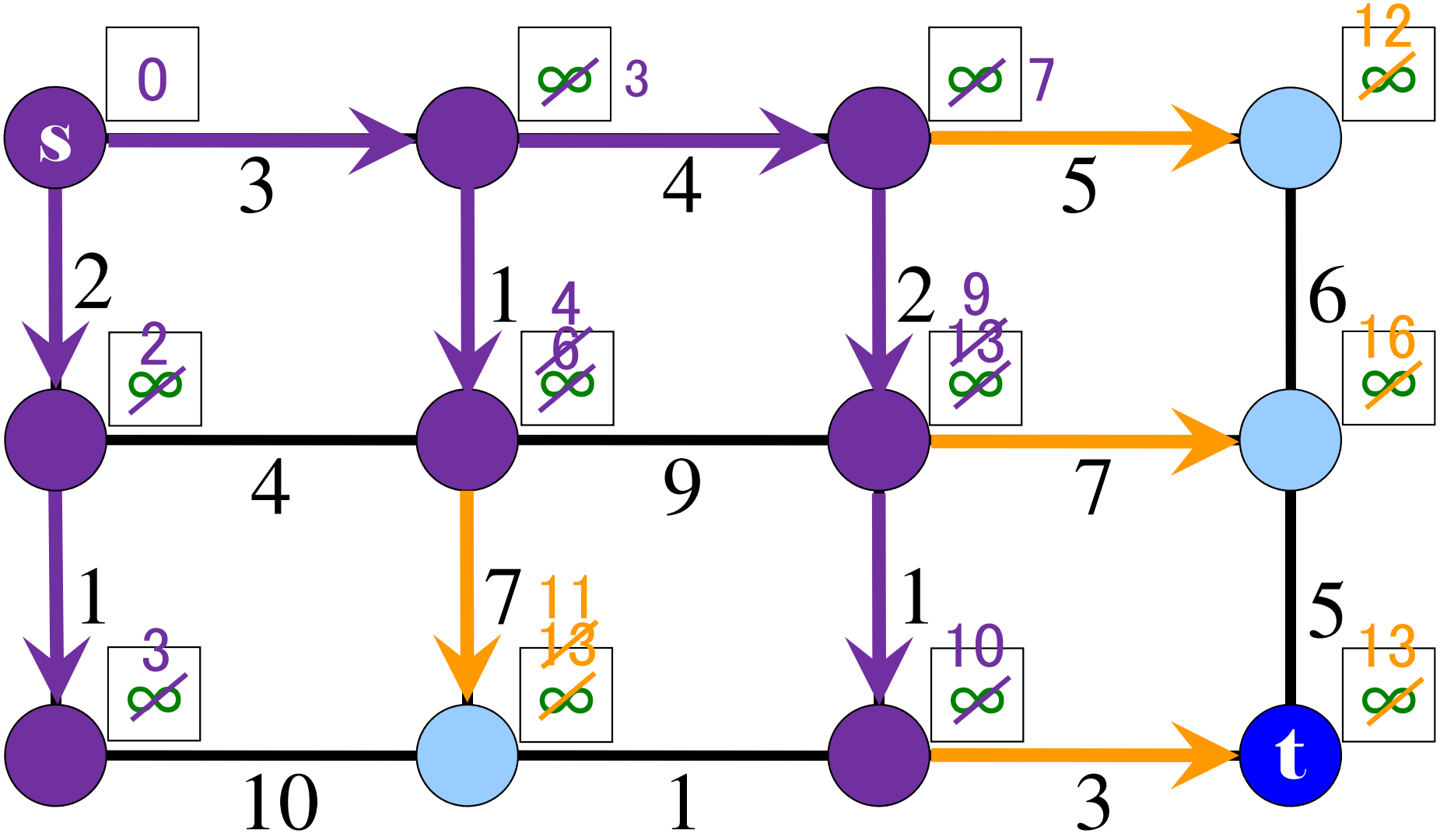


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

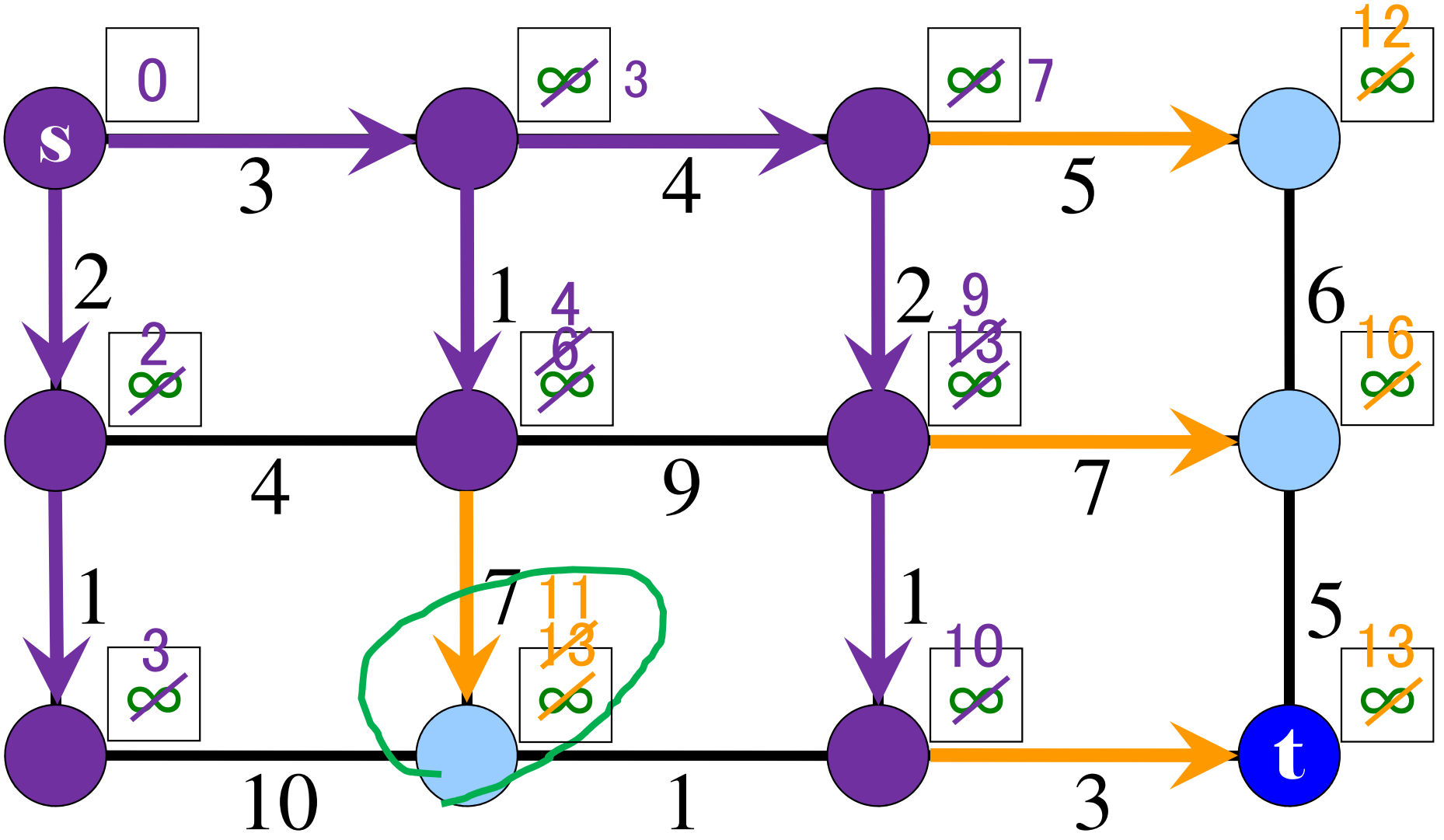
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



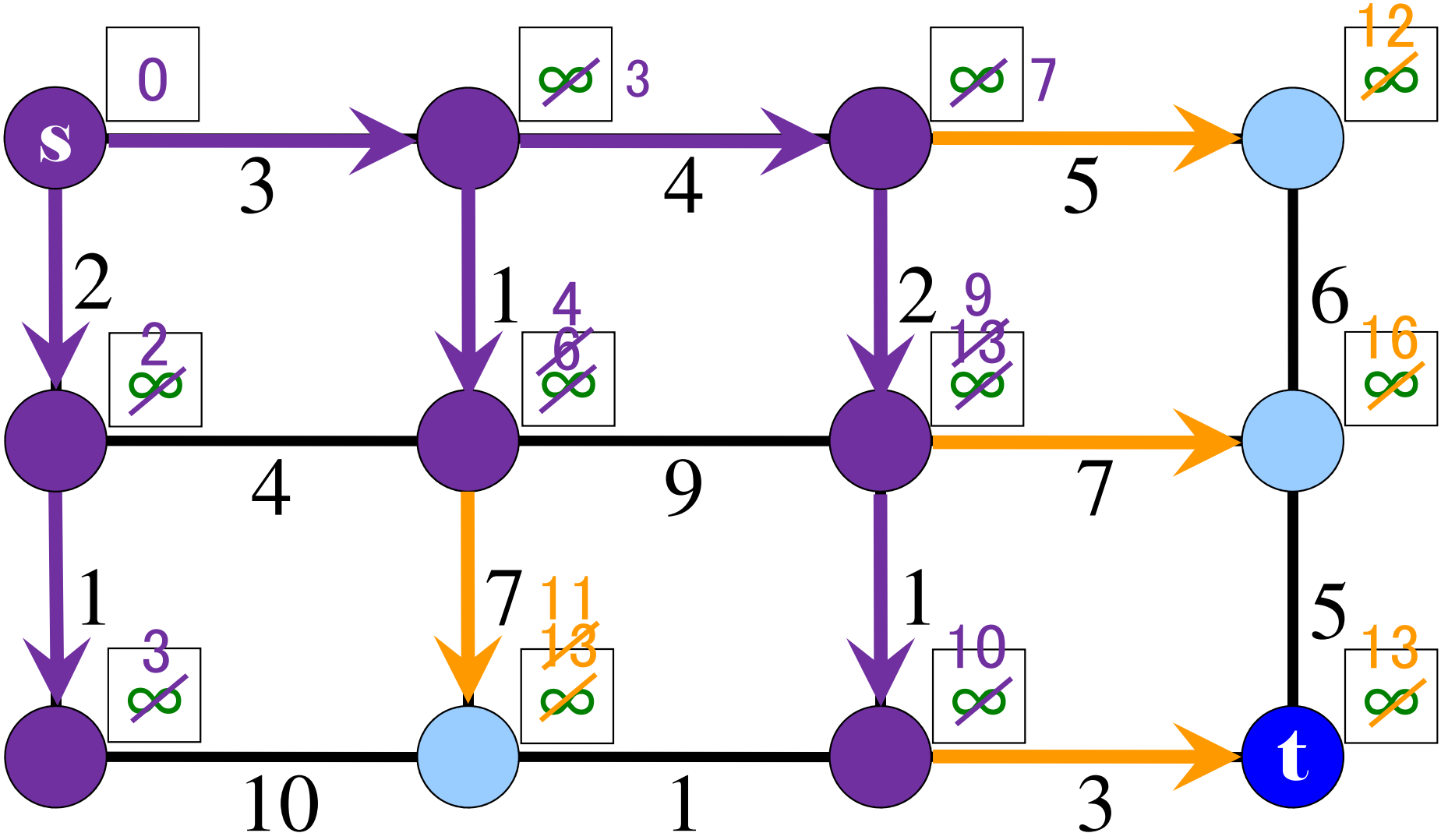
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

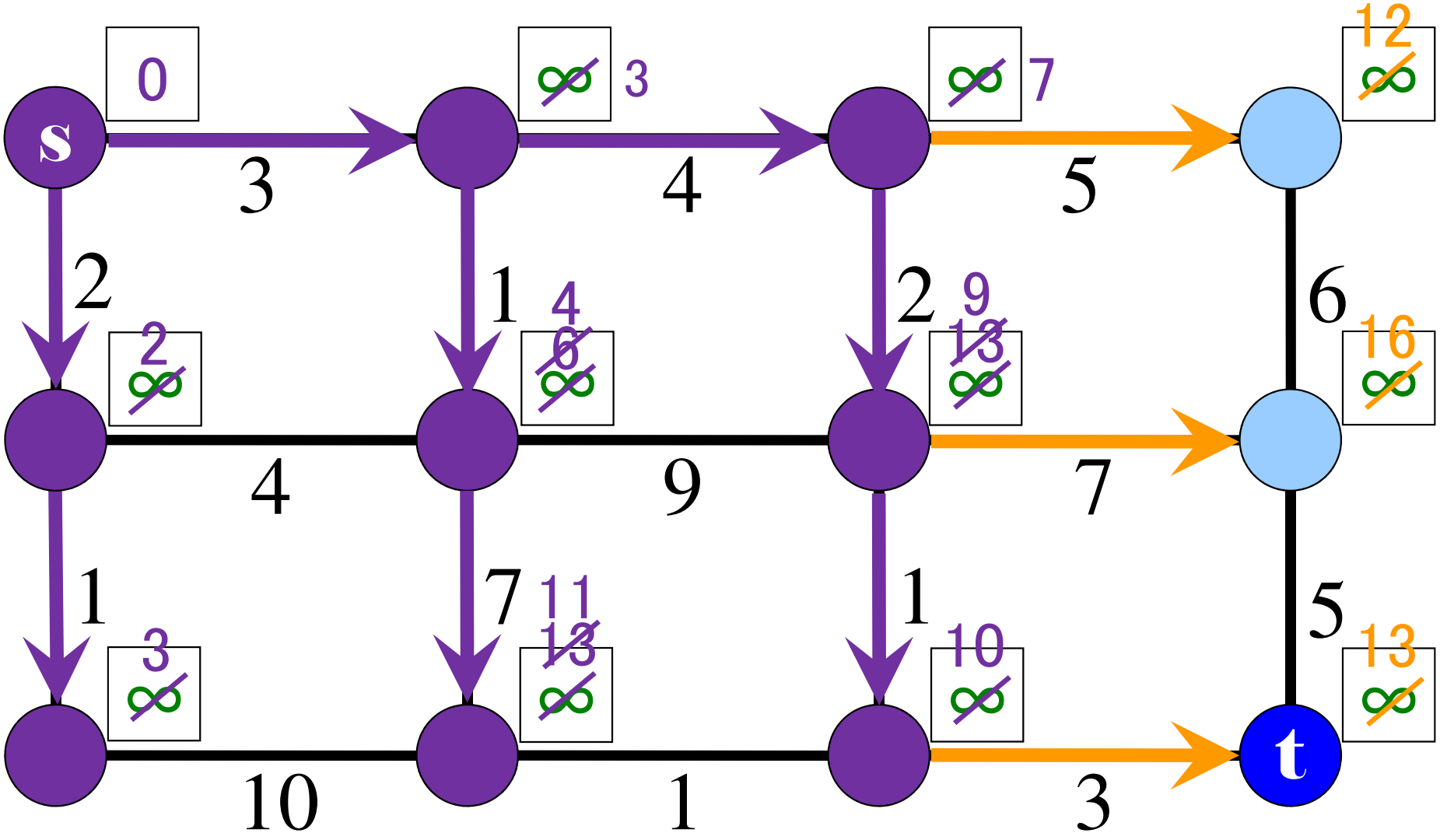


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

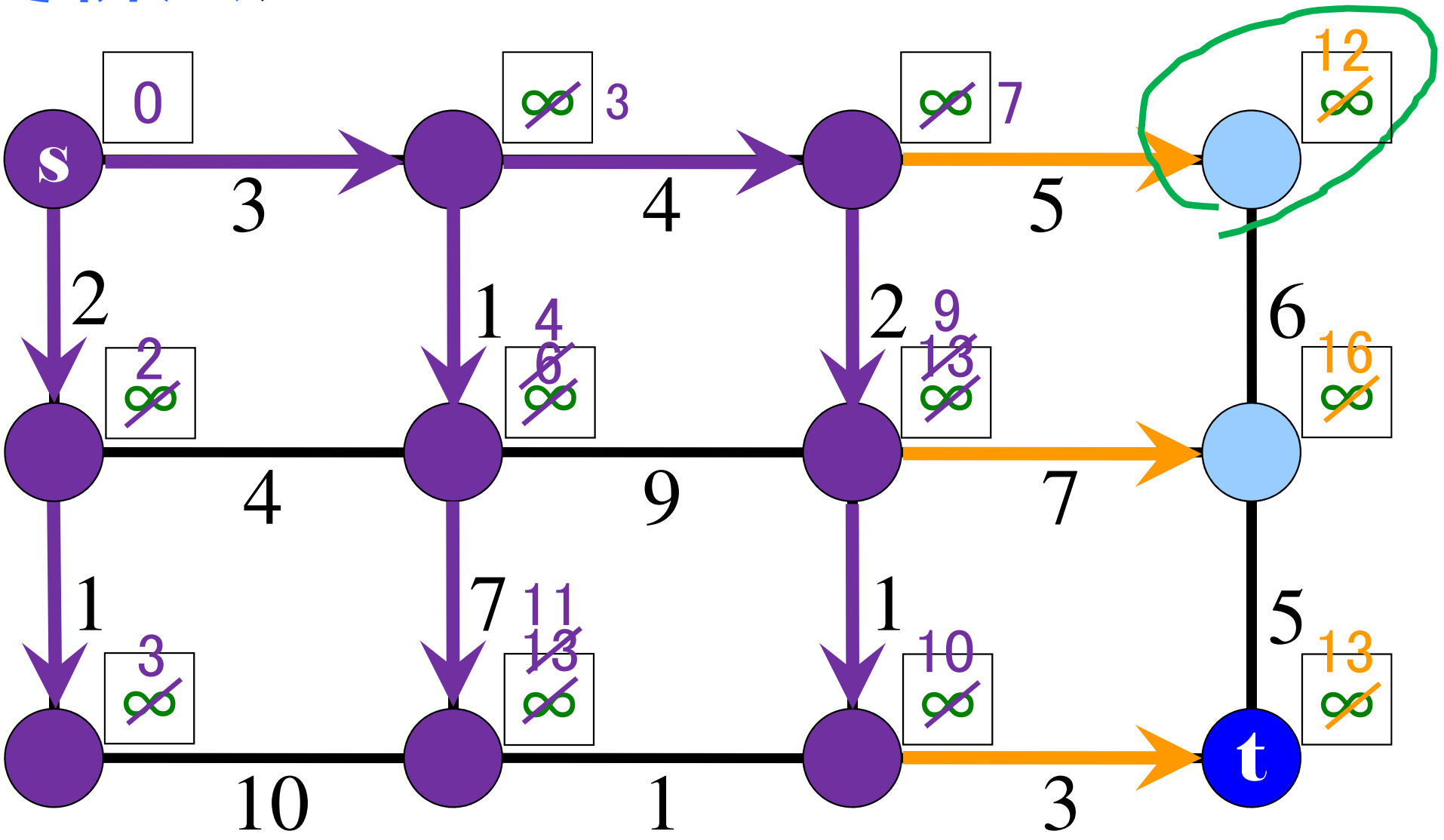
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



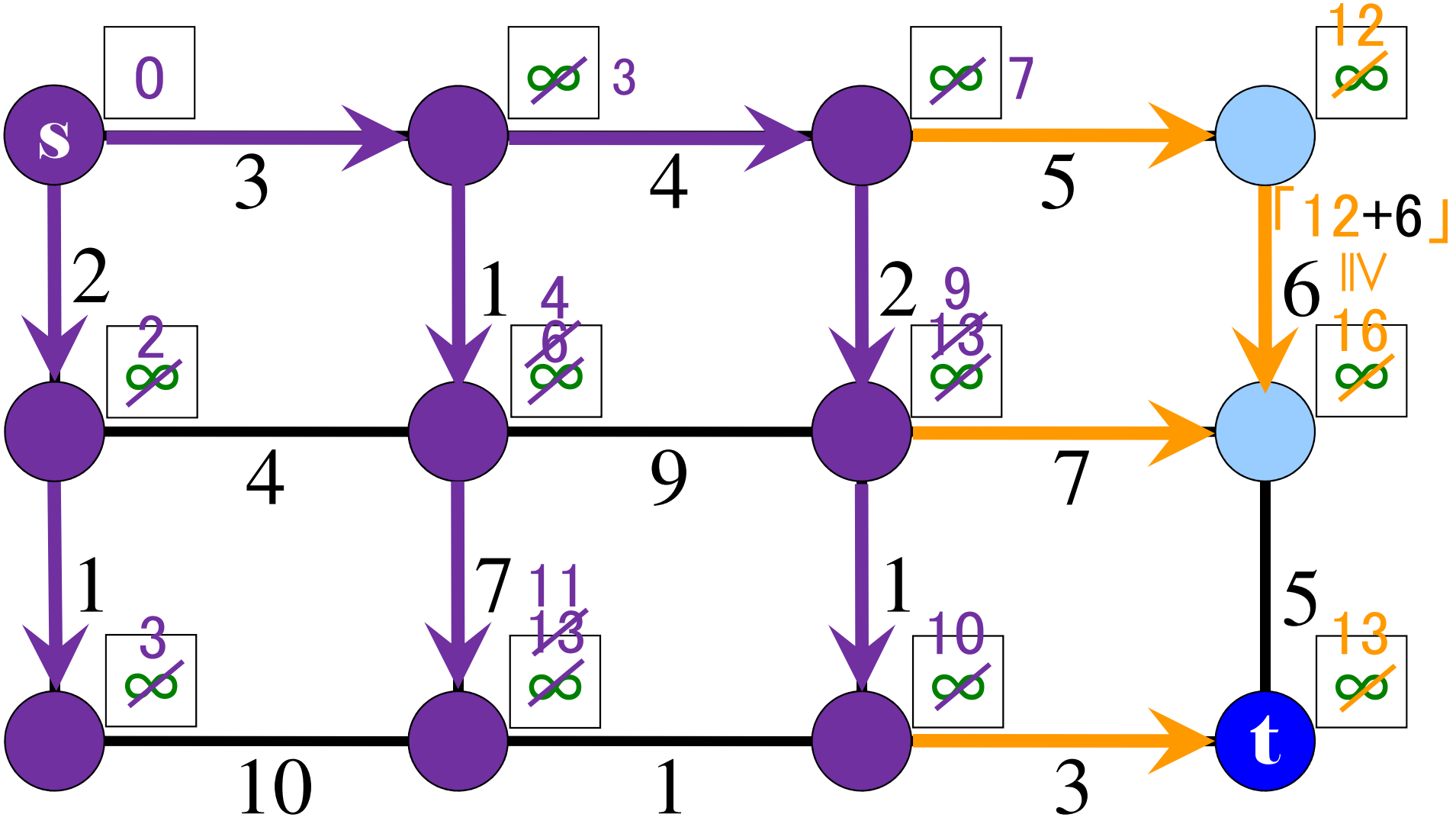
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

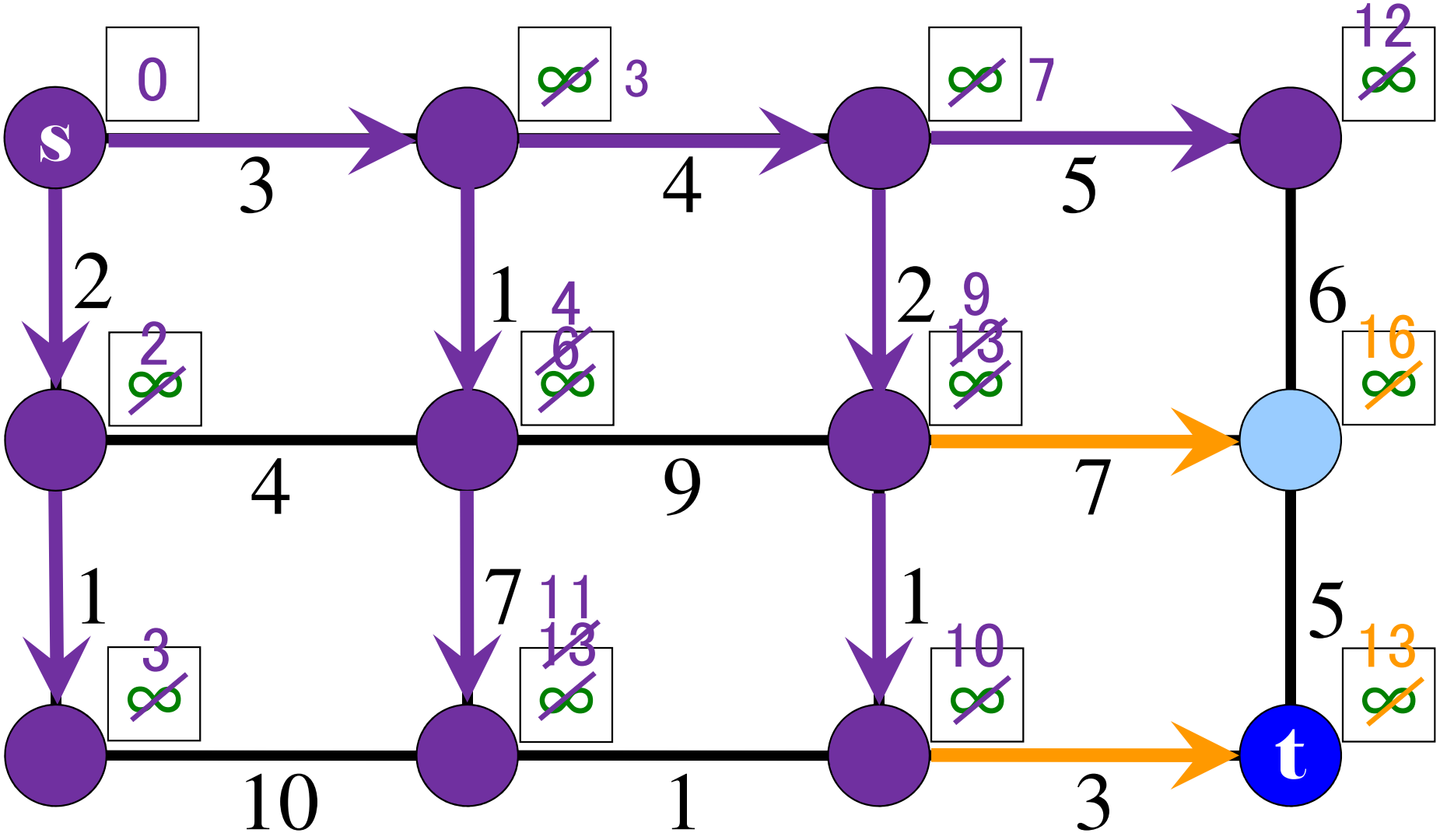


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

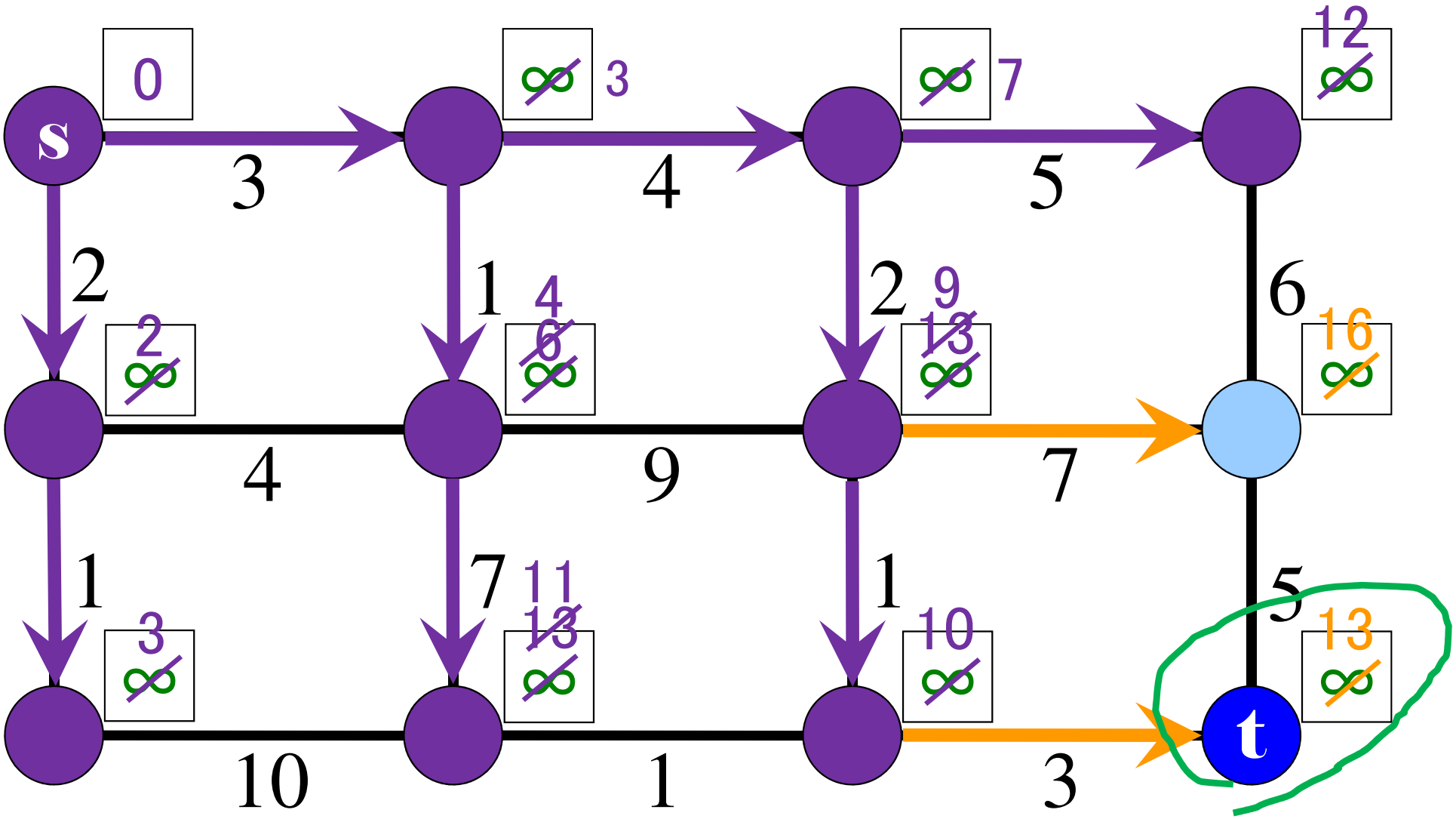
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



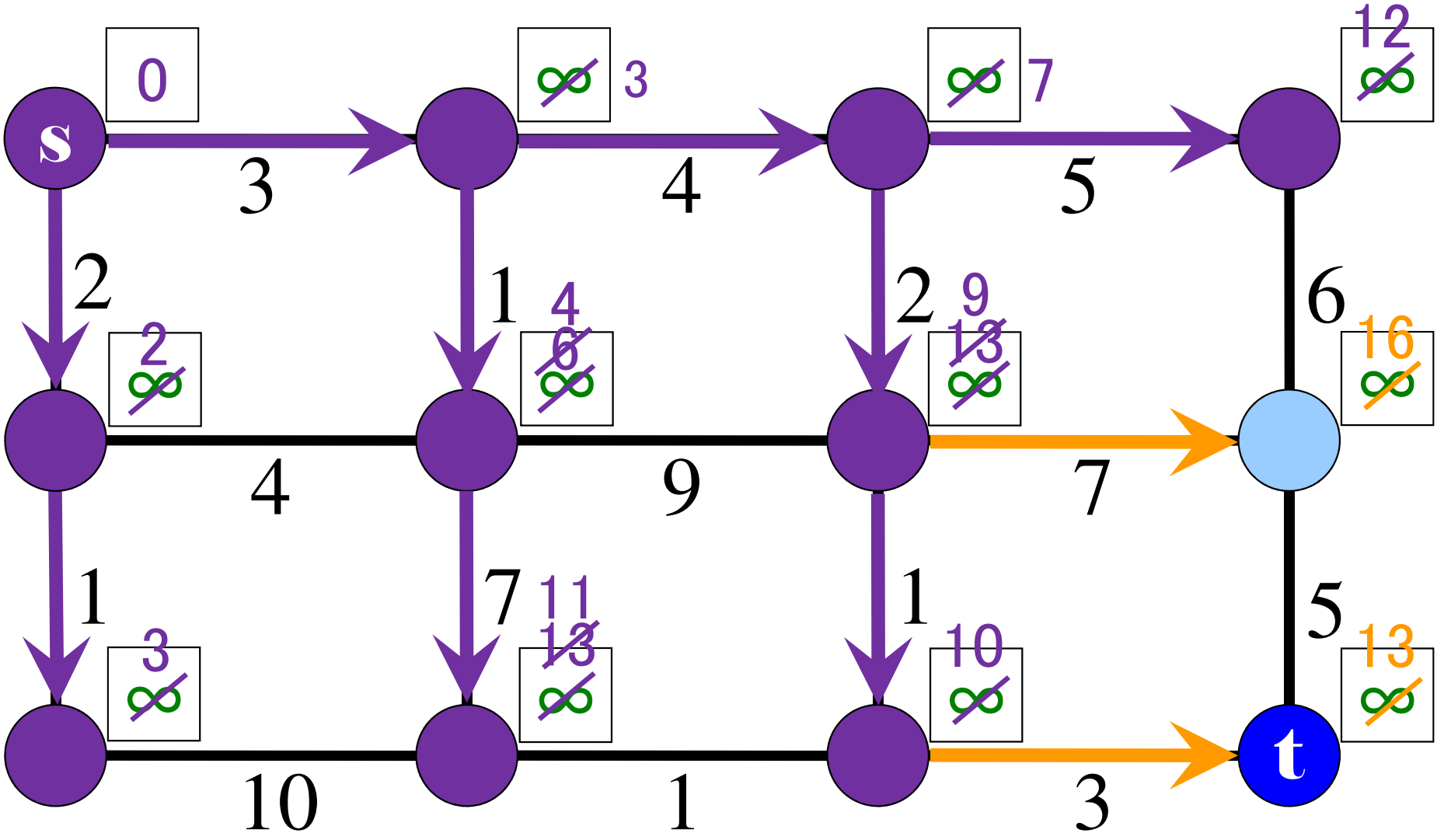
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

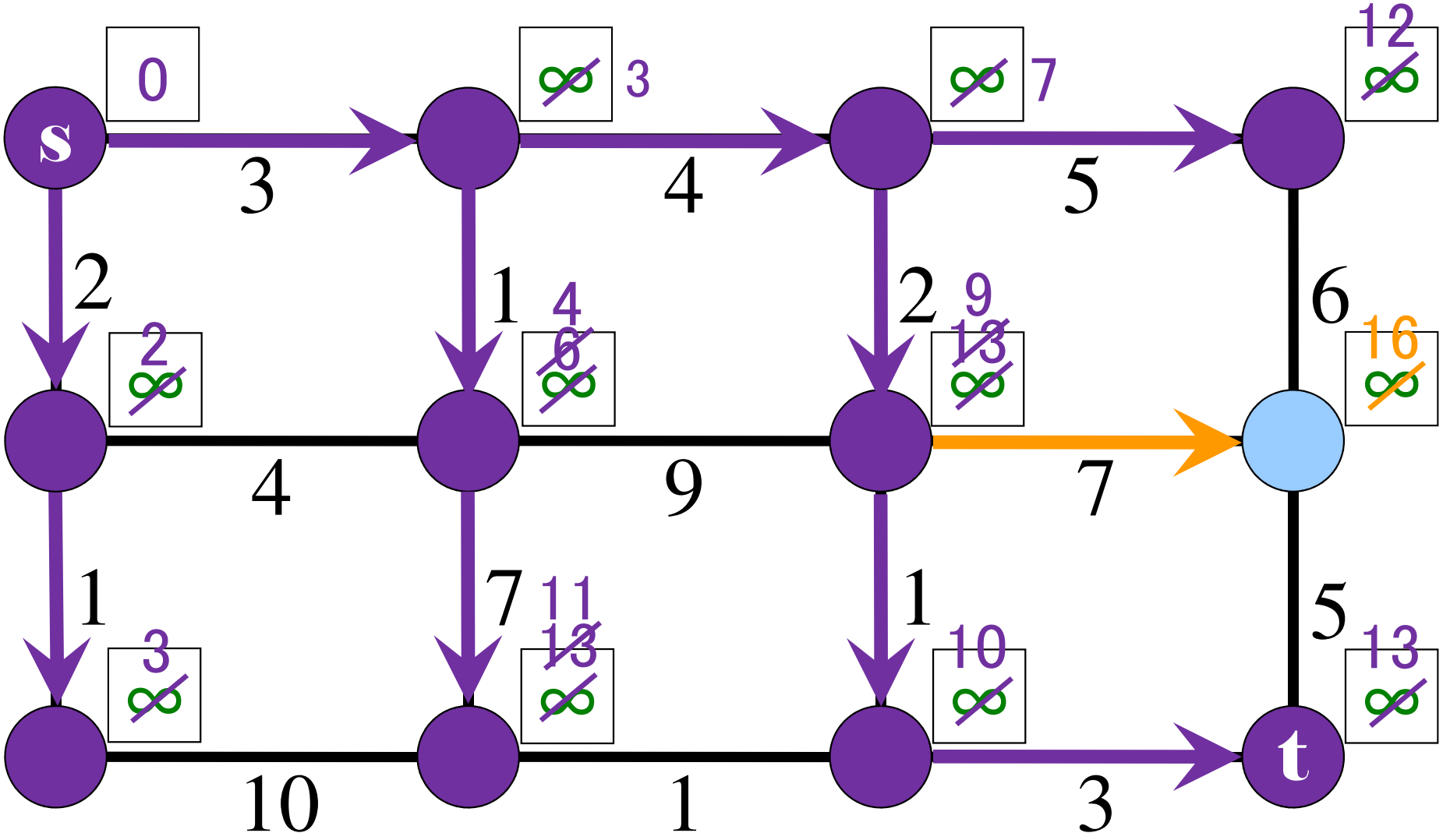


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

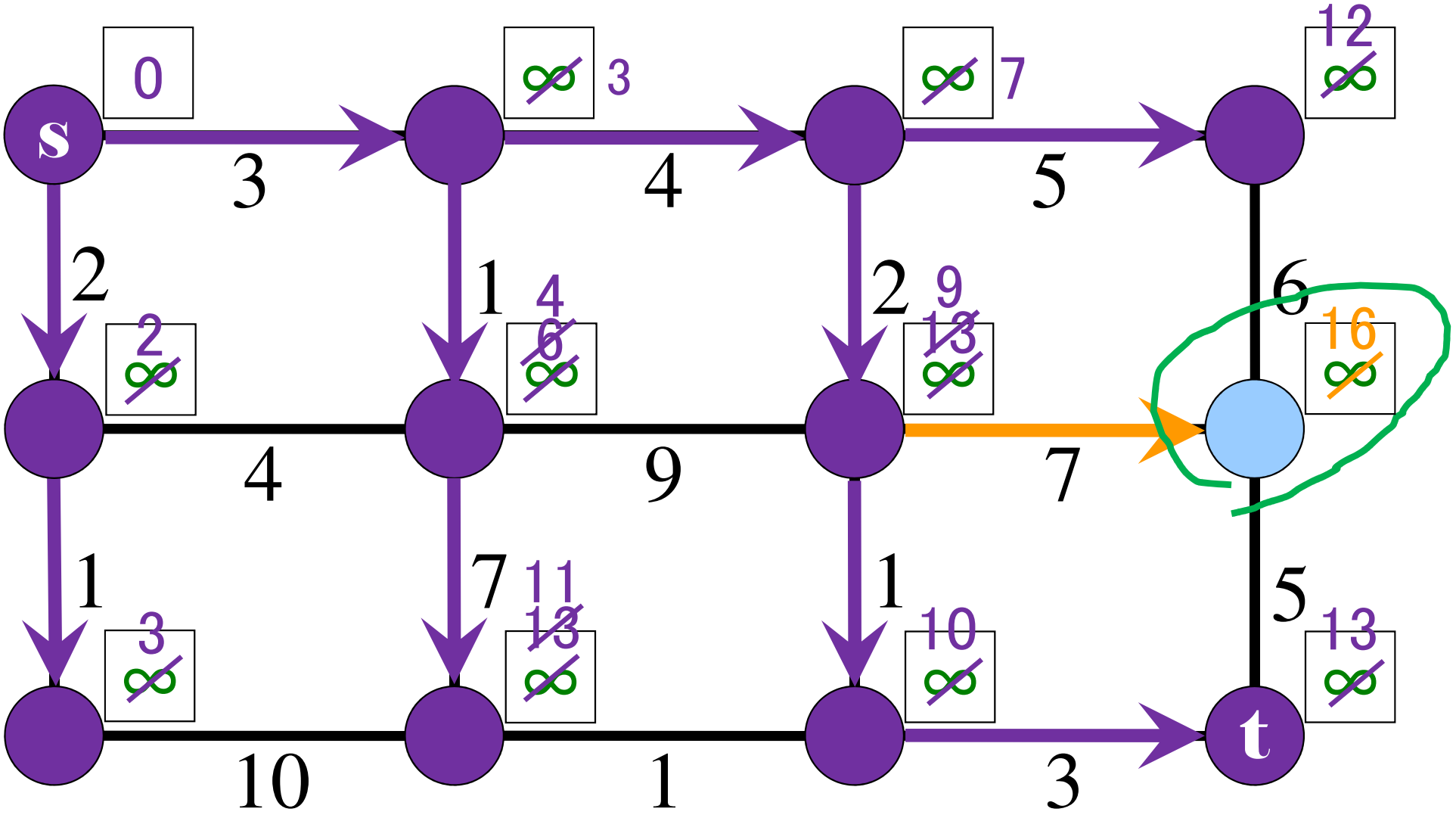
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



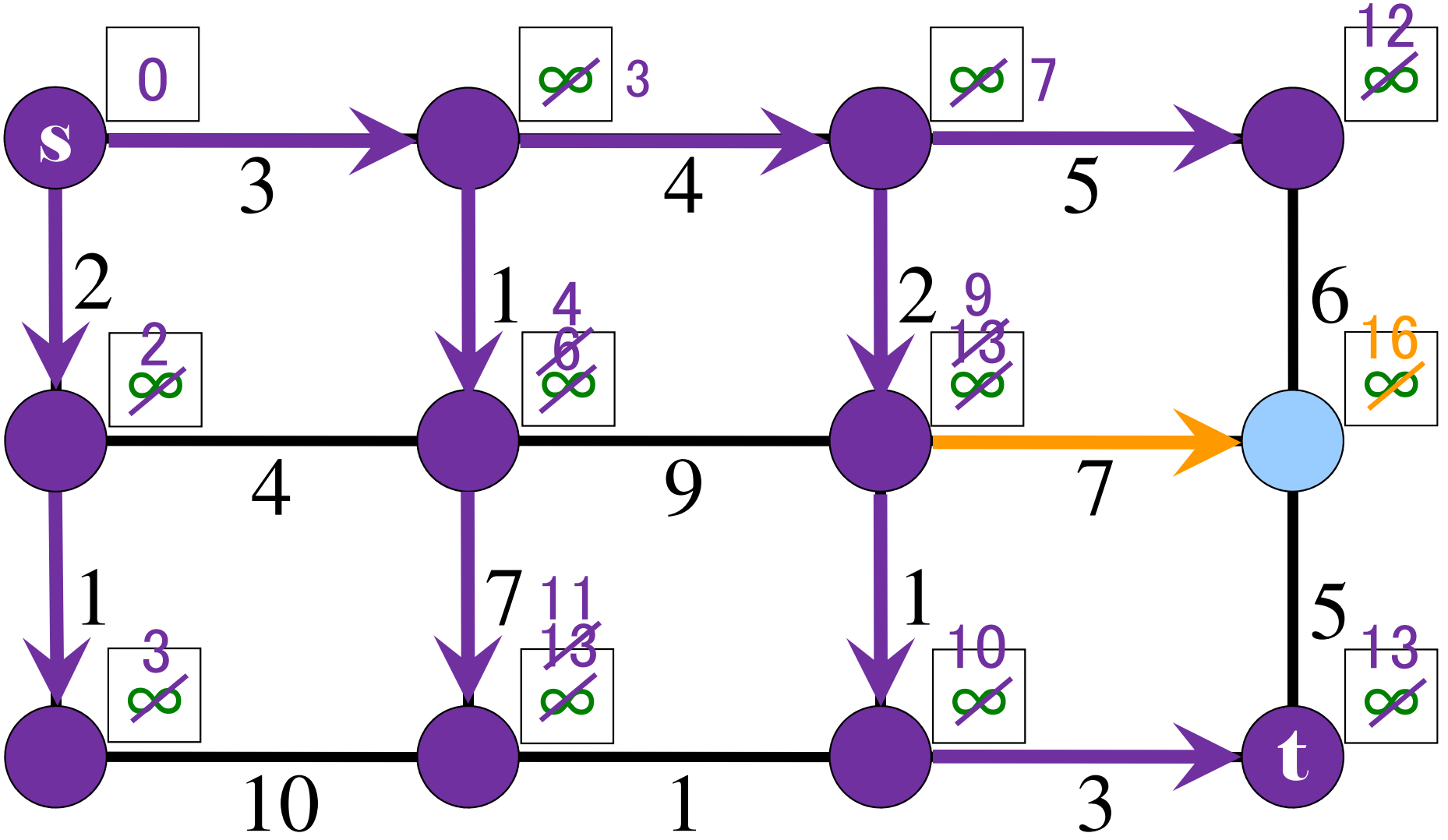
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

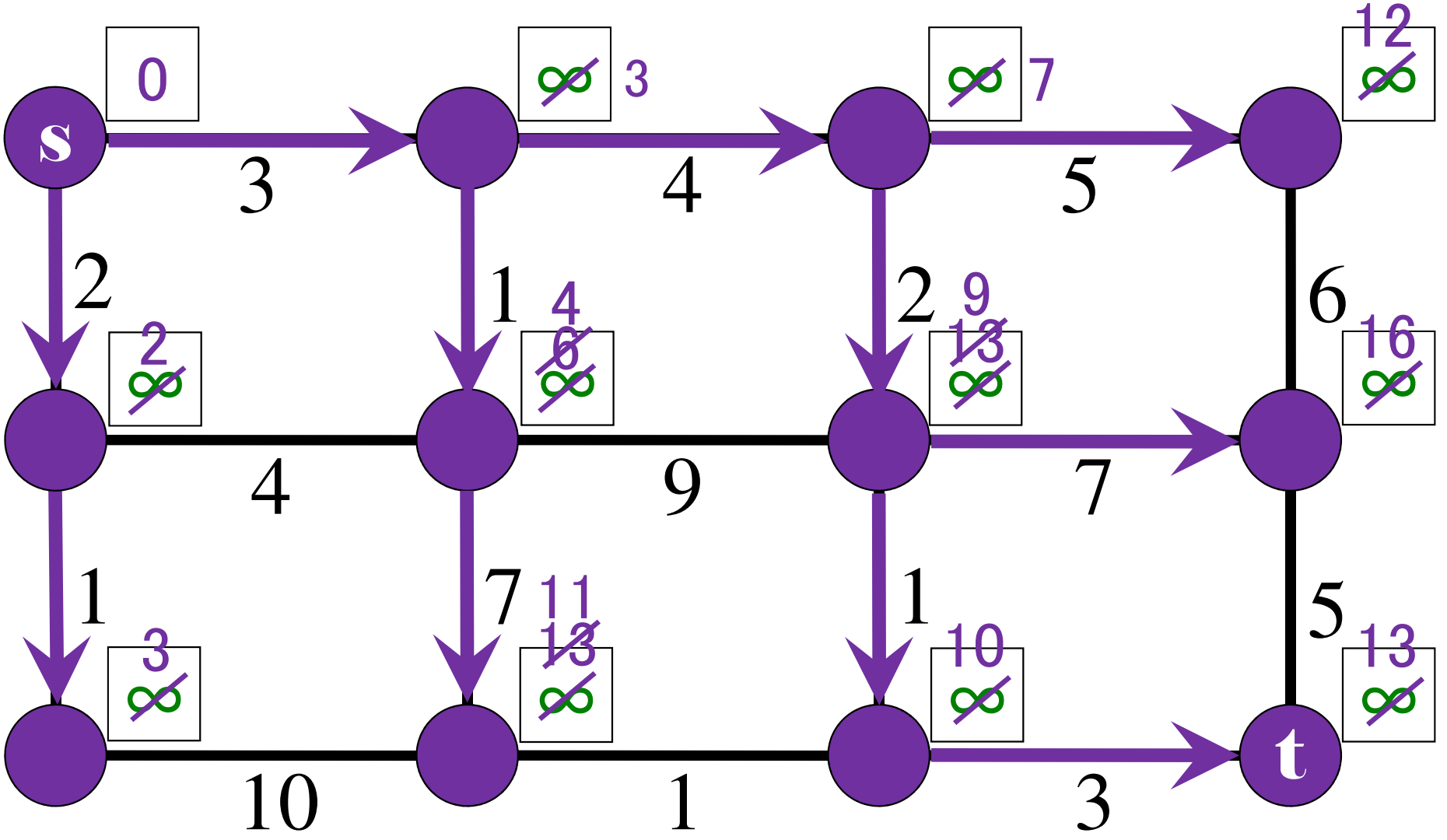


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

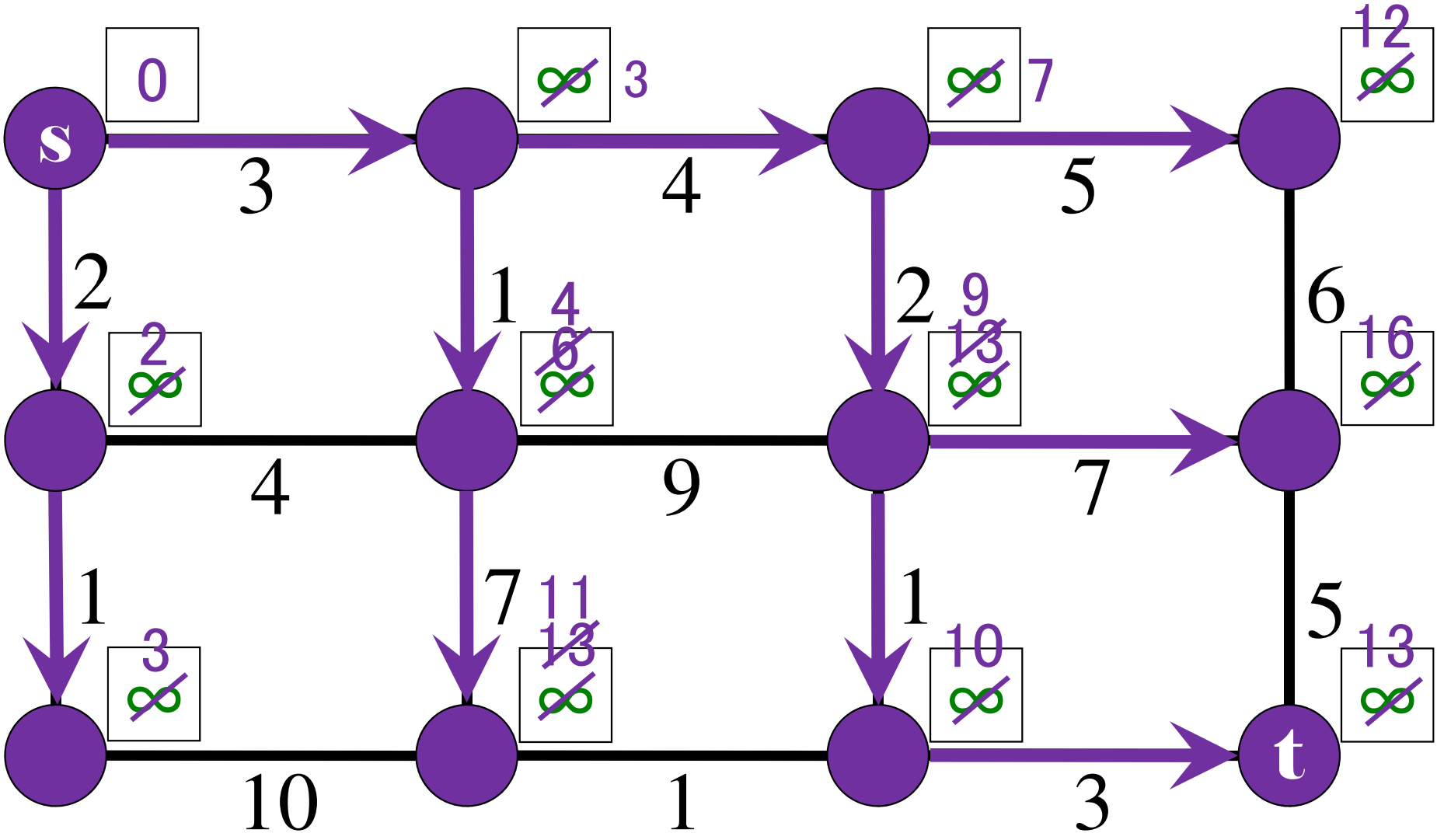
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



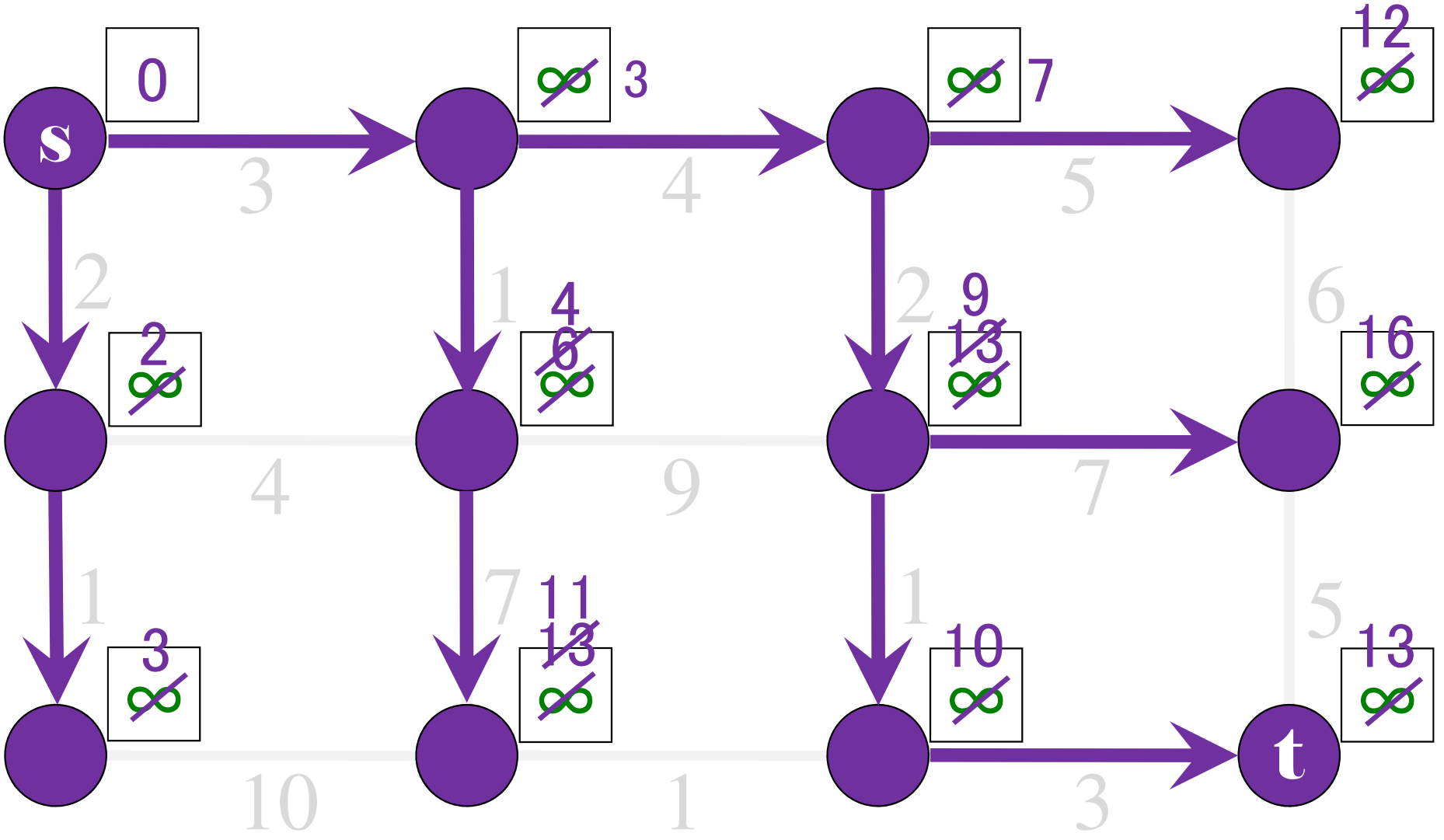
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



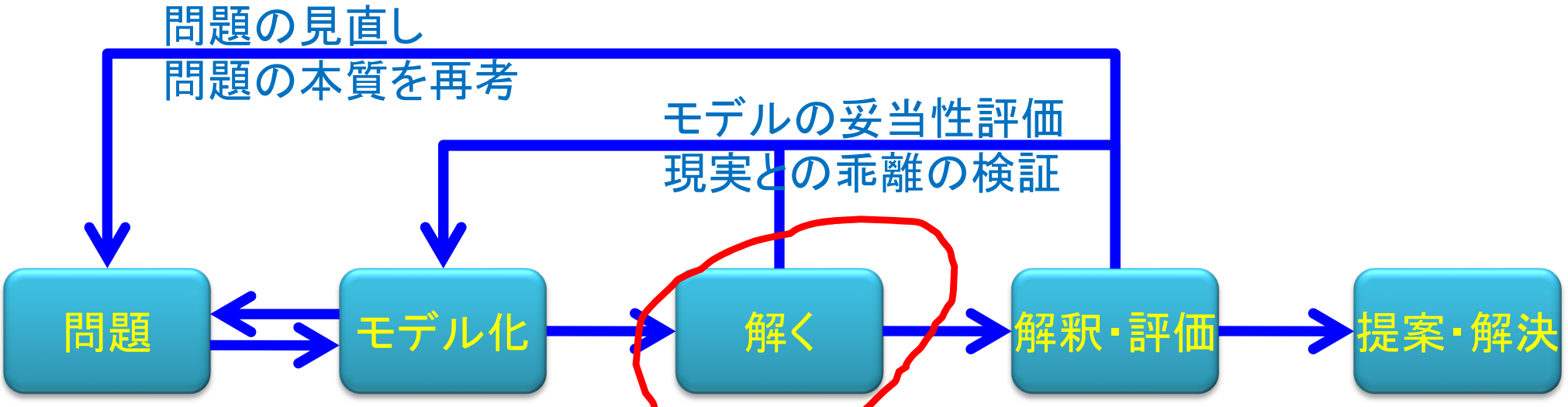
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

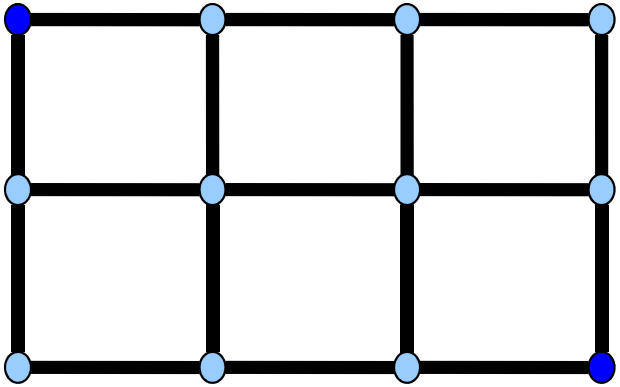
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

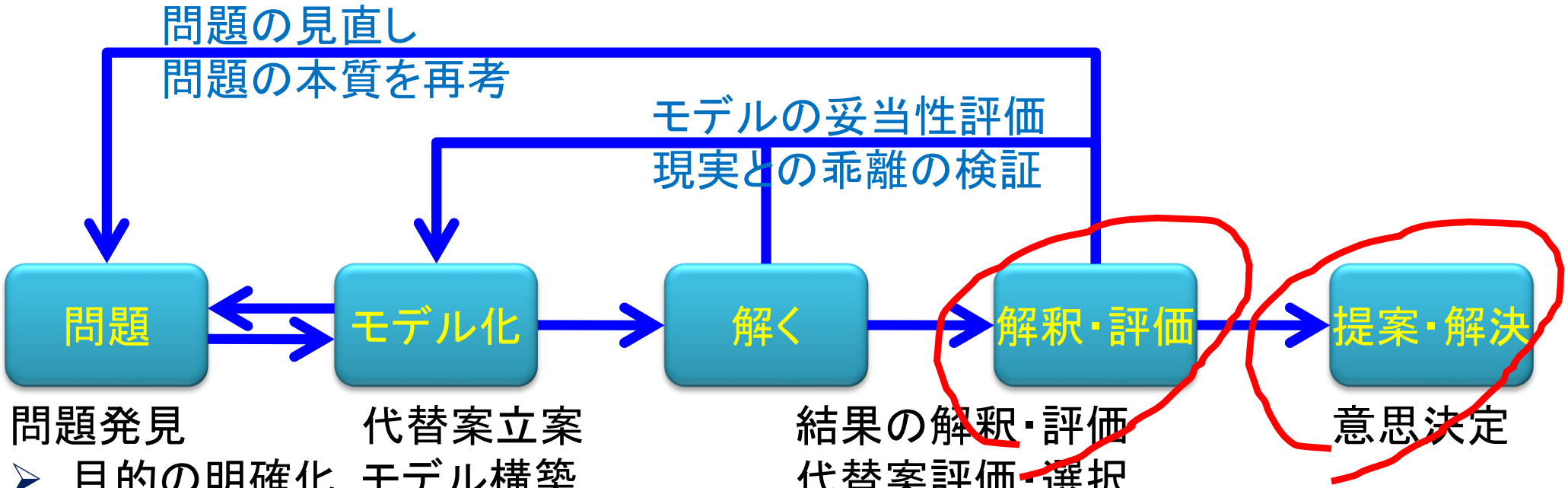
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

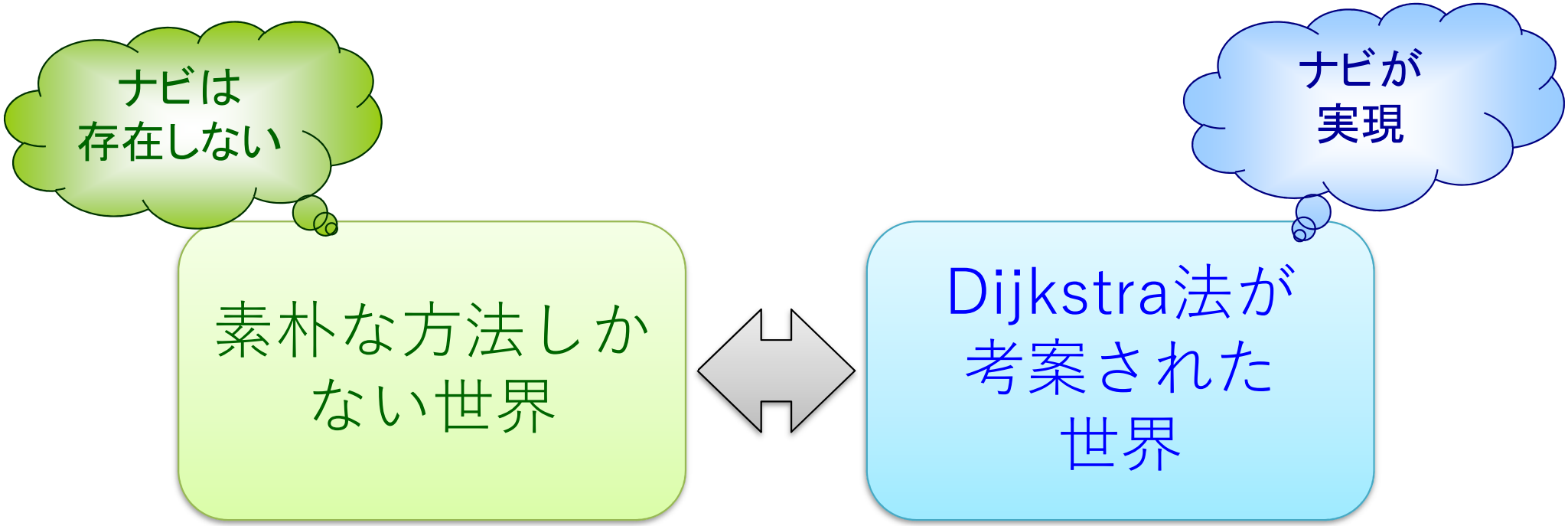
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

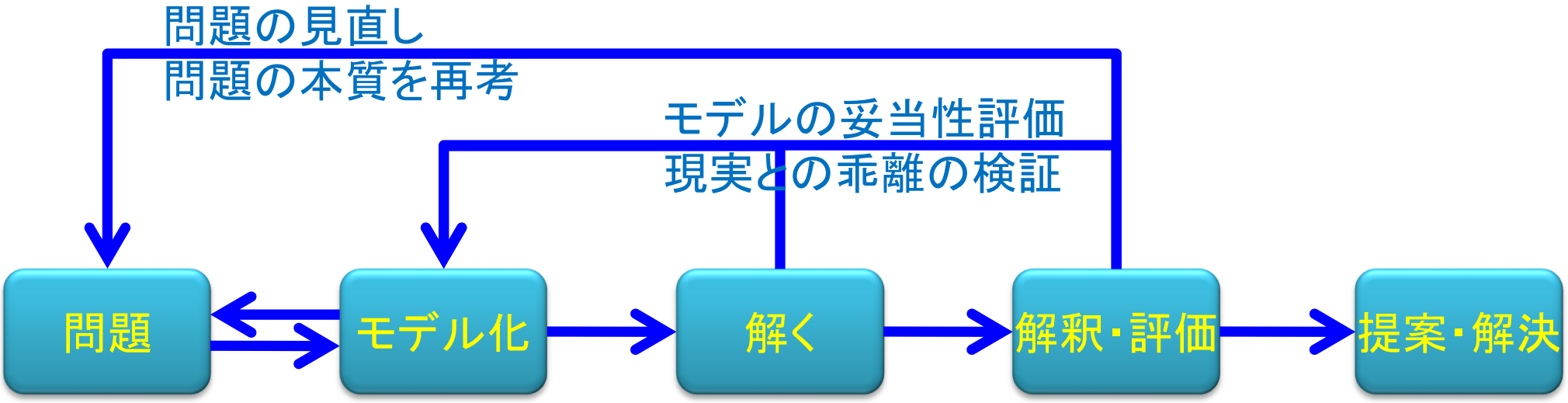
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



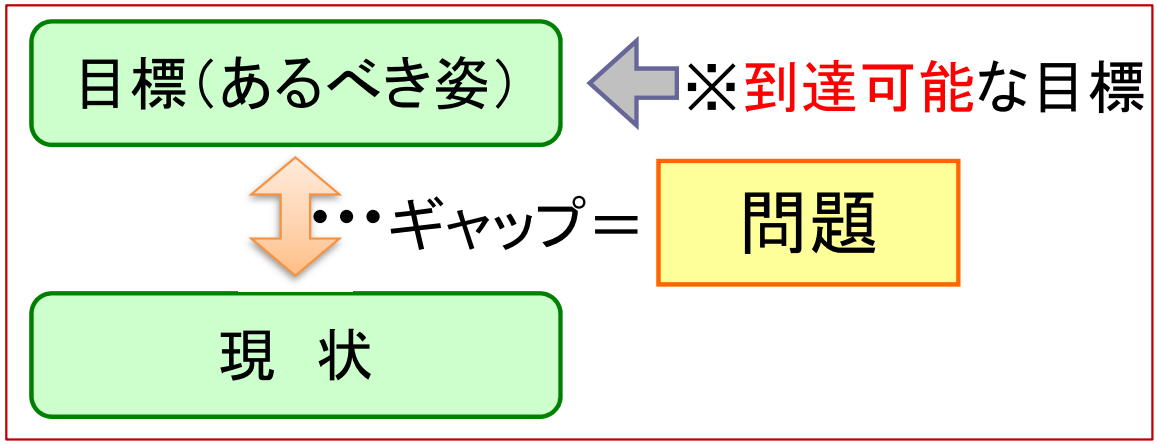
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

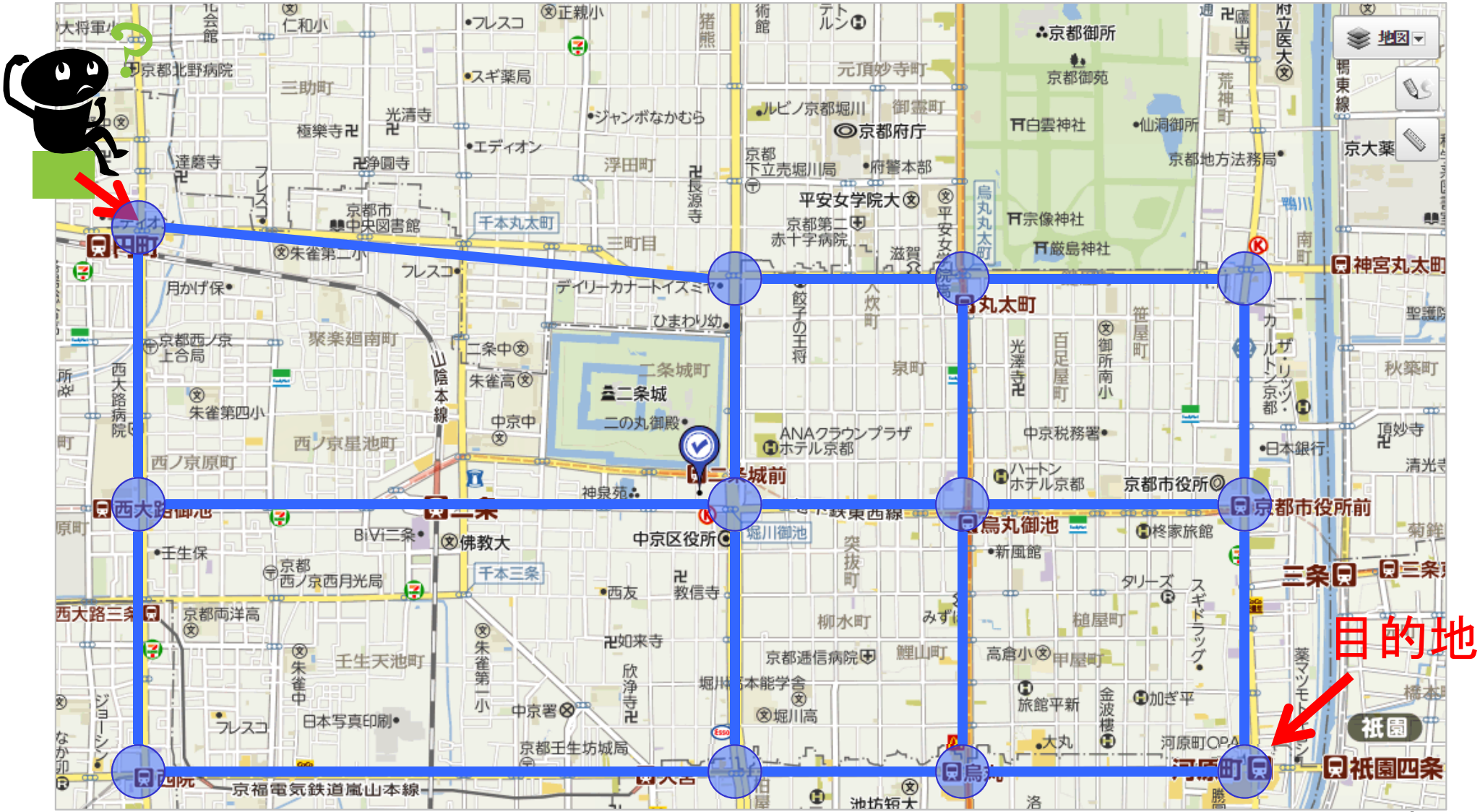
意思決定

問題の定義



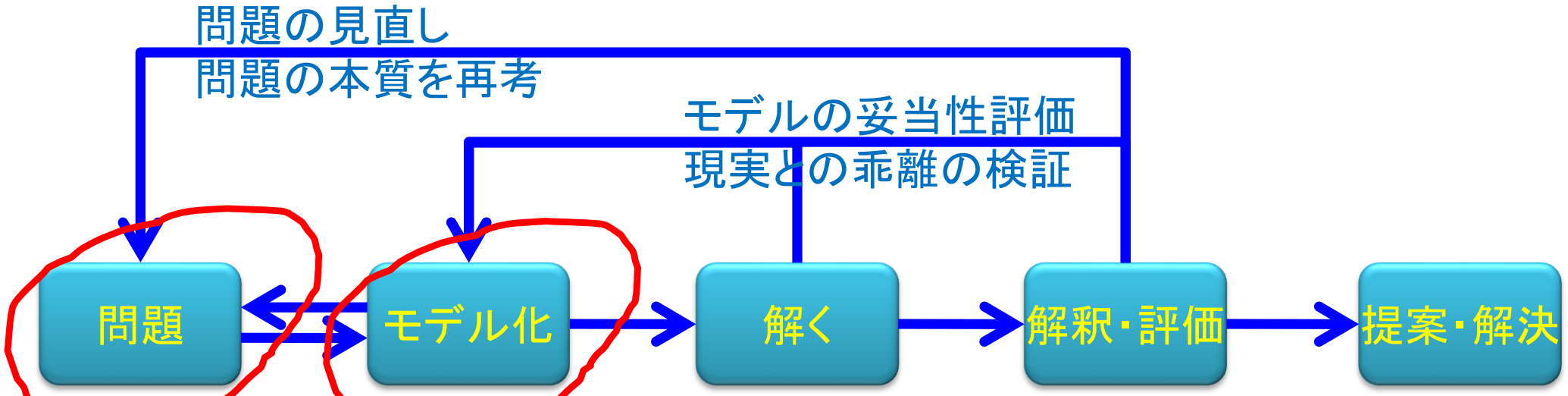
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

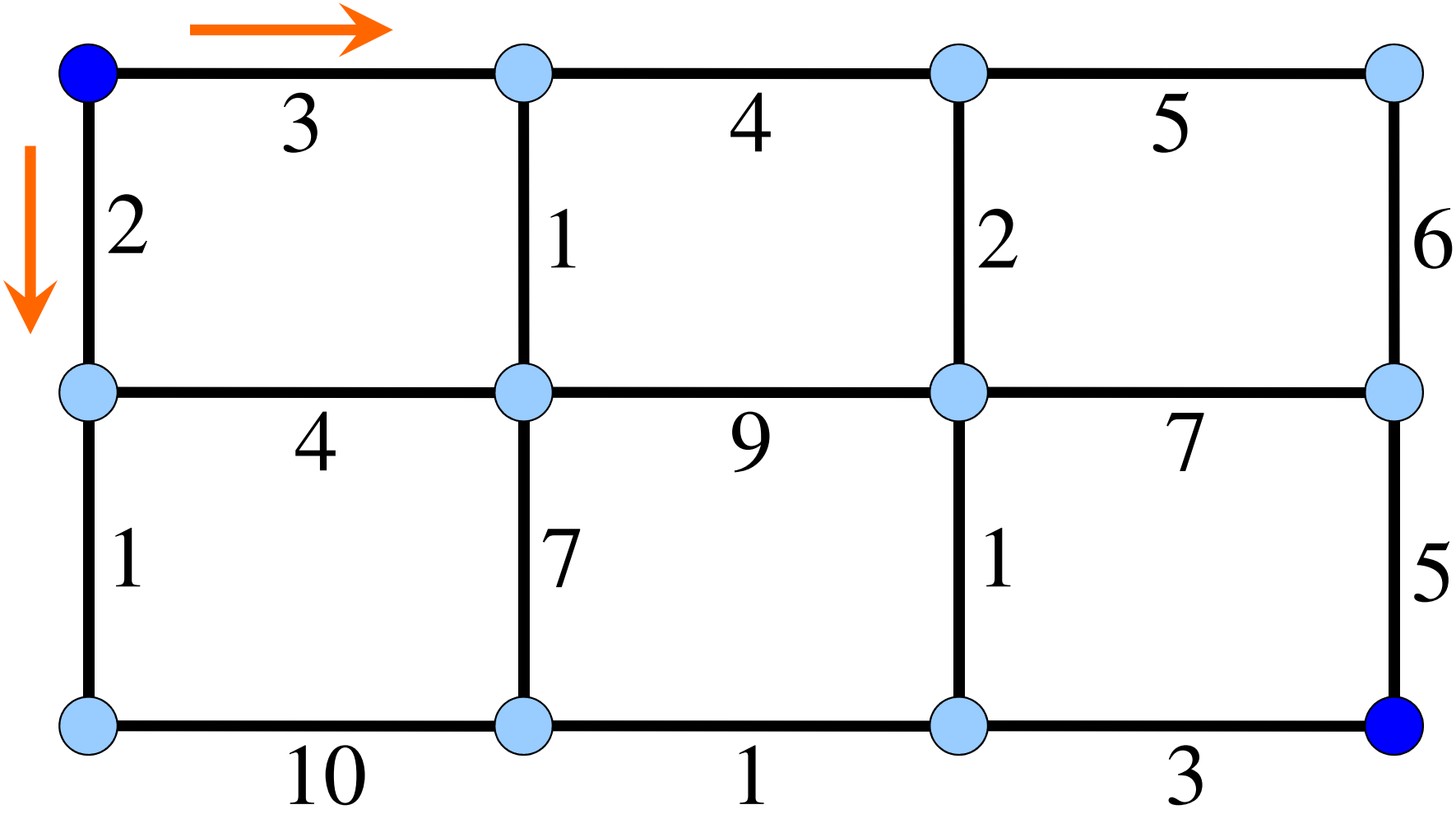
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

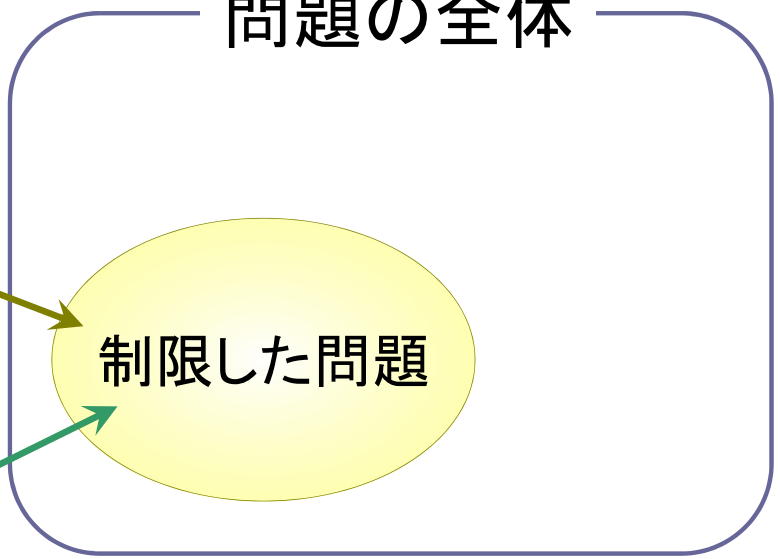


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

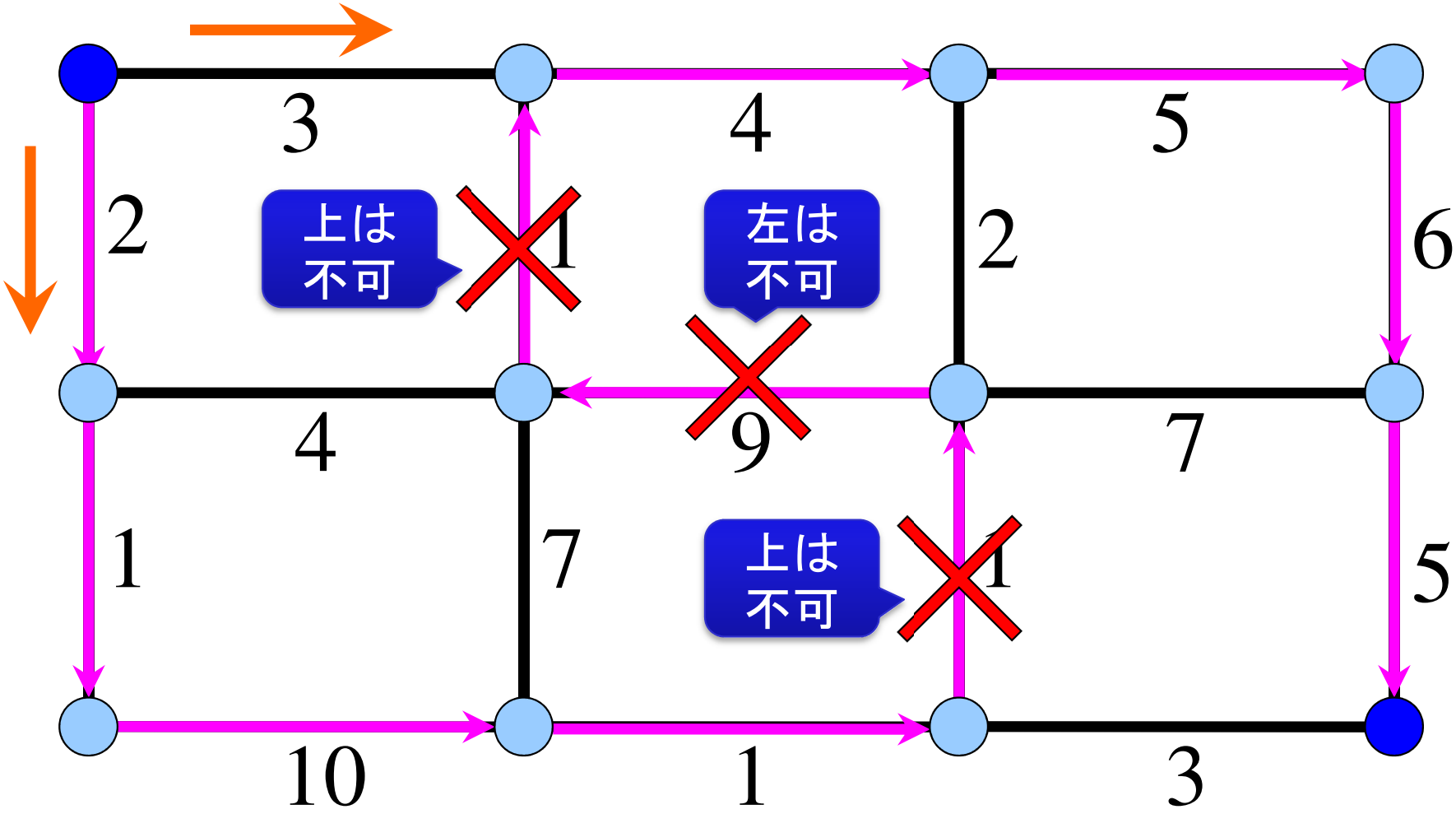
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

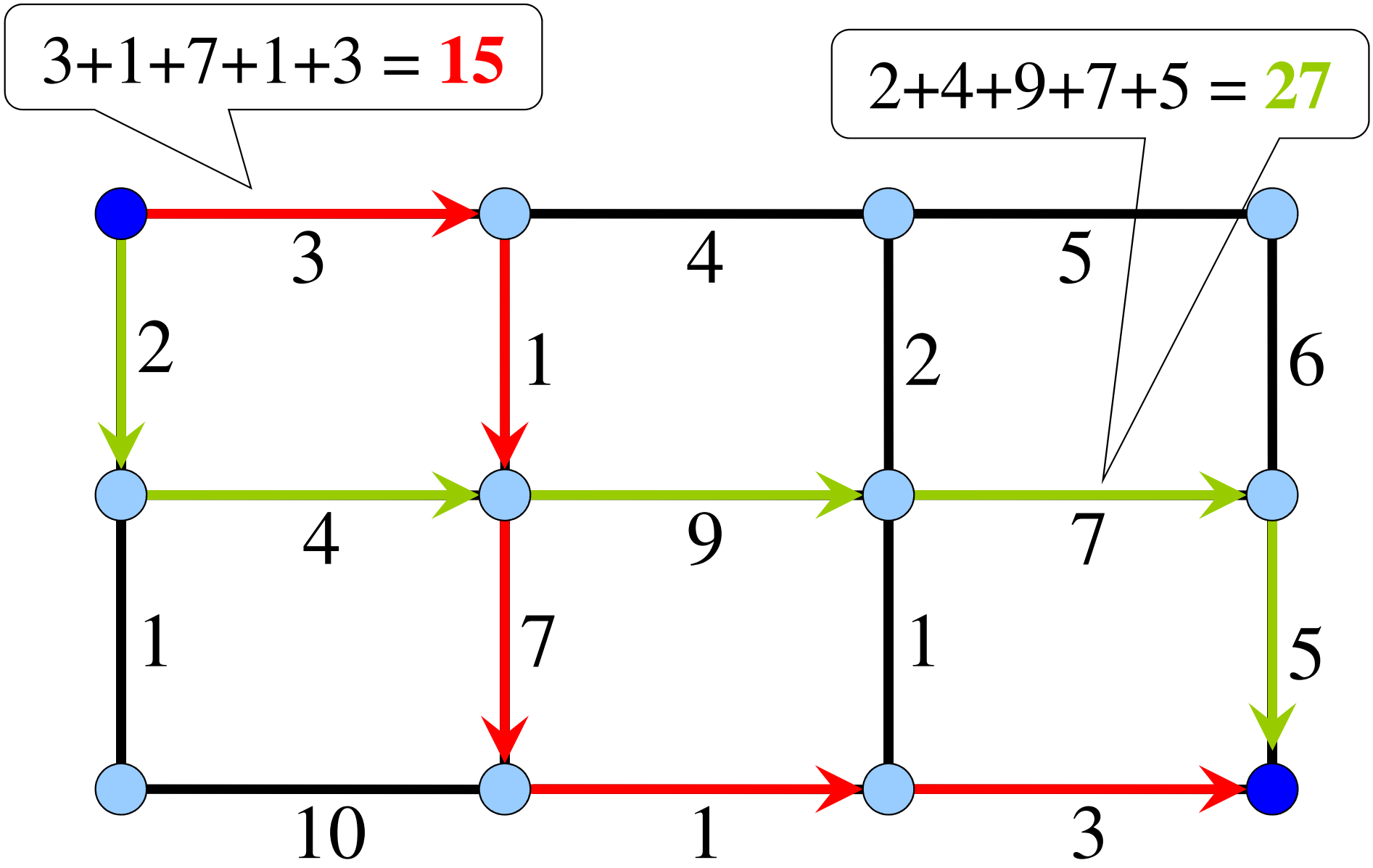
難しいなら易しくすればいいのさ！

制限した問題

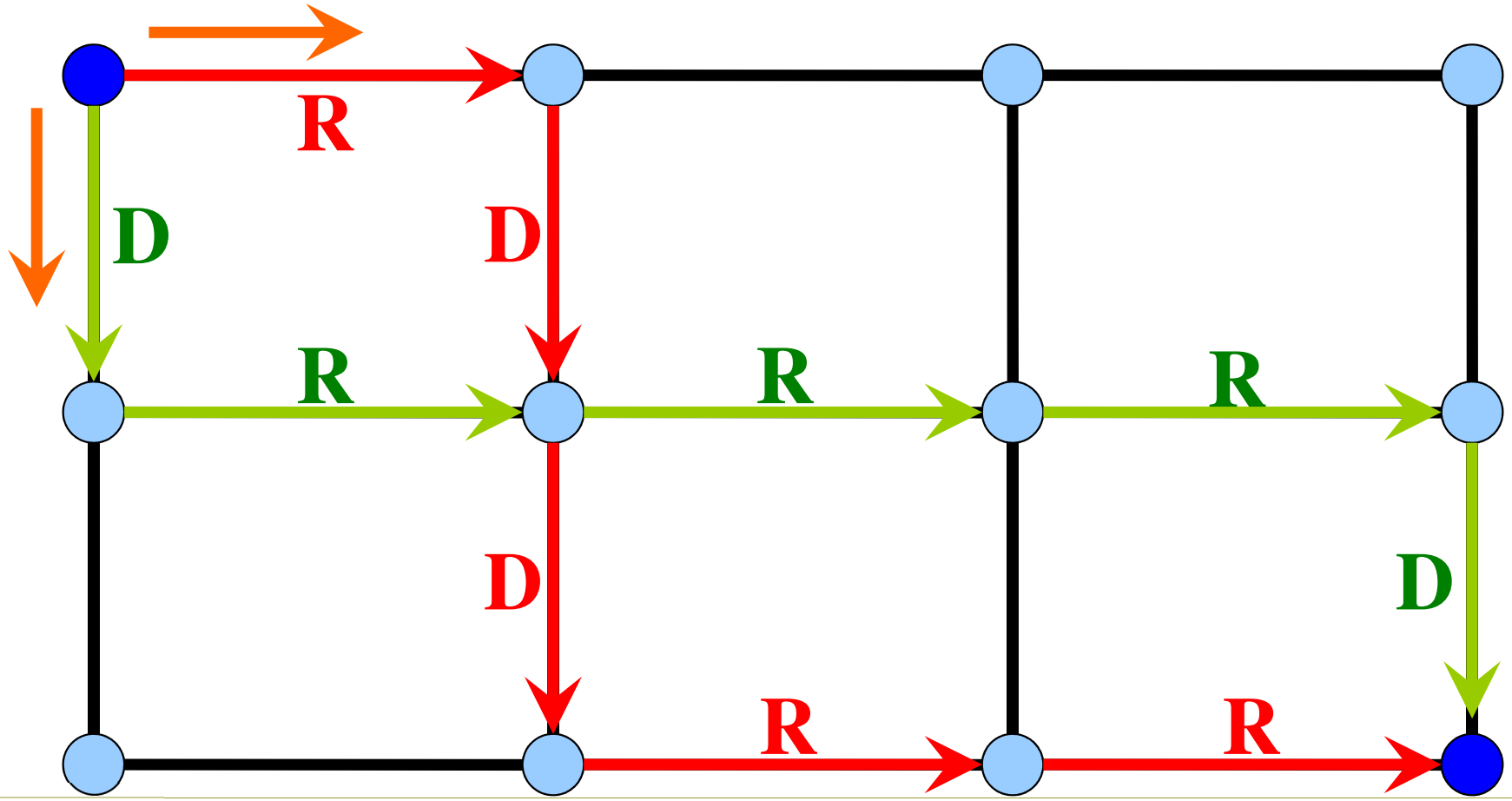
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



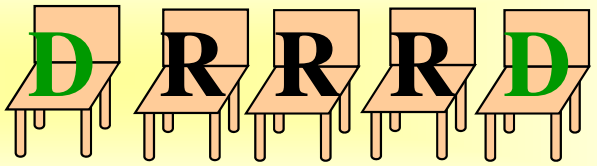
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



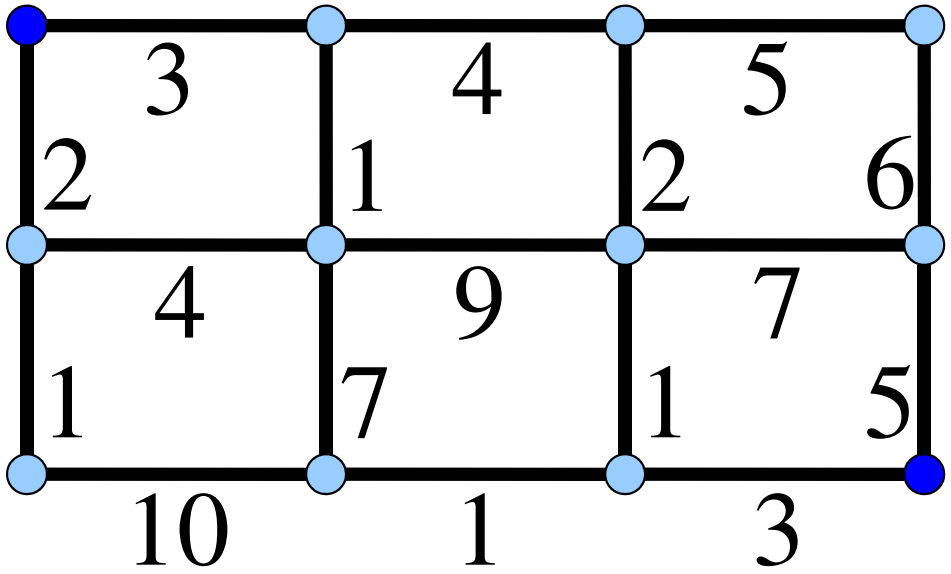
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

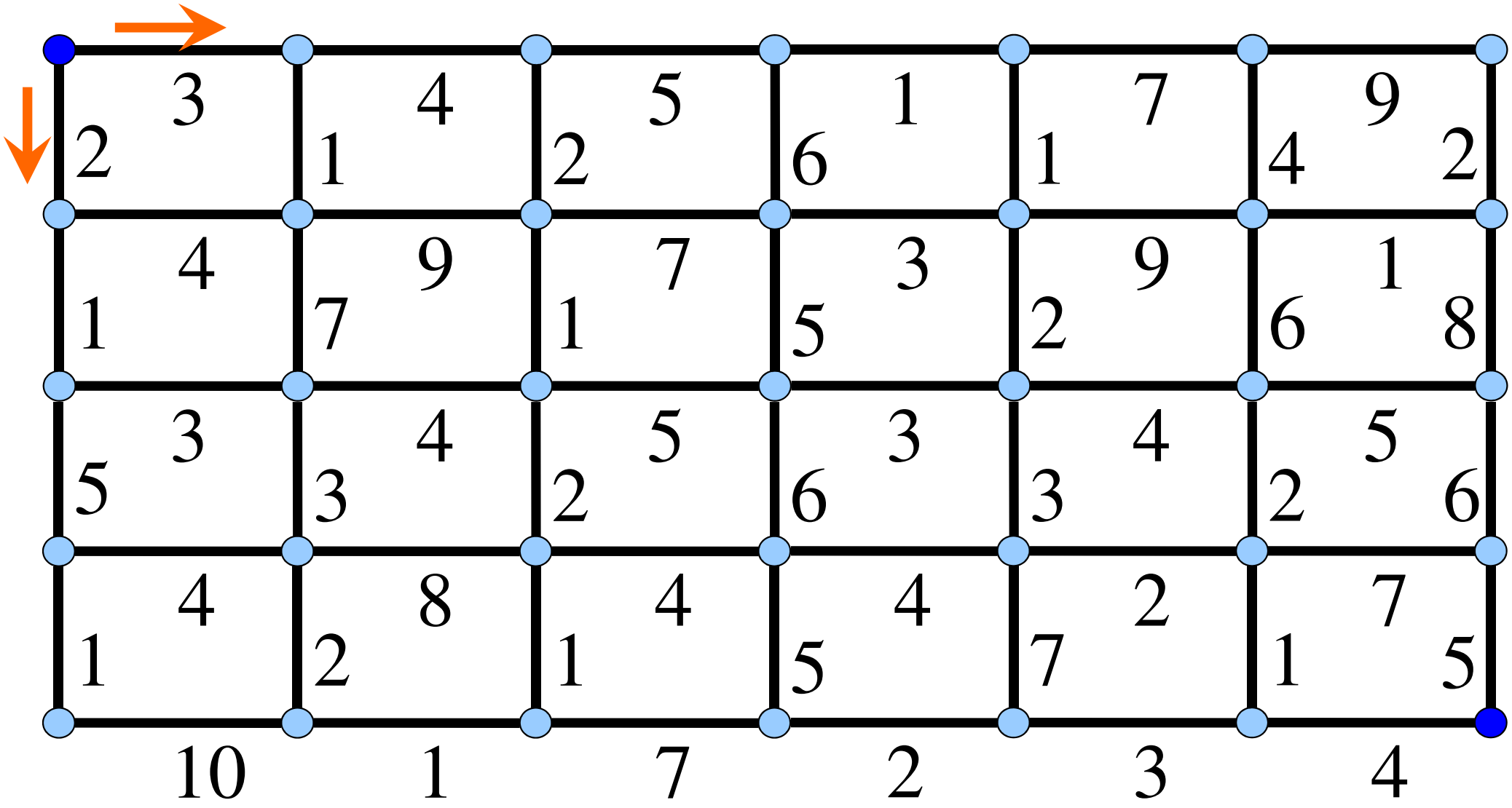
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

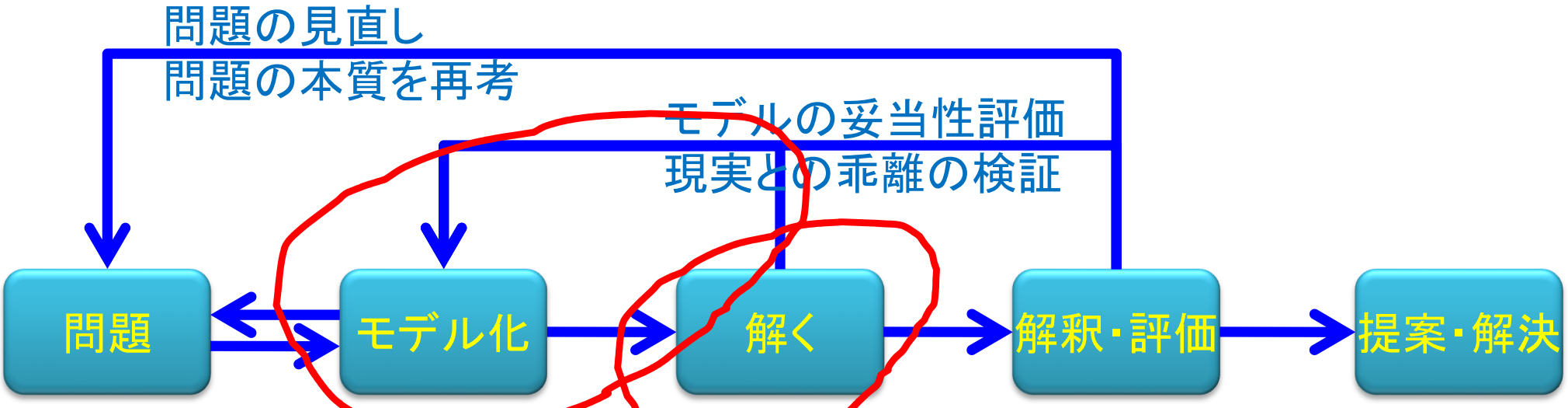
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

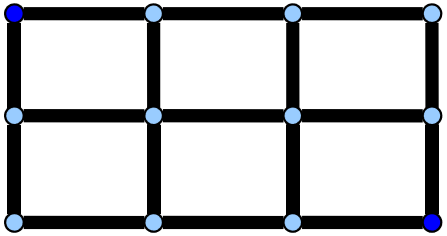
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

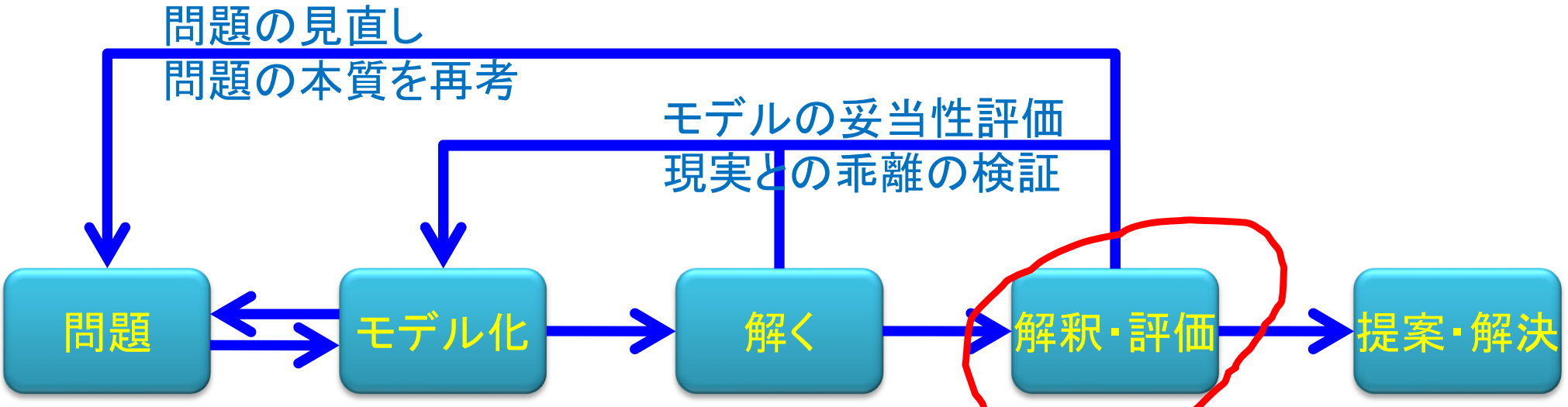
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

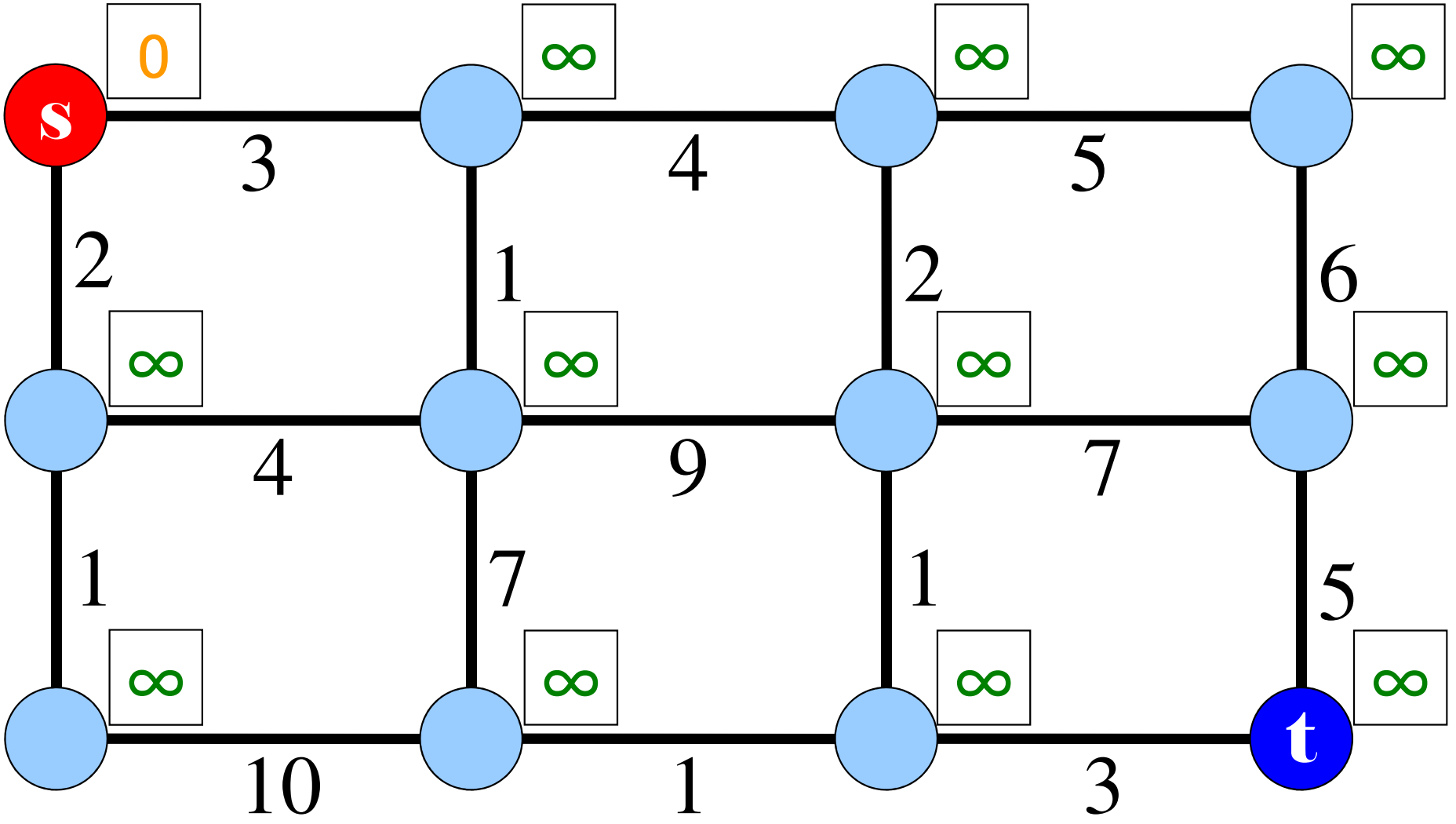
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

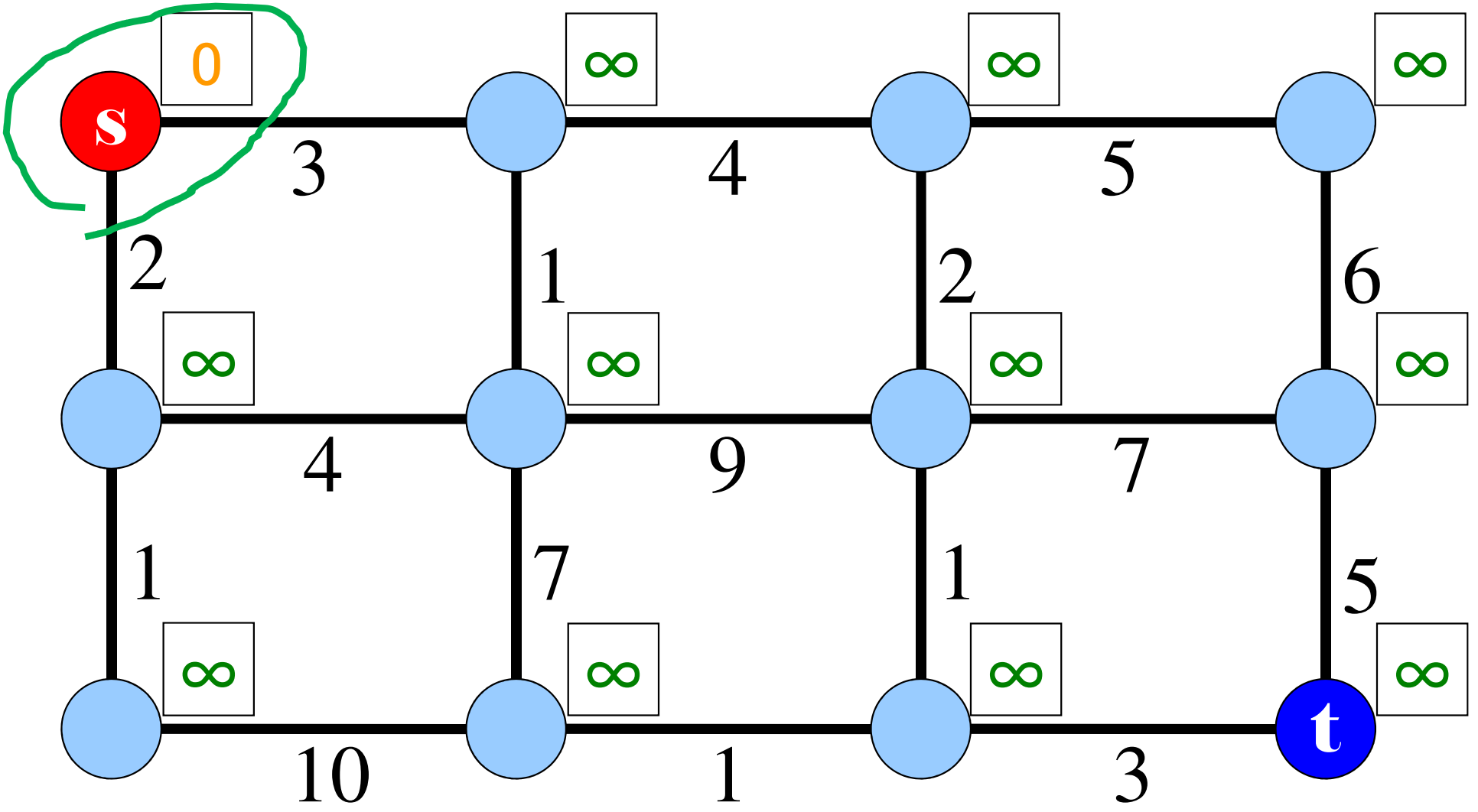
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



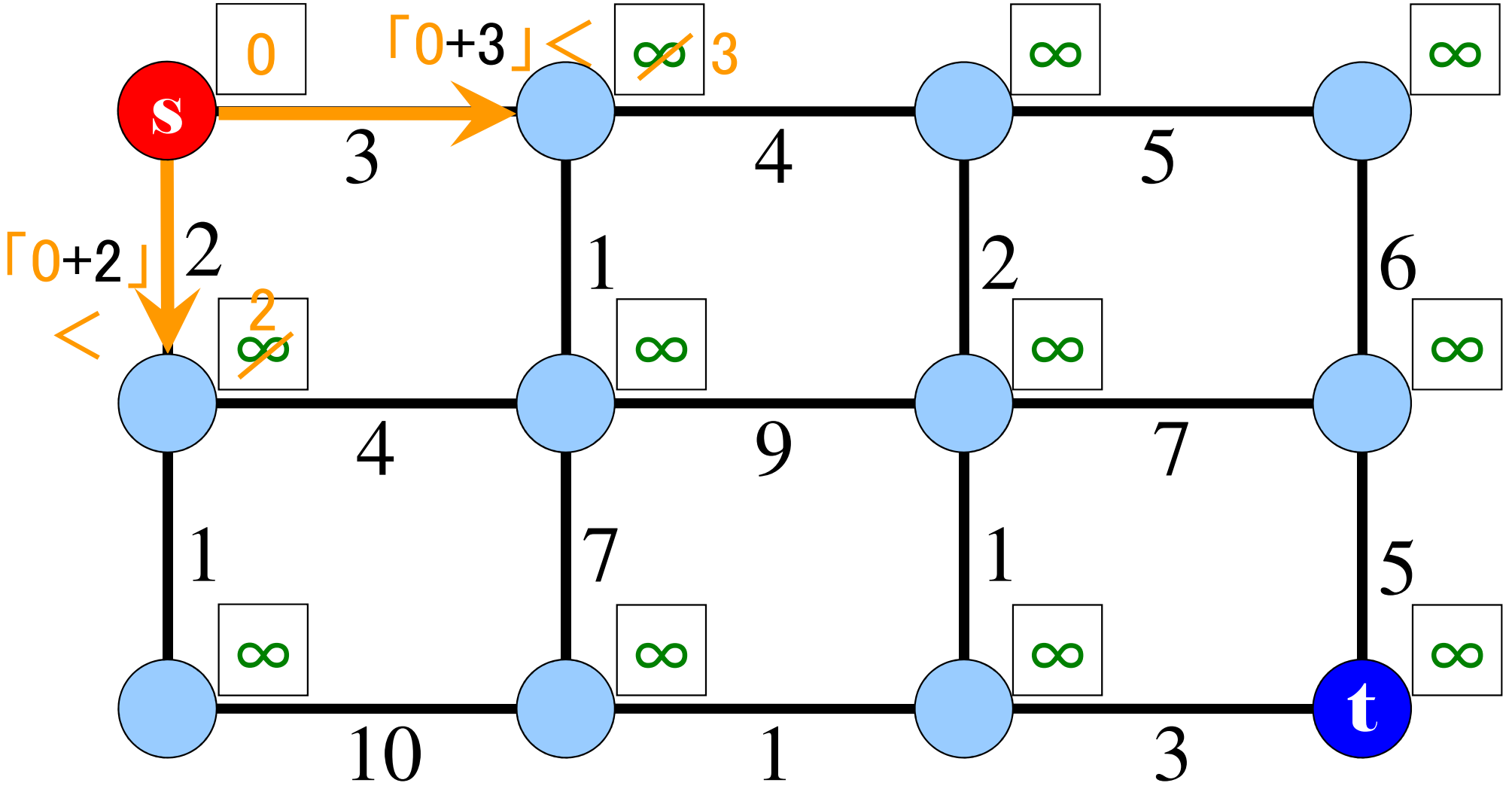
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

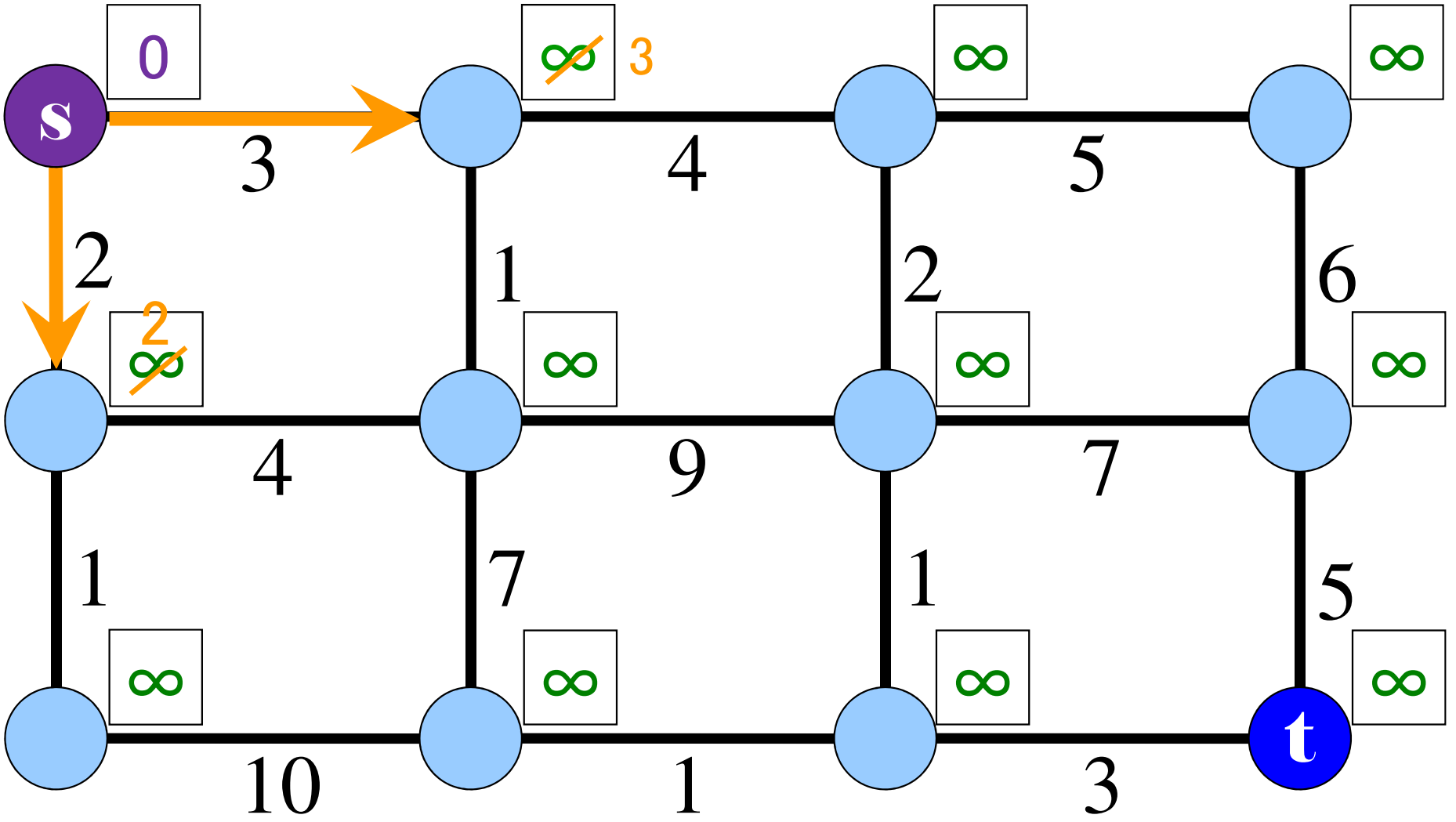


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

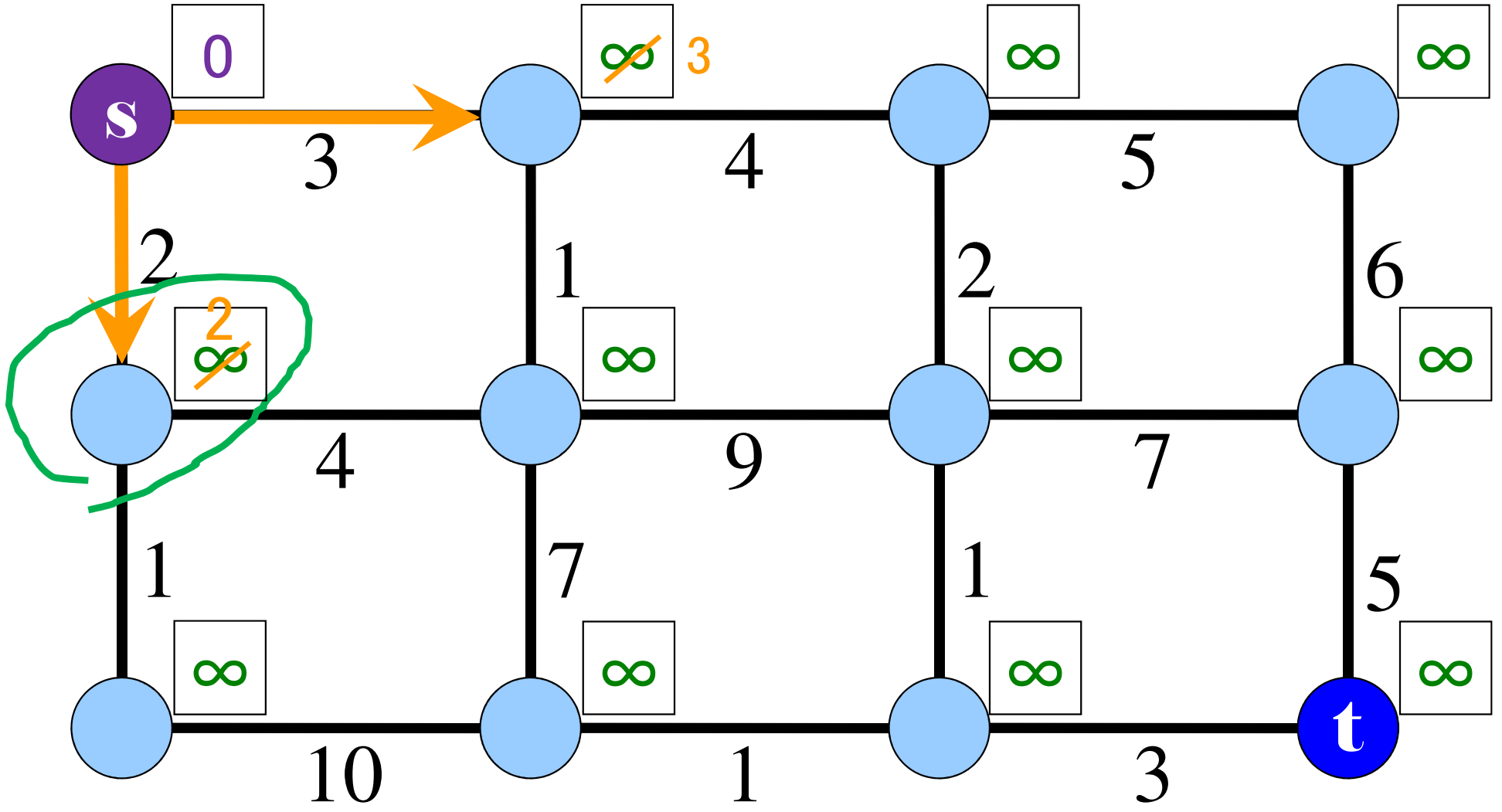
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



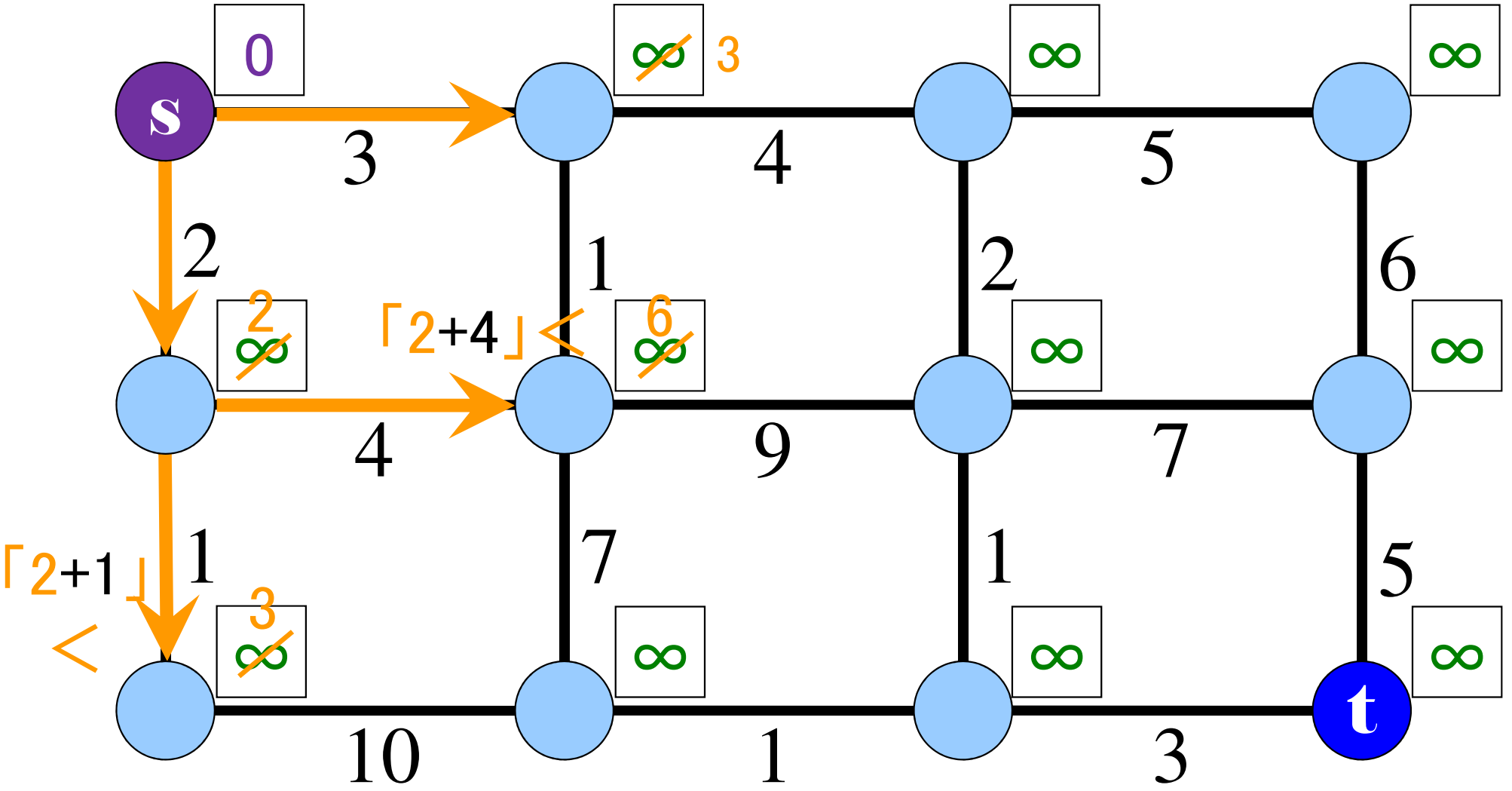
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

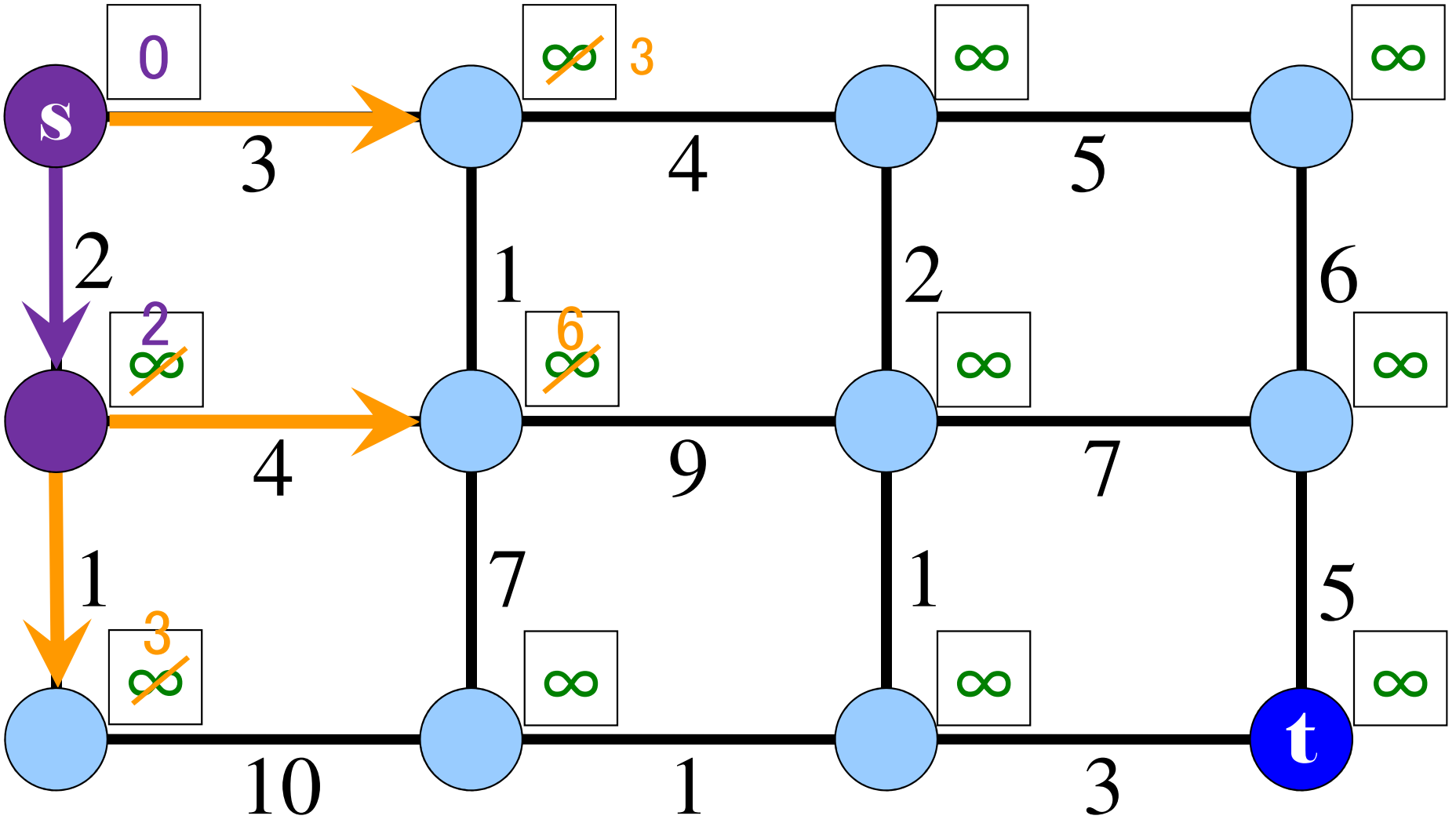


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

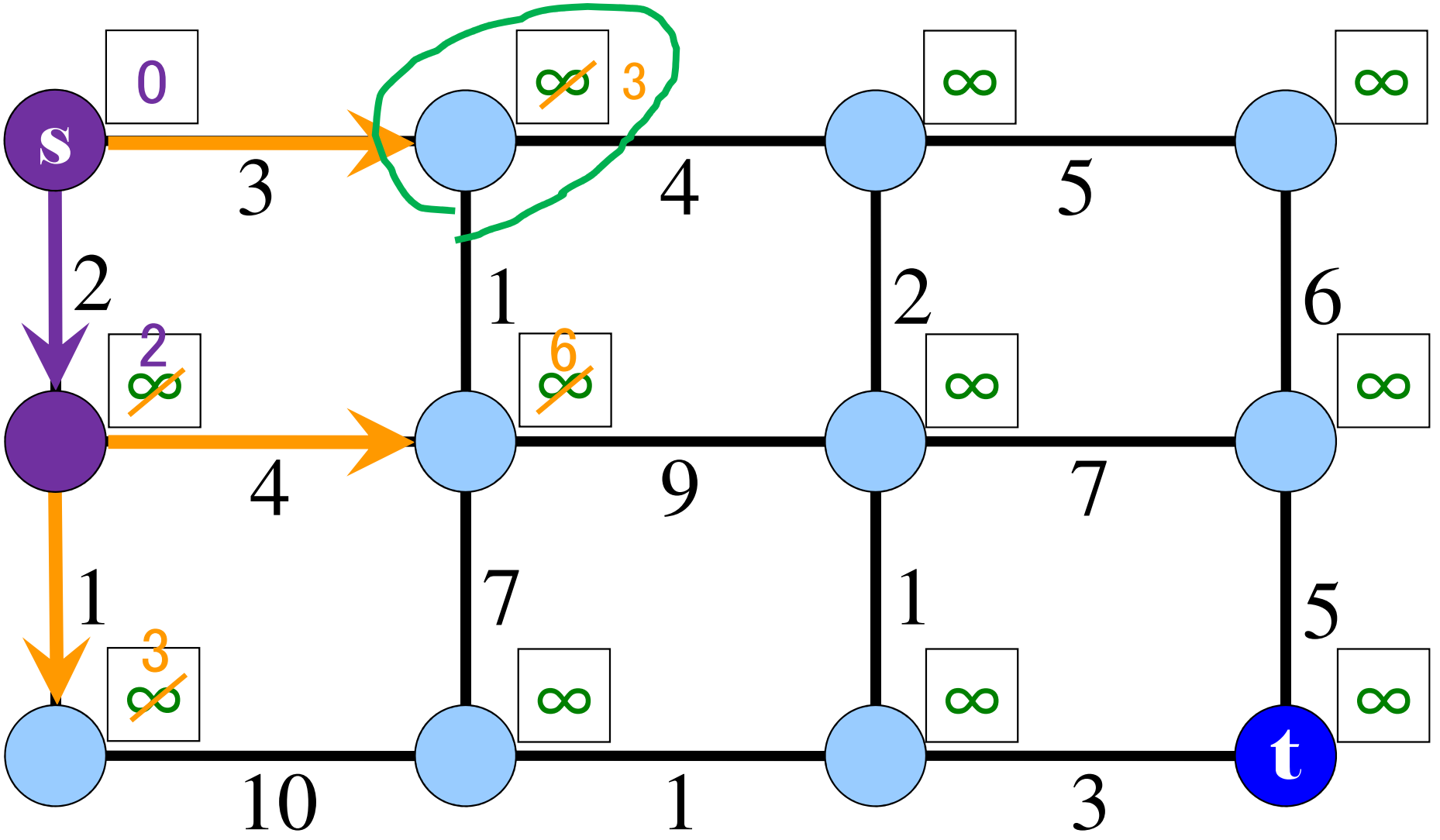
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



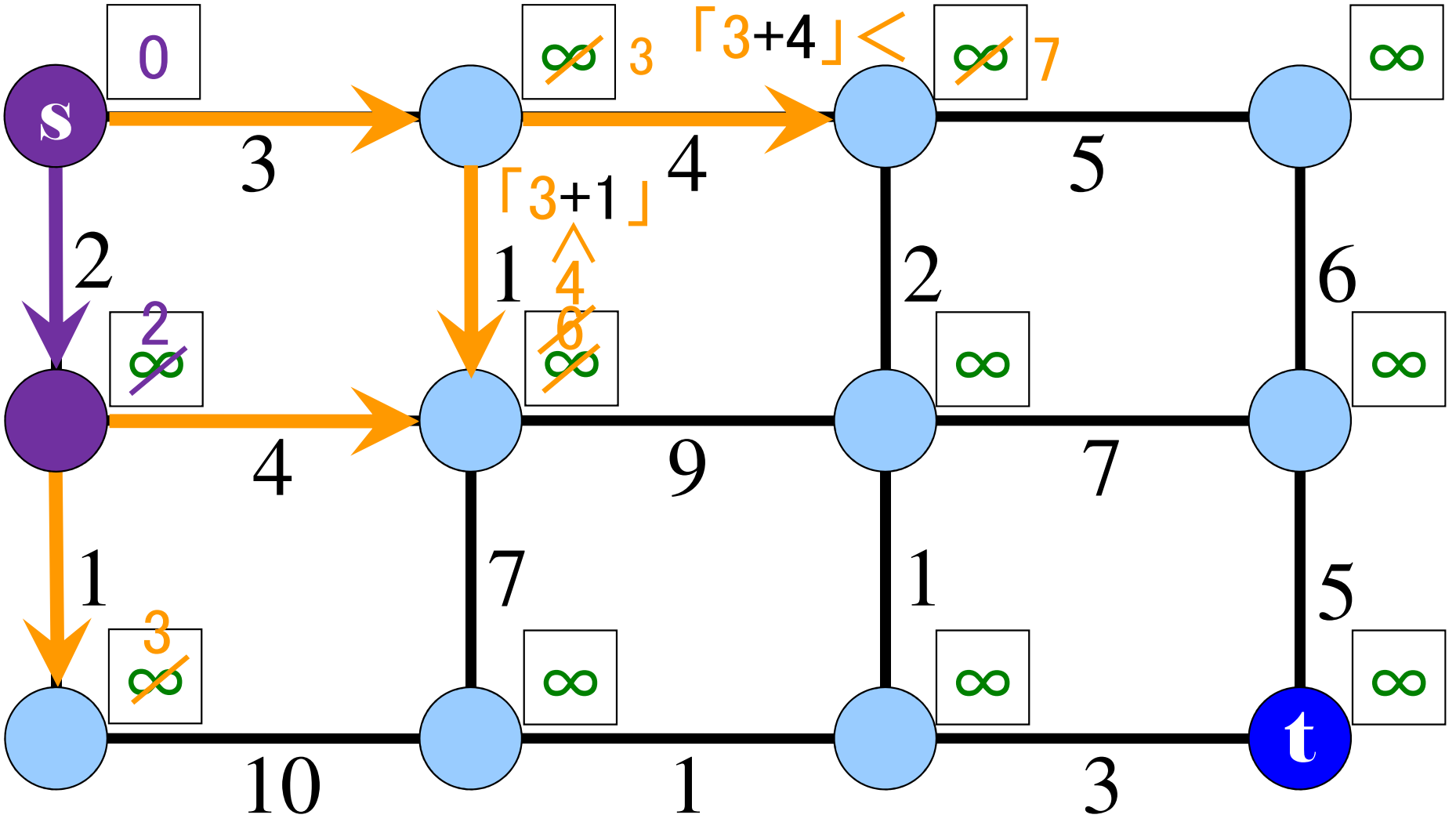
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

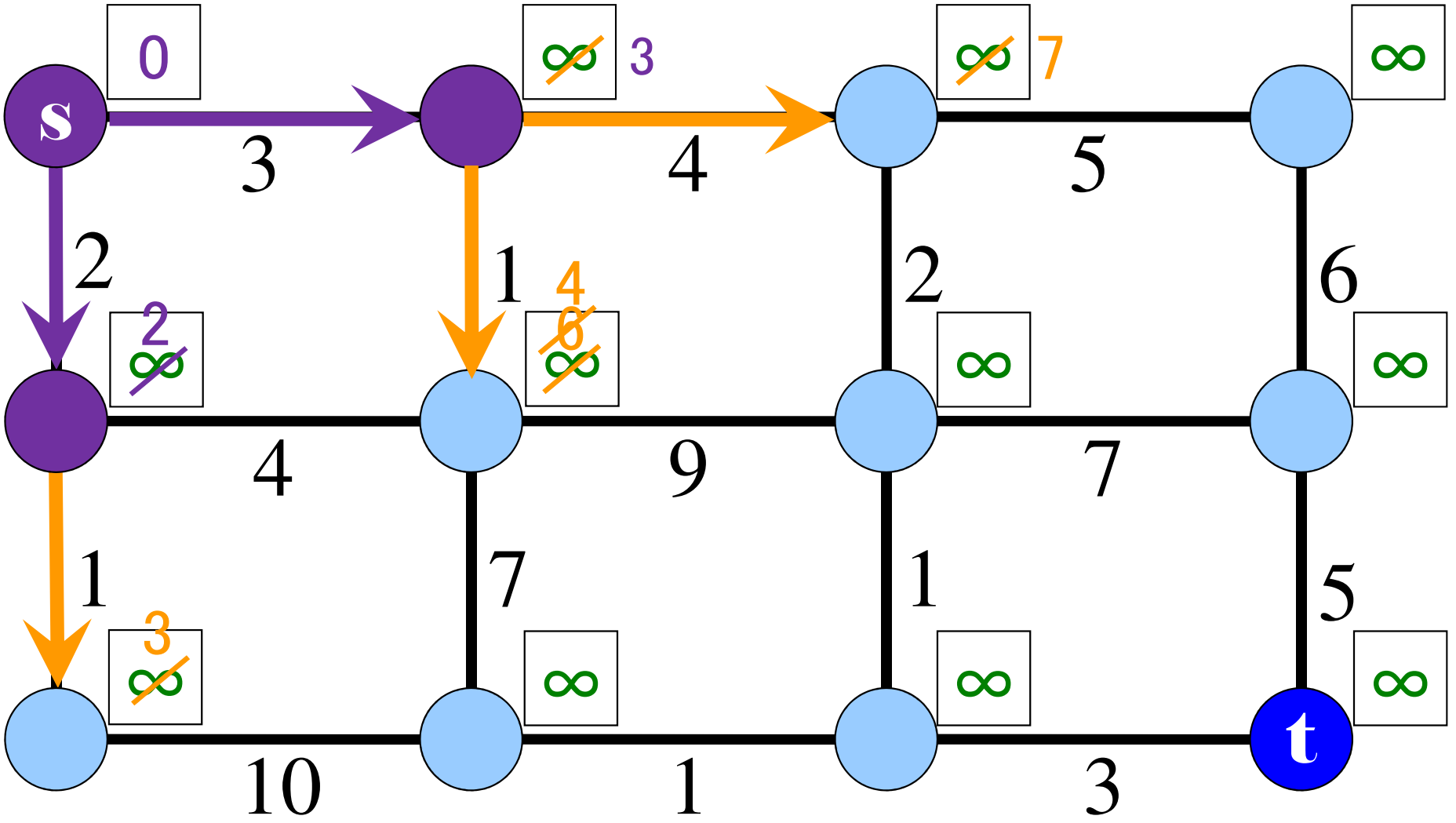


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

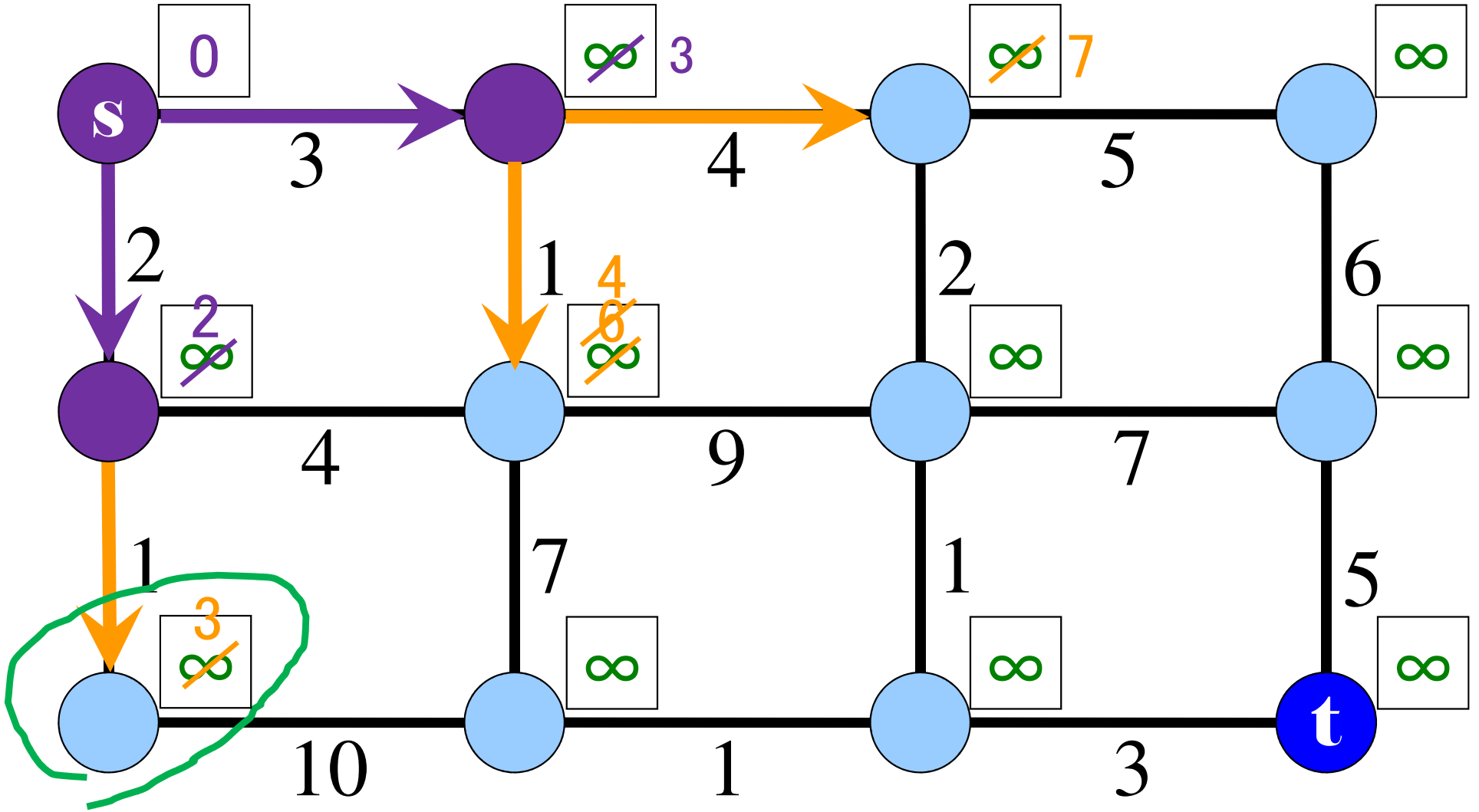
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



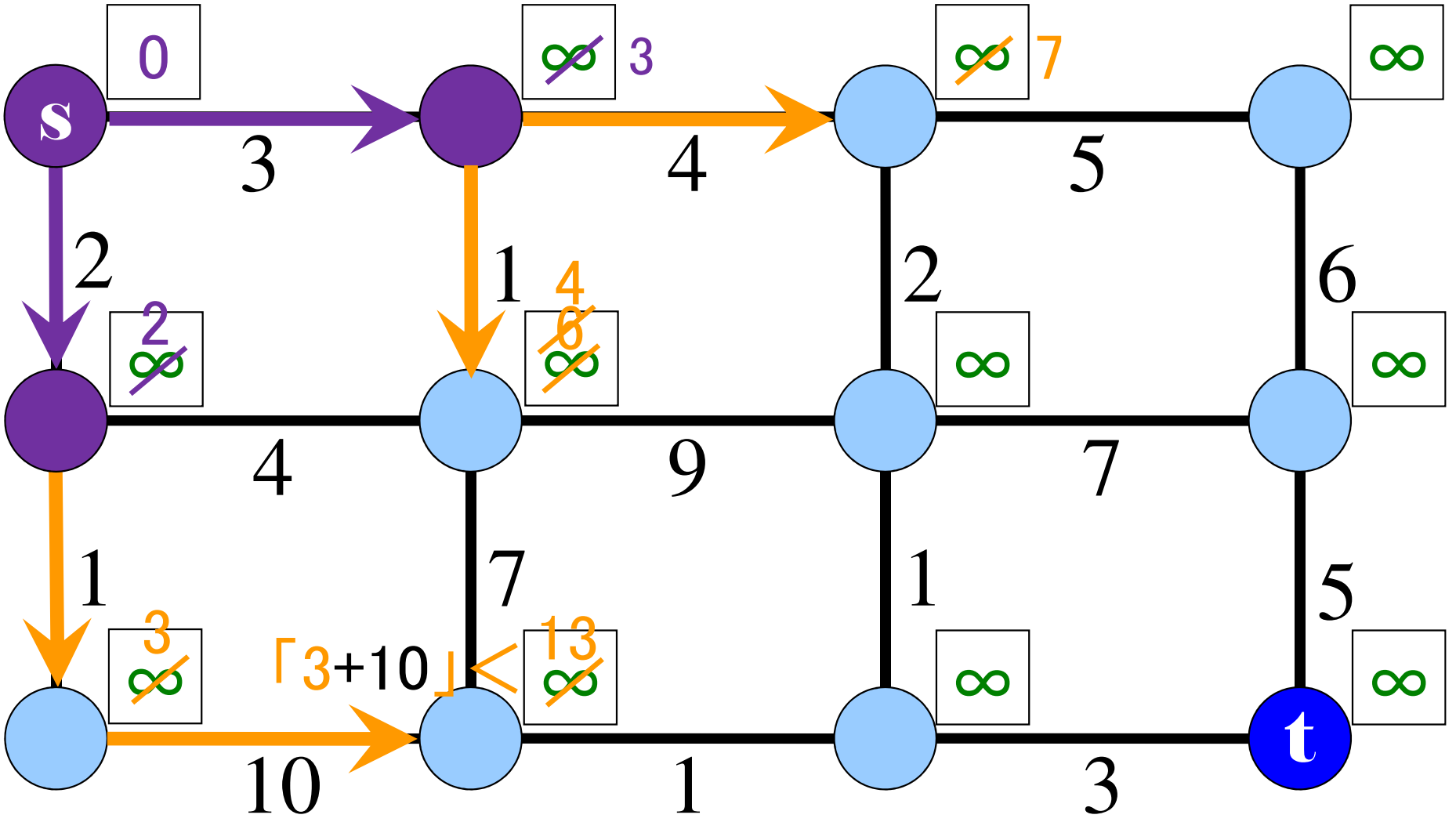
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

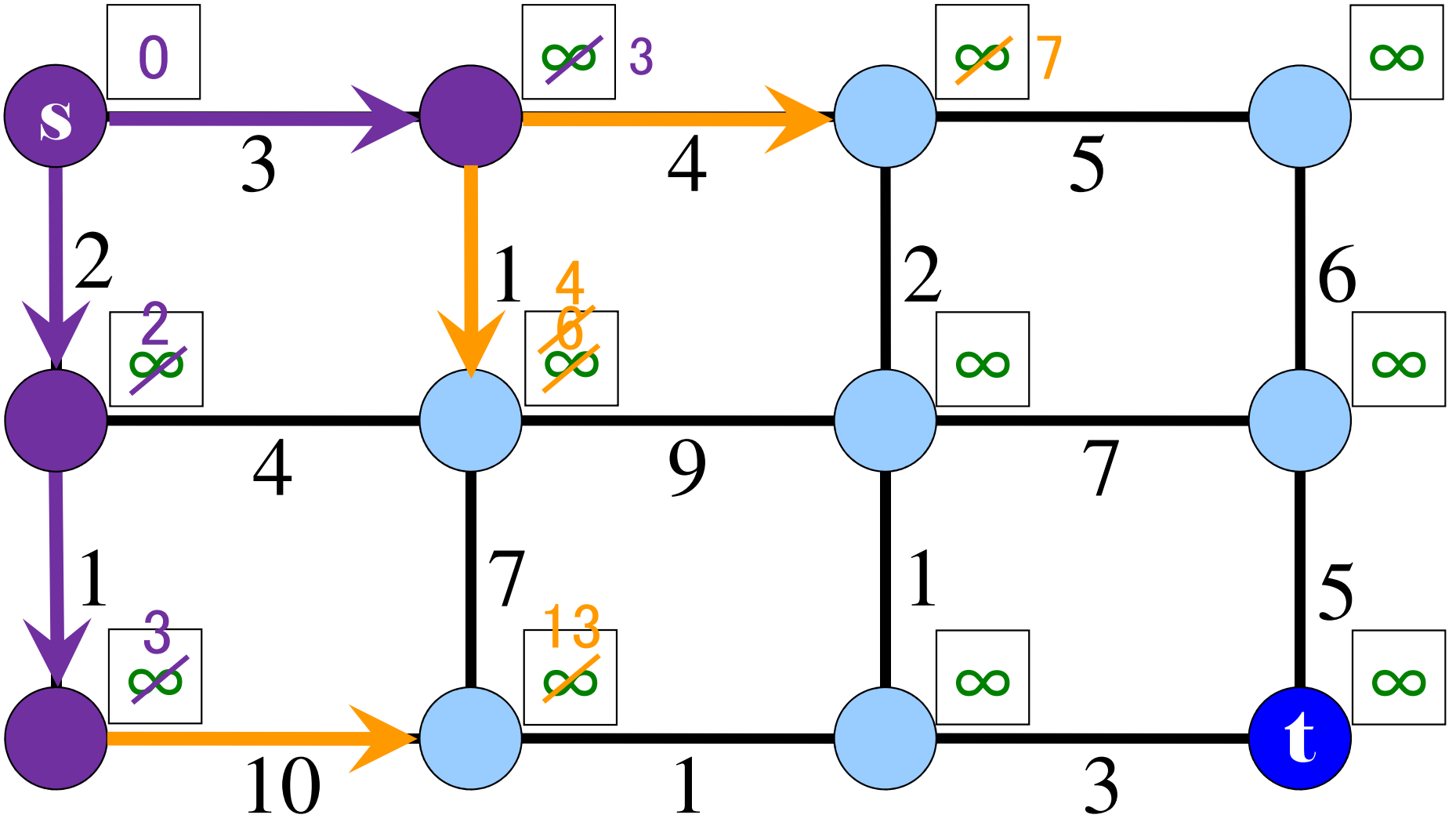


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

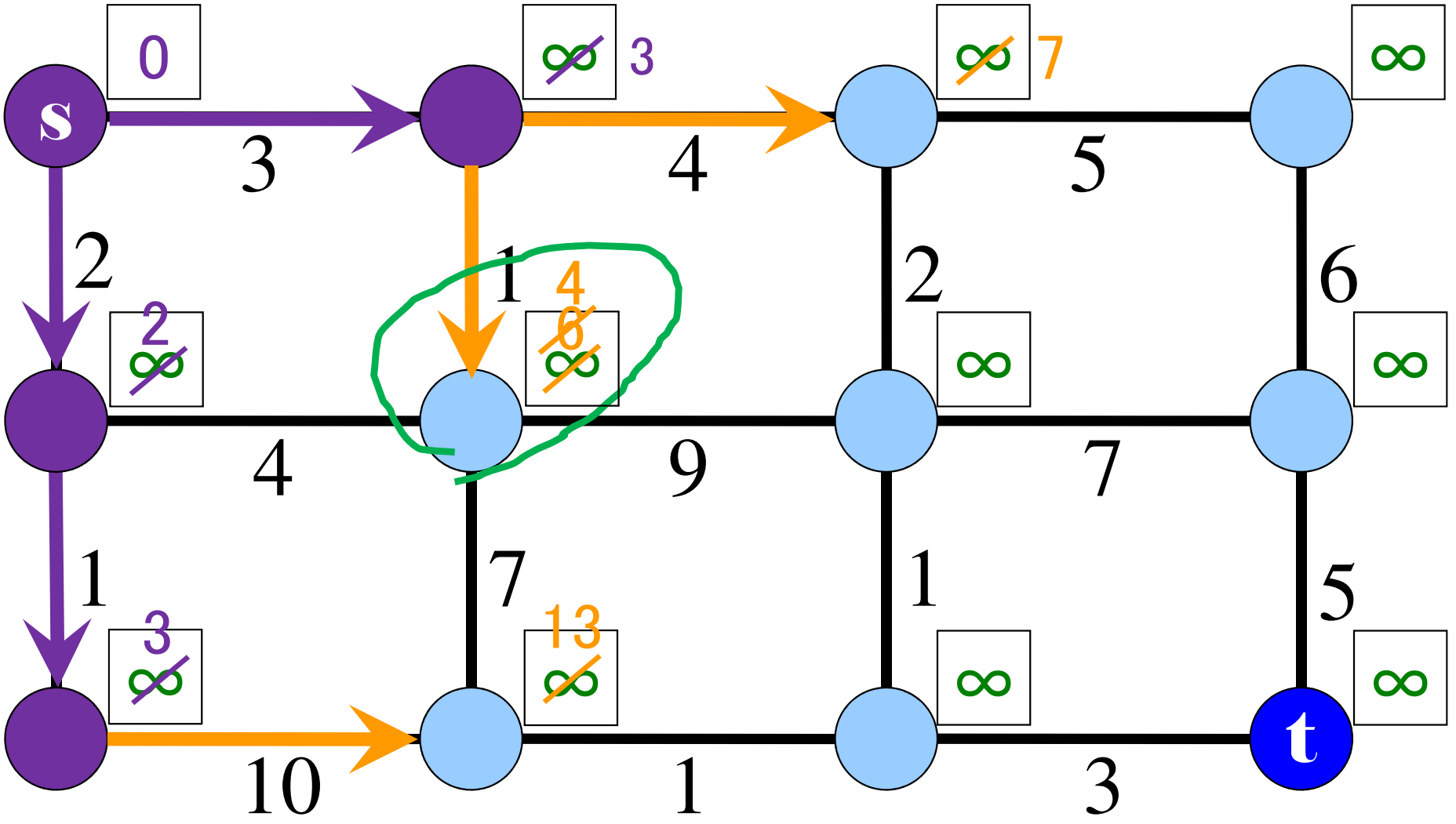
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



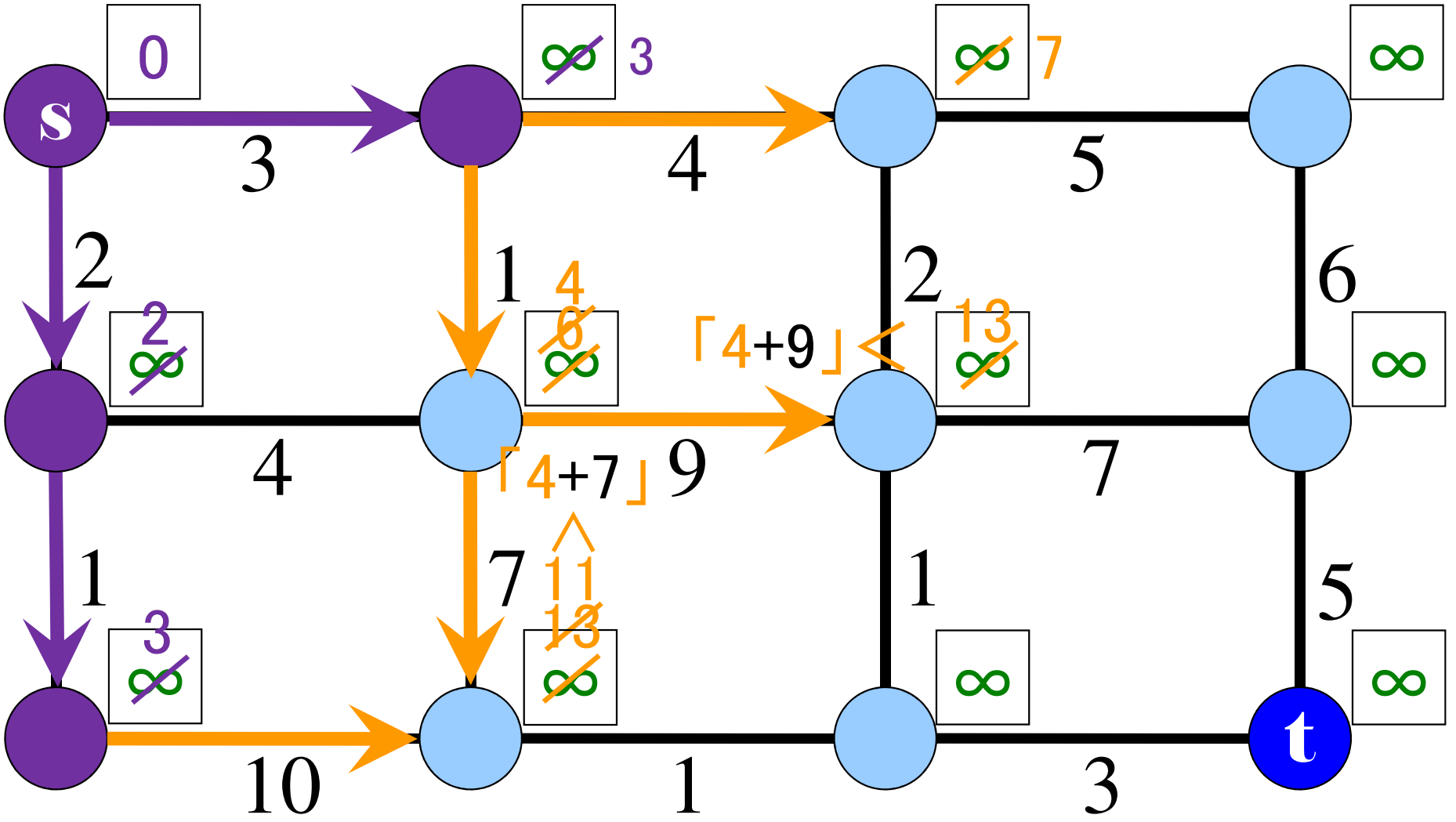
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

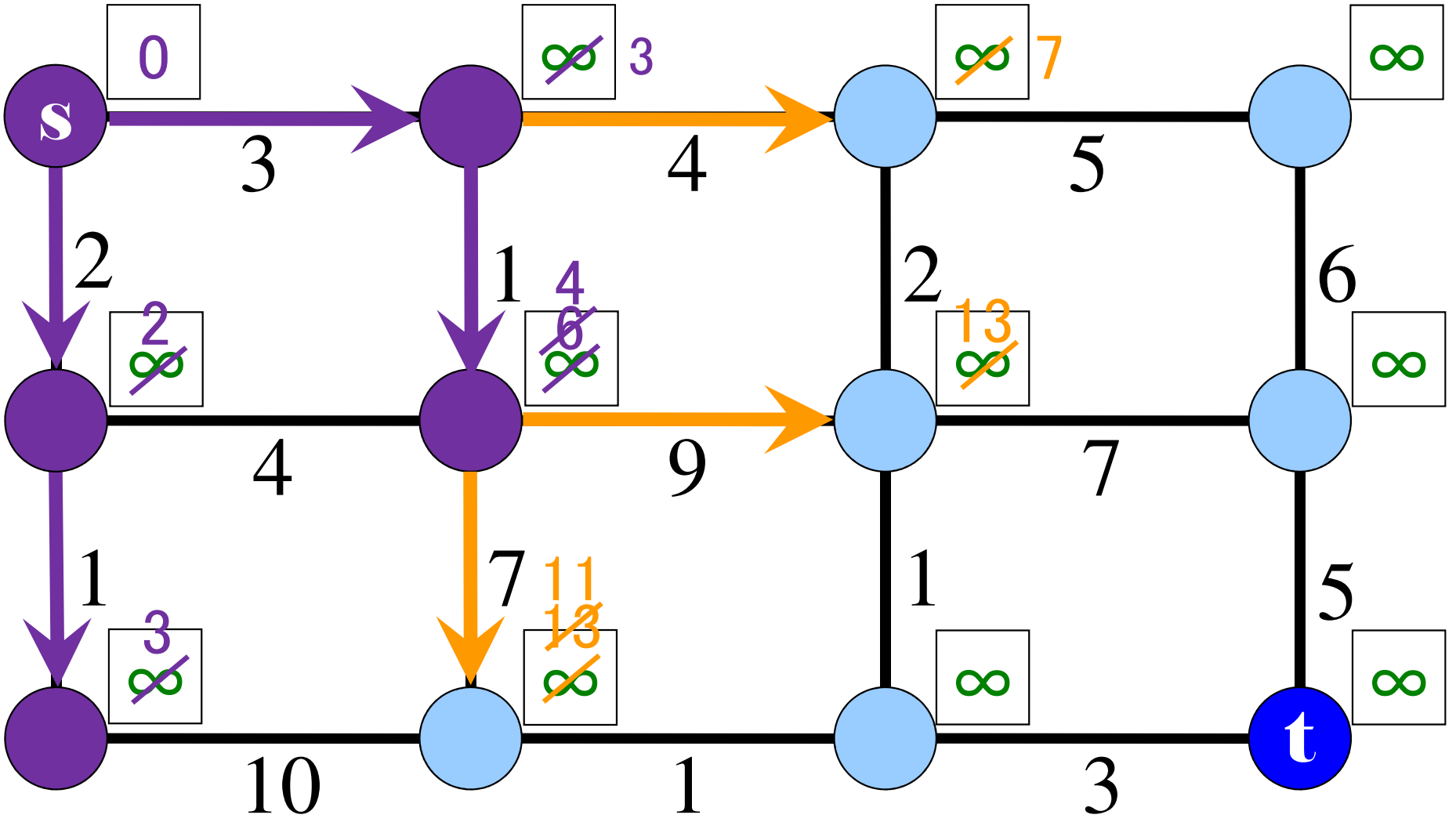


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

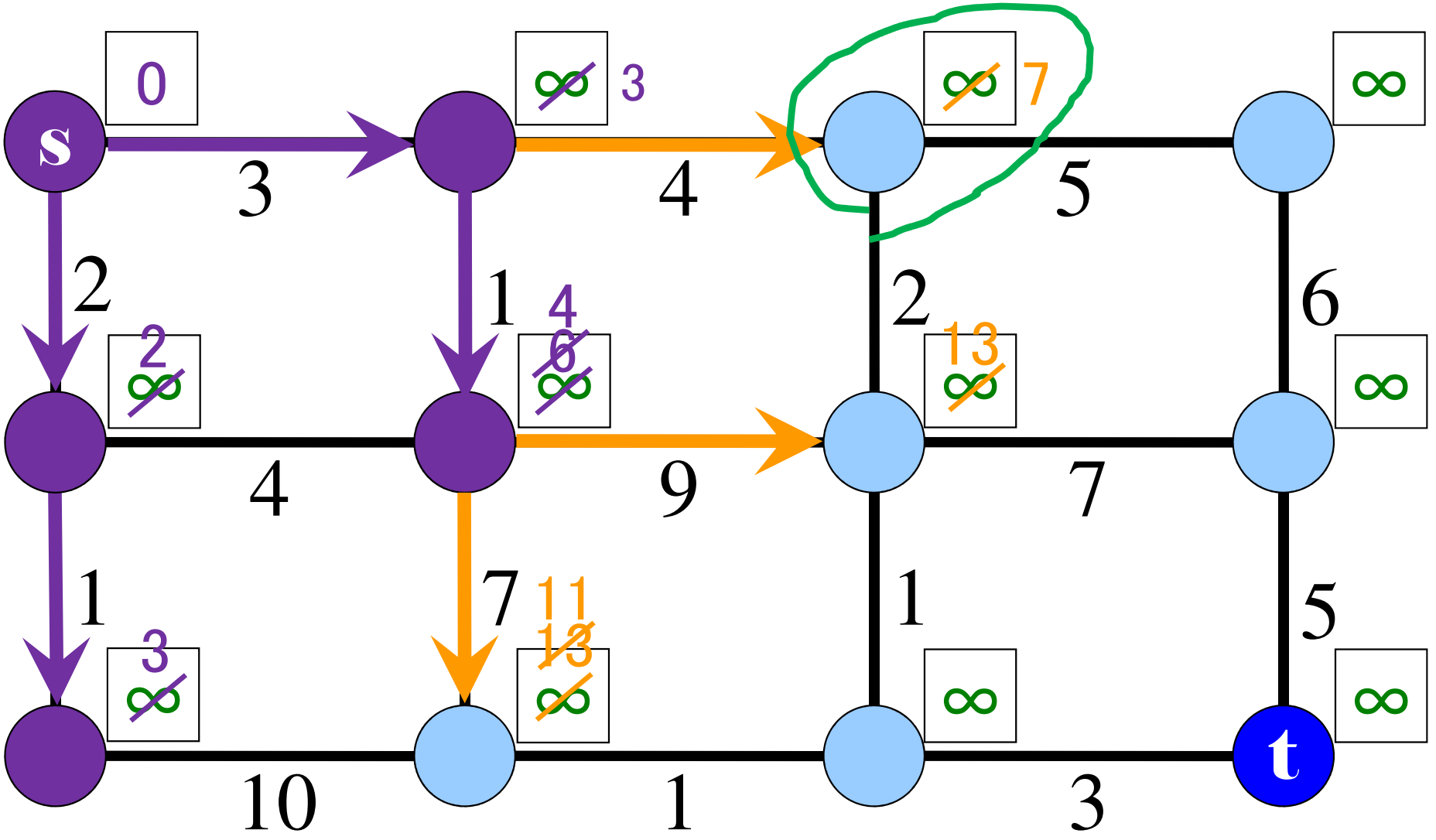
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



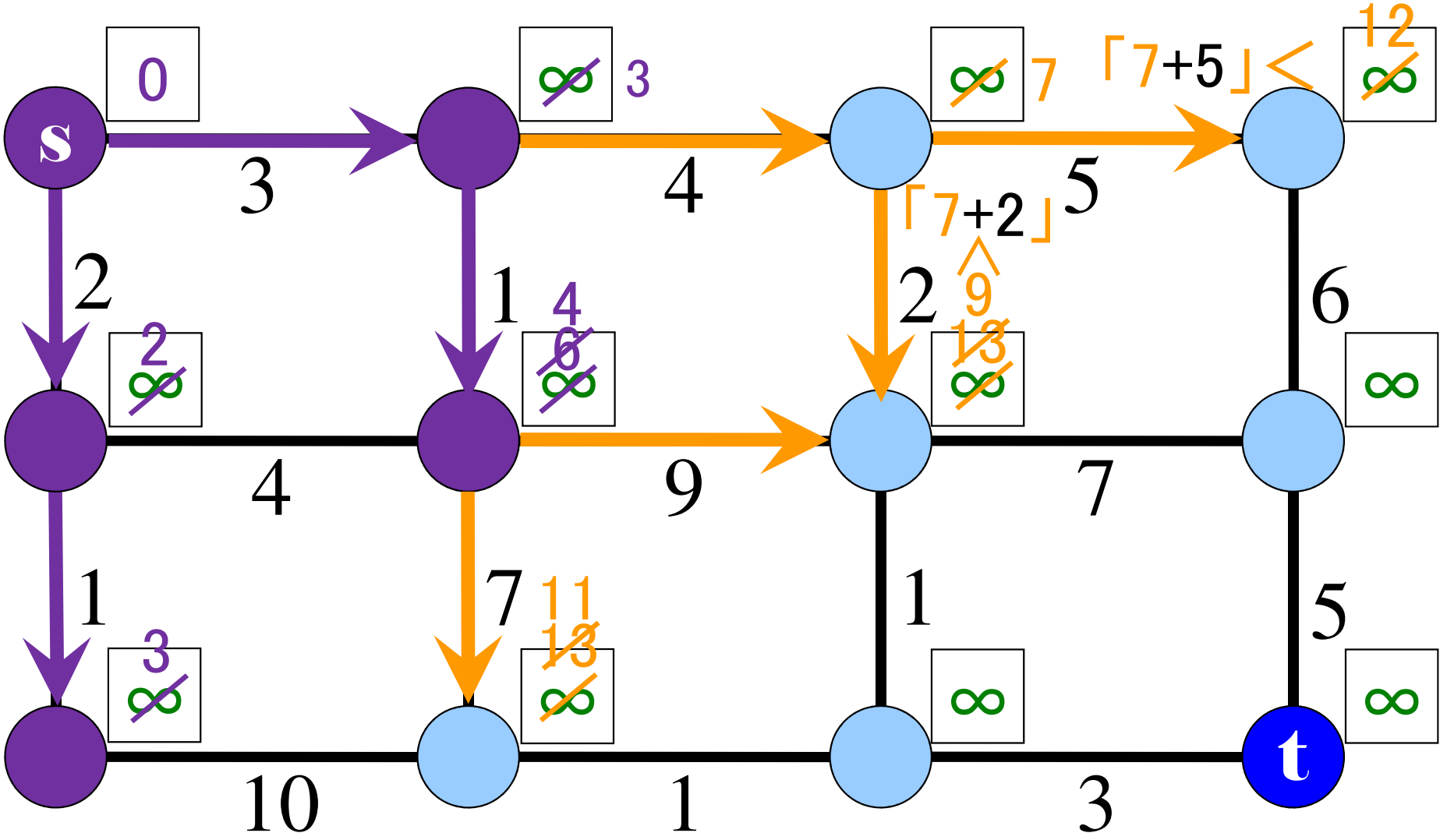
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

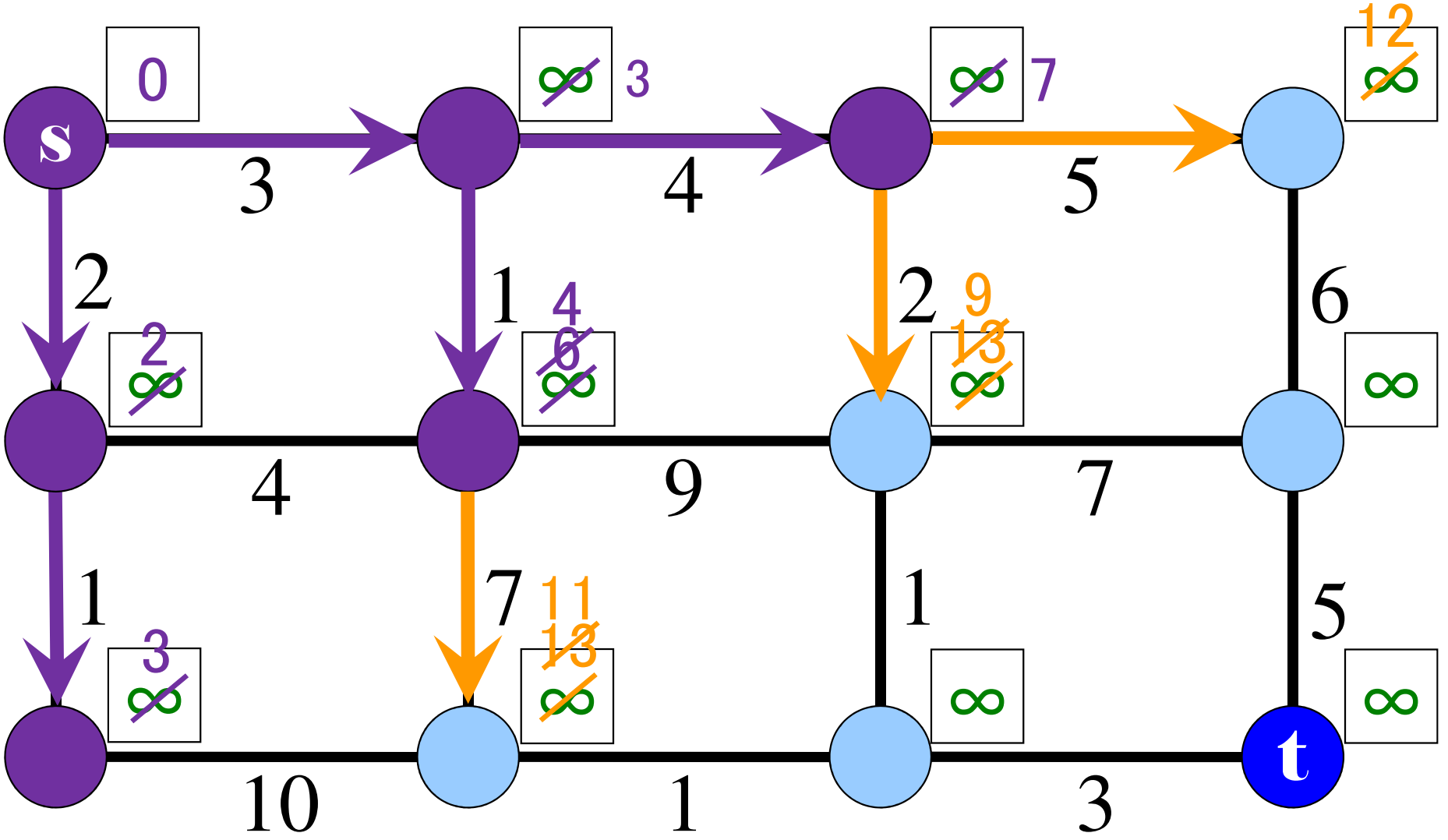


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

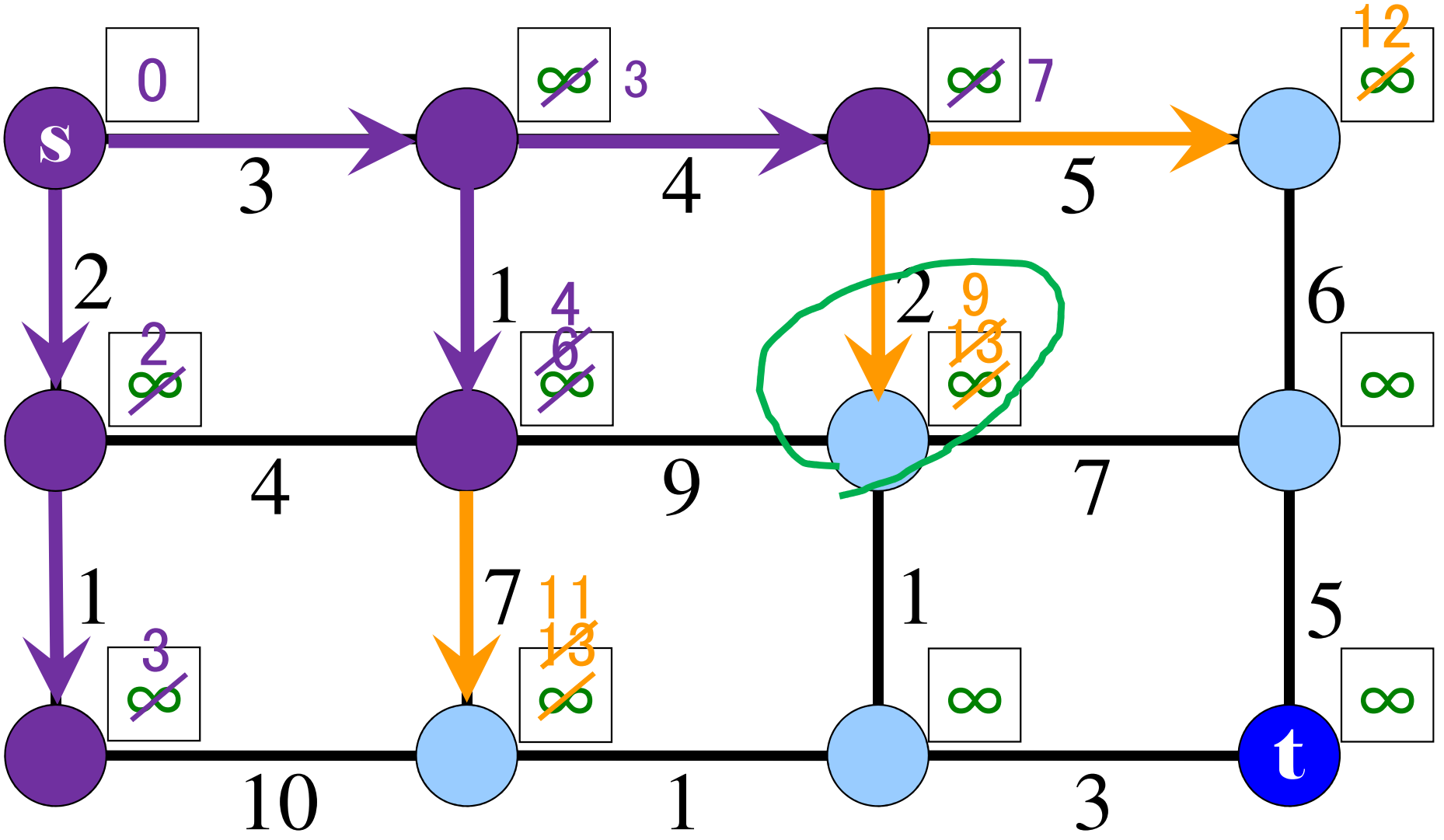
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



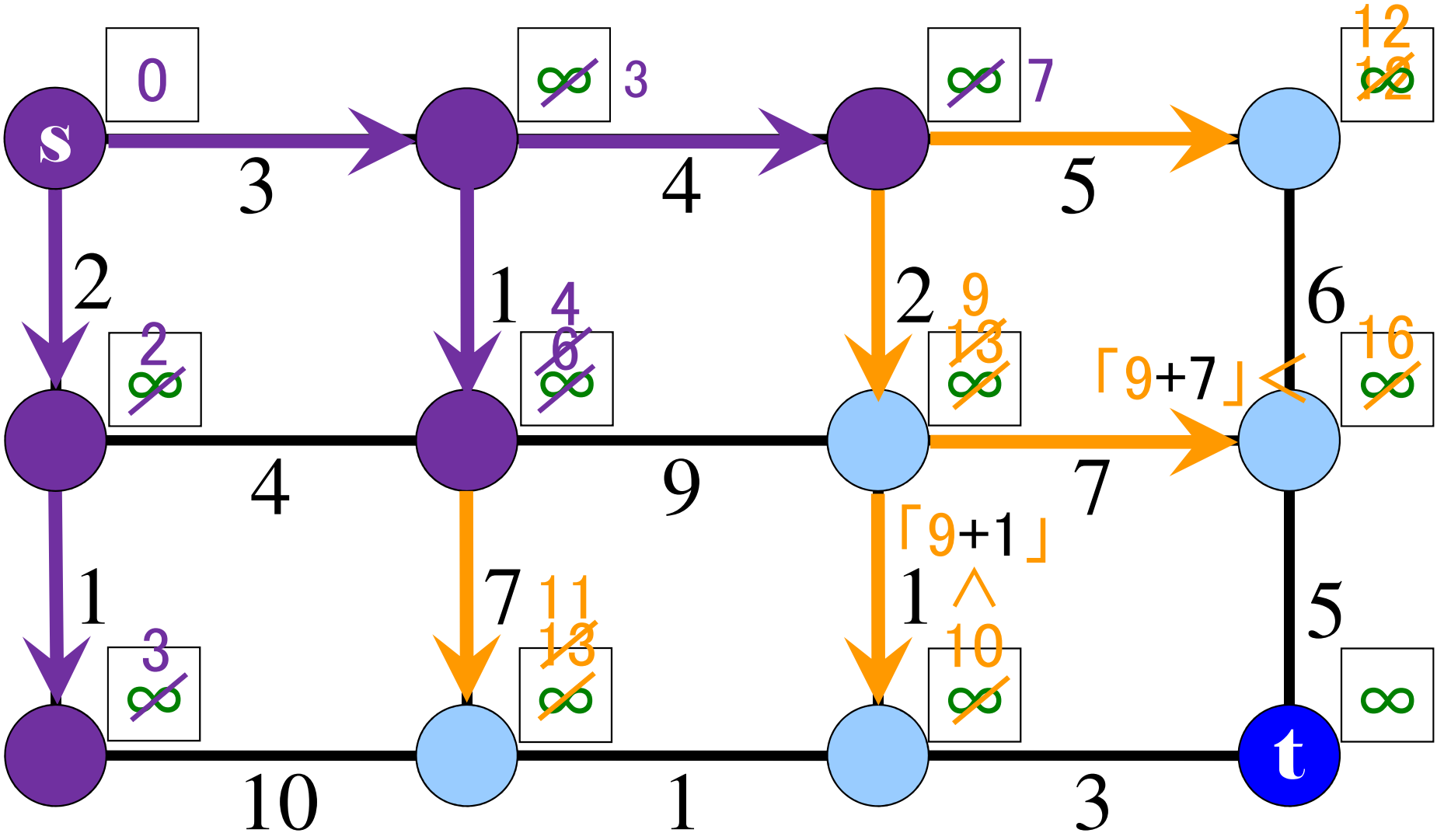
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

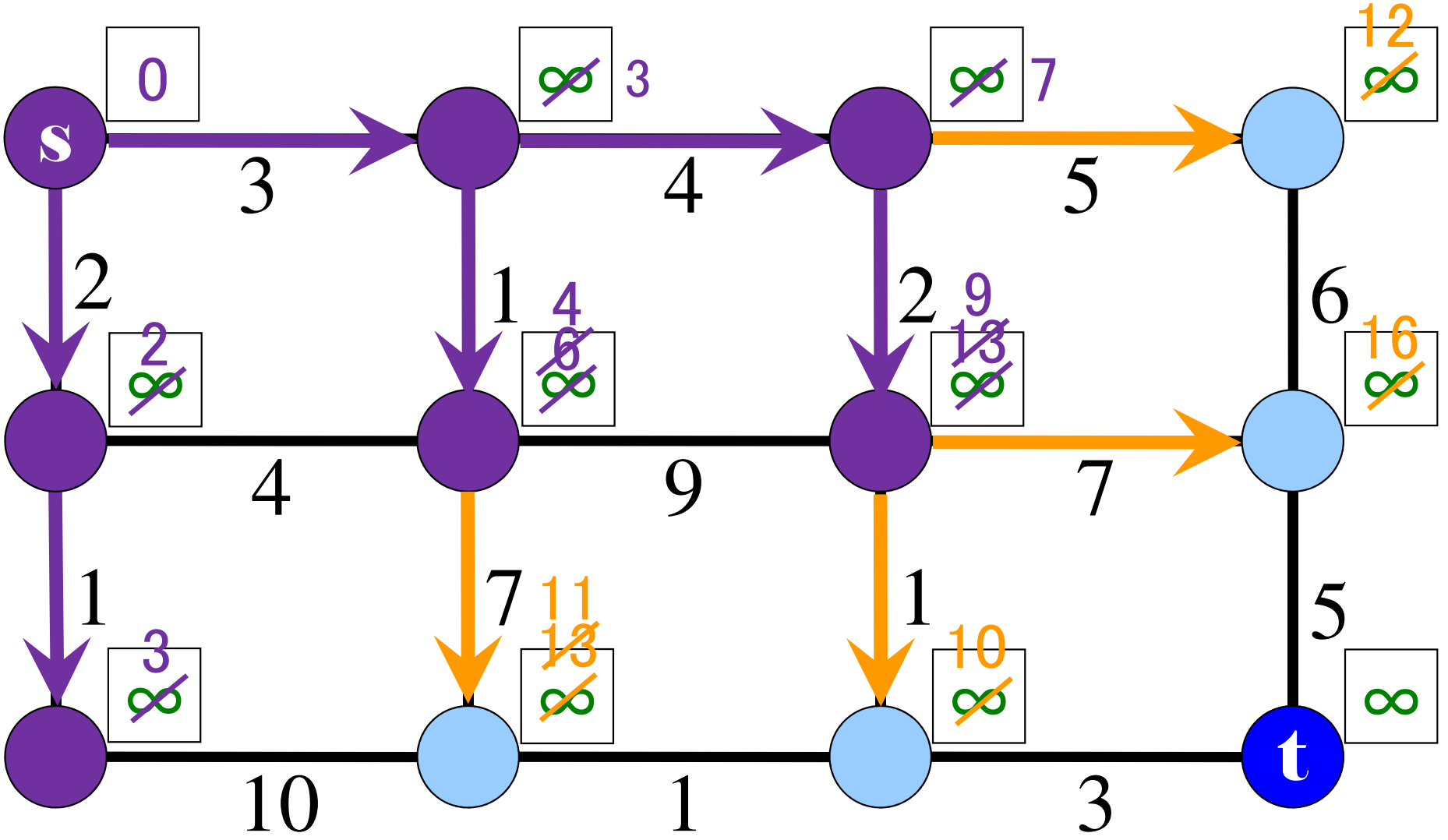


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

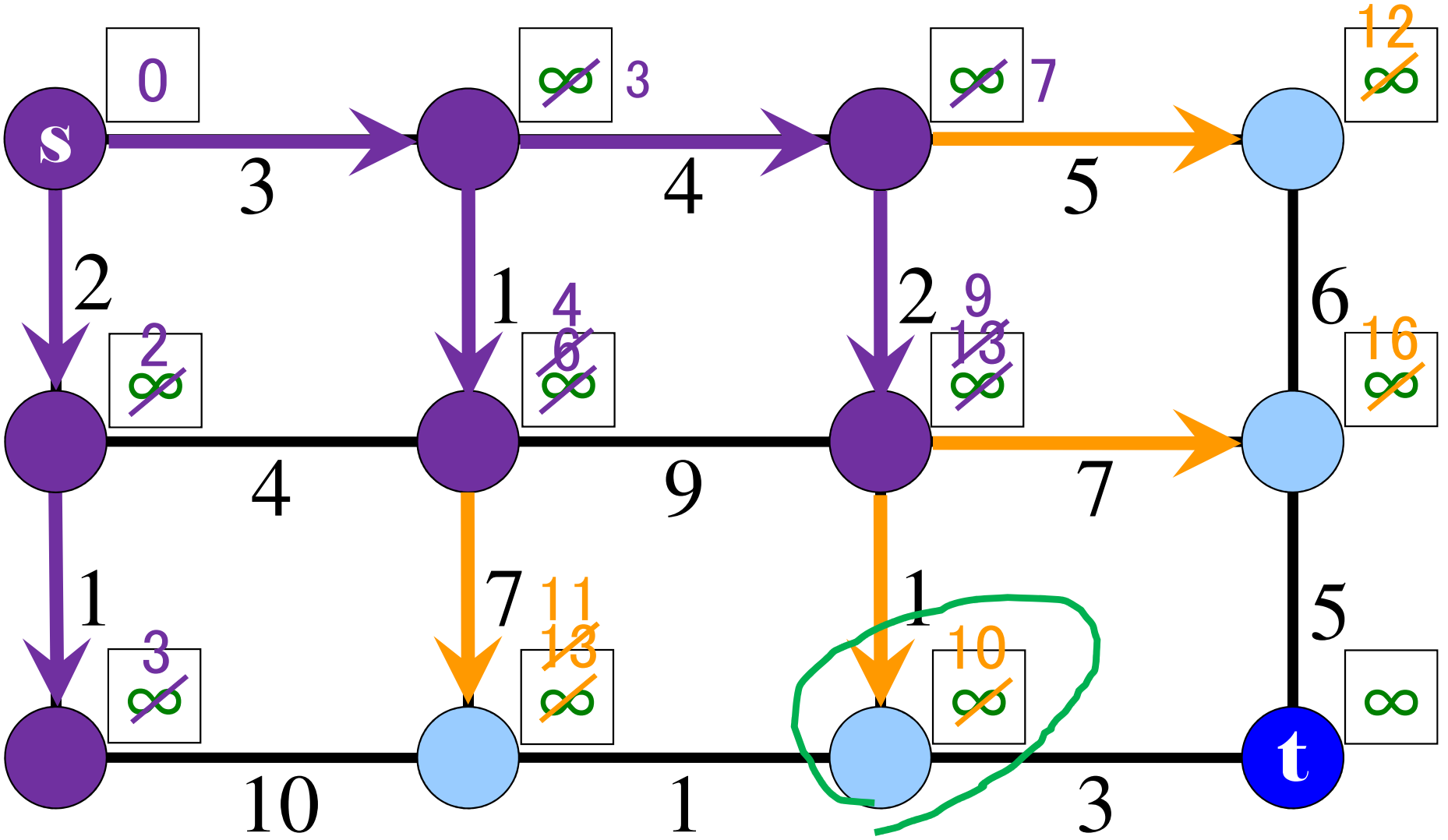
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



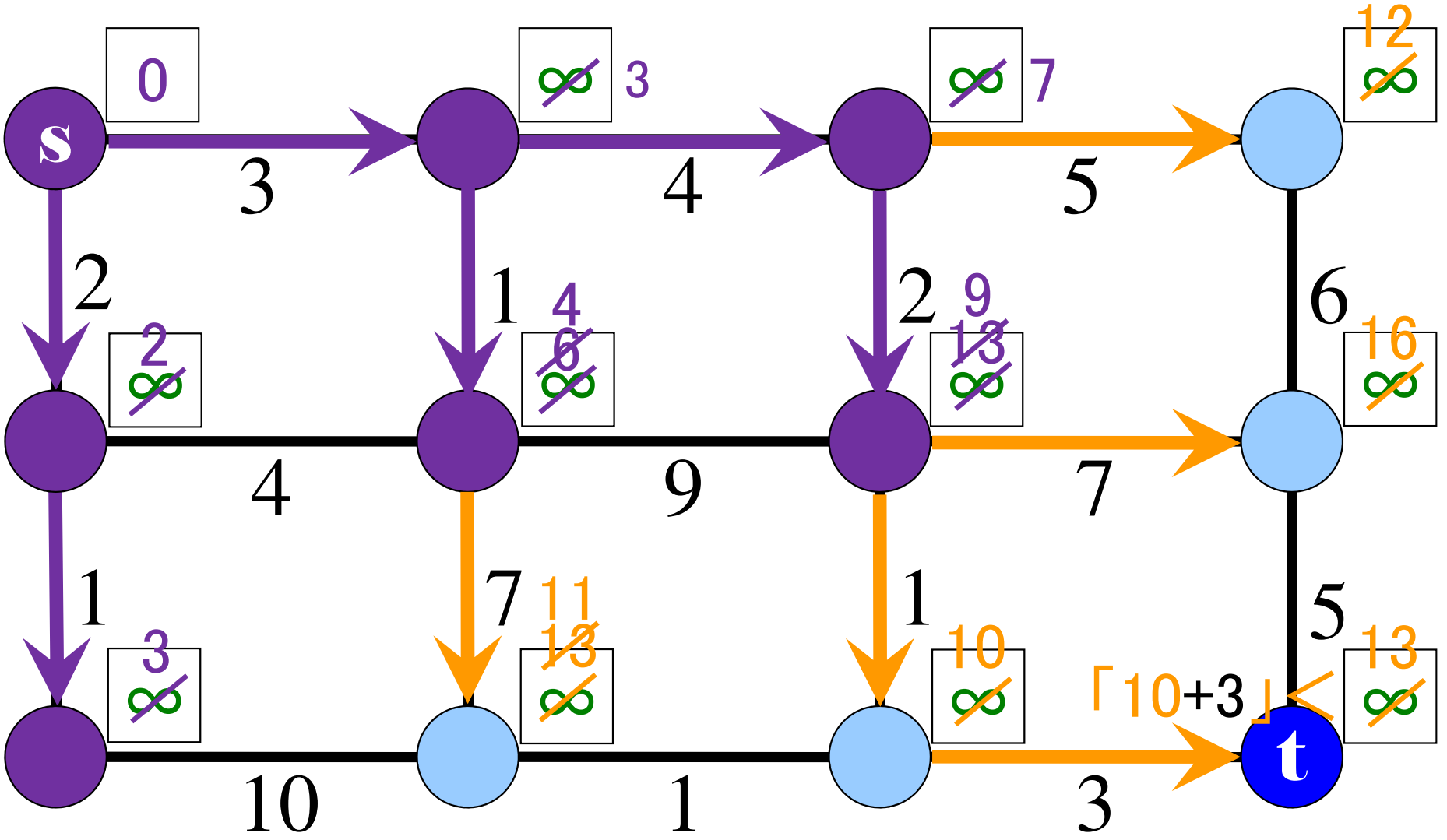
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

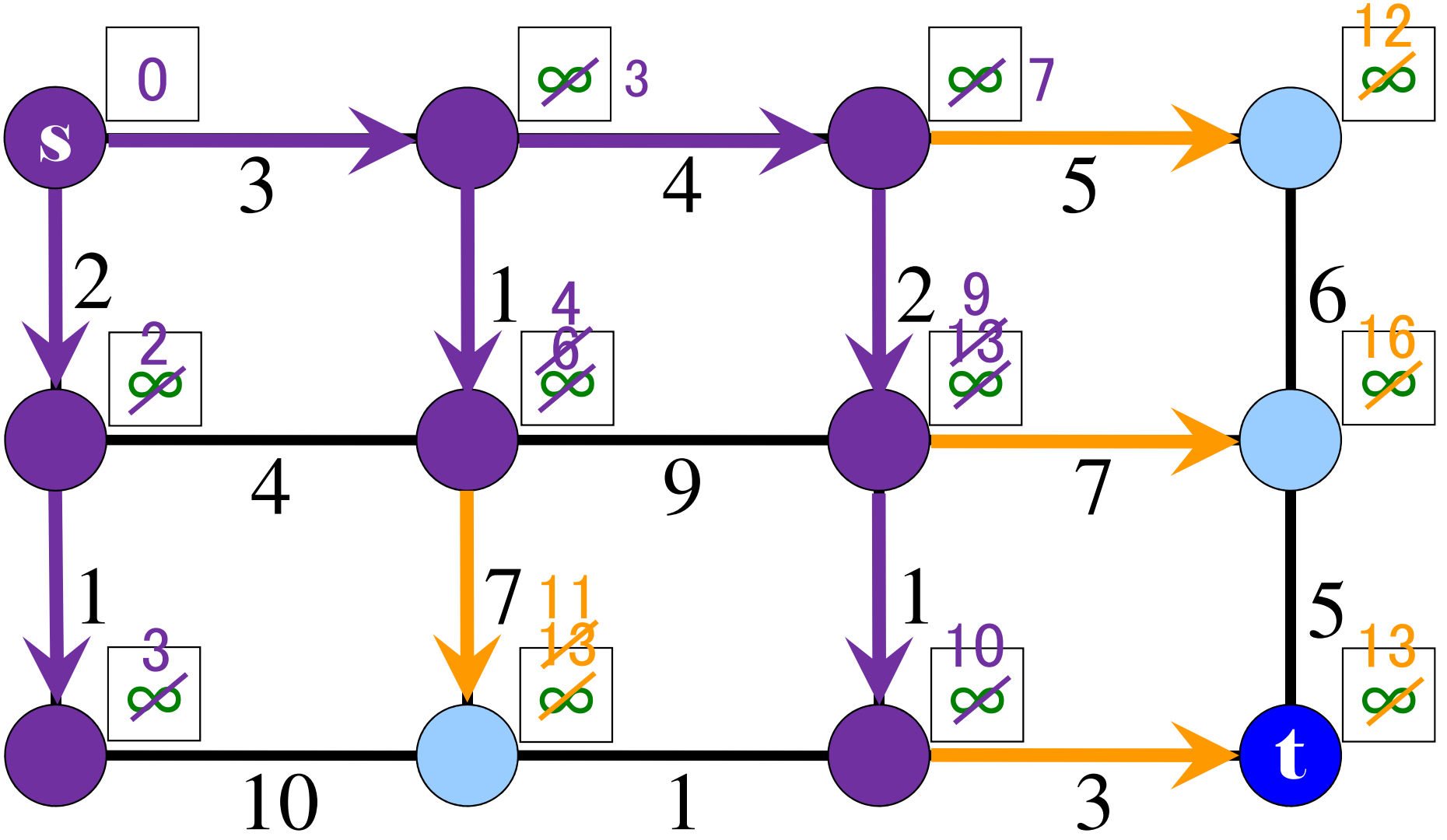


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

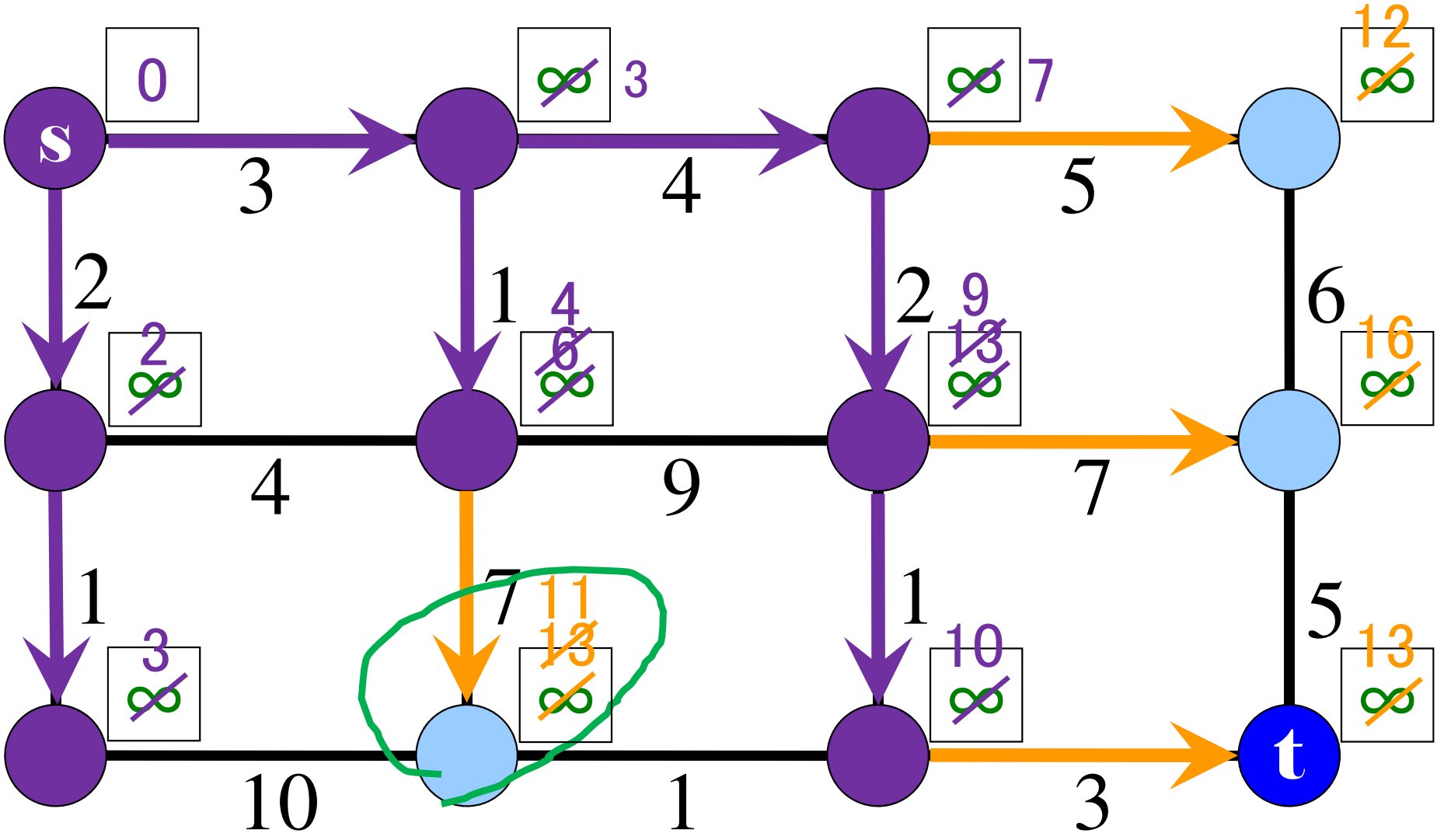
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



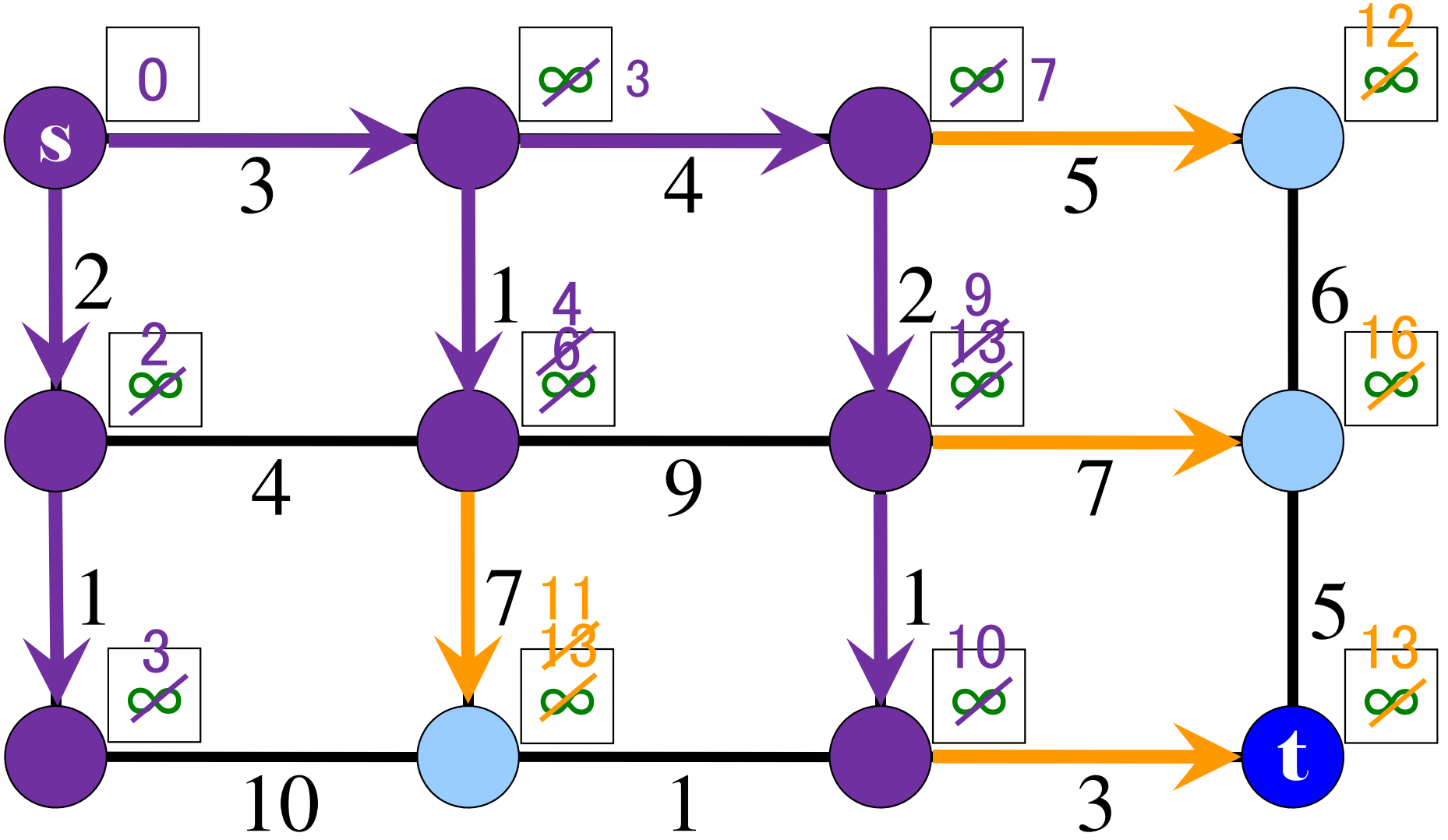
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

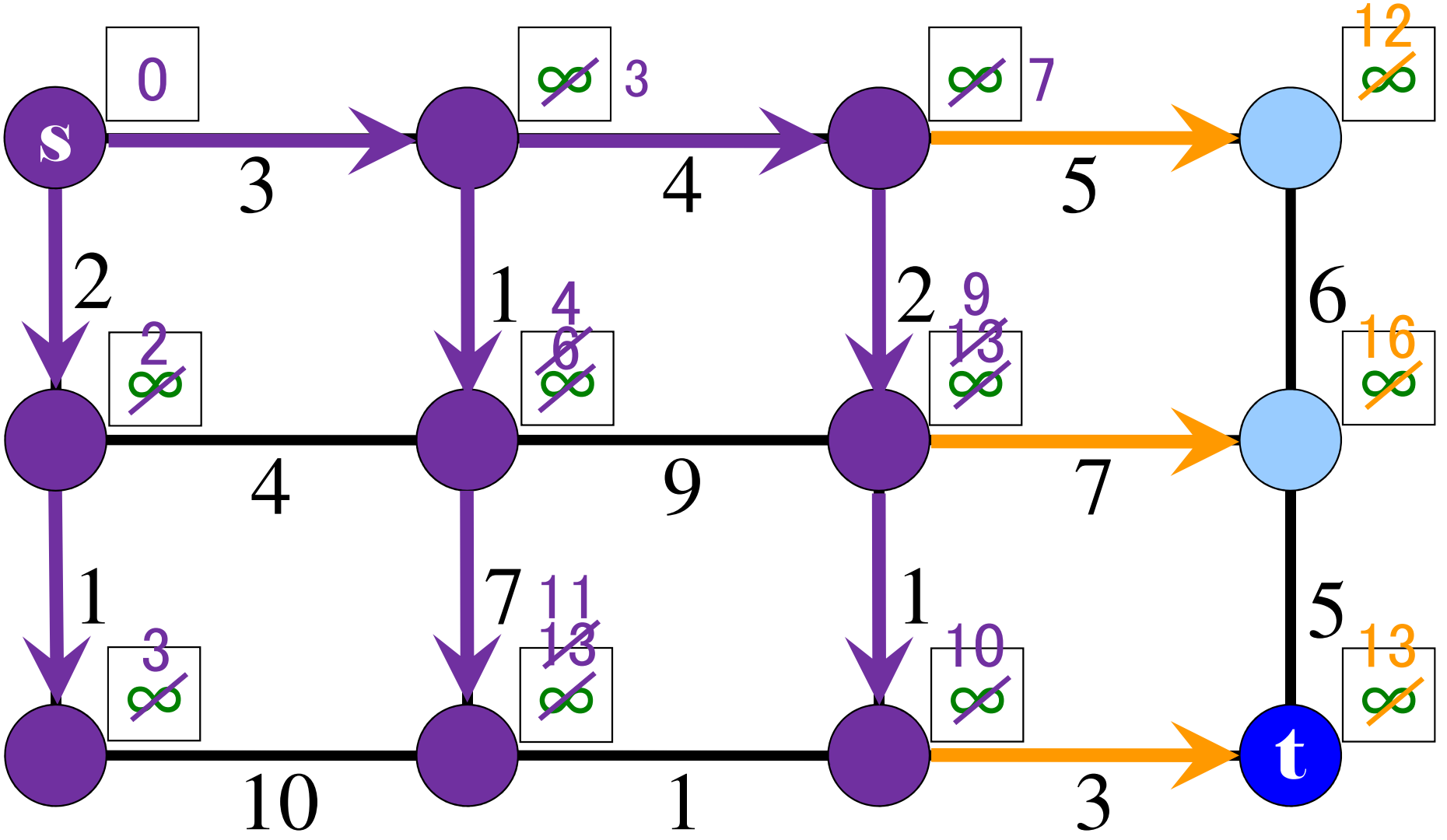


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

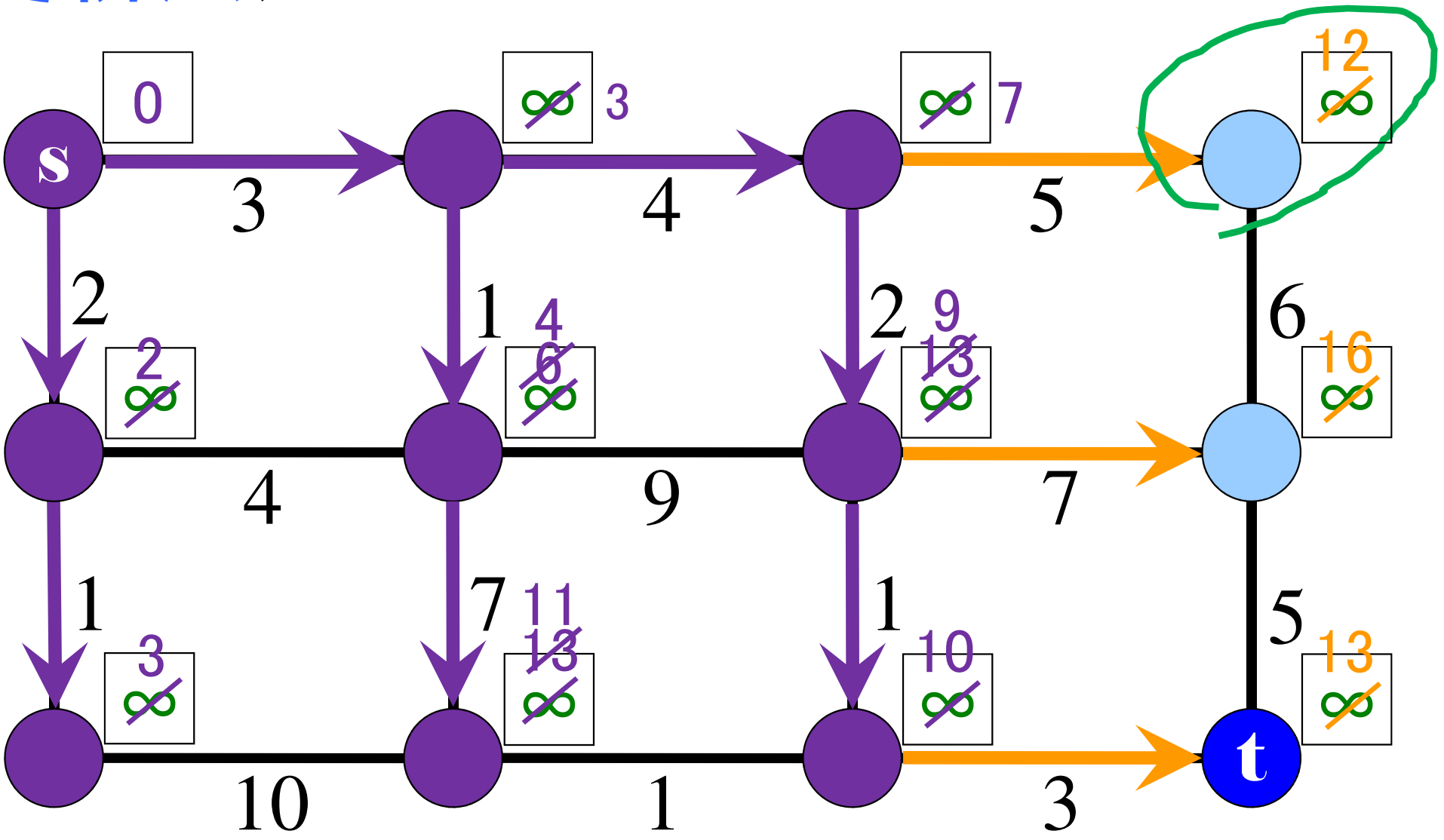
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



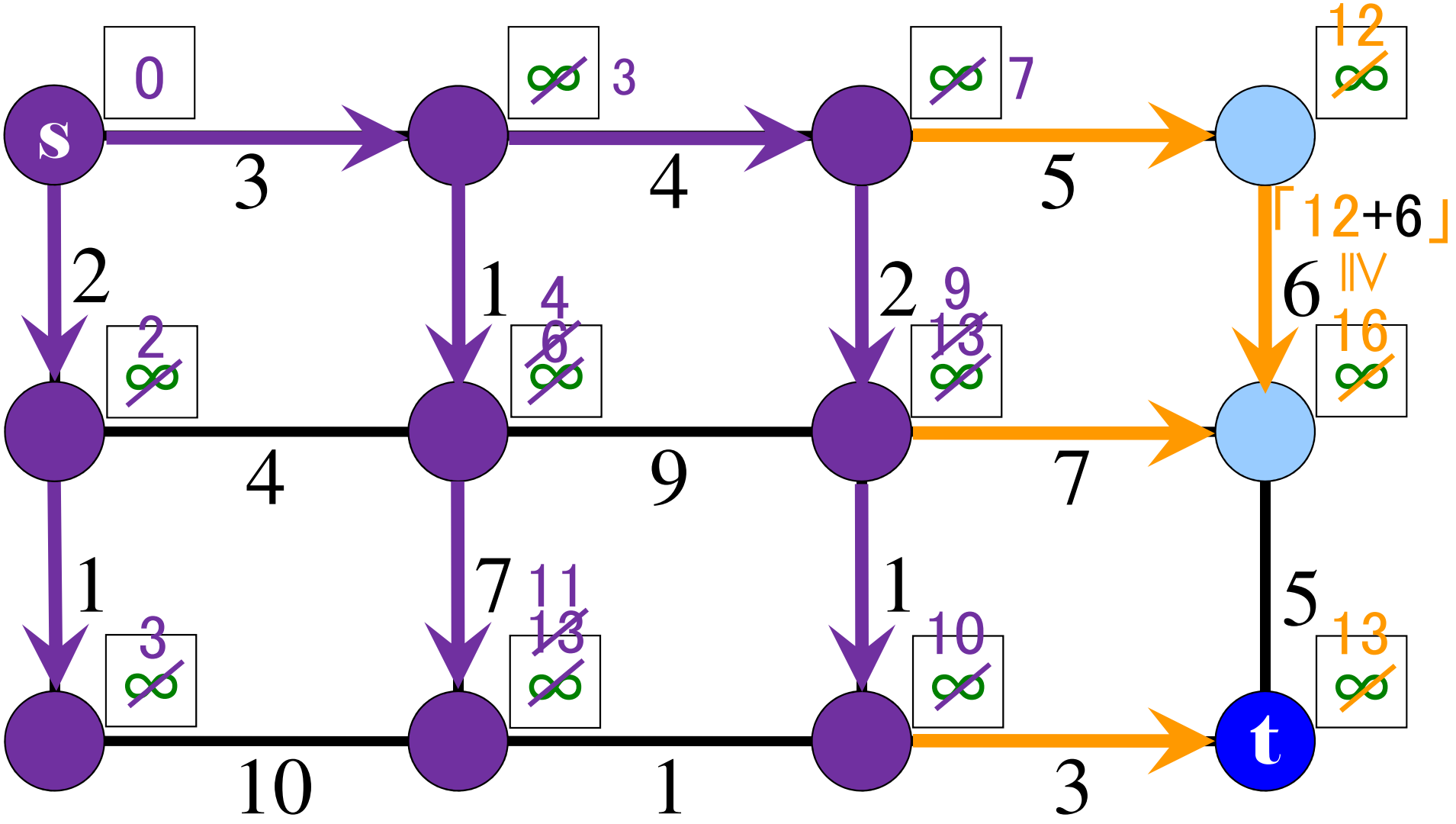
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

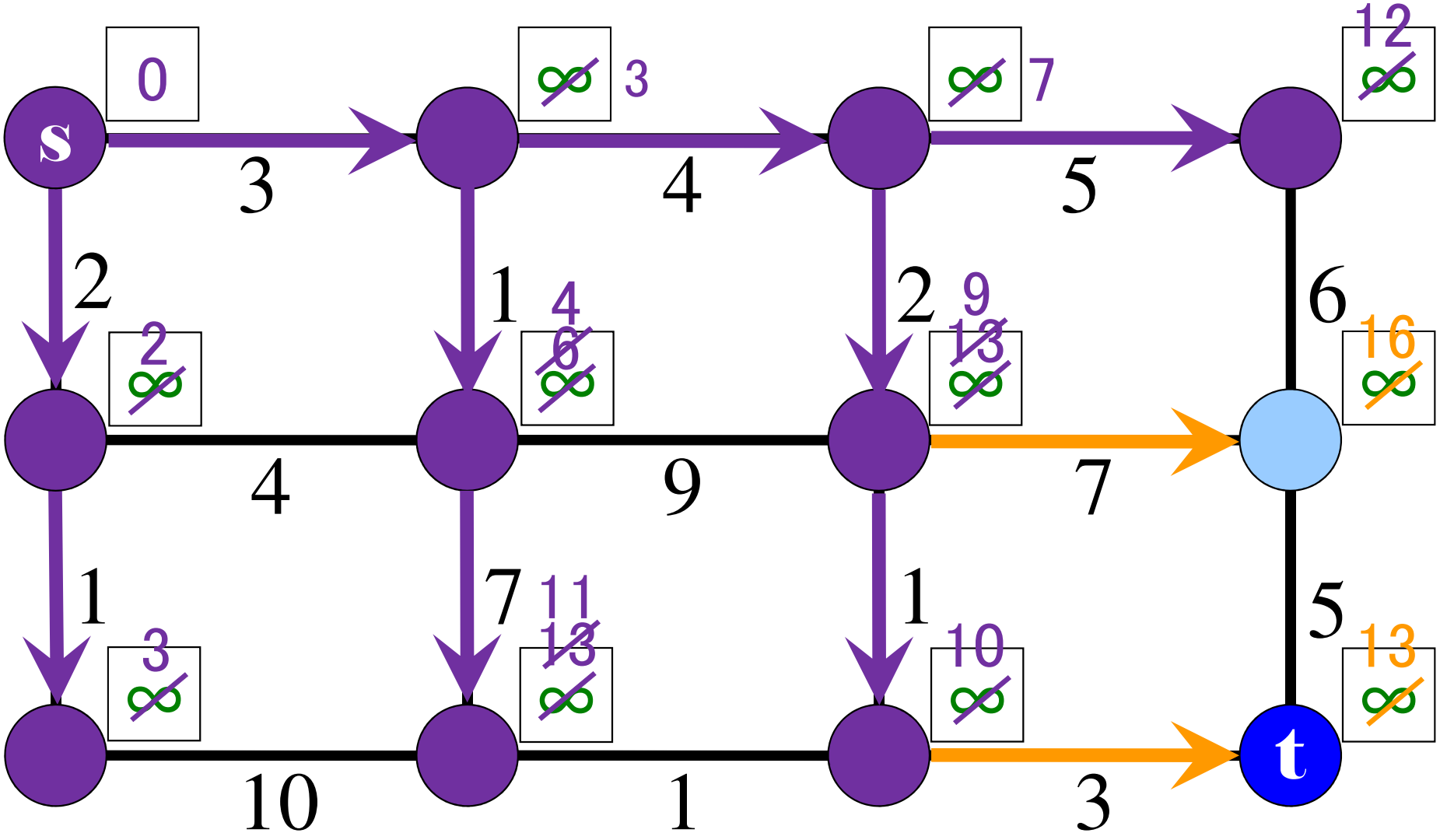


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

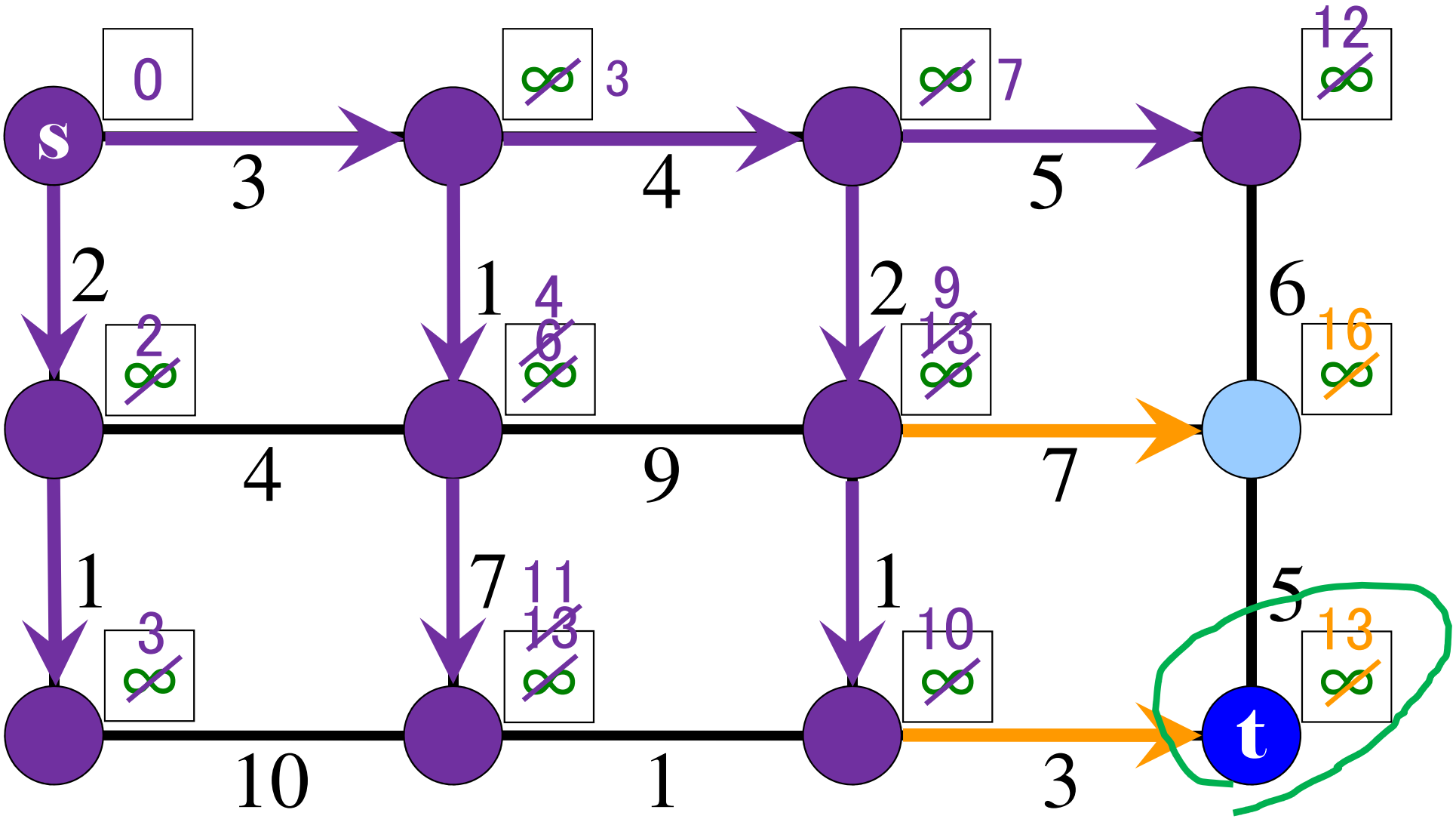
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



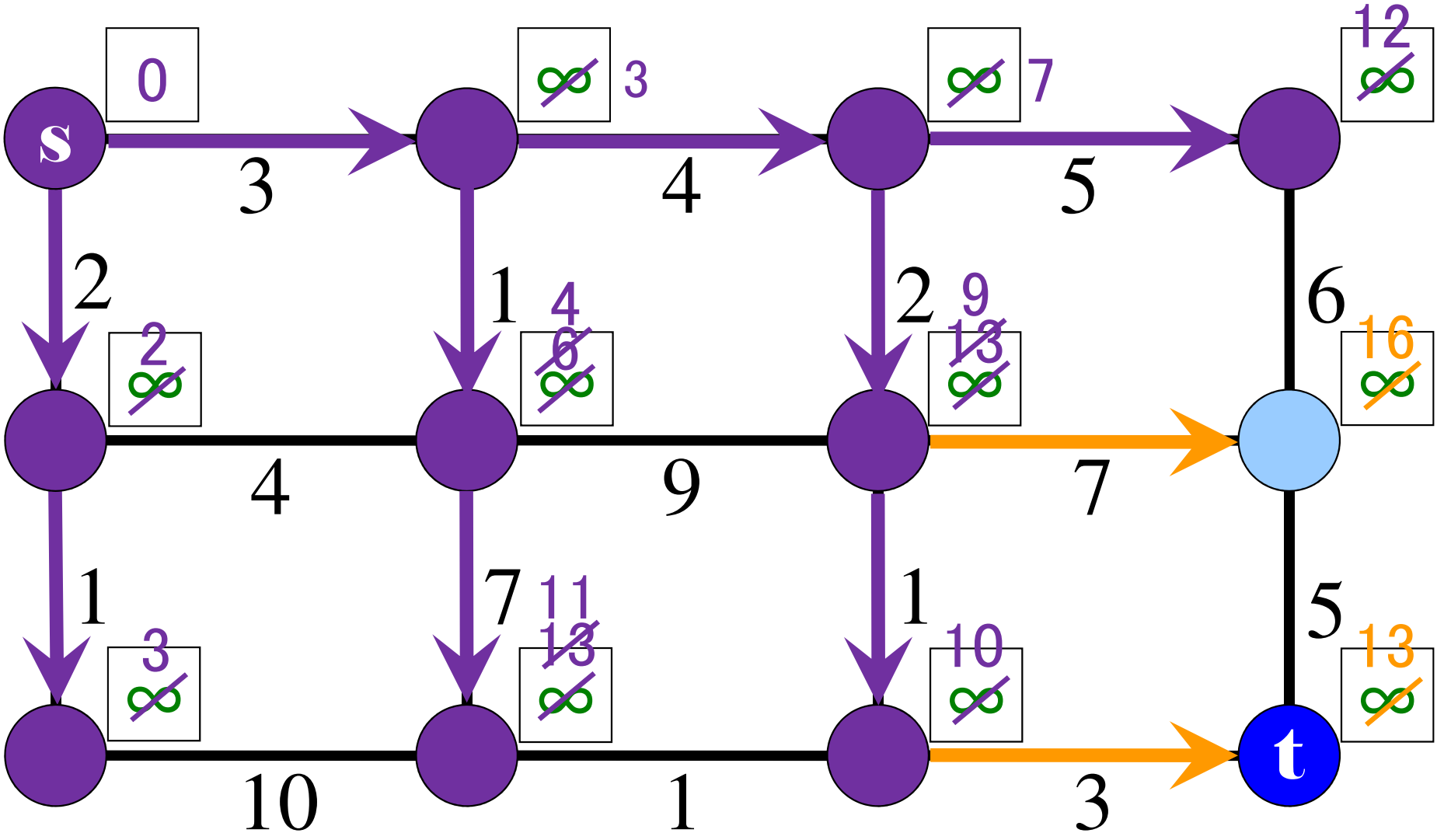
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

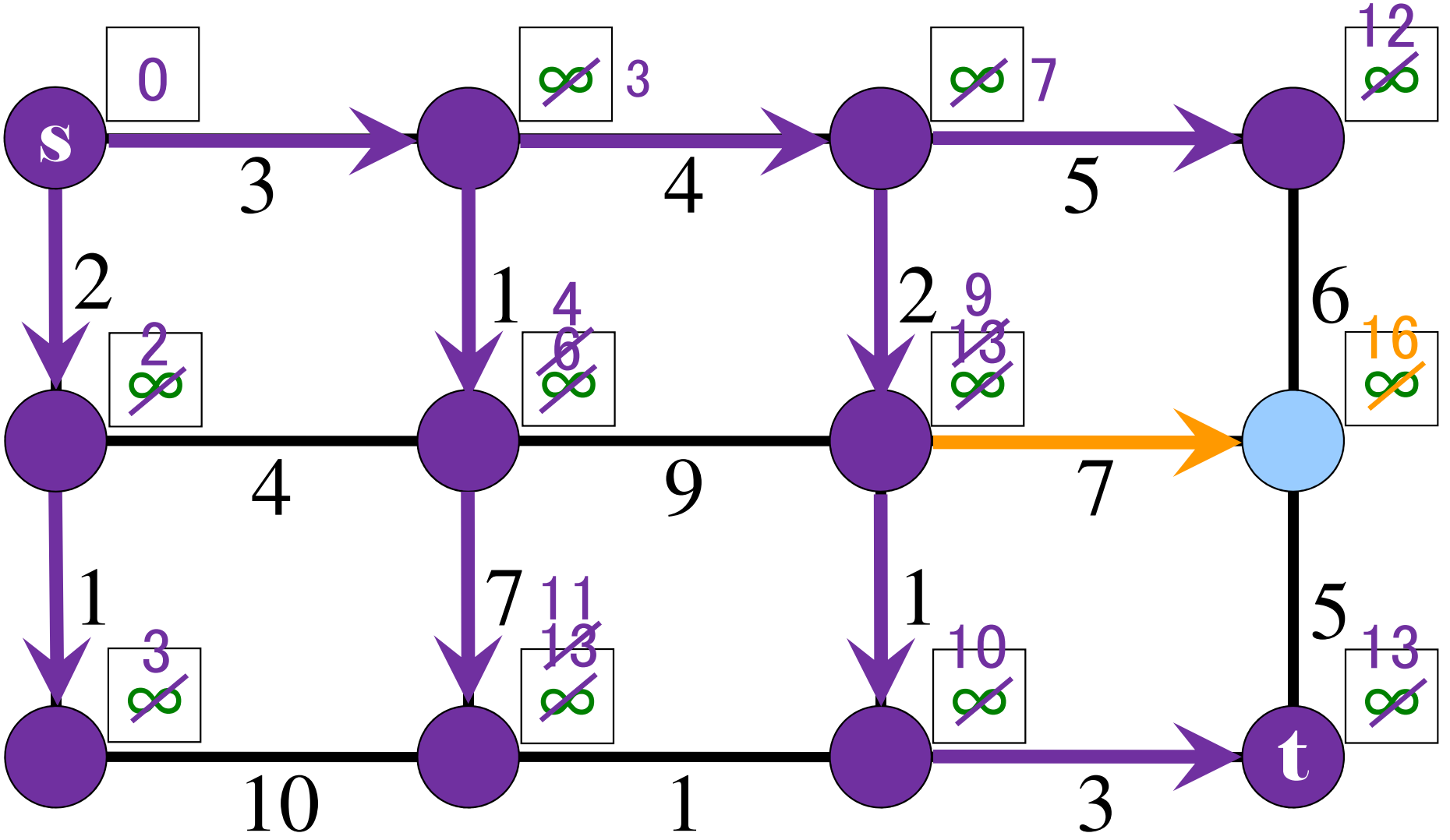


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

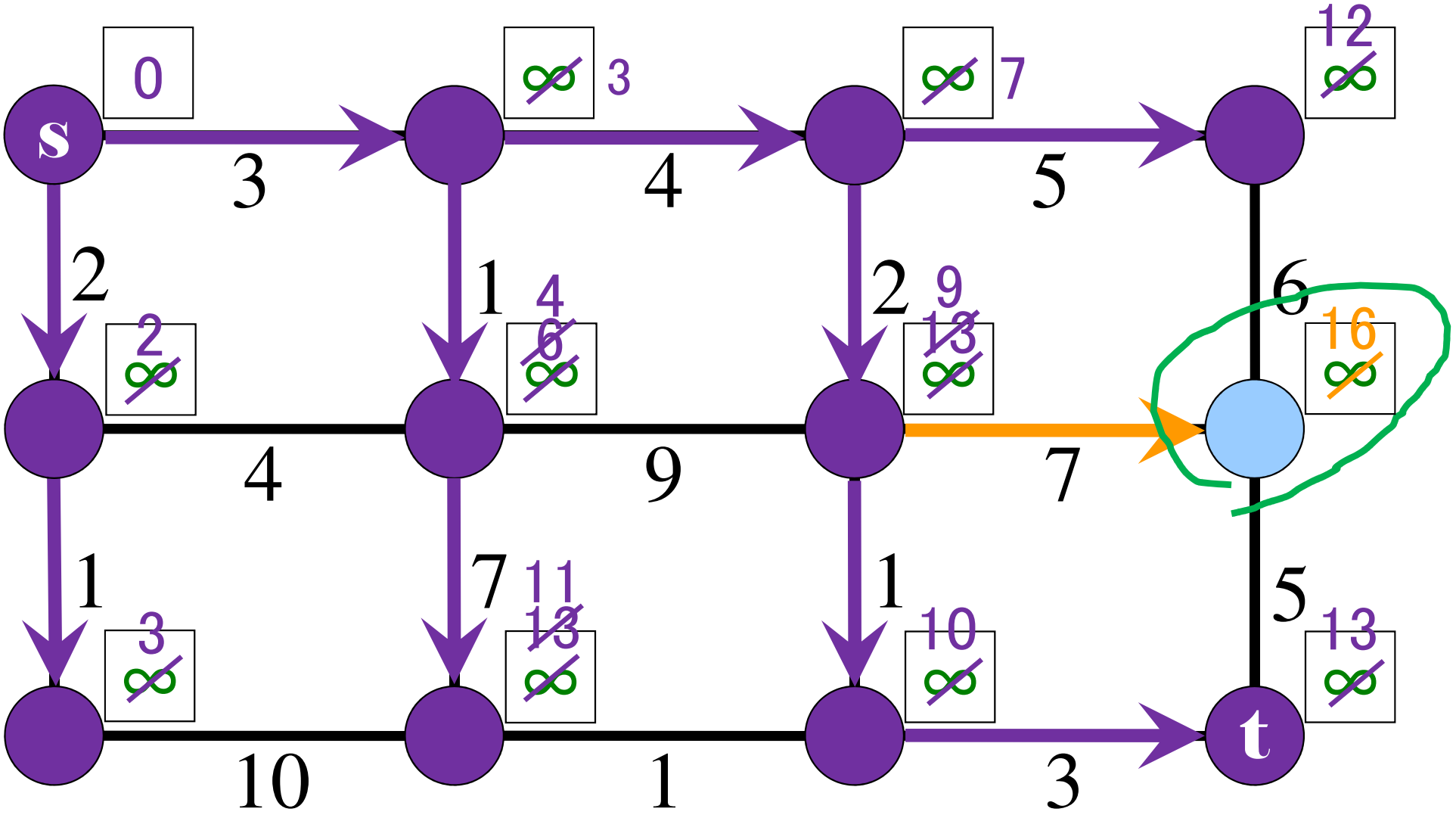
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



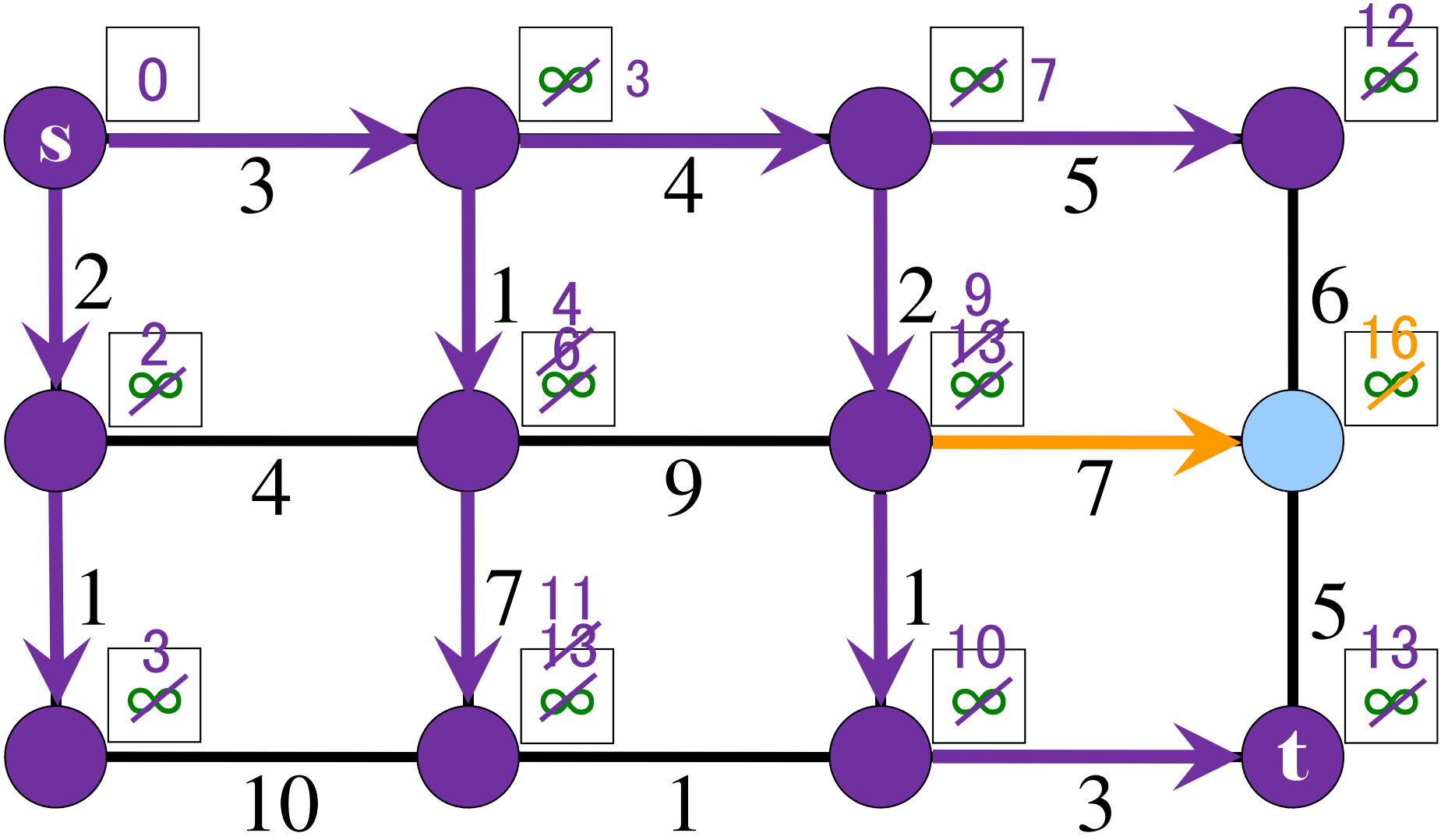
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

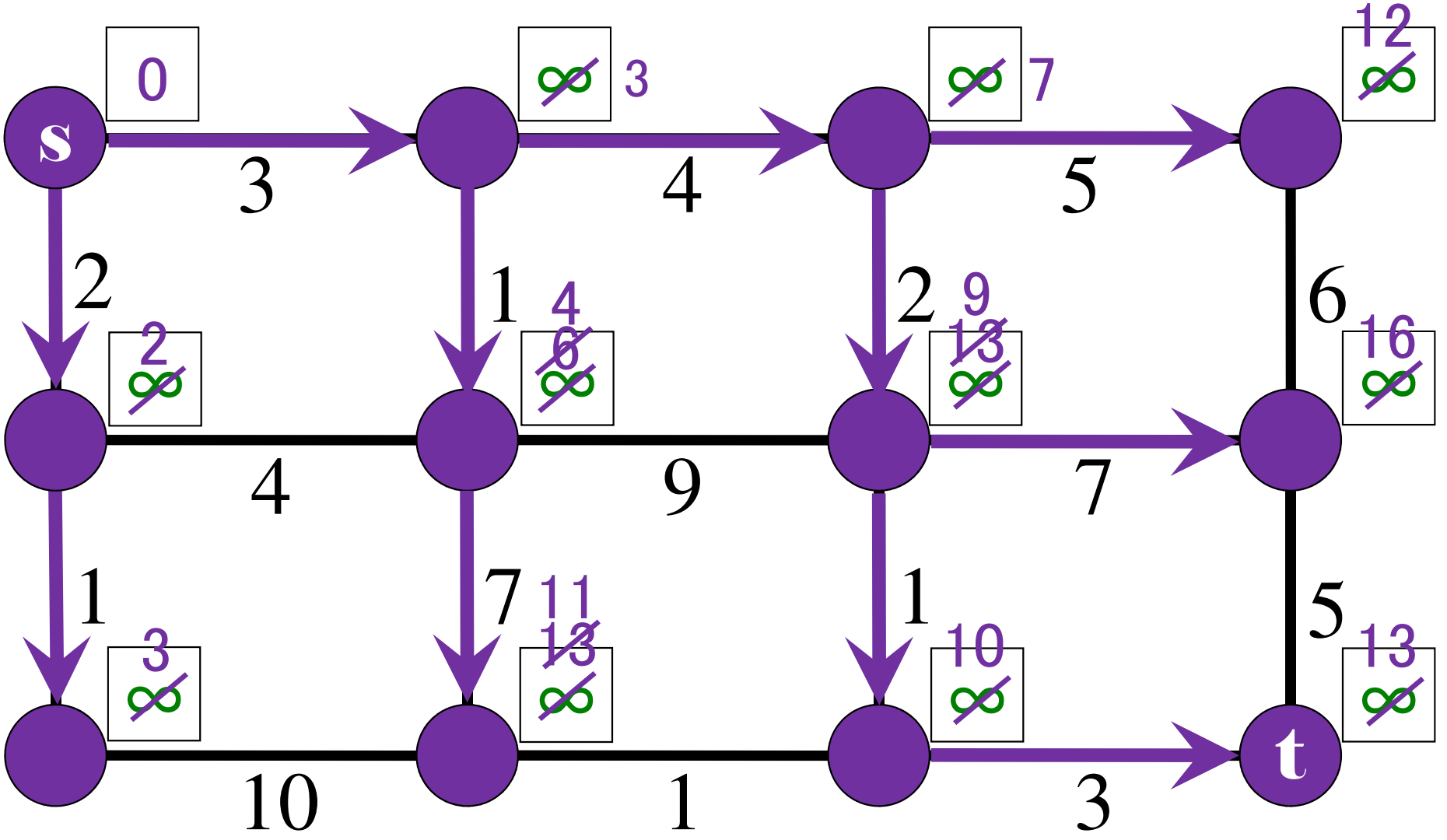


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

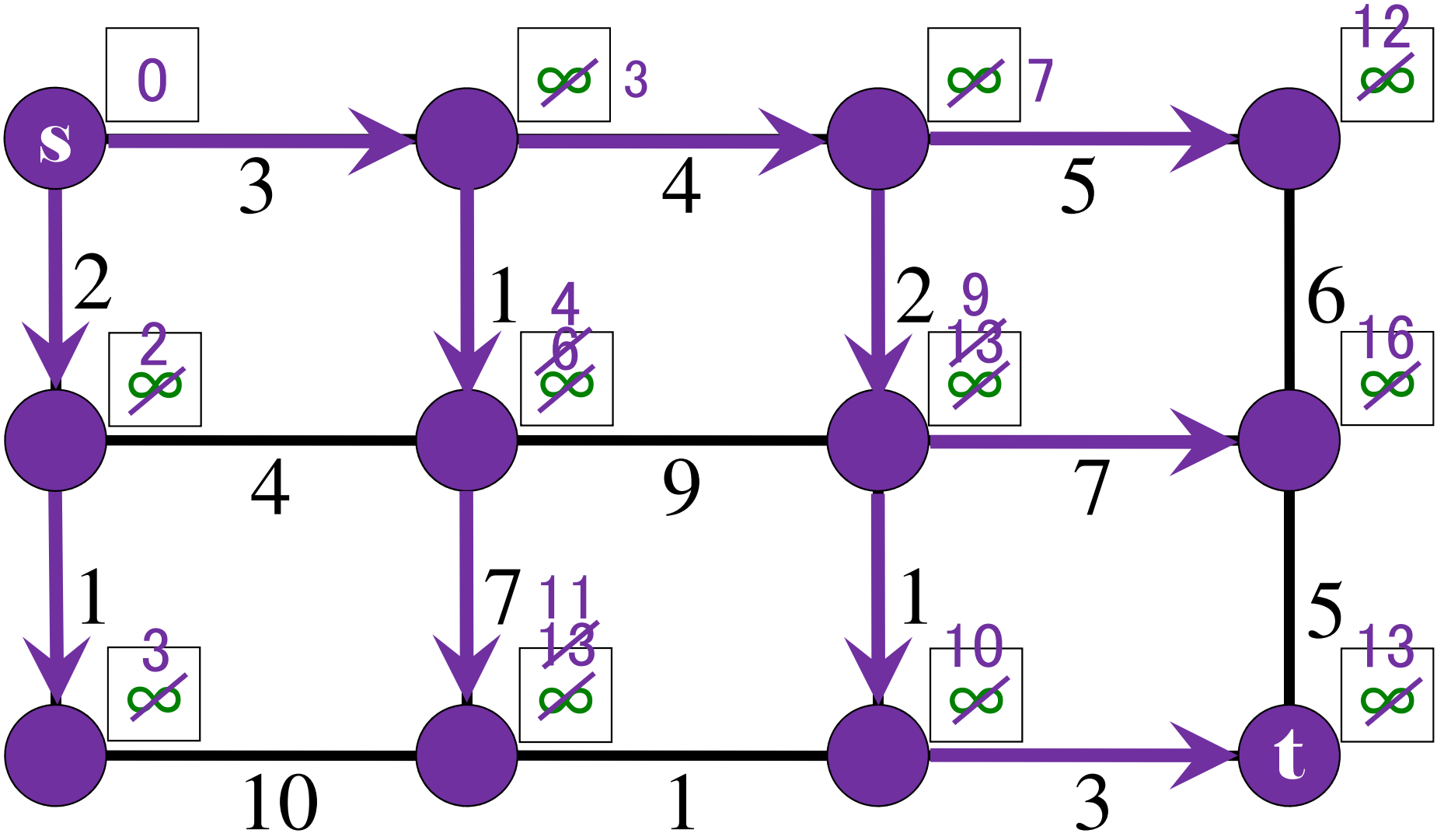
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



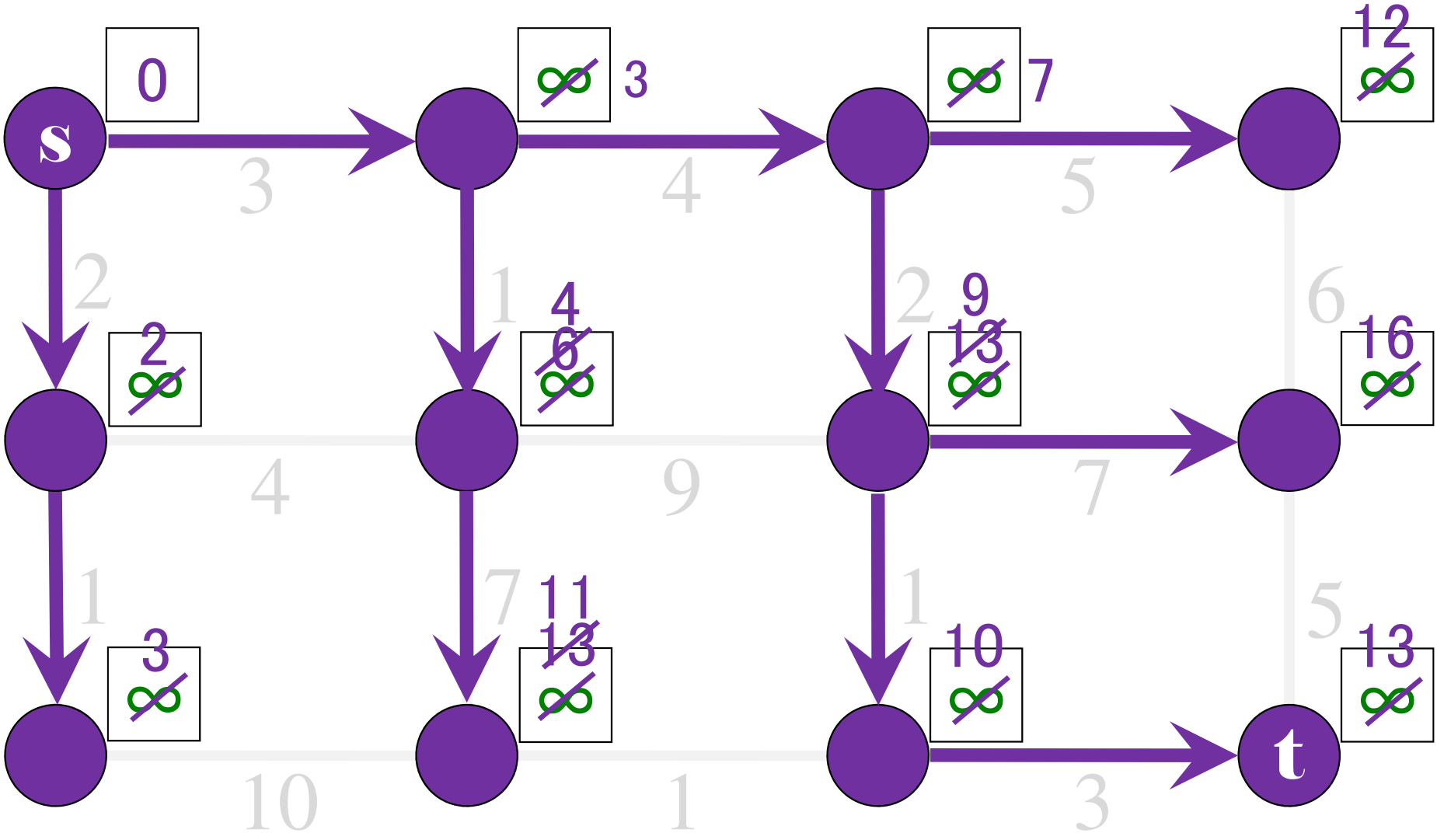
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



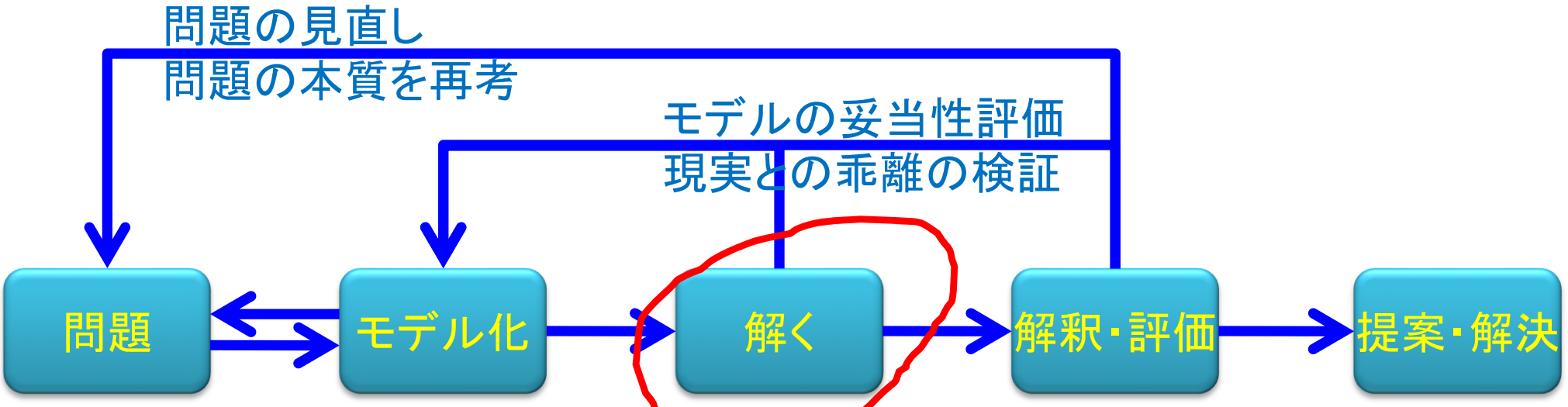
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

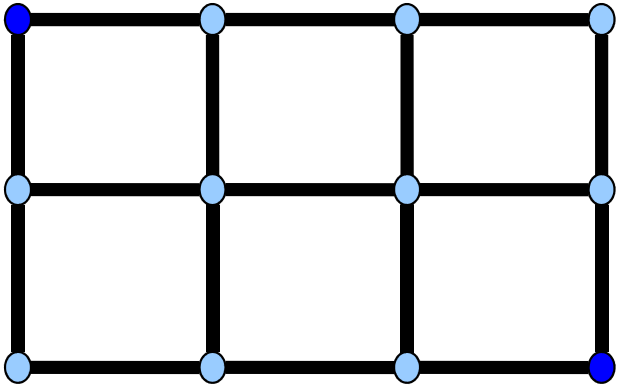
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

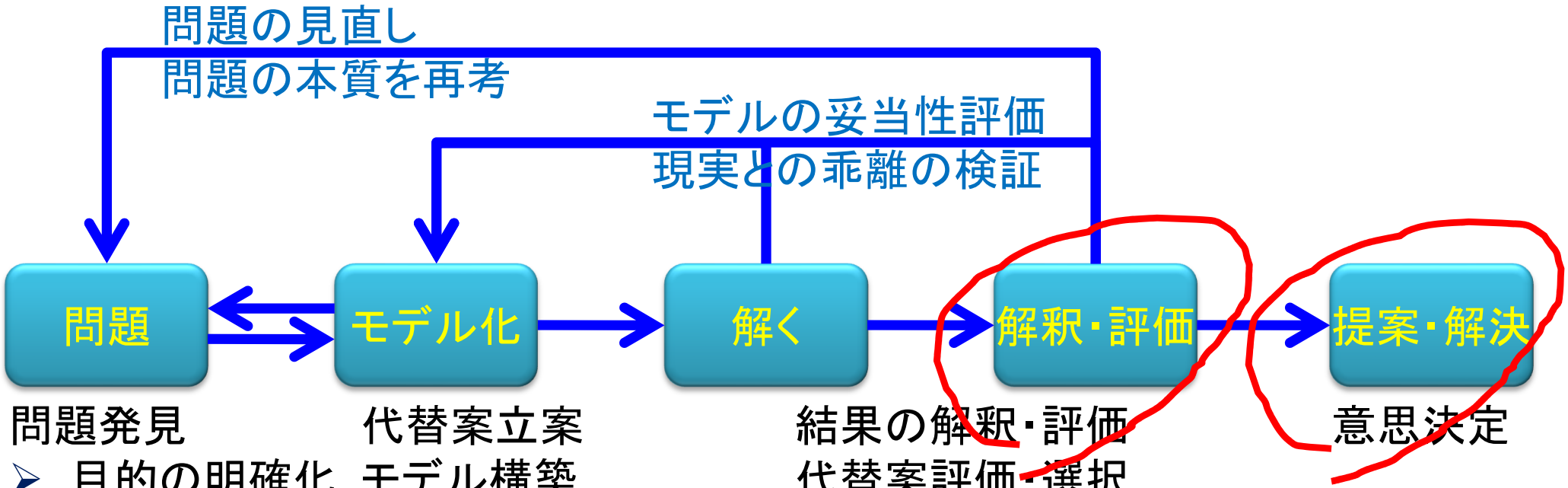
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

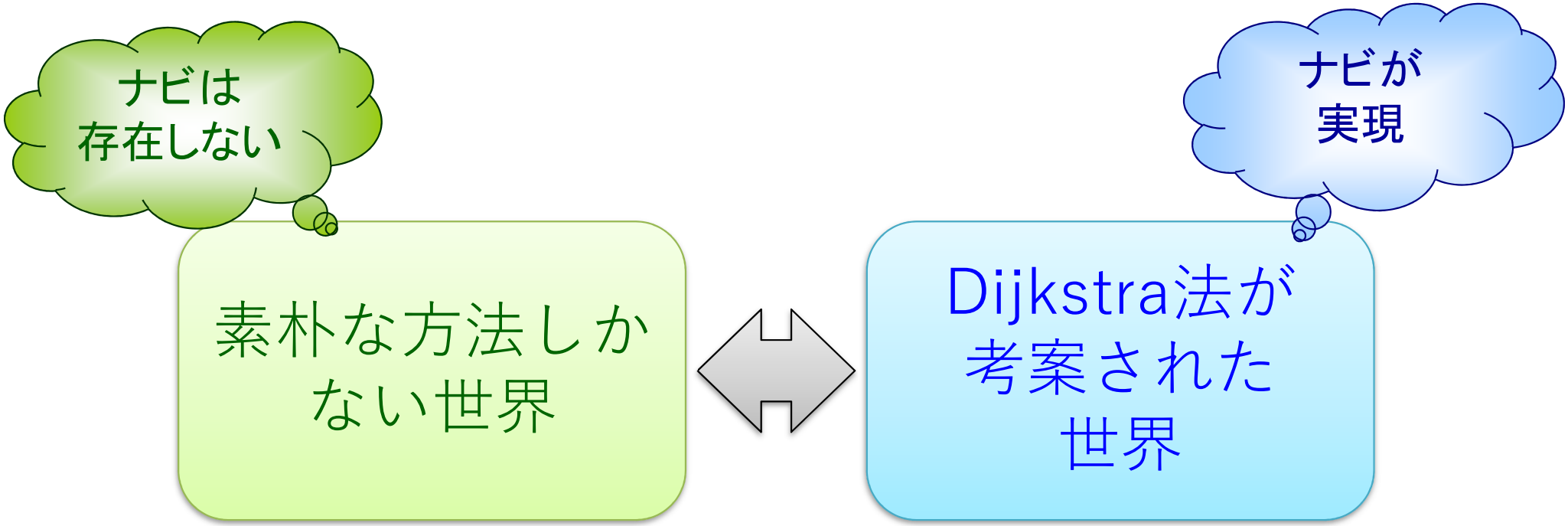
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

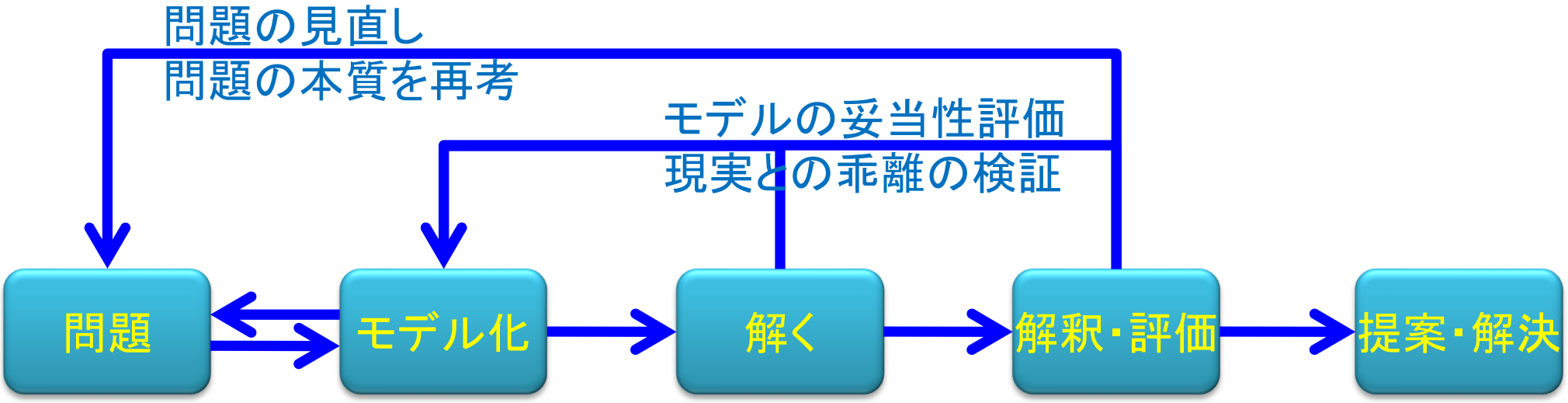
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



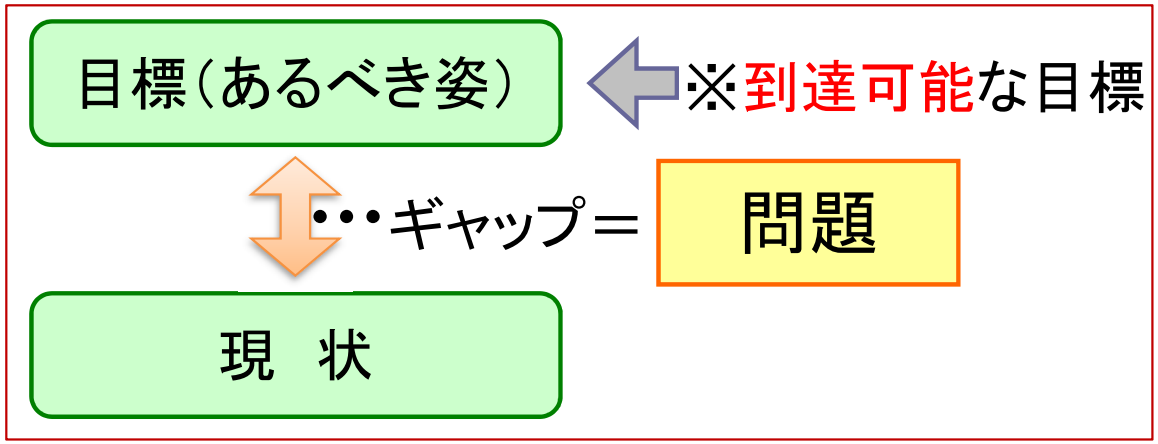
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

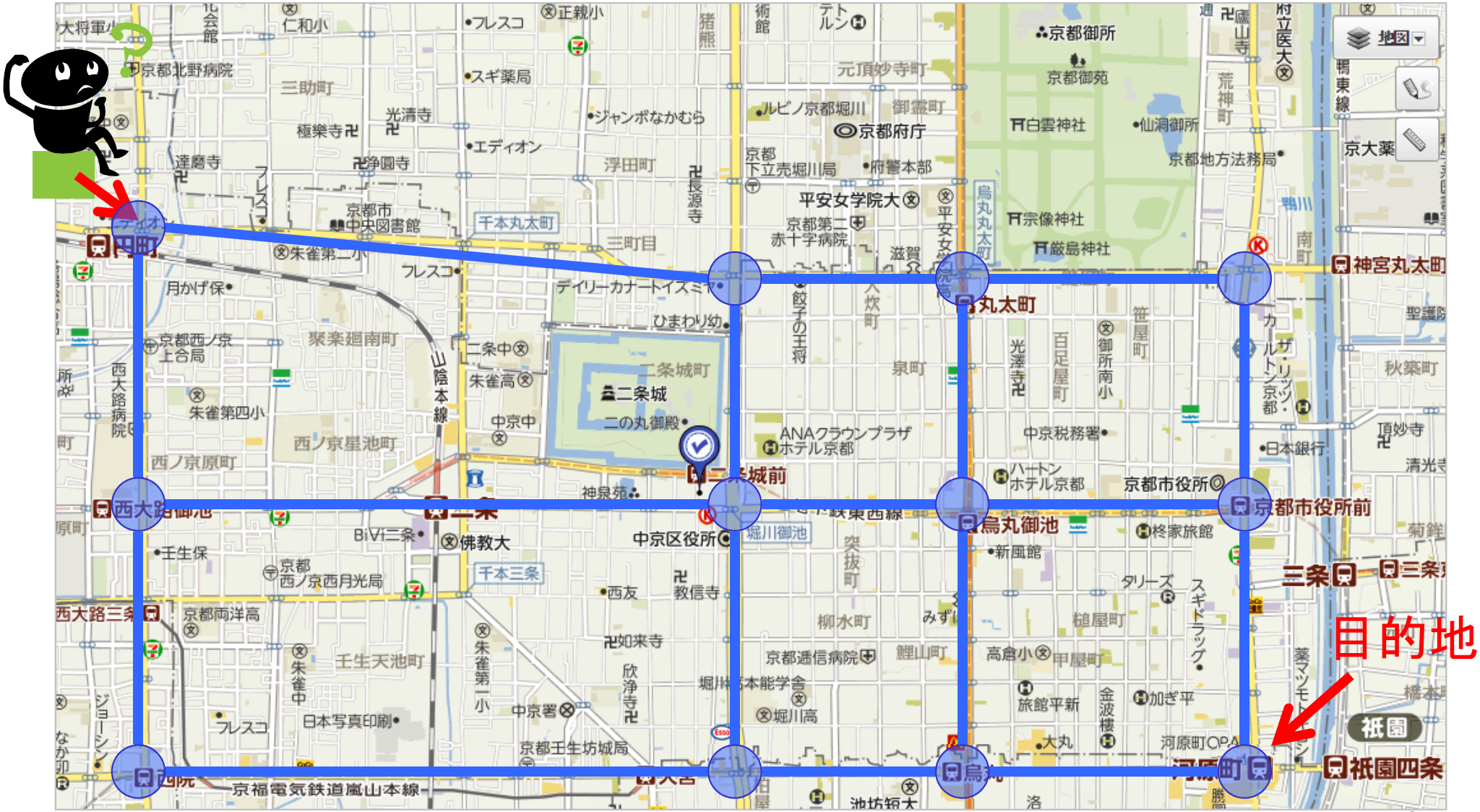
意思決定

問題の定義



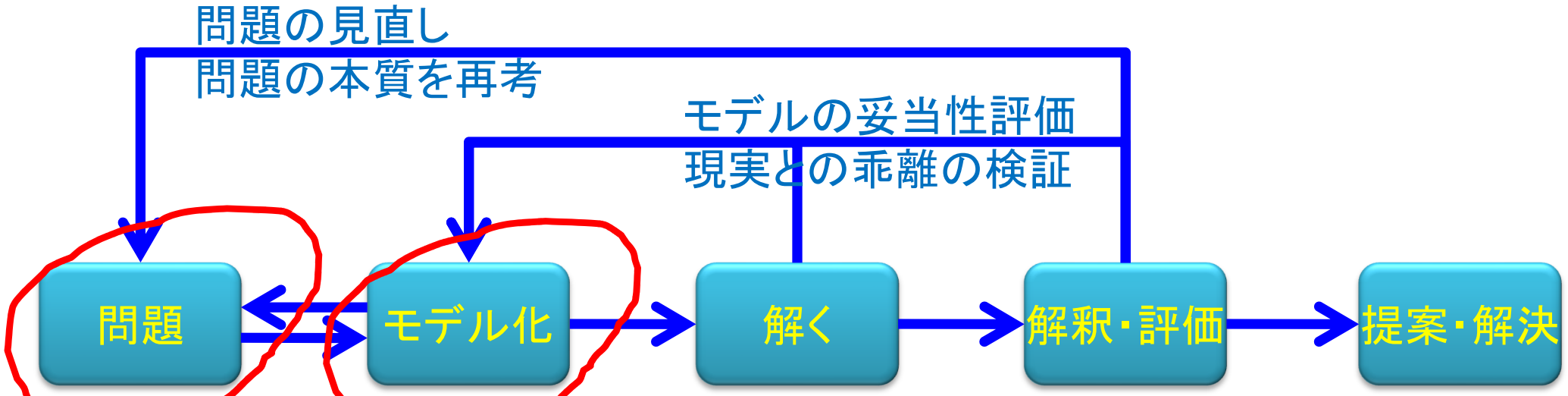
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

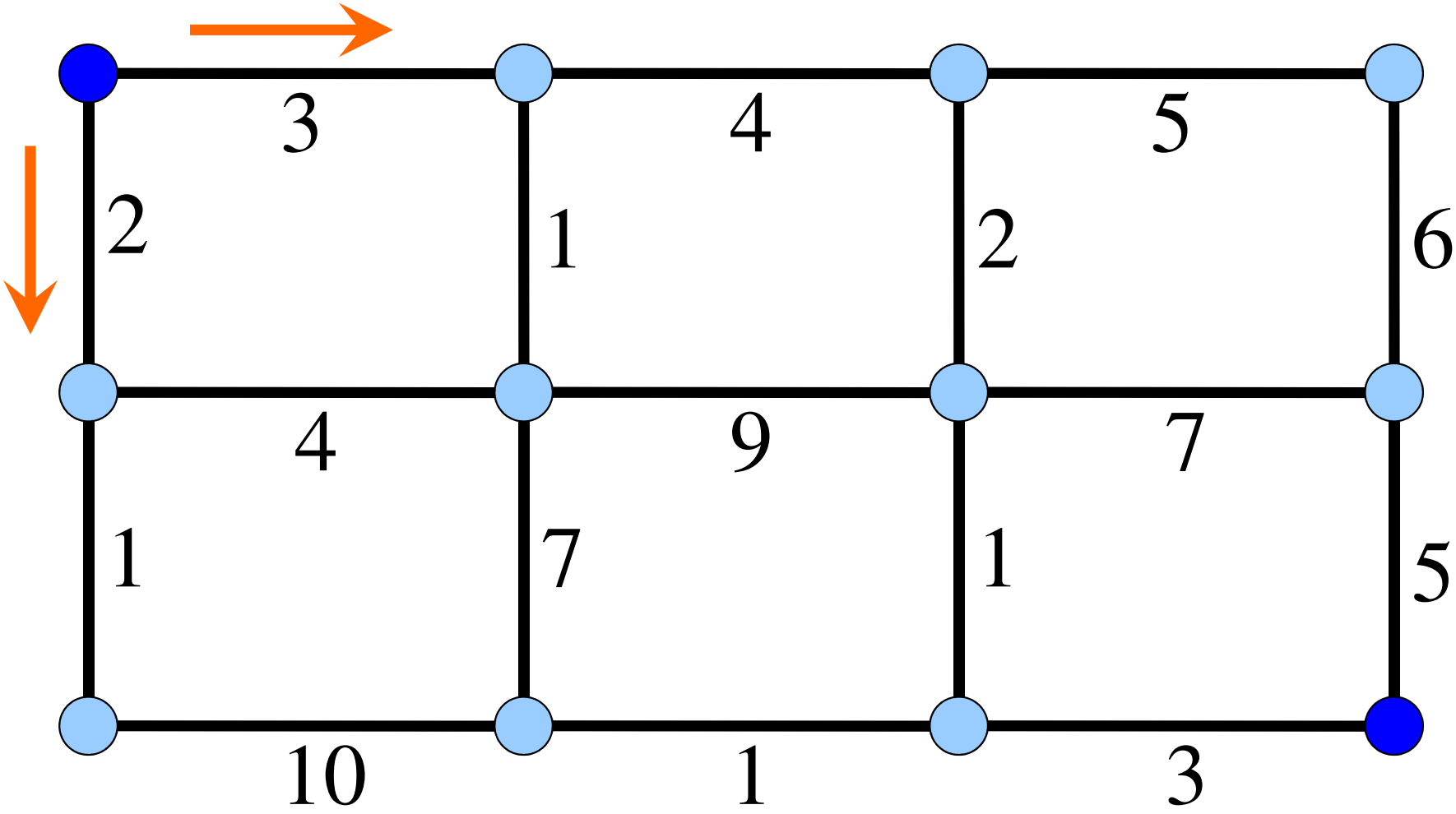
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

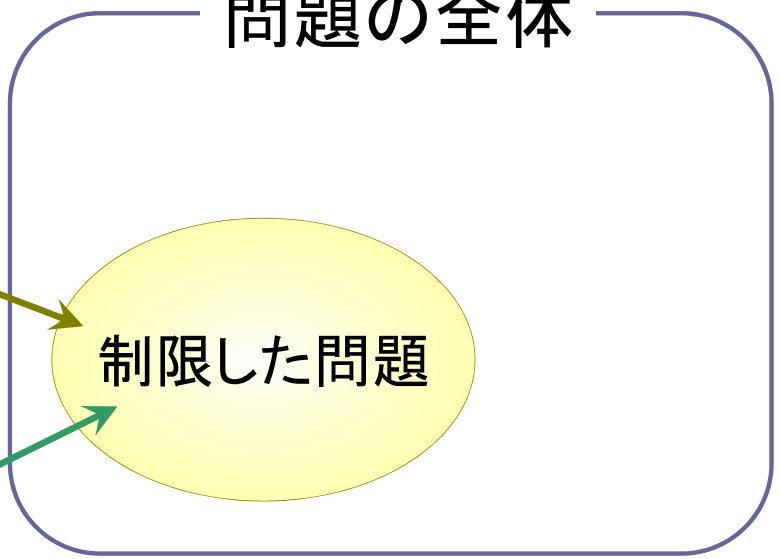


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

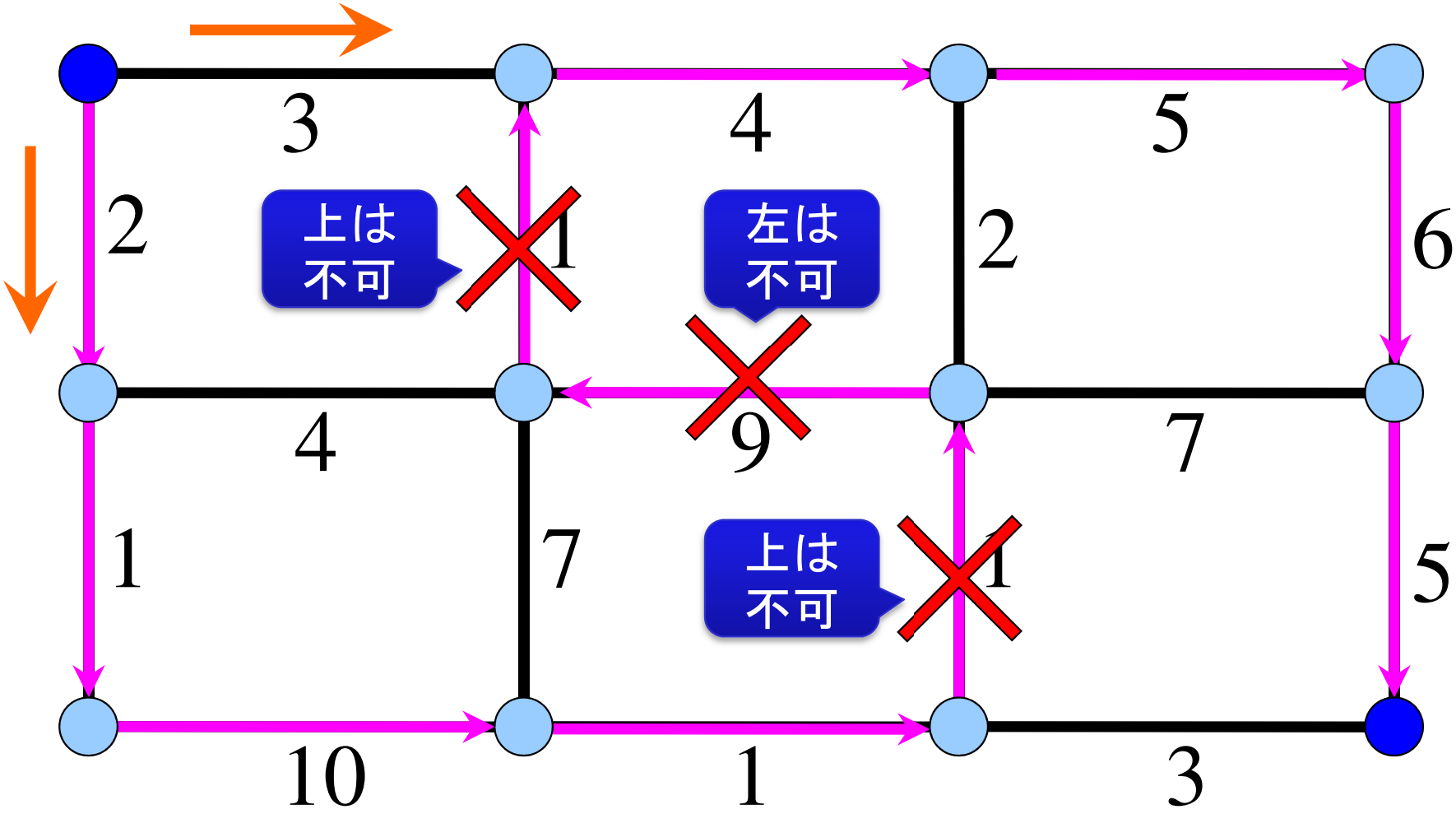
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

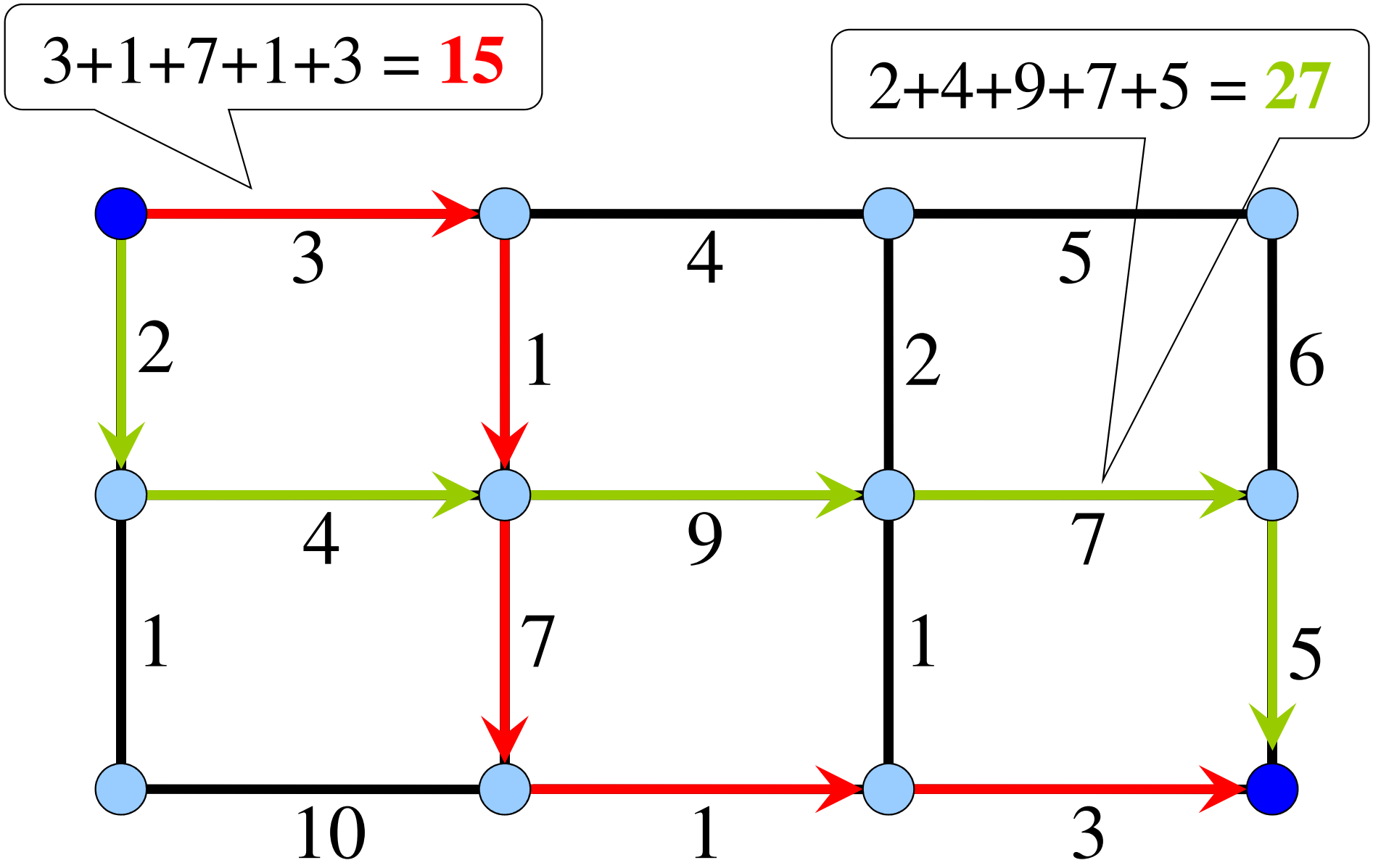
難しいなら易しくすればいいのさ！

制限した問題

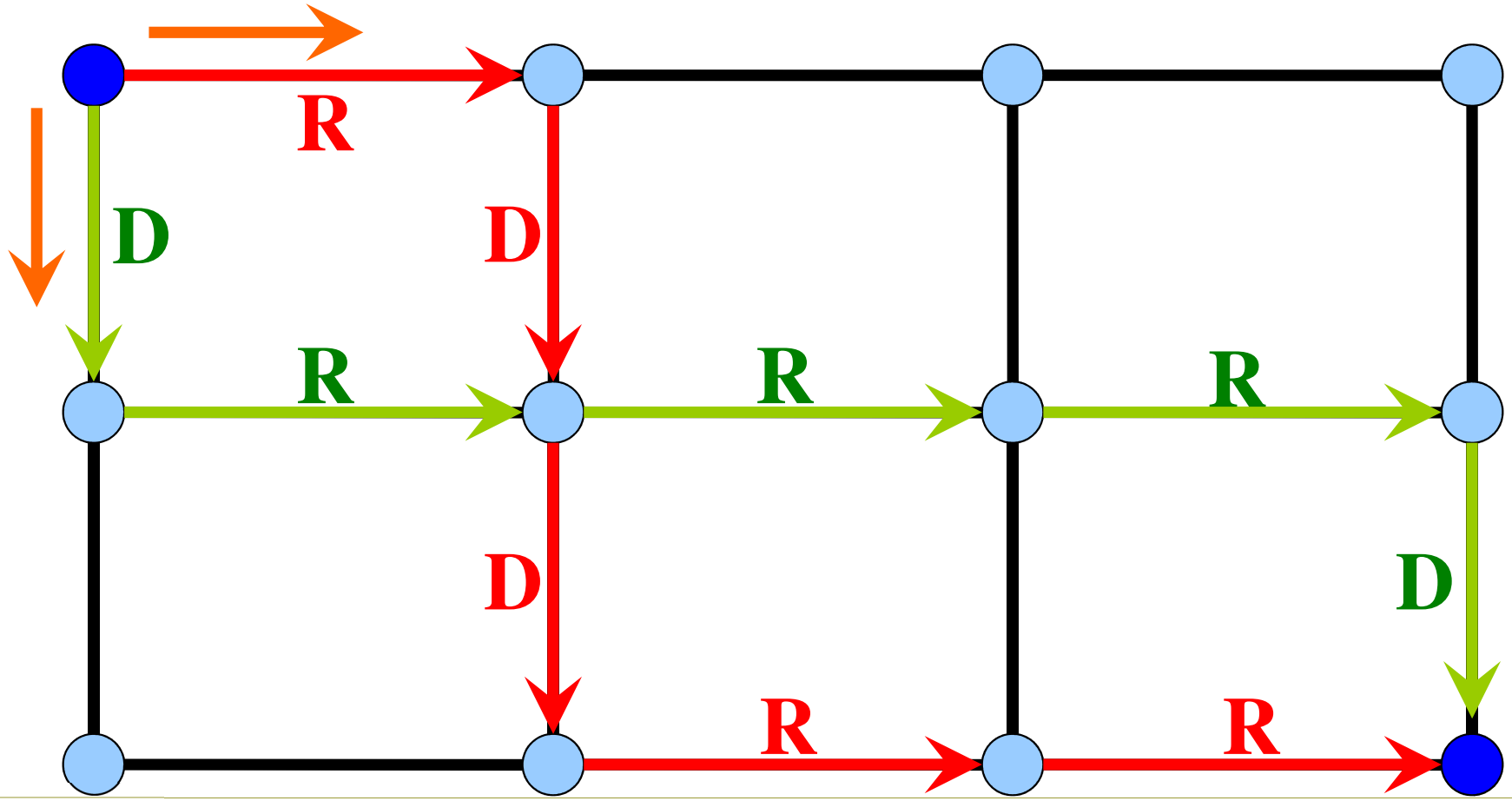
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



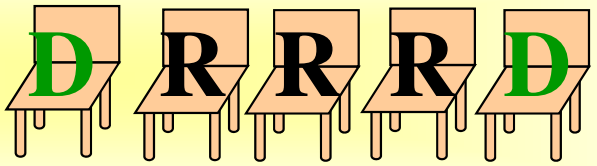
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



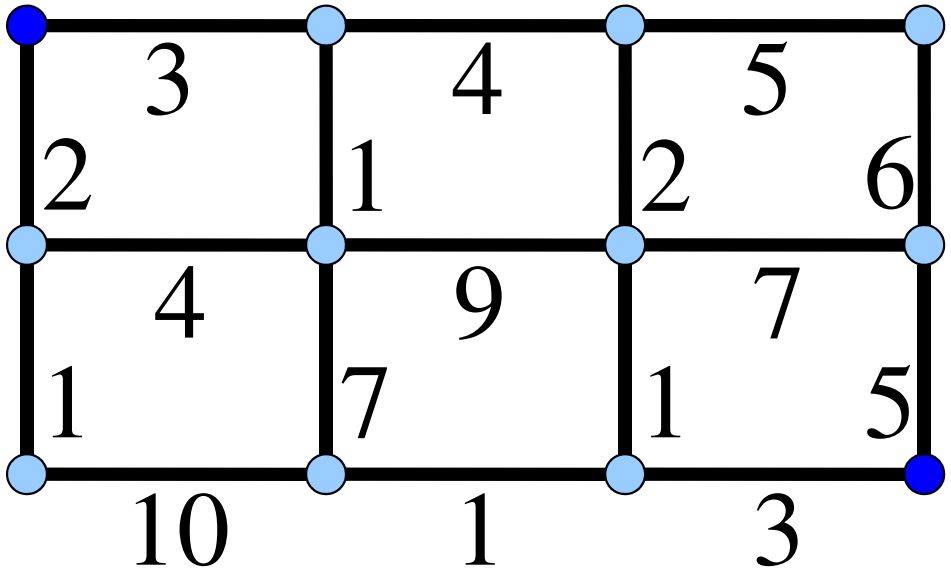
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

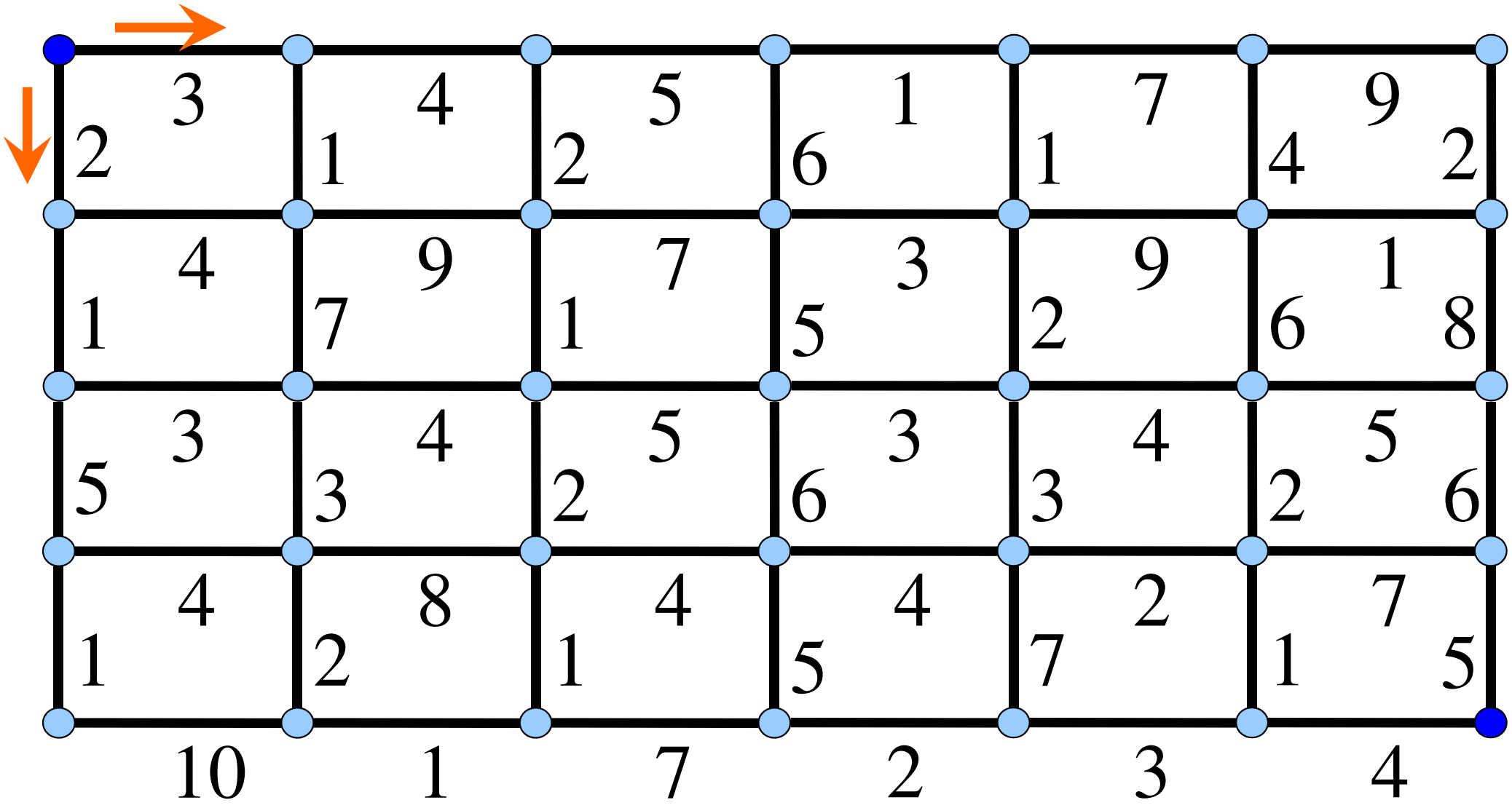
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

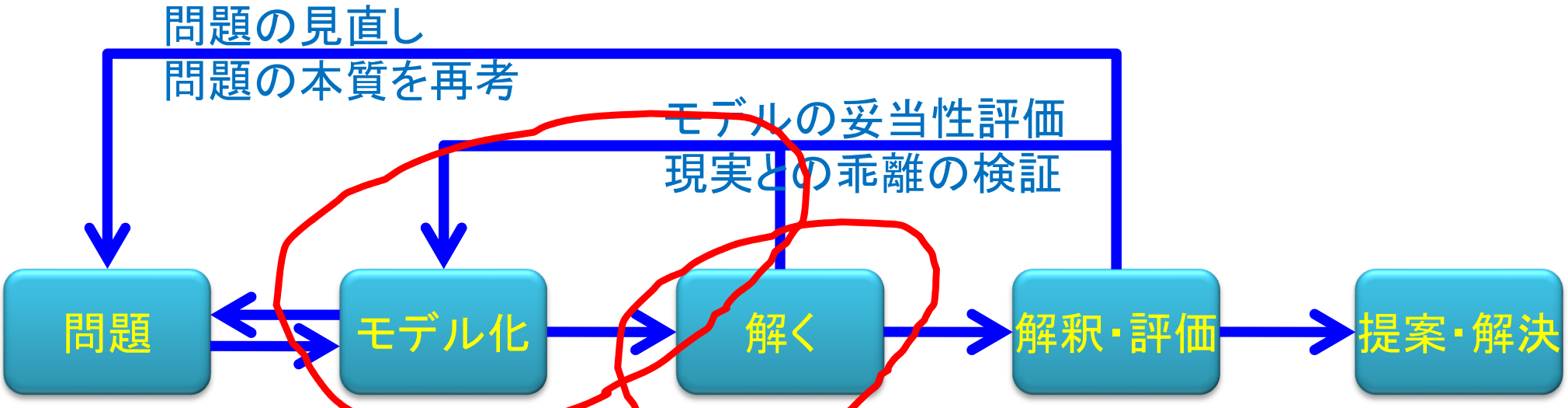
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

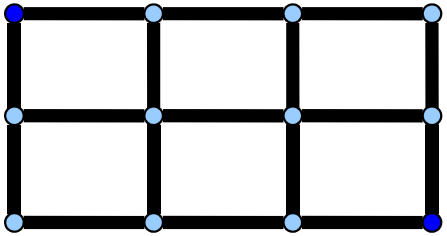
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

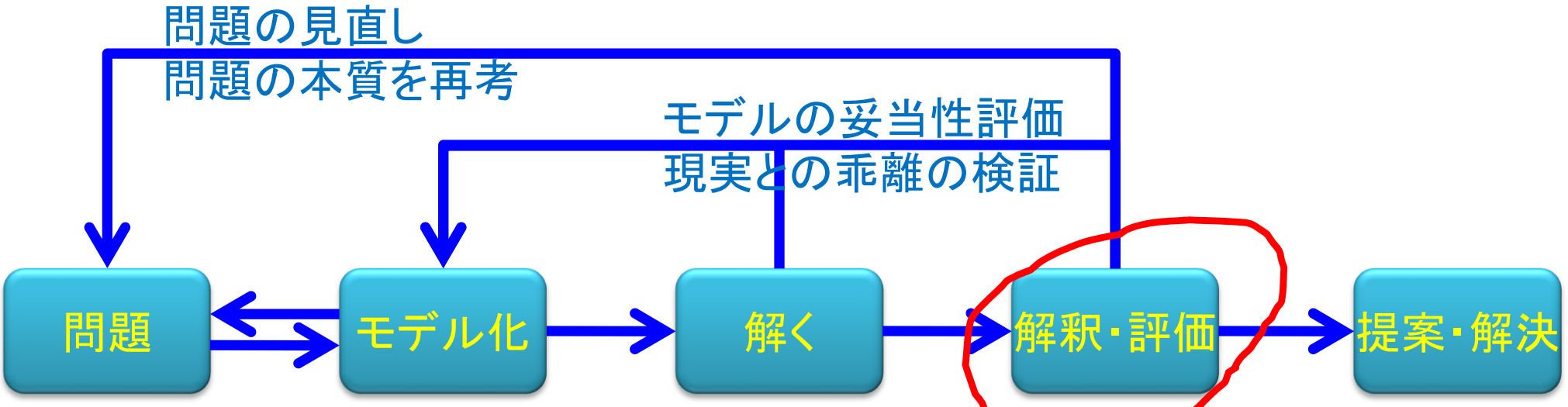
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

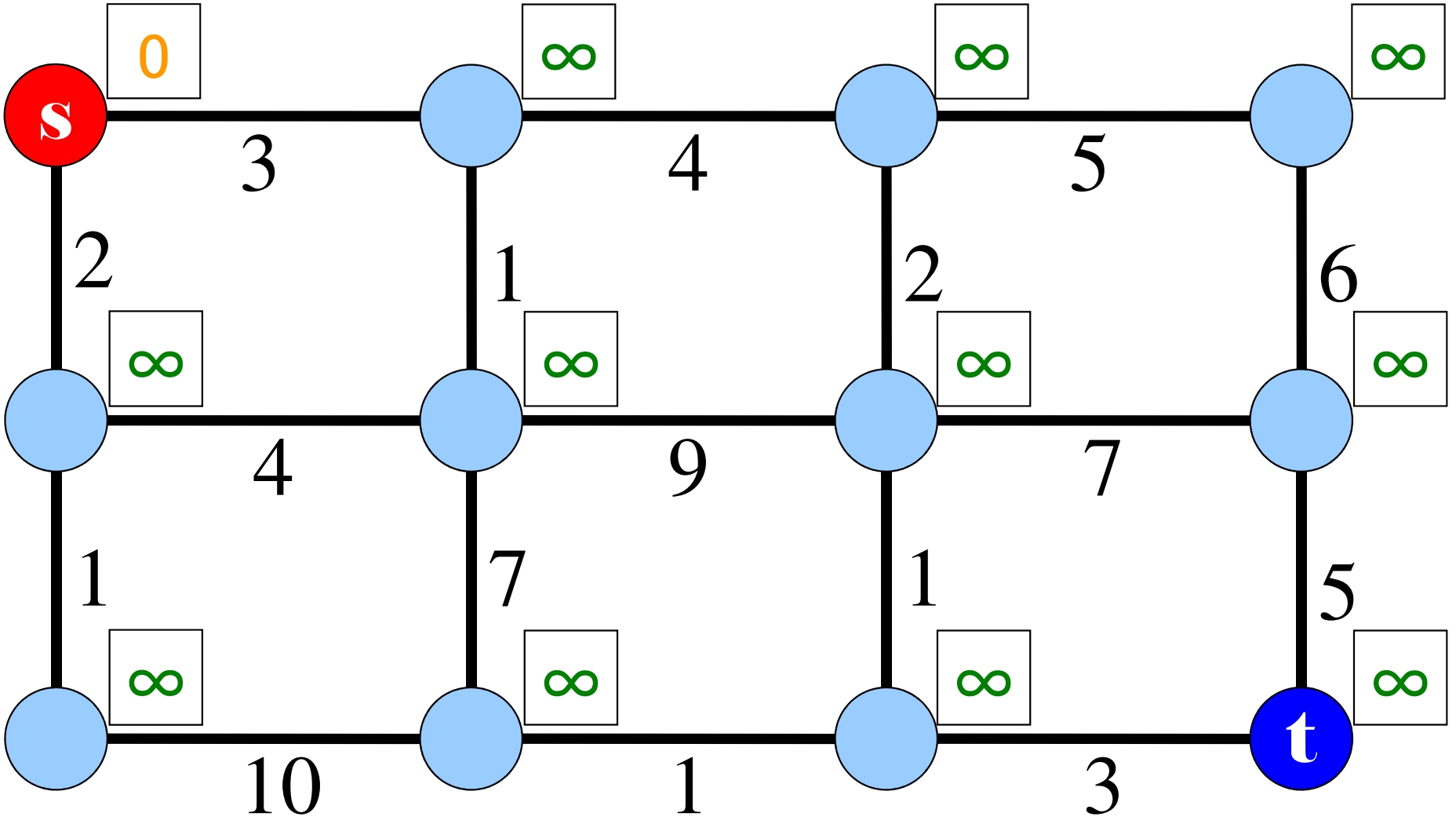
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

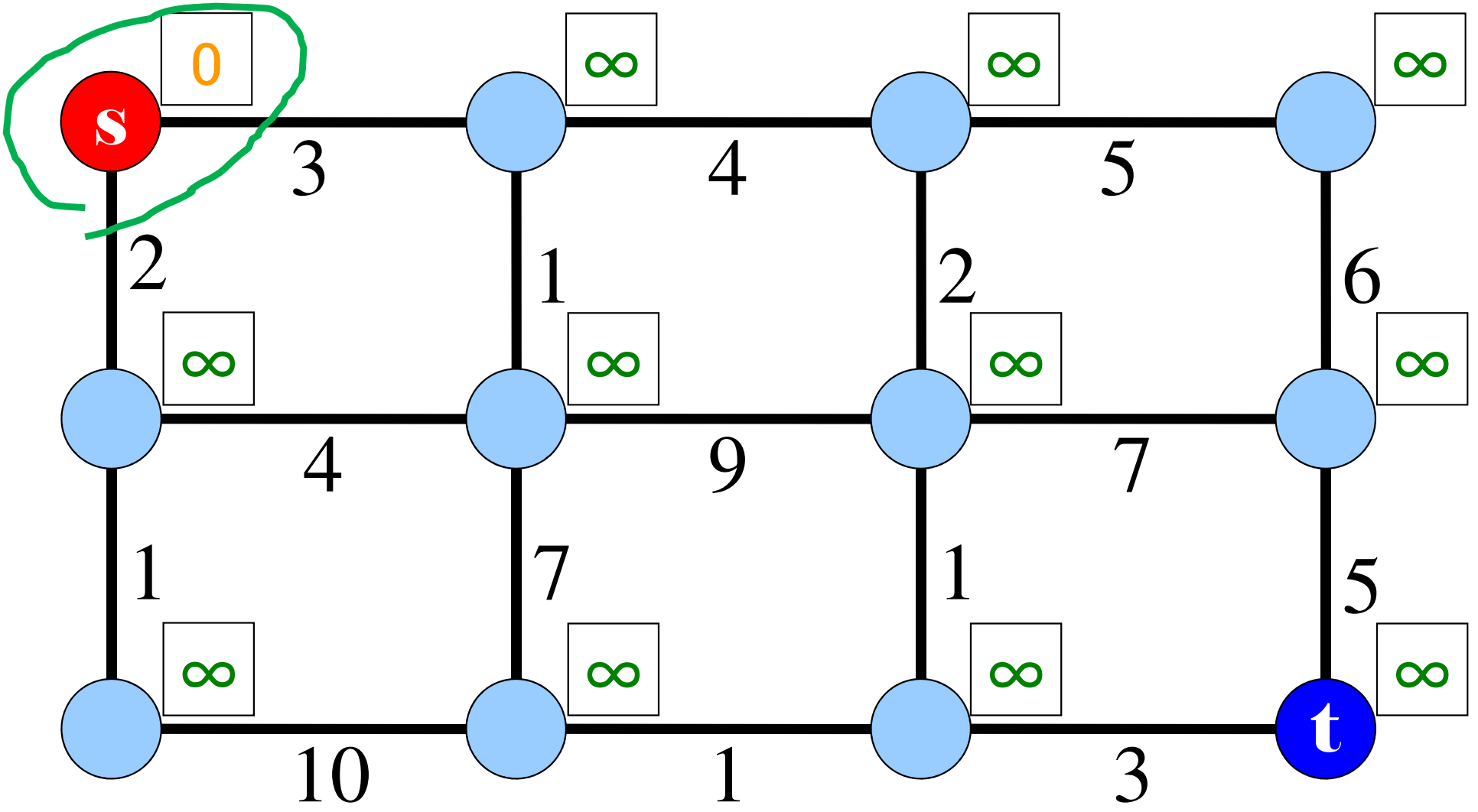
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



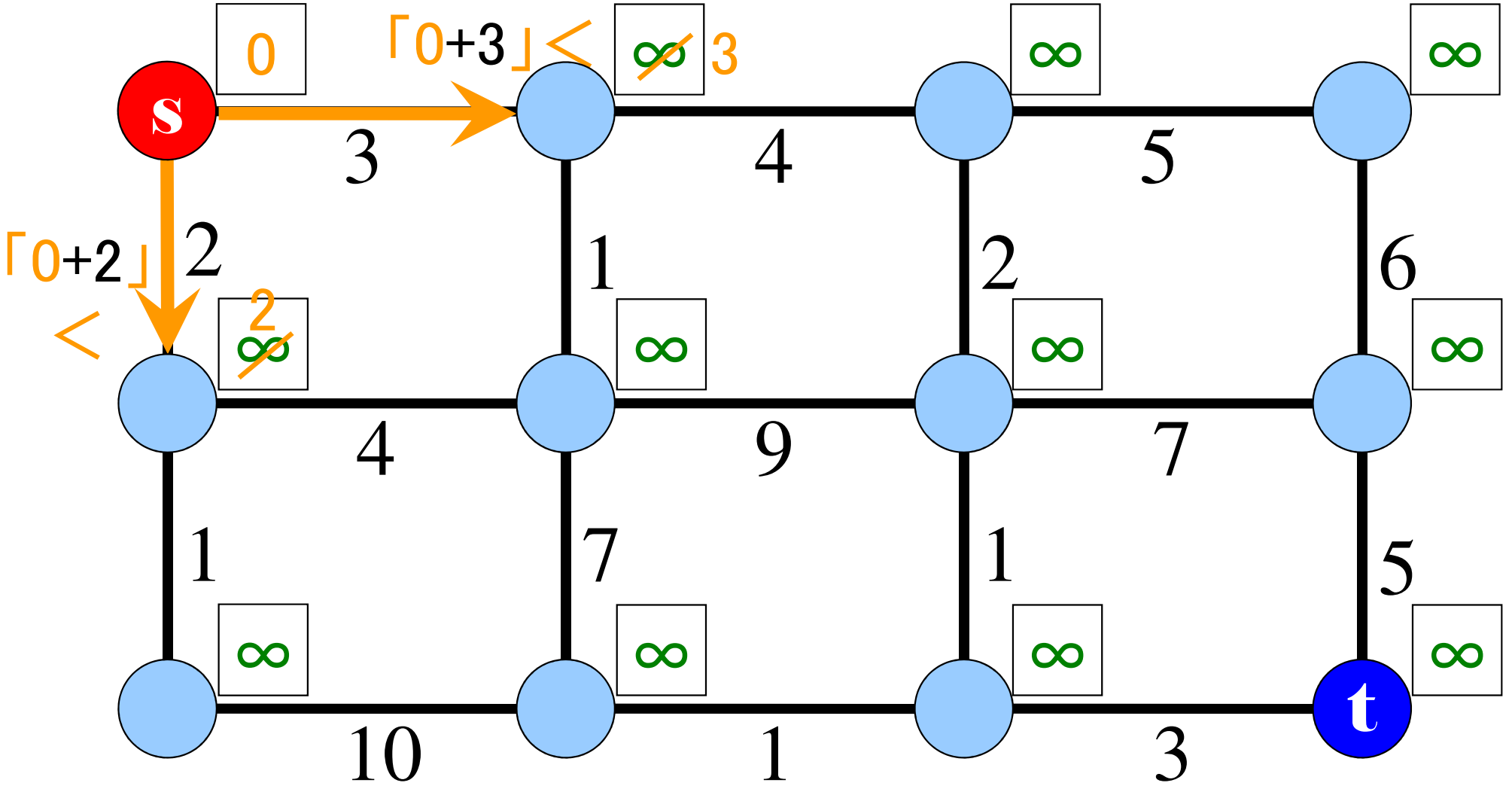
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

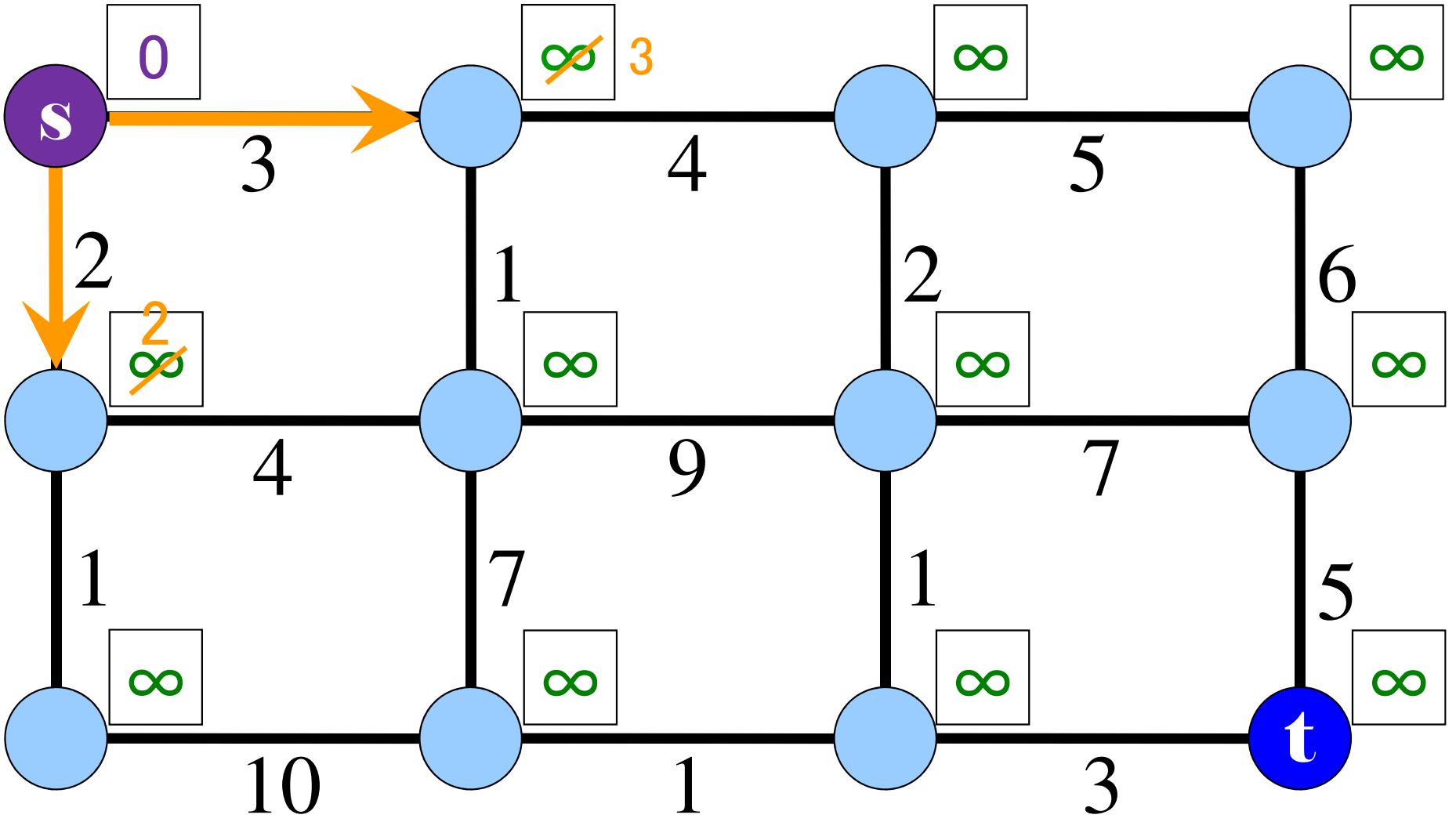


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

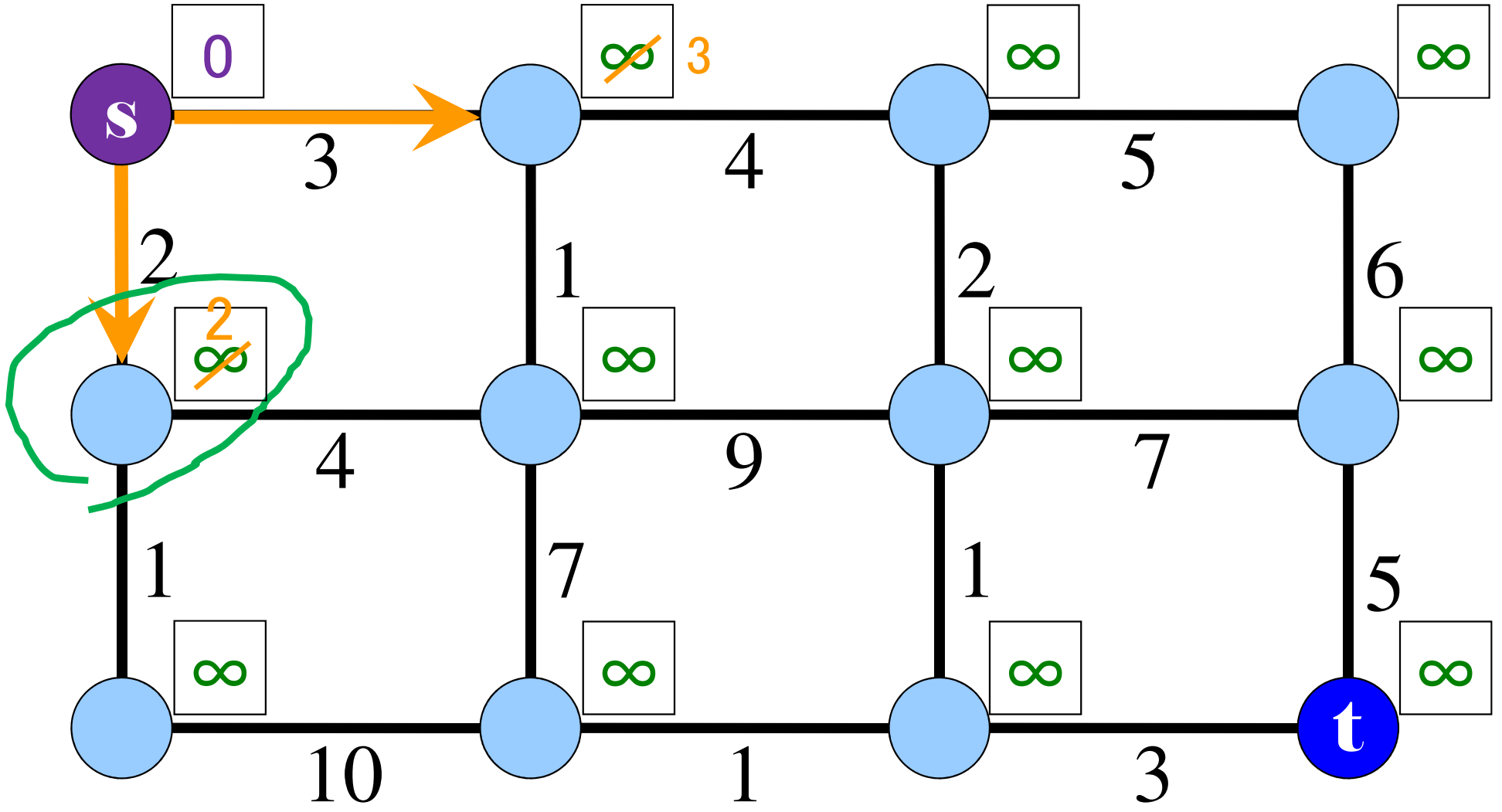
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



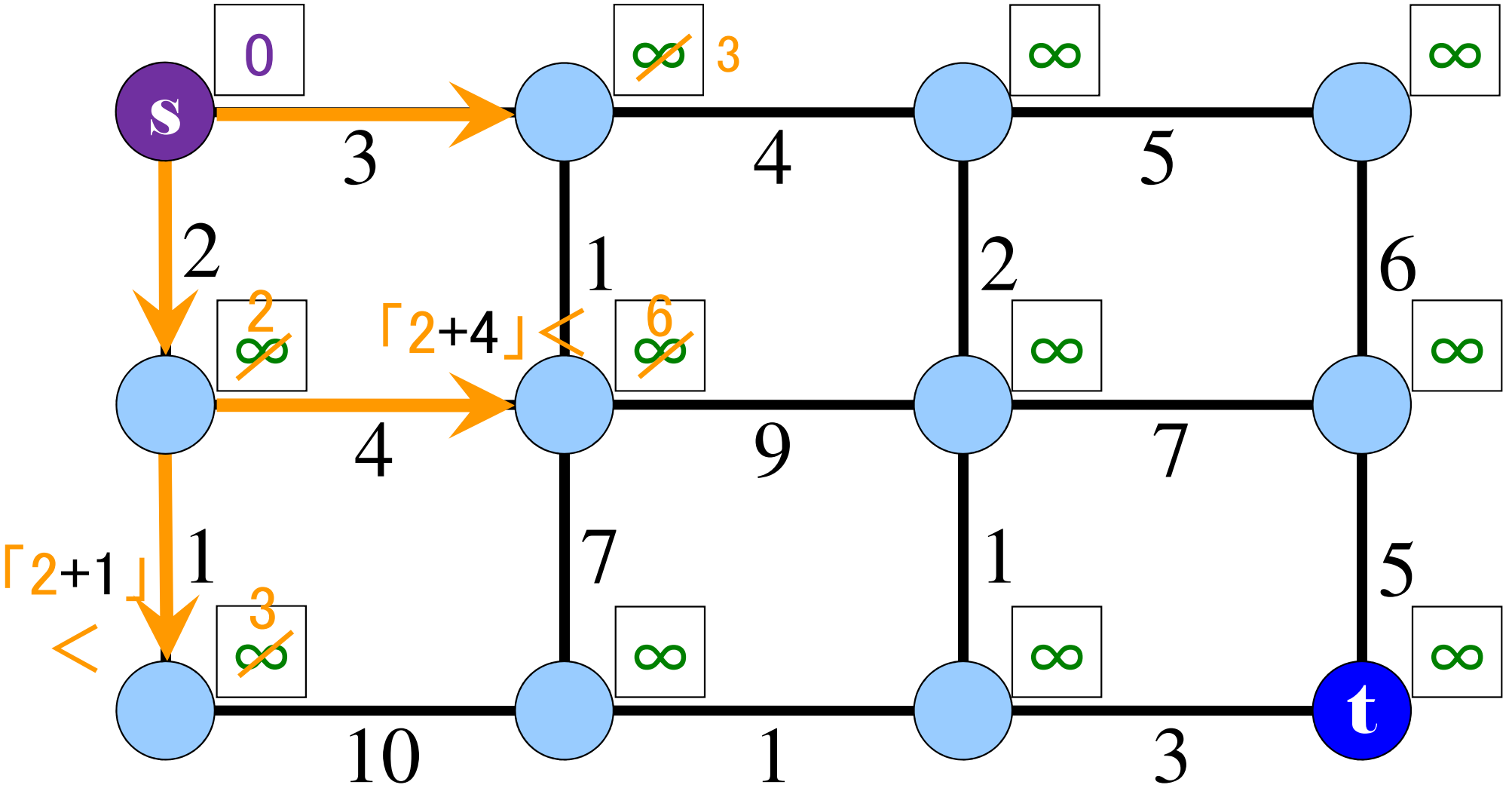
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

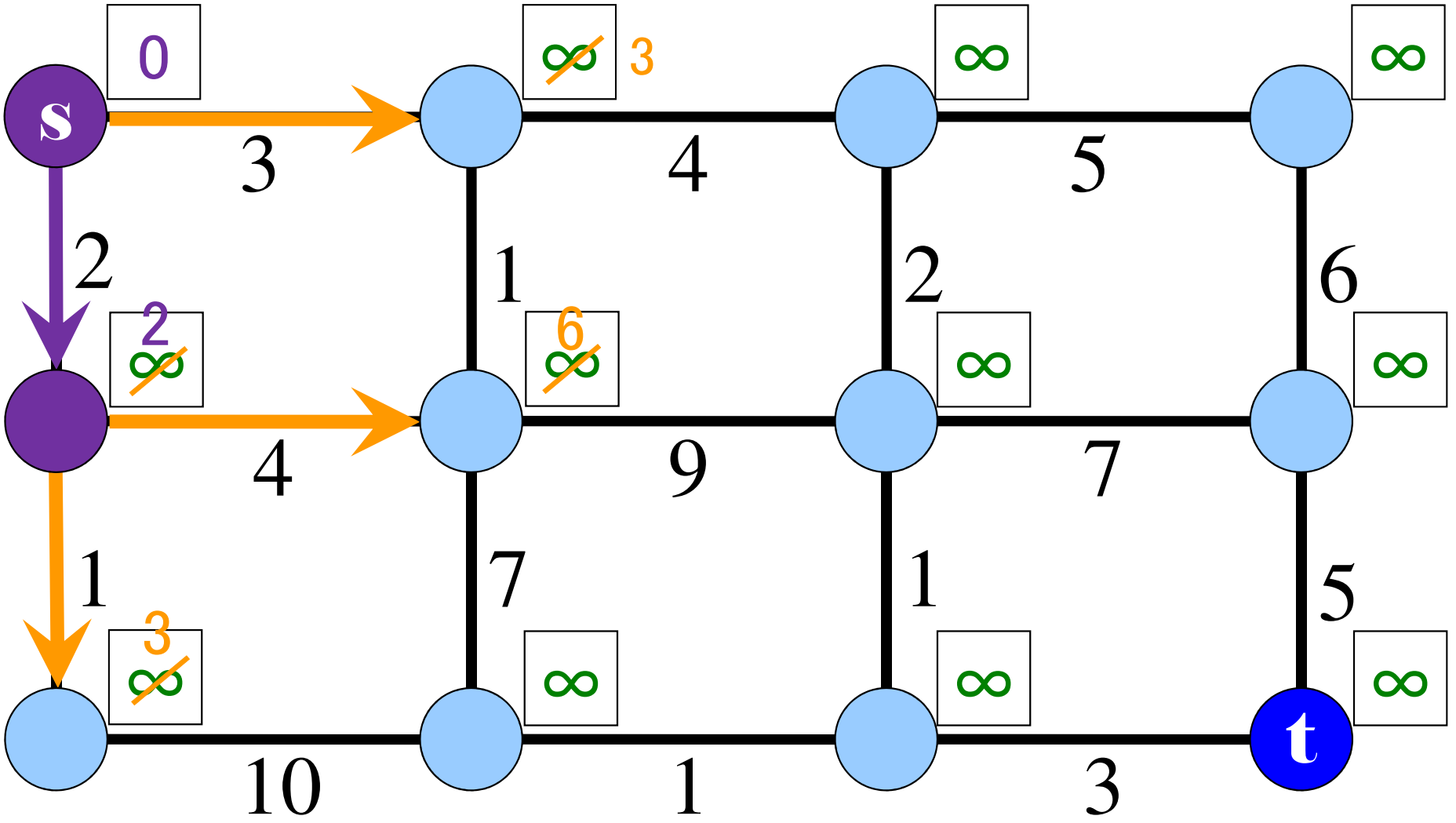


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

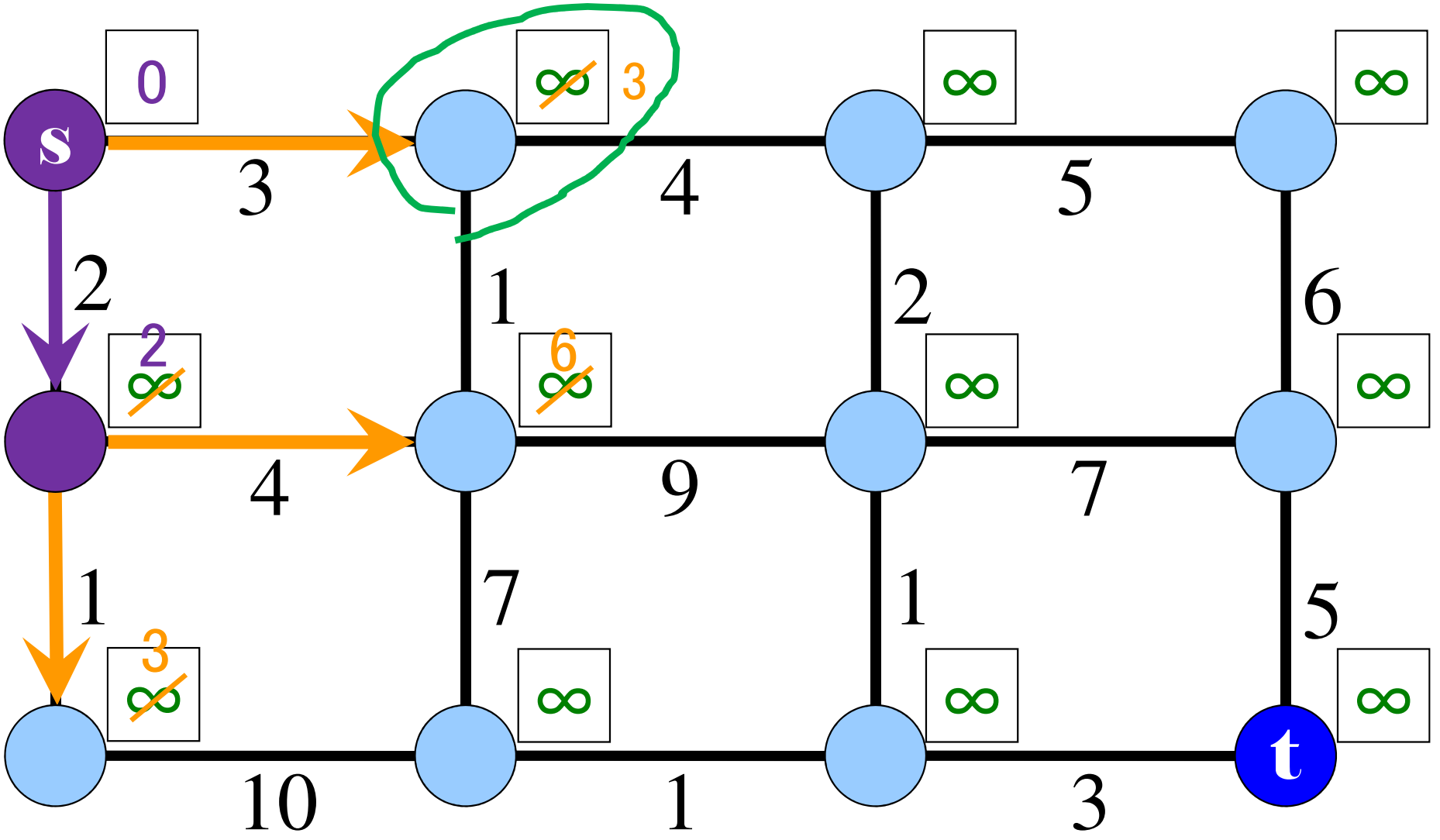
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



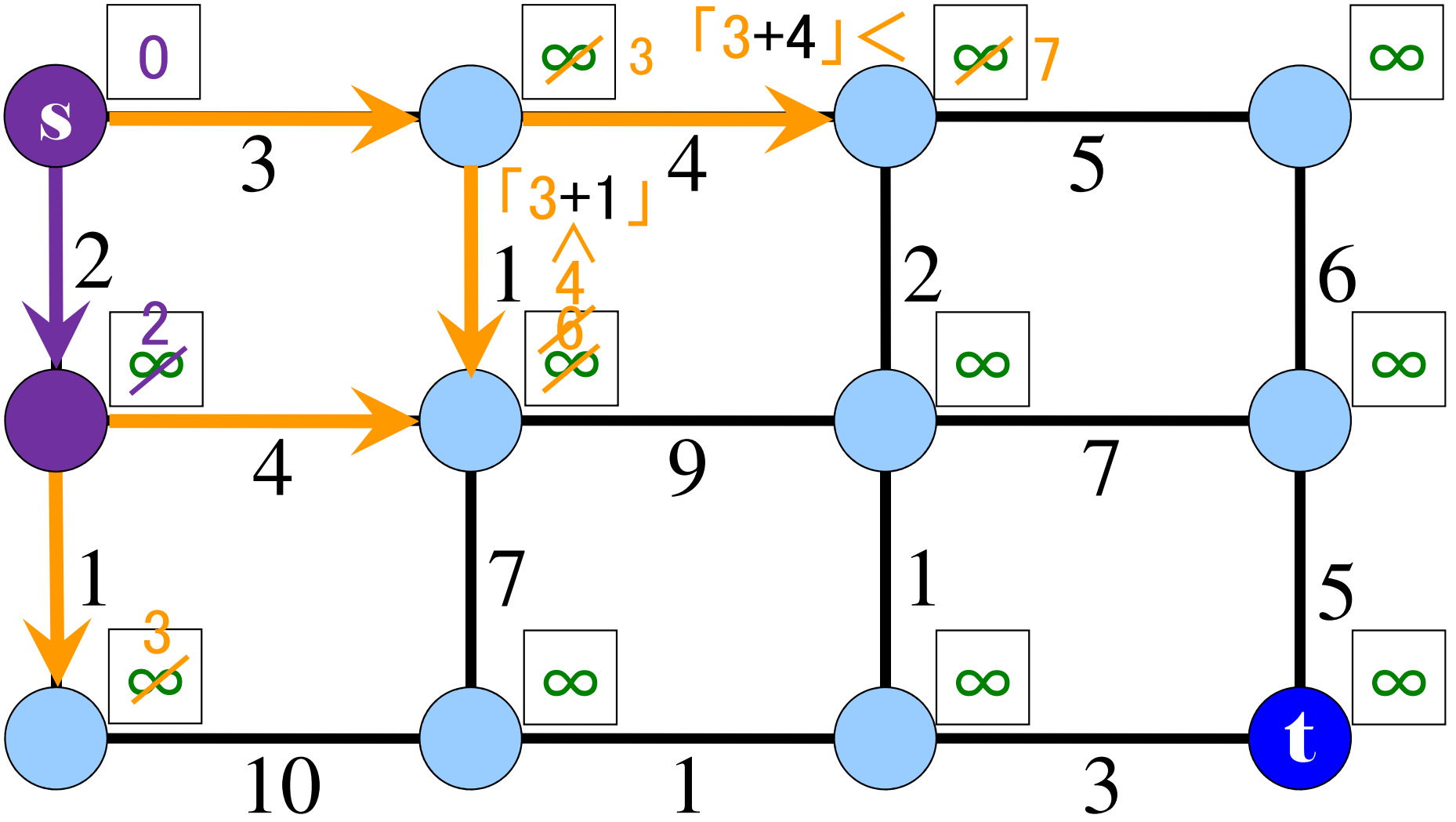
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

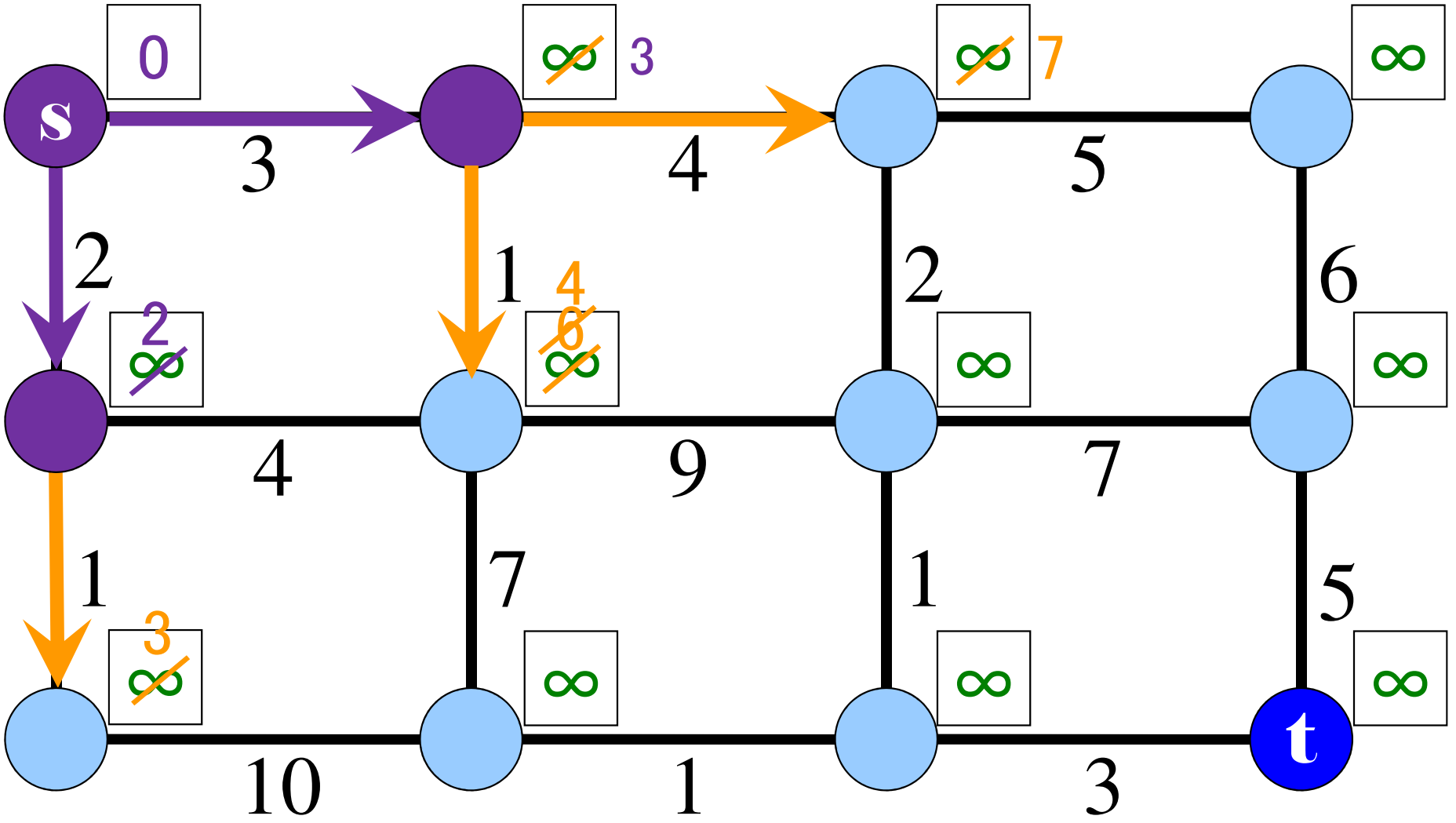


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

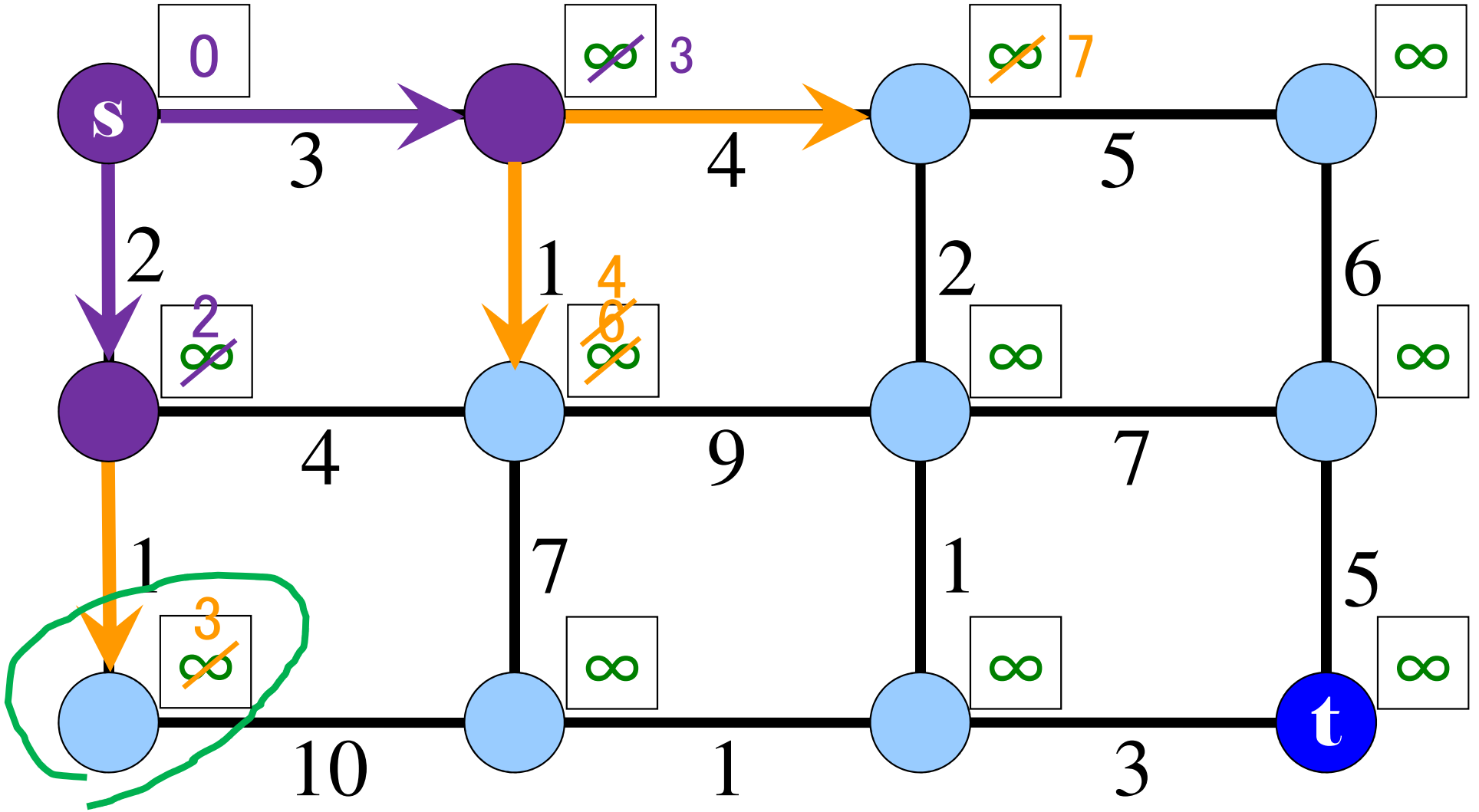
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



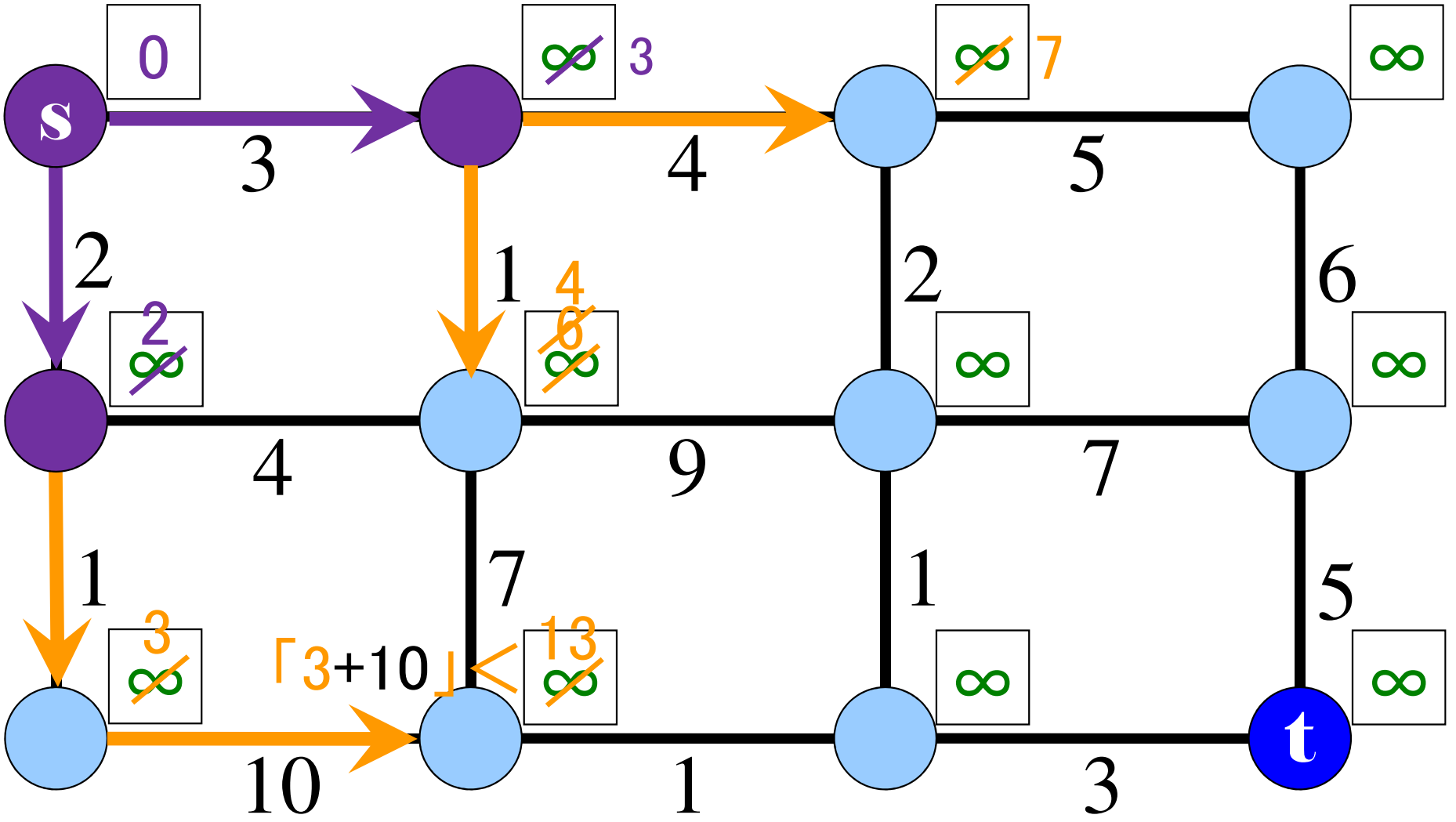
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

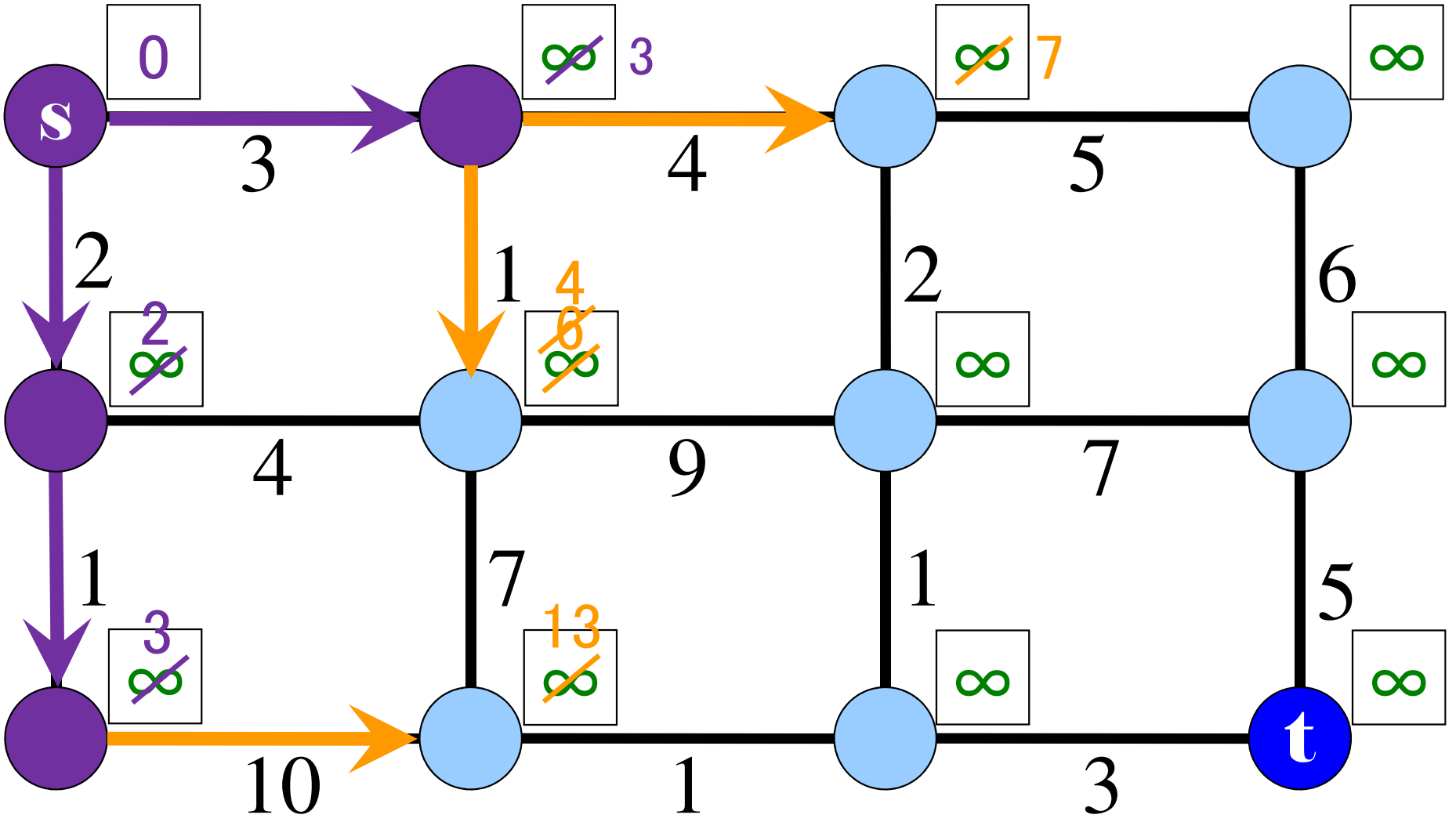


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

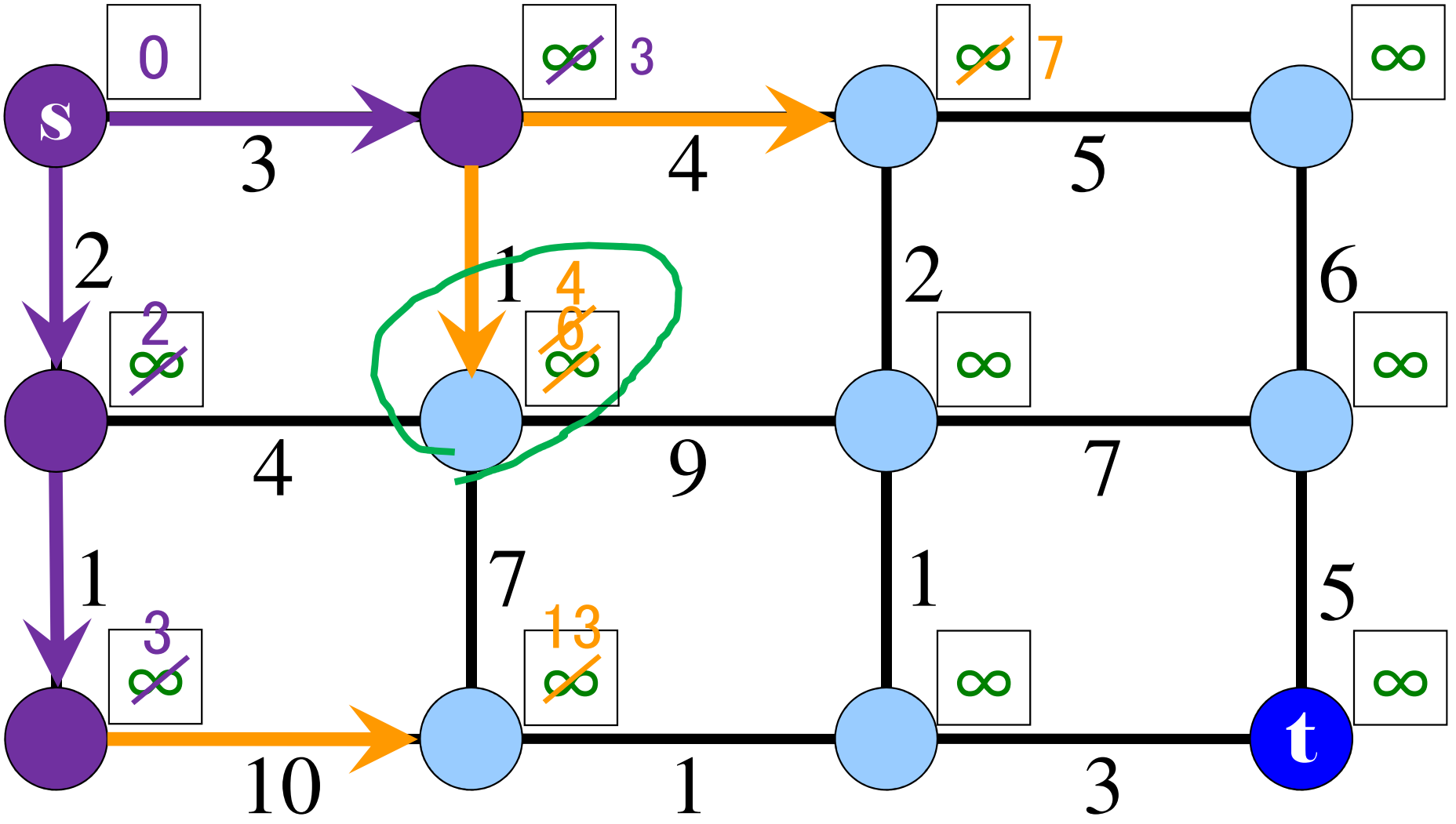
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



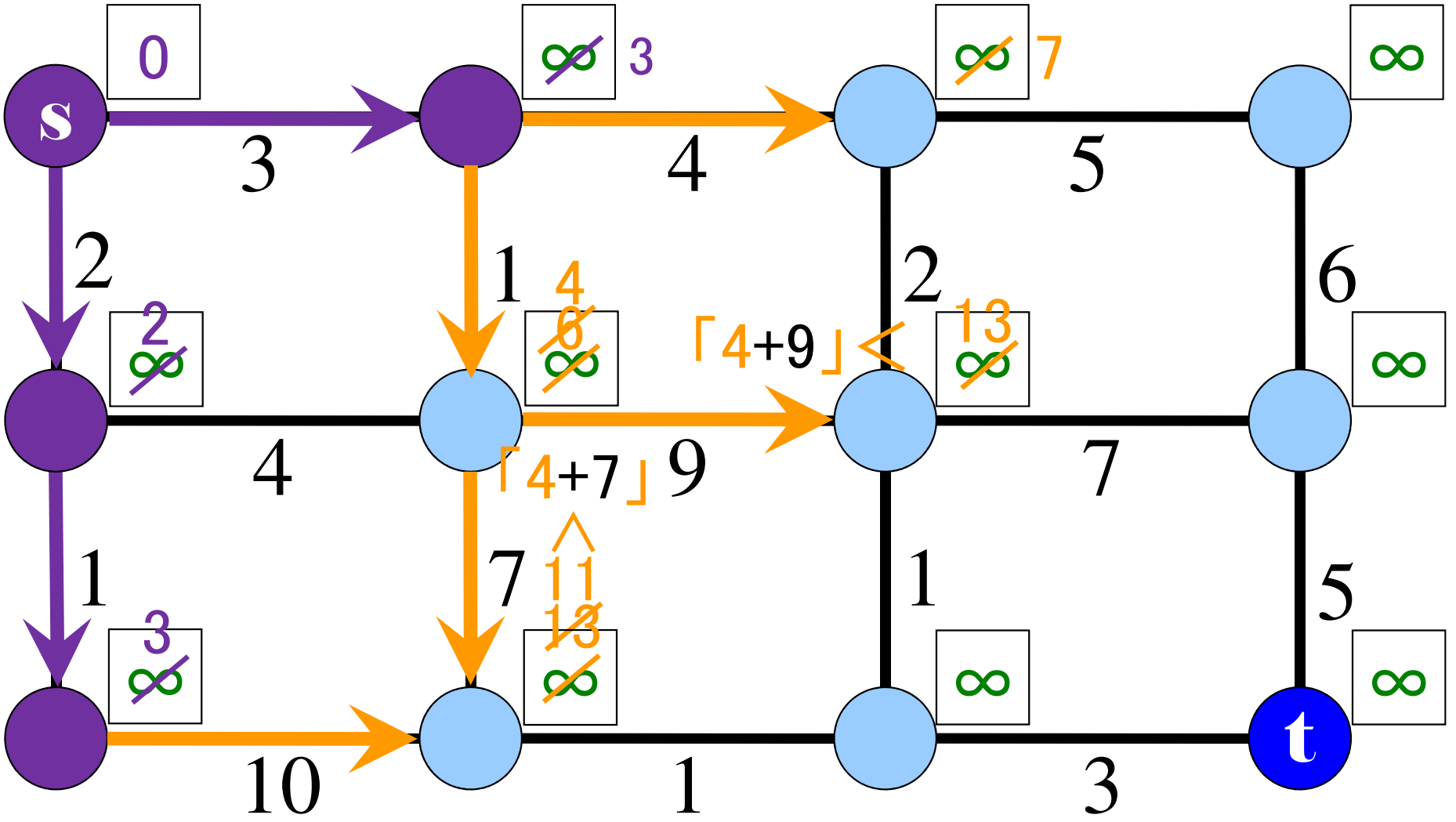
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

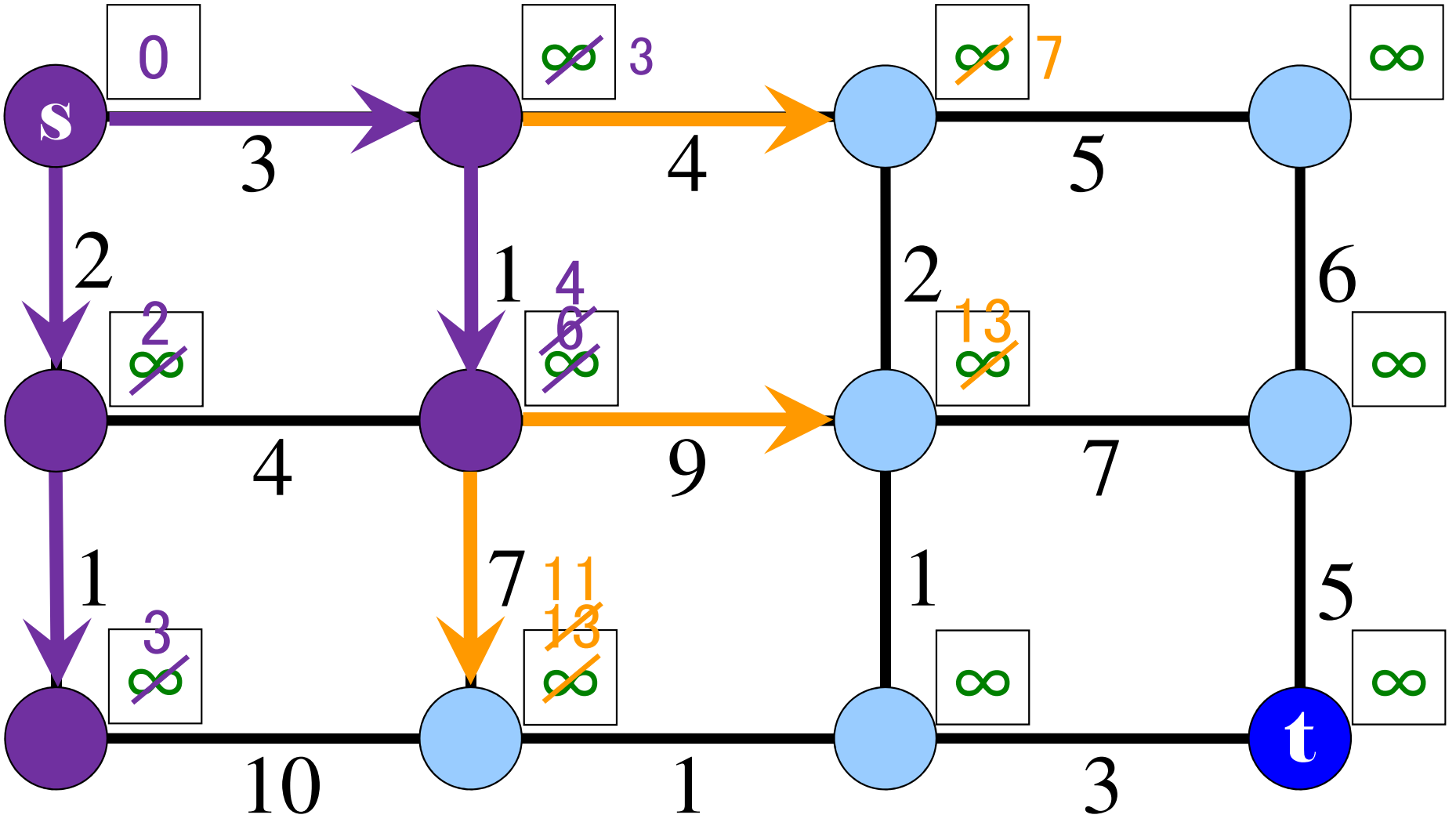


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

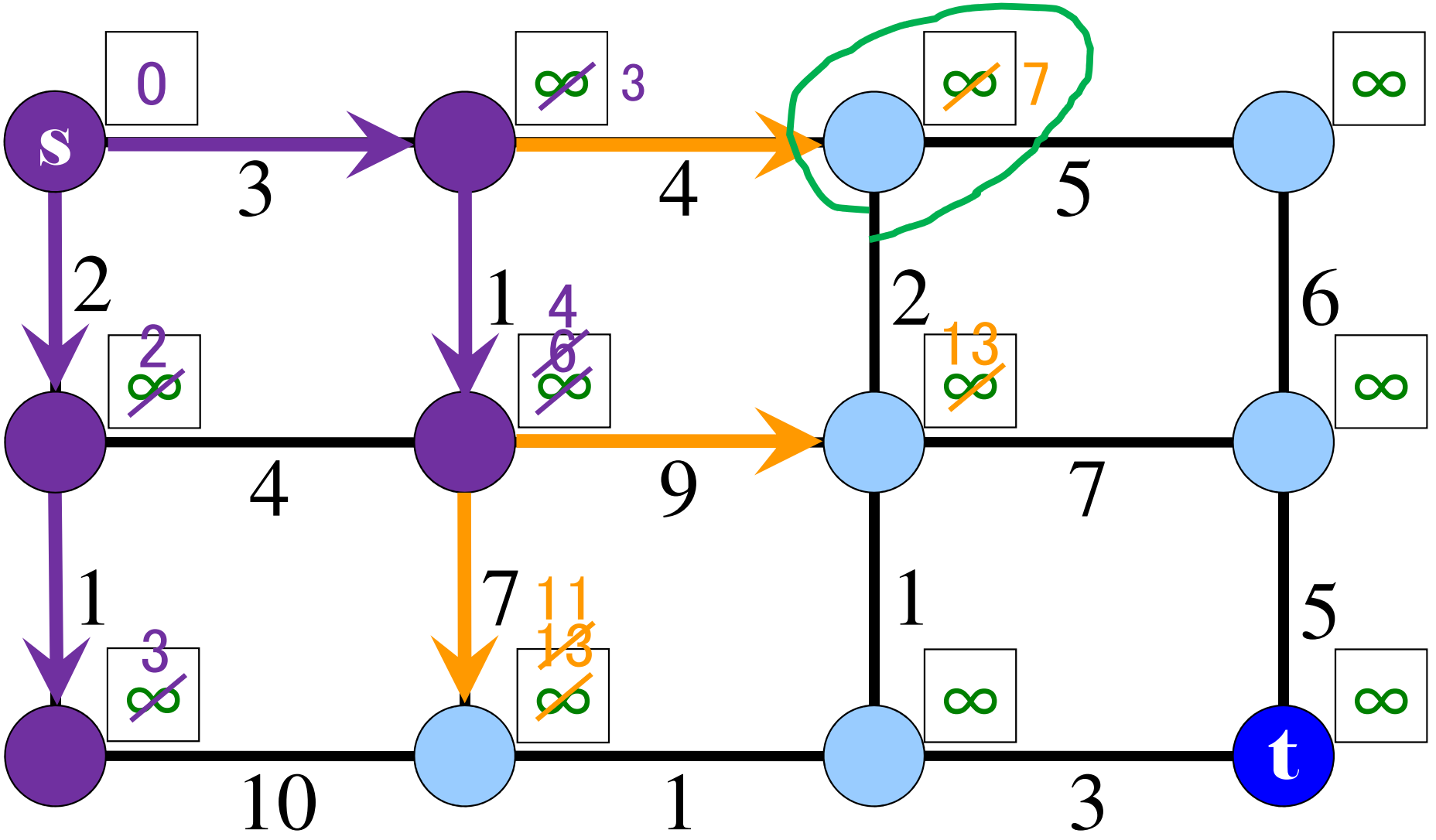
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



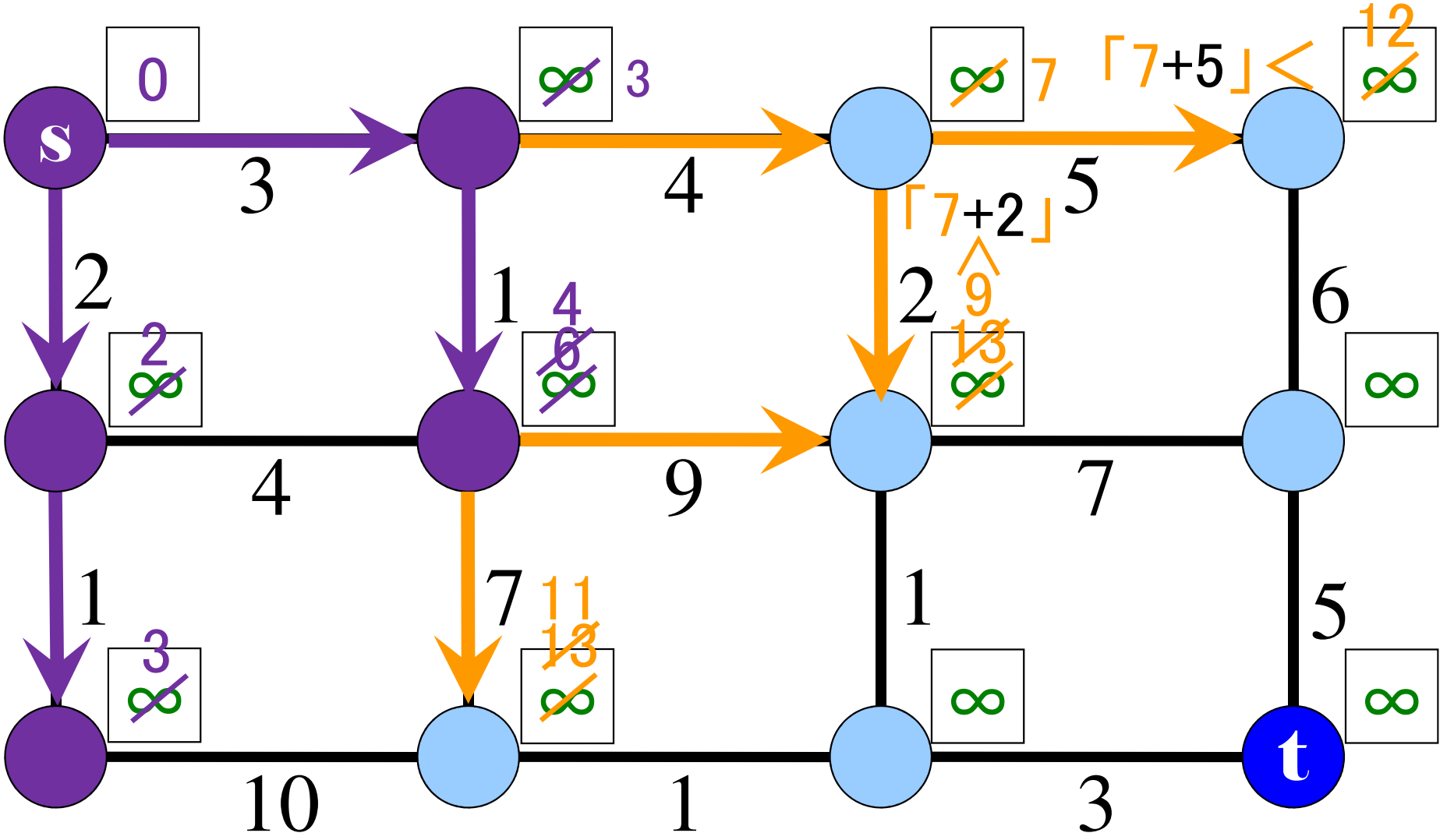
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

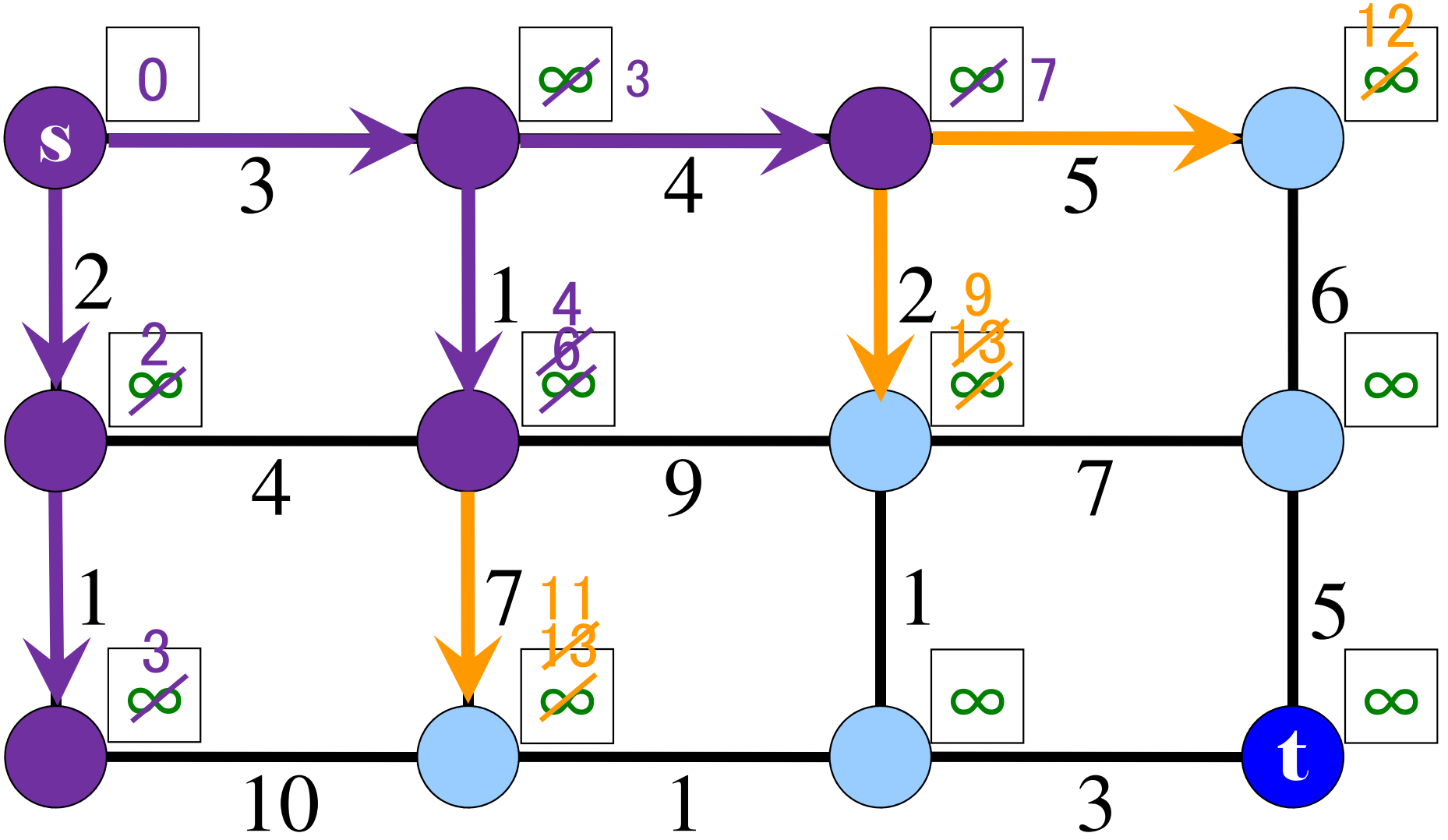


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

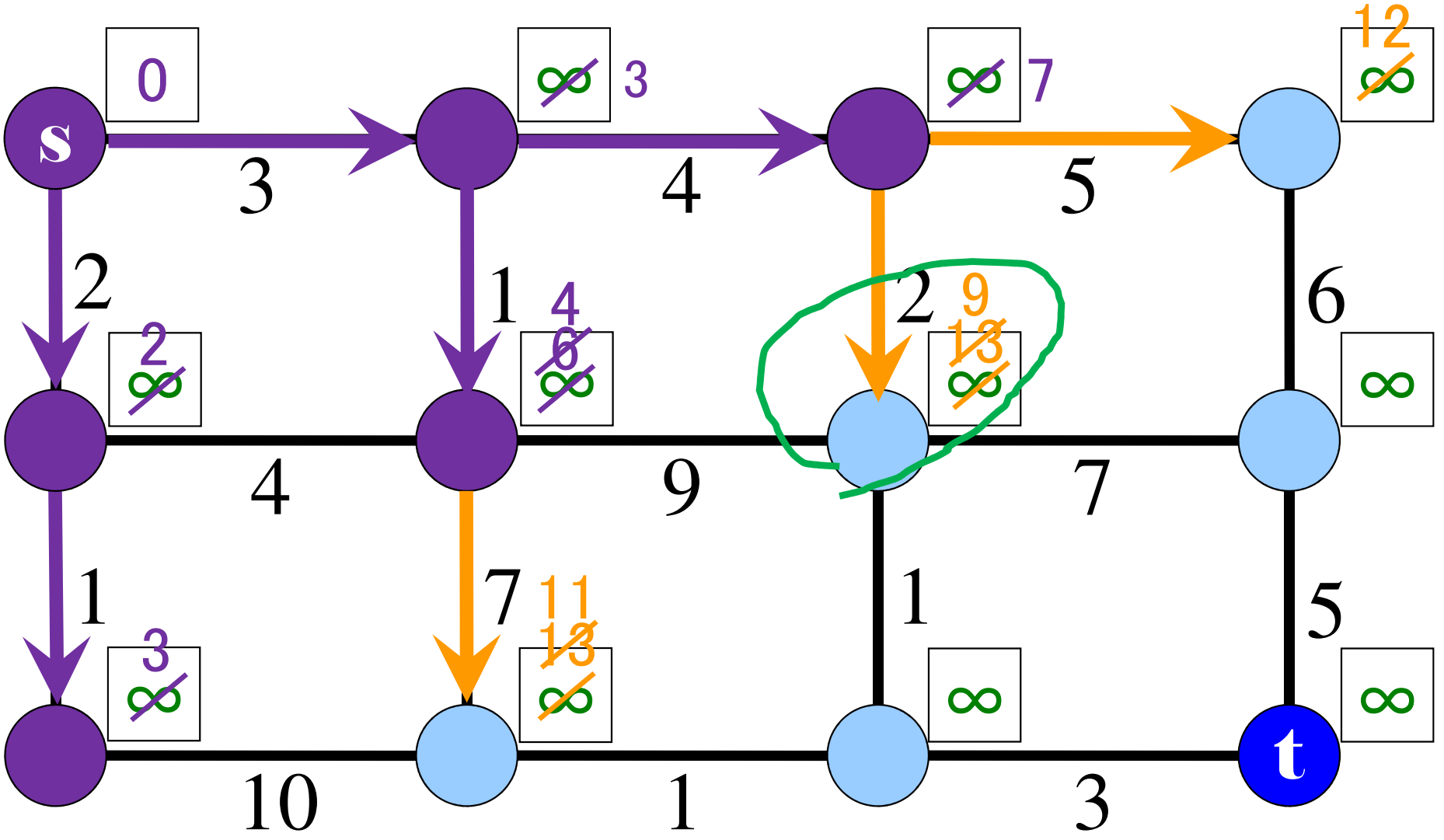
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



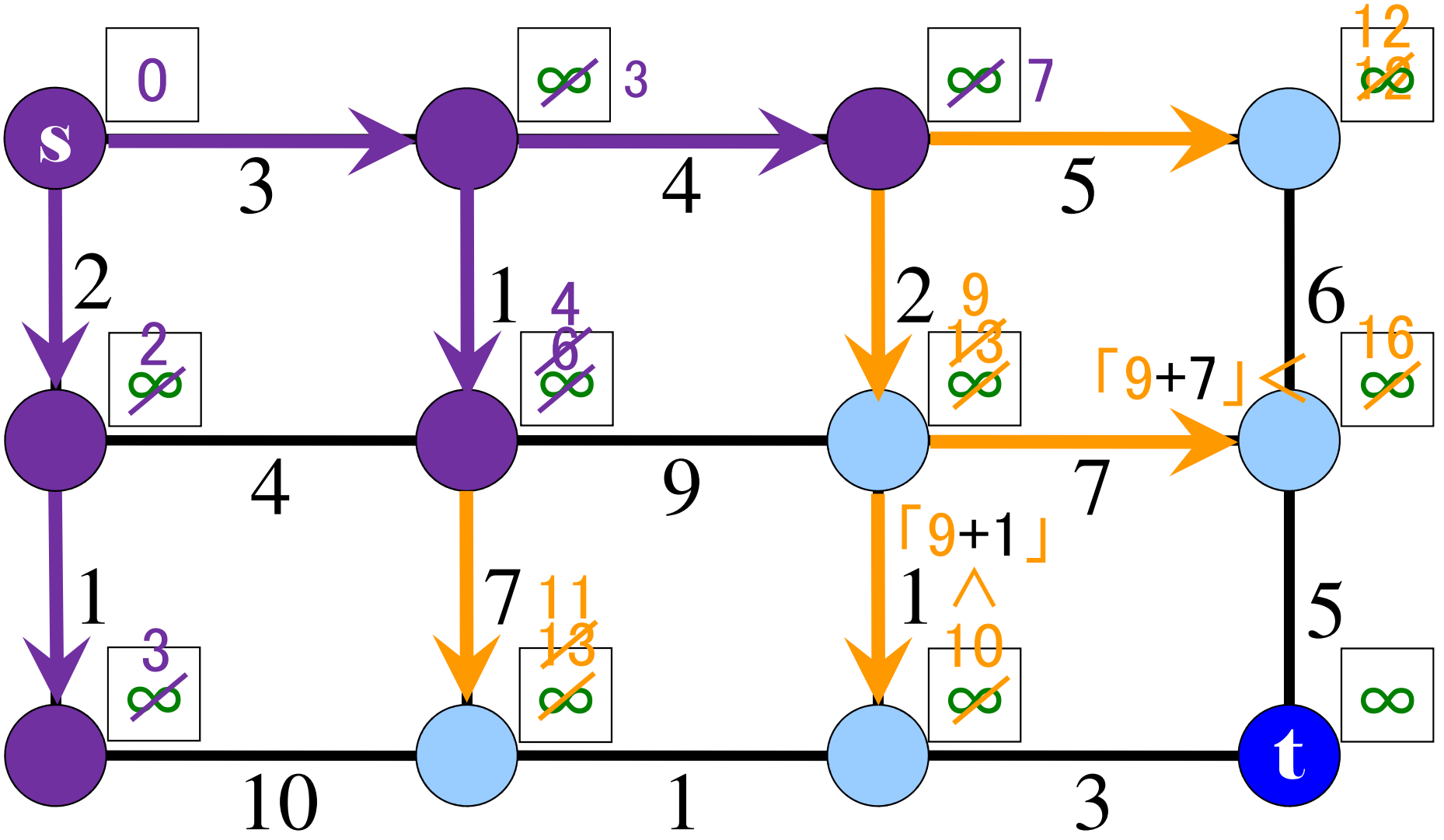
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

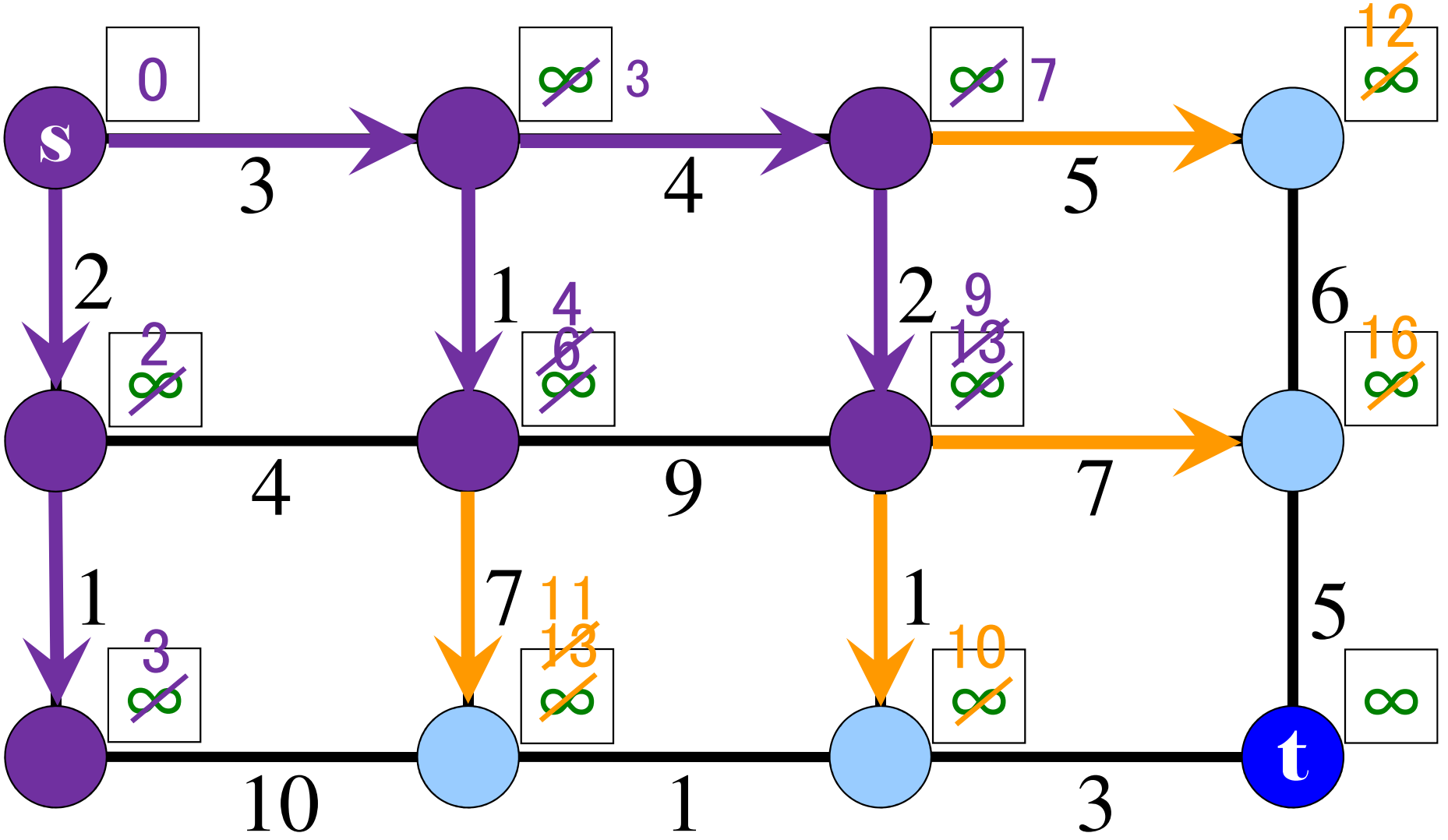


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

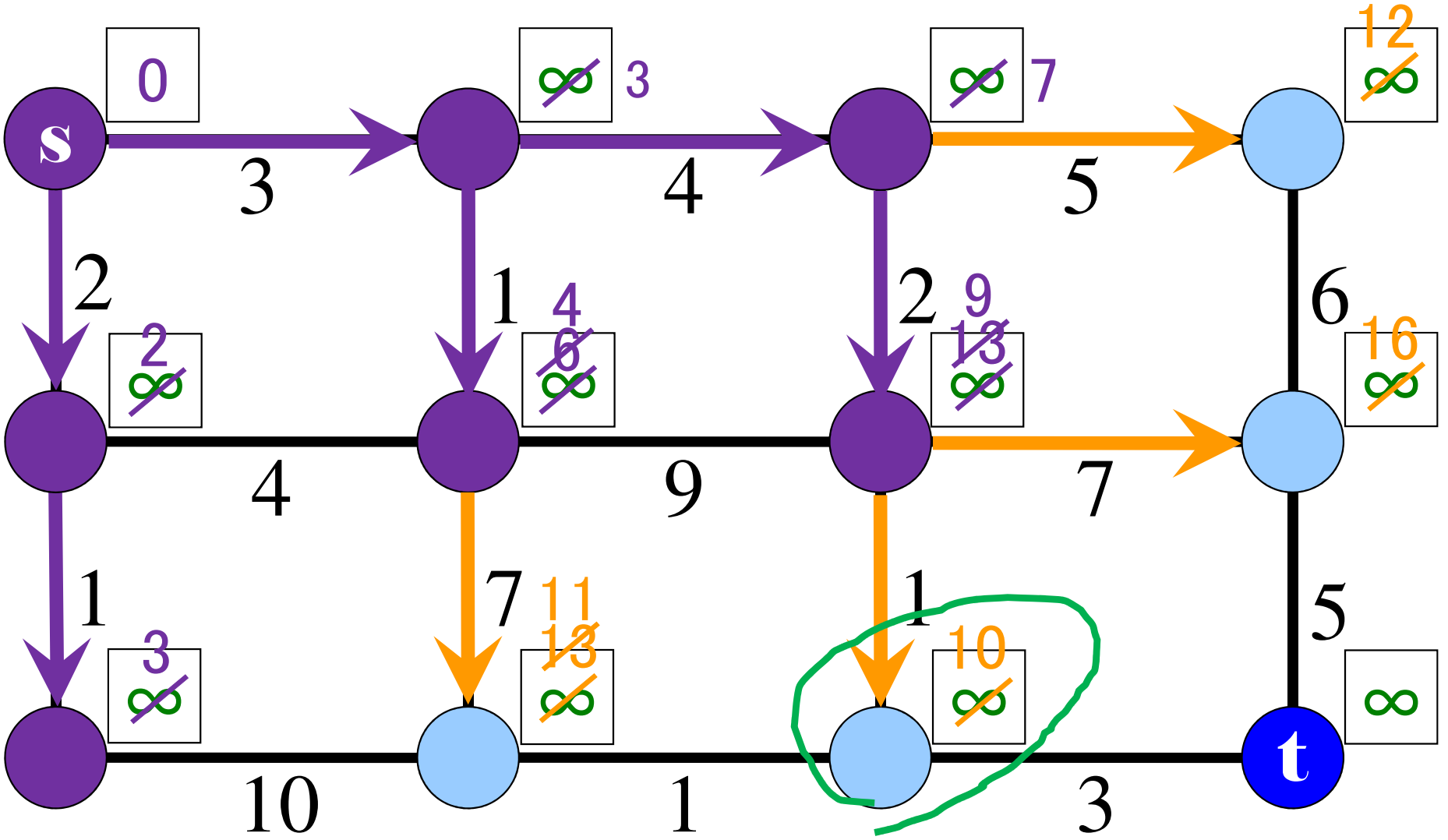
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



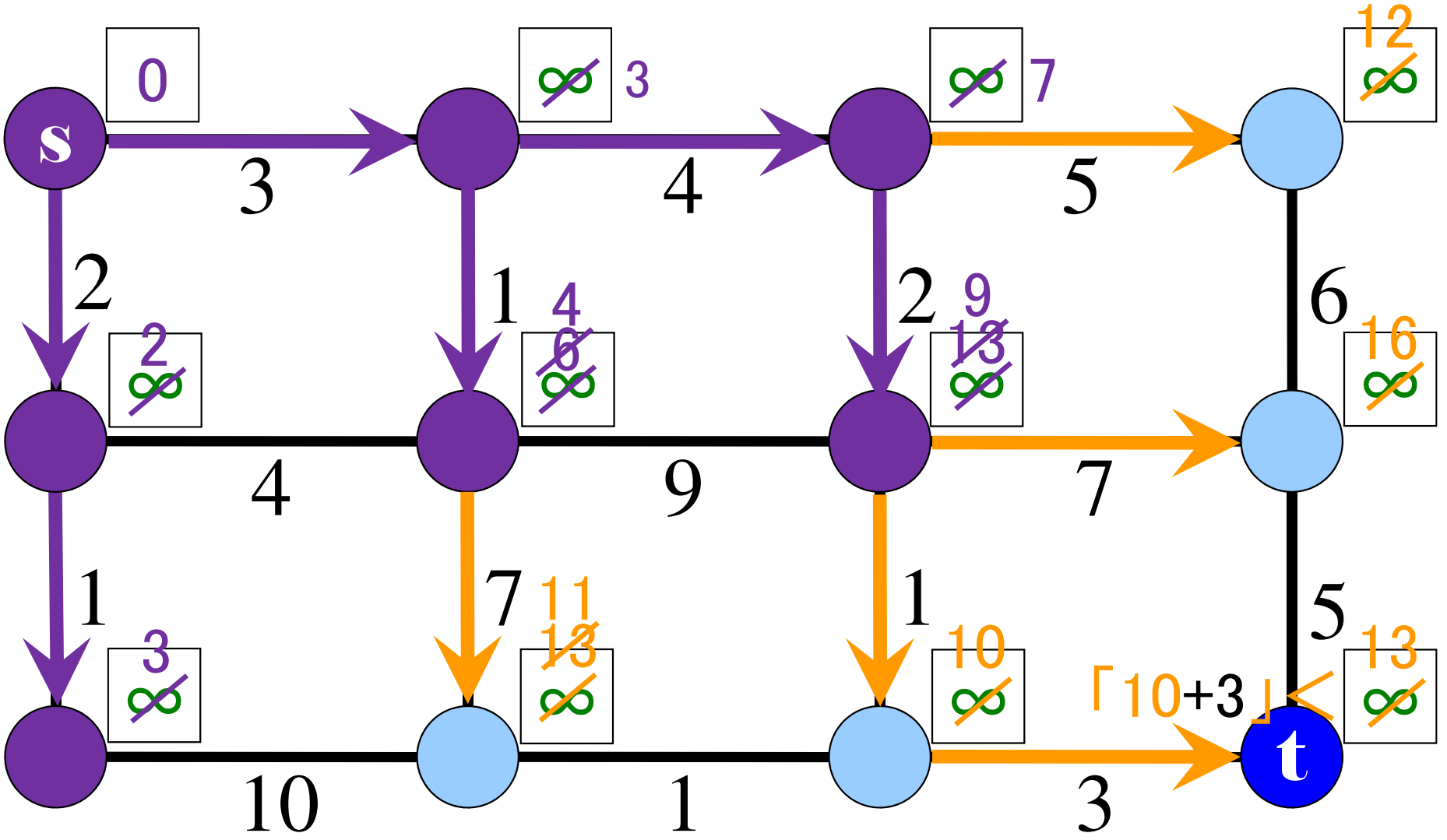
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

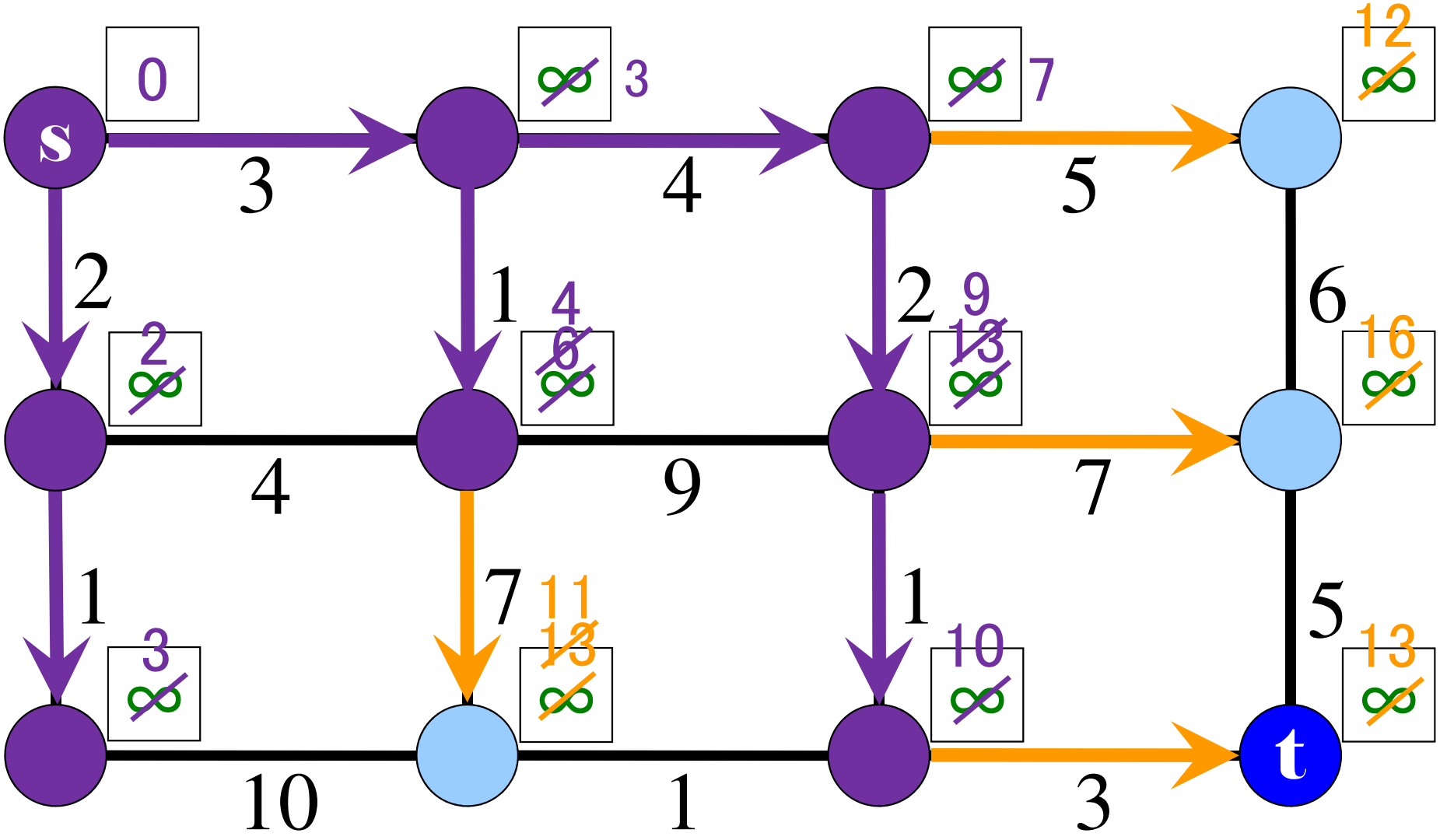


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

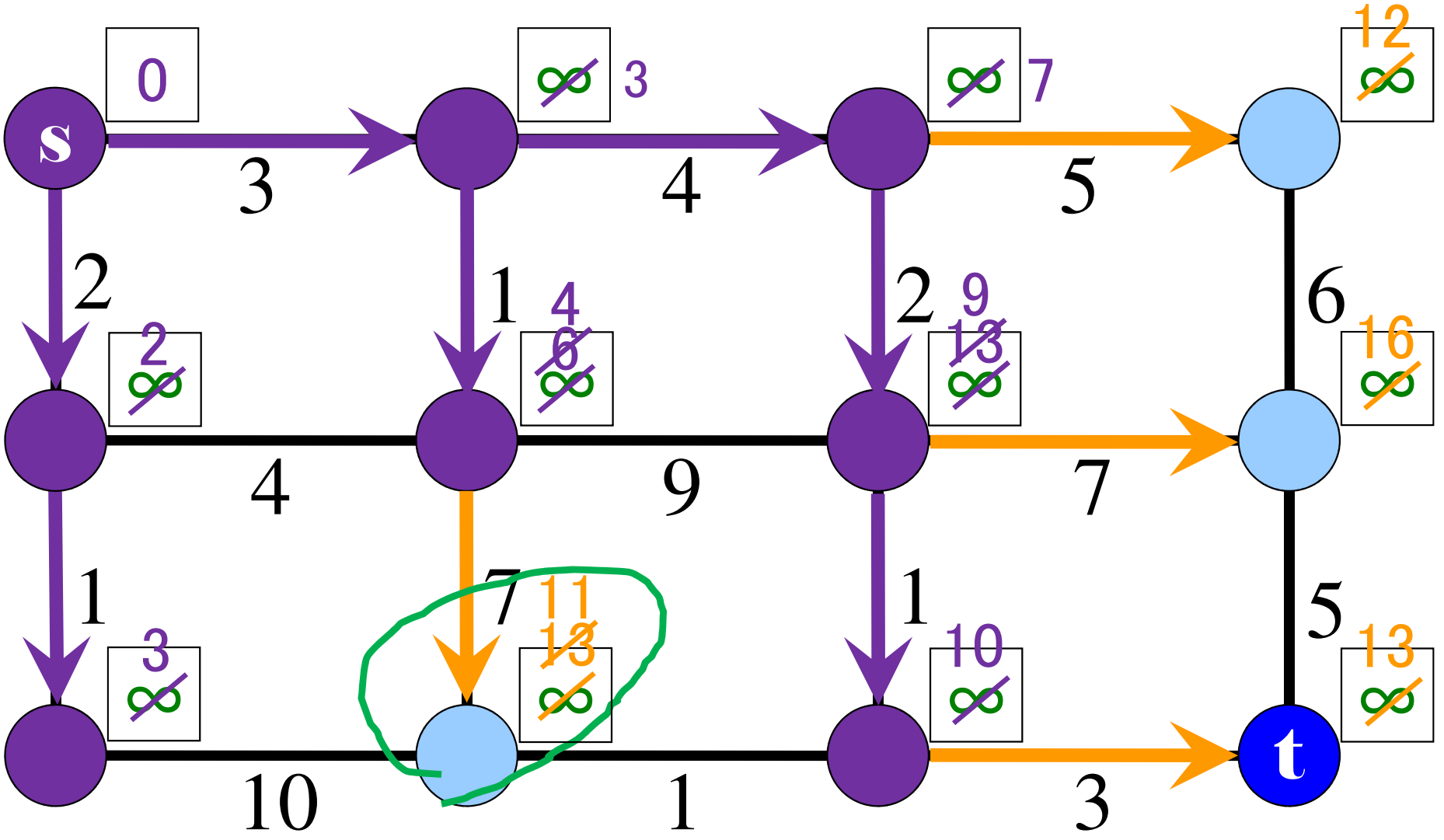
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



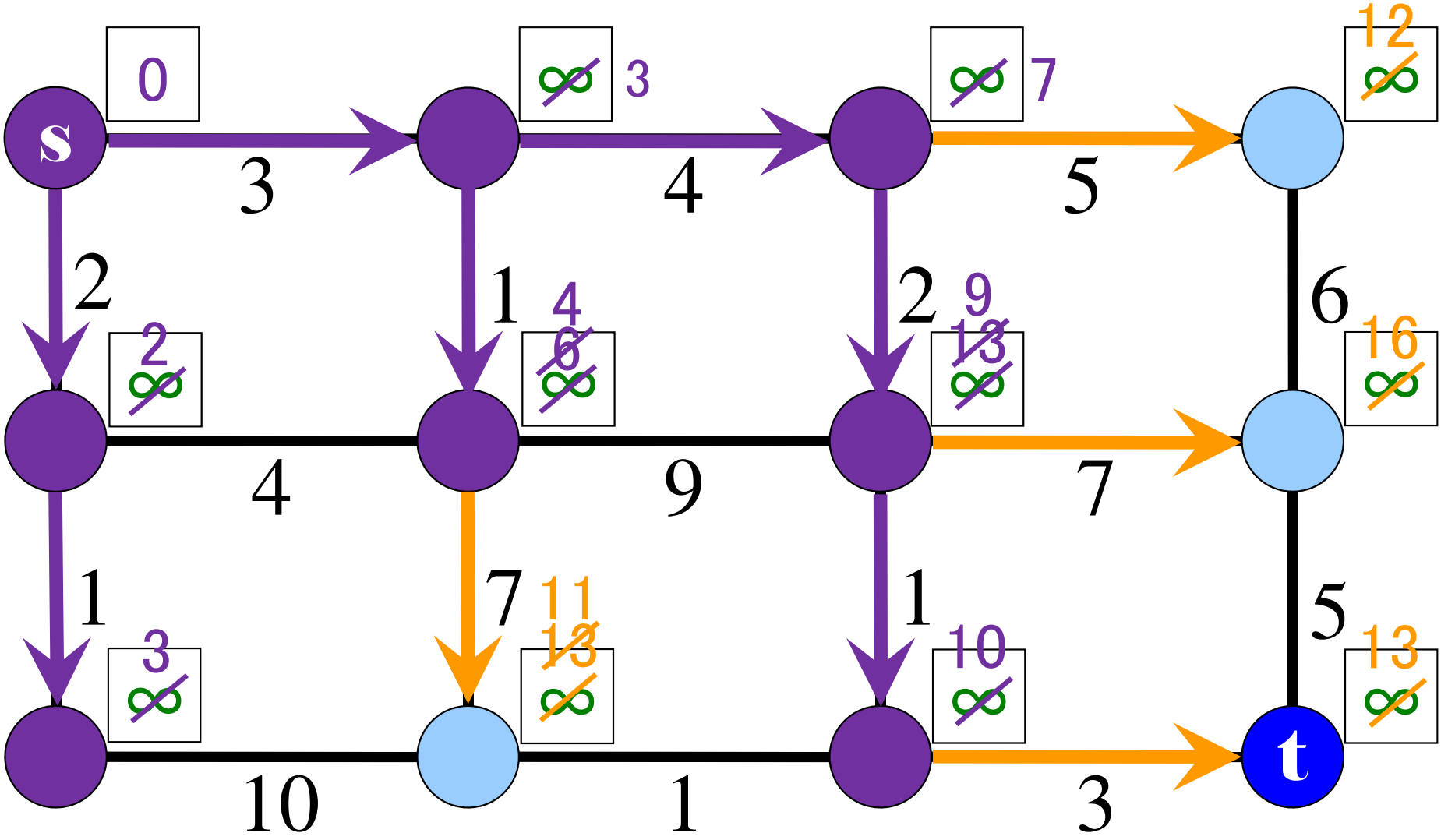
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

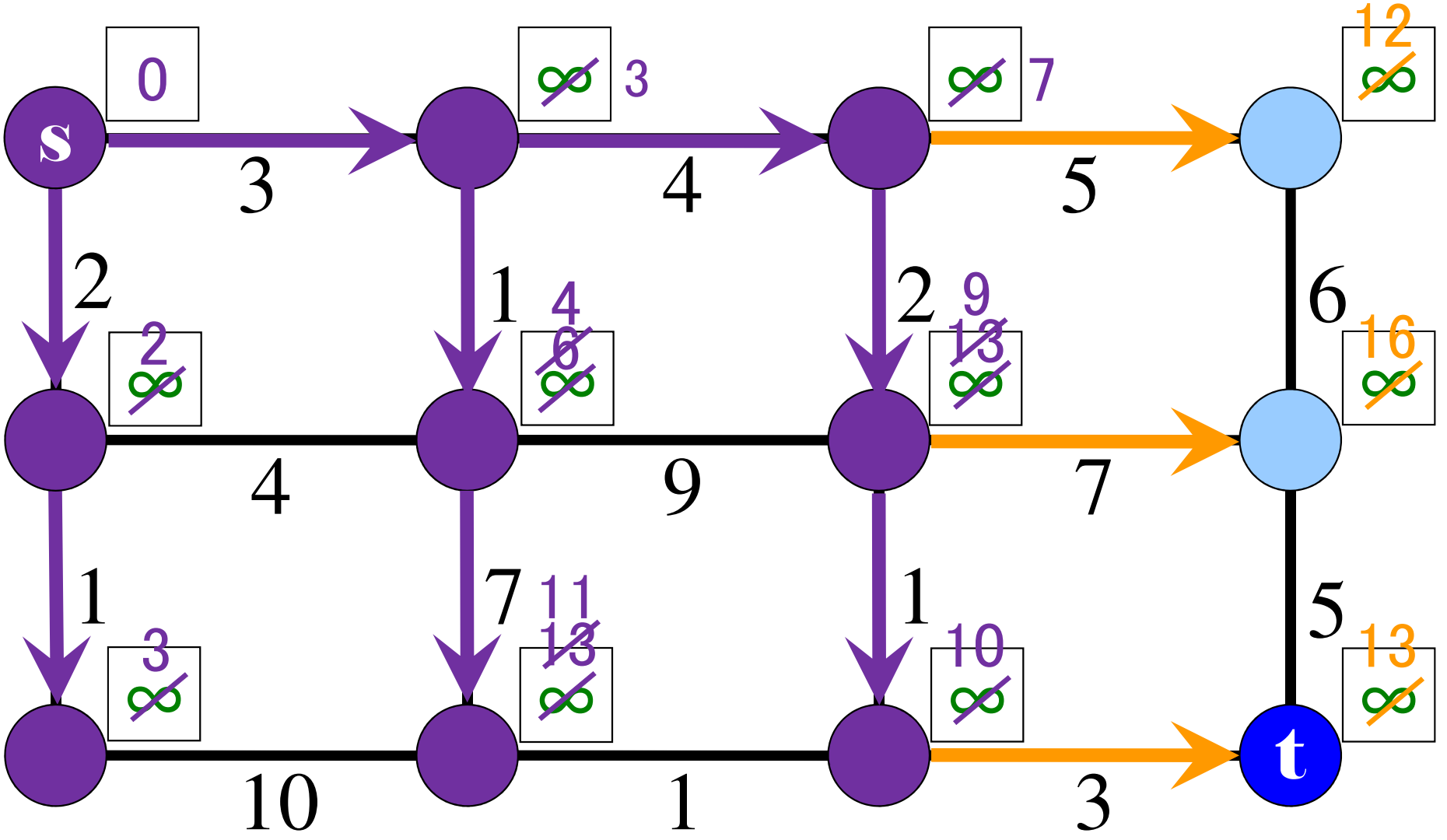


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

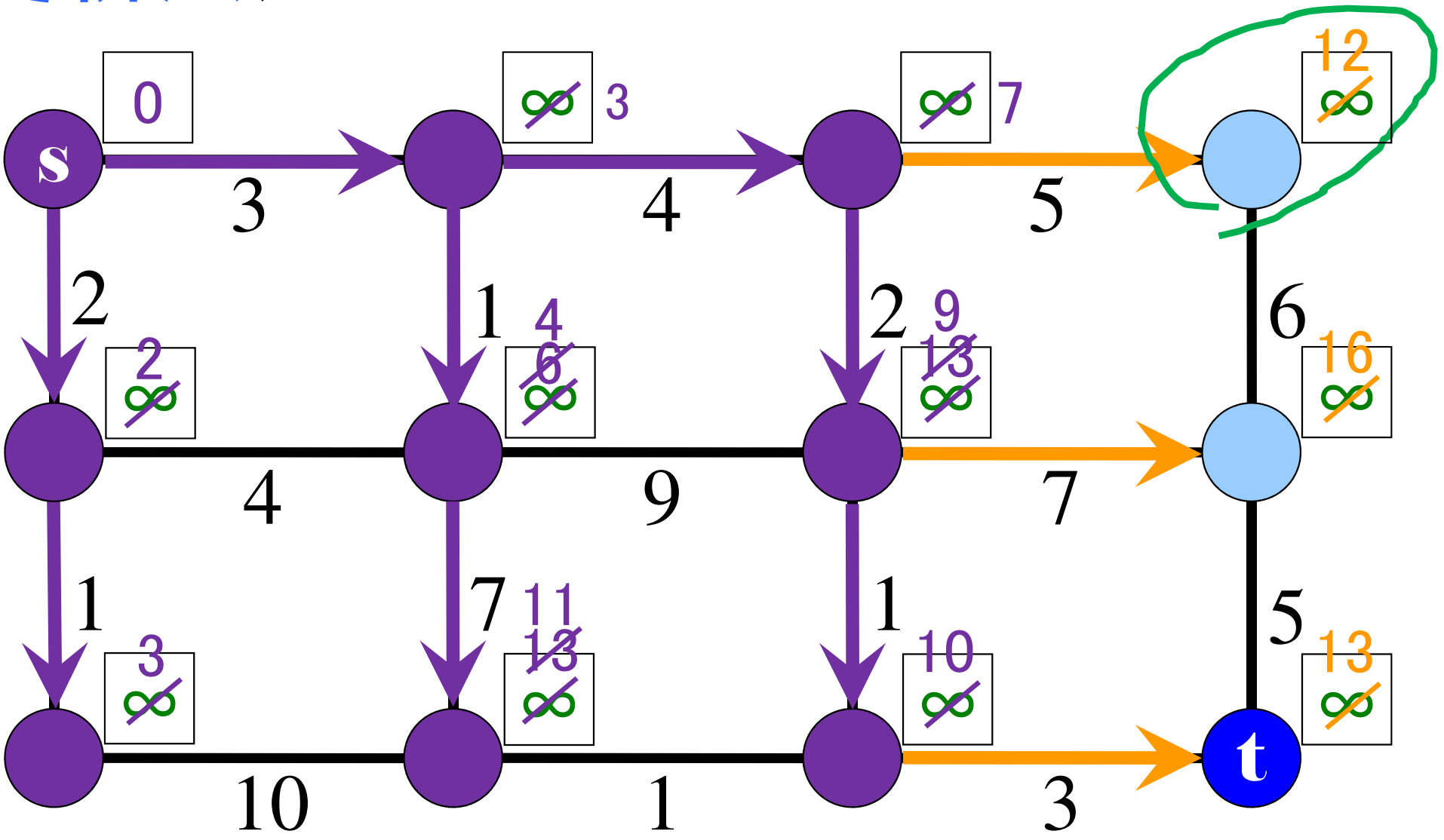
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



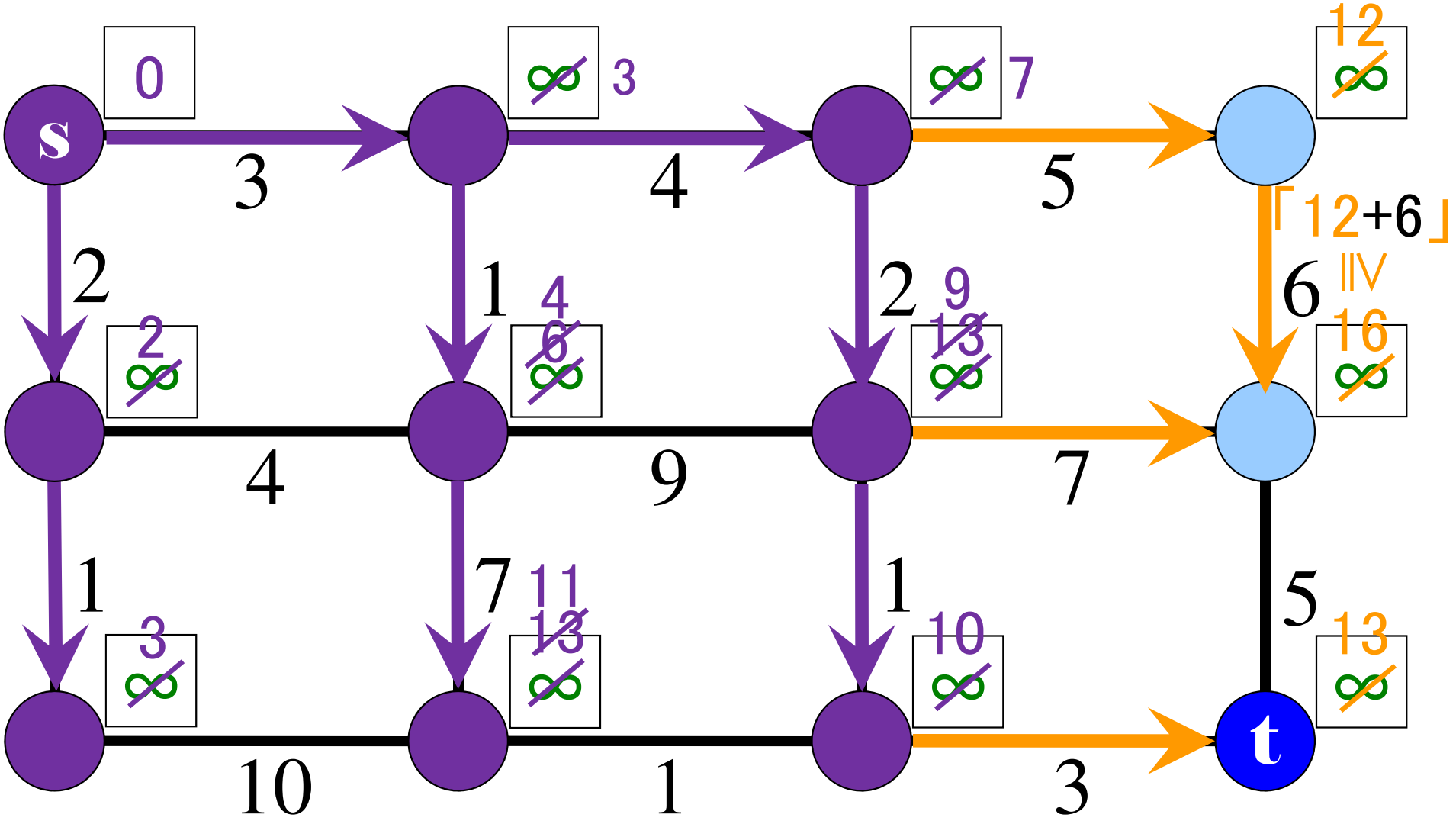
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

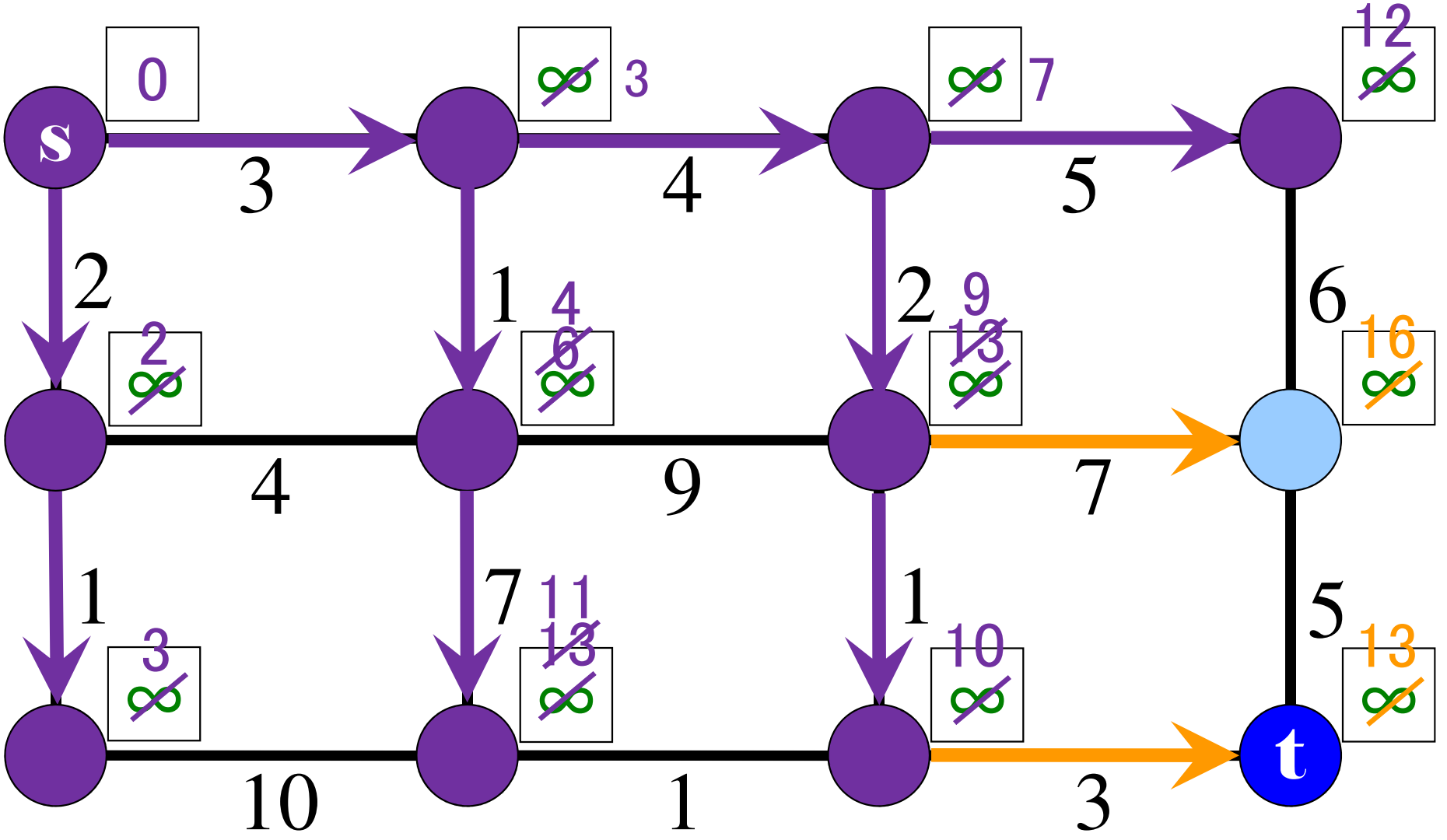


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

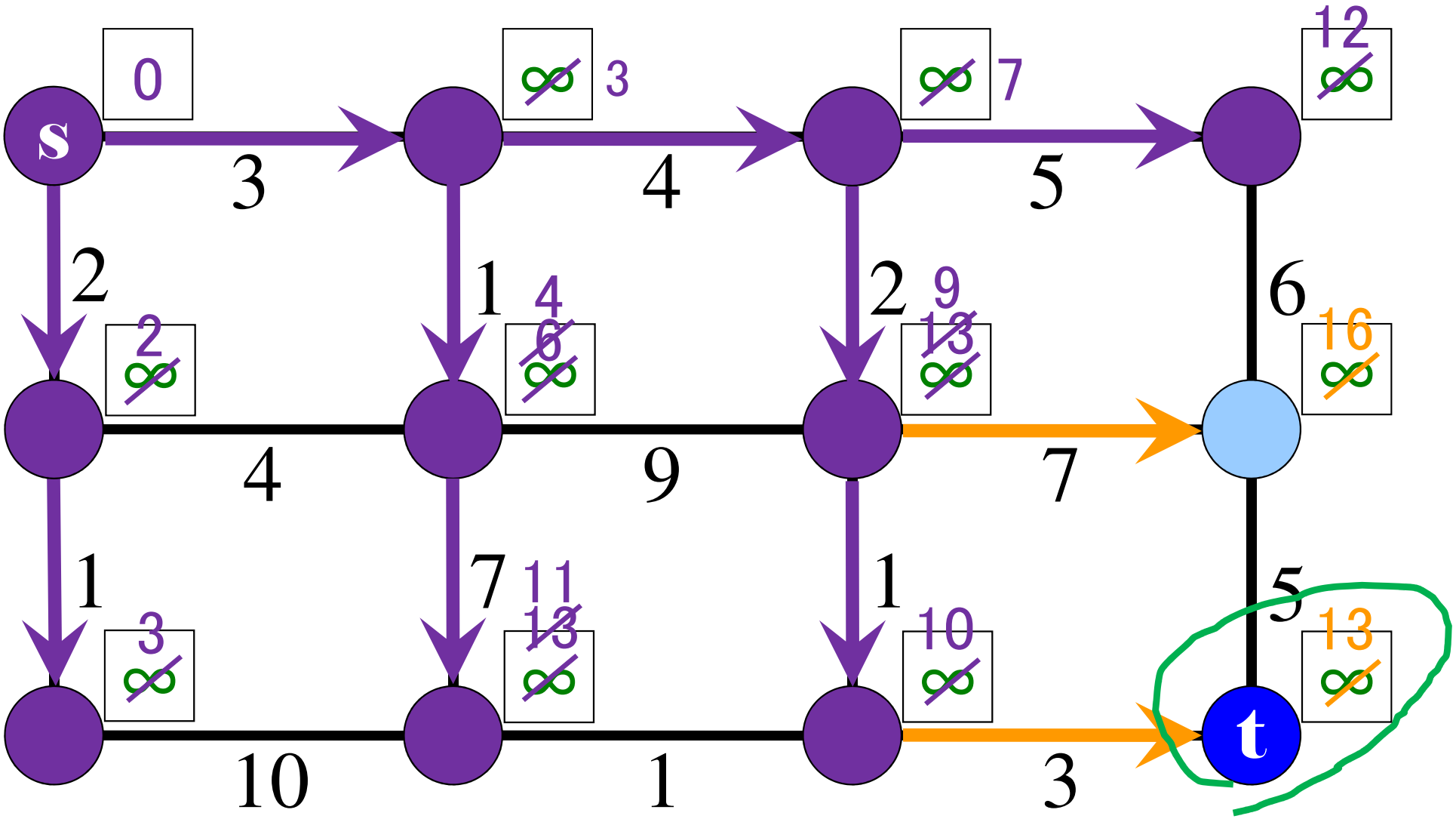
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



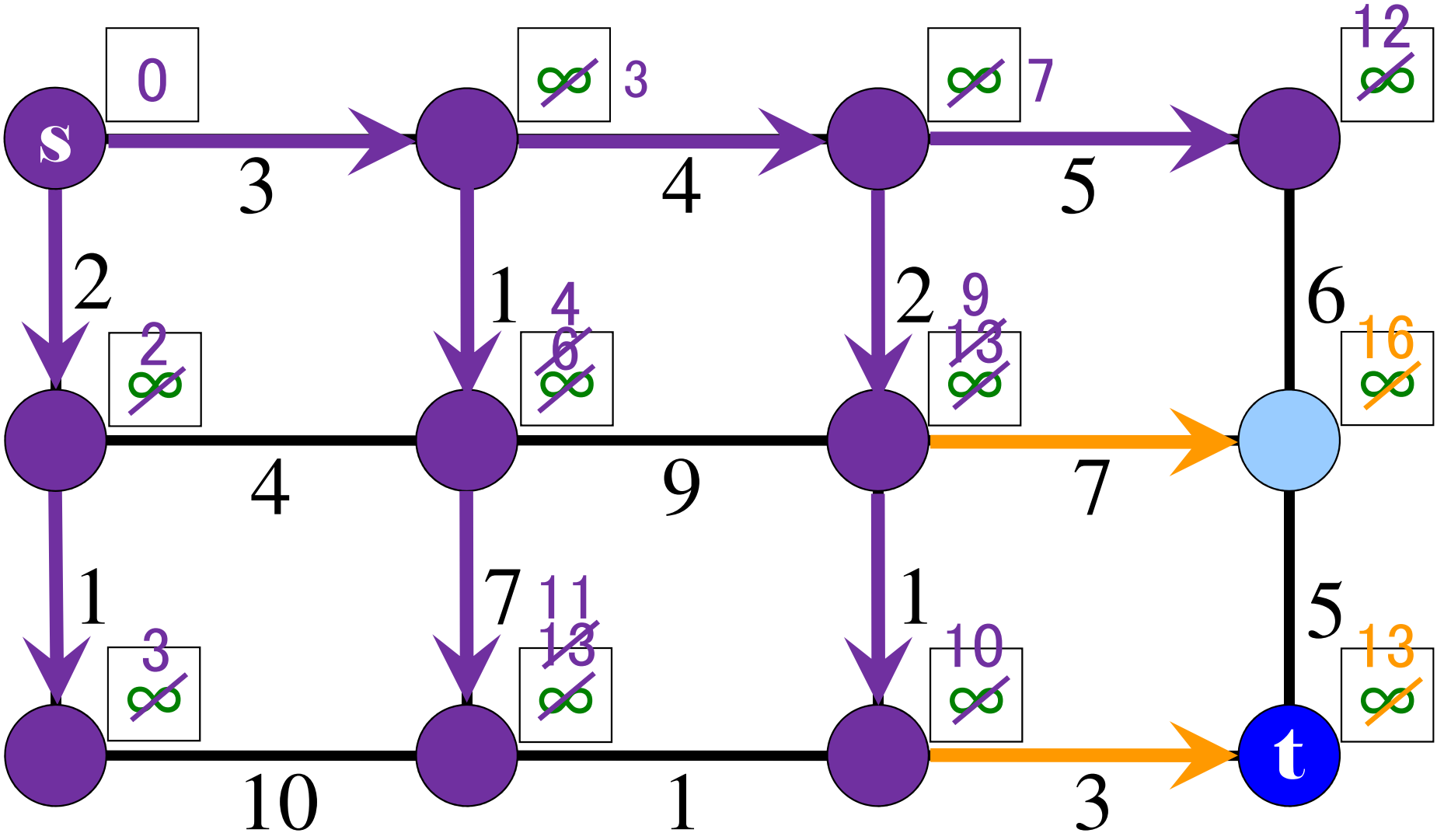
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

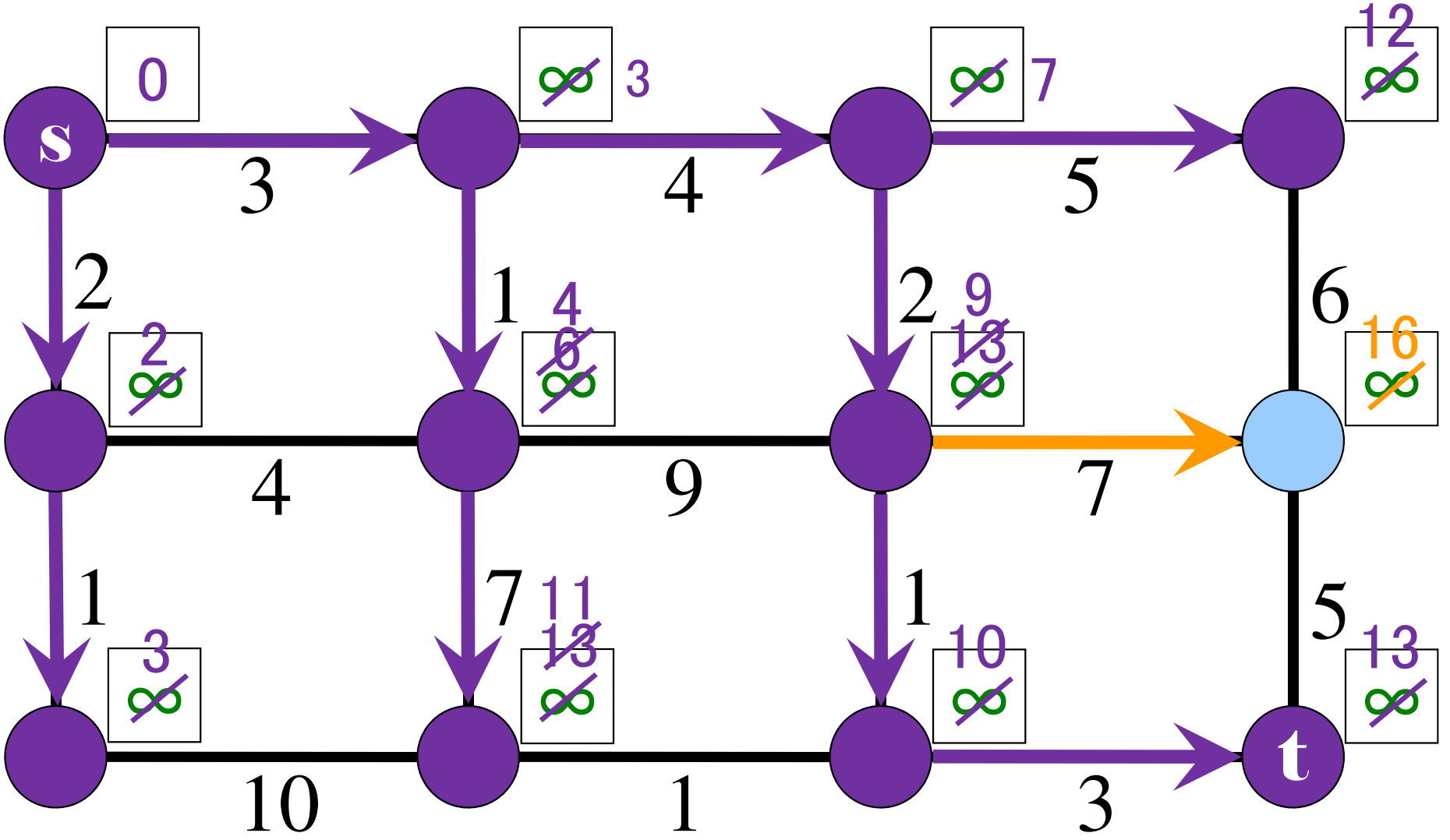


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

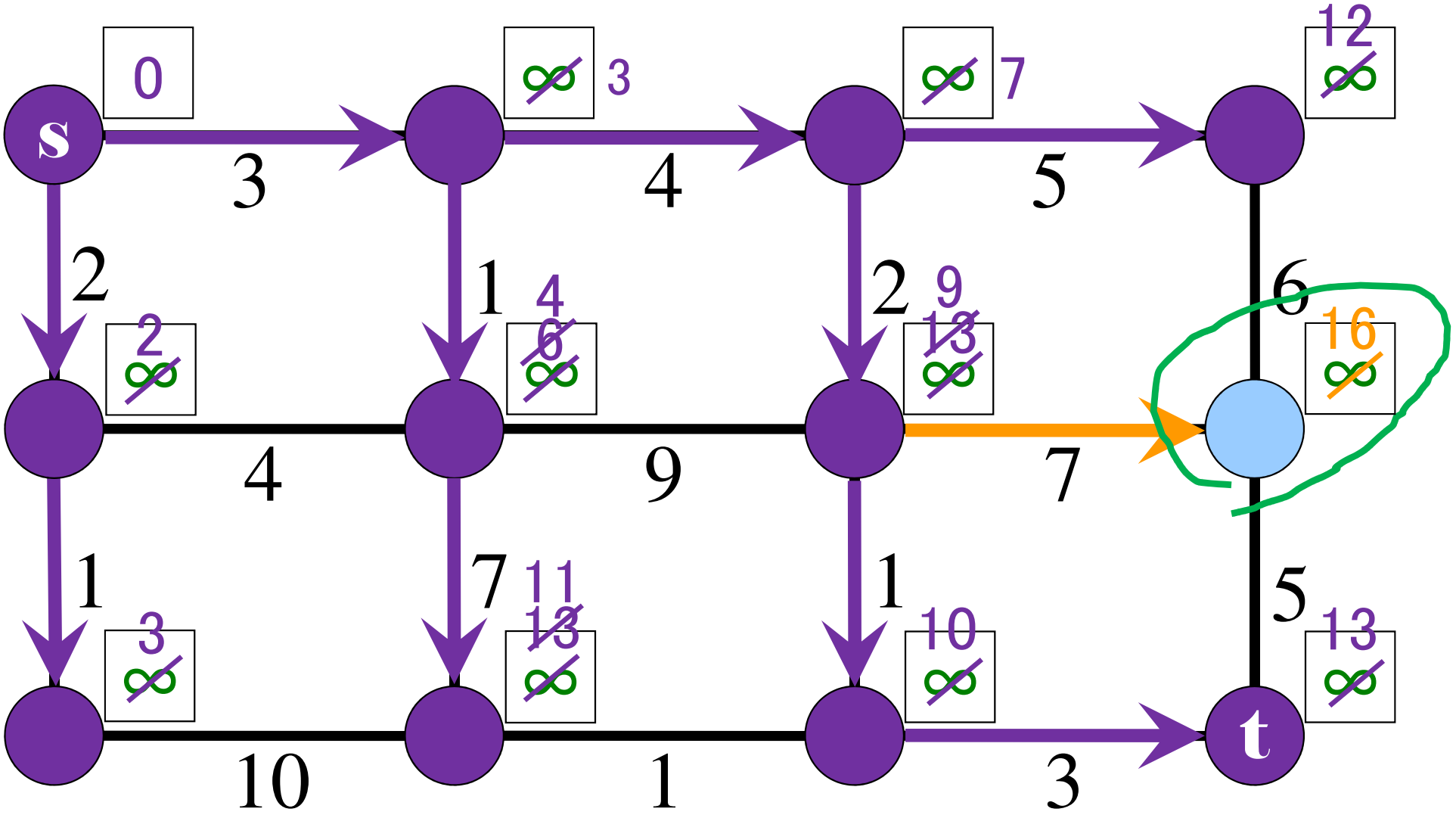
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



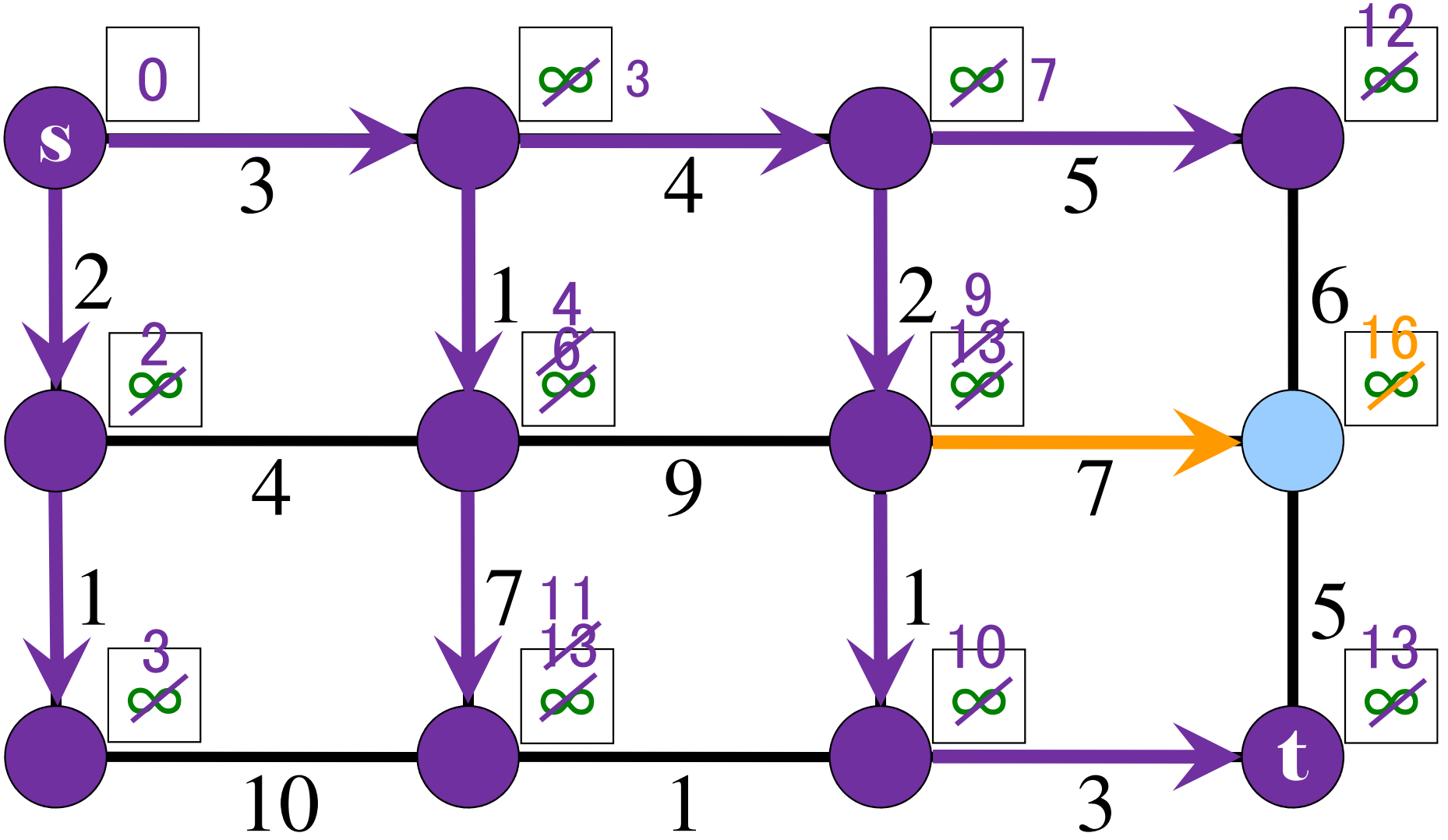
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

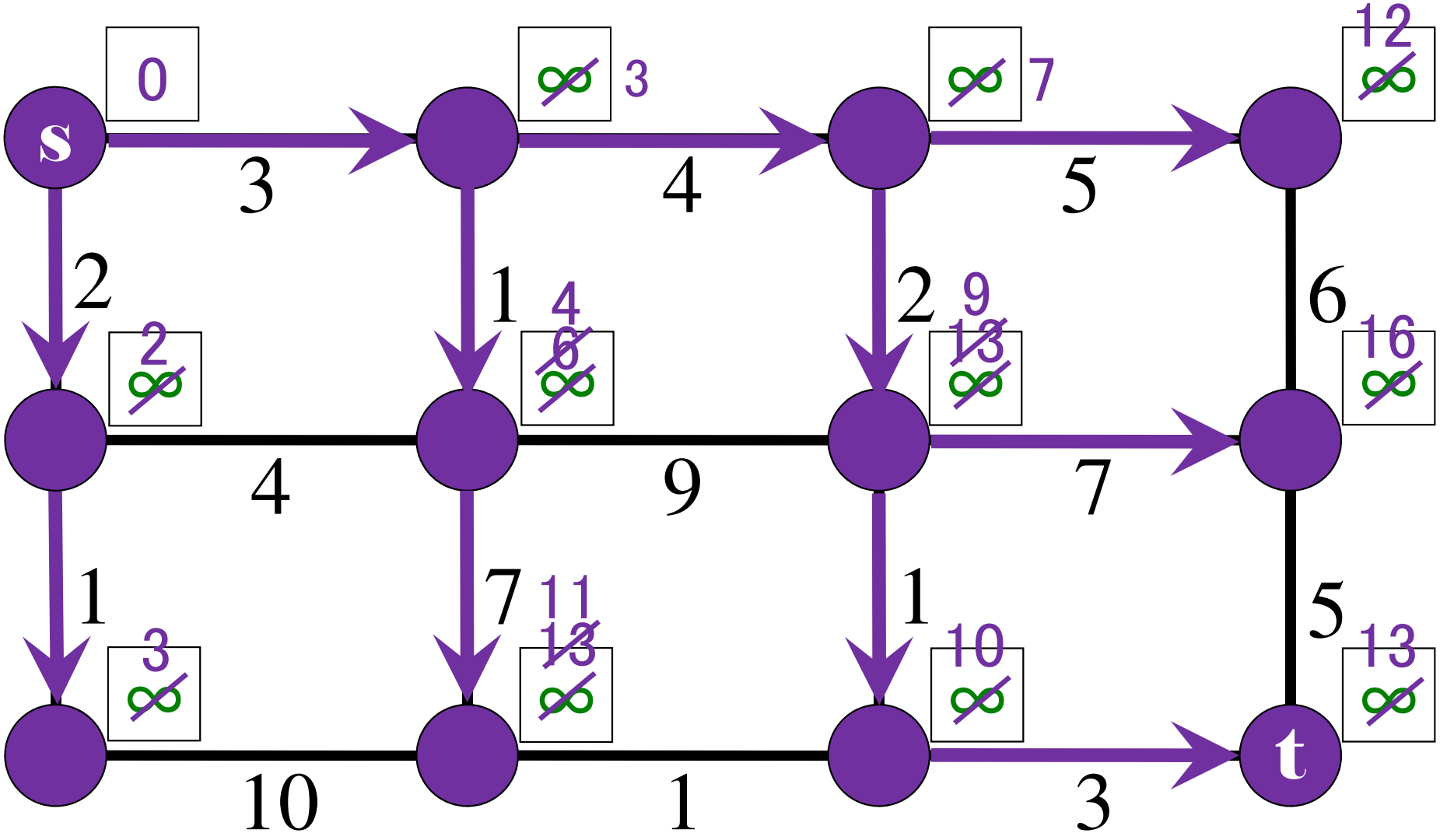


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

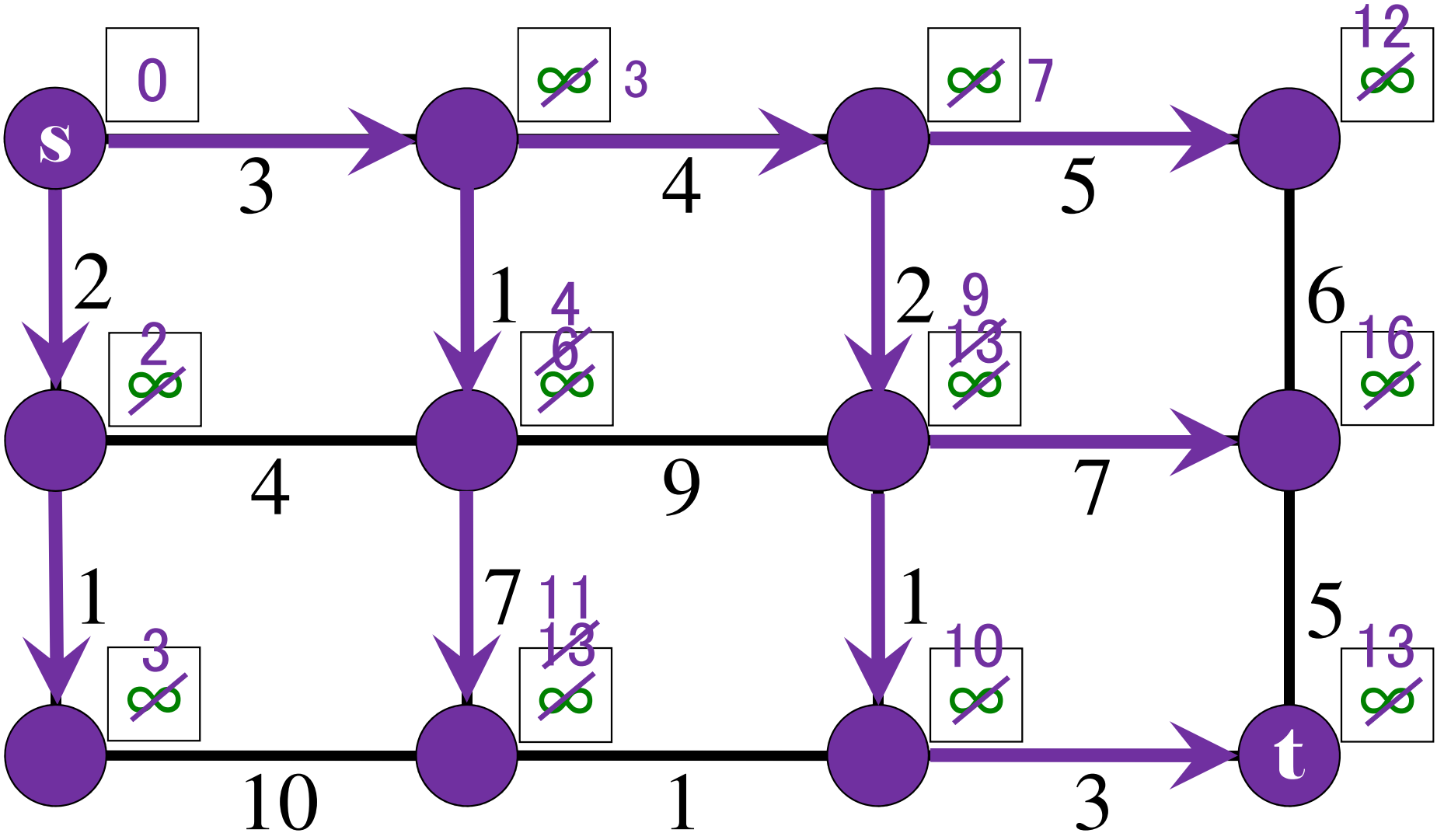
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



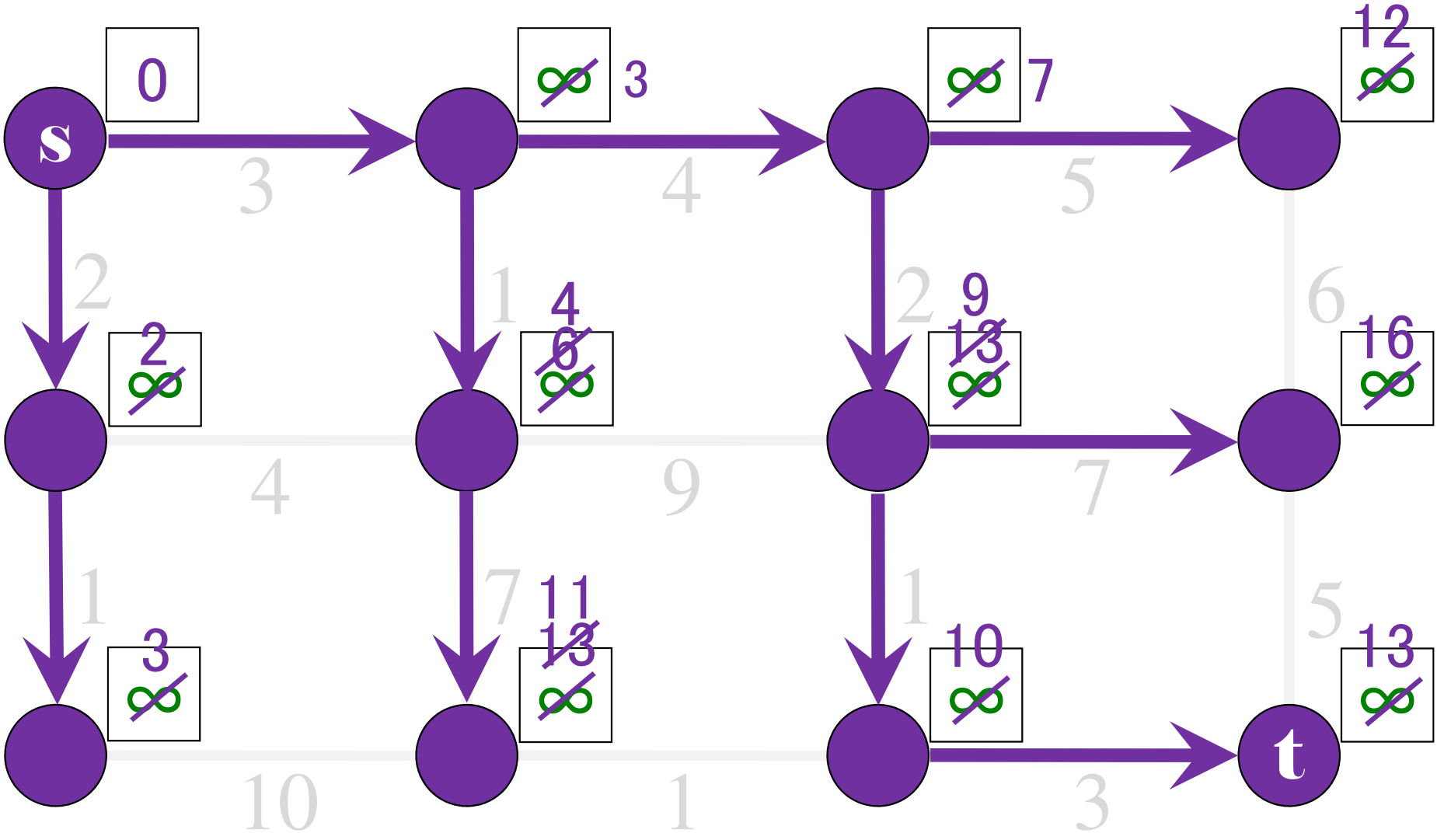
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



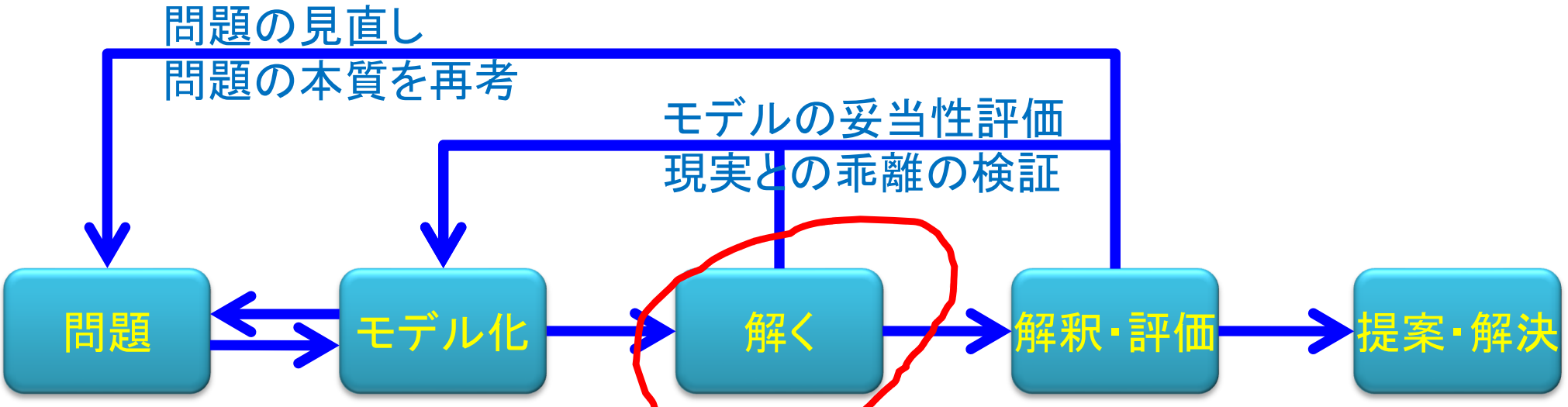
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

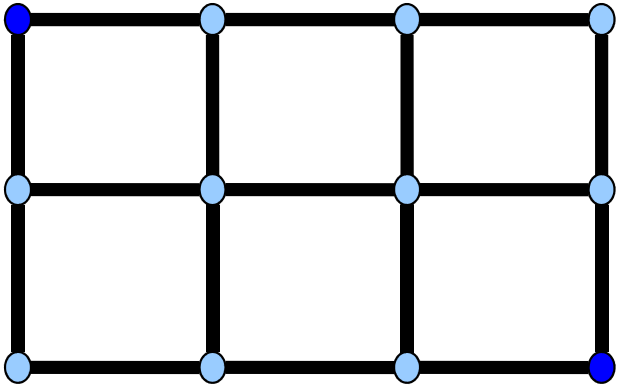
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

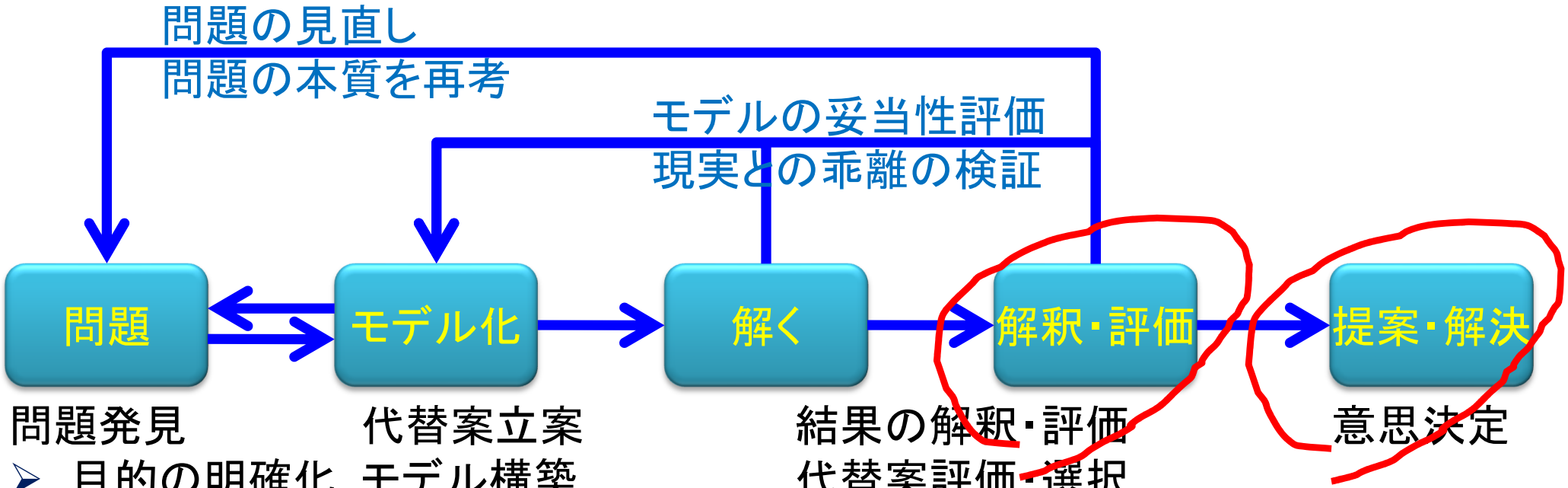
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

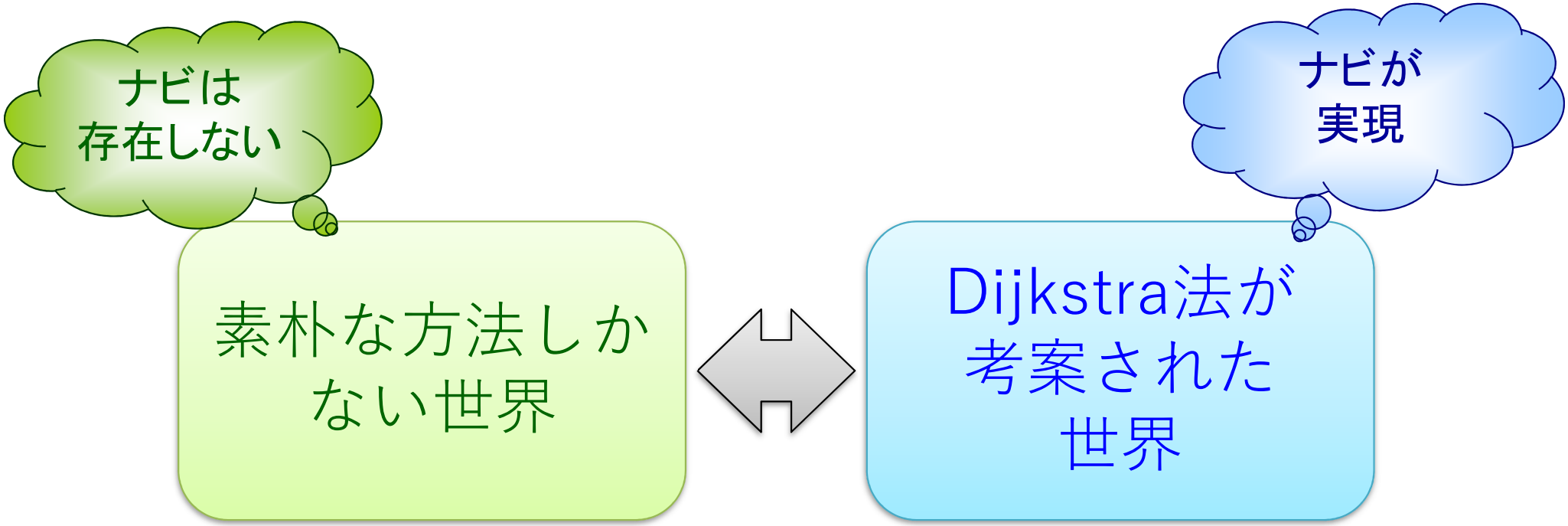
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

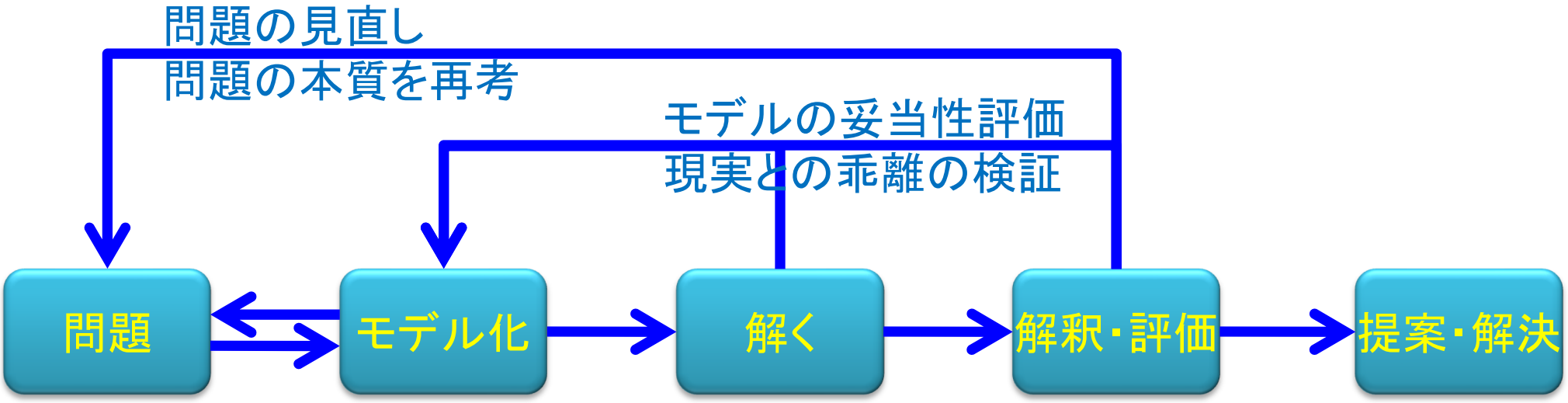
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



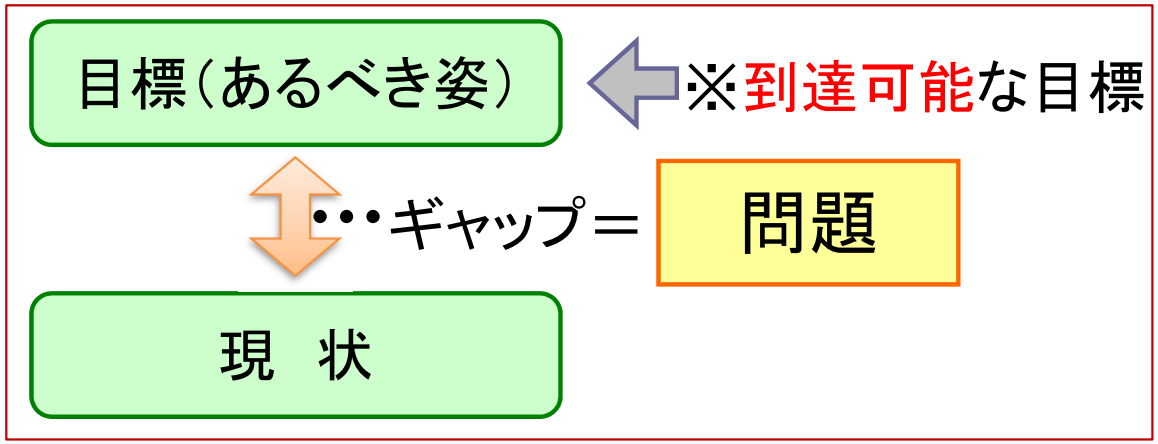
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

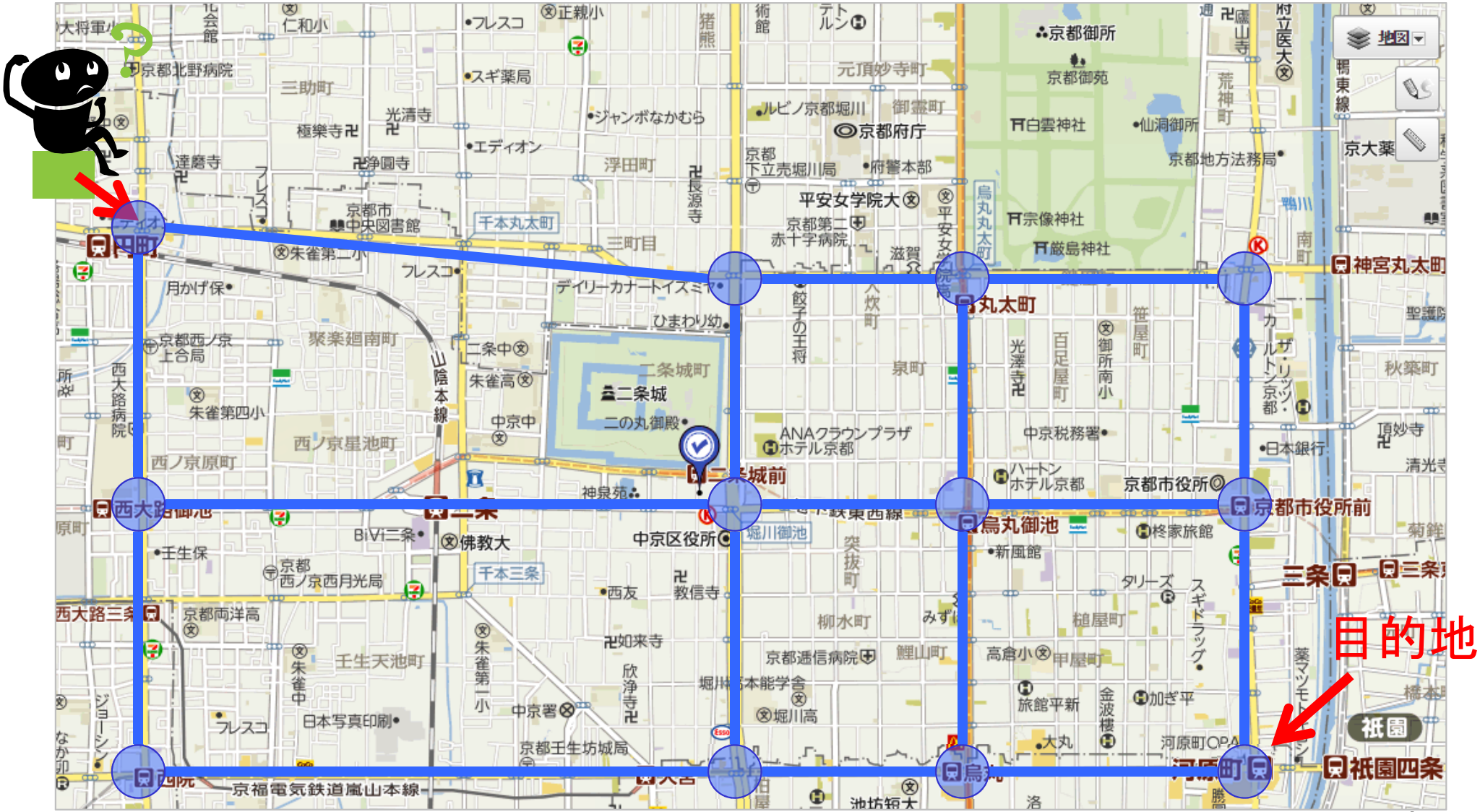
意思決定

問題の定義



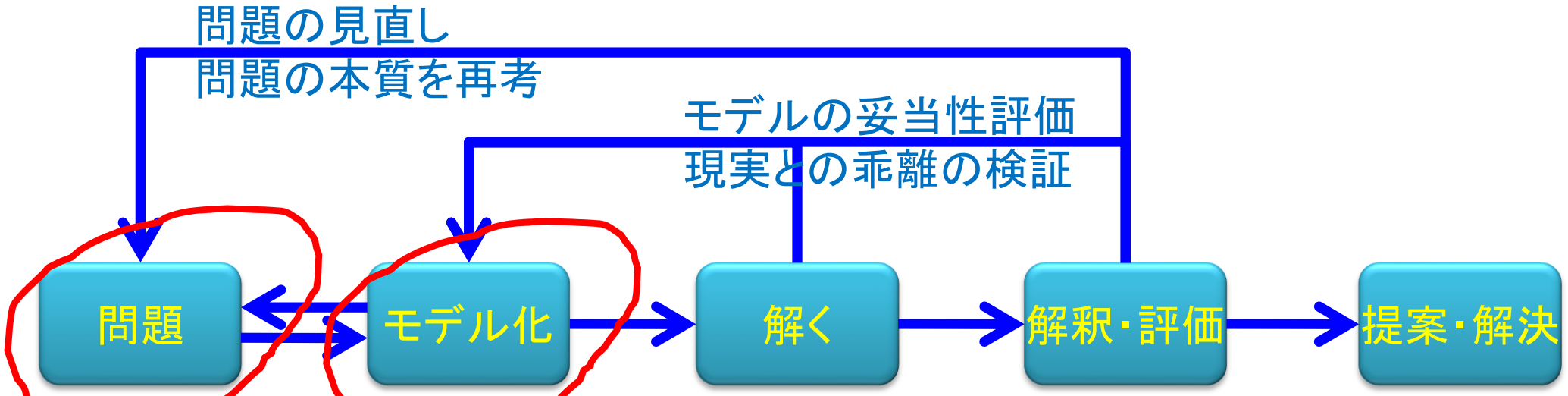
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

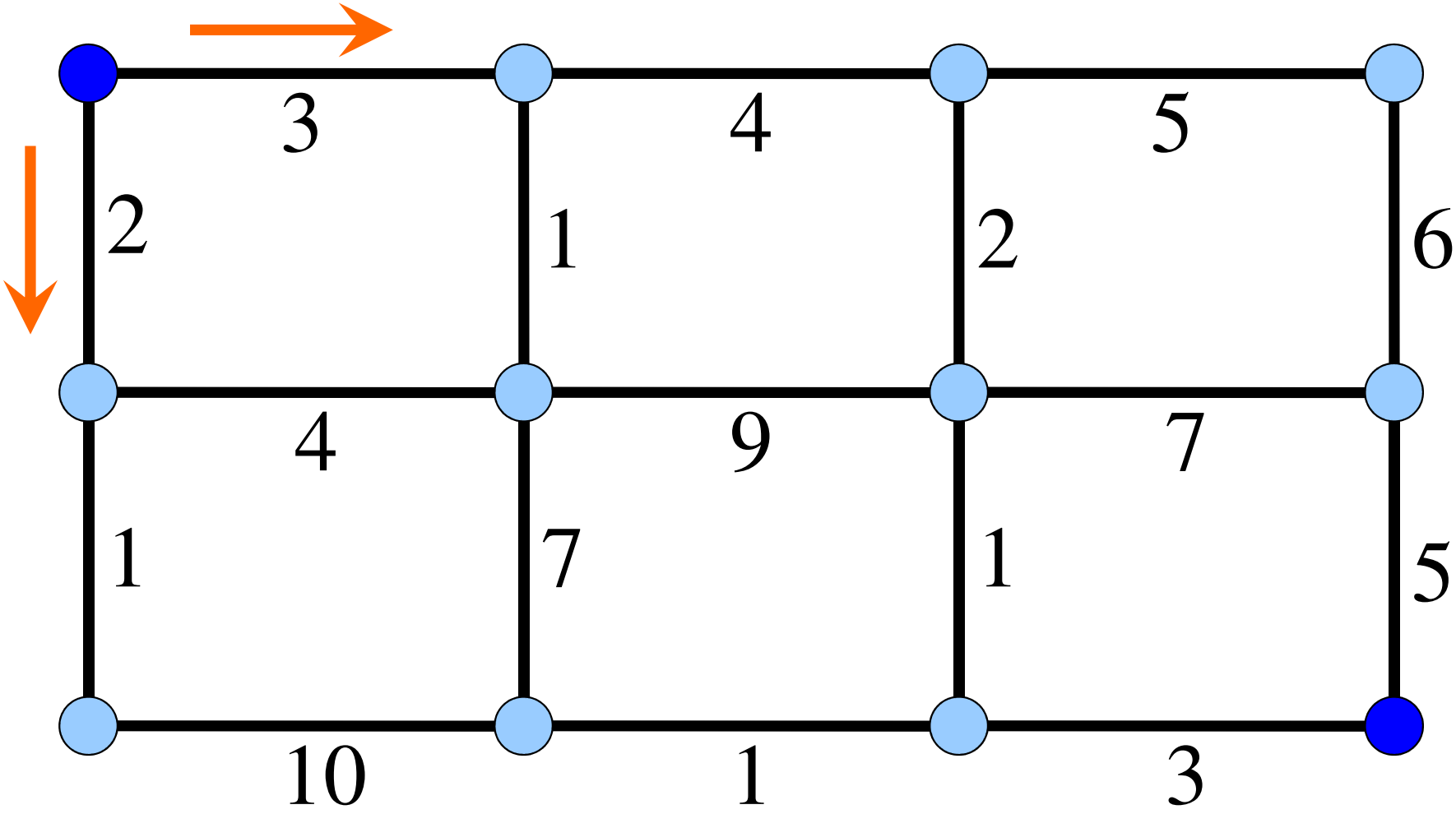
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

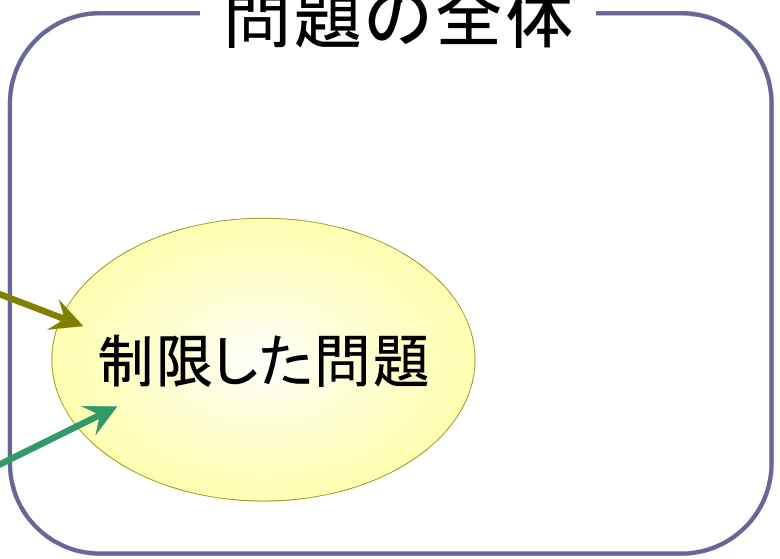


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

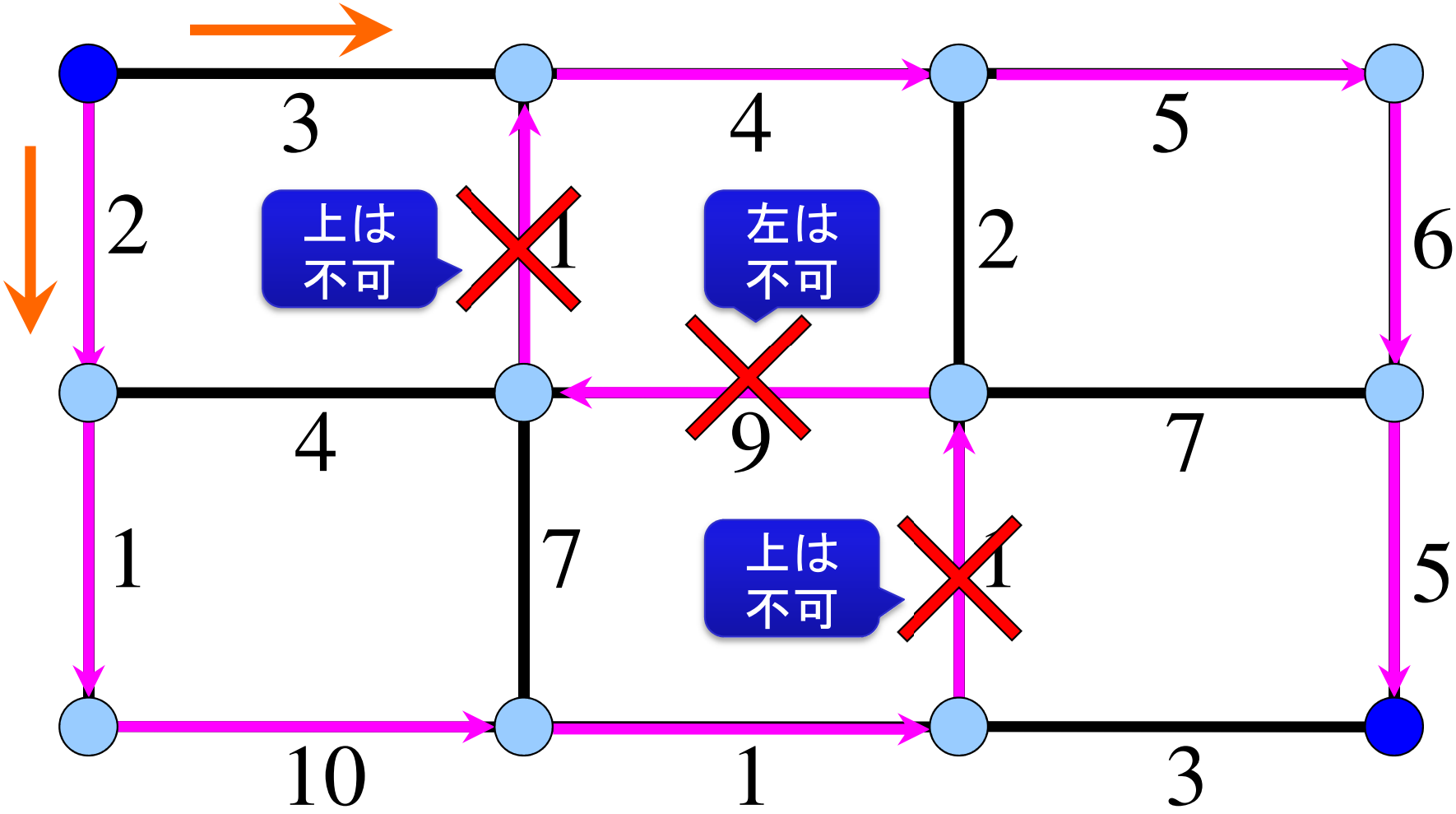
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

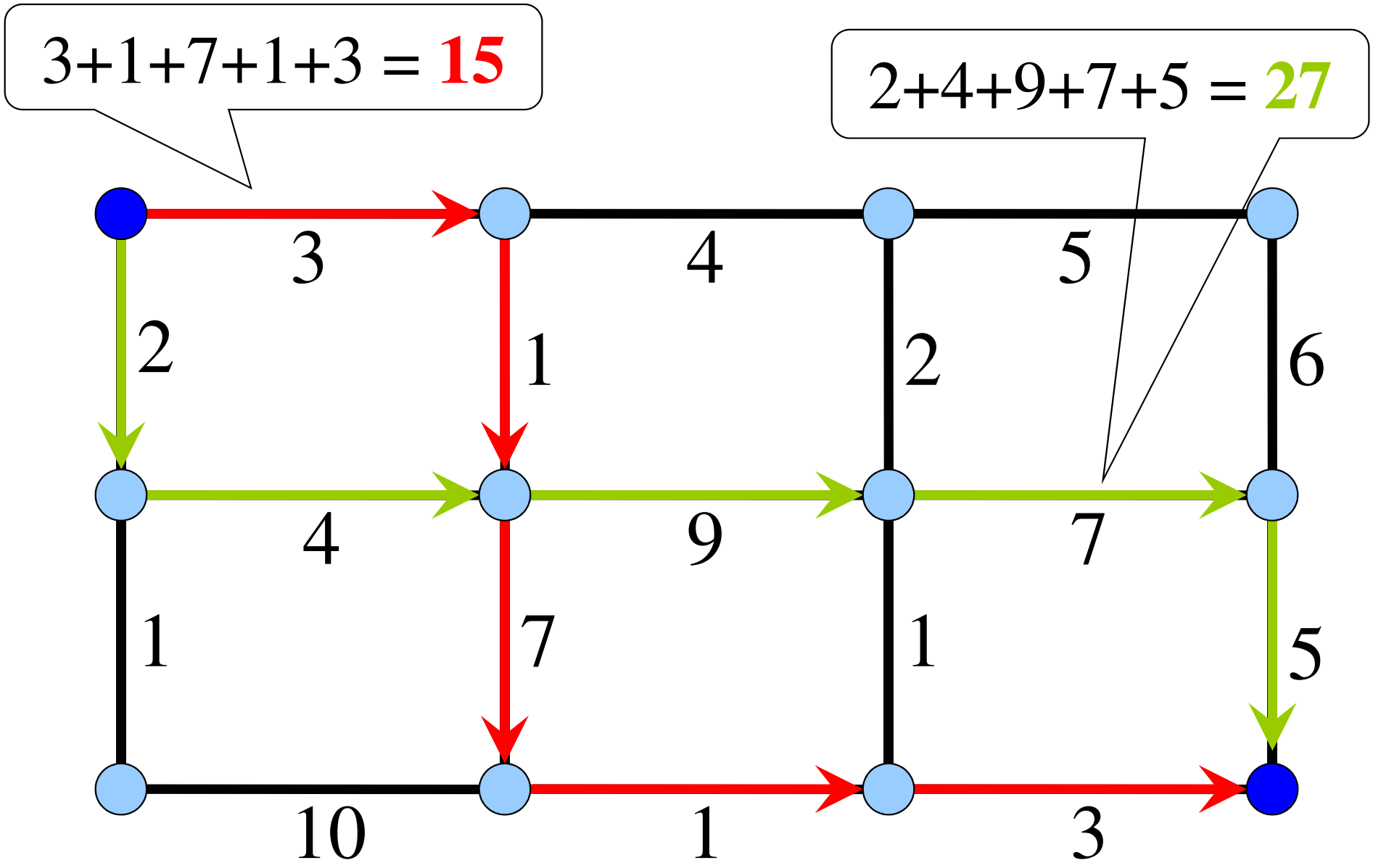
難しいなら易しくすればいいのさ！

制限した問題

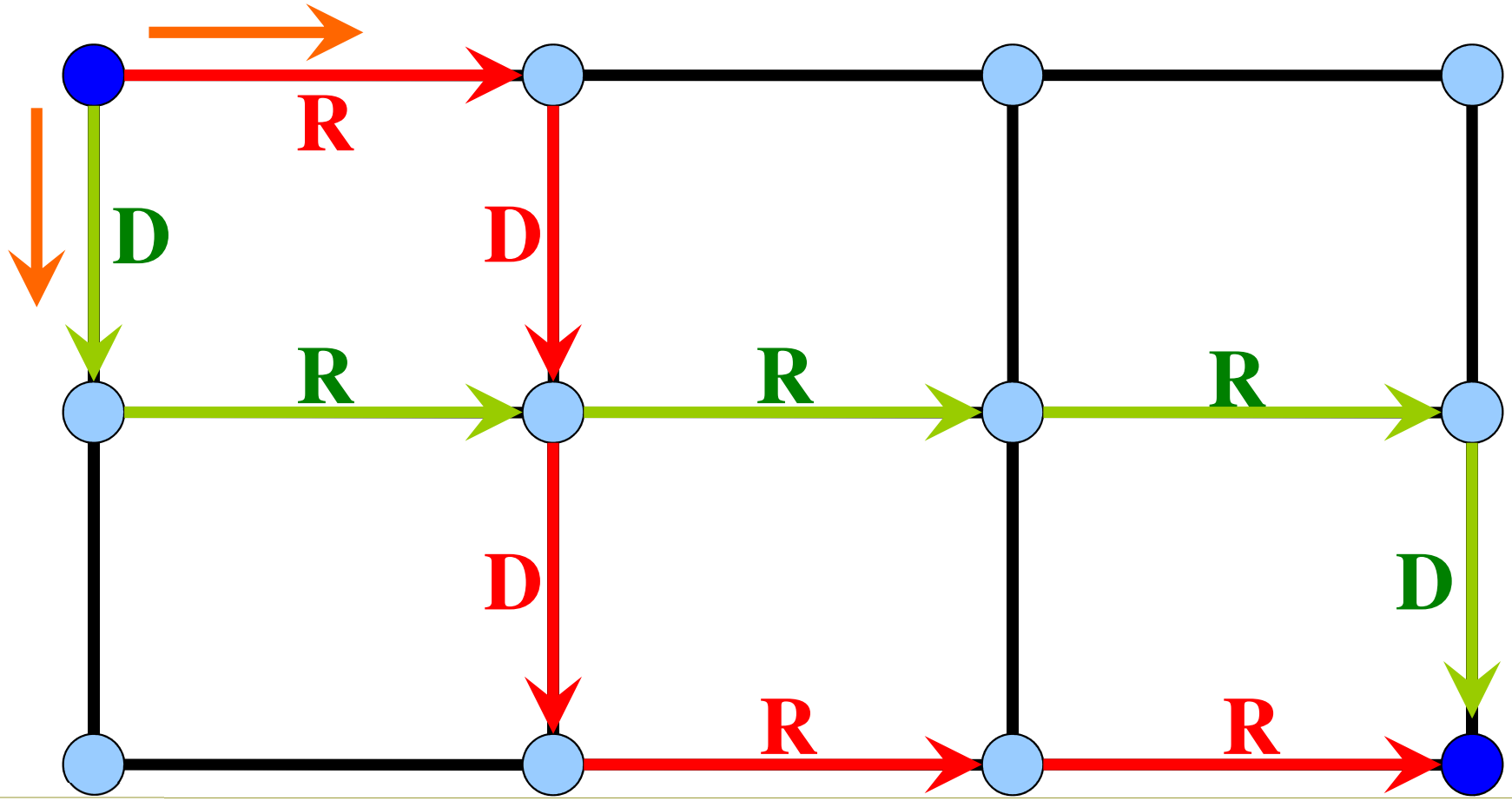
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



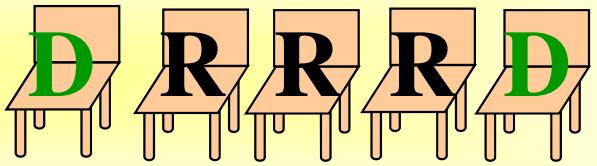
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



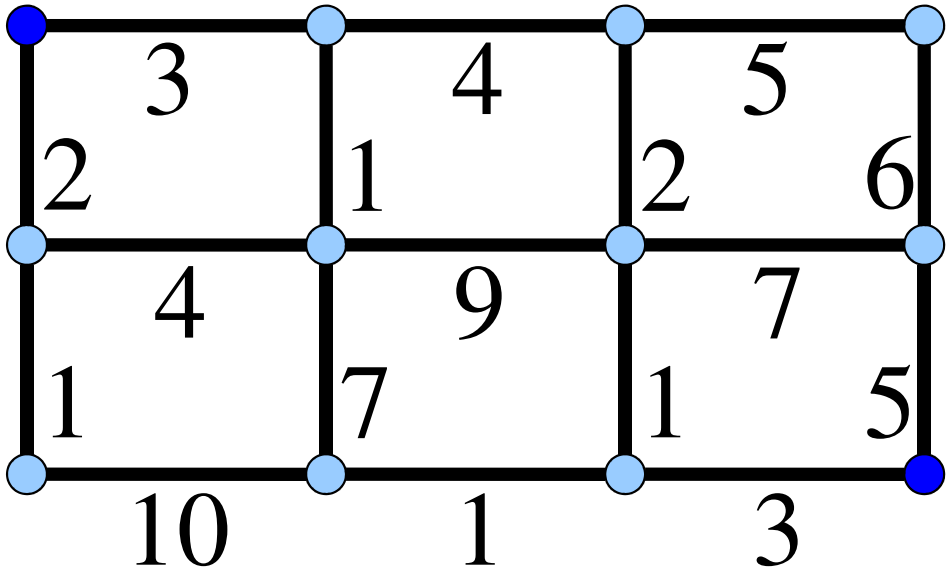
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

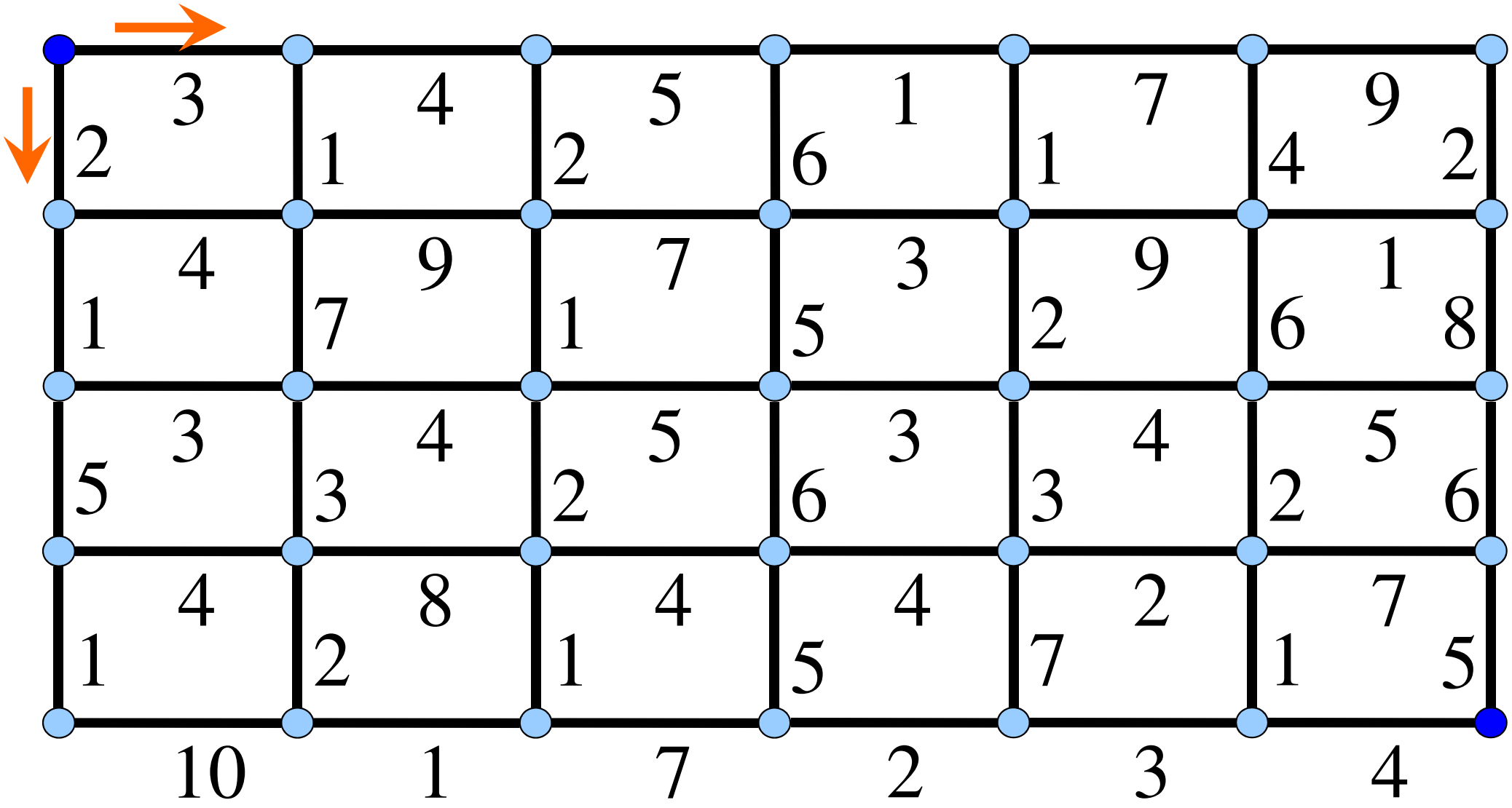
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

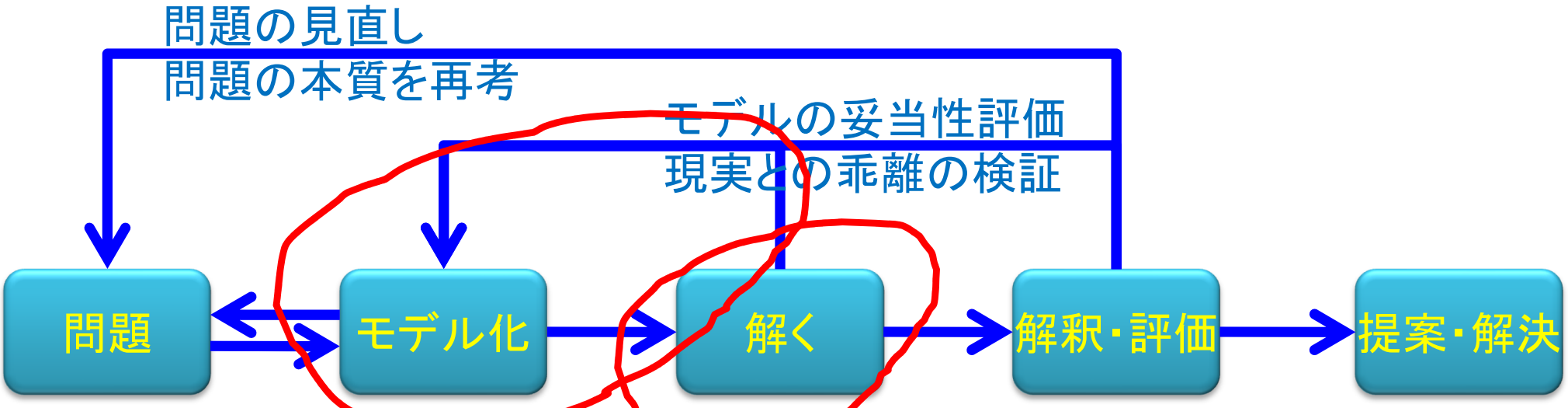
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

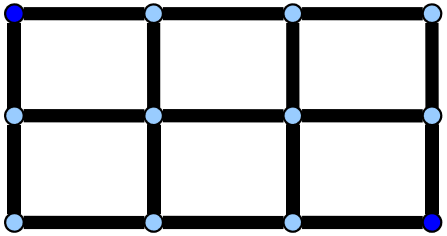
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

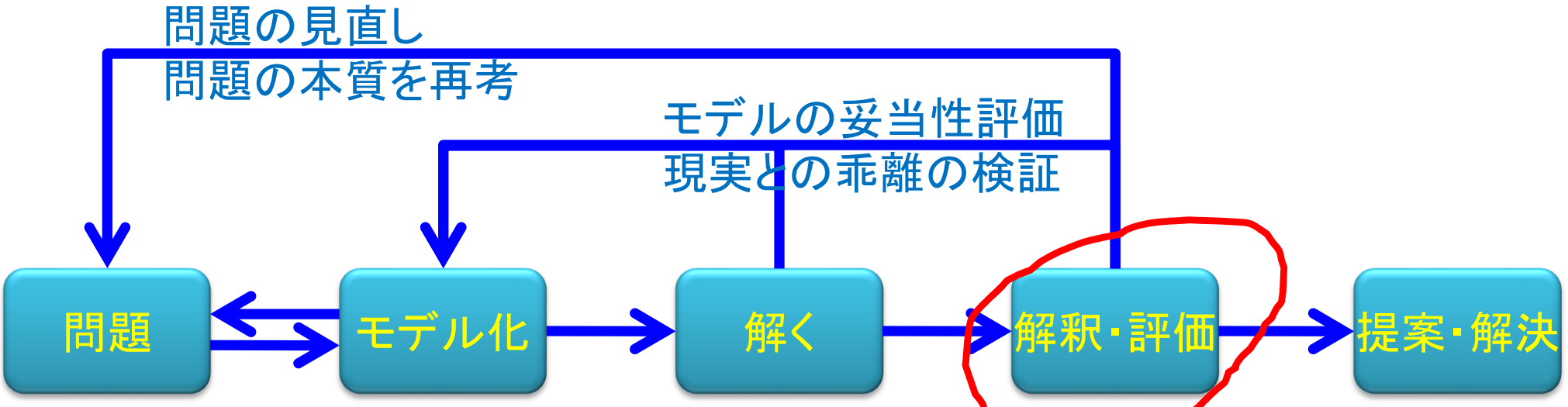
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

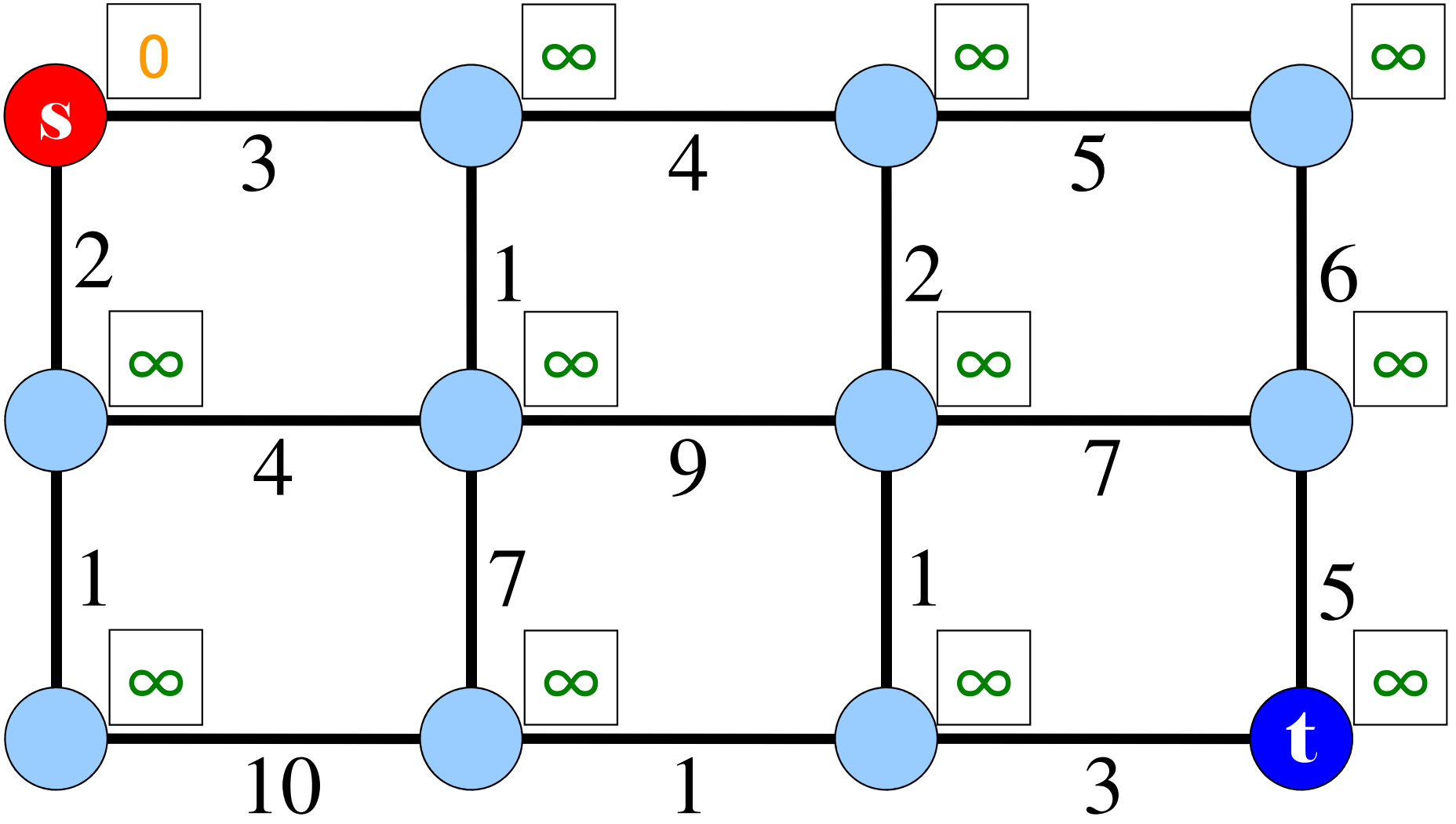
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

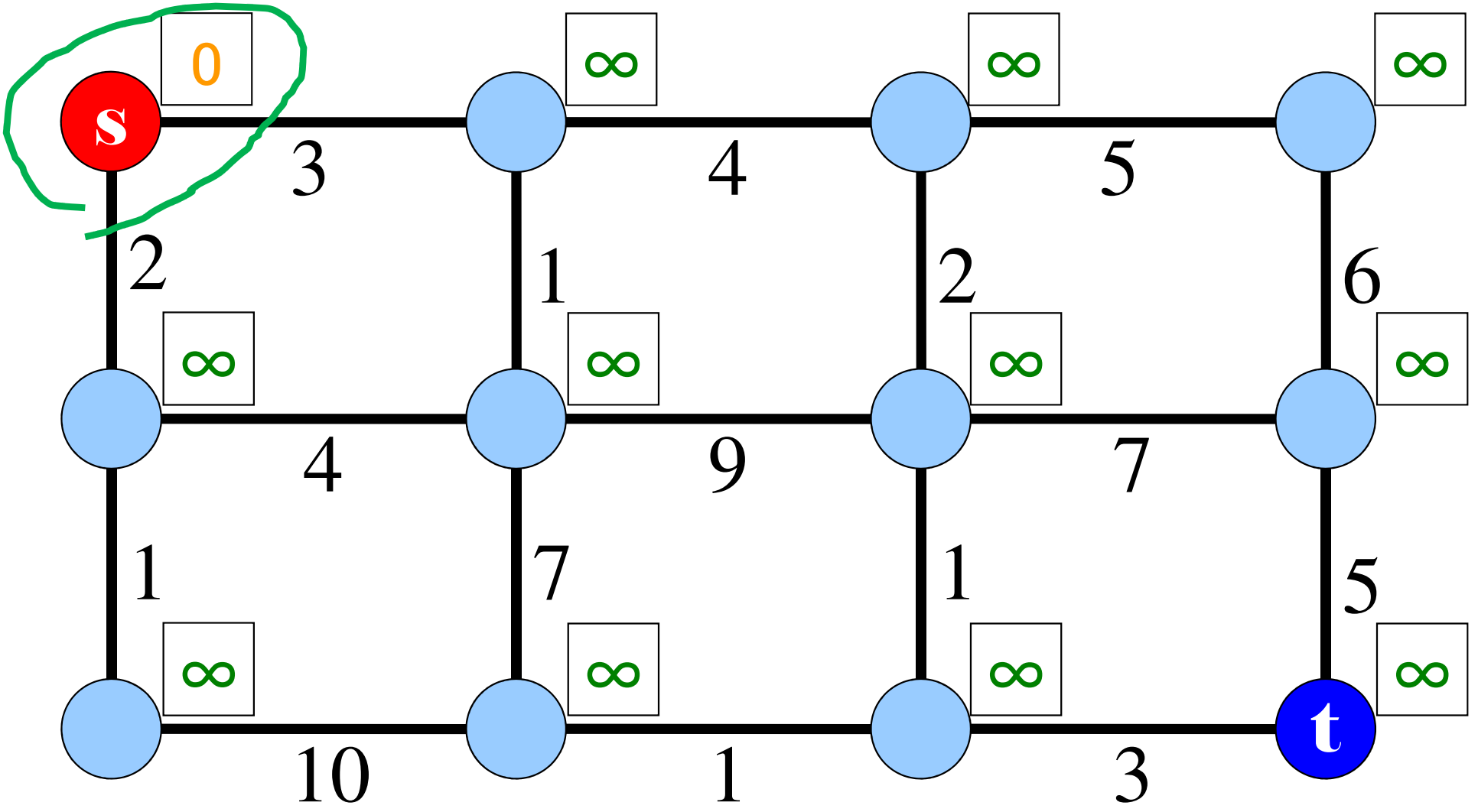
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



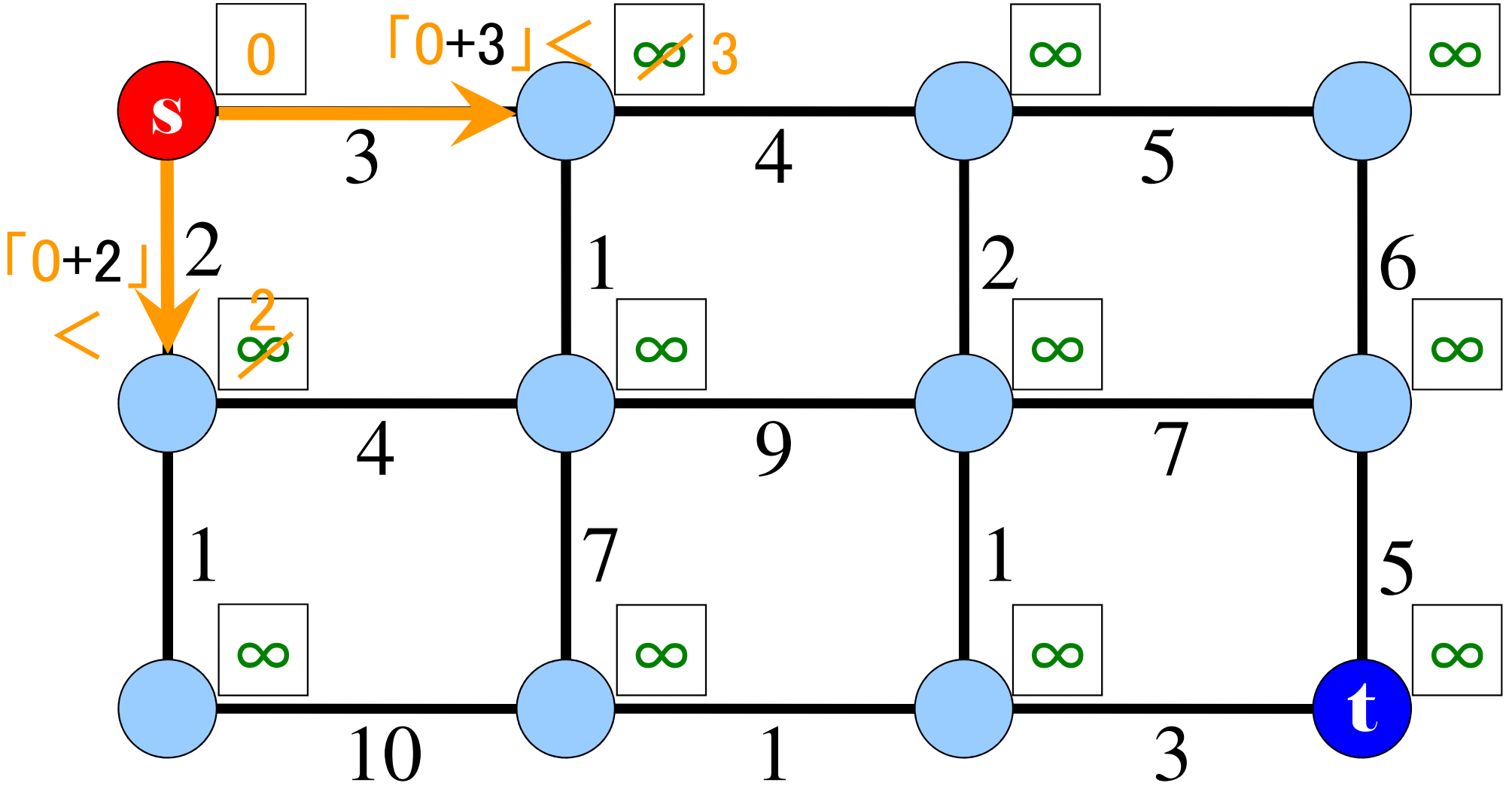
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

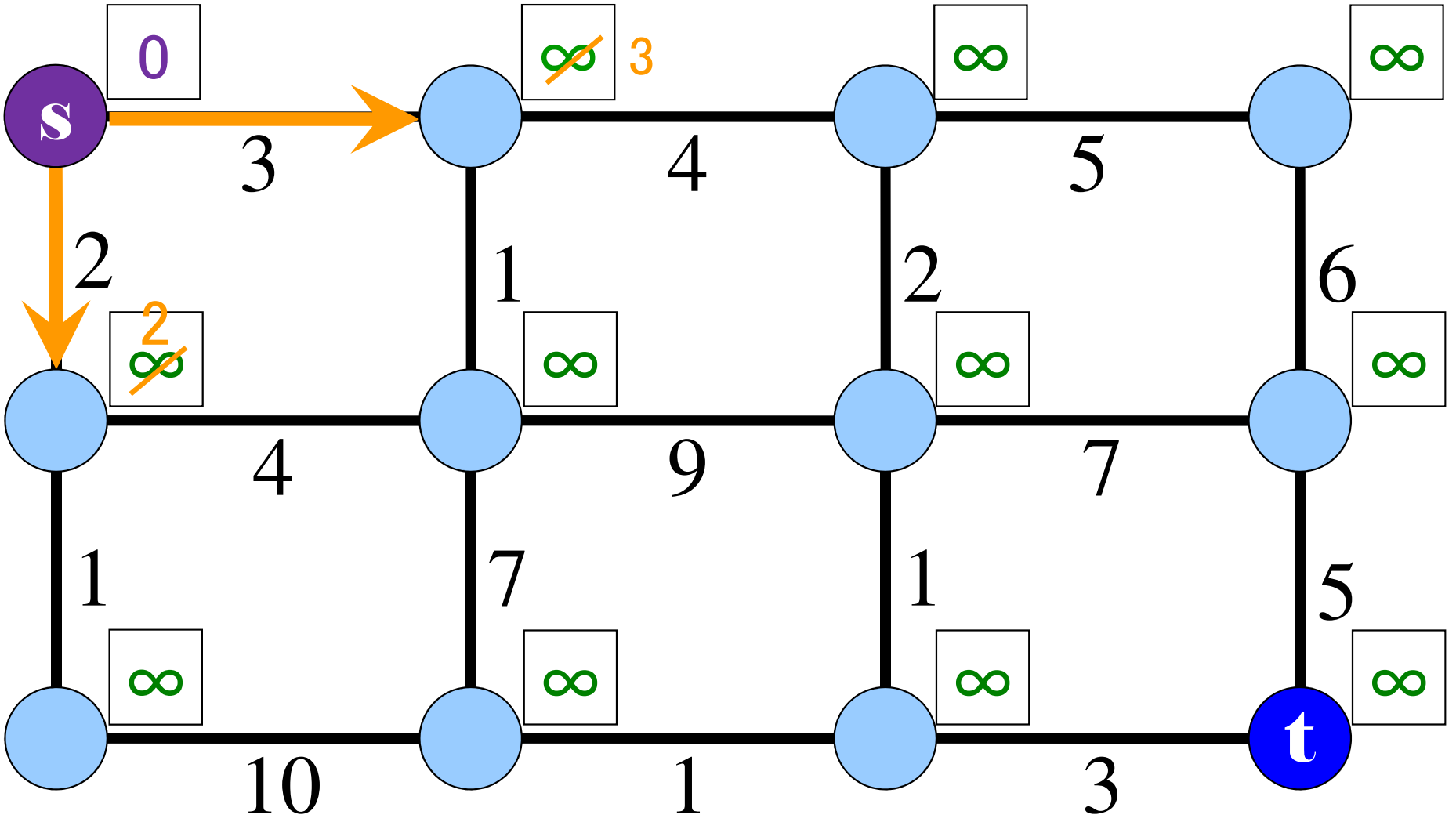


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

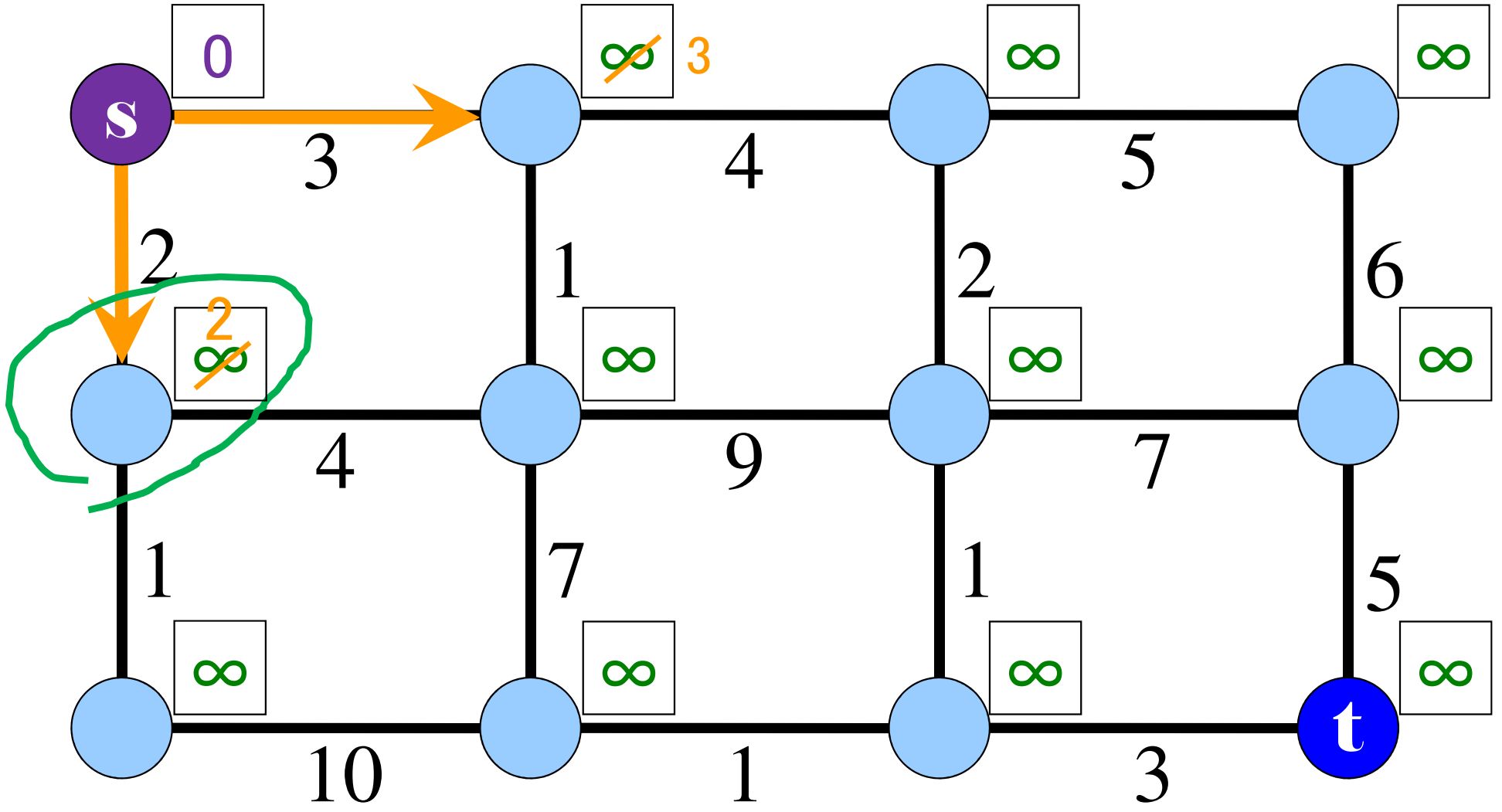
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



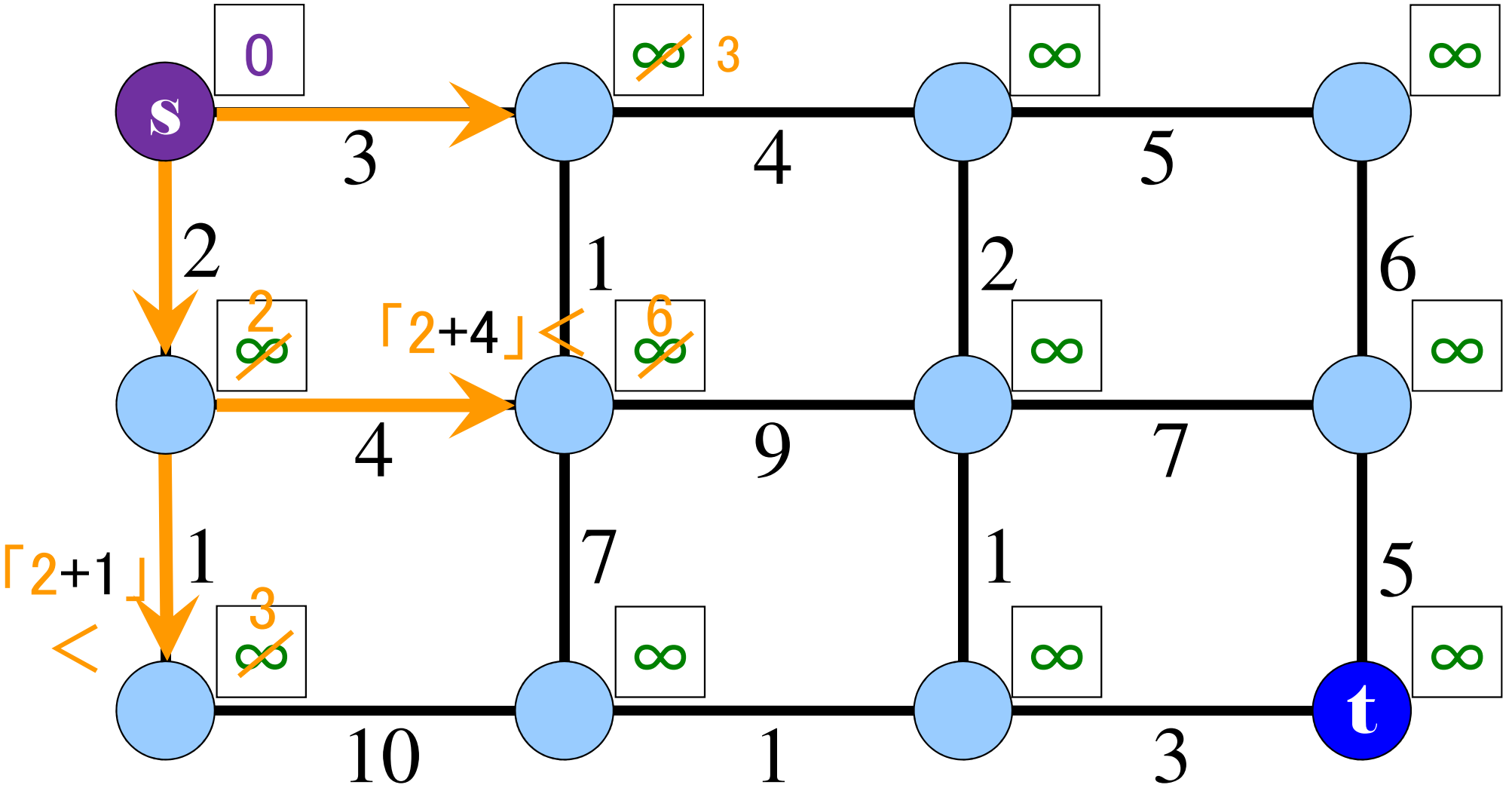
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

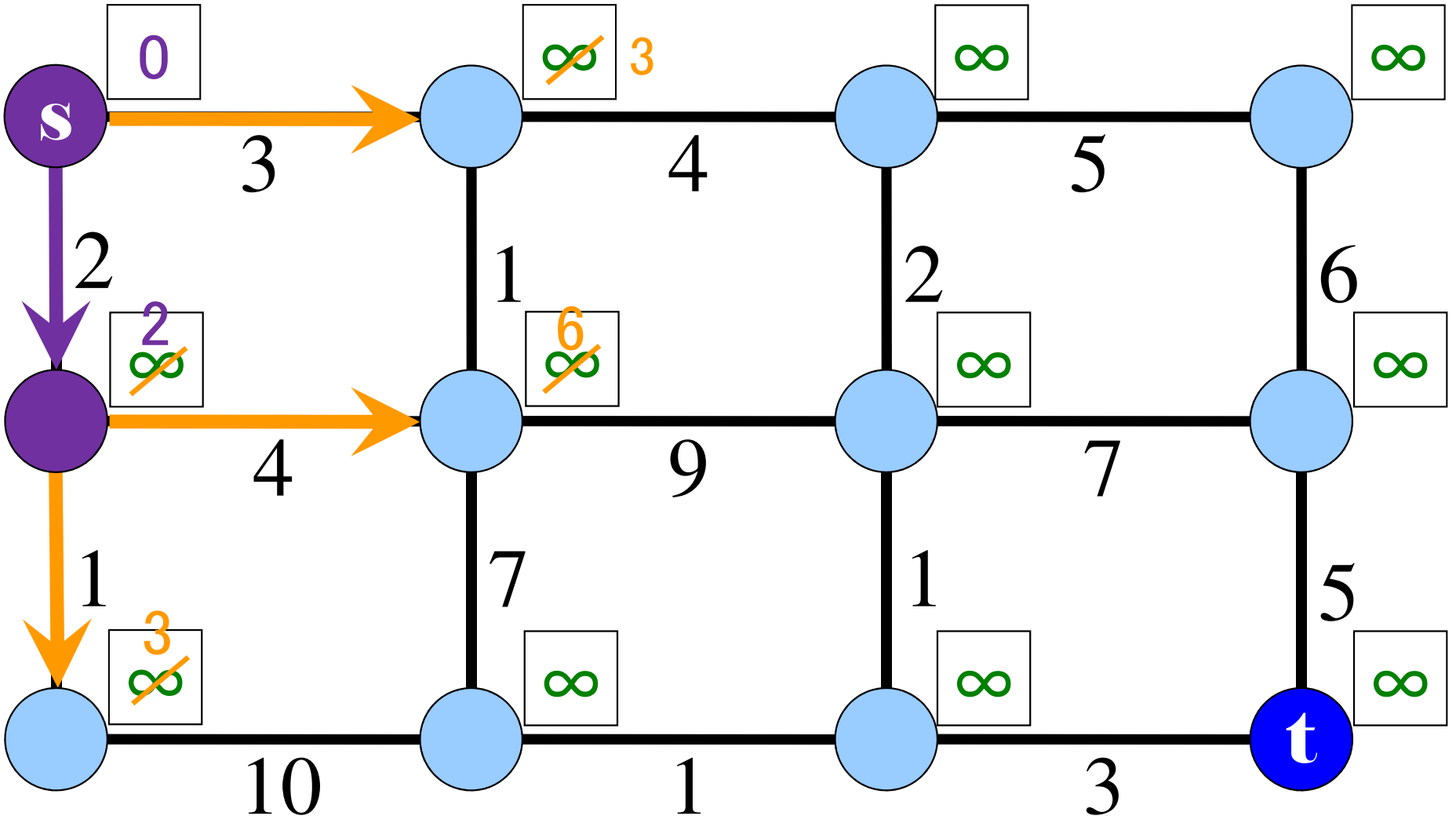


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

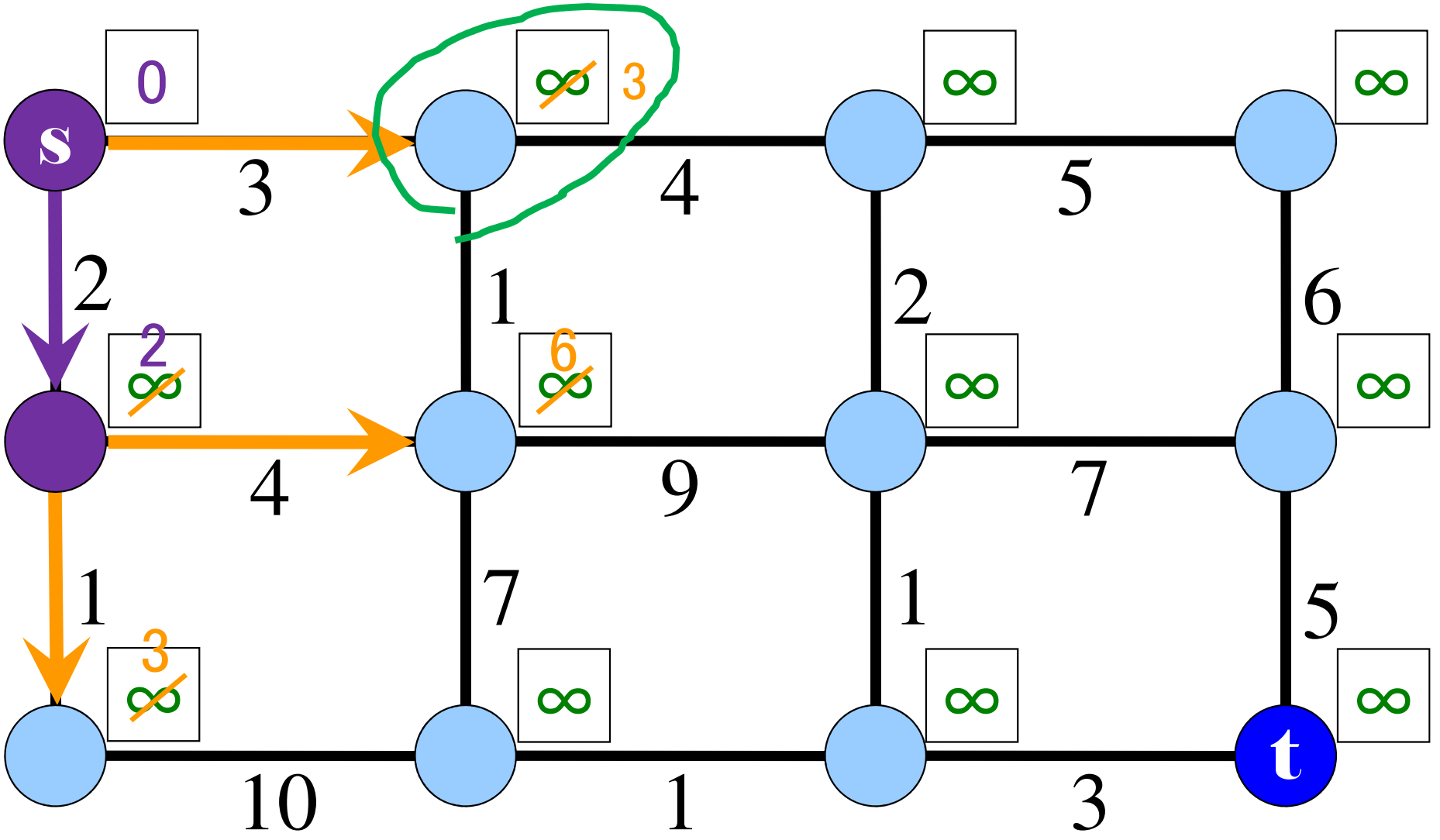
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



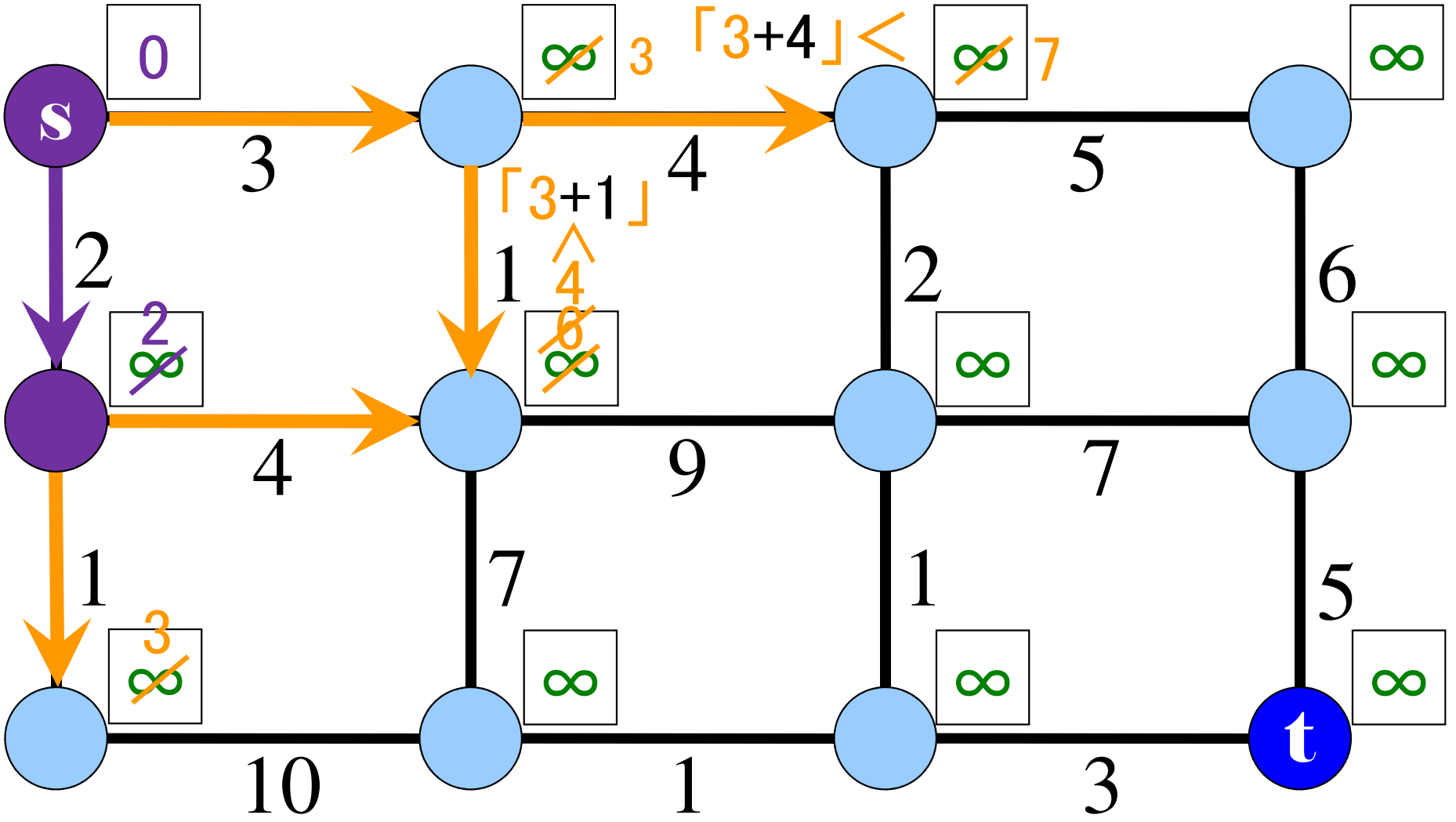
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

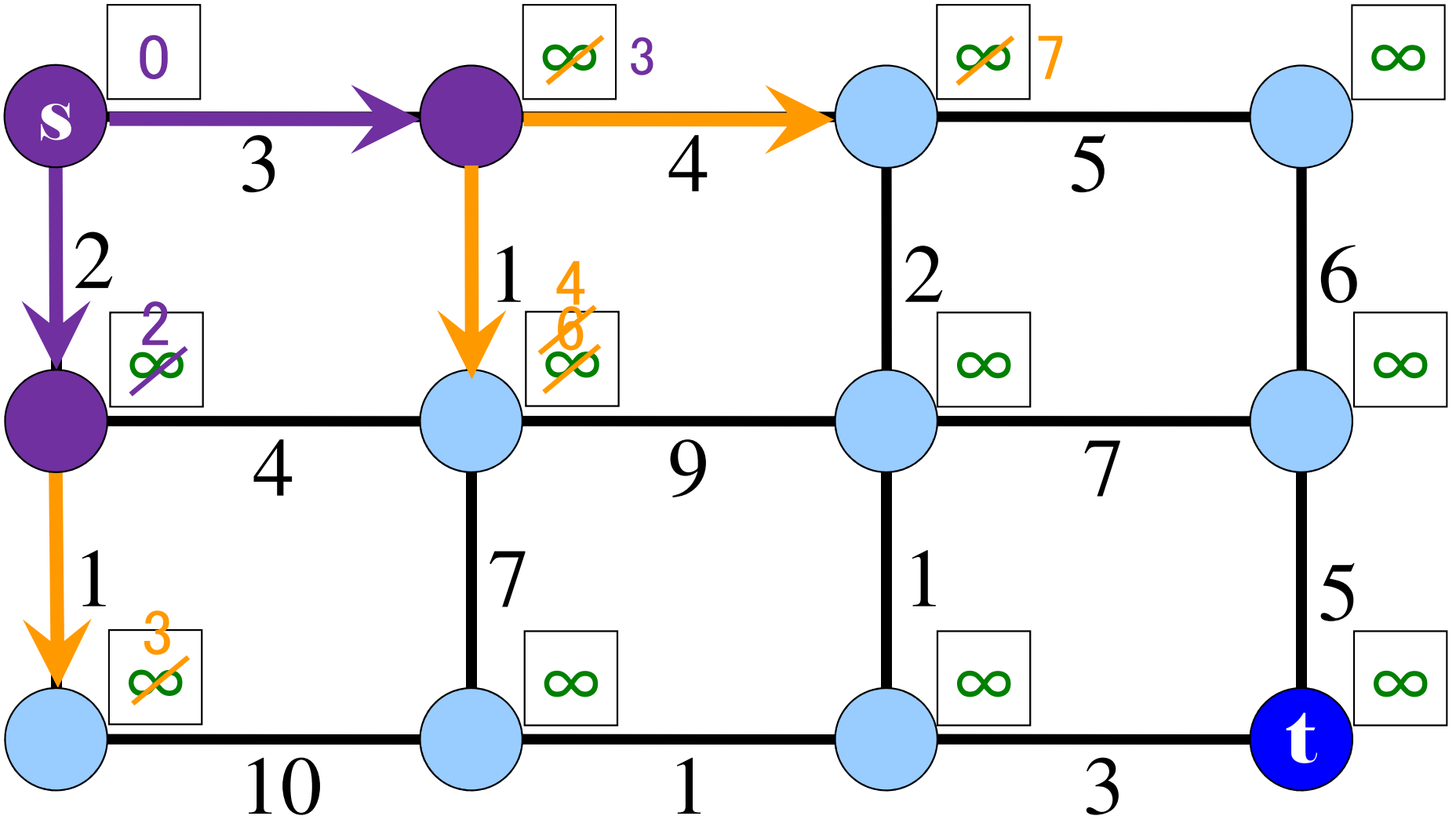


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

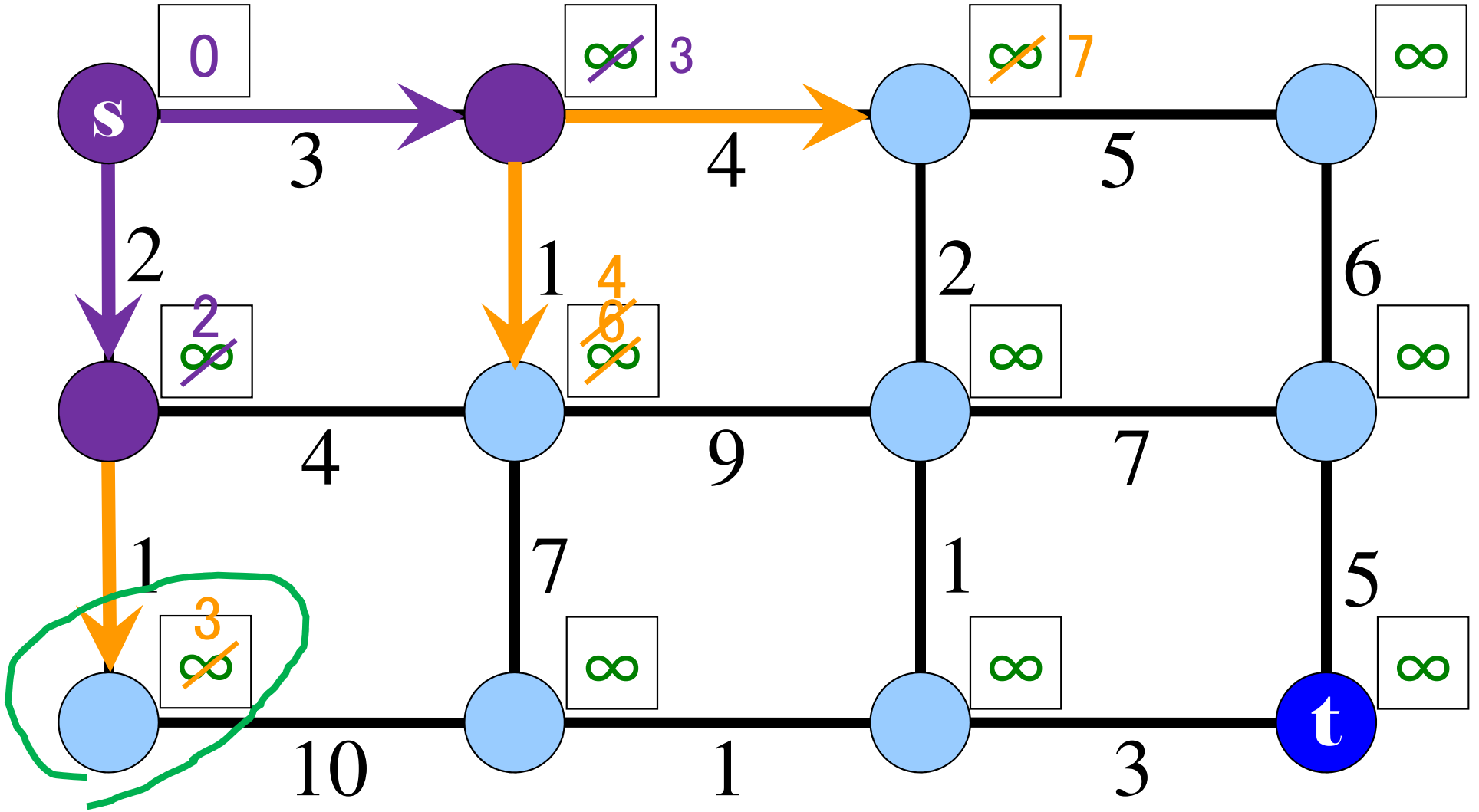
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



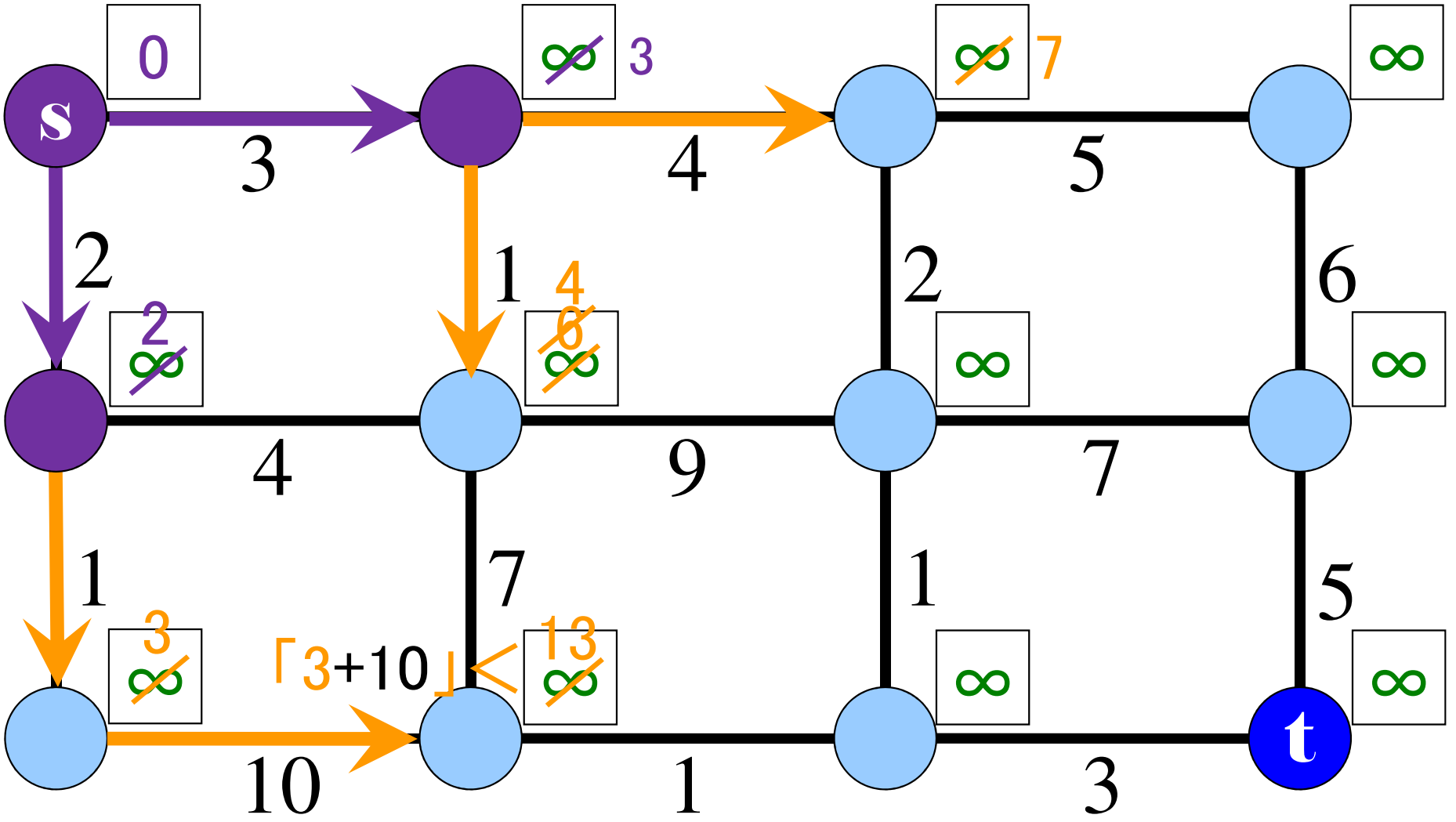
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

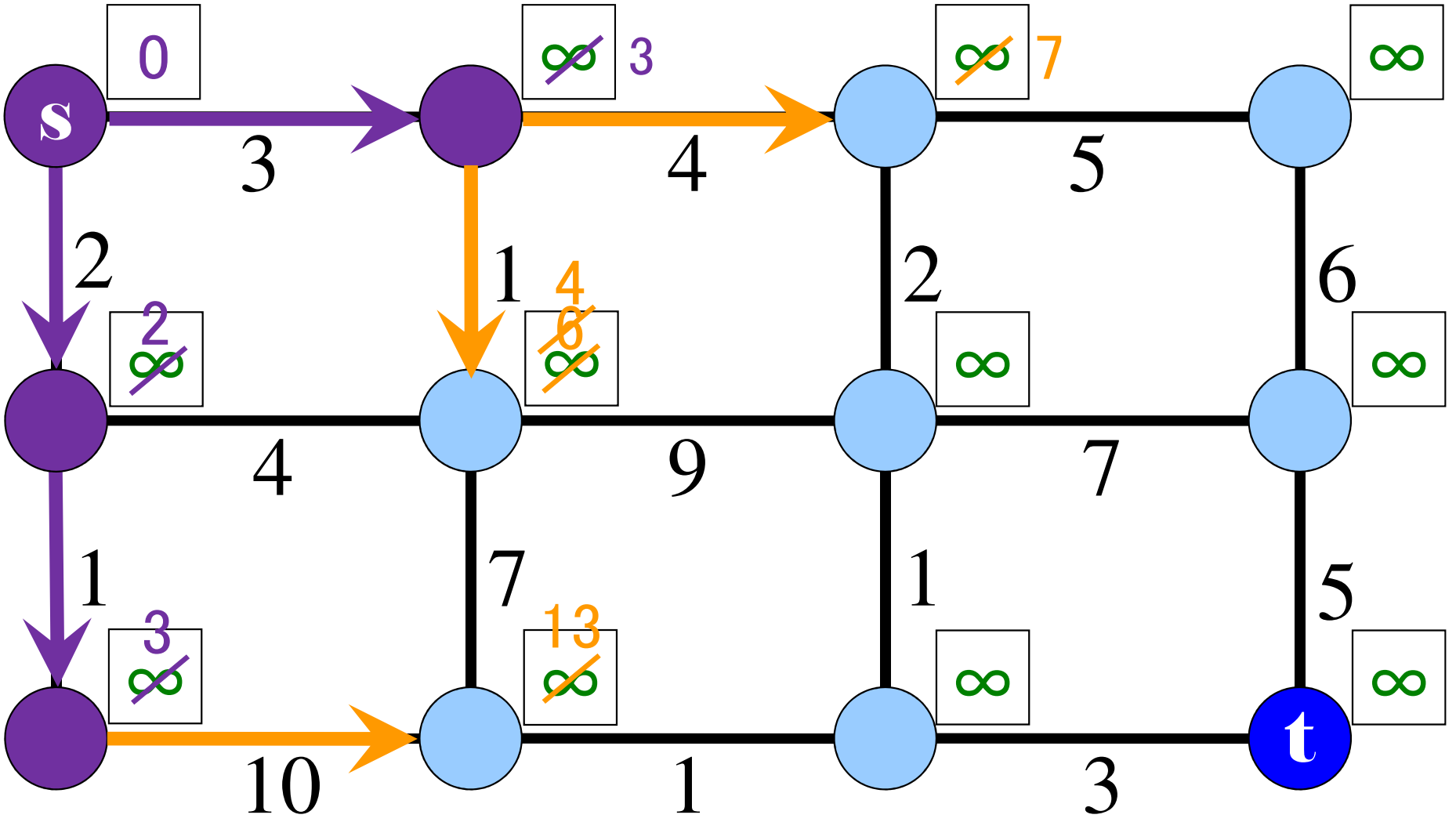


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

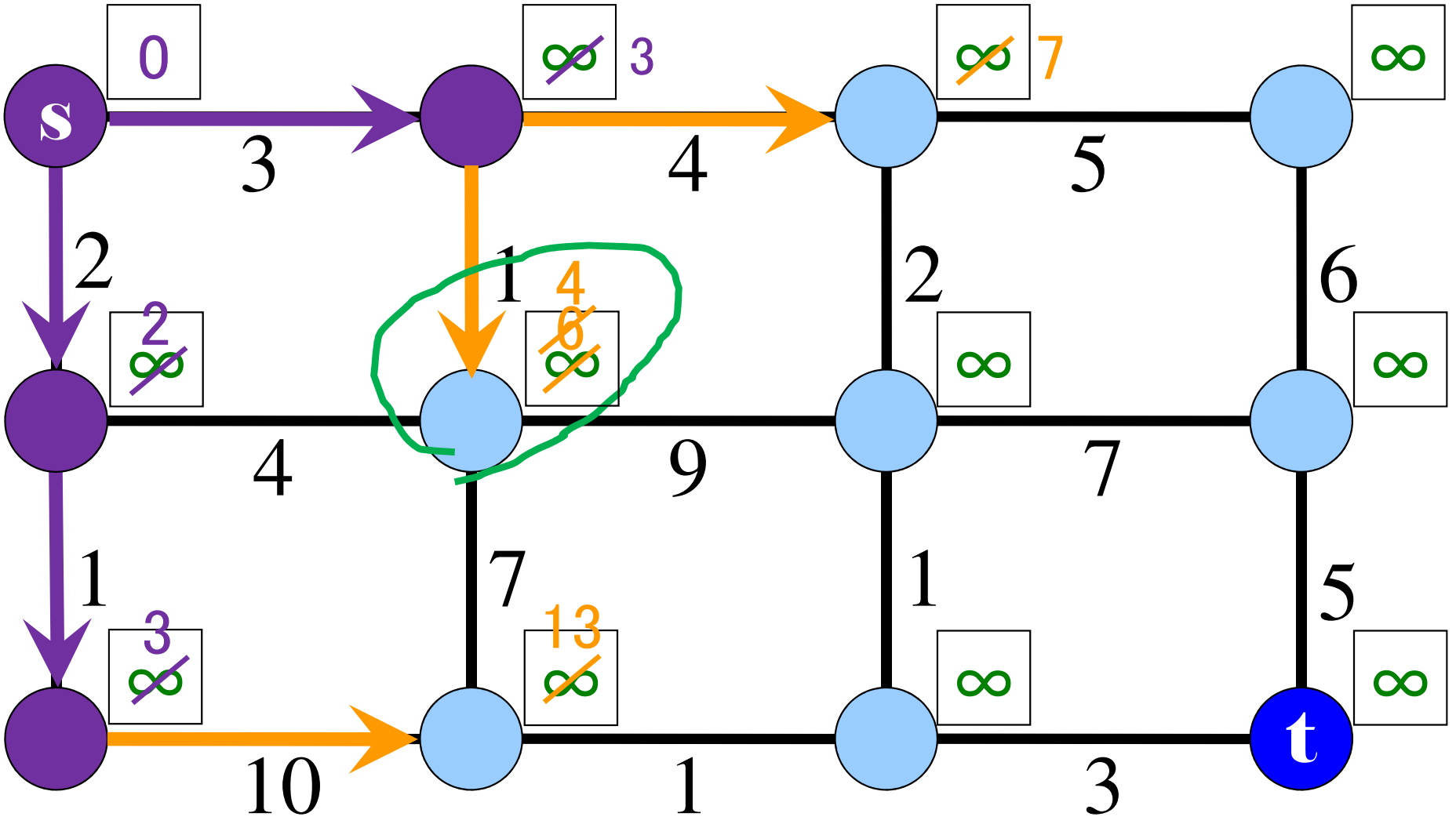
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



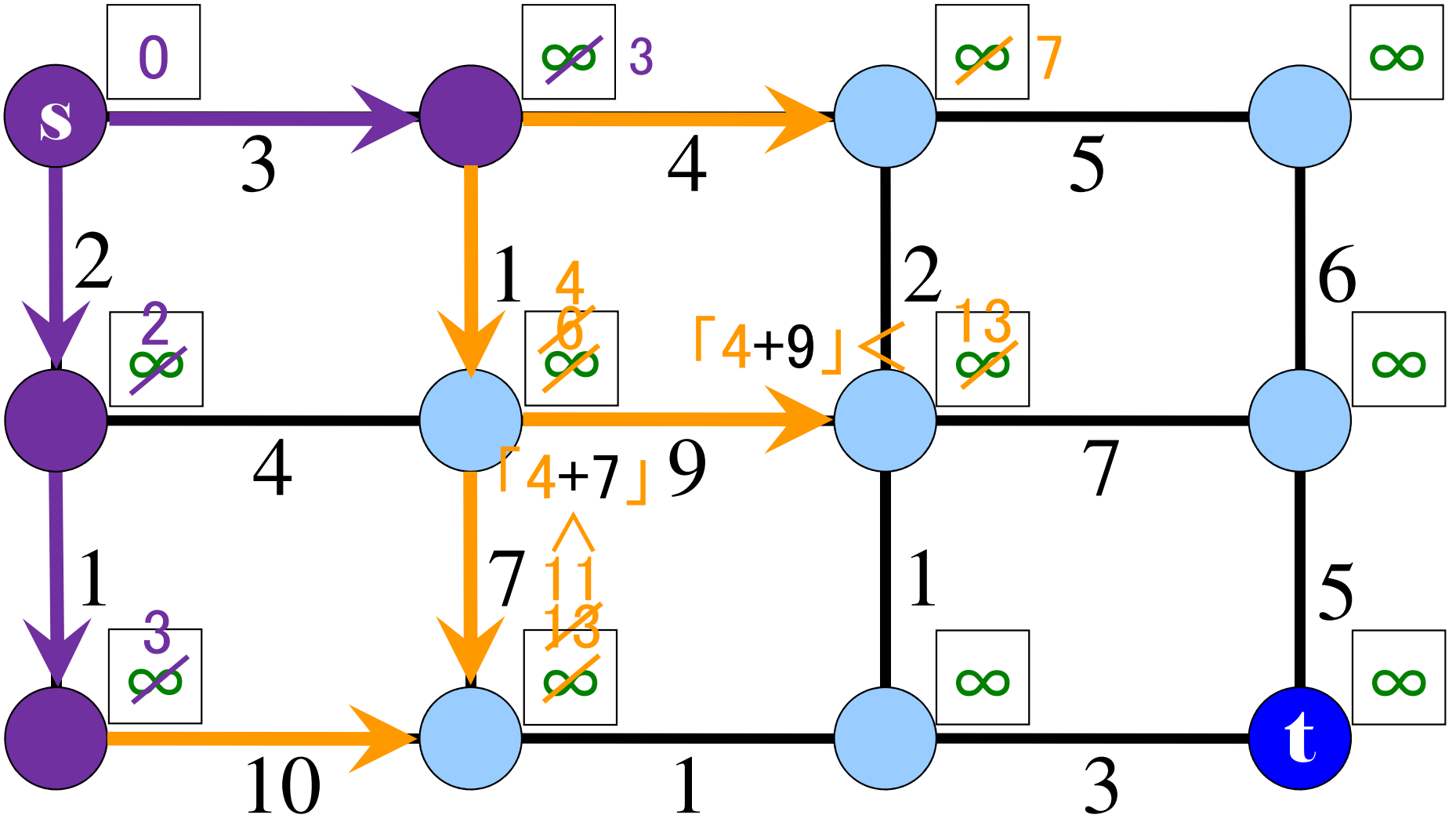
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

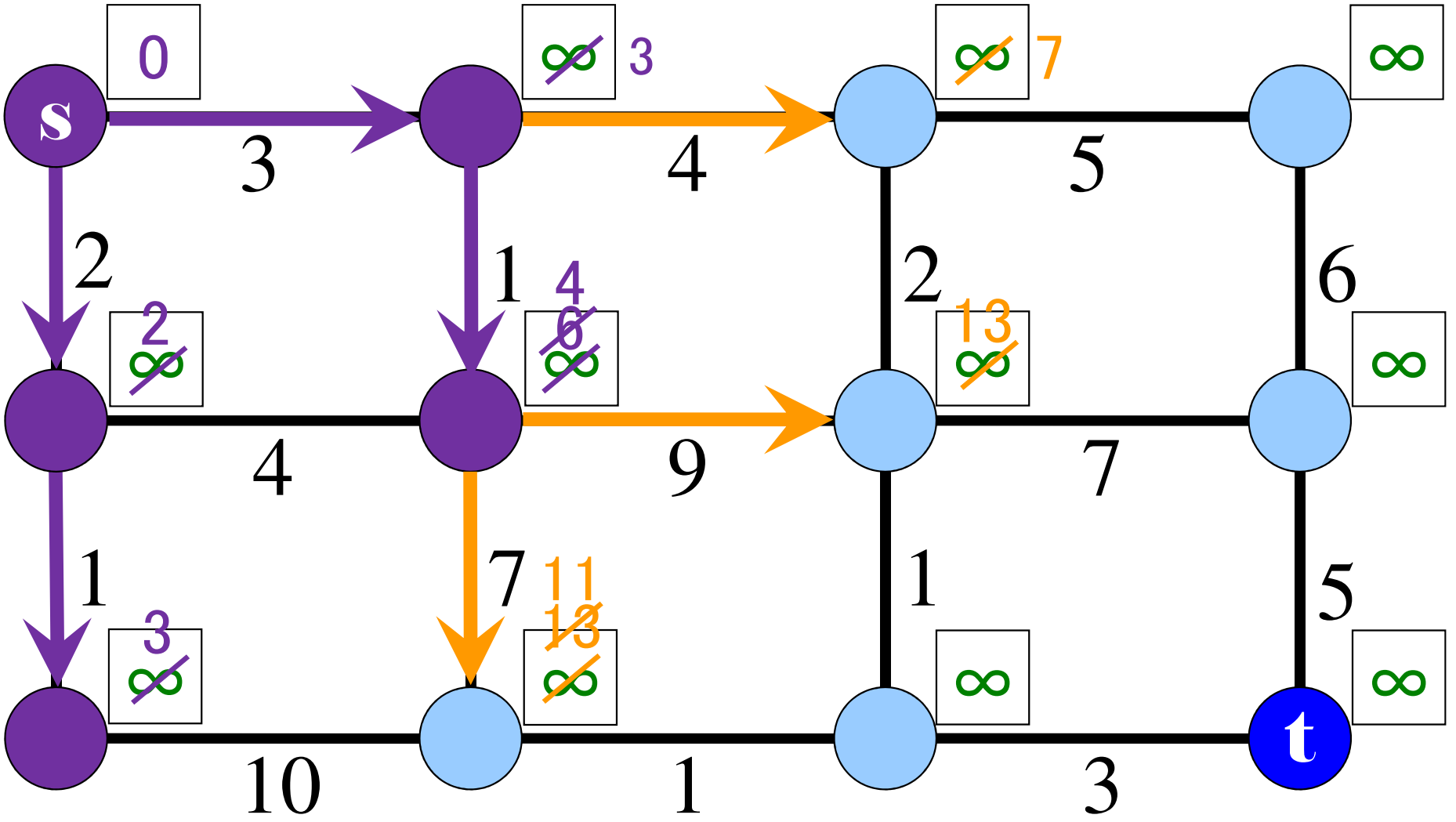


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

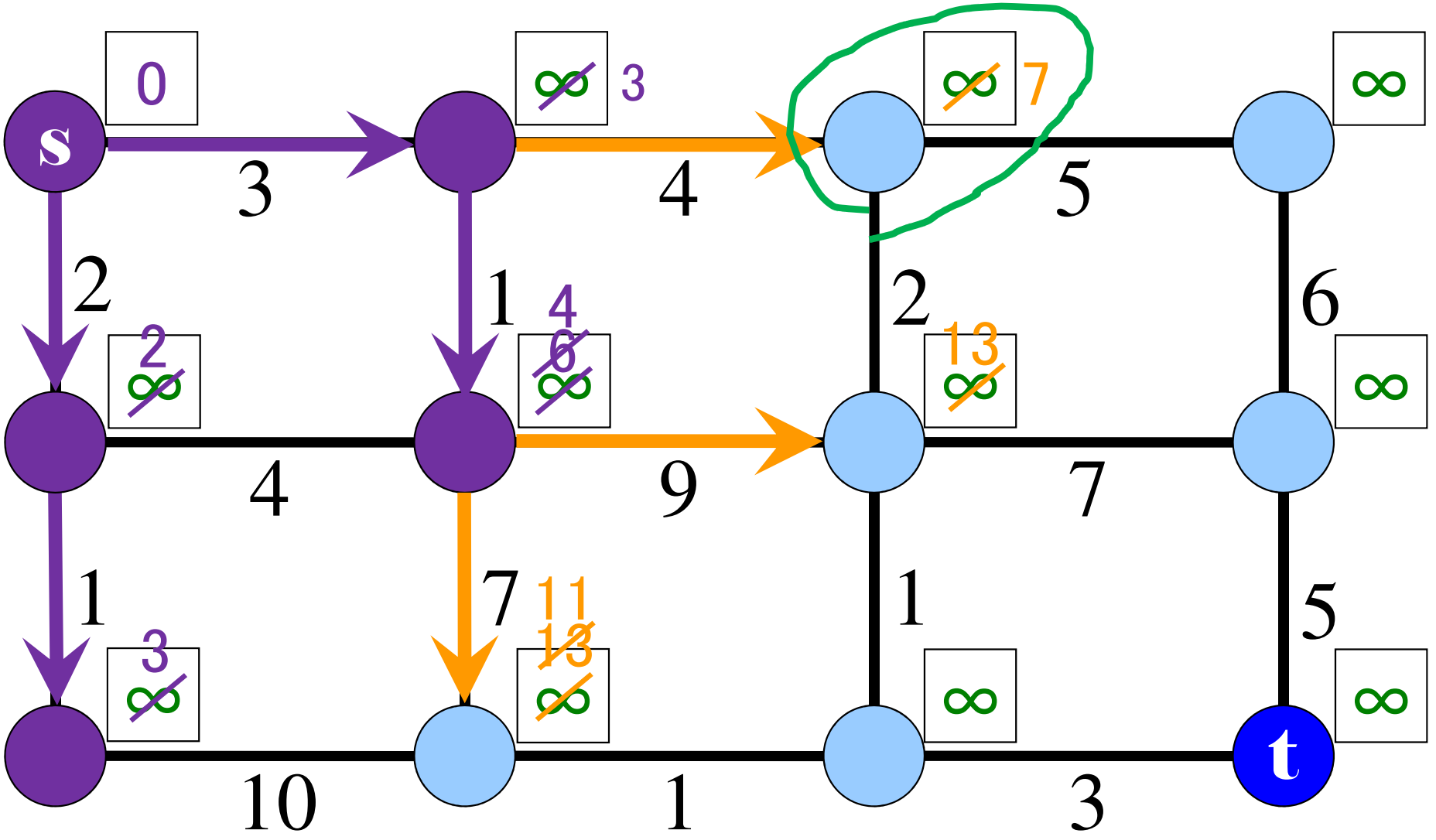
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



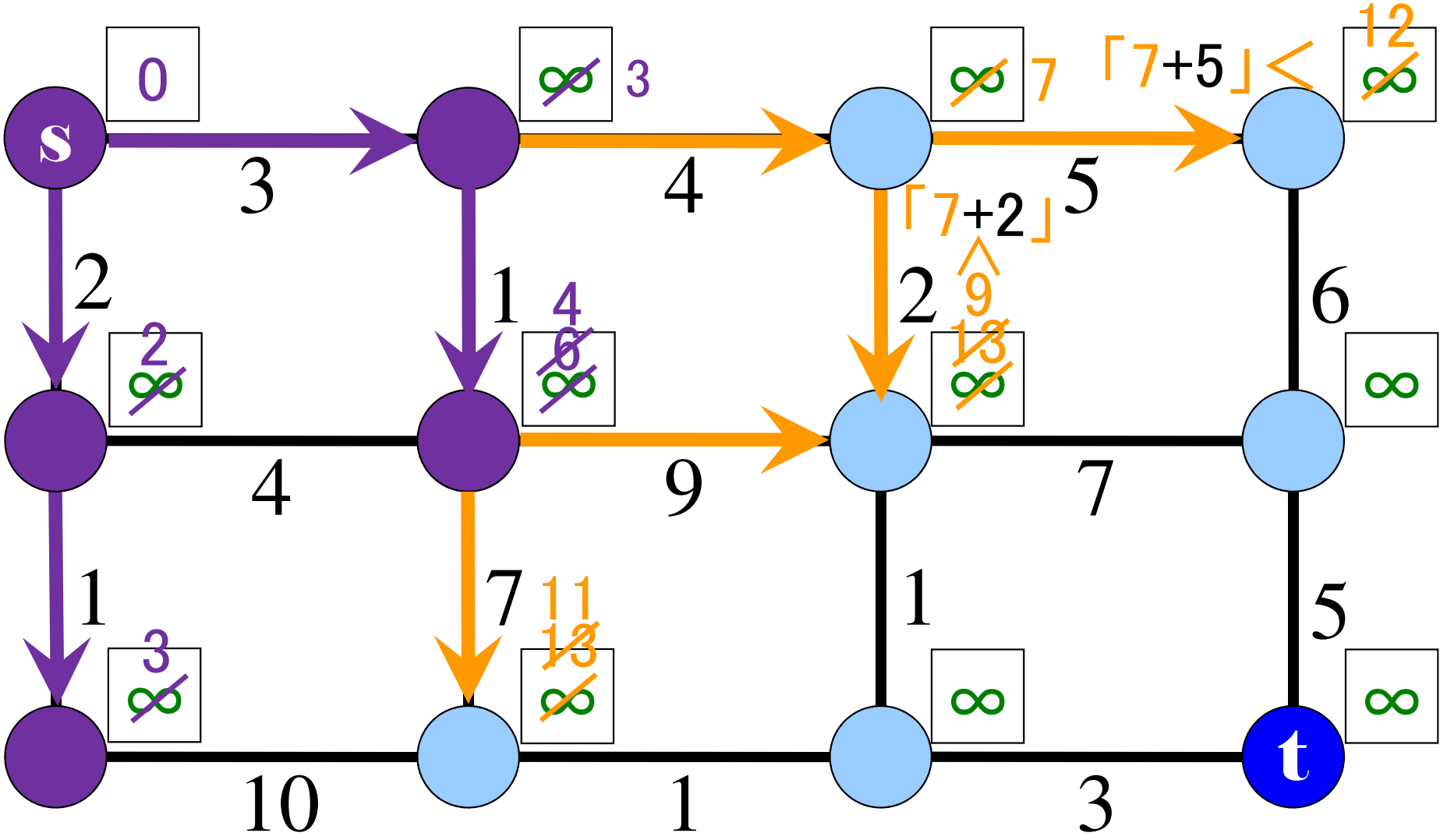
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

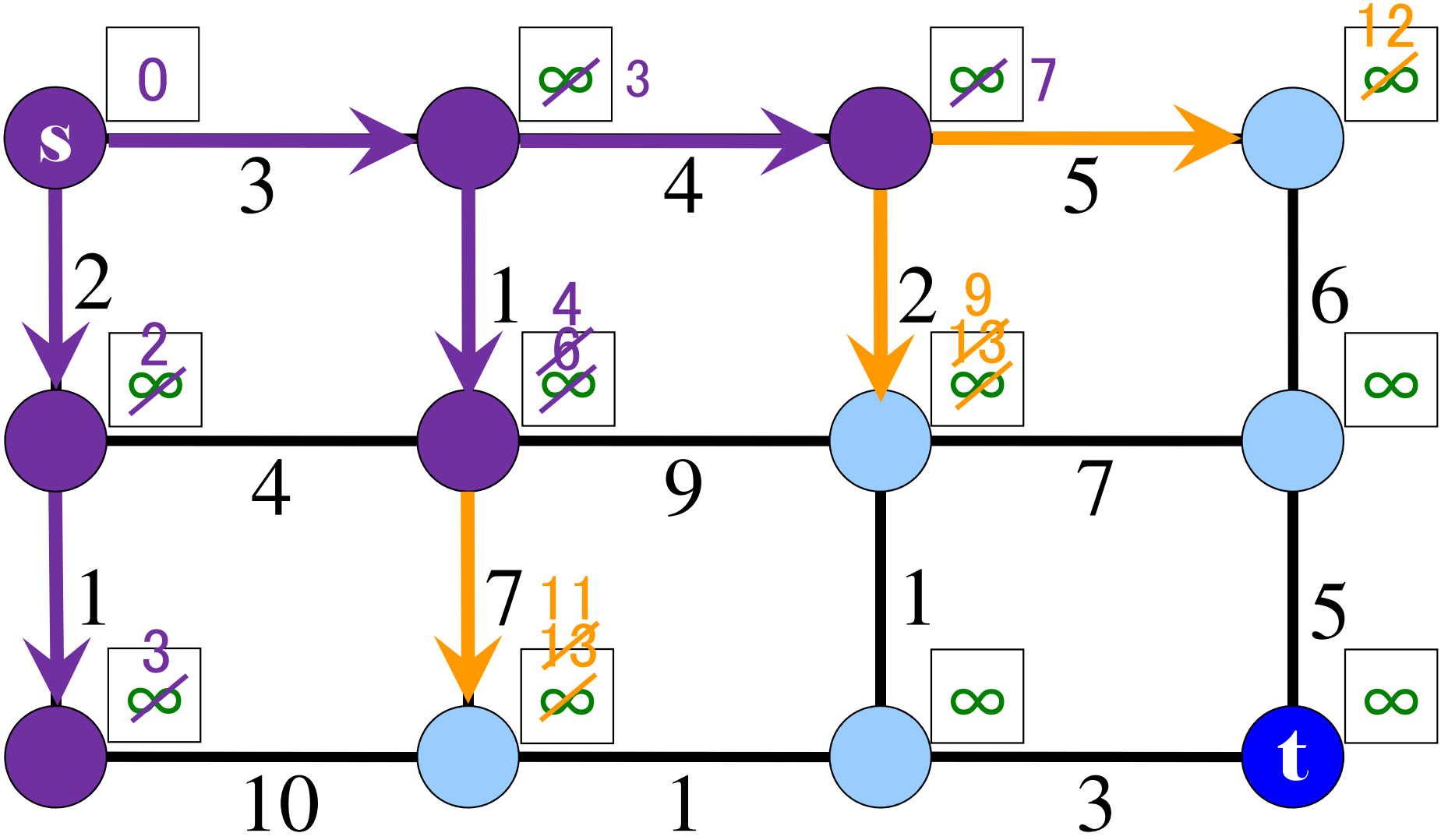


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

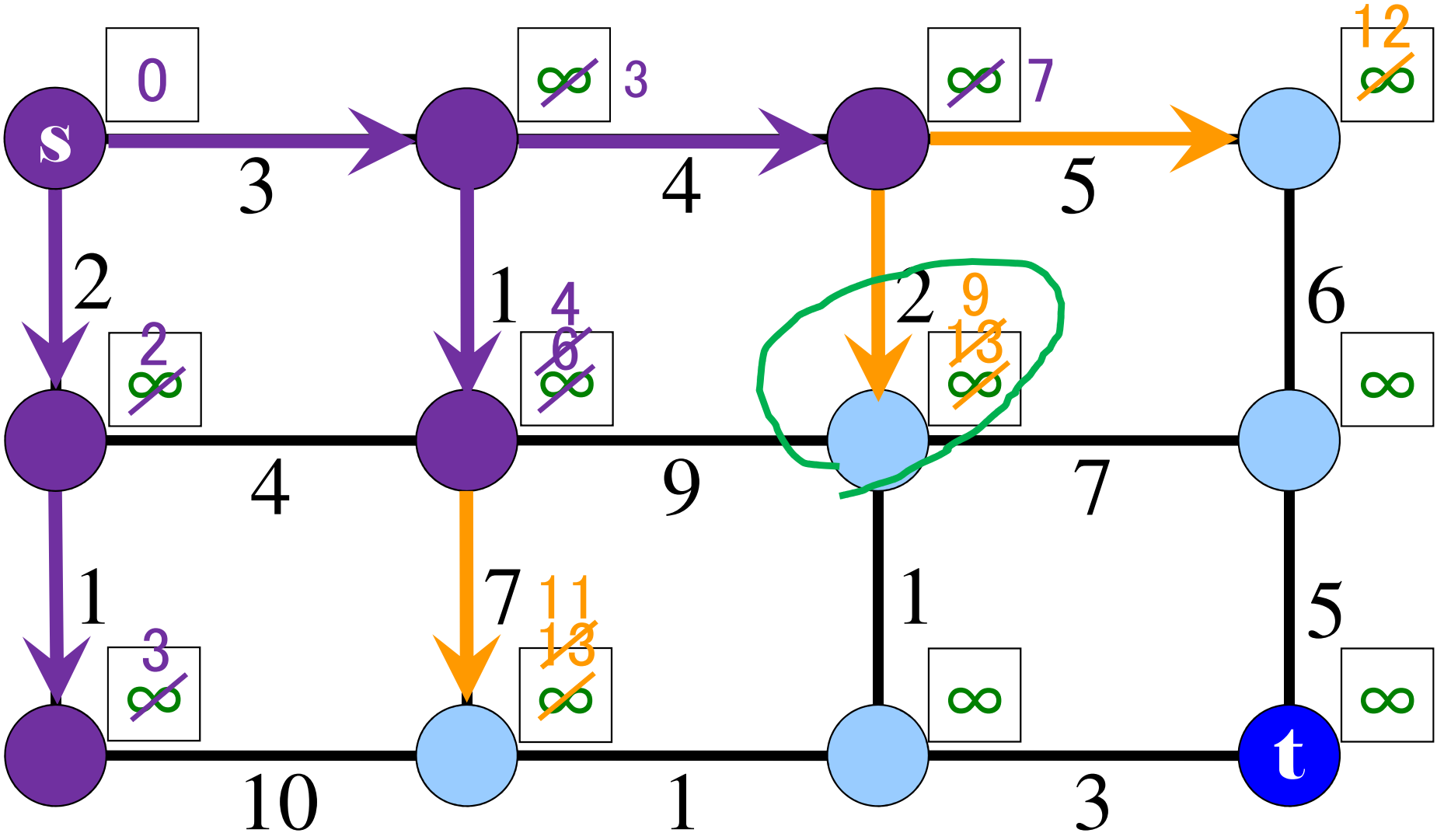
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



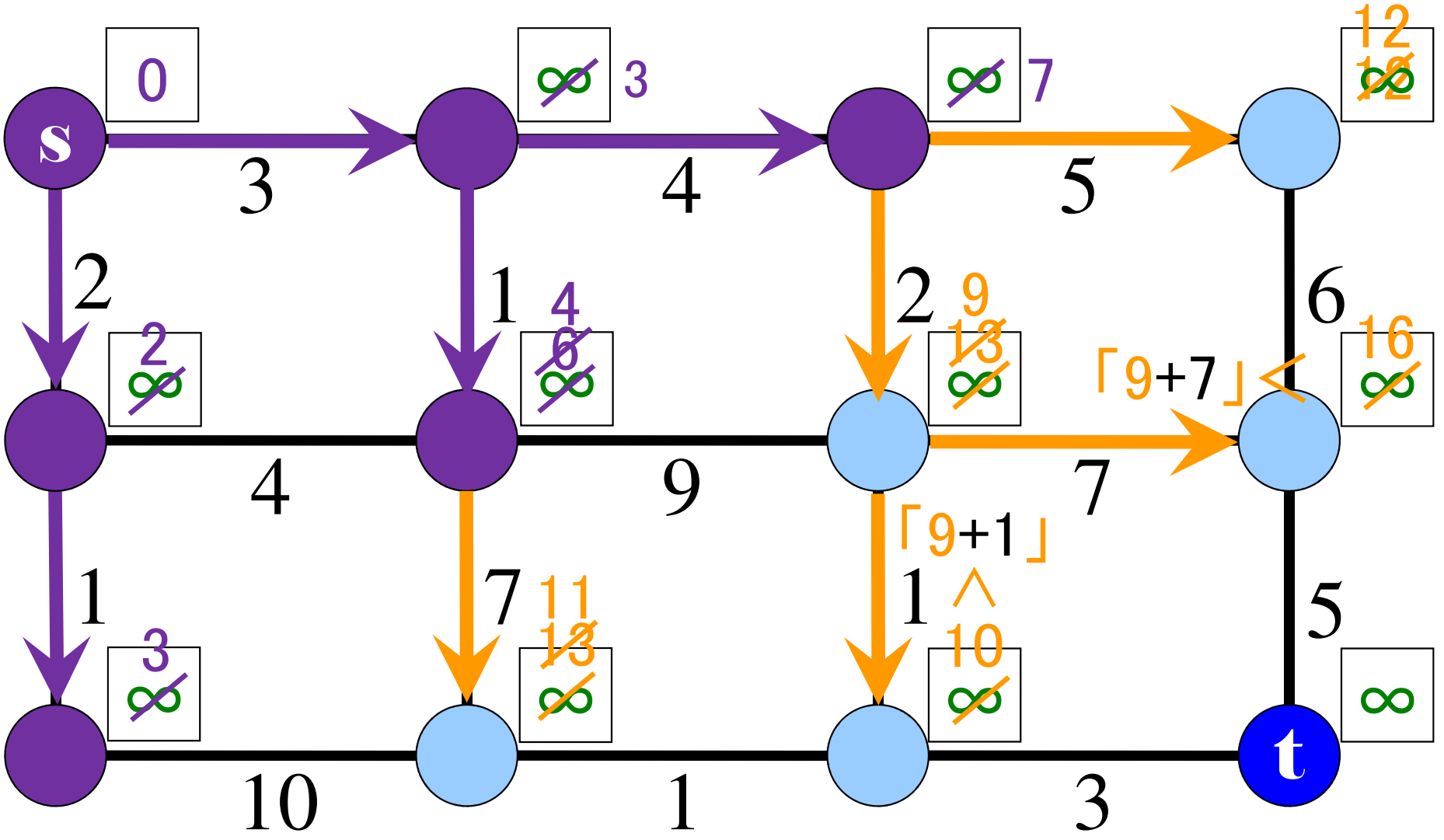
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

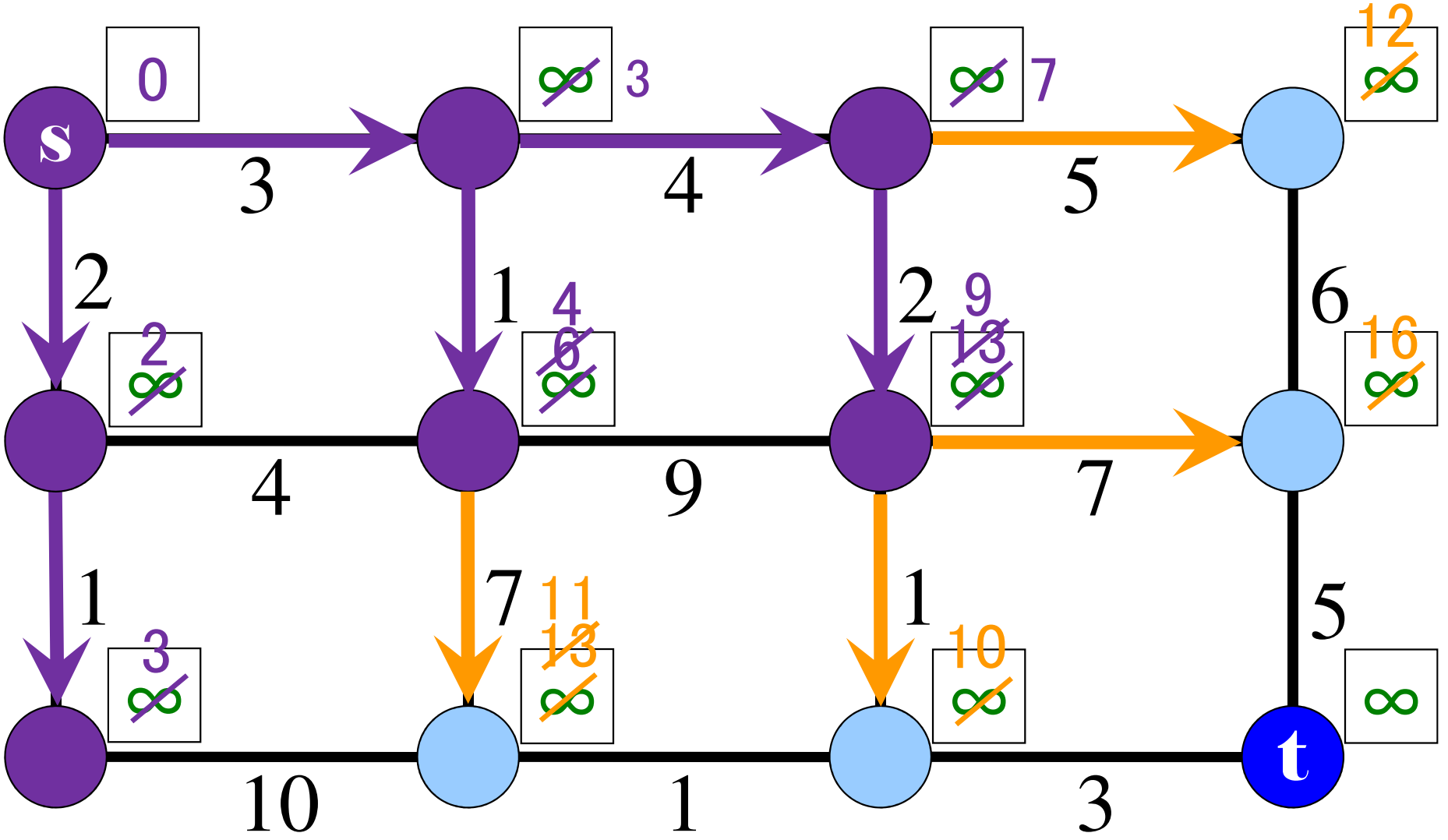


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

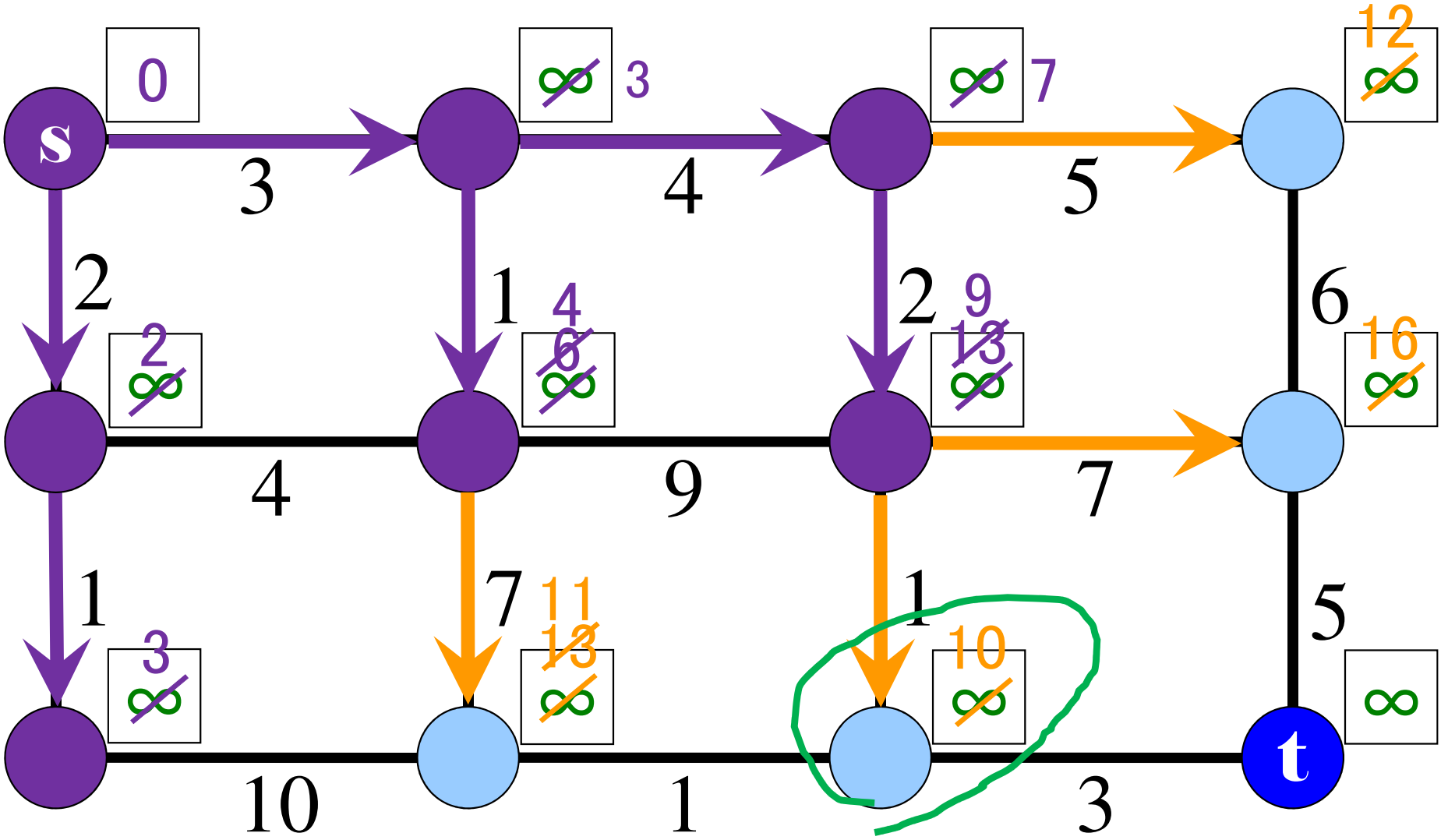
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



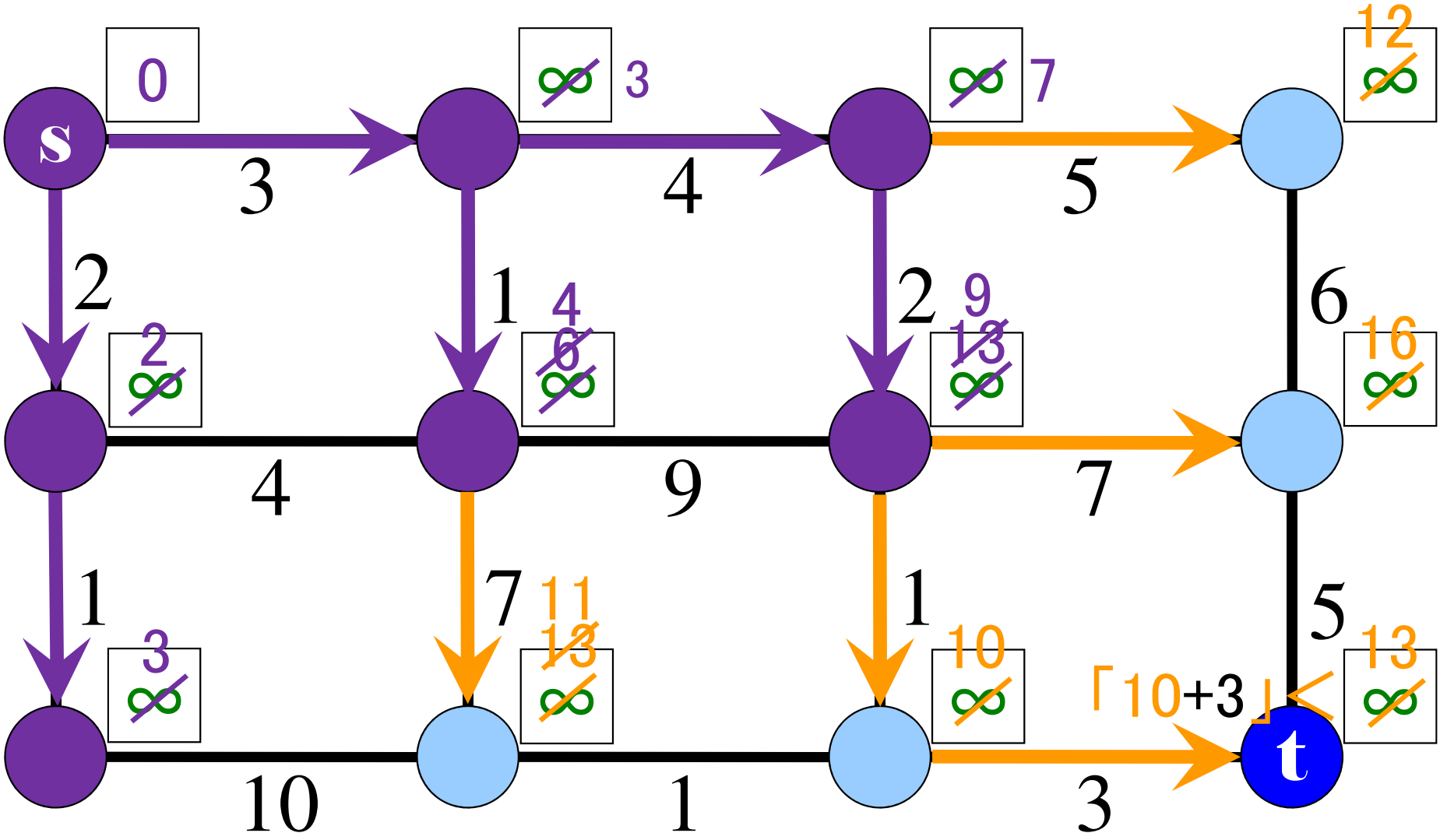
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

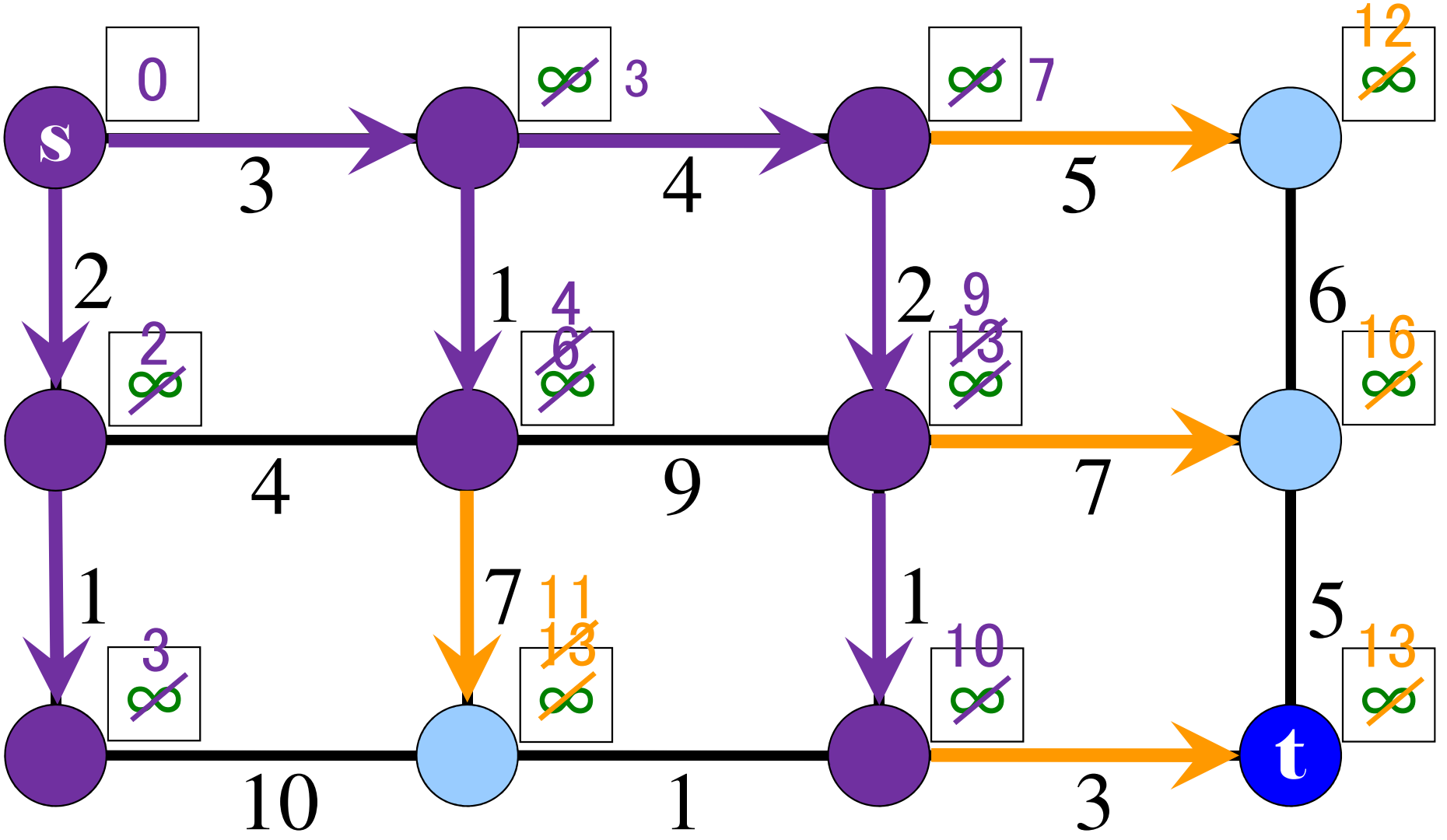


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

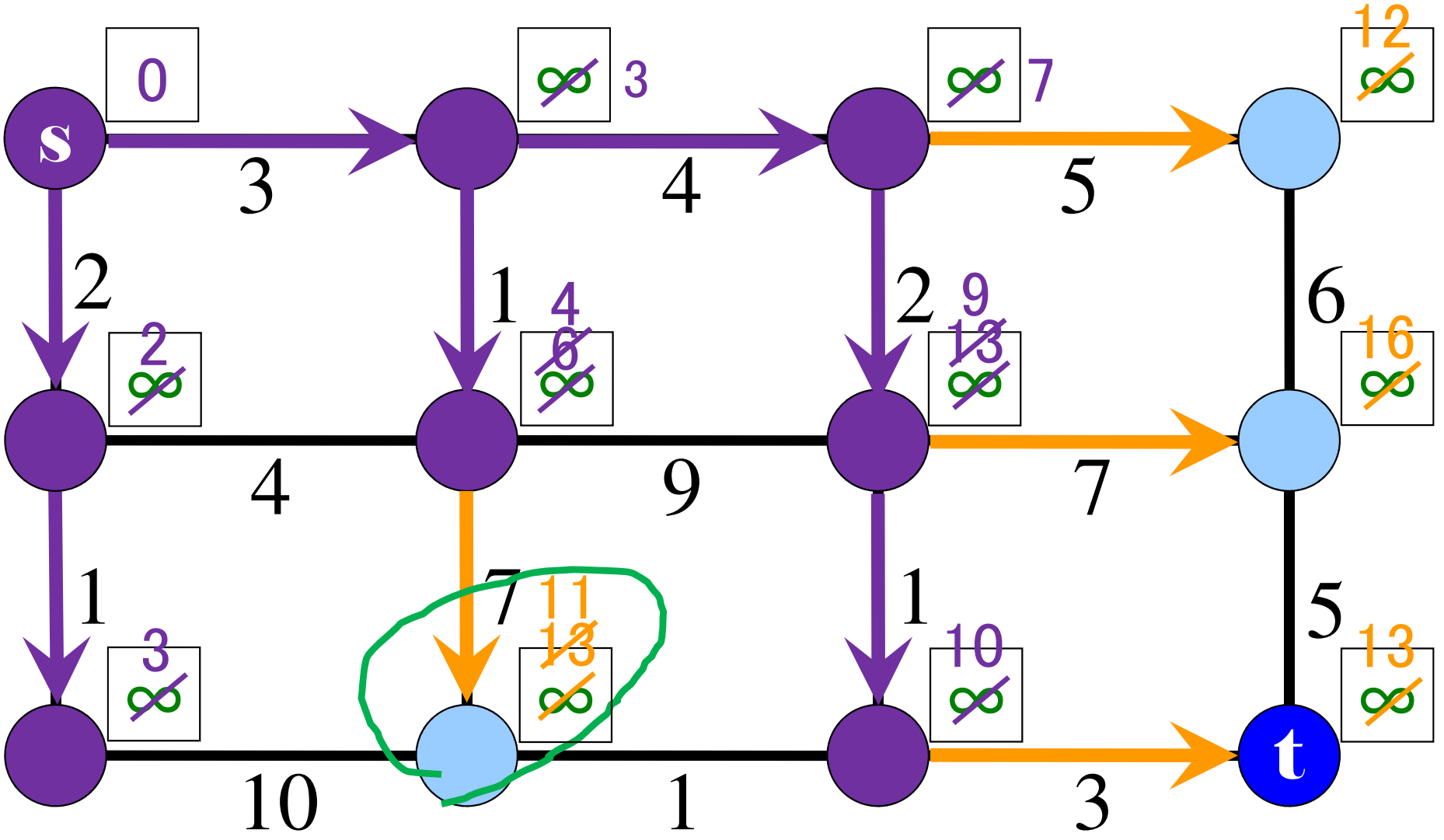
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



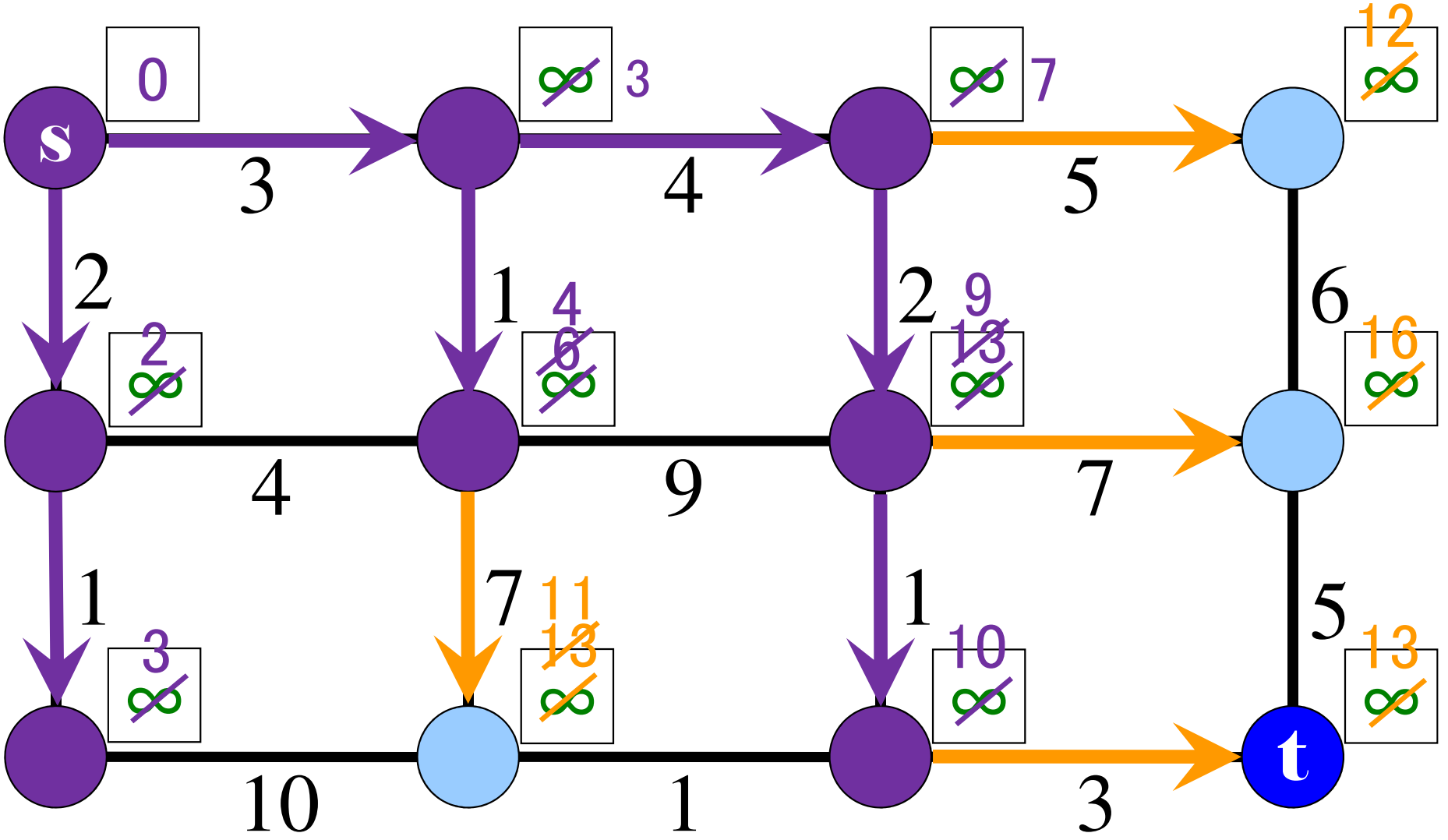
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

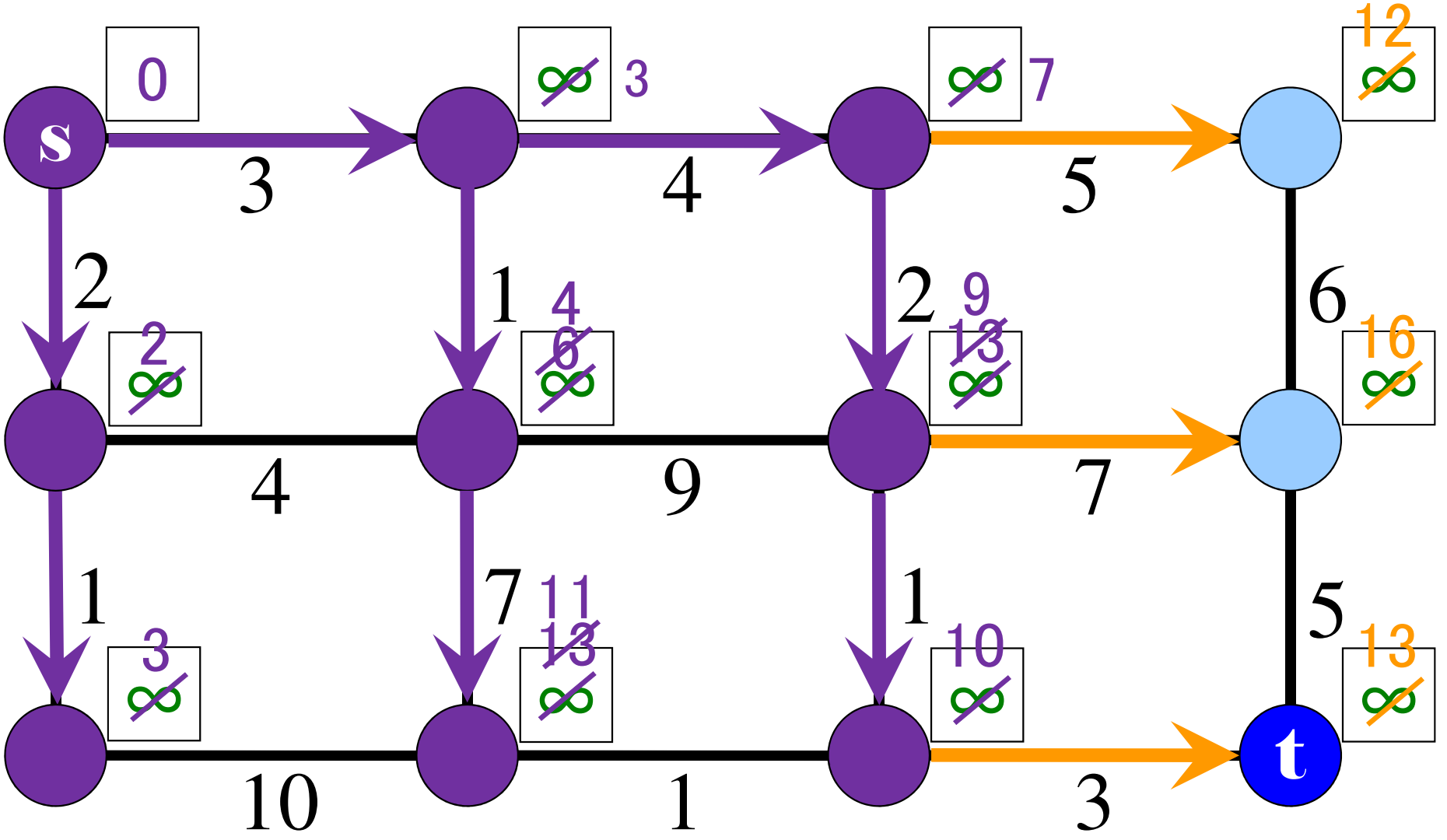


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

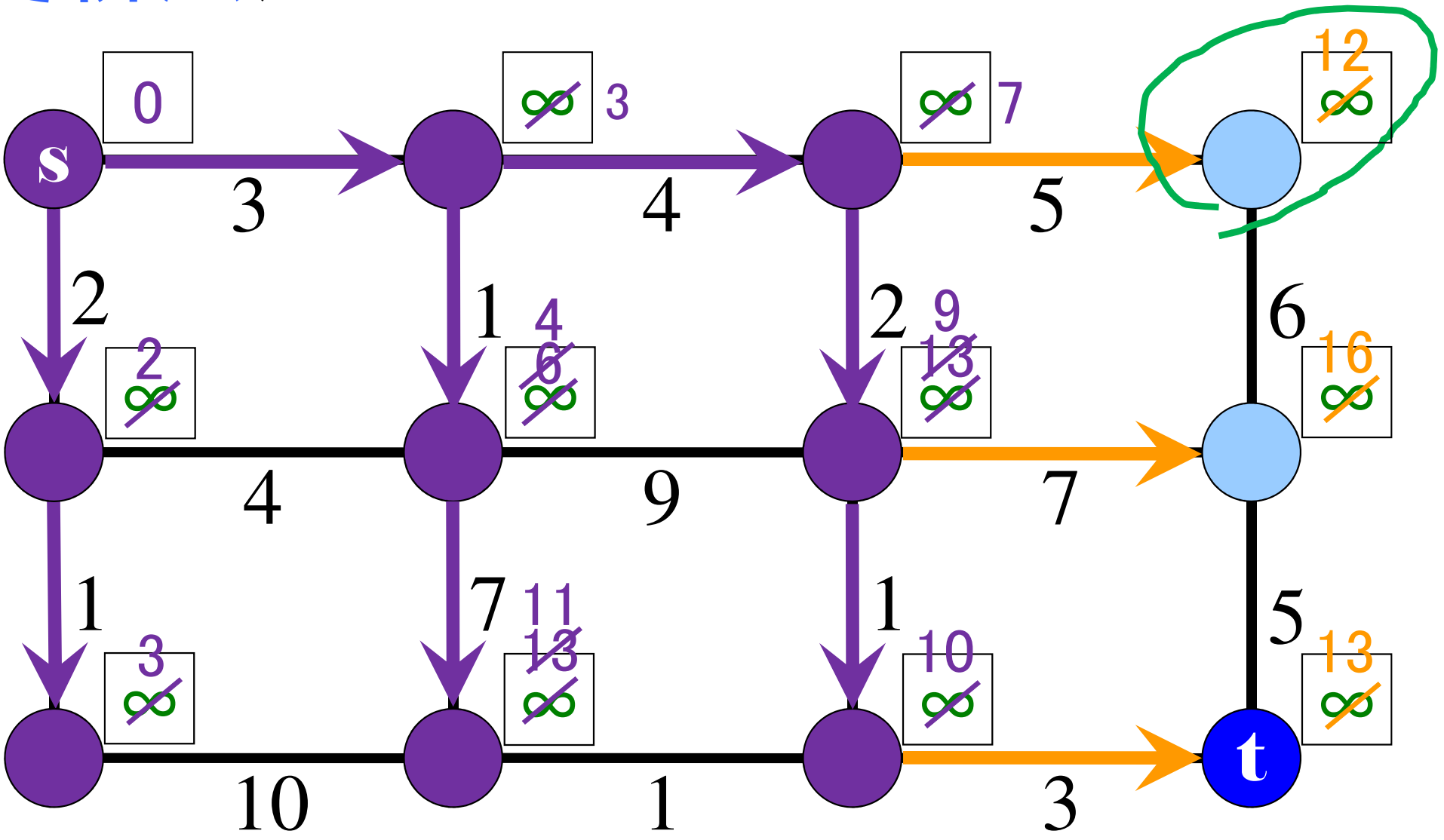
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



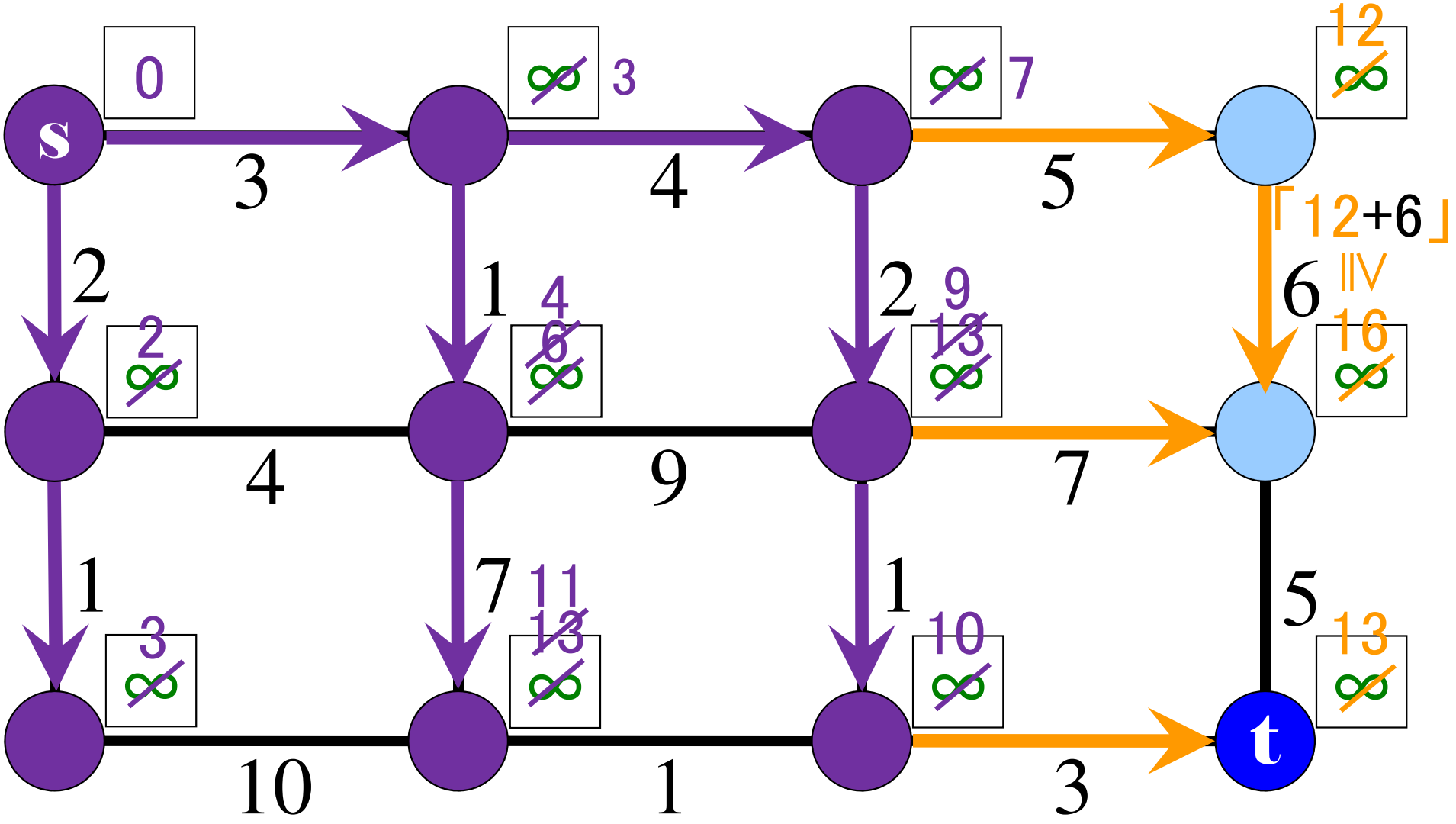
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

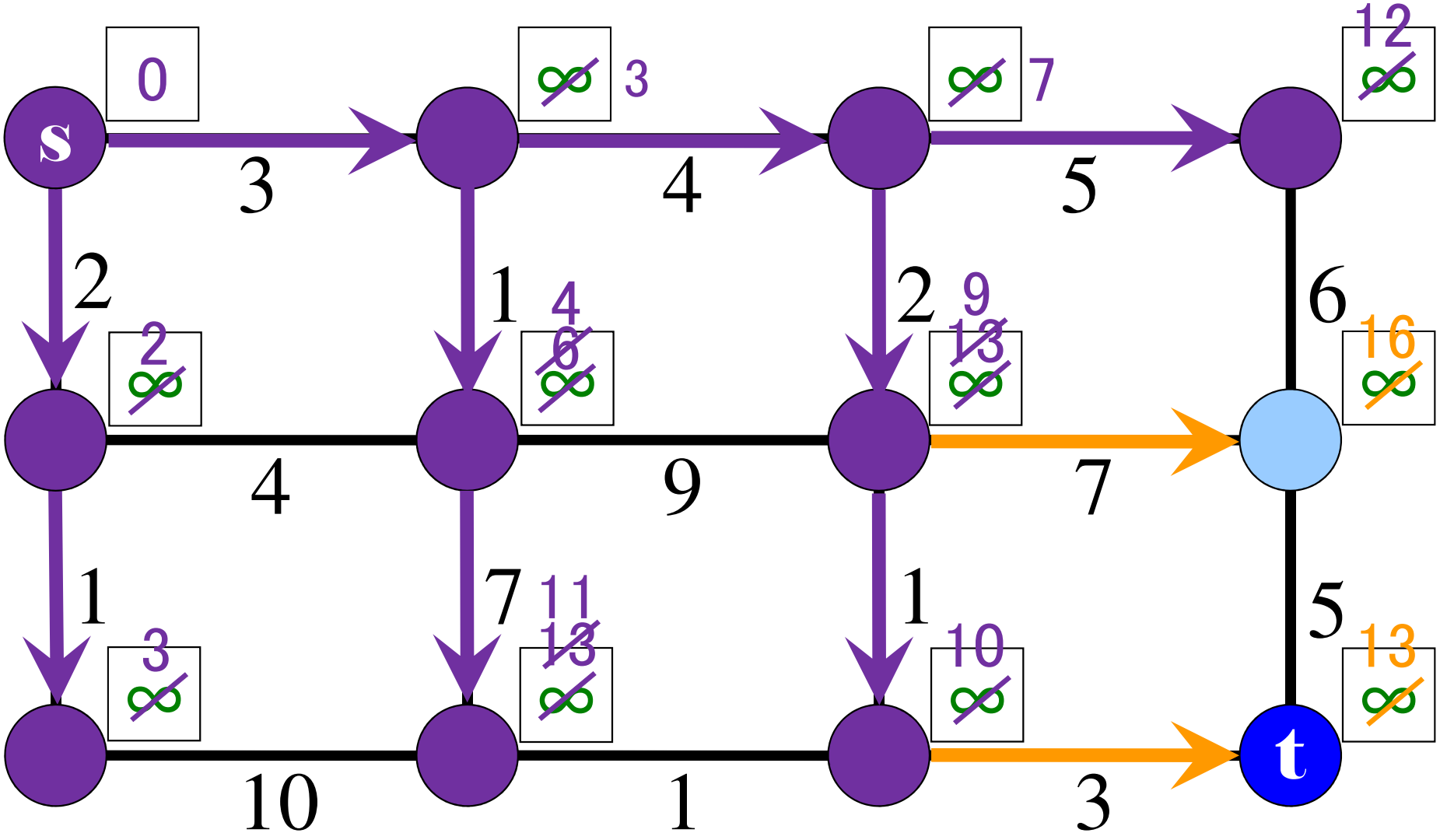


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

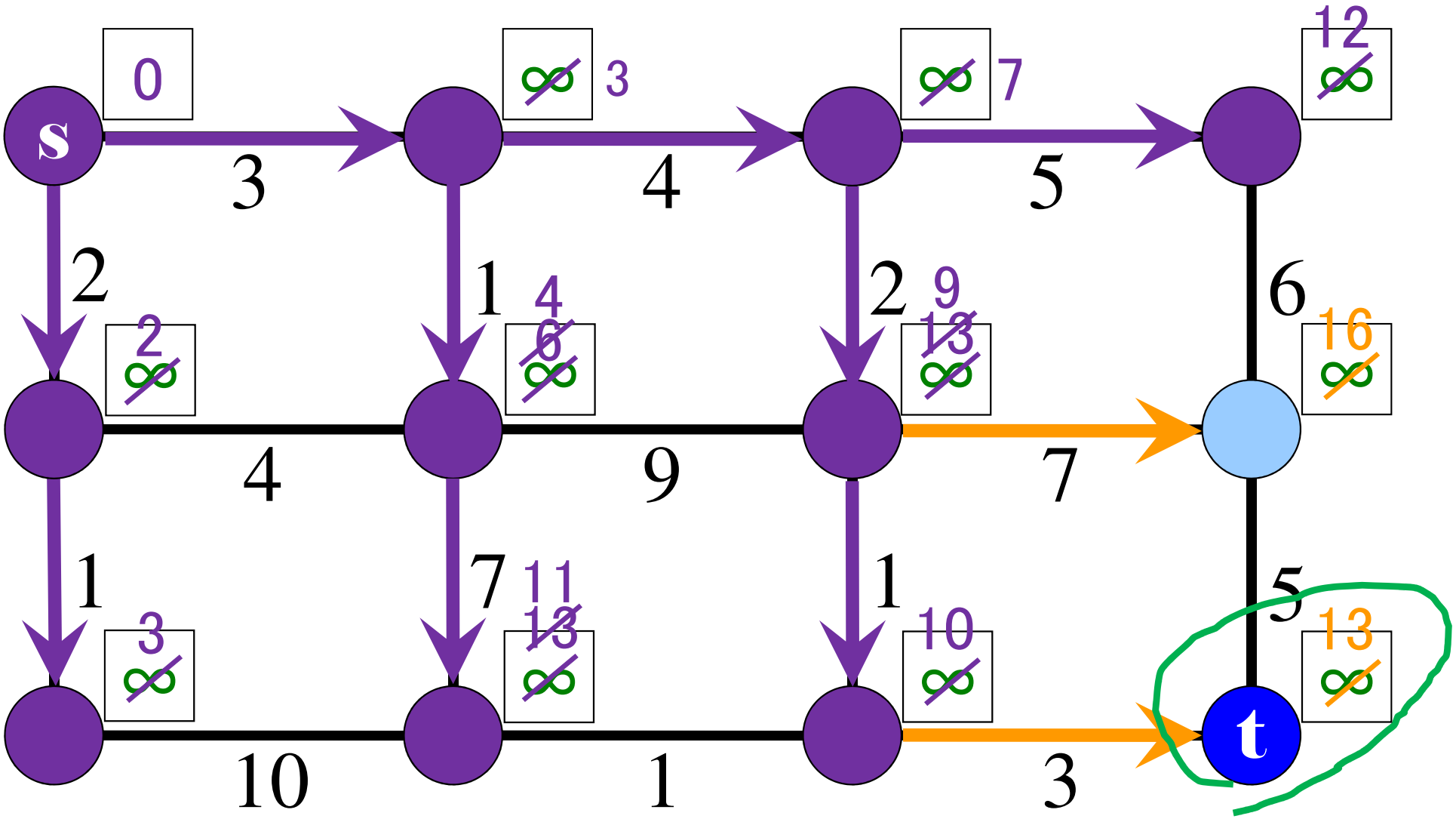
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



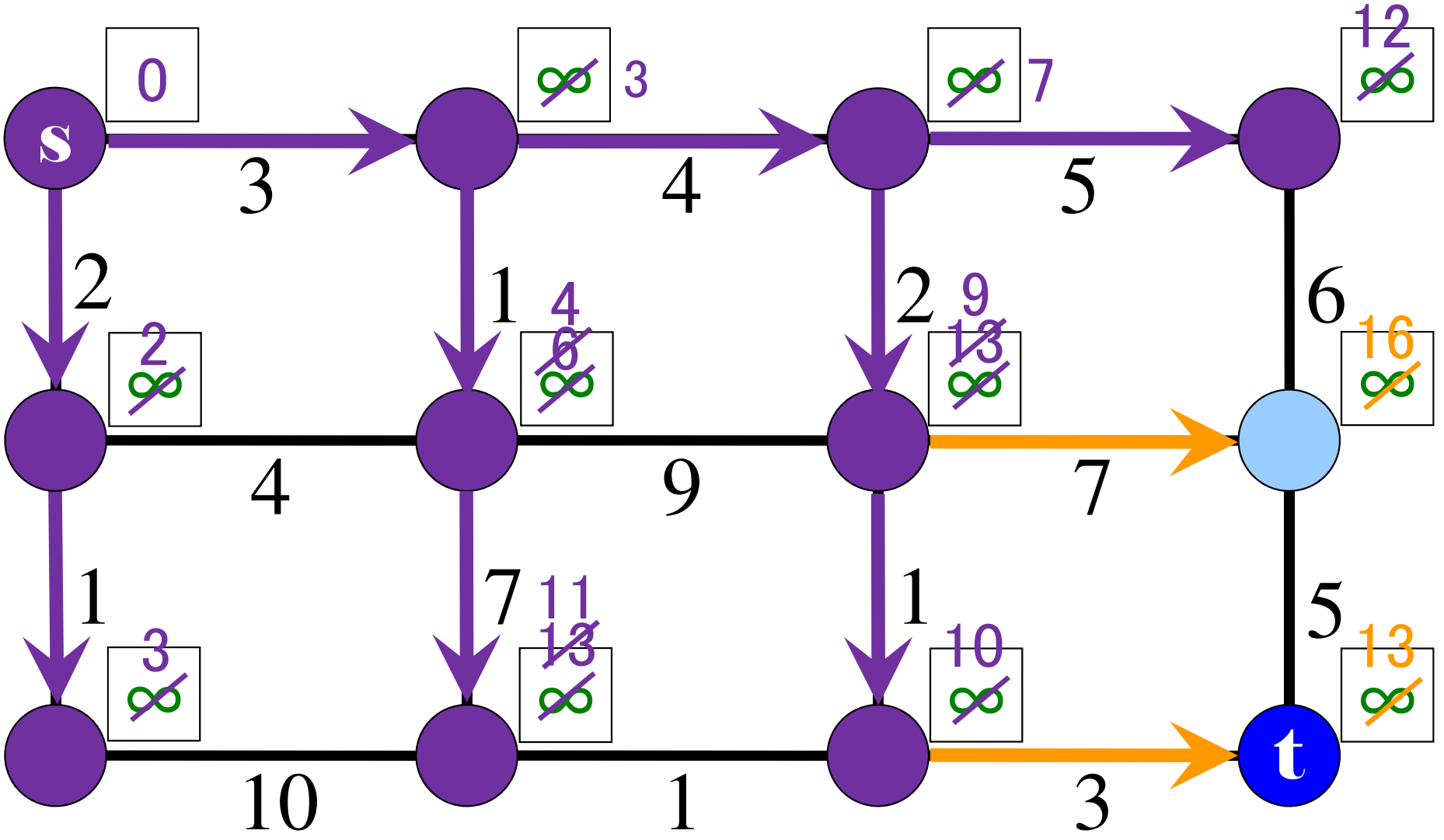
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

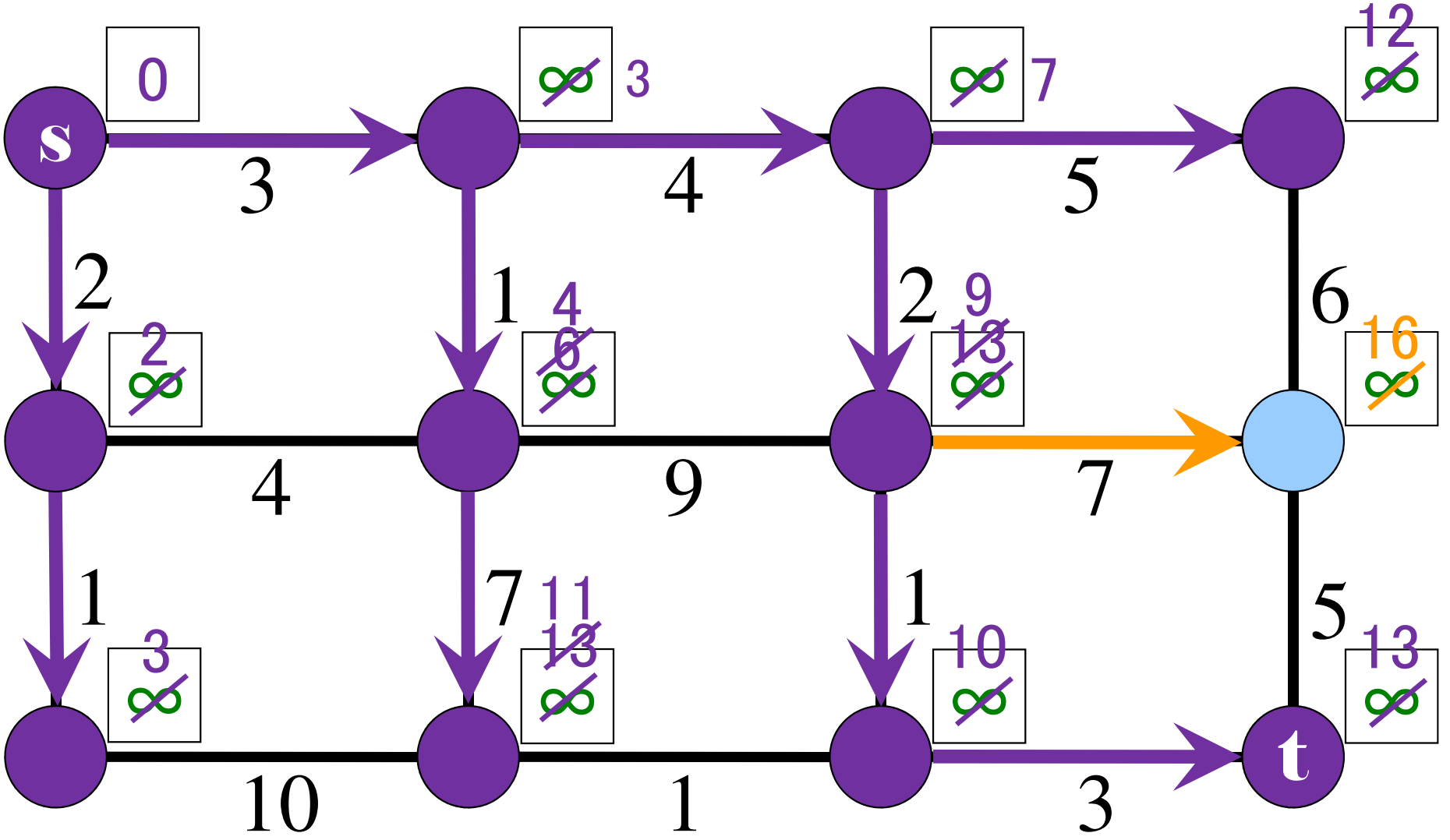


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

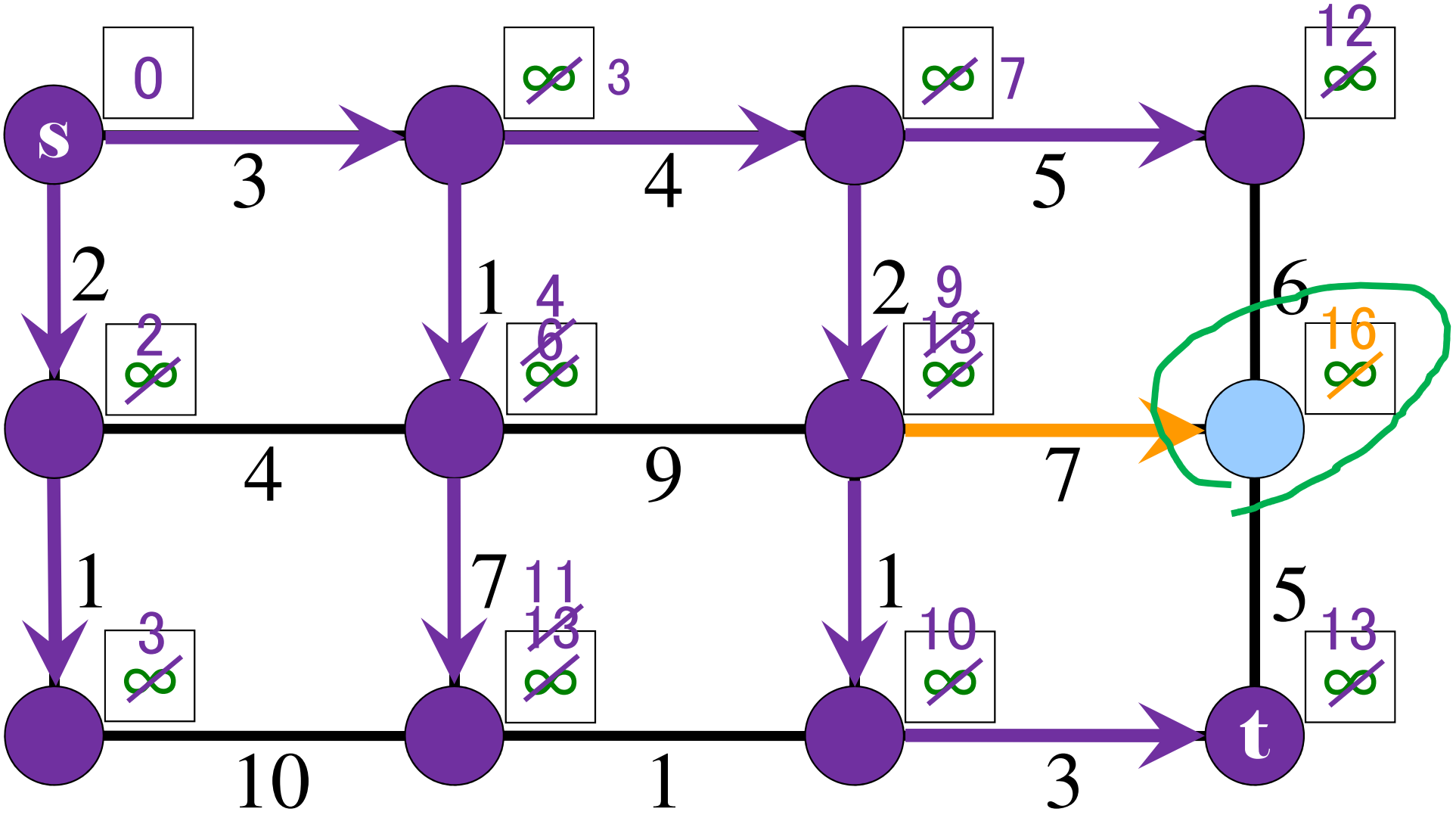
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



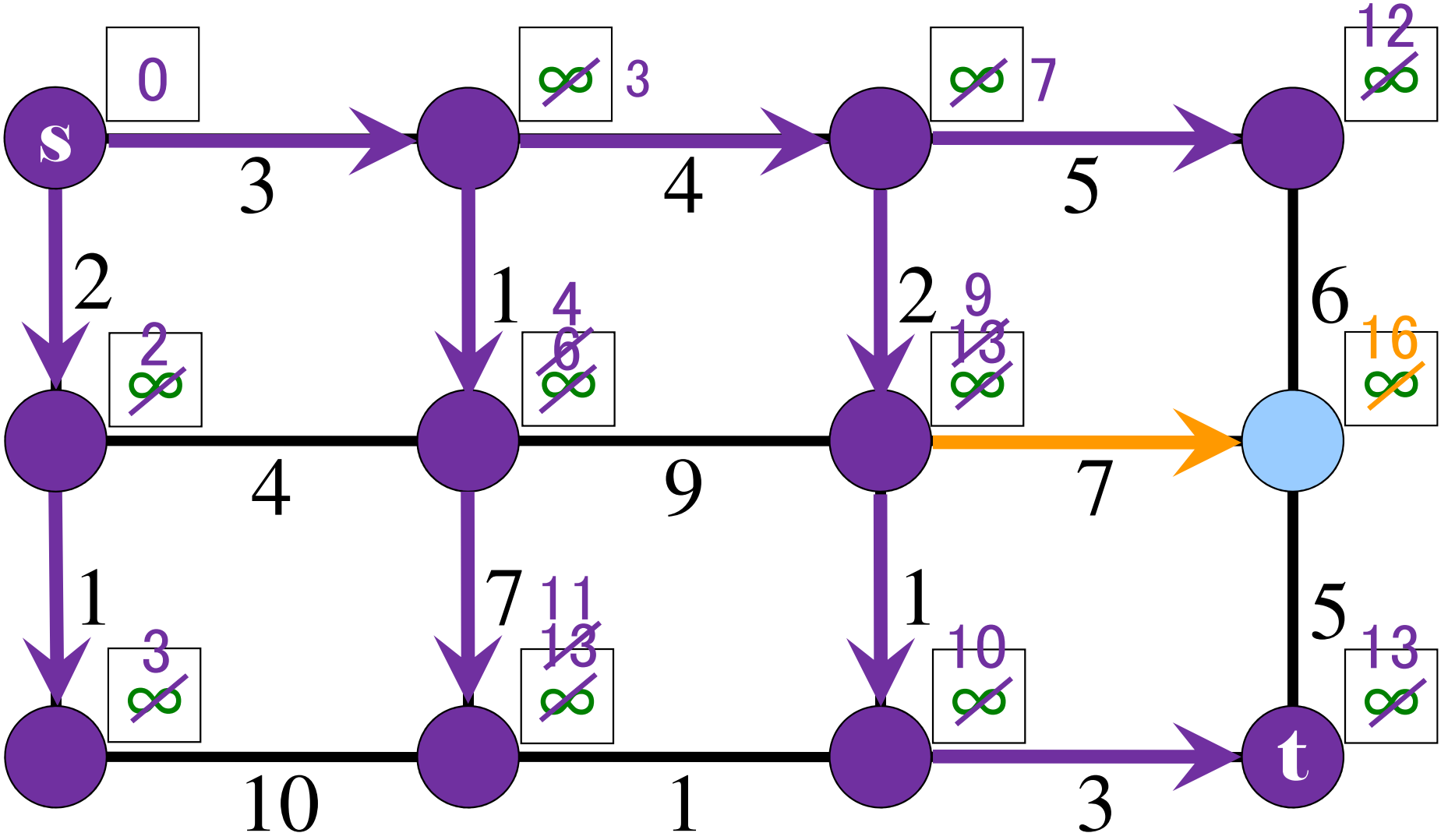
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

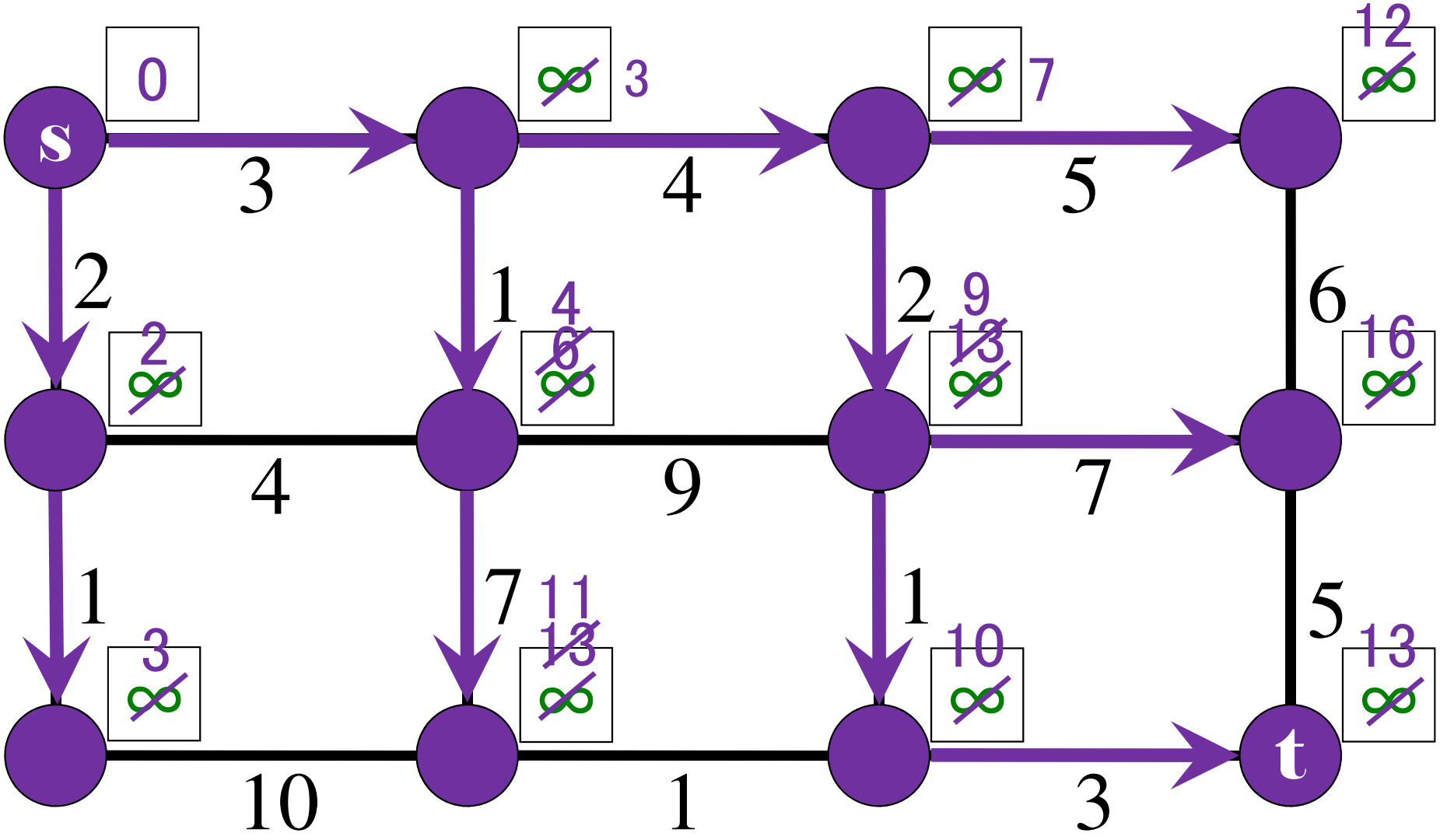


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

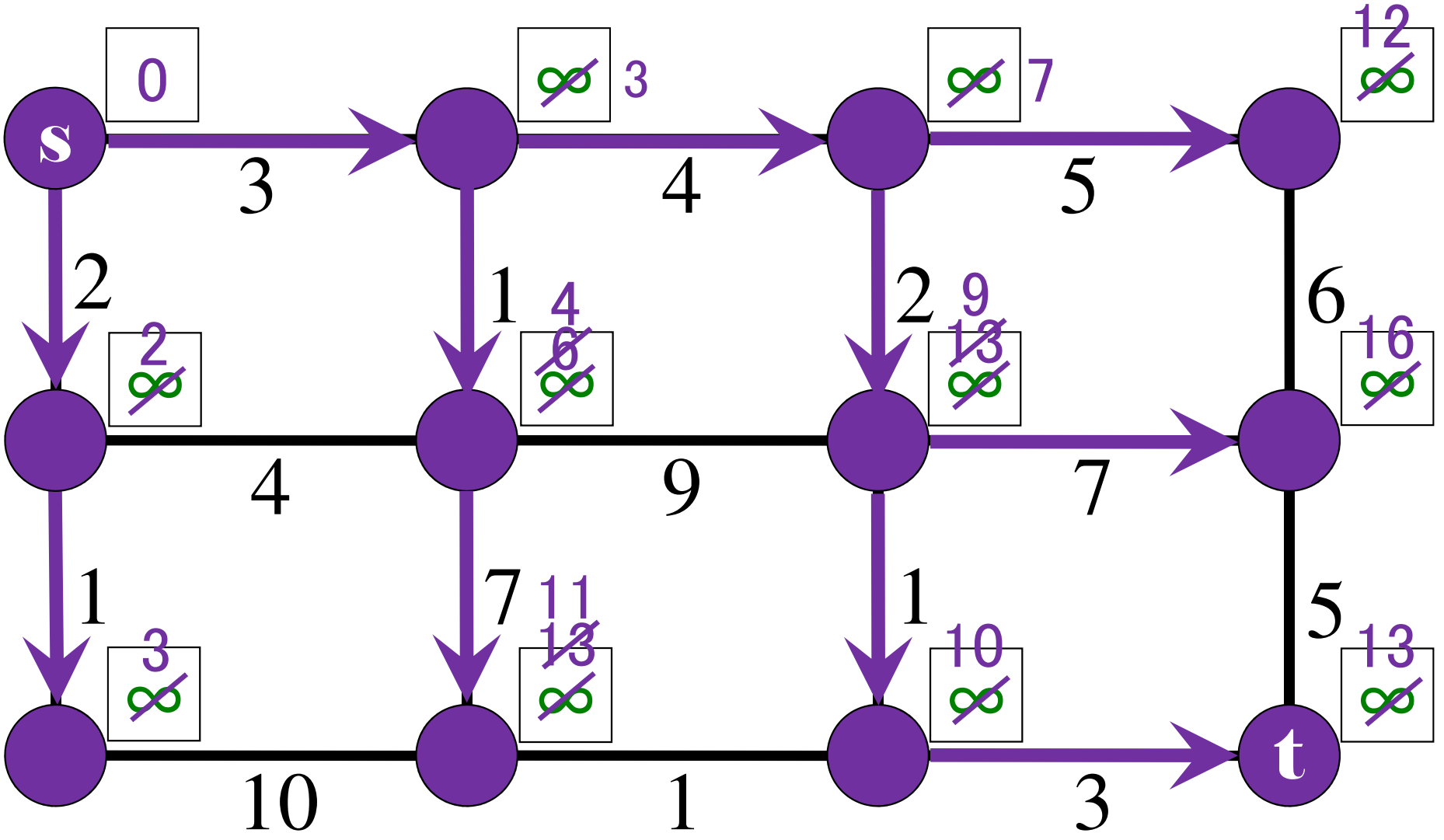
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



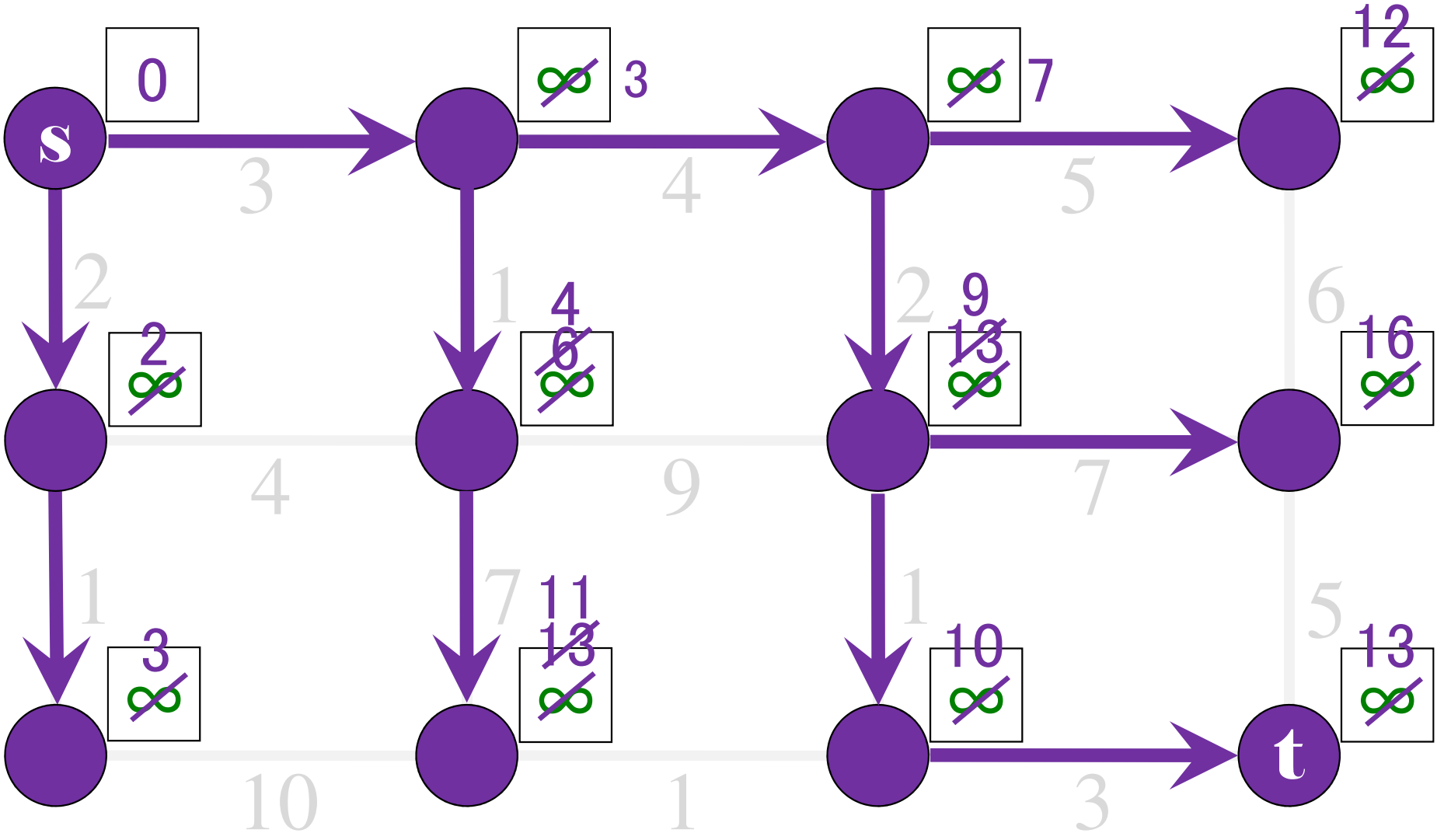
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



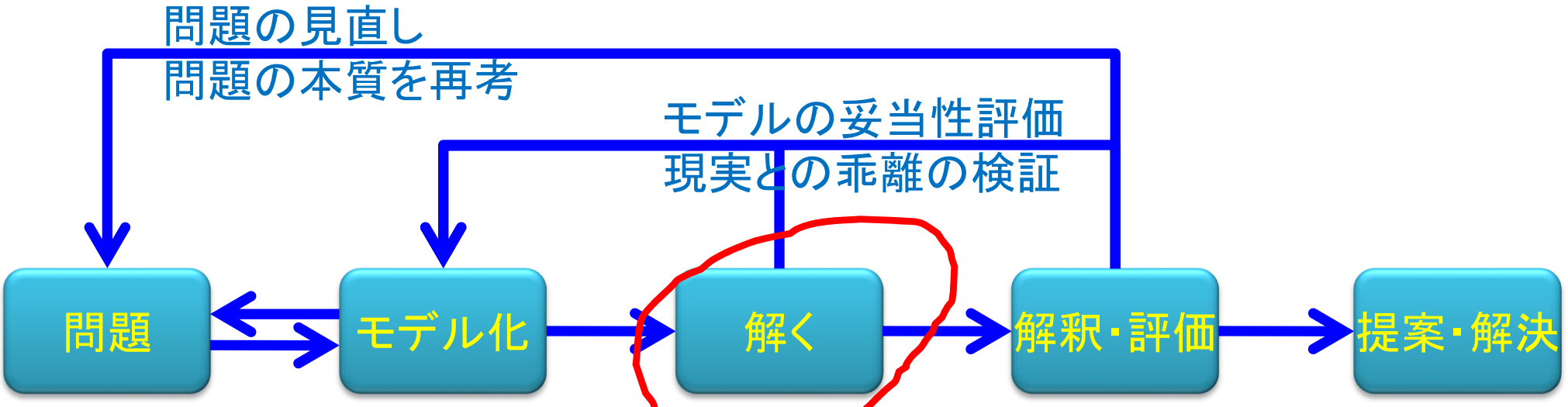
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

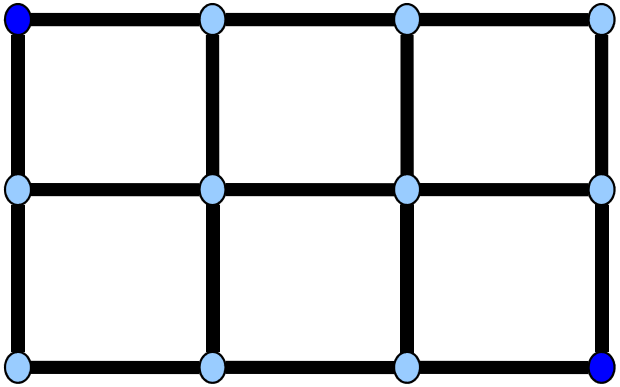
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

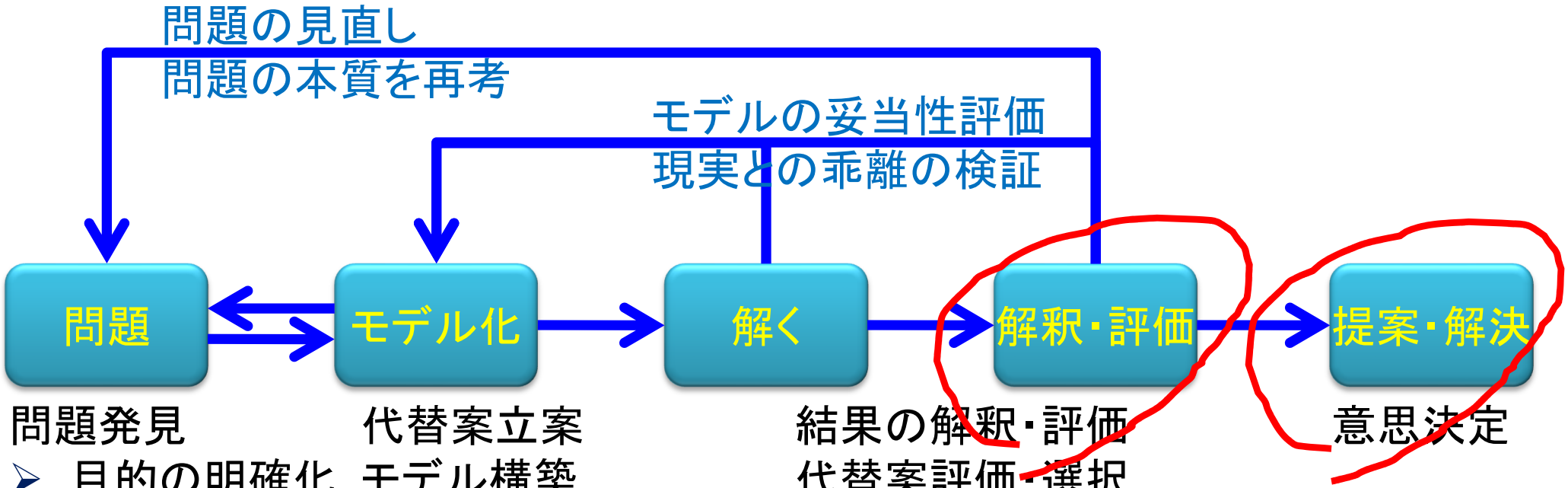
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

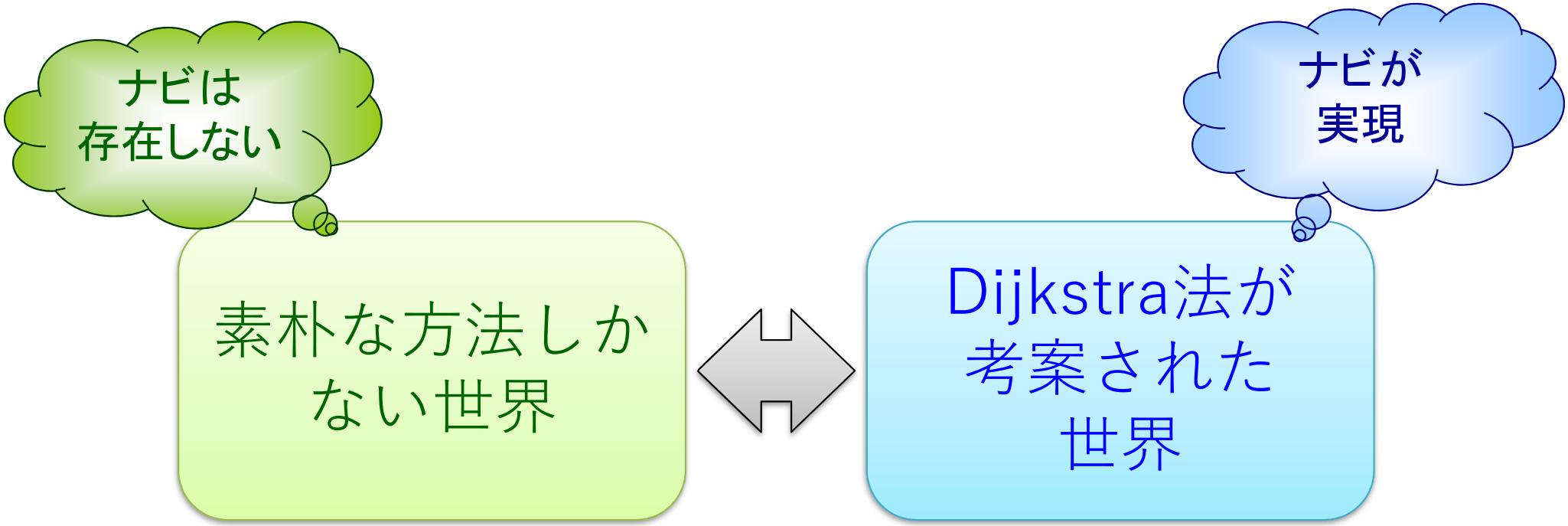
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

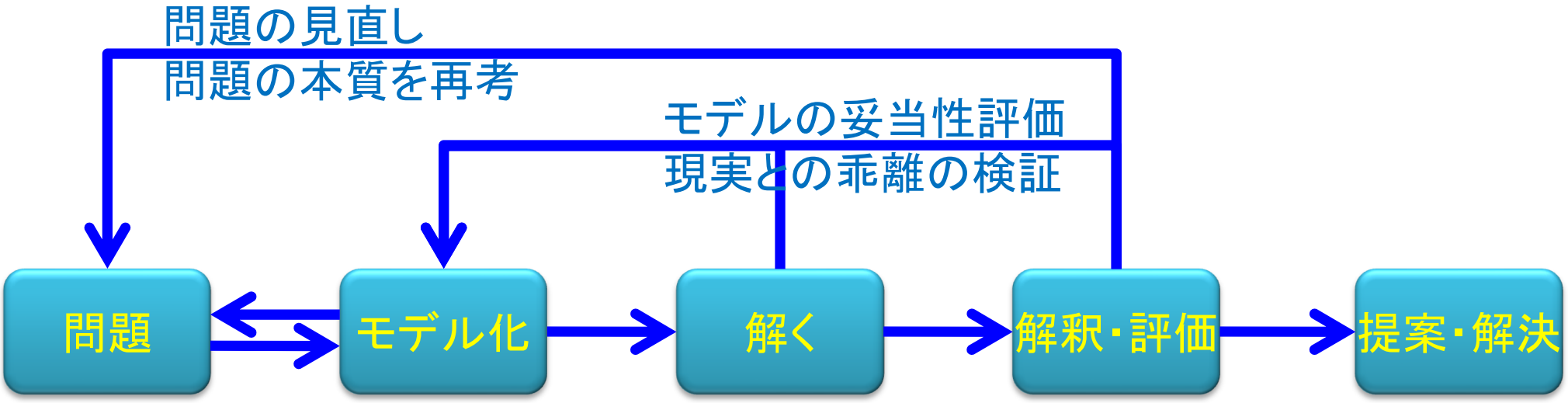
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



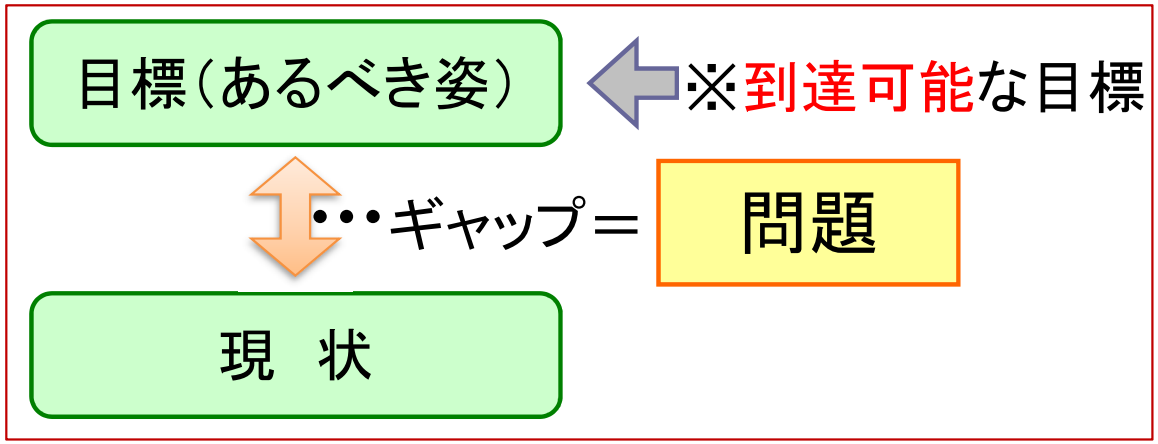
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

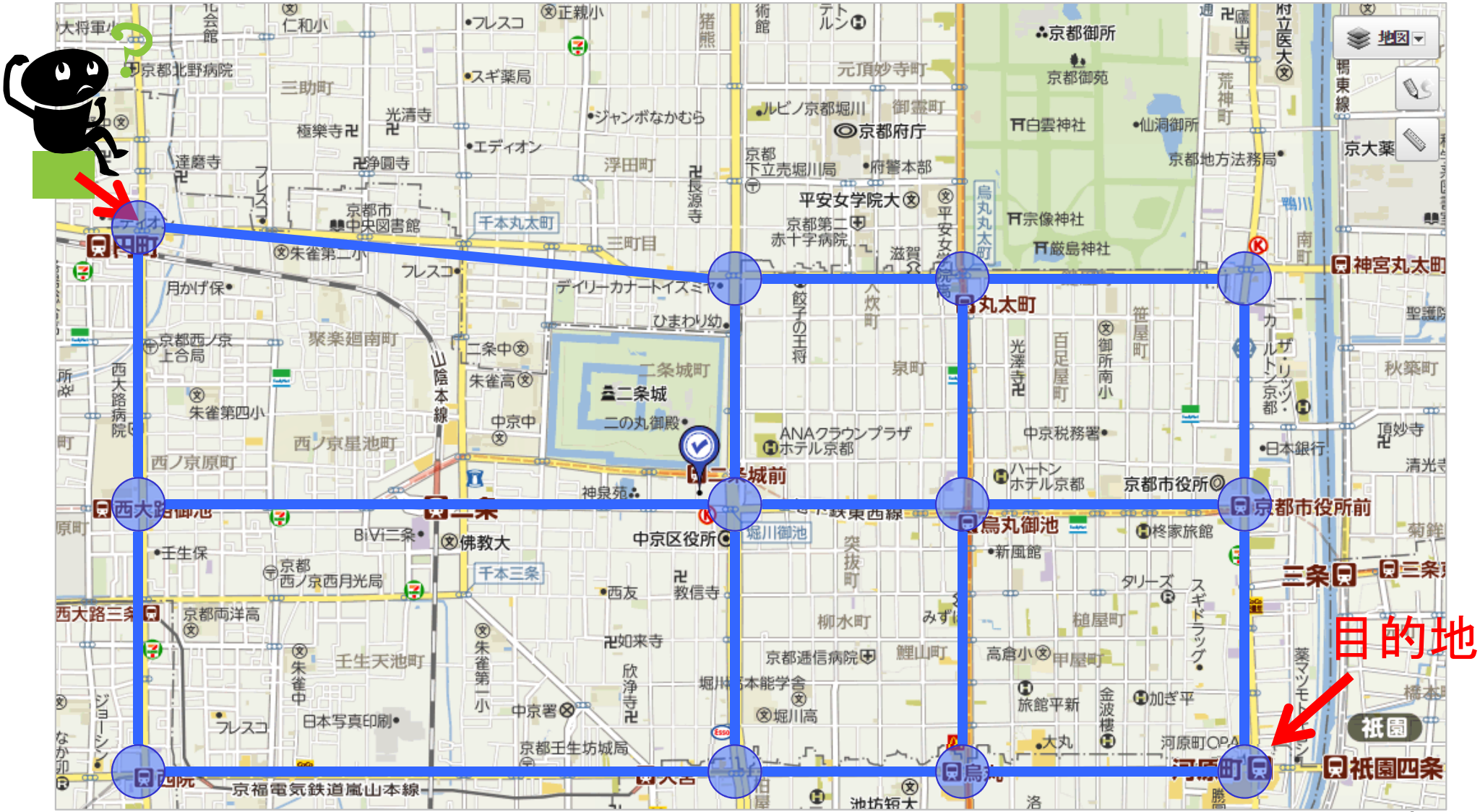
意思決定

問題の定義



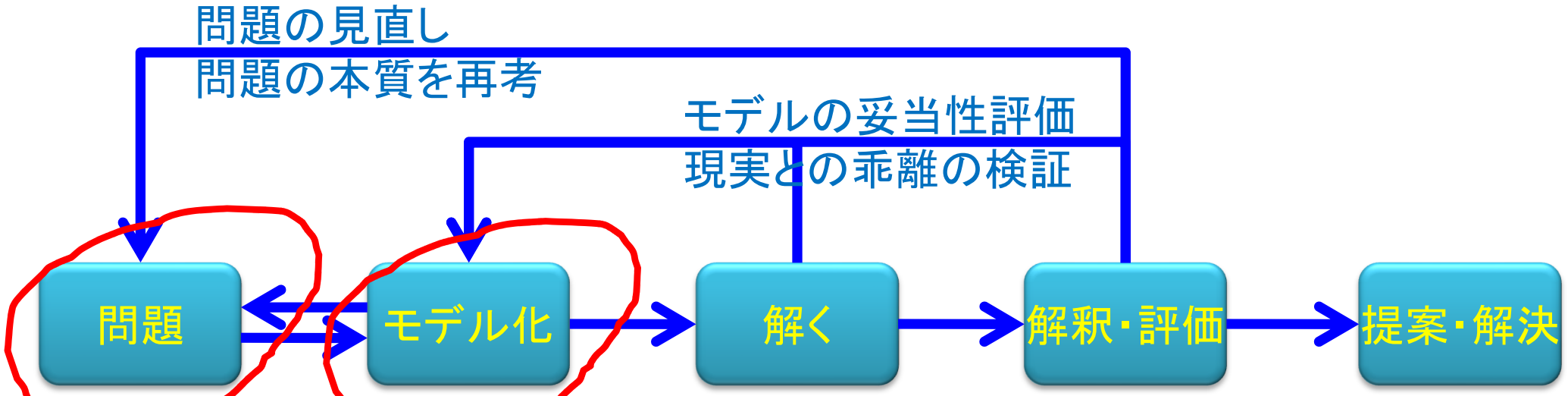
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

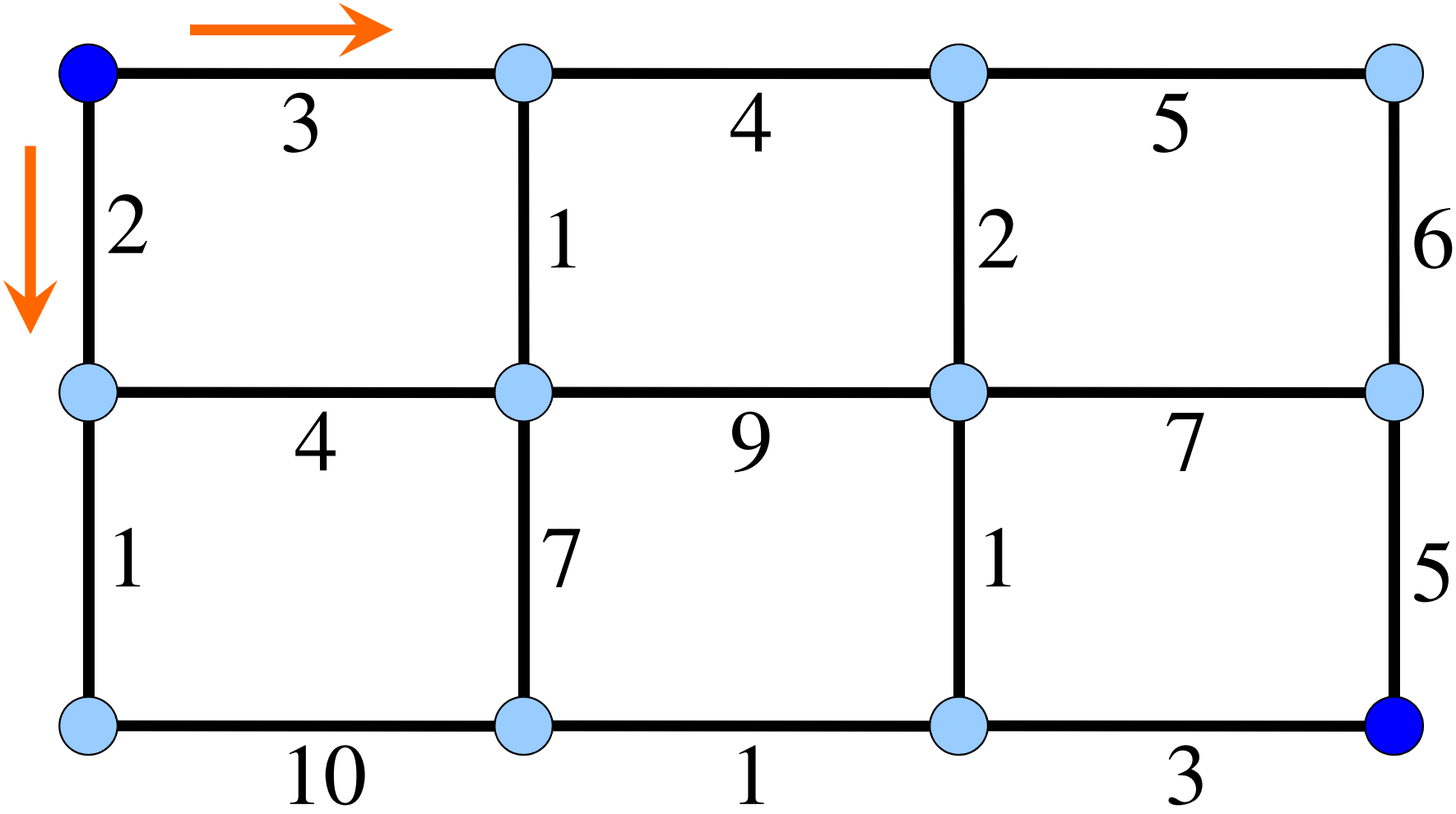
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

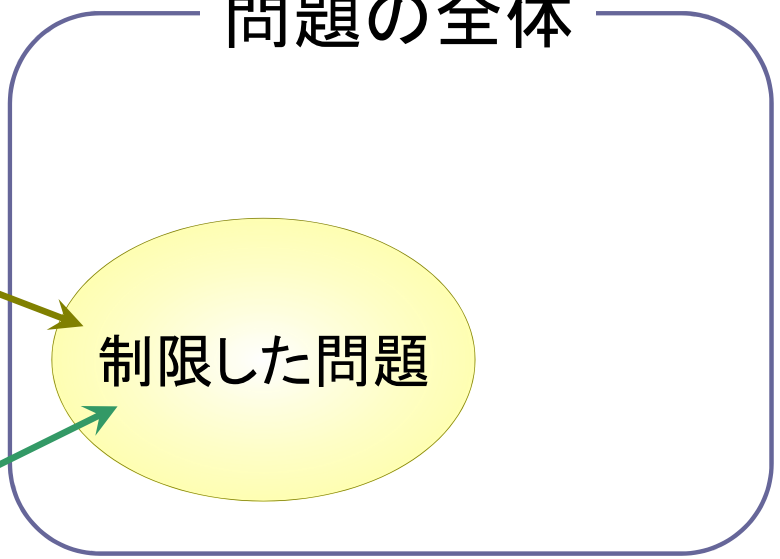


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

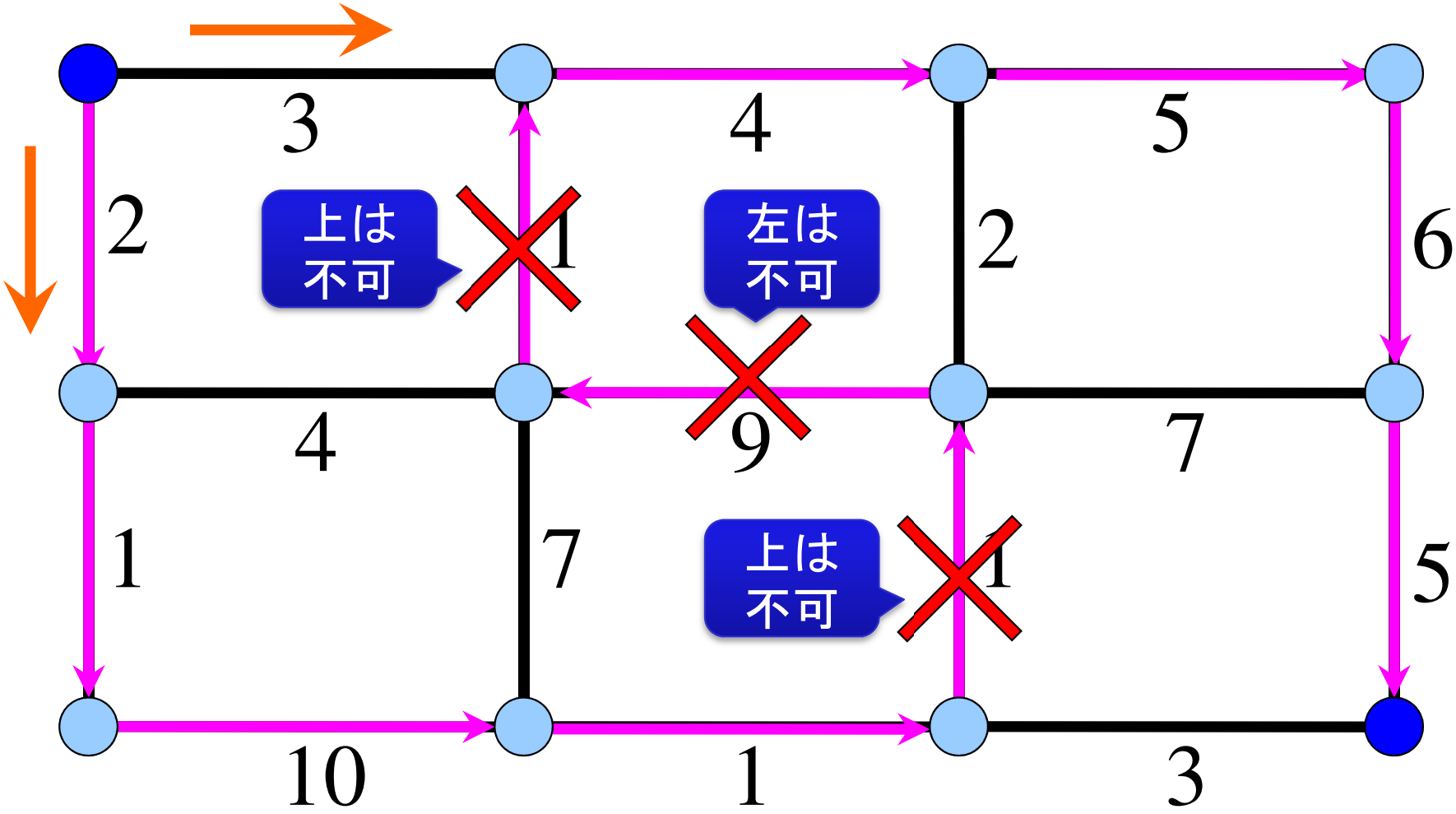
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

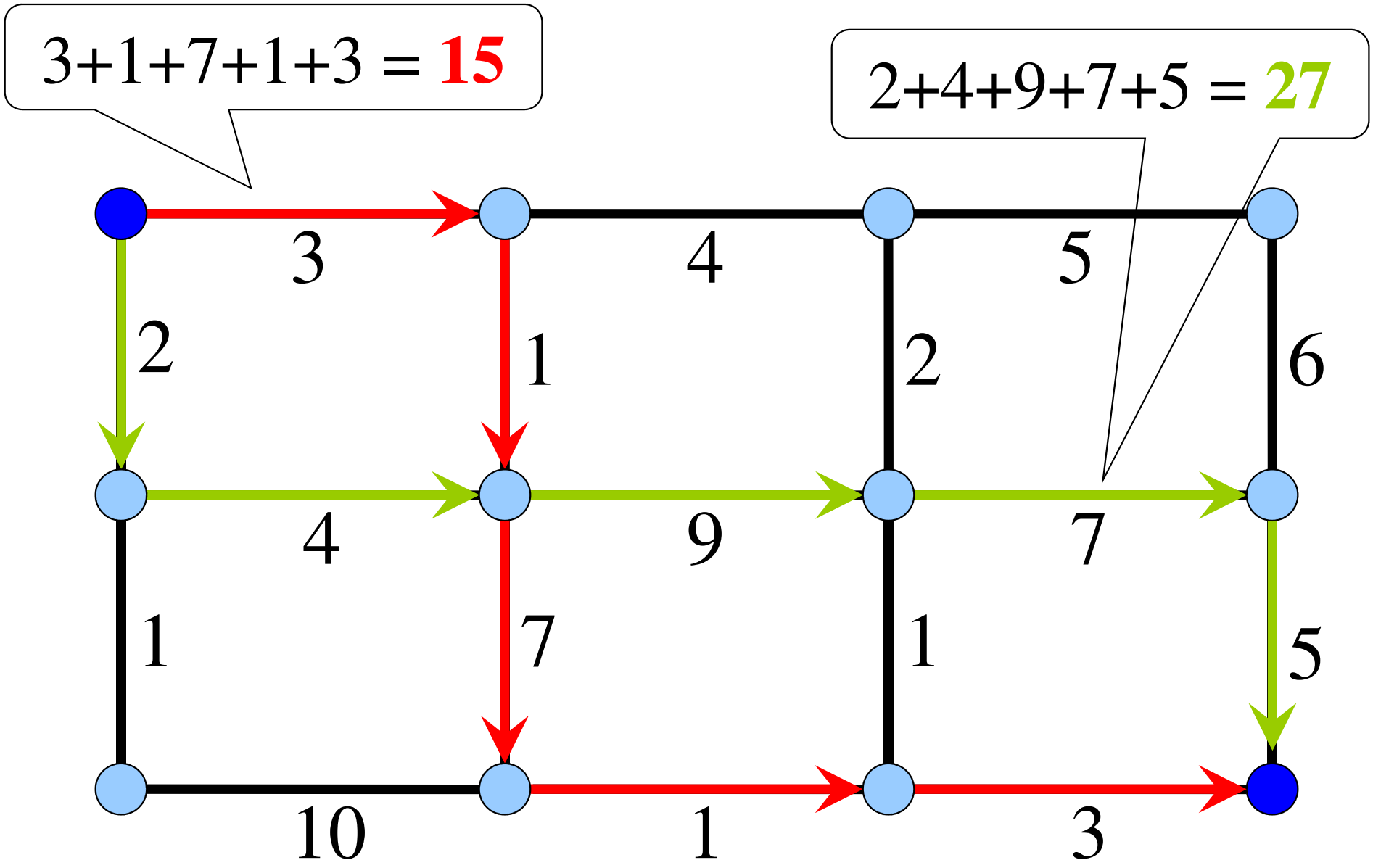
難しいなら易しくすればいいのさ！

制限した問題

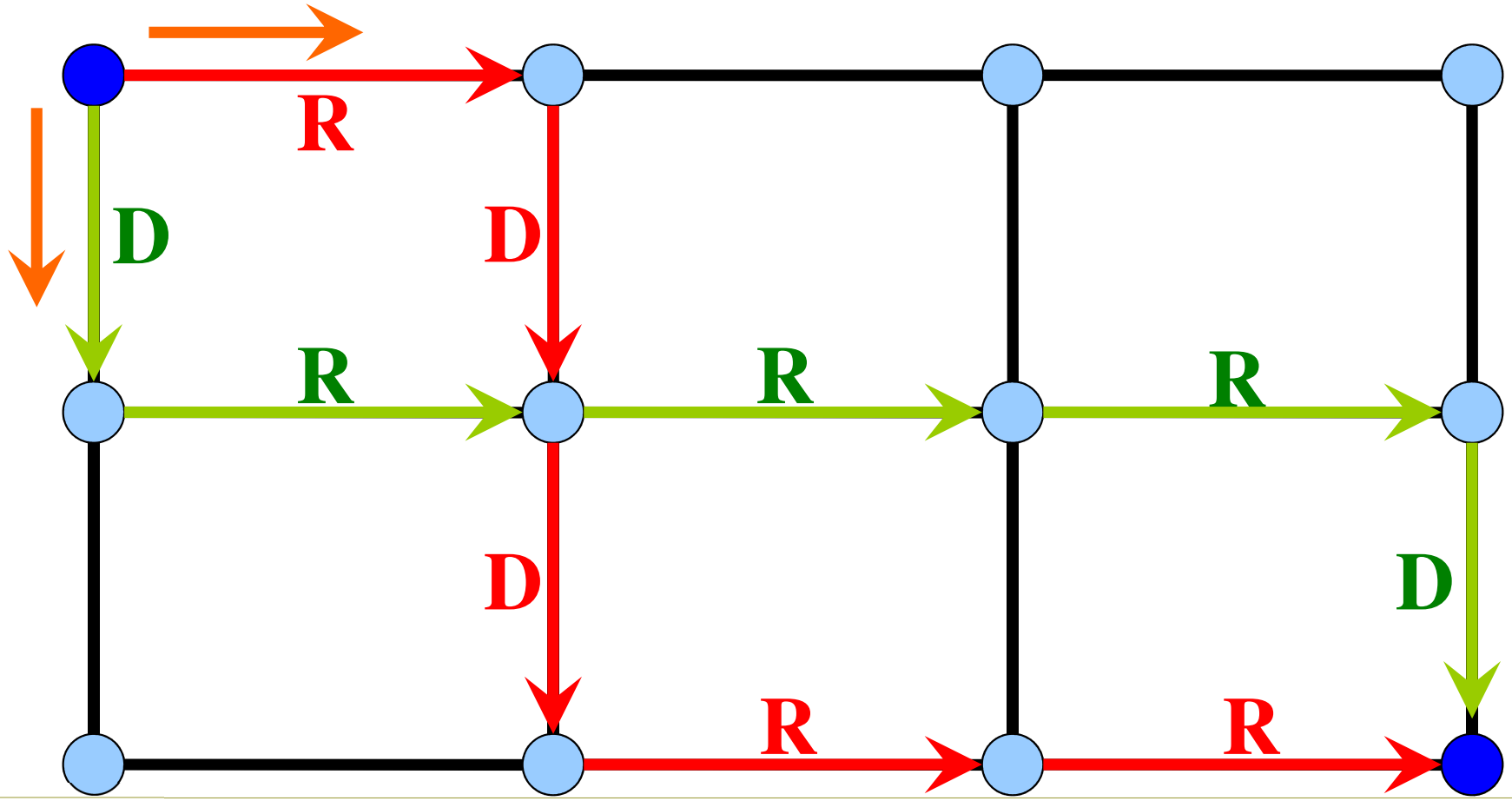
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



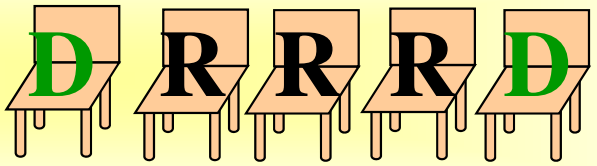
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



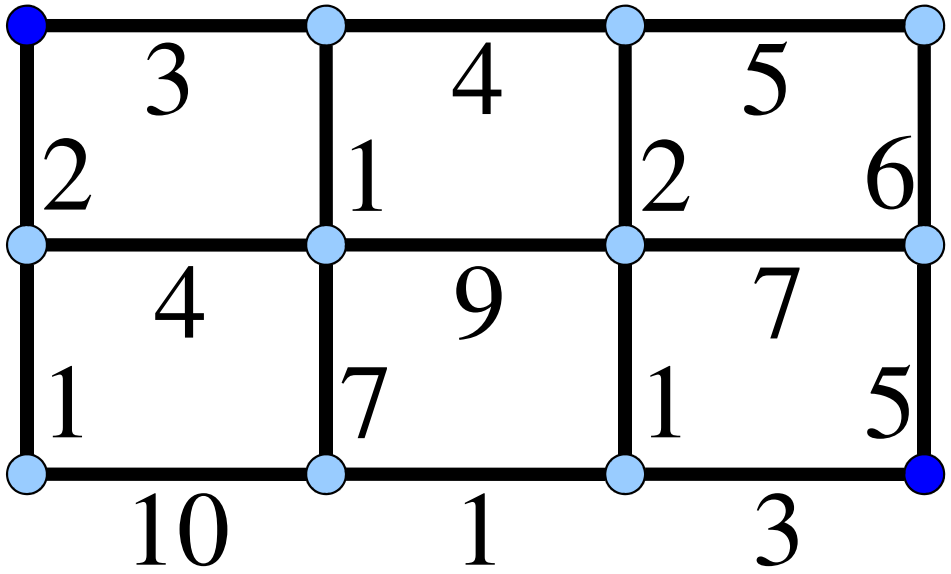
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

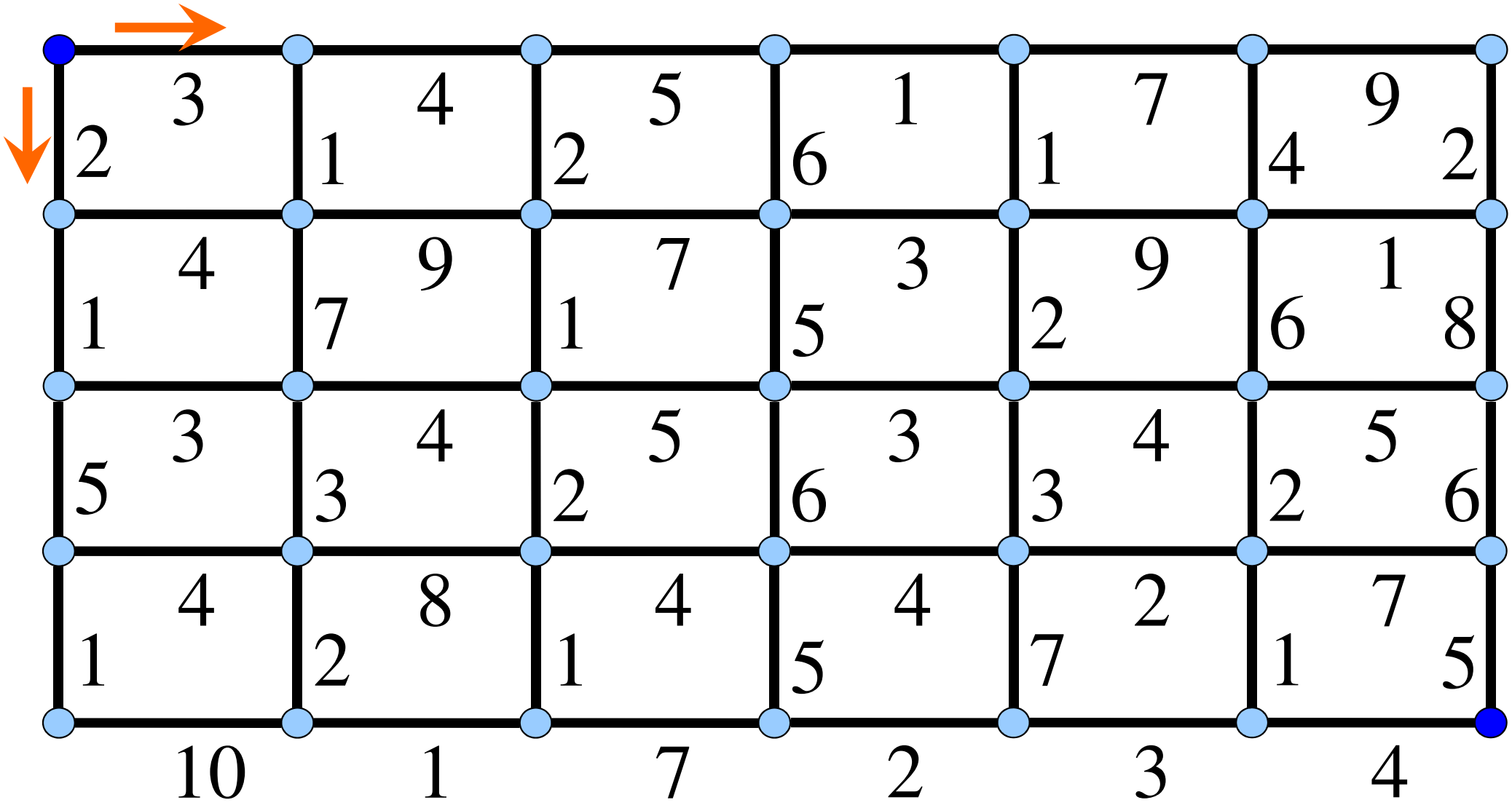
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

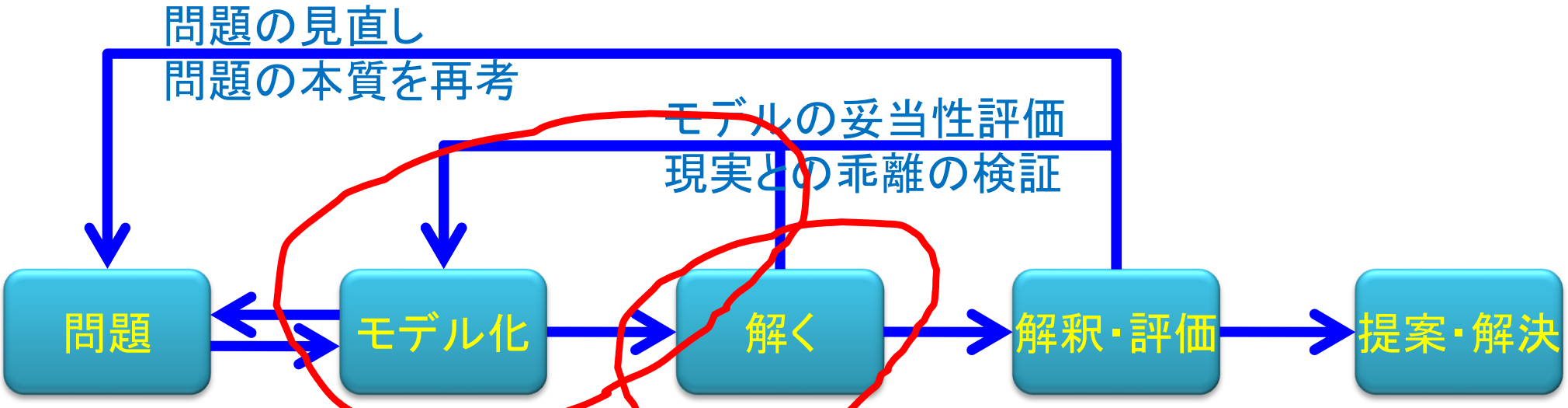
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

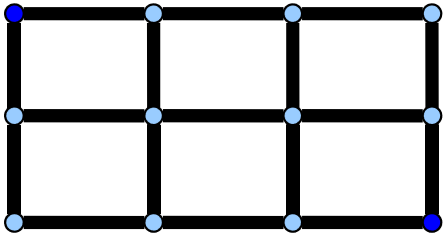
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

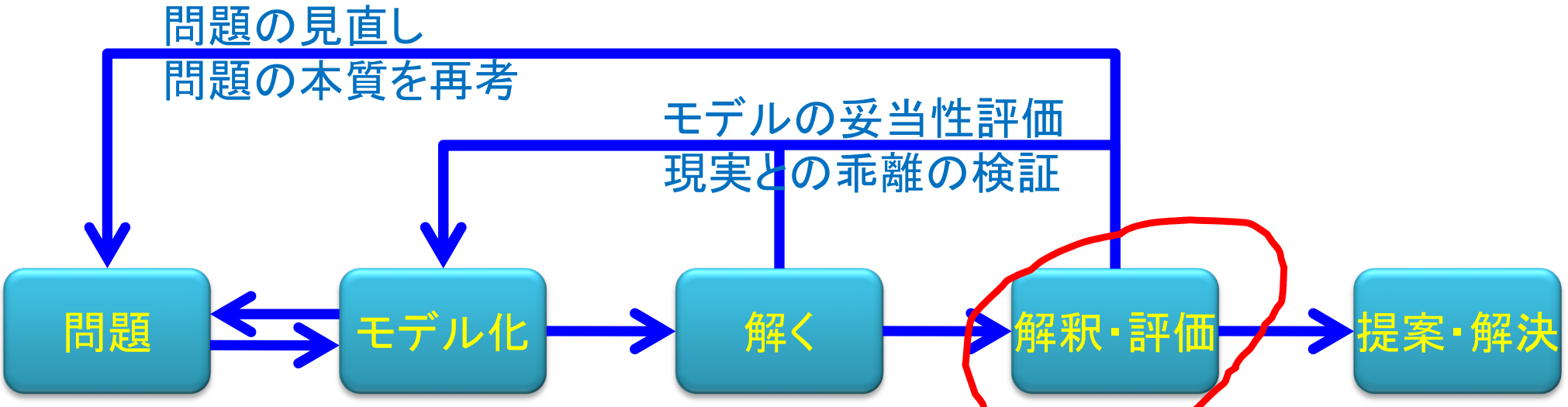
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

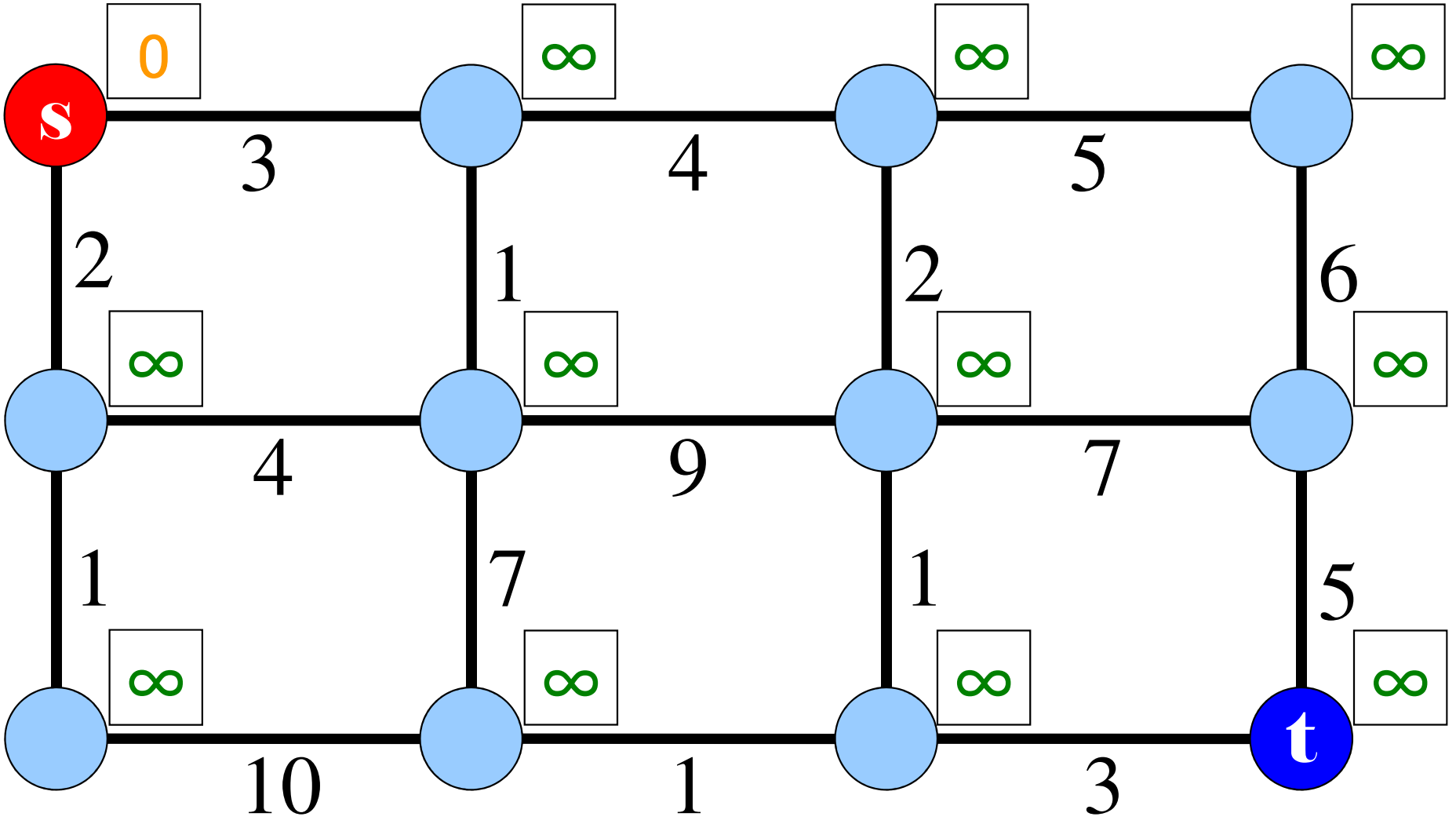
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

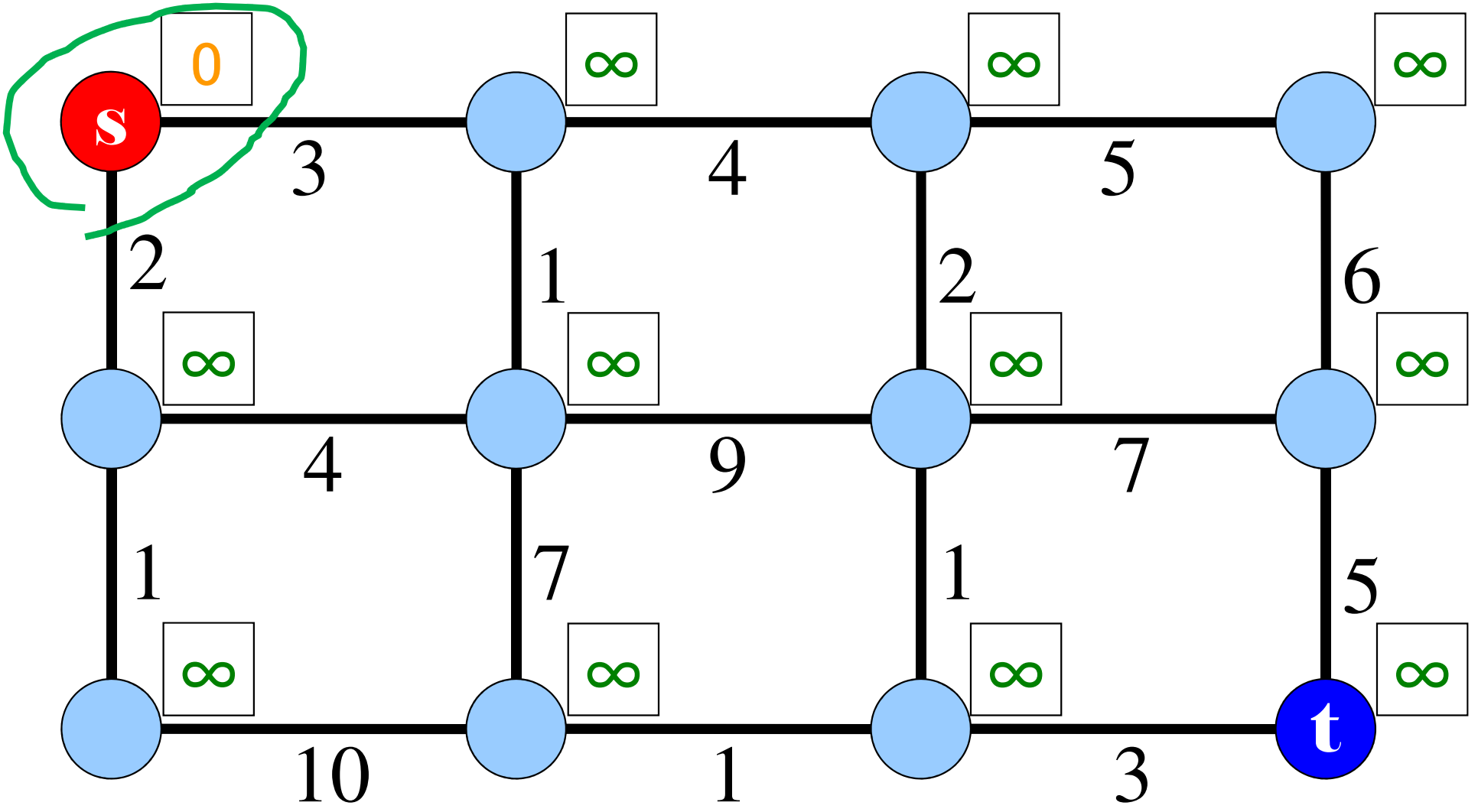
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



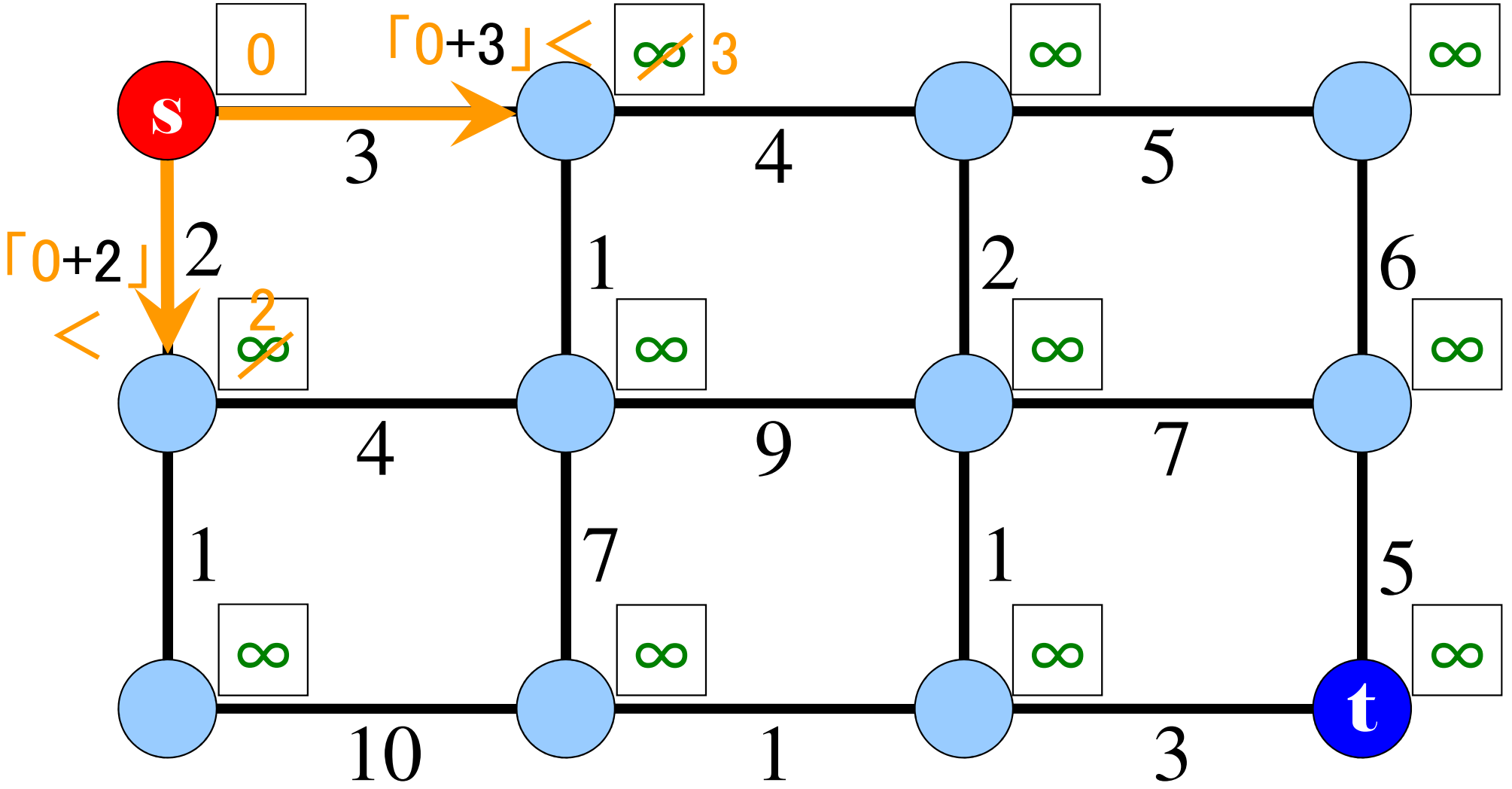
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

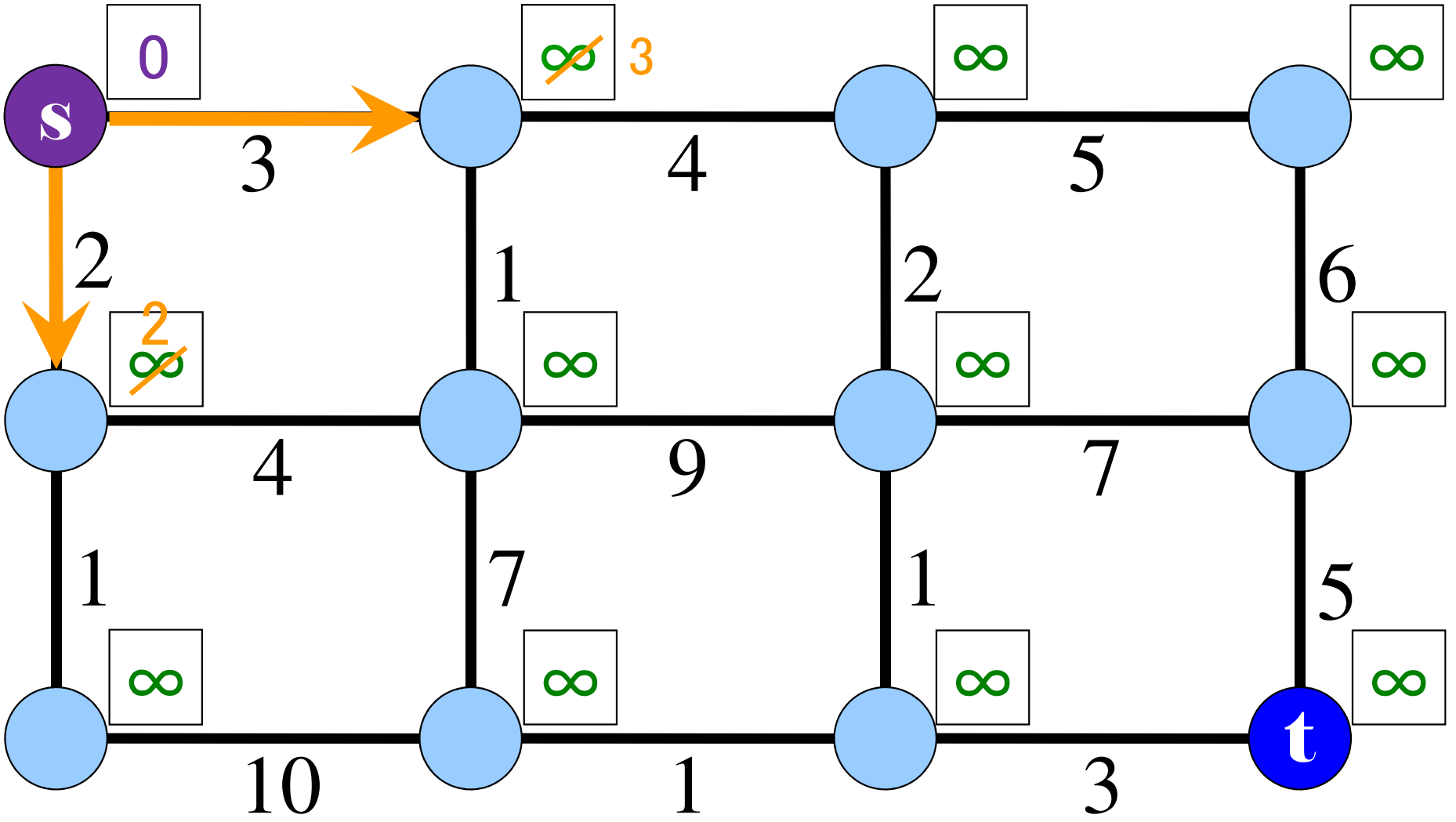


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

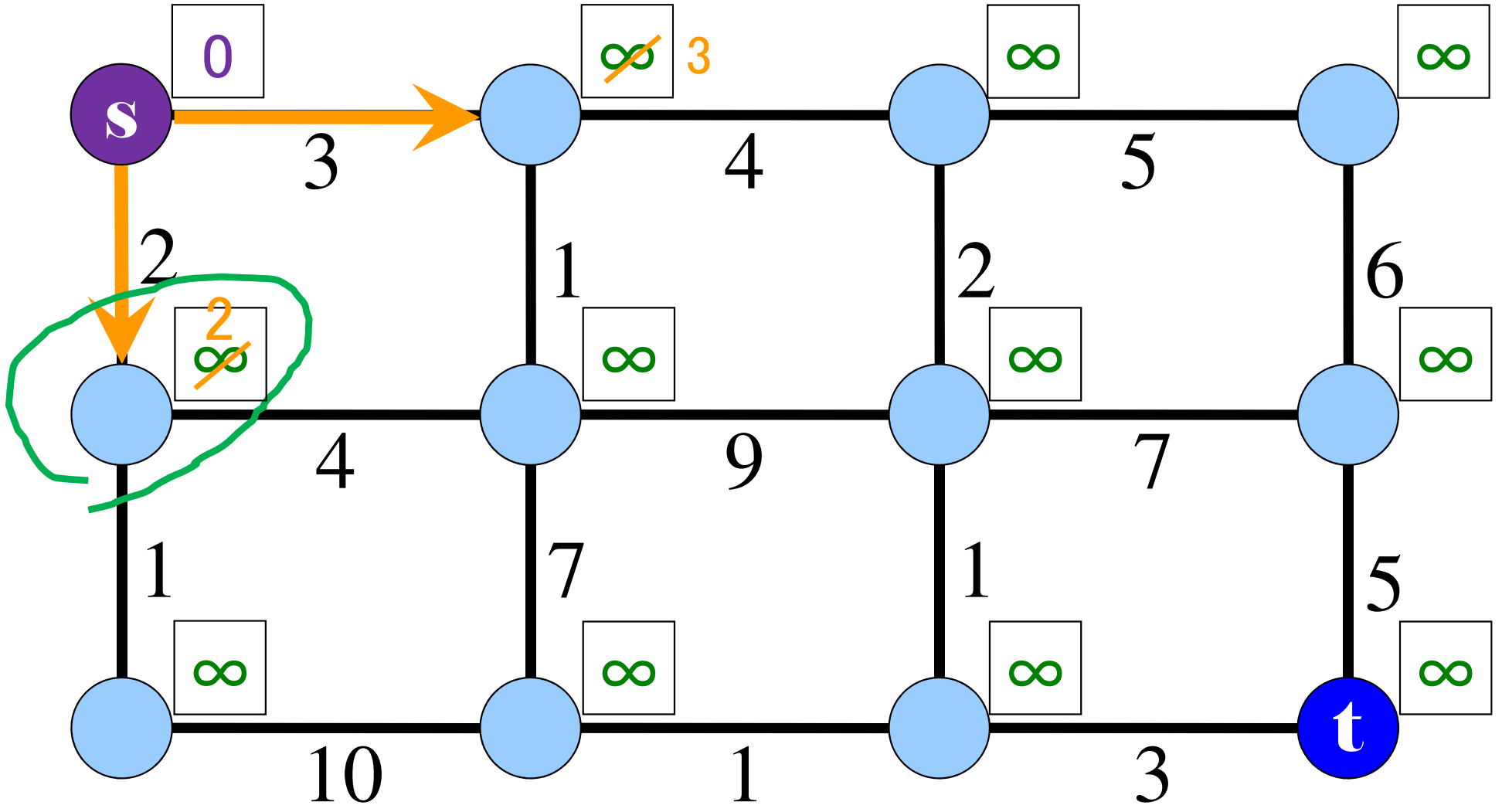
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



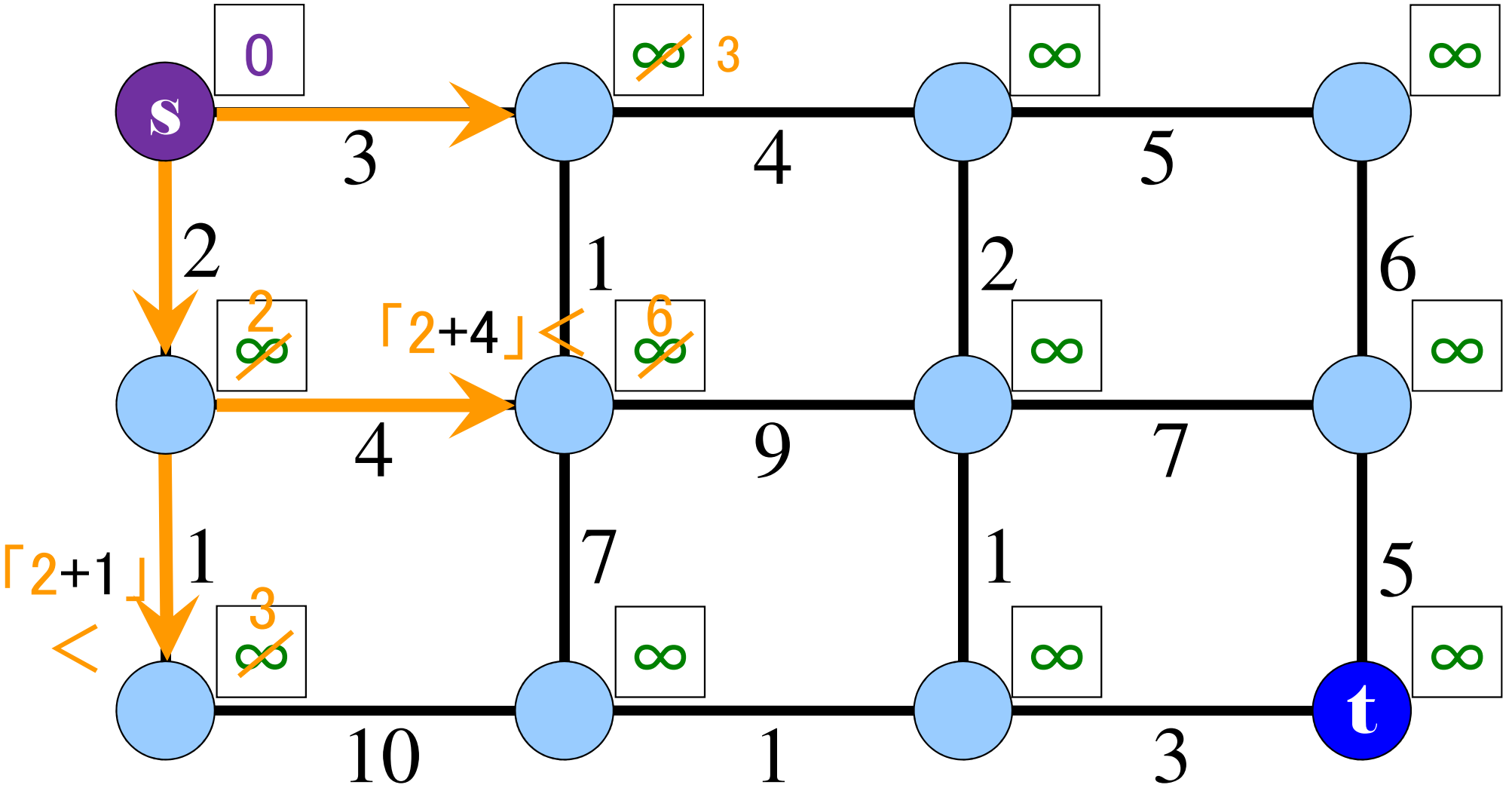
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

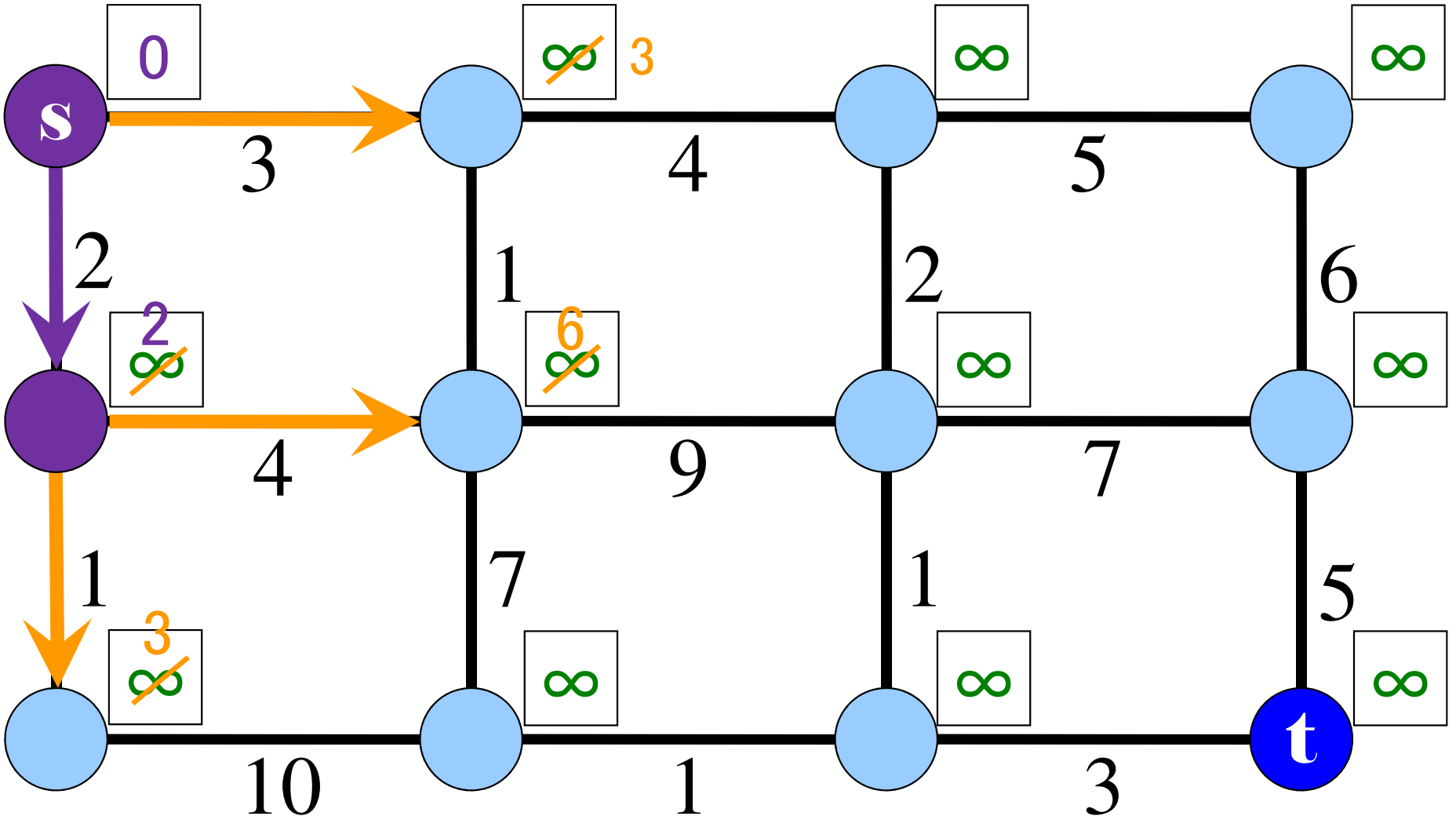


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

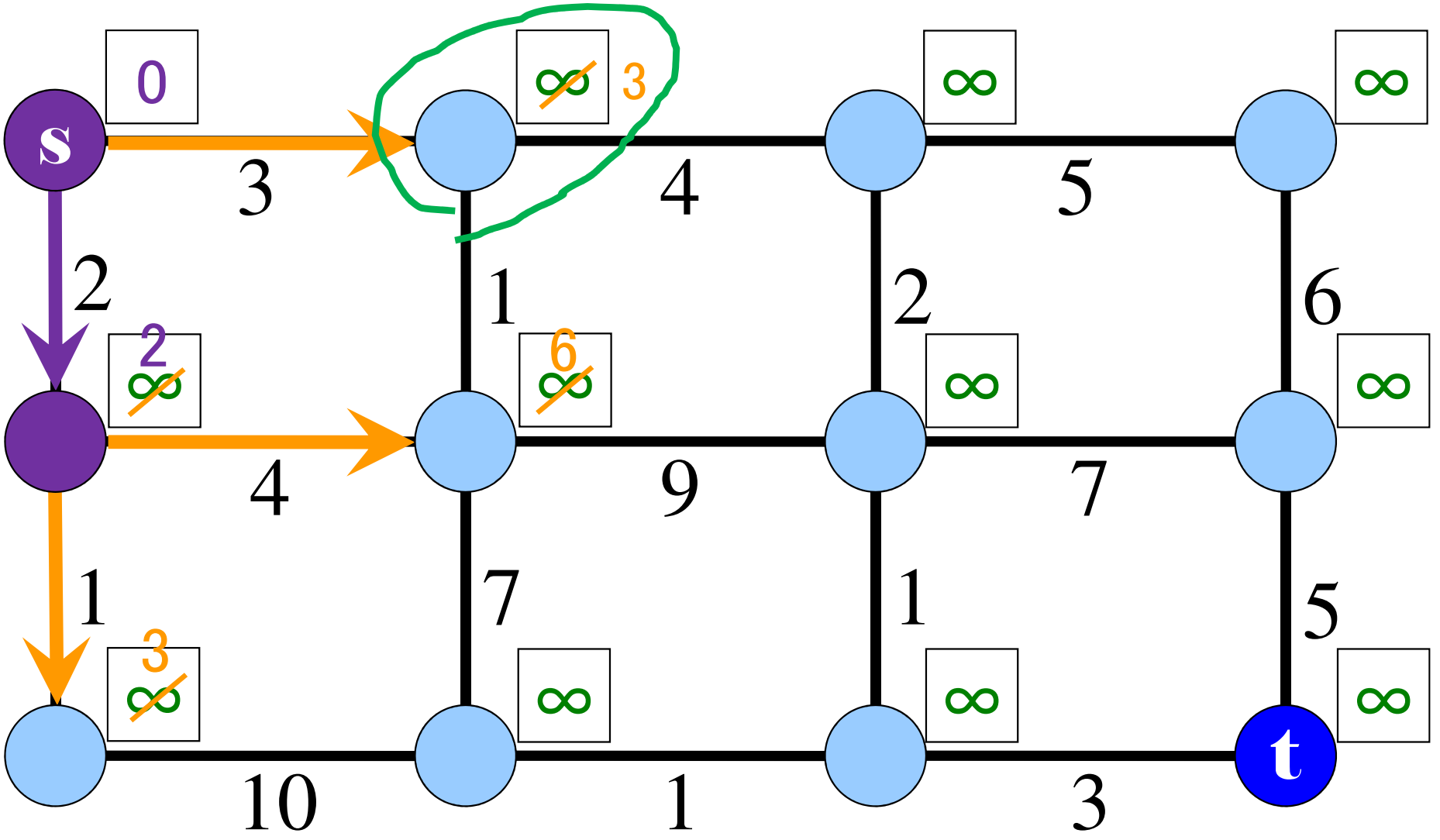
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



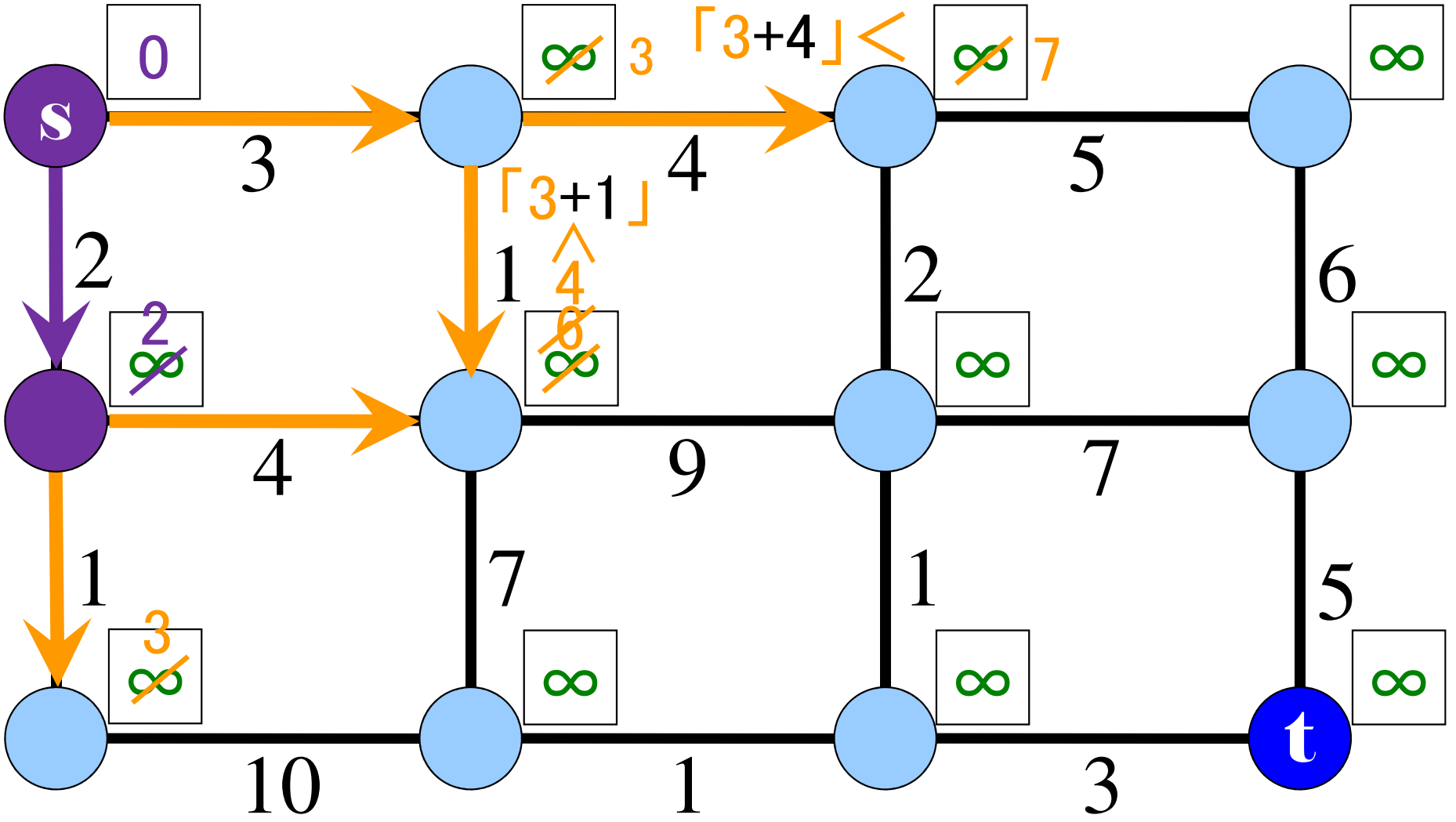
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

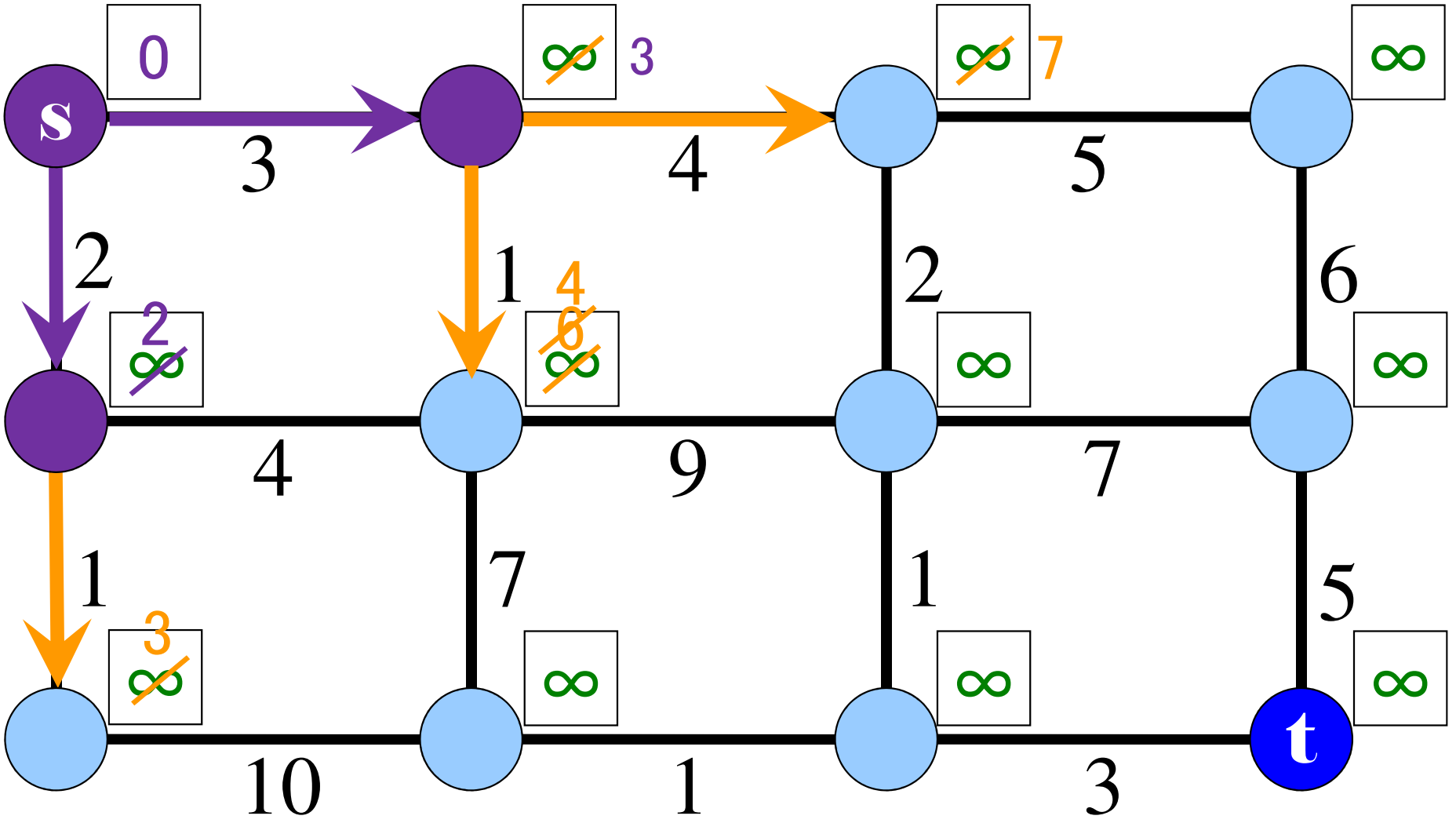


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

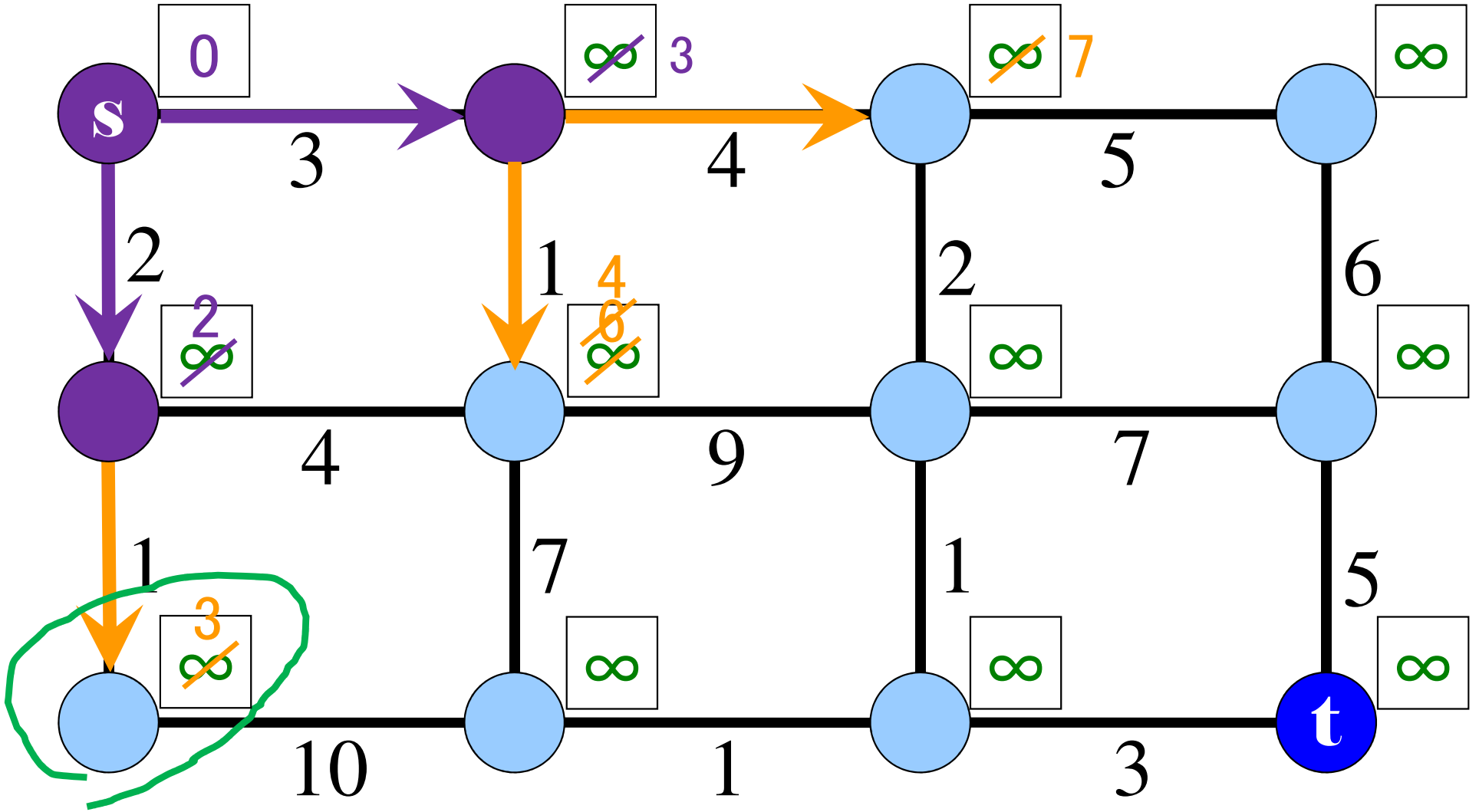
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



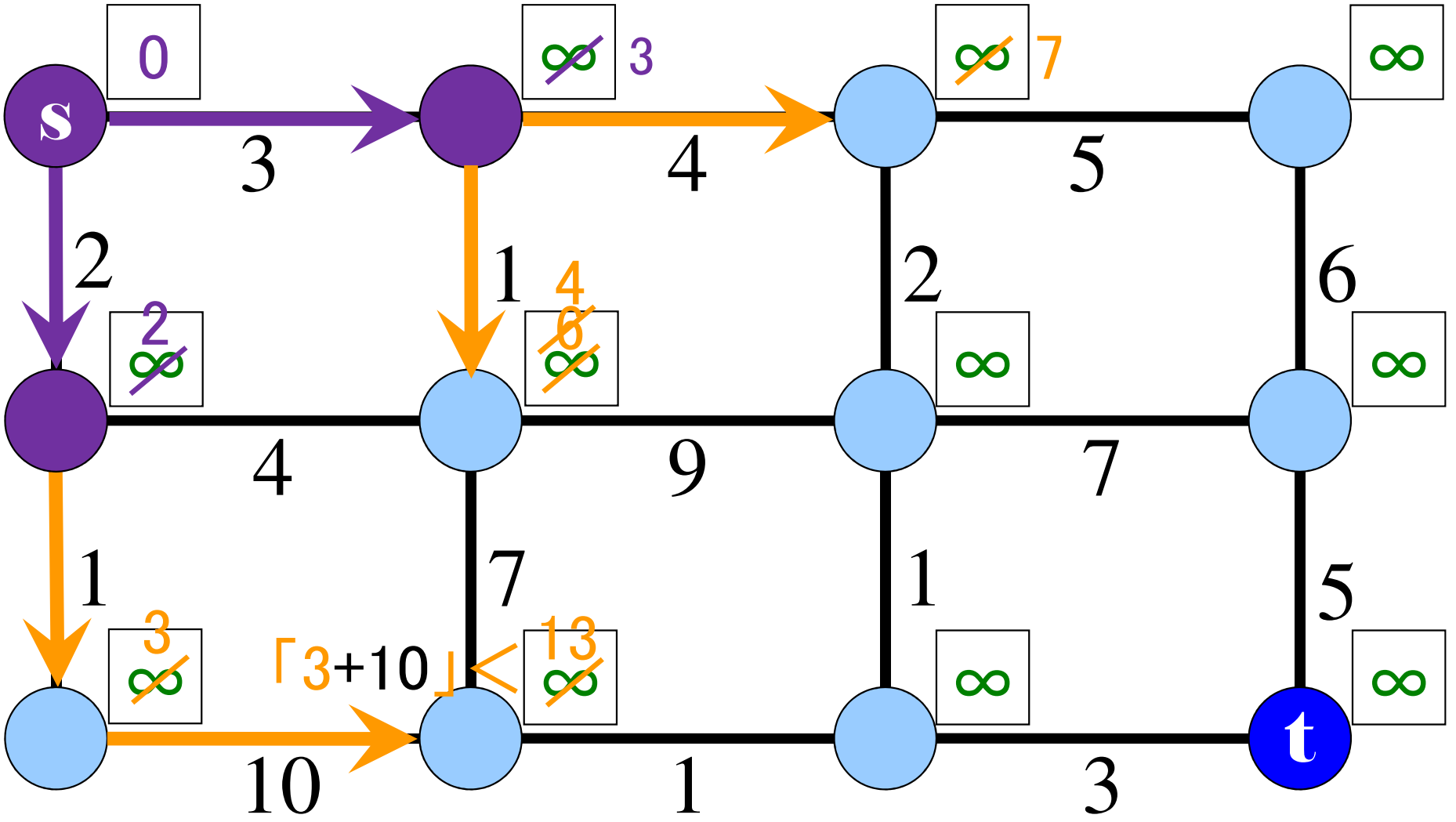
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

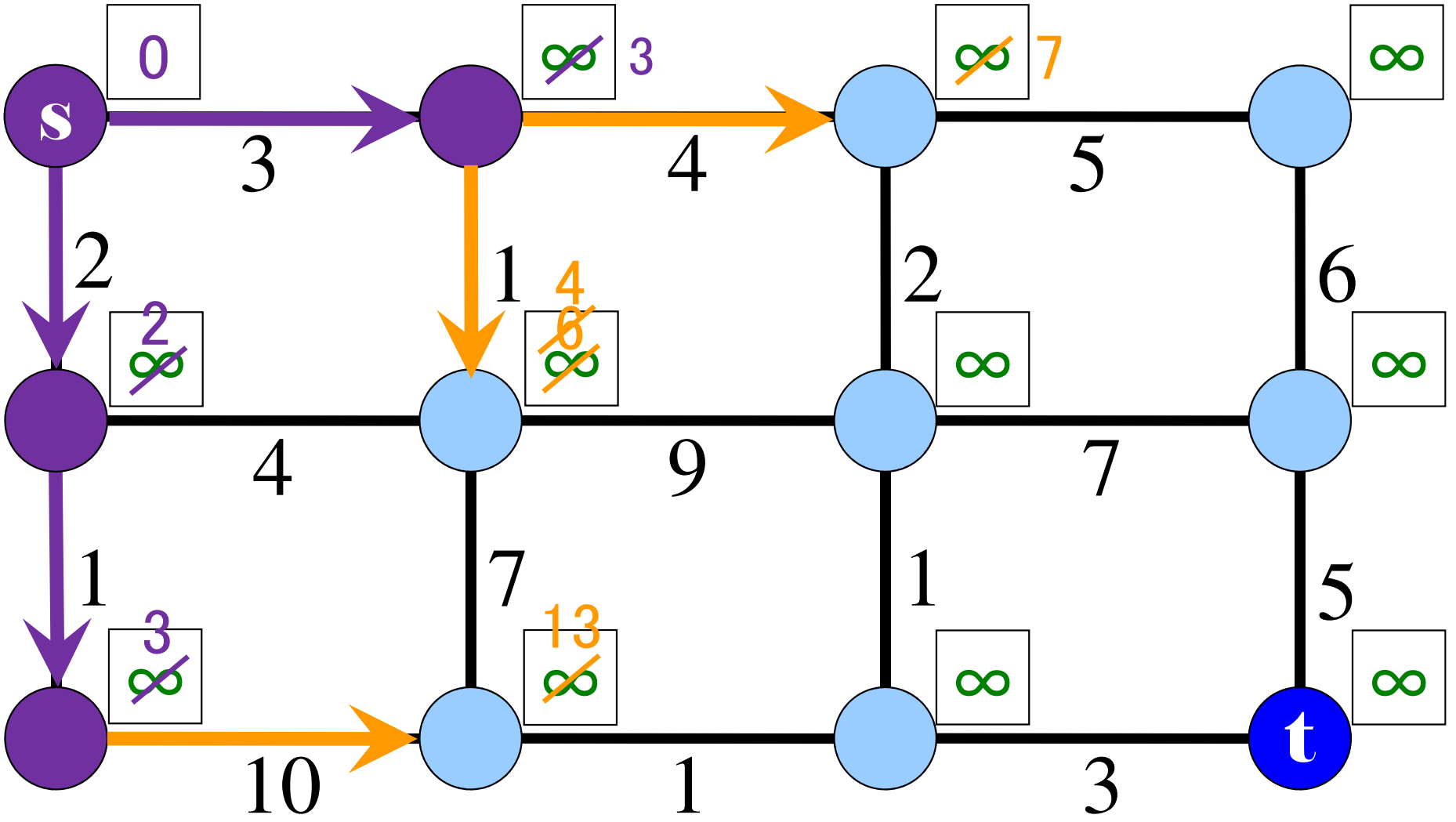


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

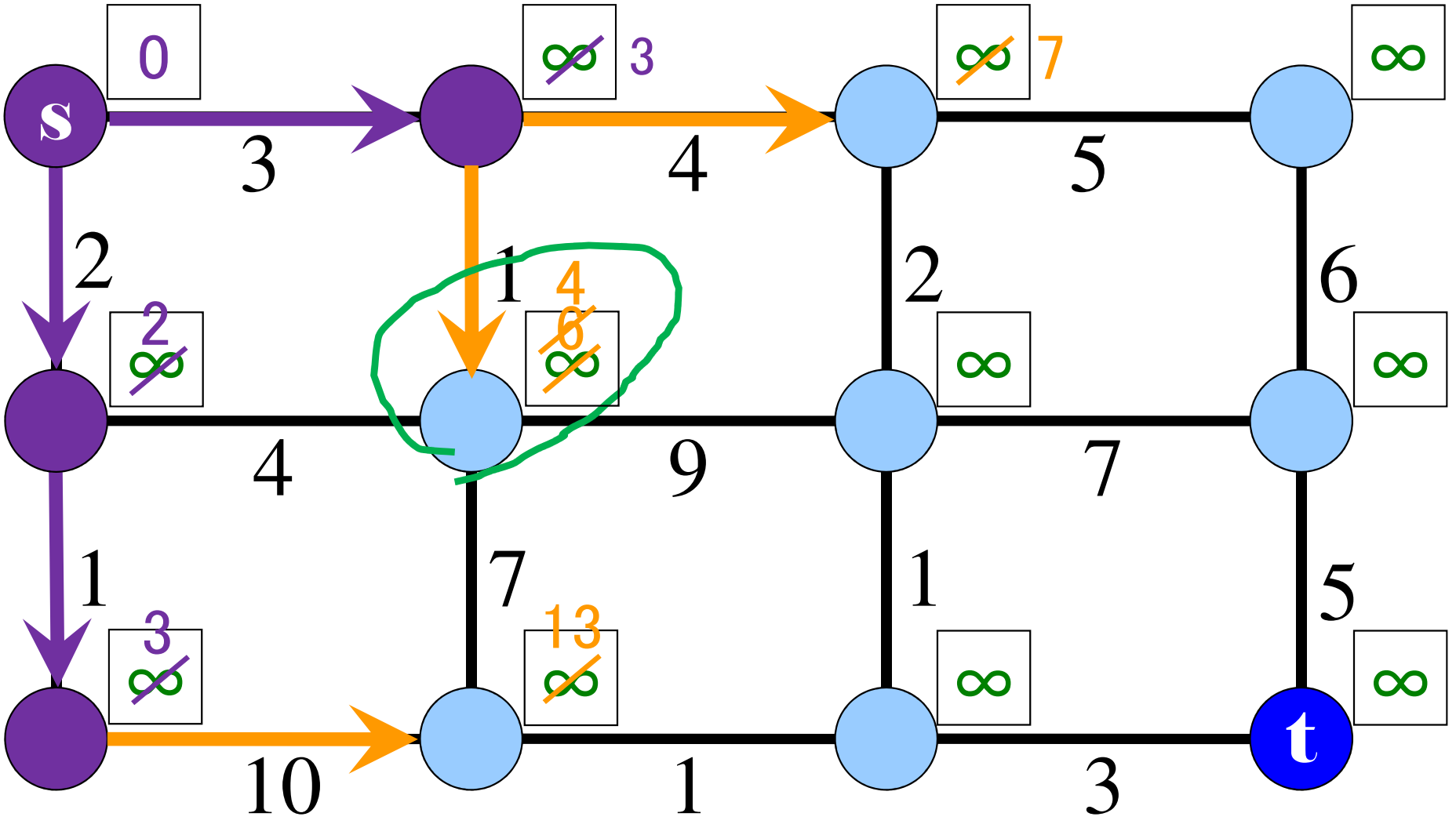
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



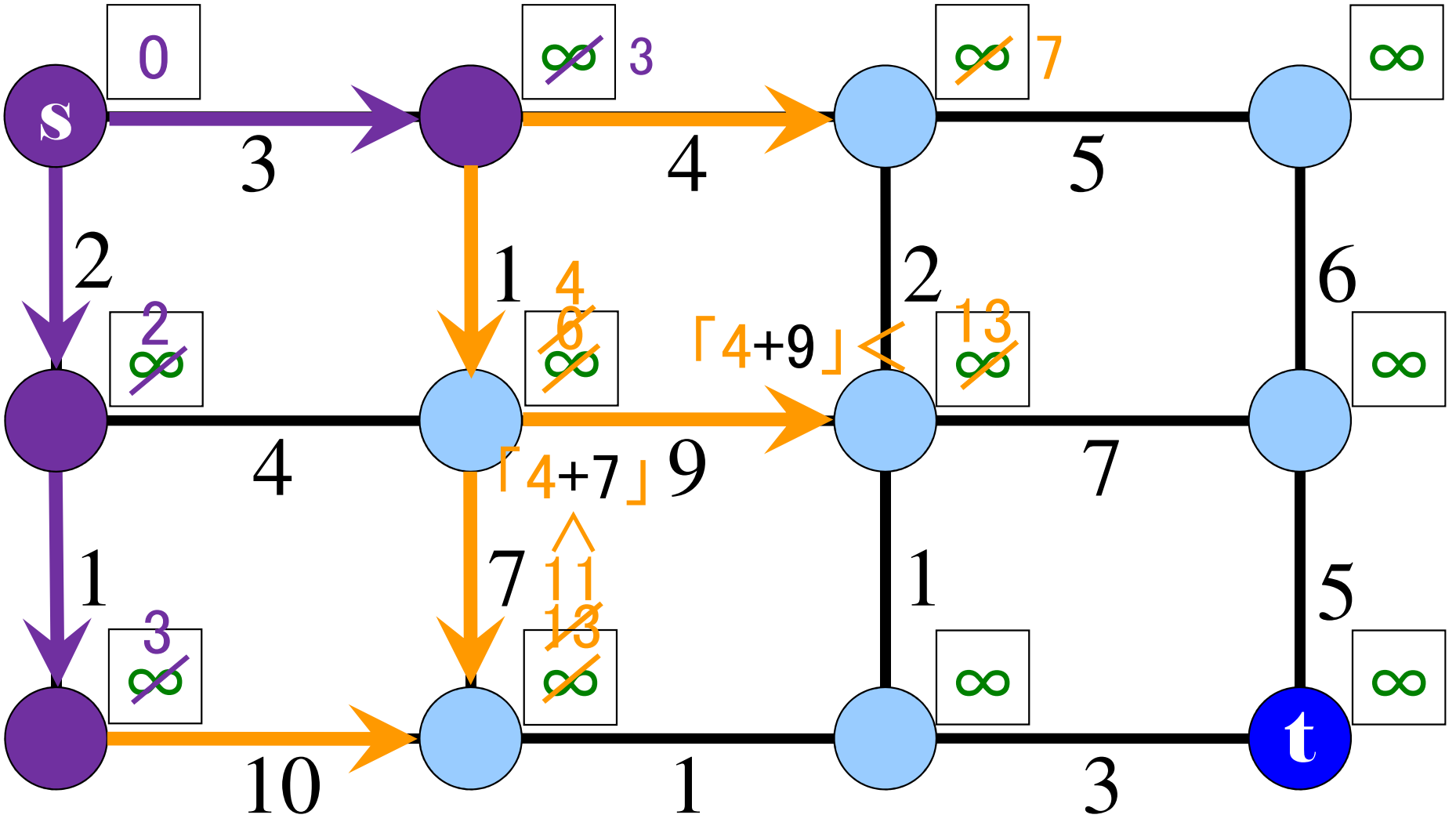
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

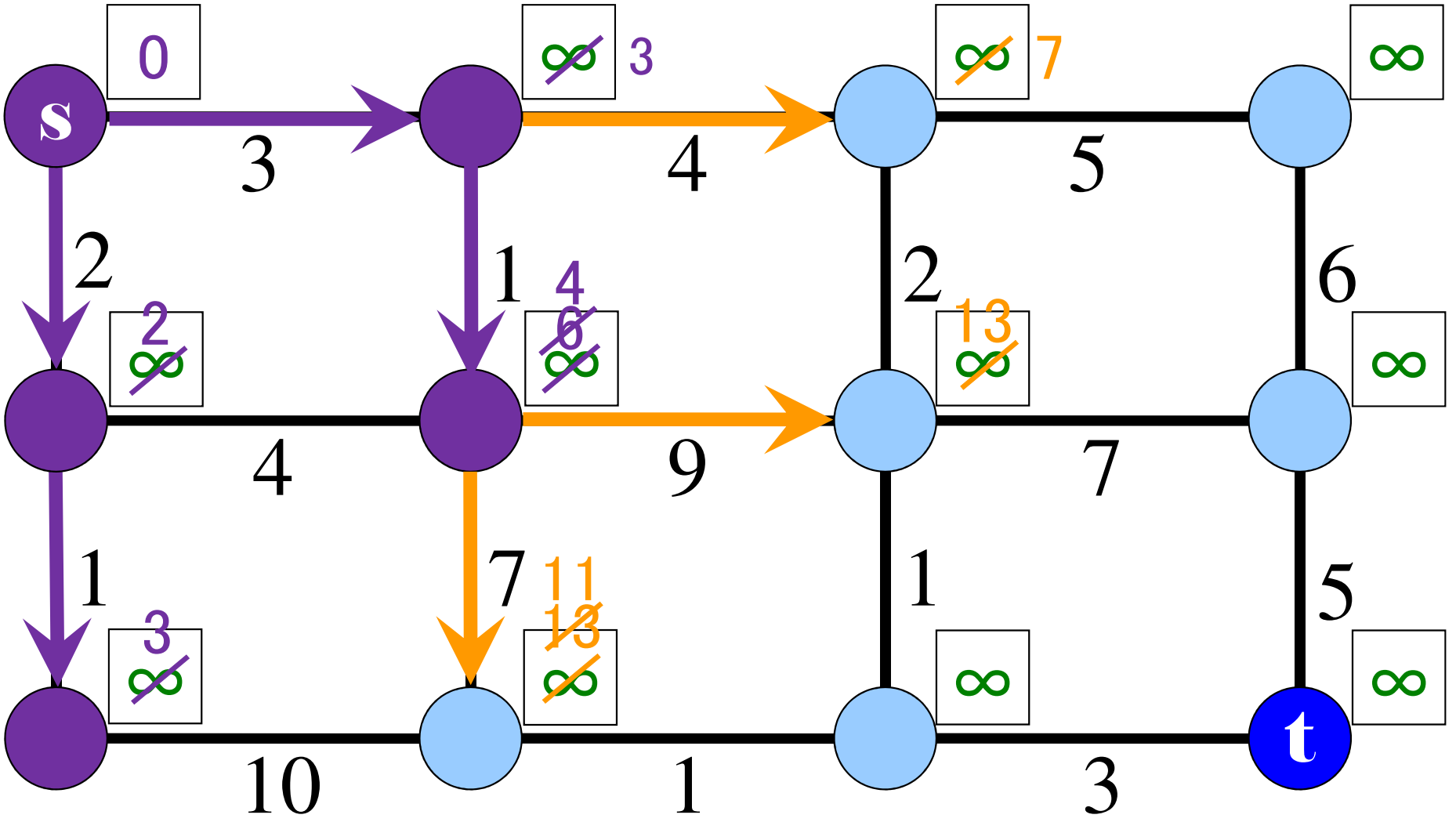


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

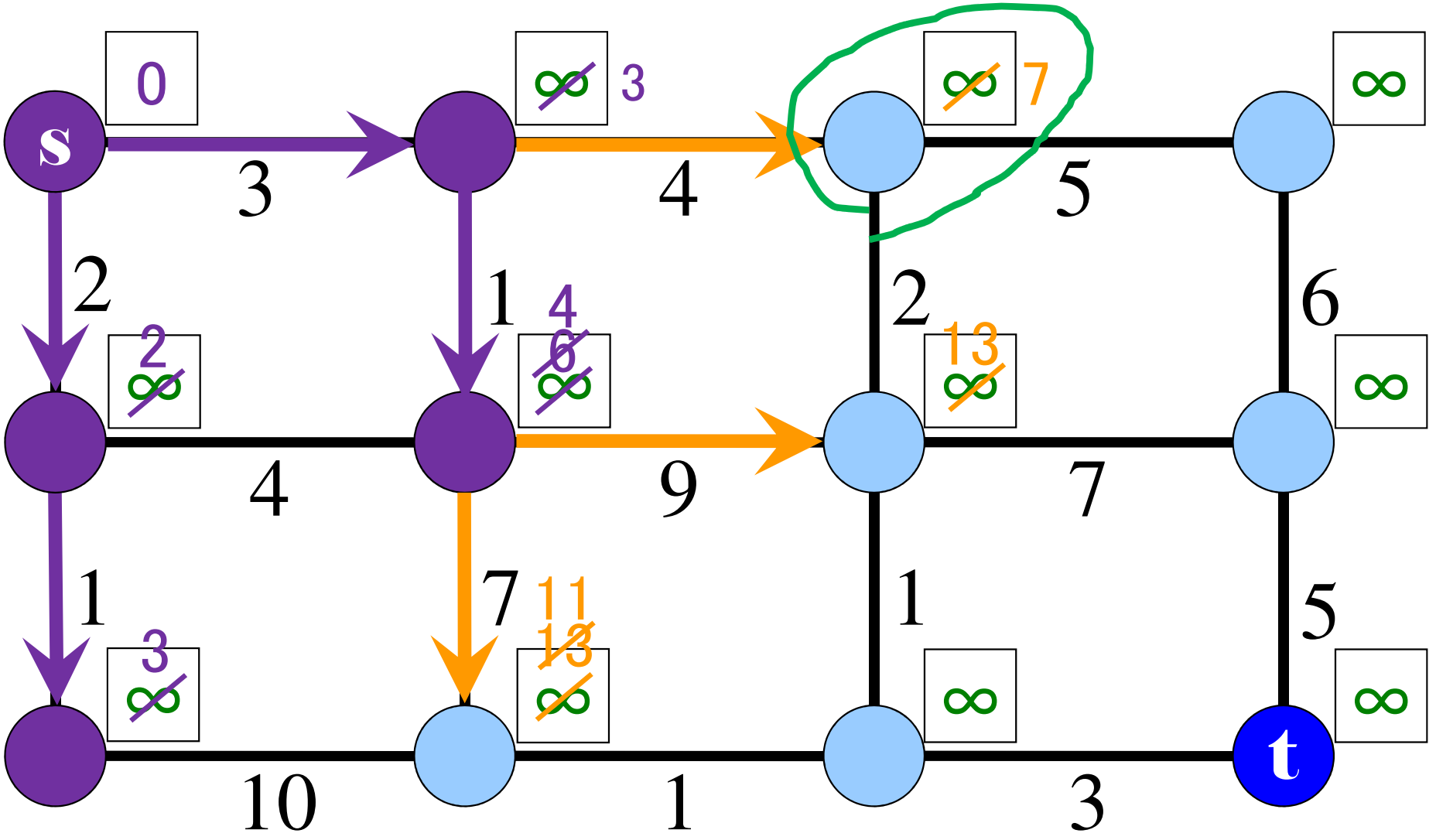
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



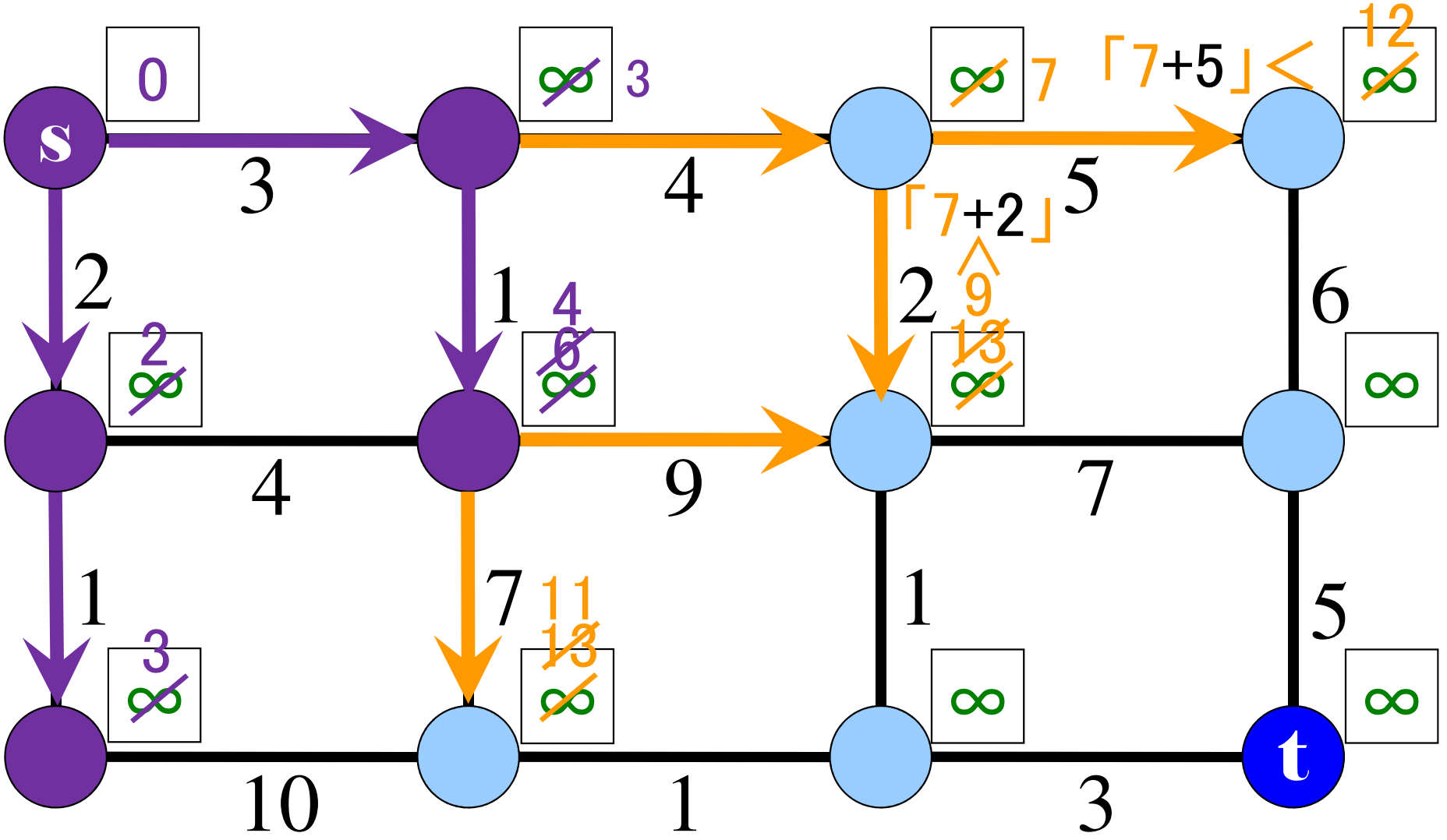
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

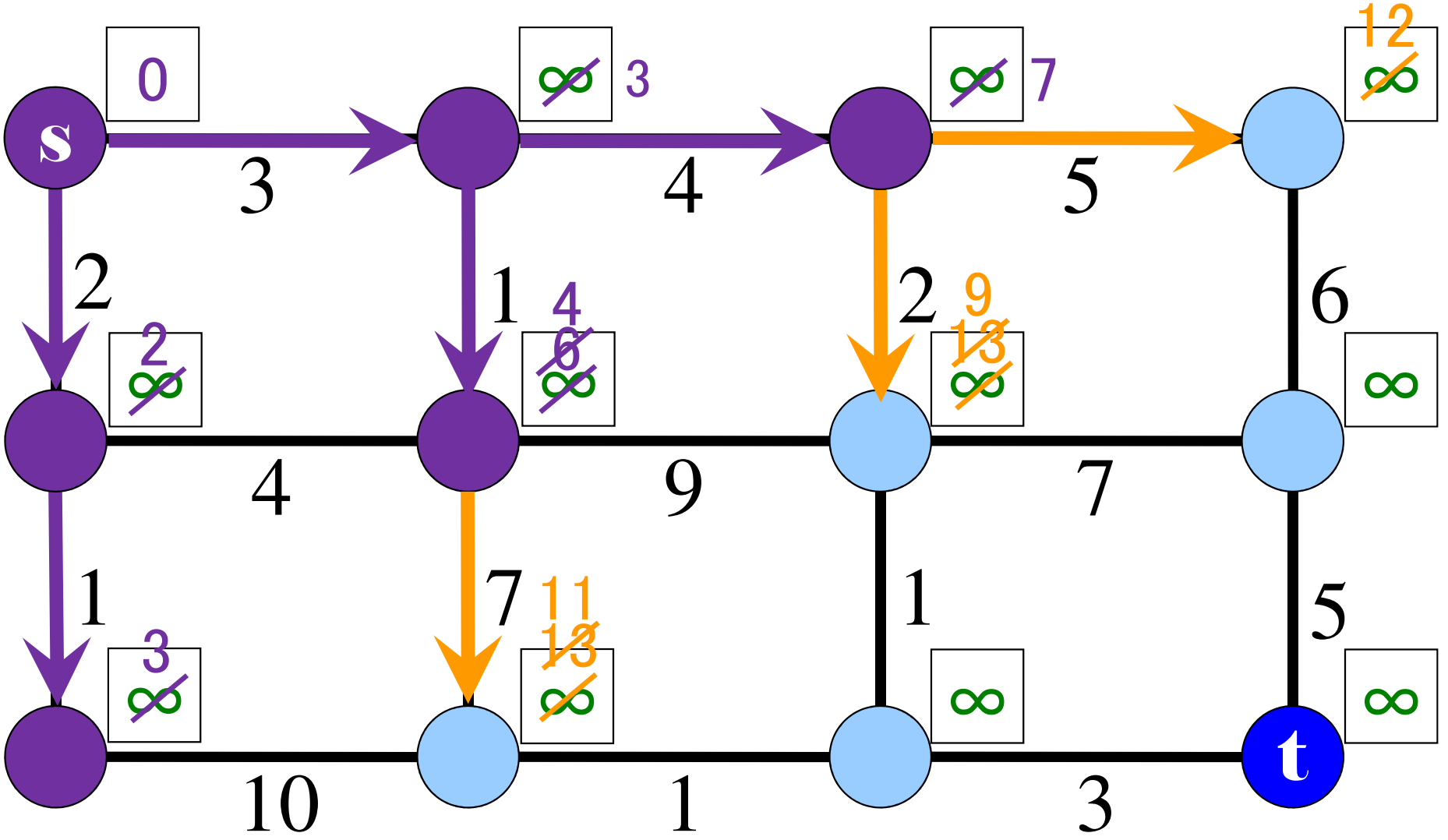


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

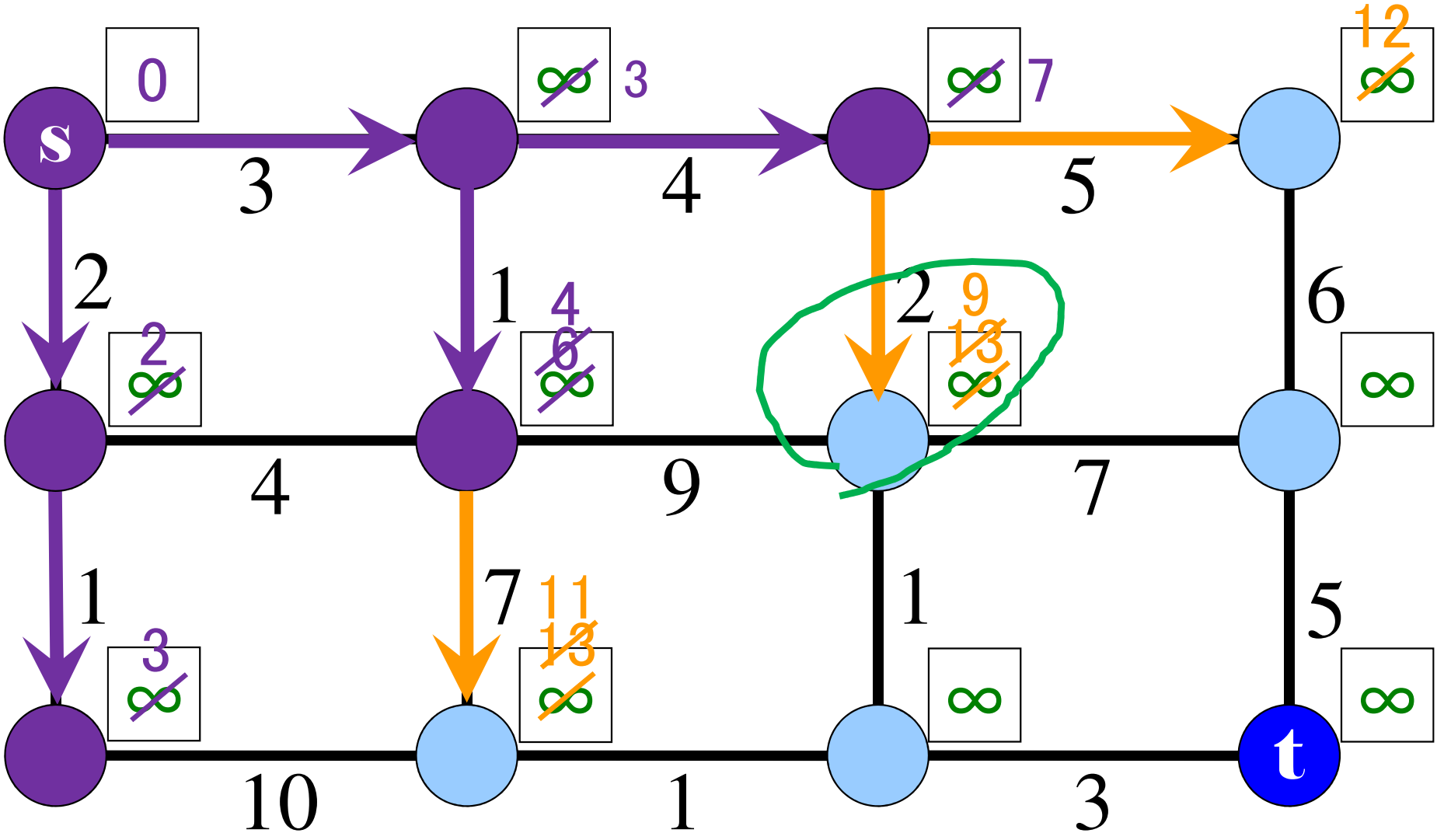
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



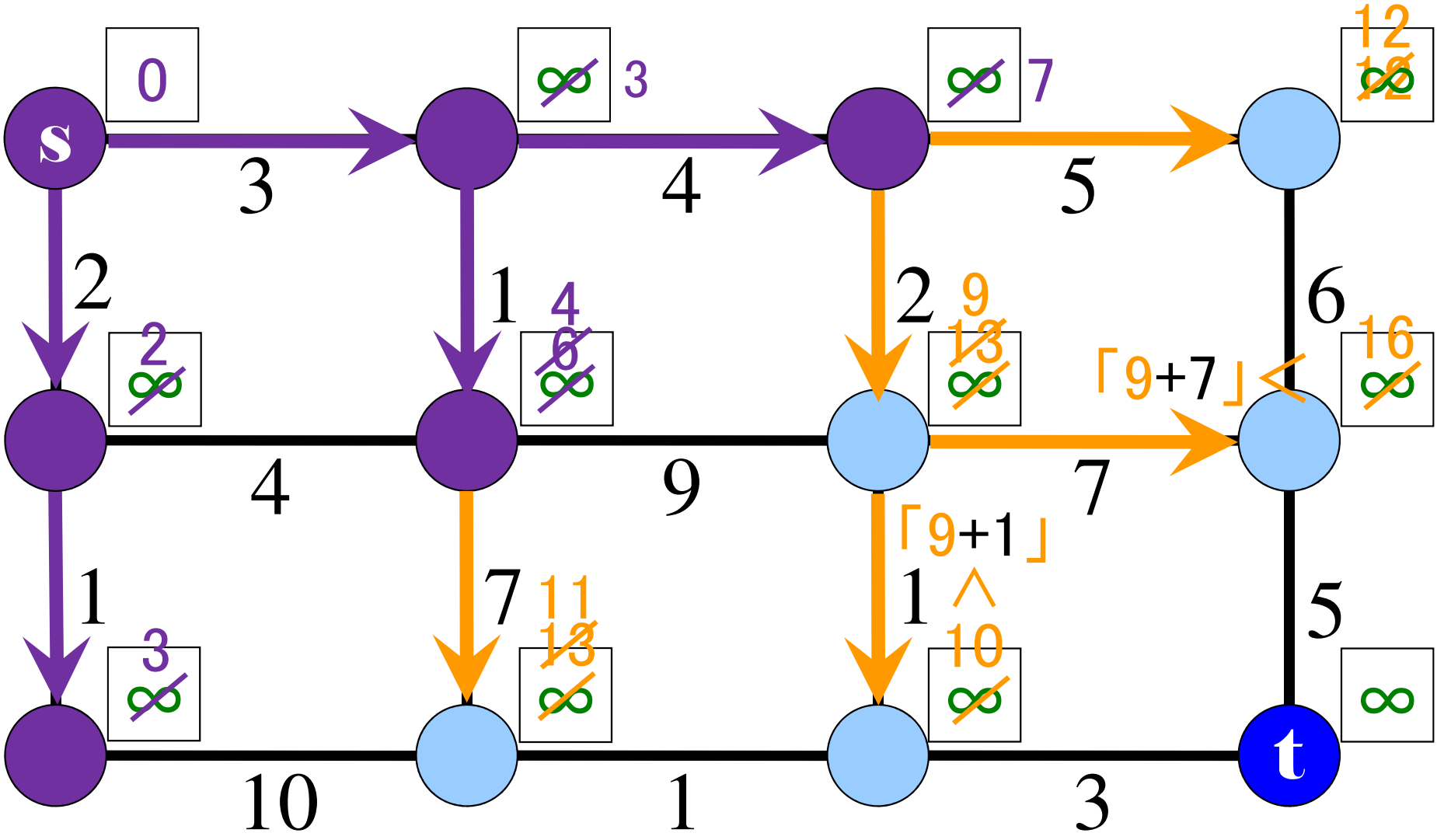
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

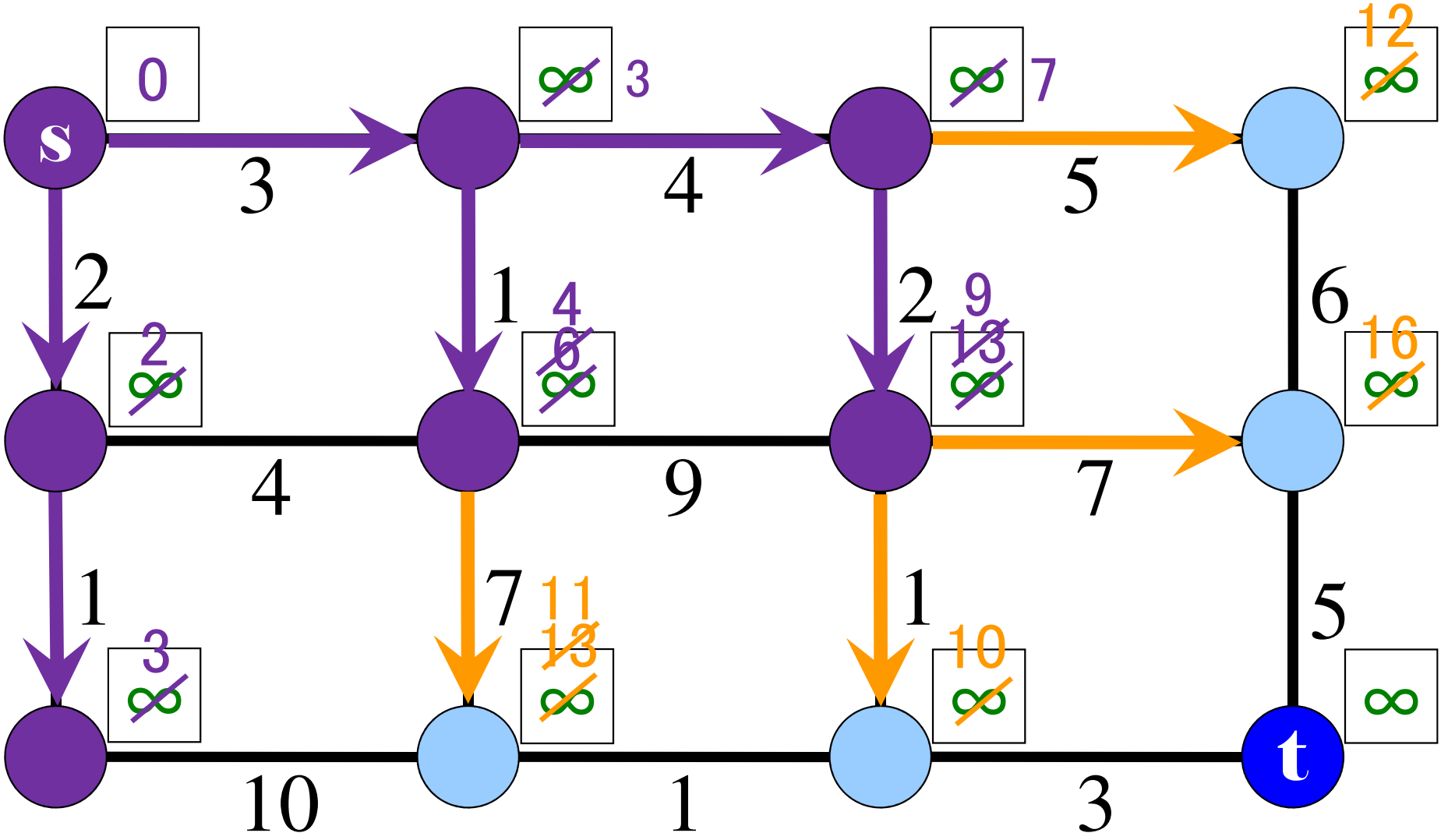


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

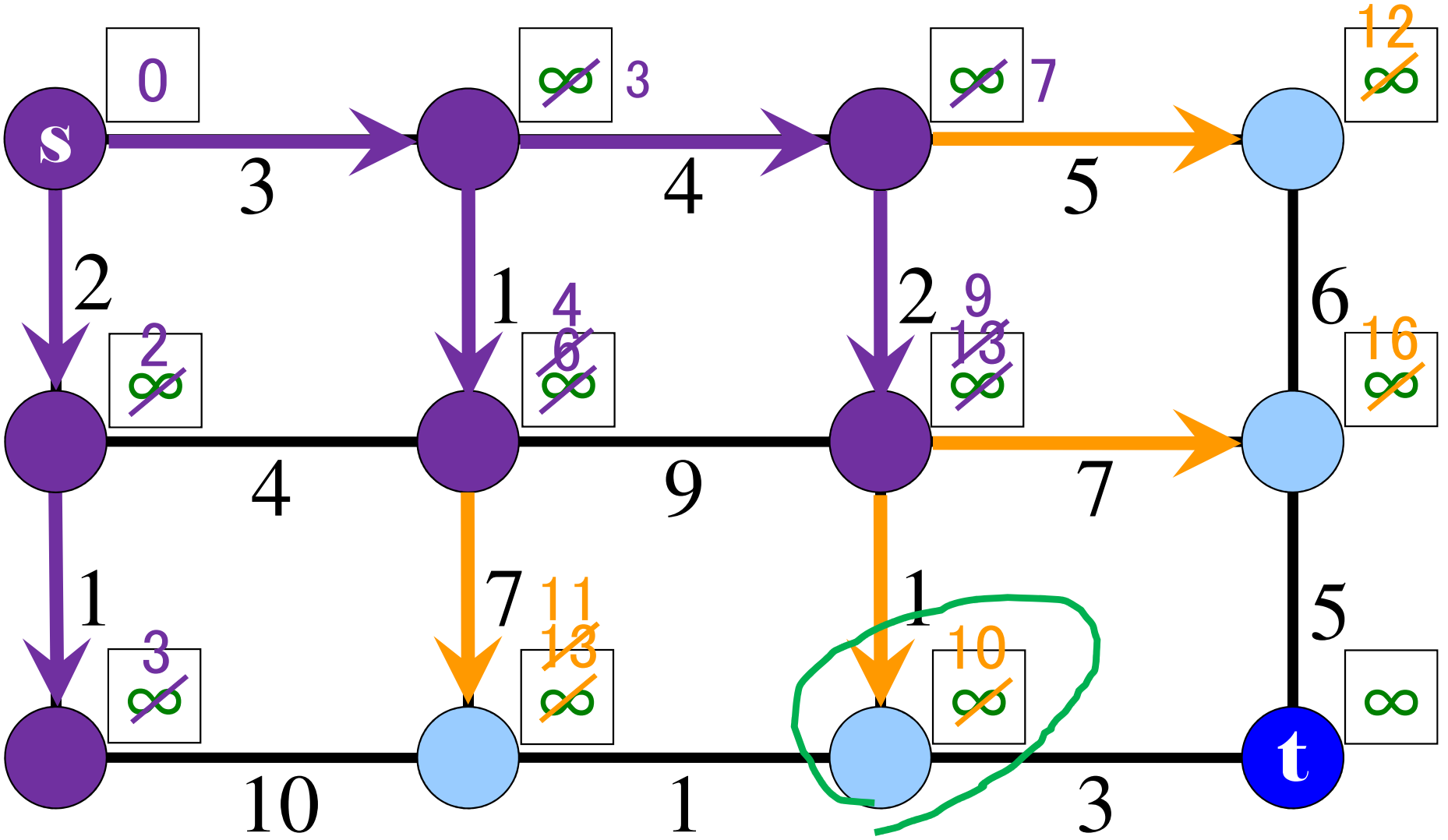
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



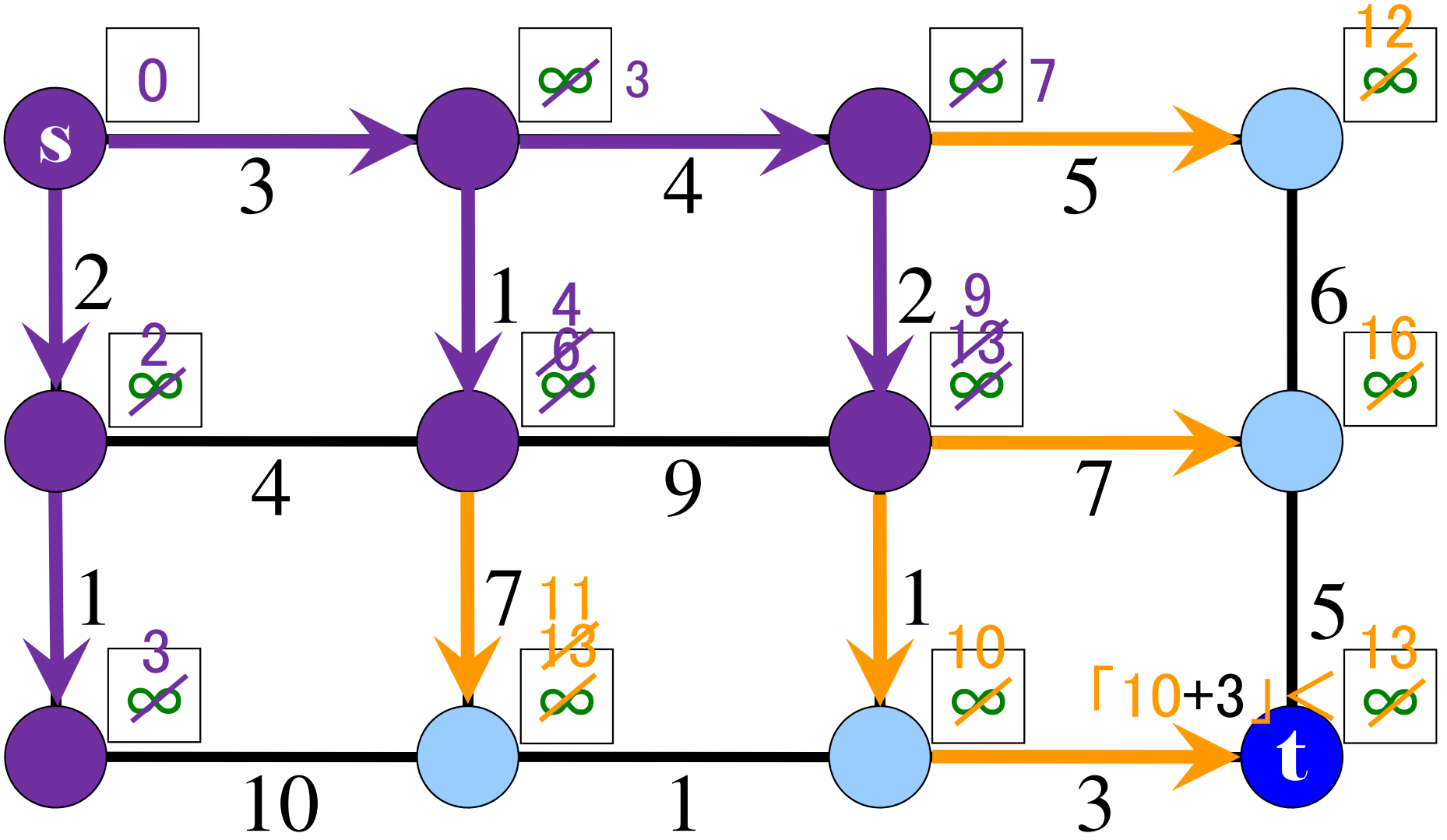
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

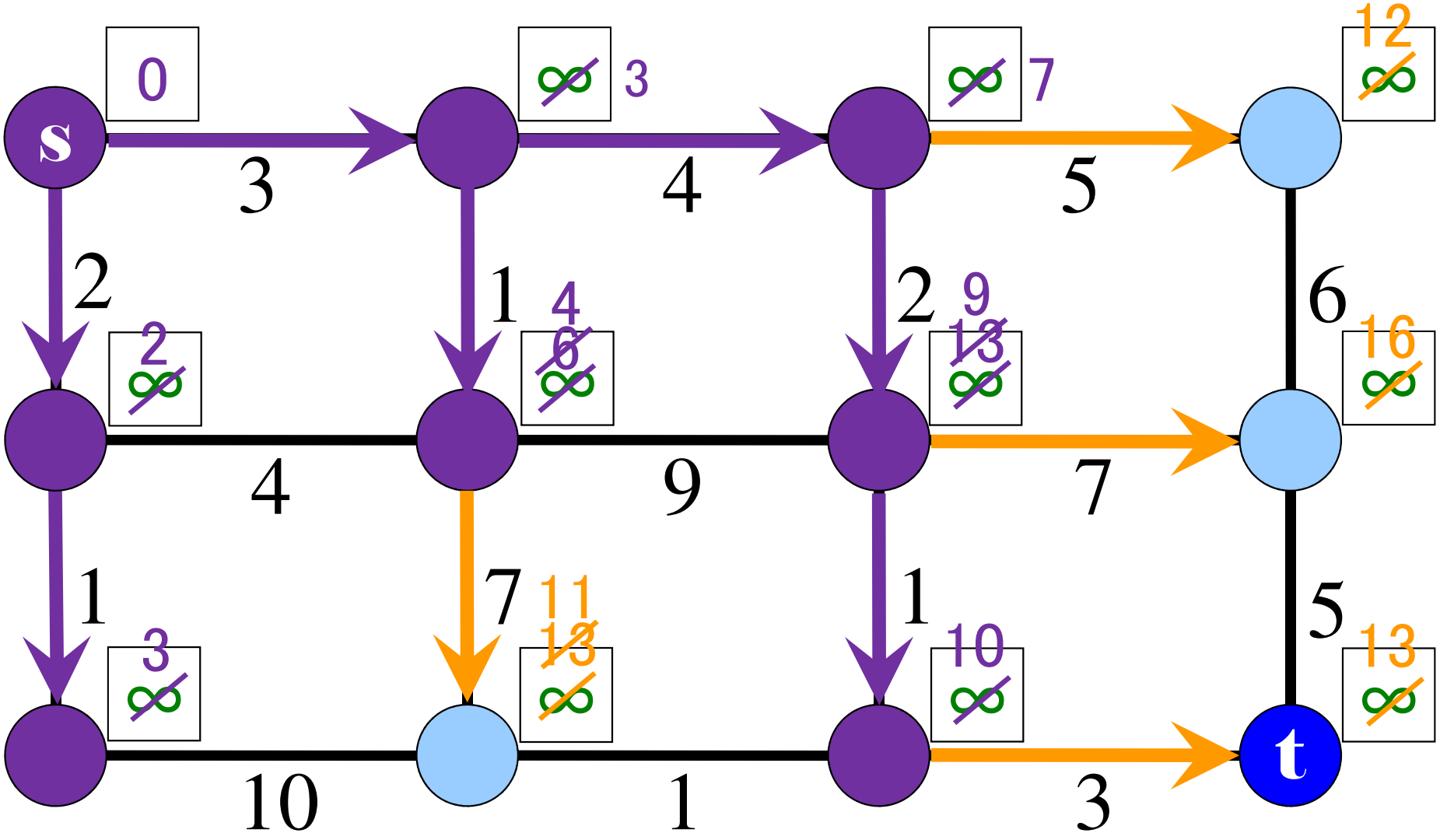


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

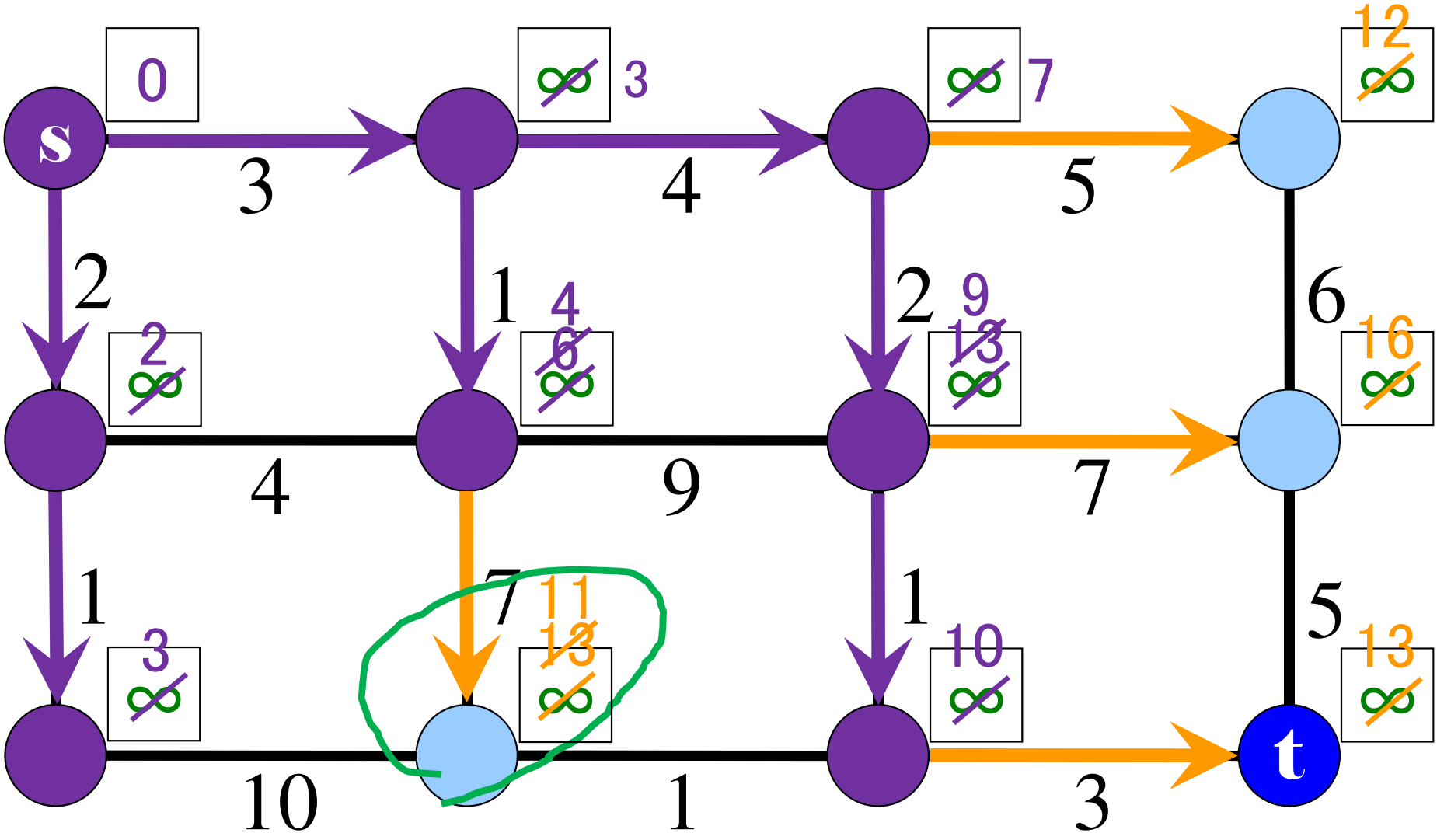
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



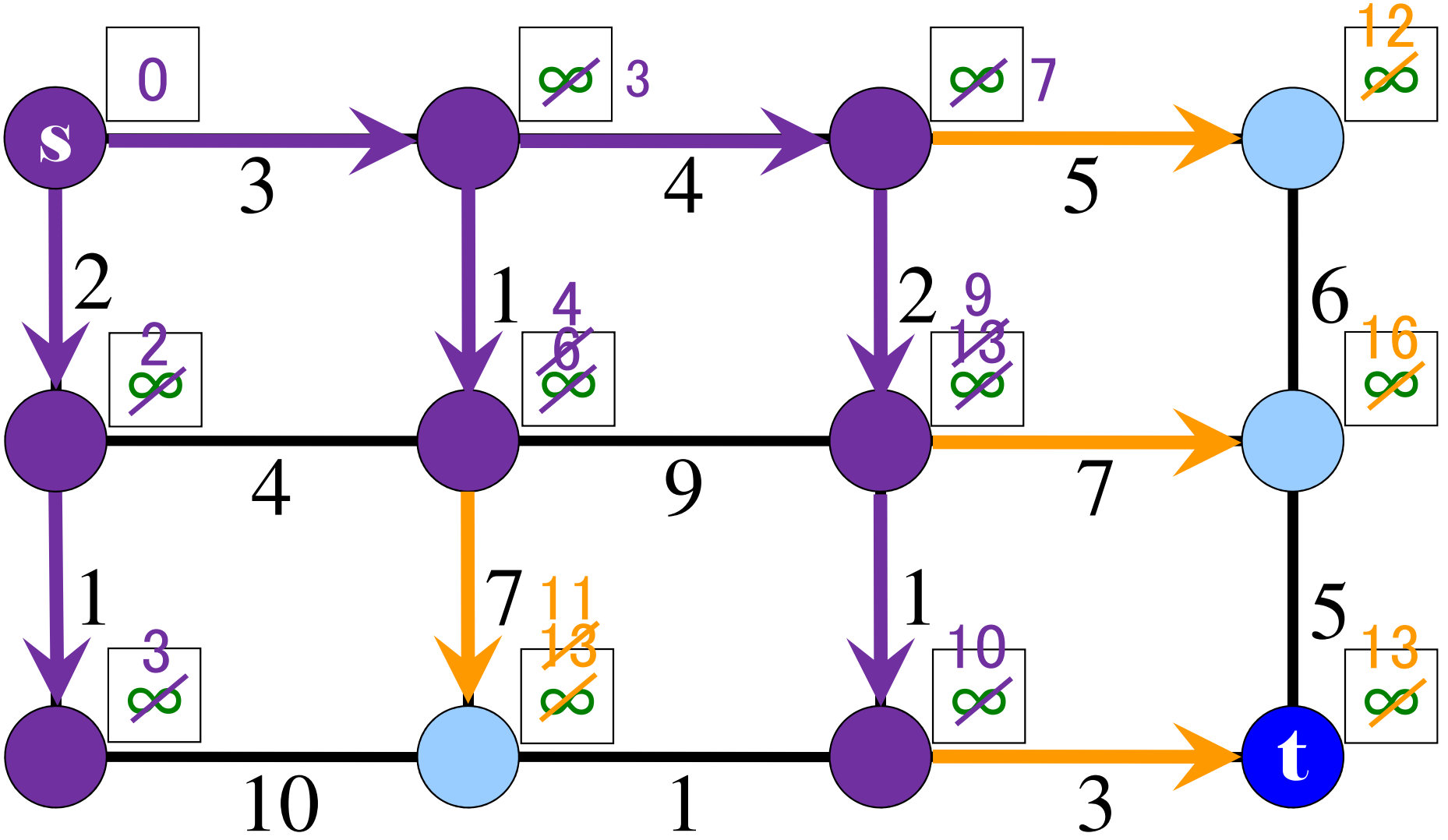
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

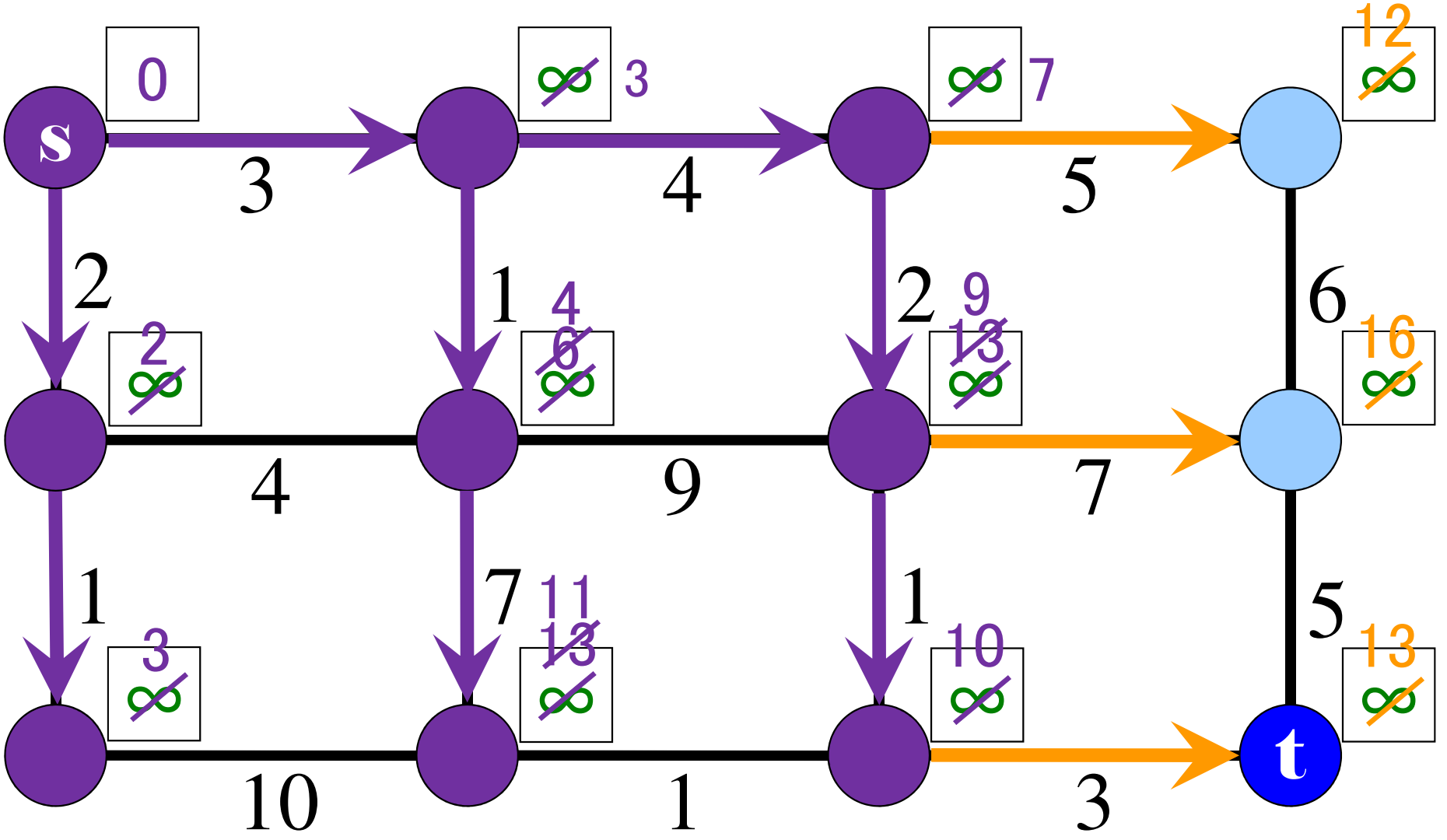


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

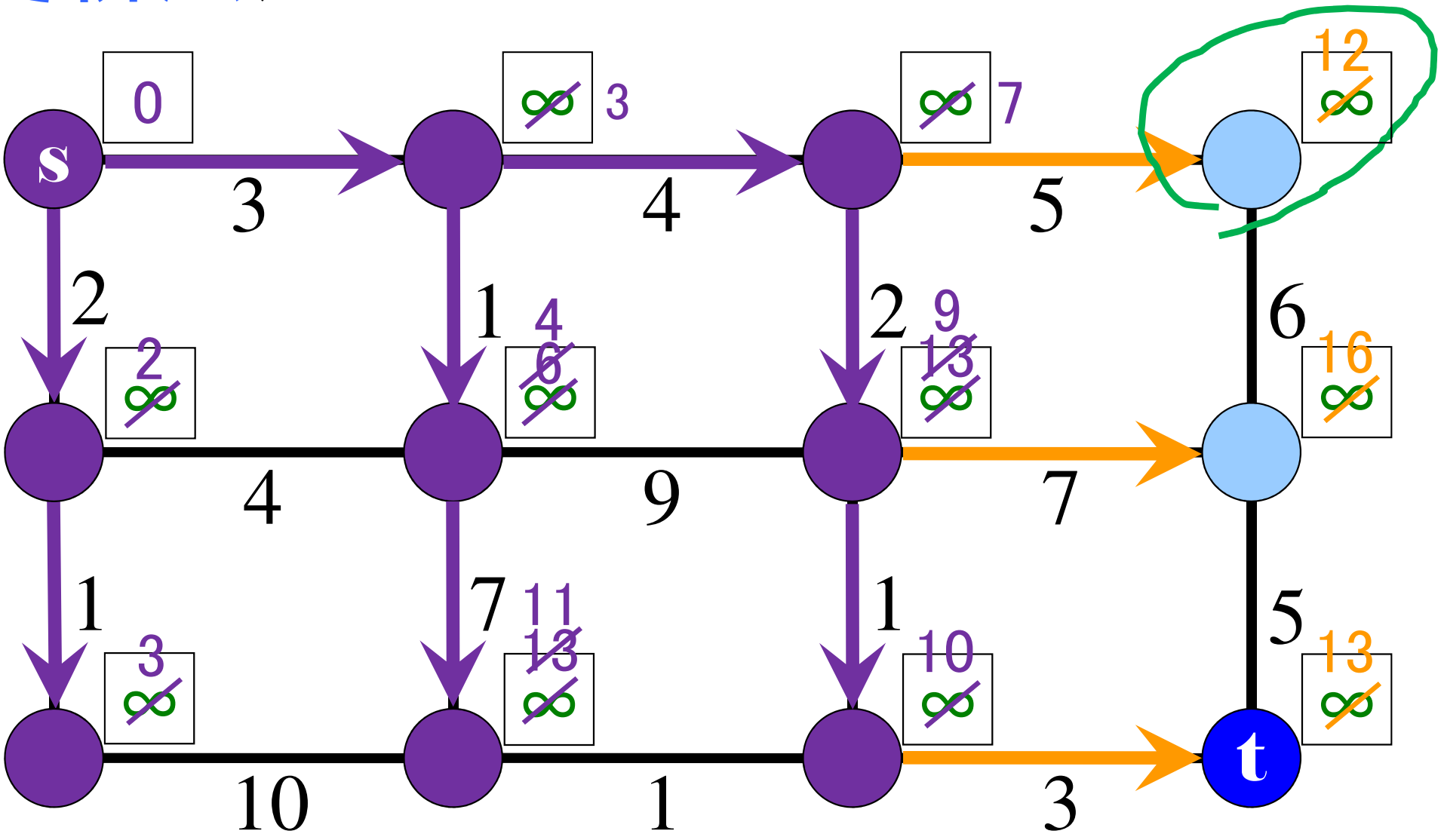
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



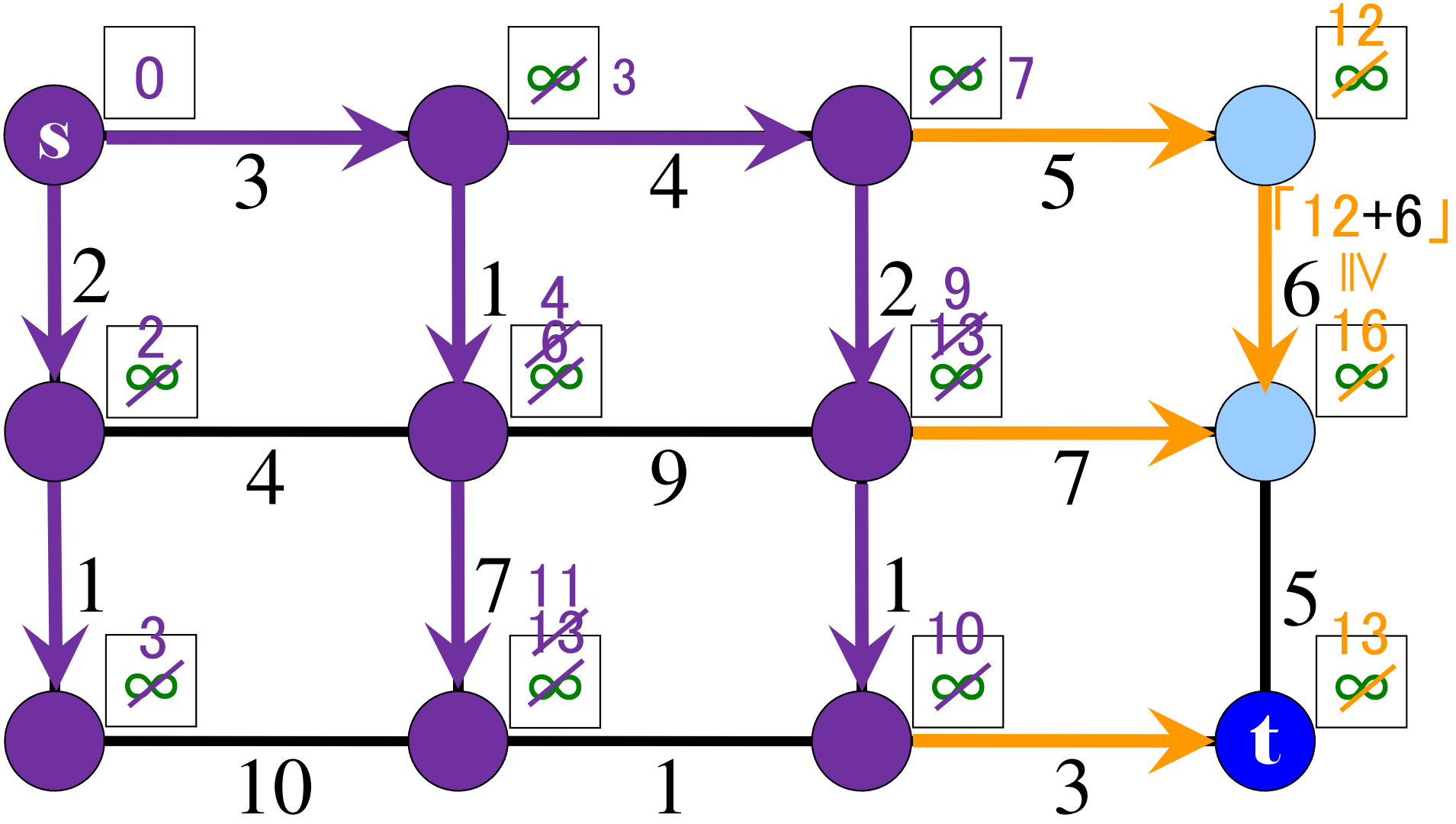
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

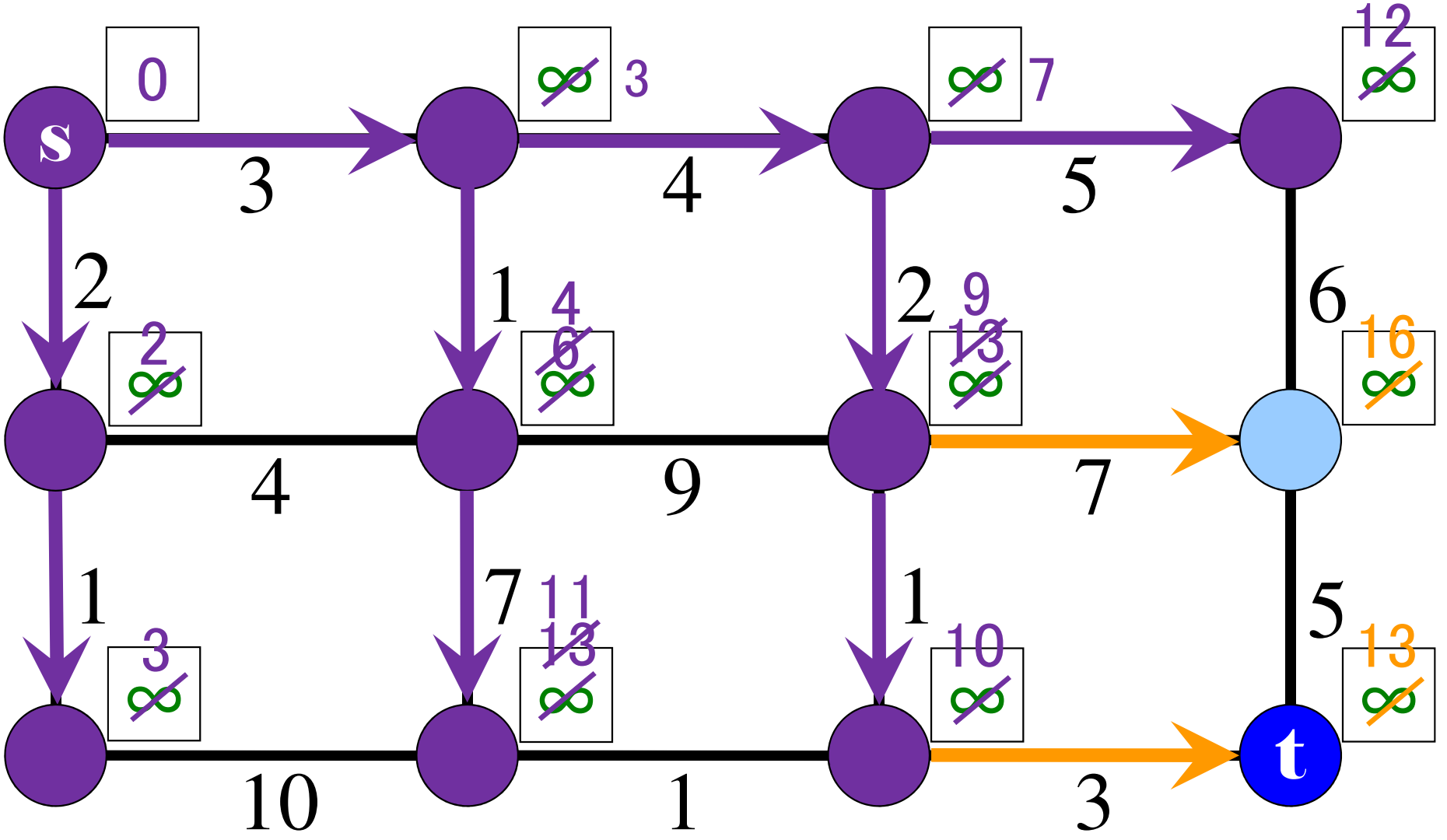


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

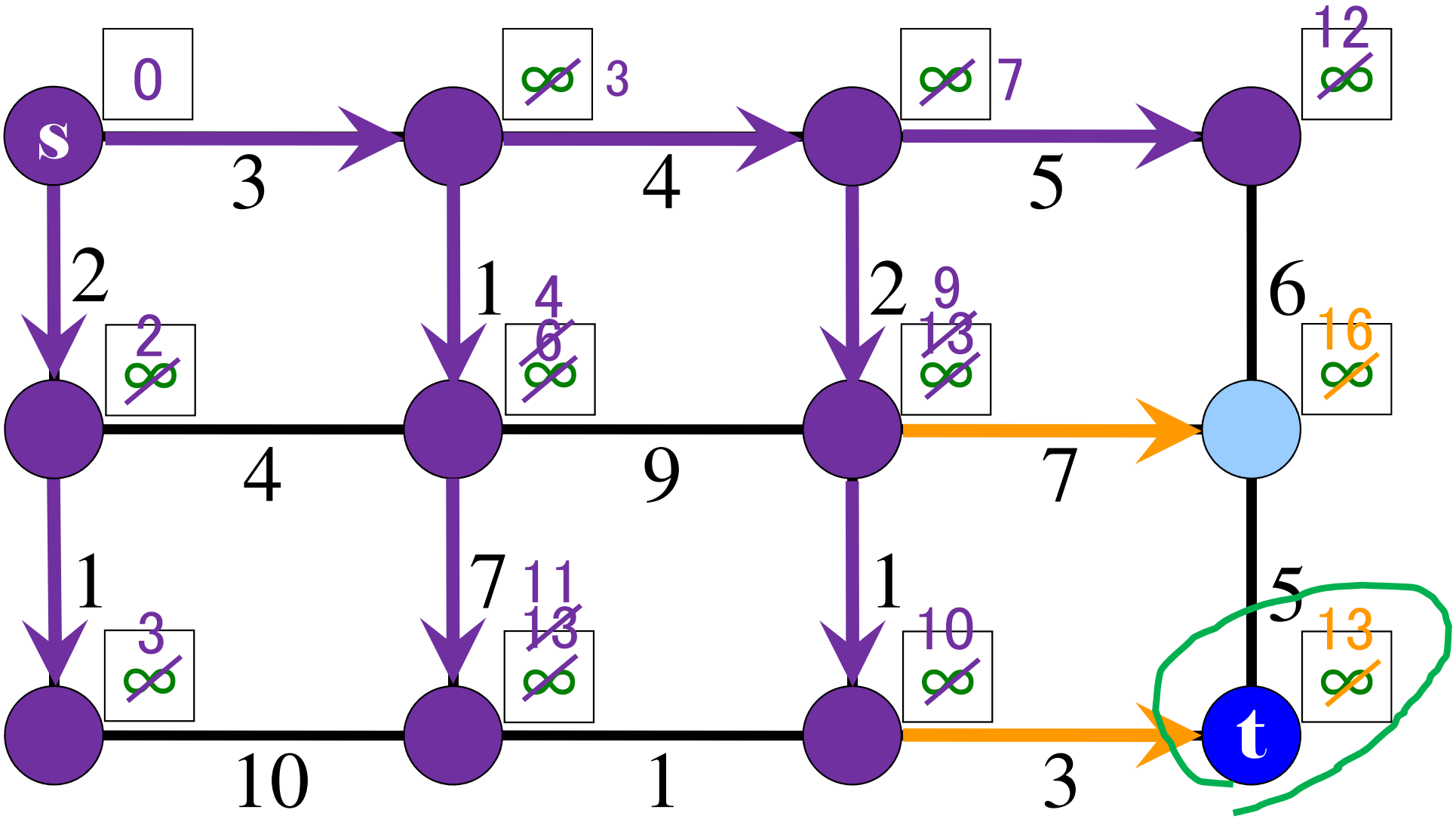
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



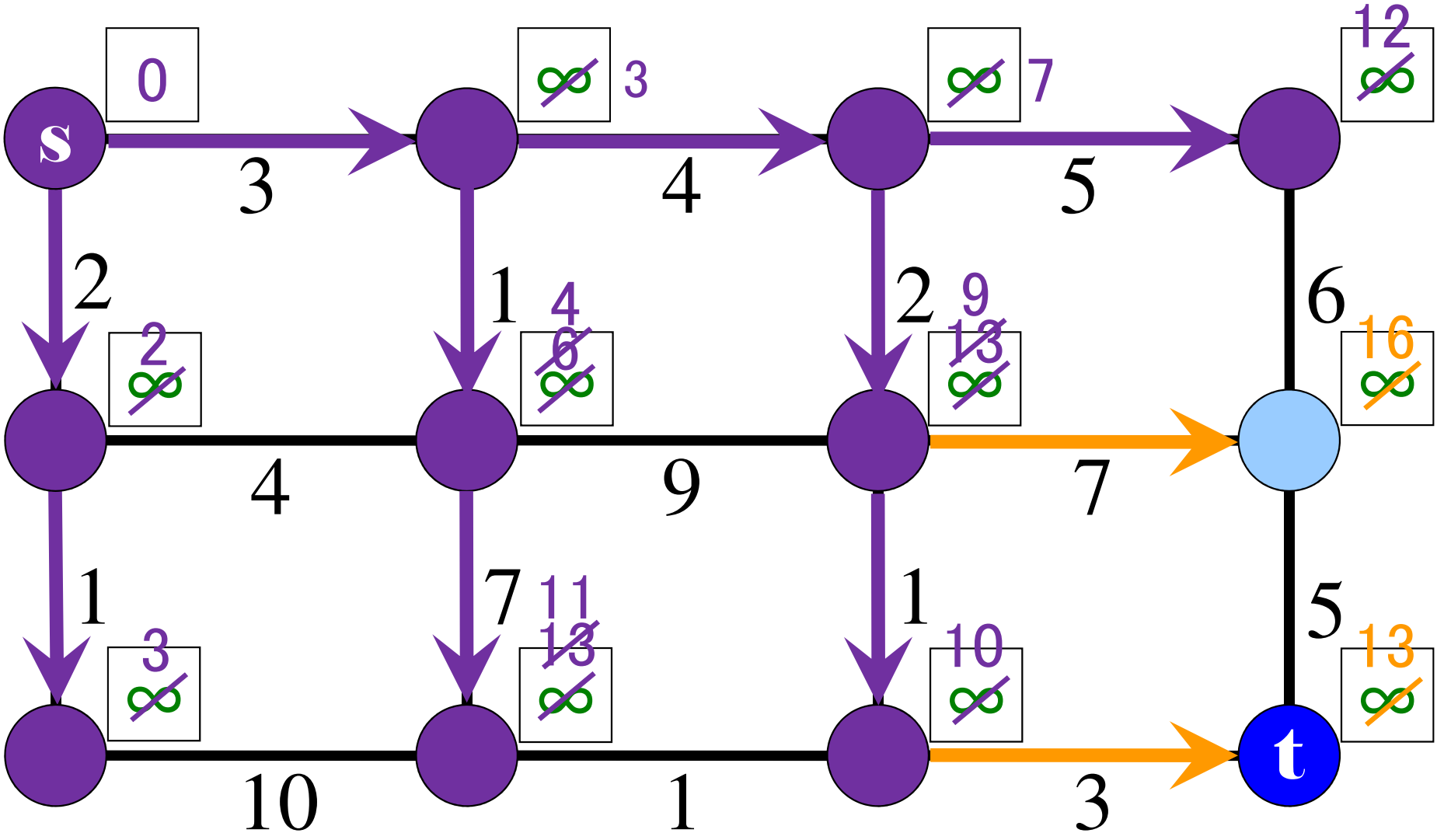
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

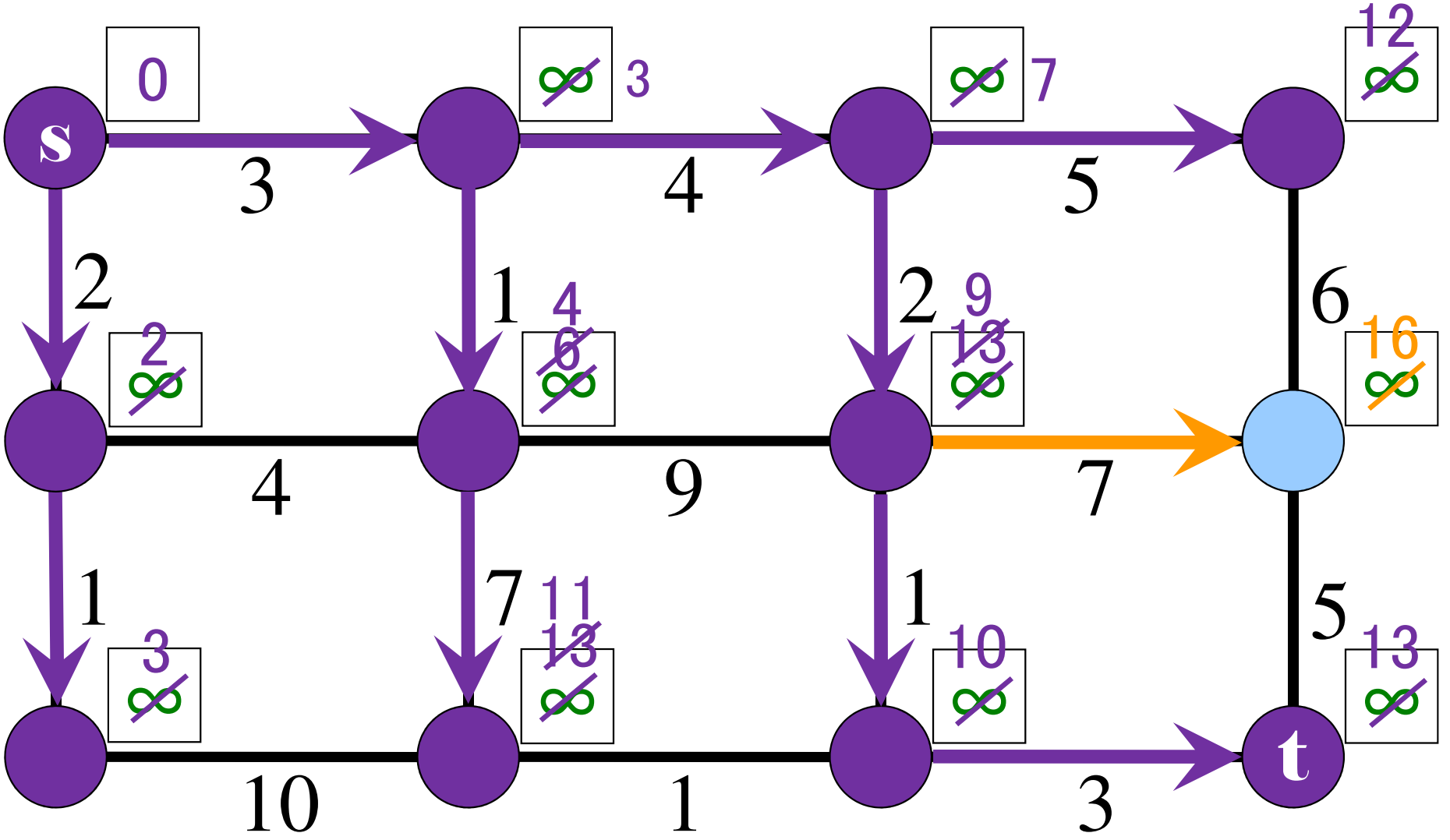


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

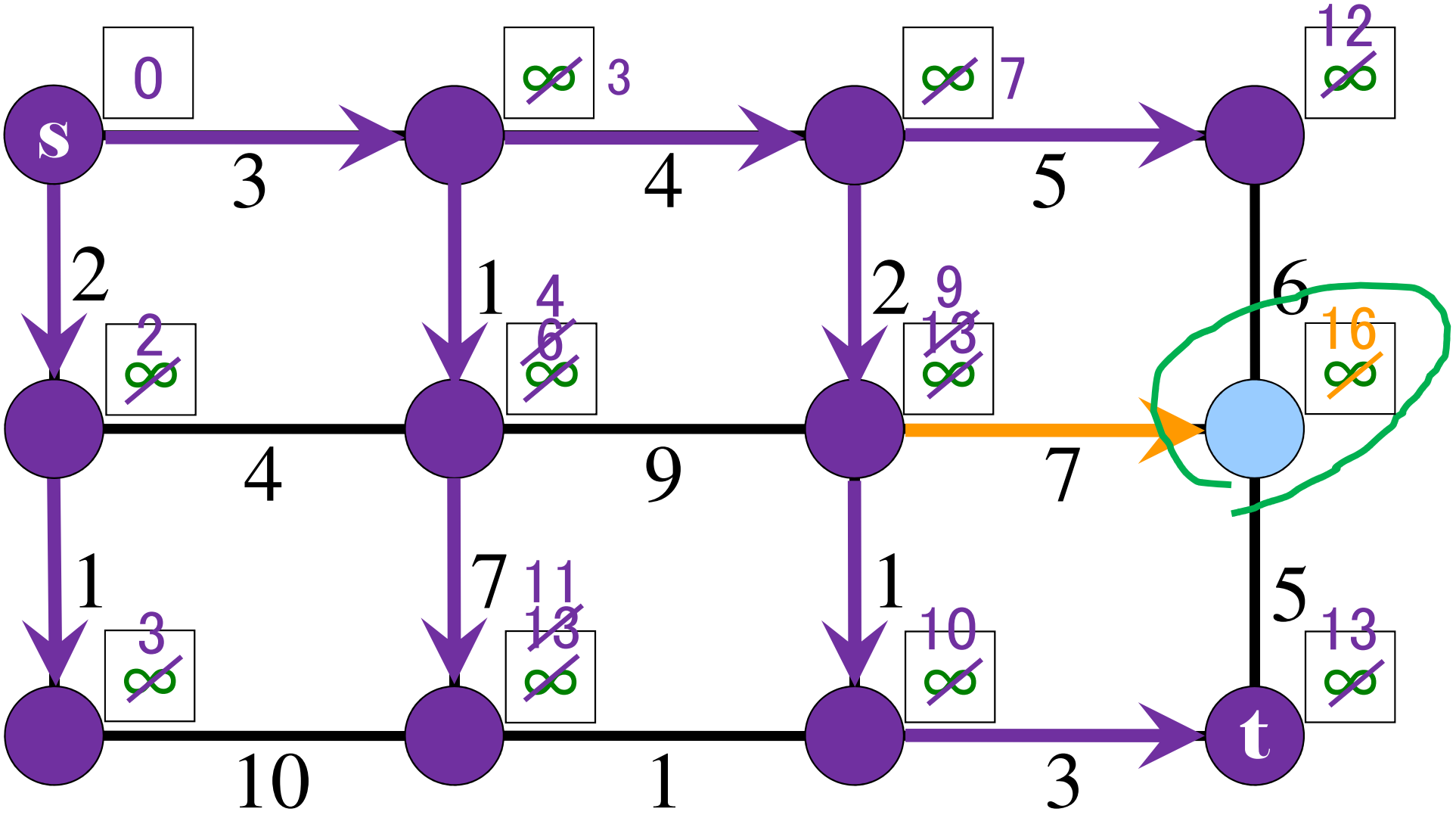
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



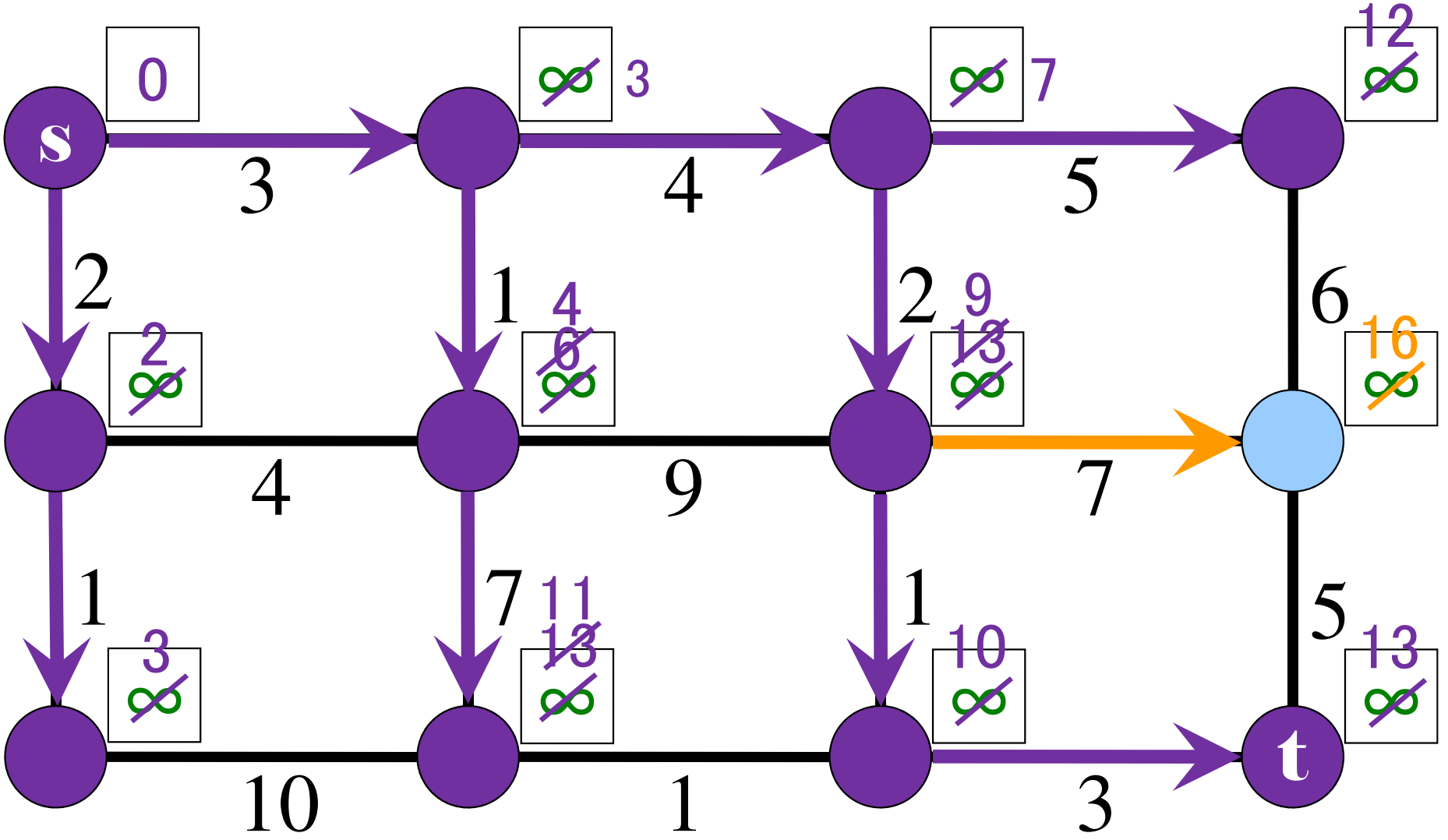
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

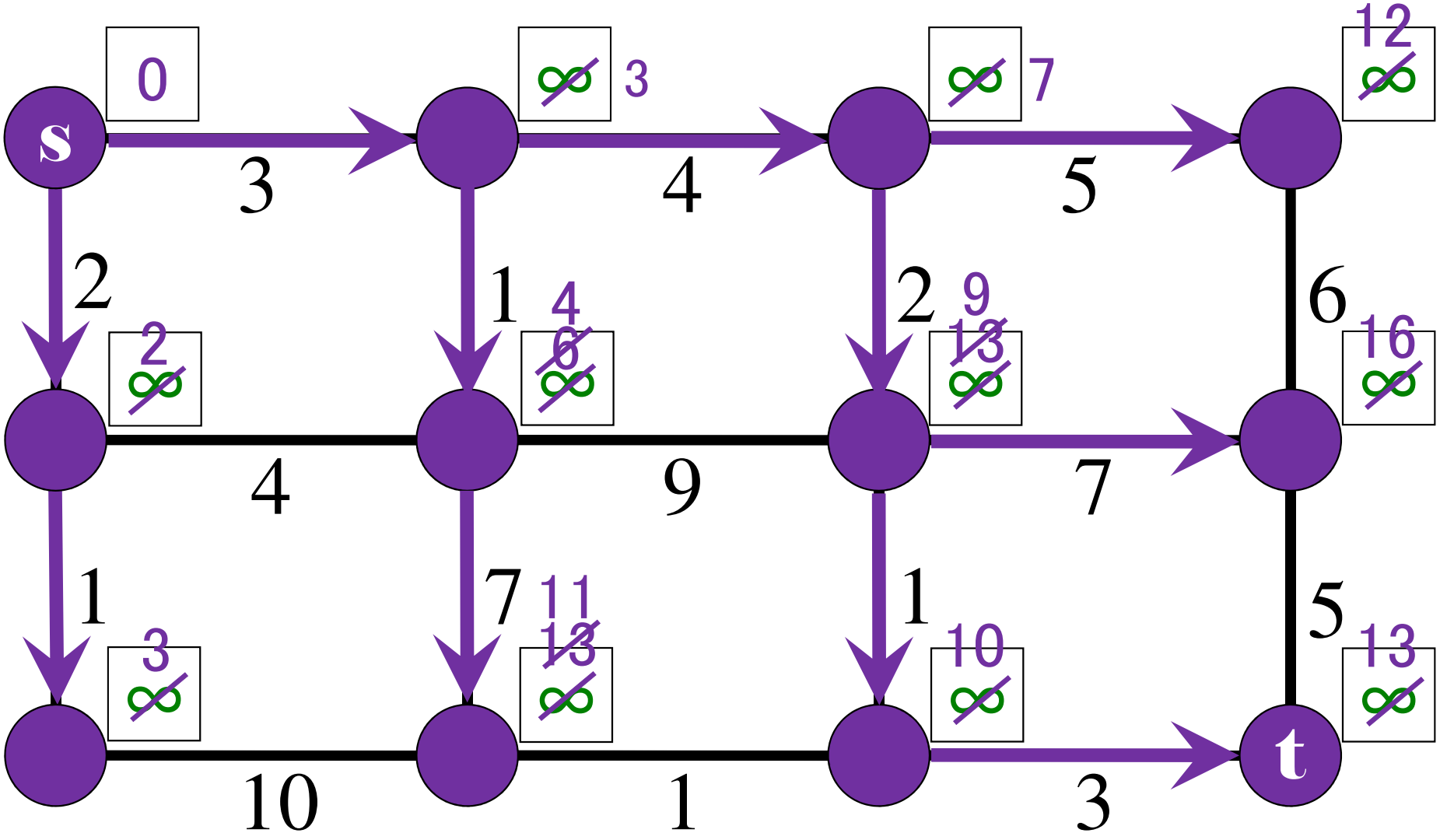


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

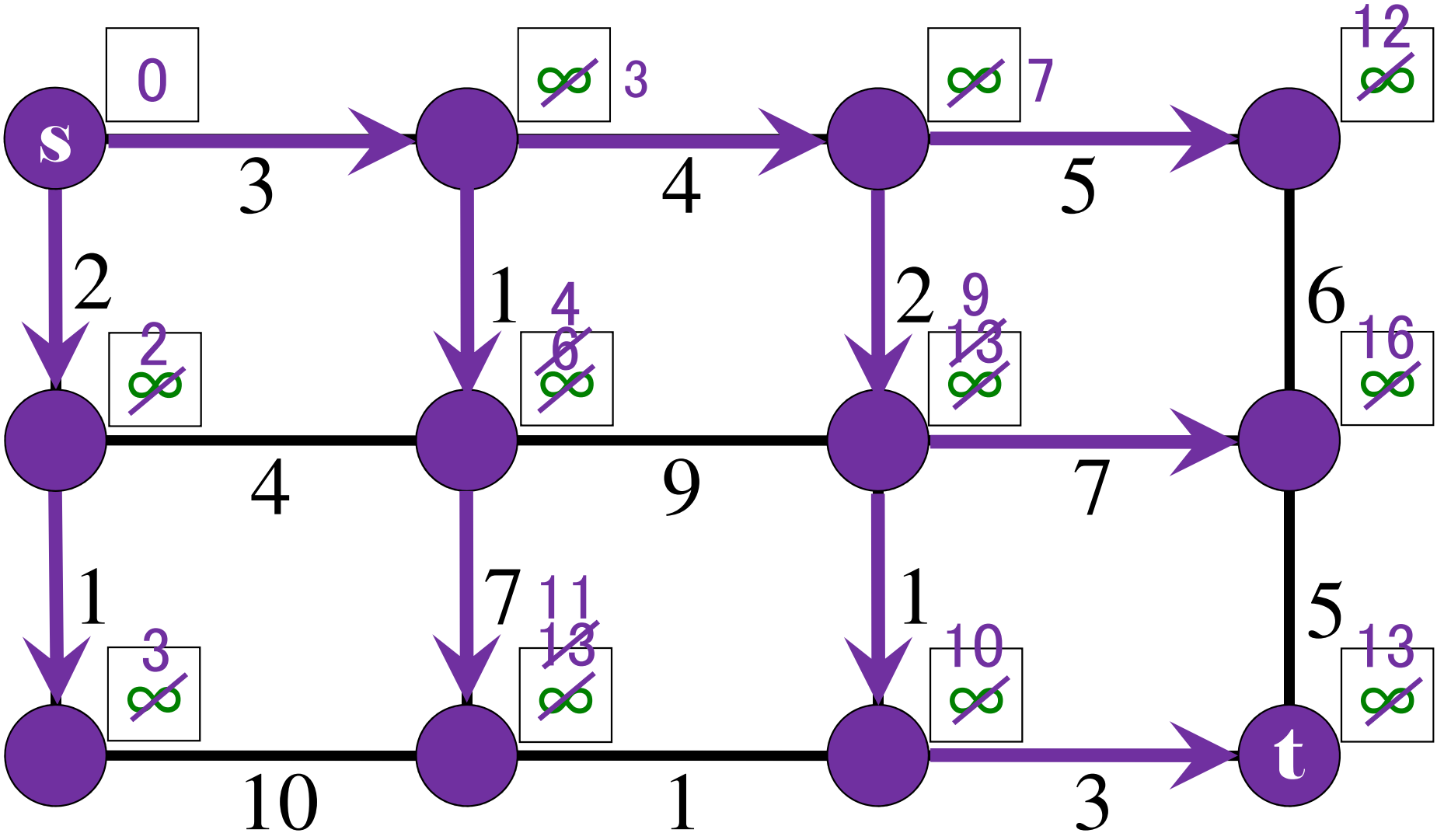
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



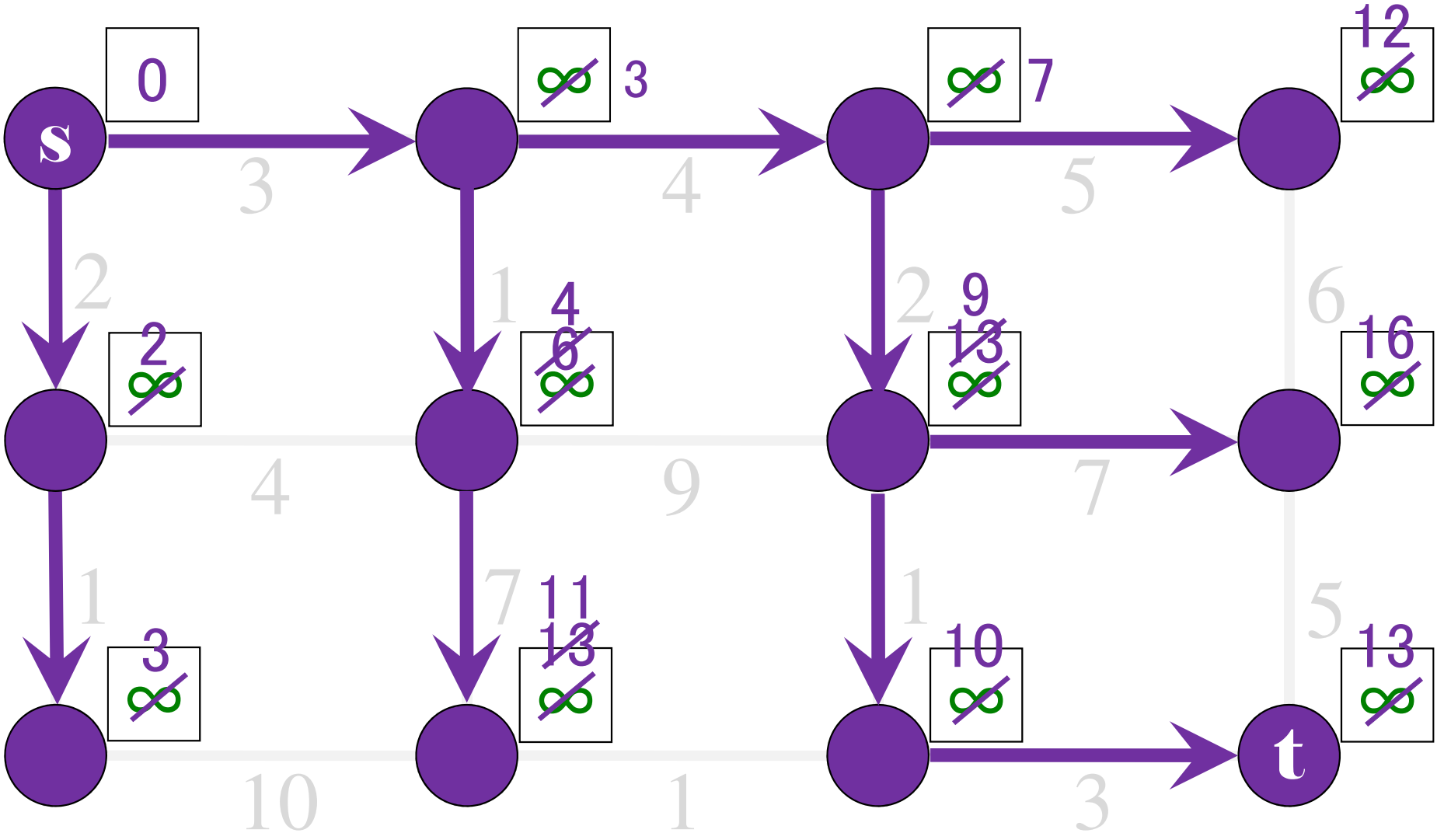
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



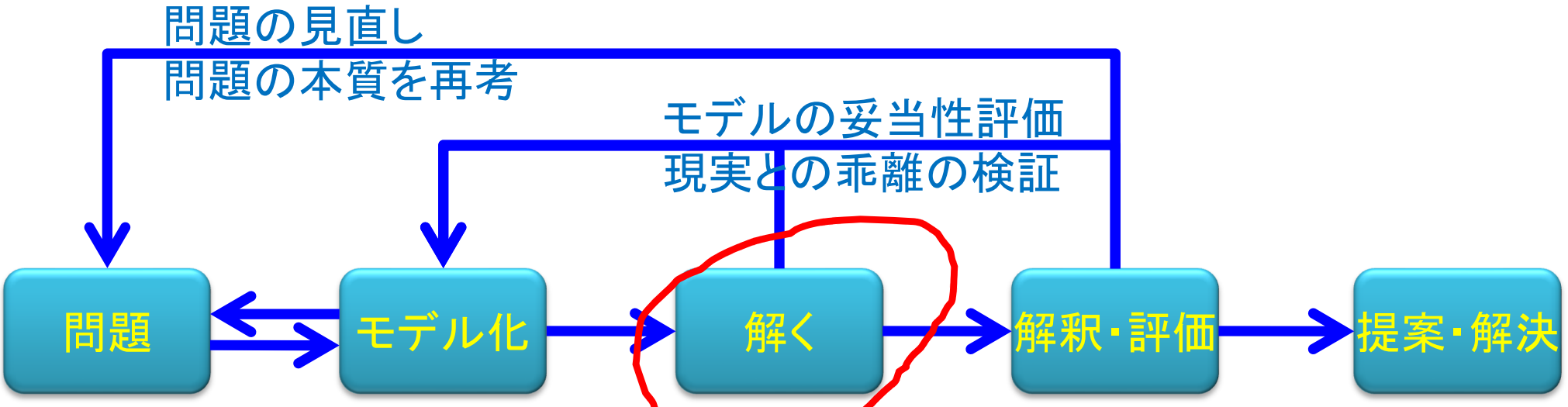
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

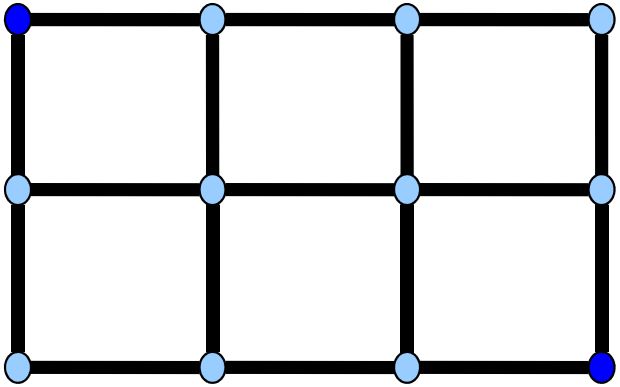
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

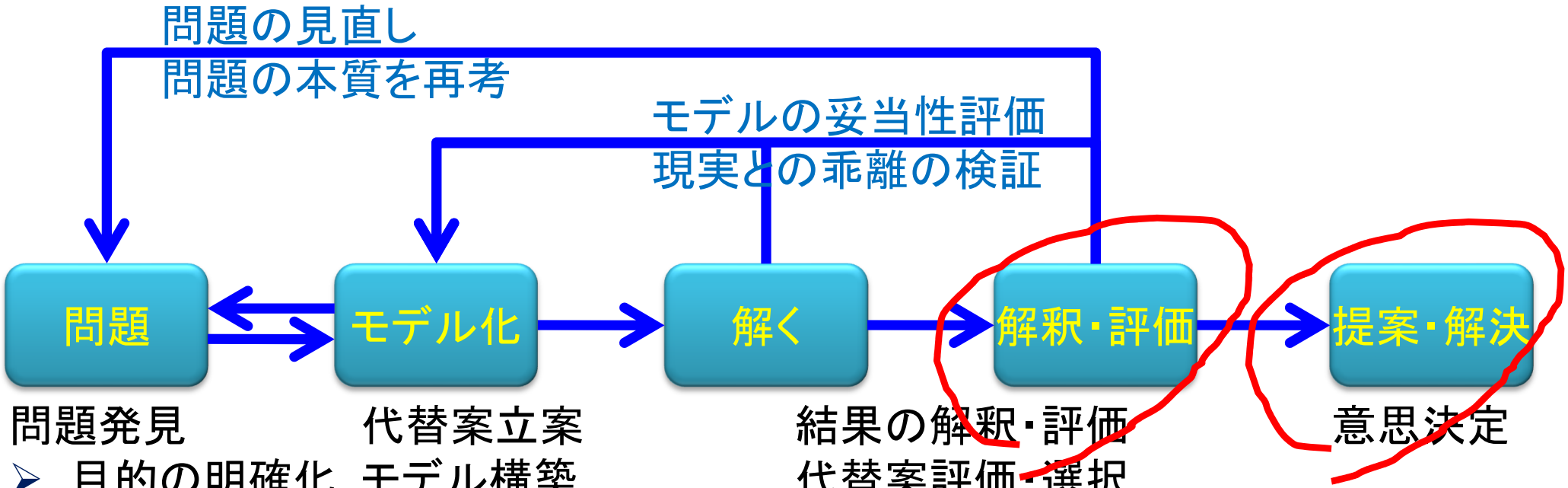
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

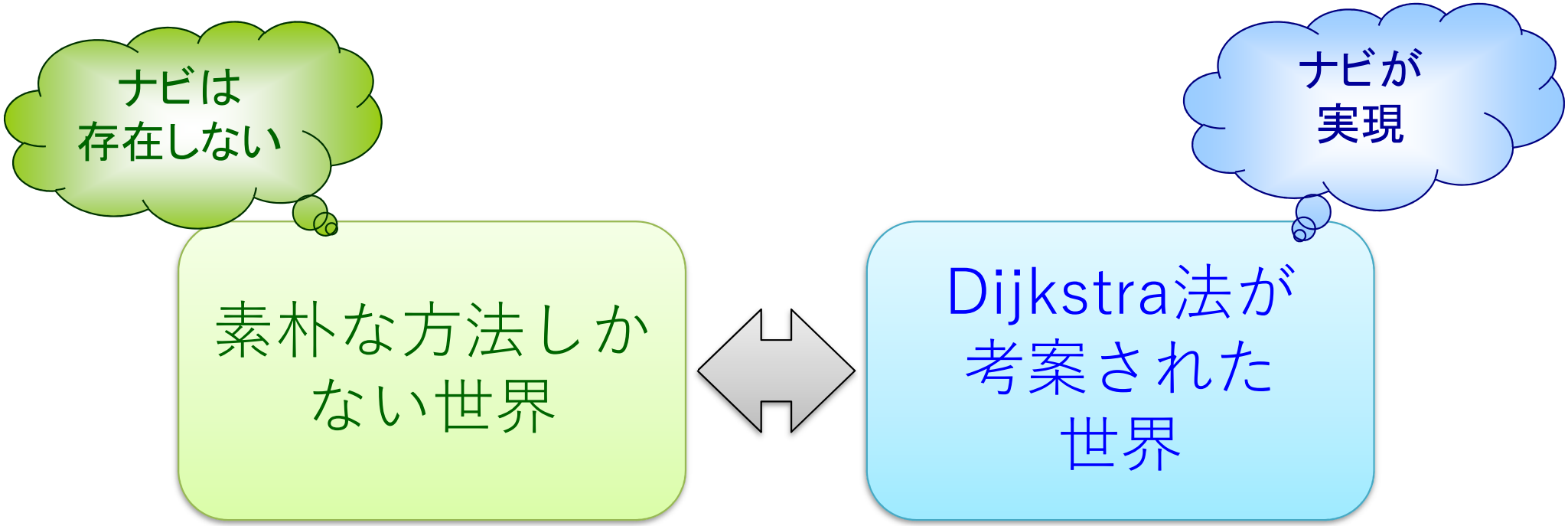
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

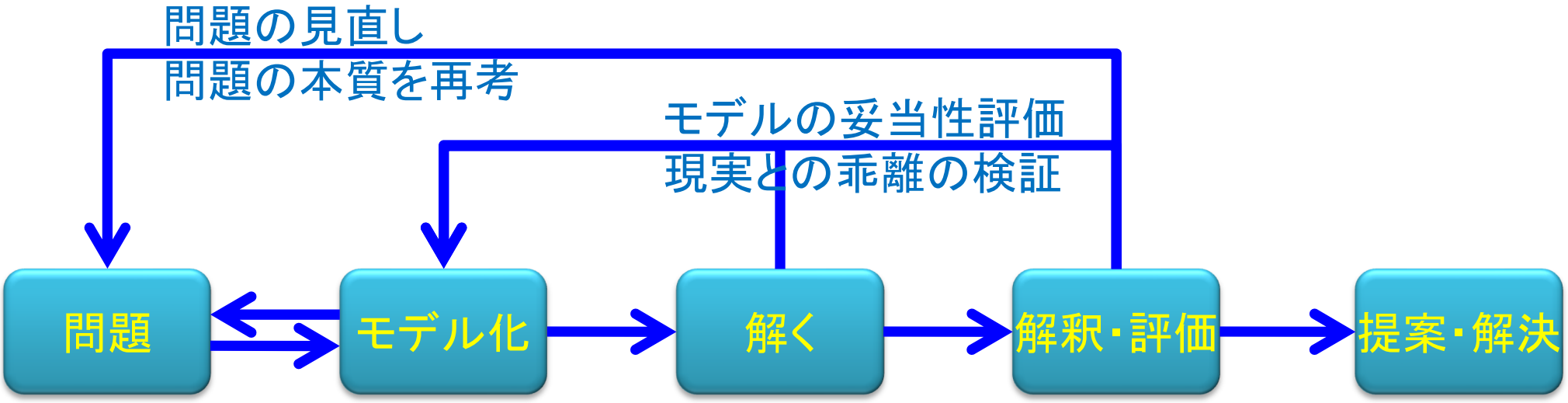
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



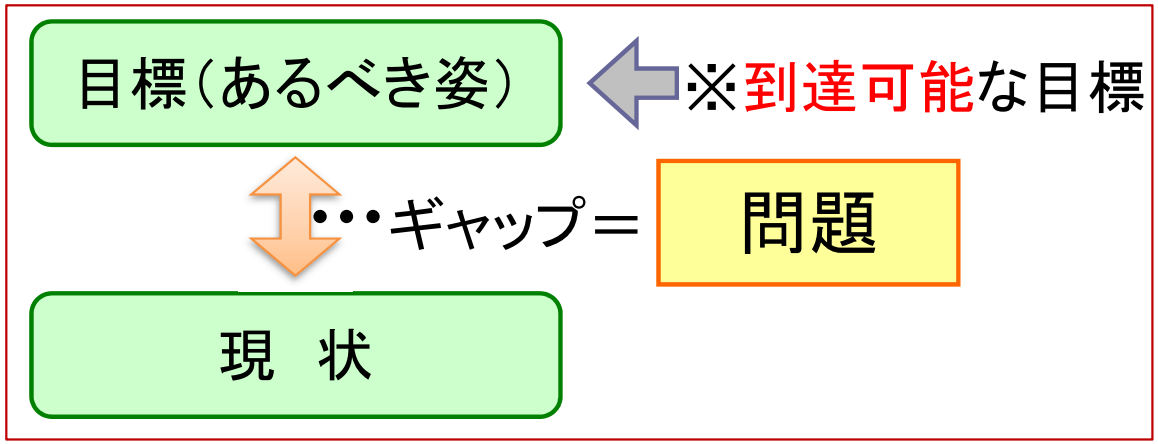
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

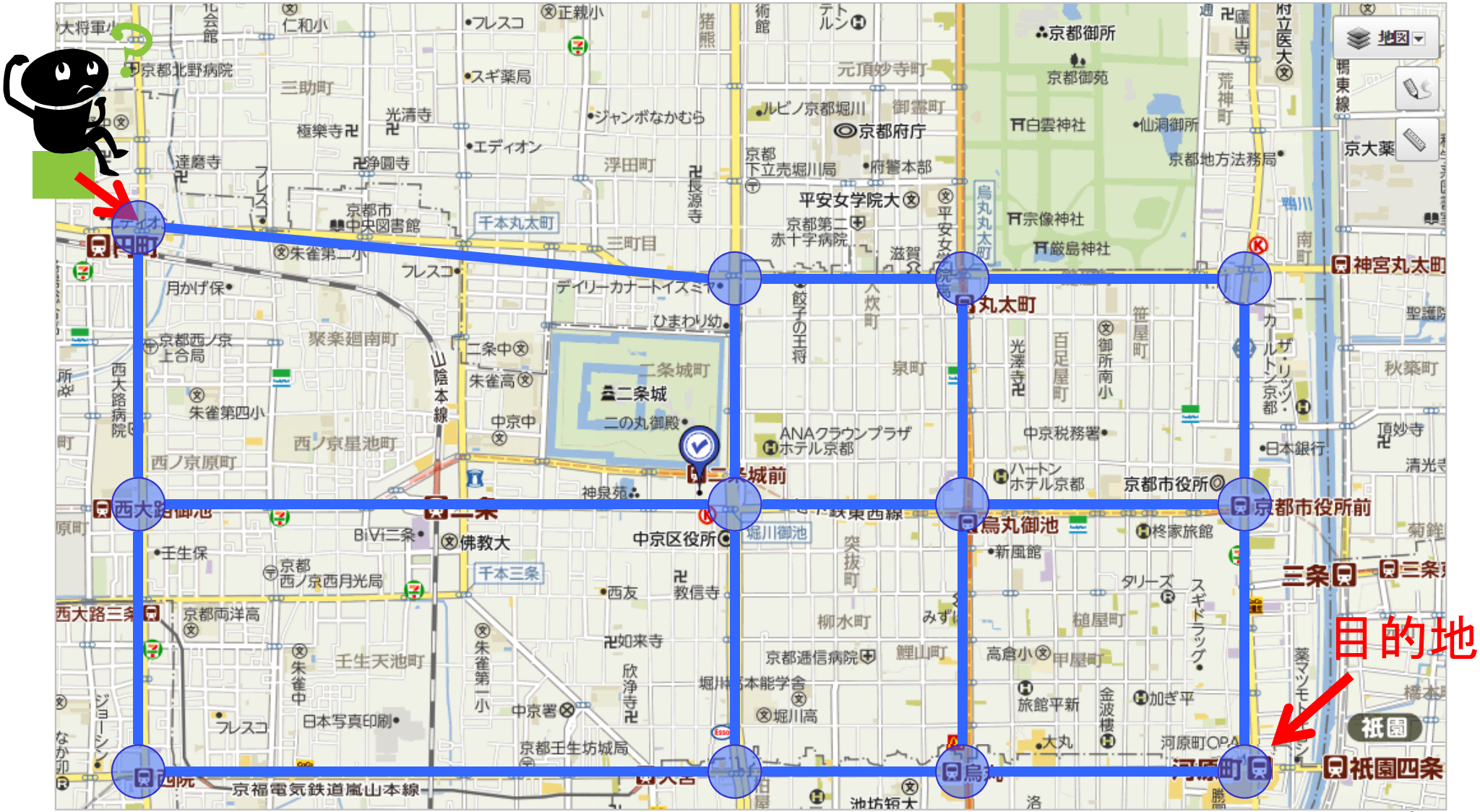
意思決定

問題の定義



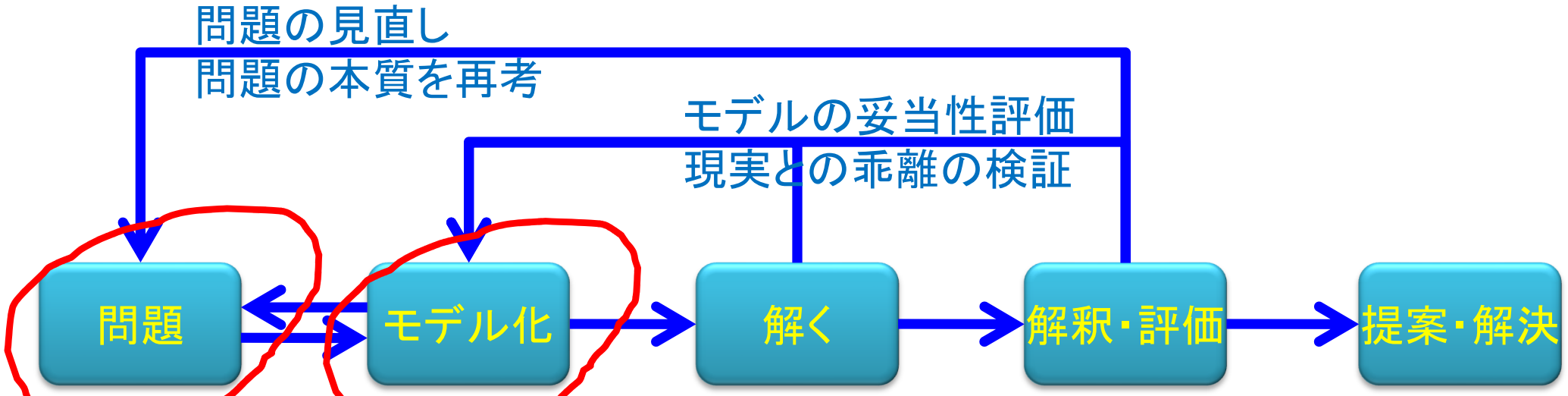
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

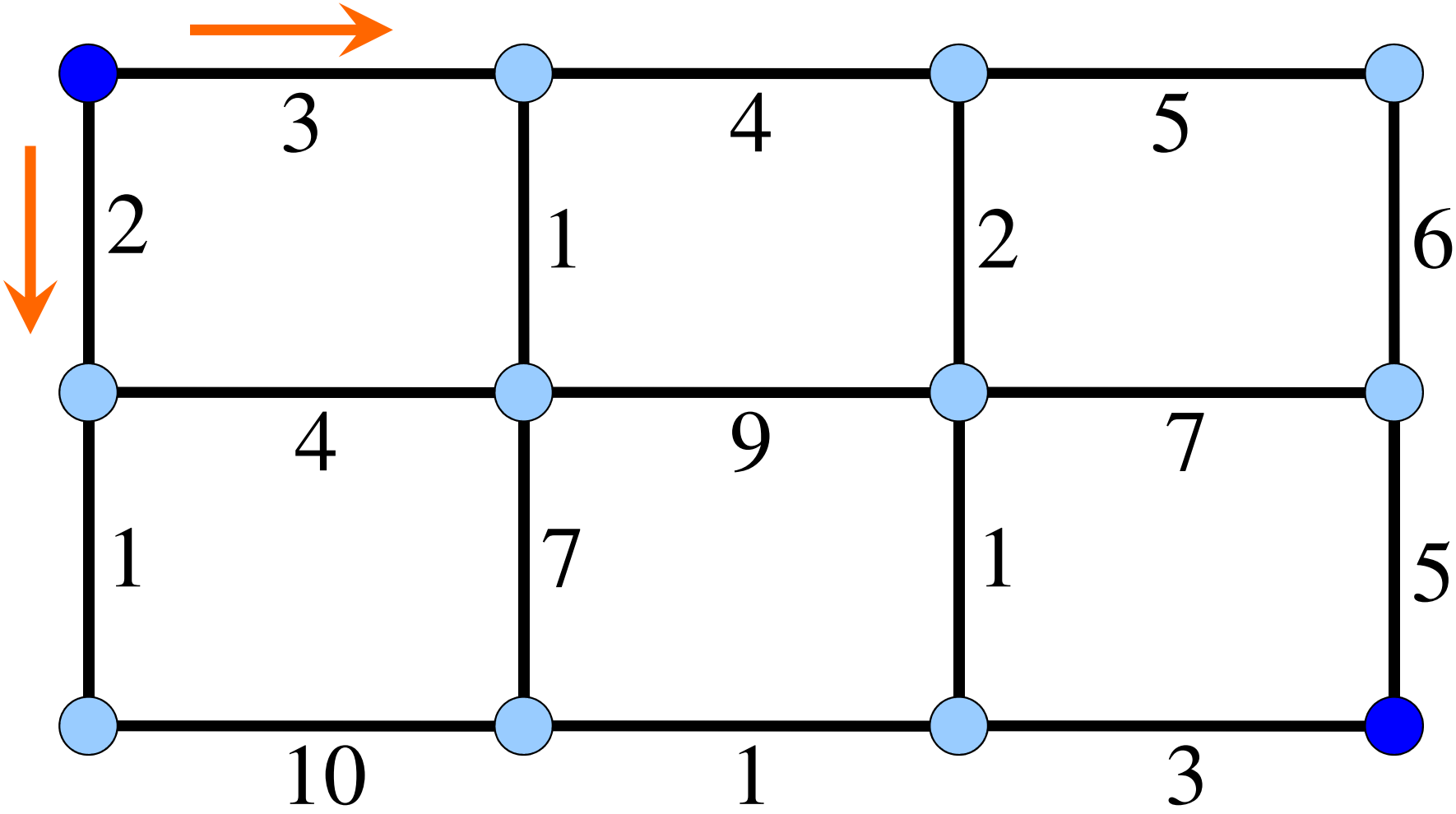
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

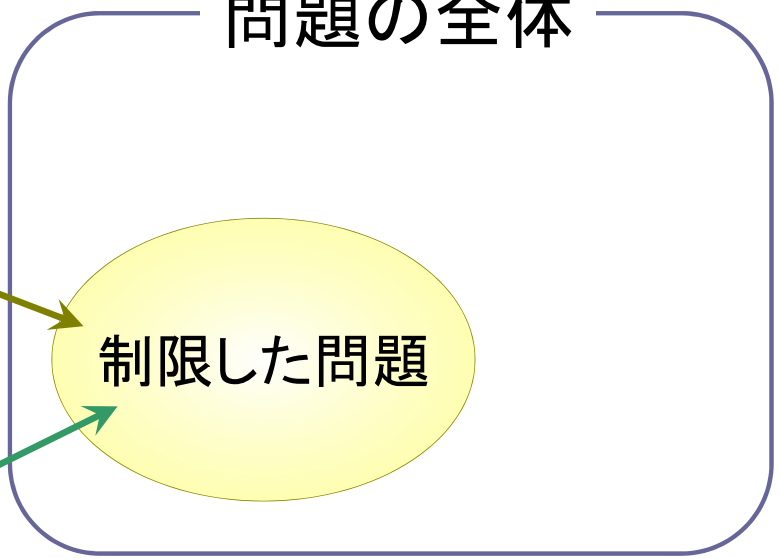


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

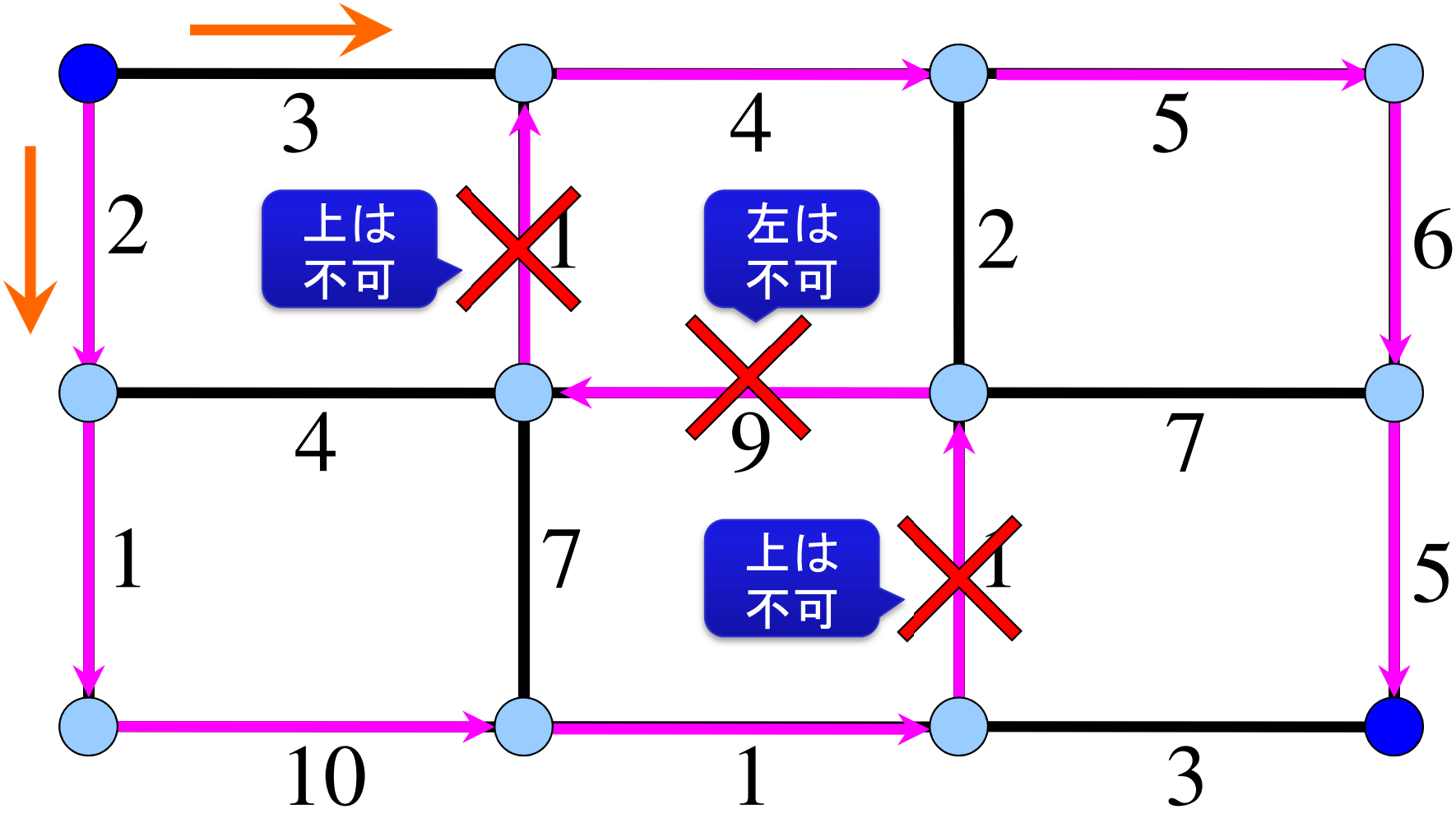
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

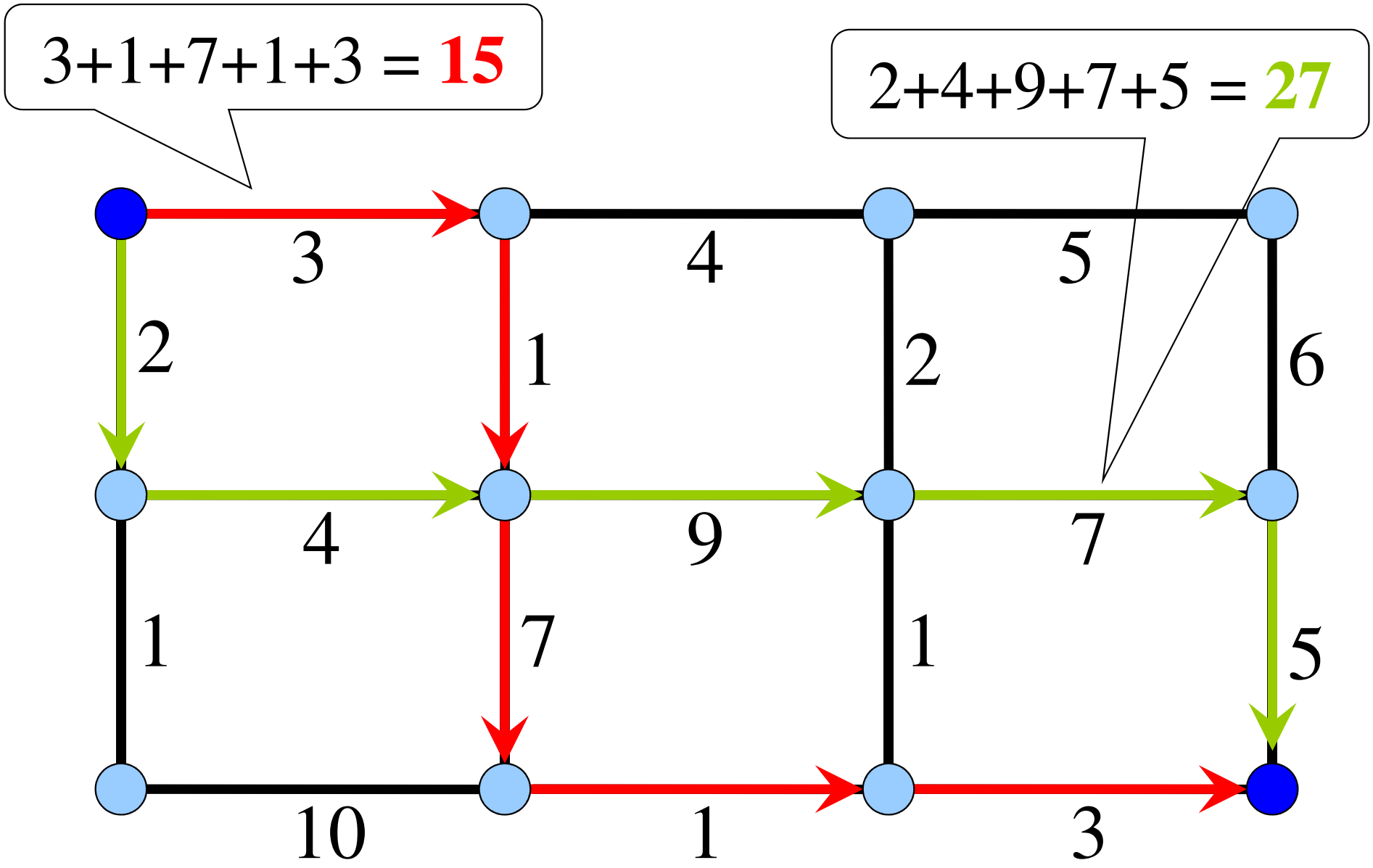
難しいなら易しくすればいいのさ！

制限した問題

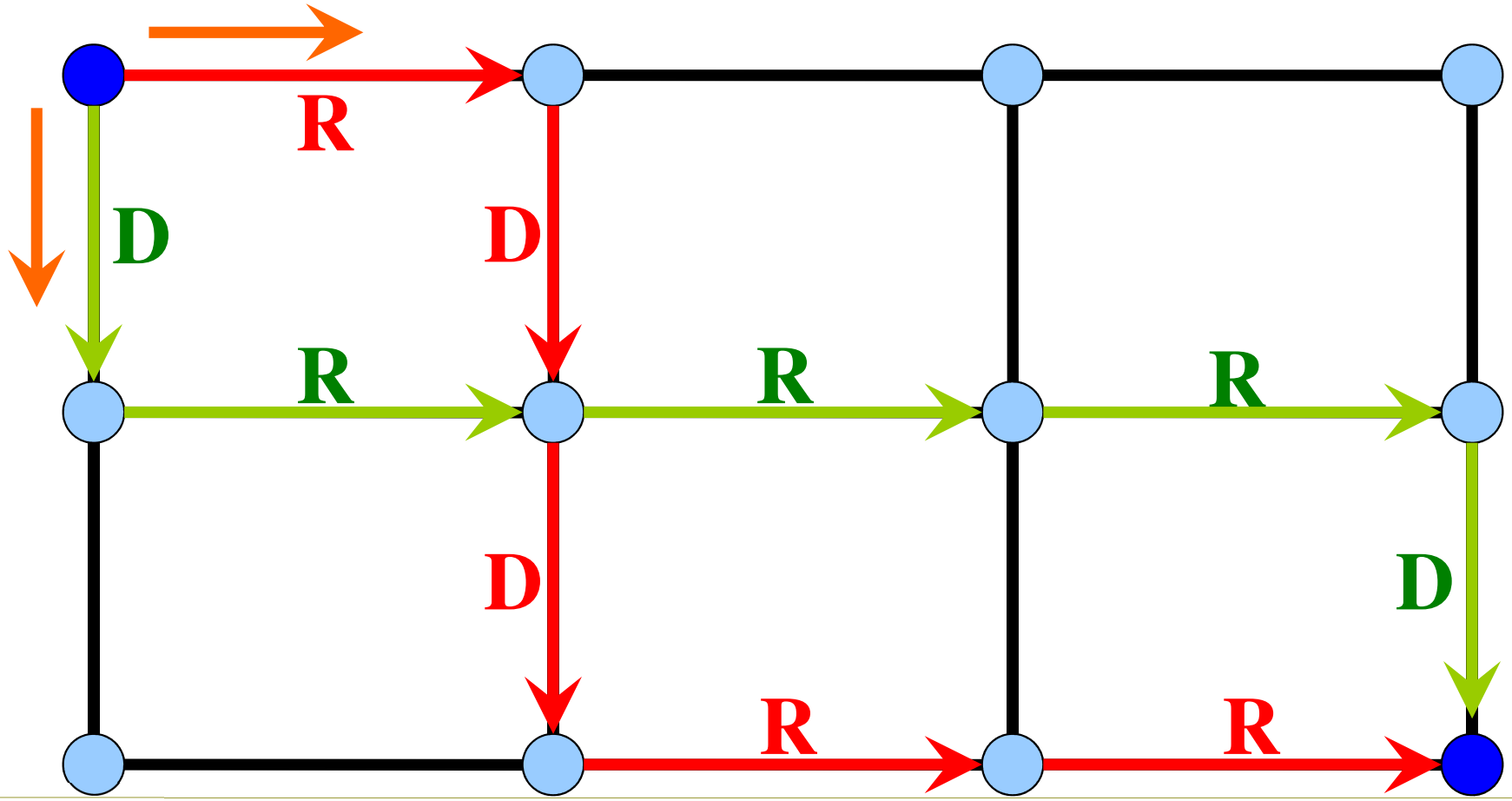
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



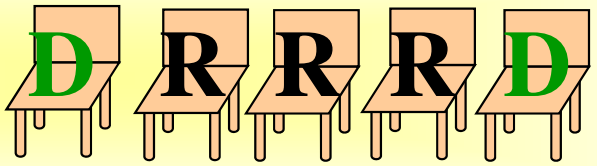
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



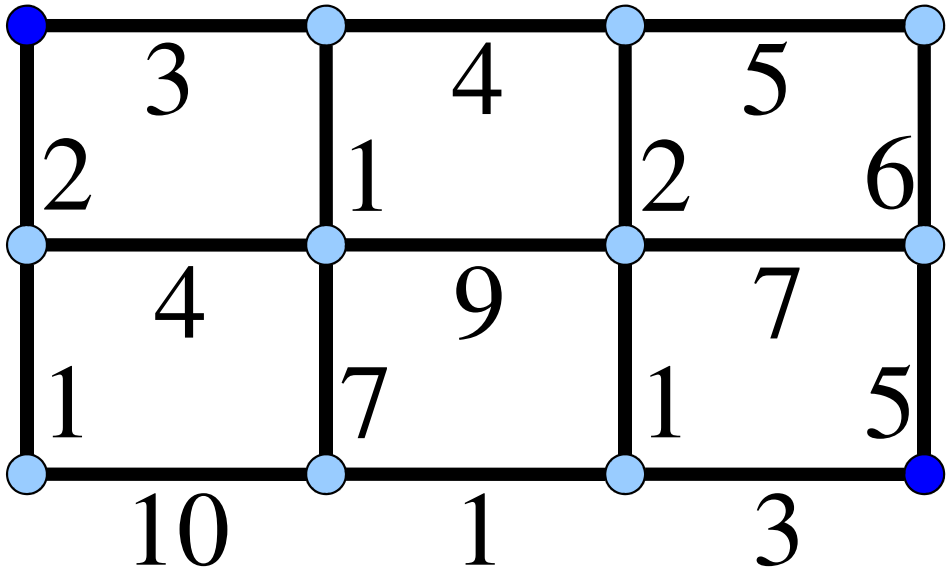
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

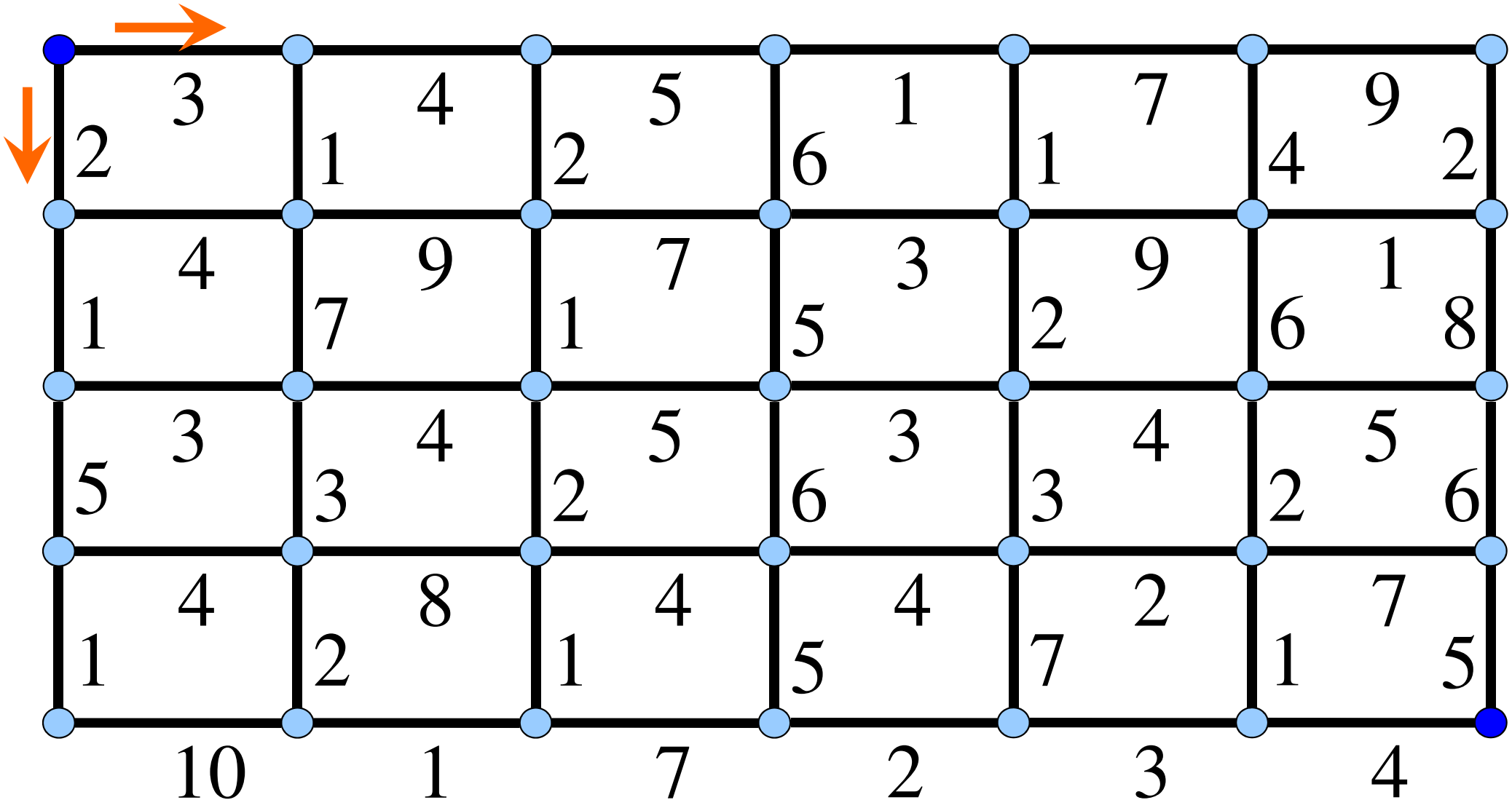
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

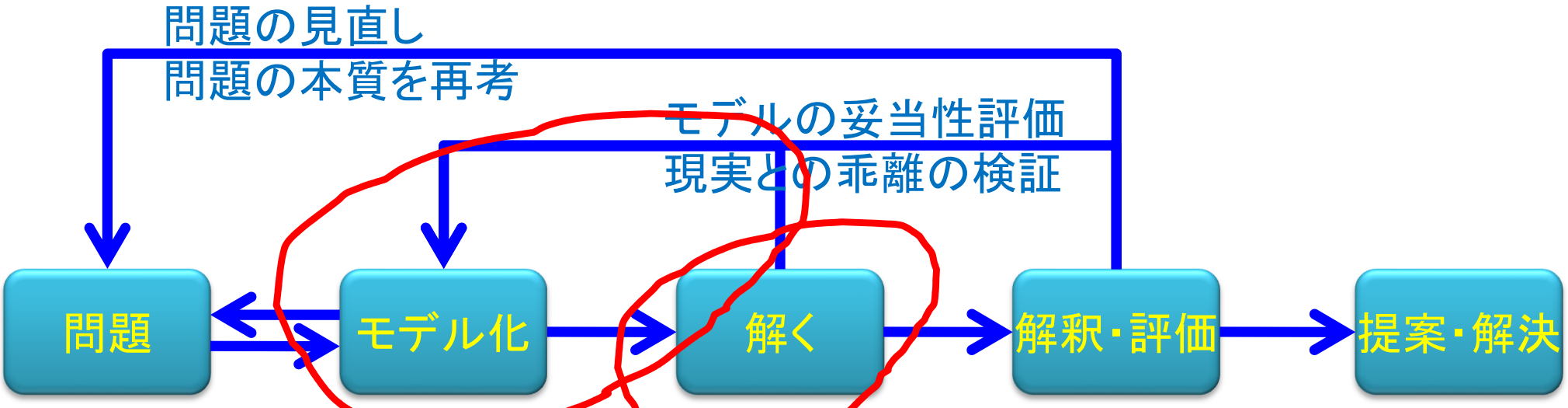
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

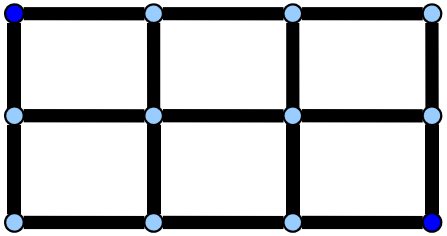
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

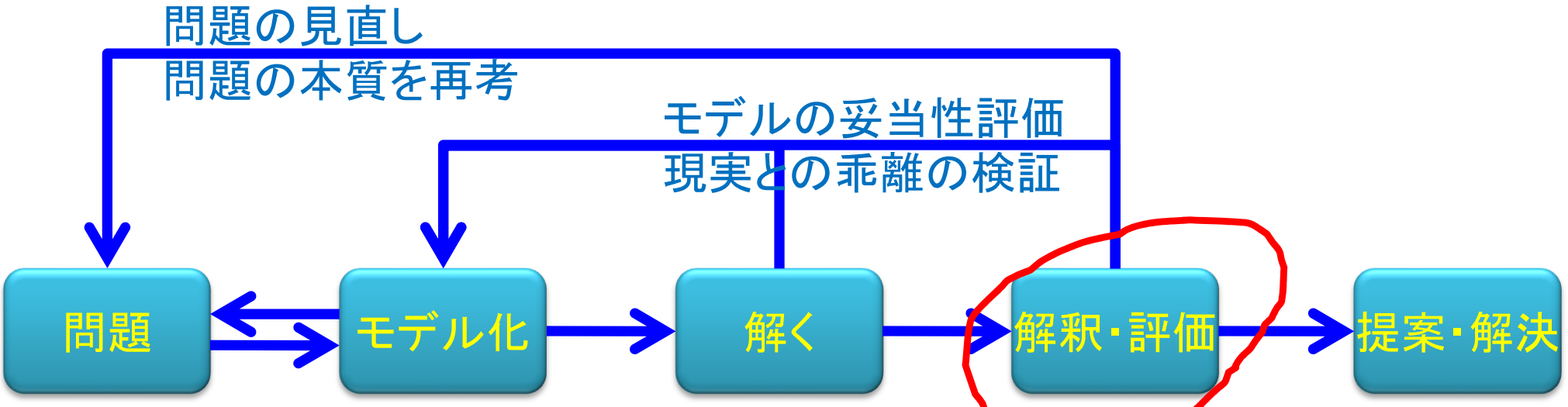
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

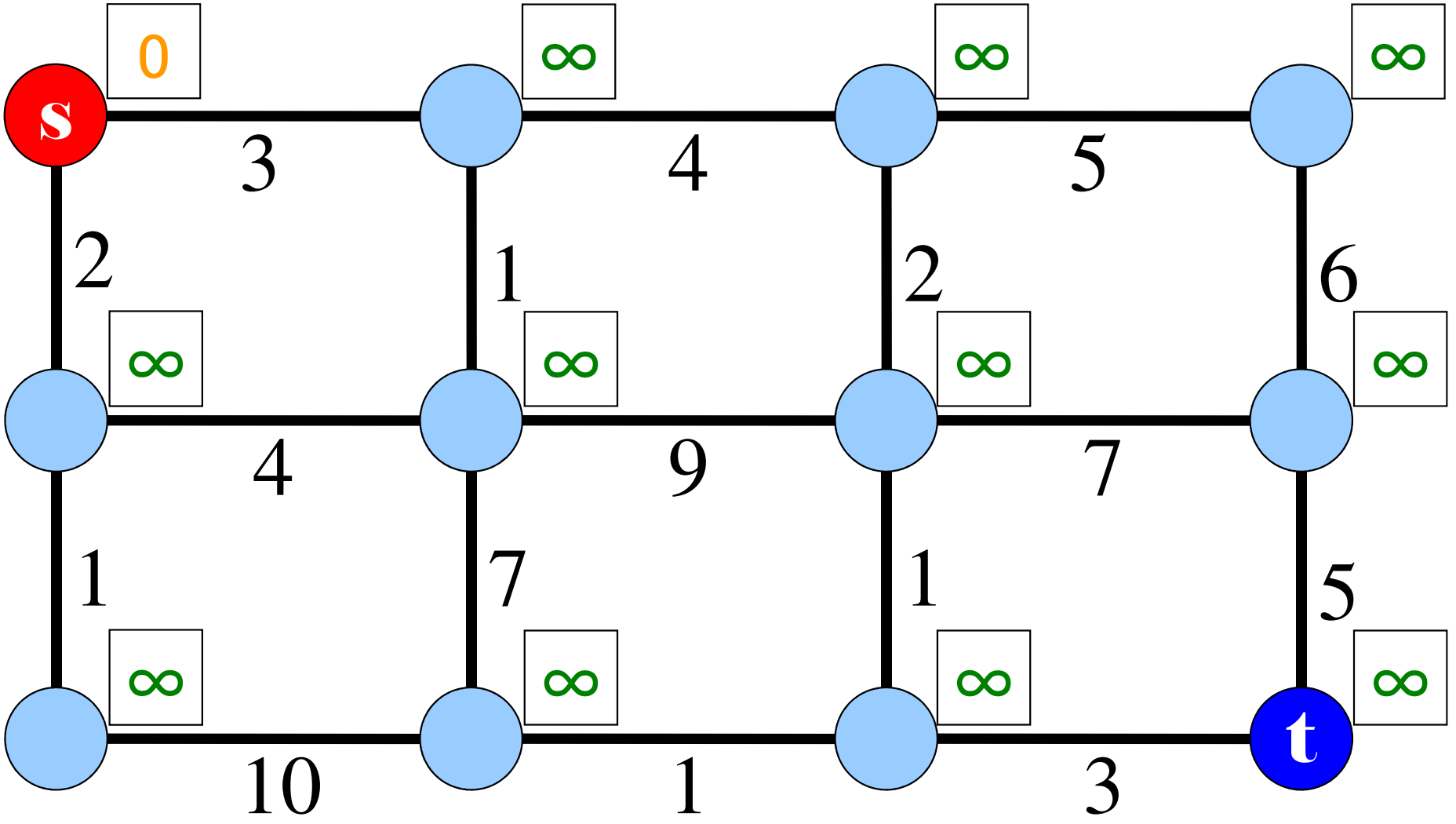
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

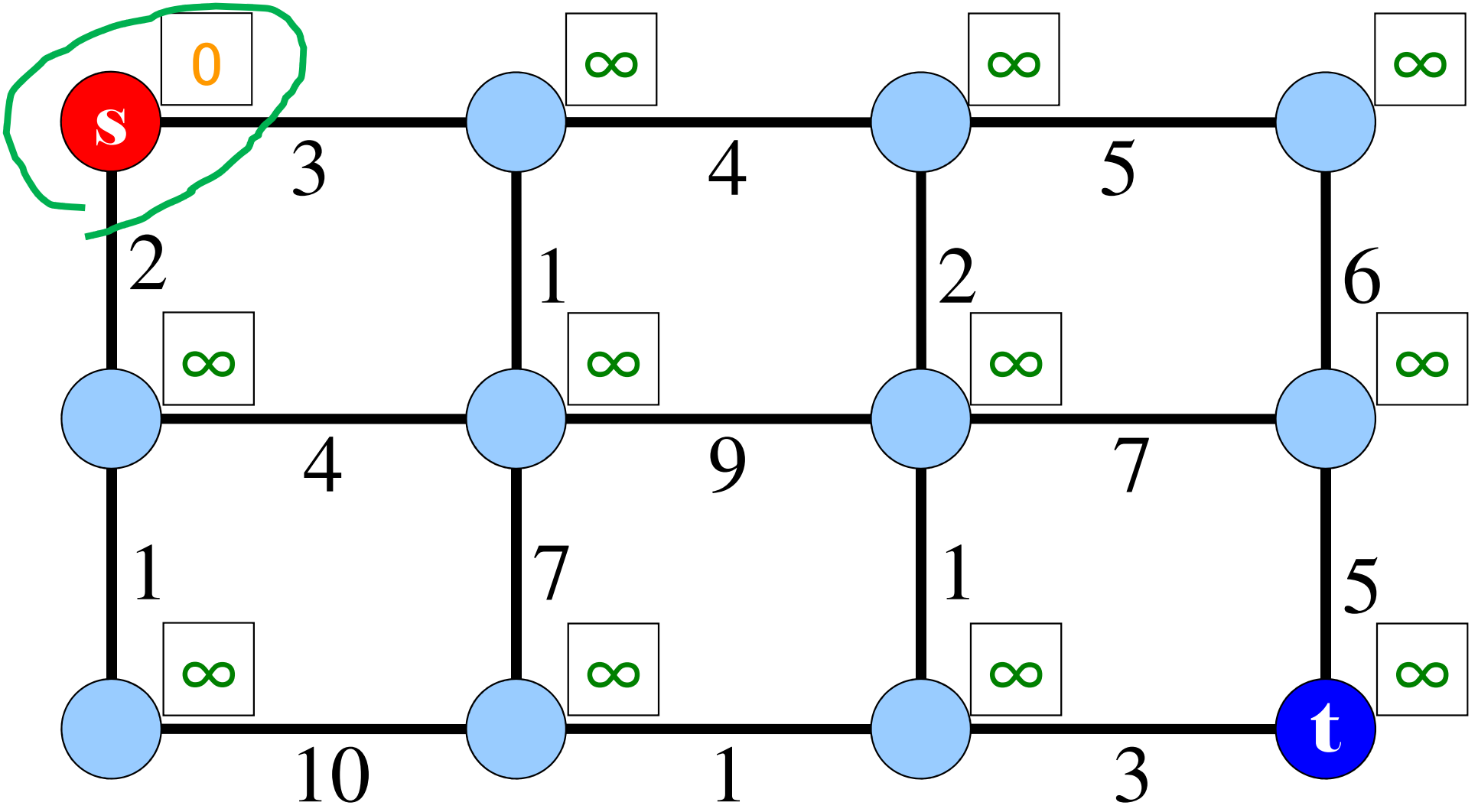
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



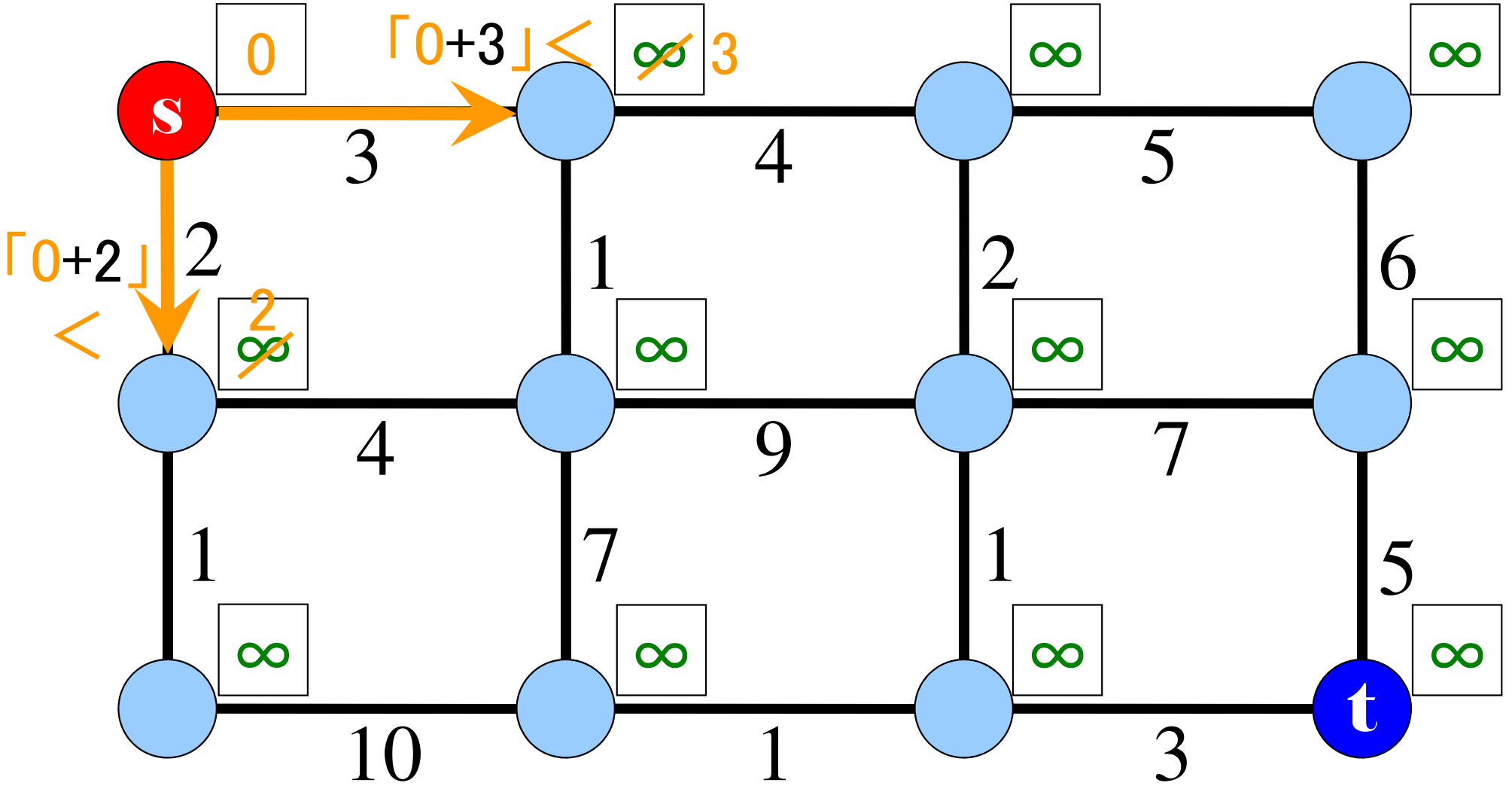
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

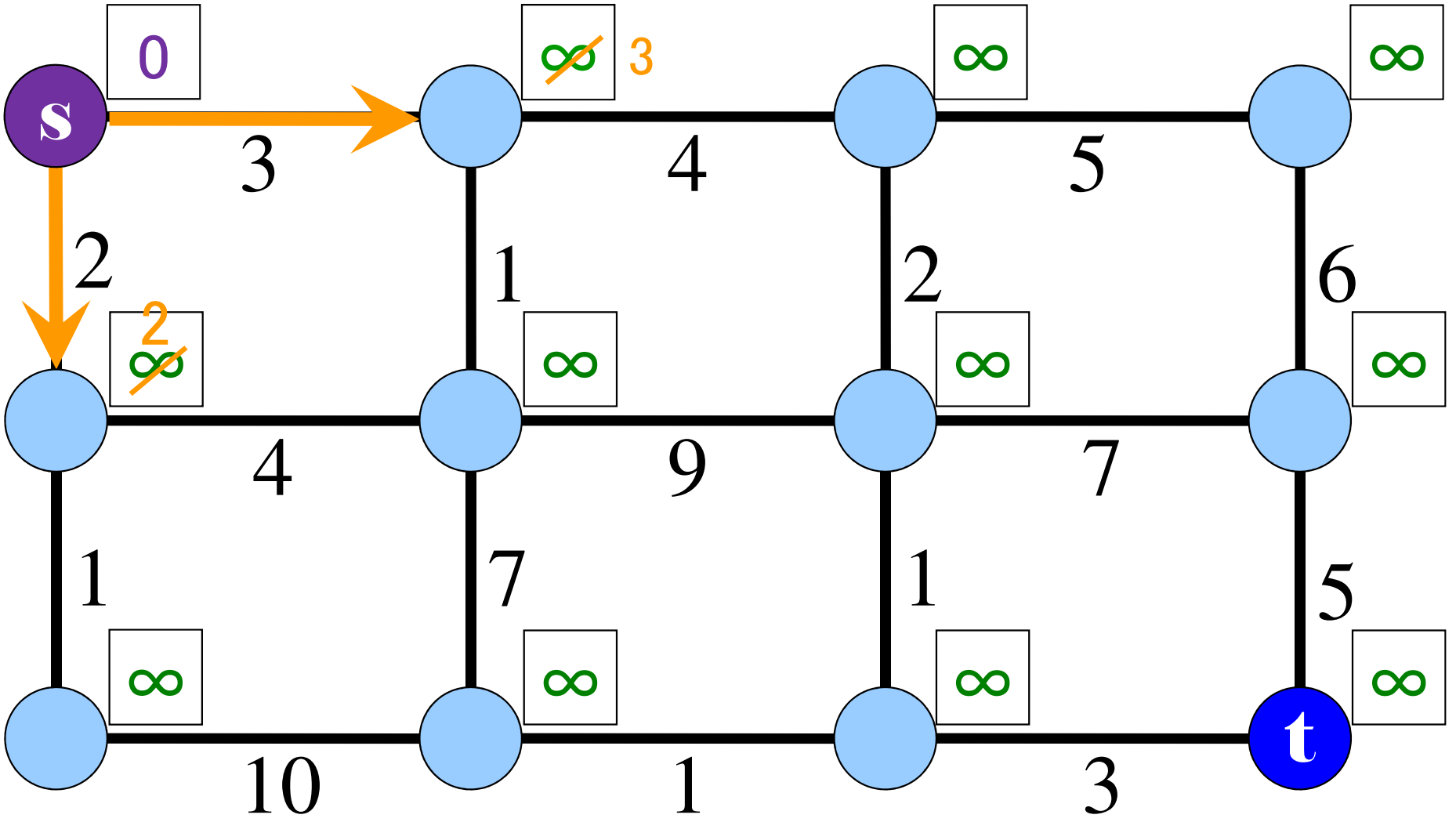


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

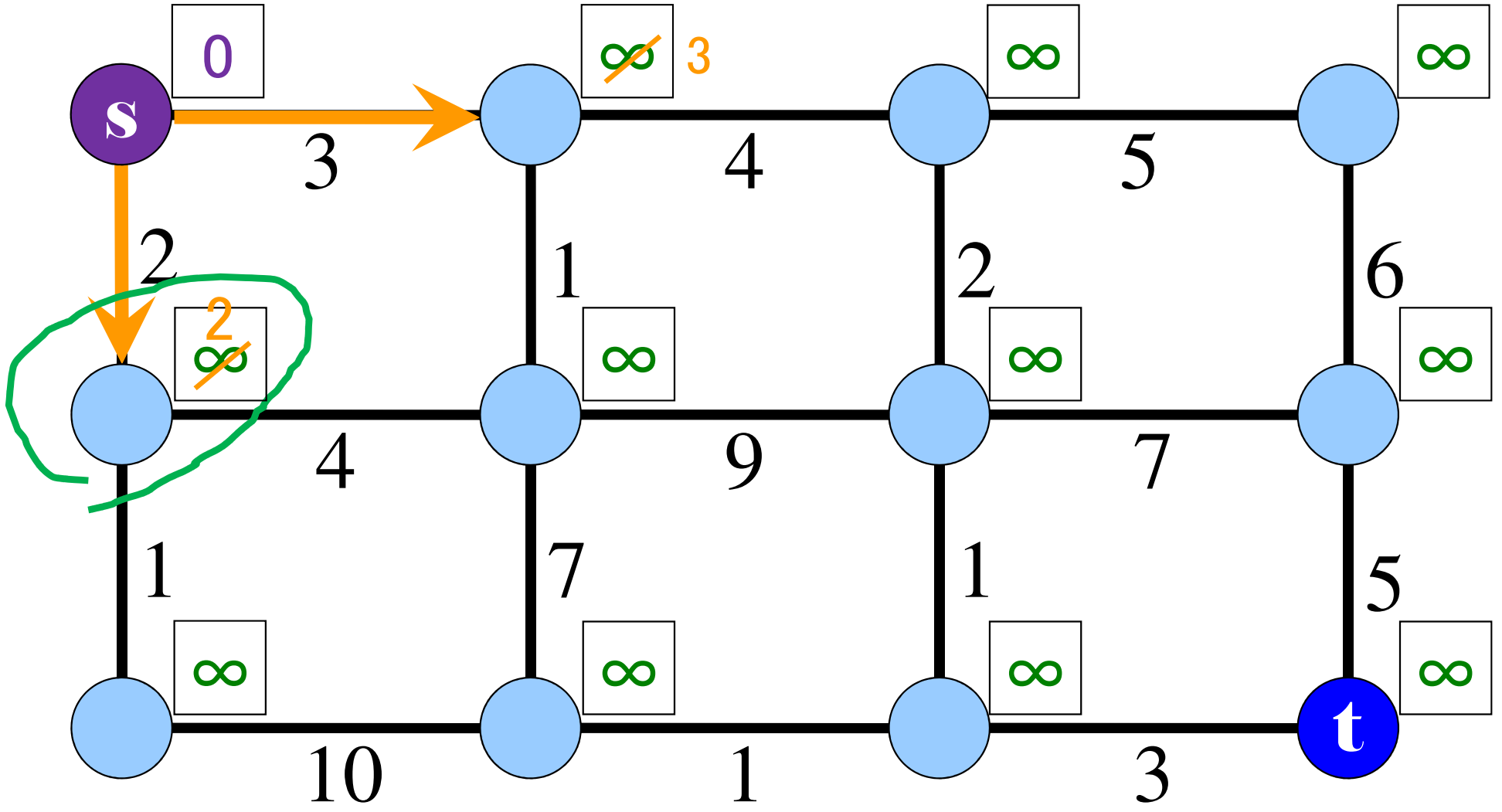
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



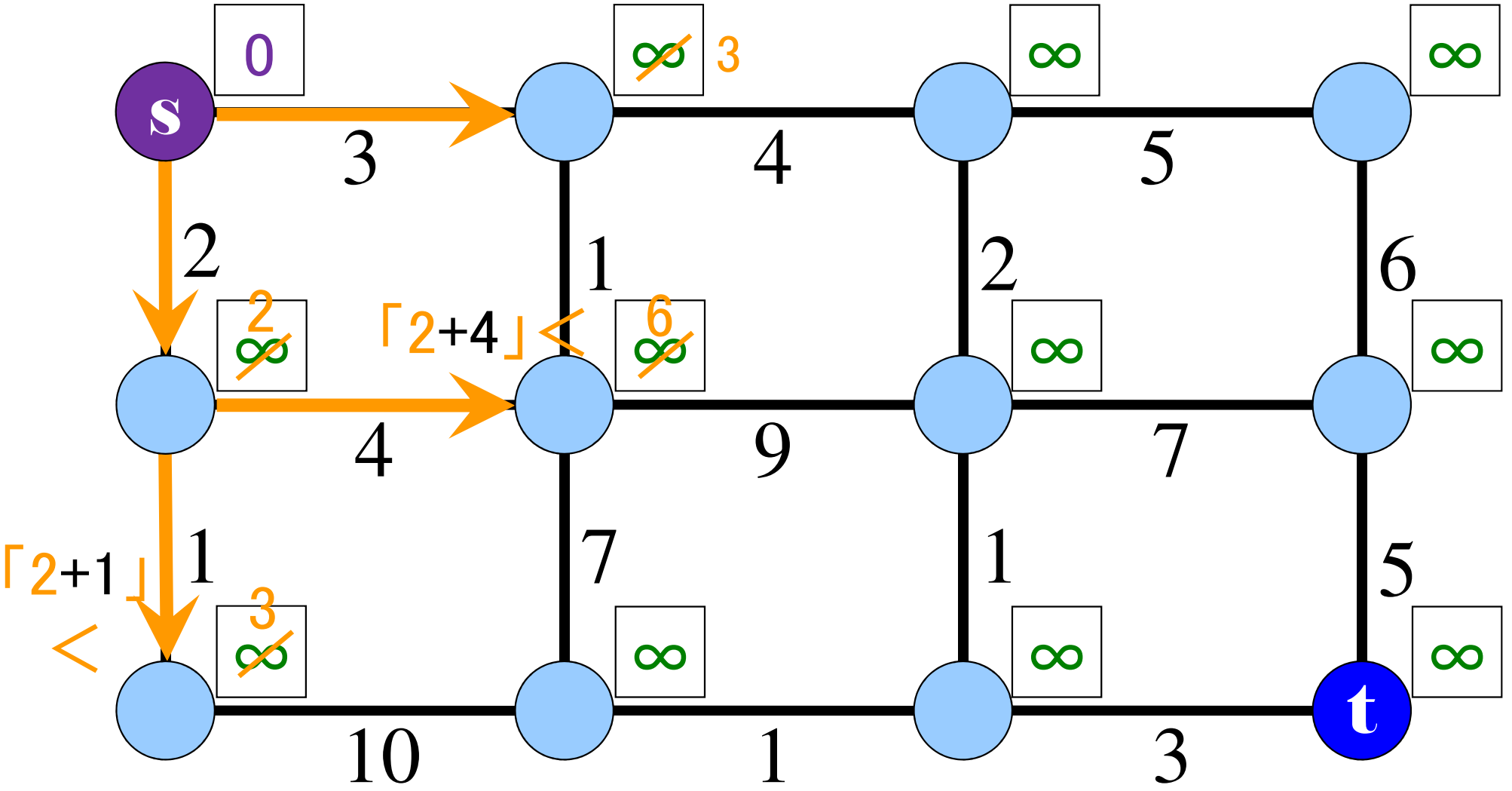
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

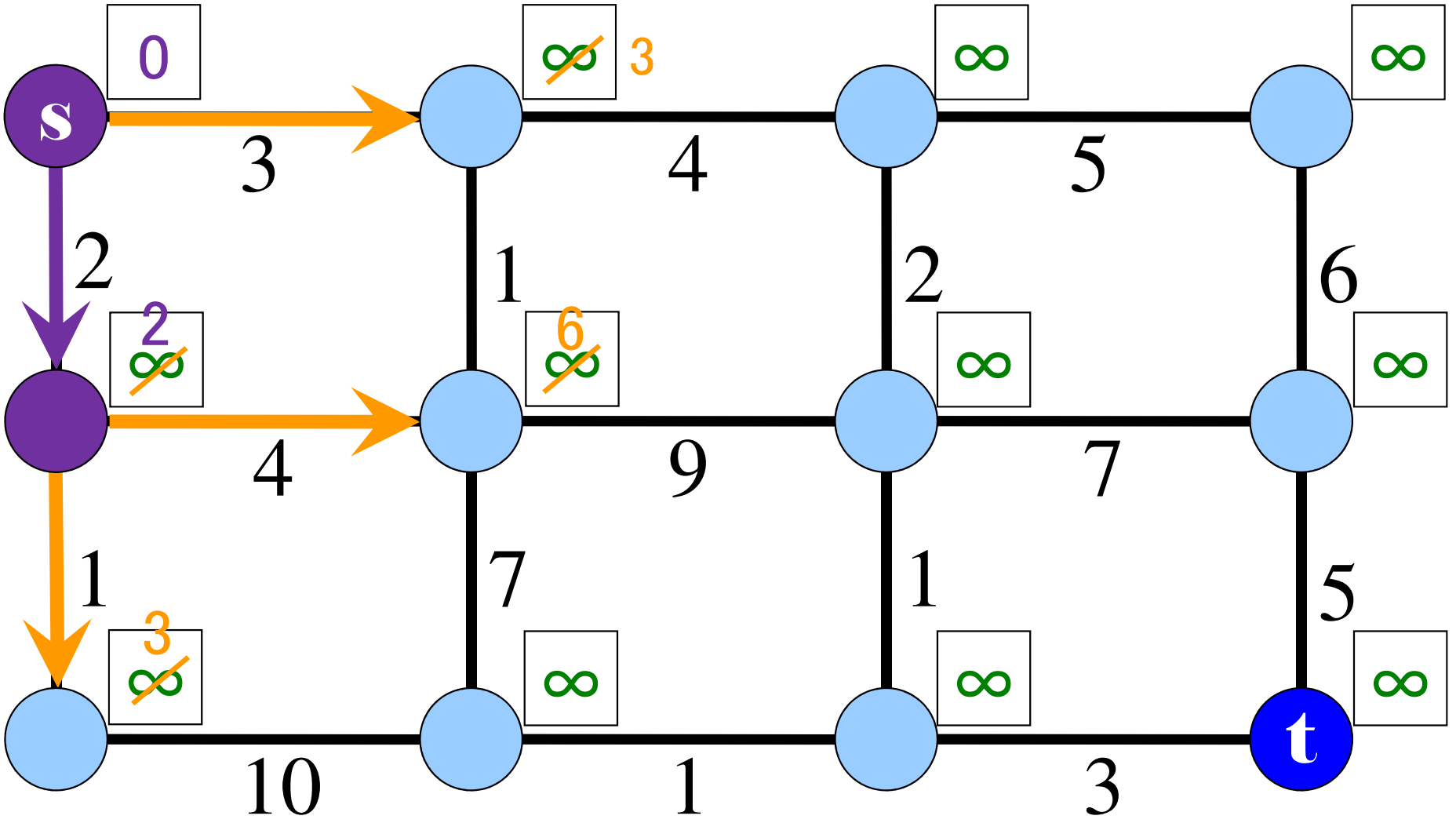


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

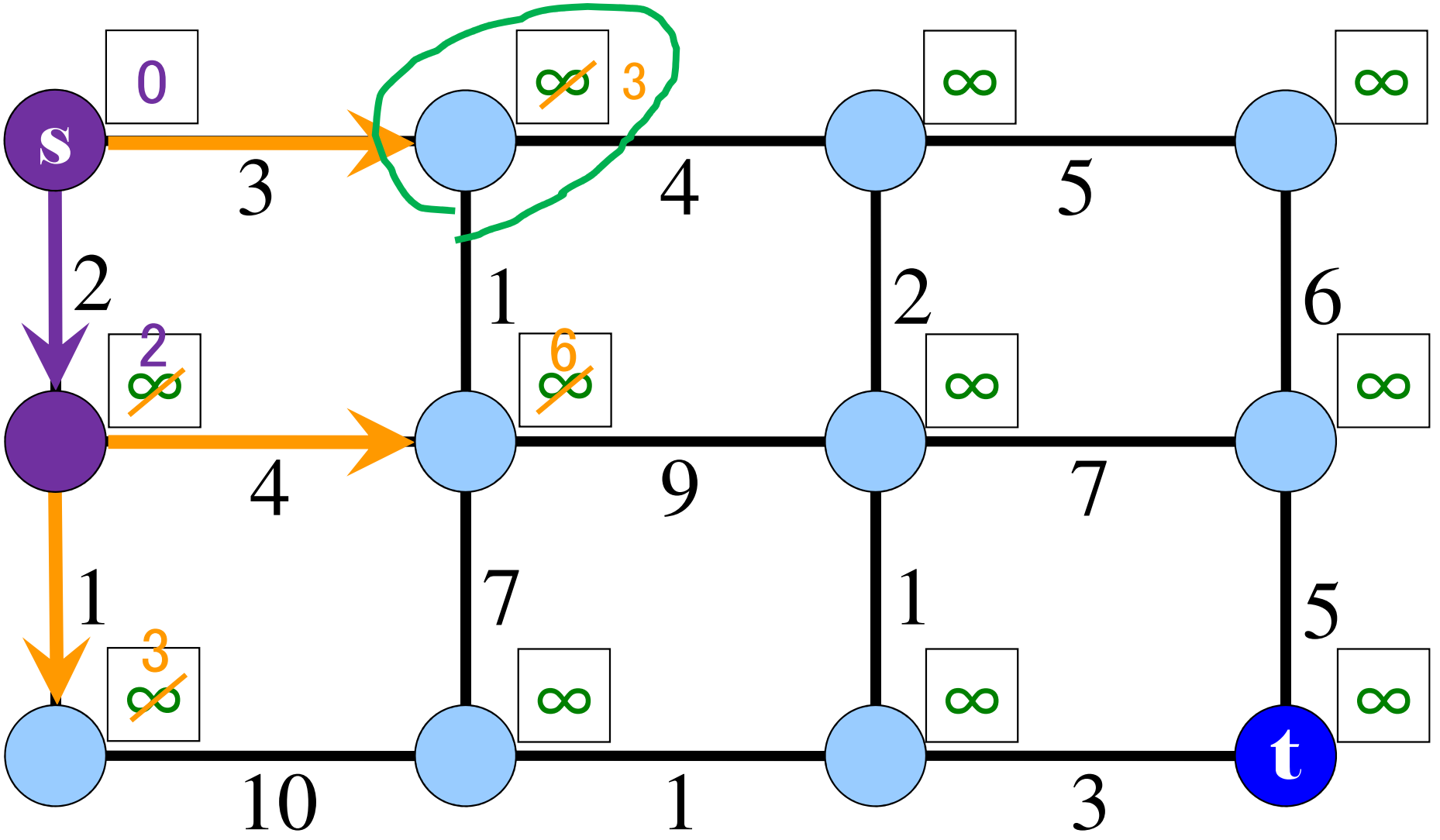
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



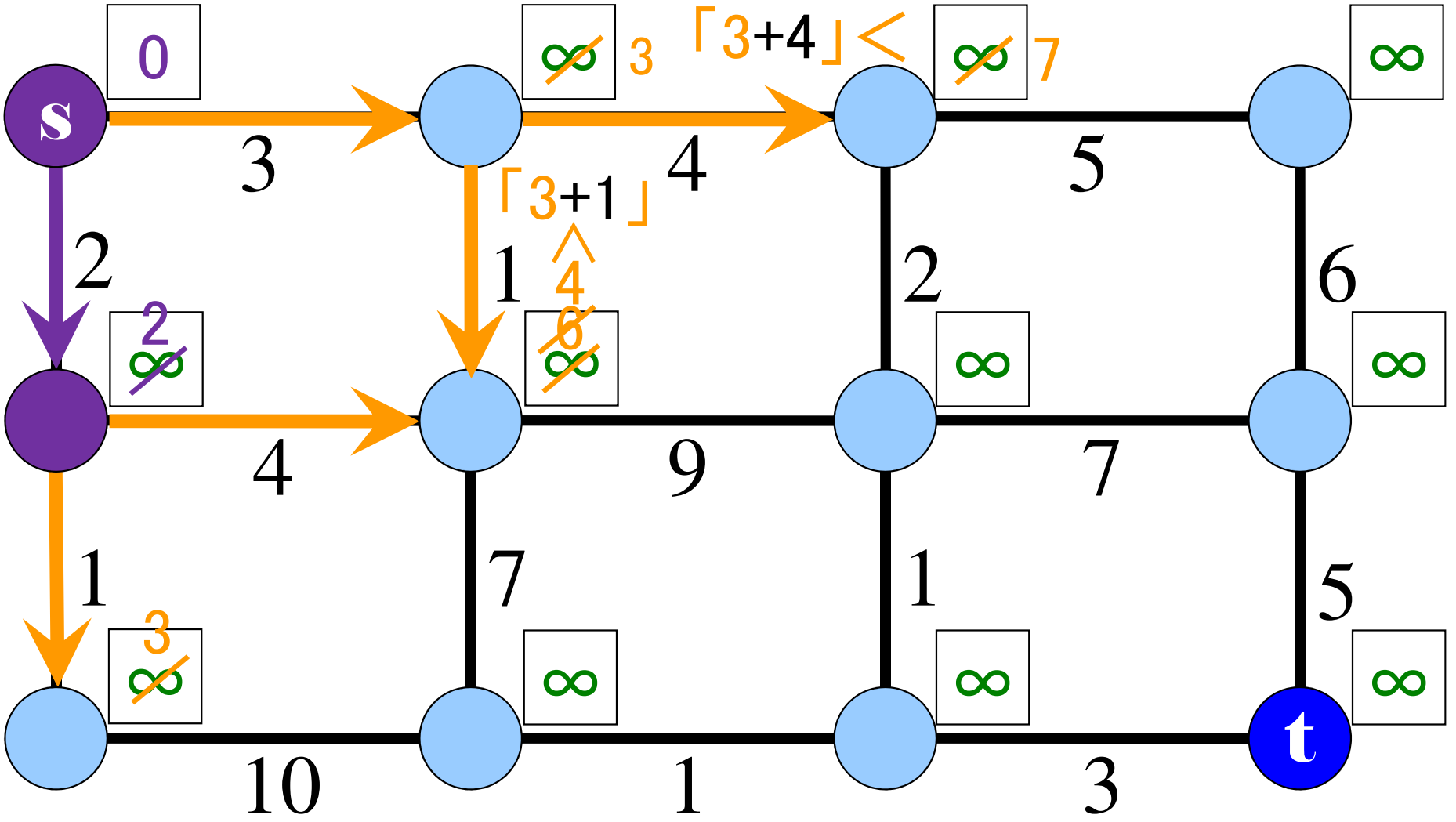
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

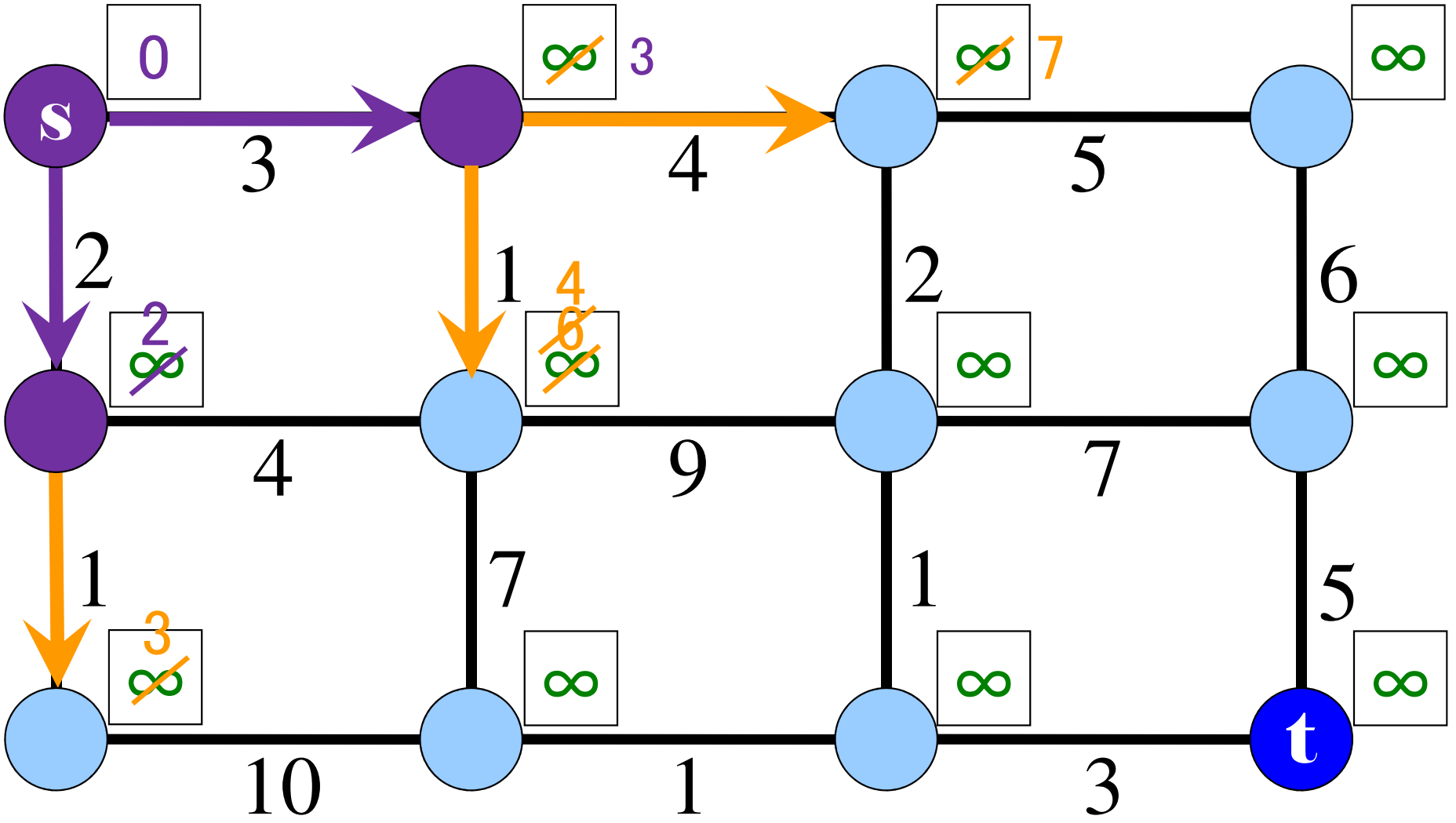


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

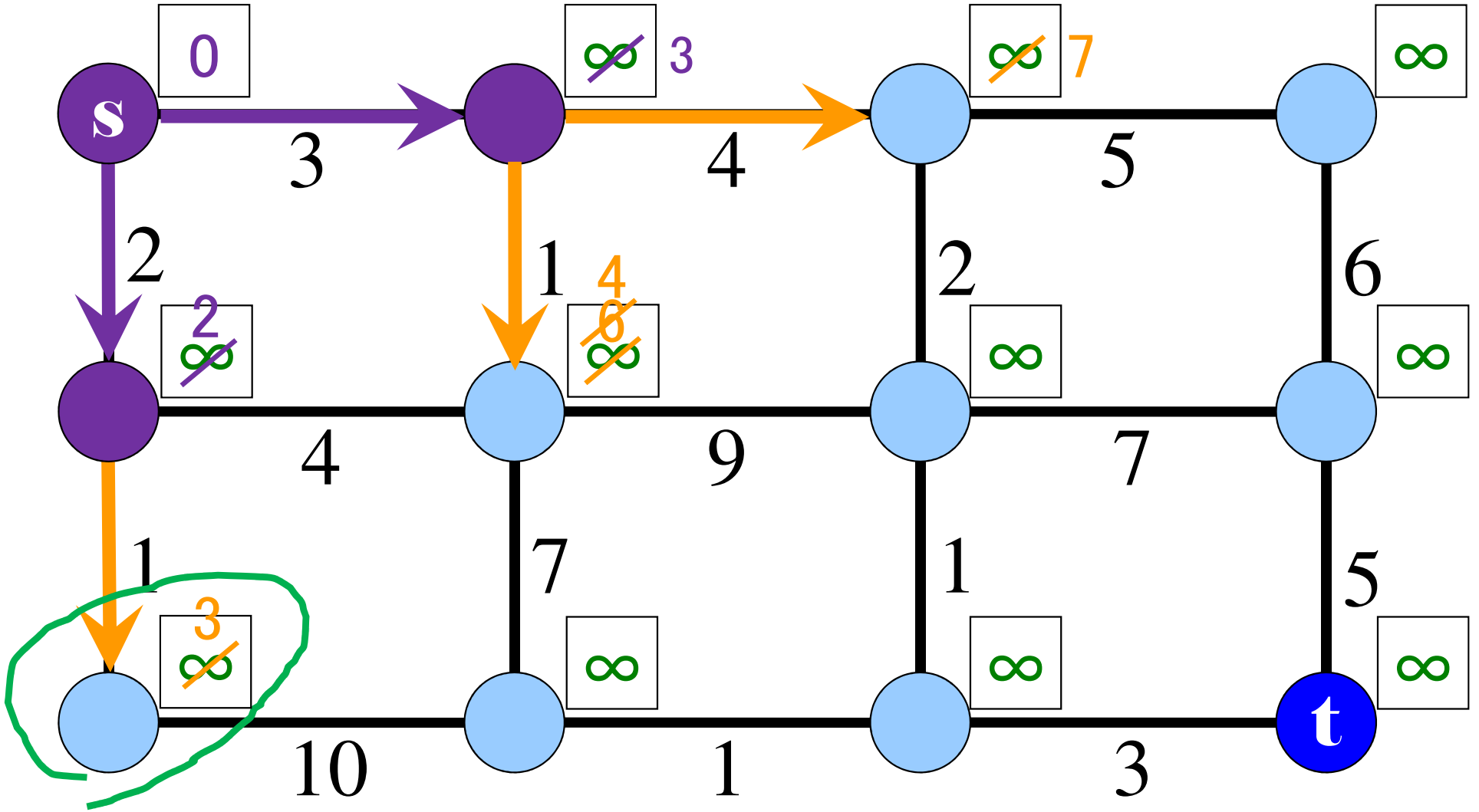
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



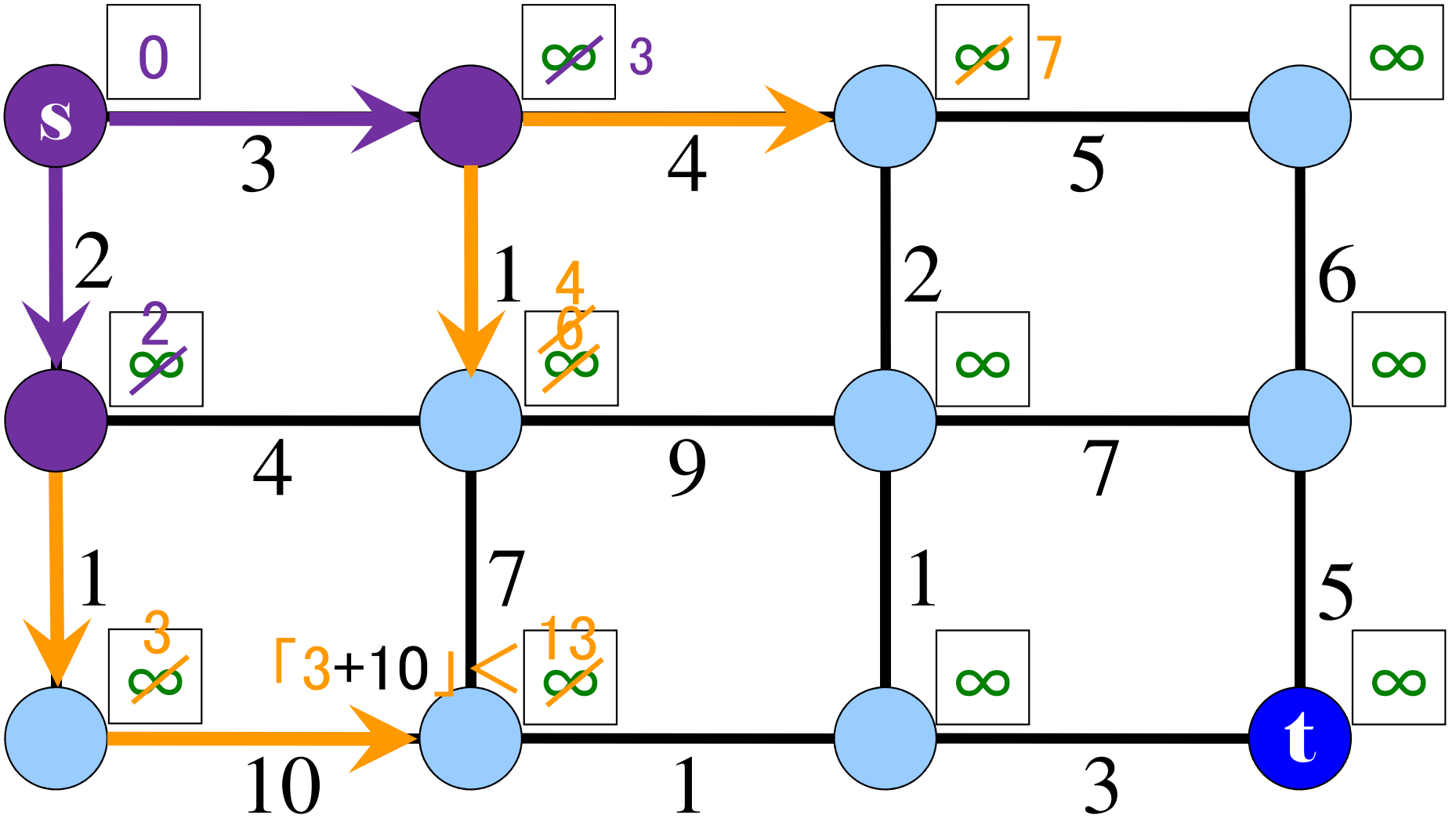
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

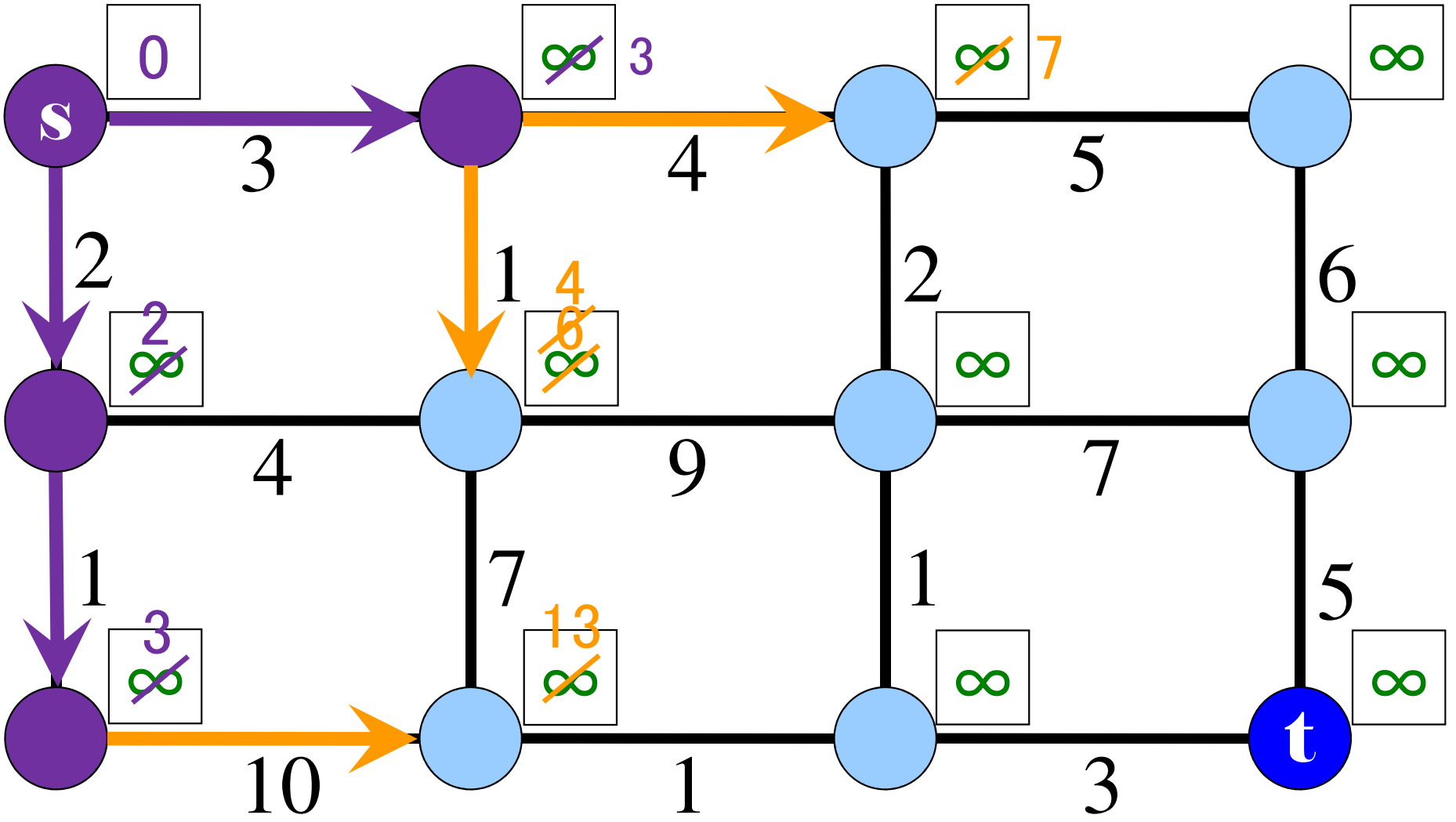


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

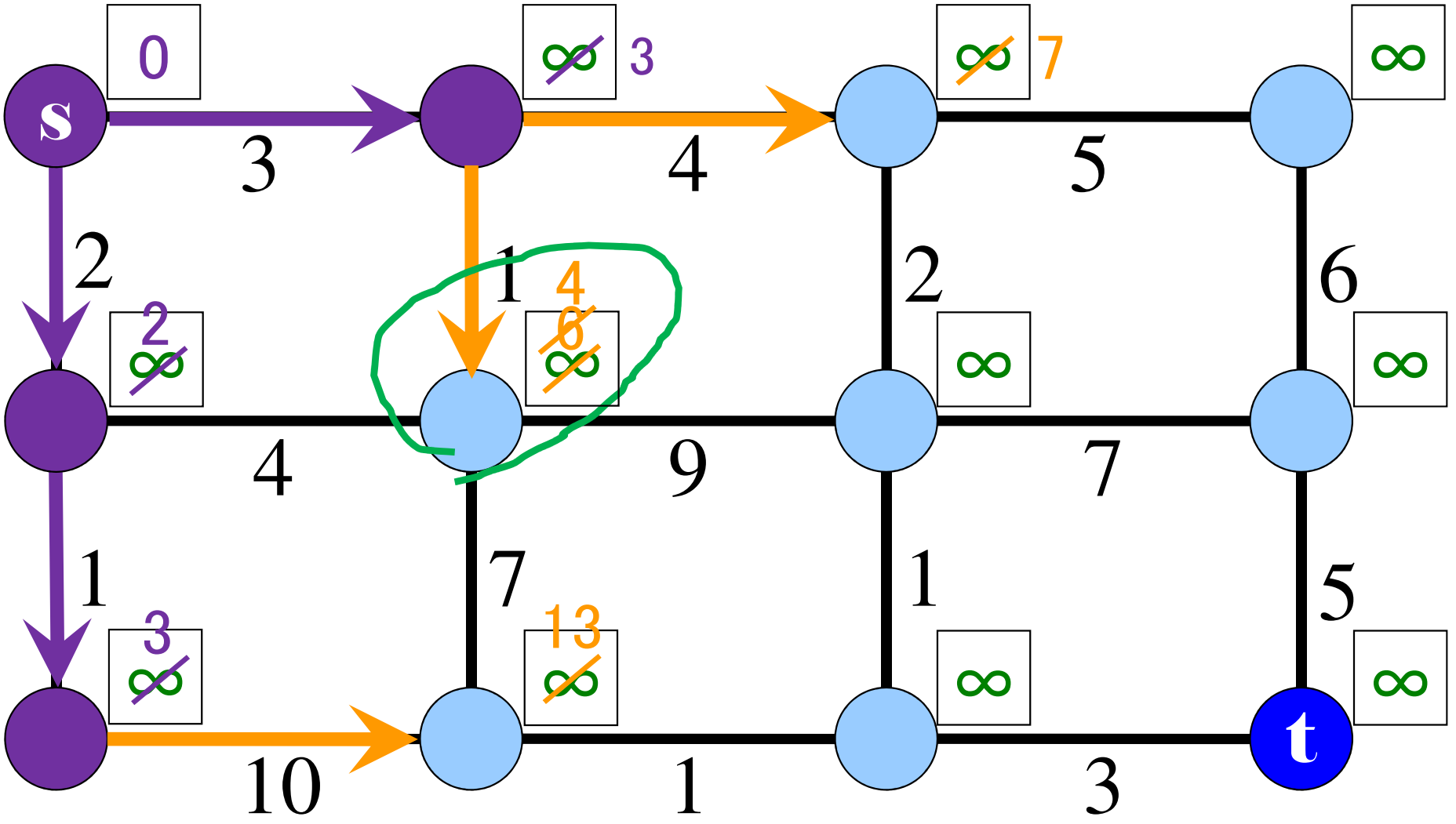
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



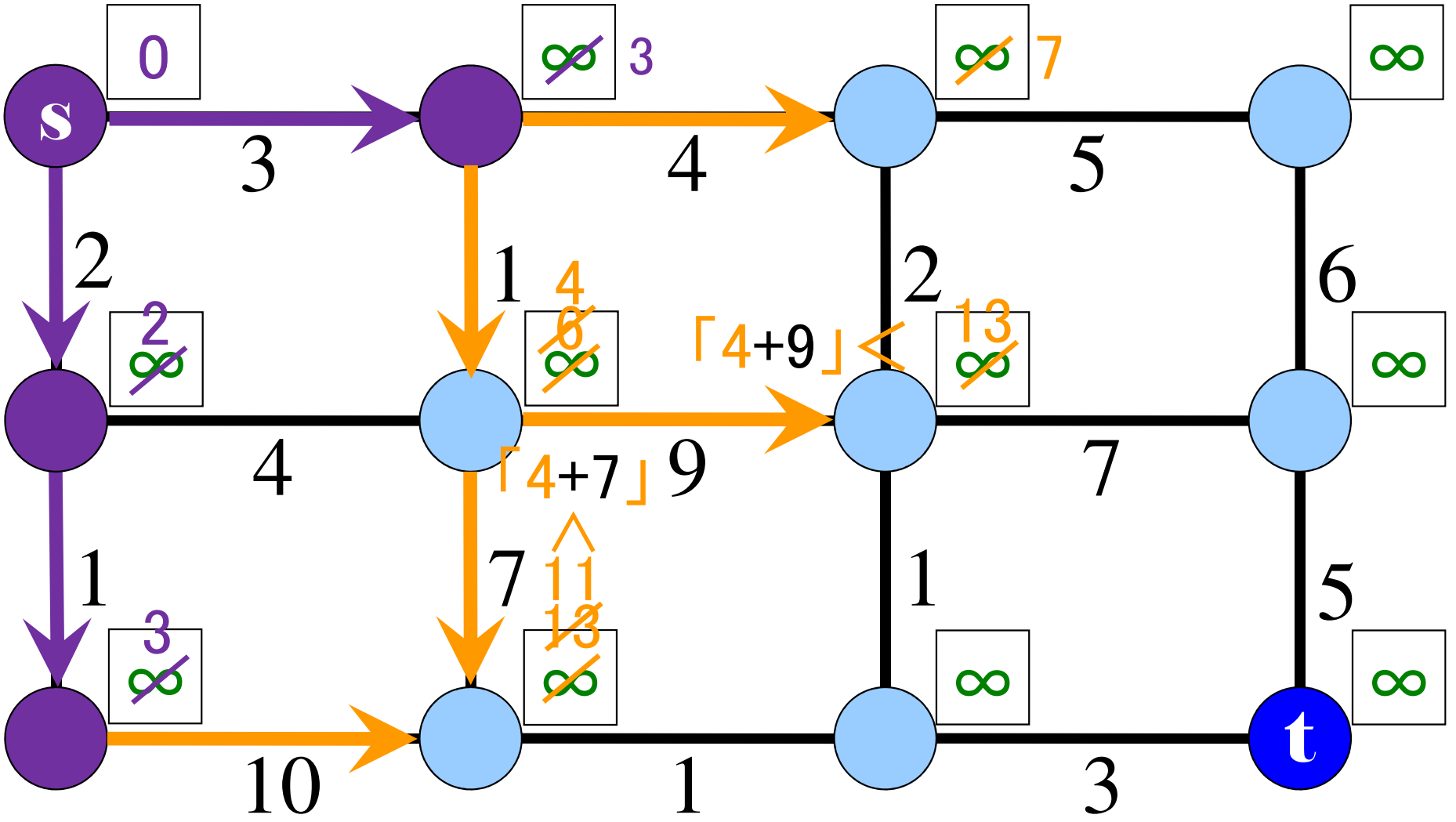
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

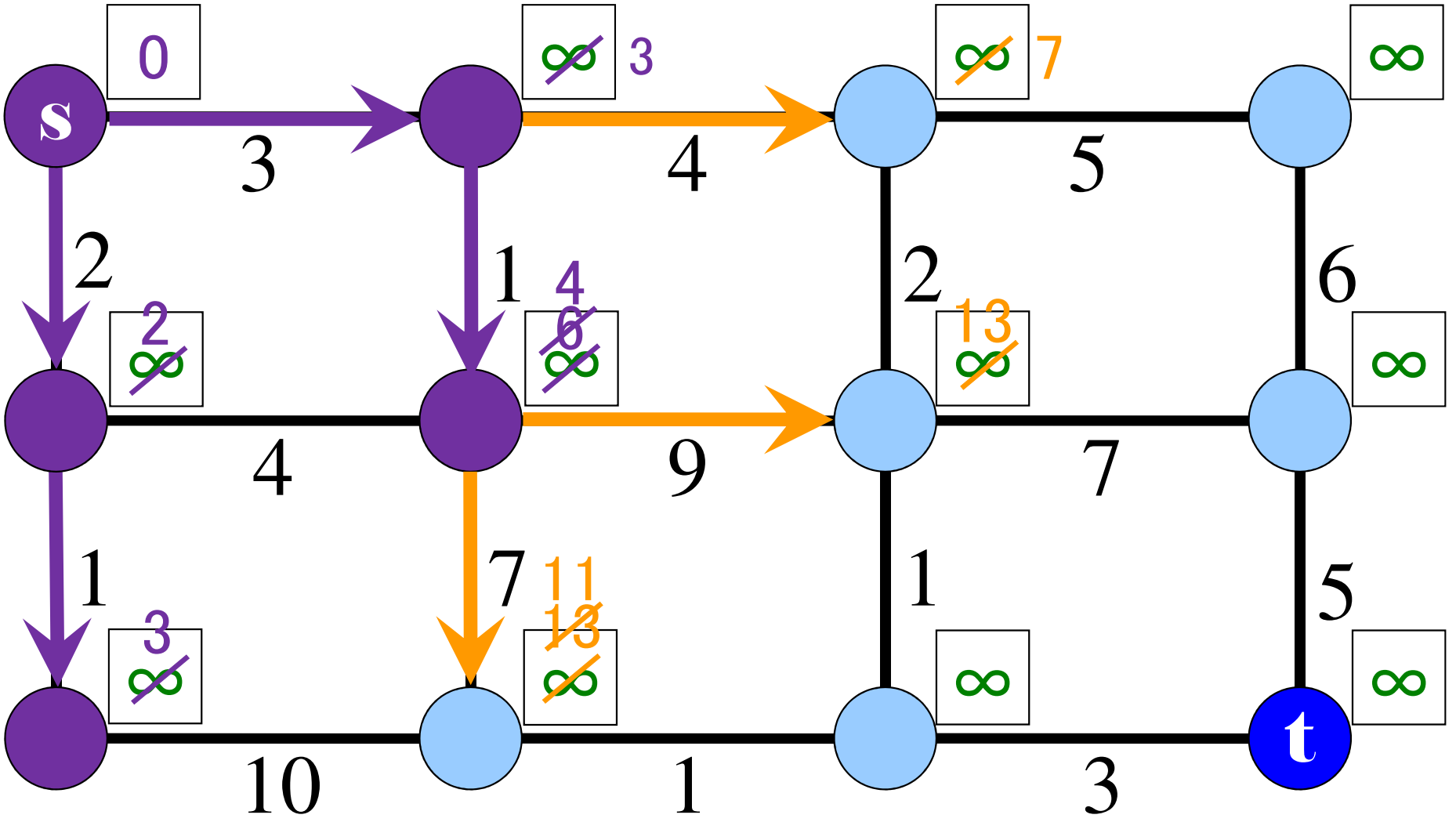


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

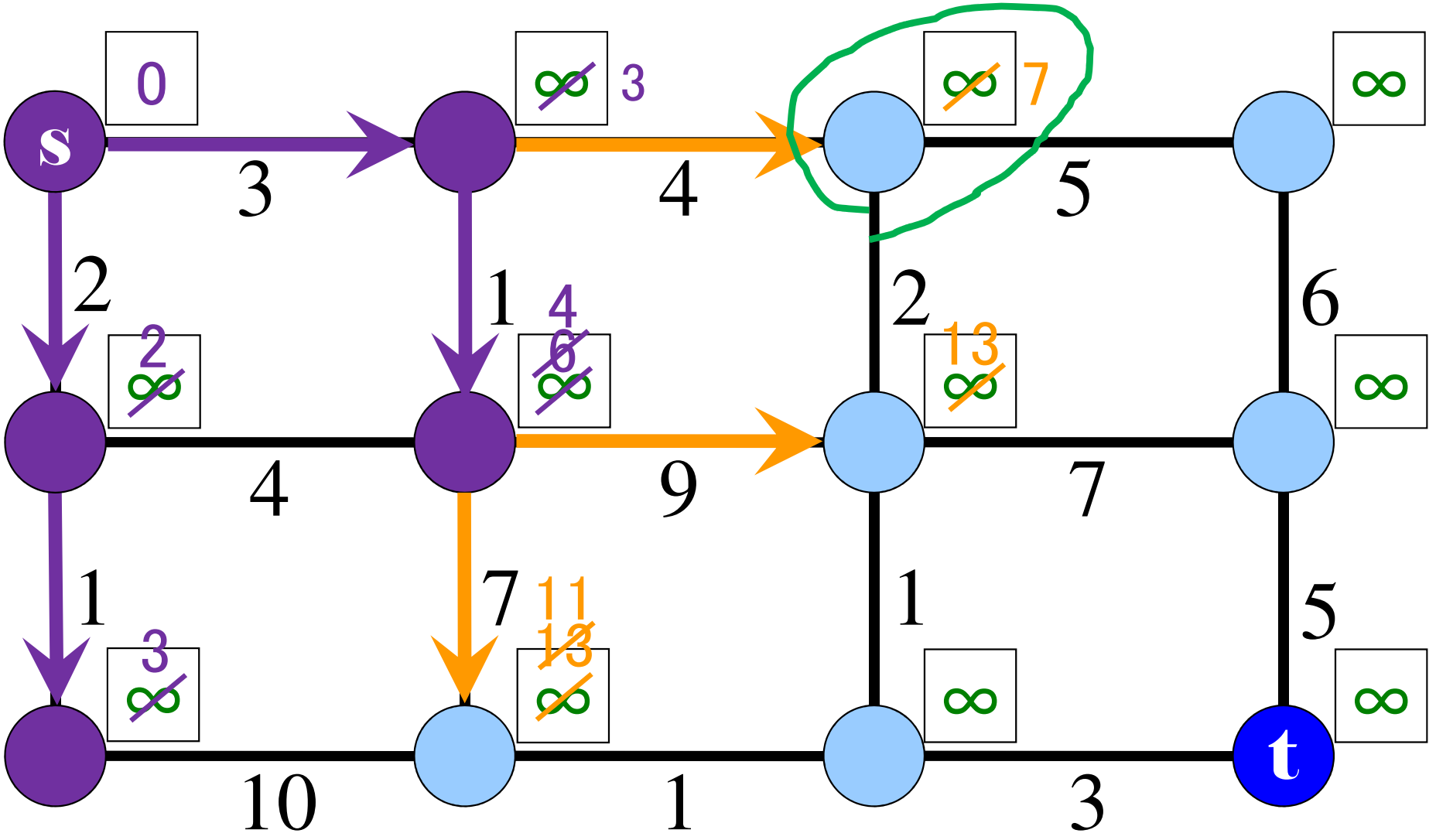
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



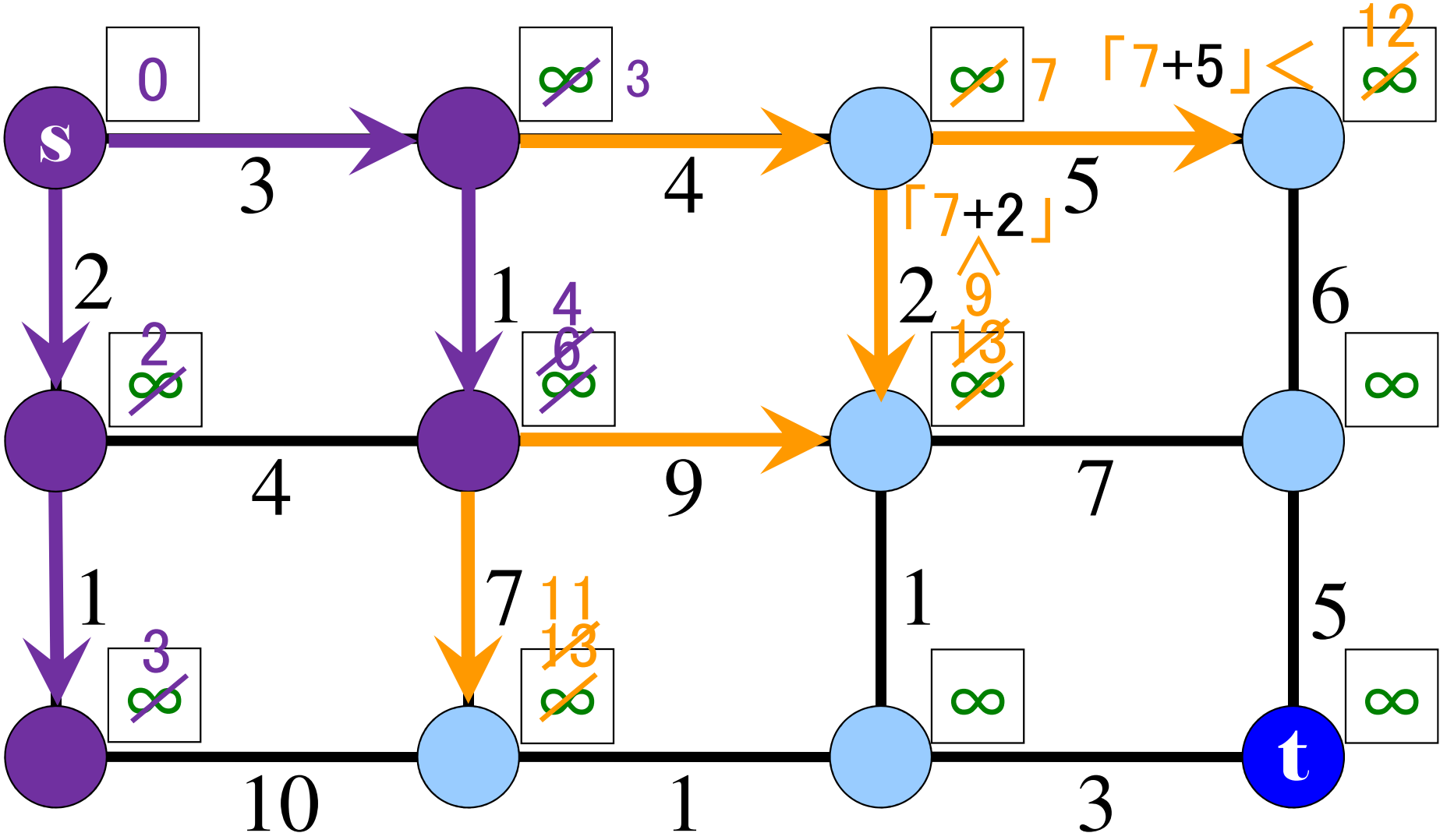
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

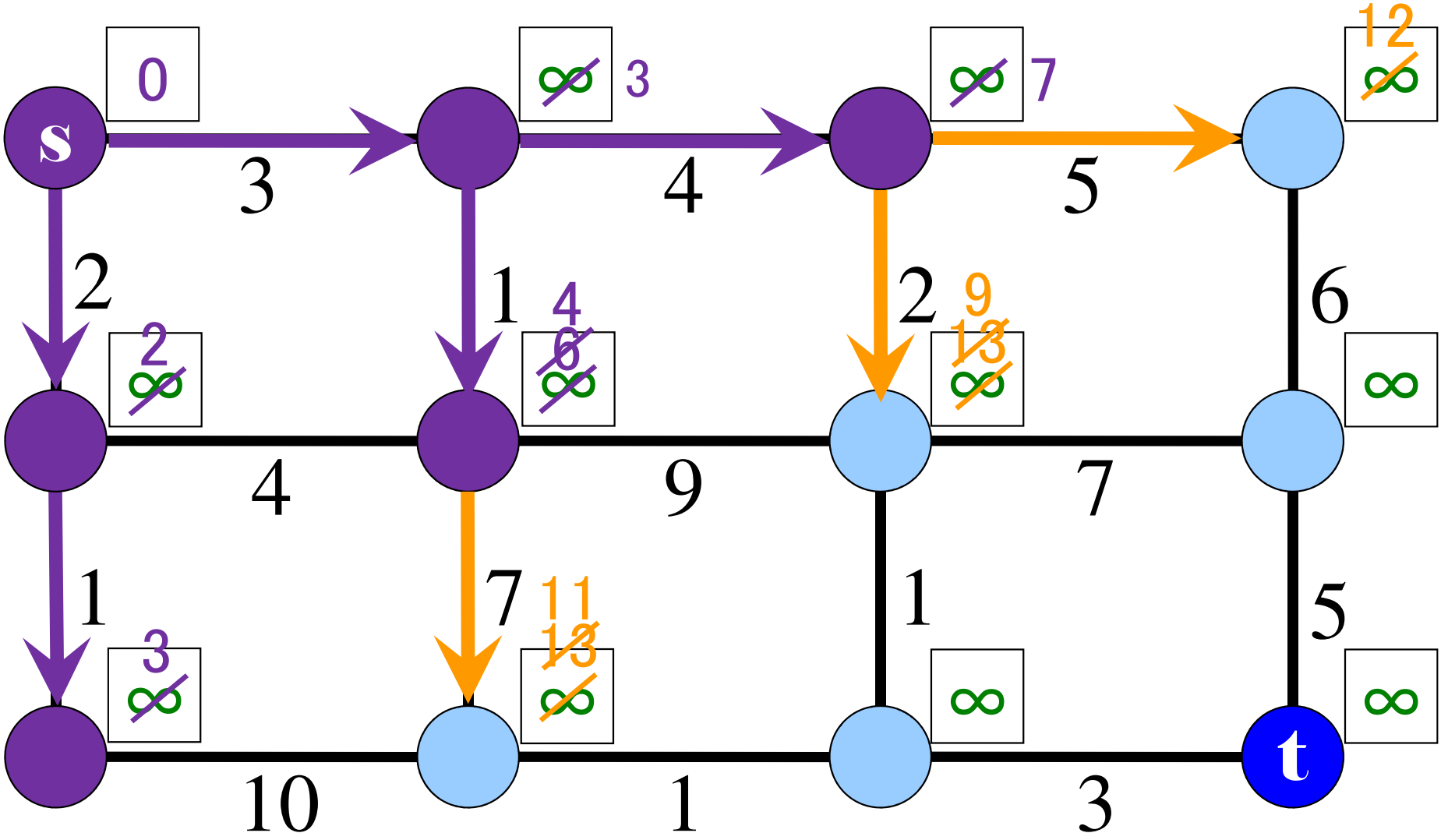


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

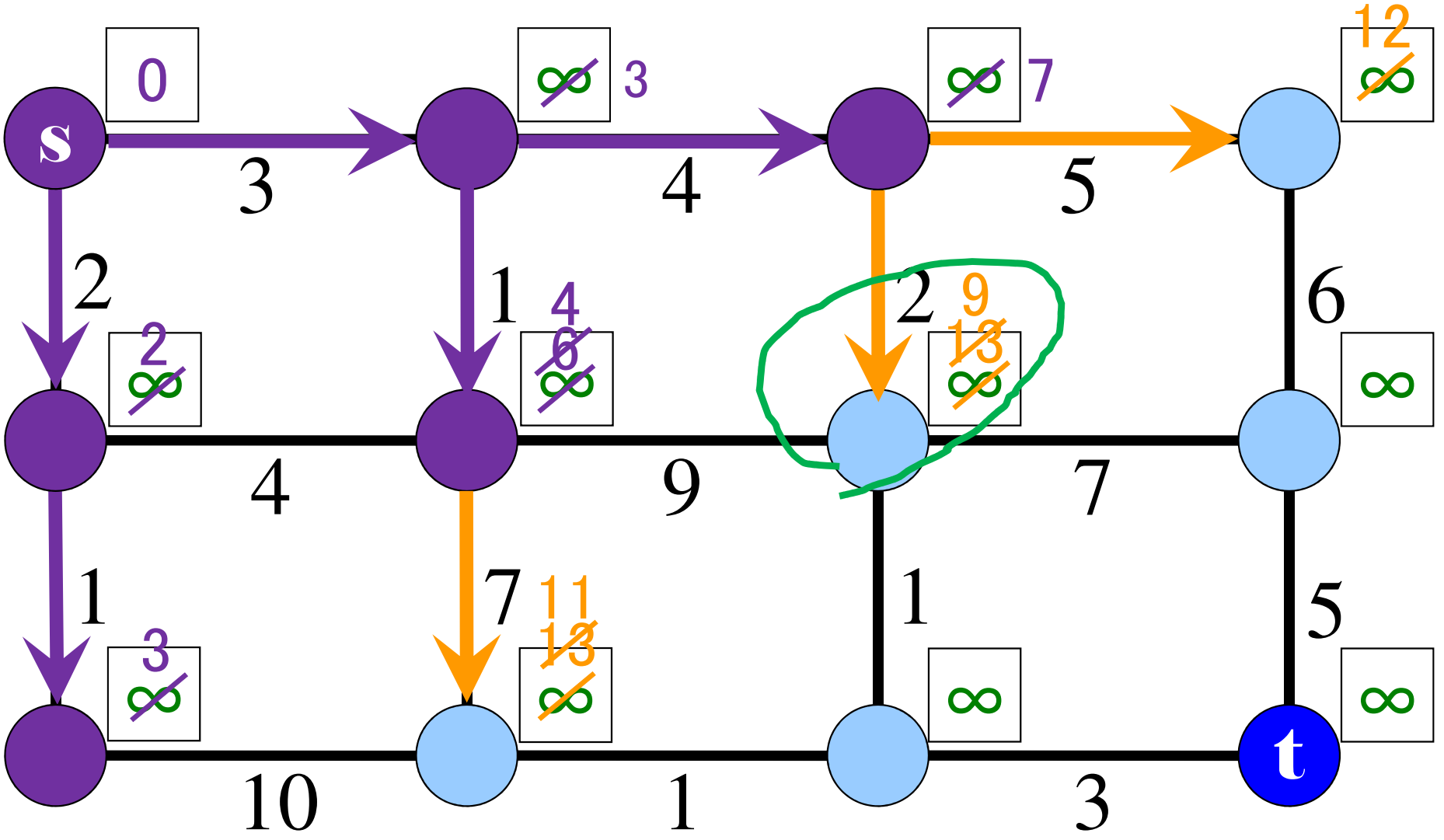
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



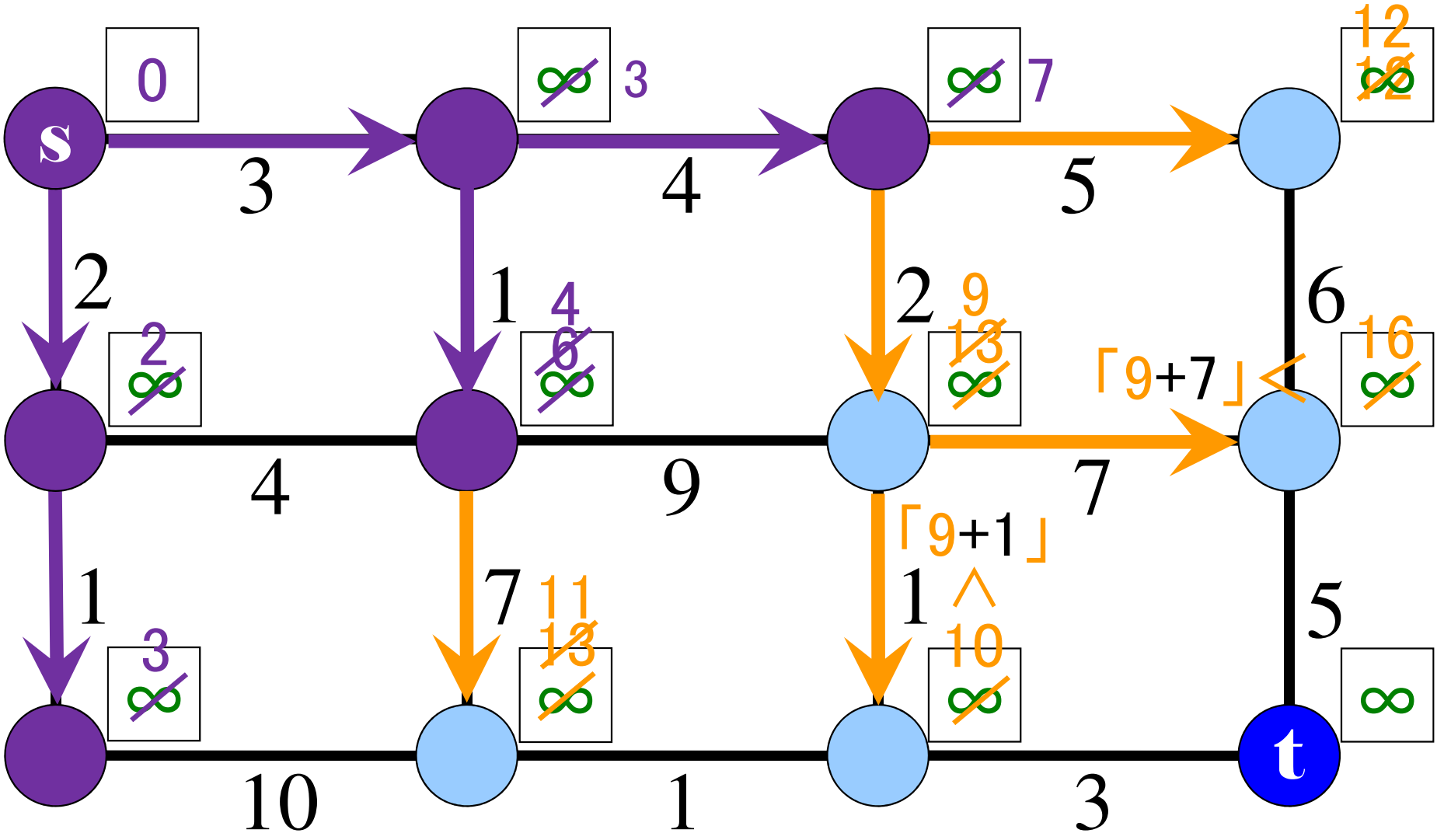
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

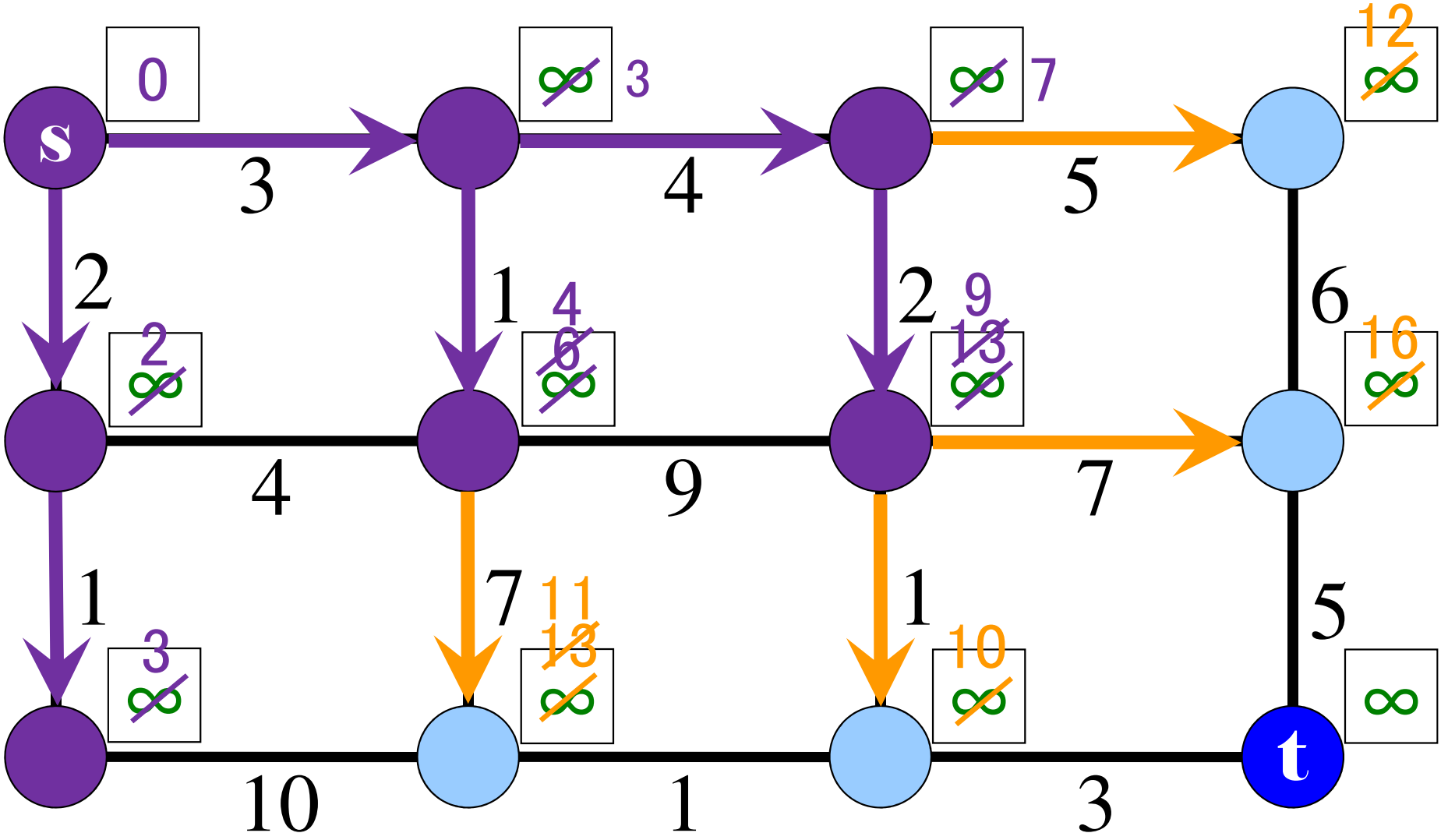


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

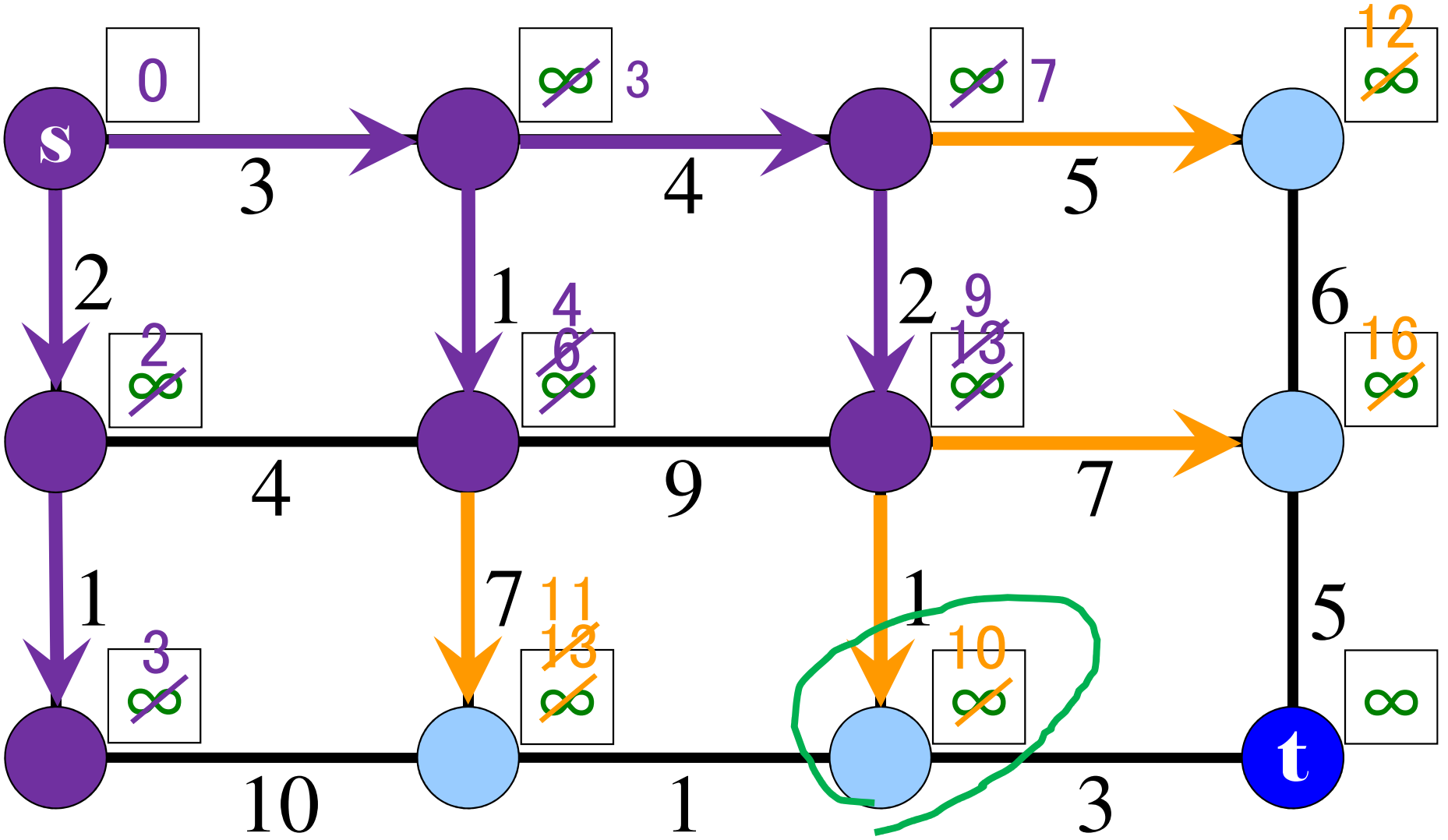
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



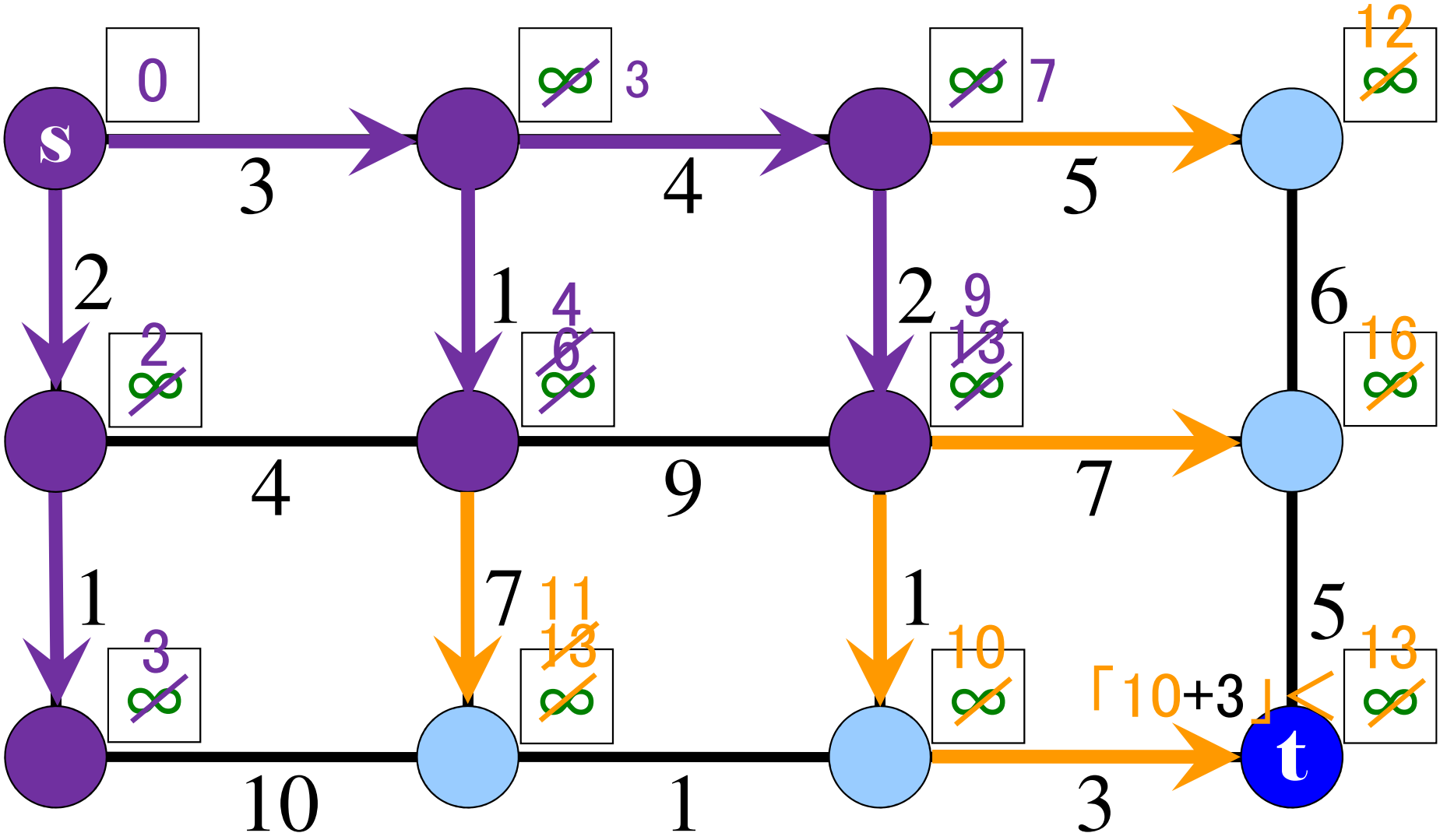
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

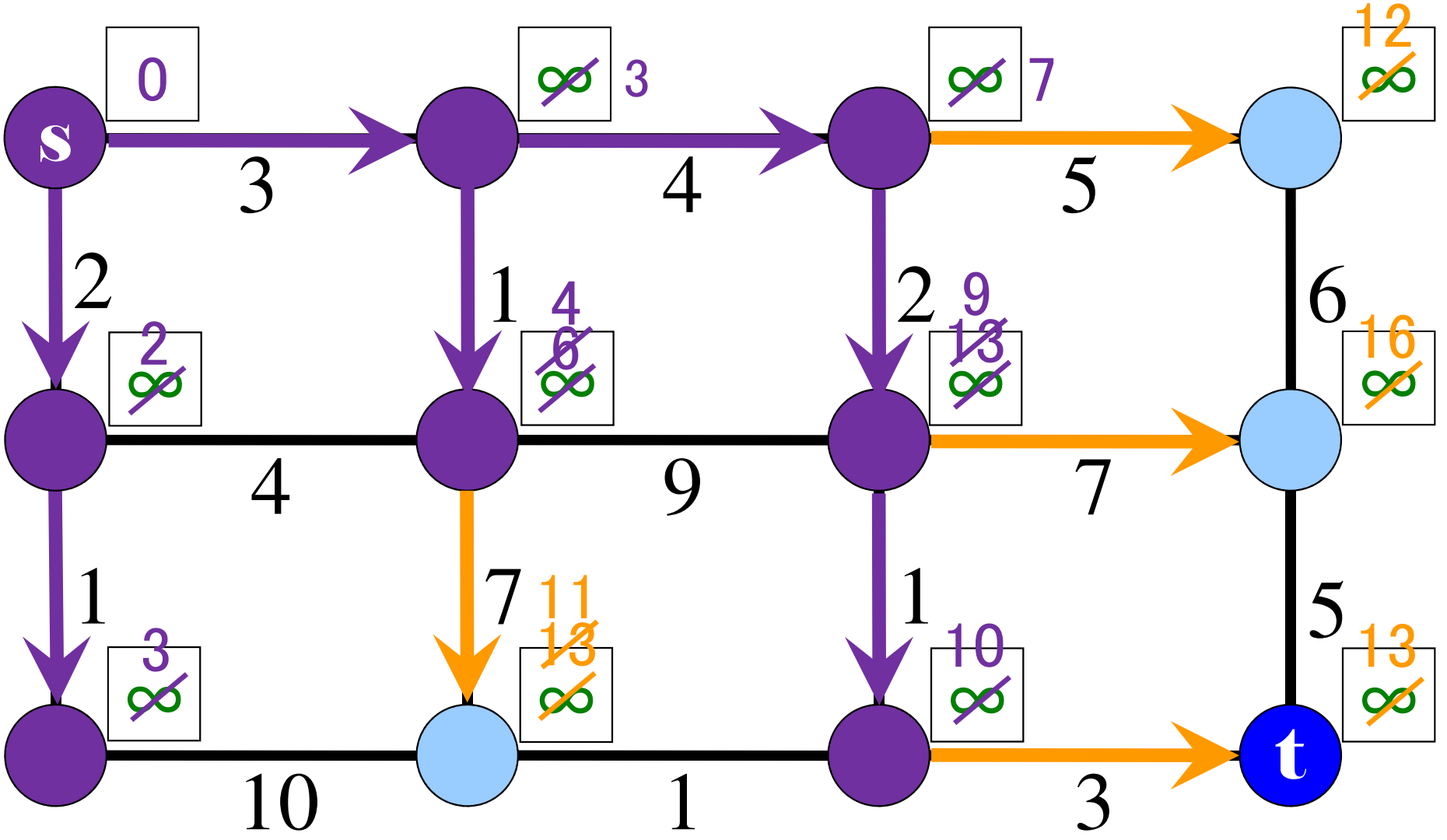


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

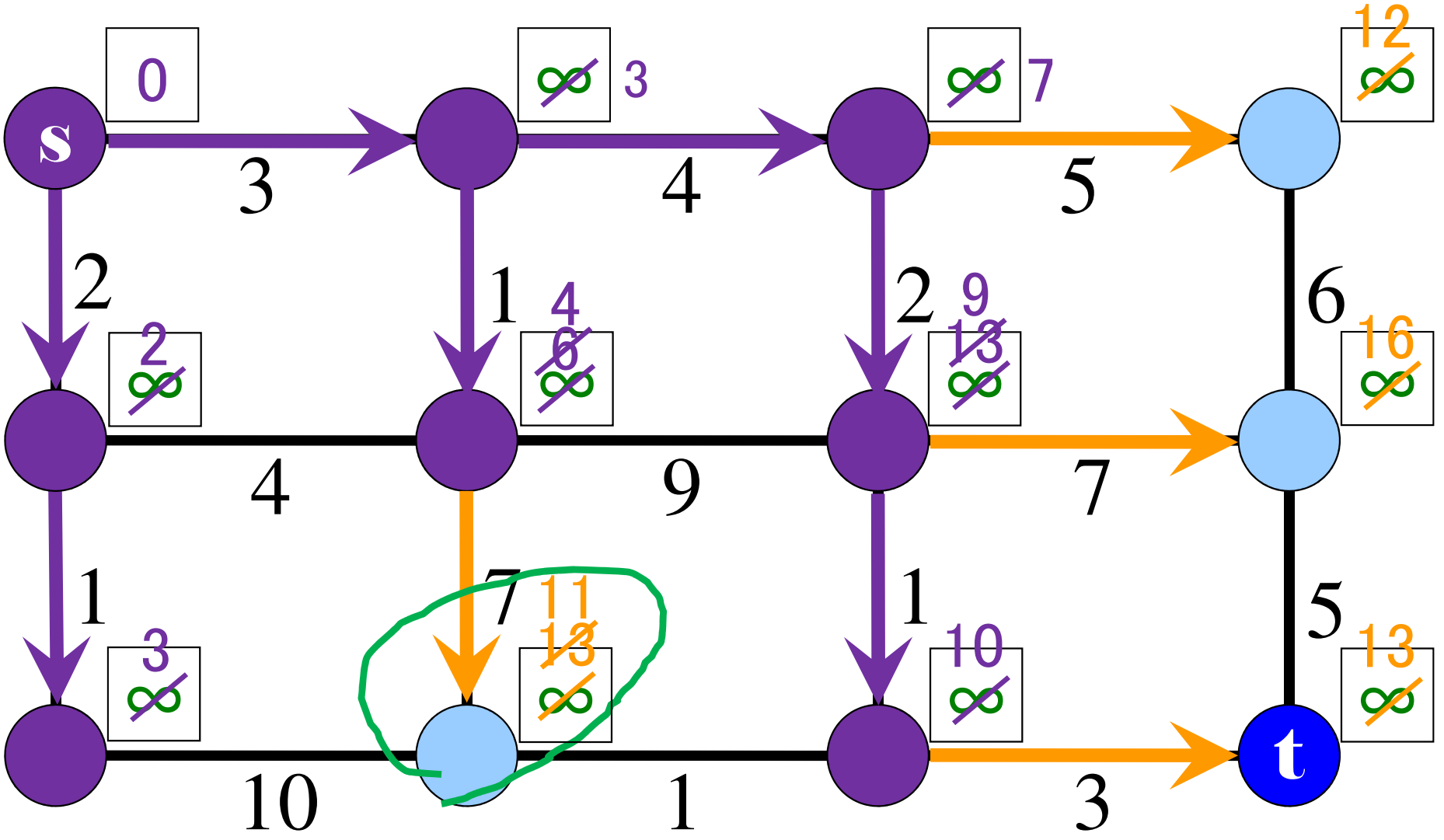
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



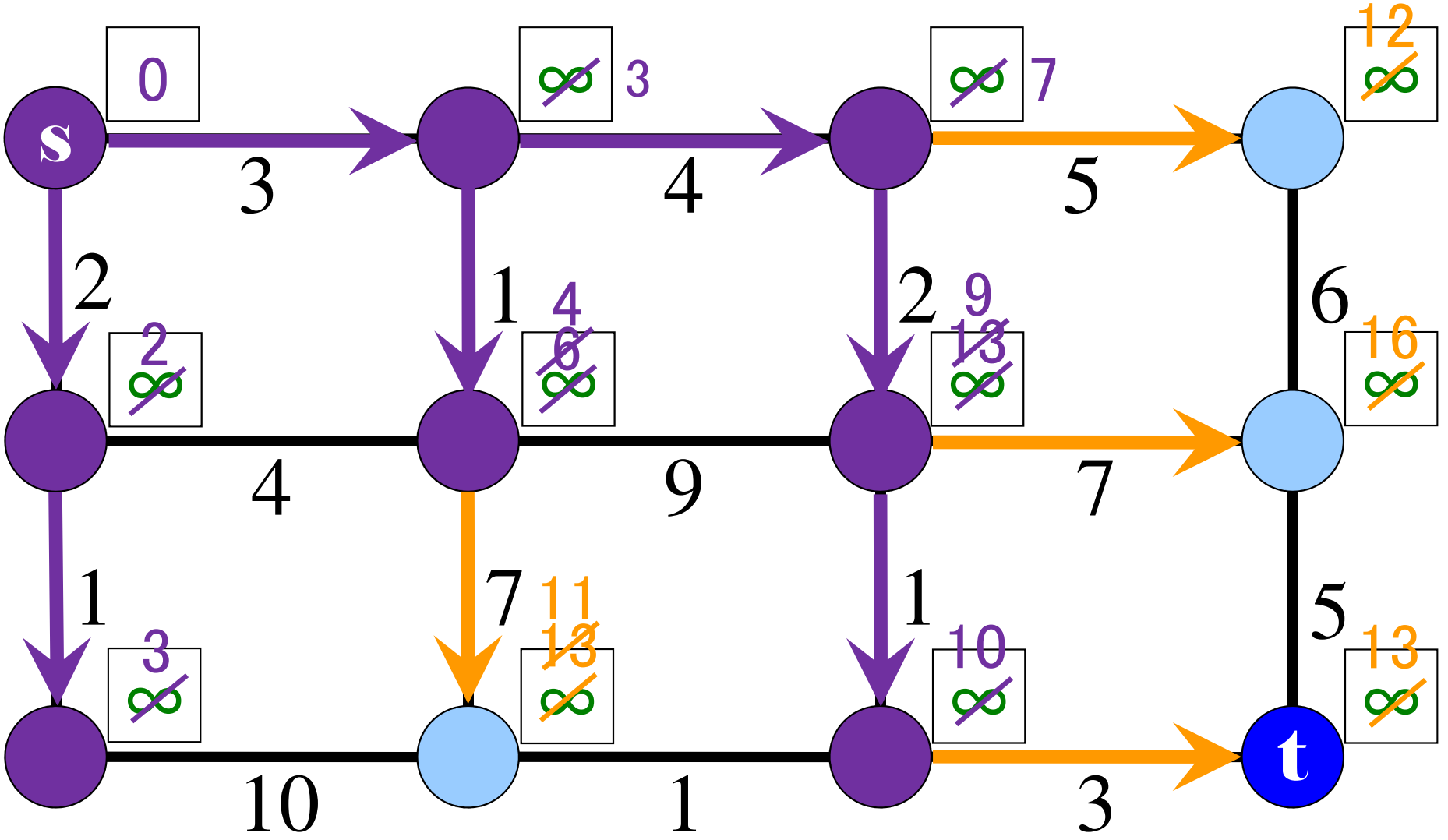
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

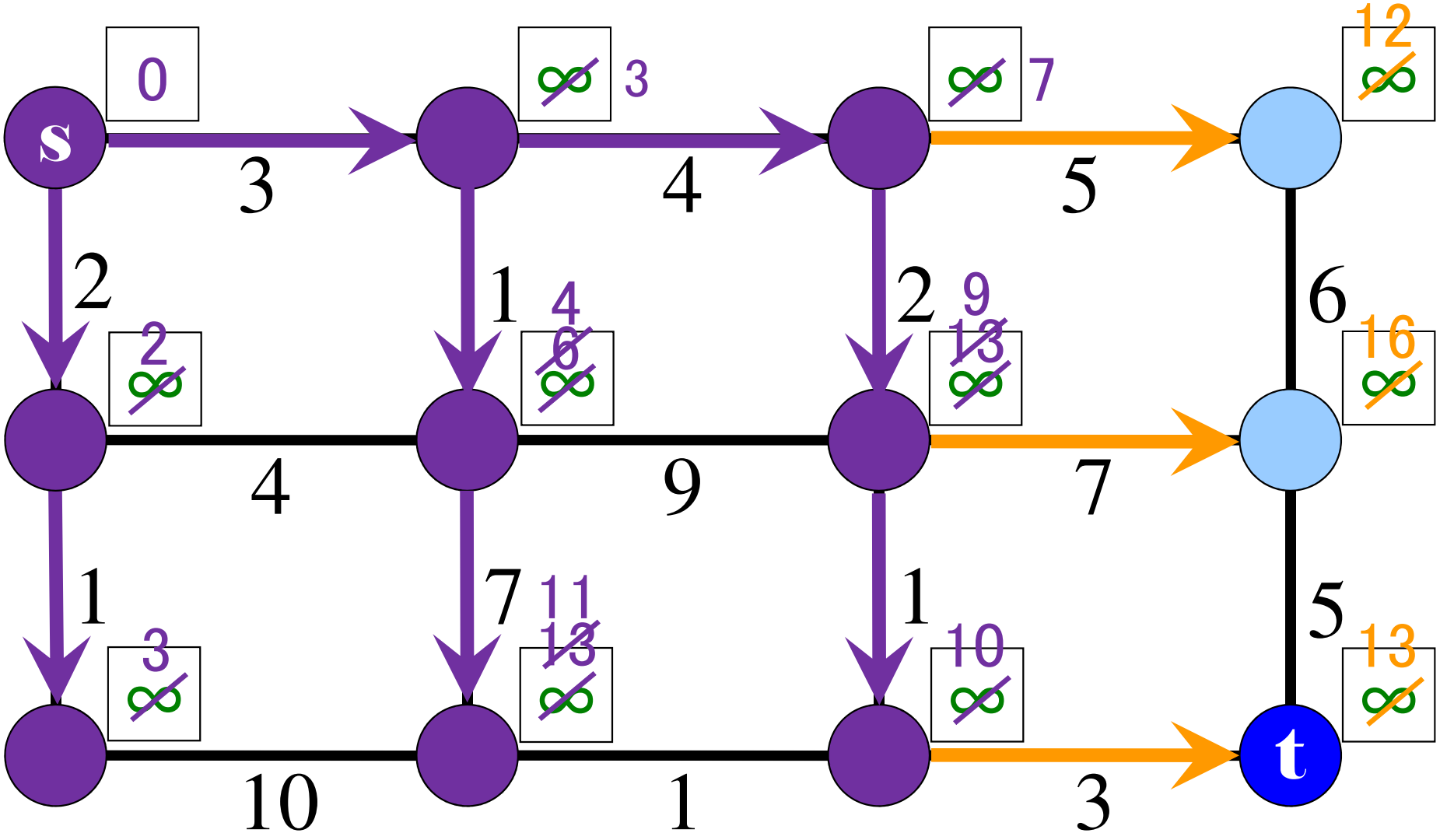


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

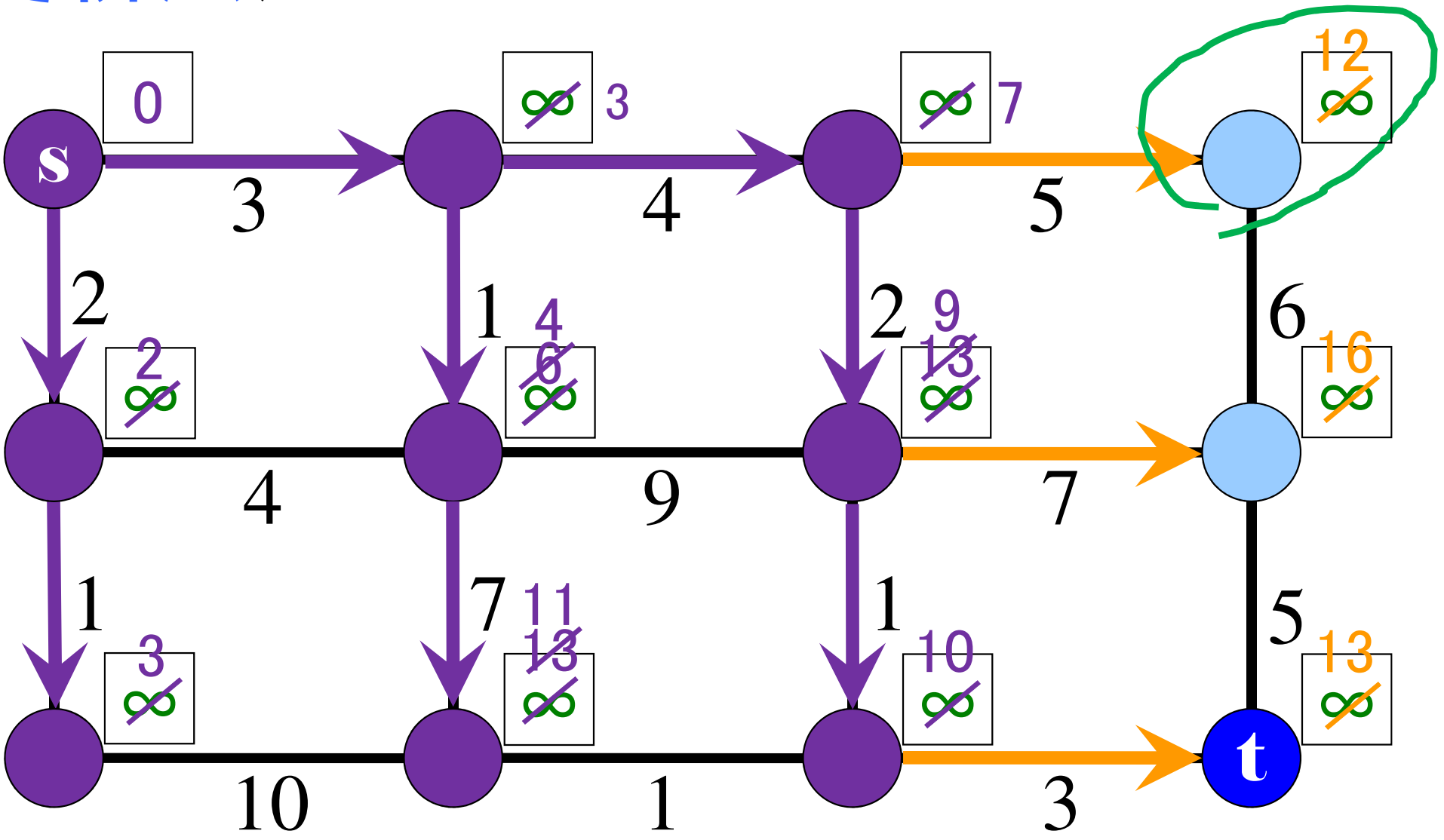
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



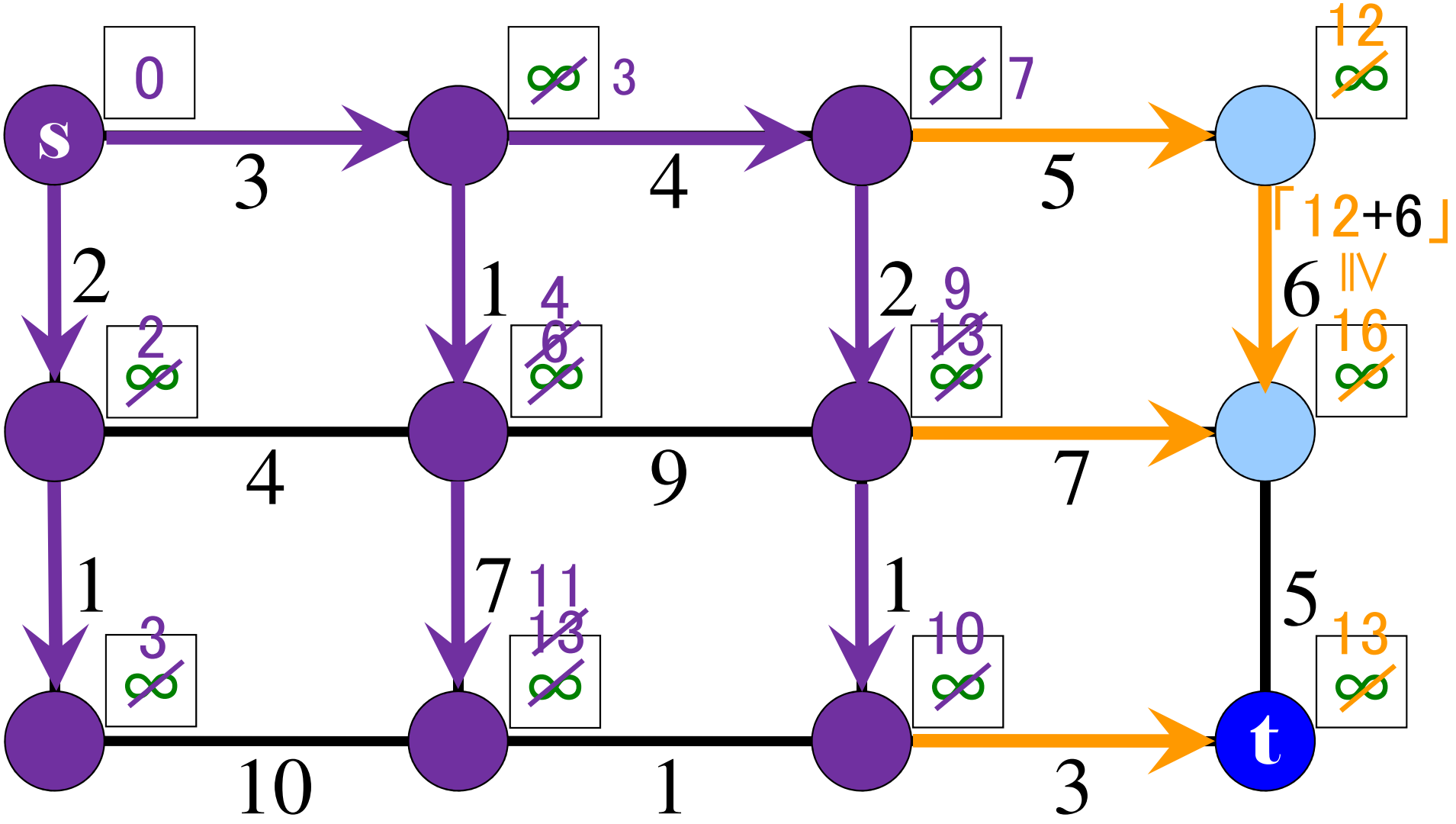
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

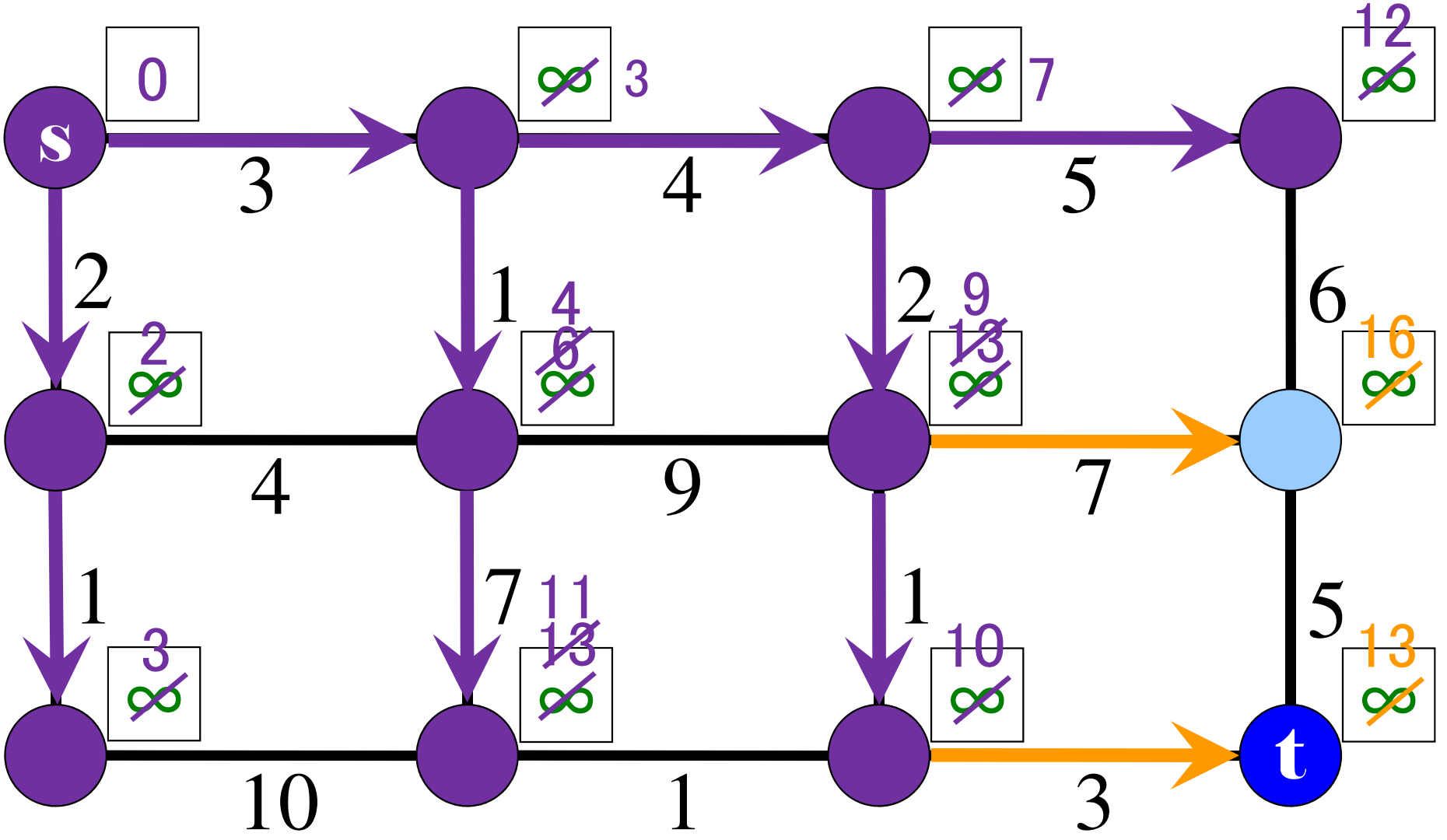


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

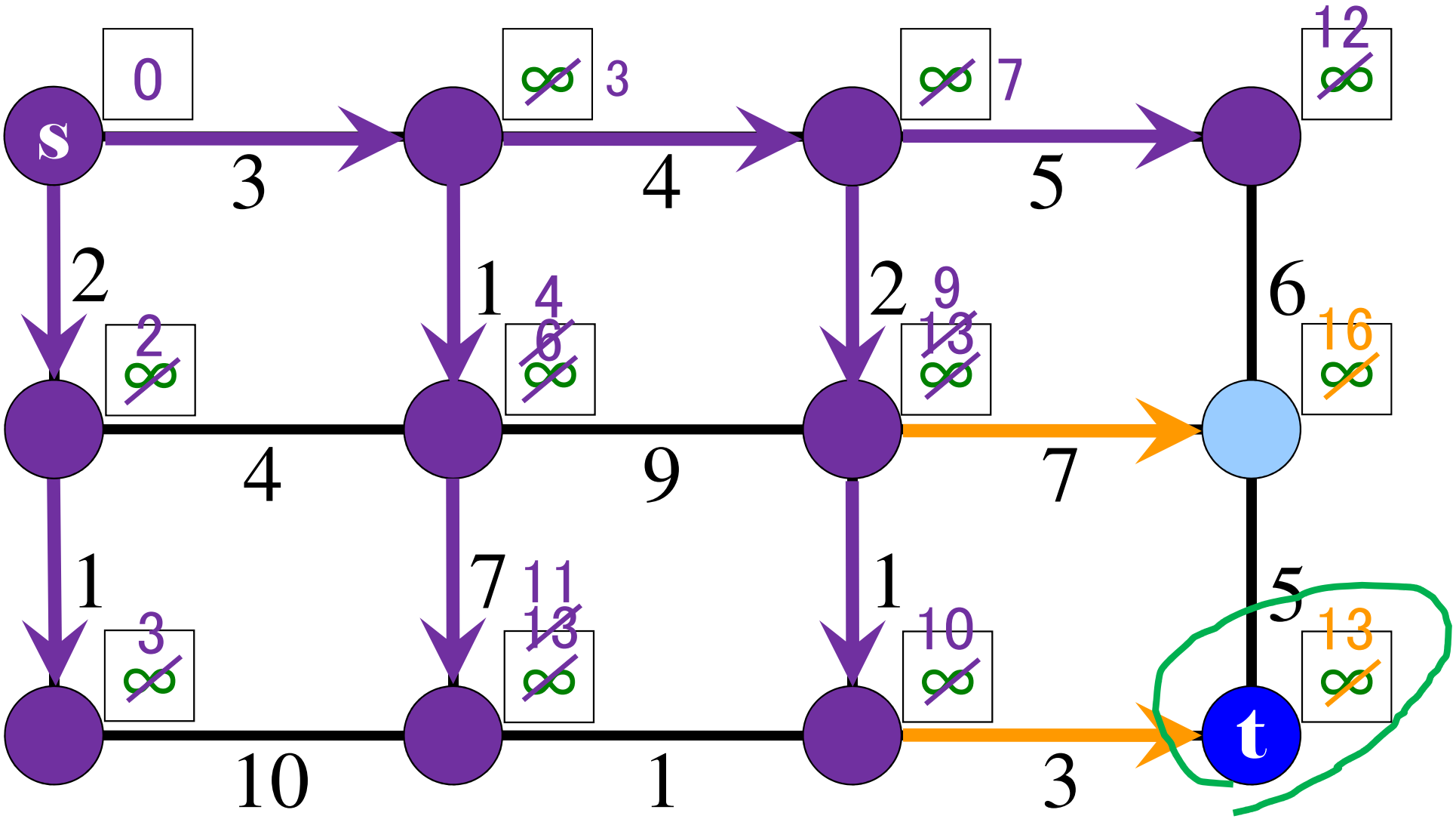
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



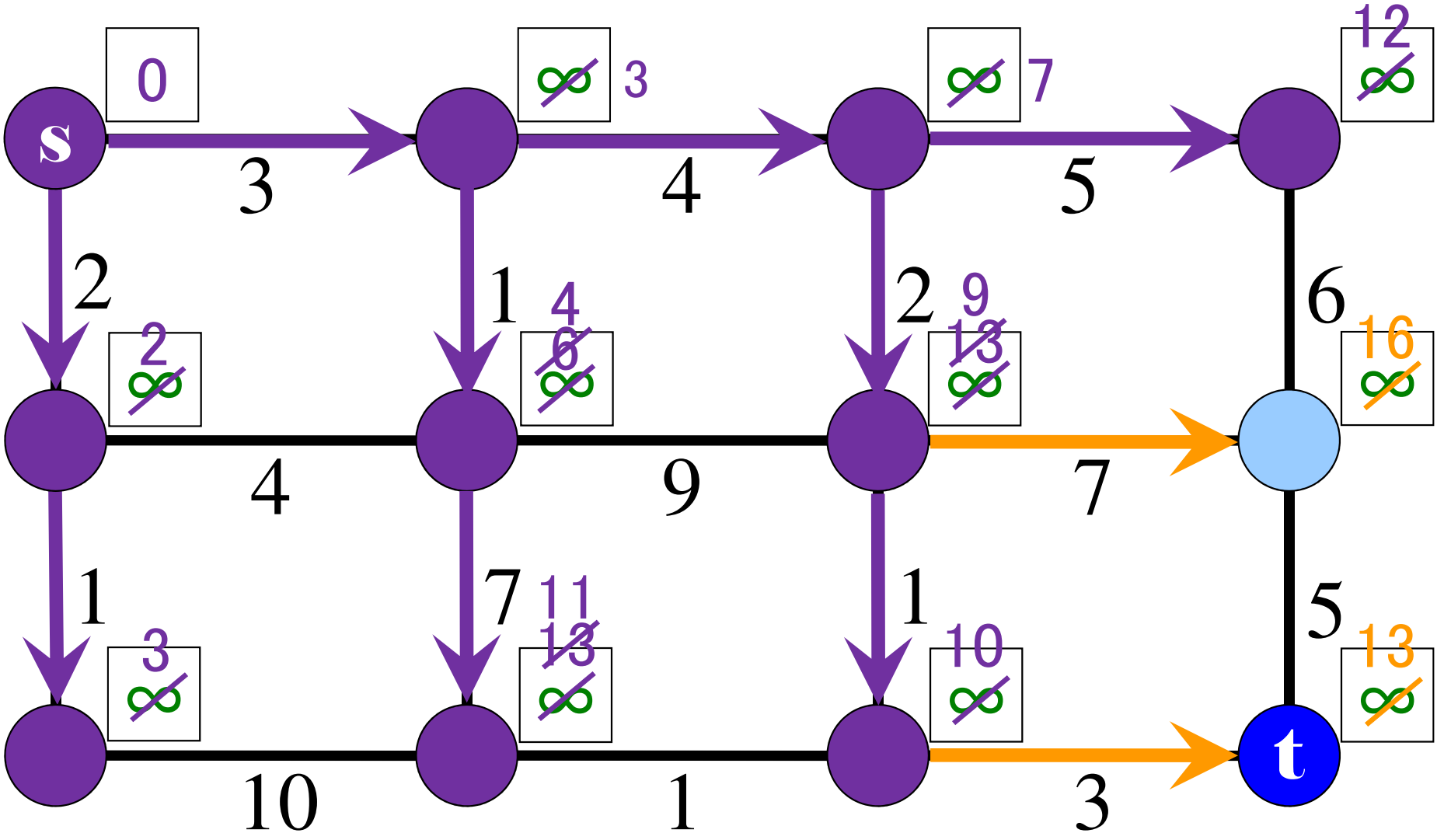
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

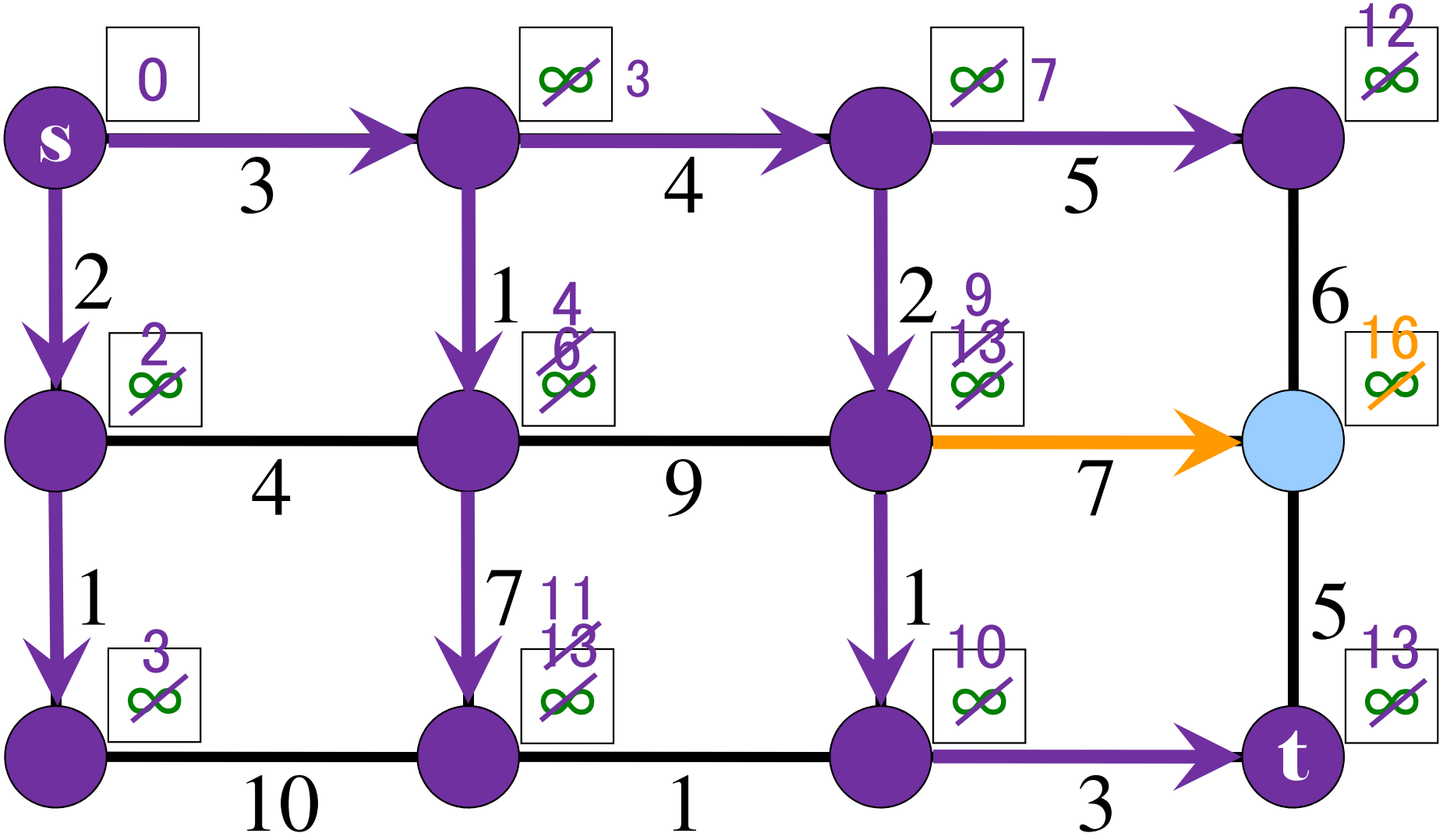


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

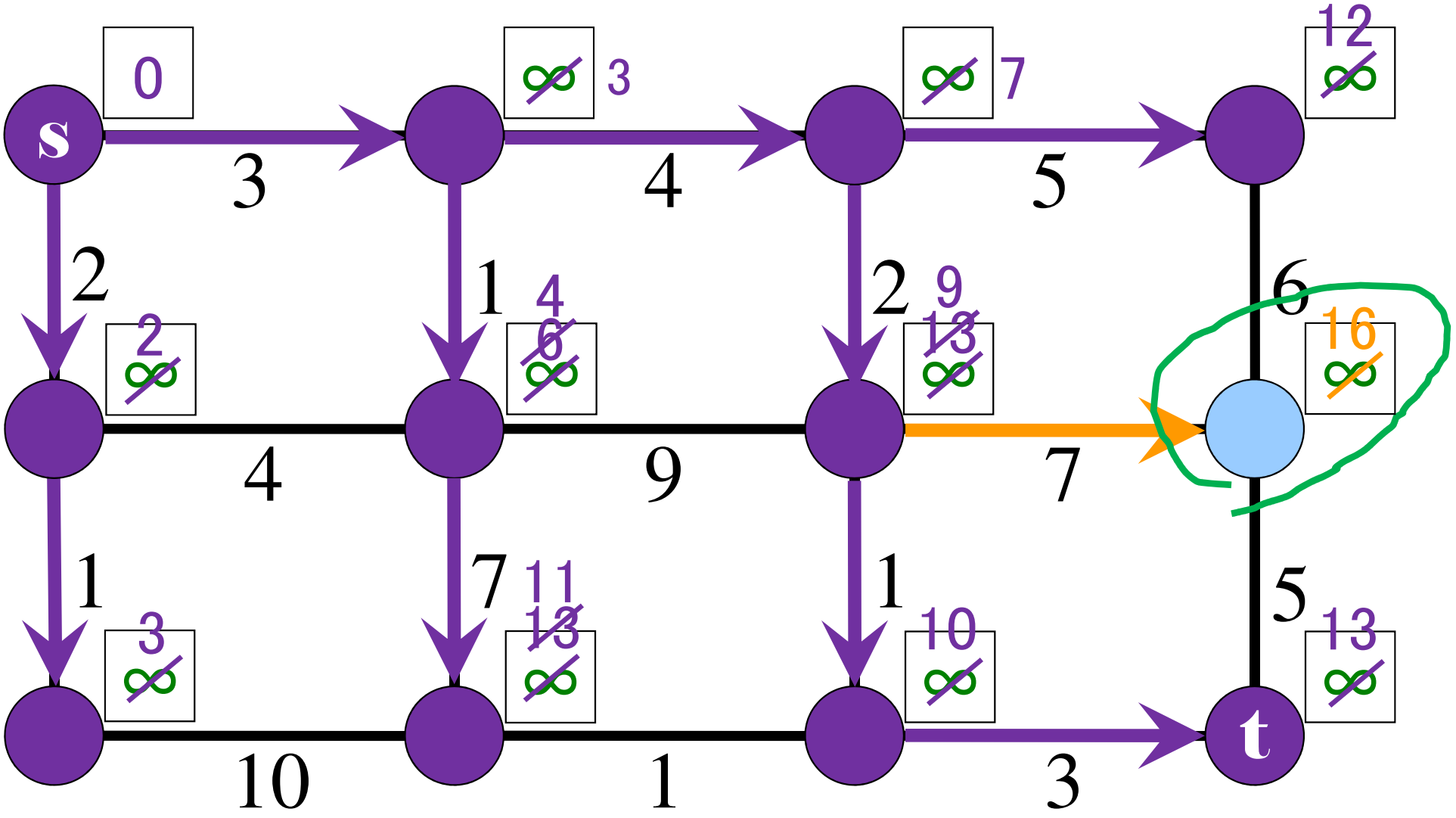
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



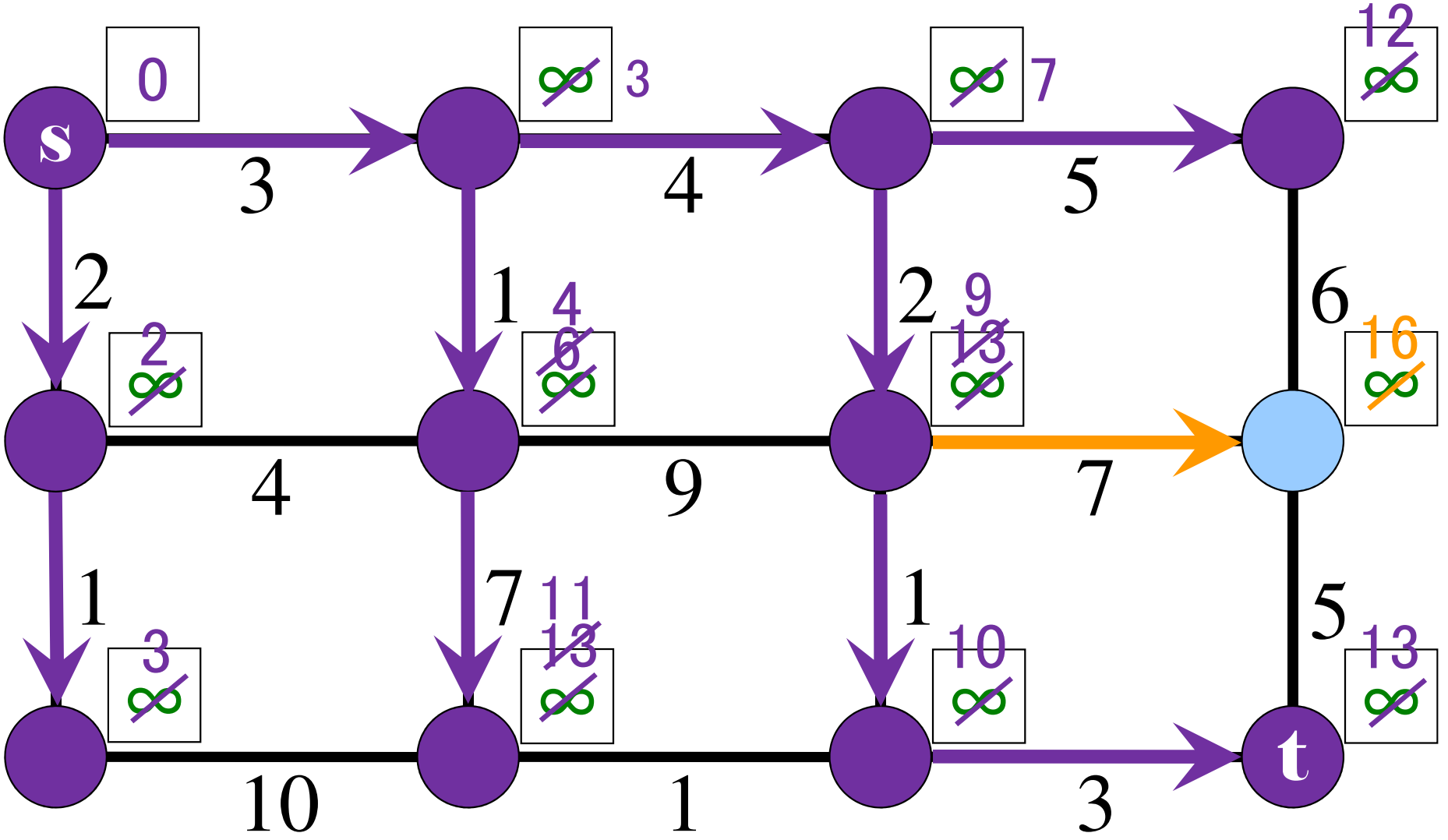
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

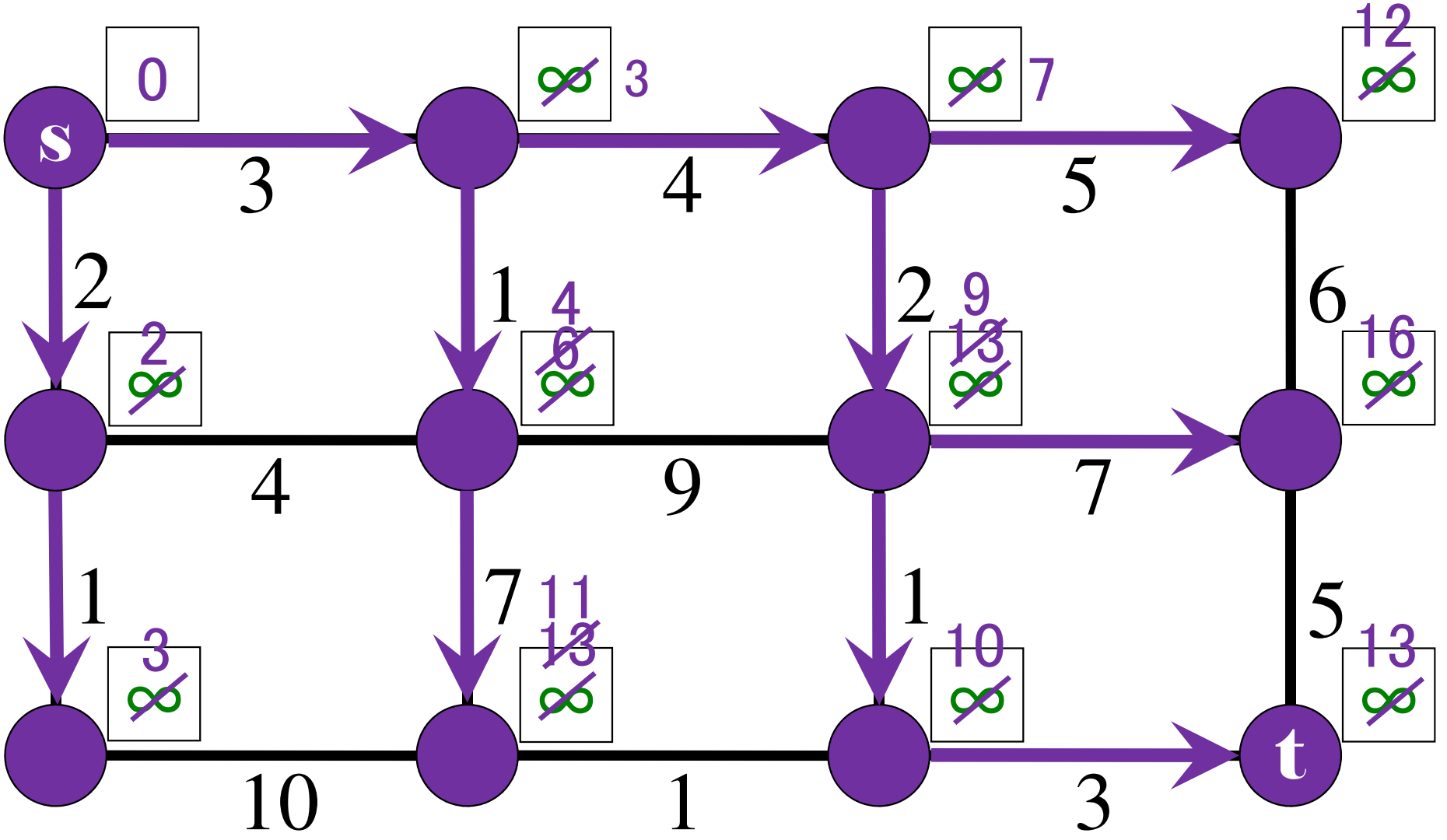


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

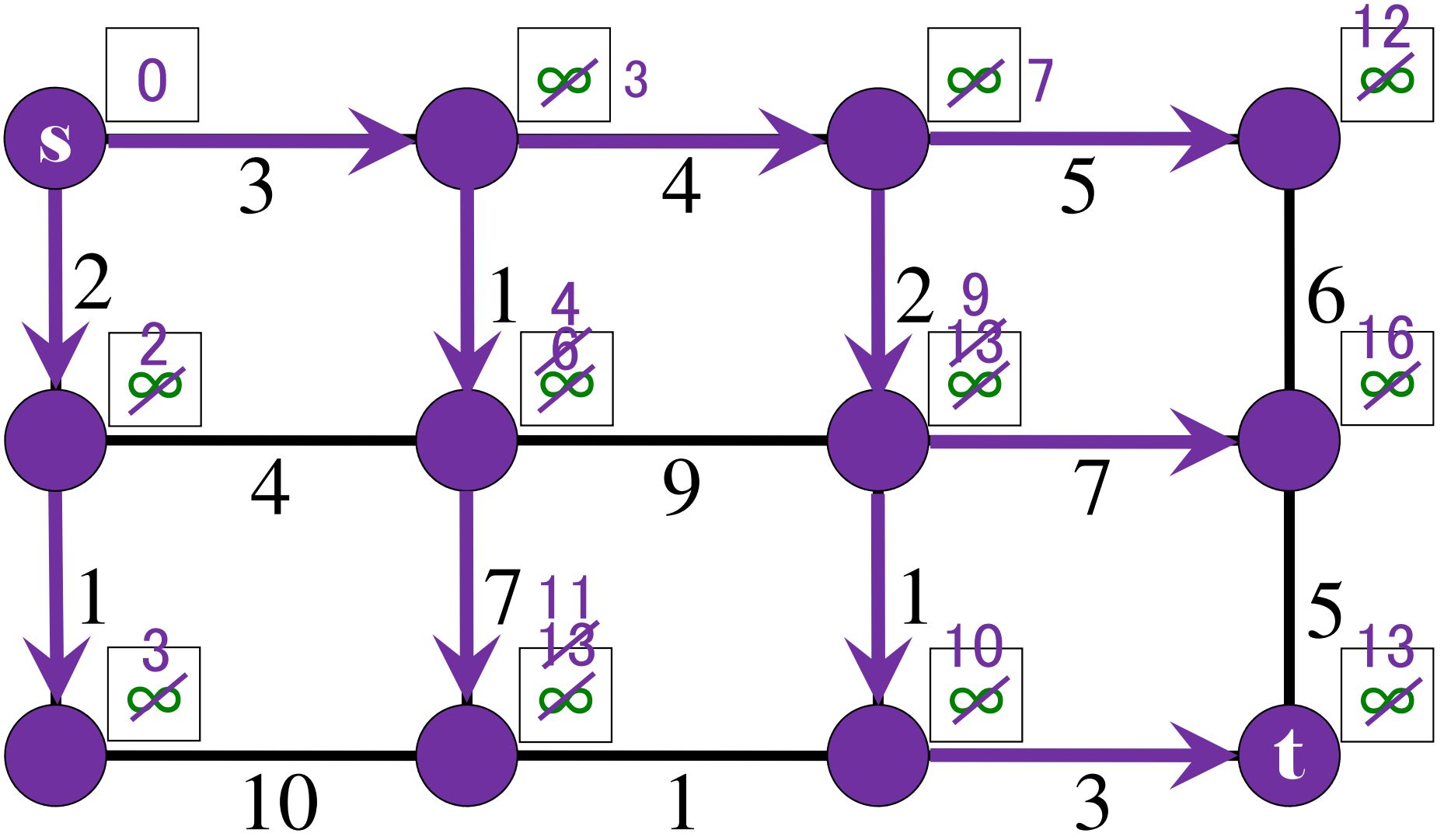
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



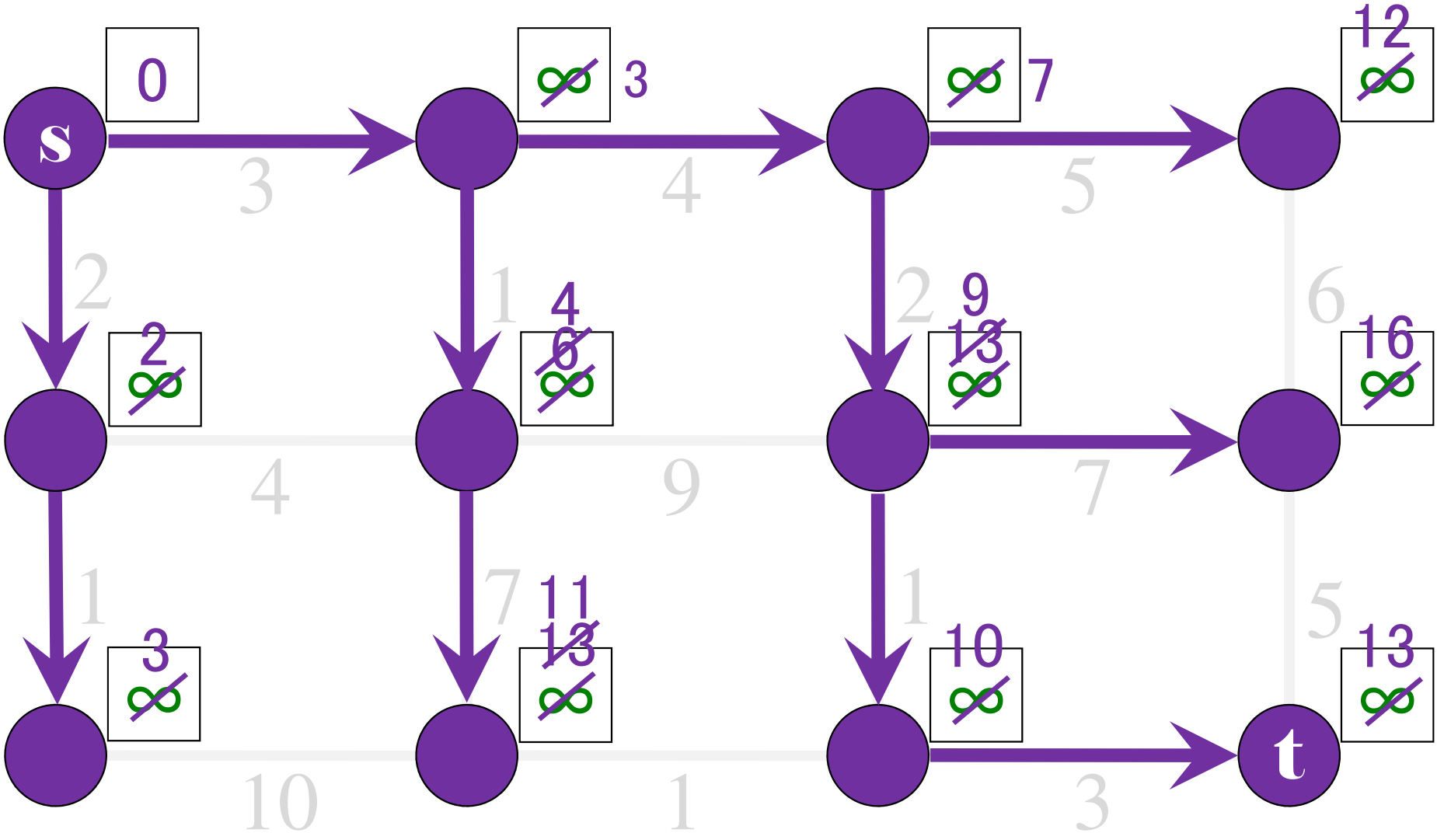
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



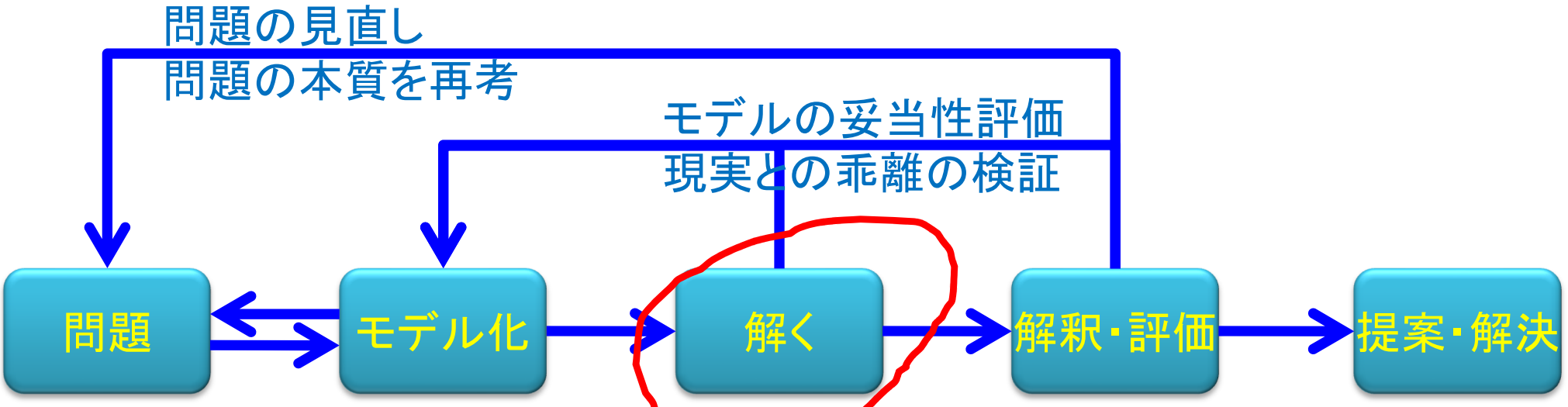
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

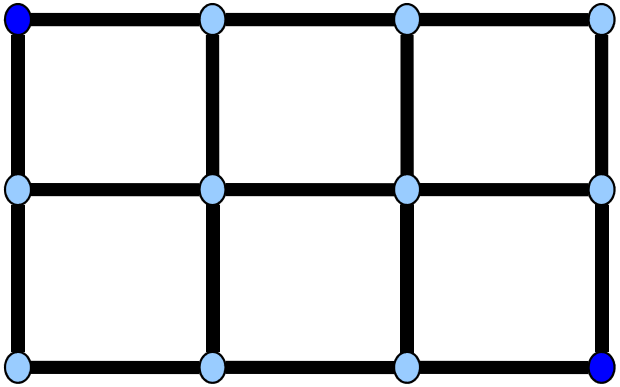
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

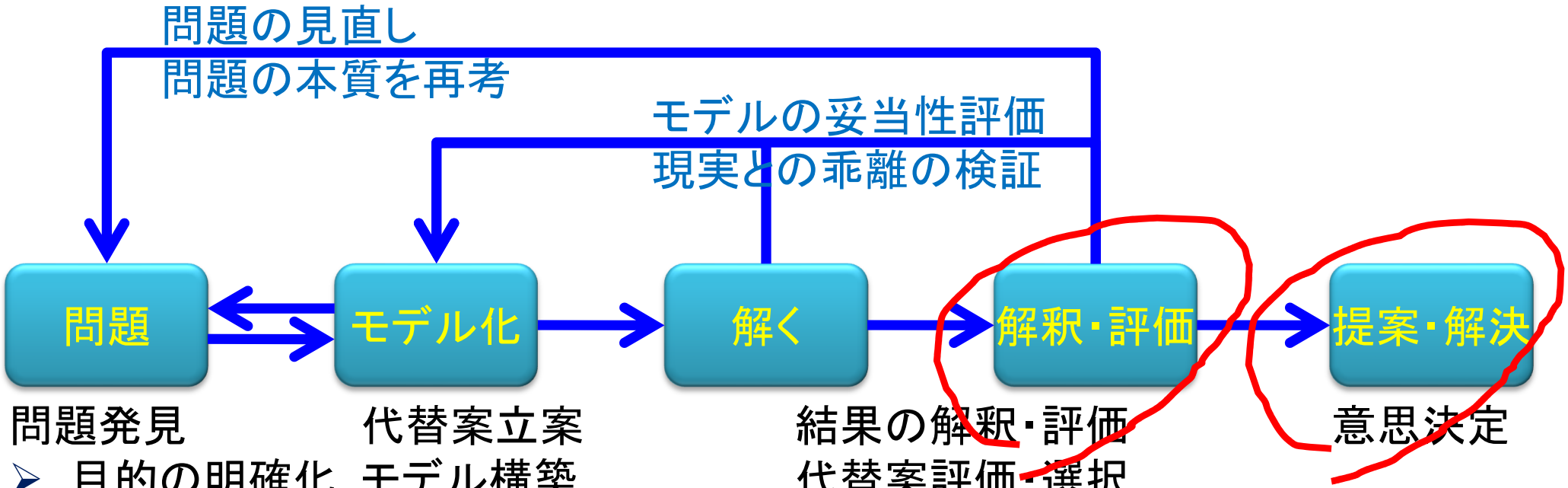
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

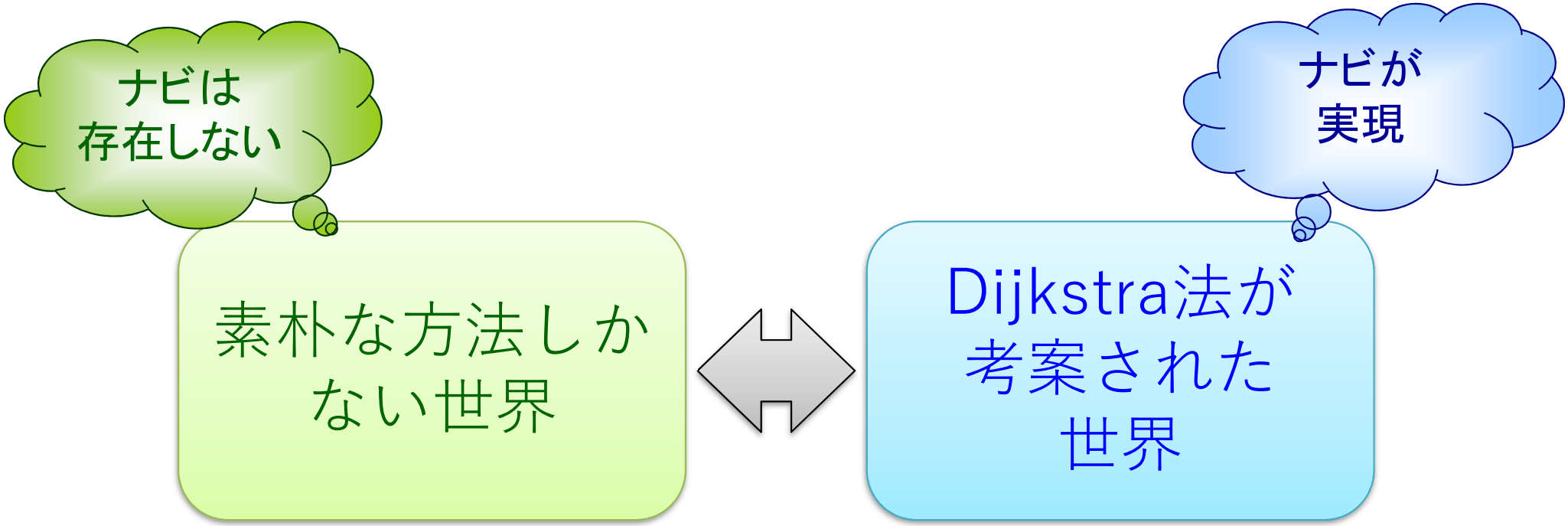
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

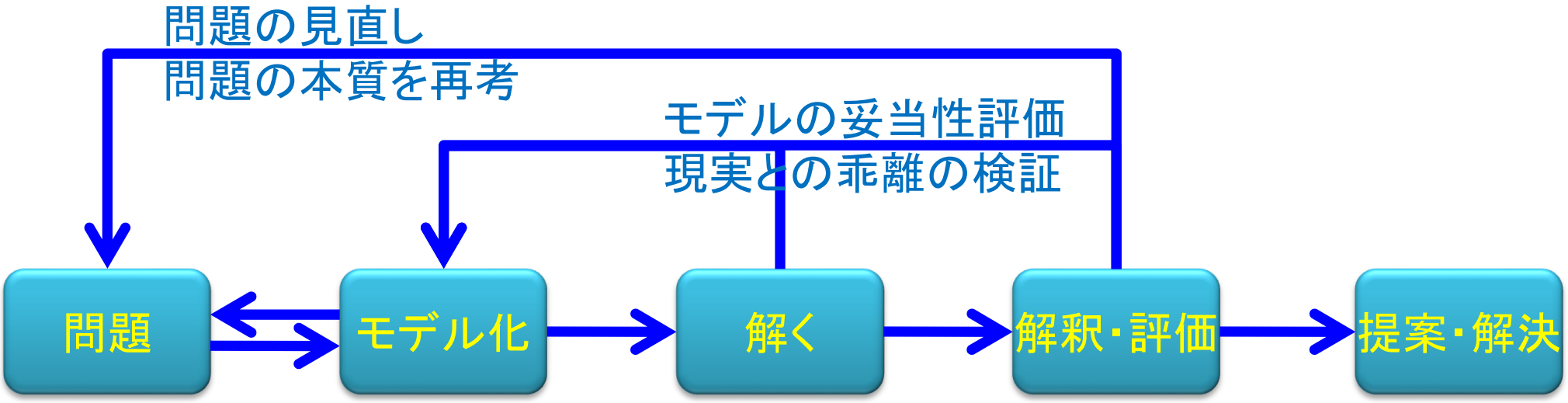
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



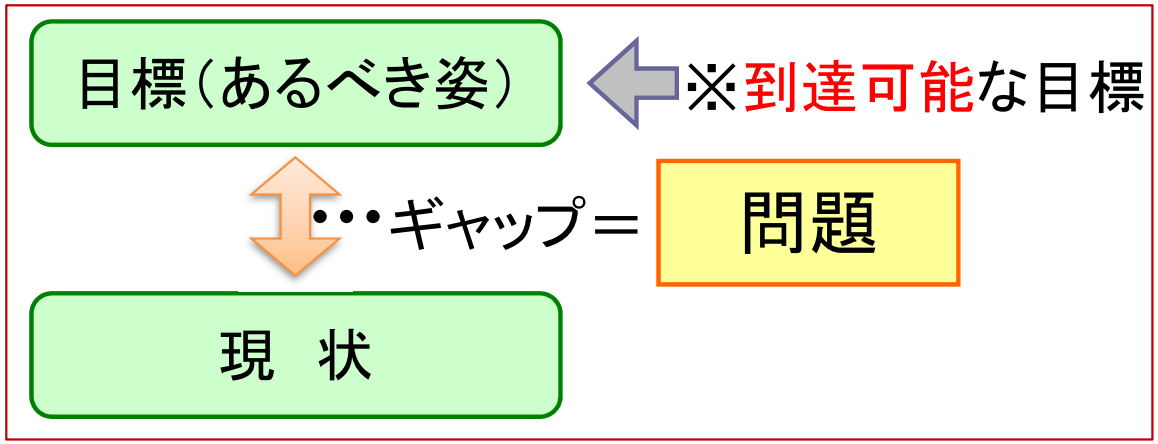
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

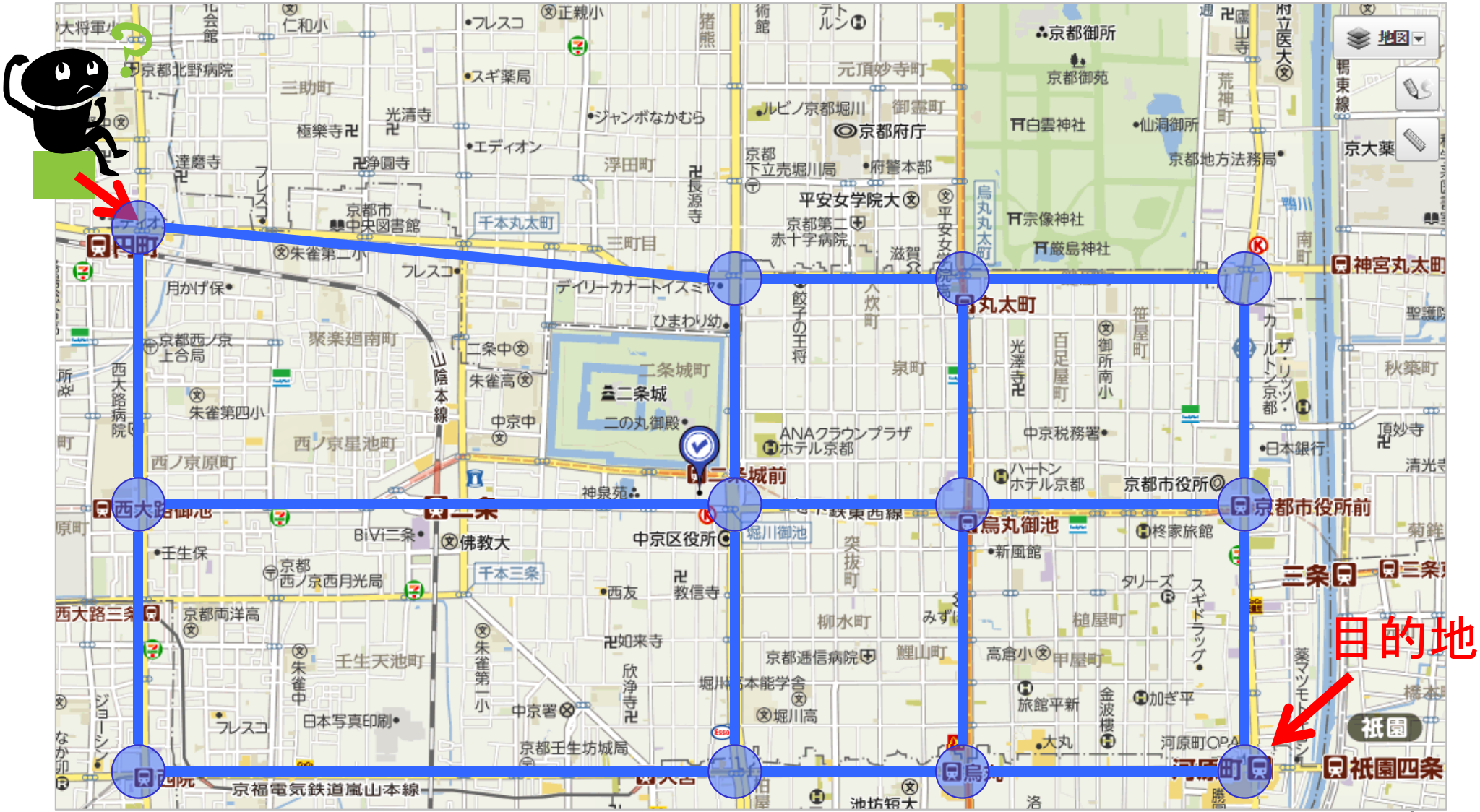
意思決定

問題の定義



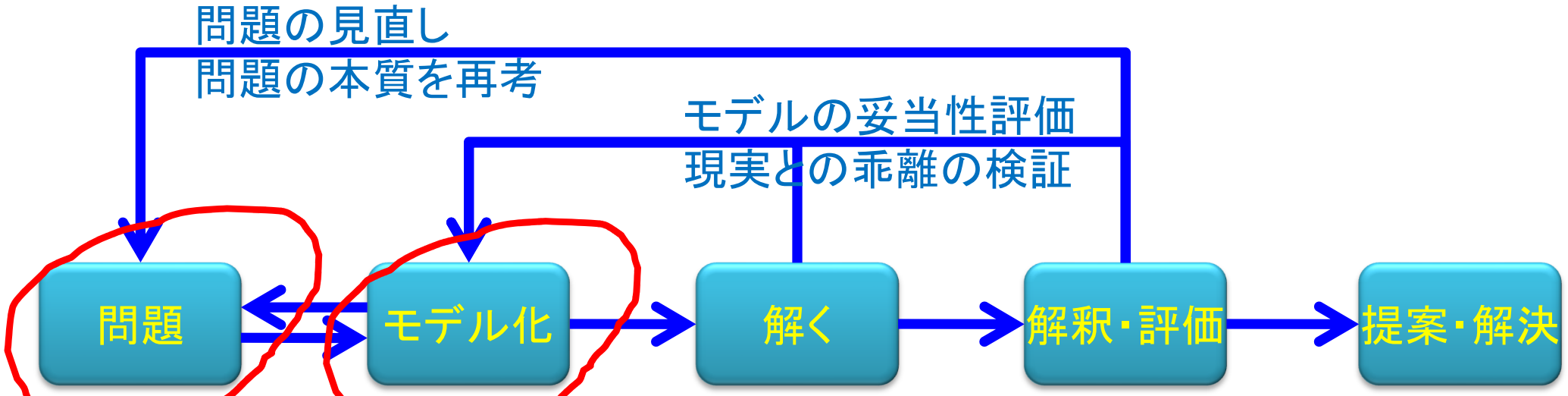
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

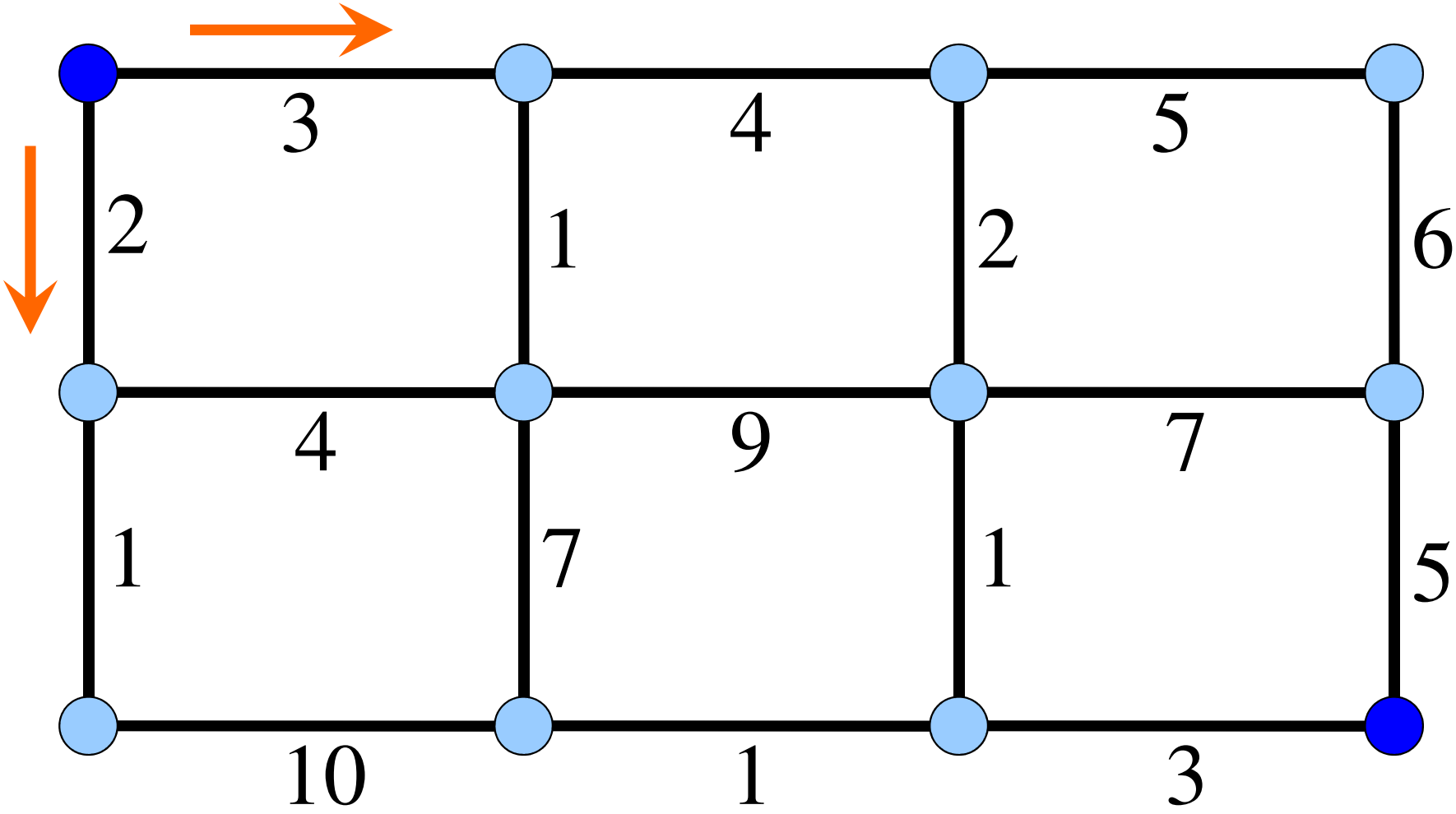
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

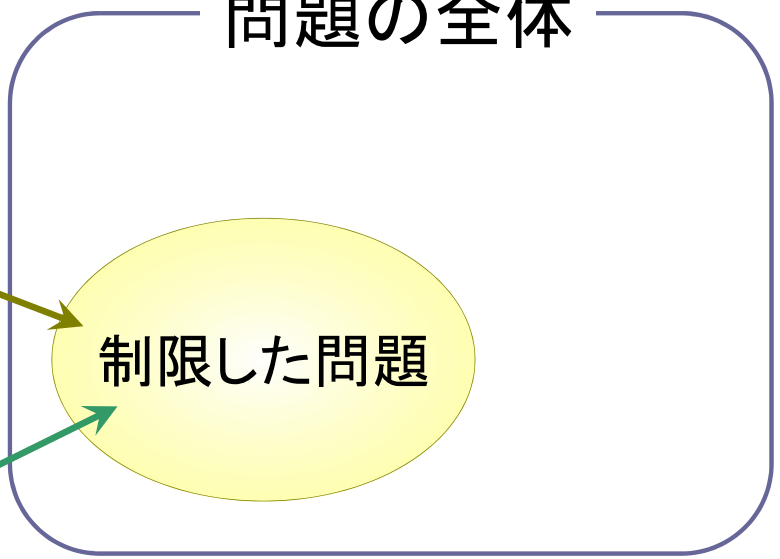


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

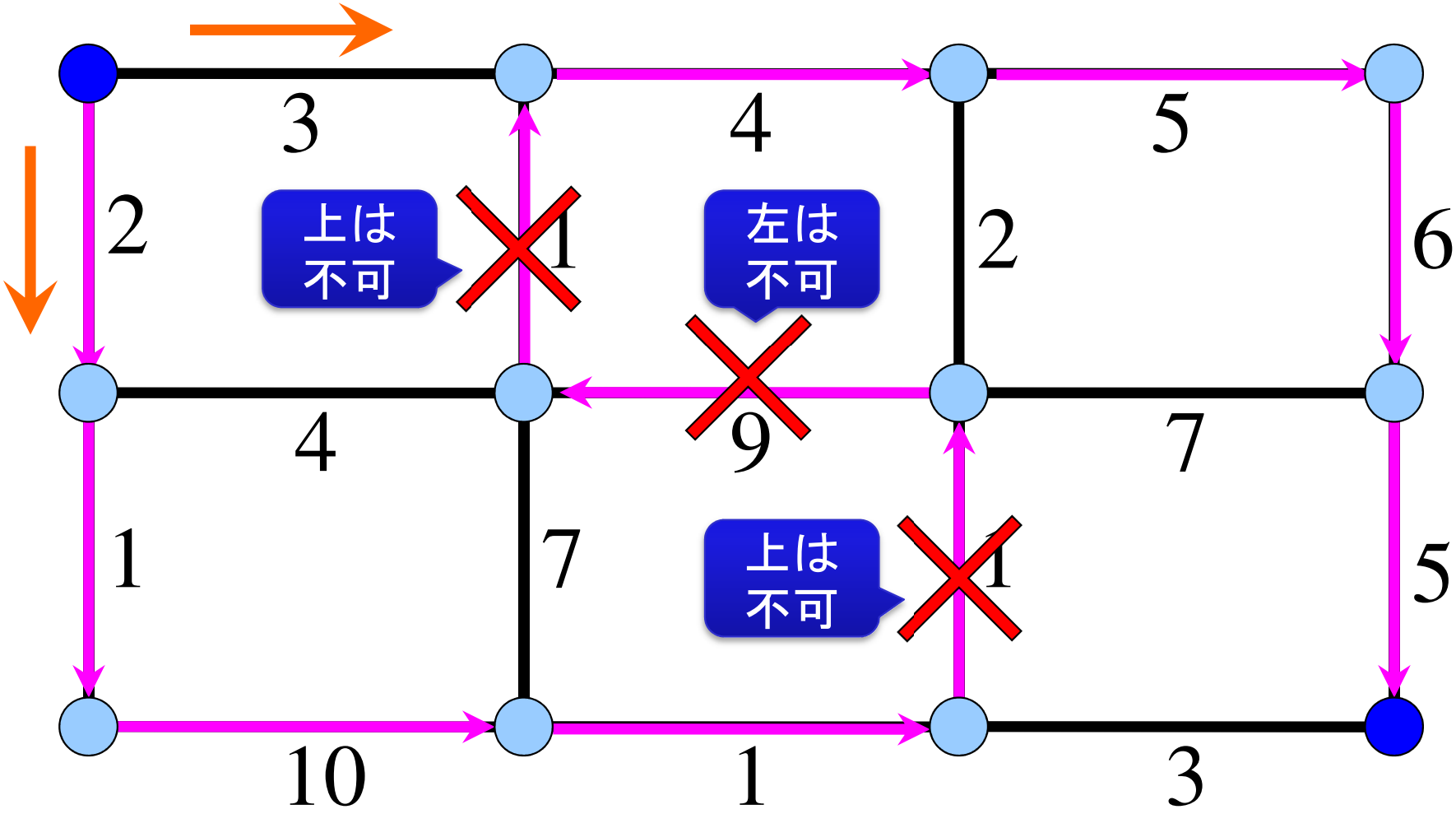
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

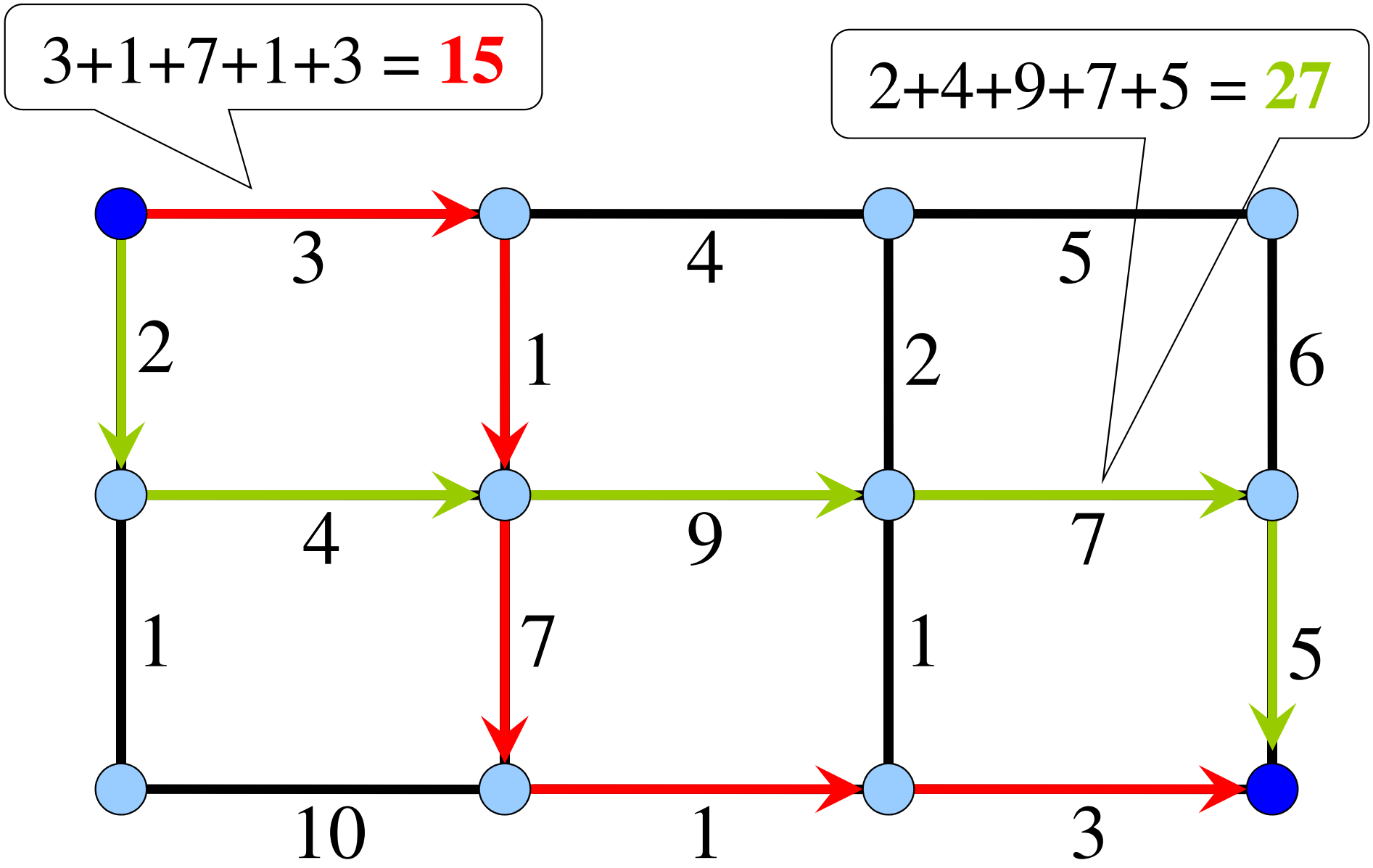
難しいなら易しくすればいいのさ！

制限した問題

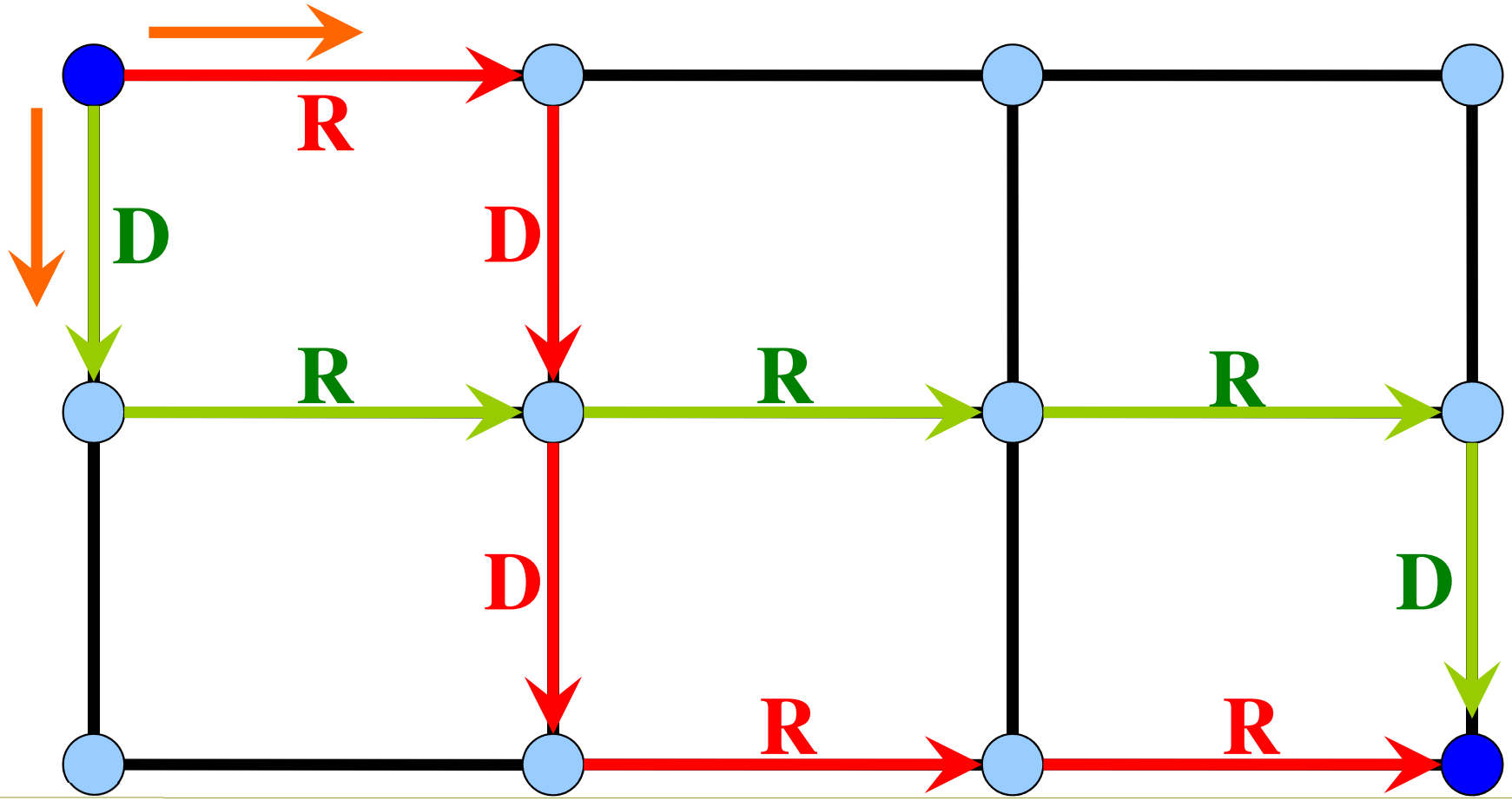
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



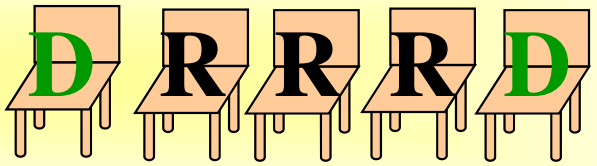
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



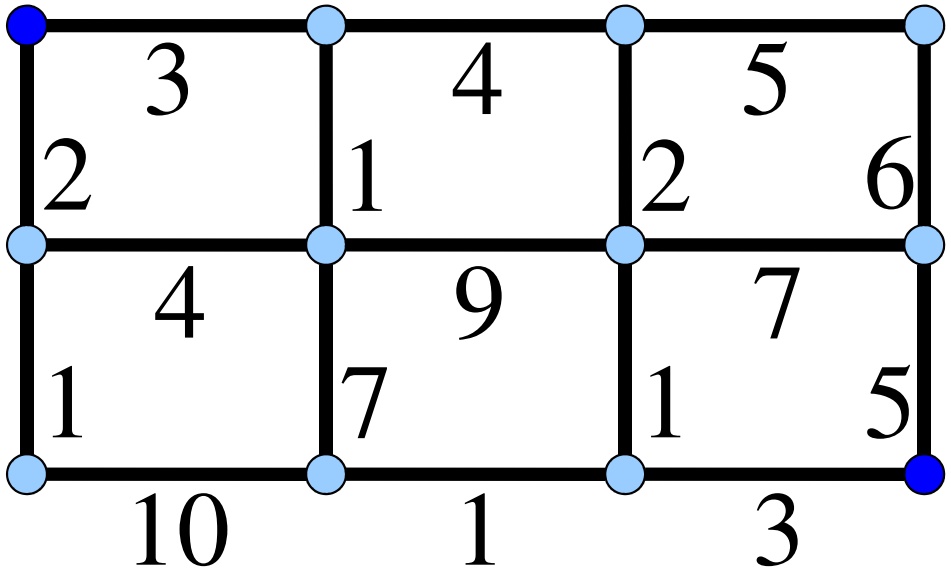
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

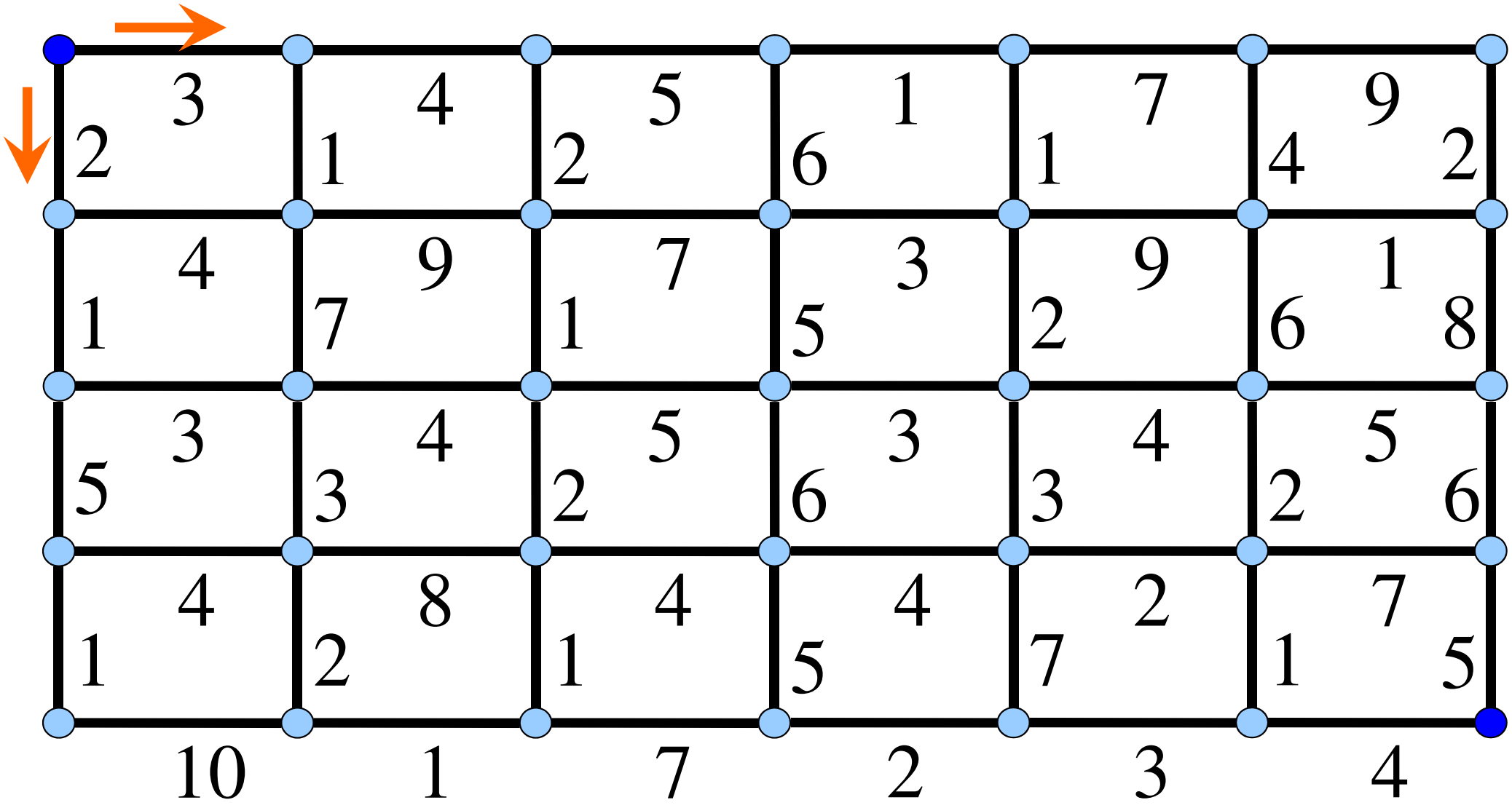
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

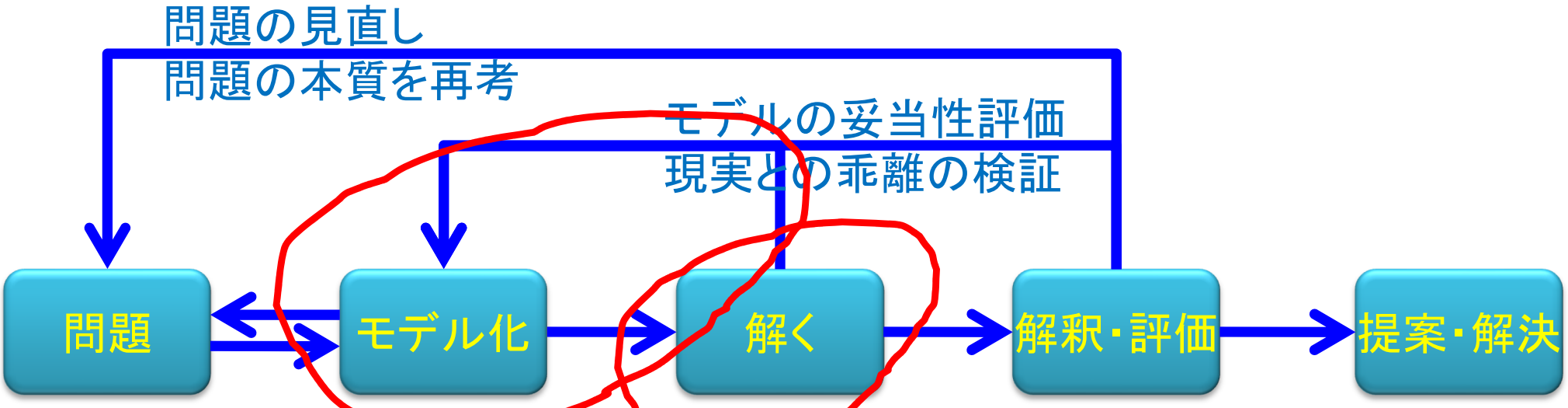
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

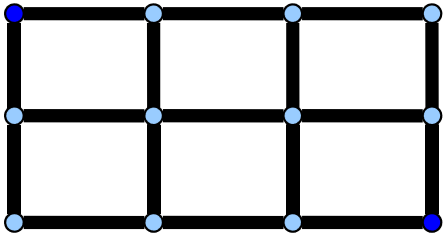
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

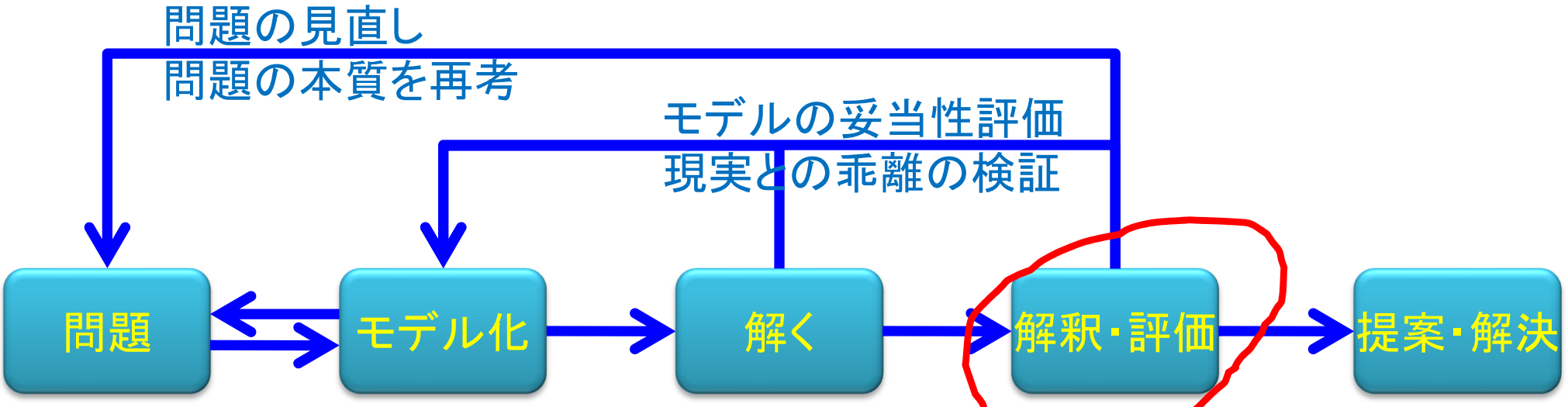
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

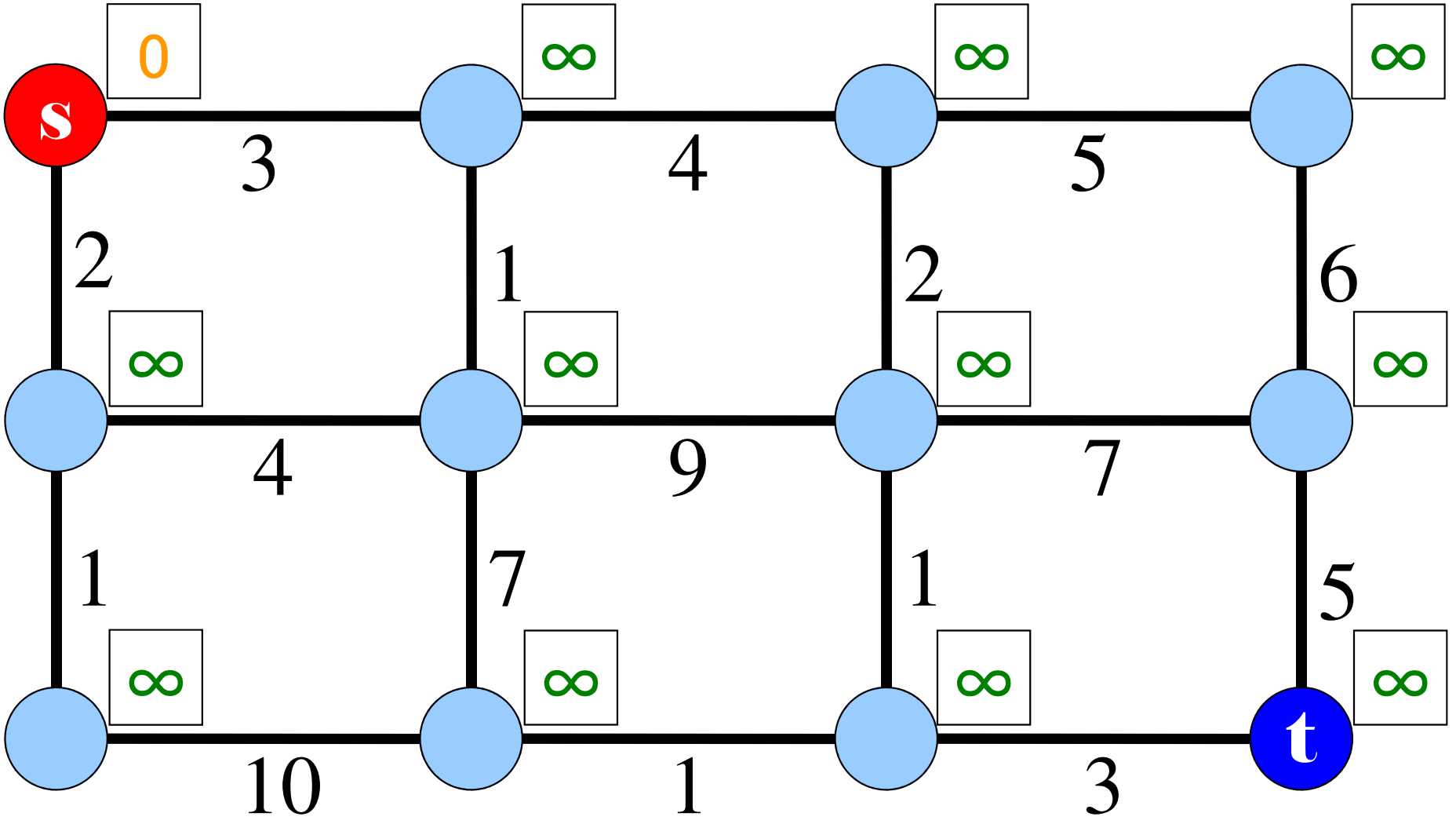
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

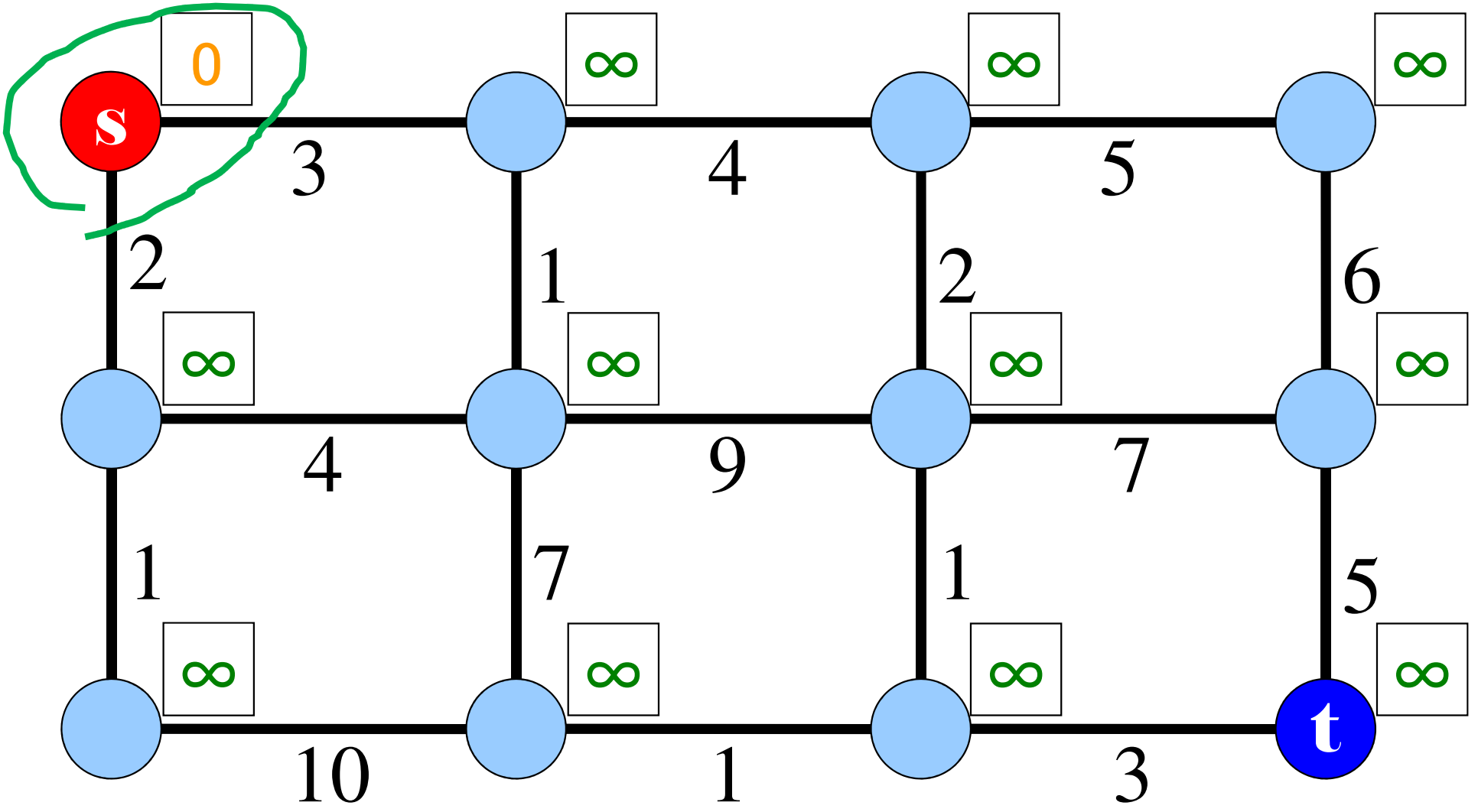
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



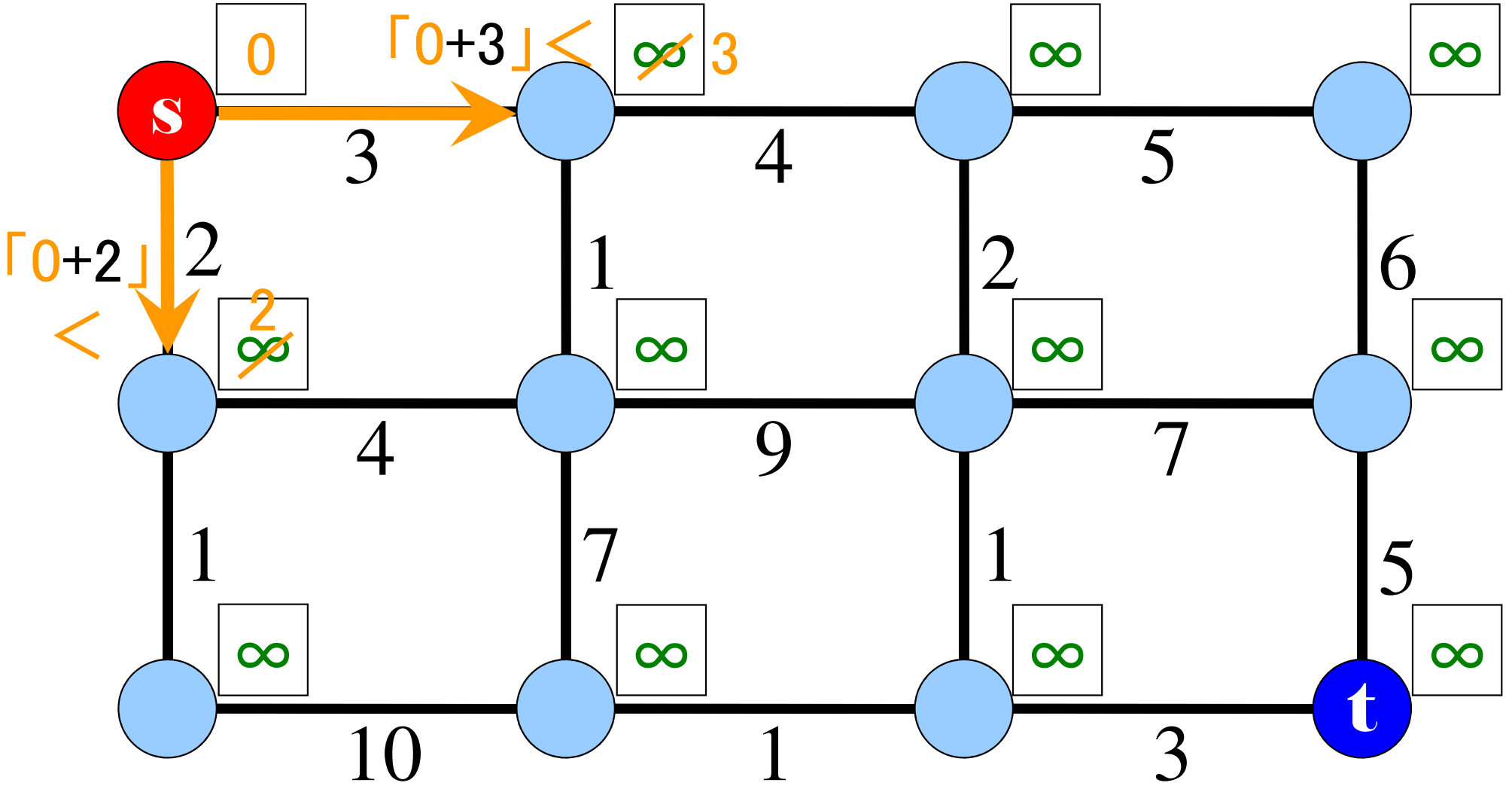
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

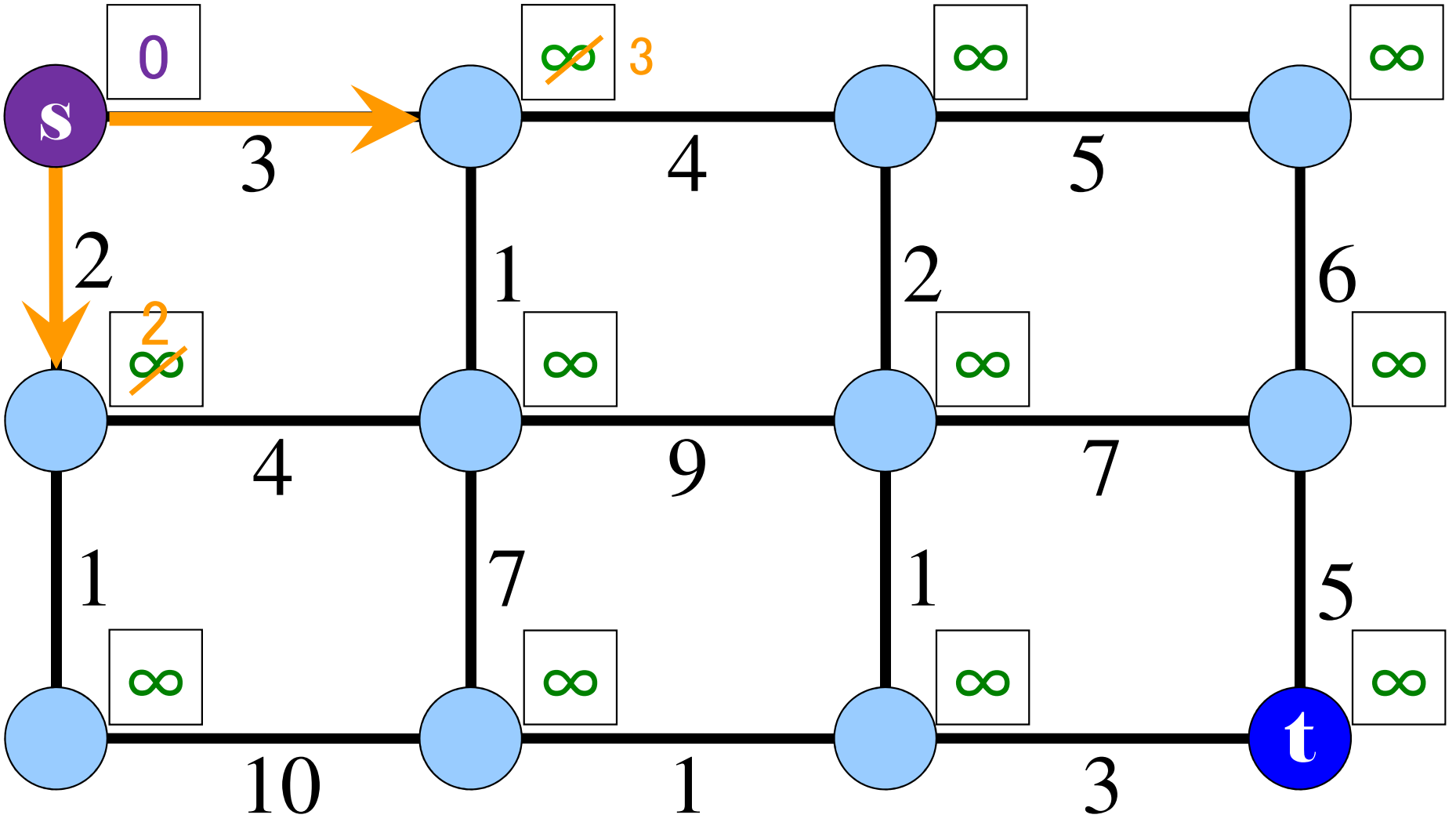


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

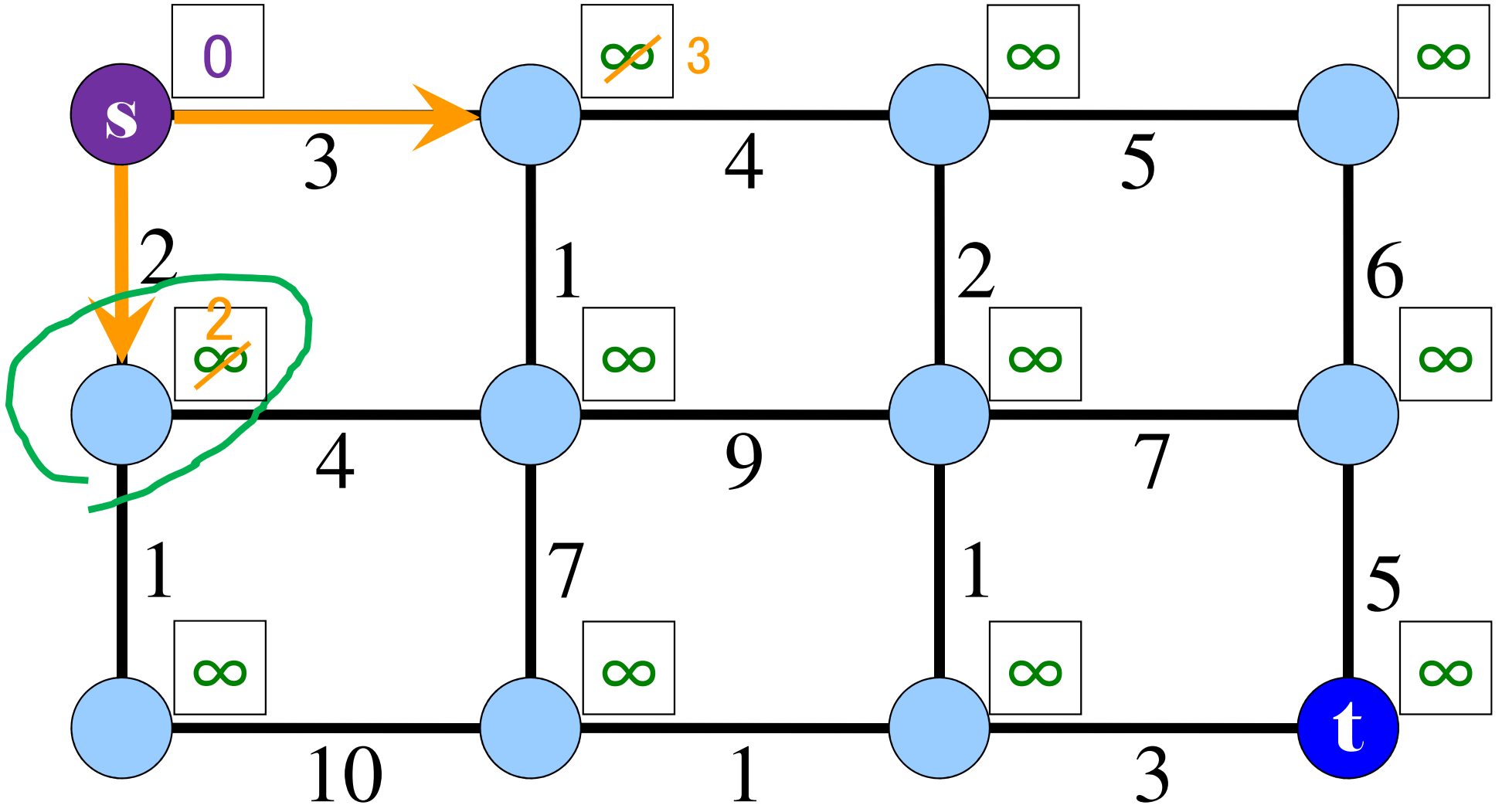
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



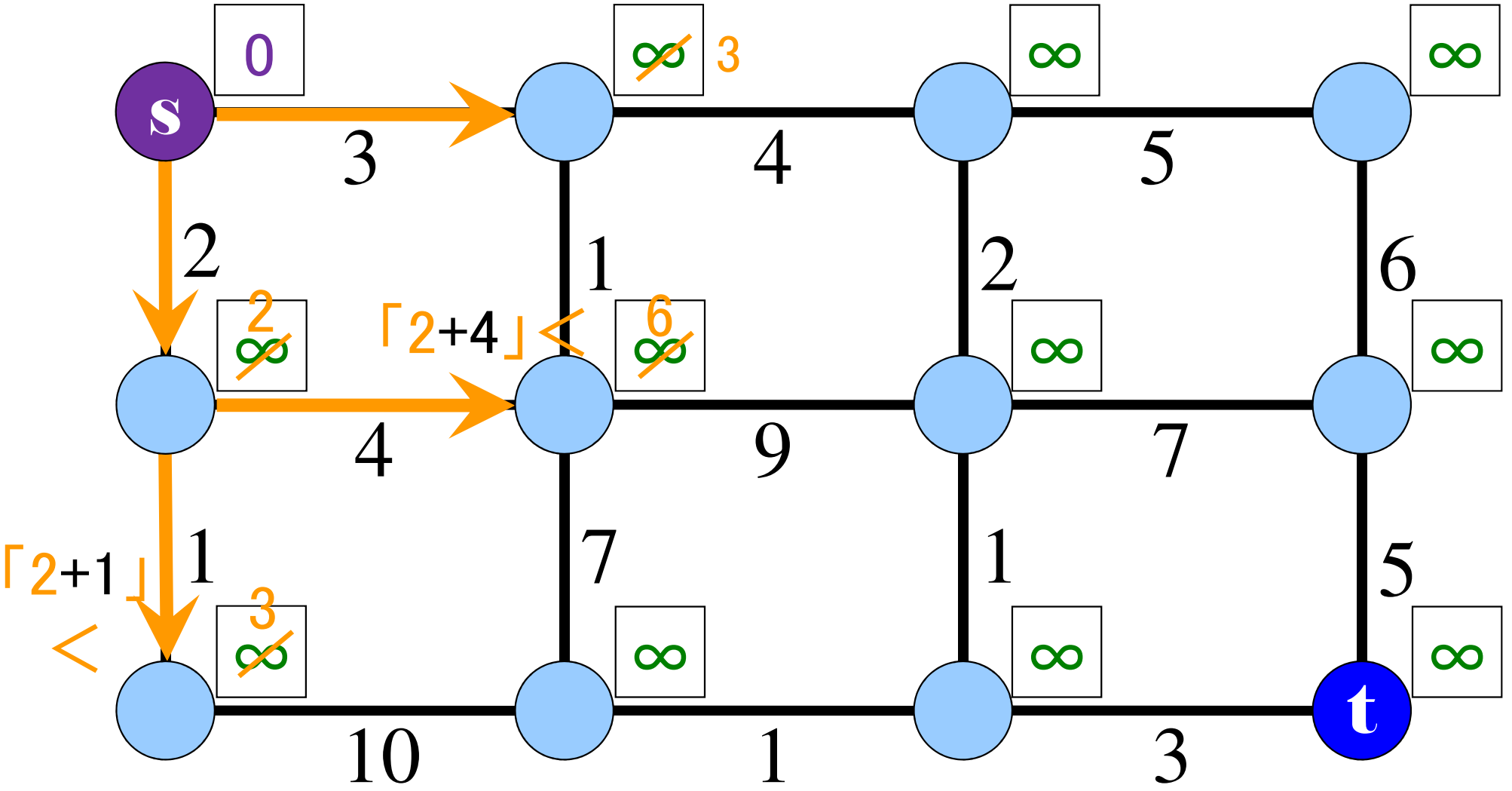
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

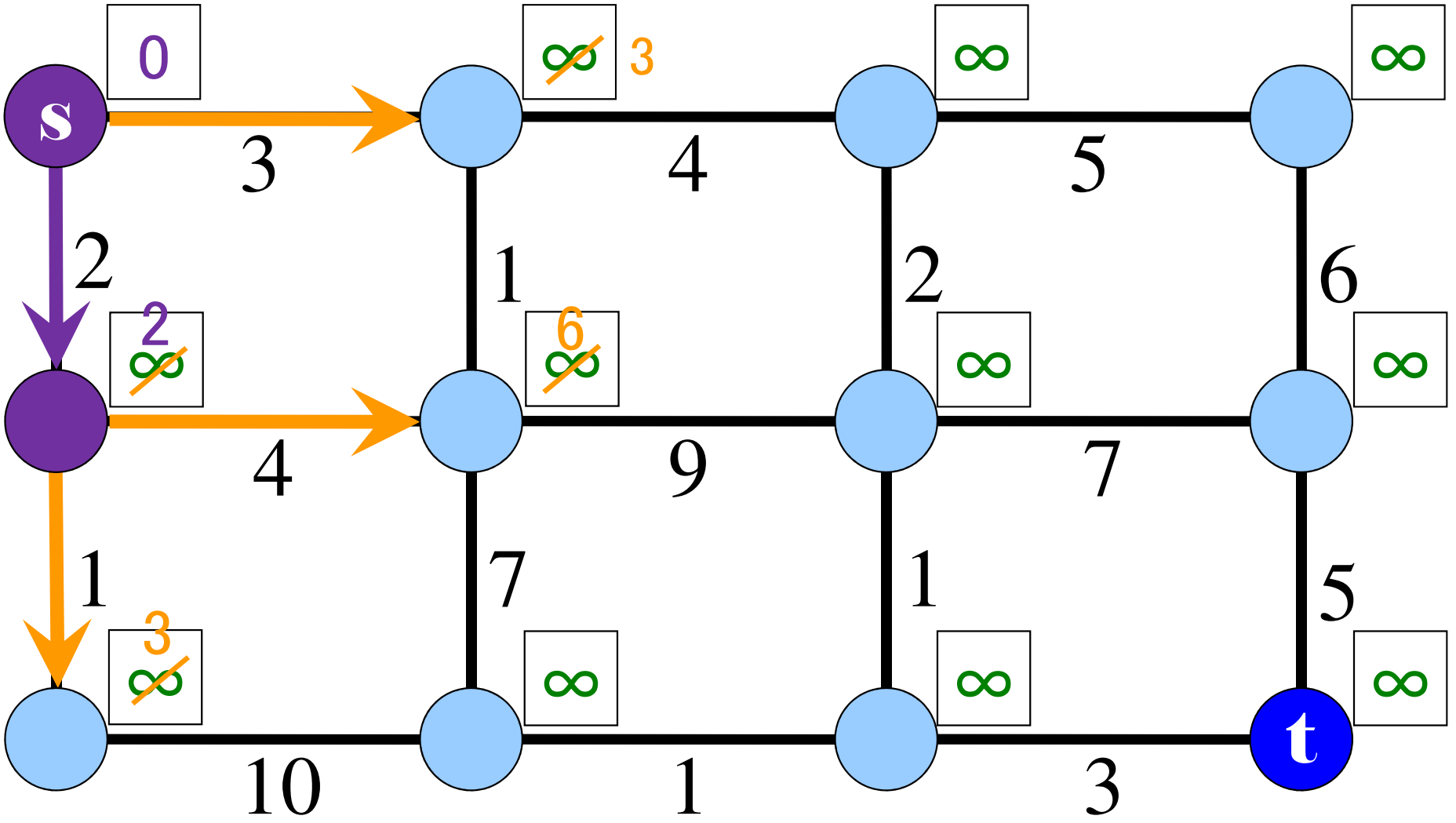


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

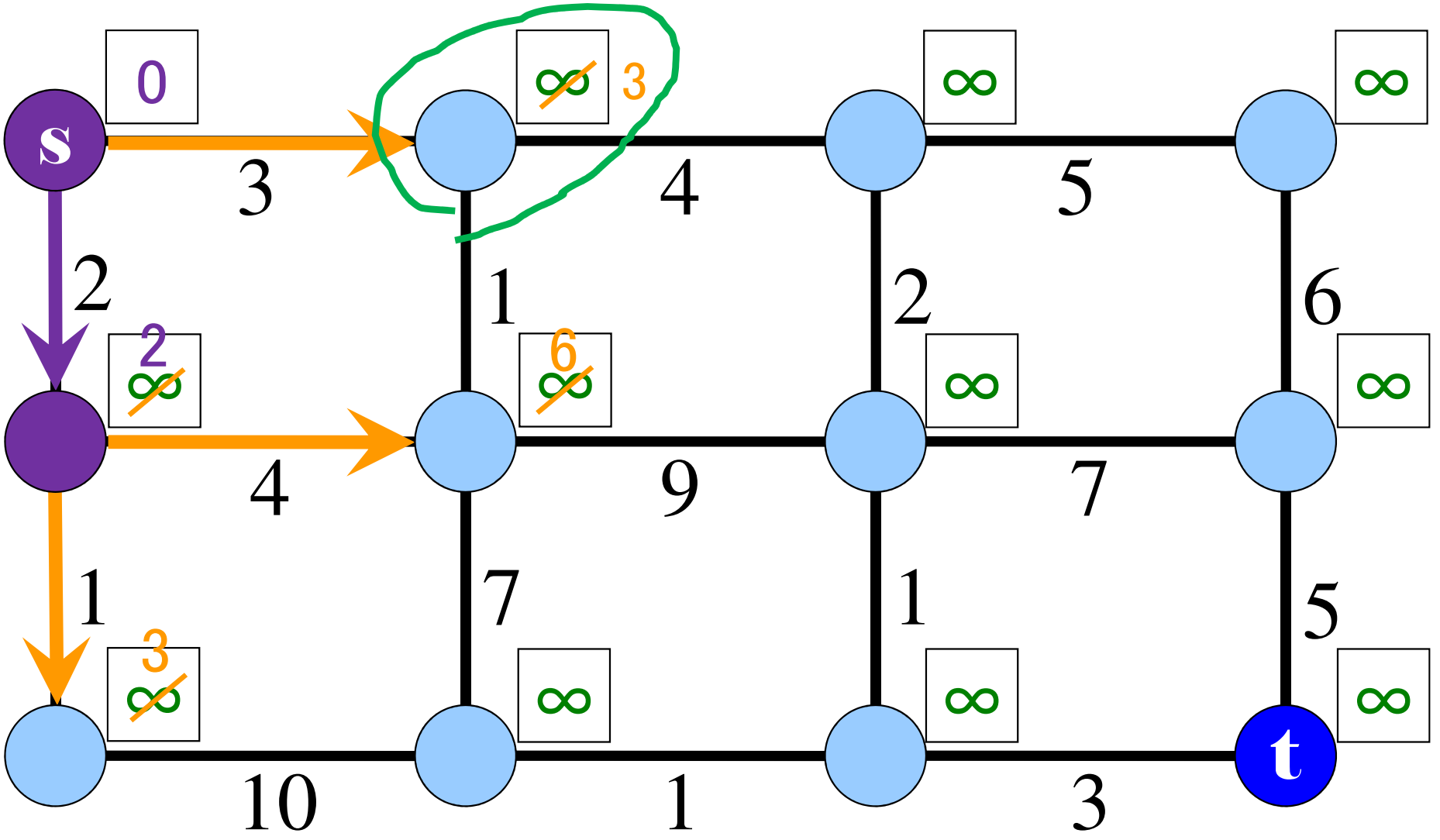
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



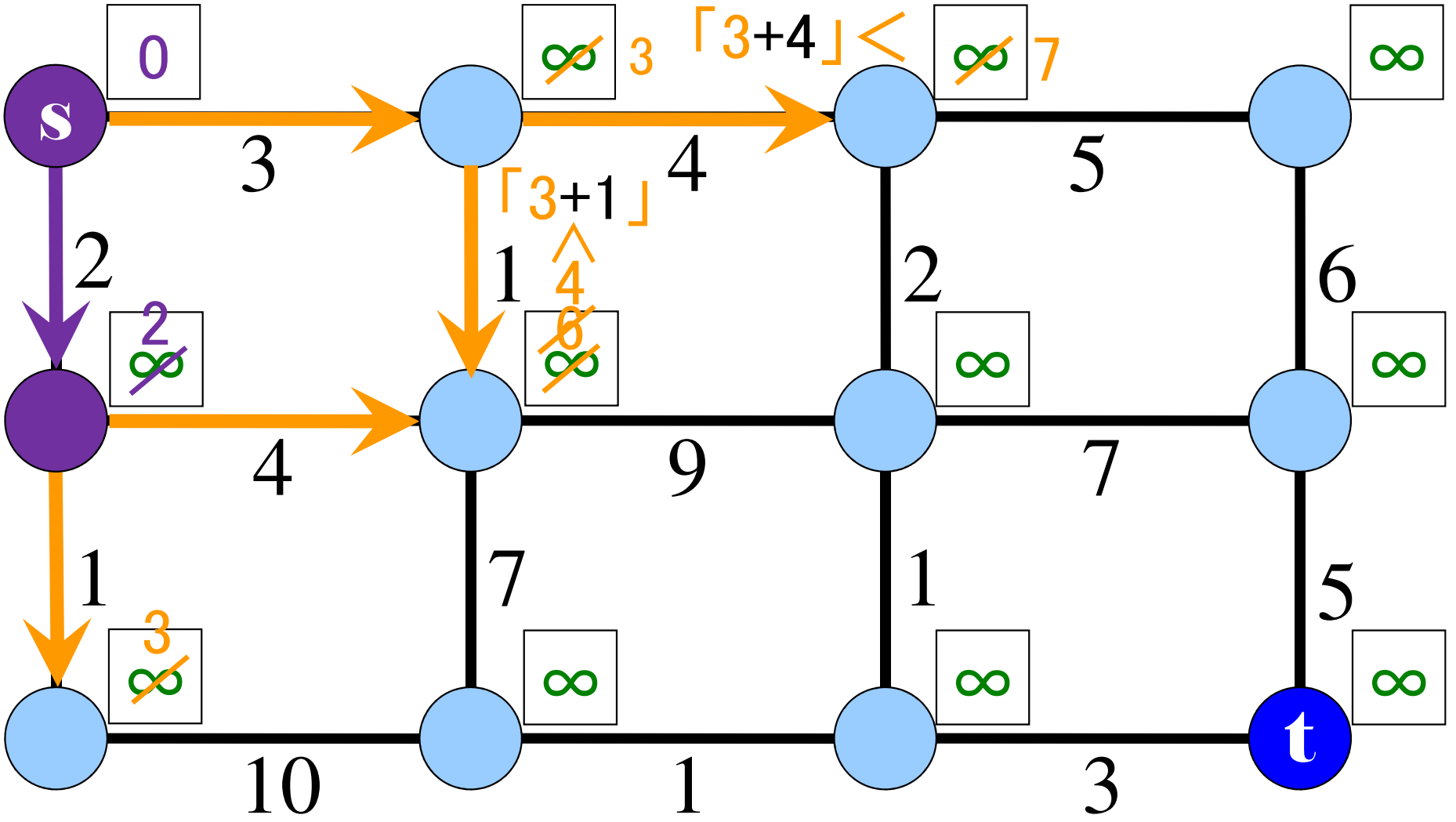
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

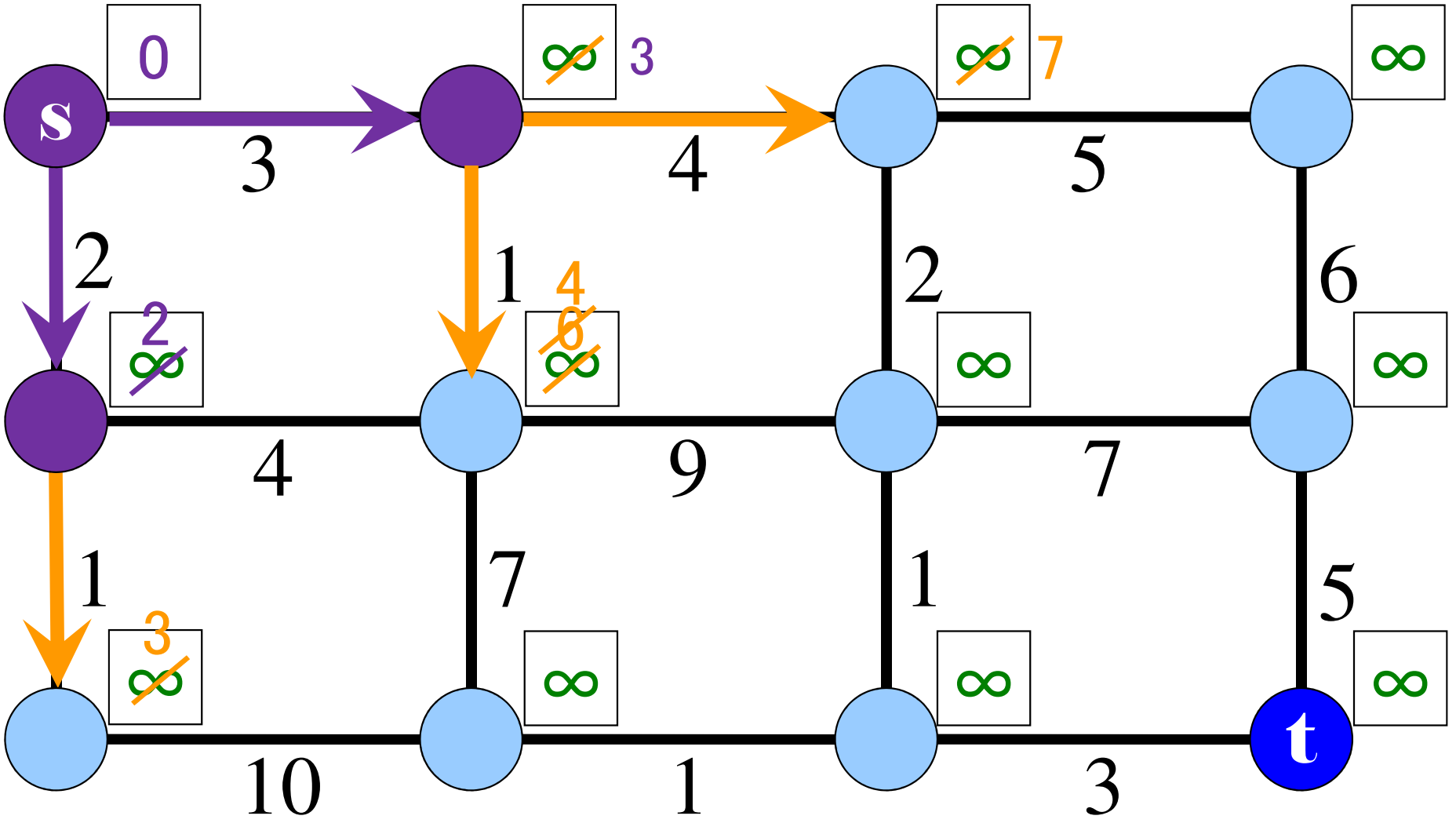


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

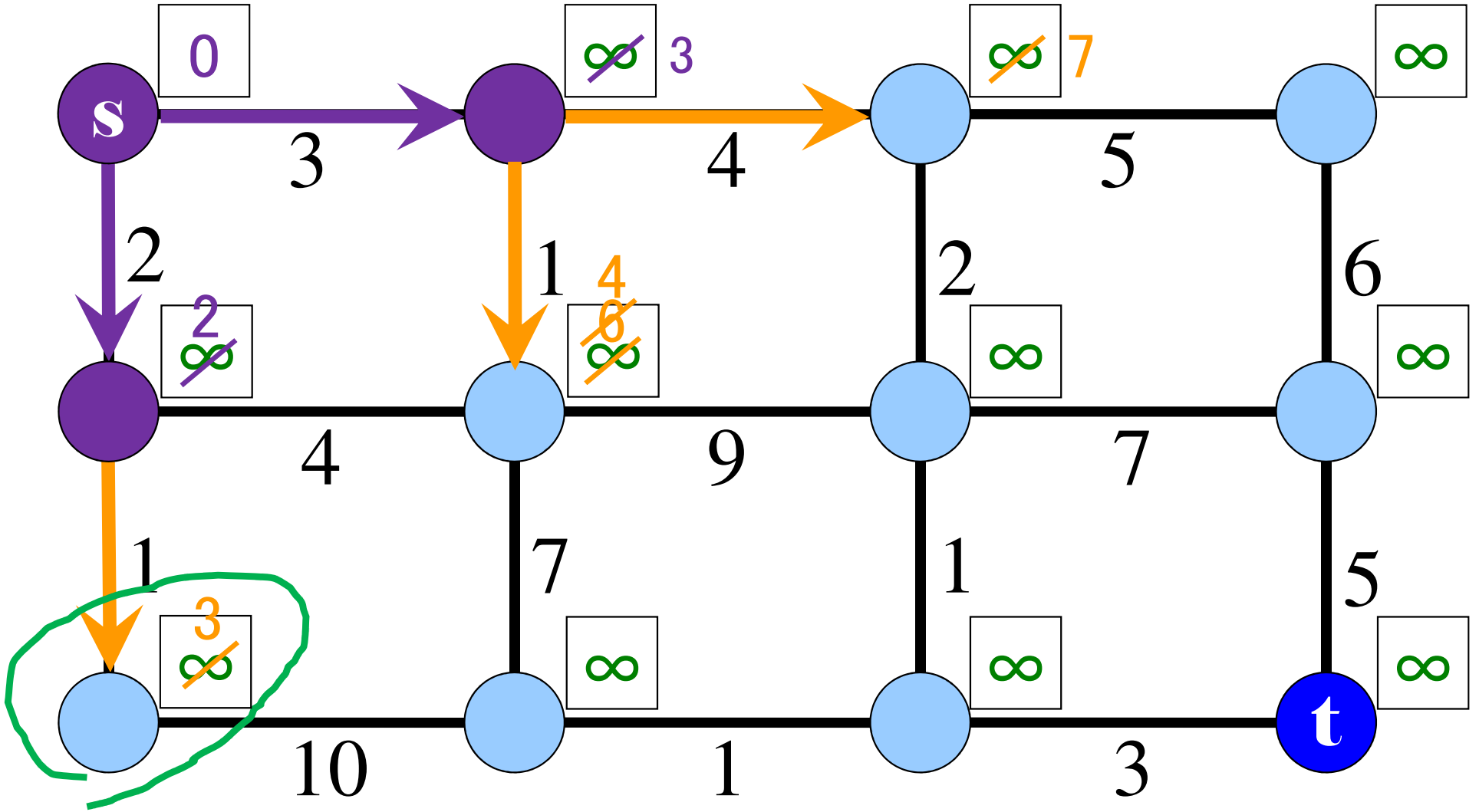
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



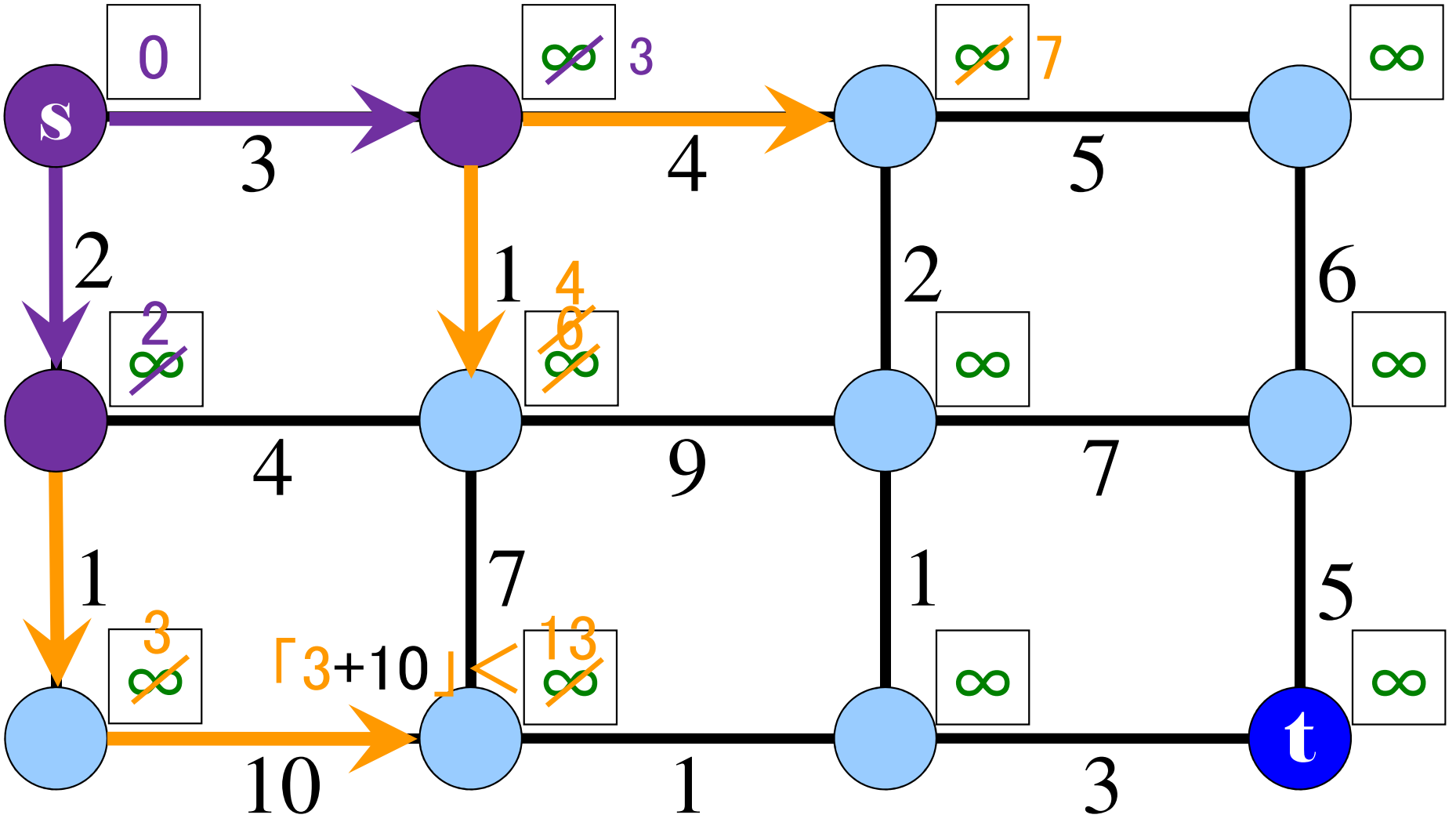
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

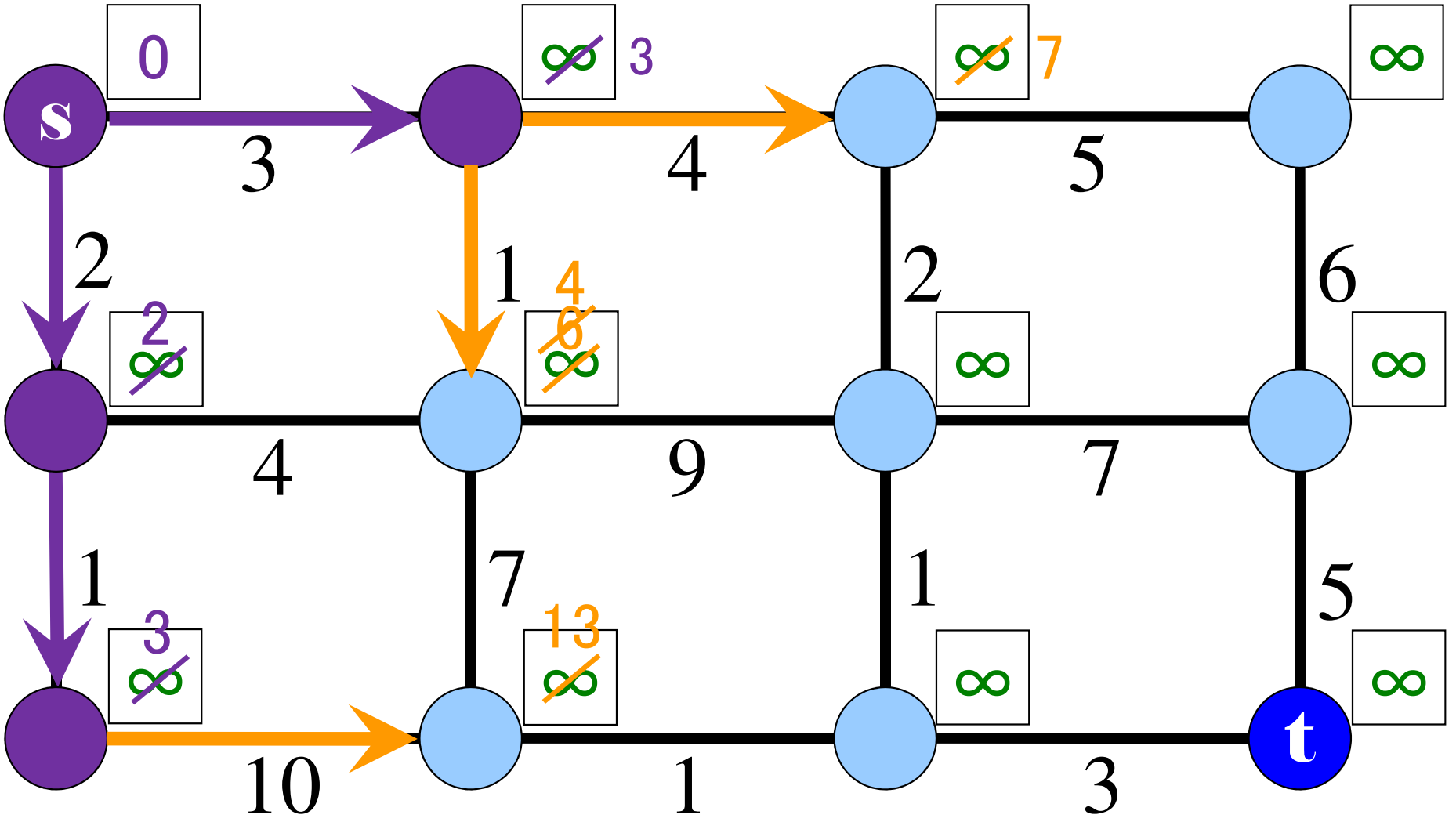


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

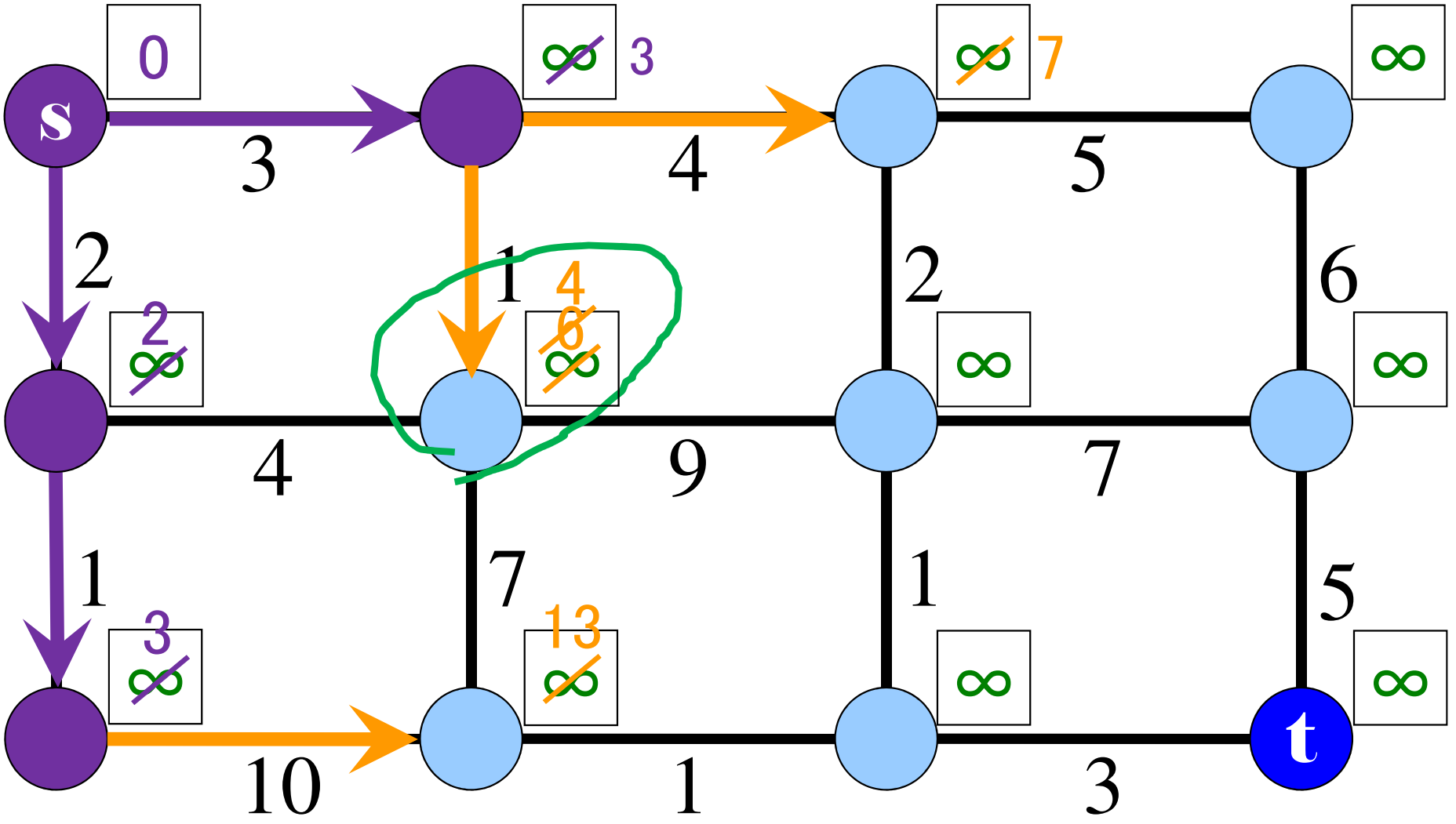
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



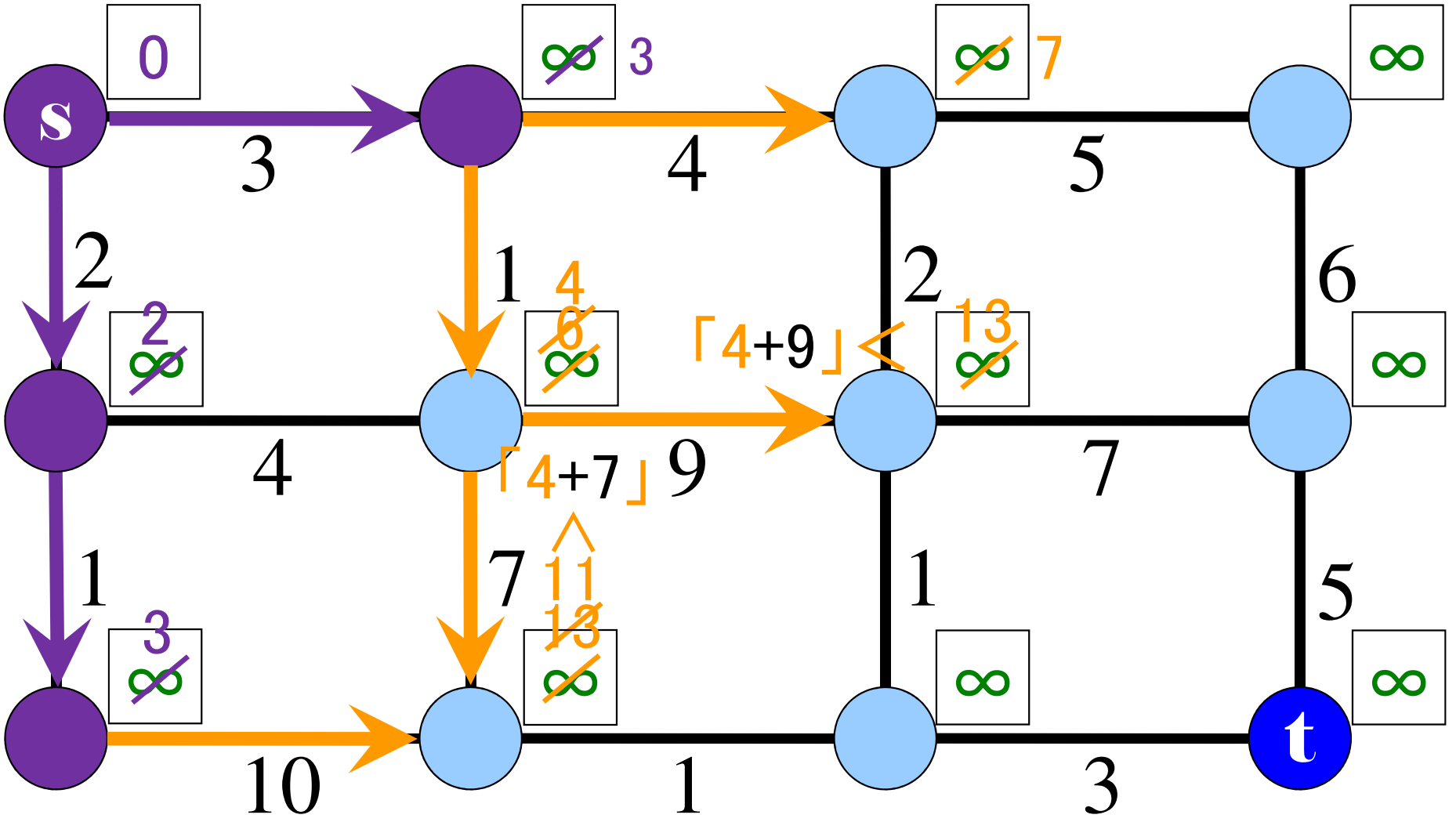
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

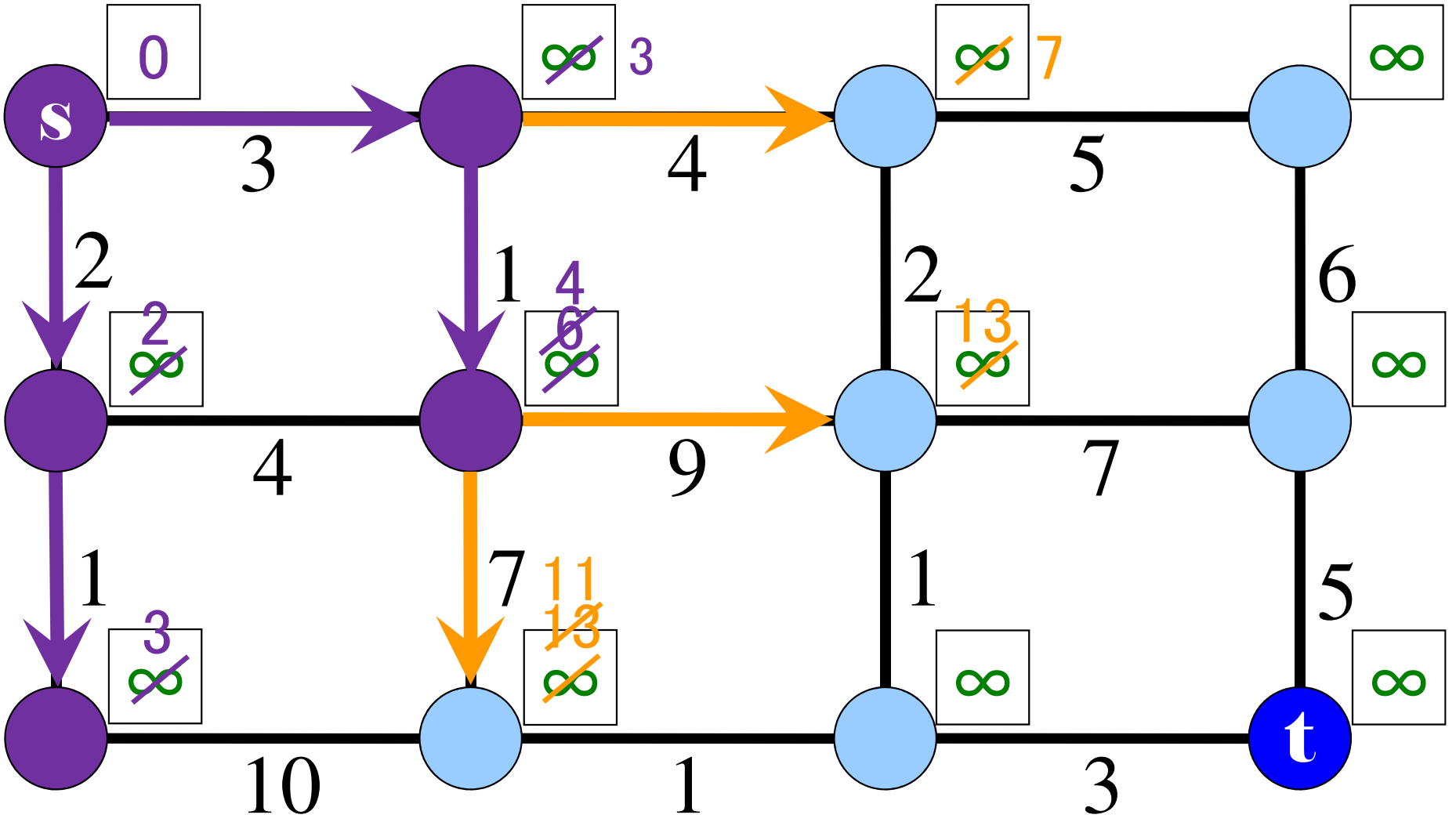


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

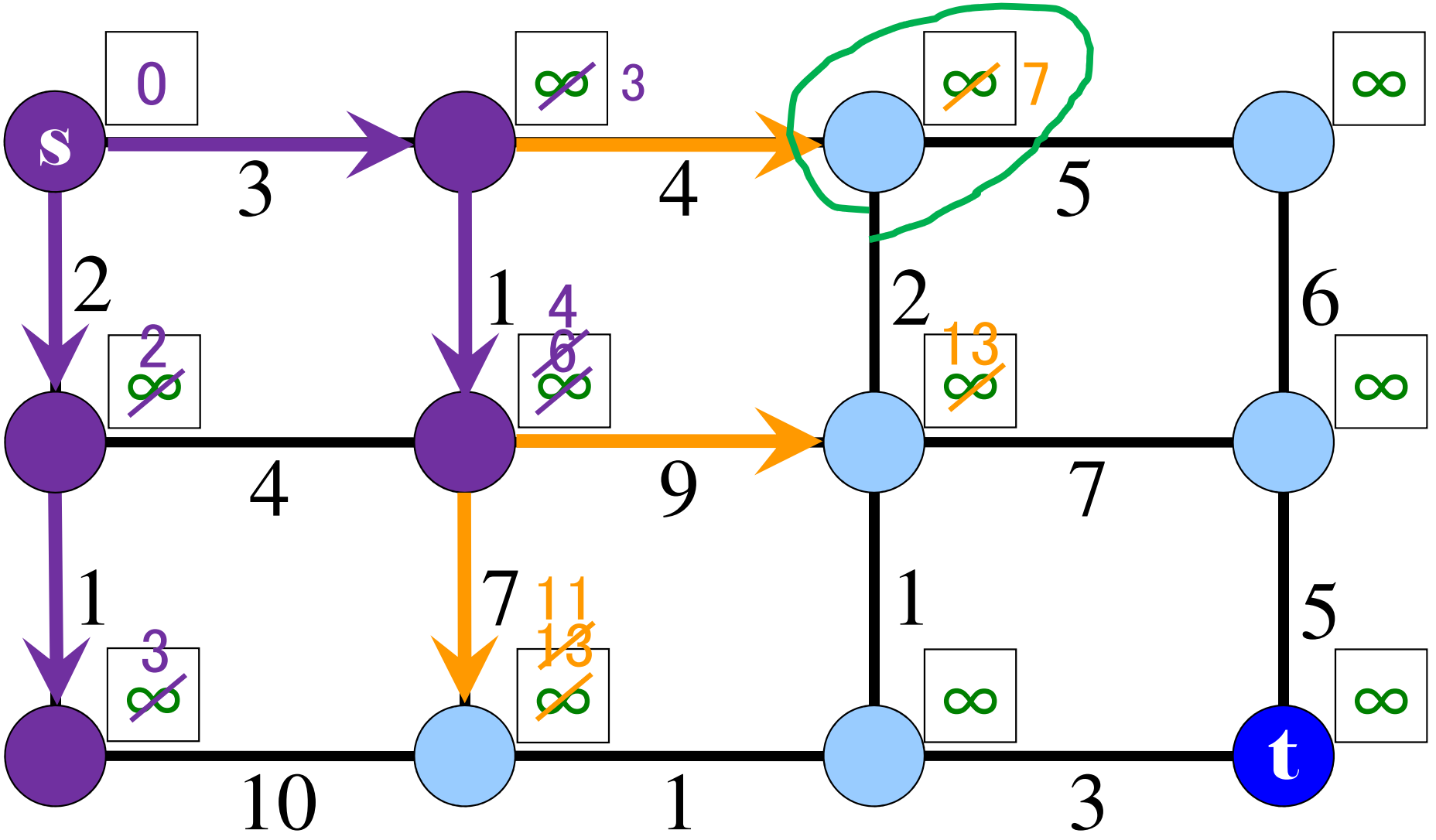
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



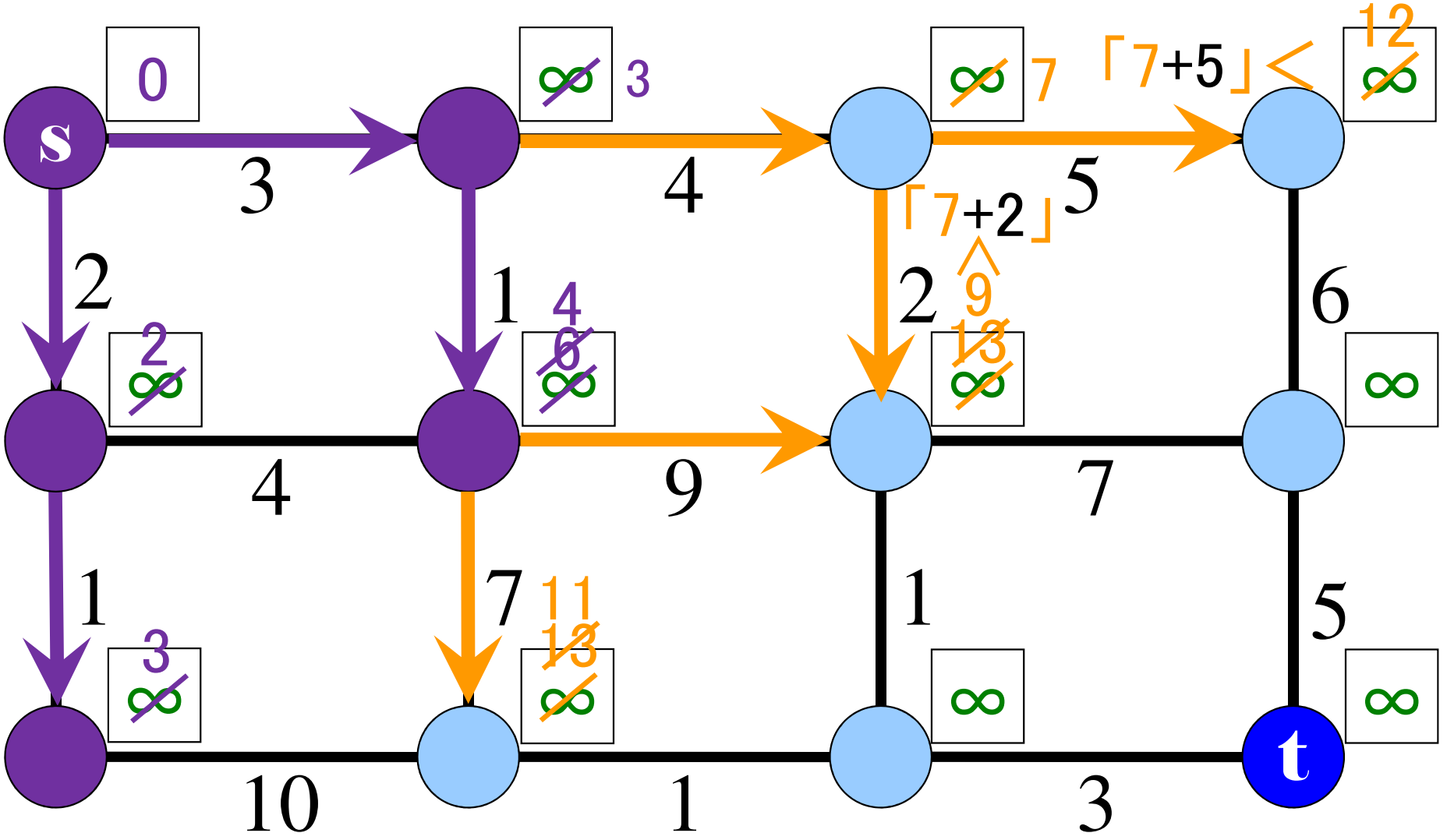
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

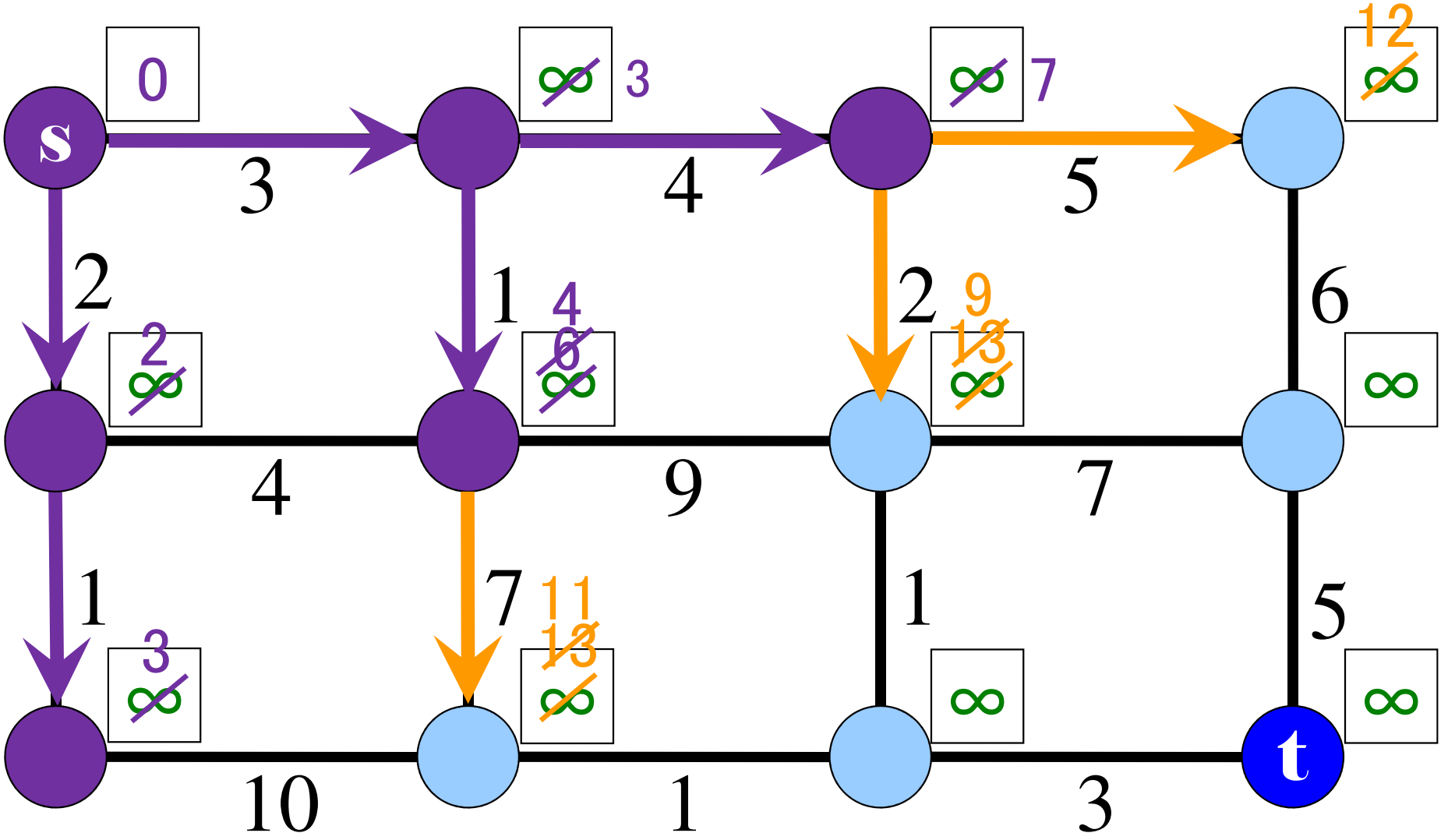


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

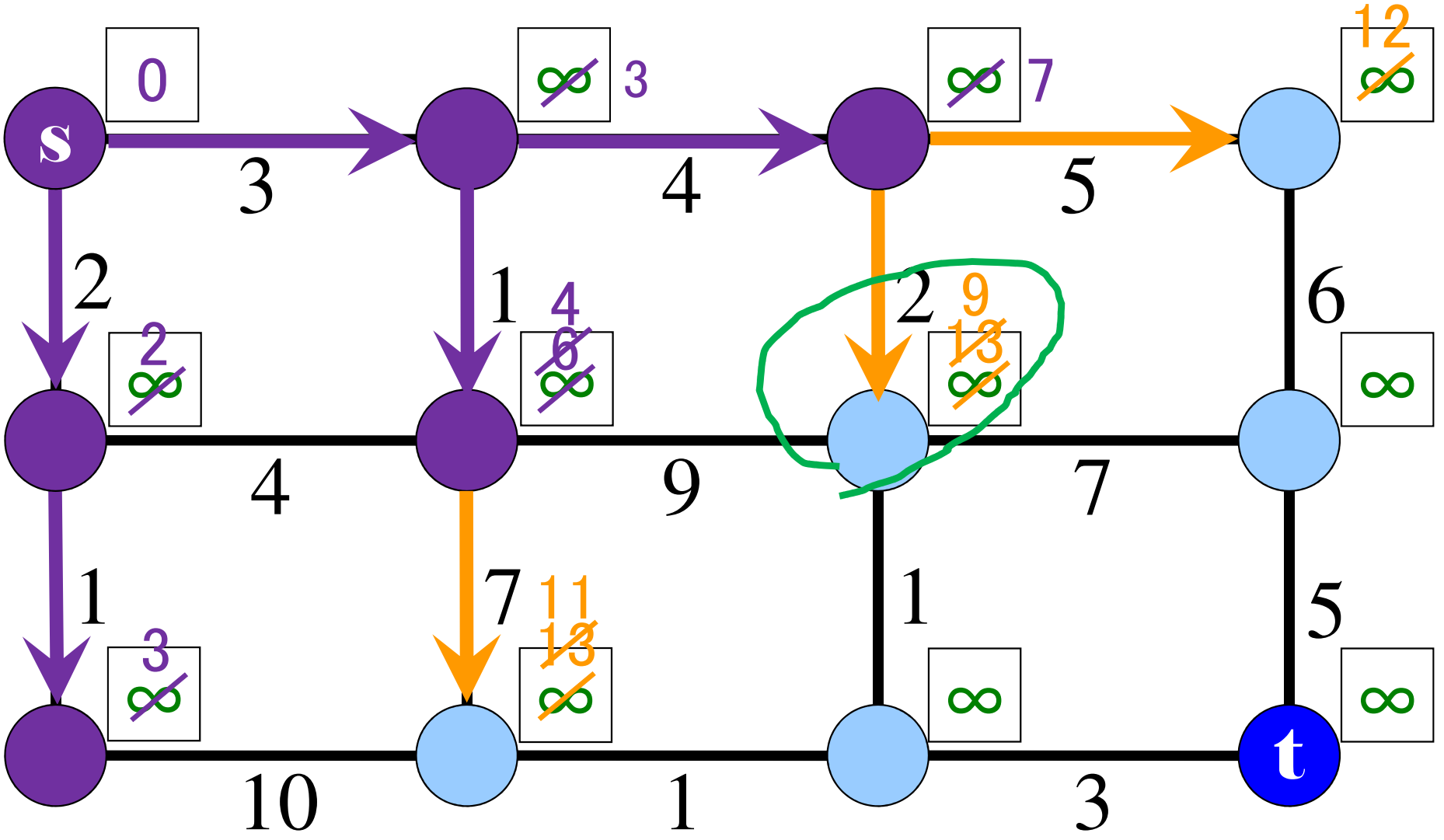
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



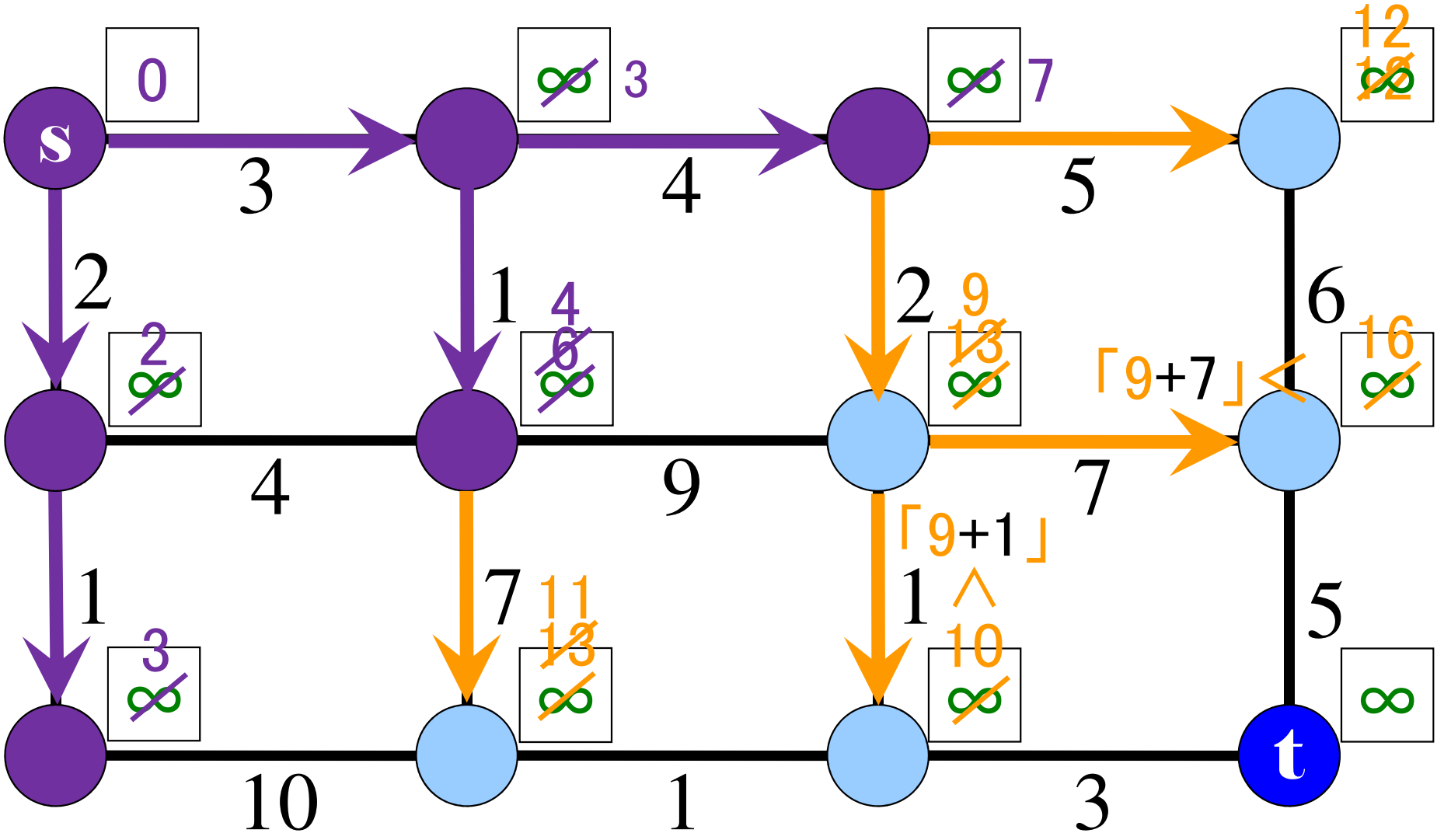
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

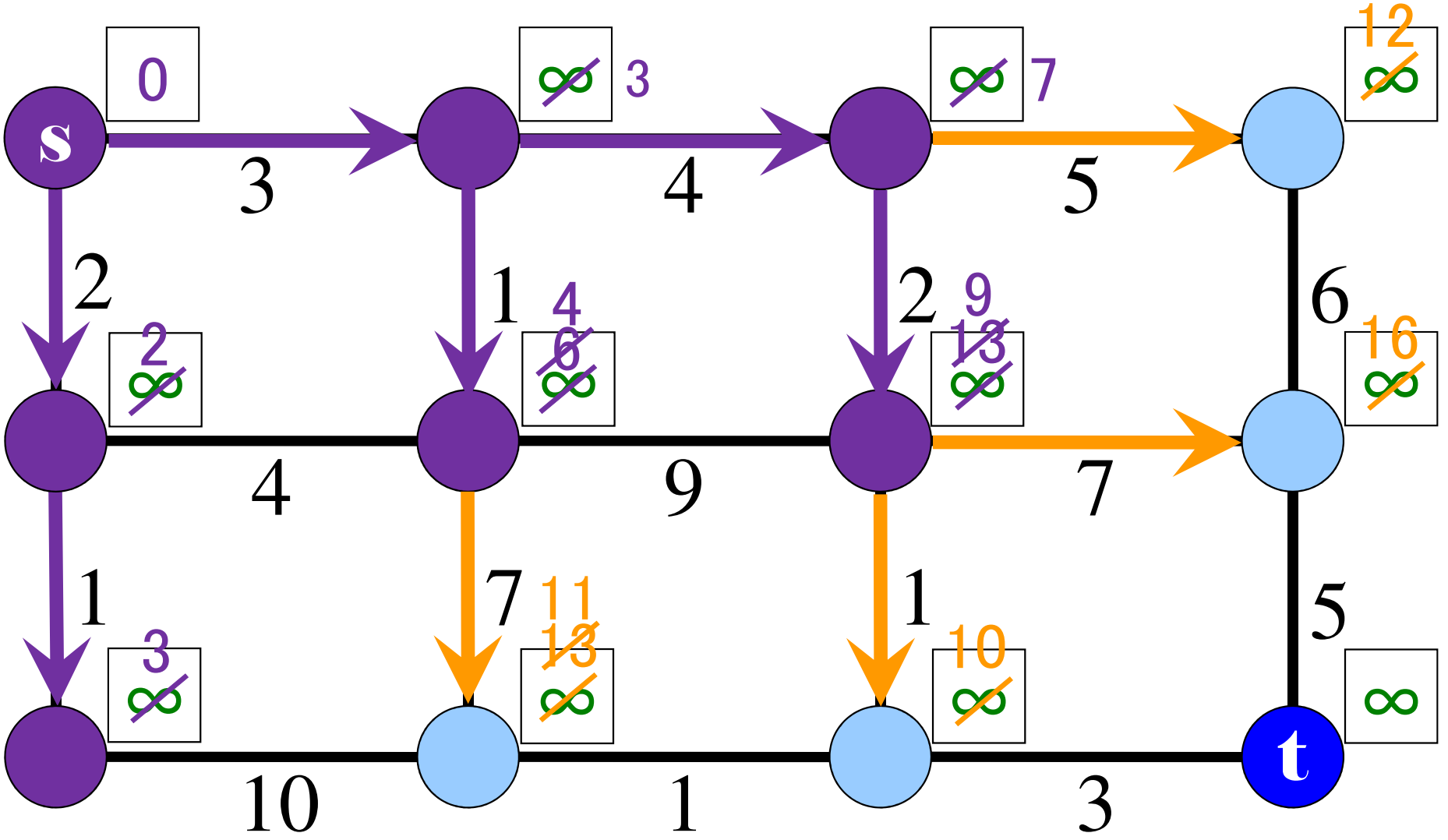


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

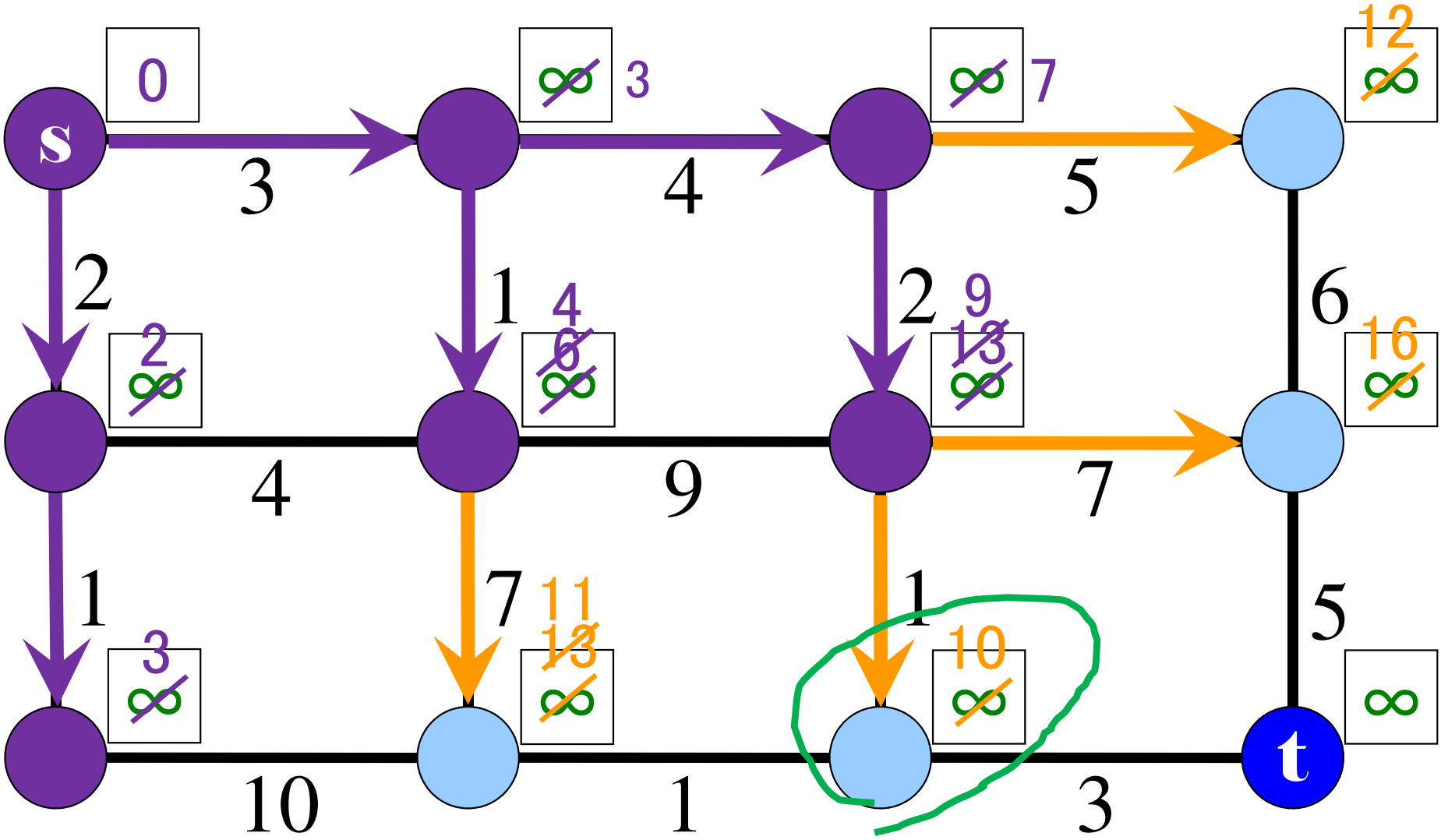
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



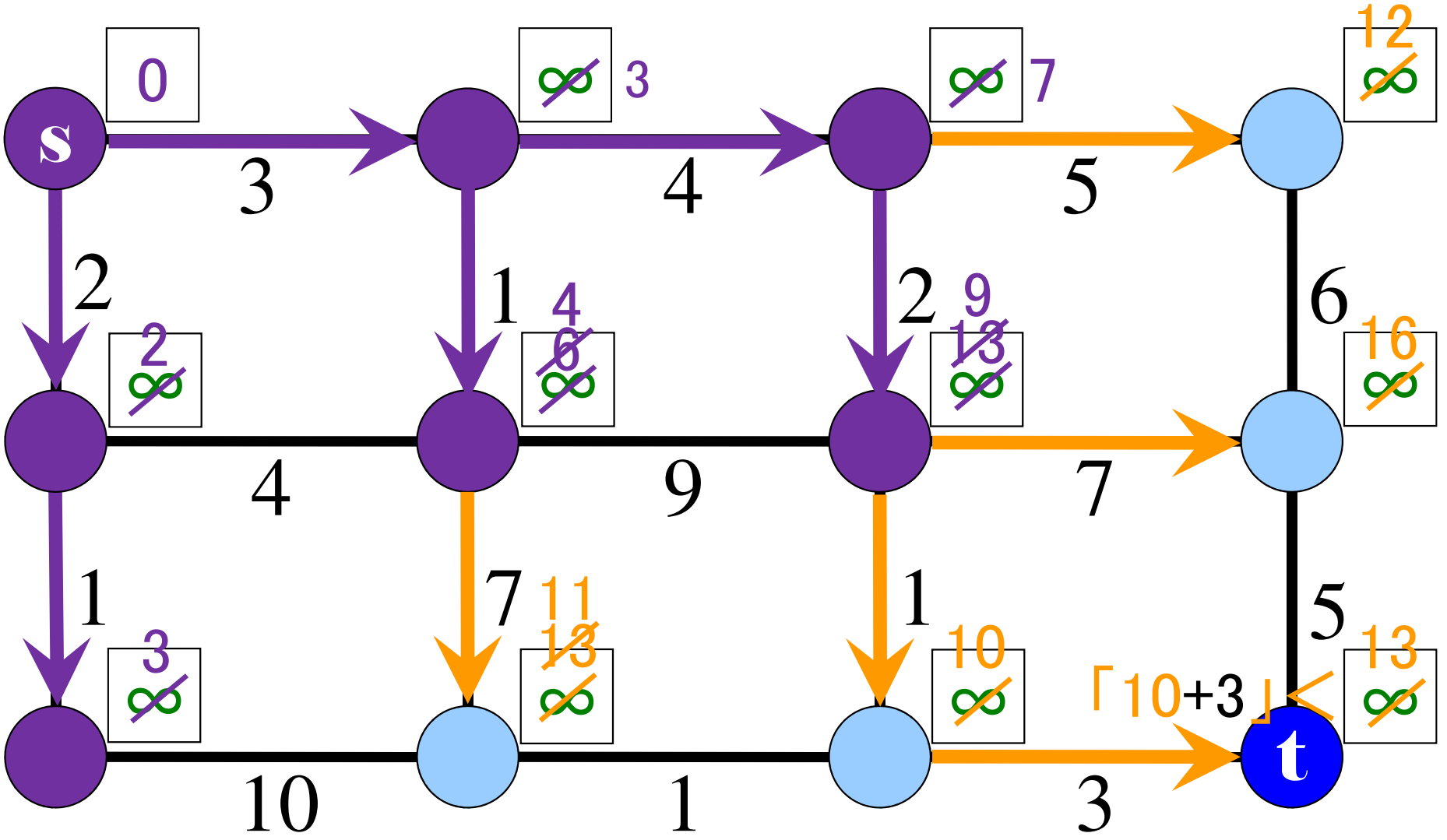
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

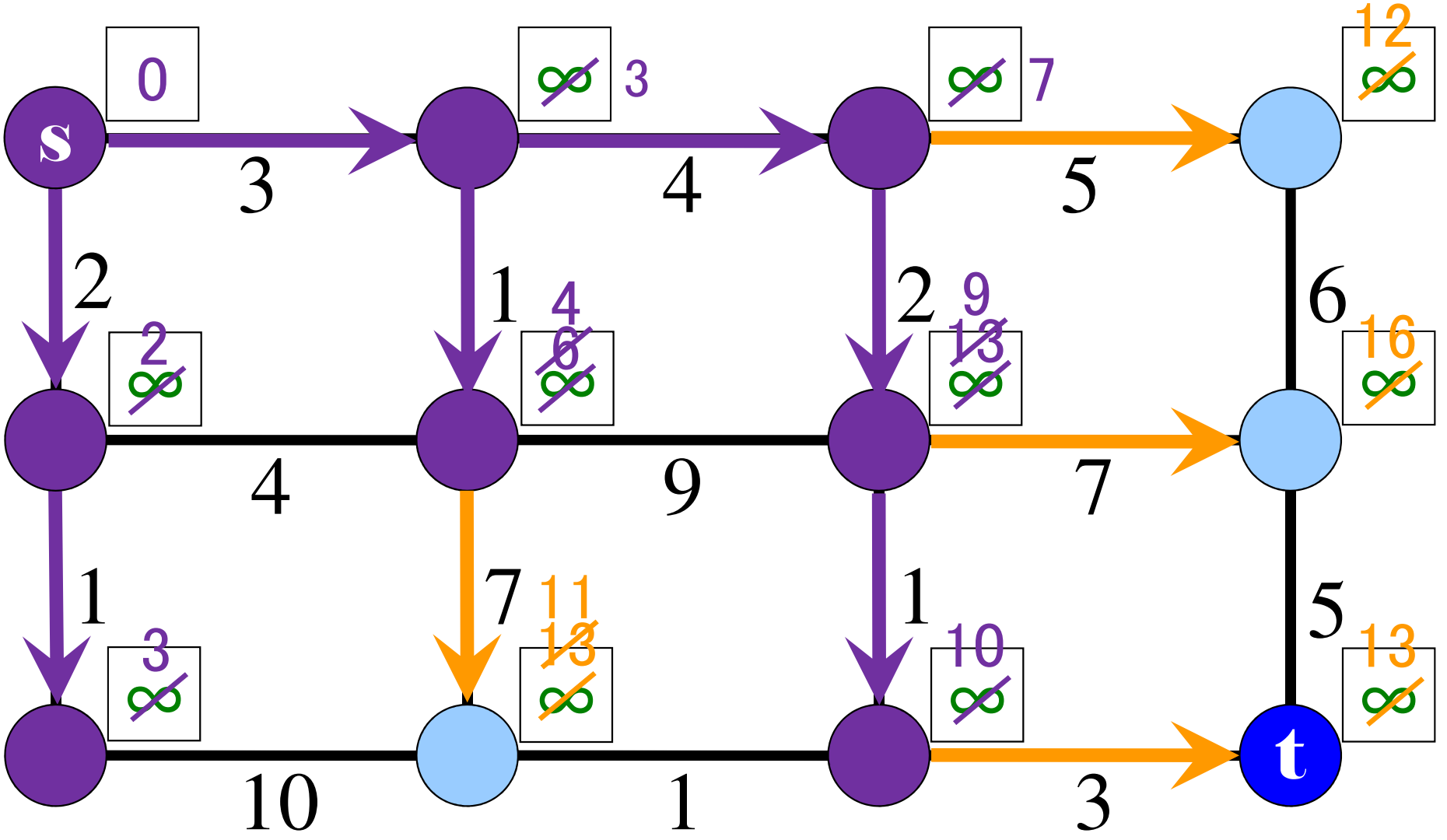


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

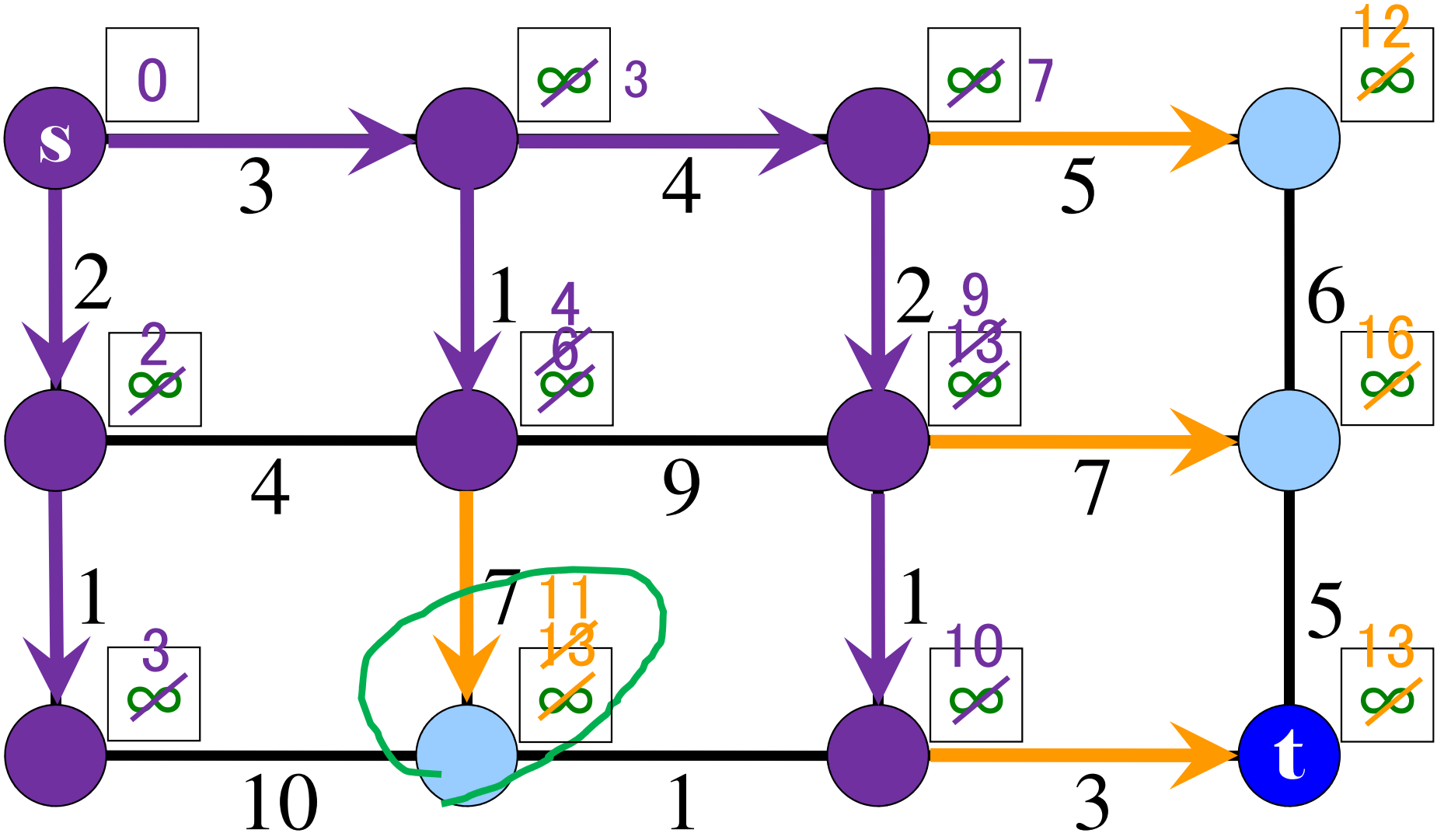
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



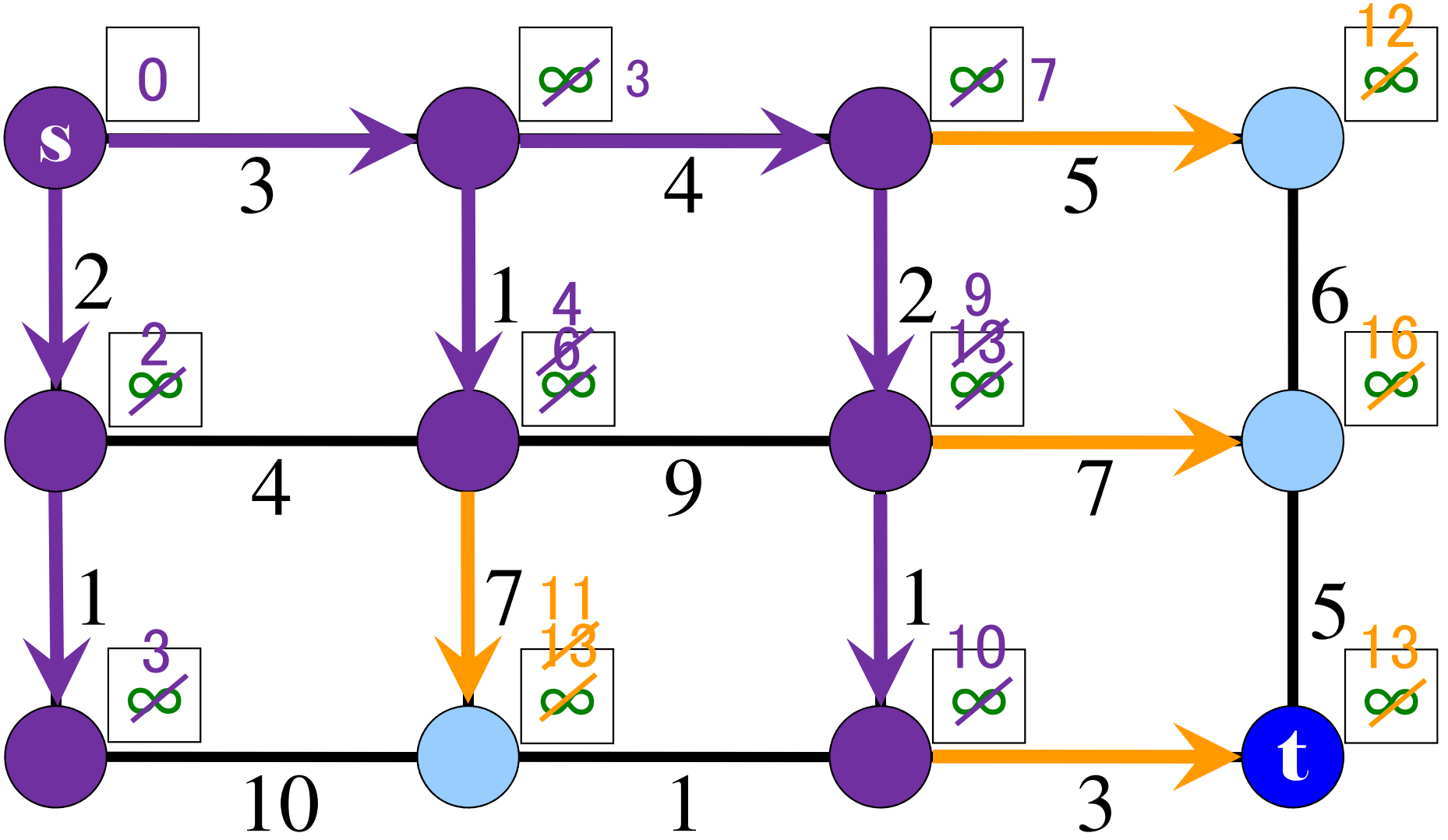
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

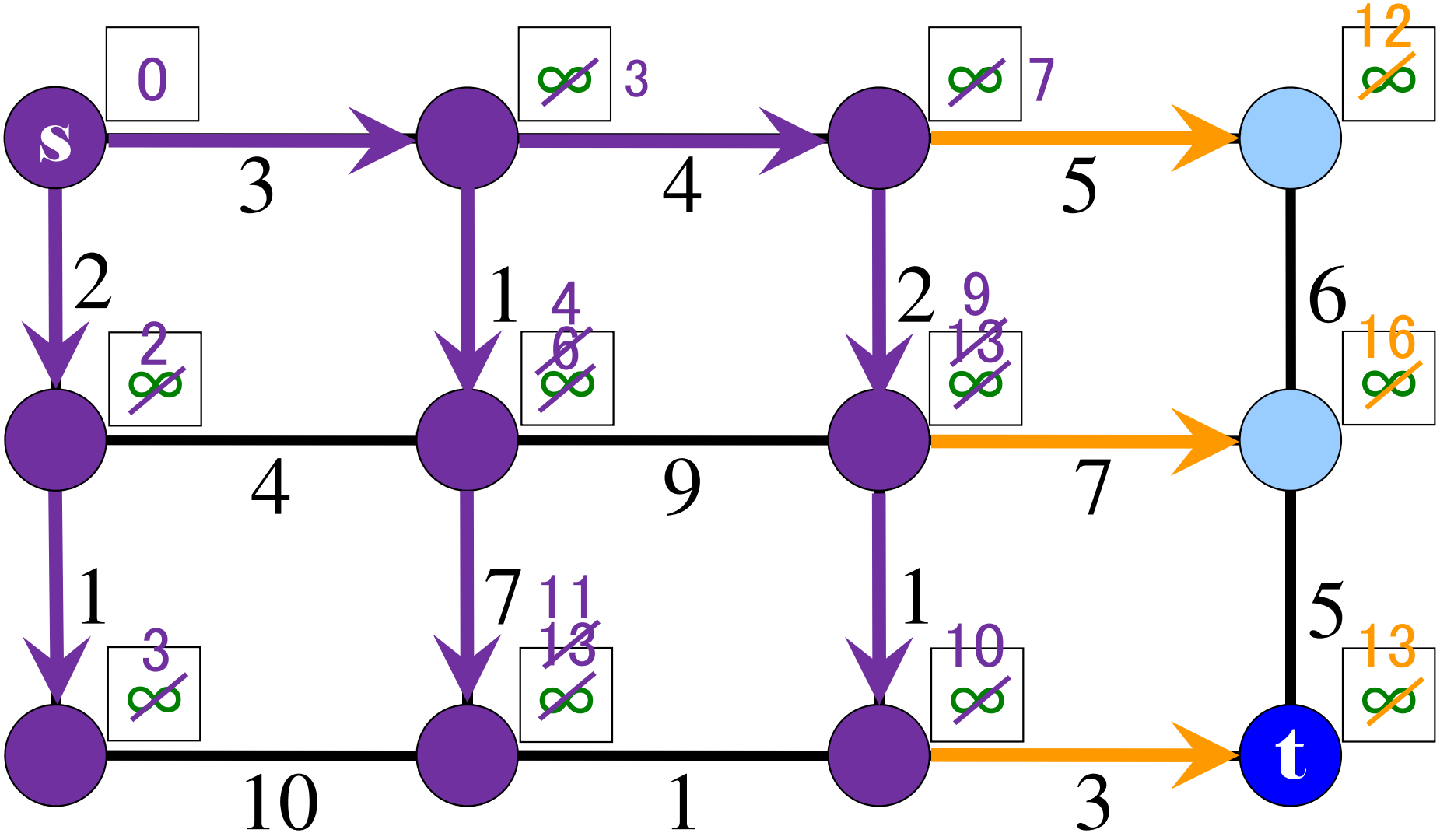


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

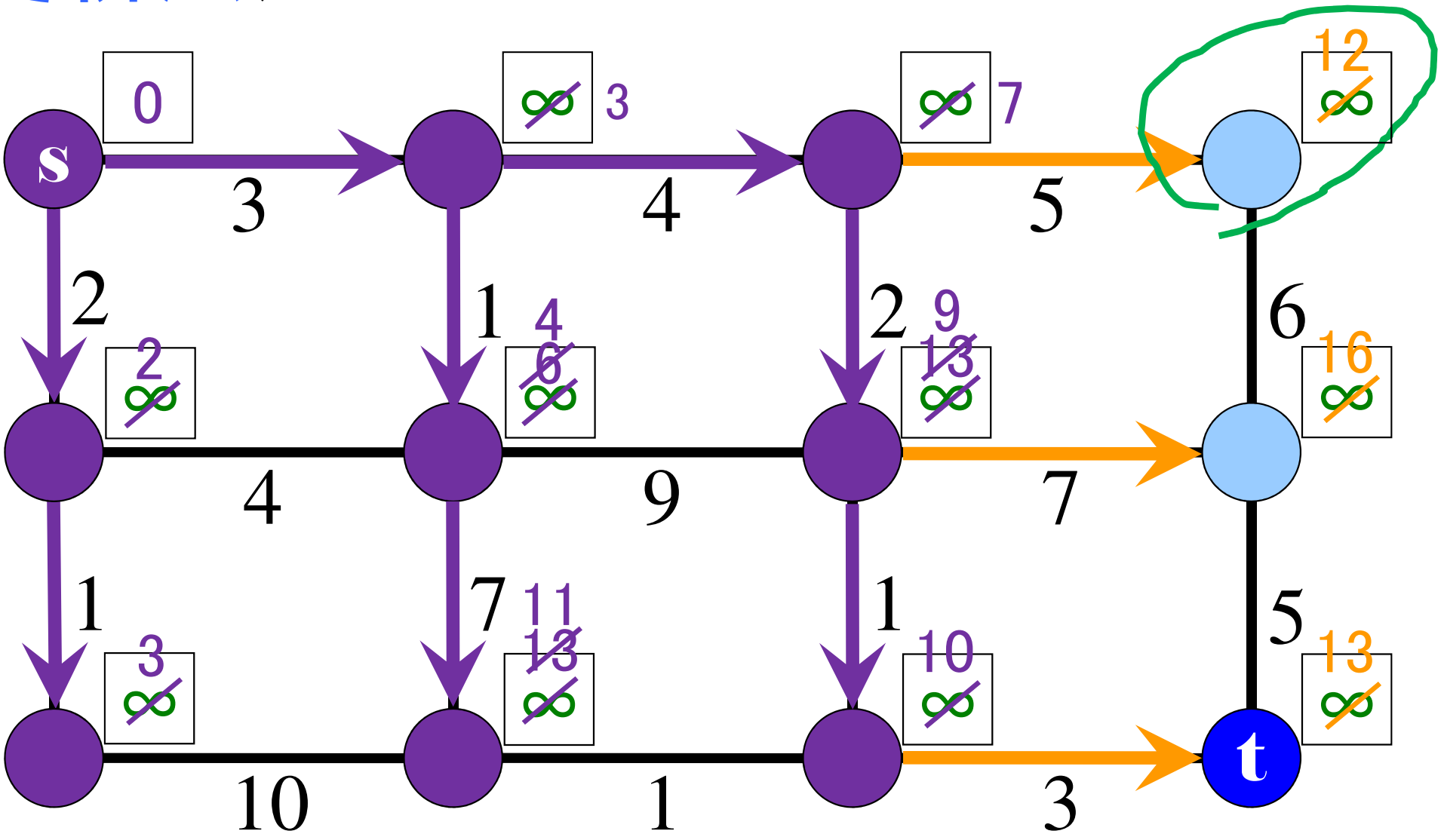
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



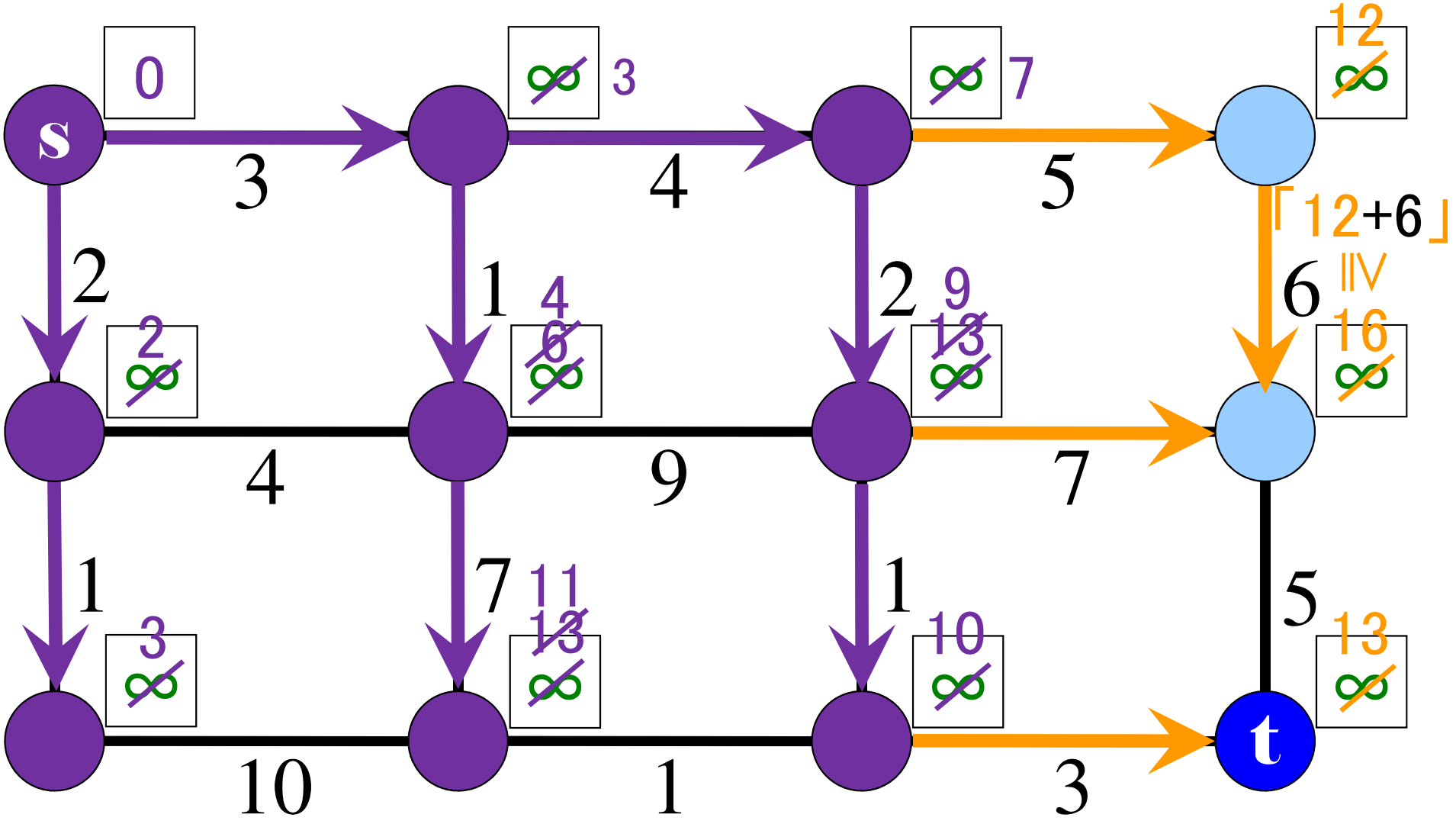
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

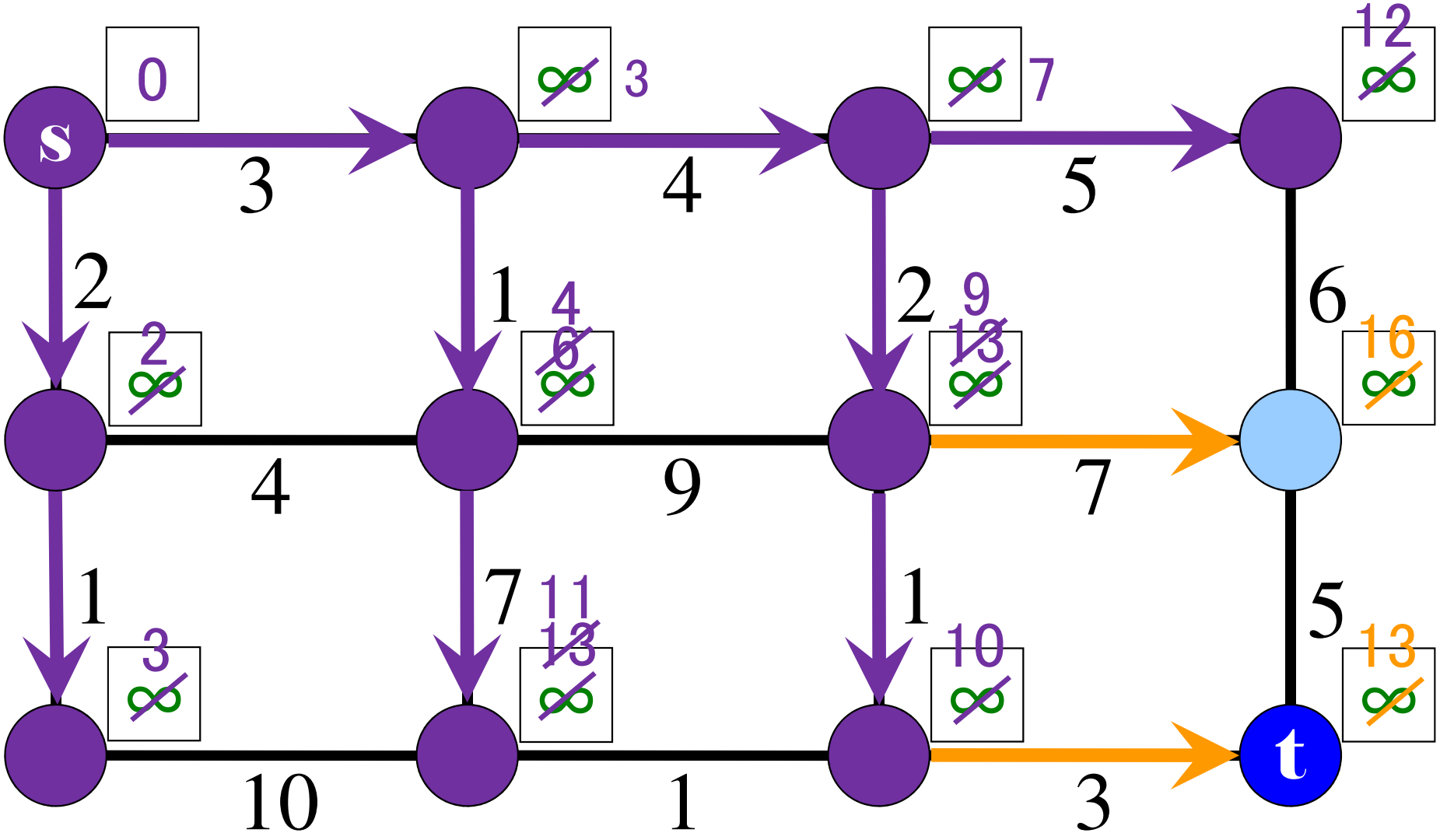


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

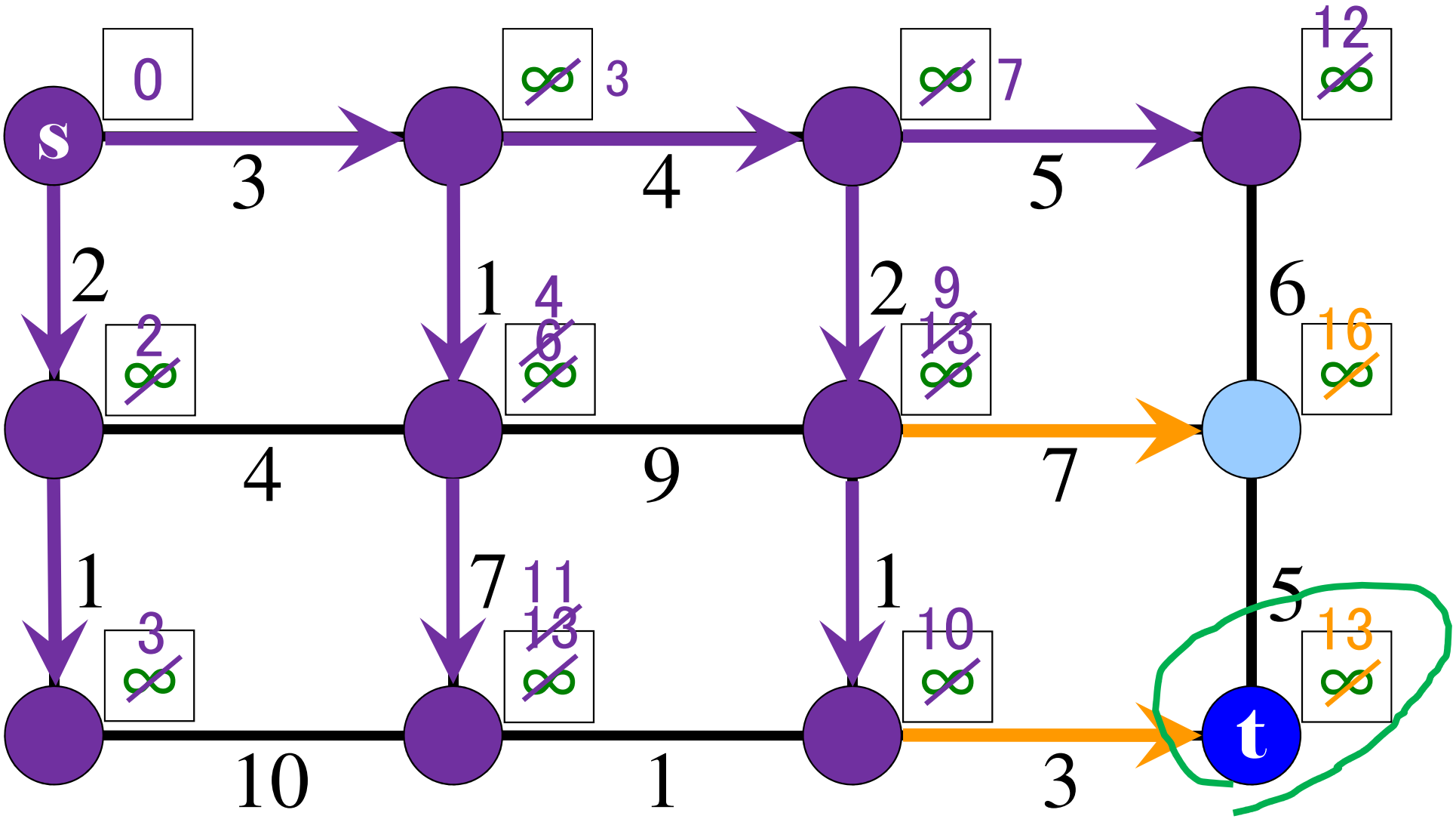
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



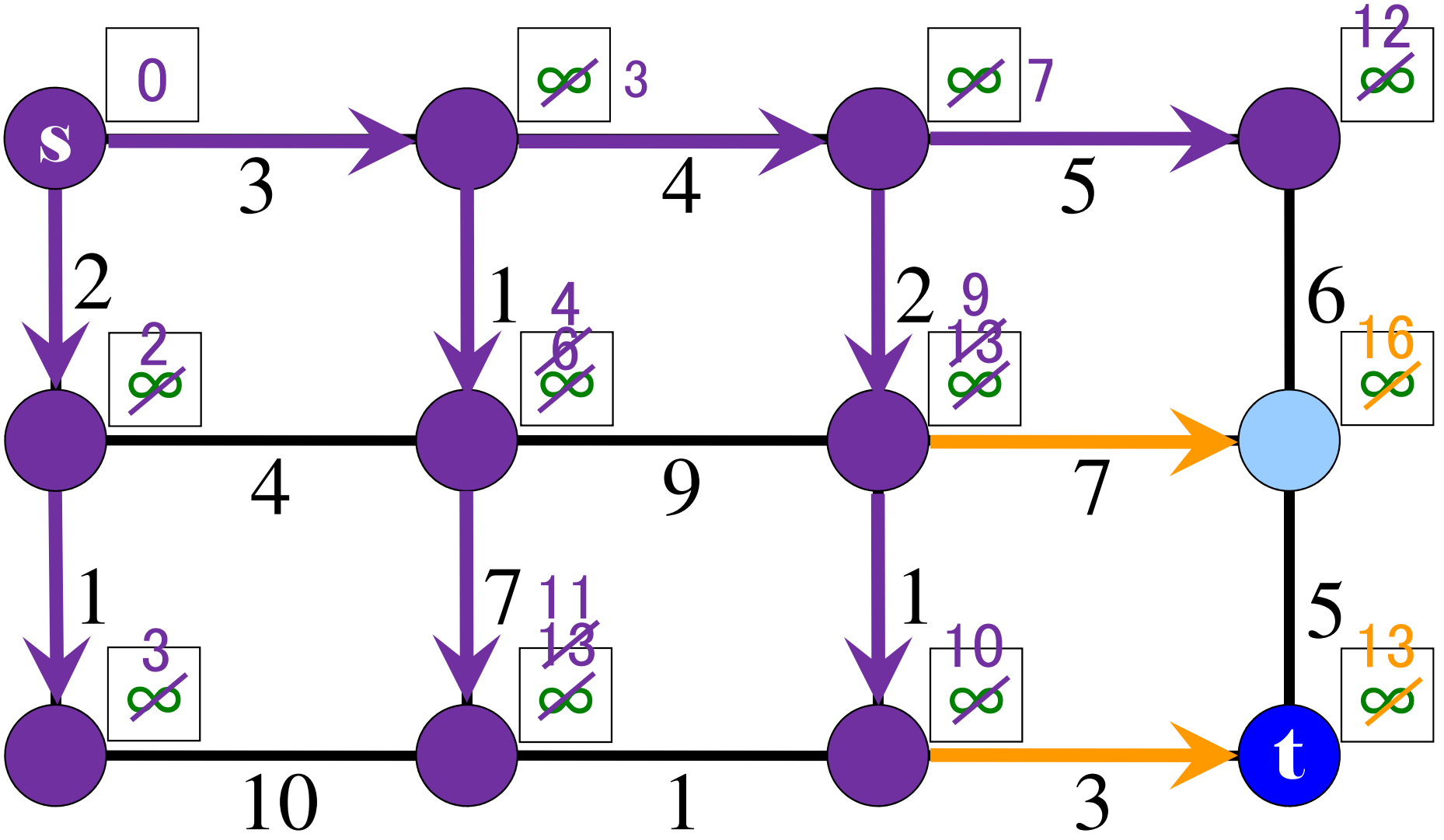
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

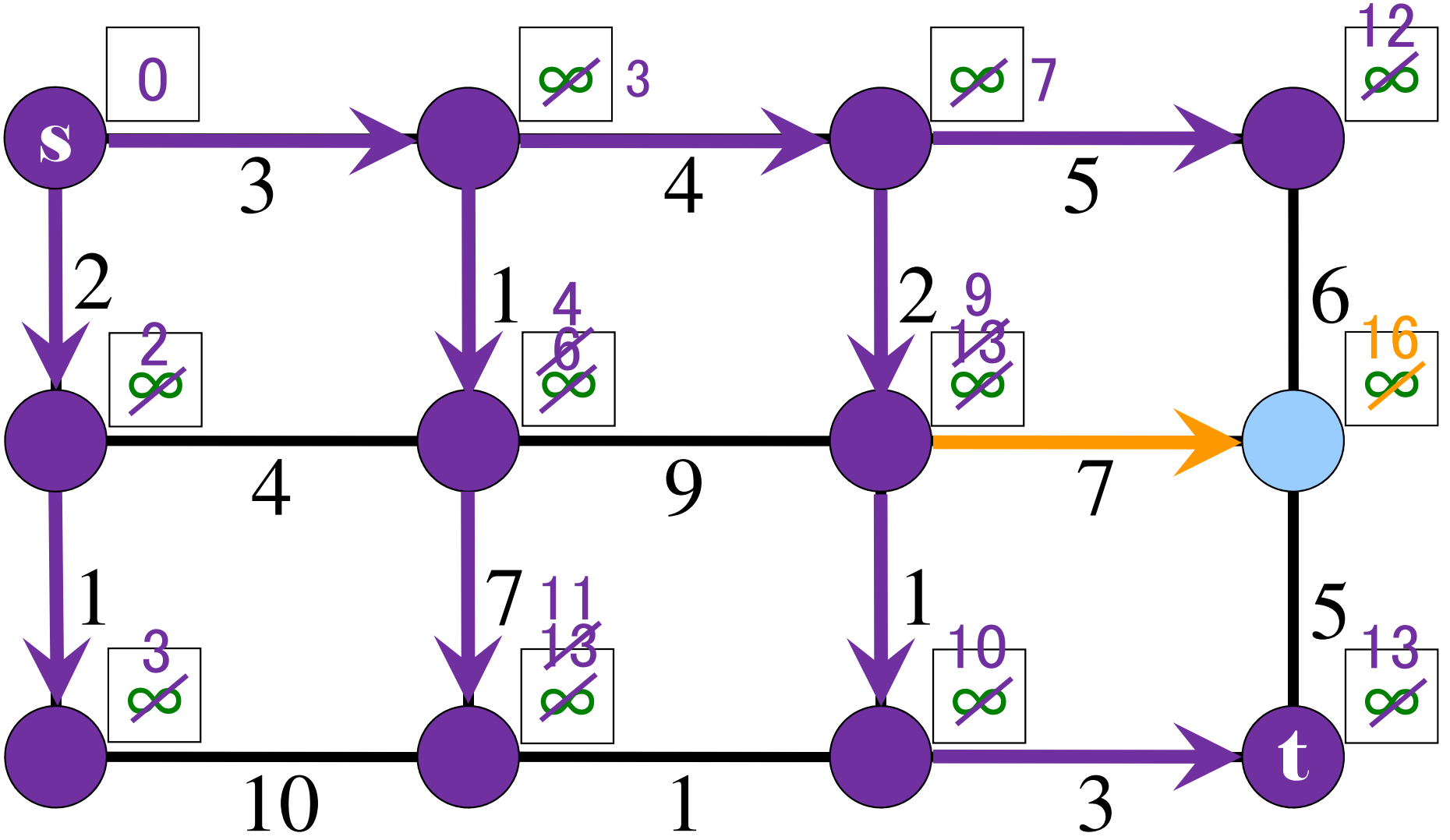


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

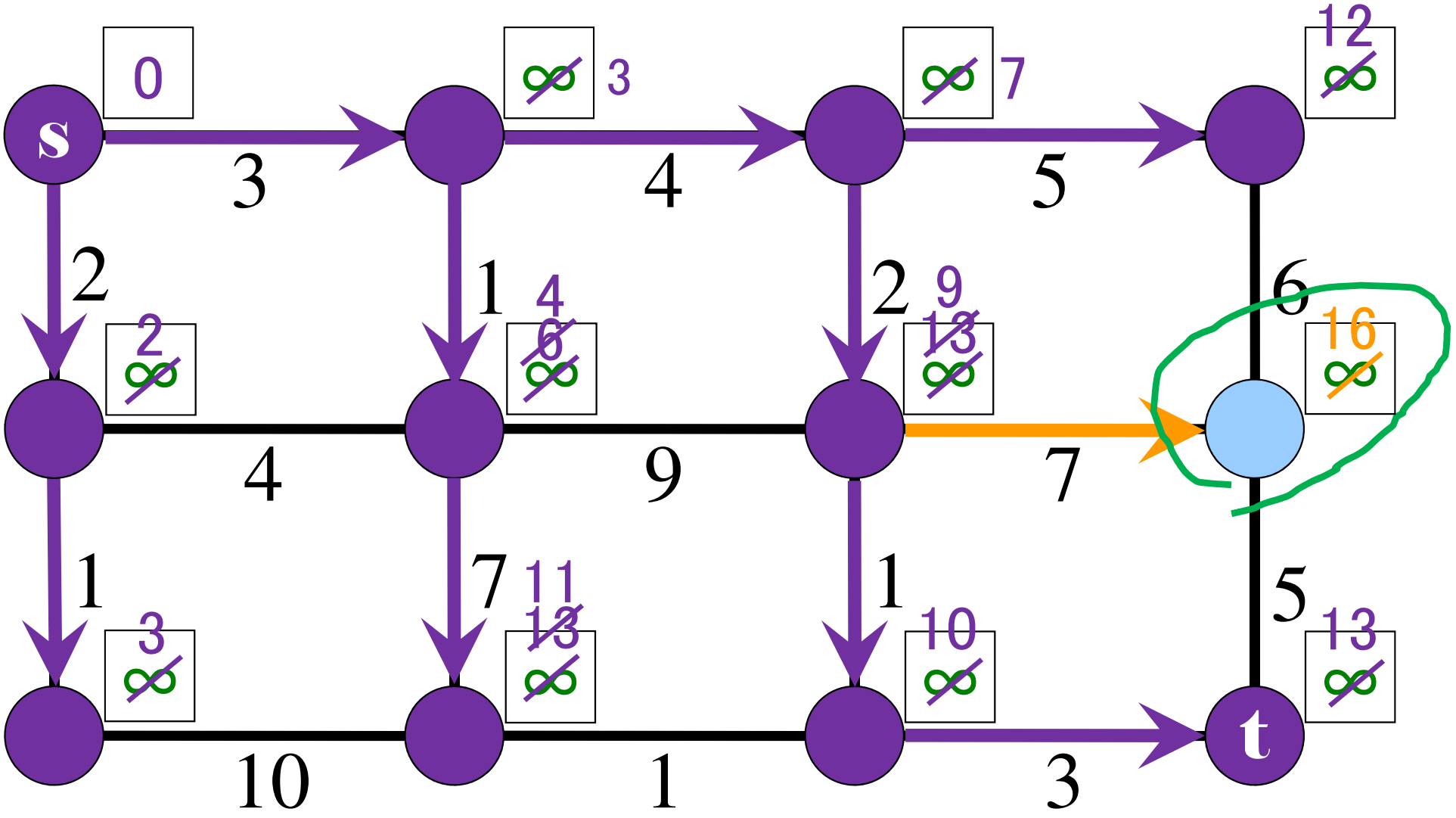
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



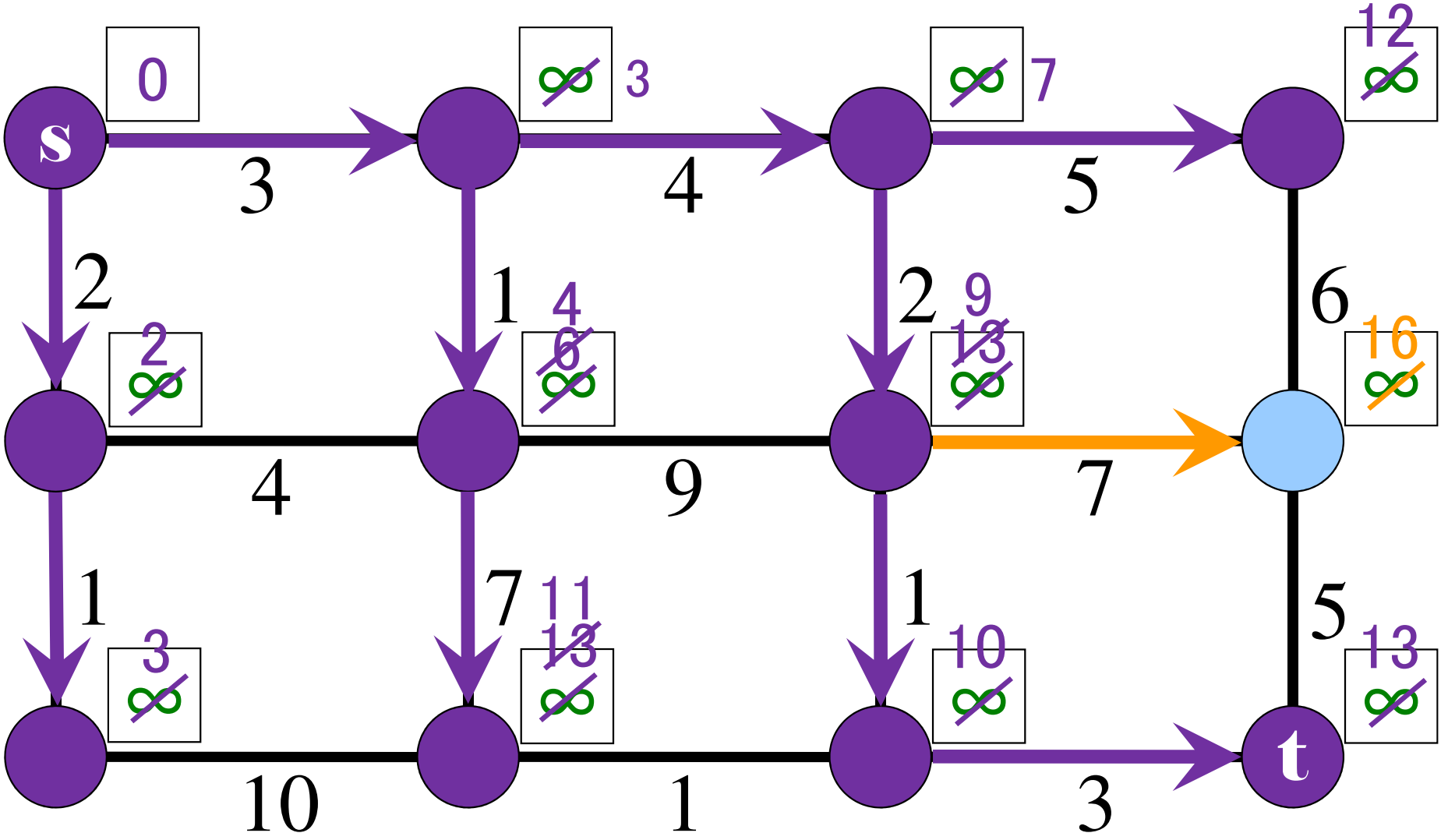
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

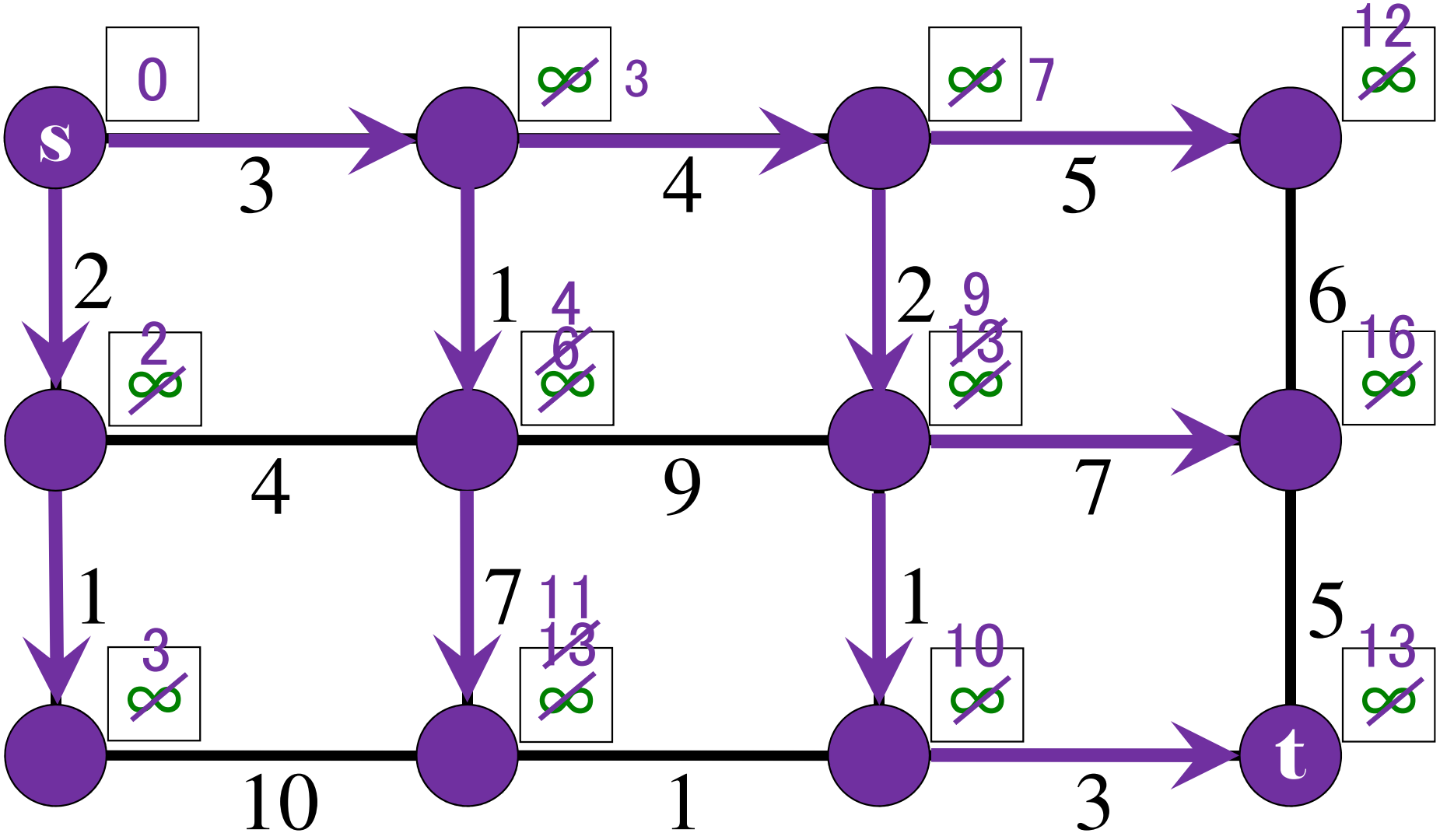


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

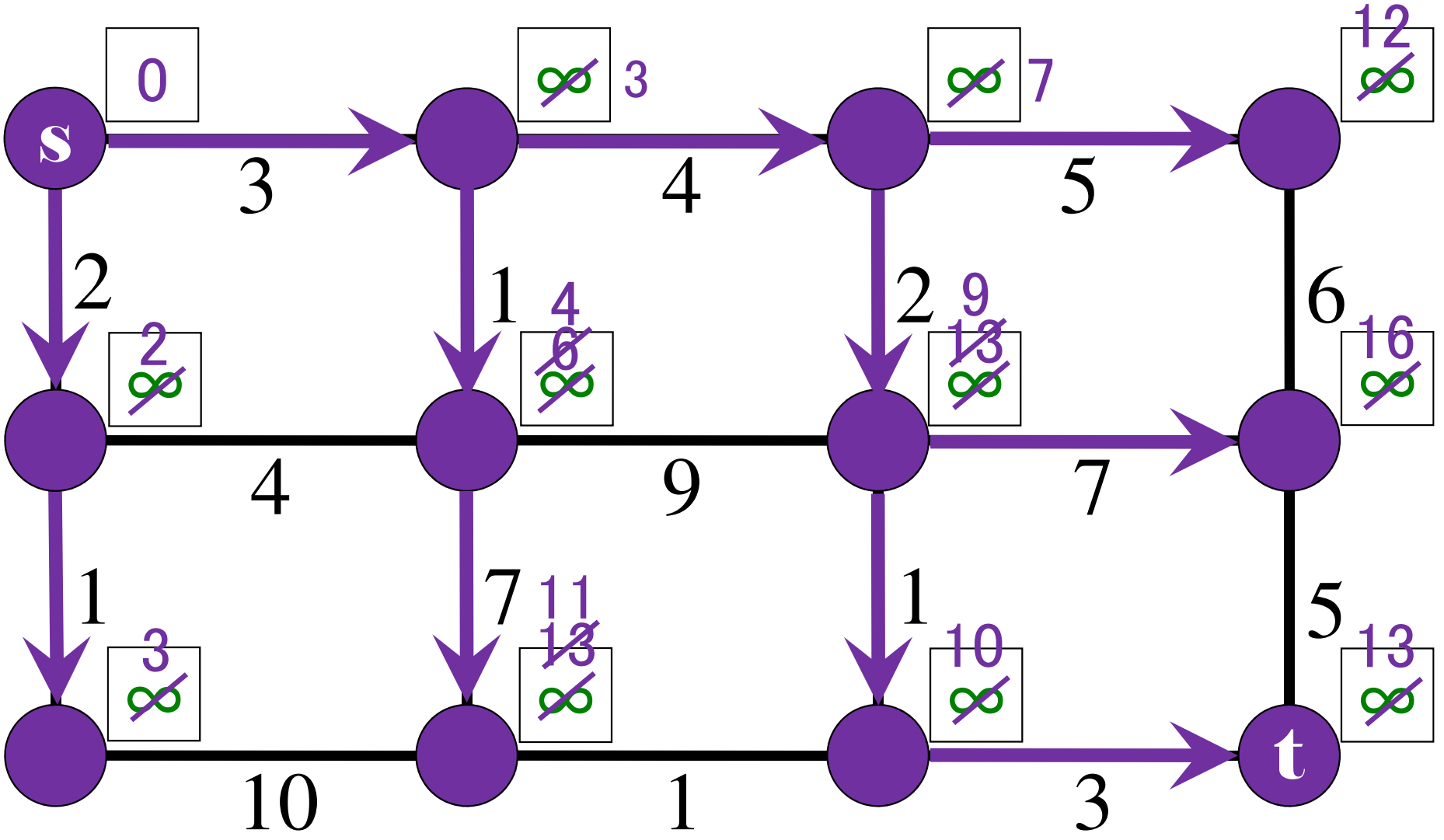
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



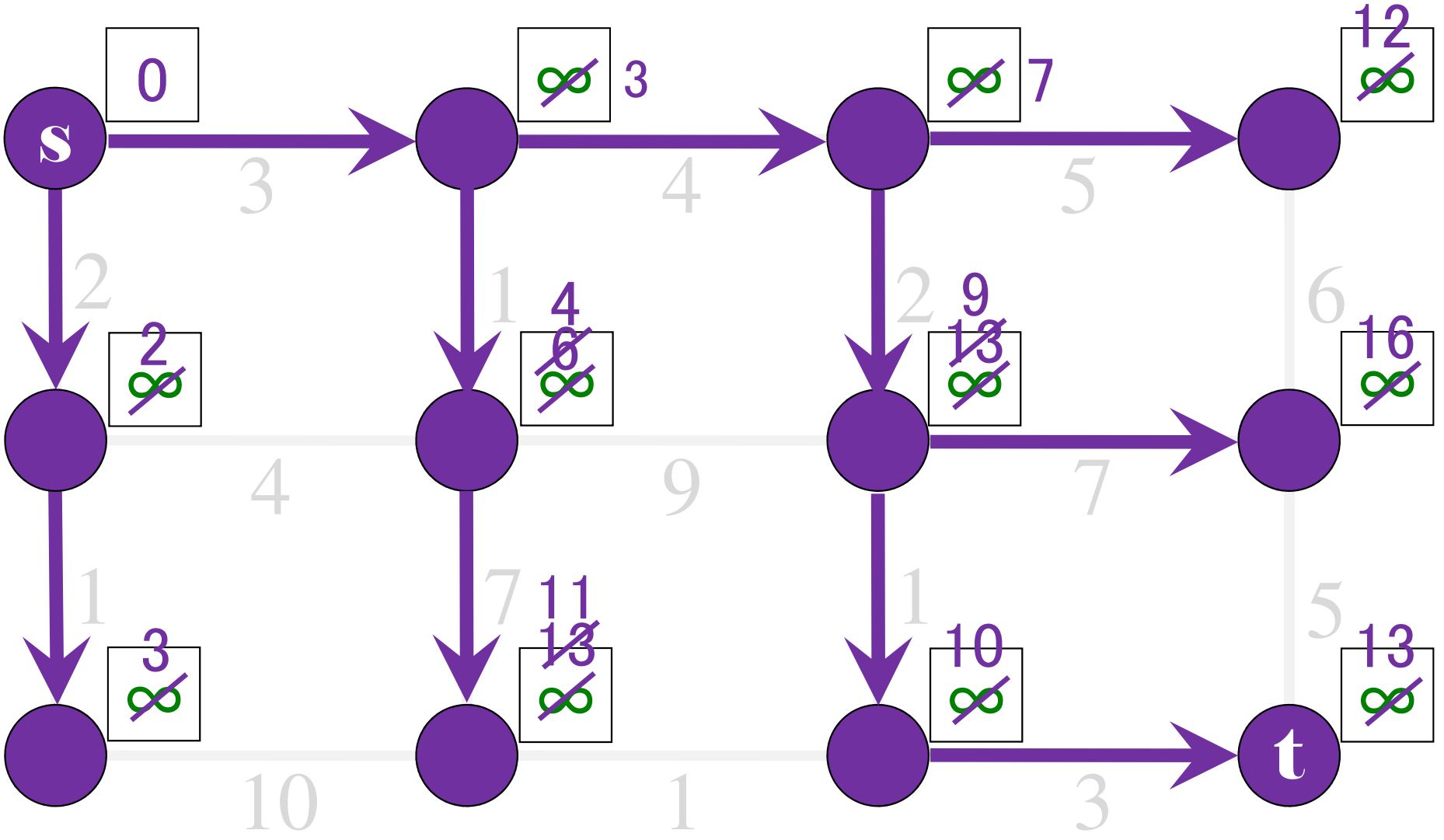
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



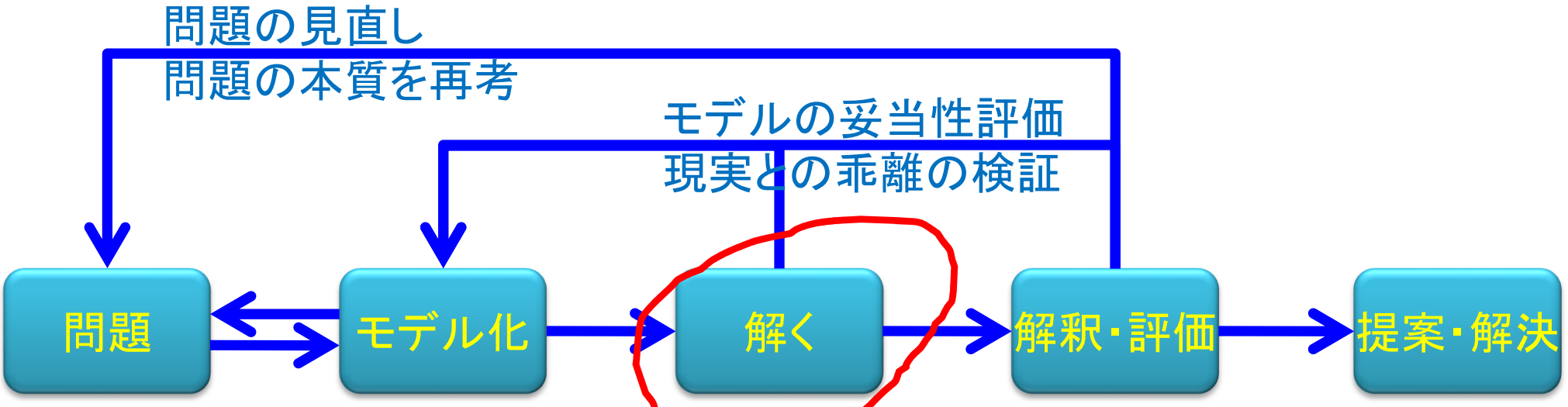
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

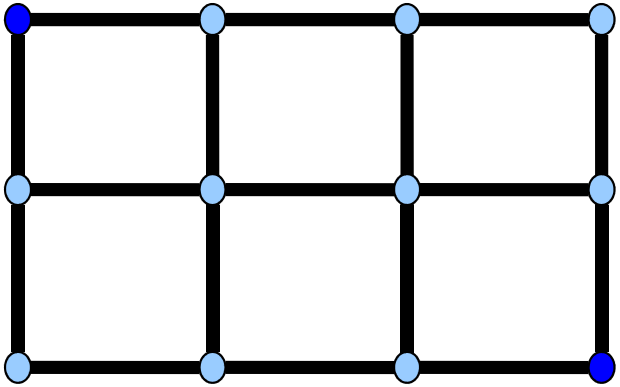
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

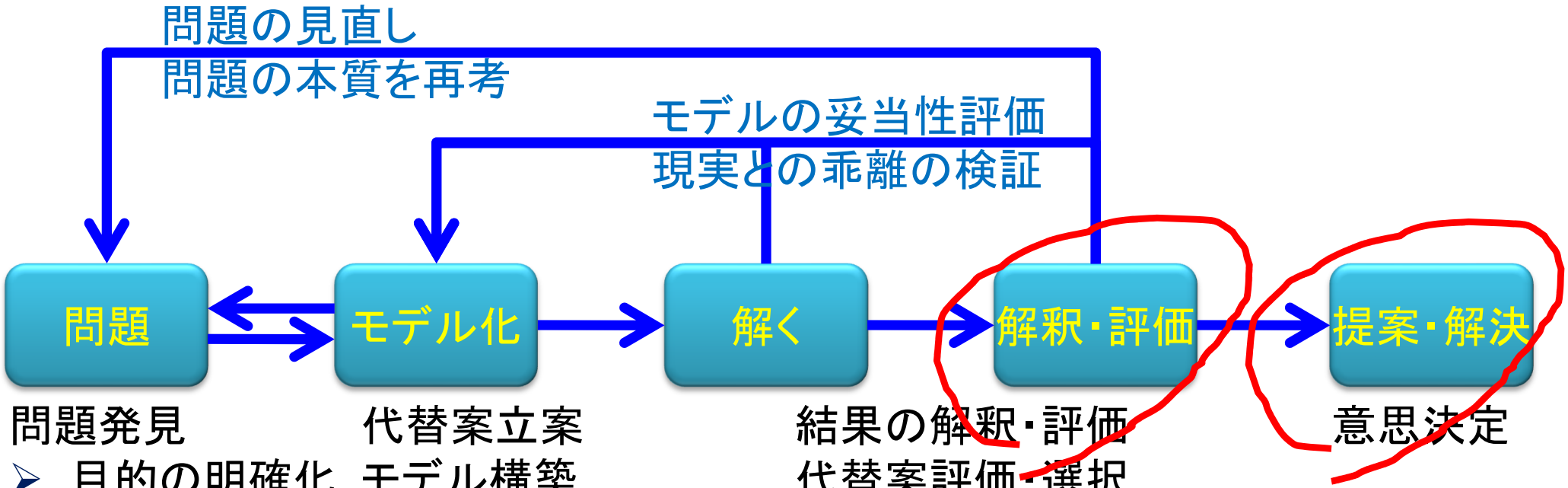
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

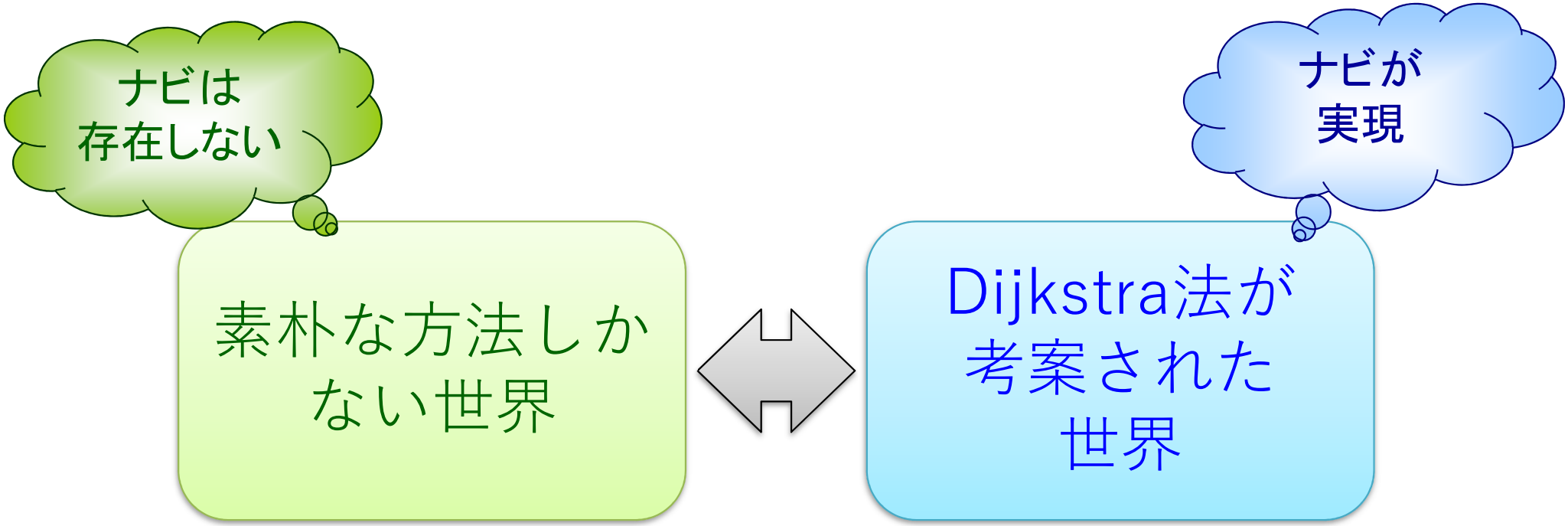
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

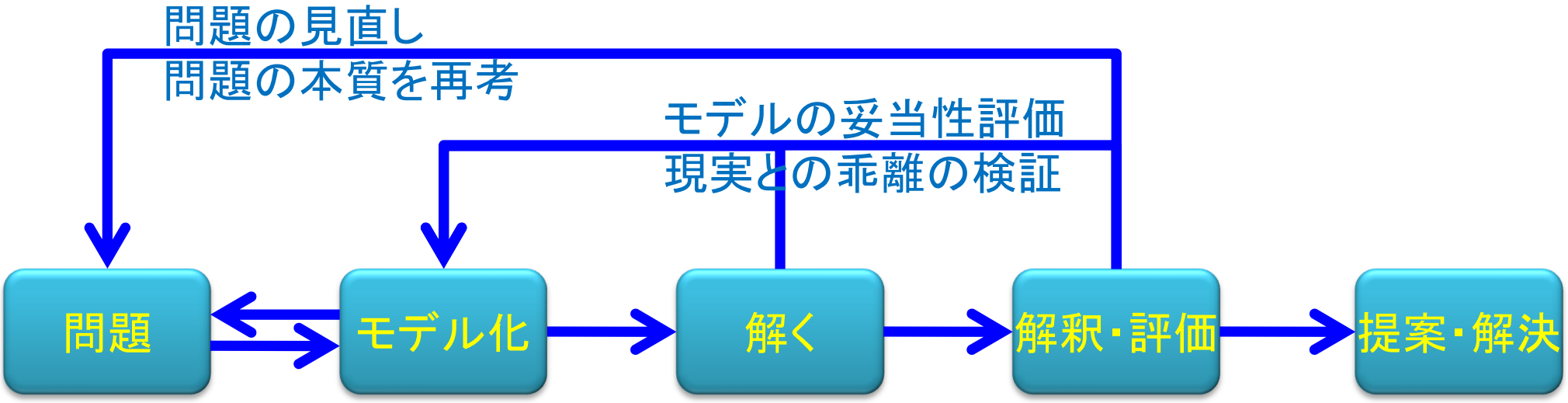
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



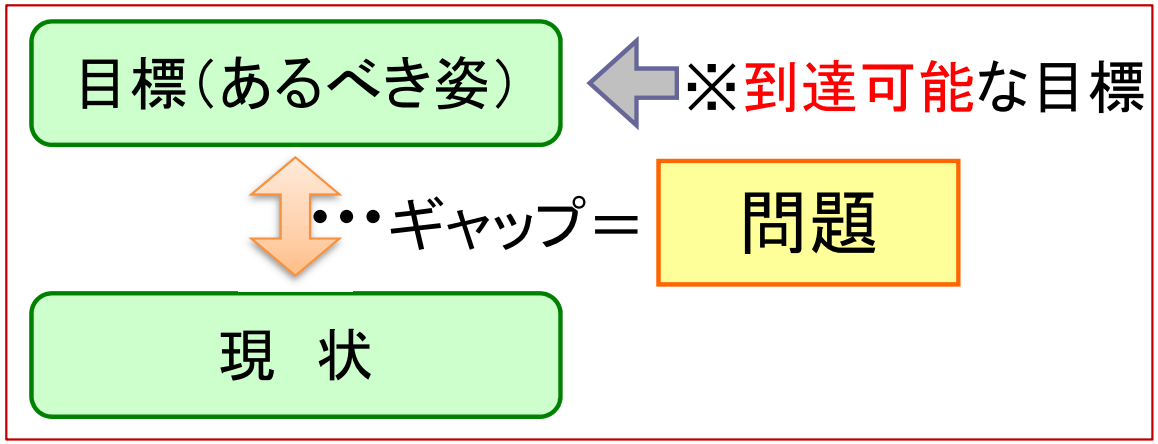
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

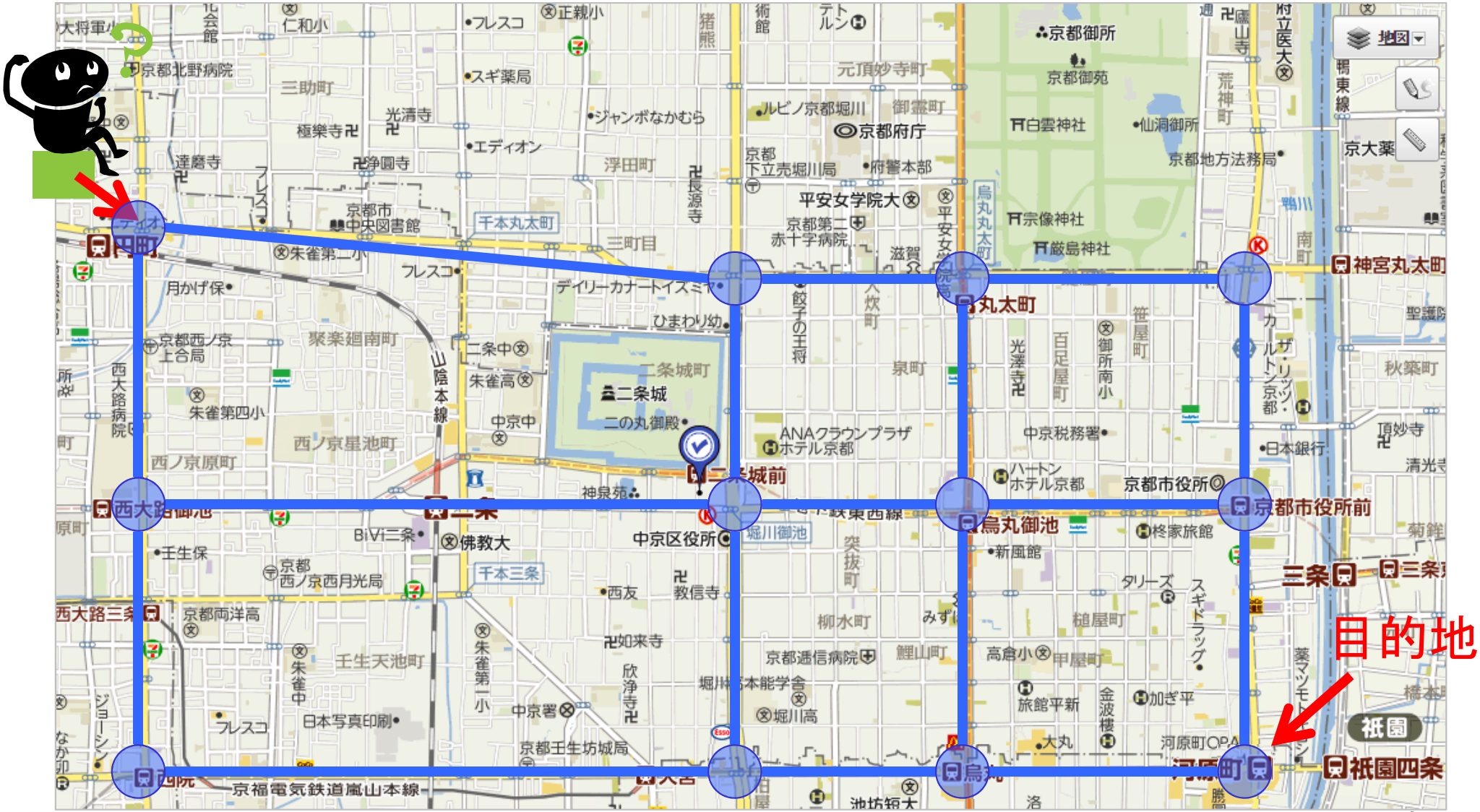
意思決定

問題の定義



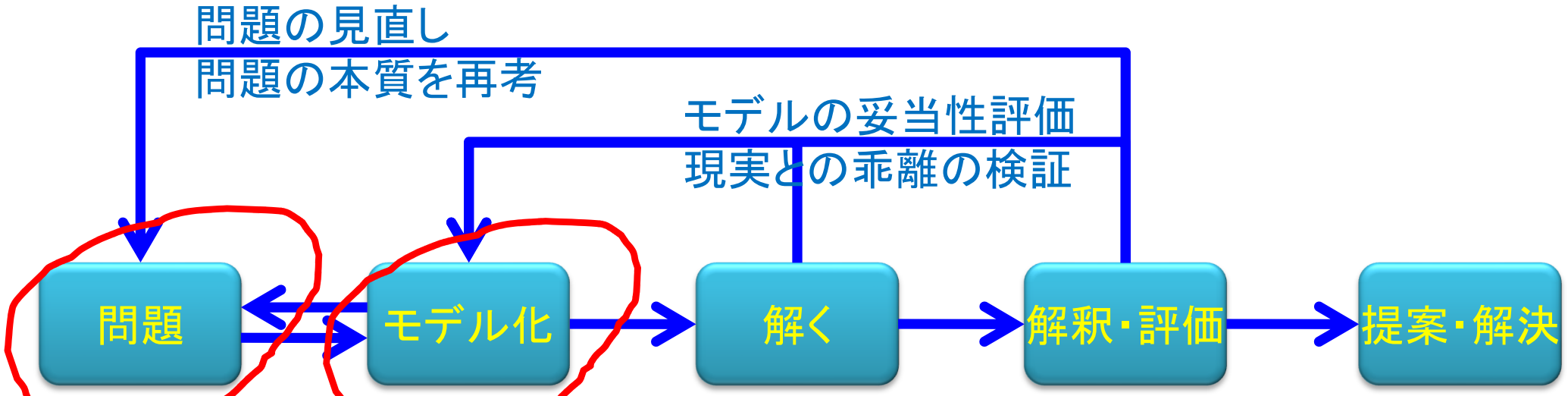
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

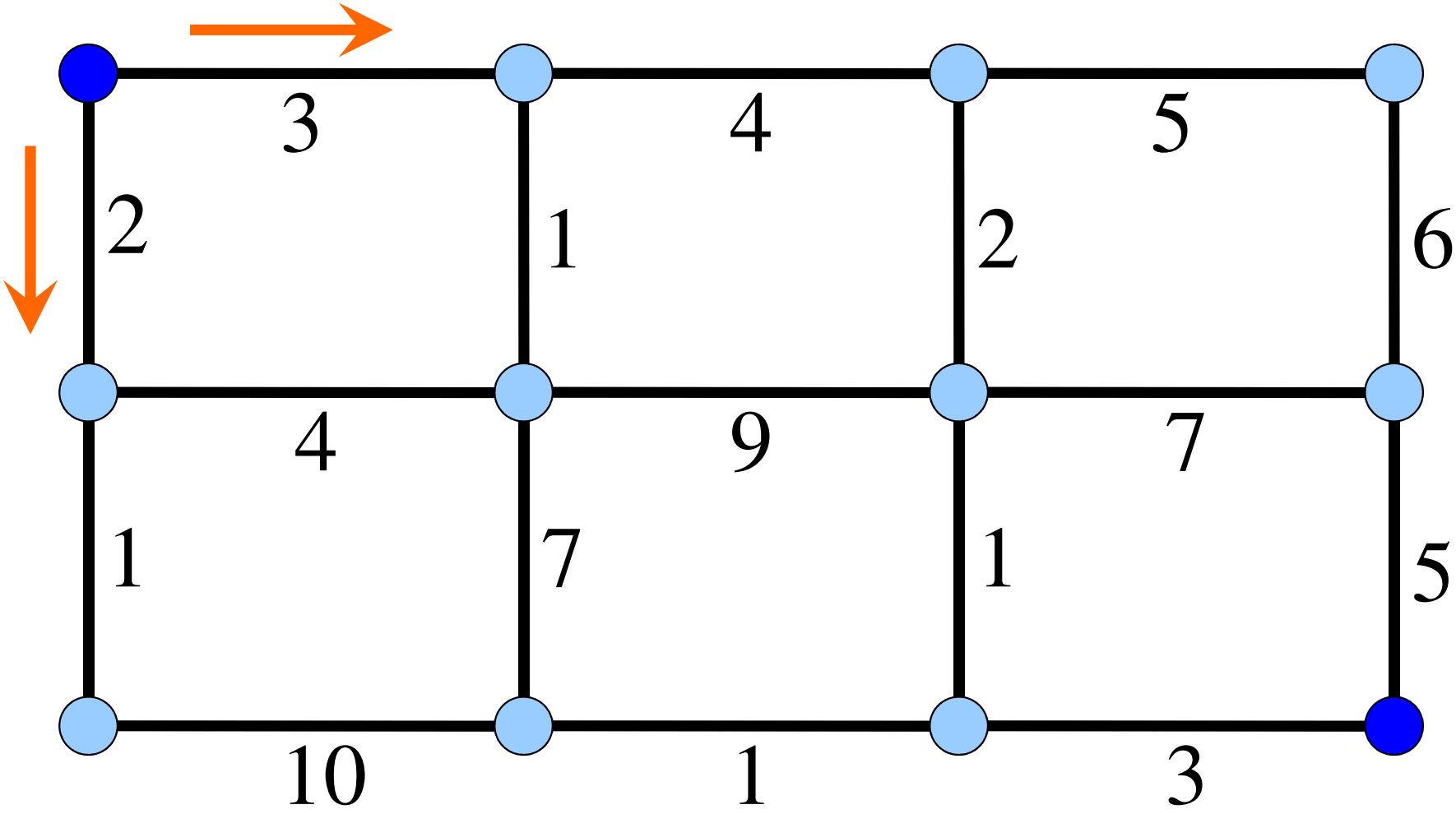
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

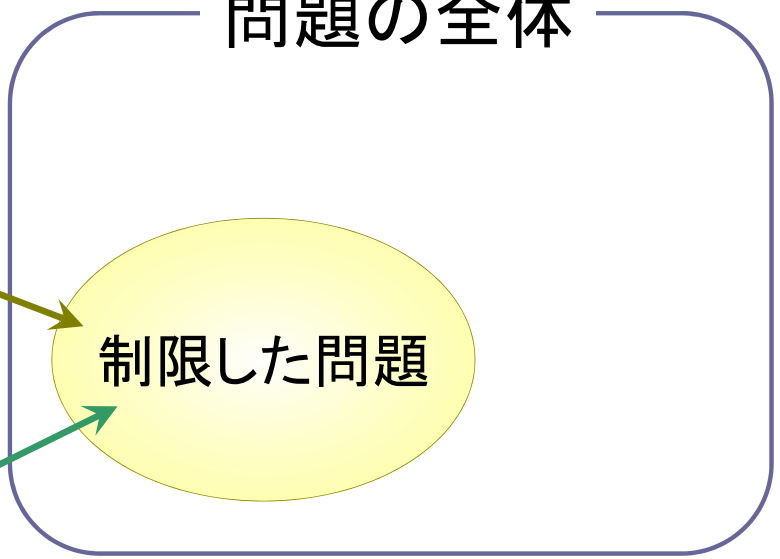


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

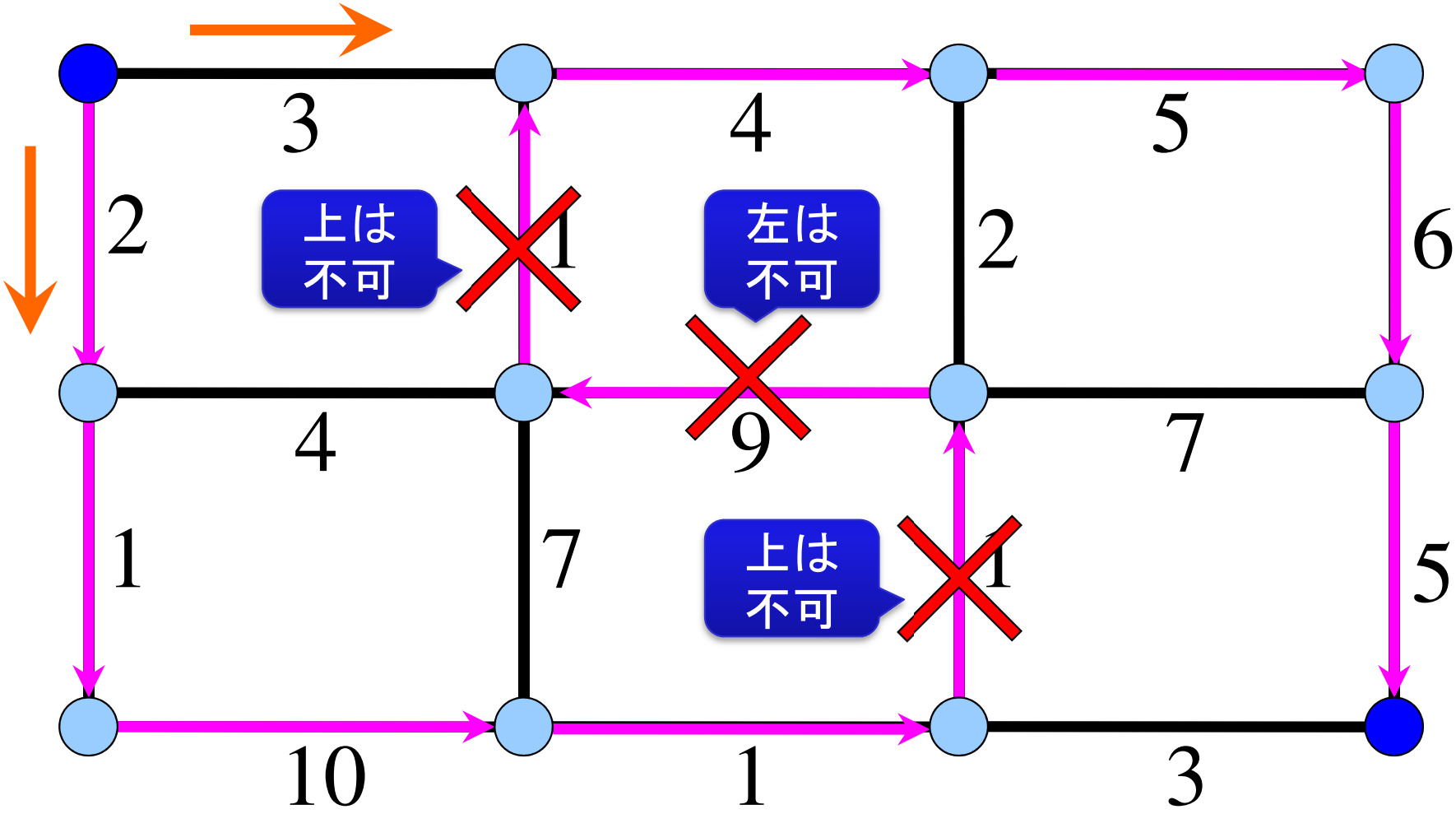
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

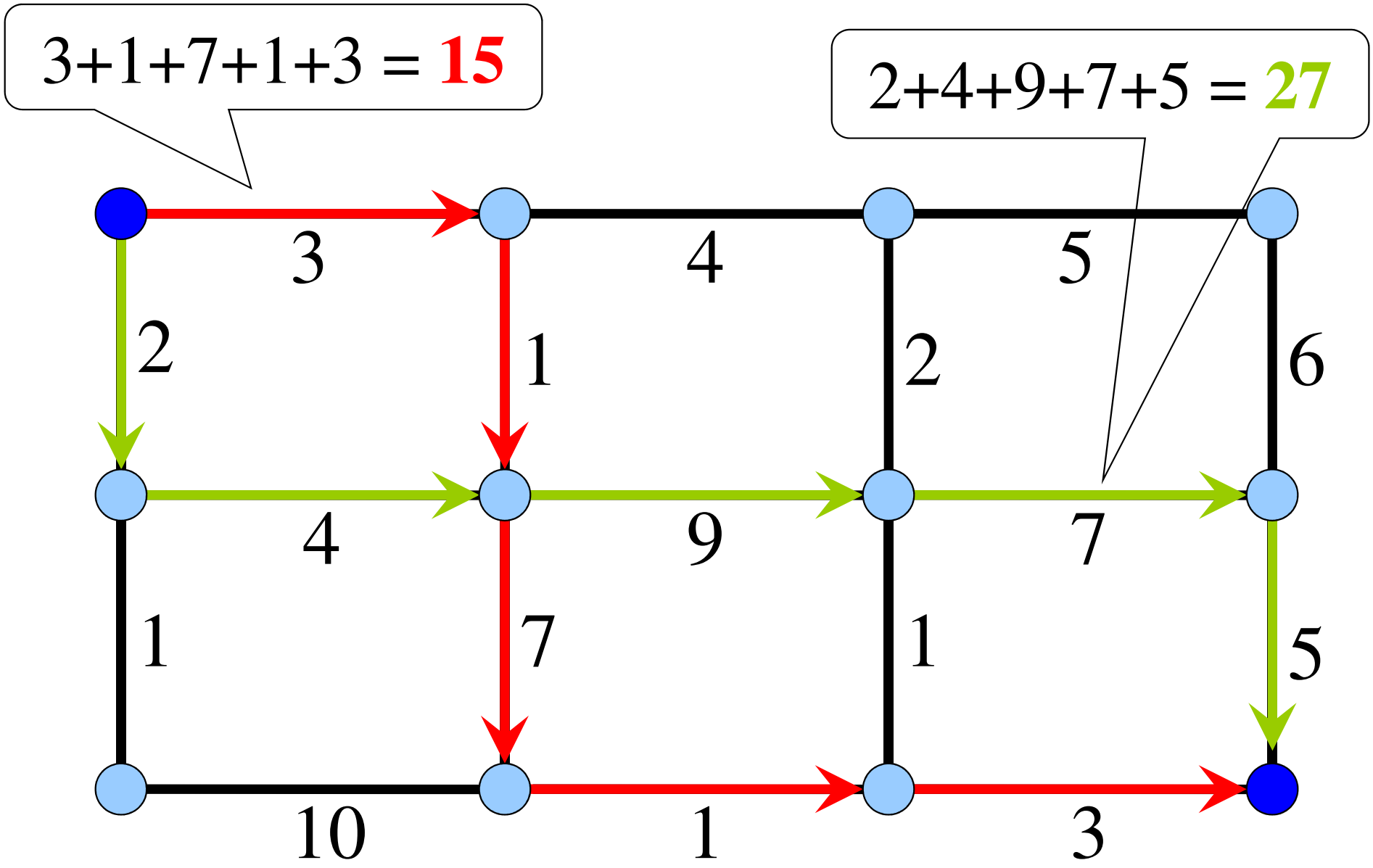
難しいなら易しくすればいいのさ！

制限した問題

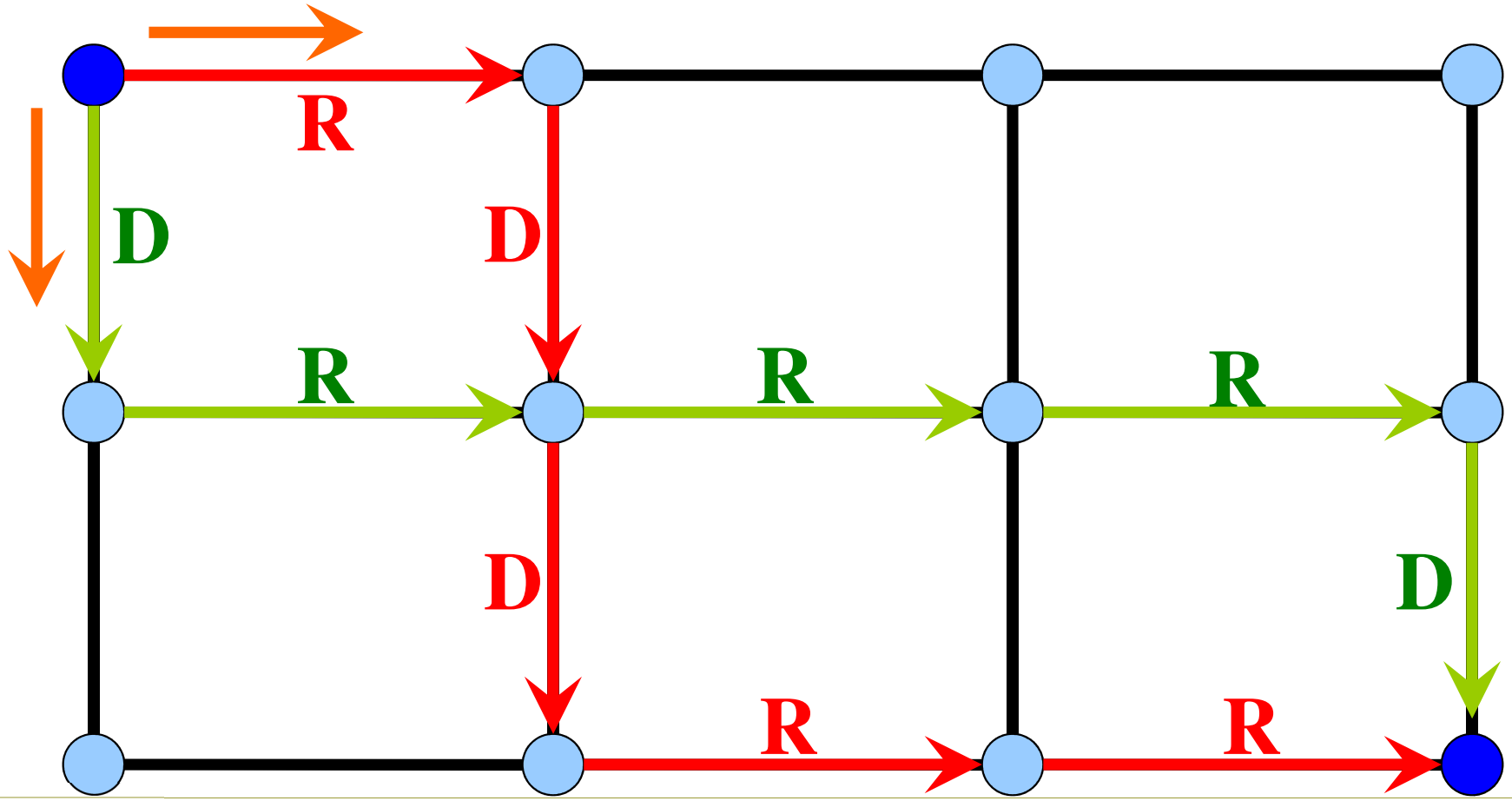
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



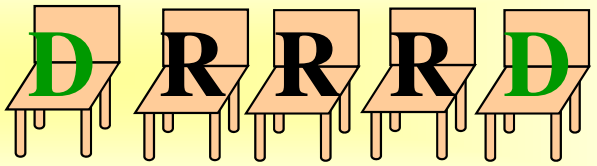
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



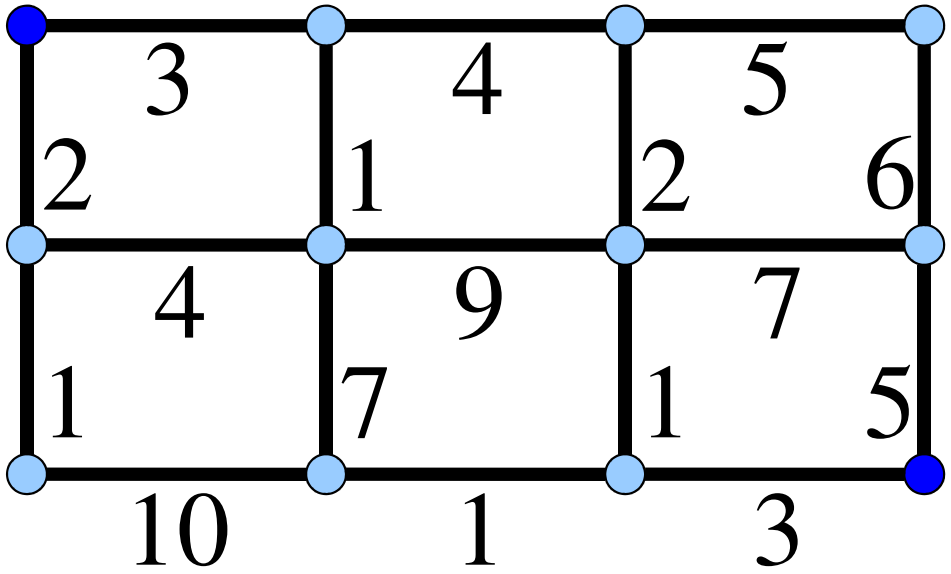
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

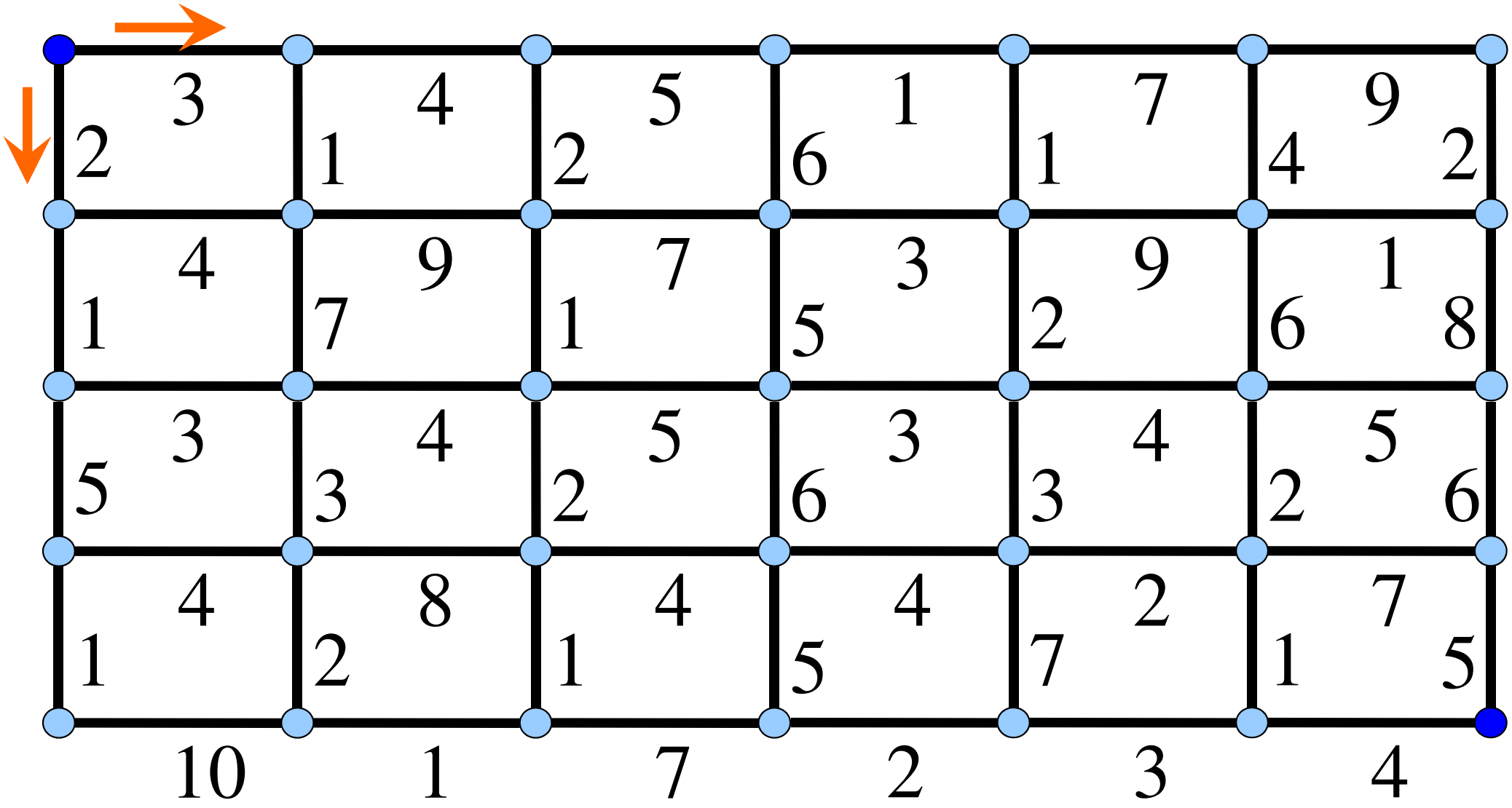
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

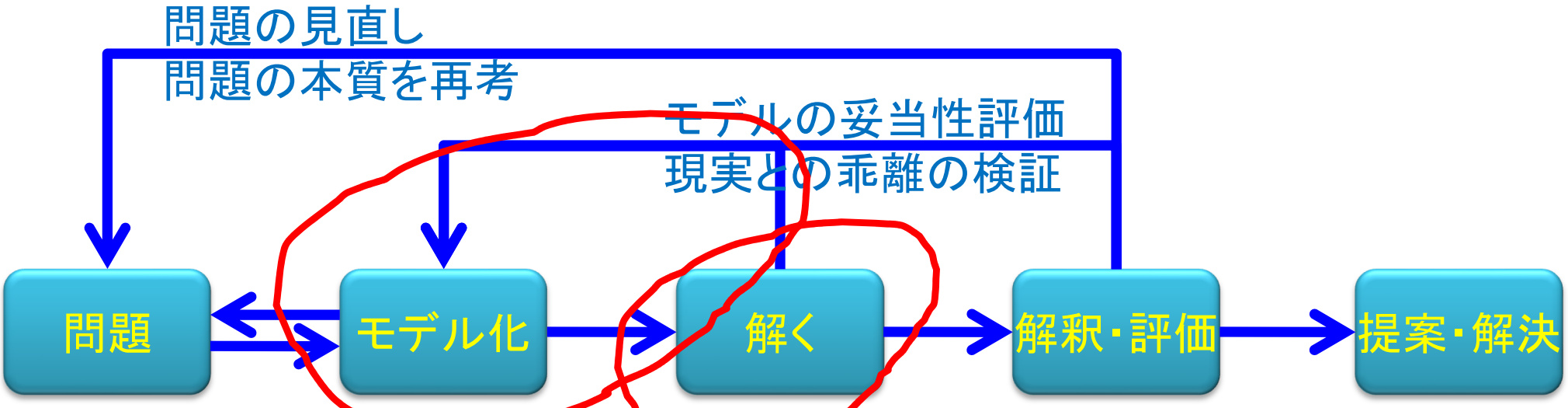
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

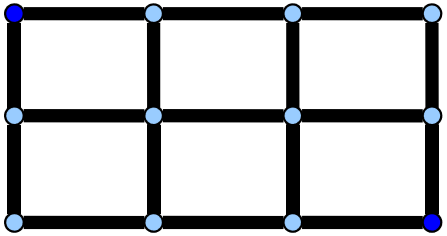
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

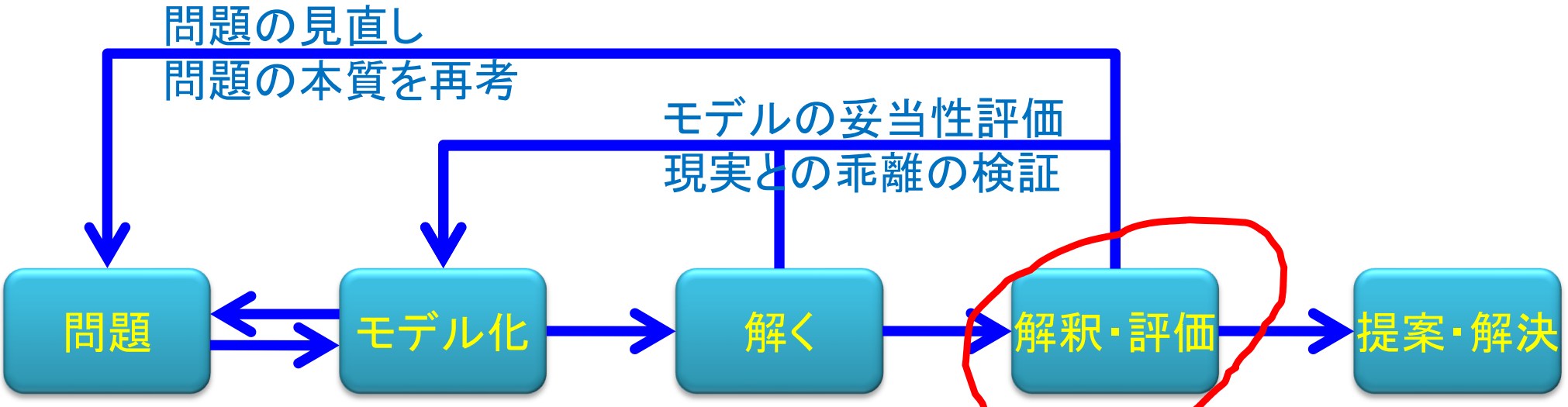
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

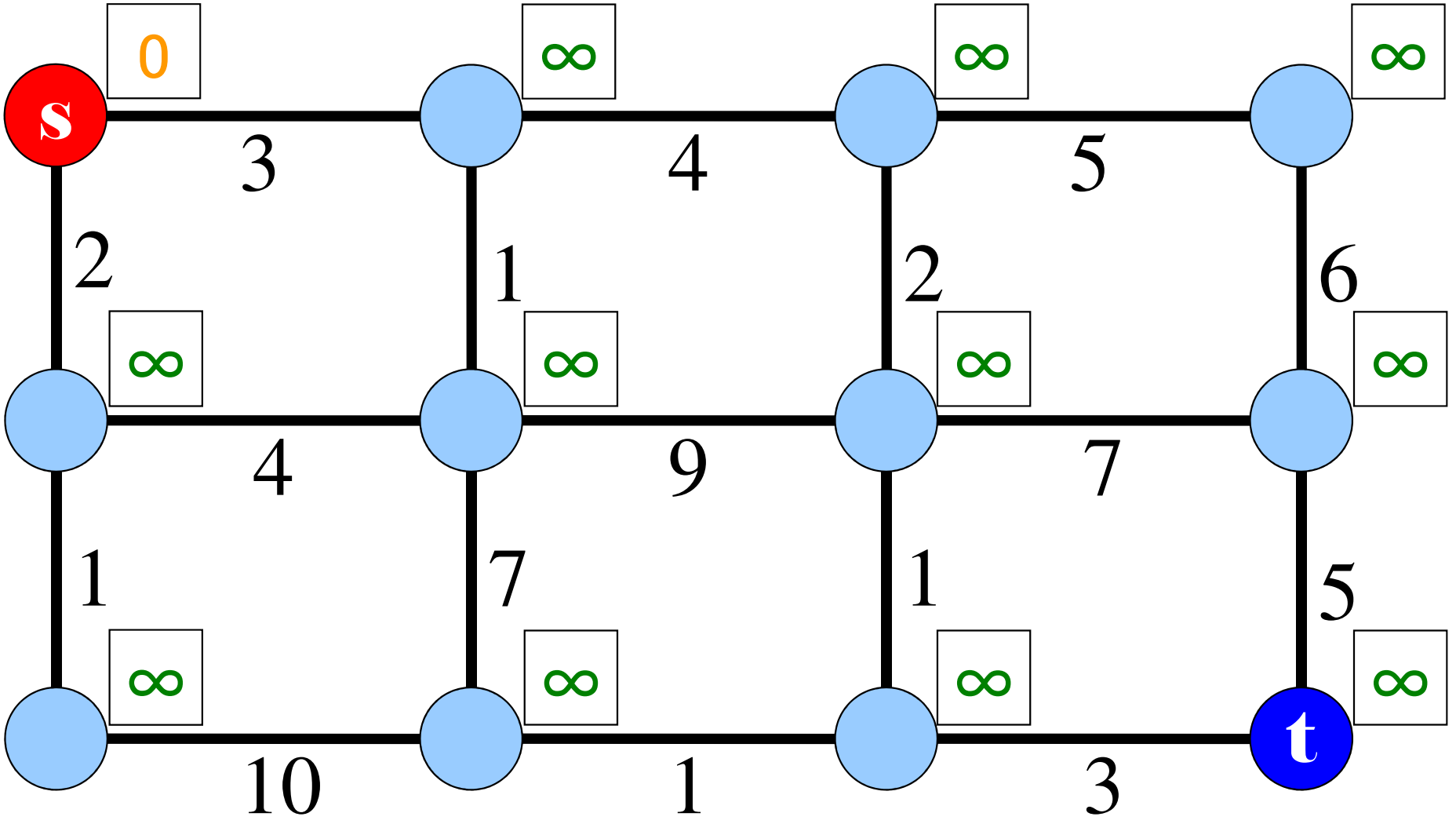
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

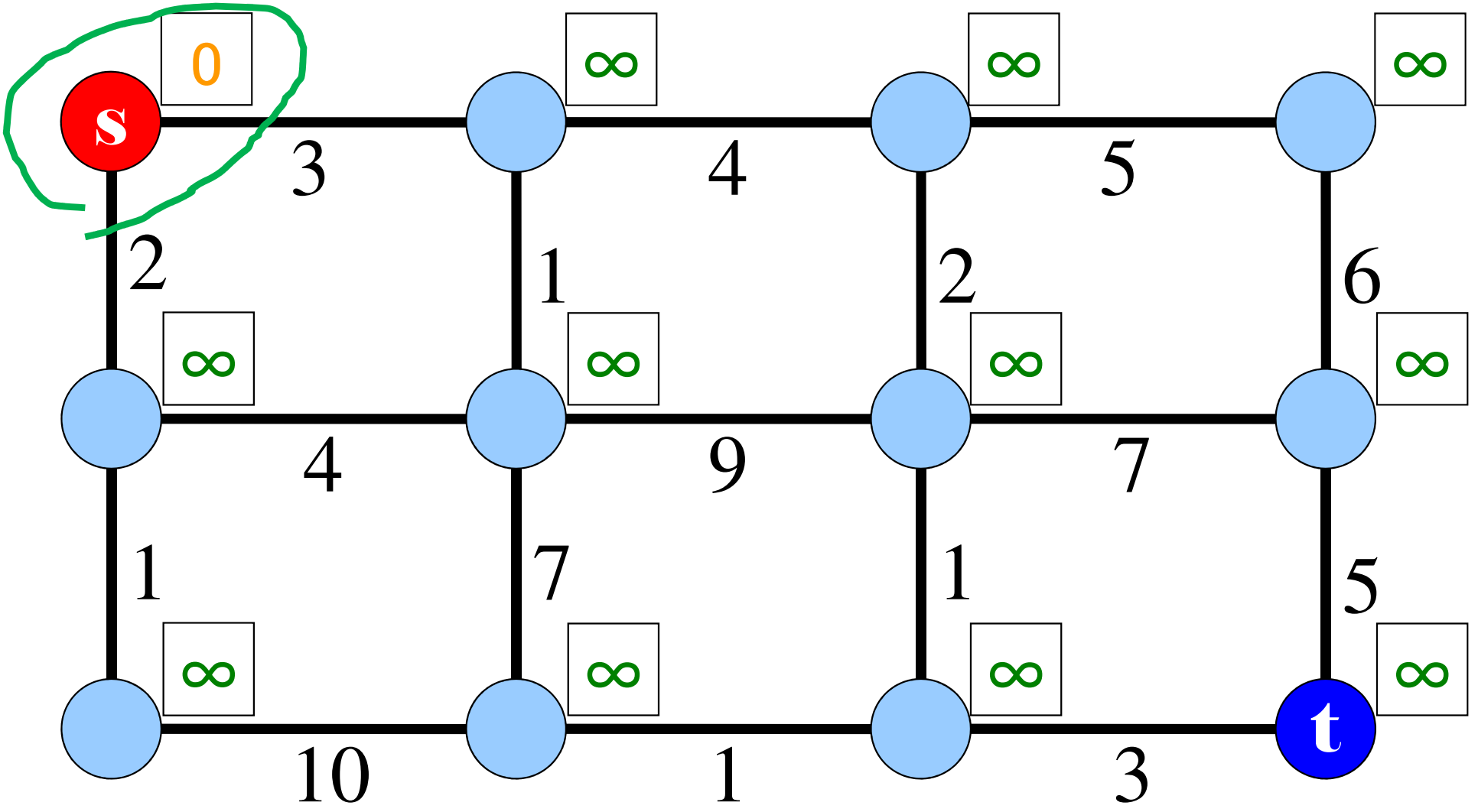
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



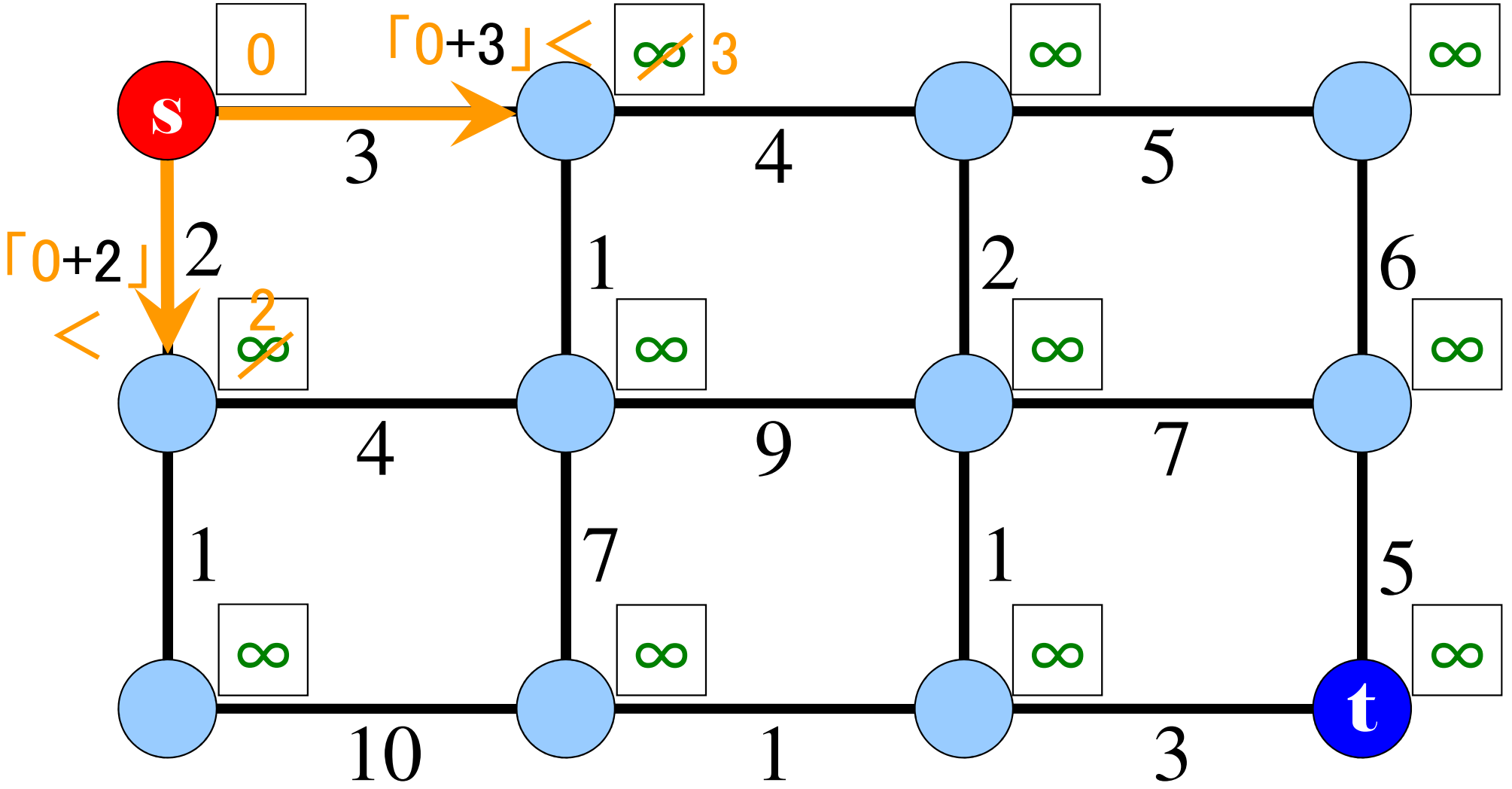
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

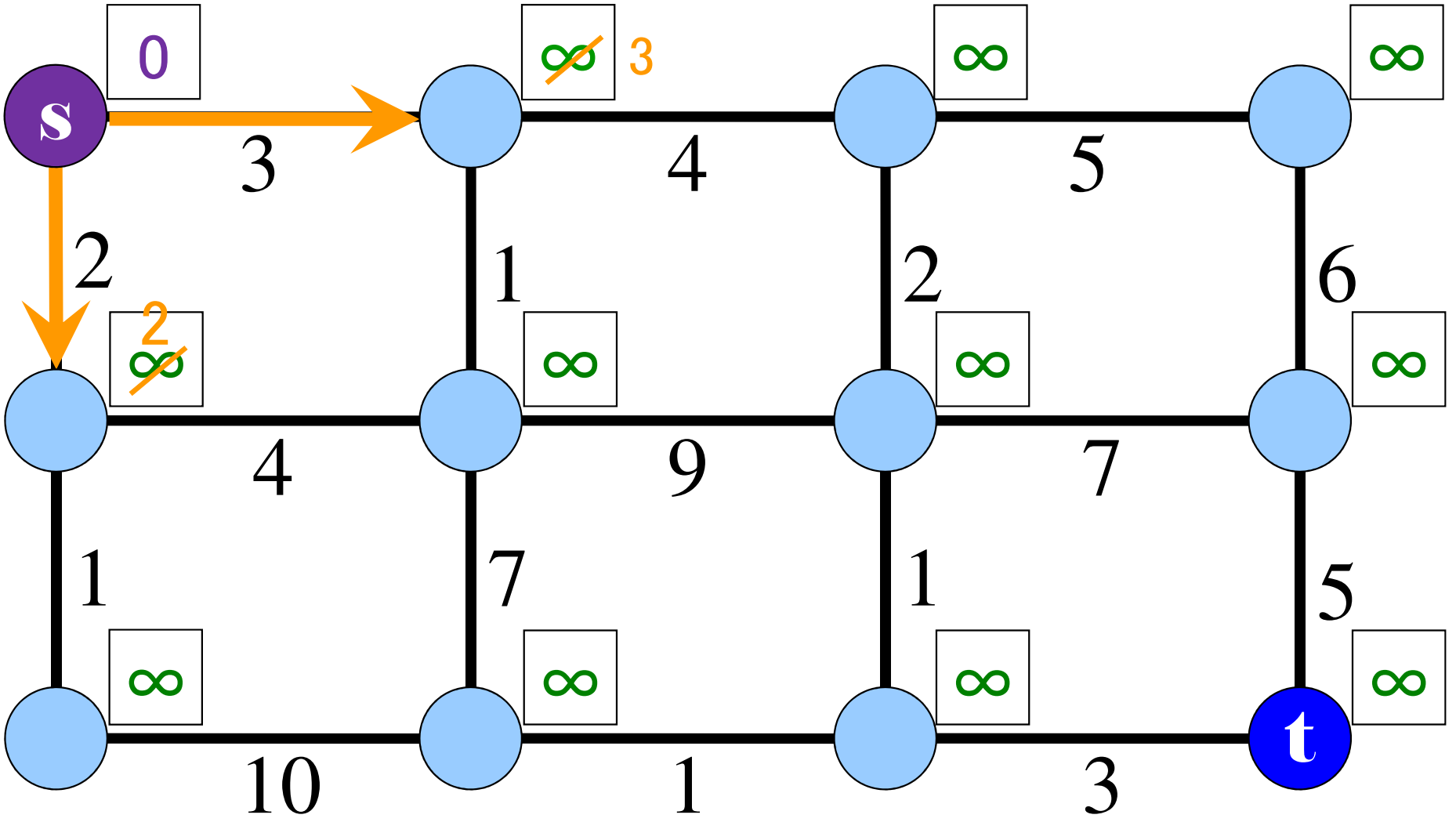


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

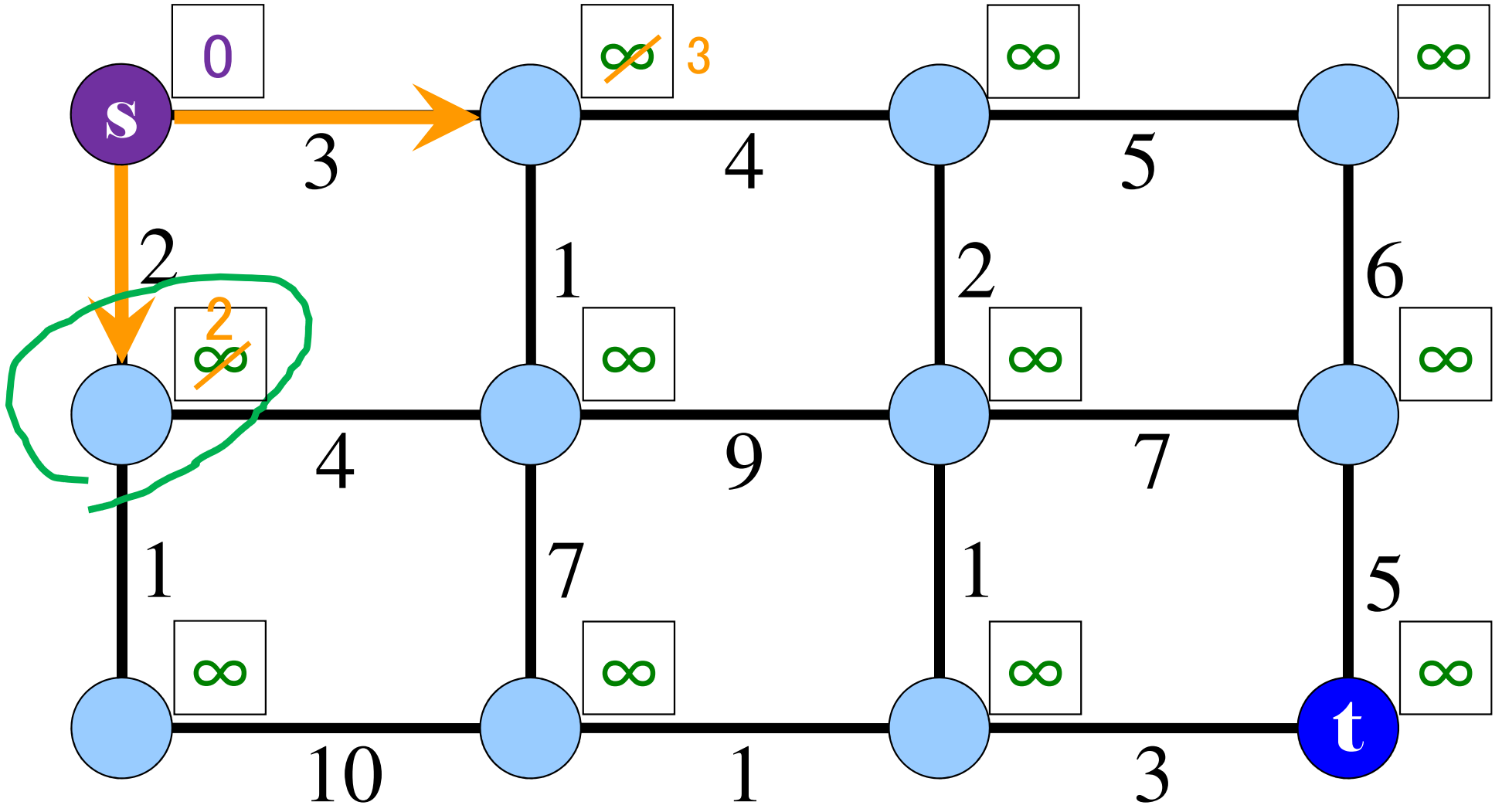
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



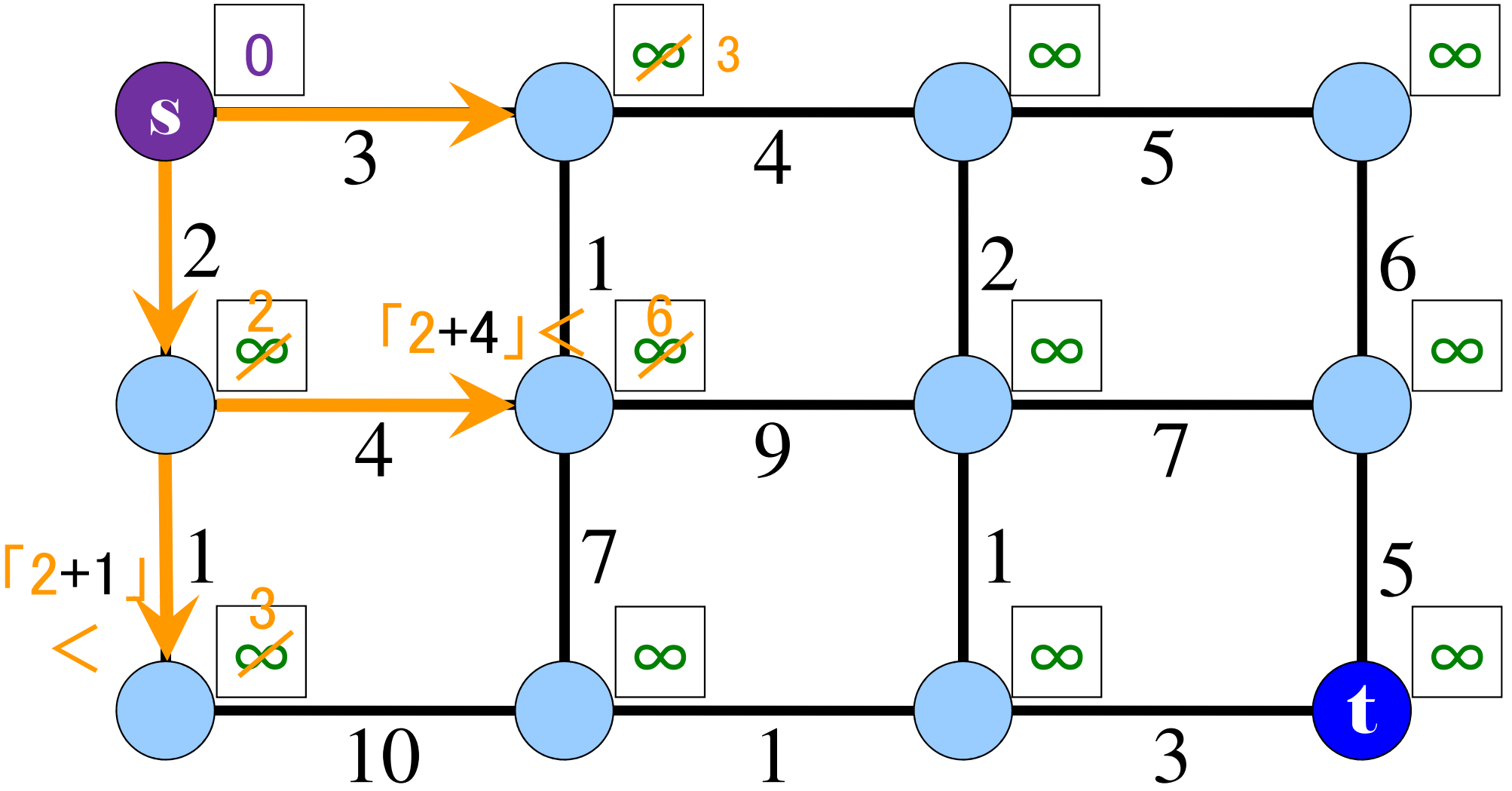
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

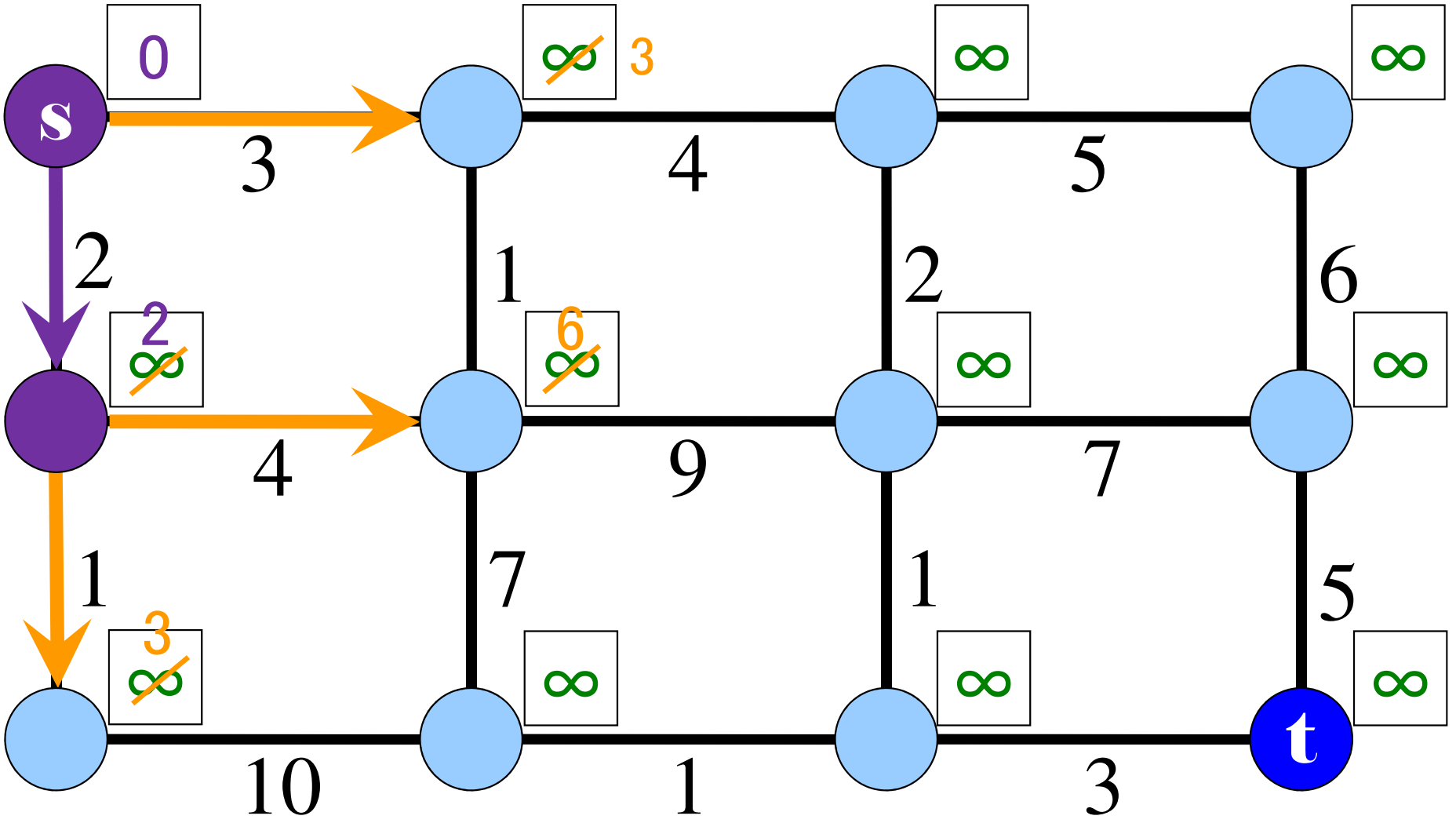


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

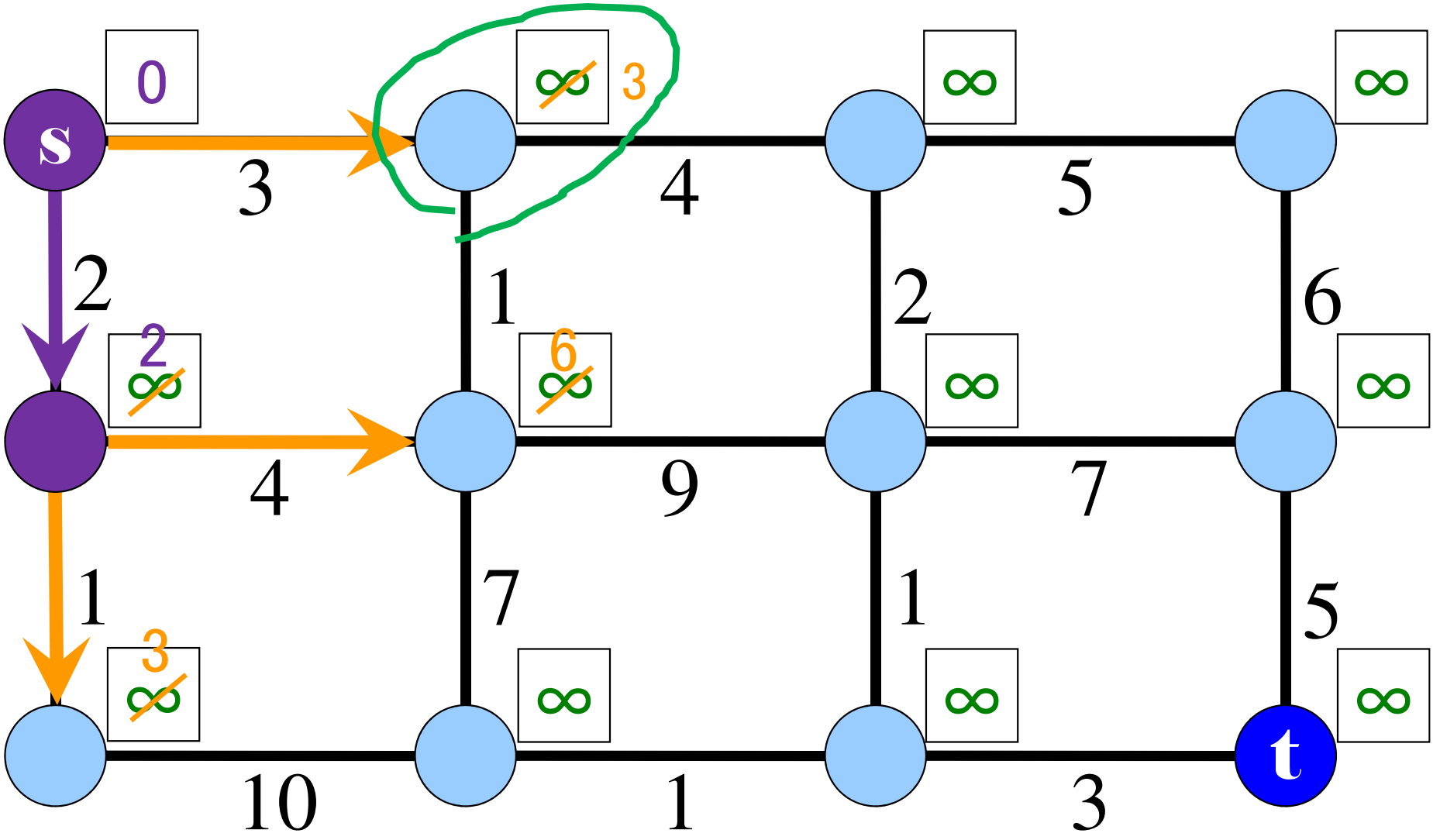
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



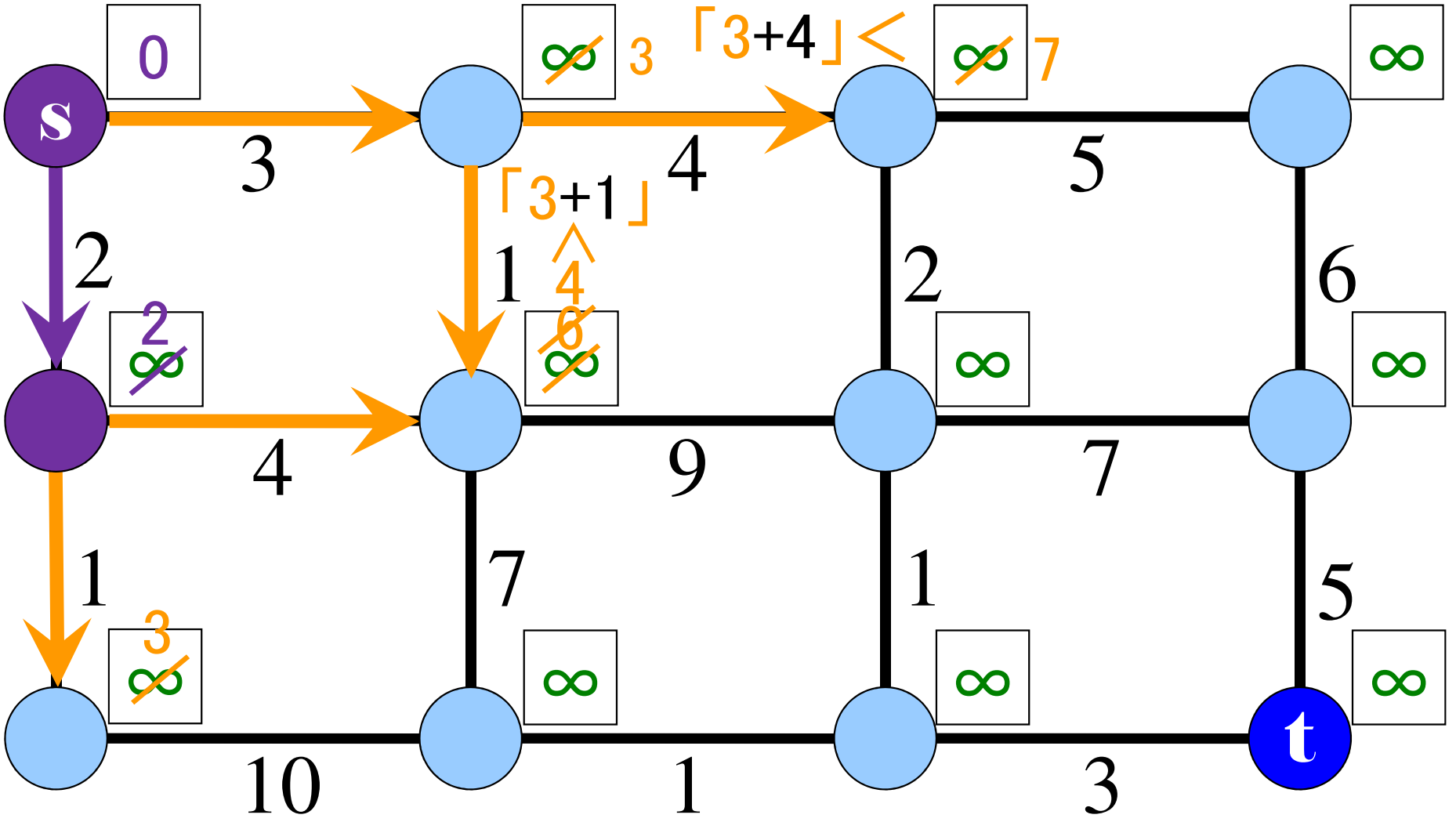
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

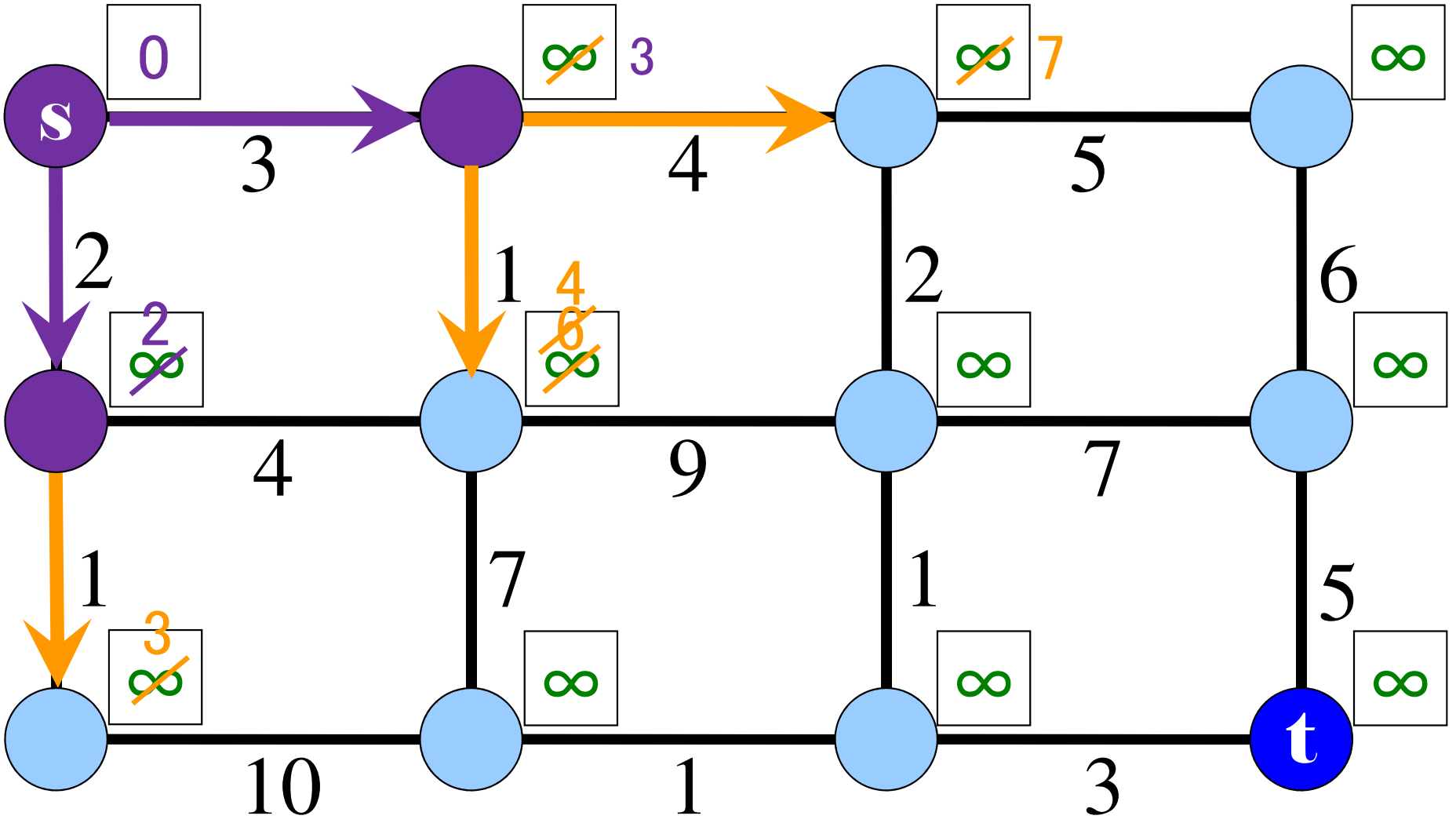


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

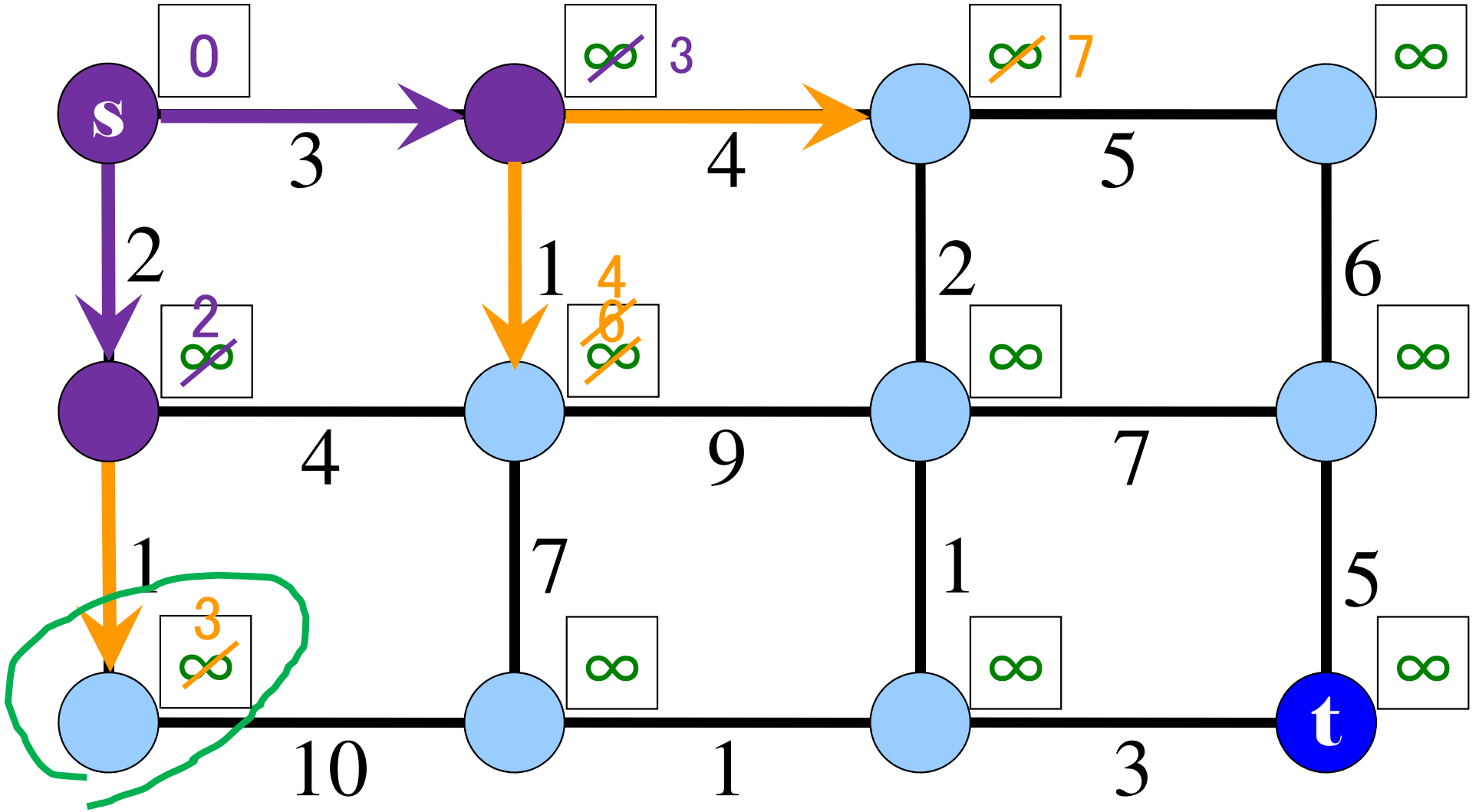
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



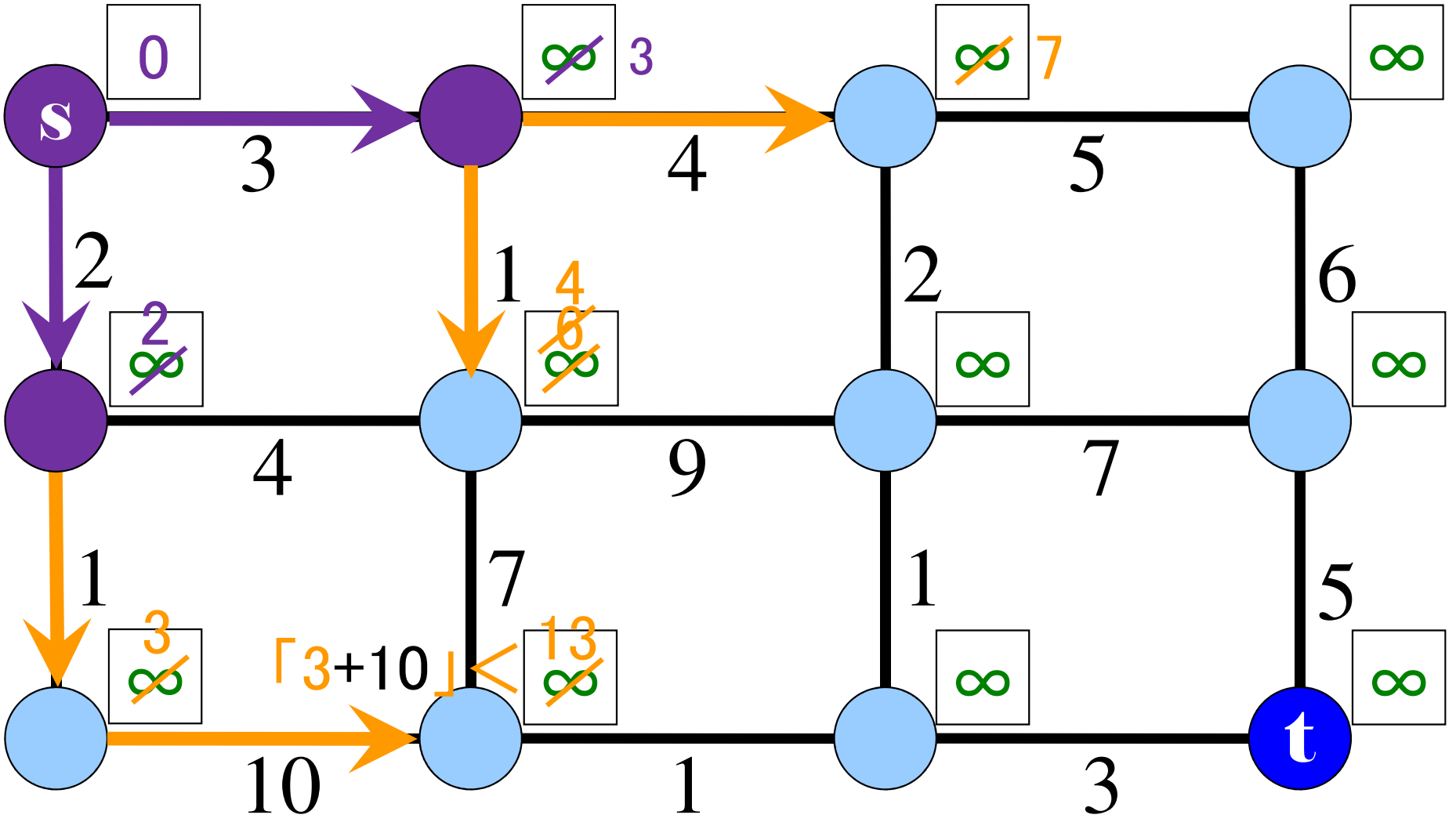
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

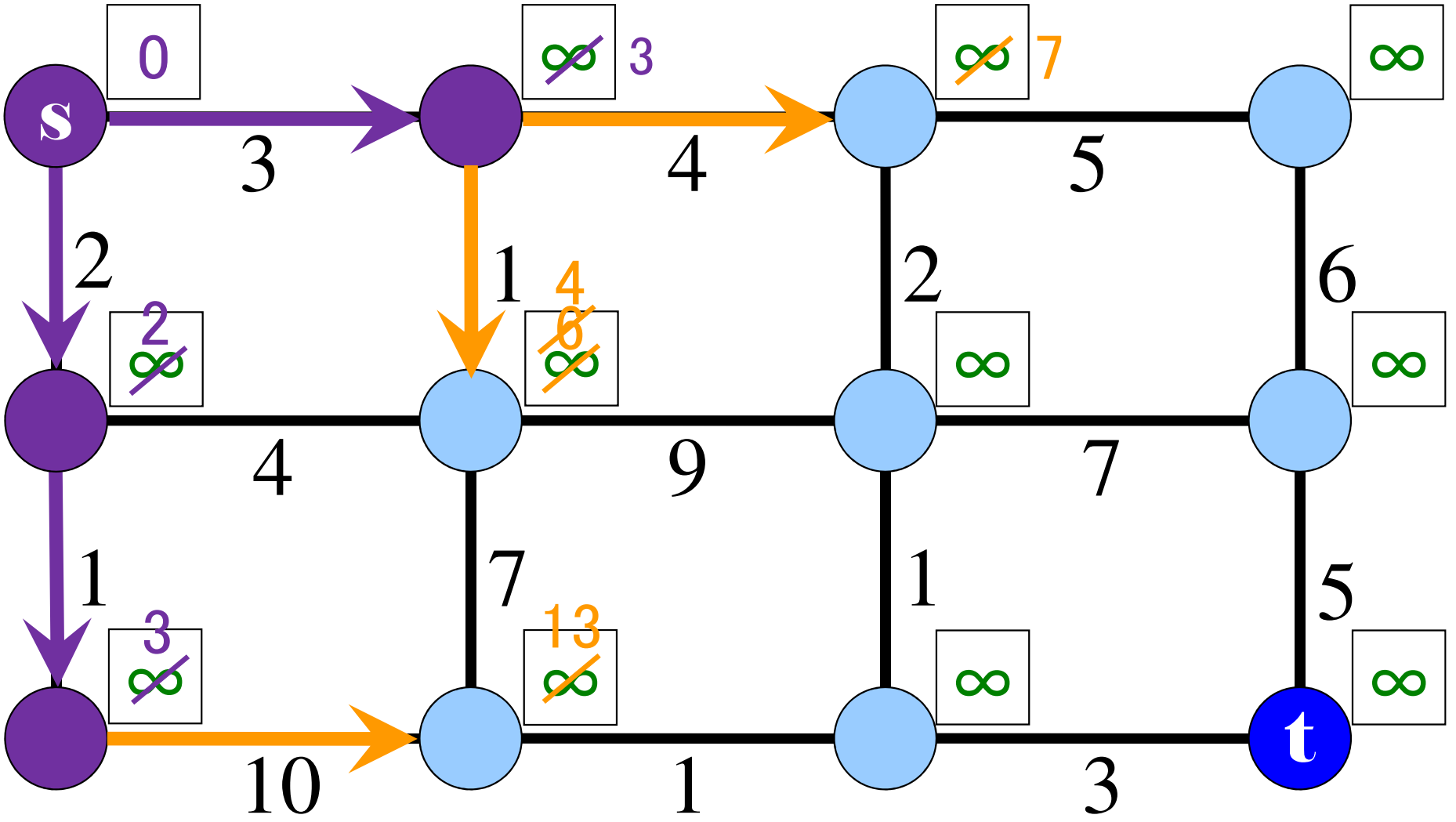


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

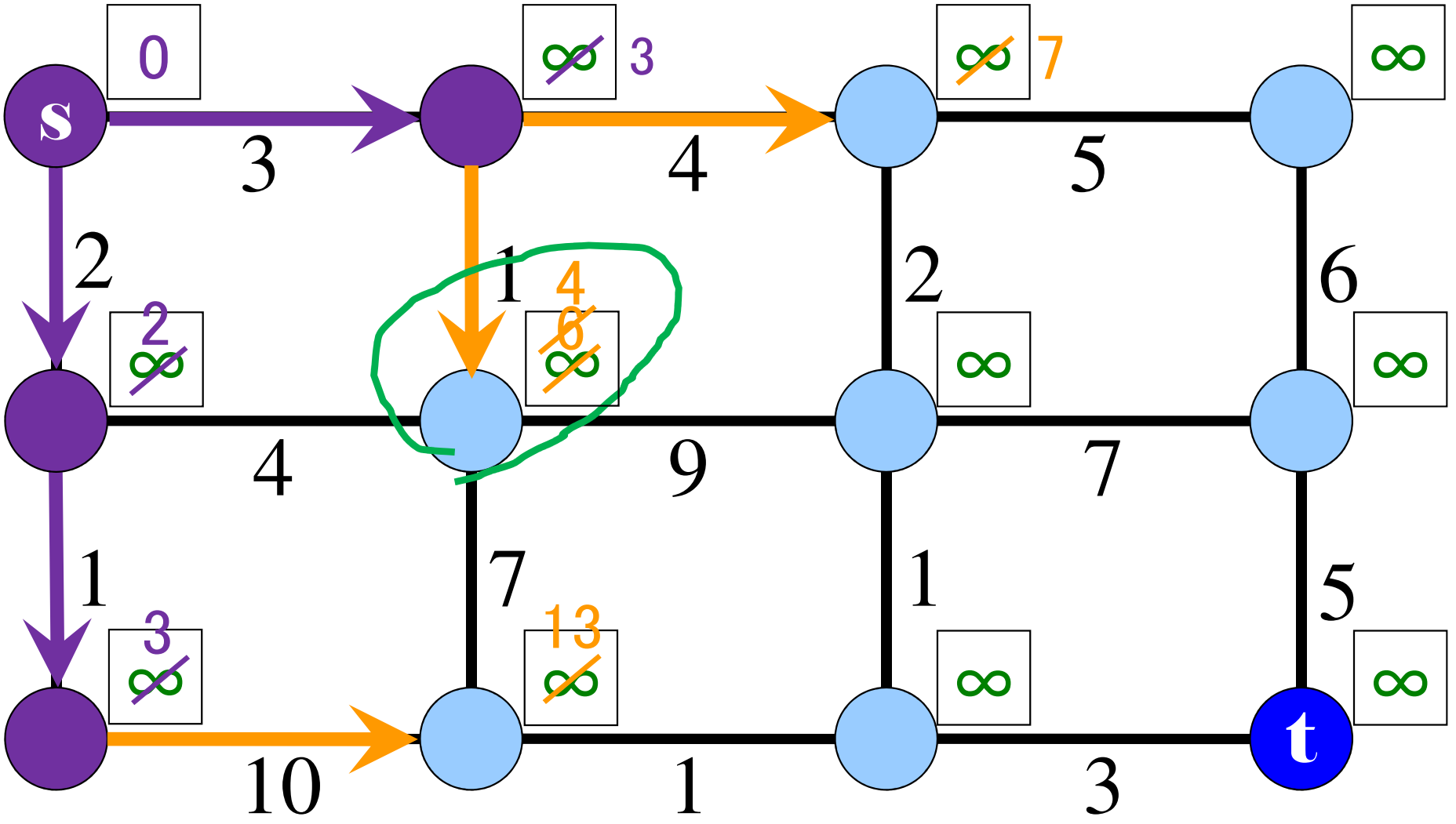
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



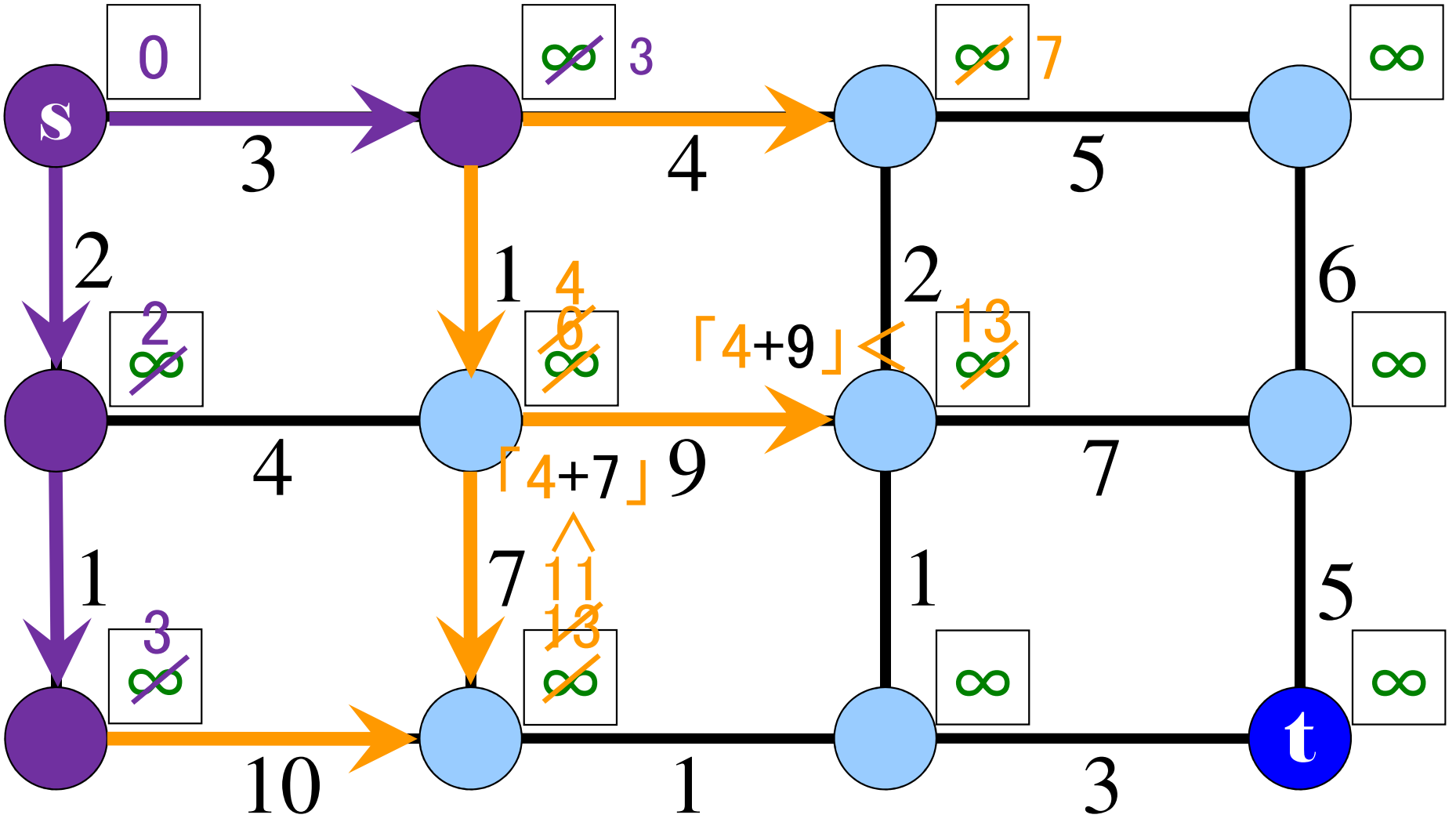
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

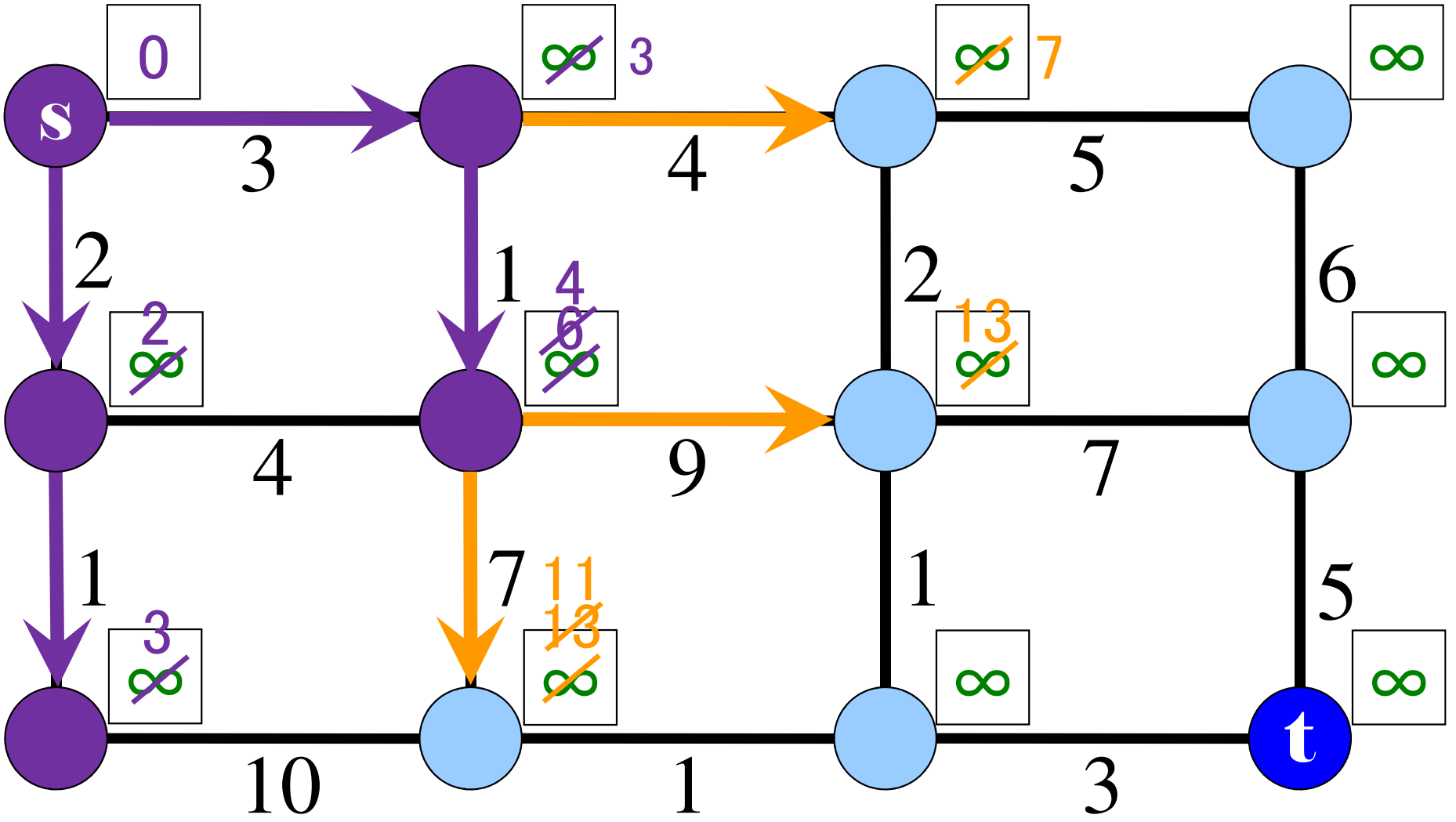


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

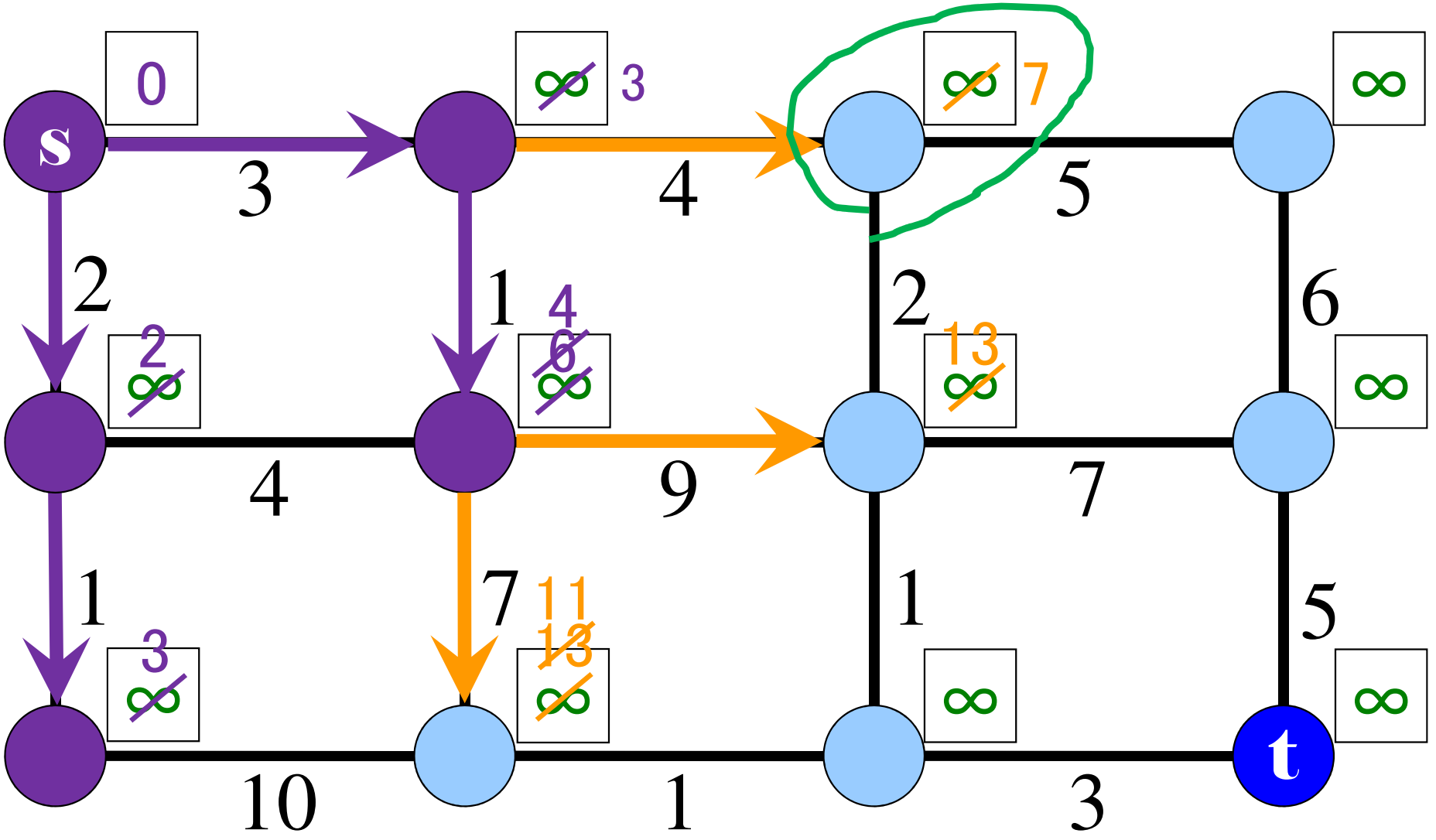
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



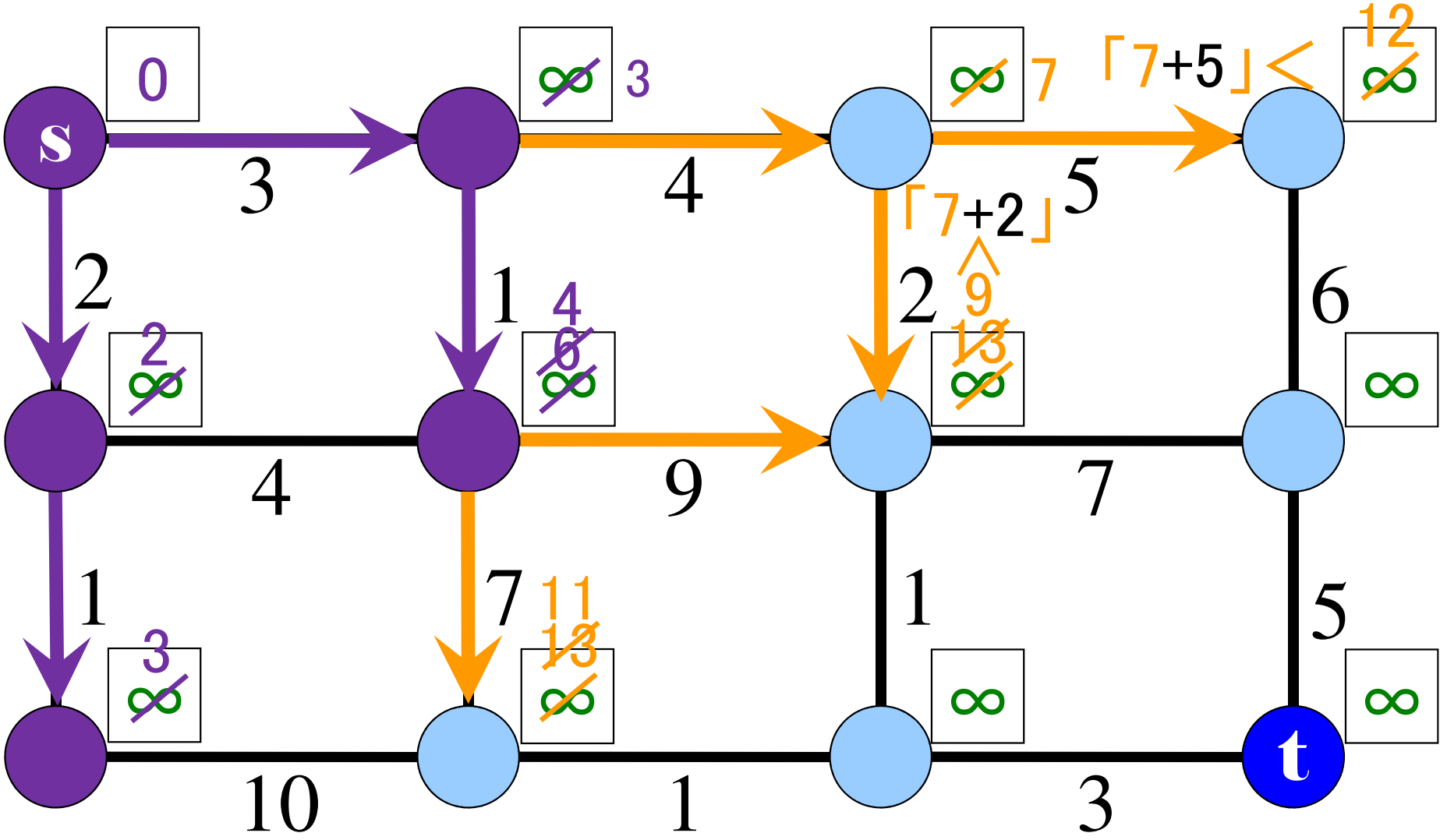
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

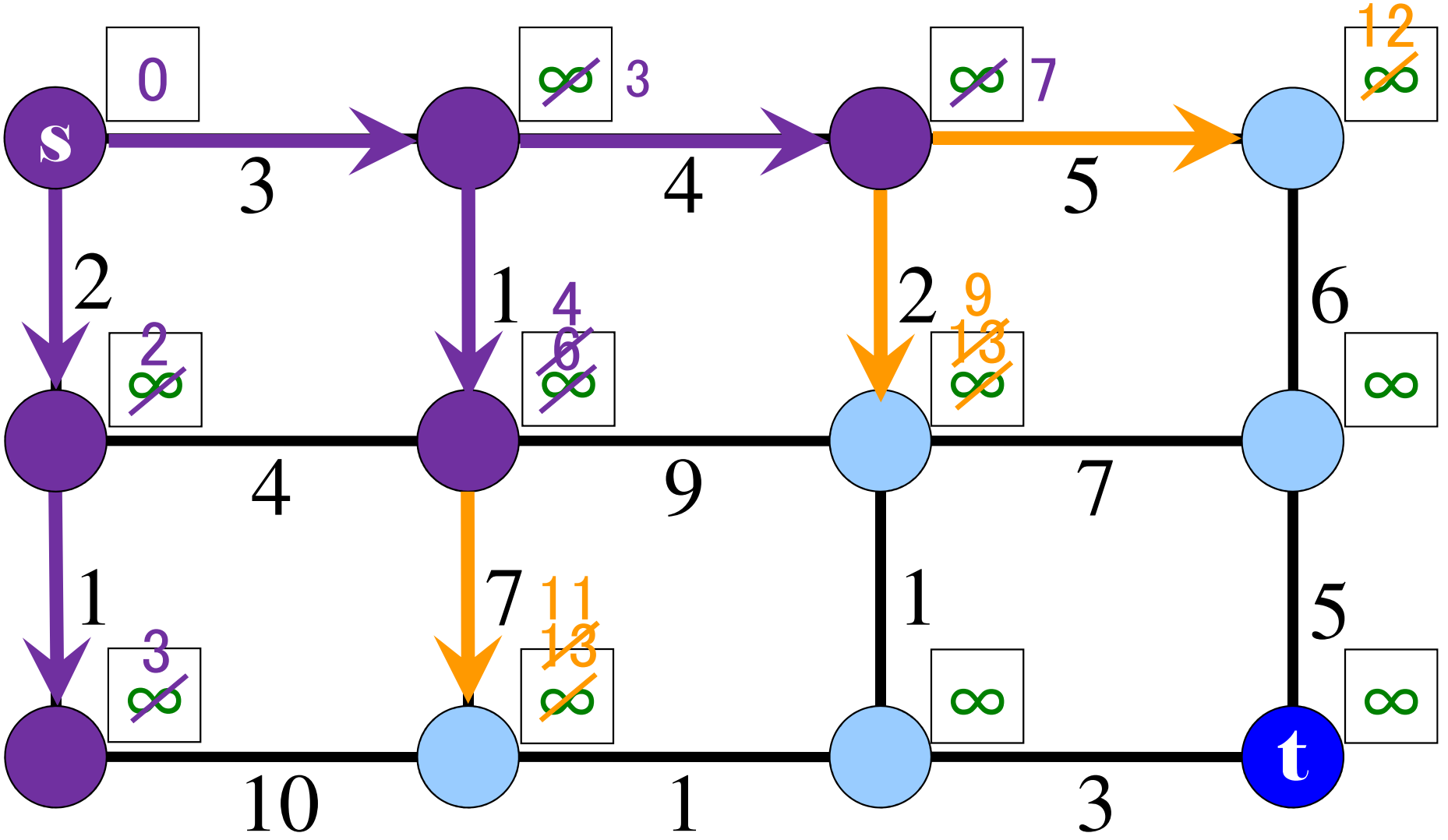


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

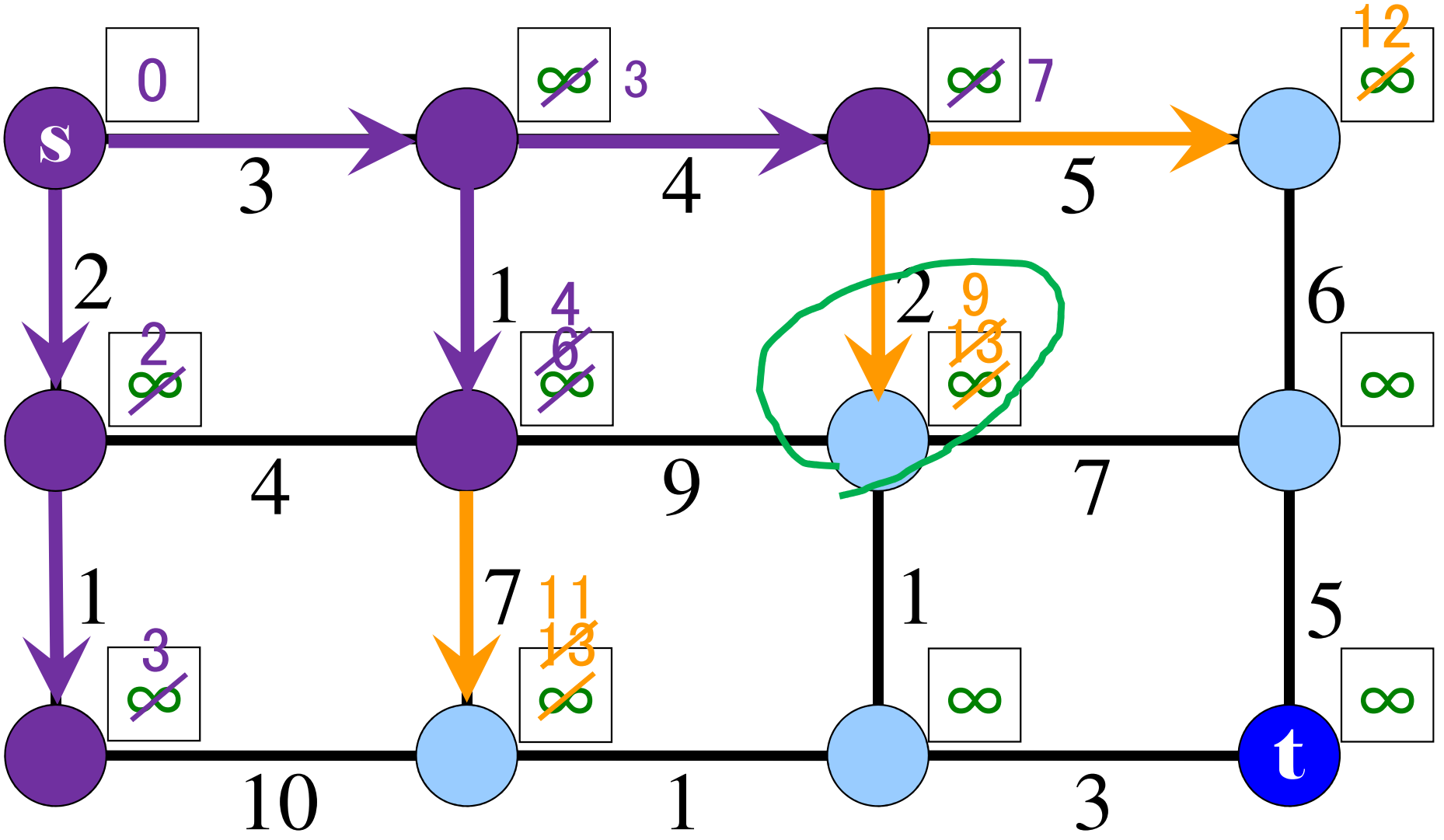
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



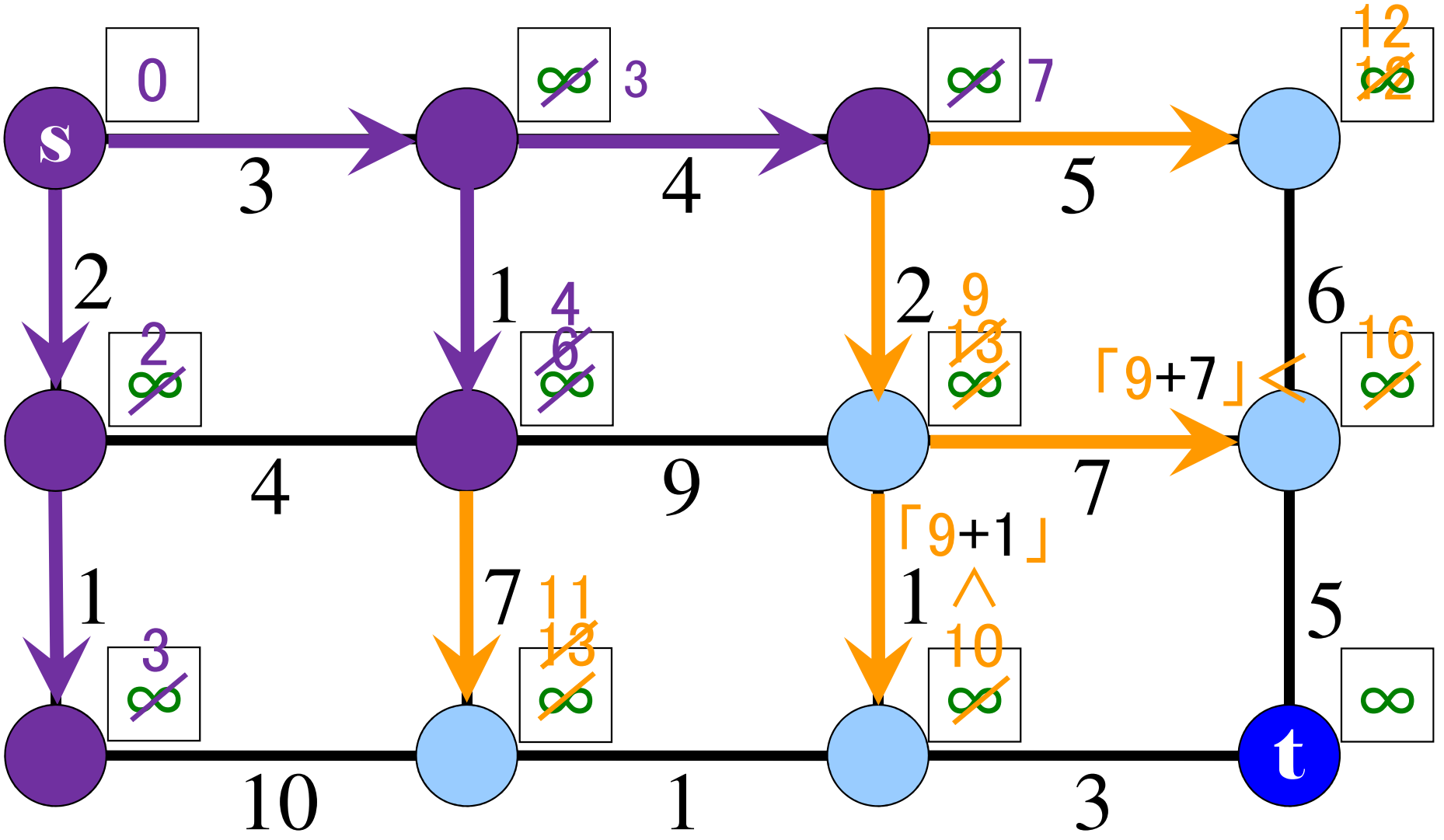
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

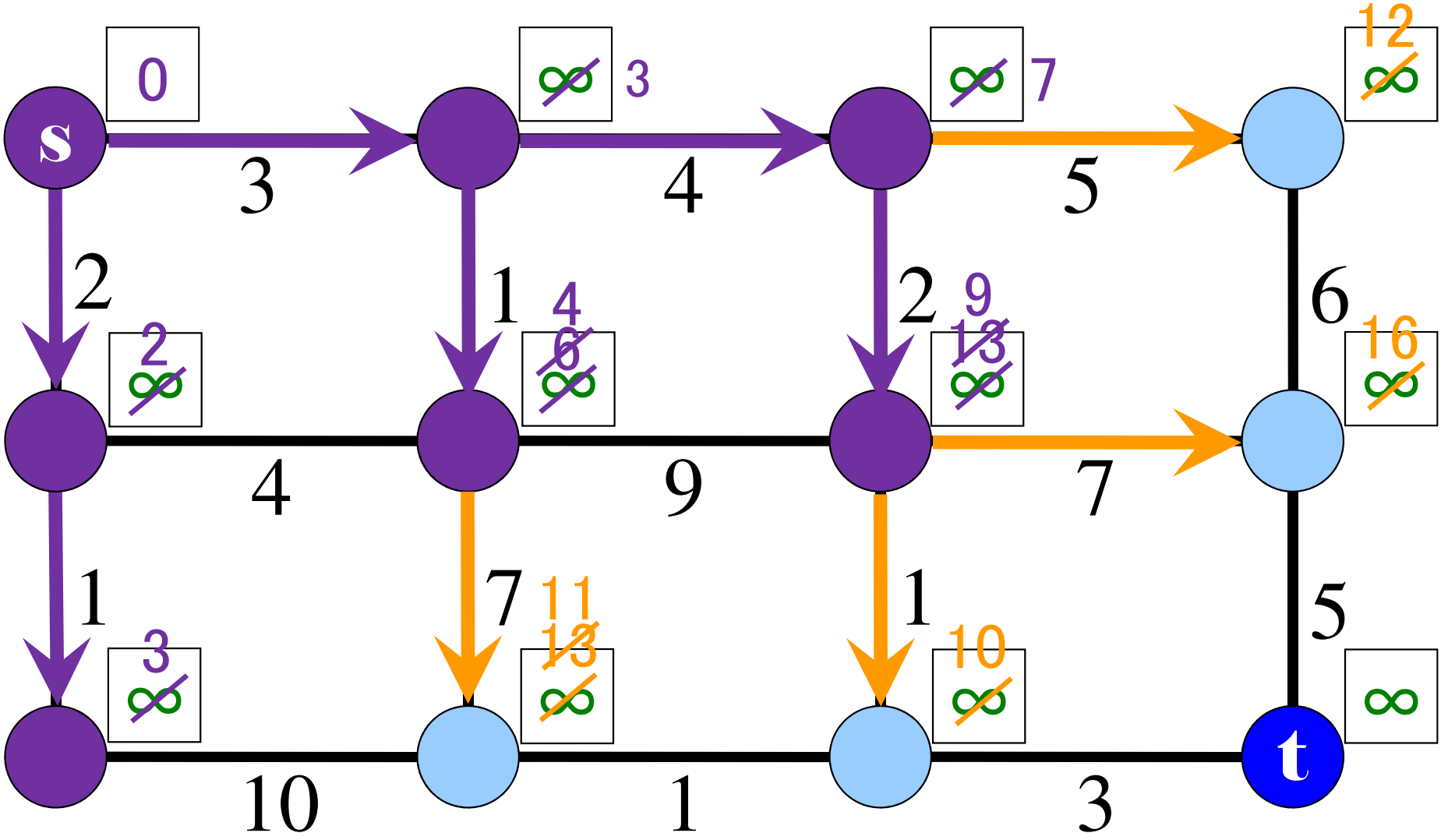


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

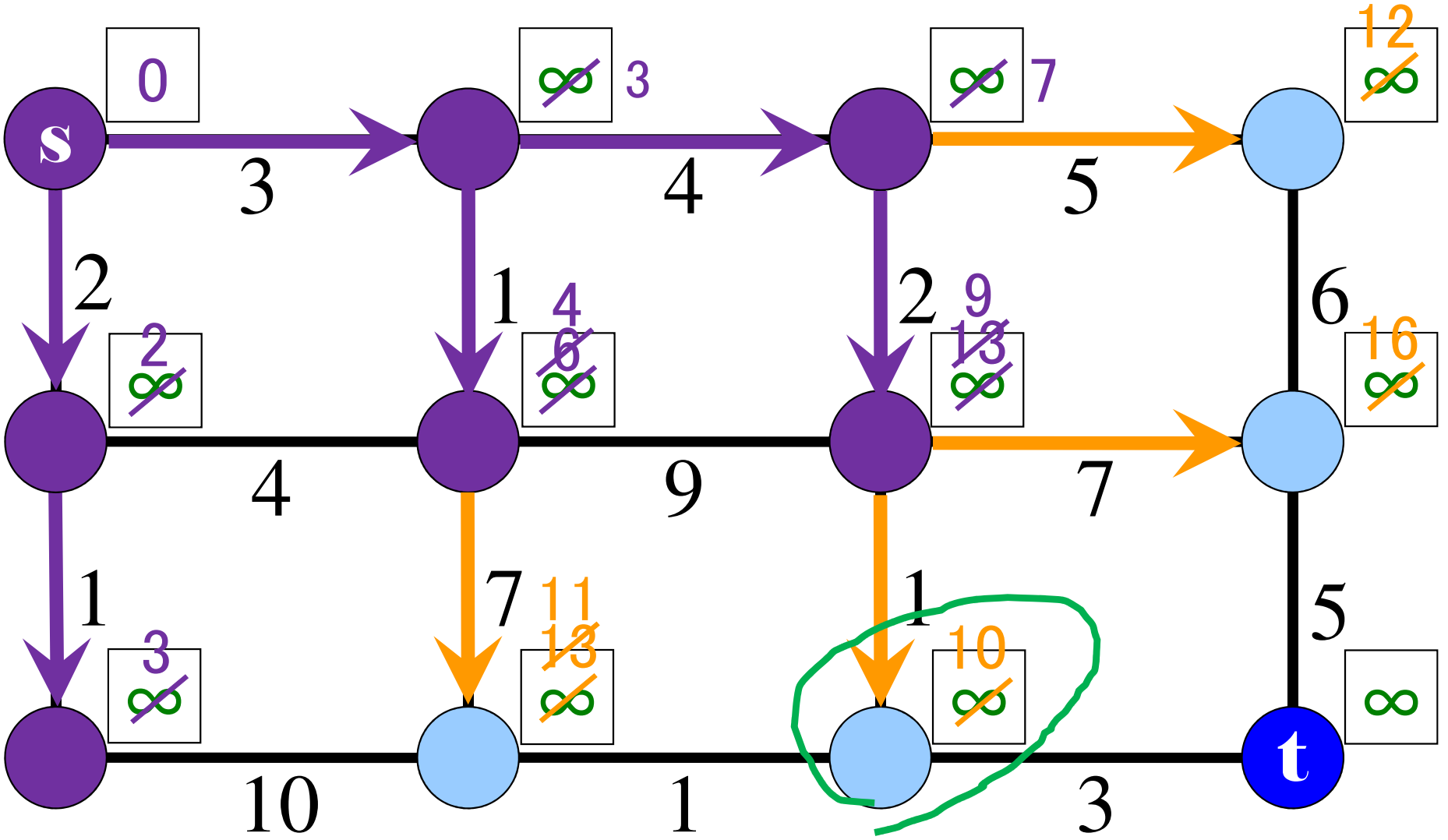
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



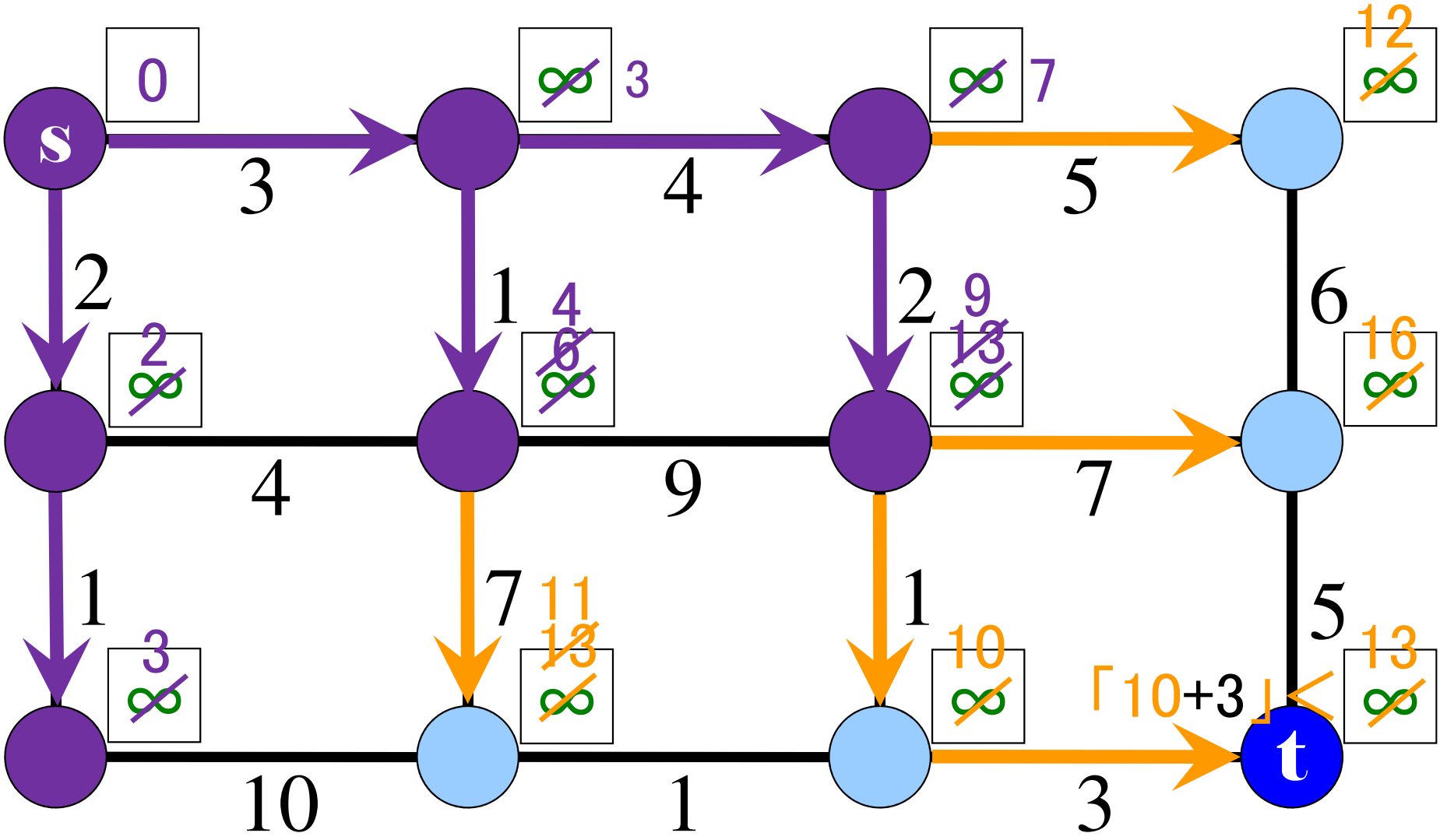
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

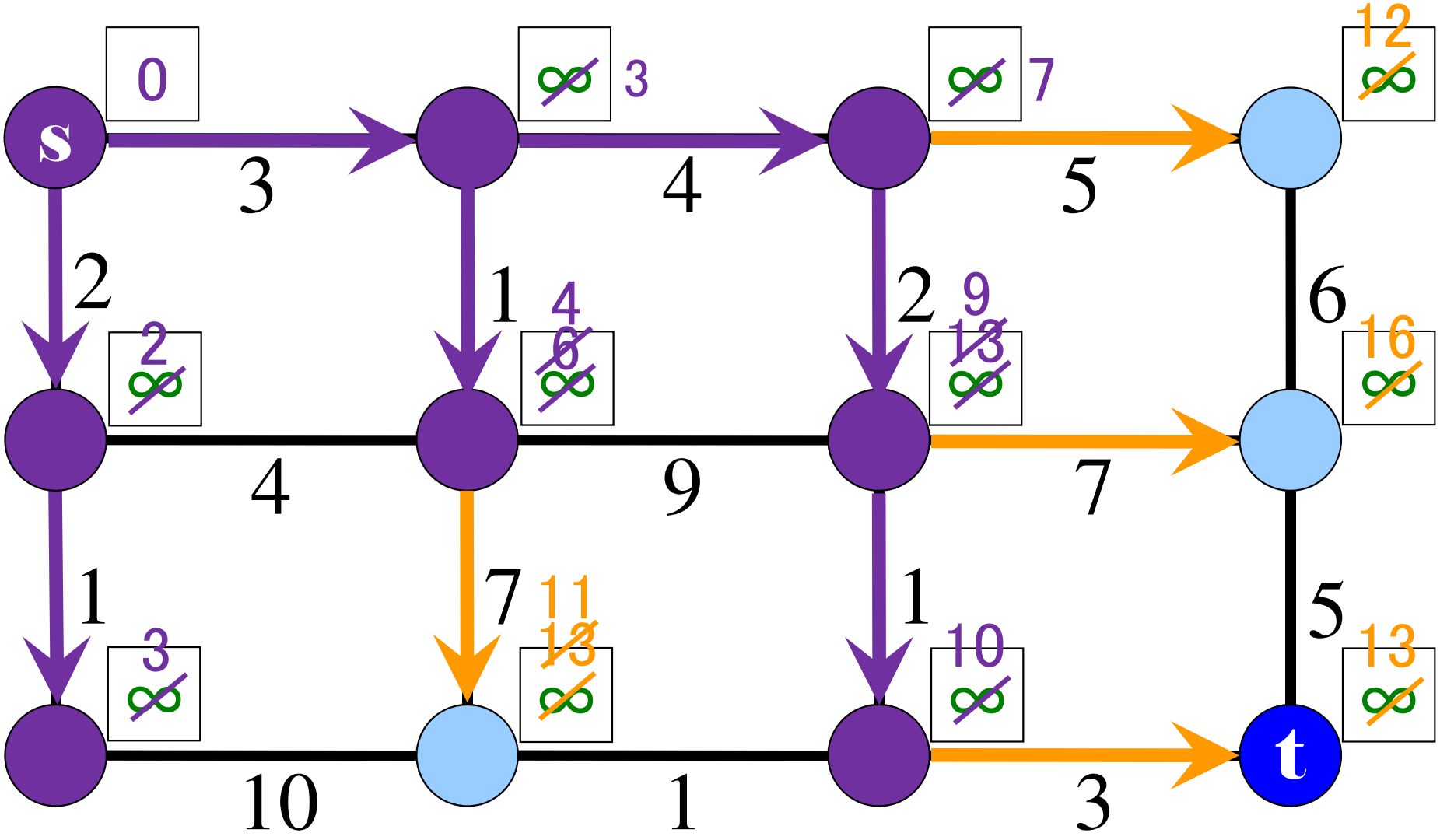


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

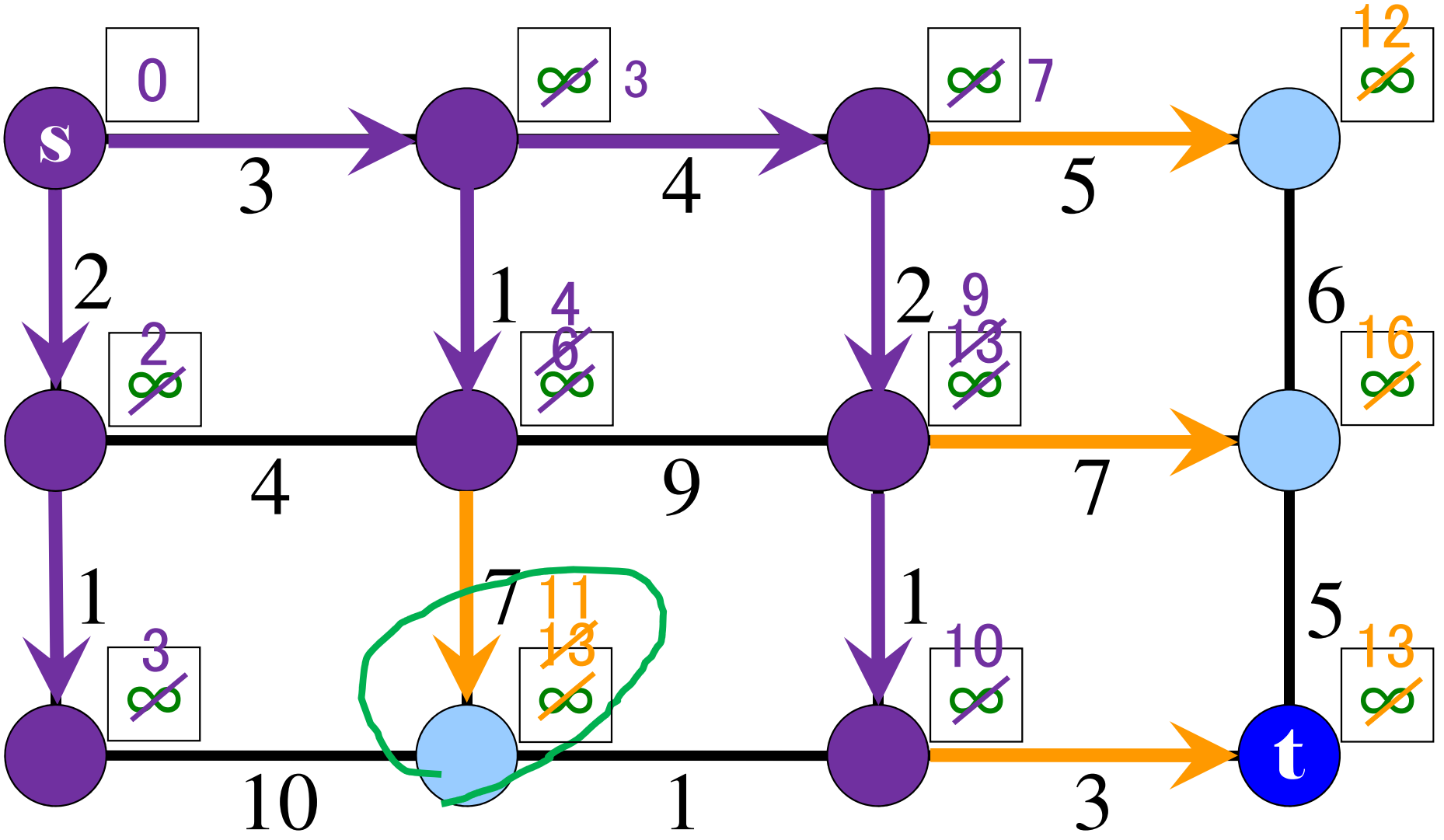
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



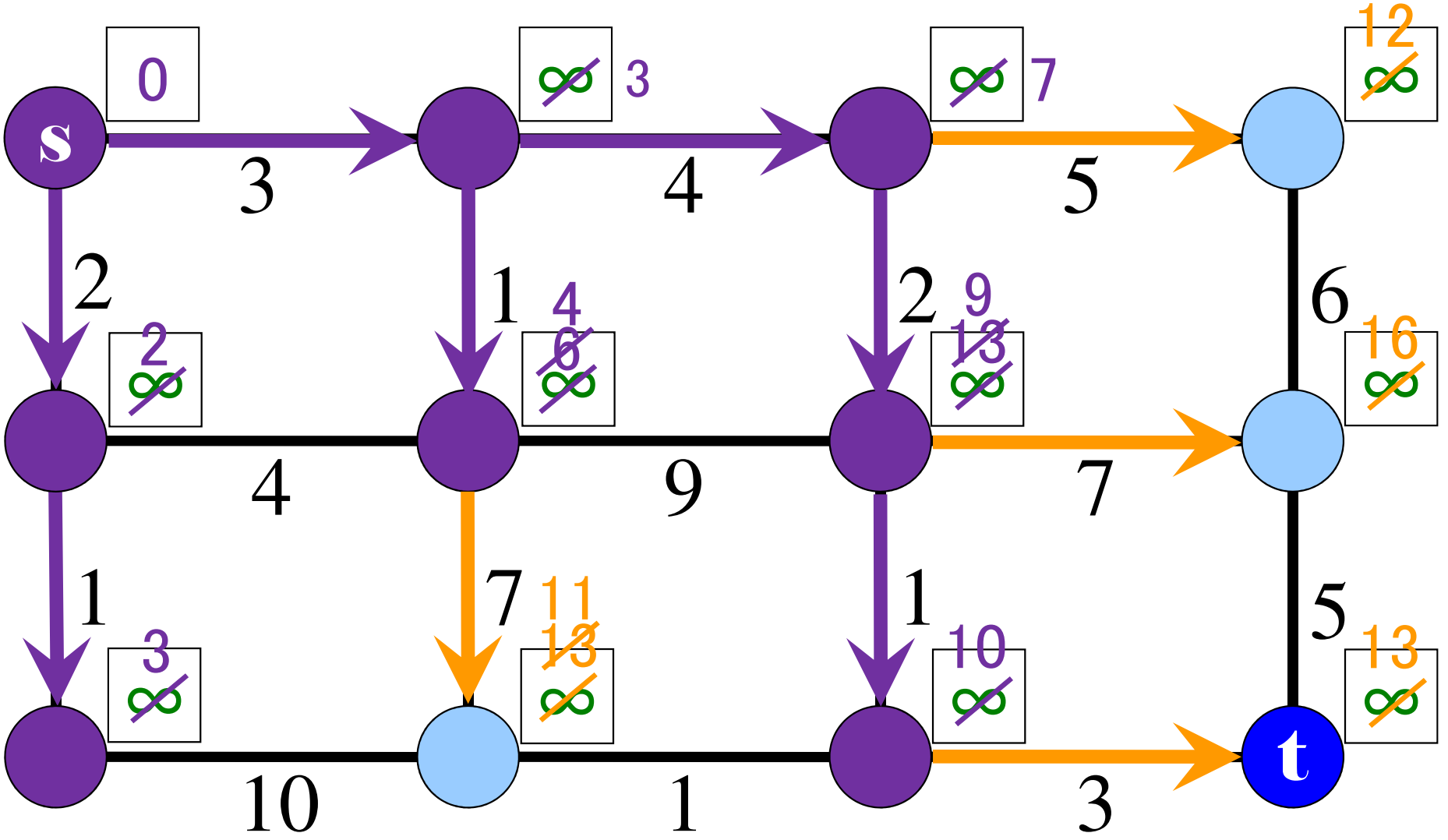
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

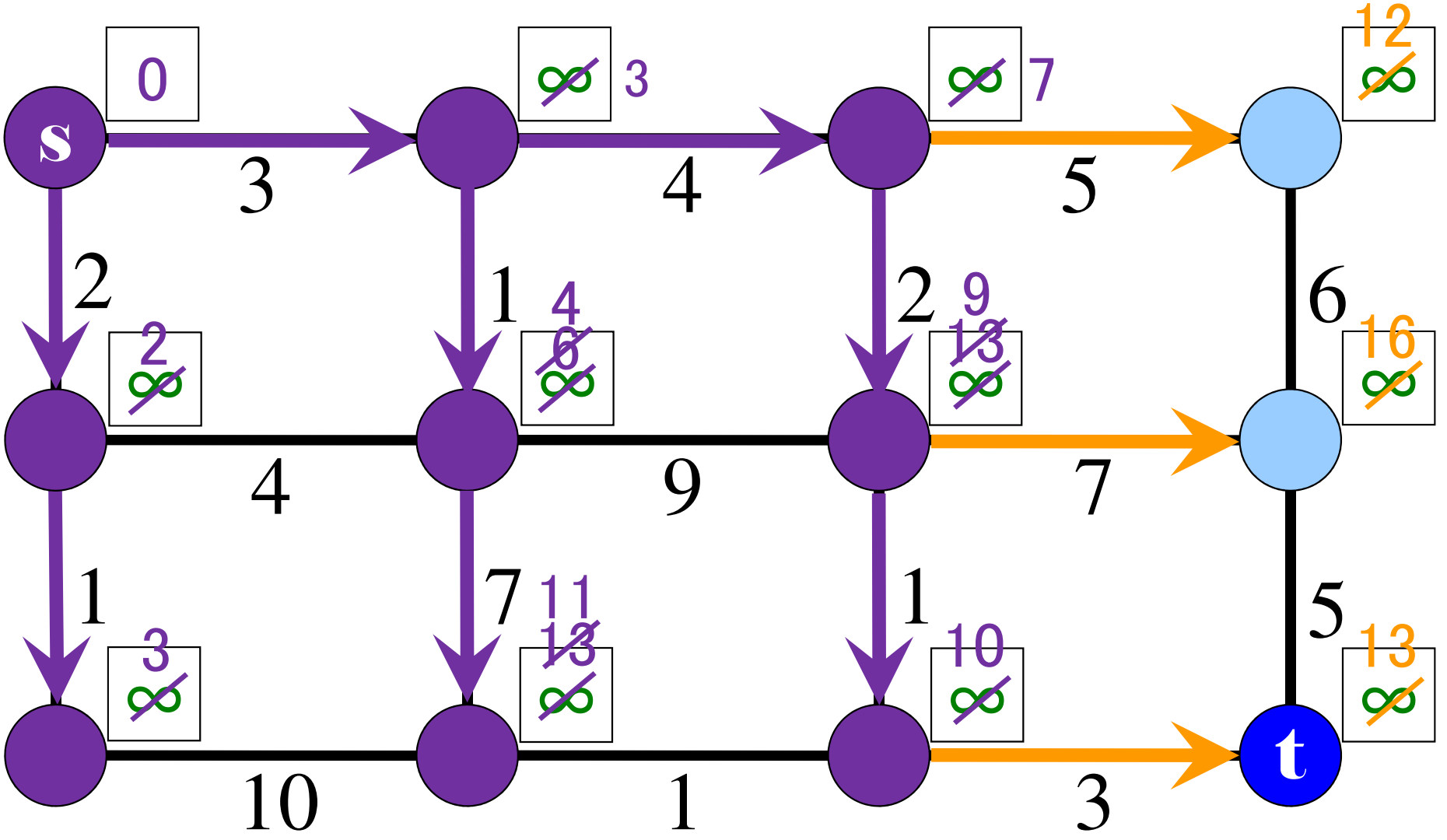


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

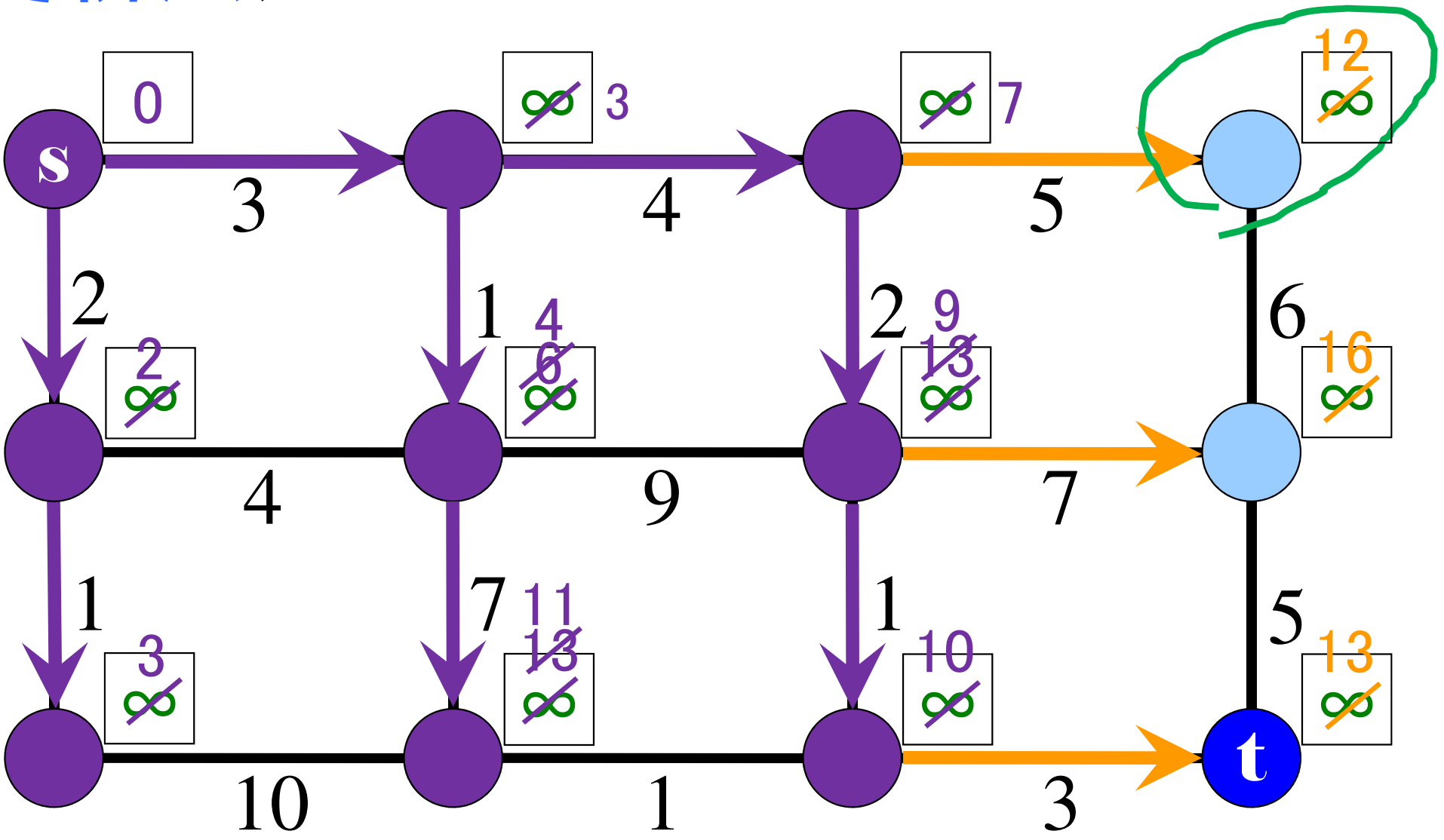
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



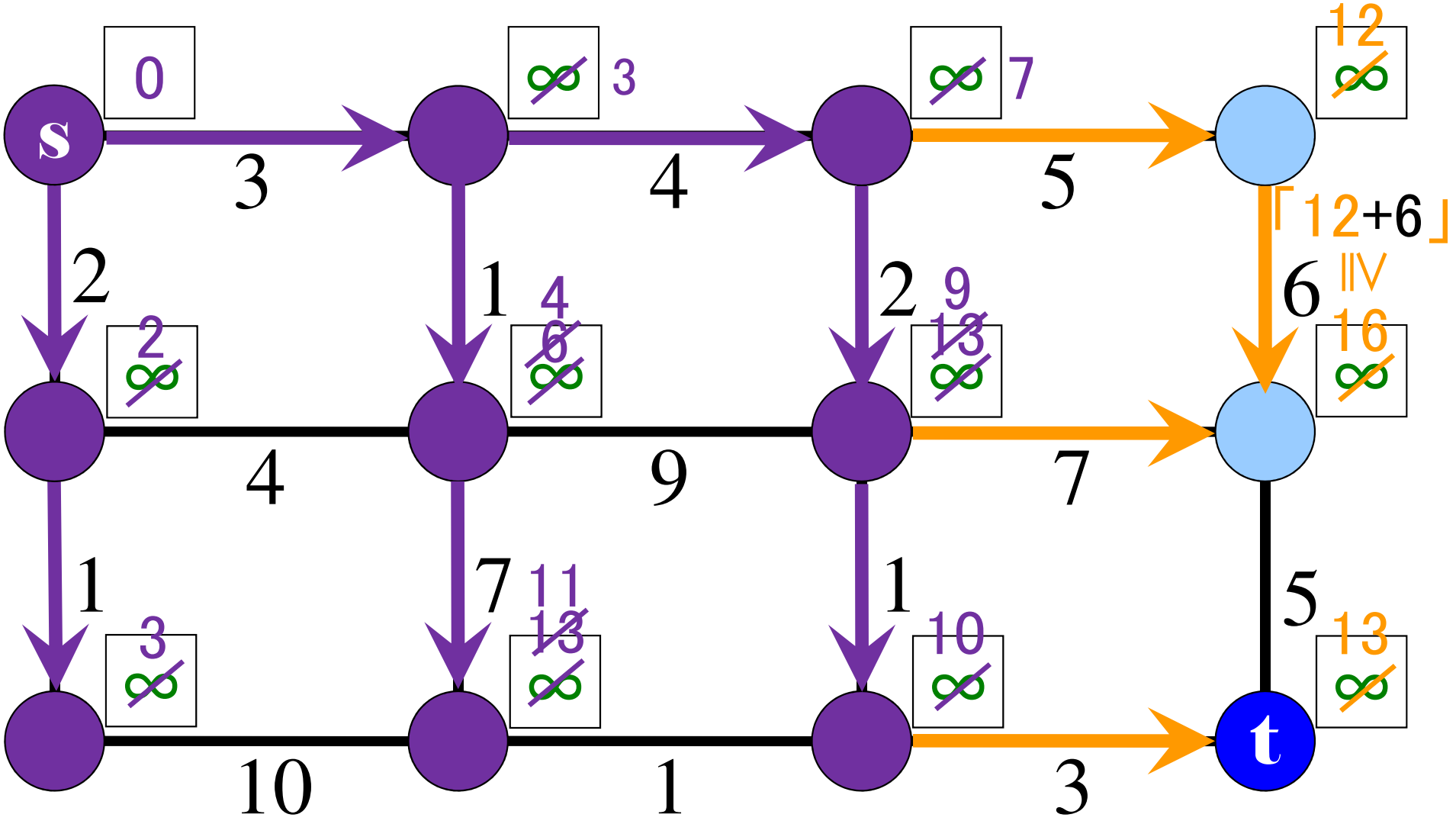
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

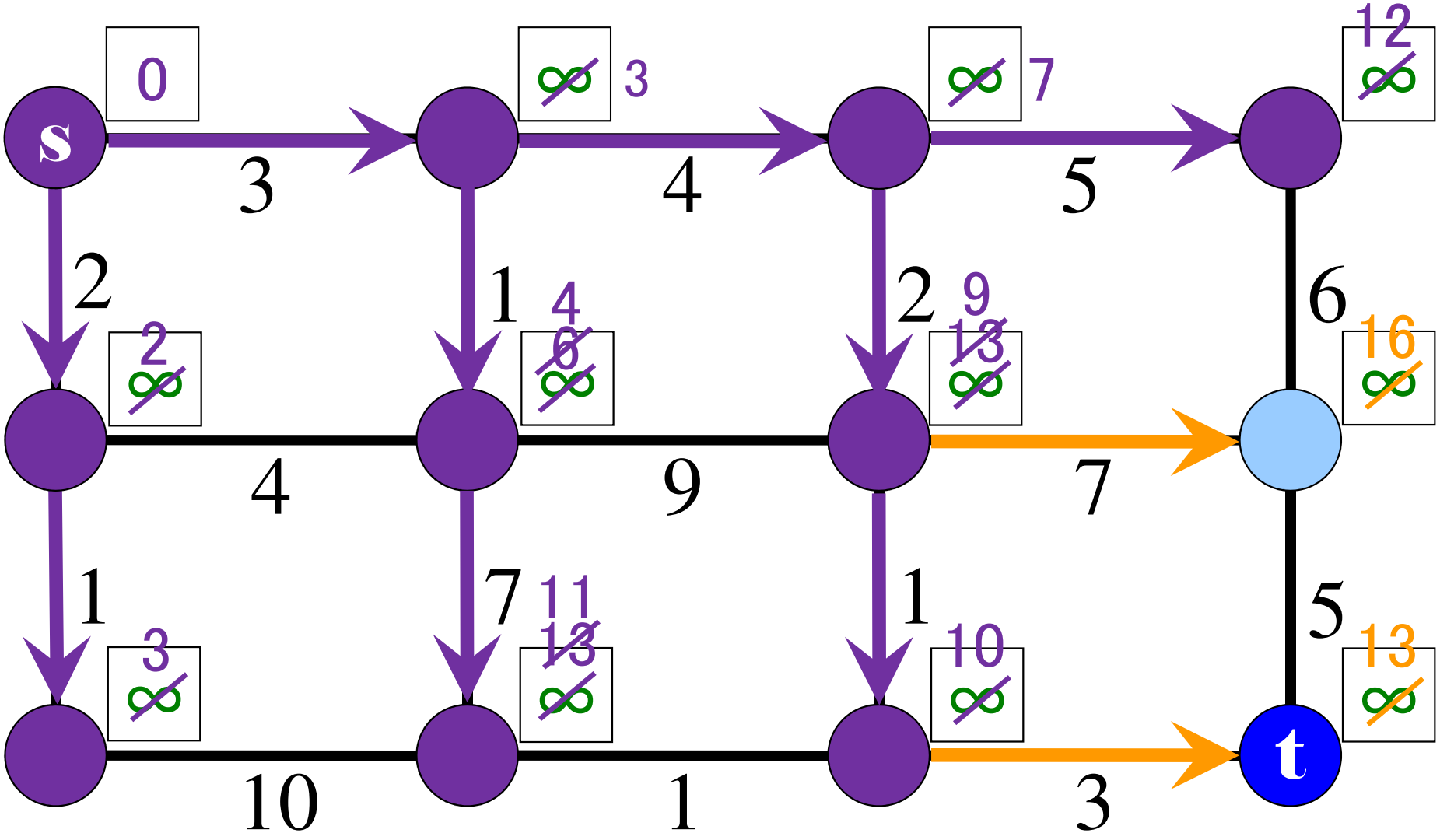


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

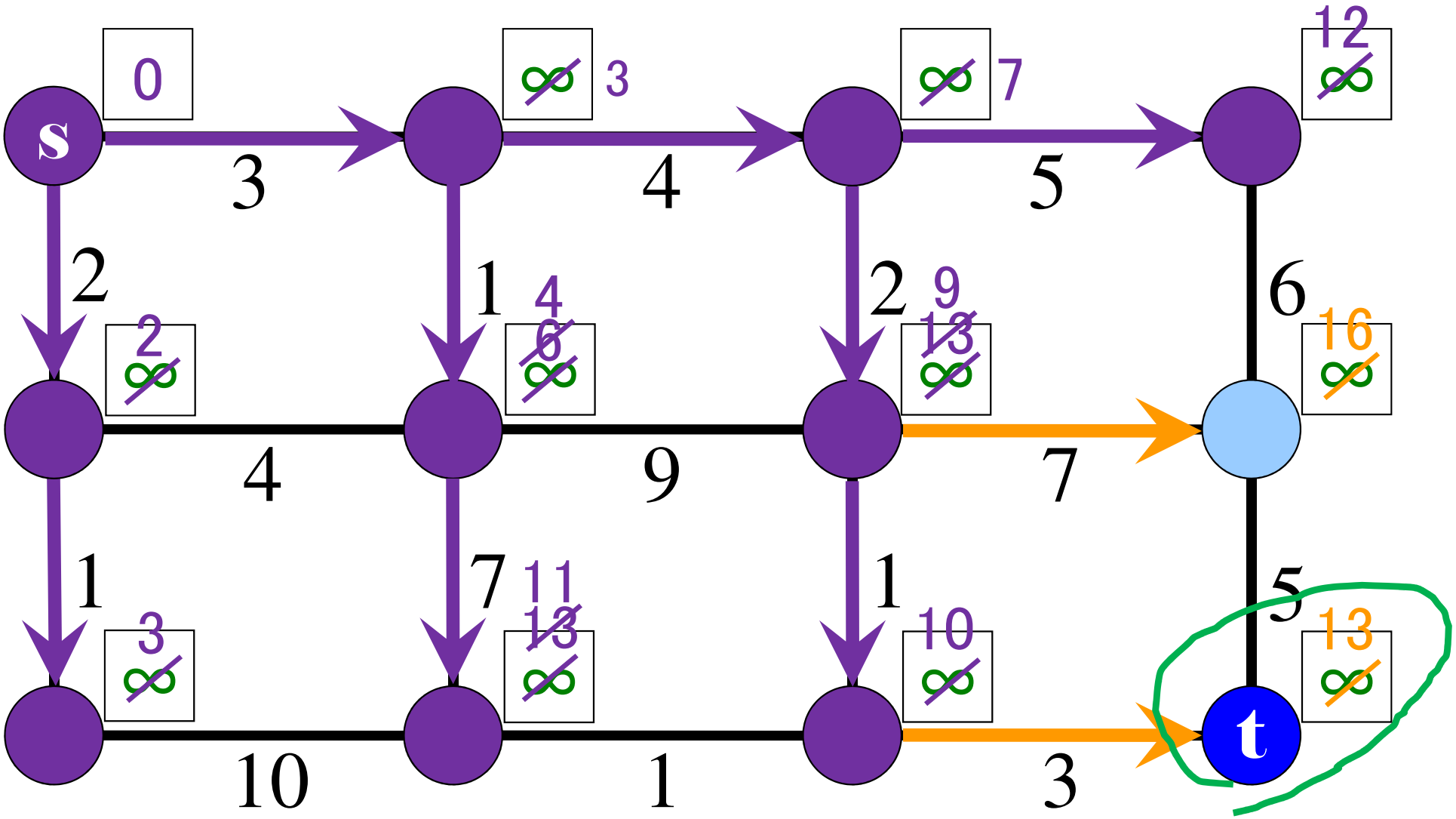
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



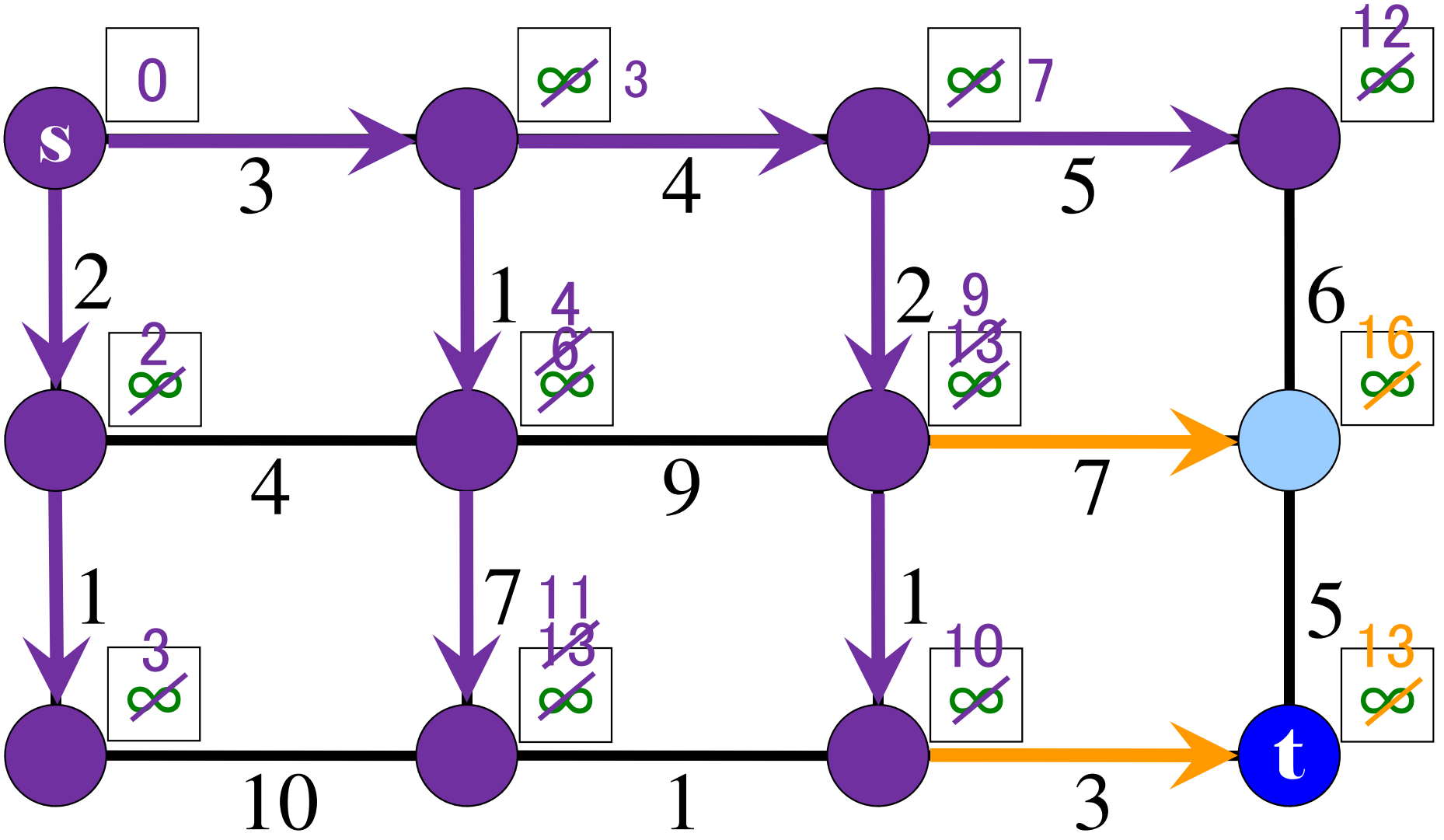
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

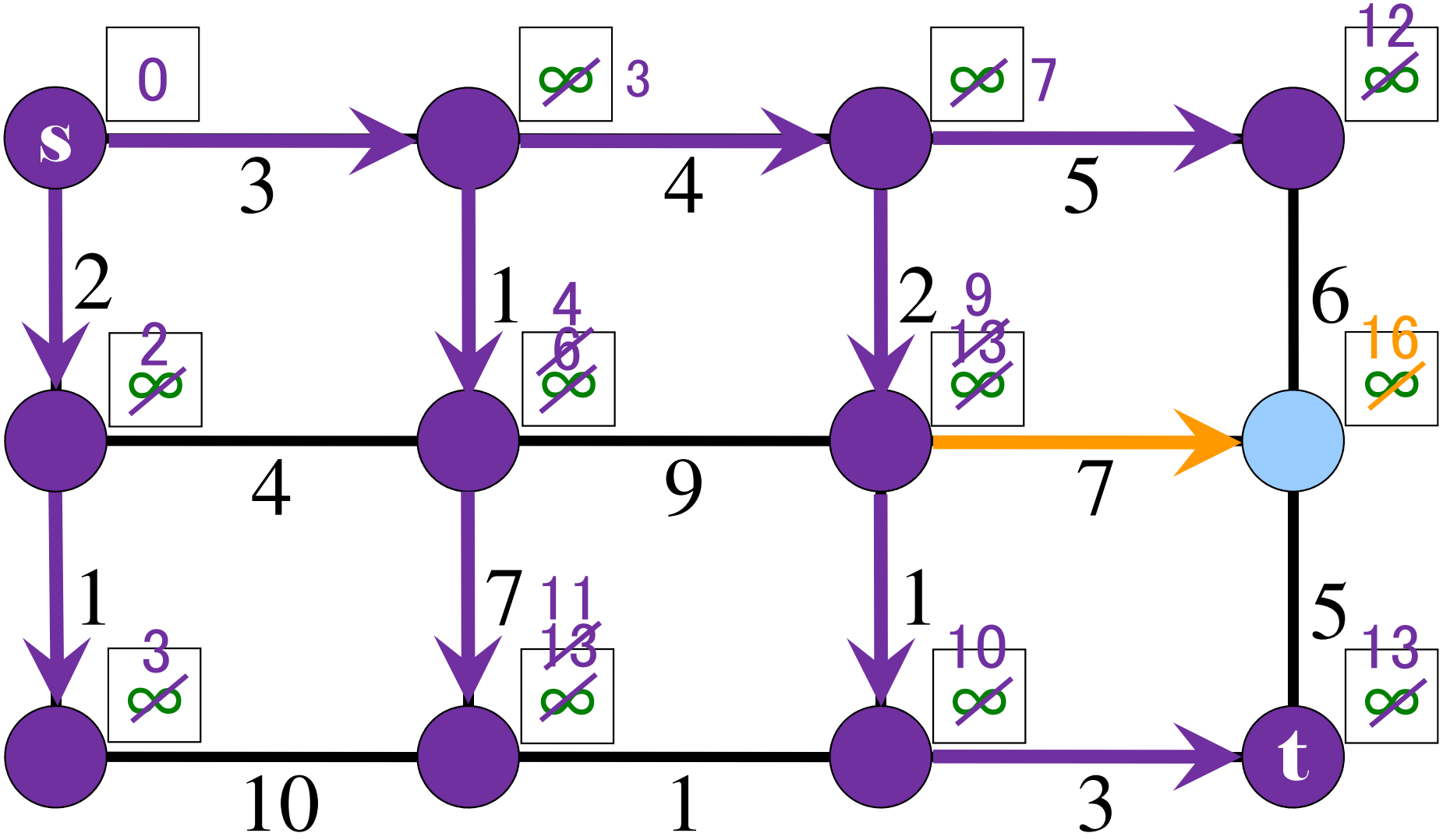


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

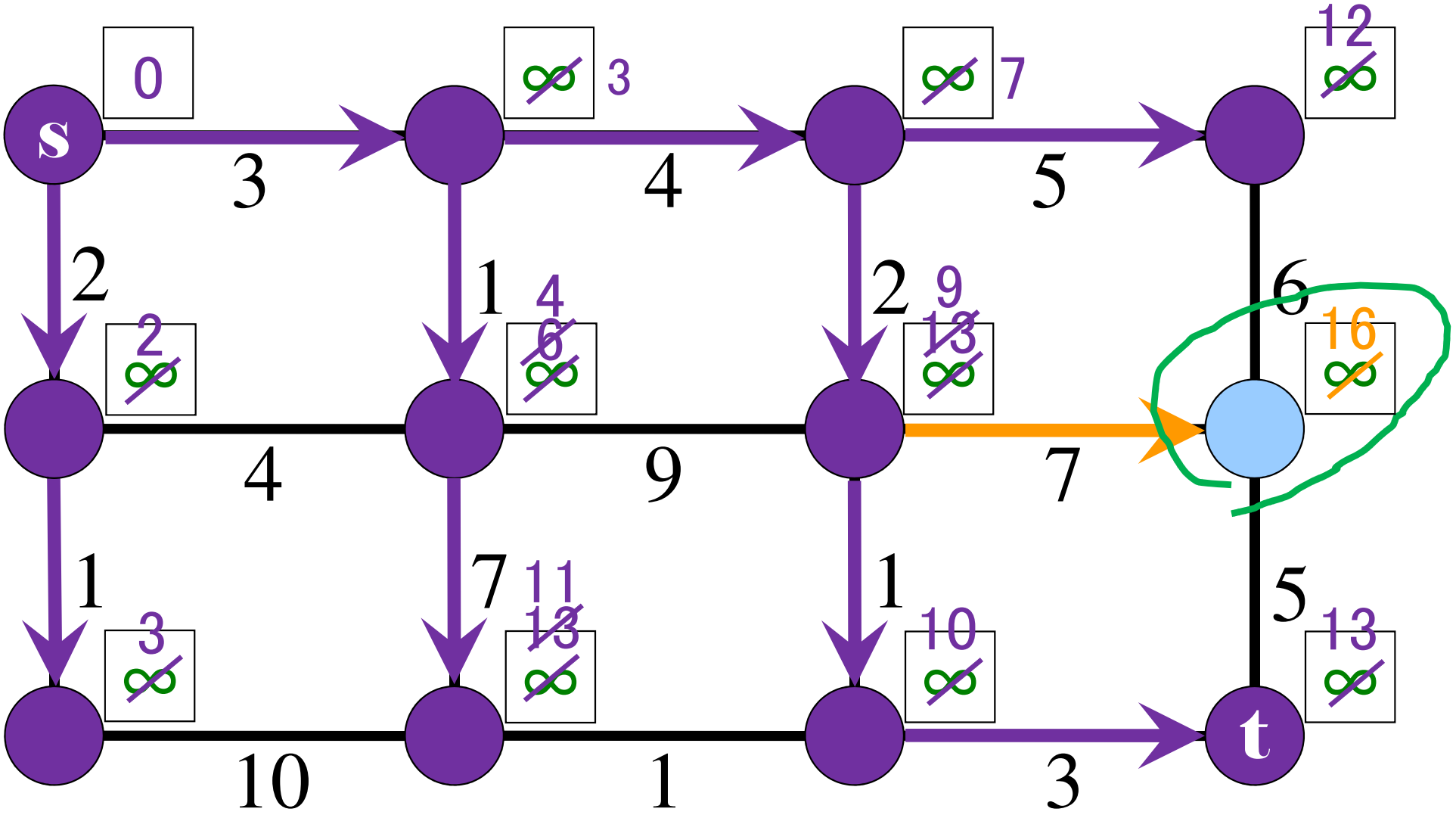
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



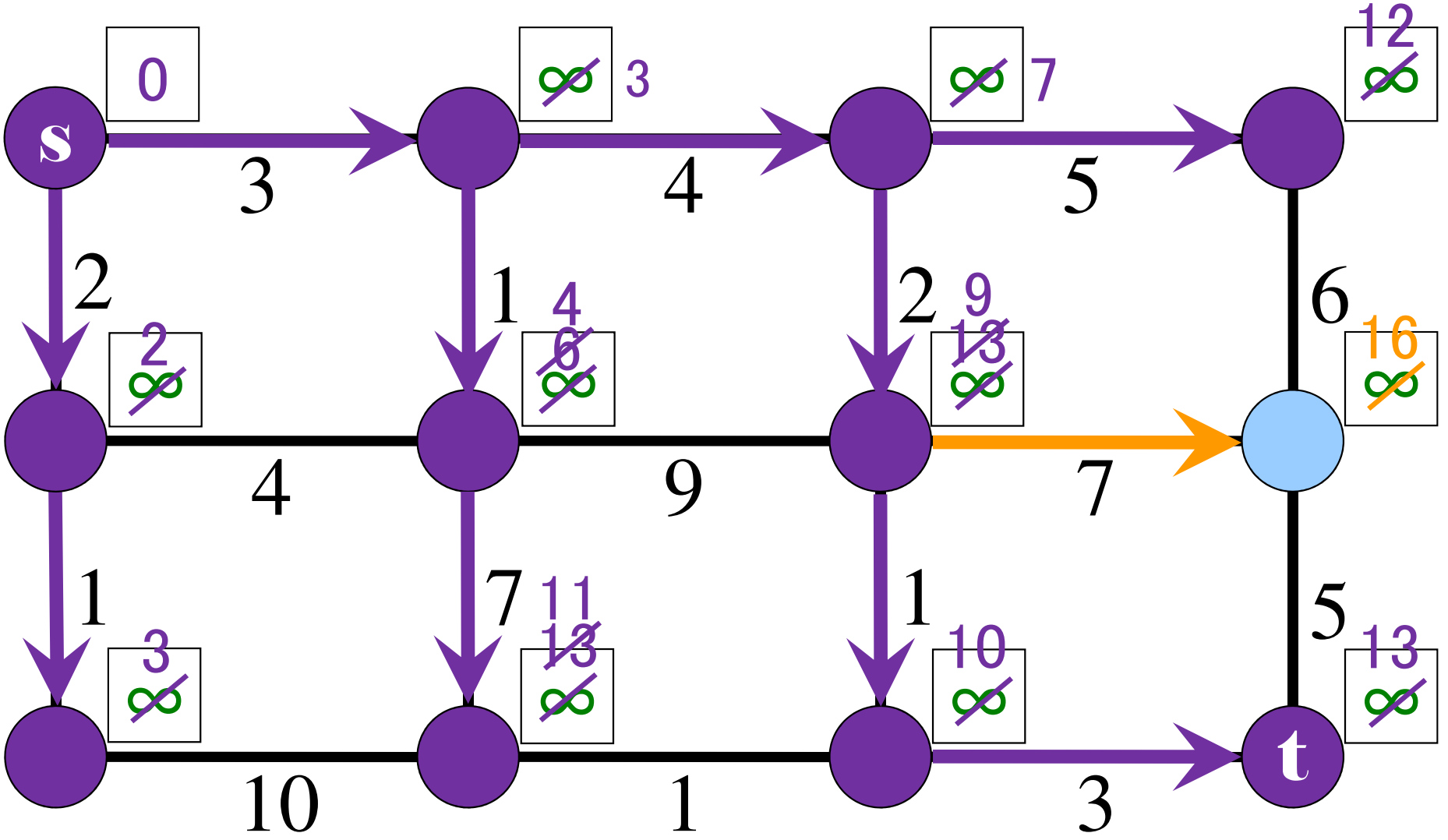
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

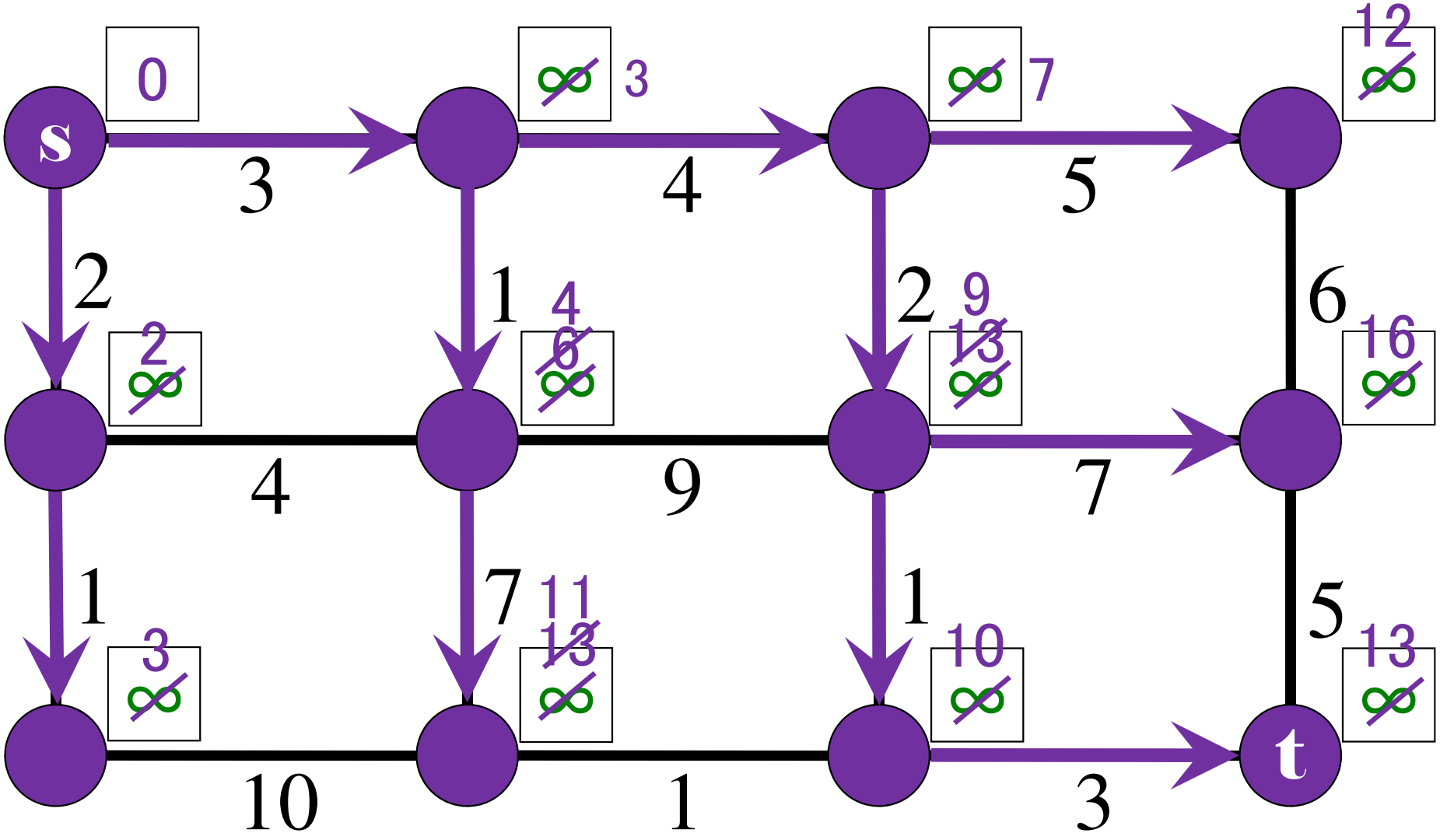


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

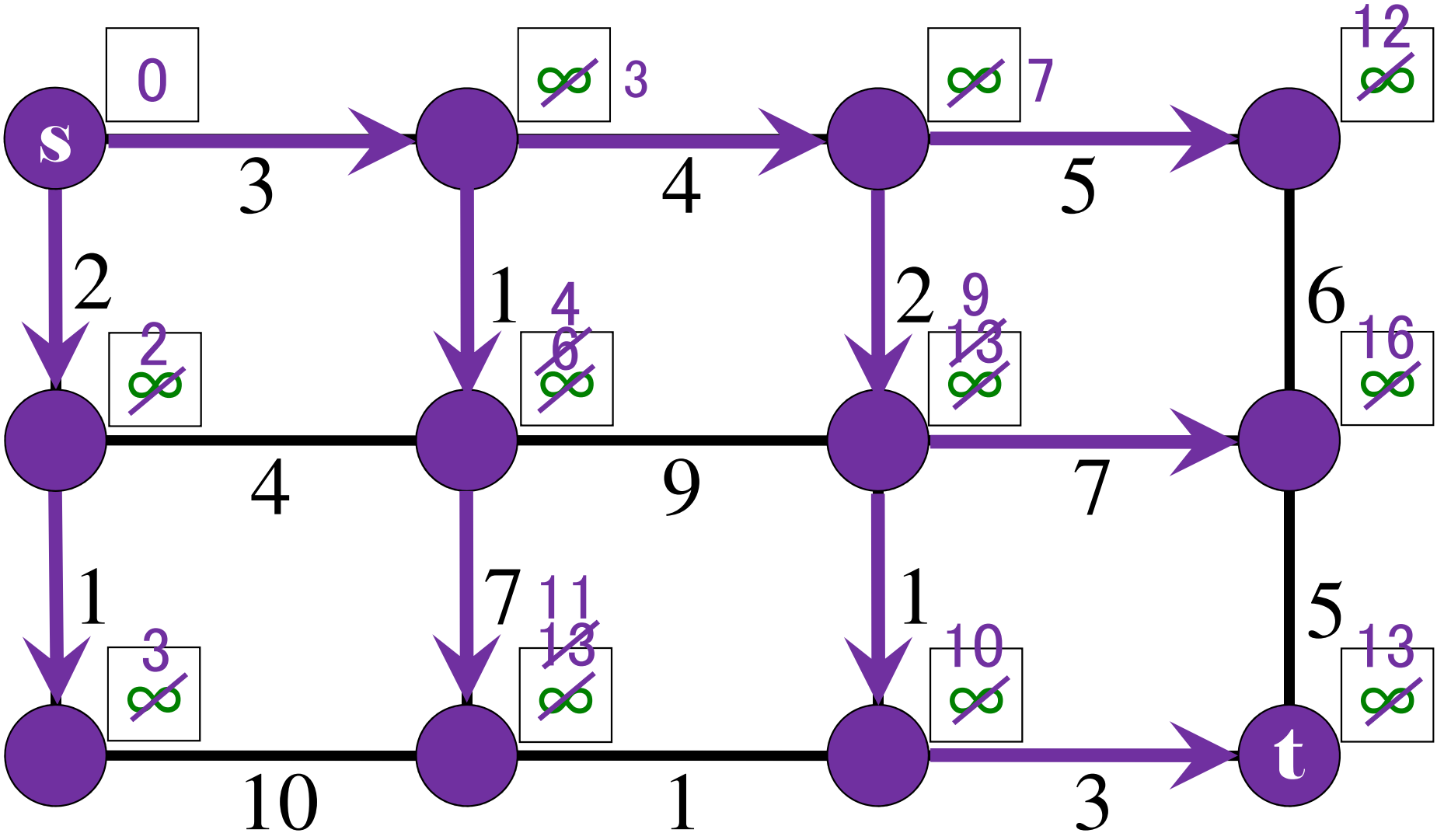
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



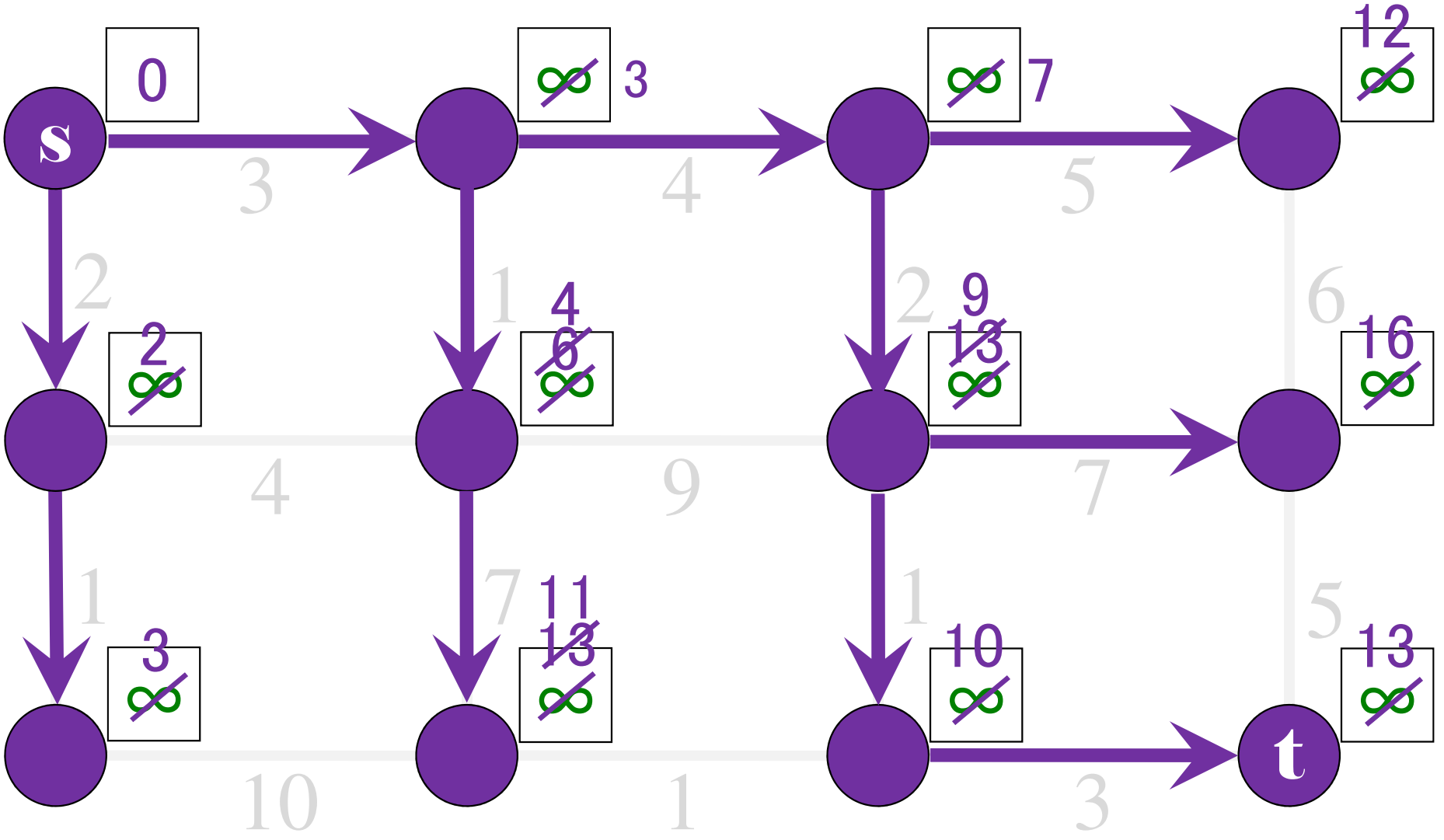
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



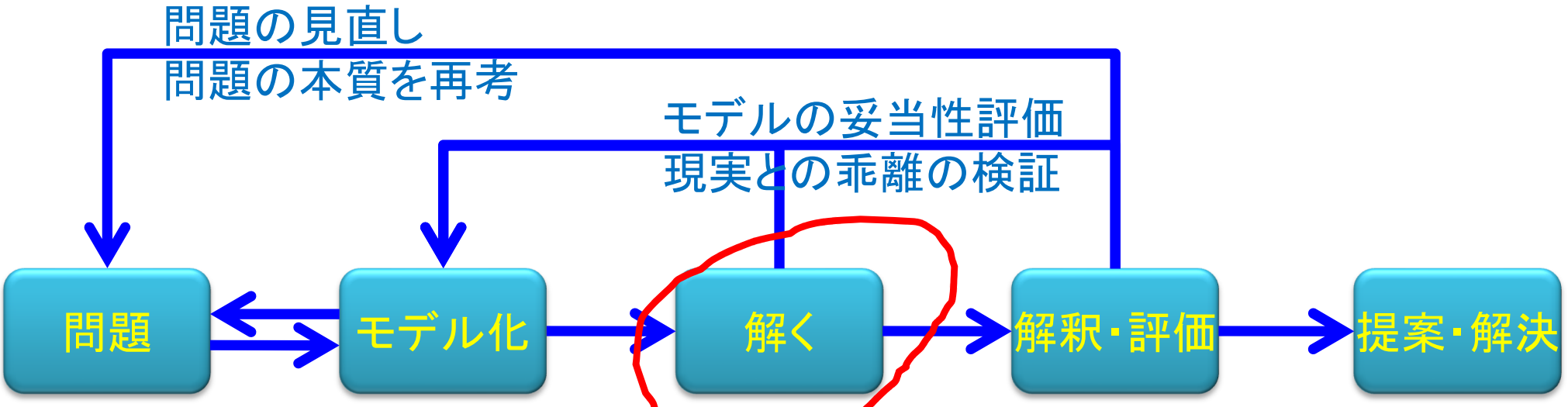
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

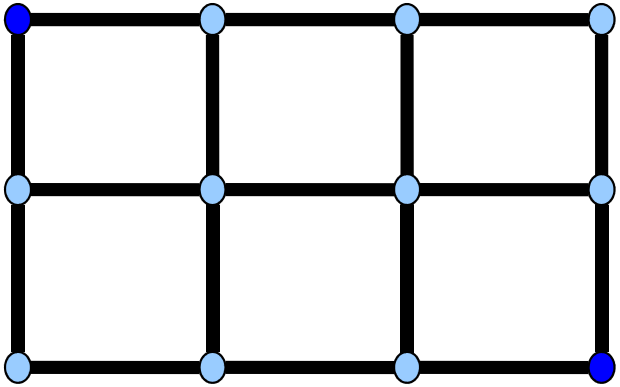
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

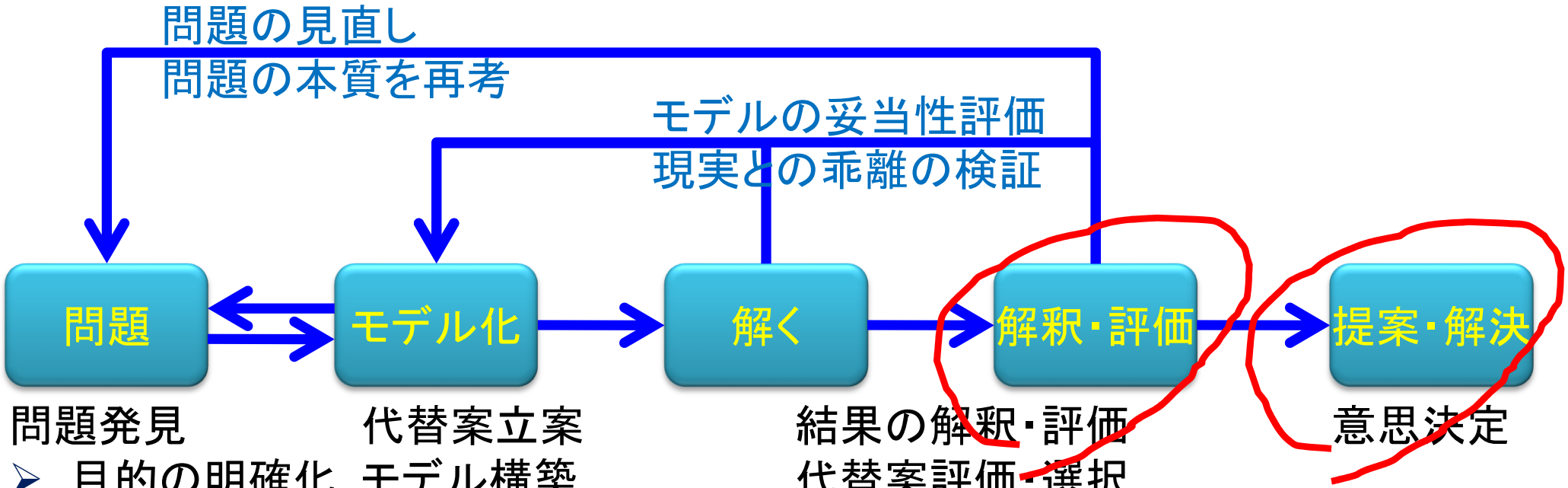
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

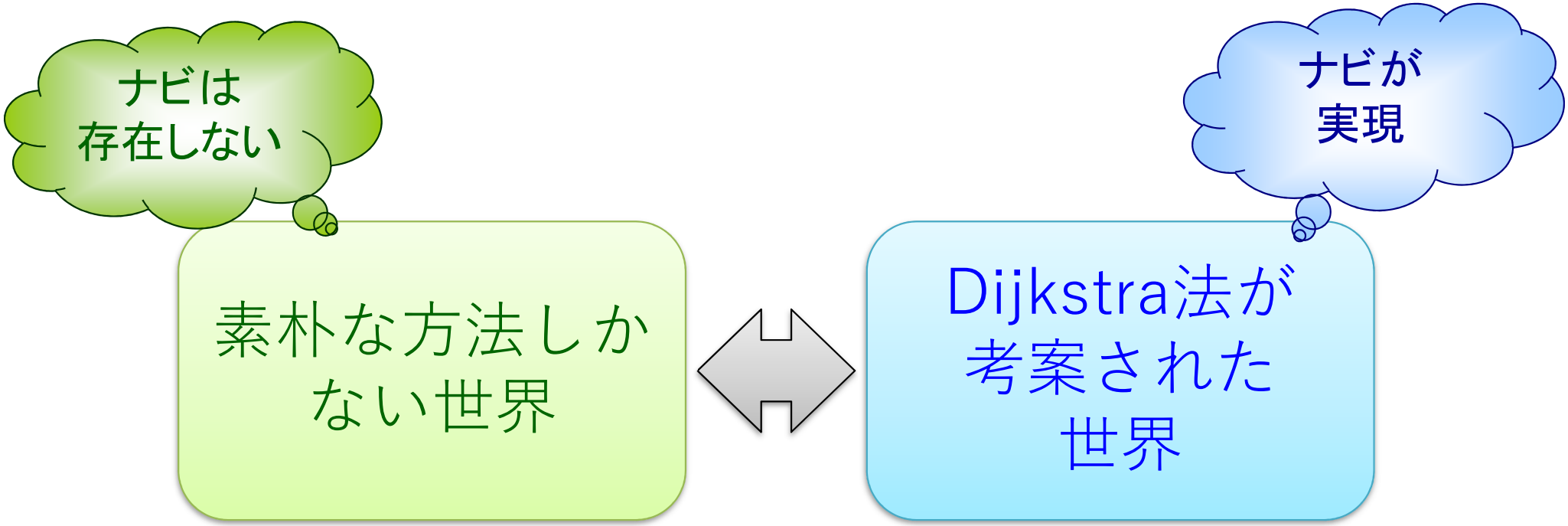
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

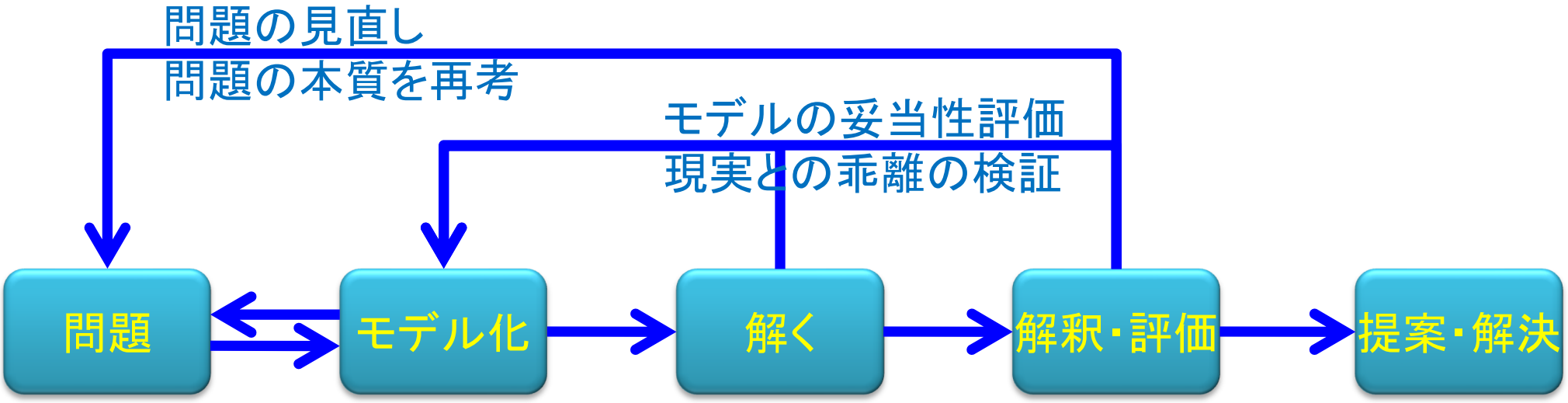
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



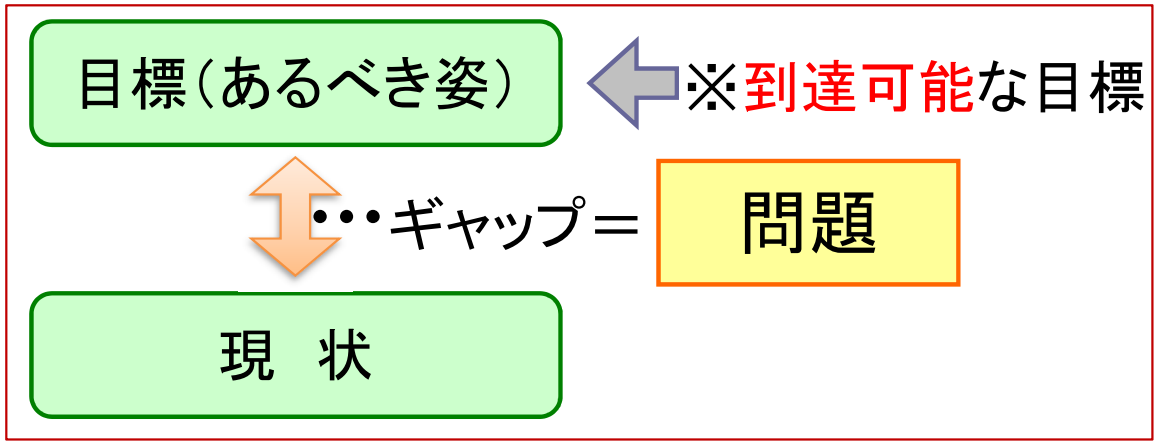
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

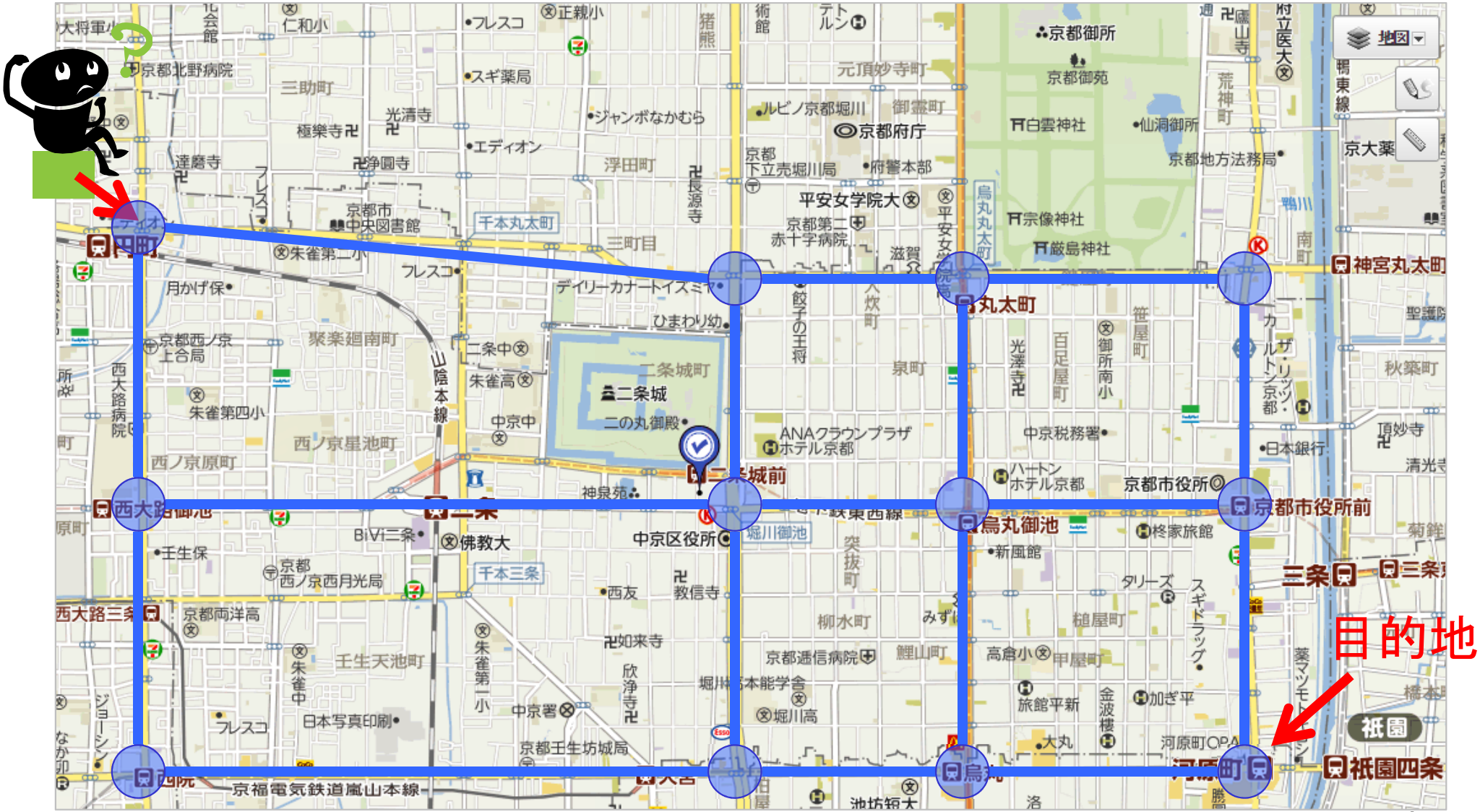
意思決定

問題の定義



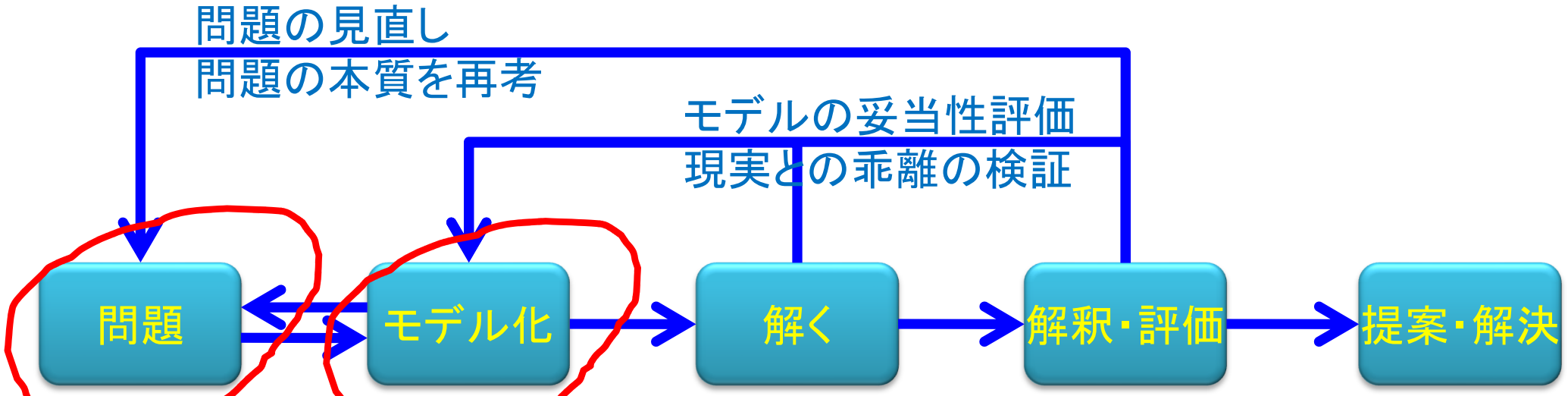
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

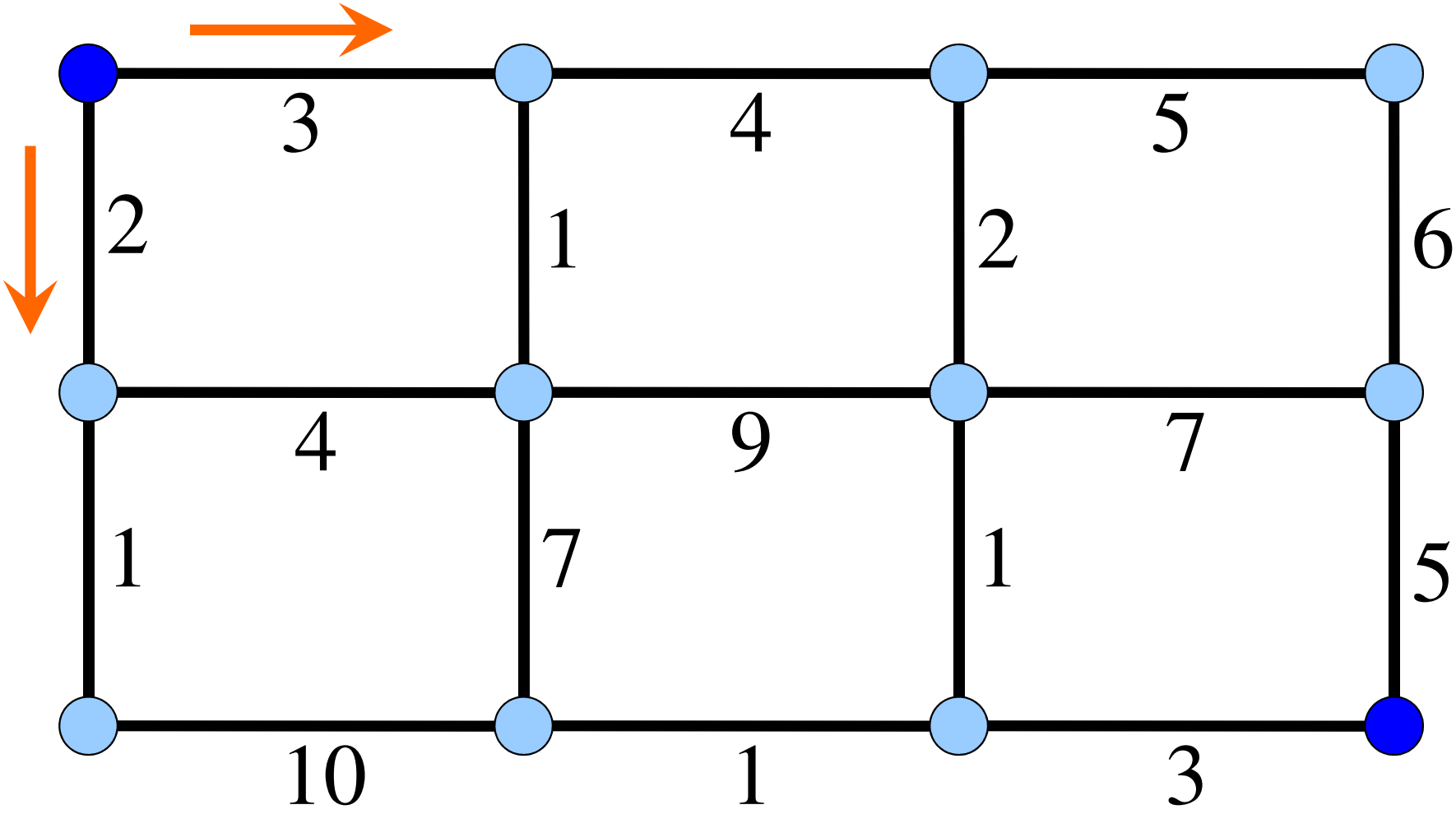
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

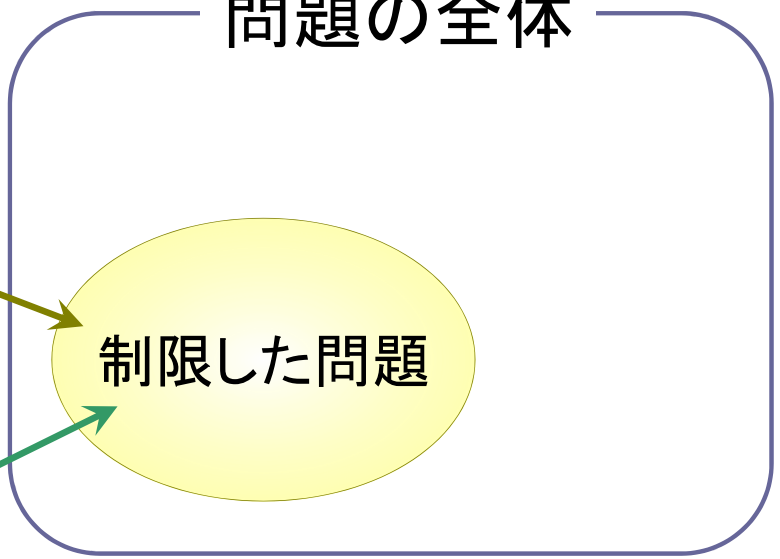


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

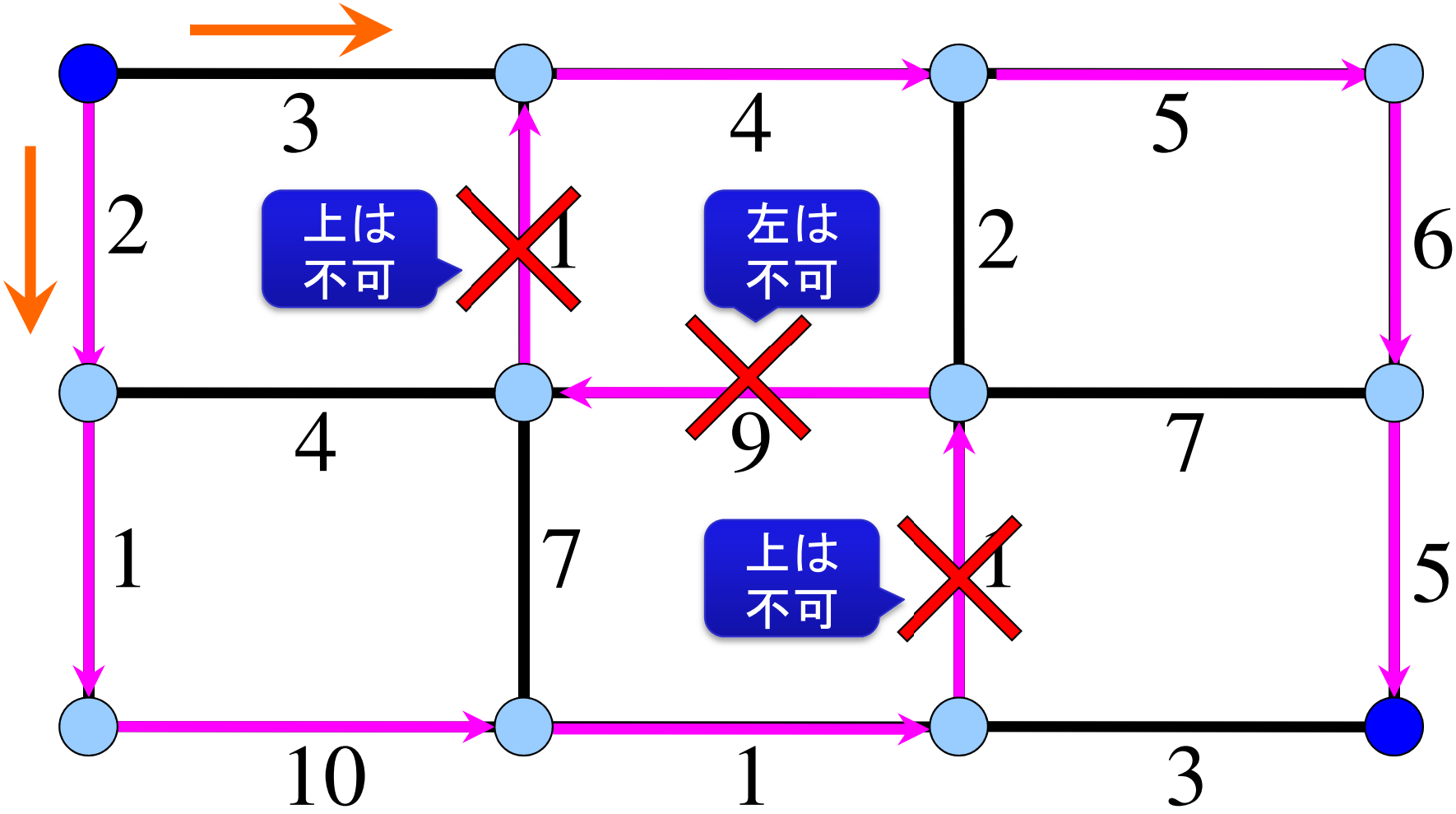
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

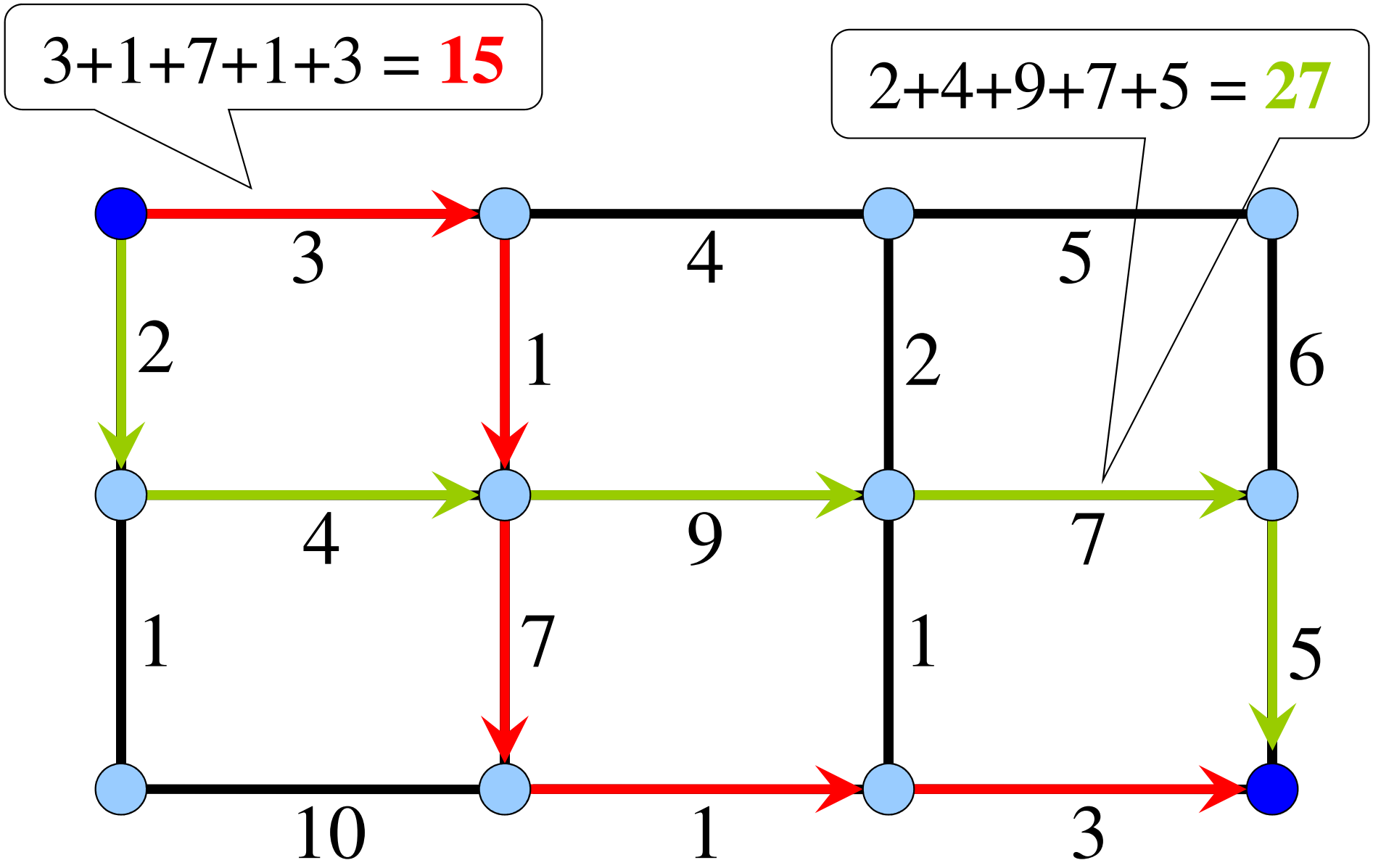
難しいなら易しくすればいいのさ！

制限した問題

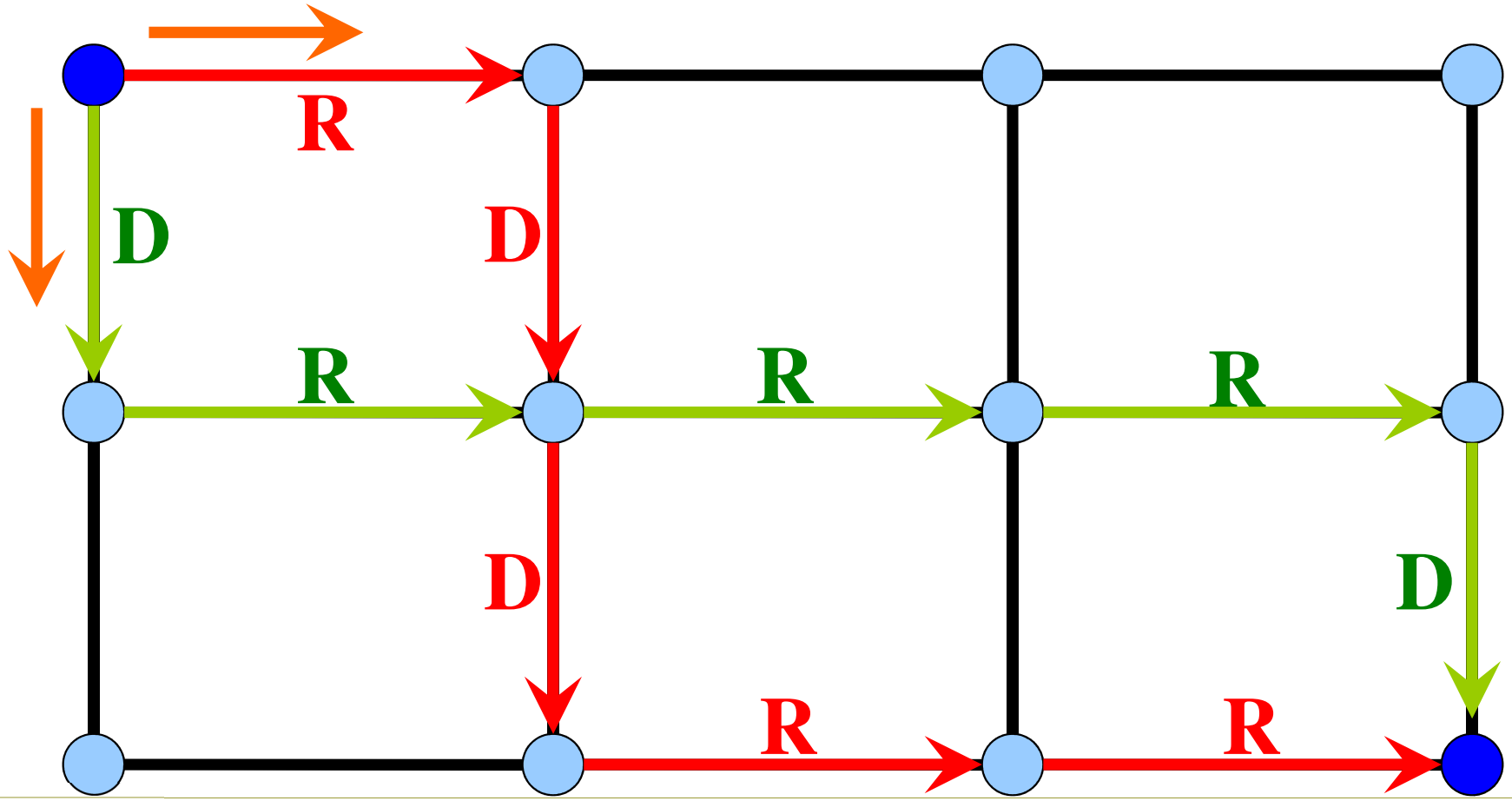
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



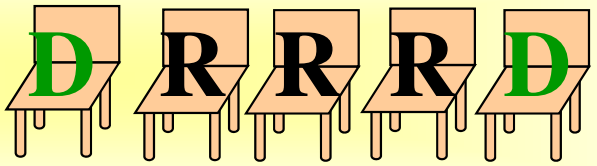
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



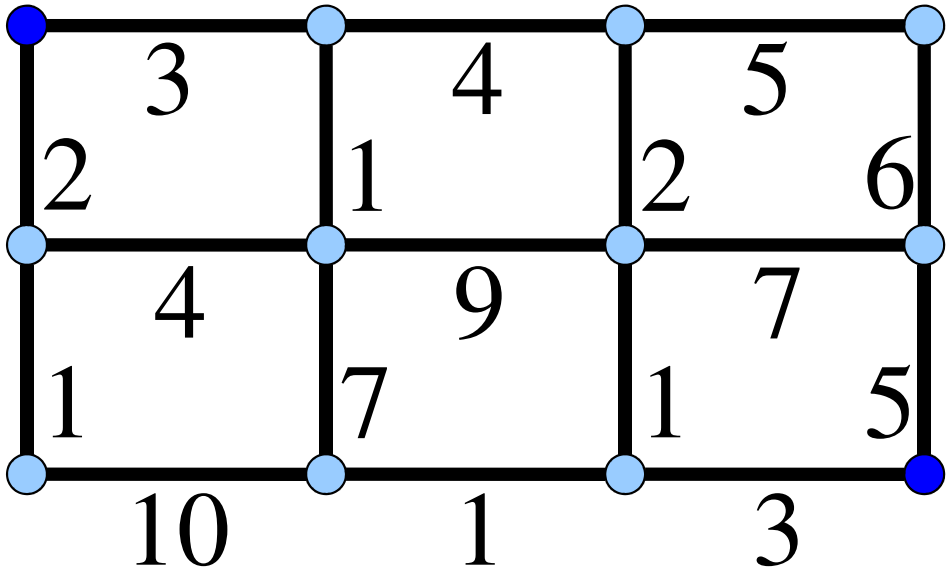
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

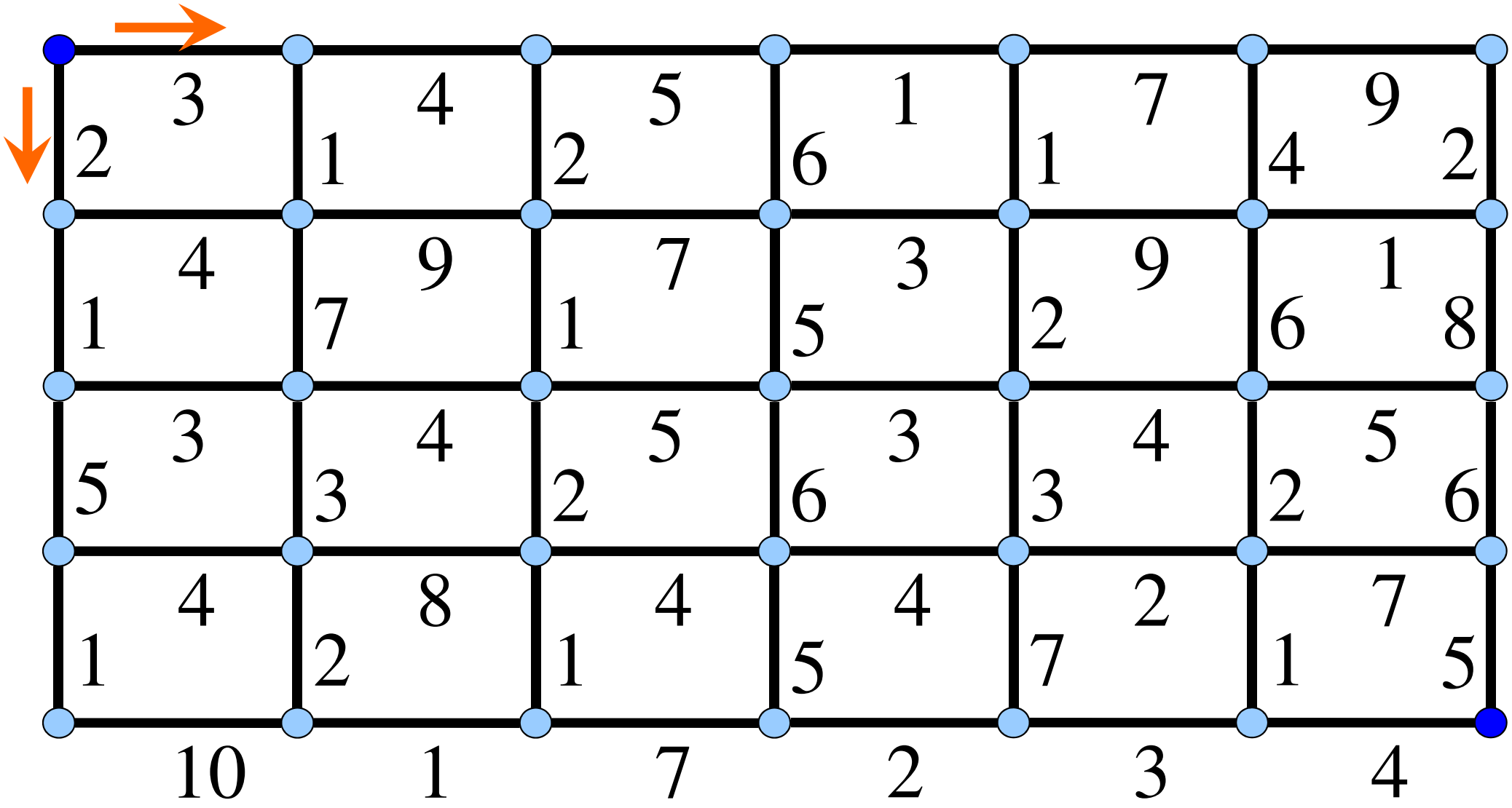
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

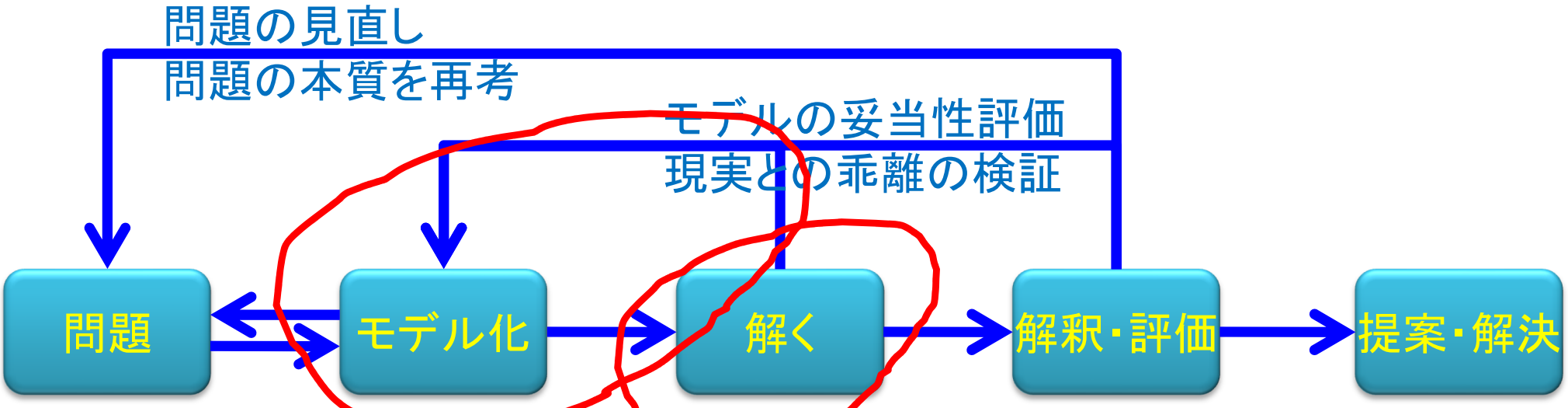
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

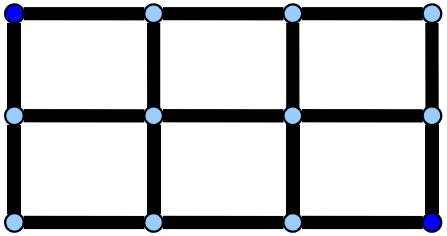
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

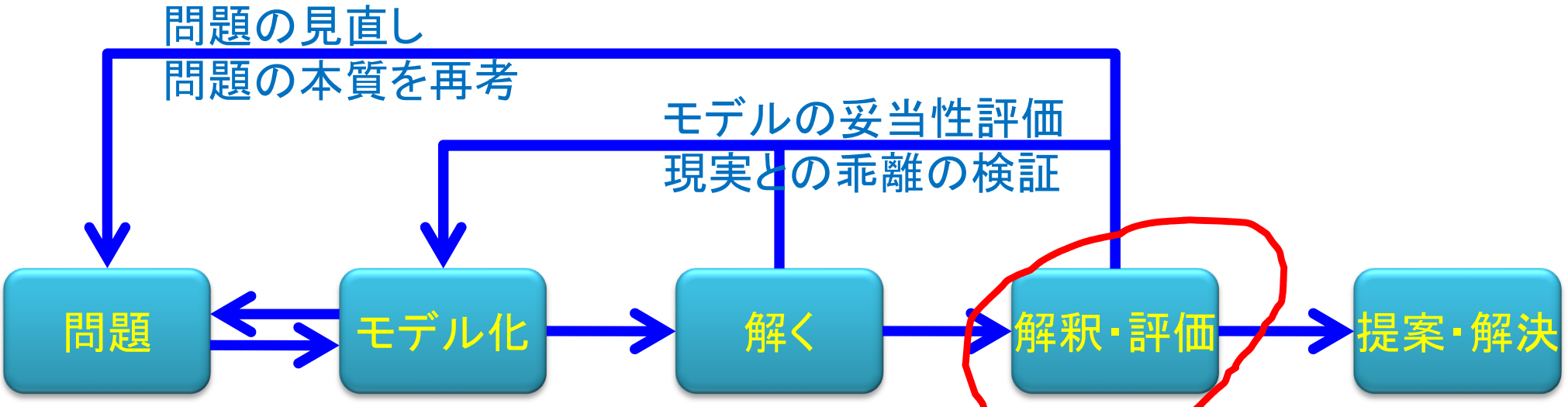
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

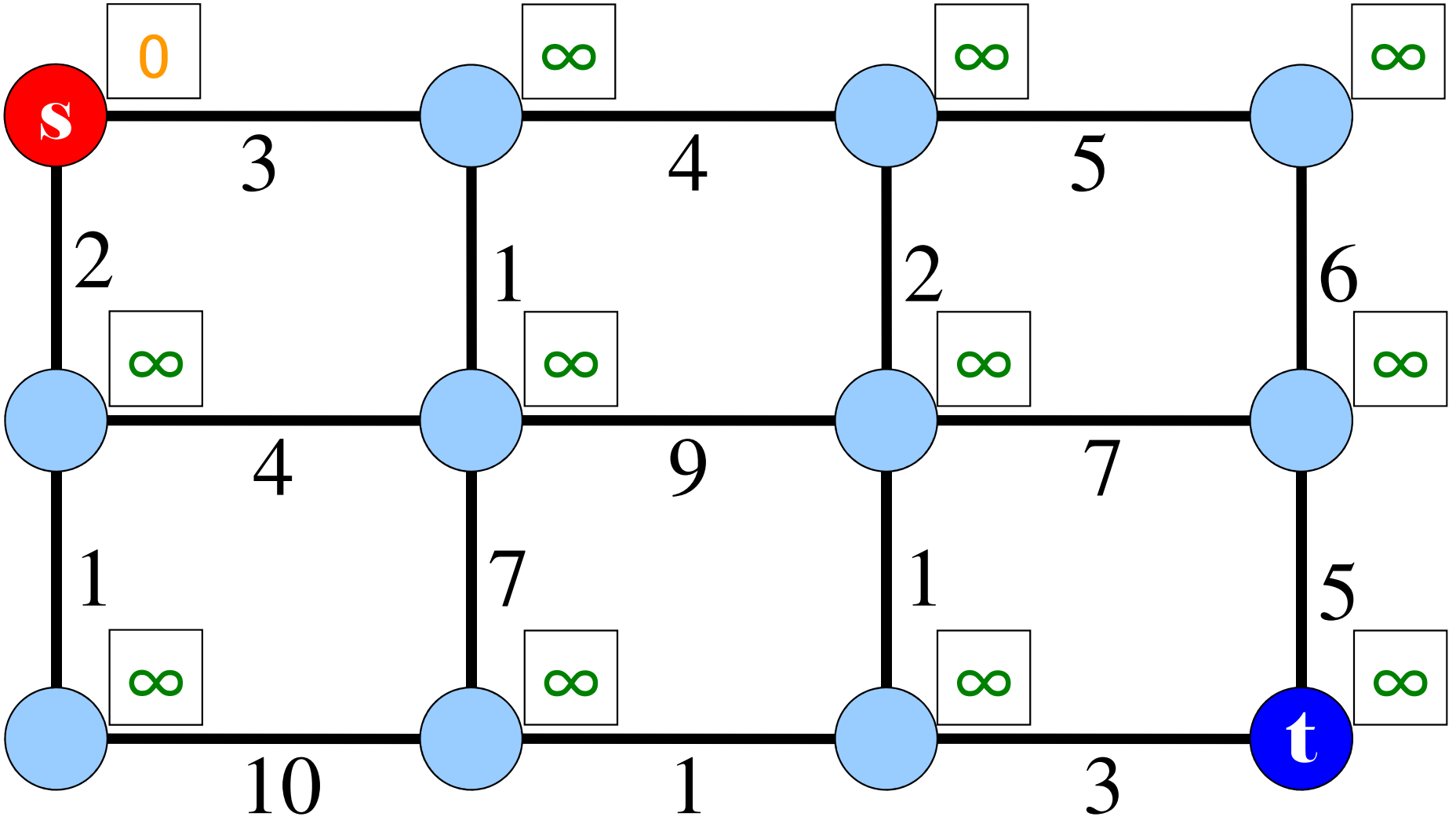
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

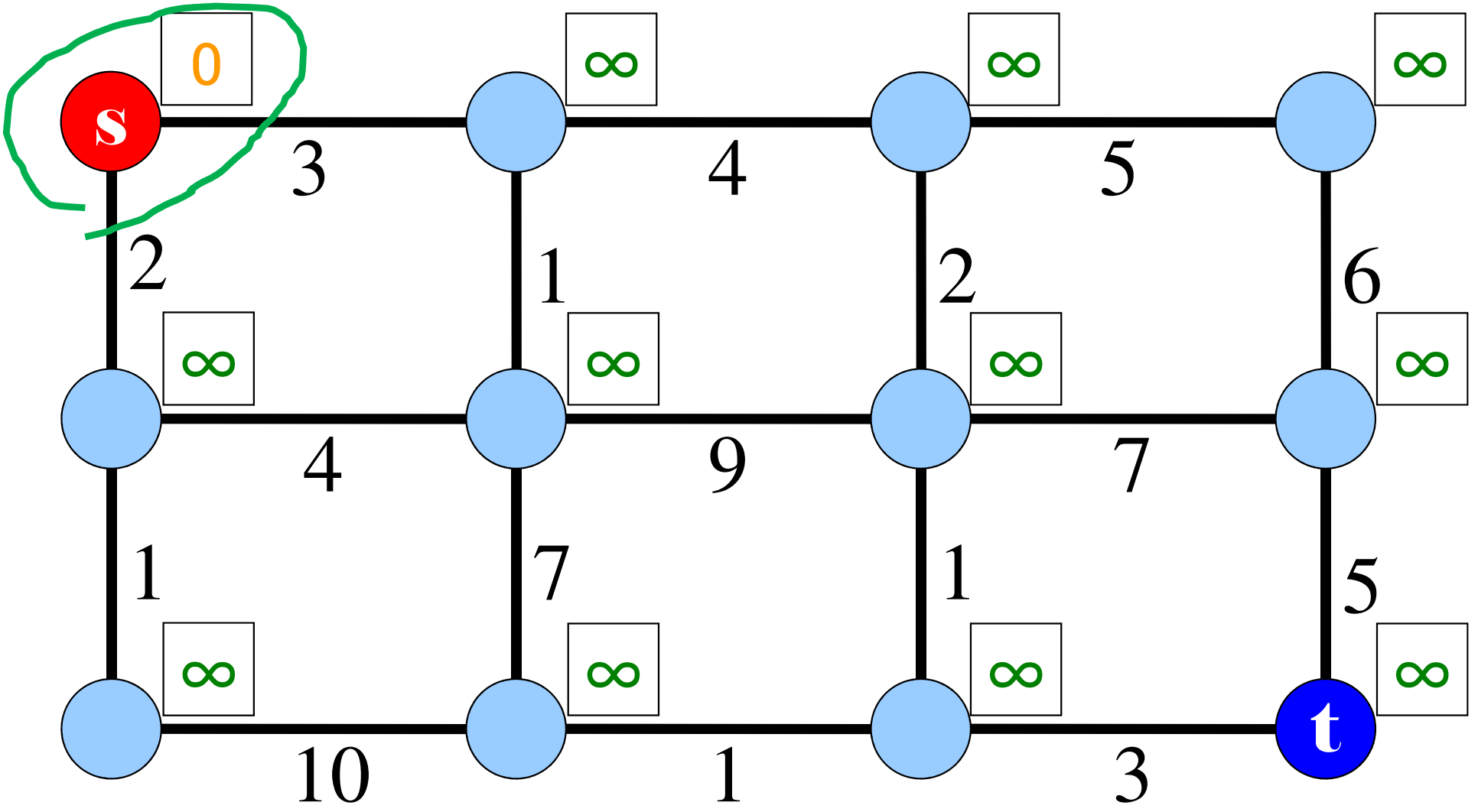
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



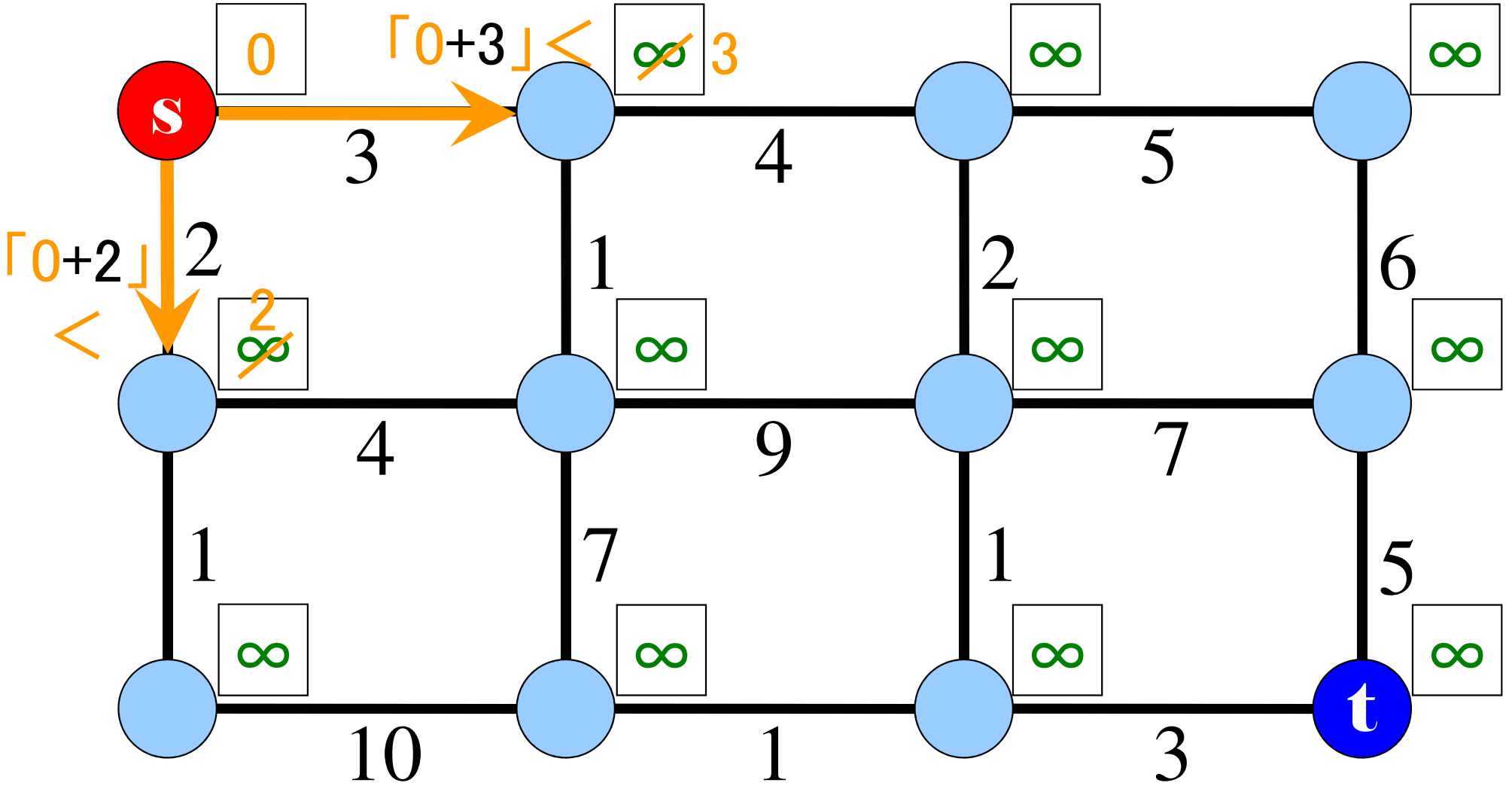
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

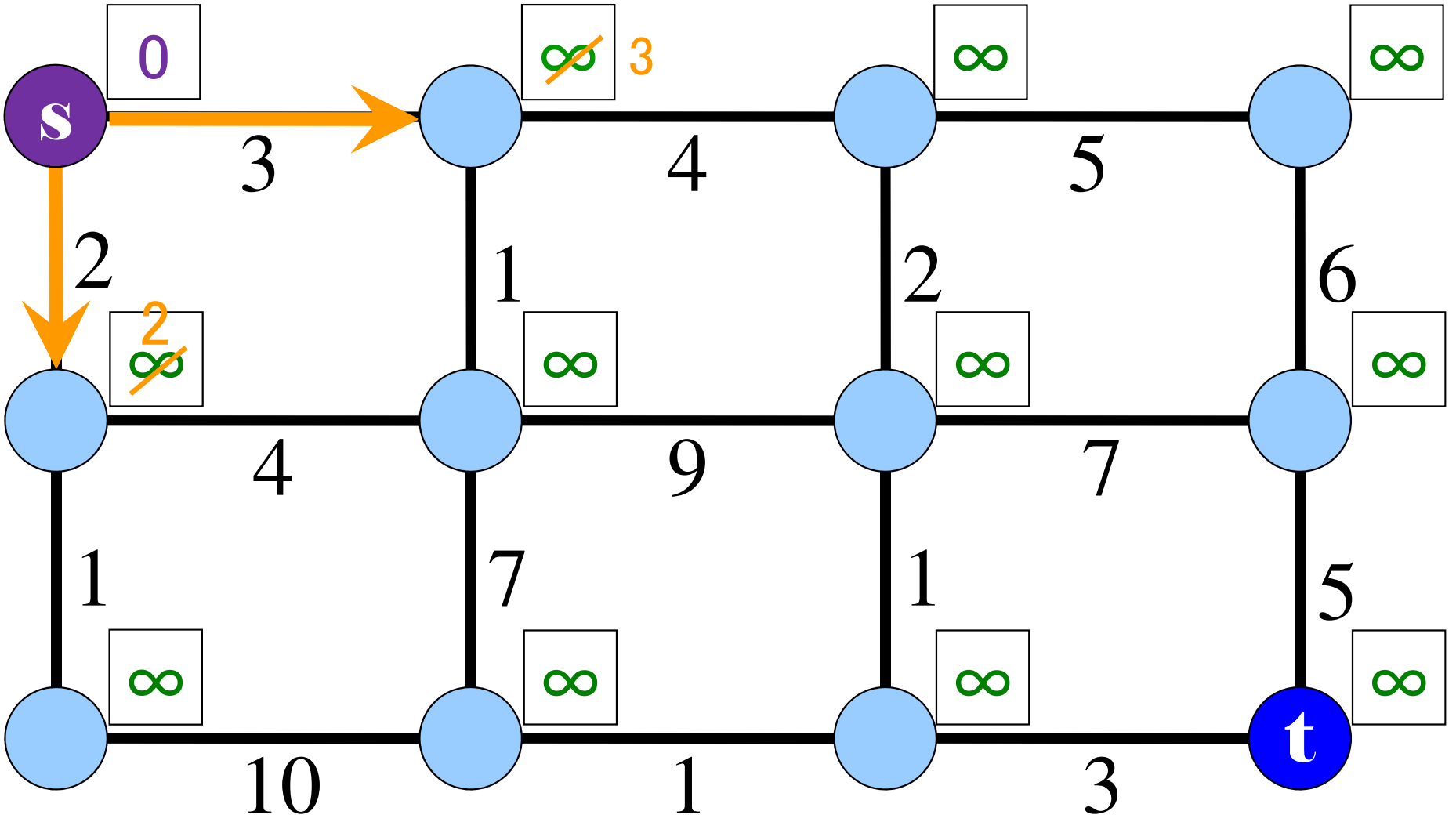


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

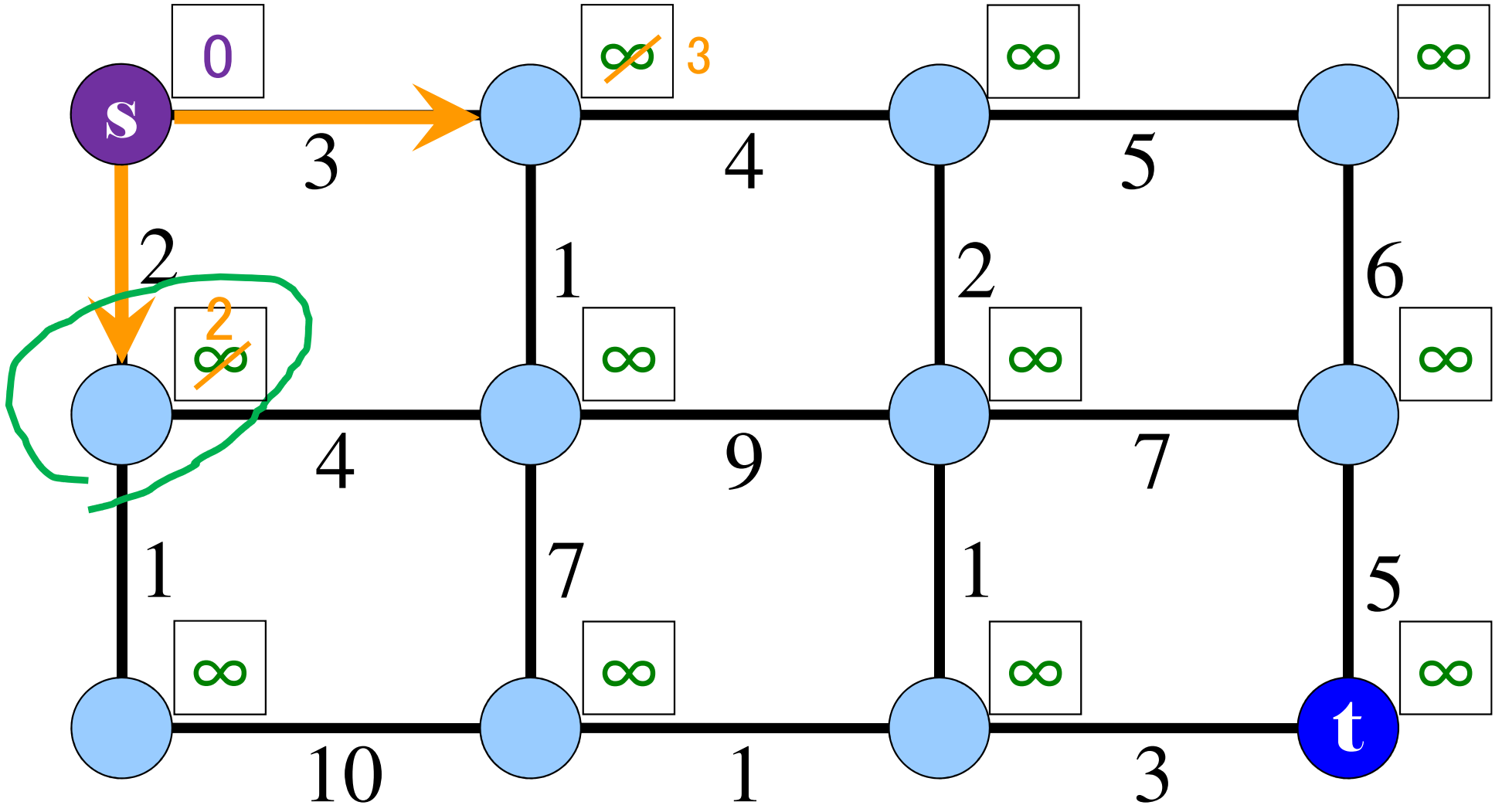
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



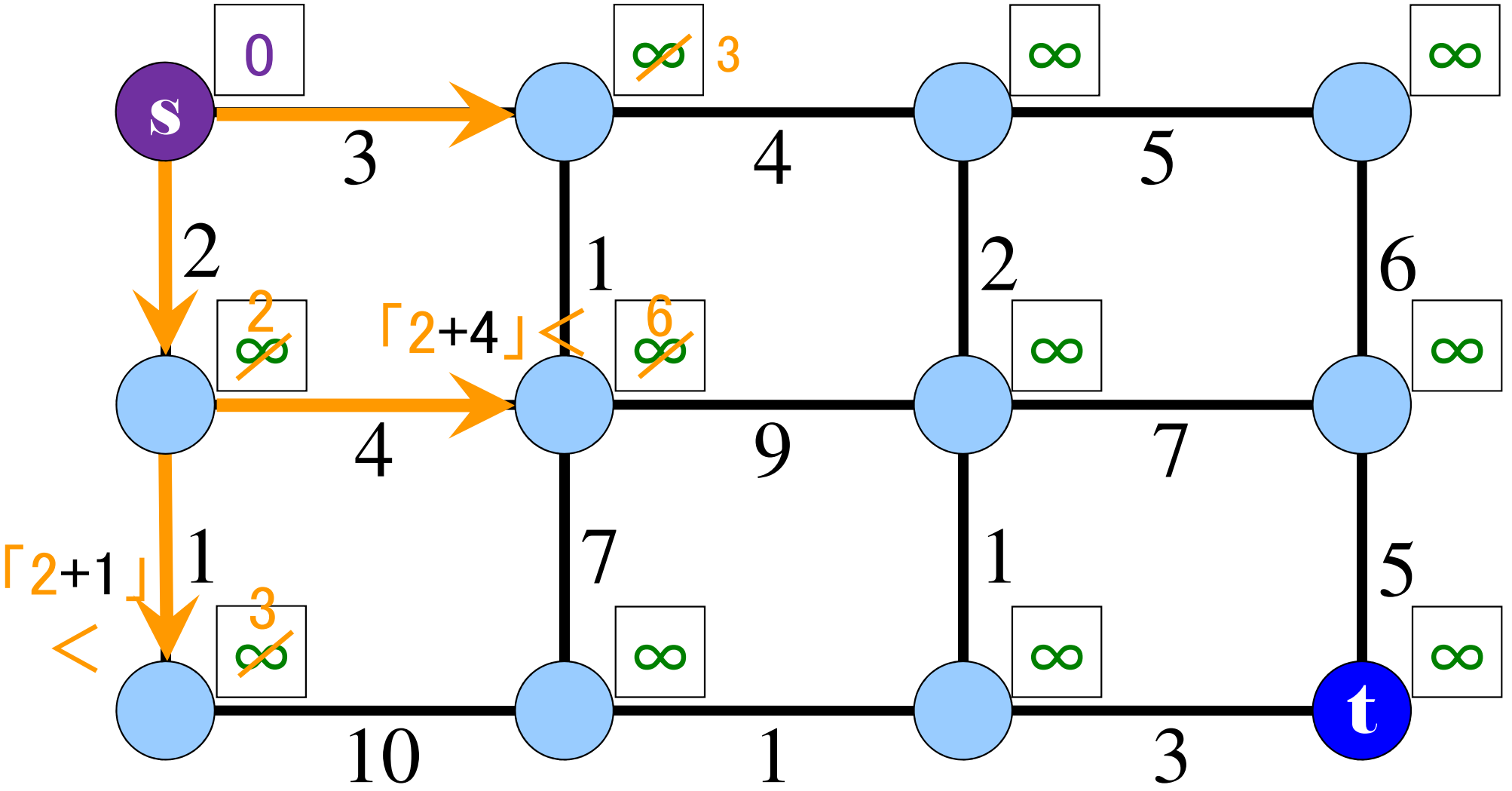
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

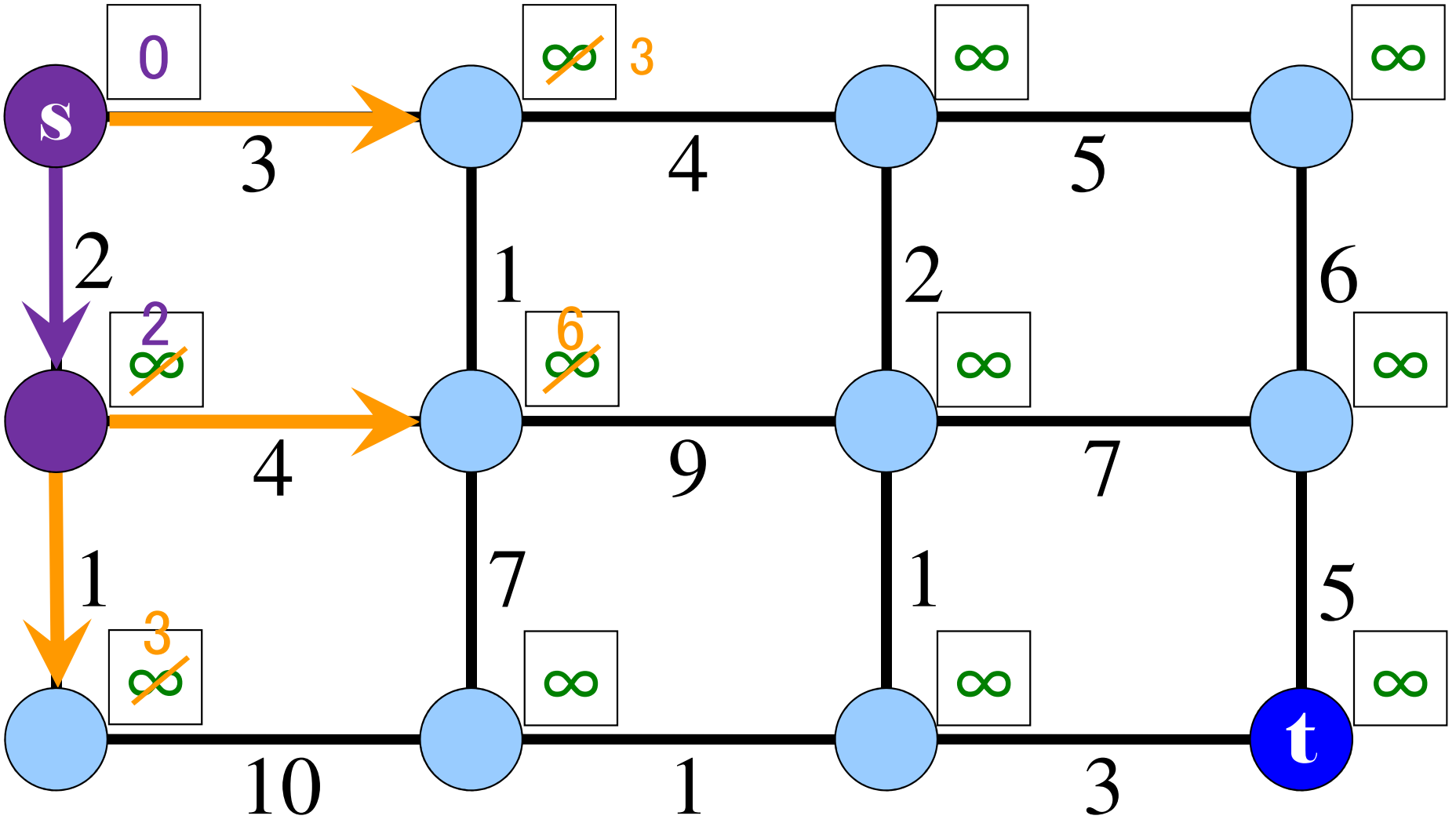


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

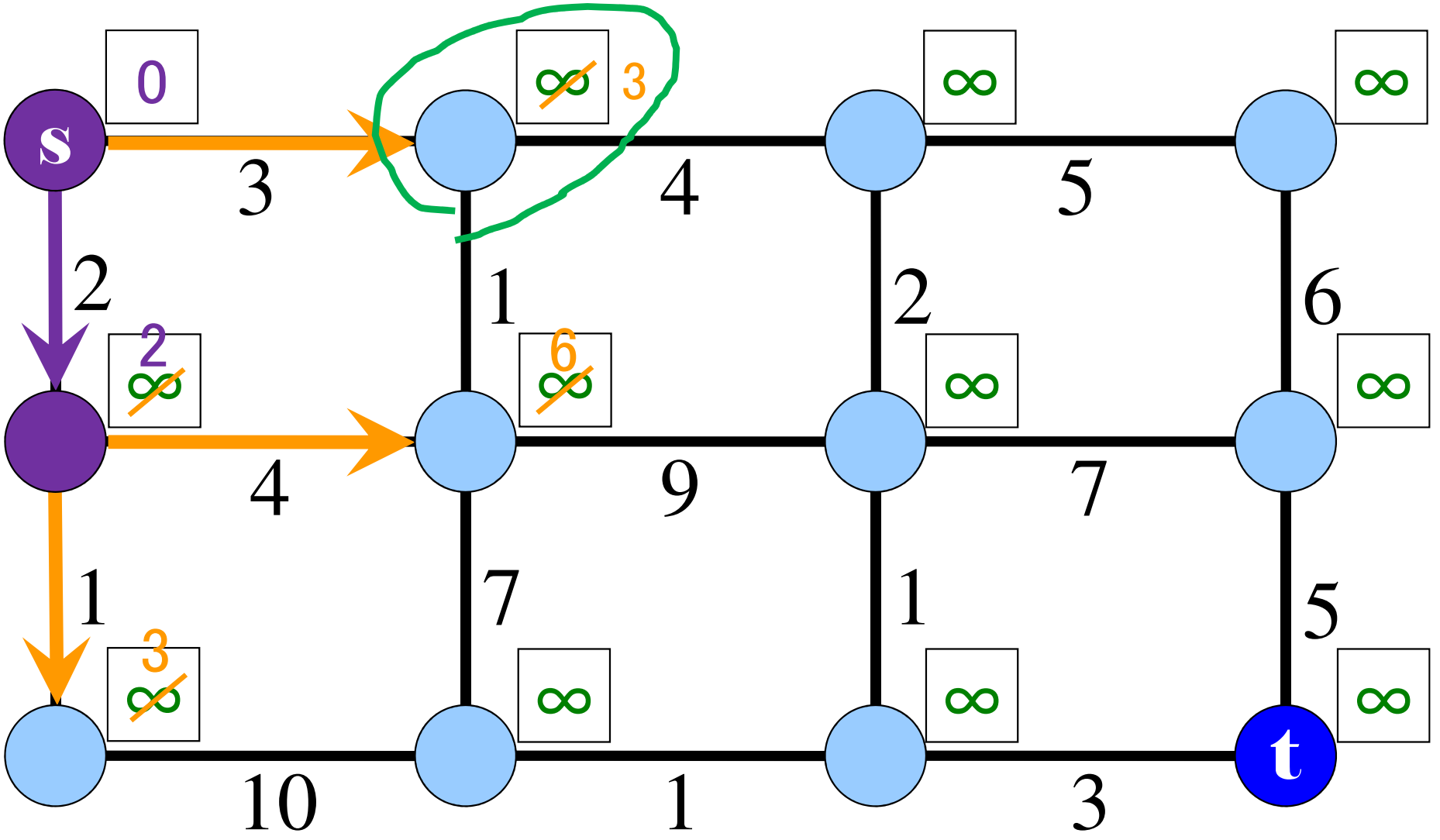
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



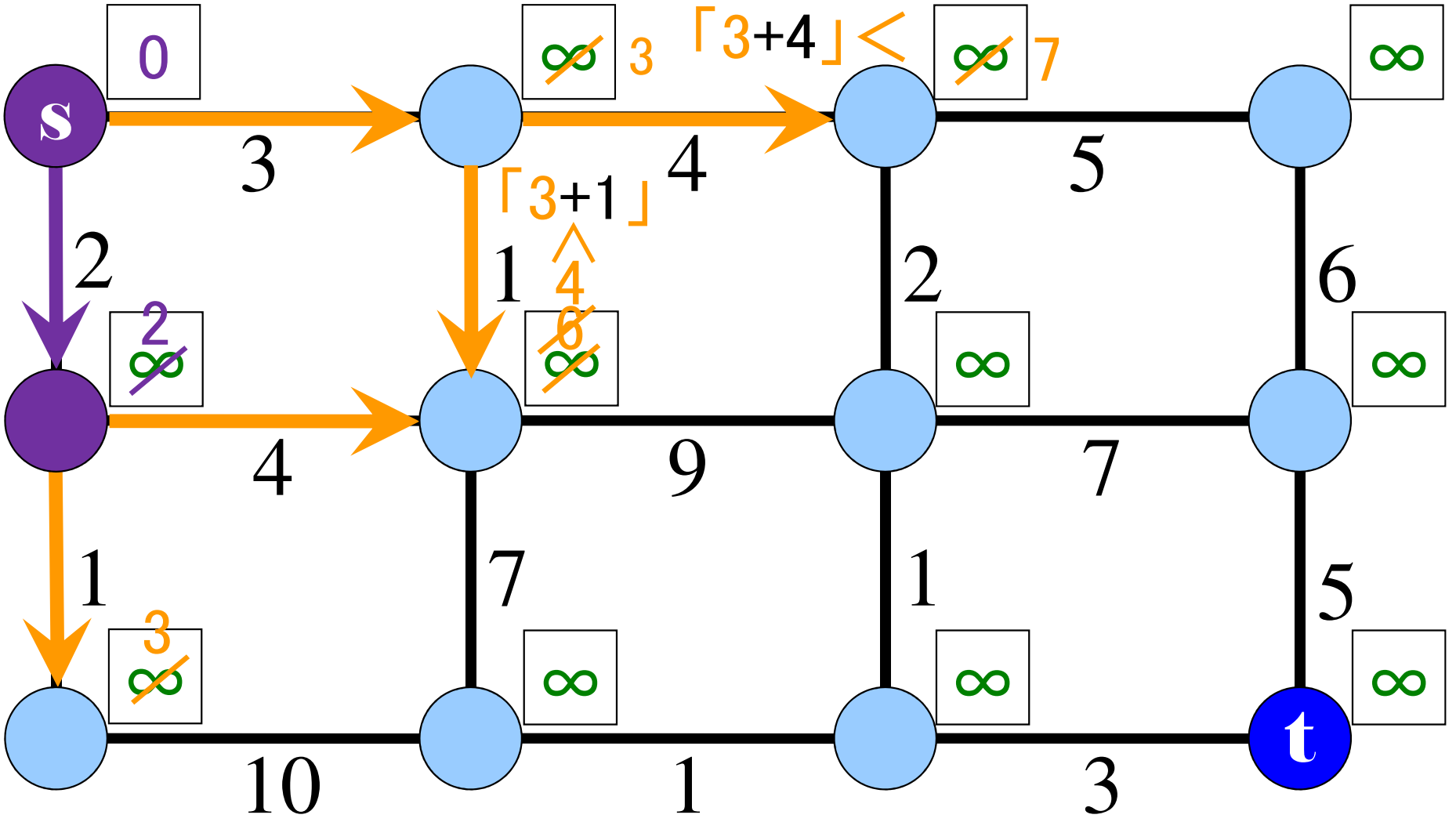
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

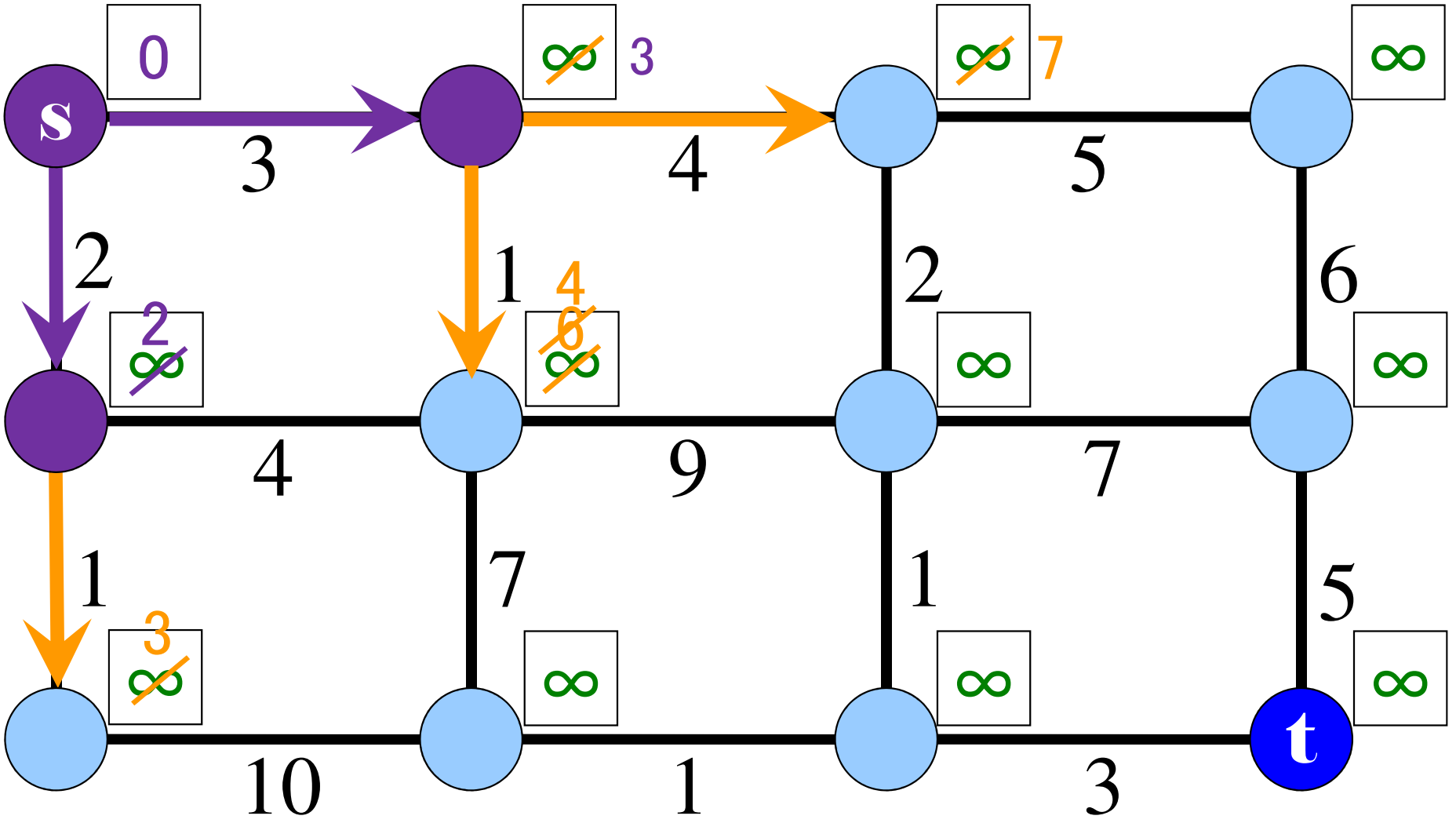


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

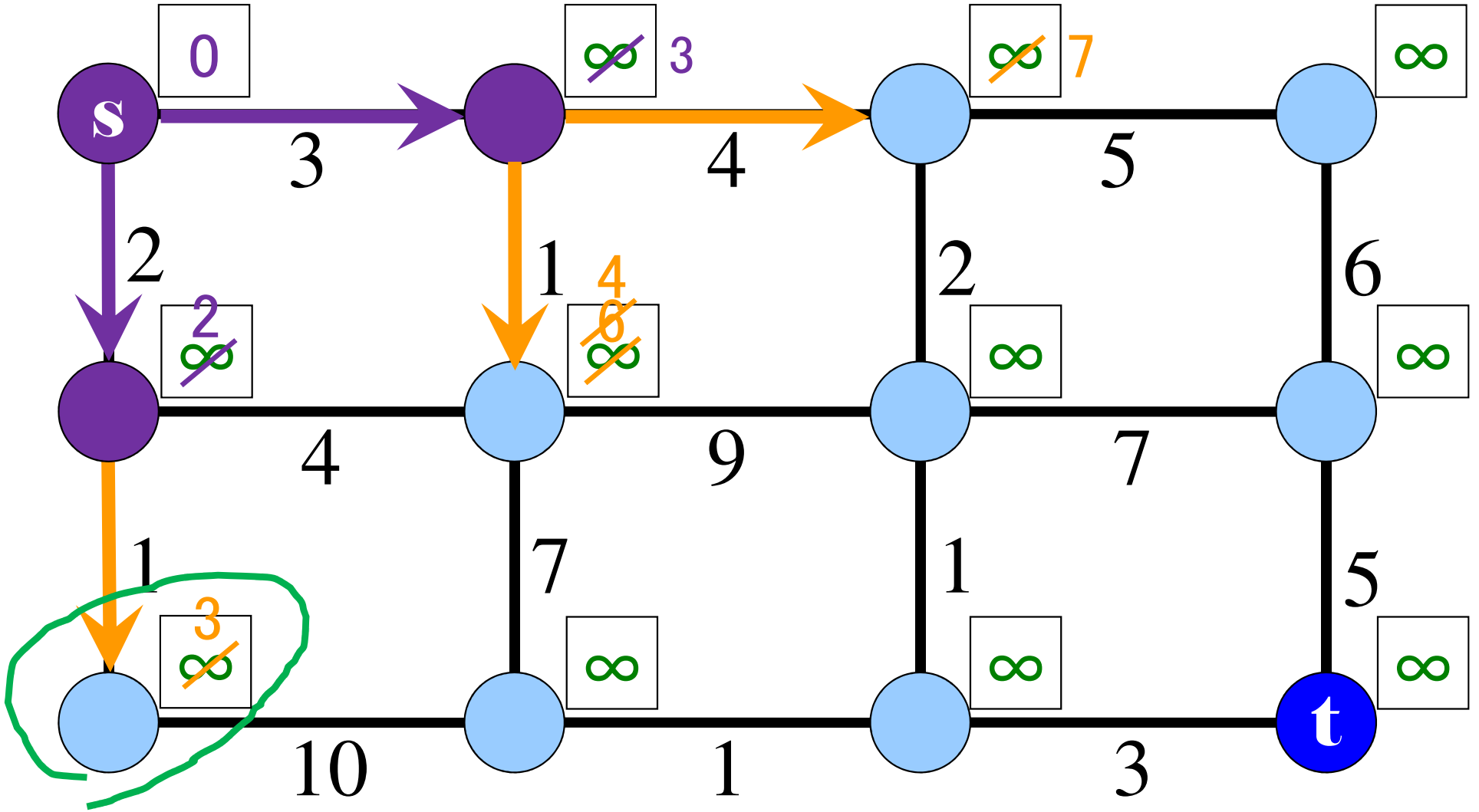
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



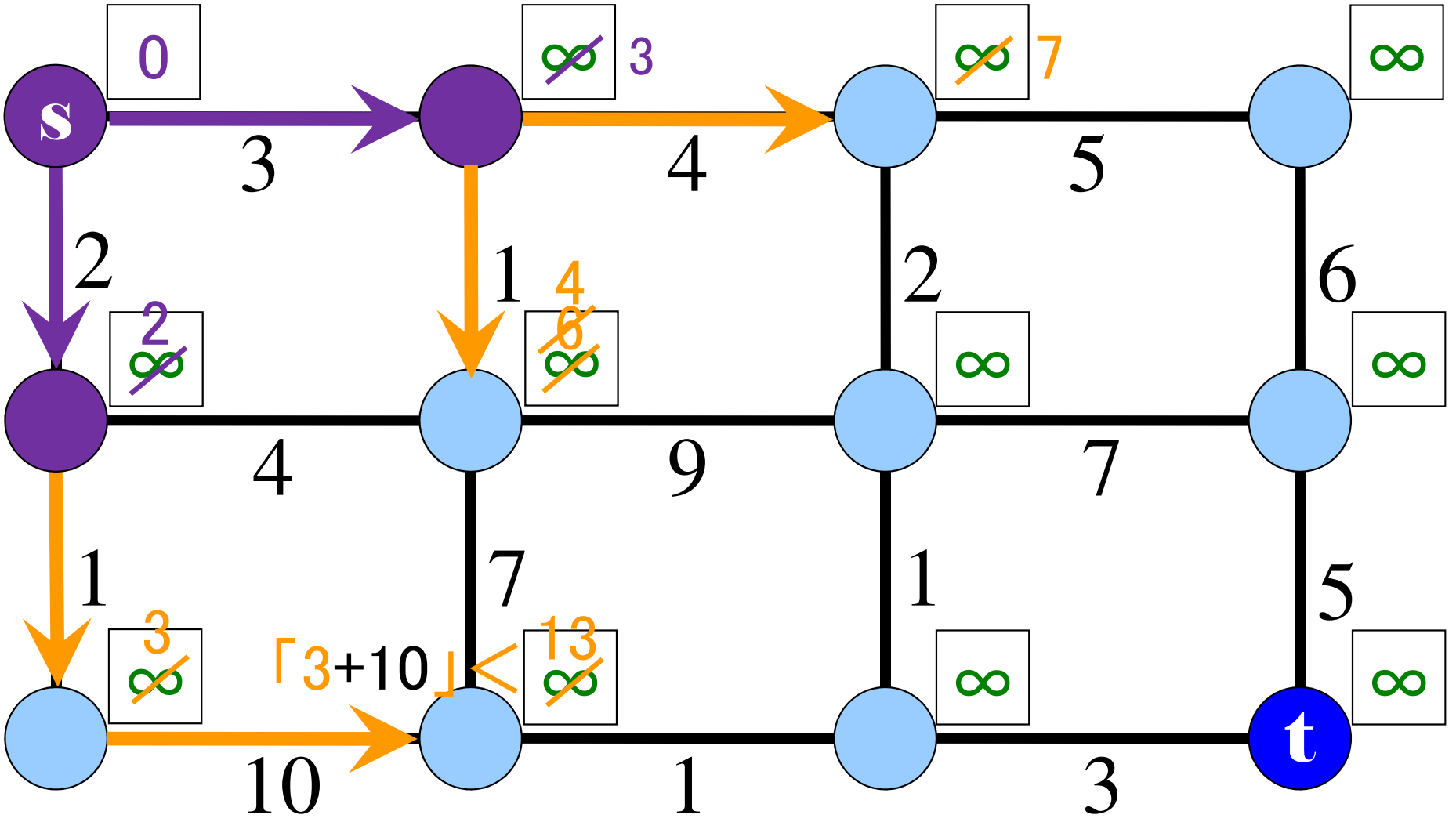
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

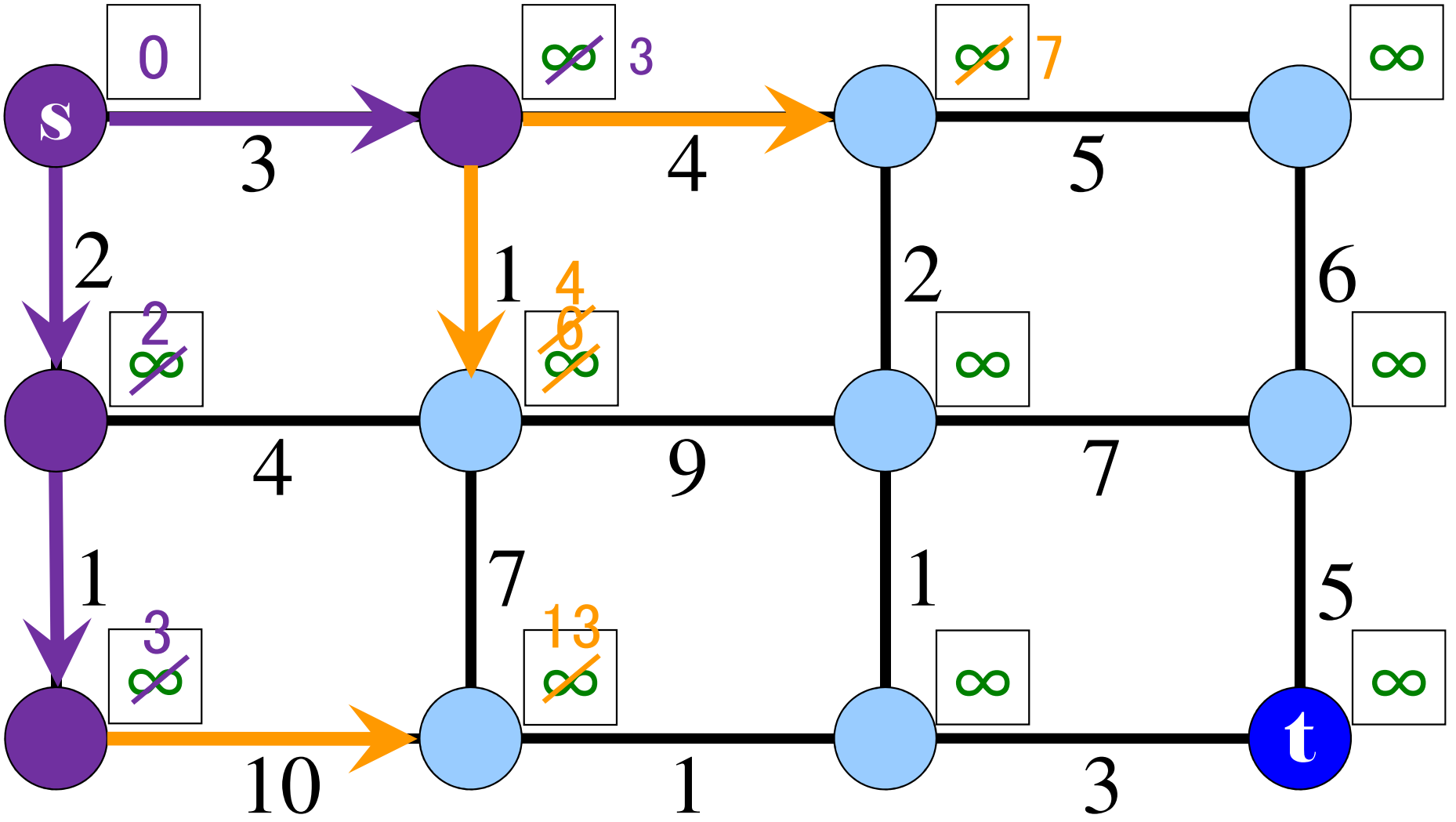


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

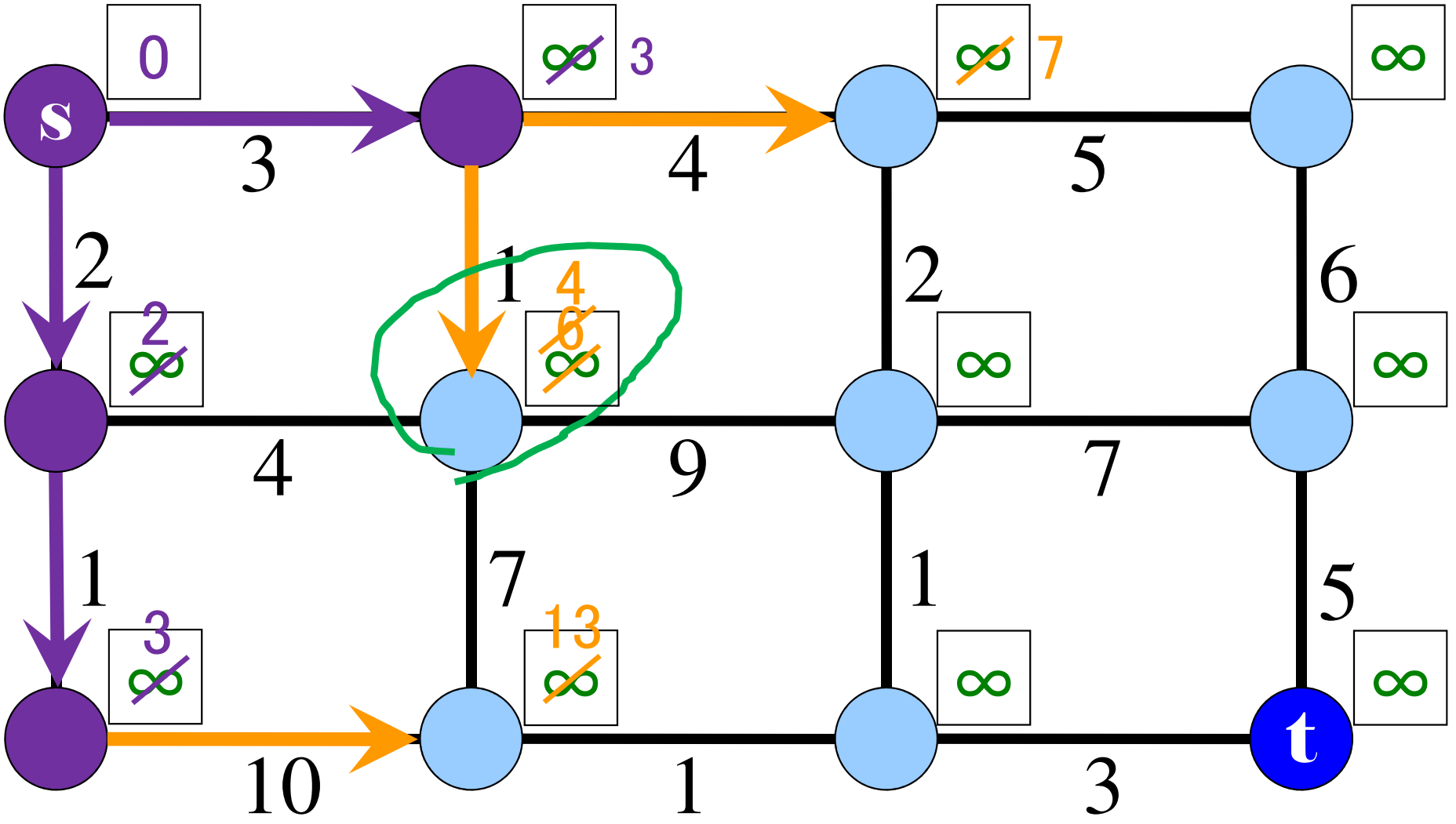
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



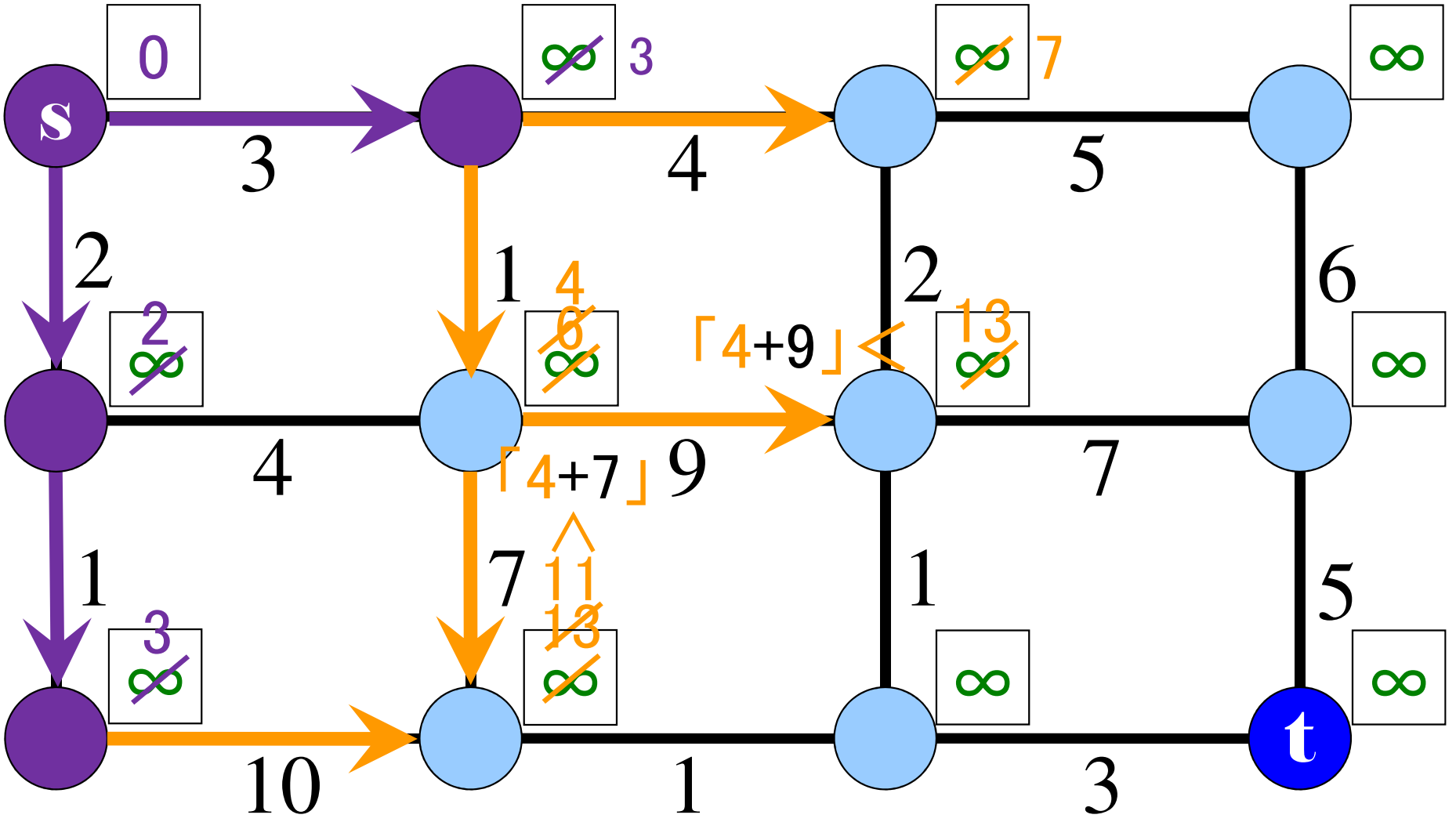
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

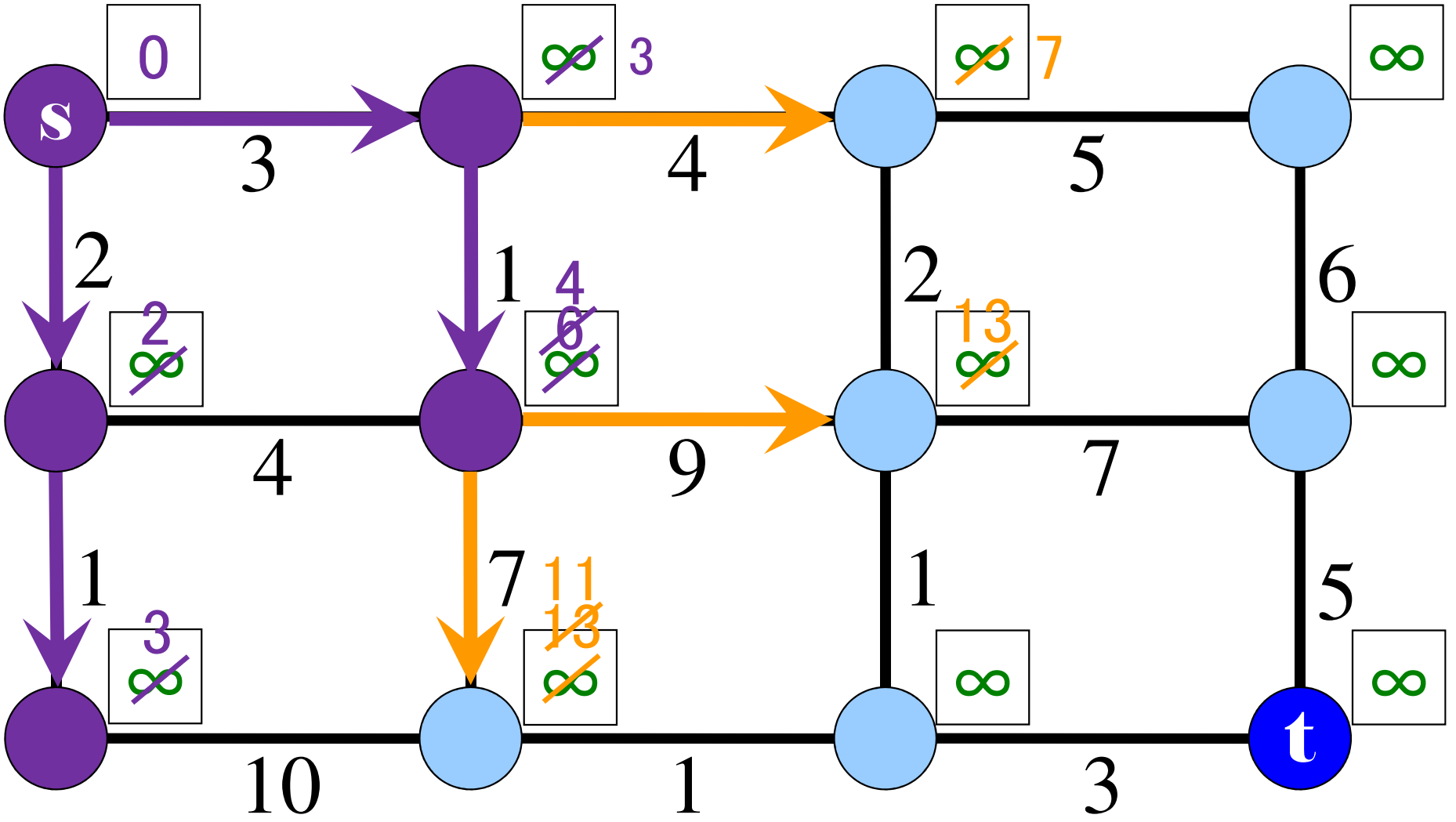


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

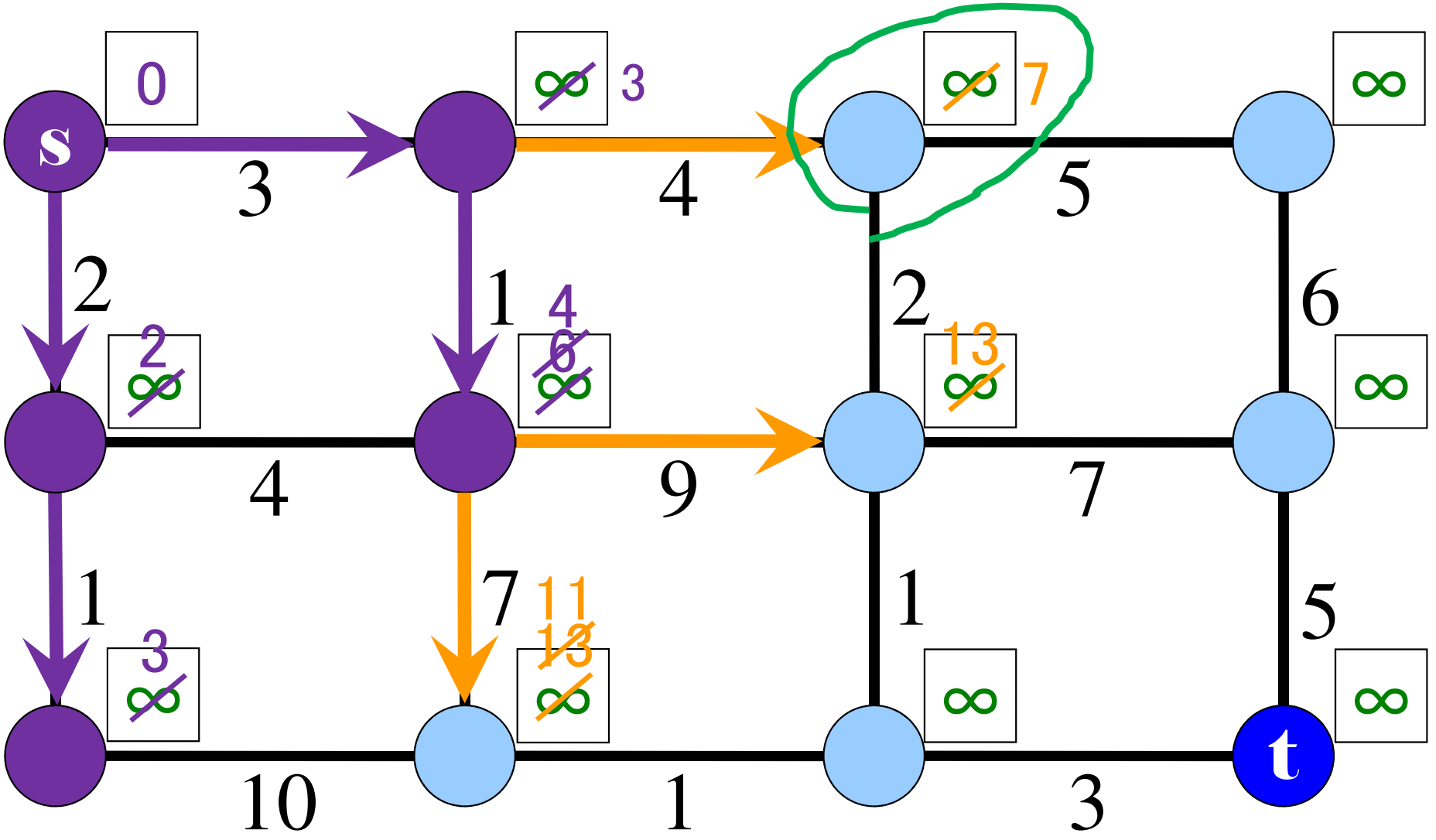
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



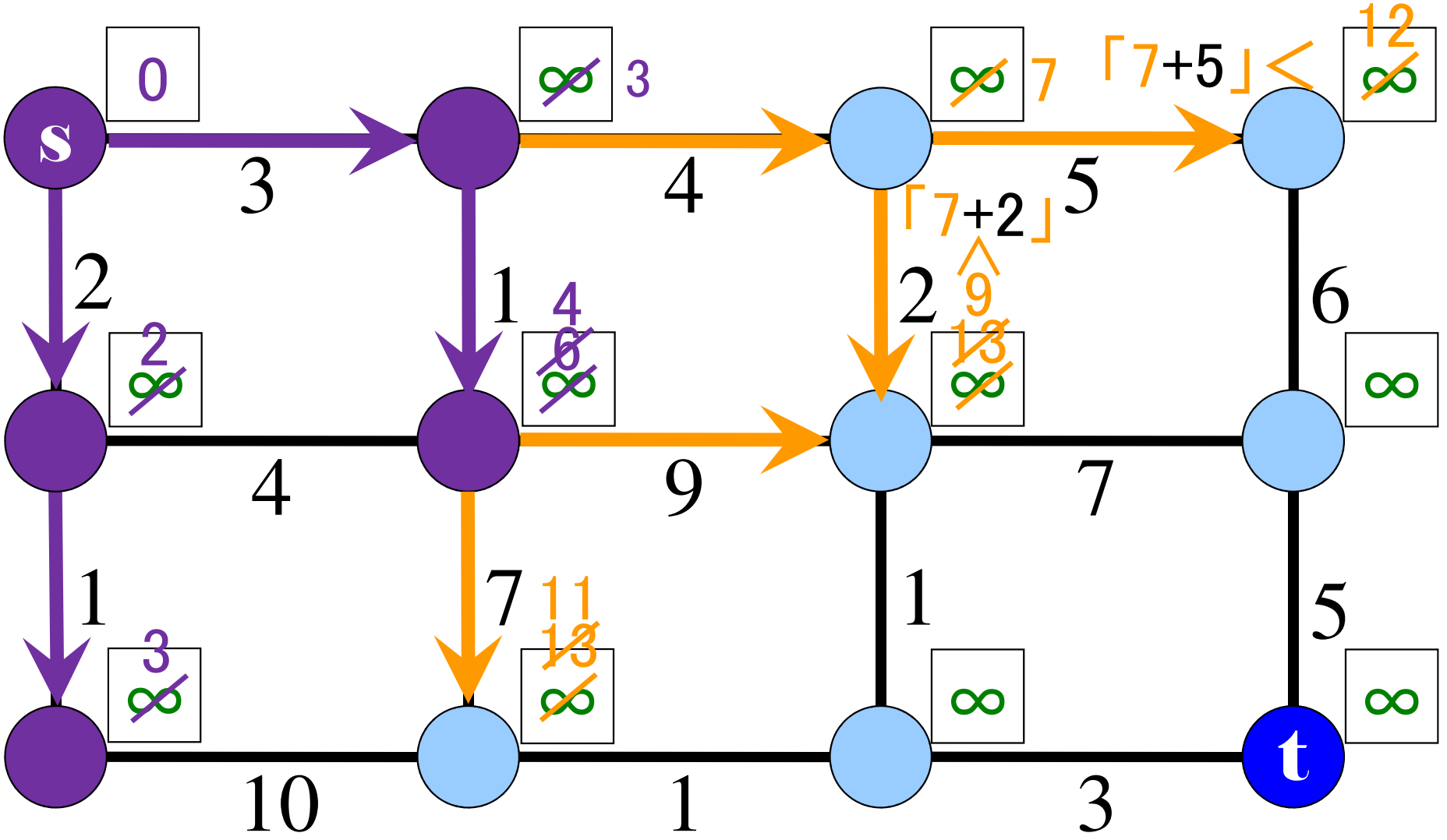
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

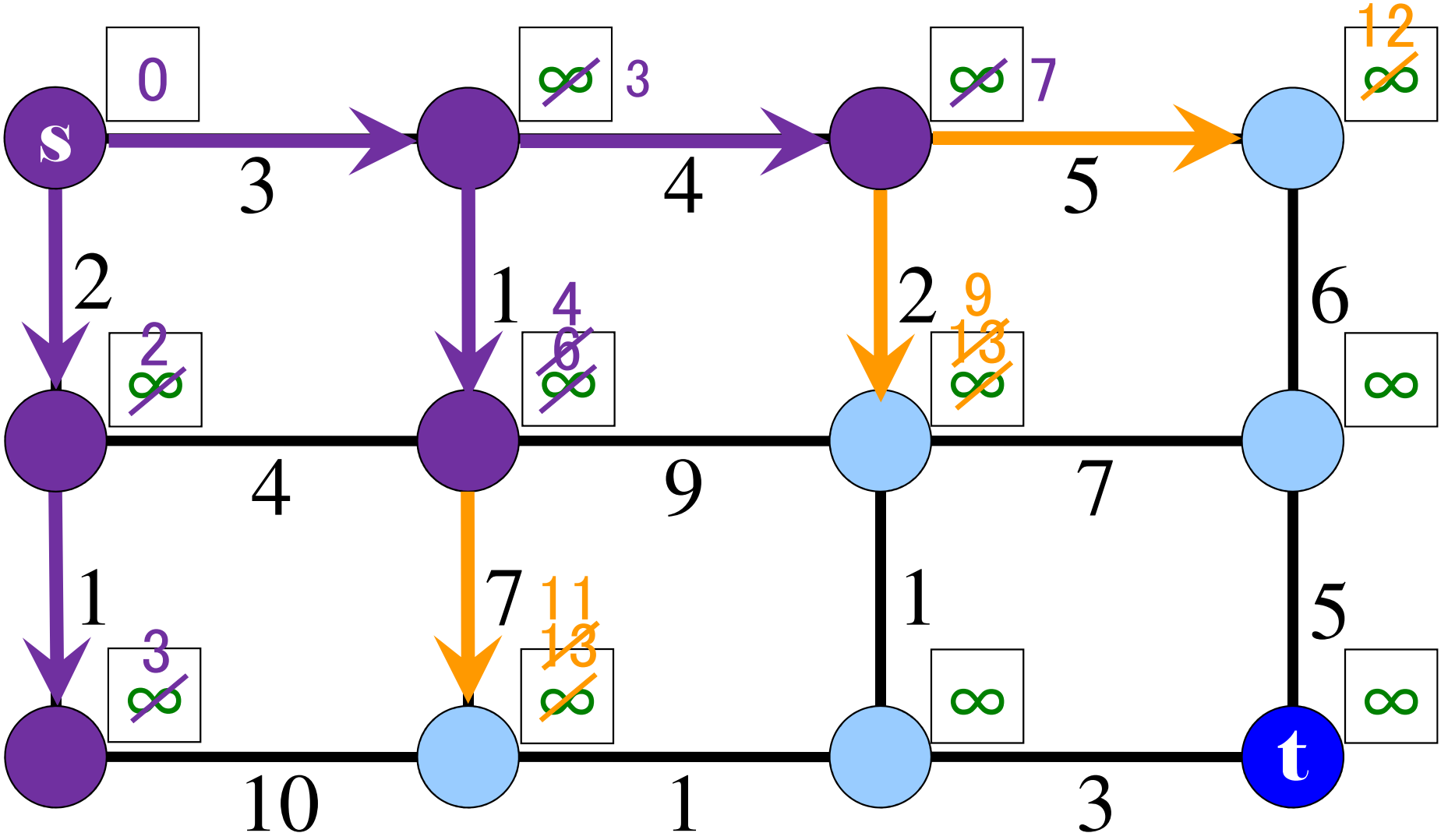


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

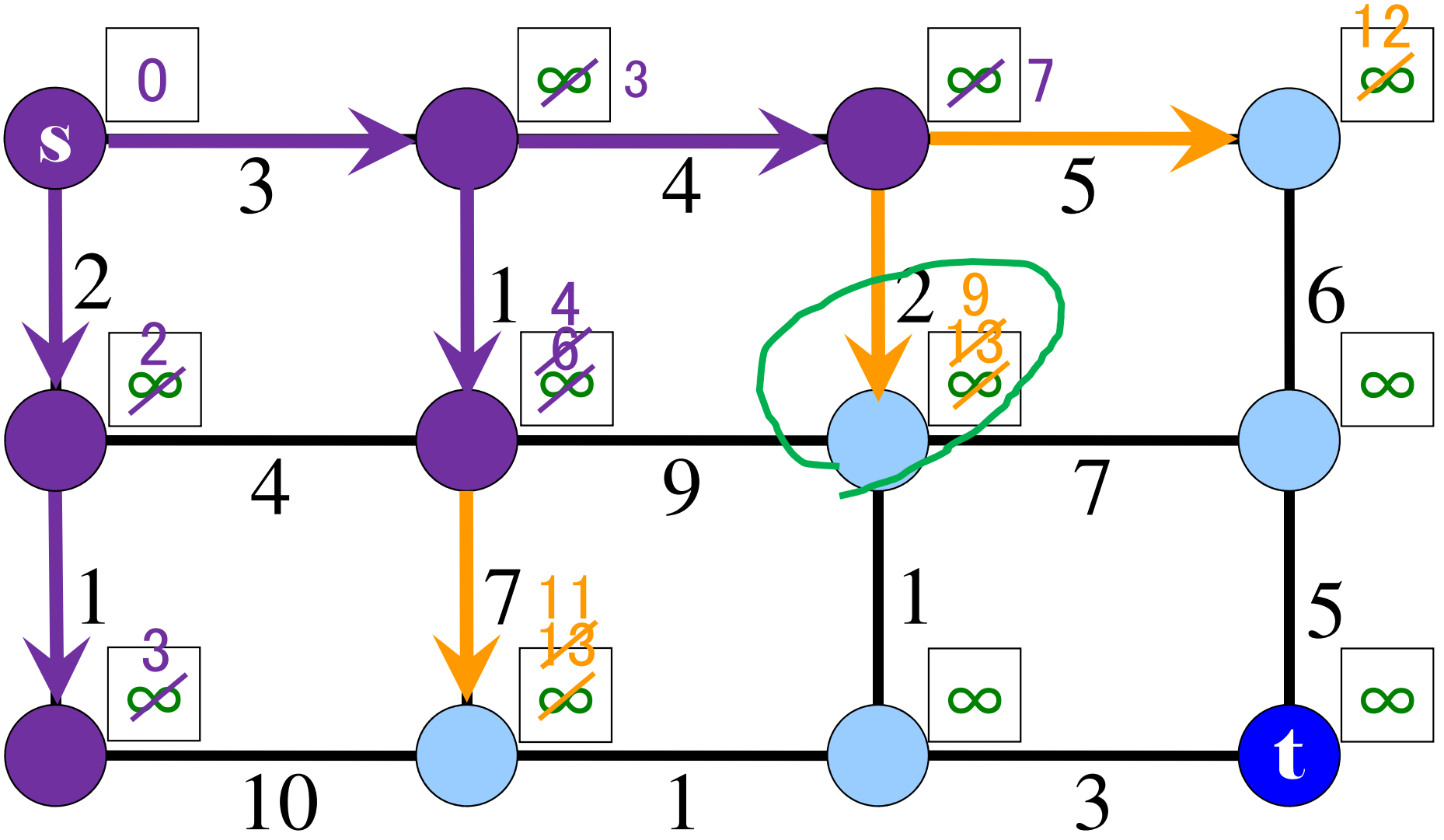
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



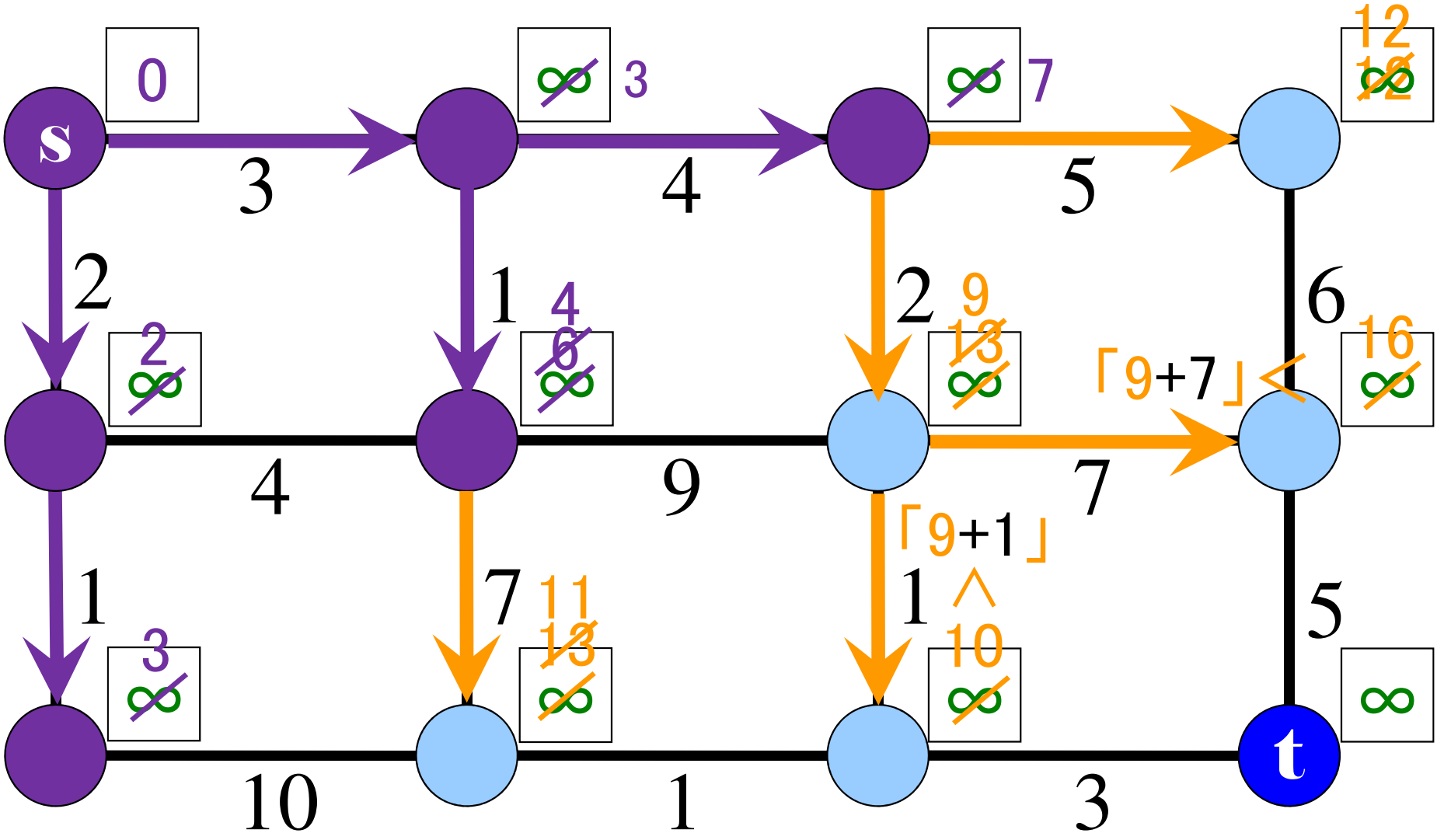
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

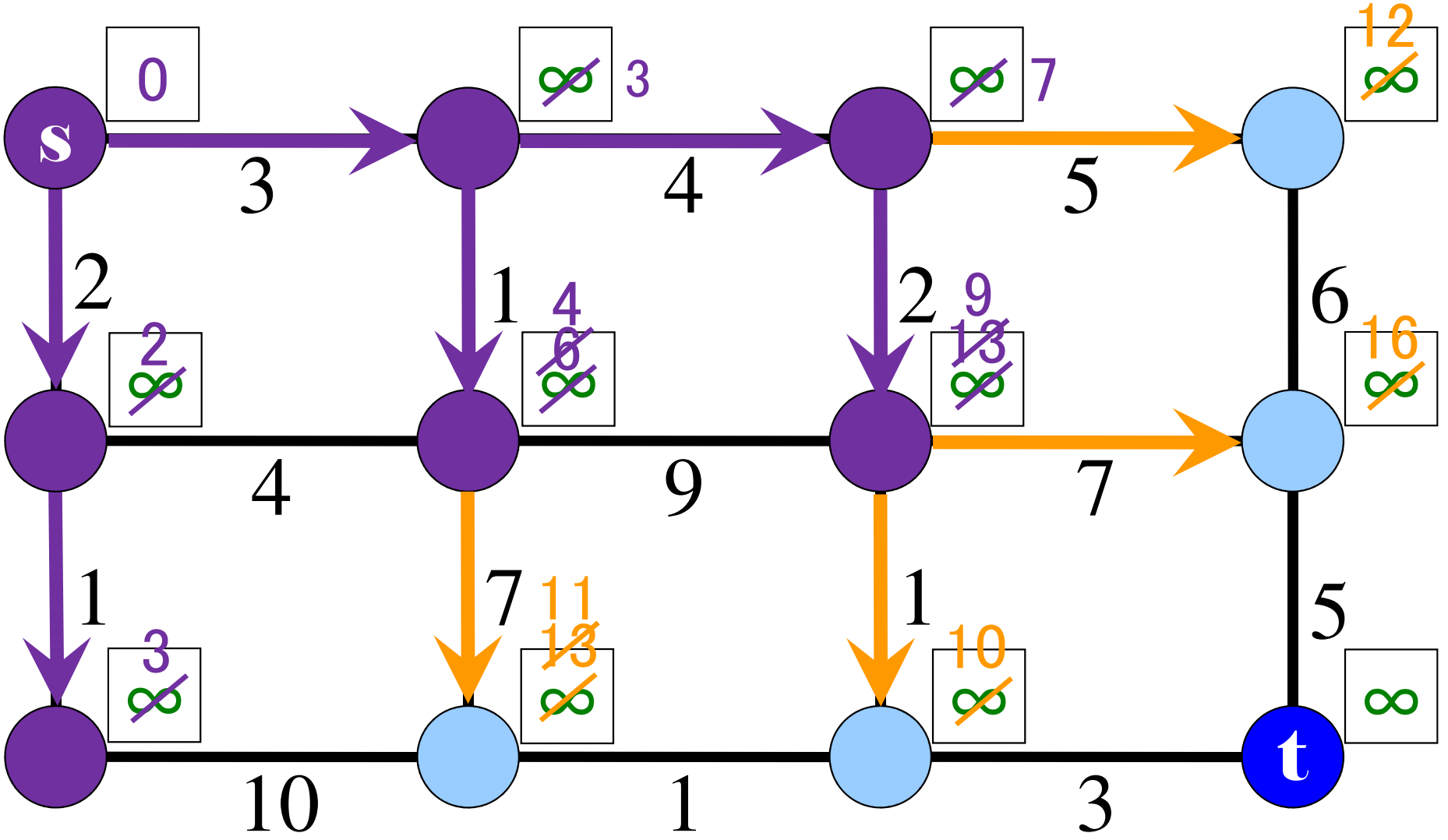


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

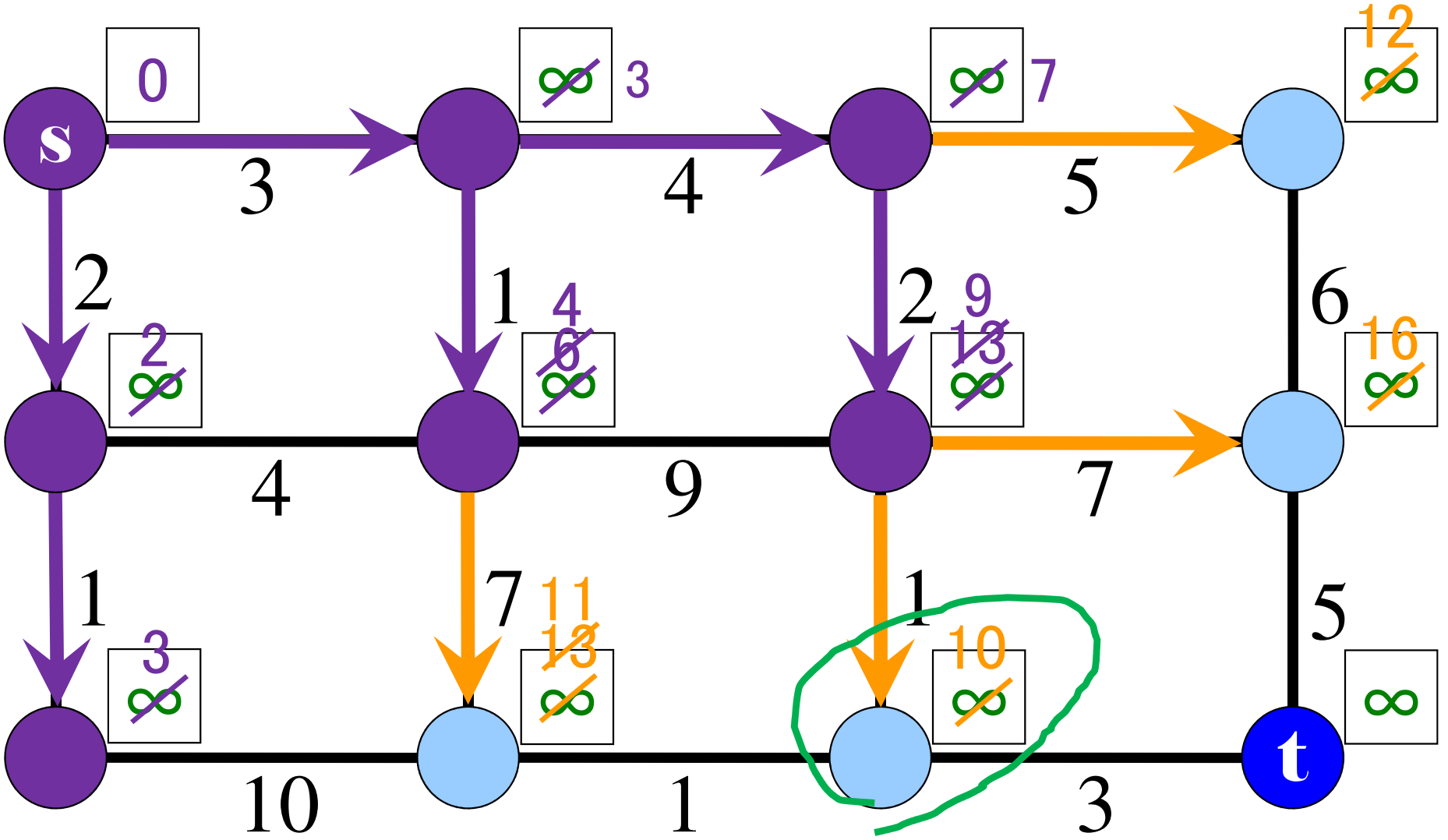
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



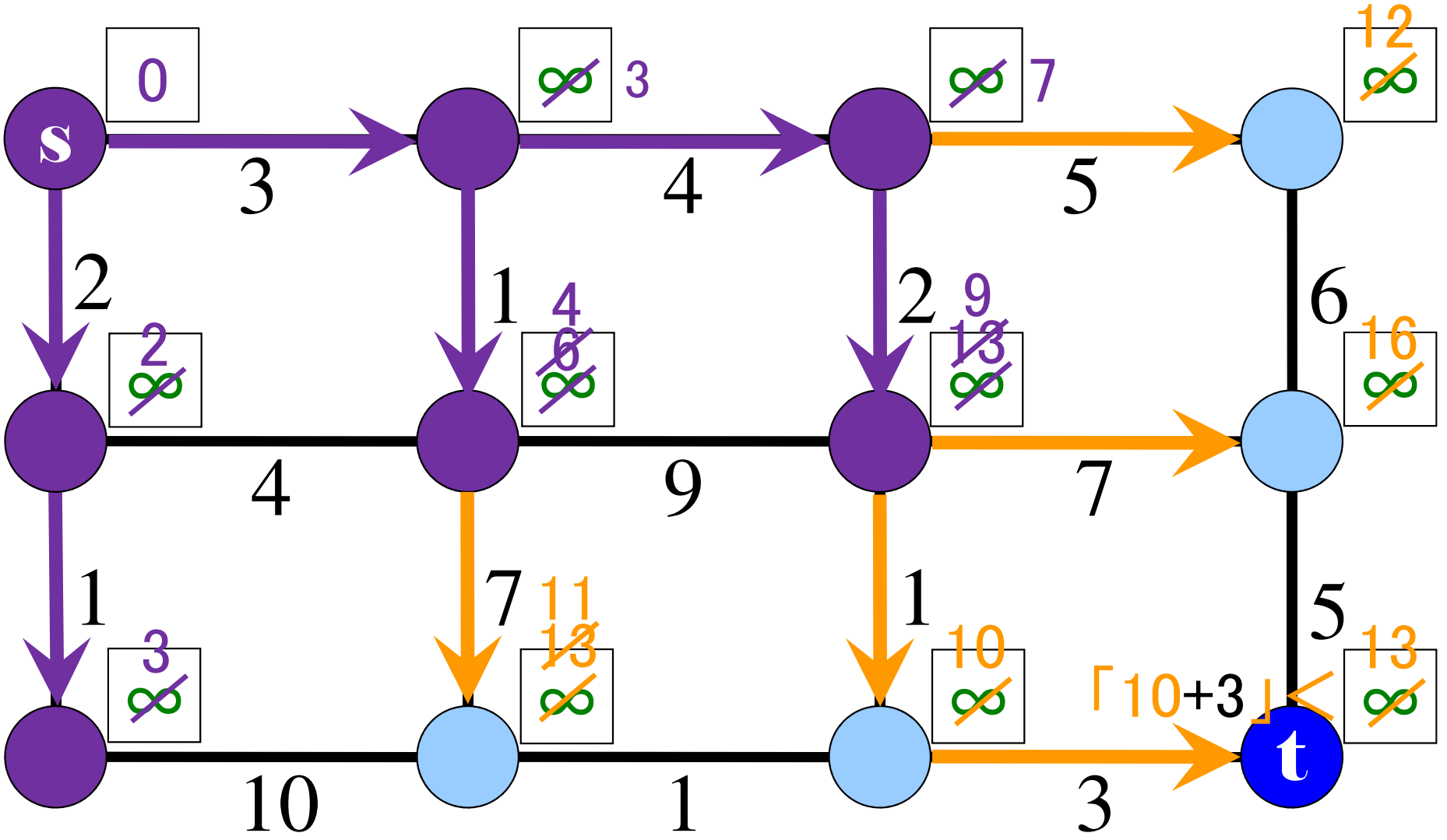
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

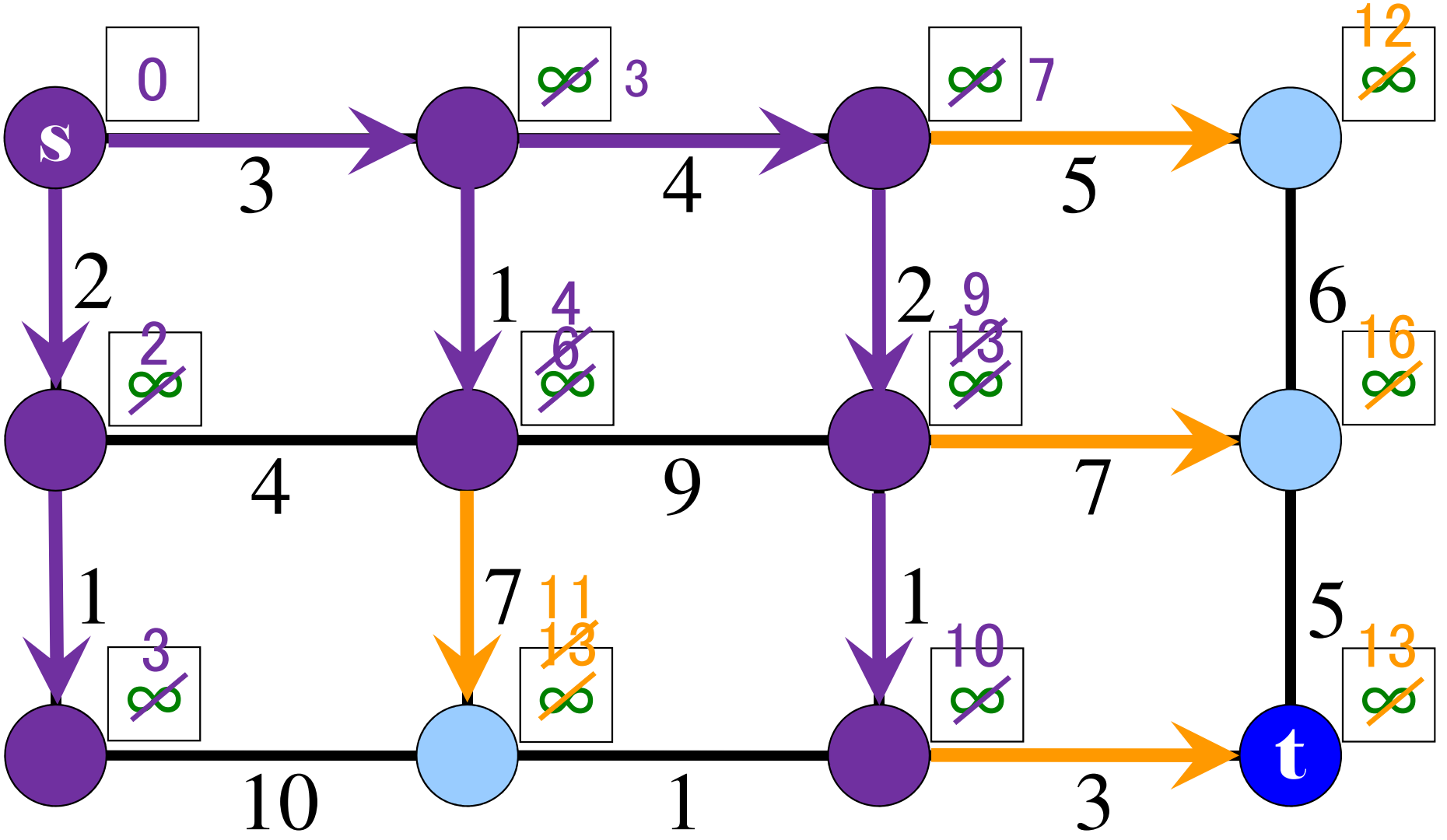


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

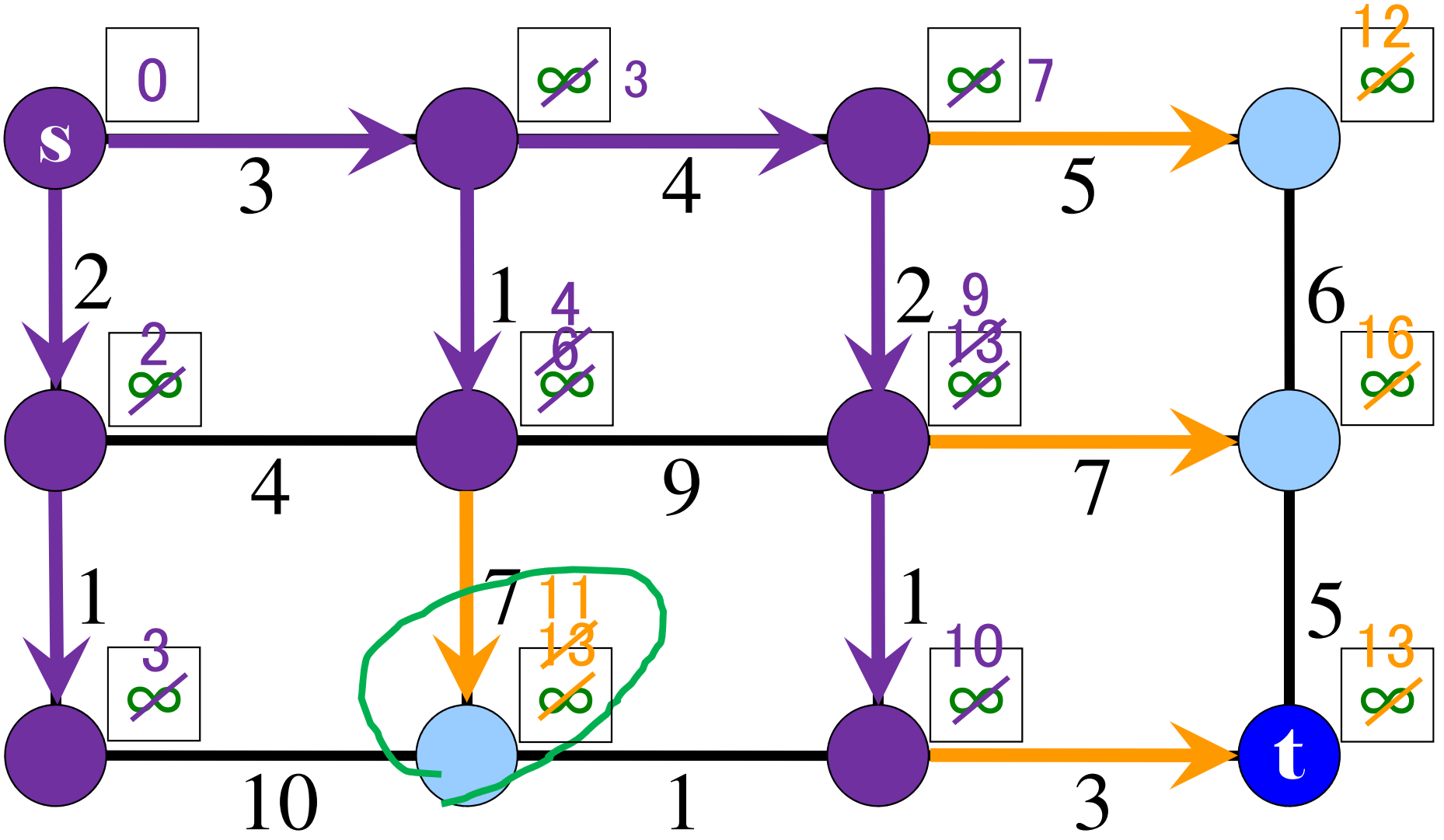
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



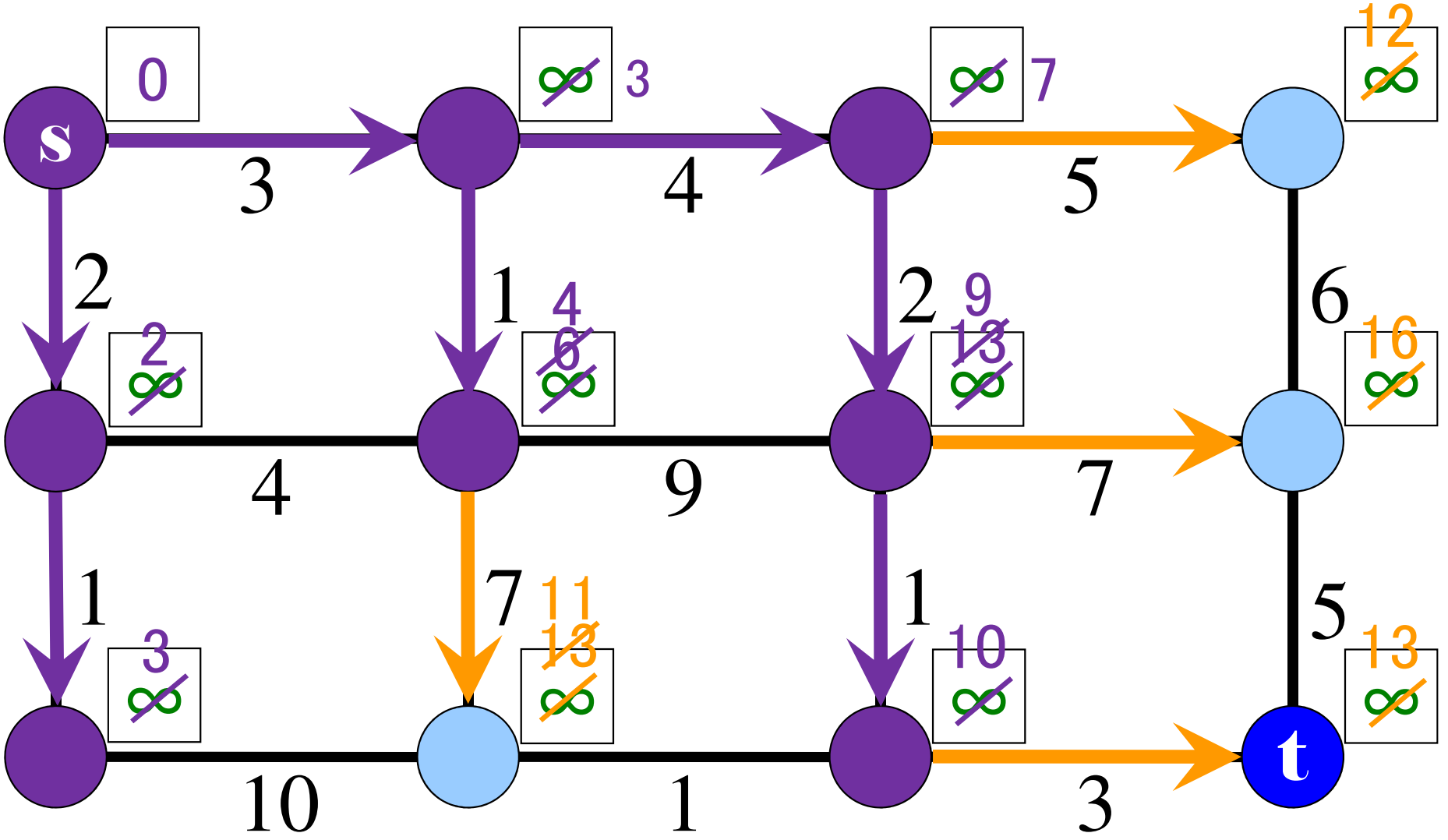
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

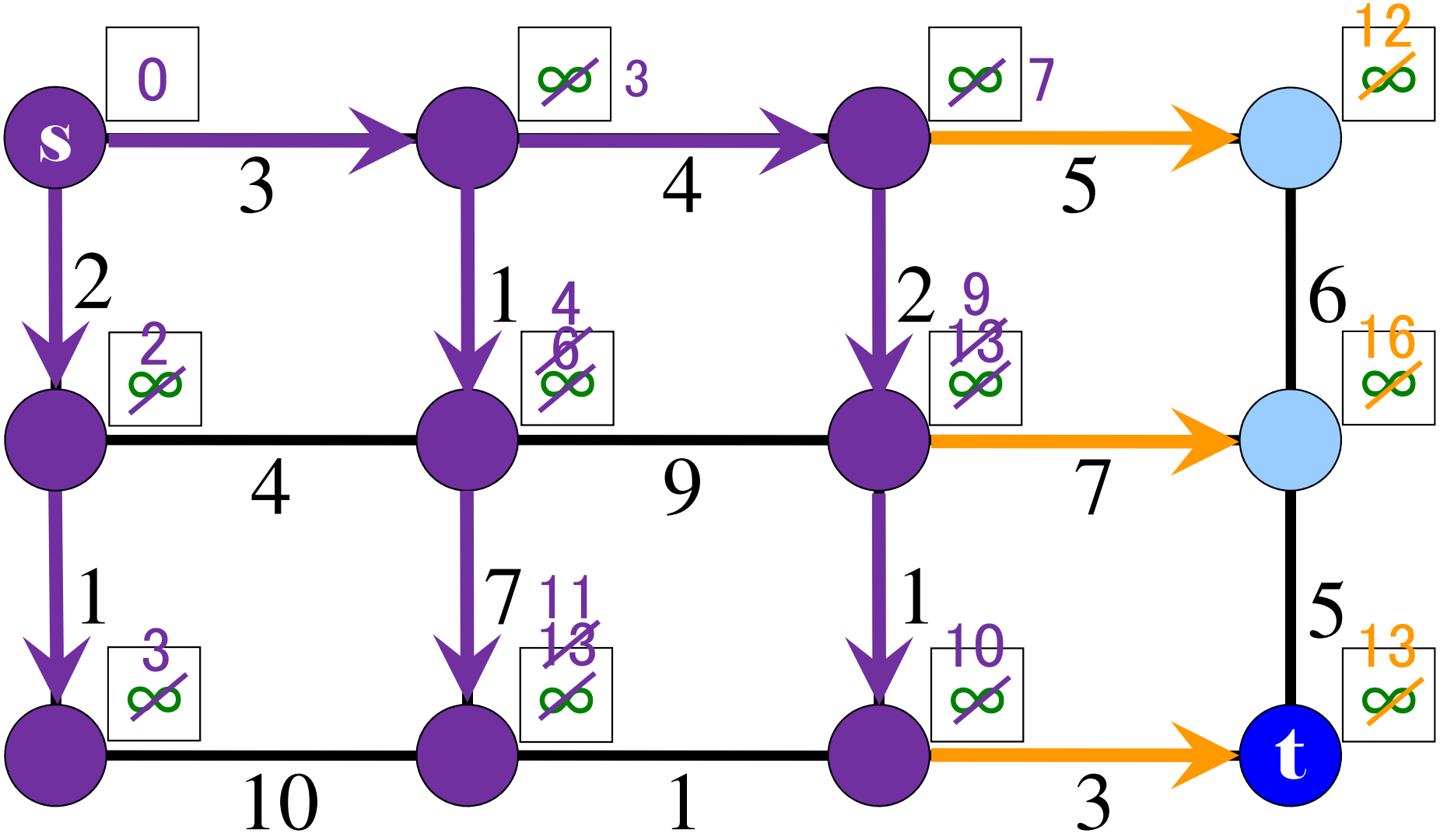


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

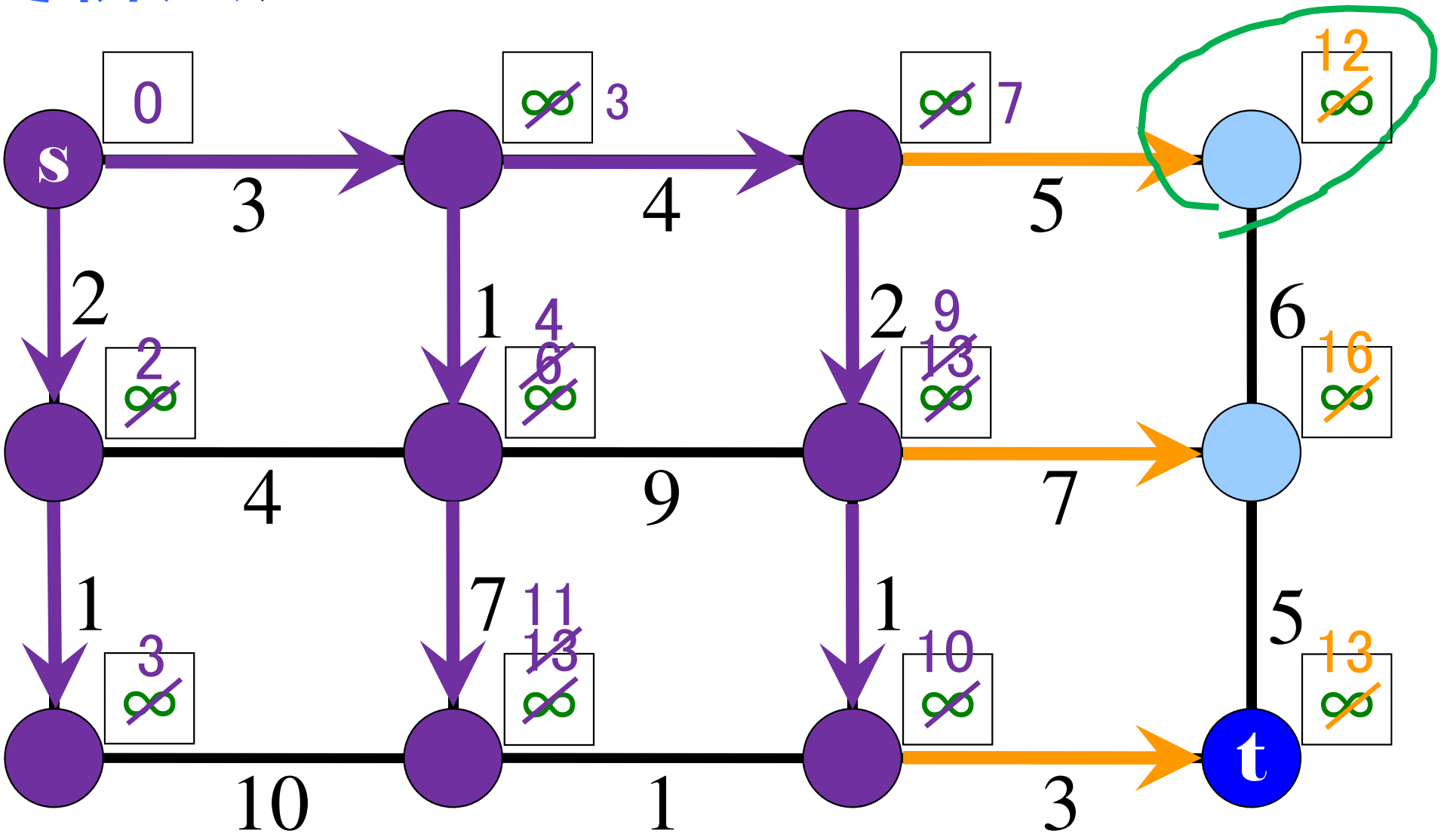
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



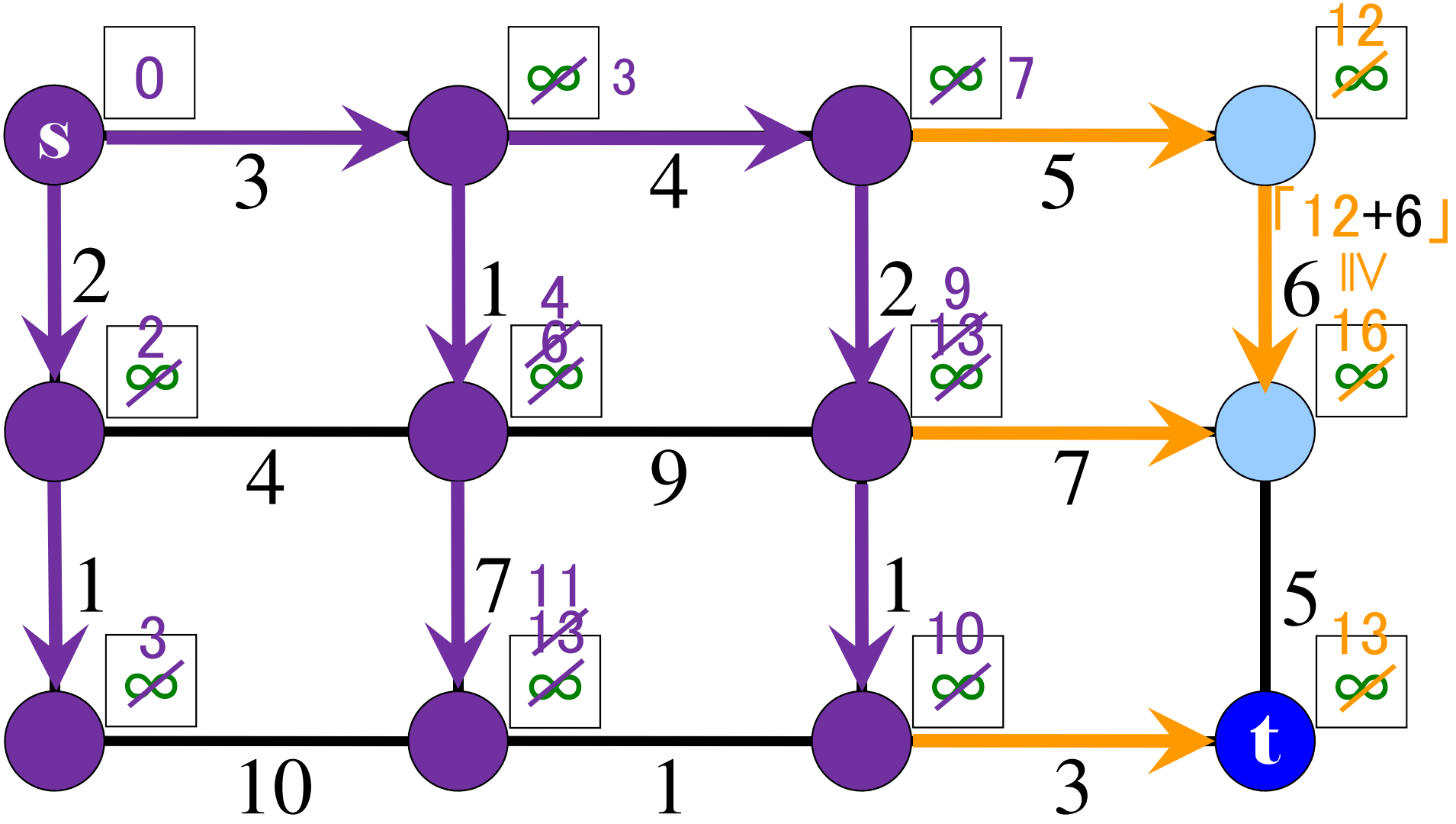
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

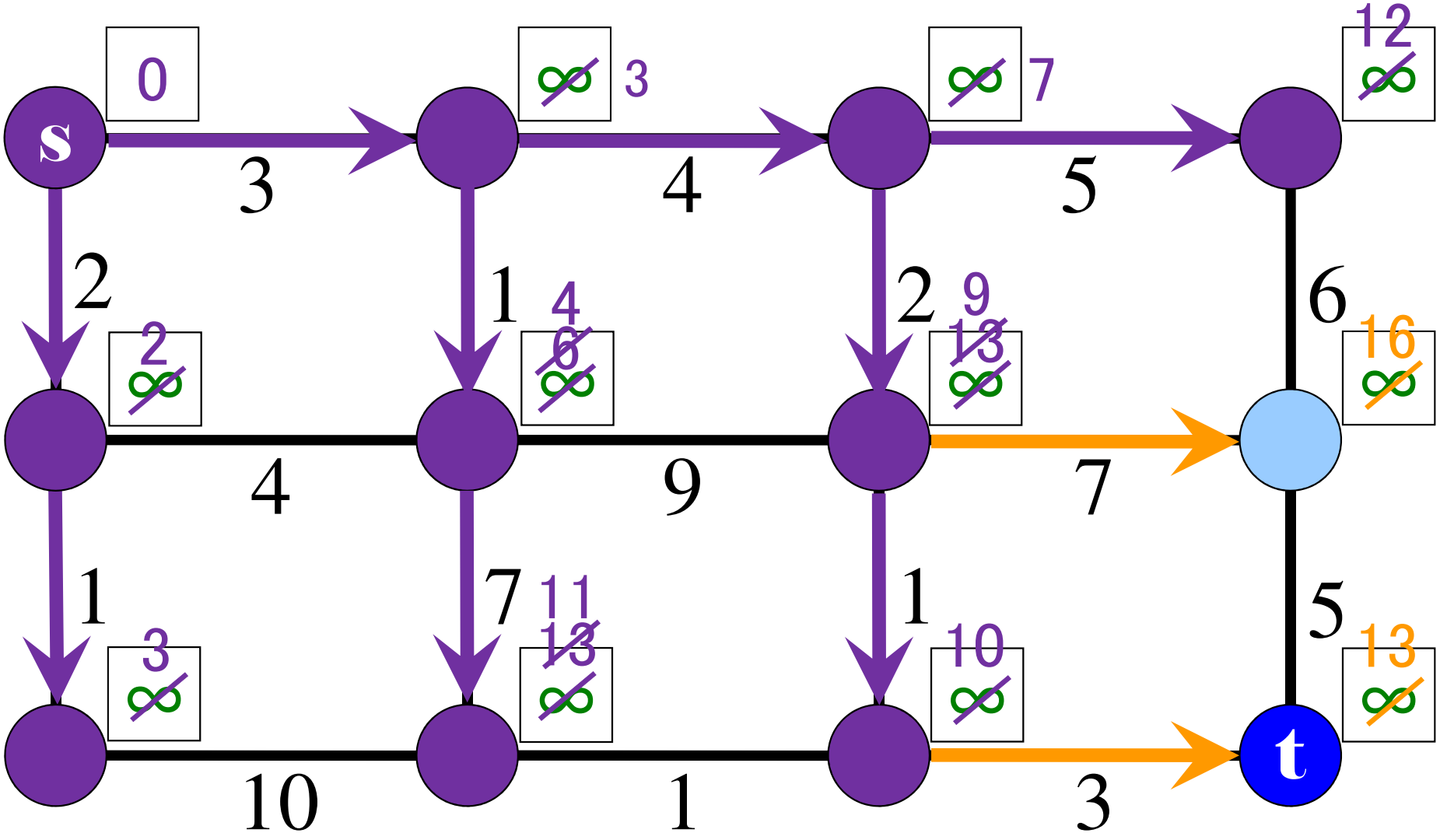


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

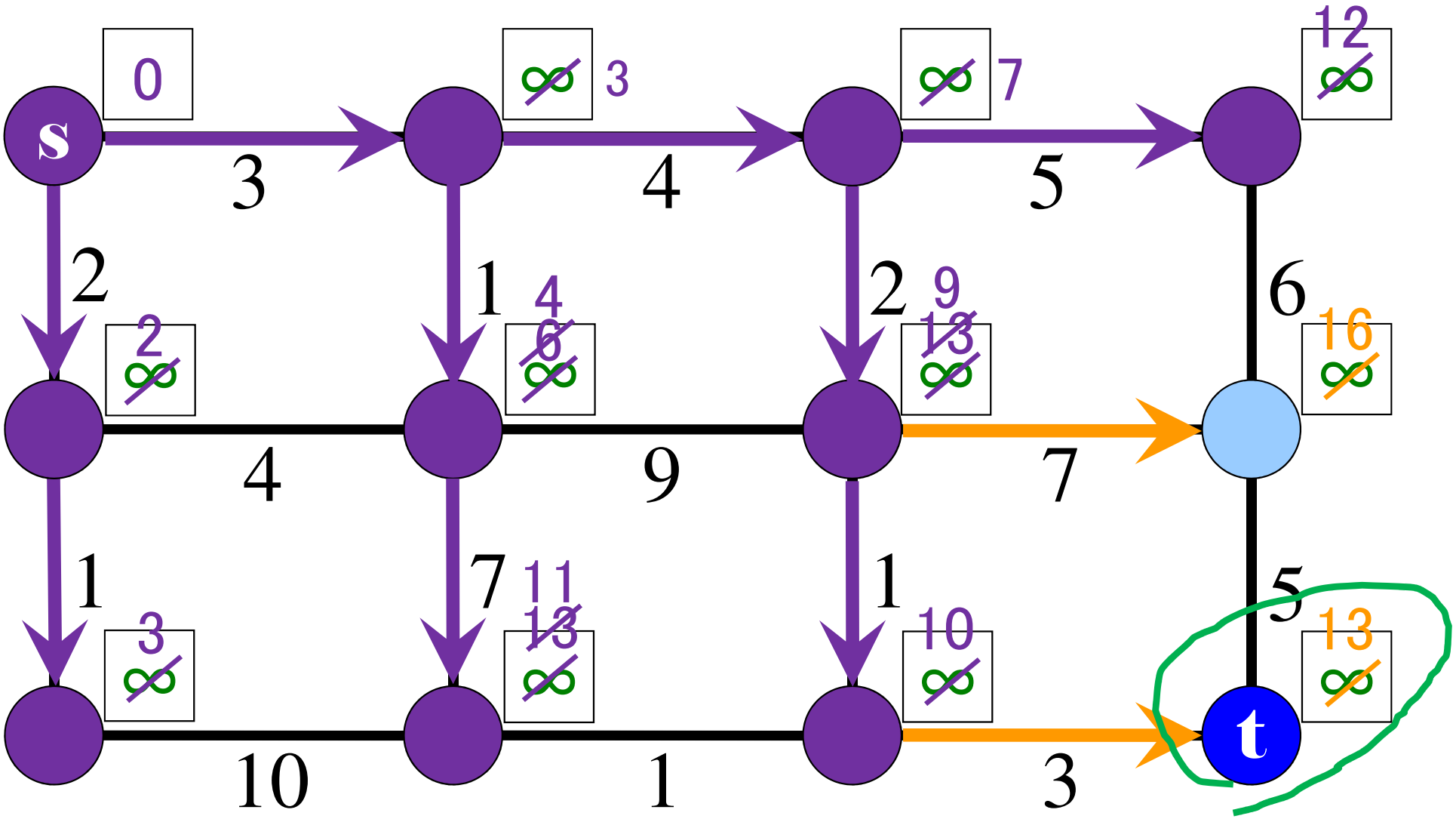
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



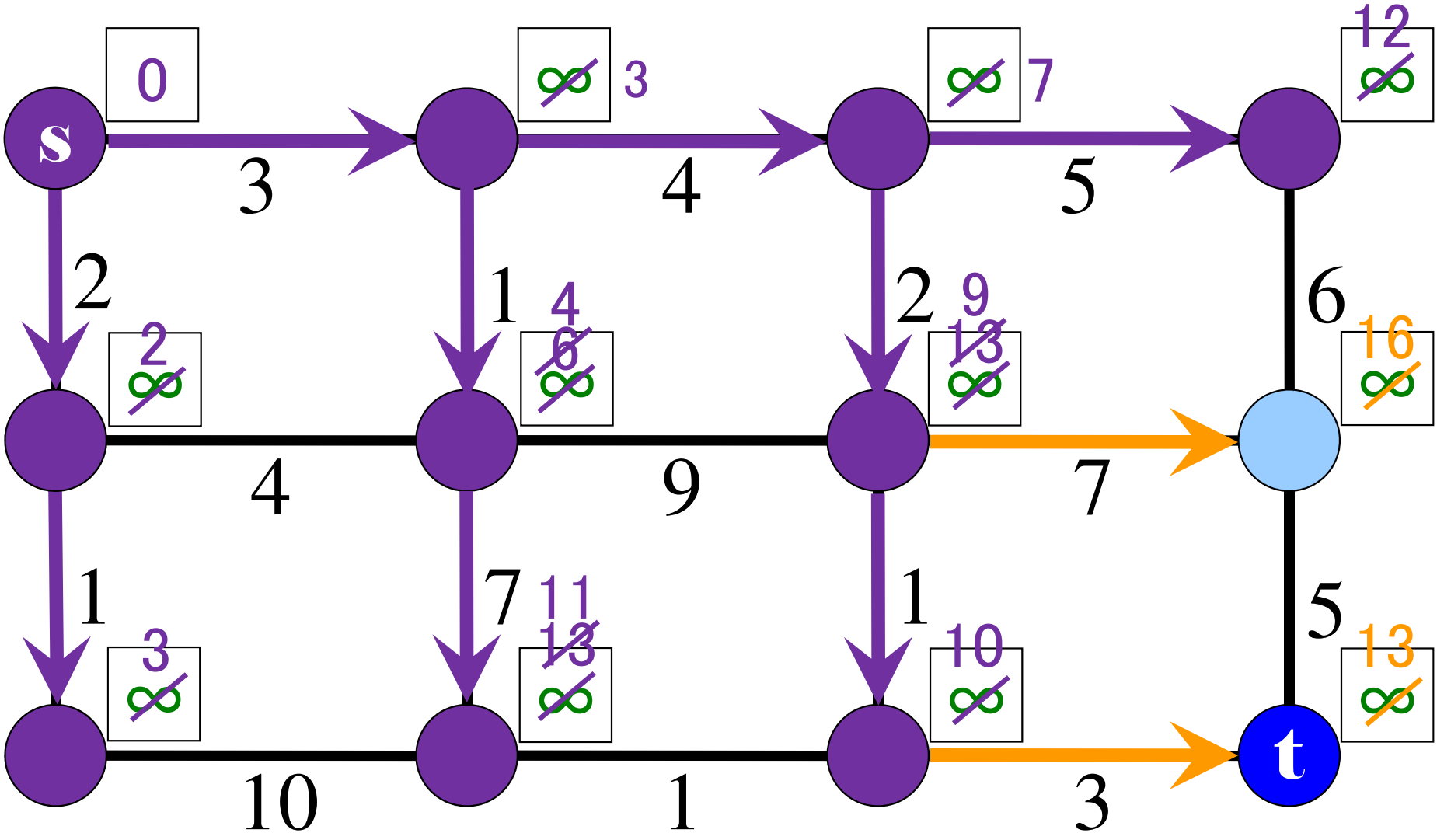
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

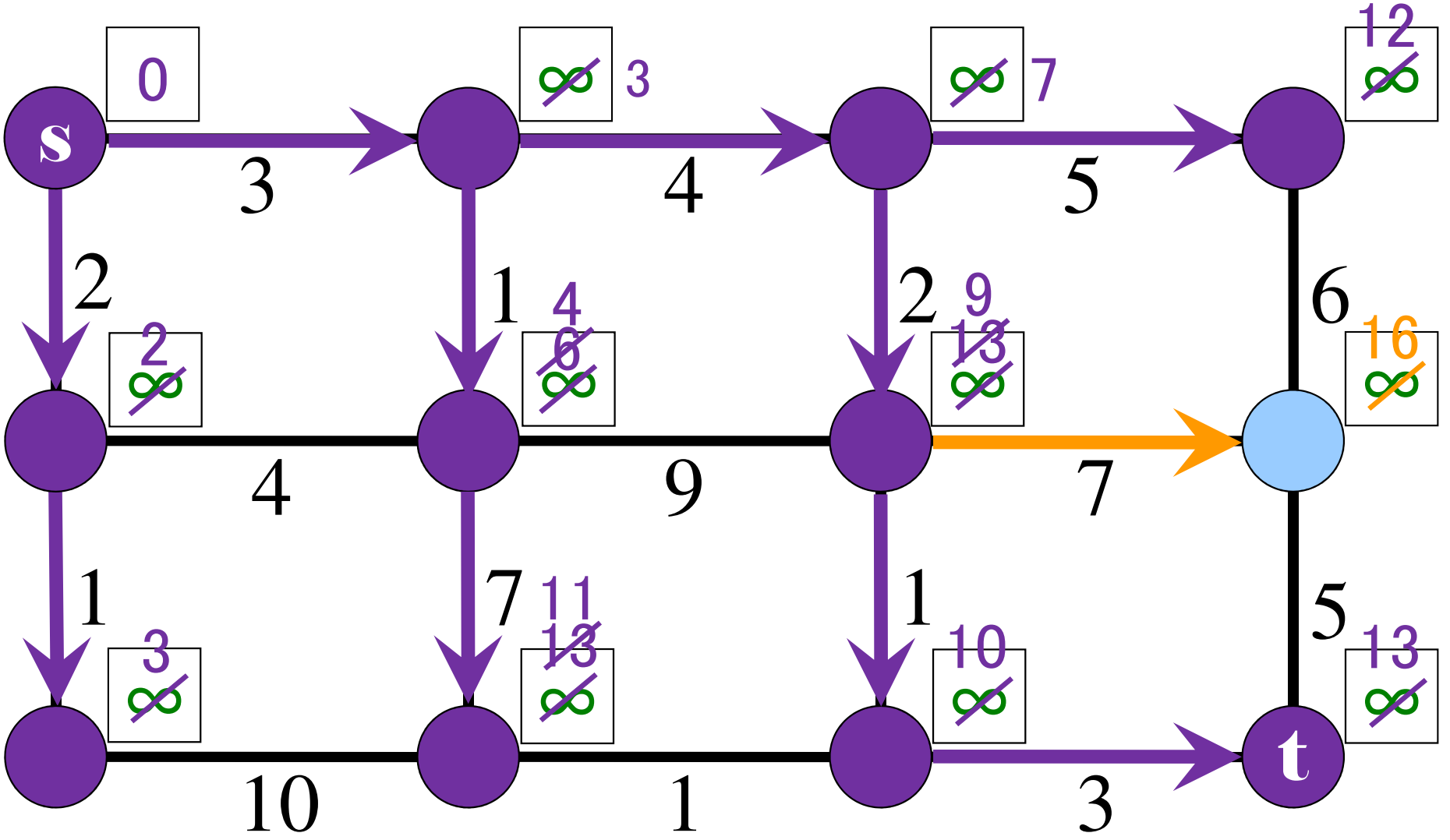


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

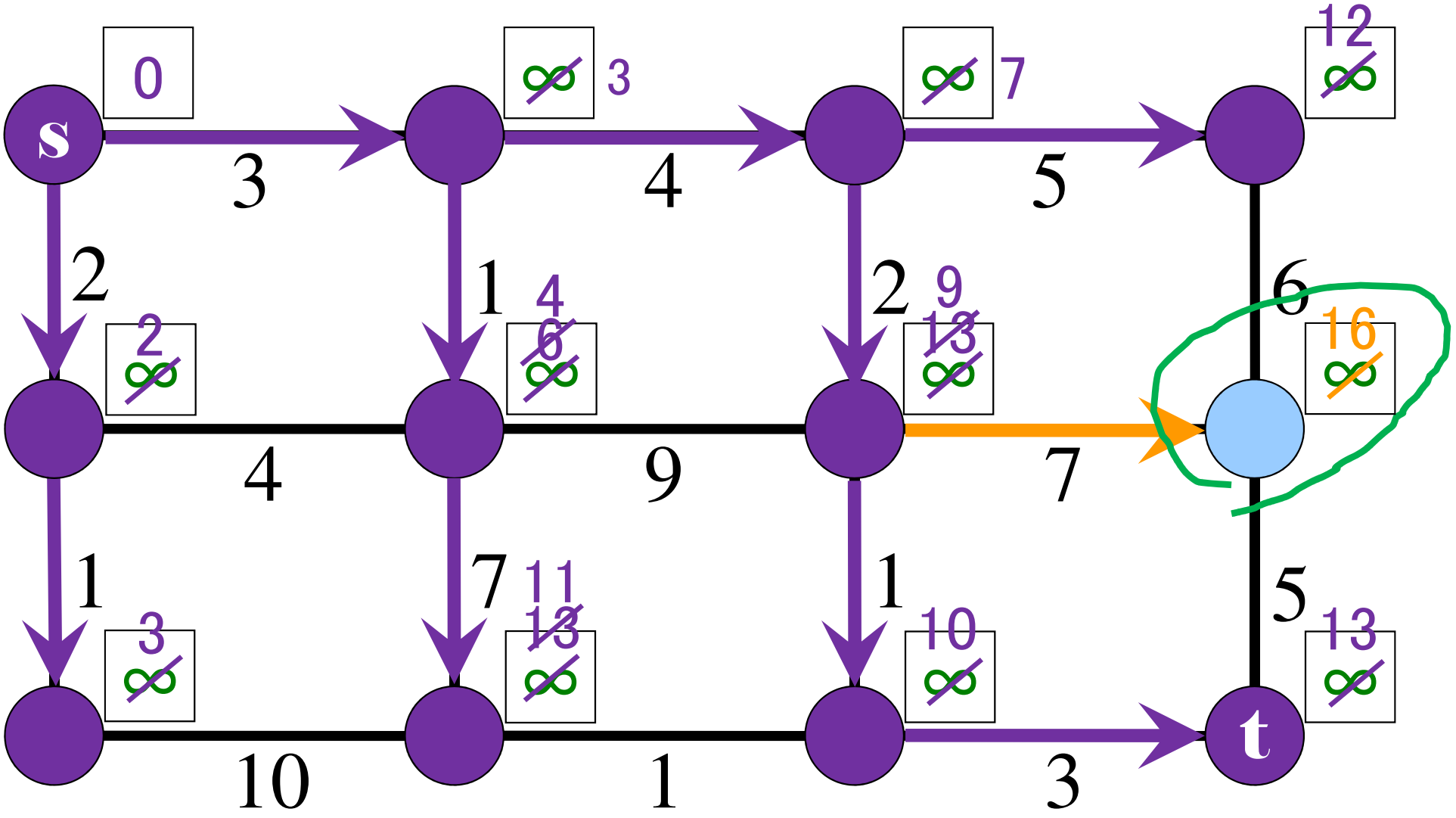
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



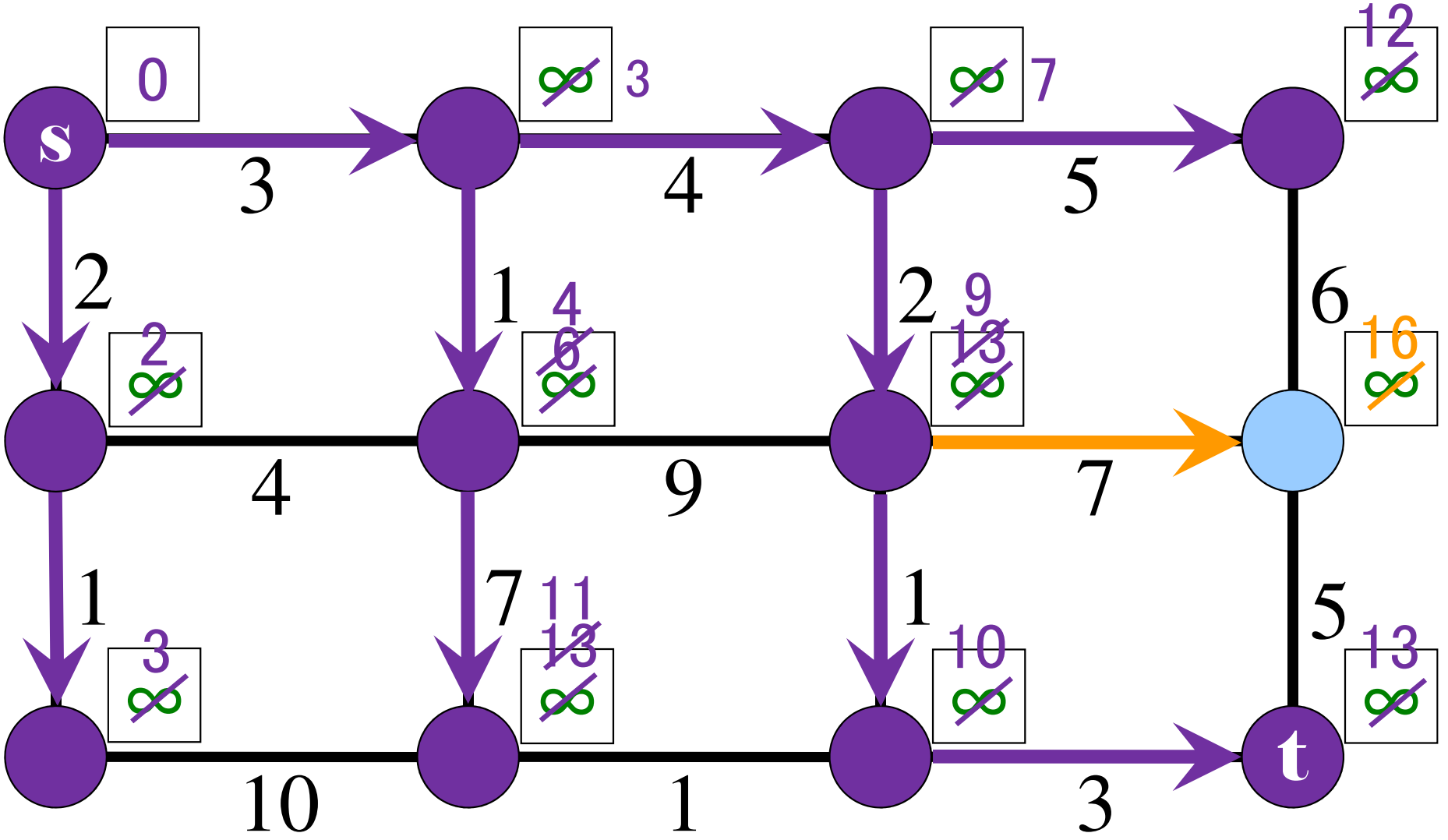
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

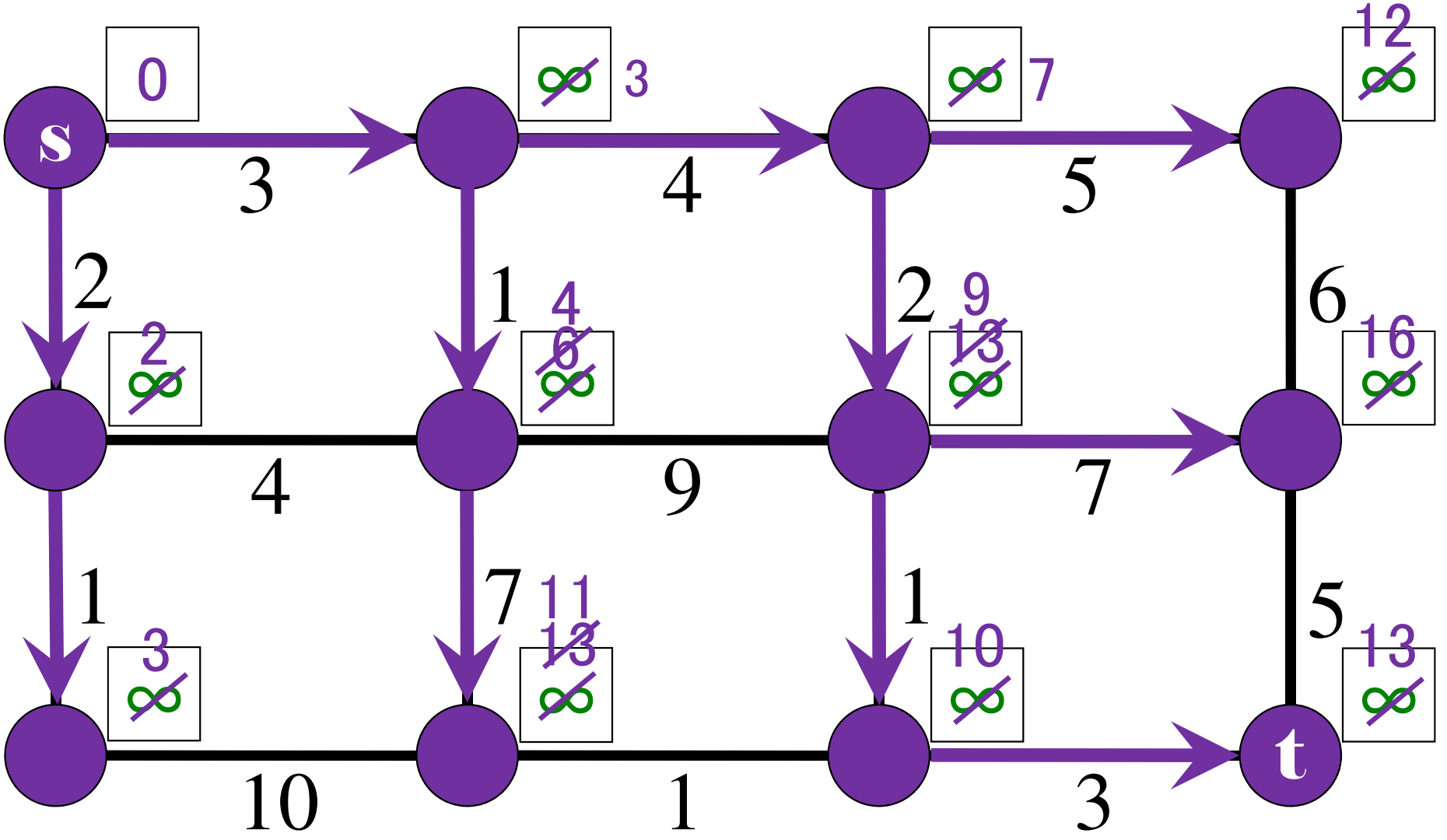


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

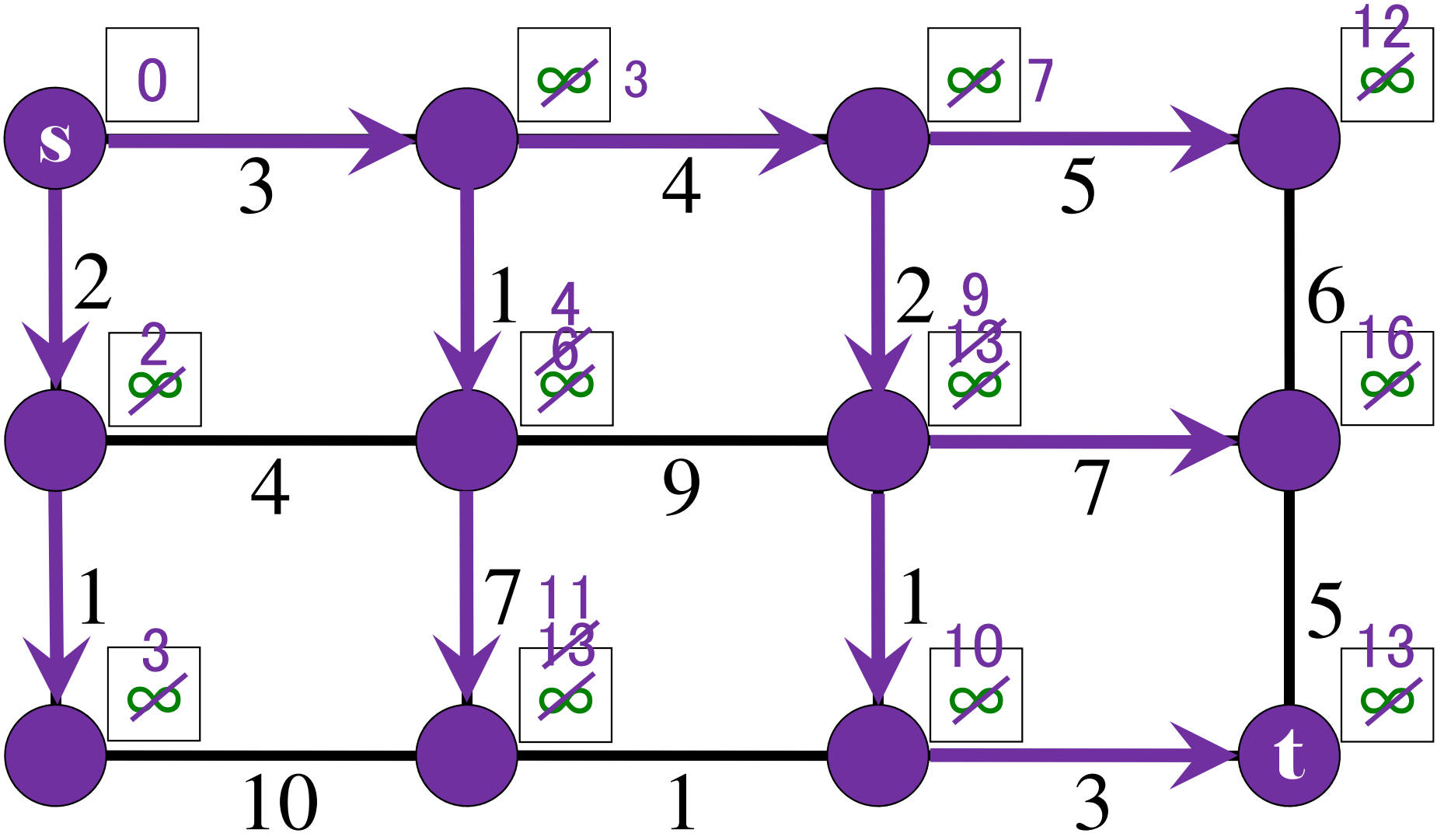
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



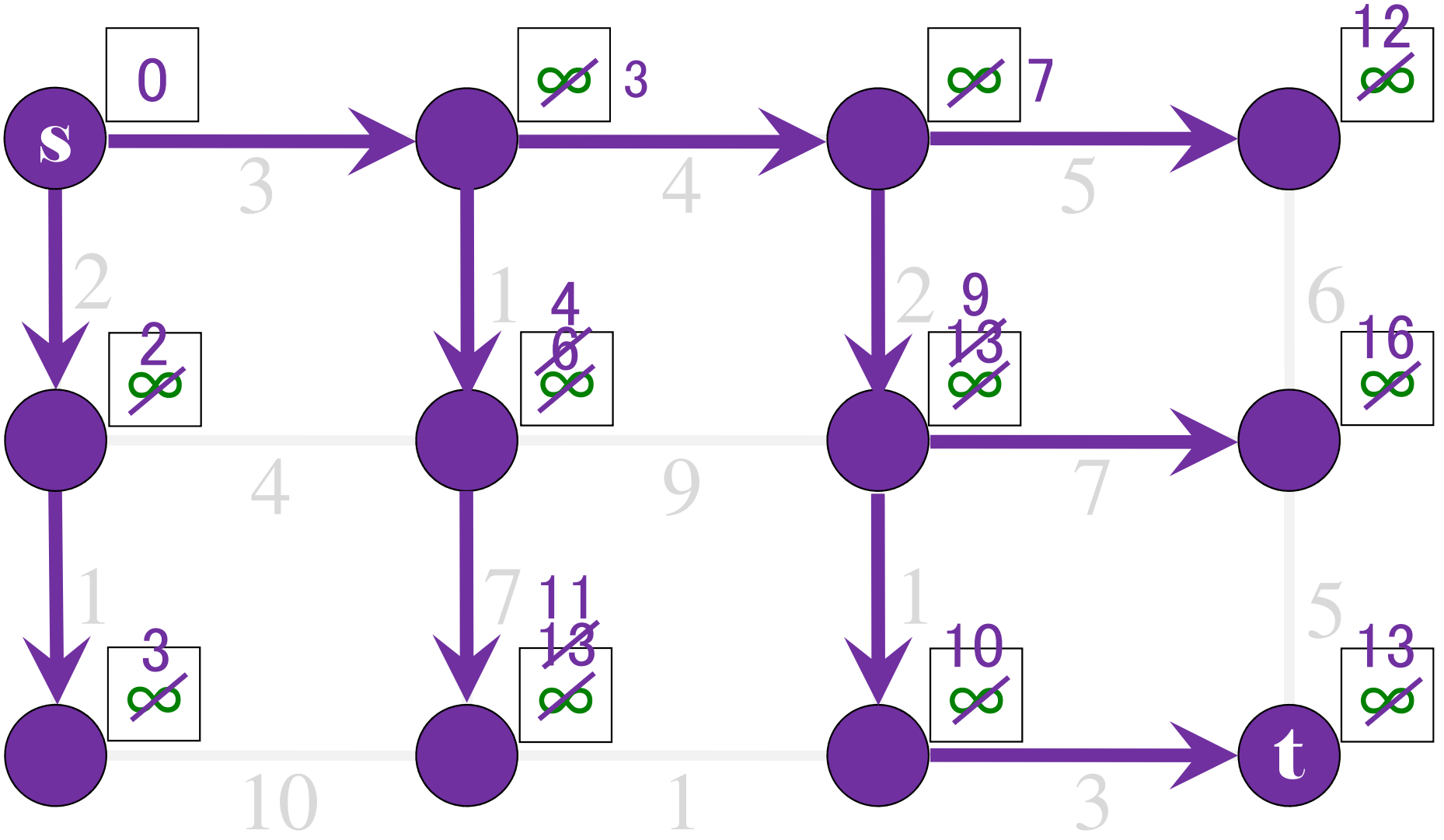
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



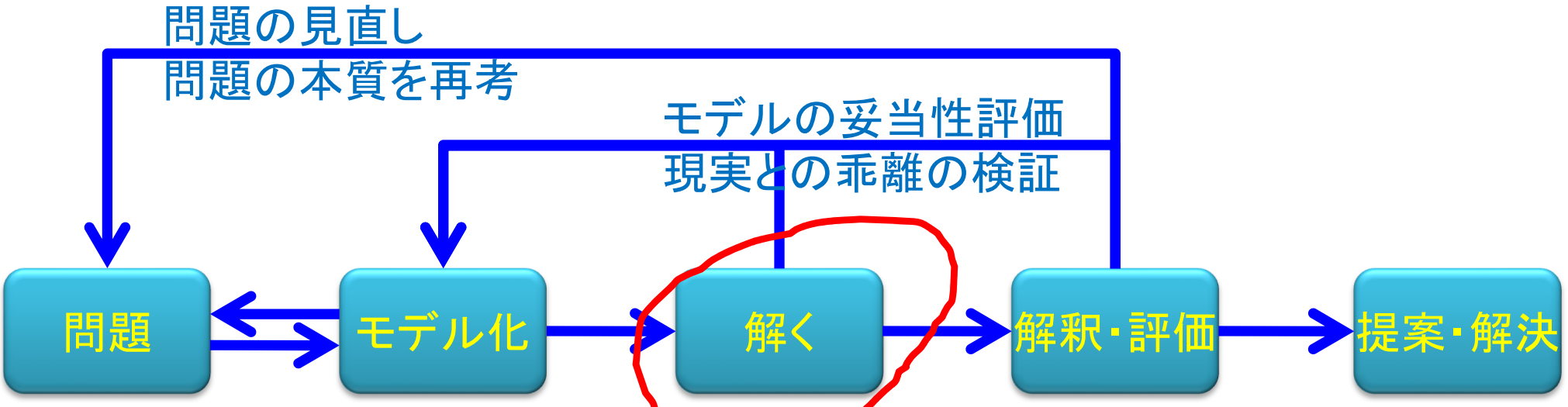
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

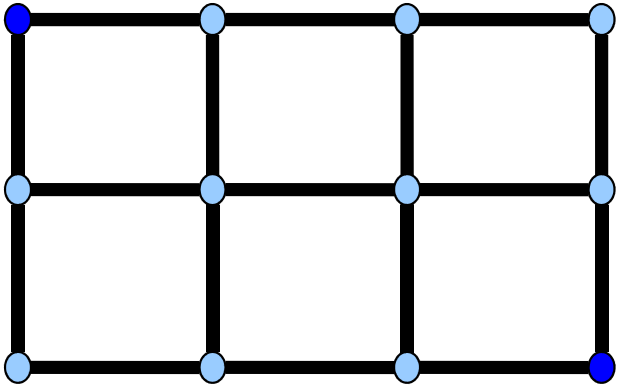
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

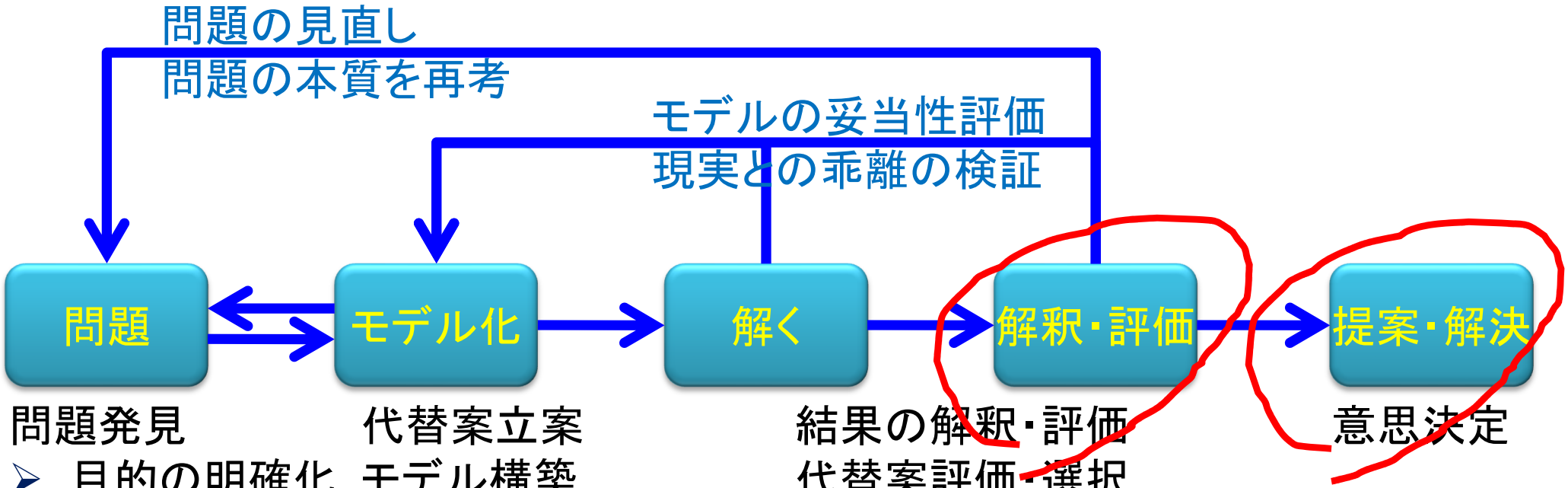
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

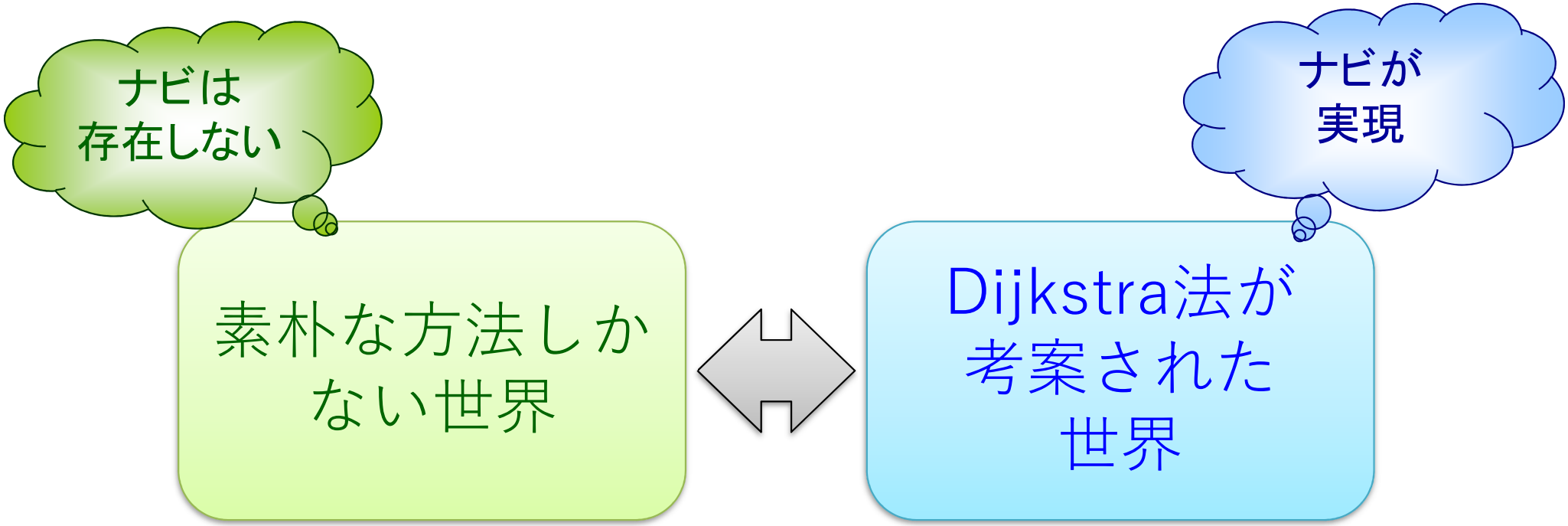
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

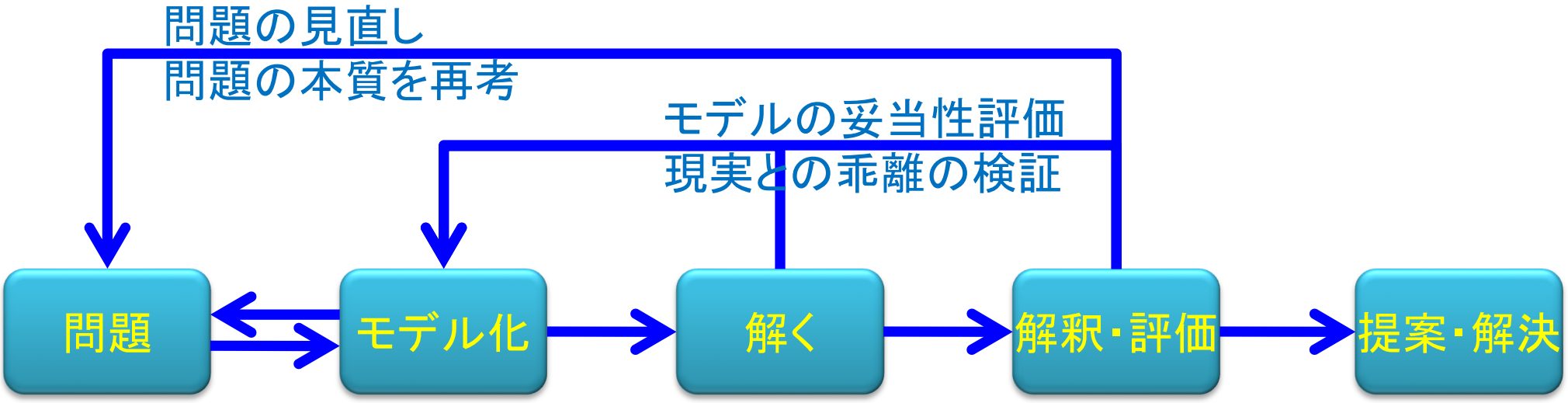
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



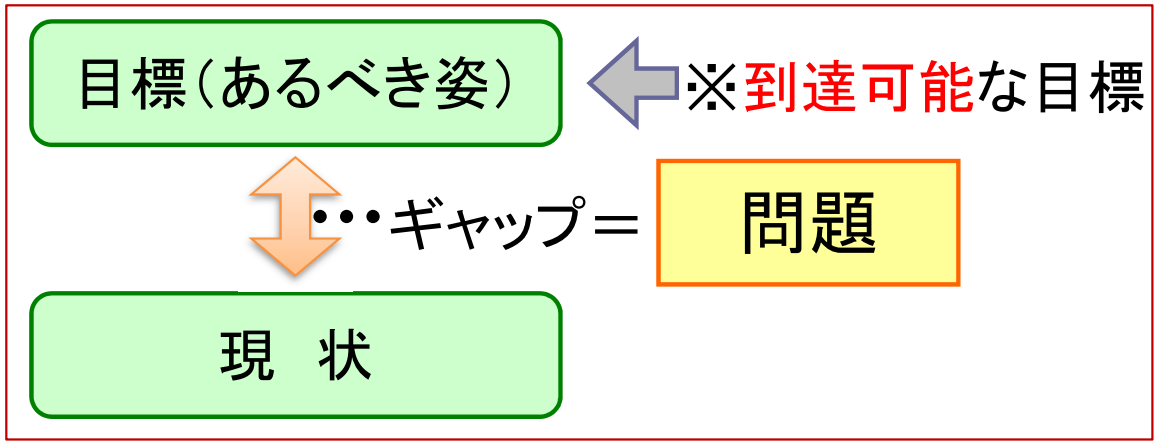
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

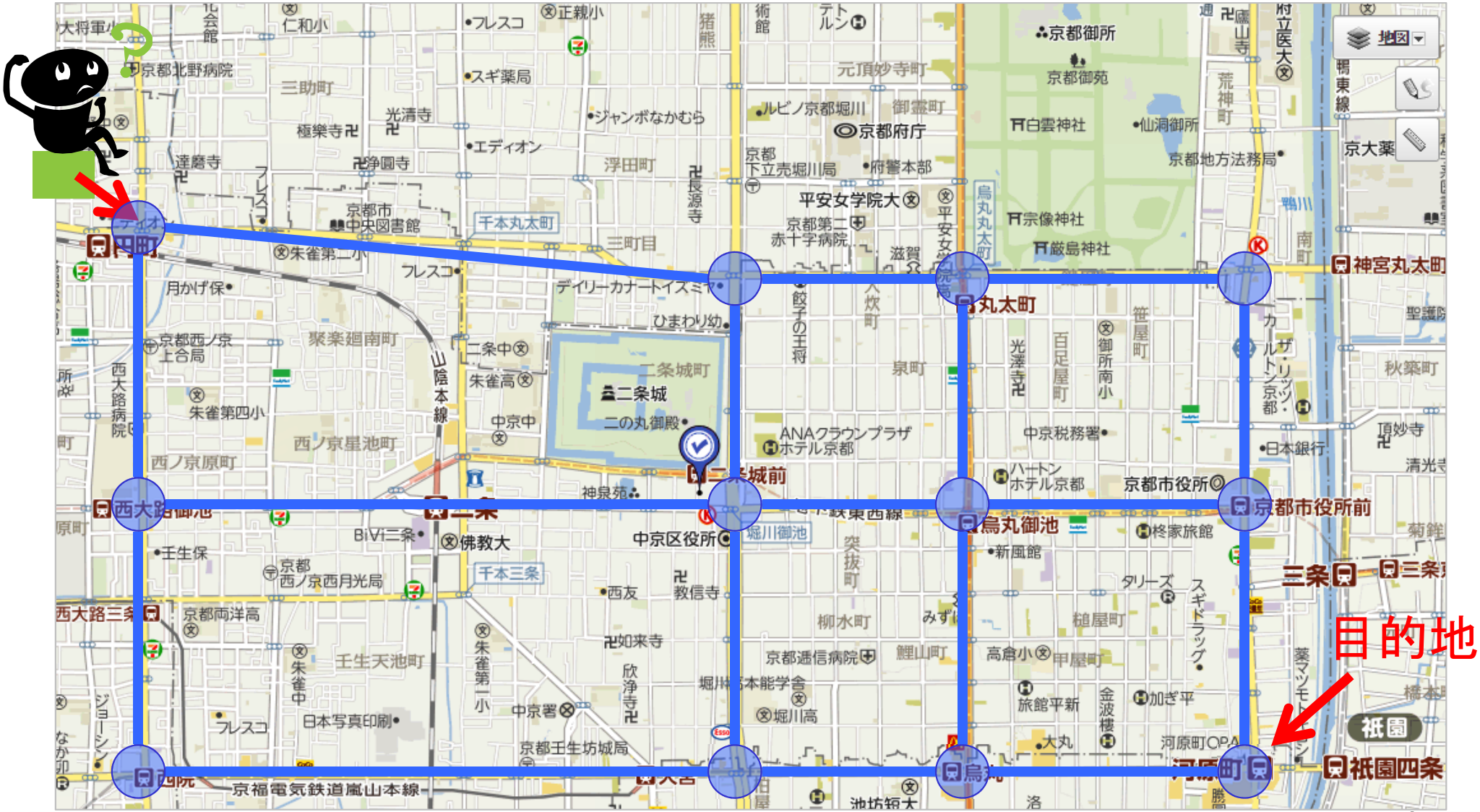
意思決定

問題の定義



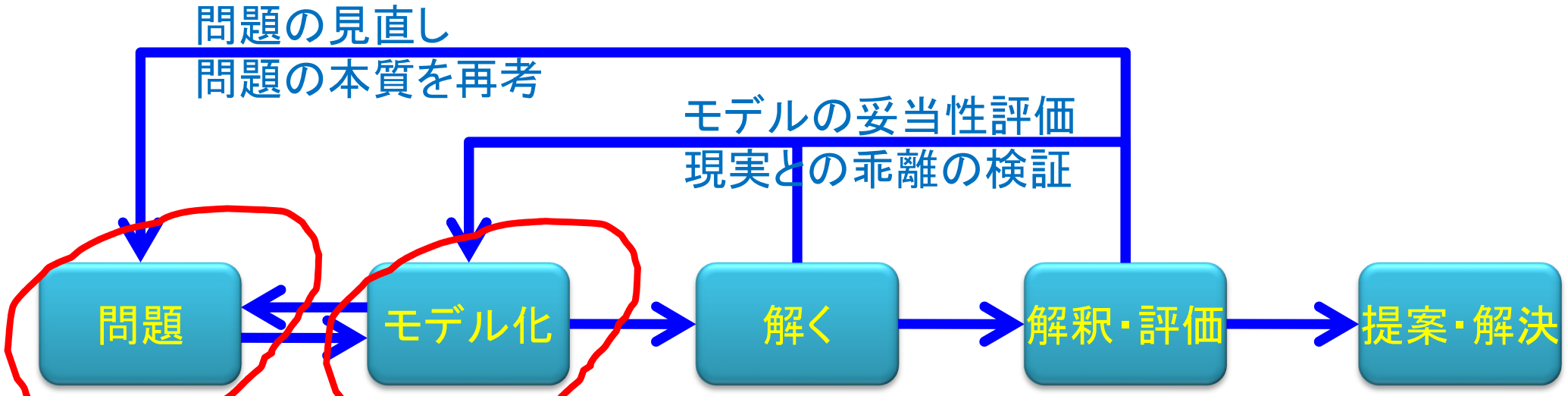
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

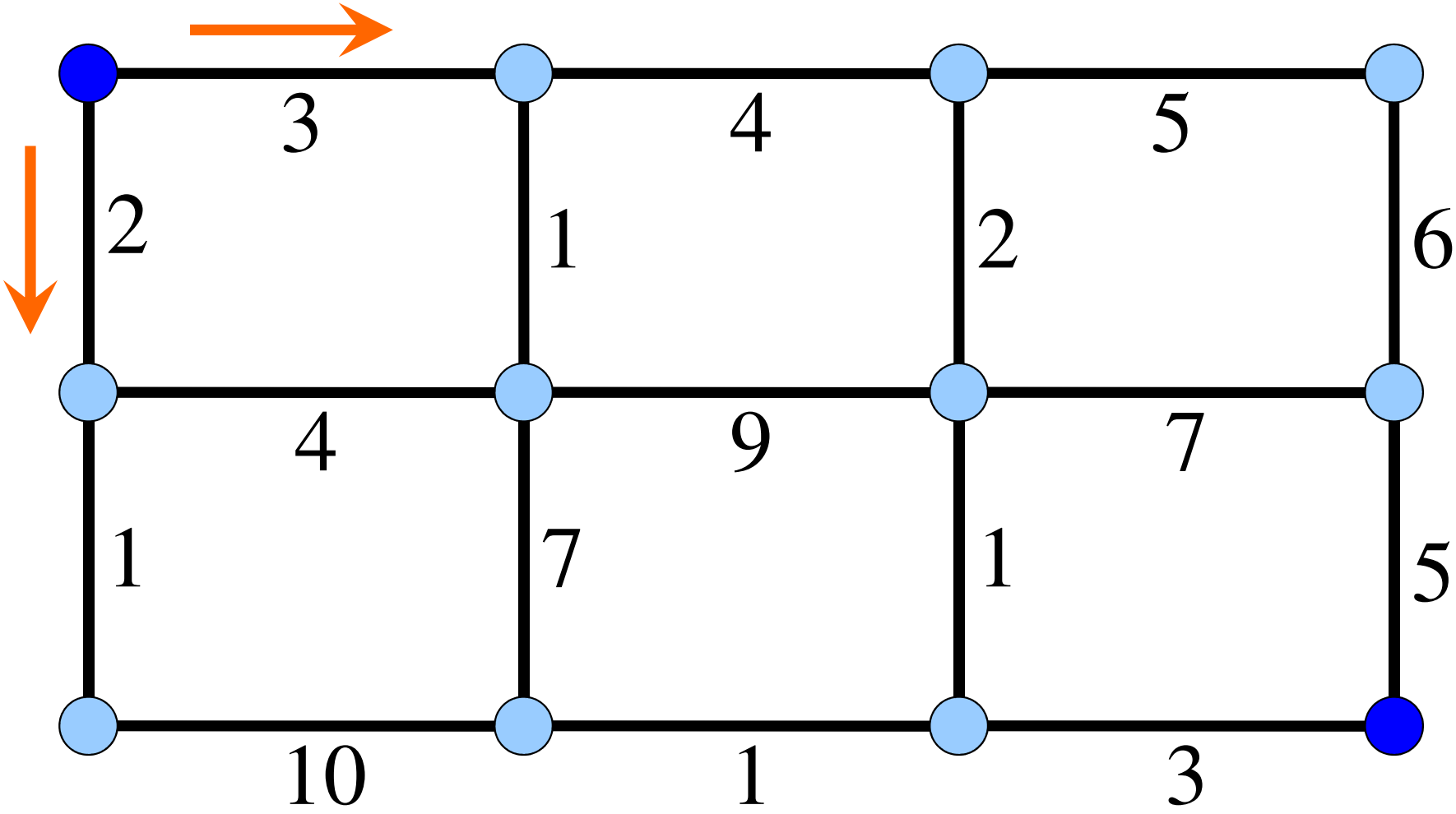
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

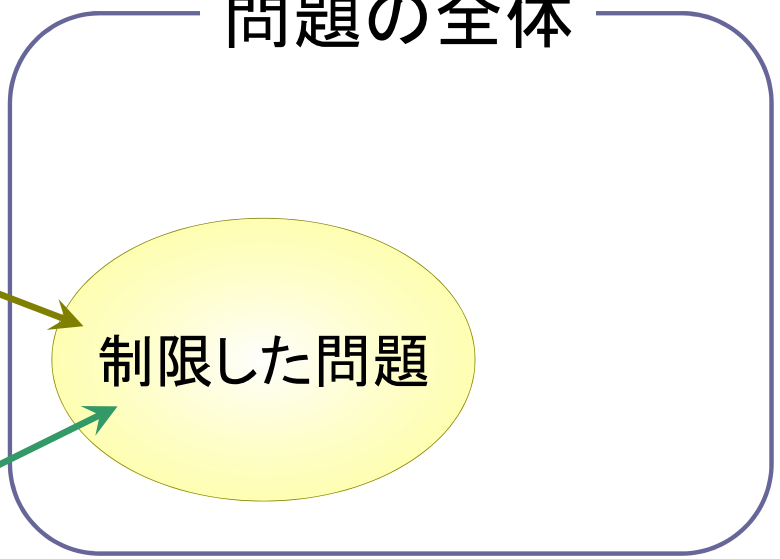


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

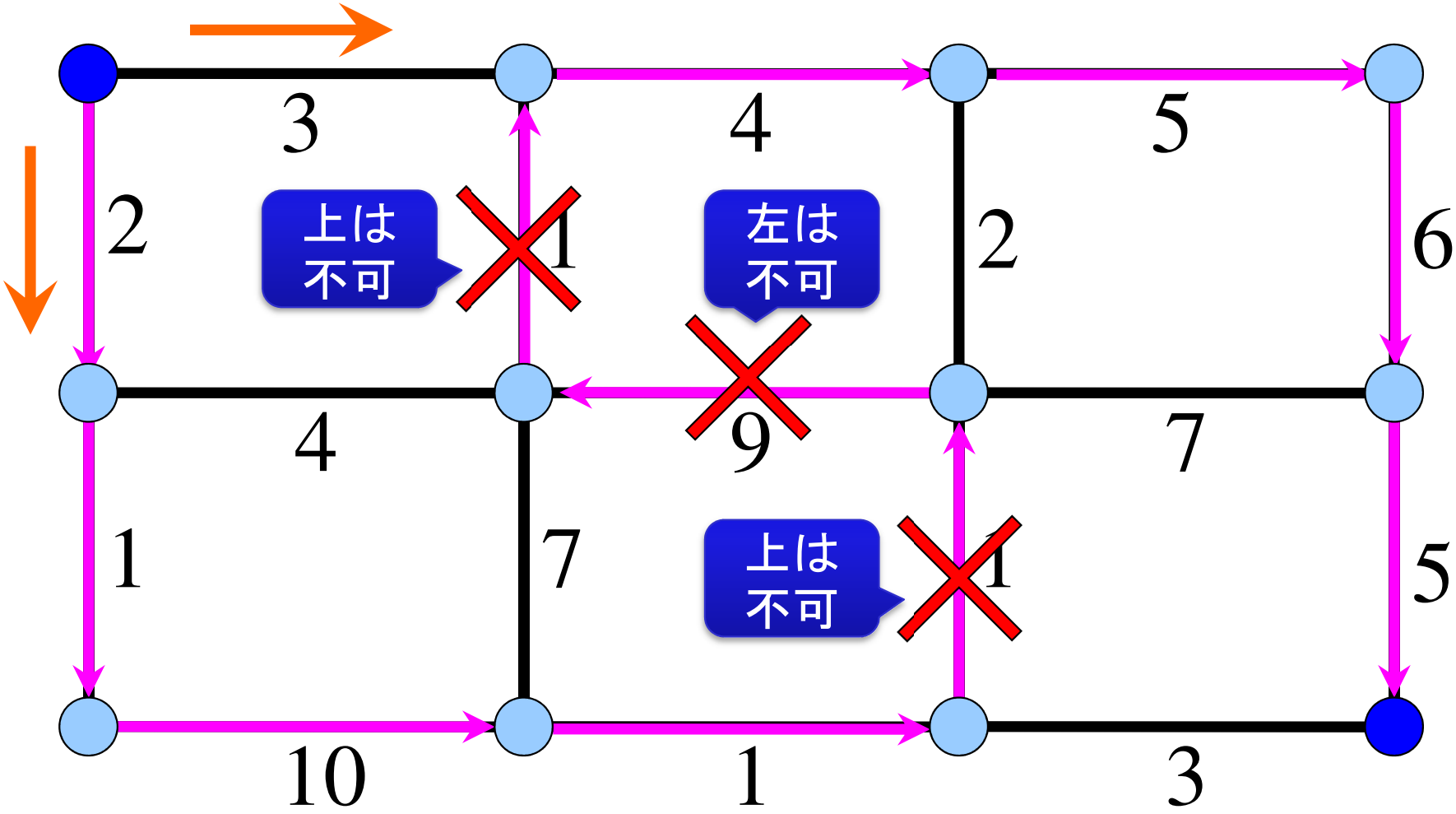
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

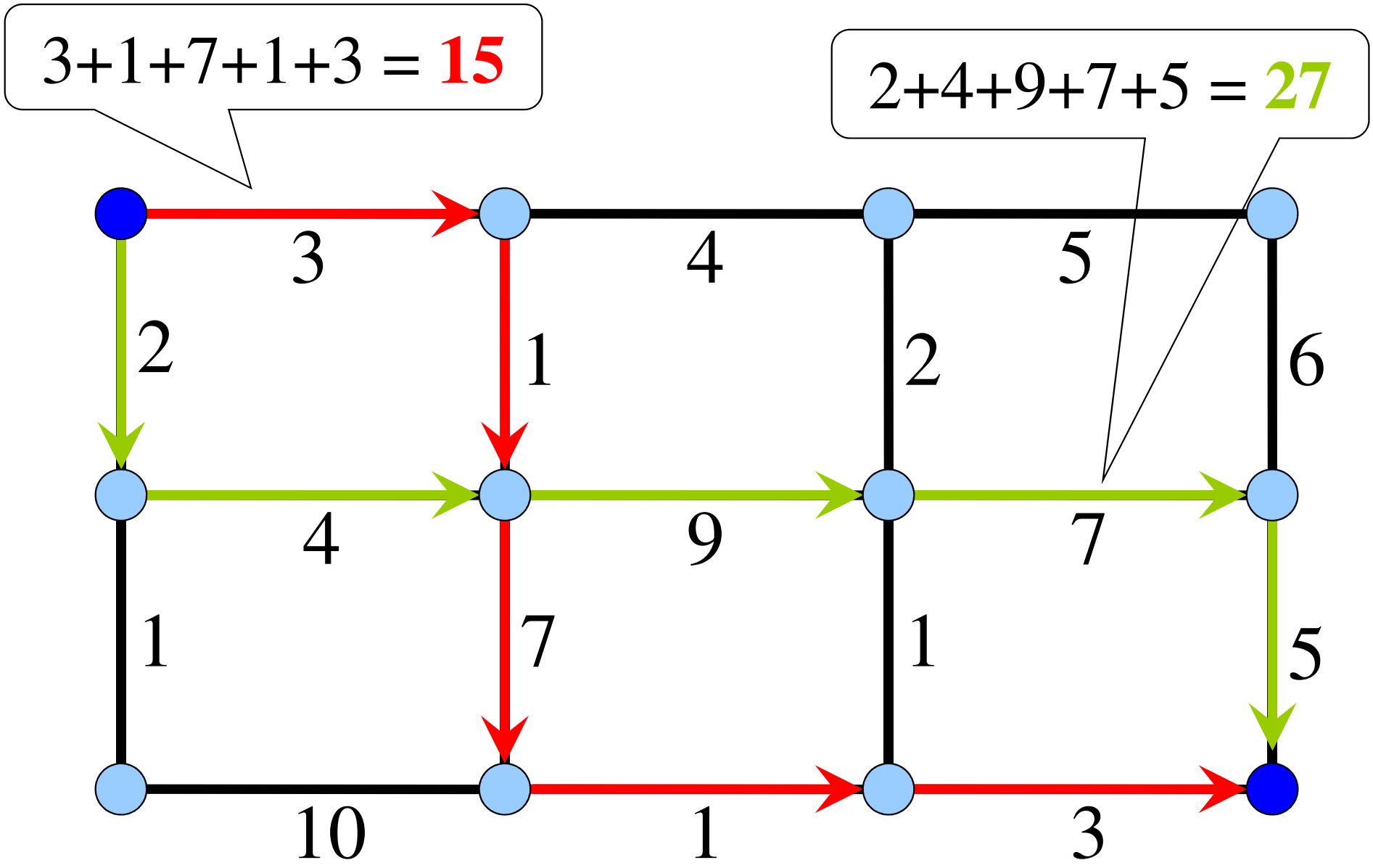
難しいなら易しくすればいいのさ！

制限した問題

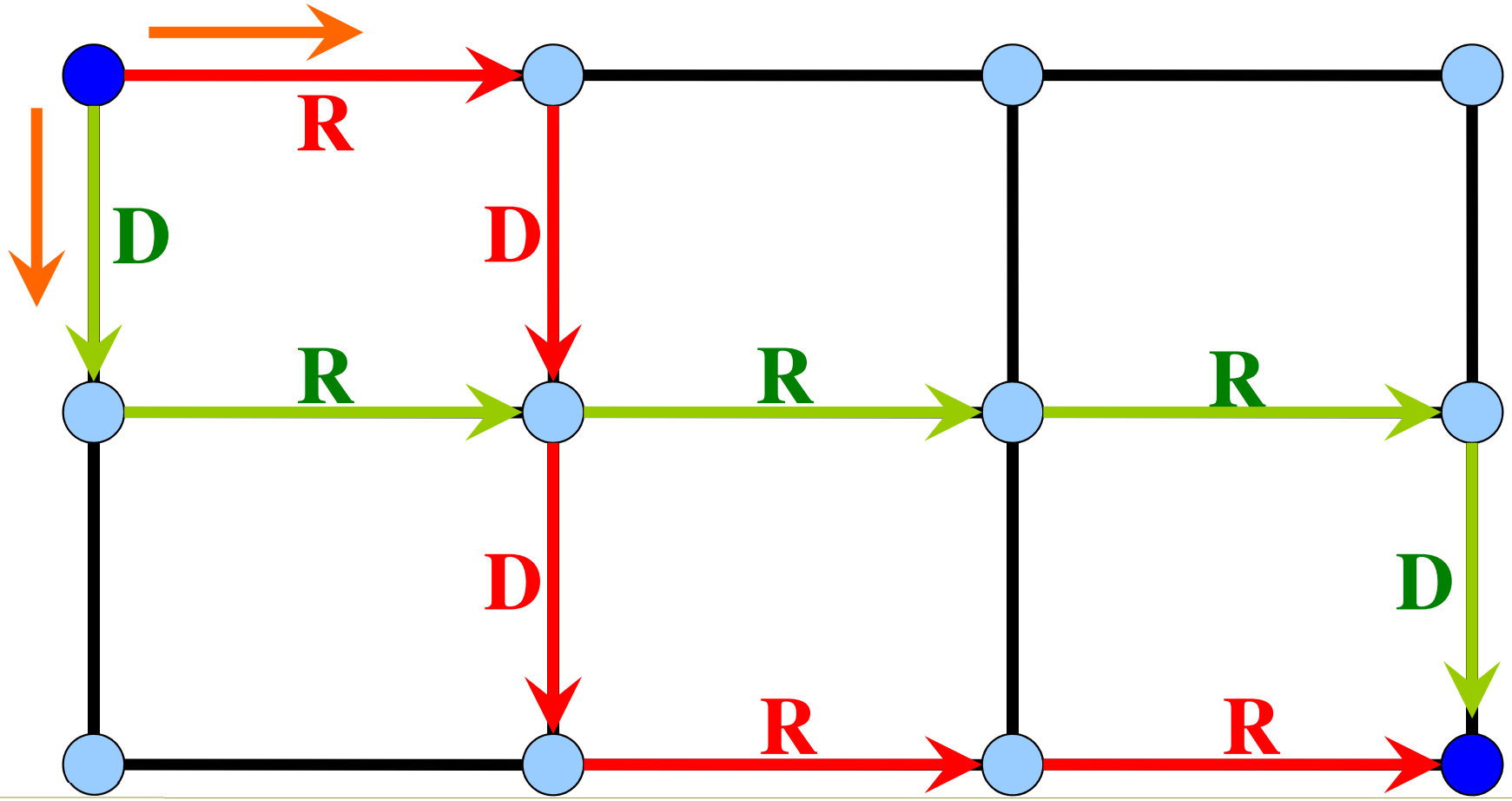
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



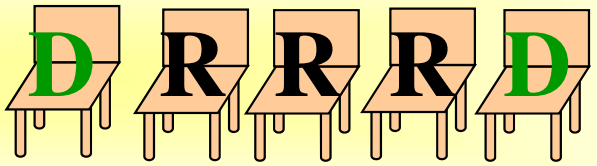
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



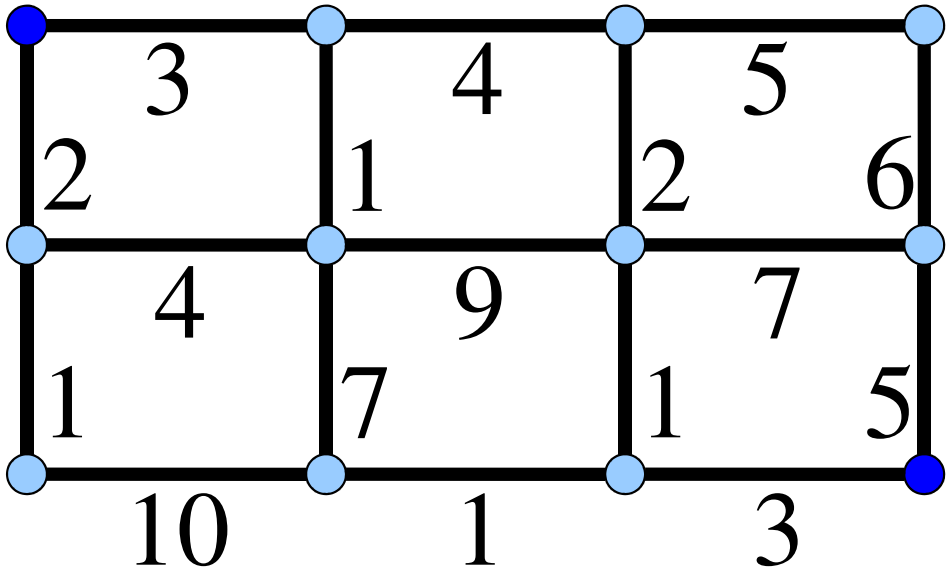
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

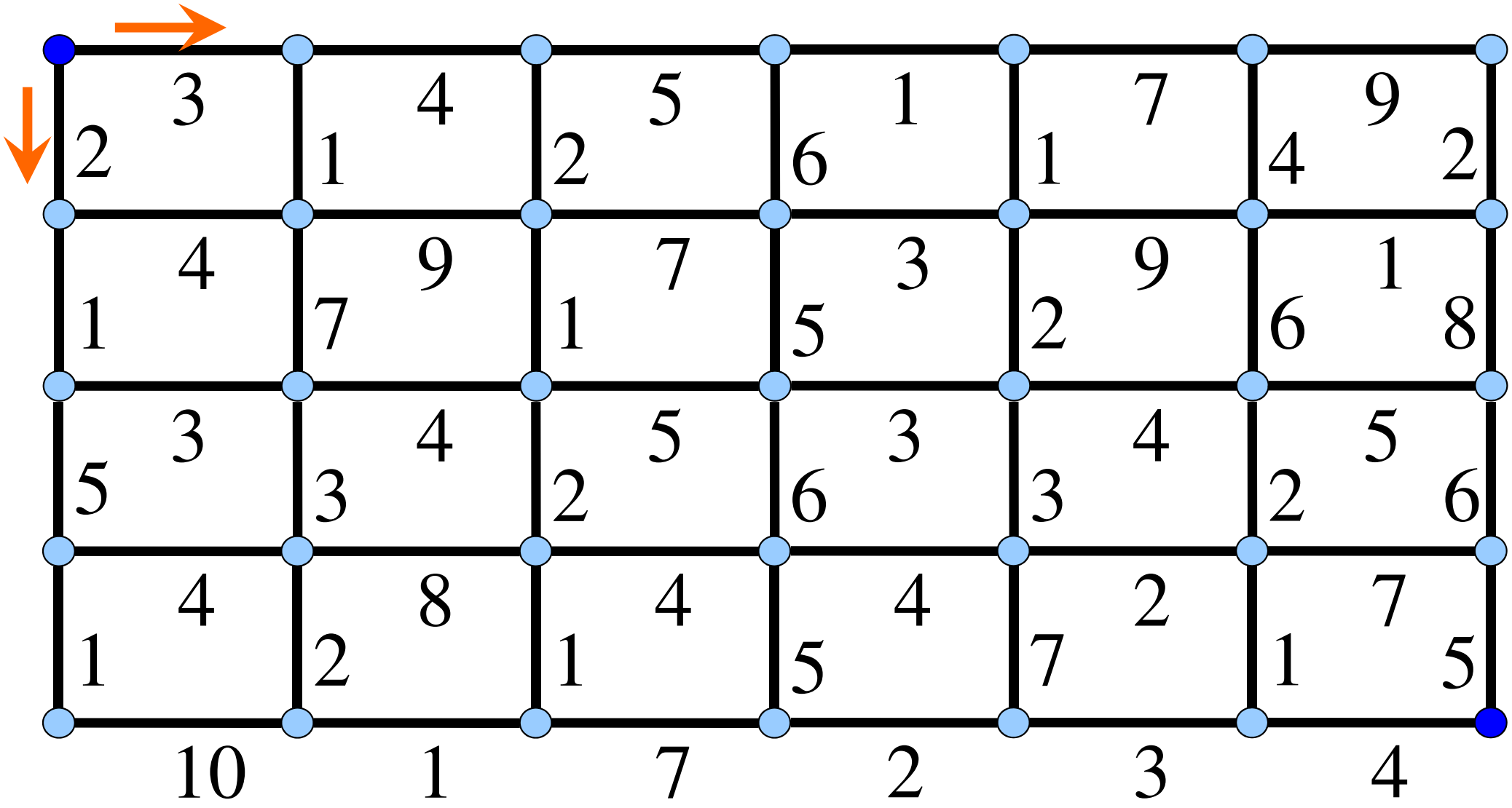
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

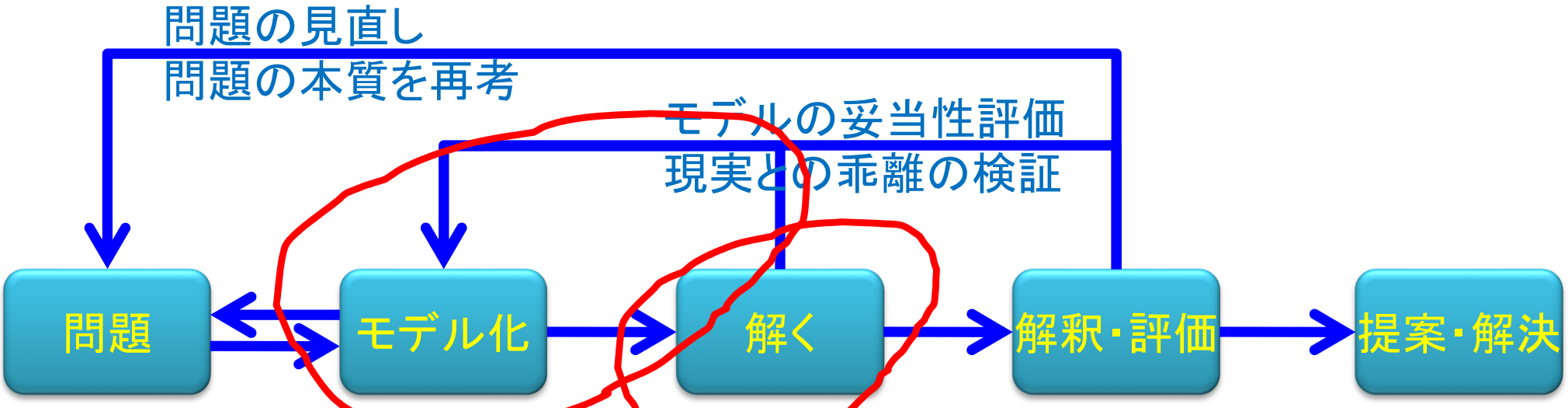
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

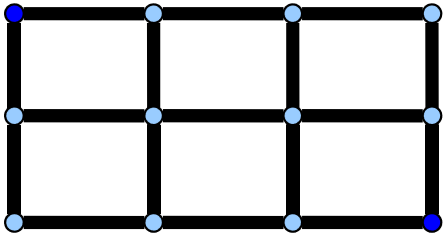
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

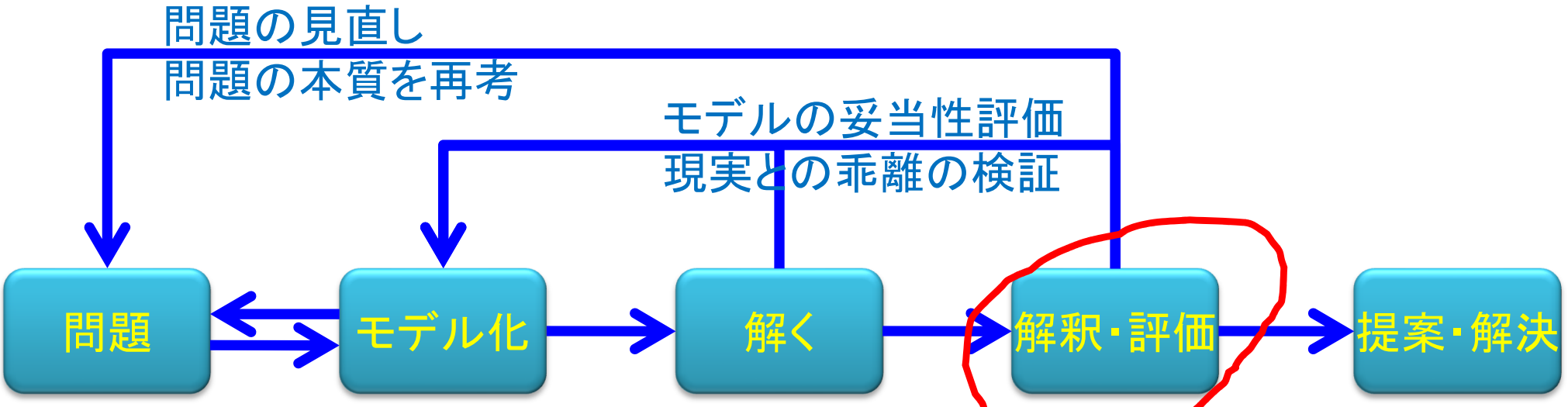
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

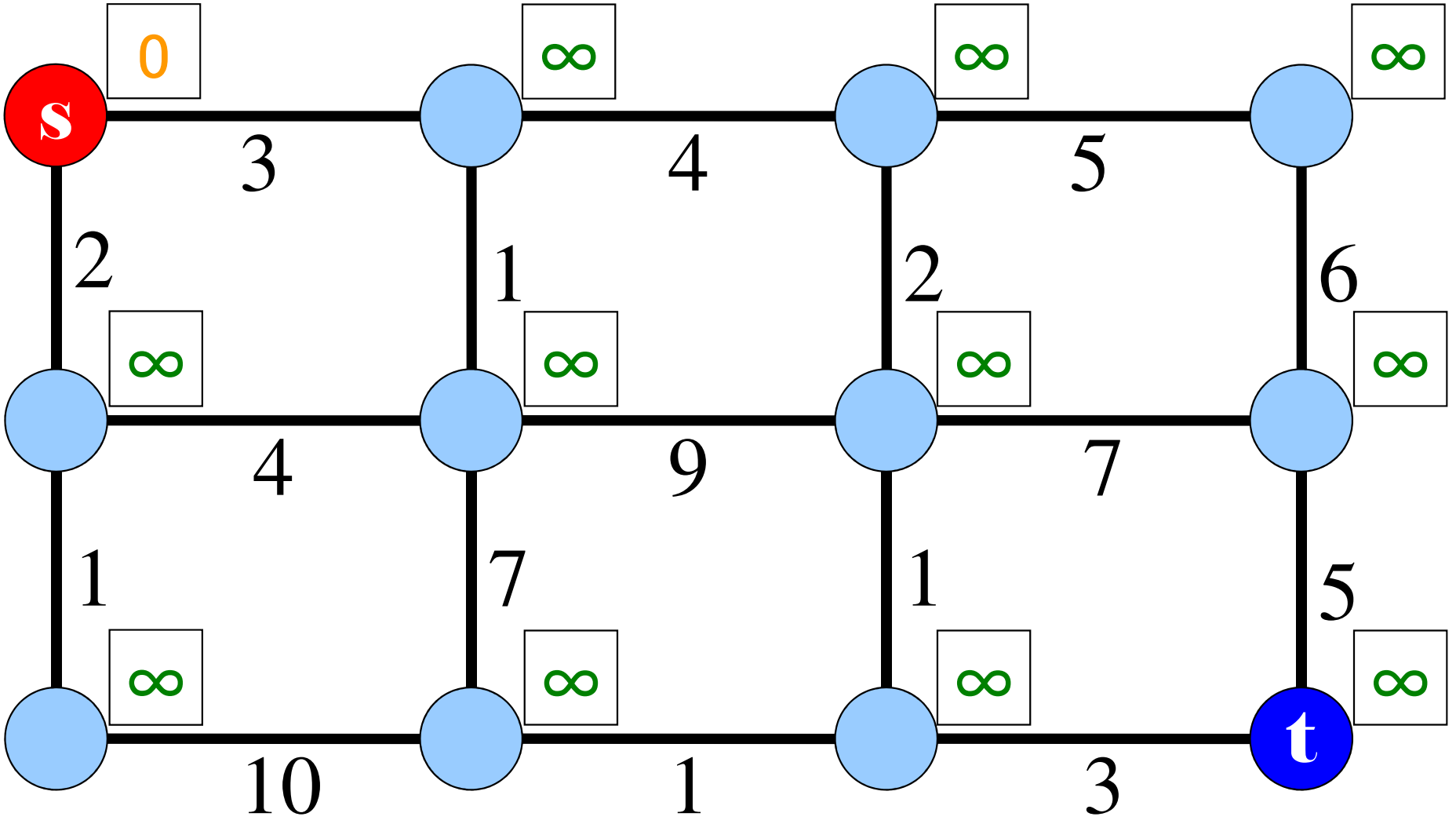
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

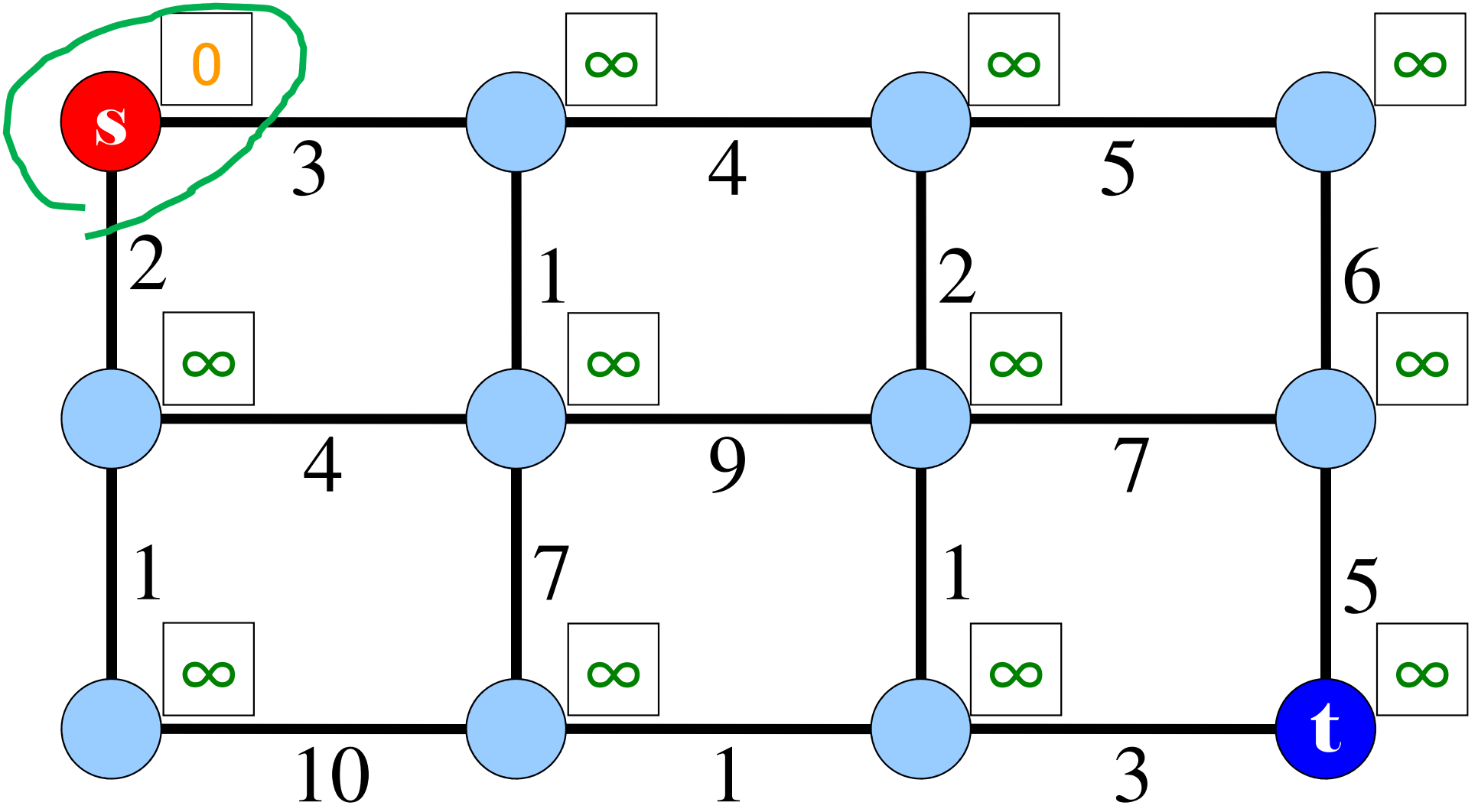
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



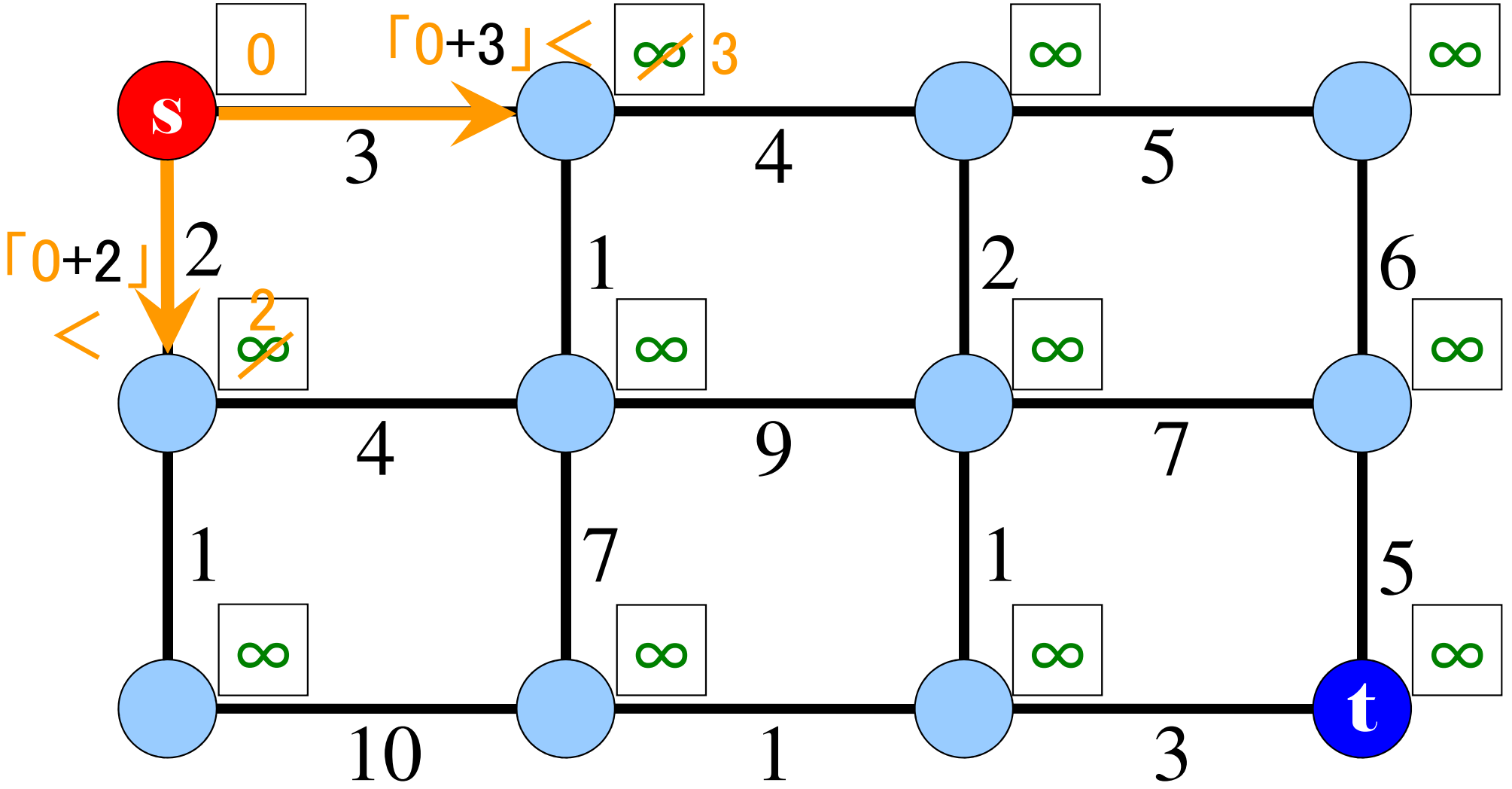
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

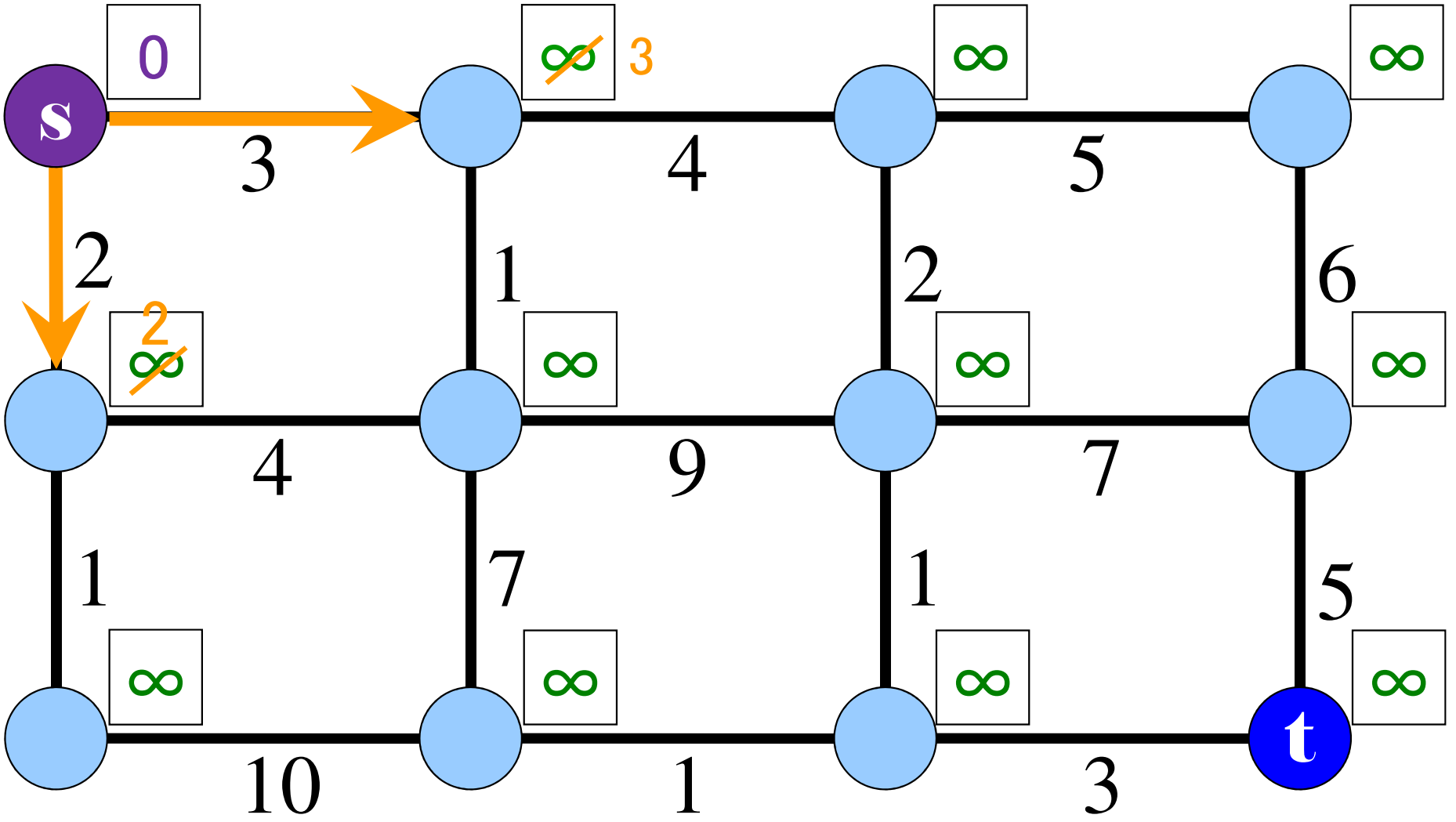


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

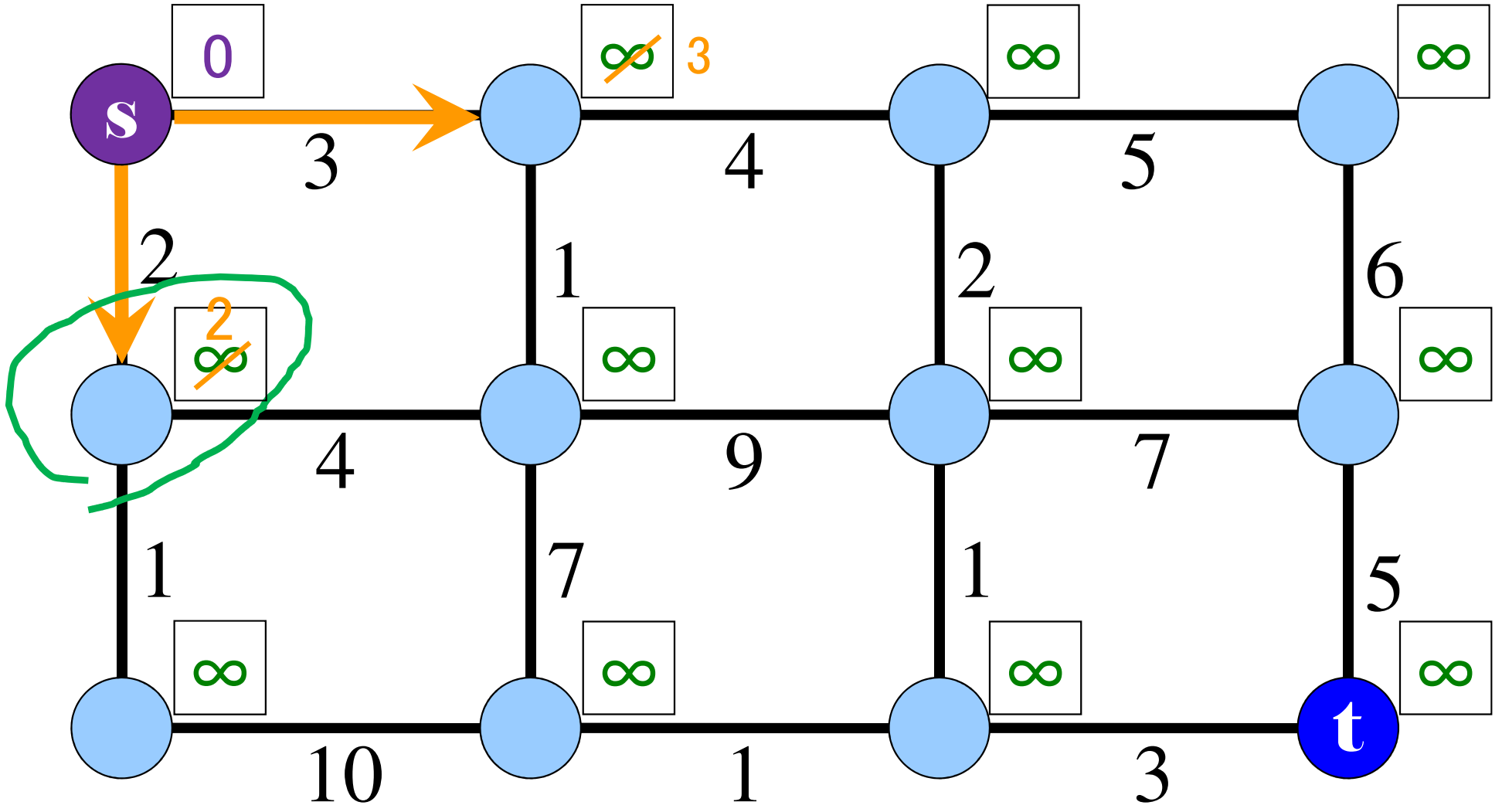
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



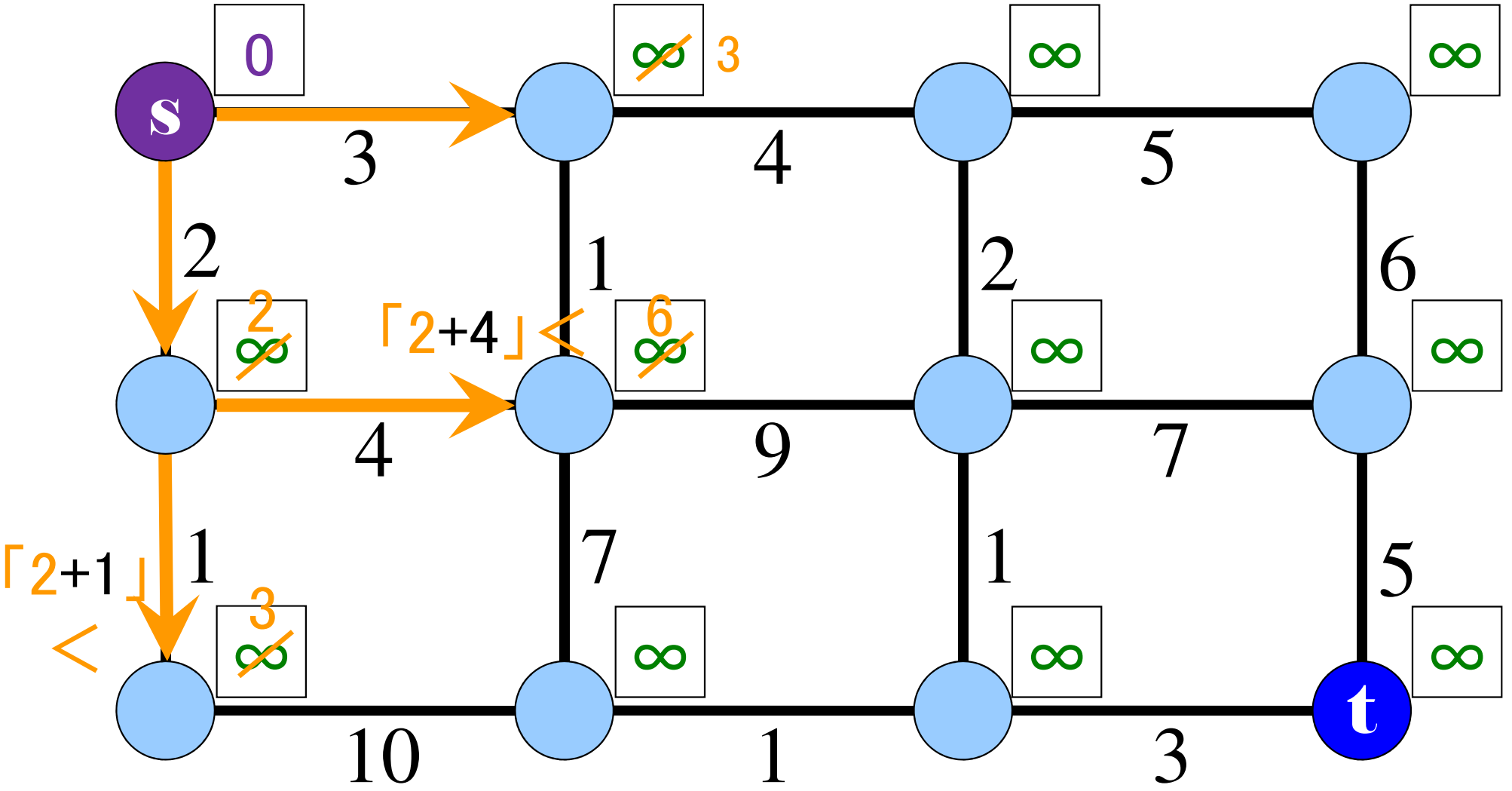
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

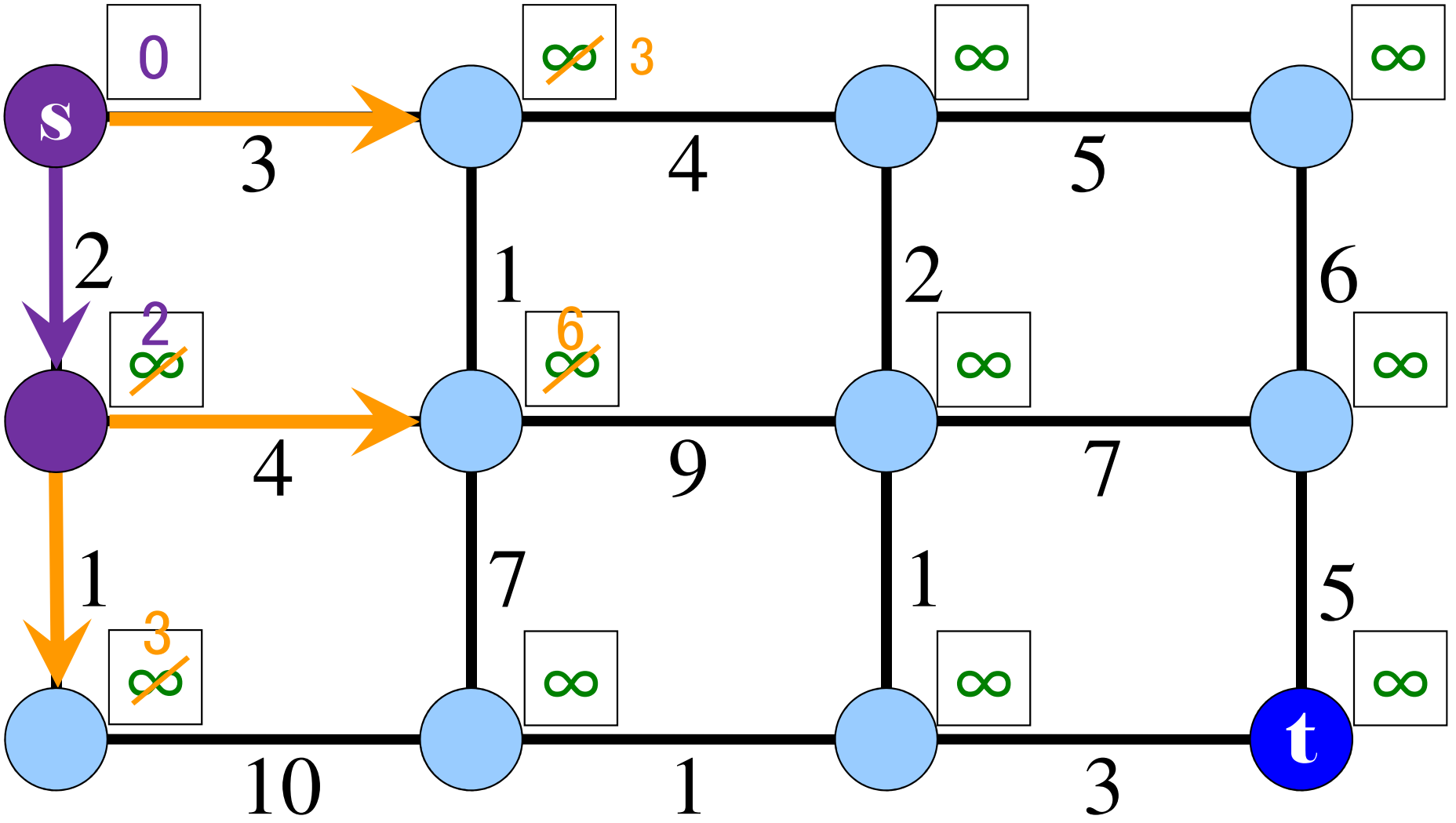


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

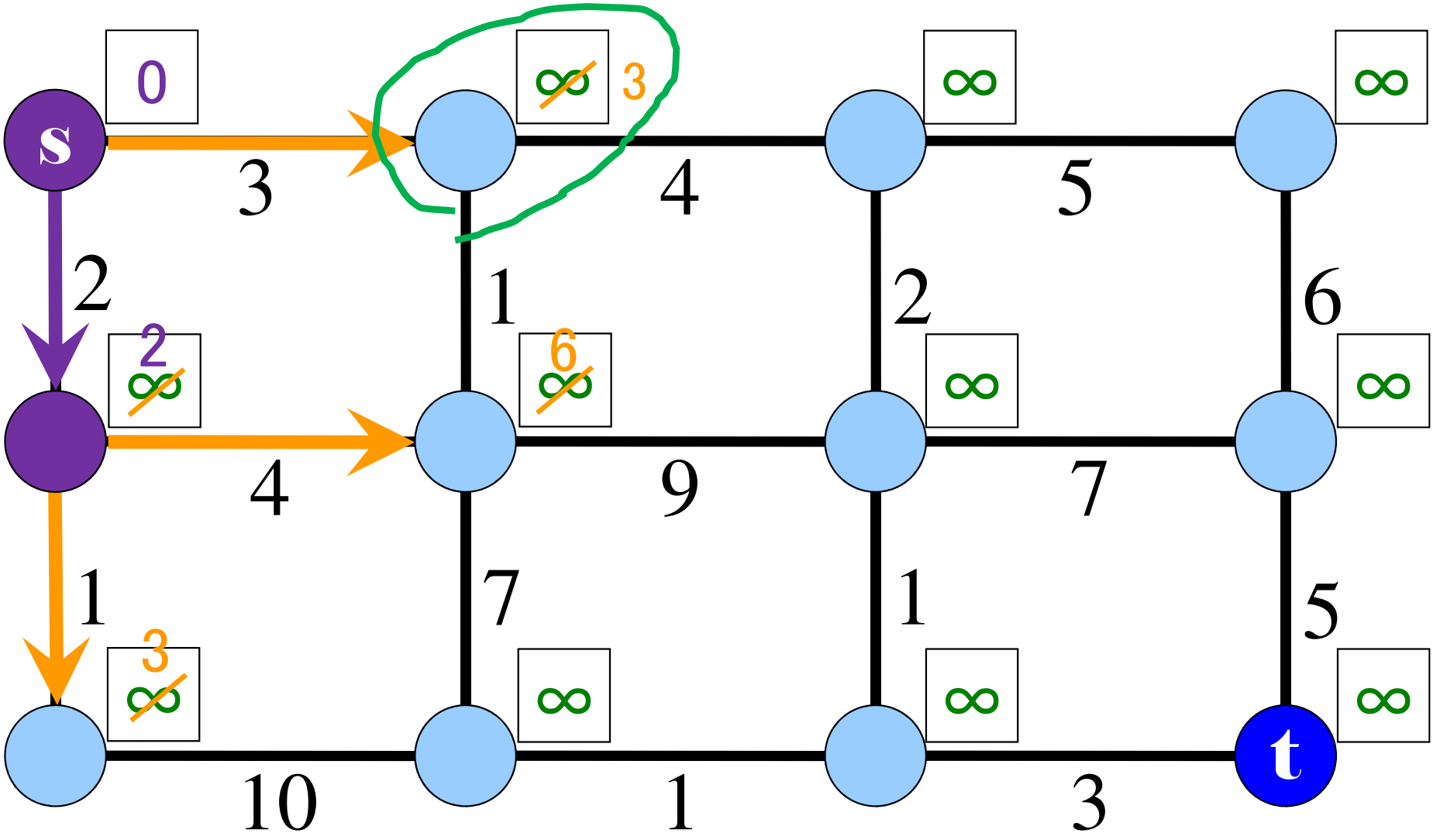
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



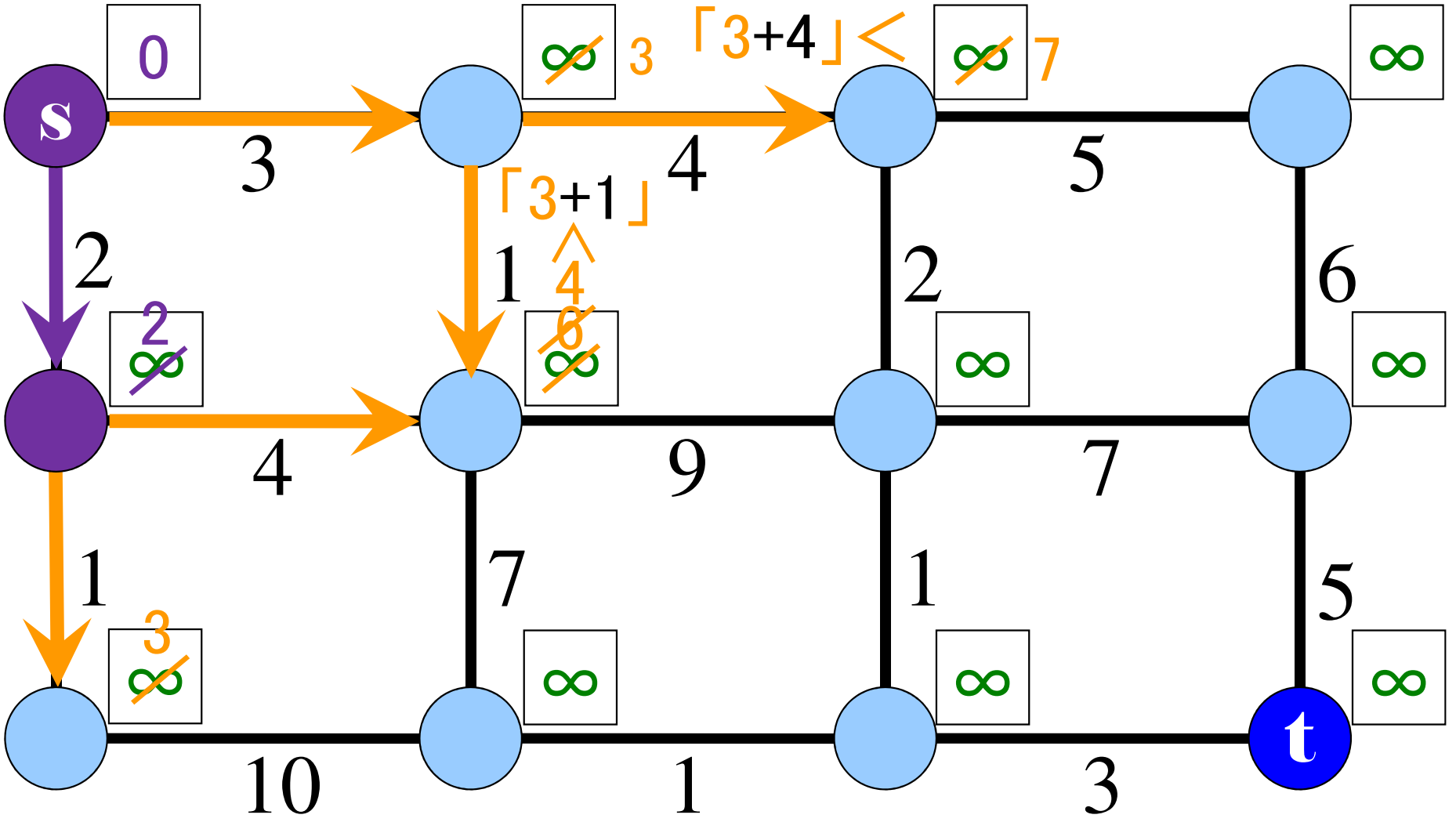
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

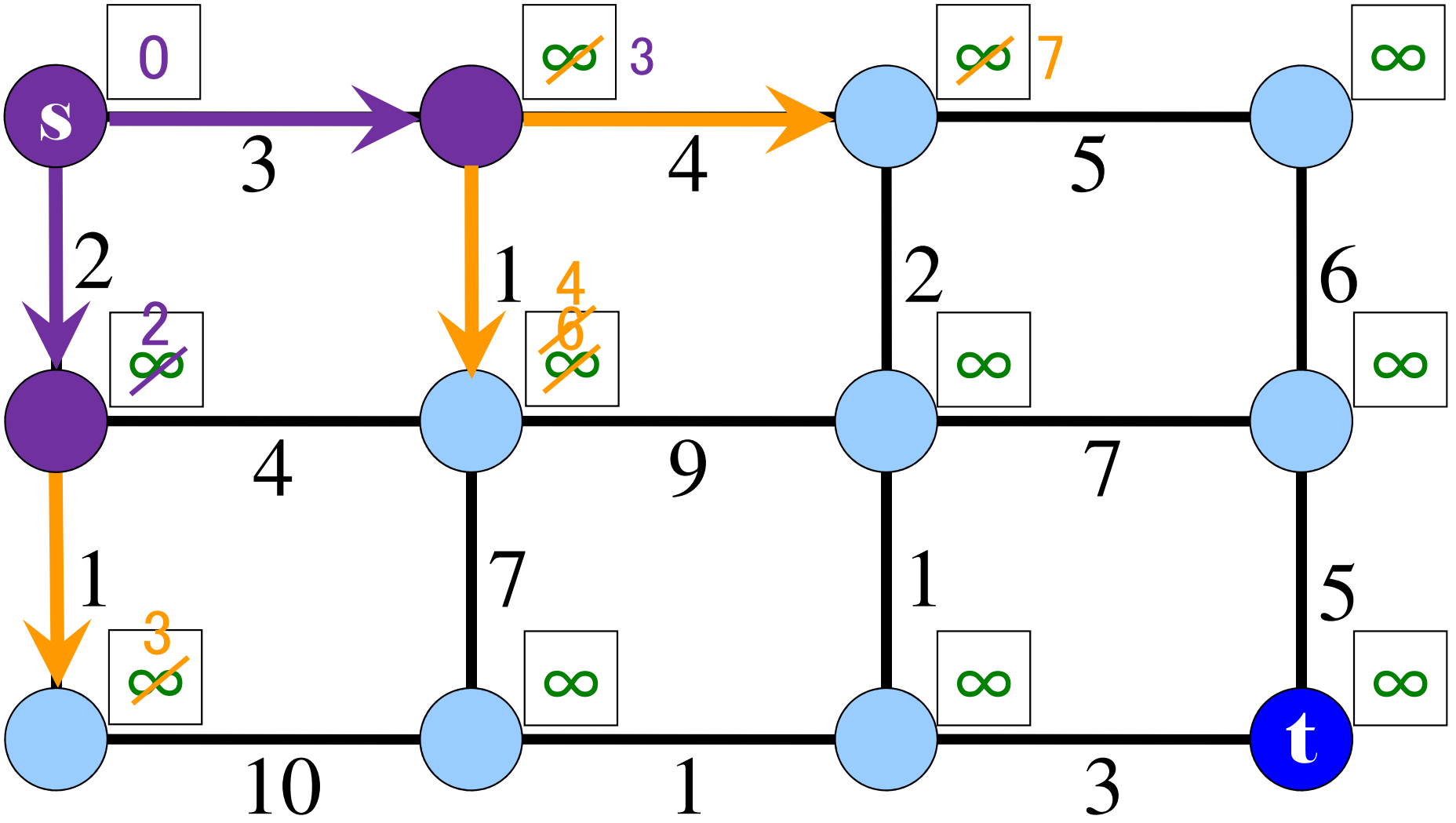


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

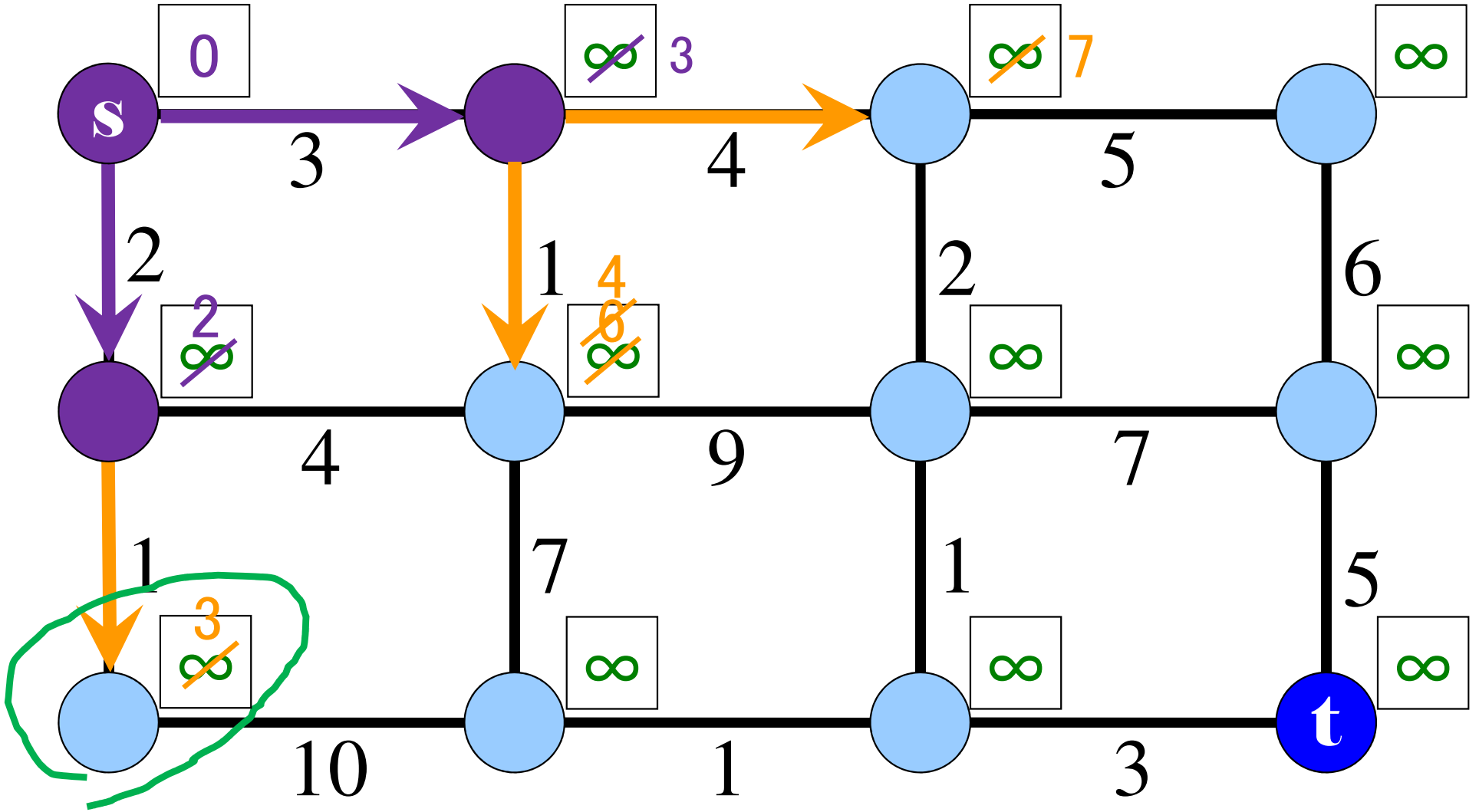
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



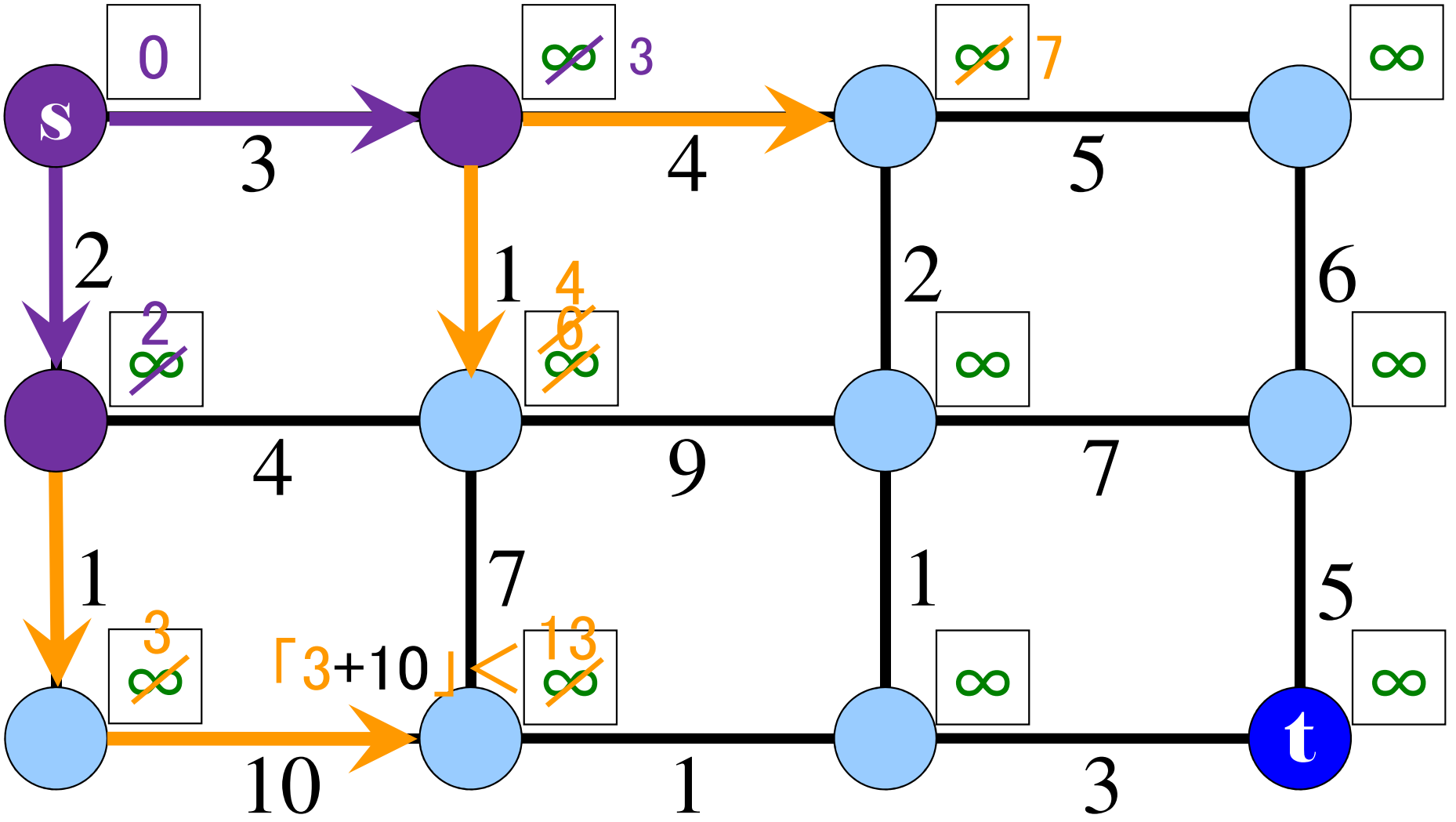
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

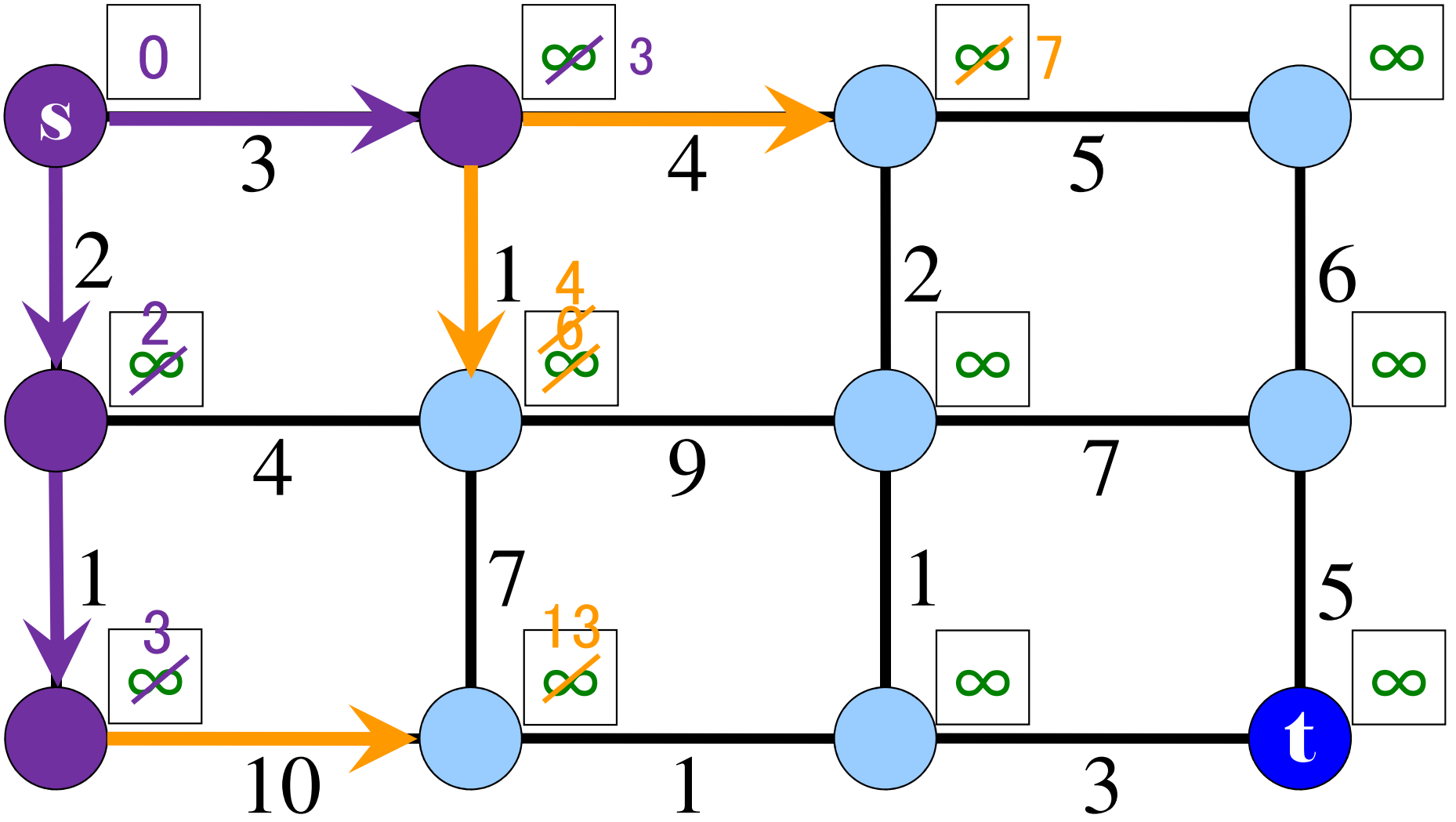


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

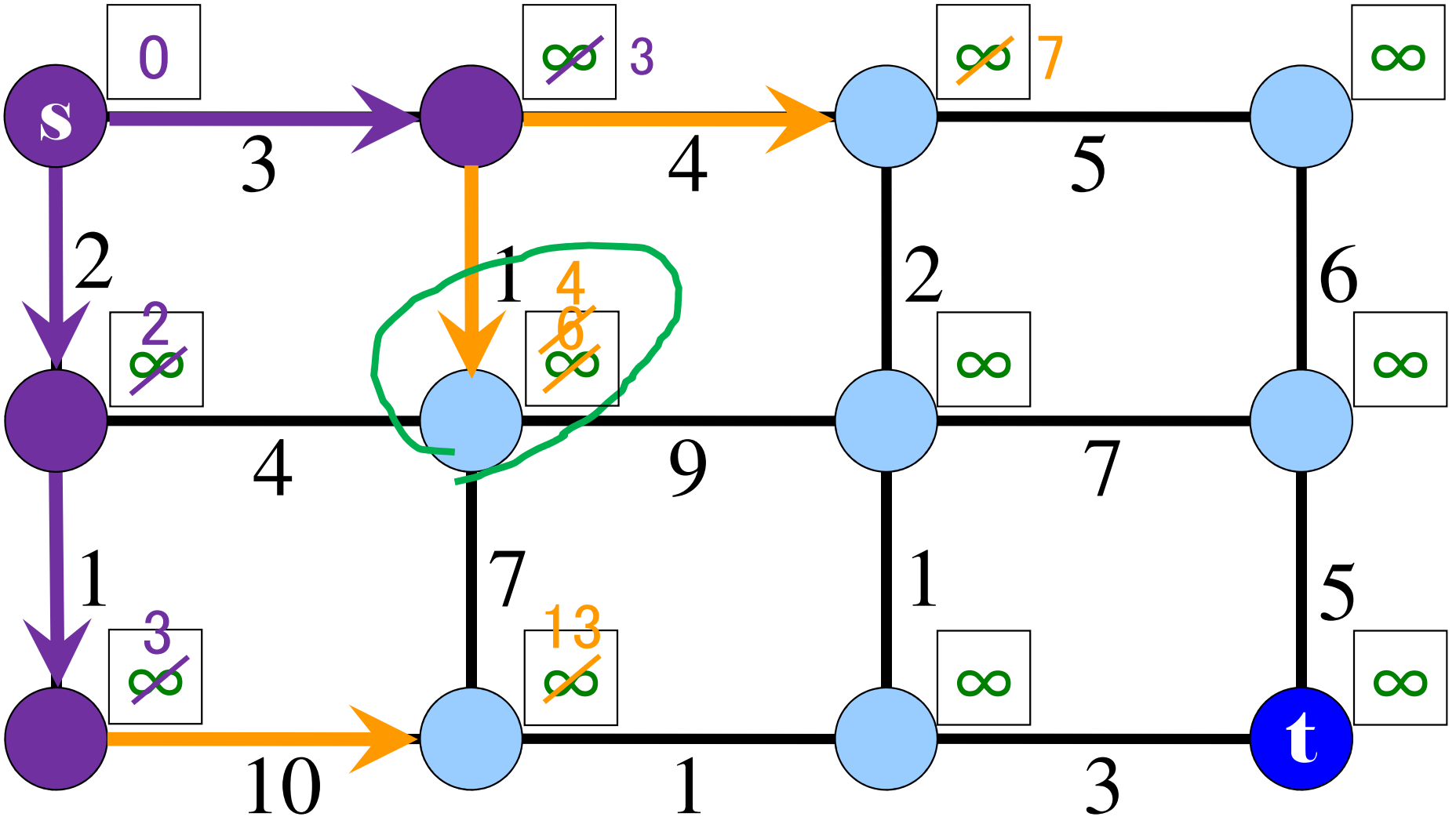
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



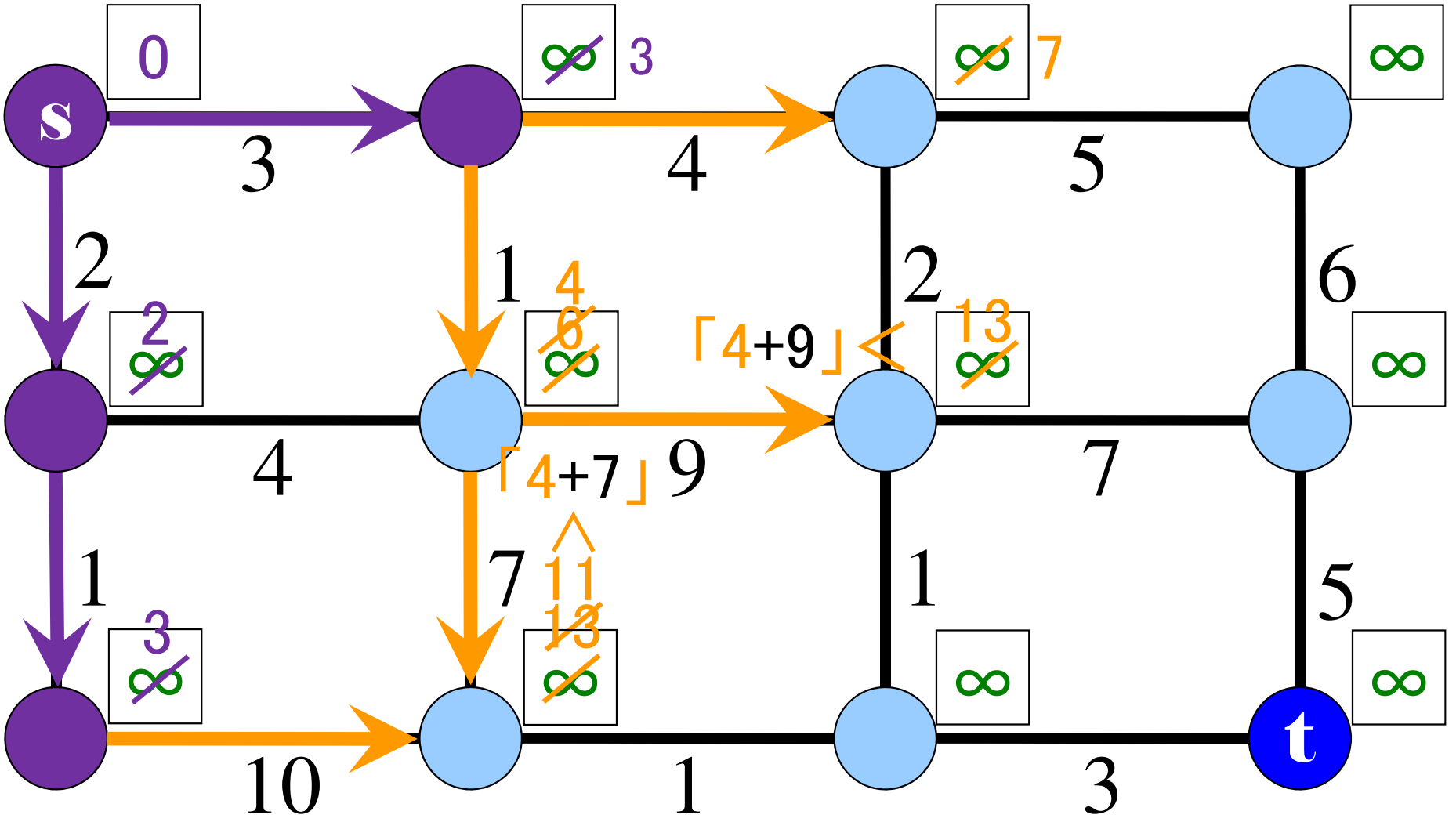
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

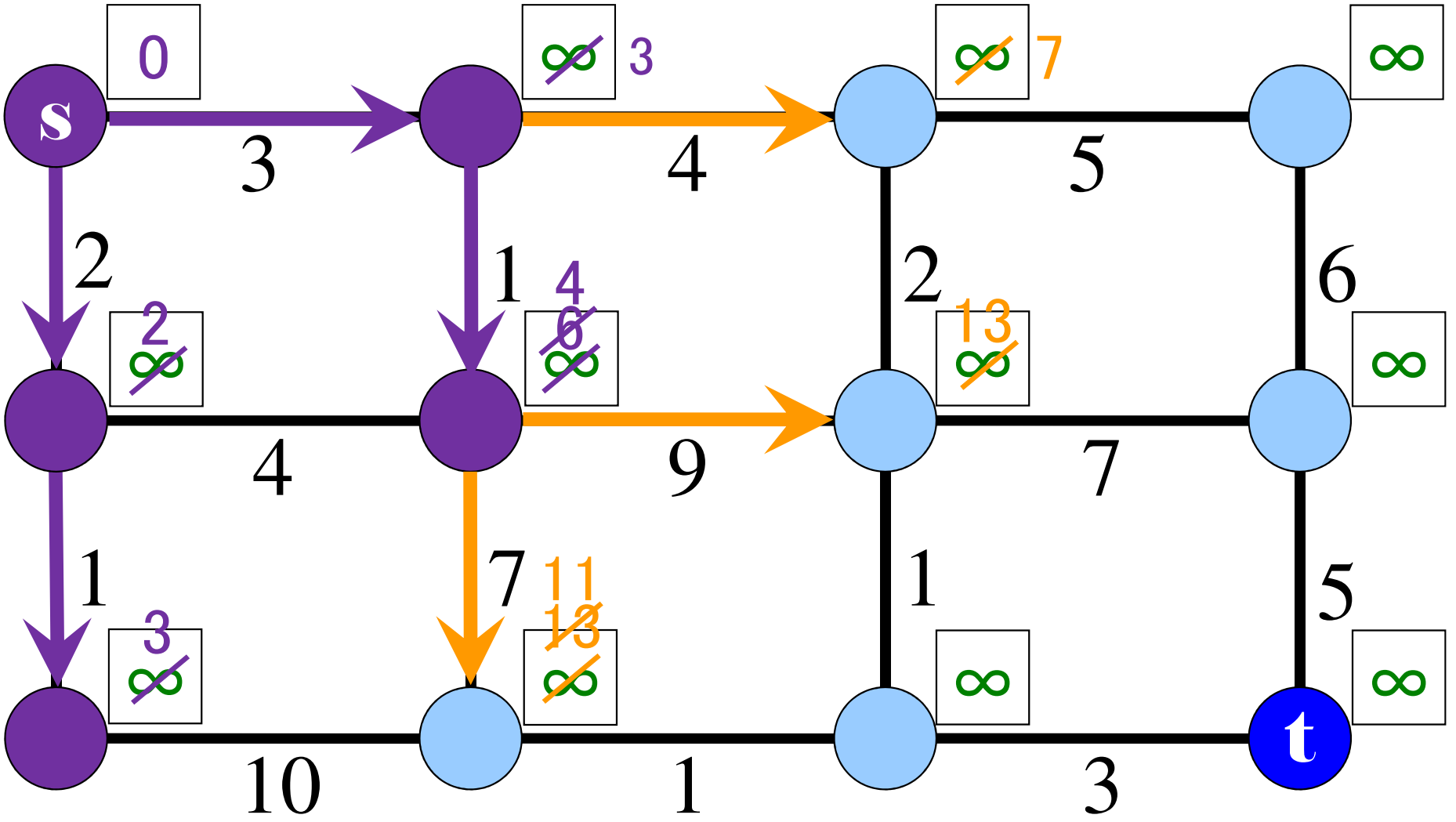


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

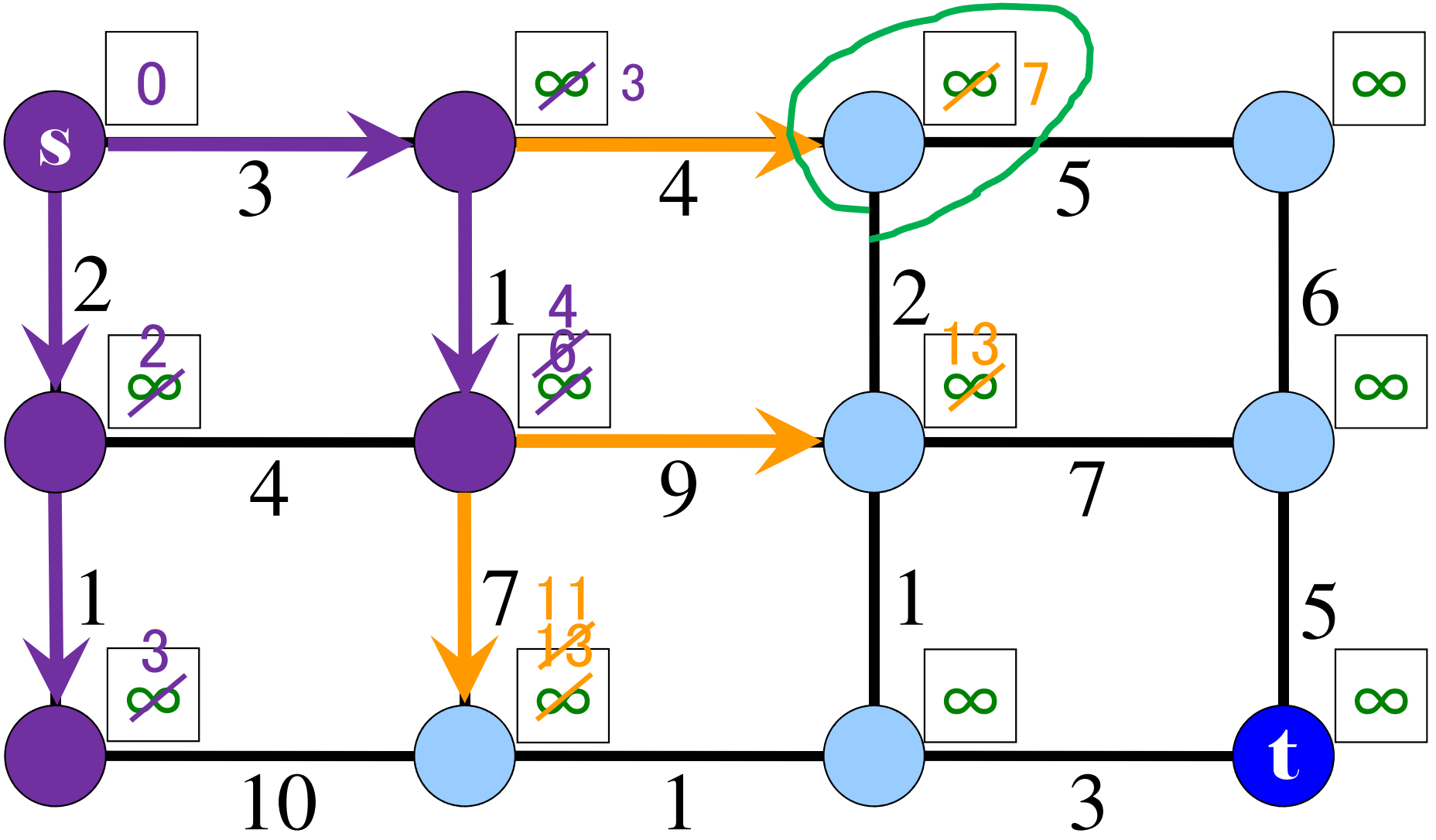
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



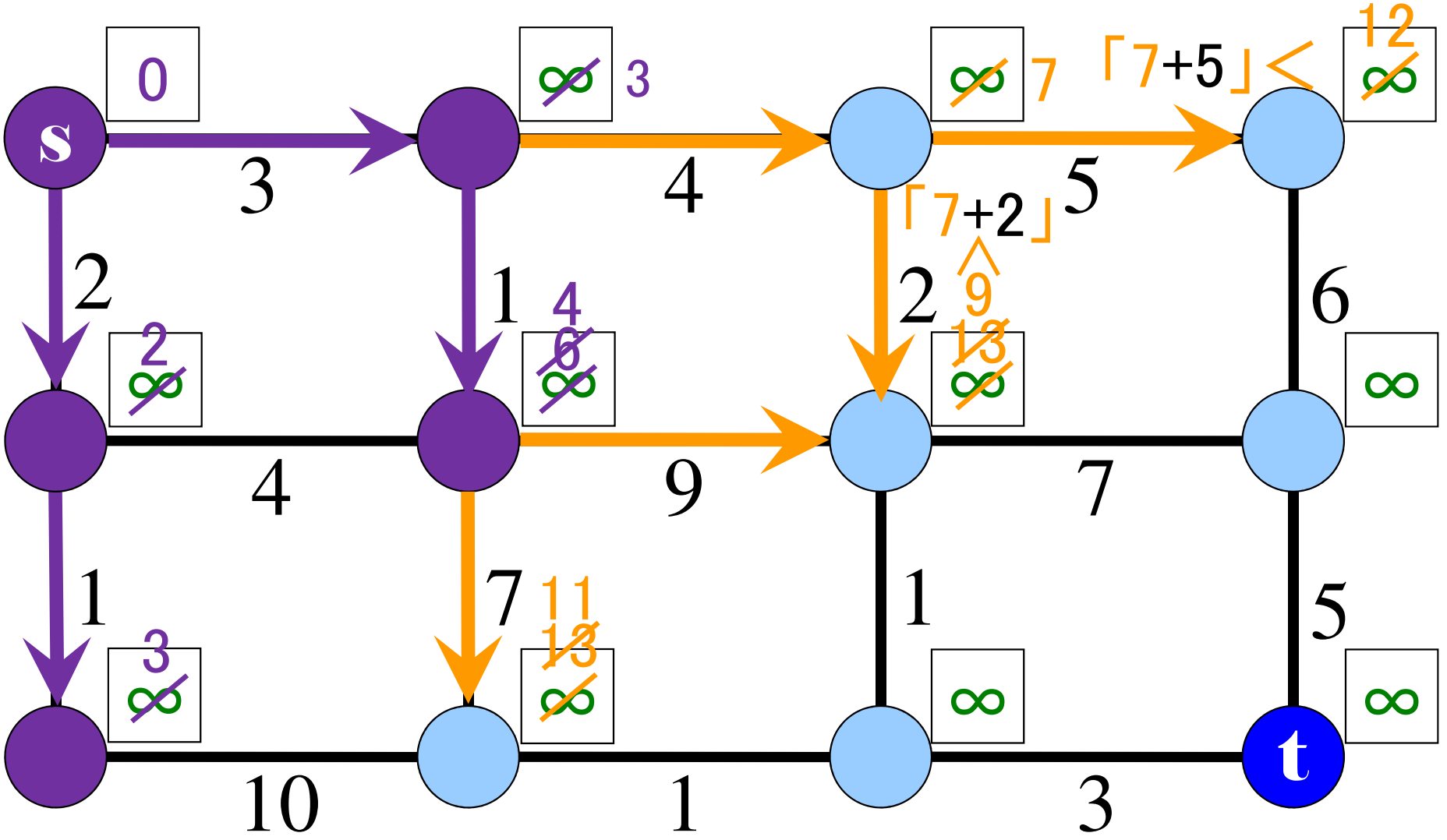
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

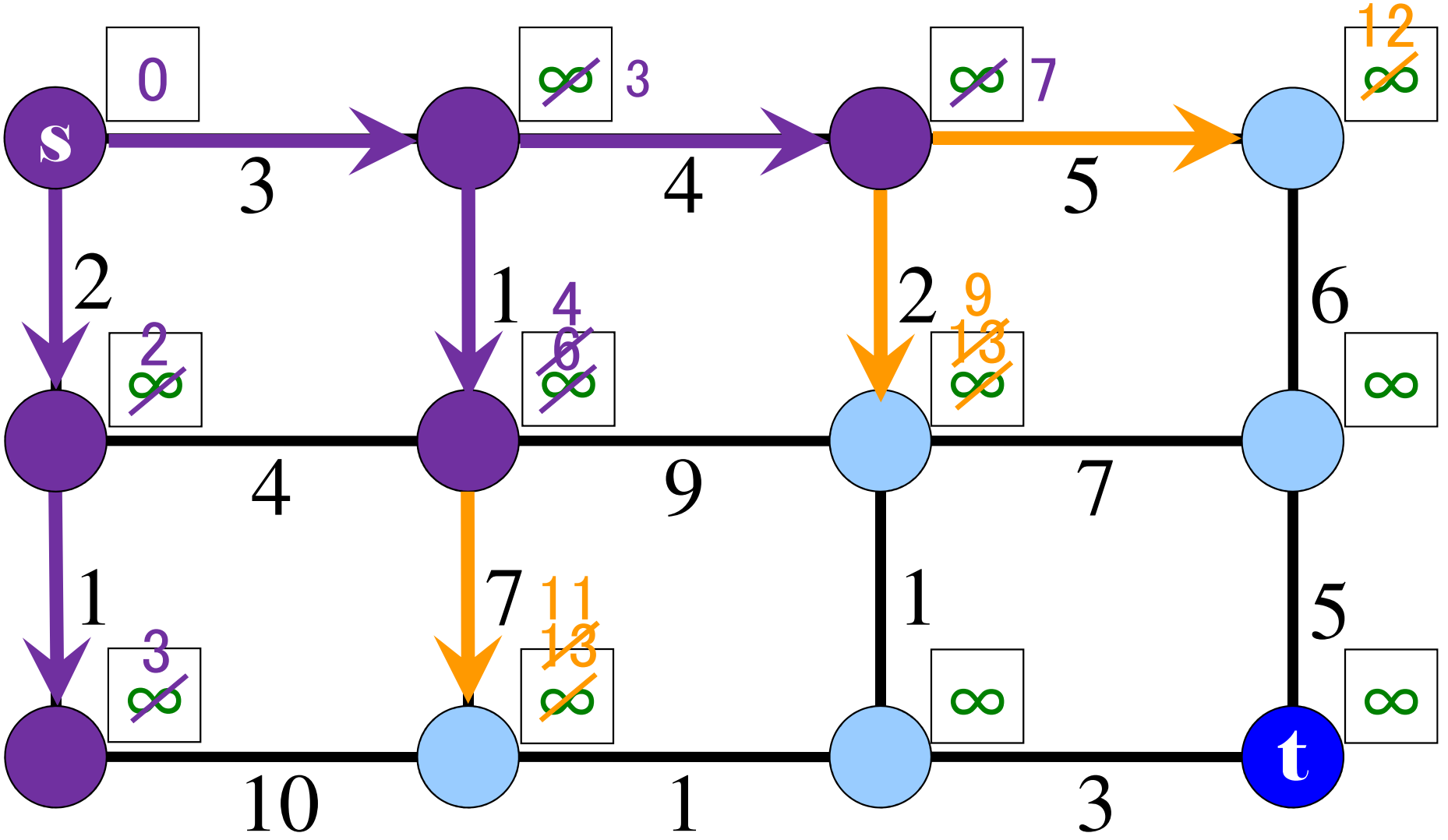


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

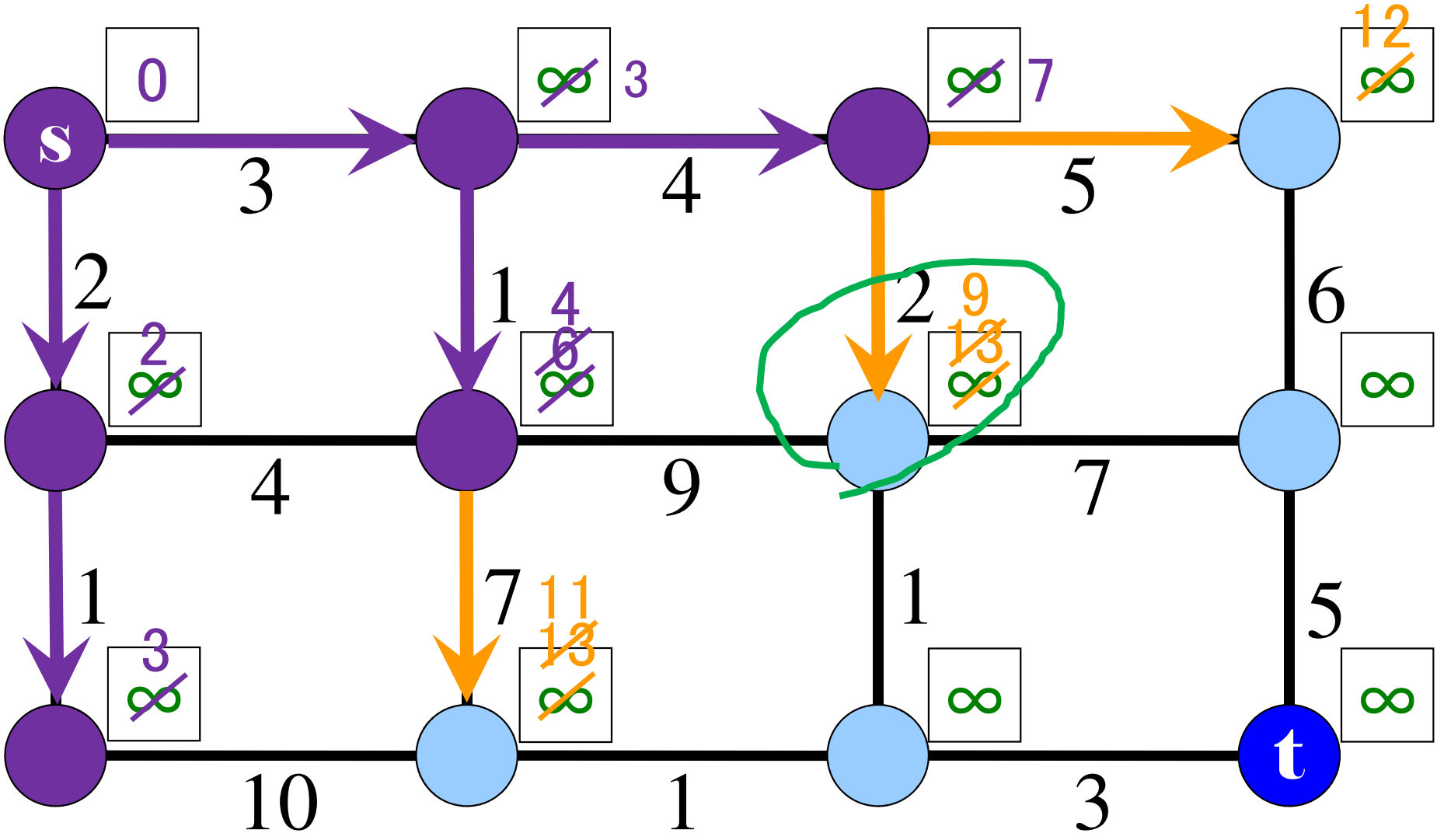
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



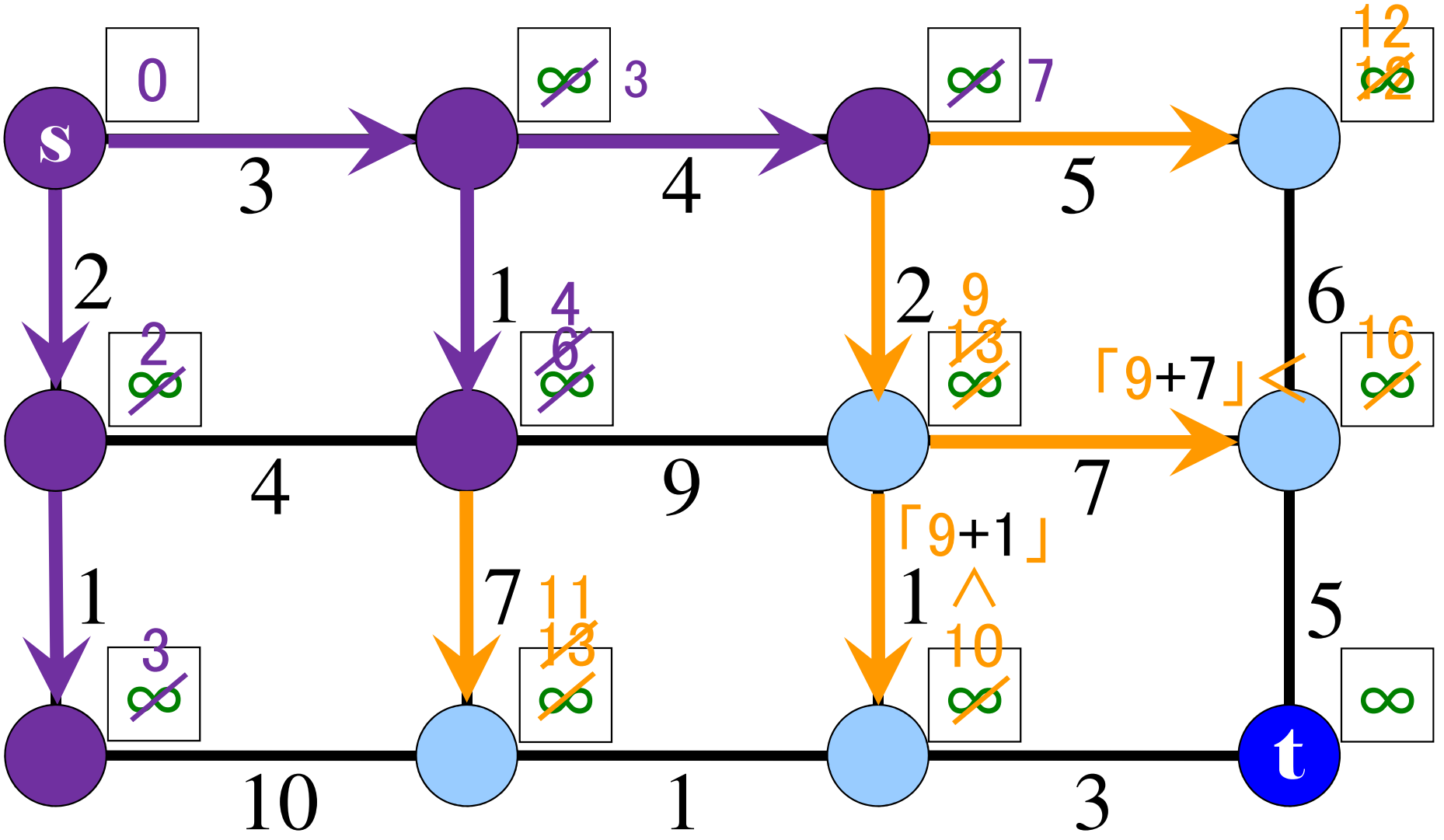
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

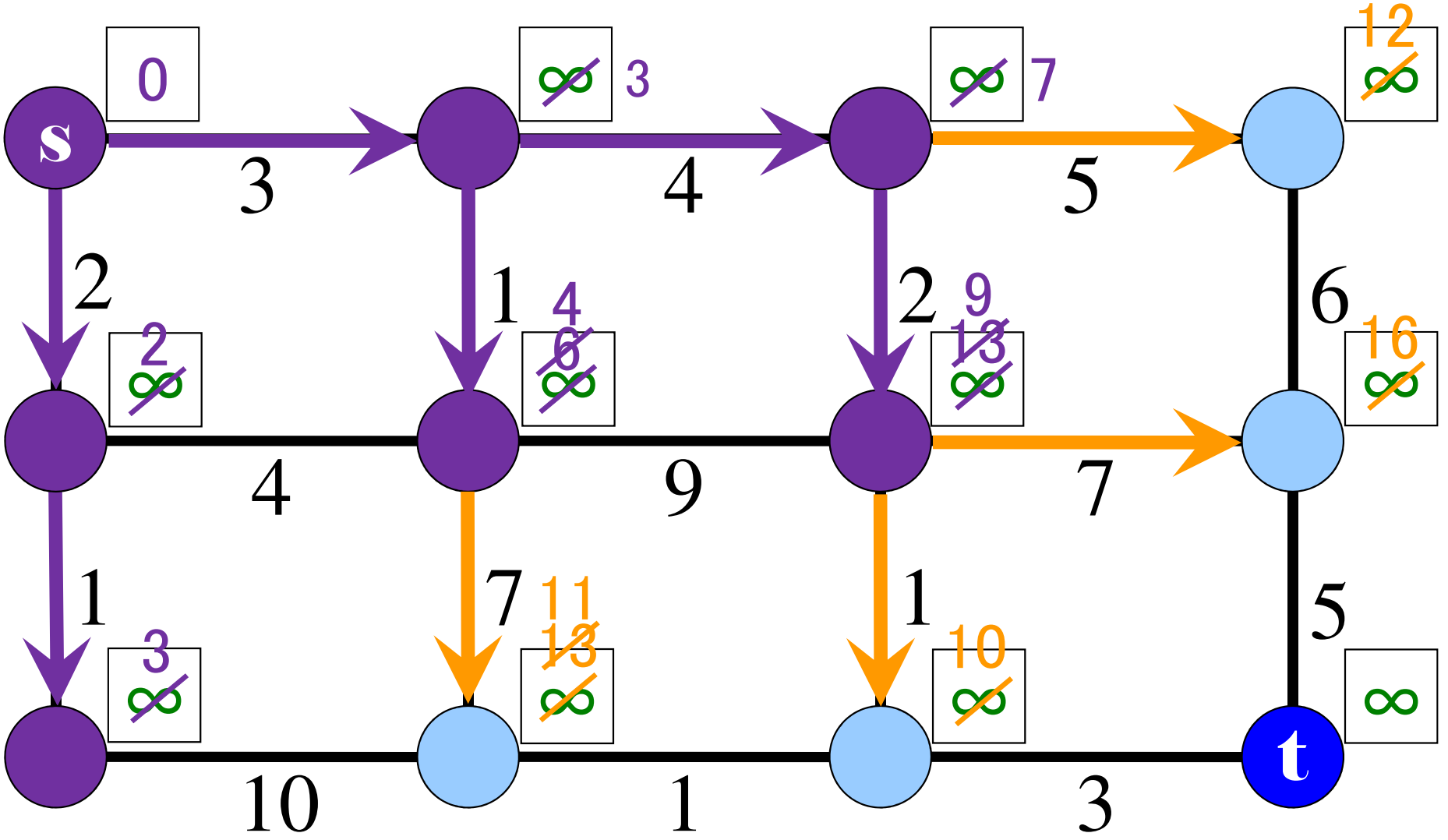


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

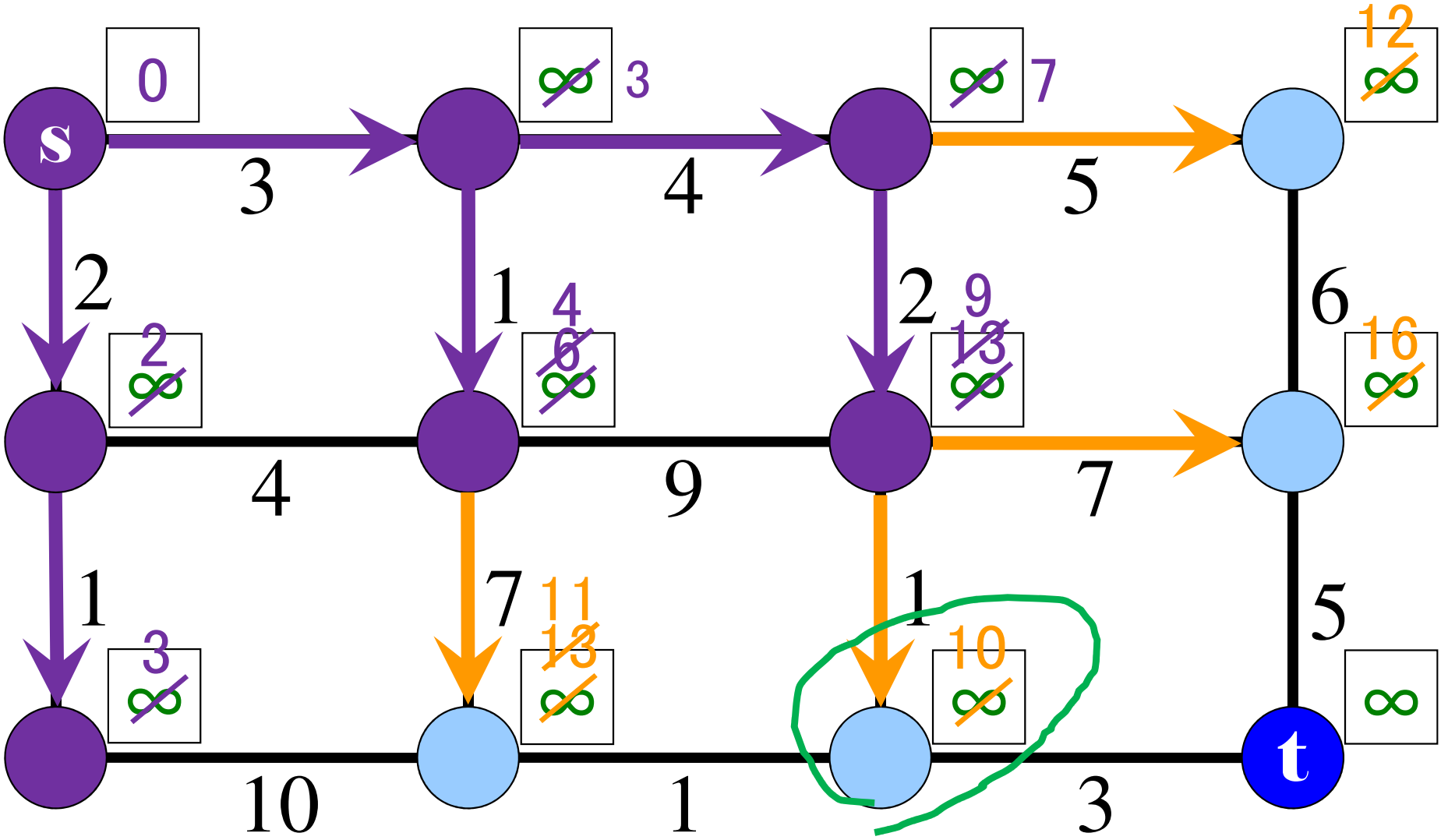
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



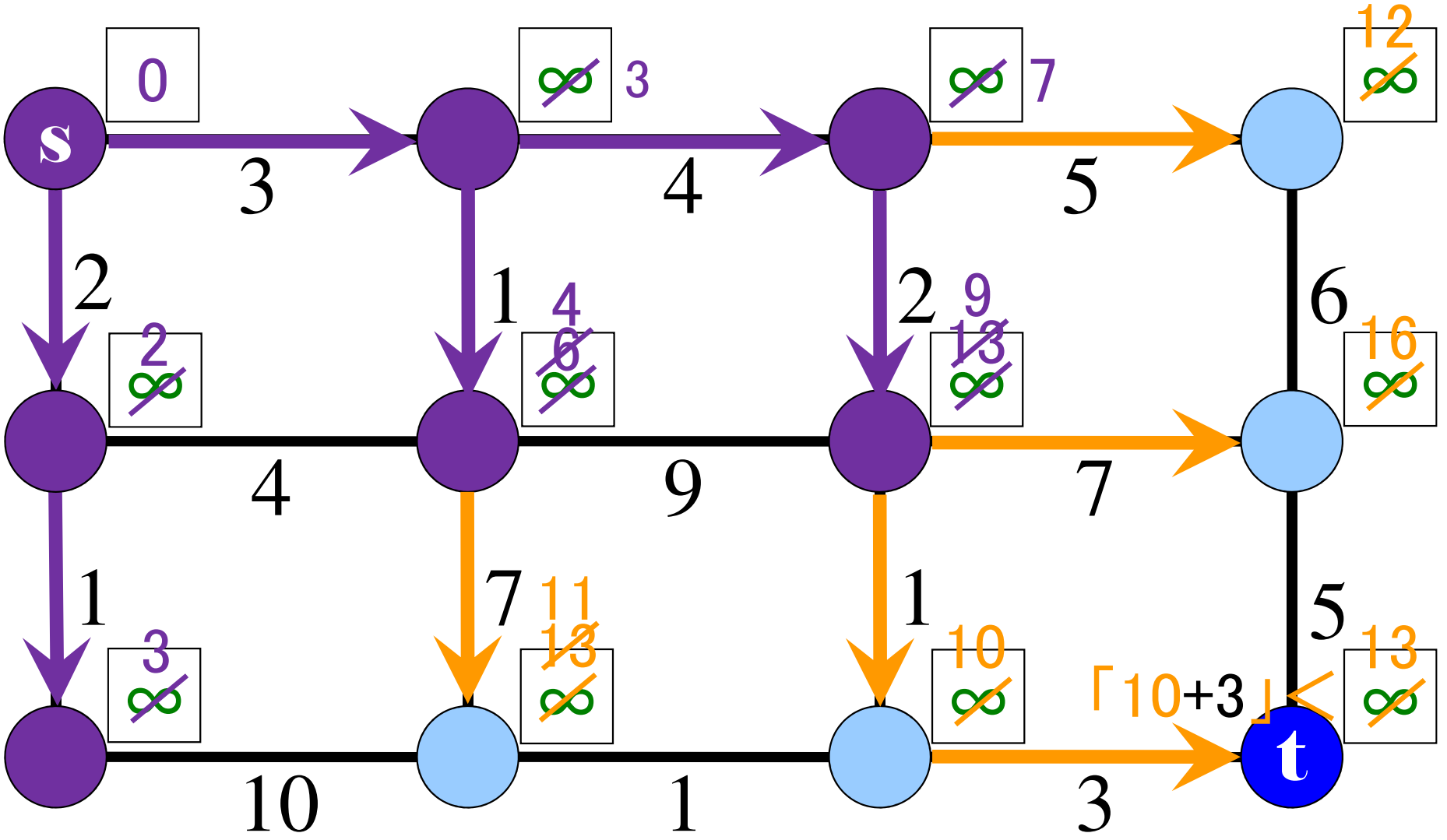
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

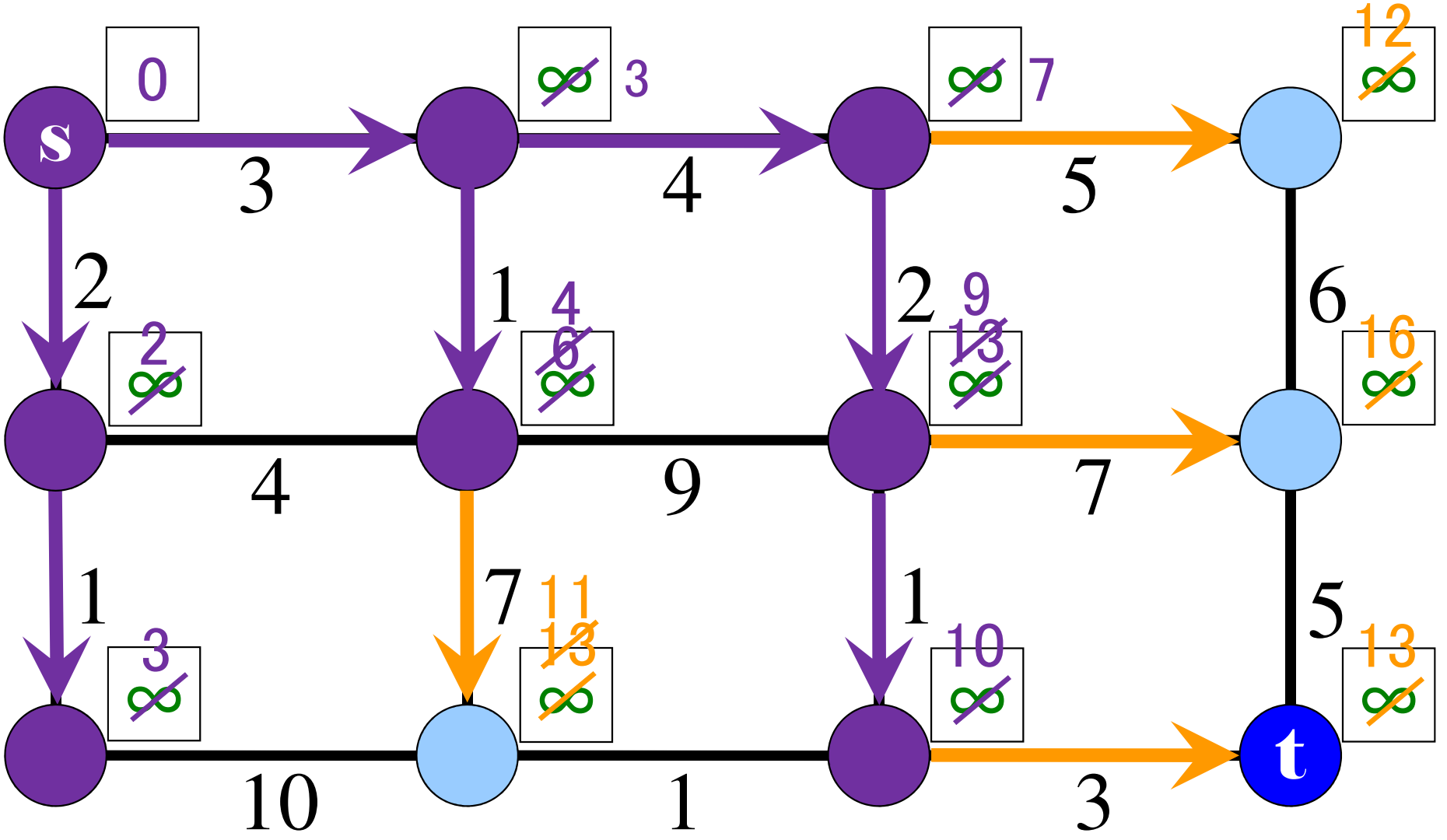


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

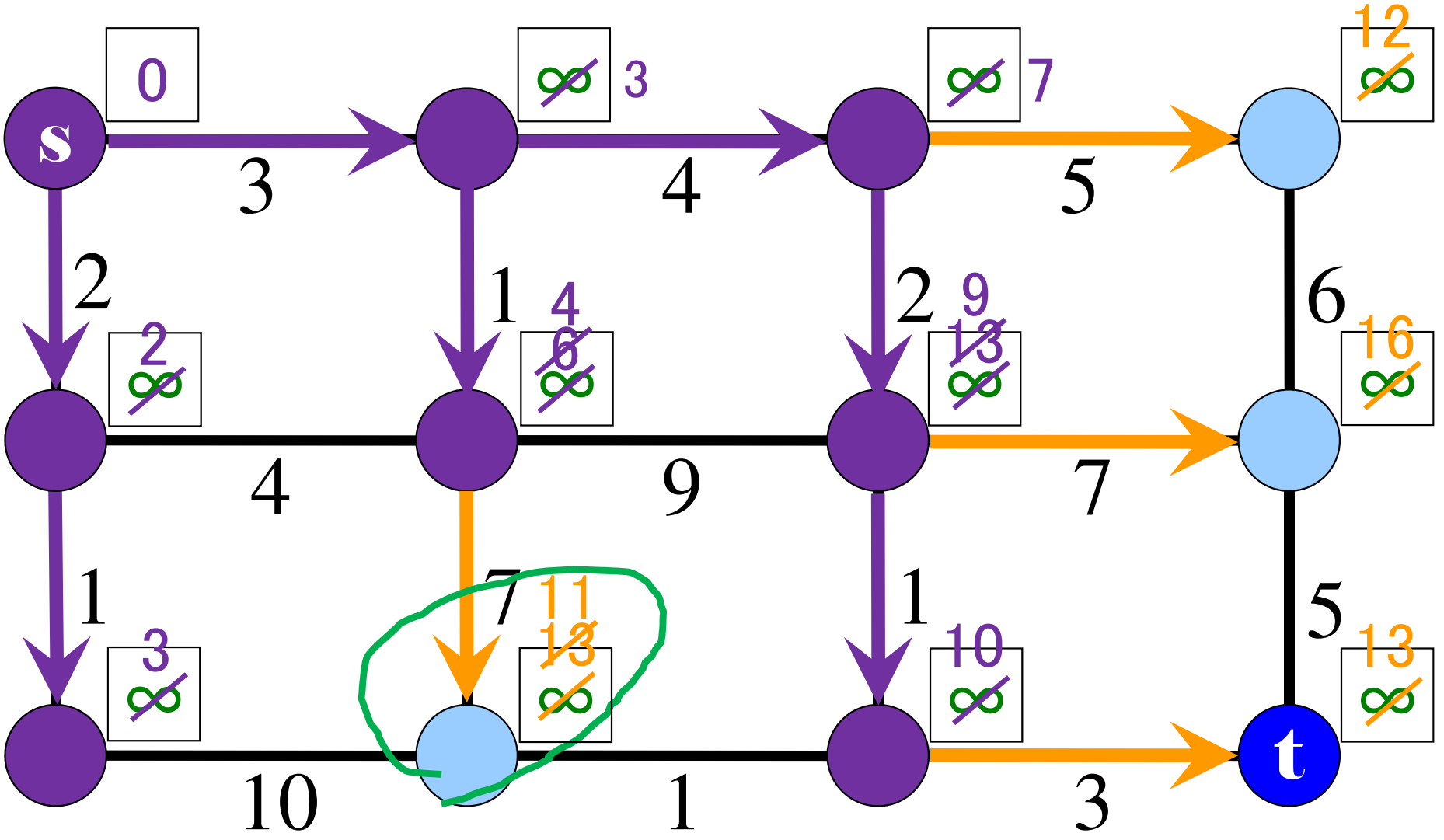
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



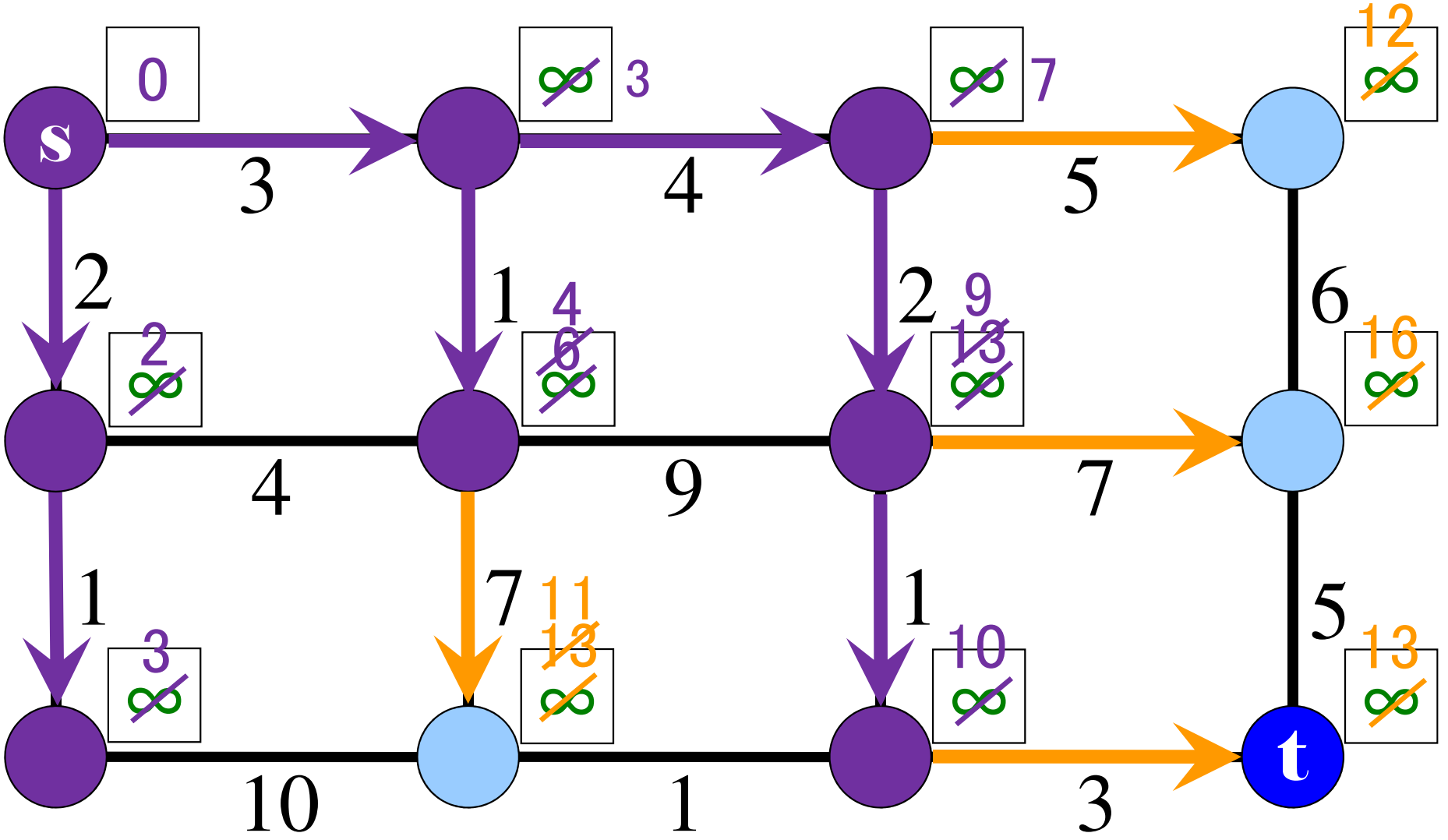
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

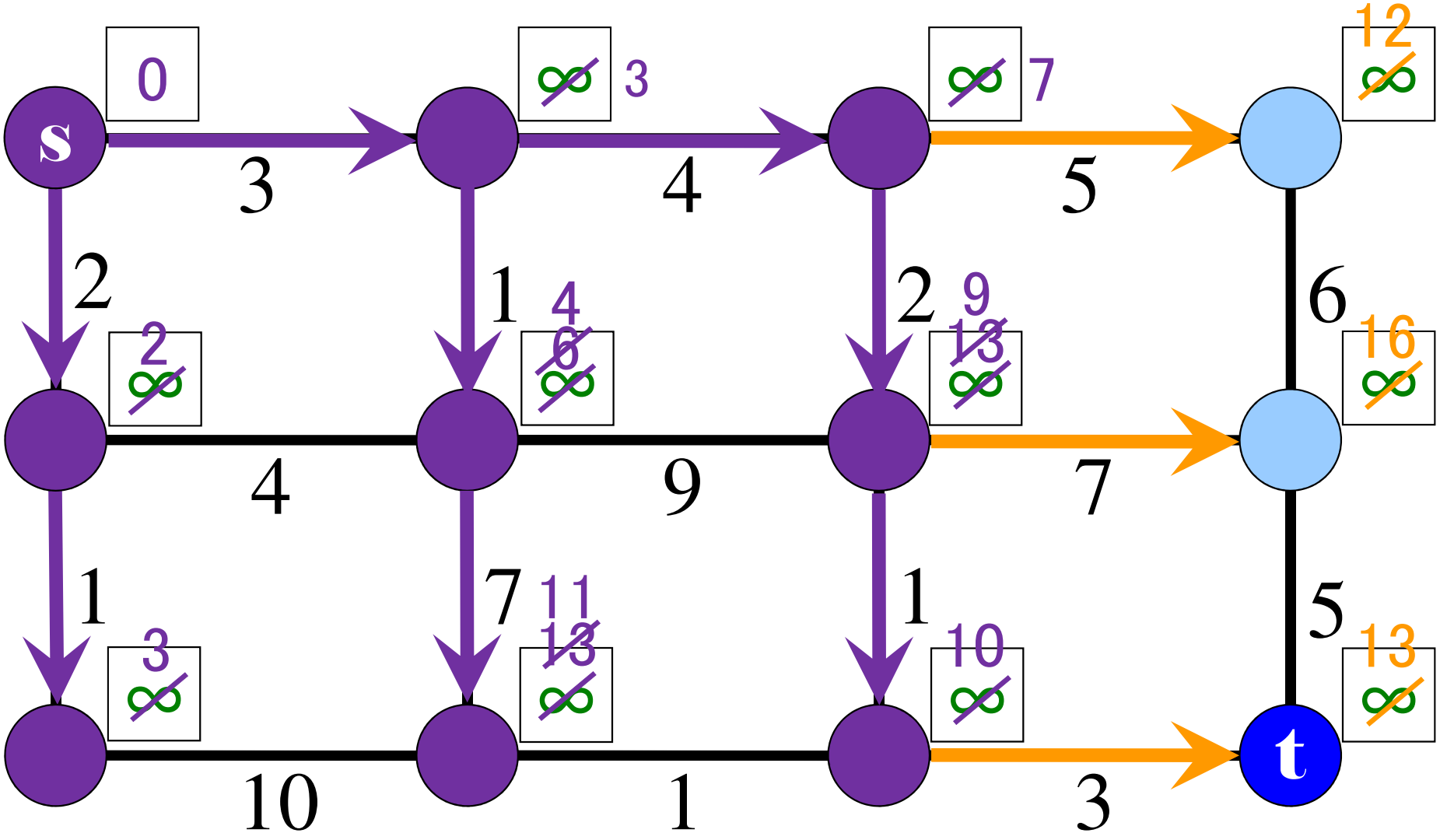


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

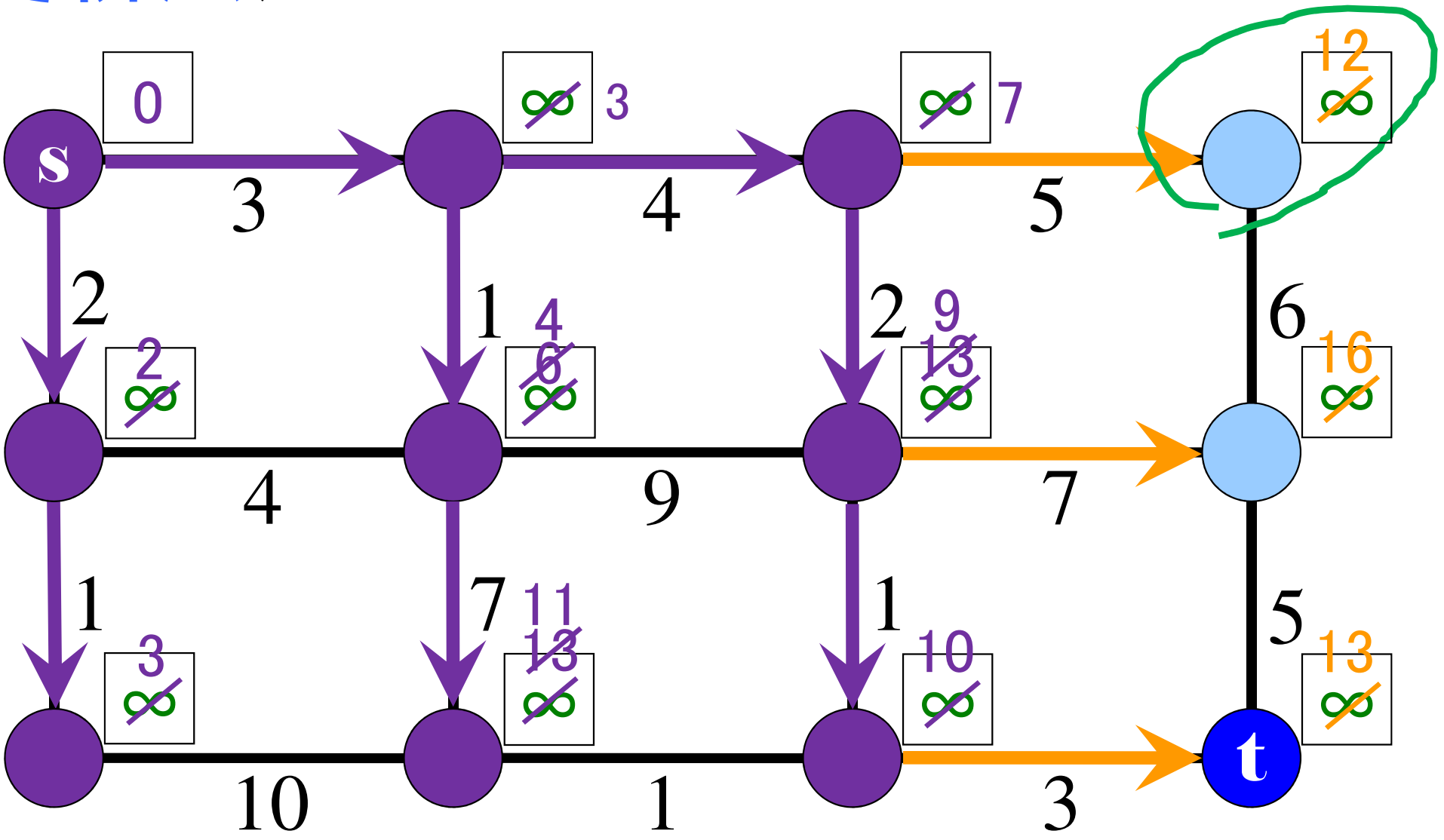
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



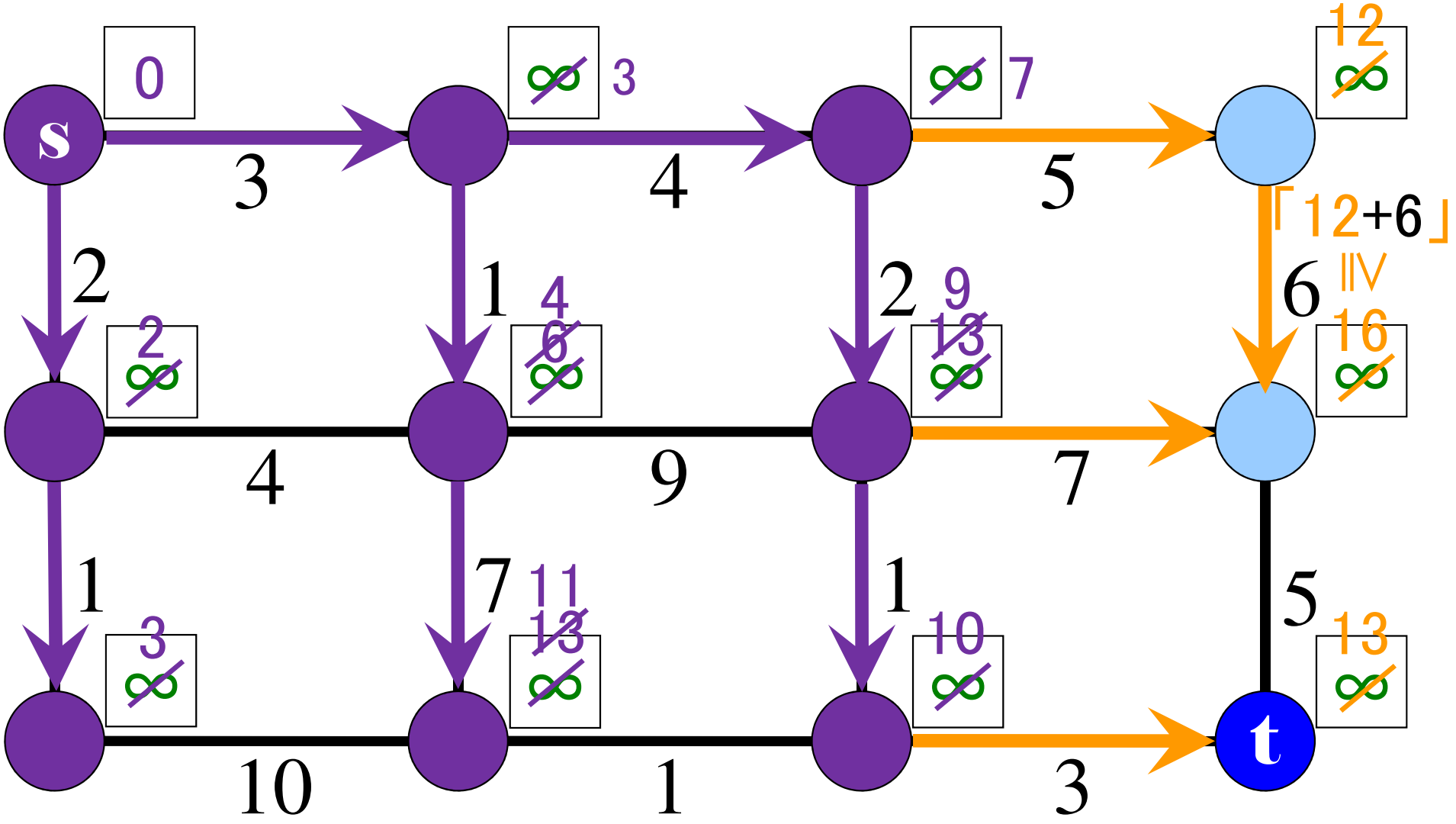
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

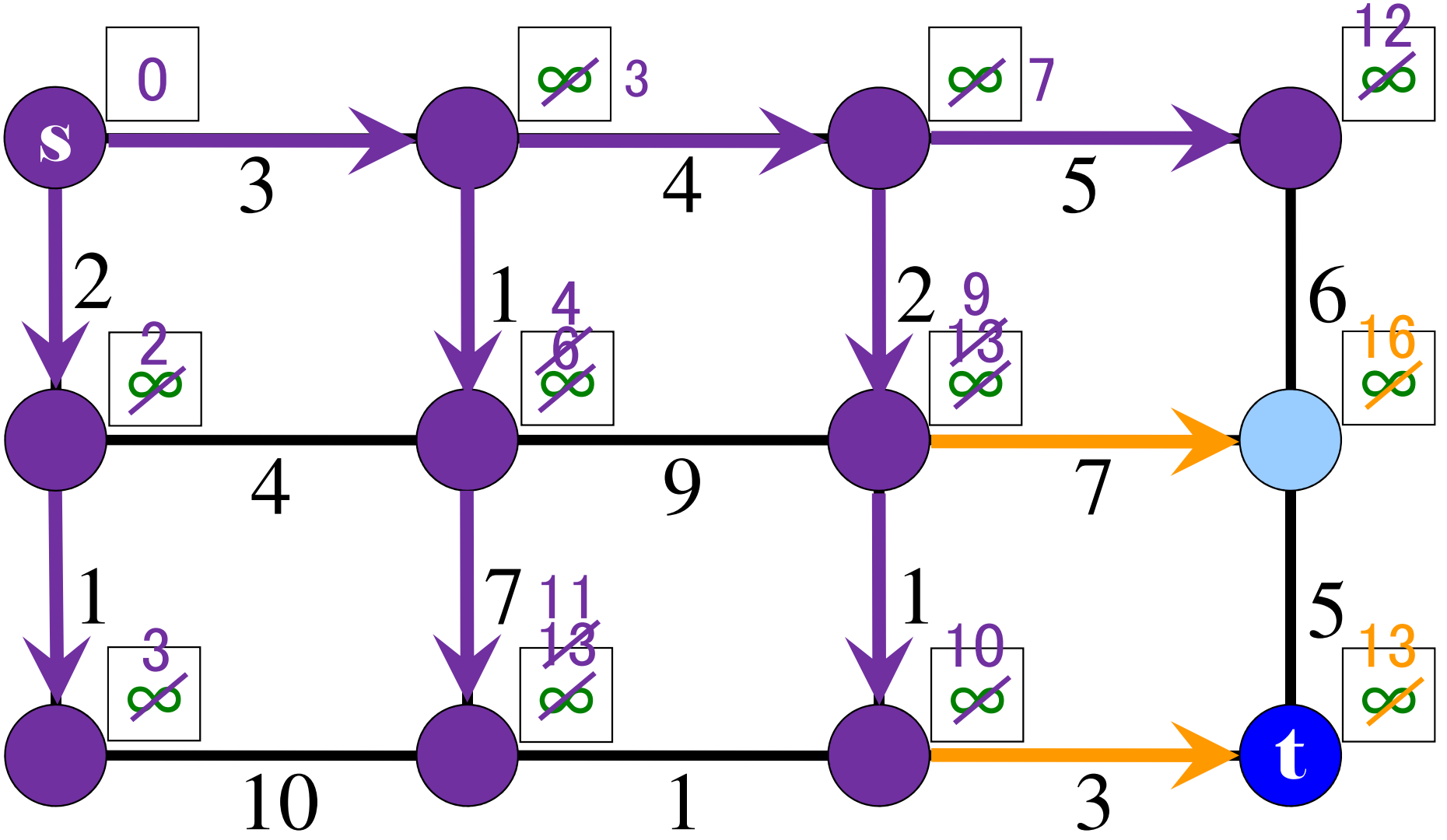


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

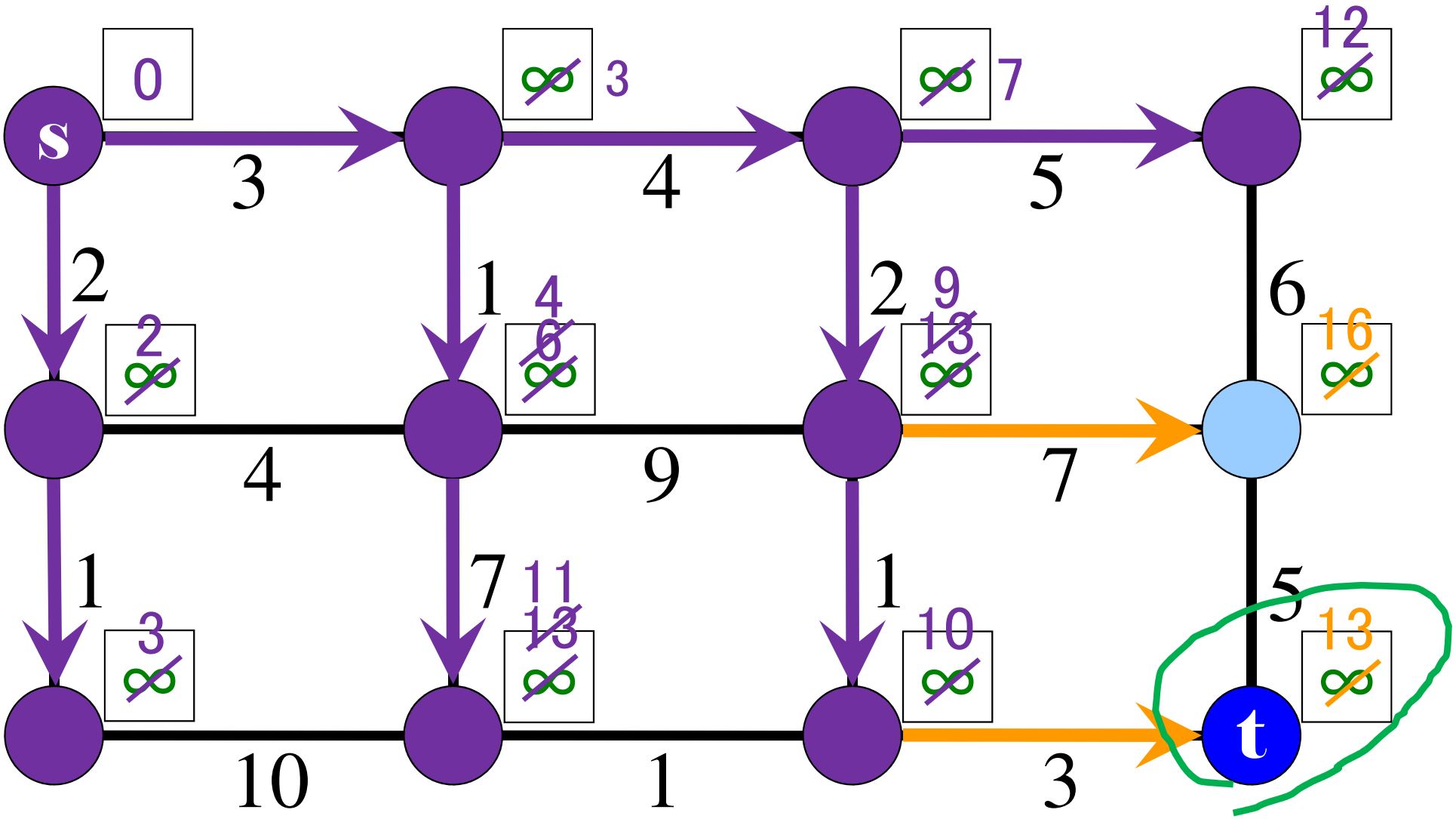
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



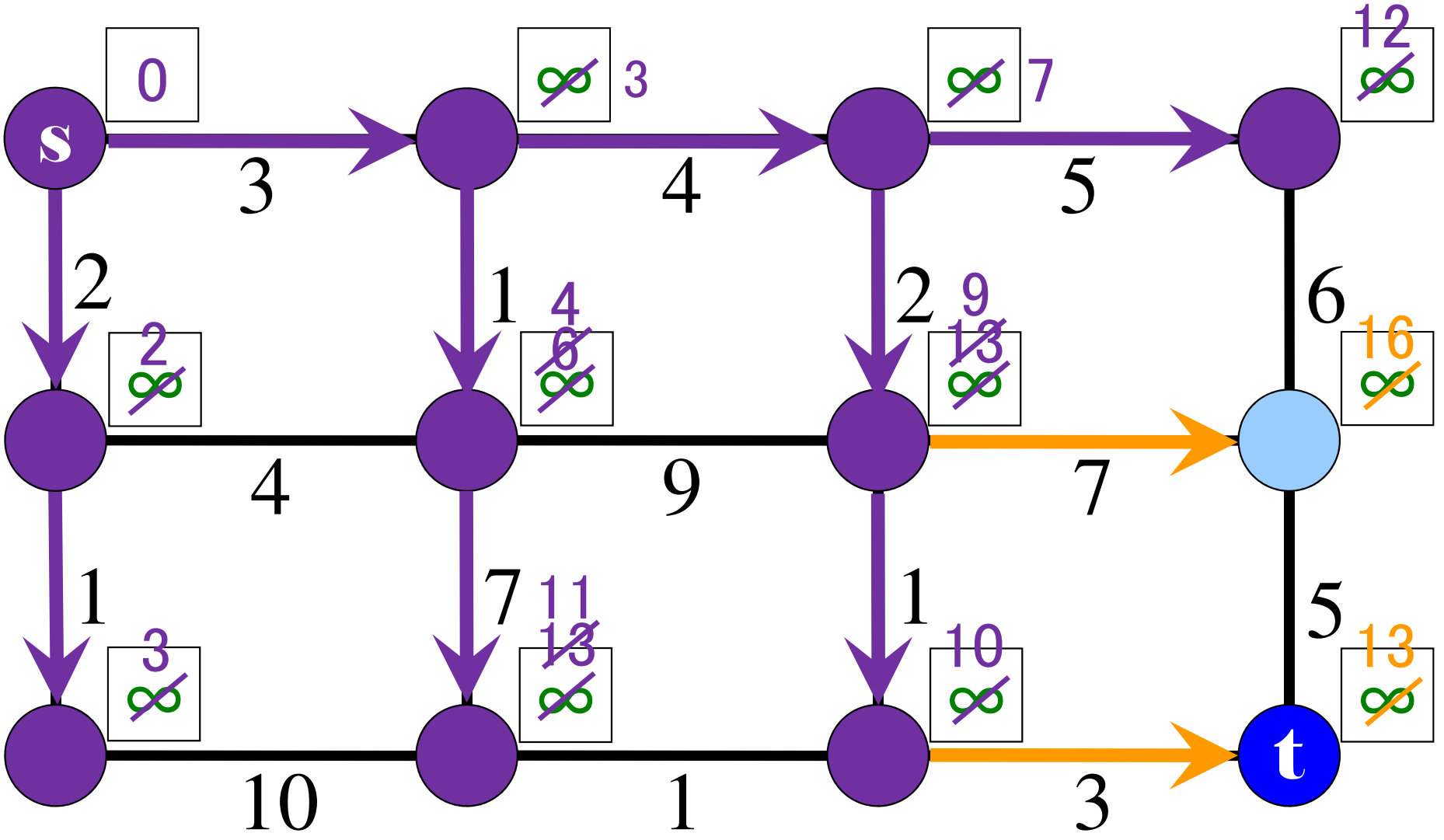
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

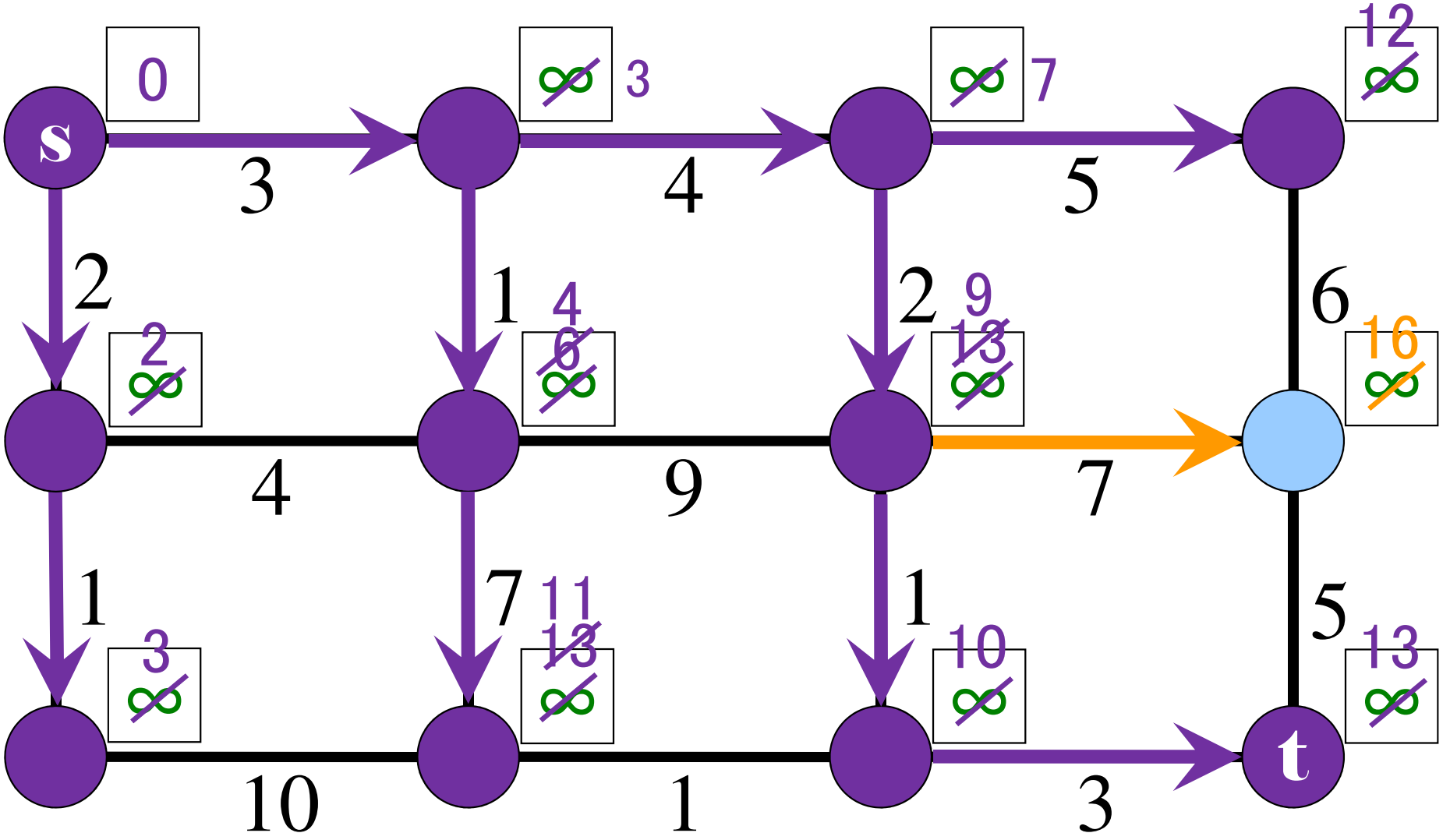


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

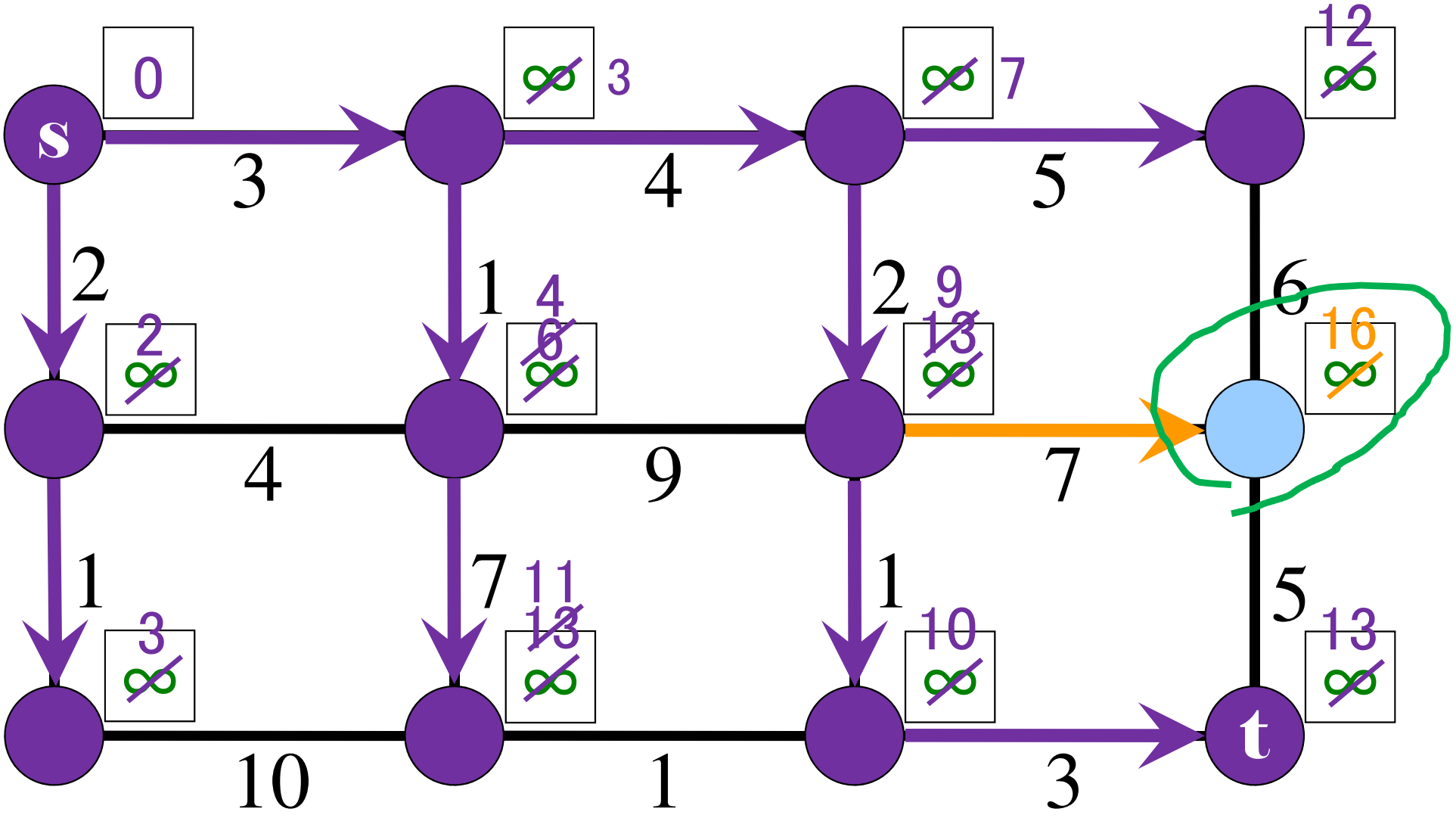
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



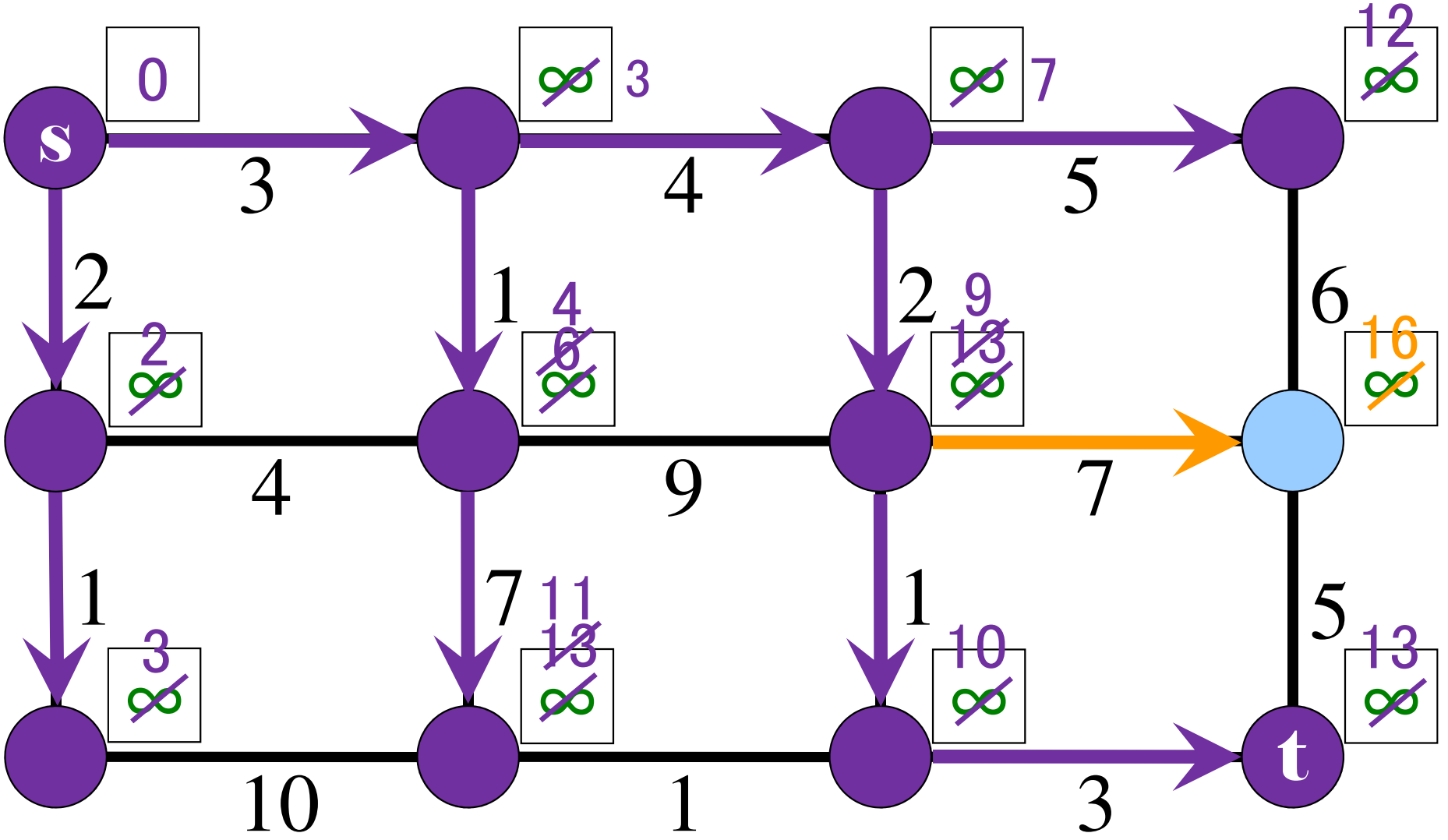
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

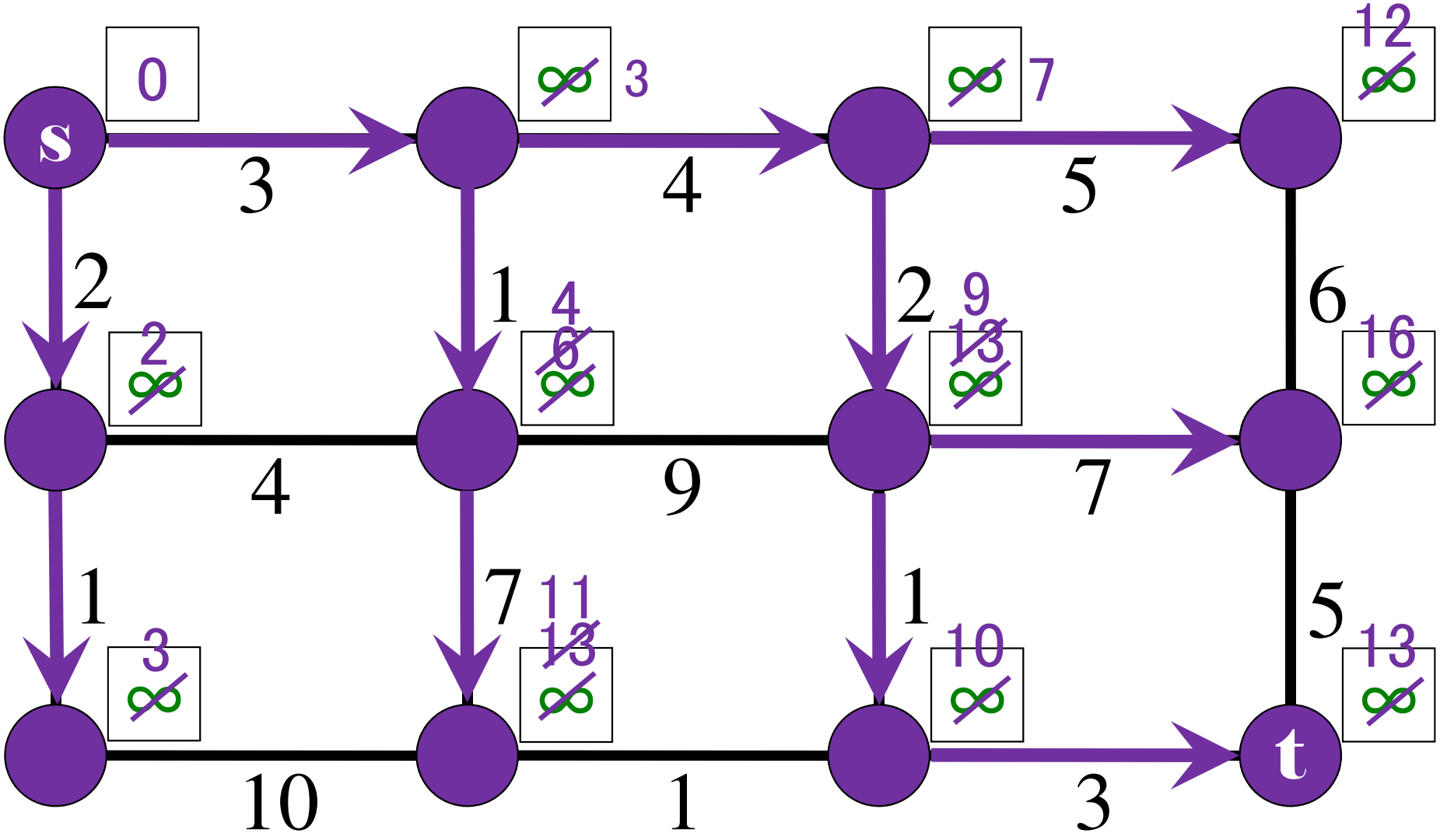


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

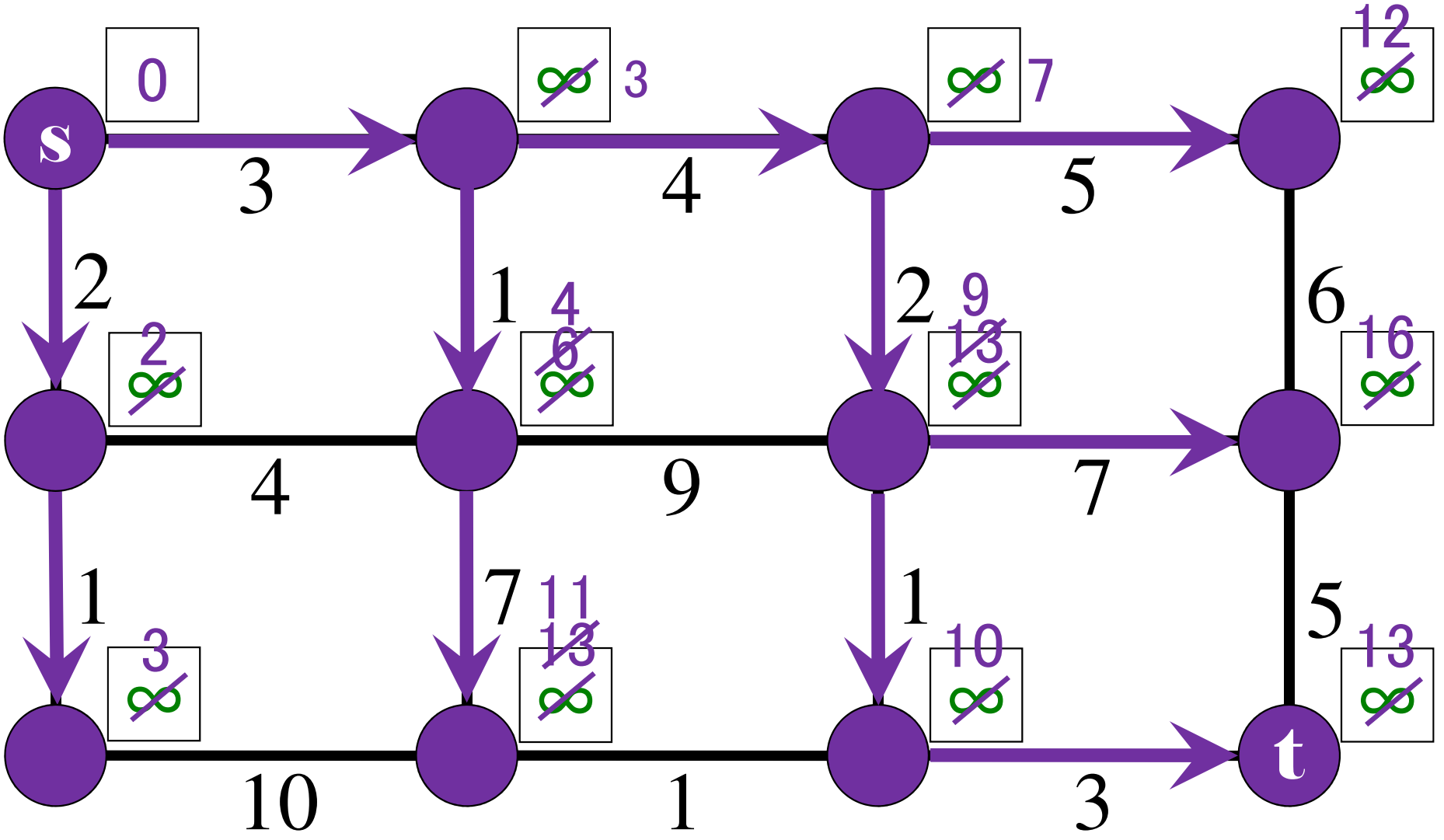
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



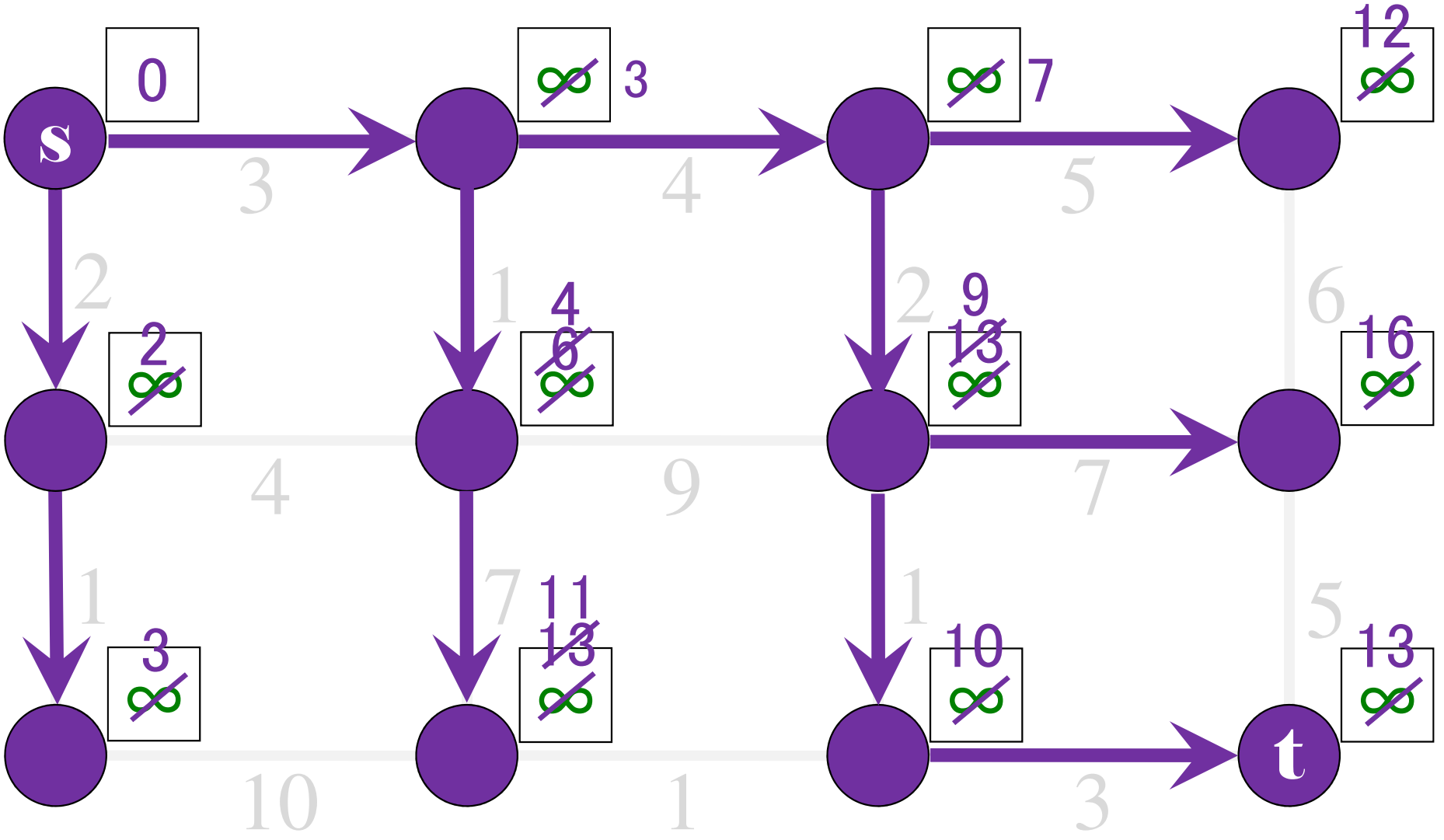
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



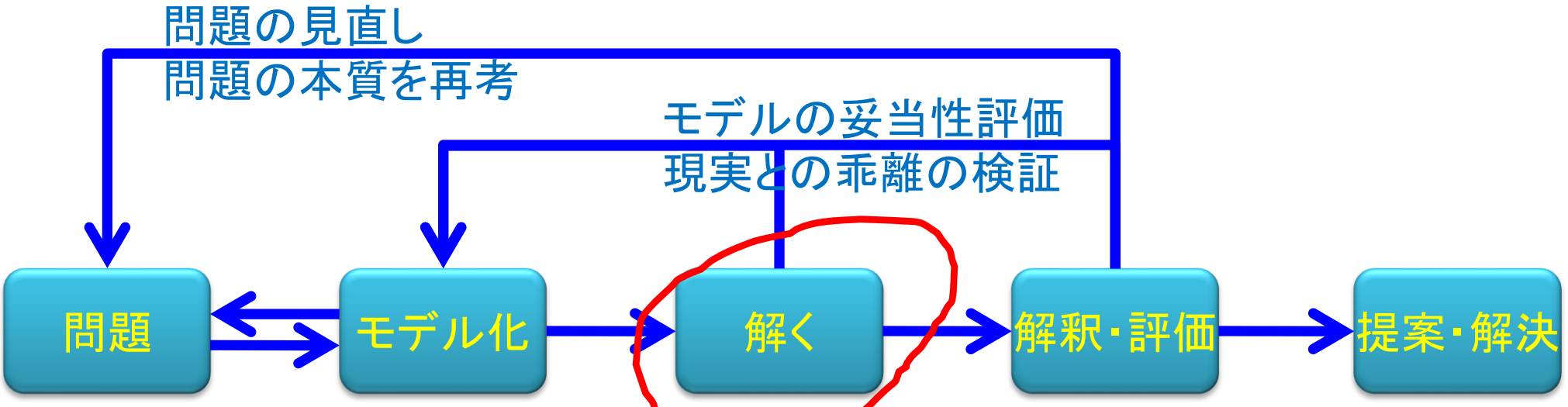
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

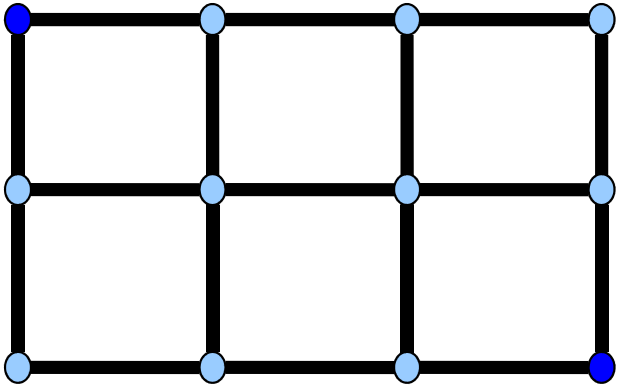
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

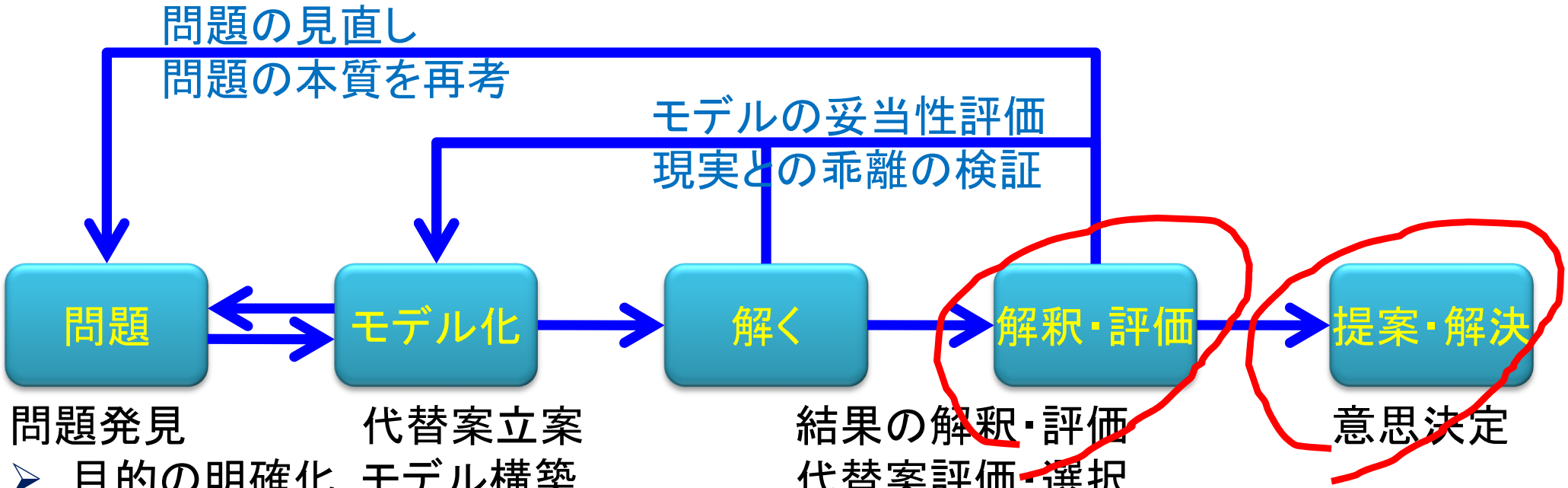
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

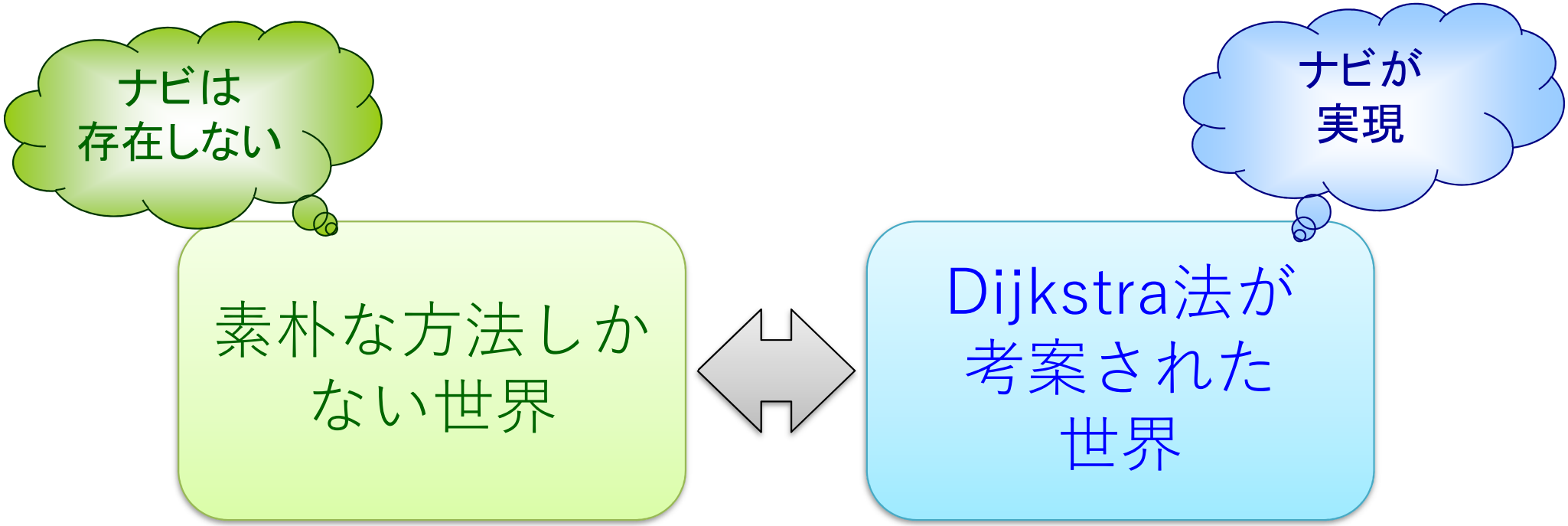
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

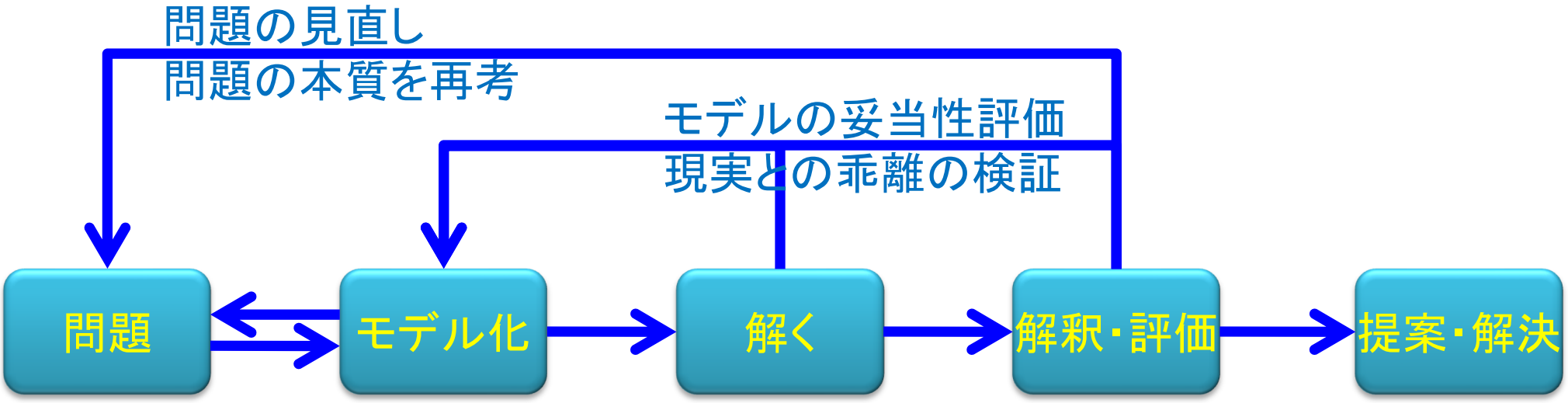
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



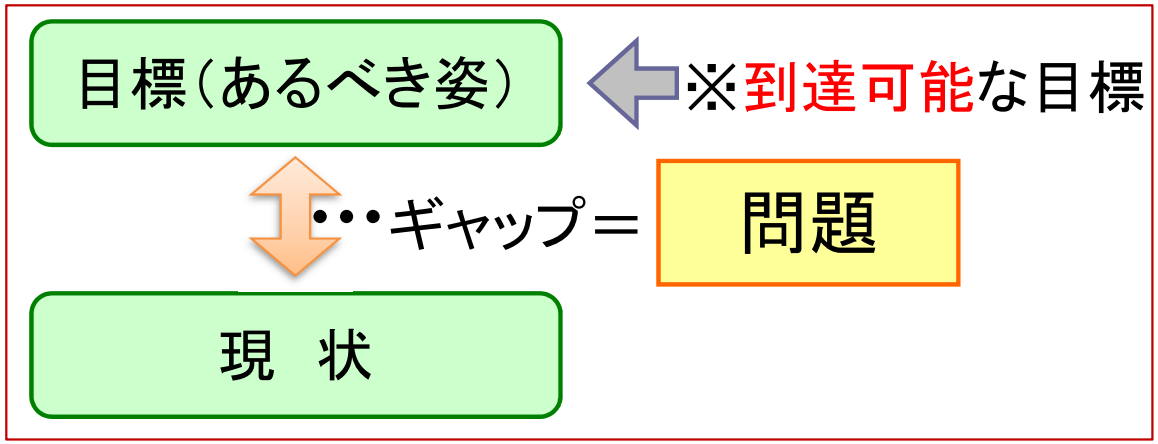
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

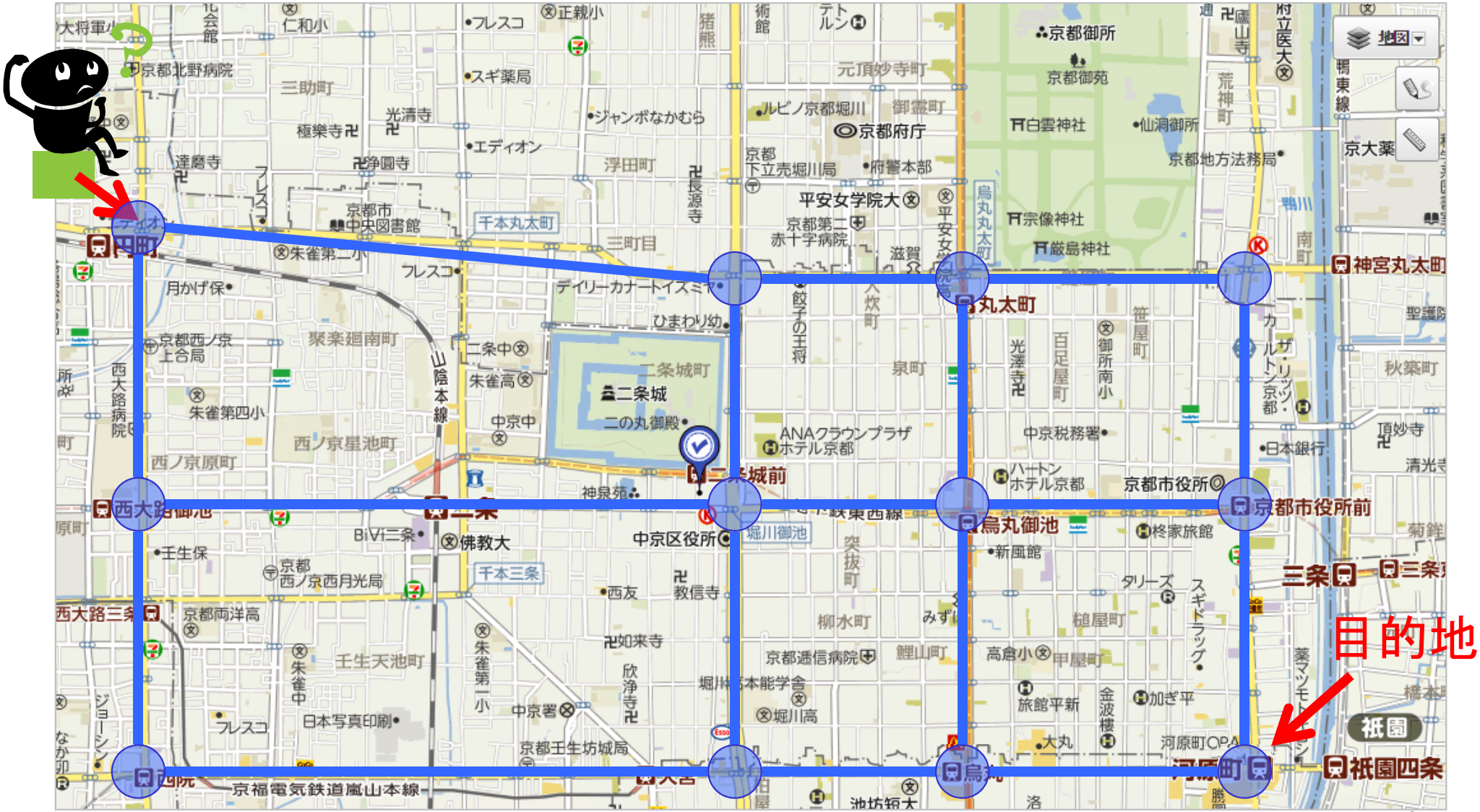
意思決定

問題の定義



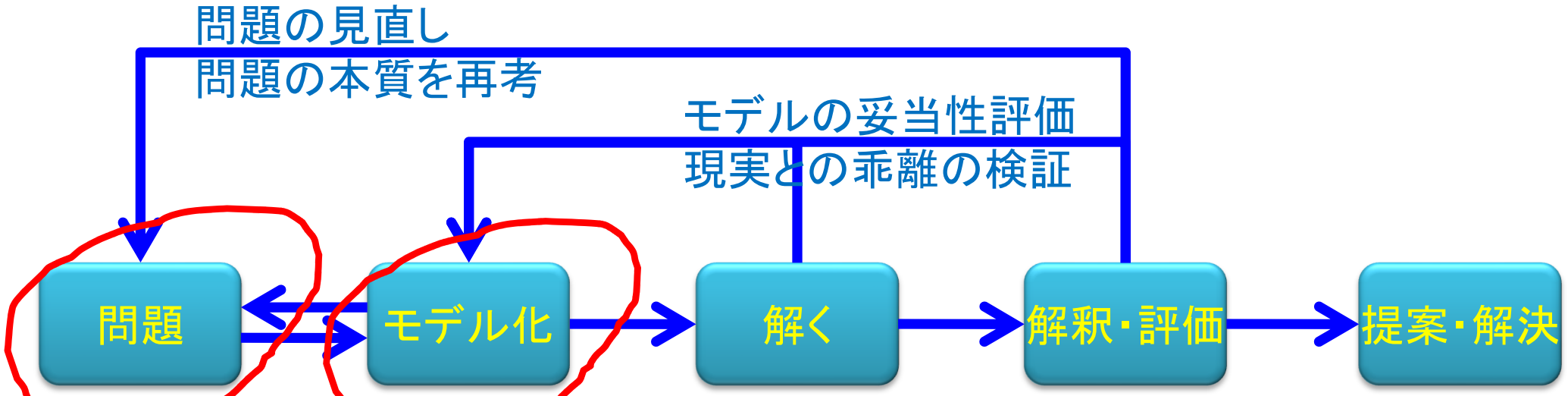
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

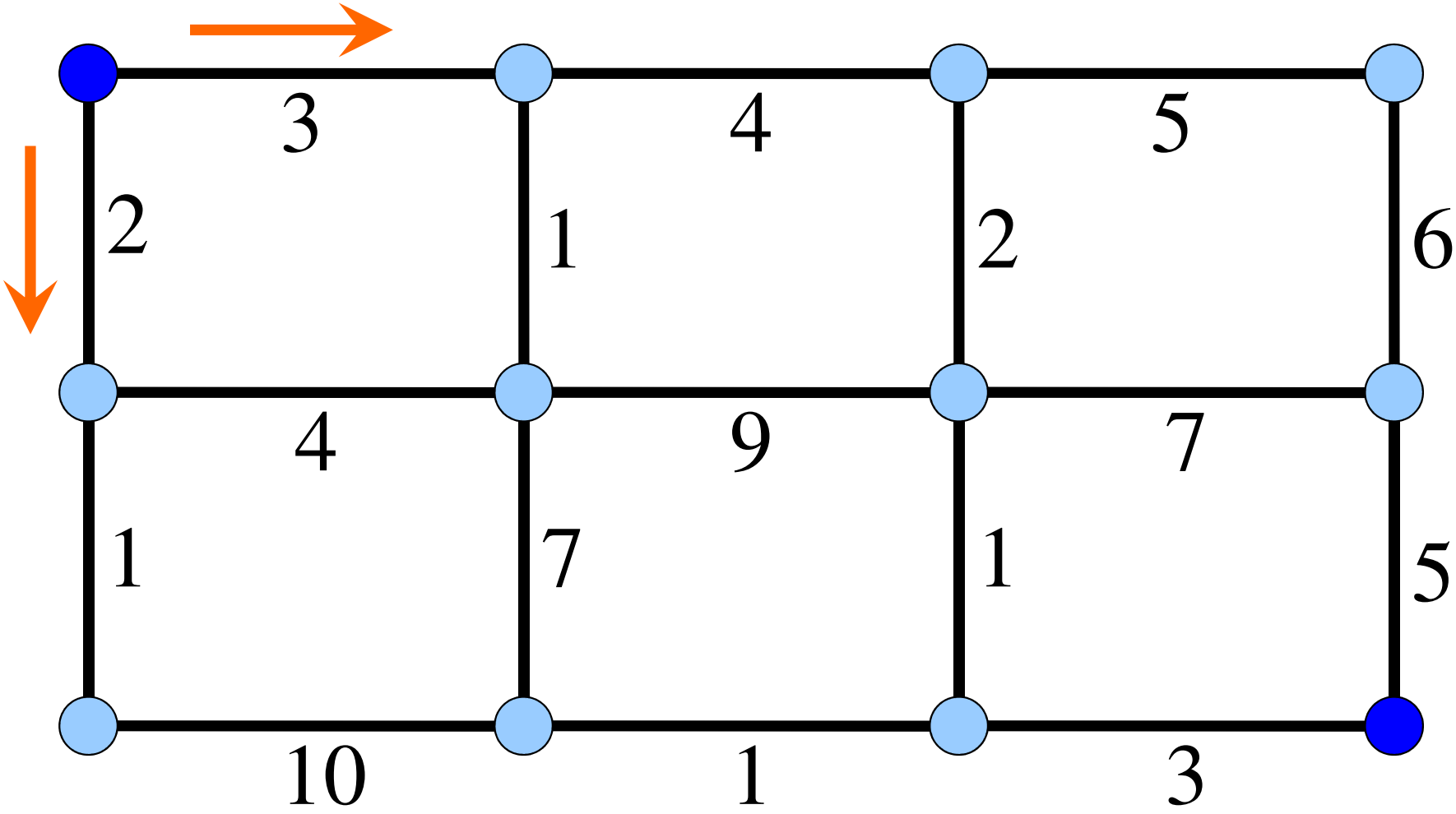
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

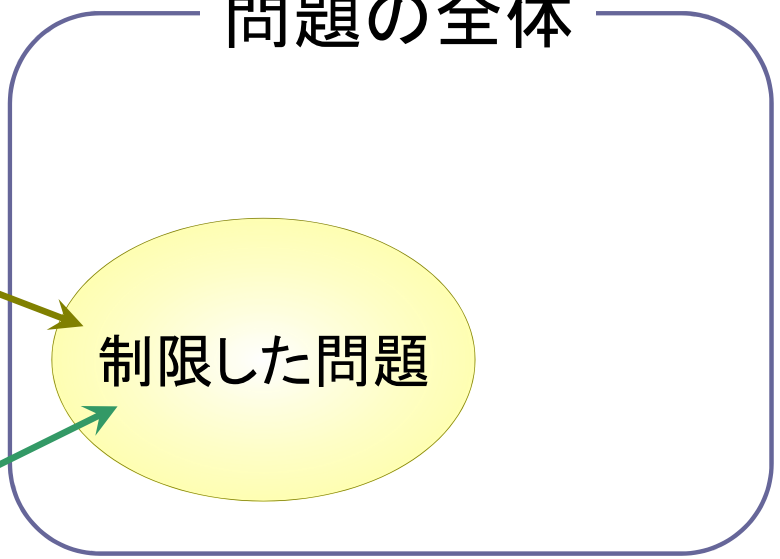


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

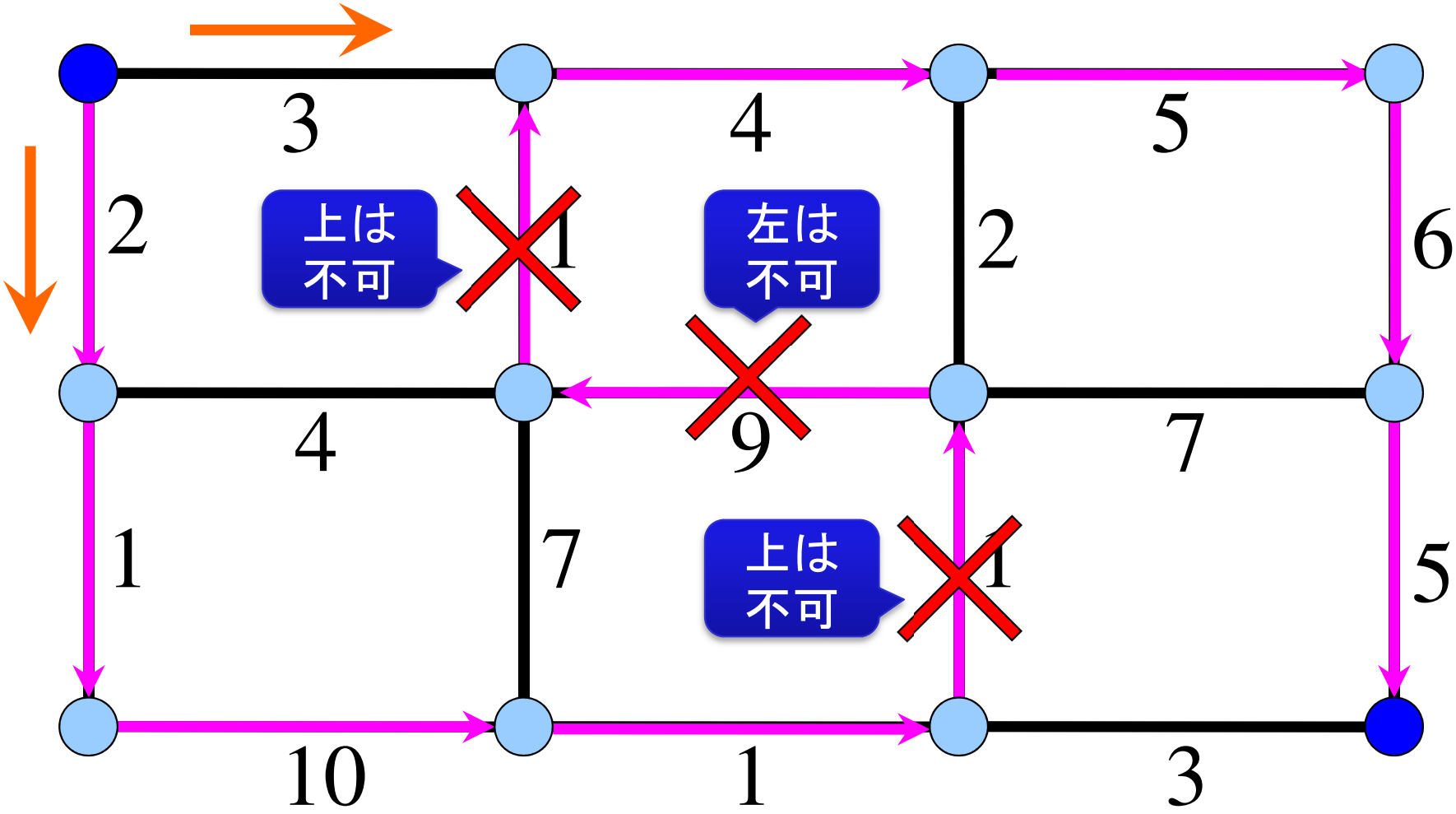
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

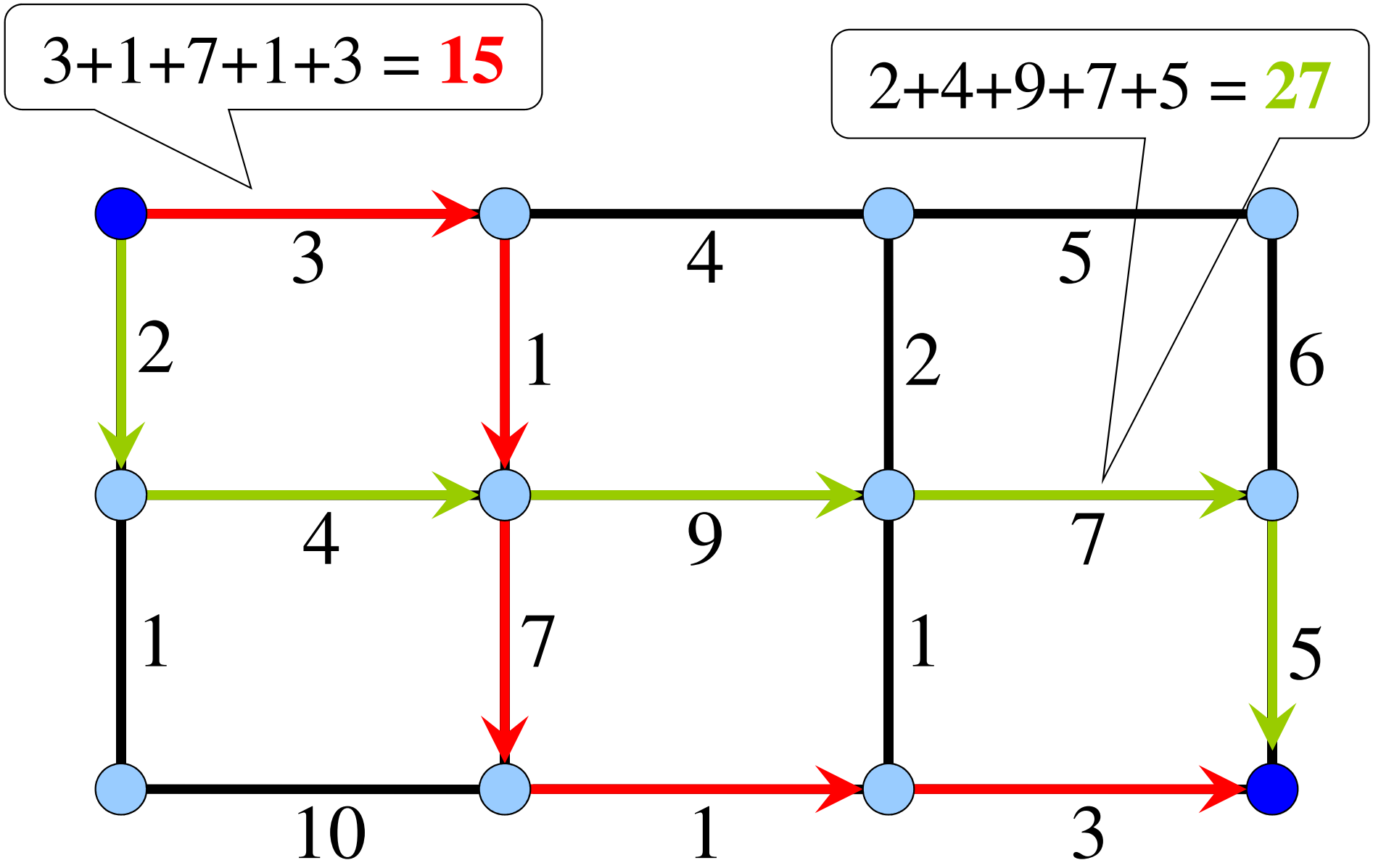
難しいなら易しくすればいいのさ！

制限した問題

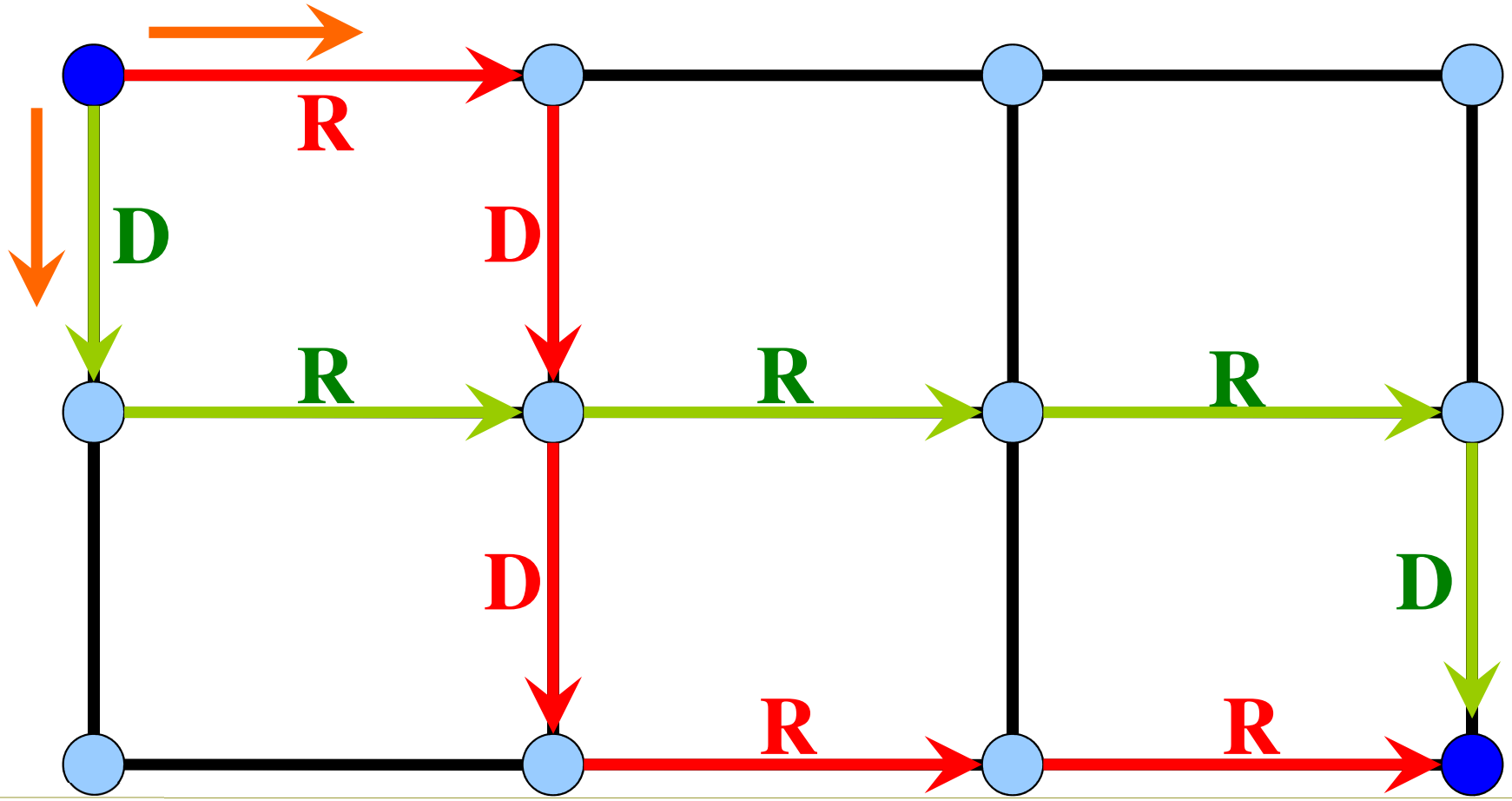
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



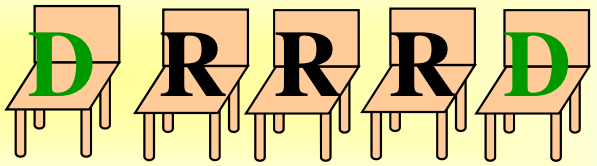
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



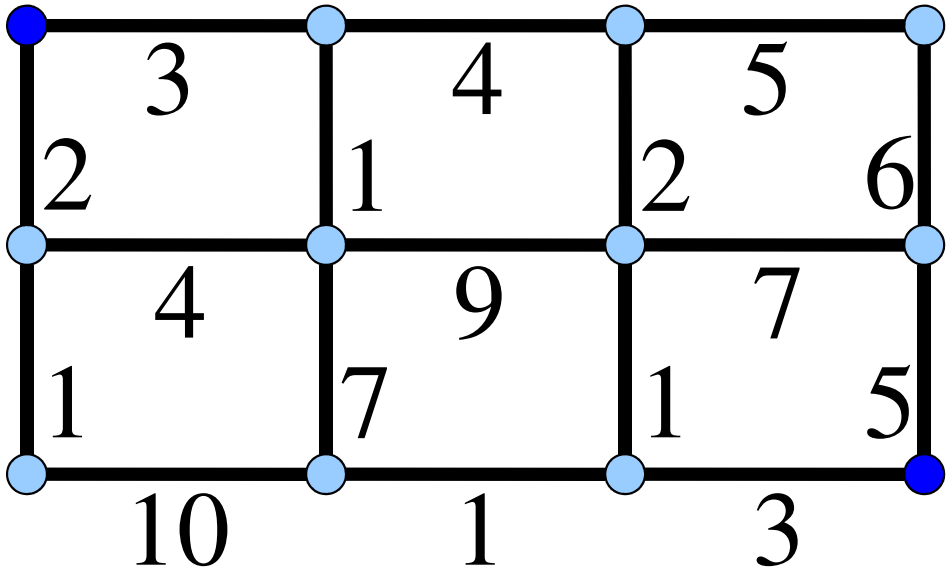
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

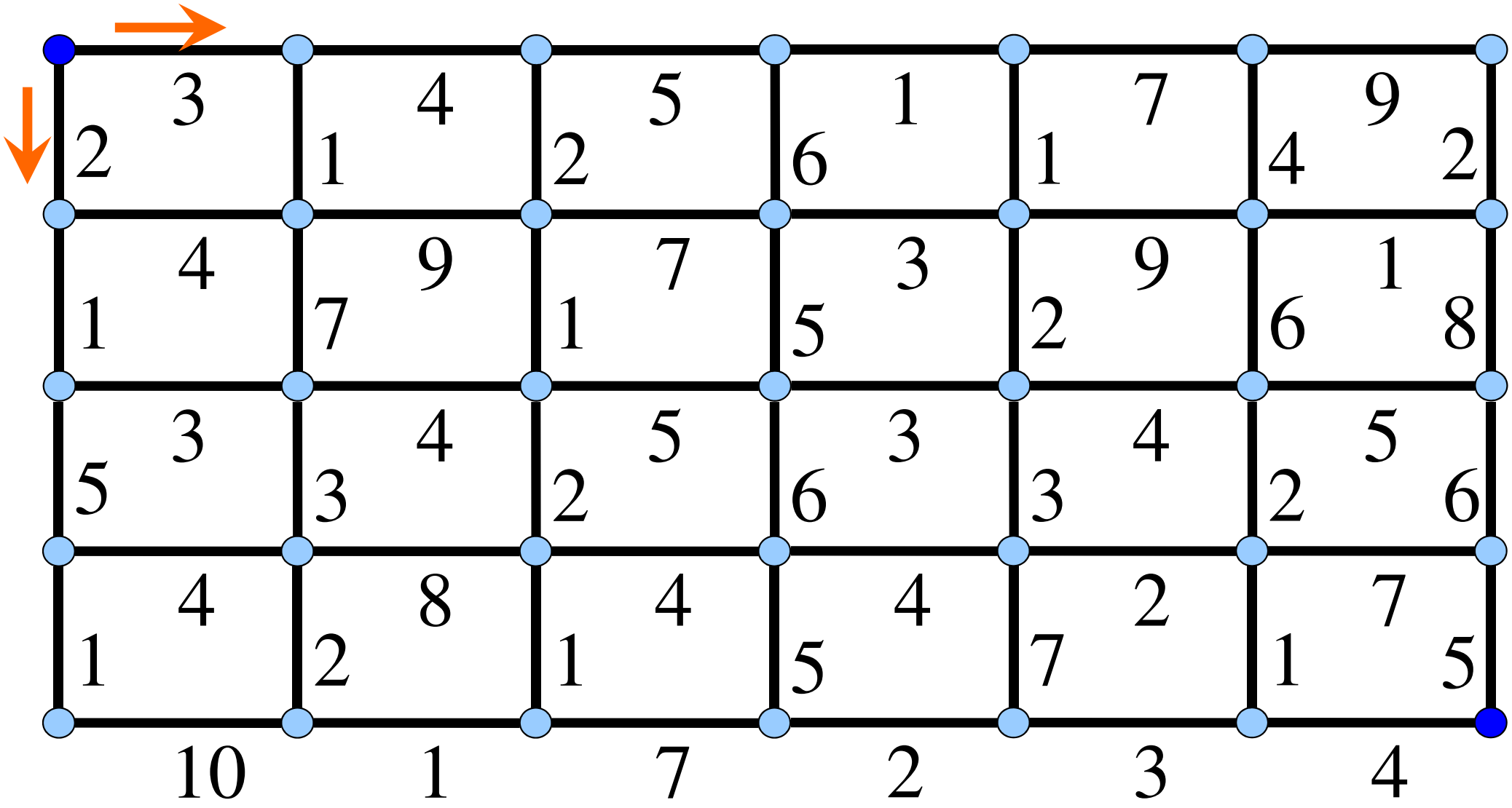
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

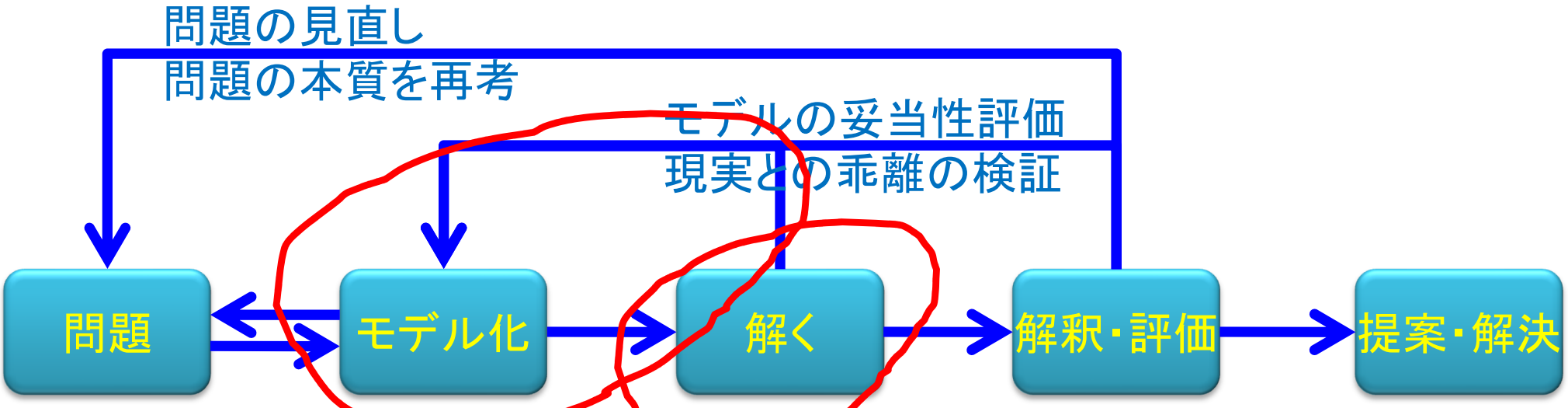
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

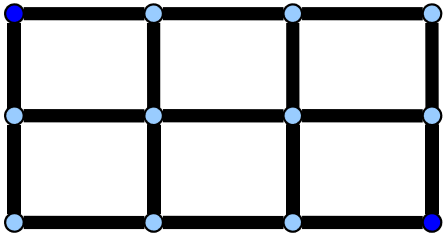
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

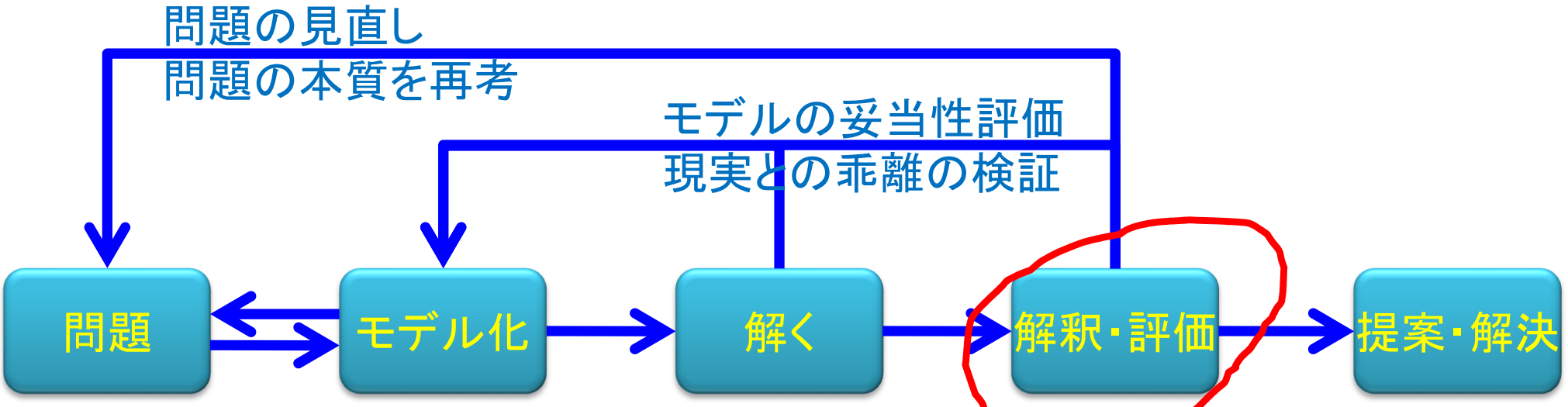
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

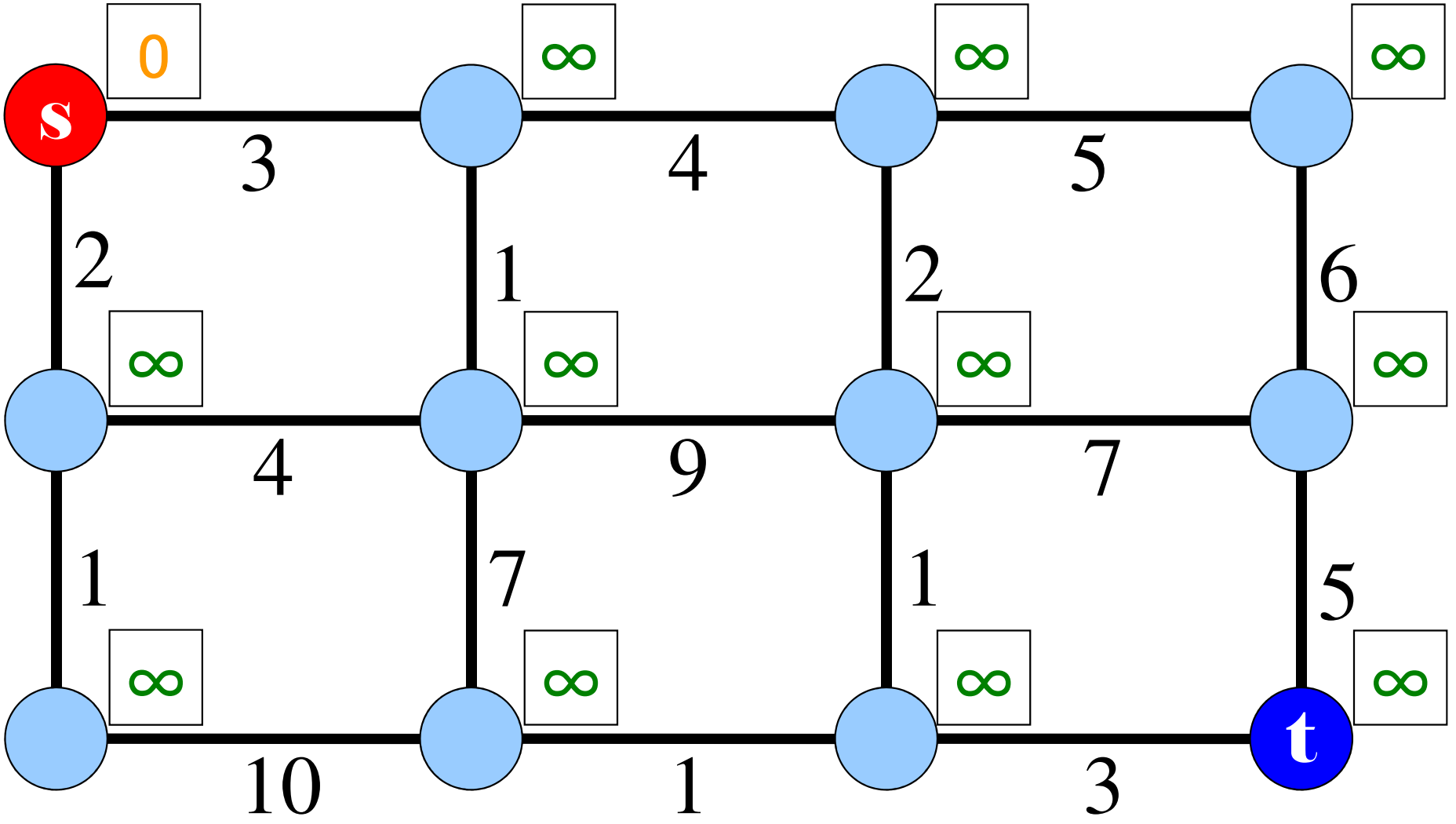
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

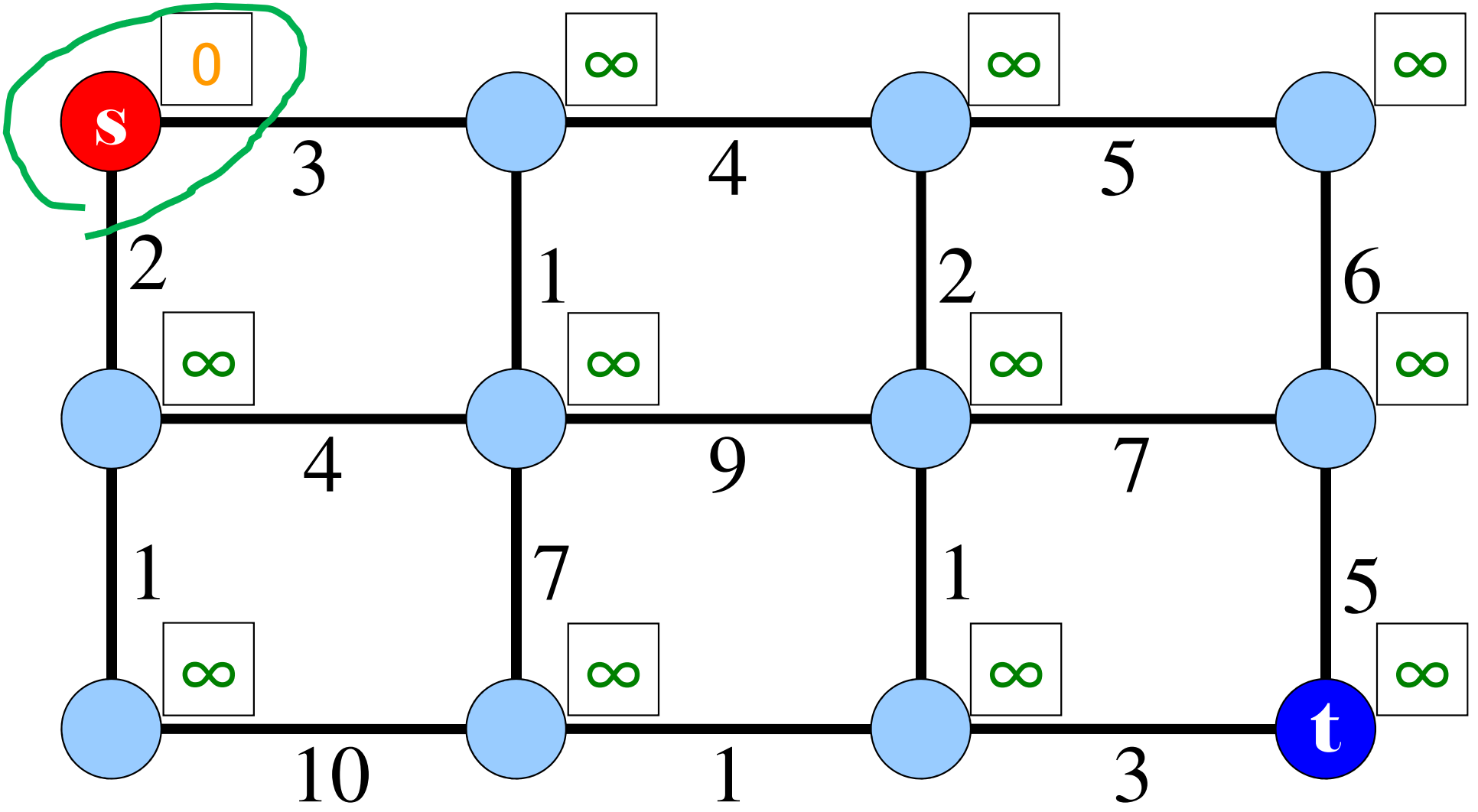
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



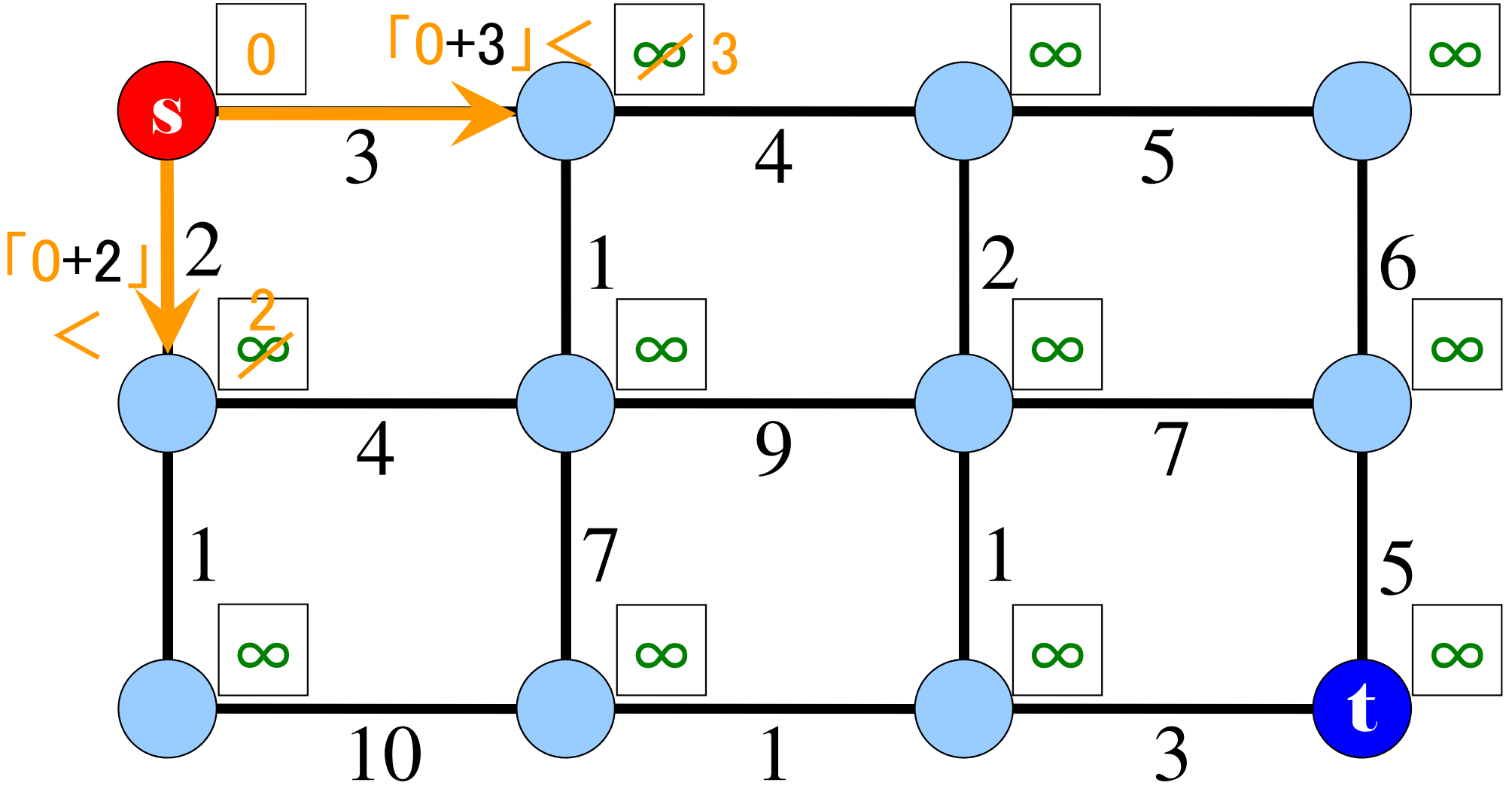
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

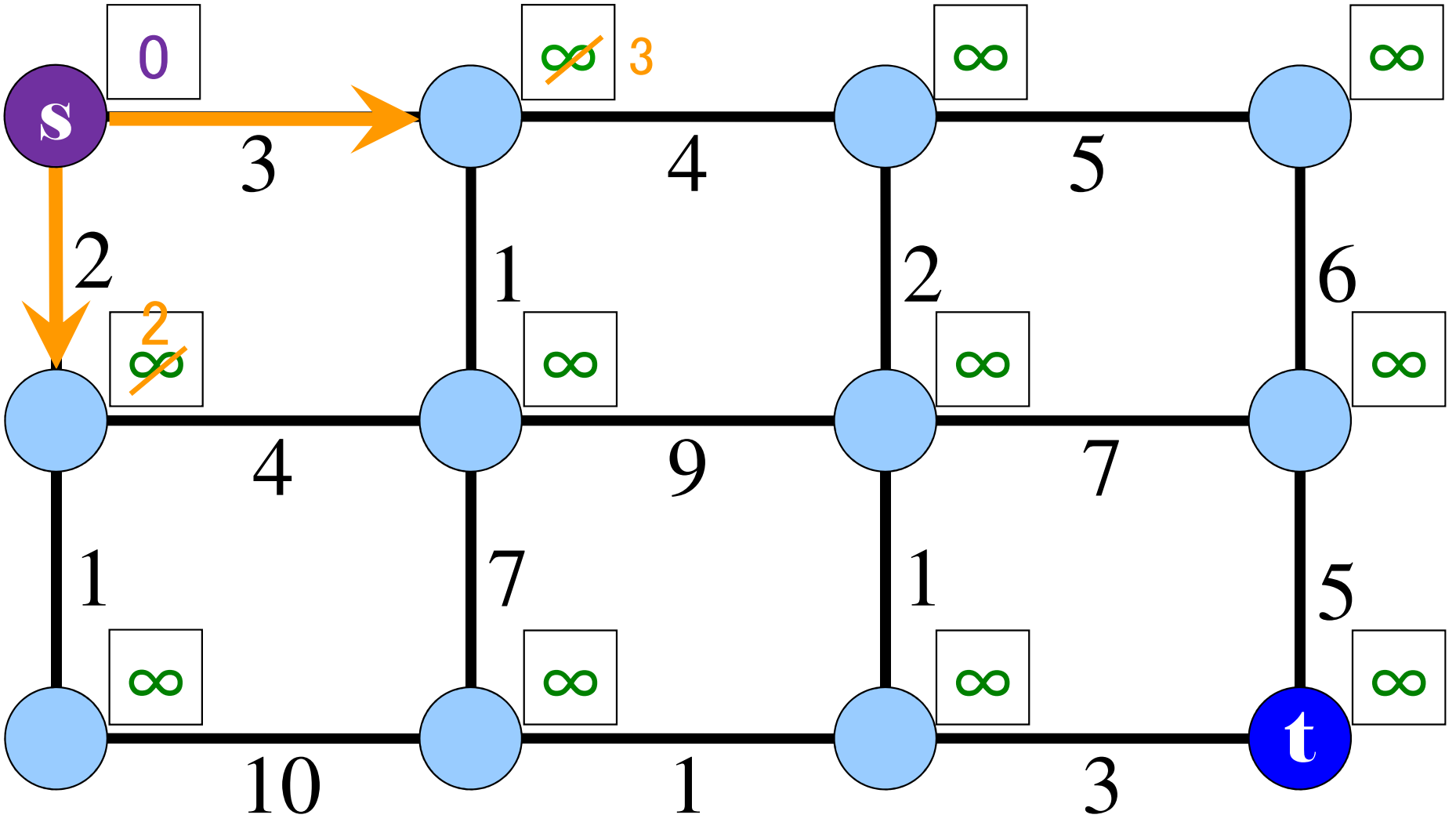


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

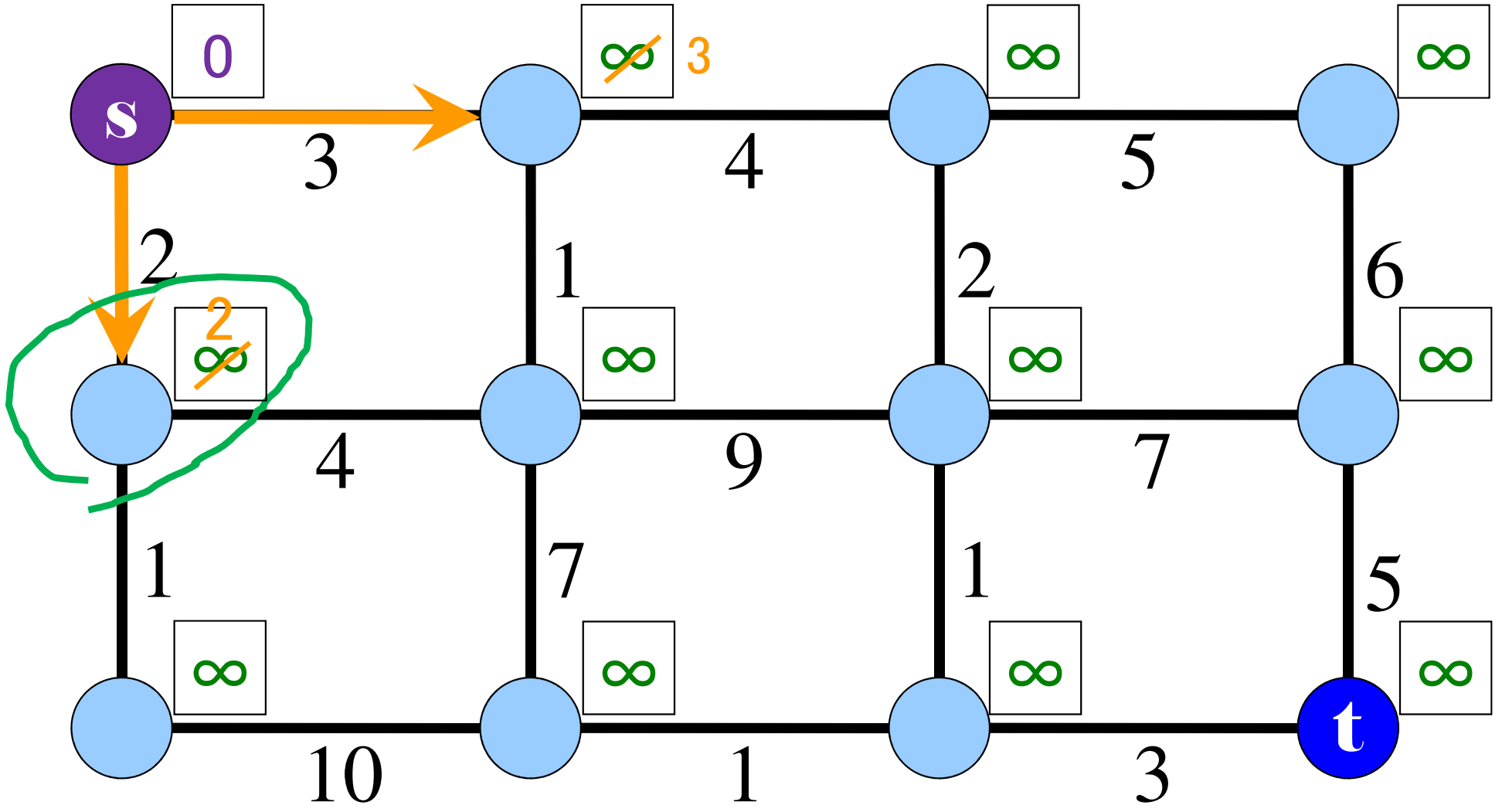
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



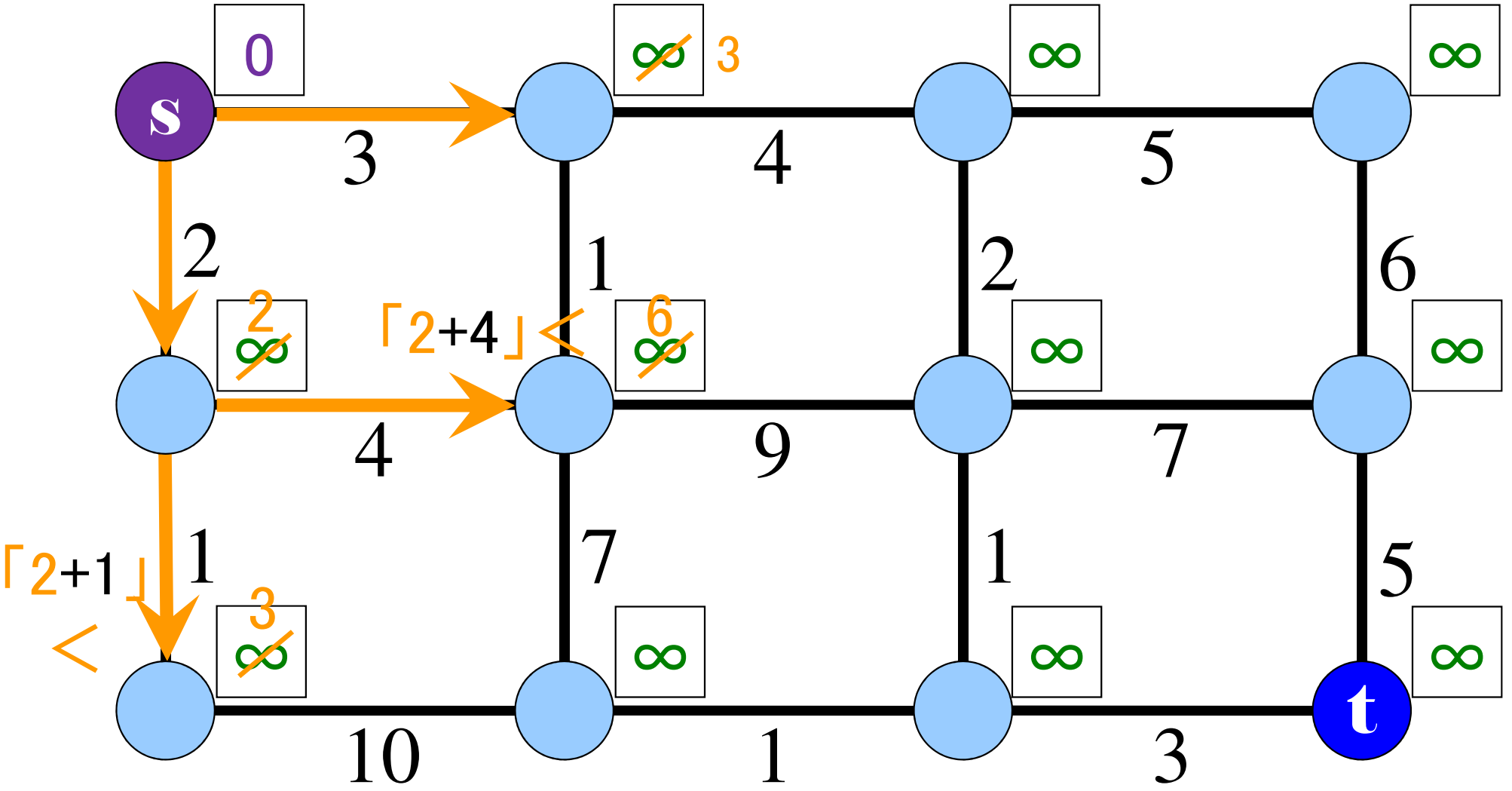
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

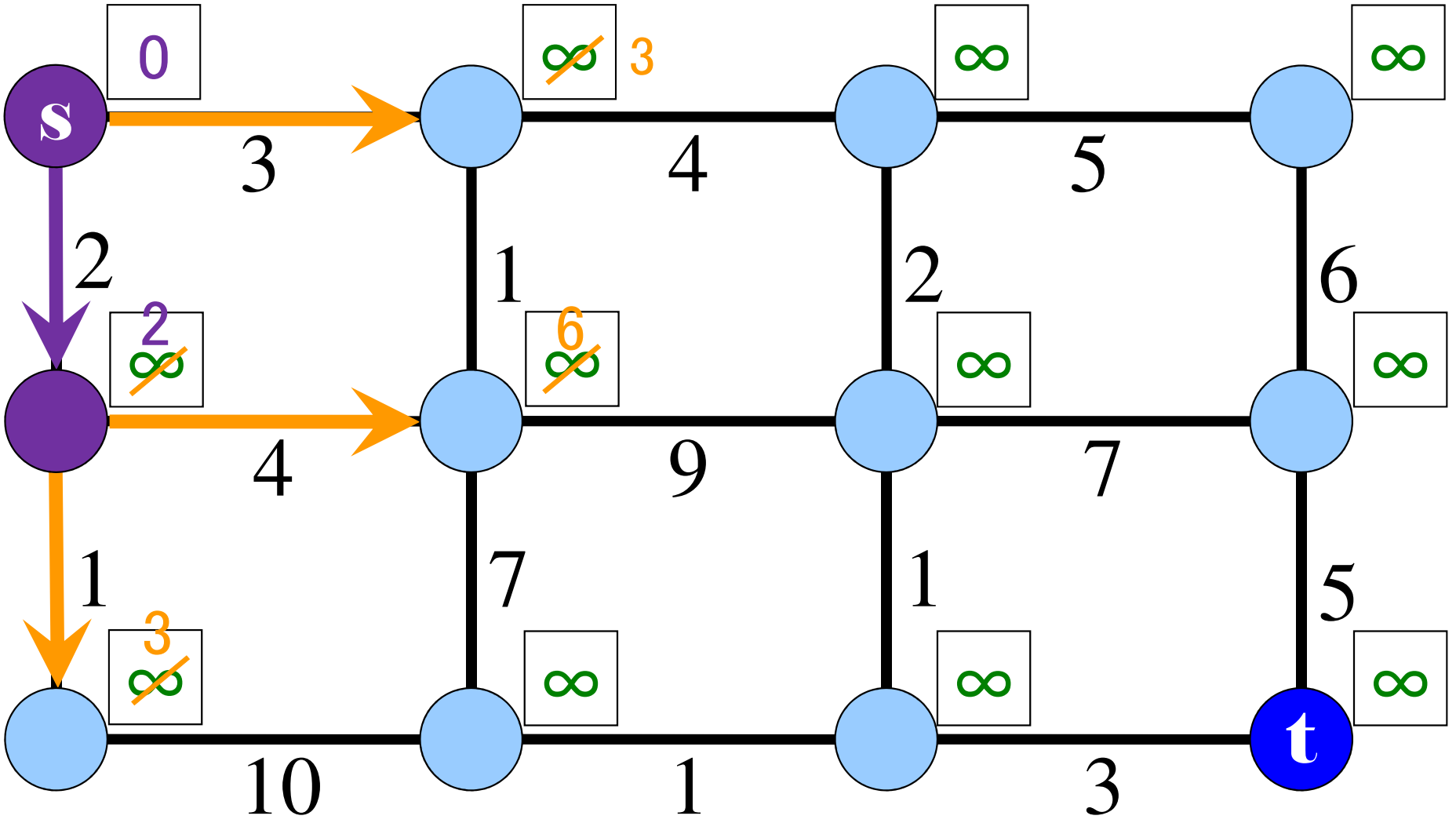


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

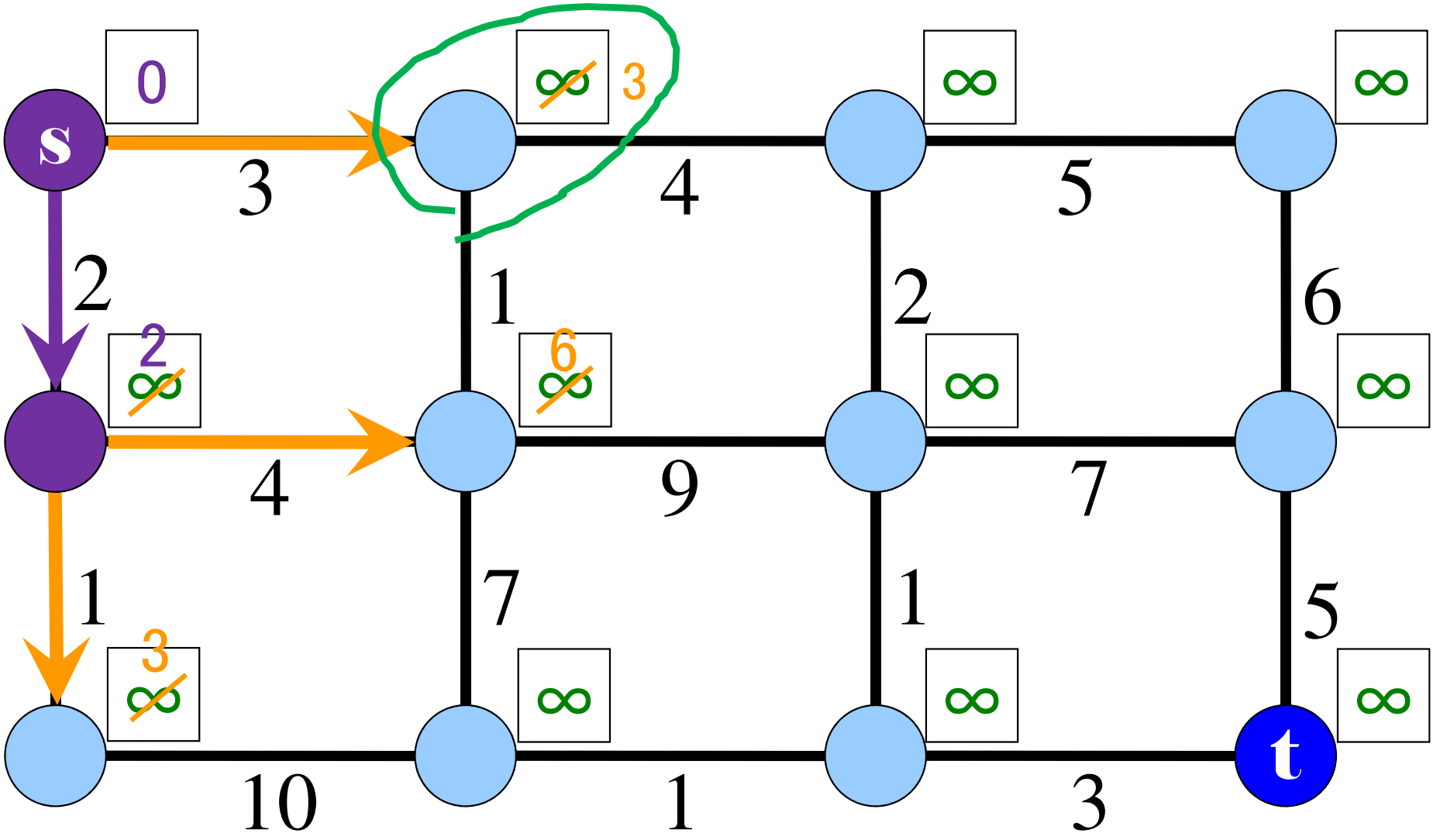
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



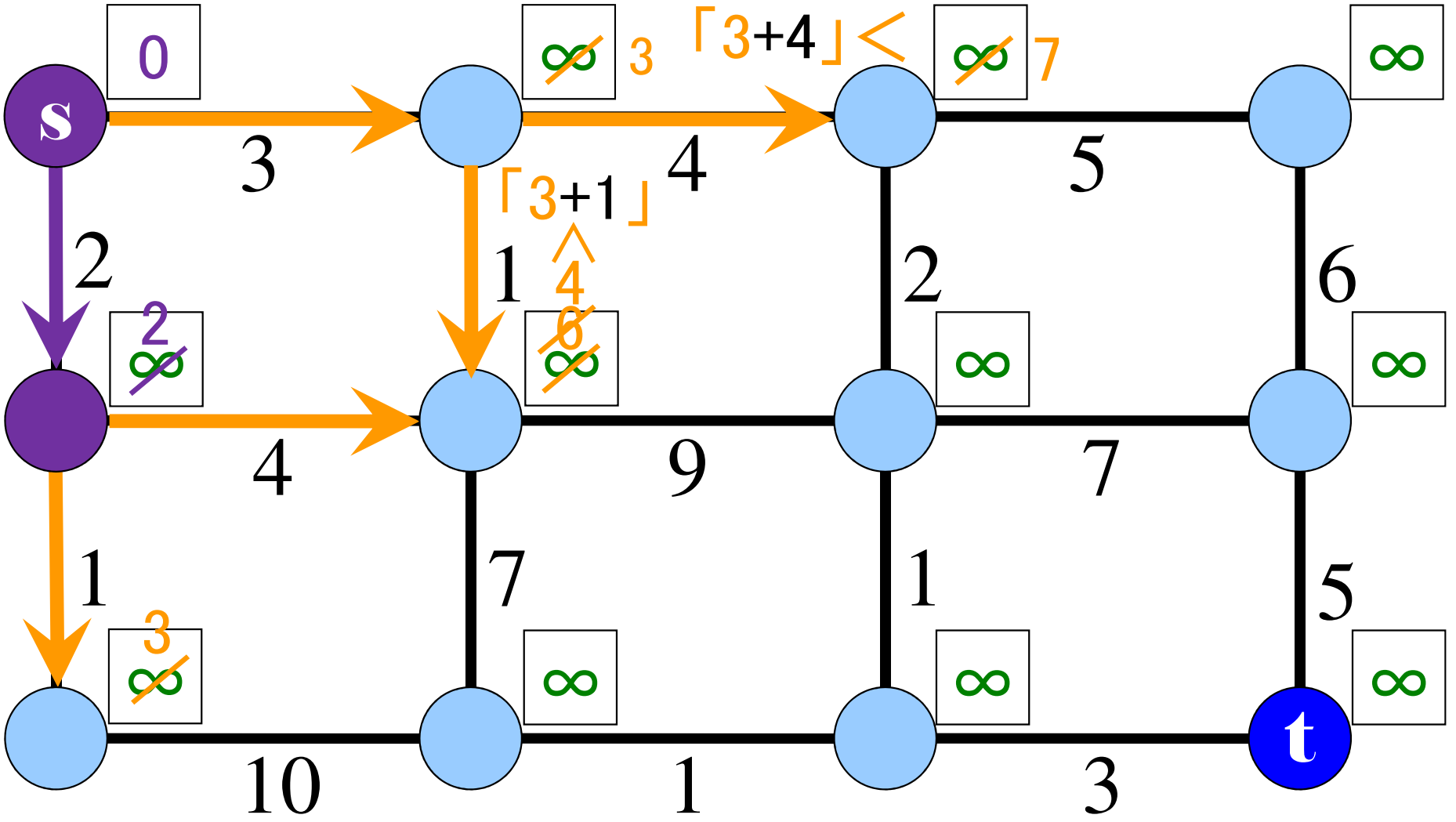
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

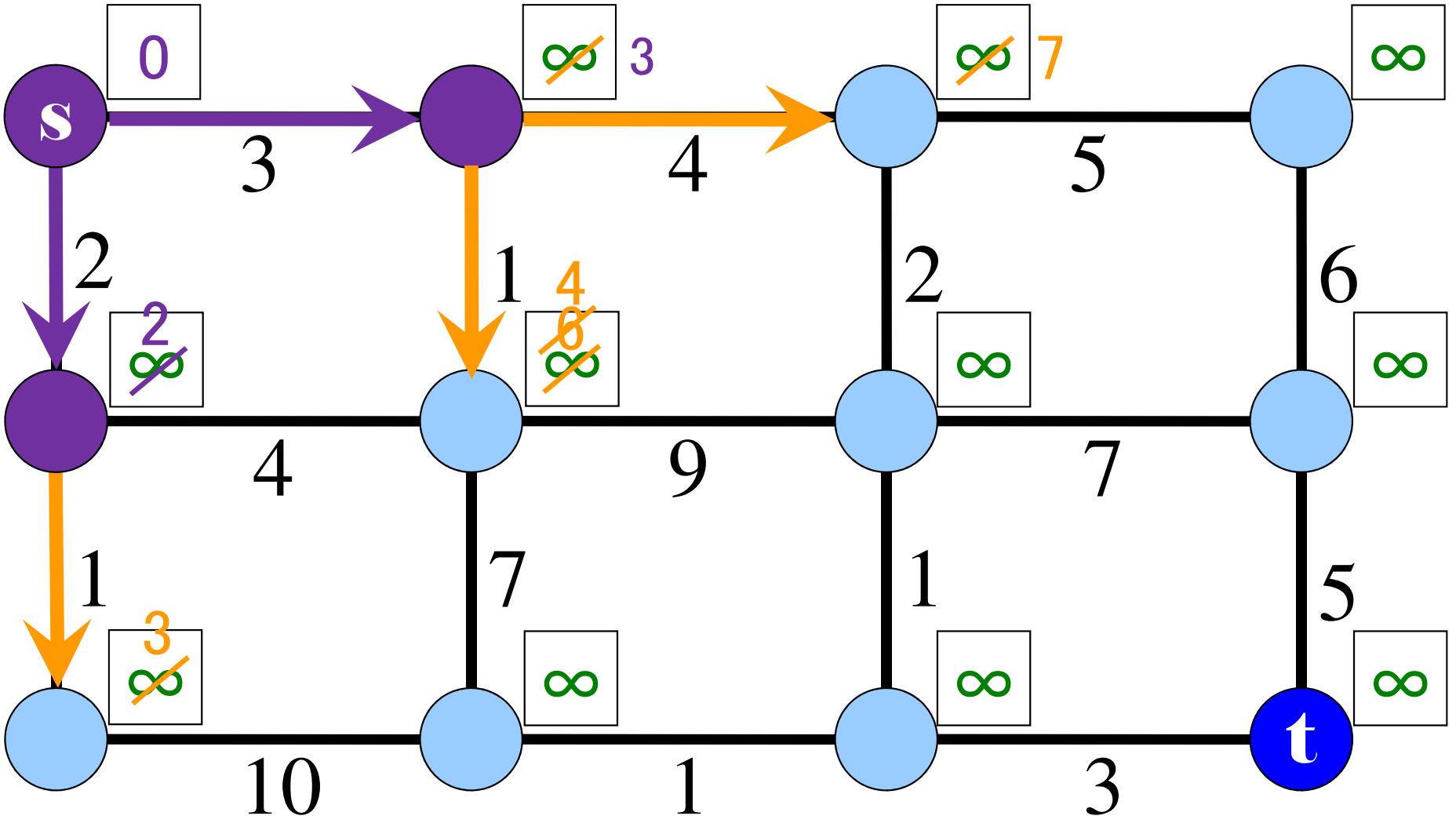


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

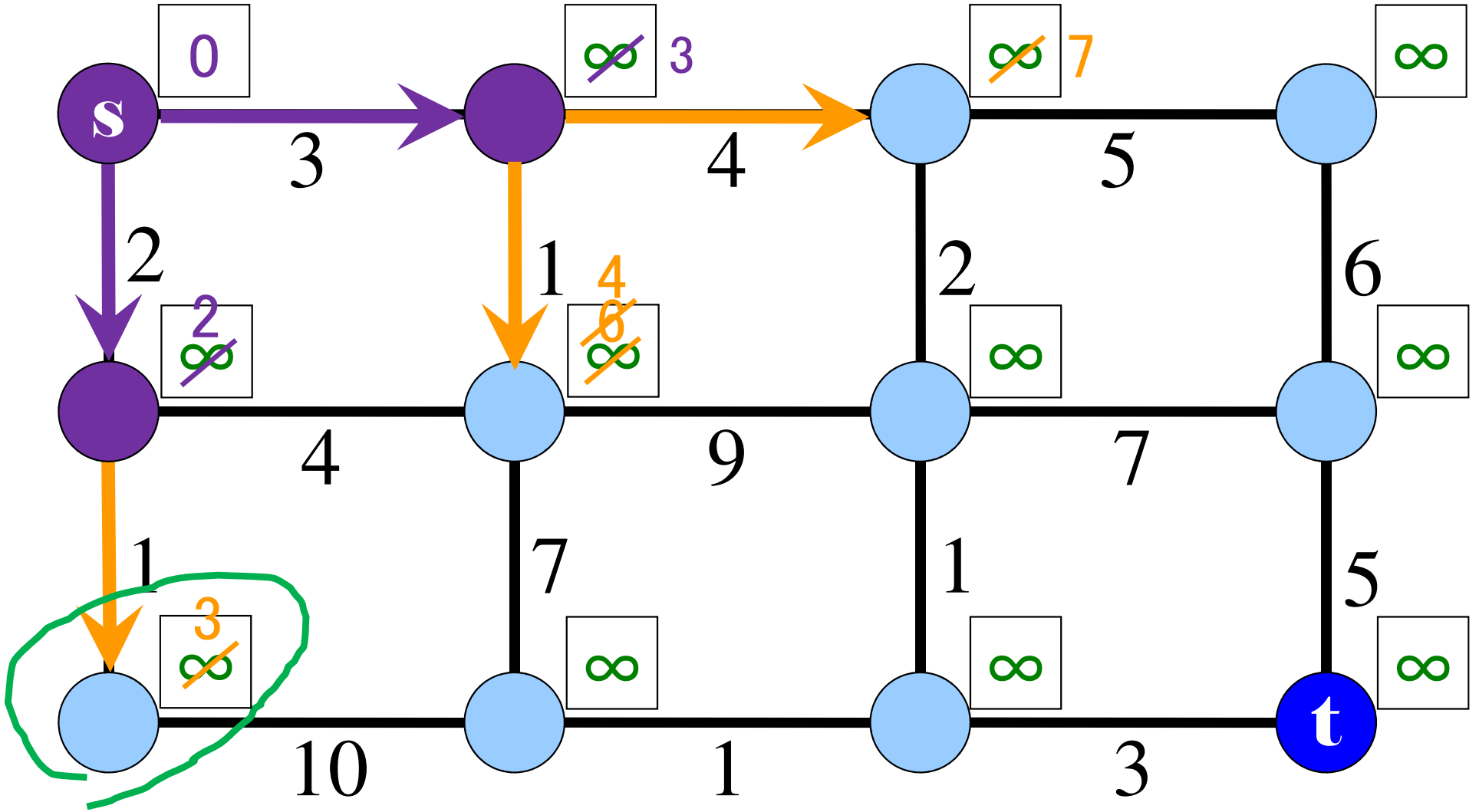
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



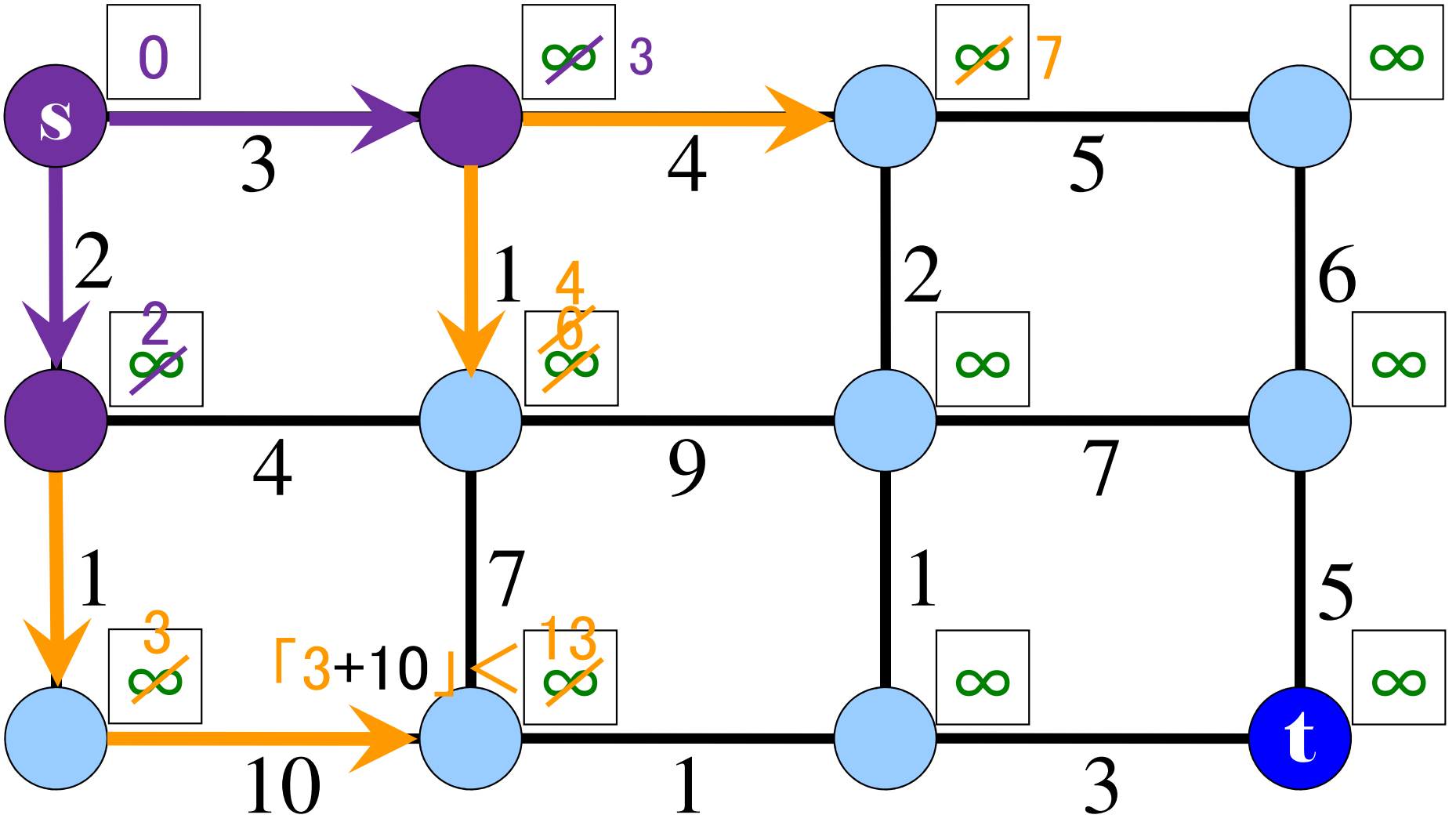
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

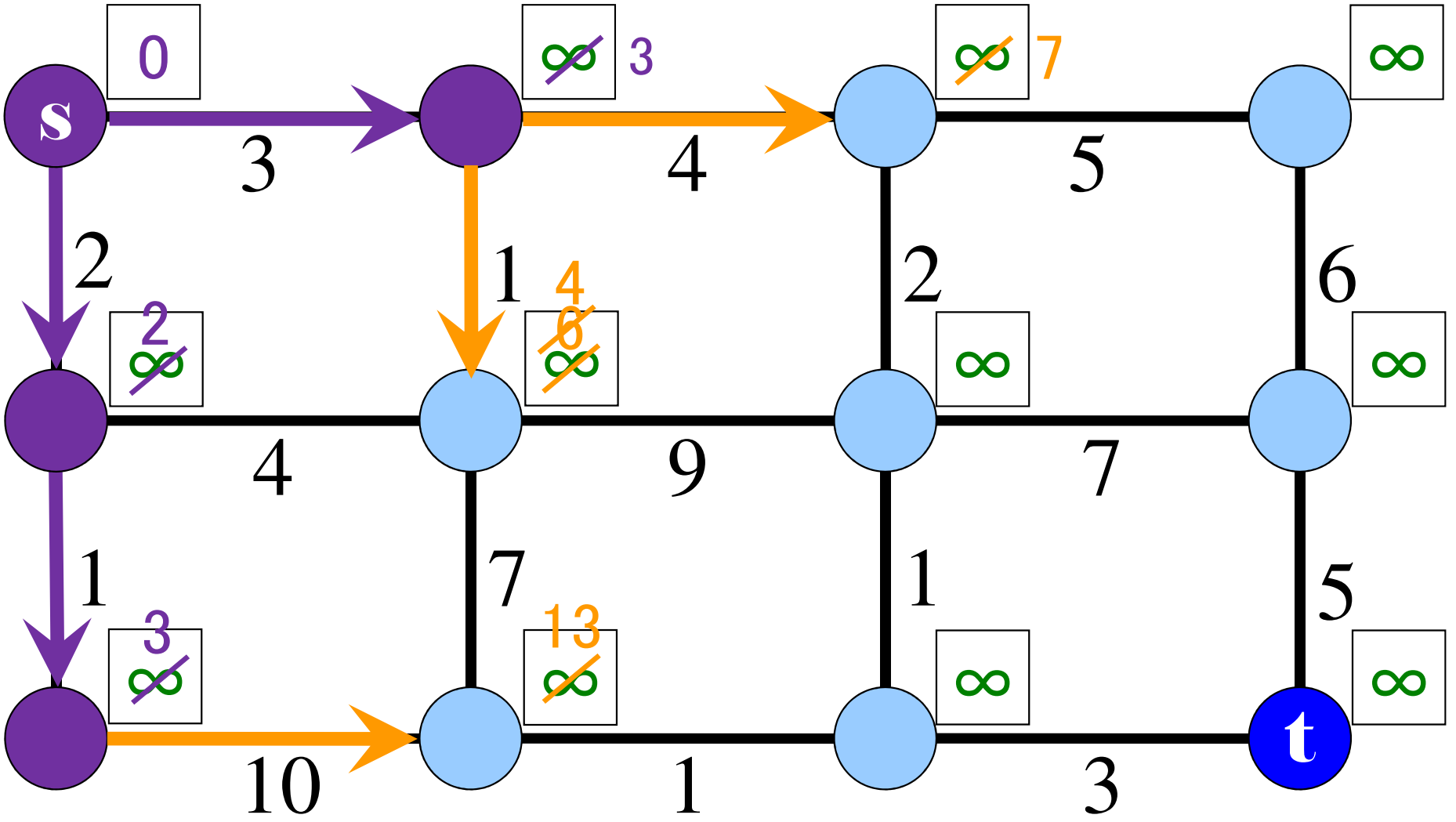


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

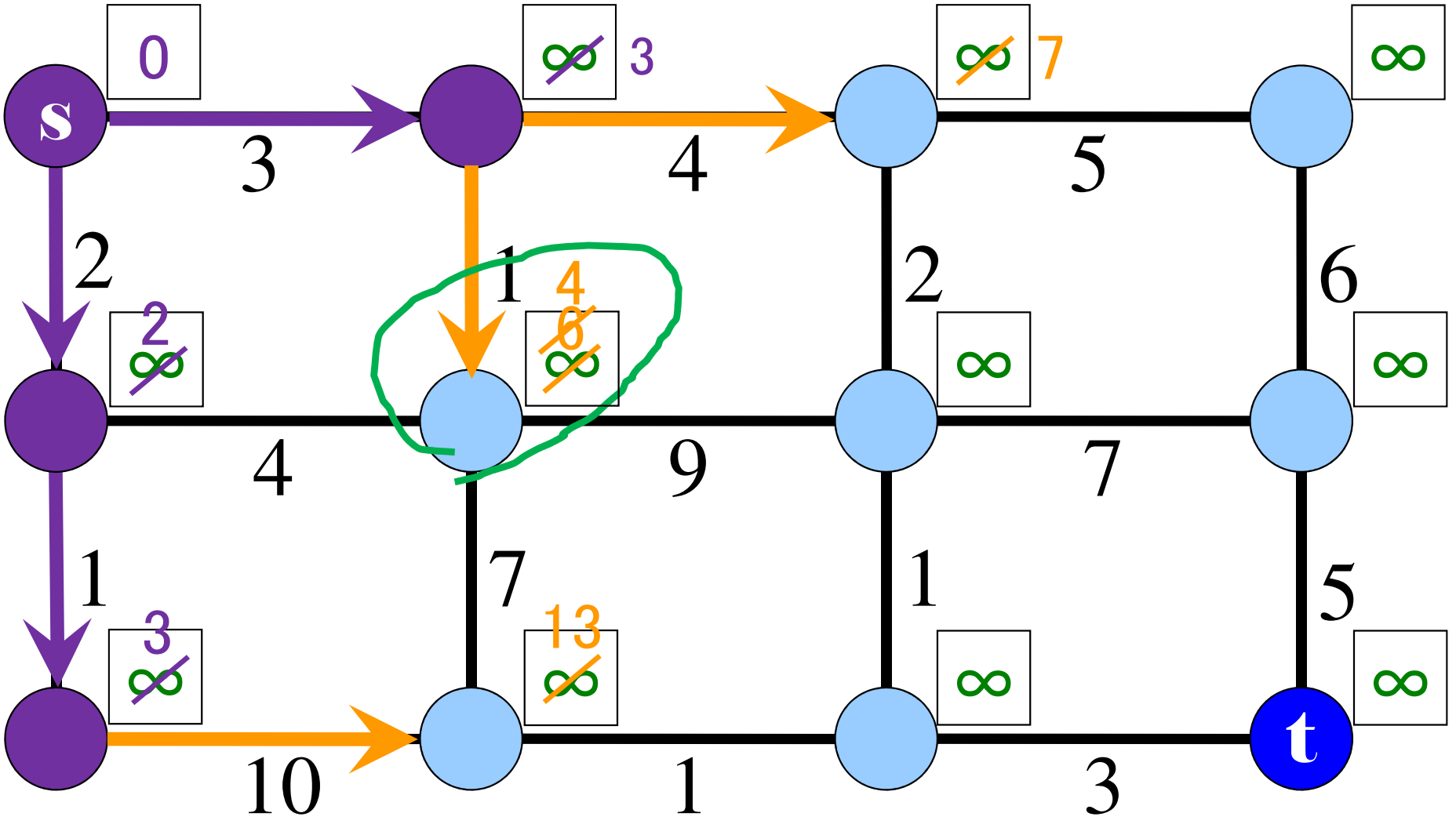
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



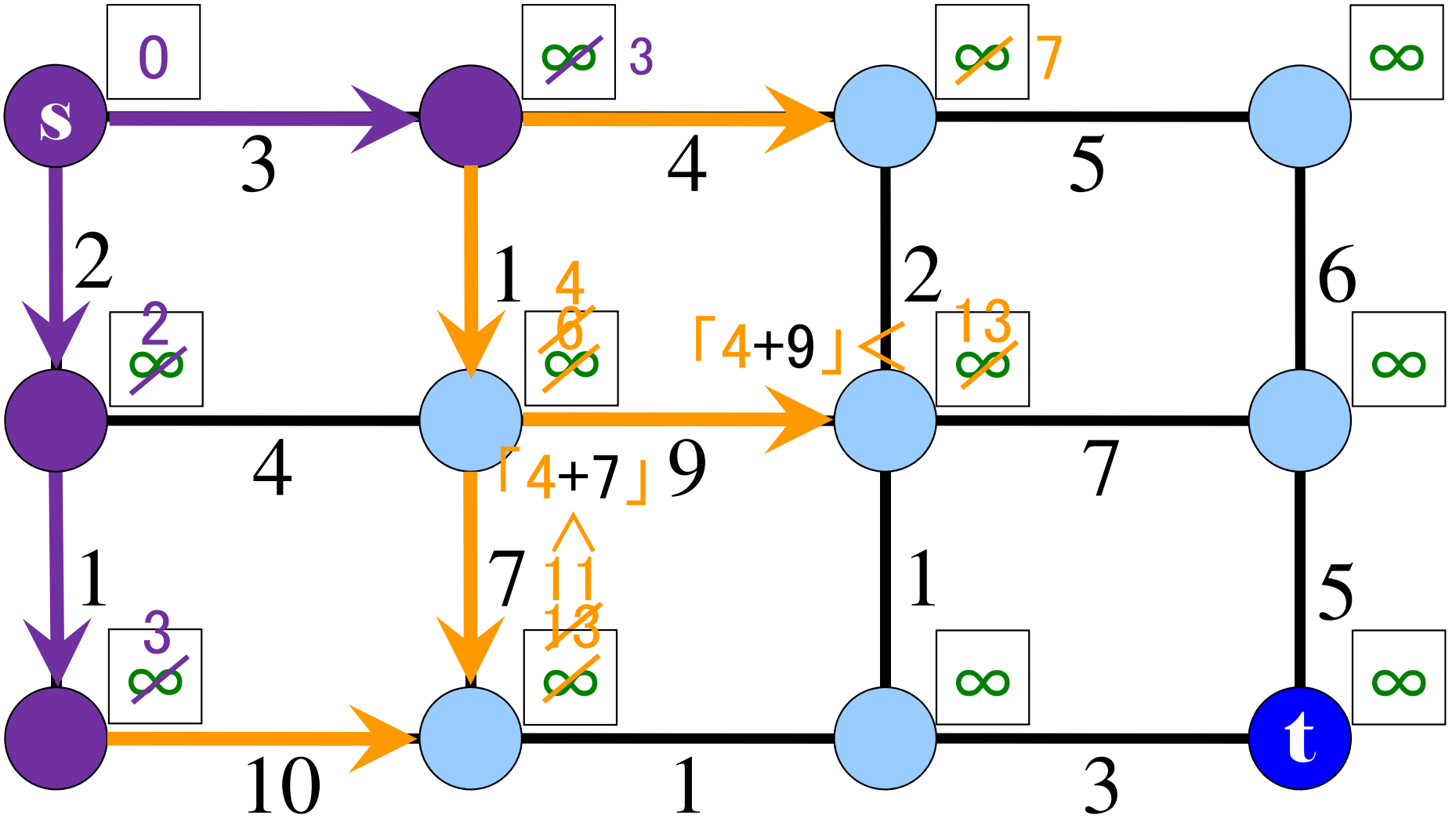
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

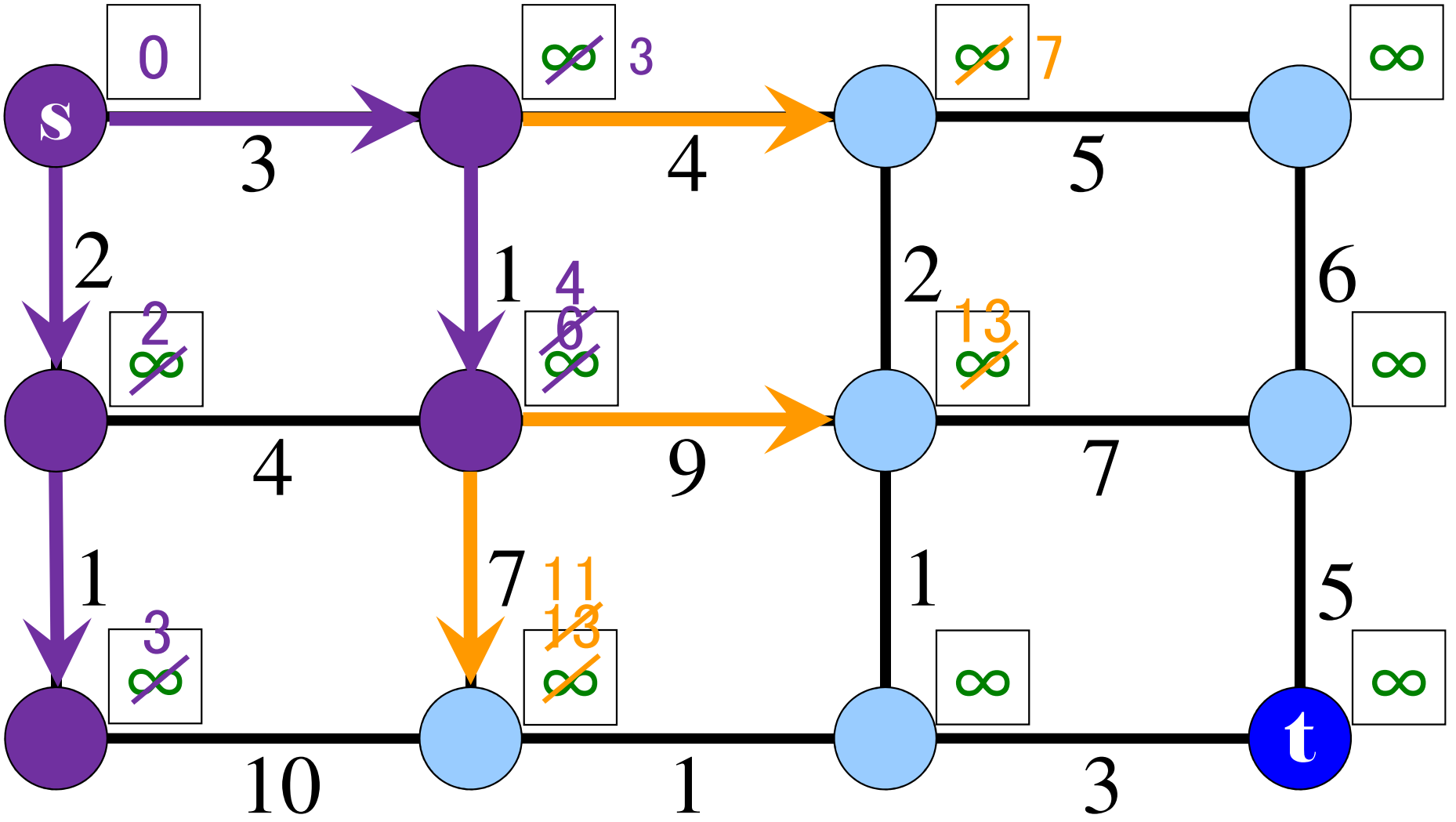


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

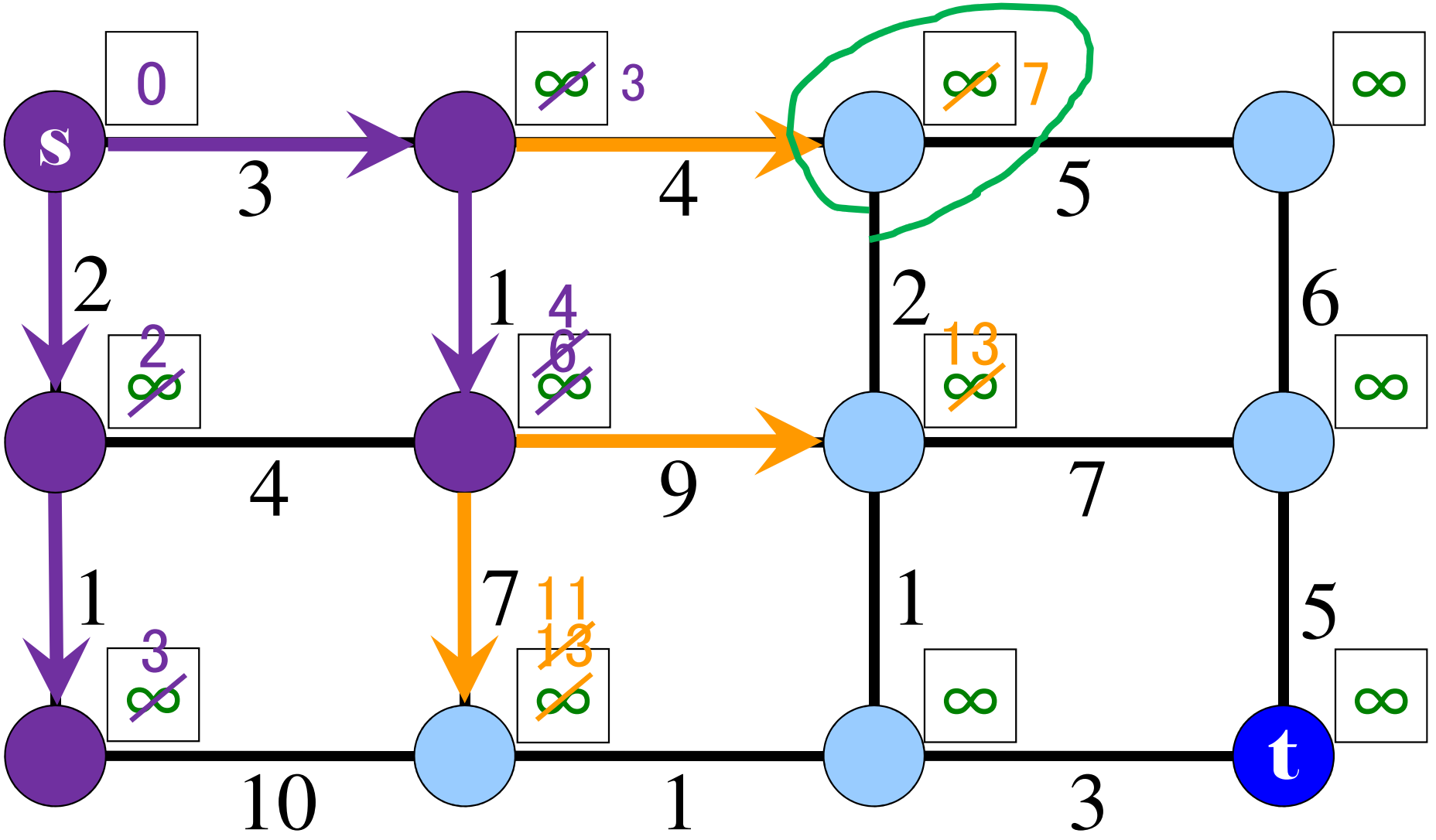
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



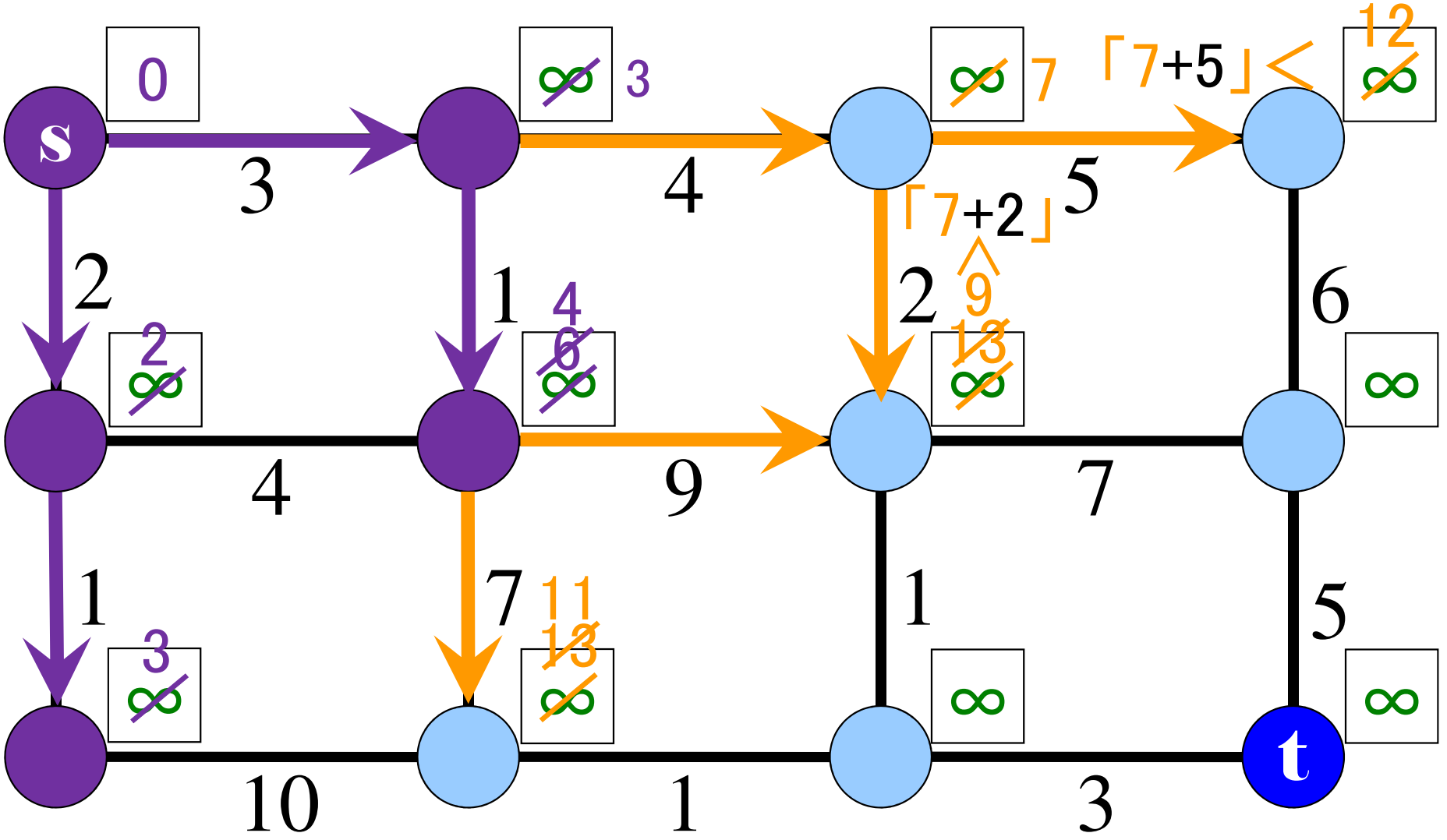
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

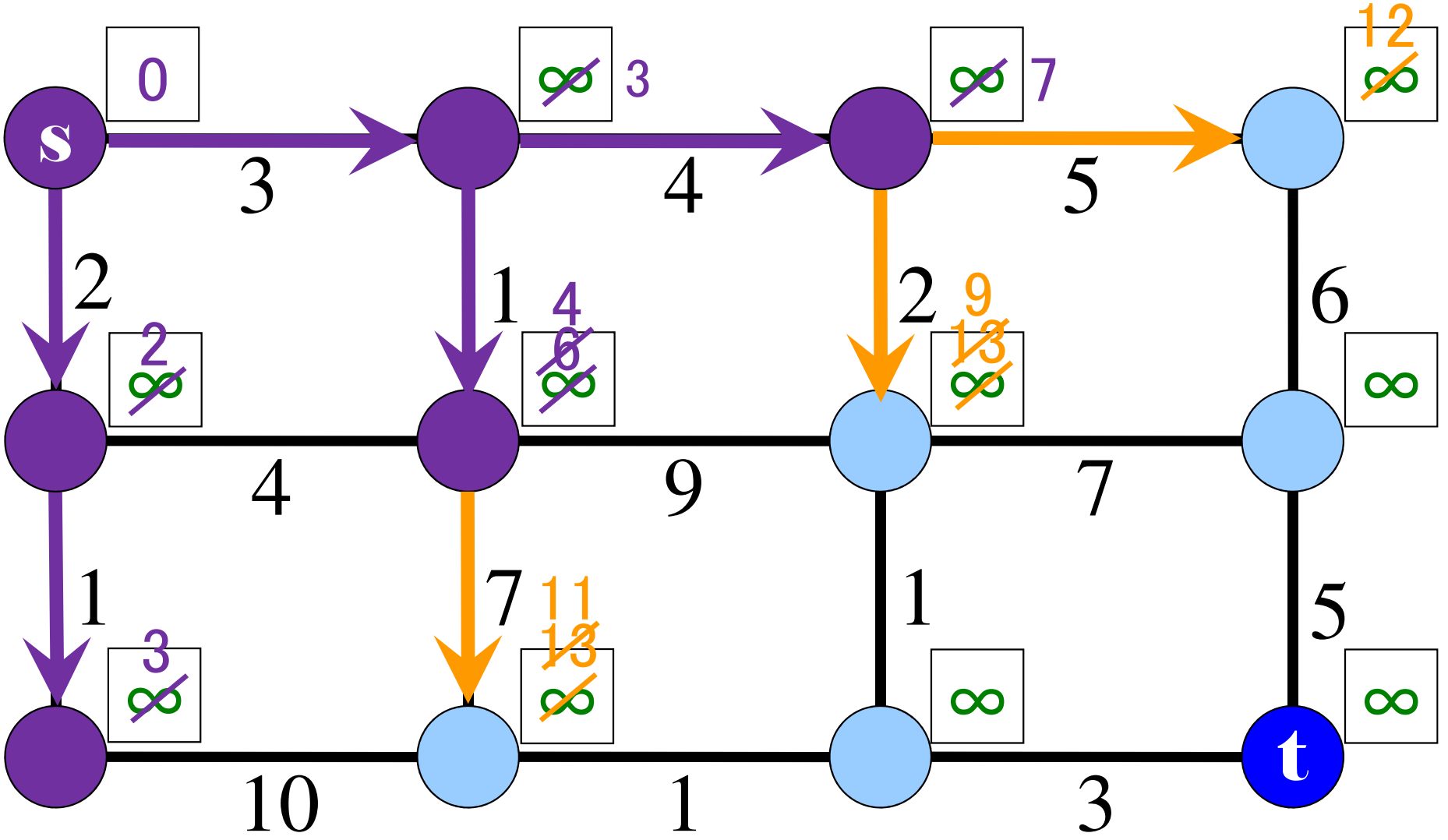


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

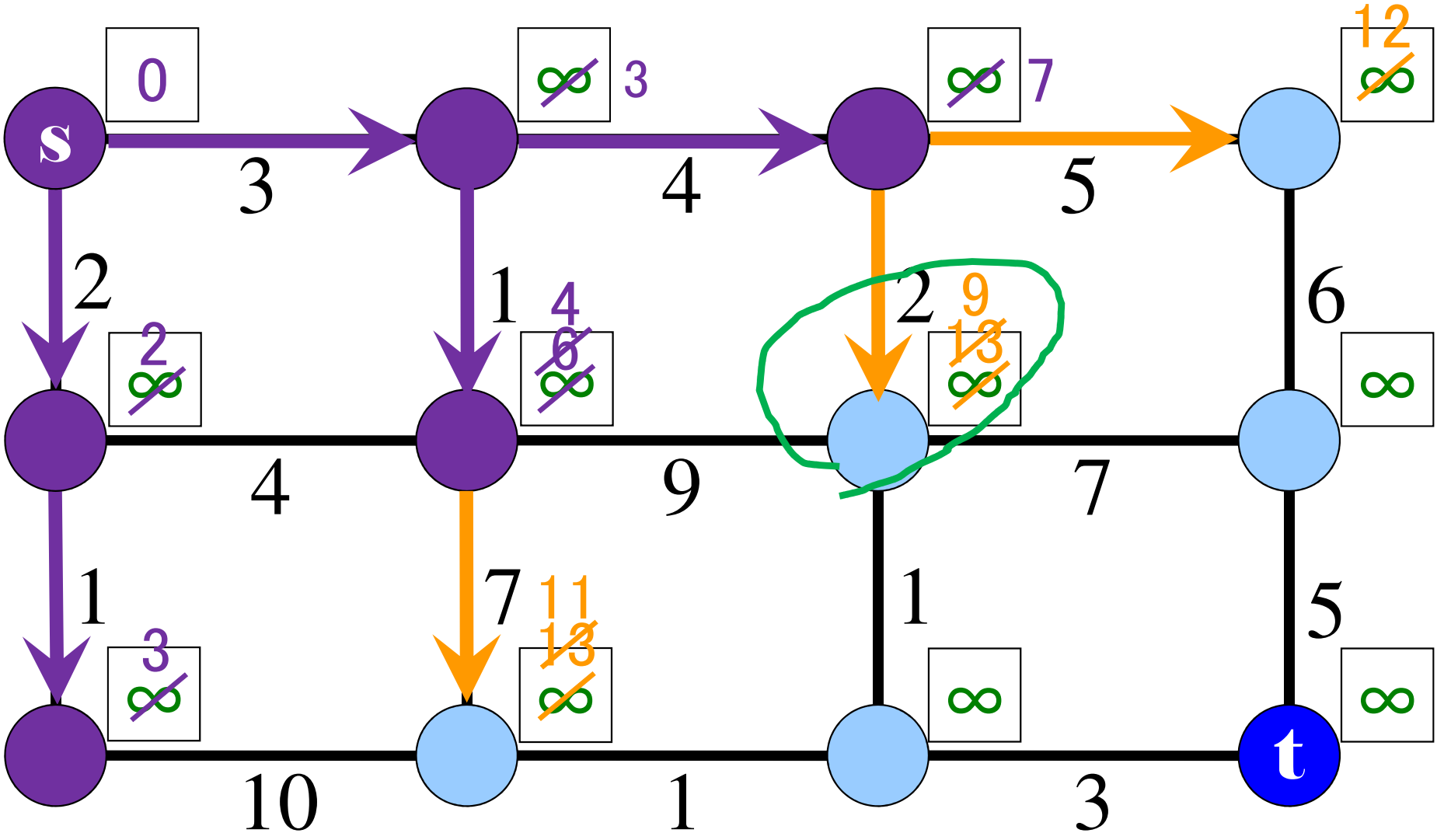
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



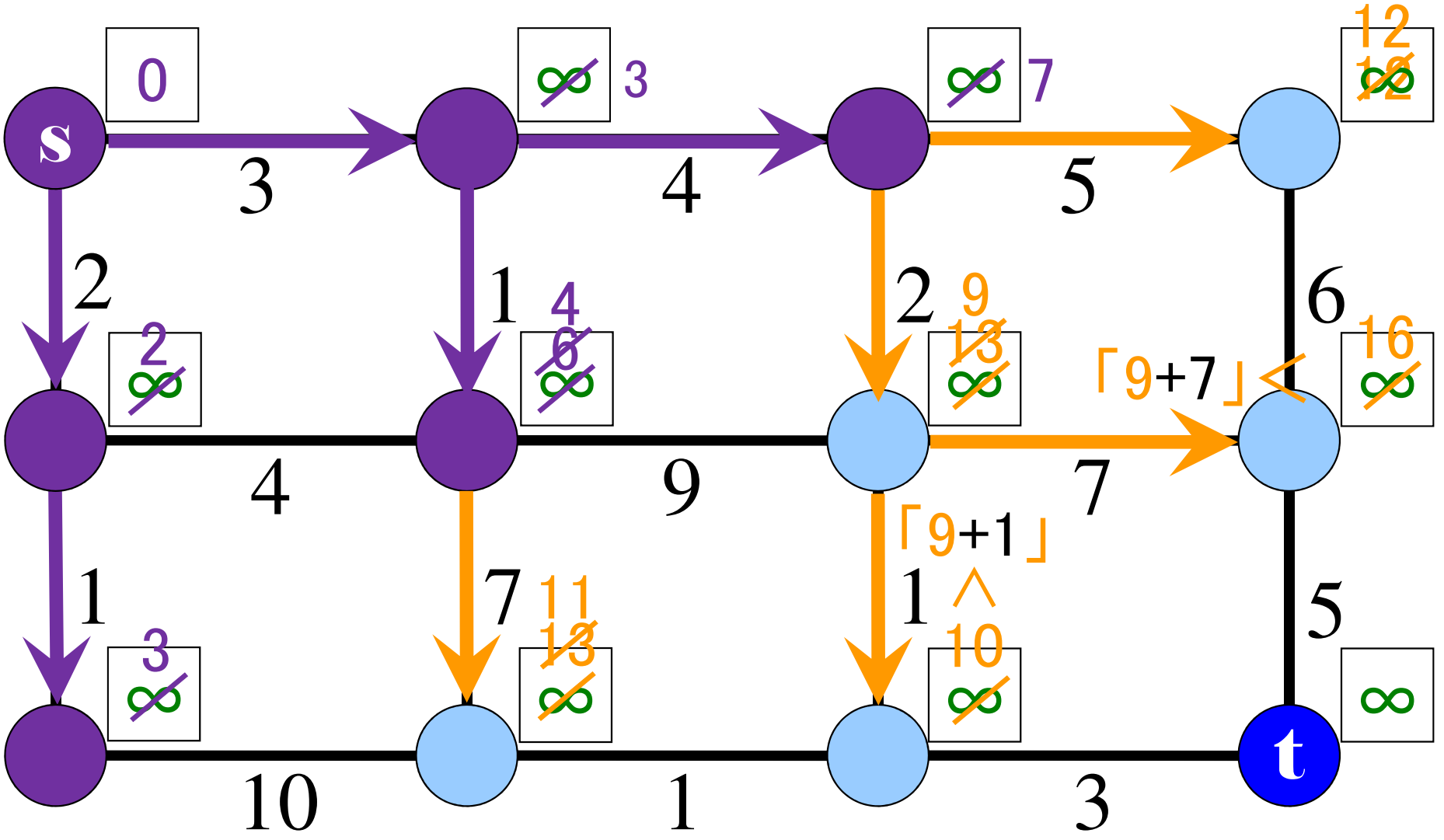
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

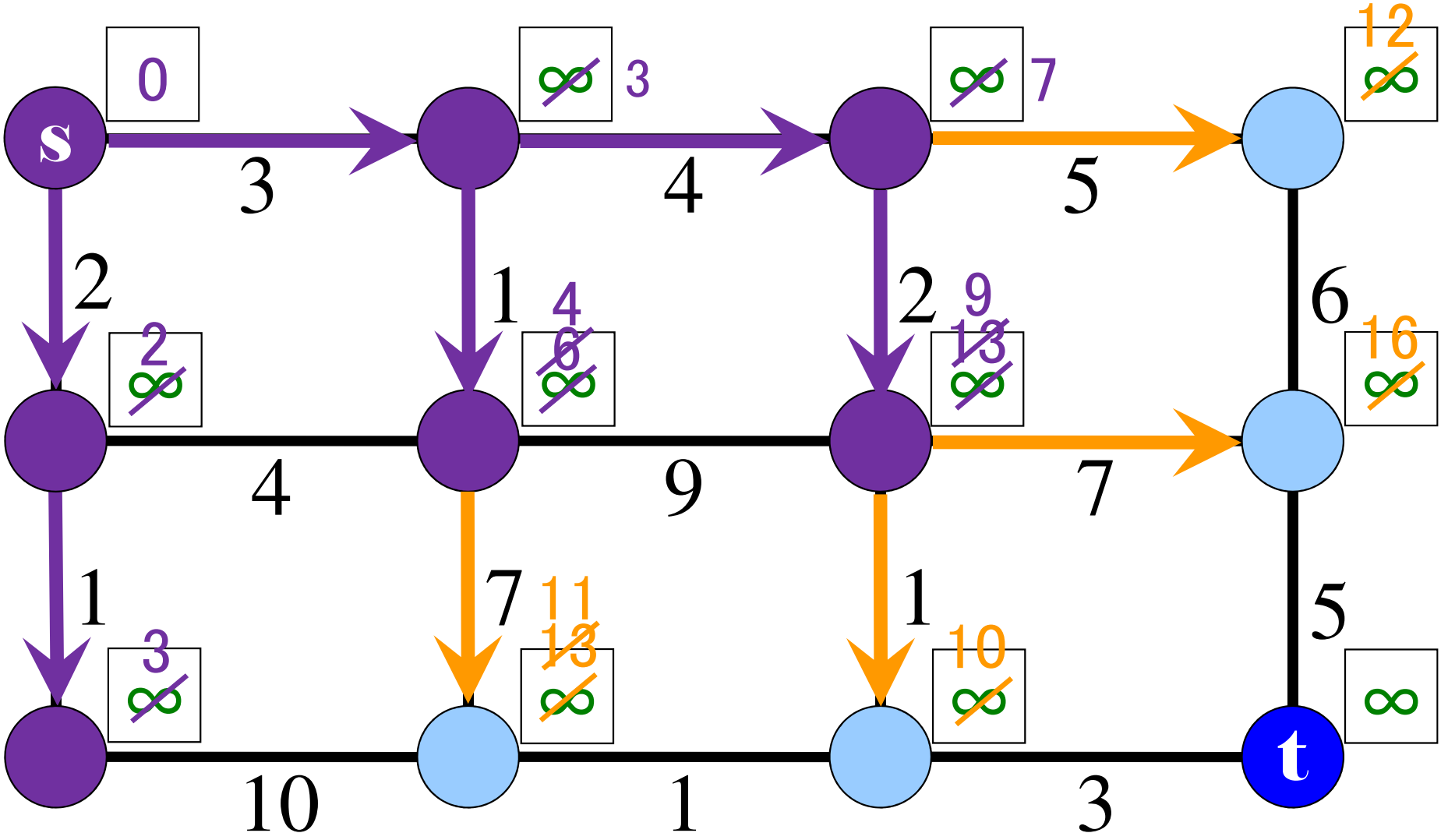


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

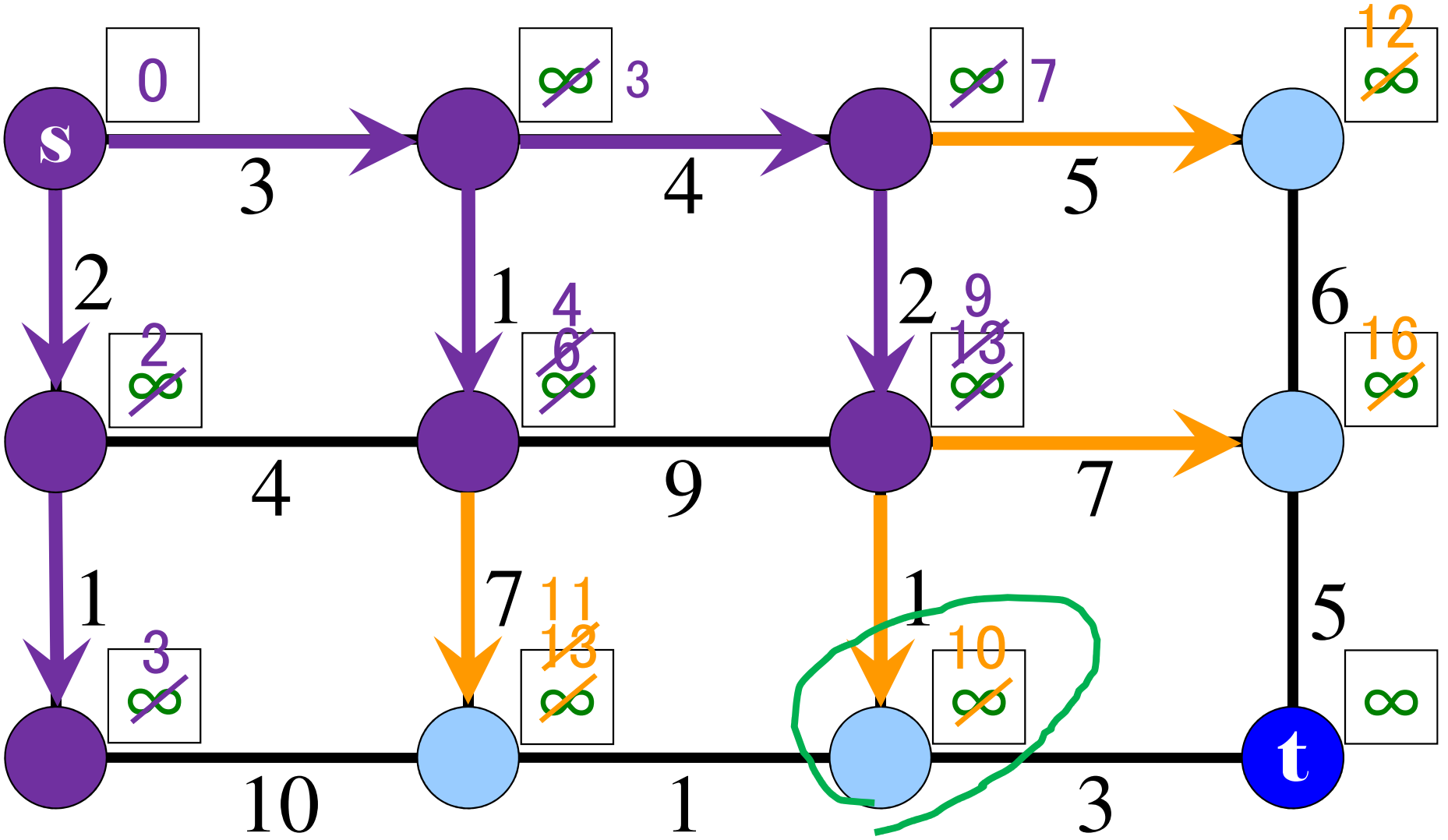
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



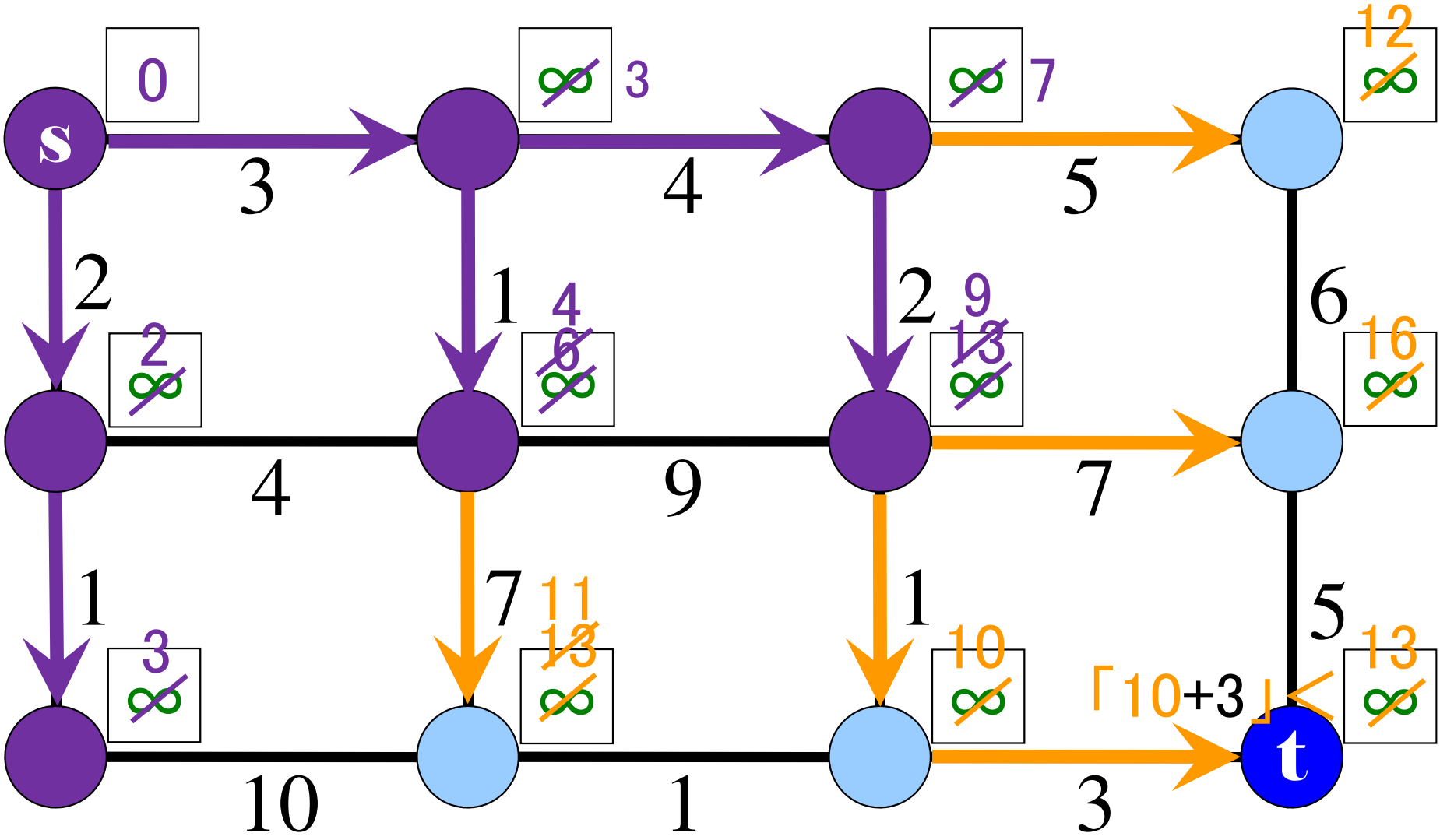
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

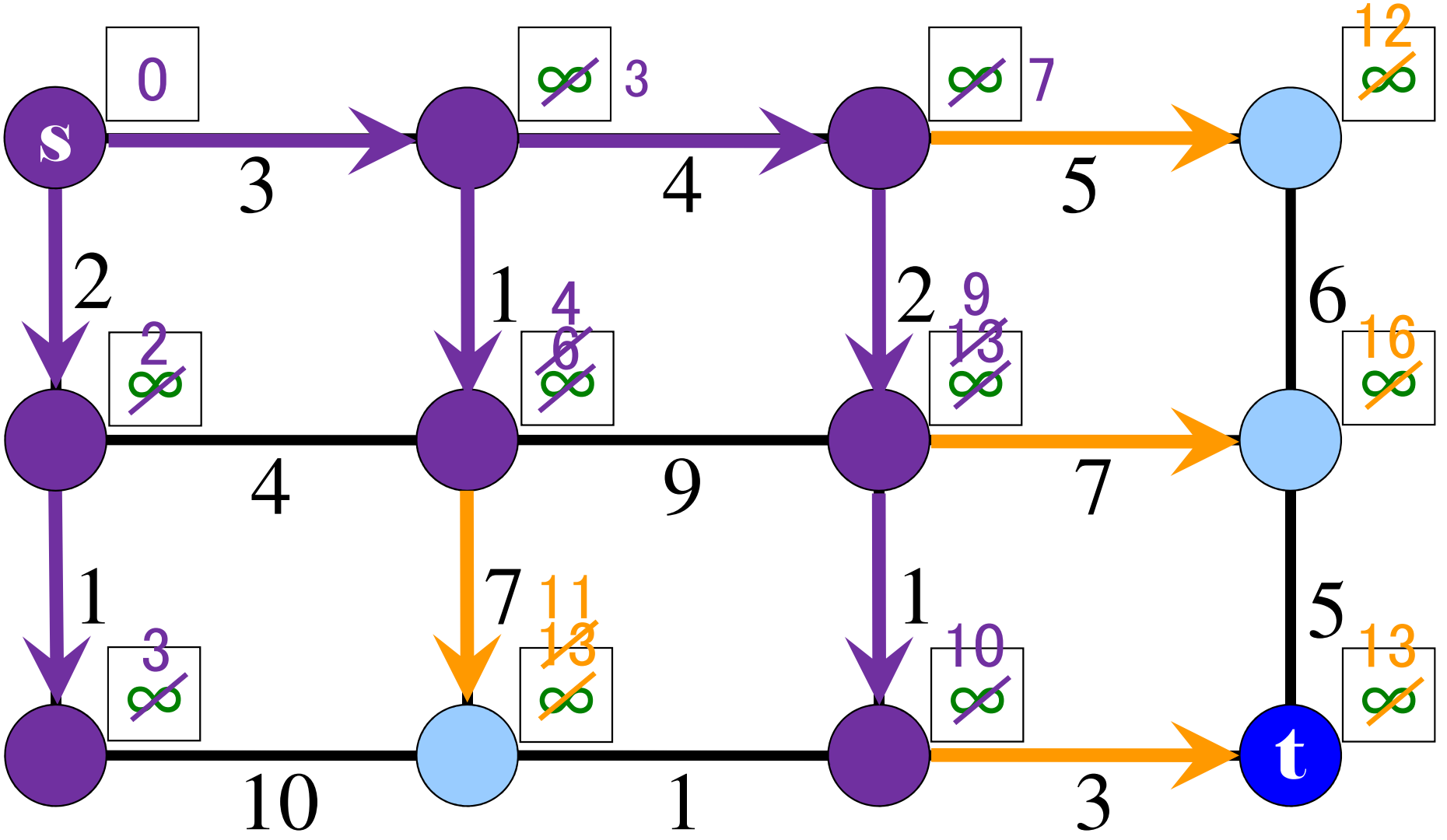


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

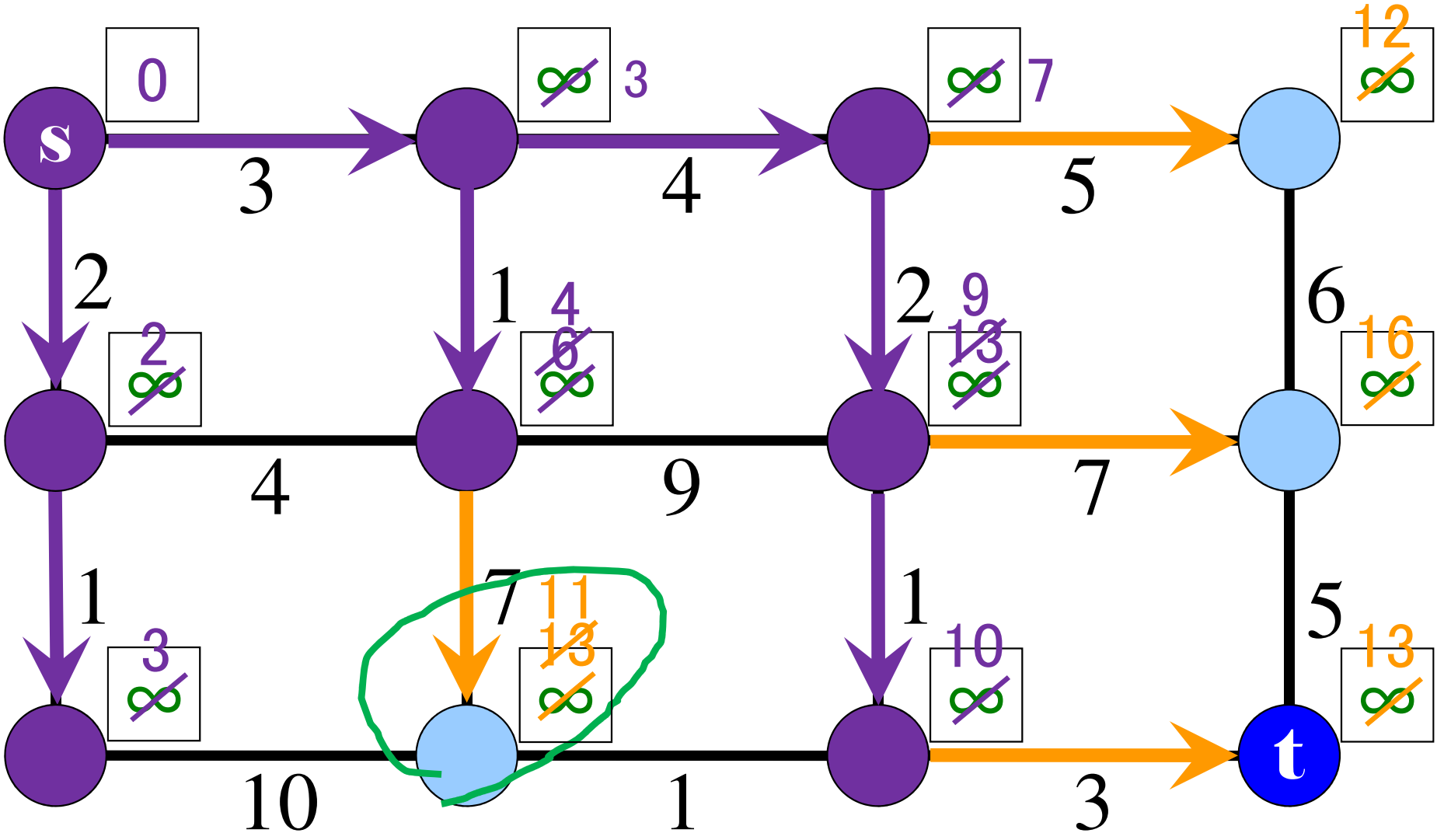
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



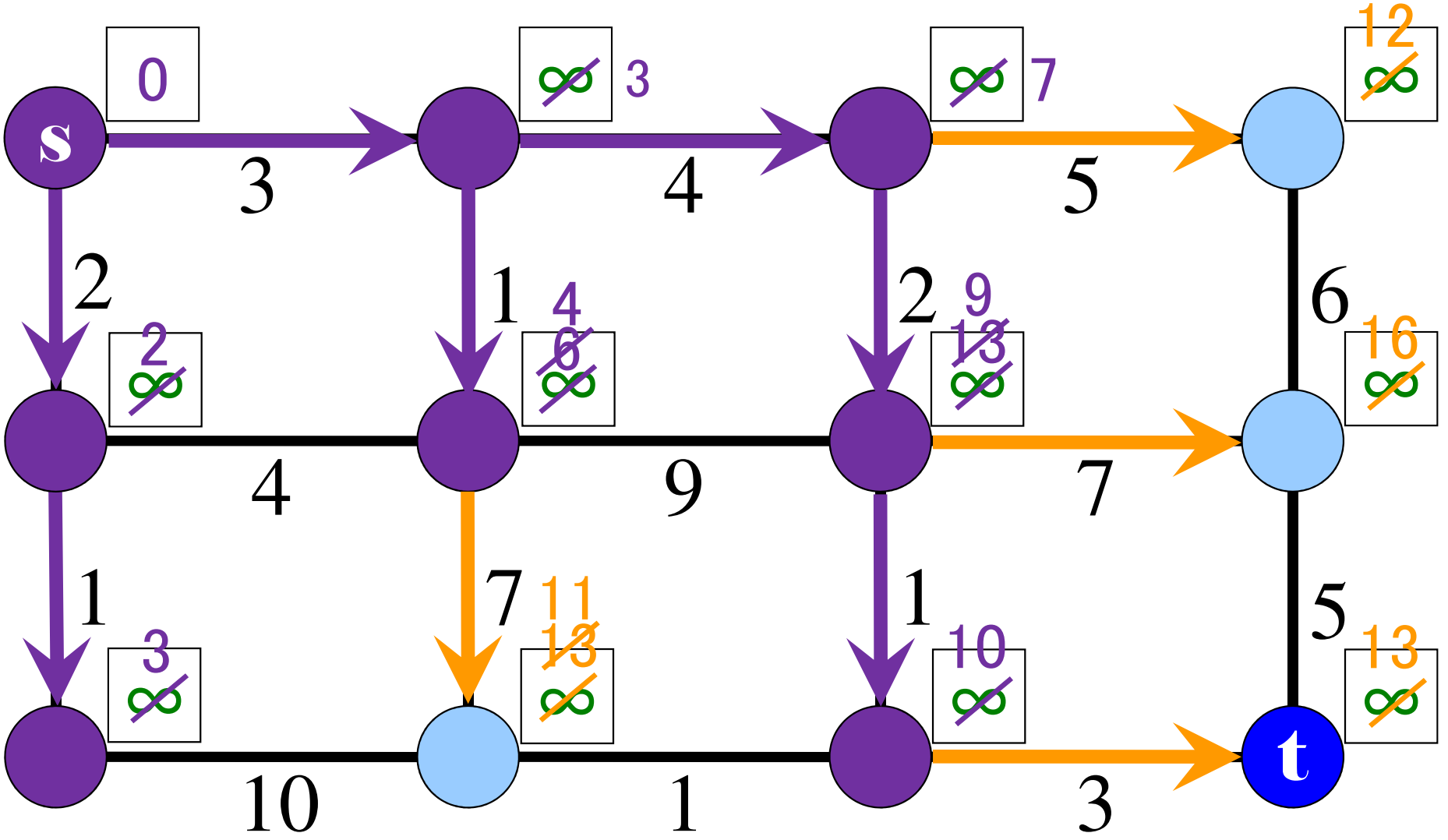
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

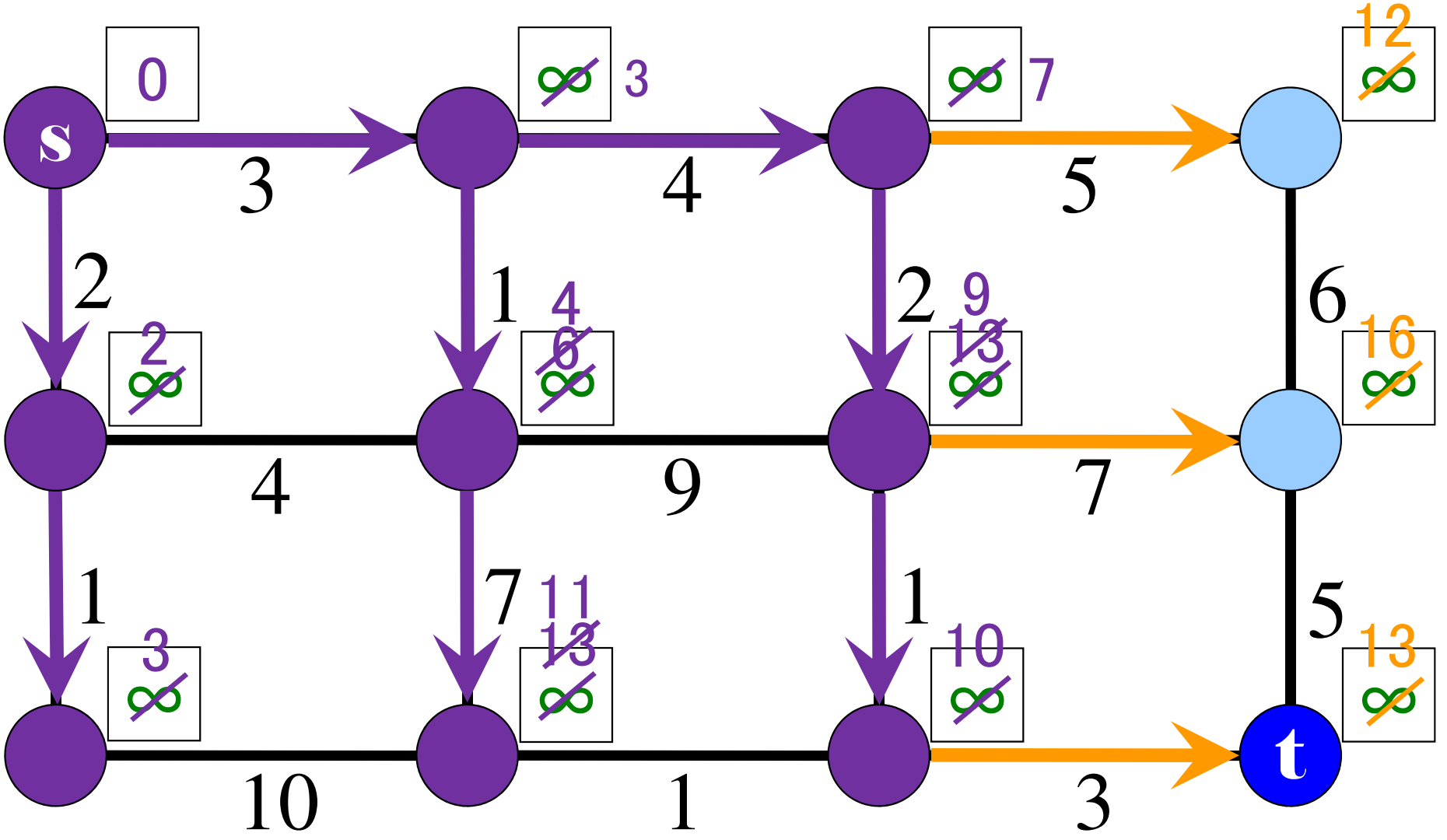


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

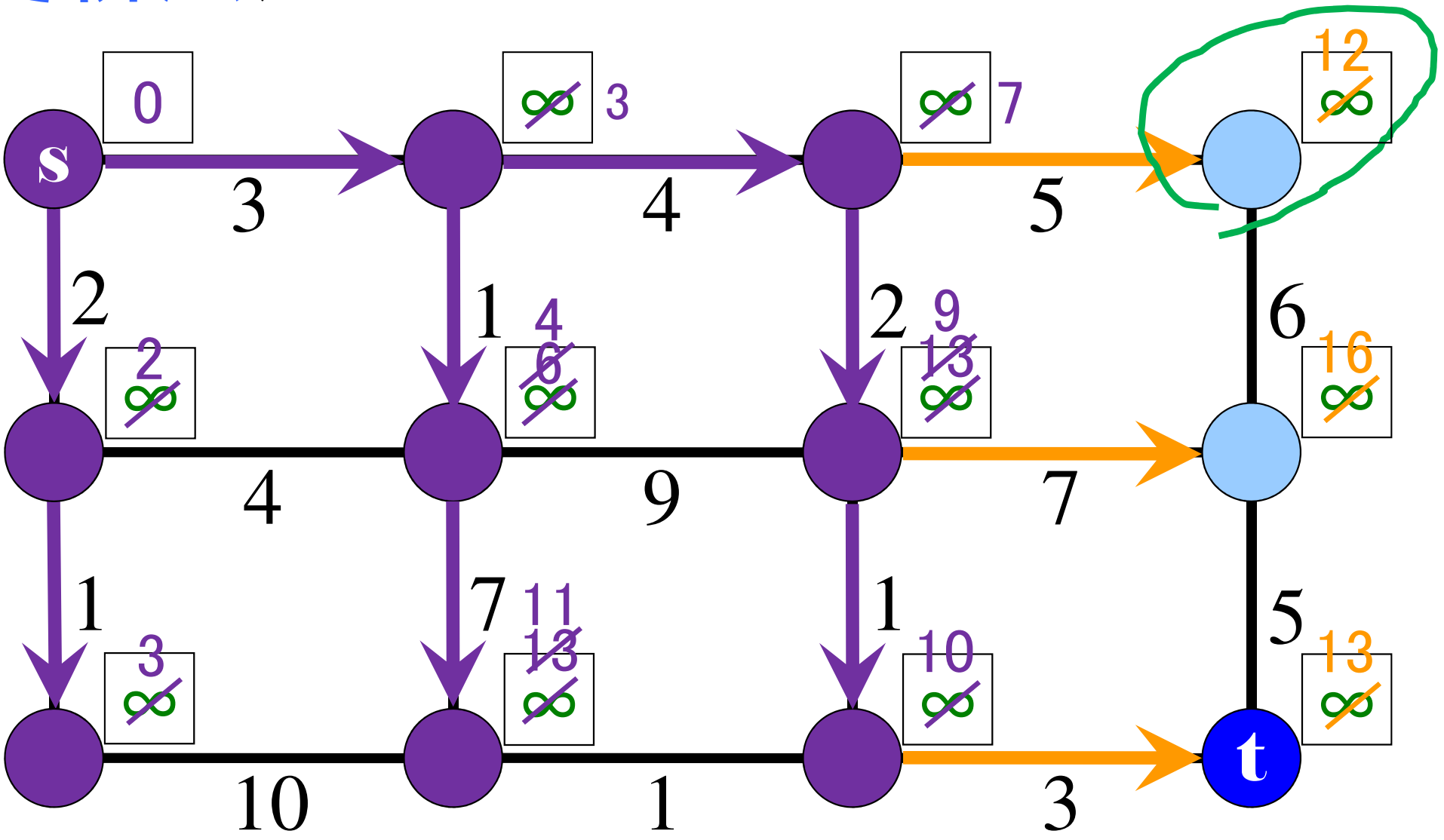
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



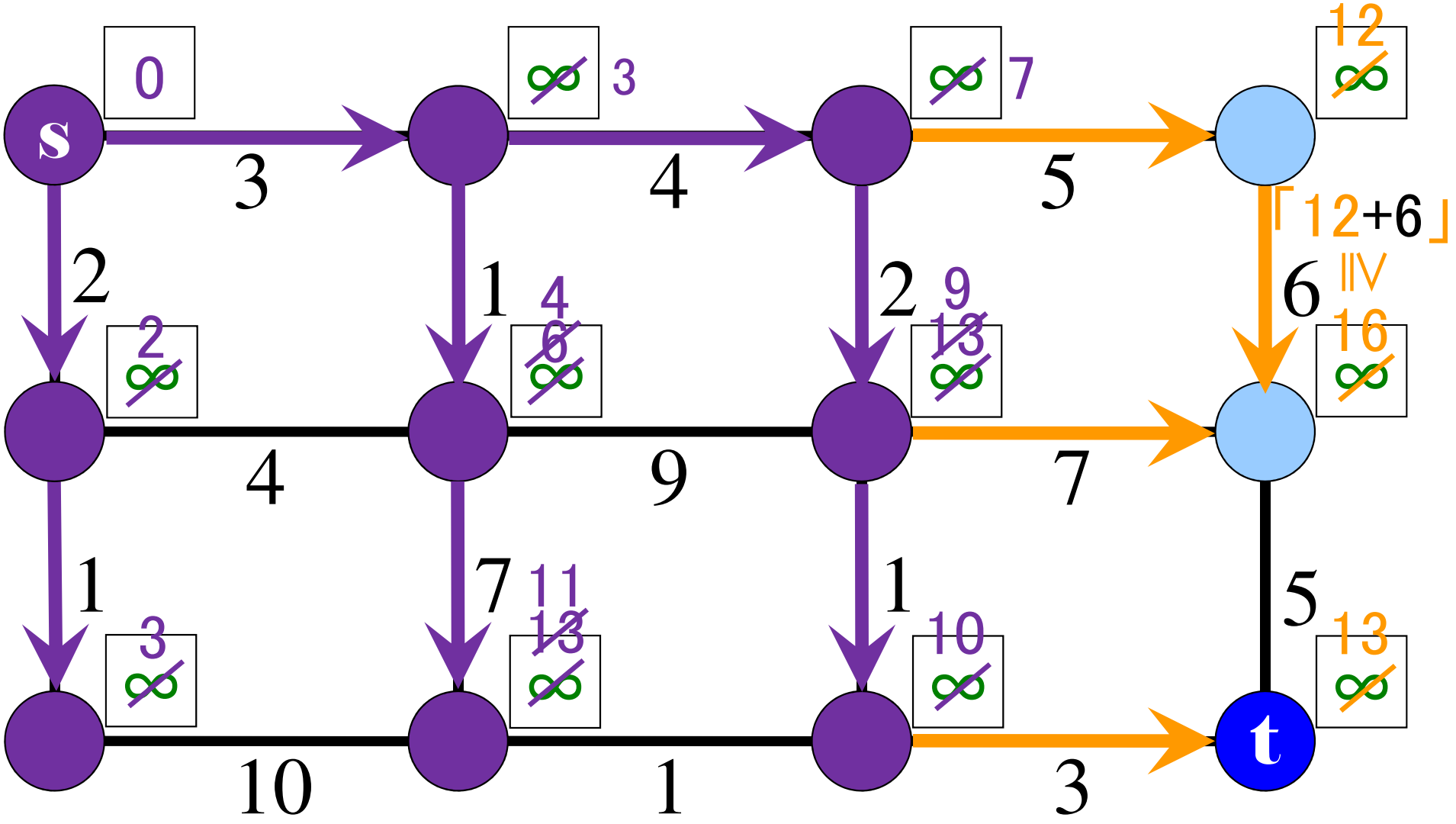
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

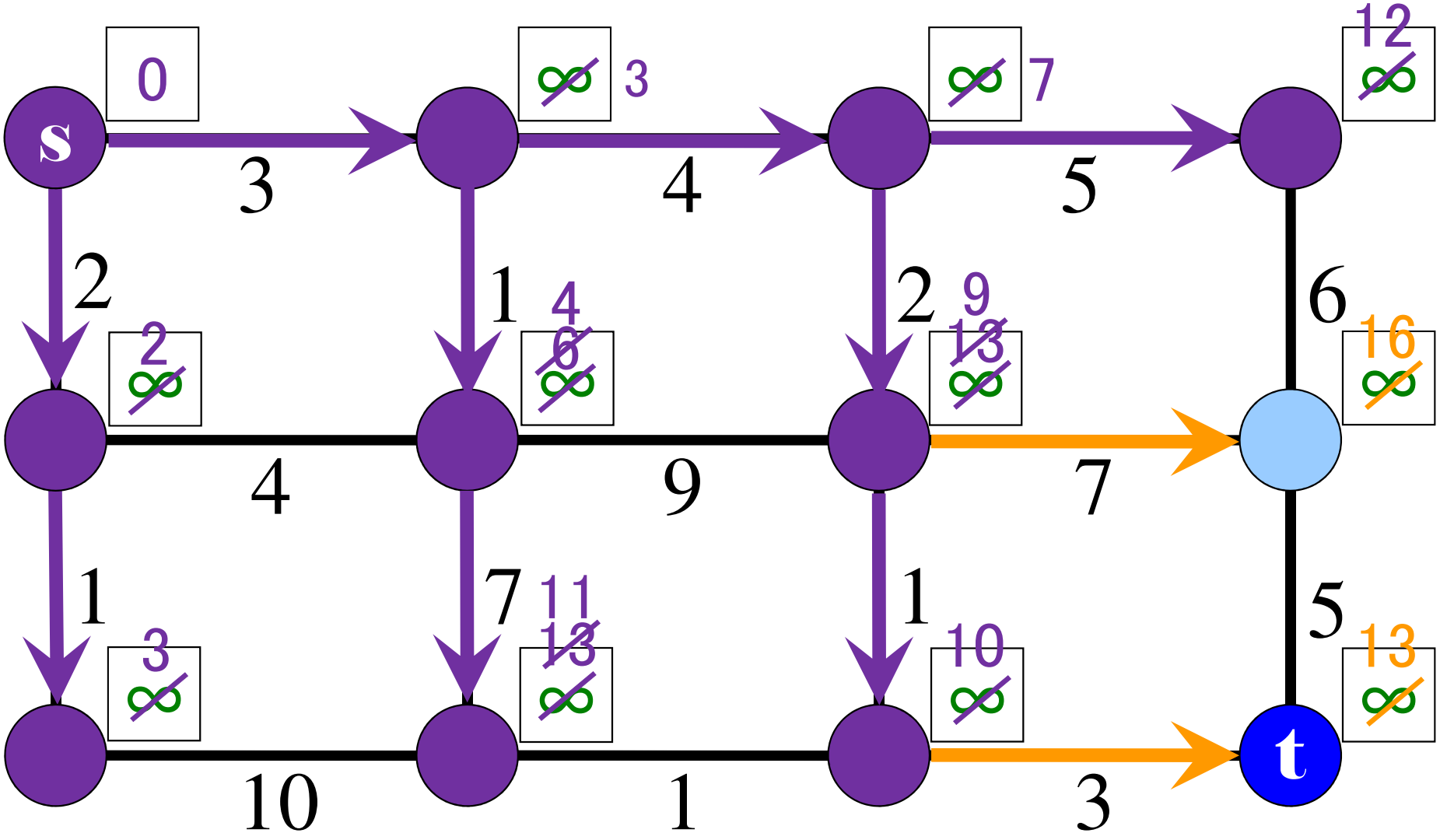


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

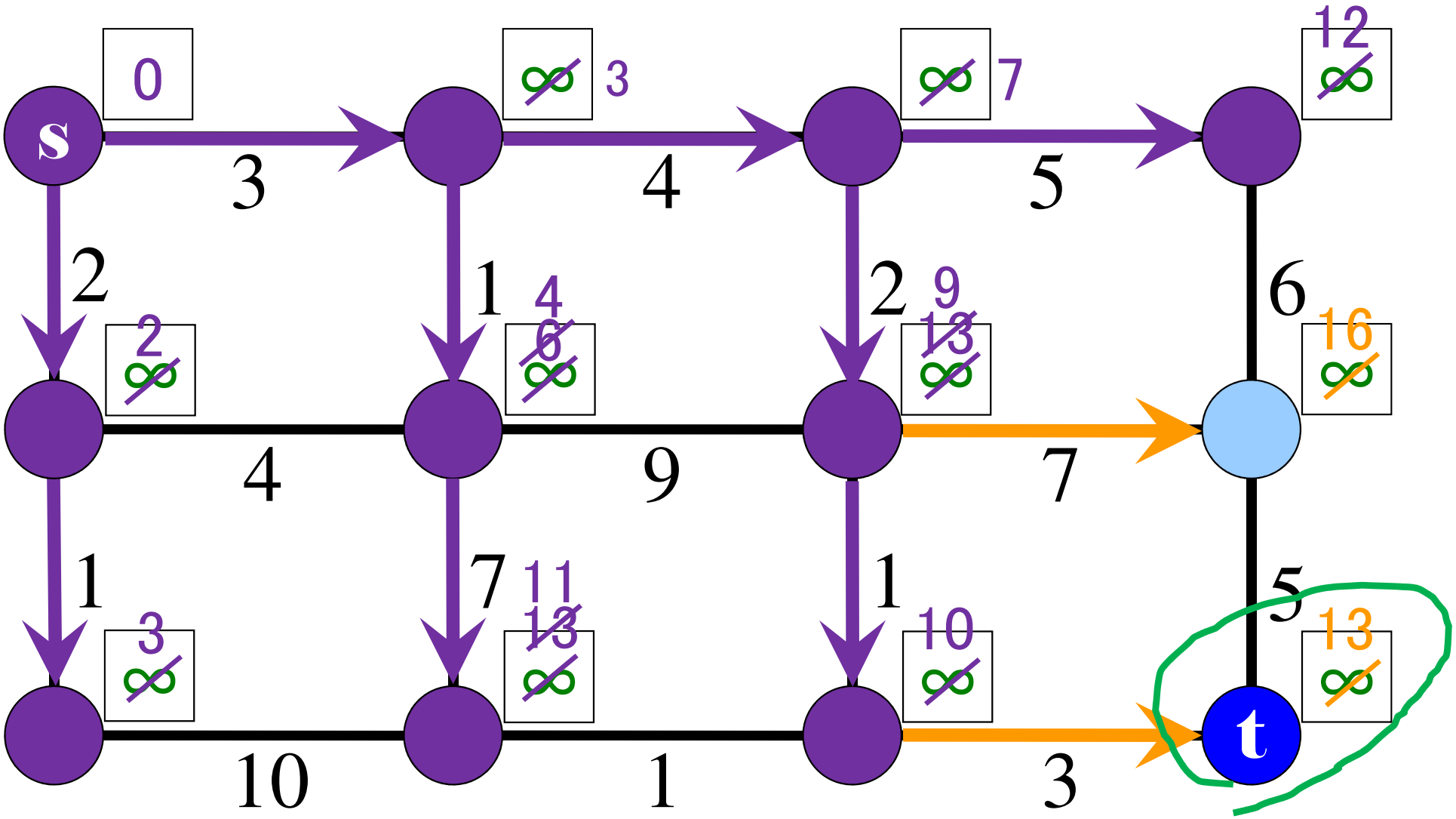
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



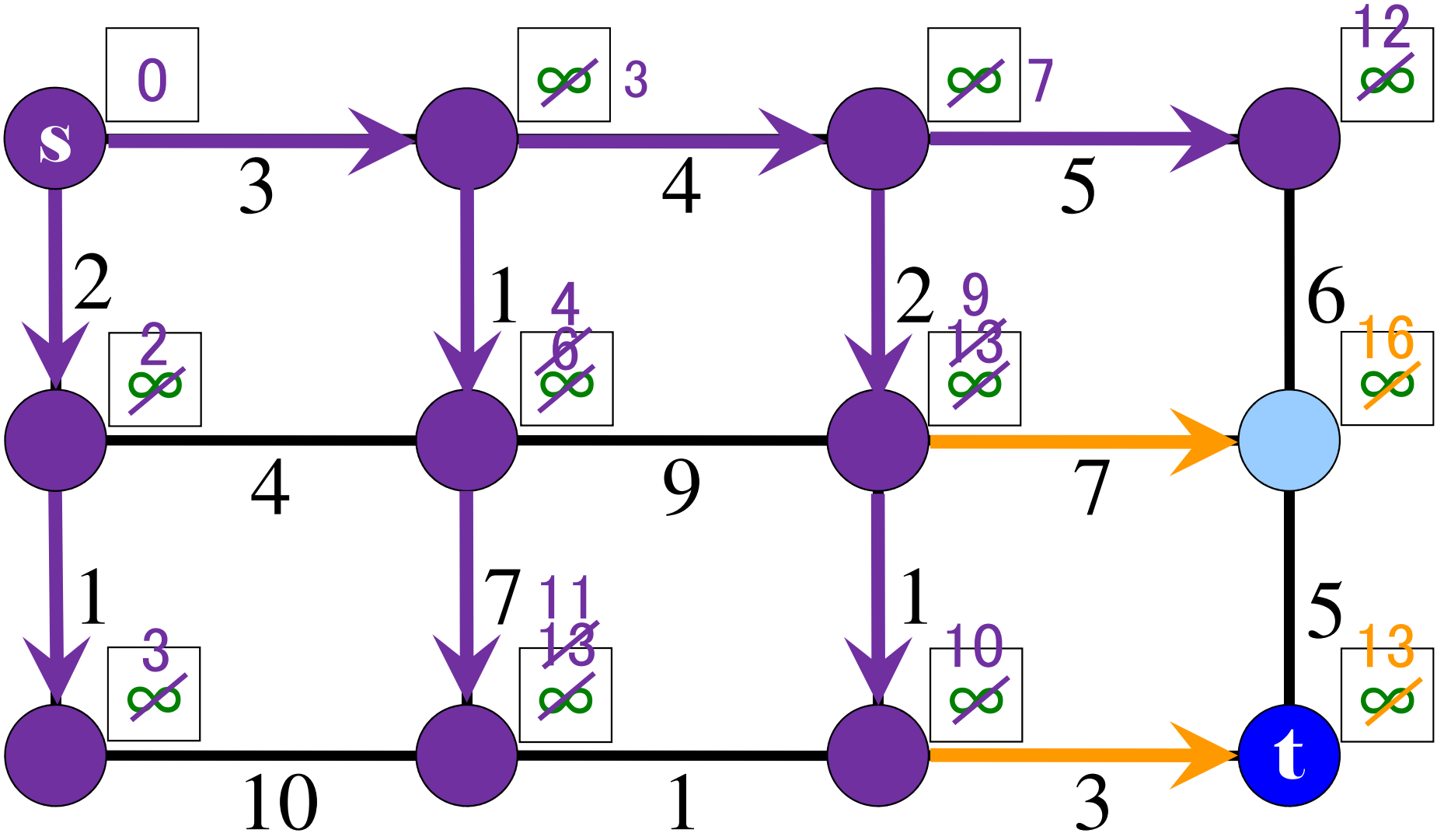
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

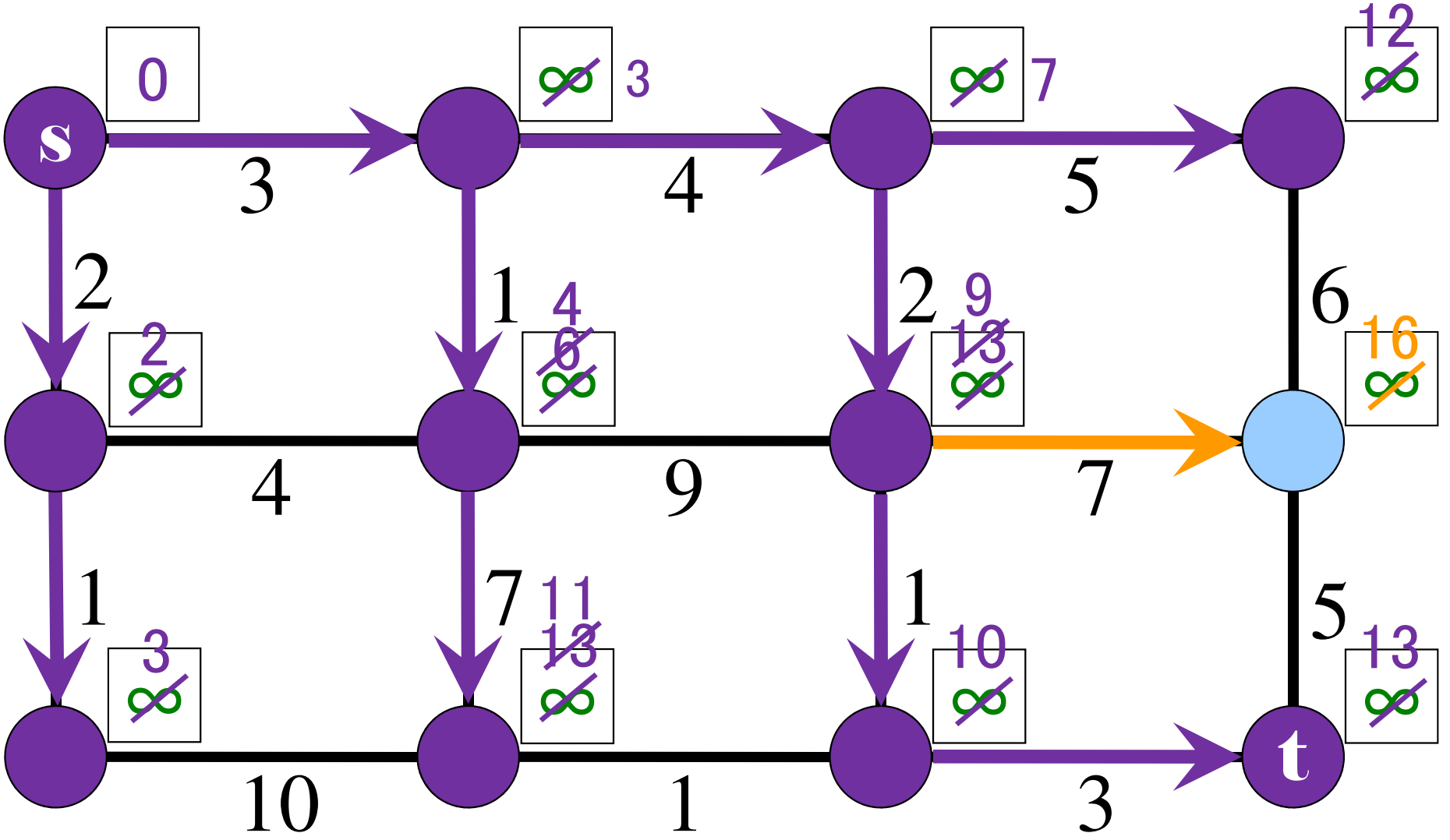


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

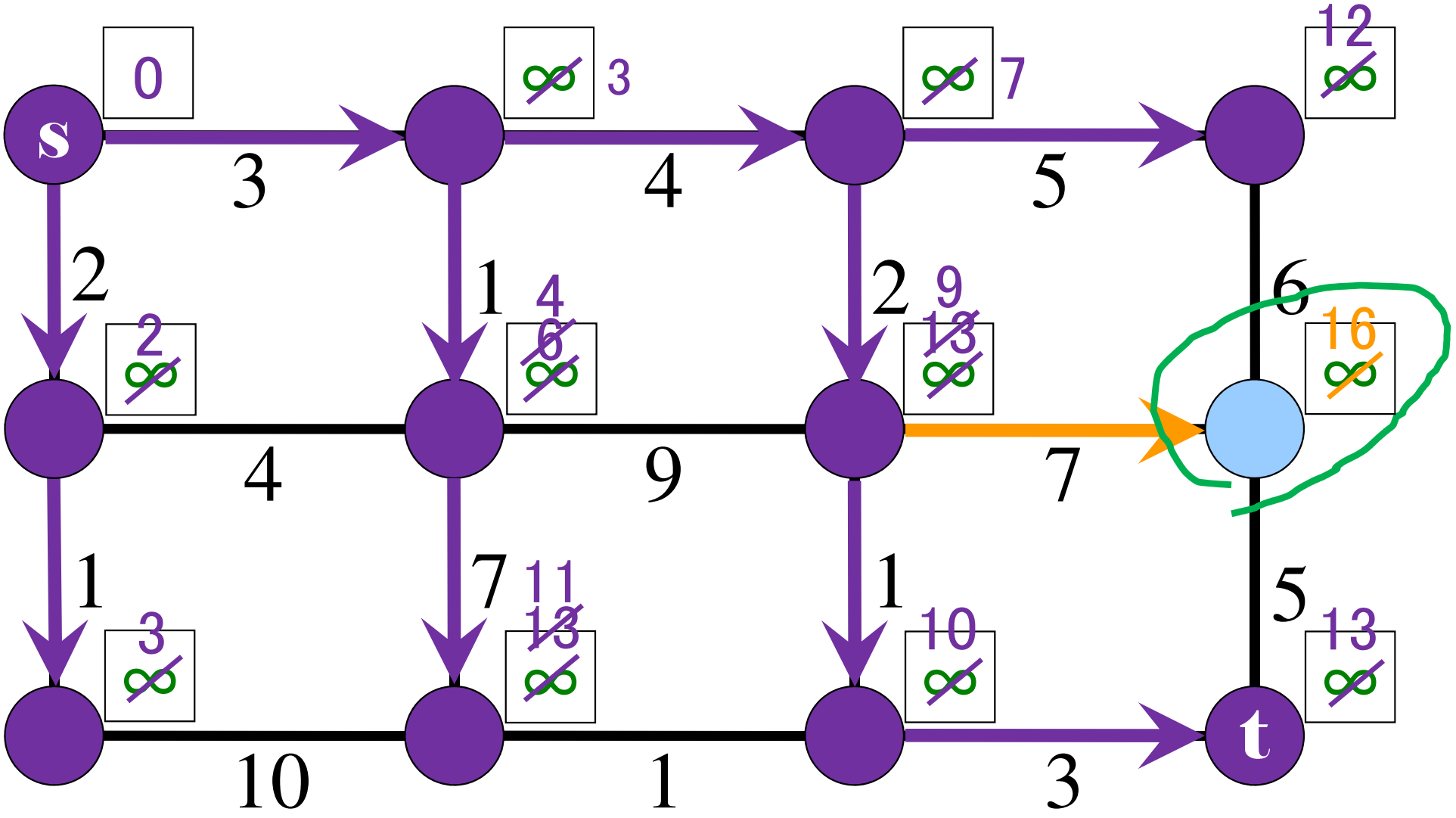
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



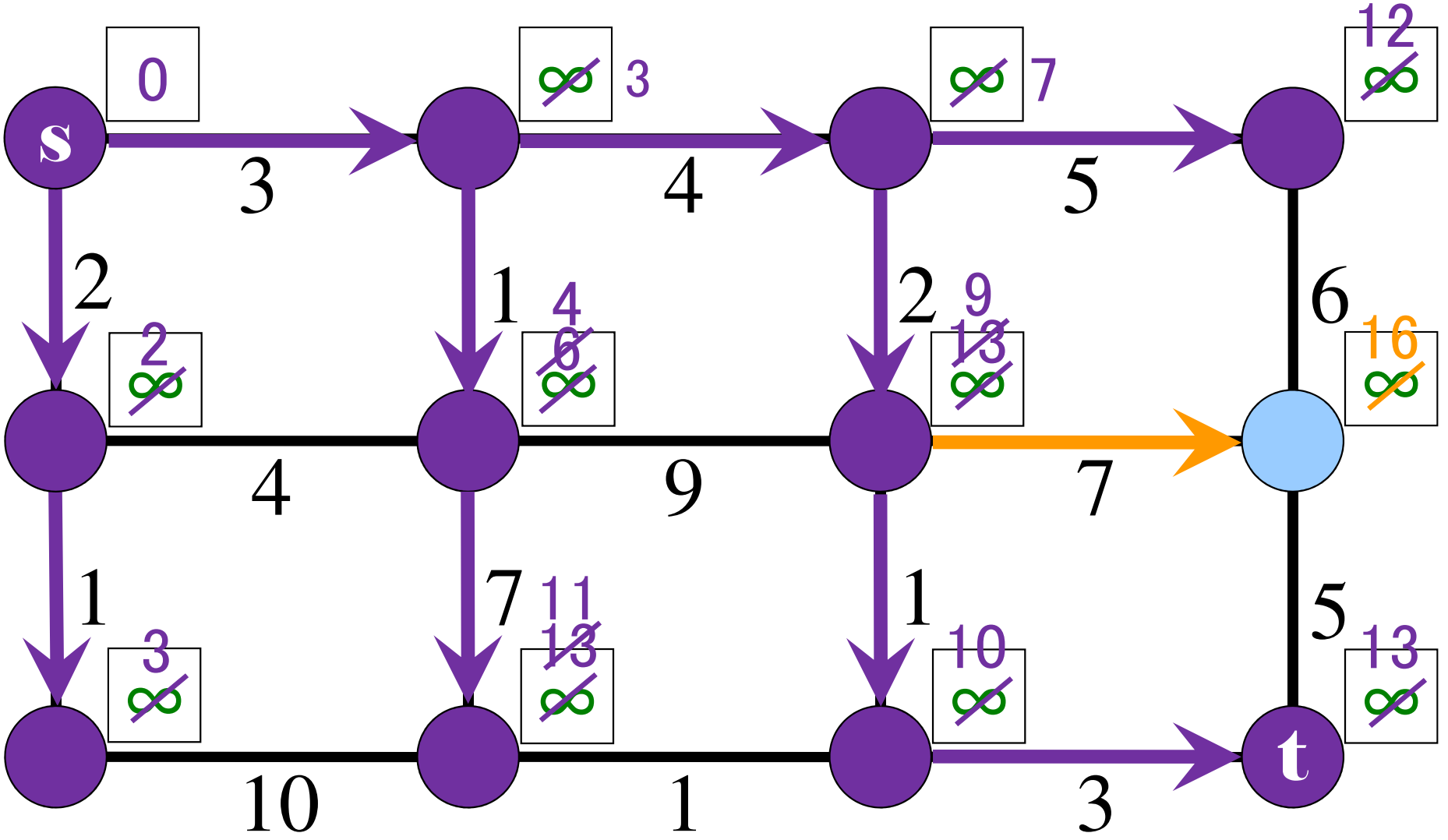
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

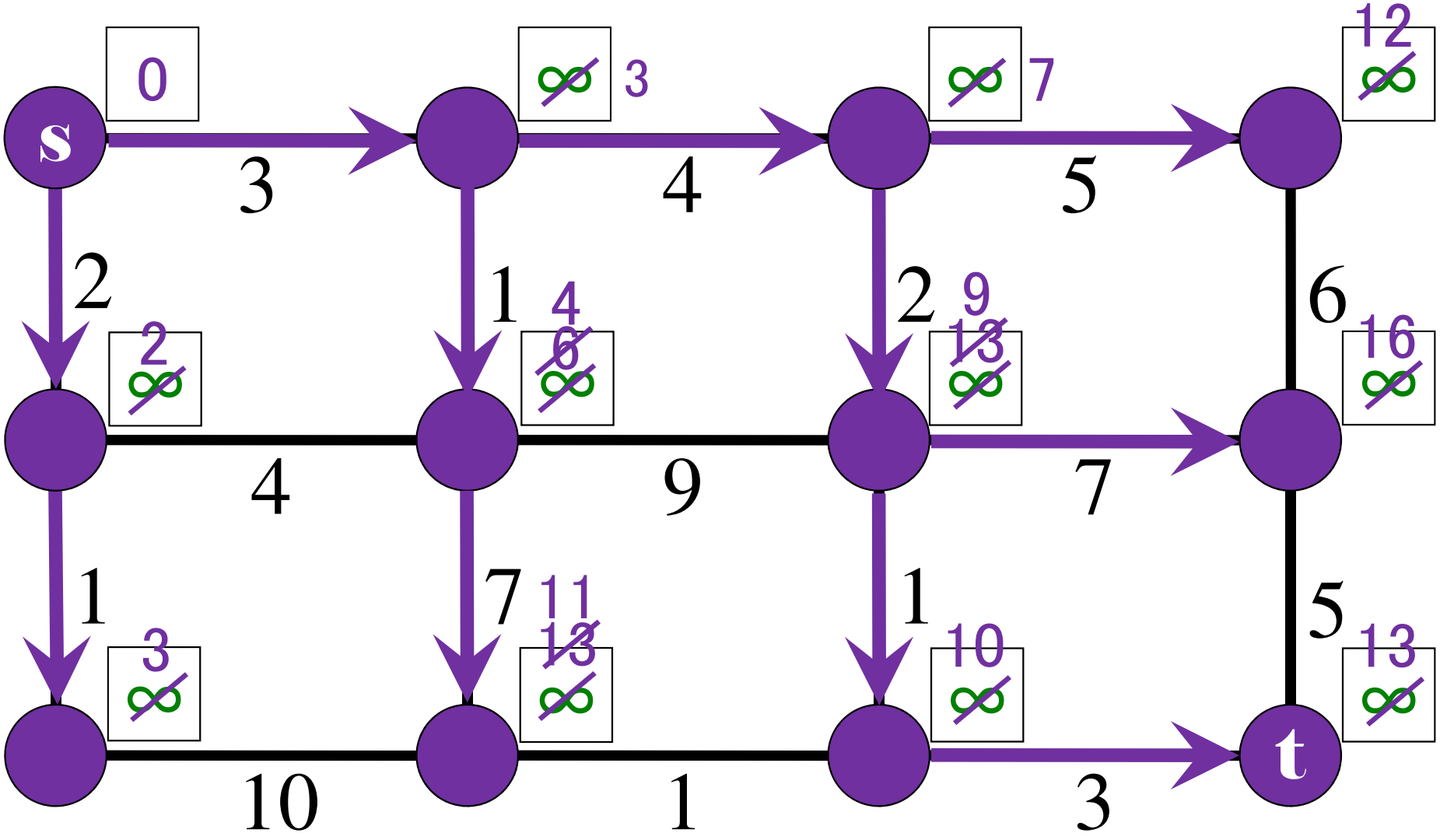


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

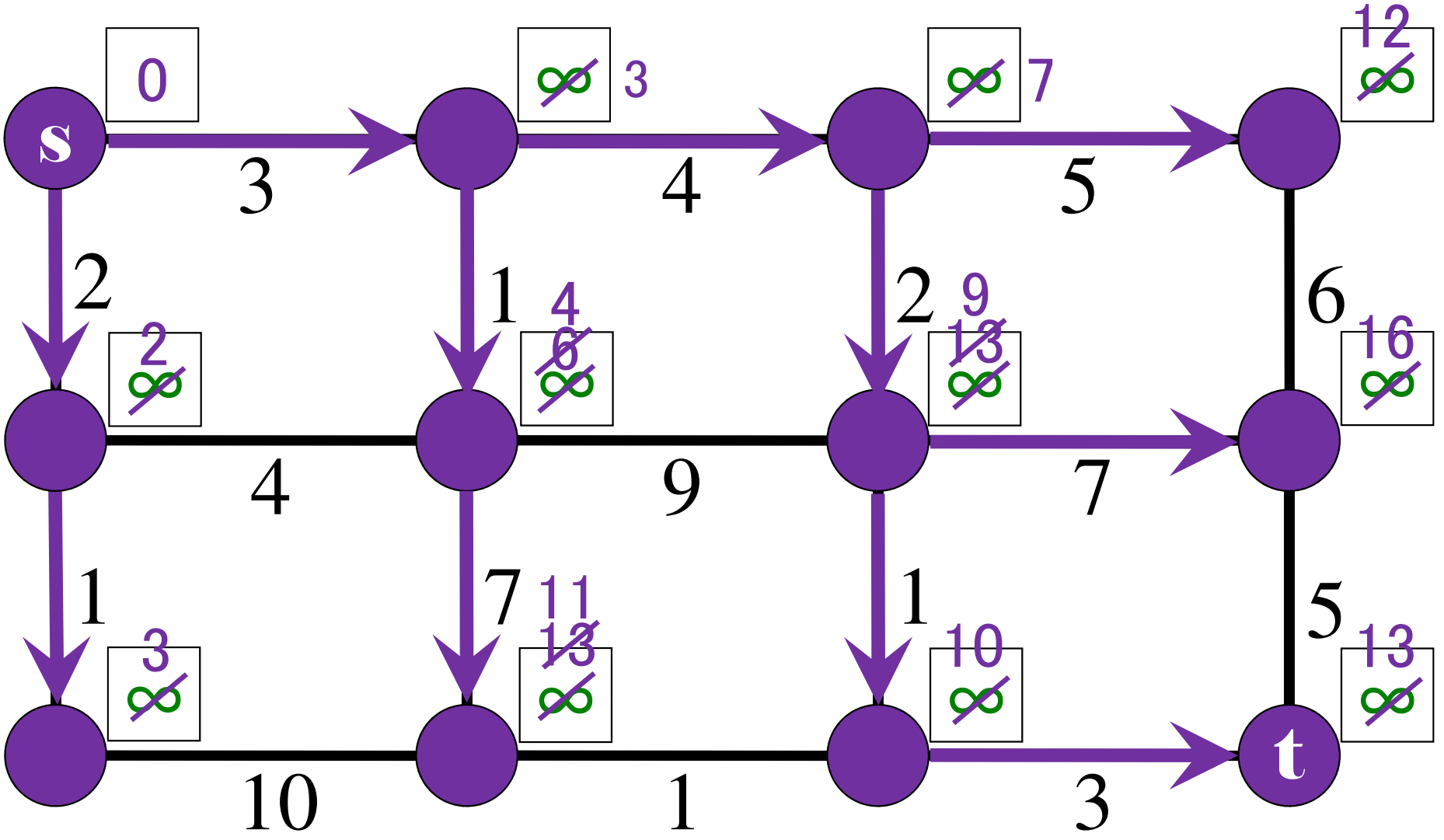
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



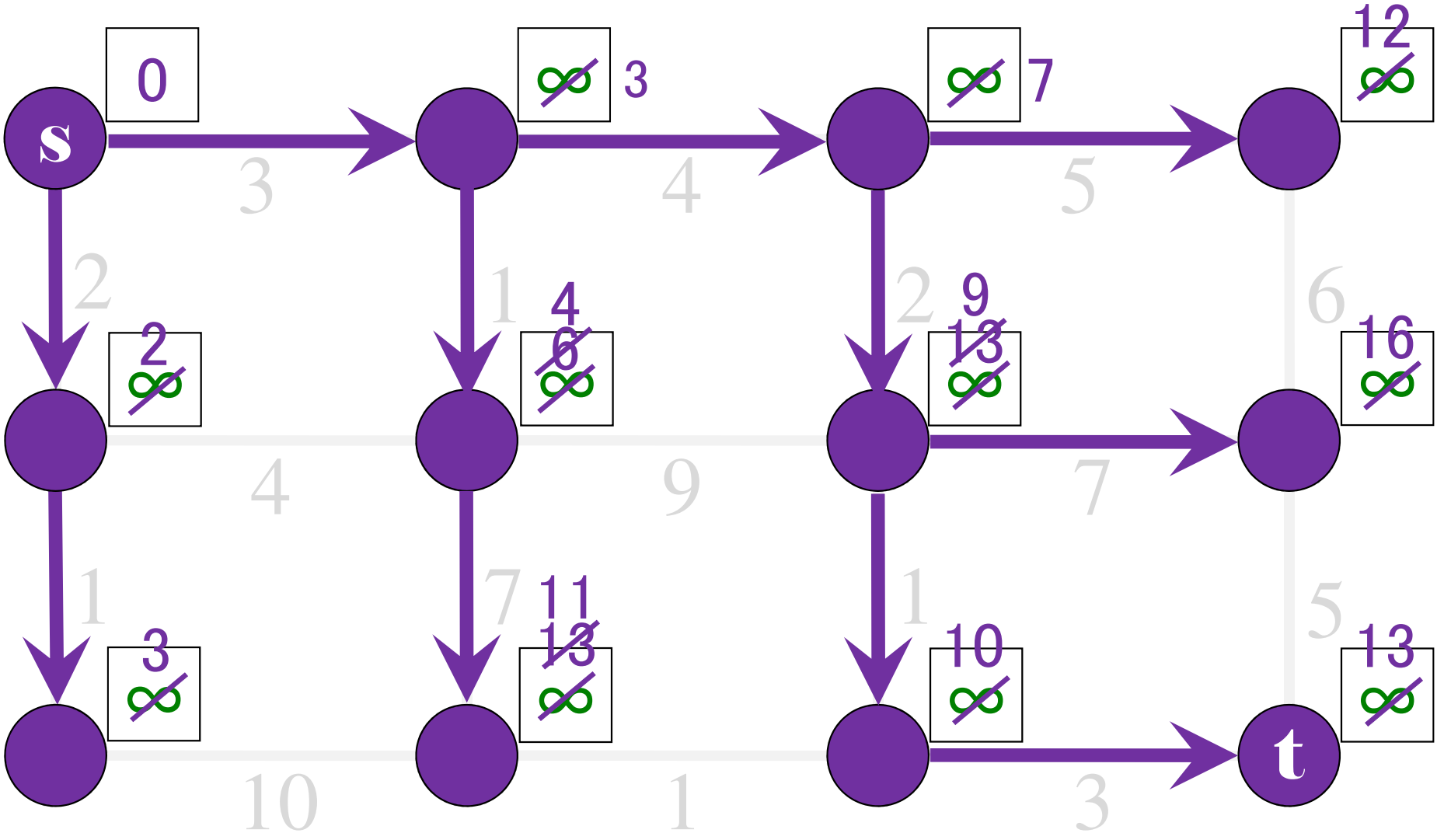
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



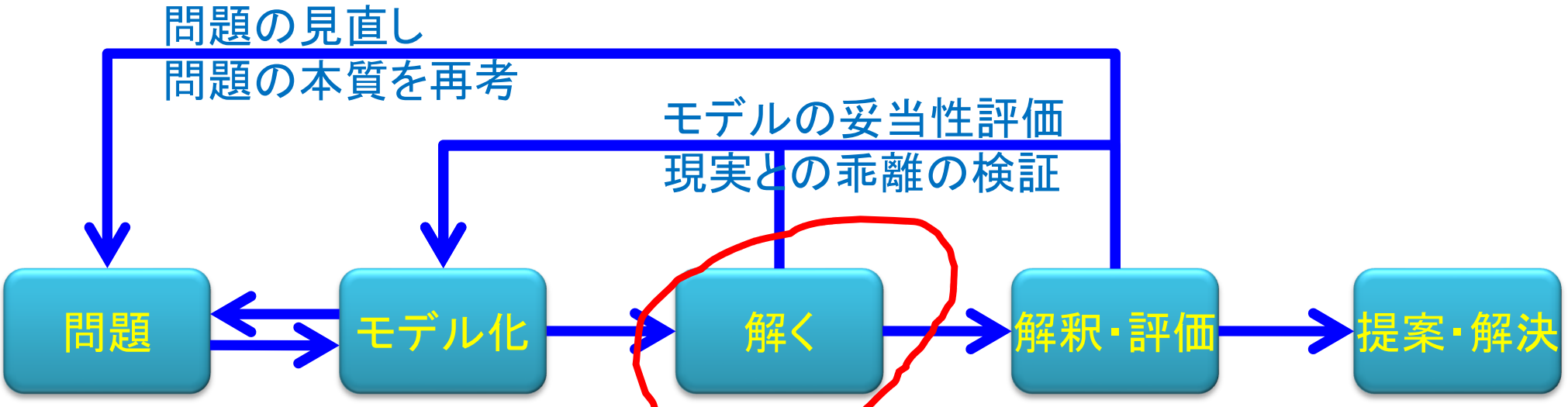
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

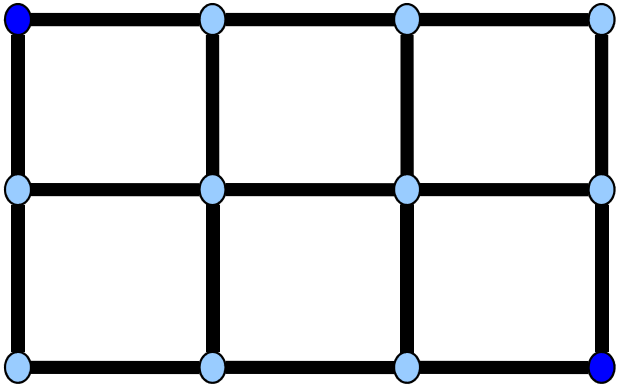
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

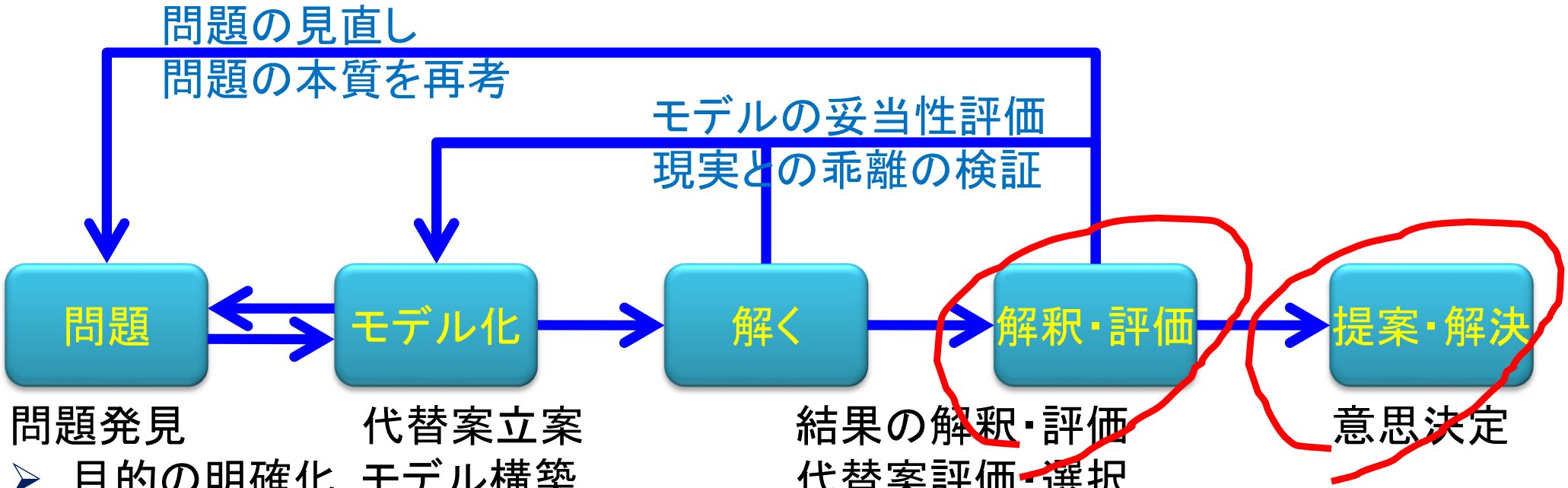
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

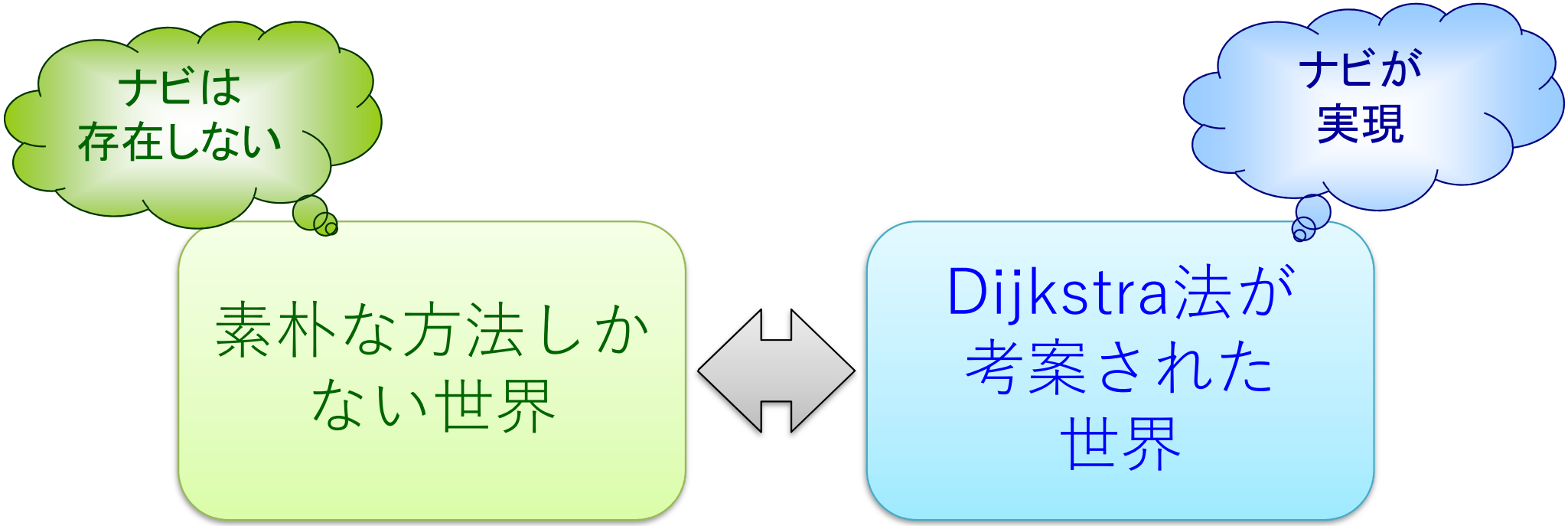
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

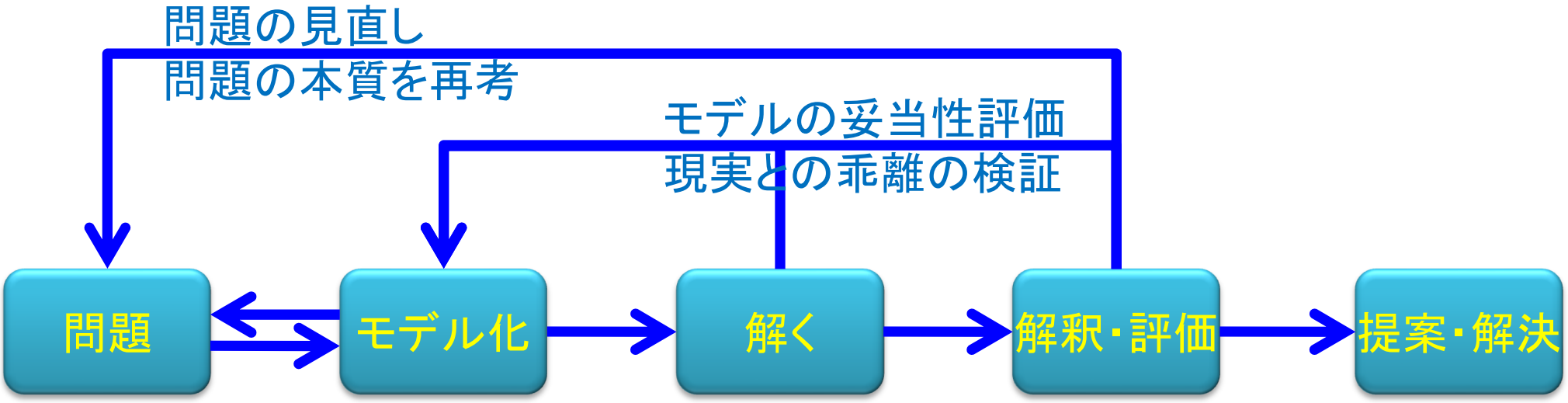
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



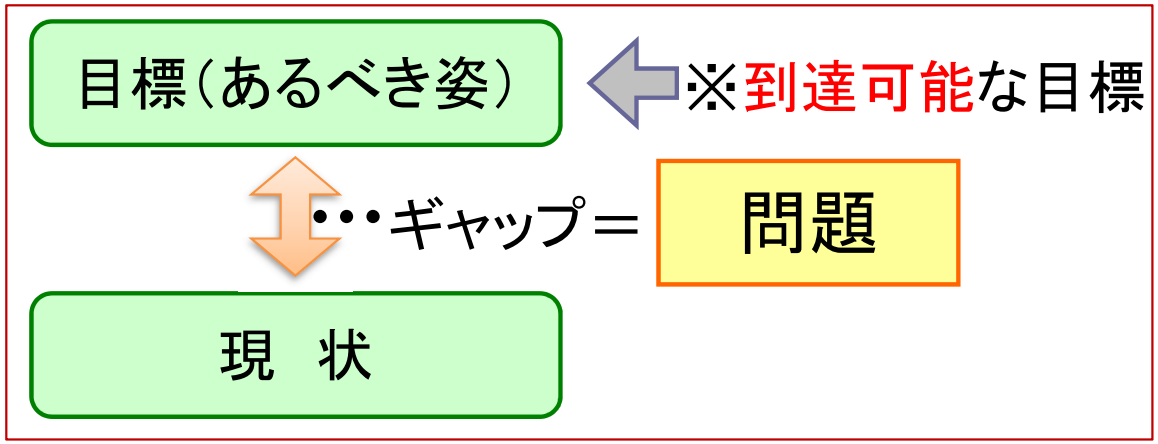
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

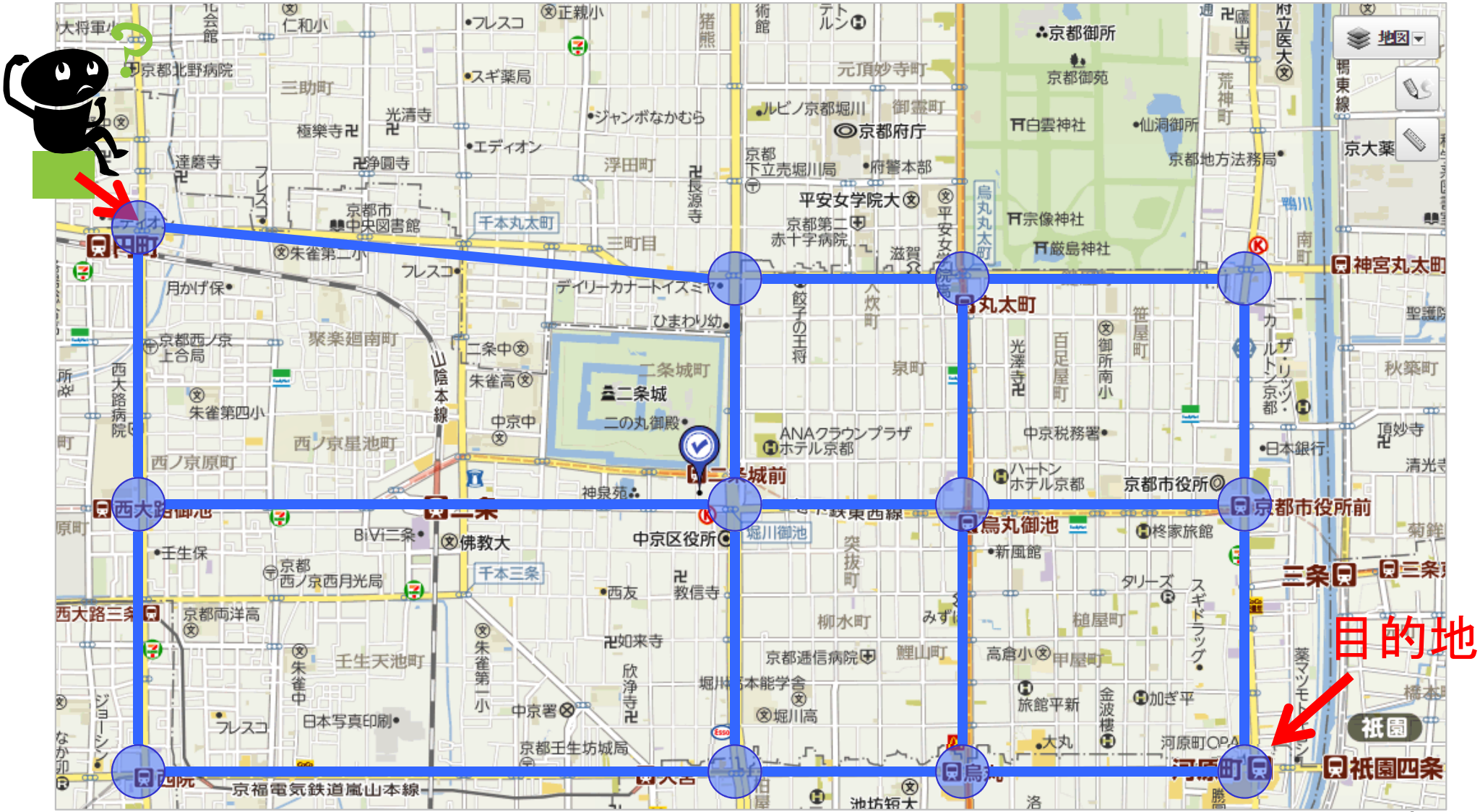
意思決定

問題の定義



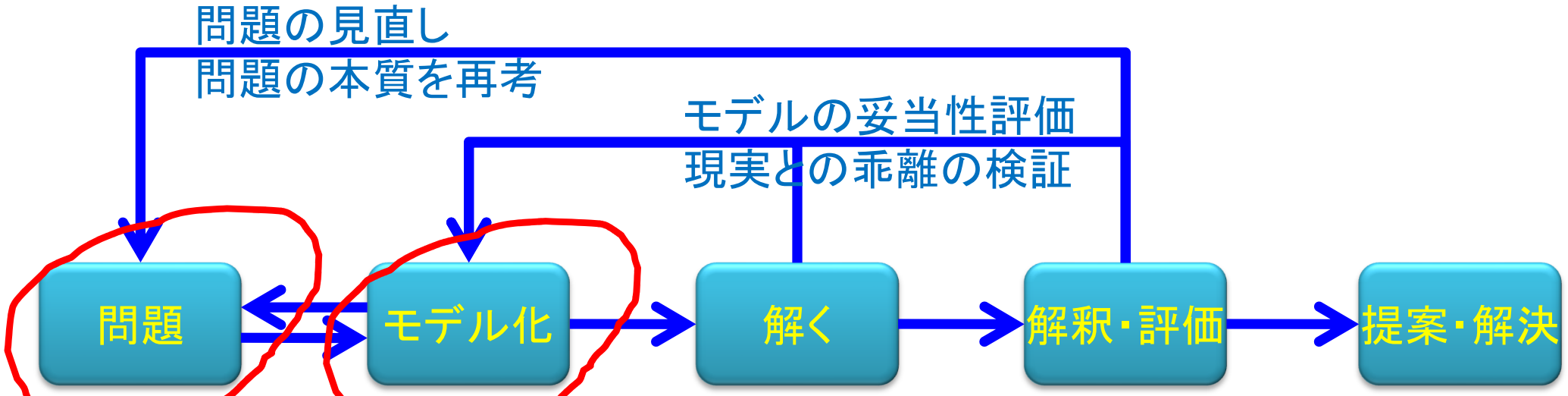
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

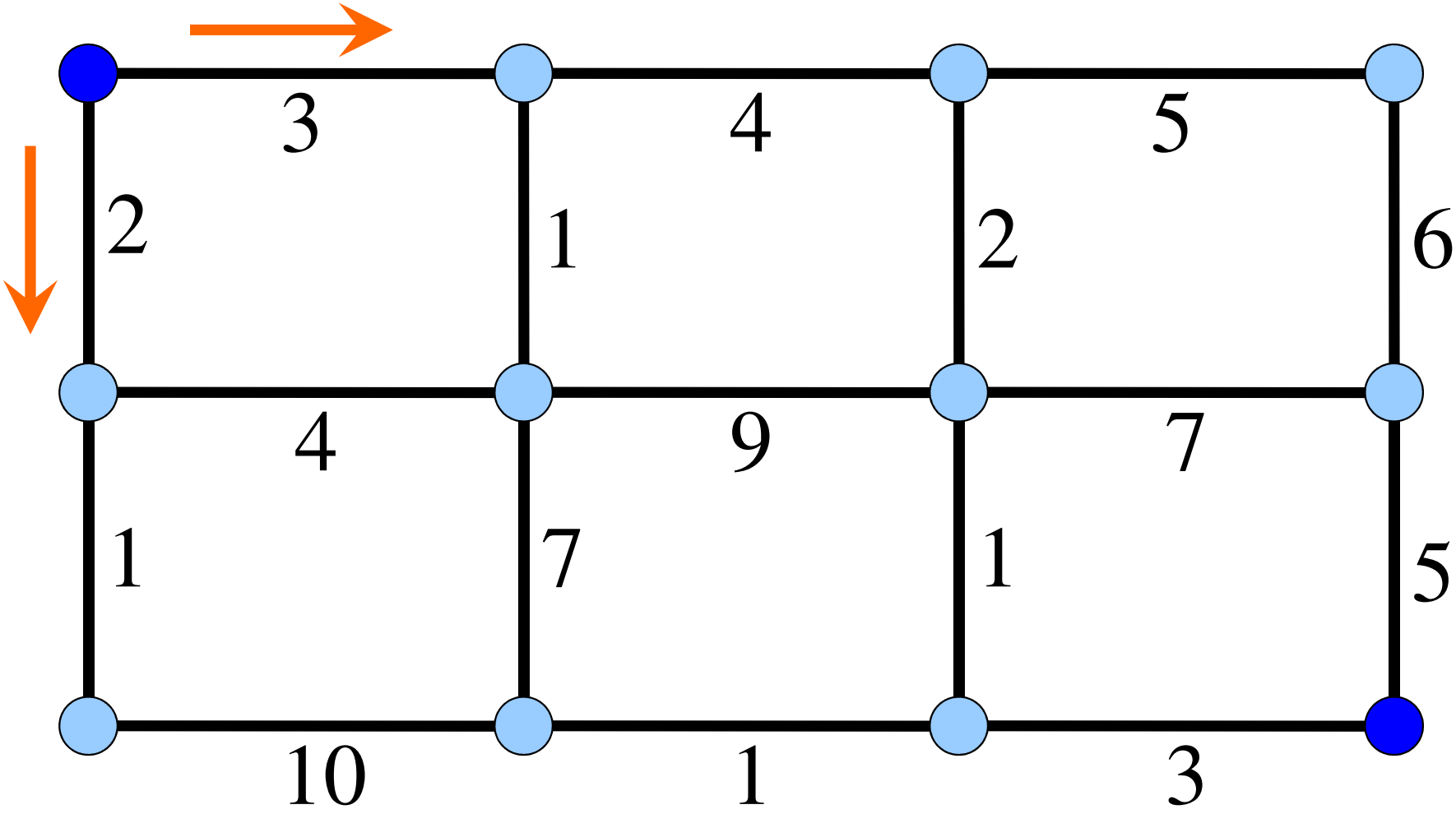
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

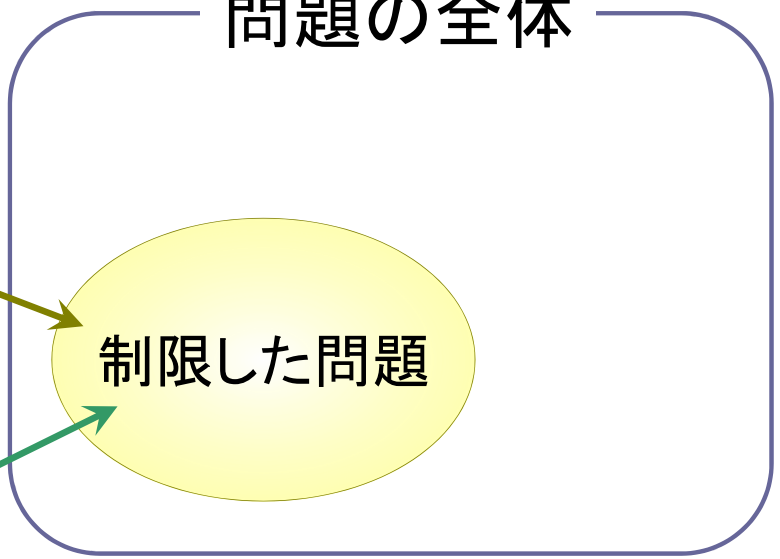


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

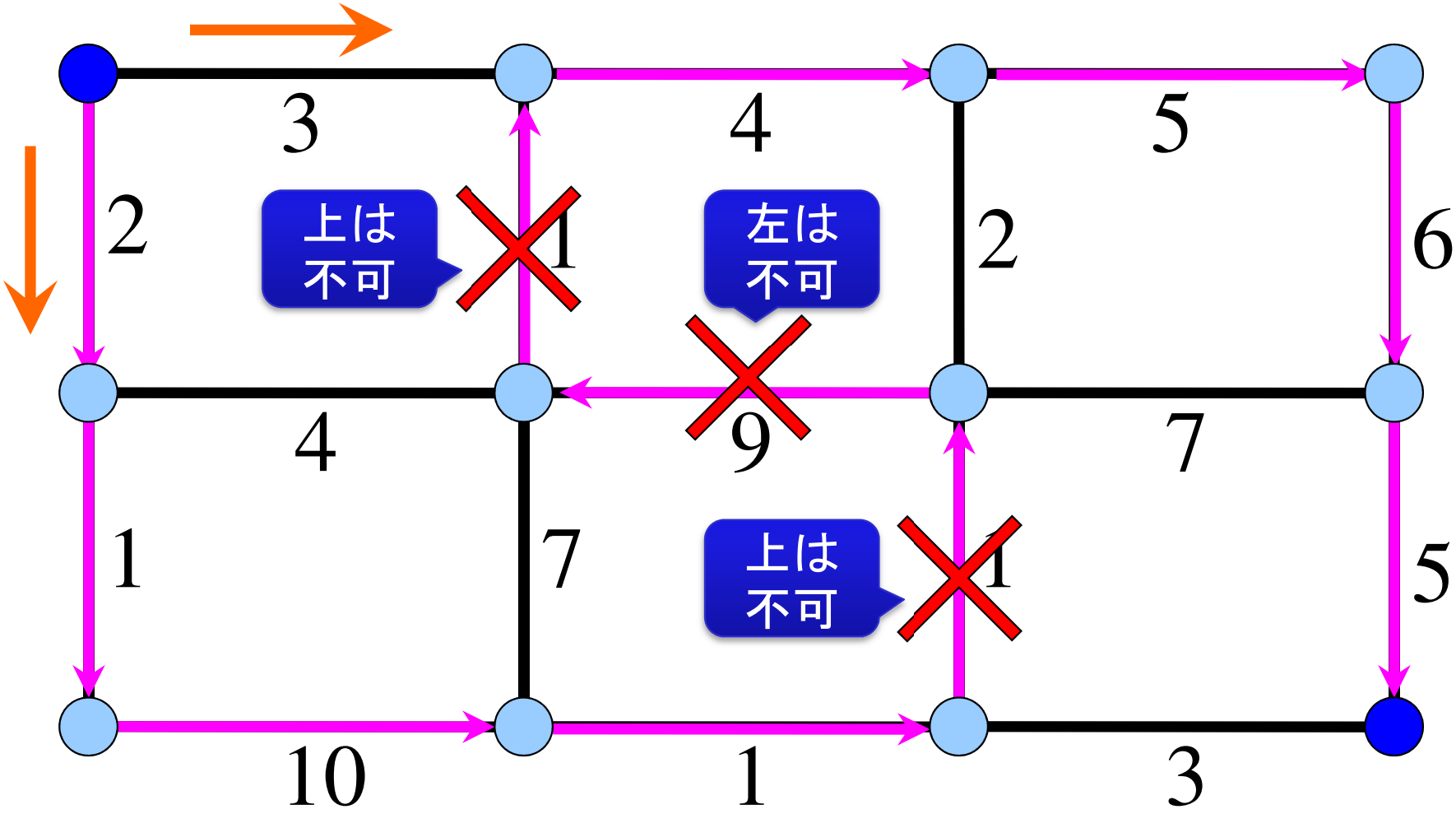
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

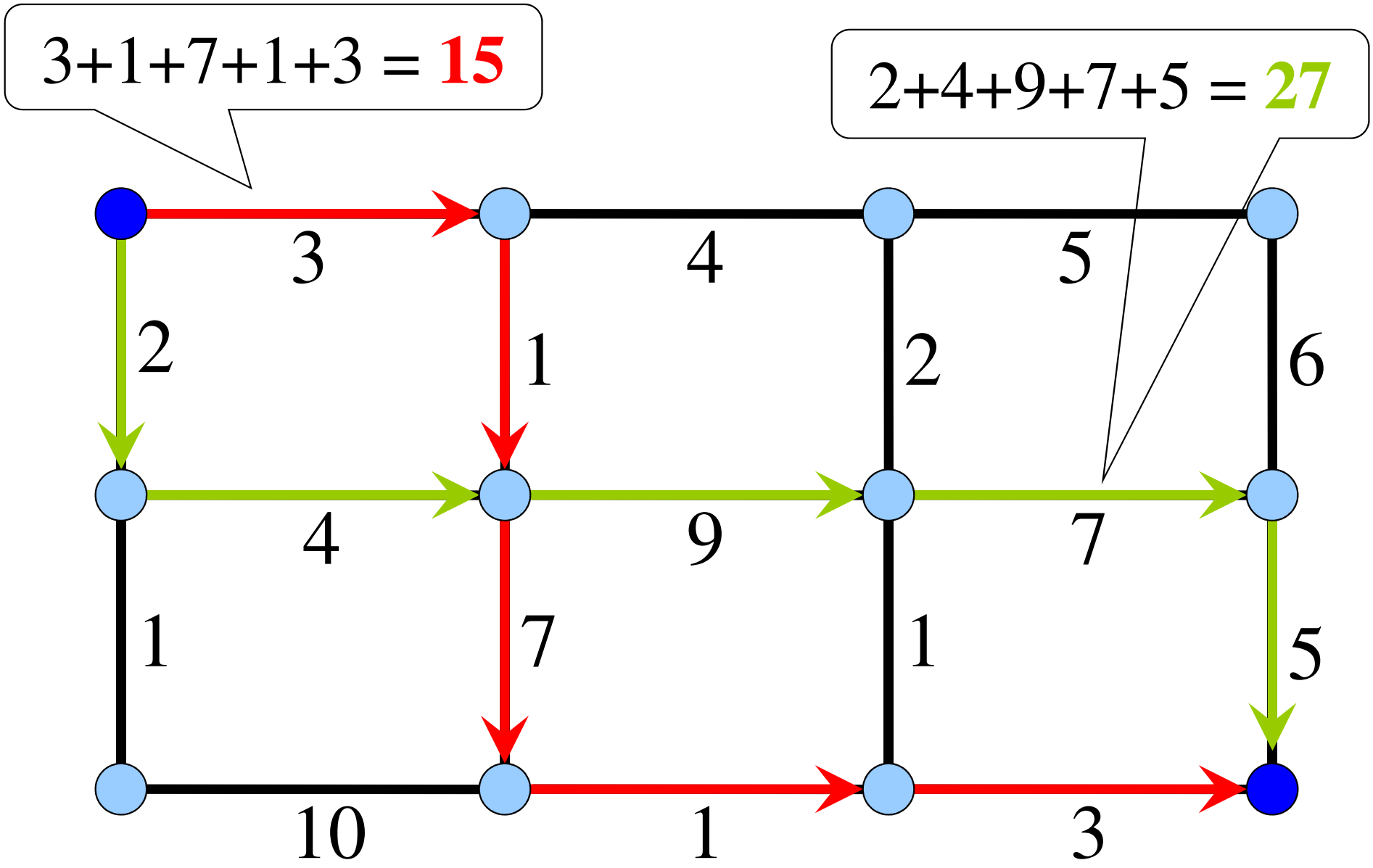
難しいなら易しくすればいいのさ！

制限した問題

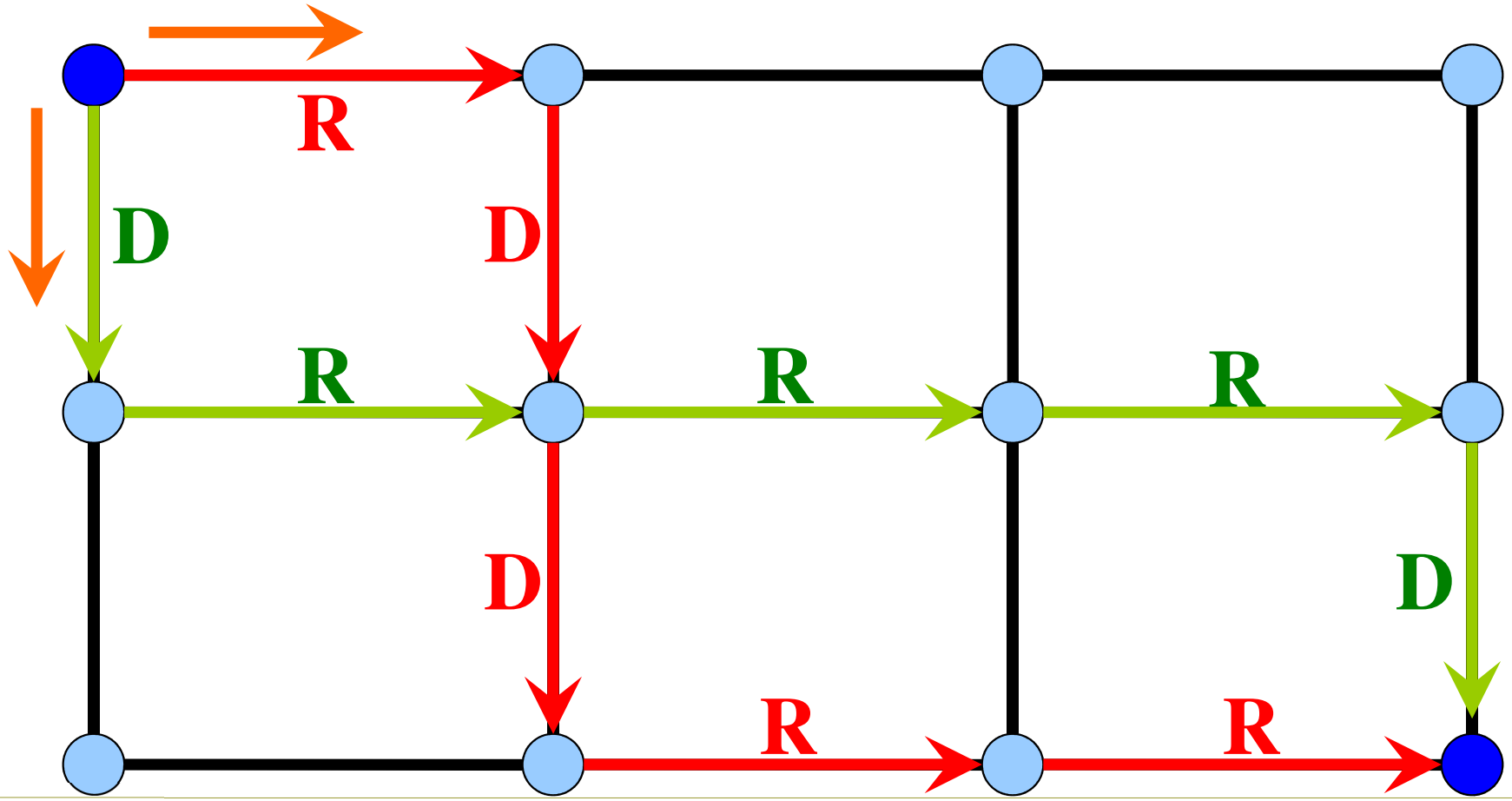
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



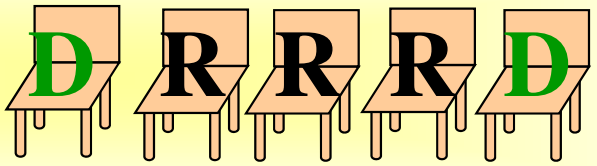
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



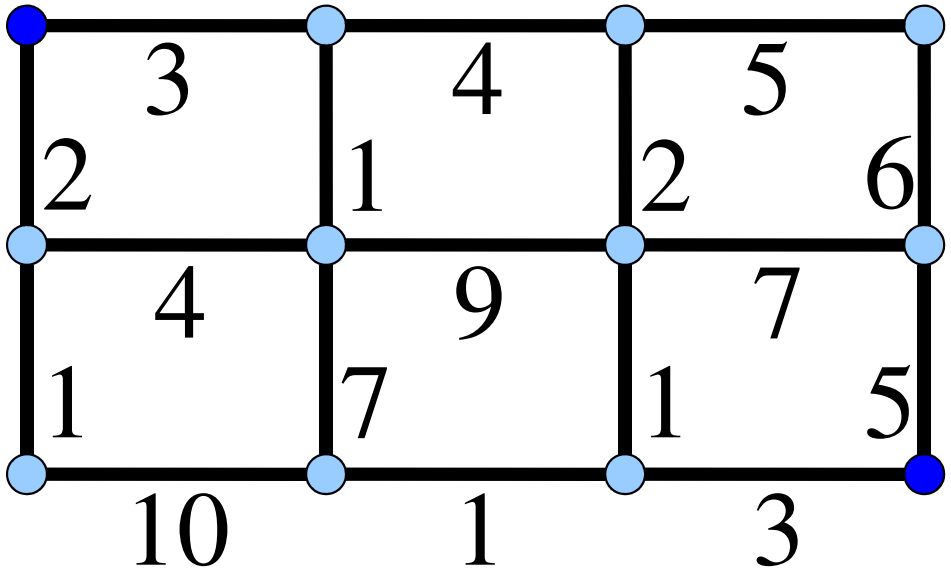
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

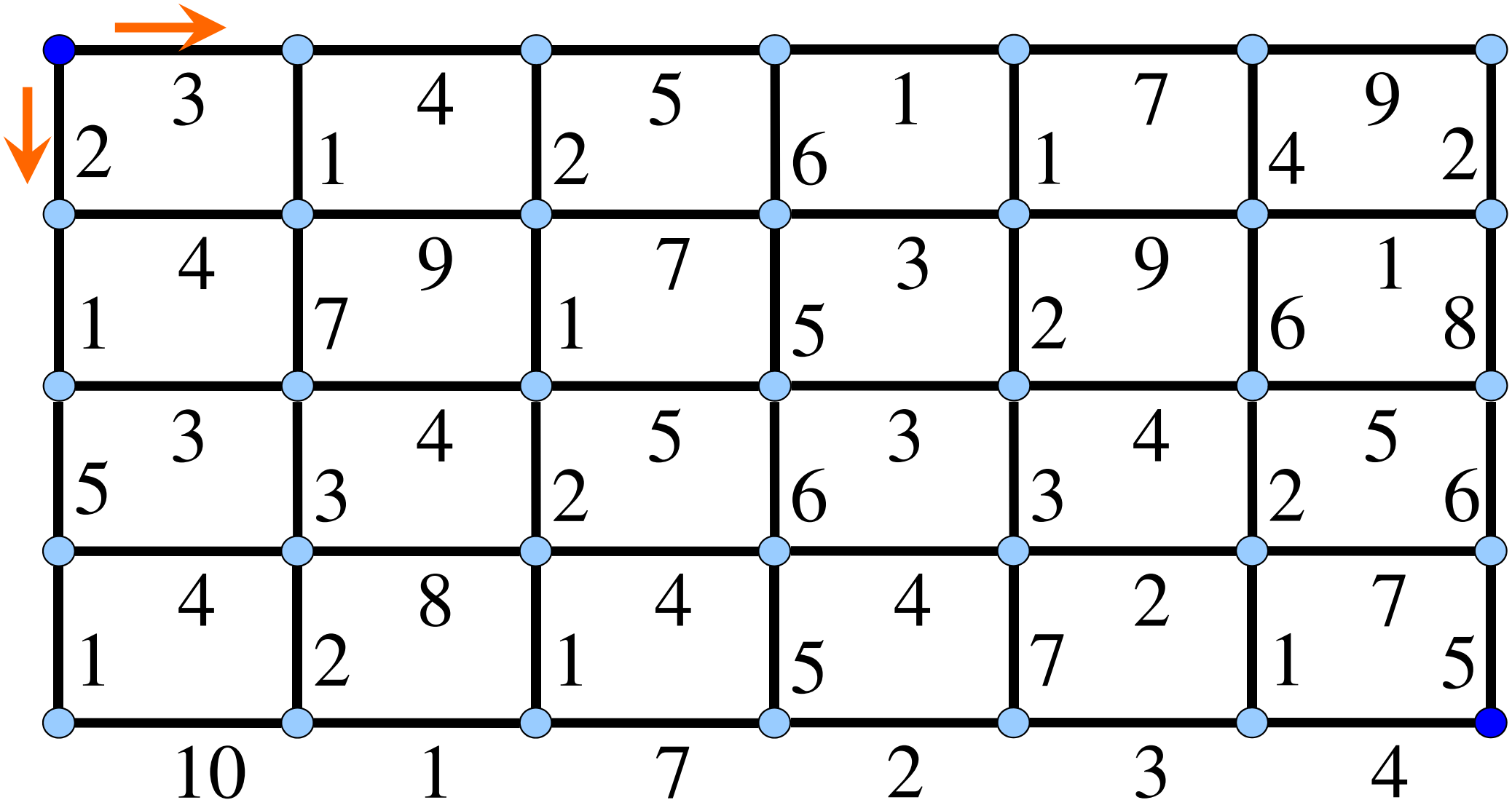
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

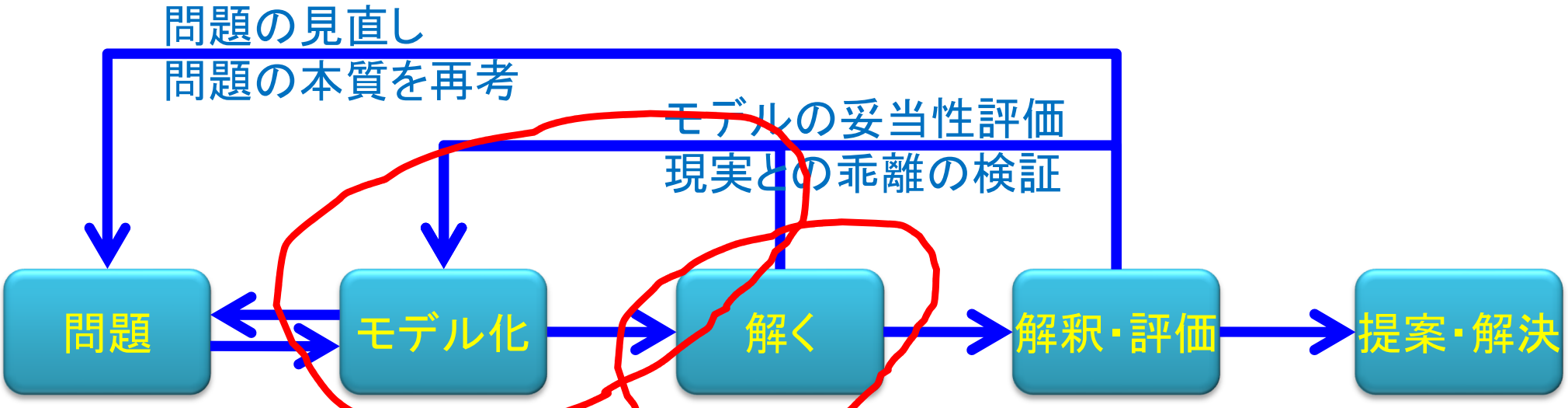
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

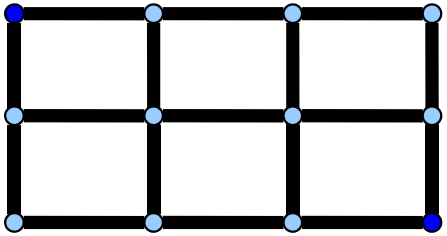
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

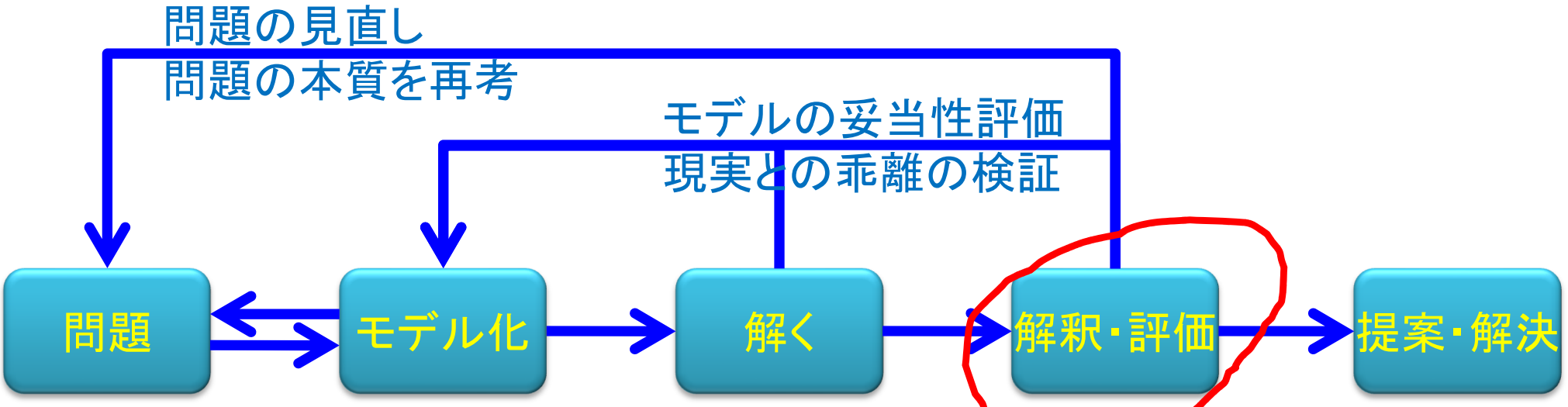
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

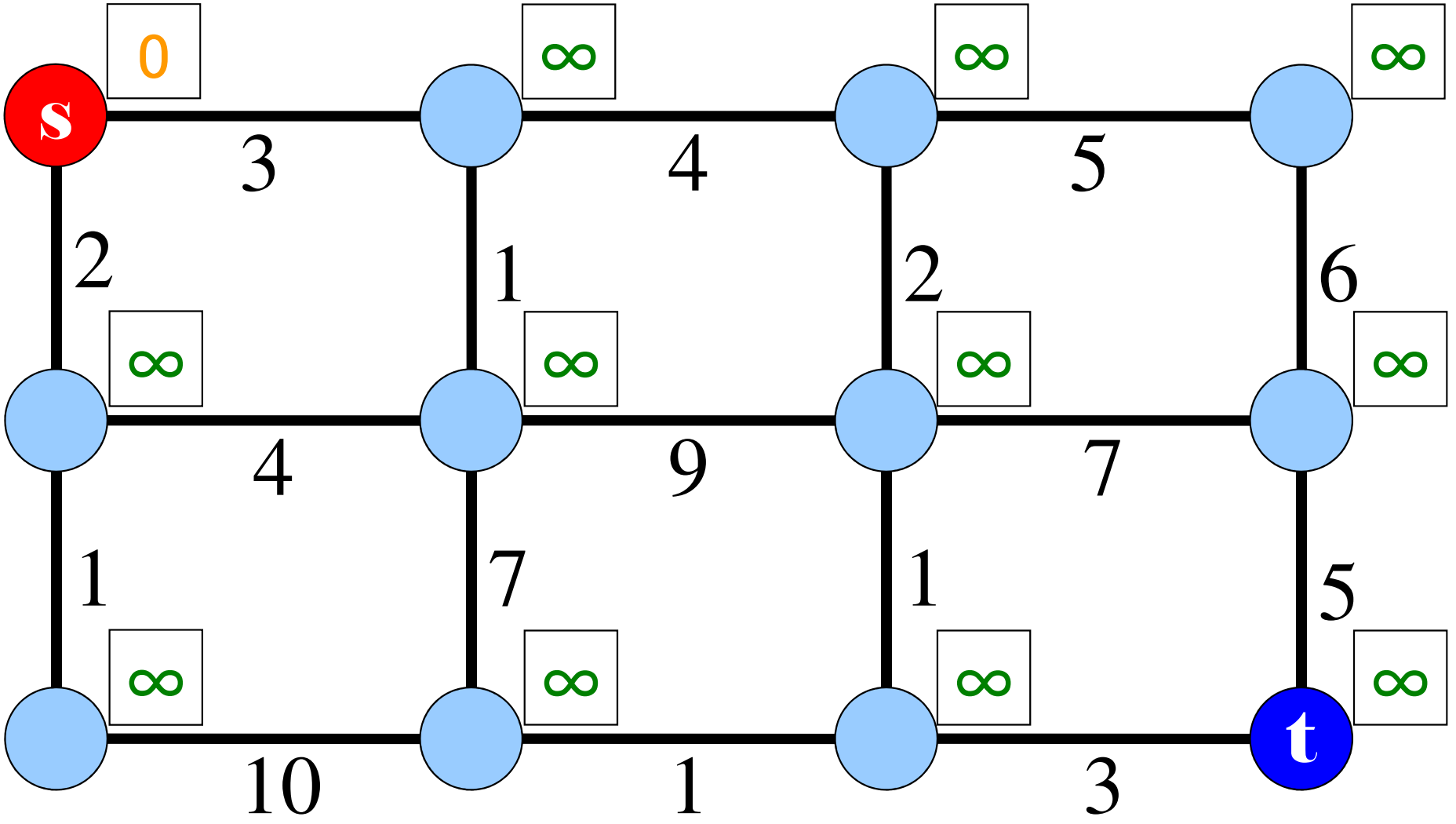
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

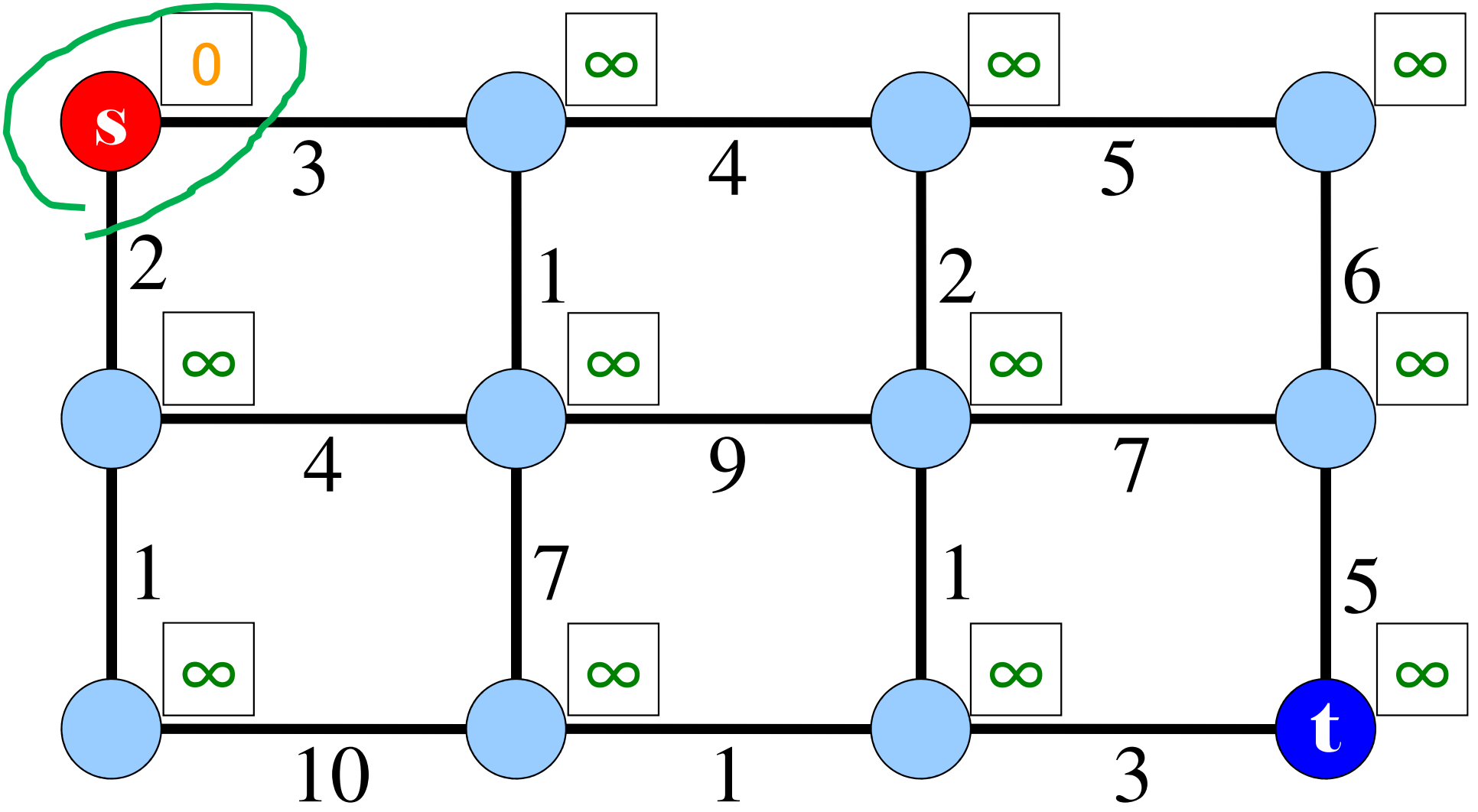
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



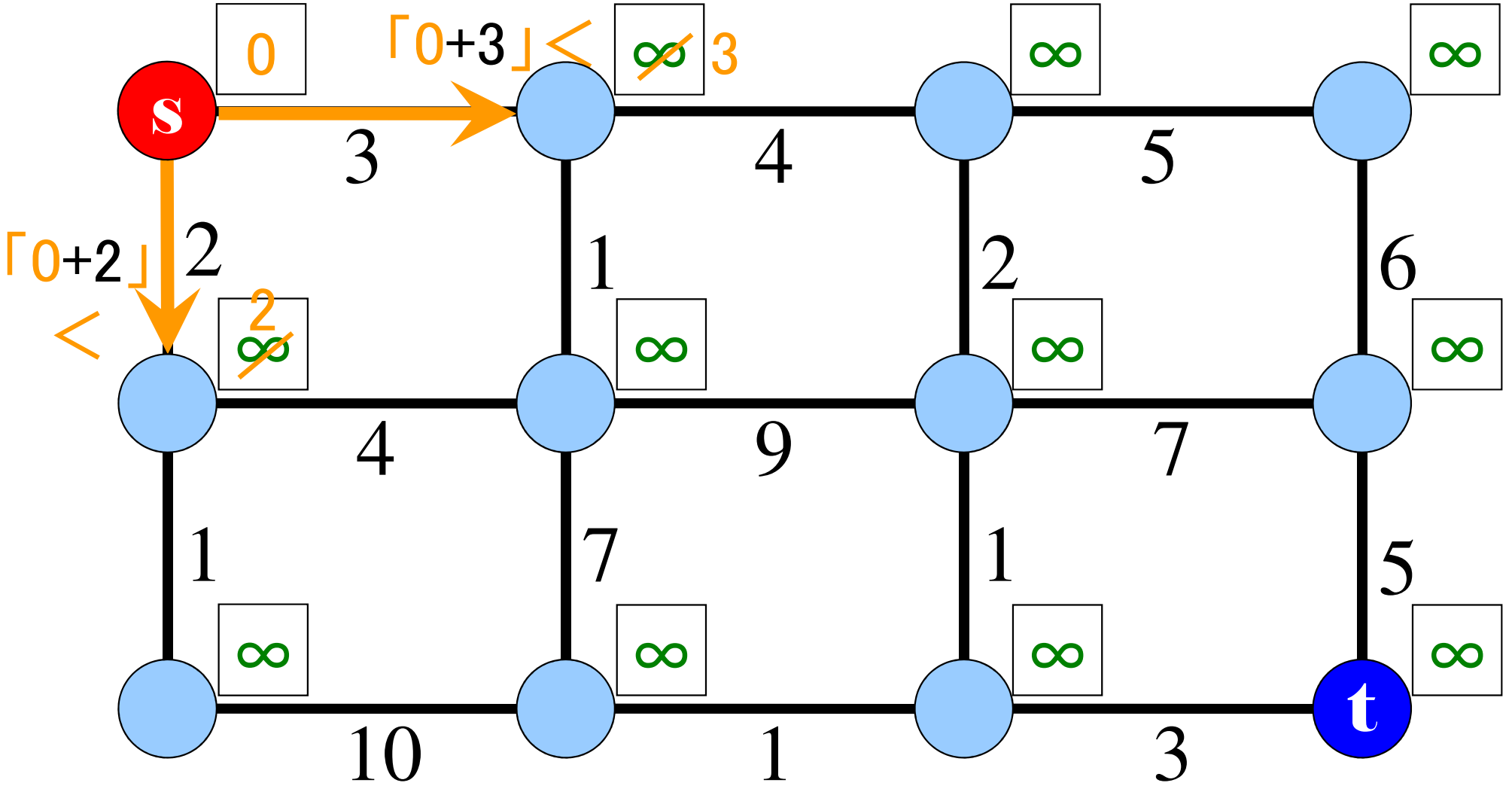
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

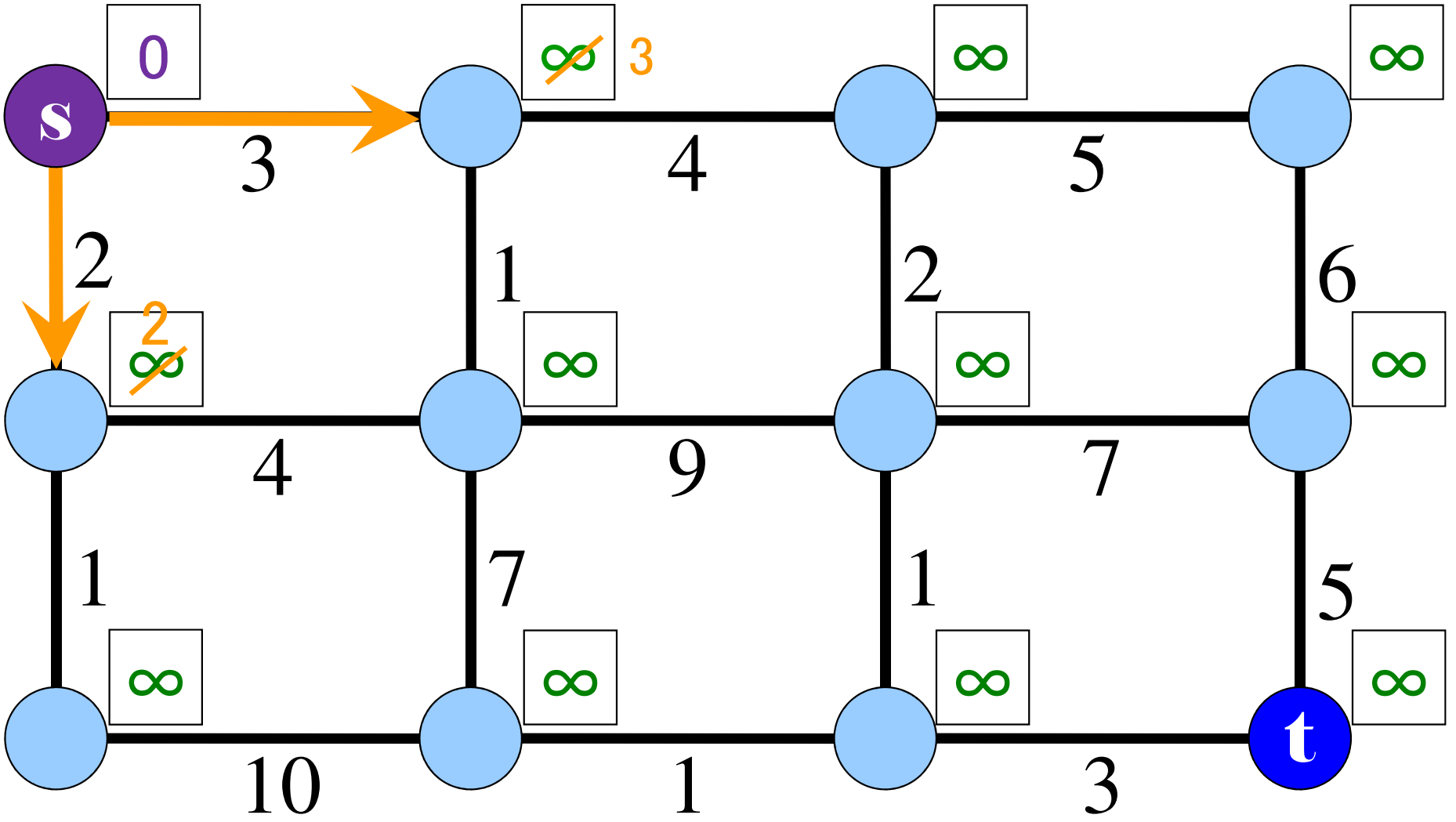


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

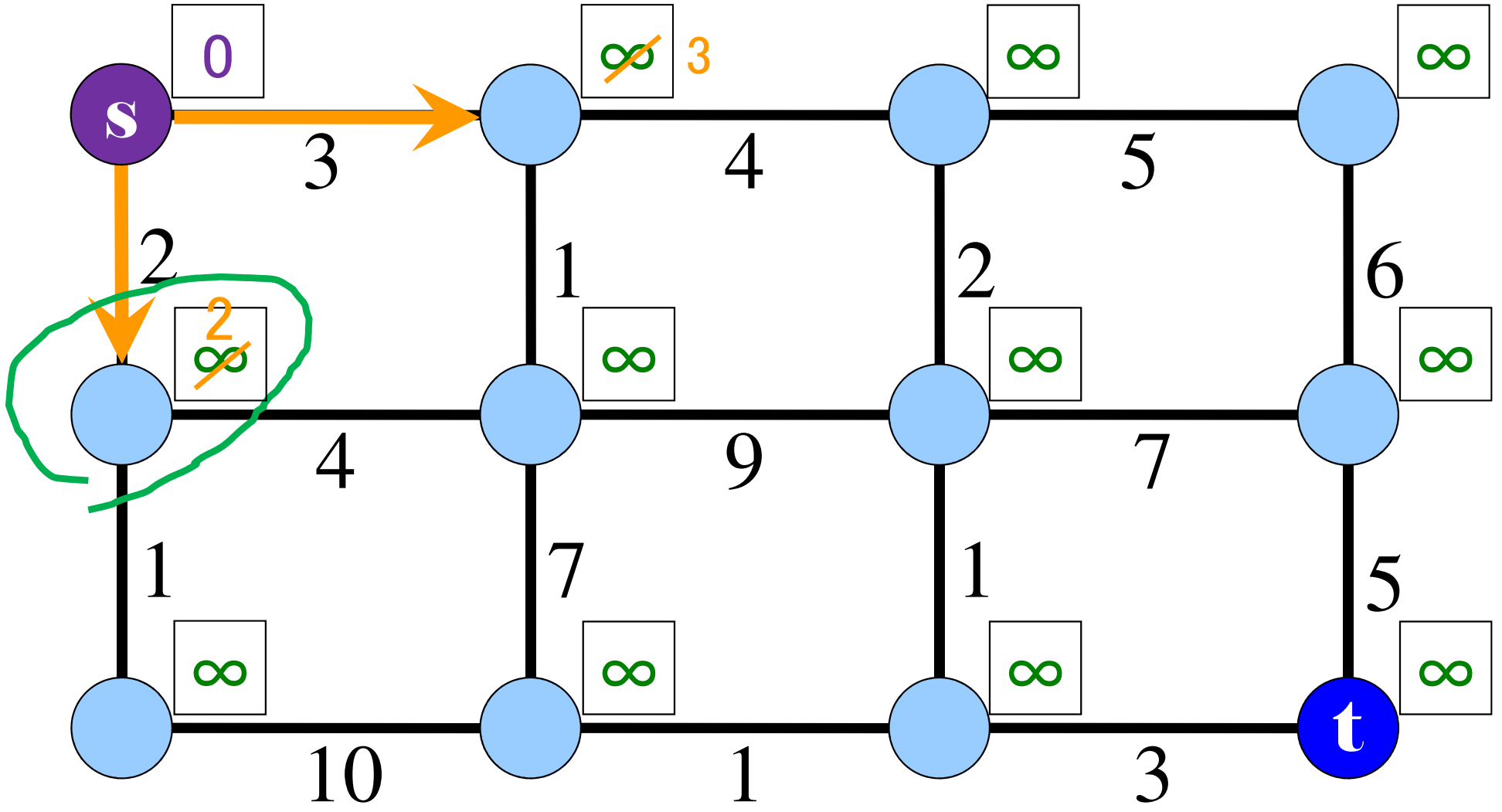
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



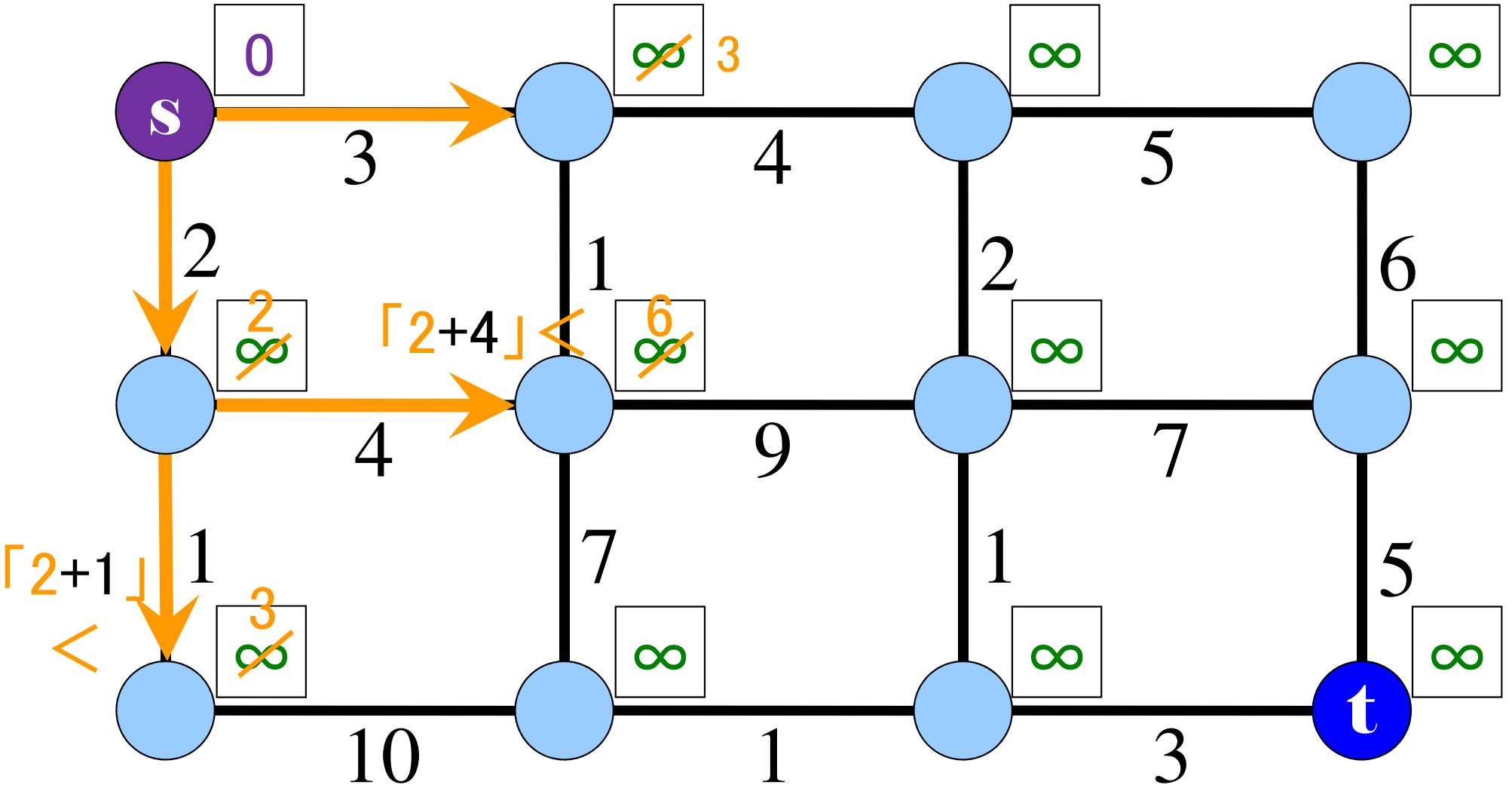
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

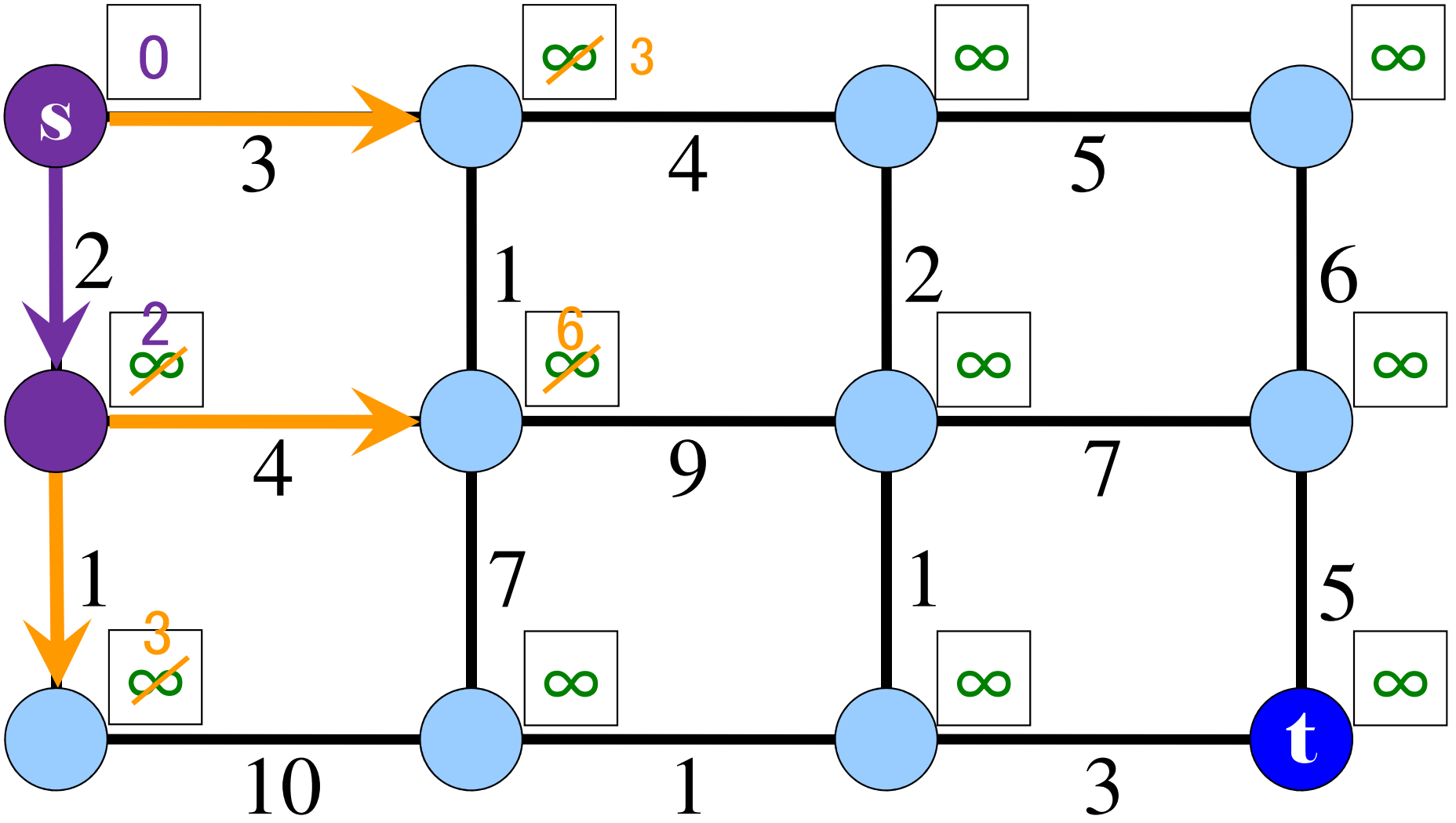


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

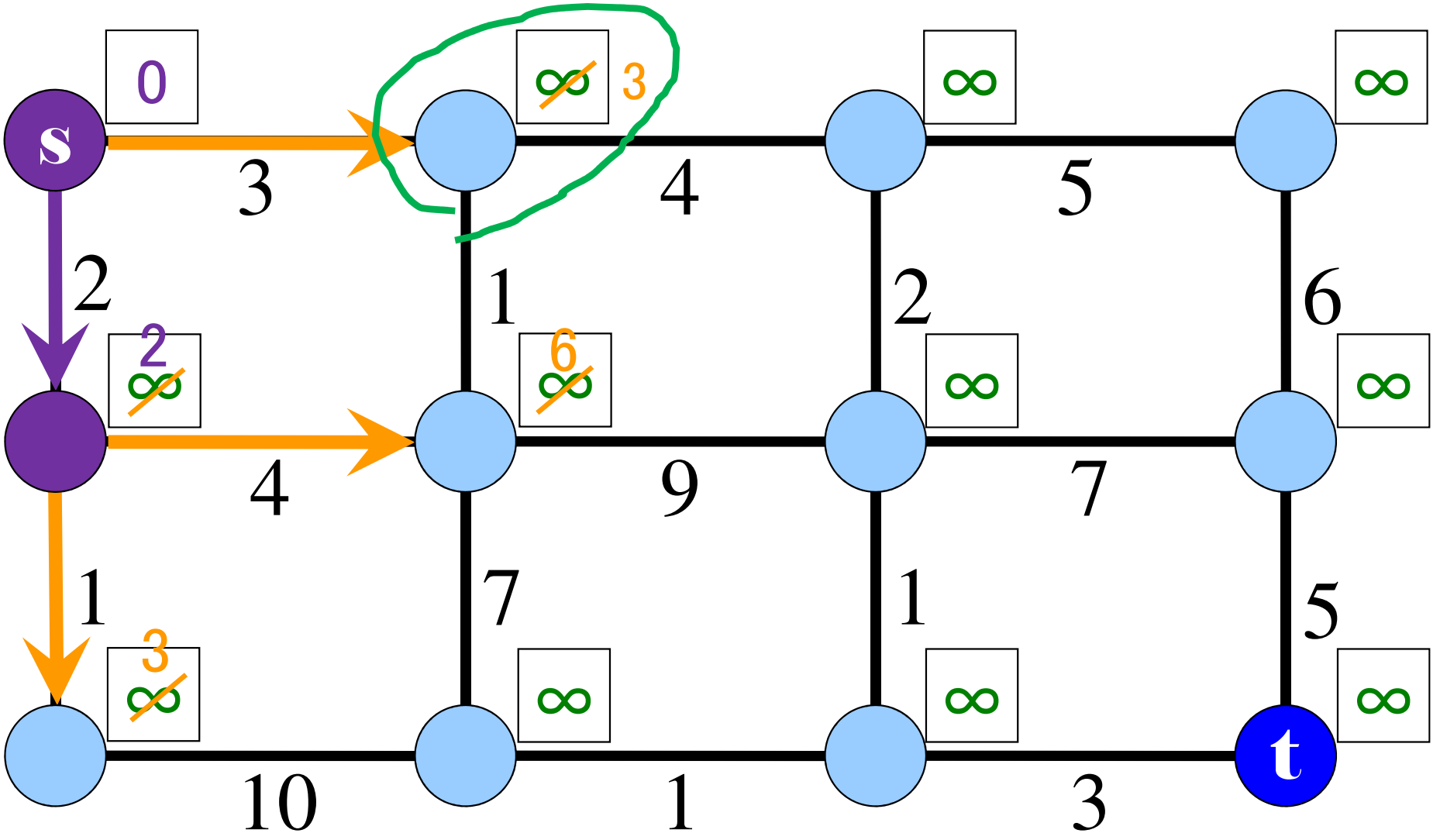
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



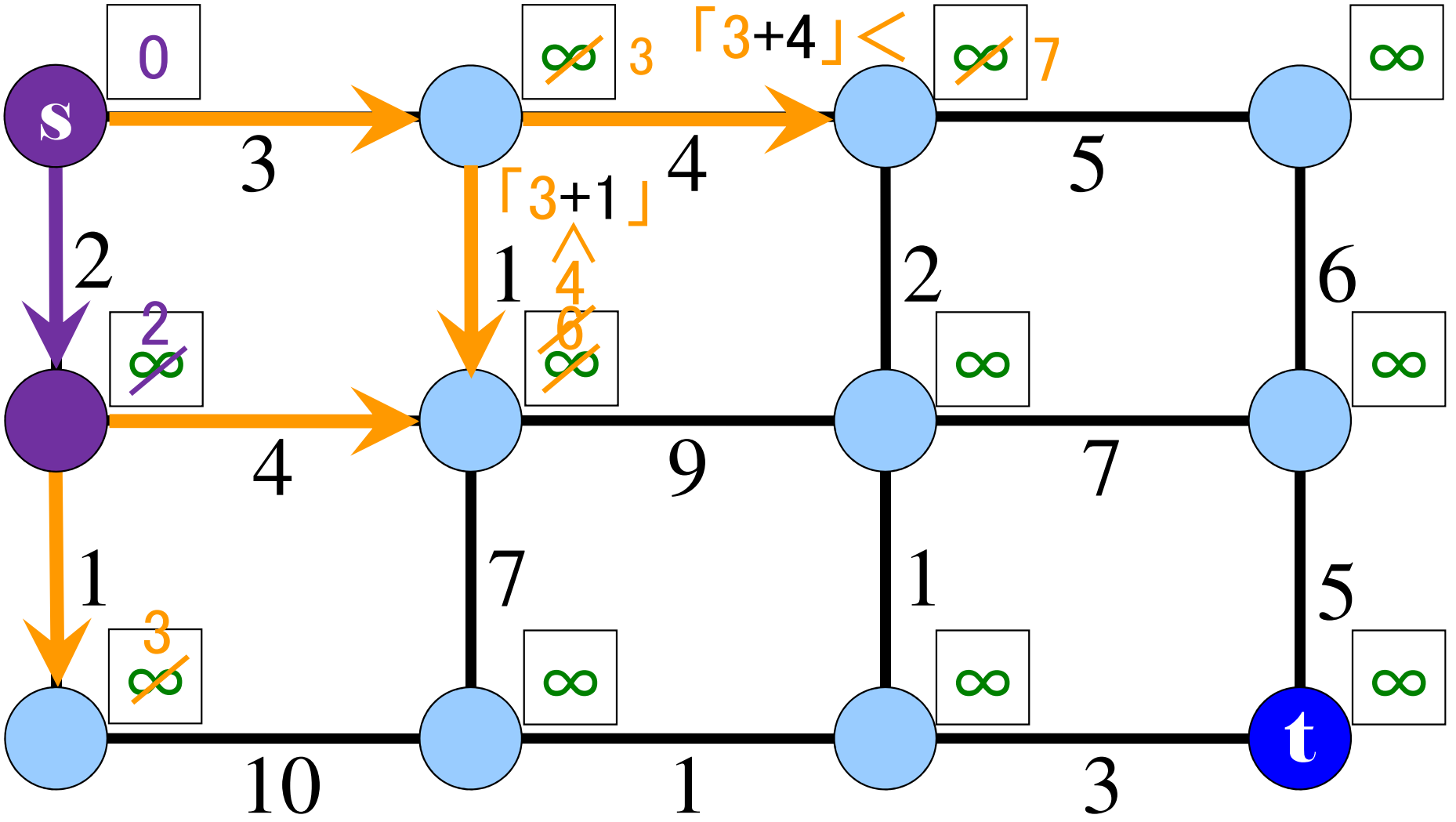
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

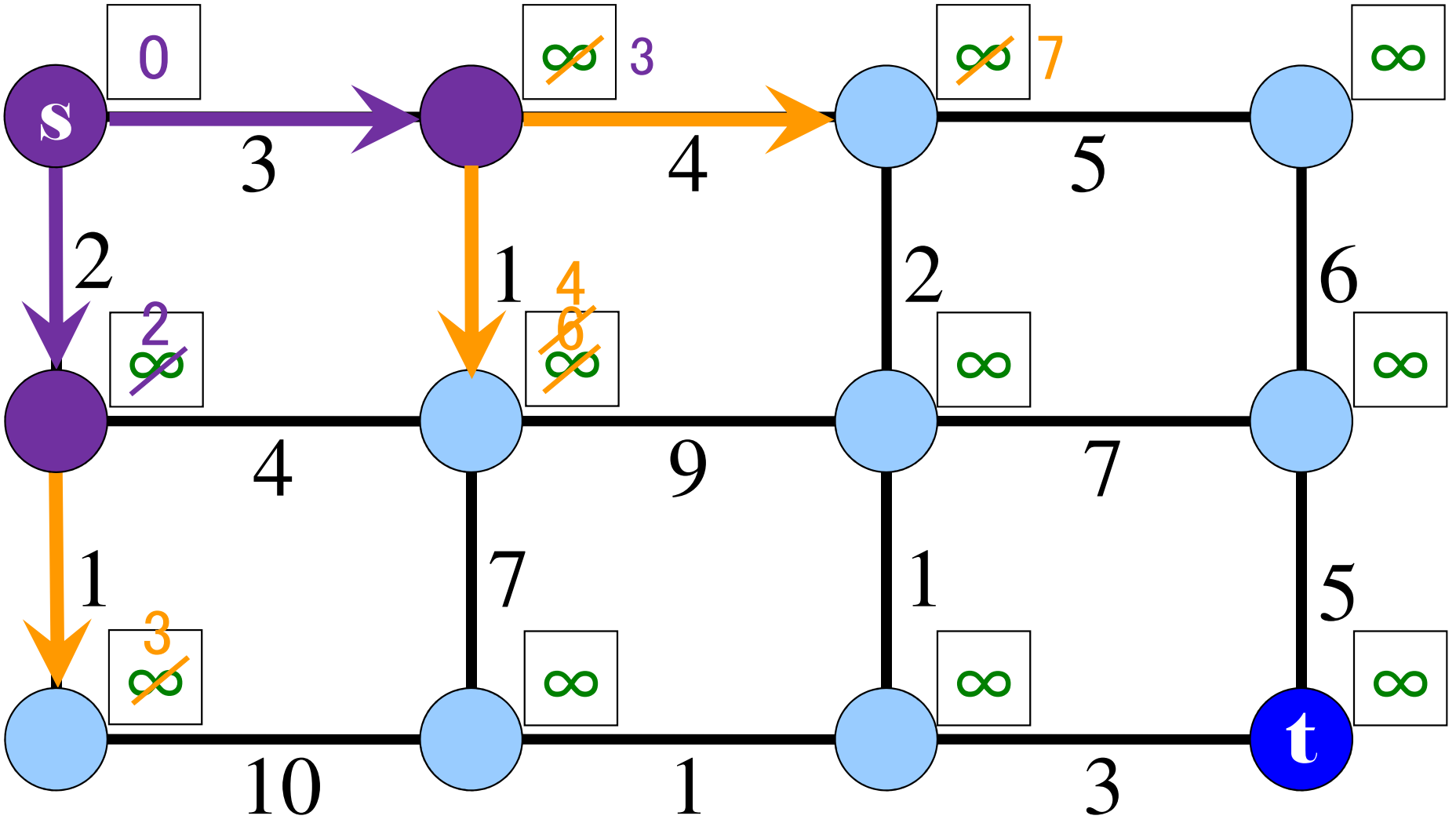


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

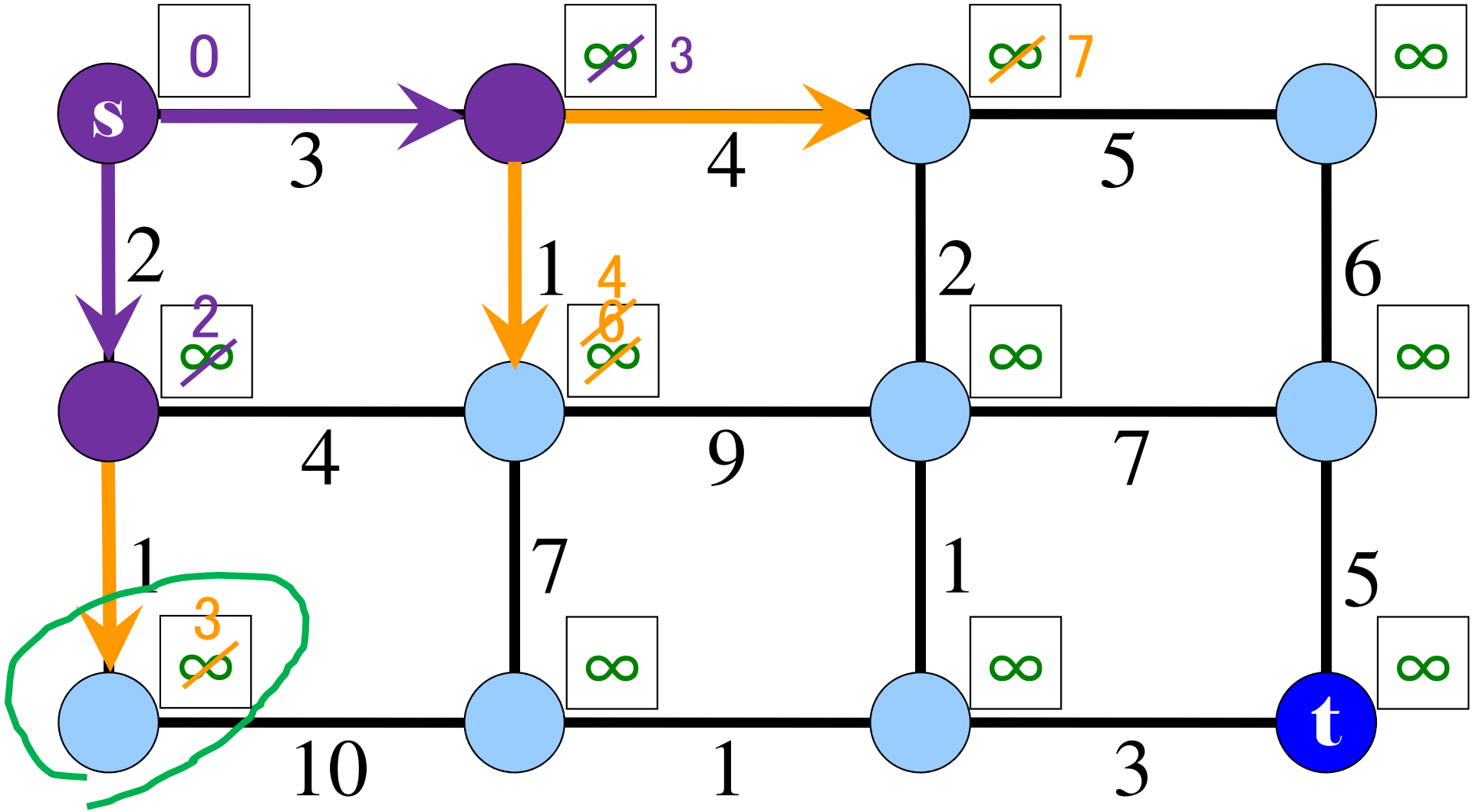
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



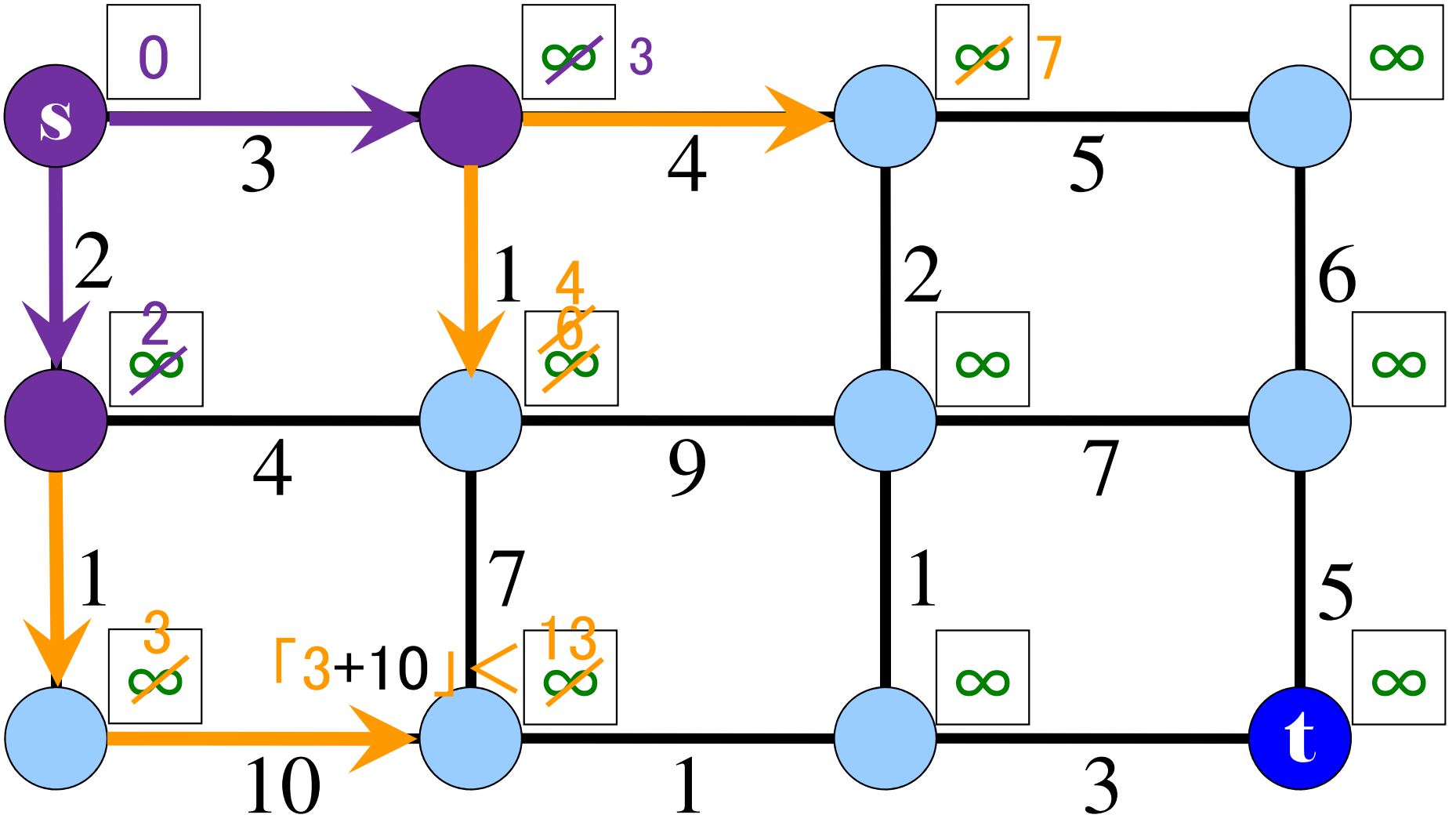
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

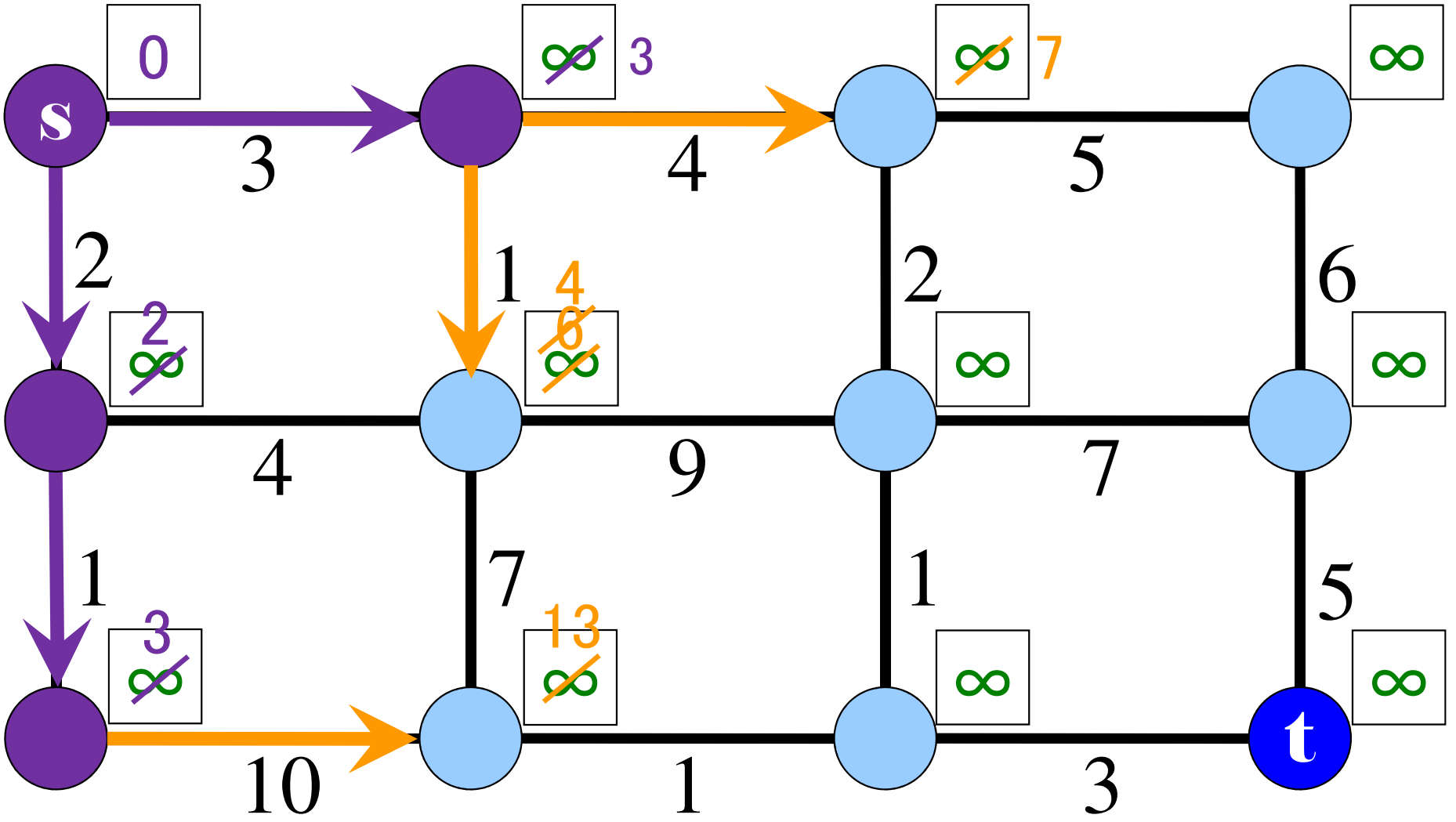


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

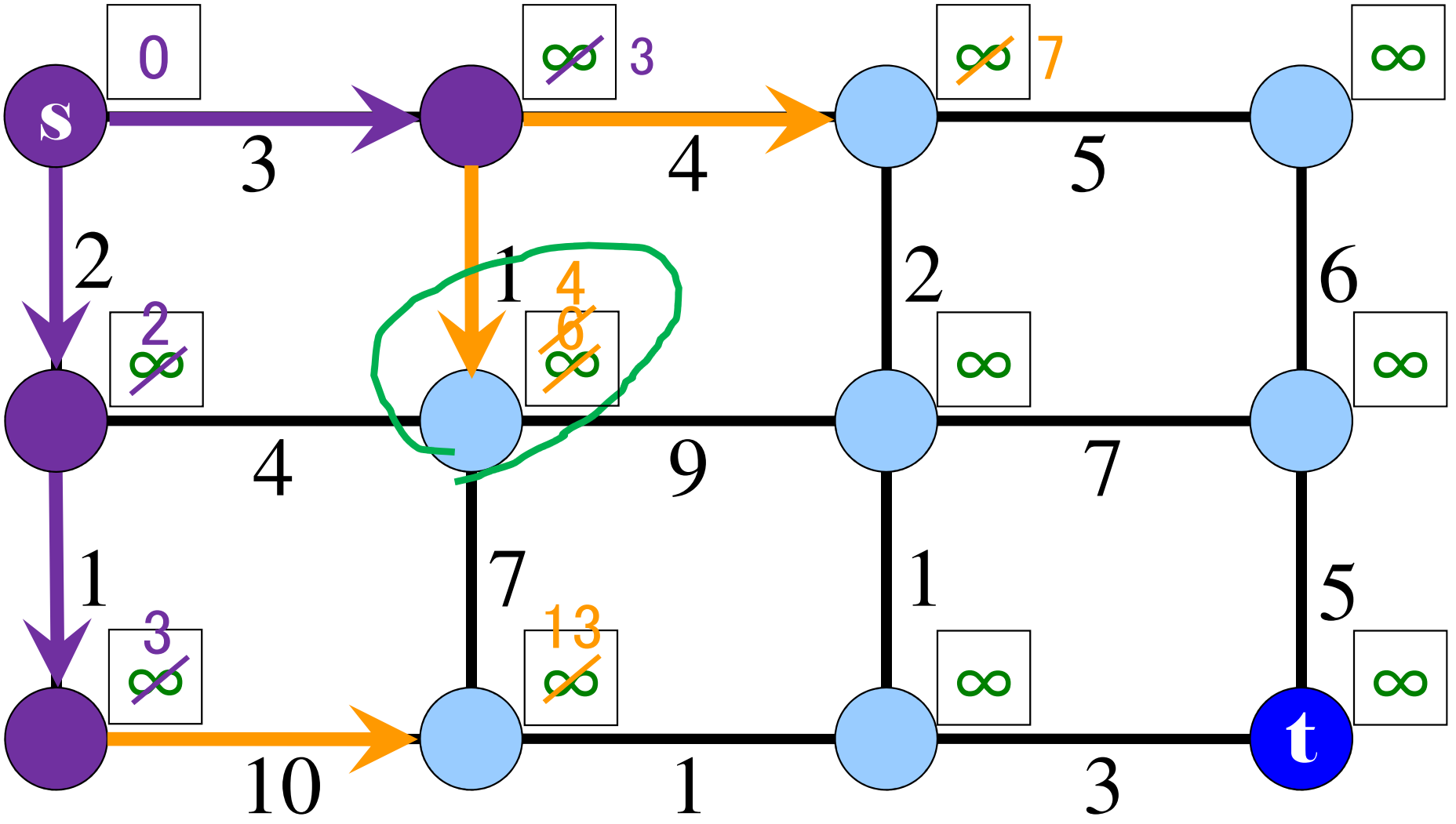
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



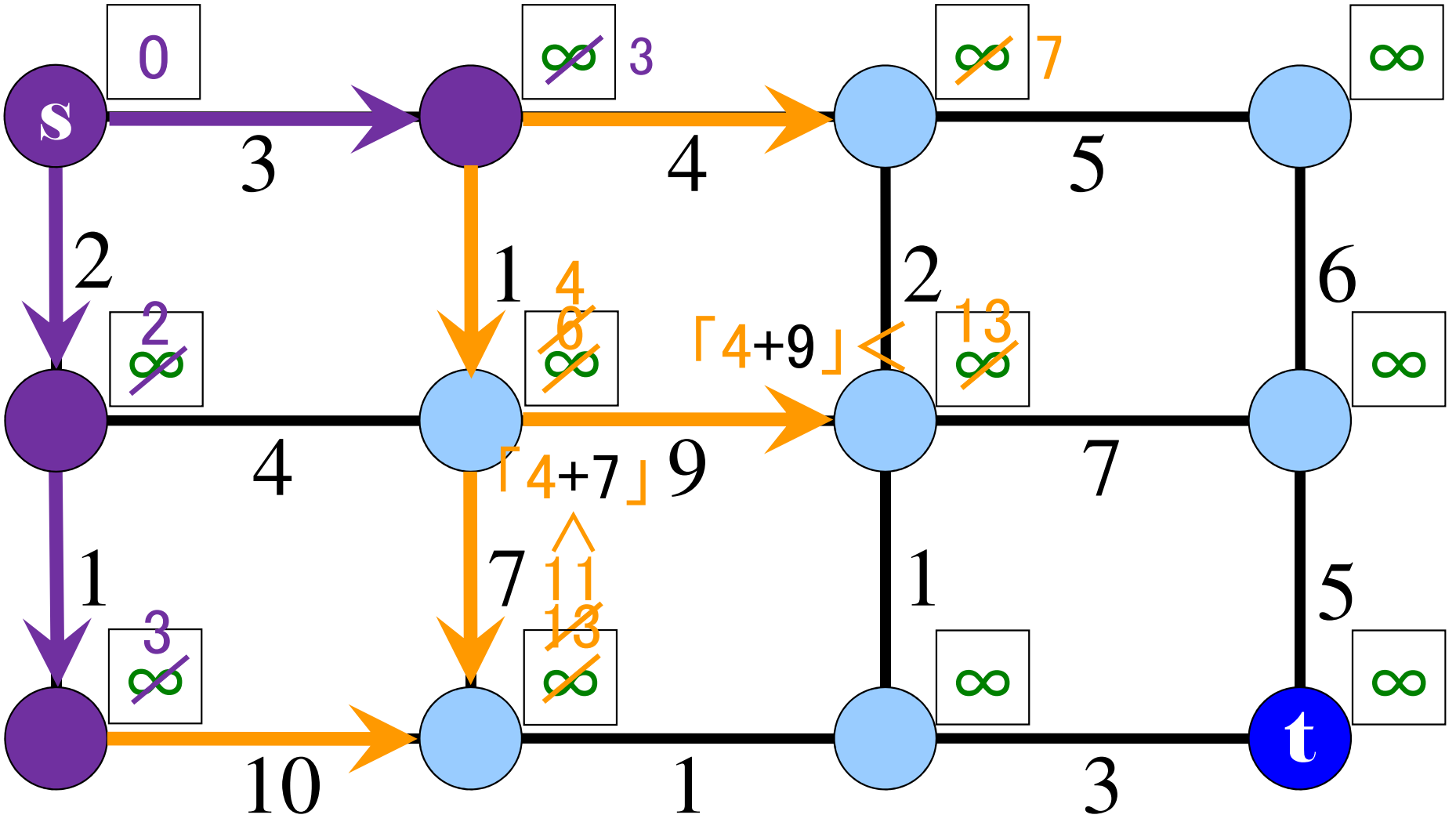
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

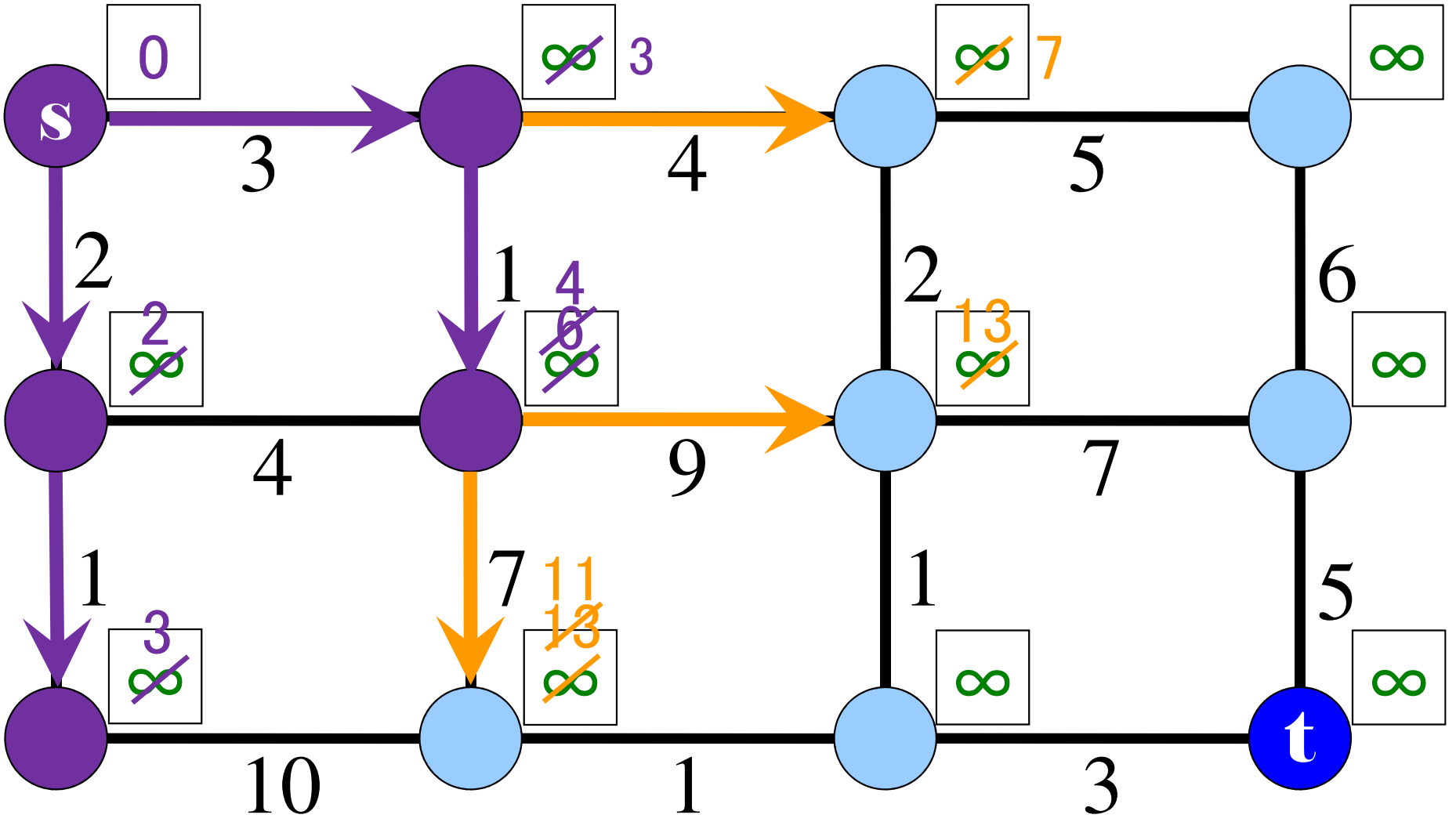


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

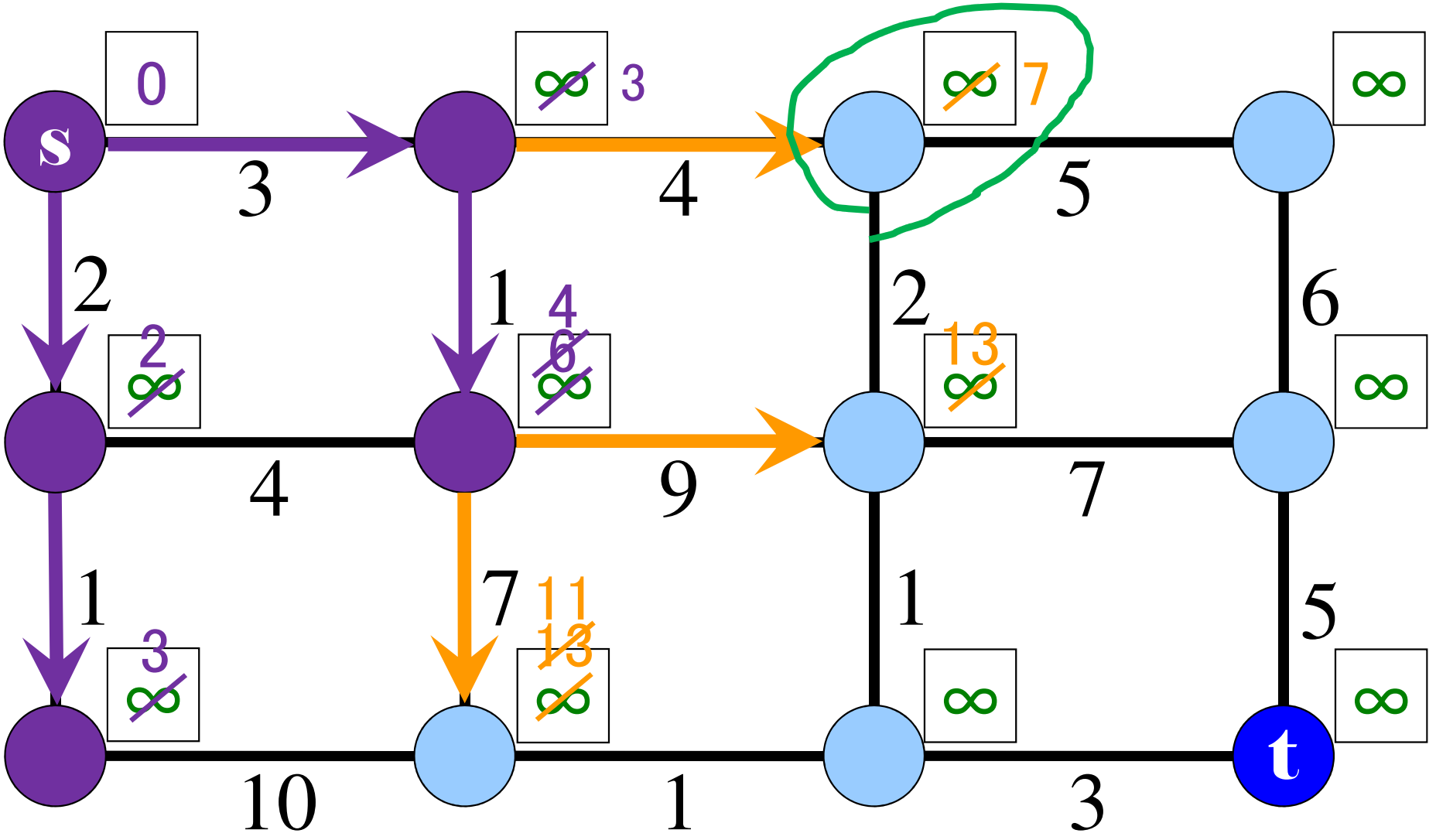
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



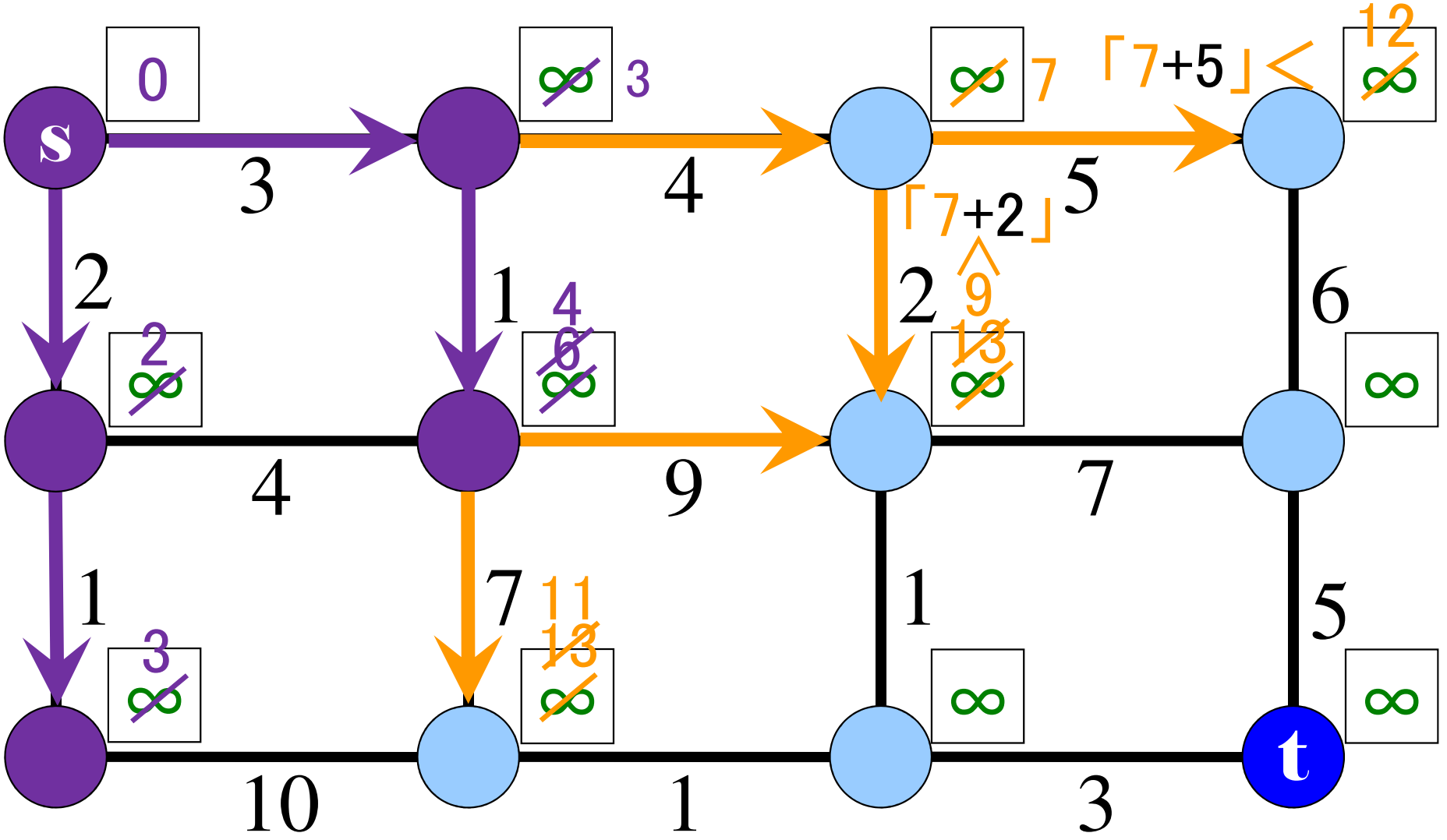
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

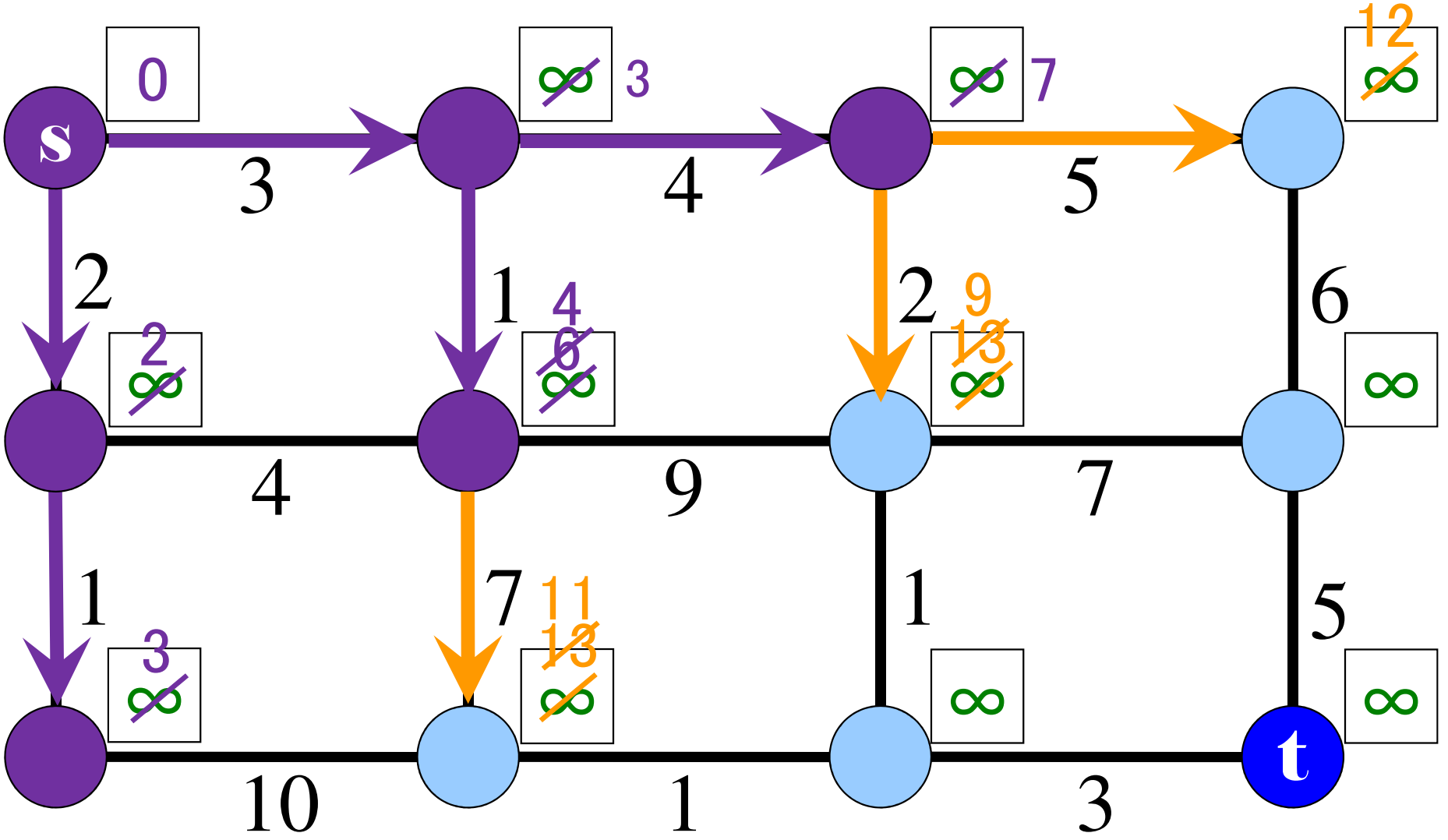


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

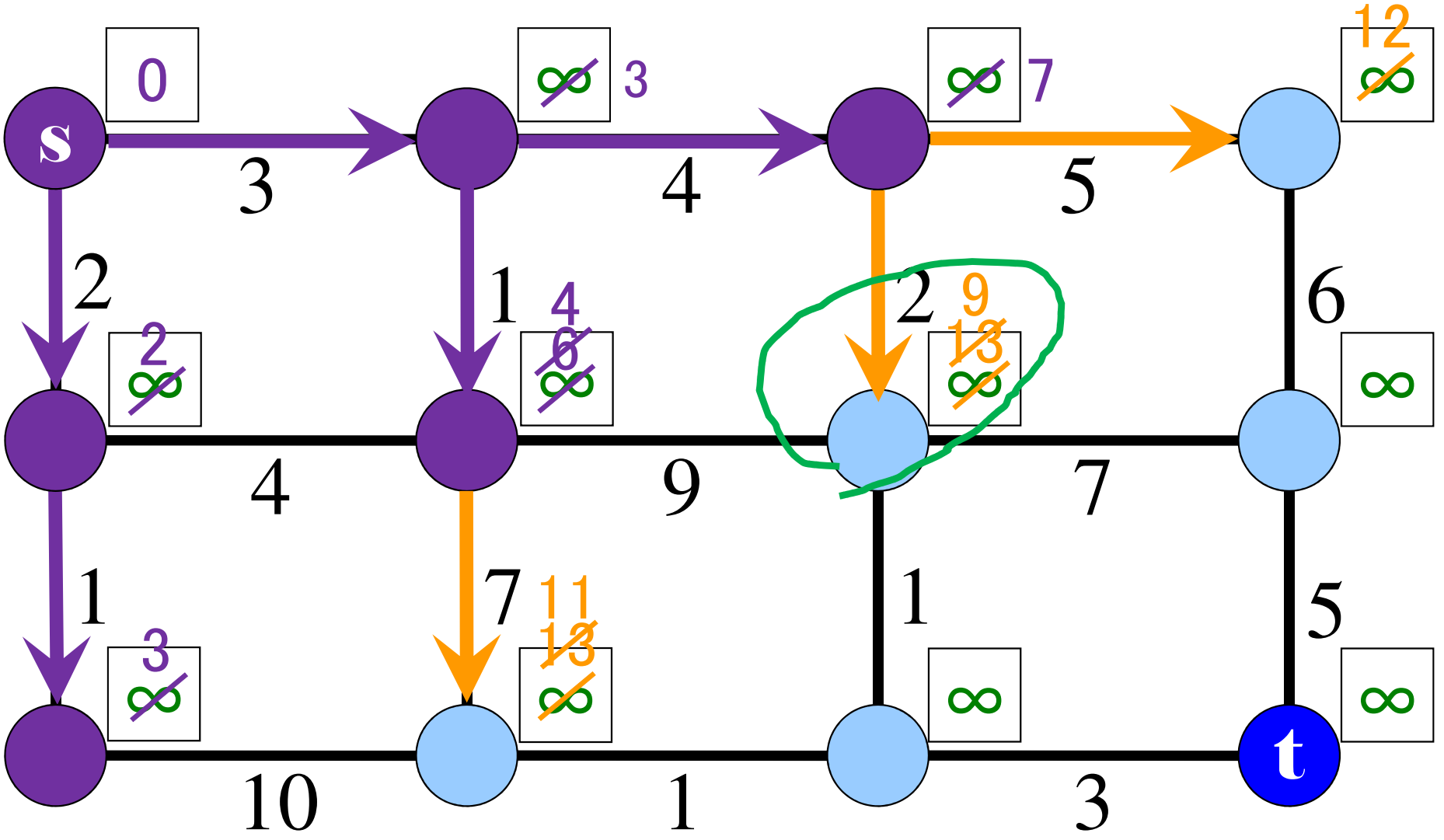
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



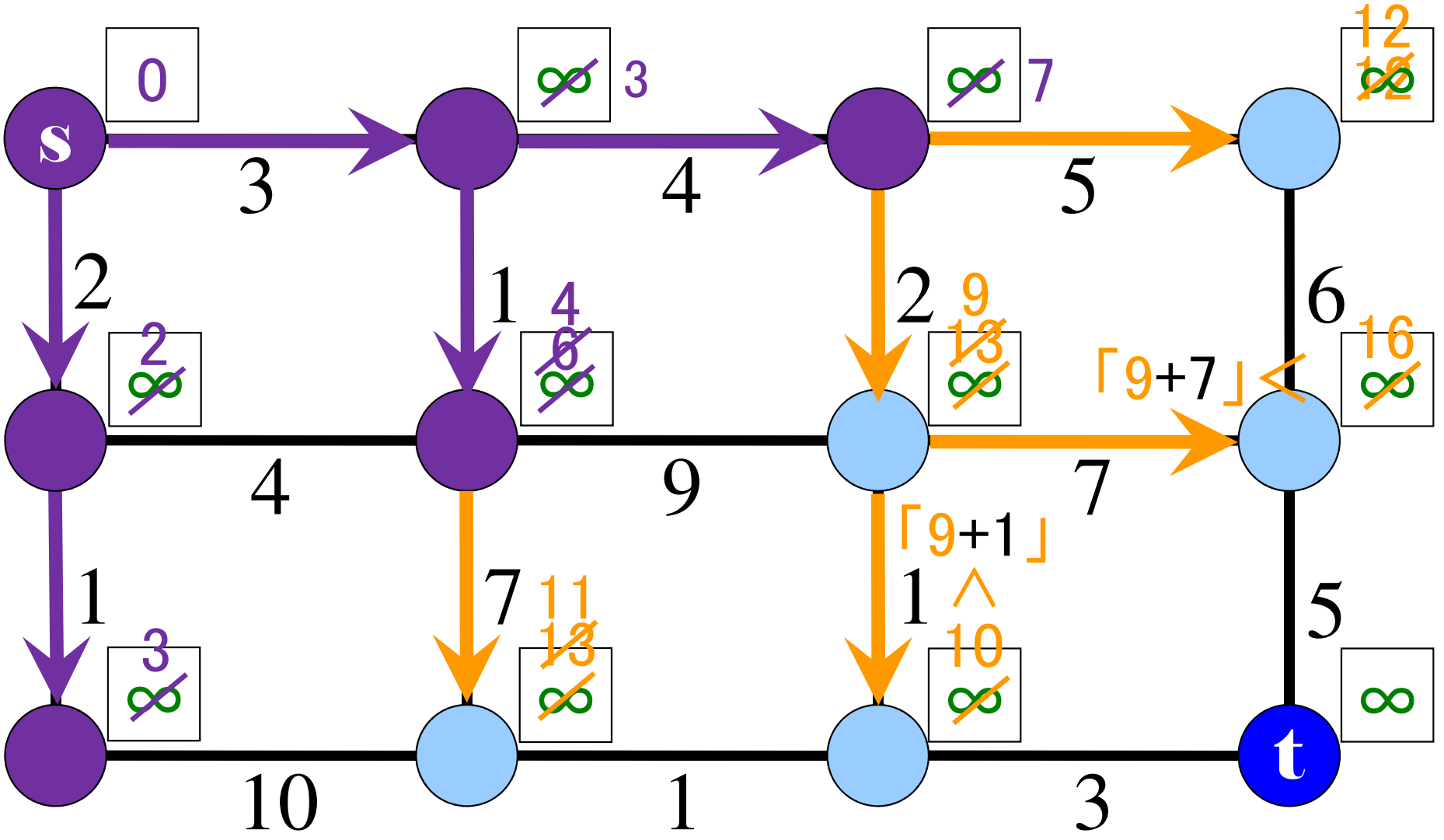
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

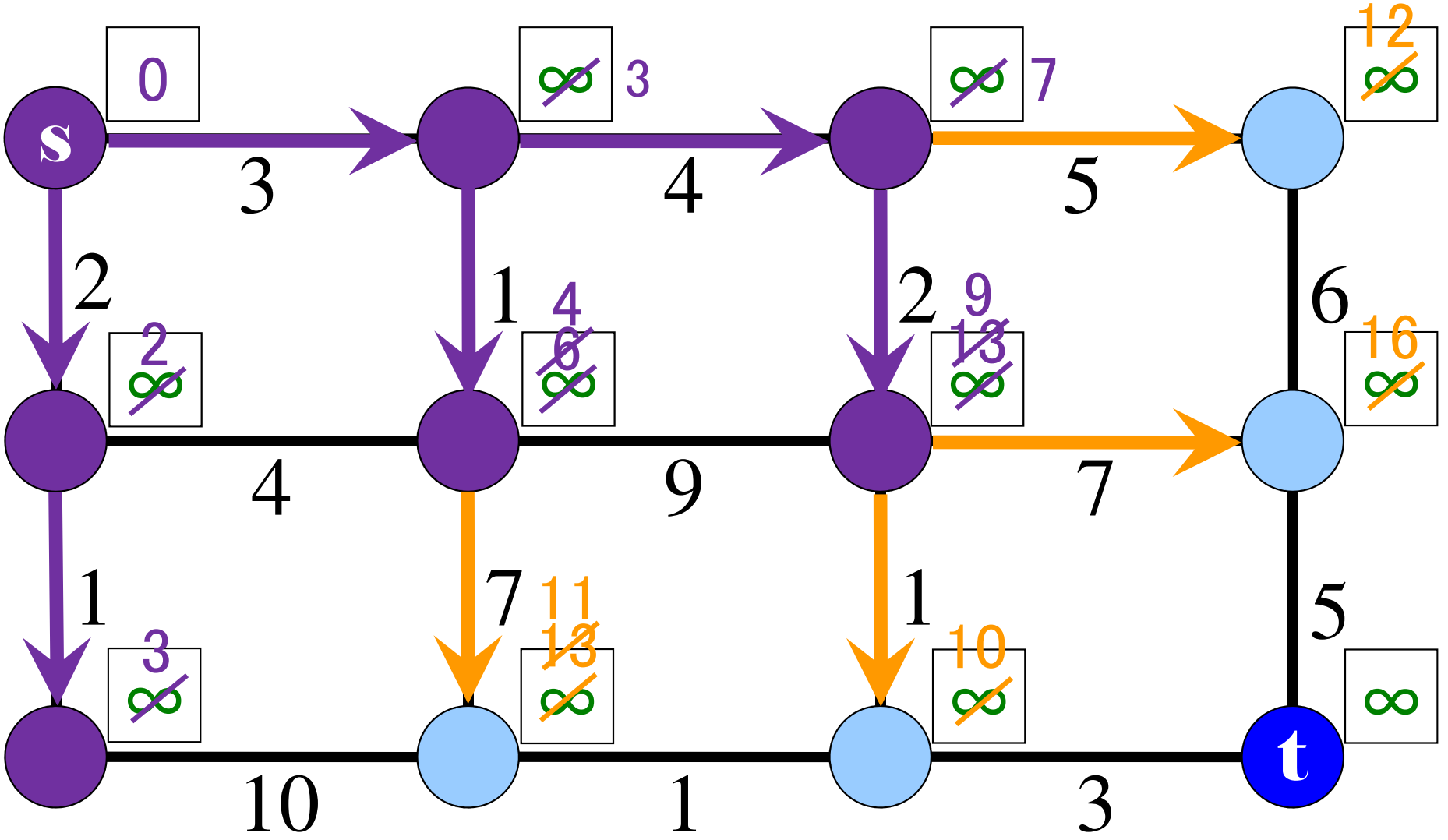


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

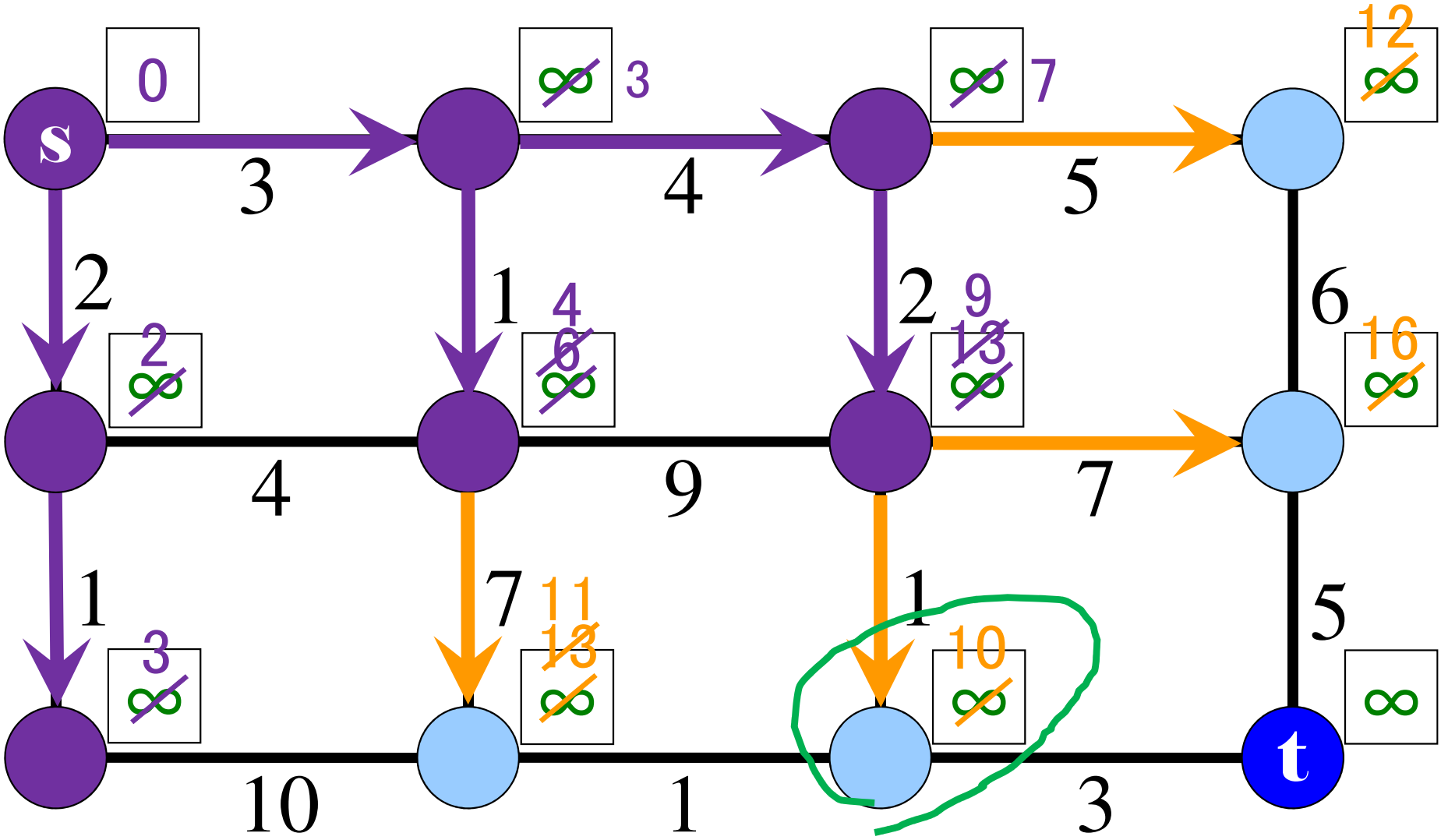
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



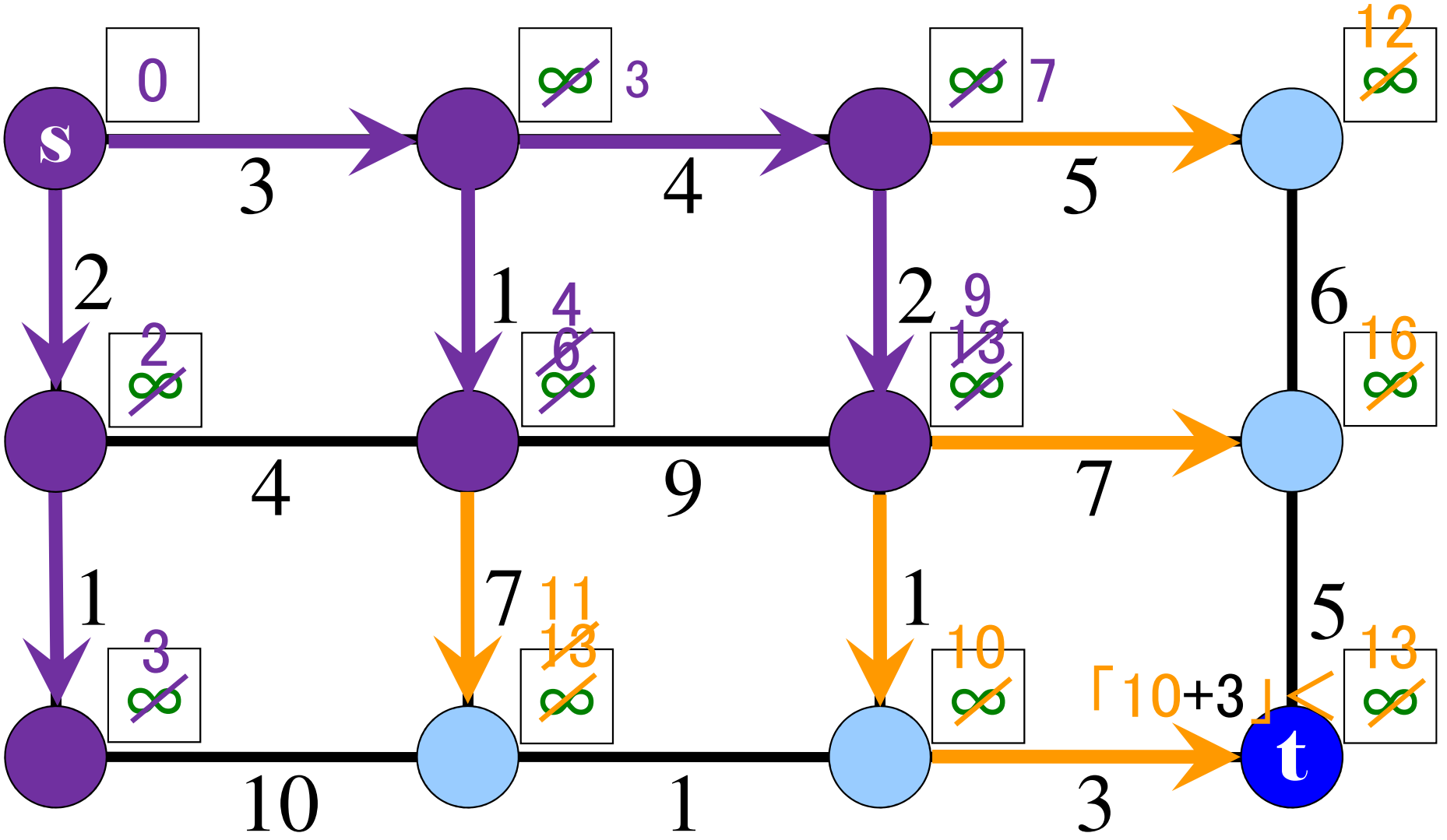
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

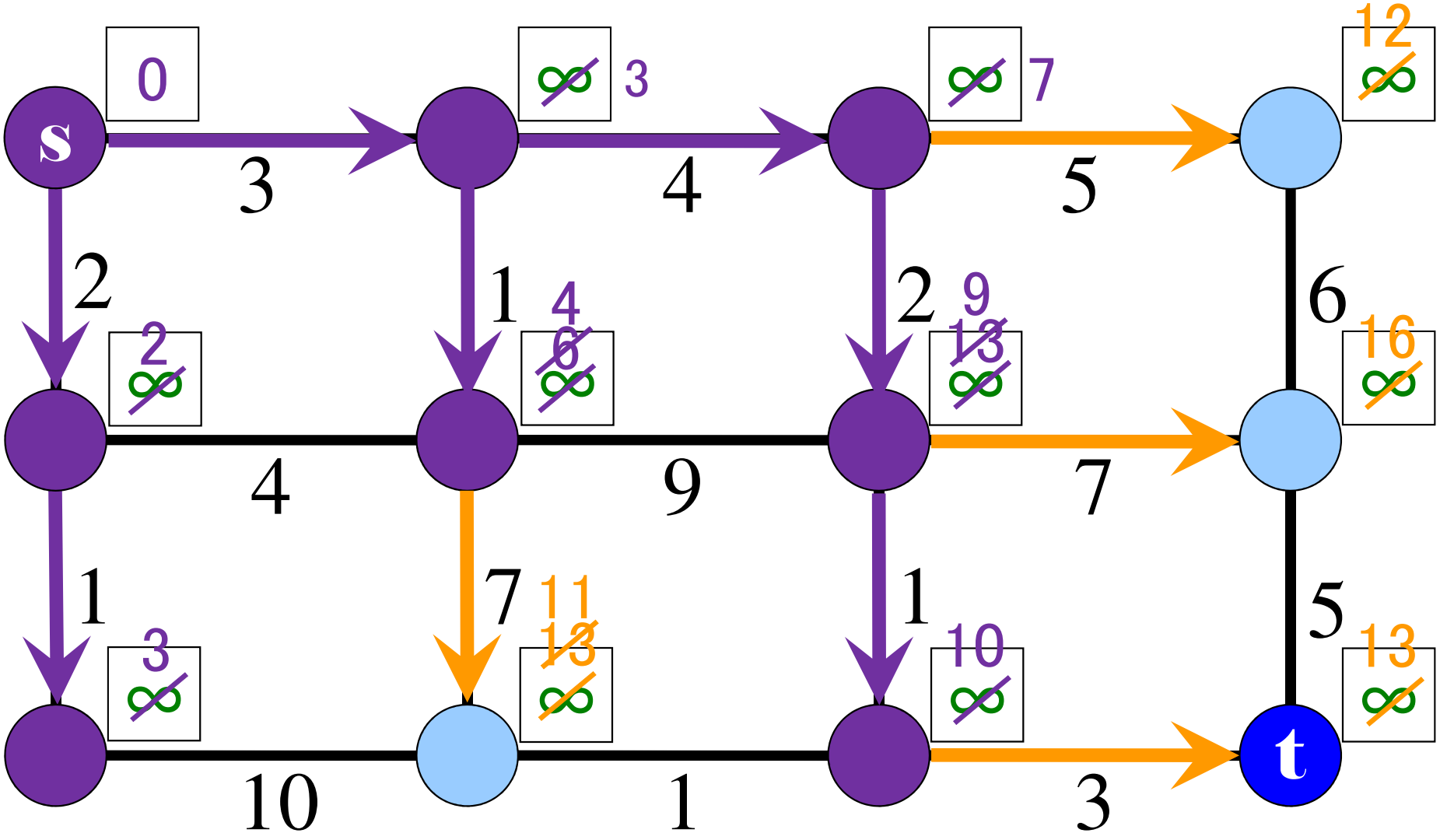


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

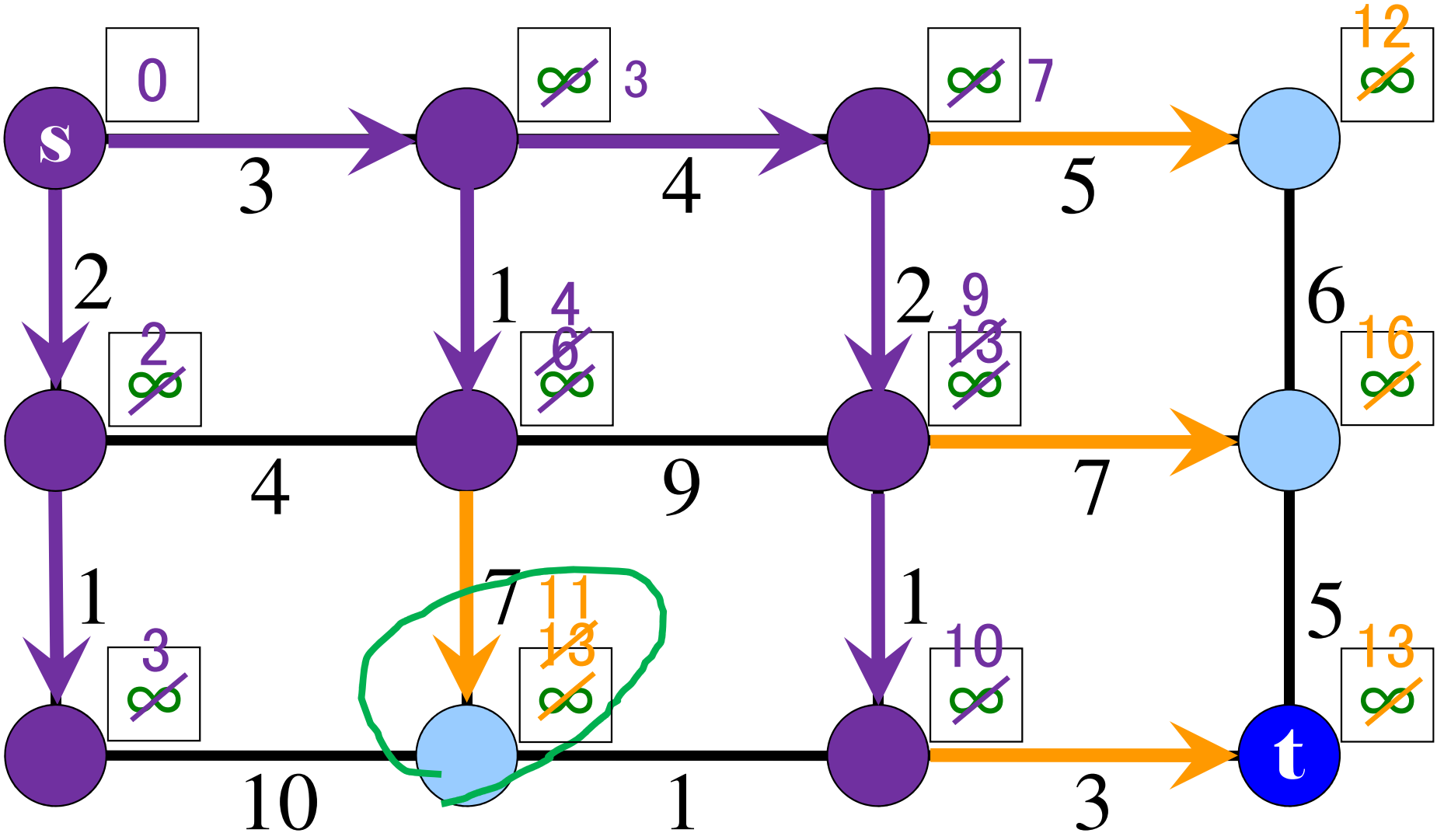
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



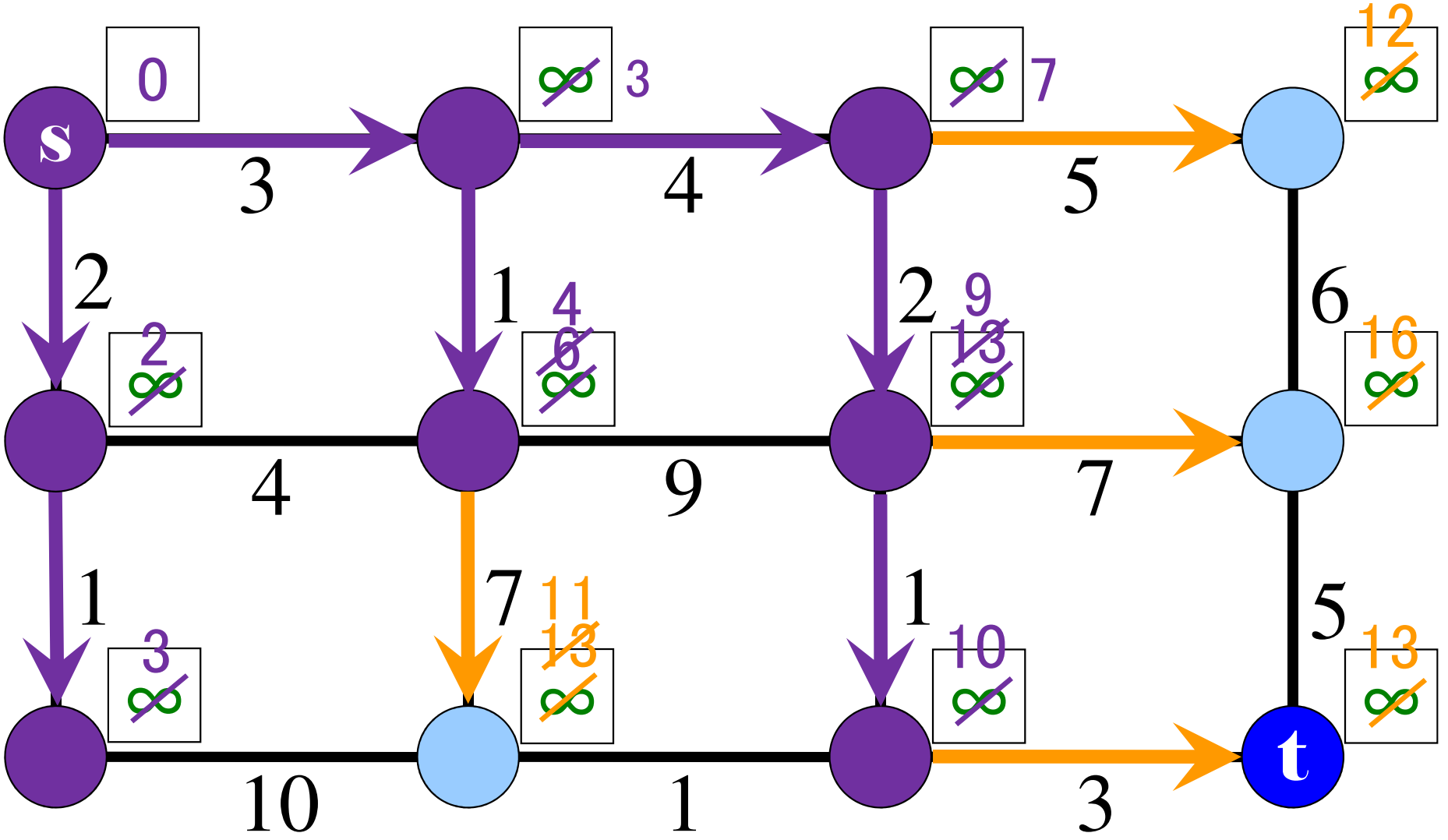
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

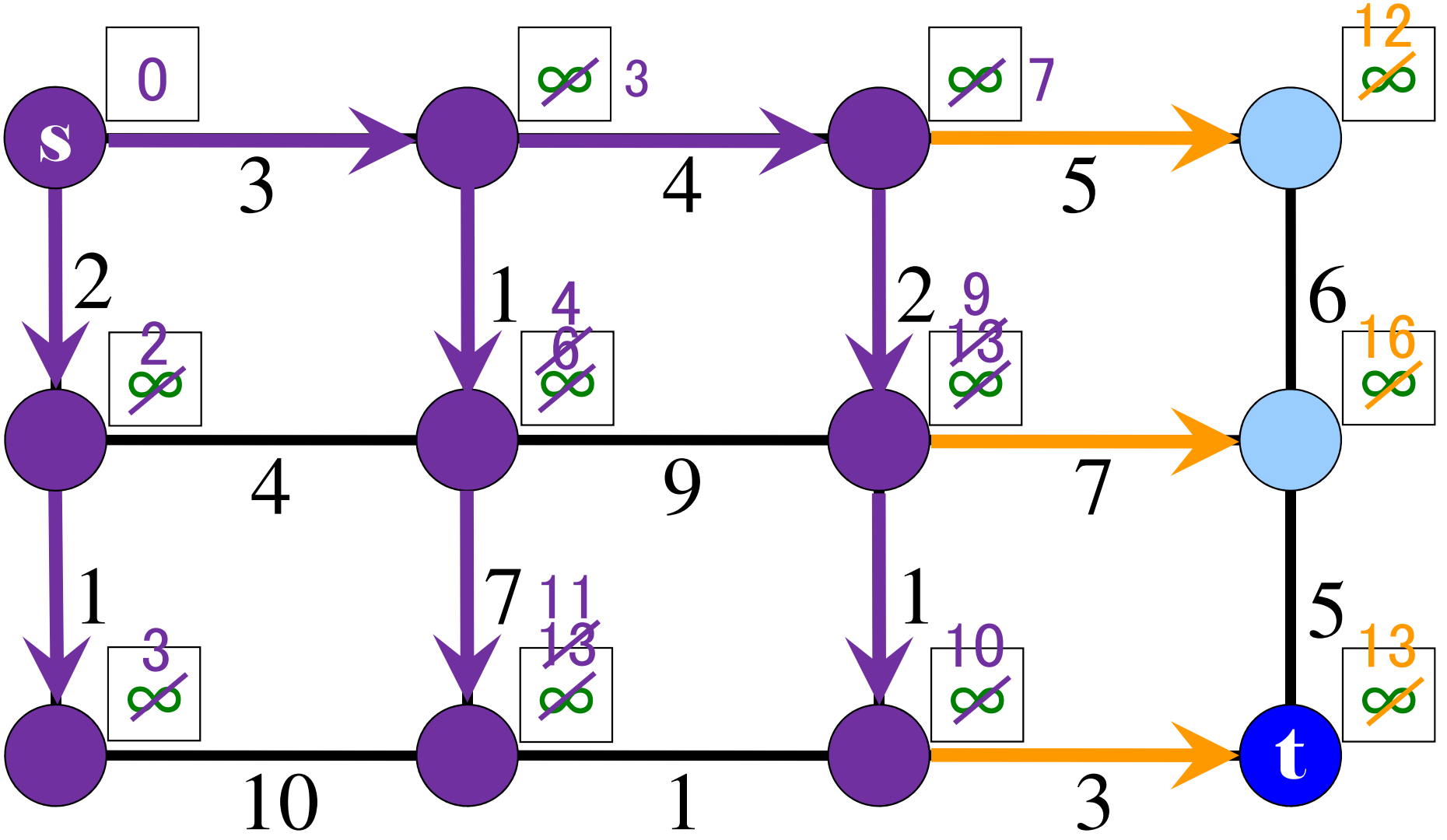


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

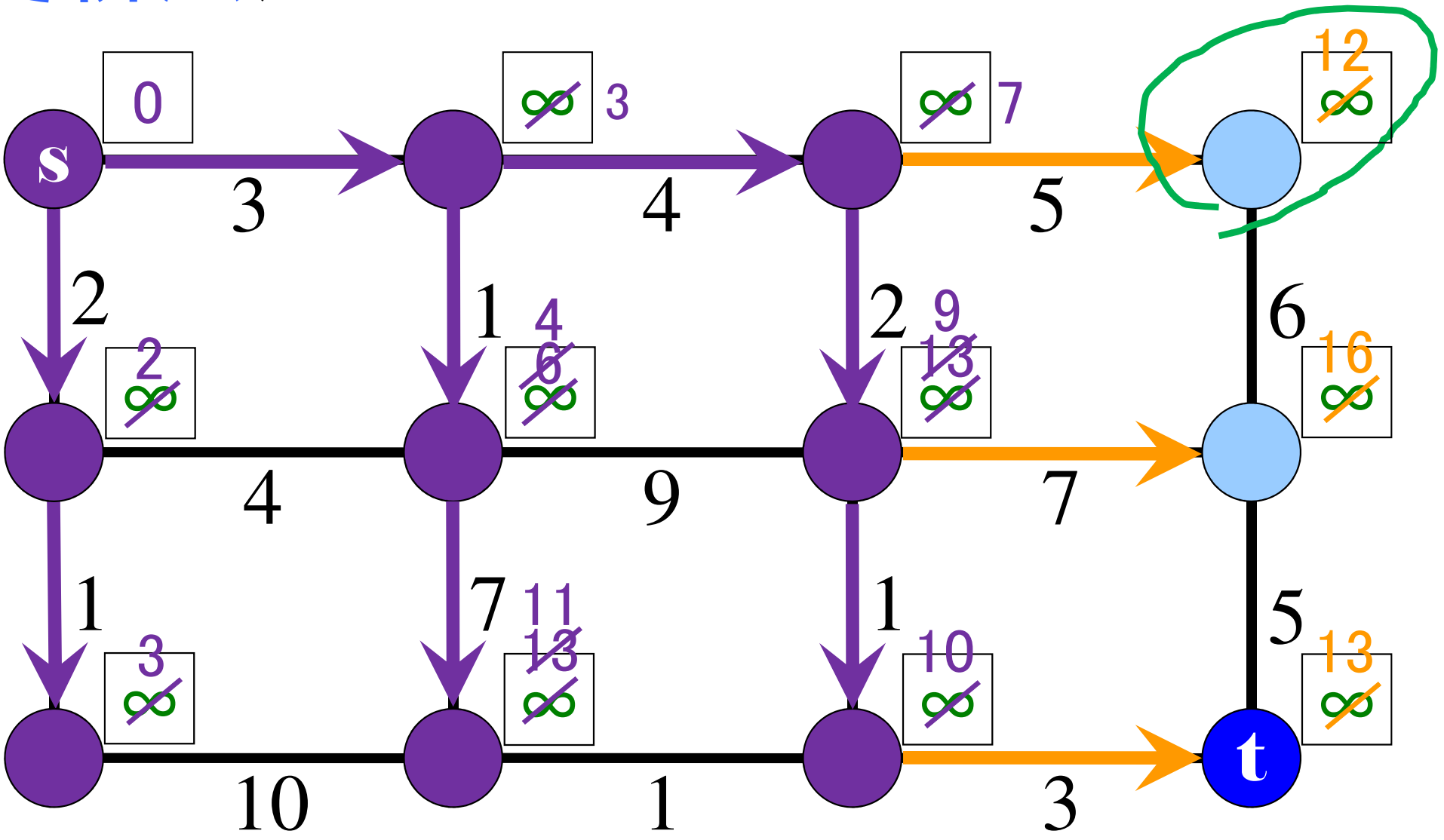
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



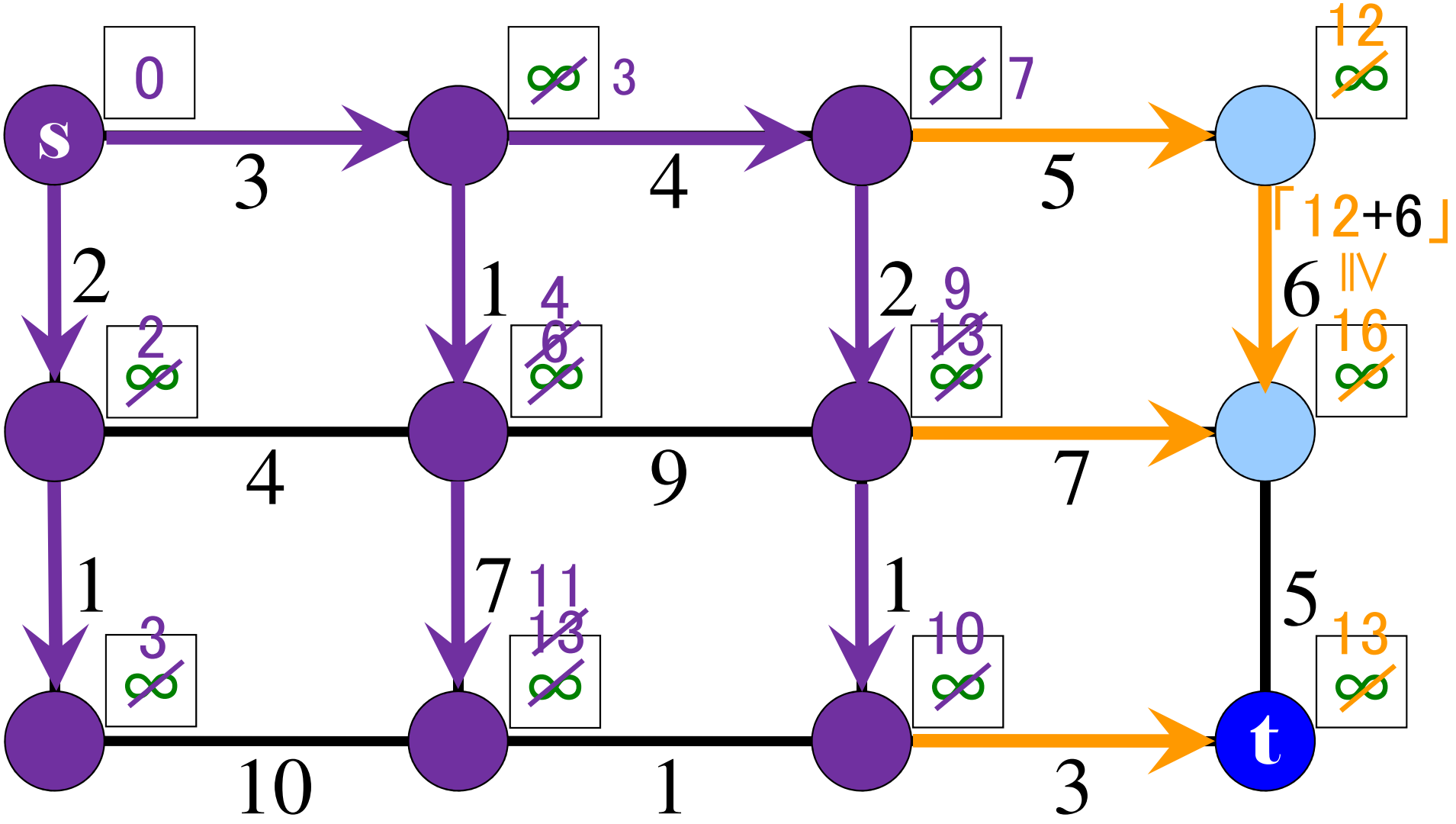
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

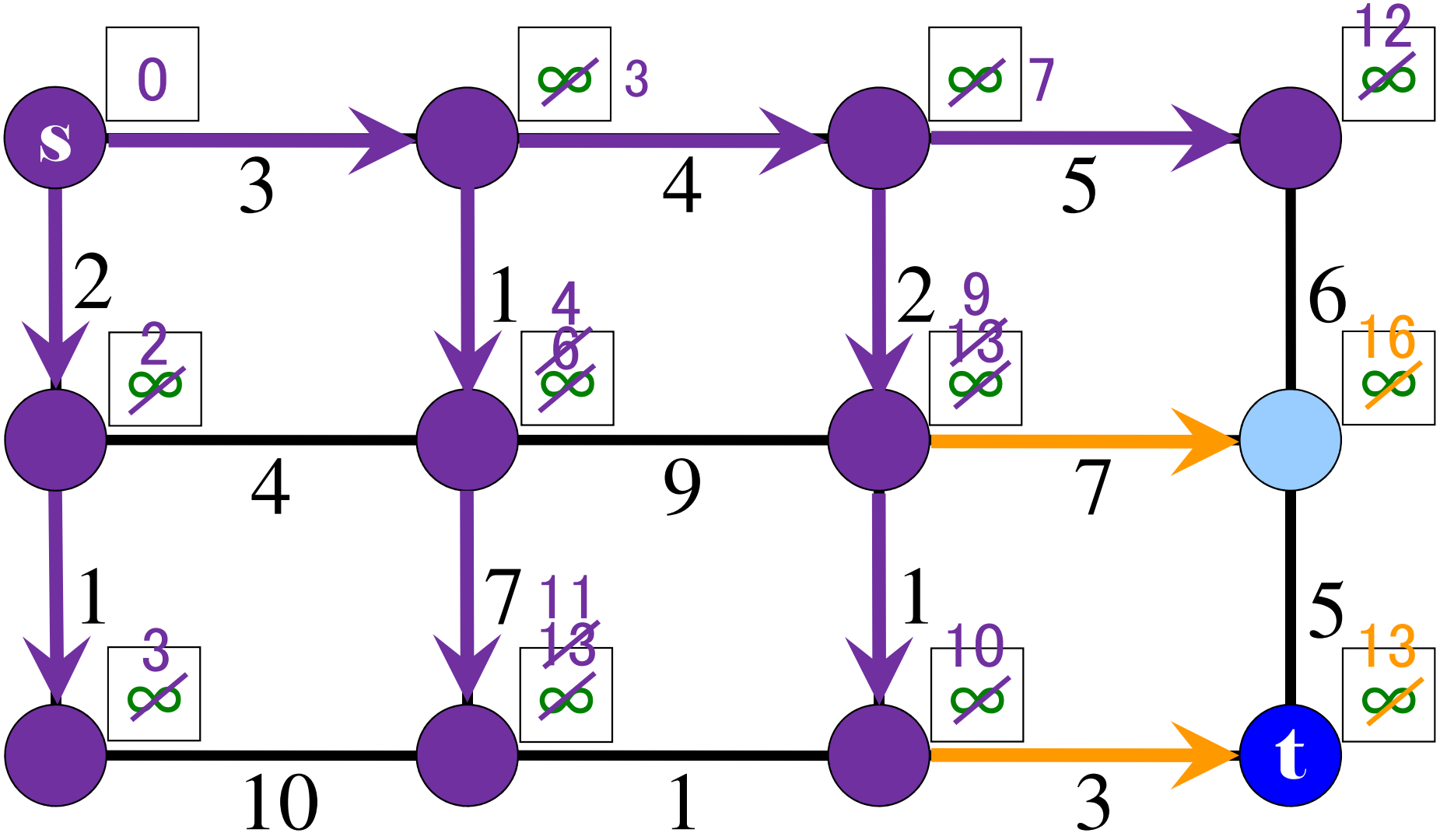


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

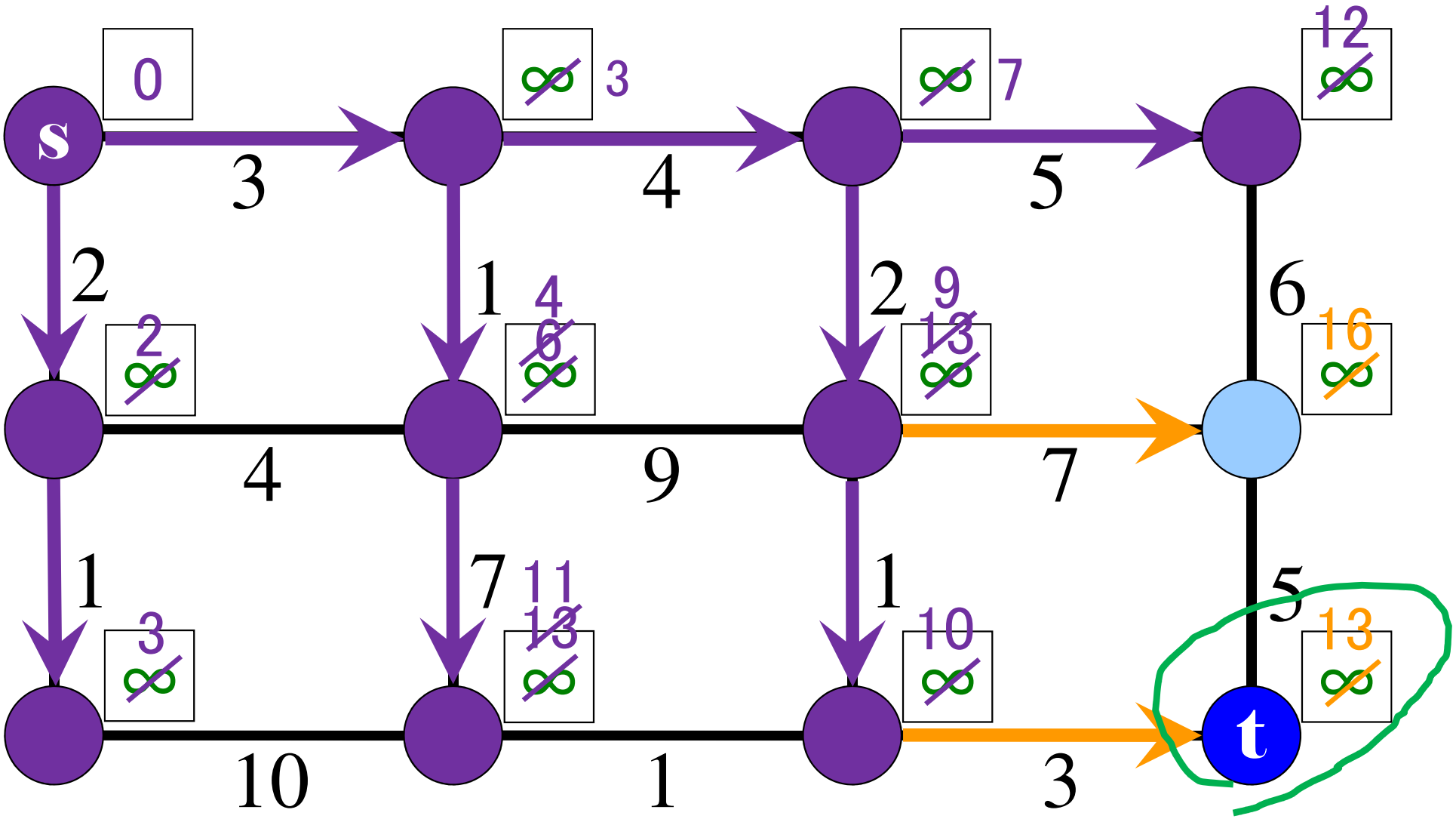
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



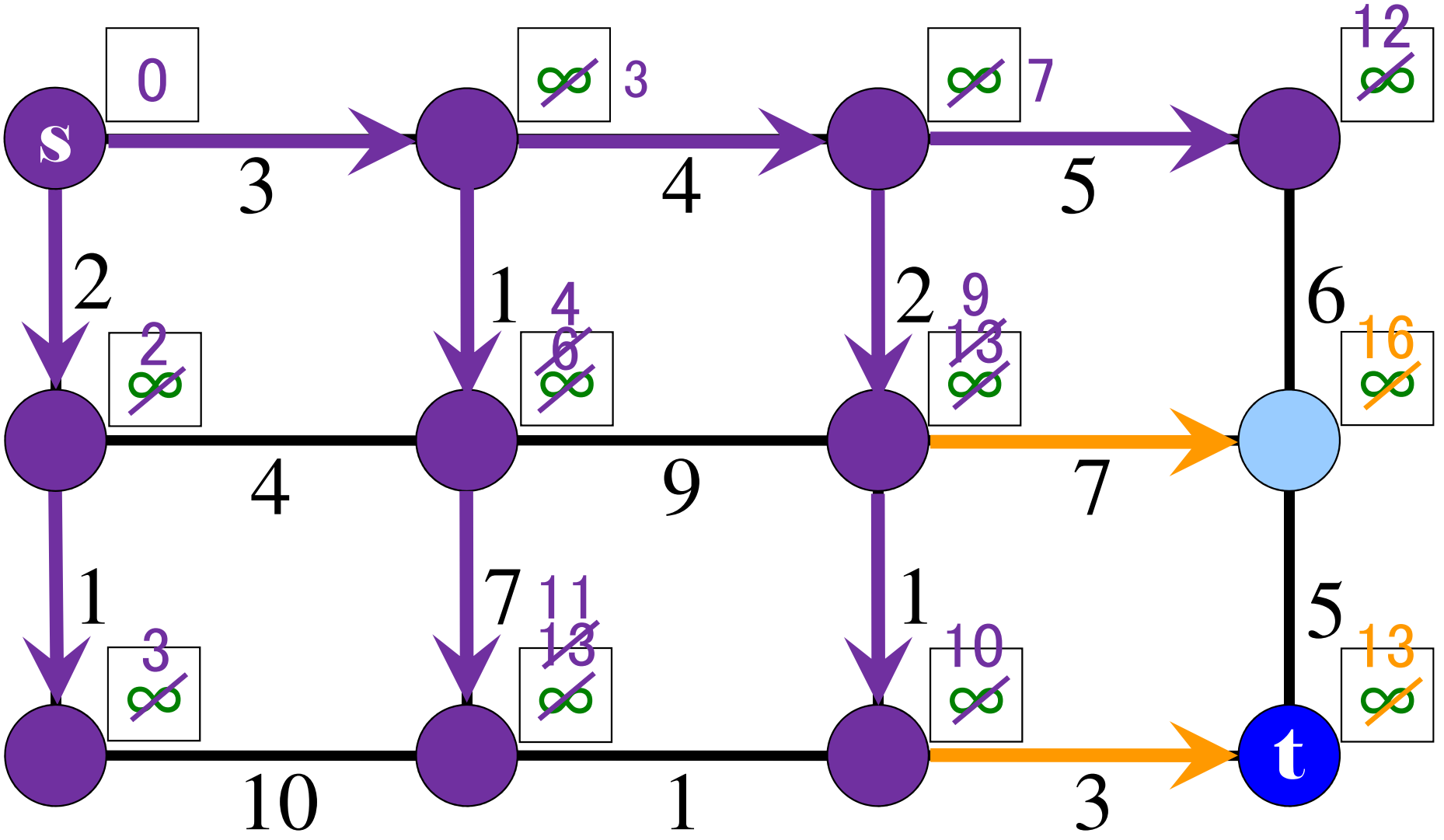
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

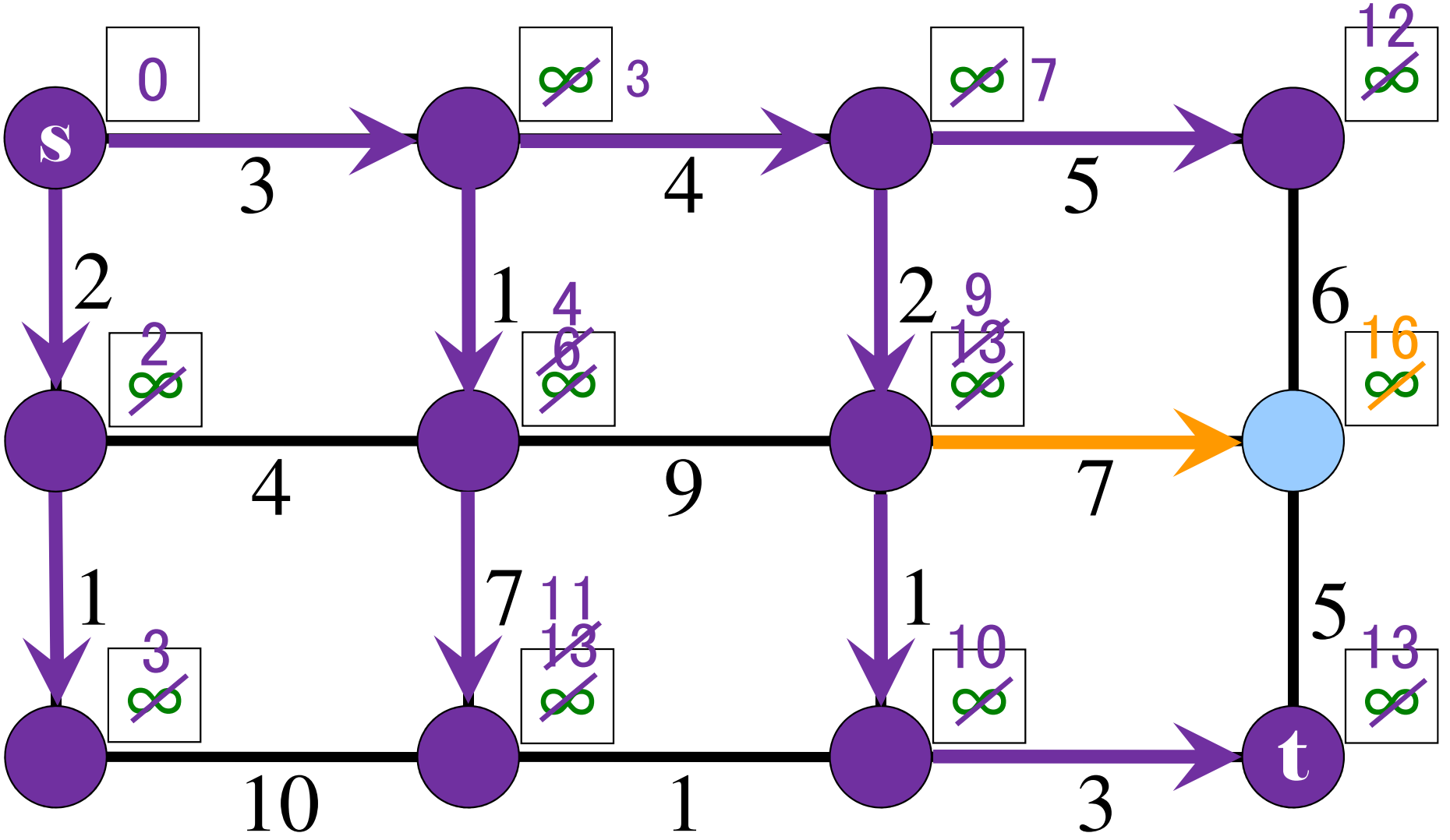


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

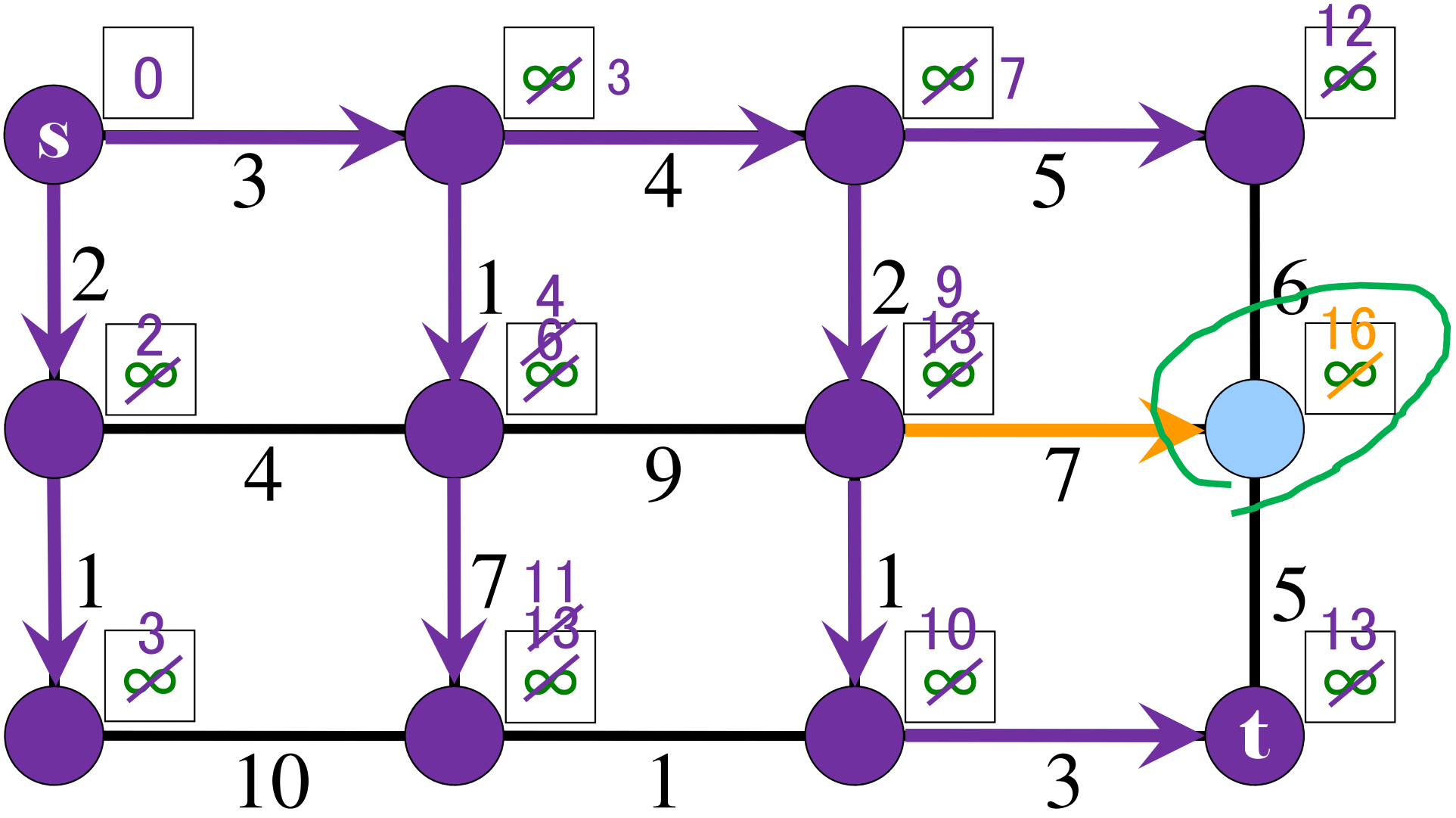
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



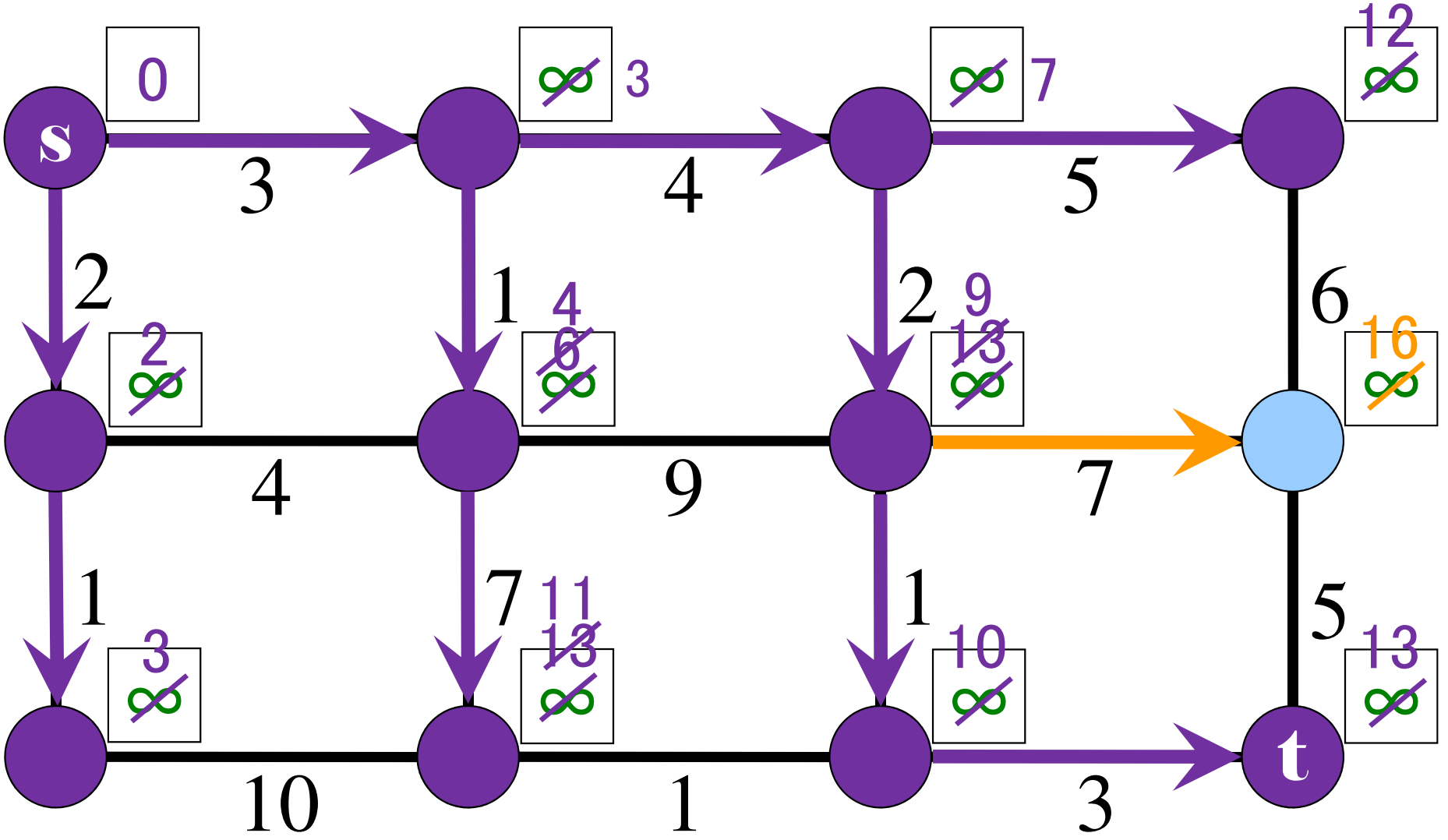
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

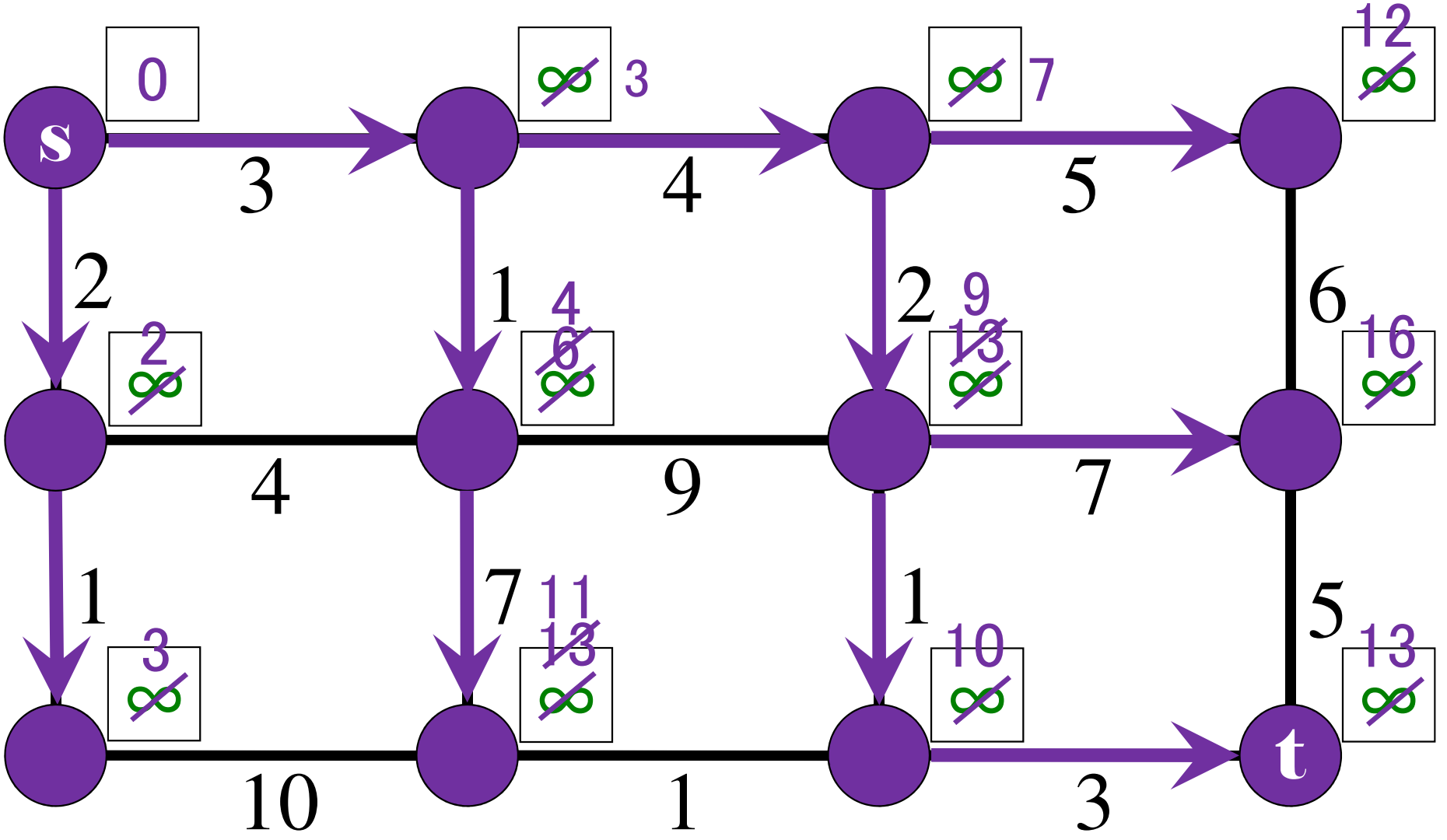


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

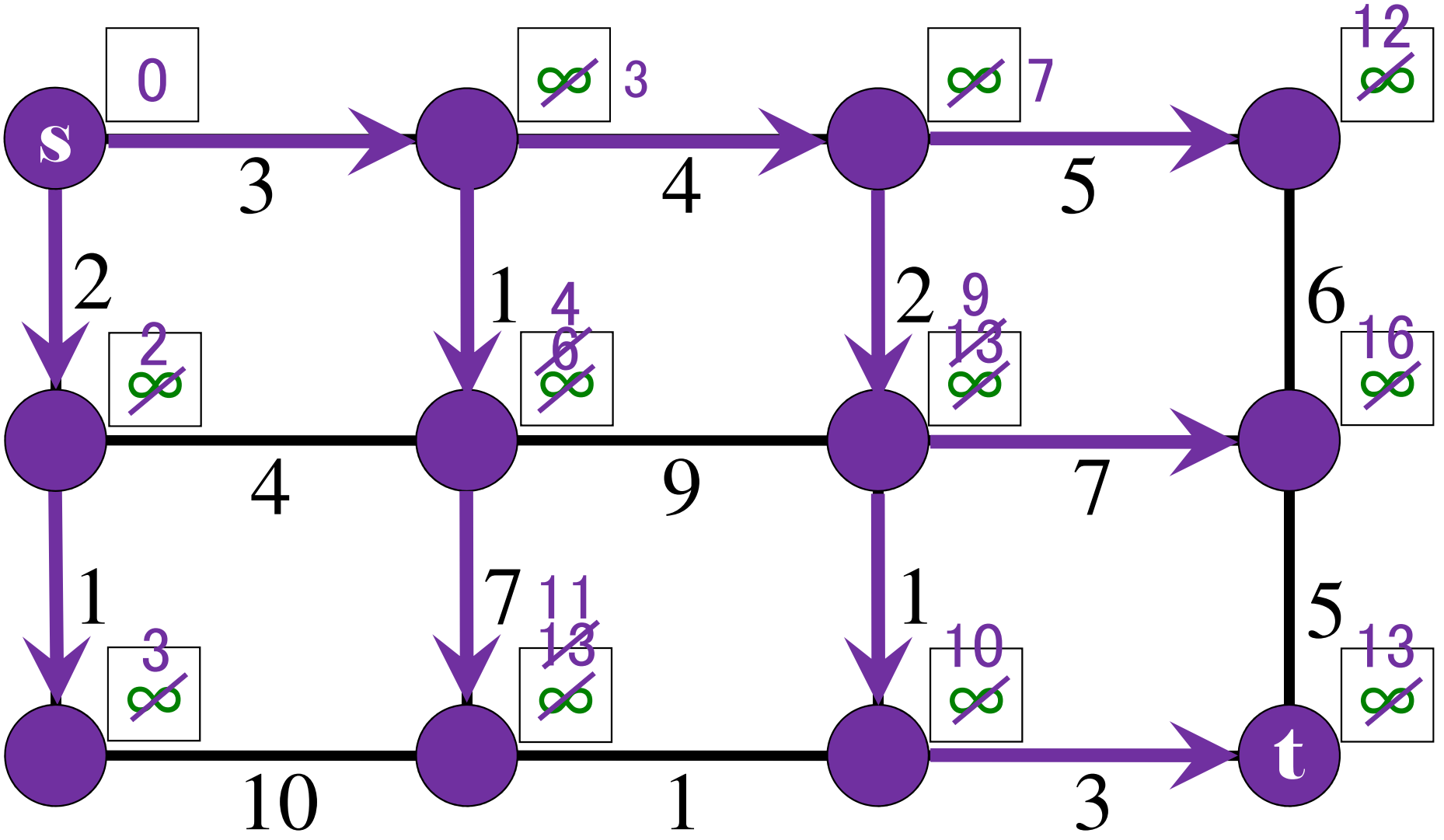
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



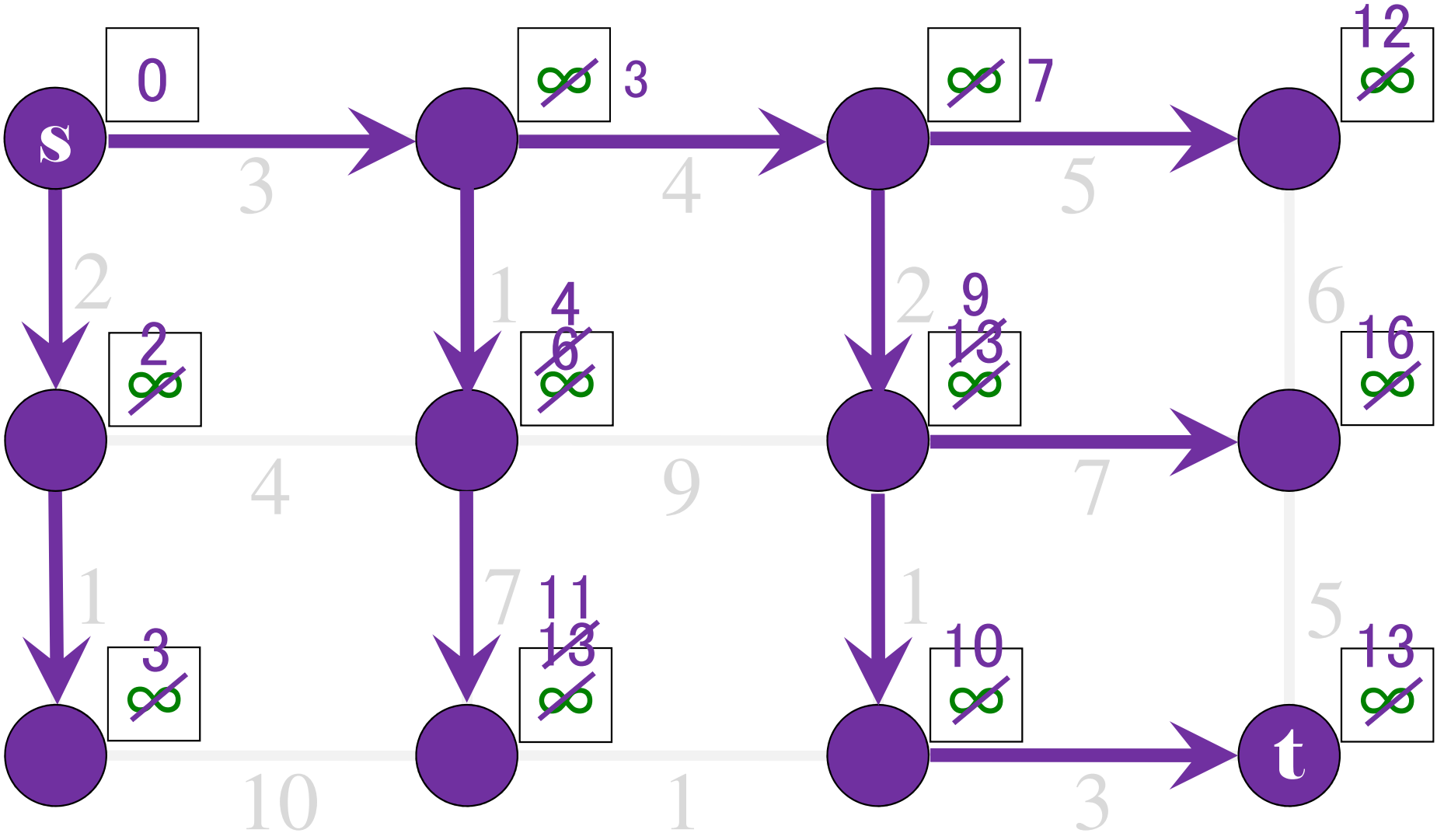
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



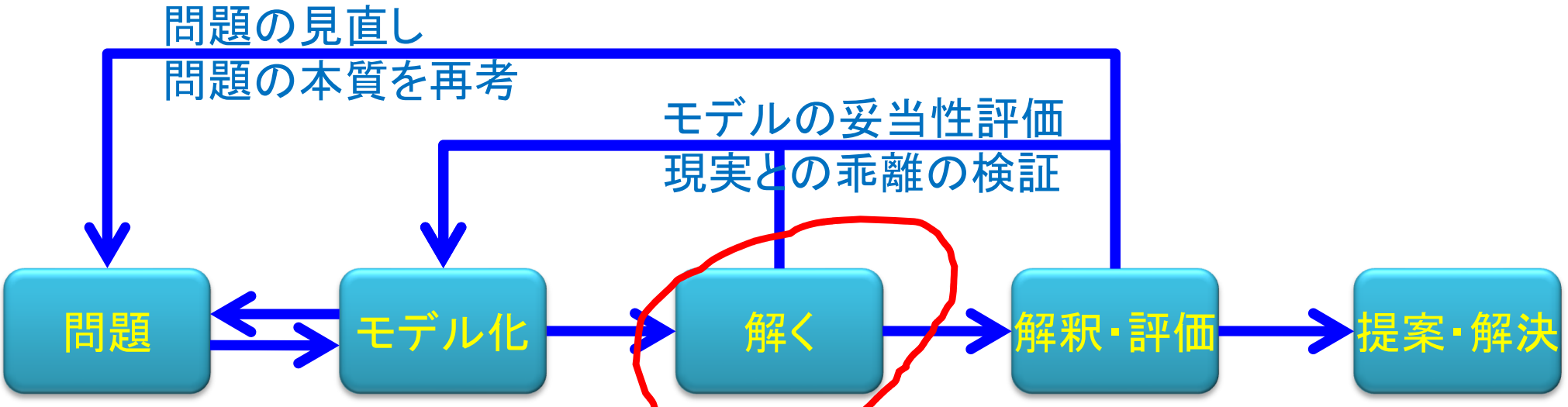
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

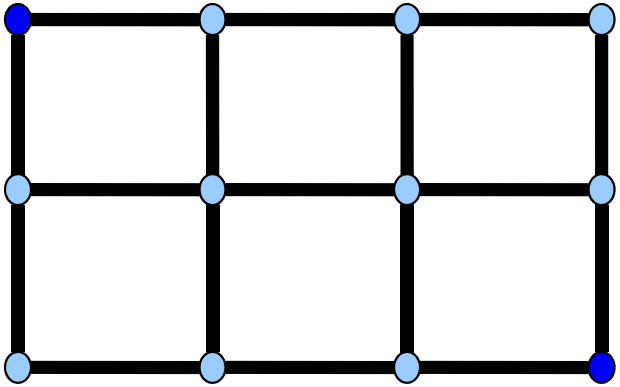
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

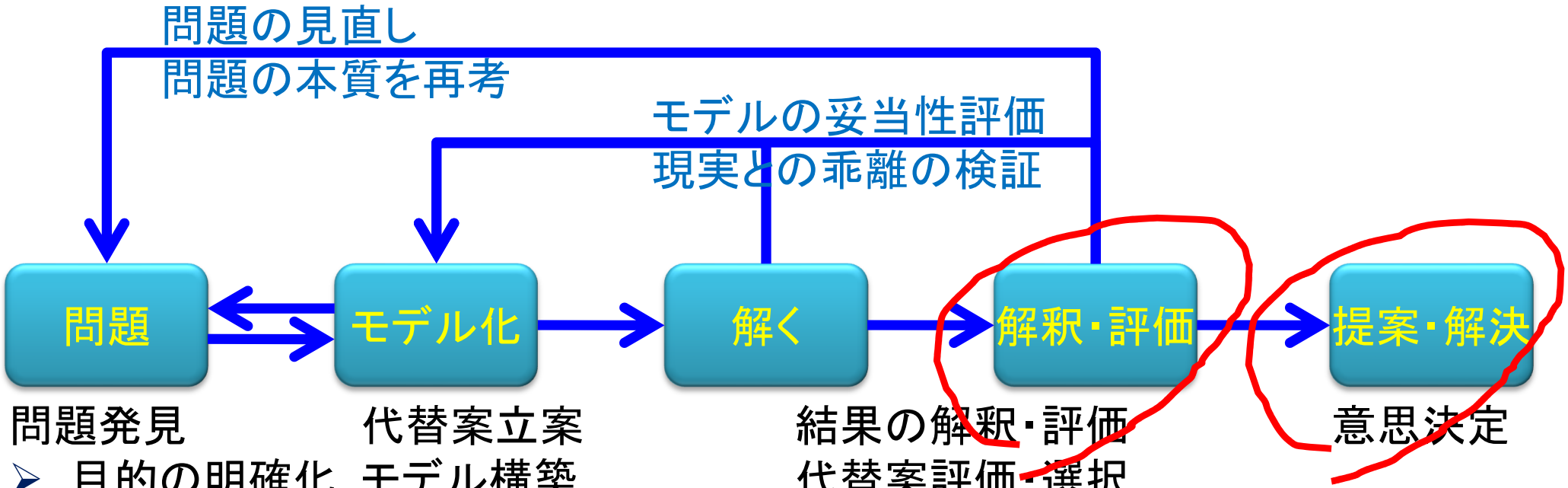
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

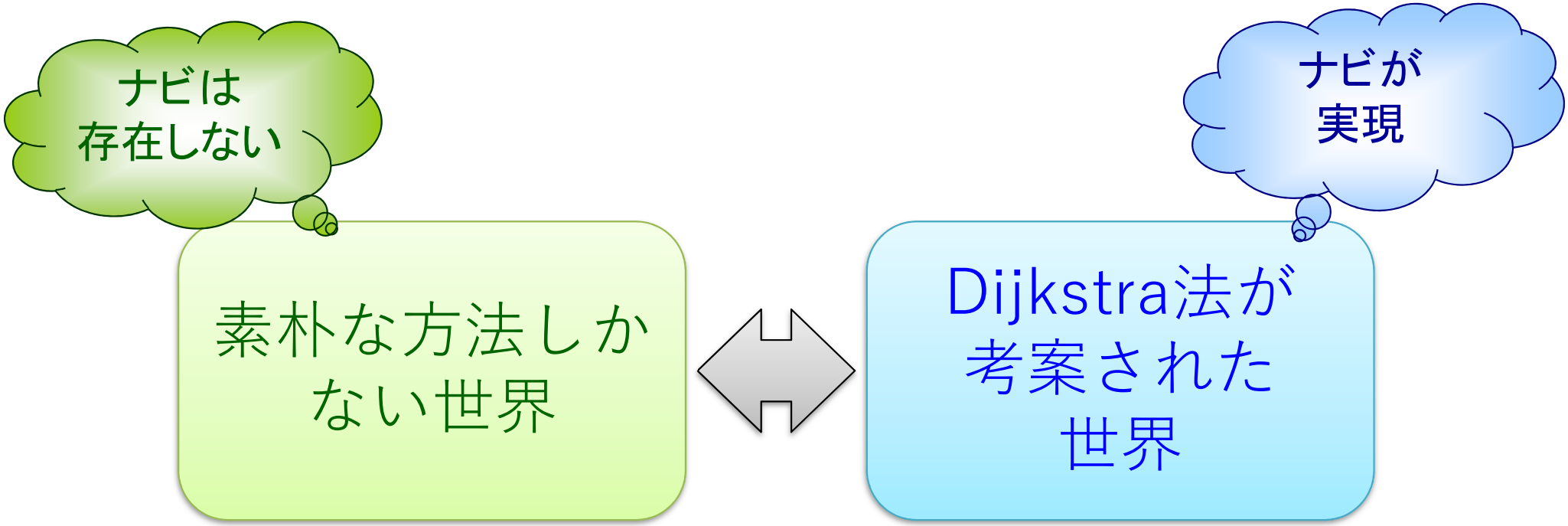
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

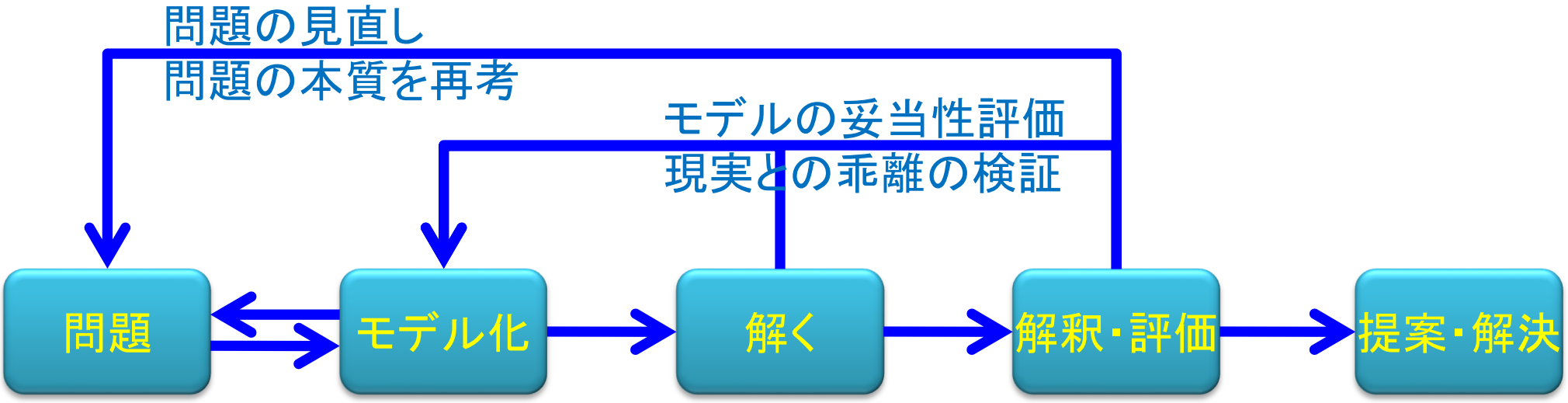
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



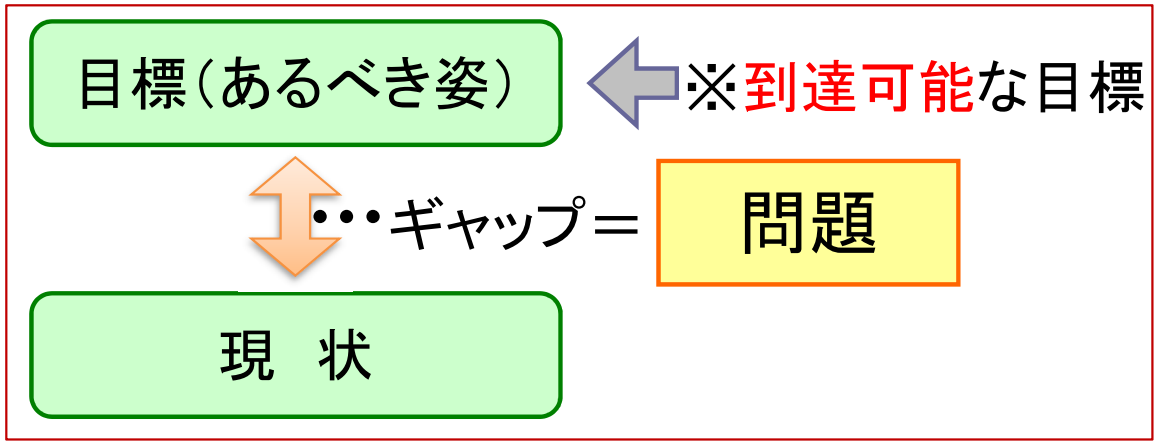
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

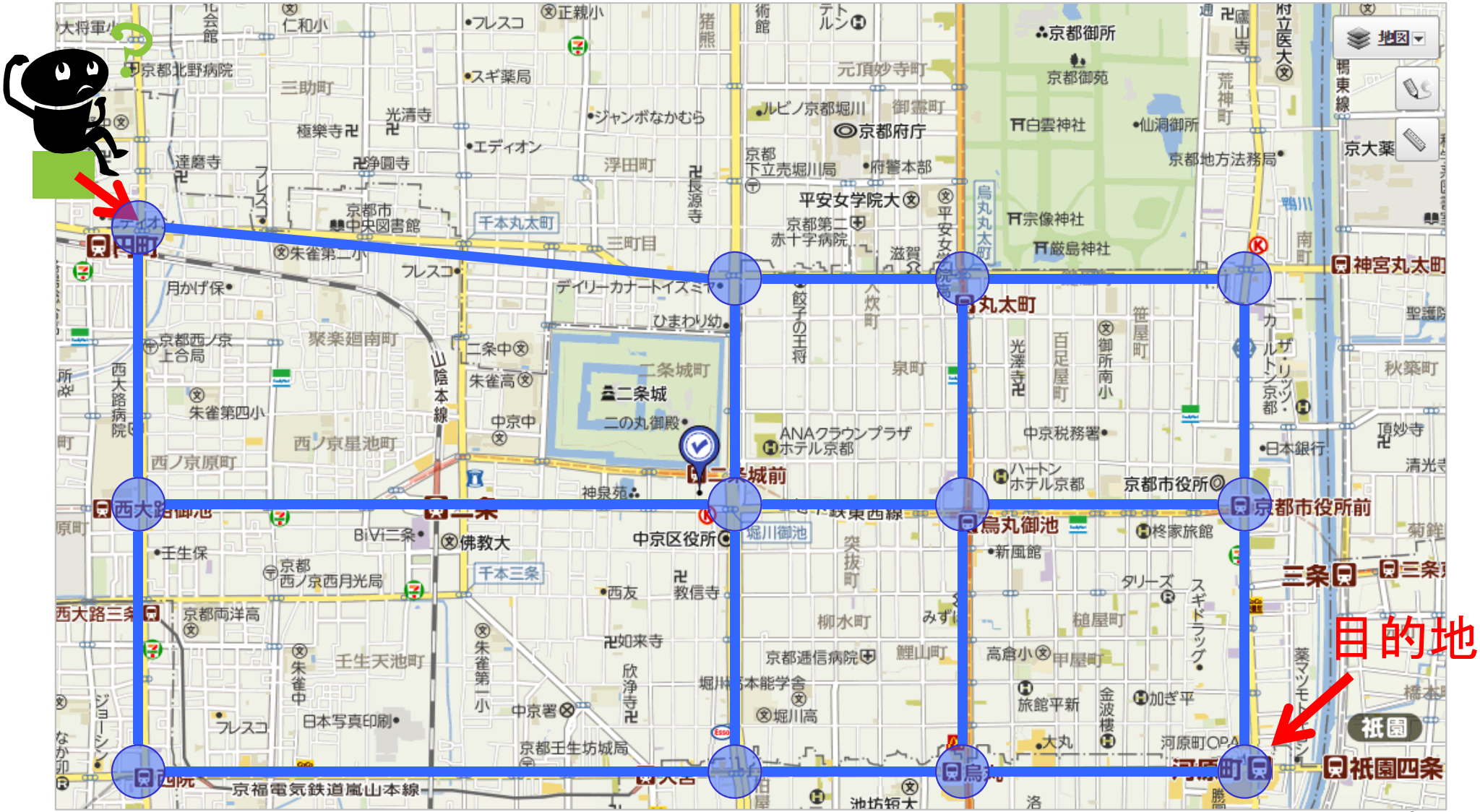
意思決定

問題の定義



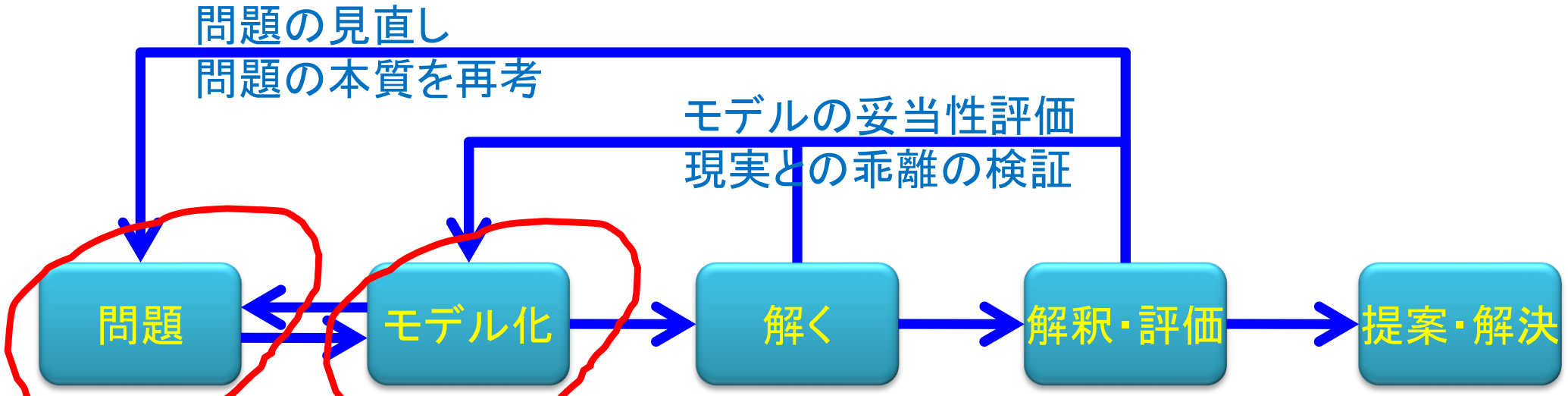
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

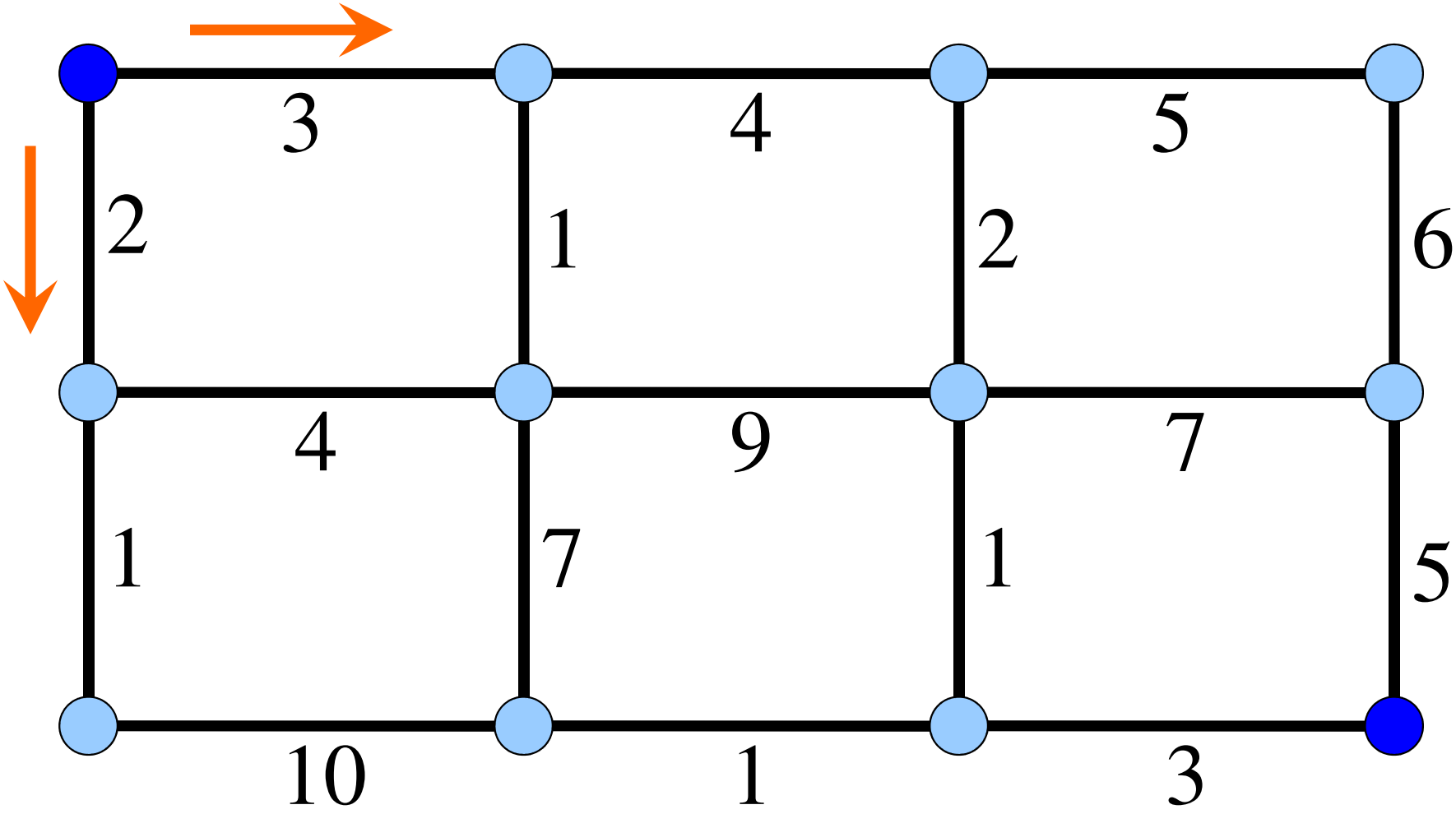
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

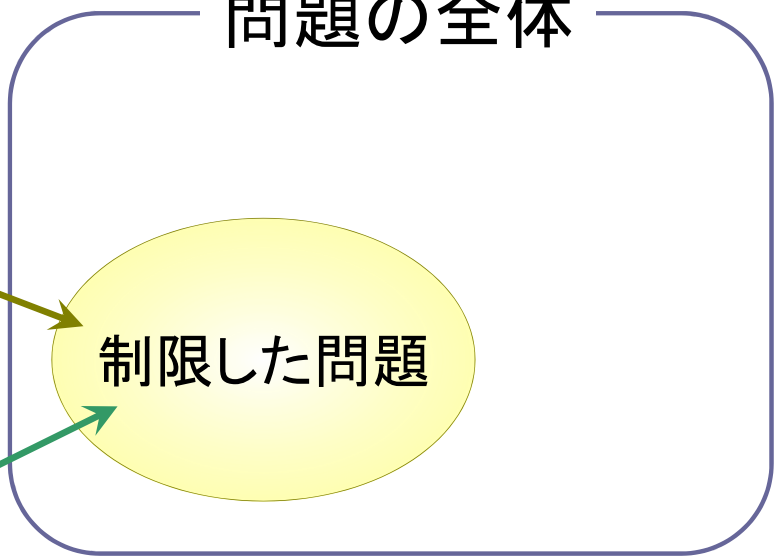


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

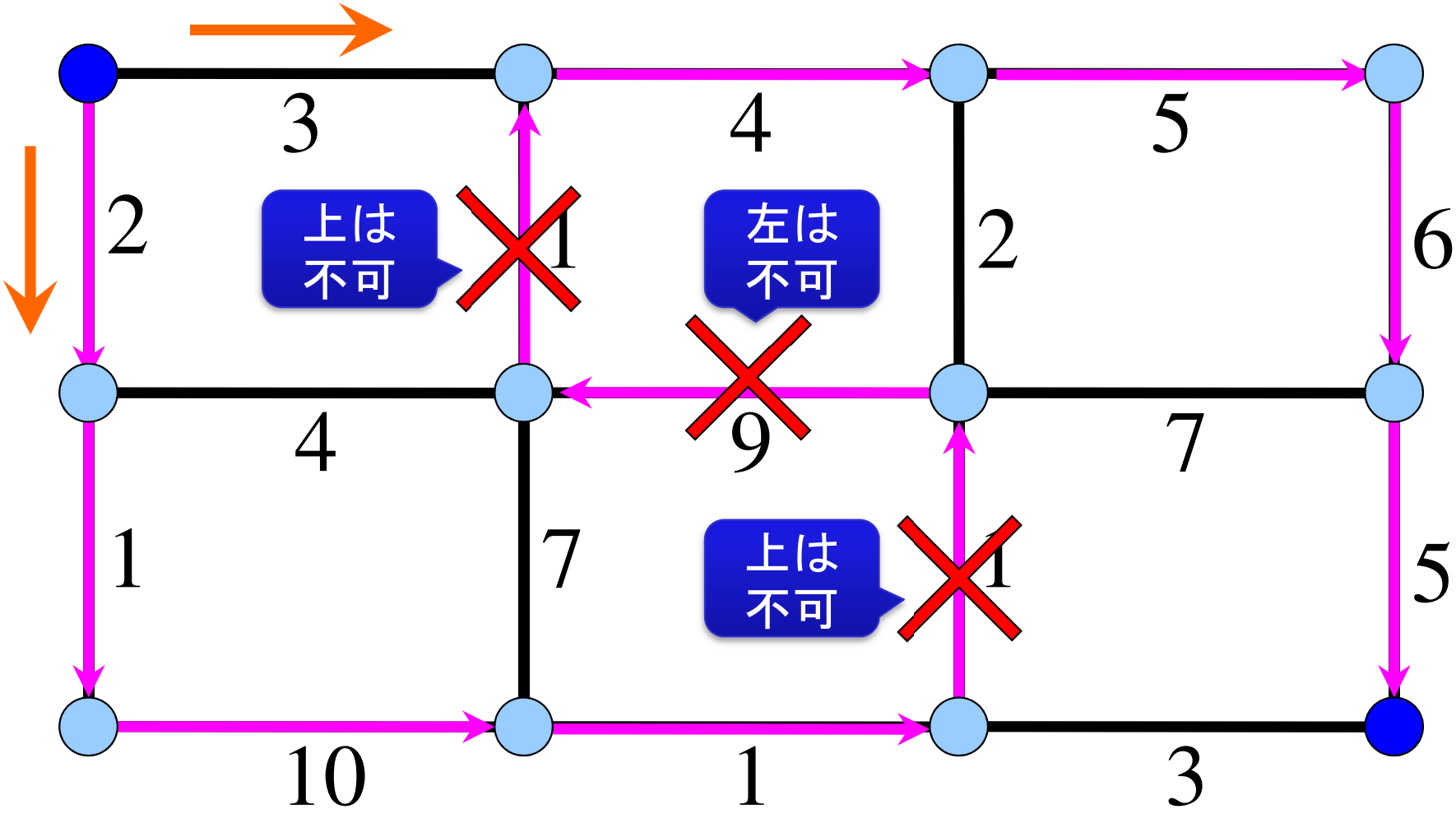
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

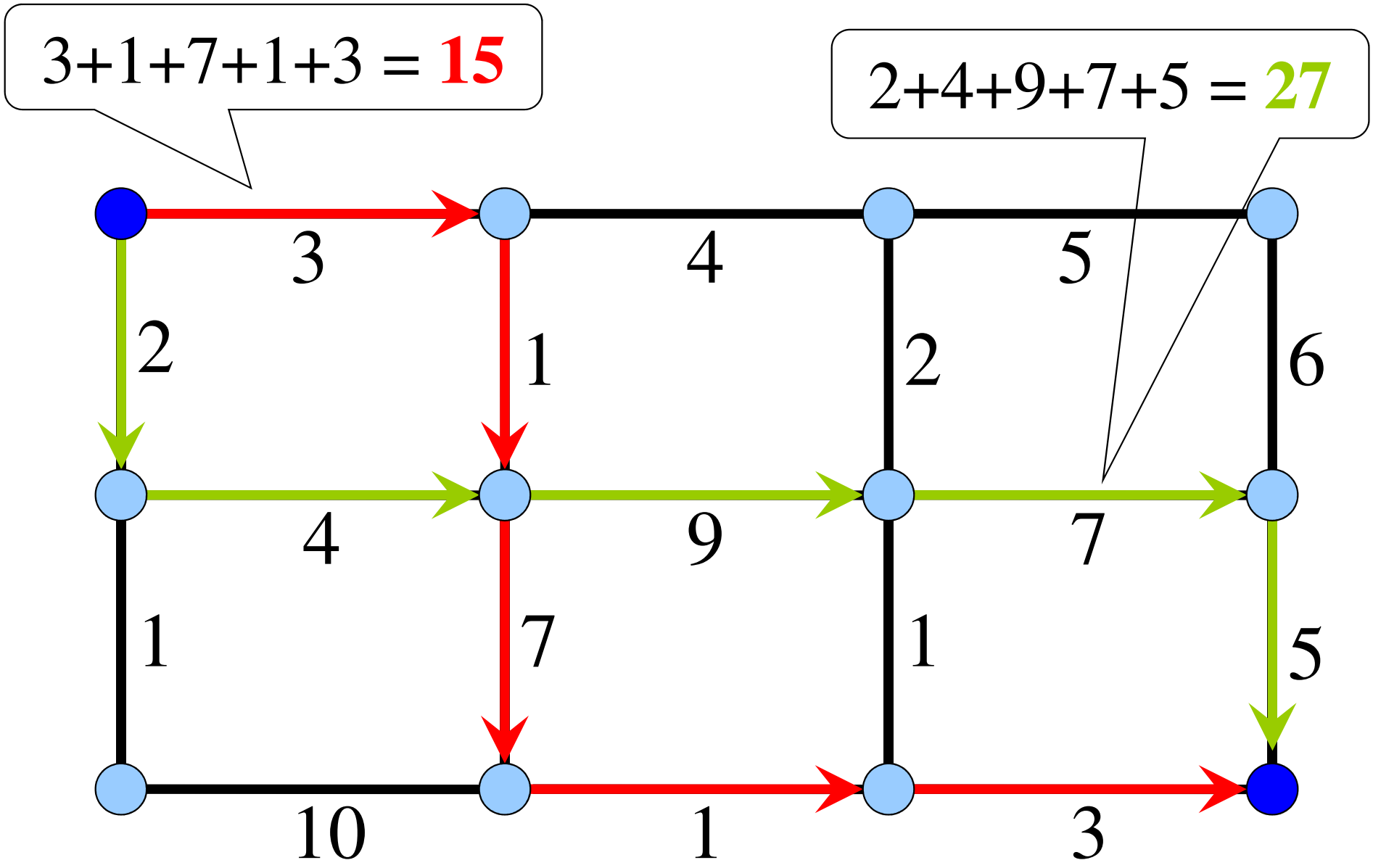
難しいなら易しくすればいいのさ！

制限した問題

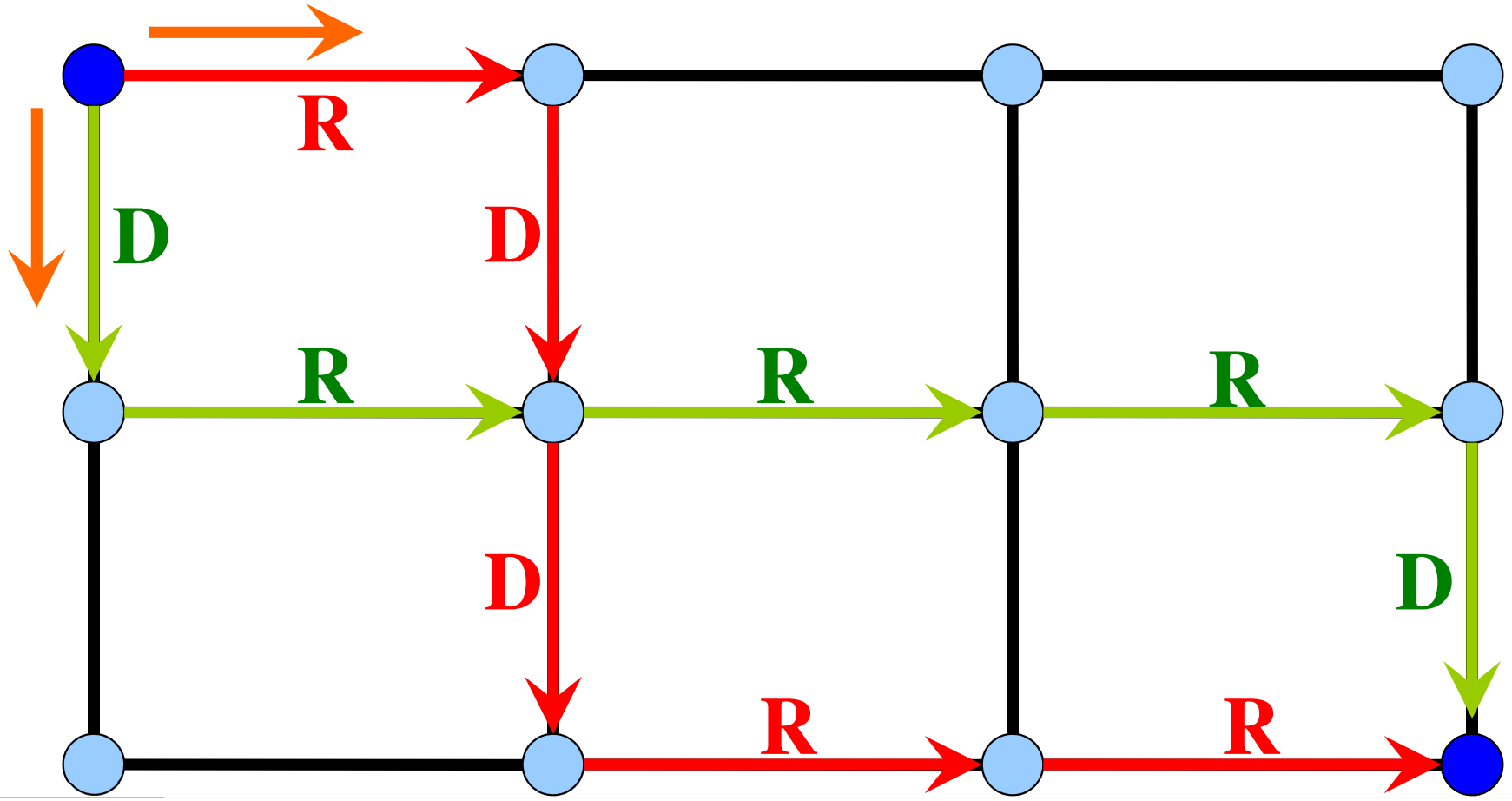
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



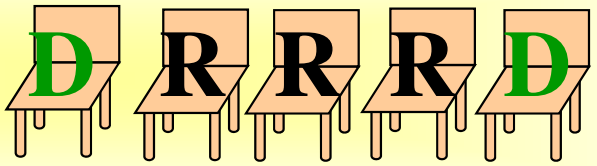
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



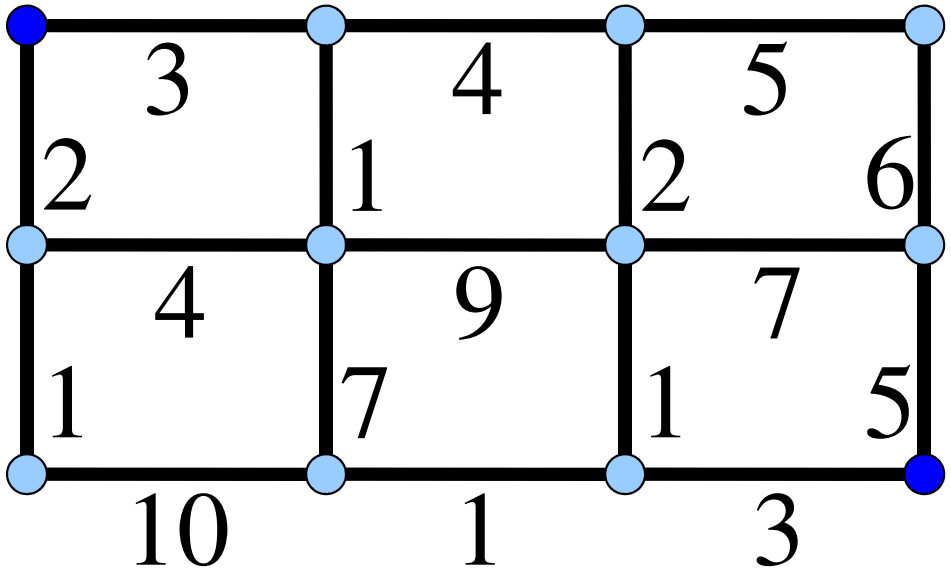
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

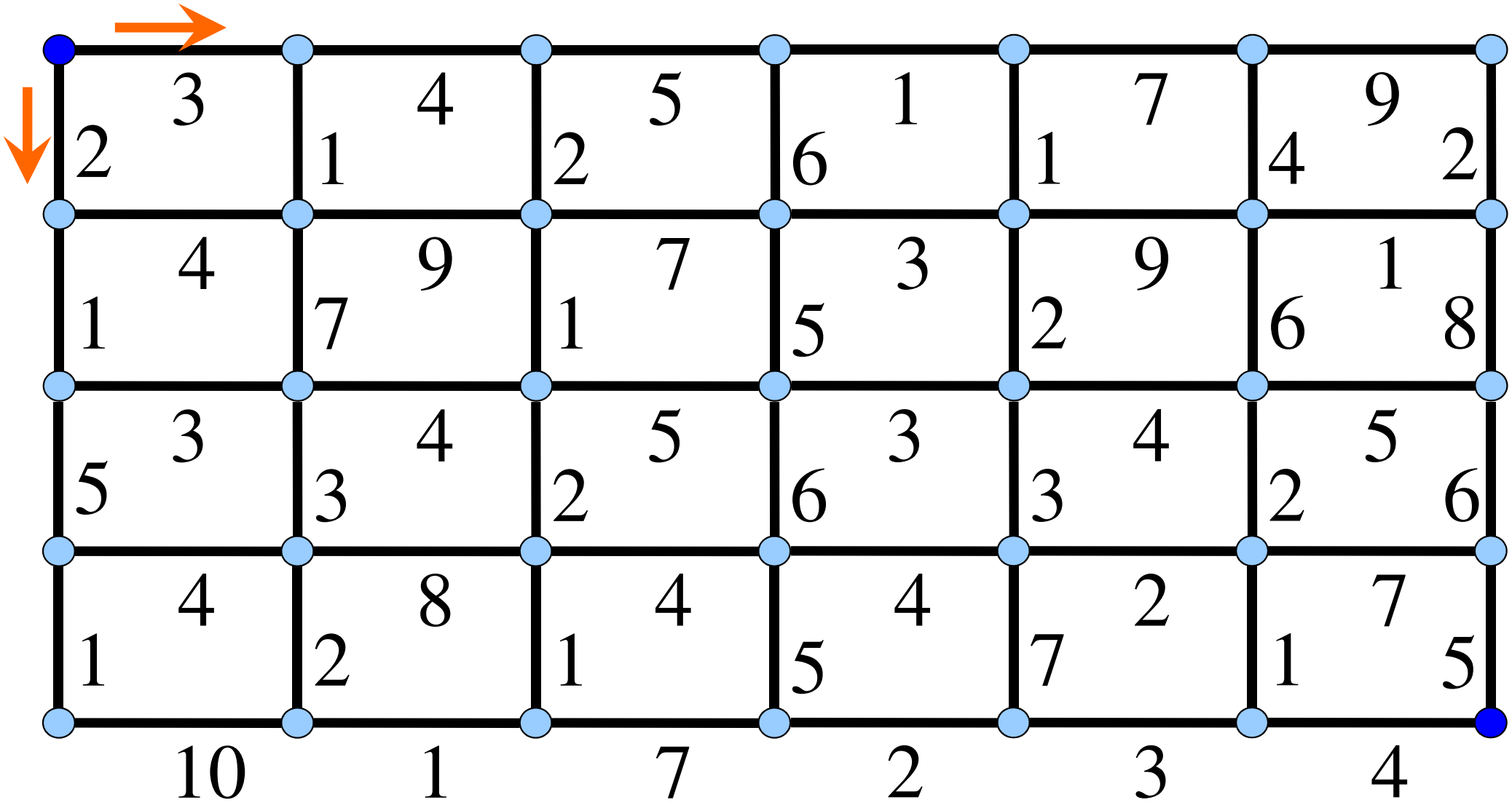
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

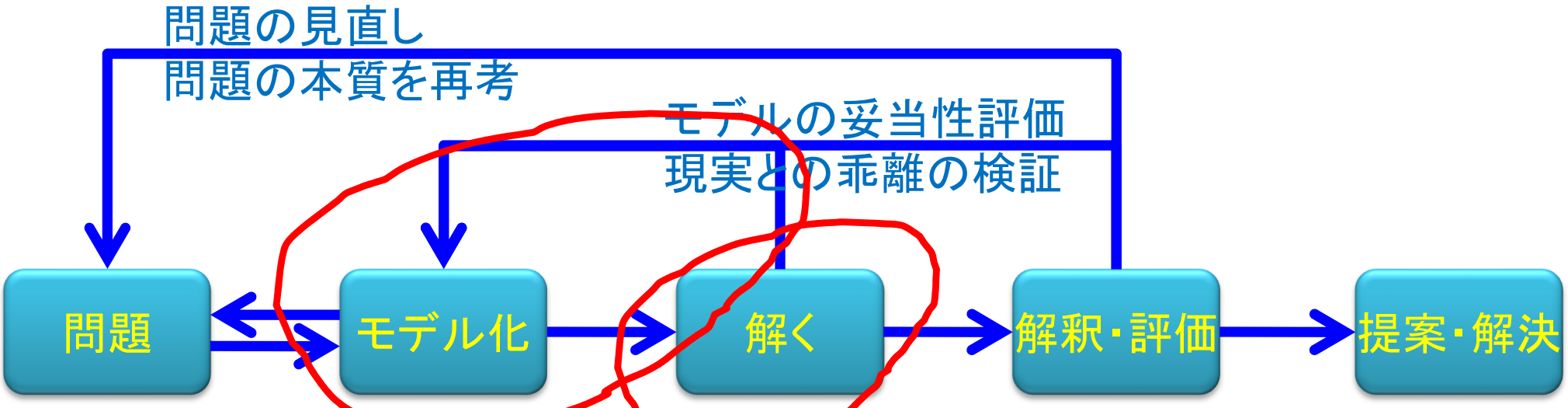
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

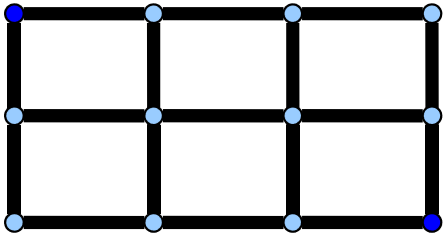
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

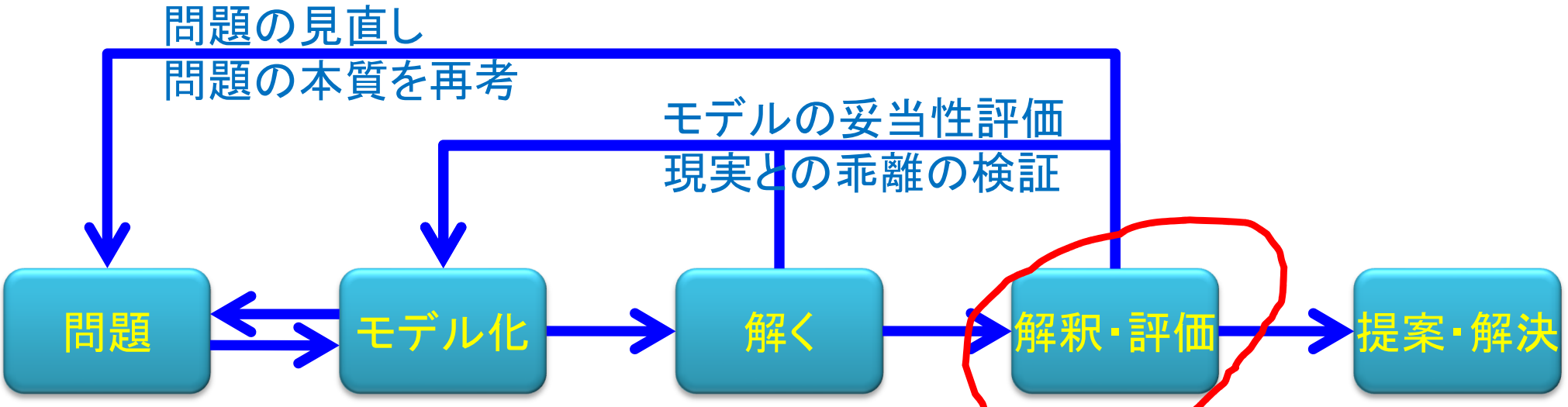
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

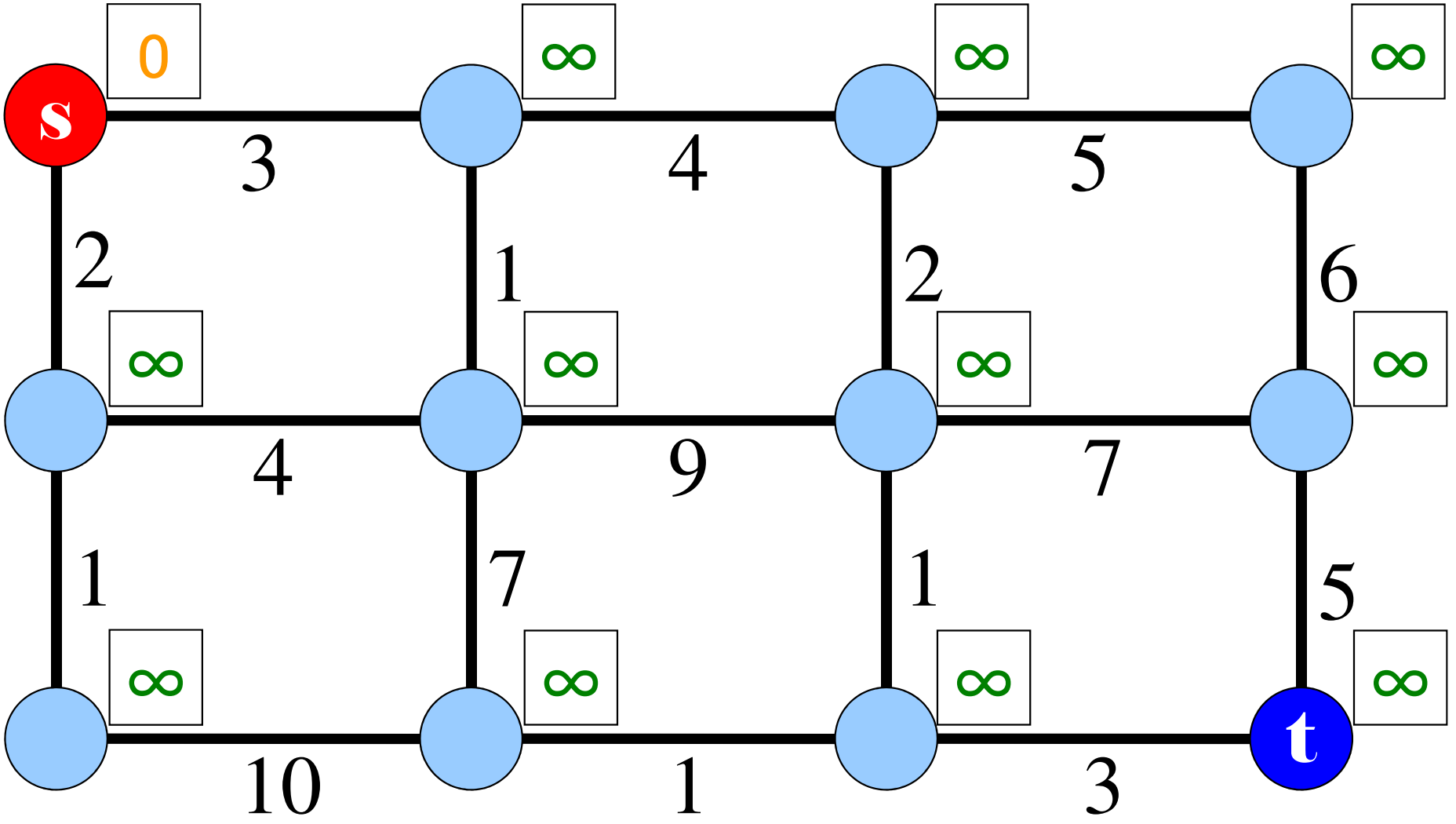
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

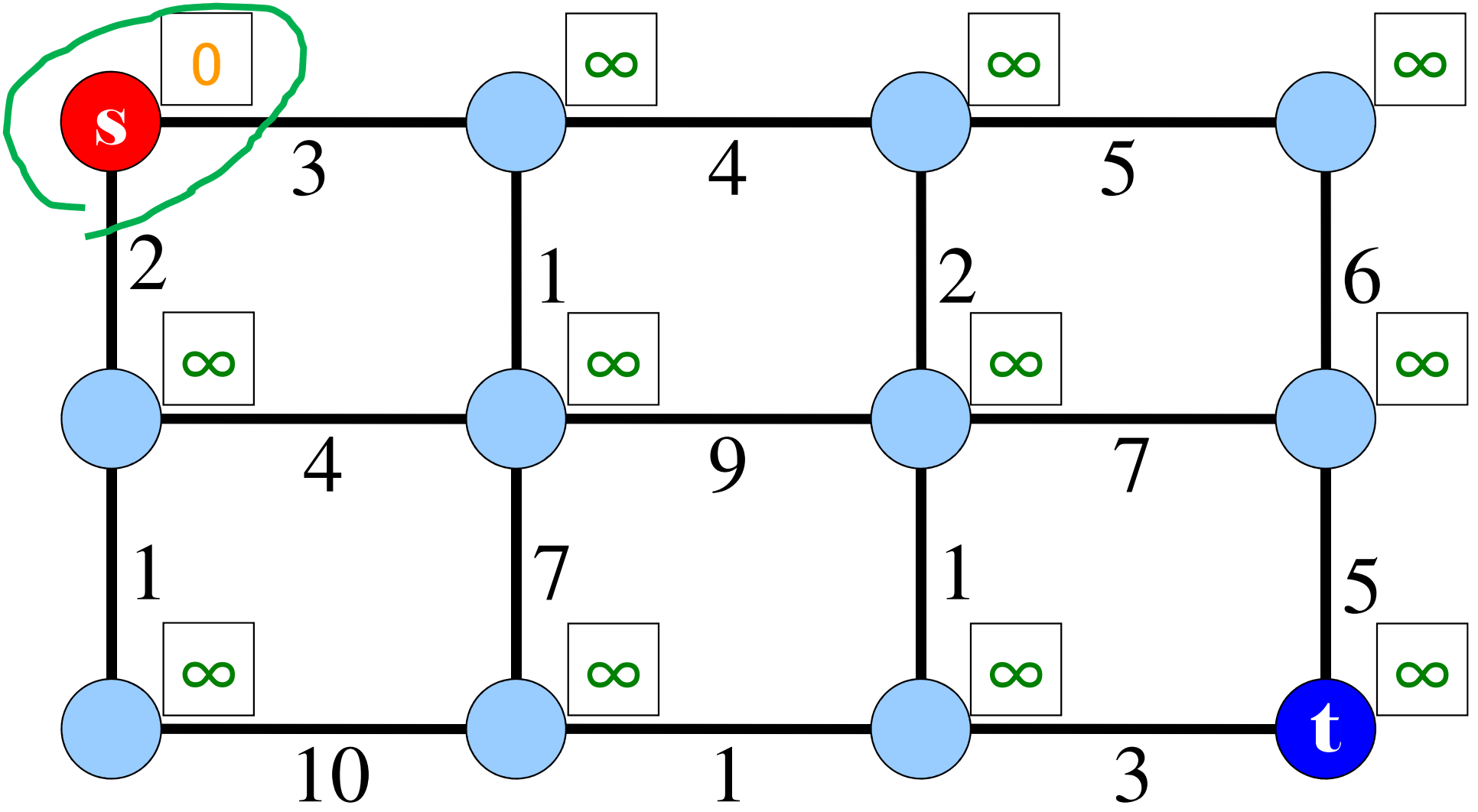
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



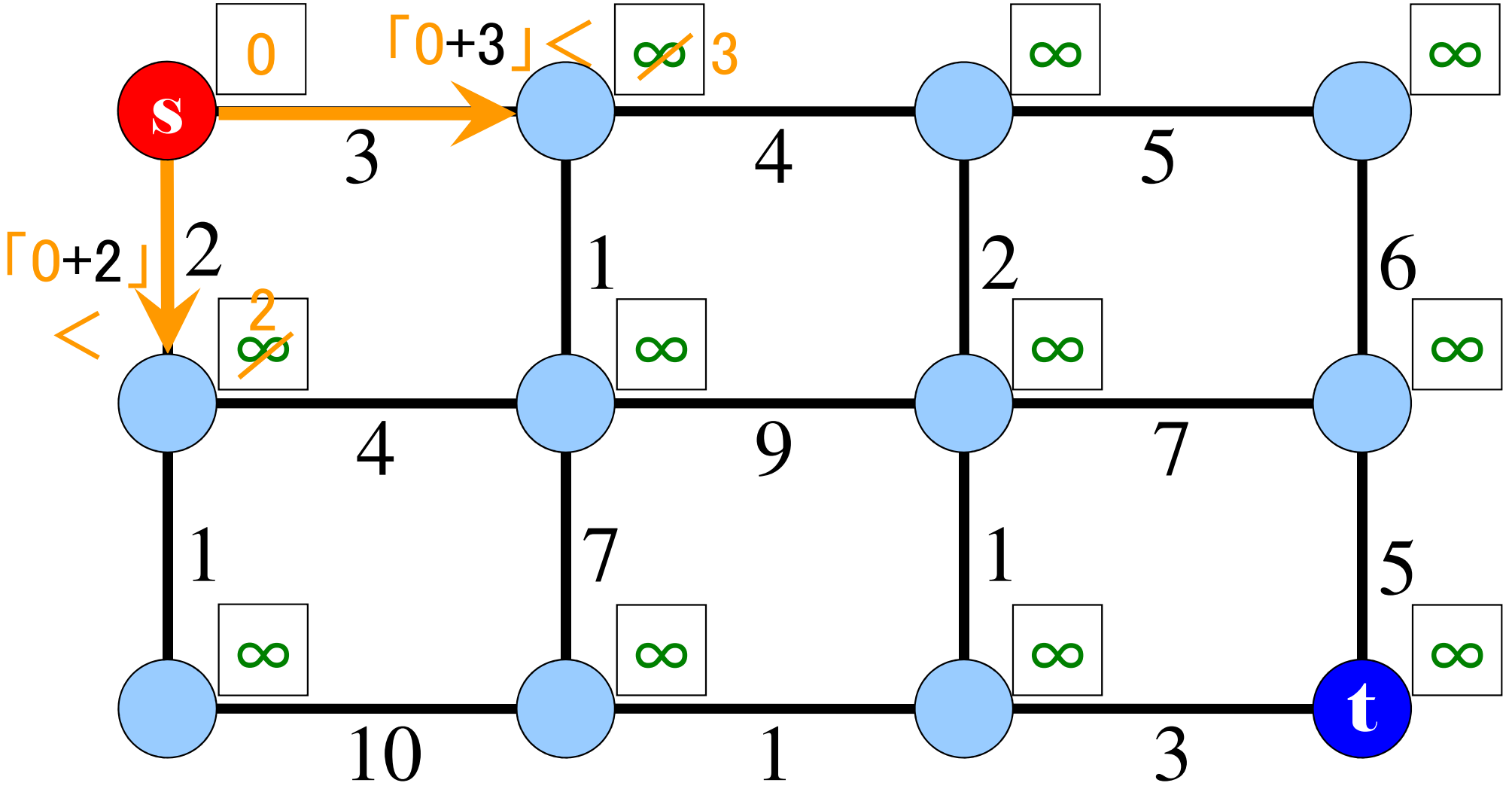
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

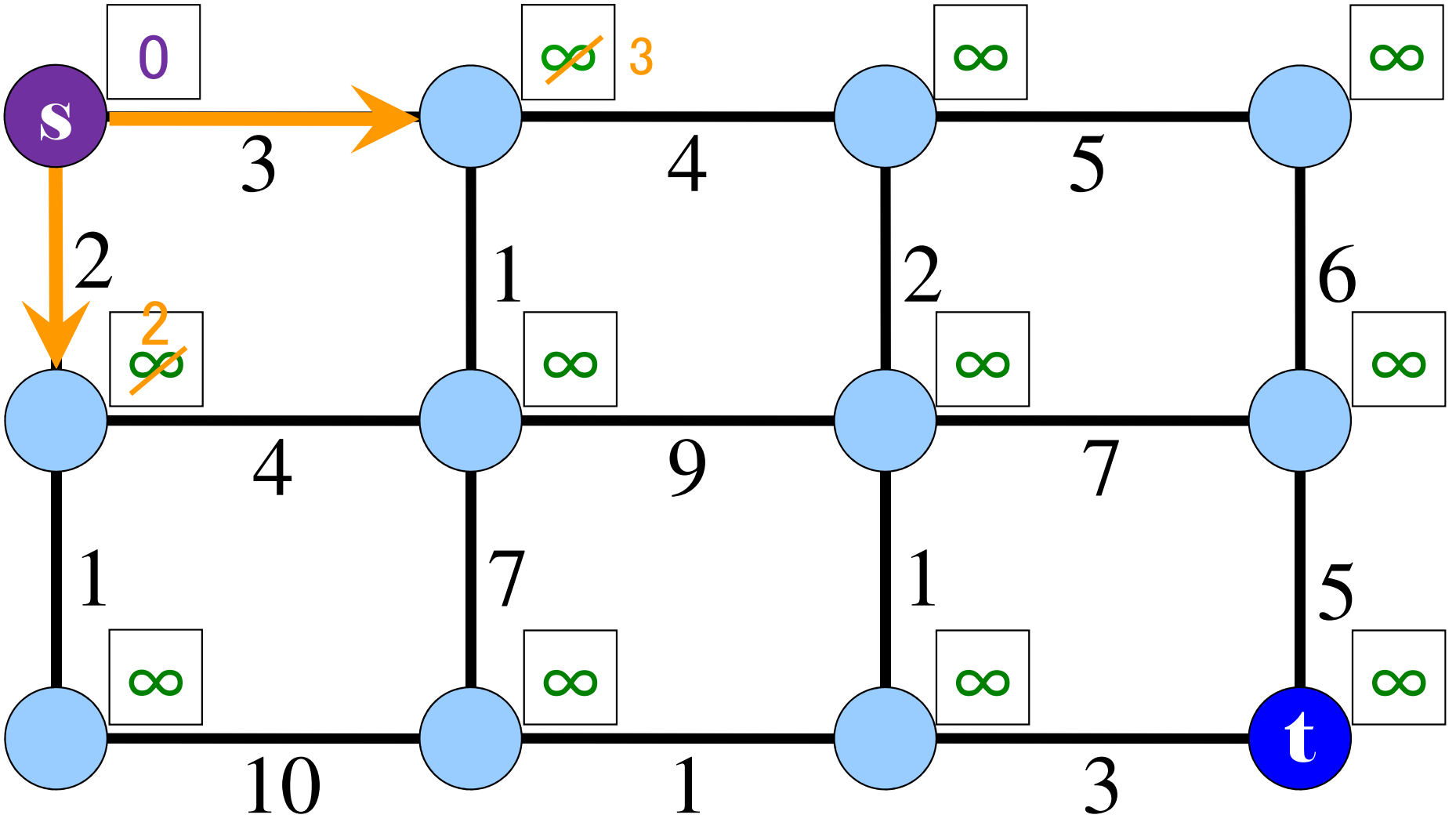


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

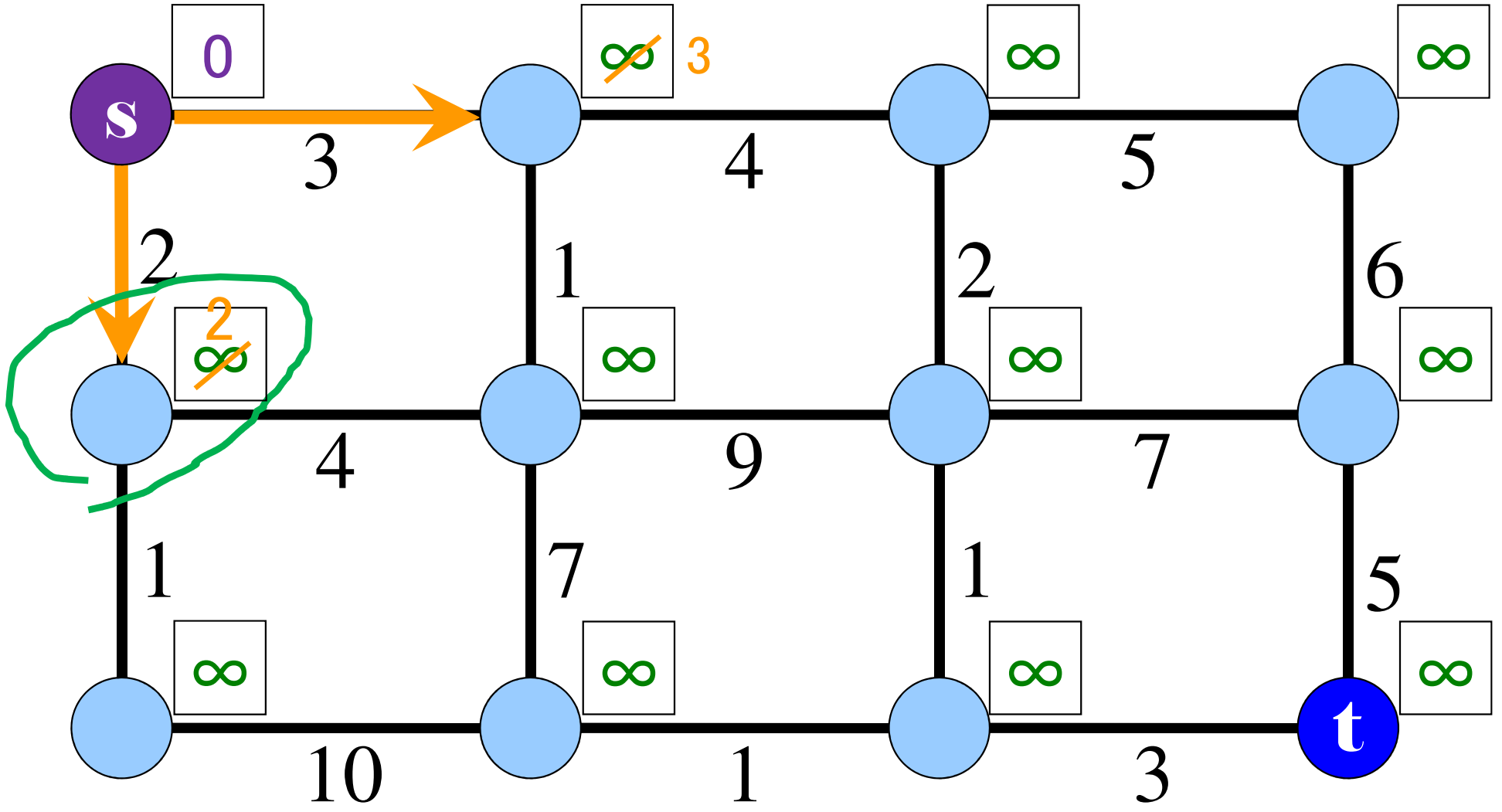
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



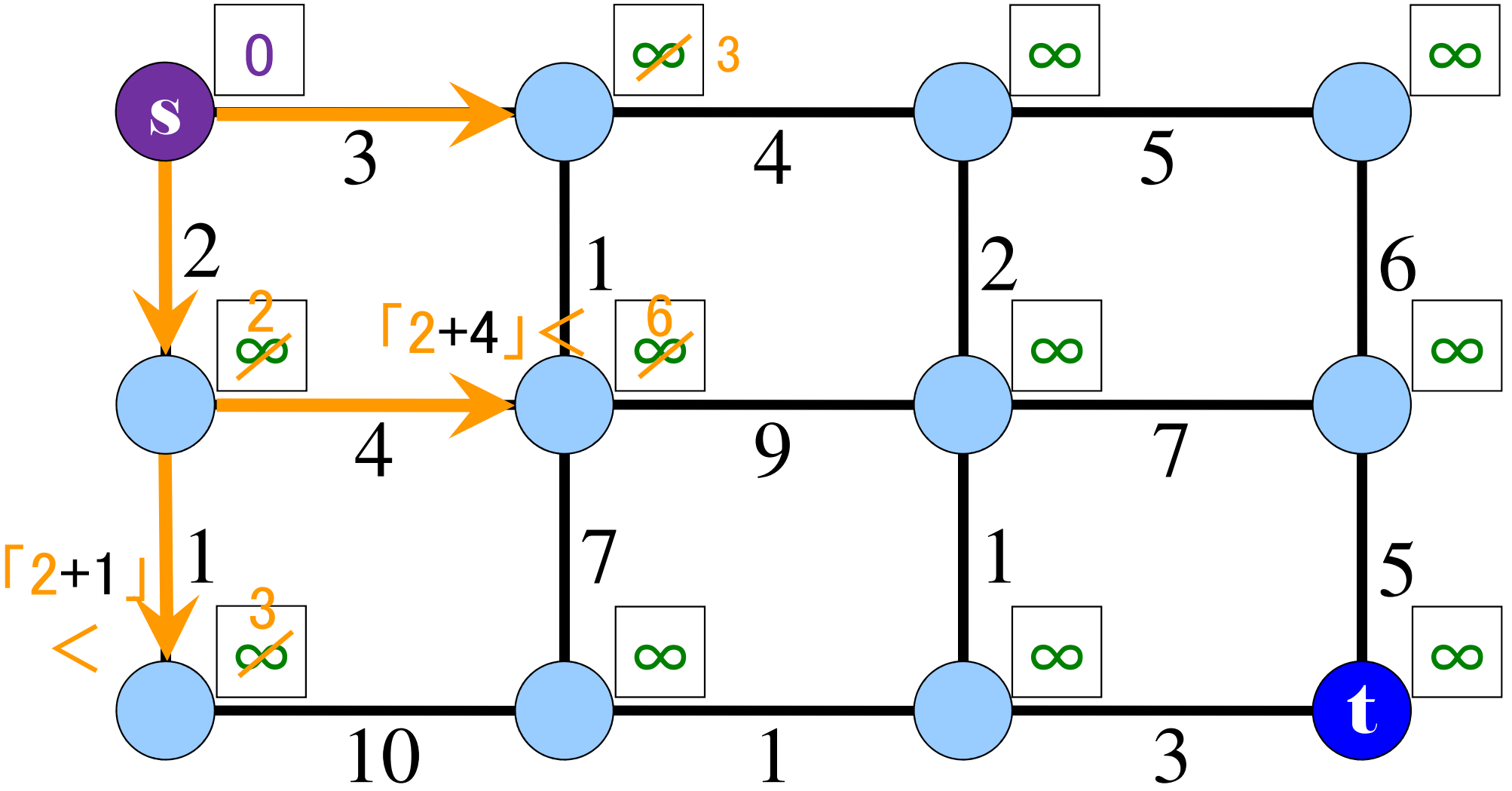
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

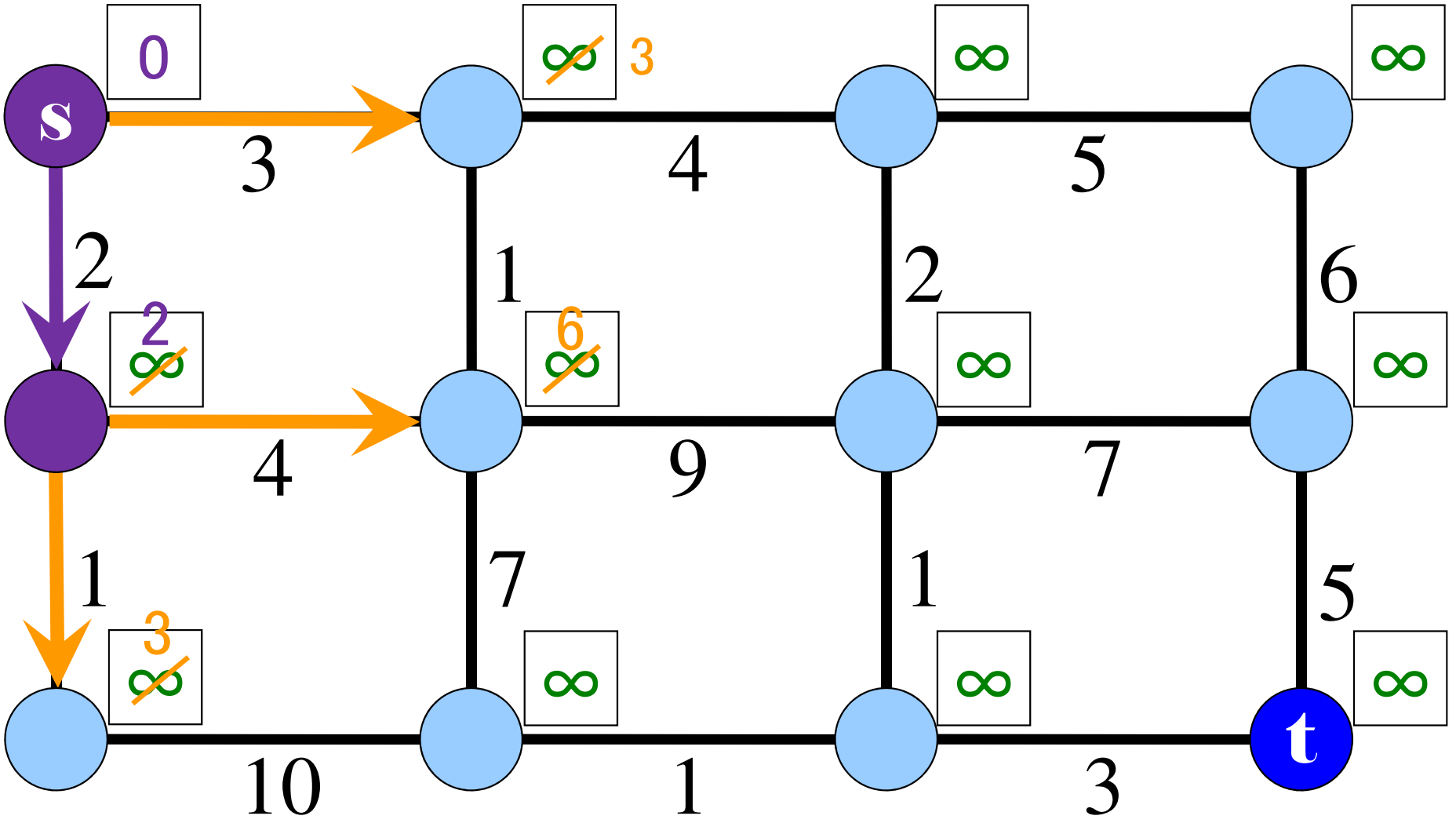


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

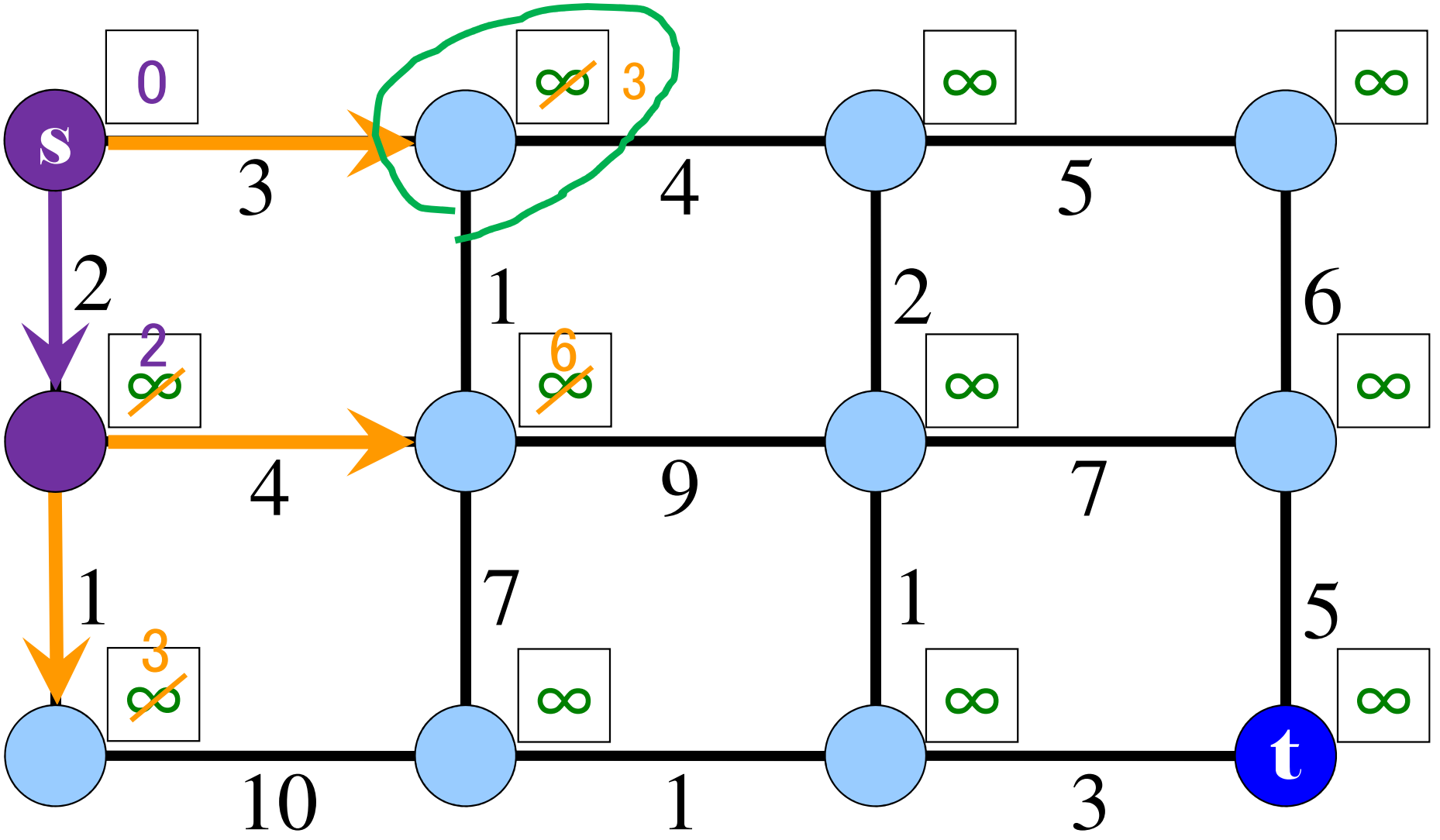
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



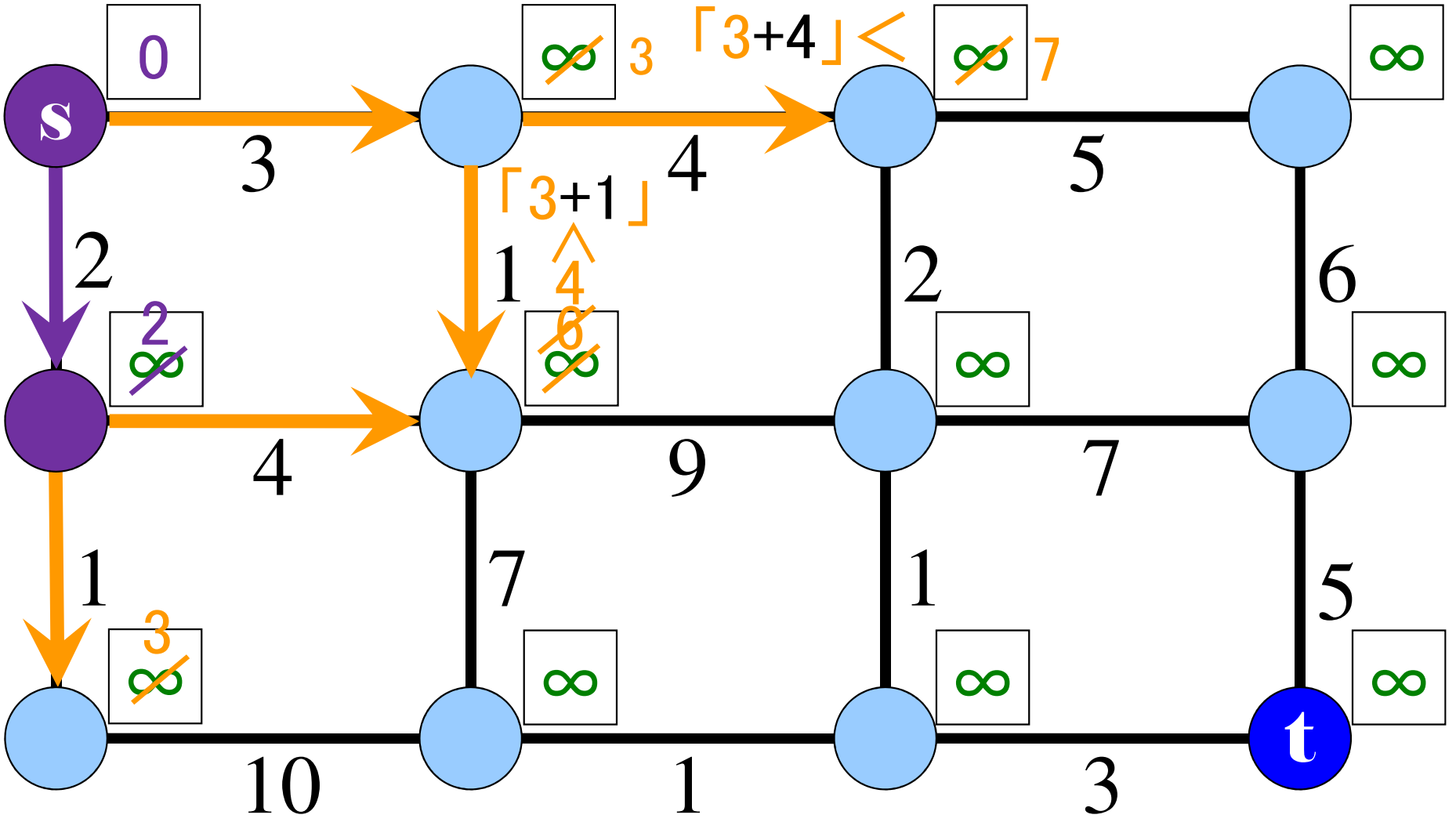
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

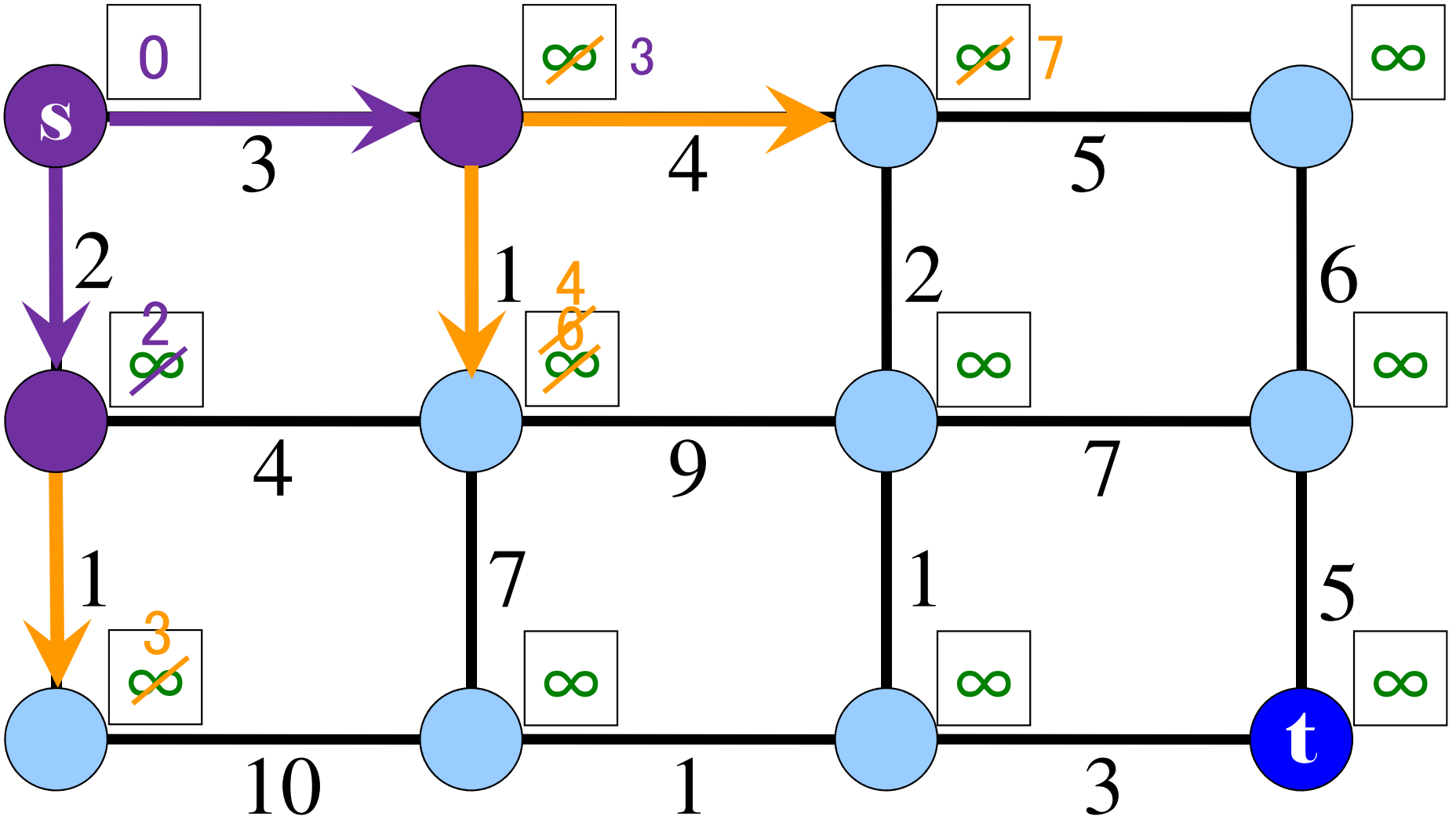


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

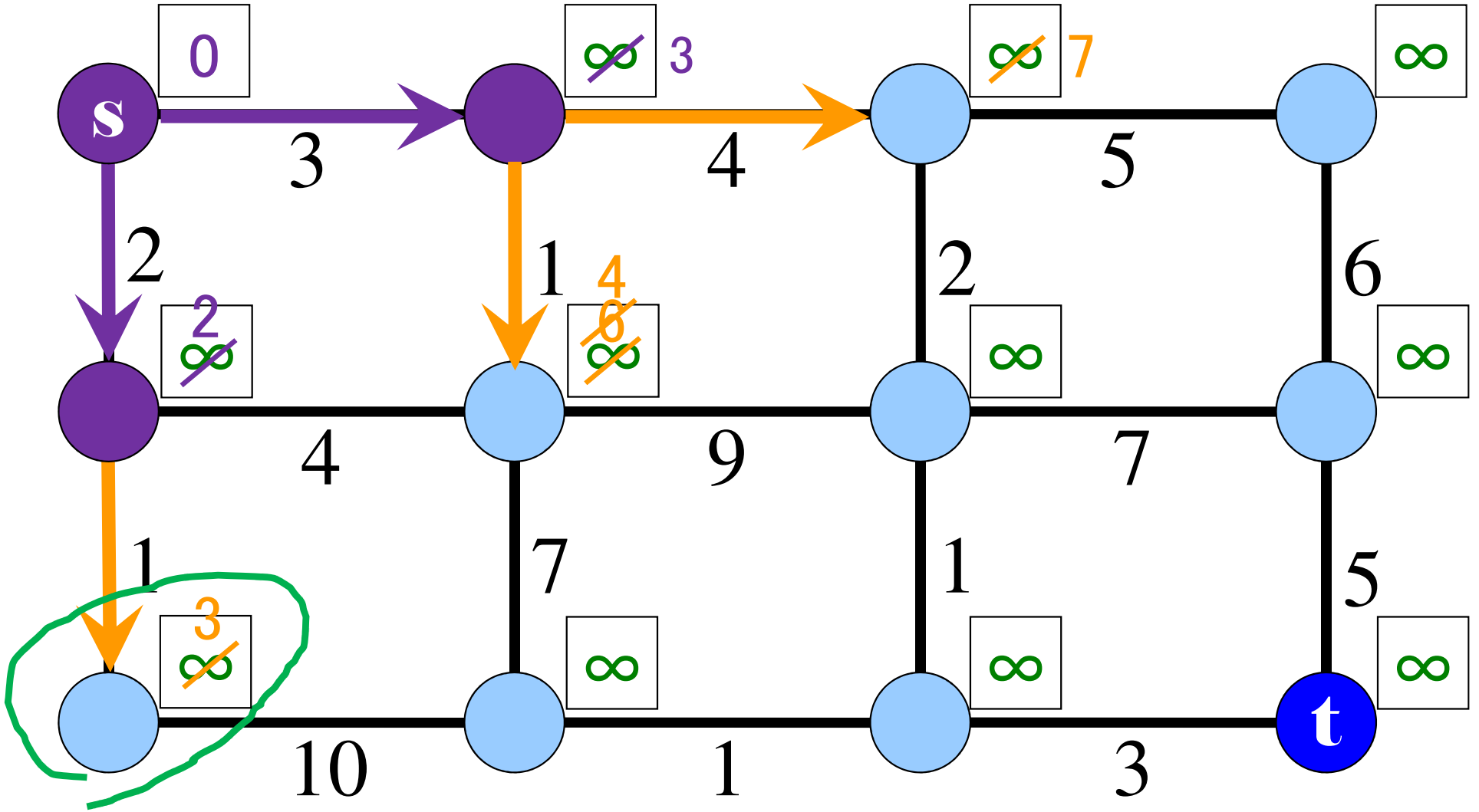
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



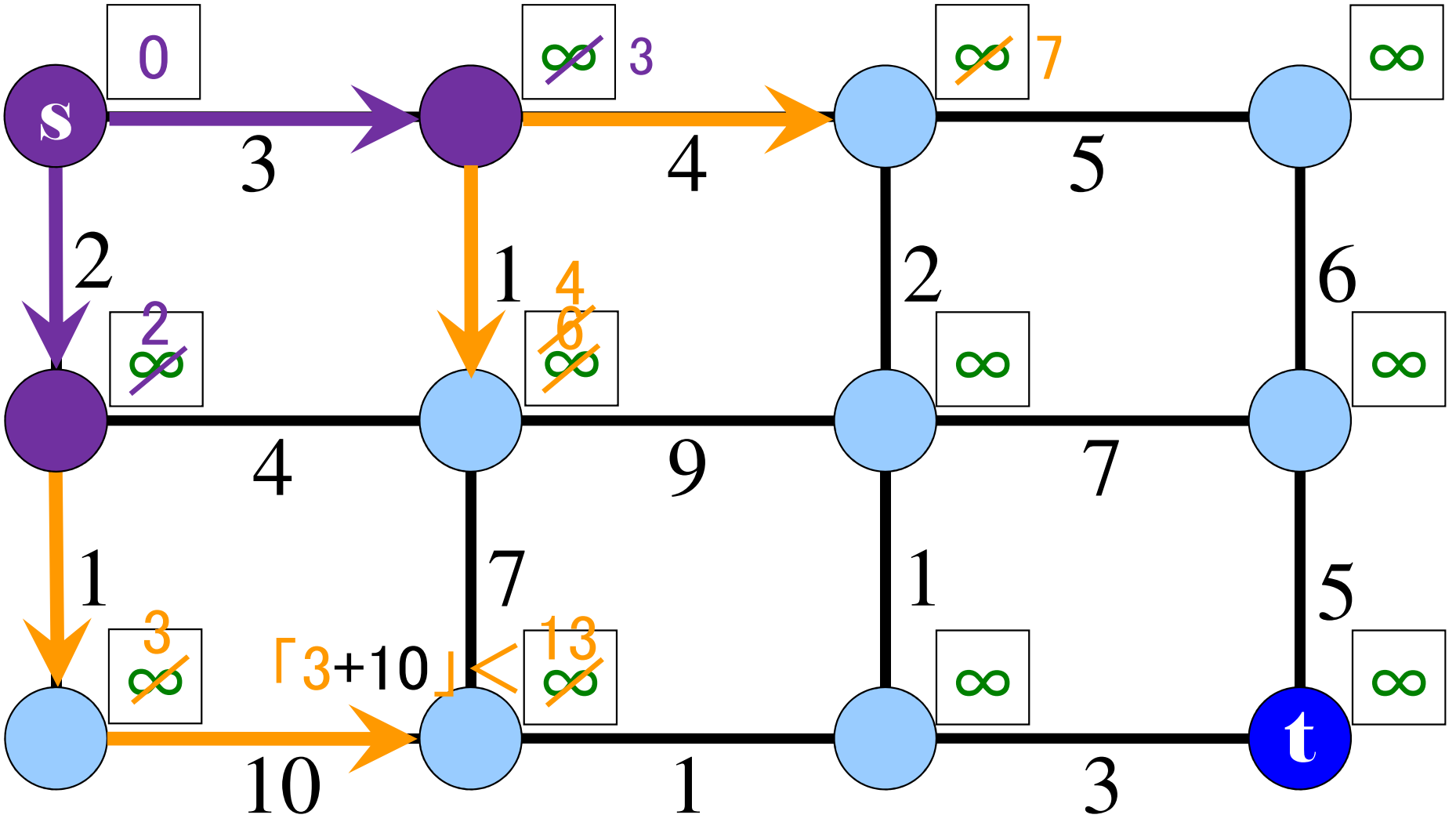
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

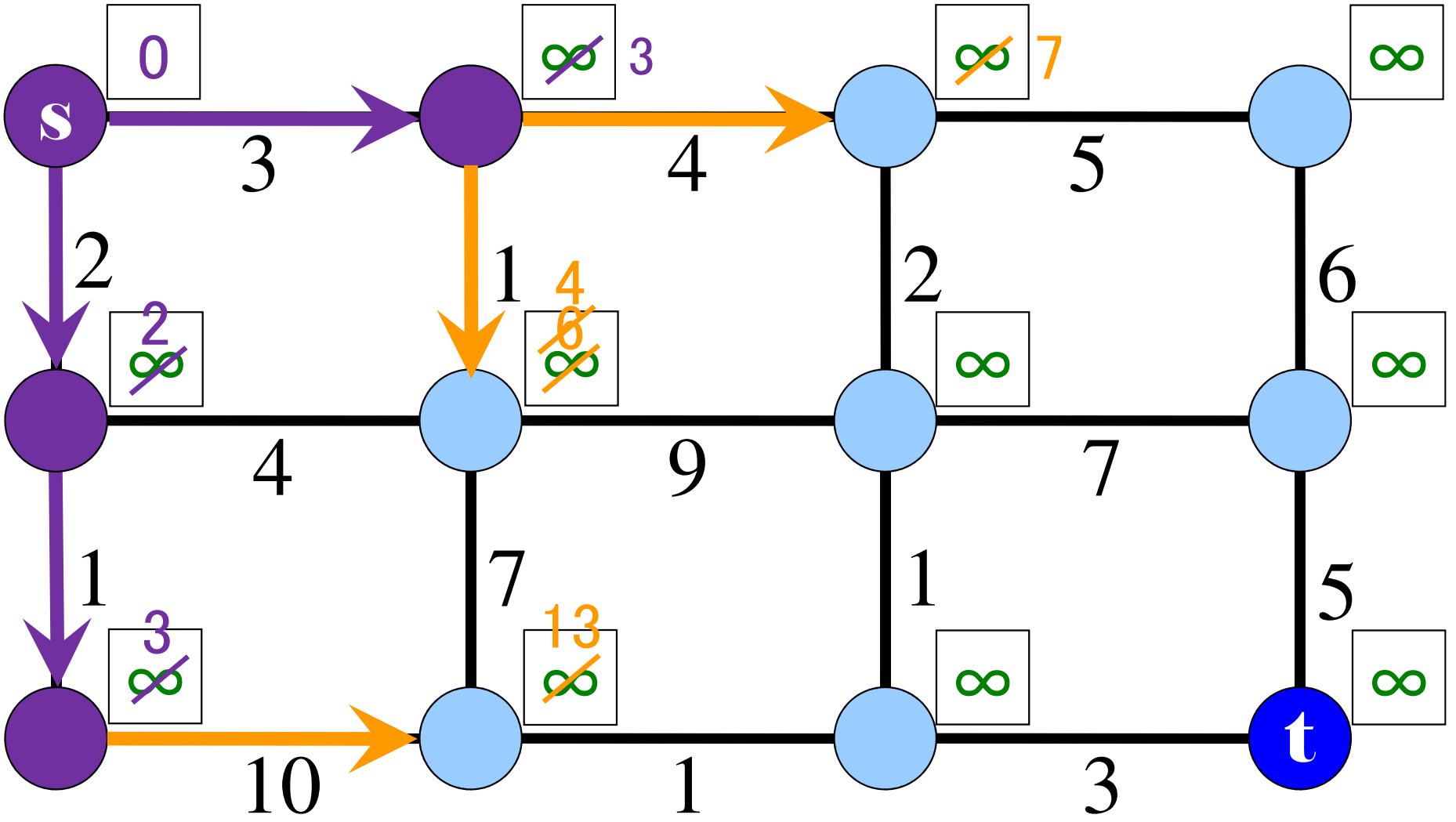


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

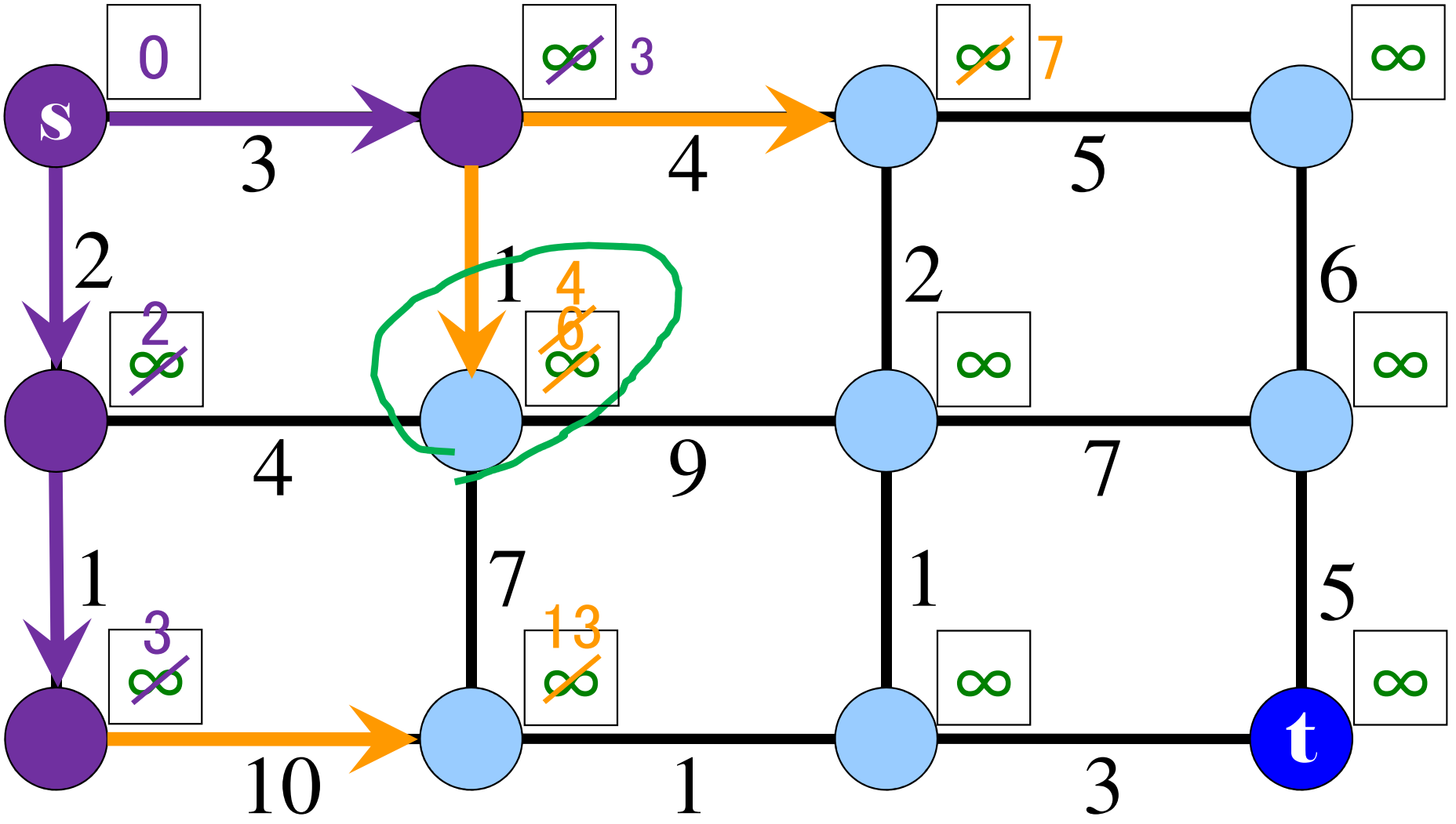
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



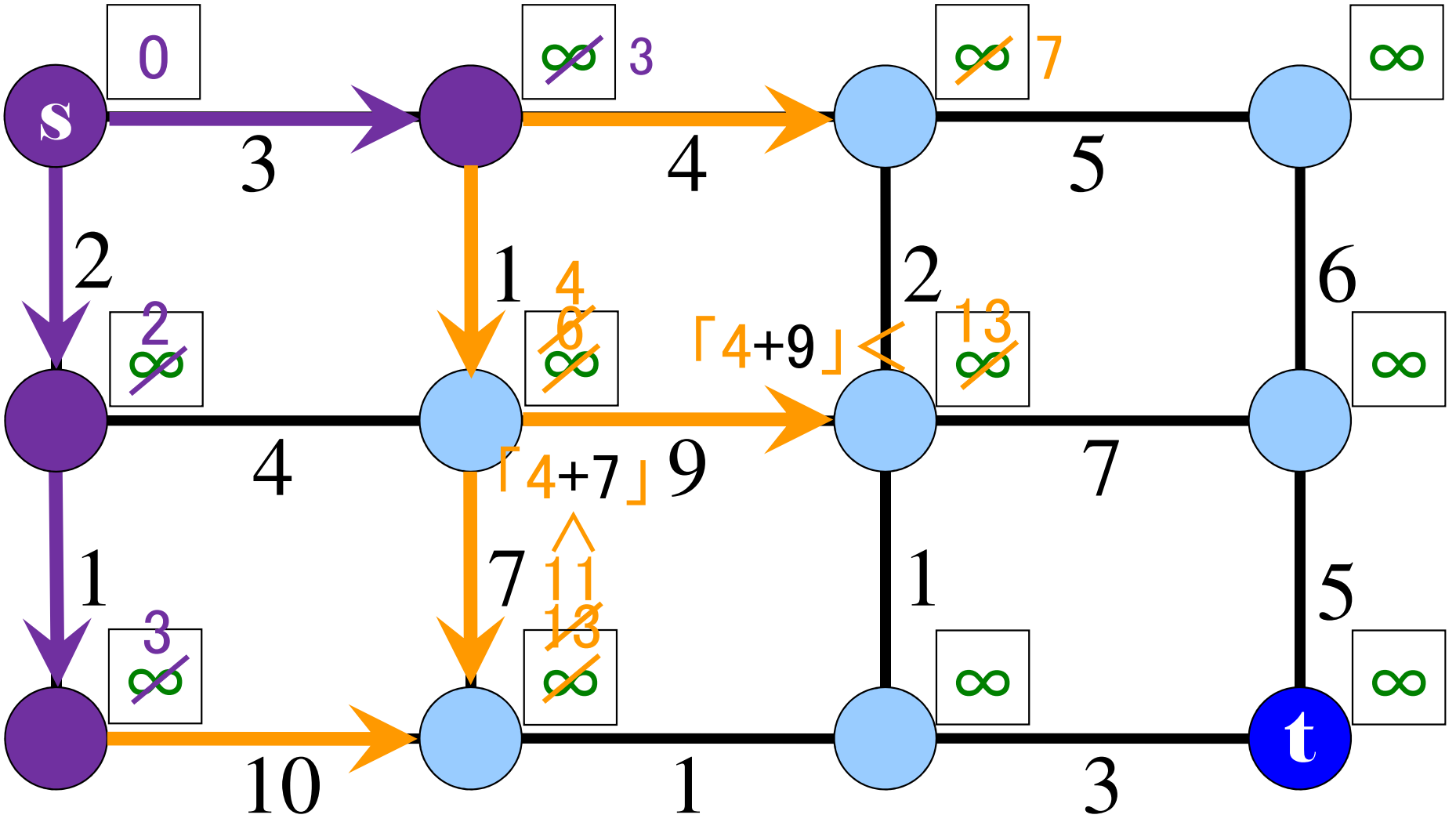
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

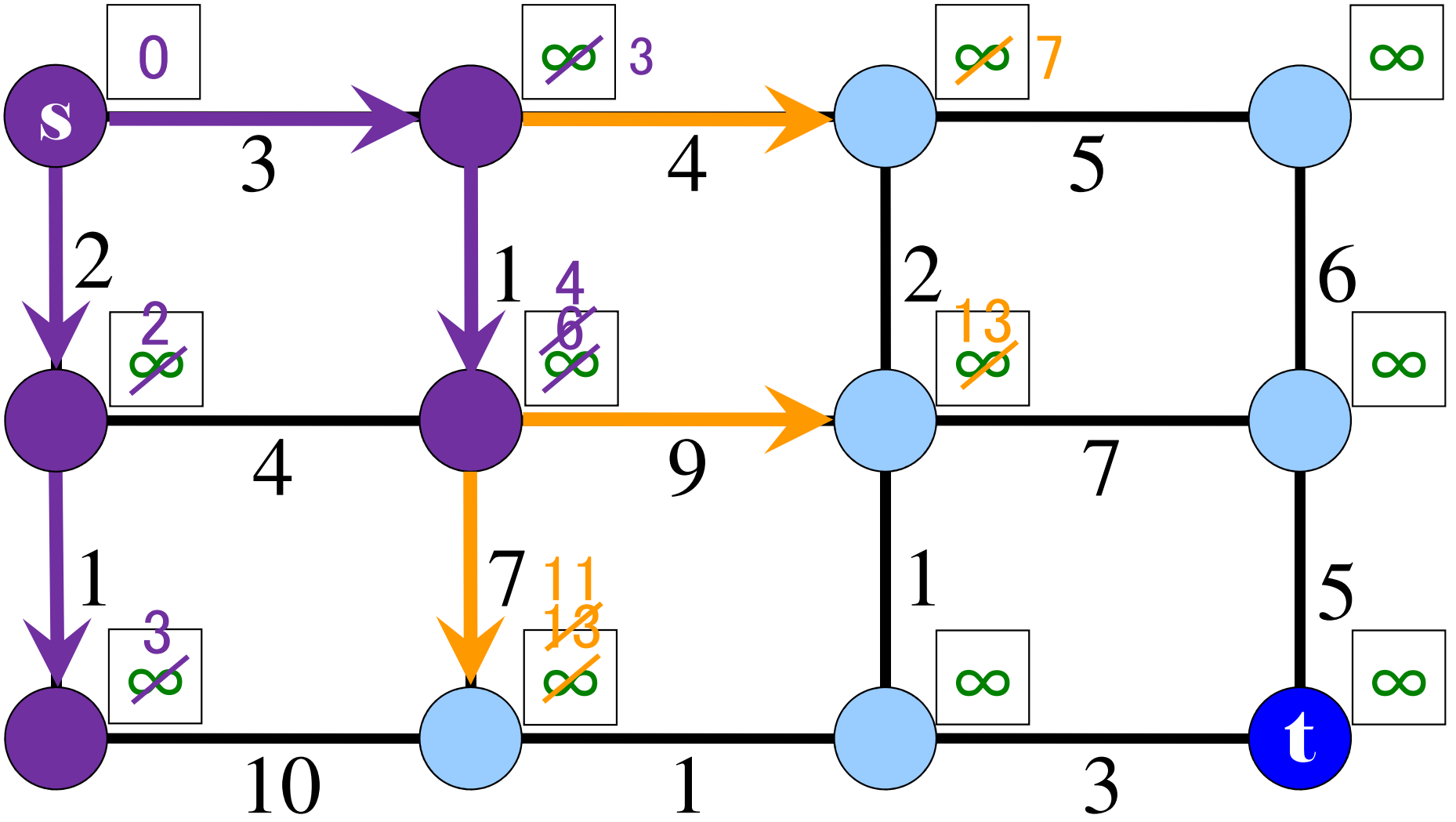


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

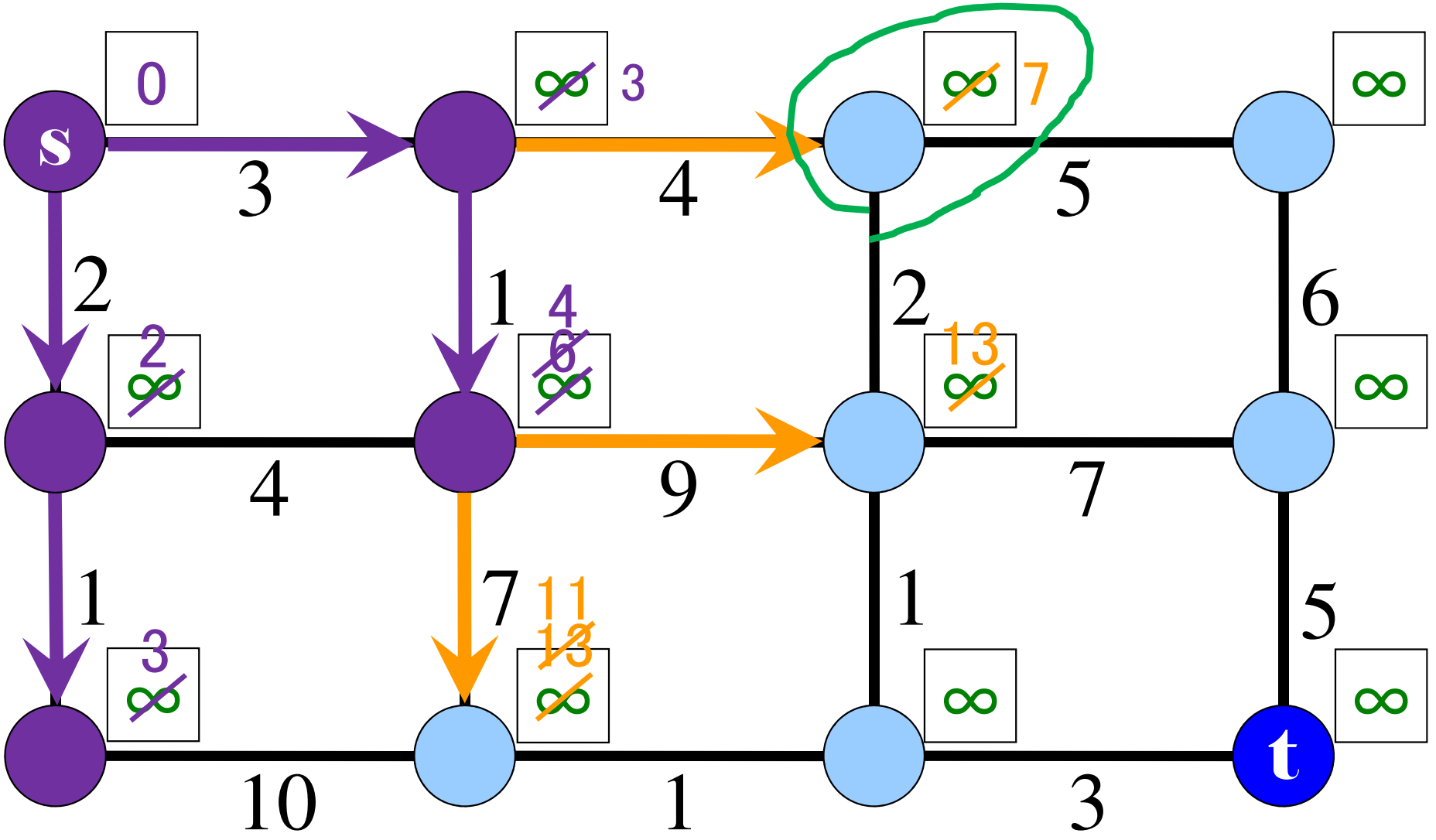
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



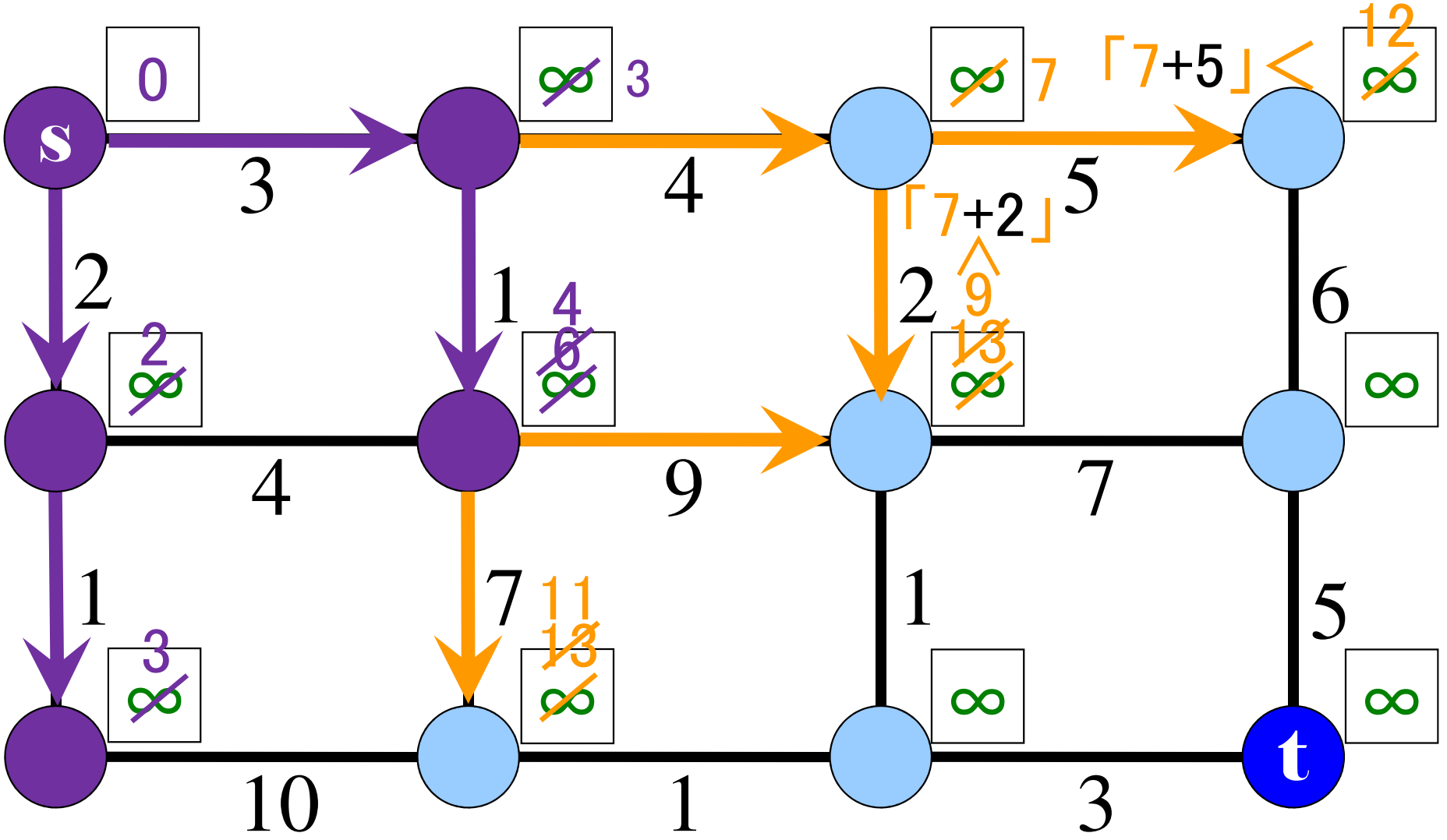
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

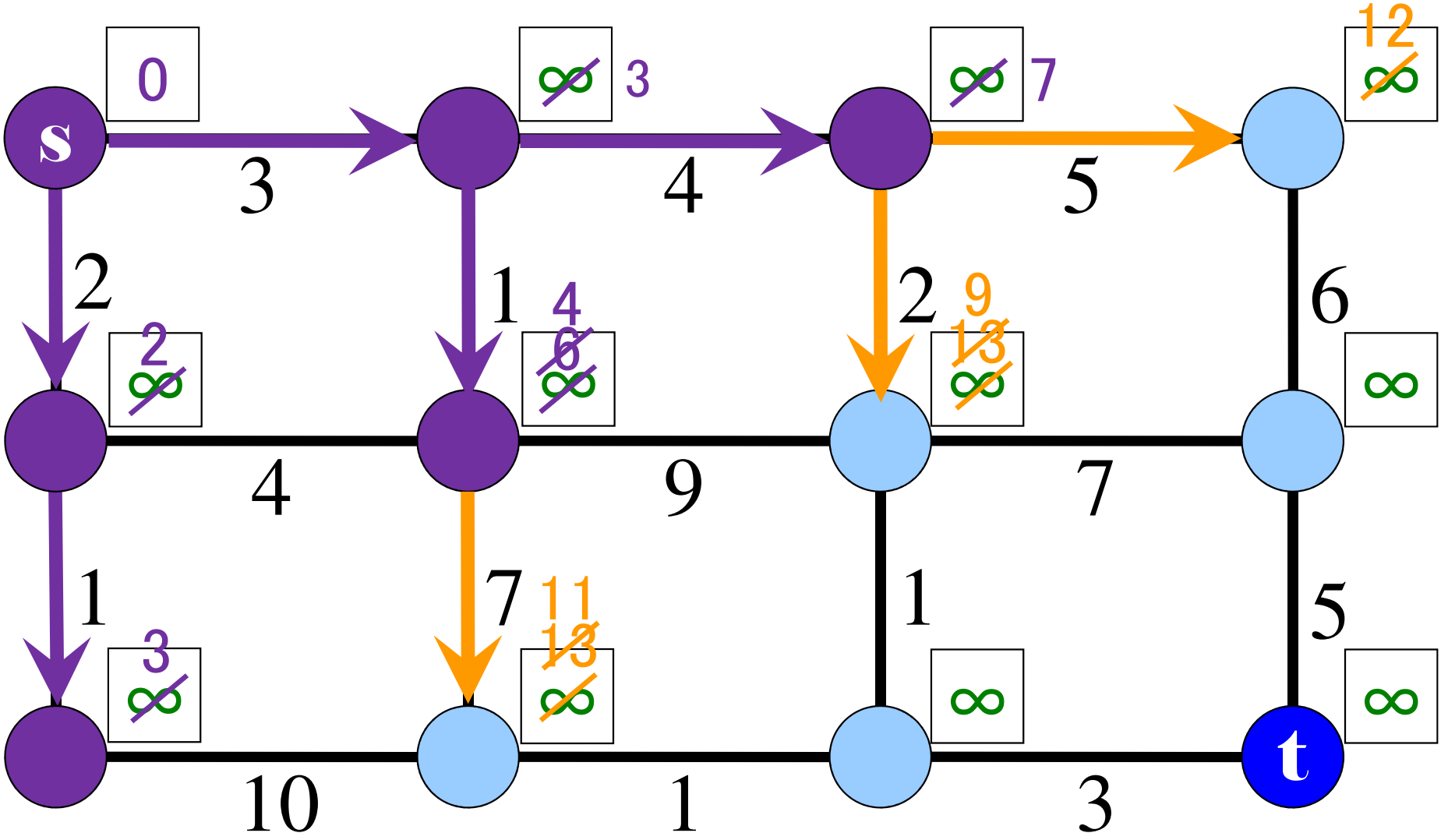


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

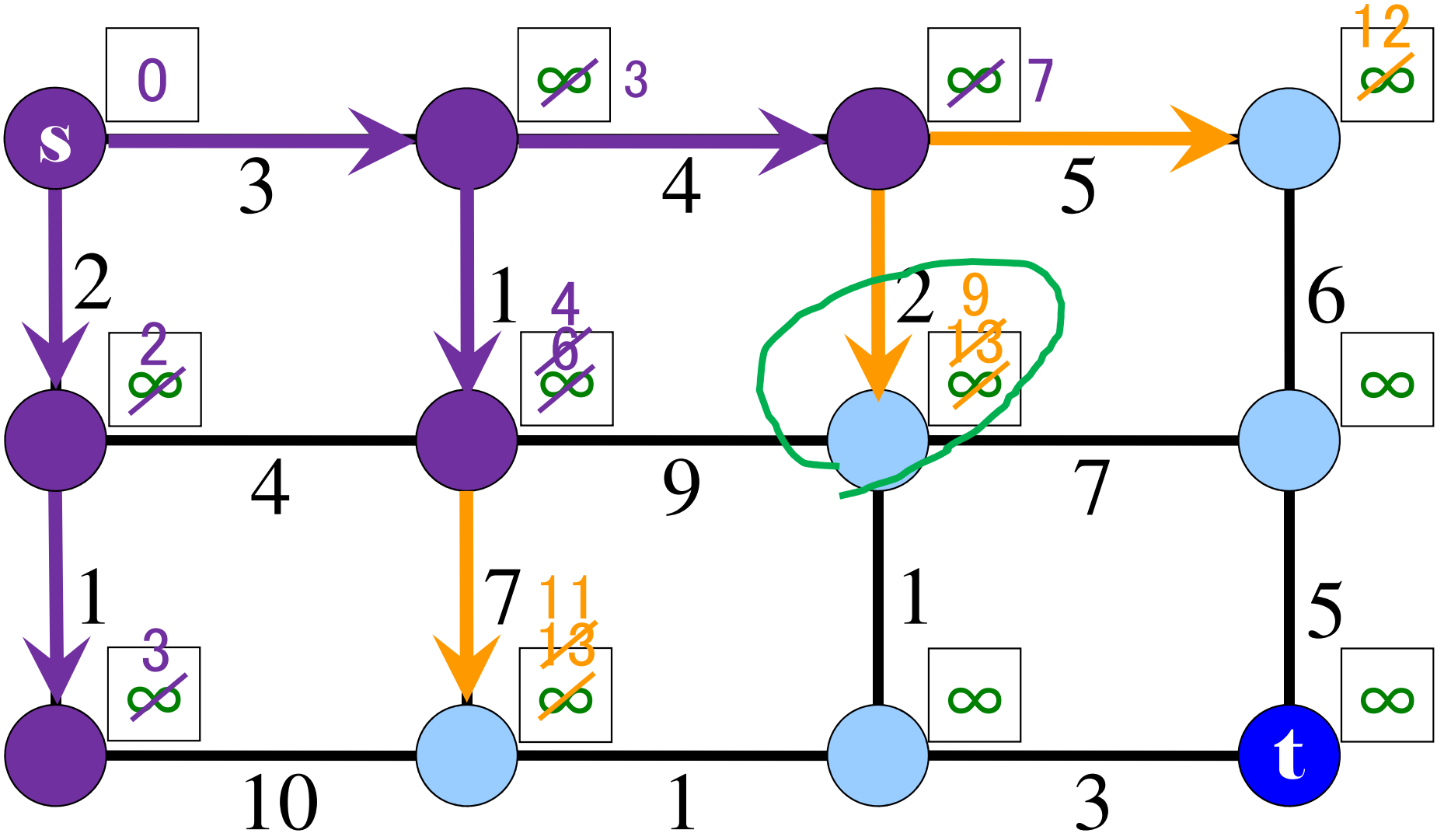
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



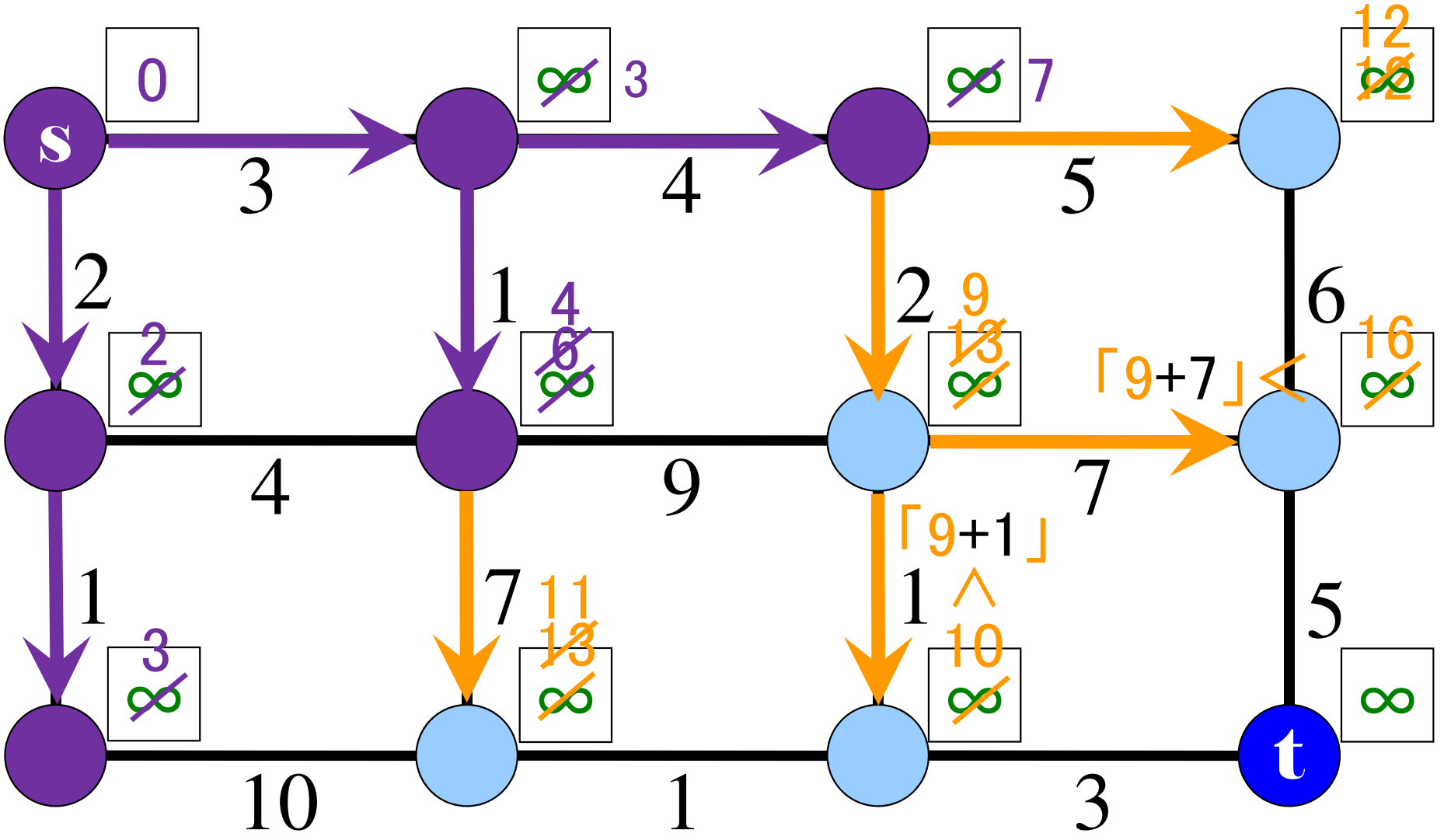
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

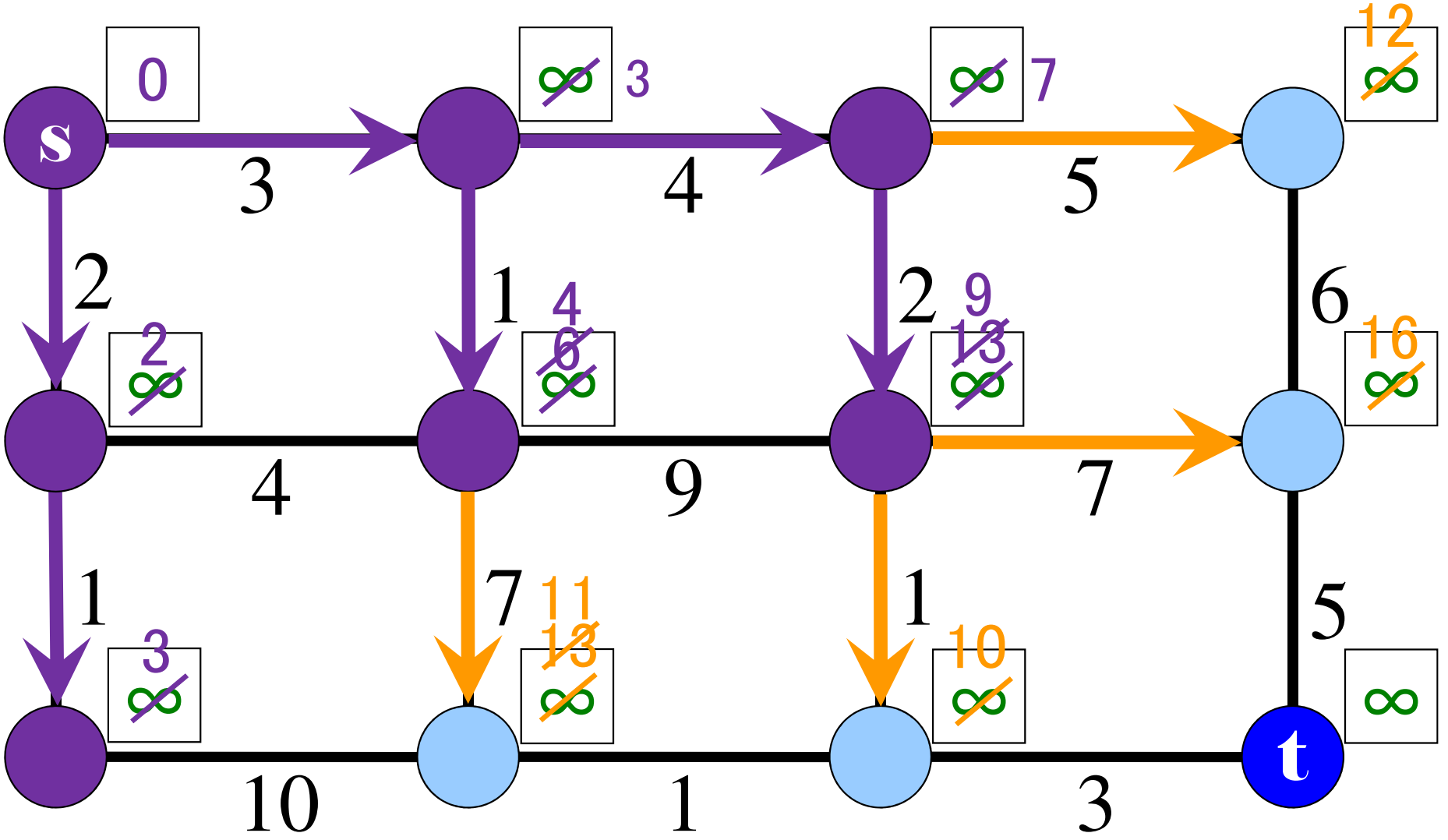


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

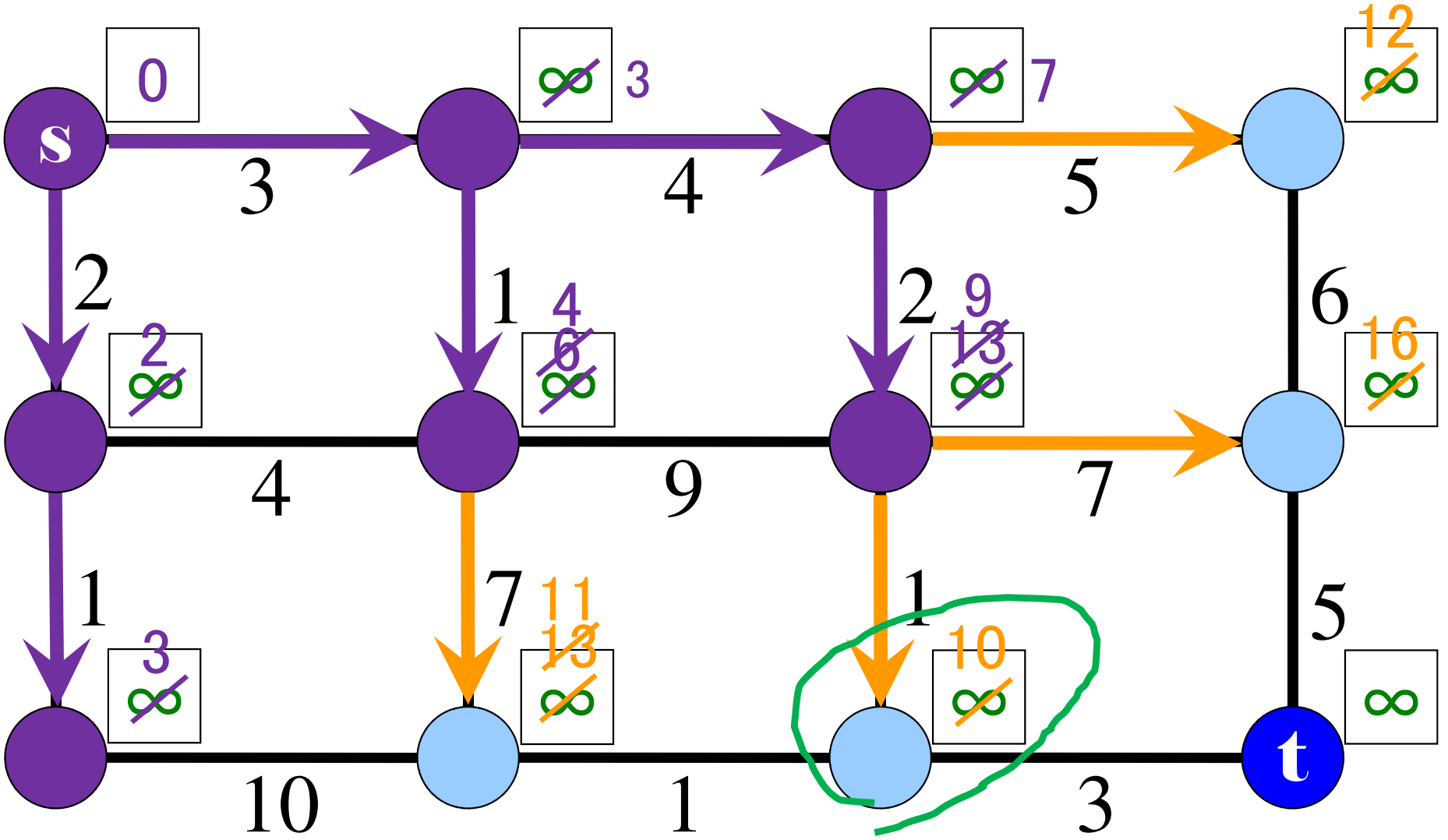
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



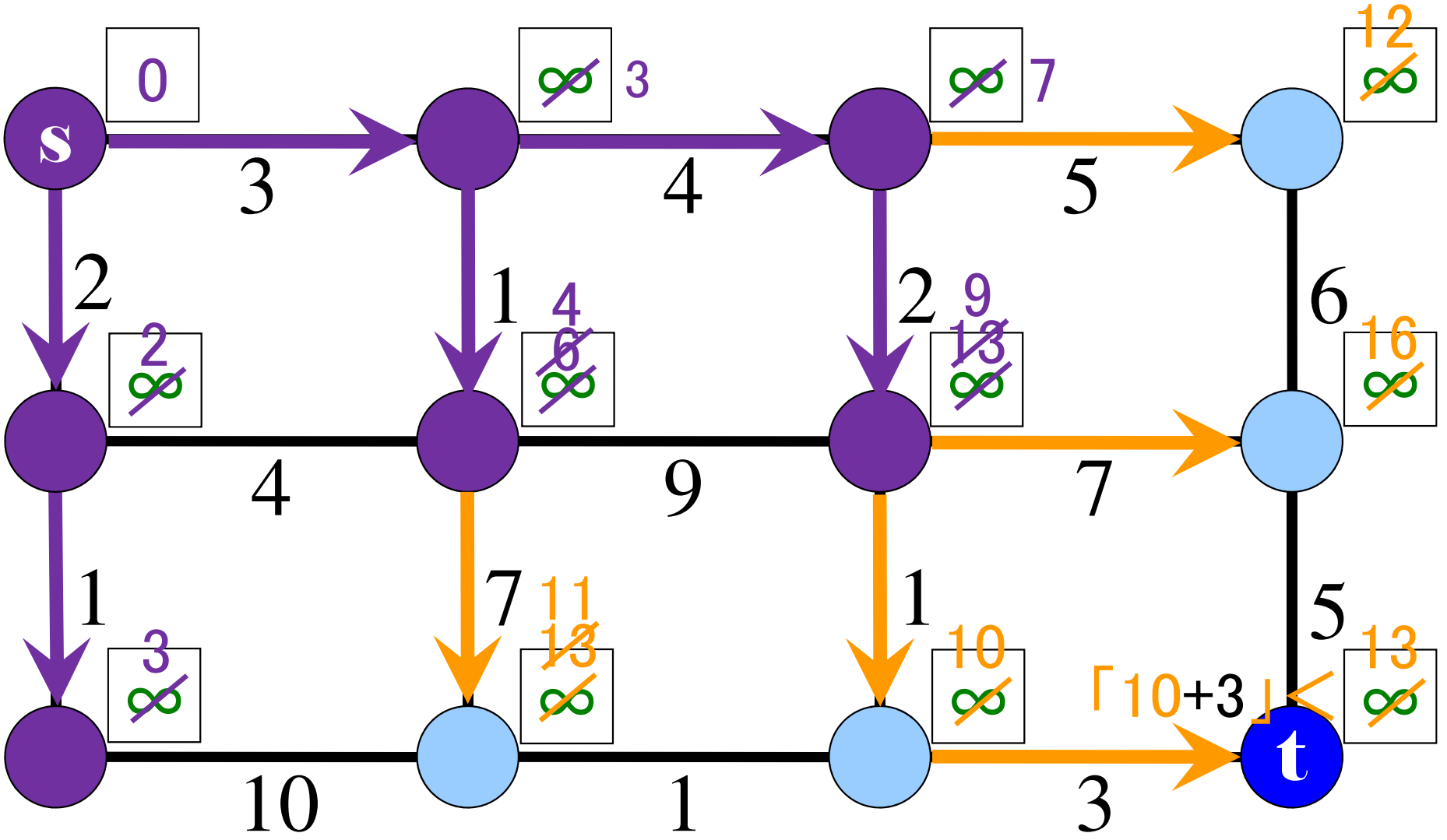
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

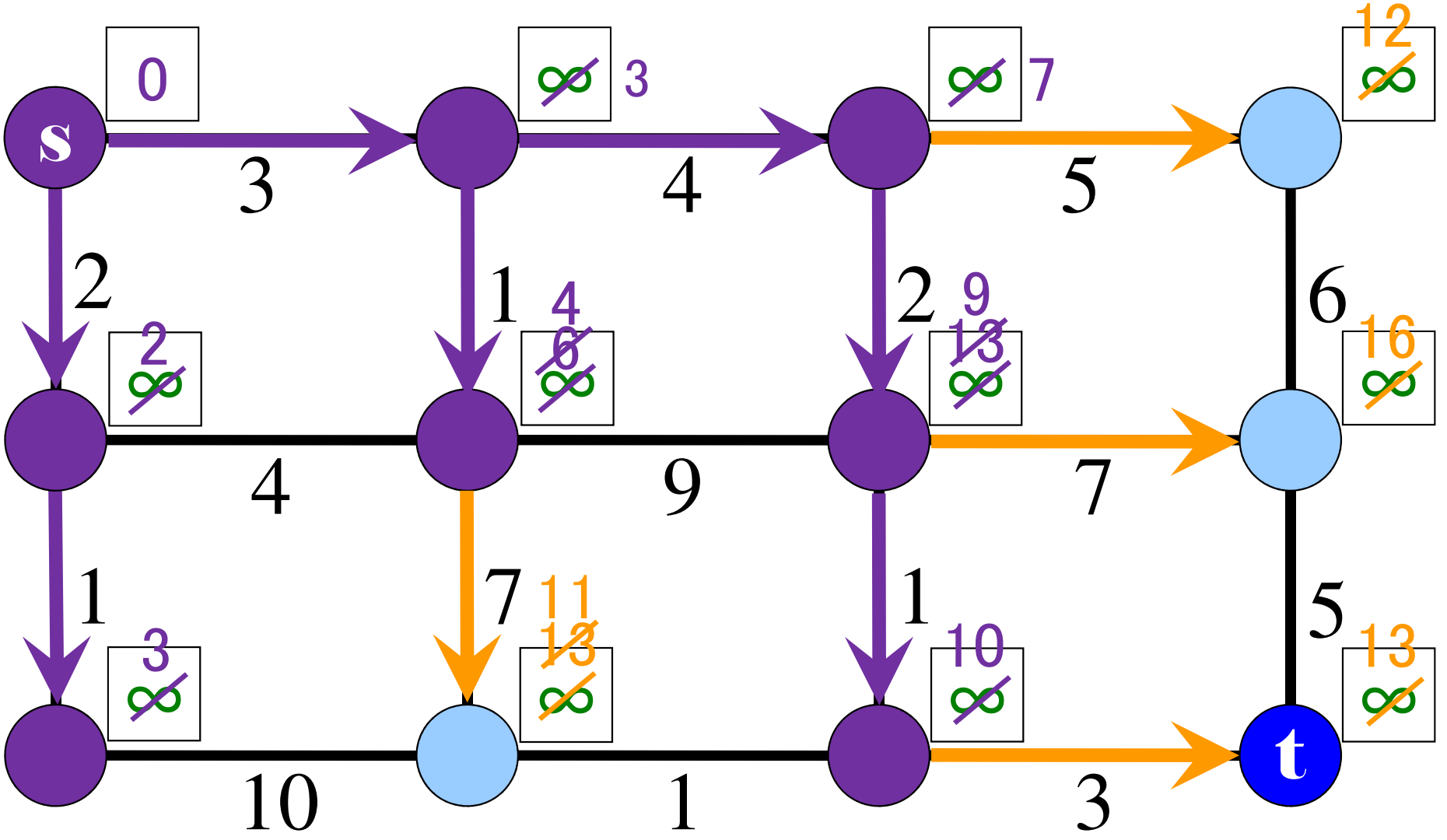


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

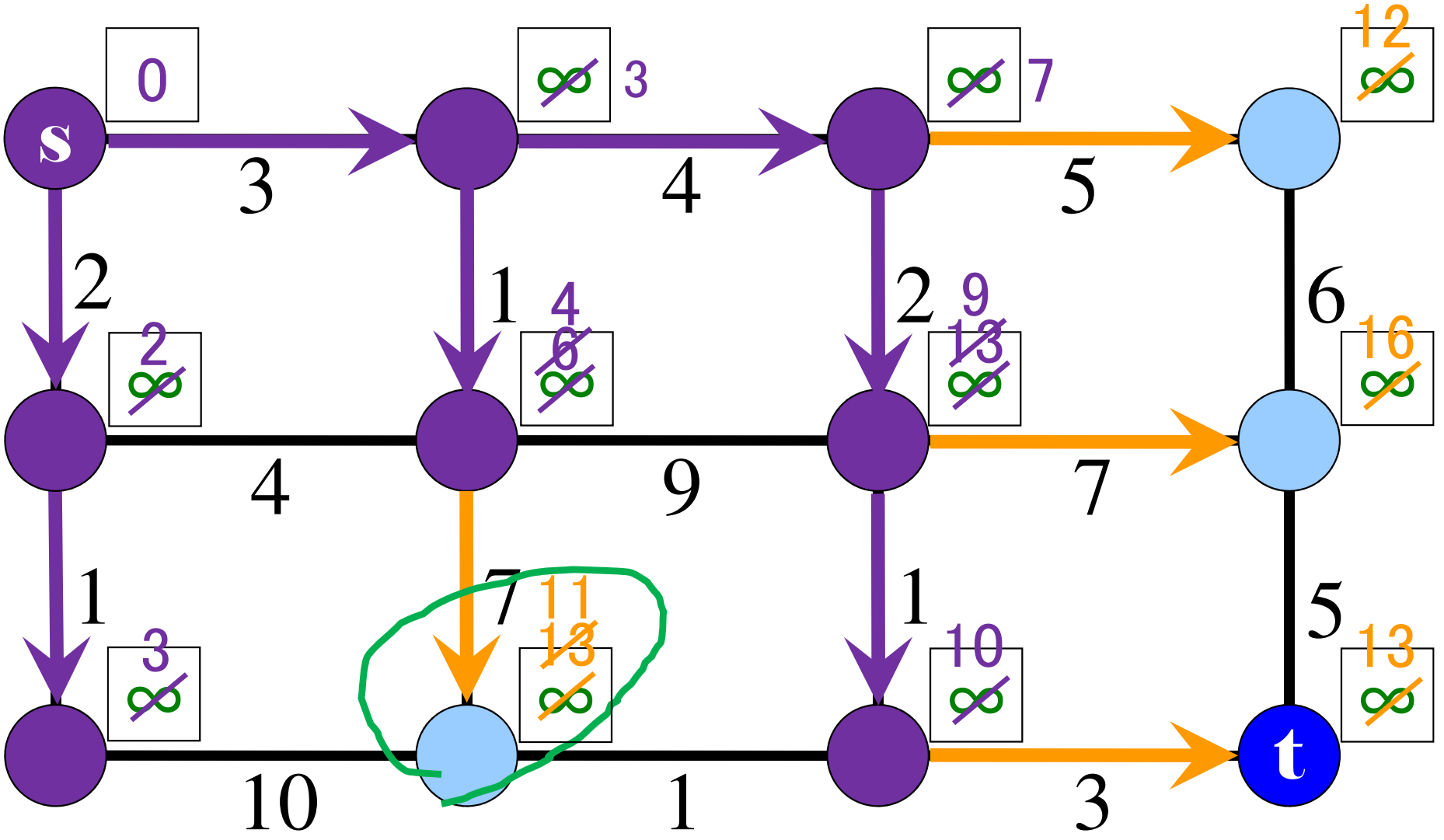
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



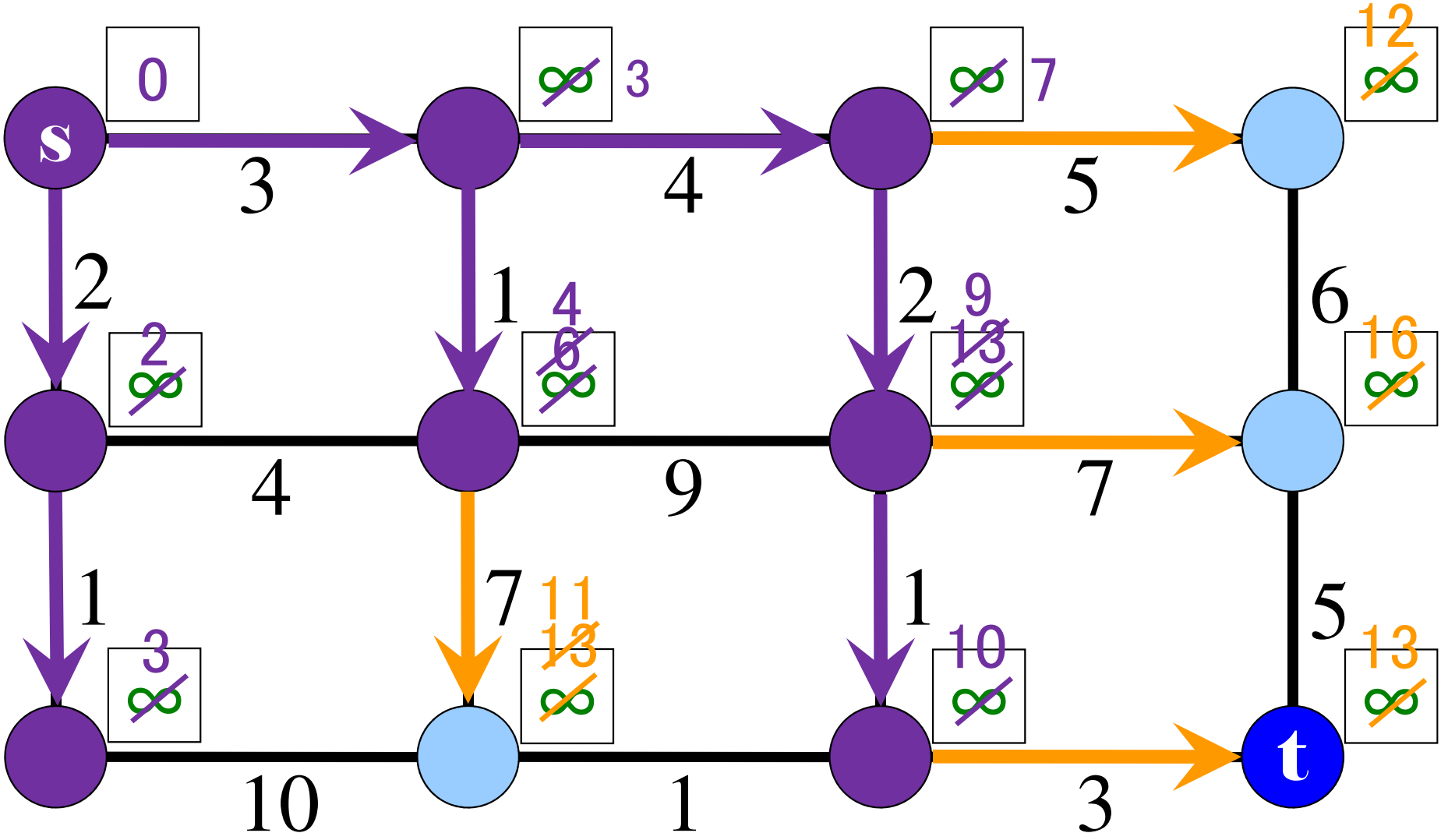
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

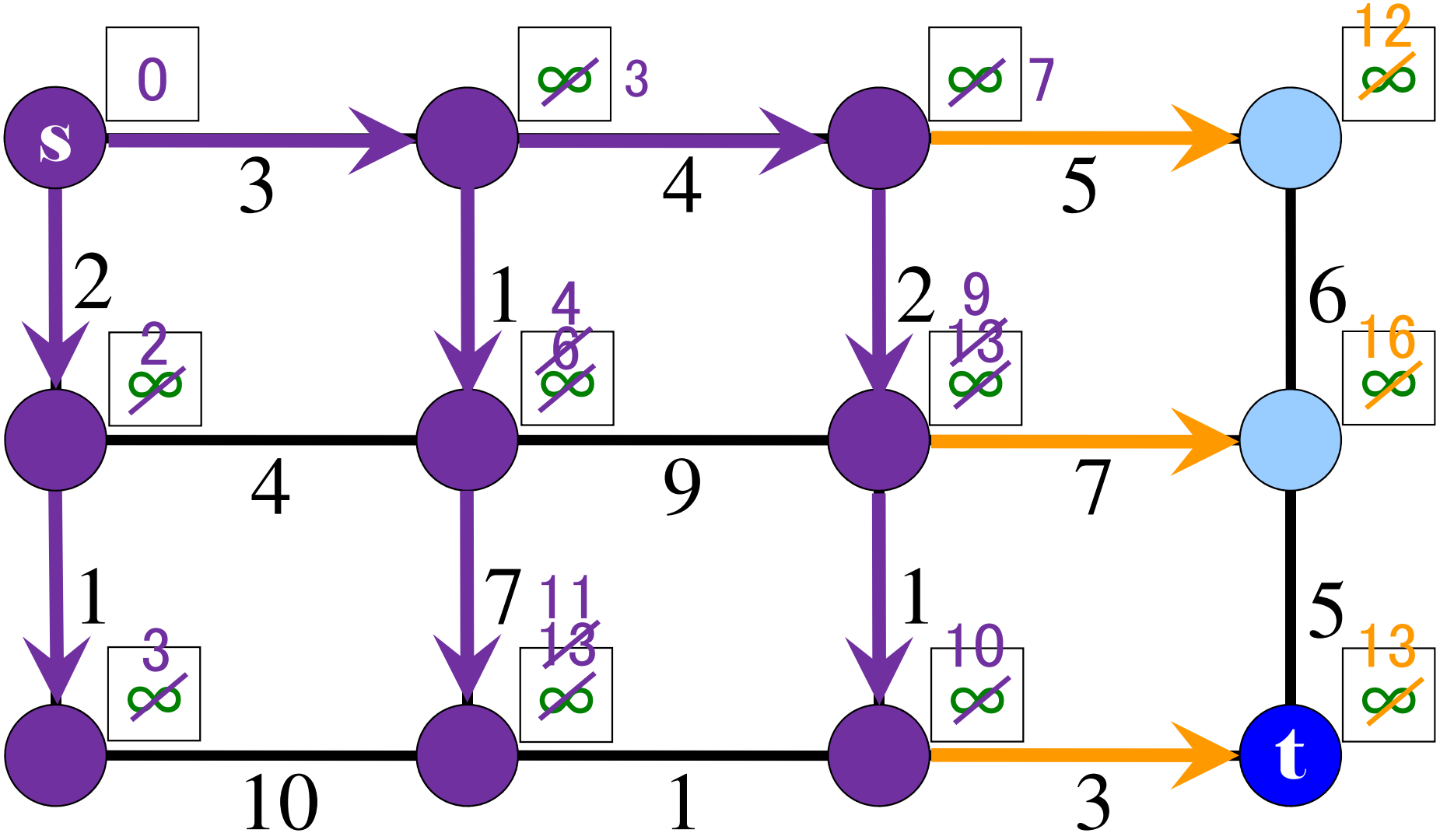


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

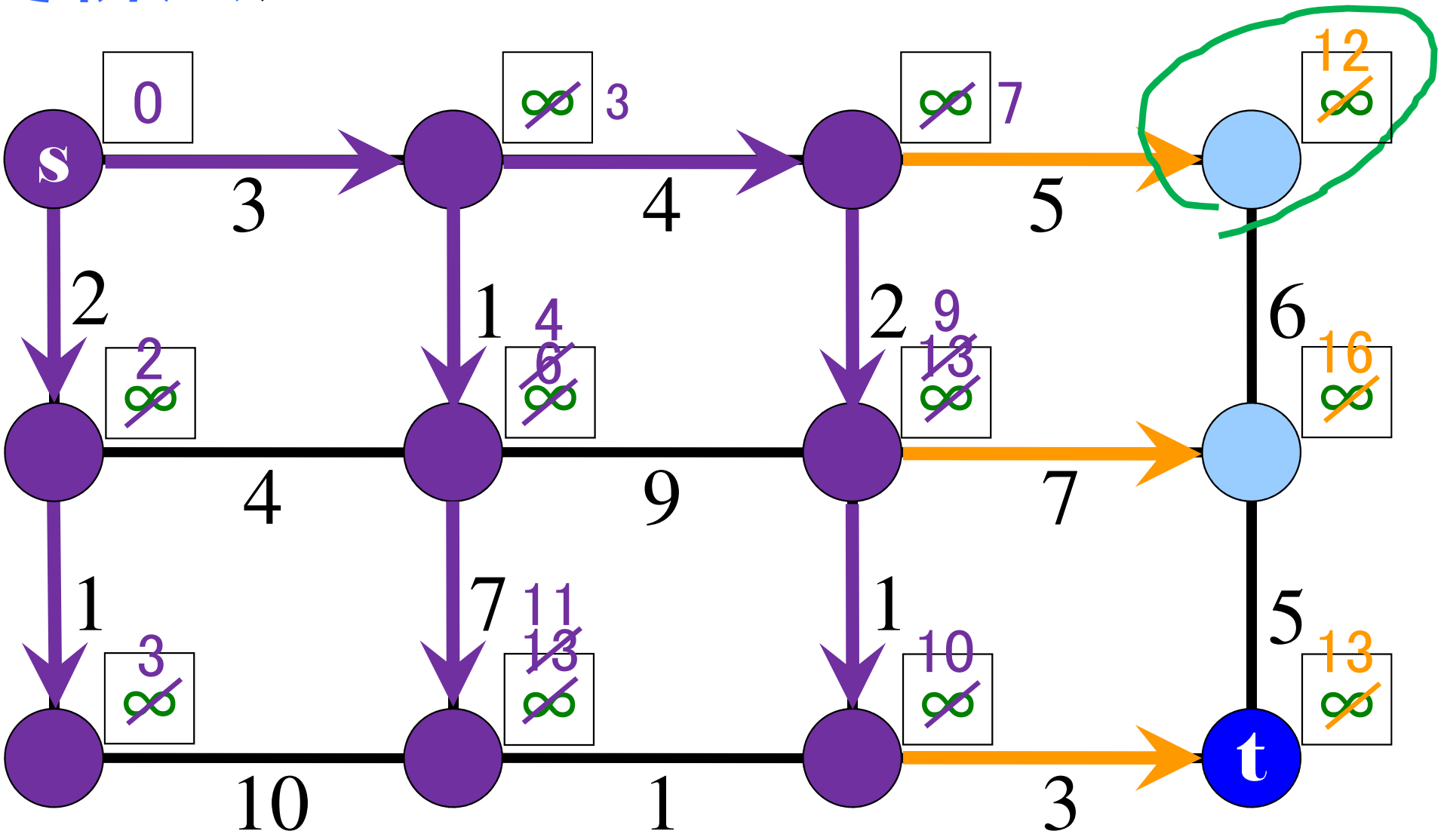
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



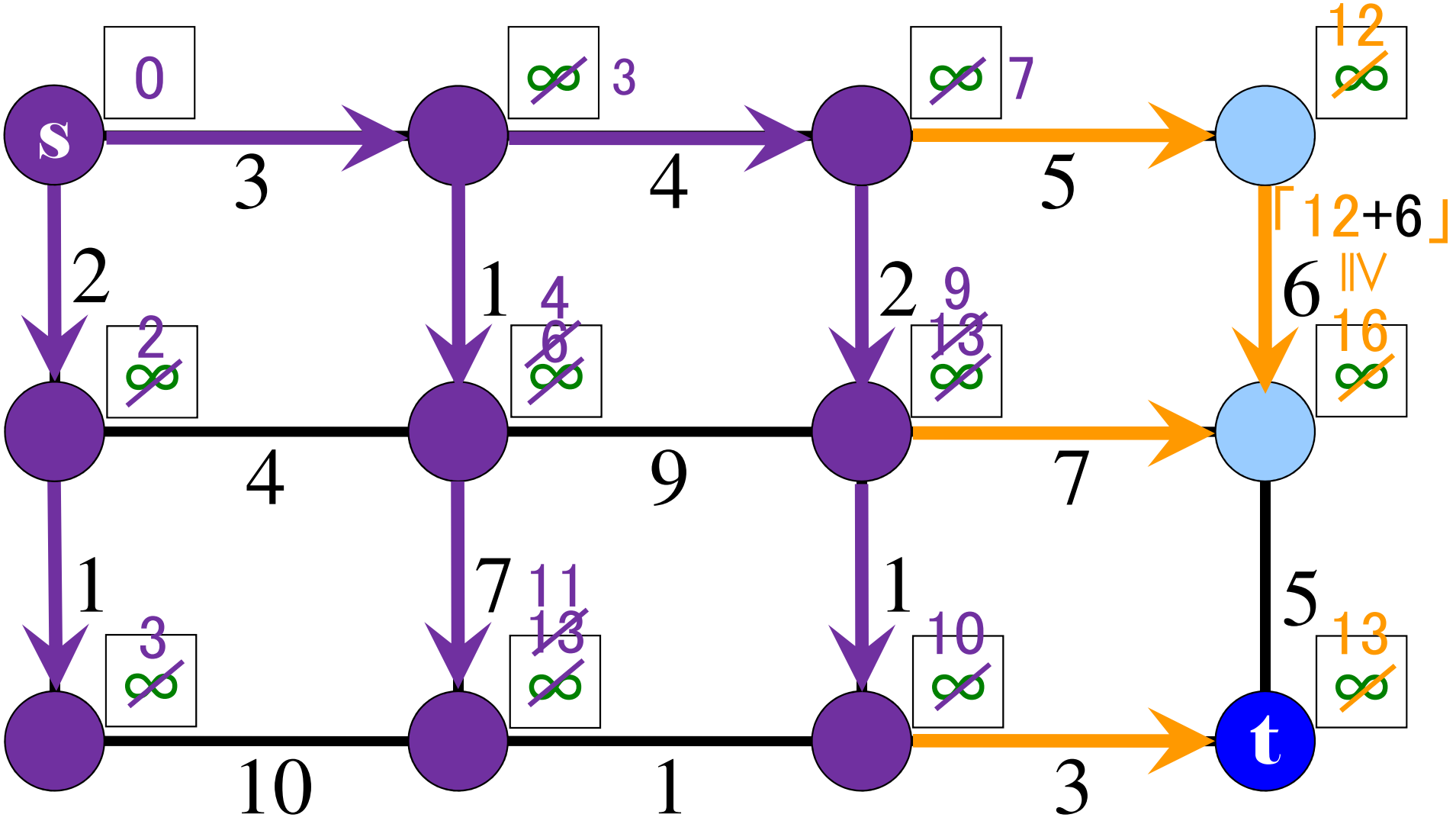
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

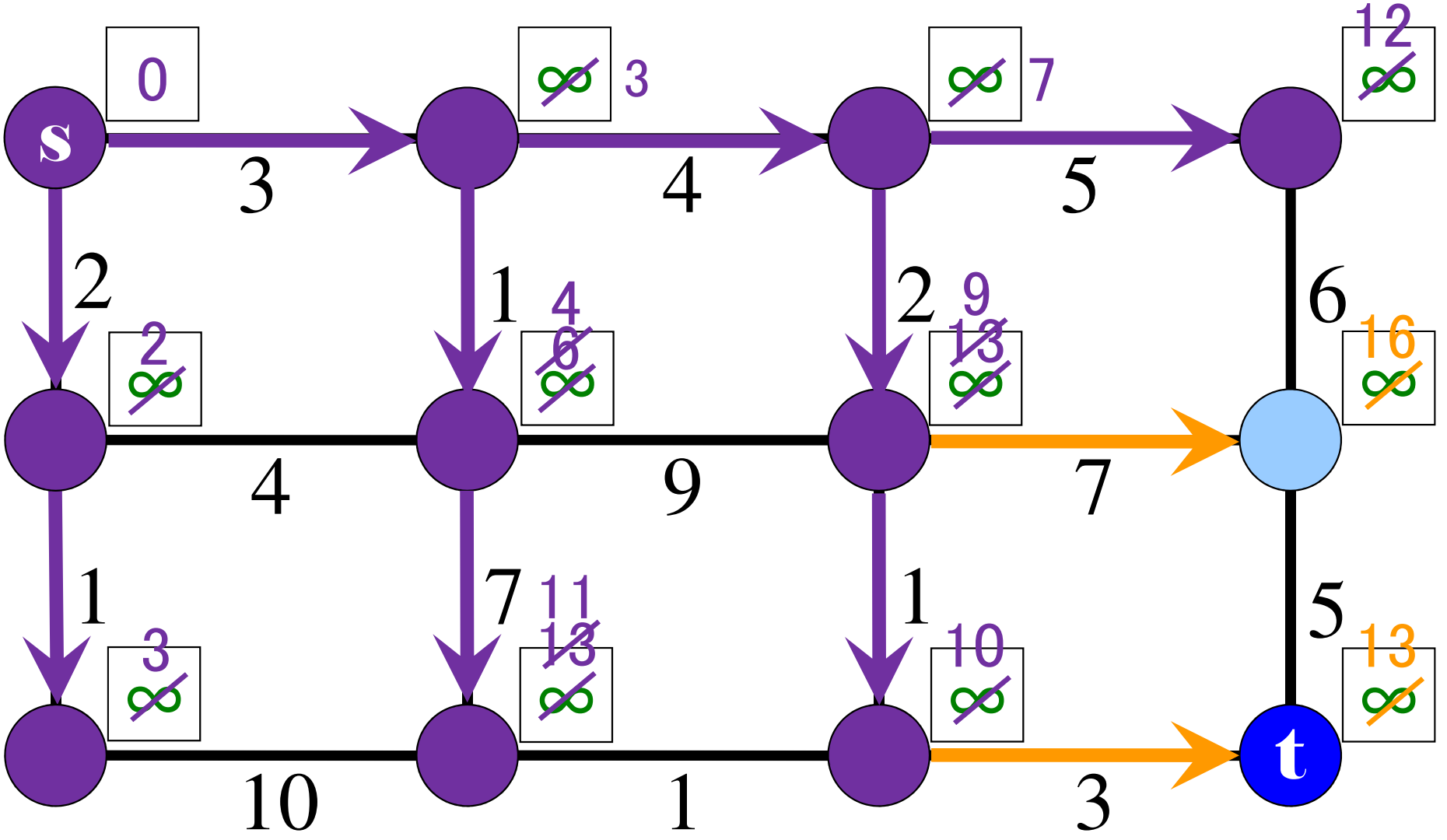


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

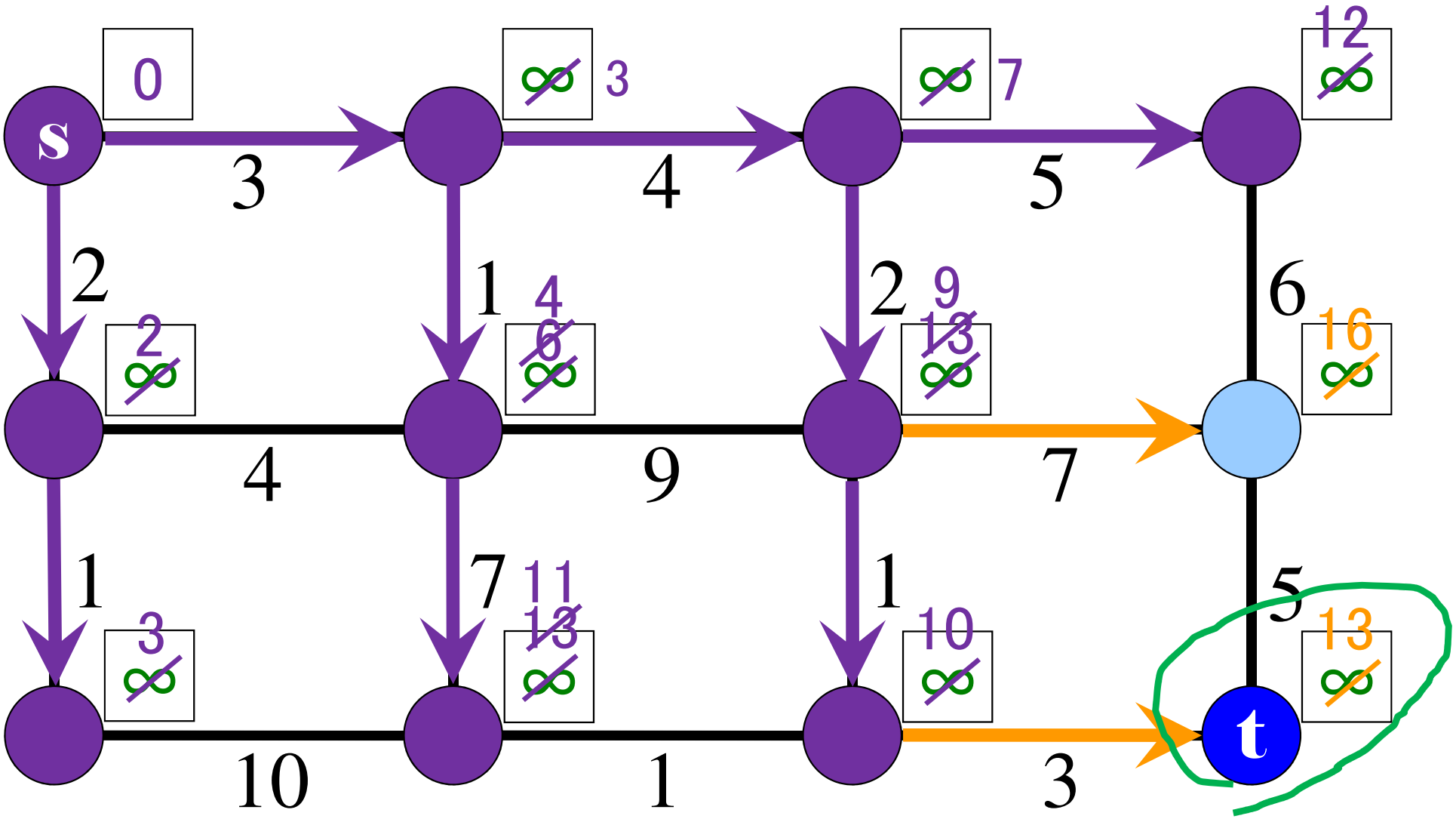
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



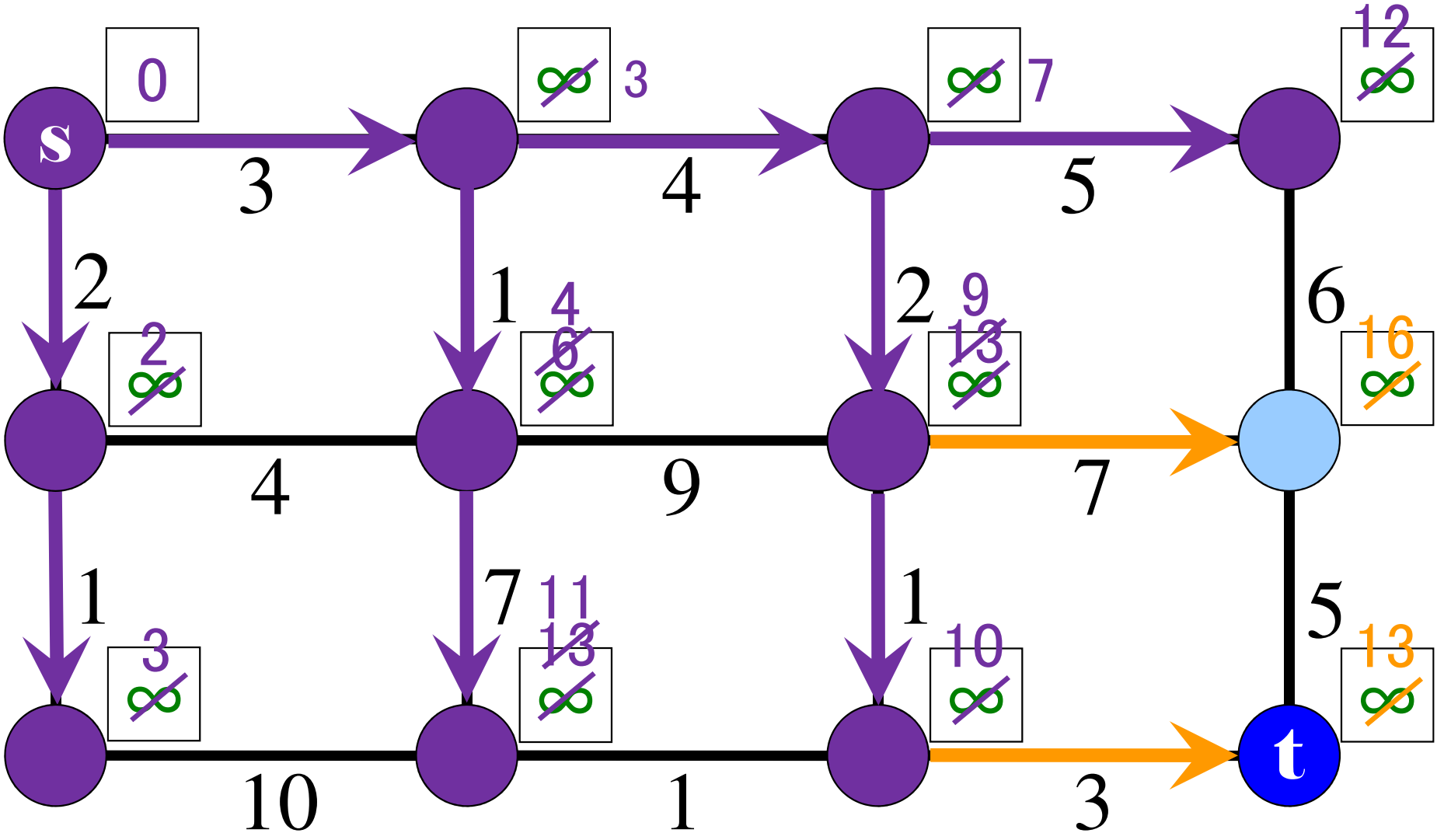
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

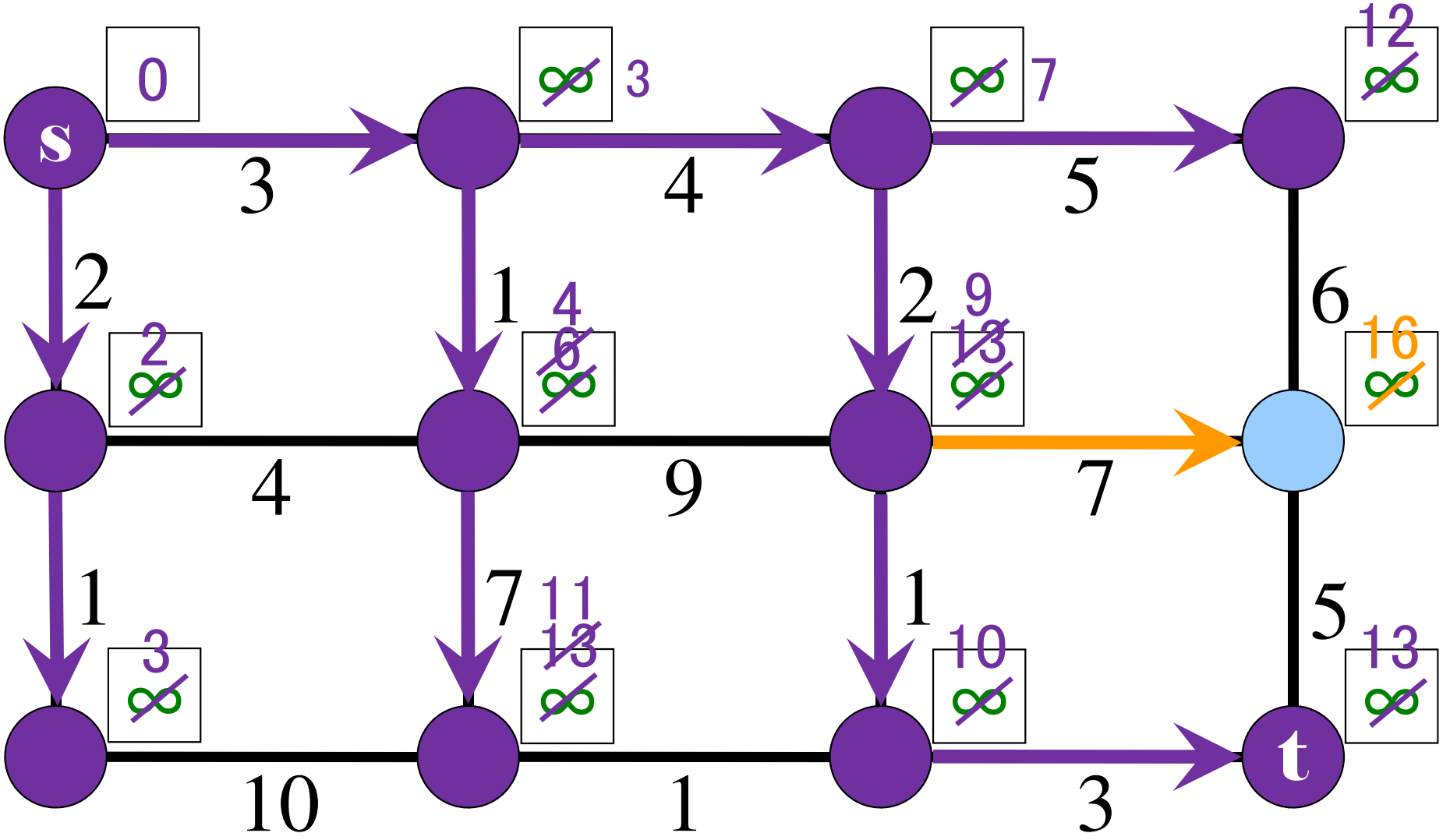


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

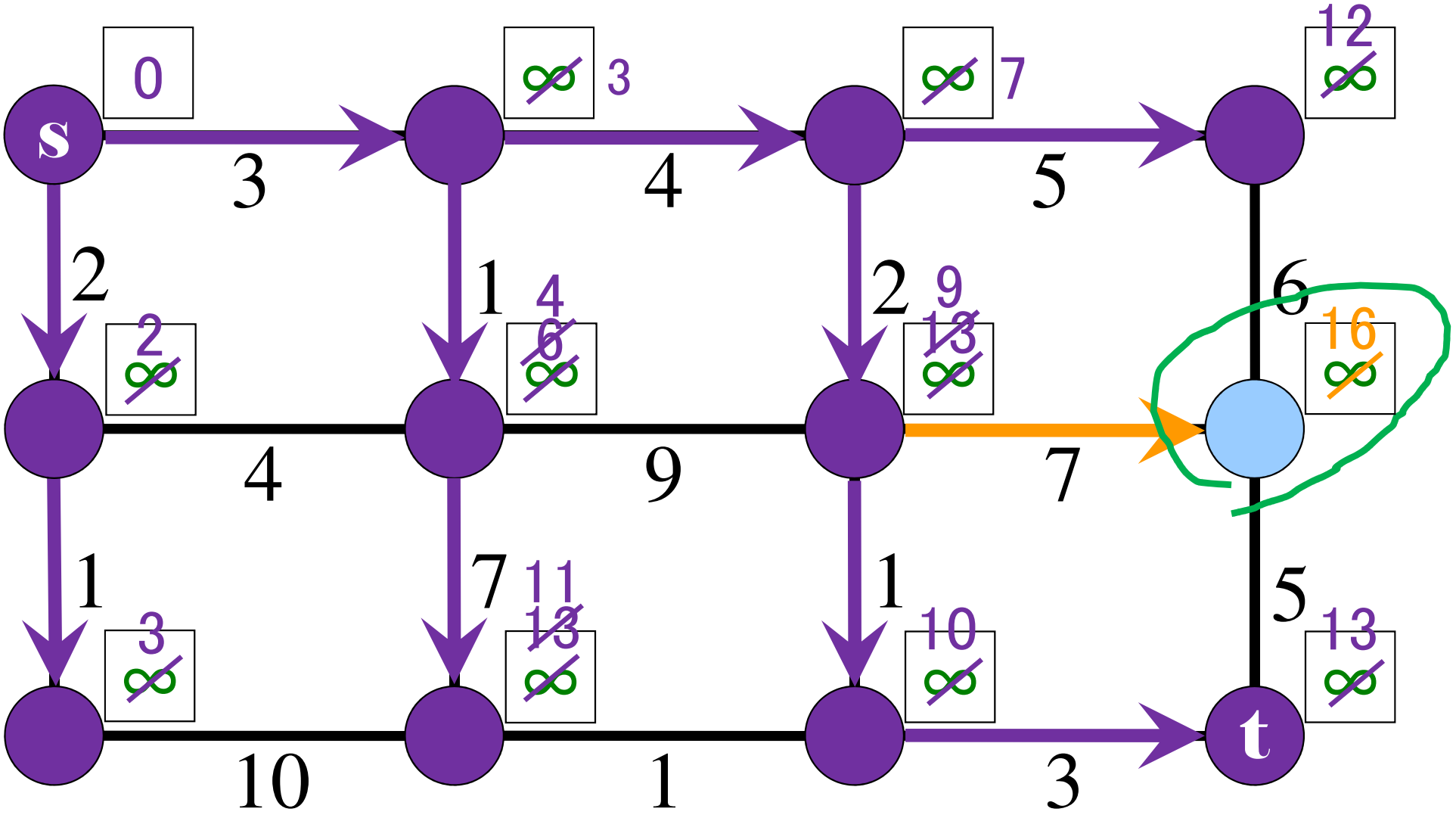
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



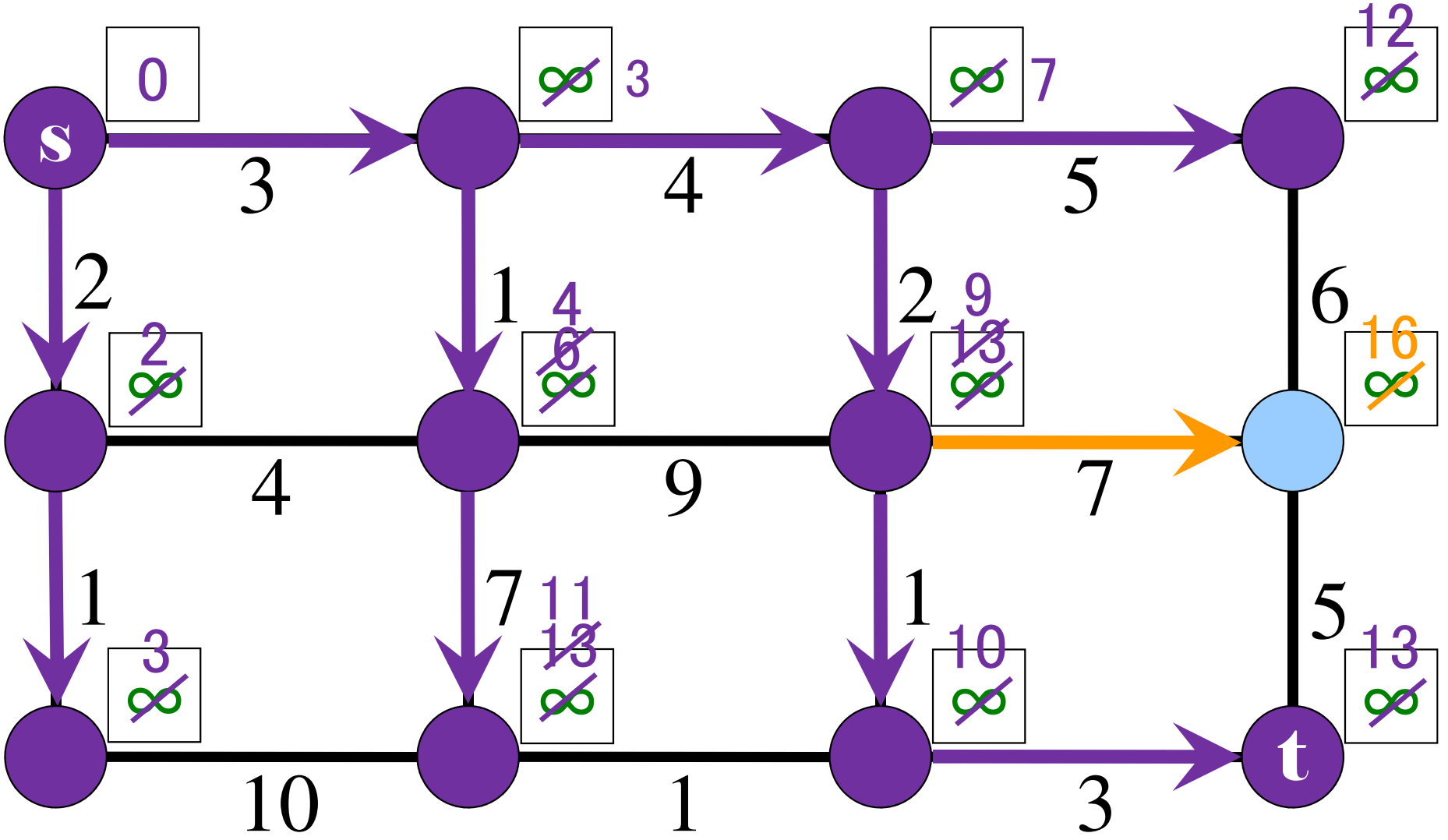
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

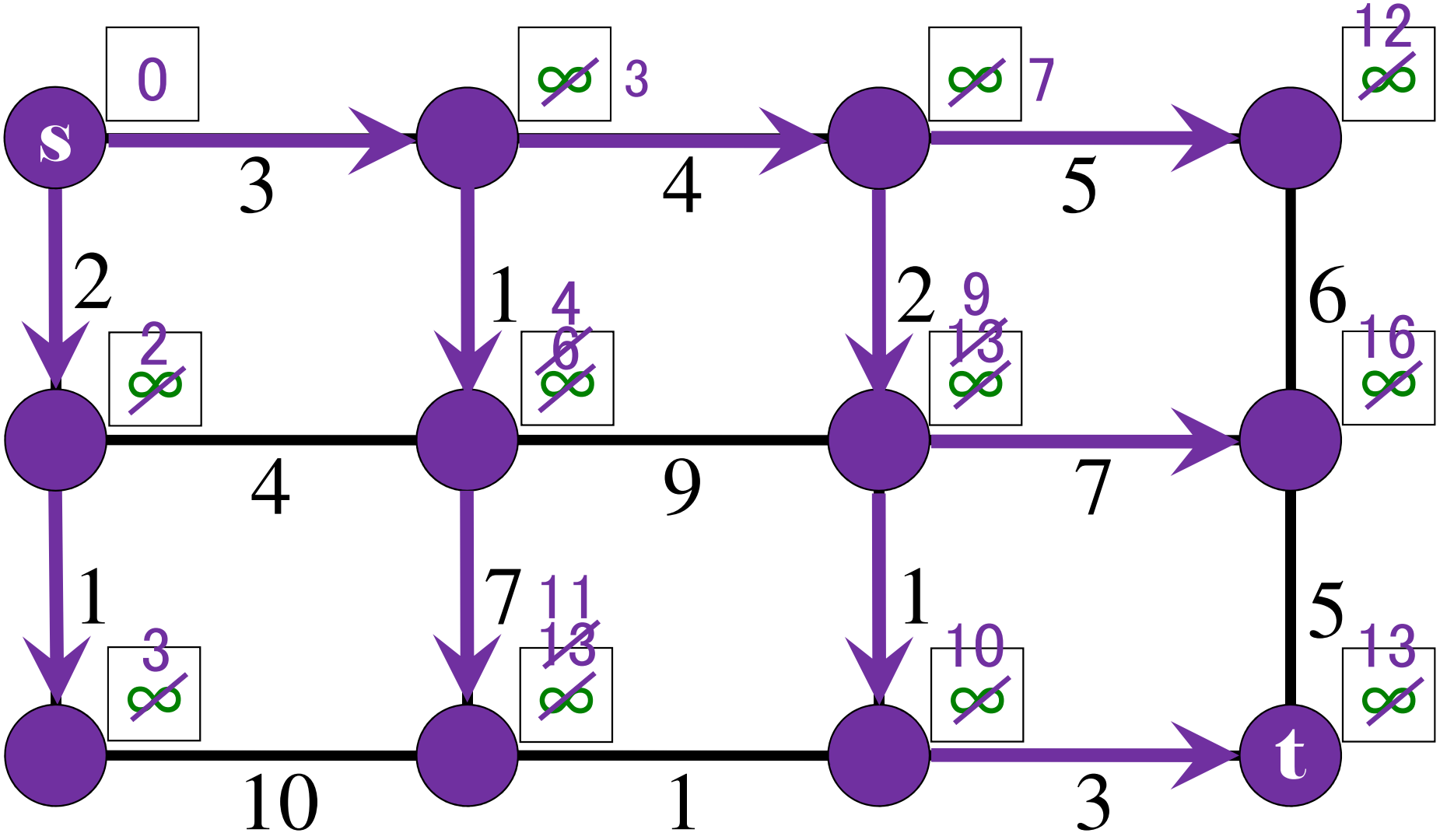


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

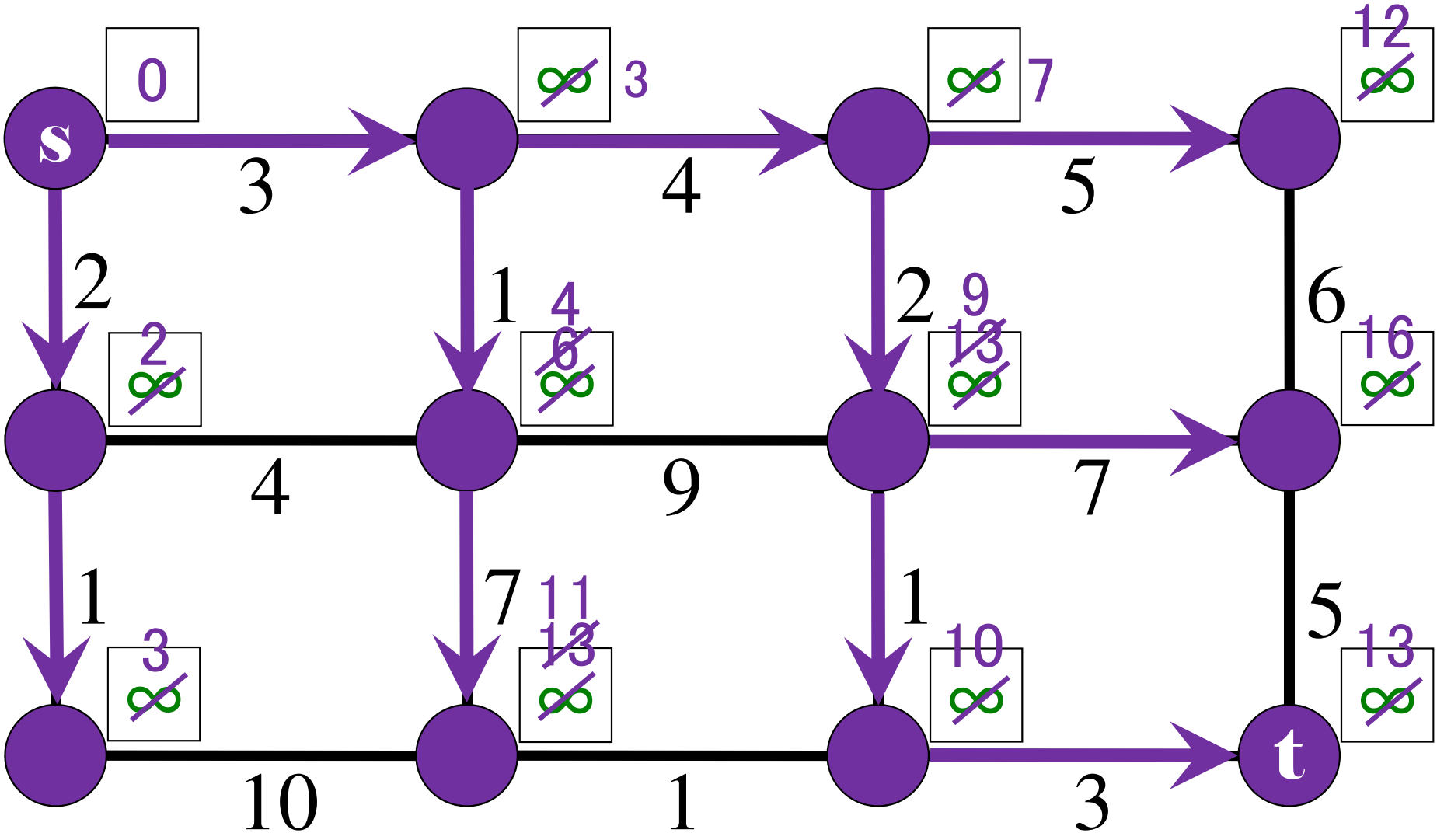
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



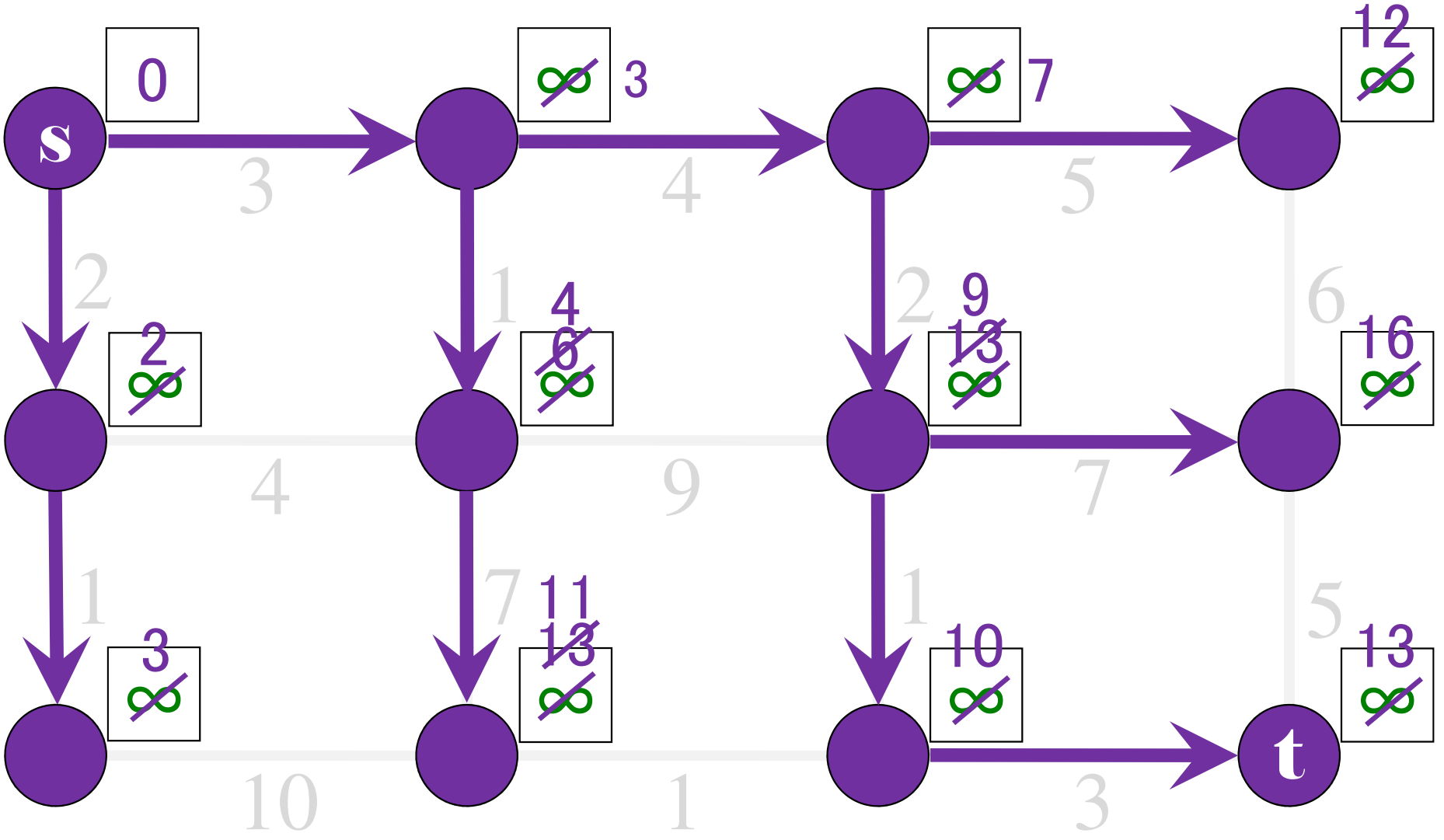
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



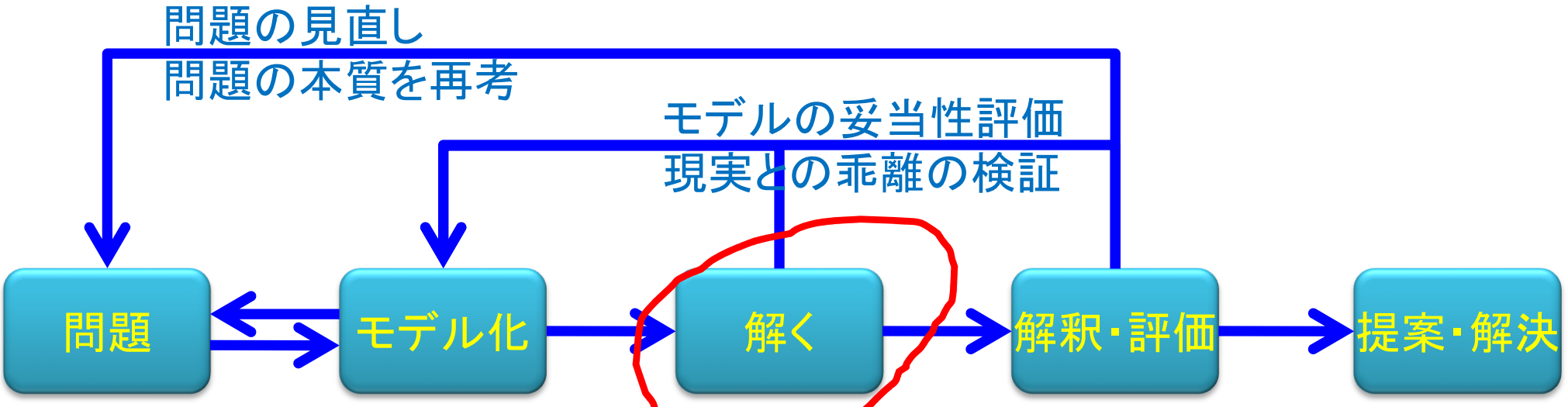
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

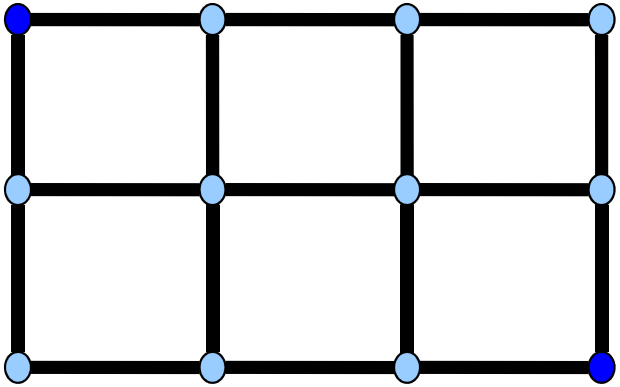
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

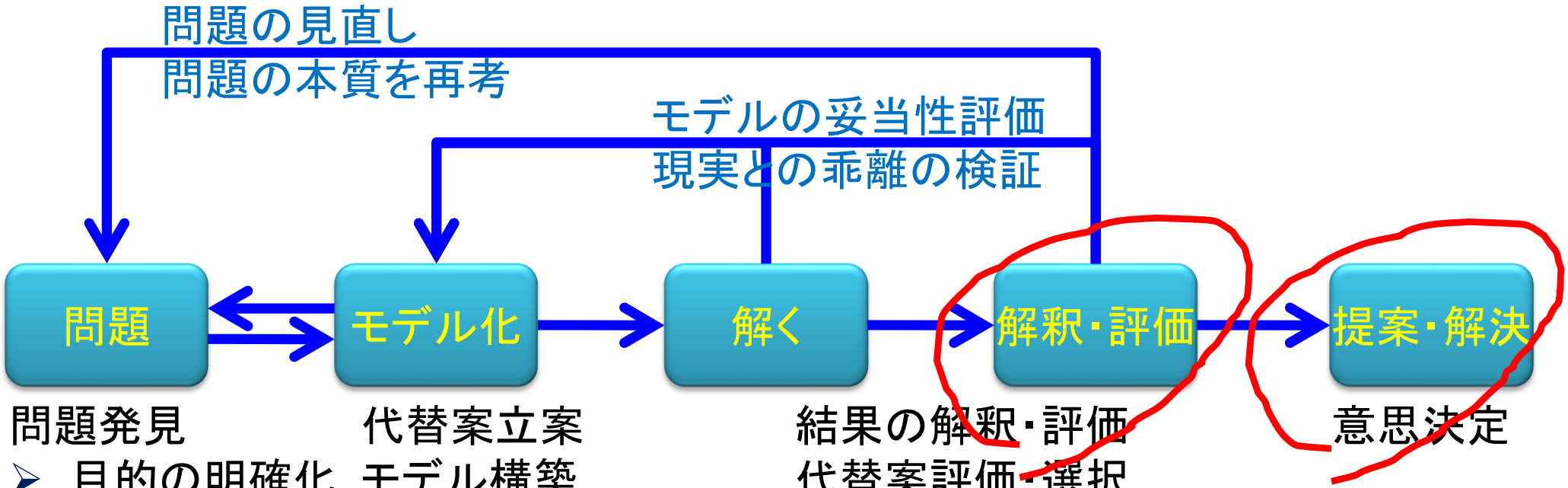
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

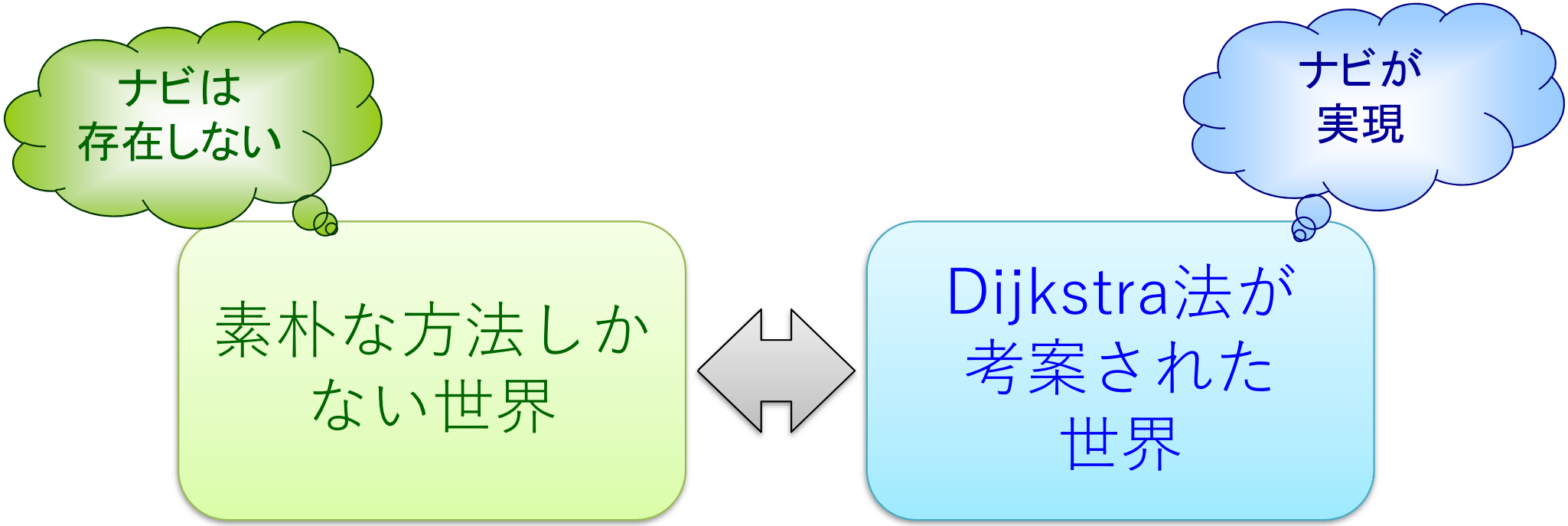
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

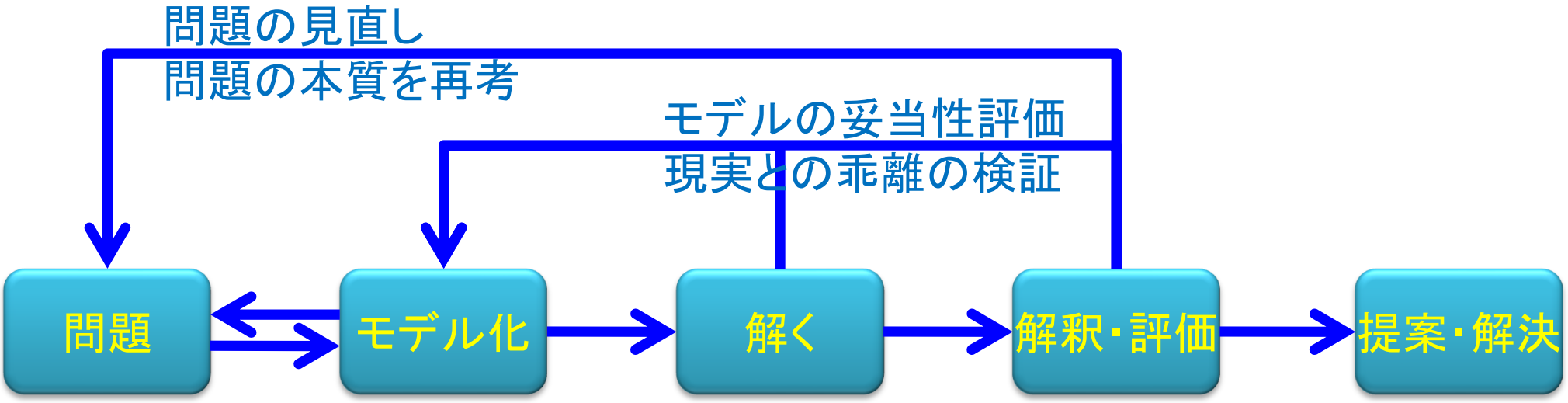
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



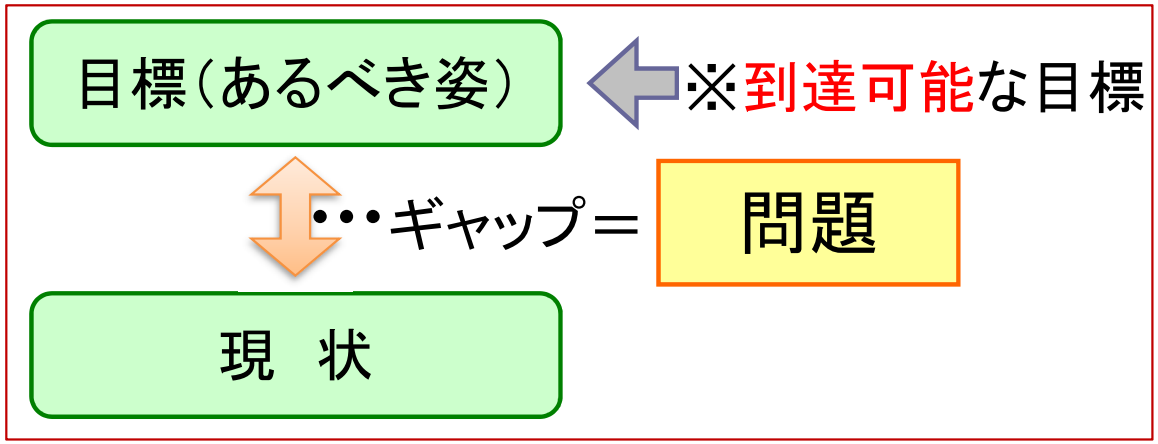
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

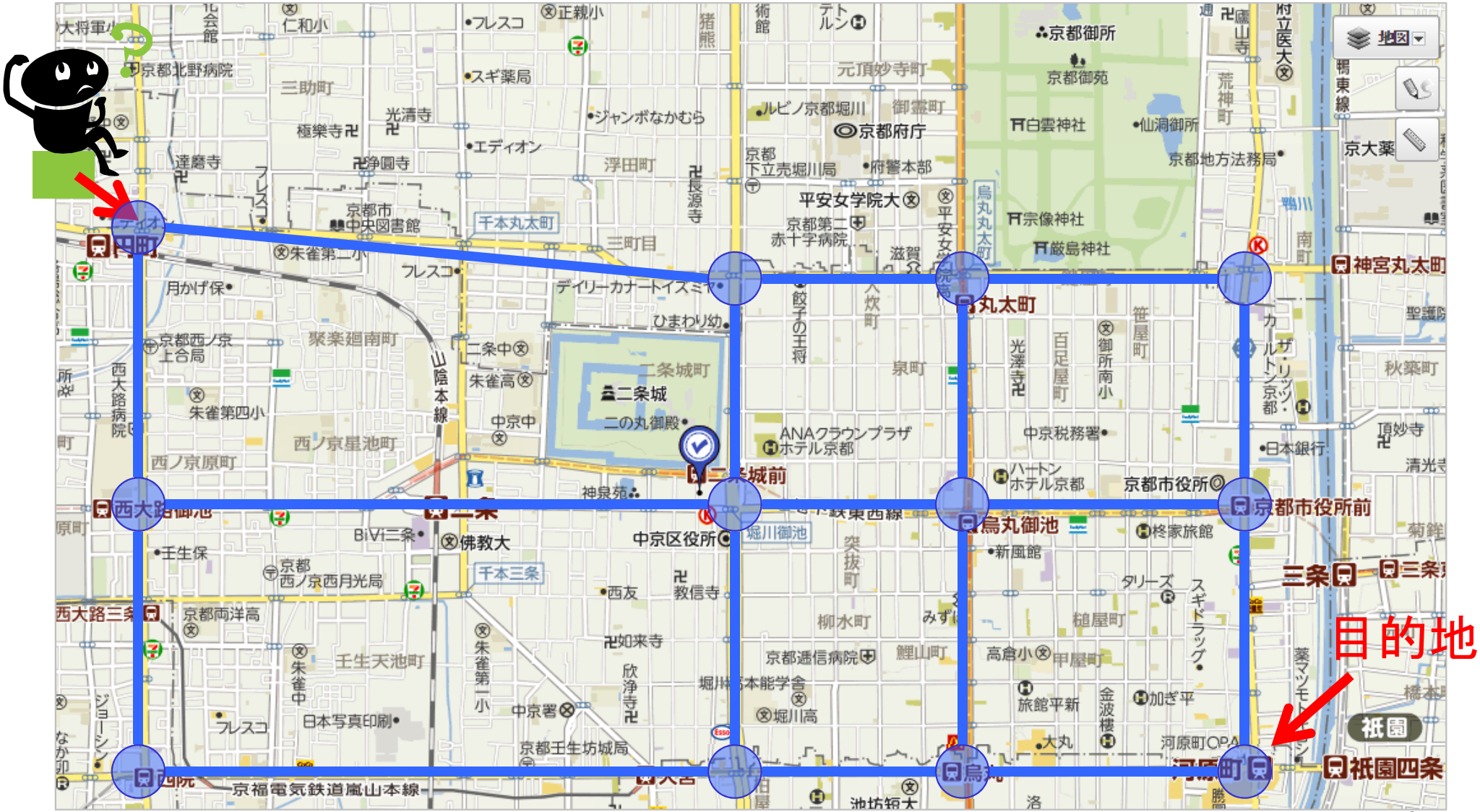
意思決定

問題の定義



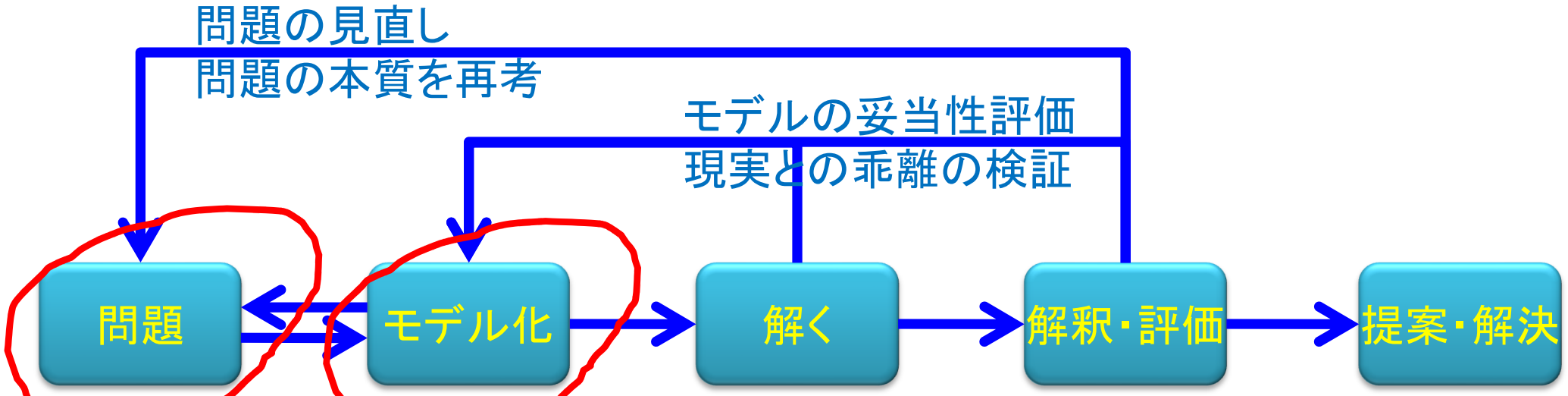
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

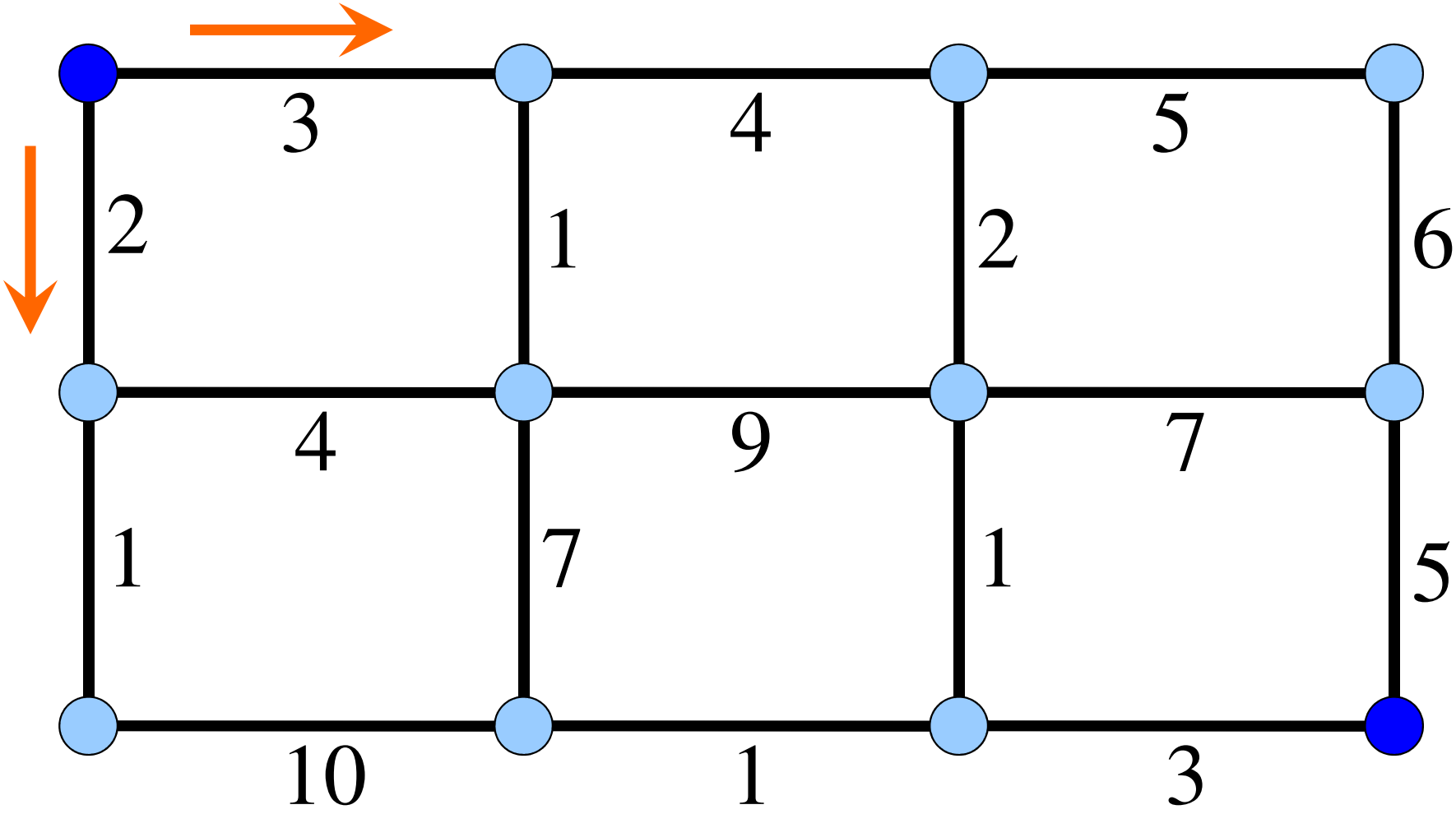
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

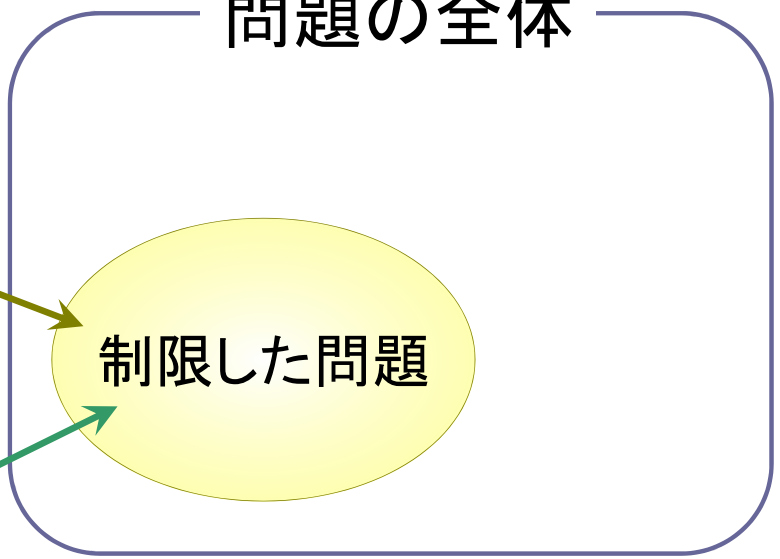


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

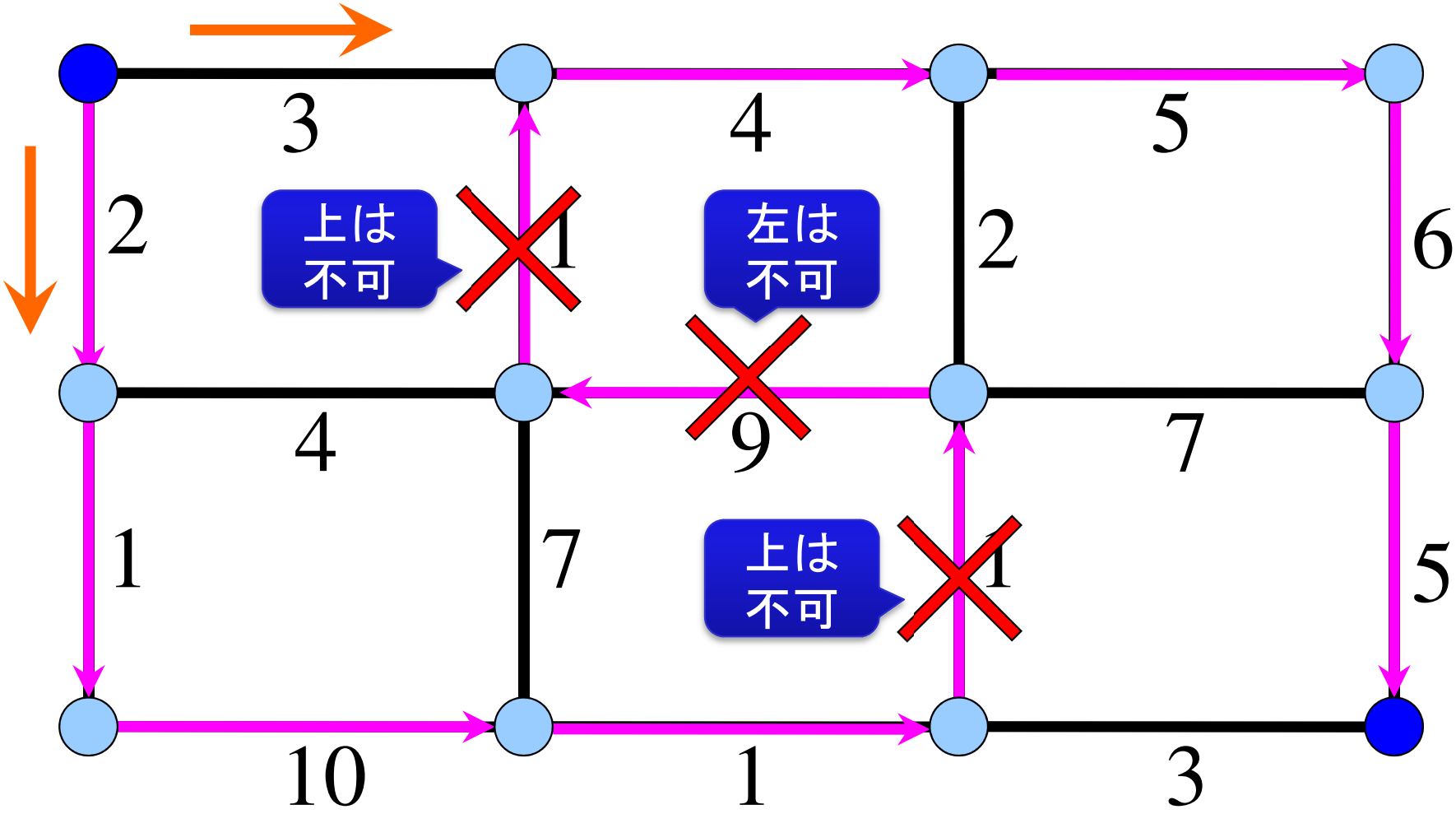
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

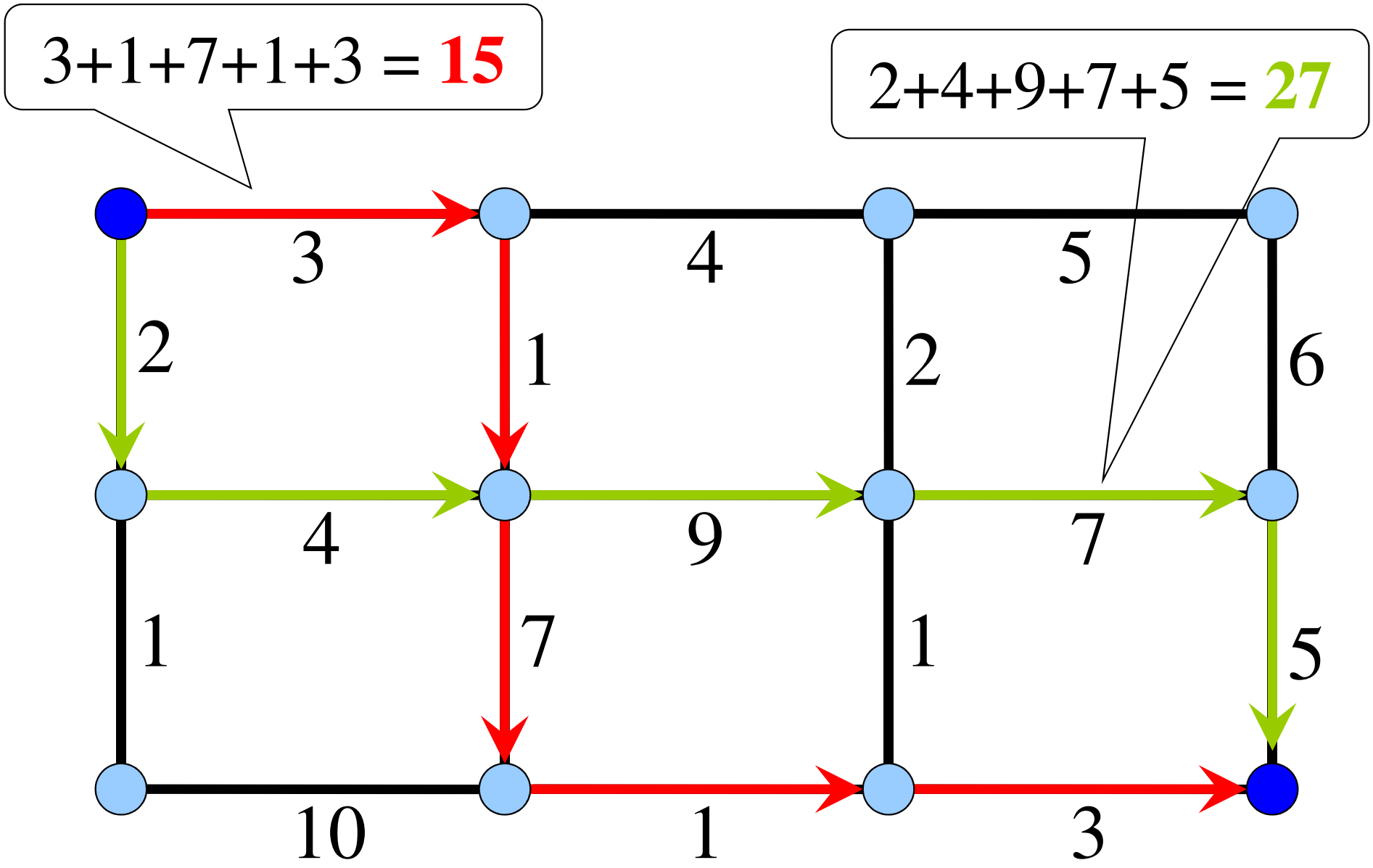
難しいなら易しくすればいいのさ！

制限した問題

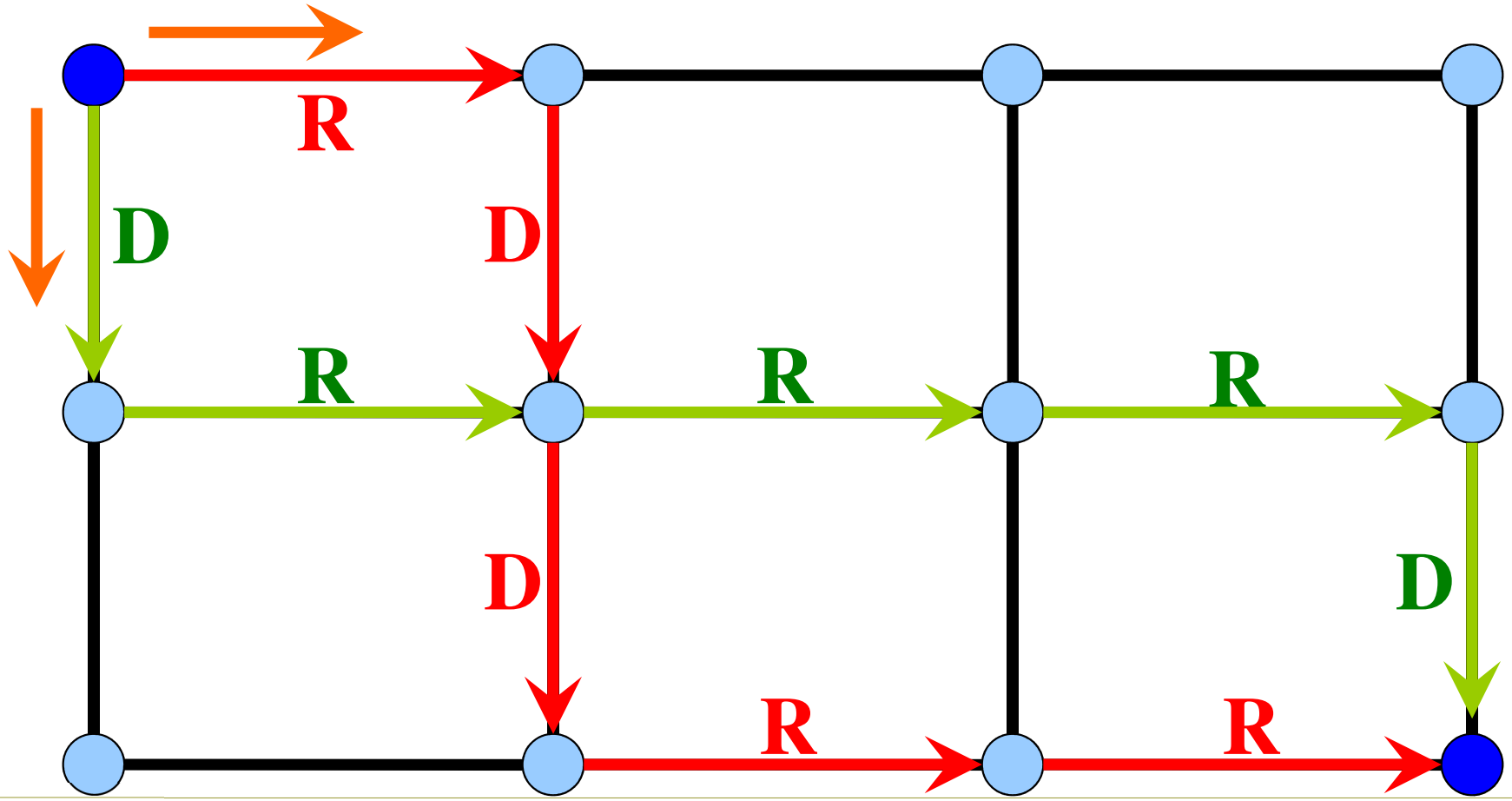
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



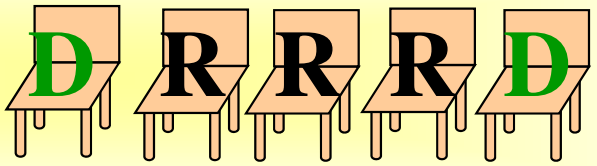
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



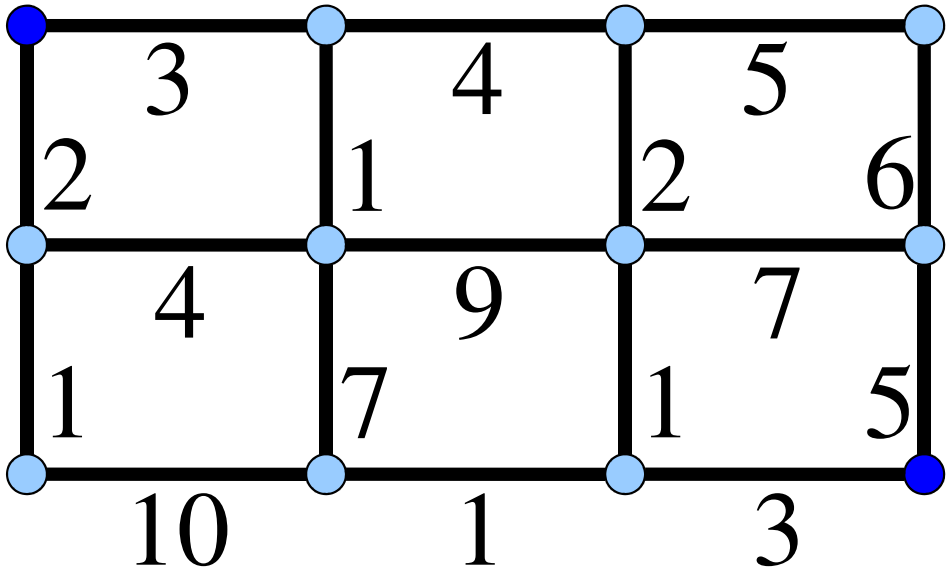
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

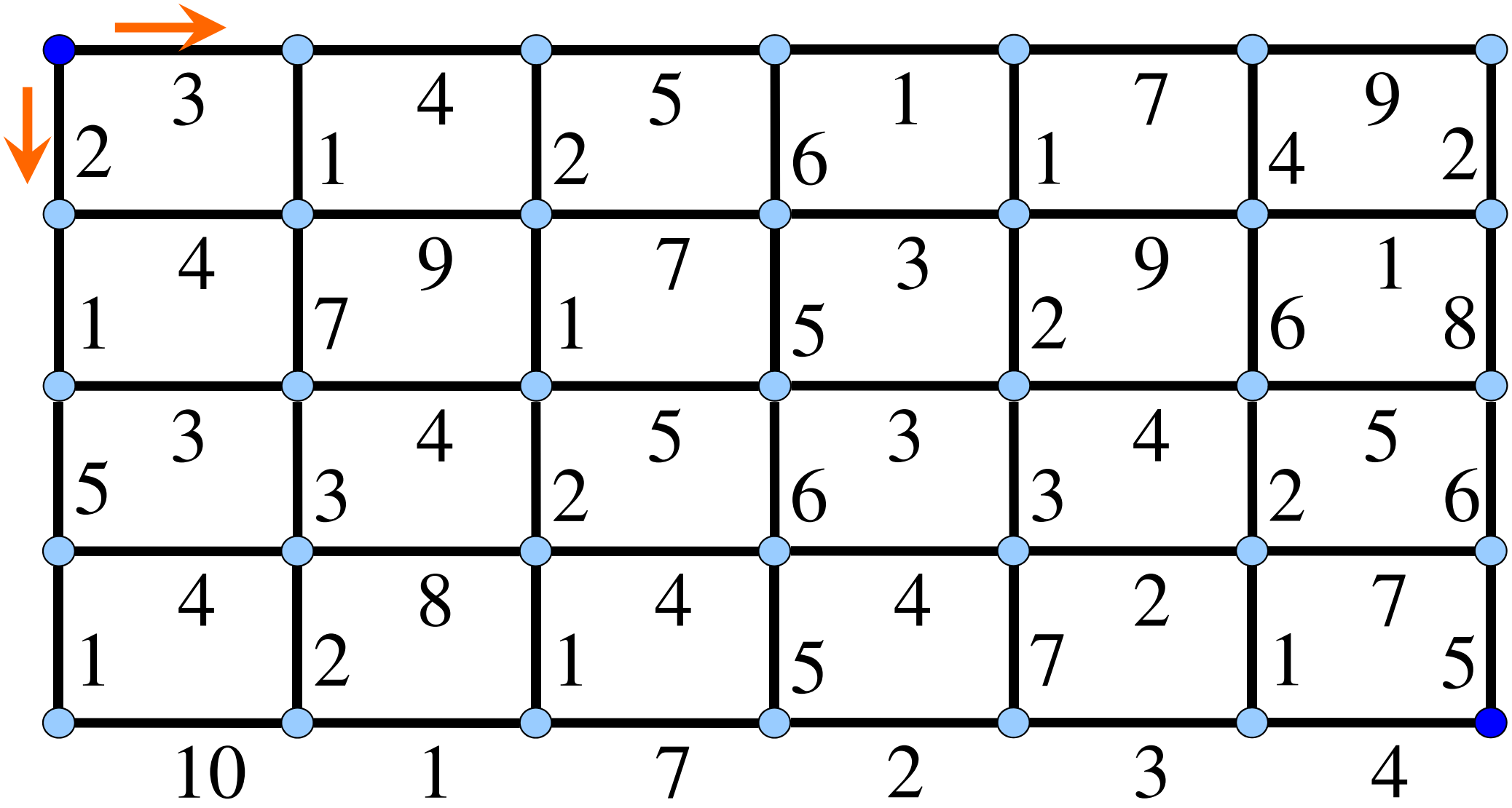
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

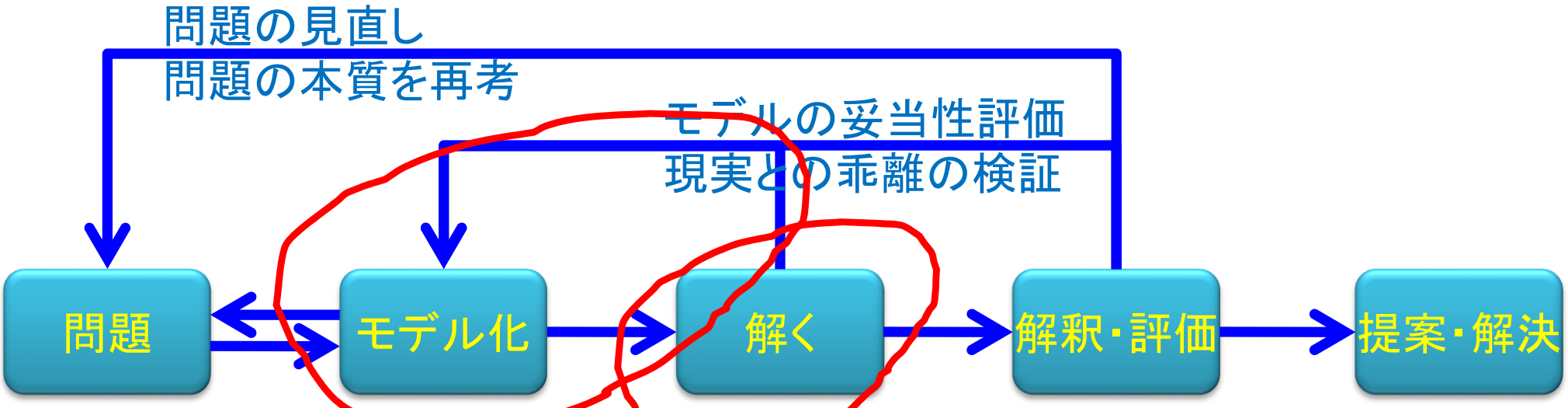
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

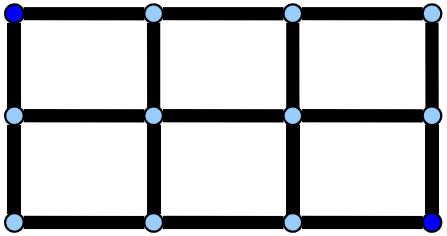
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

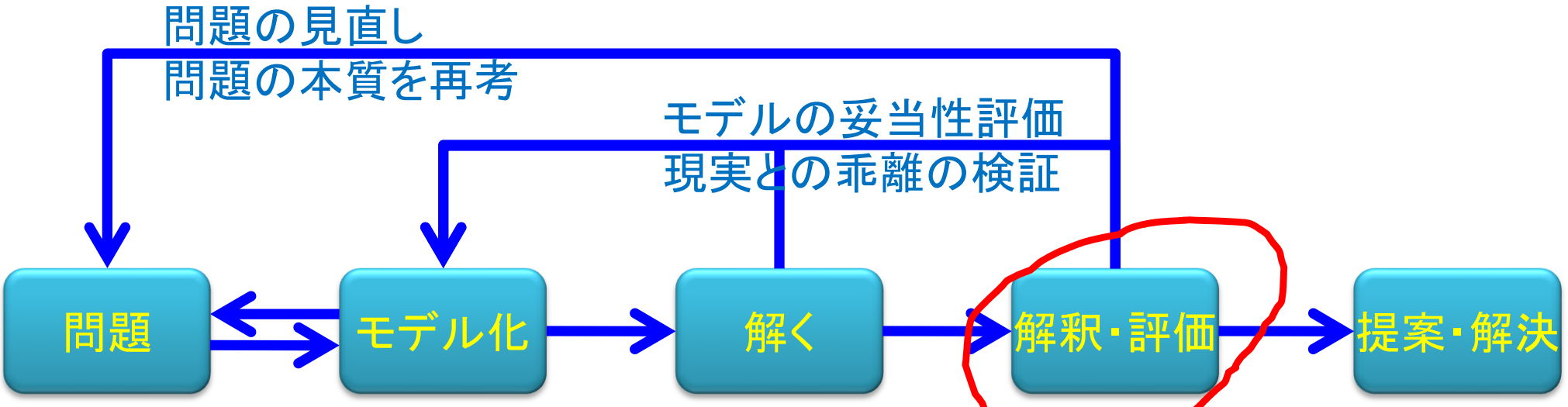
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

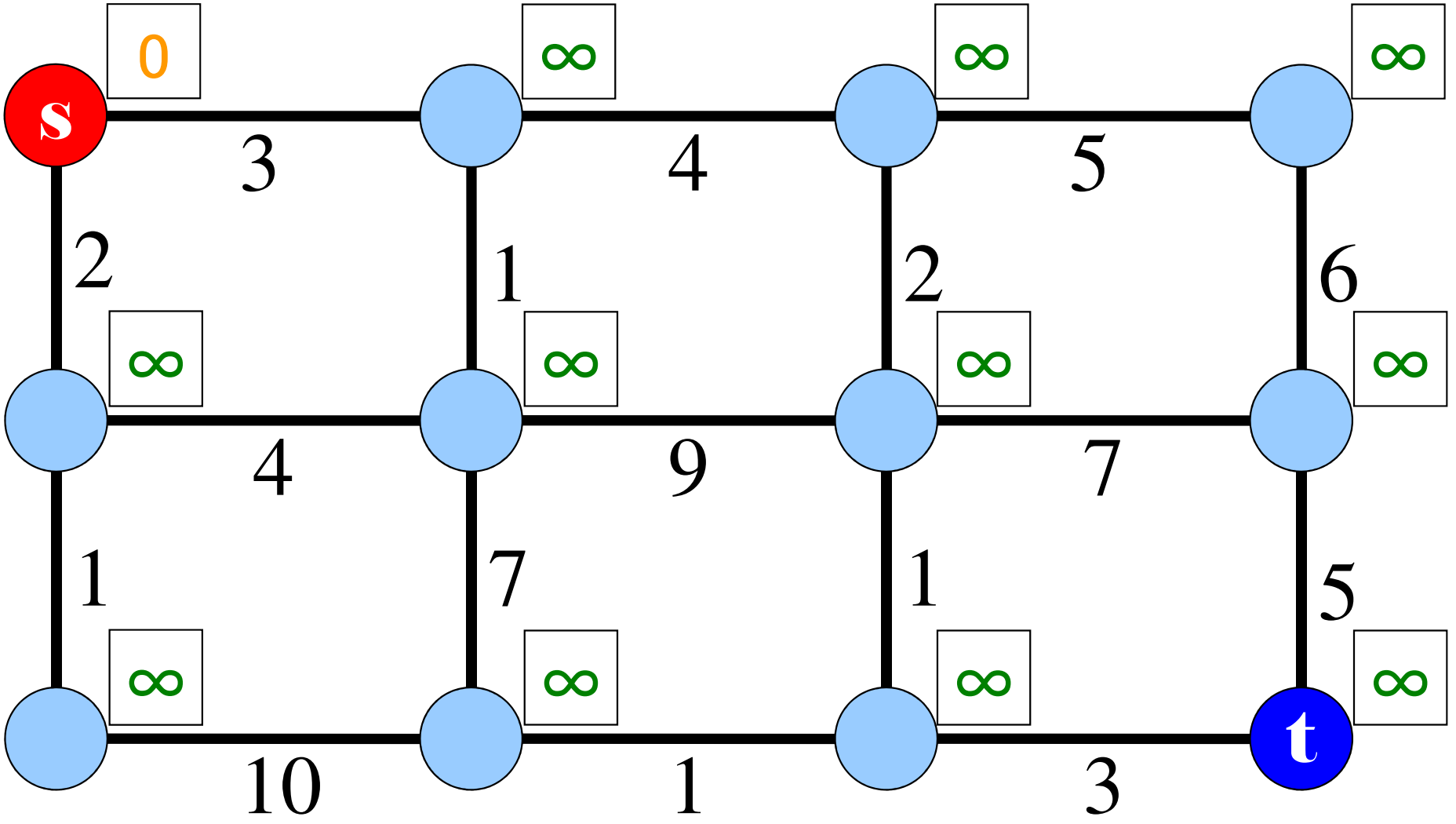
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

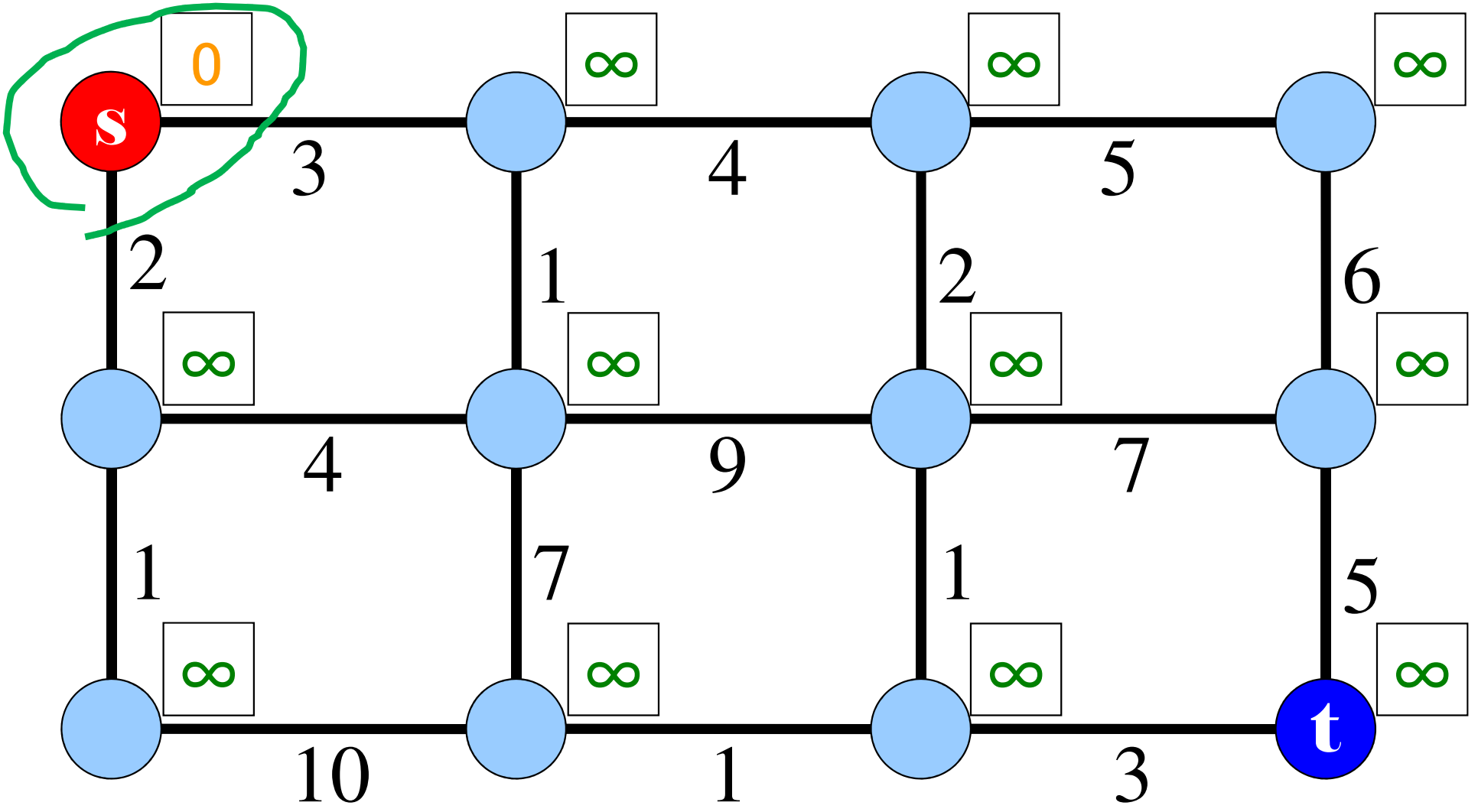
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



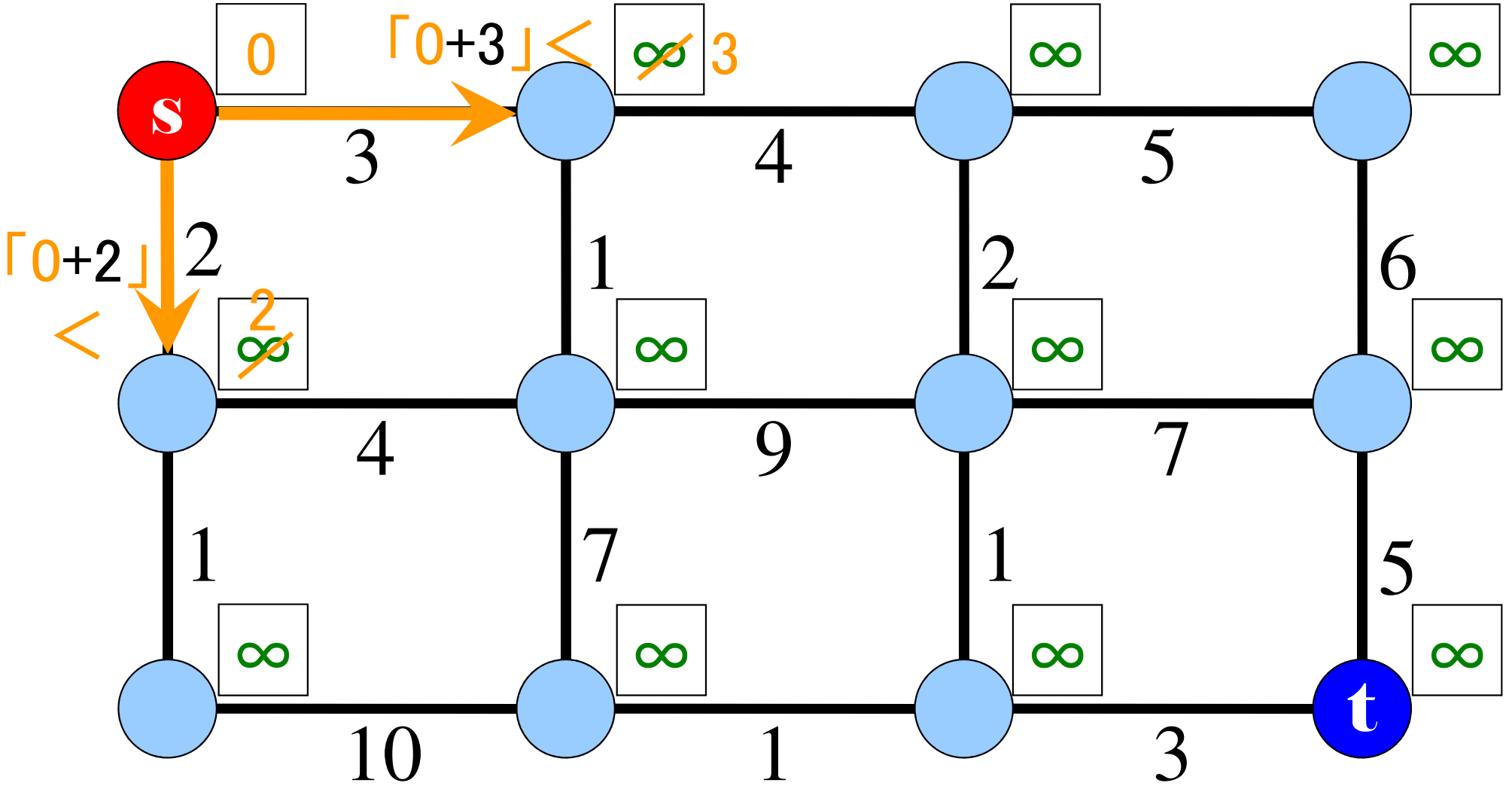
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

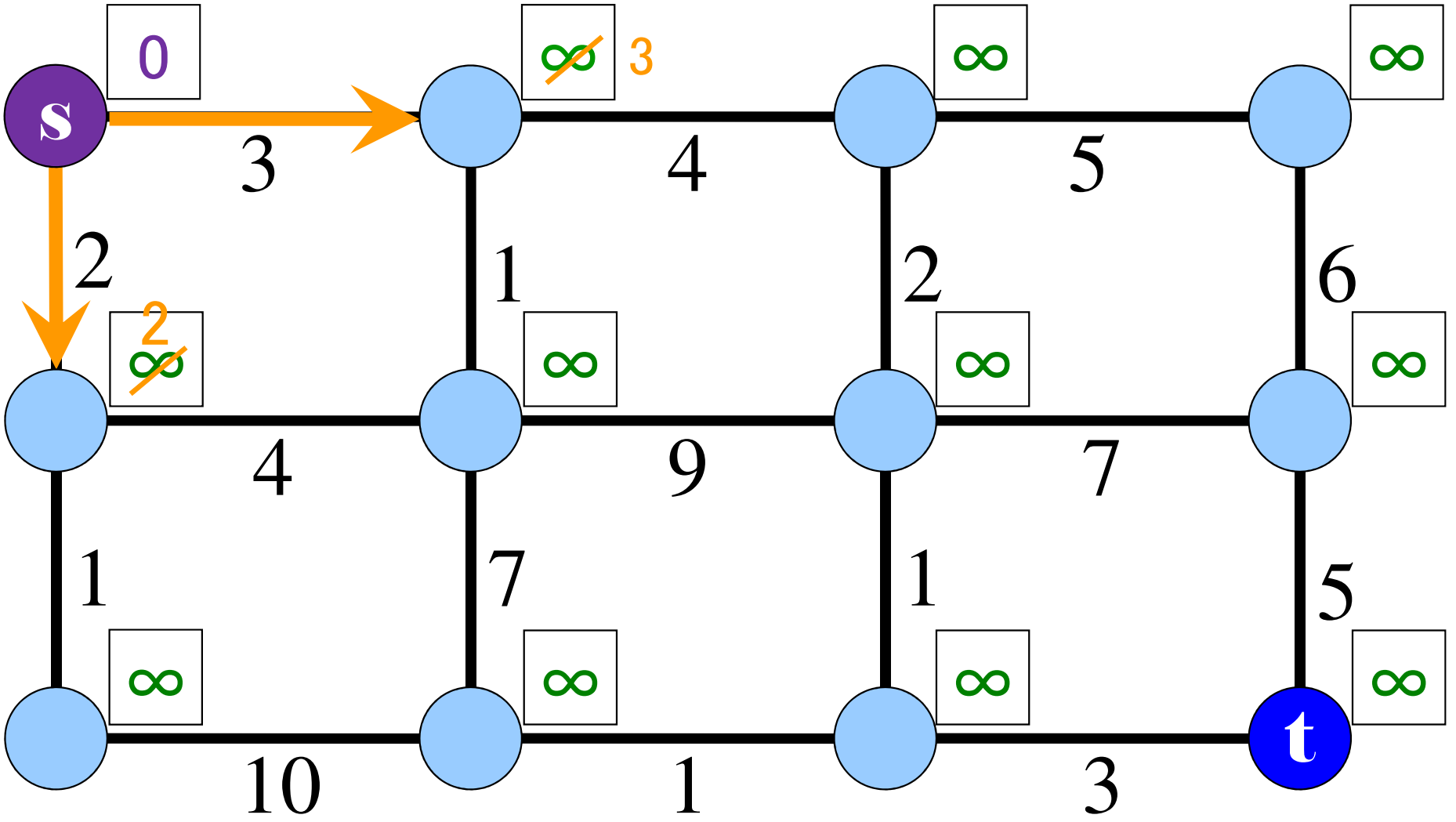


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

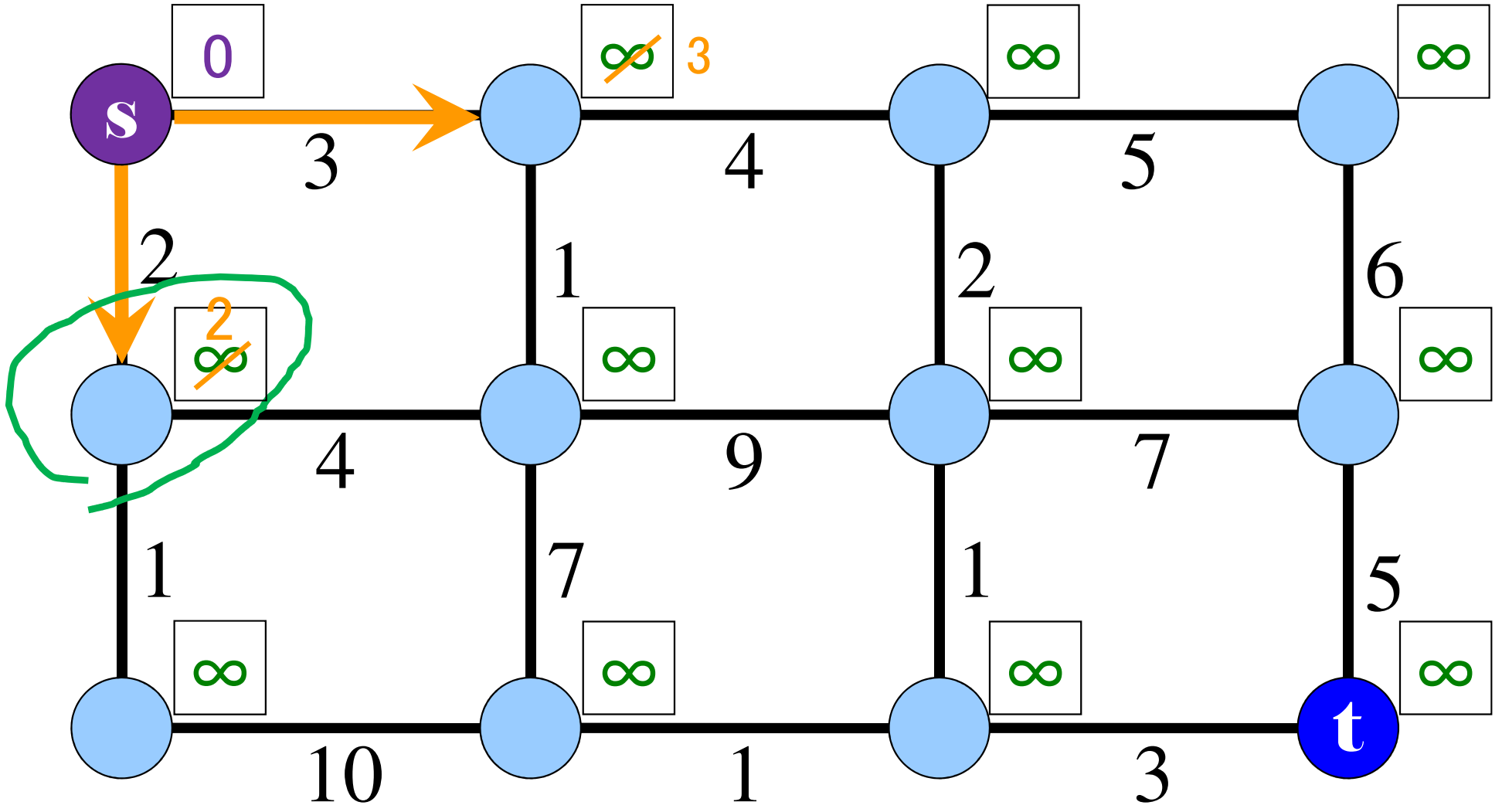
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



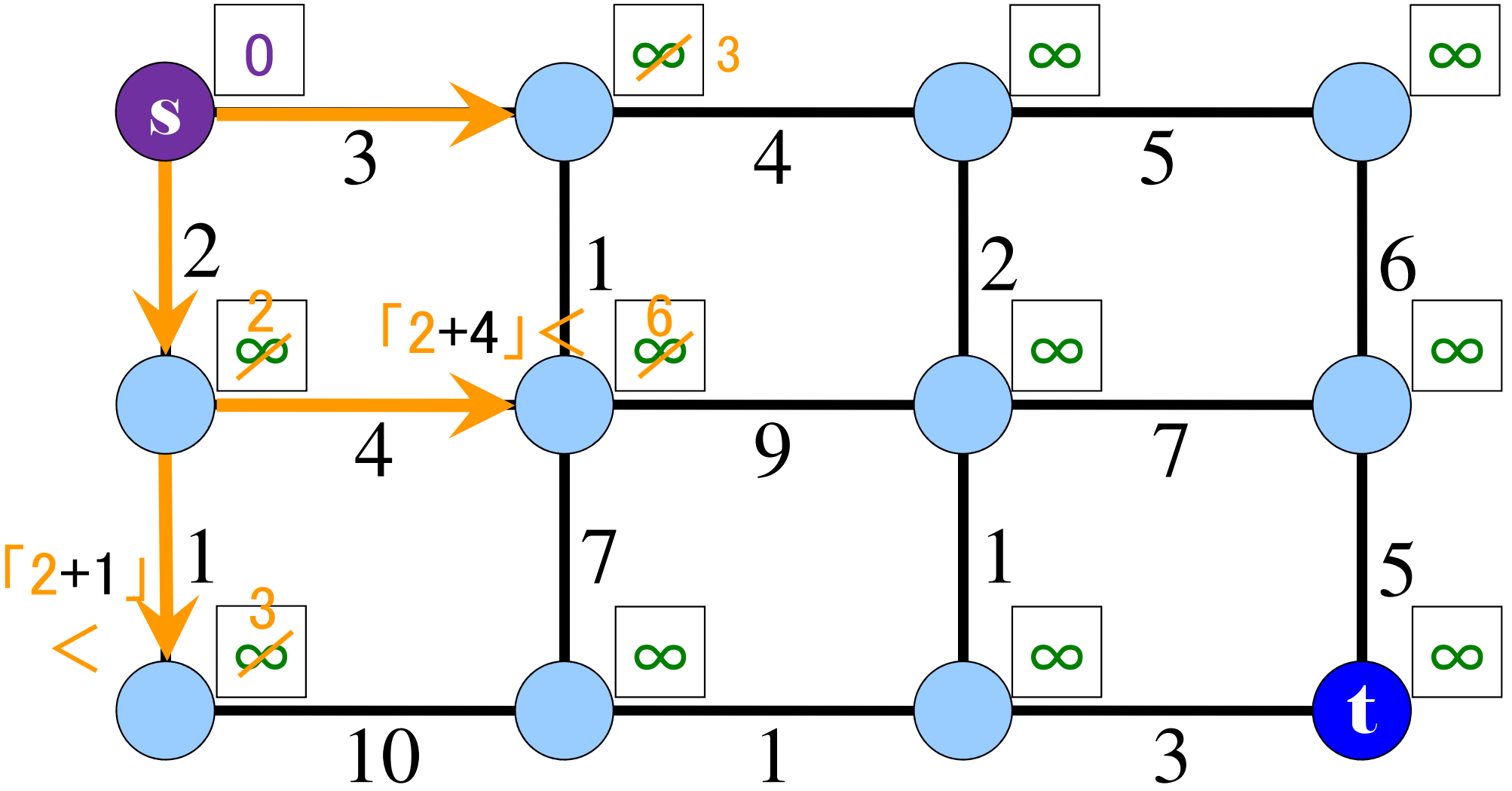
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

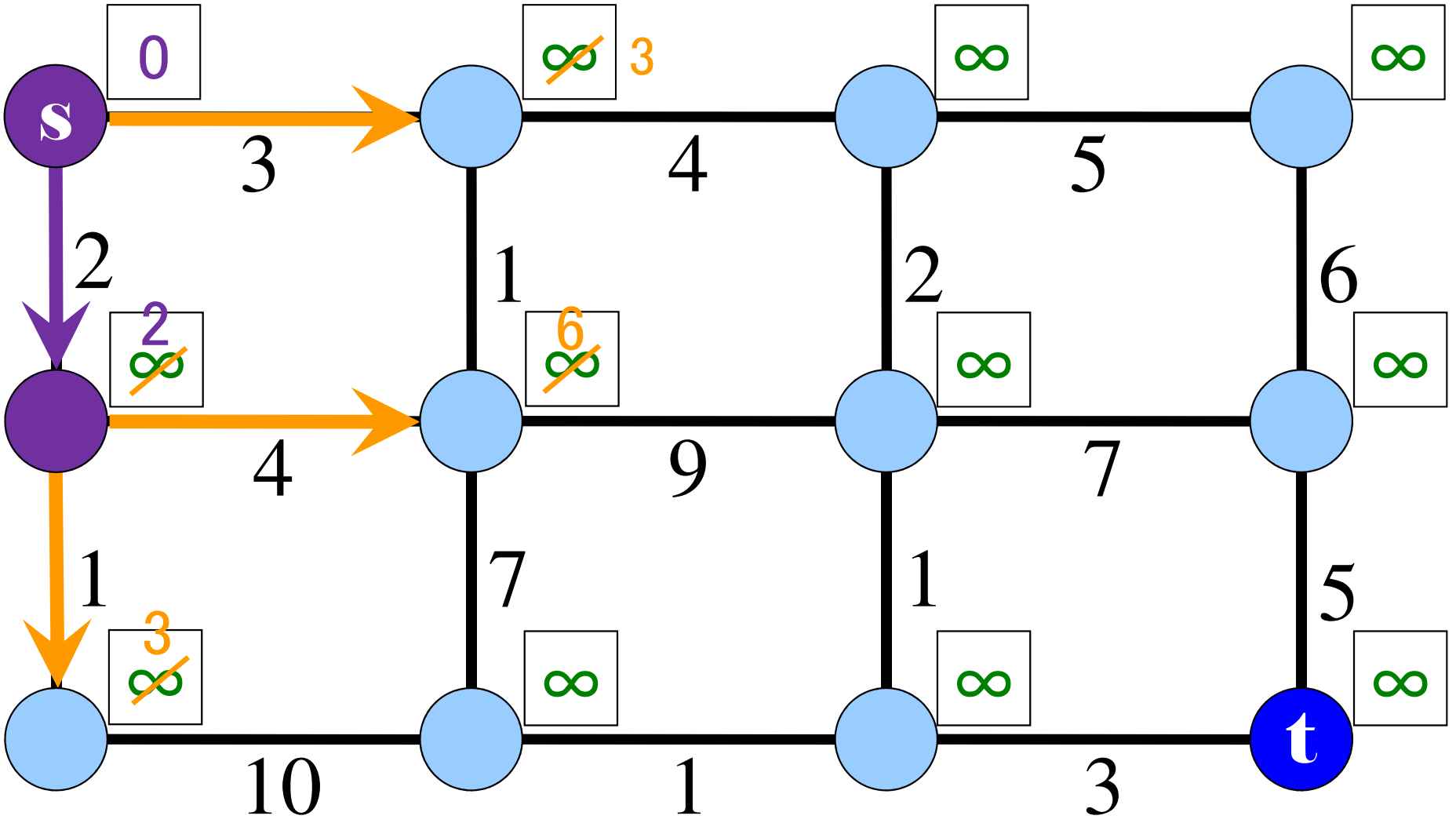


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

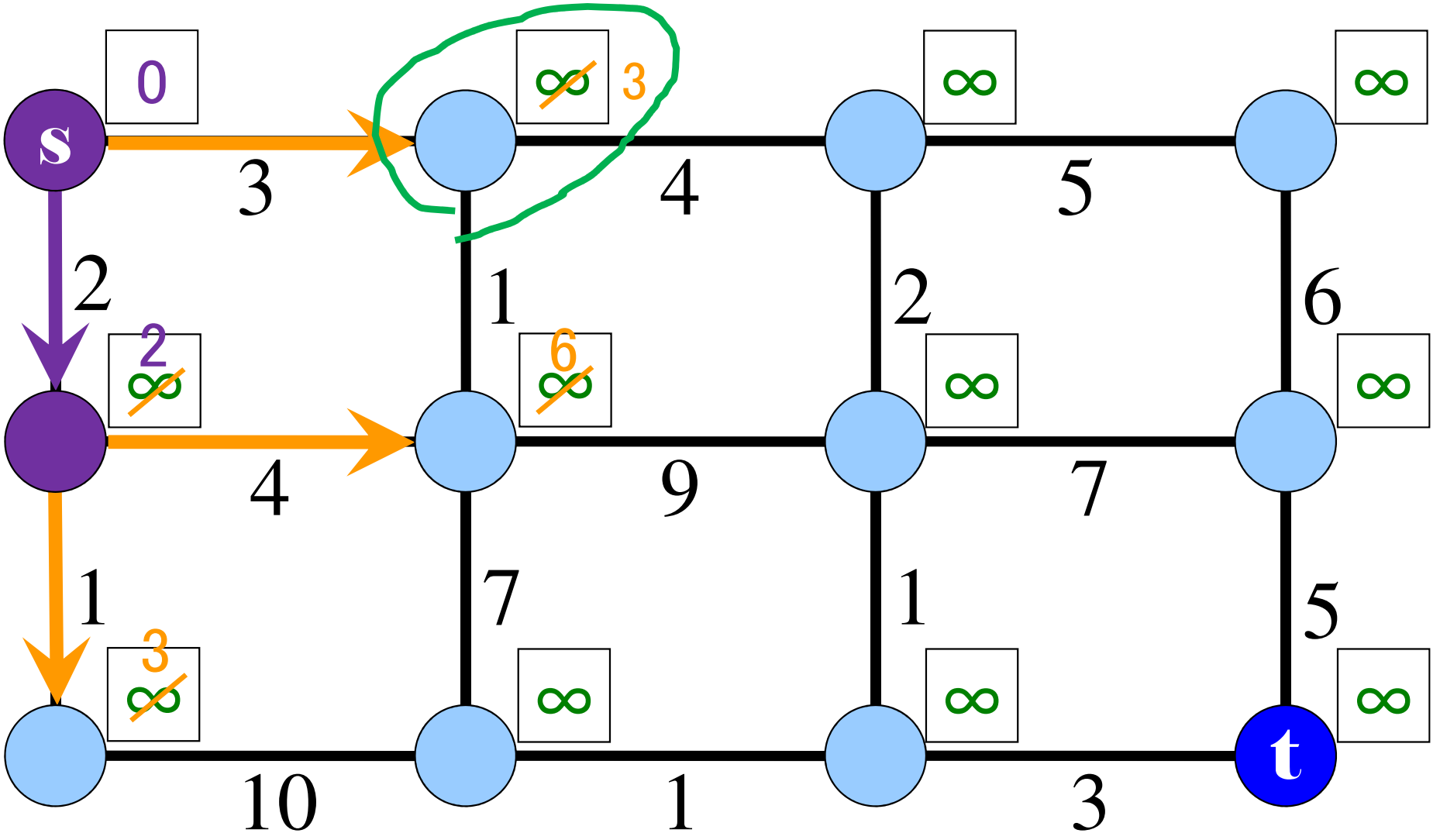
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



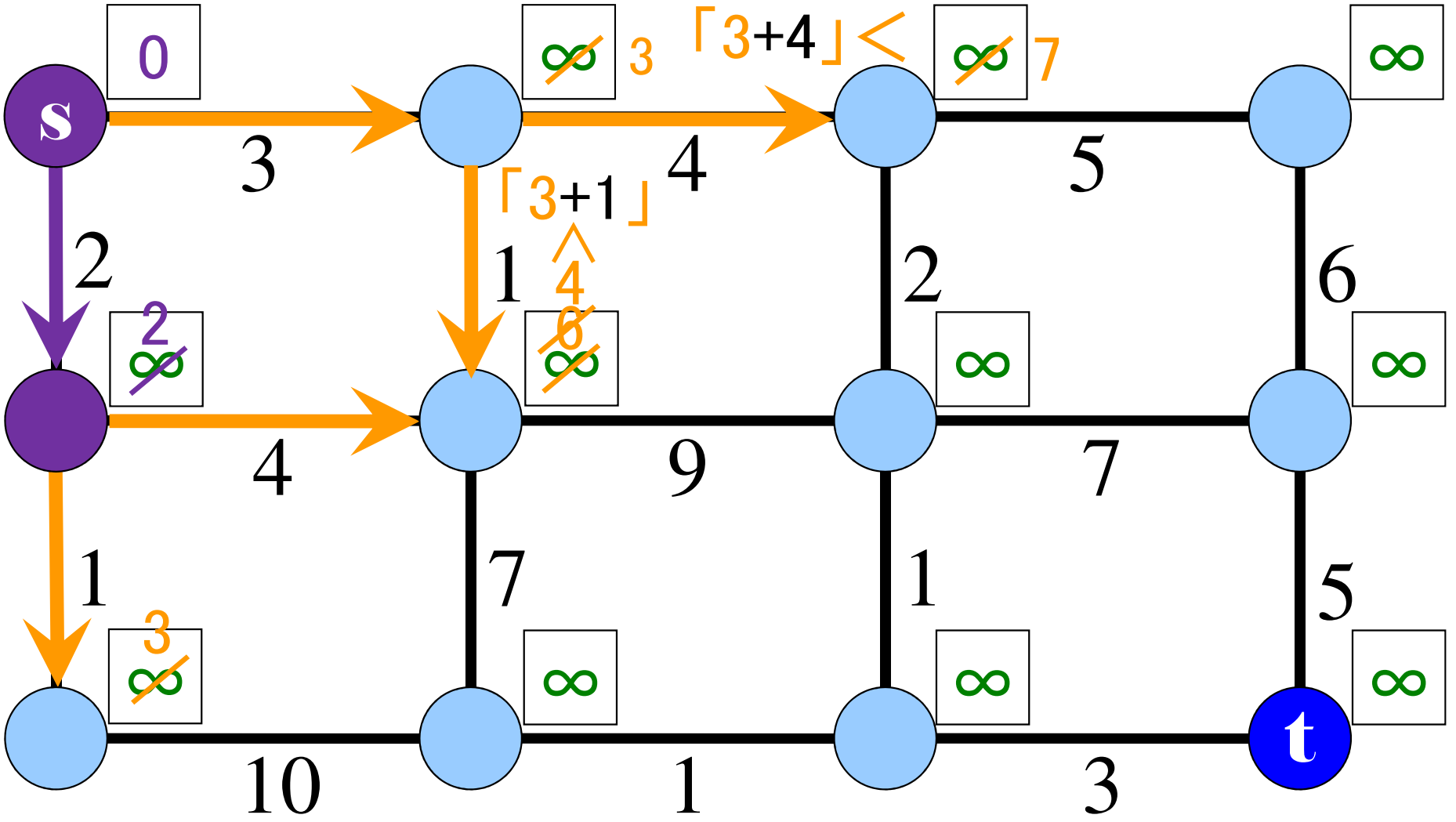
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

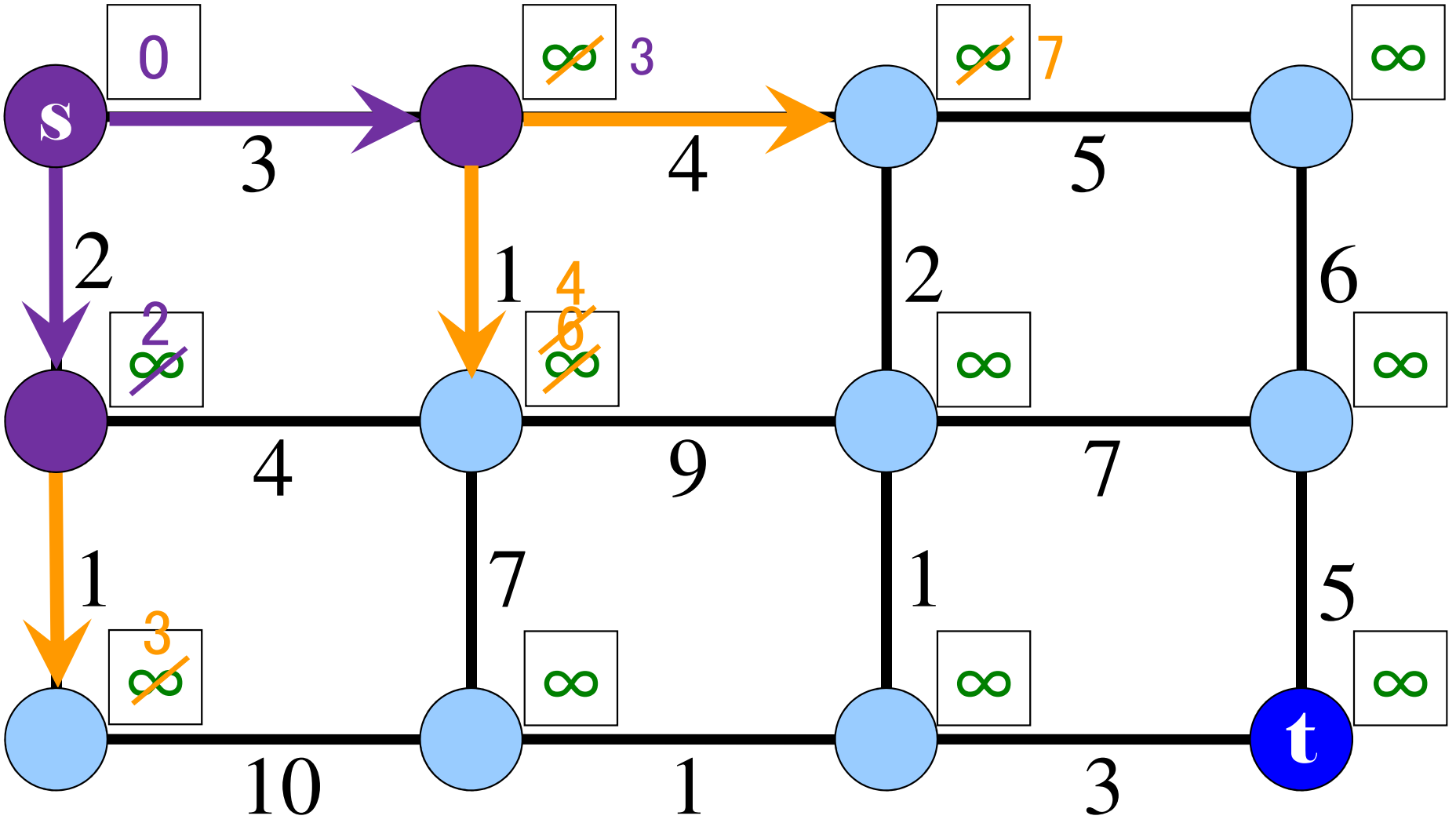


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

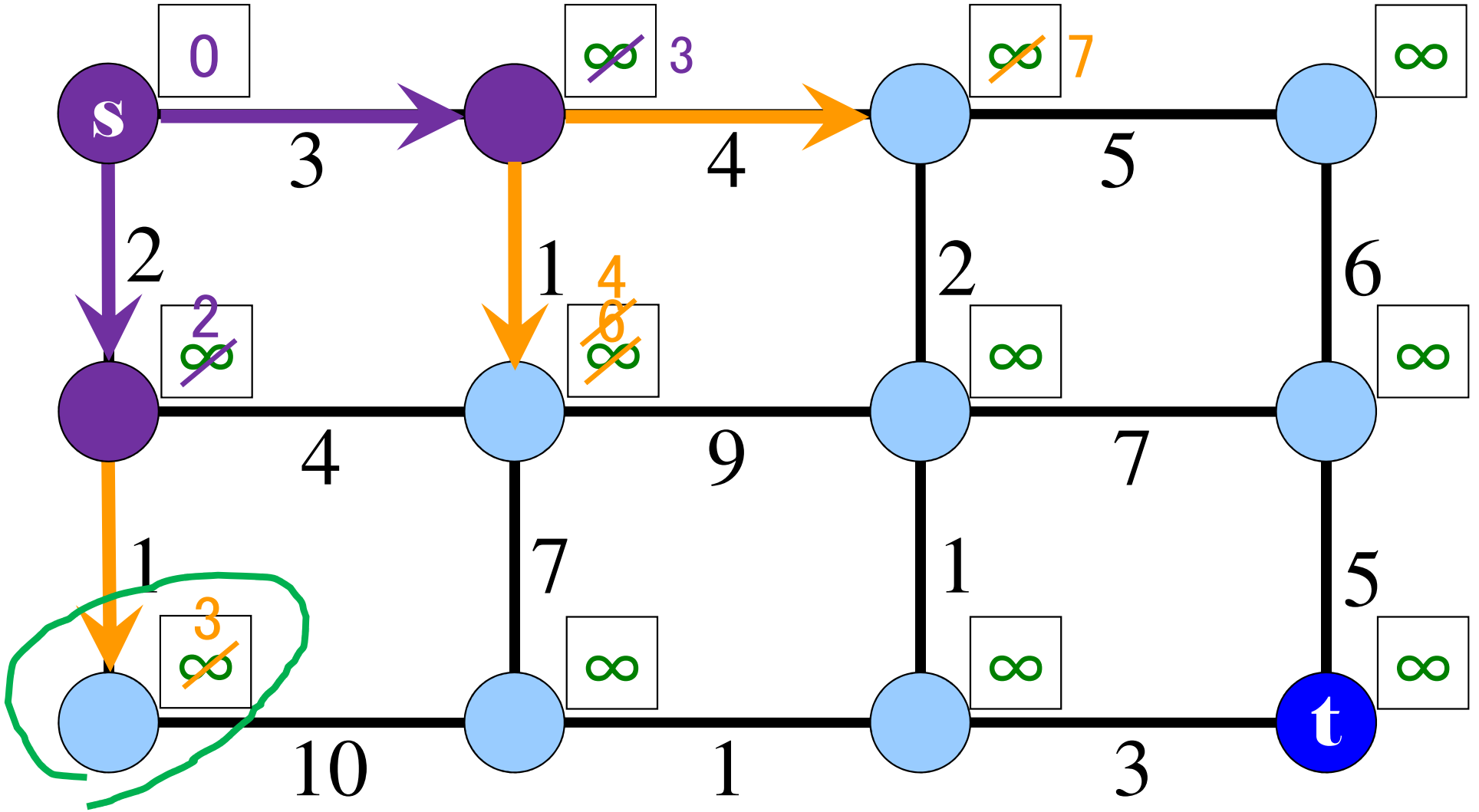
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



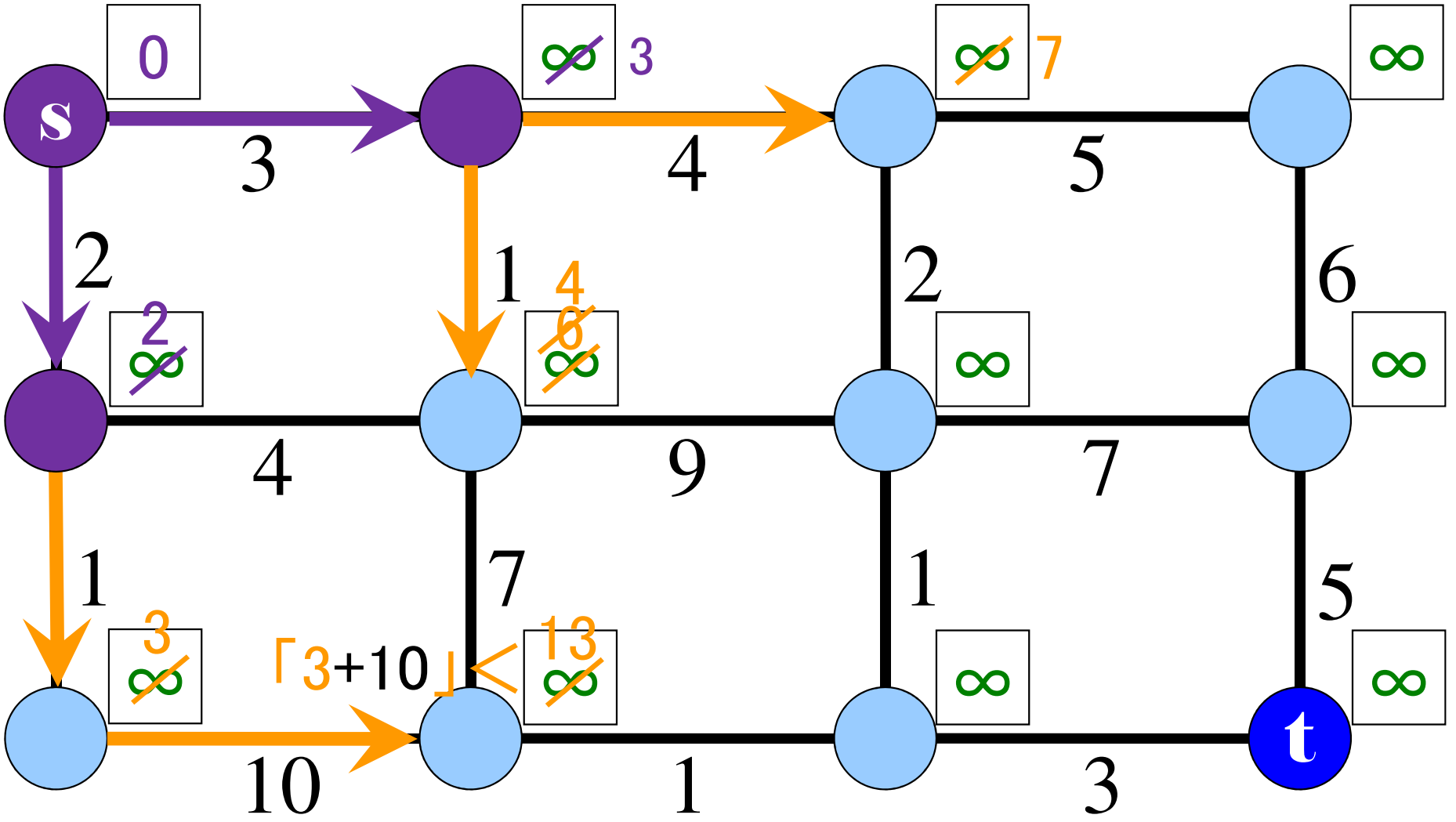
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

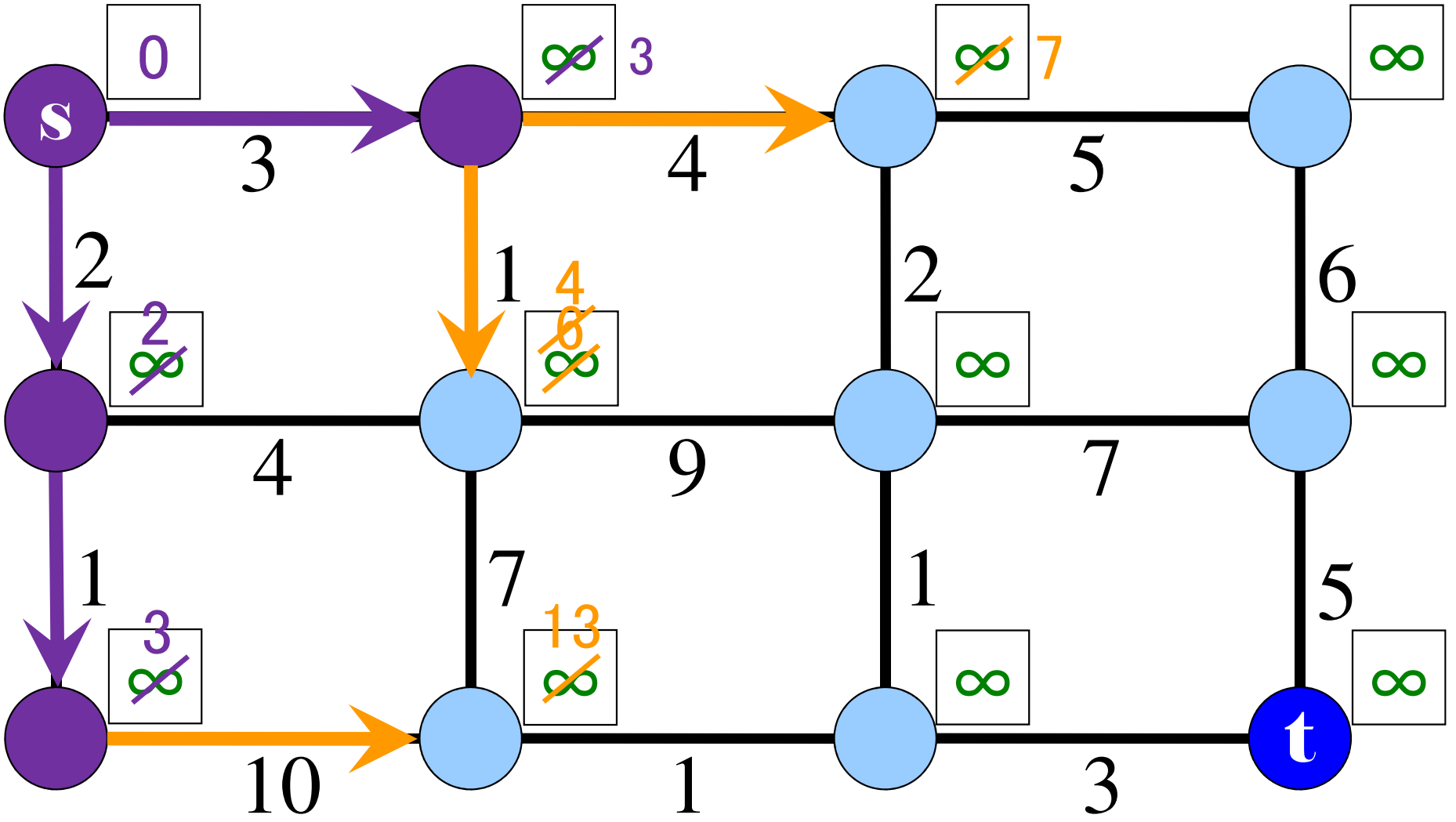


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

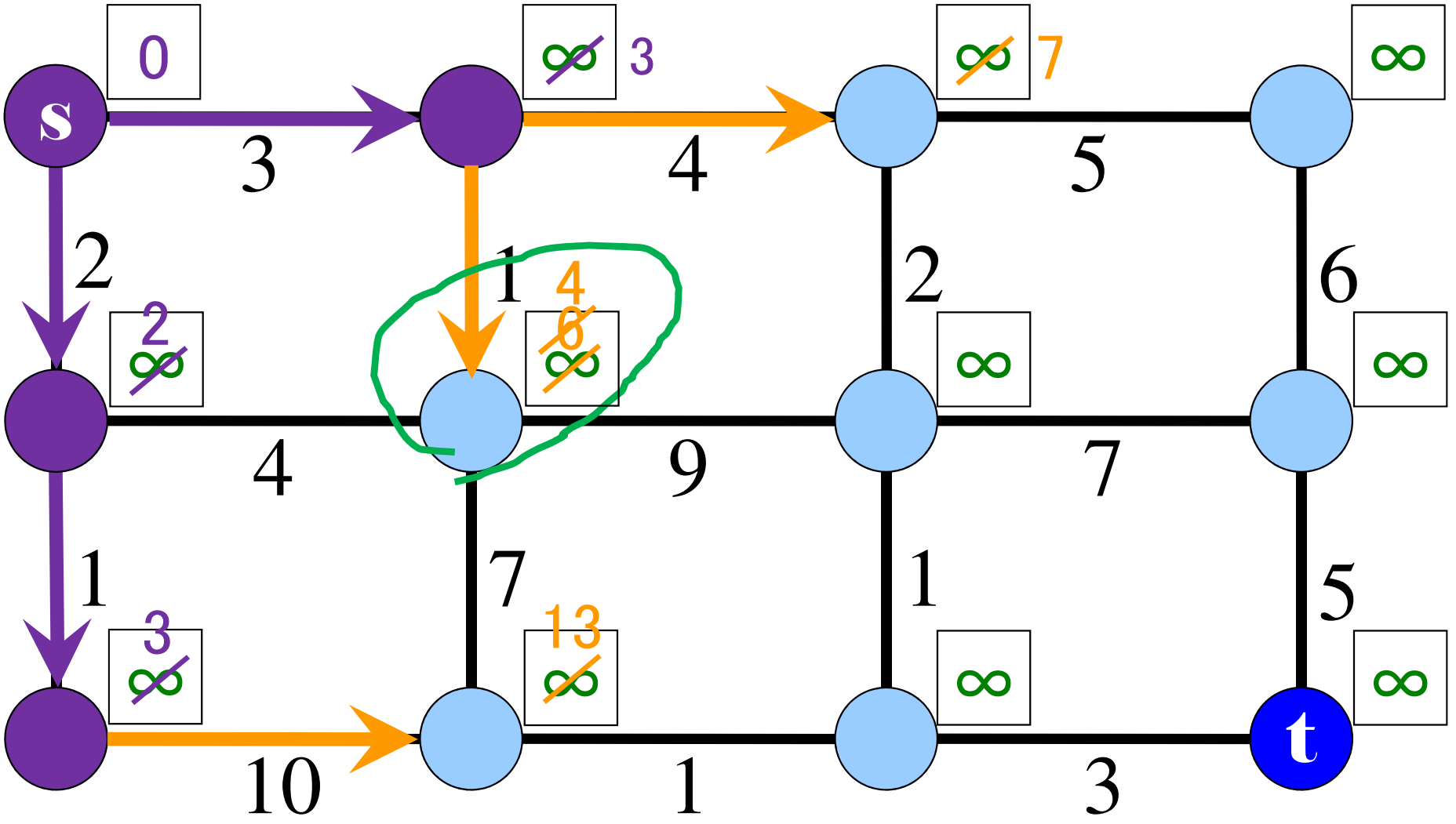
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



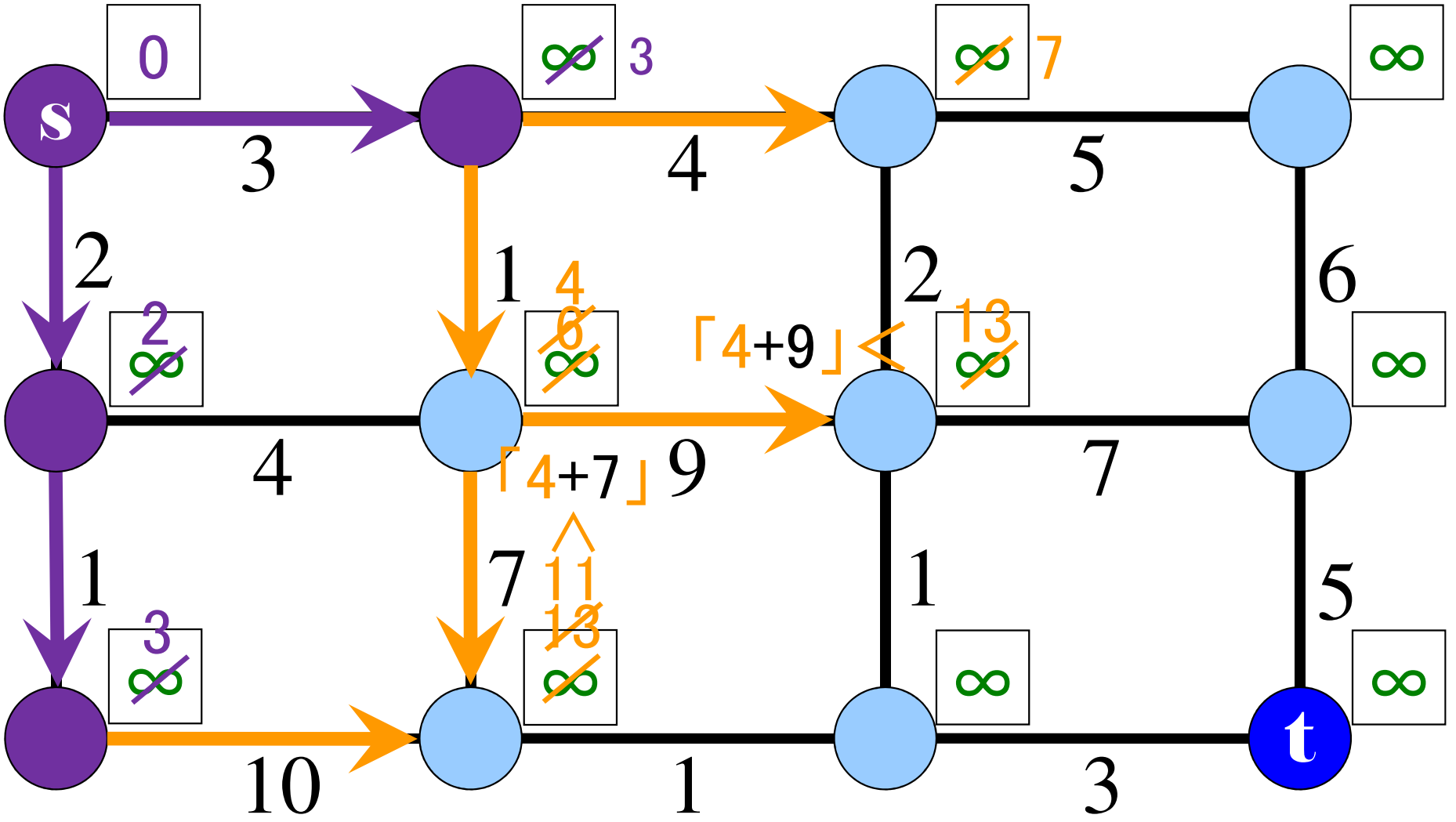
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

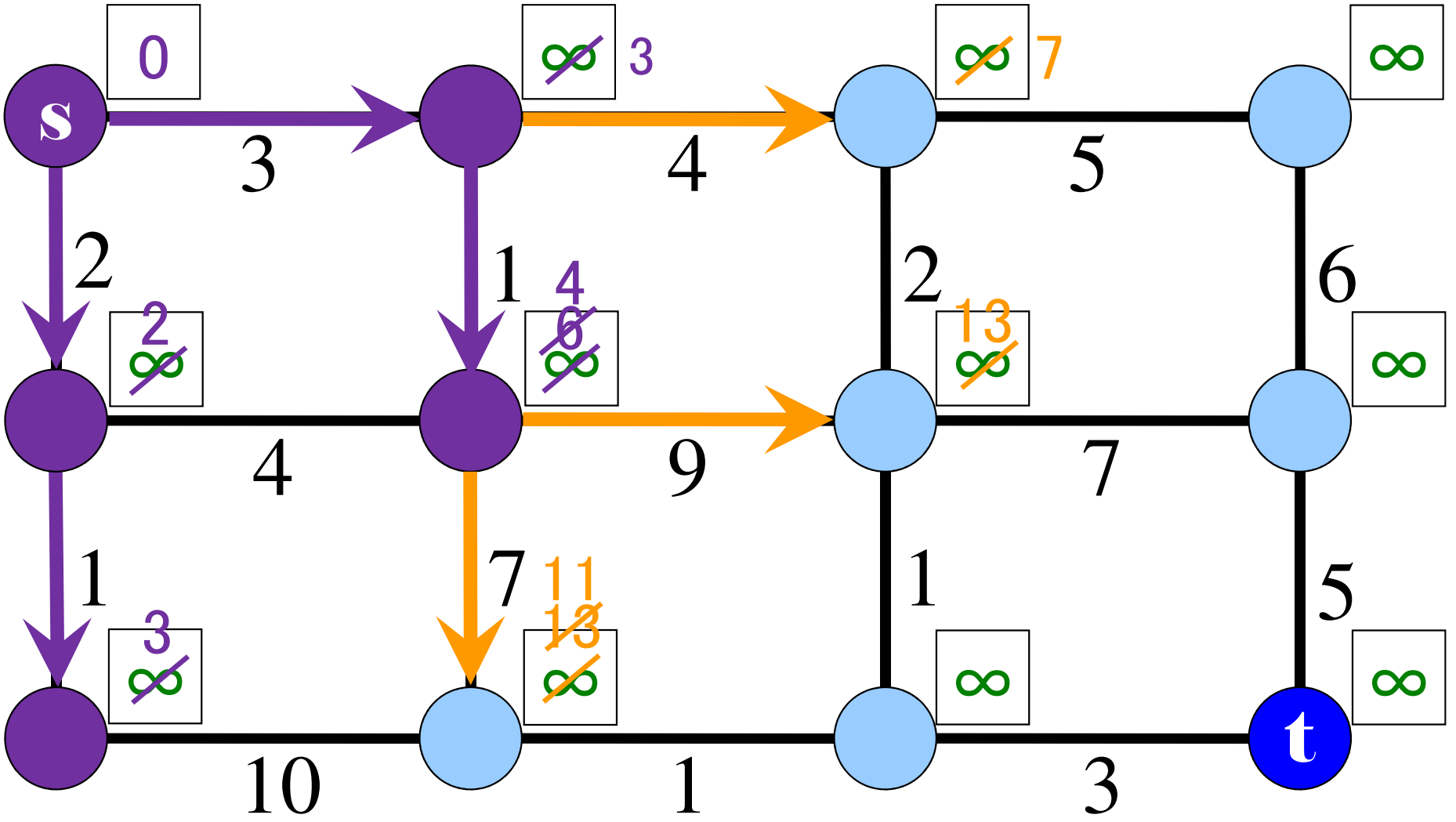


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

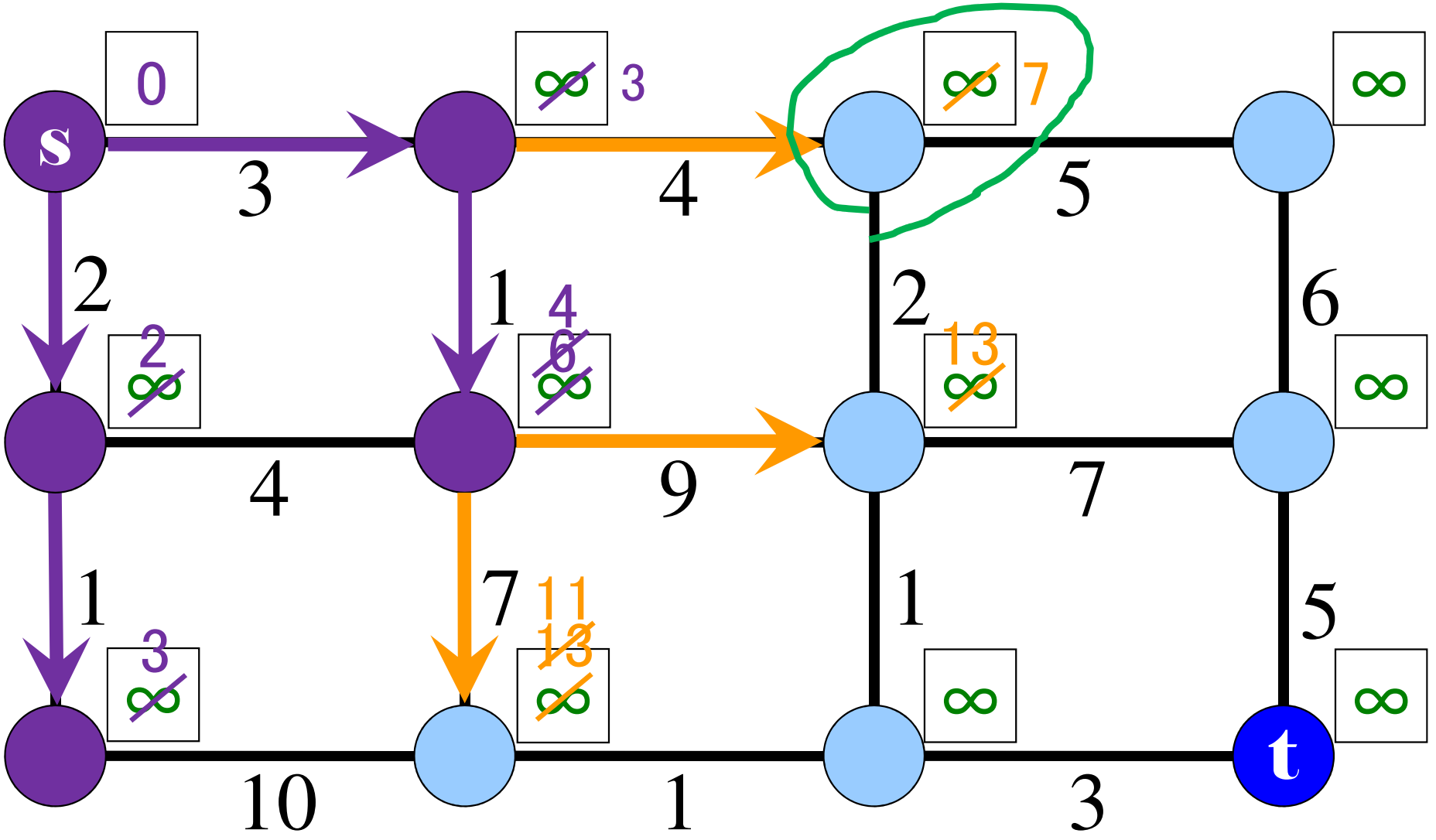
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



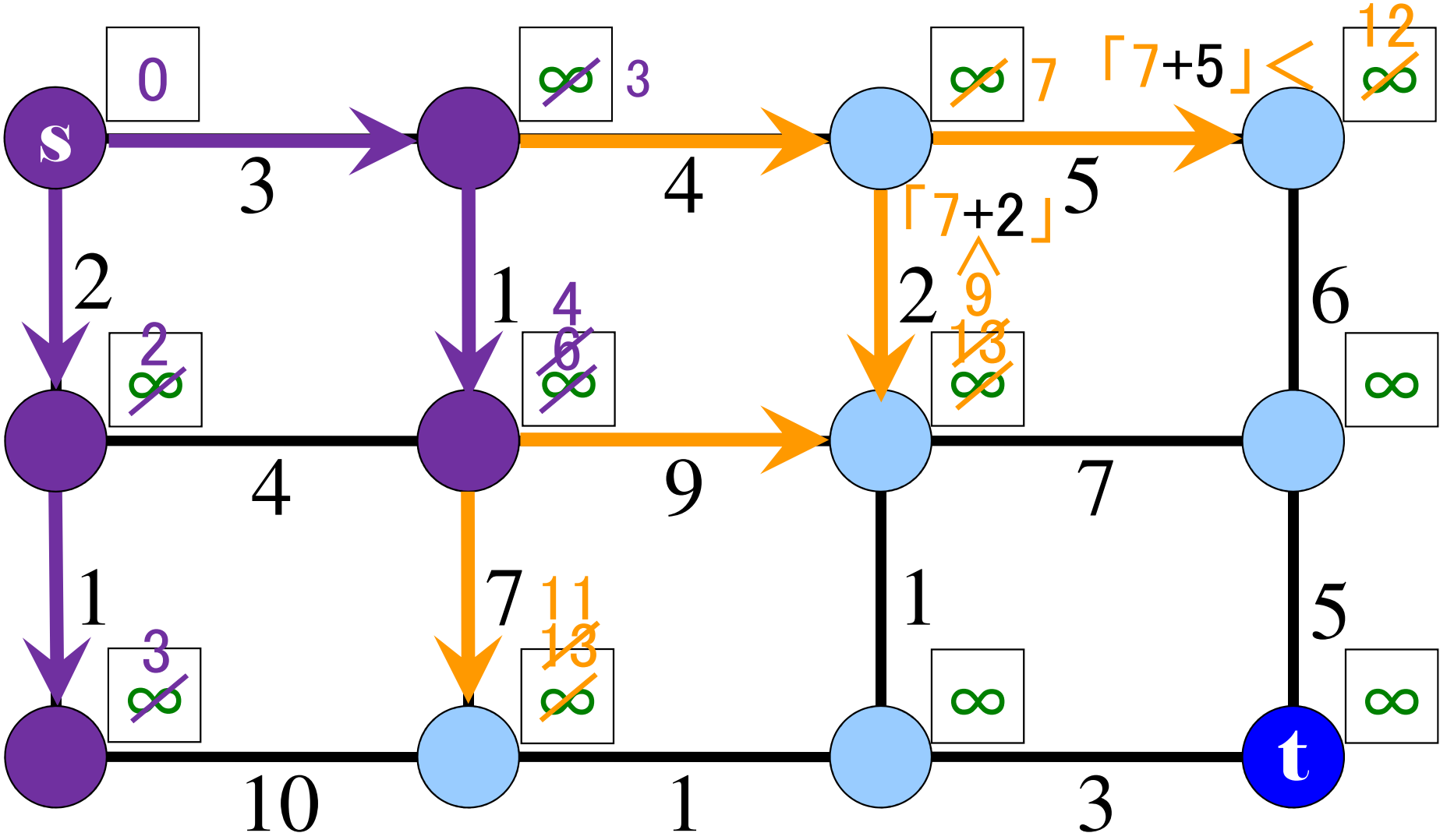
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

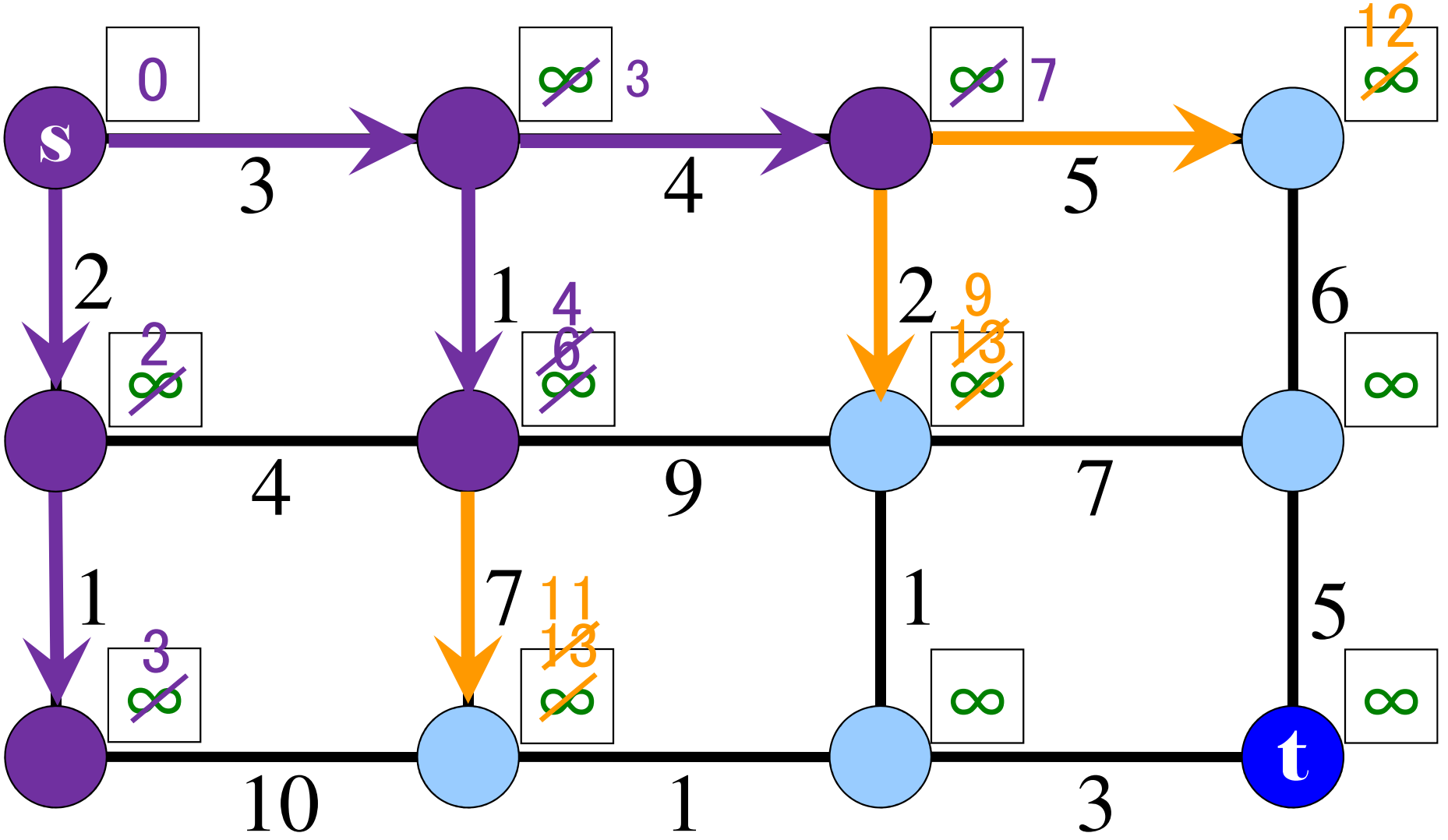


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

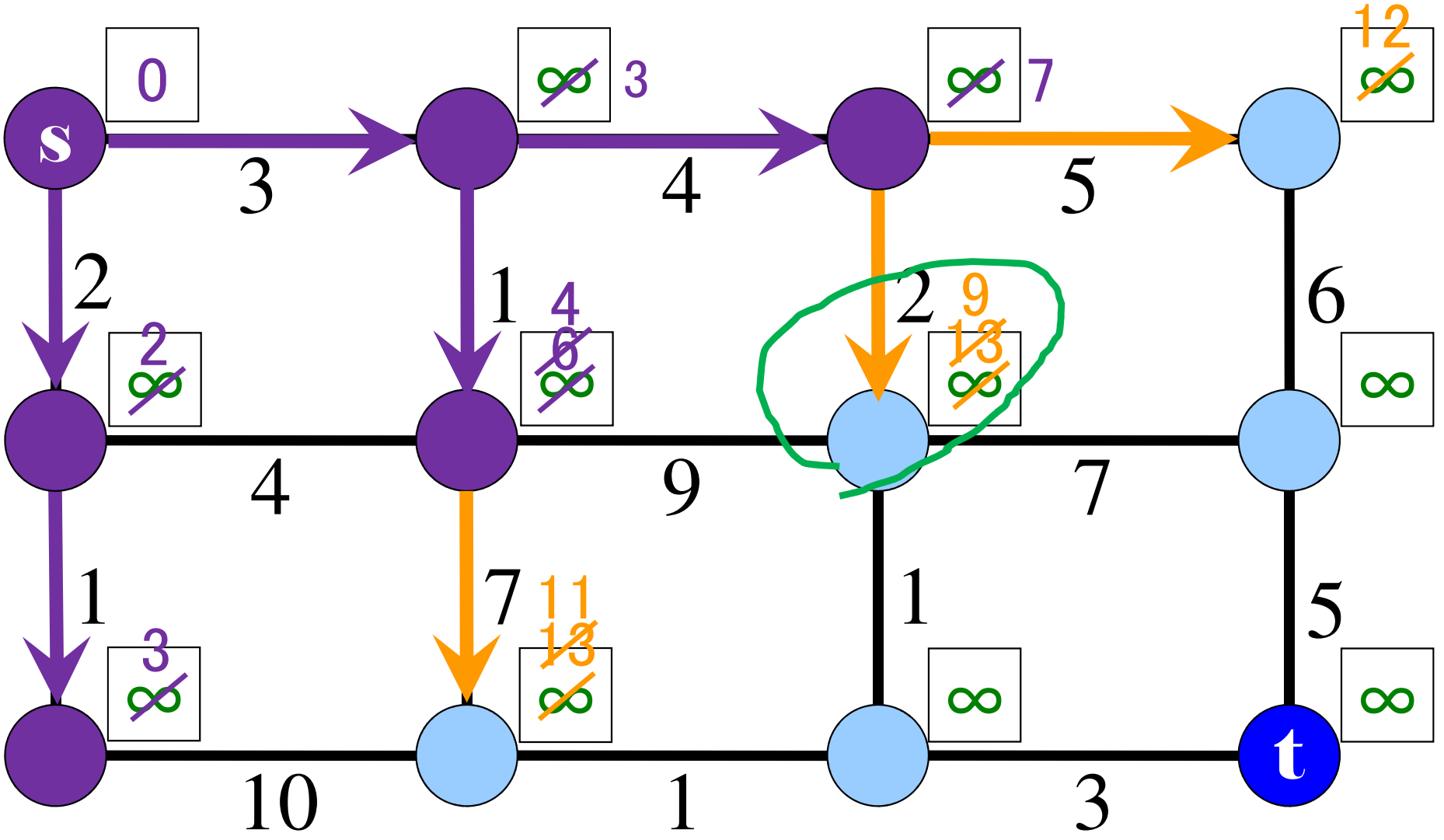
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



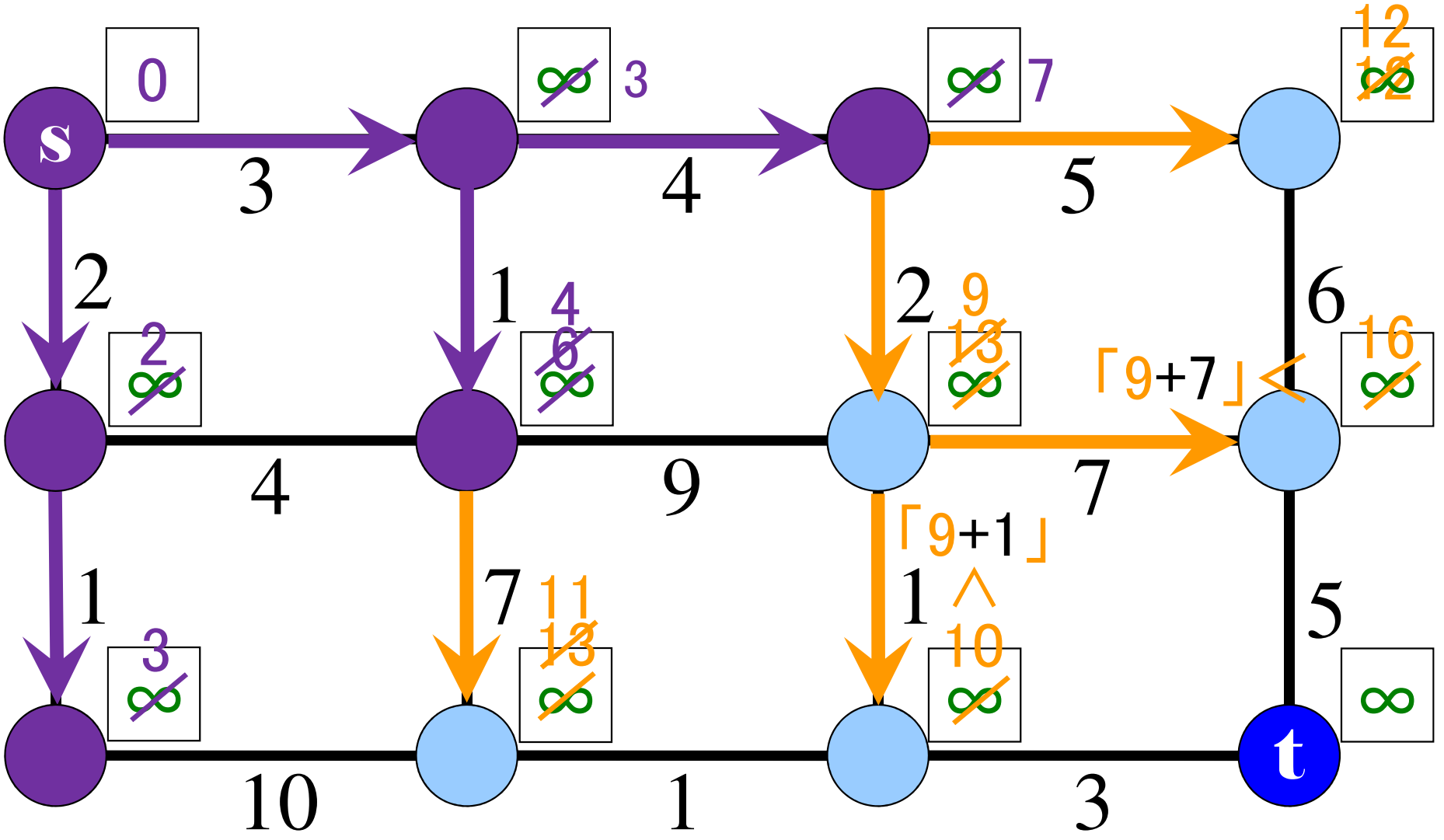
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

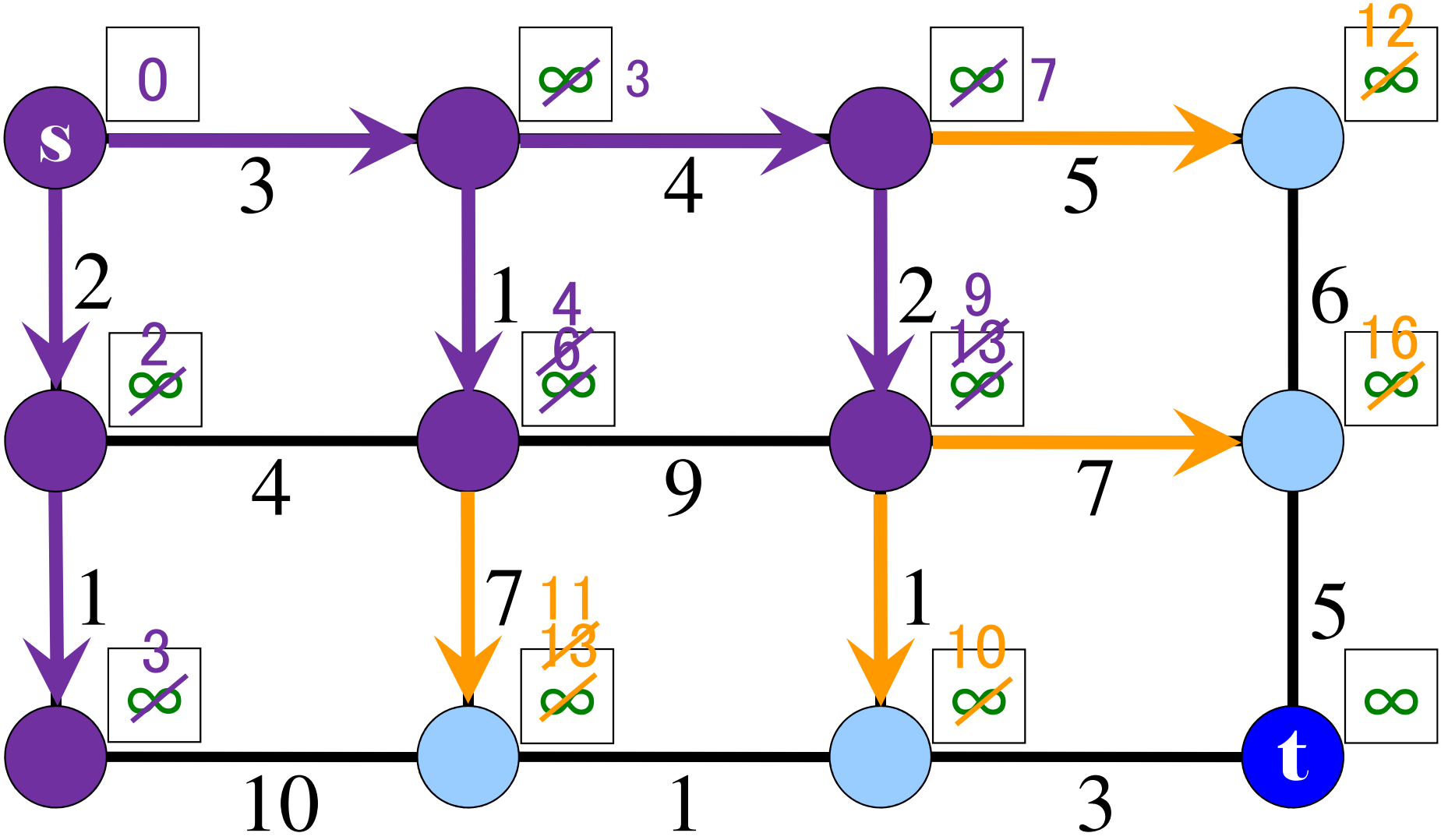


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

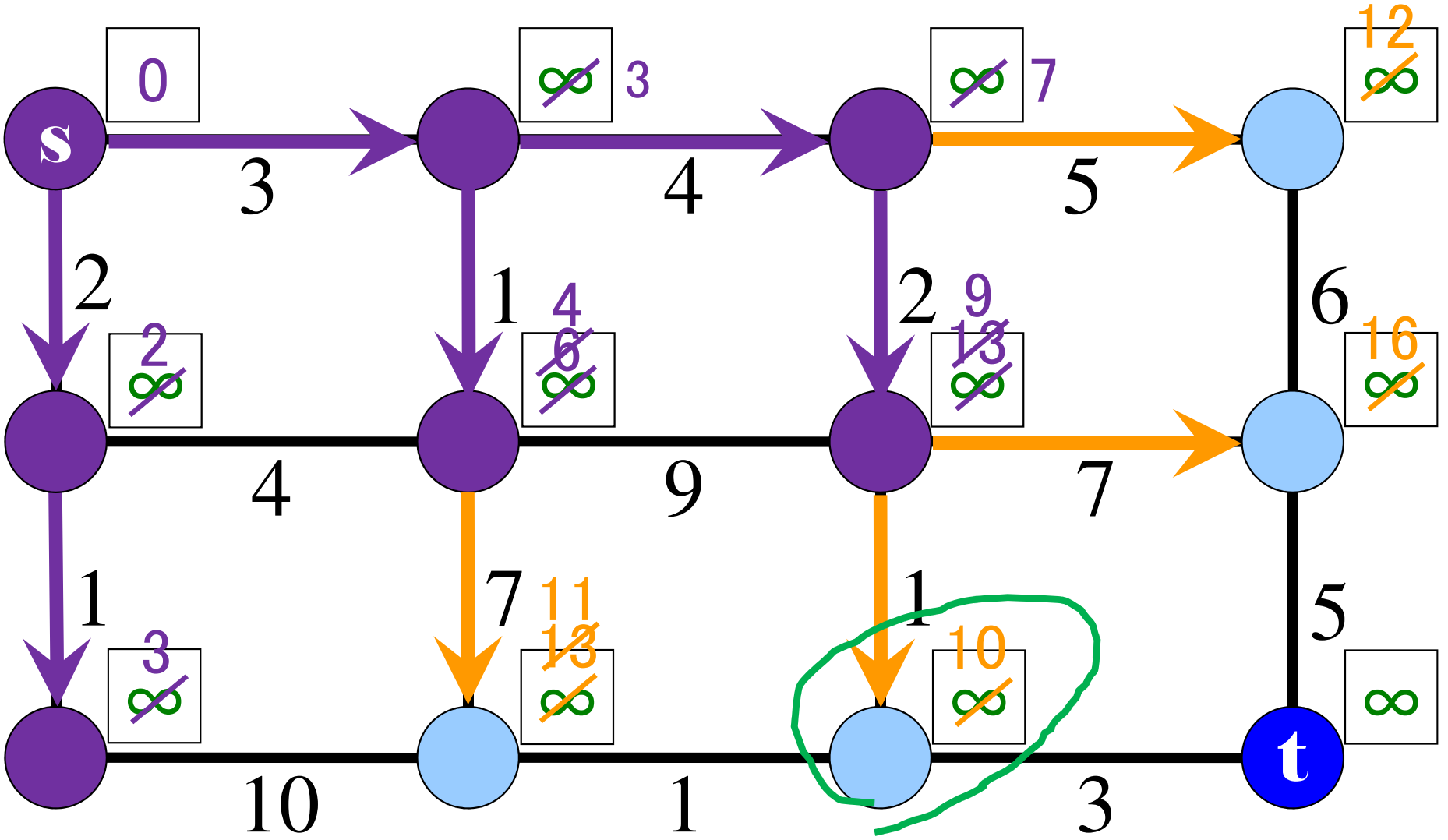
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



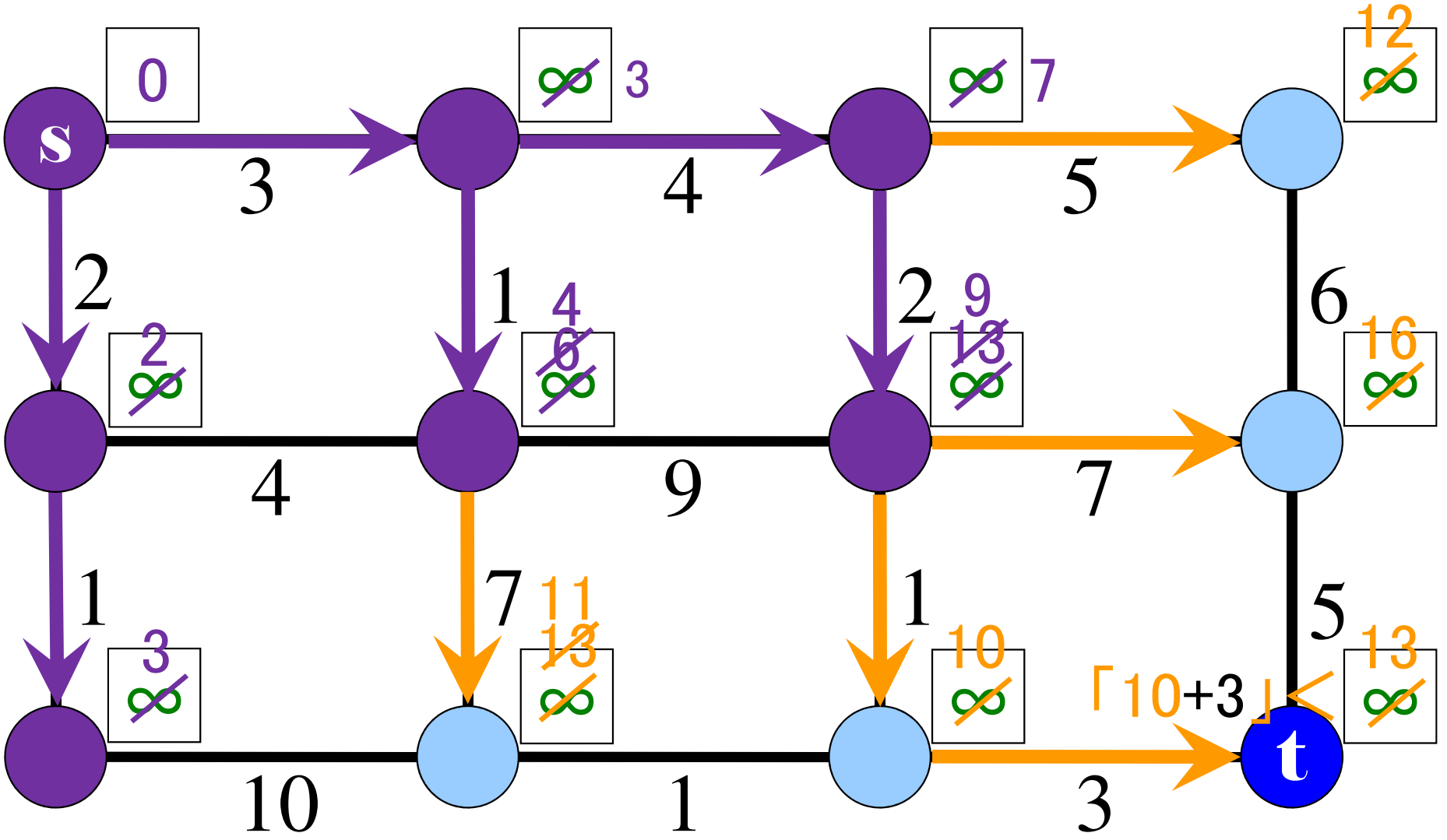
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

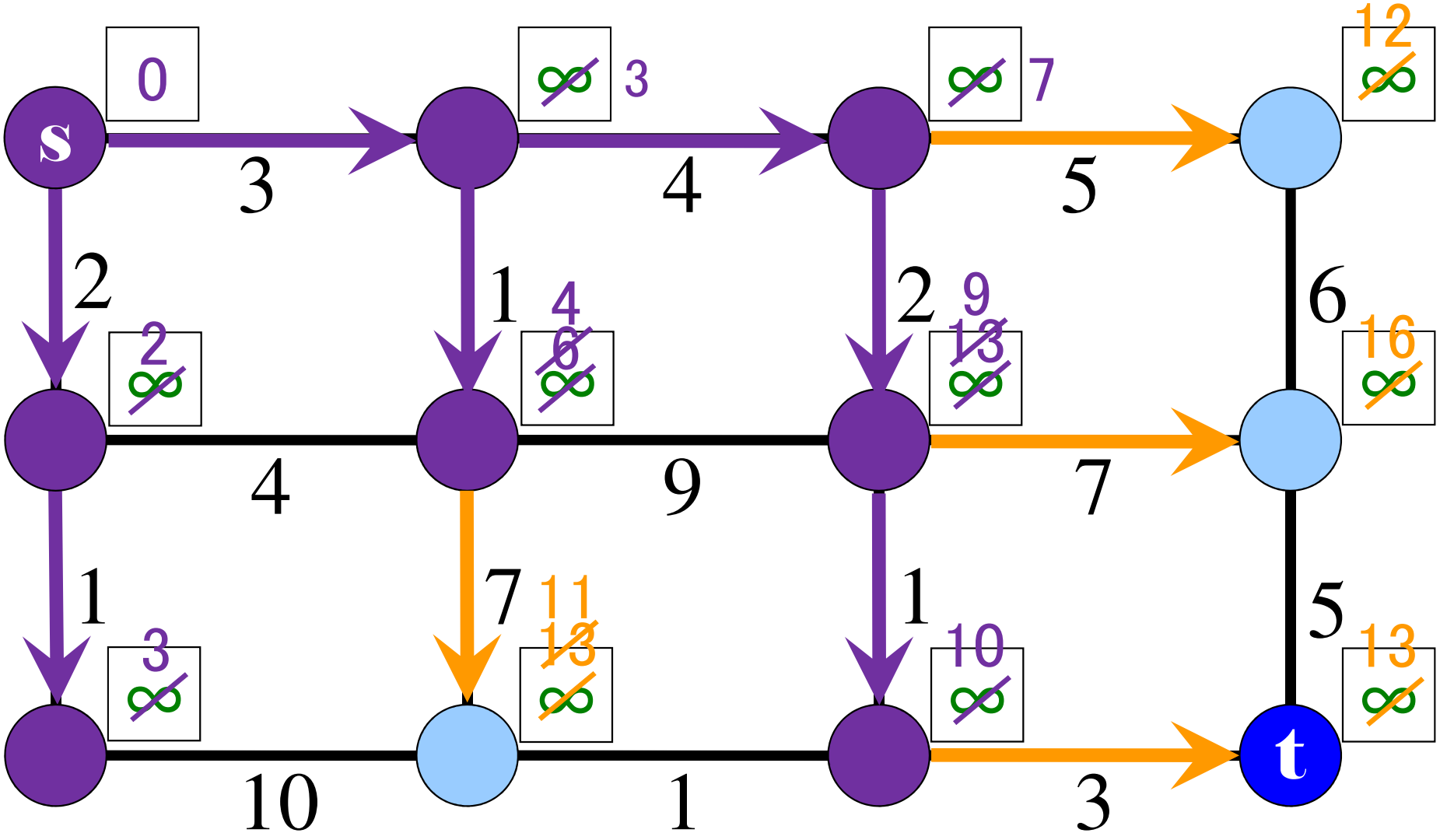


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

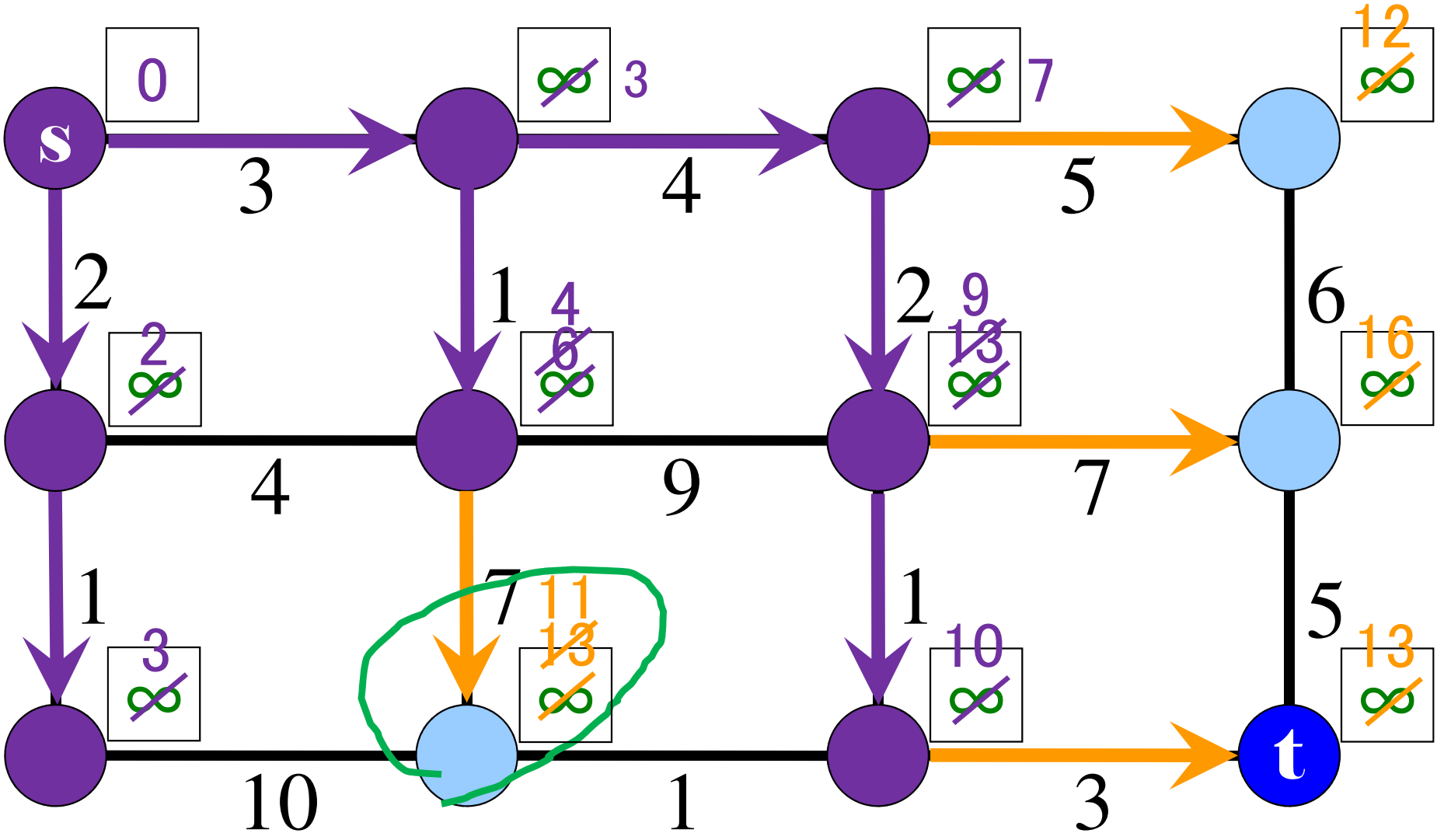
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



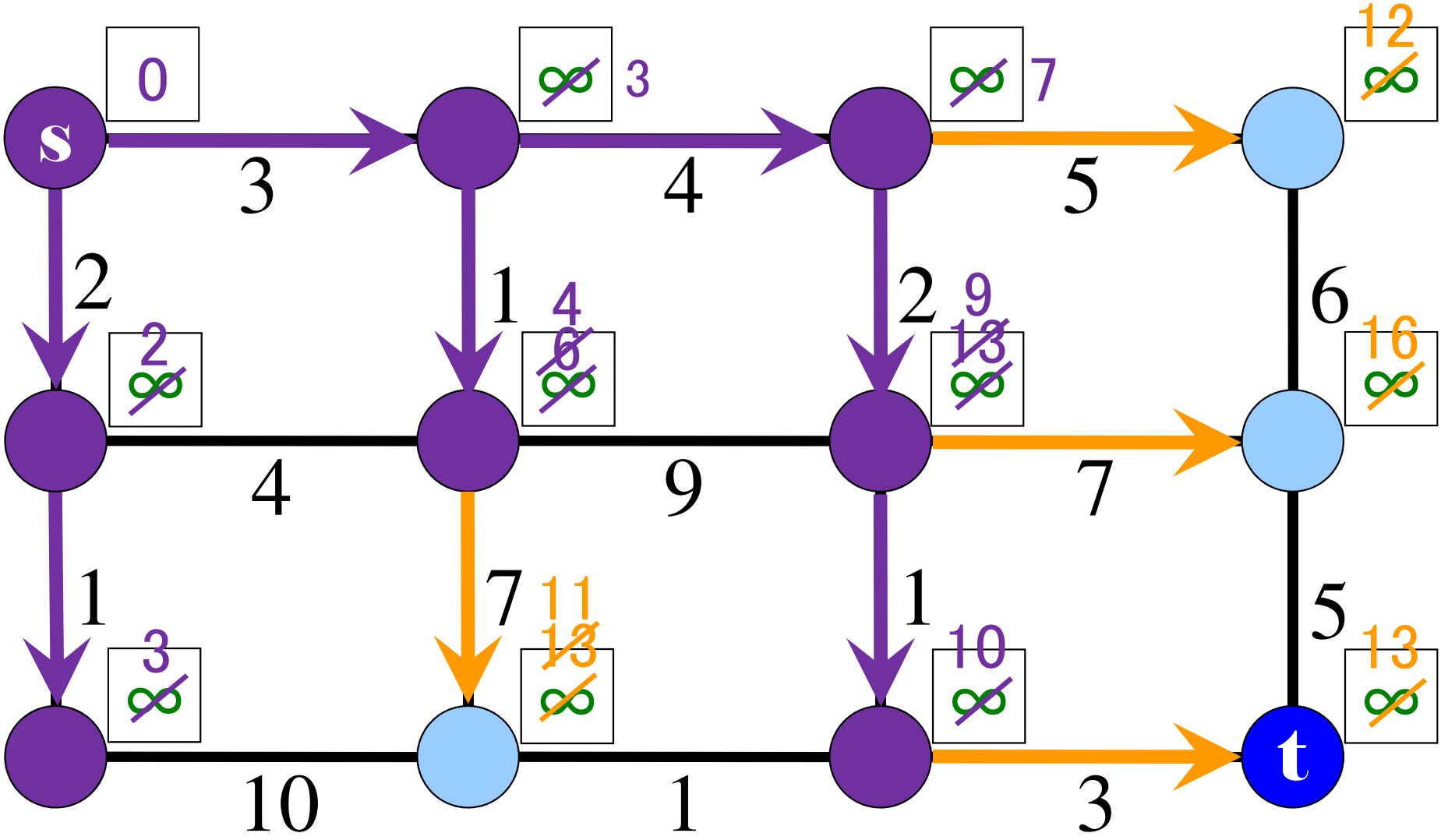
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

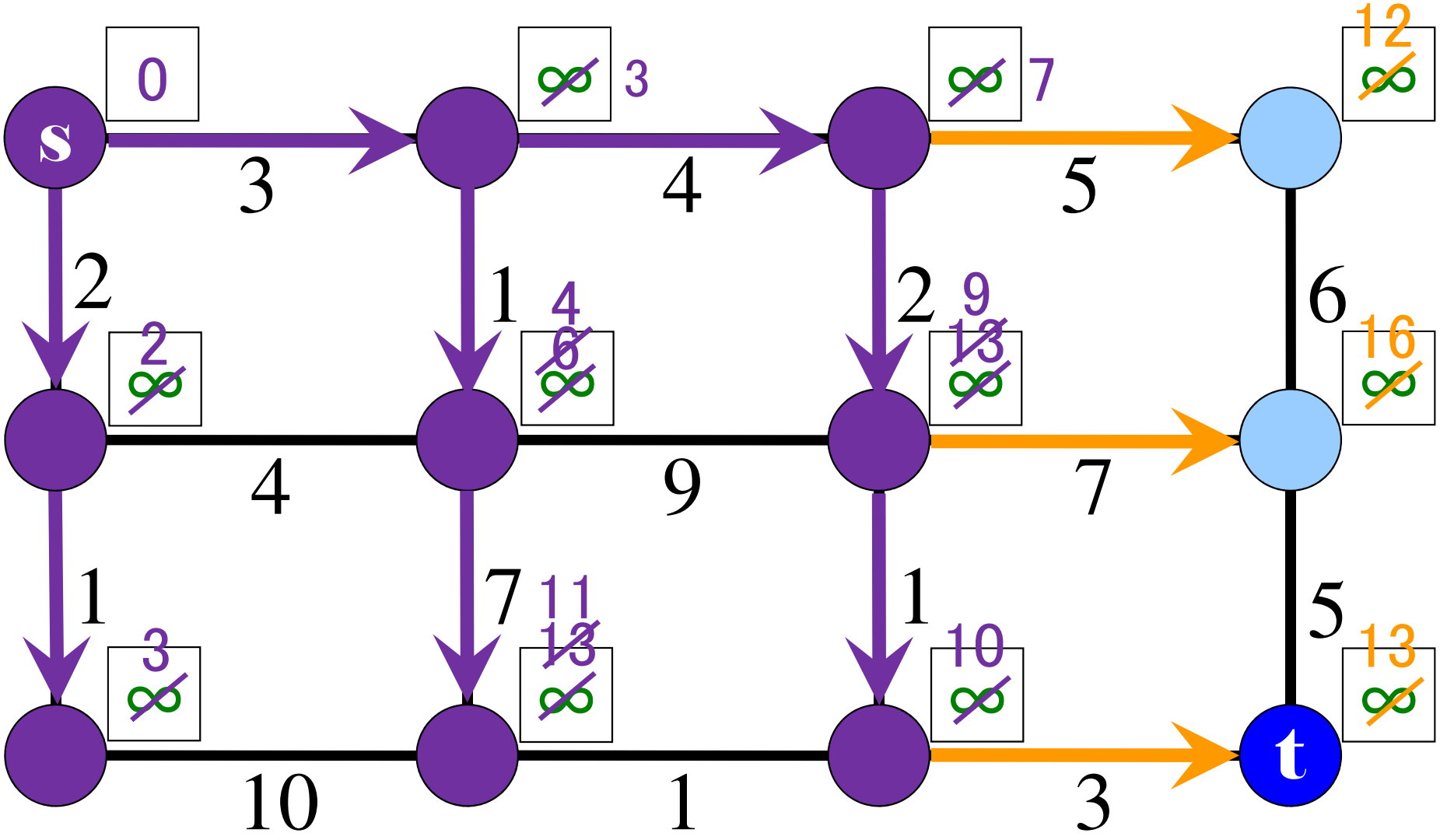


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

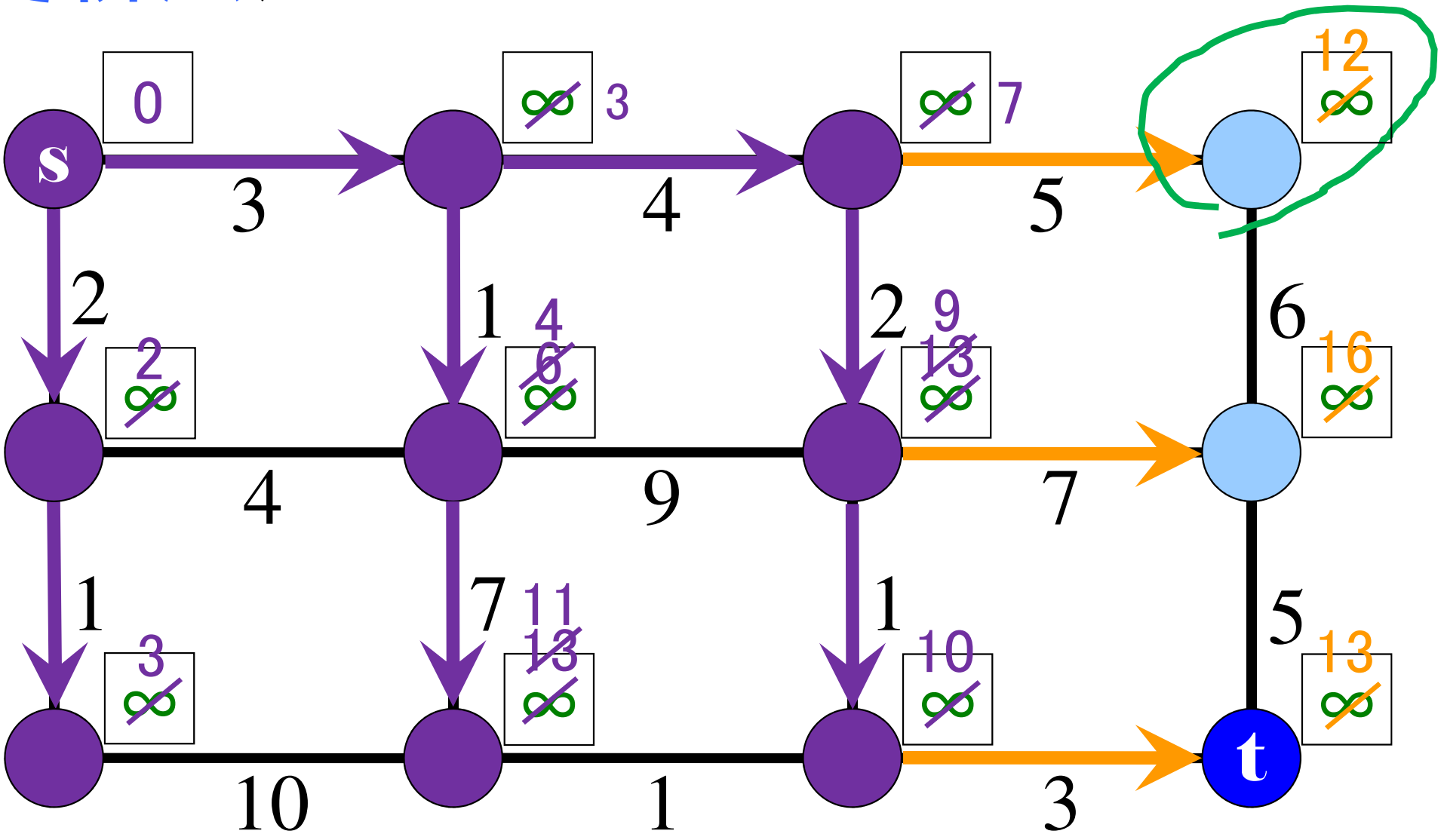
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



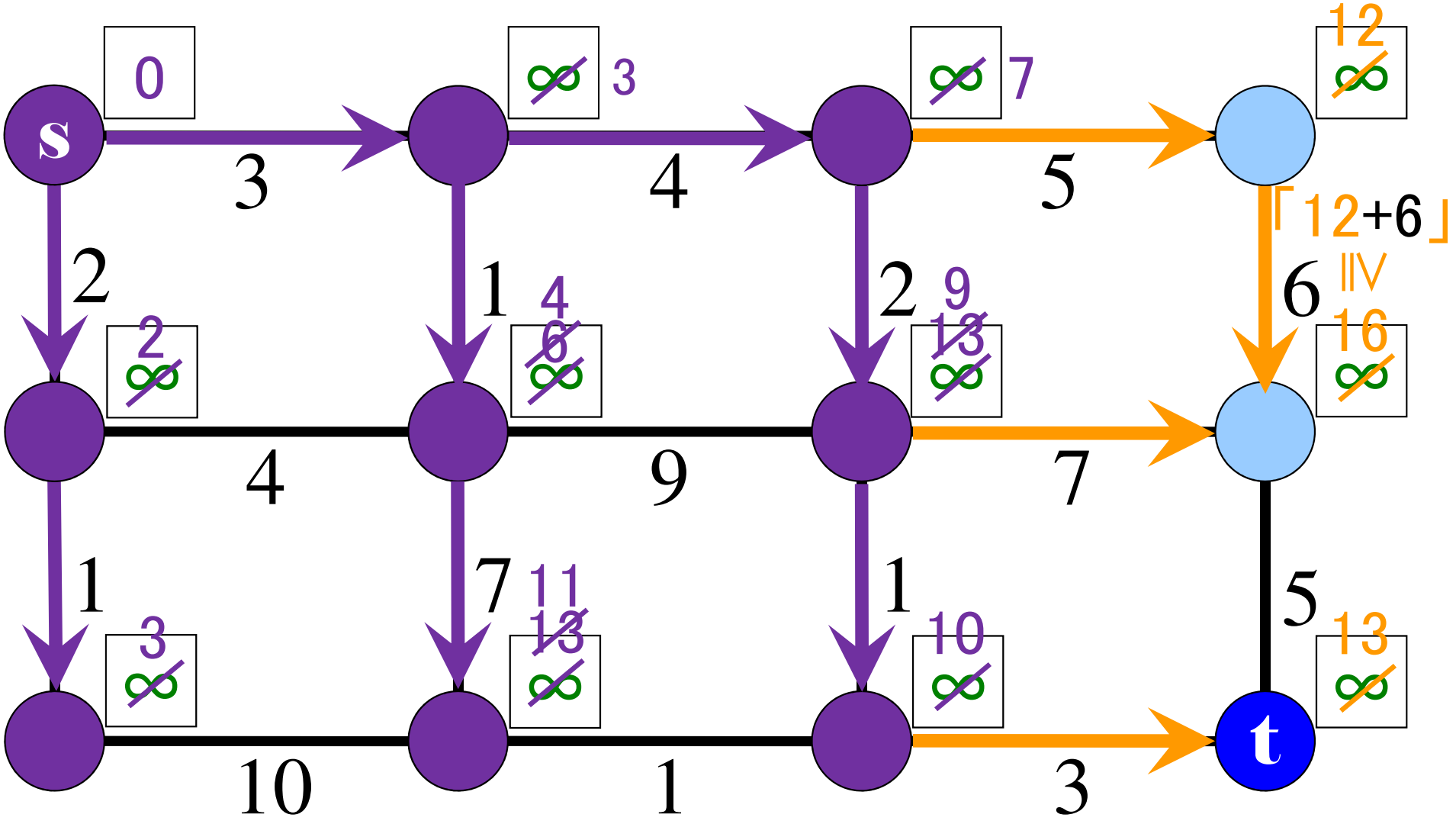
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

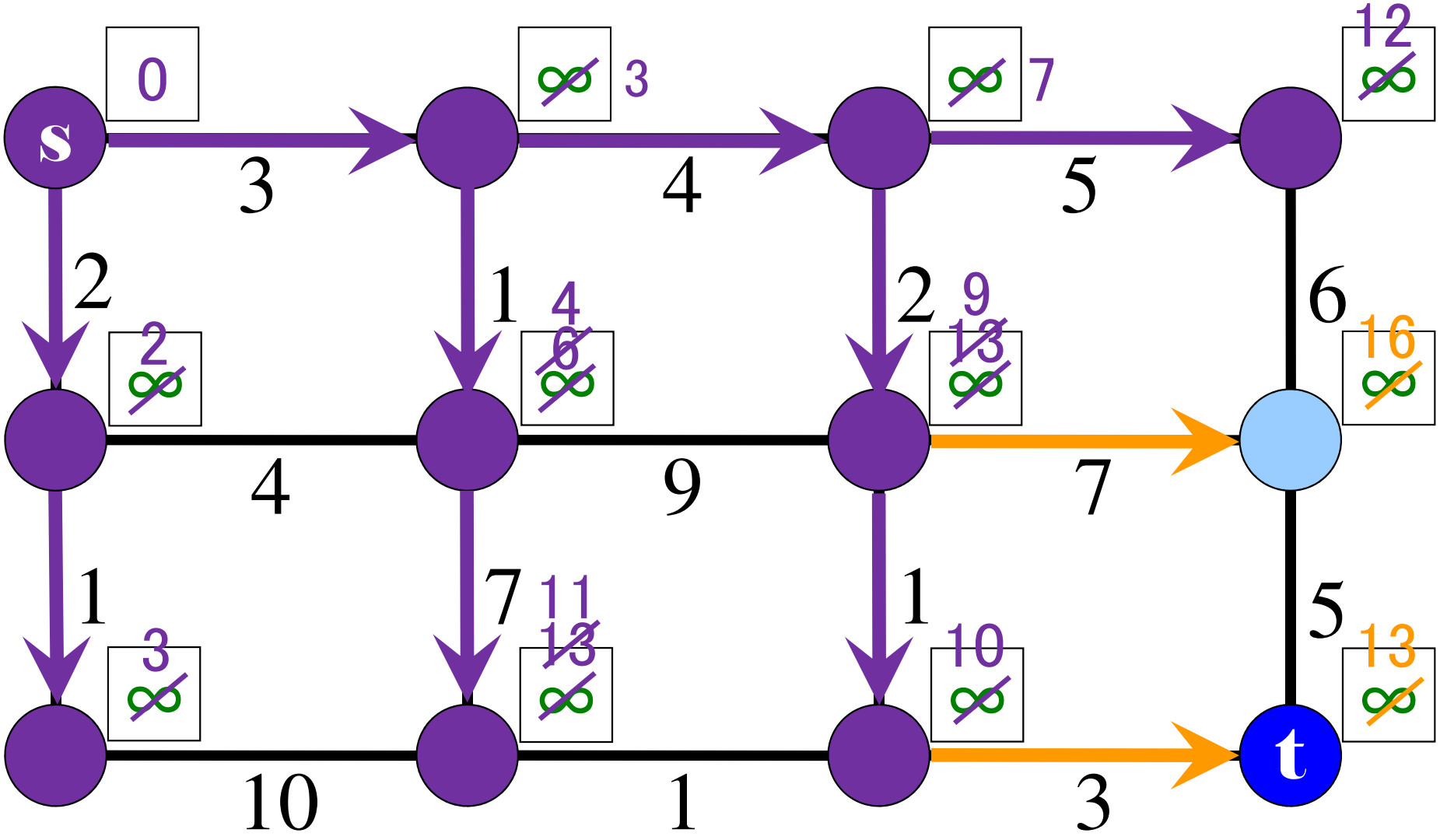


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

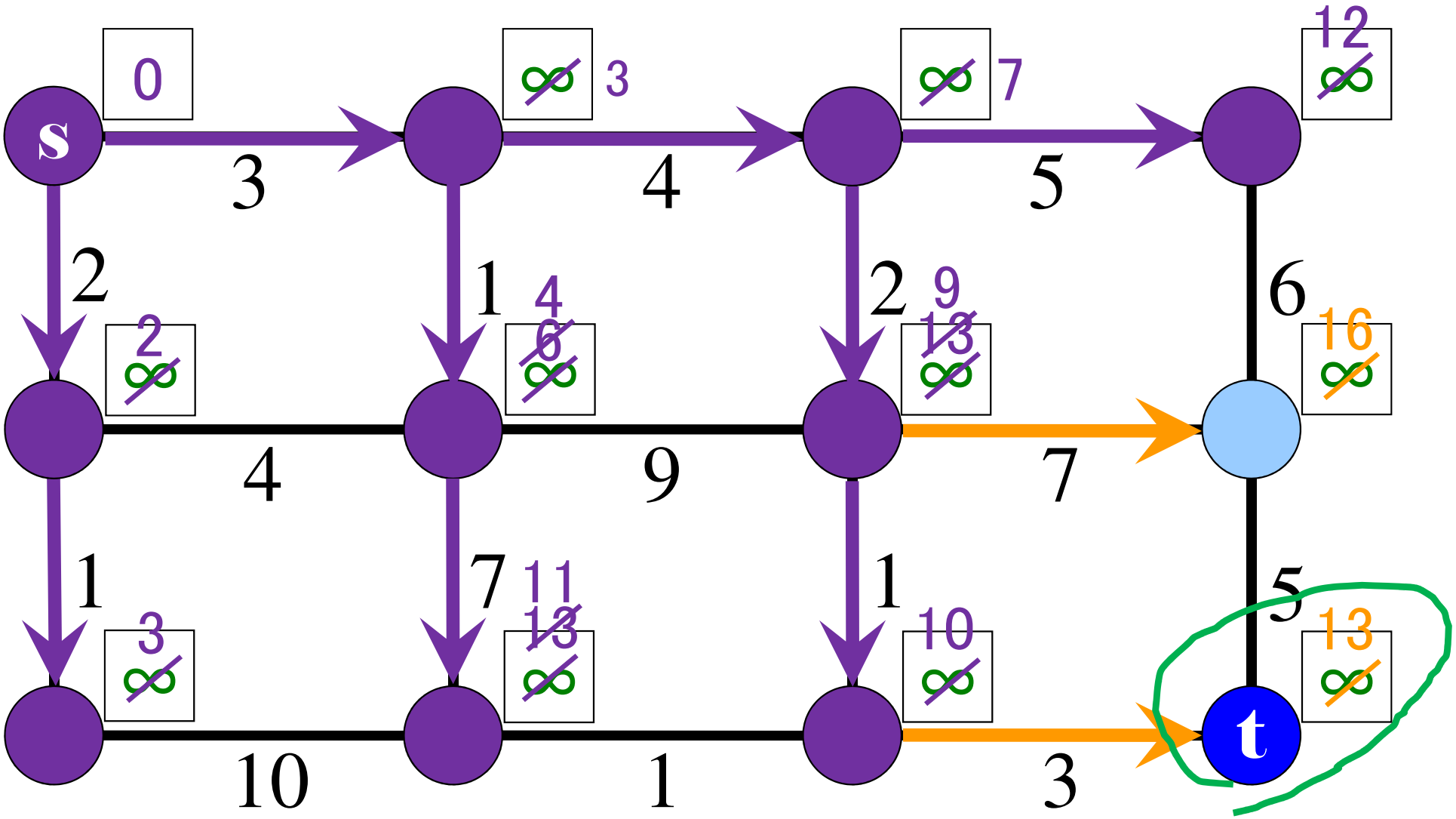
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



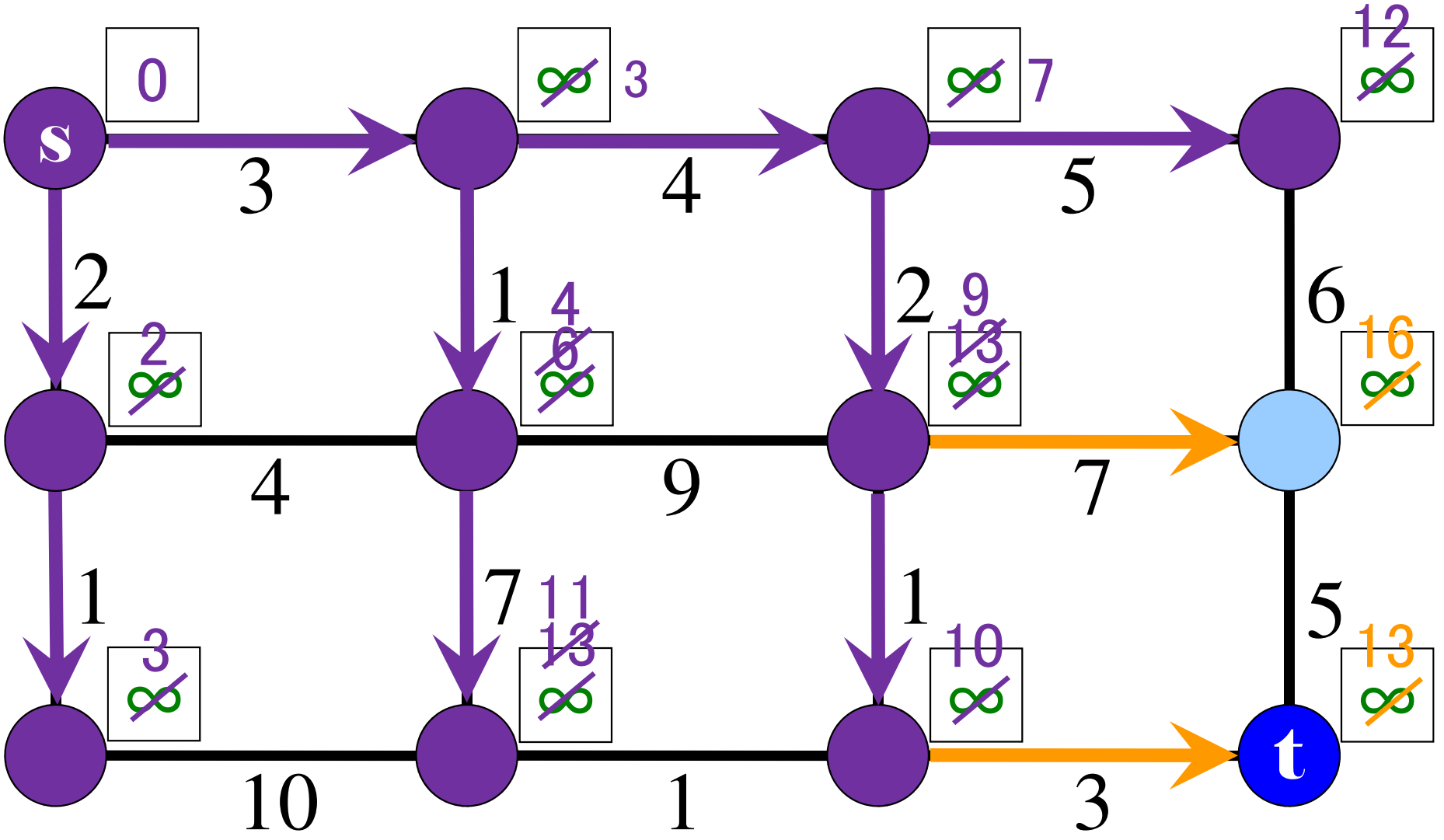
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

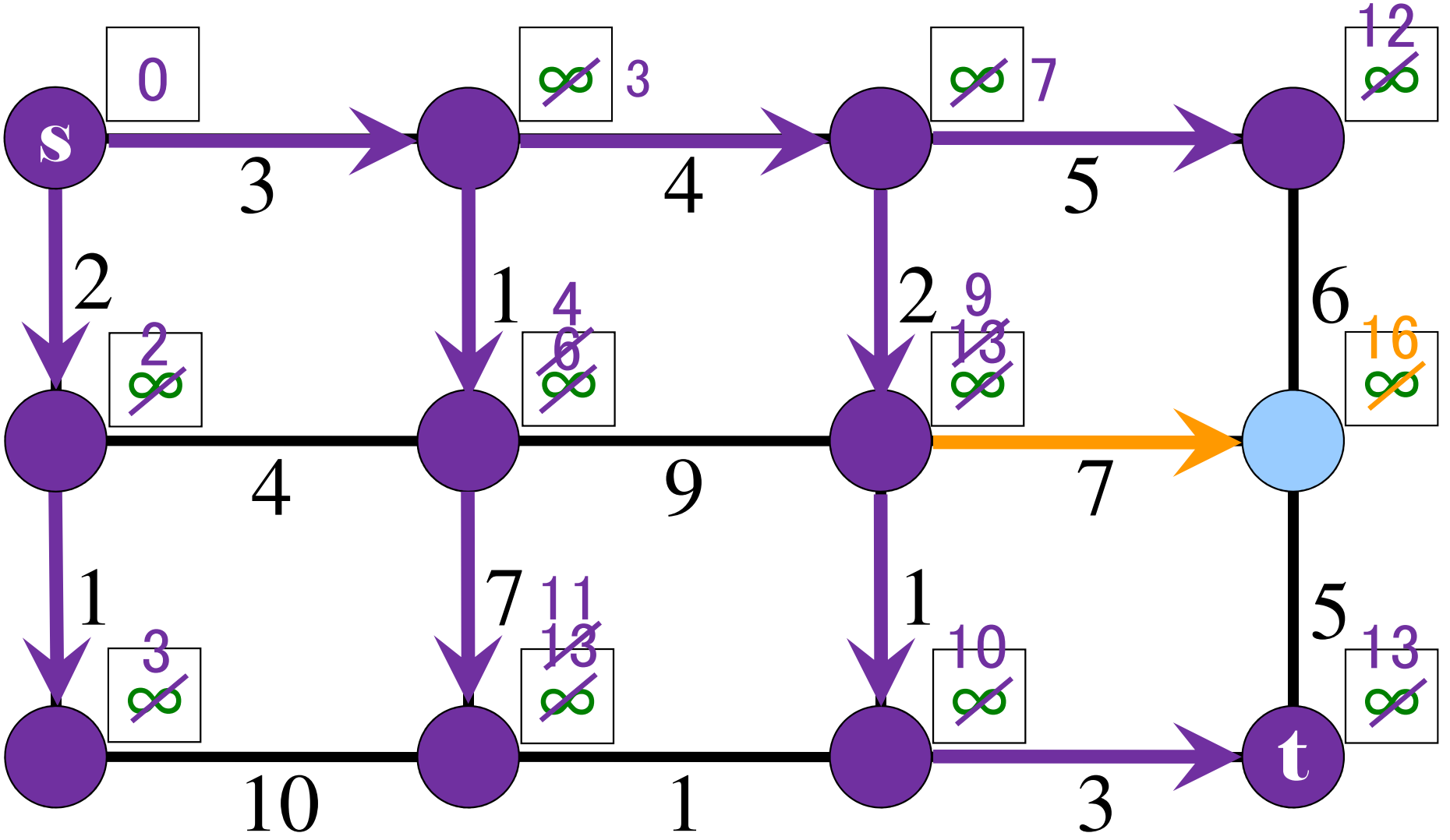


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

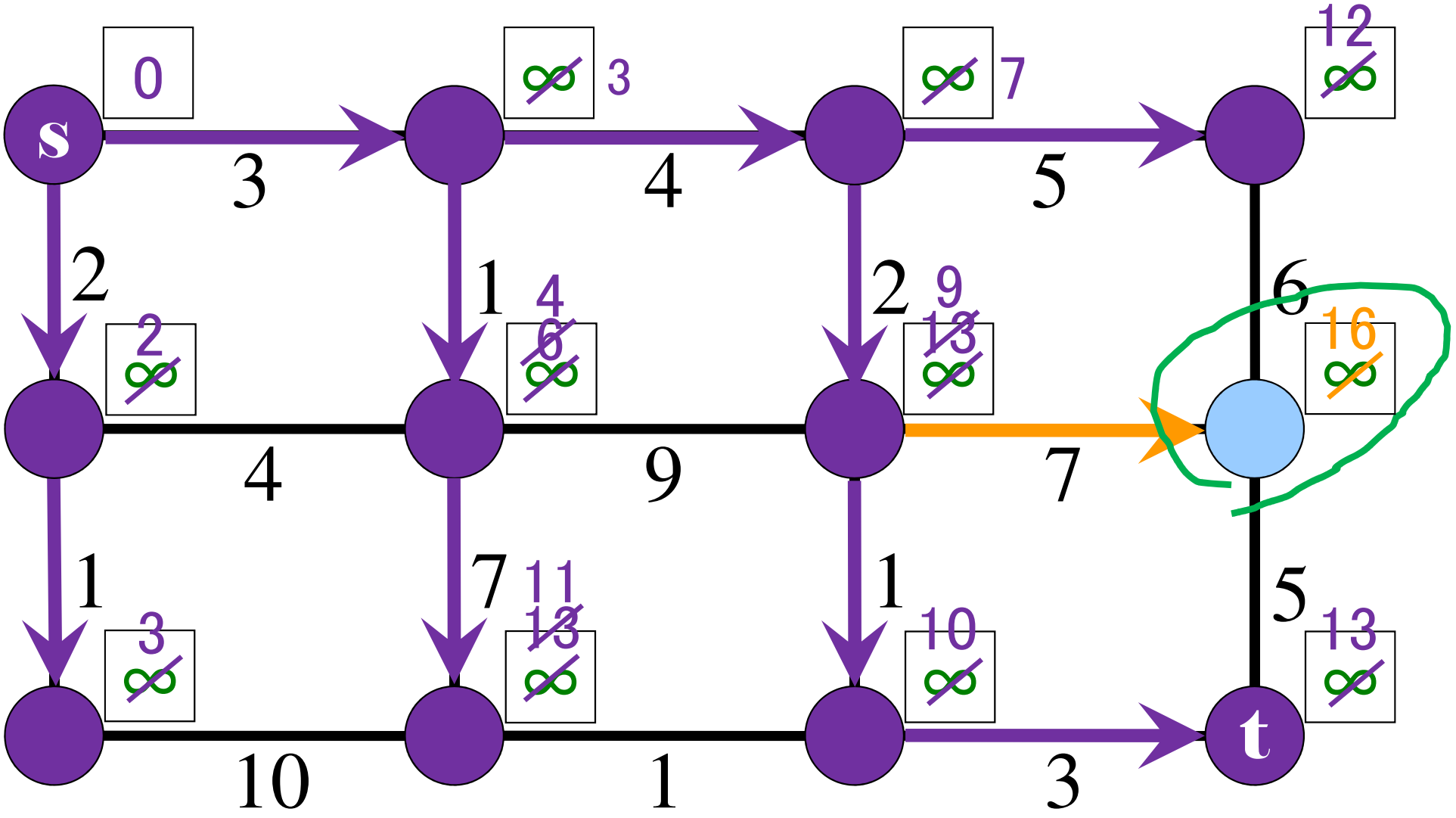
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



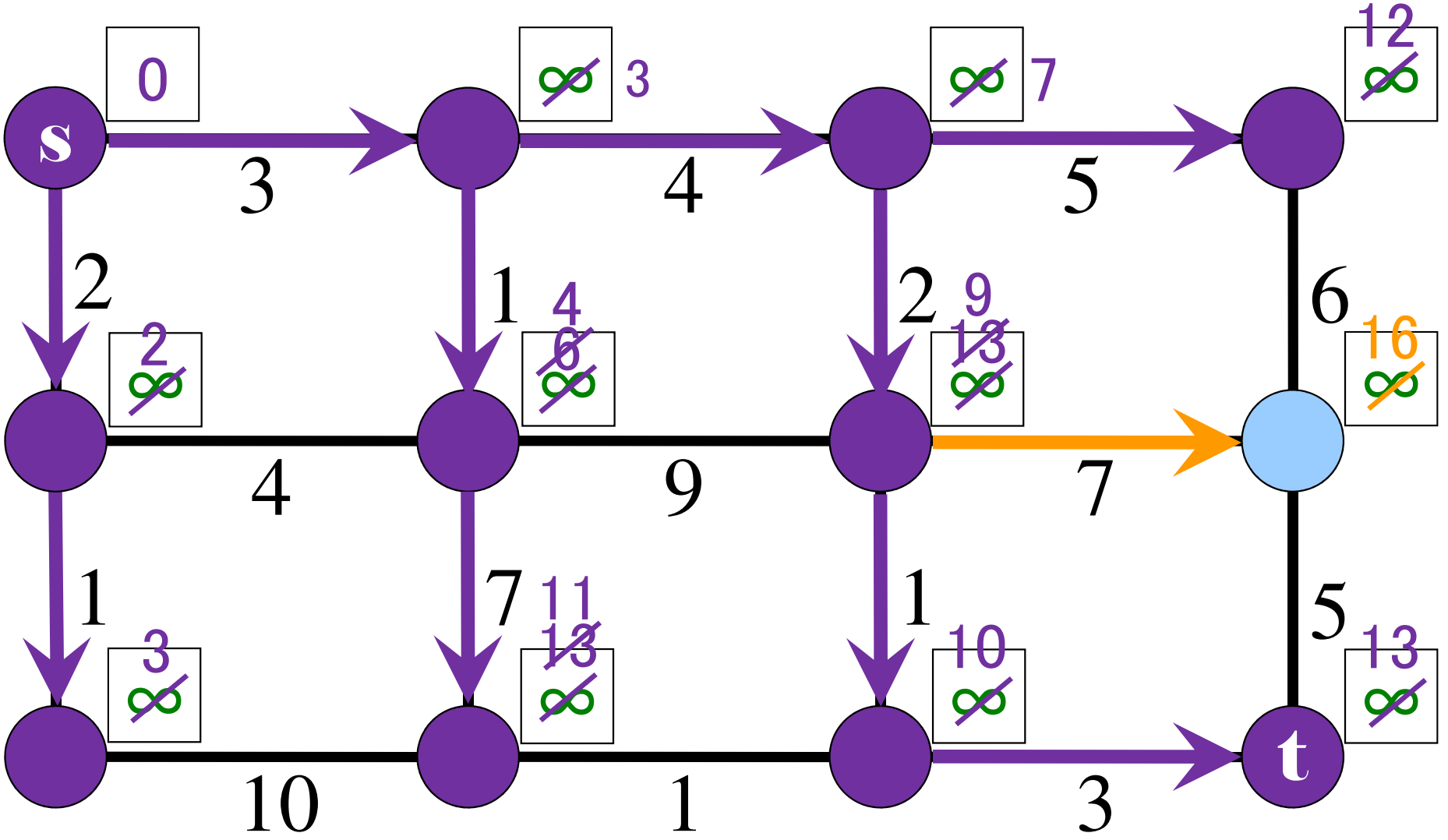
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

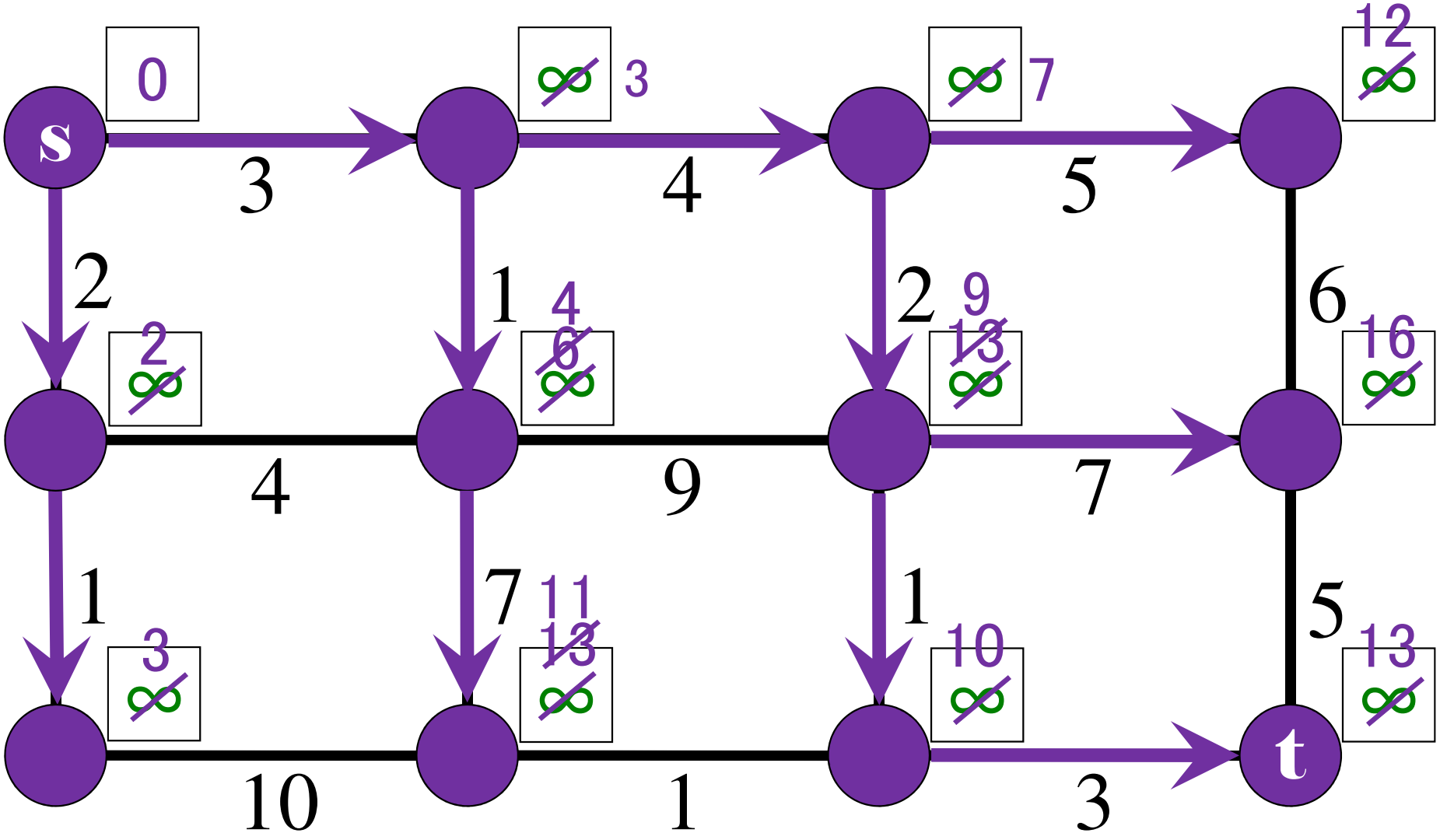


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

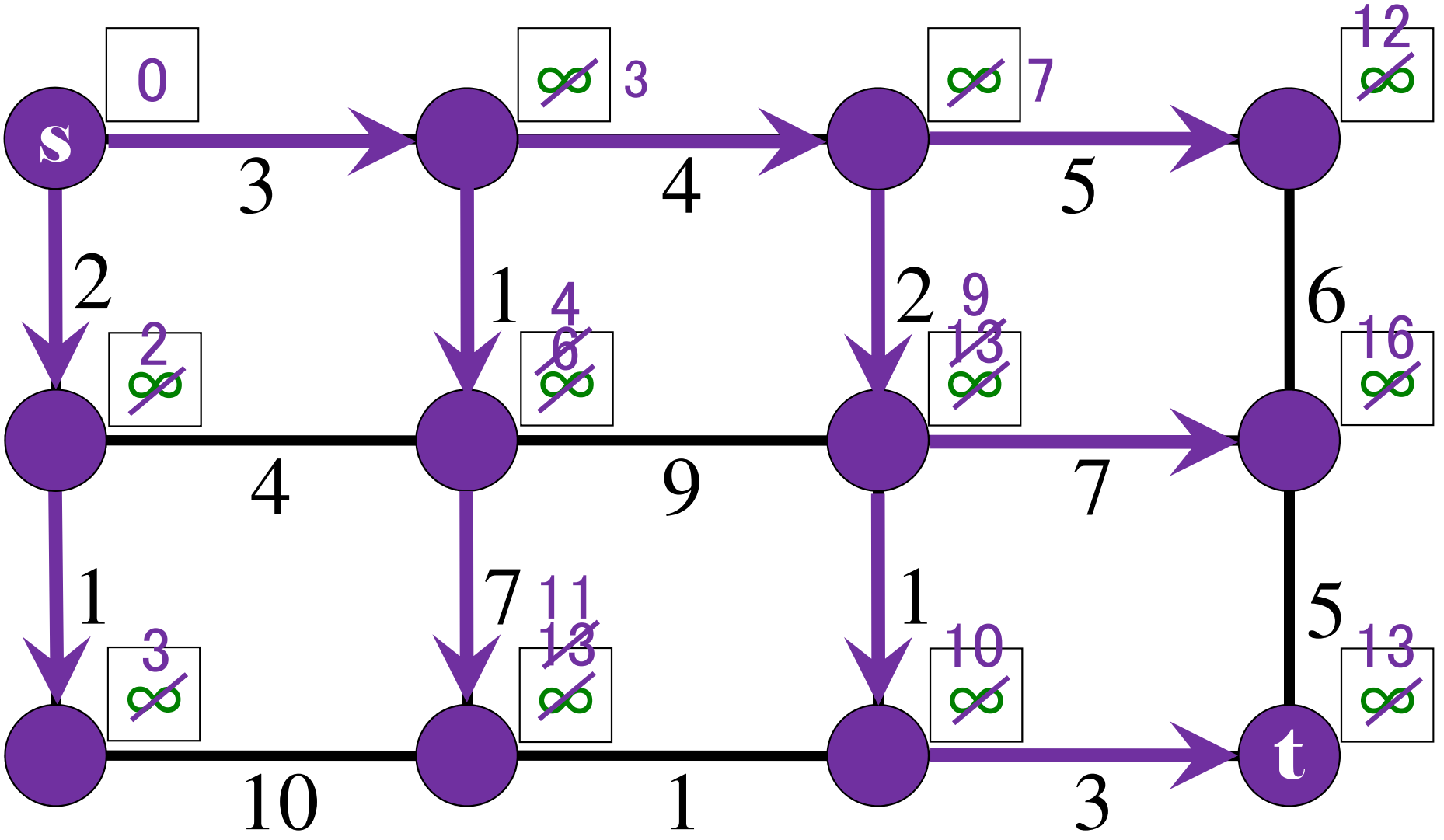
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



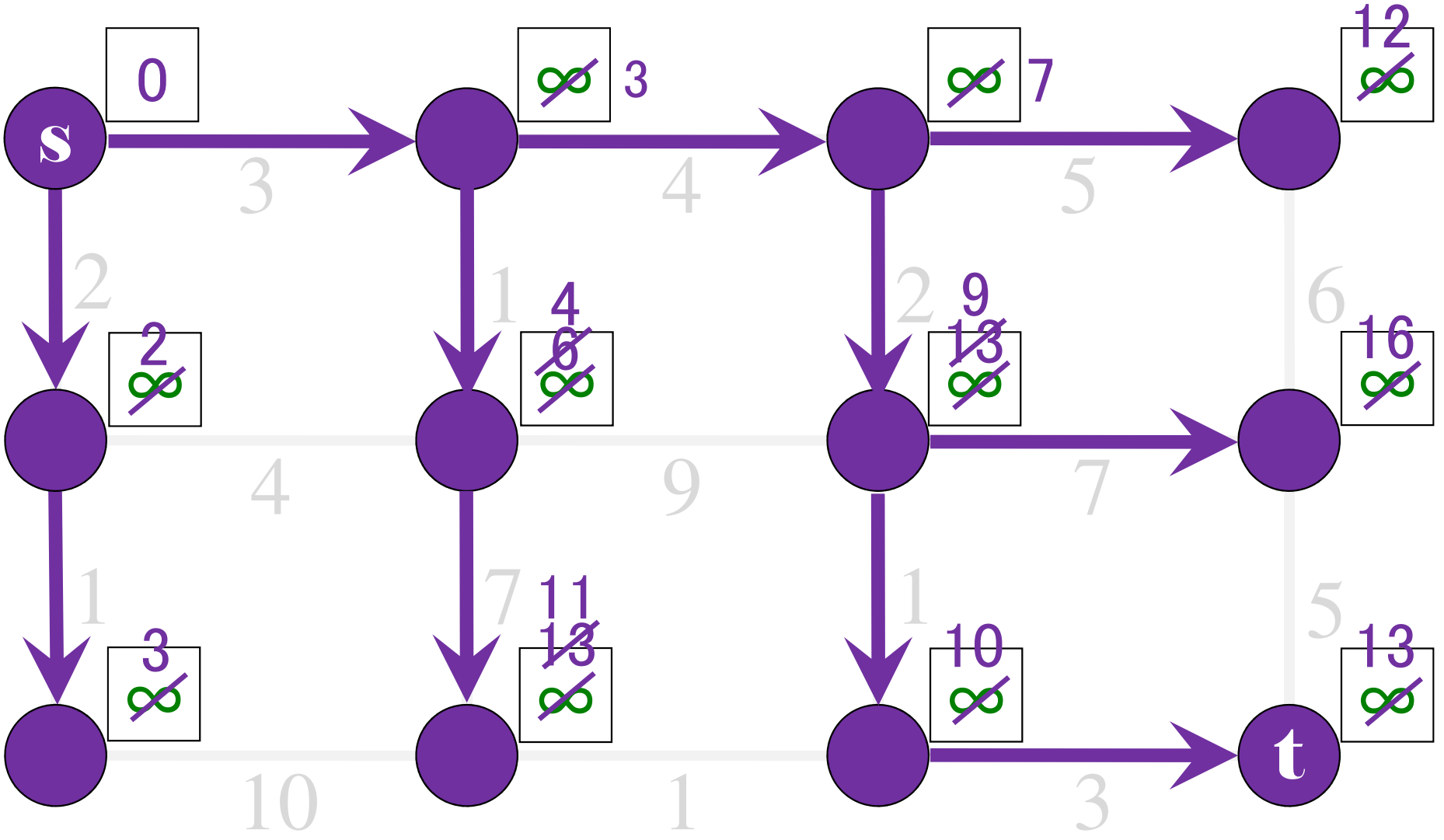
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



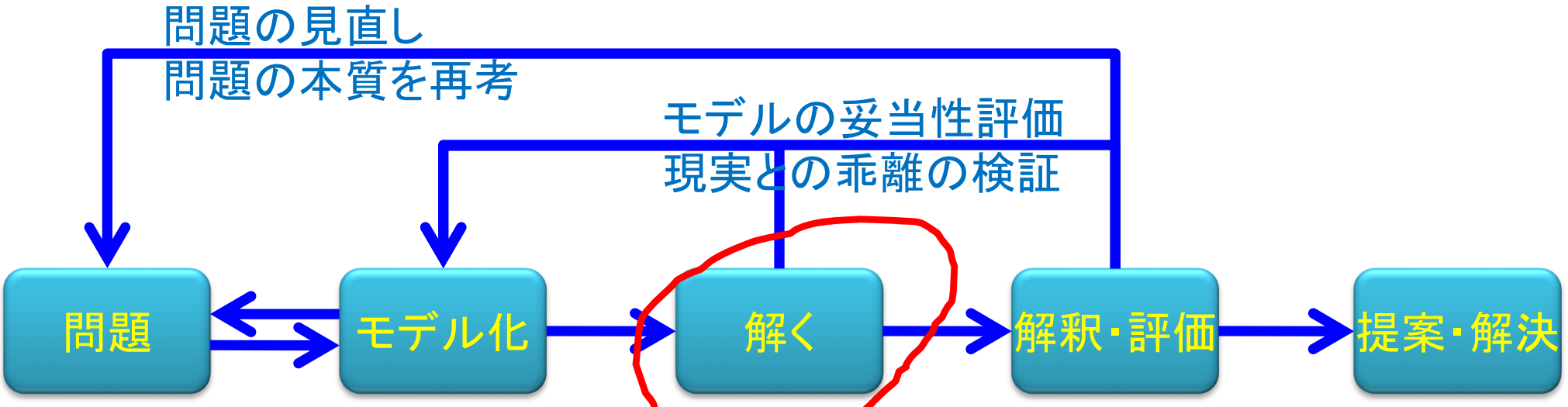
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

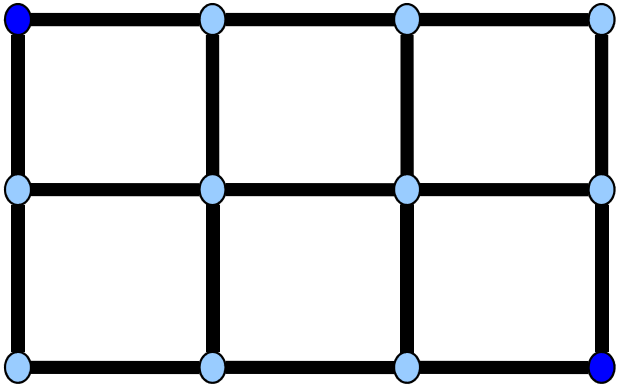
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

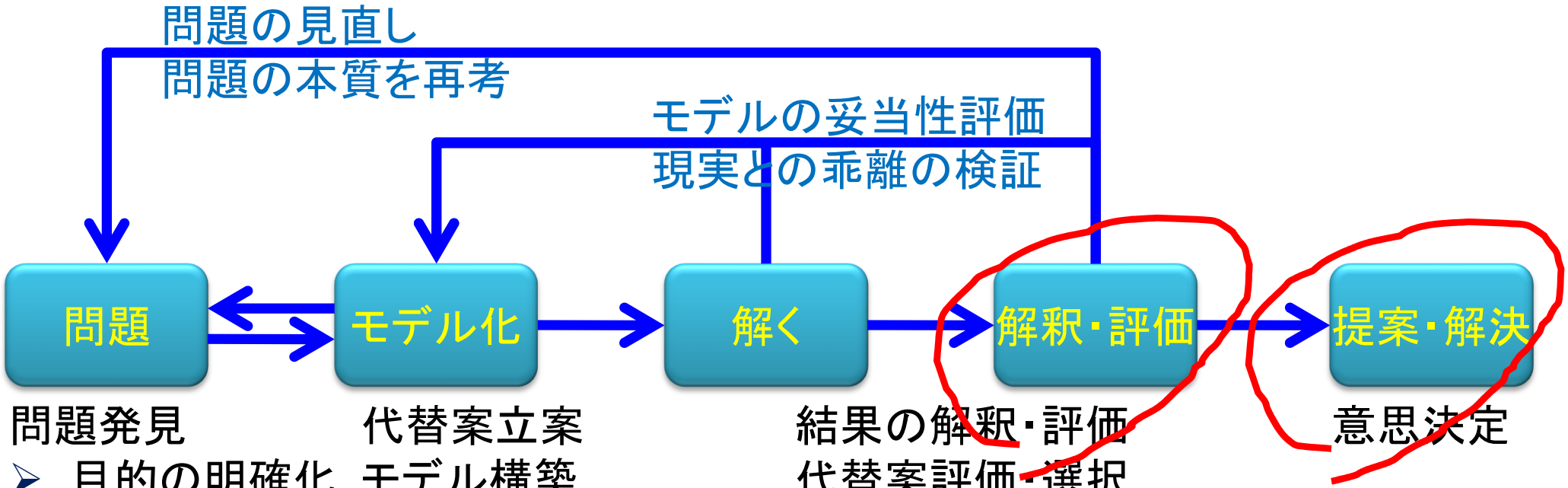
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

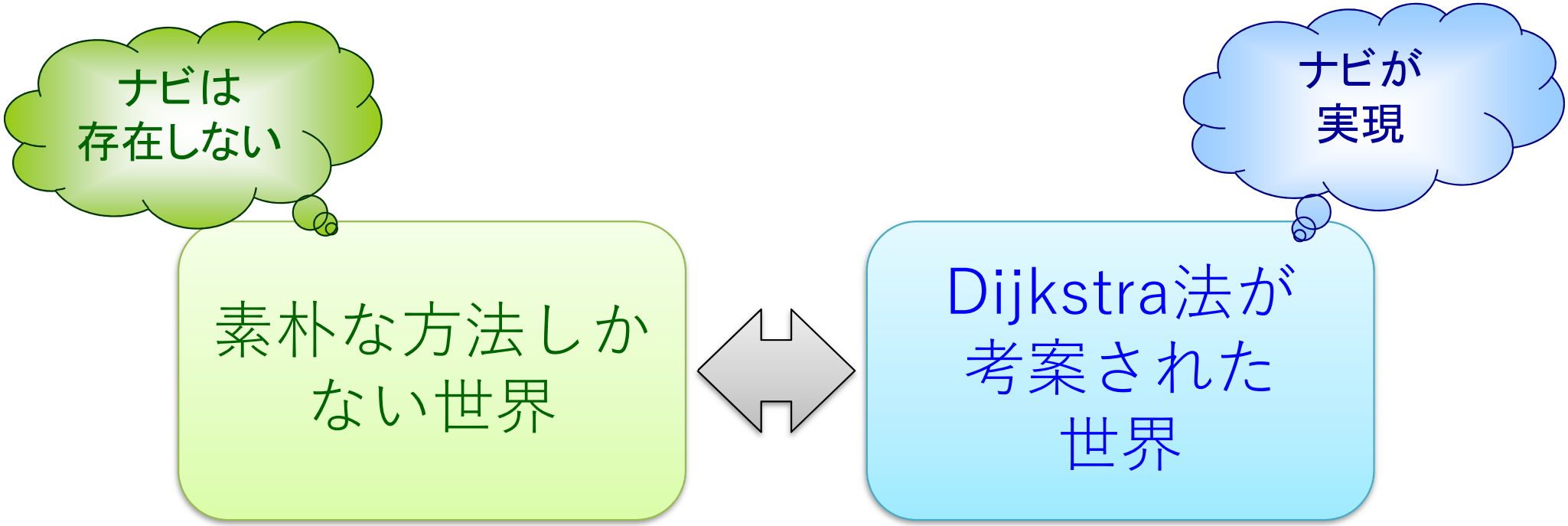
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

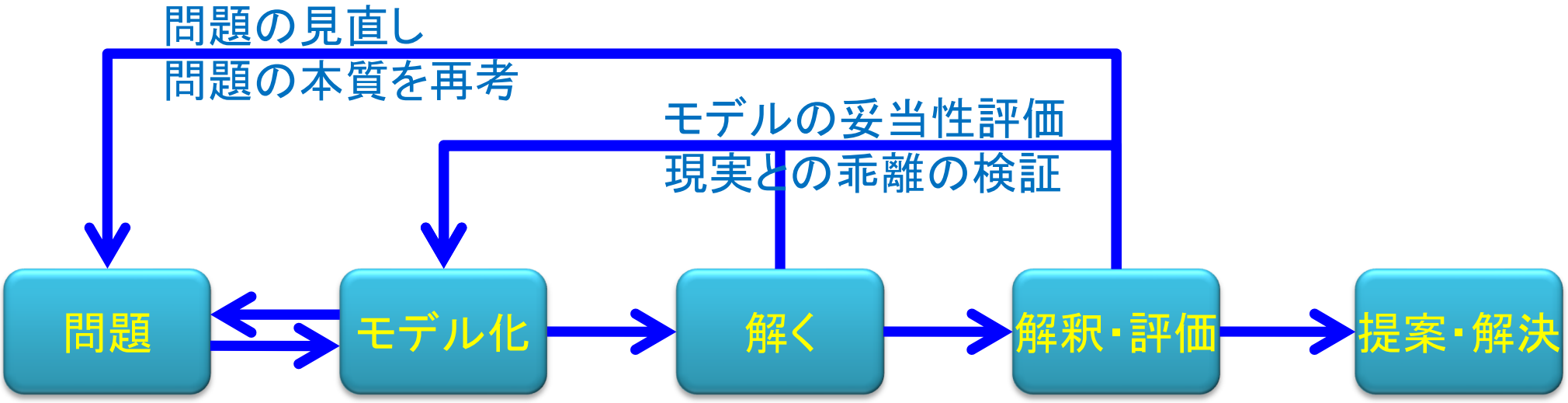
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



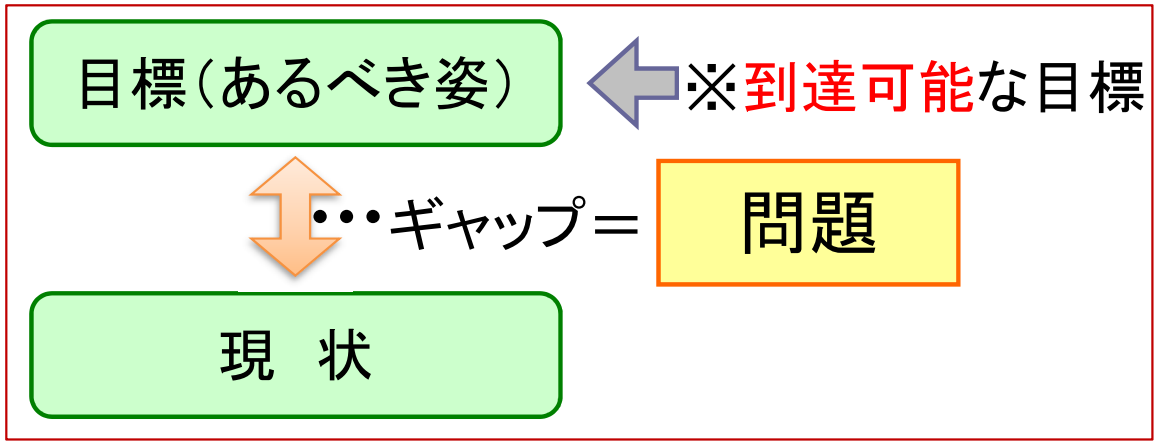
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

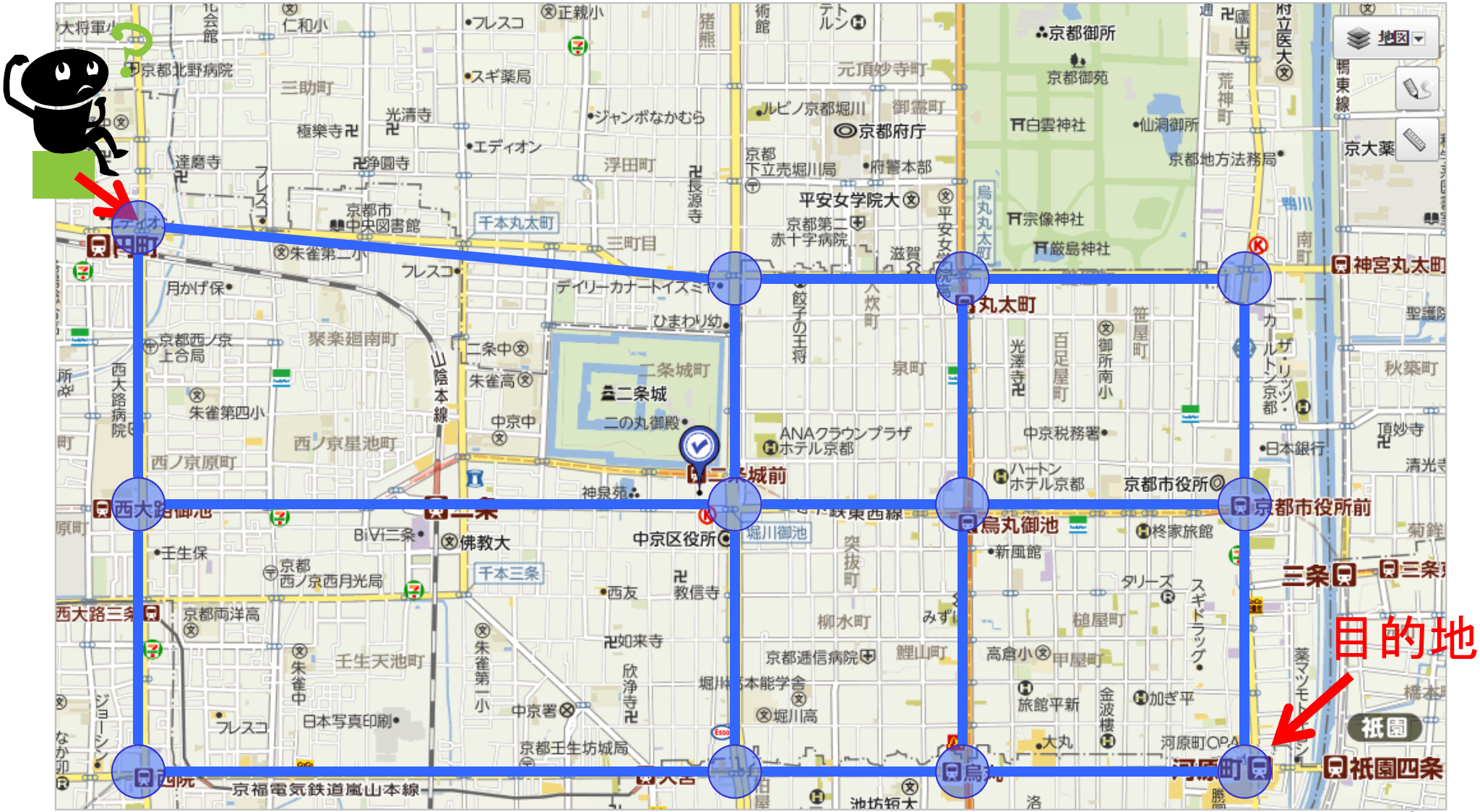
意思決定

問題の定義



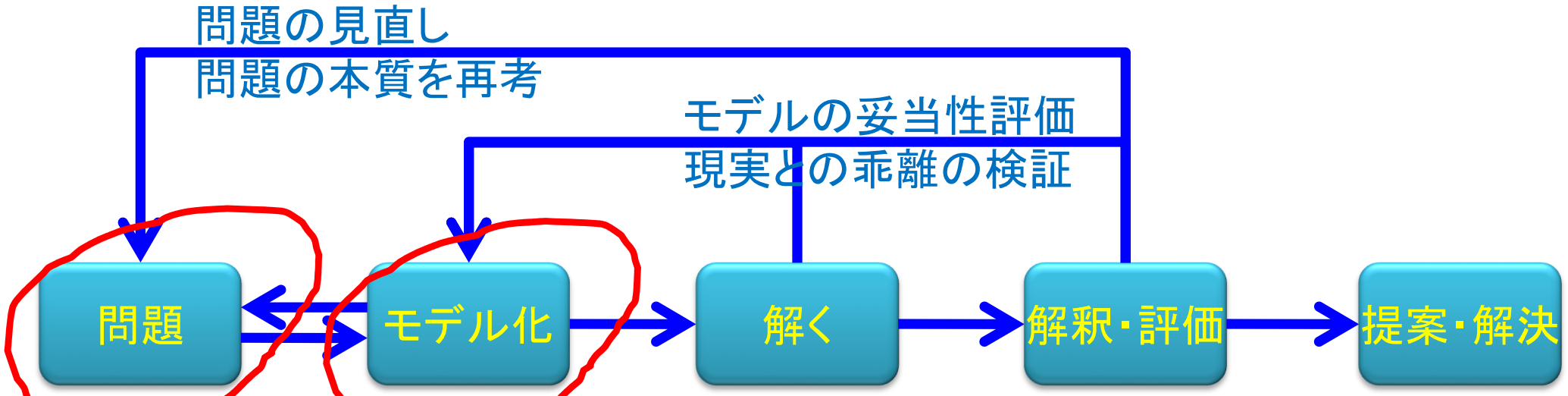
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

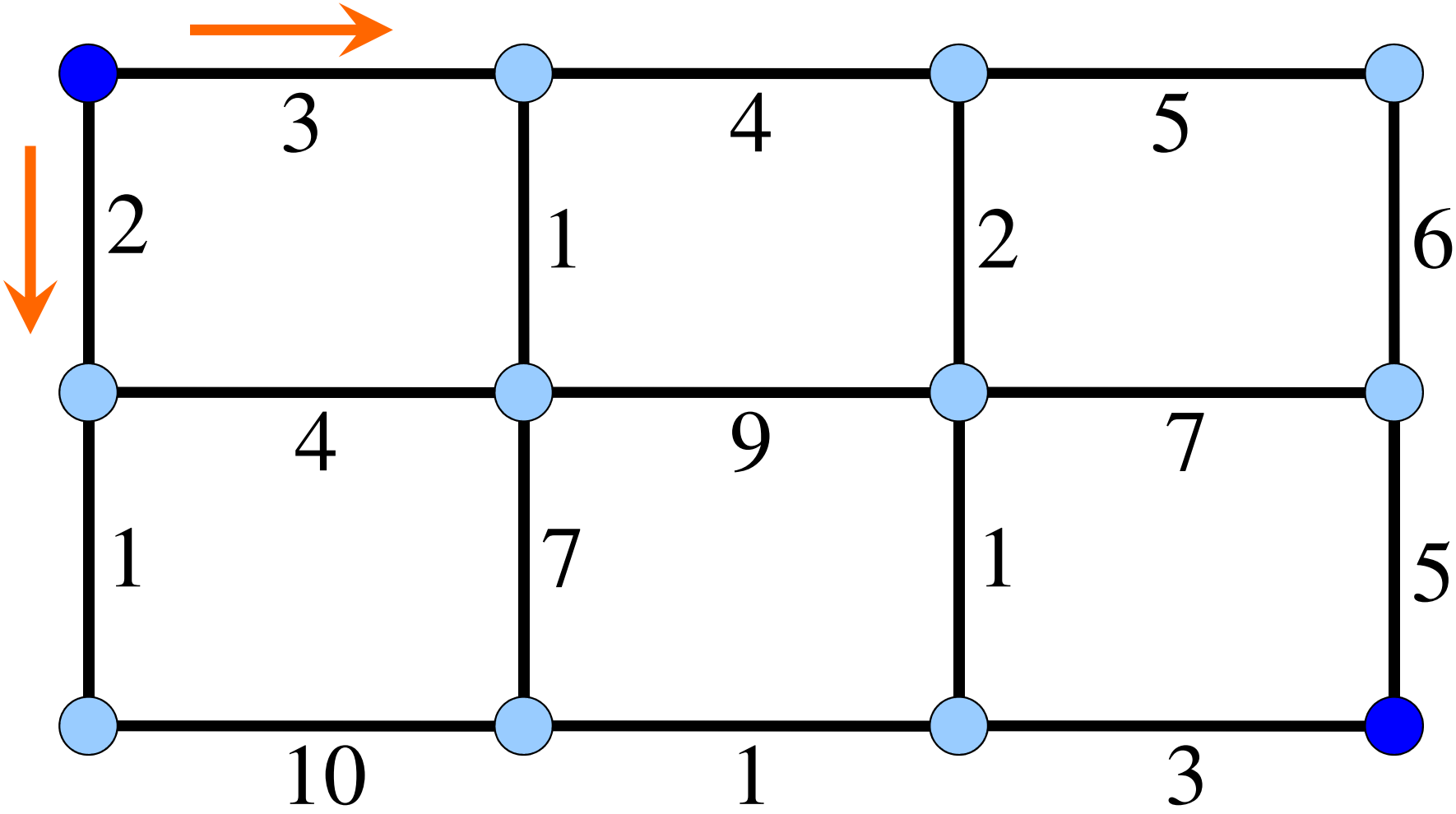
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

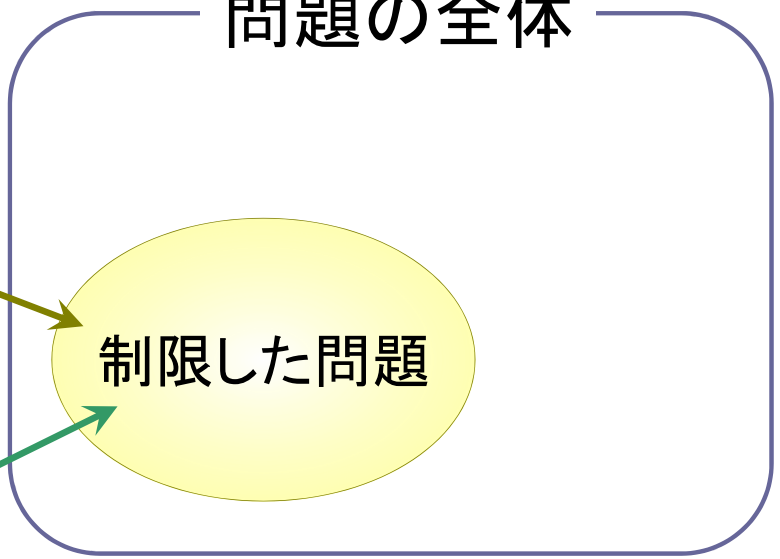


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

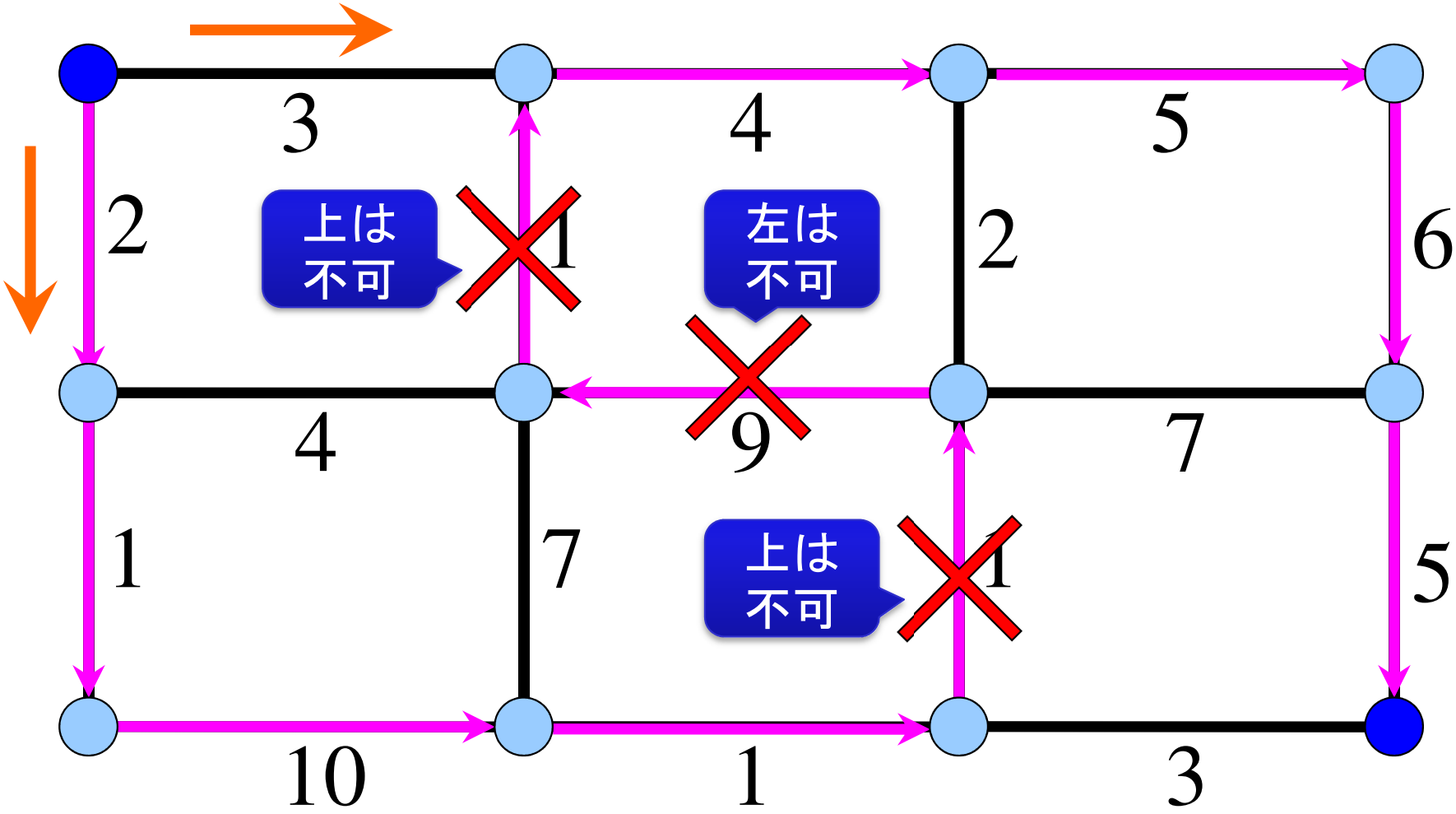
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

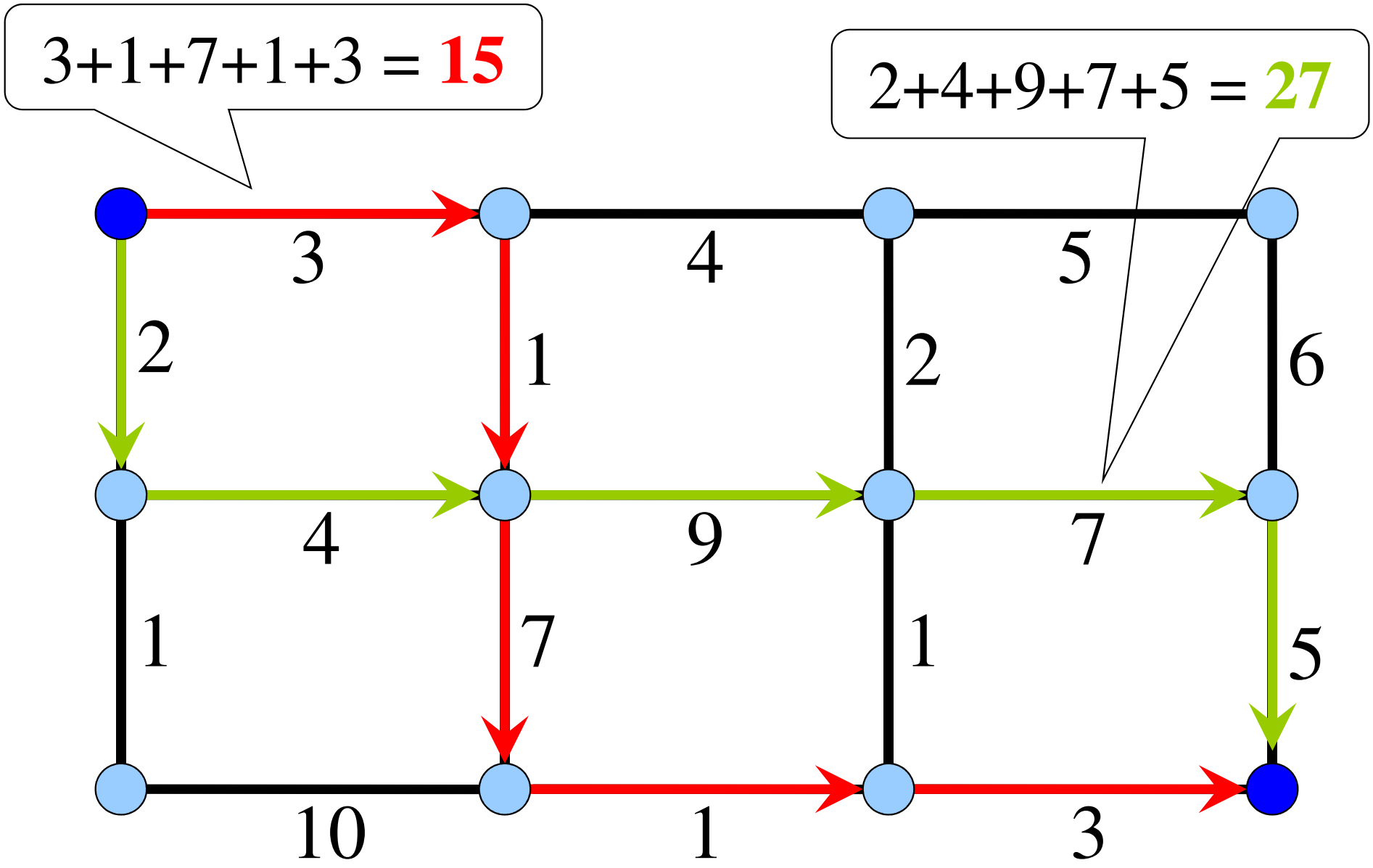
難しいなら易しくすればいいのさ！

制限した問題

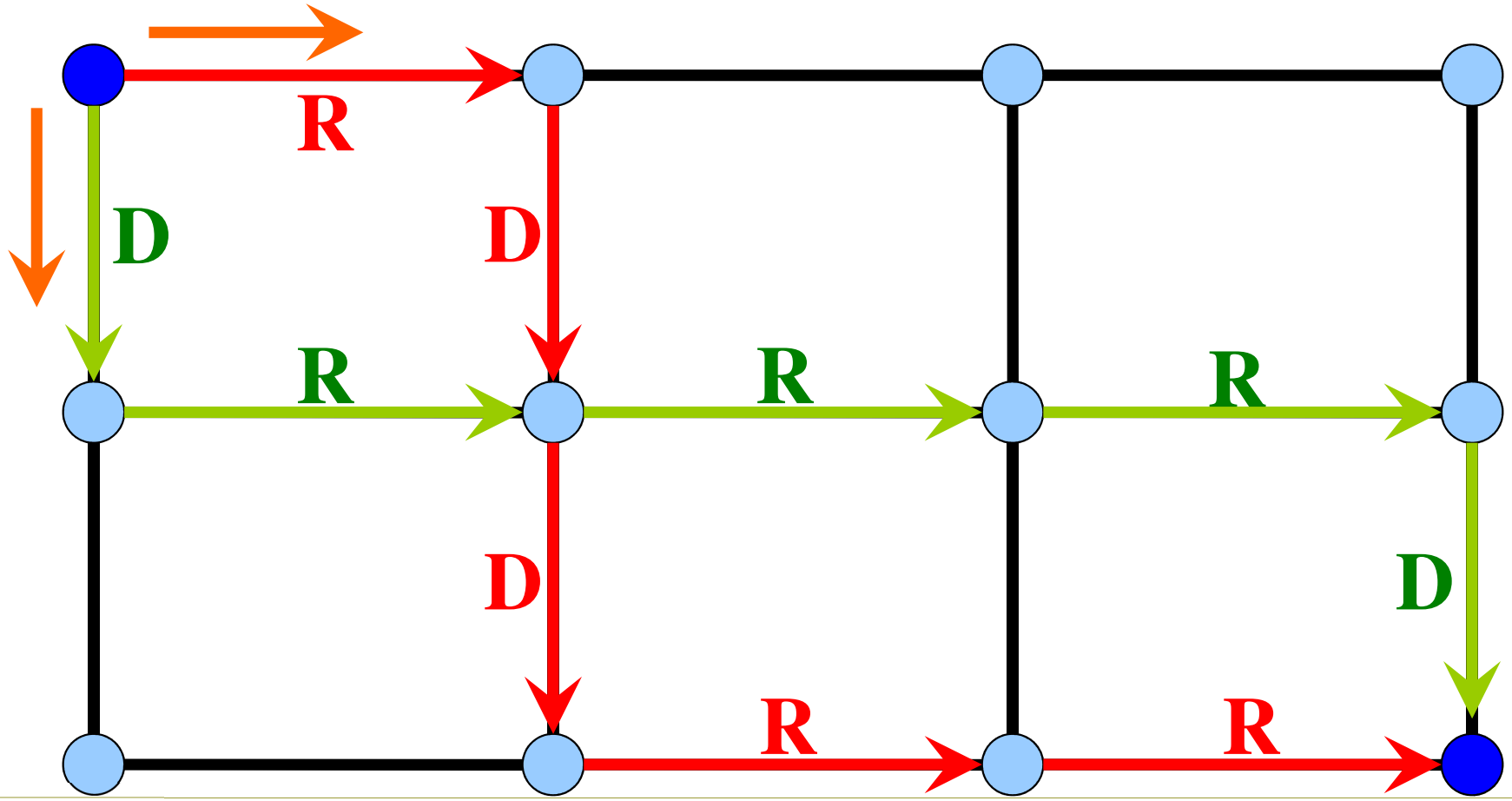
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



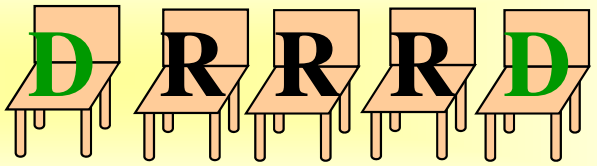
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



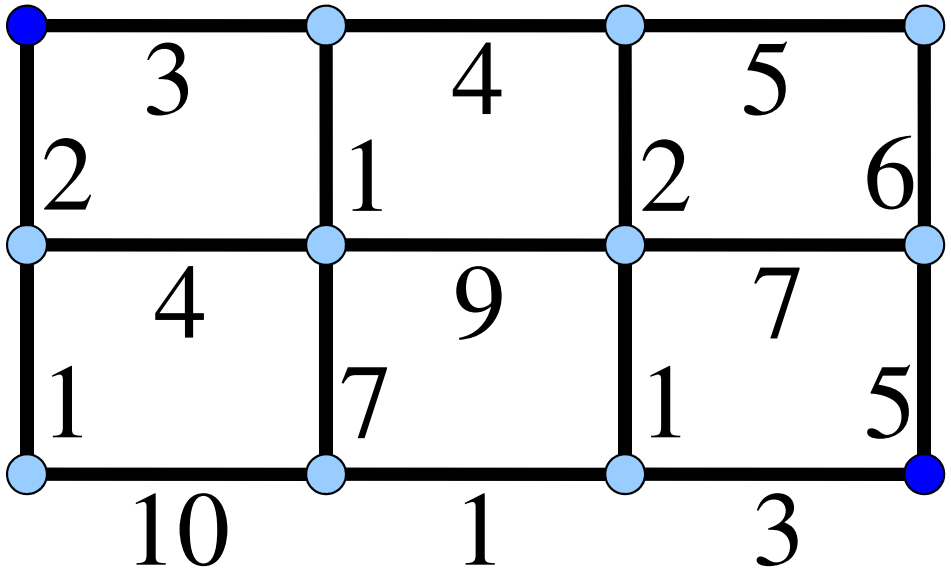
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

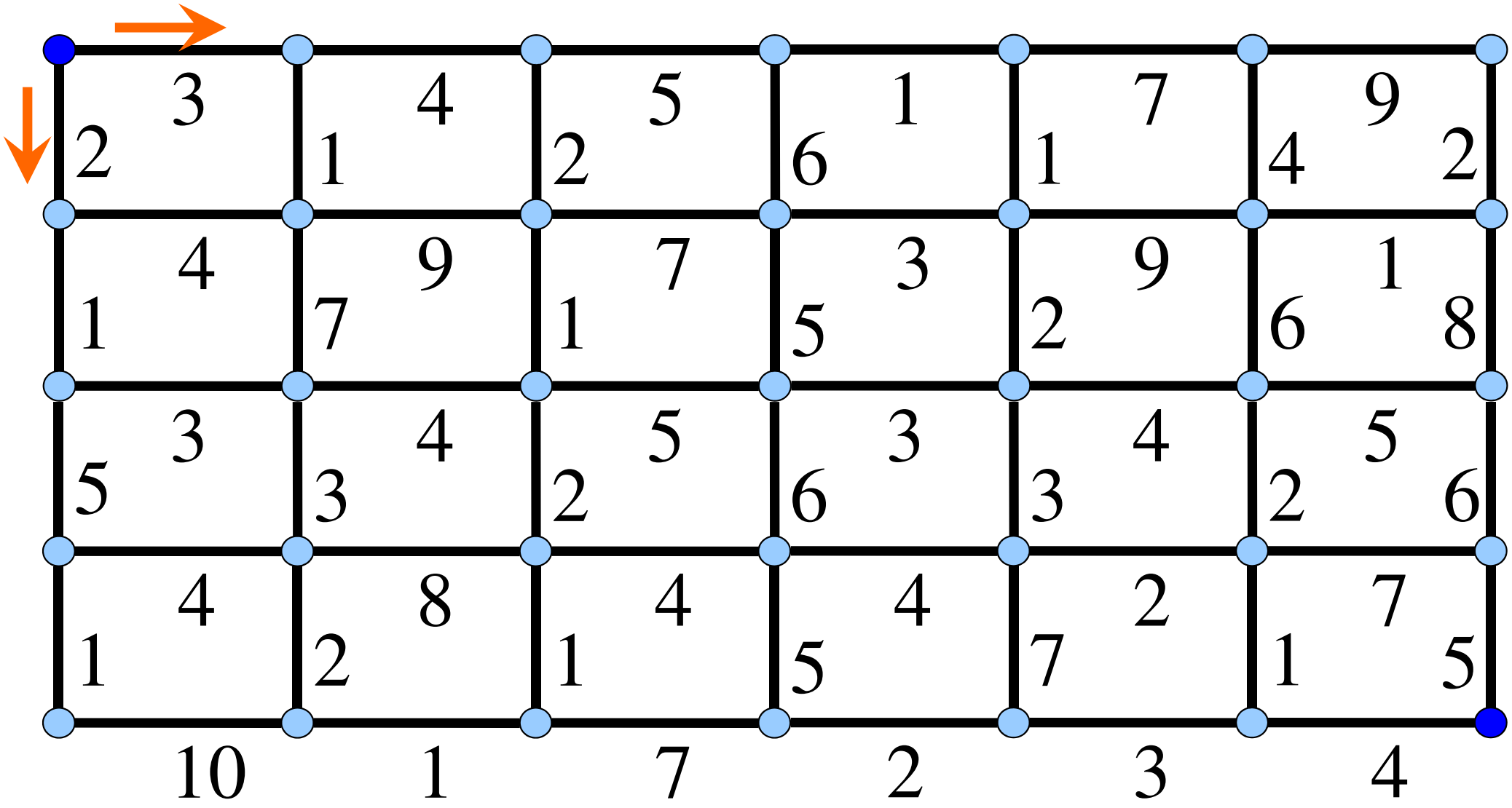
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

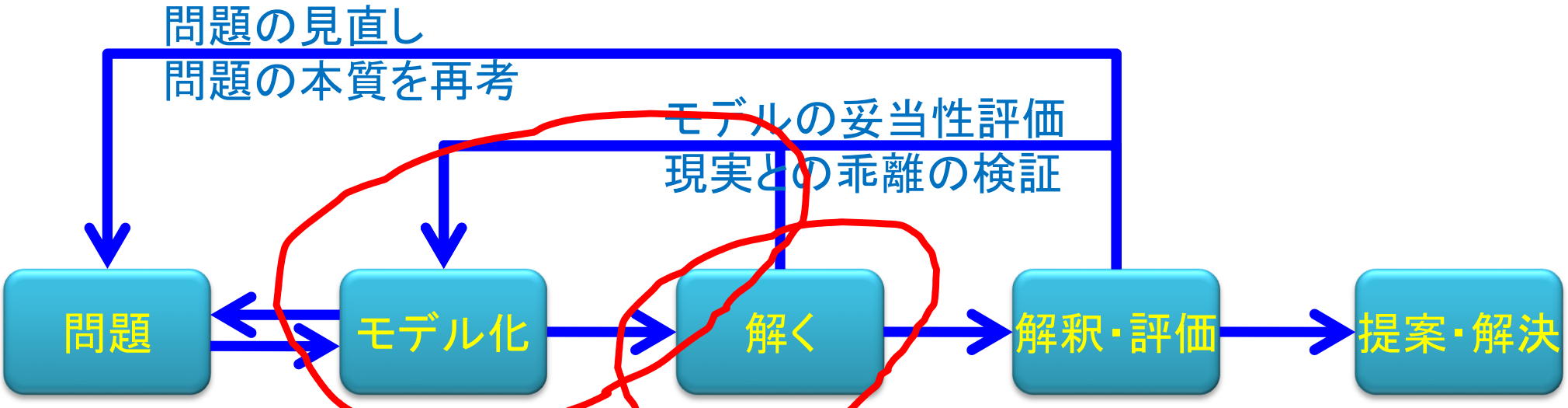
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

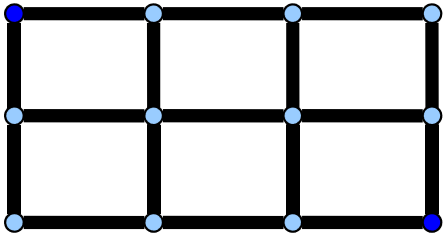
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

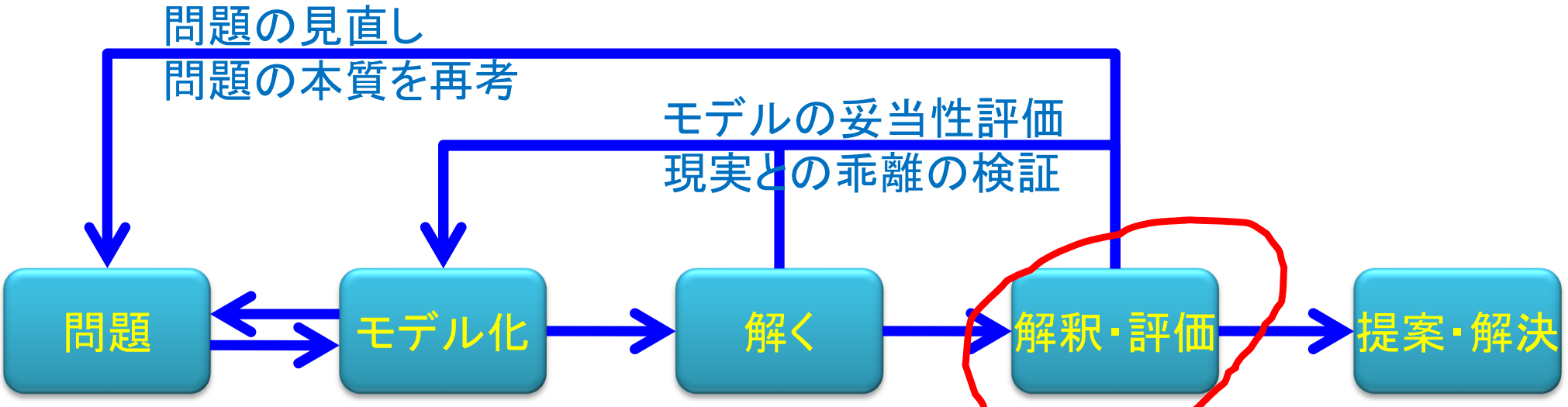
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

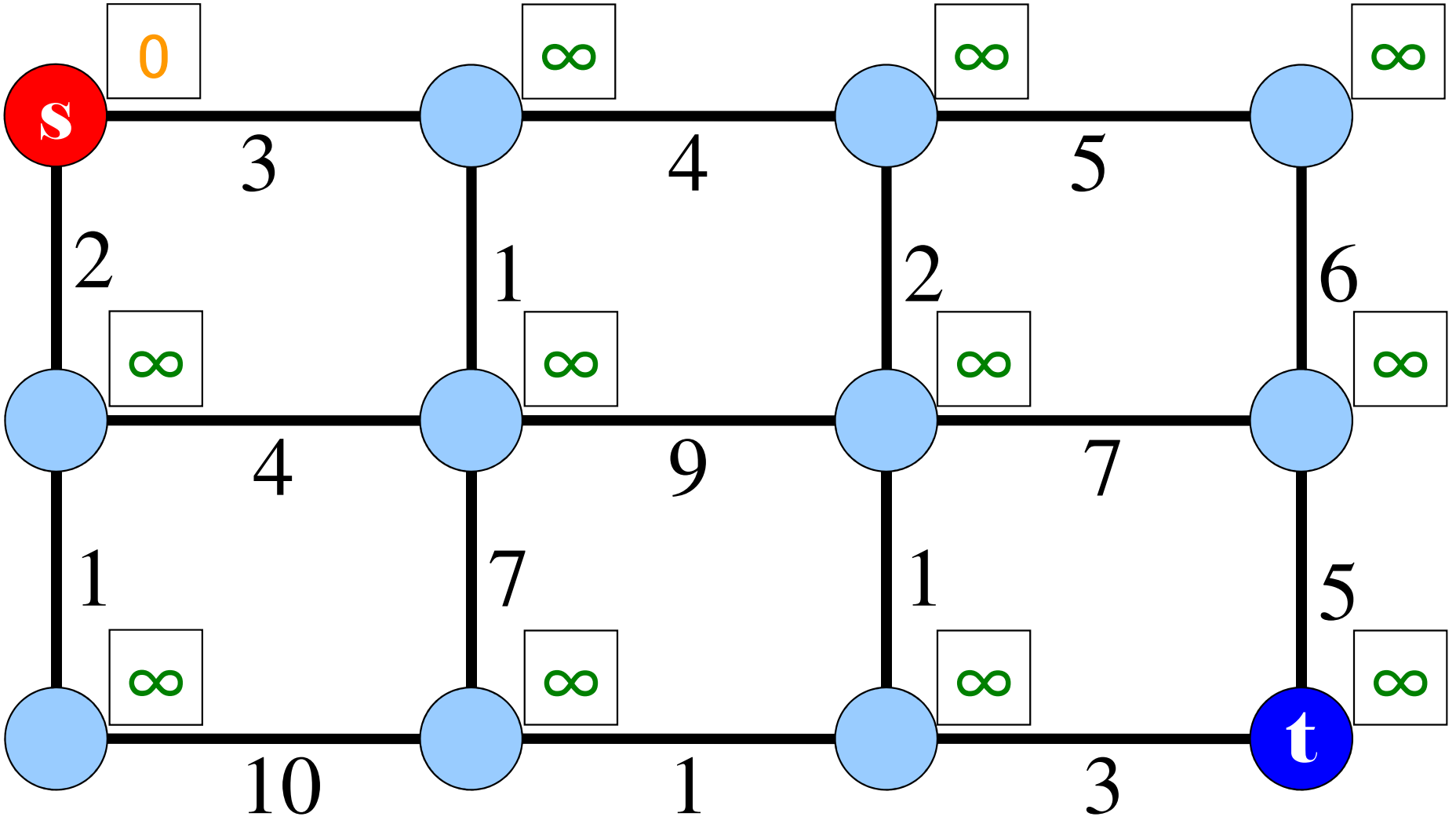
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

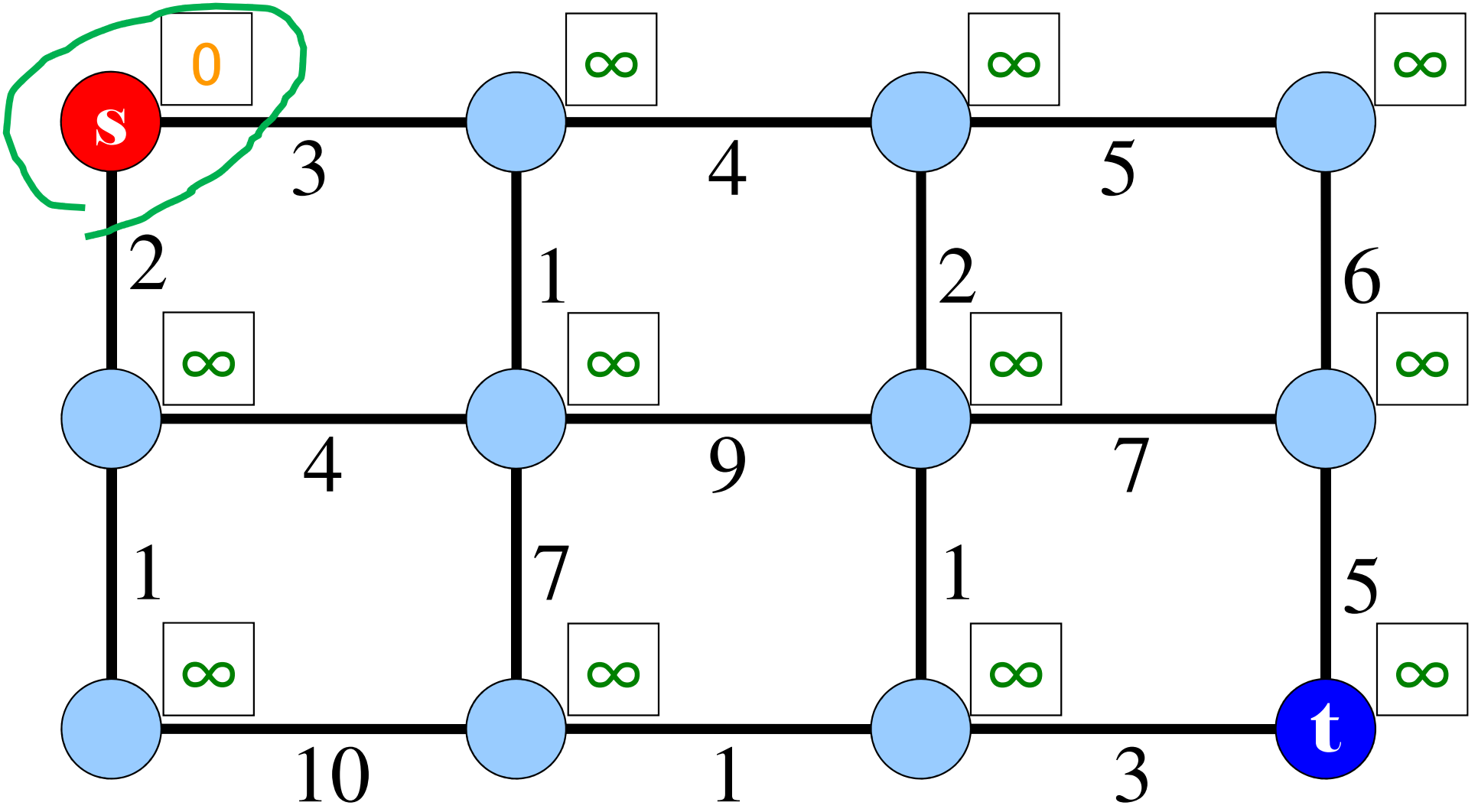
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



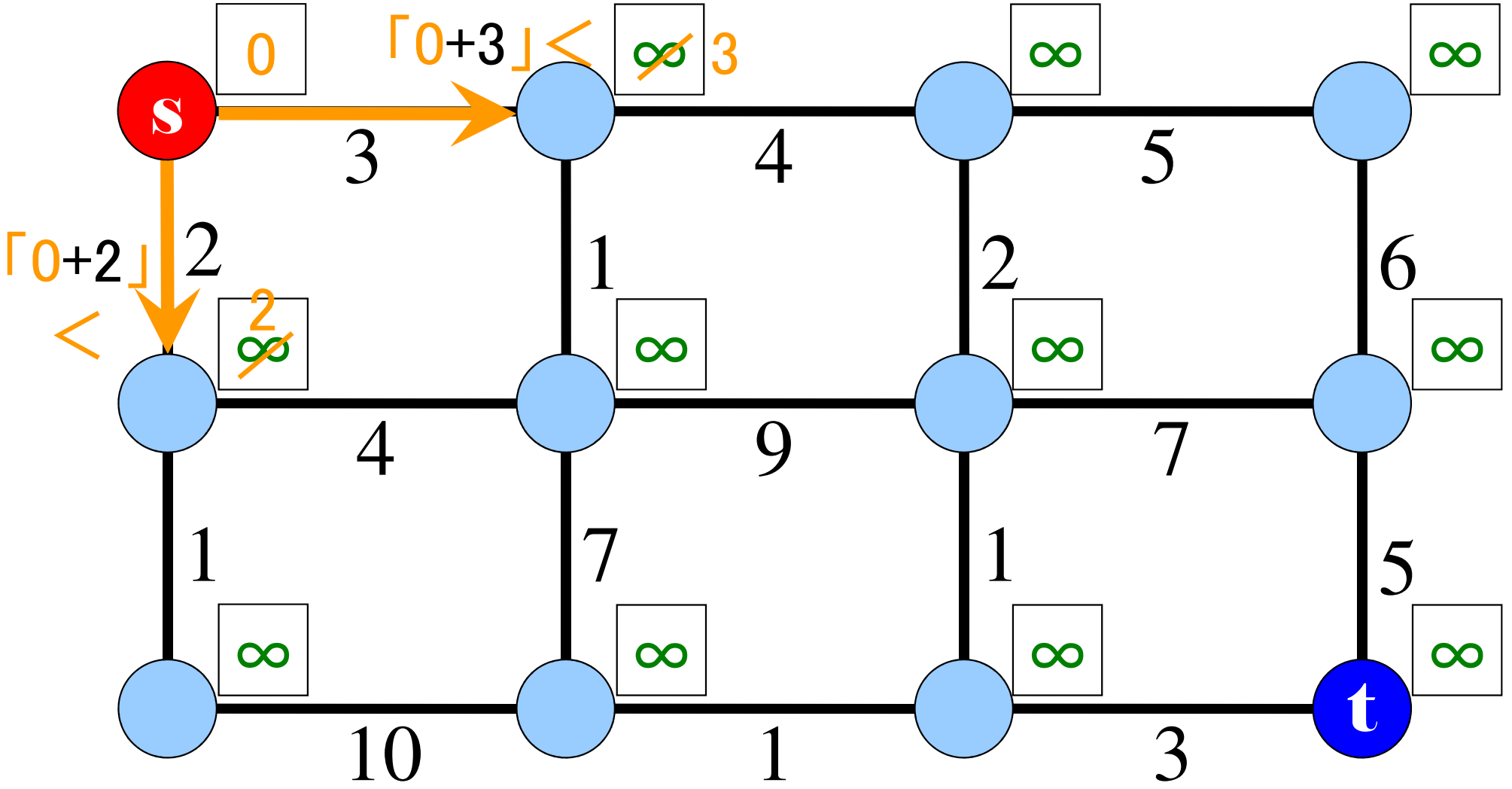
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

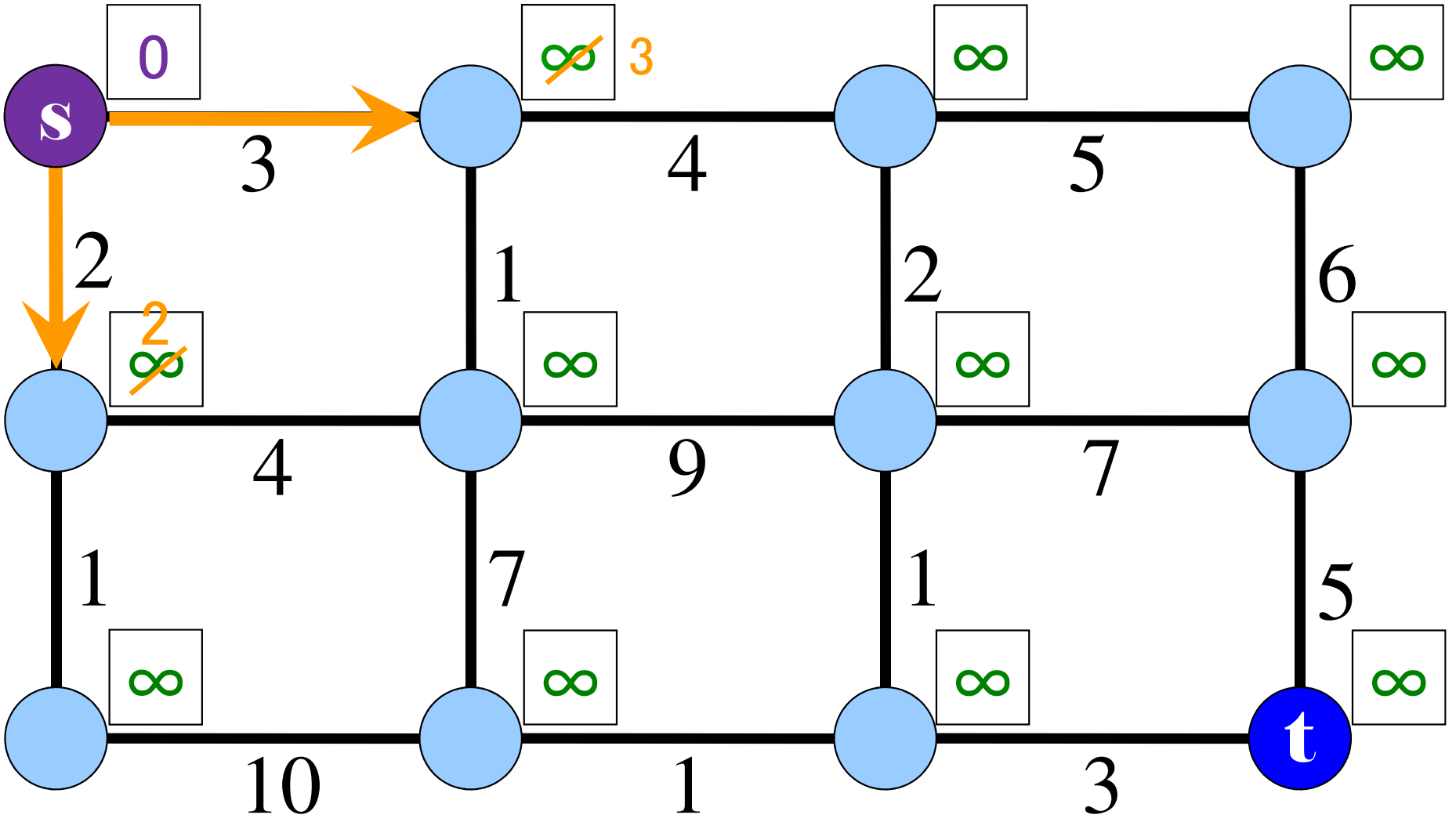


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

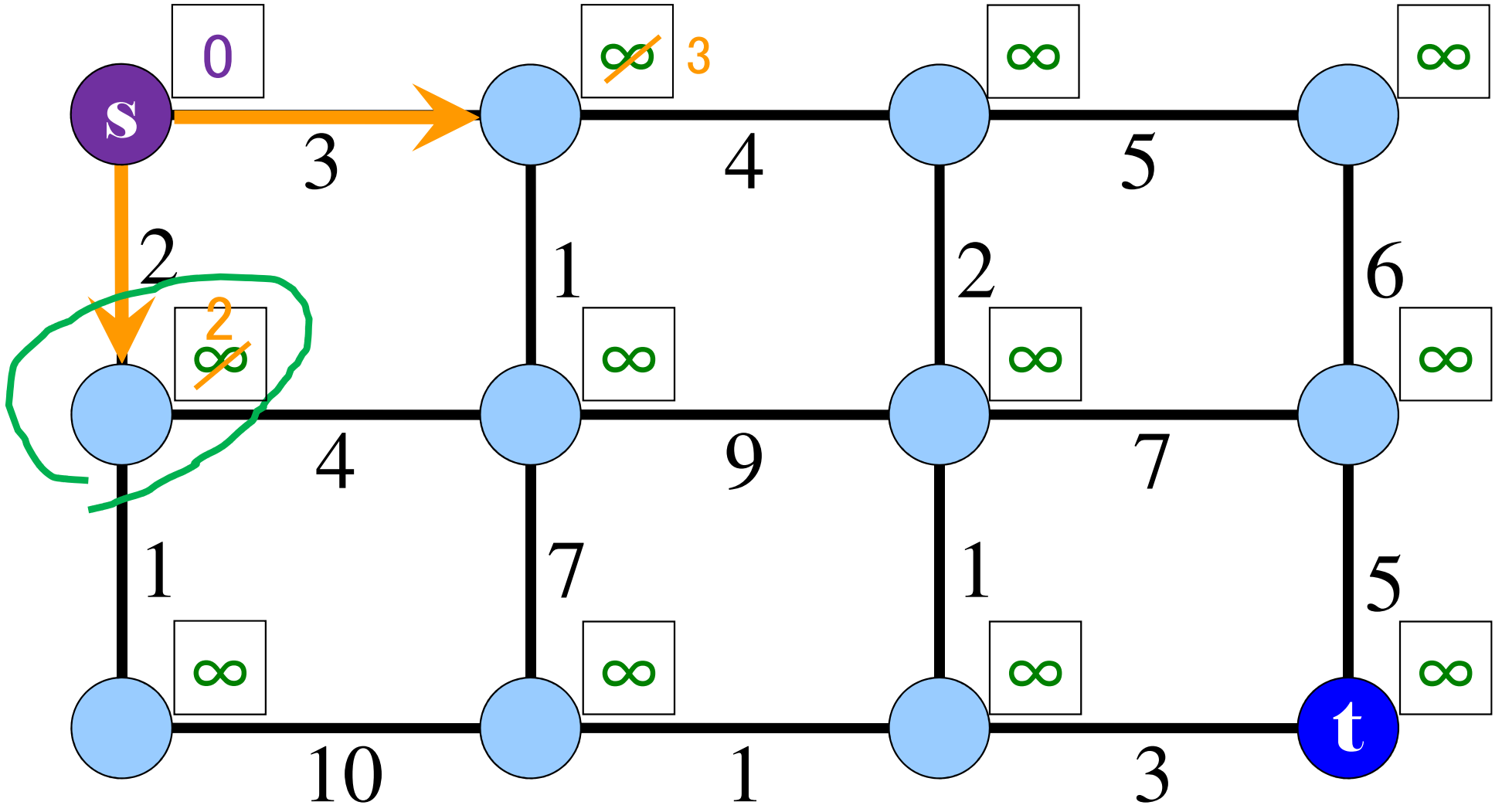
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



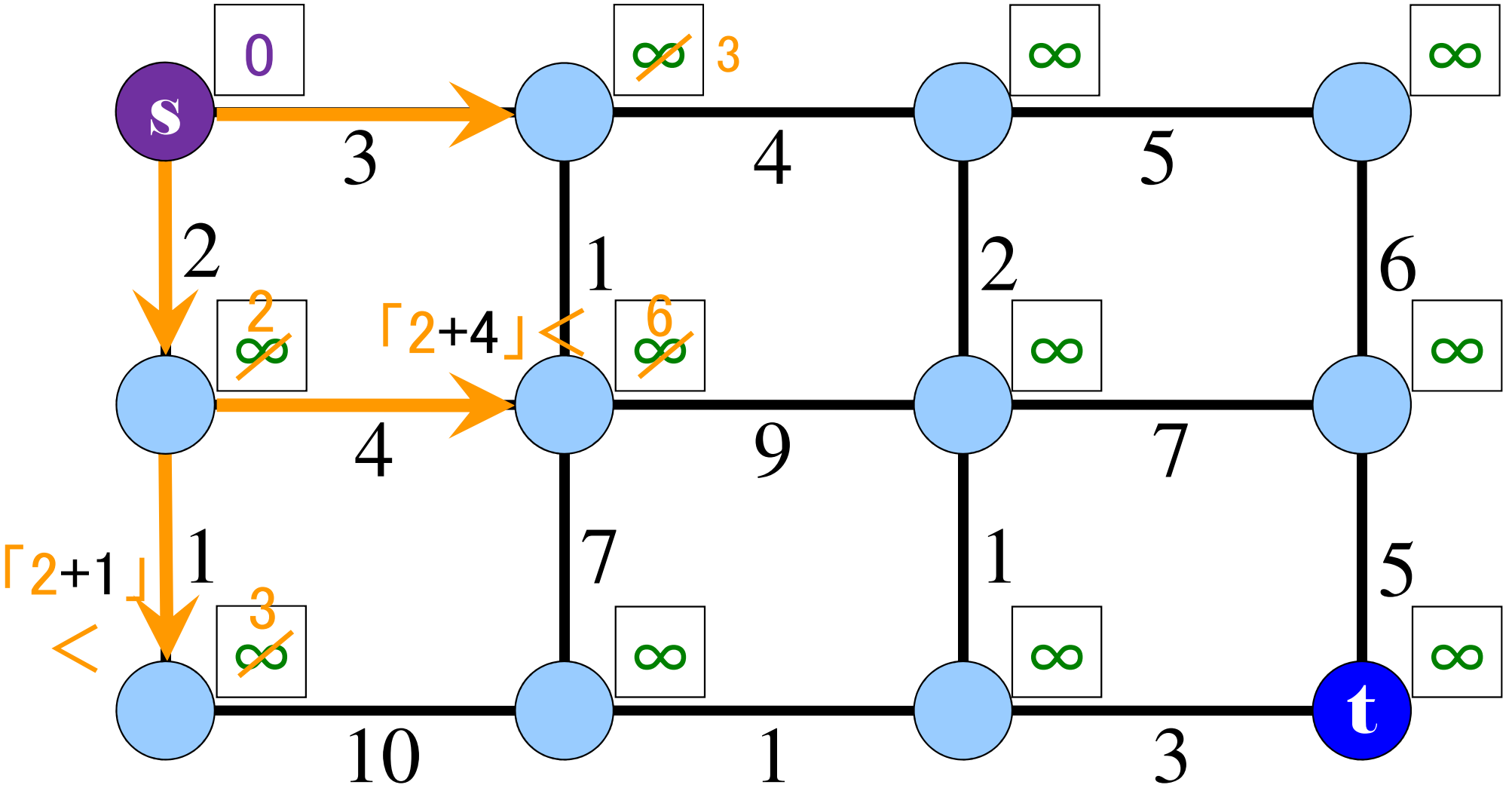
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

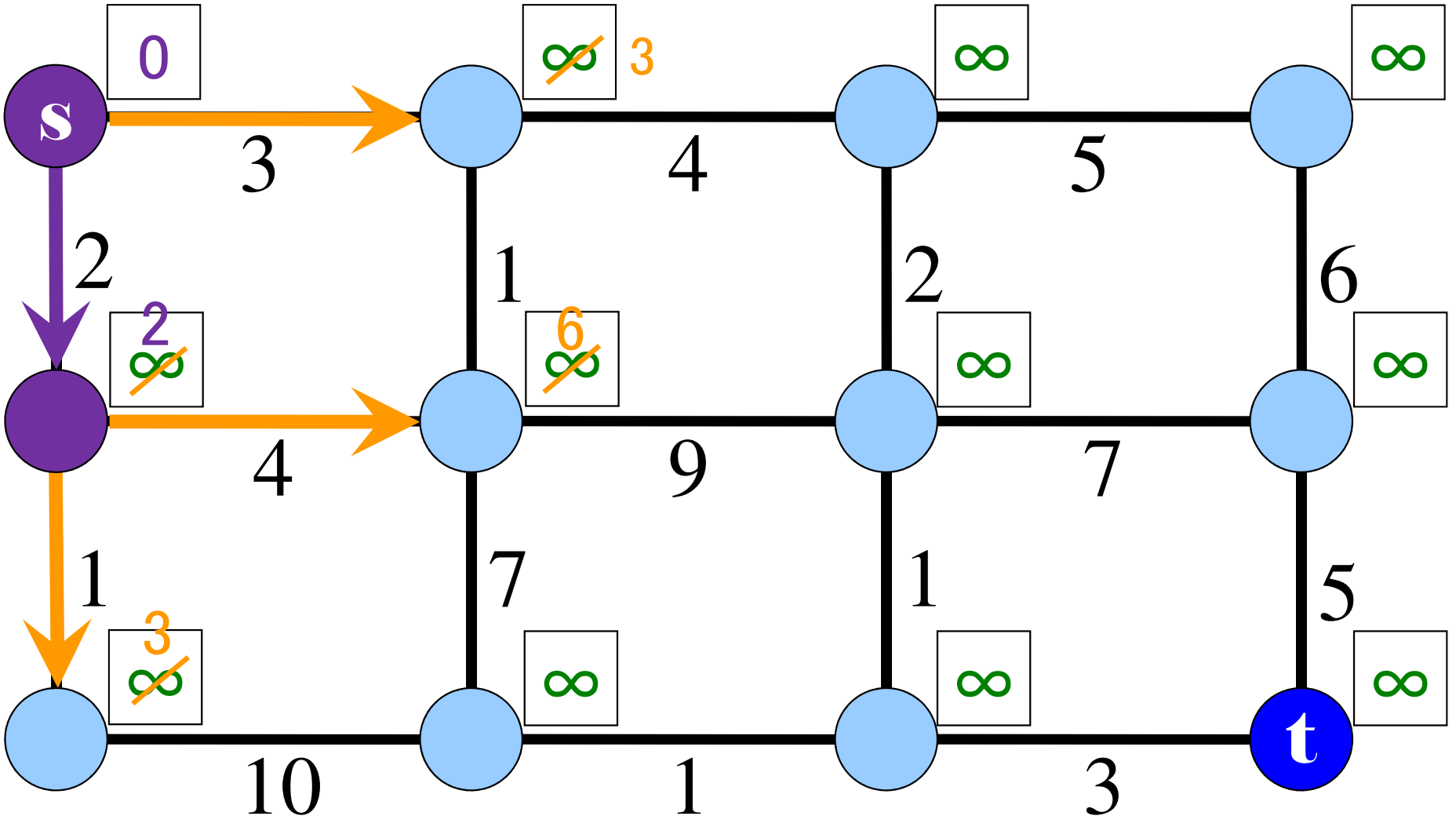


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

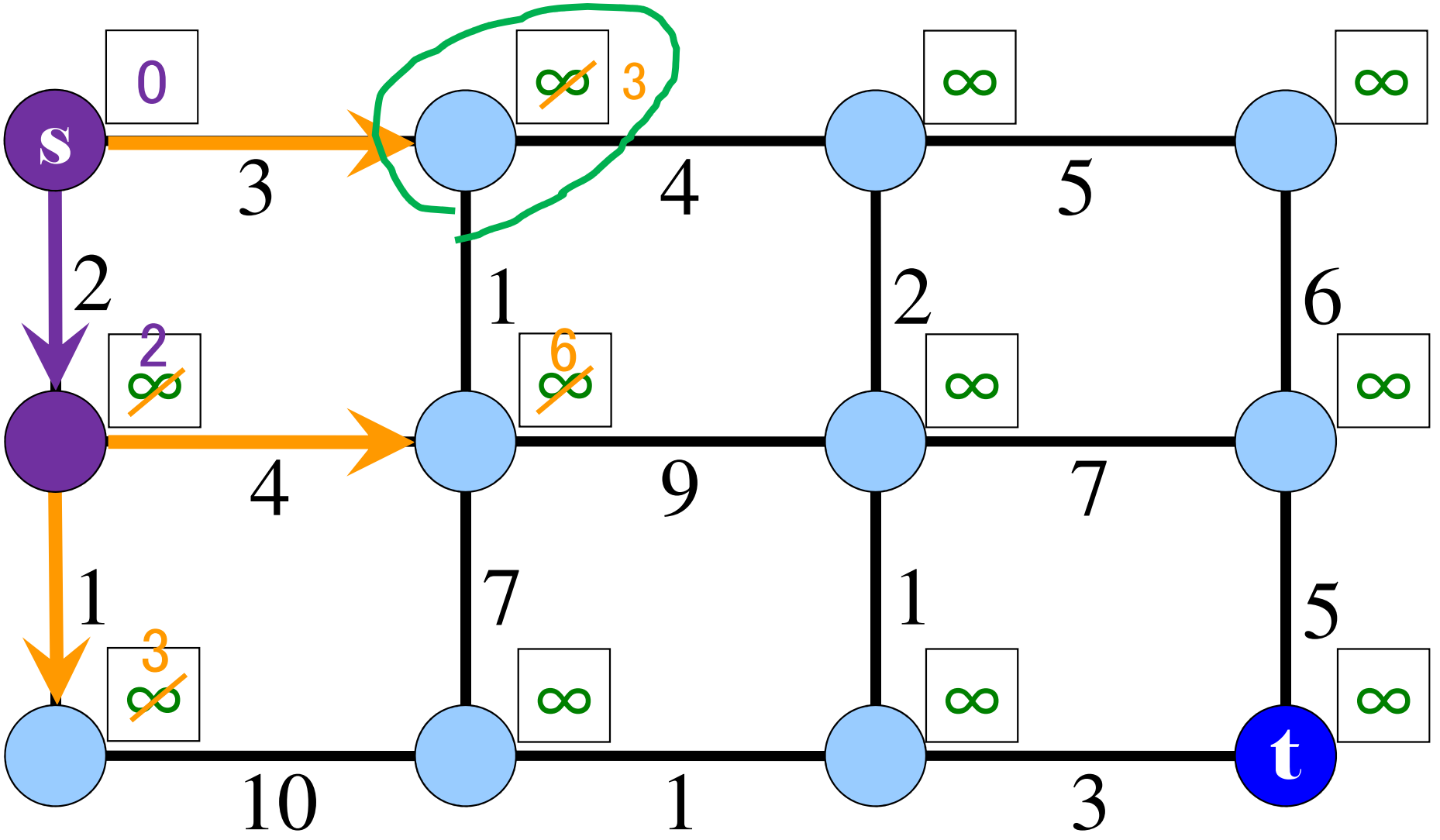
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



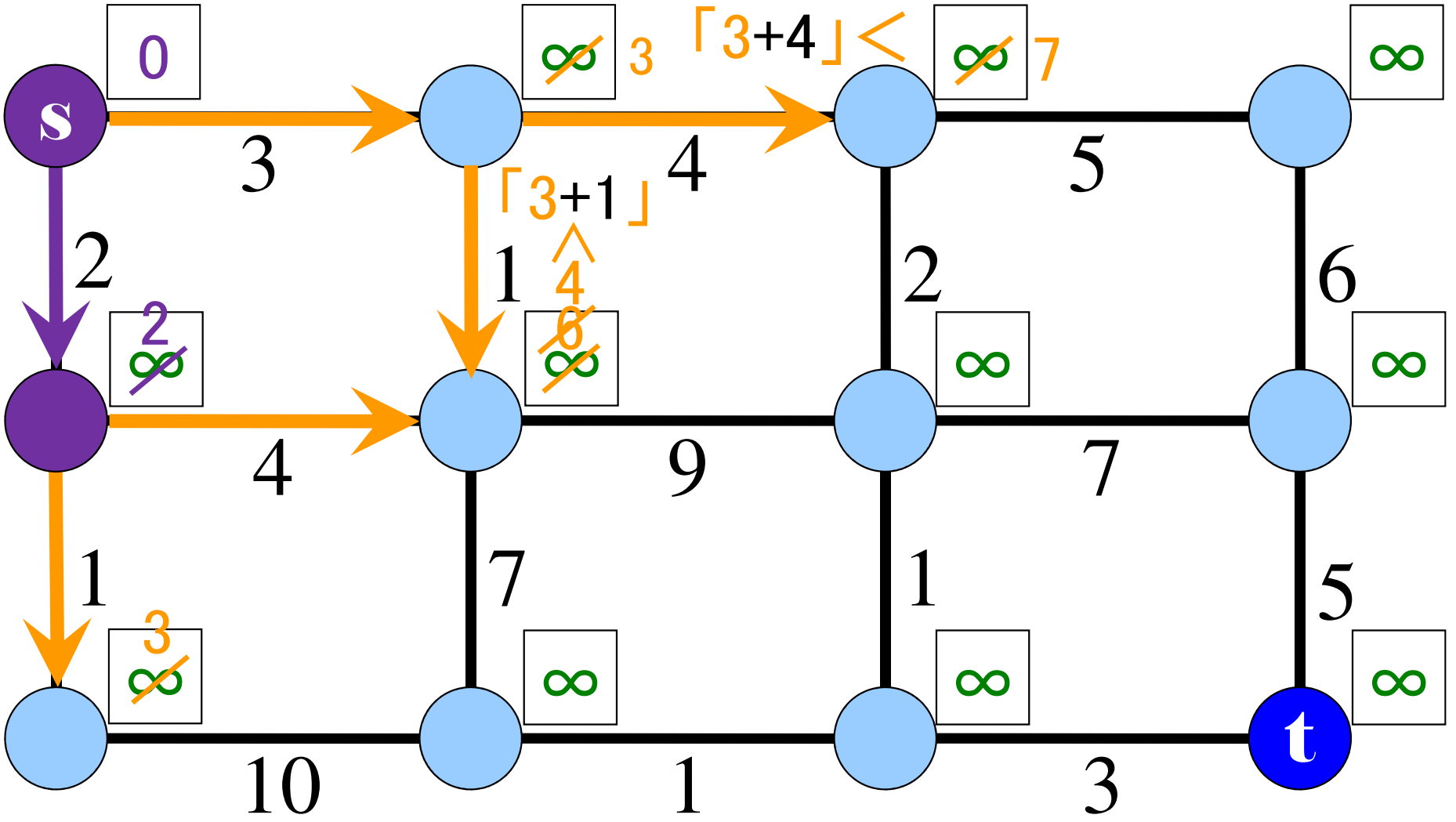
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

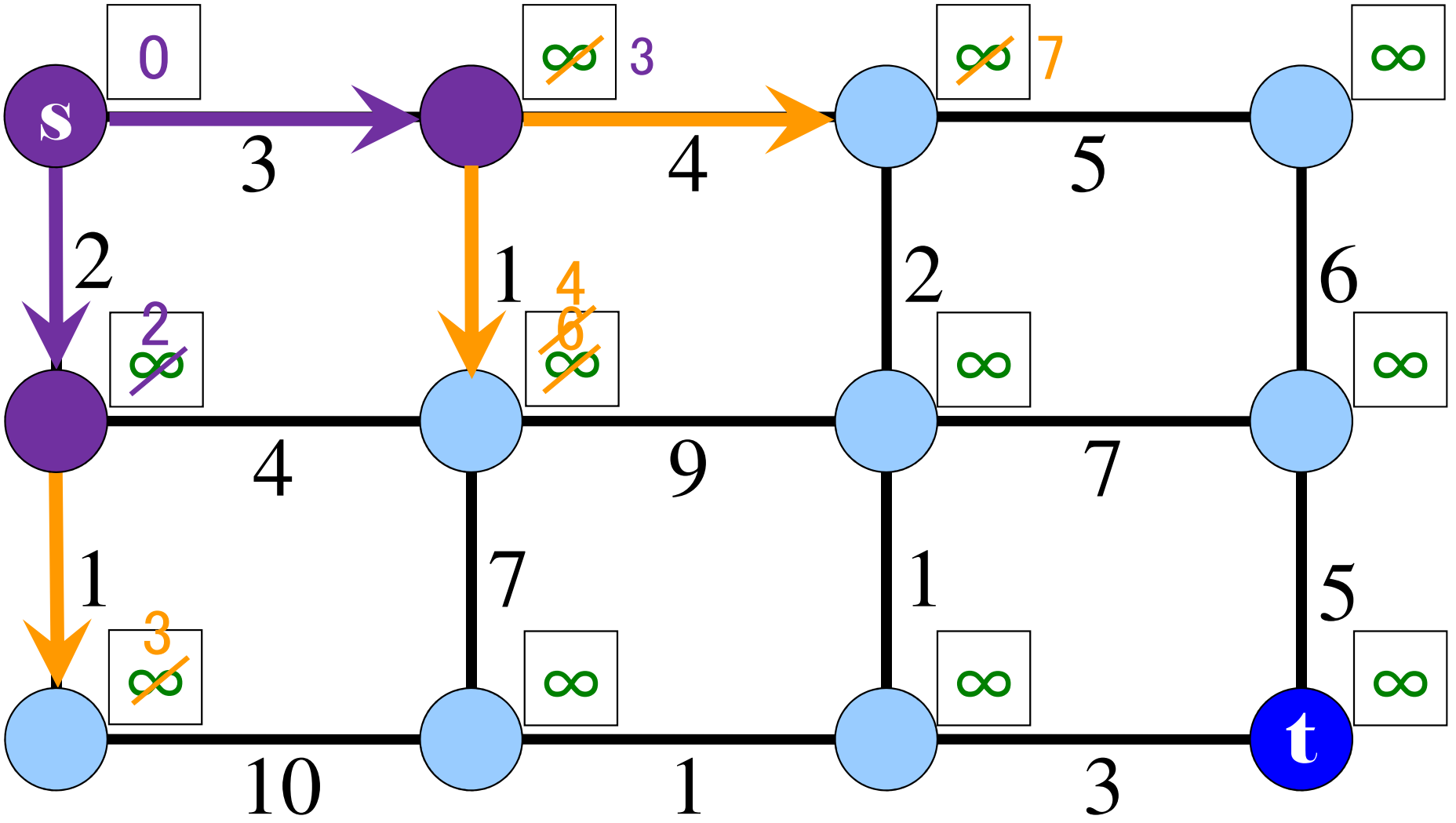


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

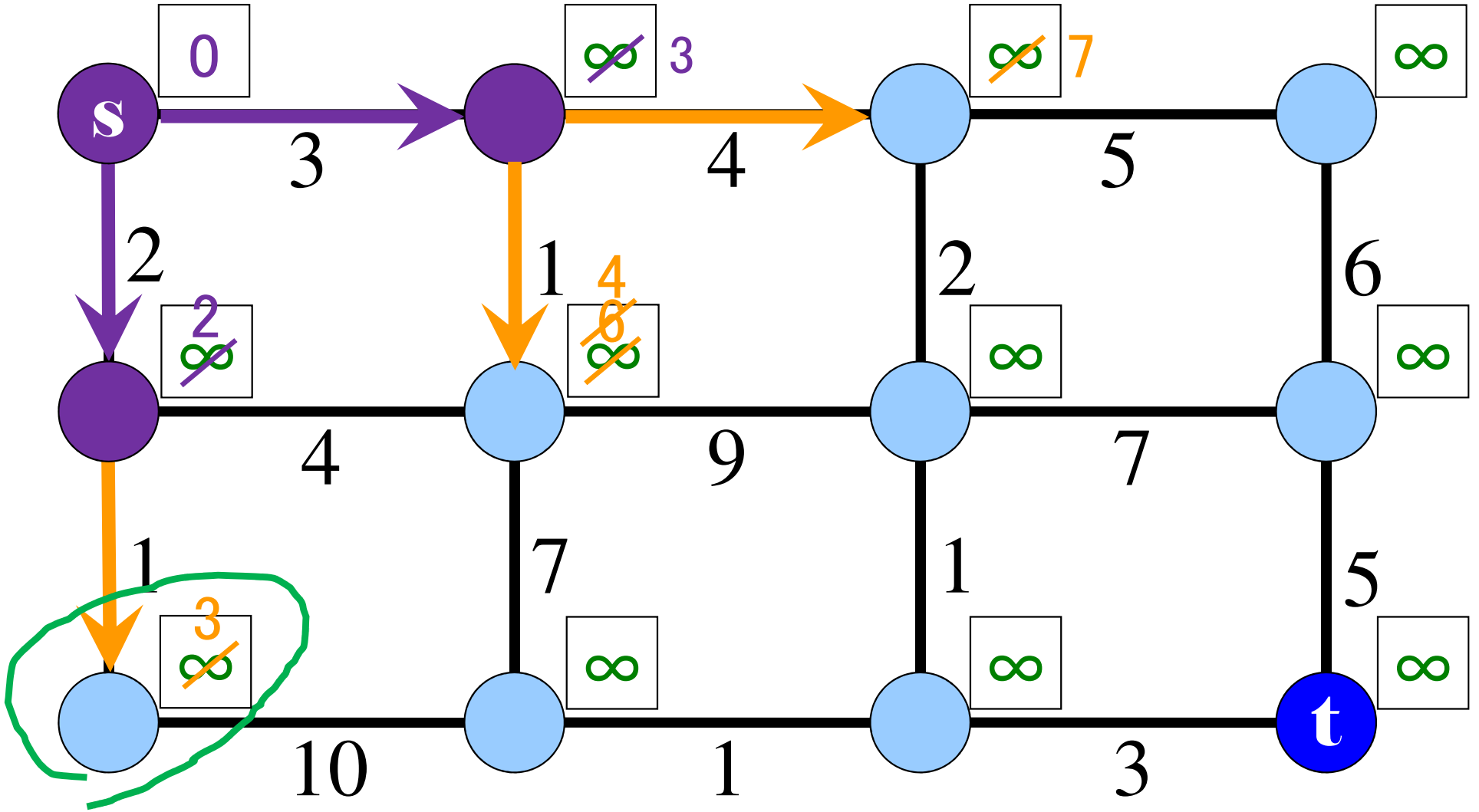
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



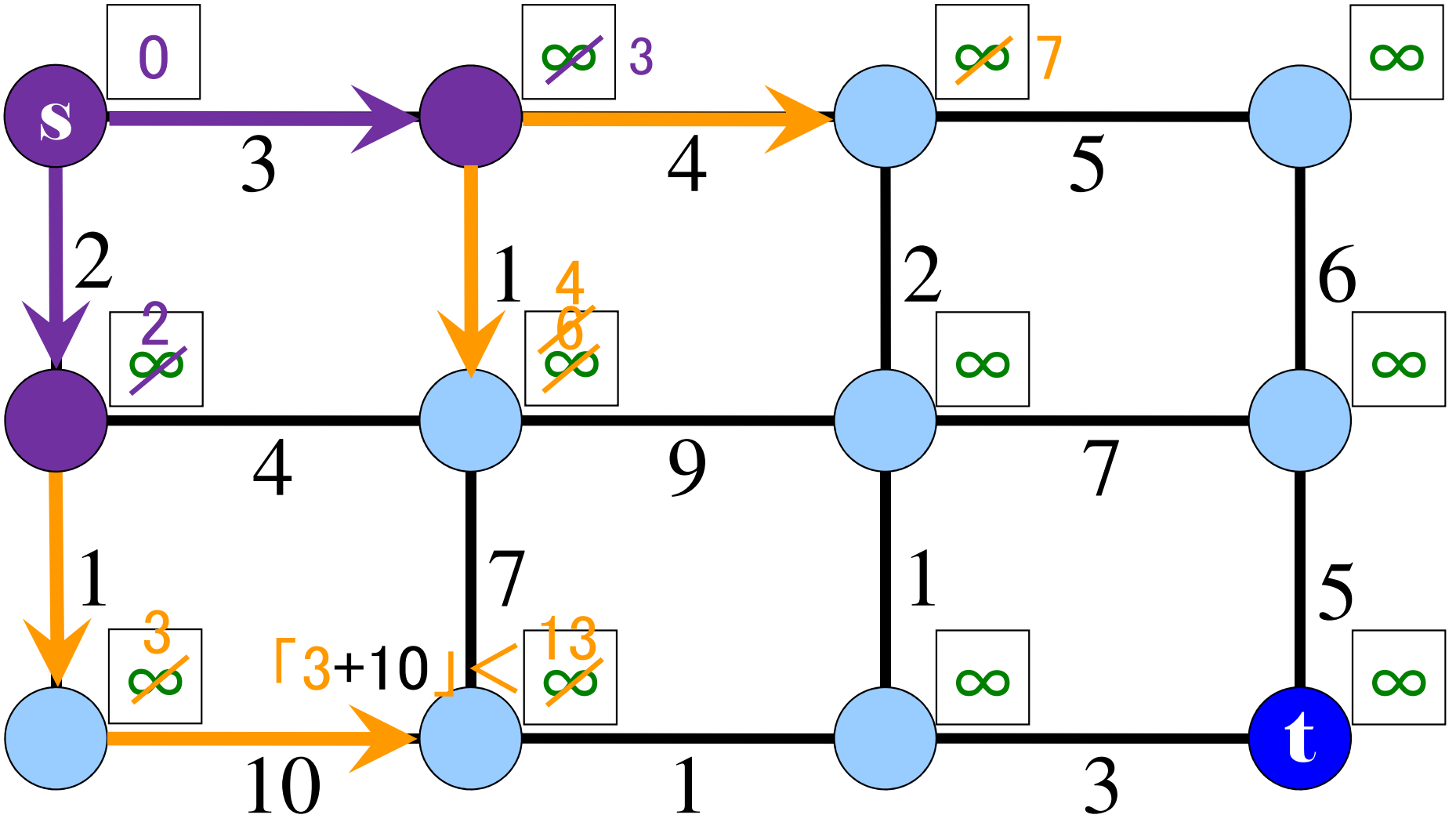
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

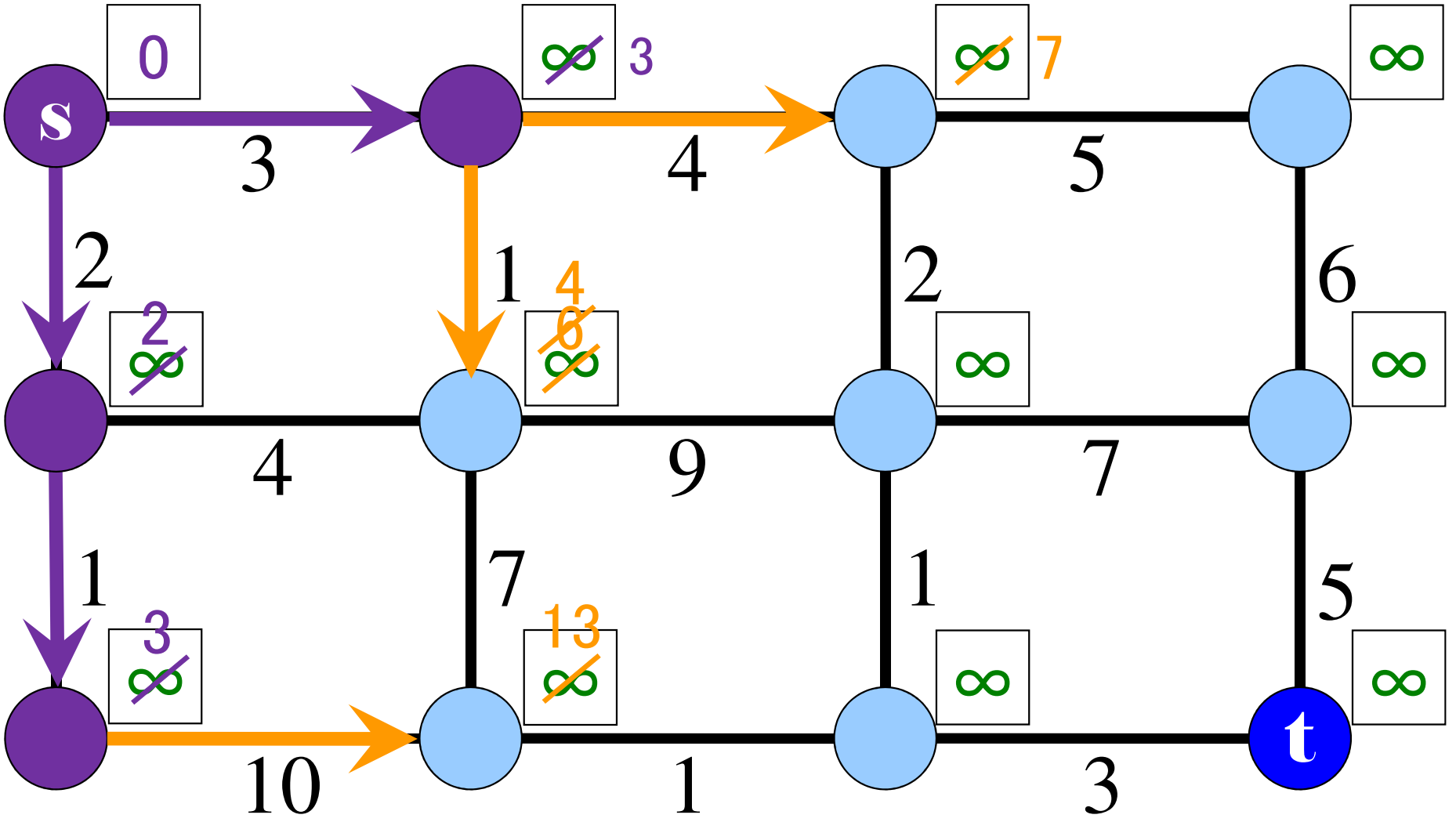


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

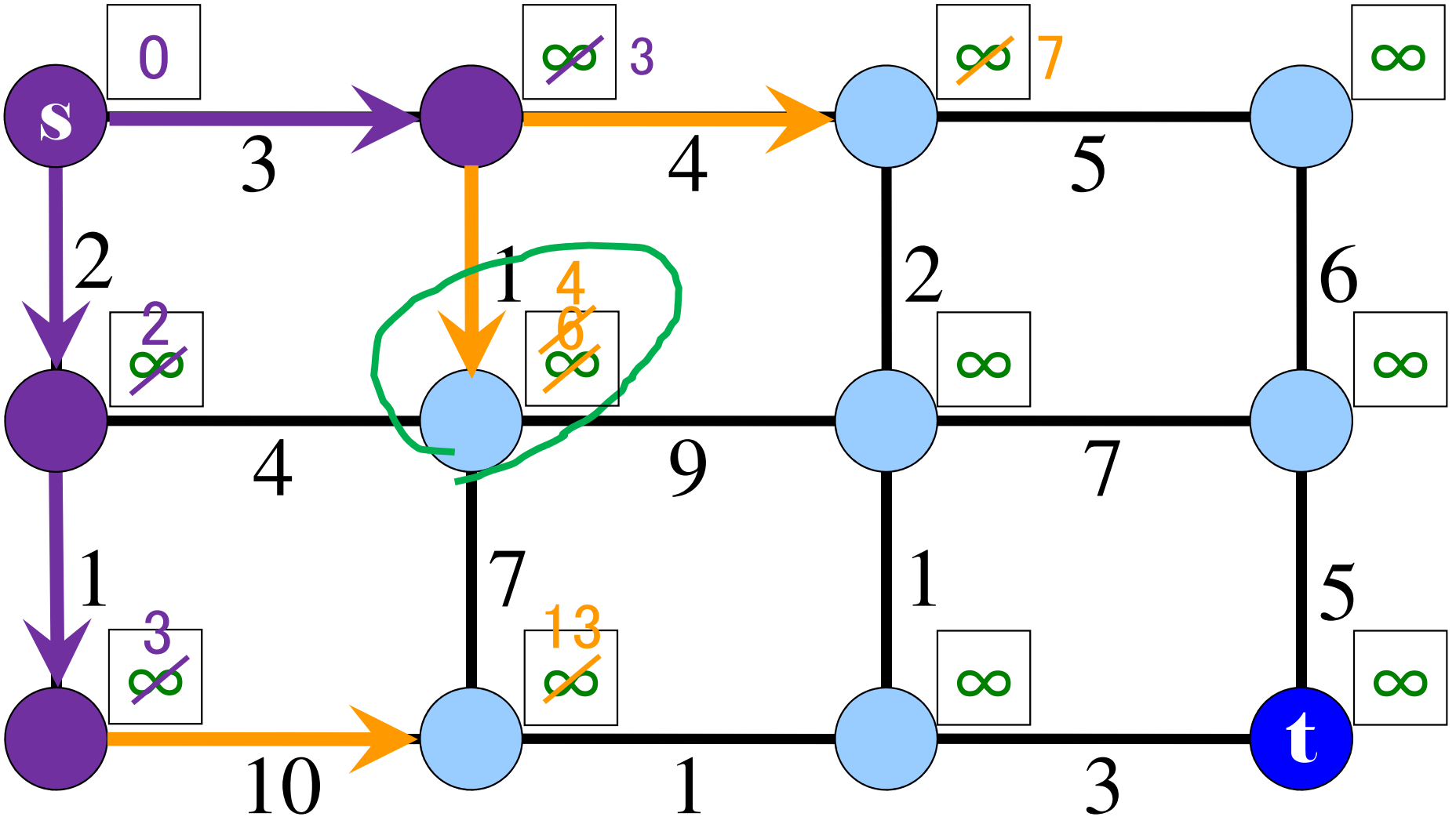
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



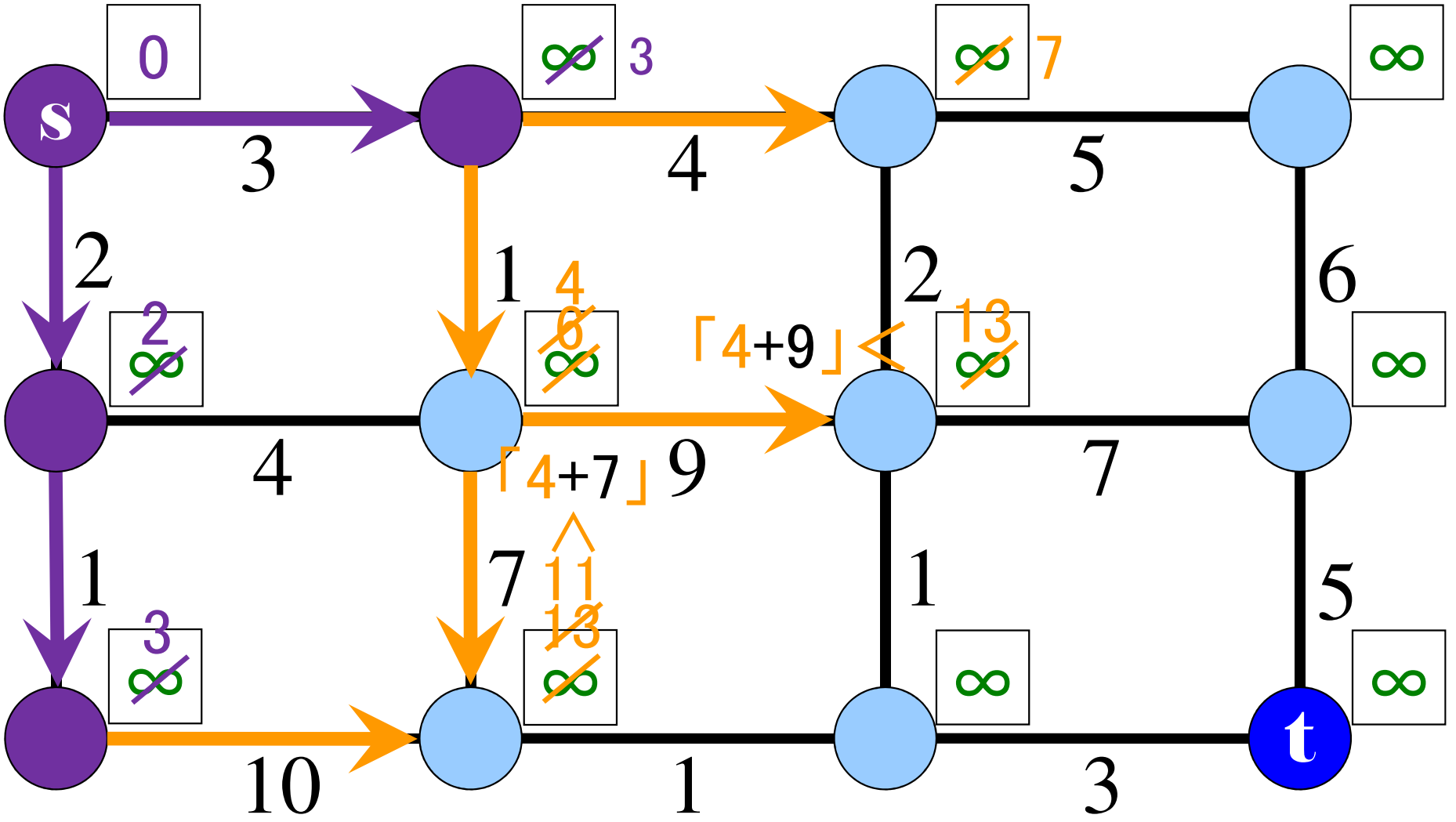
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

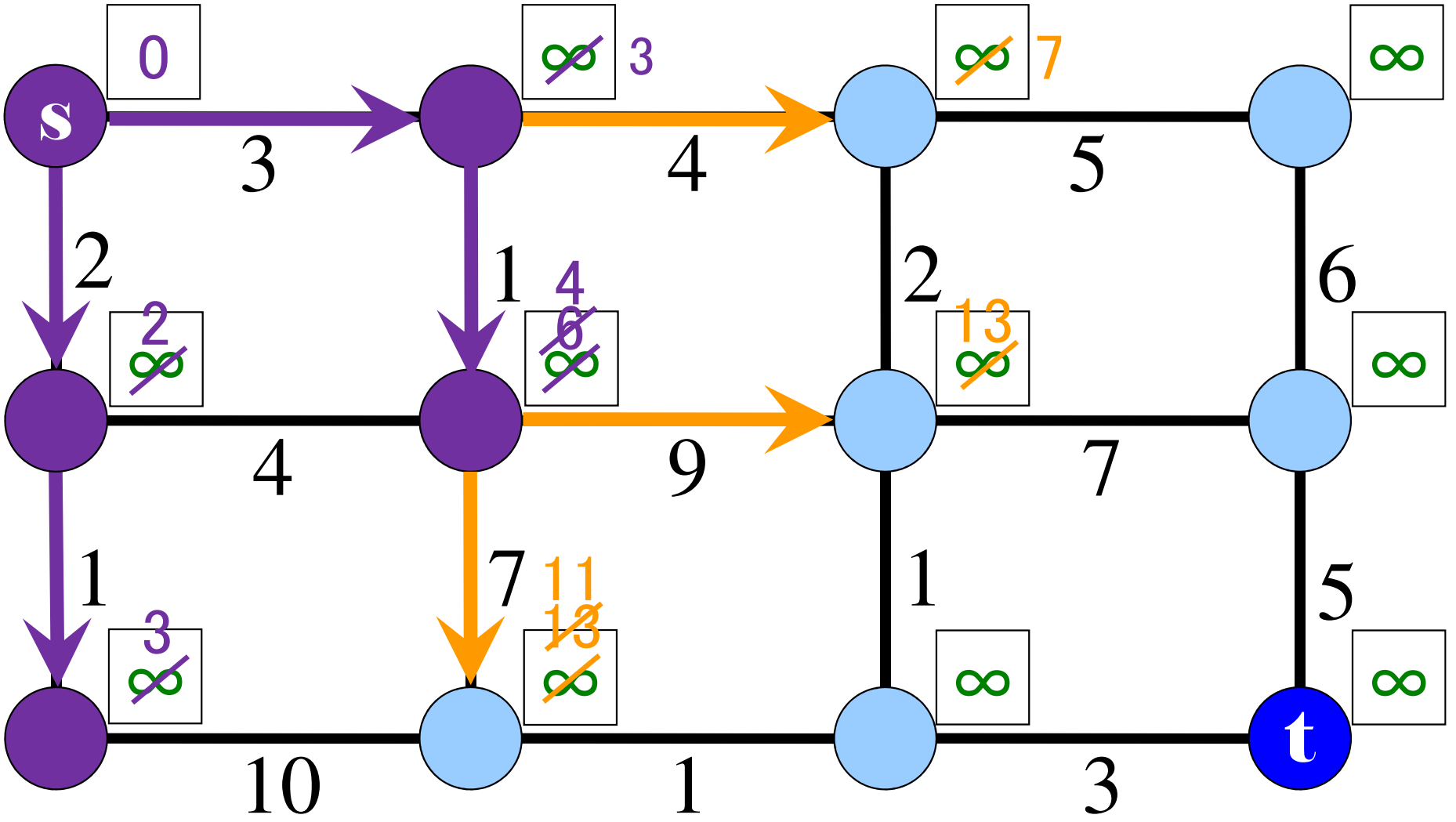


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

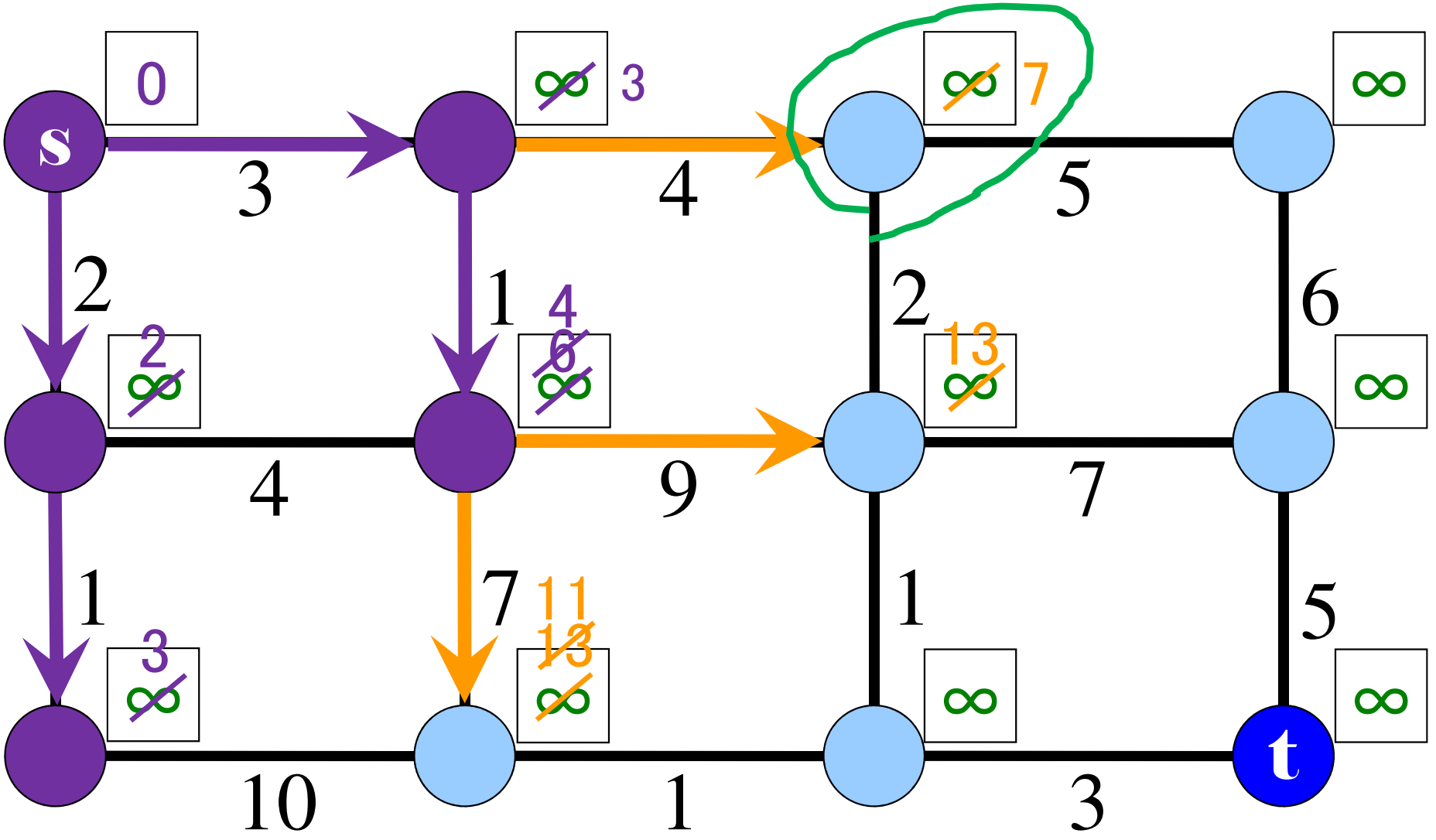
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



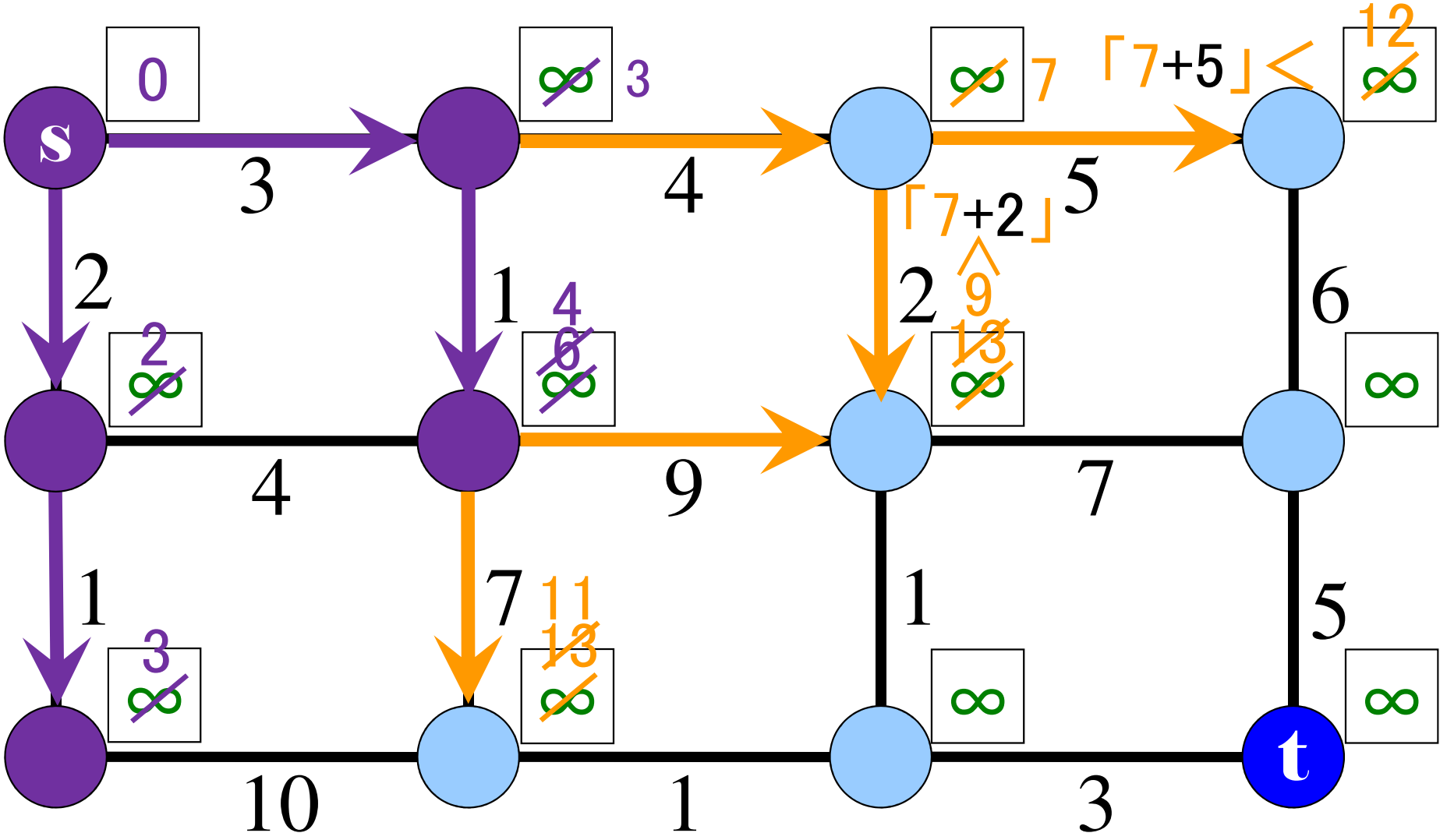
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

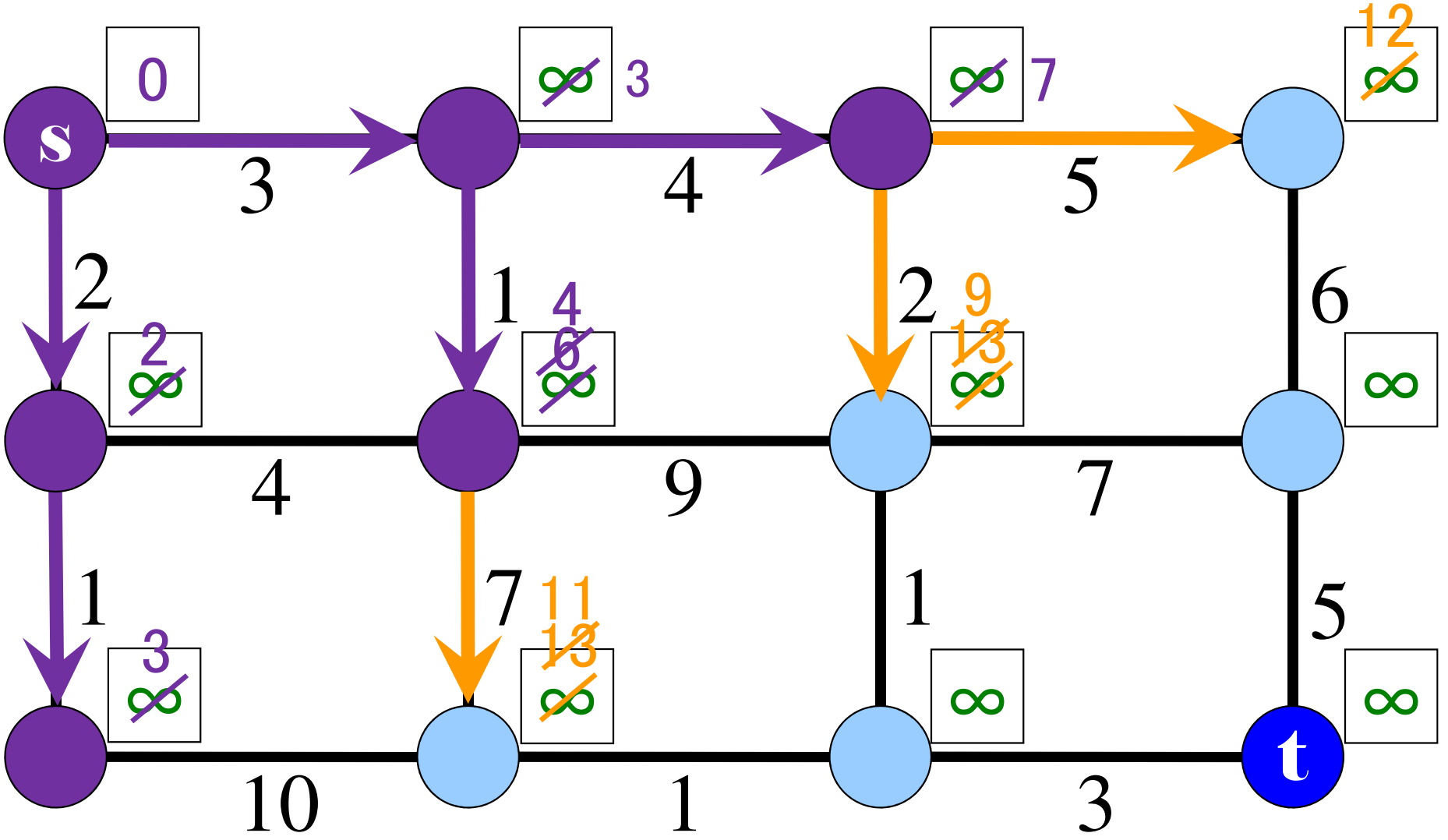


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

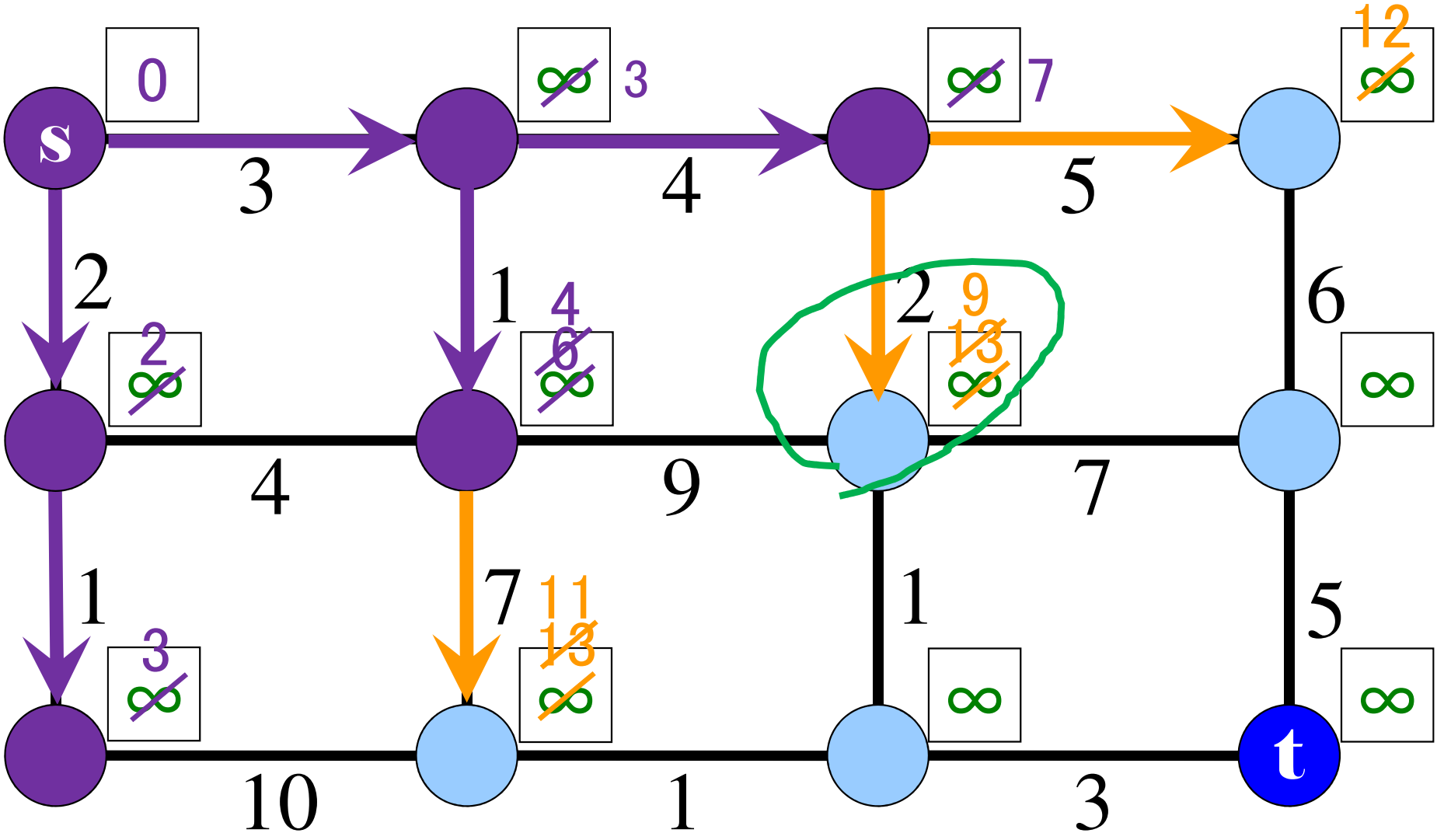
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



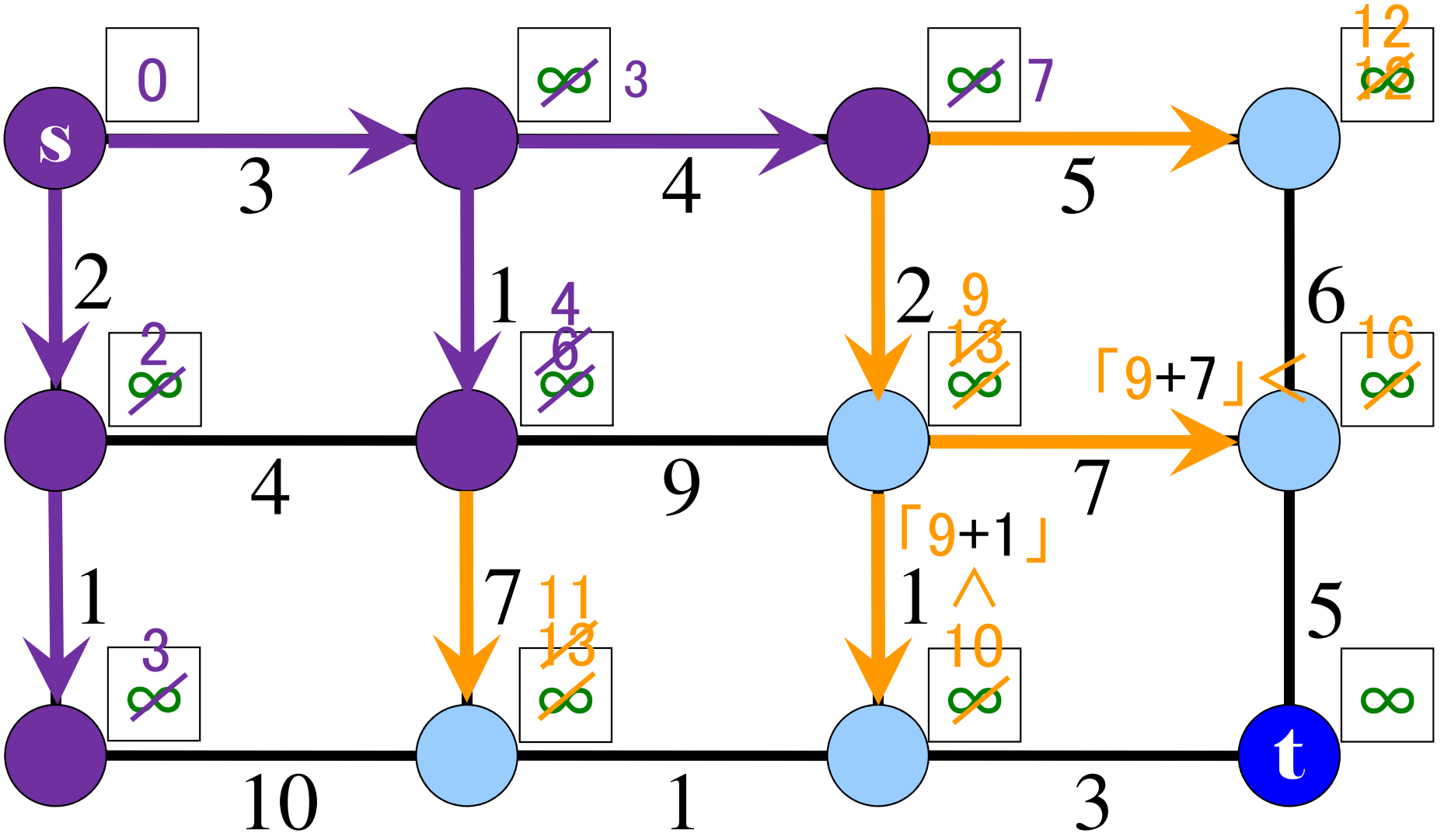
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

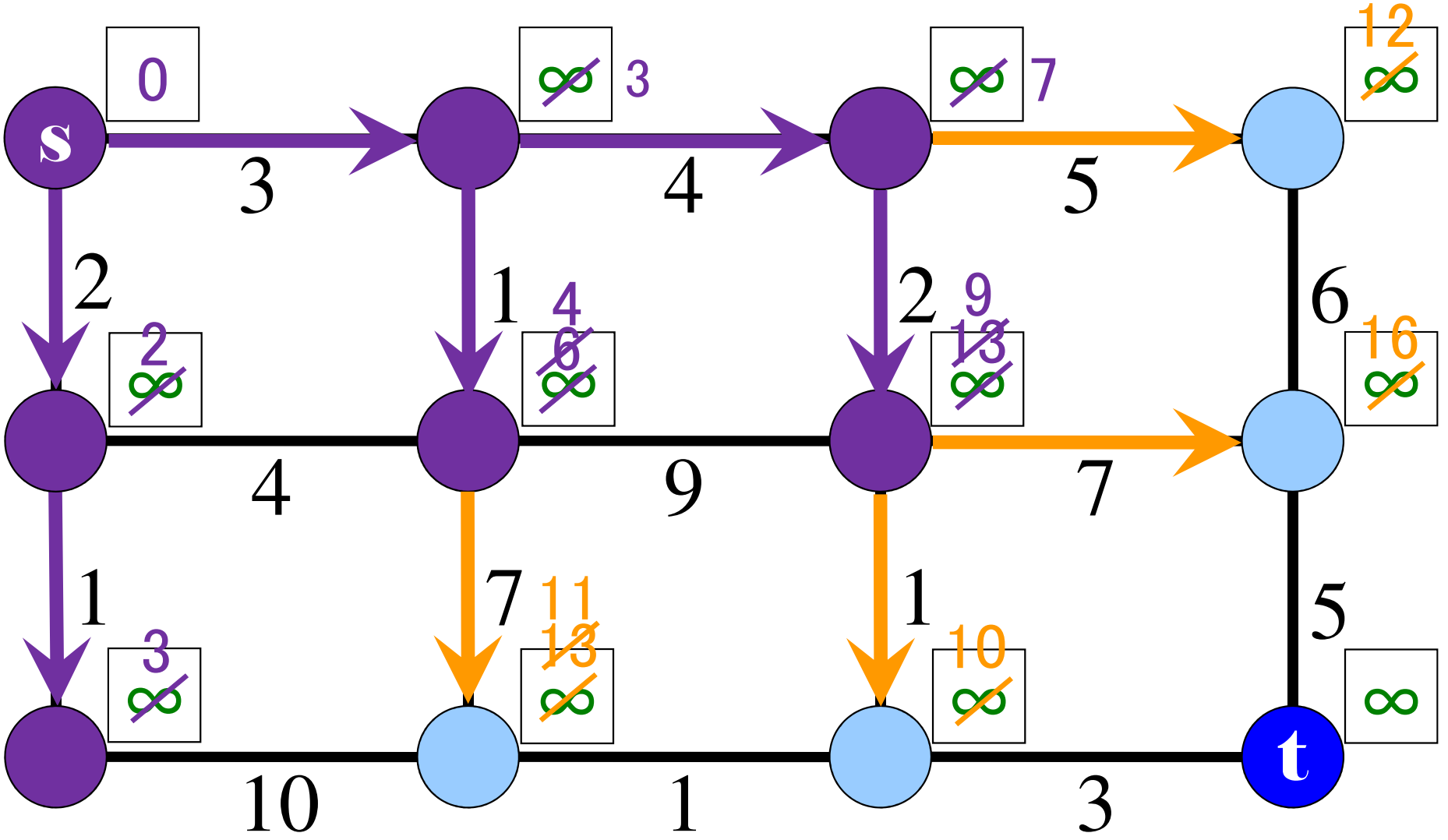


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

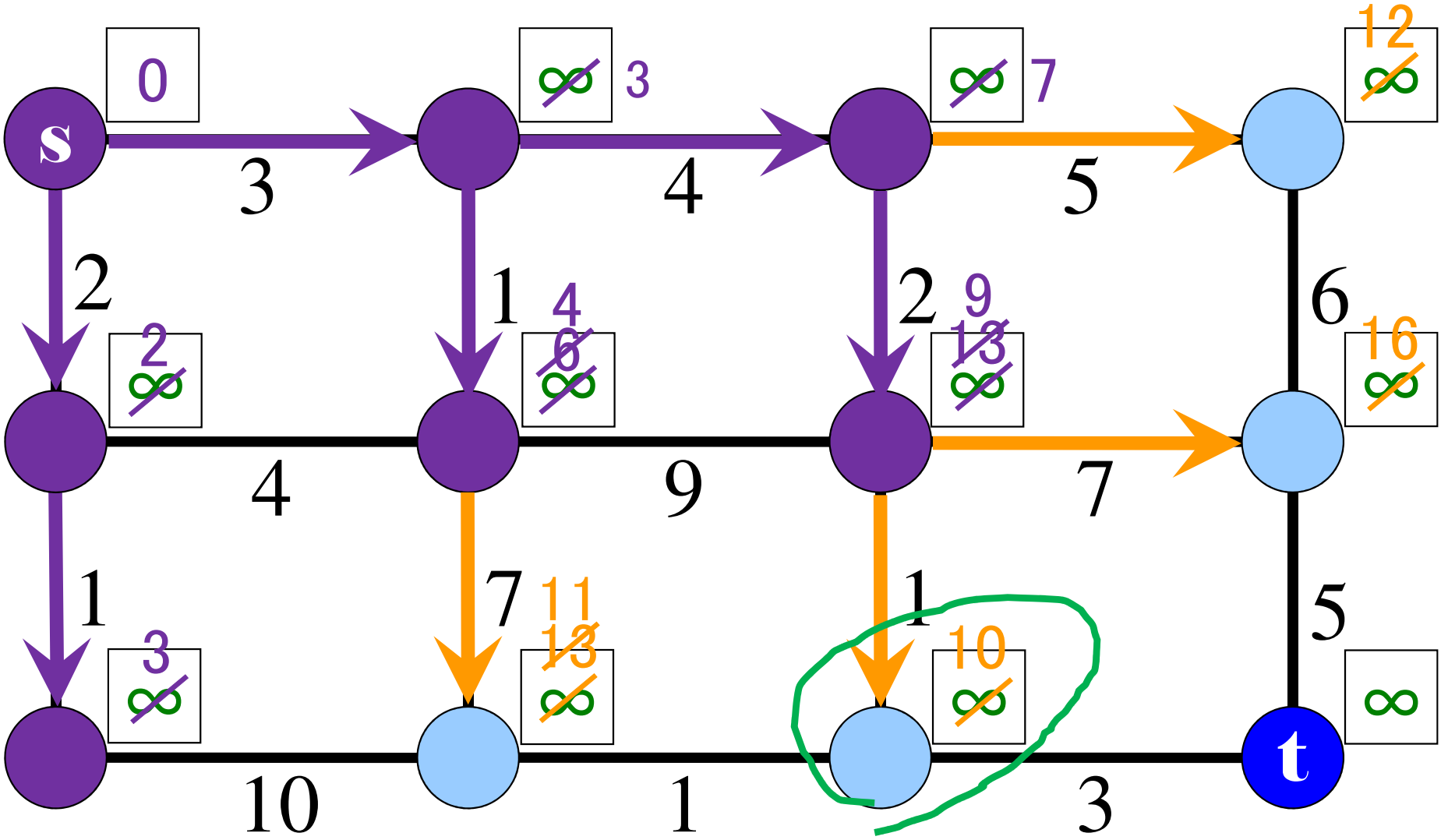
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



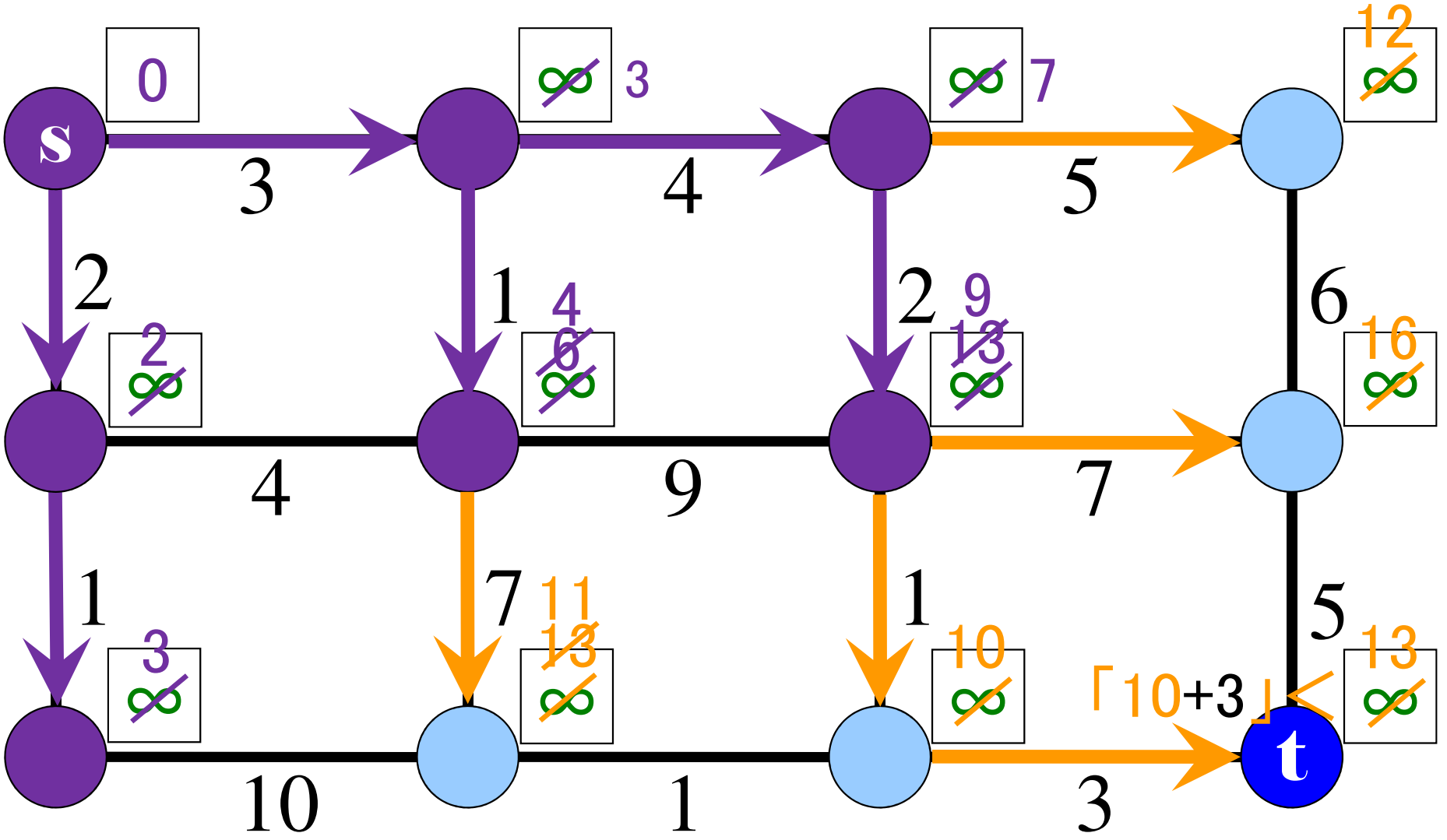
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

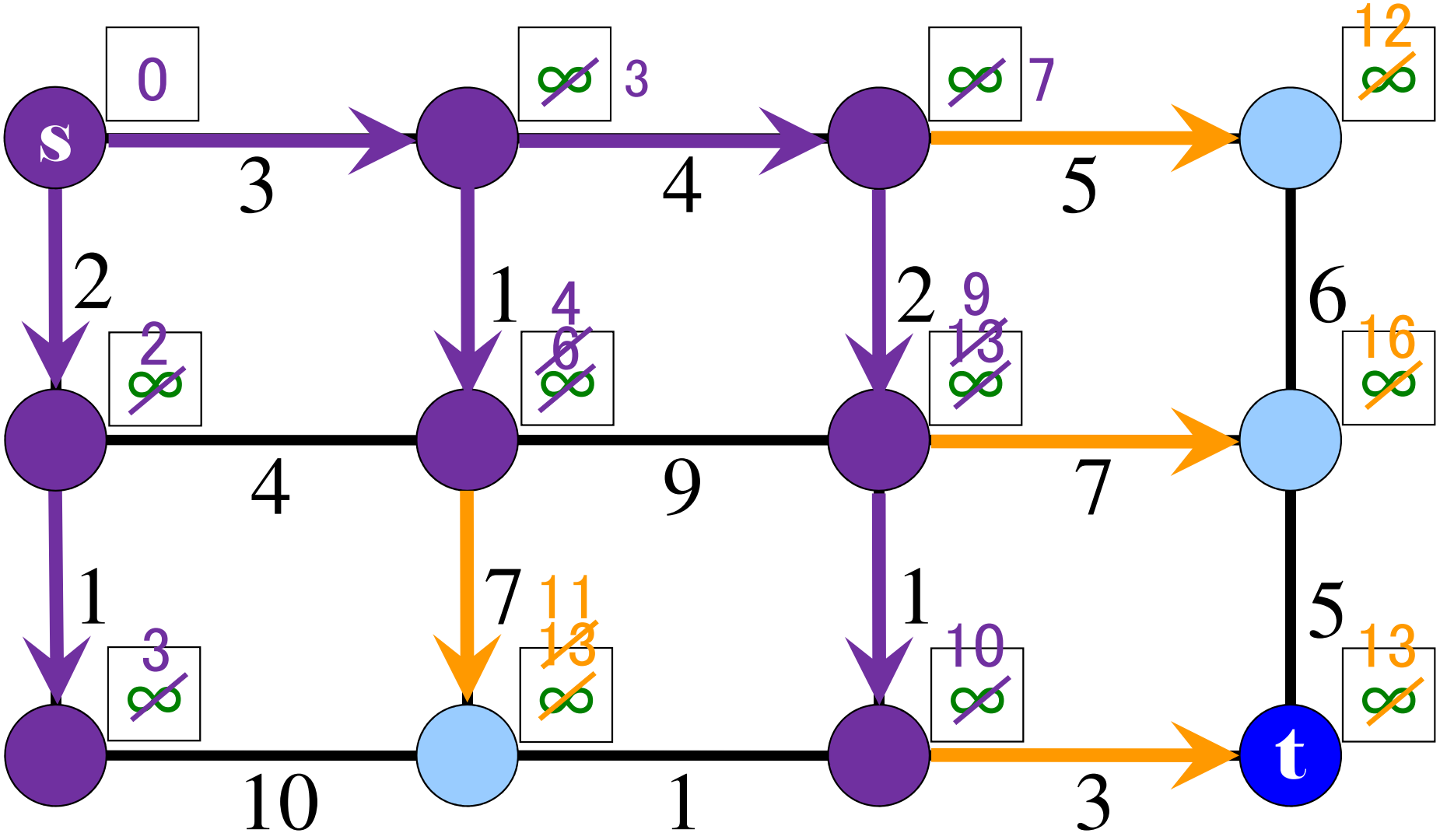


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

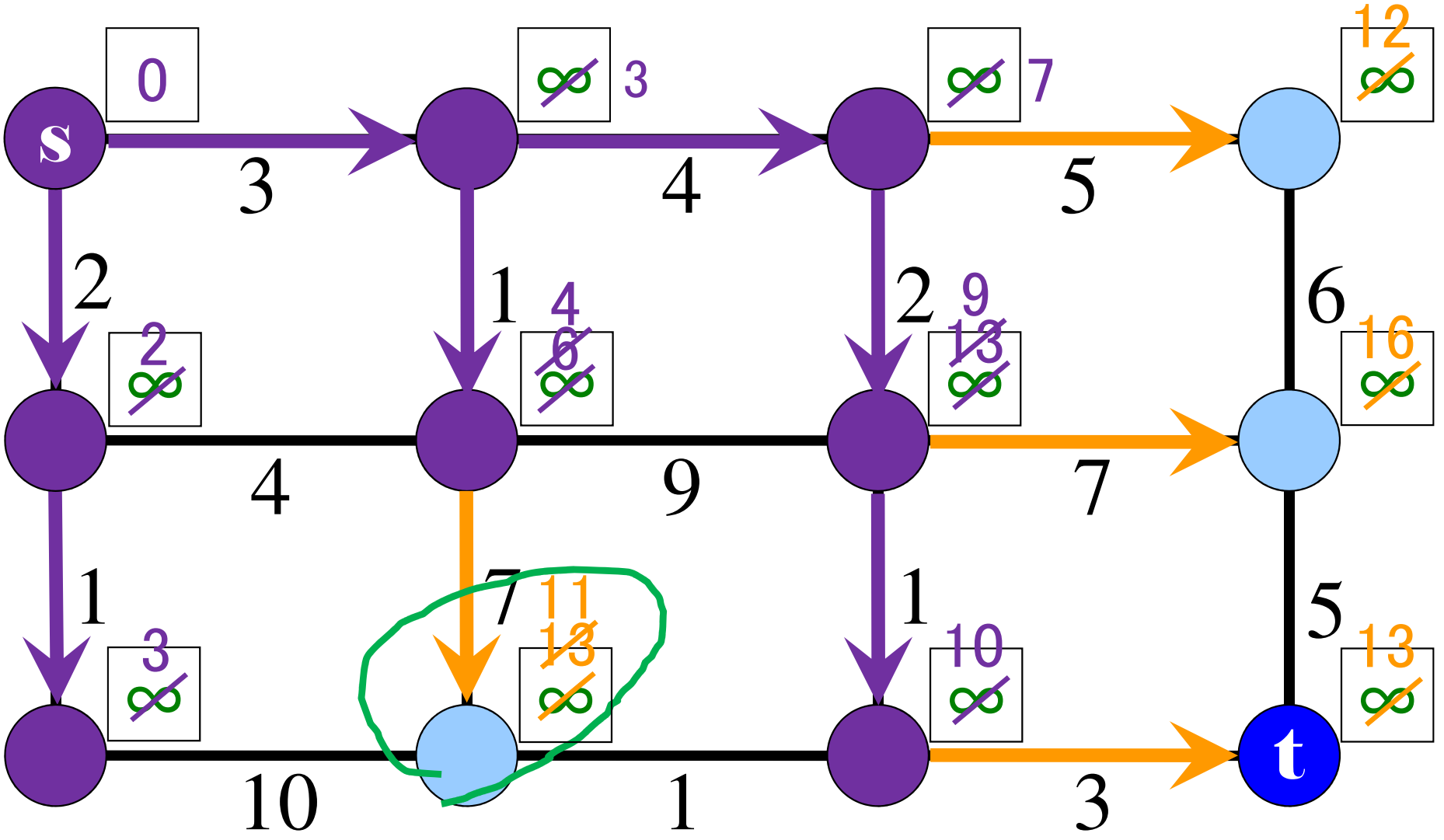
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



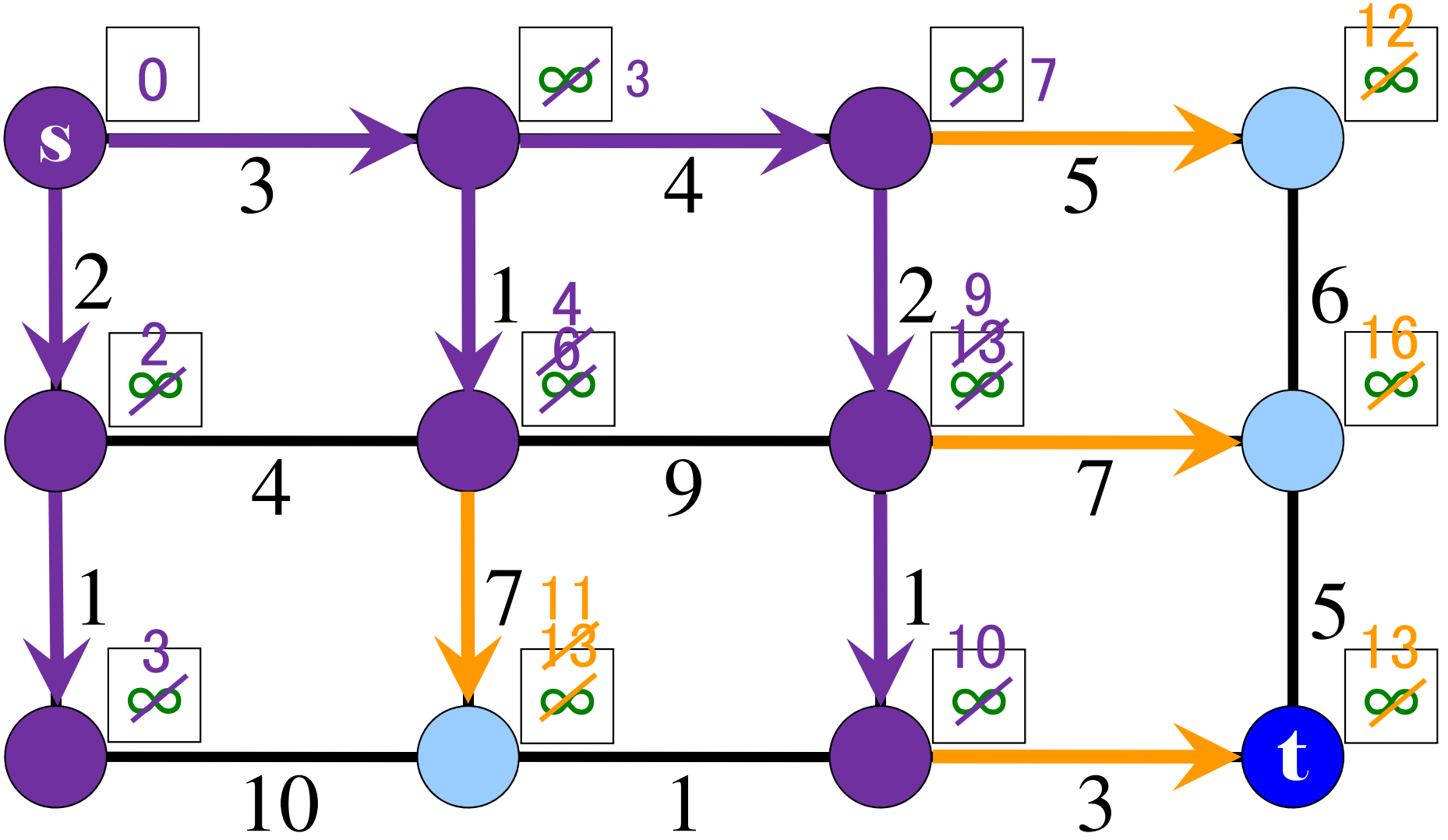
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

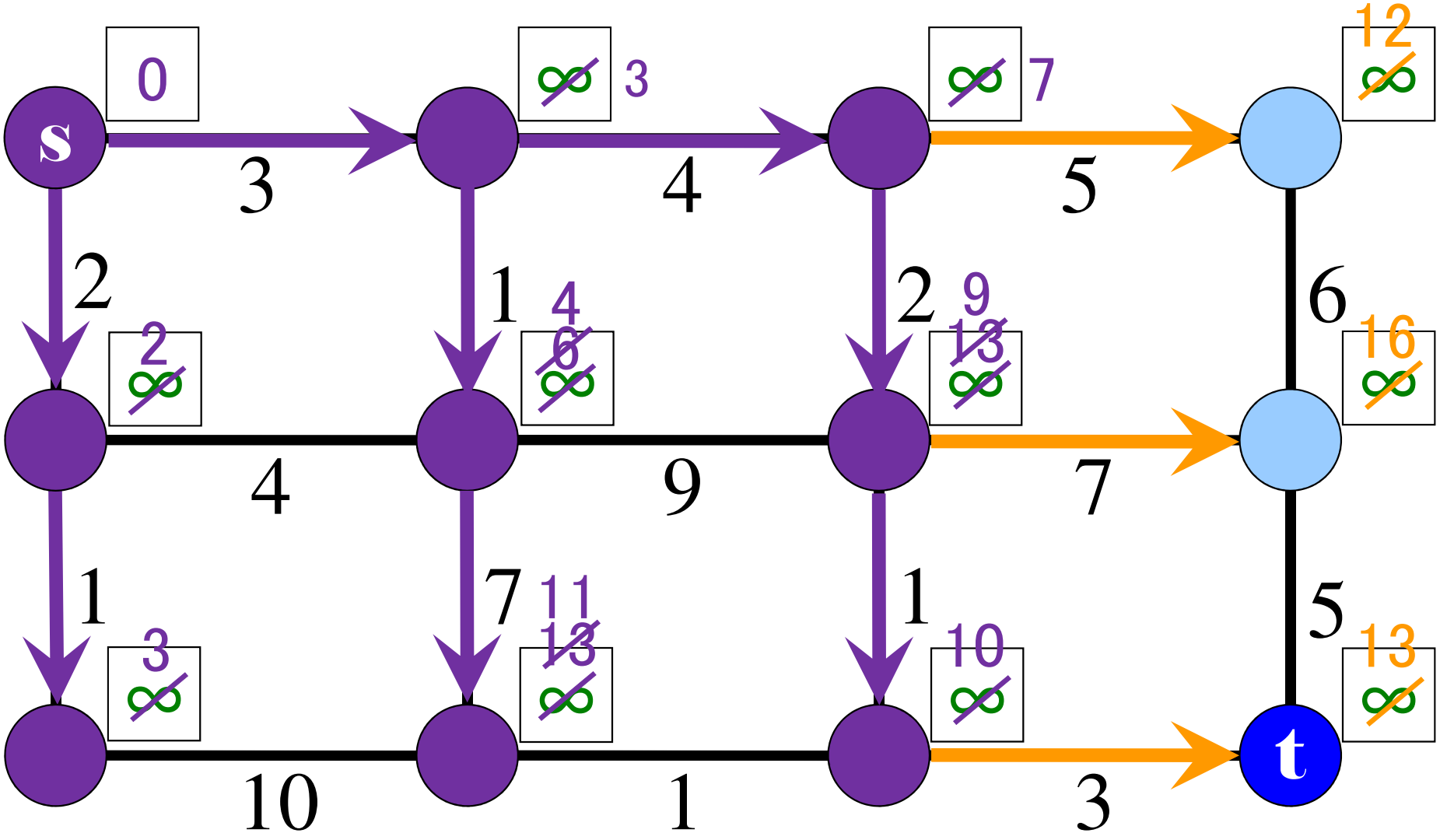


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

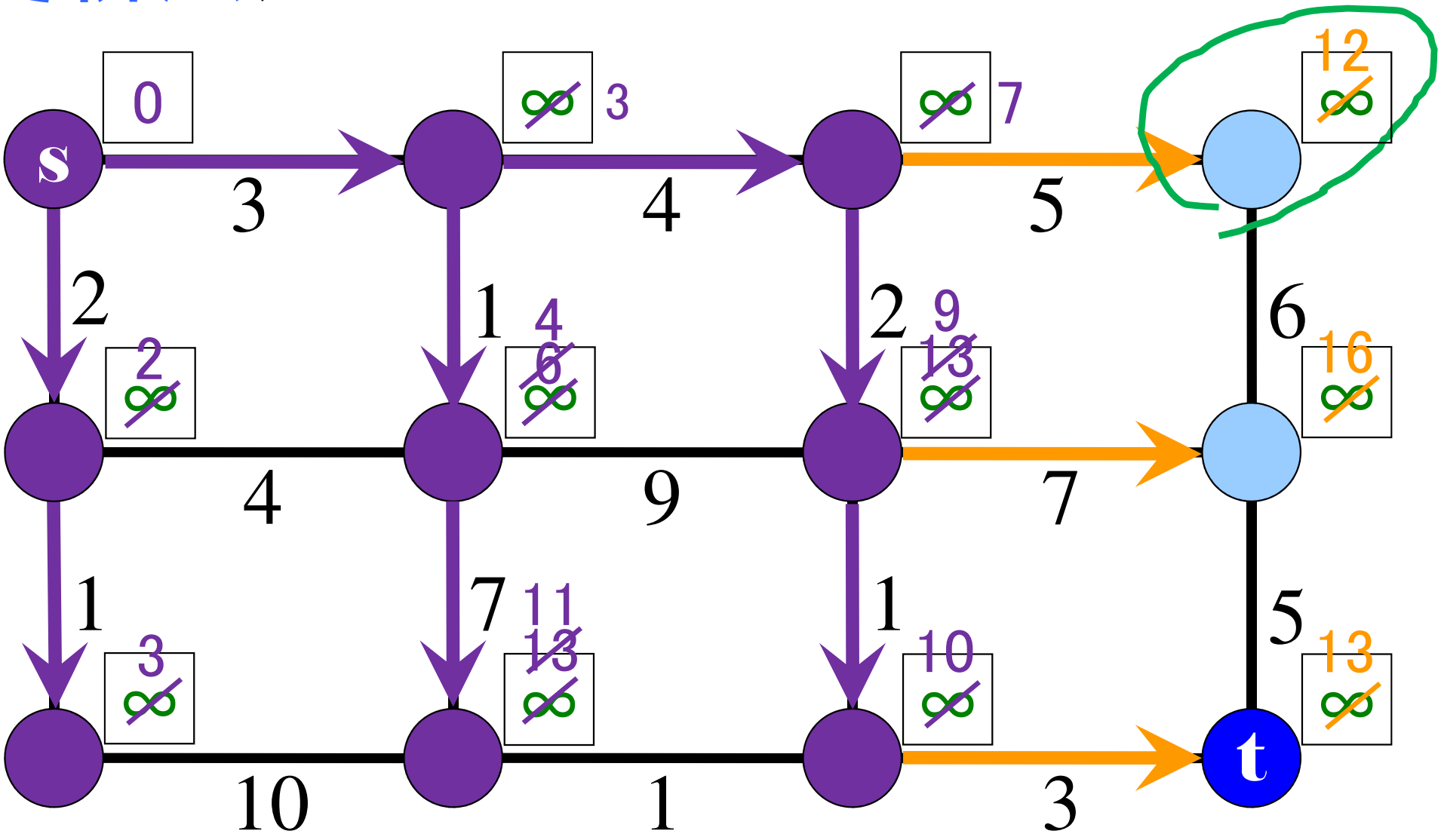
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



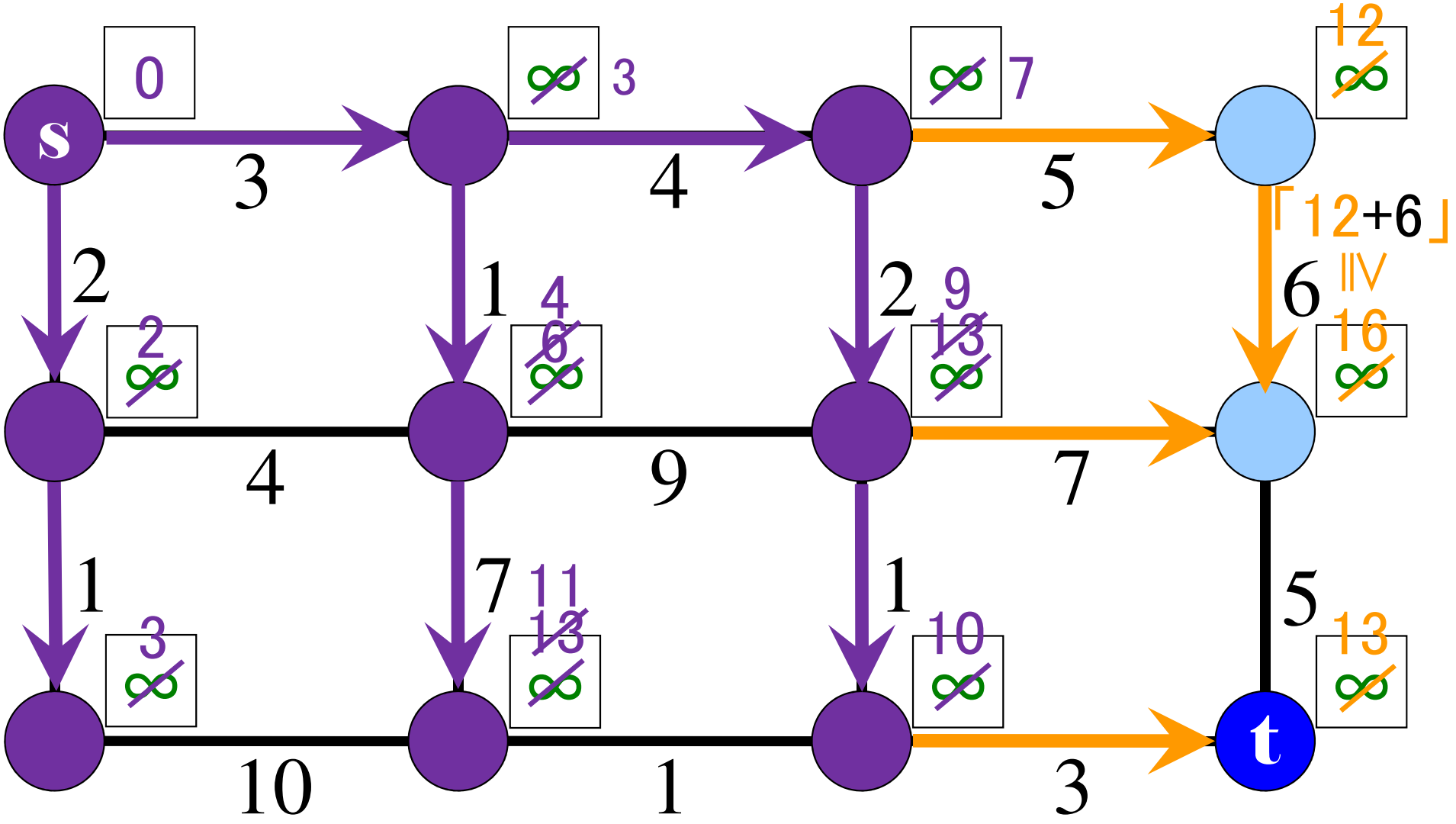
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

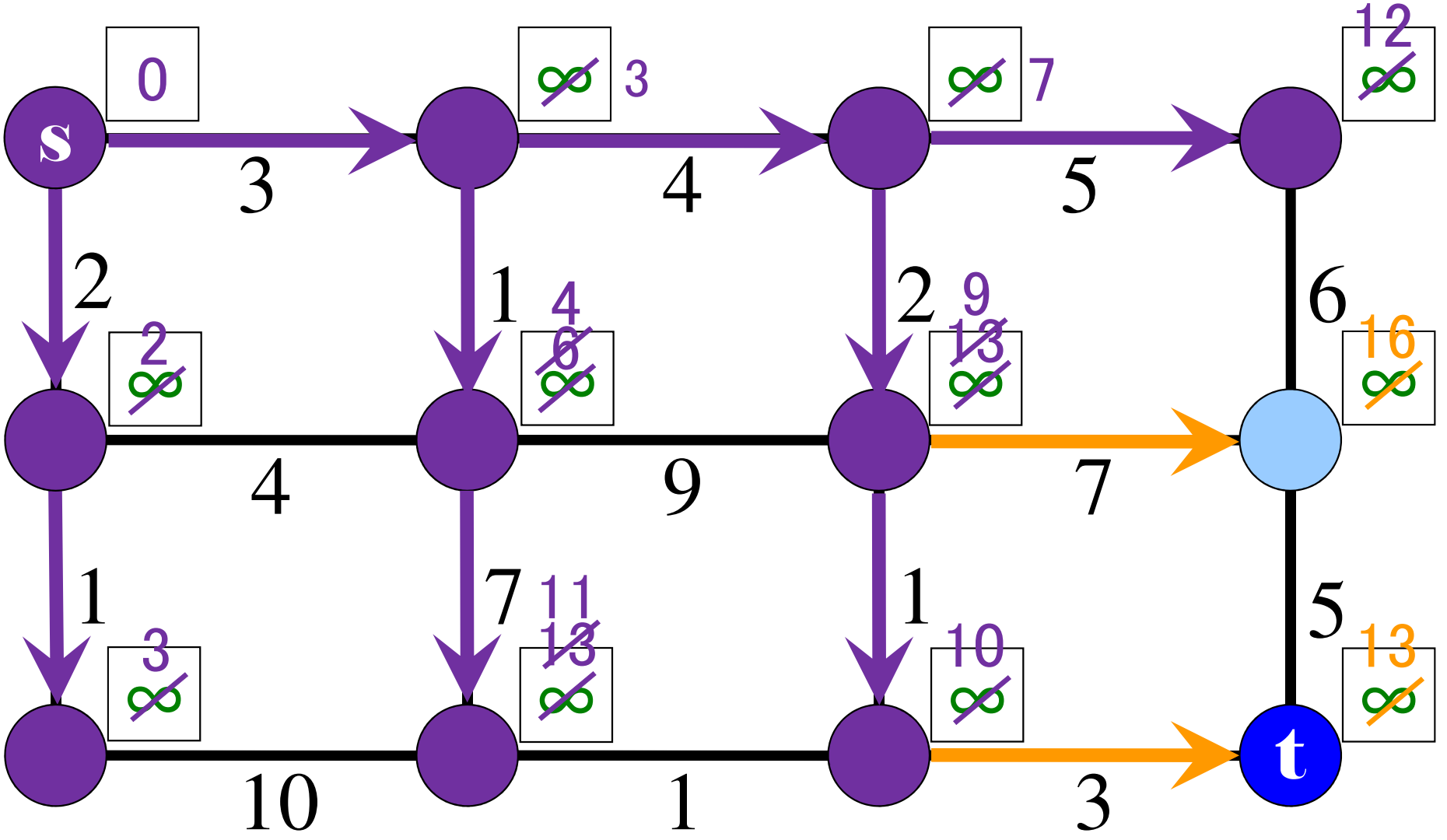


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

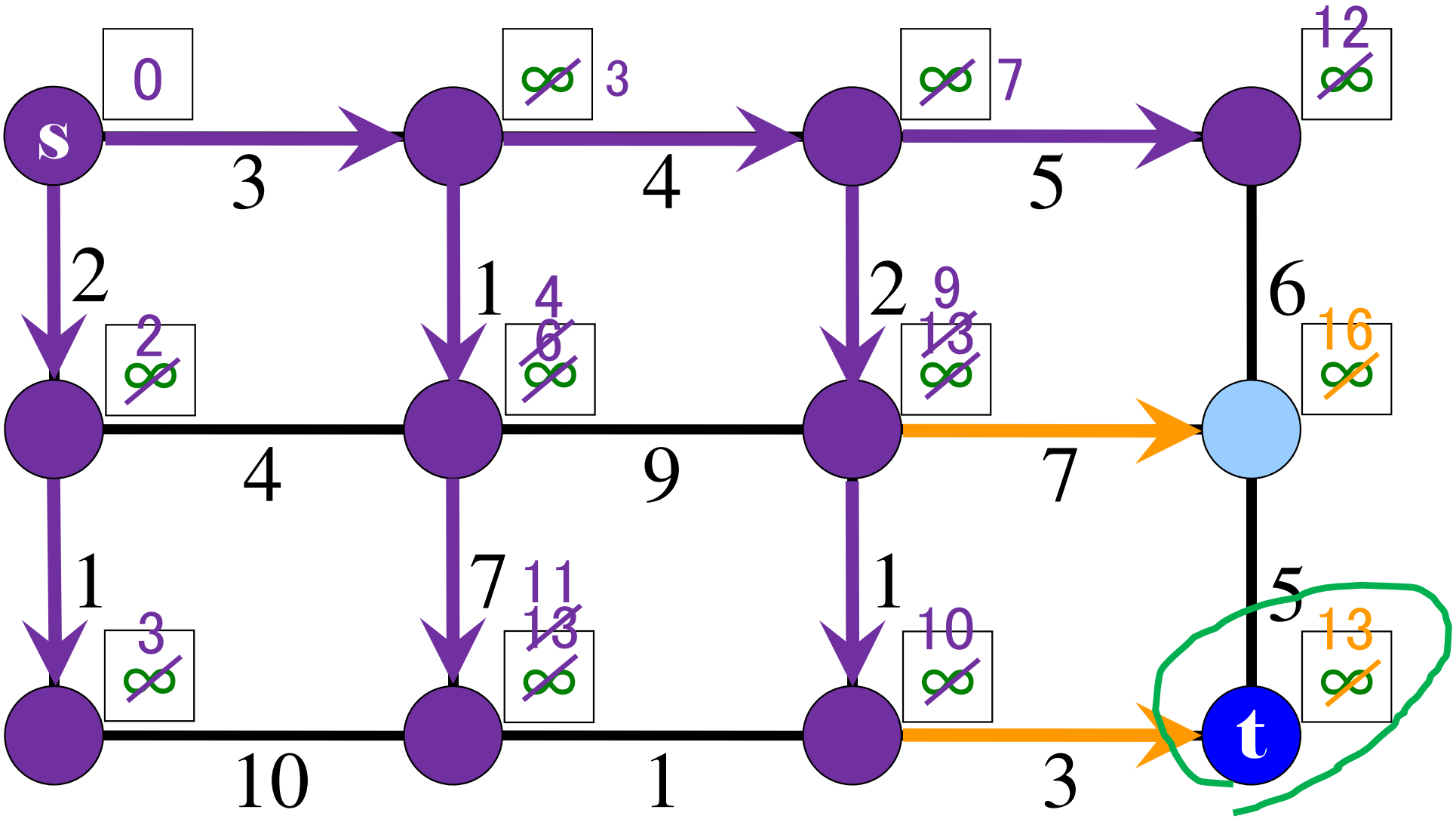
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



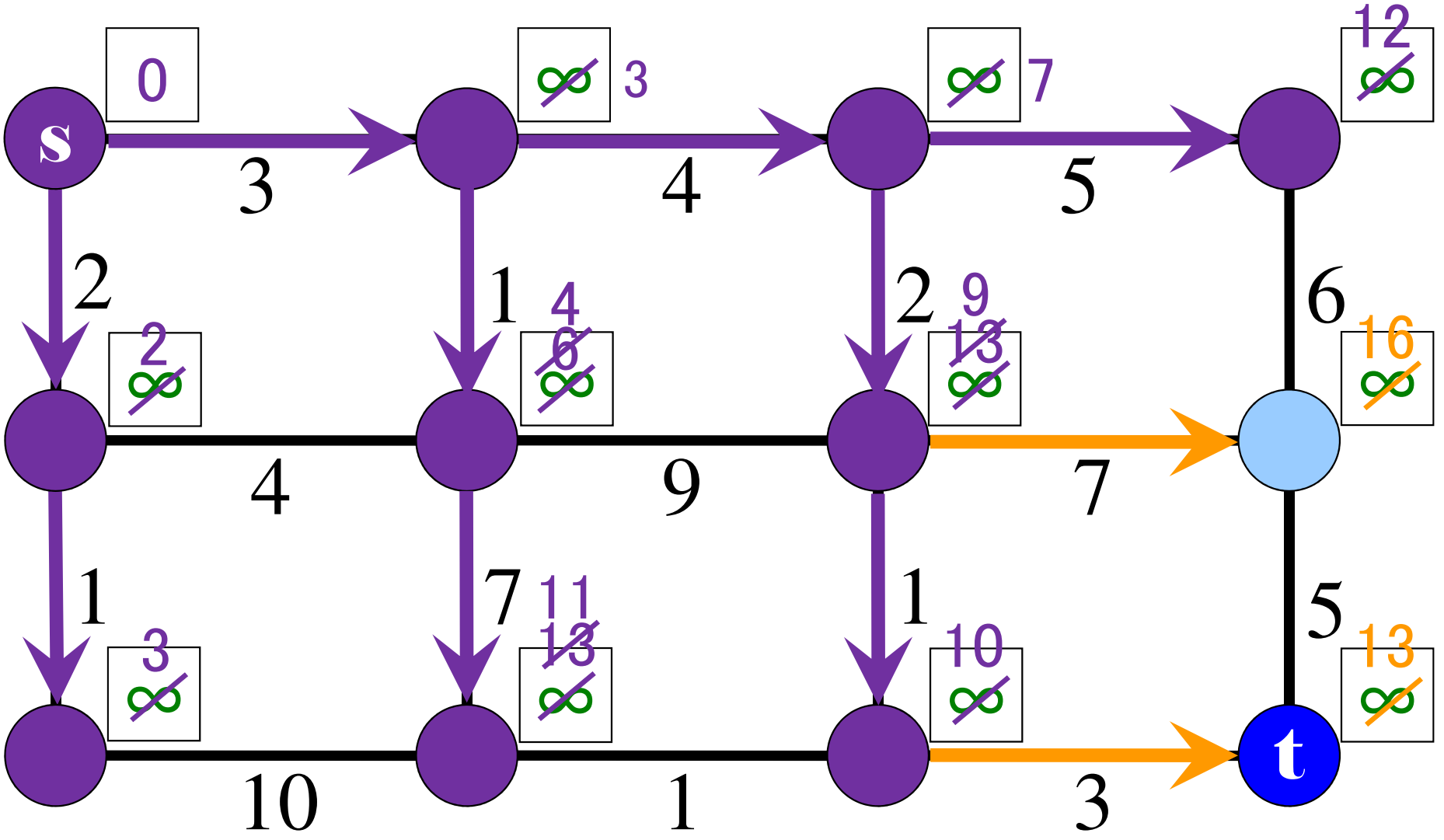
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

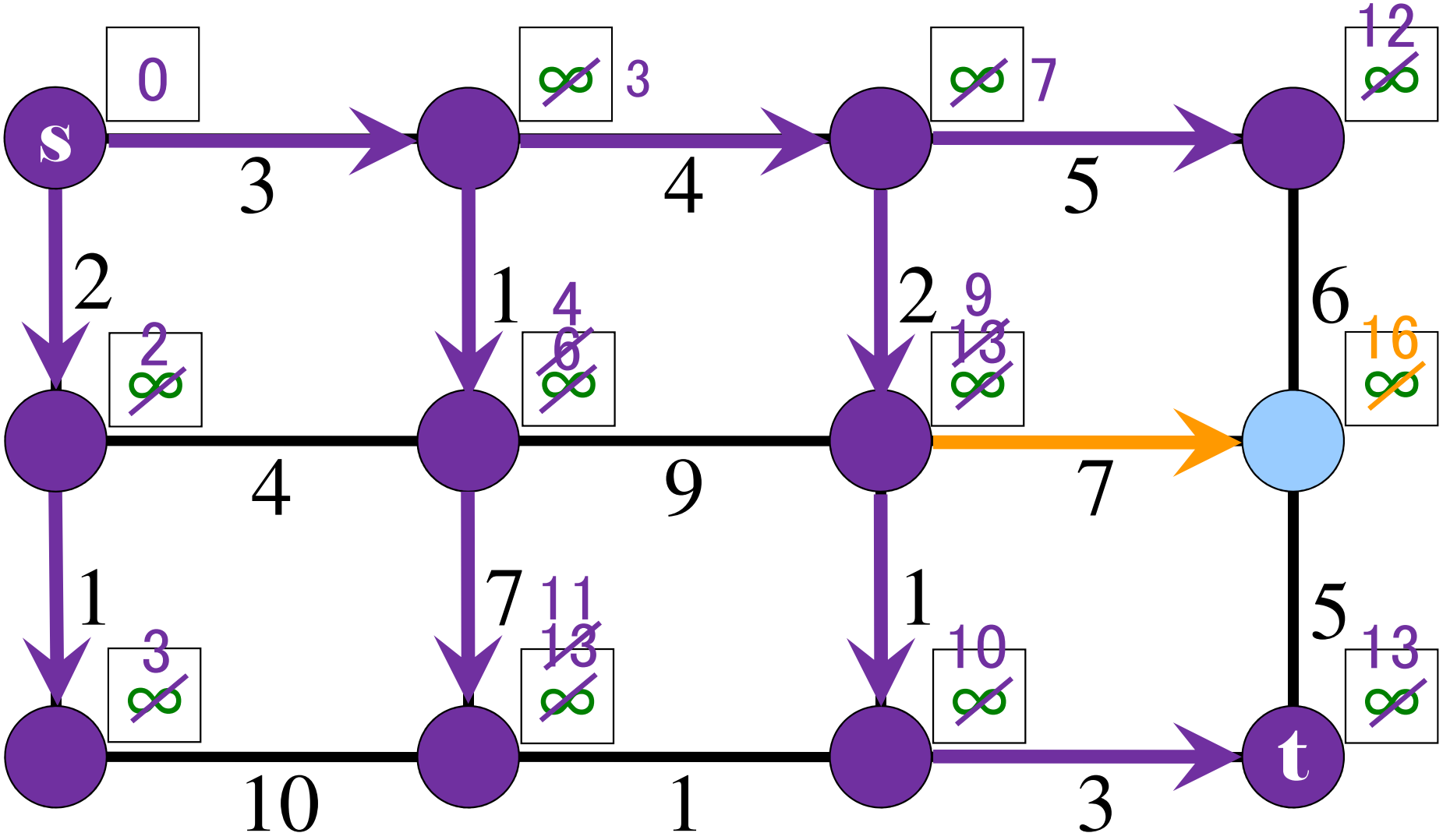


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

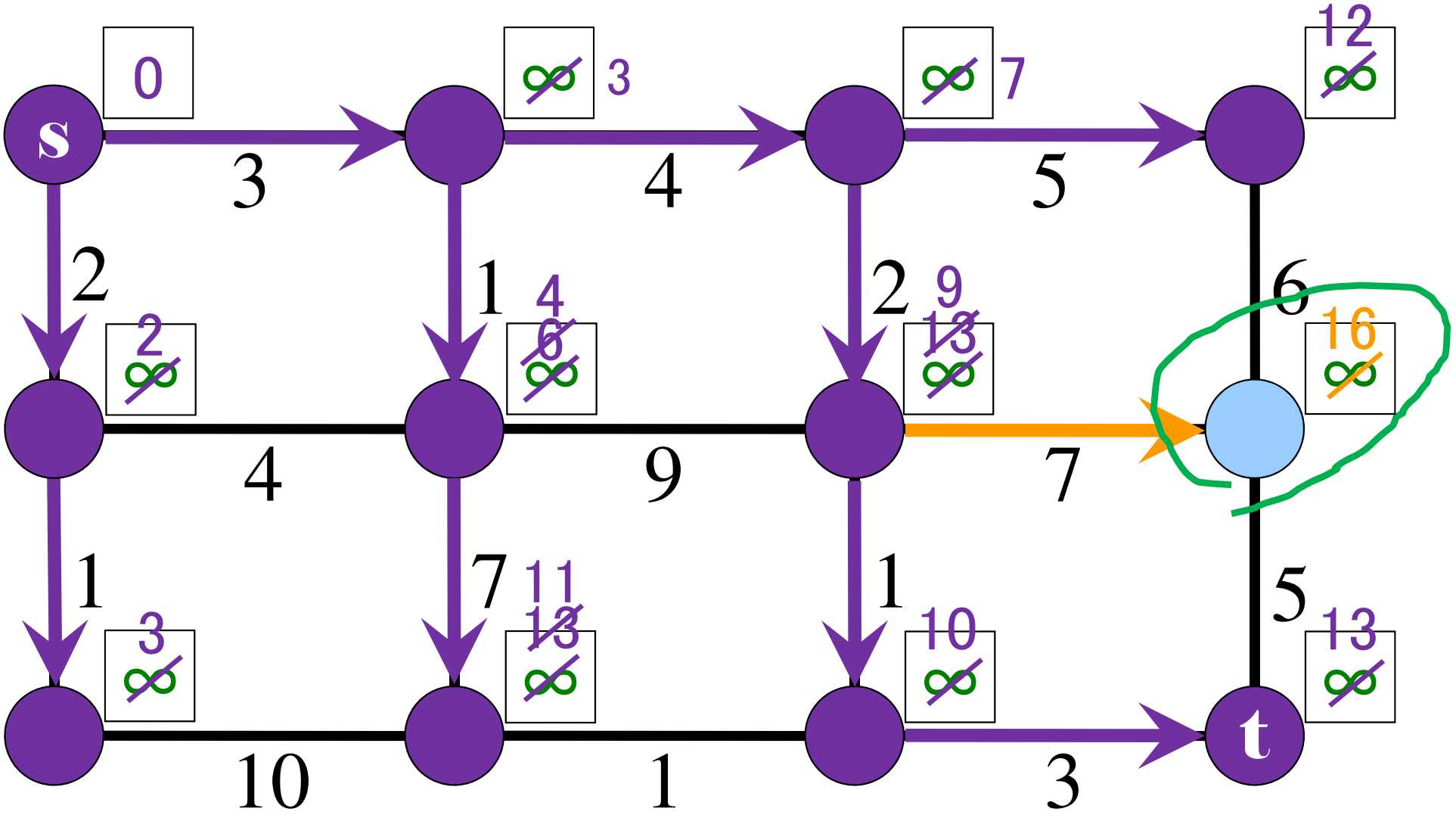
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



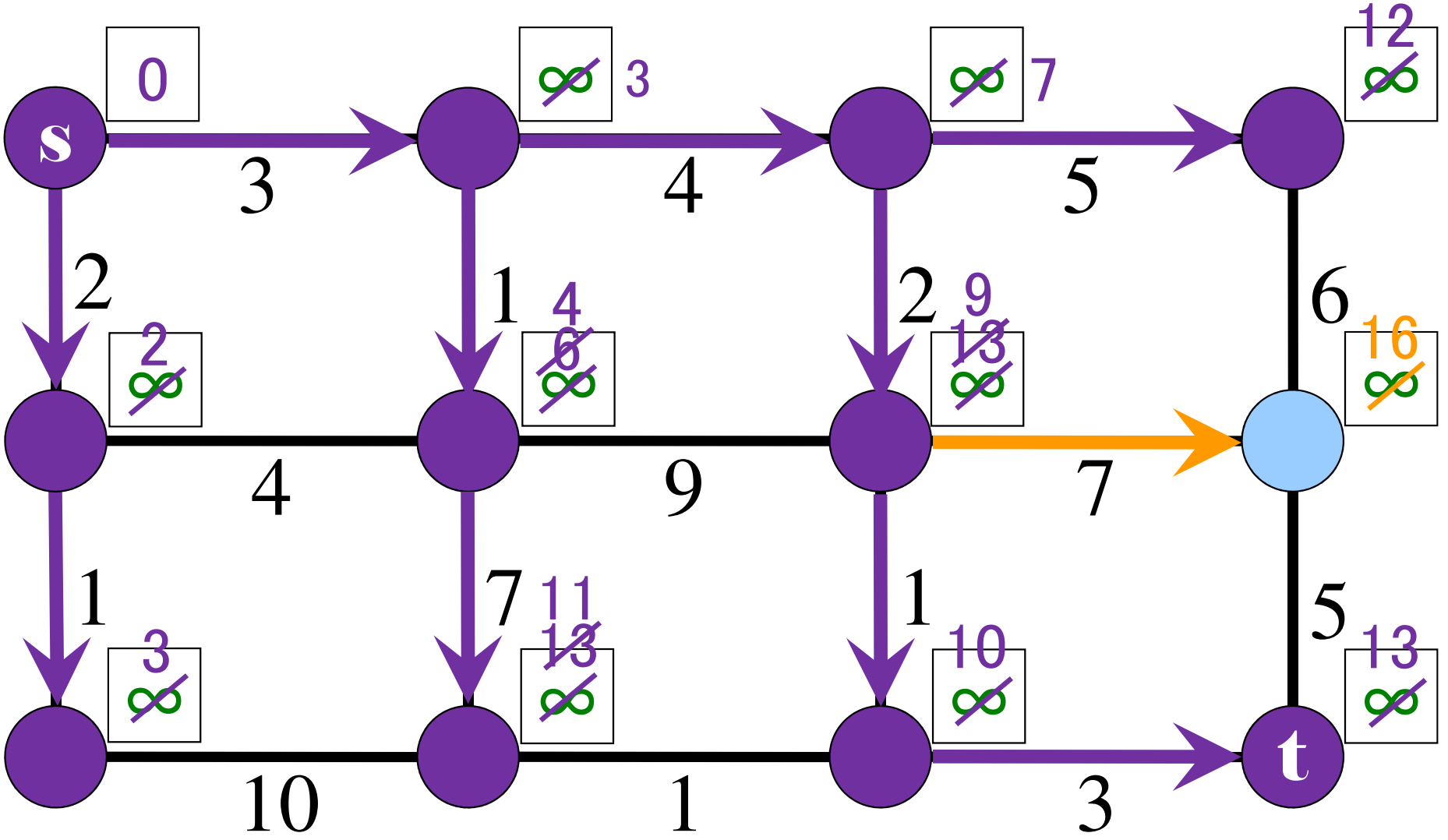
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

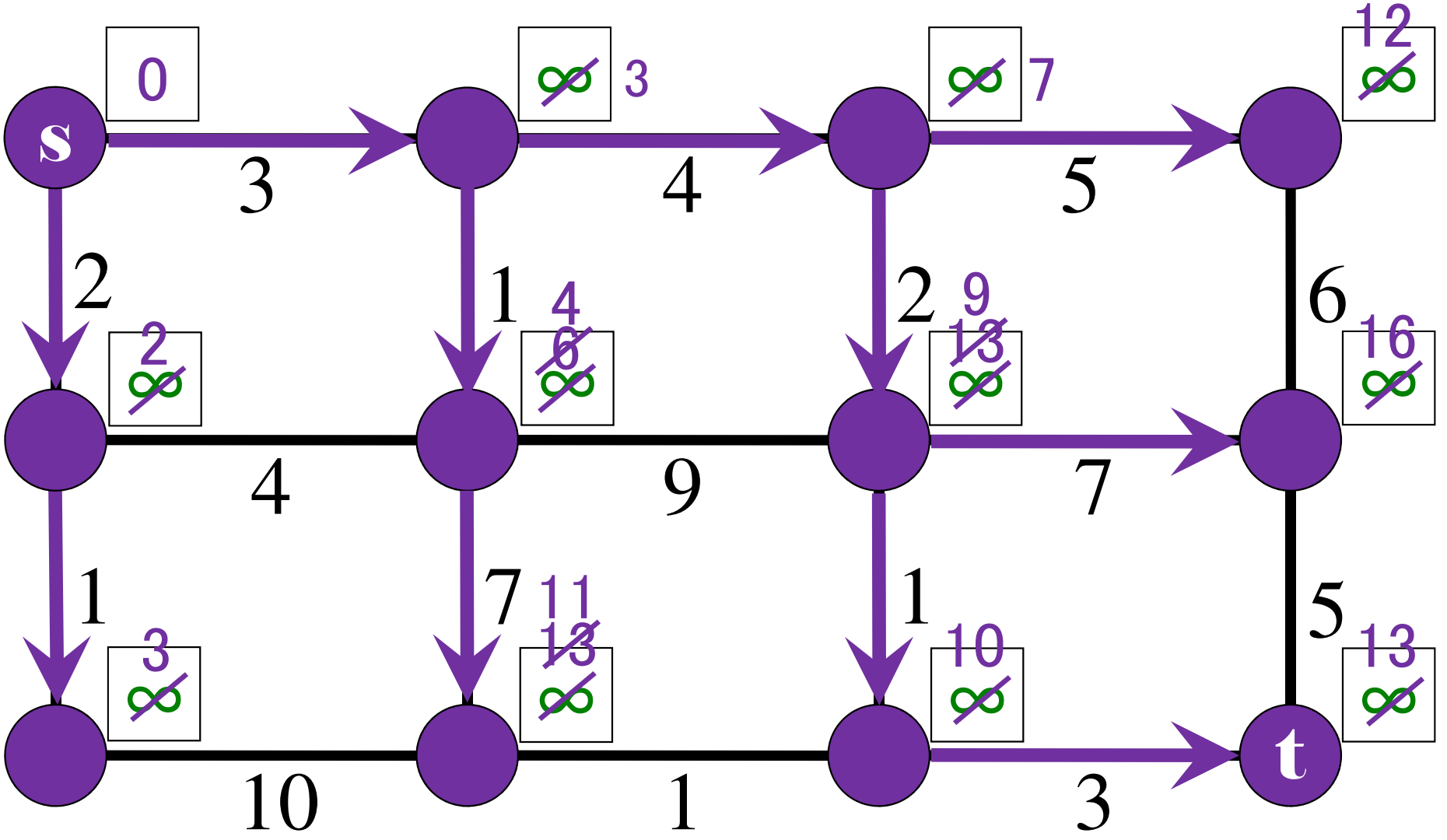


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

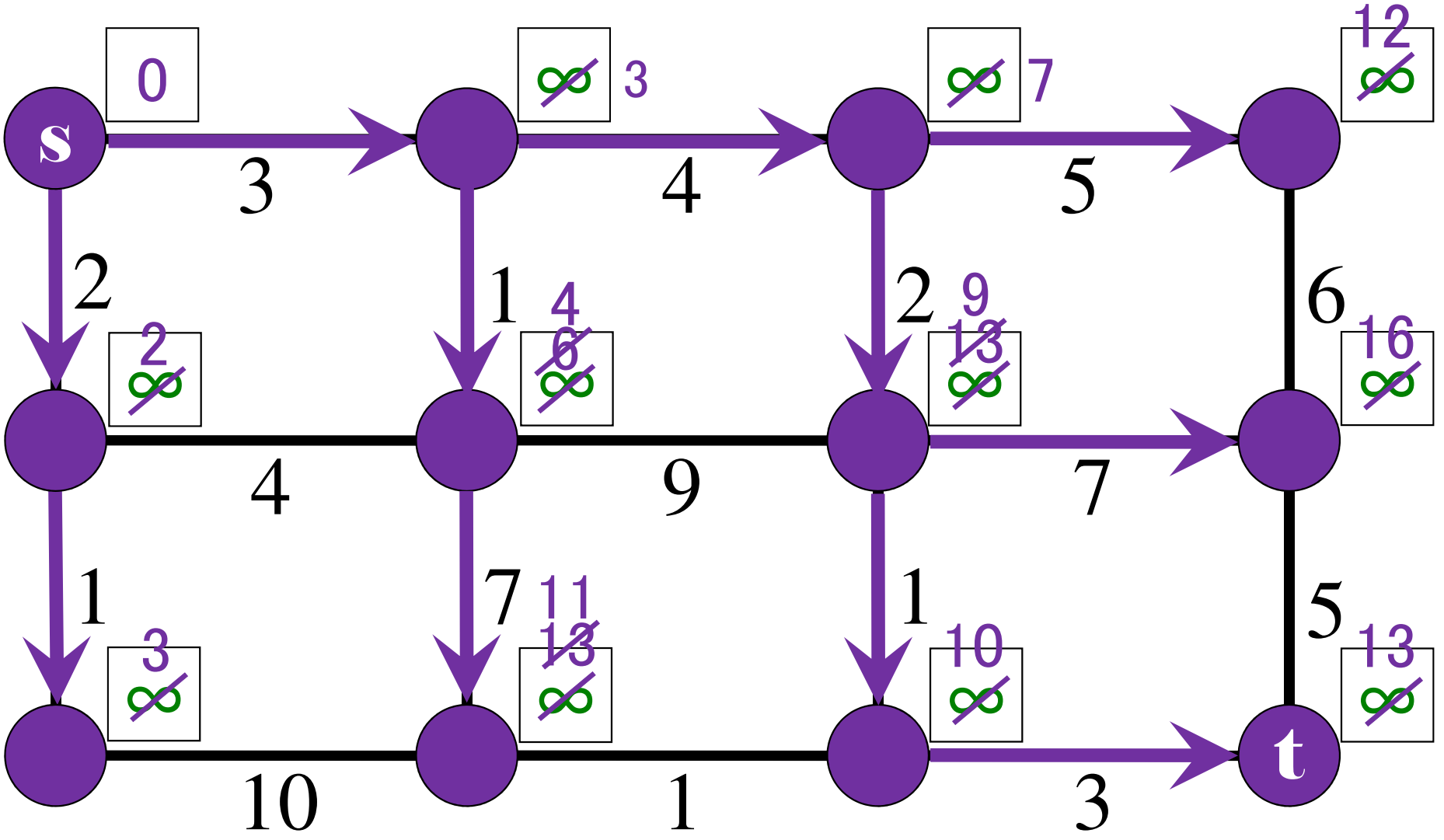
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



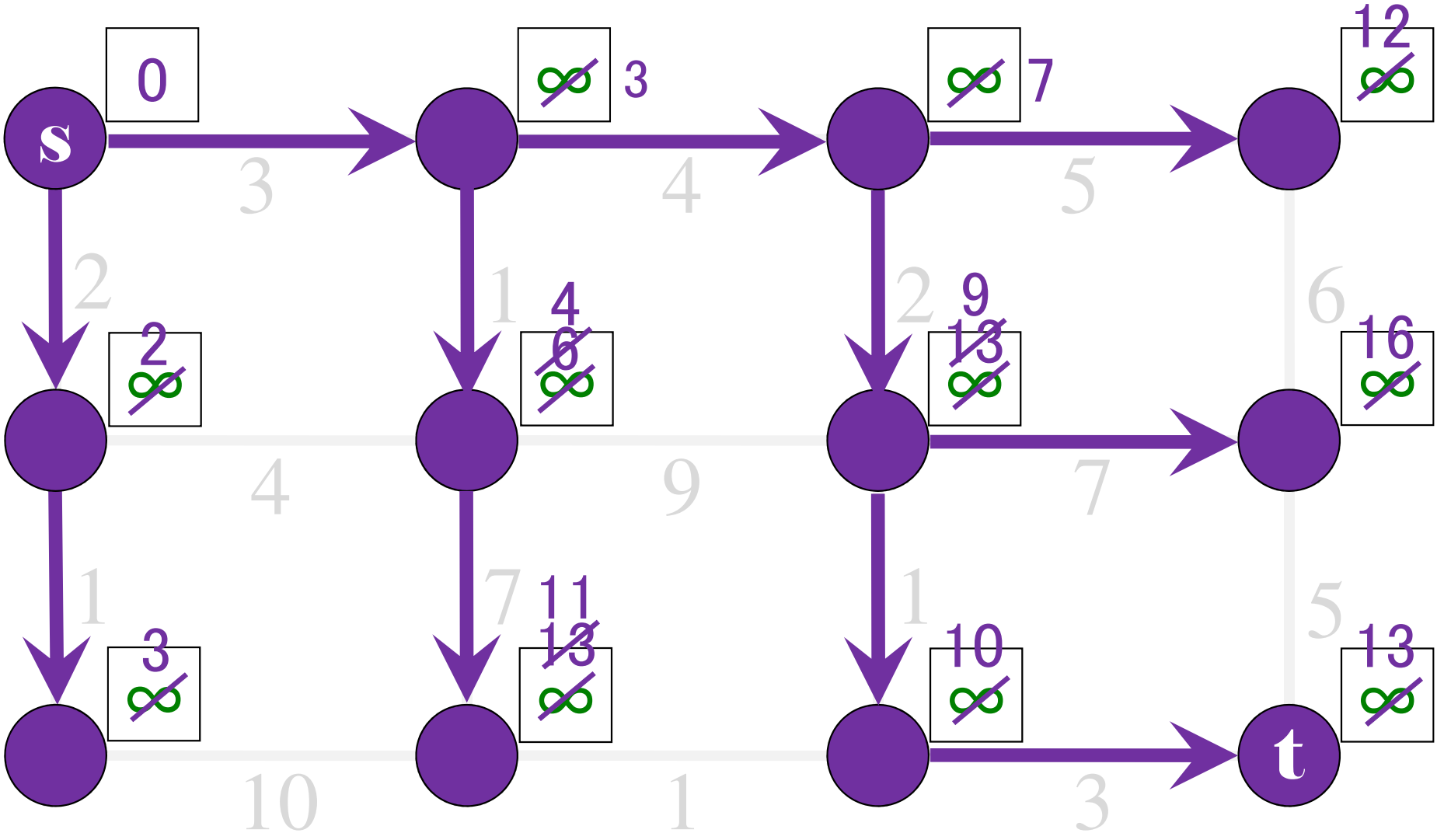
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



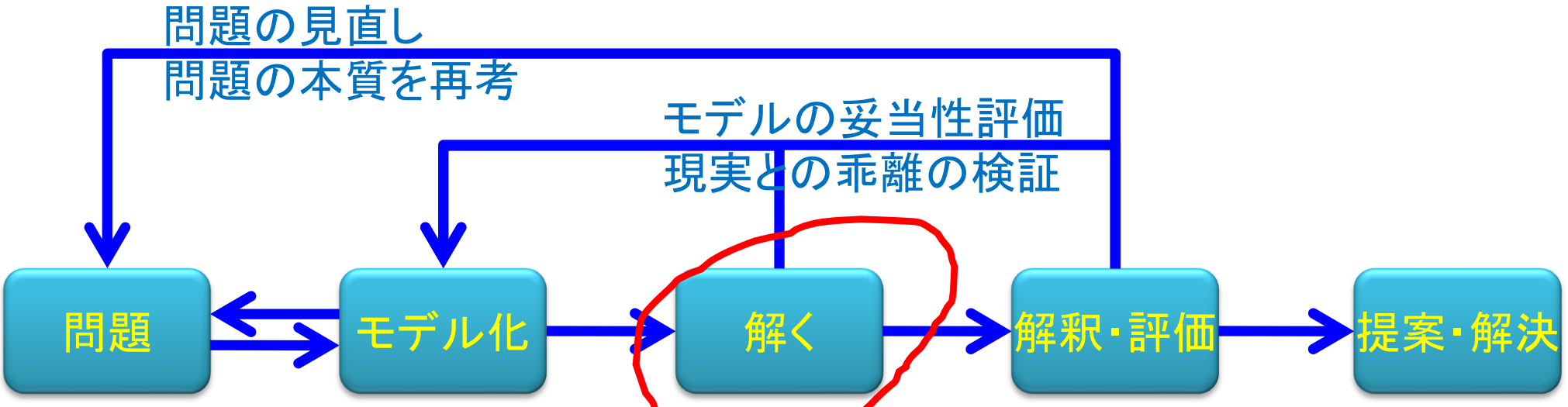
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

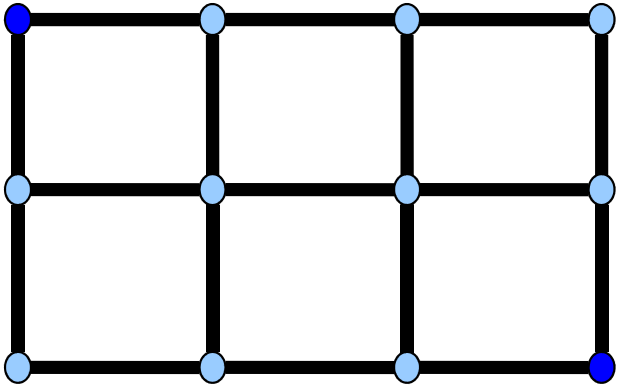
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

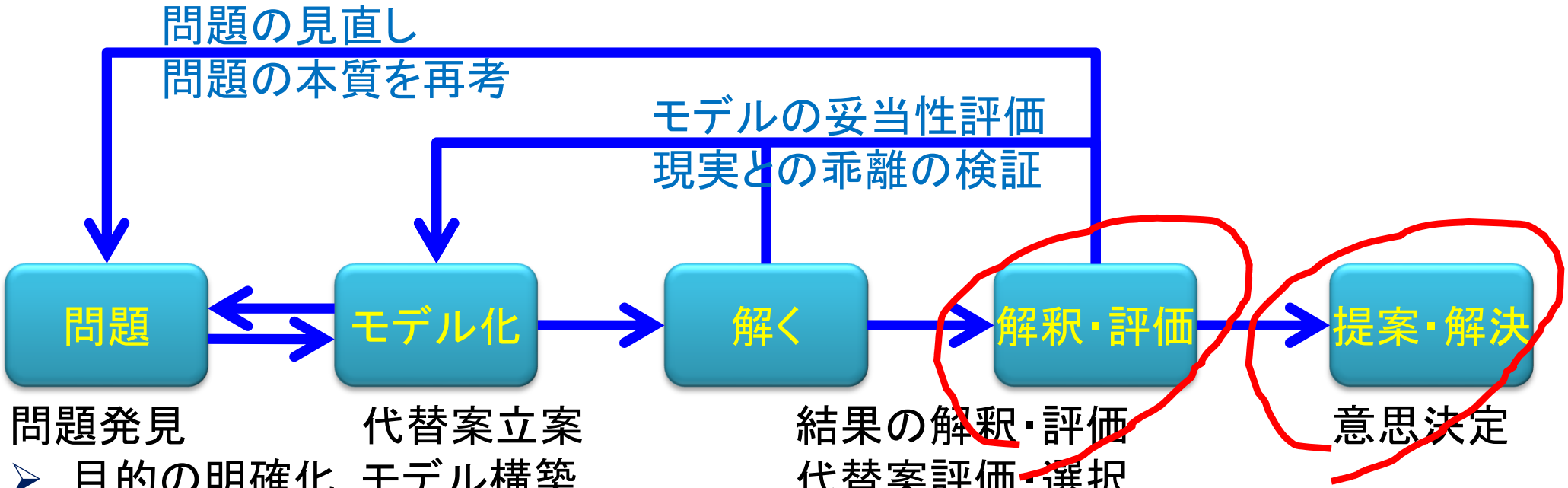
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

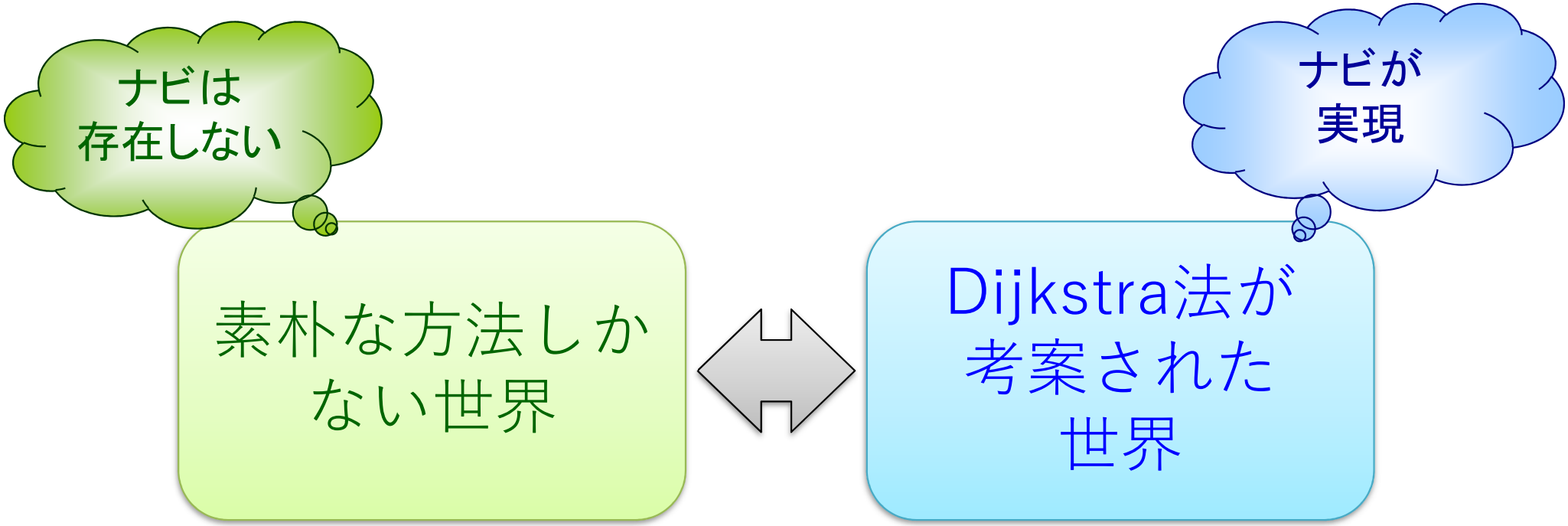
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

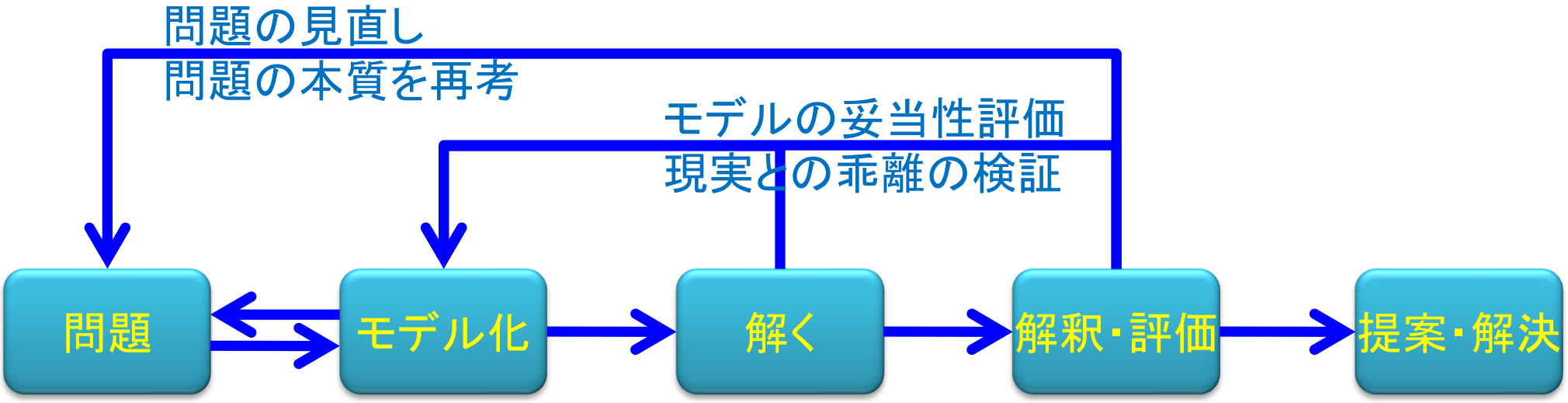
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



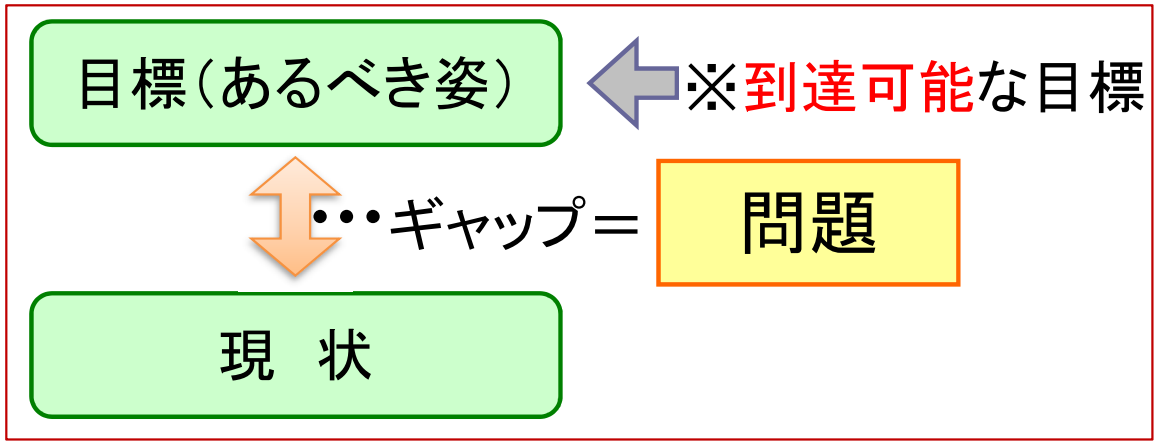
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

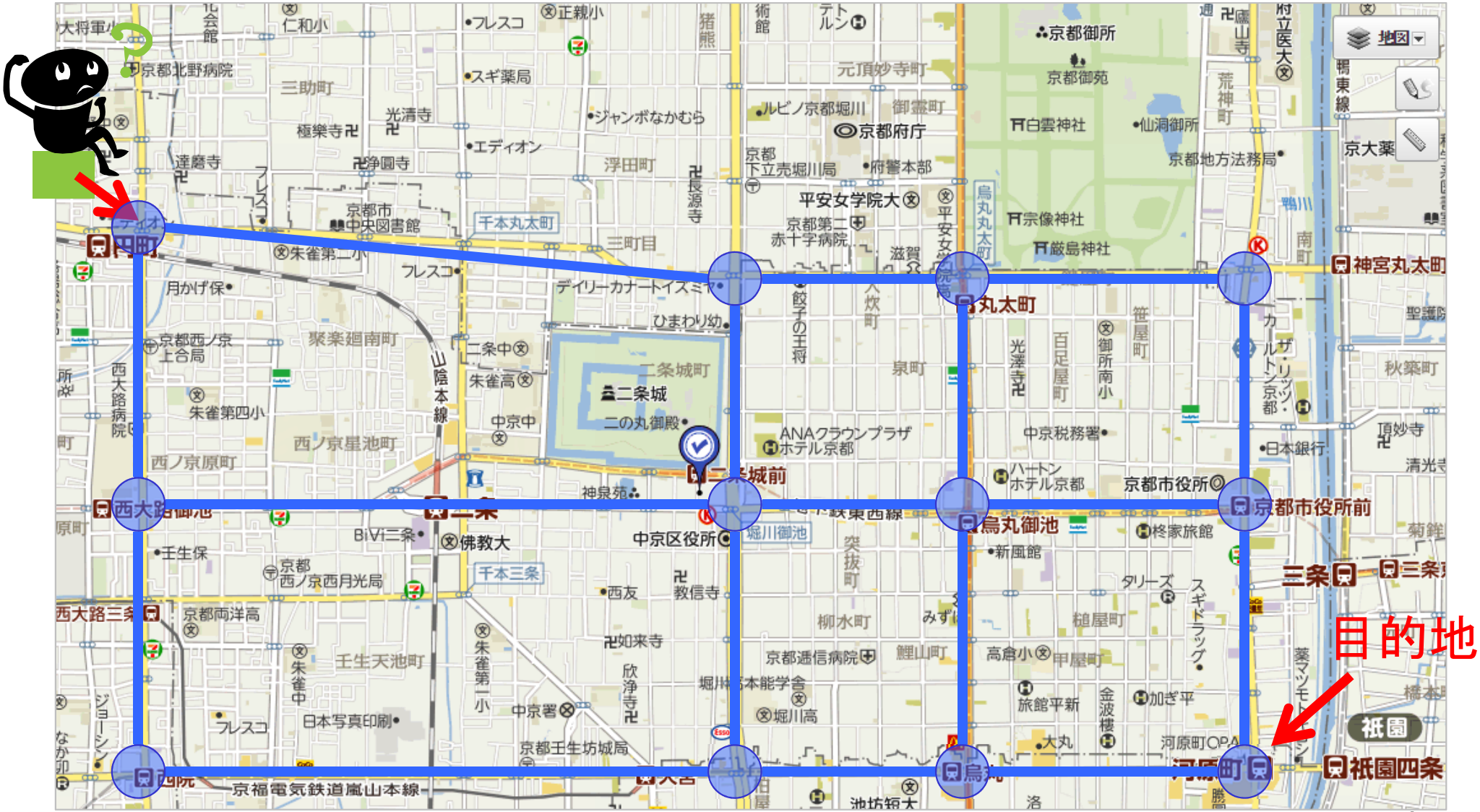
意思決定

問題の定義



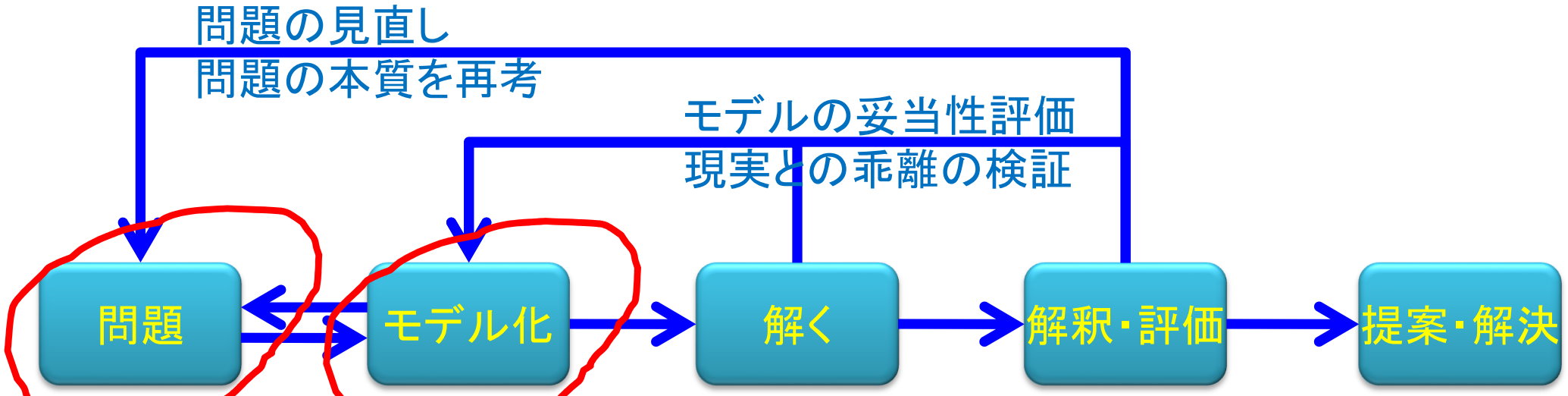
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

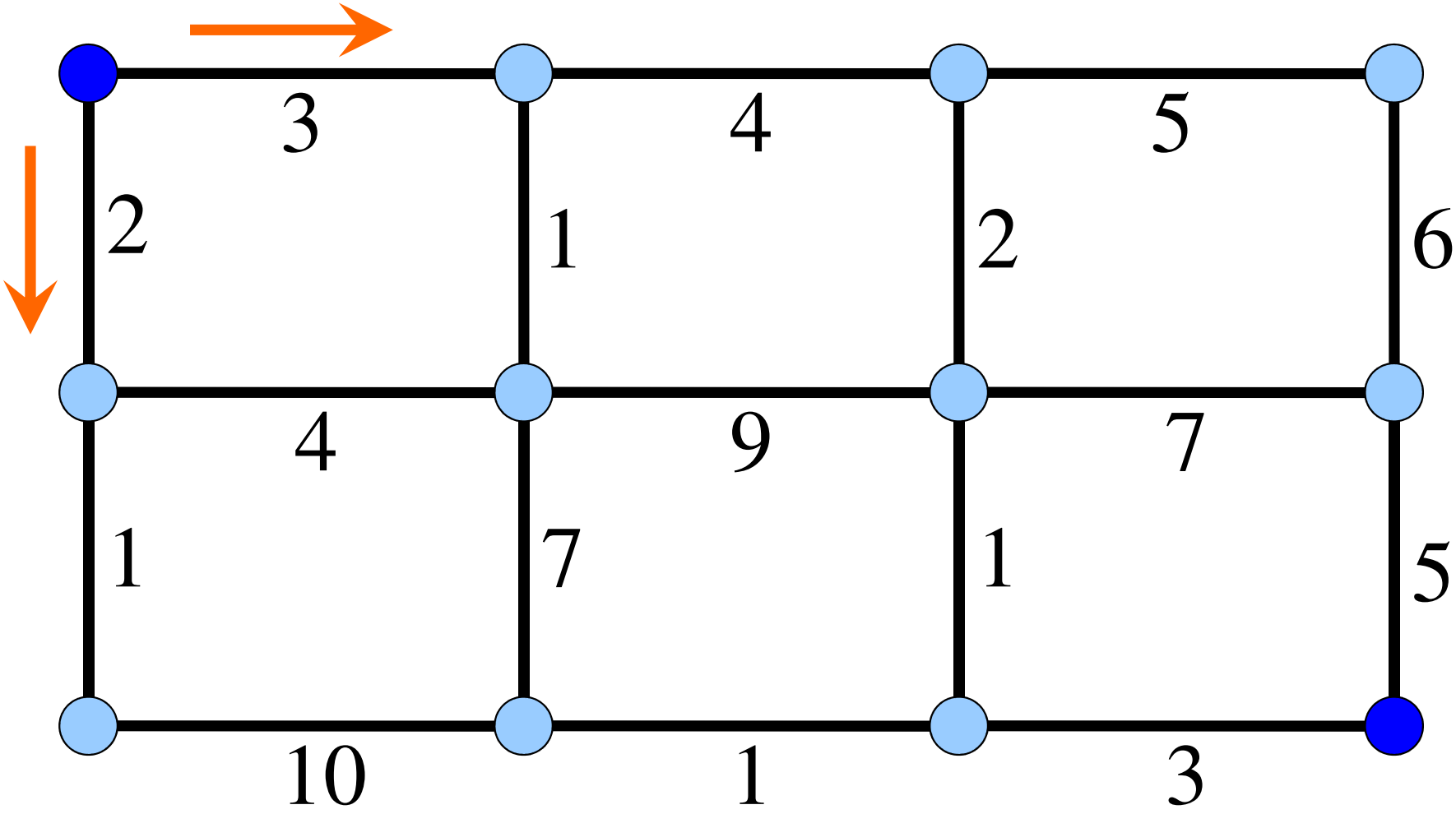
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

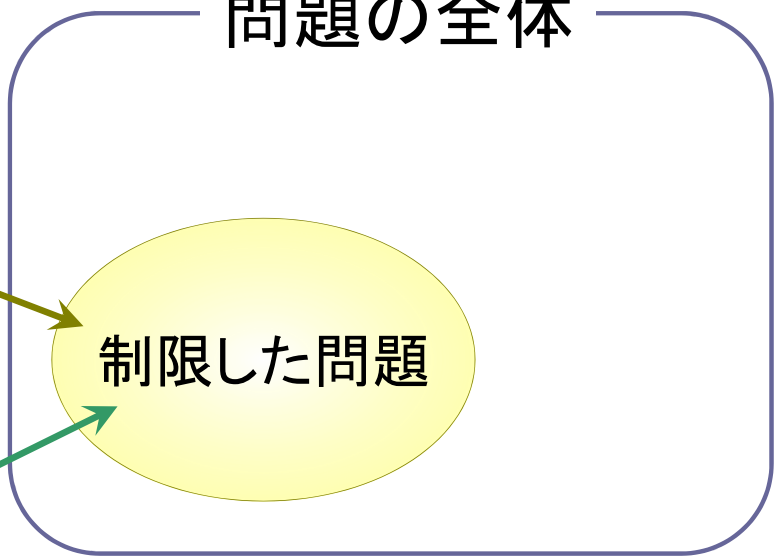


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

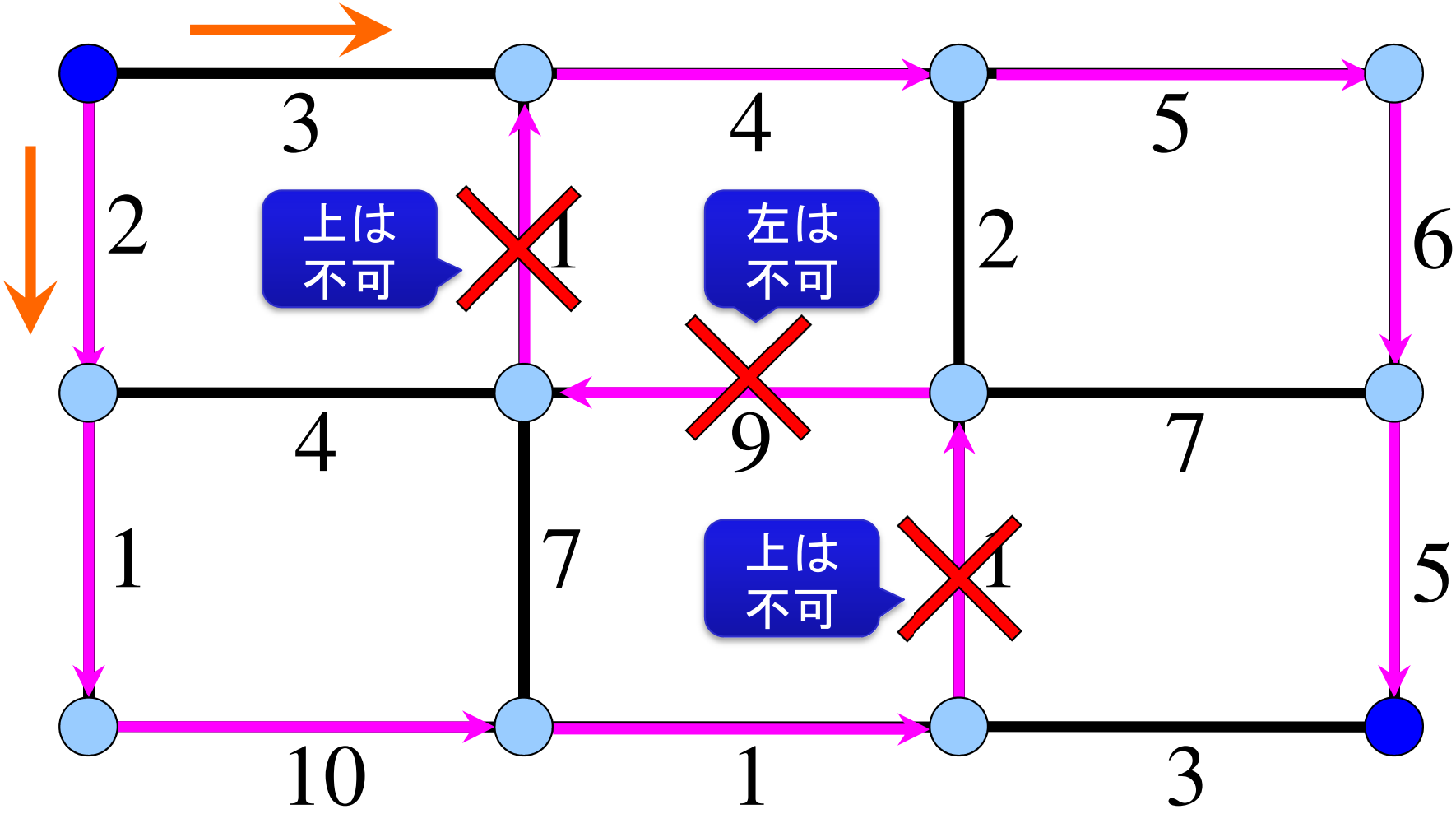
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

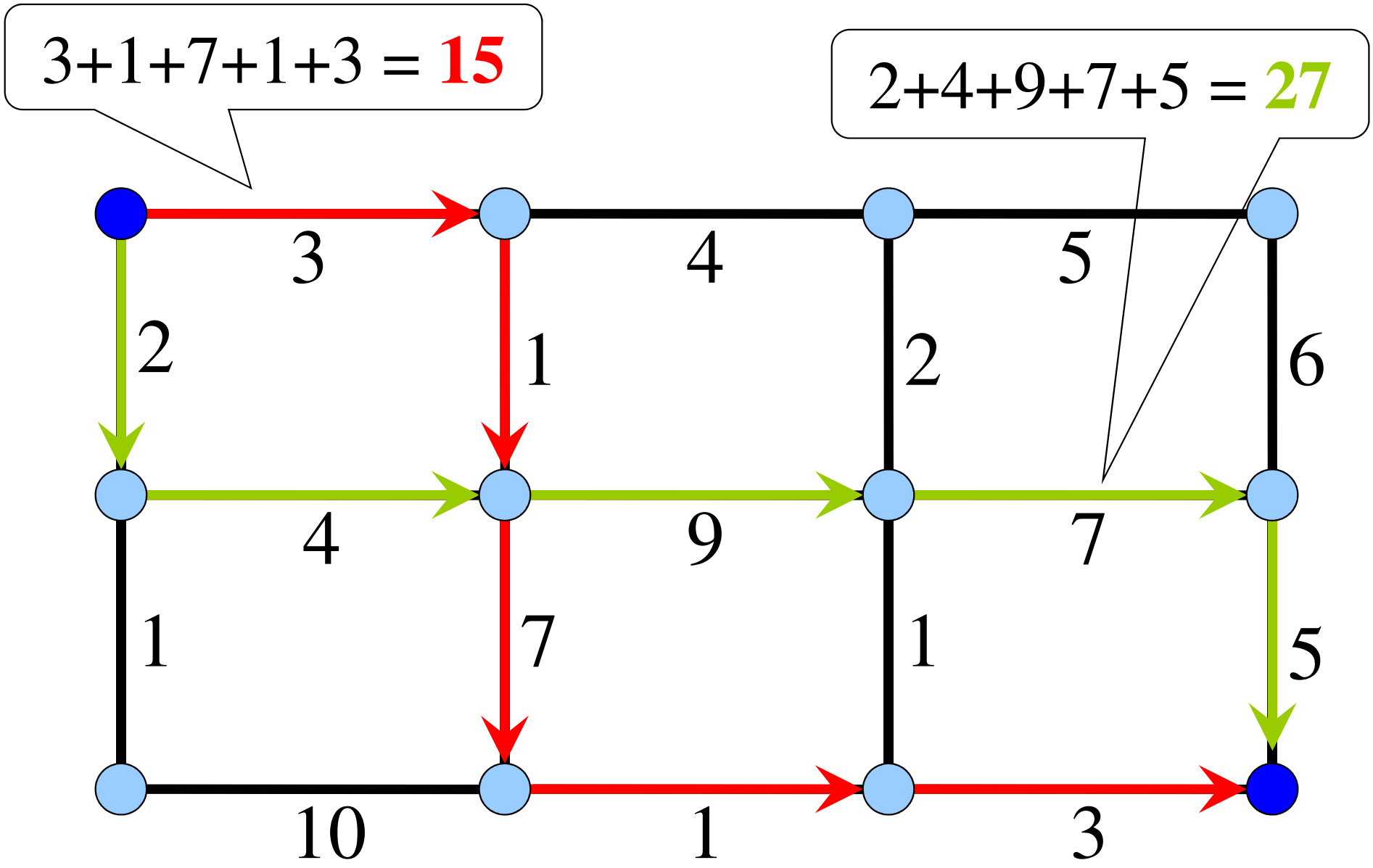
難しいなら易しくすればいいのさ！

制限した問題

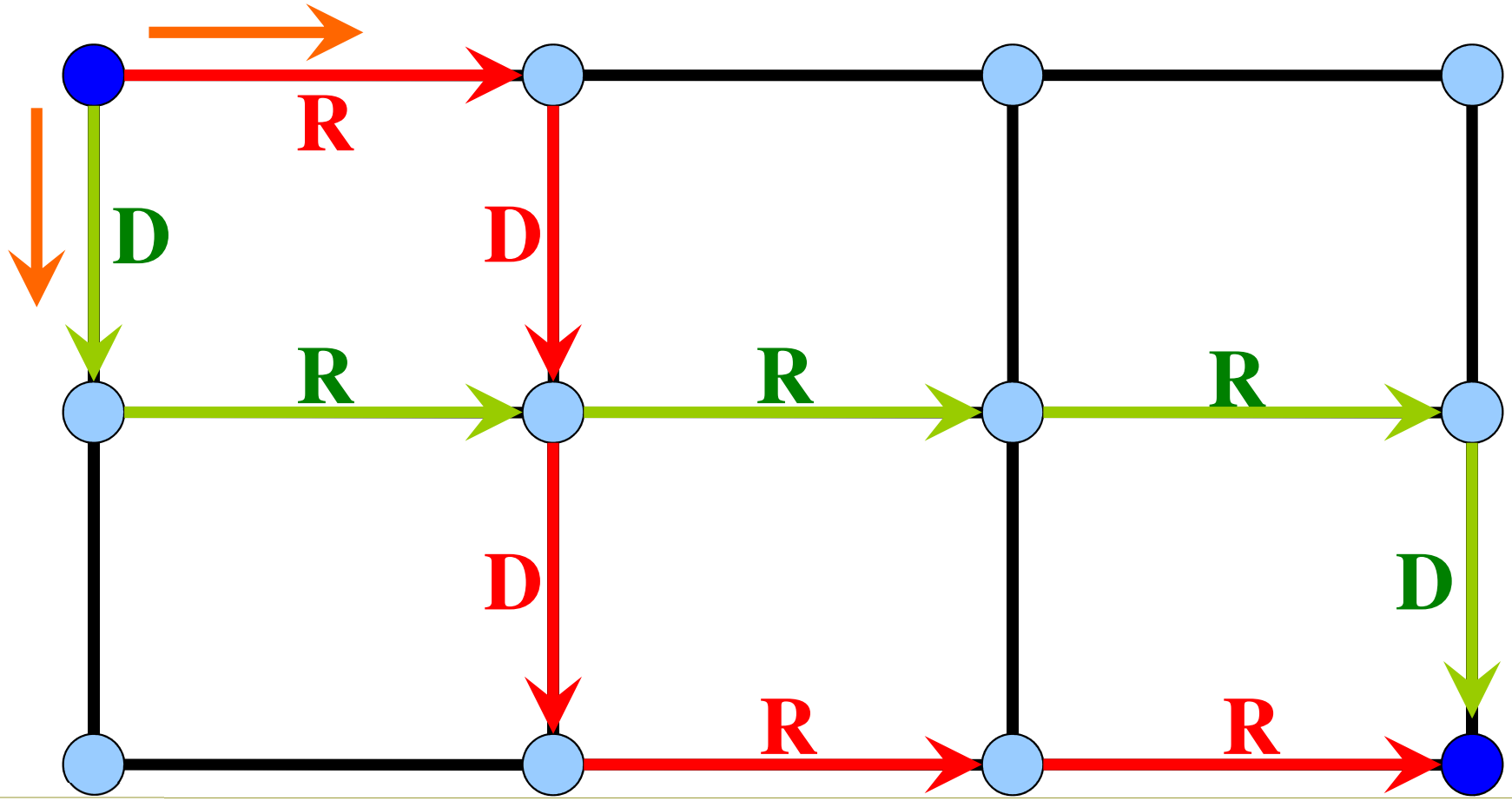
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



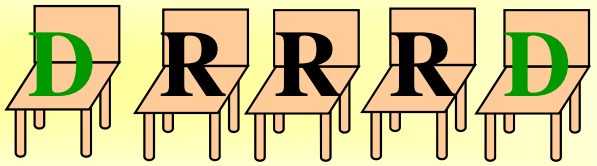
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



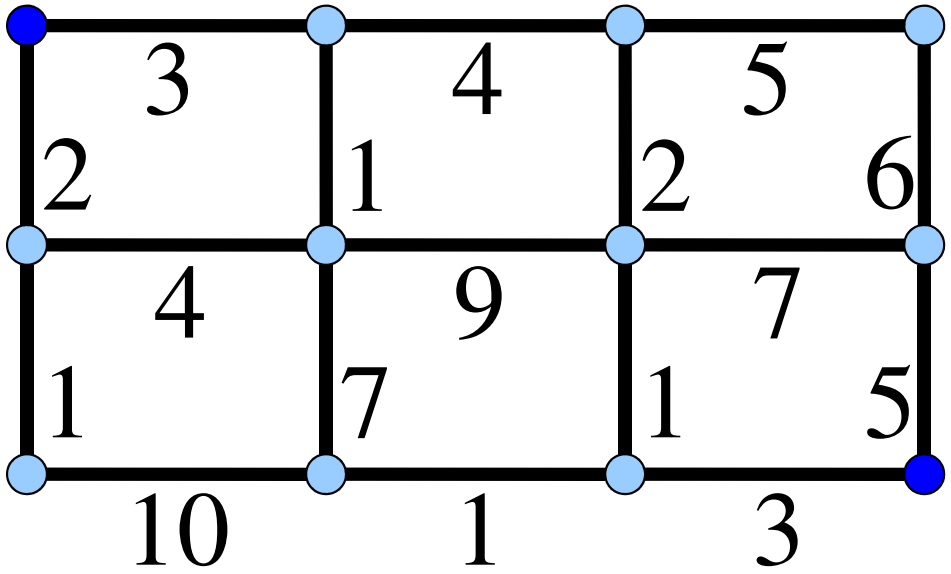
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

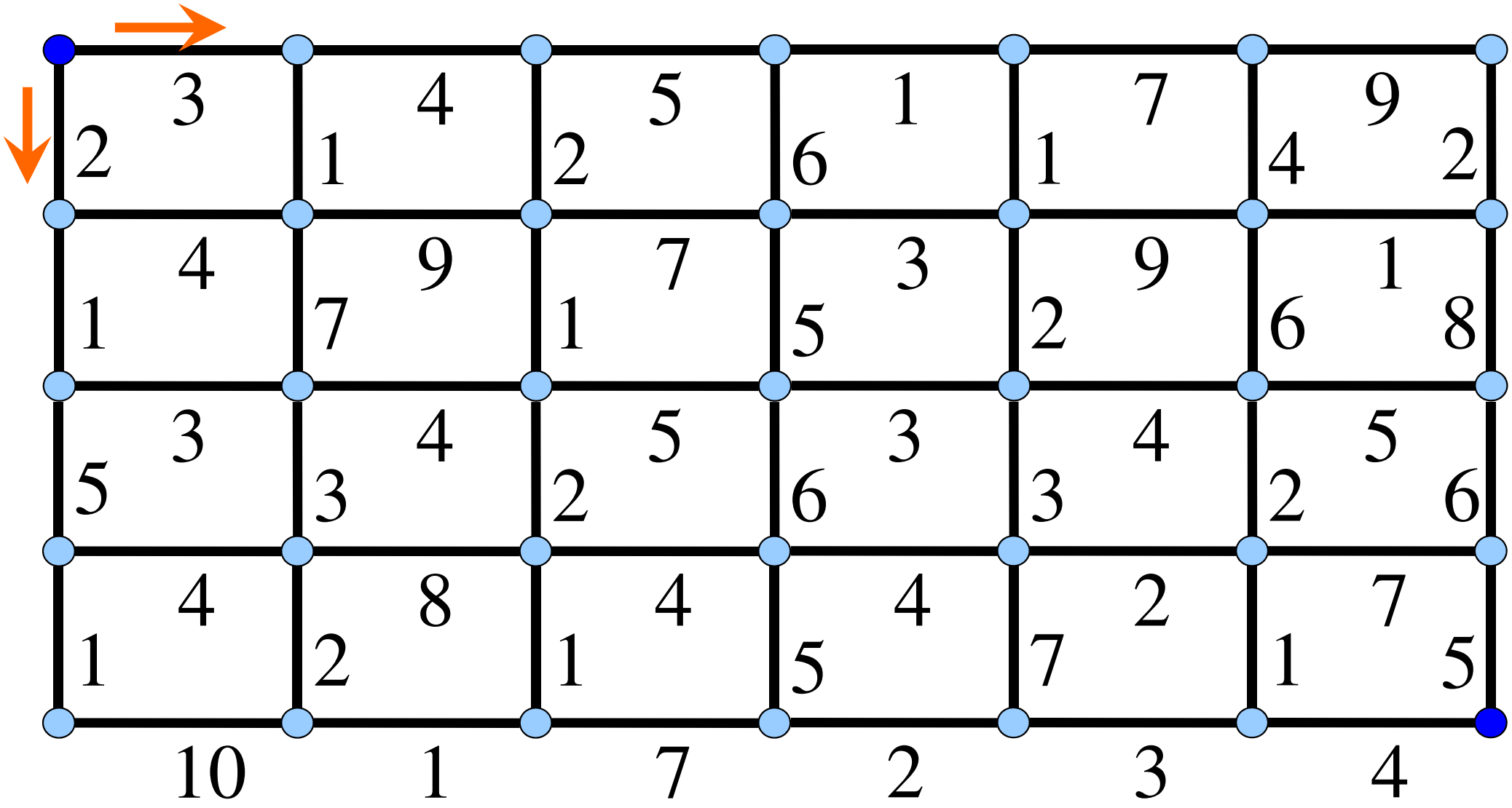
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

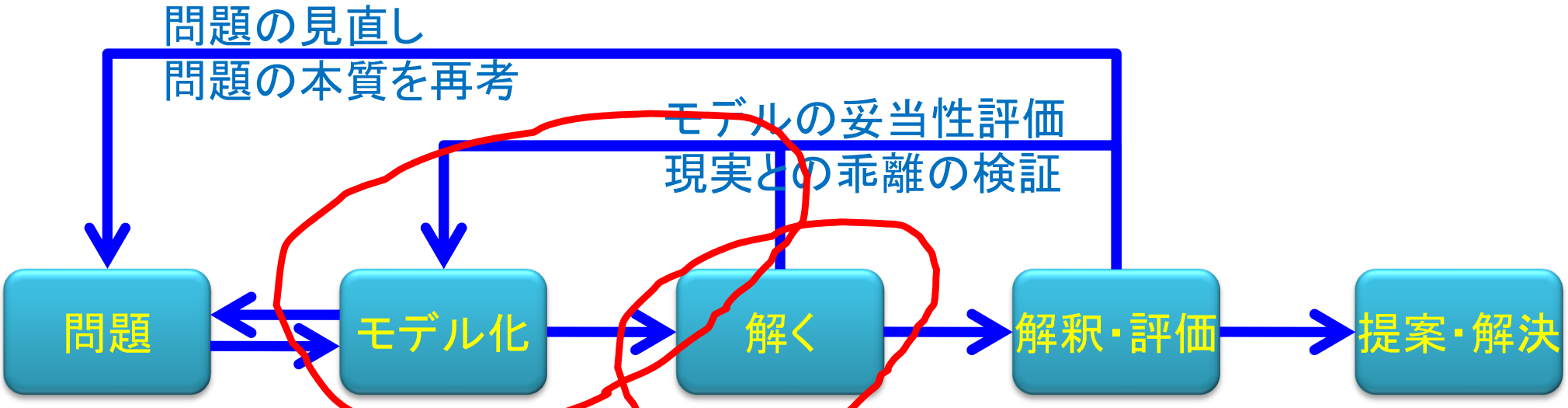
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

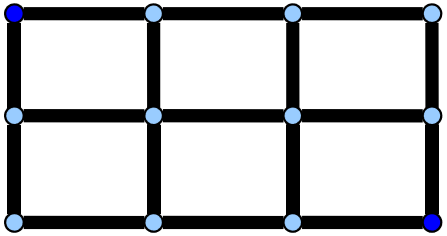
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

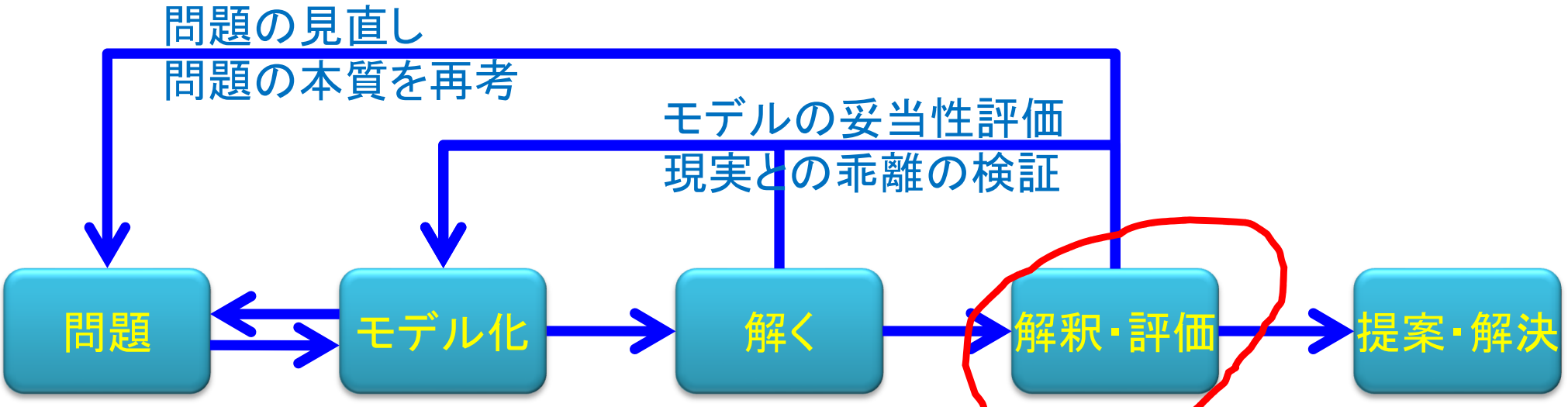
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

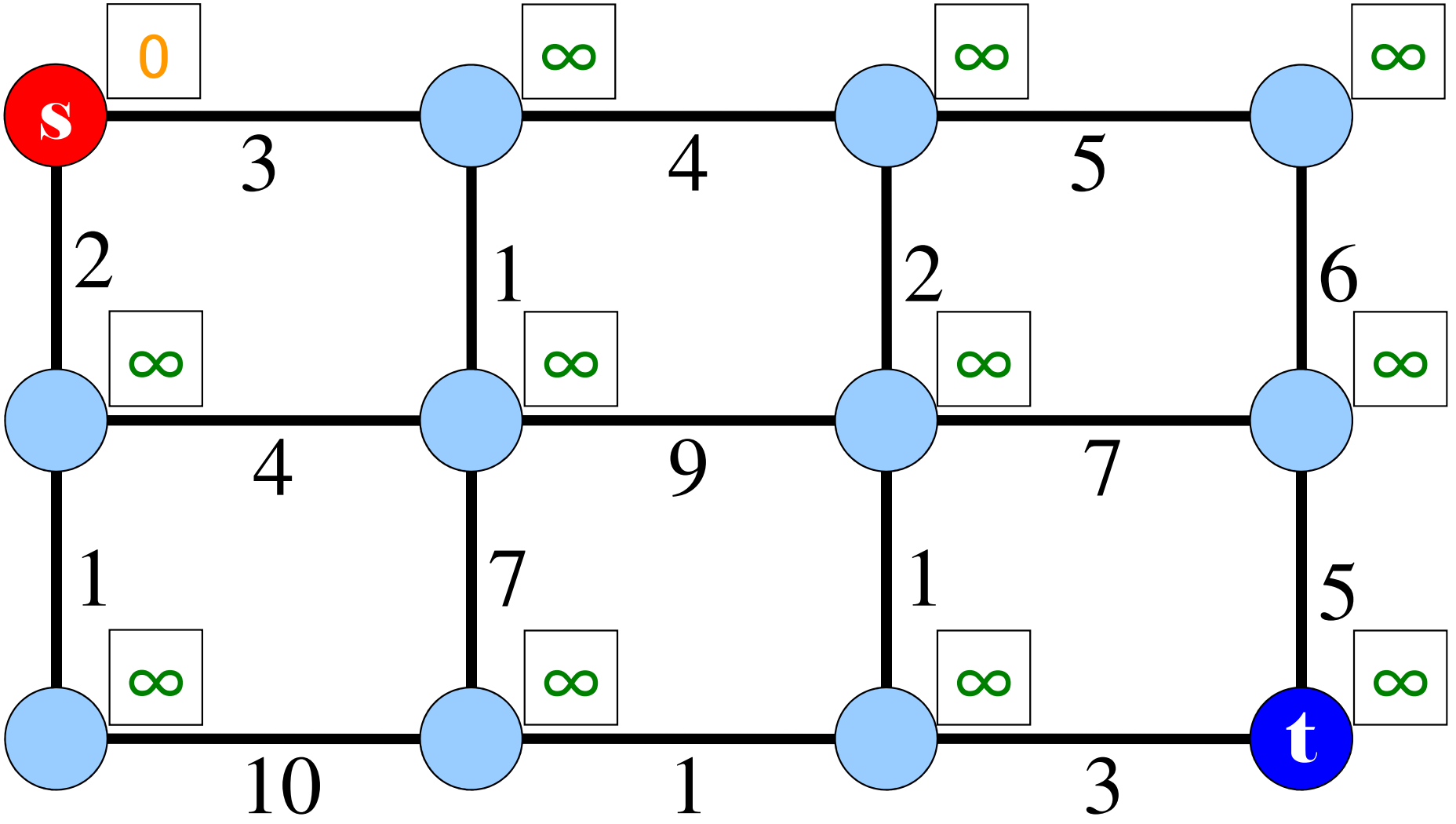
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

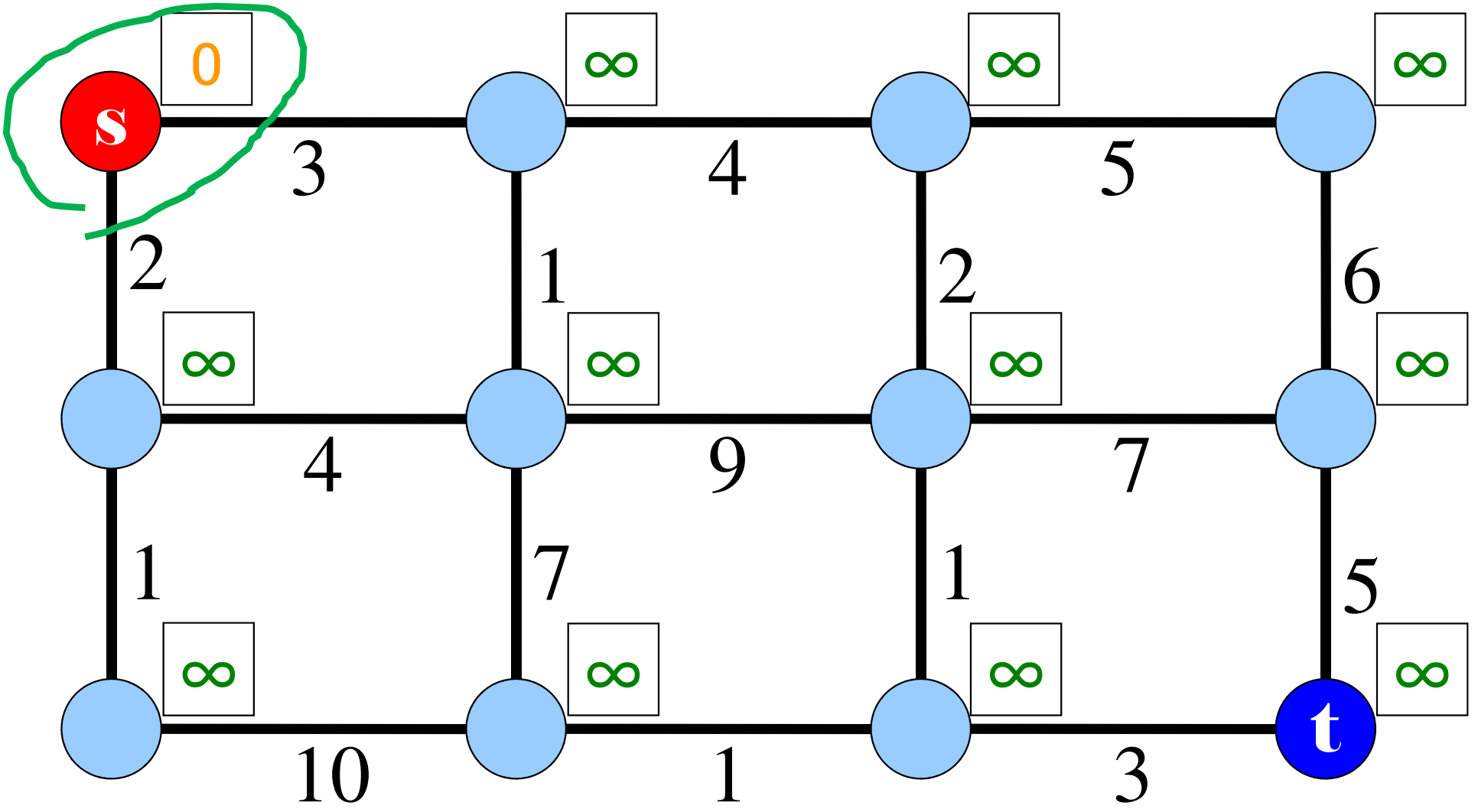
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



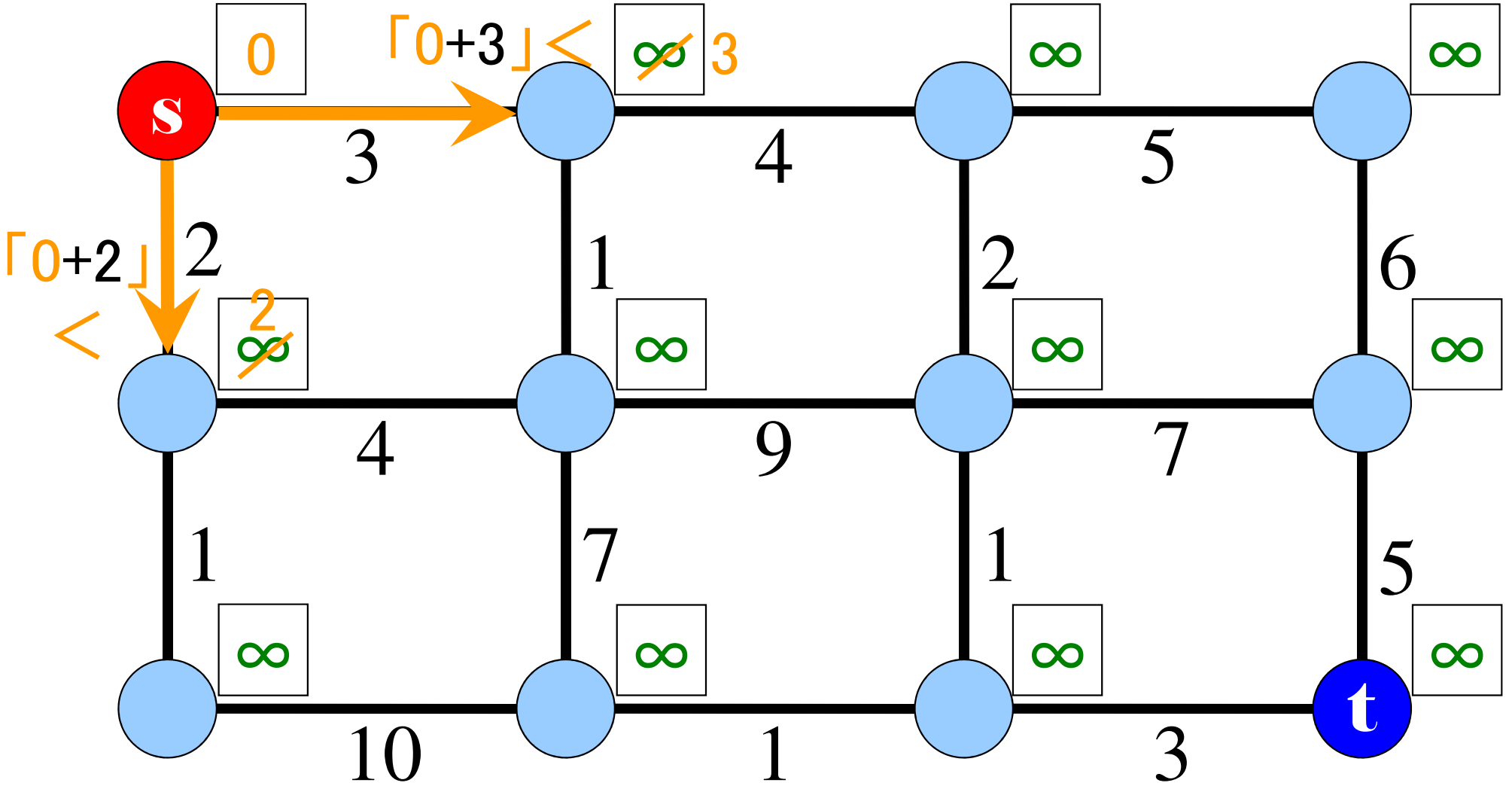
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

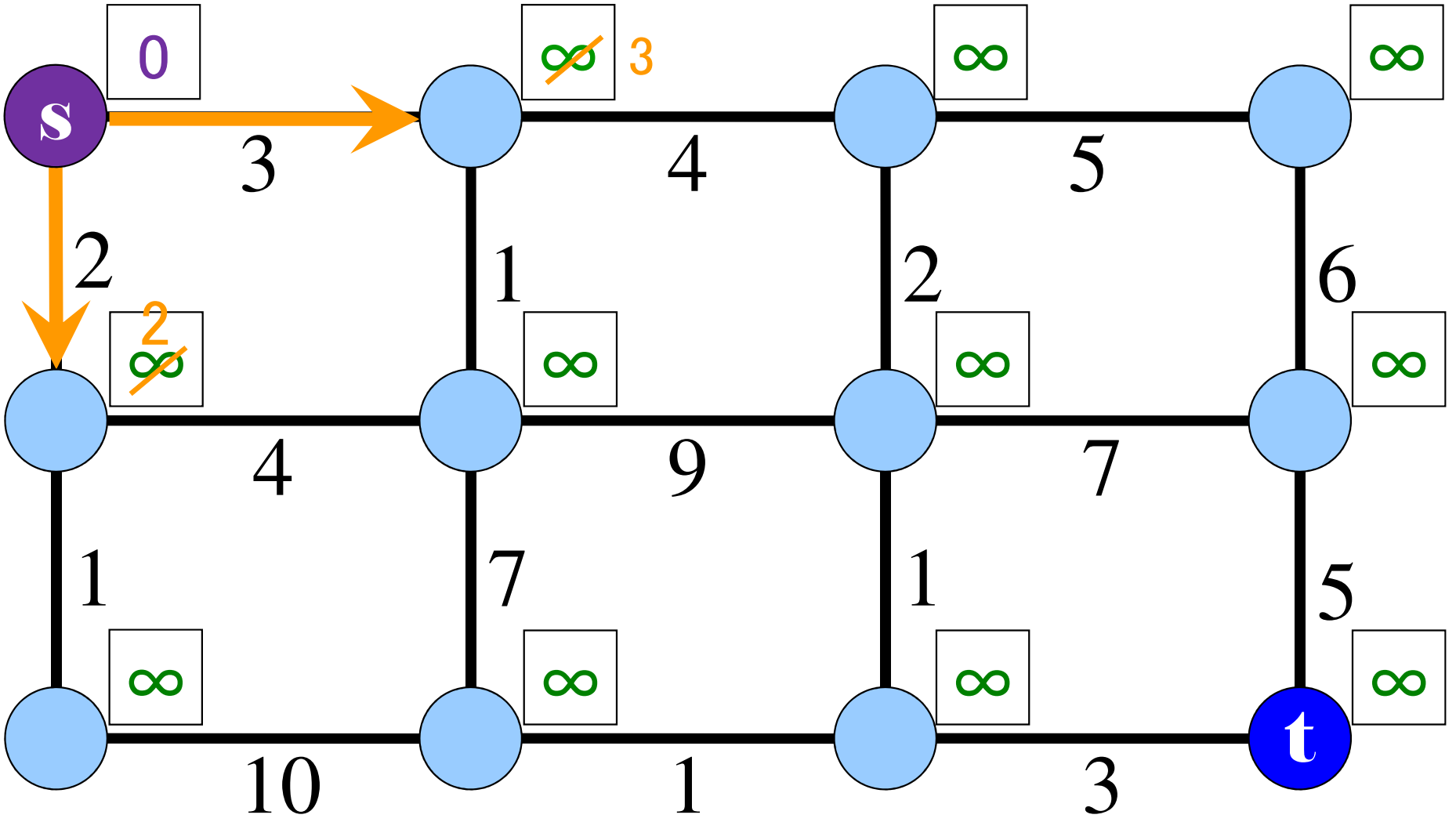


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

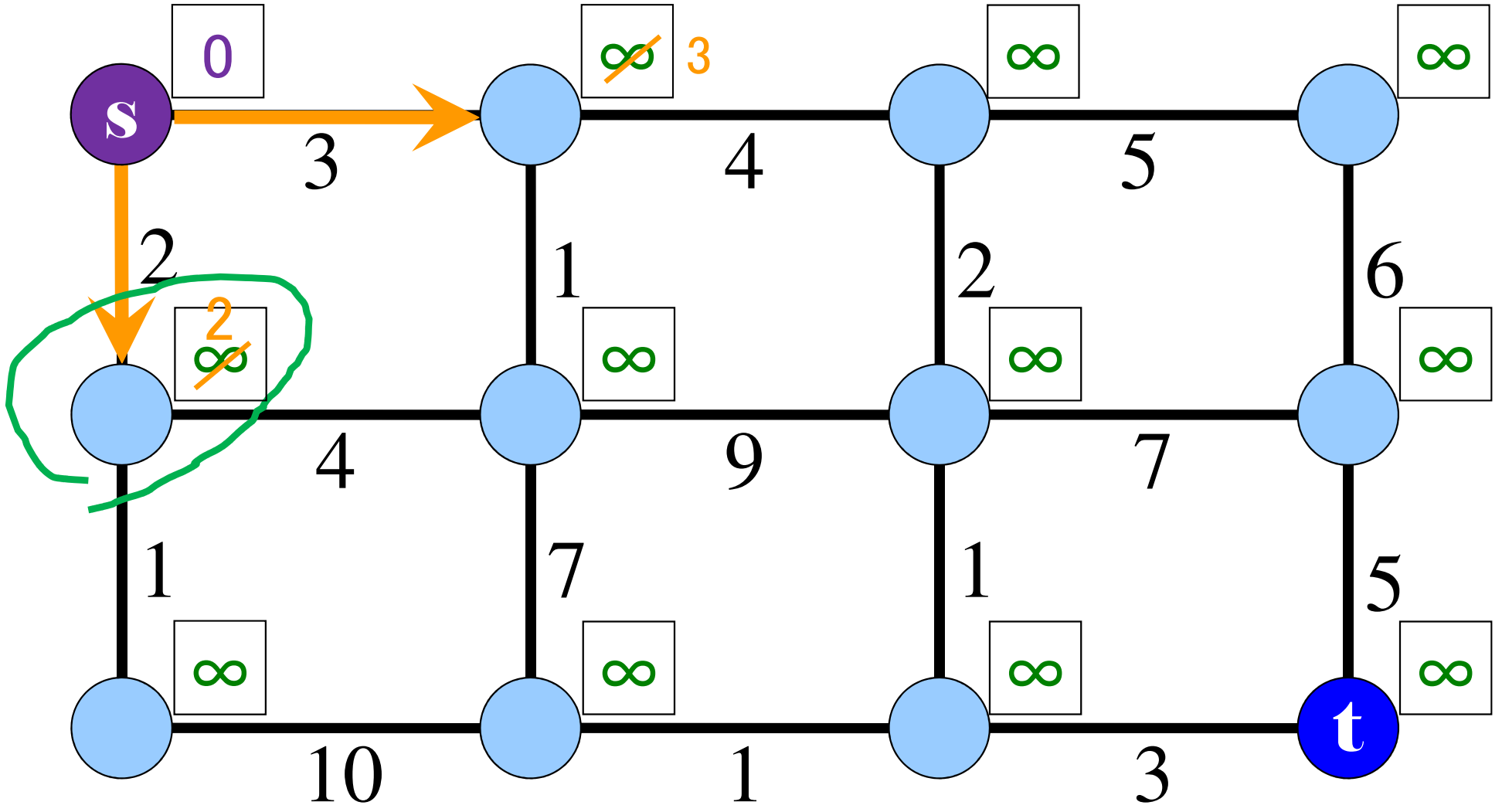
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



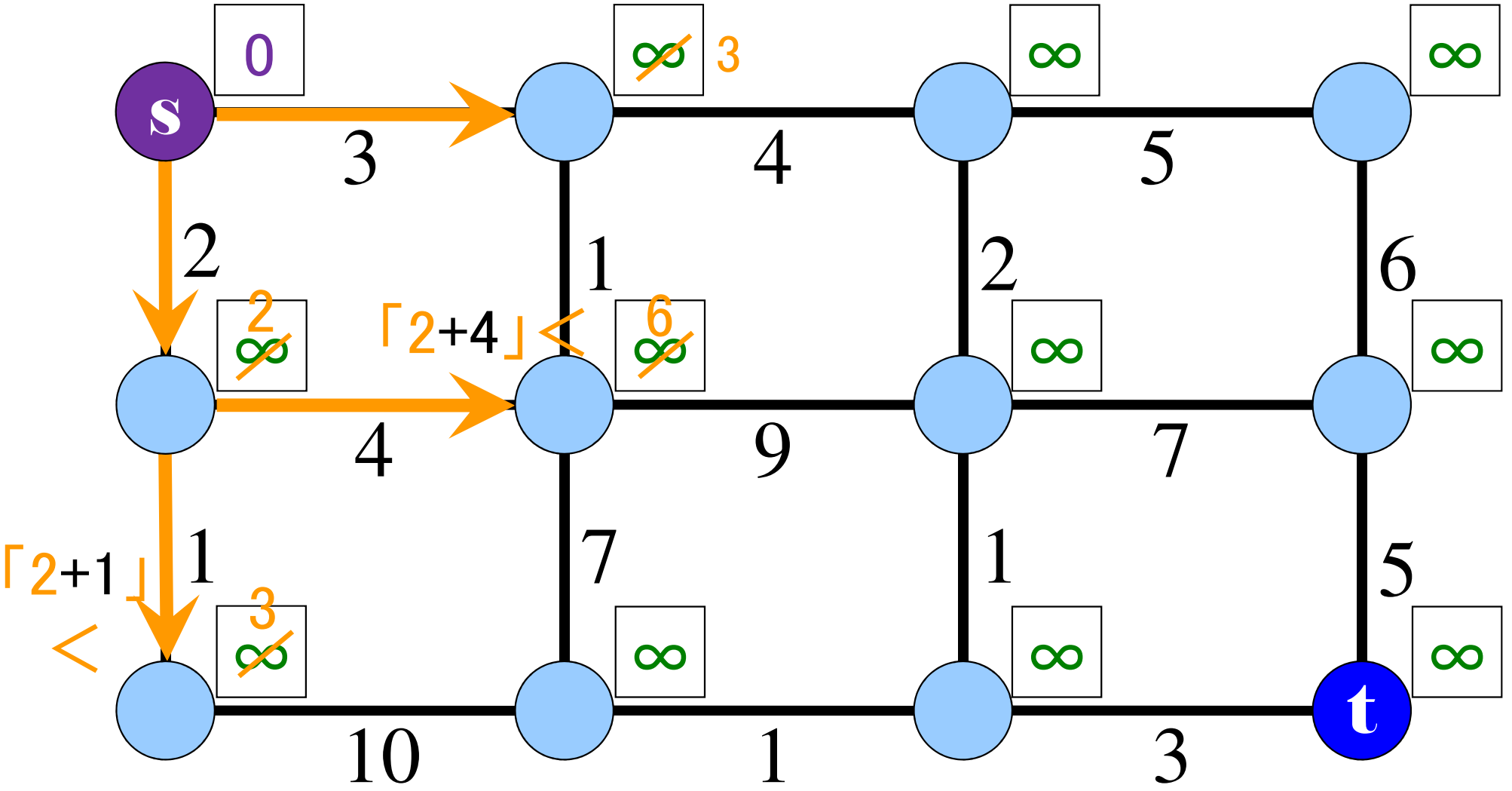
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

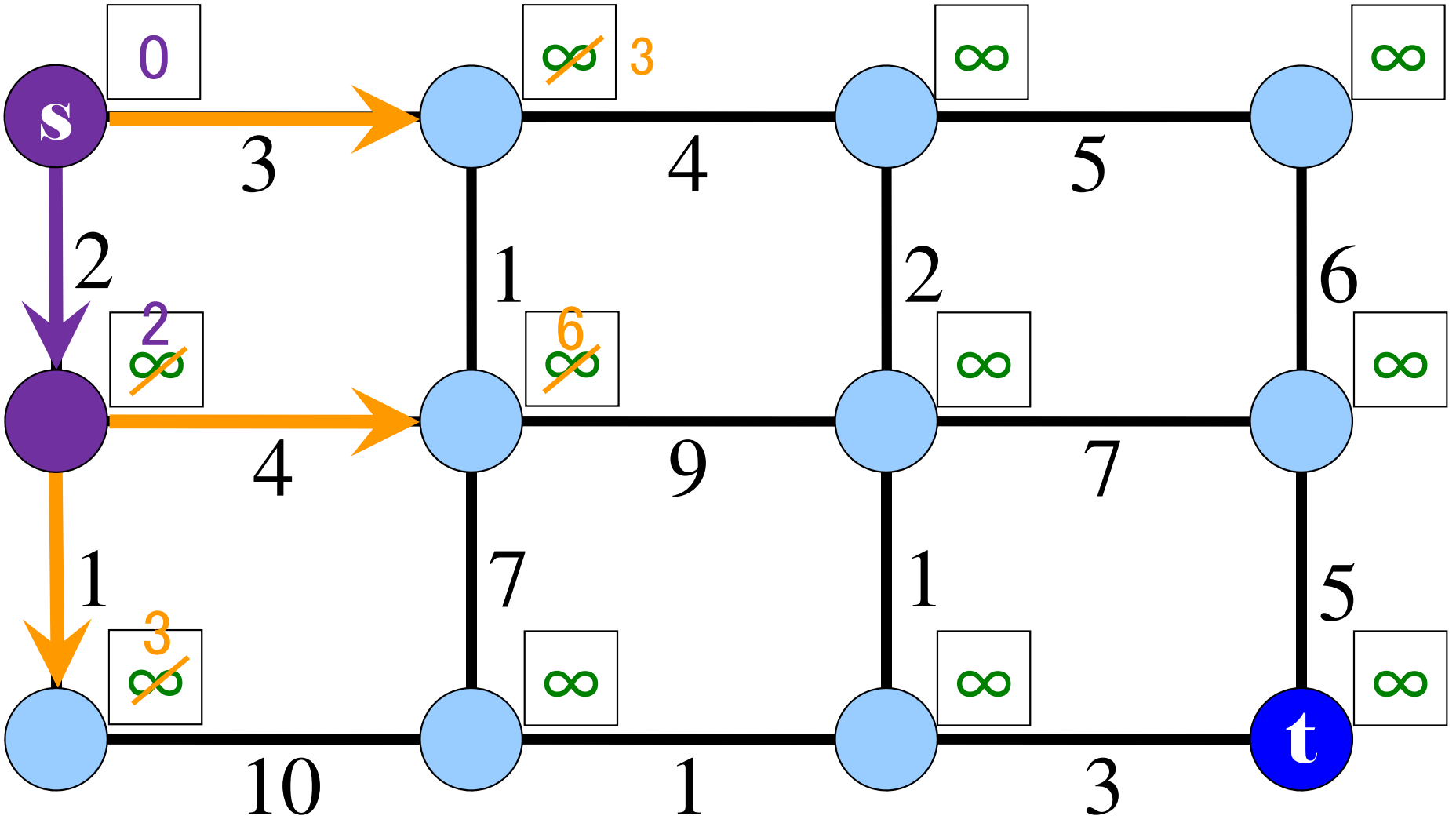


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

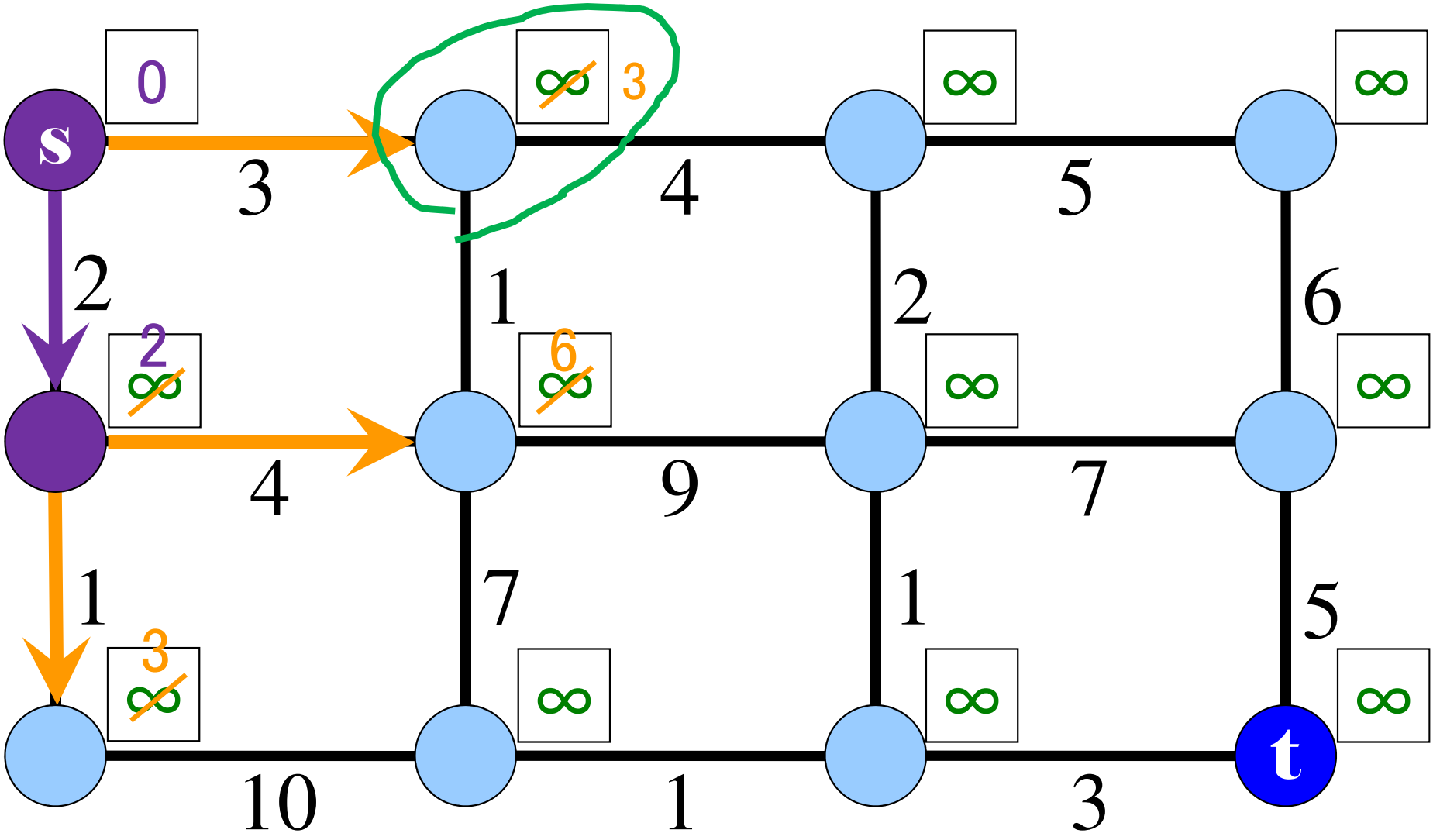
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



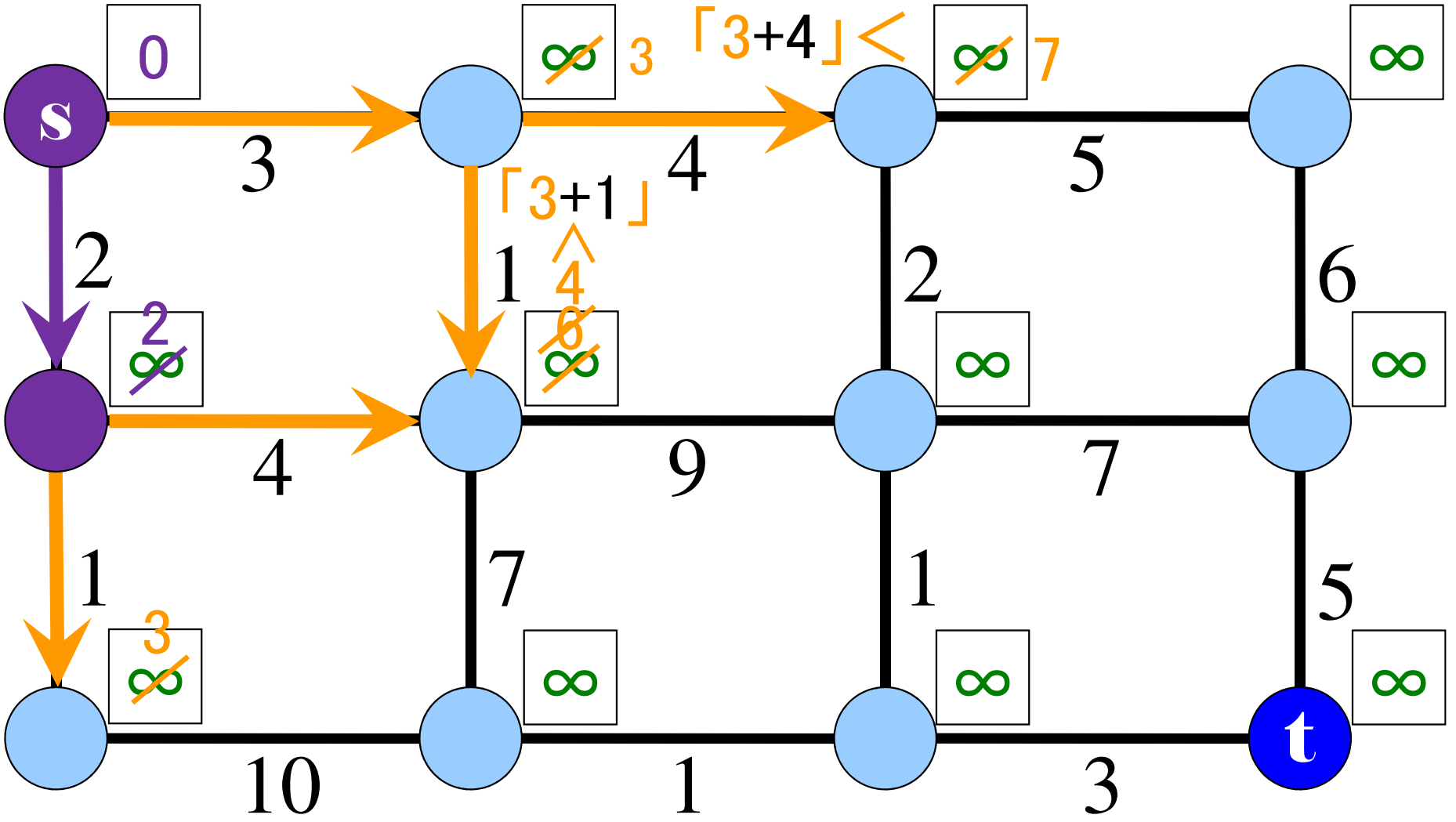
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

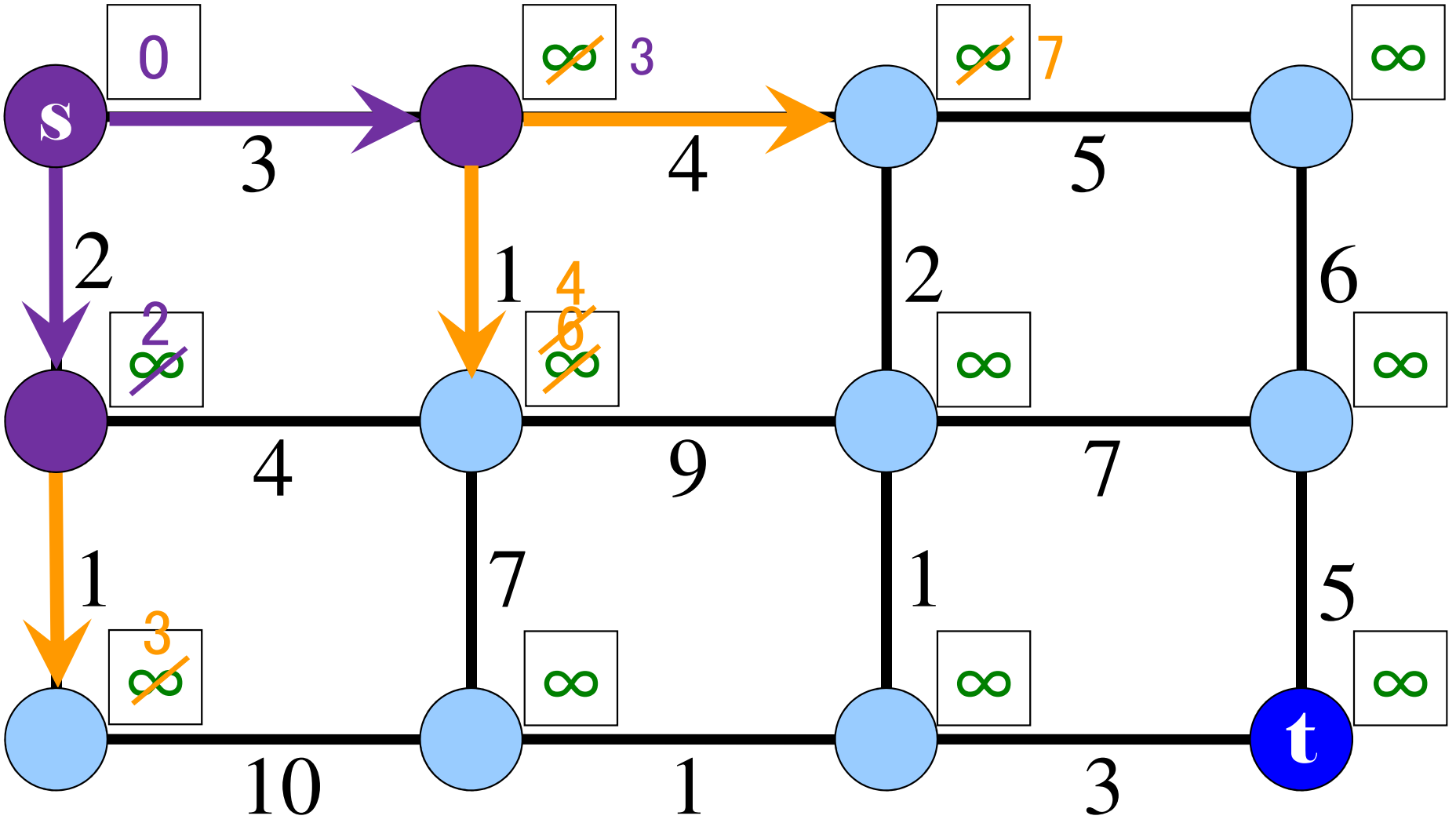


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

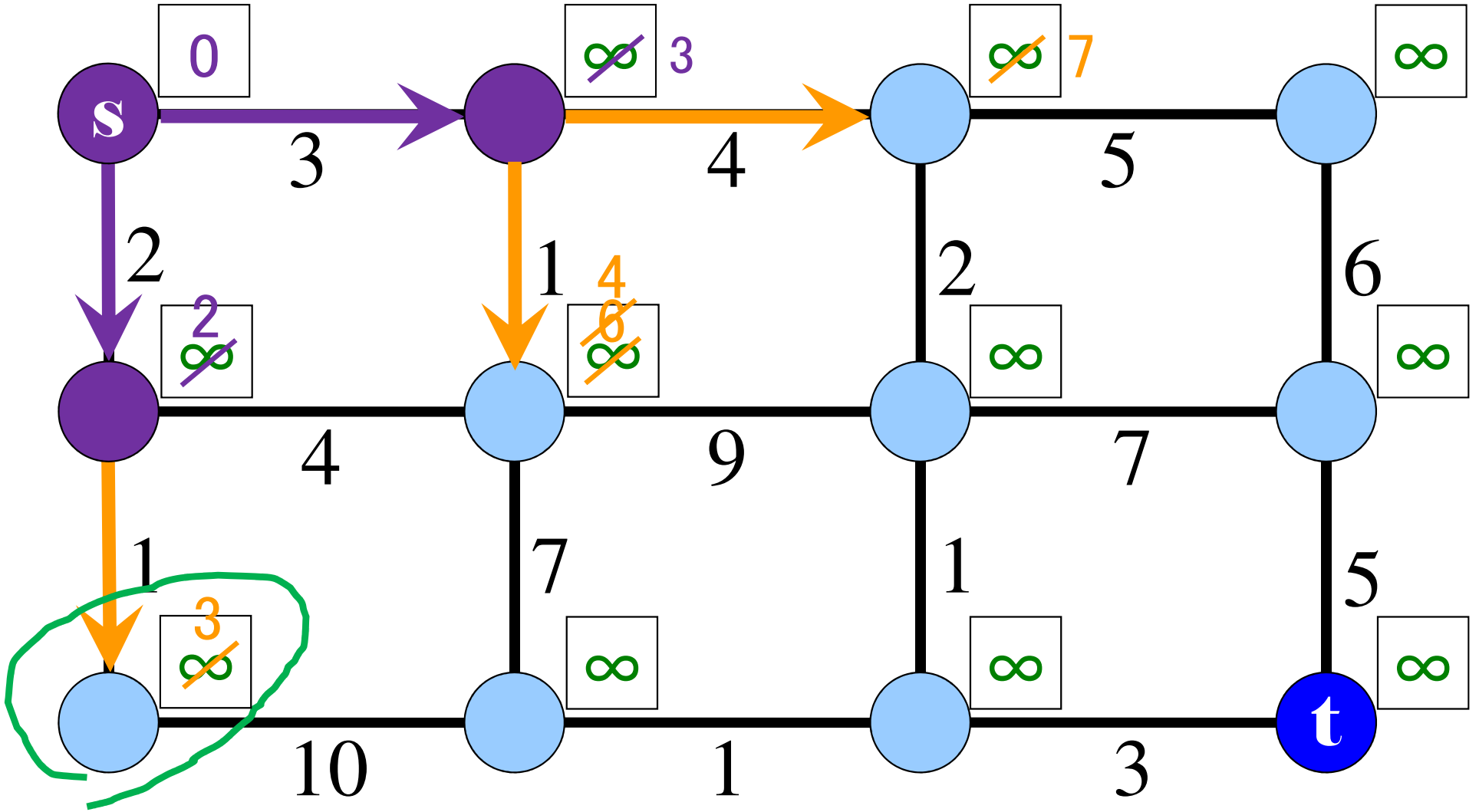
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



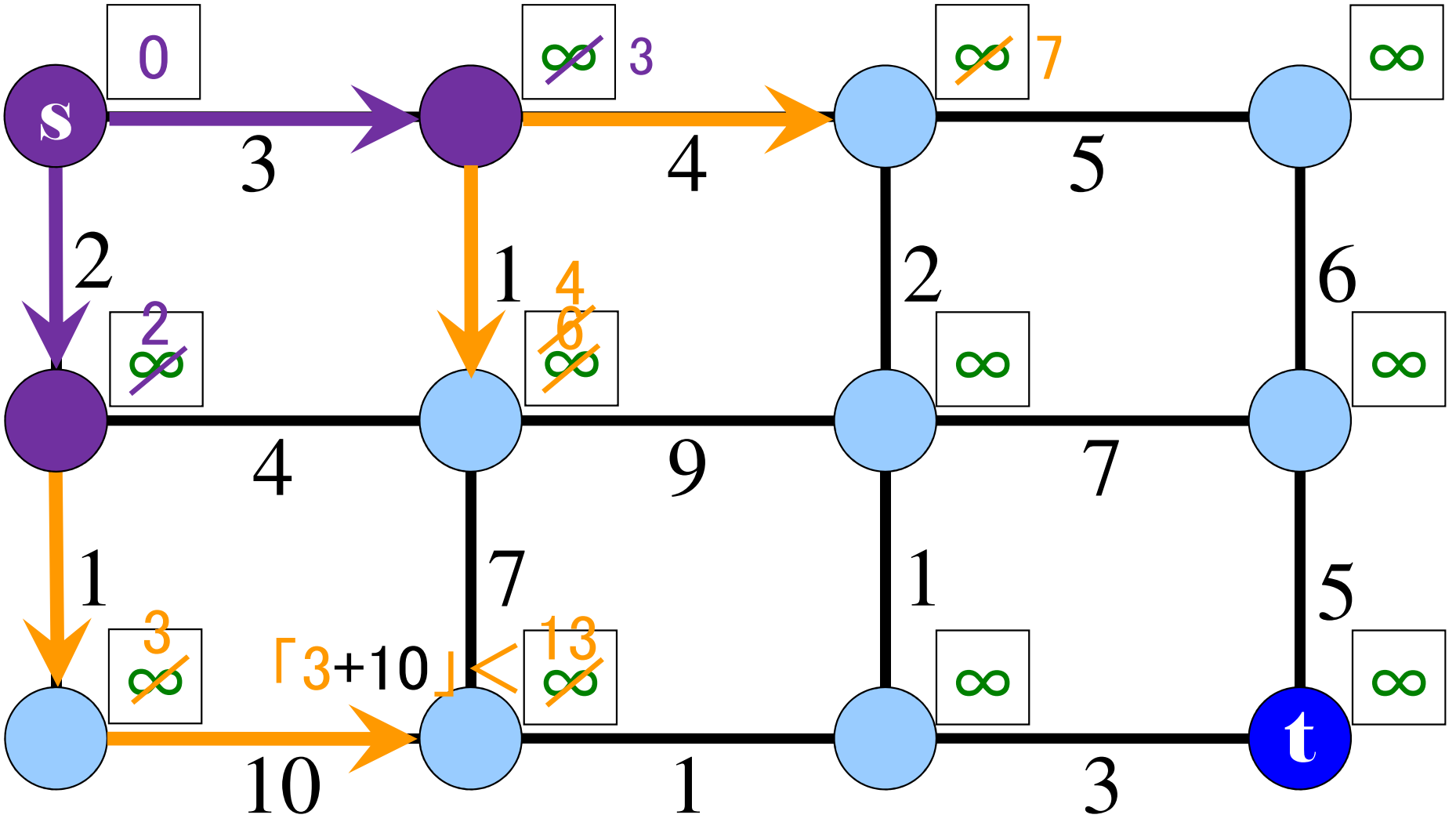
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

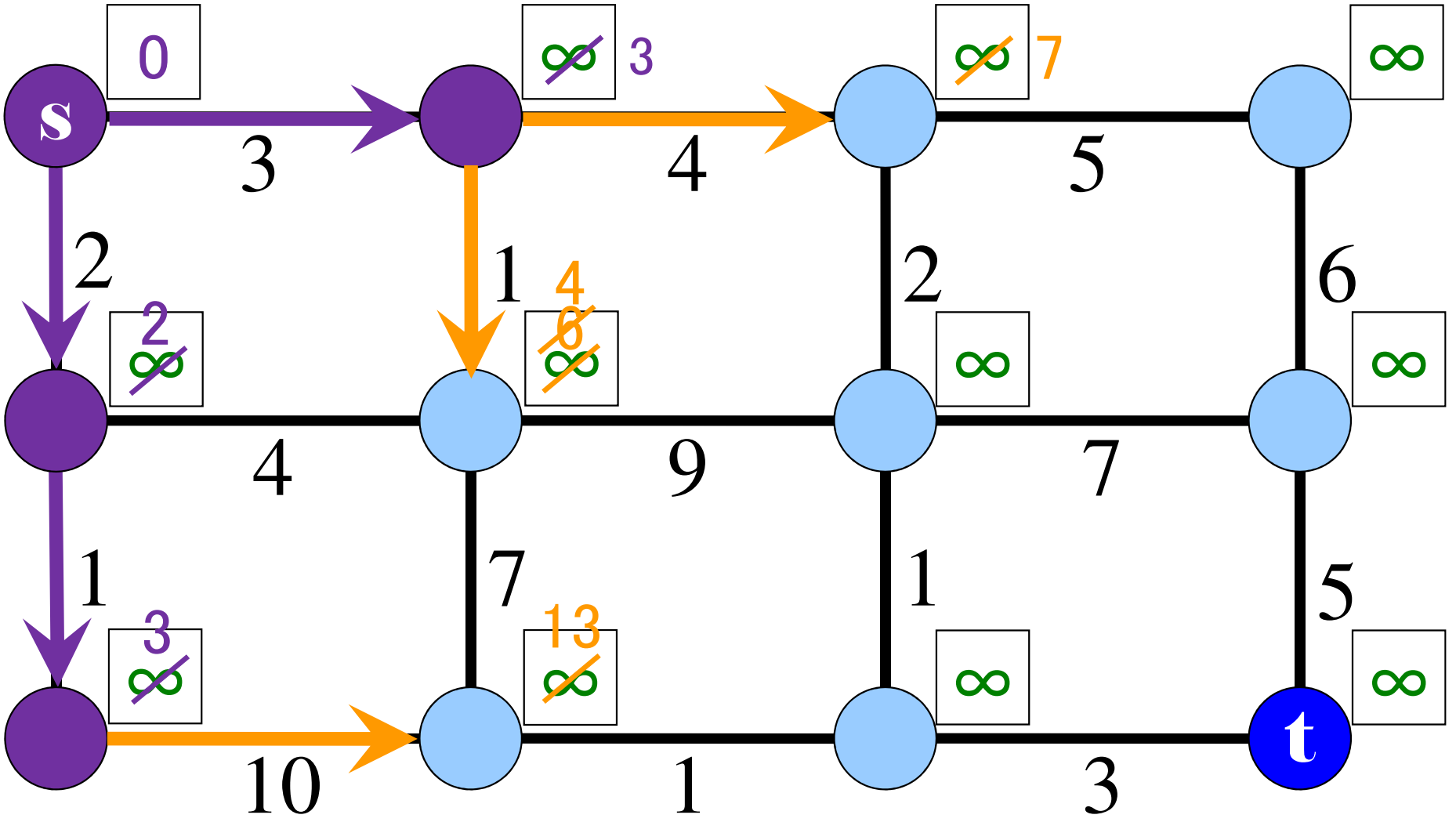


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

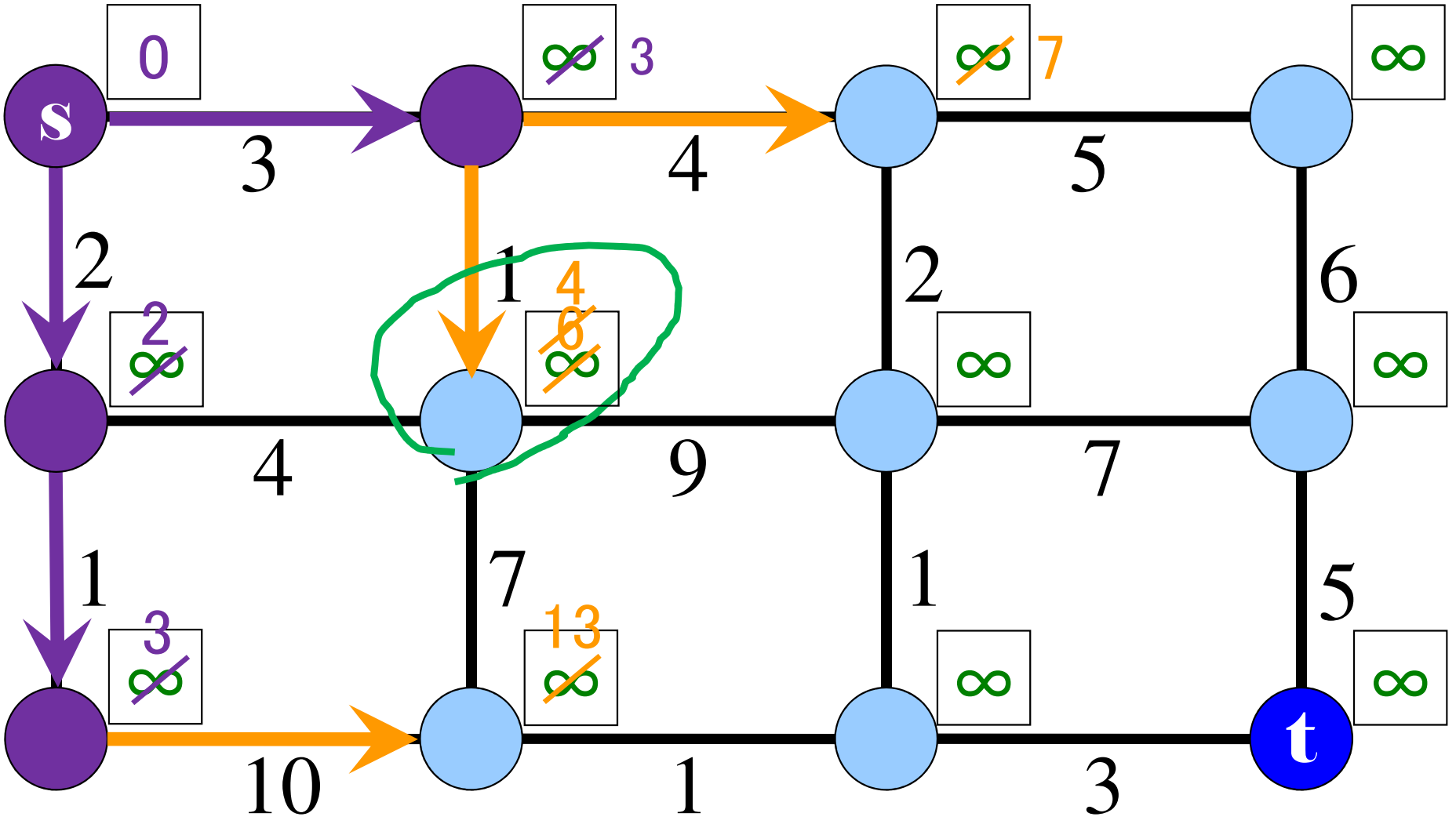
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



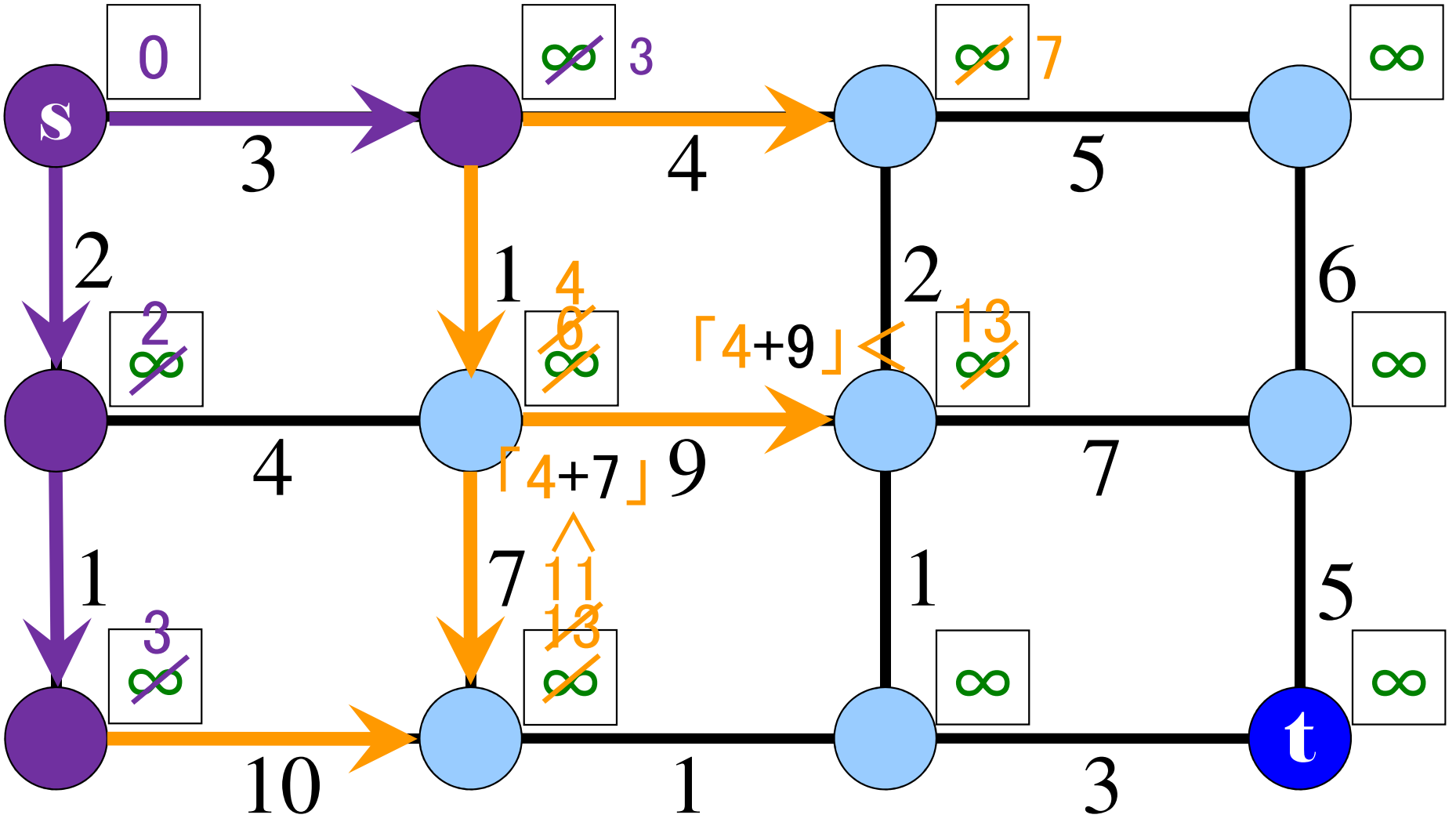
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

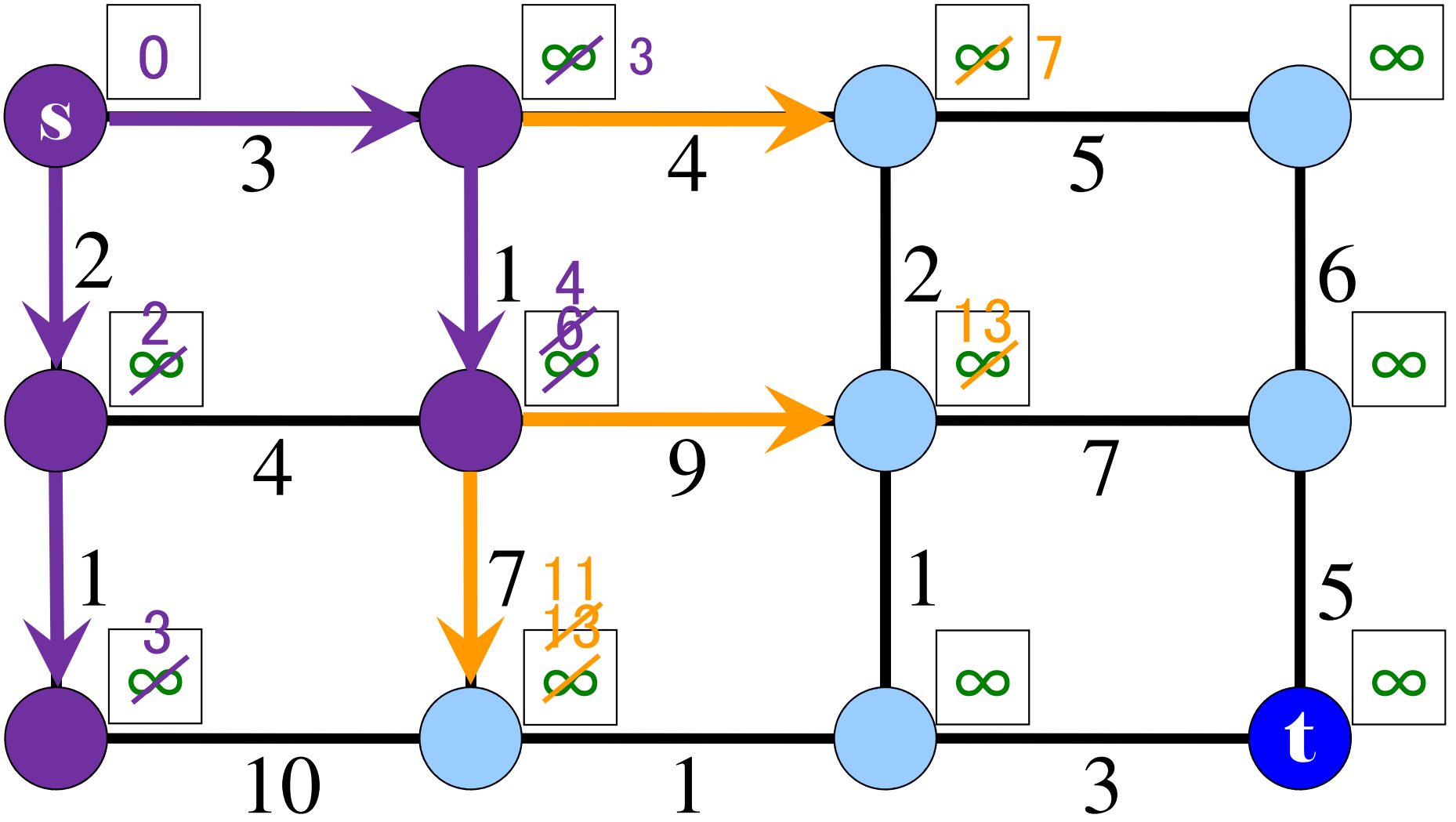


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

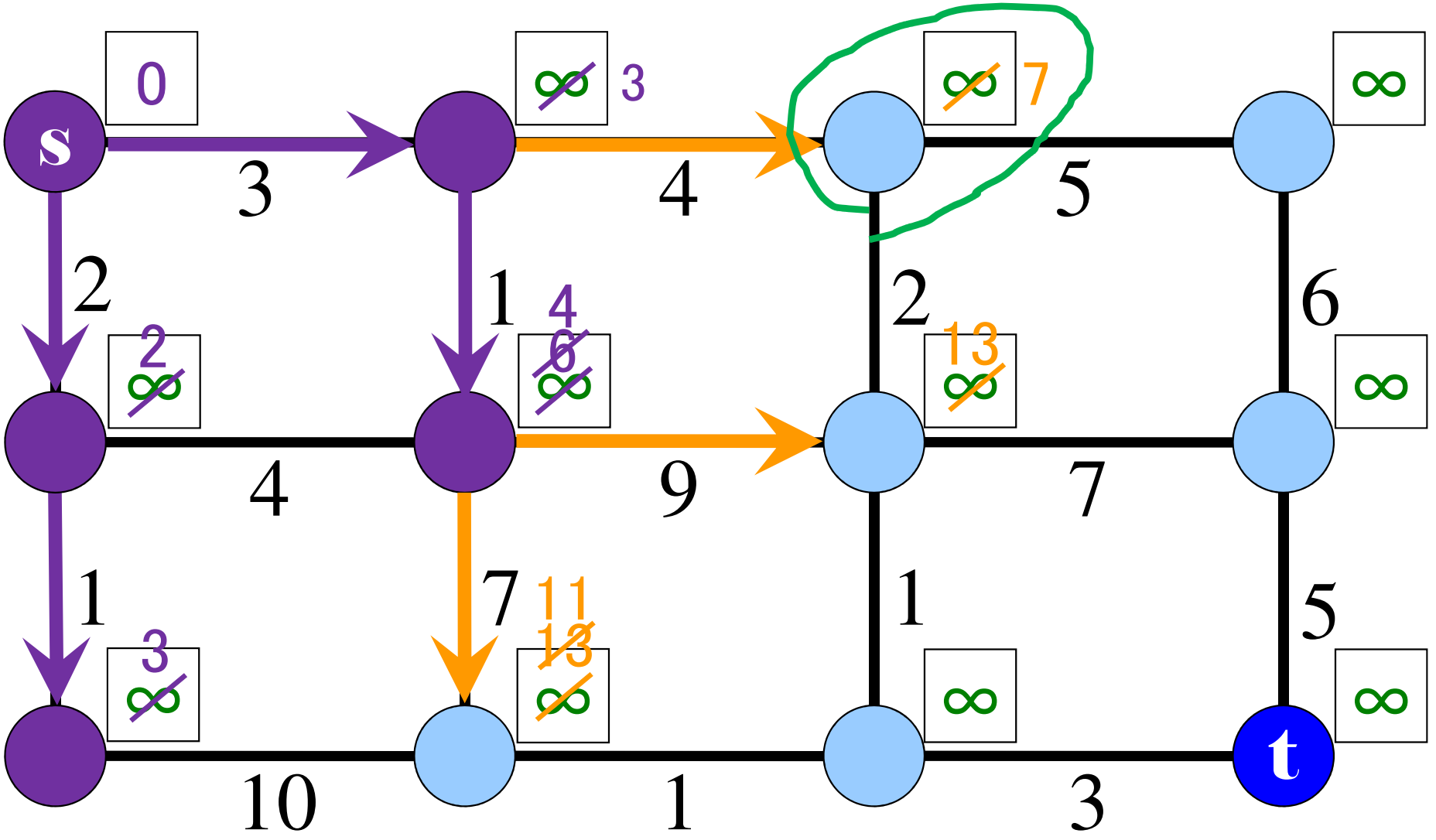
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



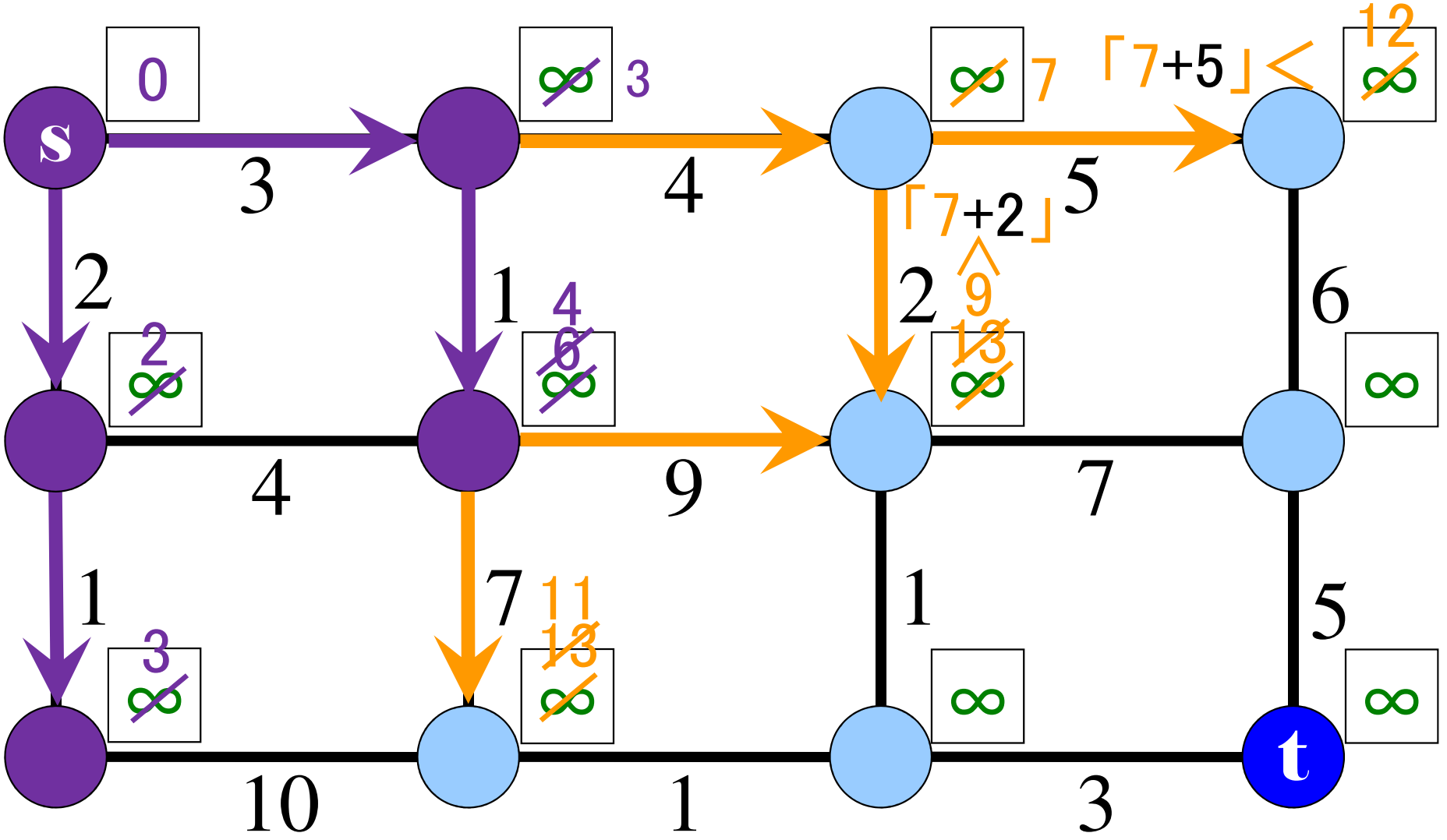
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

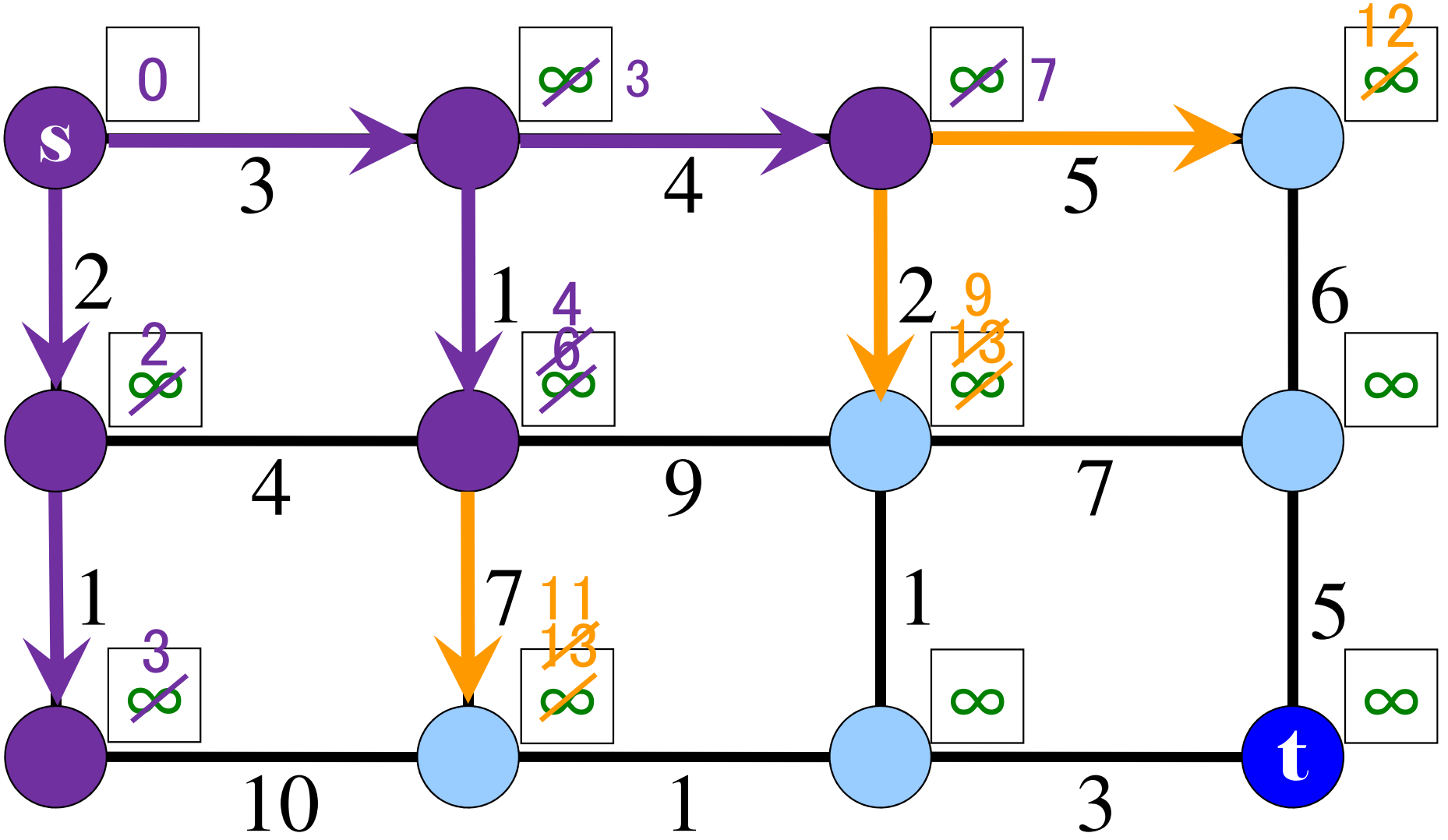


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

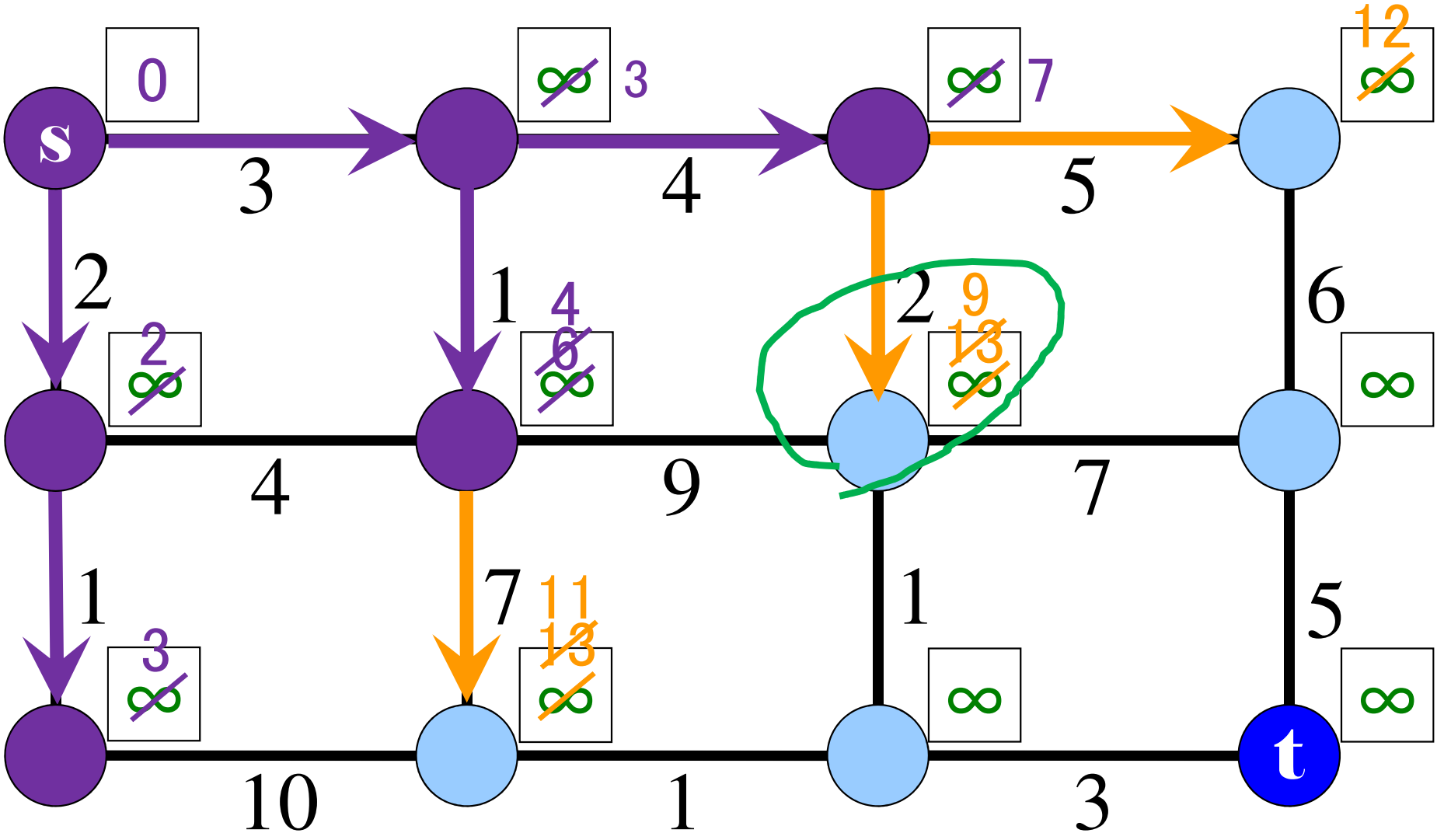
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



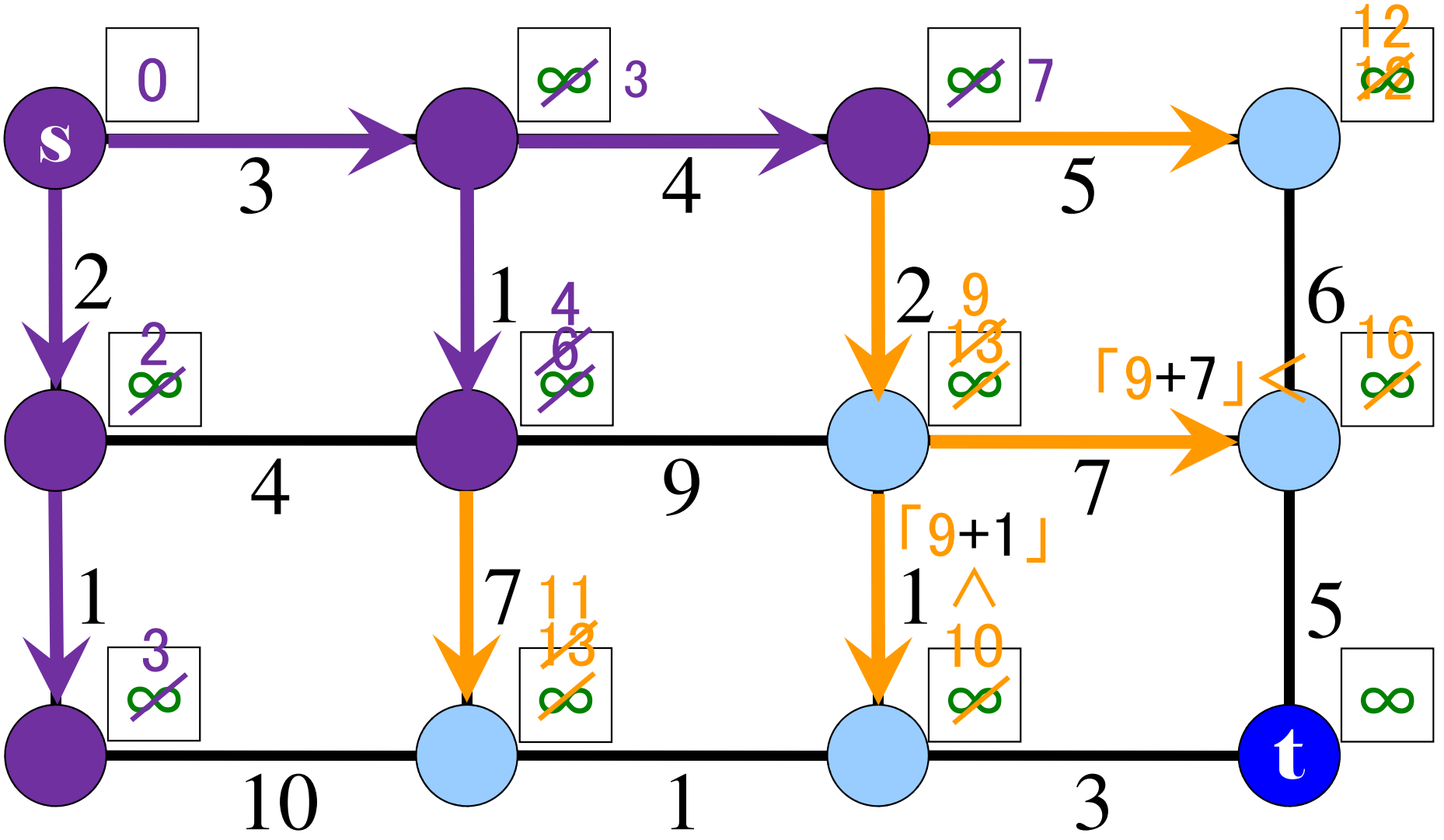
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

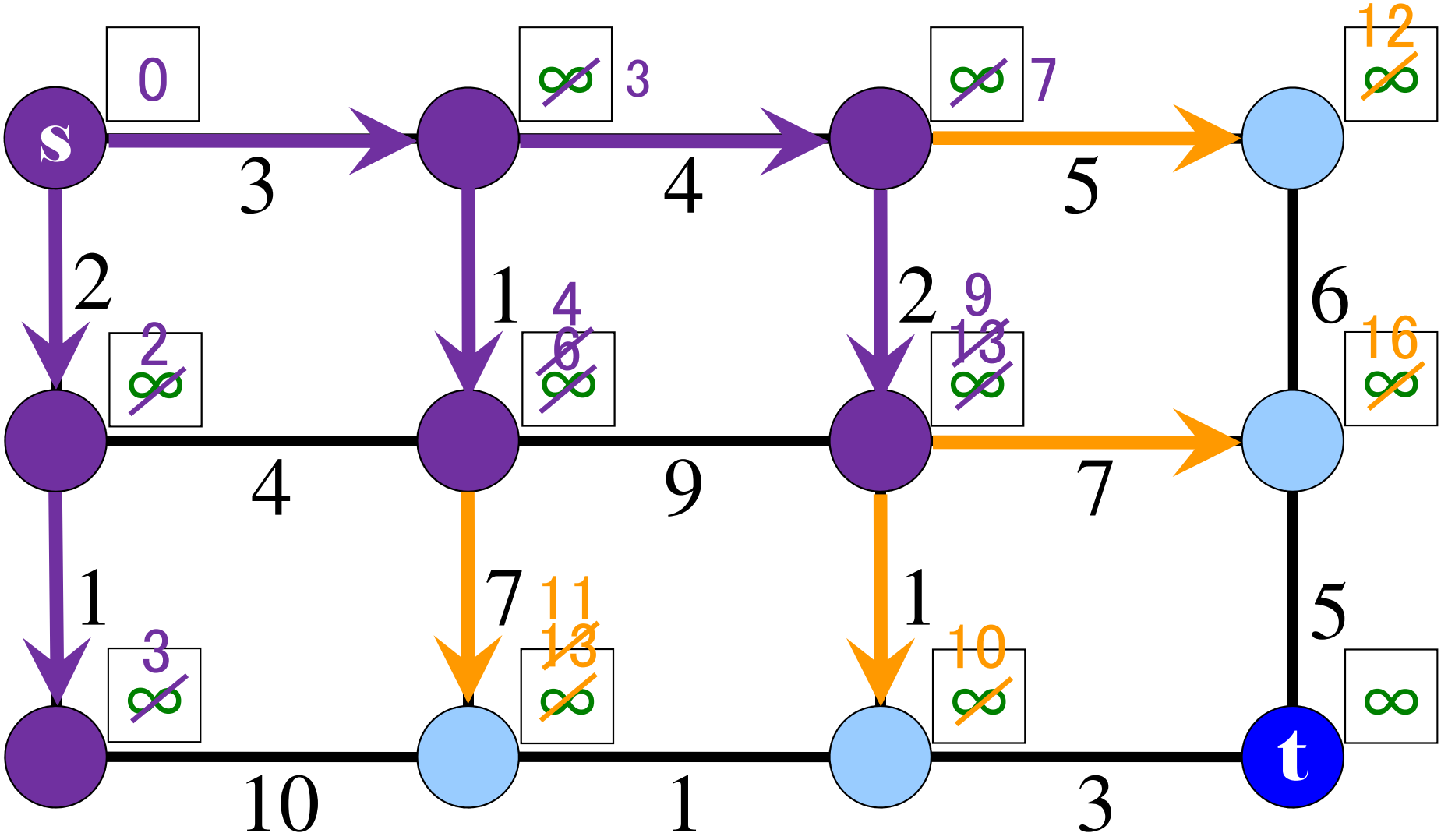


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

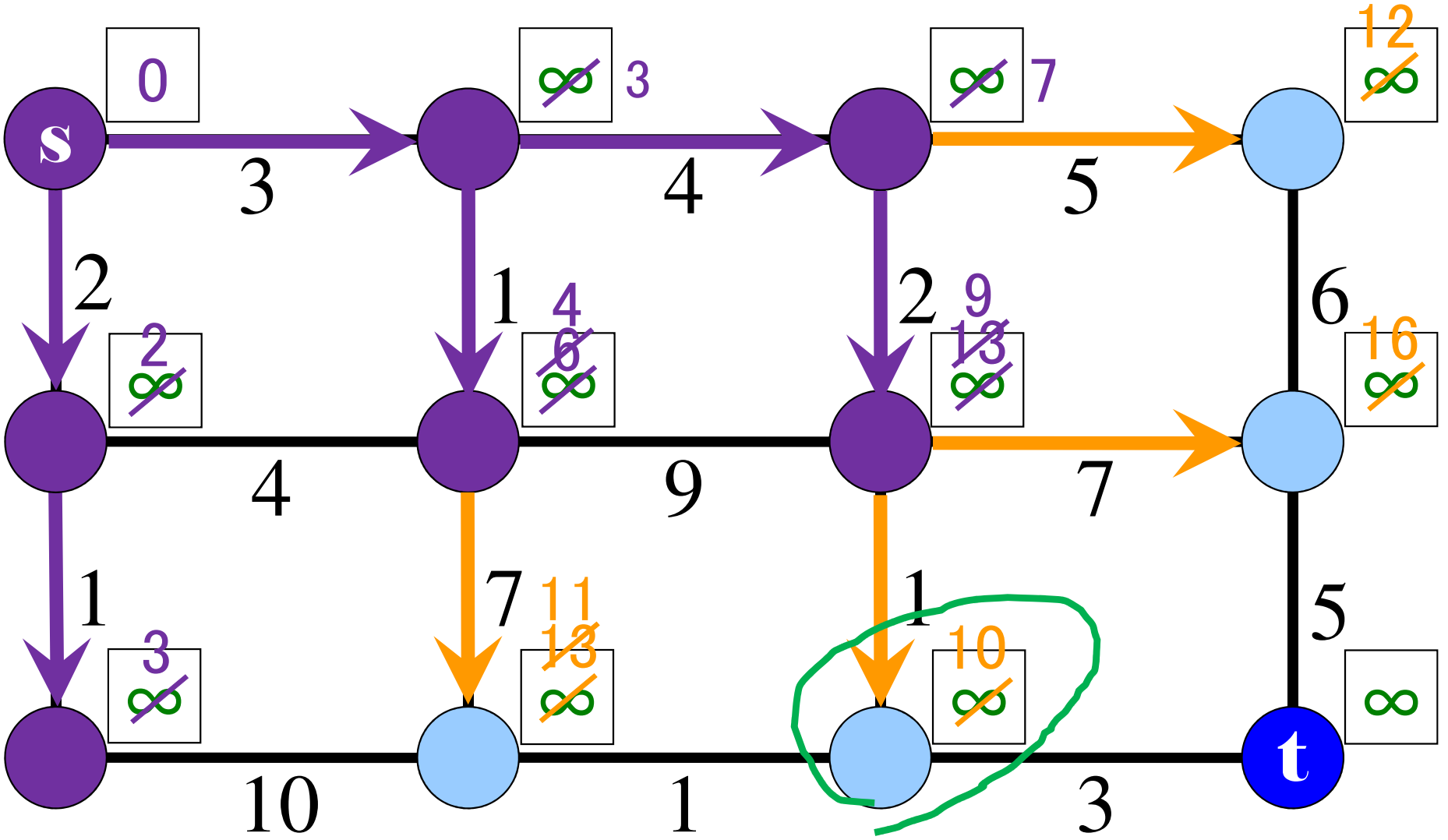
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



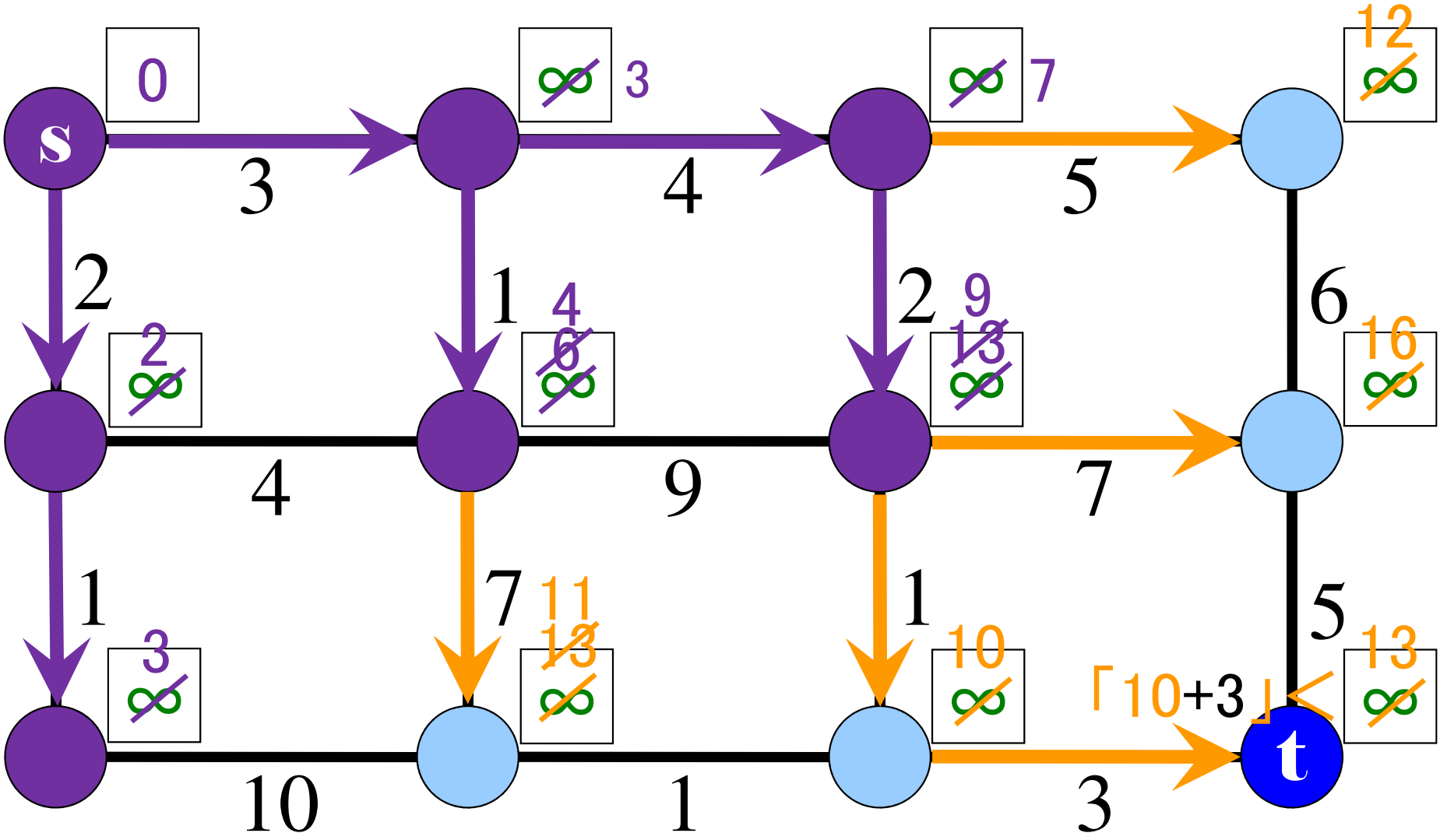
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

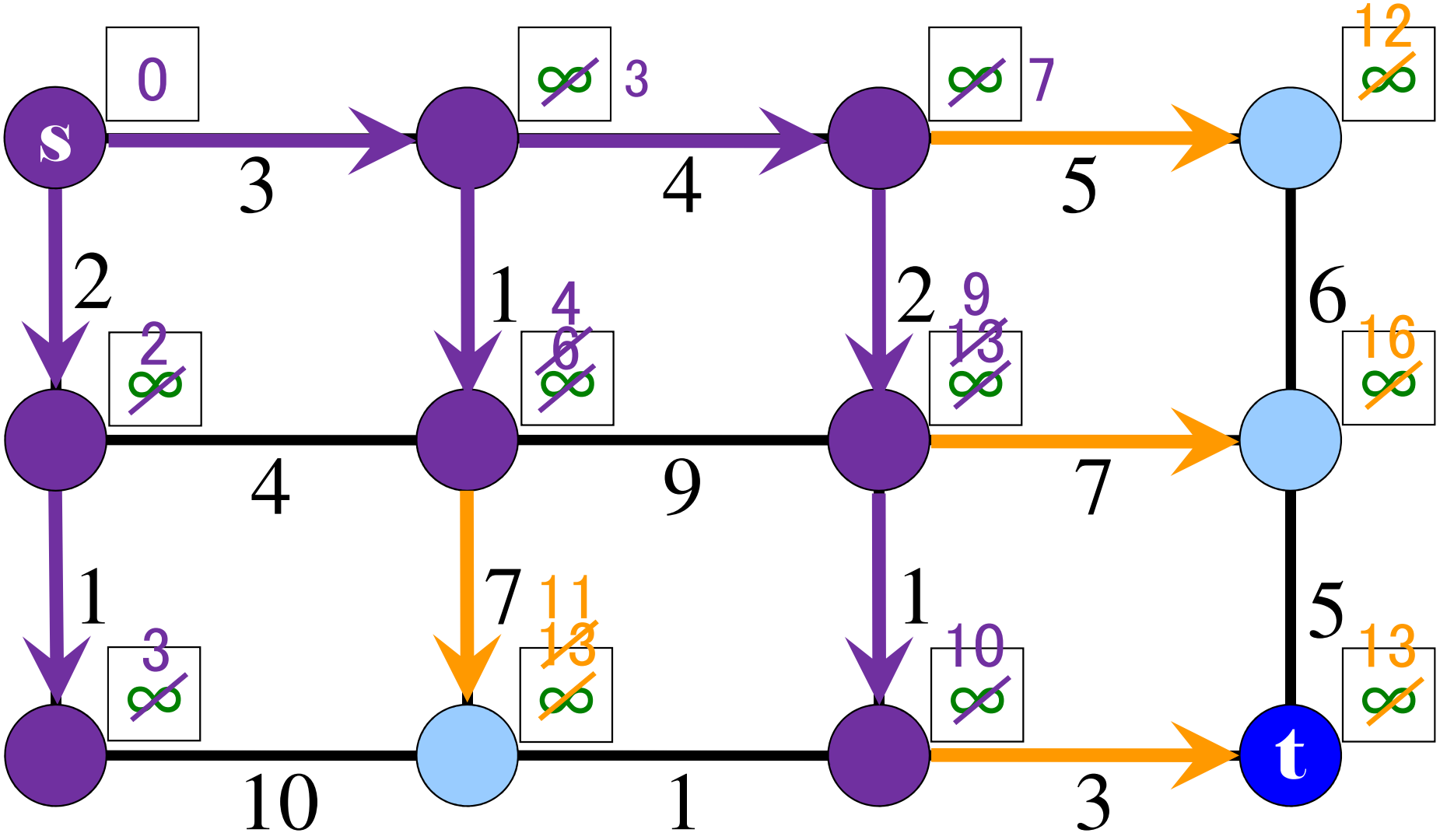


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

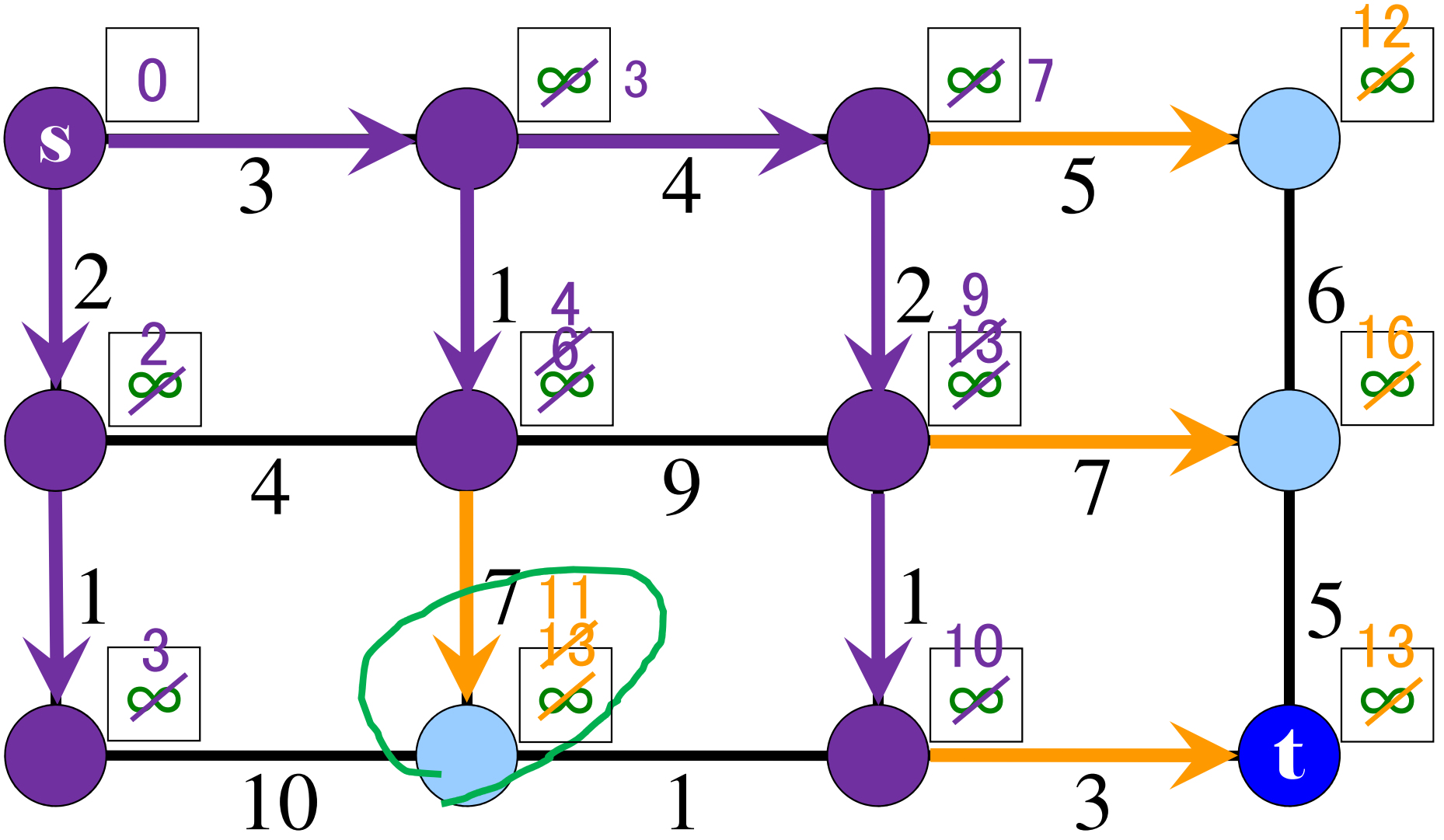
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



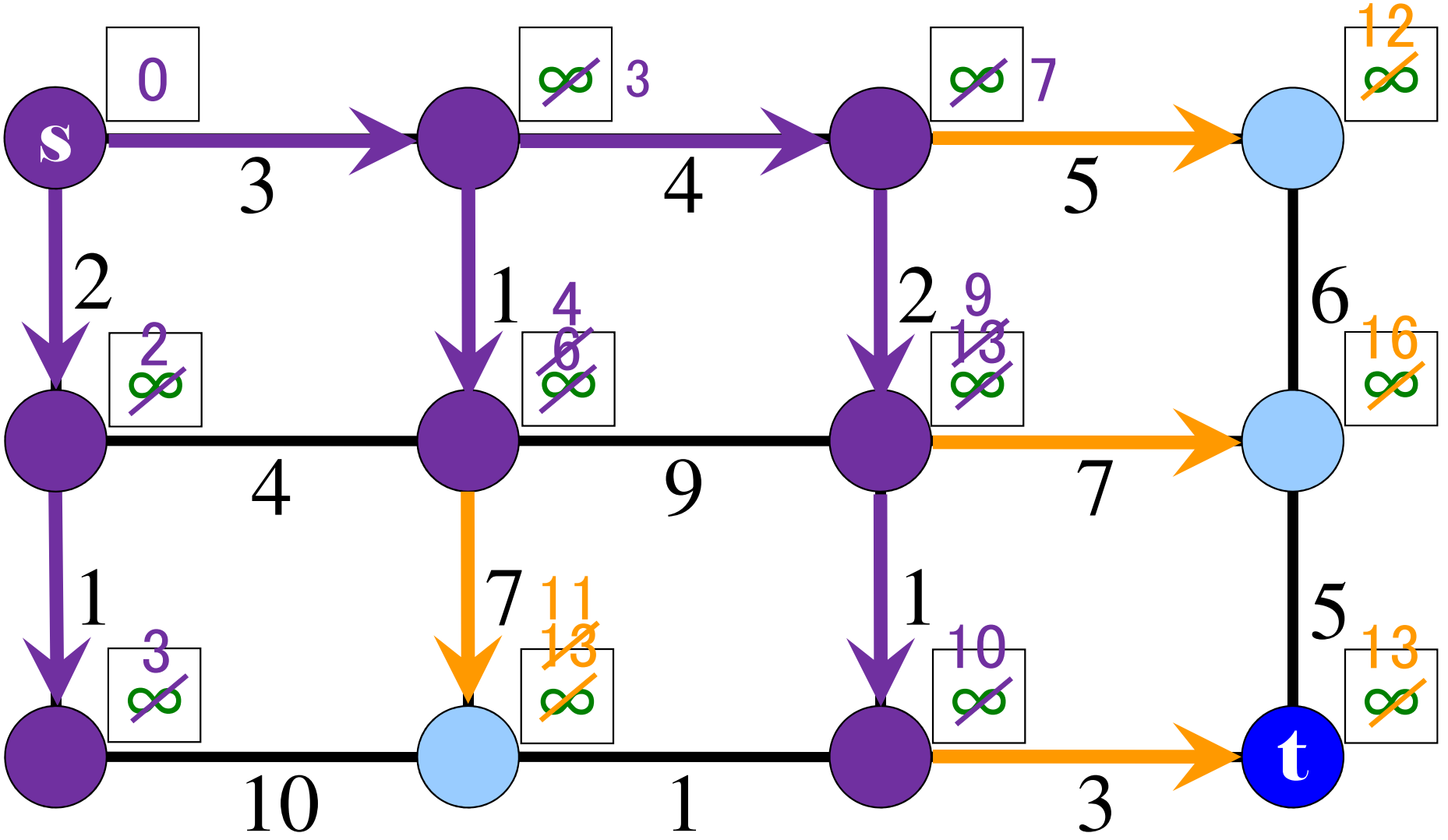
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

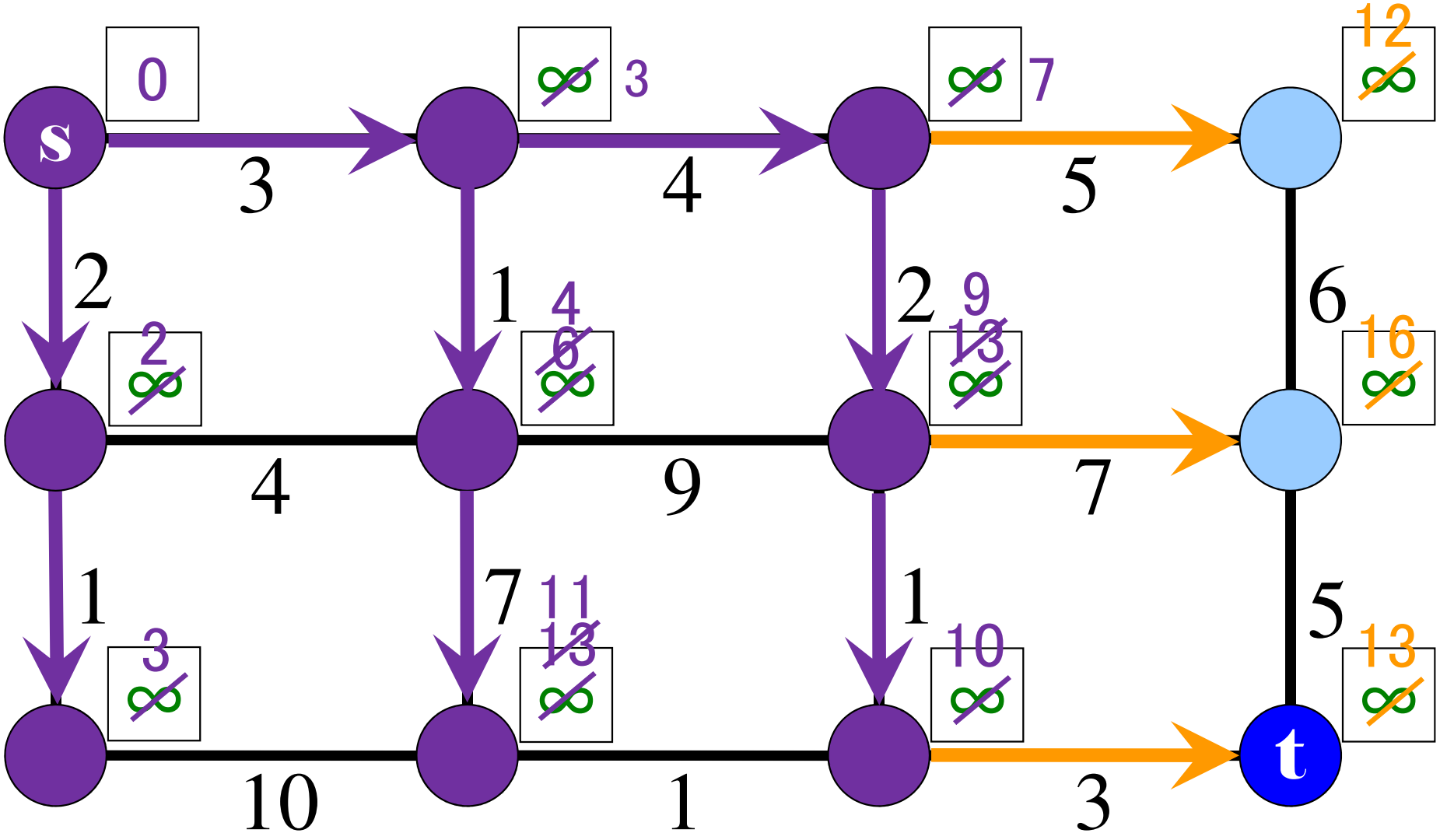


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

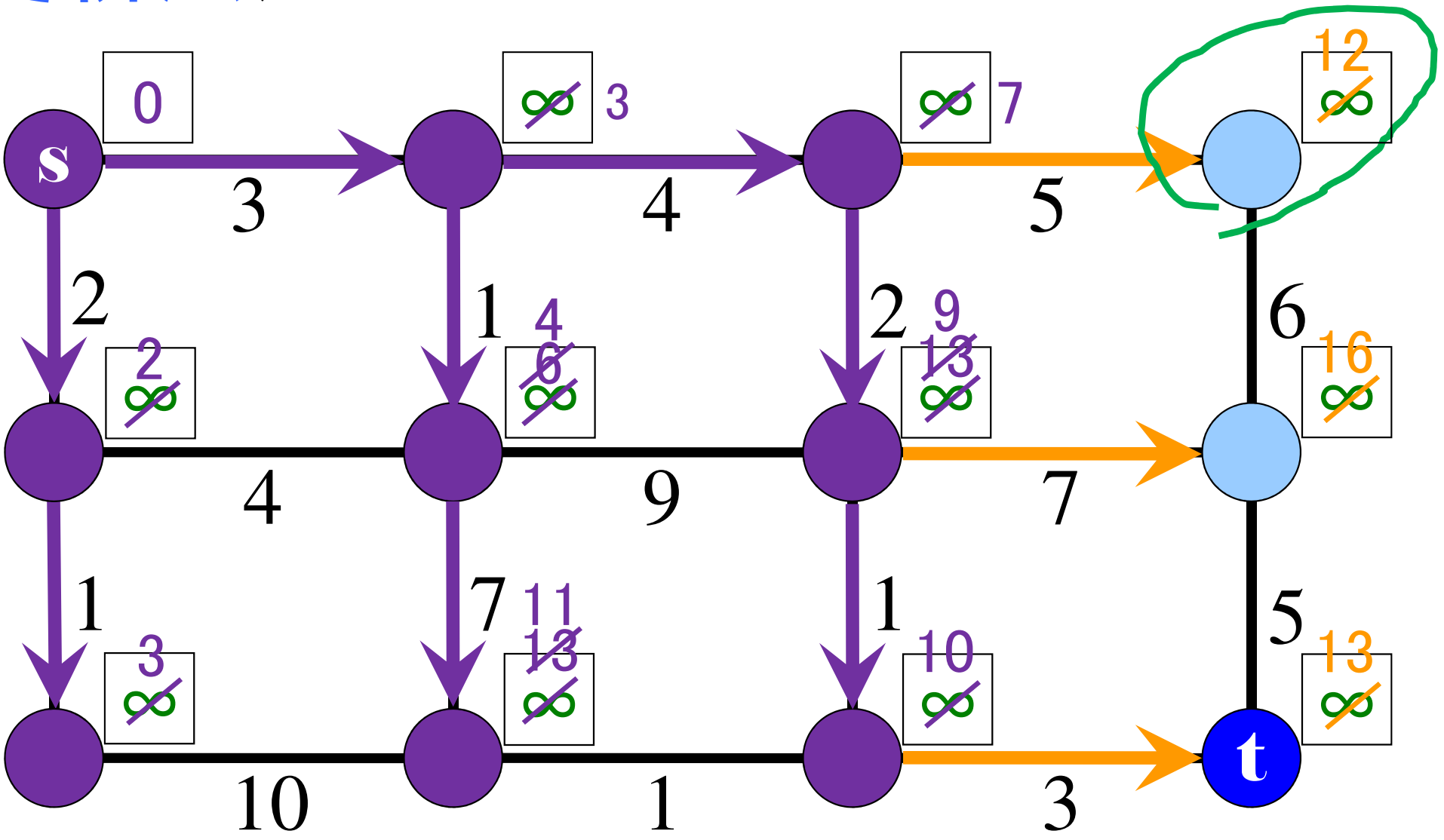
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



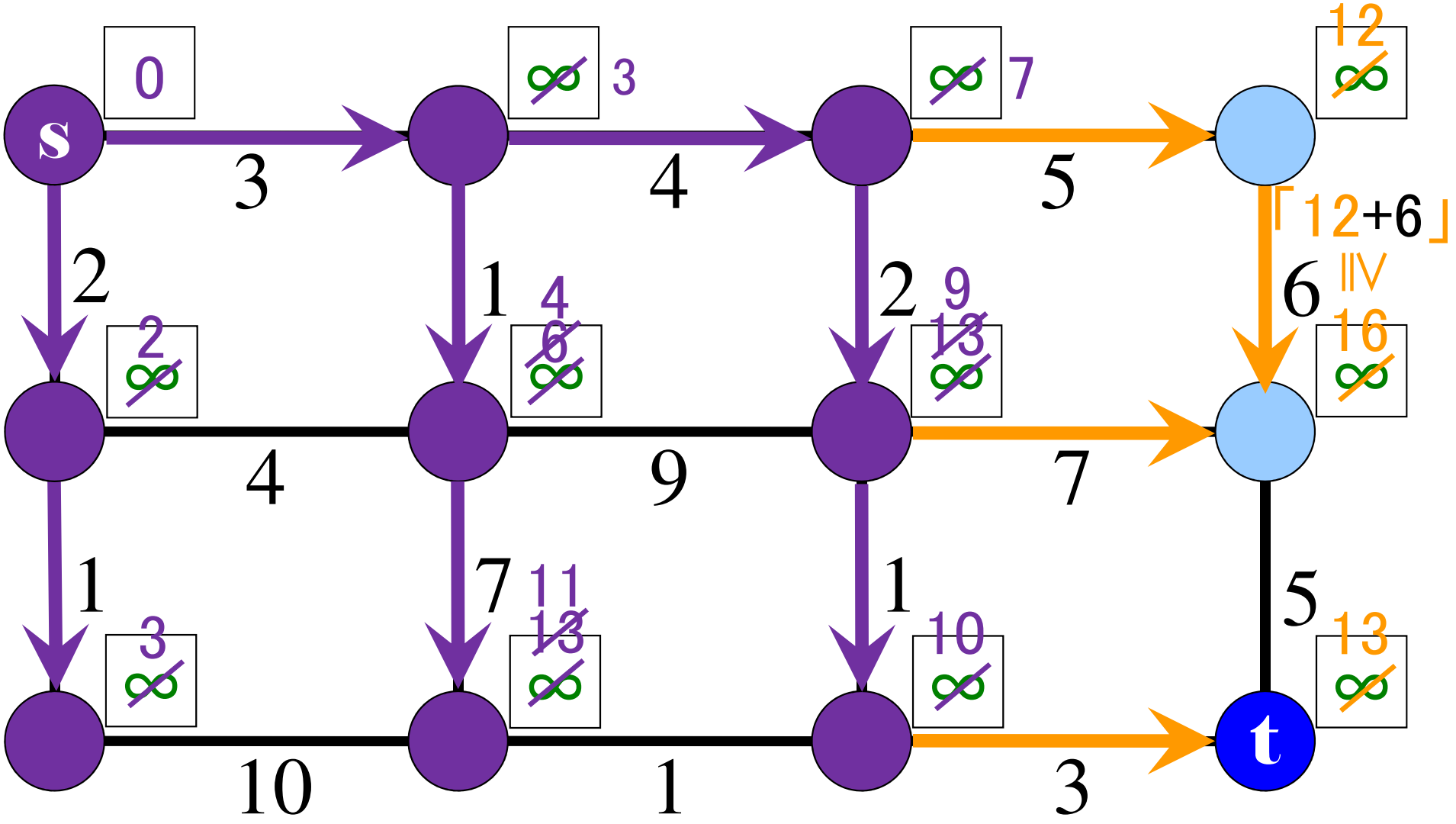
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

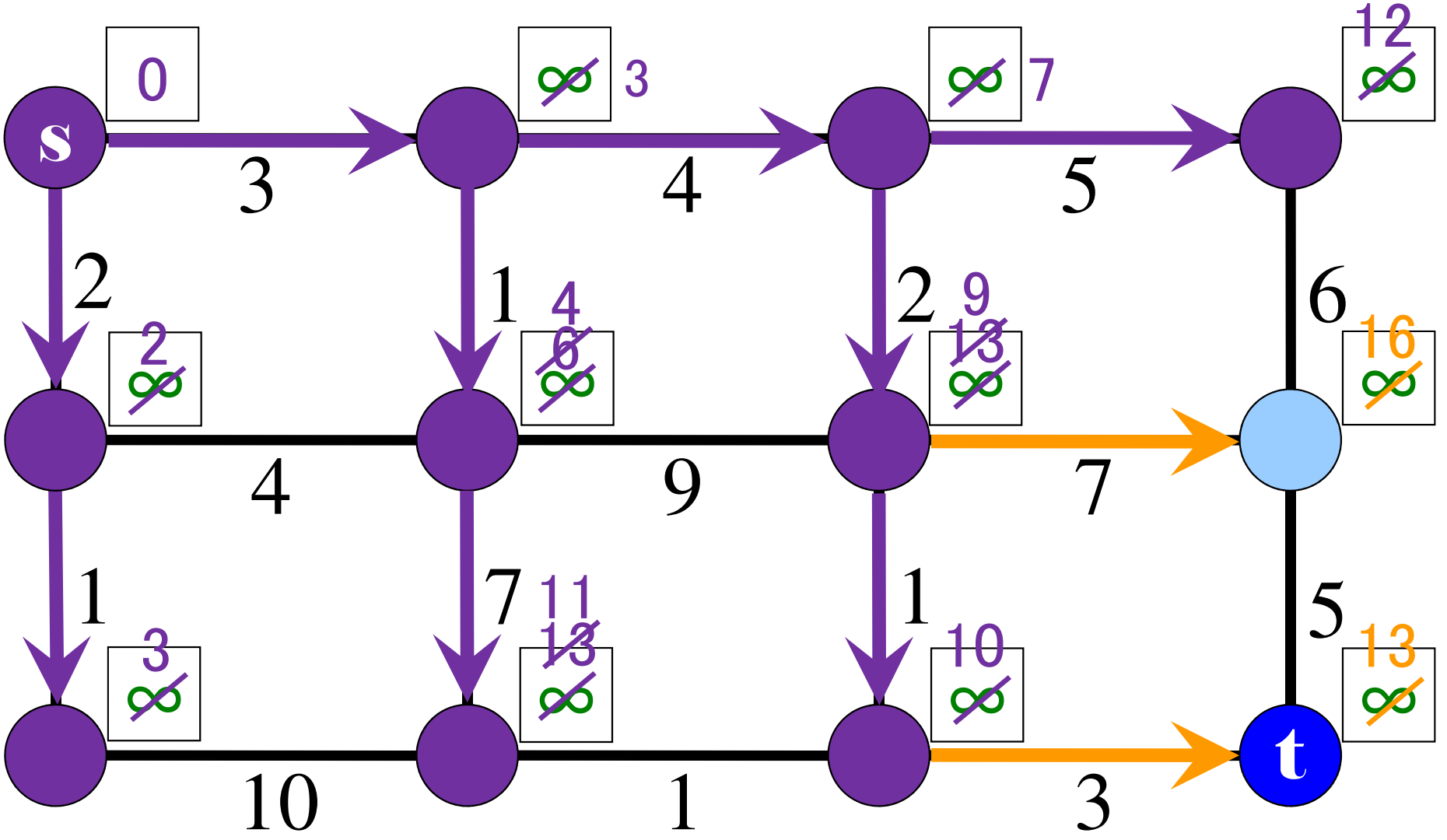


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

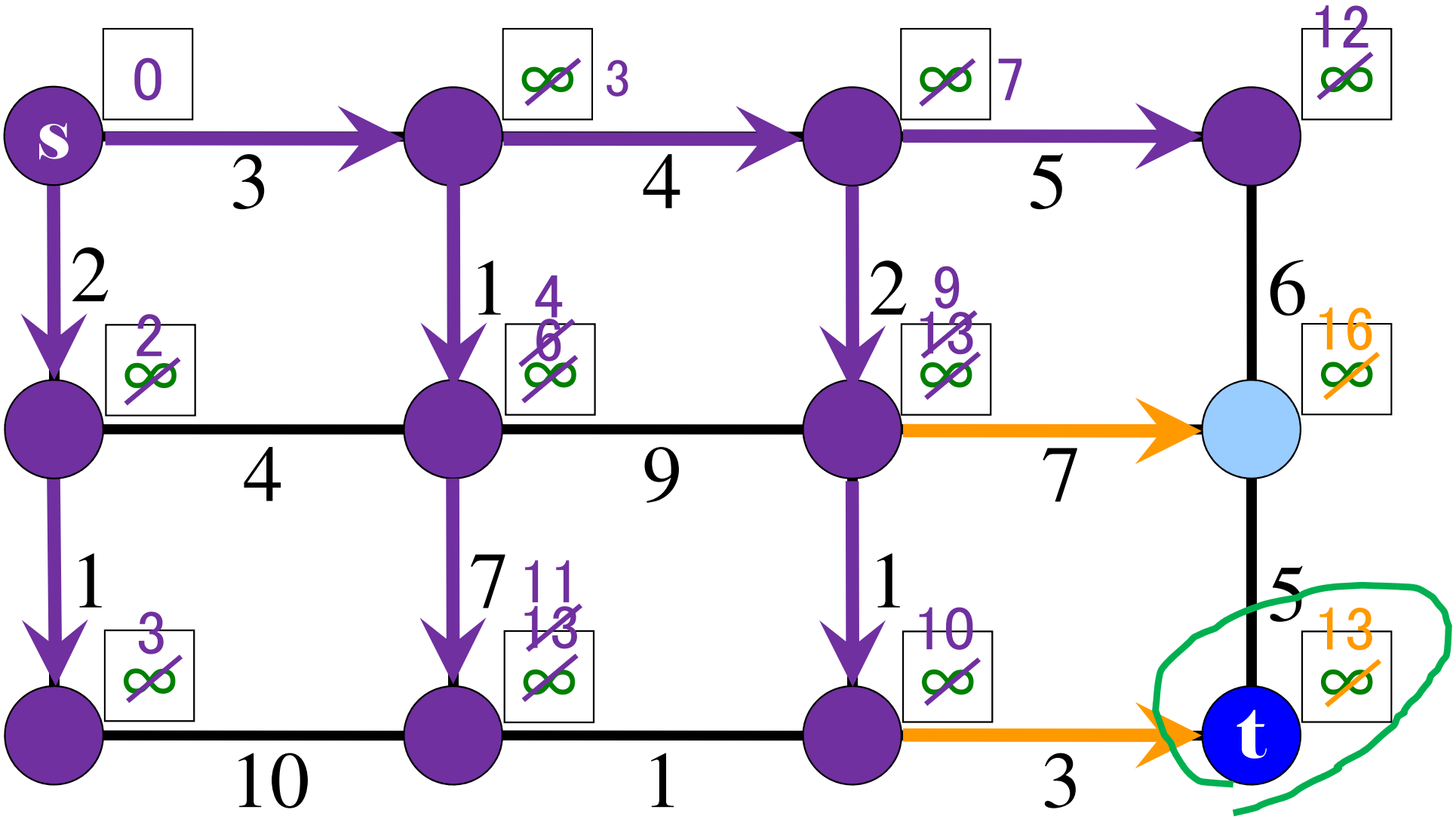
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



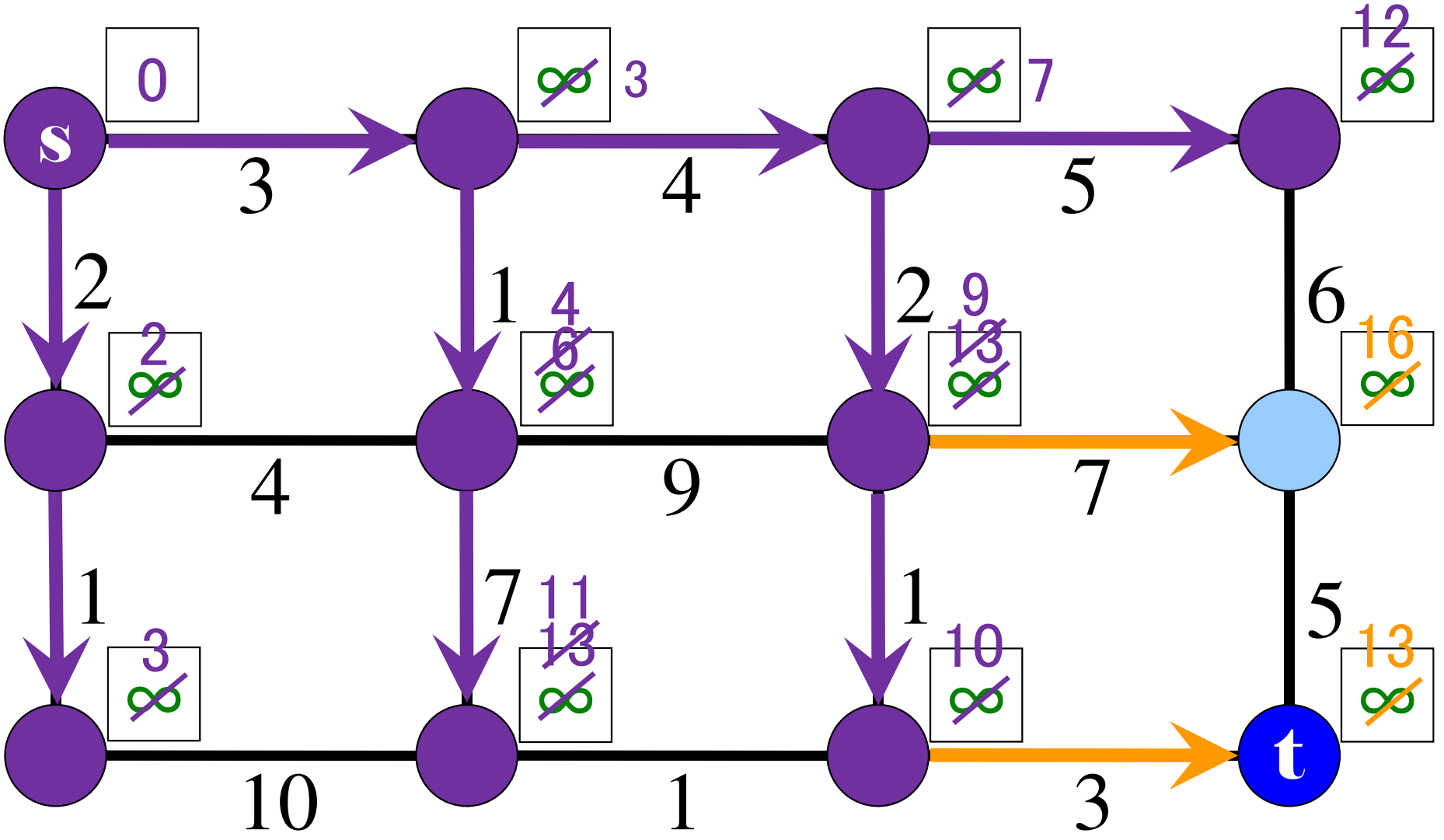
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

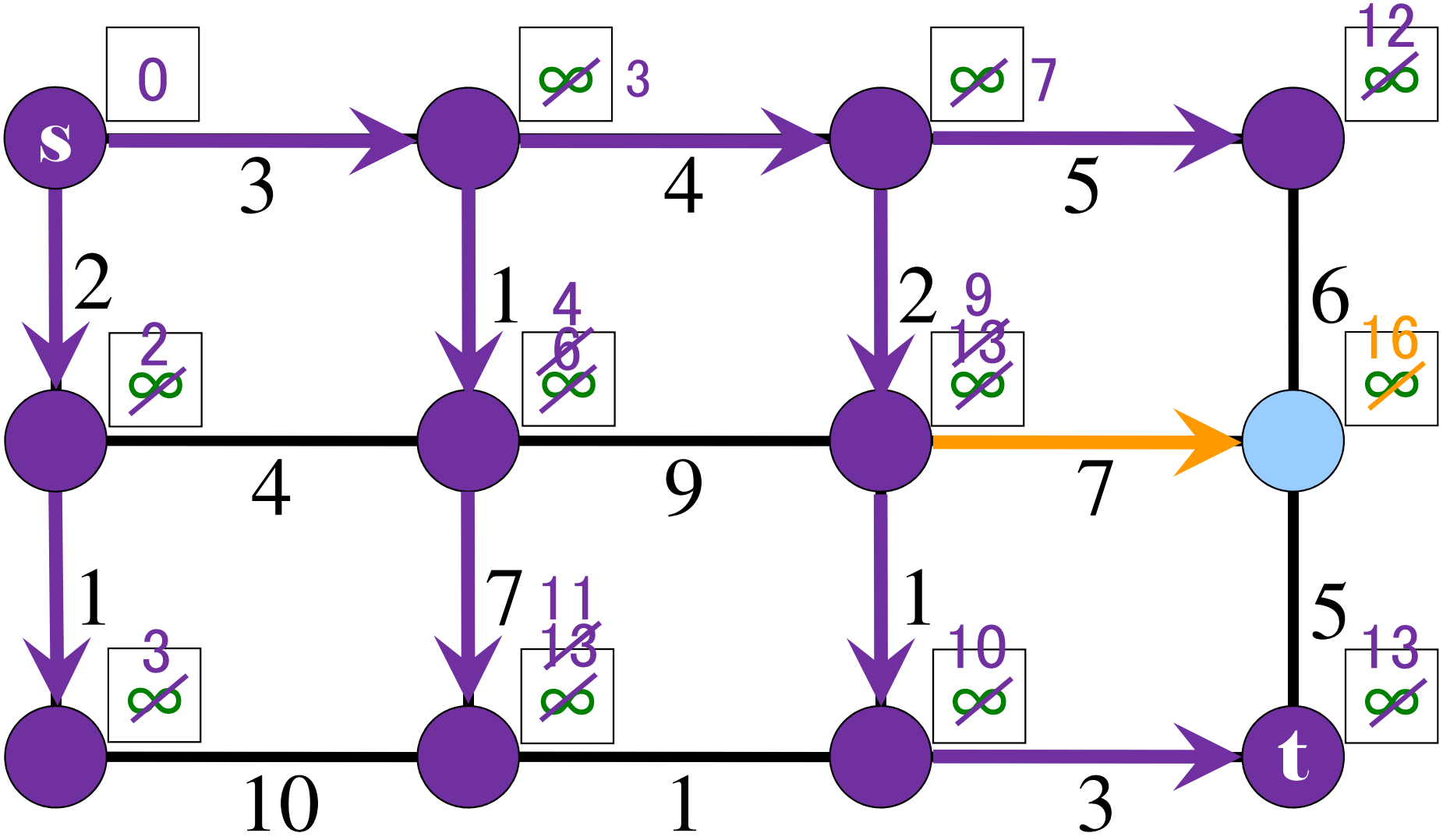


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

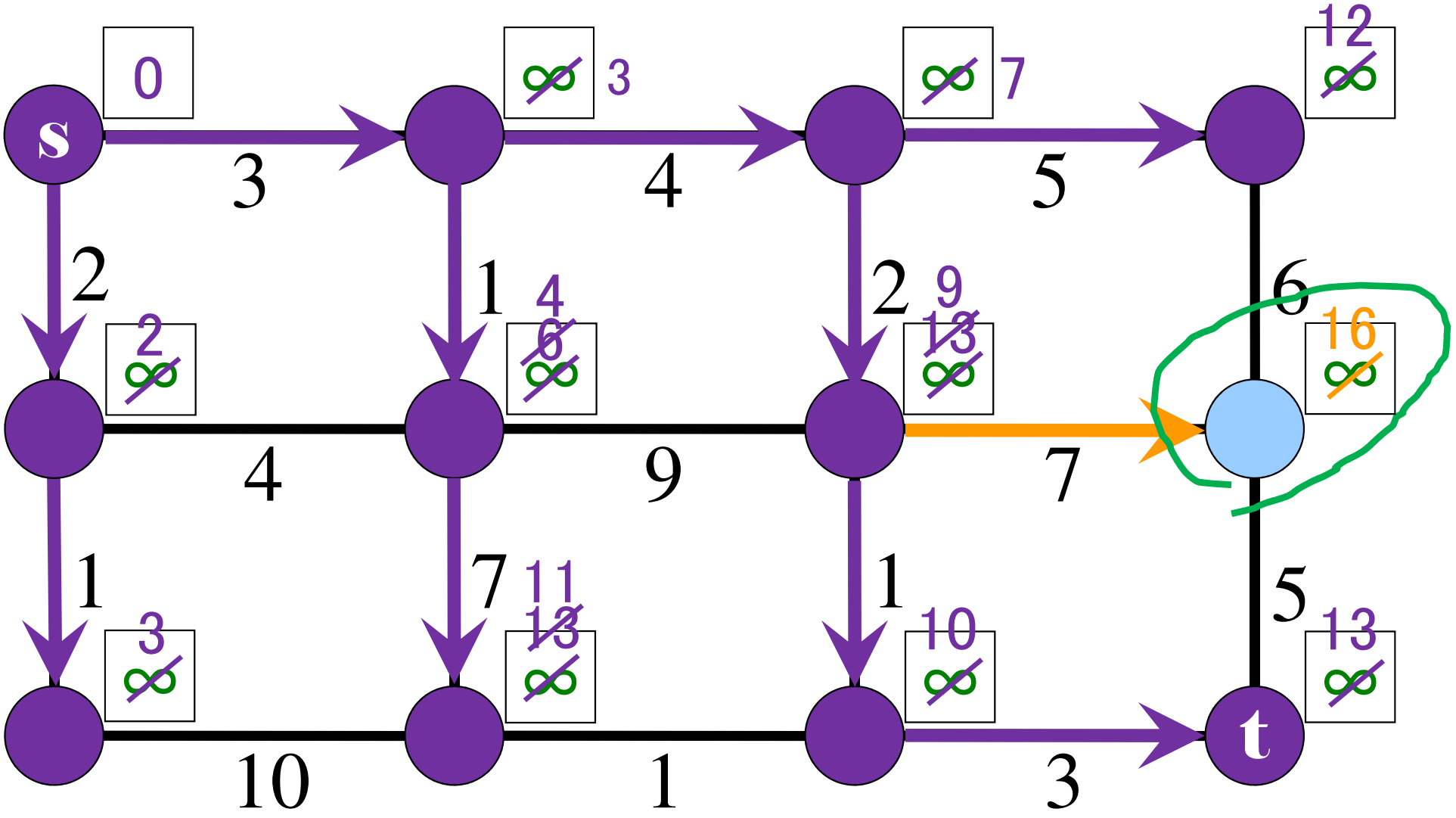
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



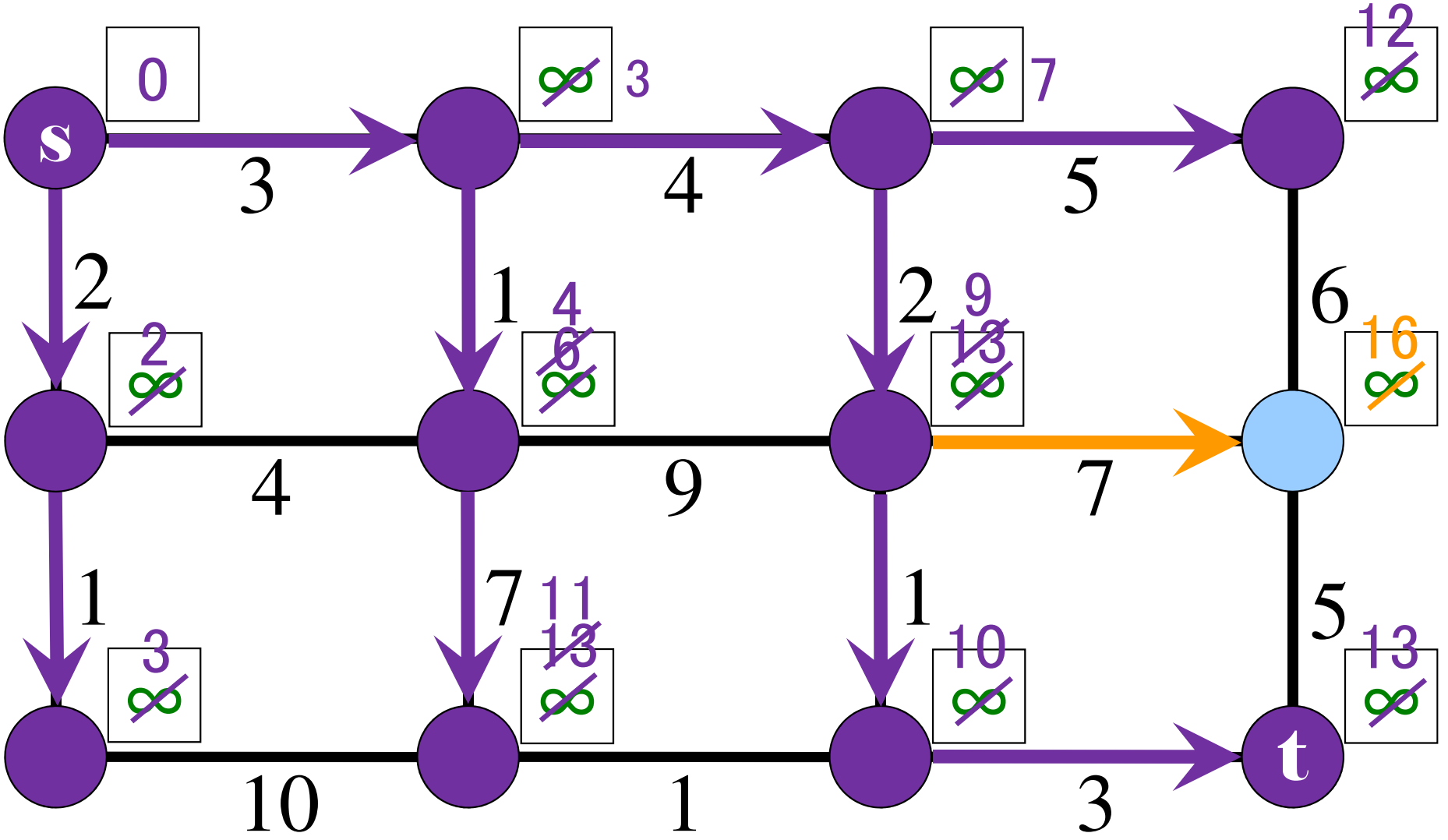
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

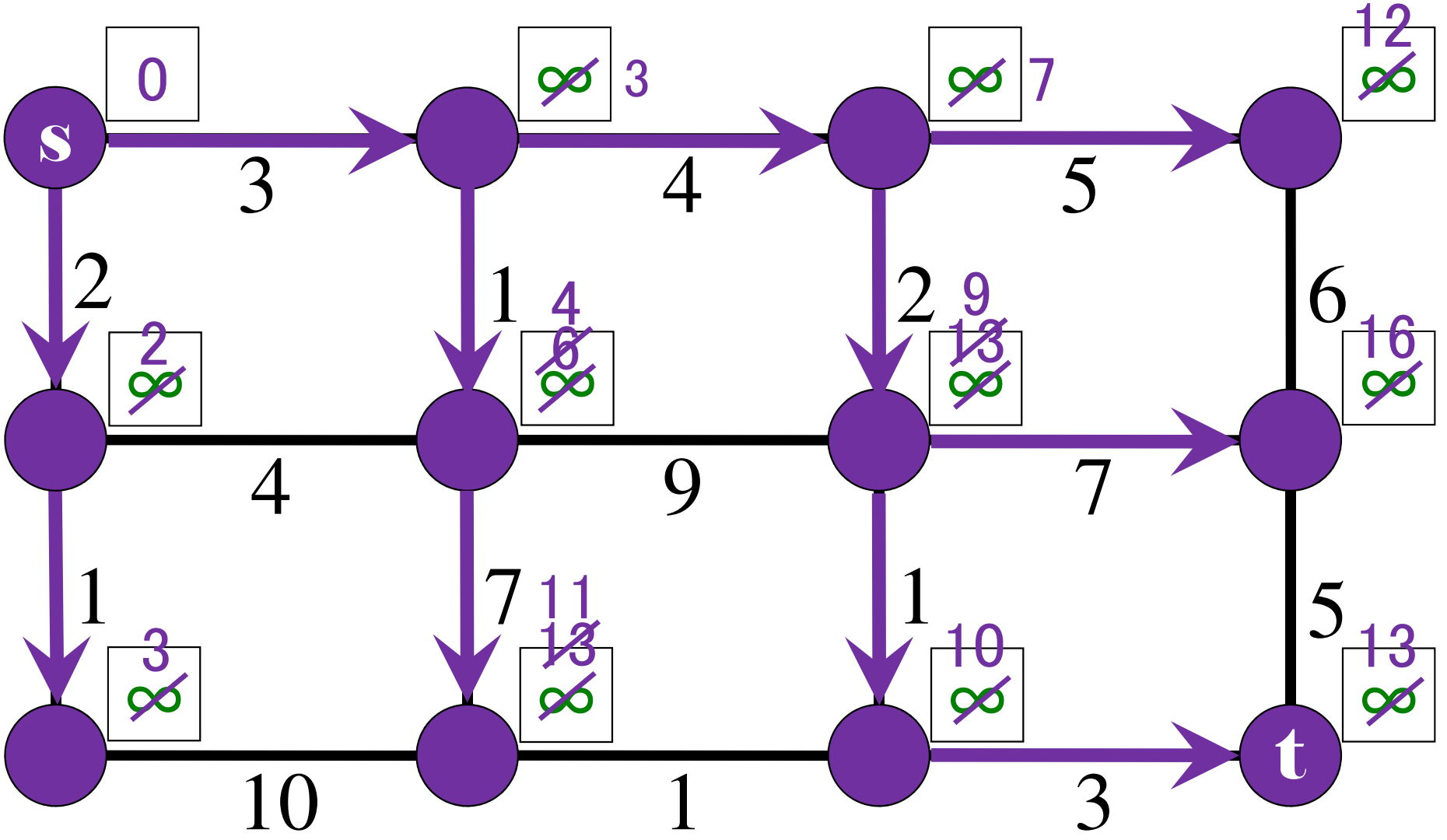


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

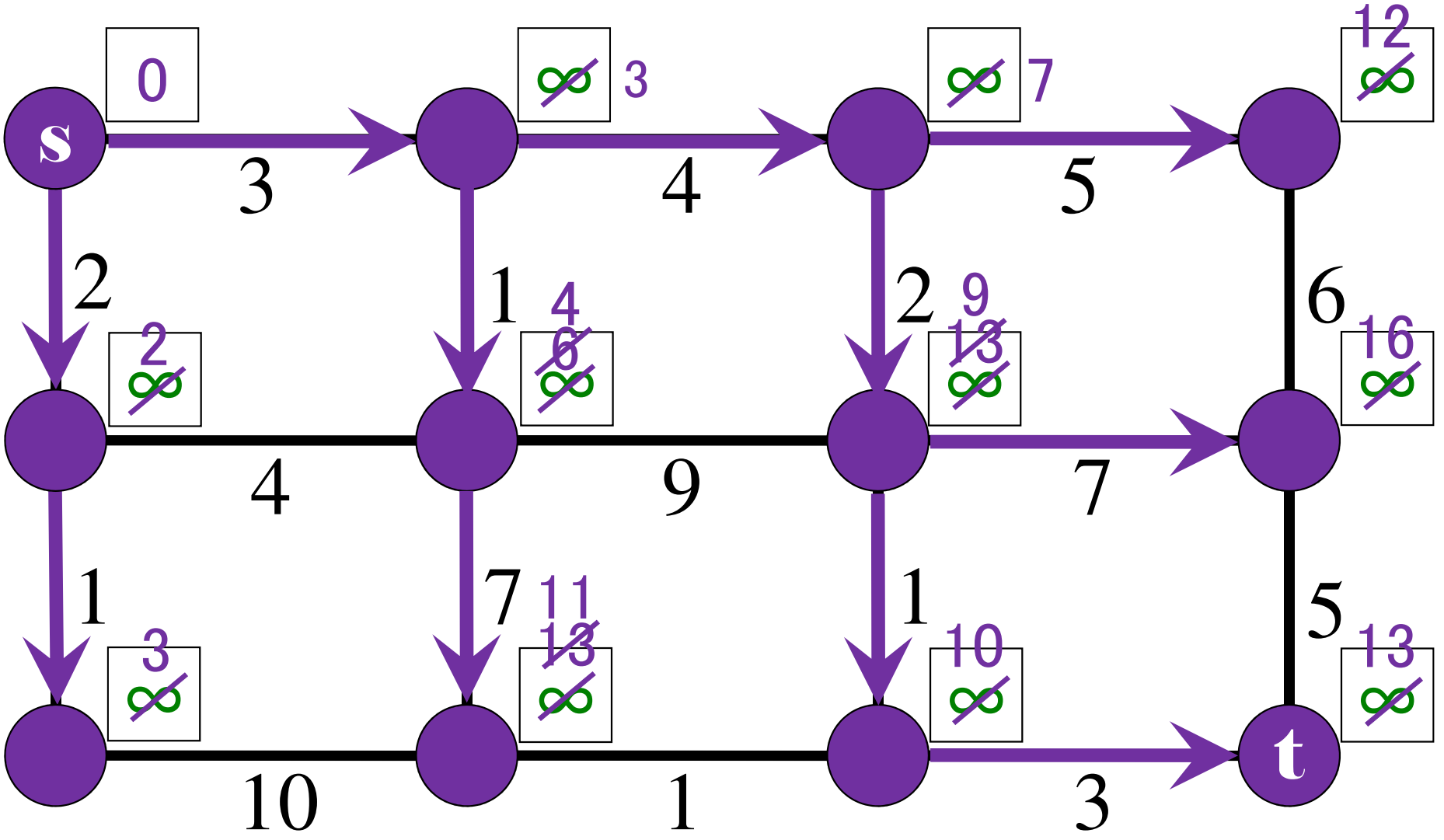
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



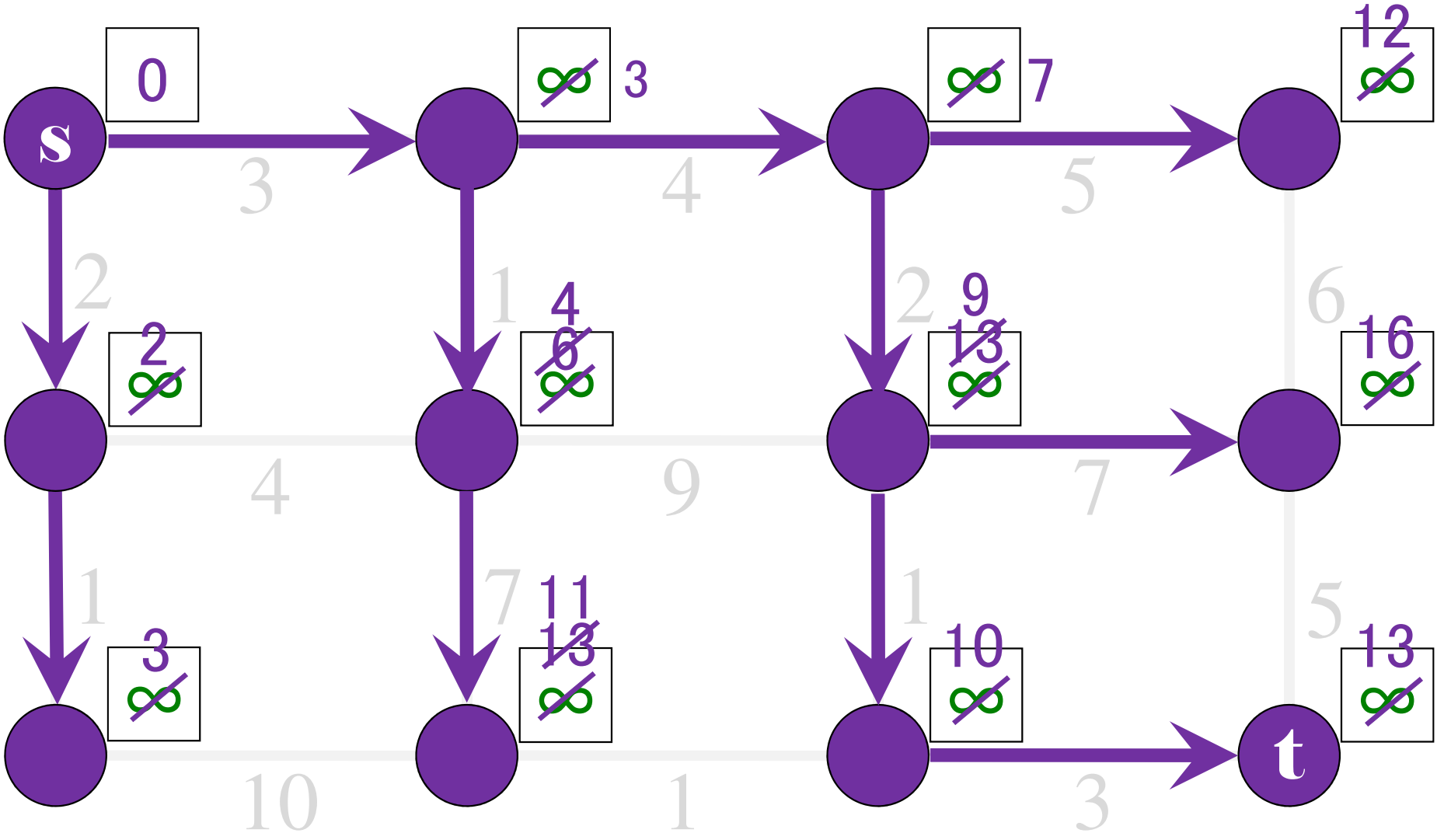
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



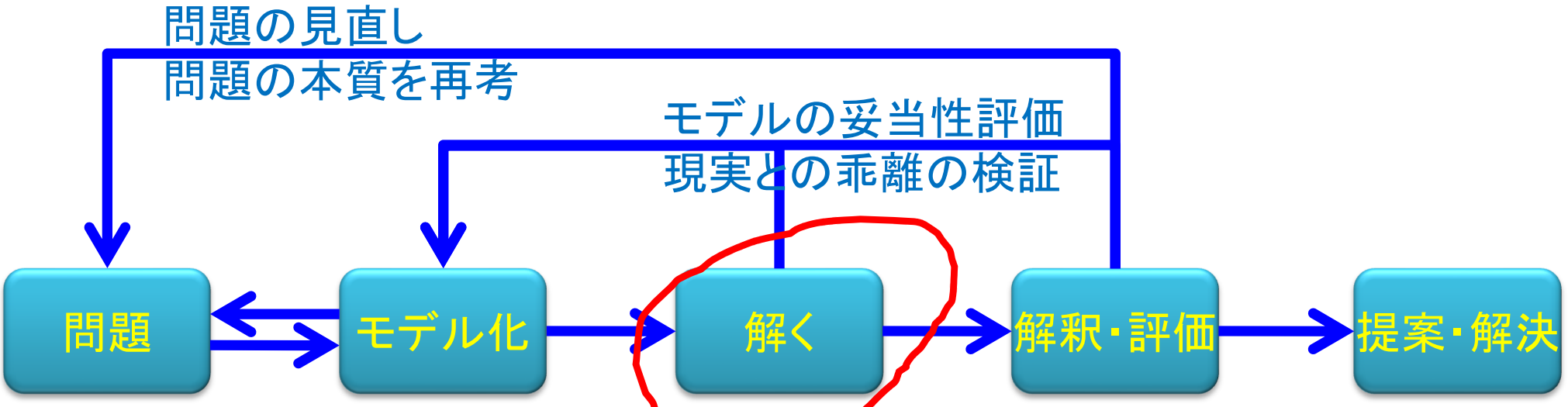
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

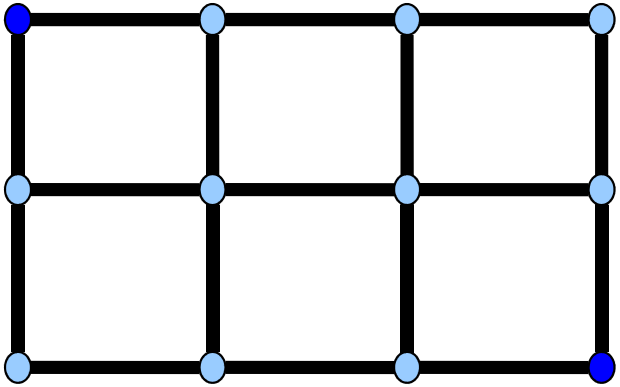
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

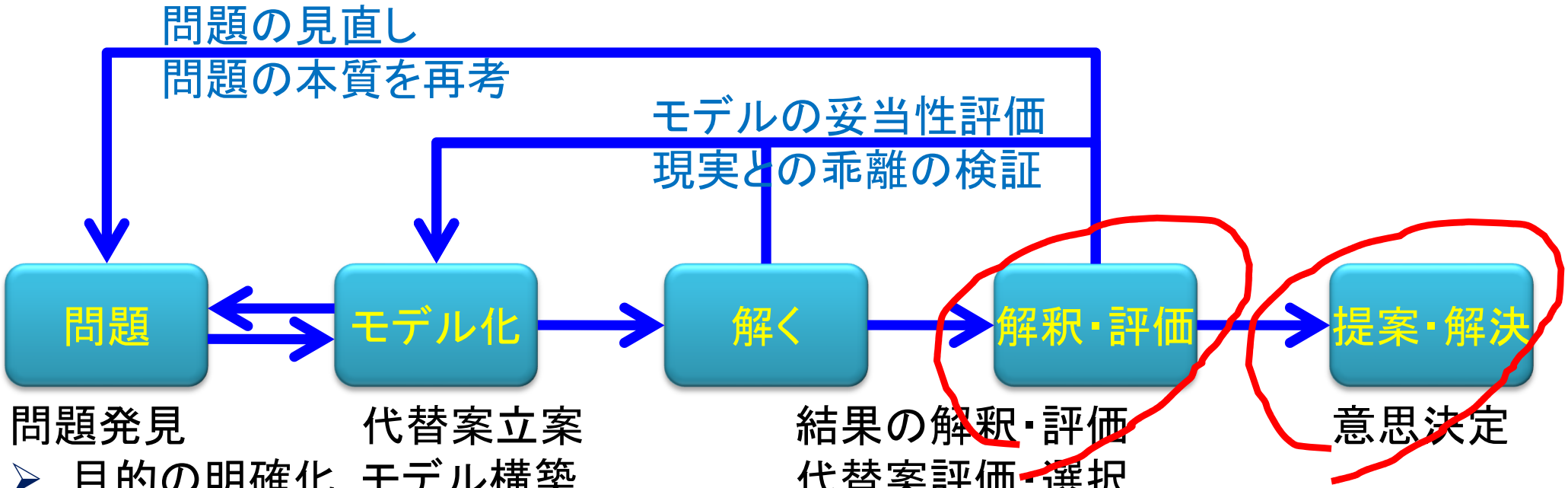
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

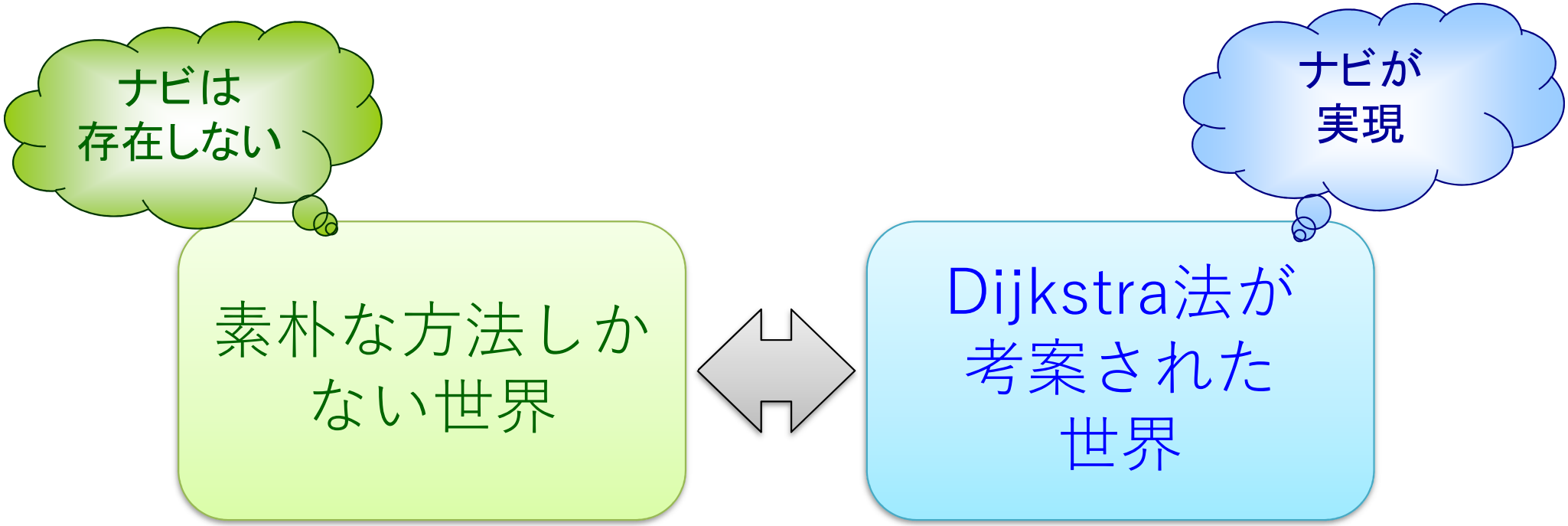
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

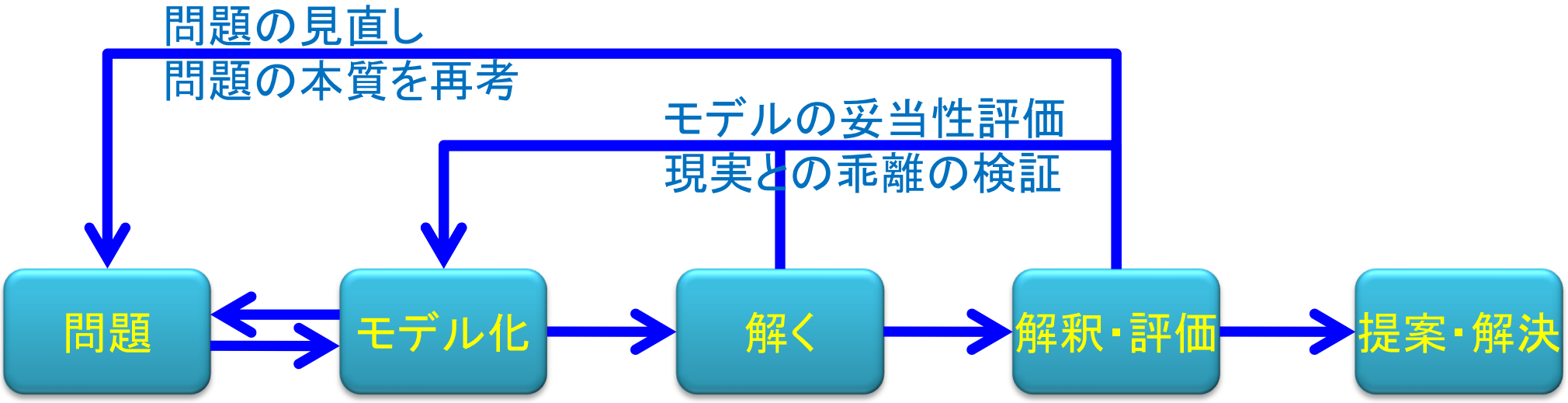
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



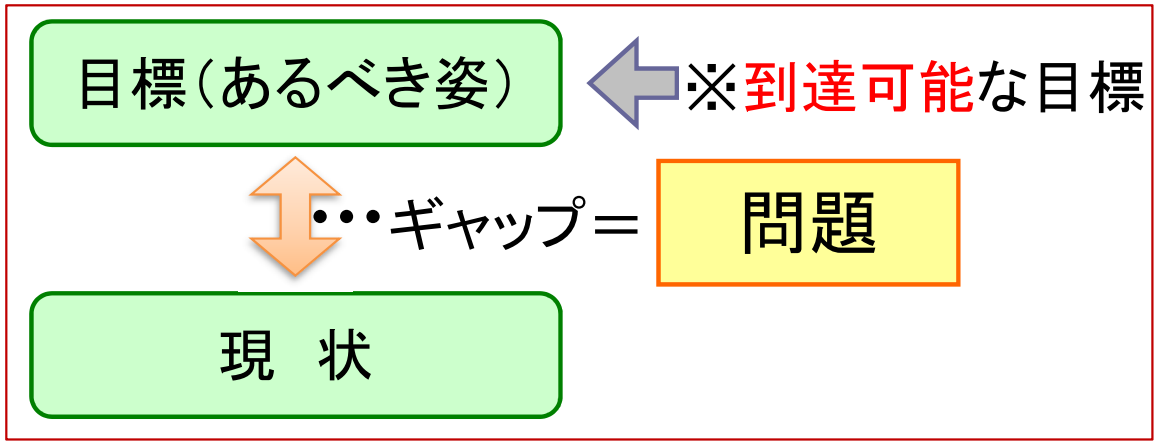
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

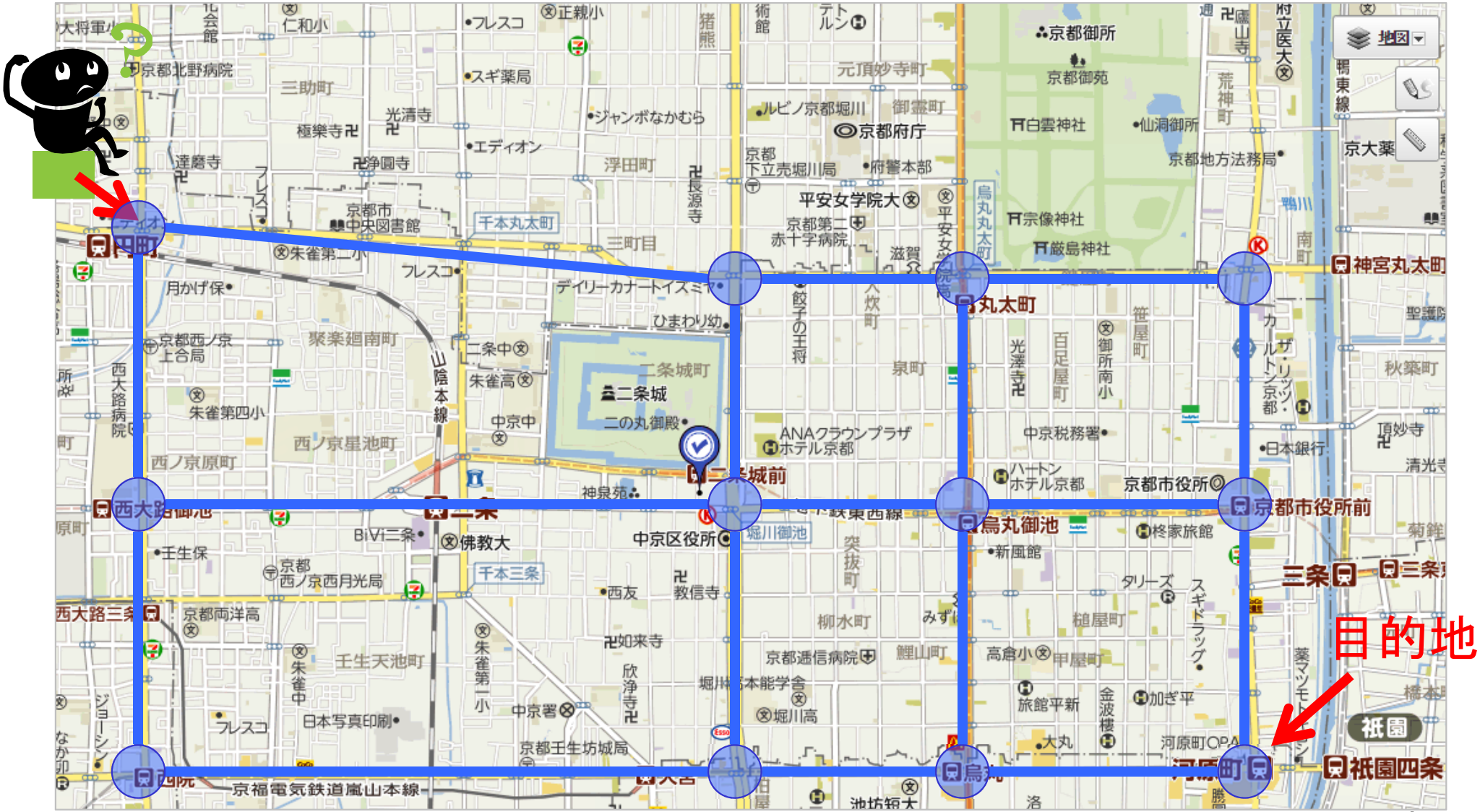
意思決定

問題の定義



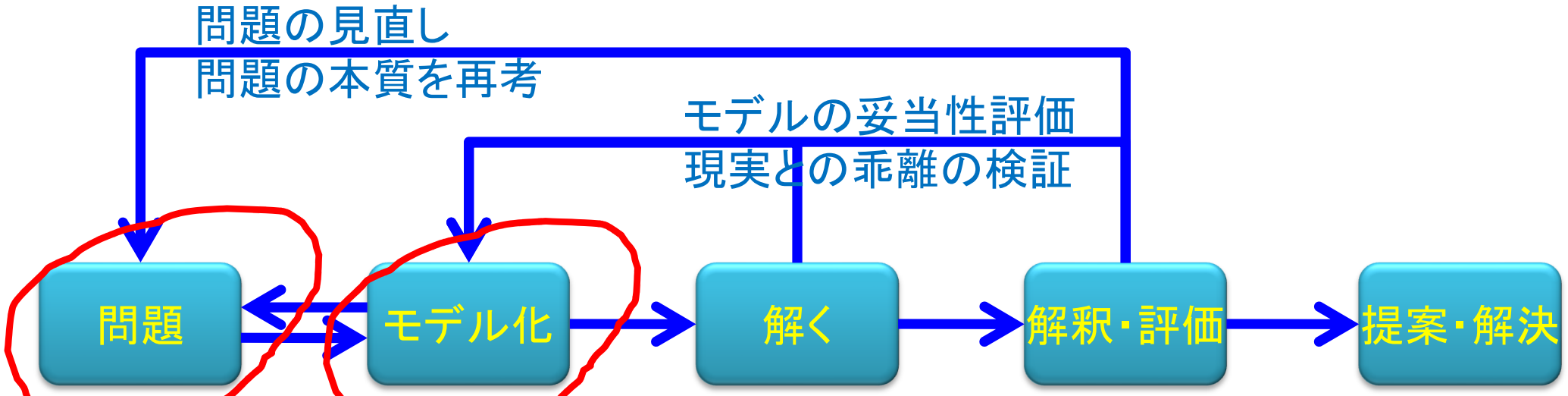
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

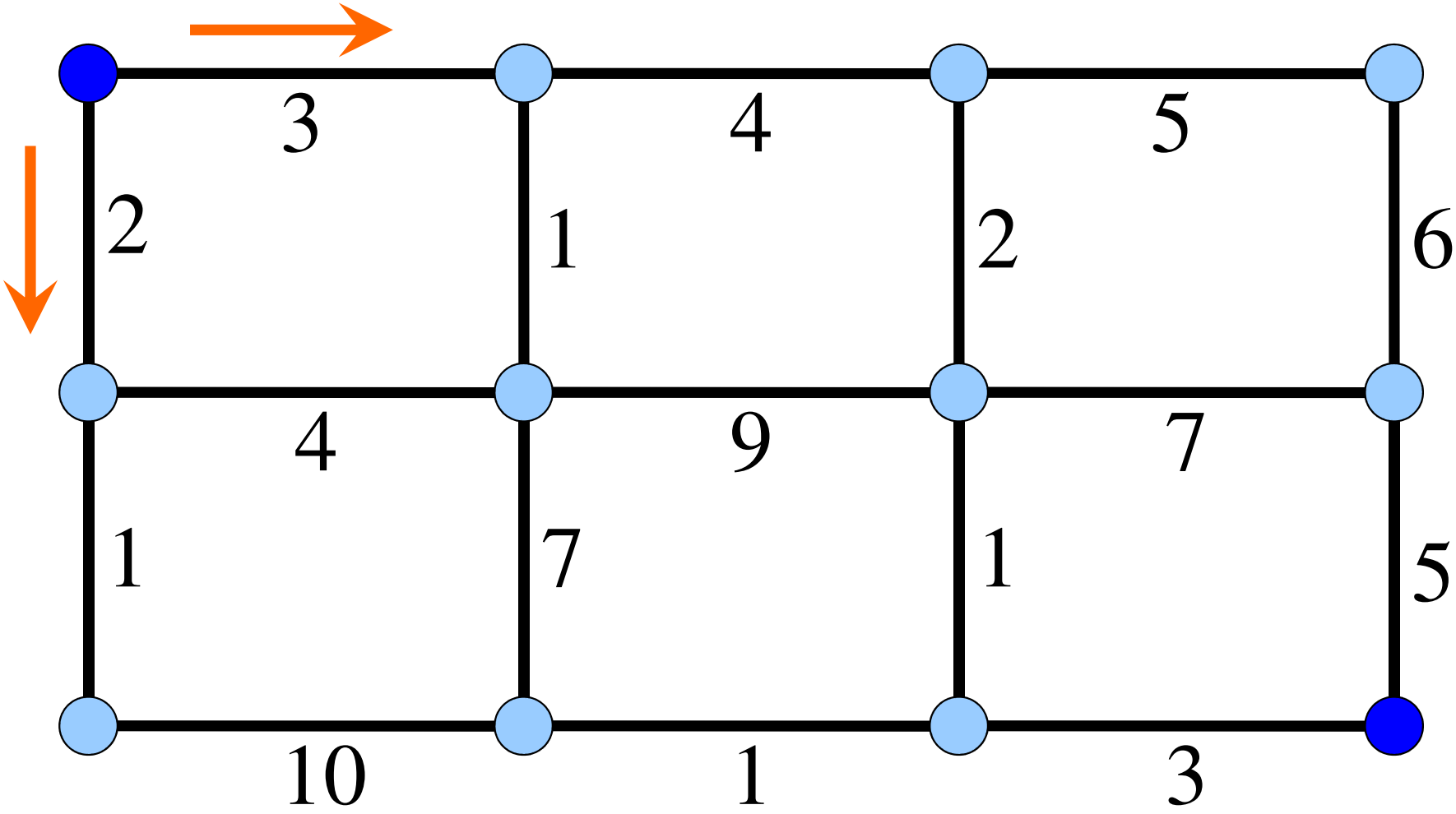
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

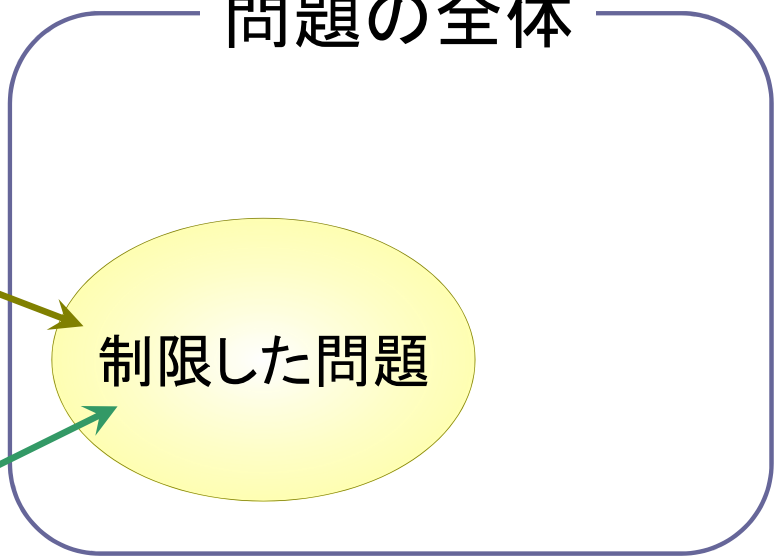


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

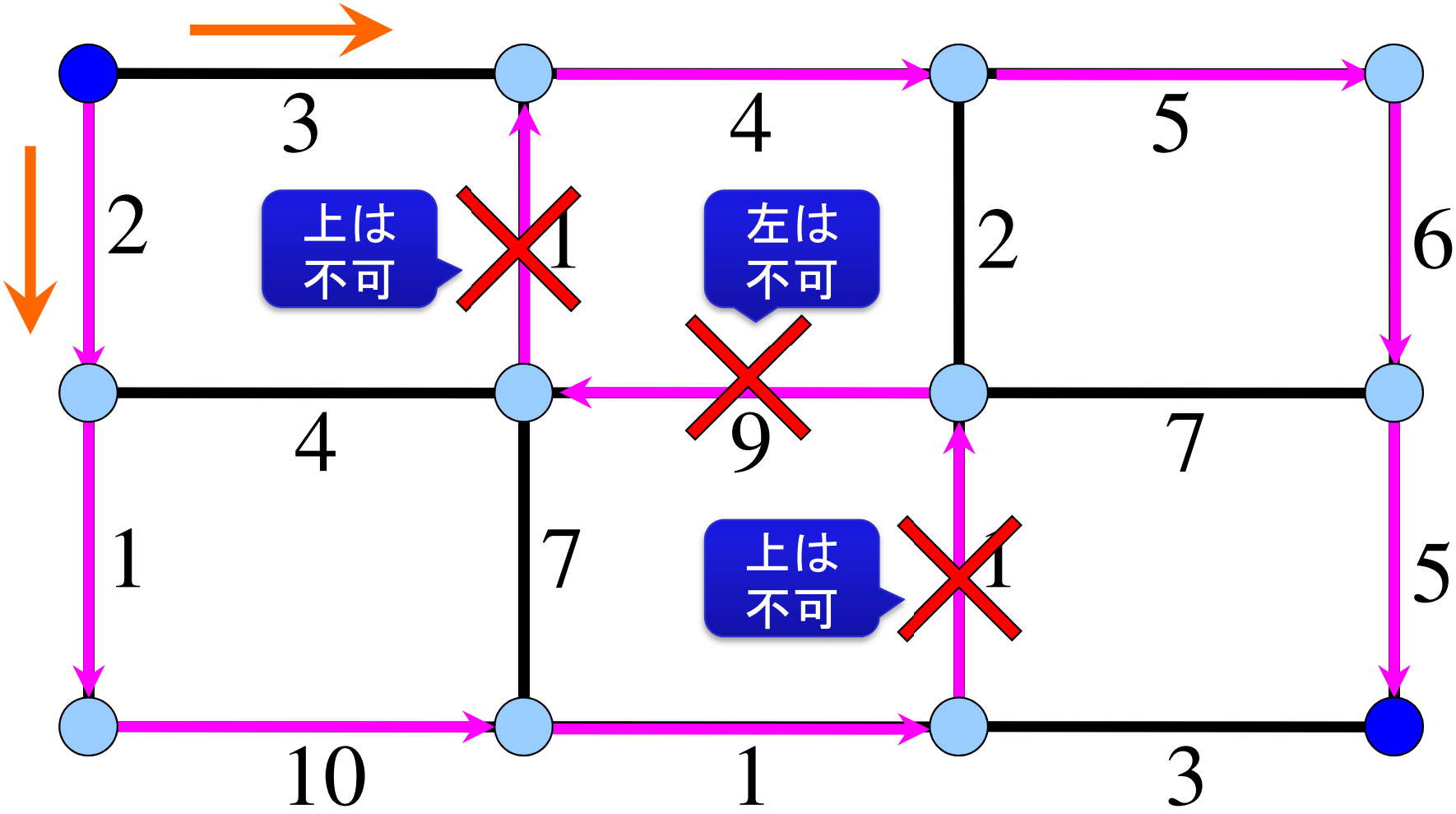
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

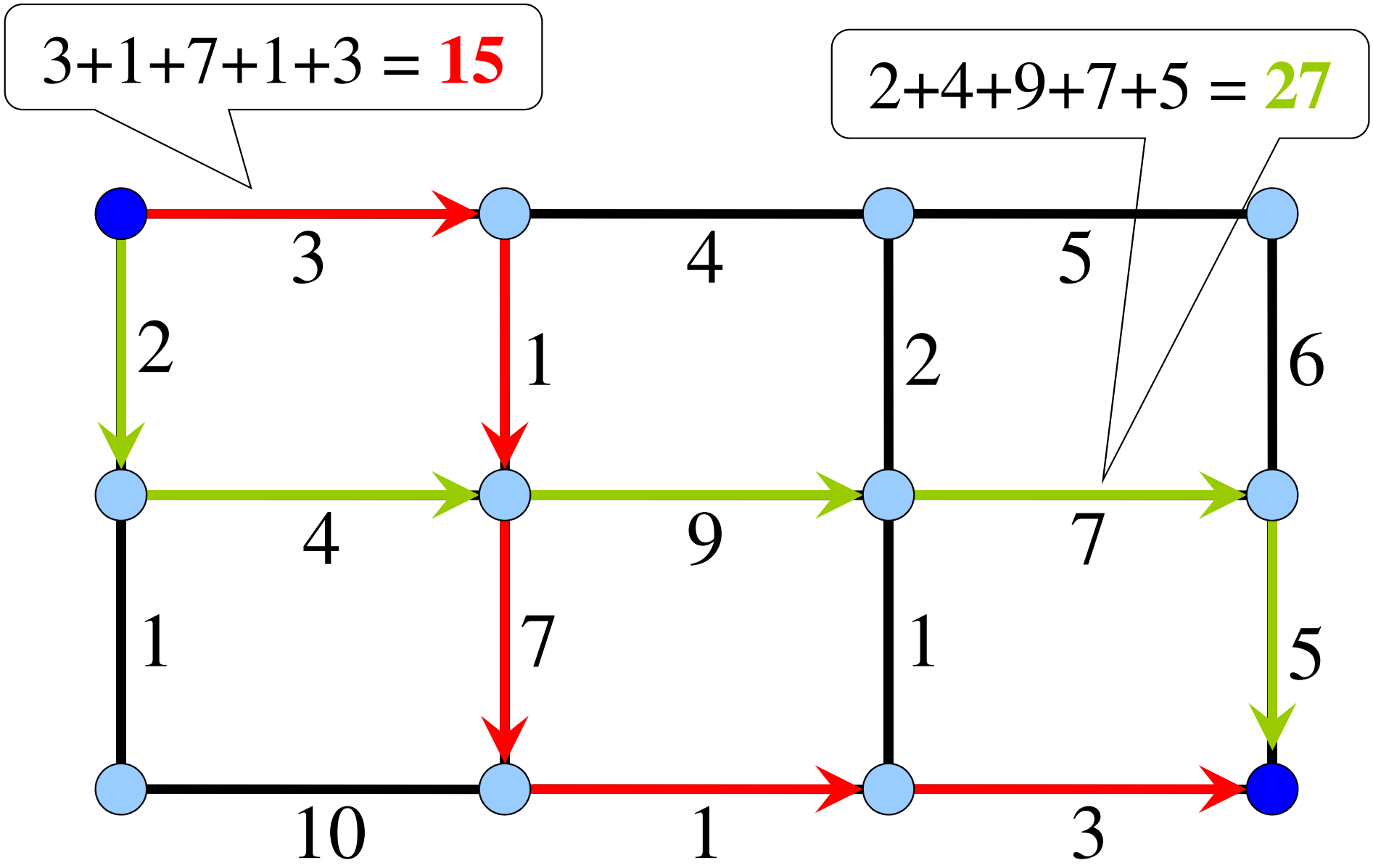
難しいなら易しくすればいいのさ！

制限した問題

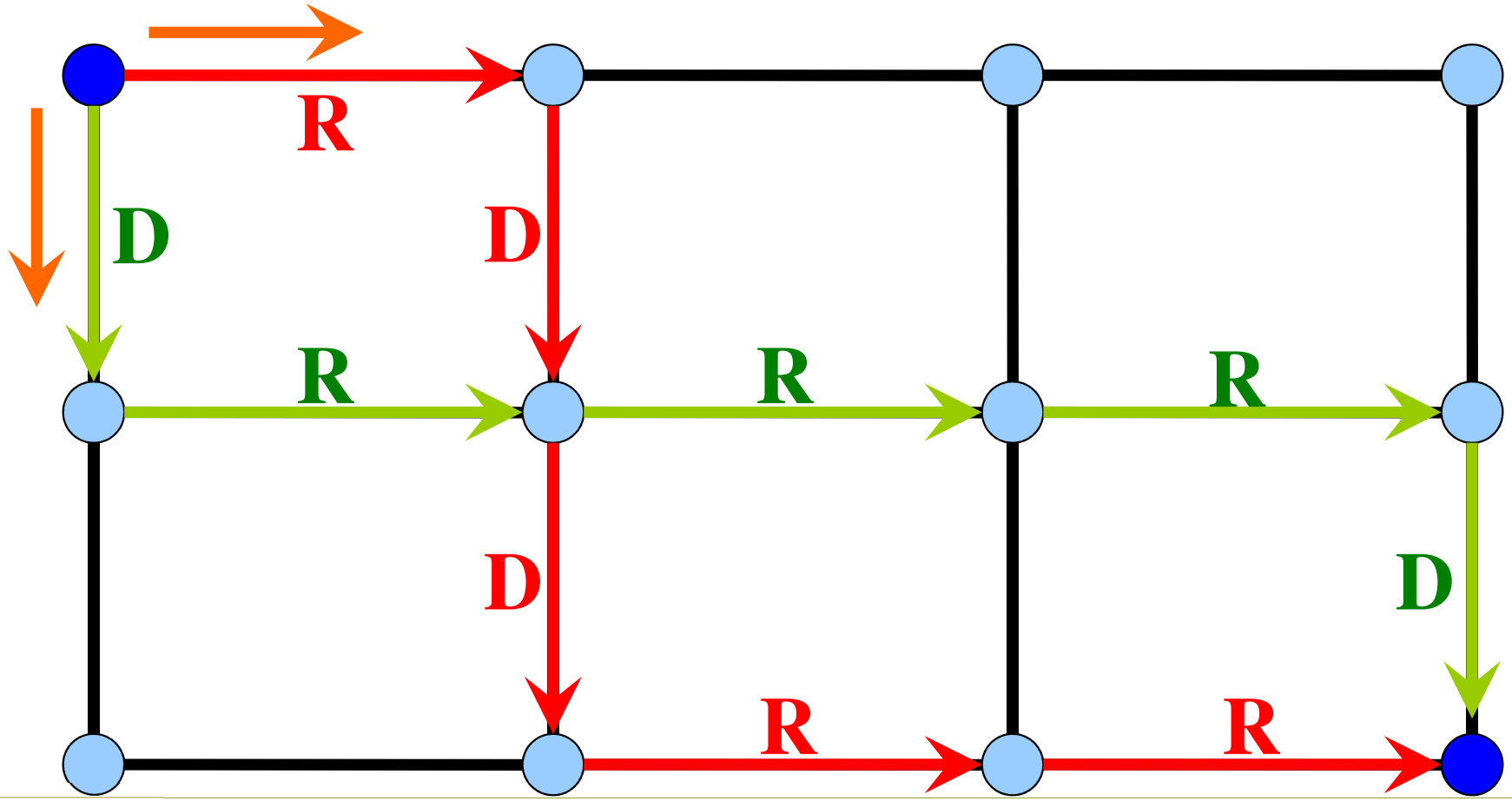
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



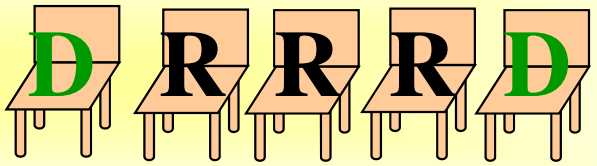
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



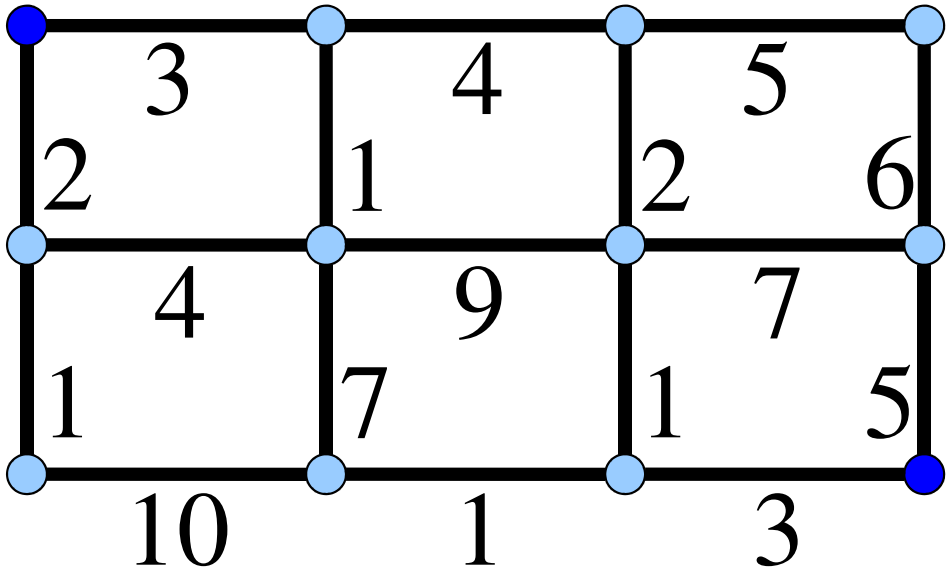
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

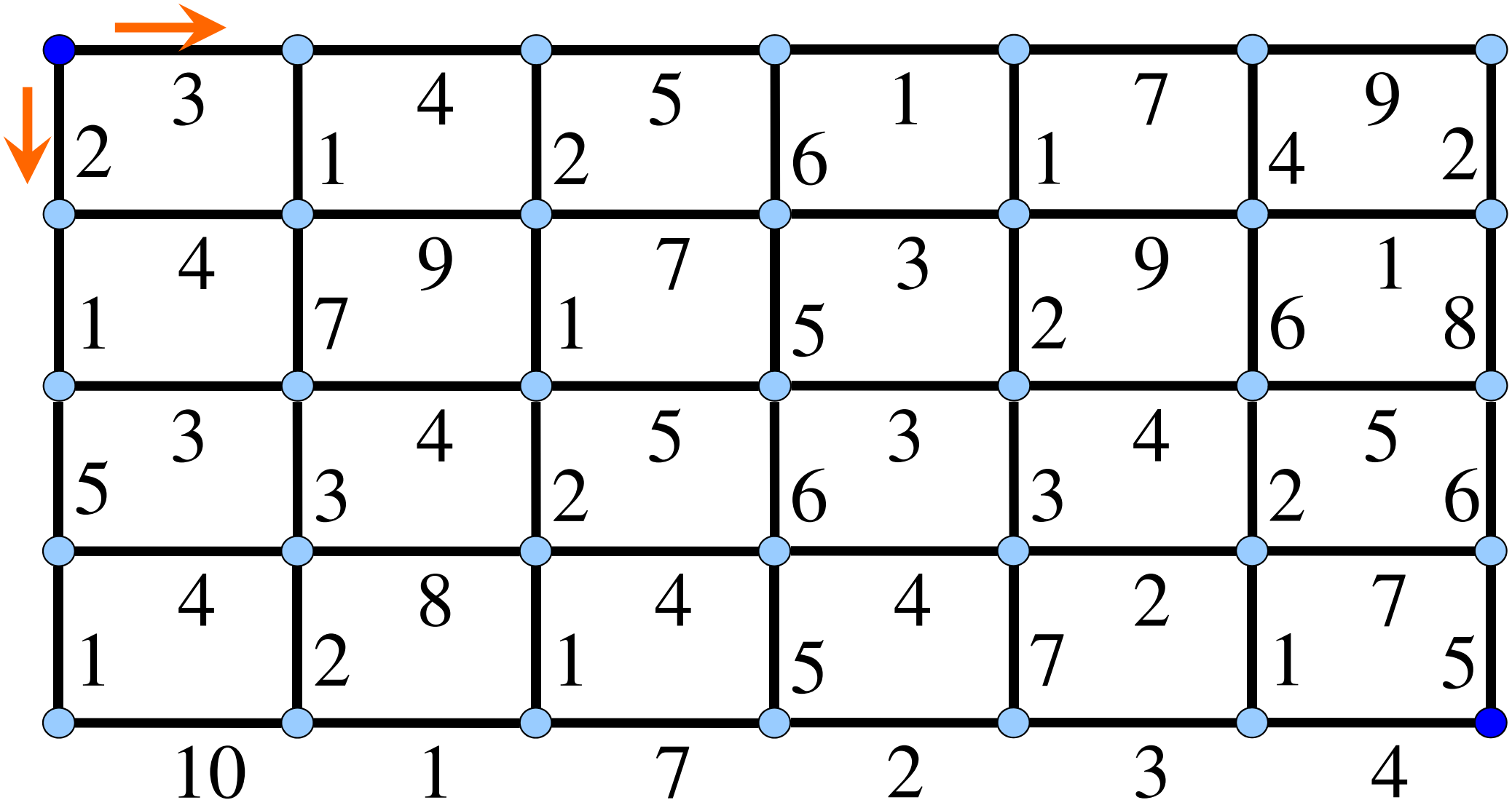
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

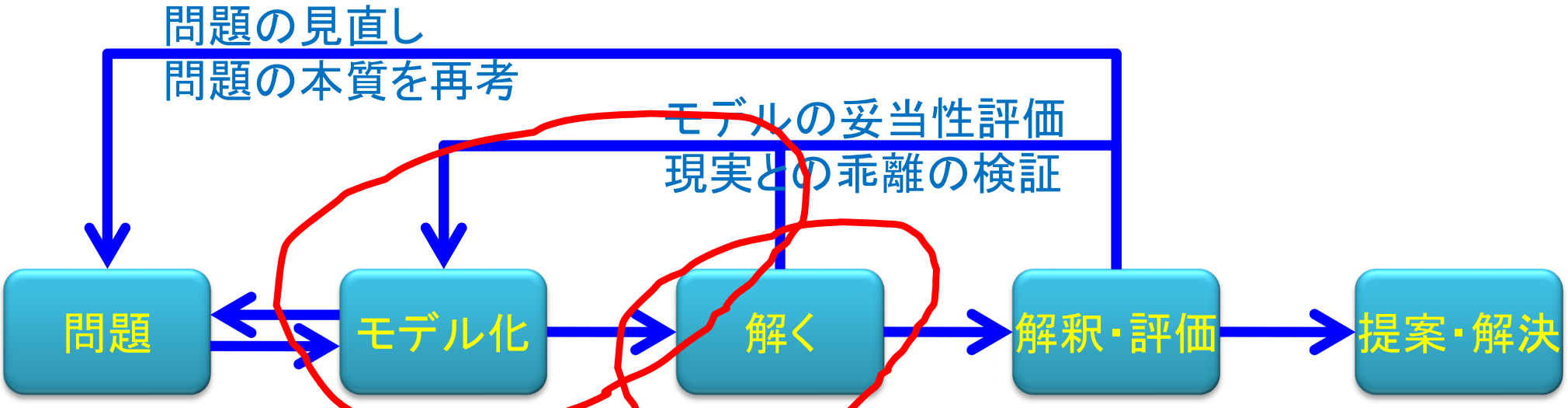
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

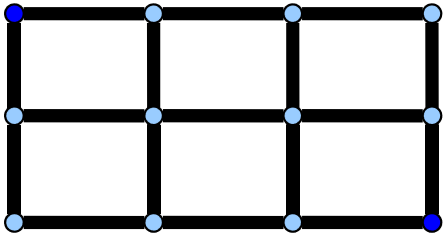
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

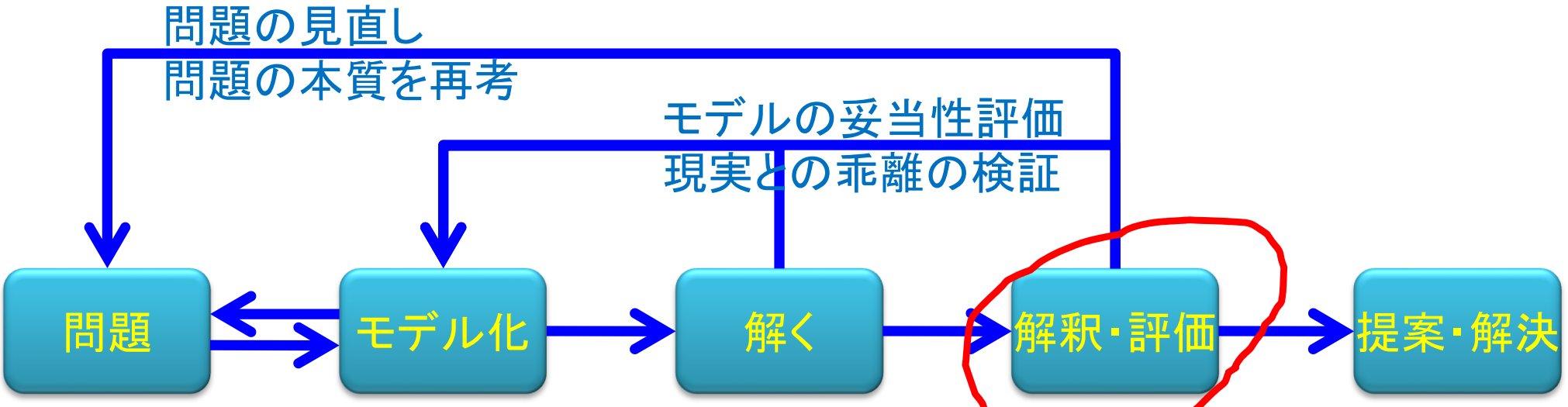
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

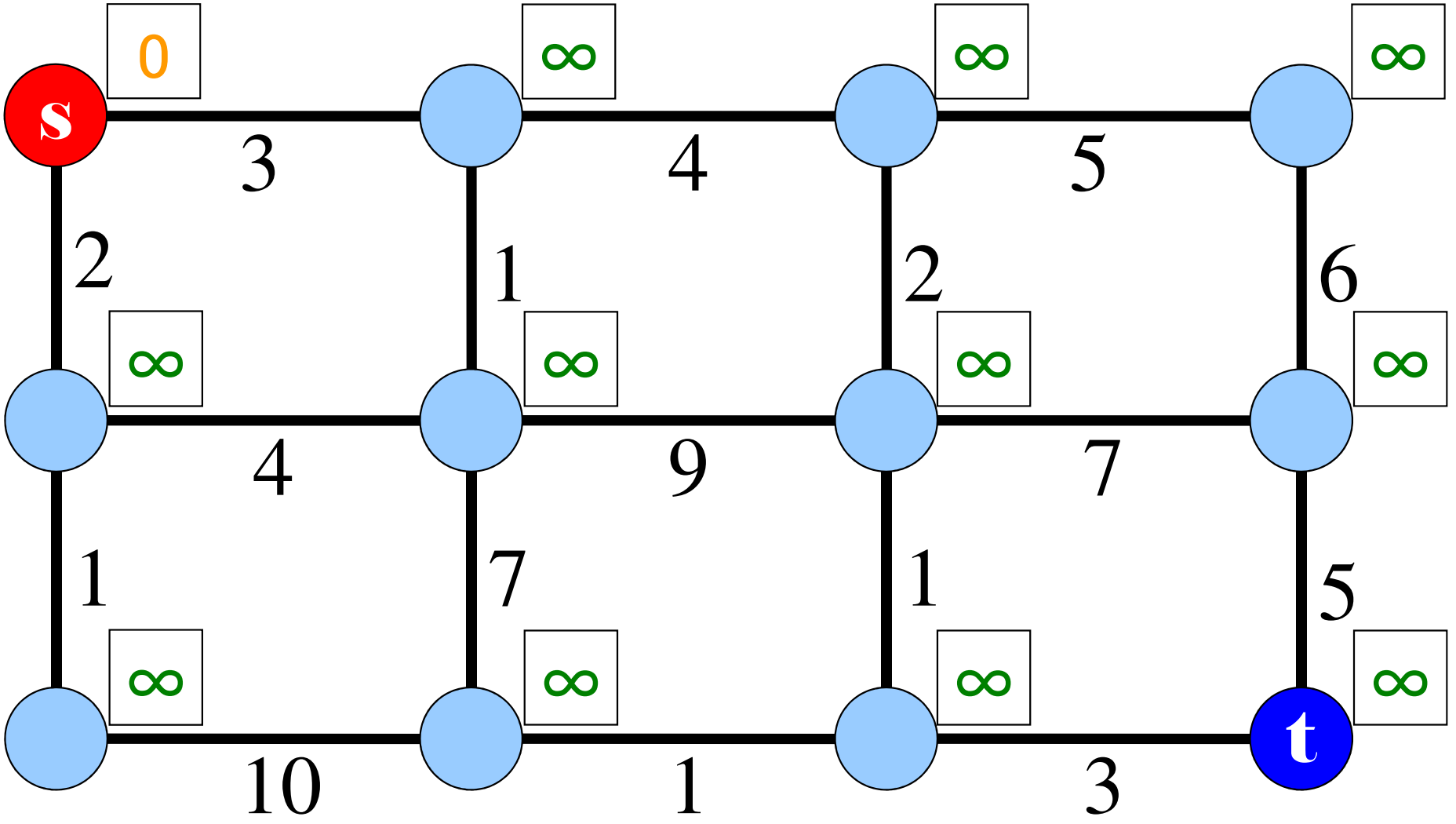
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

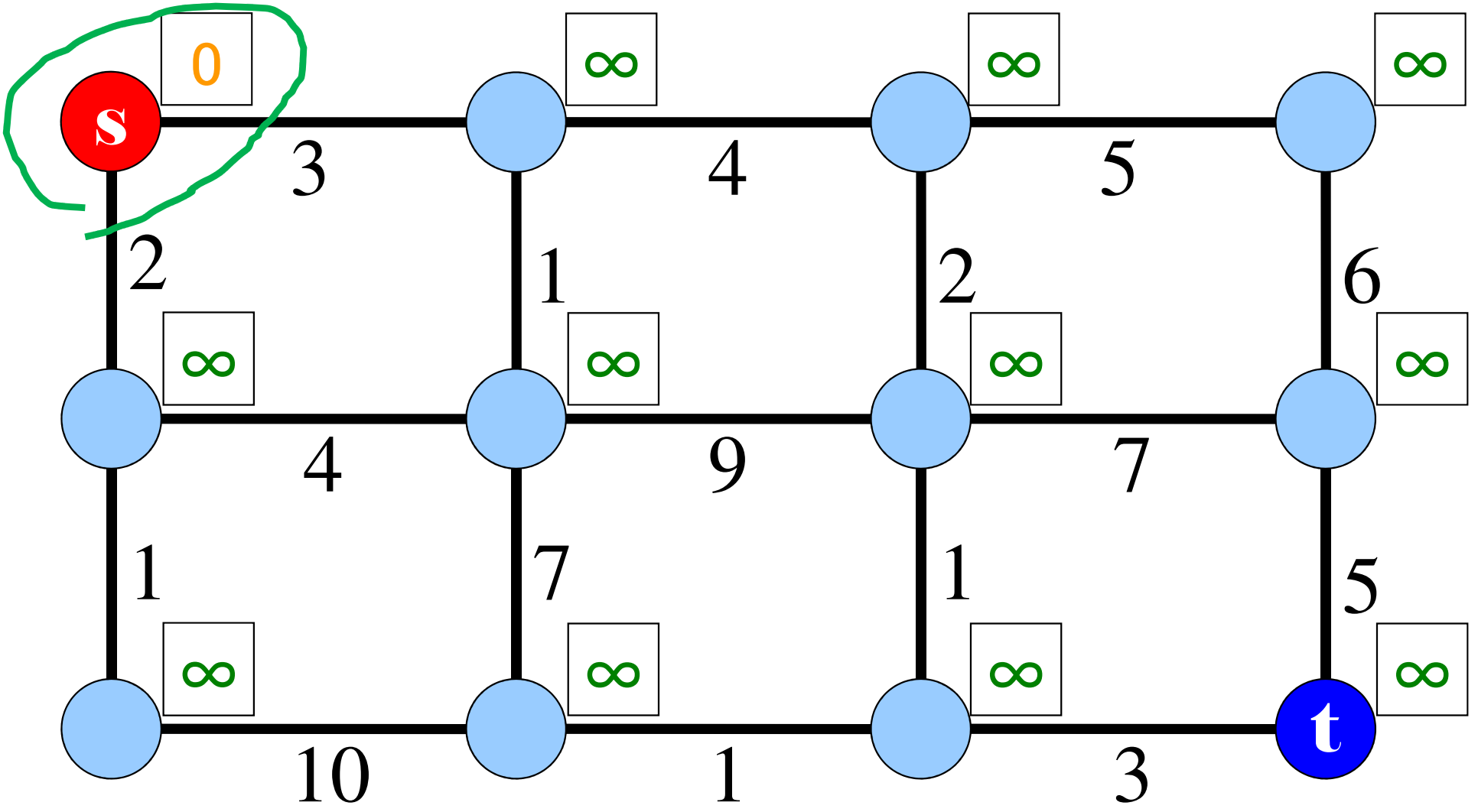
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



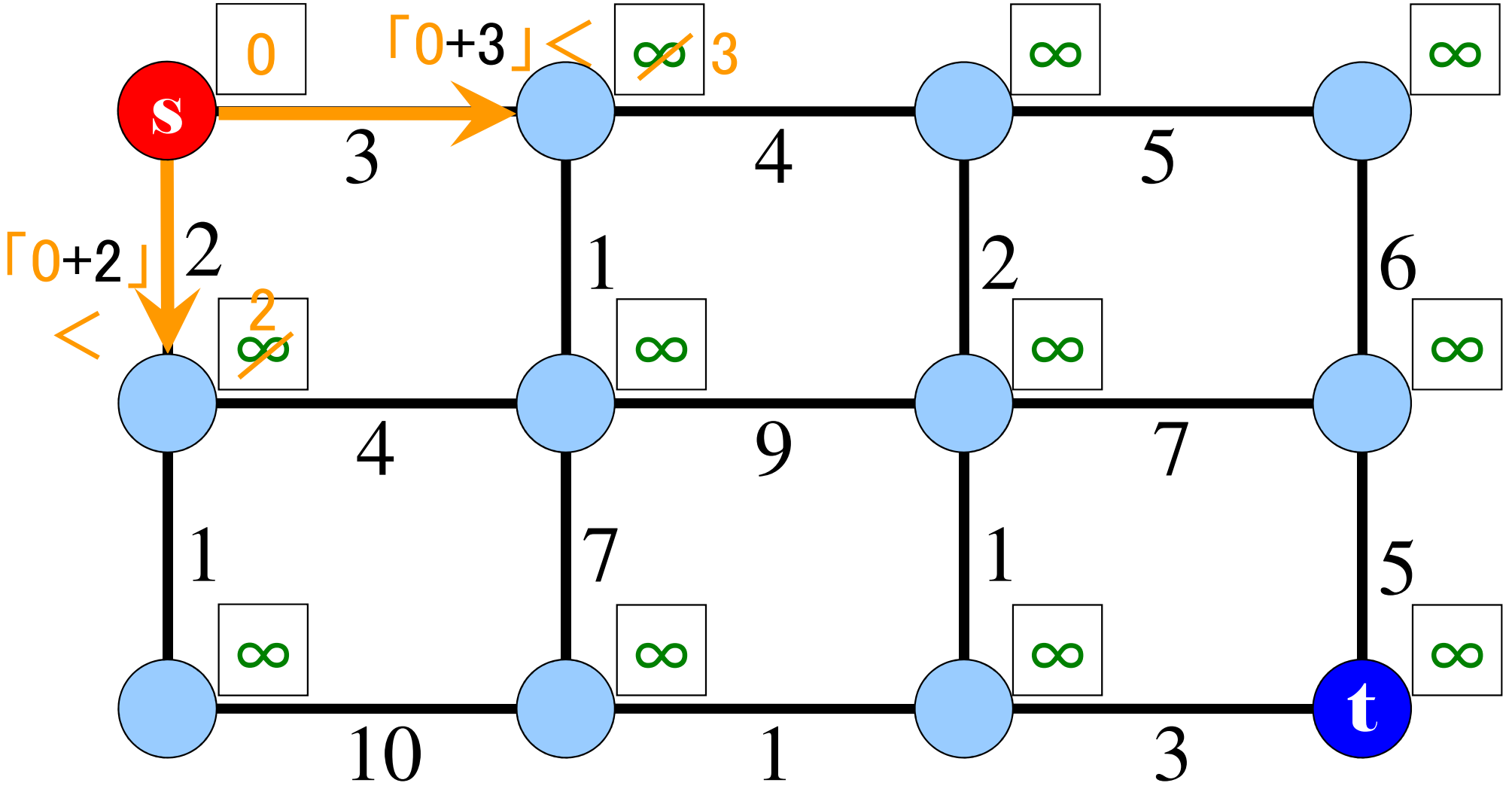
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

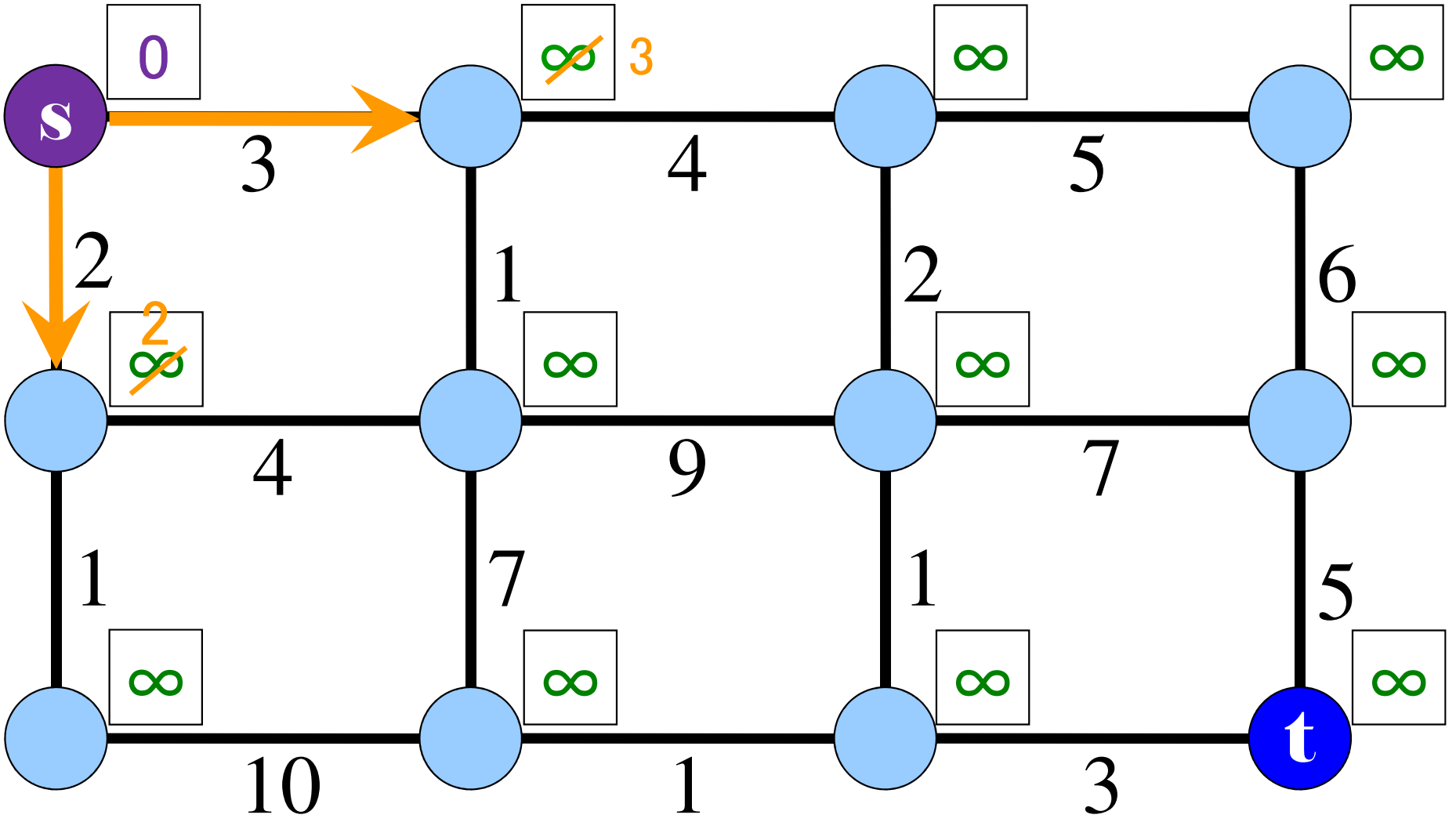


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

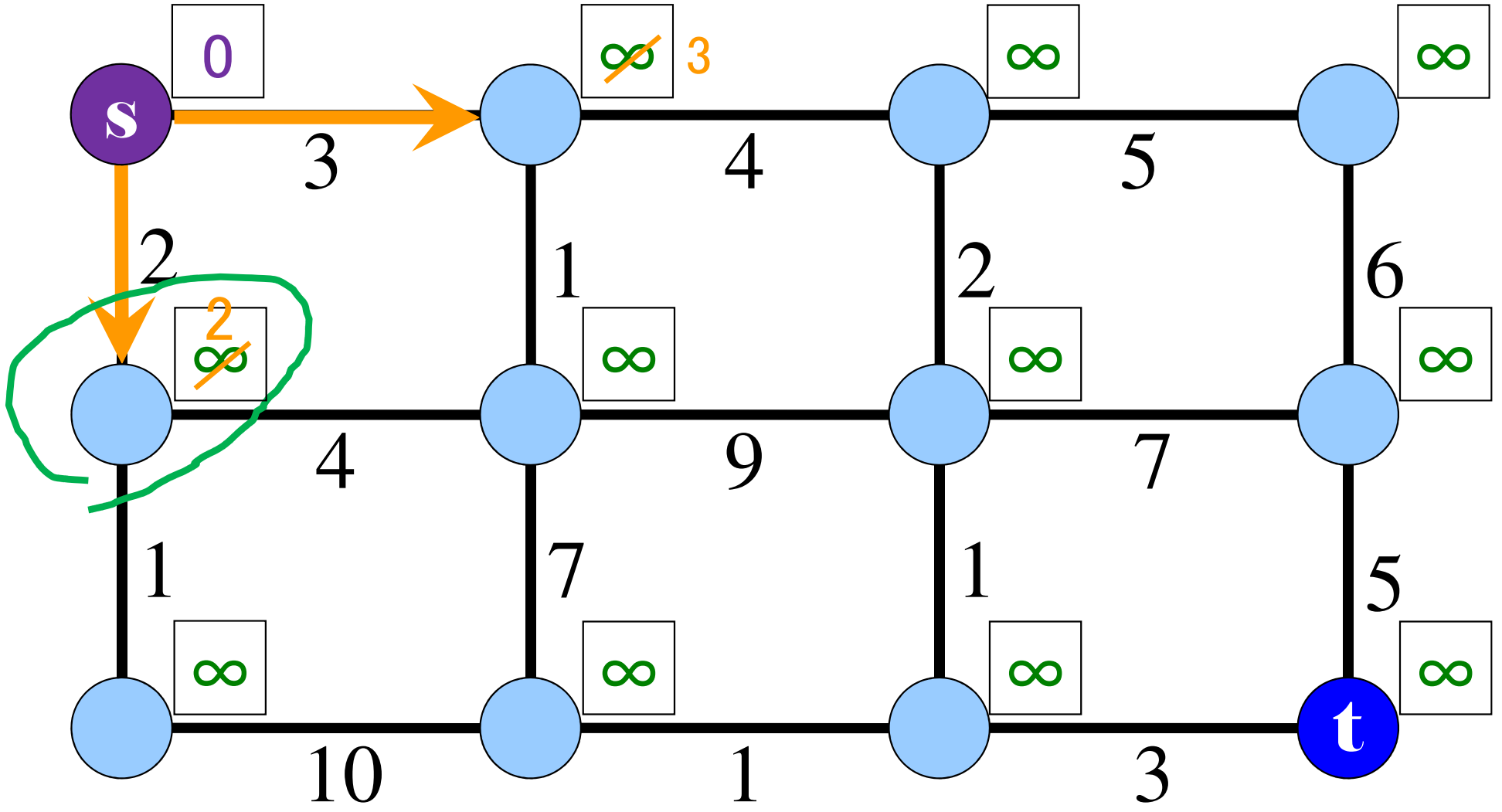
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



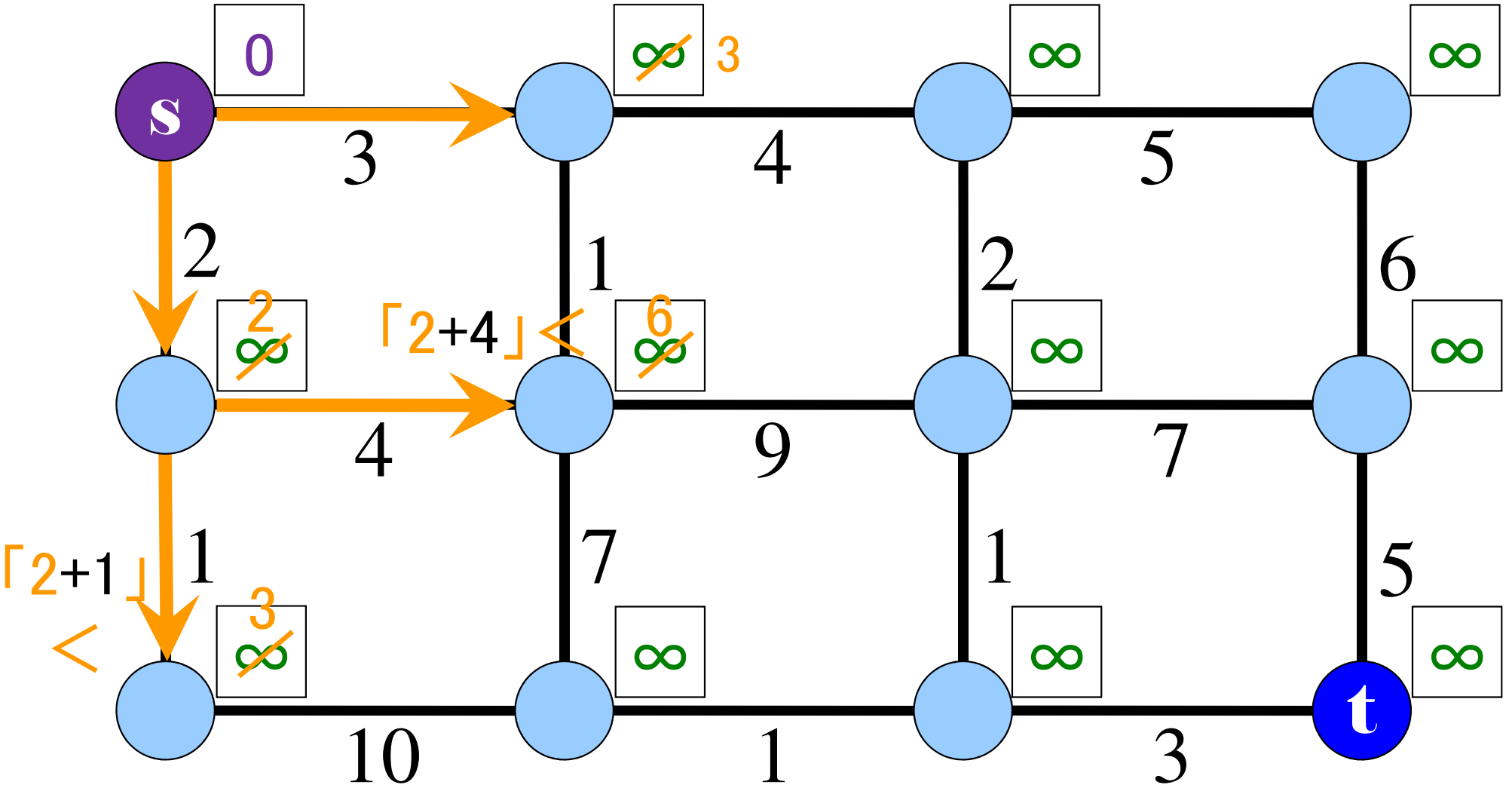
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

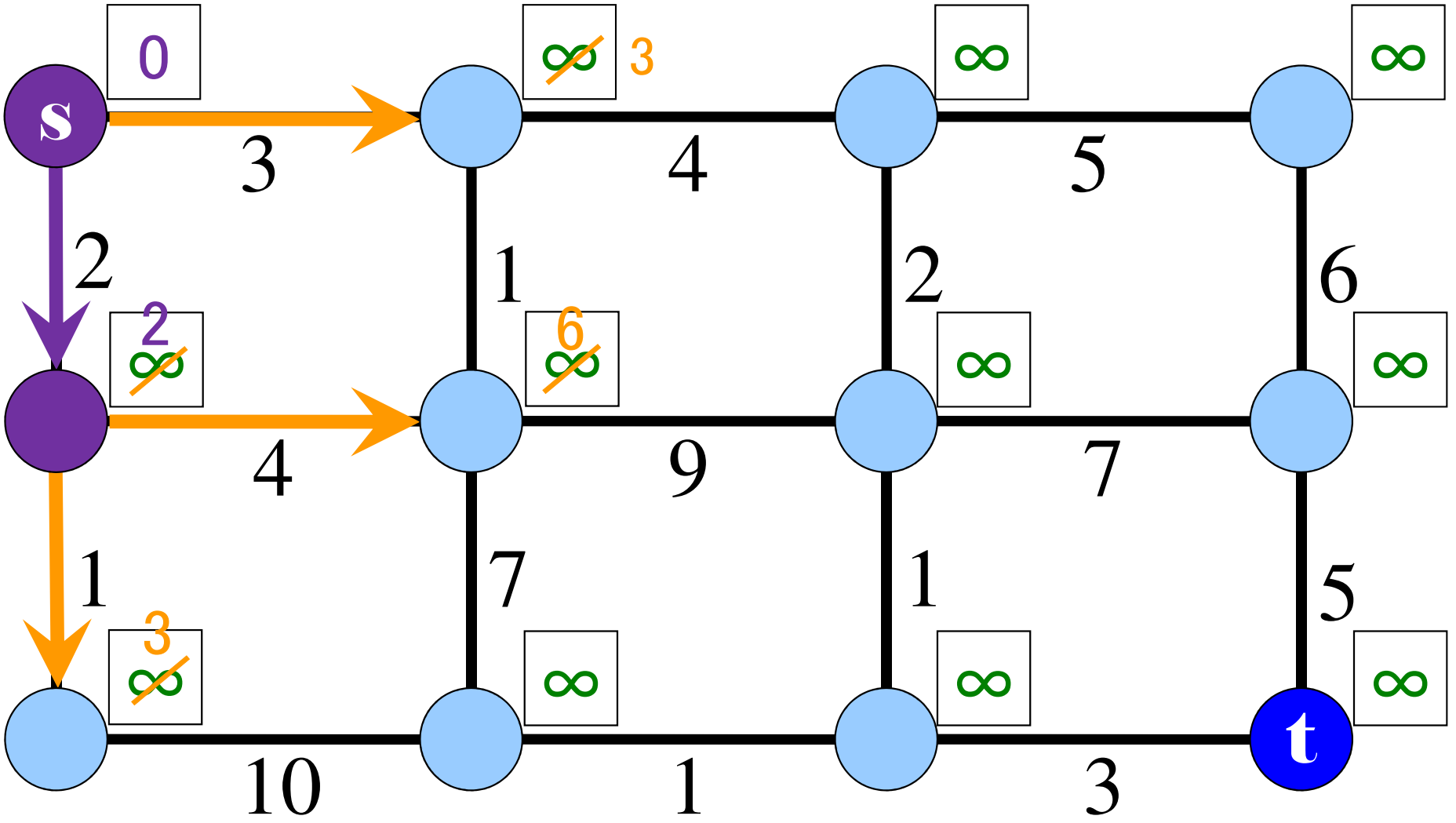


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

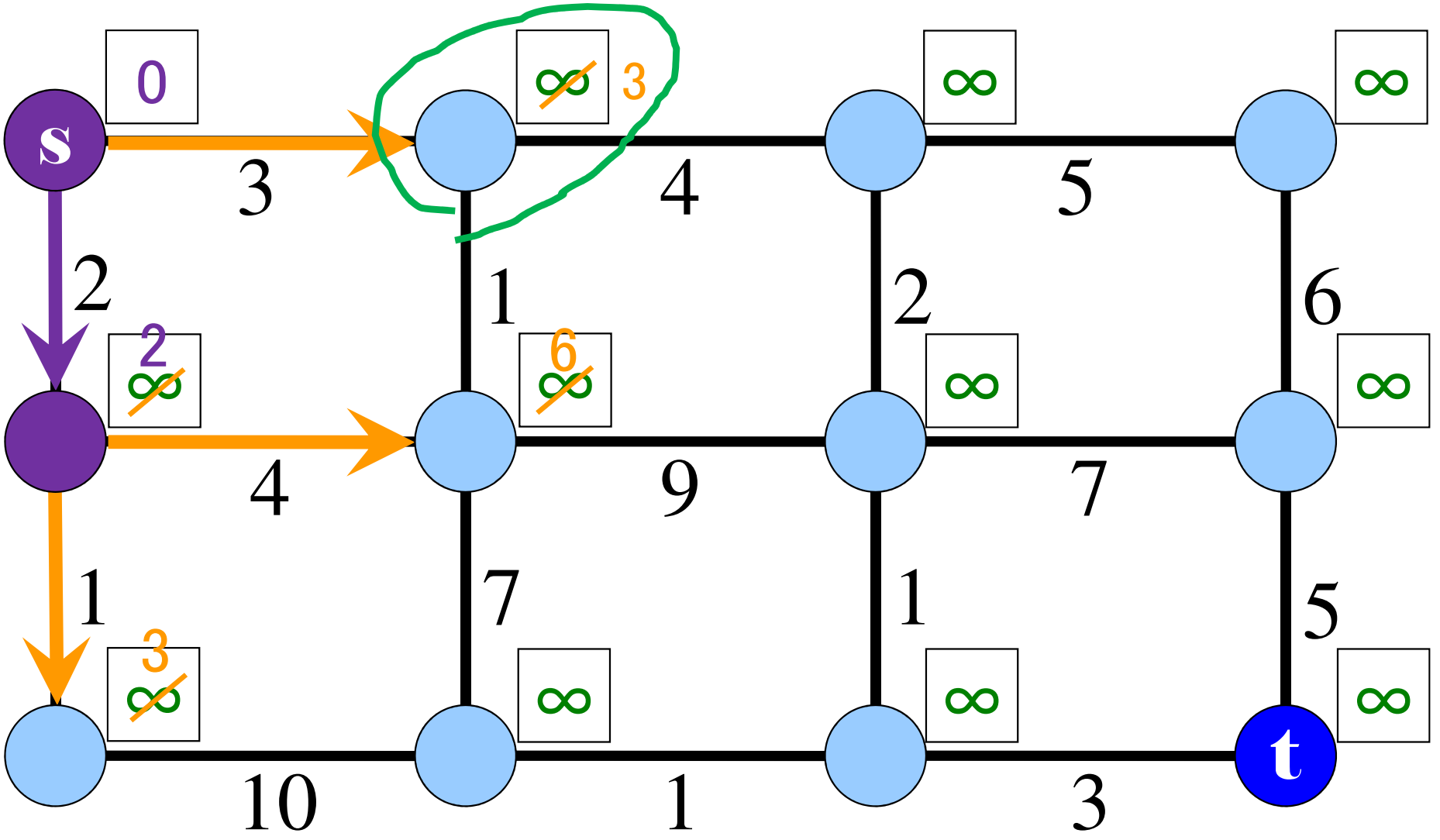
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



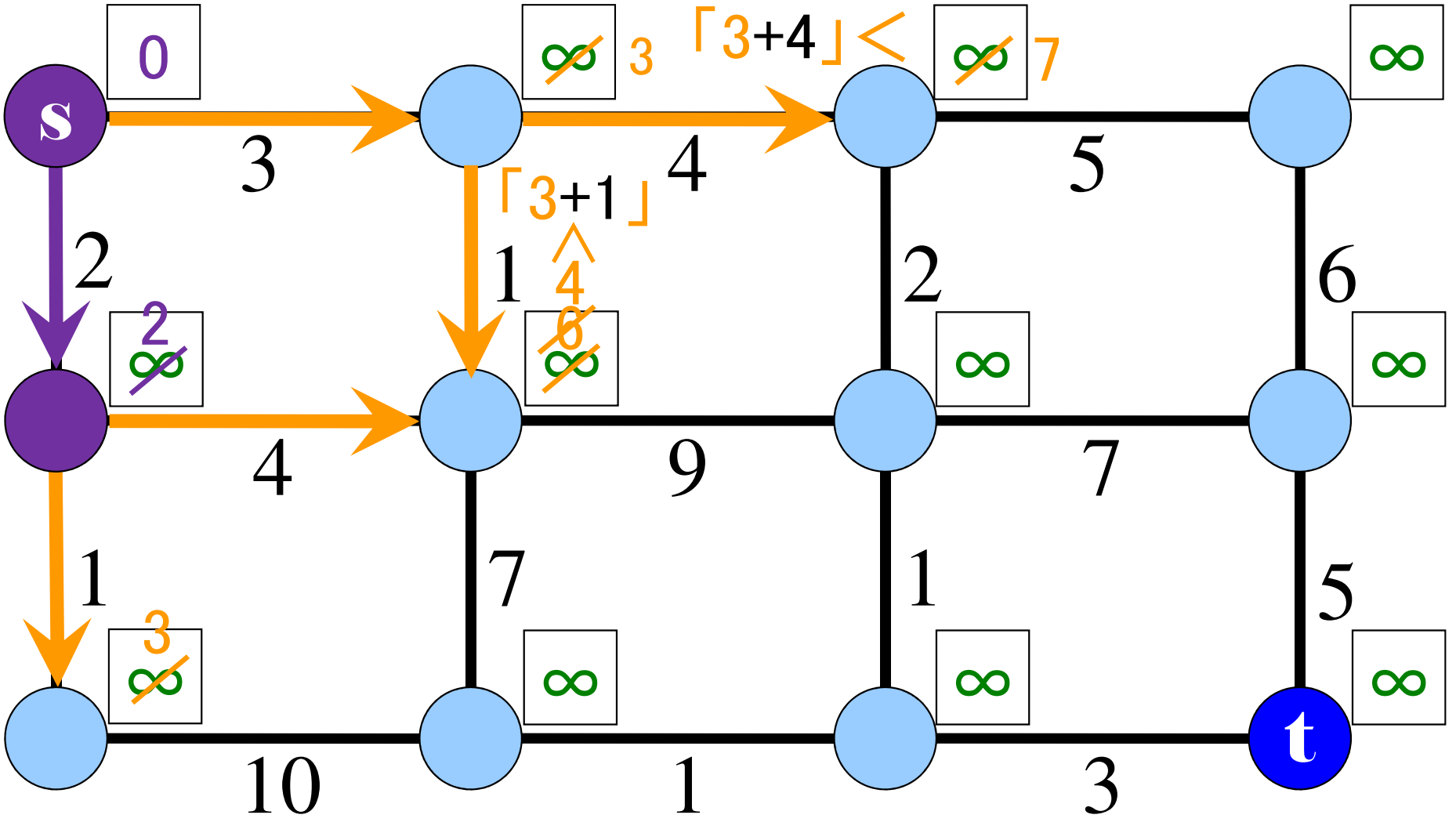
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

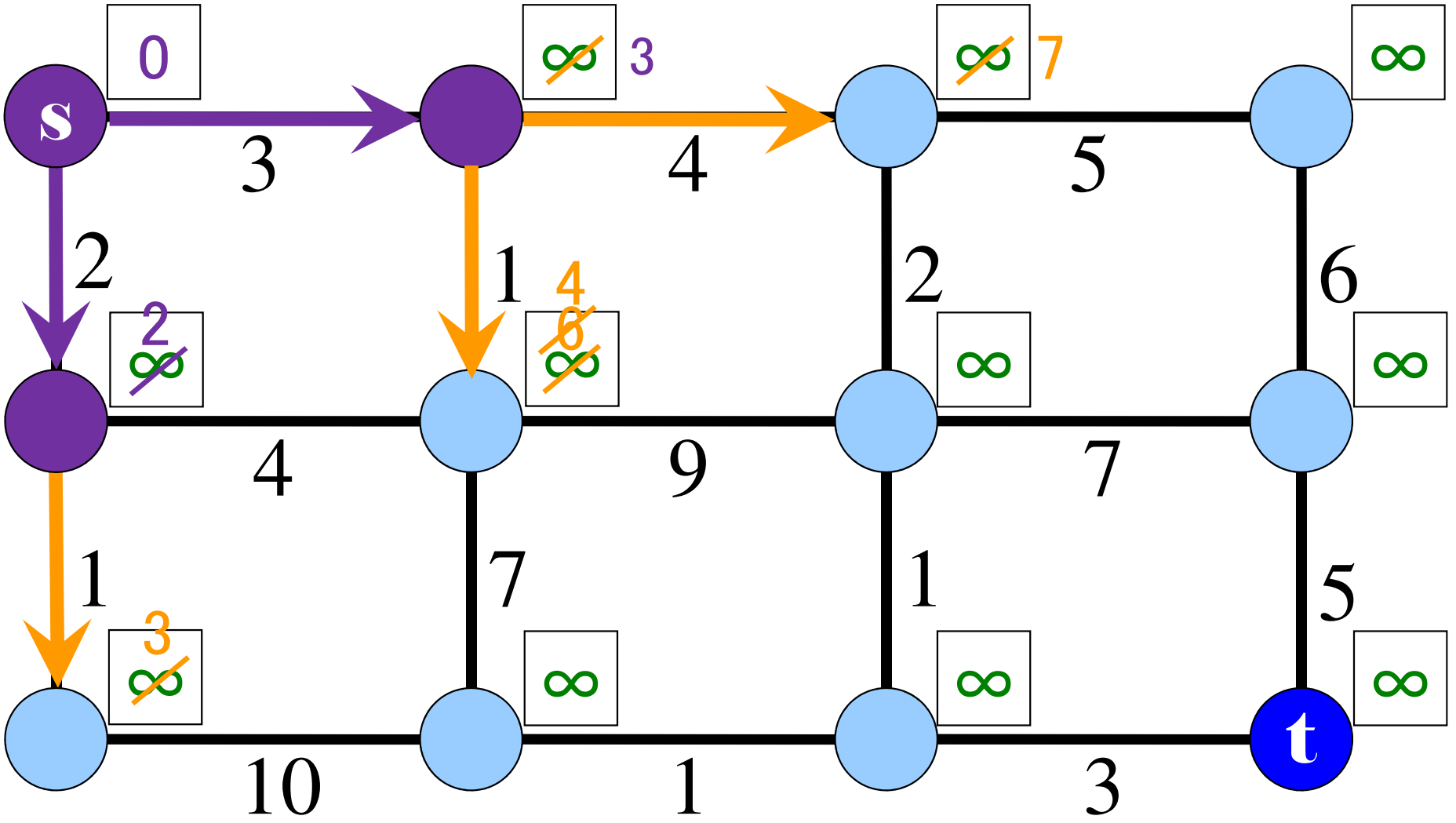


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

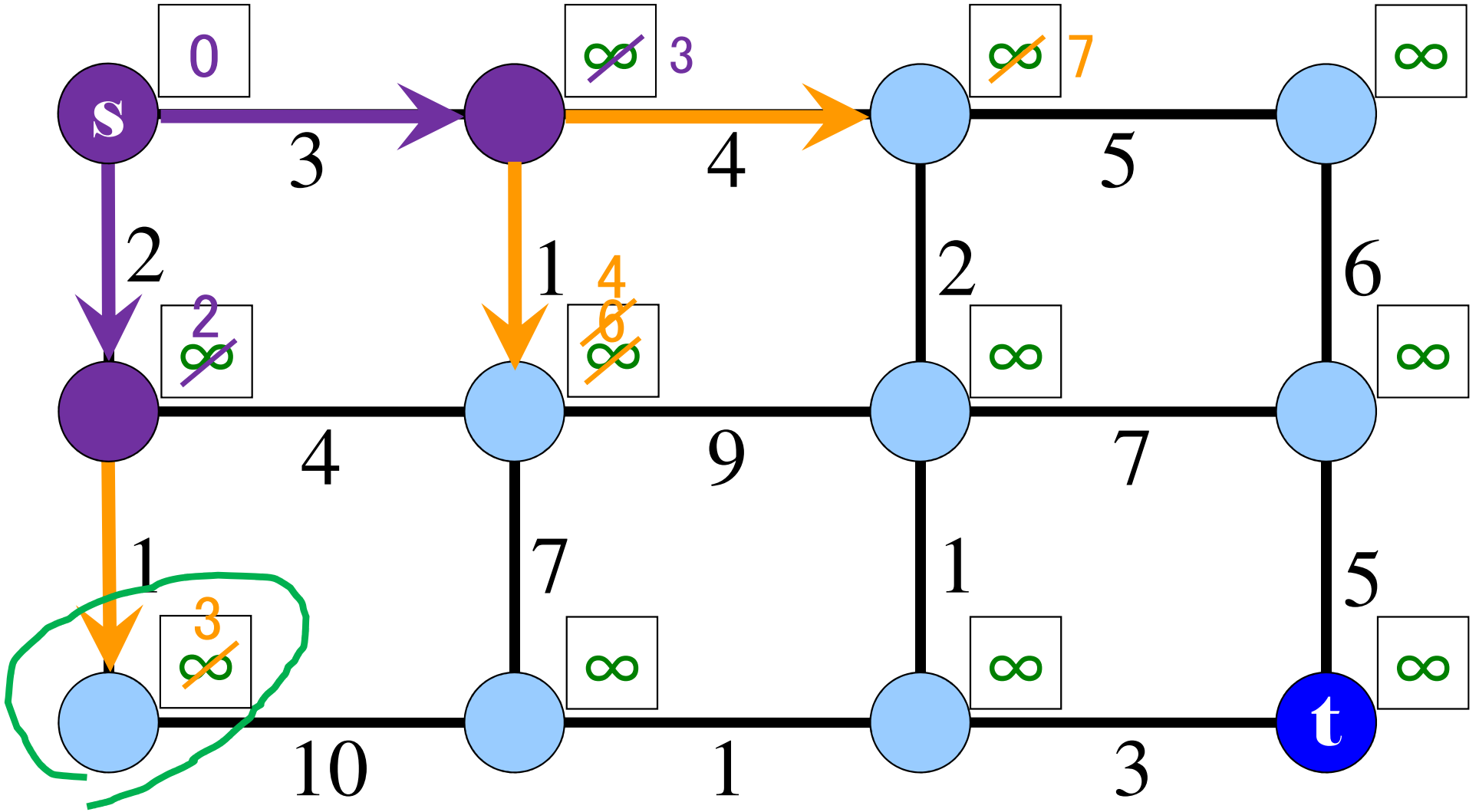
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



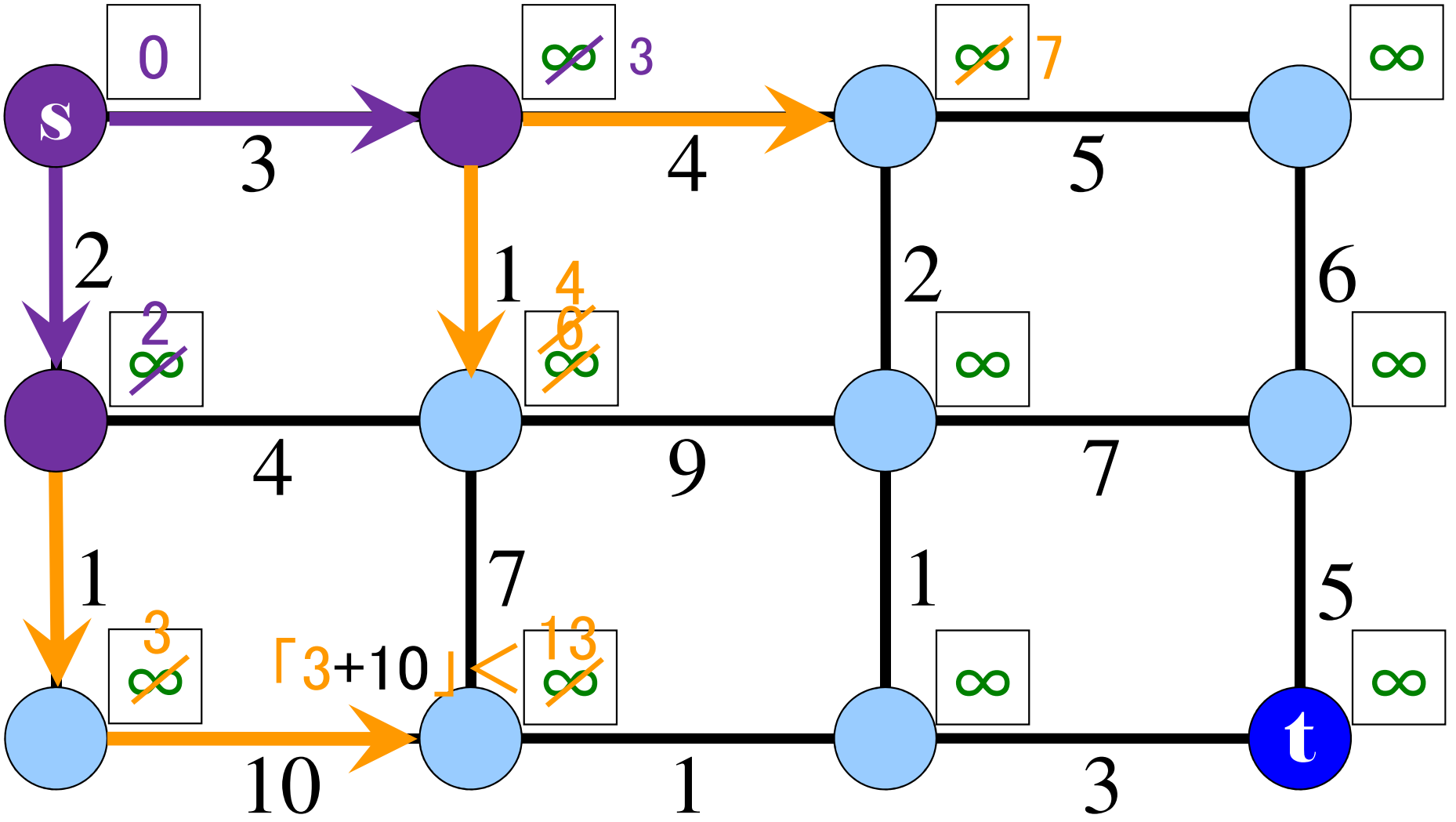
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

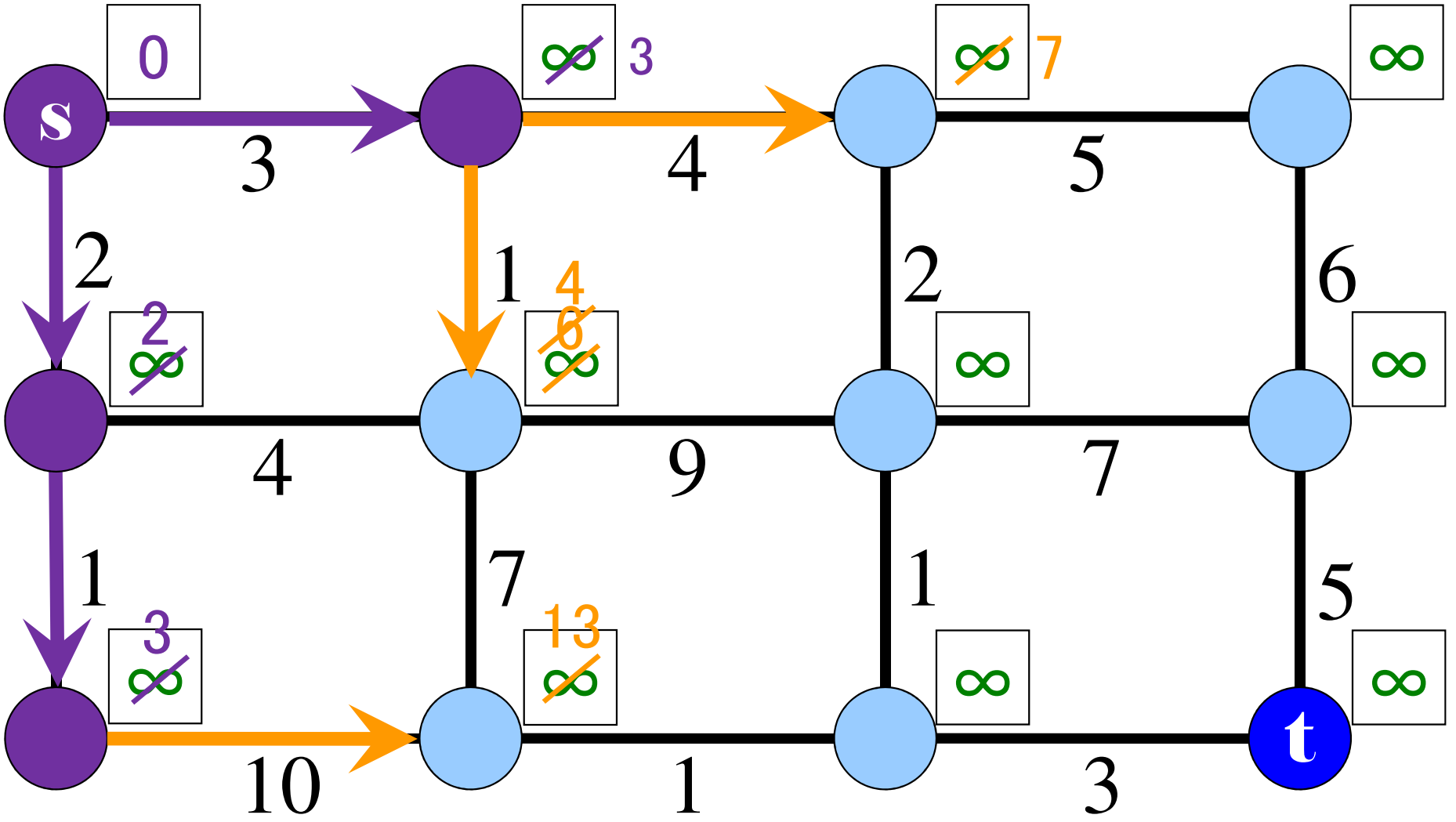


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

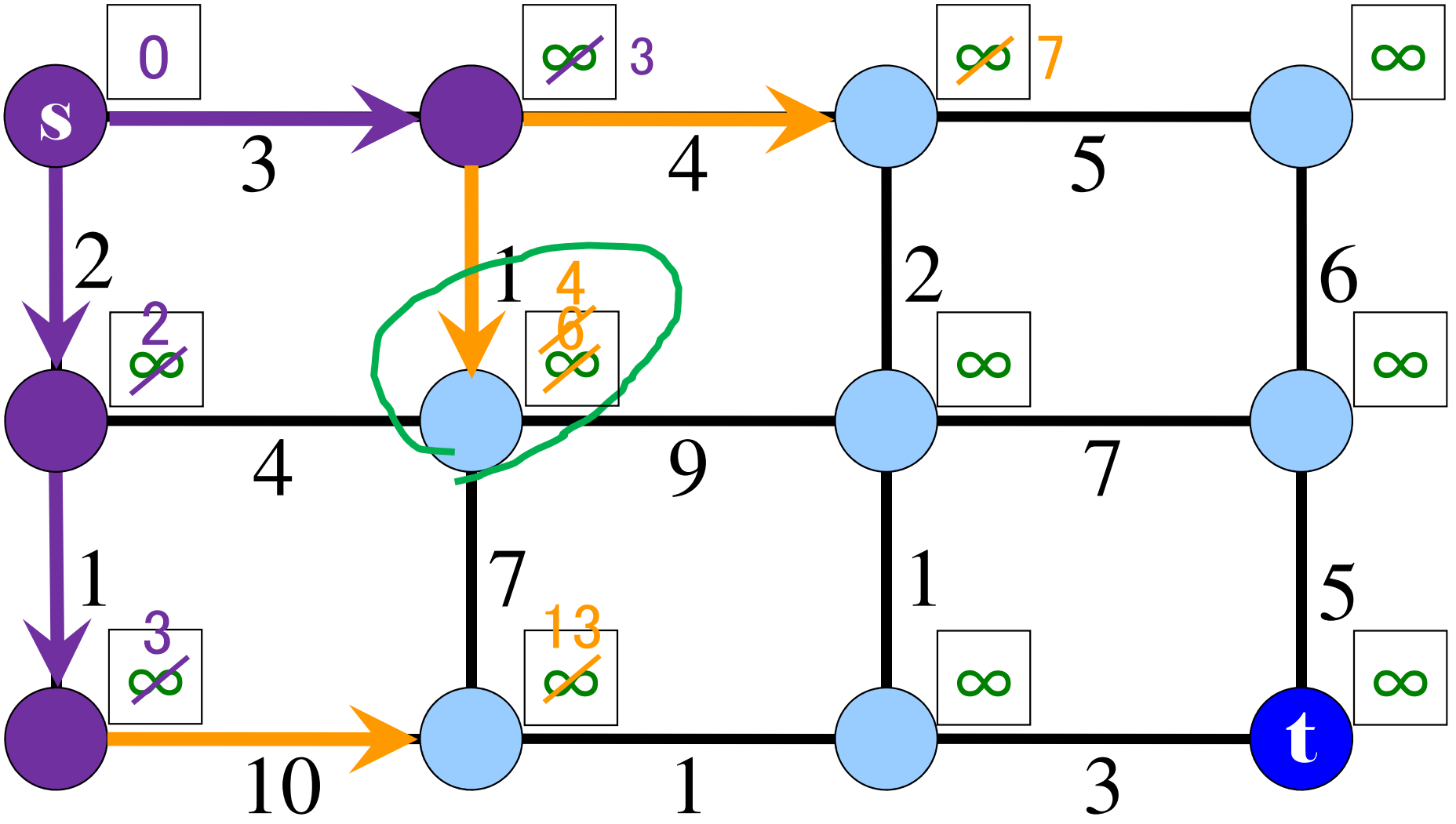
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



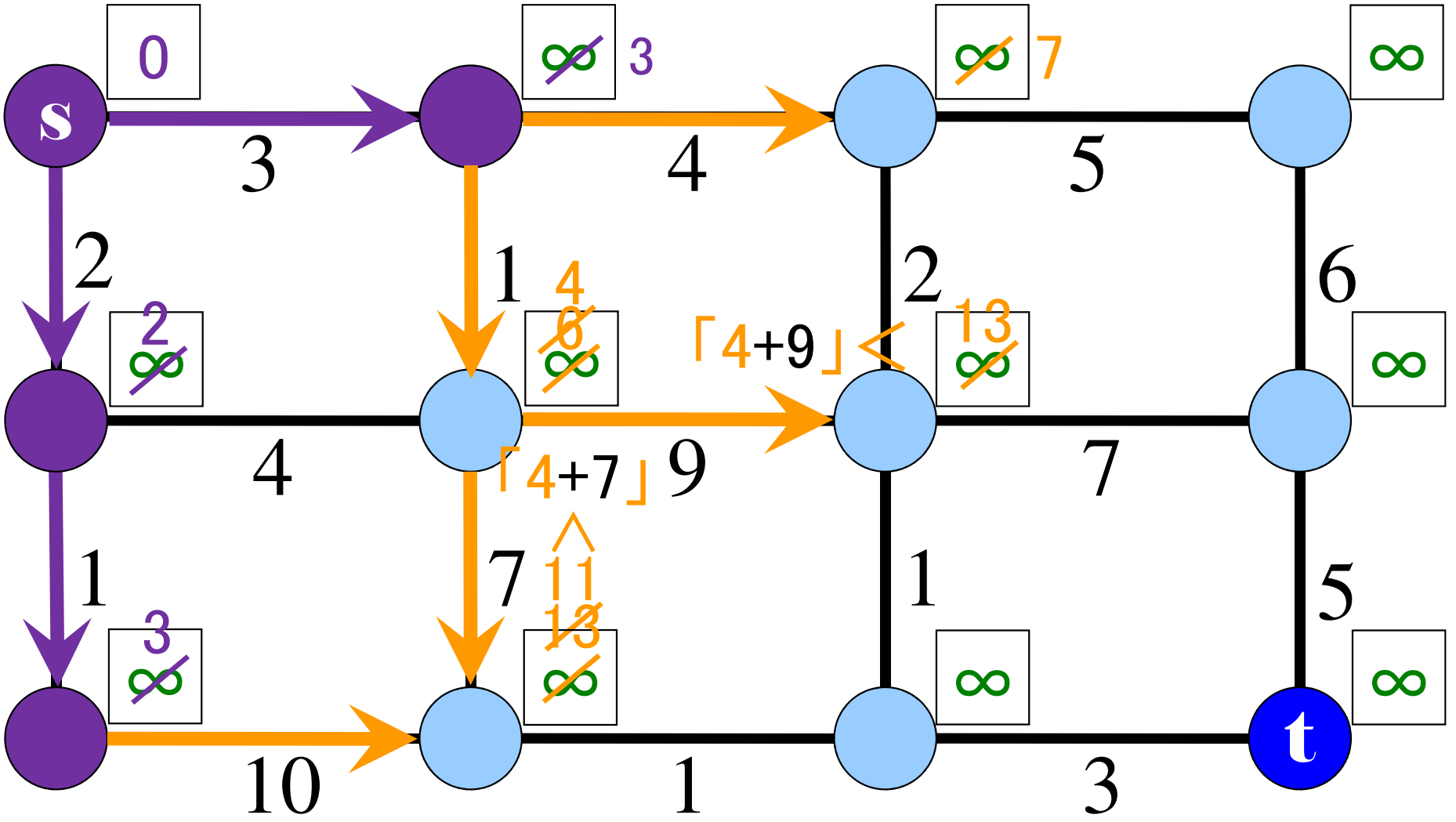
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

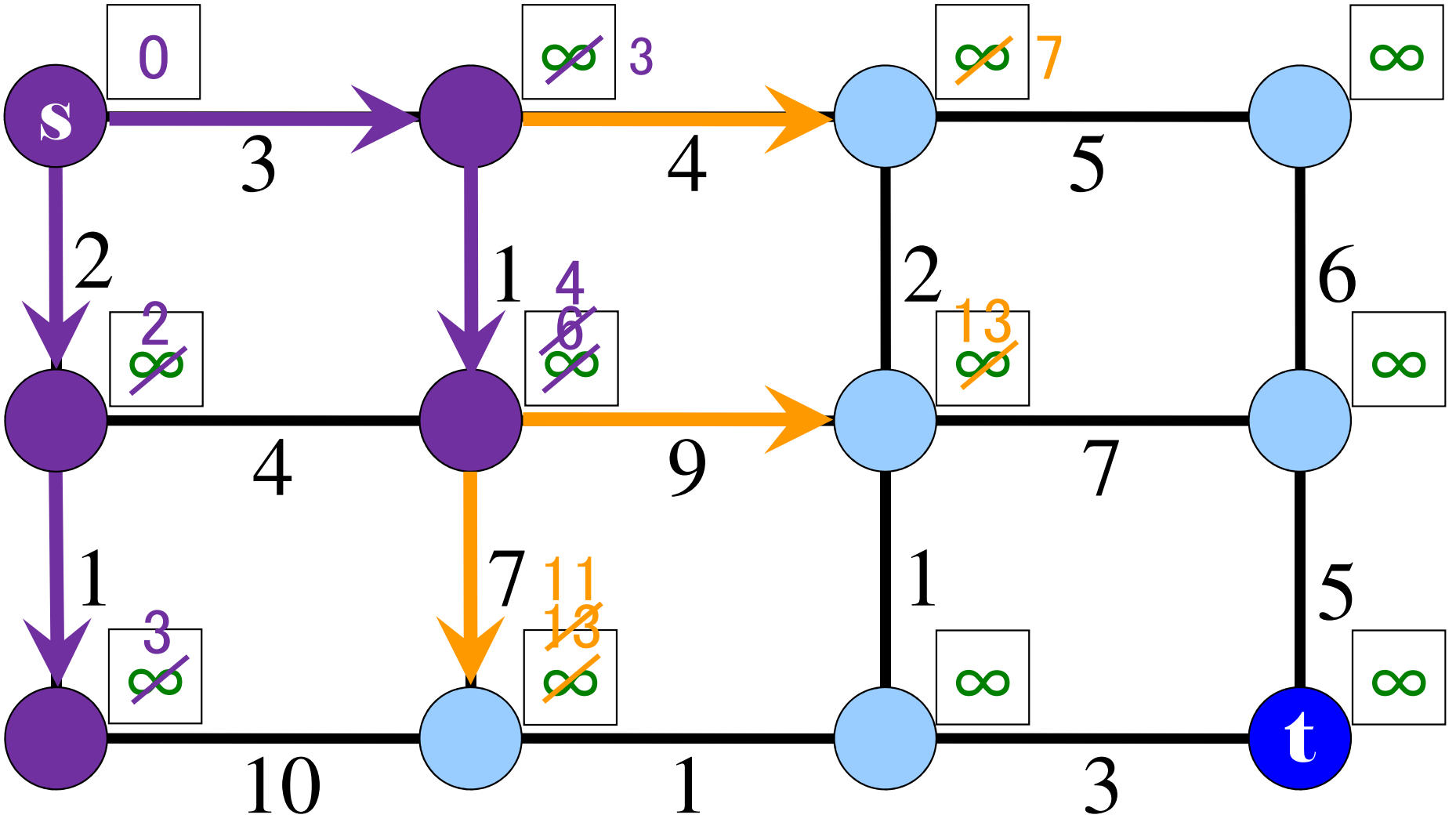


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

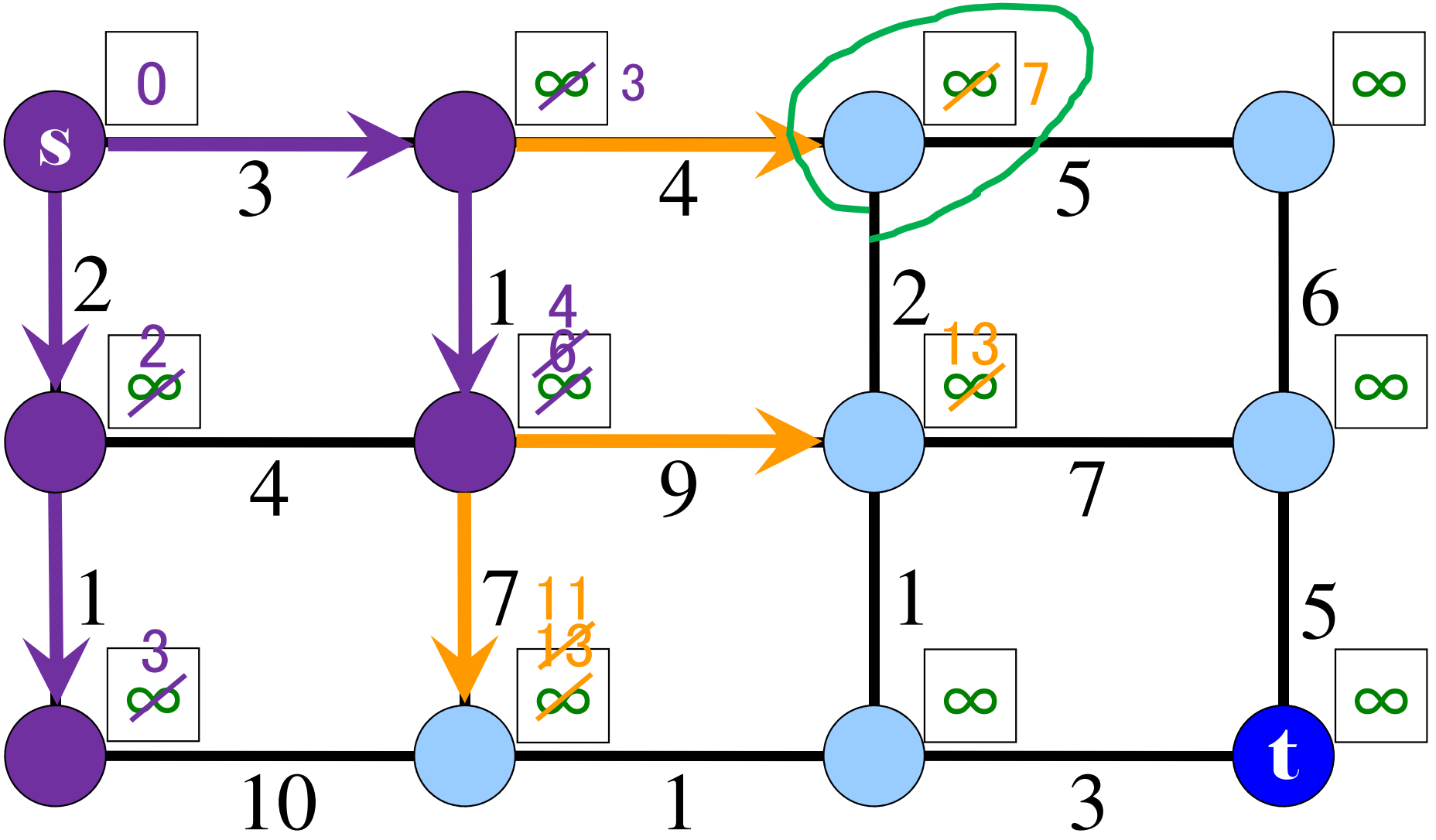
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



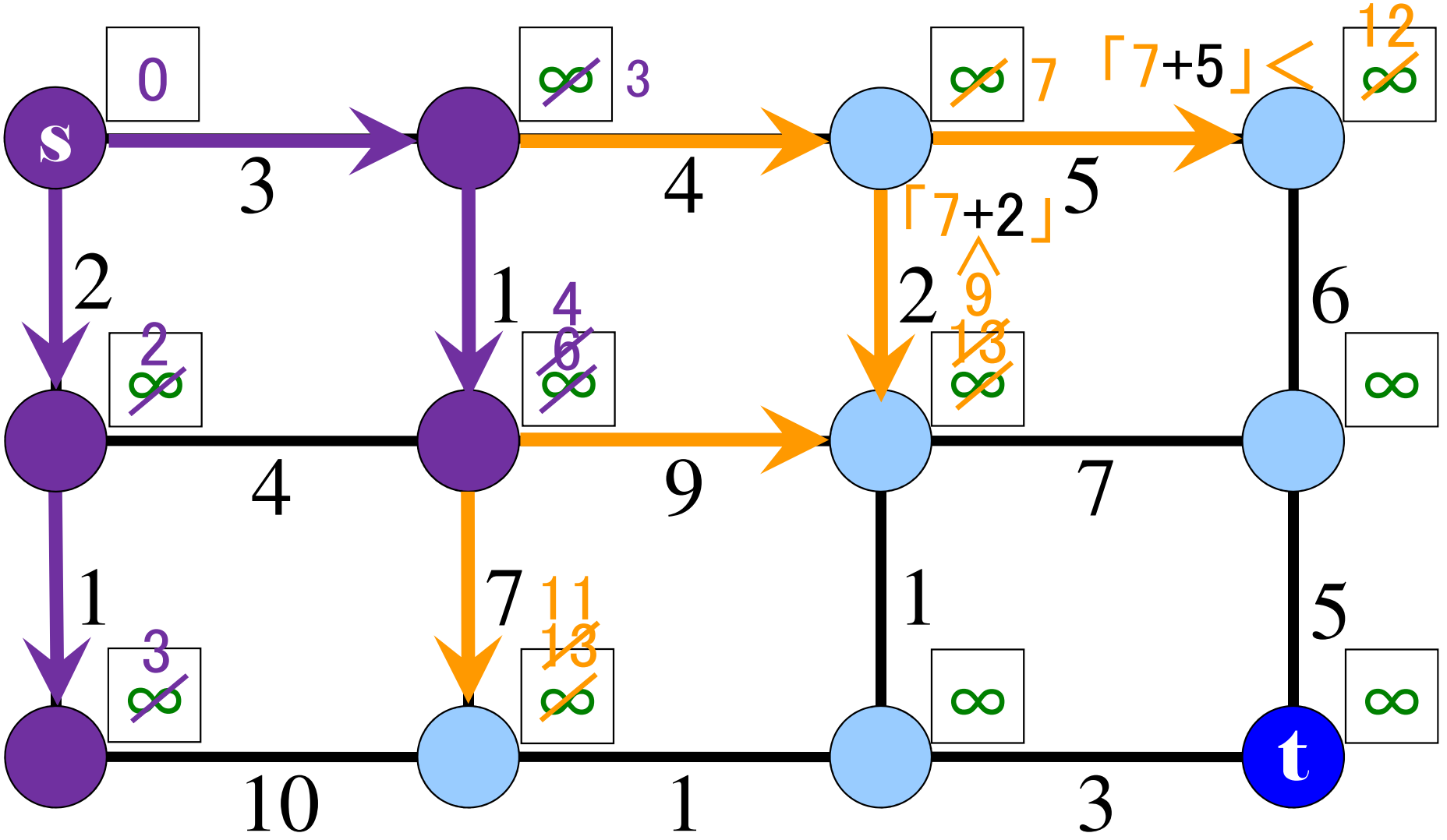
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

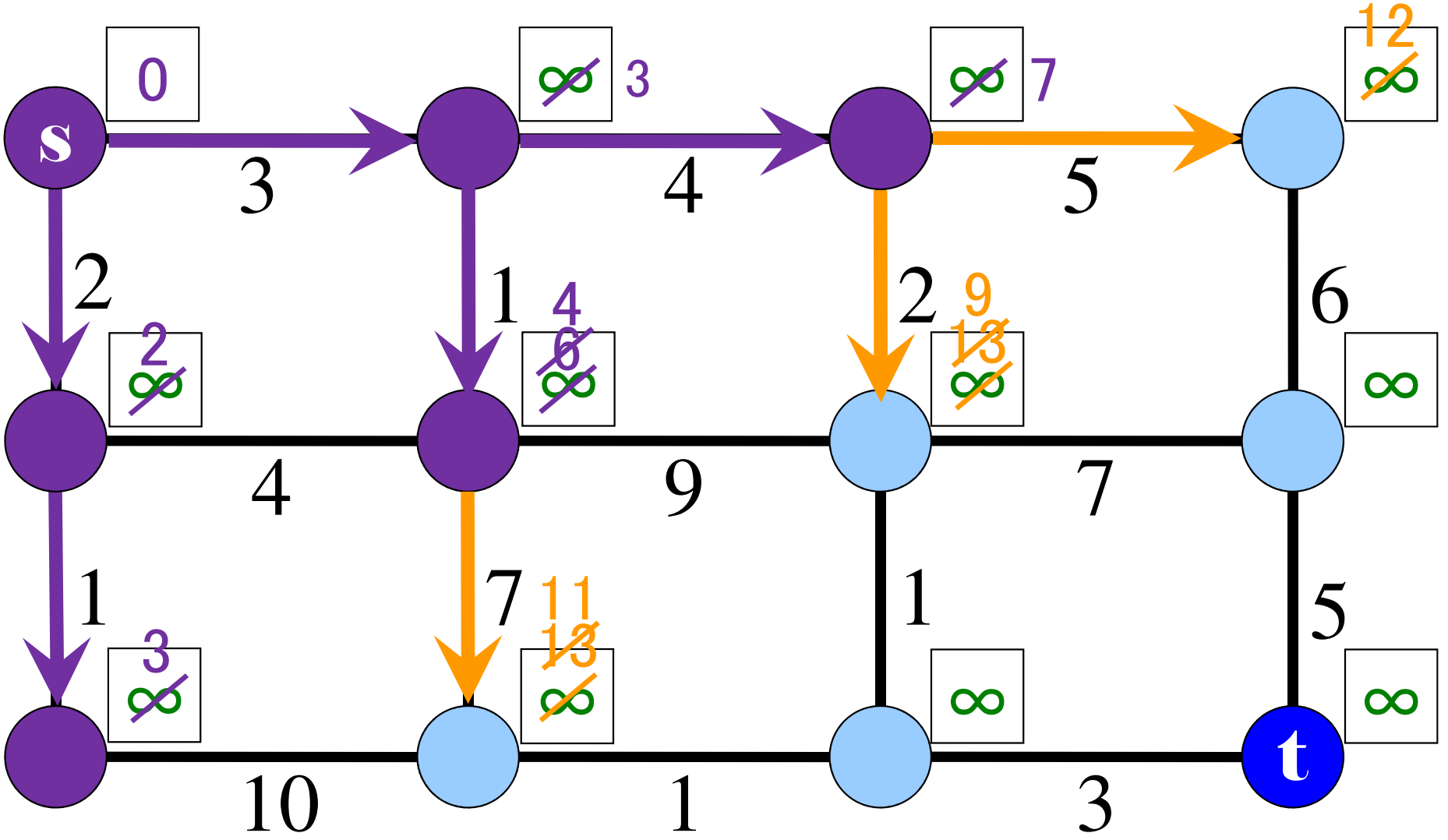


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

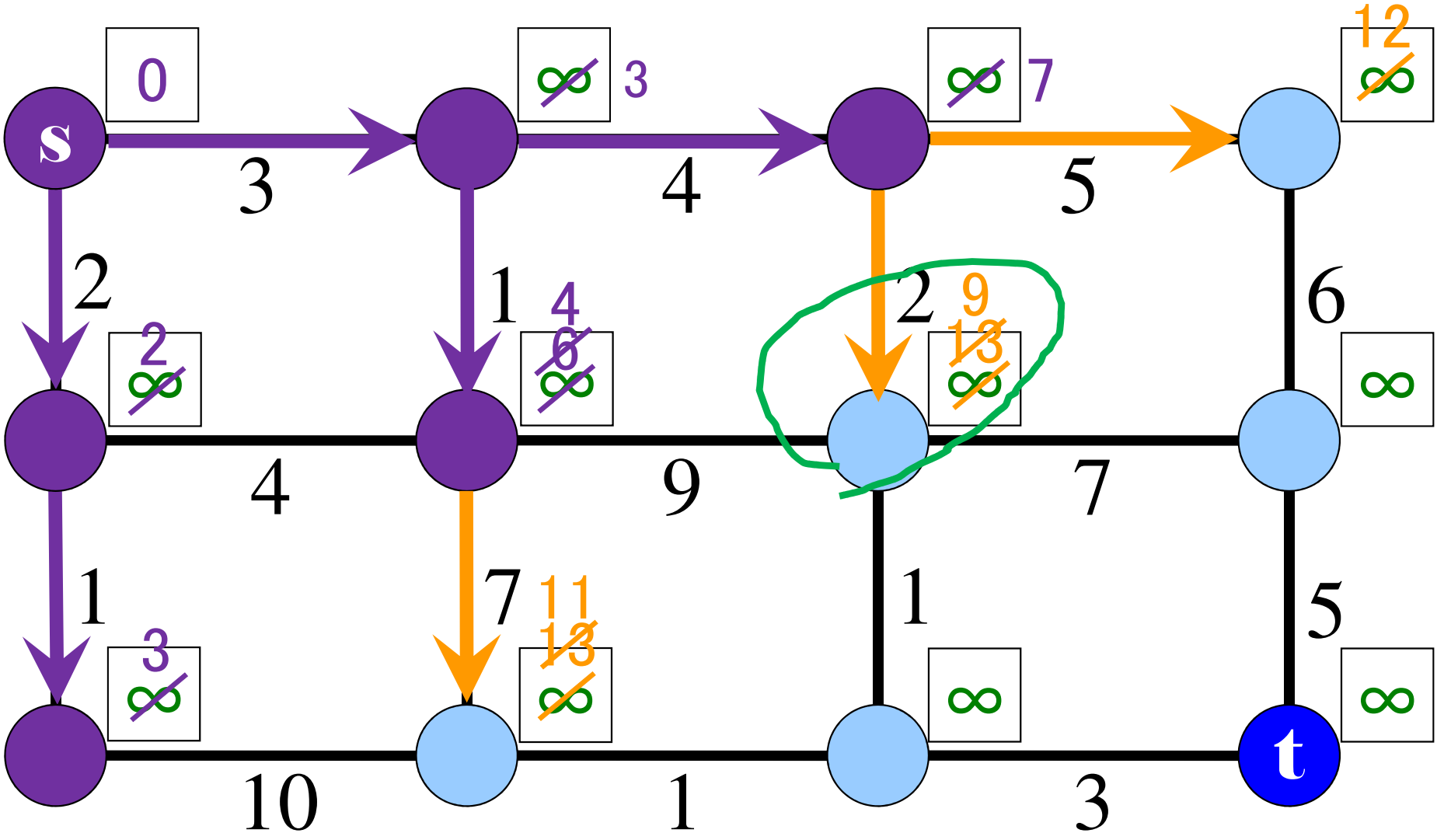
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



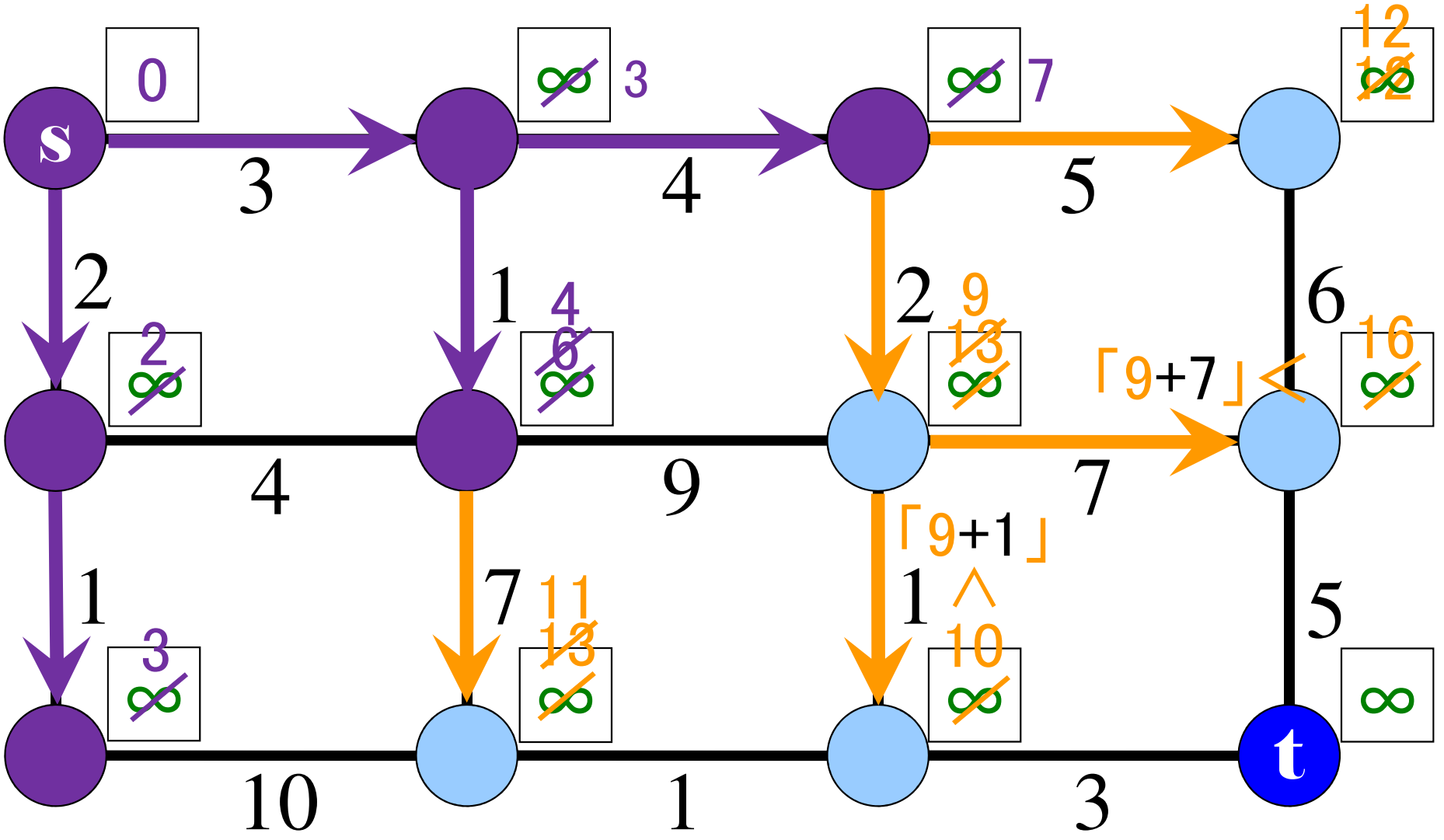
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

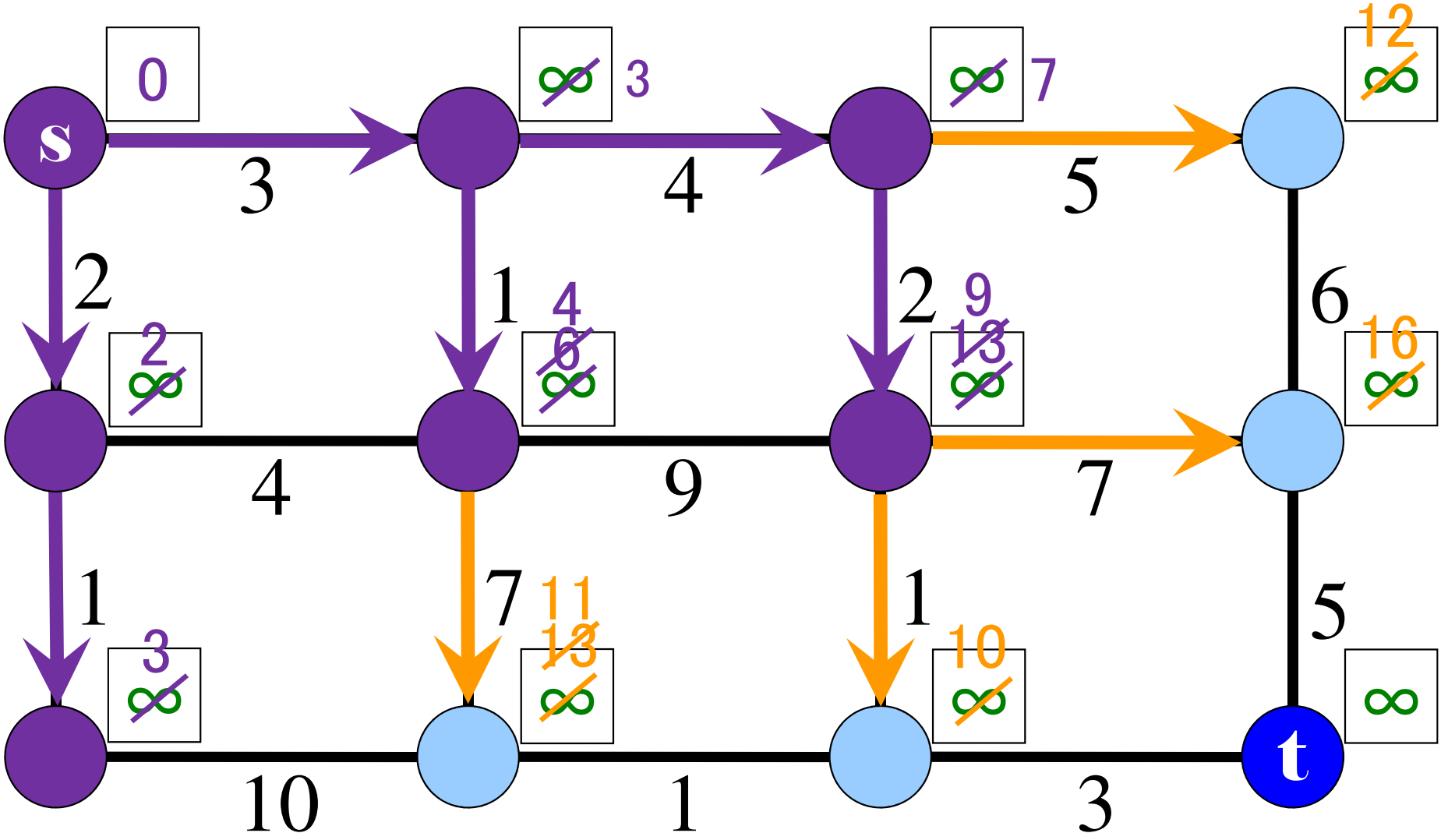


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

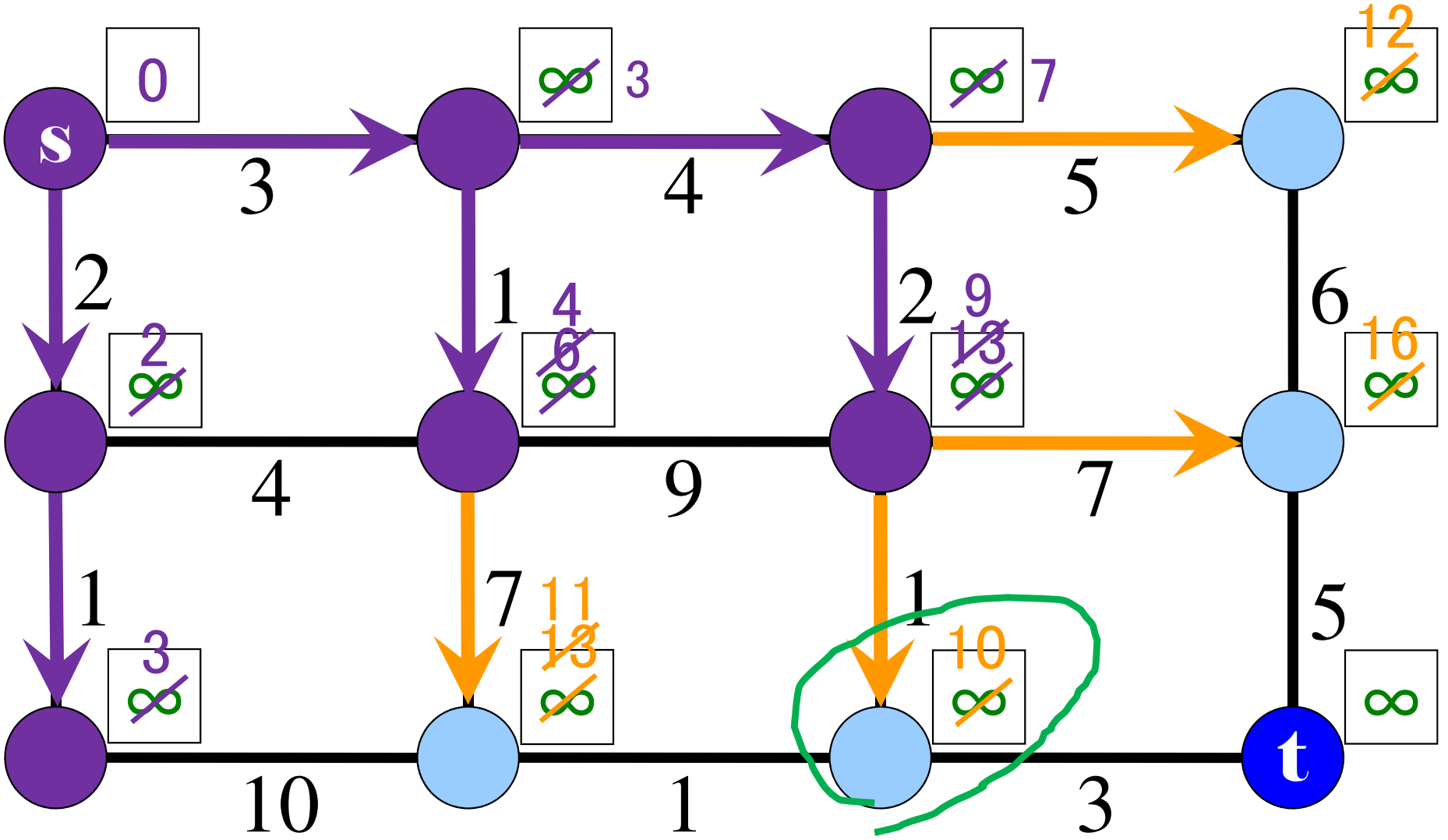
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



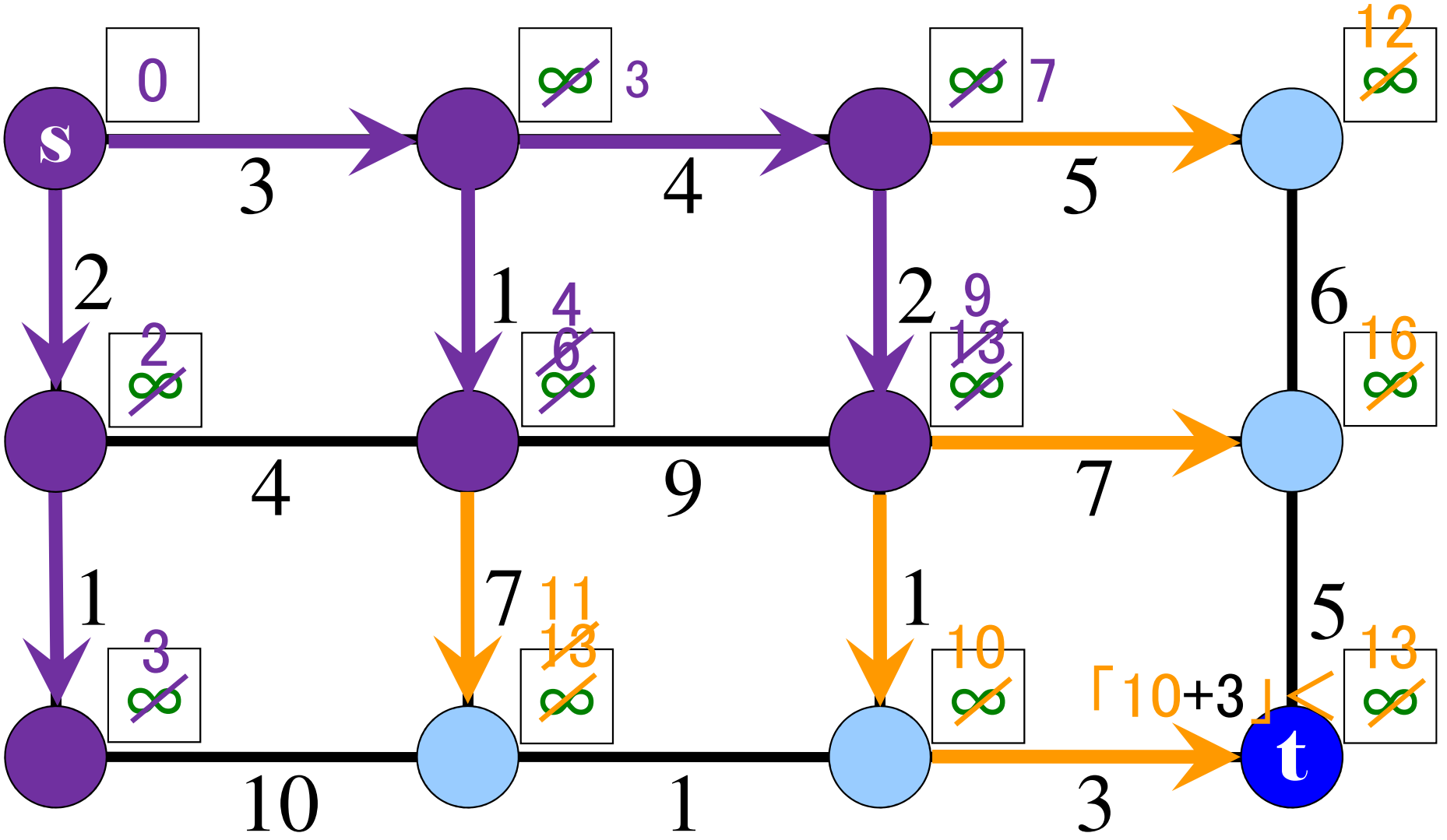
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

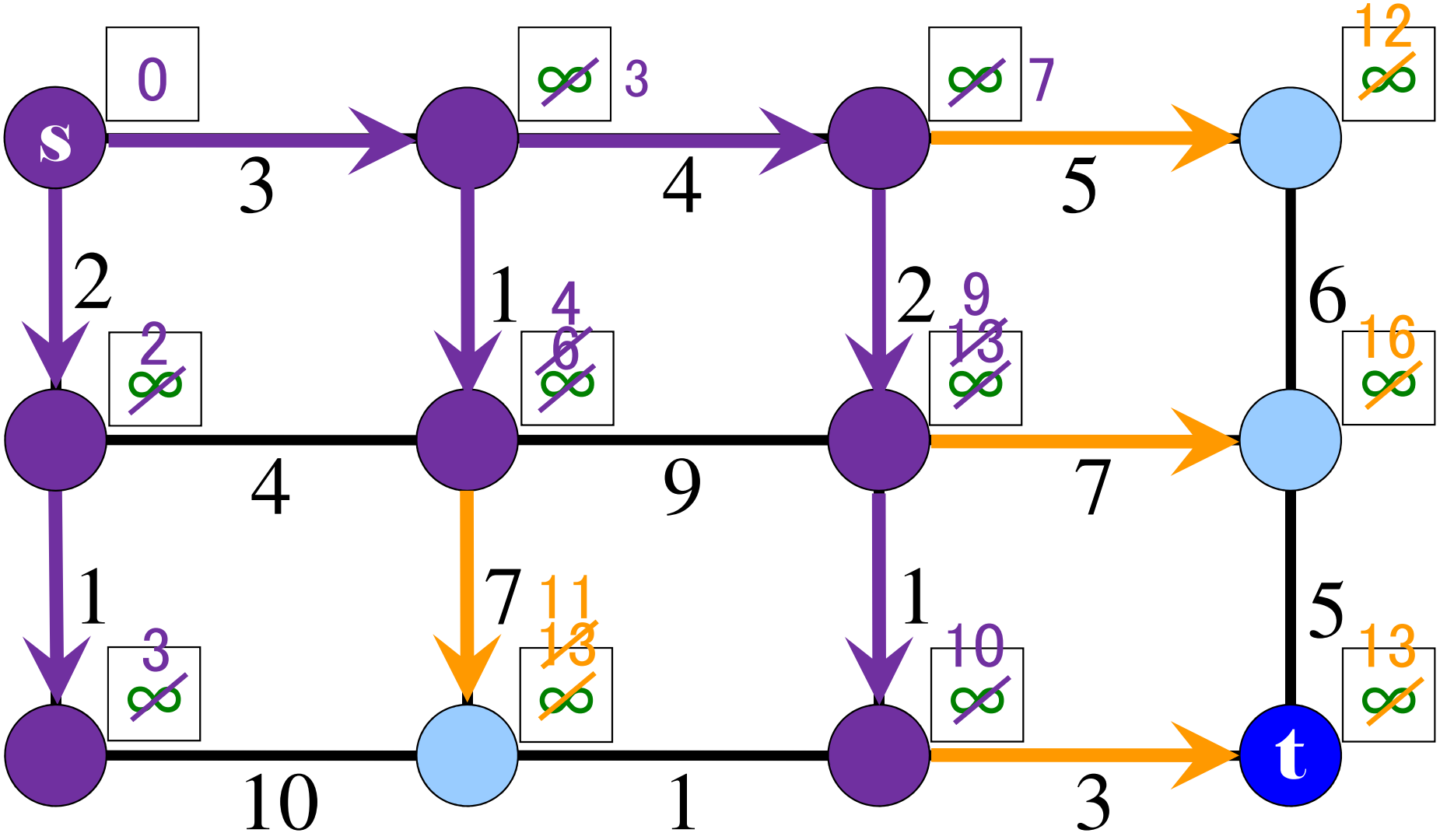


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

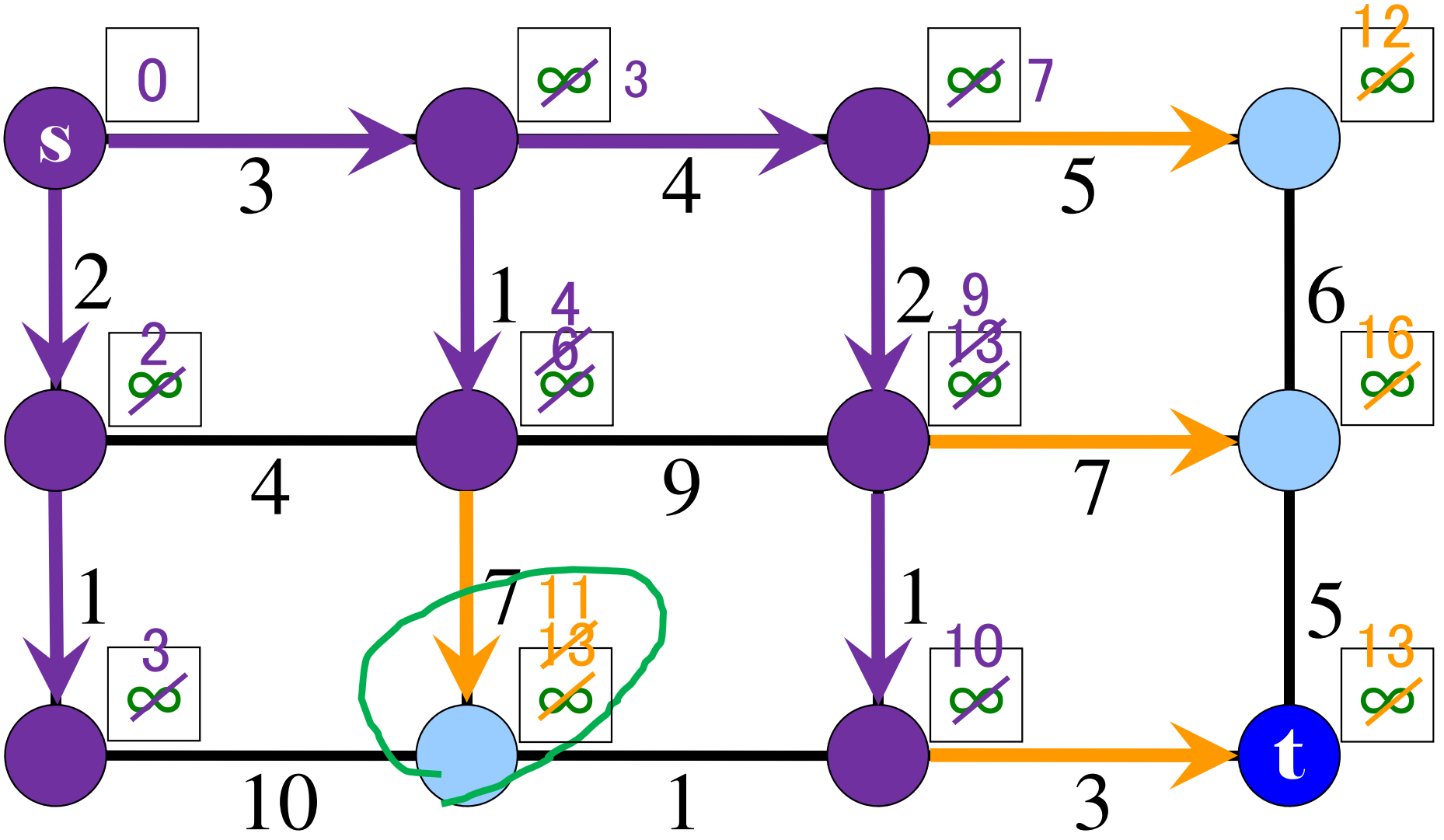
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



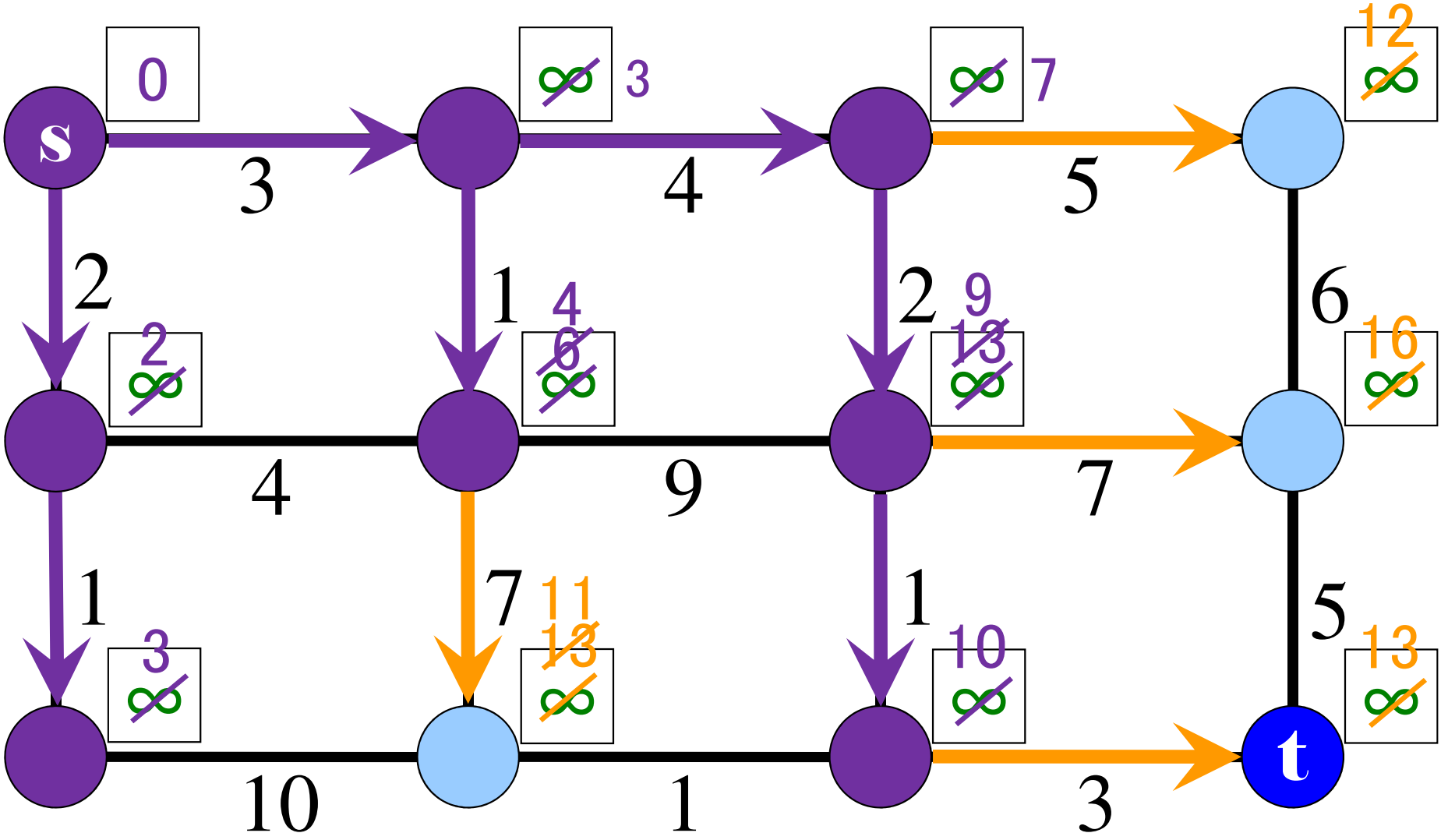
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

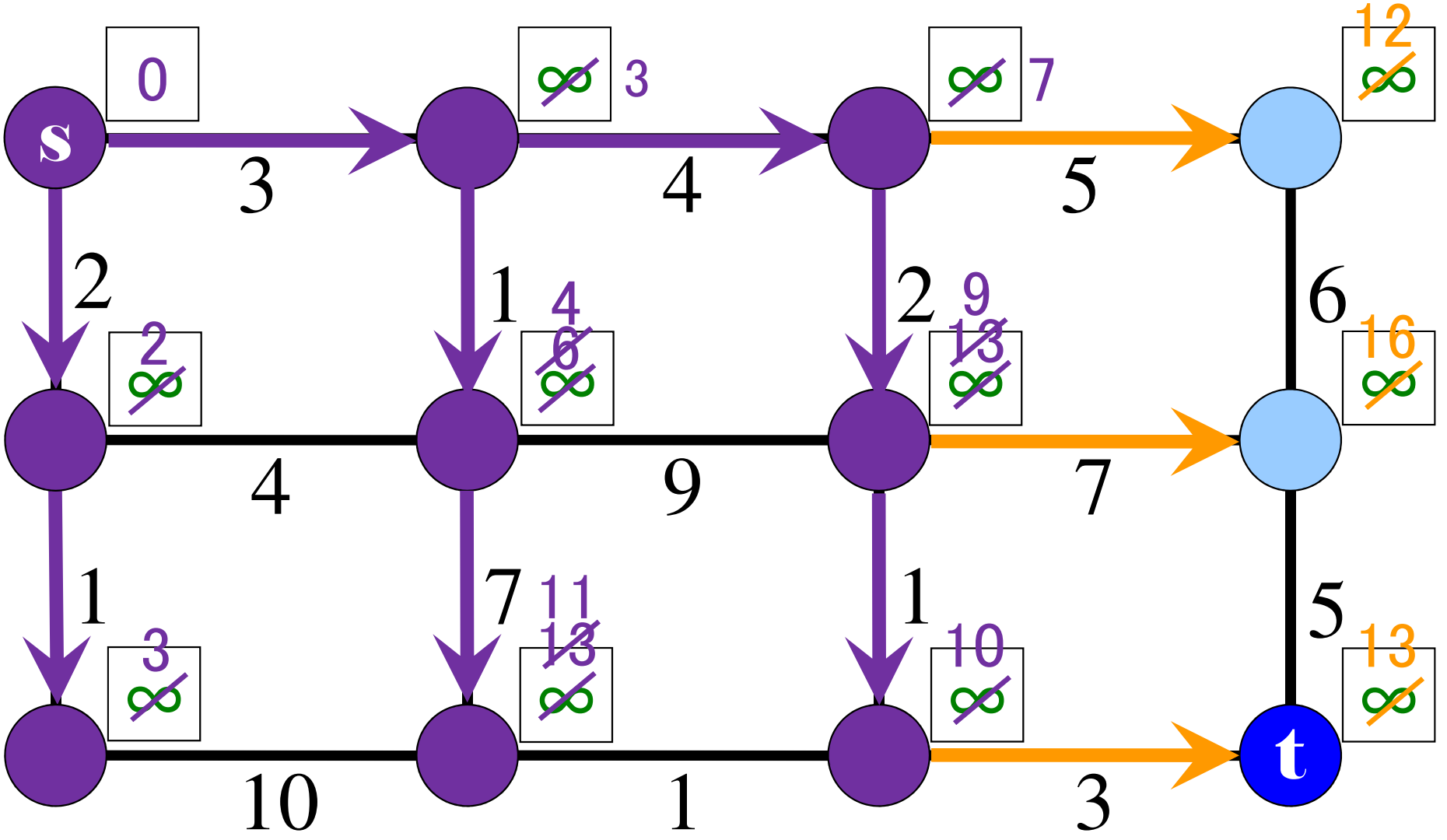


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

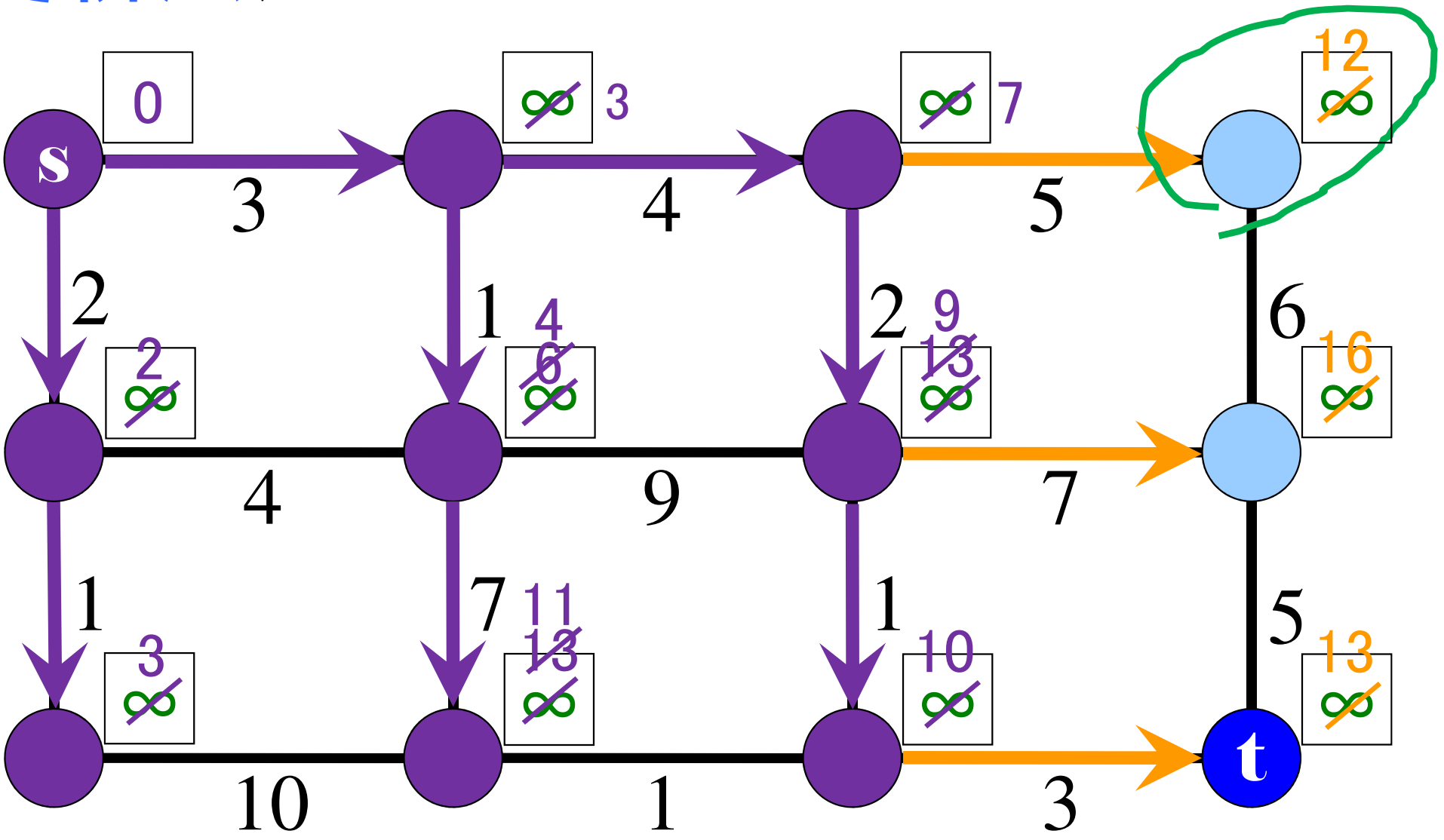
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



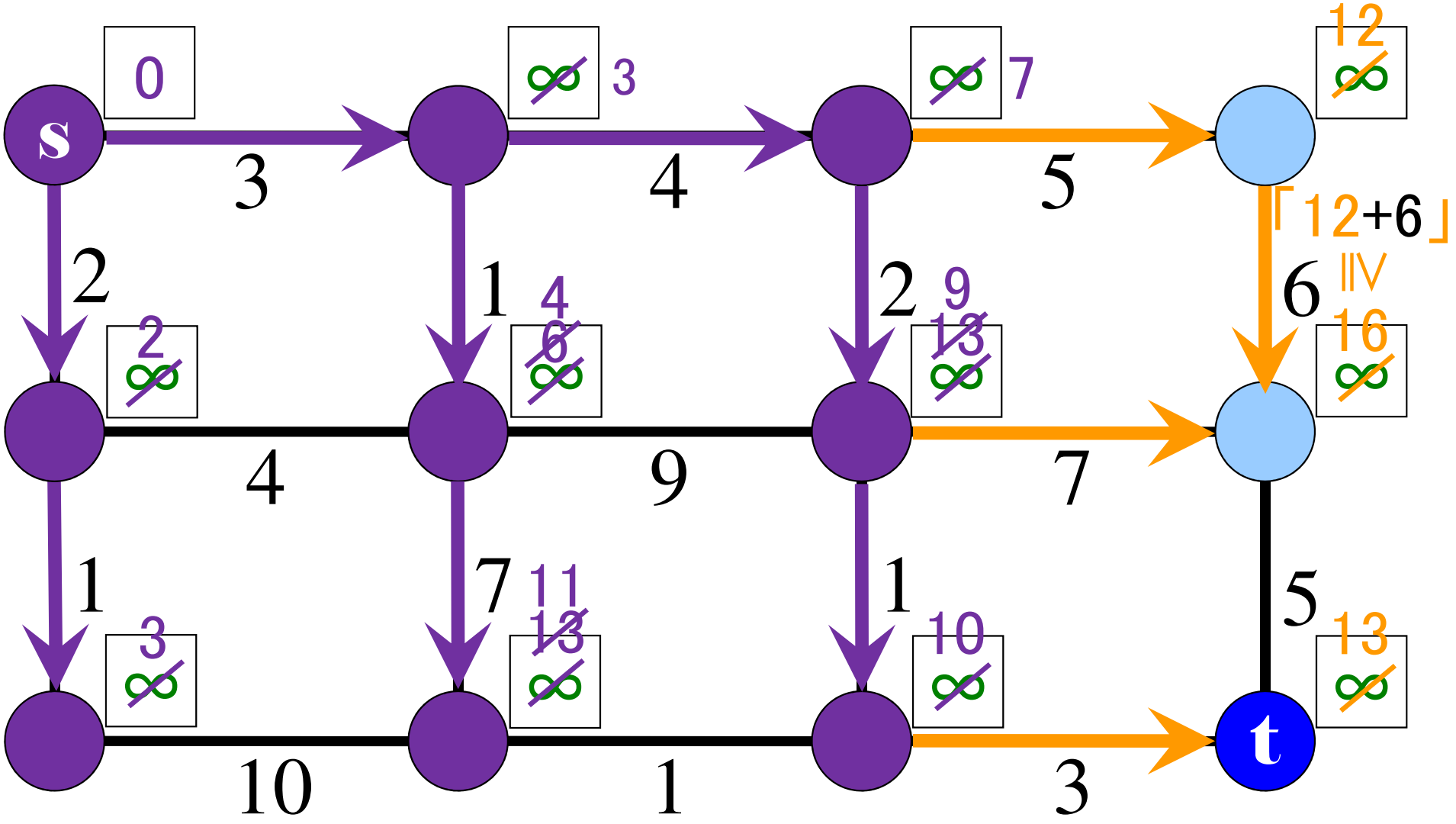
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

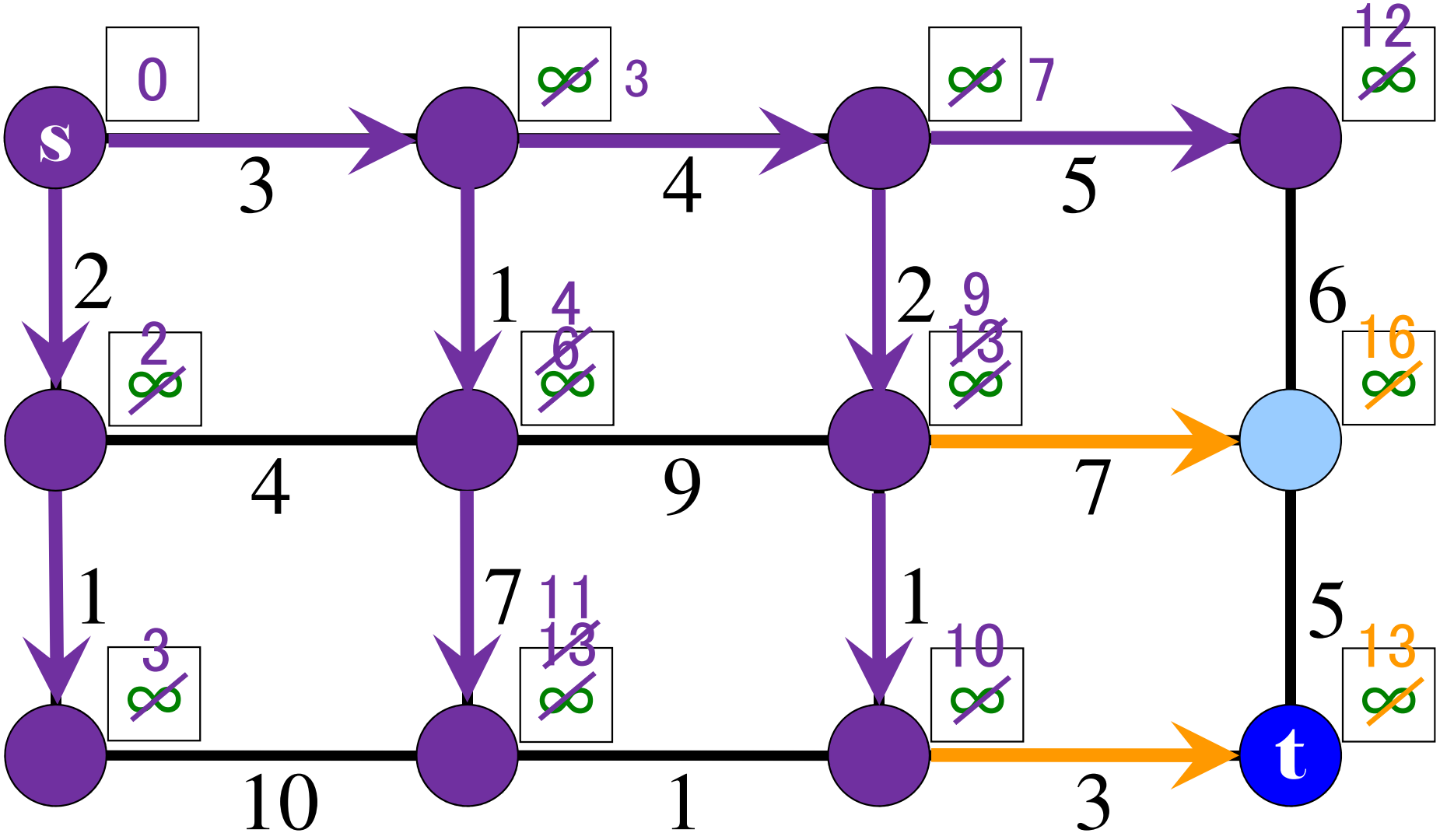


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

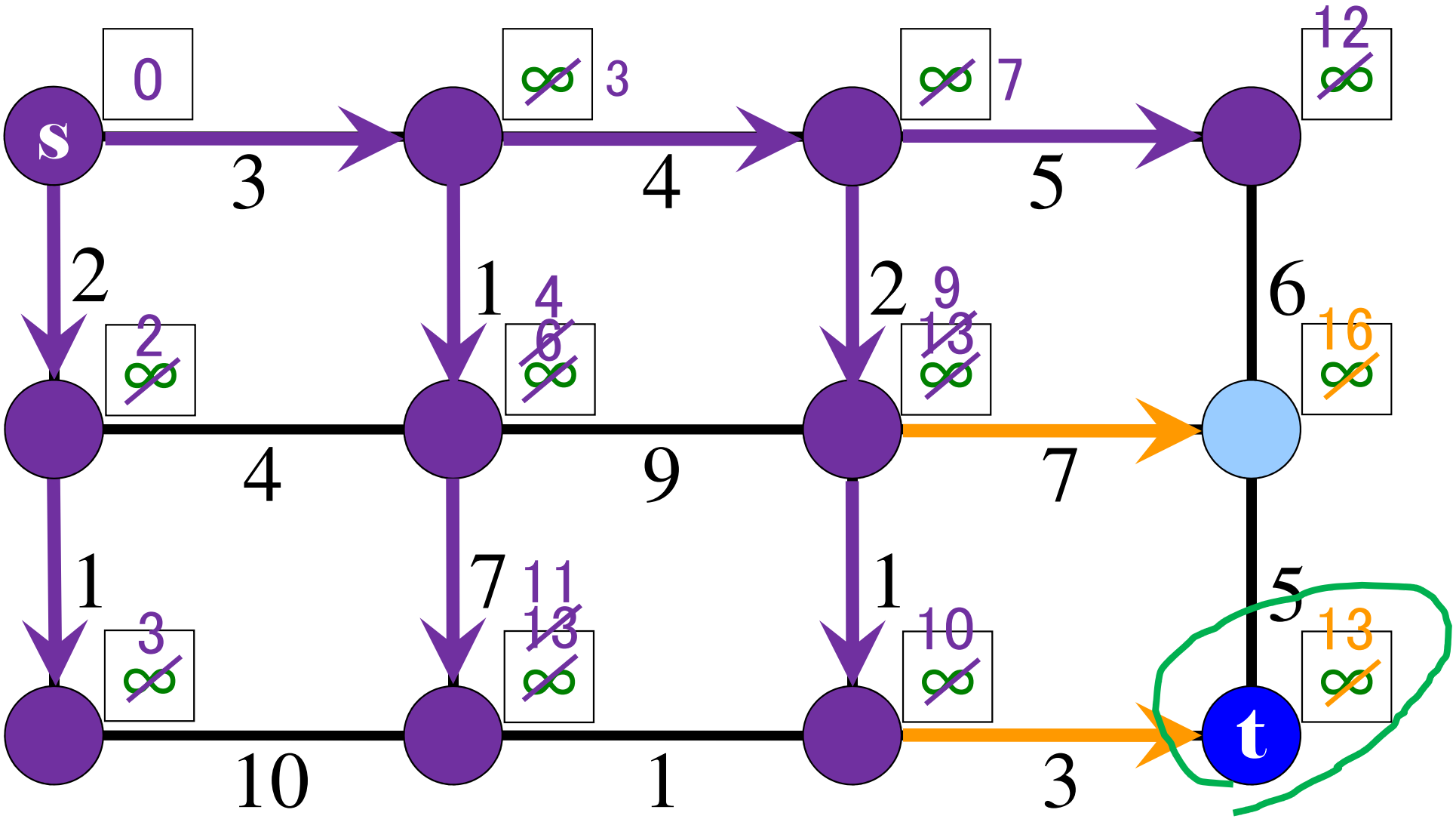
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



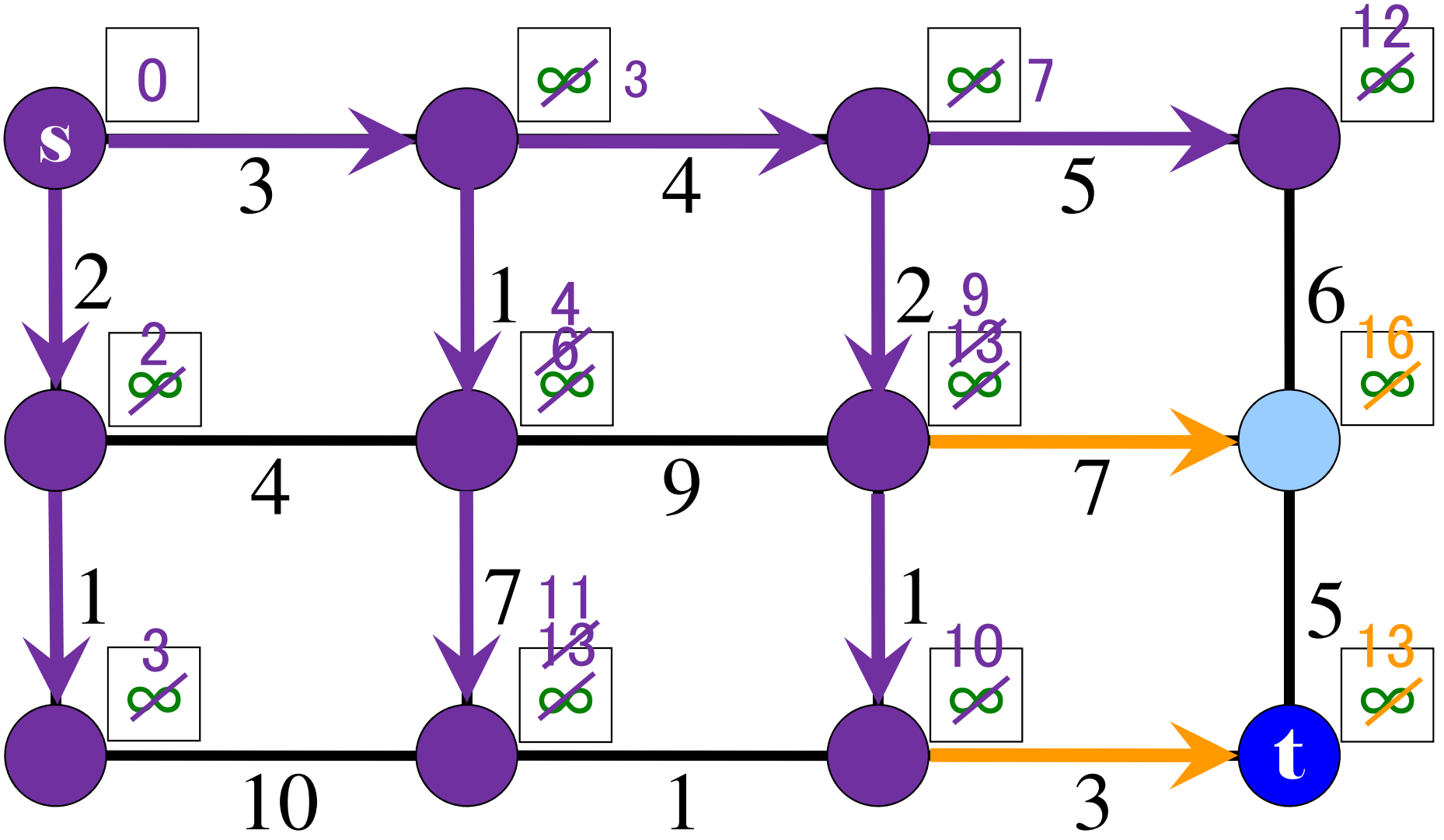
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

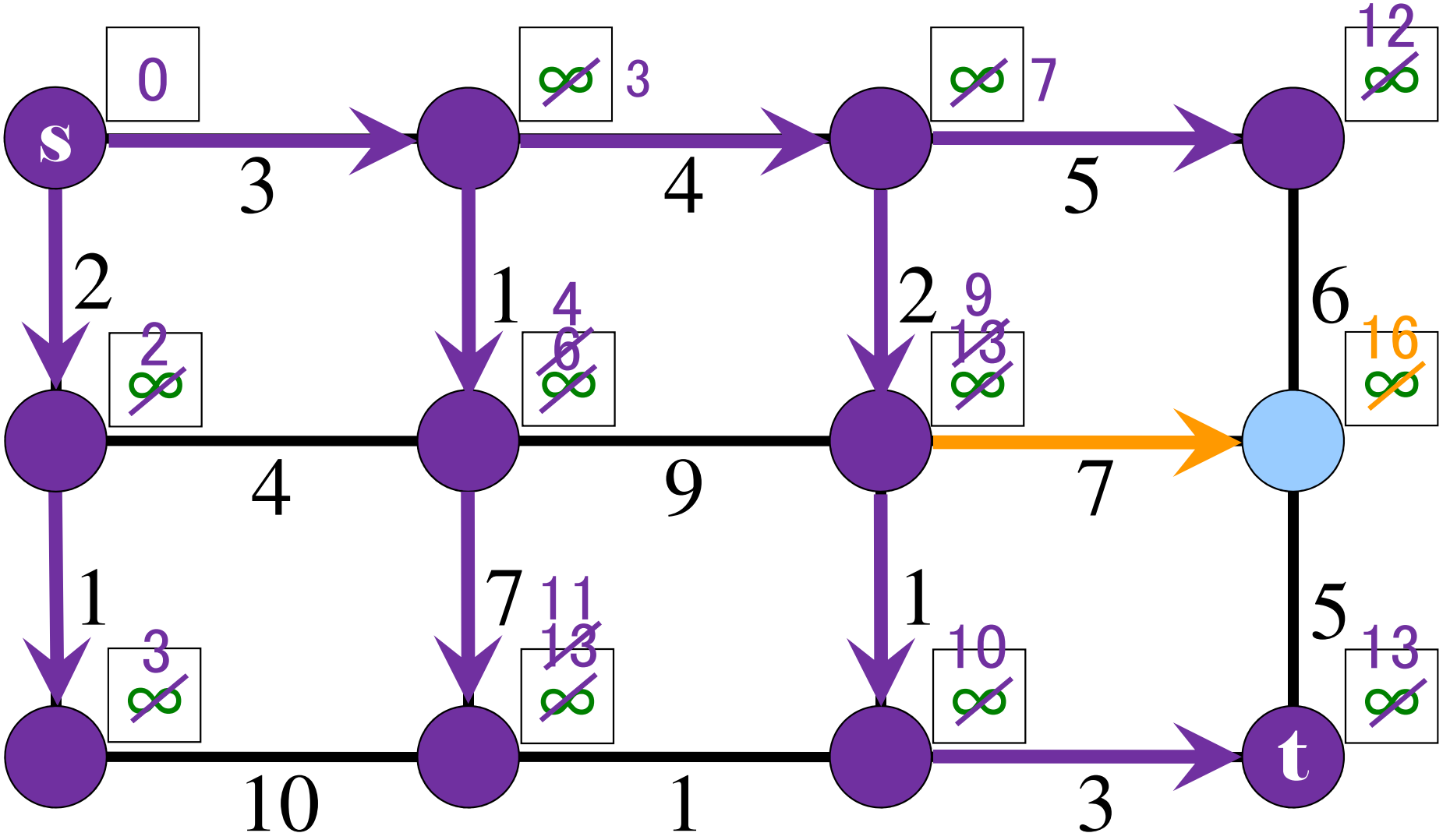


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

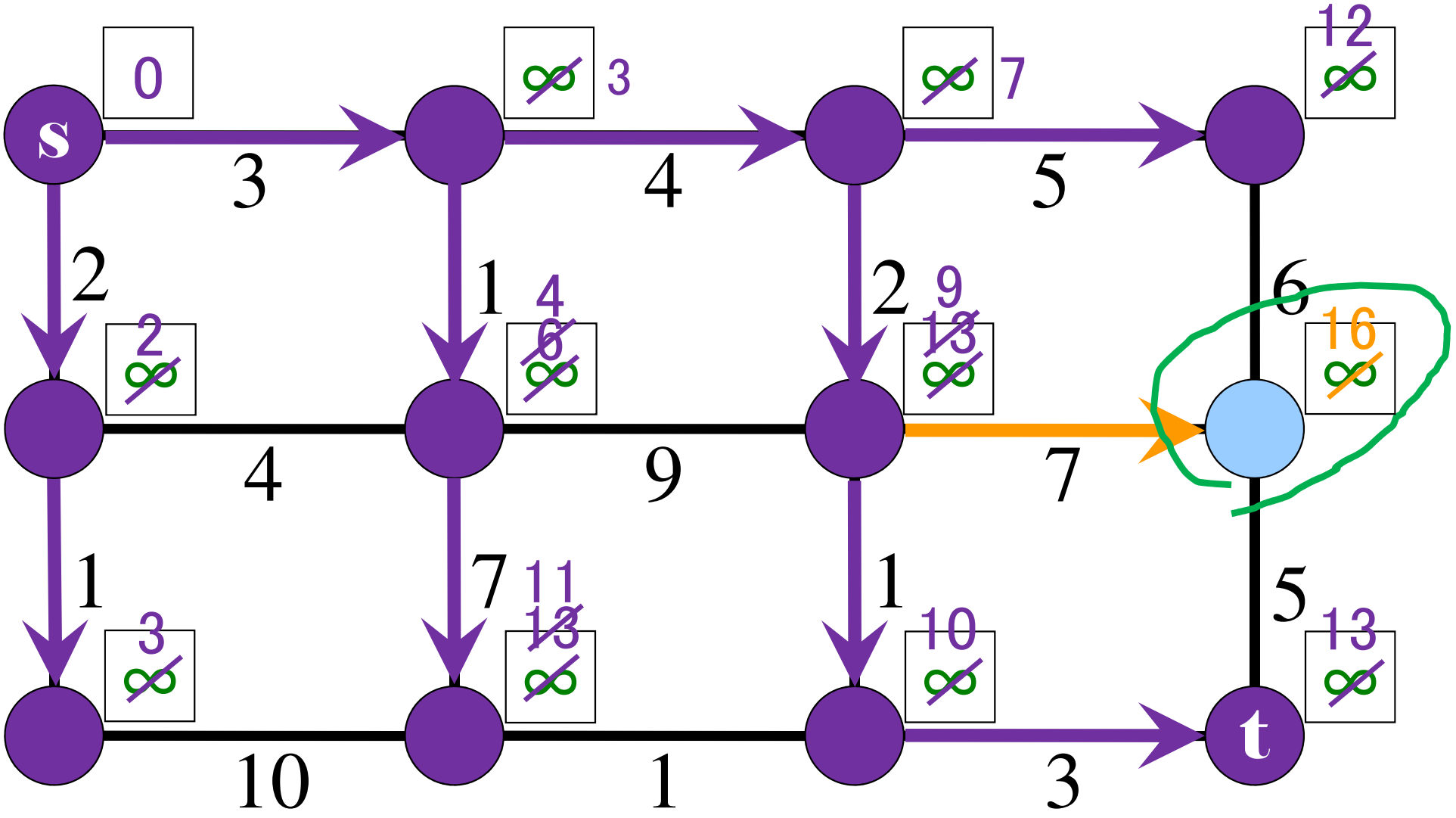
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



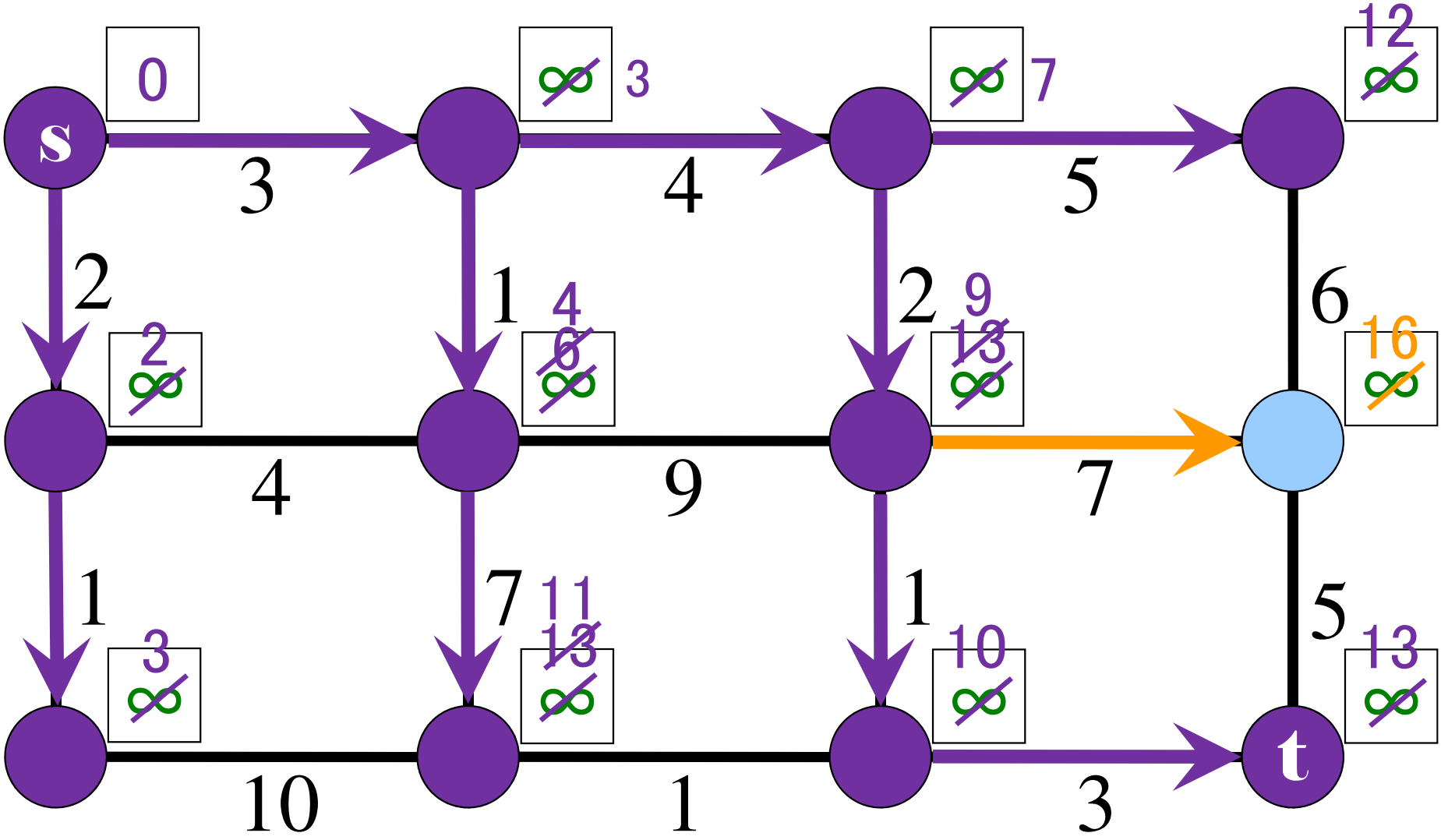
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

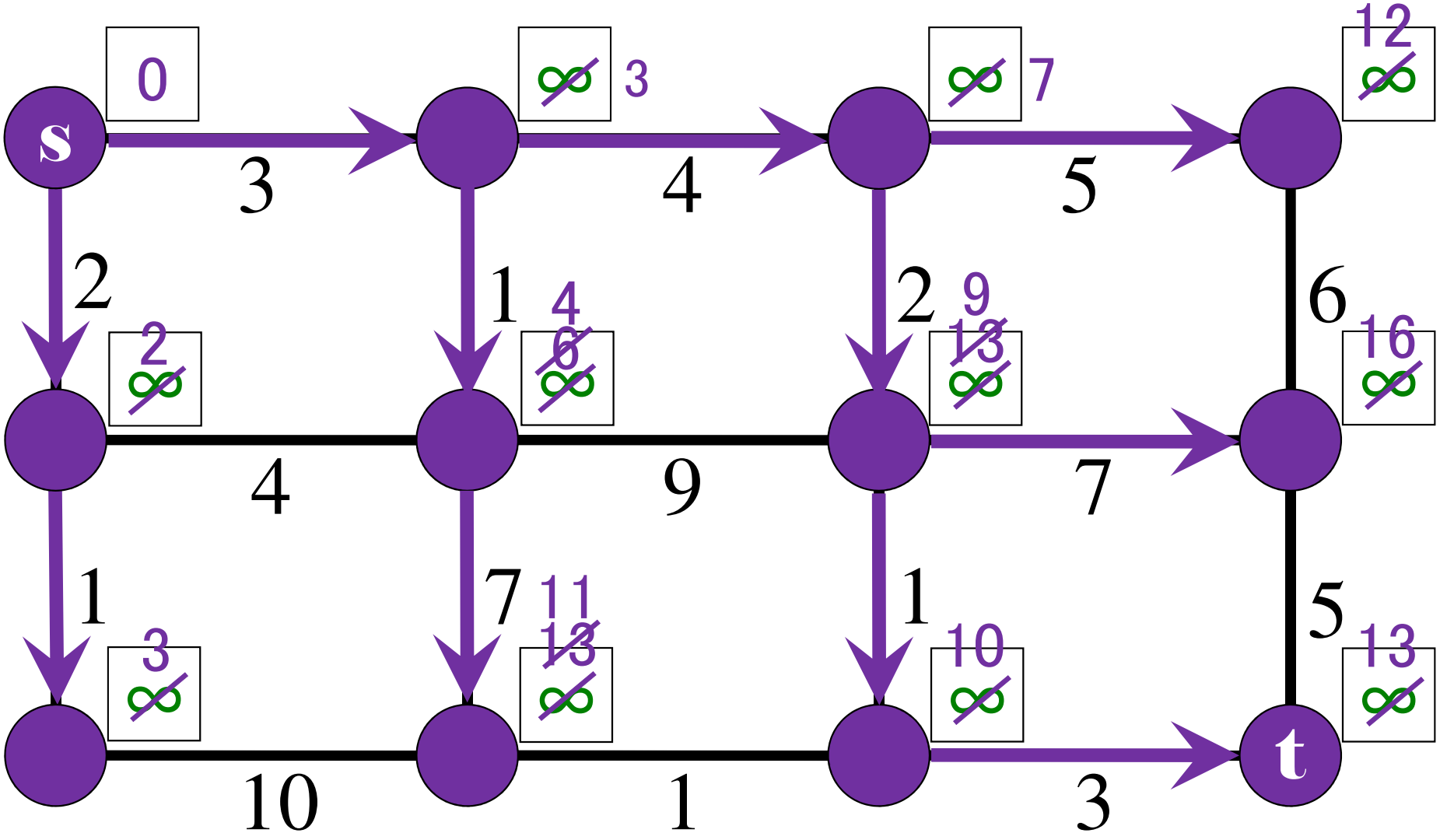


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

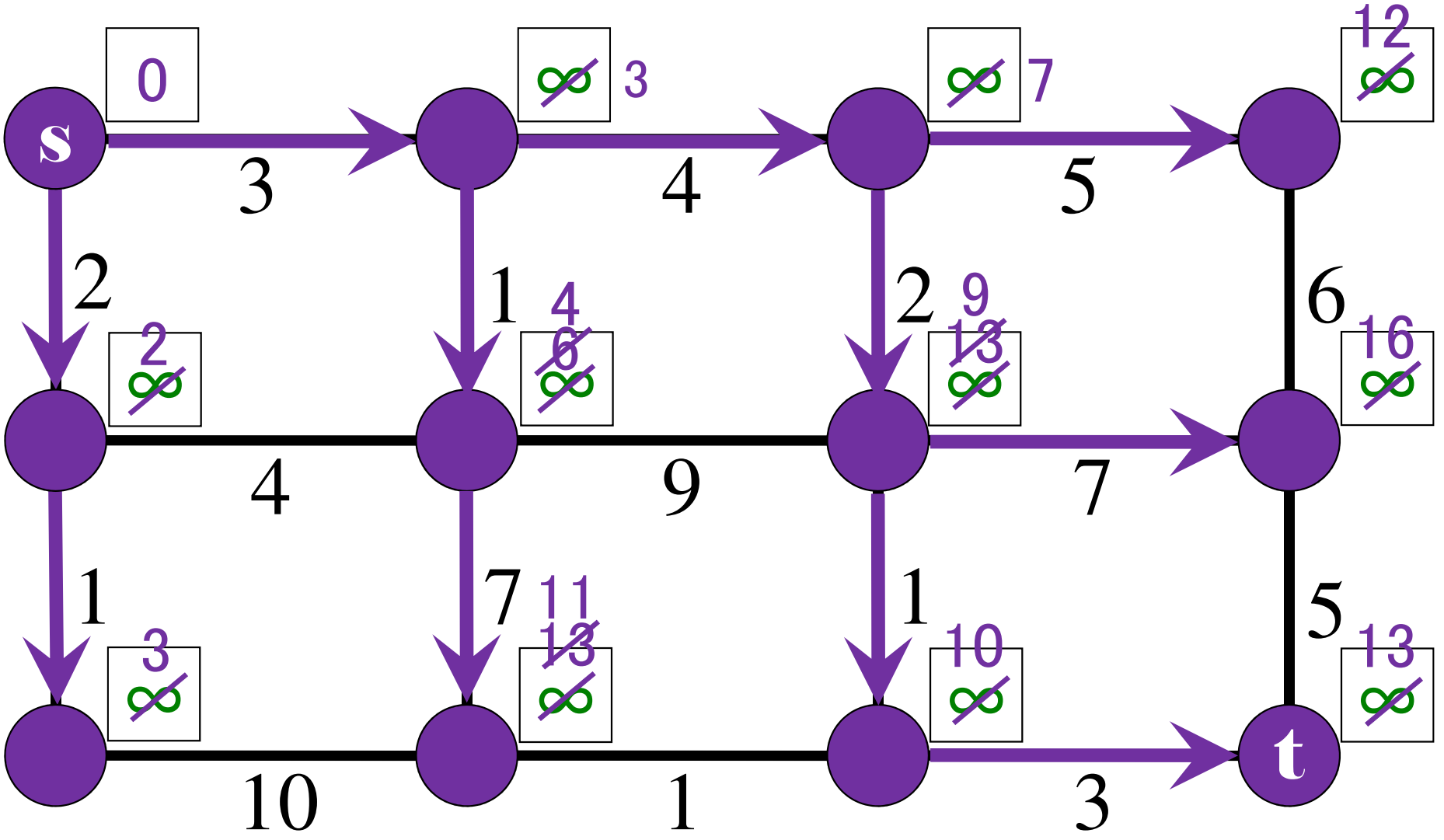
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



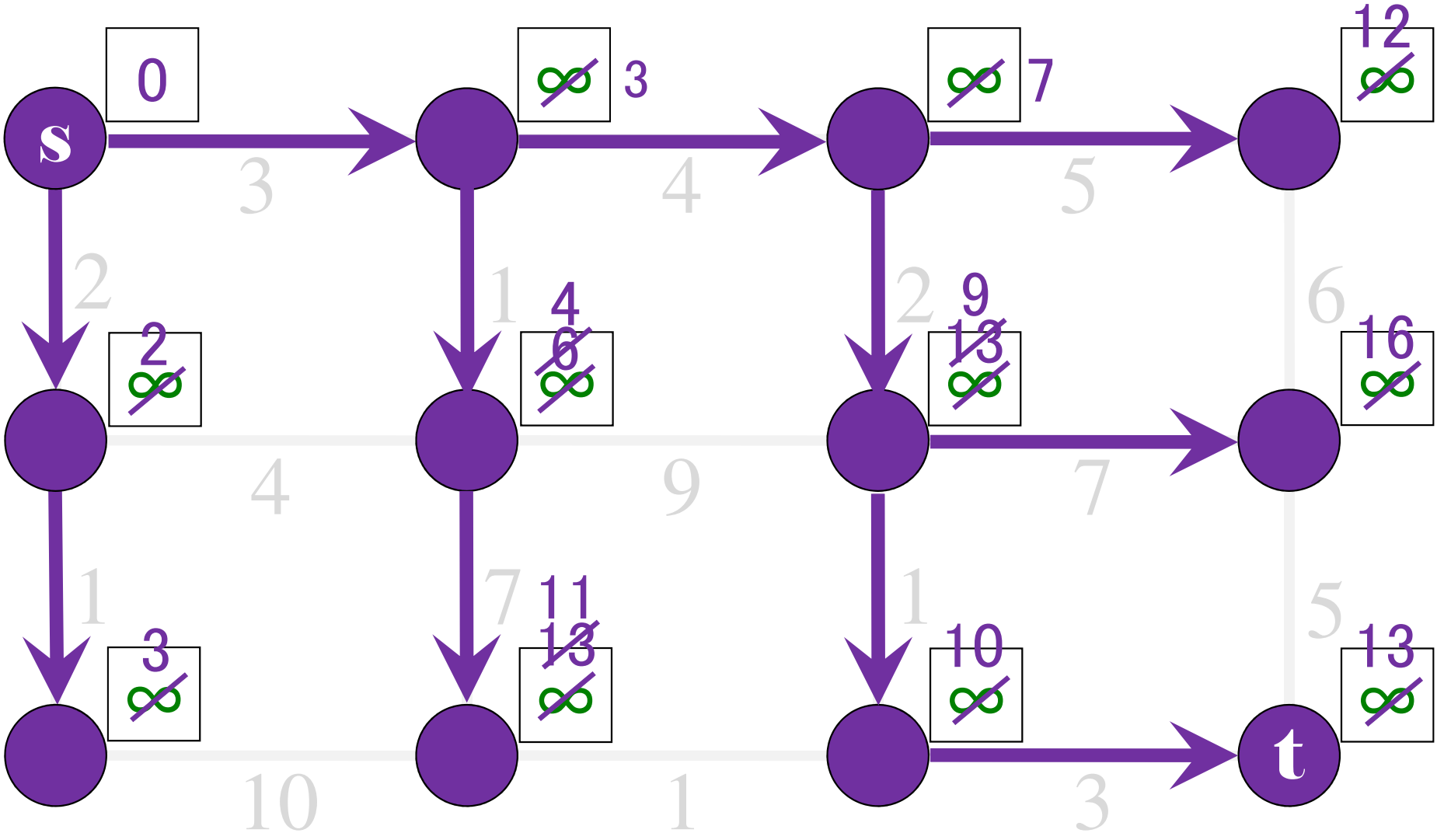
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



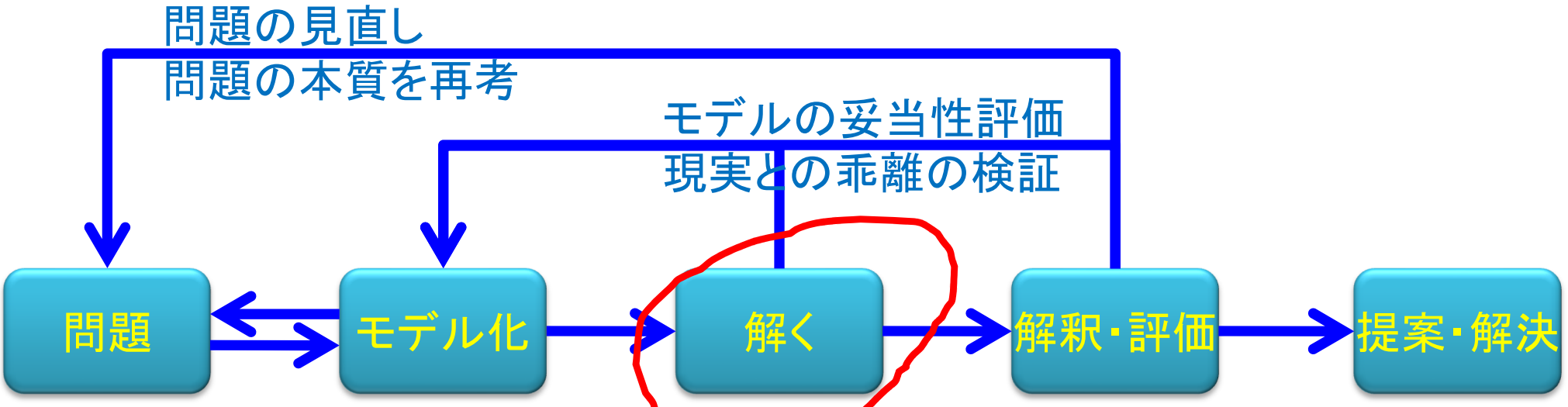
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

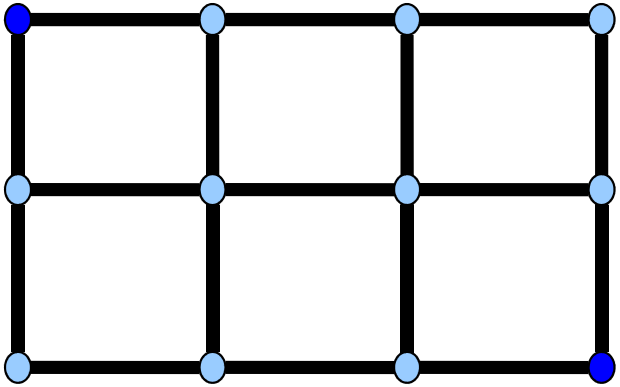
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

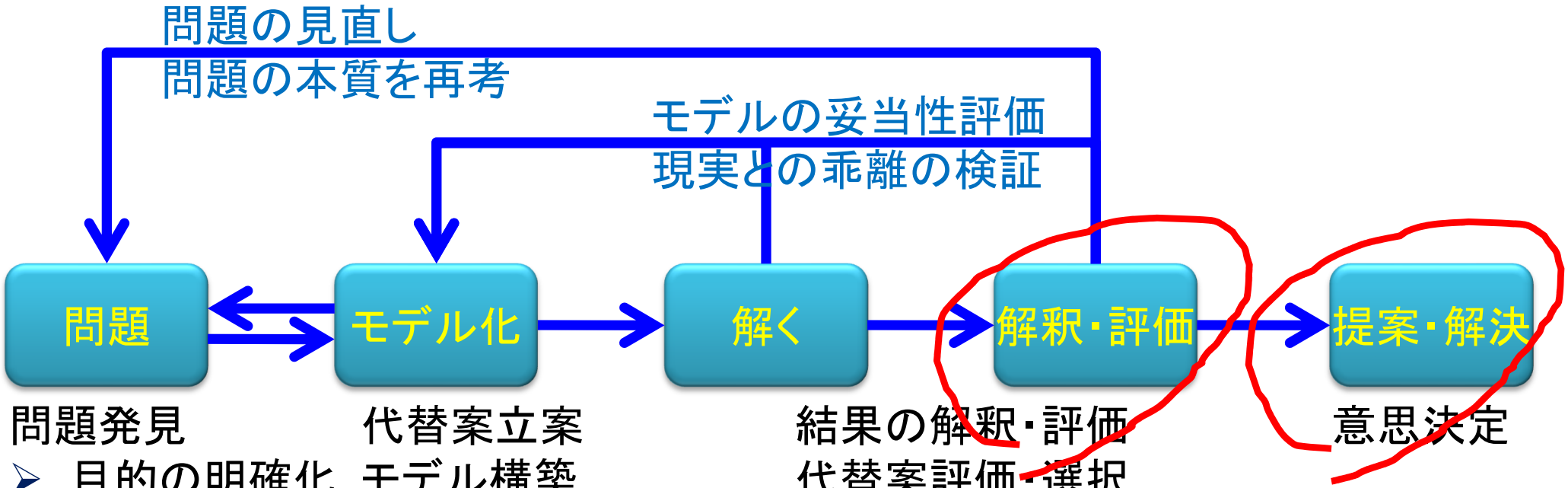
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

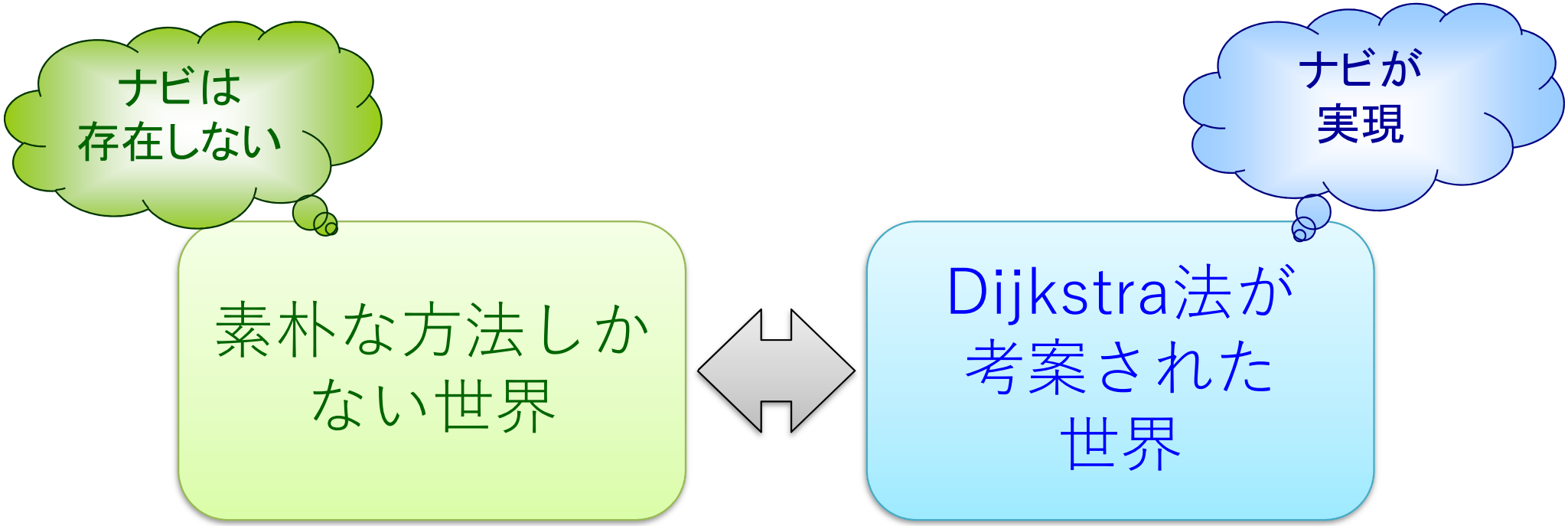
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

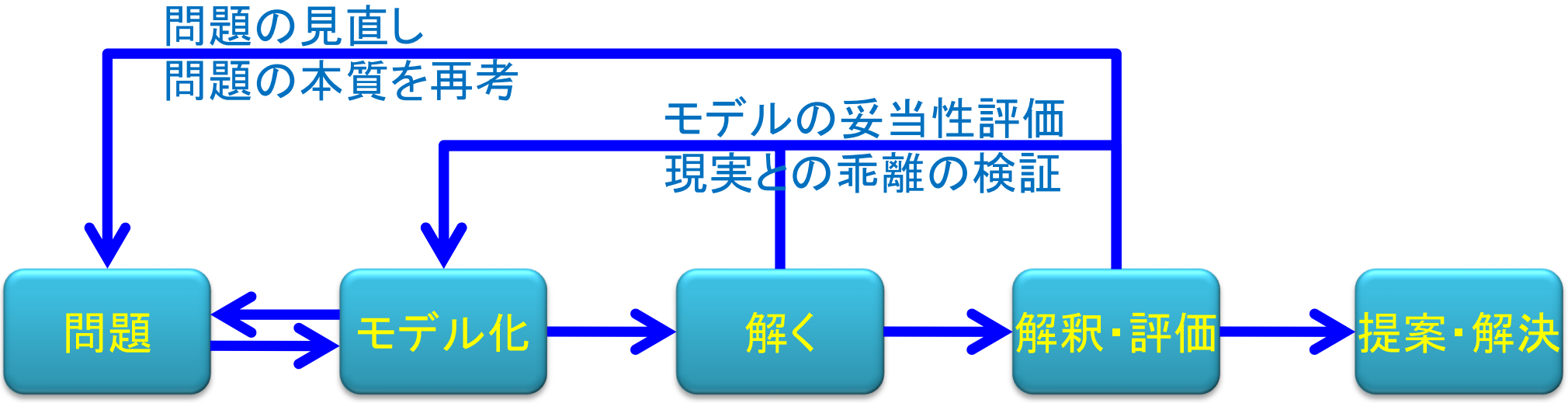
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



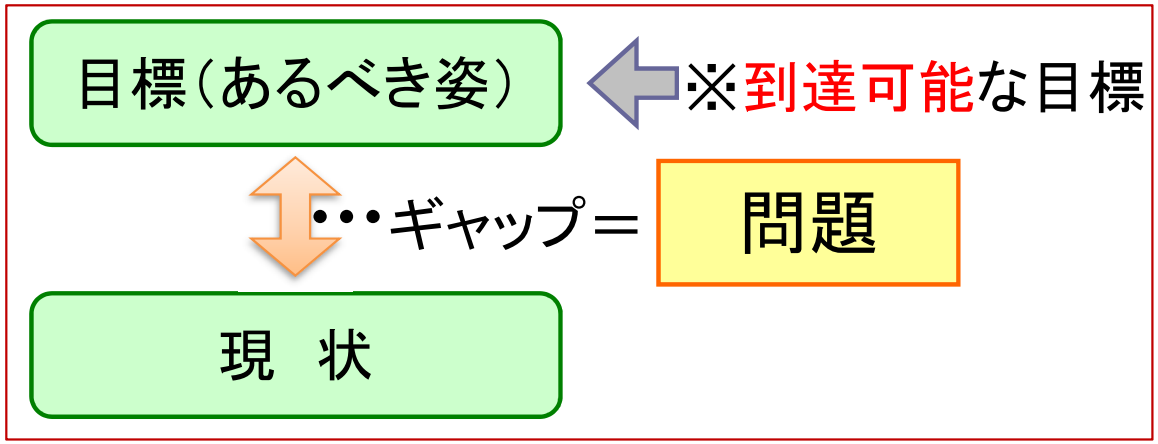
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

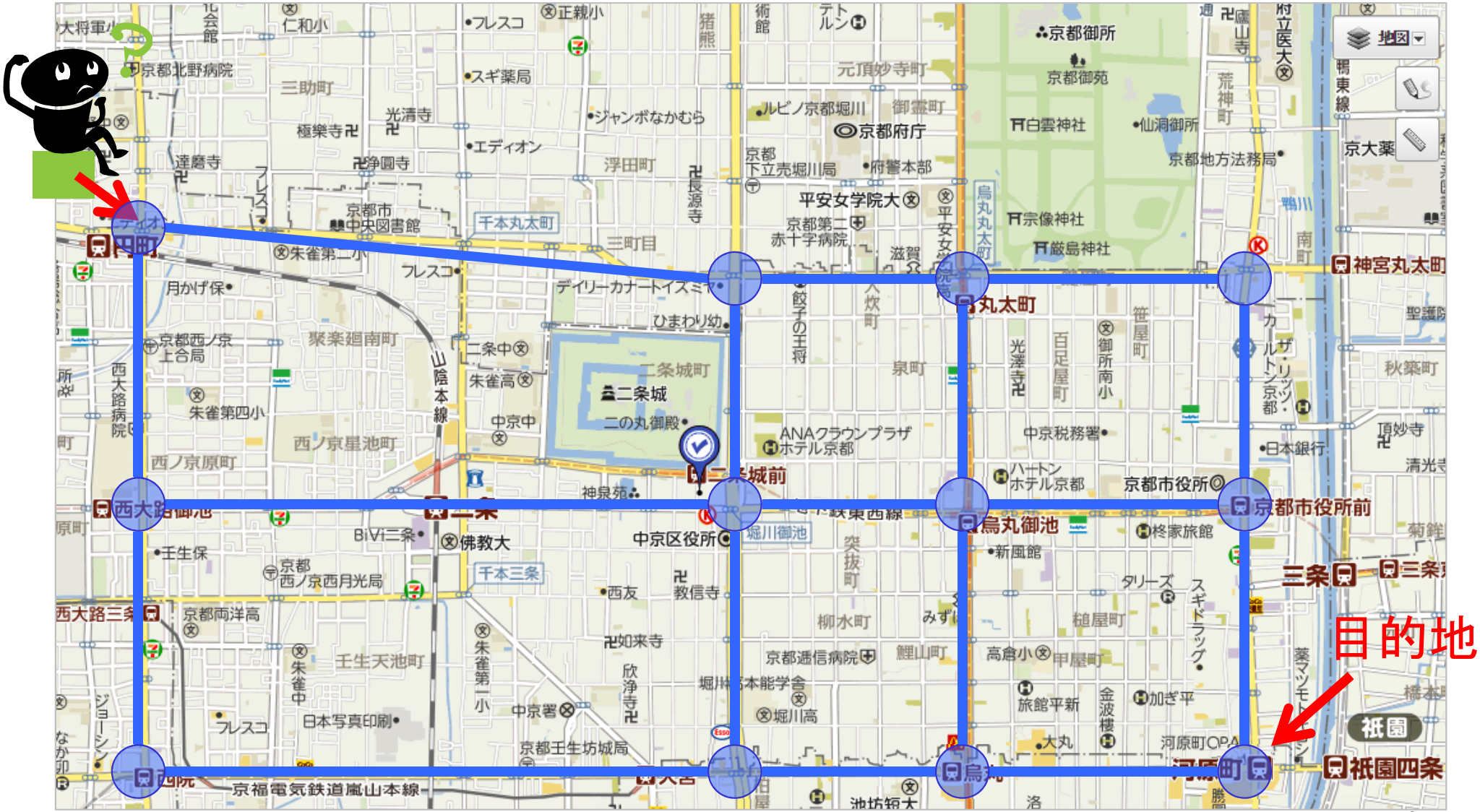
意思決定

問題の定義



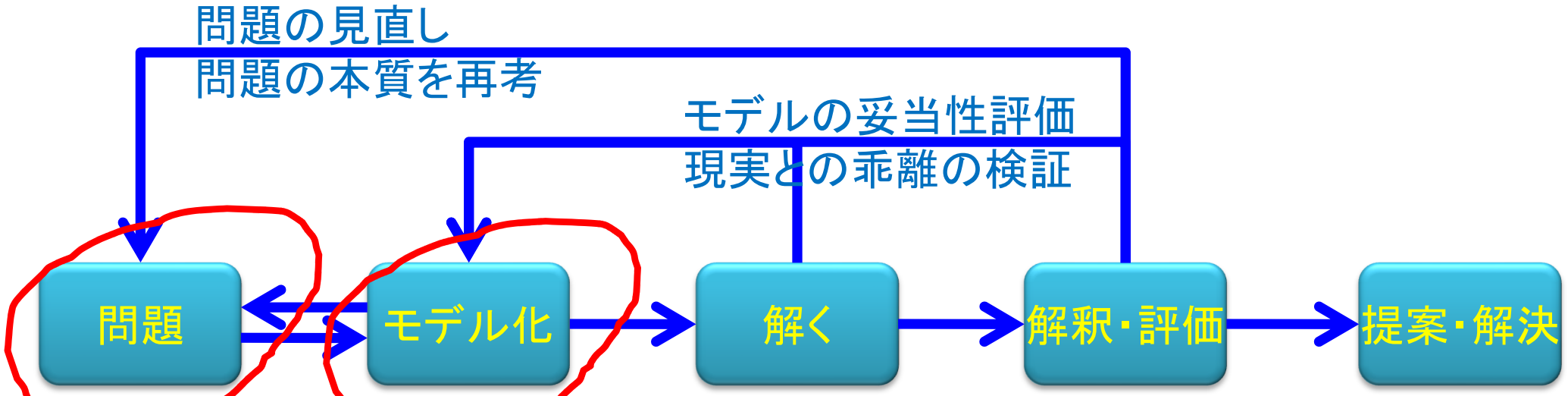
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

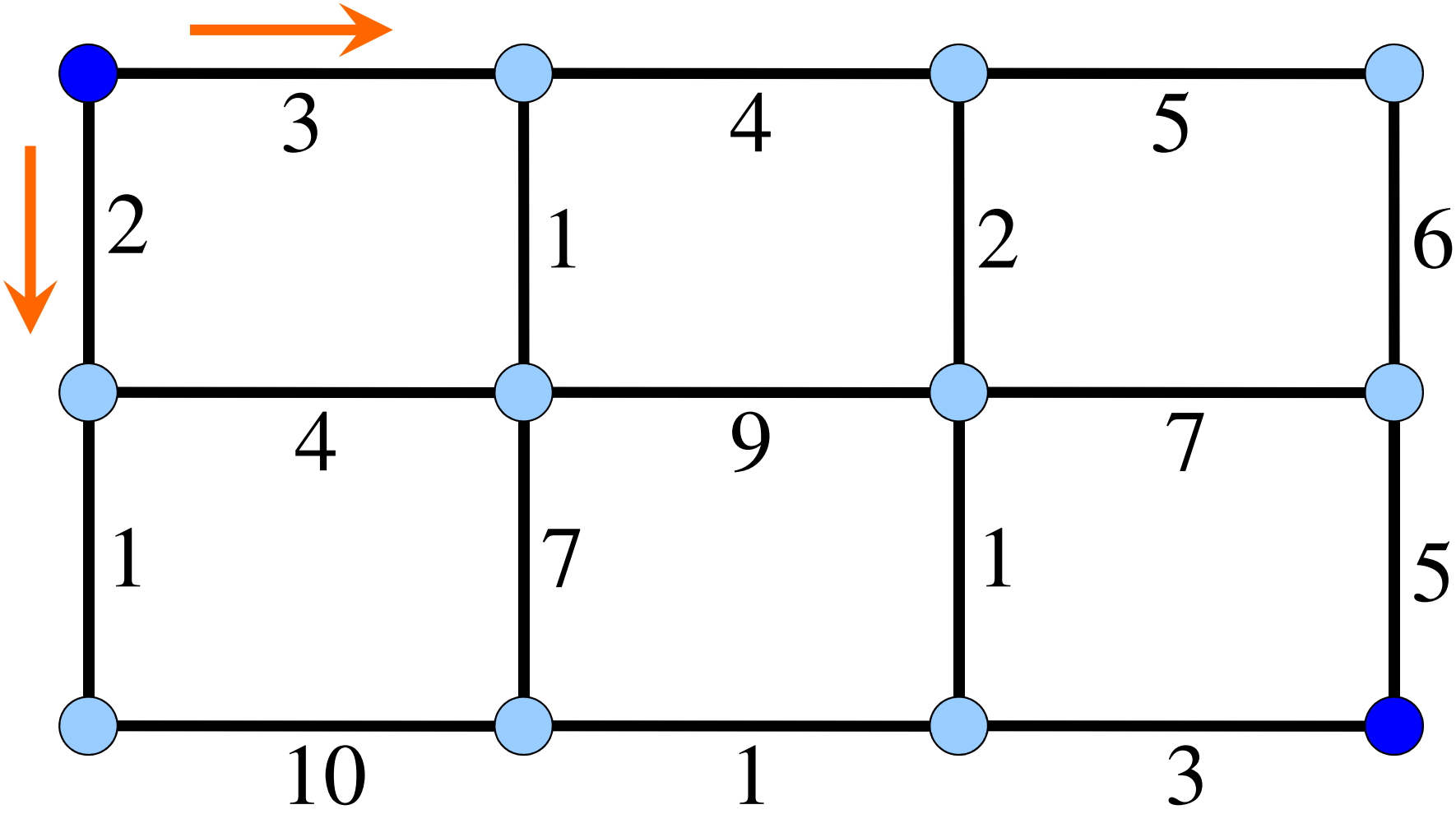
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

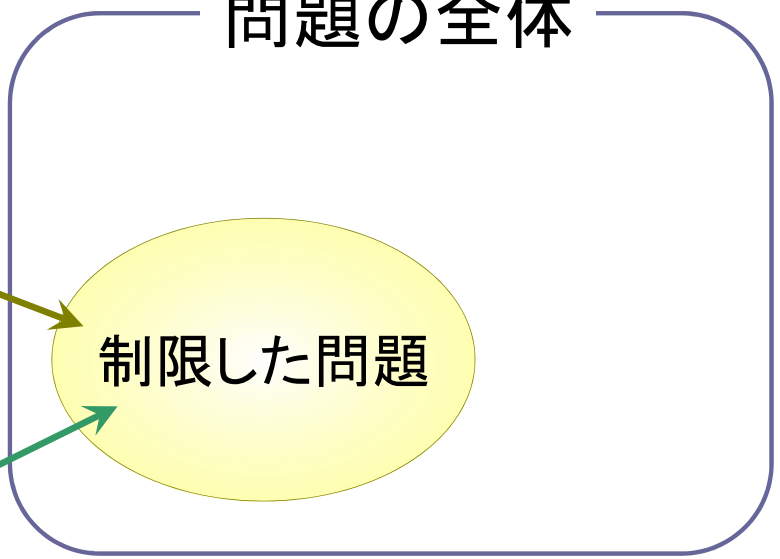


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

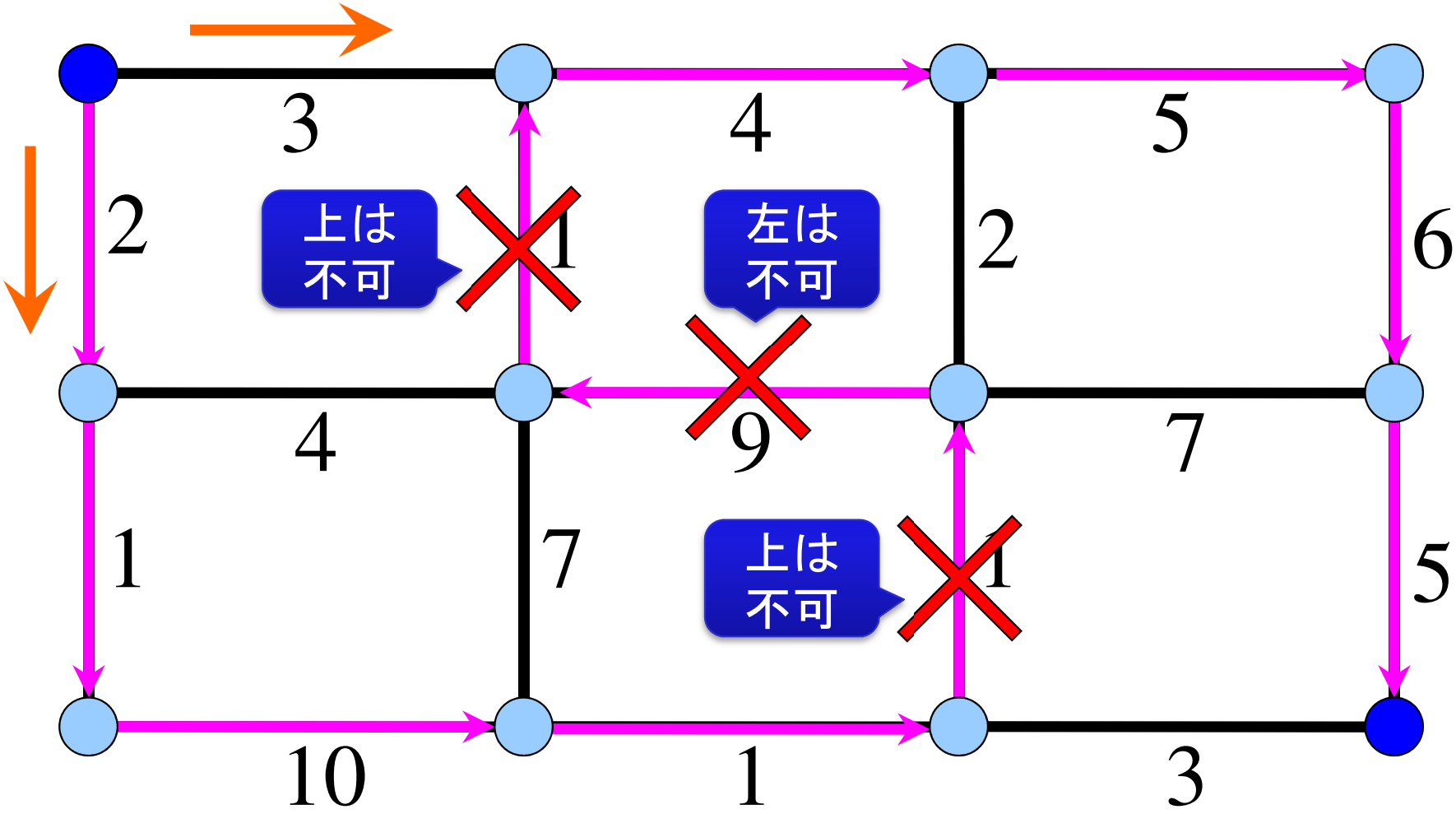
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

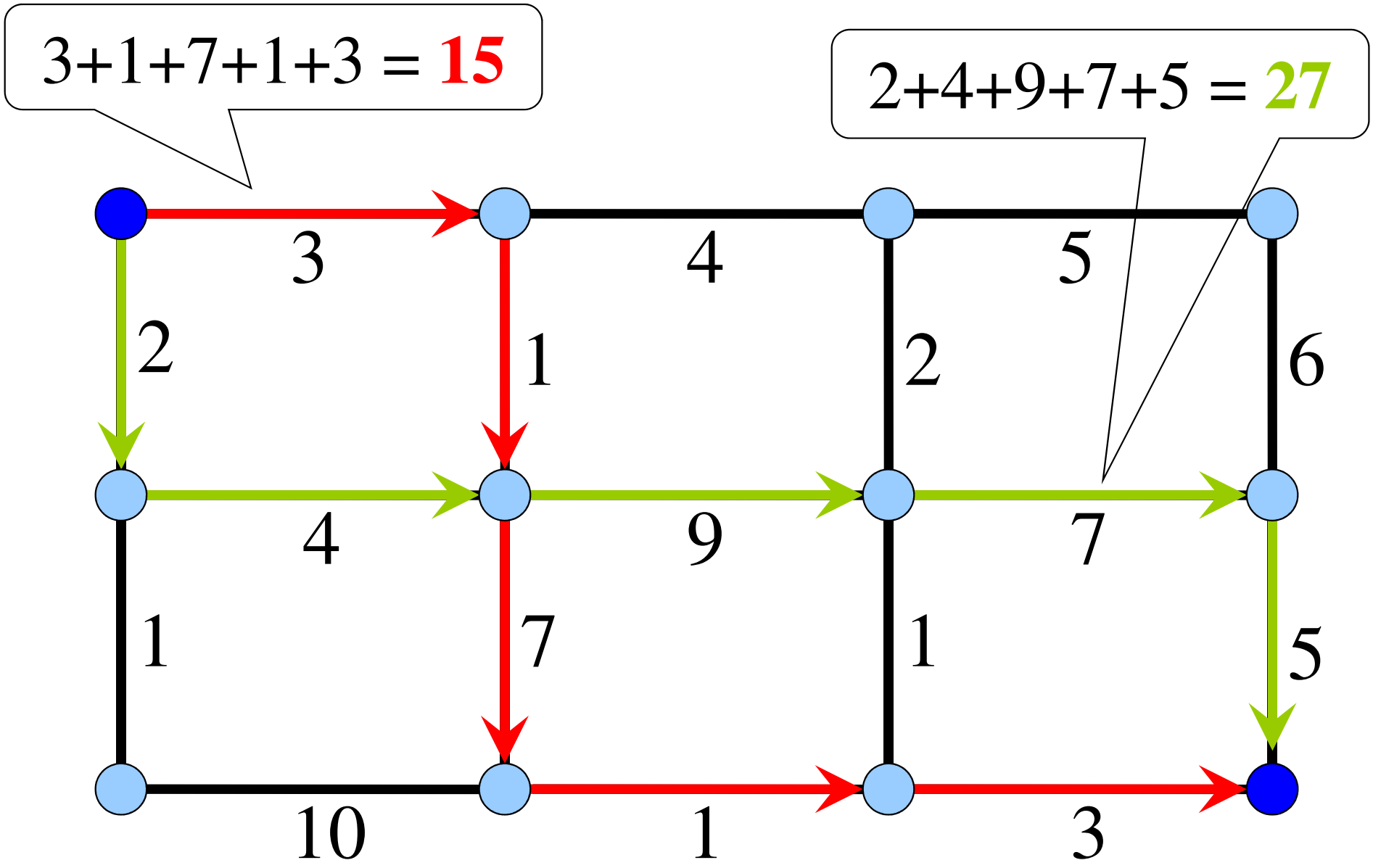
難しいなら易しくすればいいのさ！

制限した問題

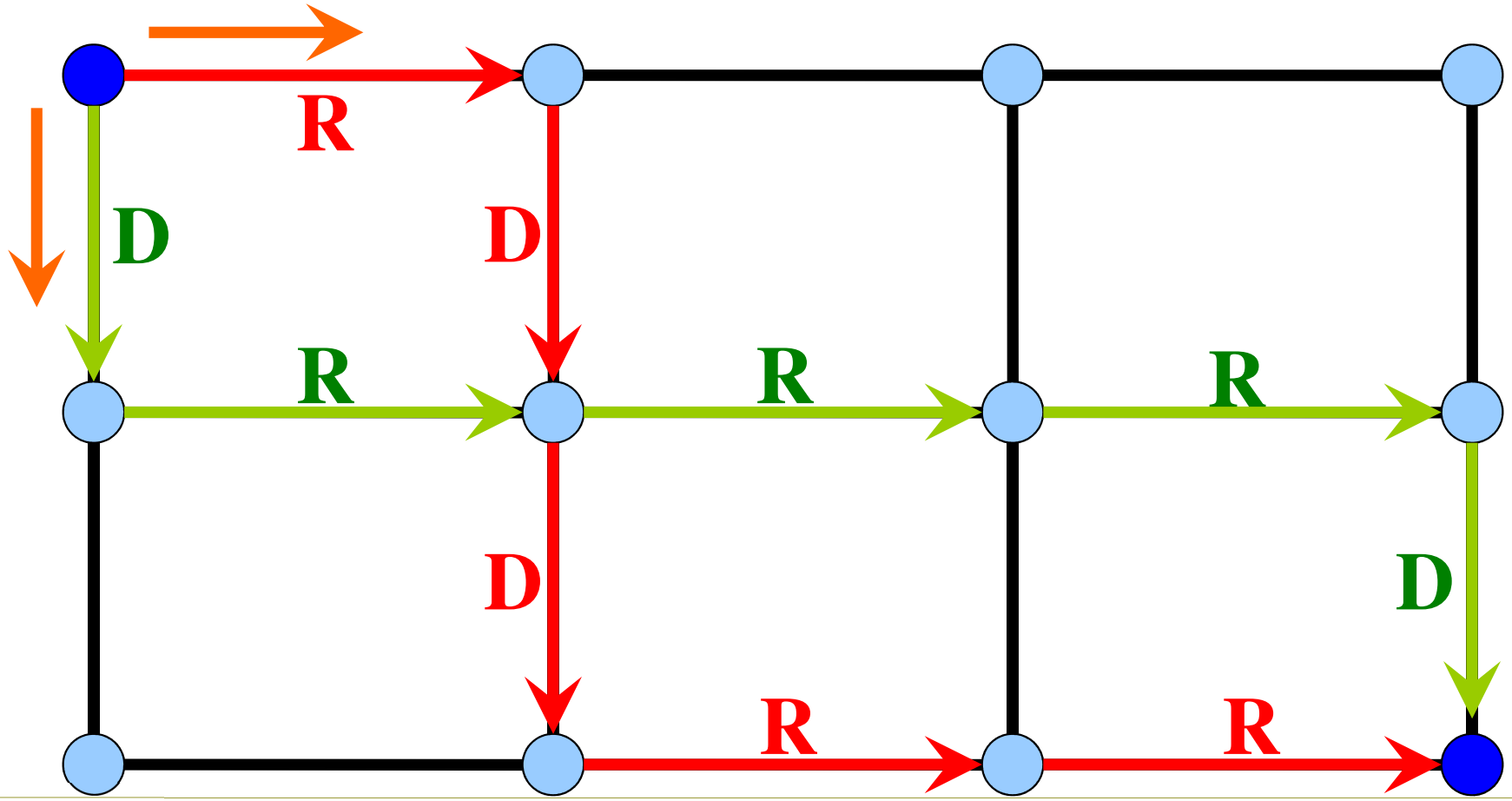
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



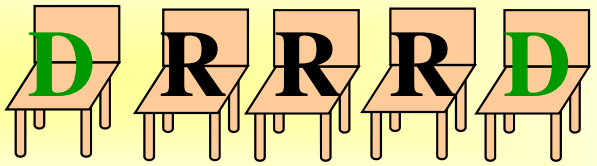
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



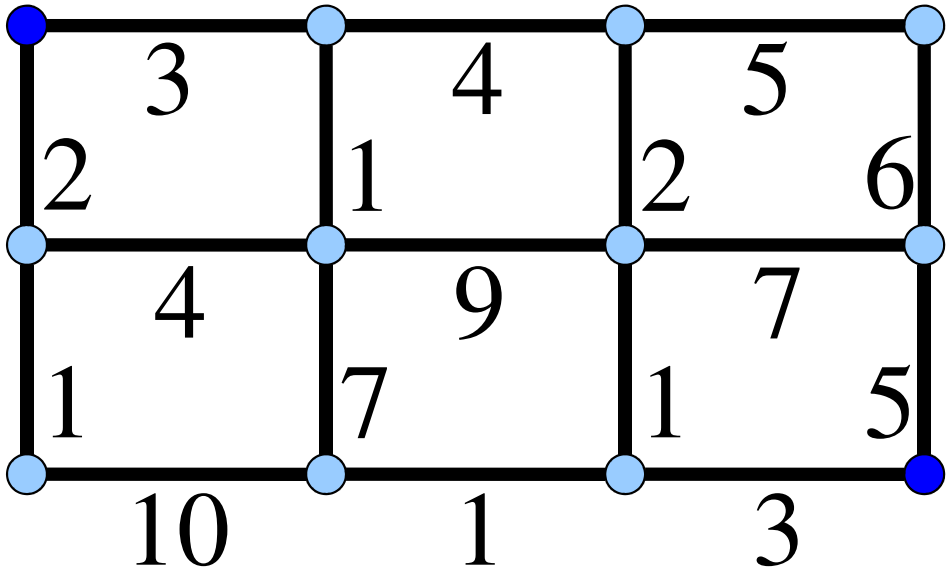
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

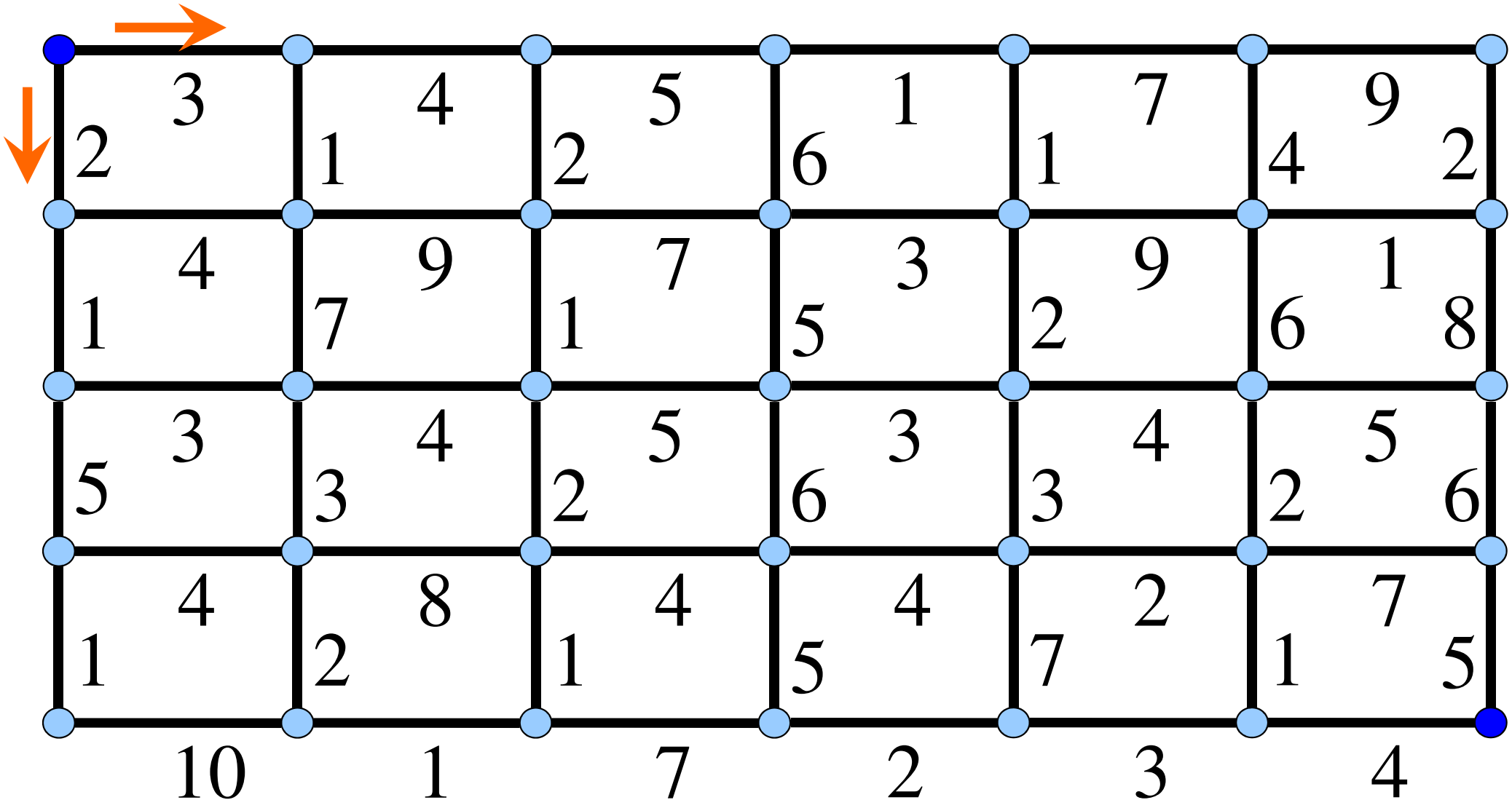
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

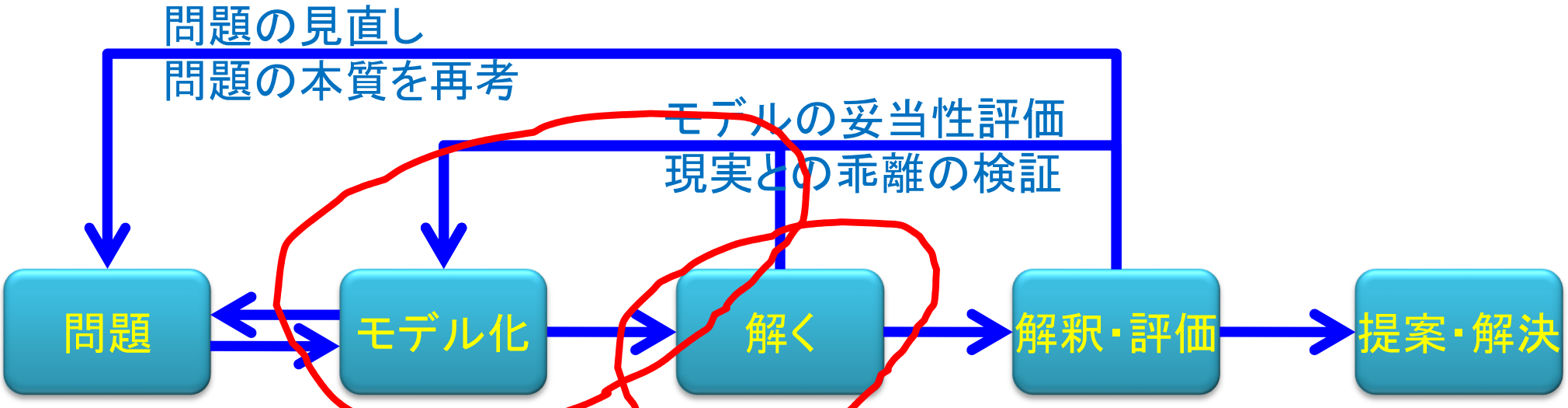
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

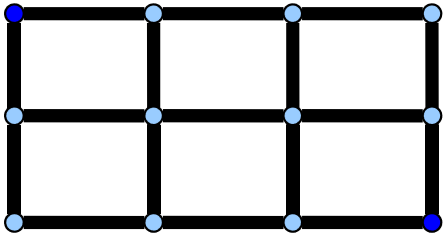
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

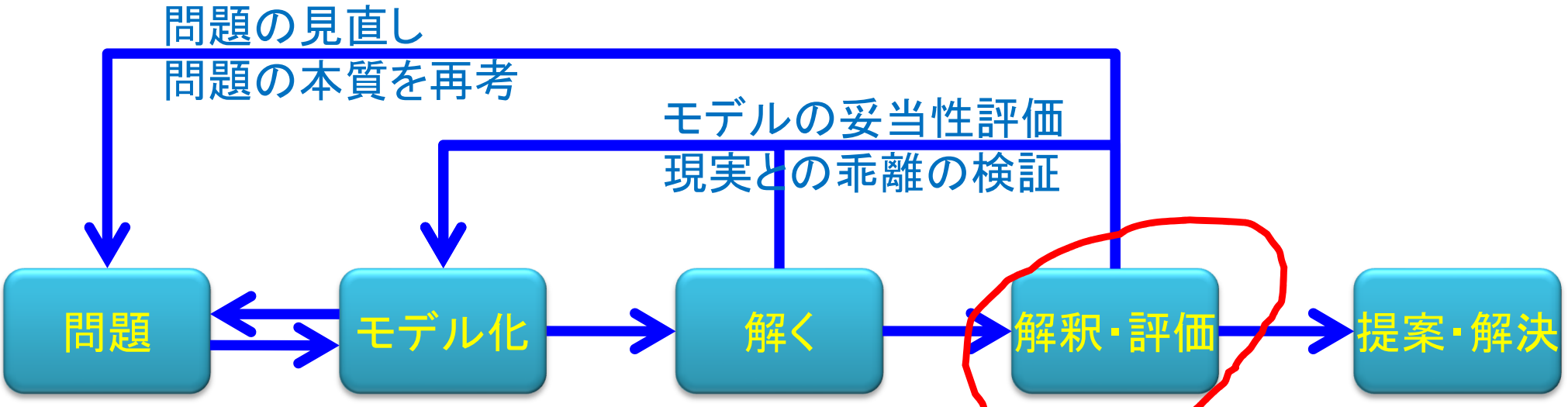
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

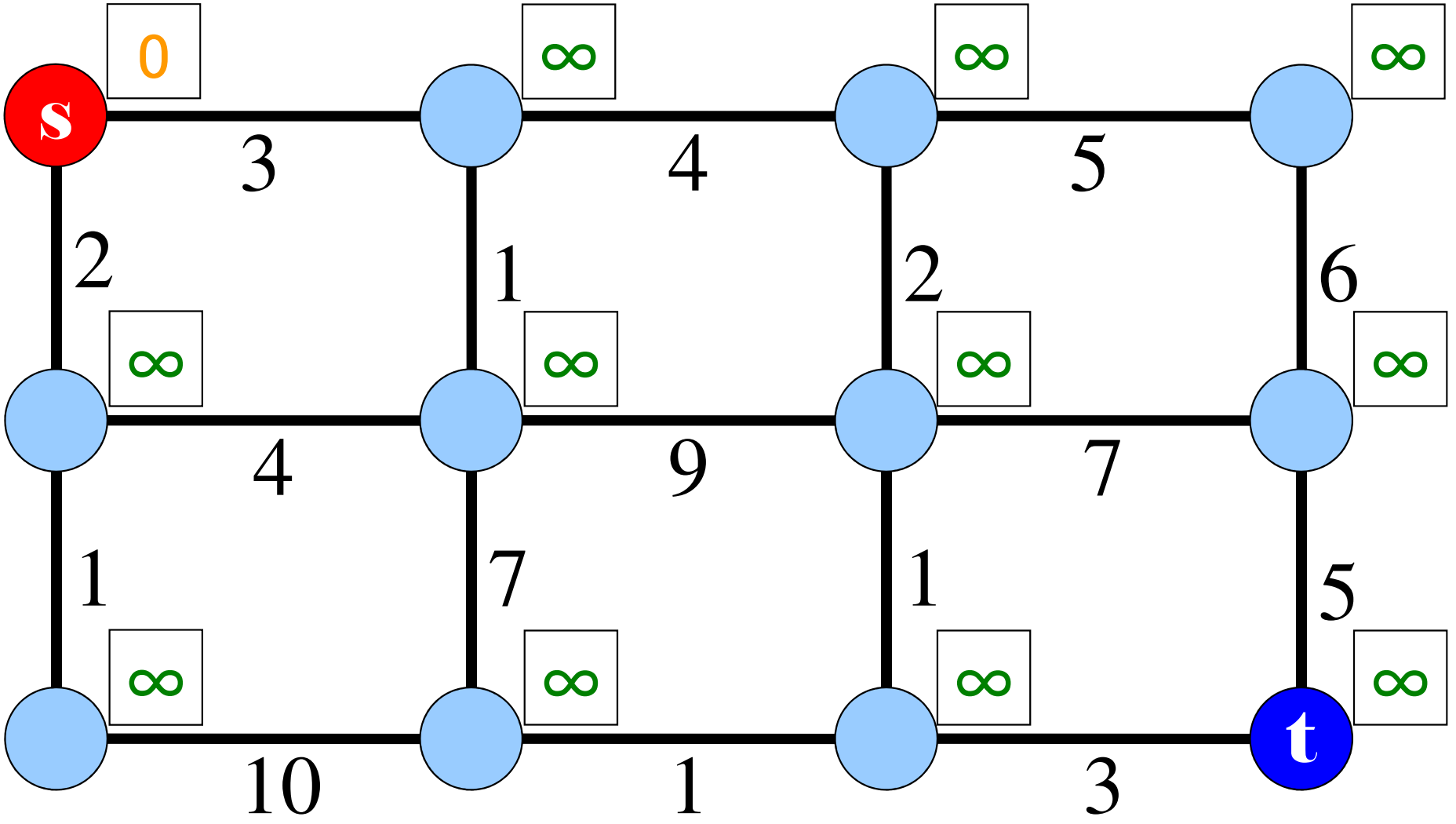
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

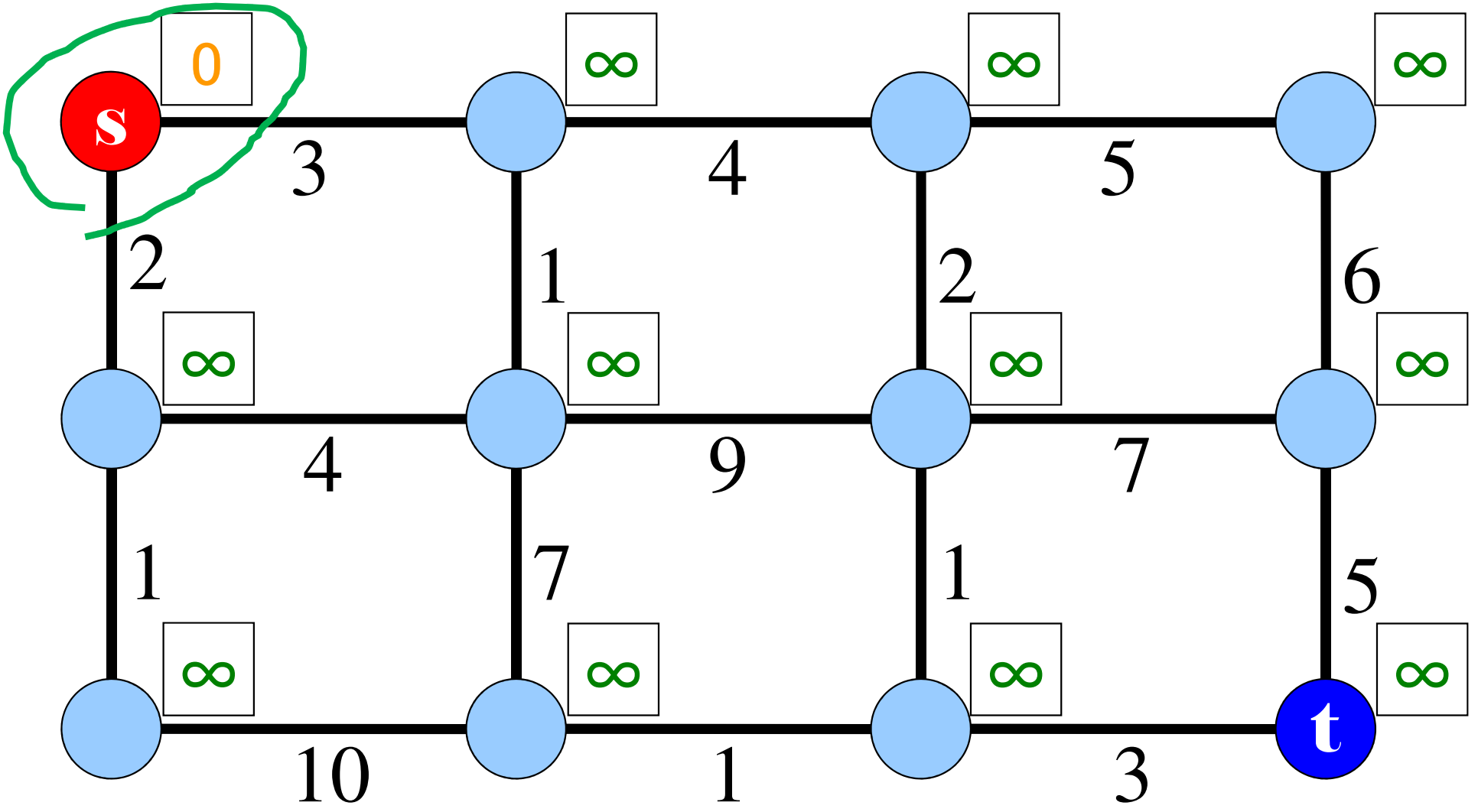
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



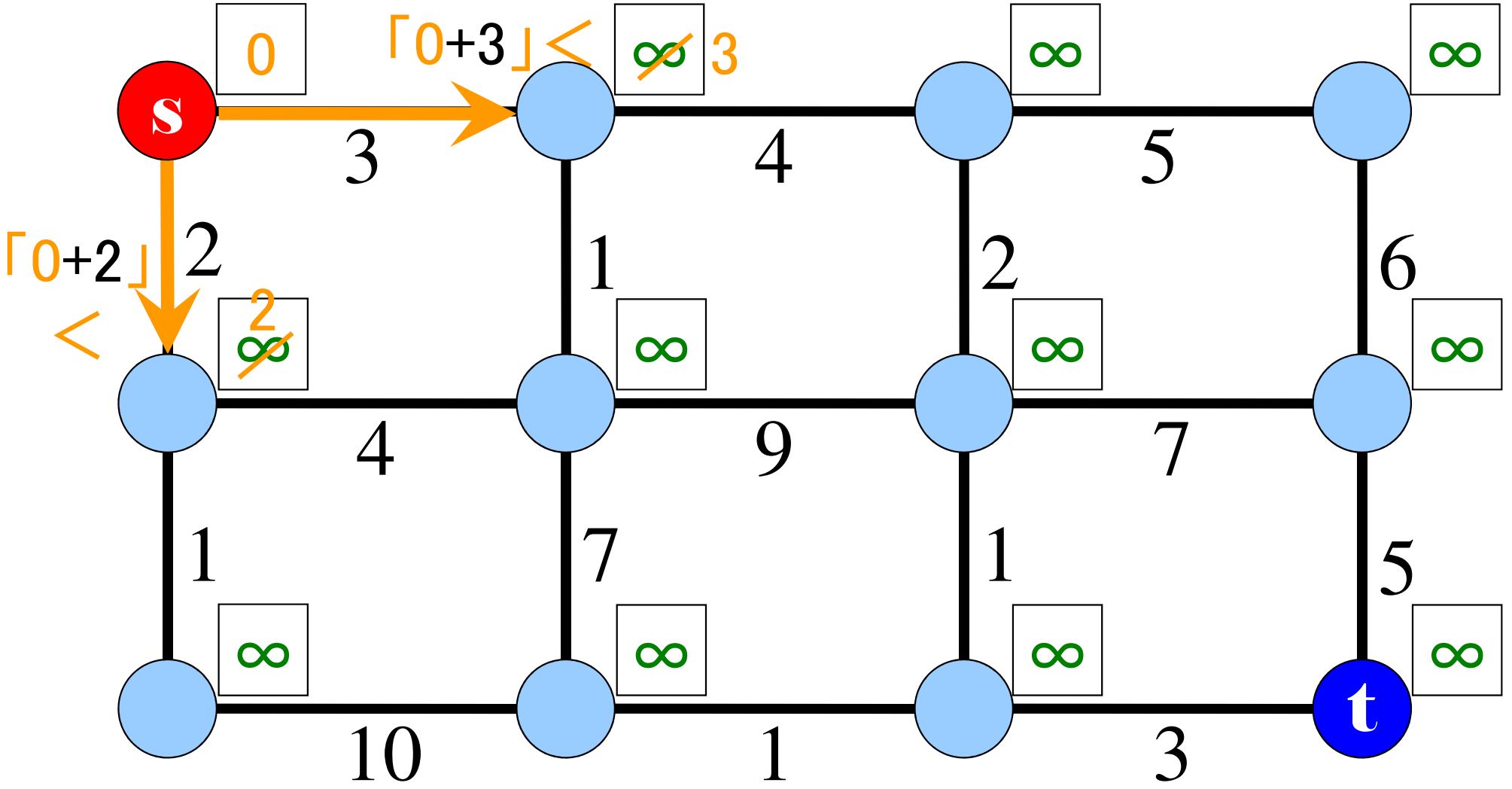
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

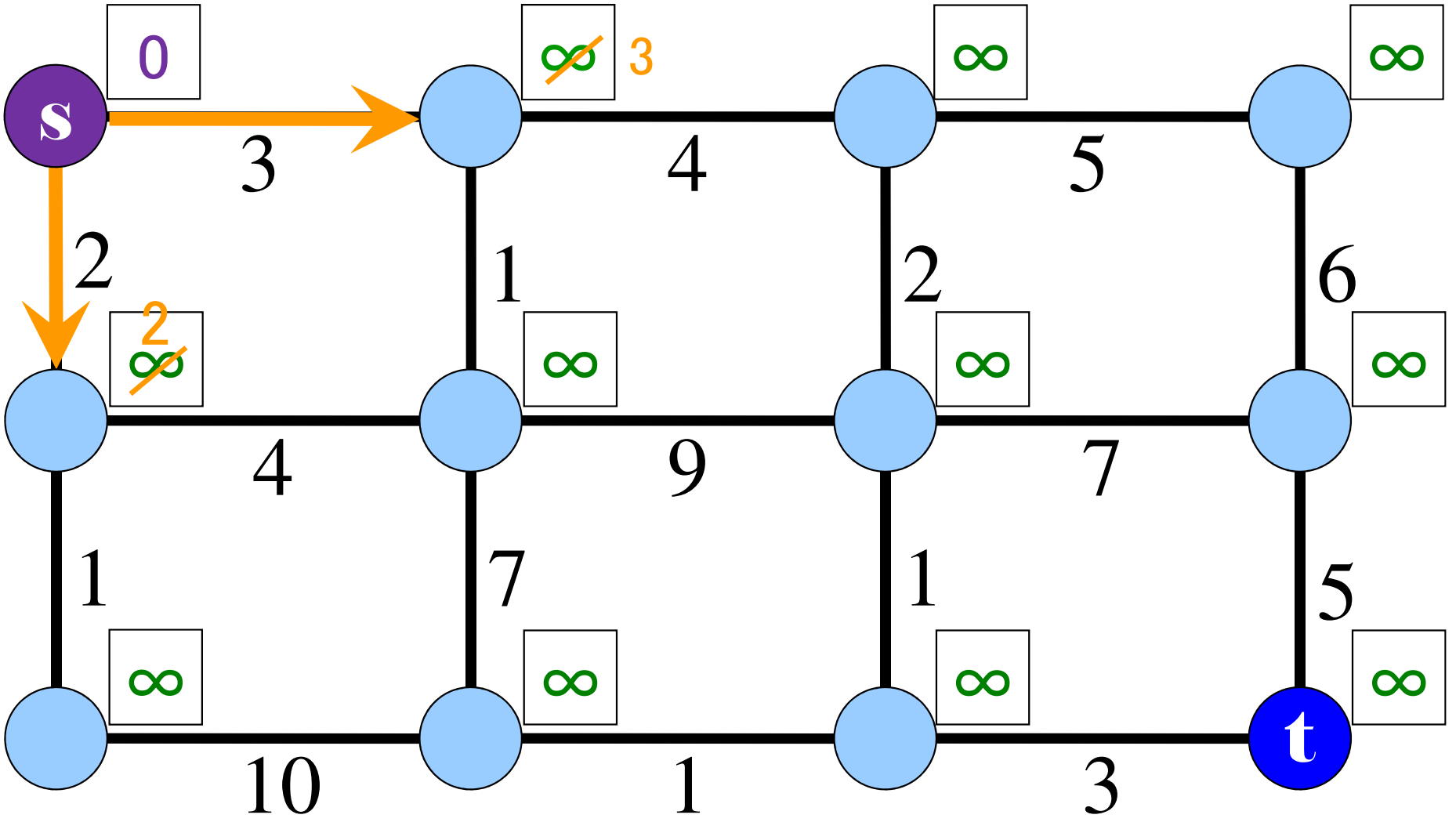


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

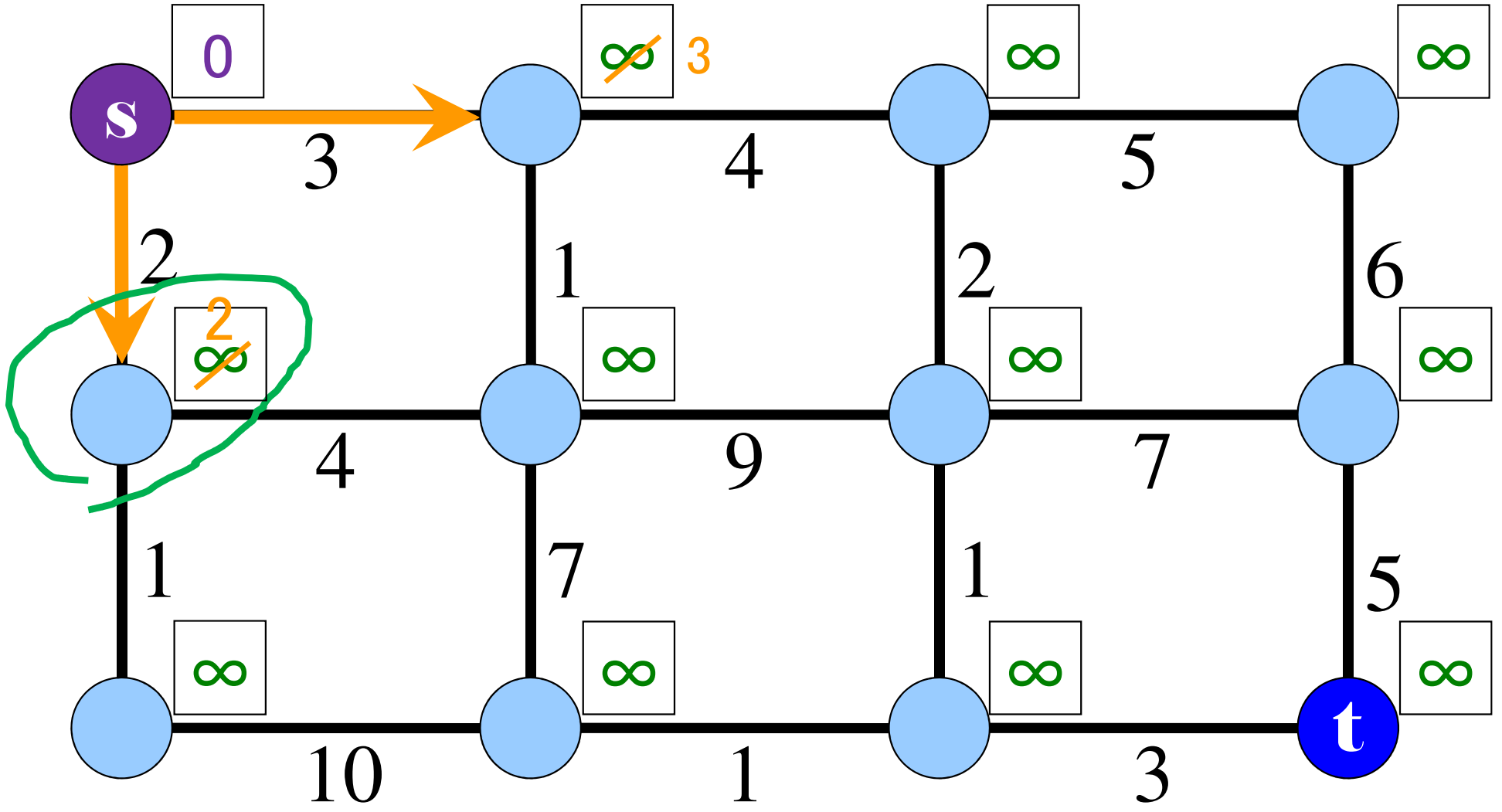
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



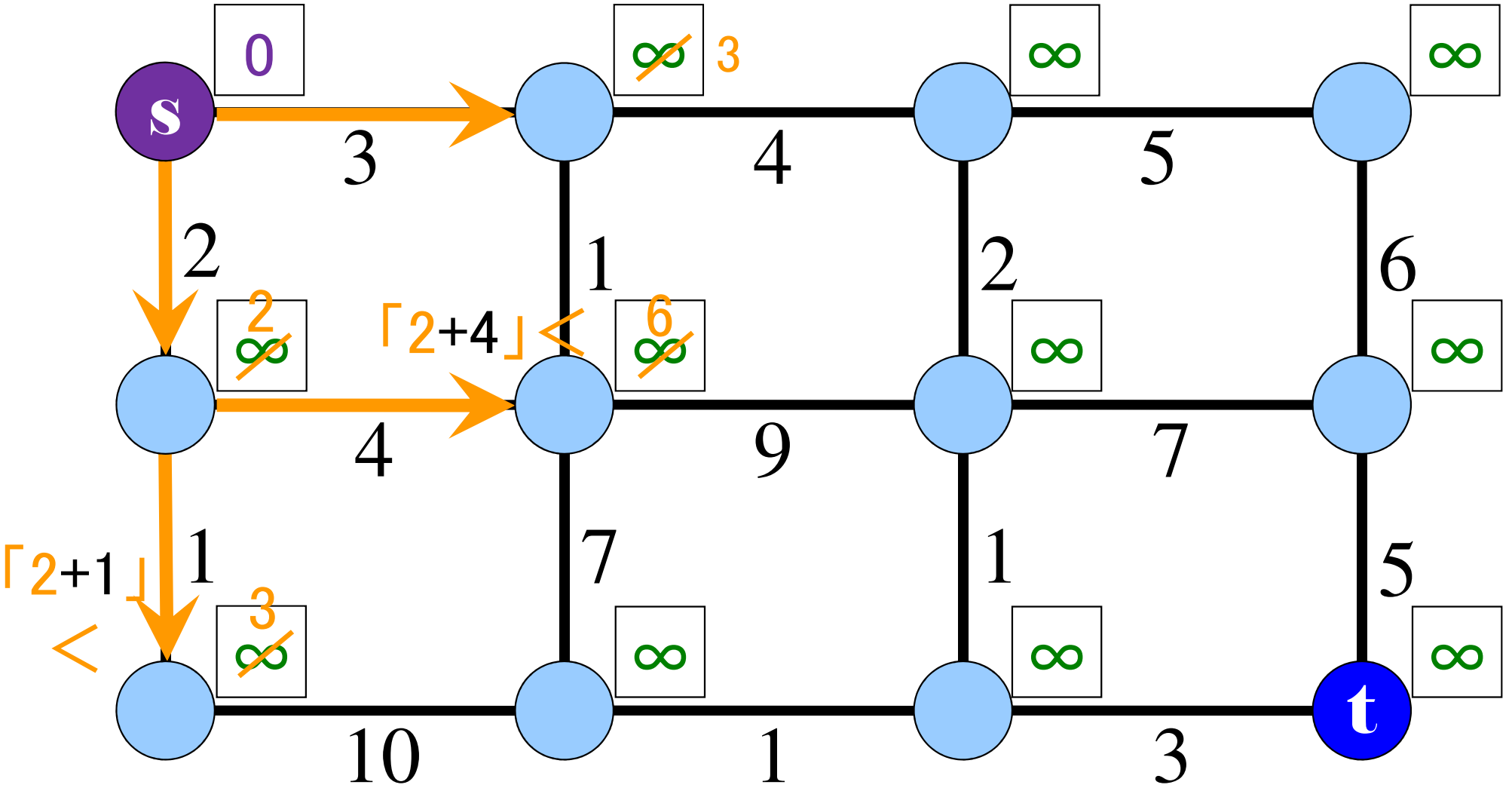
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

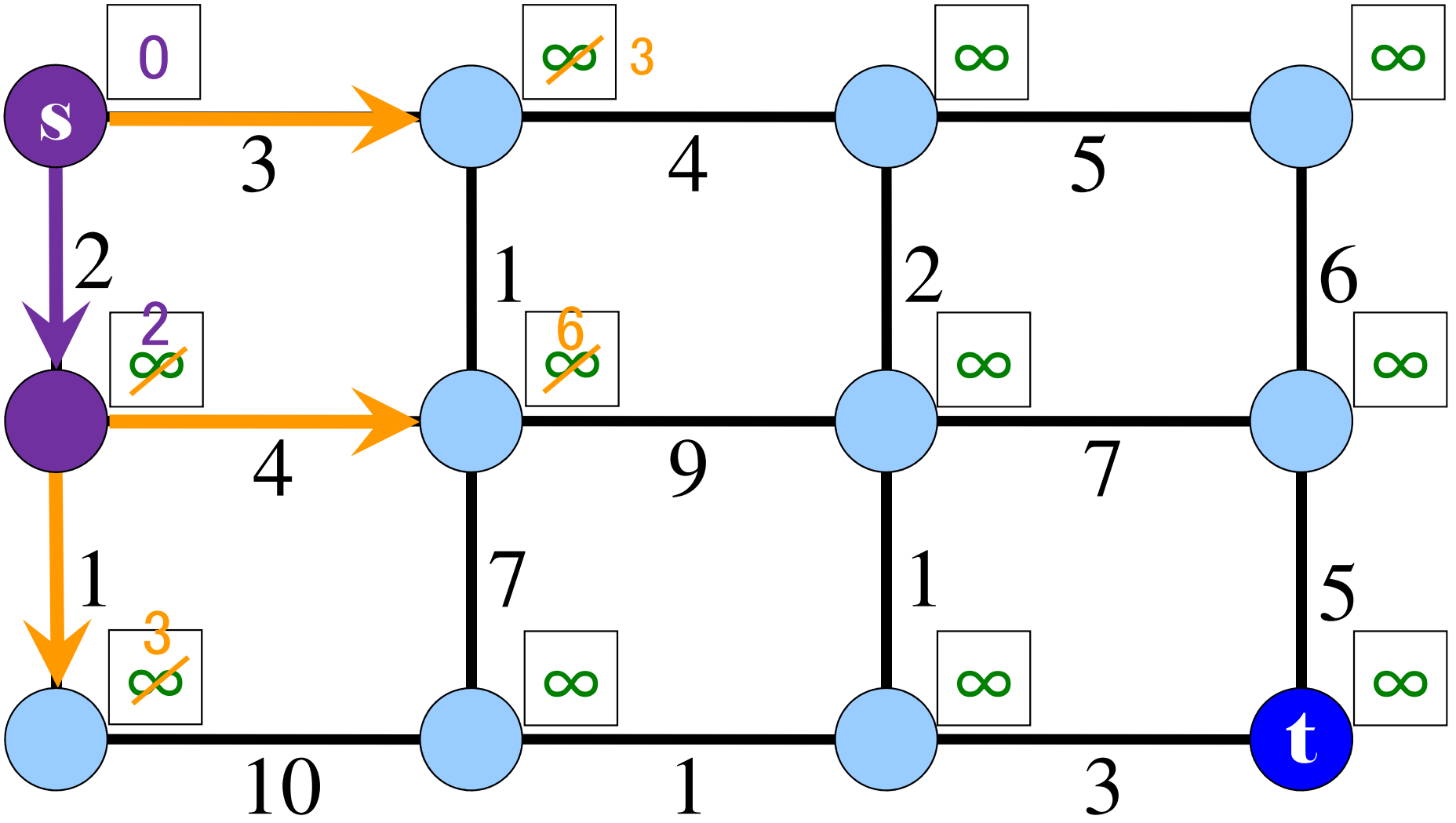


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

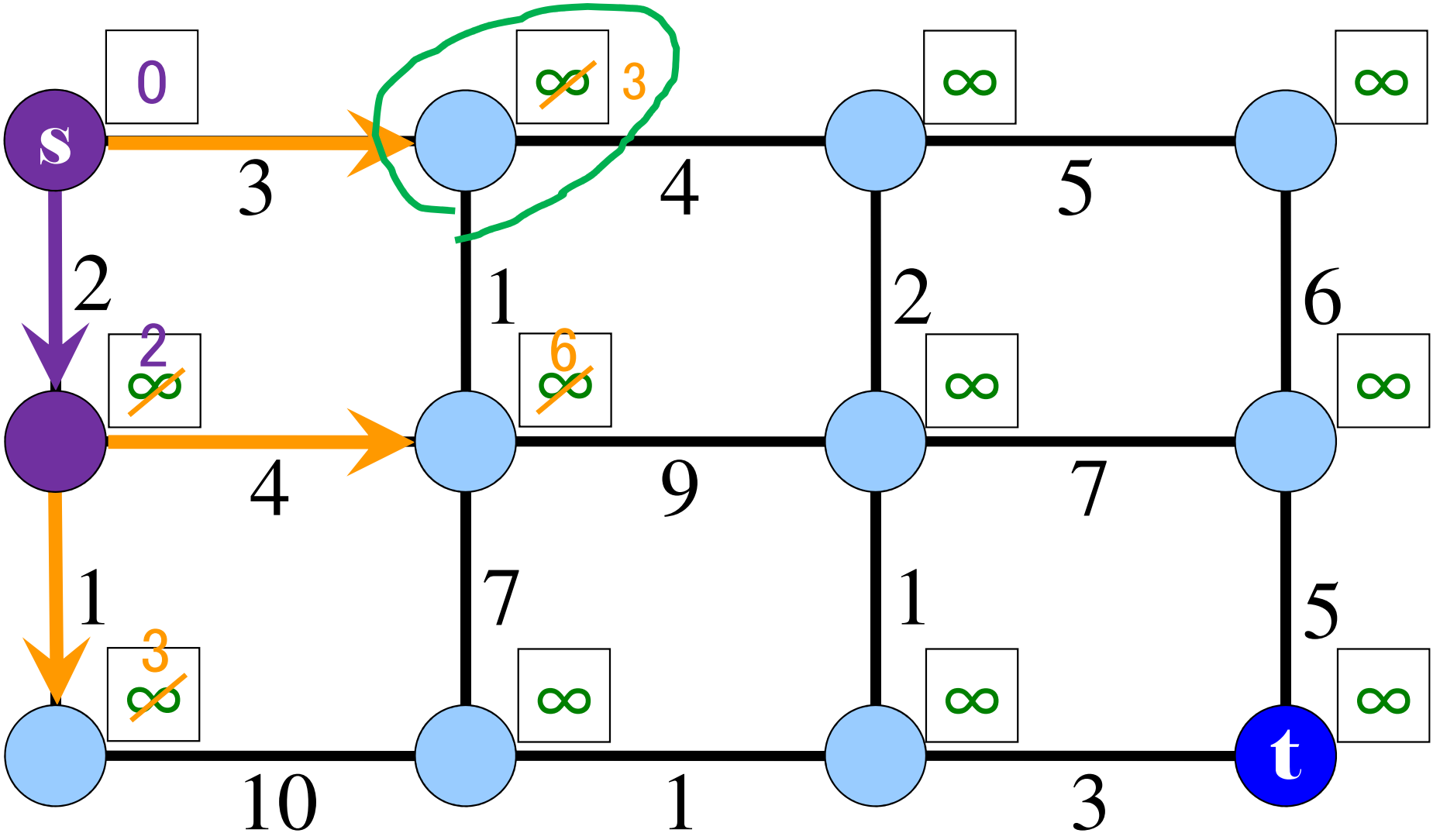
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



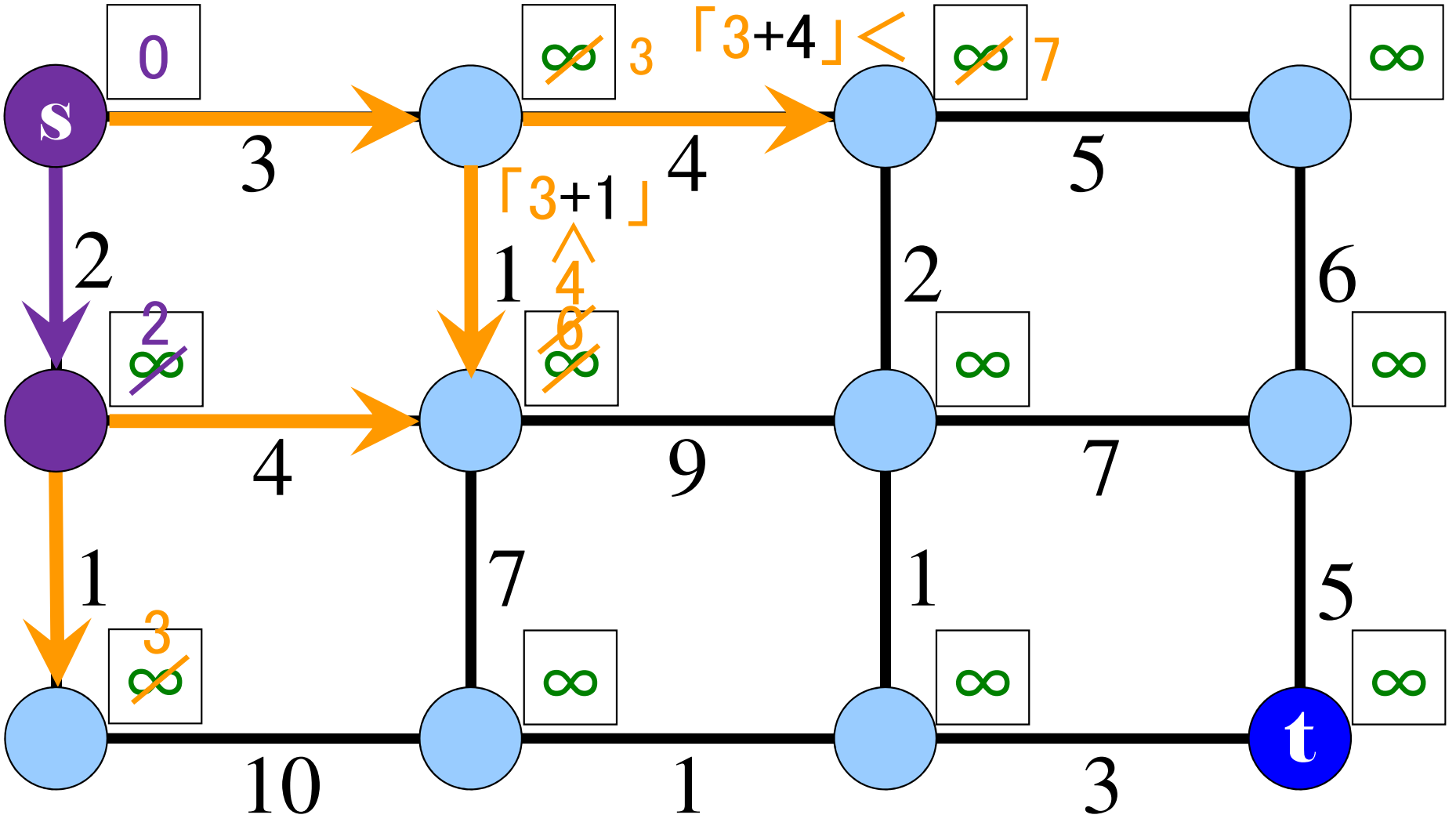
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

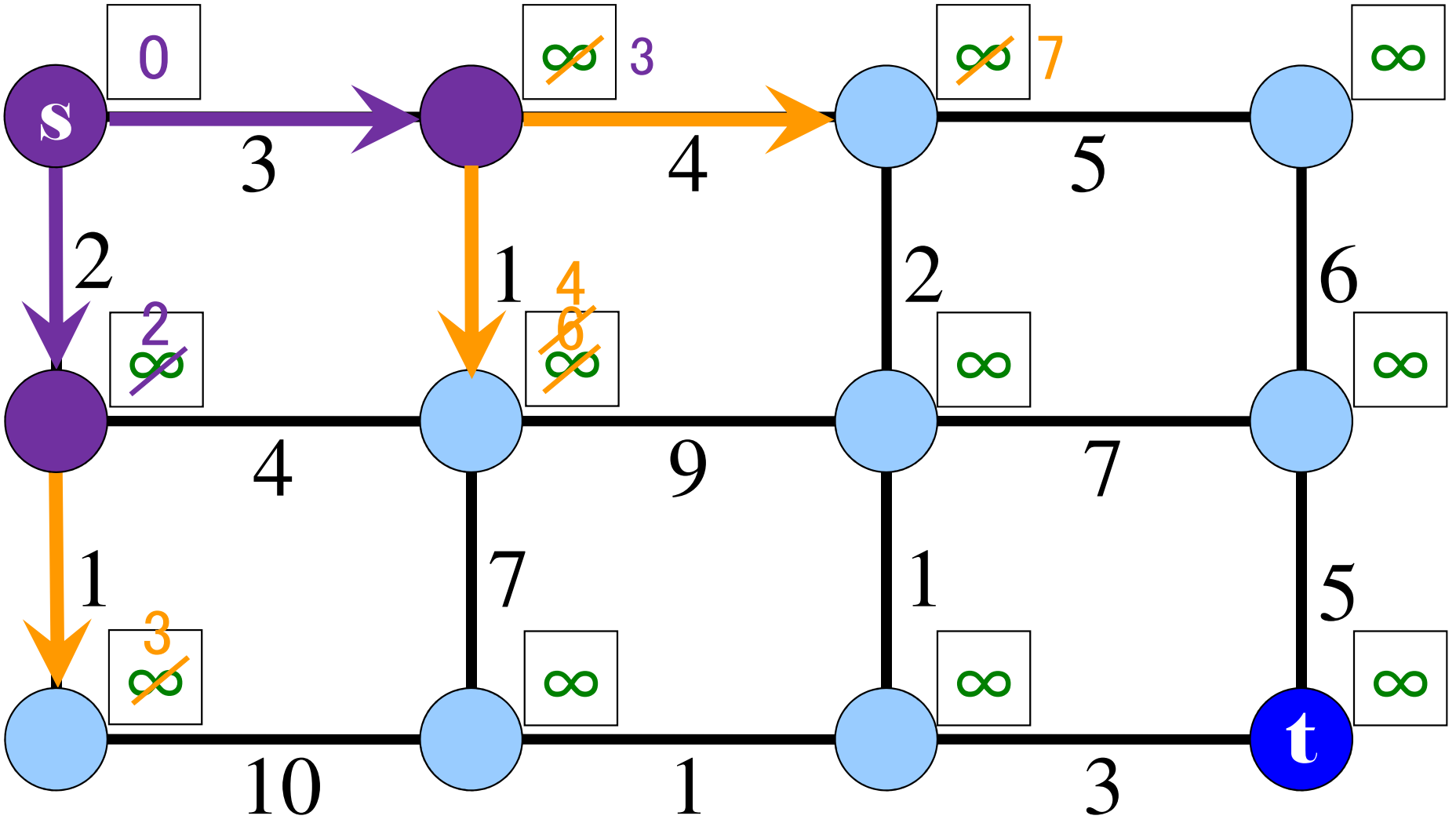


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

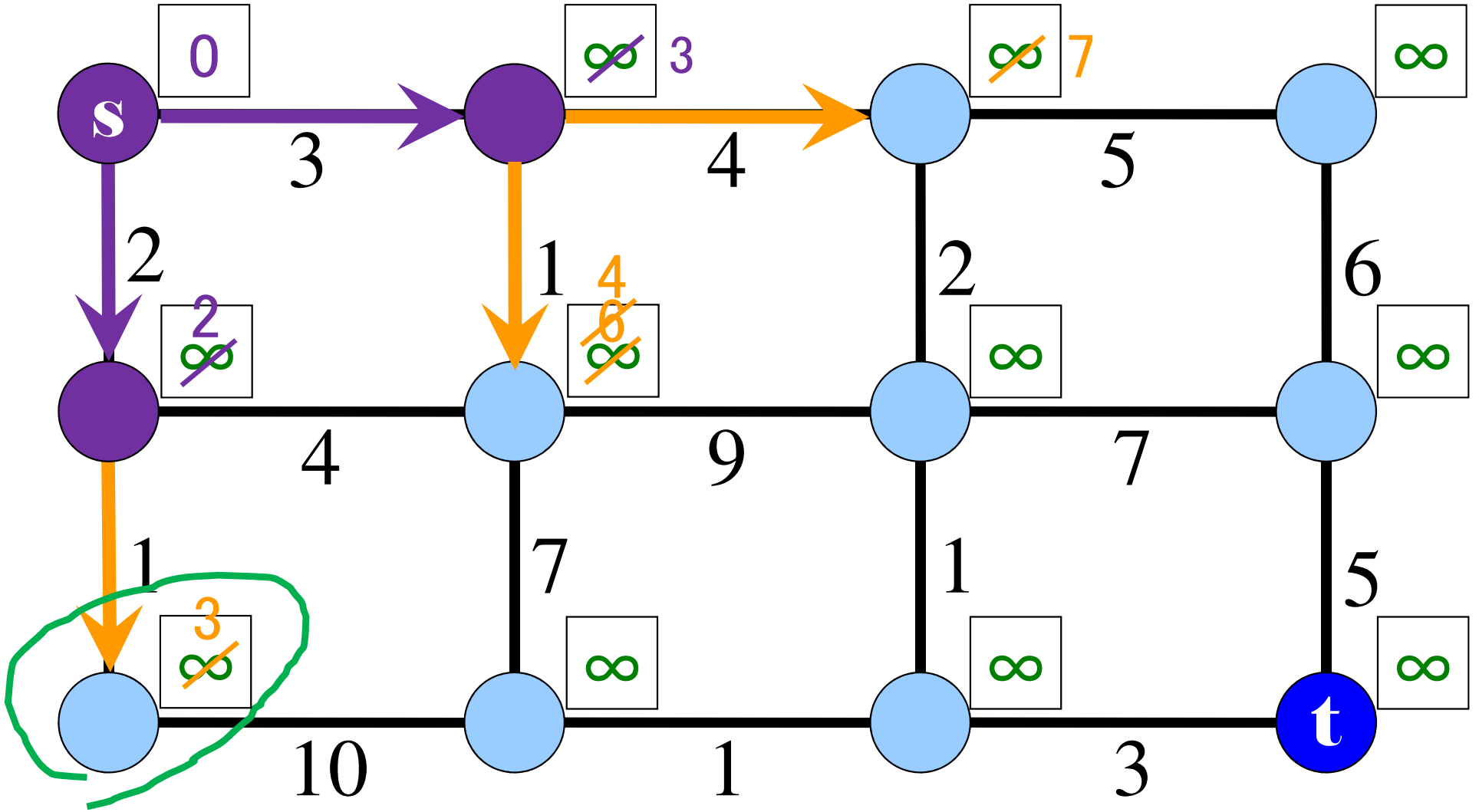
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



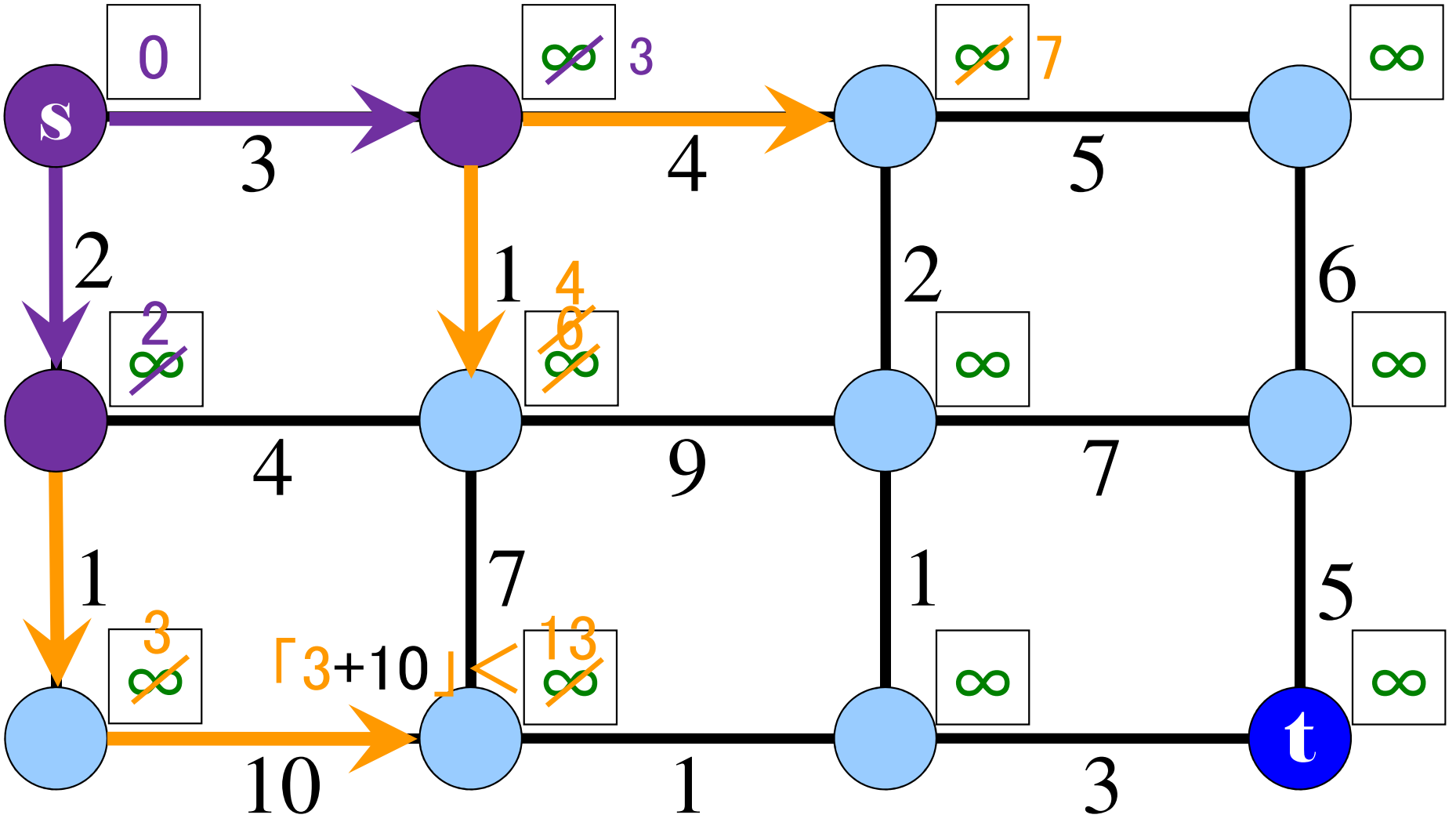
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

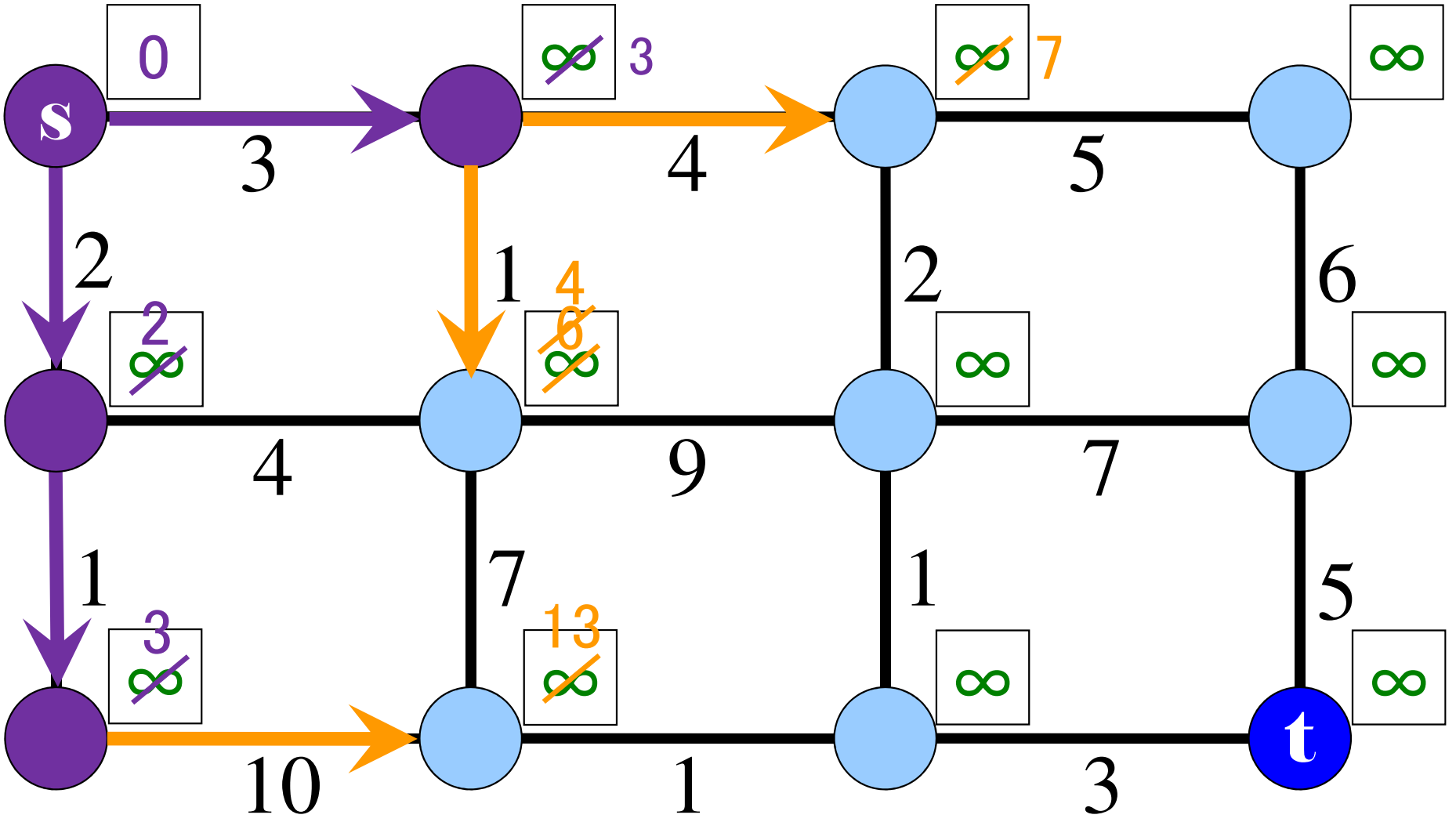


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

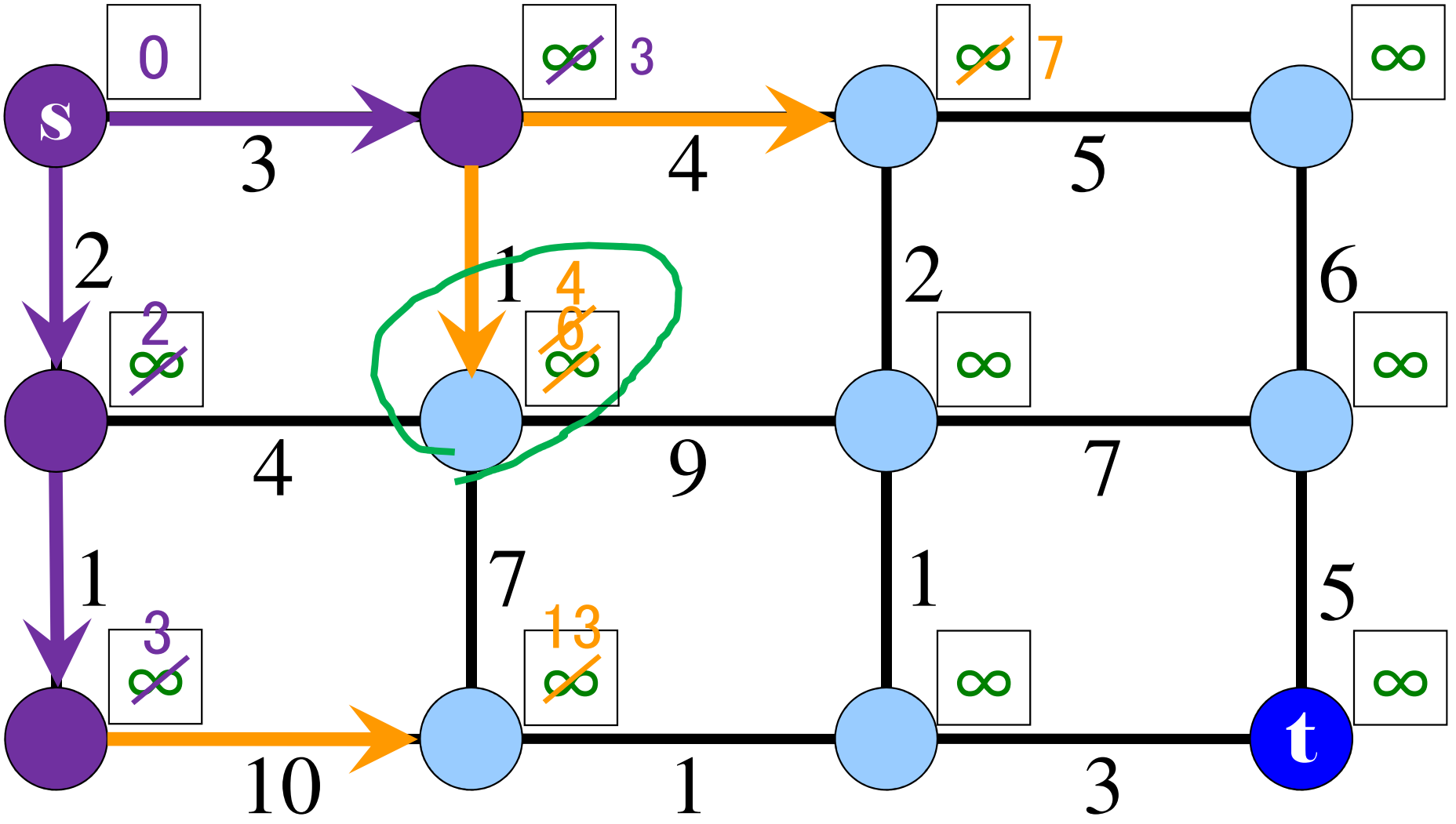
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



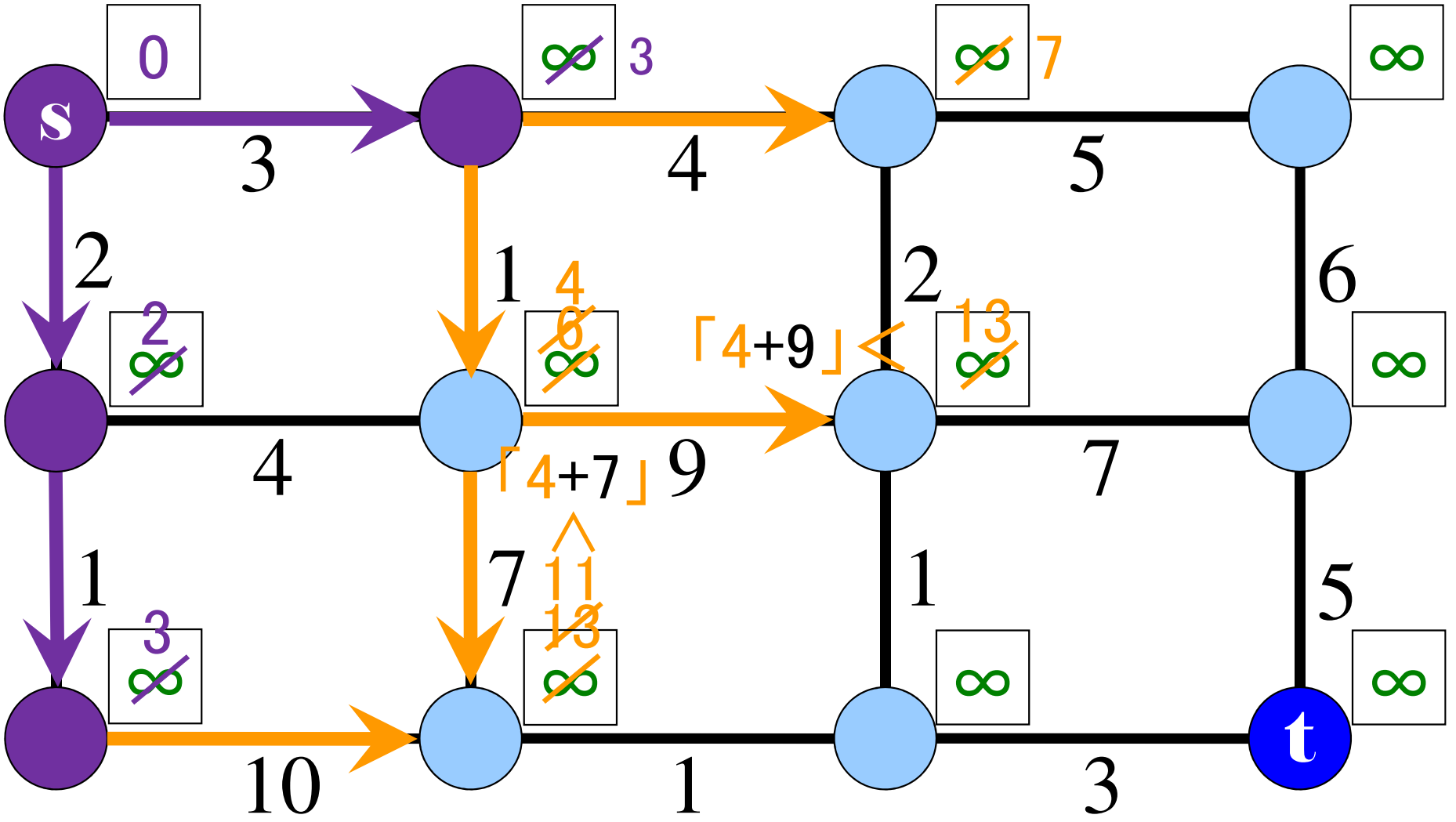
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

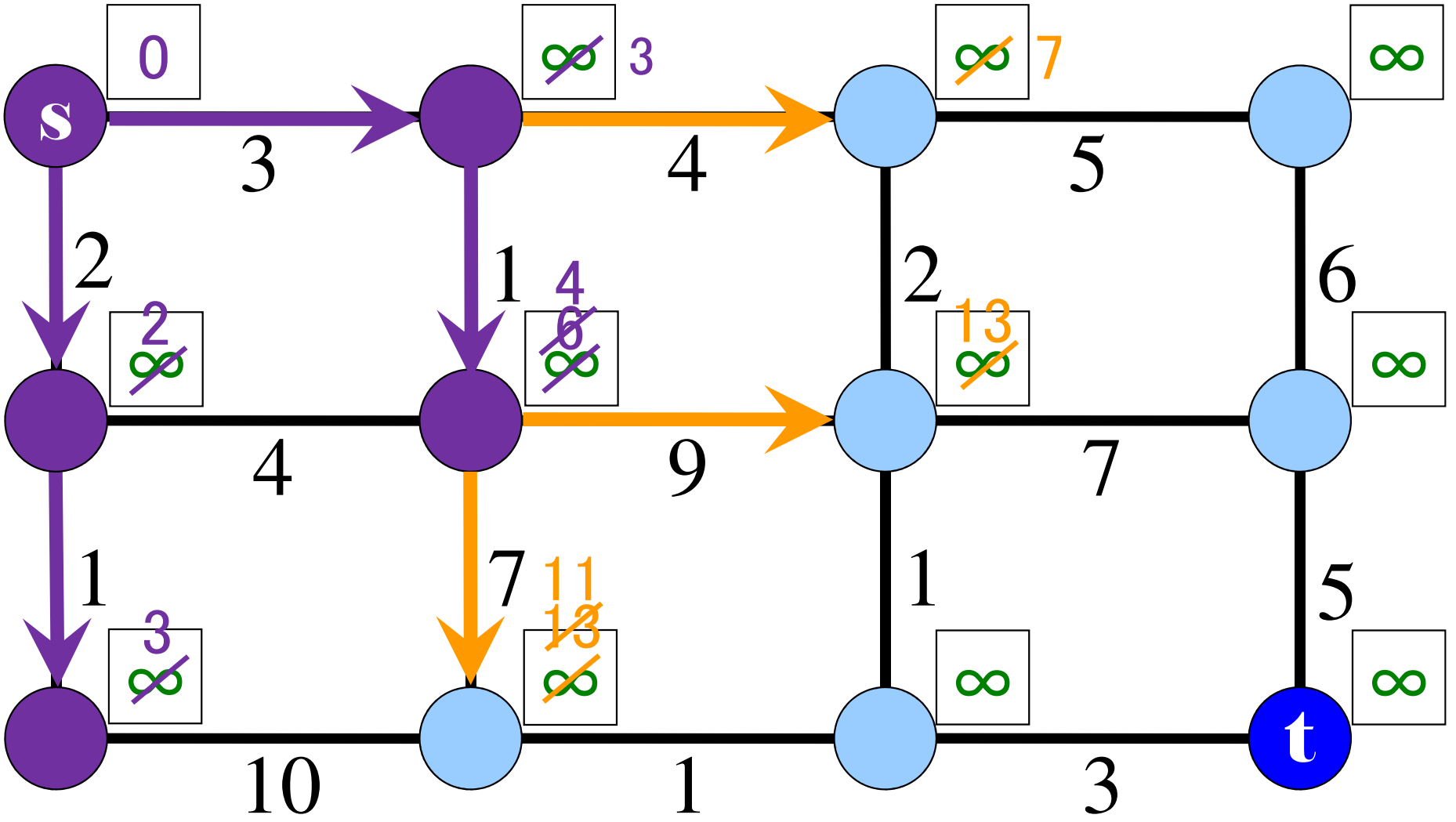


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

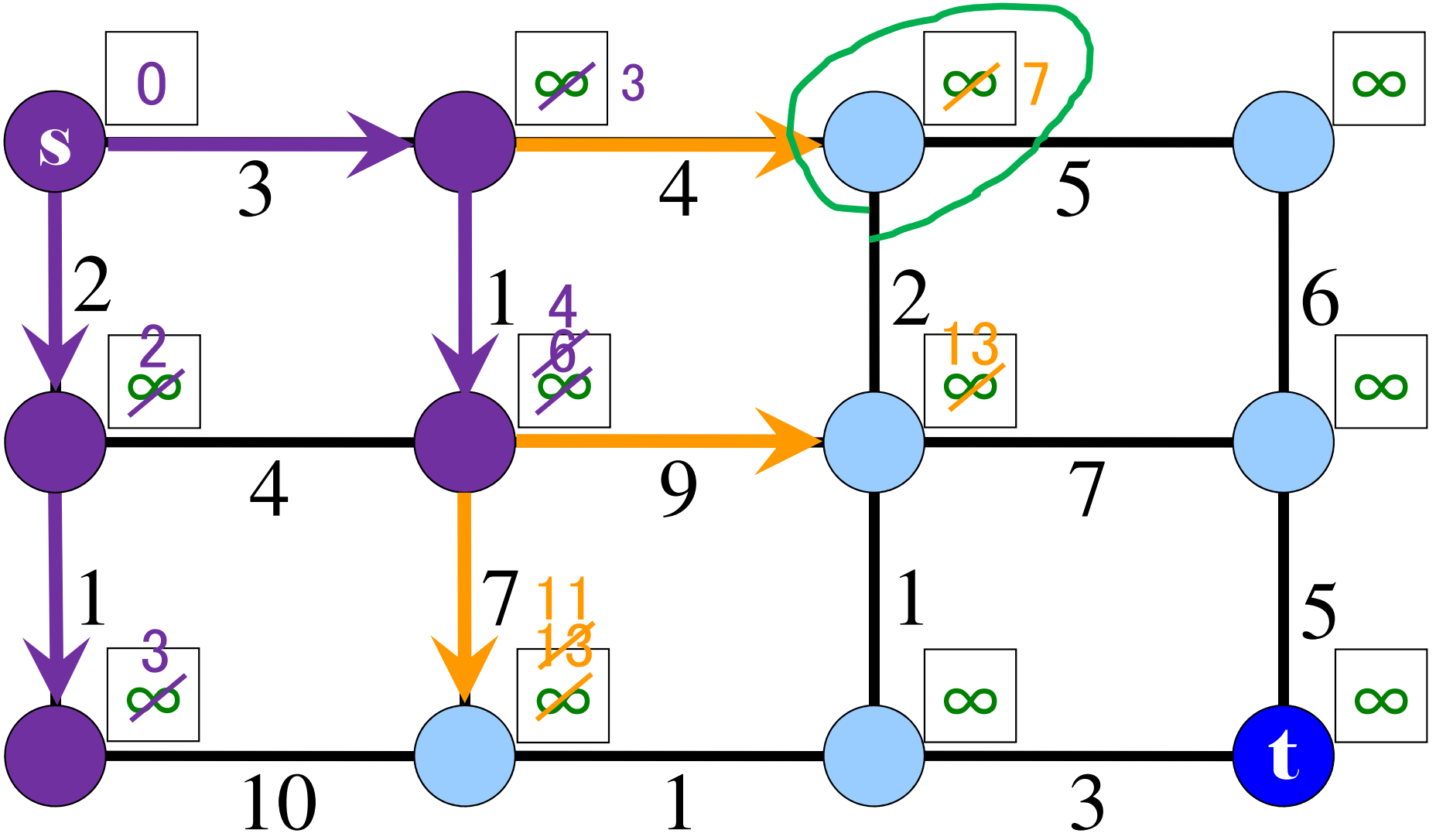
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



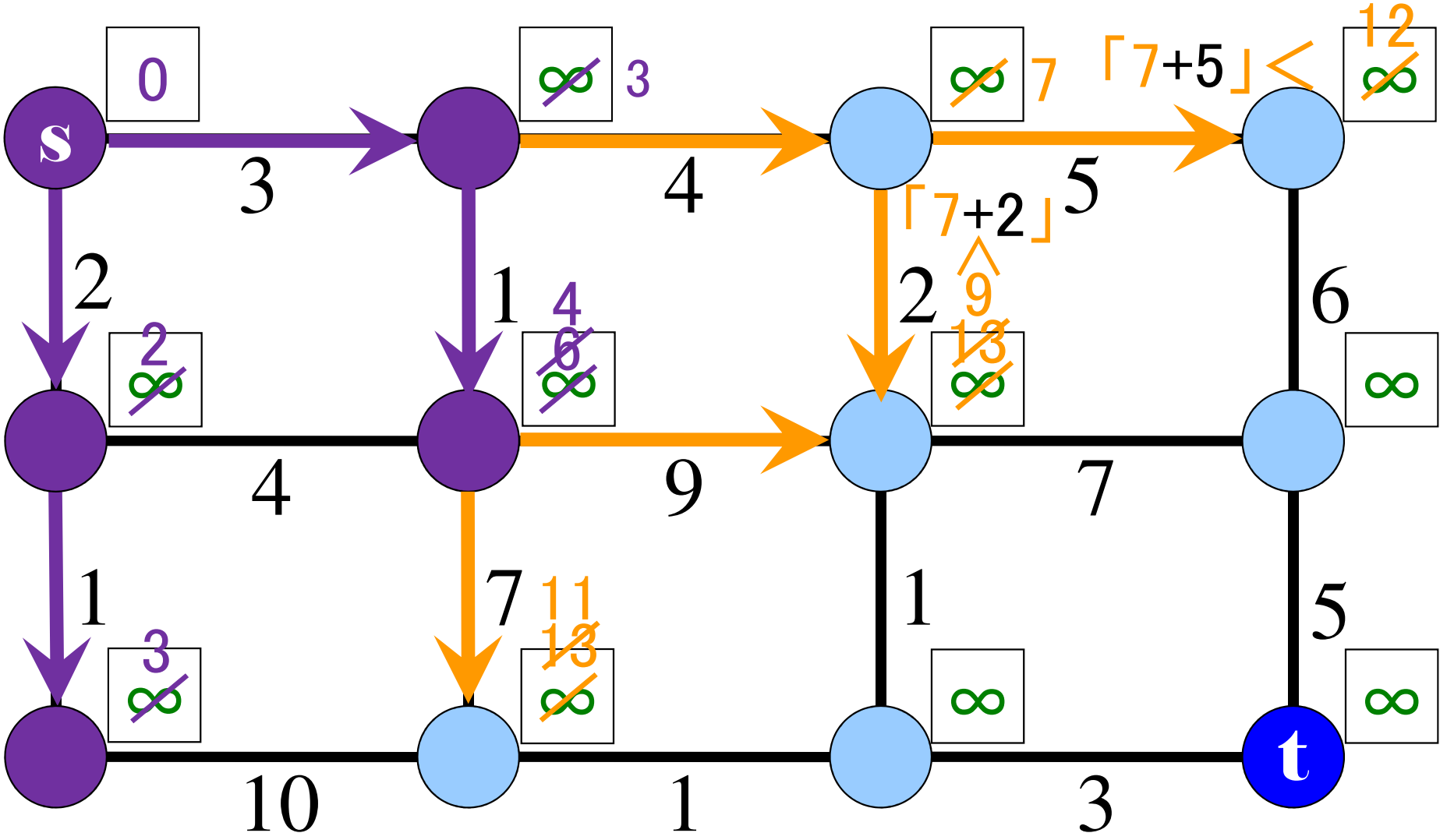
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

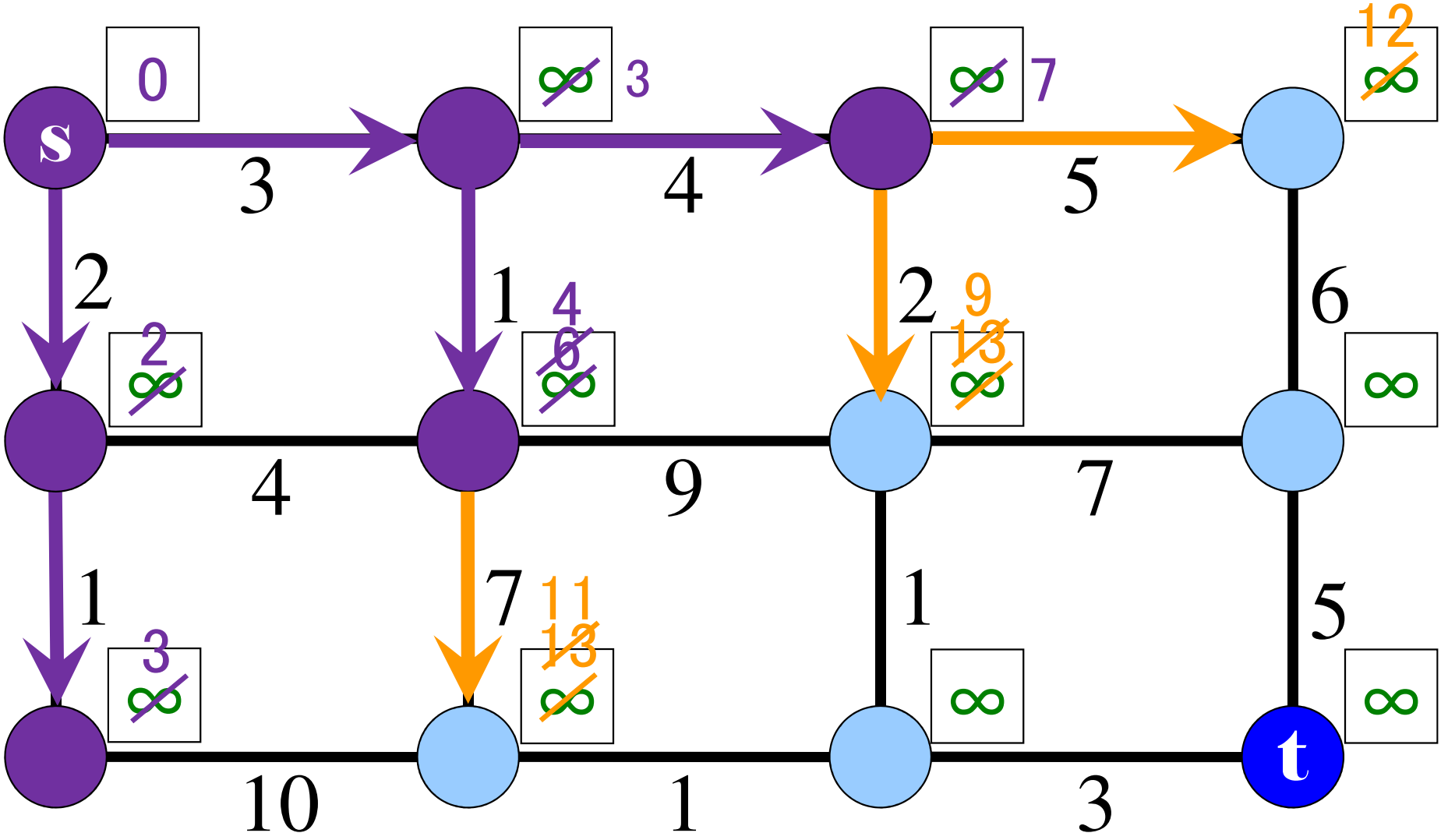


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

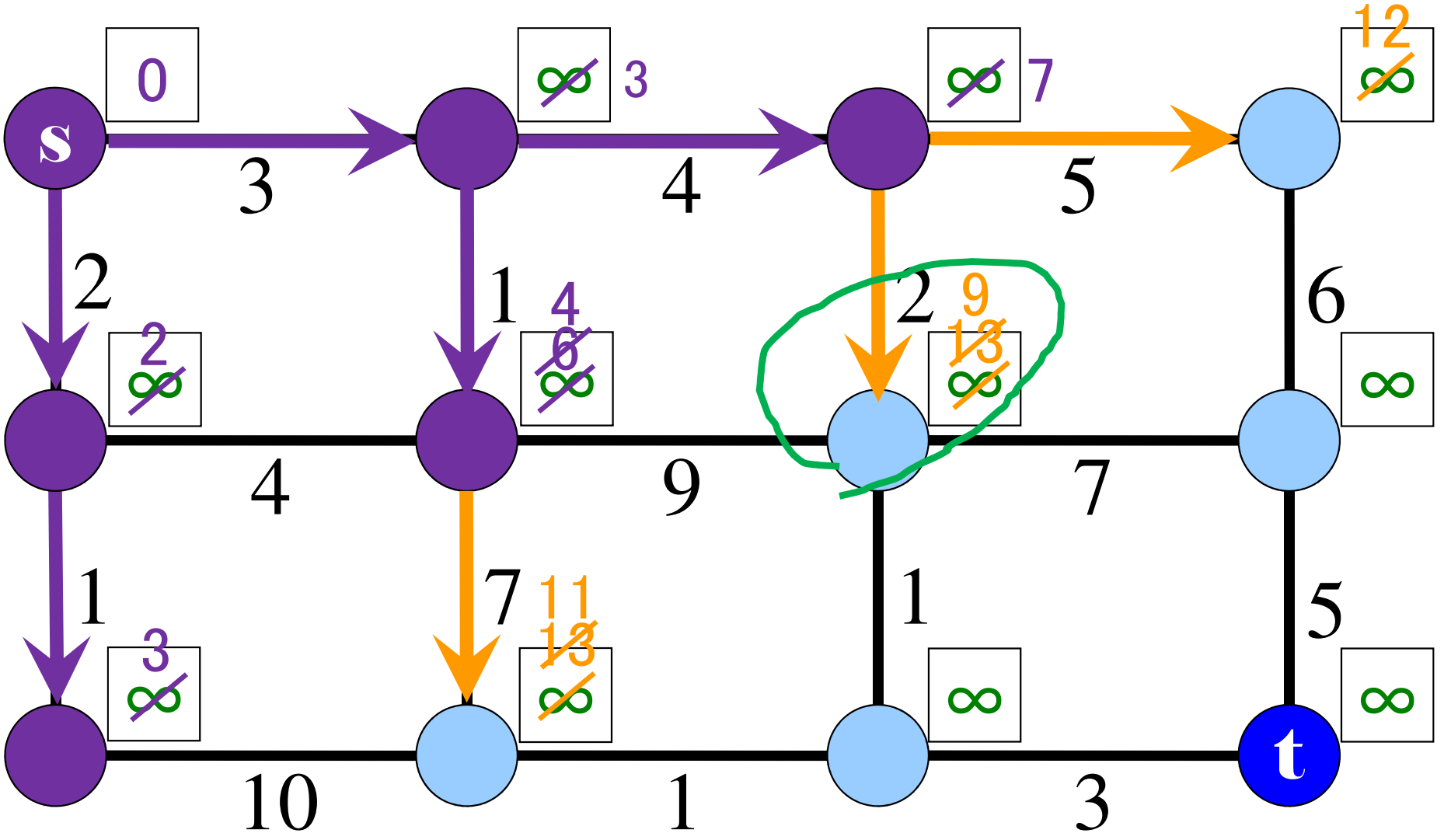
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



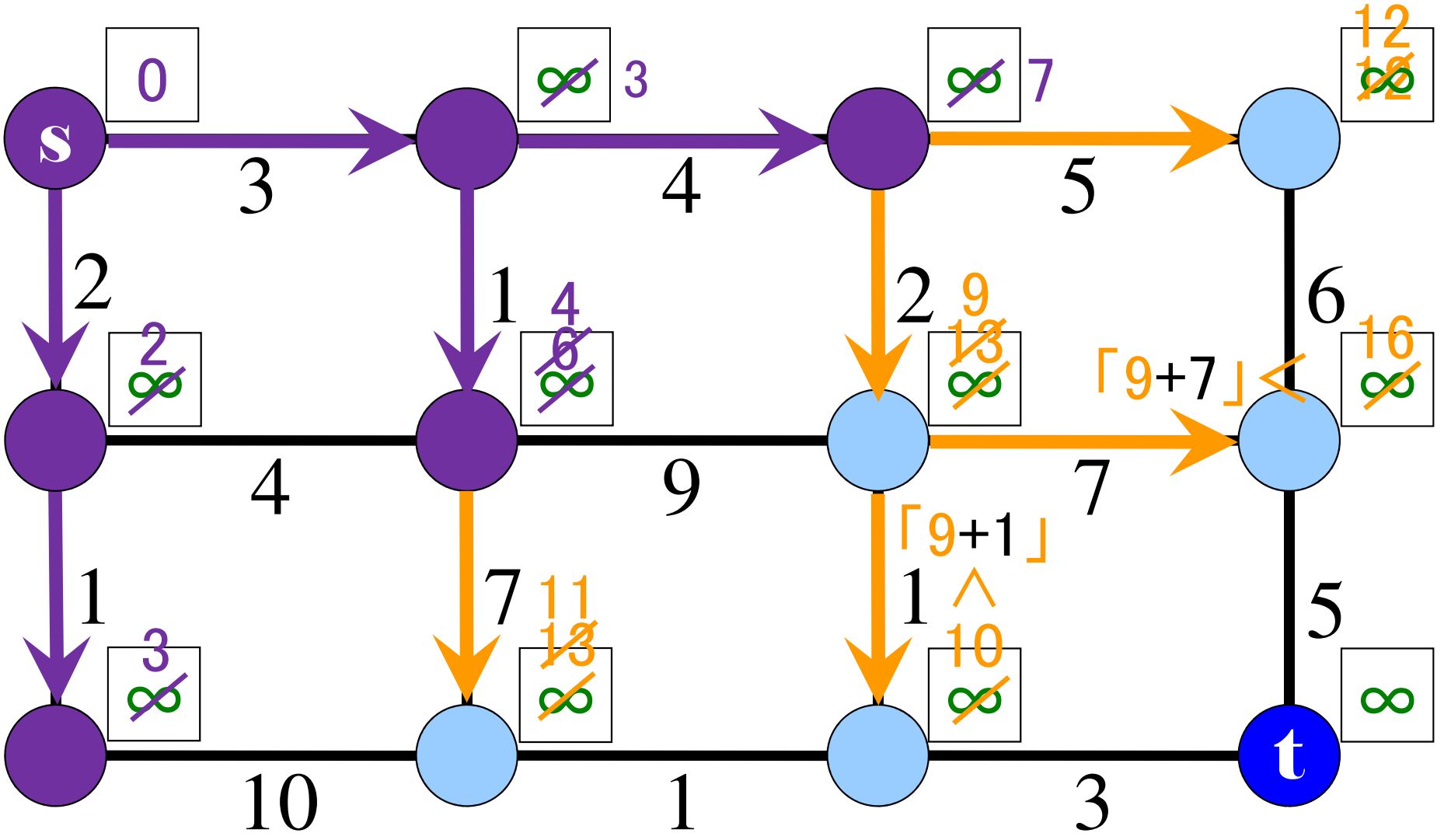
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

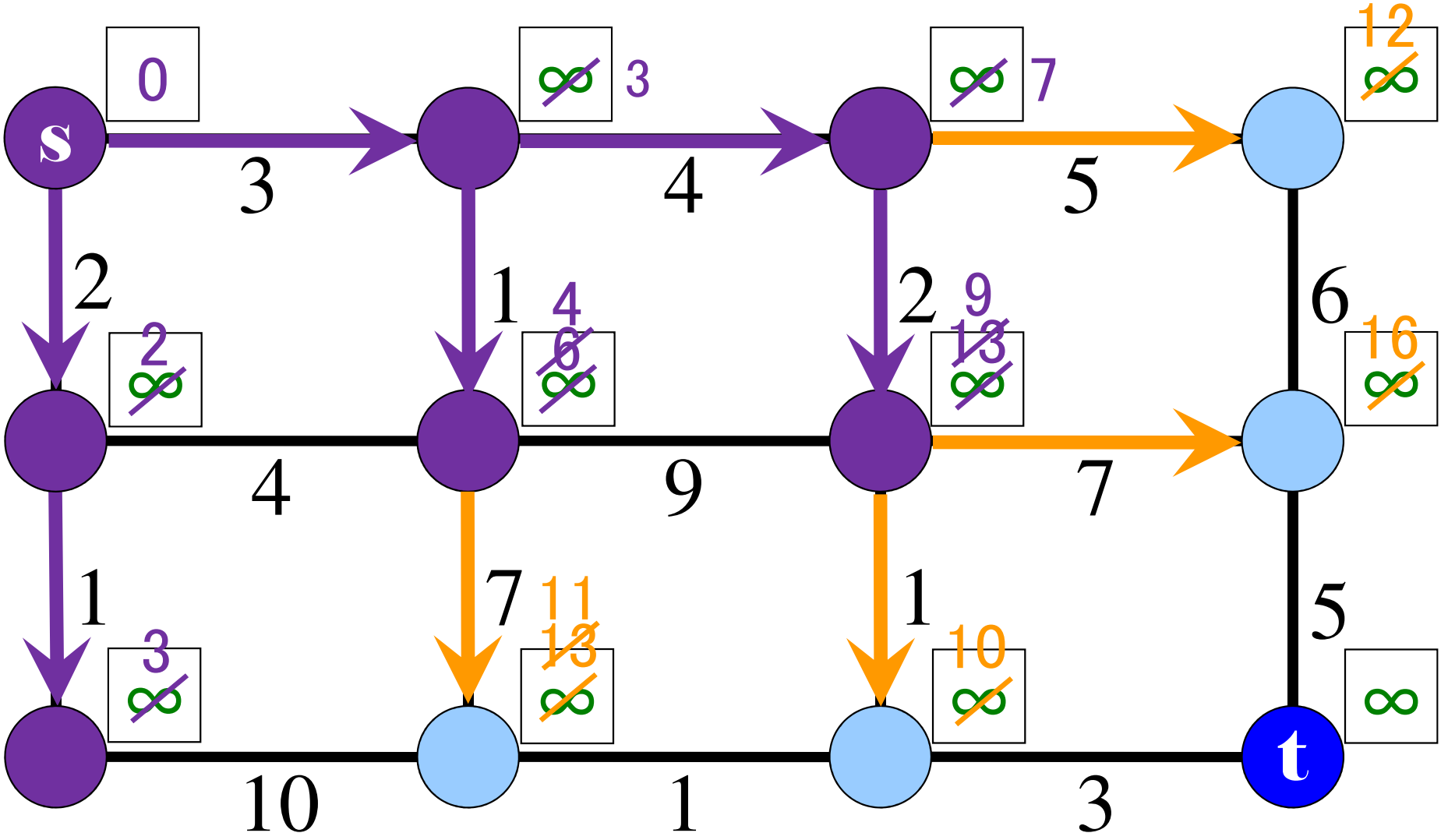


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

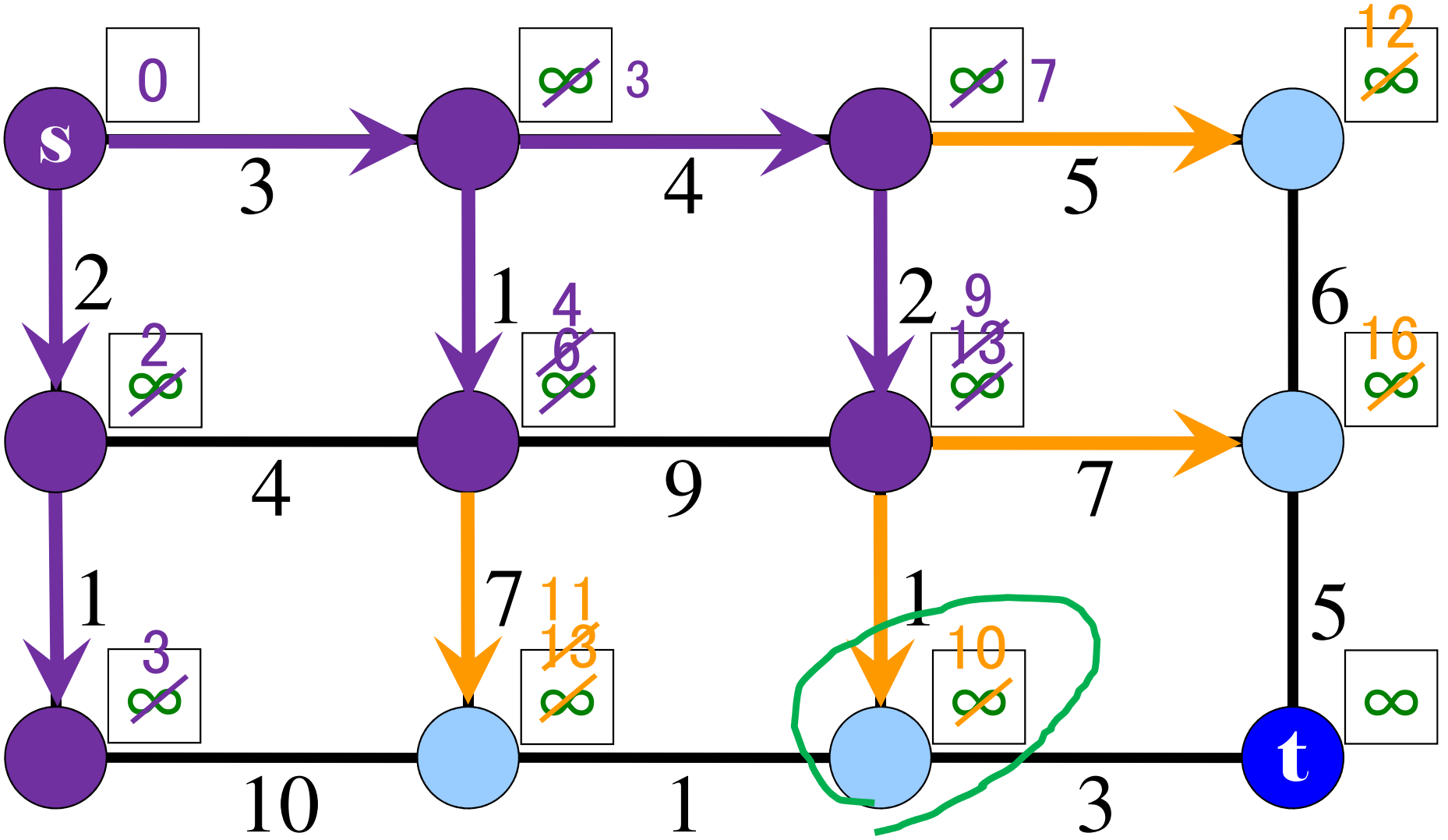
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



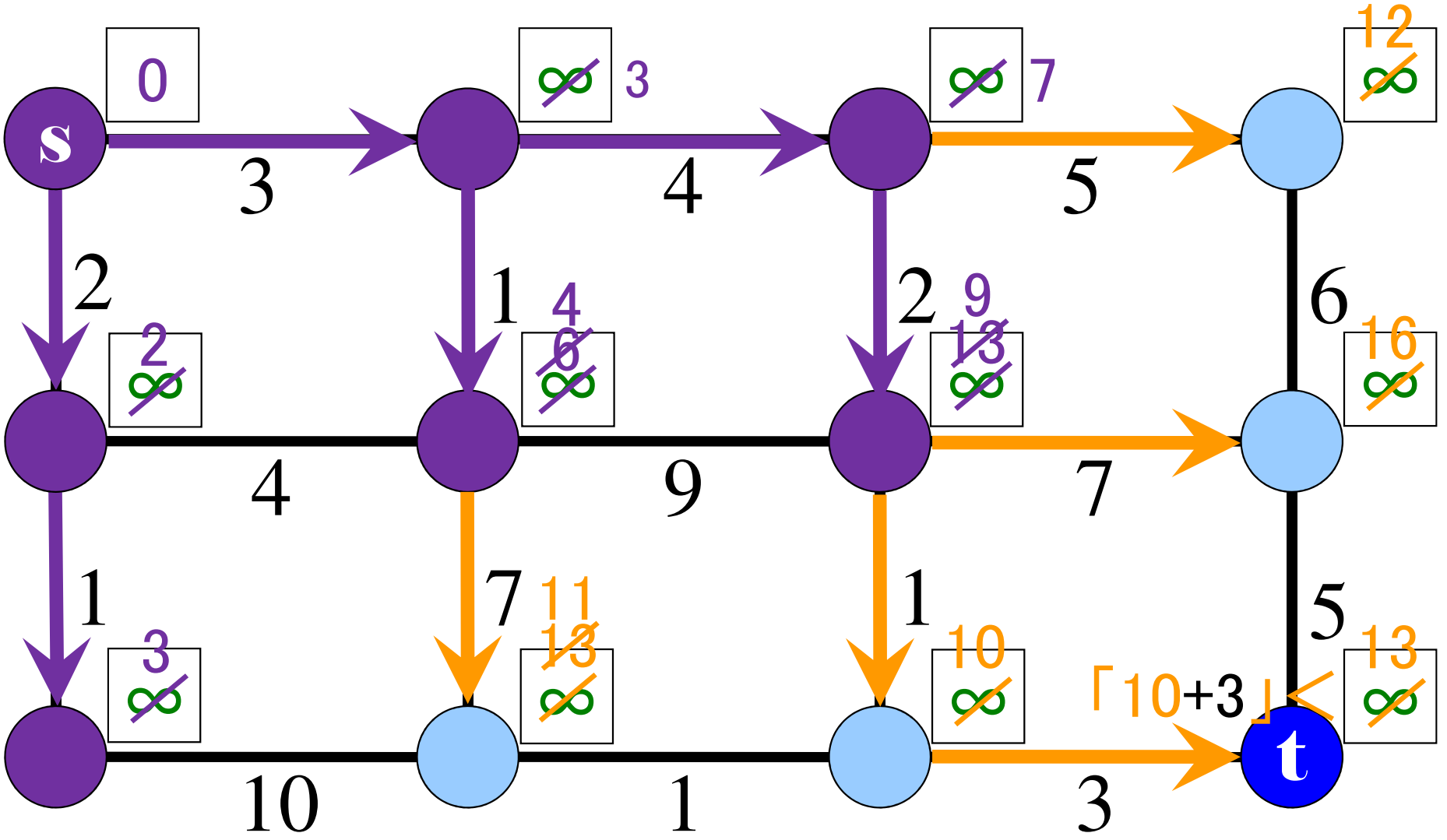
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

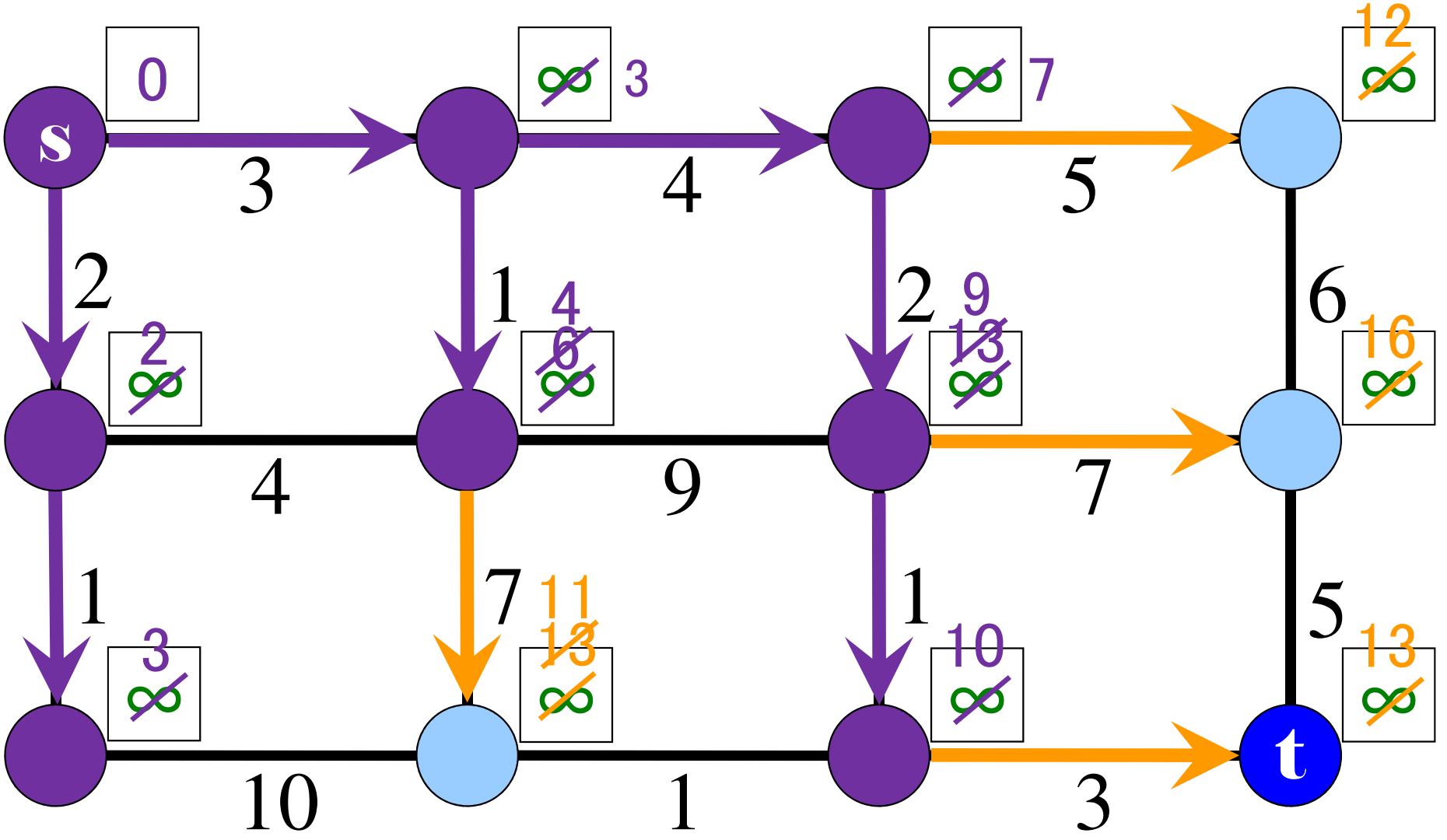


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

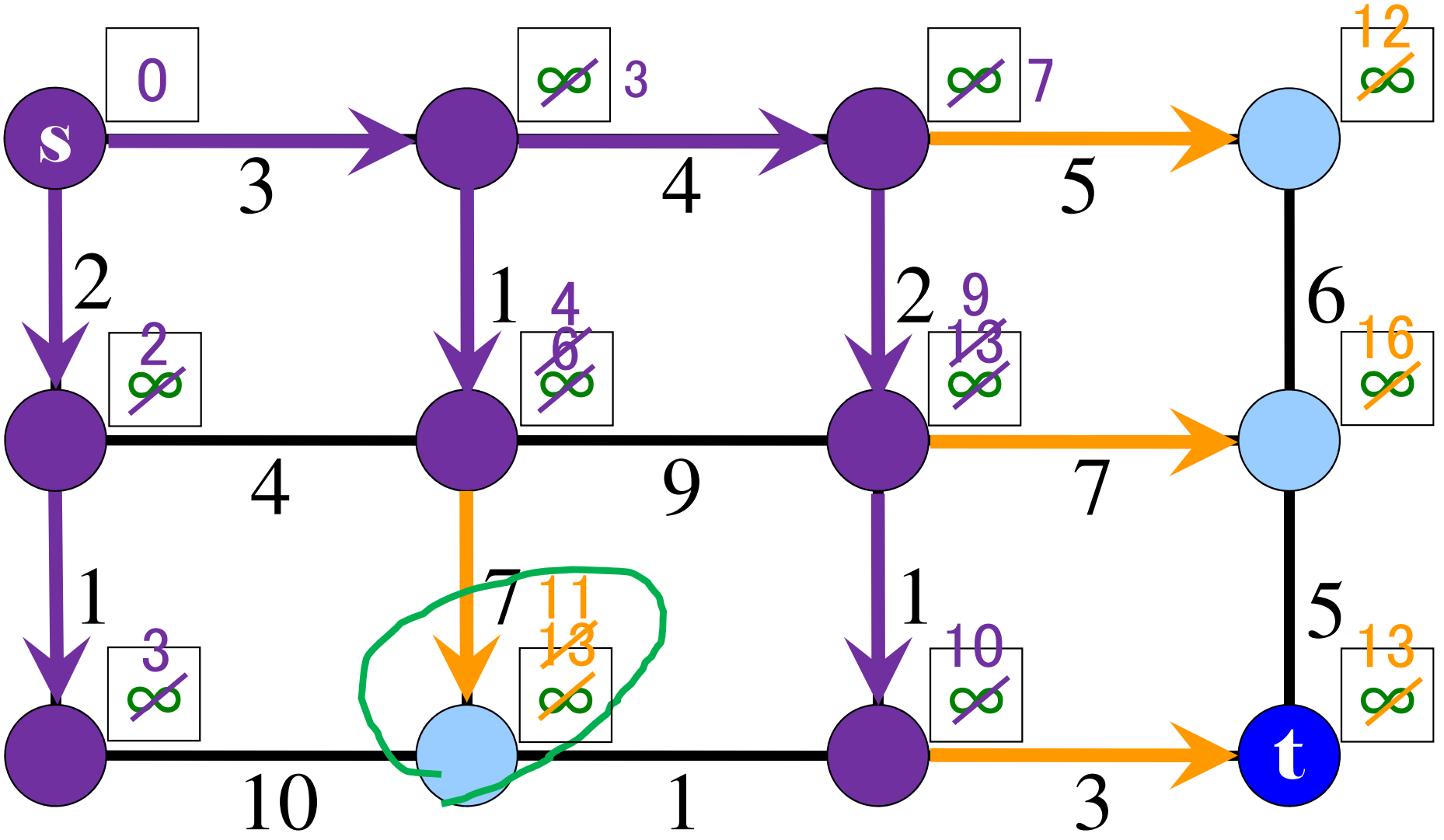
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



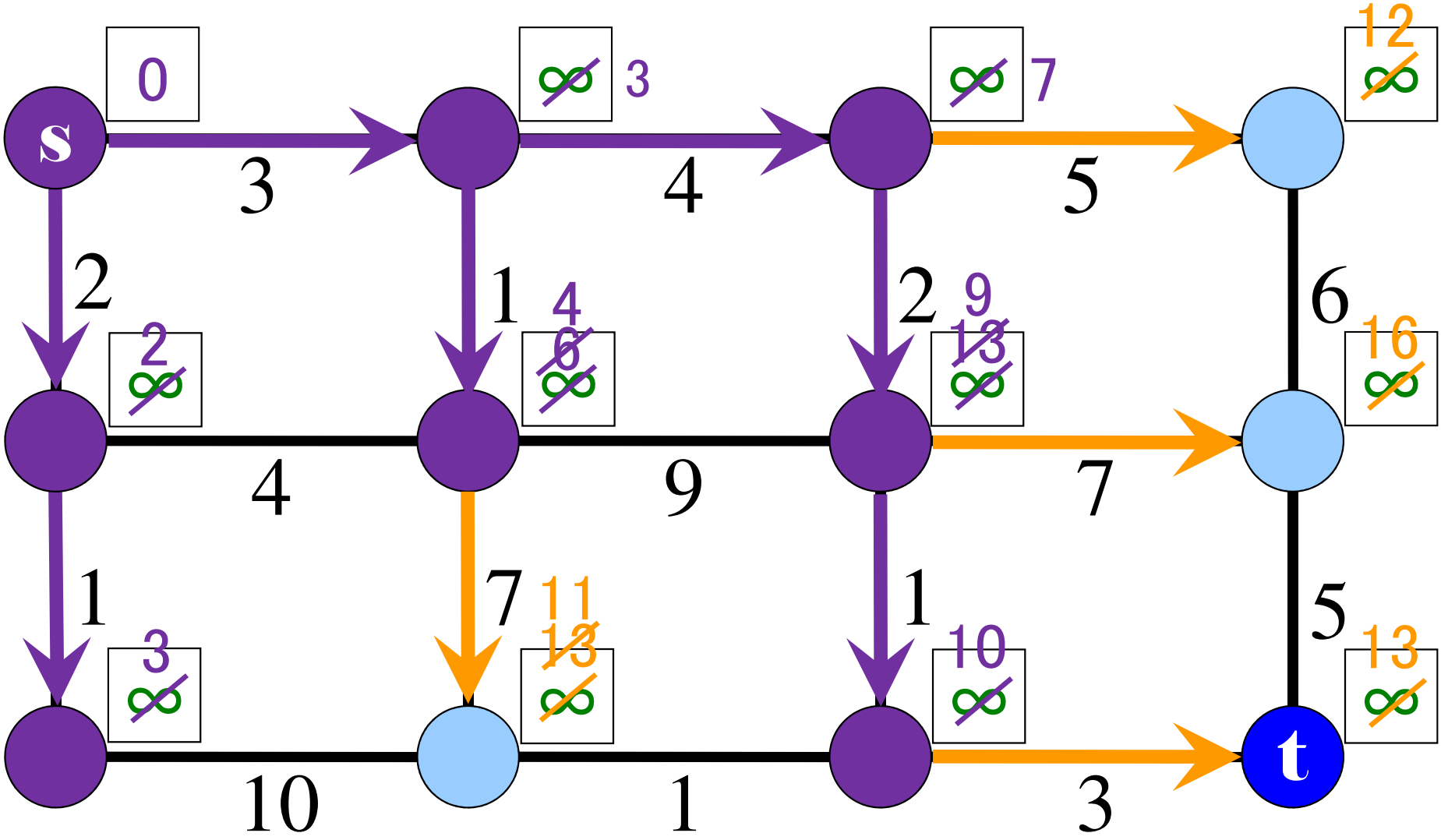
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

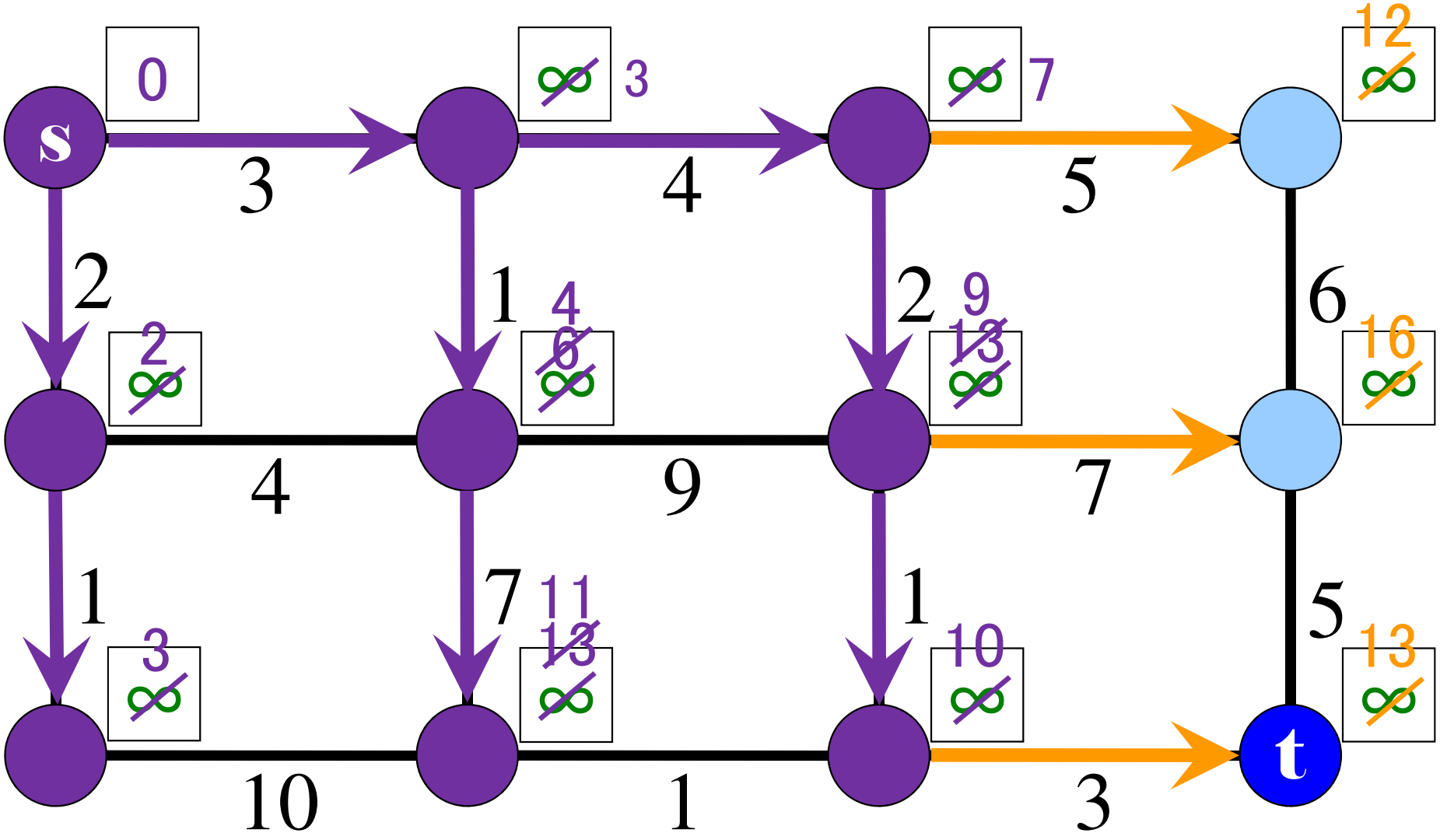


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

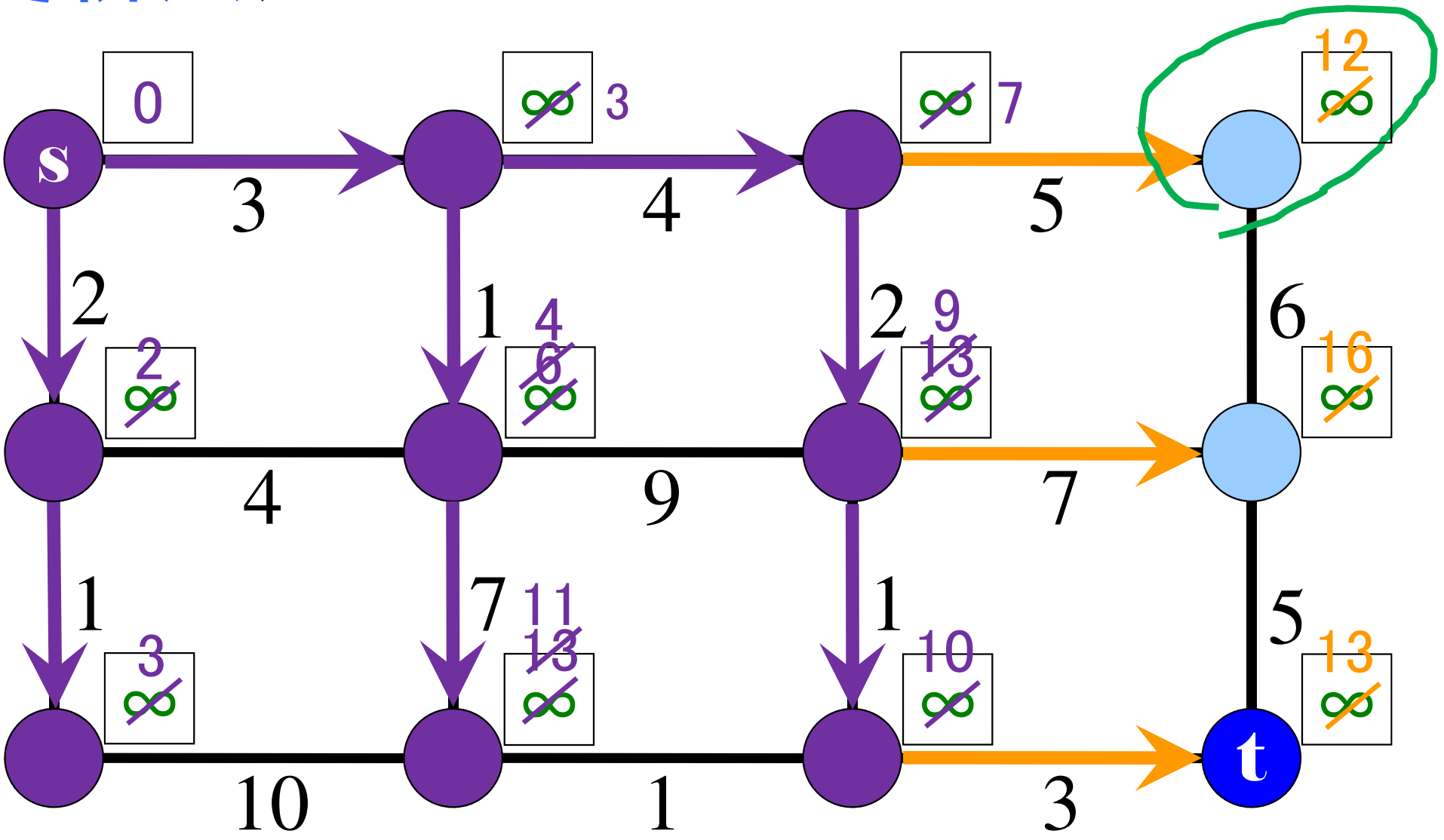
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



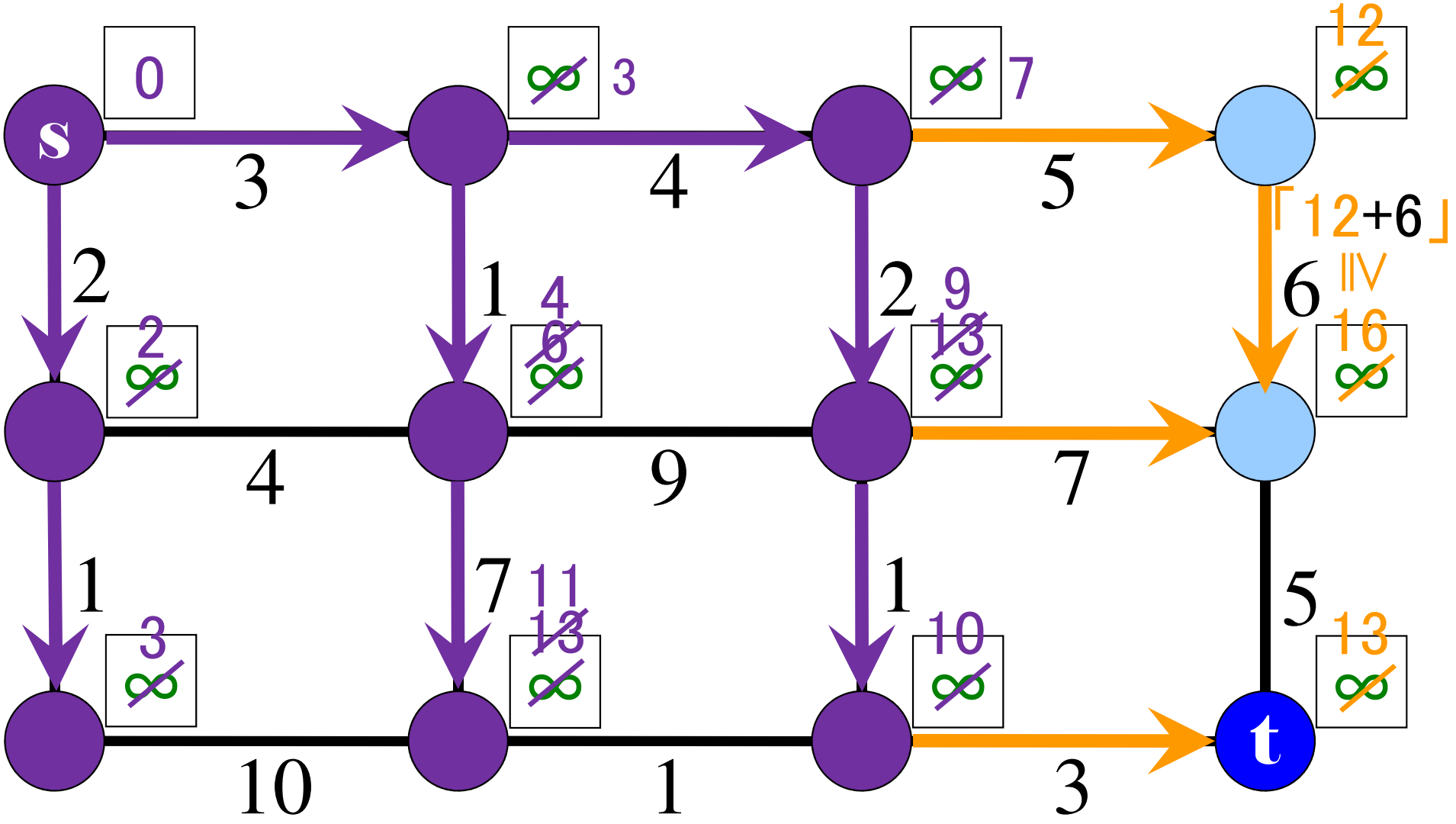
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

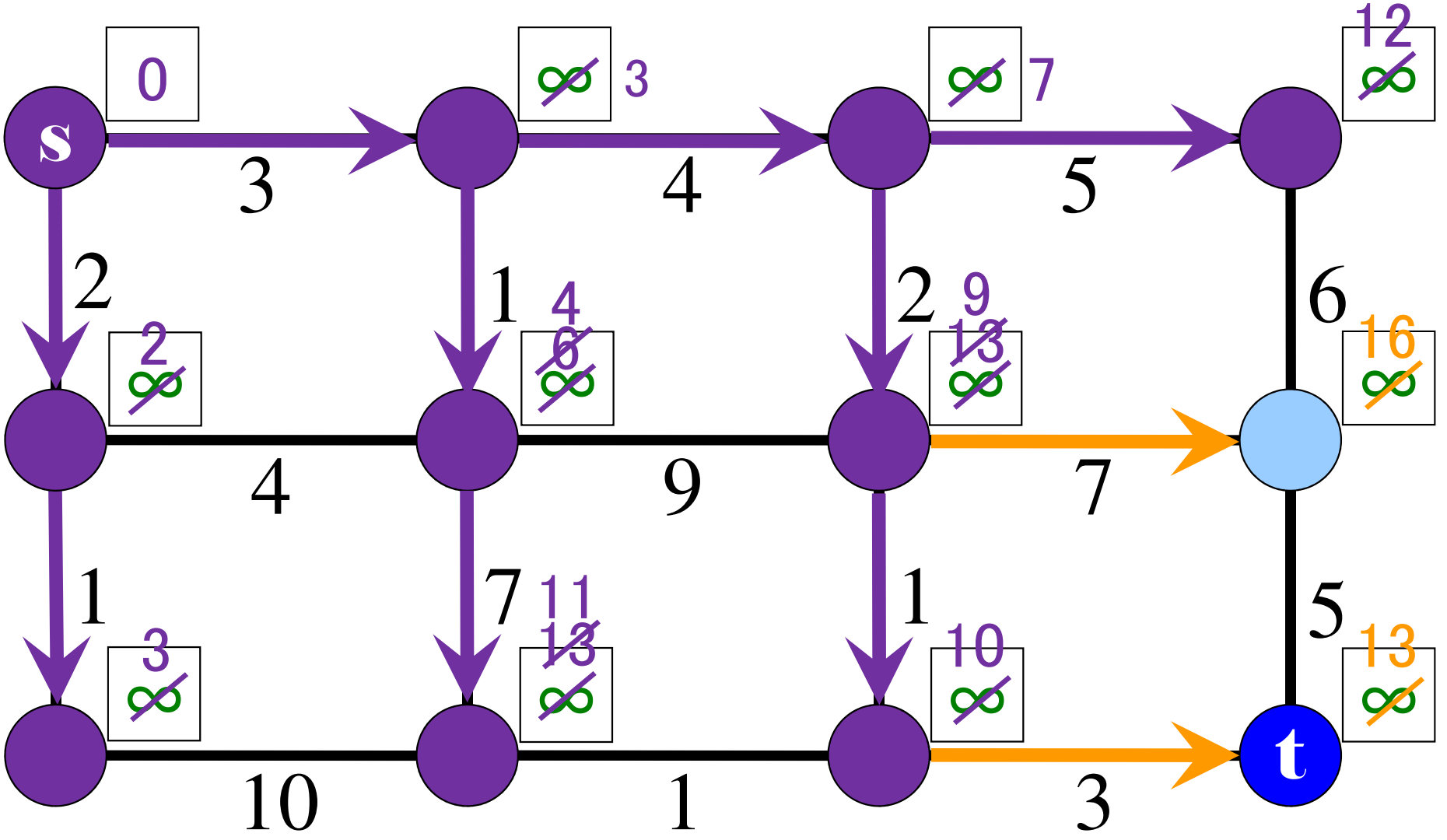


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

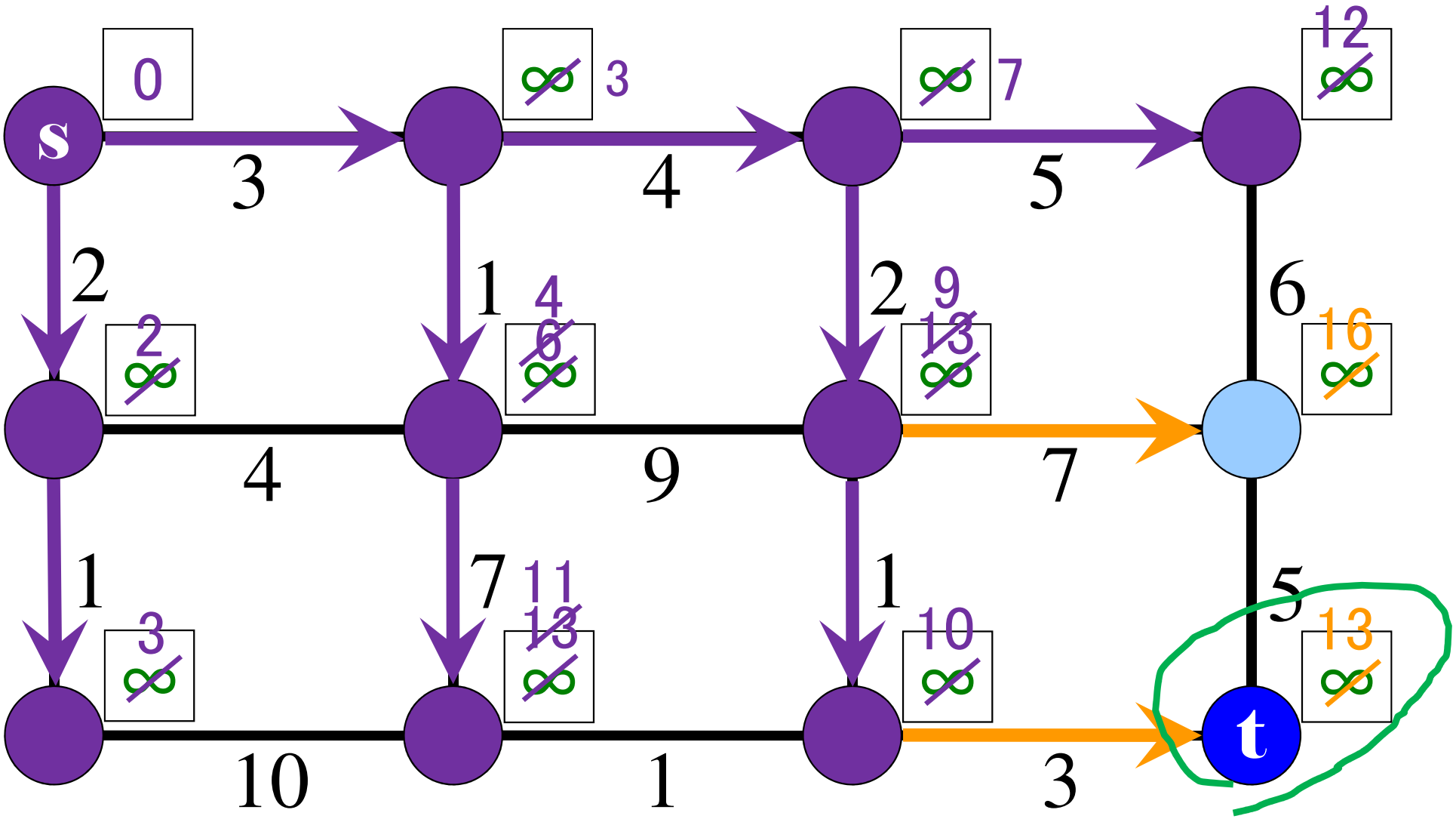
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



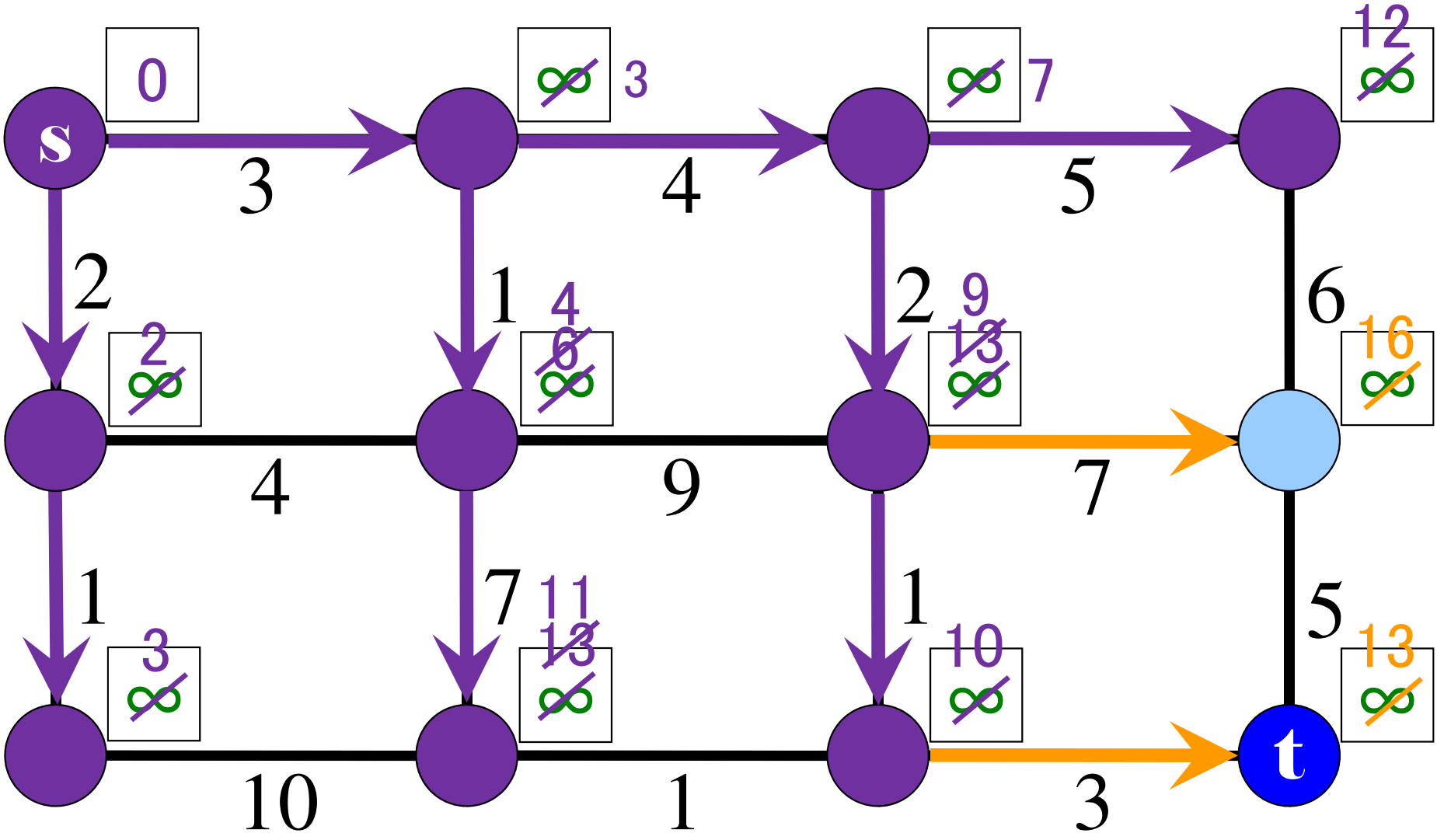
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

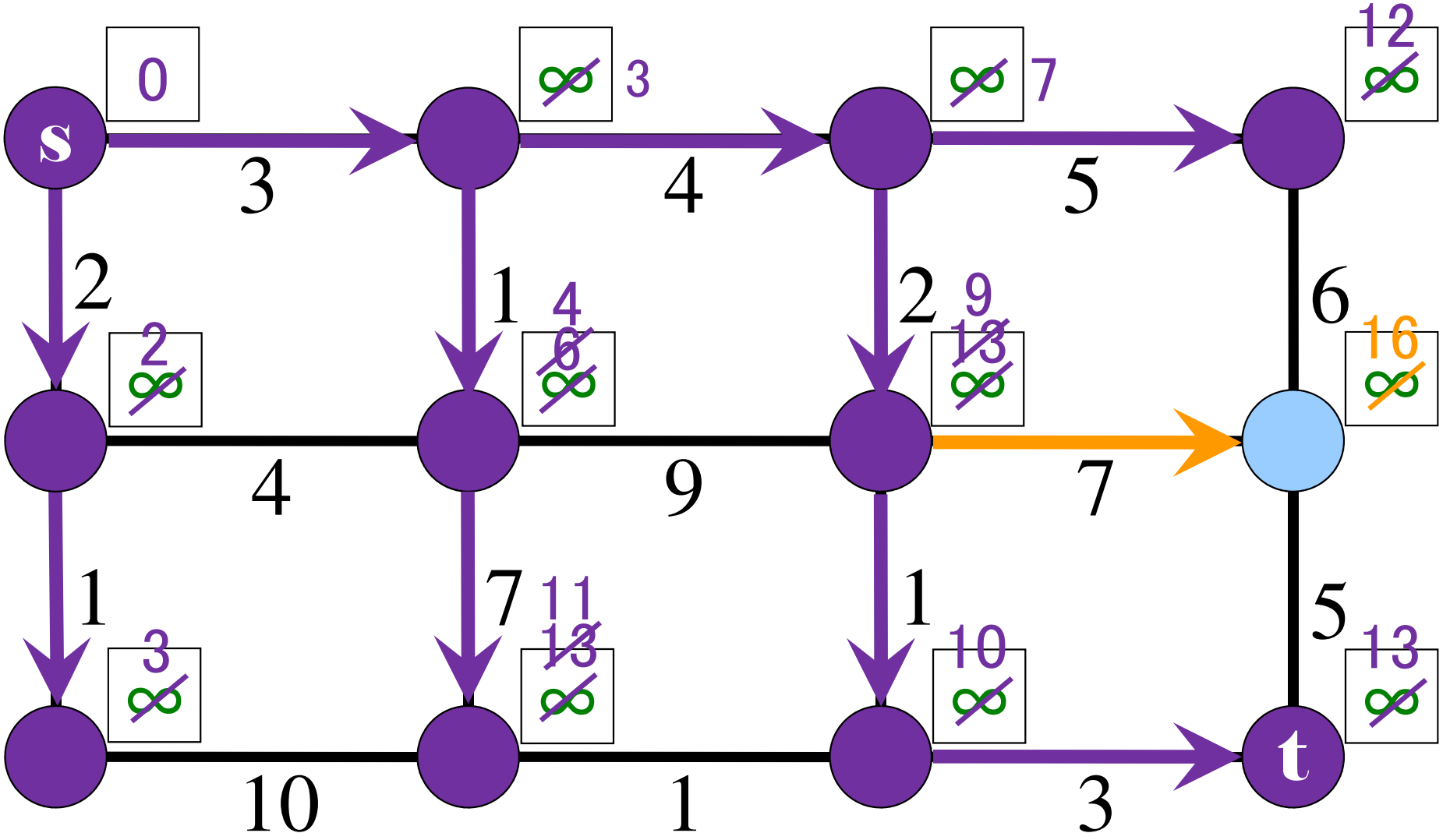


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

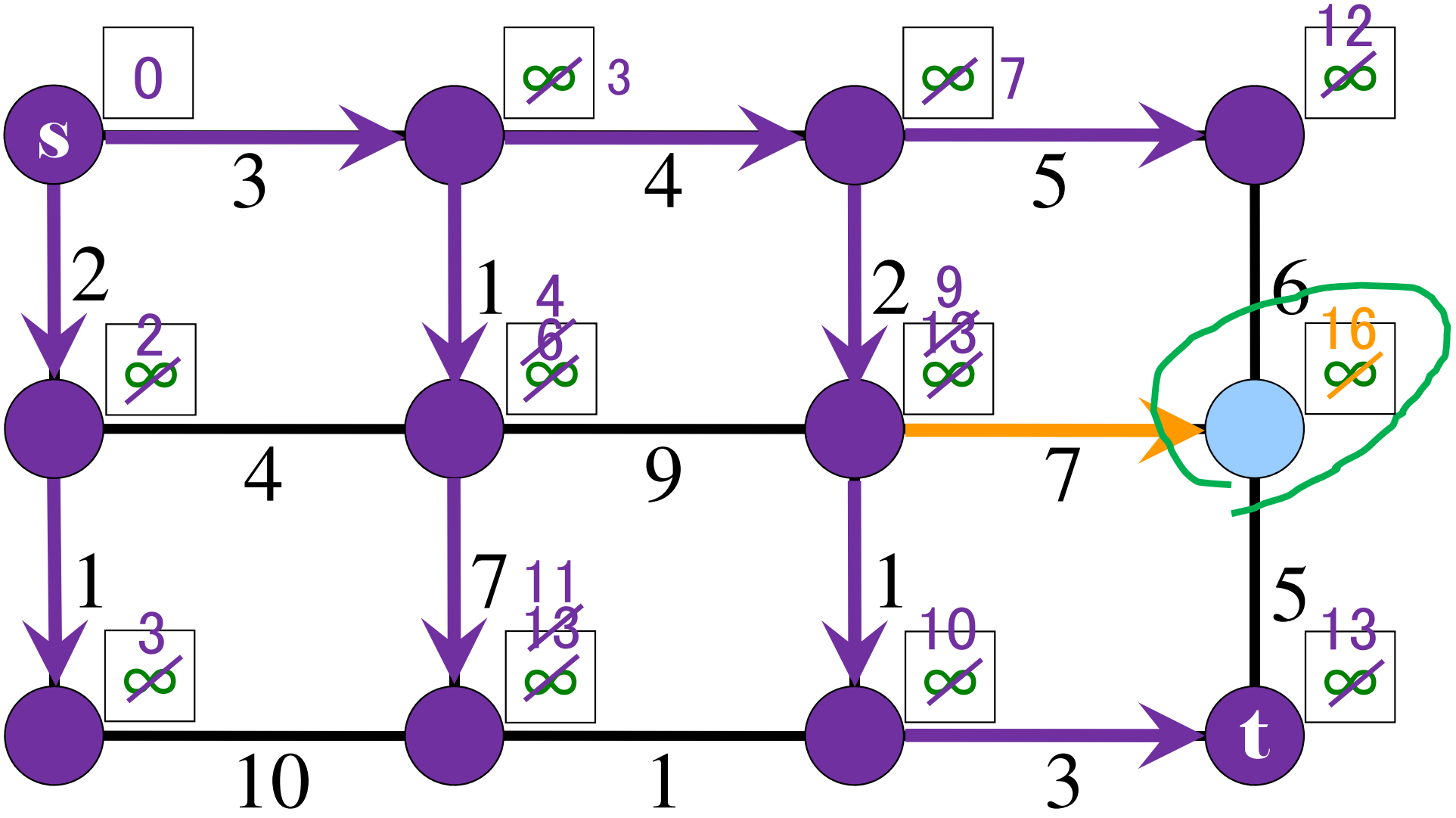
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



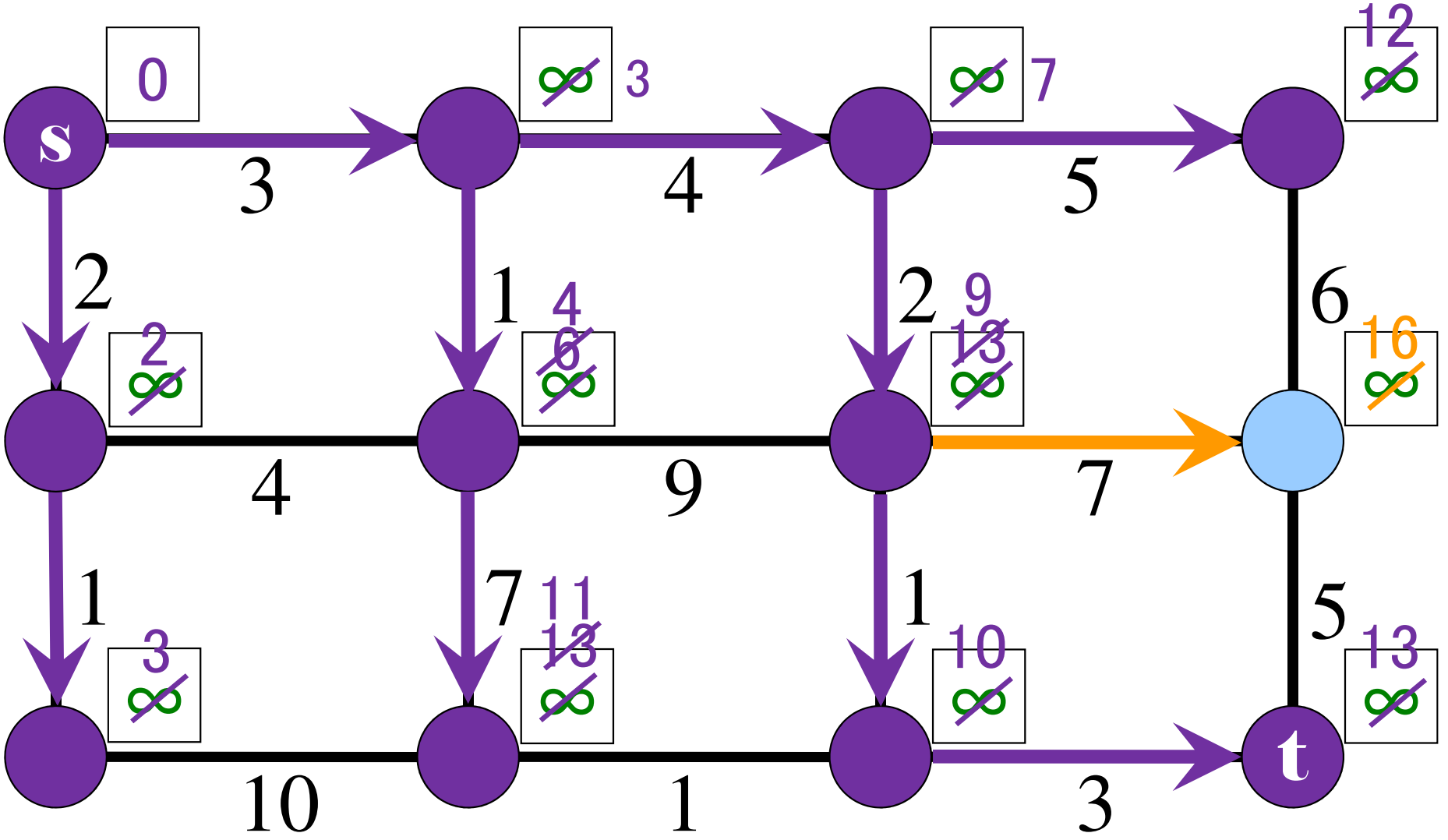
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

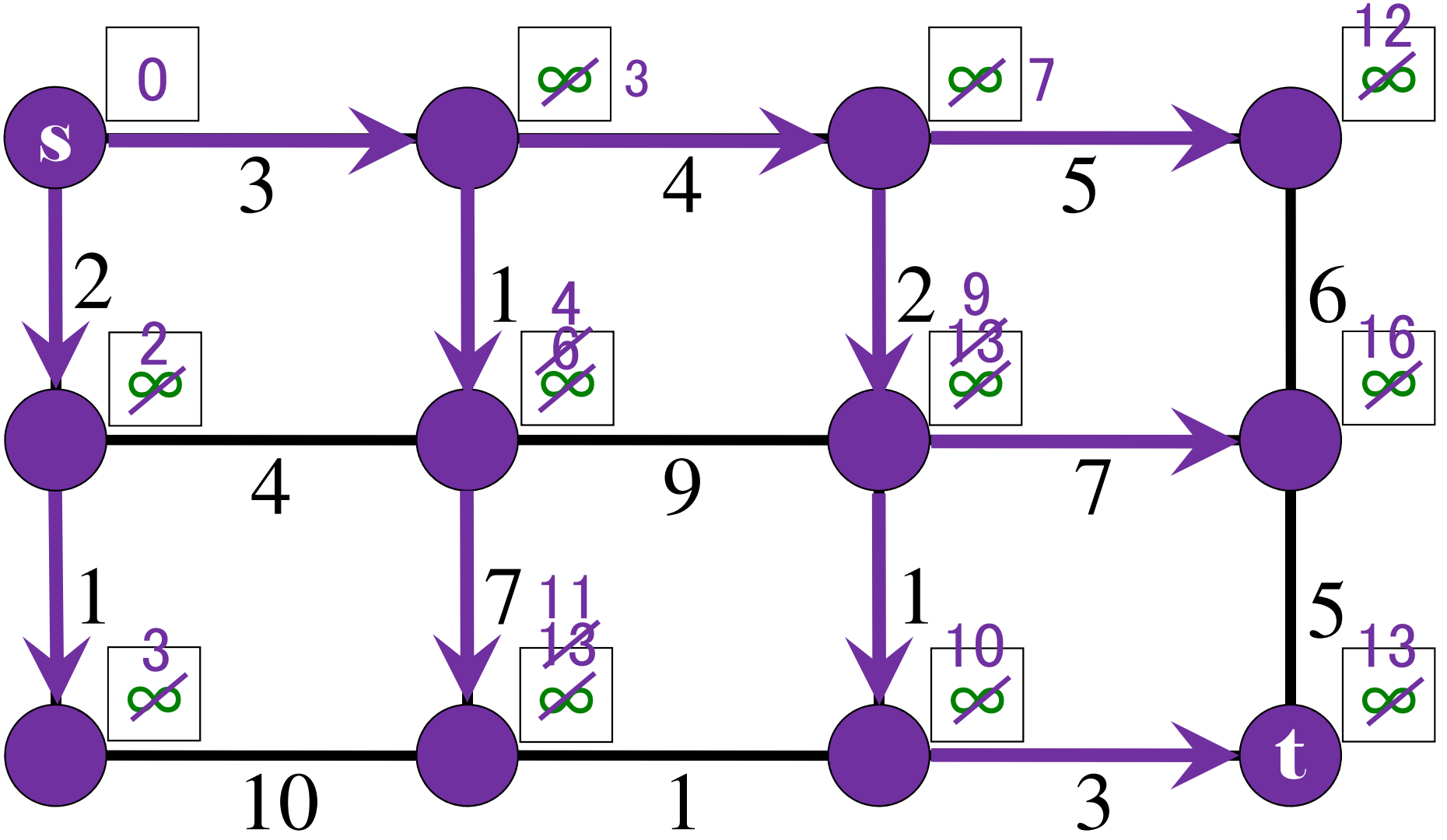


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

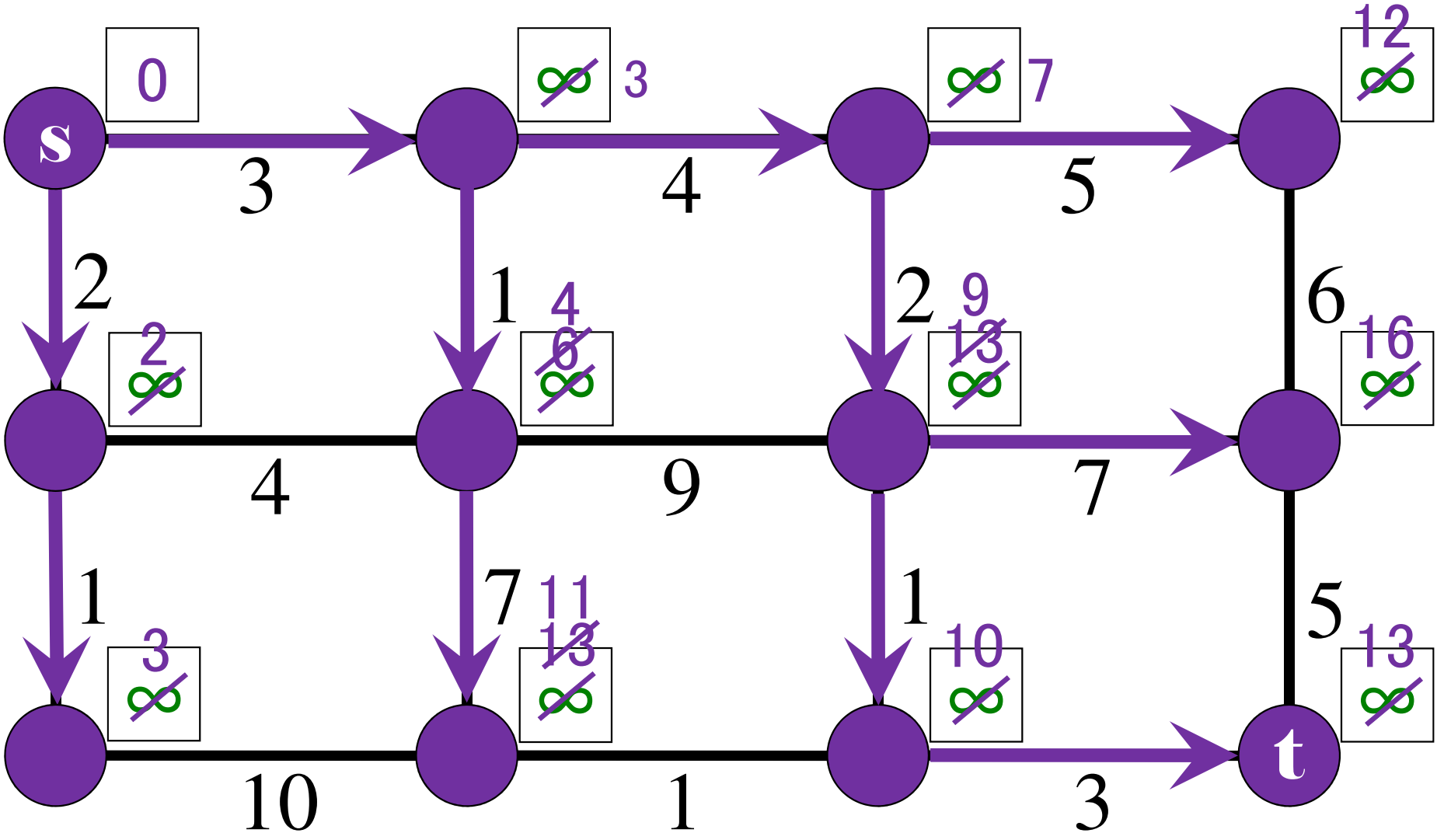
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



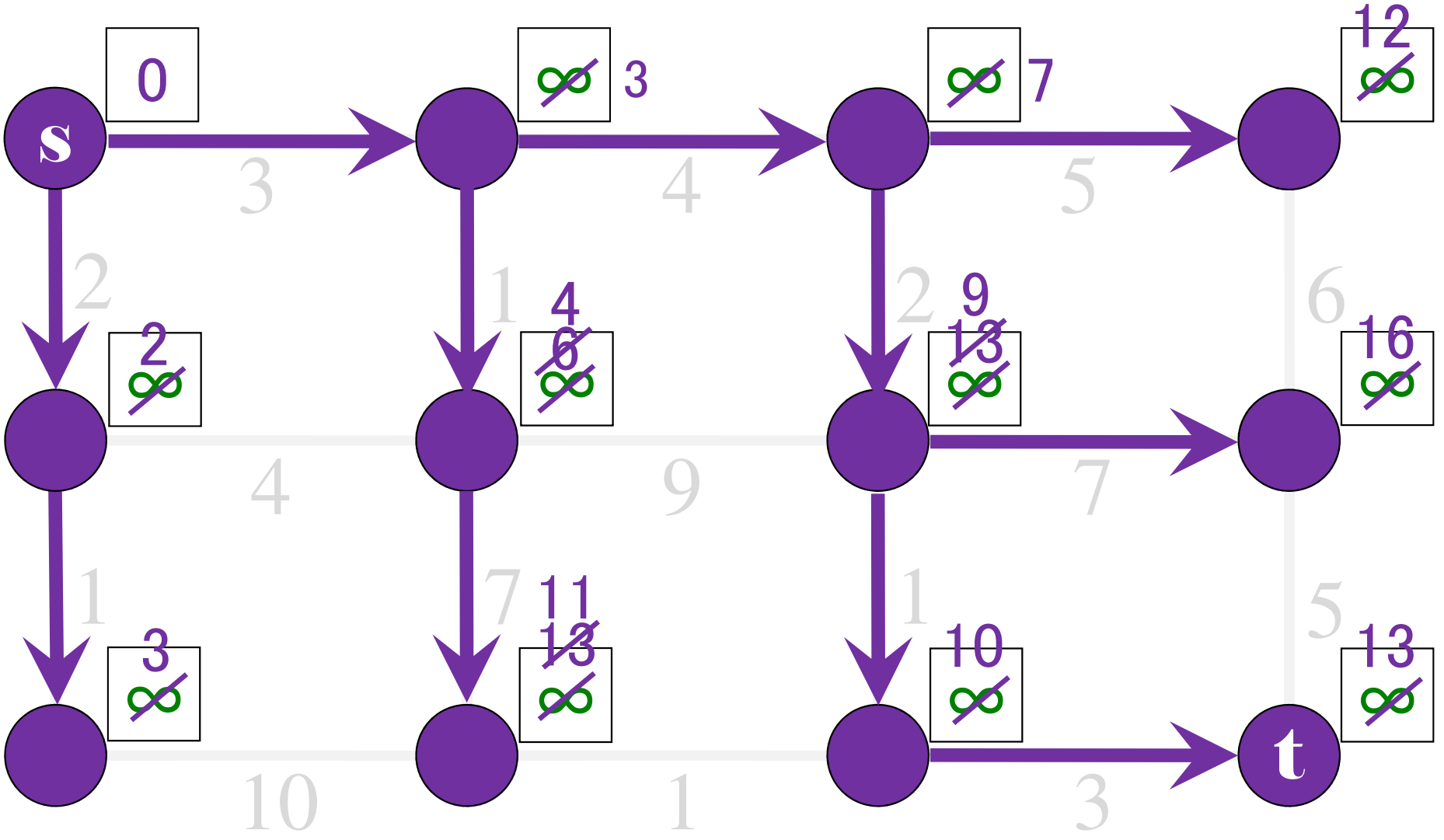
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



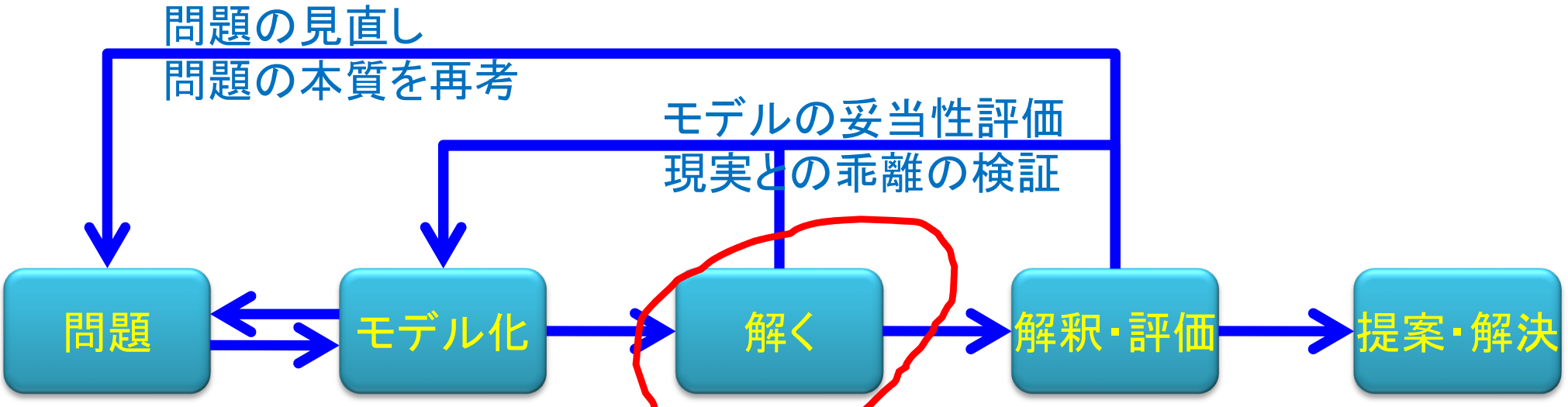
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

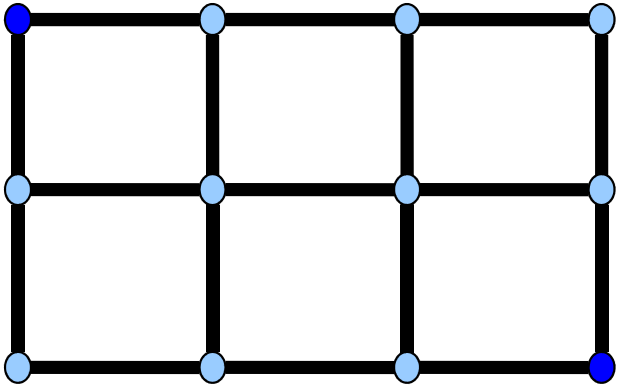
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

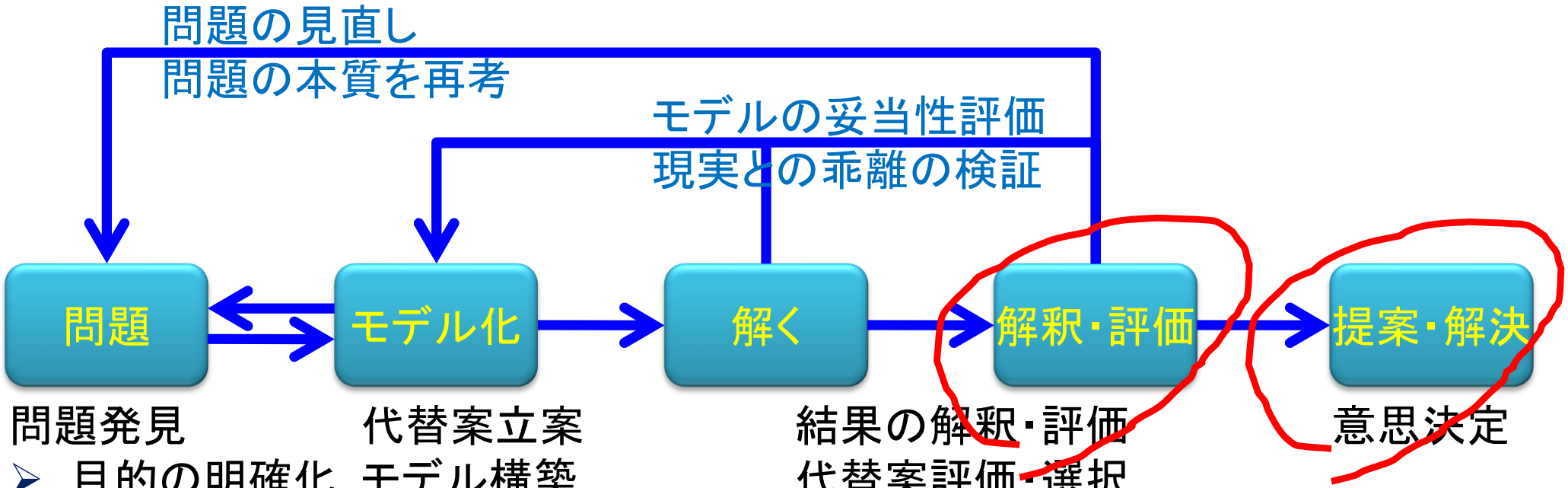
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

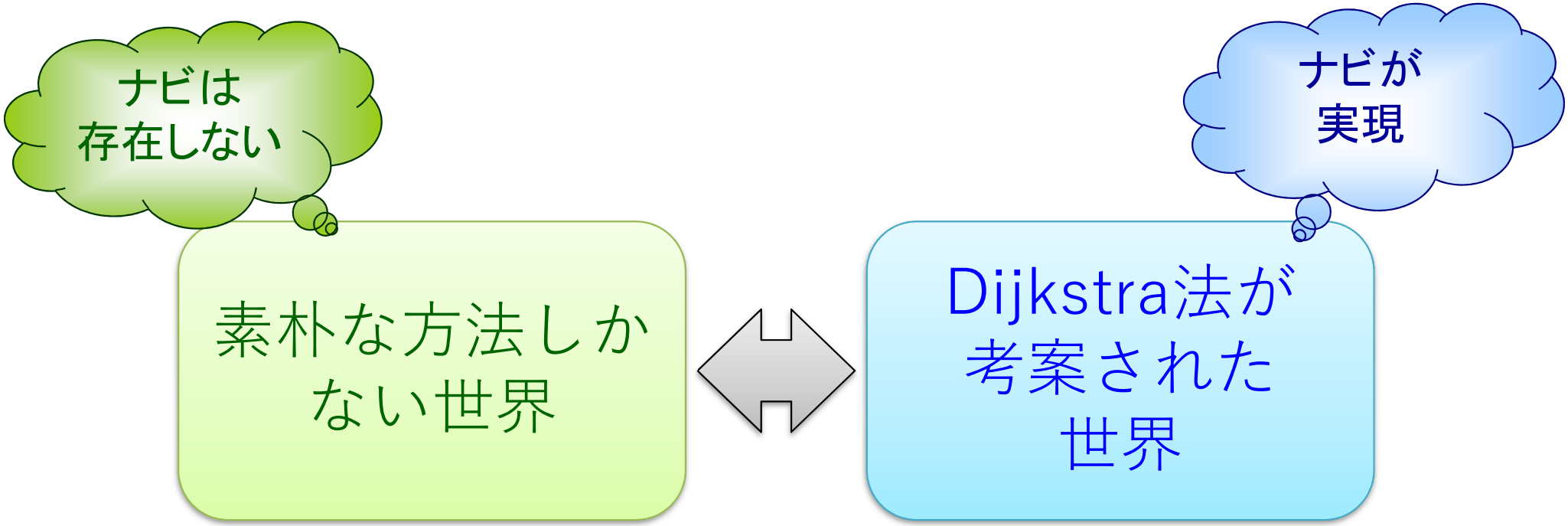
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

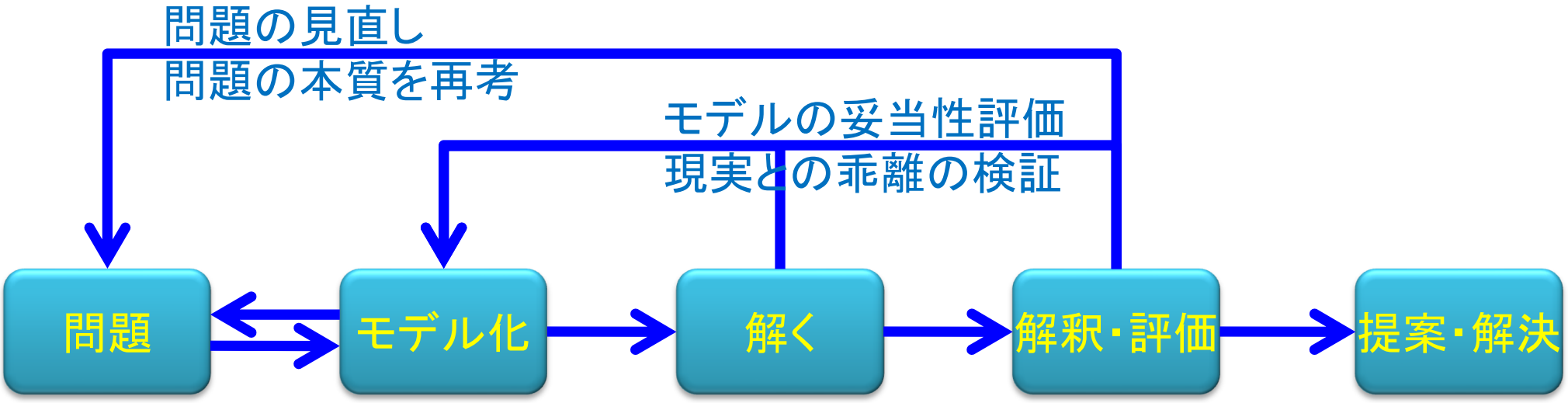
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



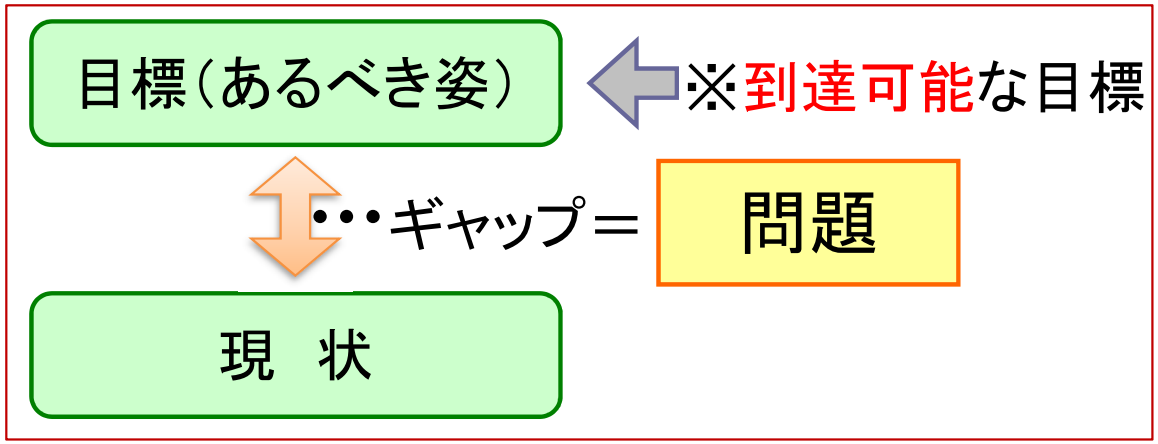
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

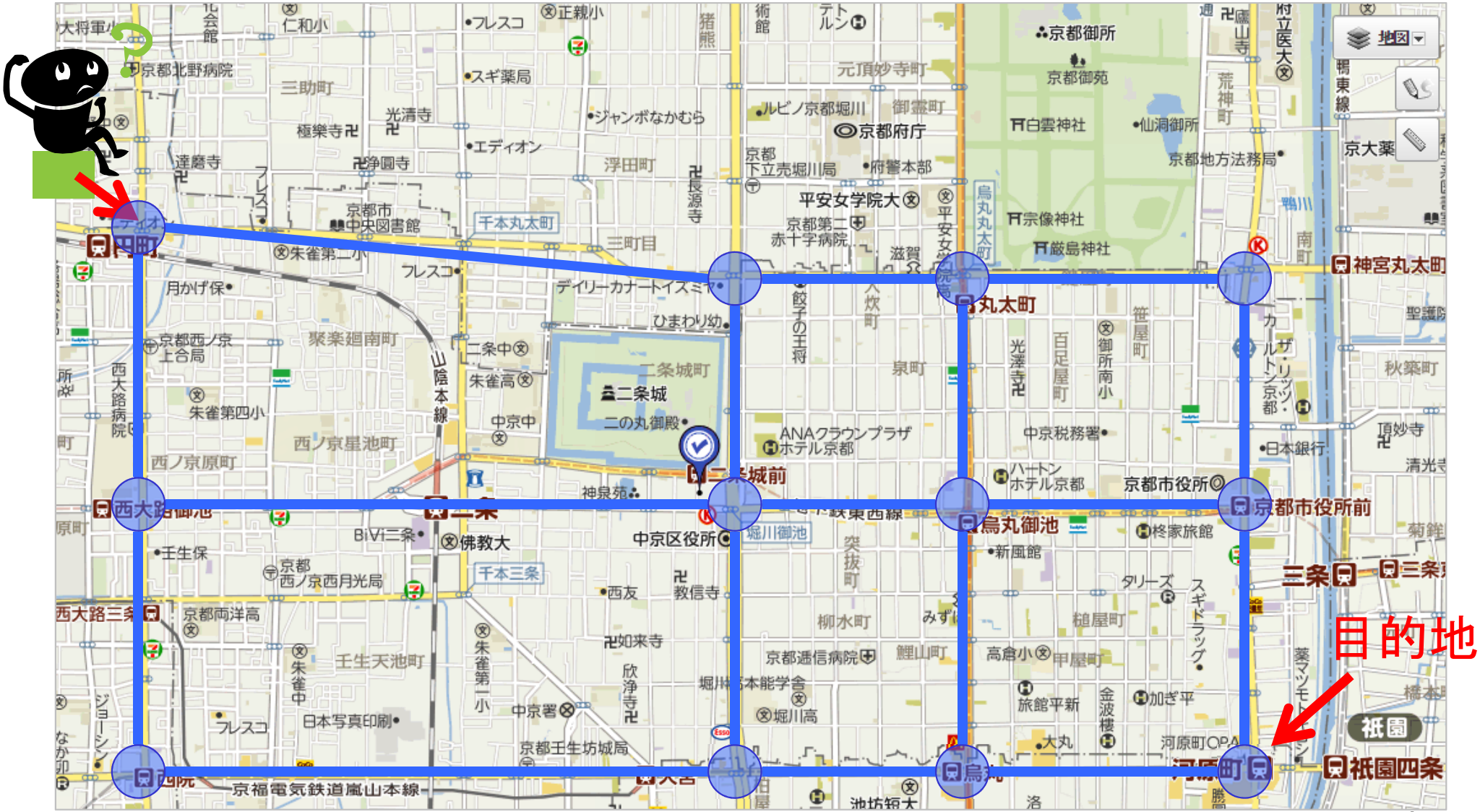
意思決定

問題の定義



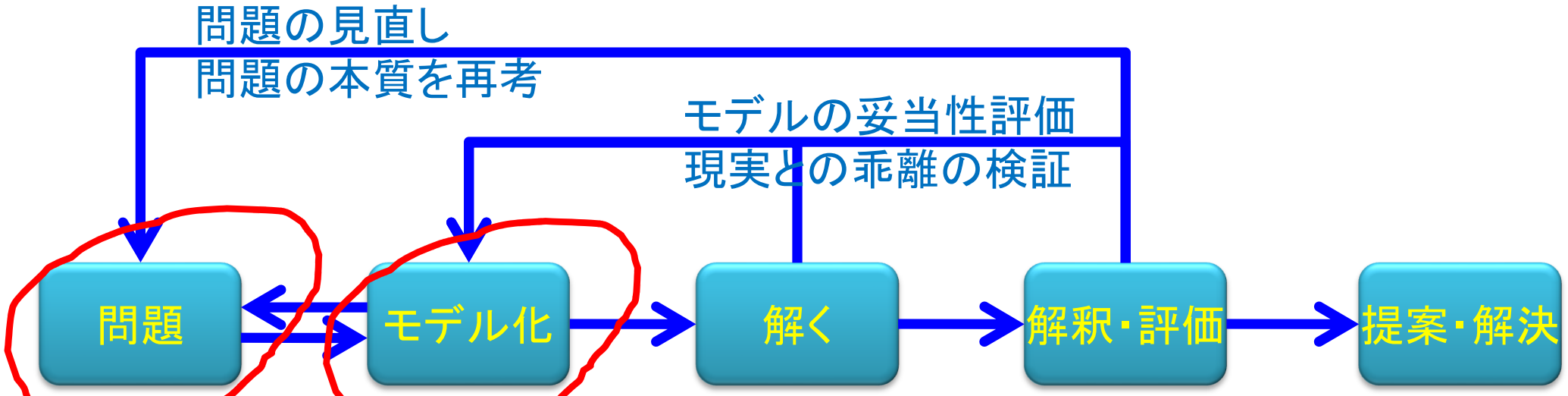
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

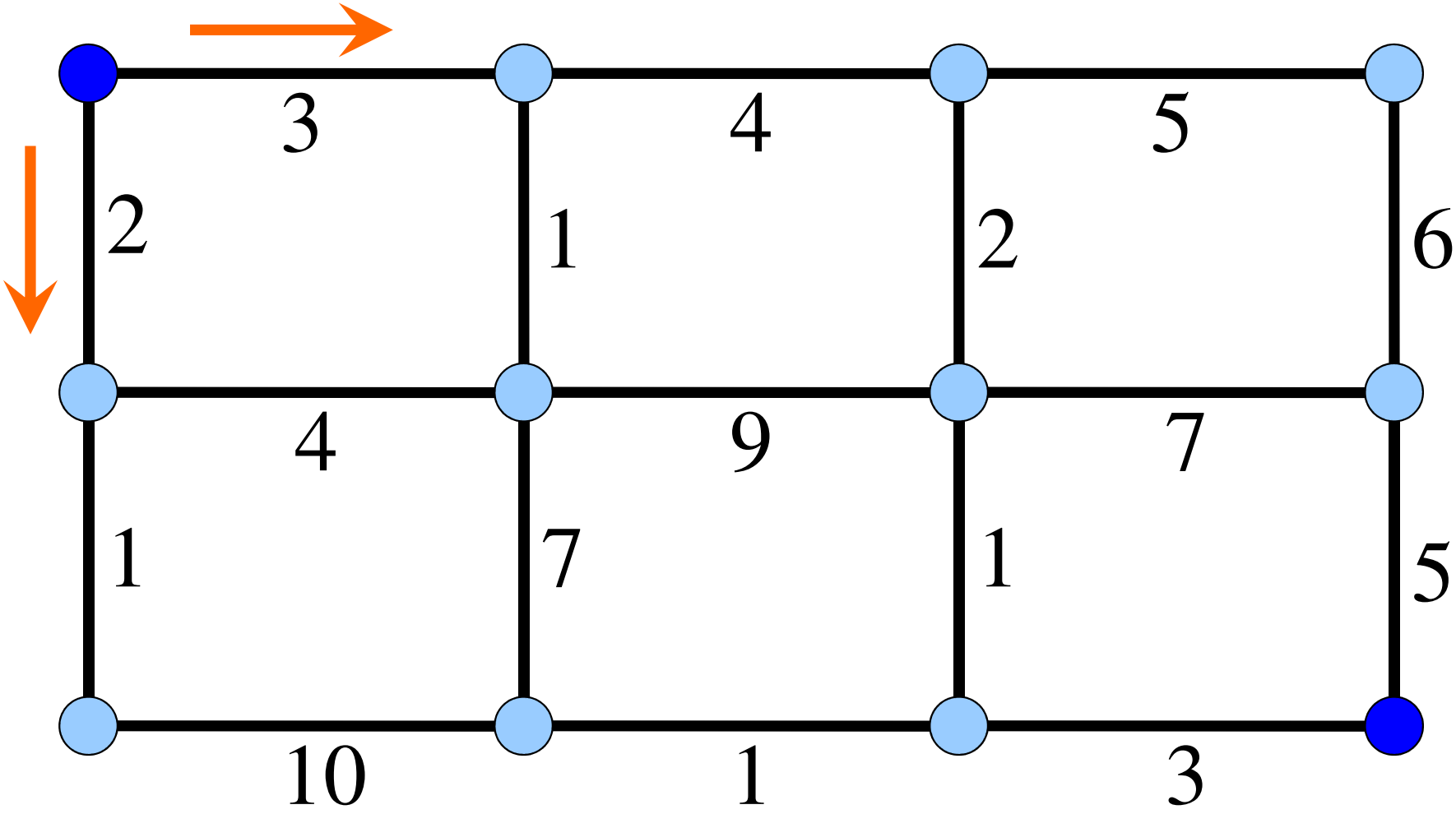
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

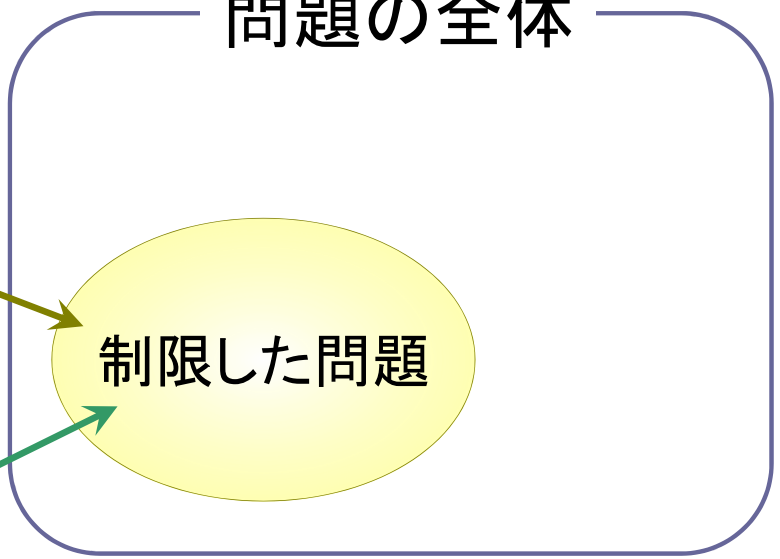


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

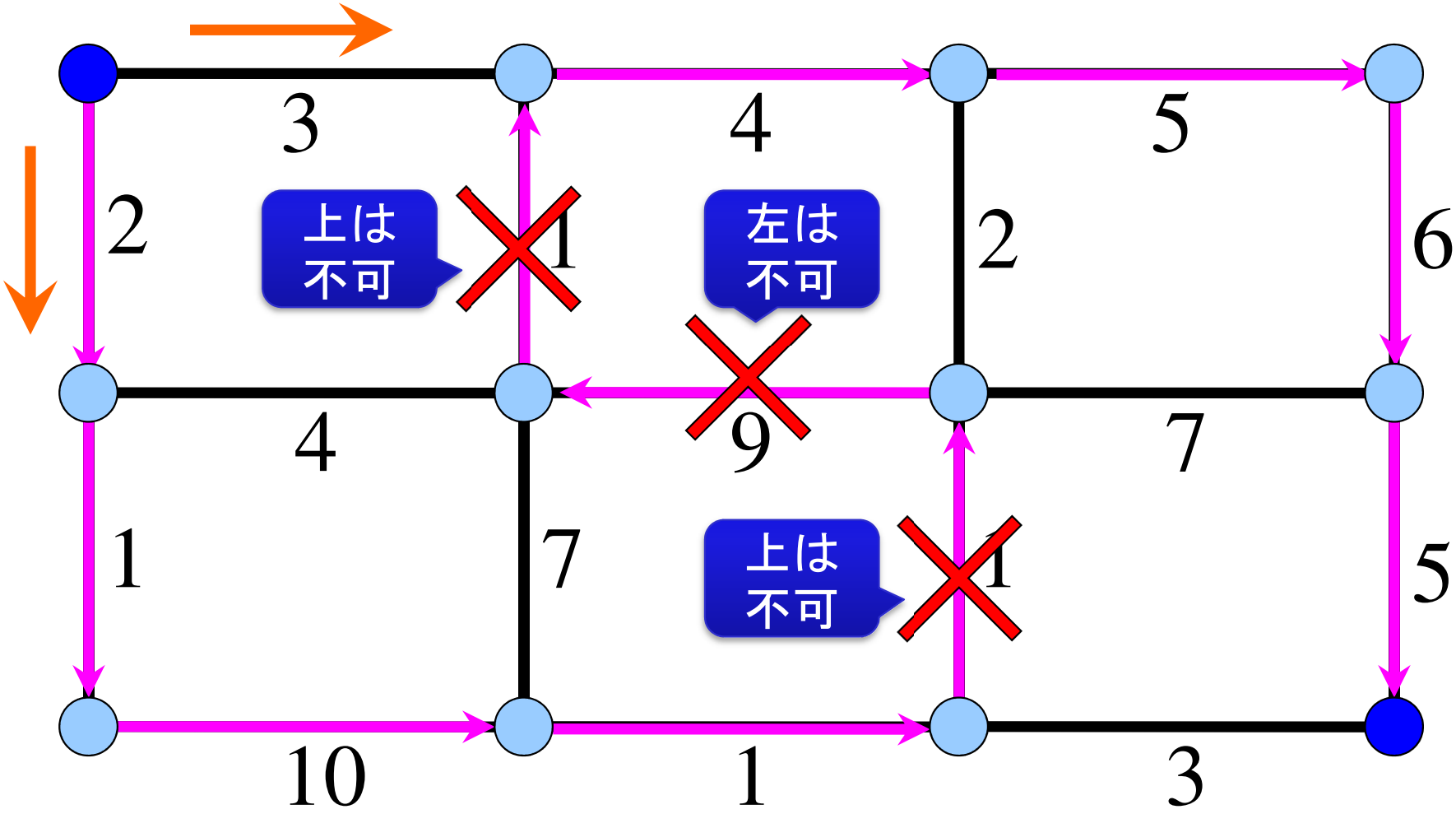
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

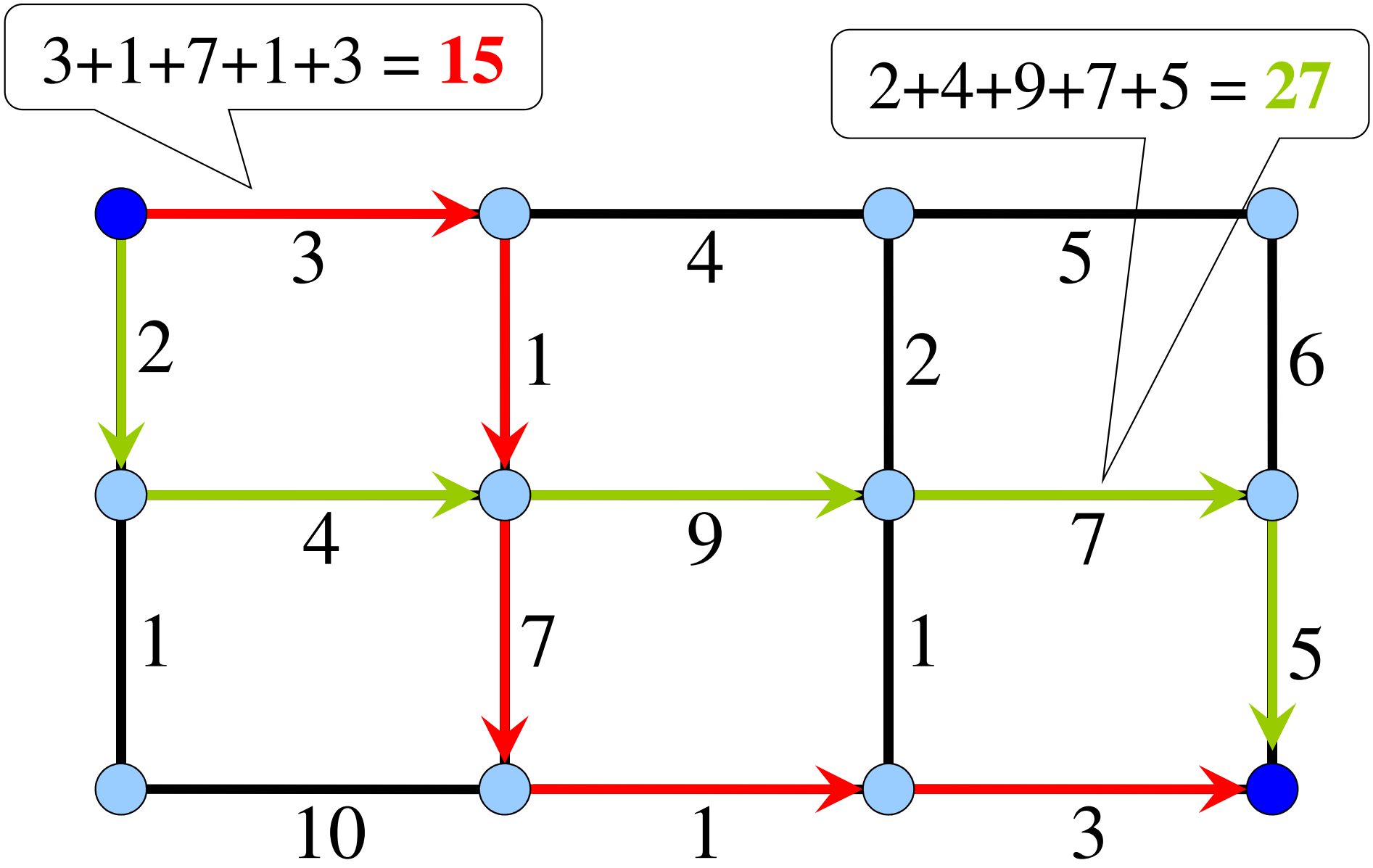
難しいなら易しくすればいいのさ！

制限した問題

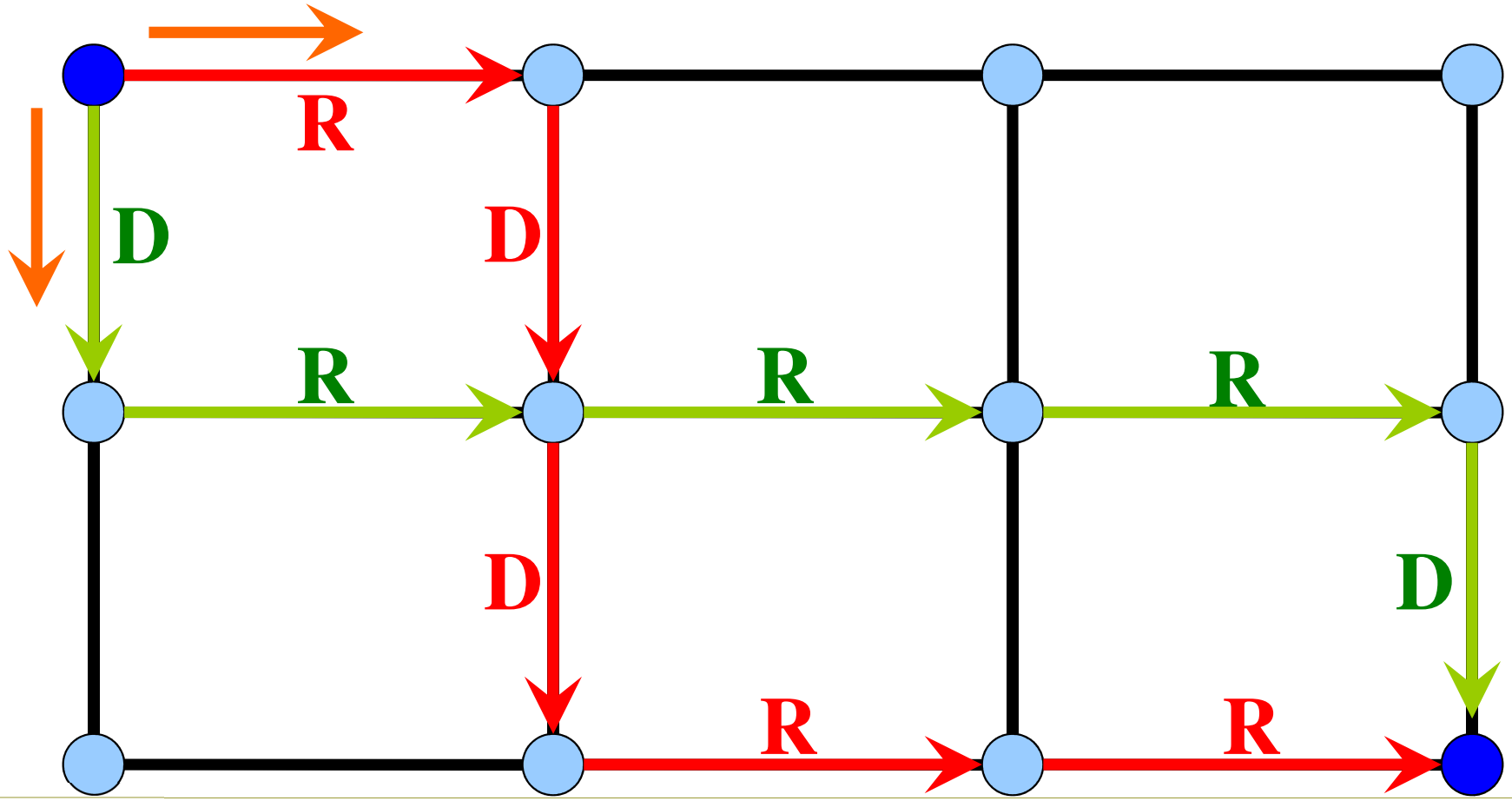
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



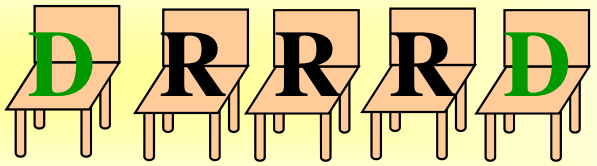
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



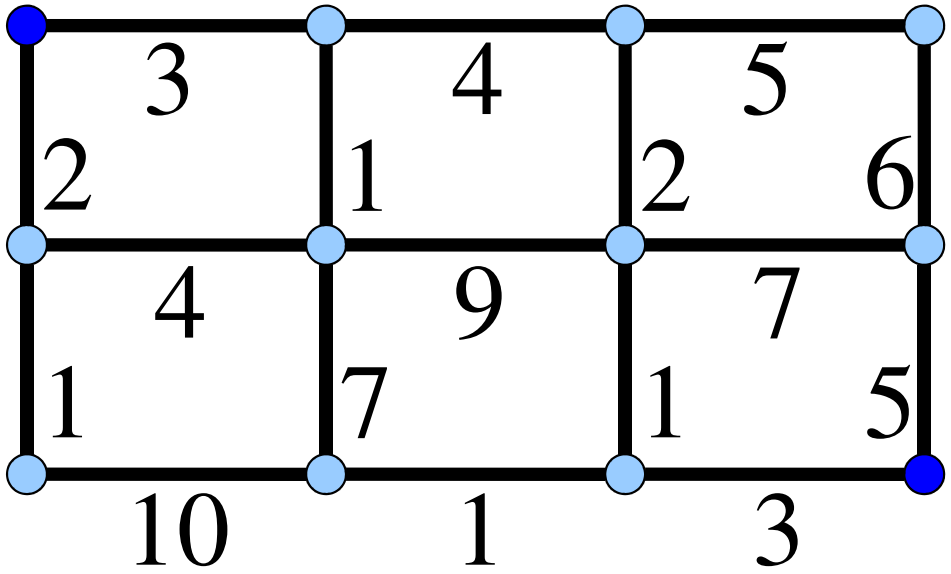
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

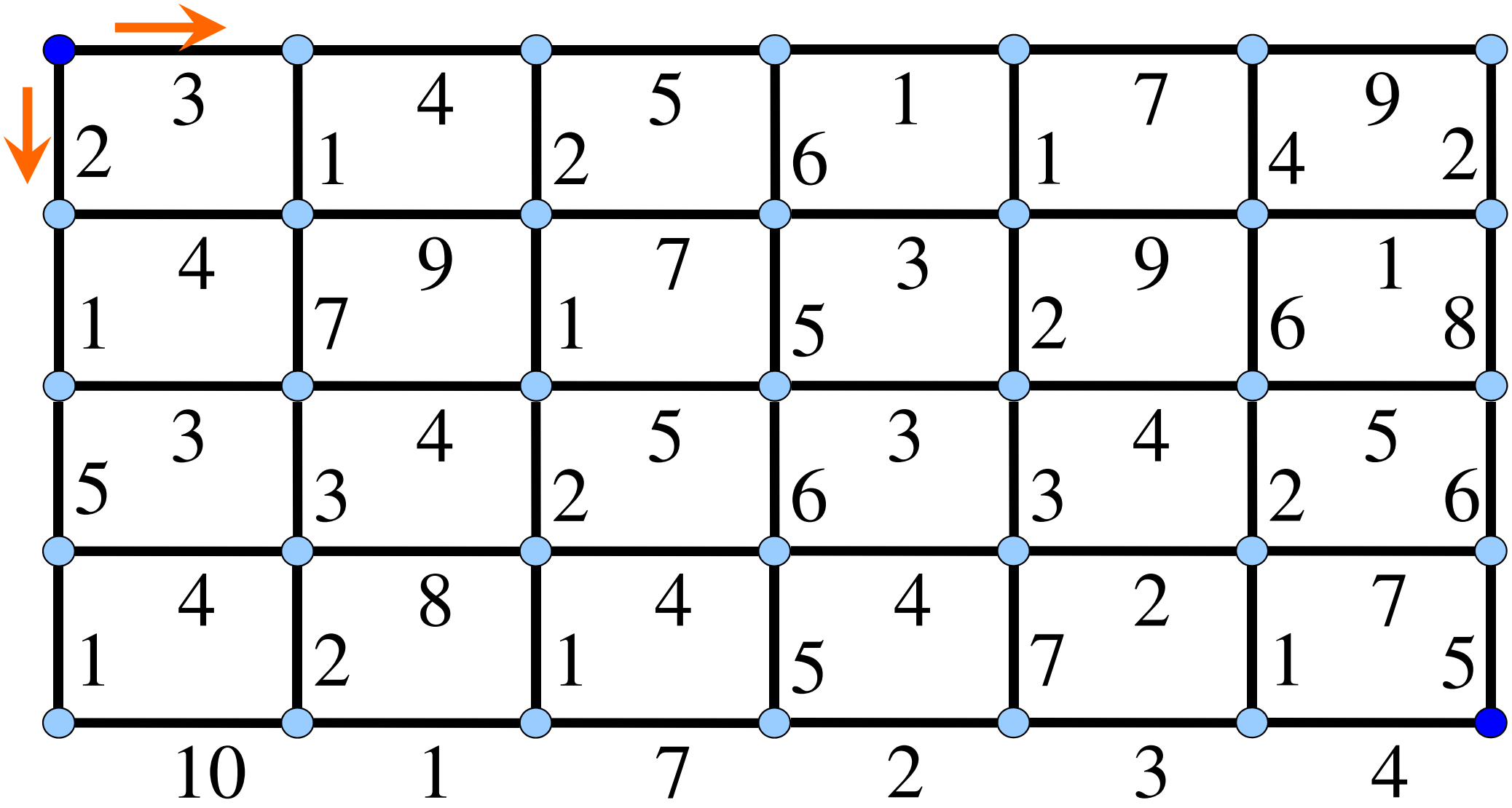
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

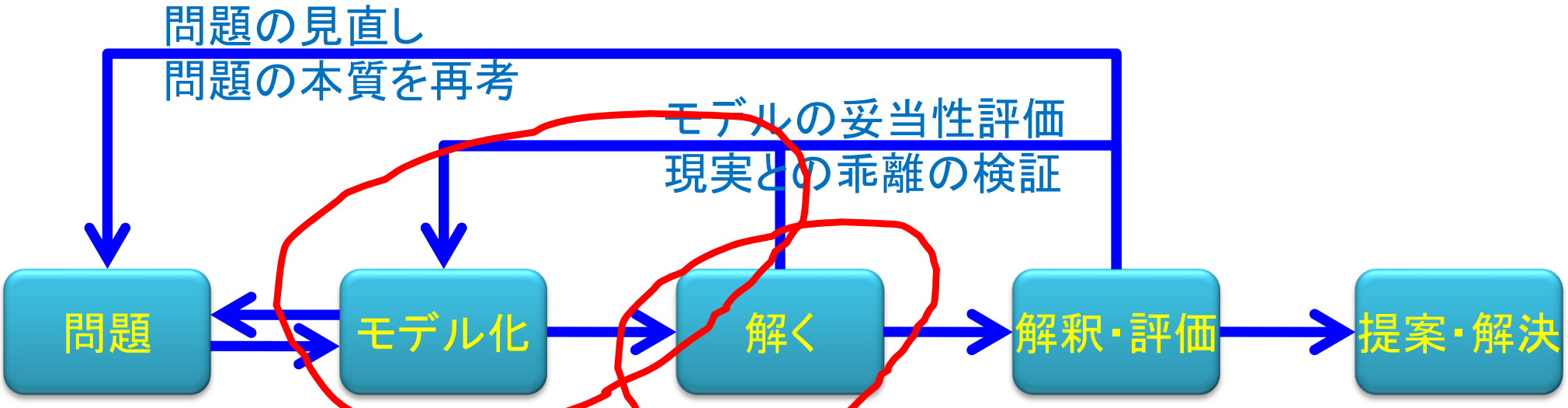
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

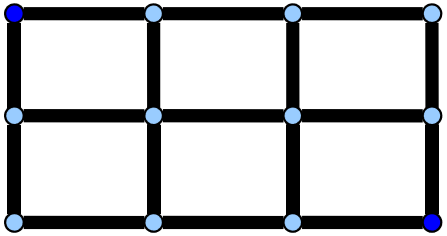
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

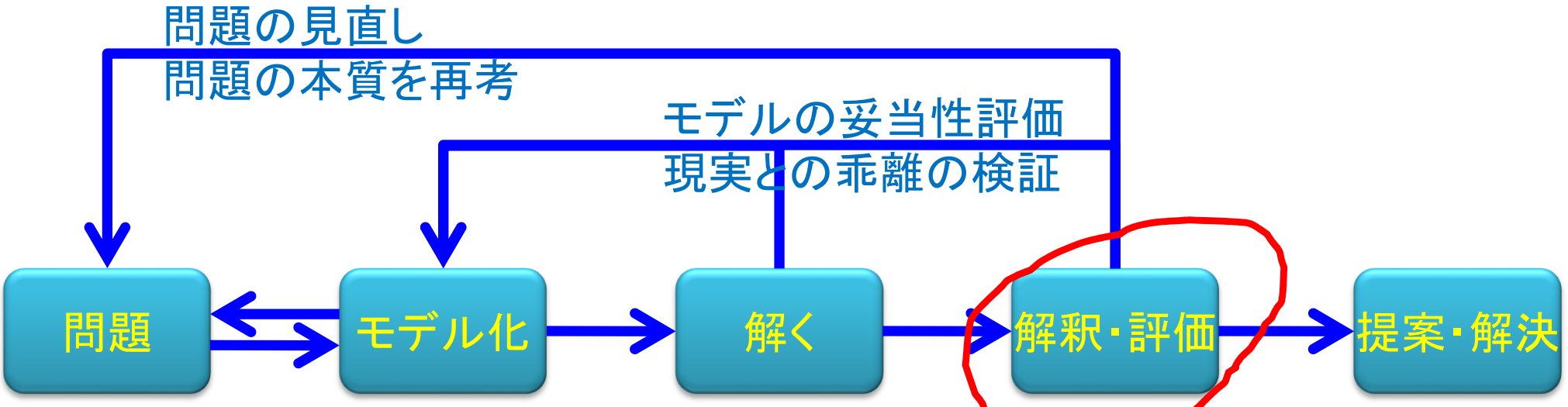
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

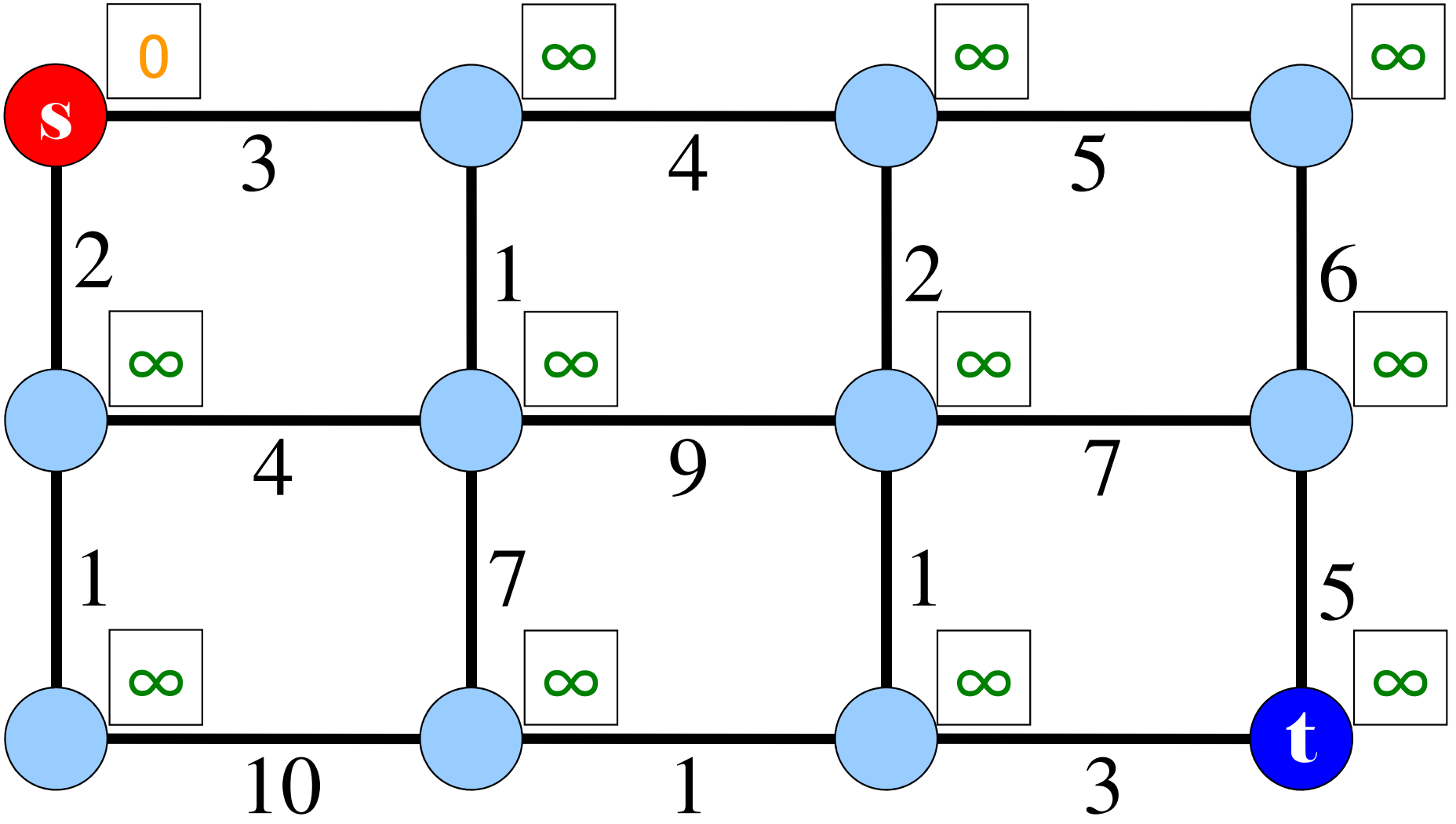
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

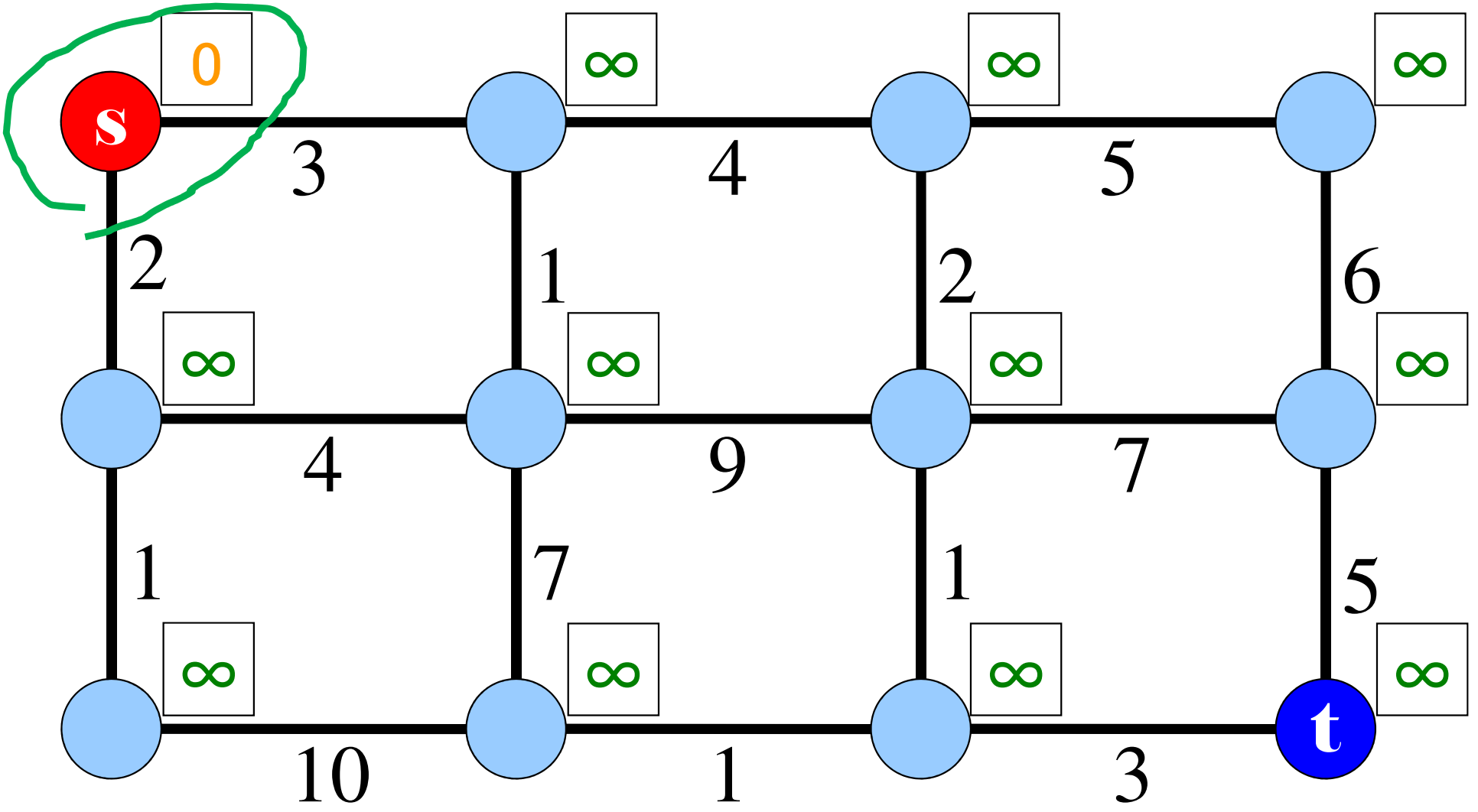
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



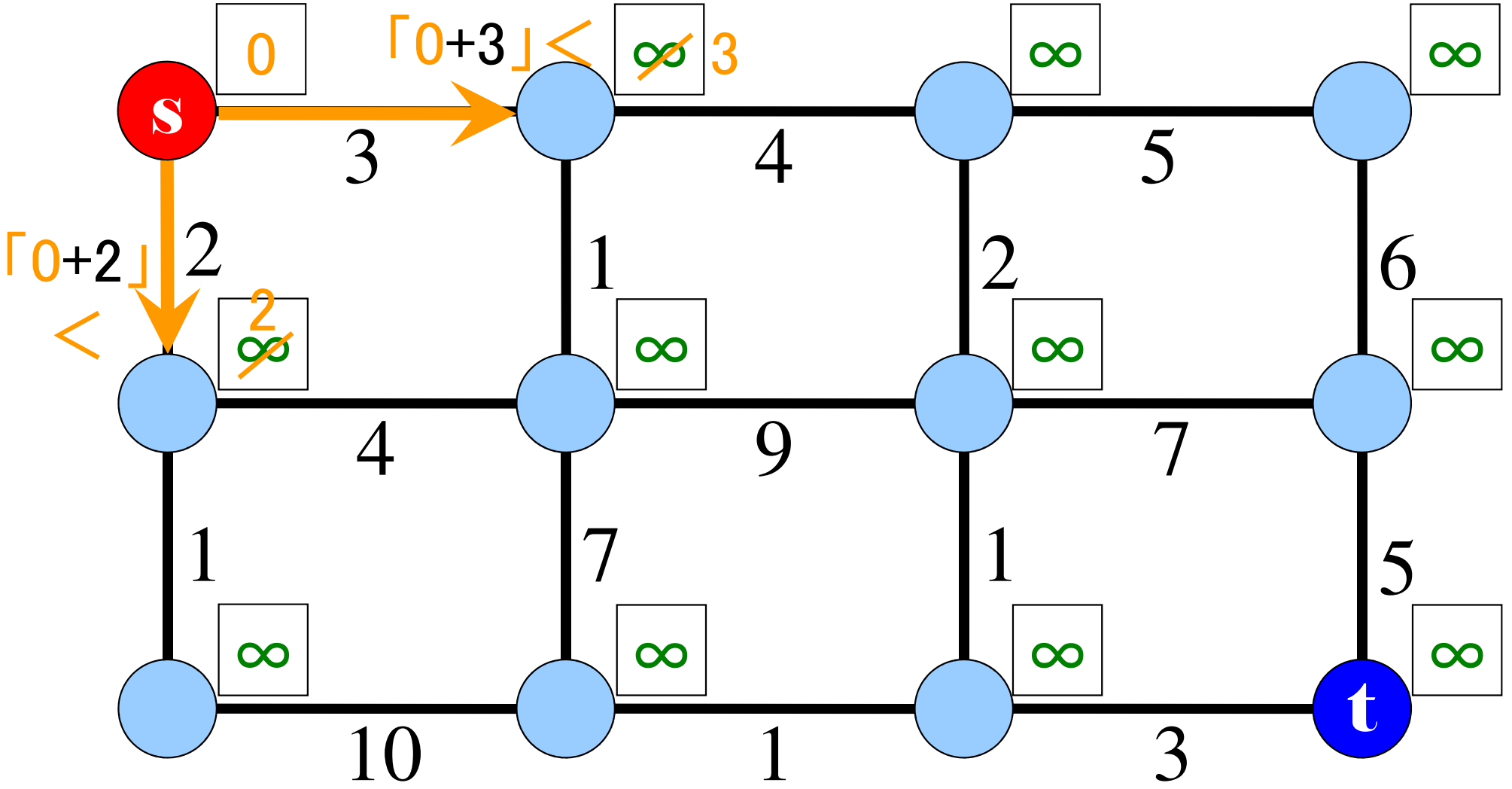
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

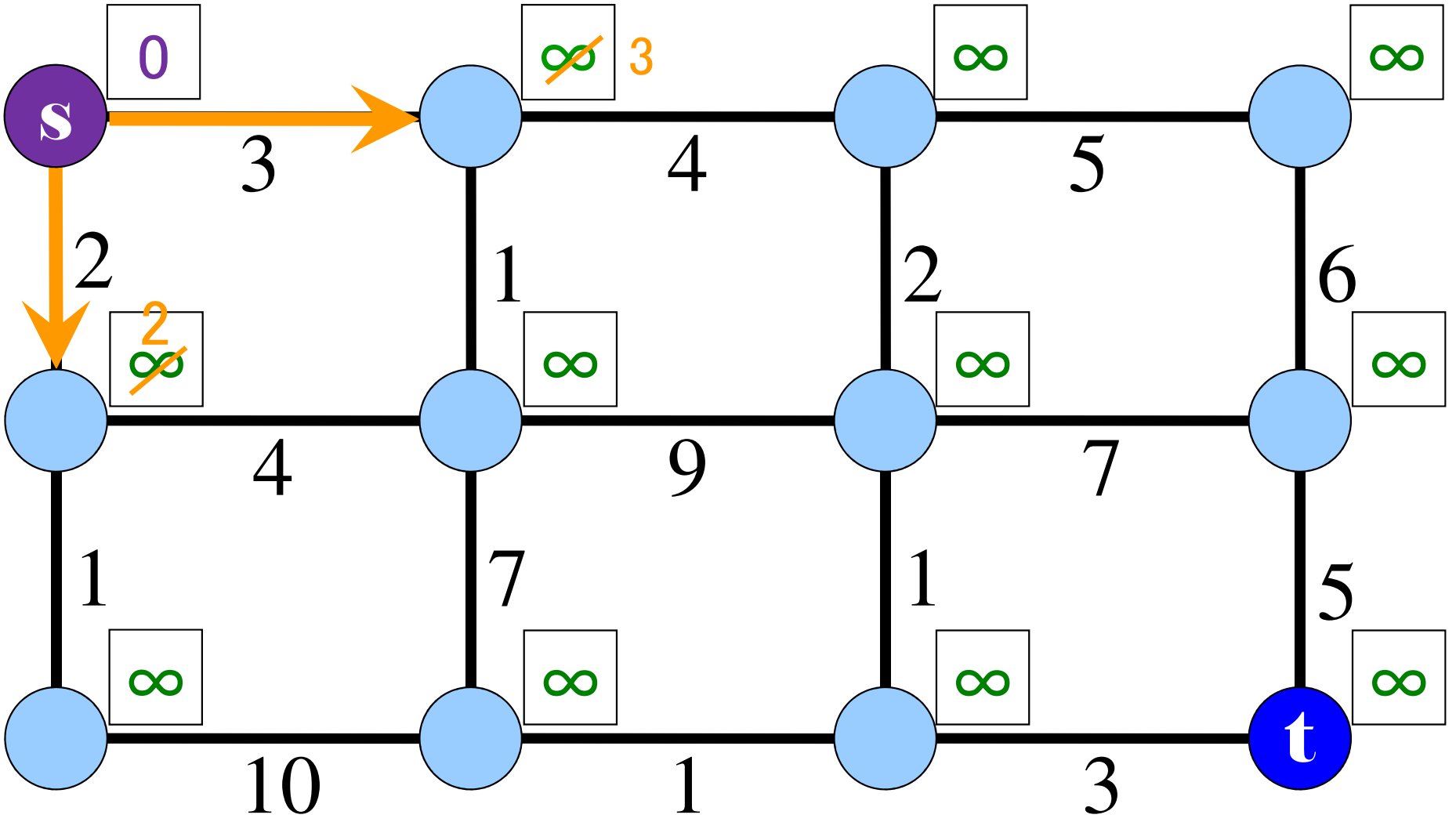


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

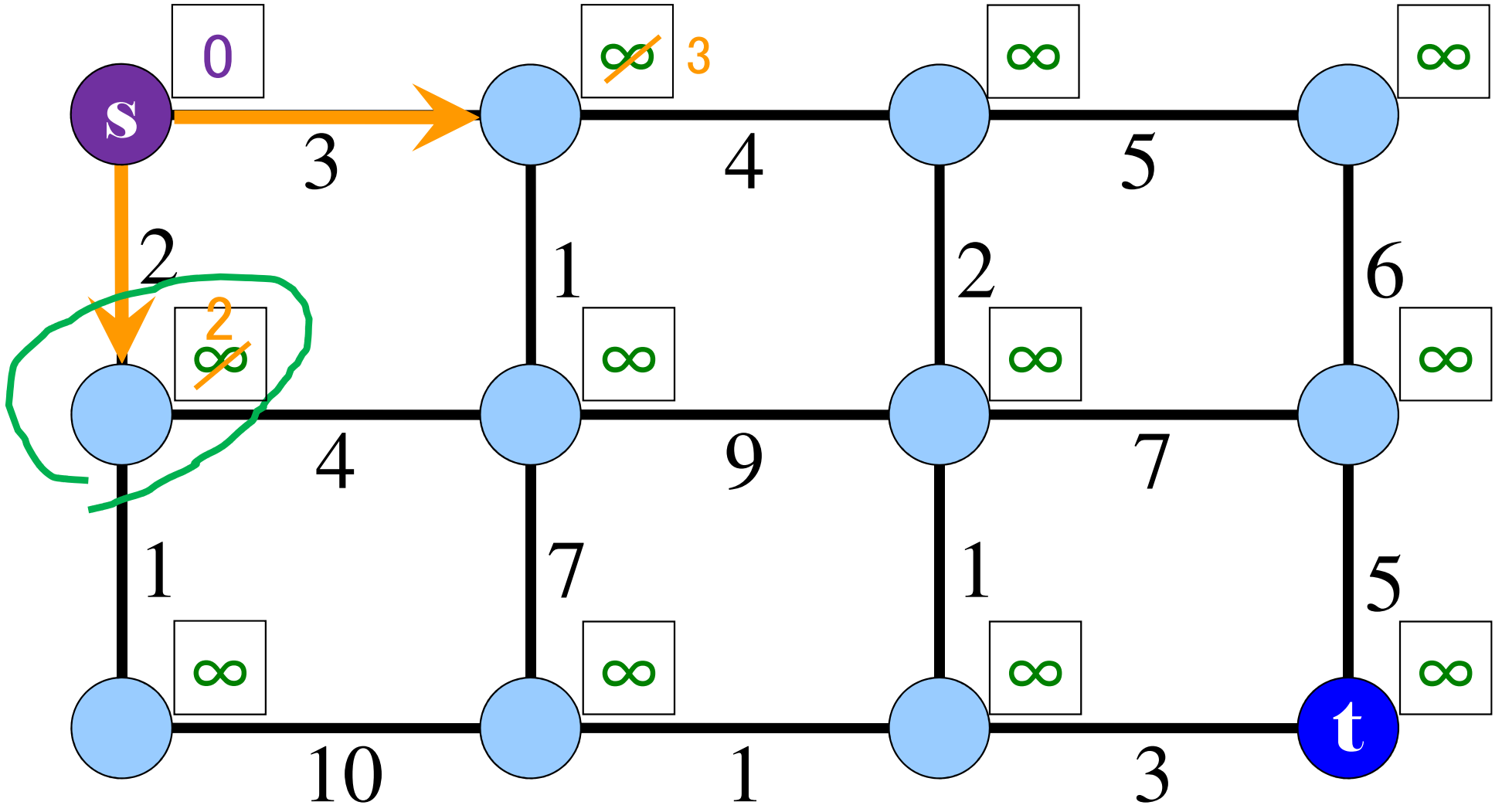
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



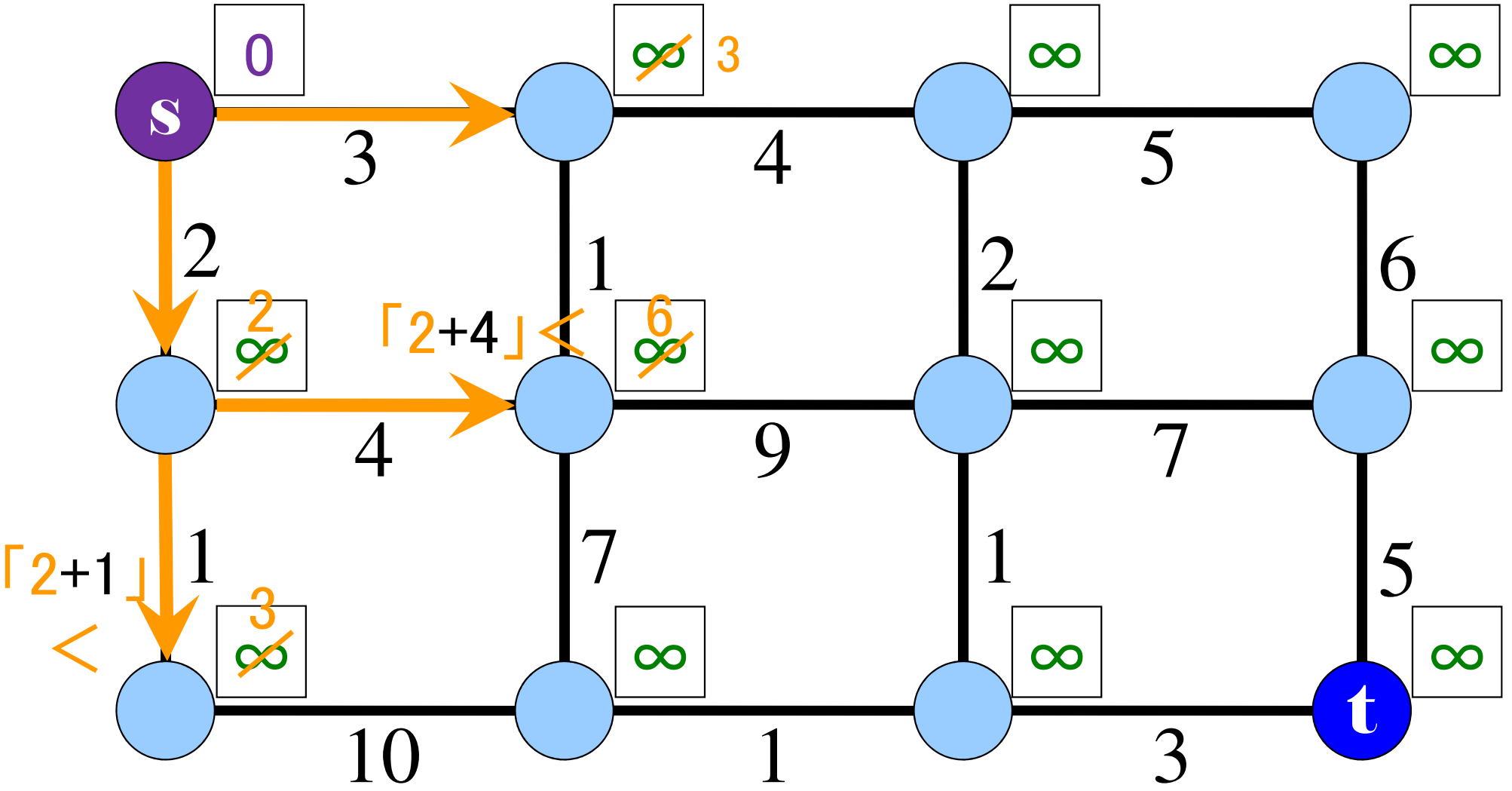
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

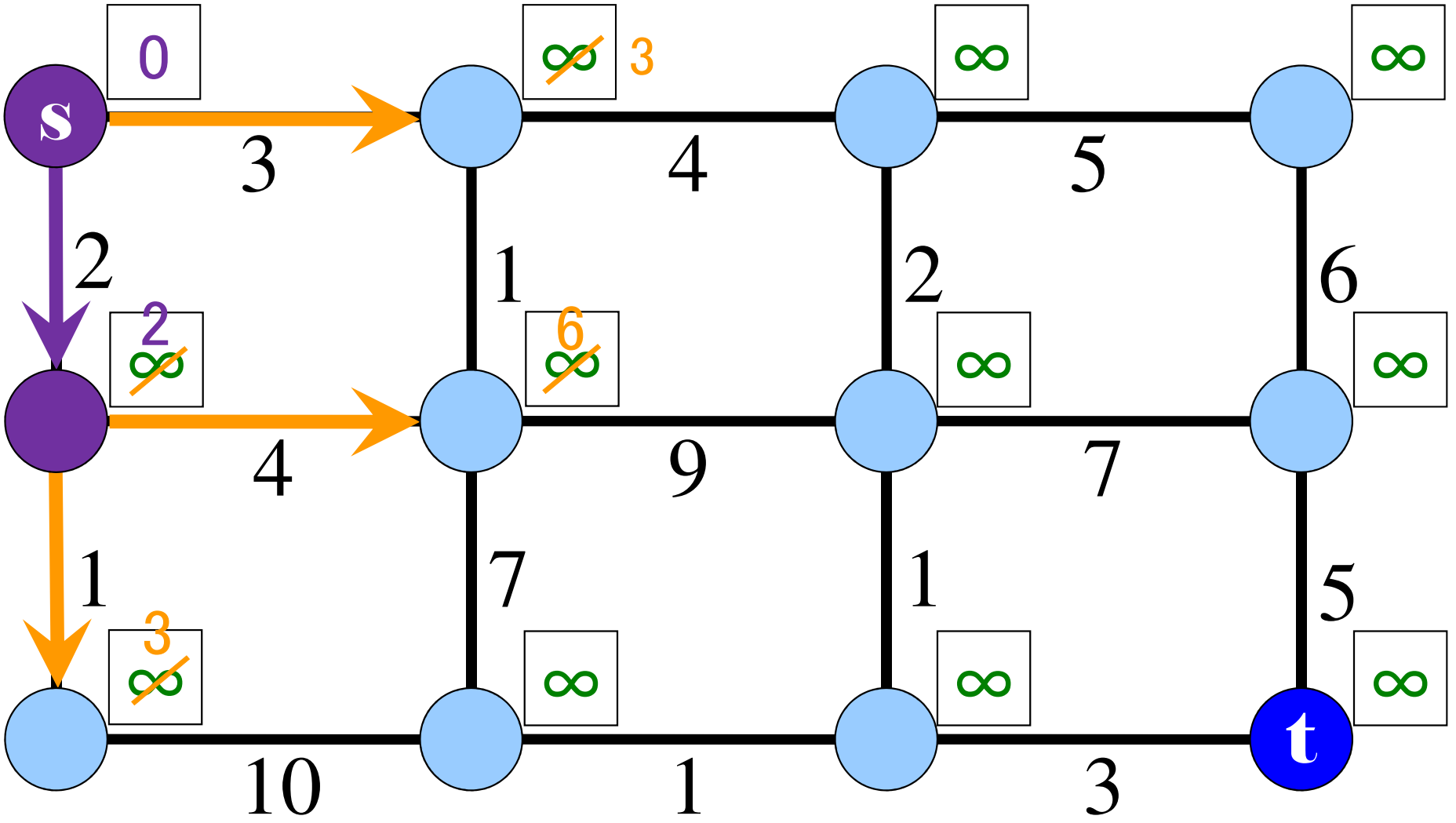


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

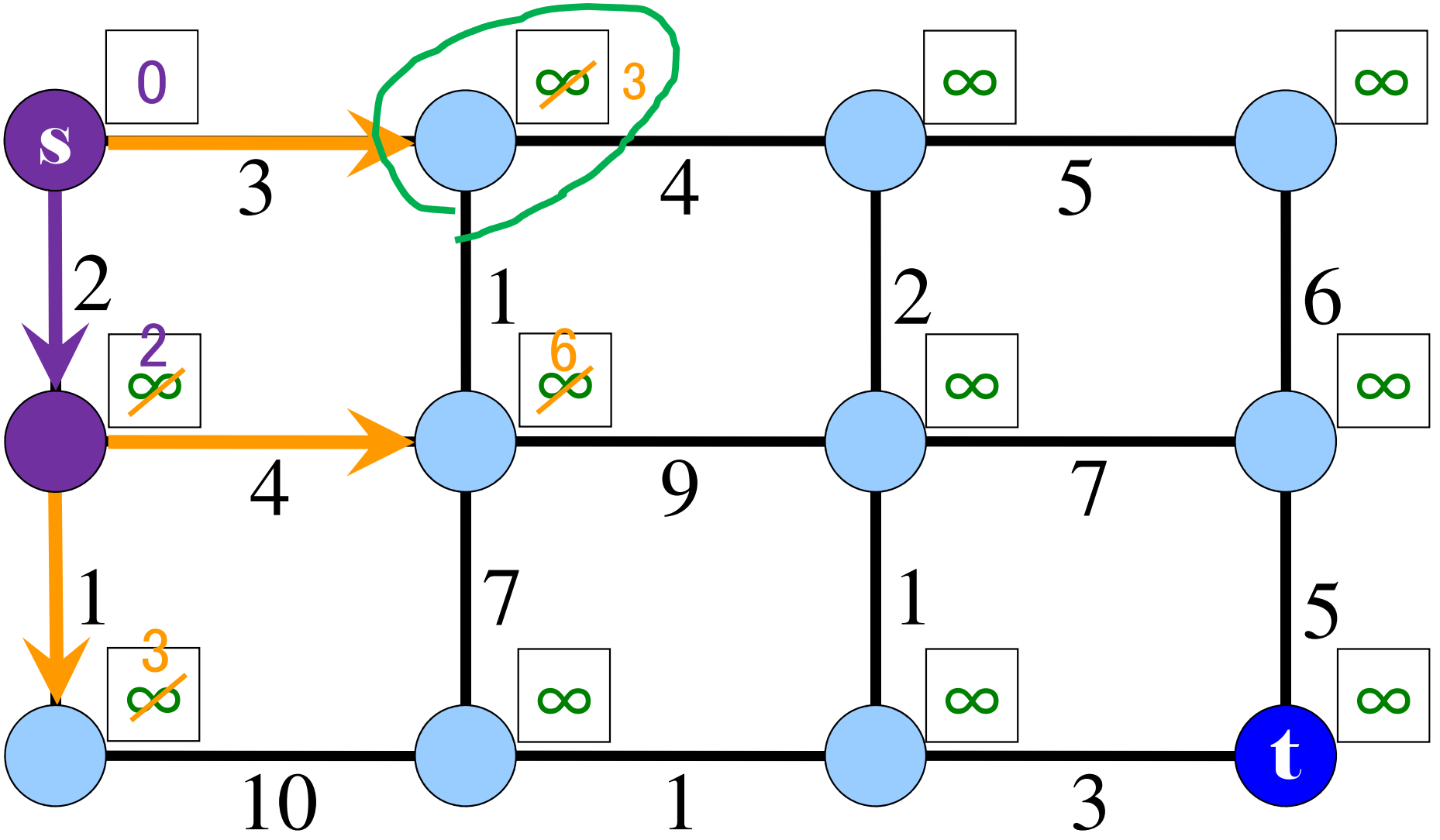
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



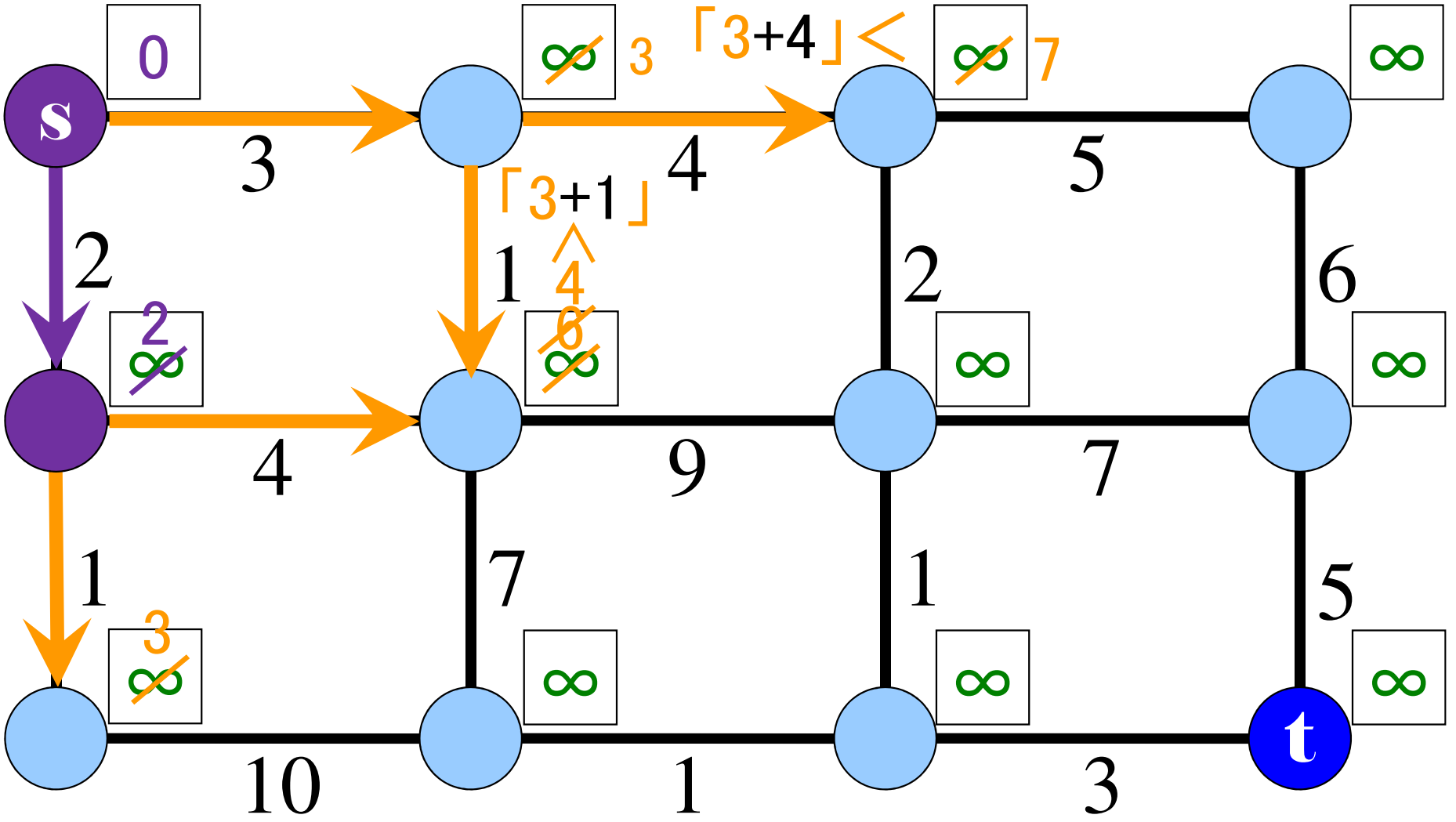
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

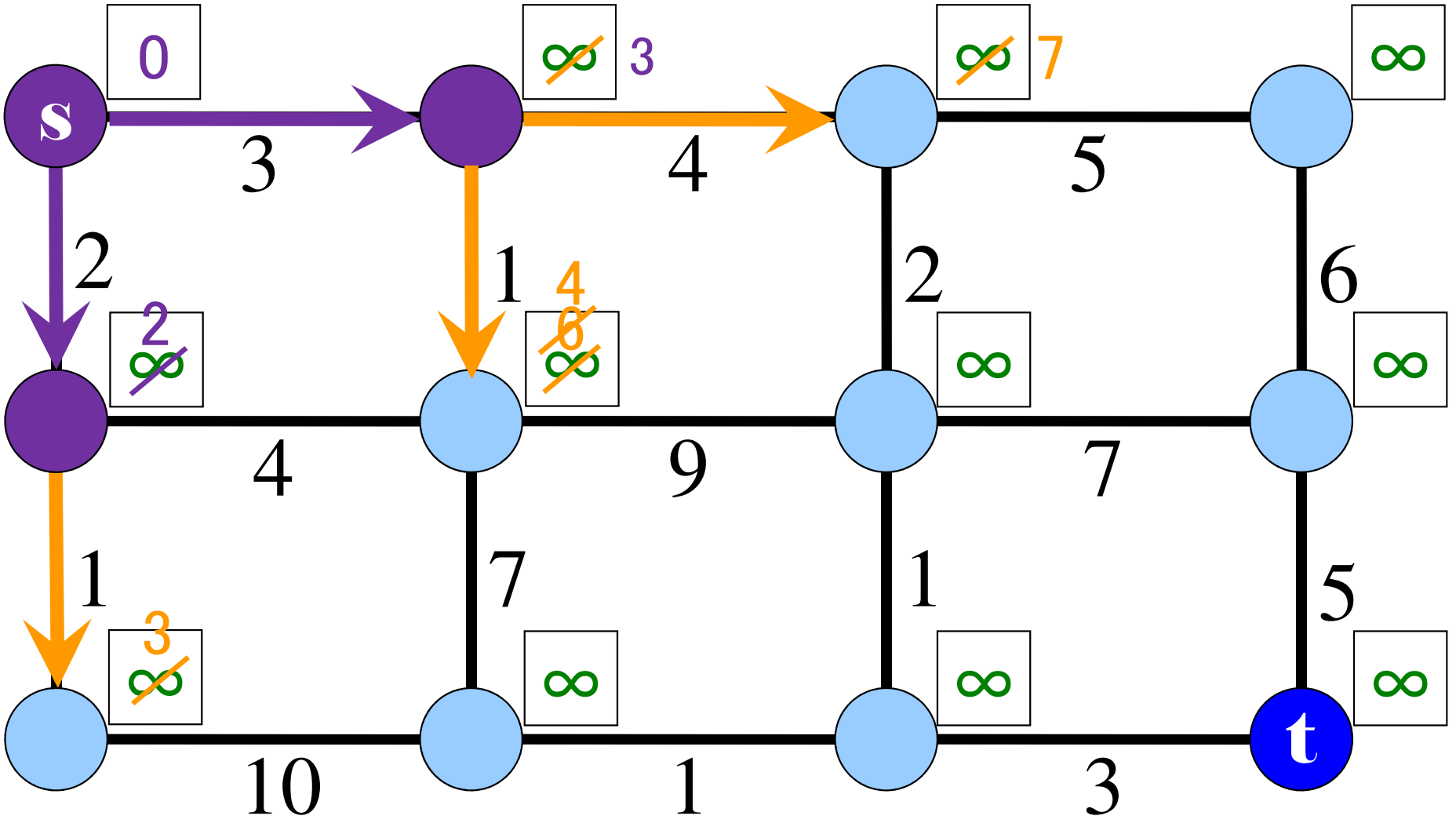


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

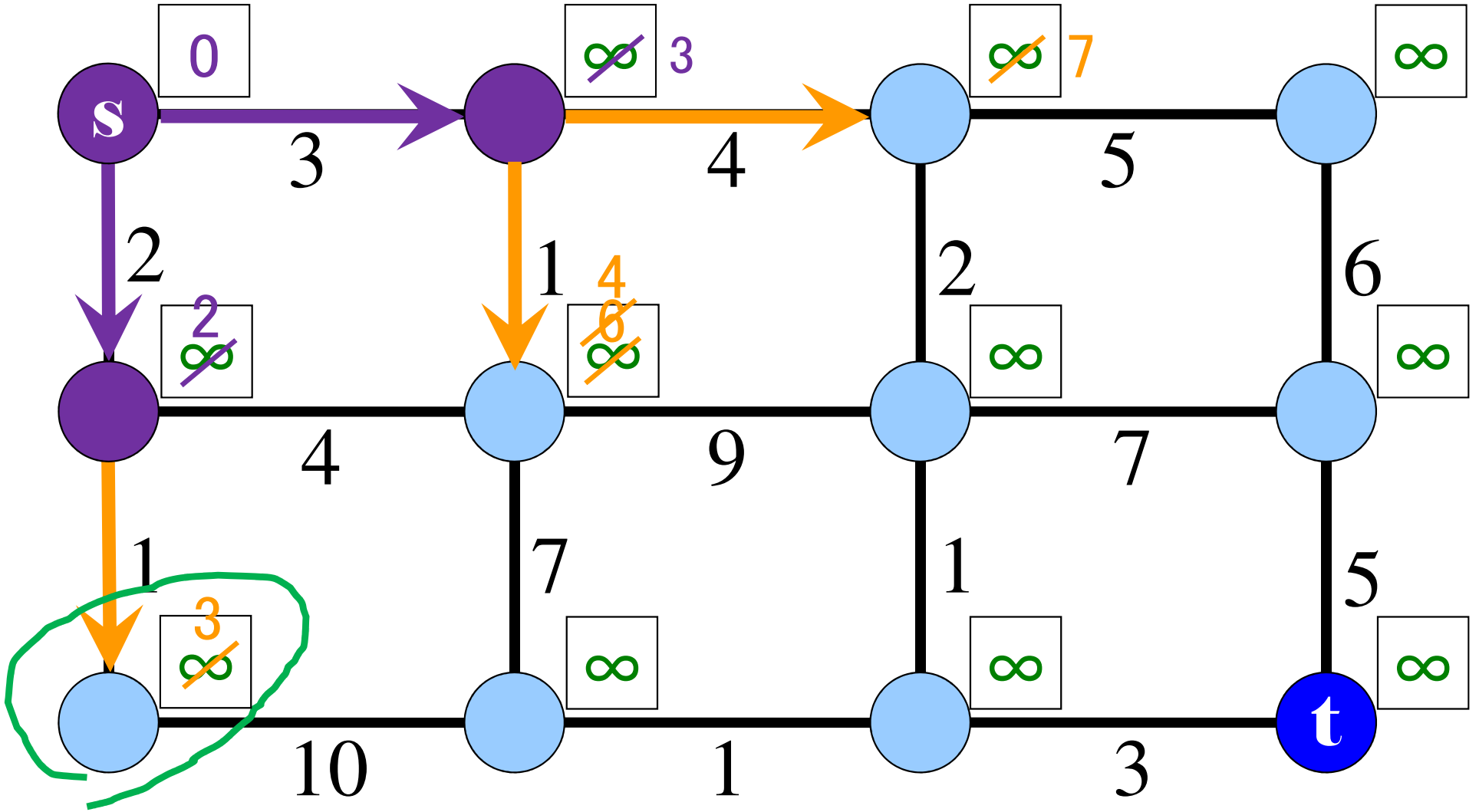
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



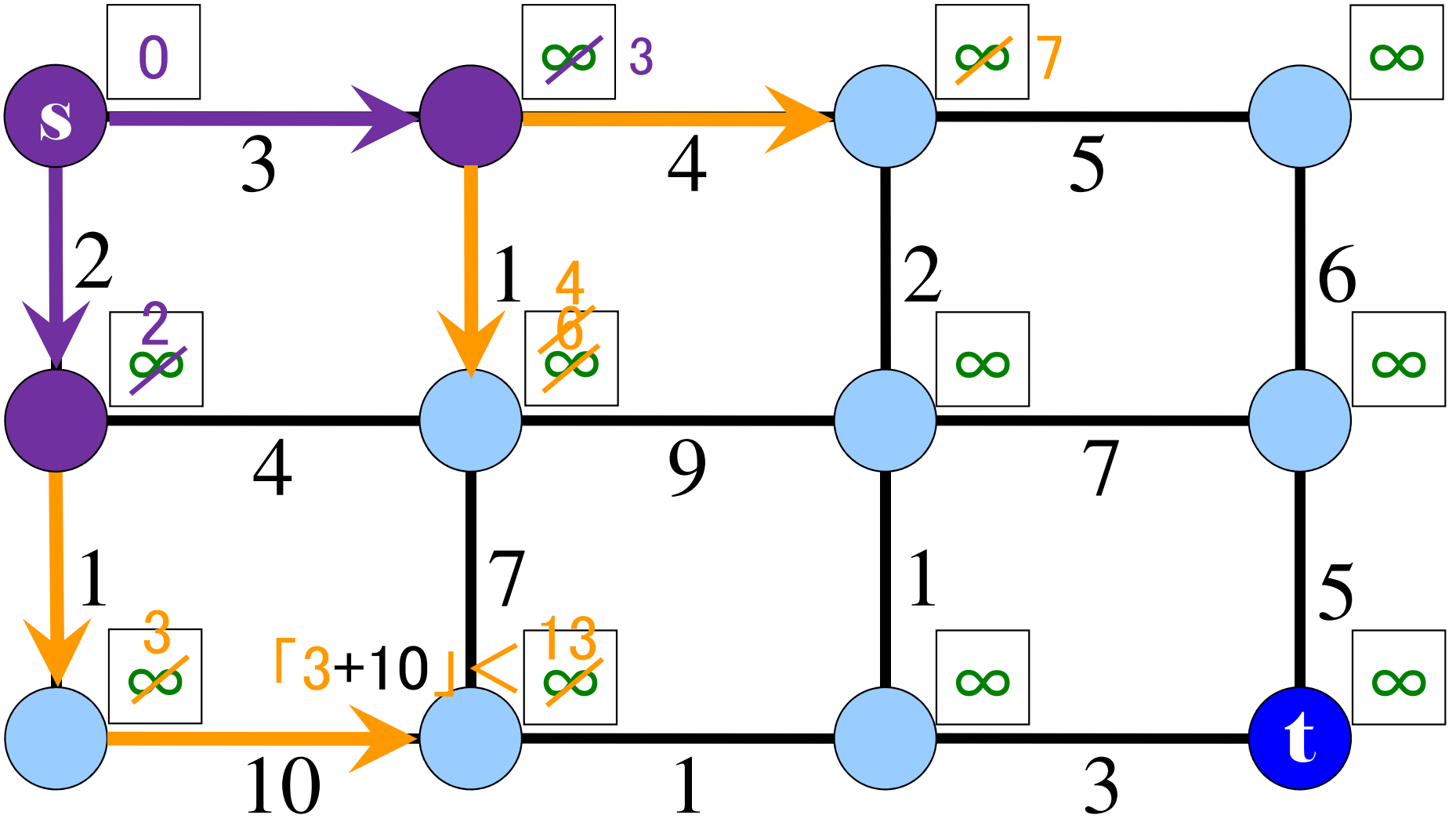
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

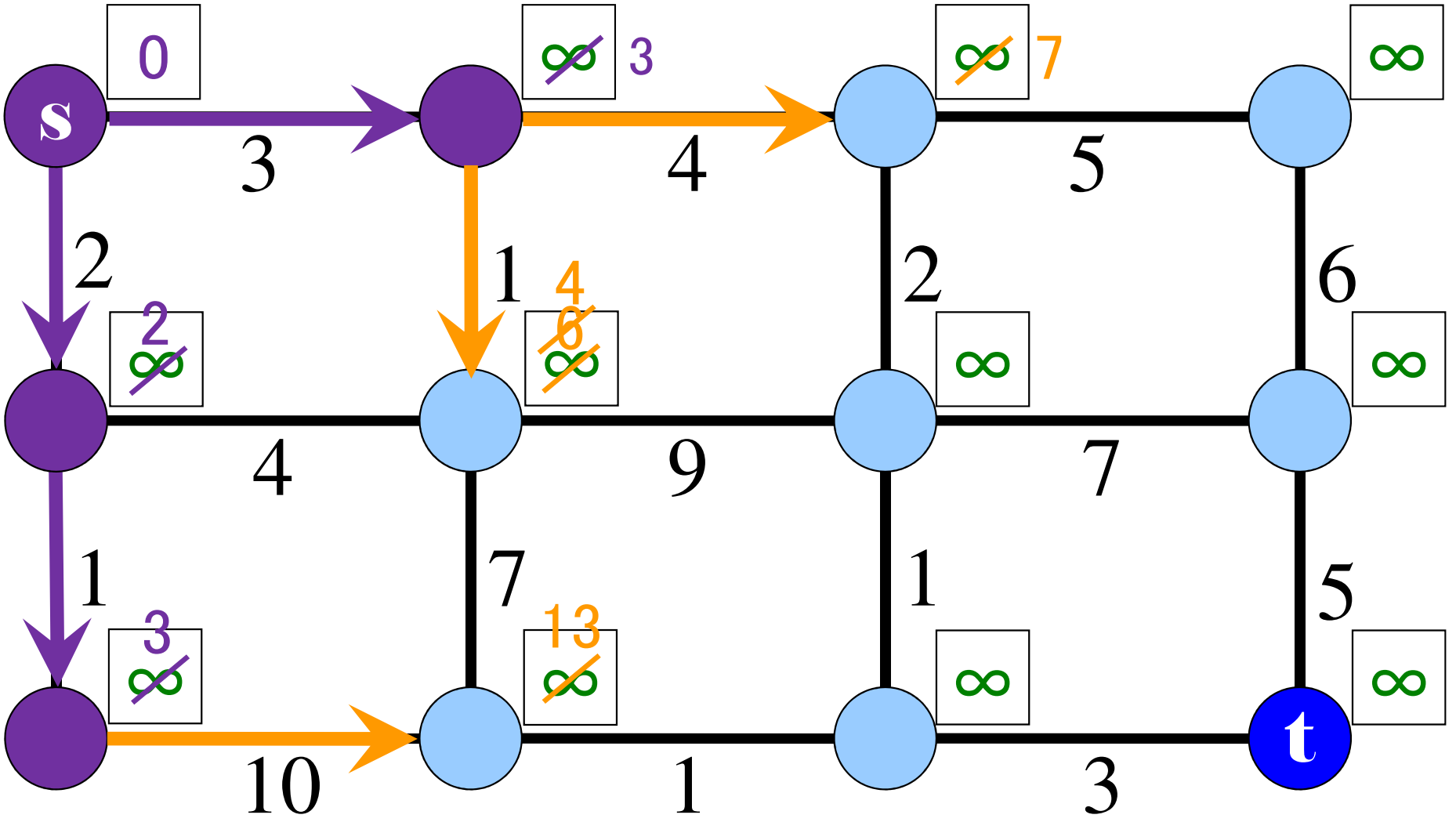


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

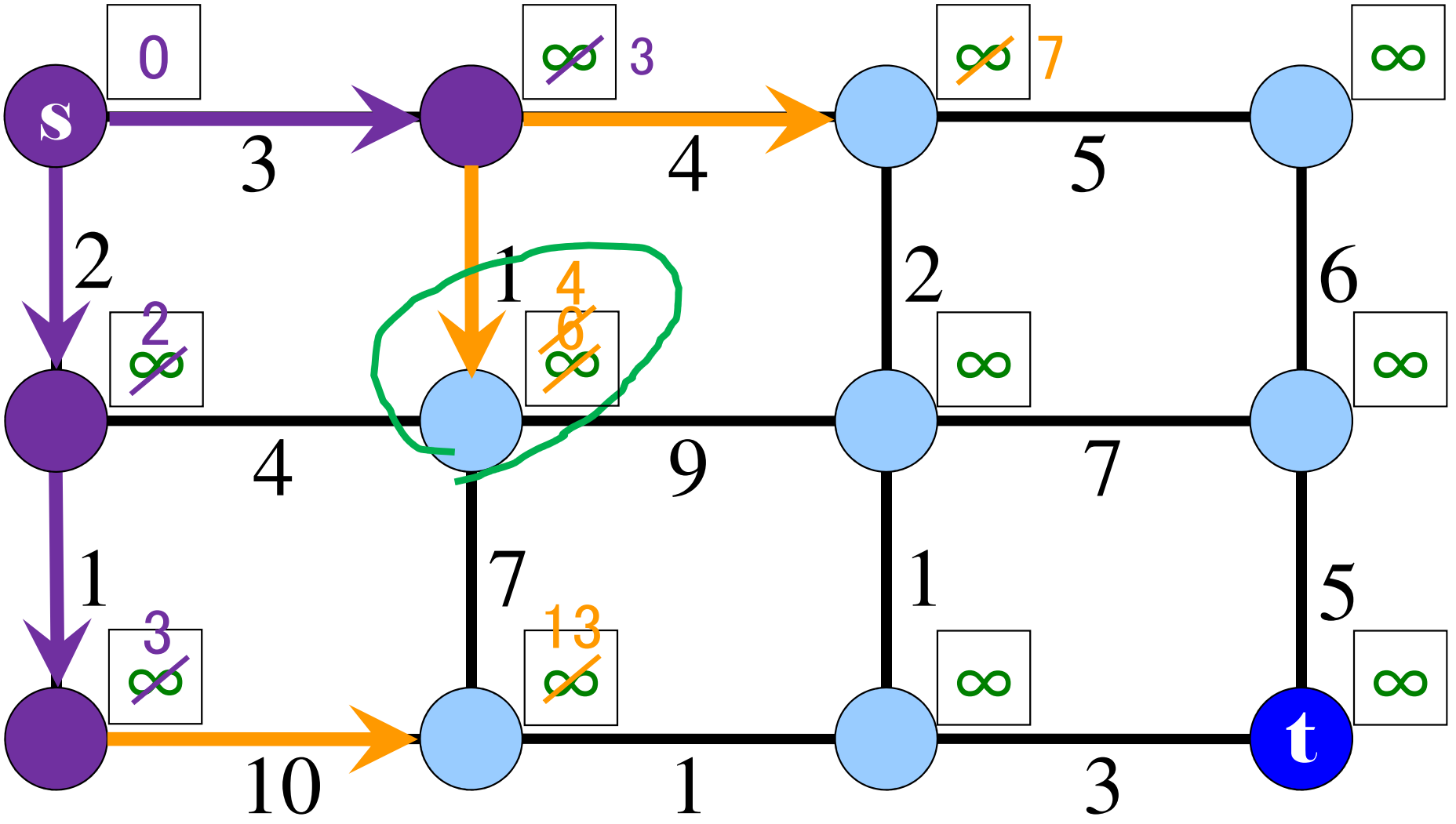
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



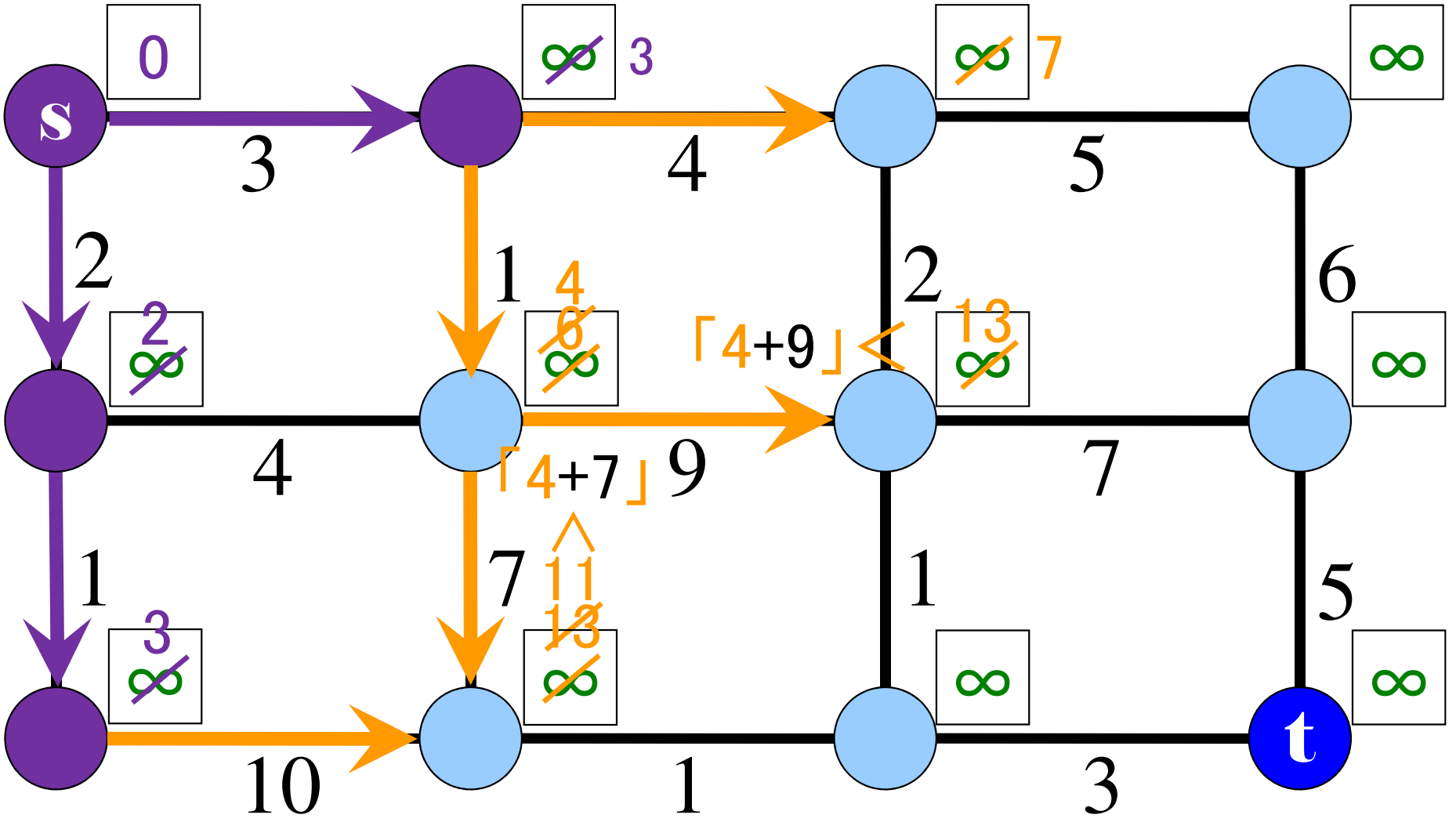
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

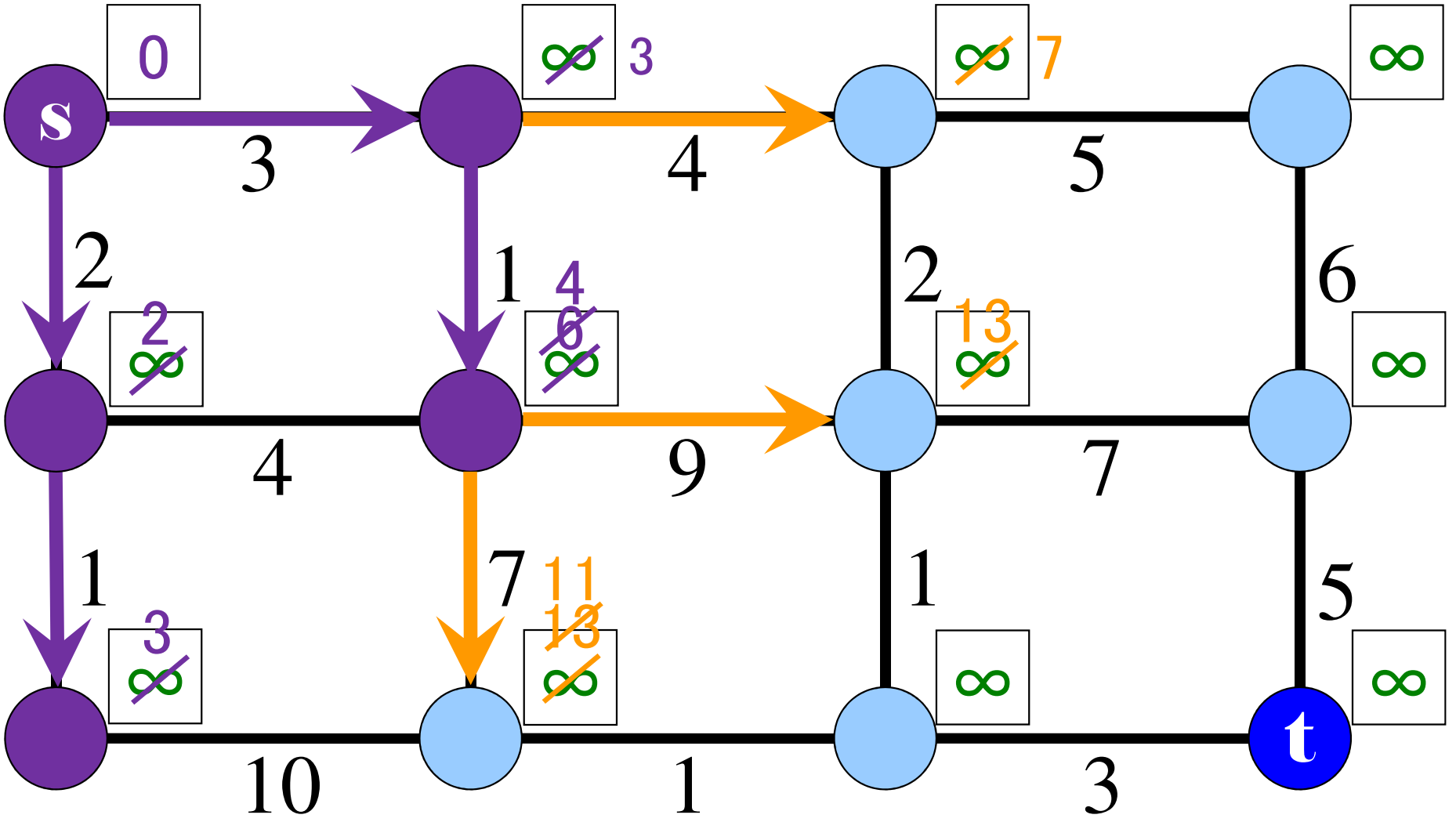


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

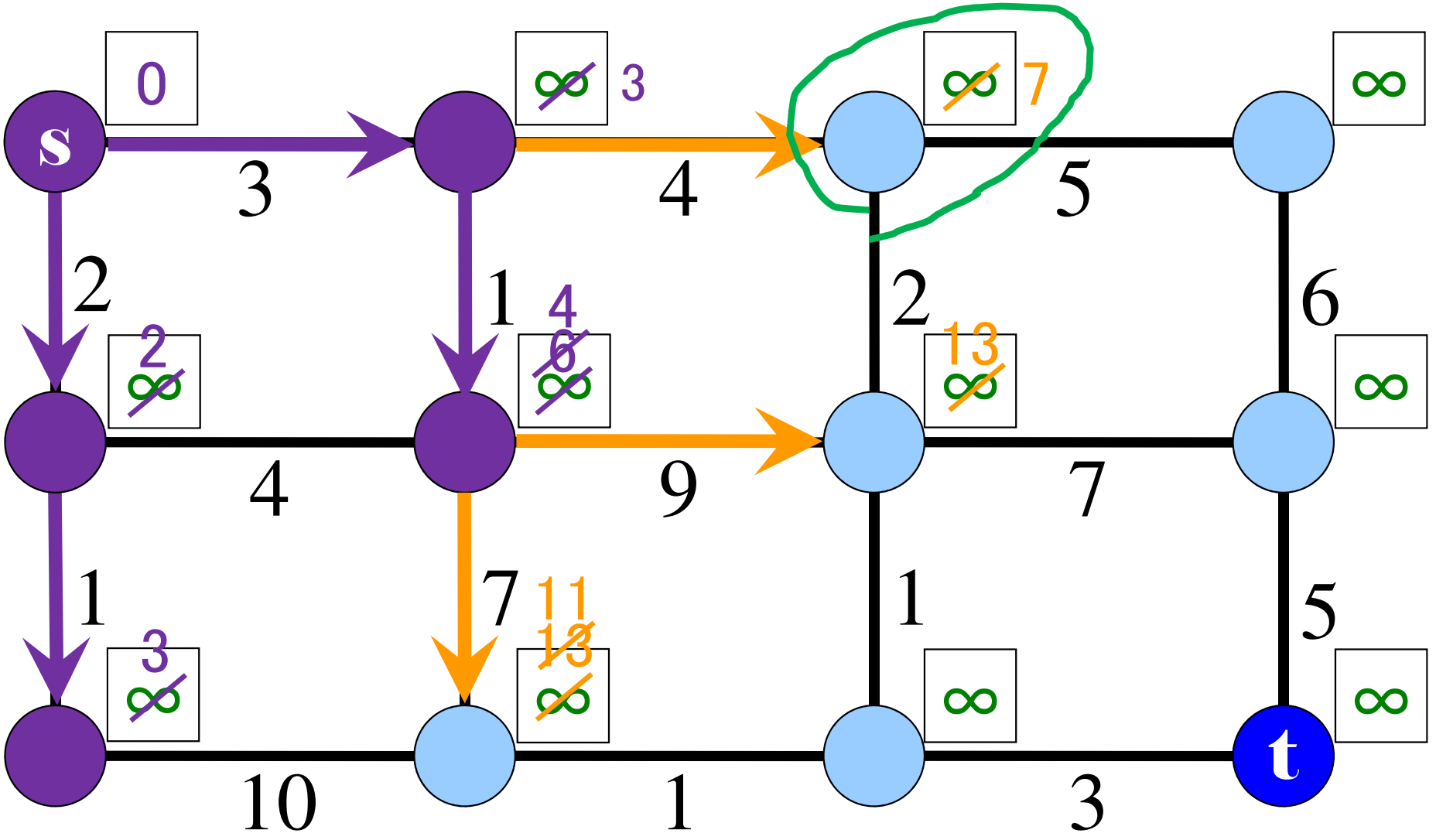
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



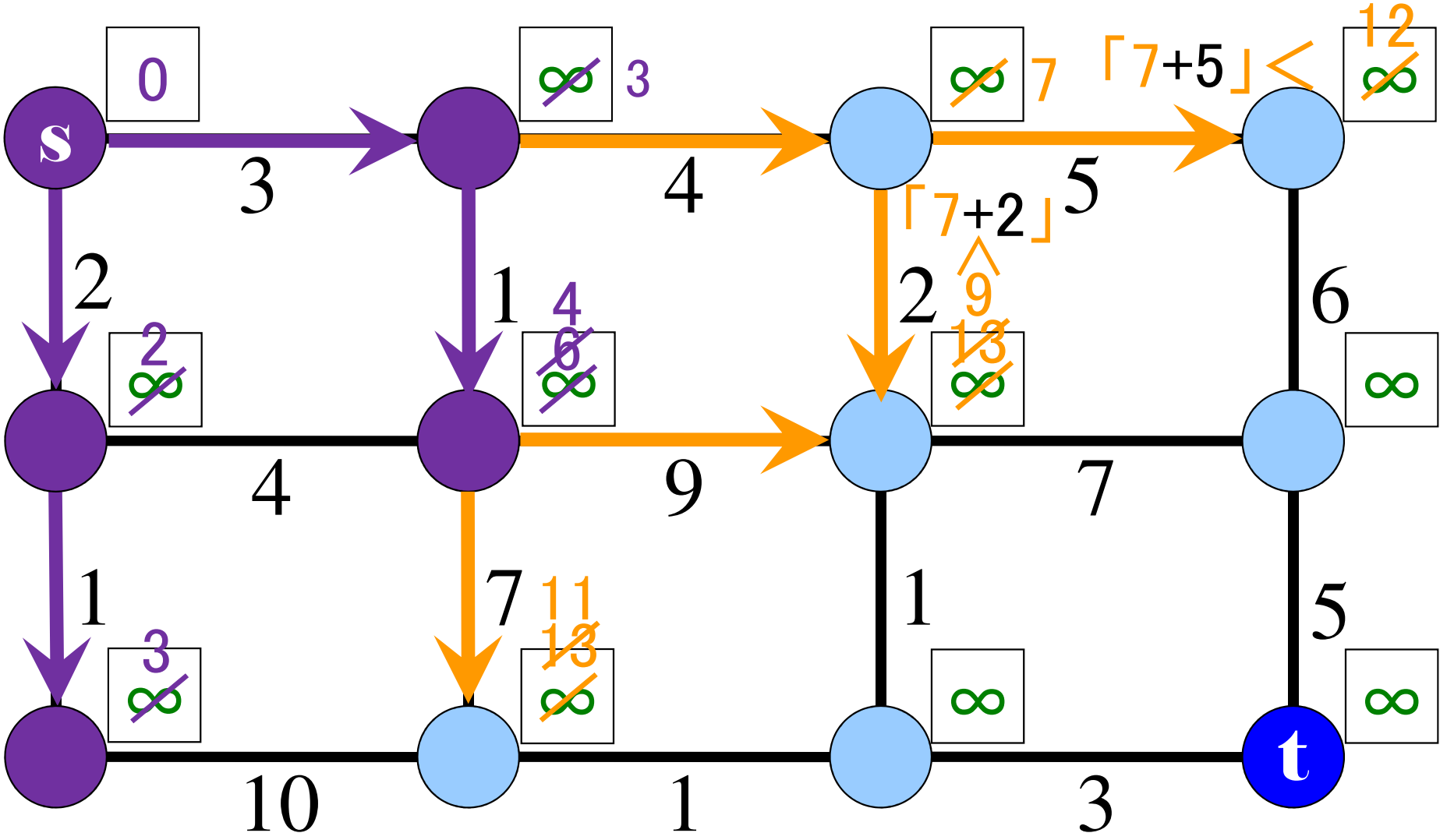
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

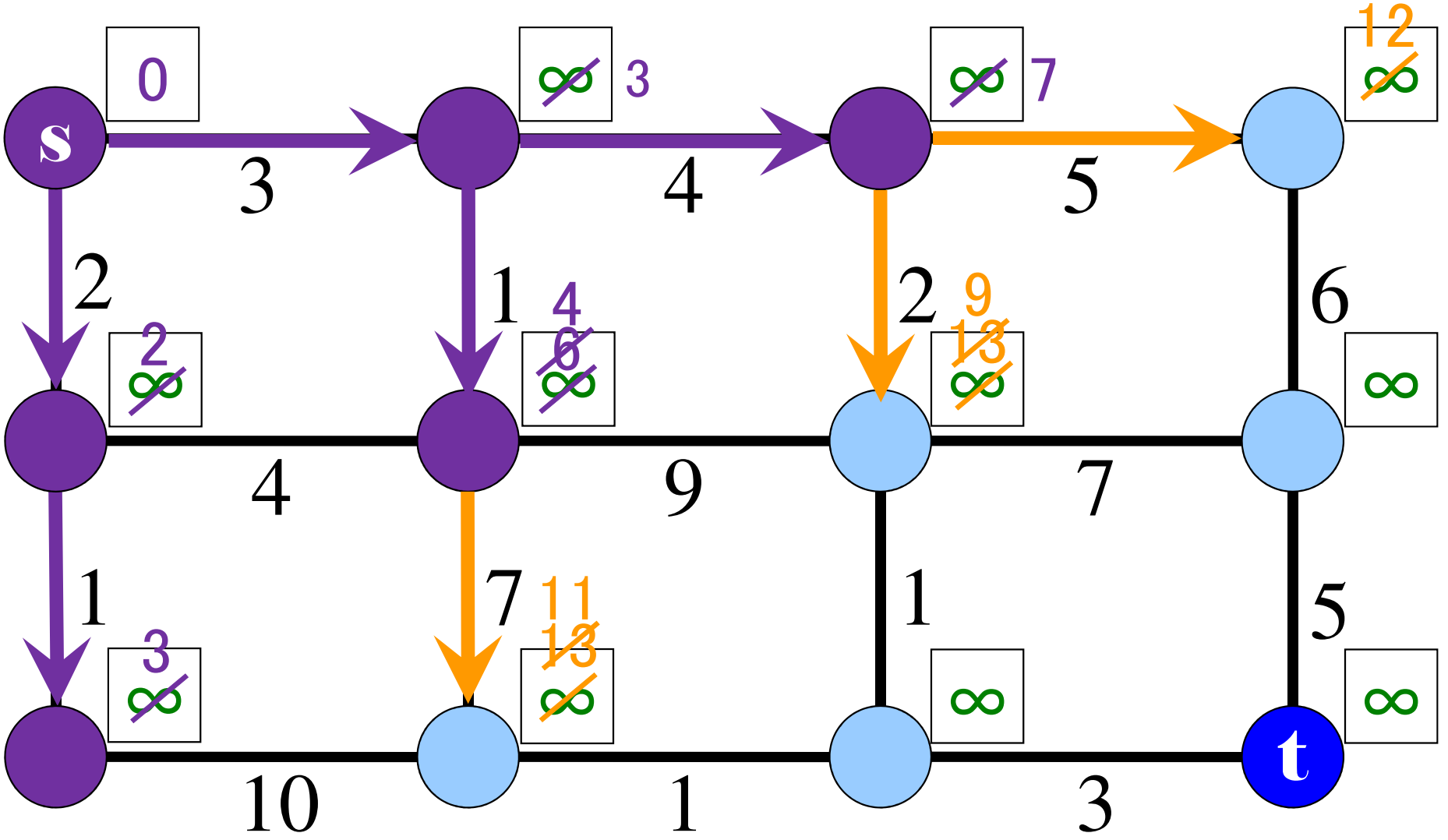


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

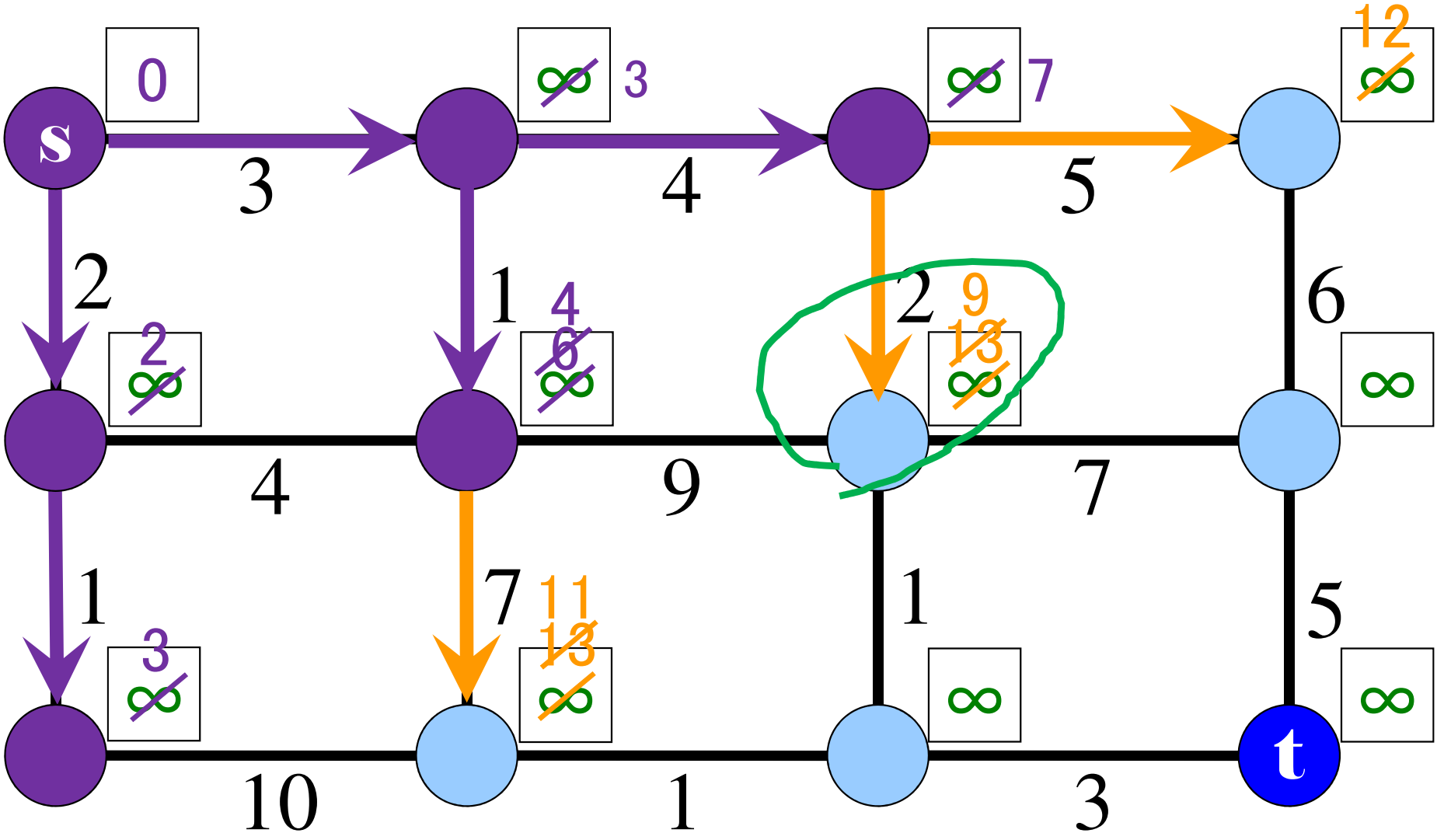
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



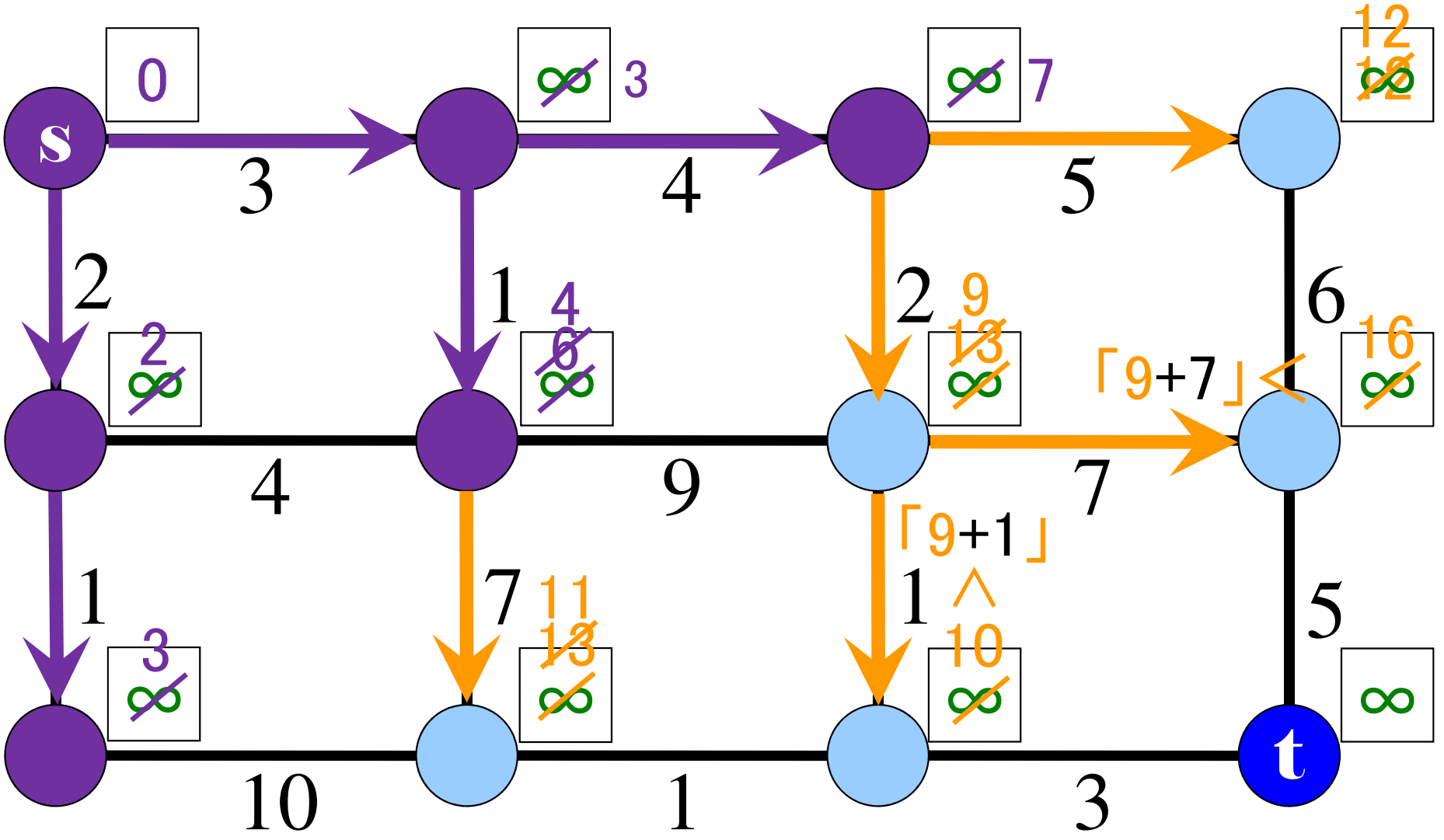
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

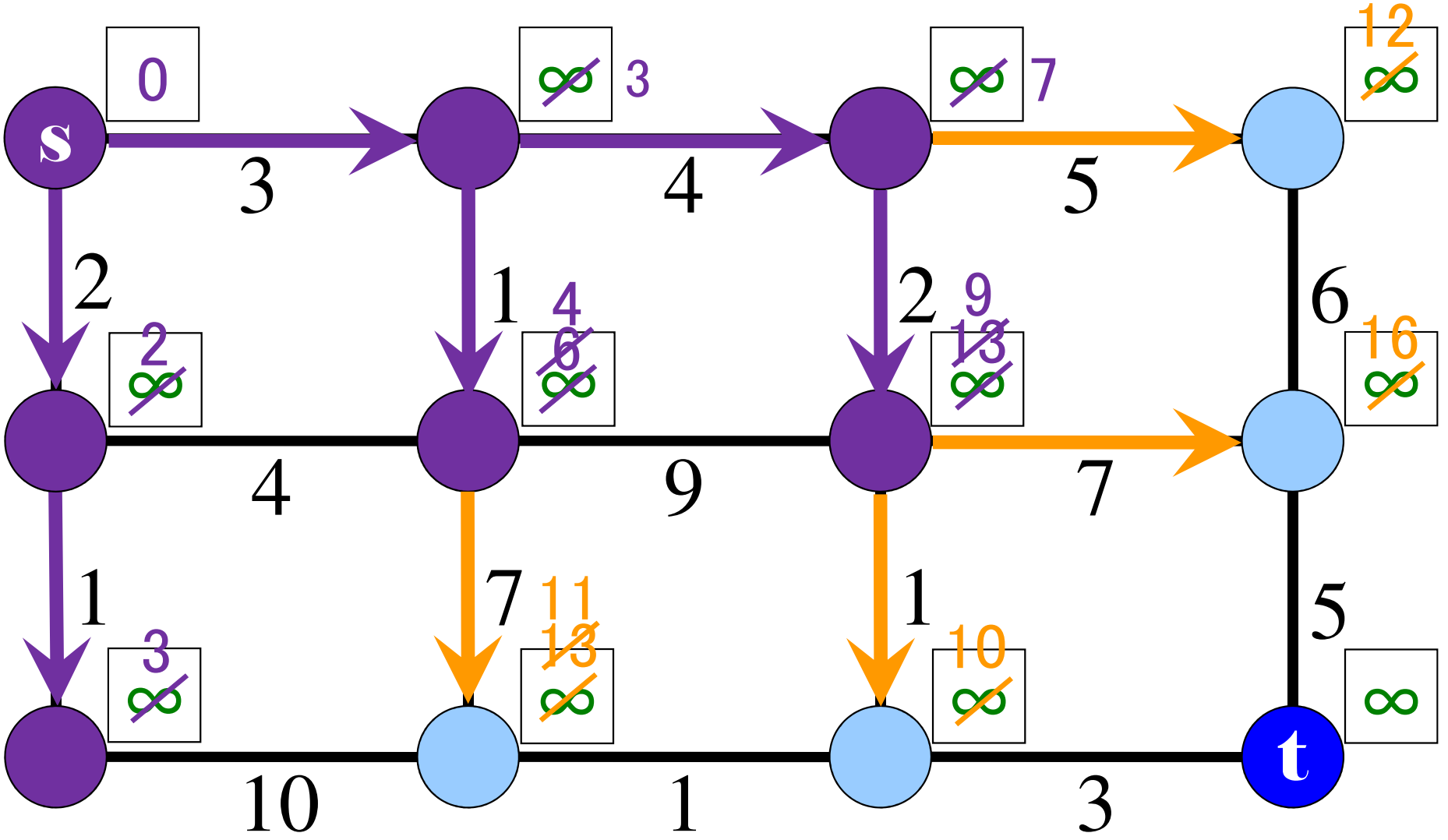


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

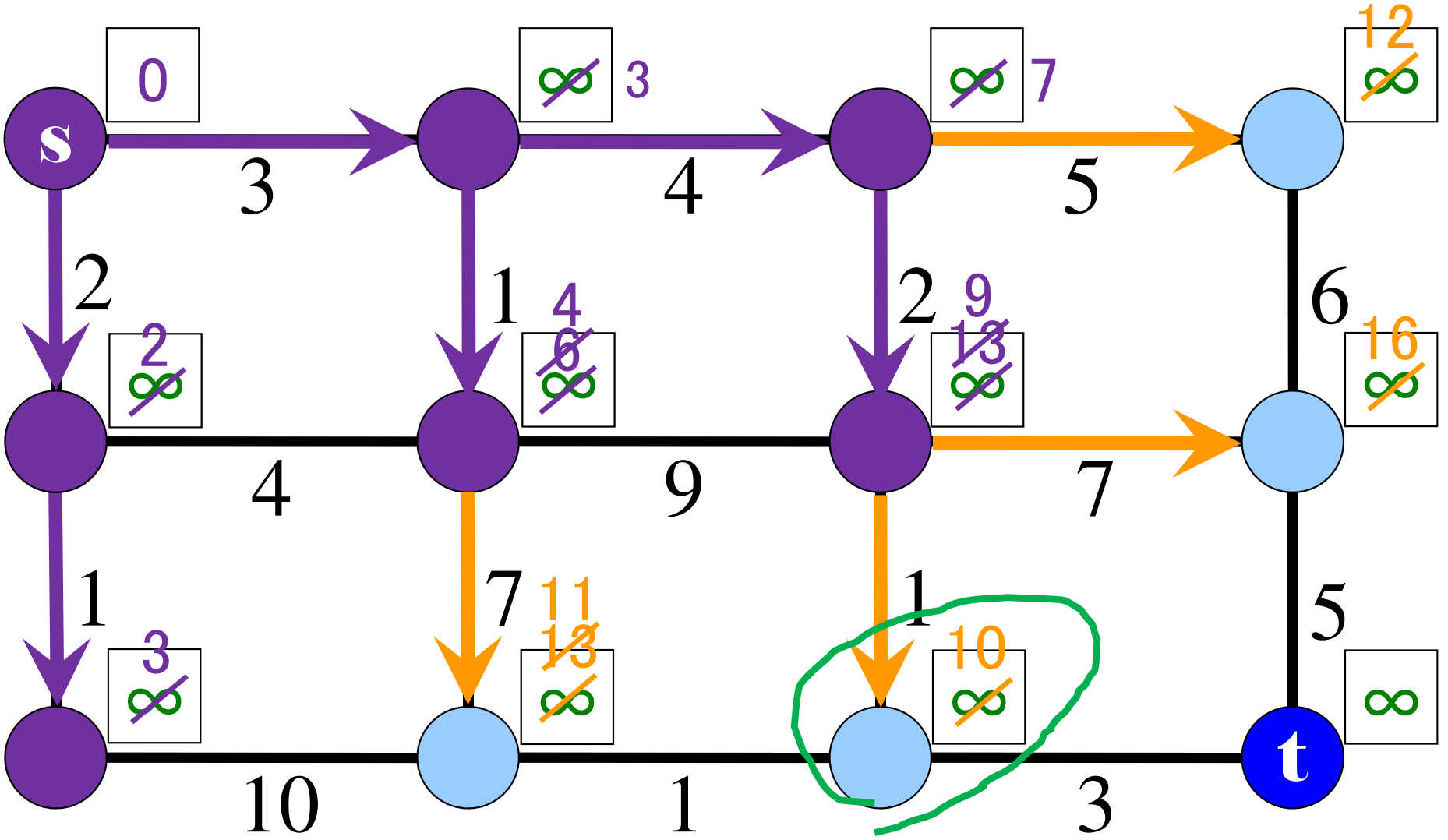
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



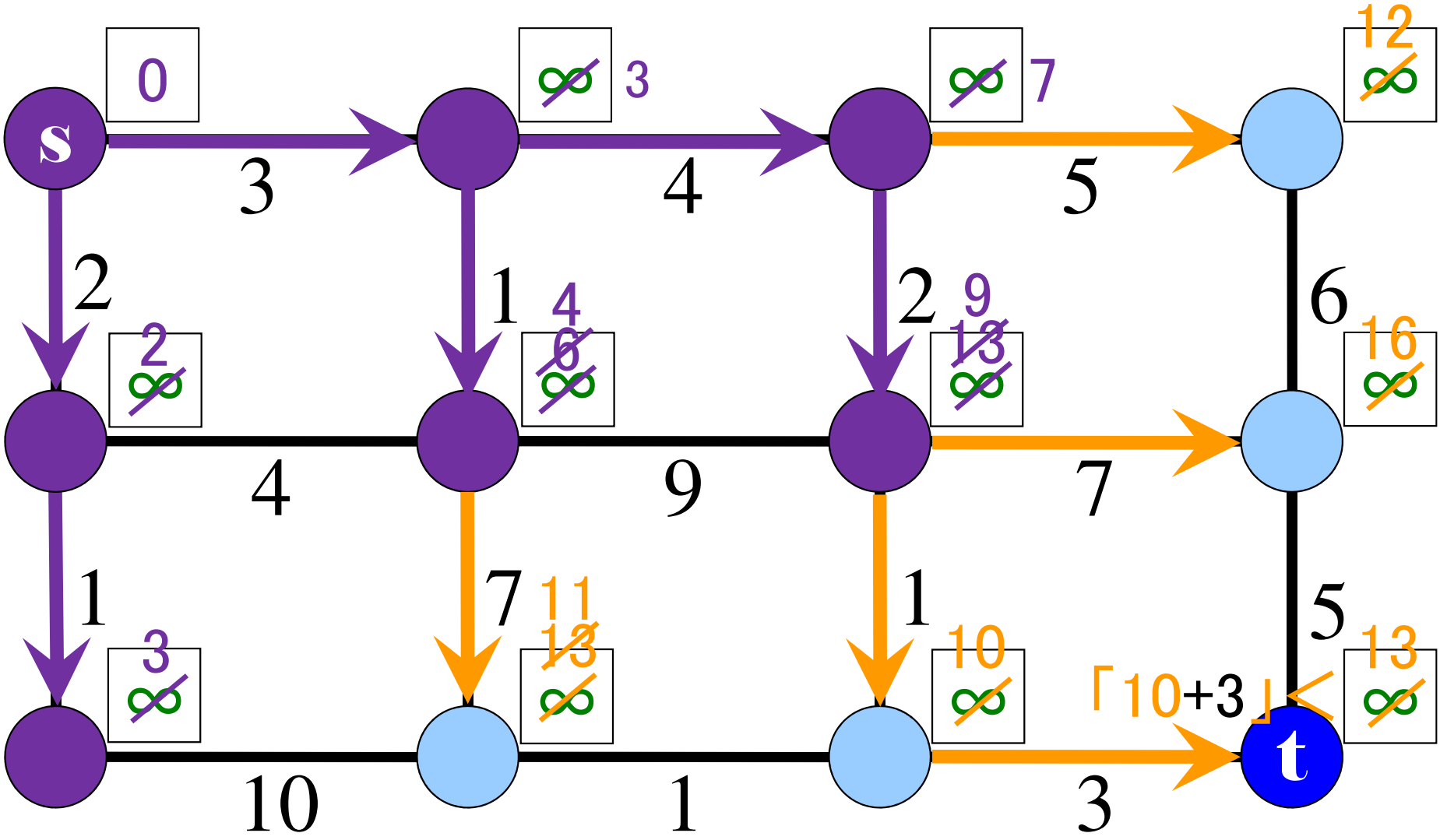
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

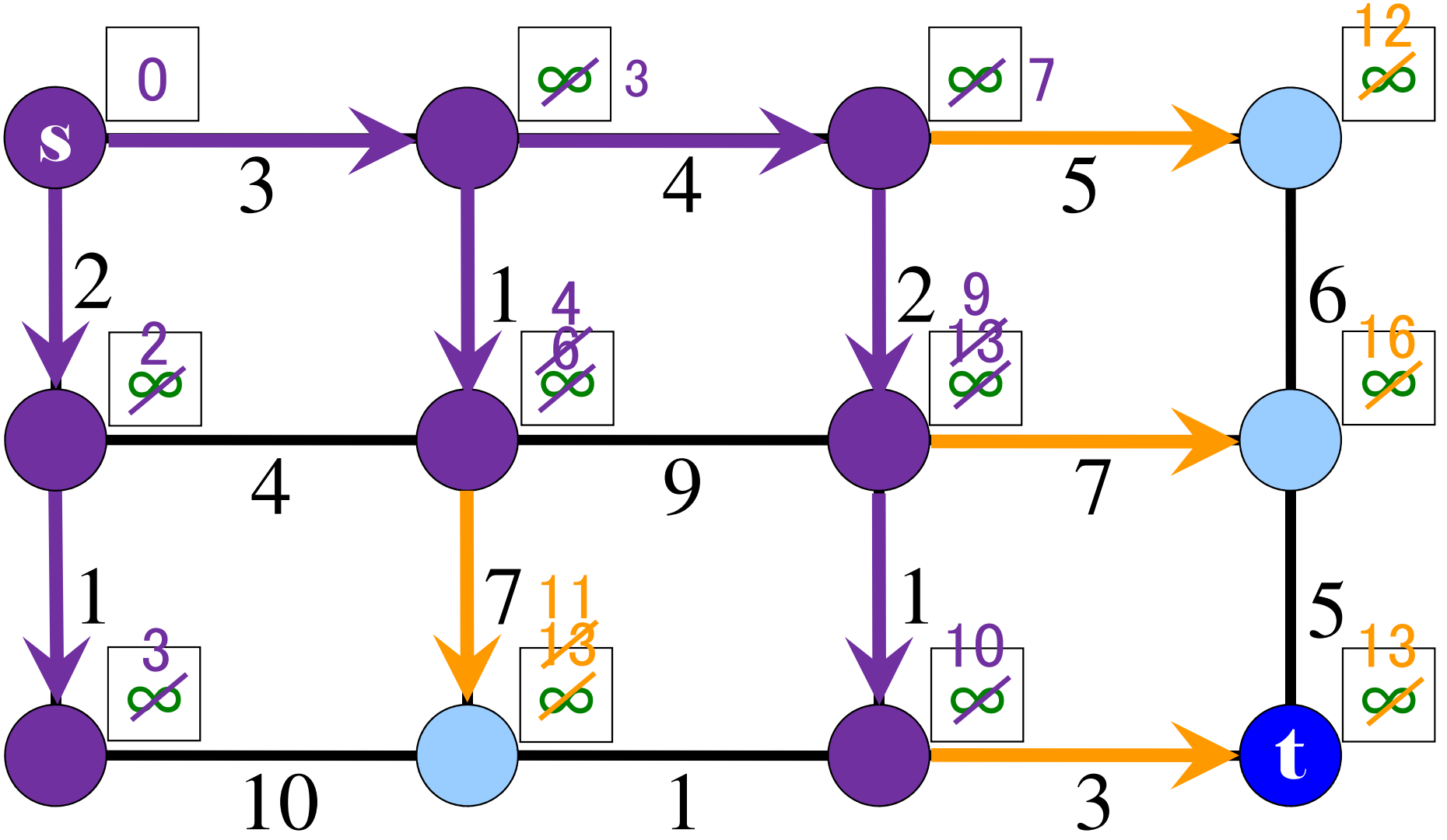


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

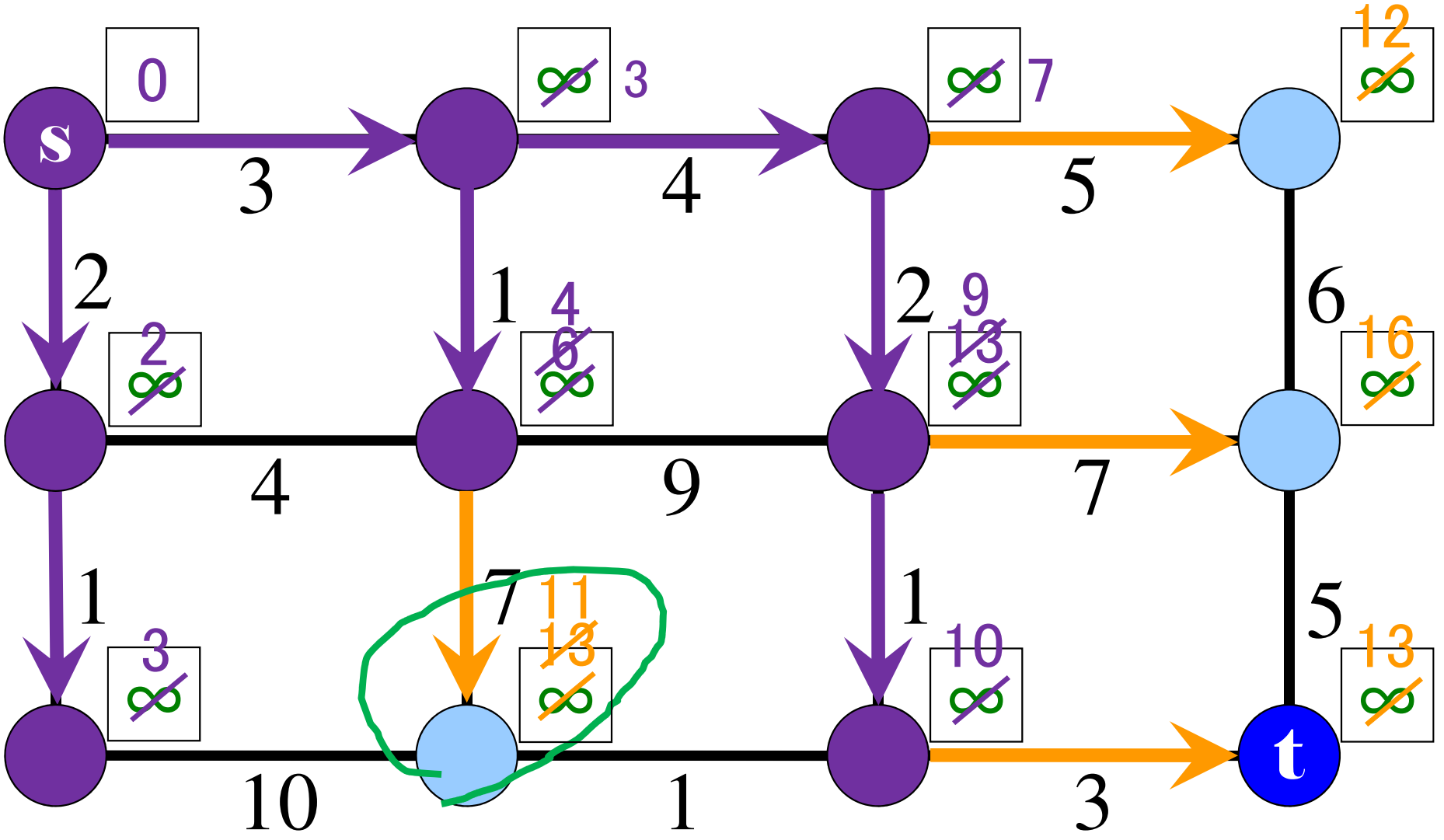
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



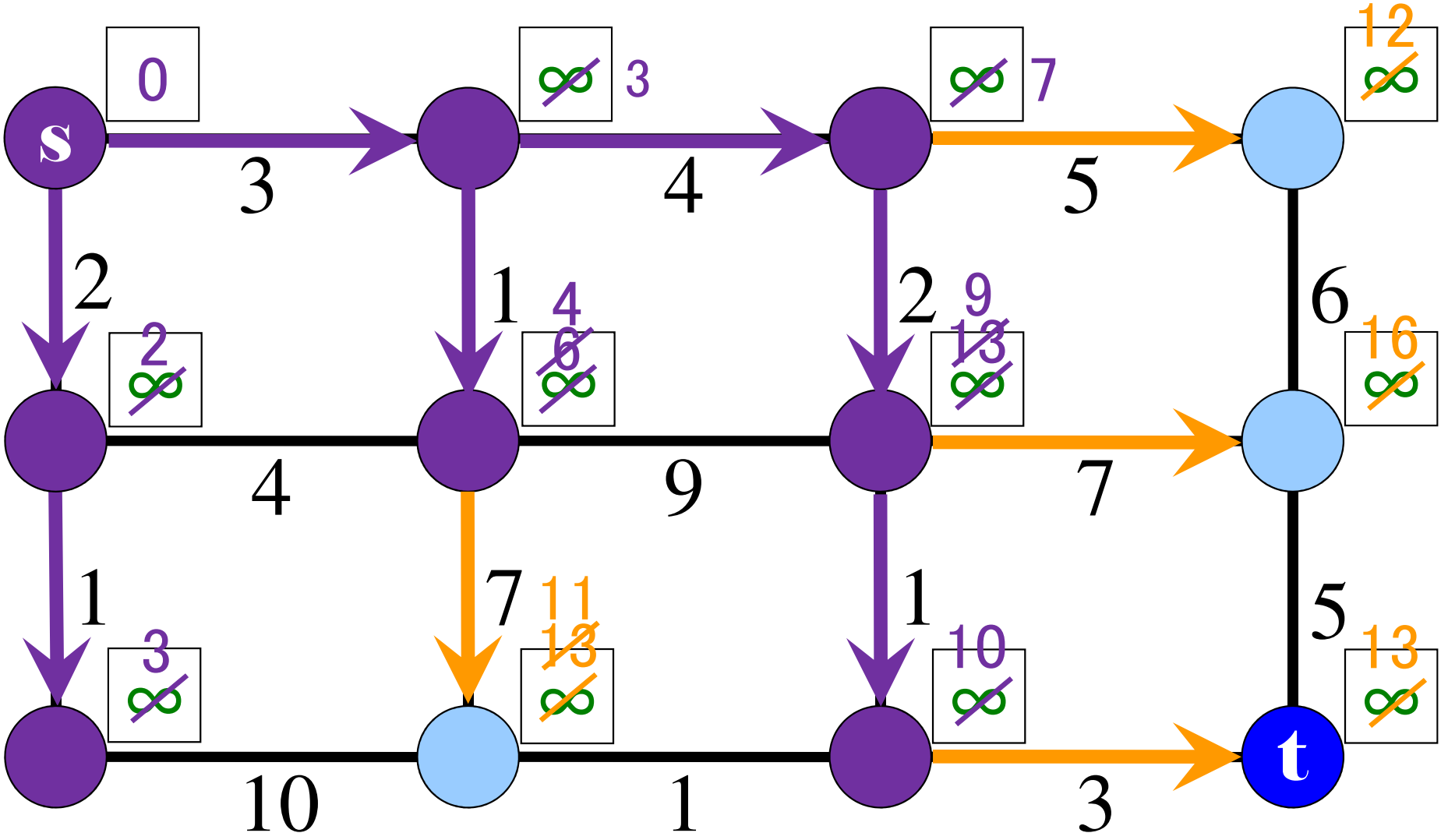
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

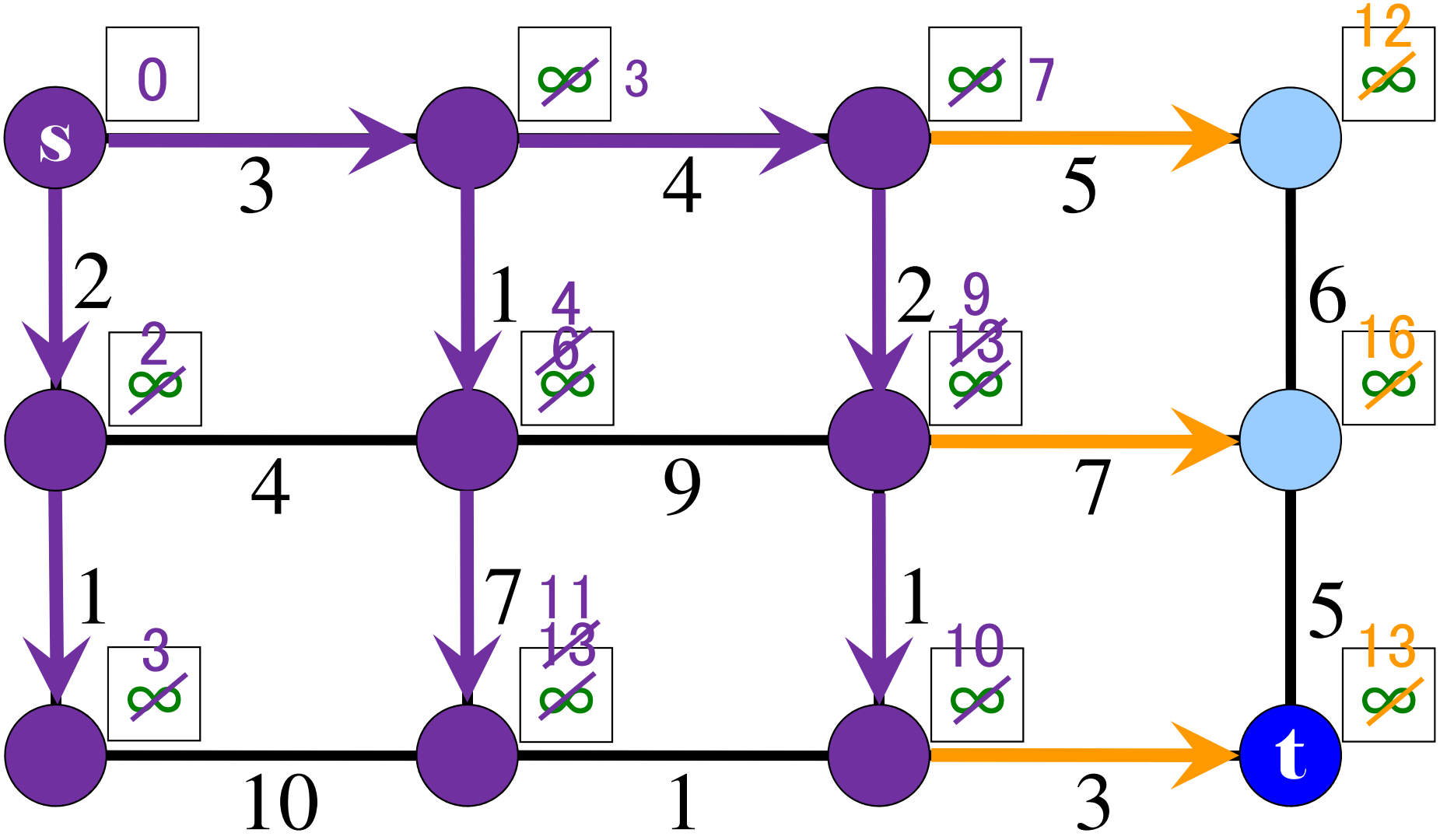


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

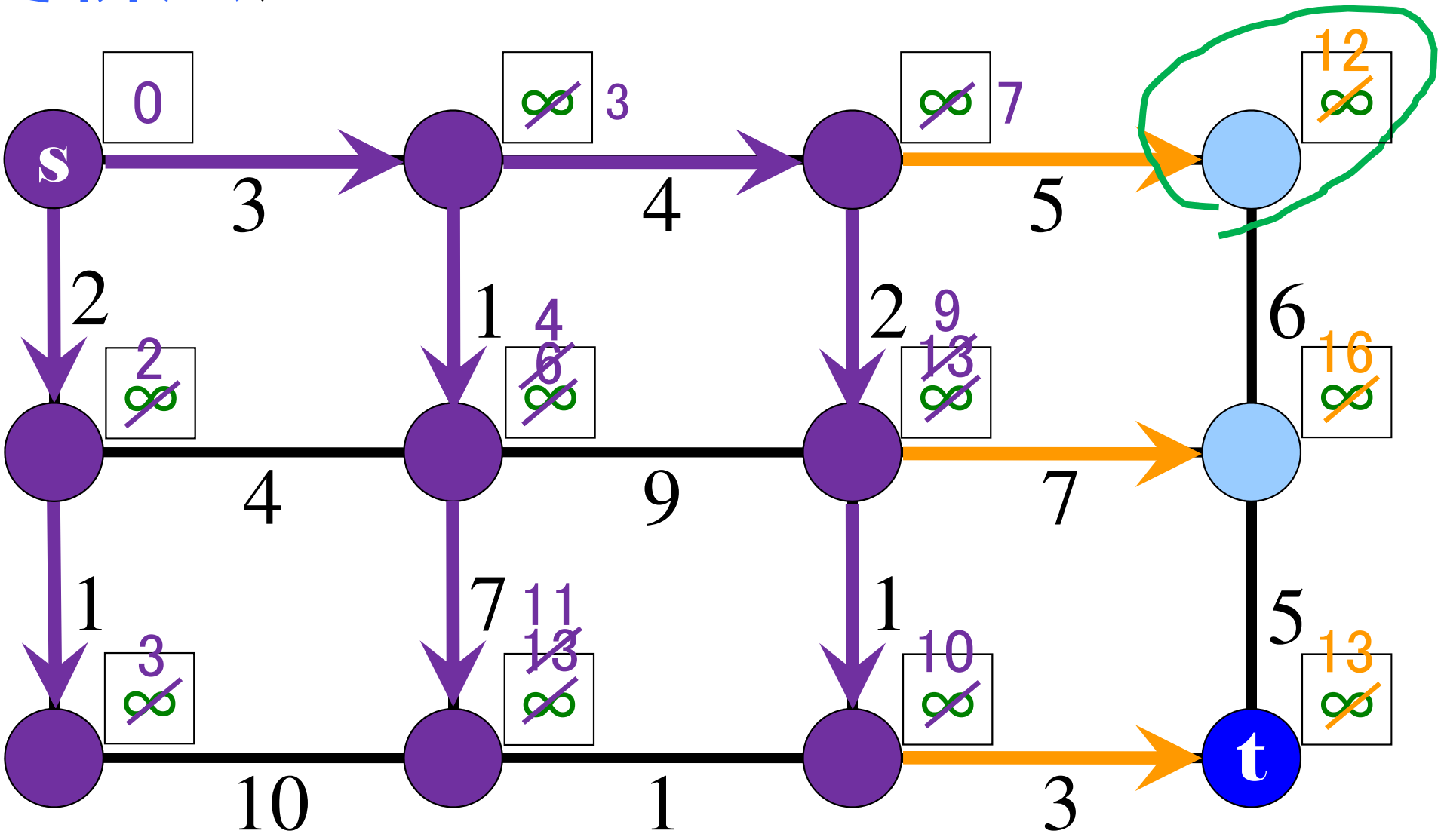
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



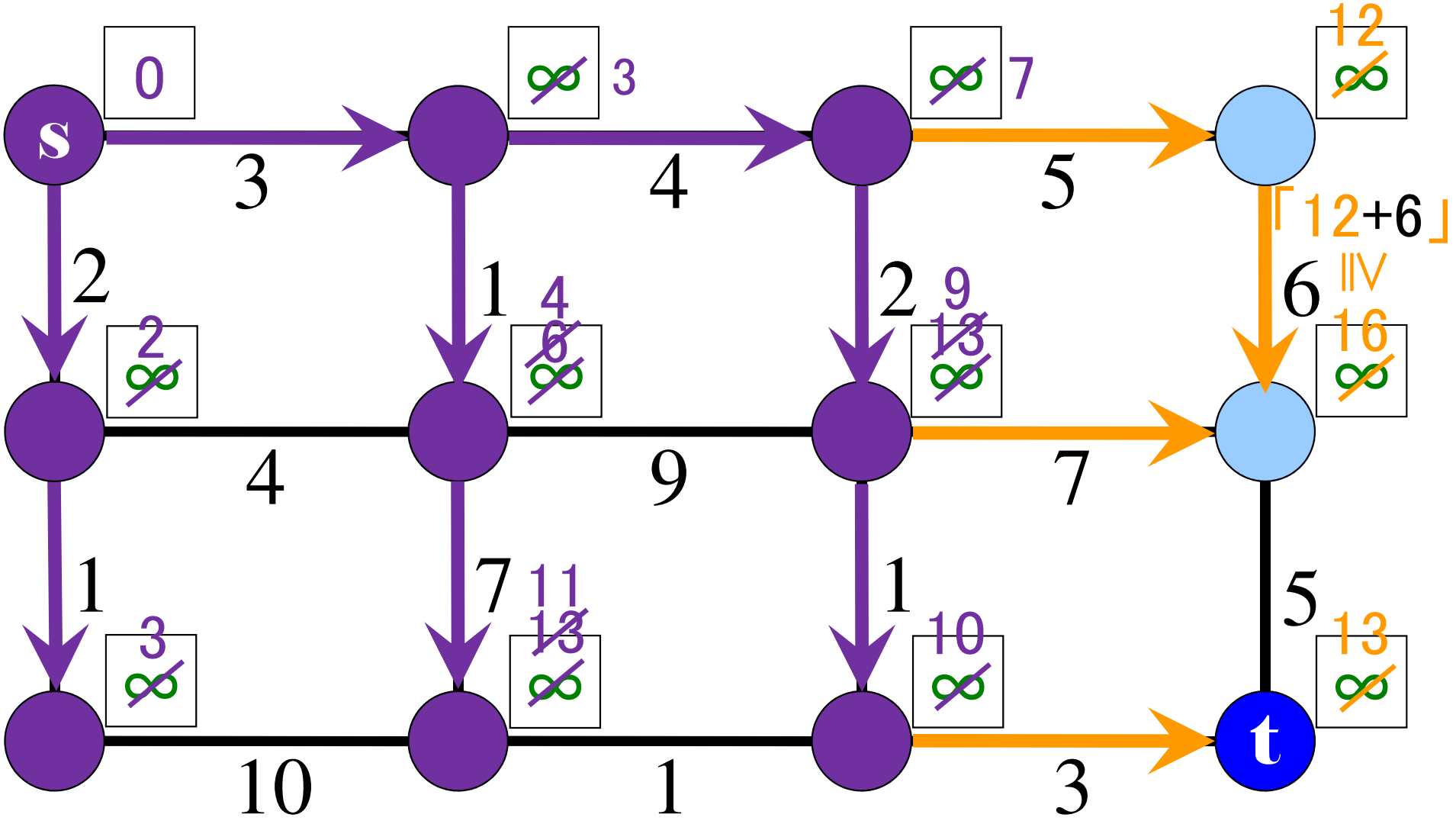
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

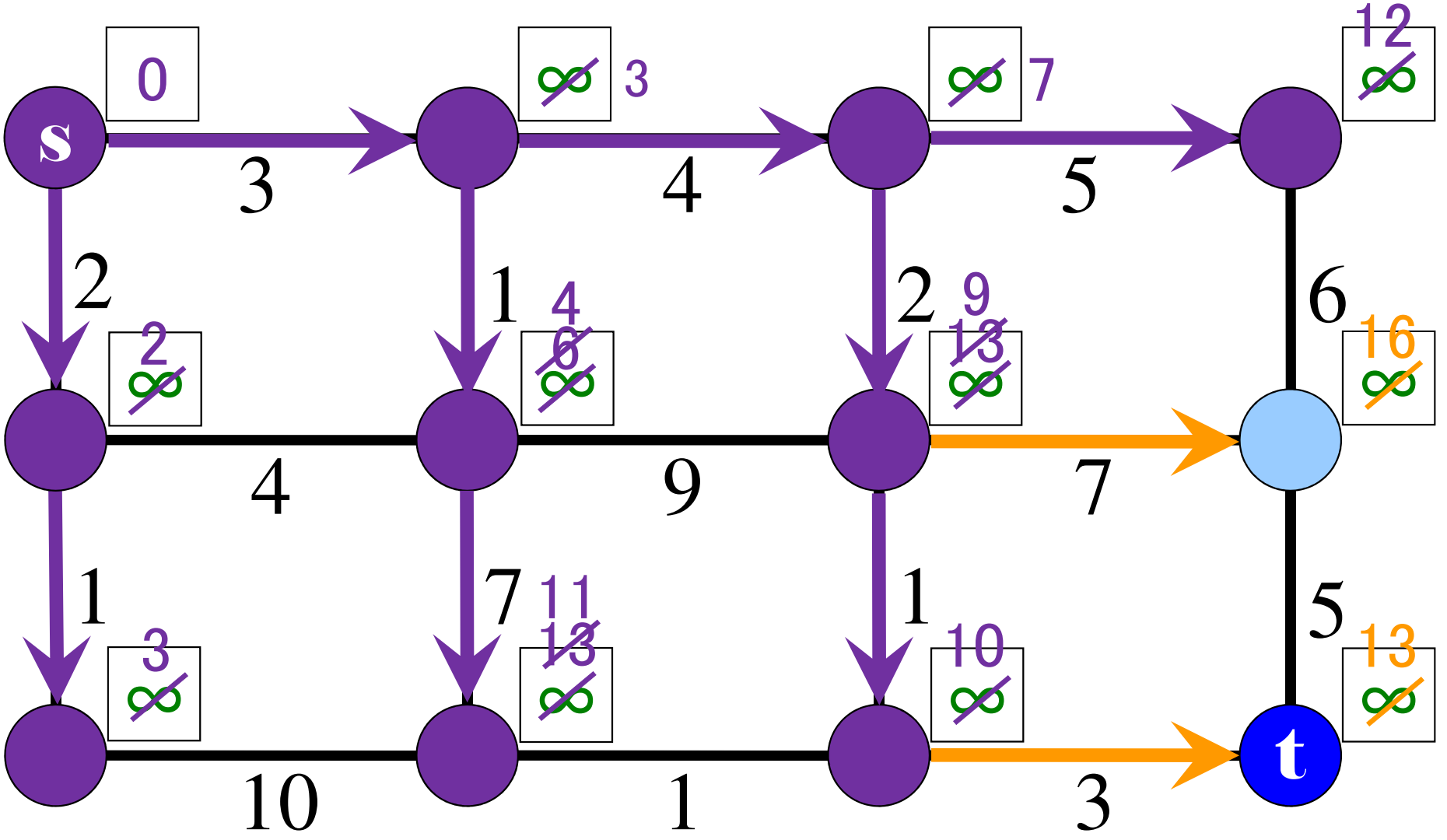


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

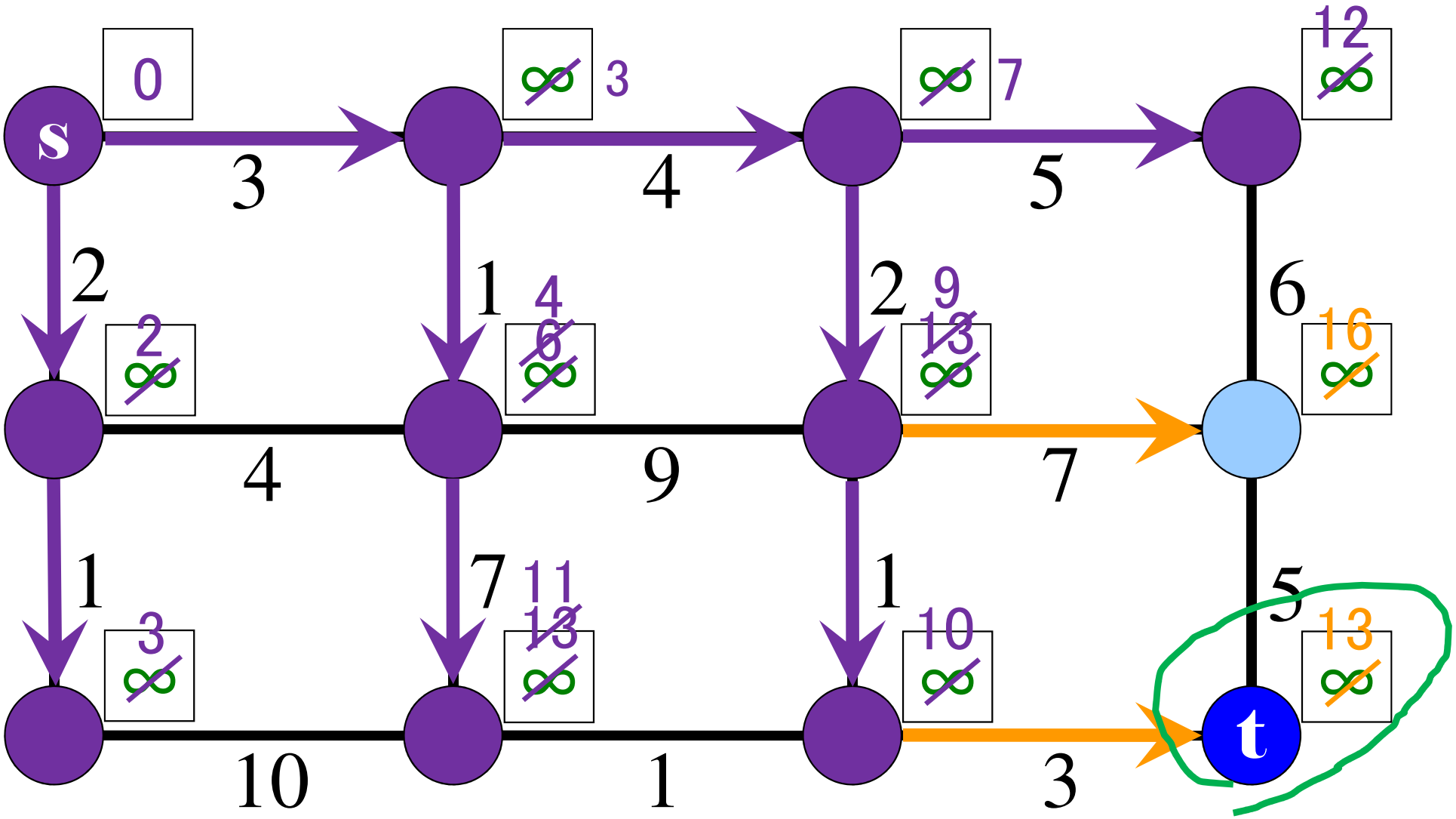
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



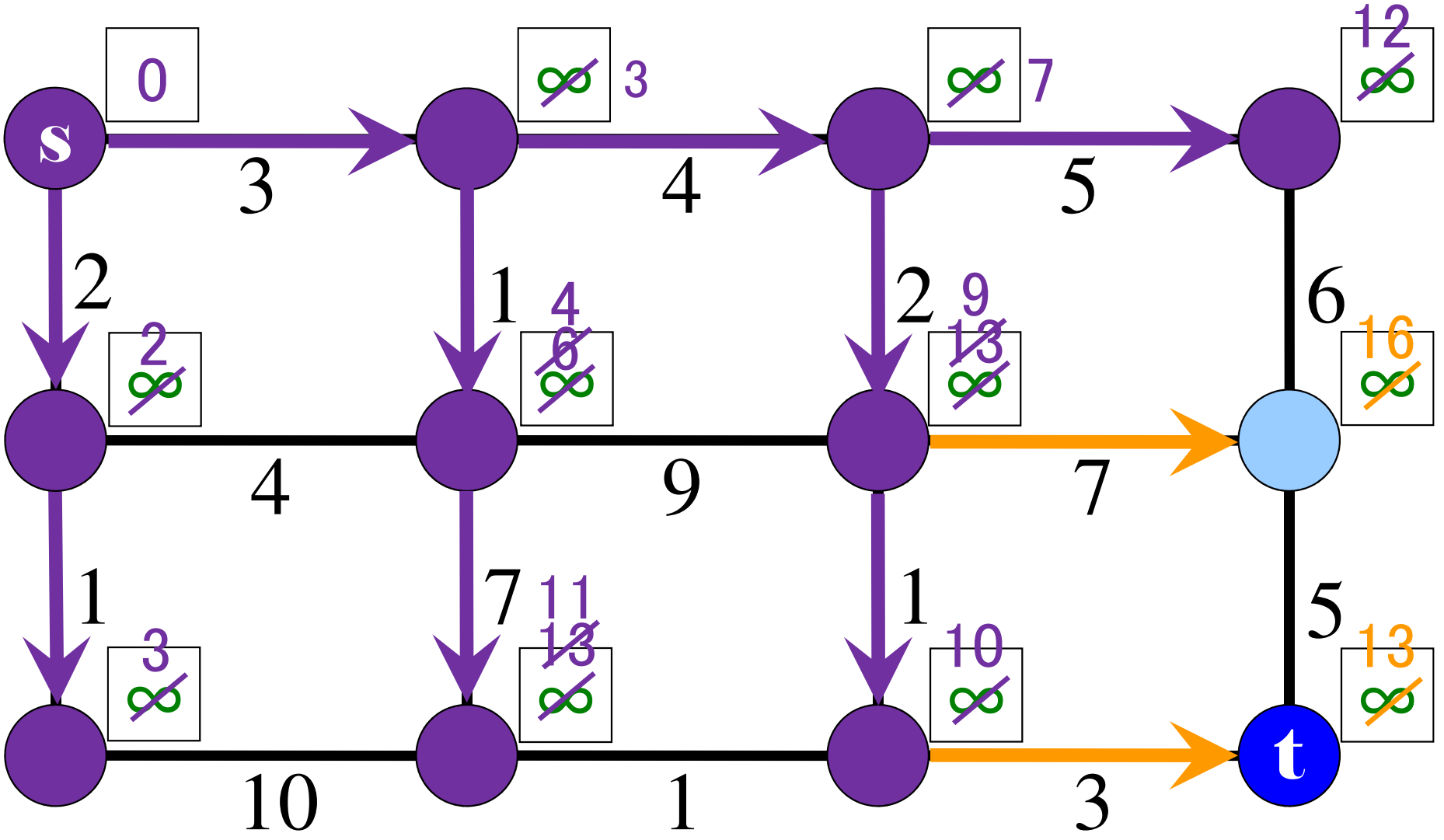
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

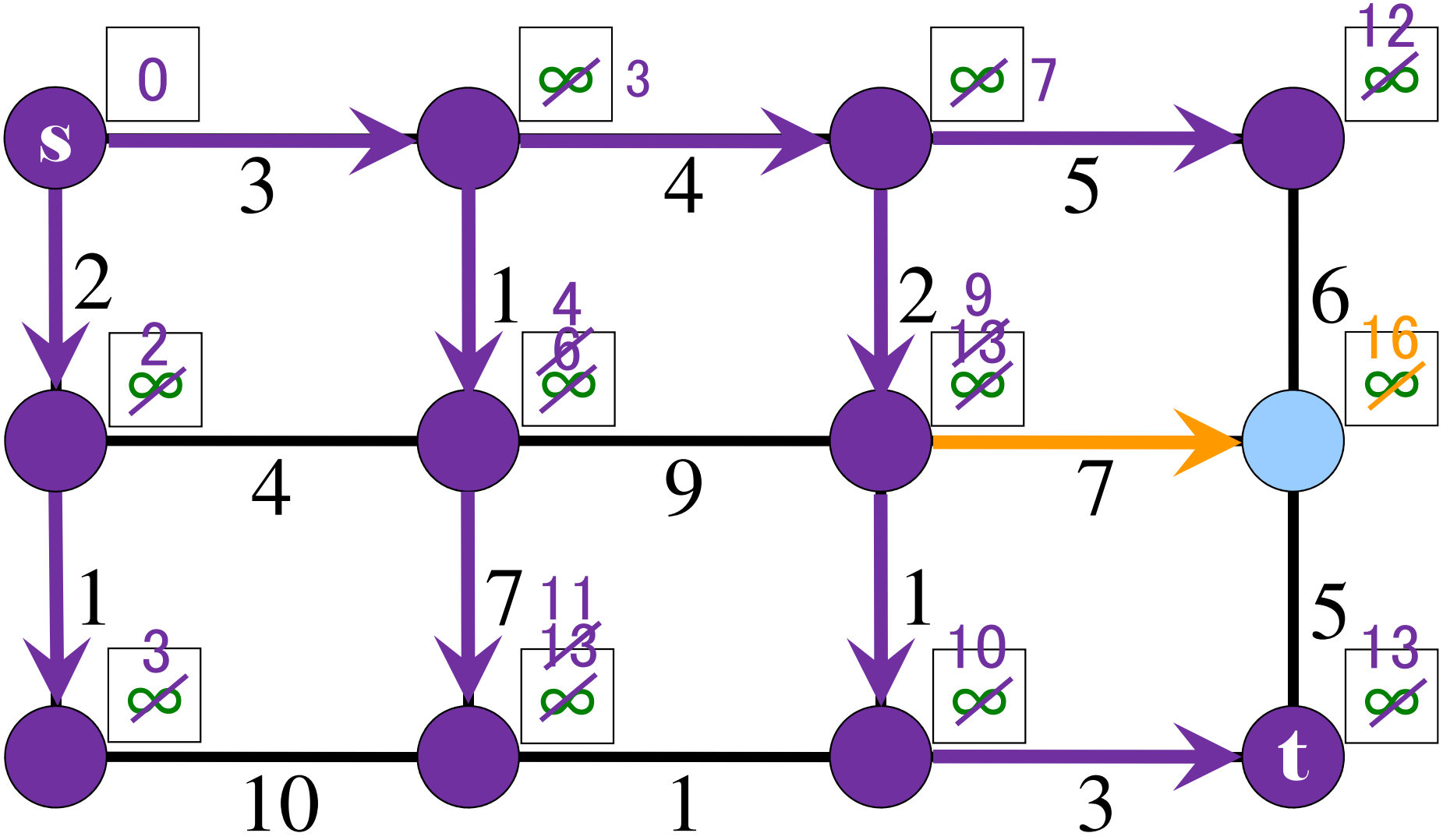


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

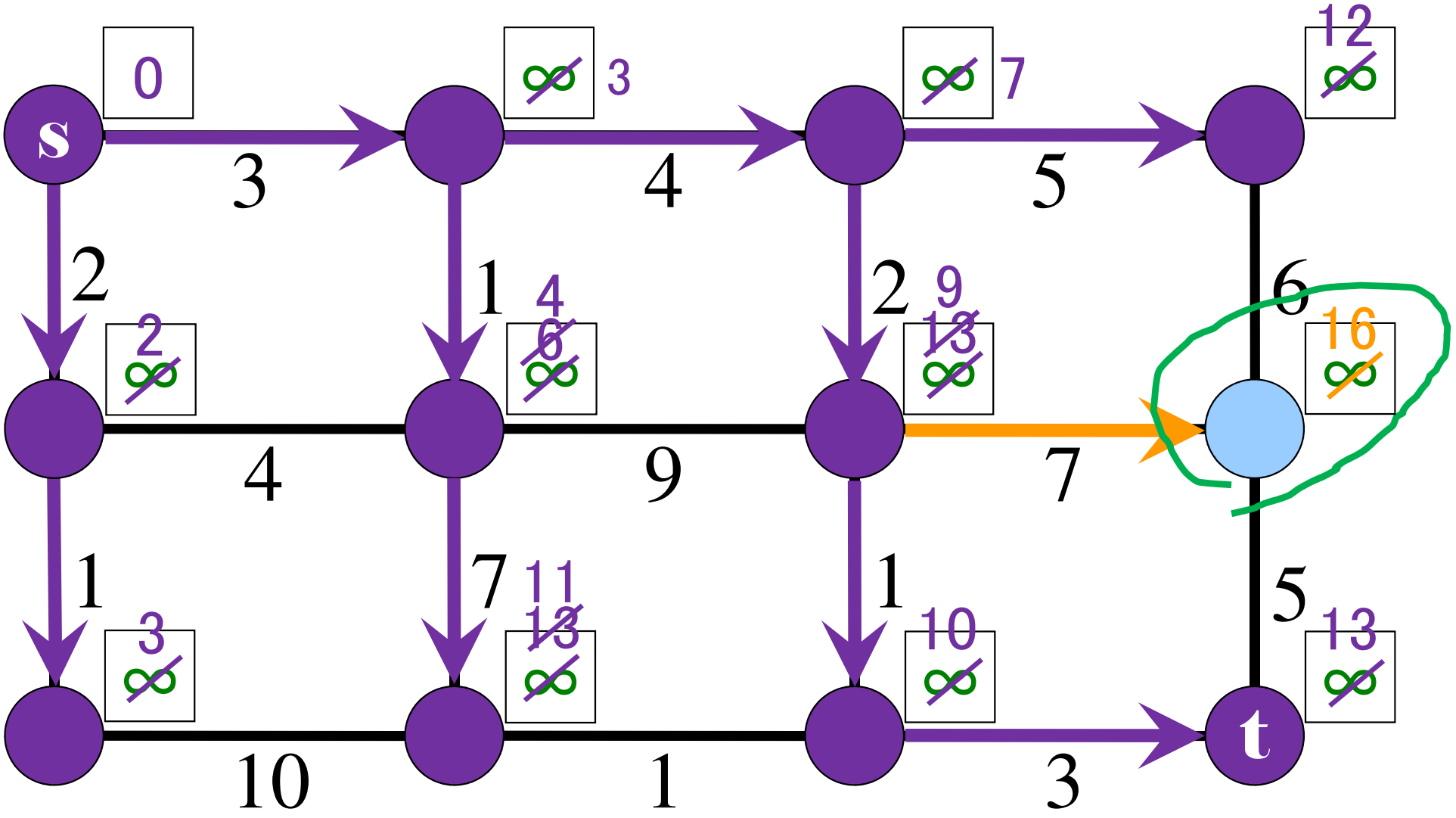
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



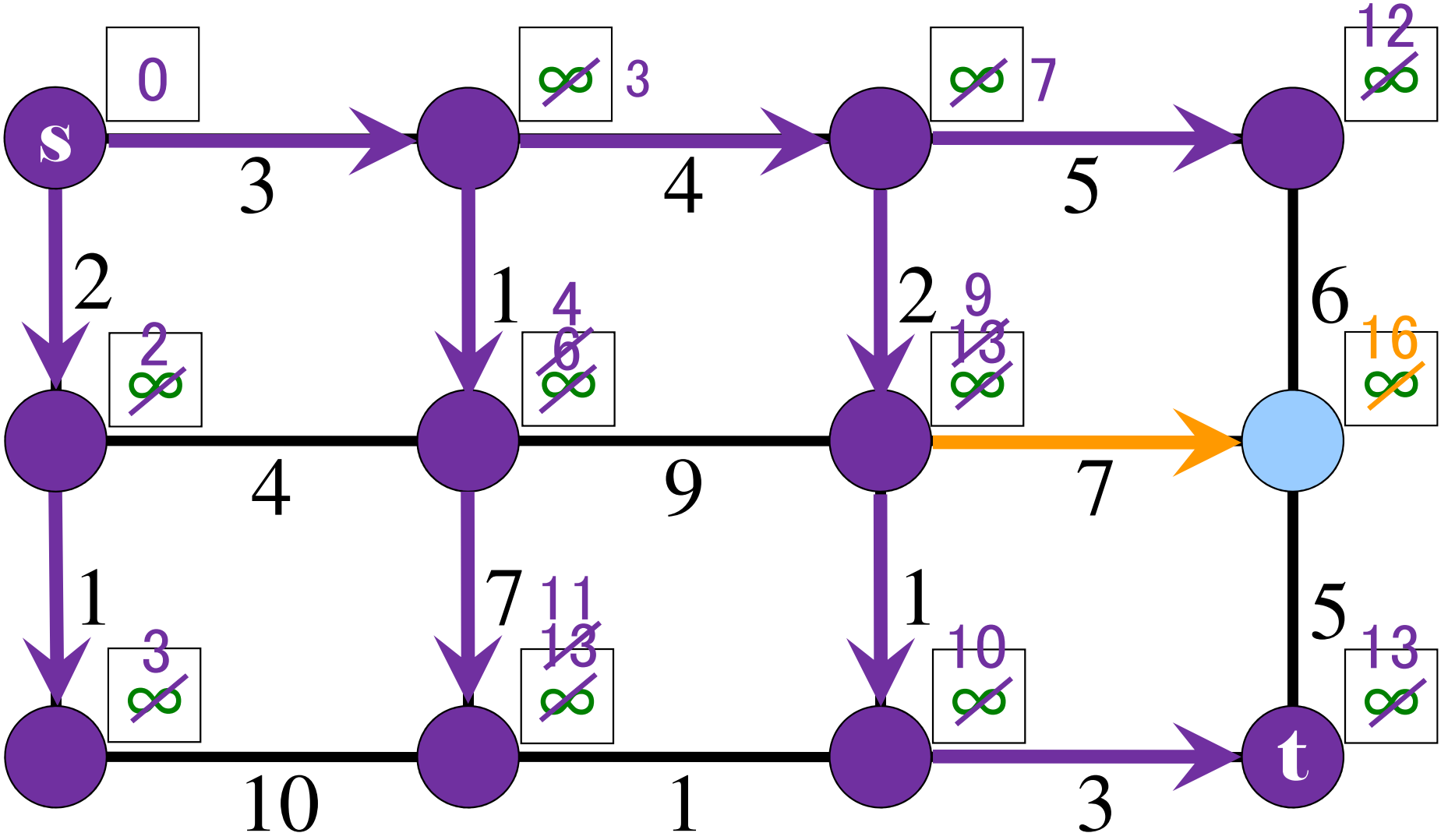
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

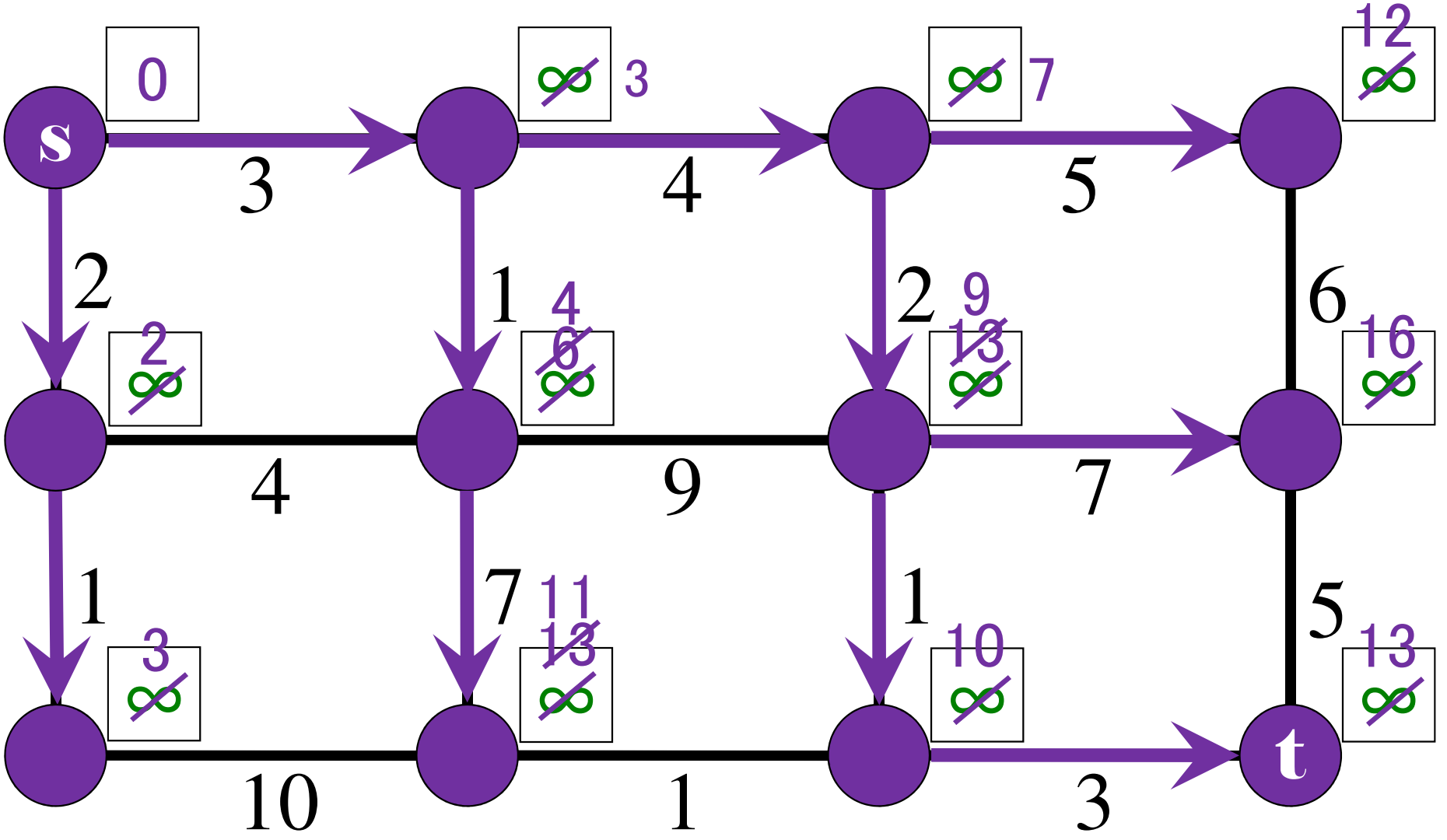


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

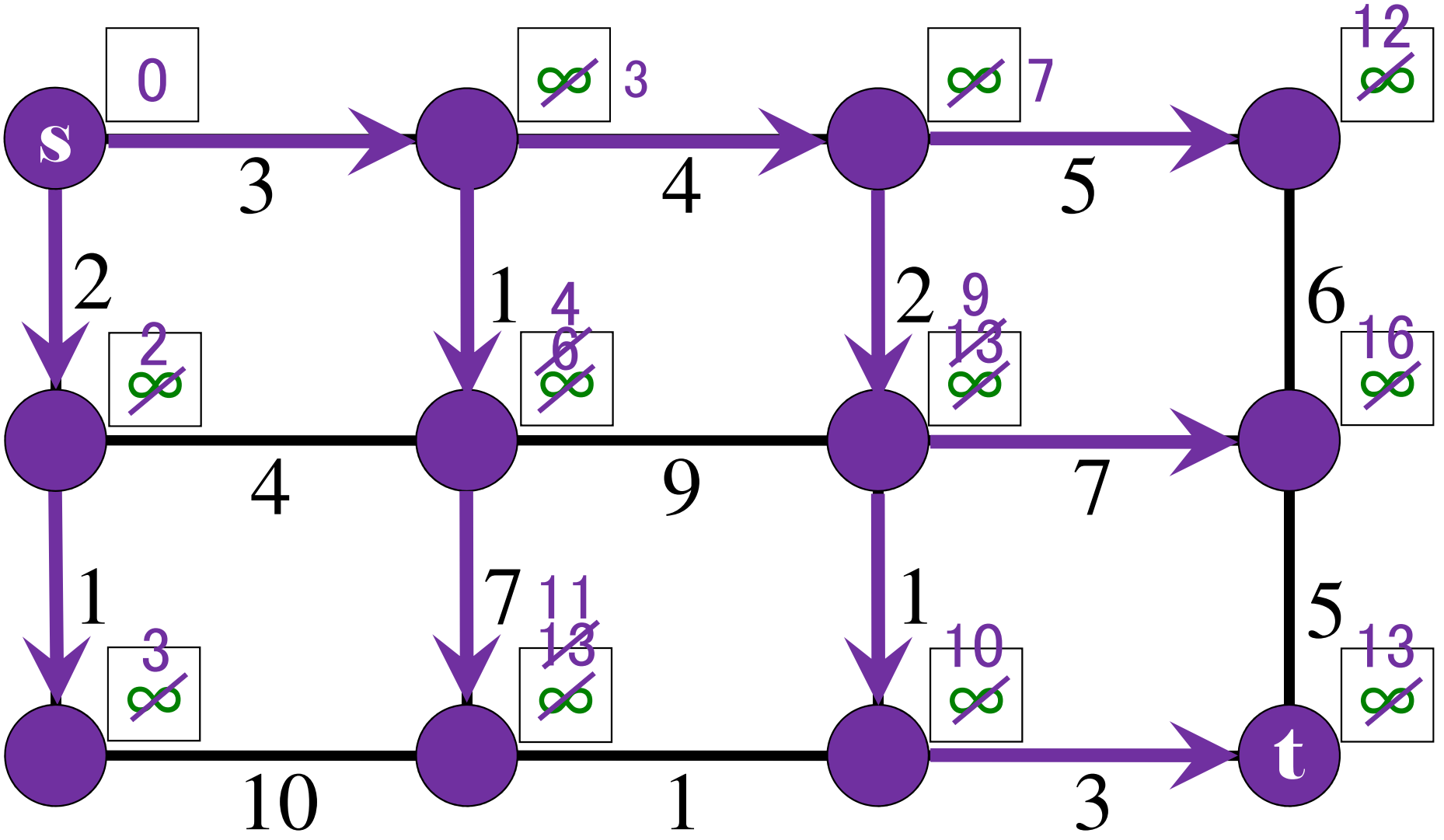
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



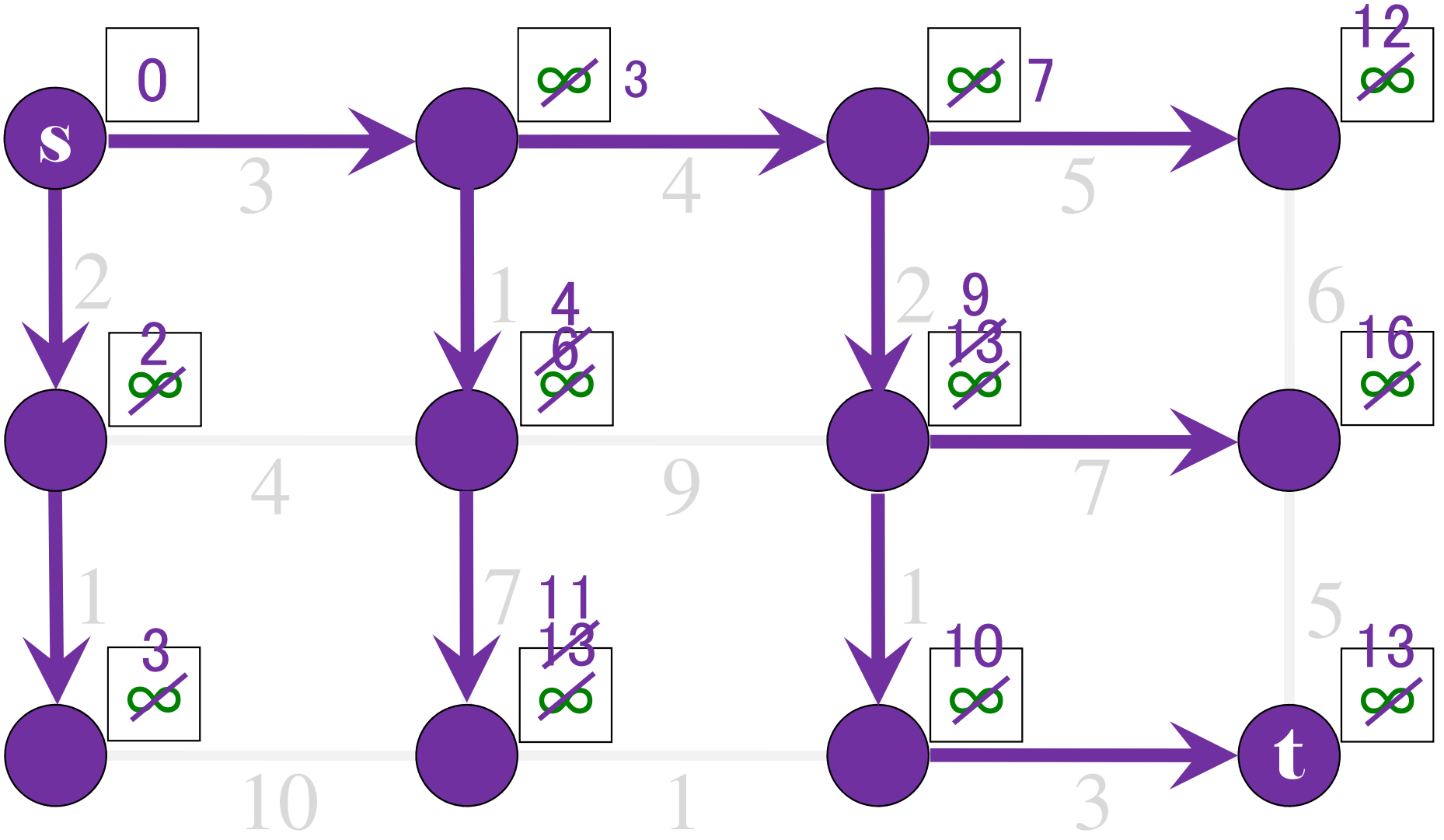
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



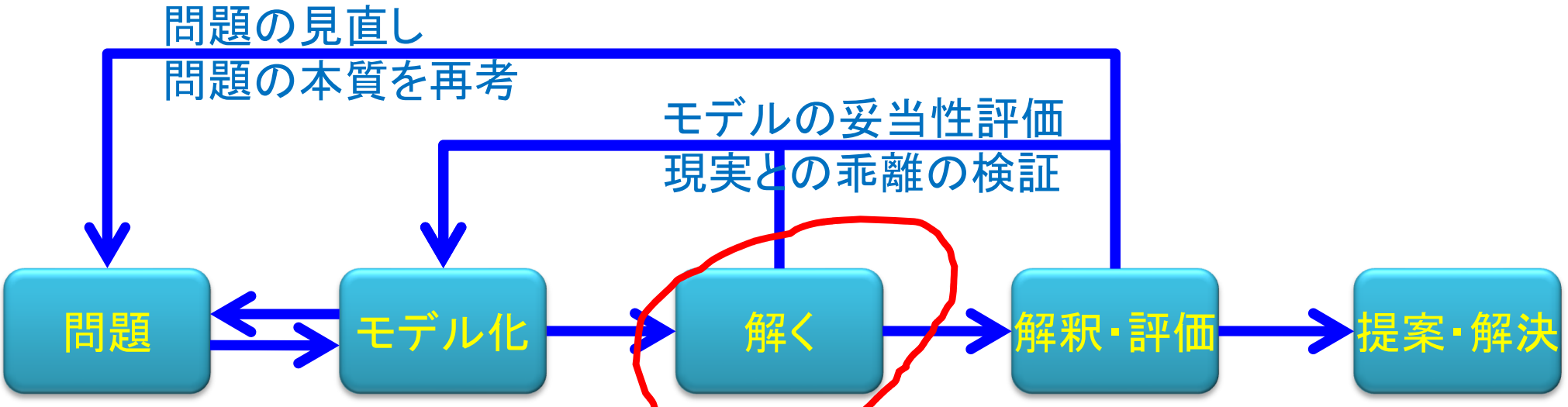
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

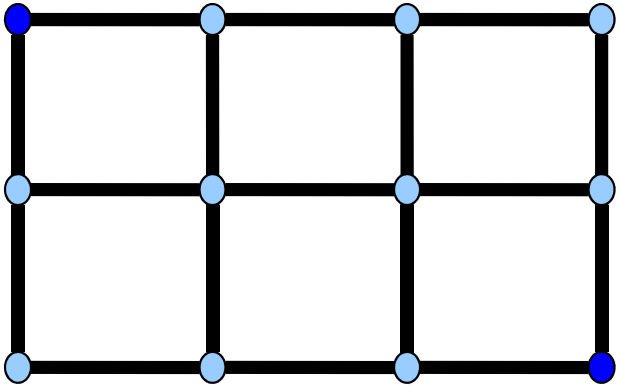
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

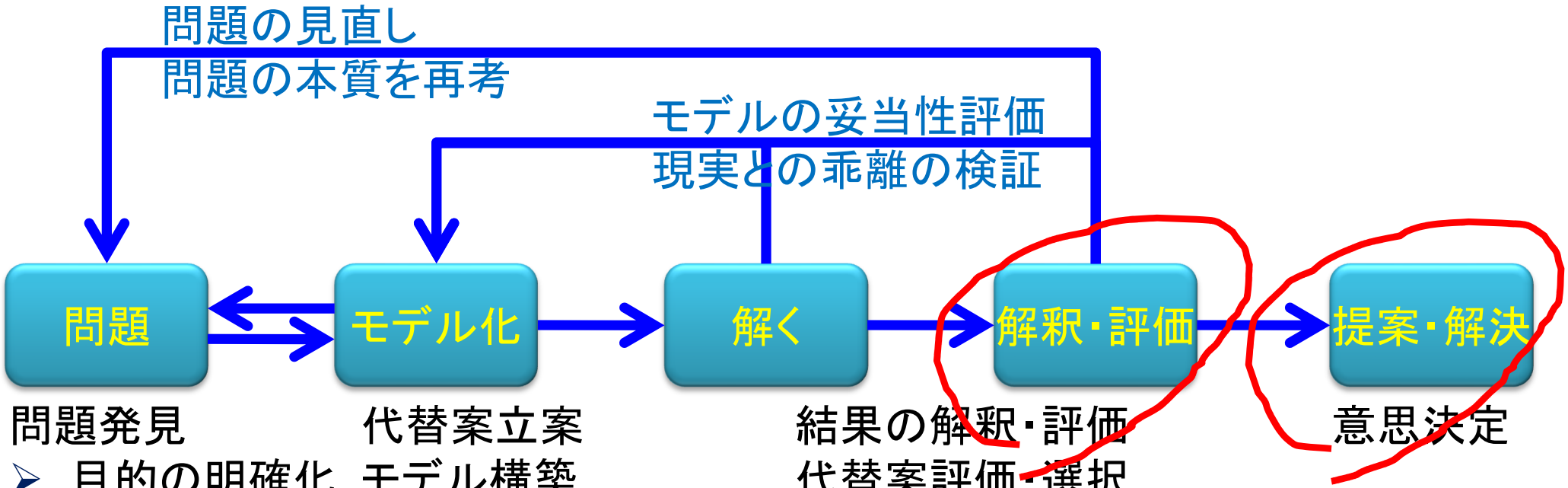
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

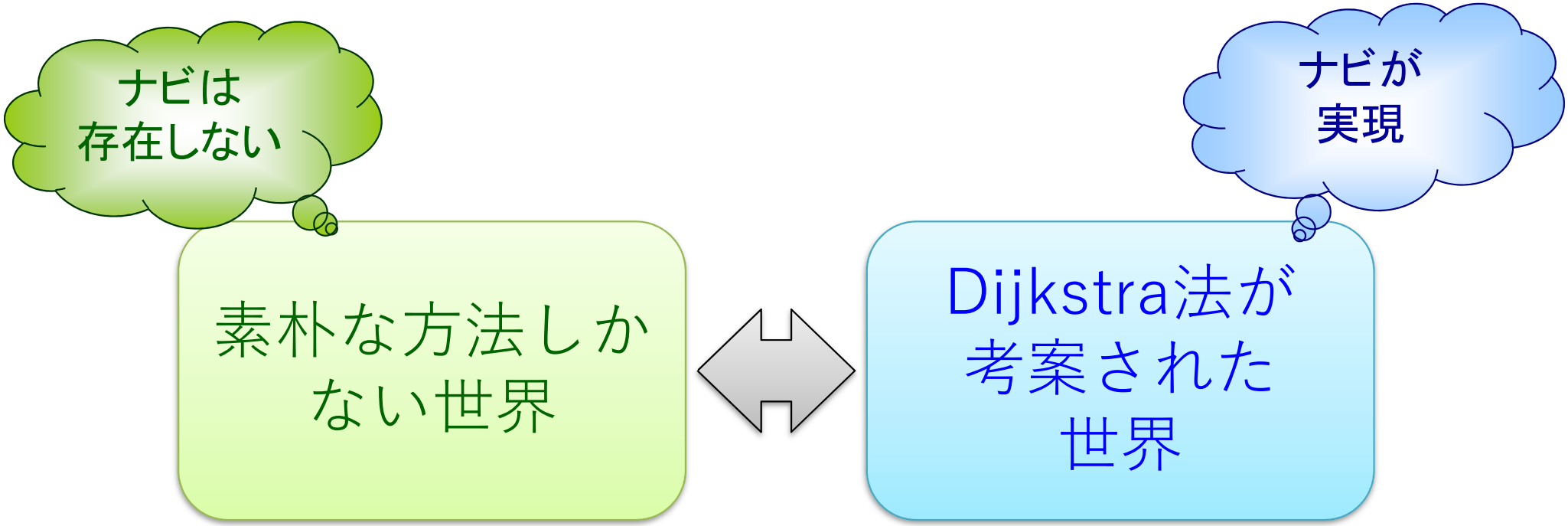
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

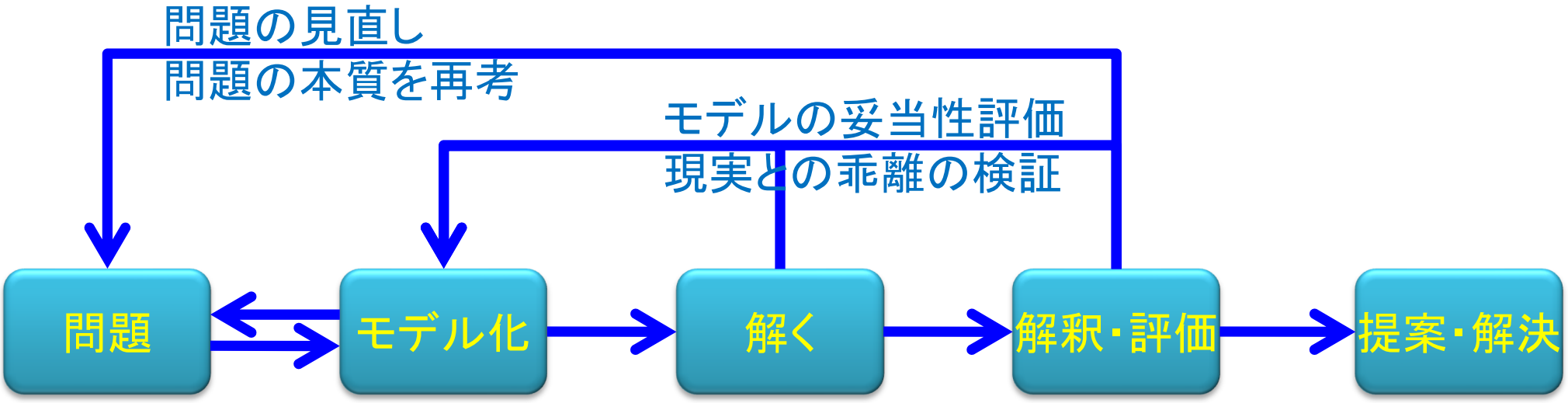
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



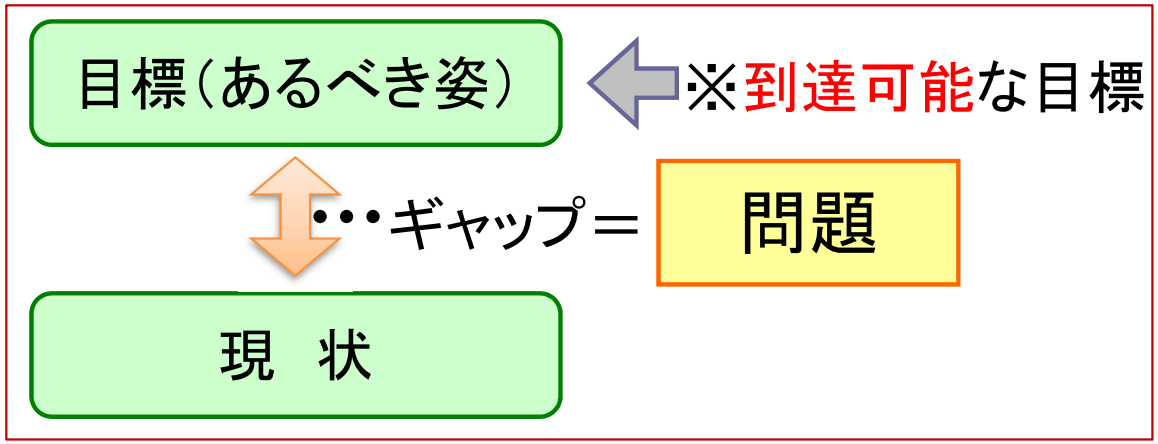
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

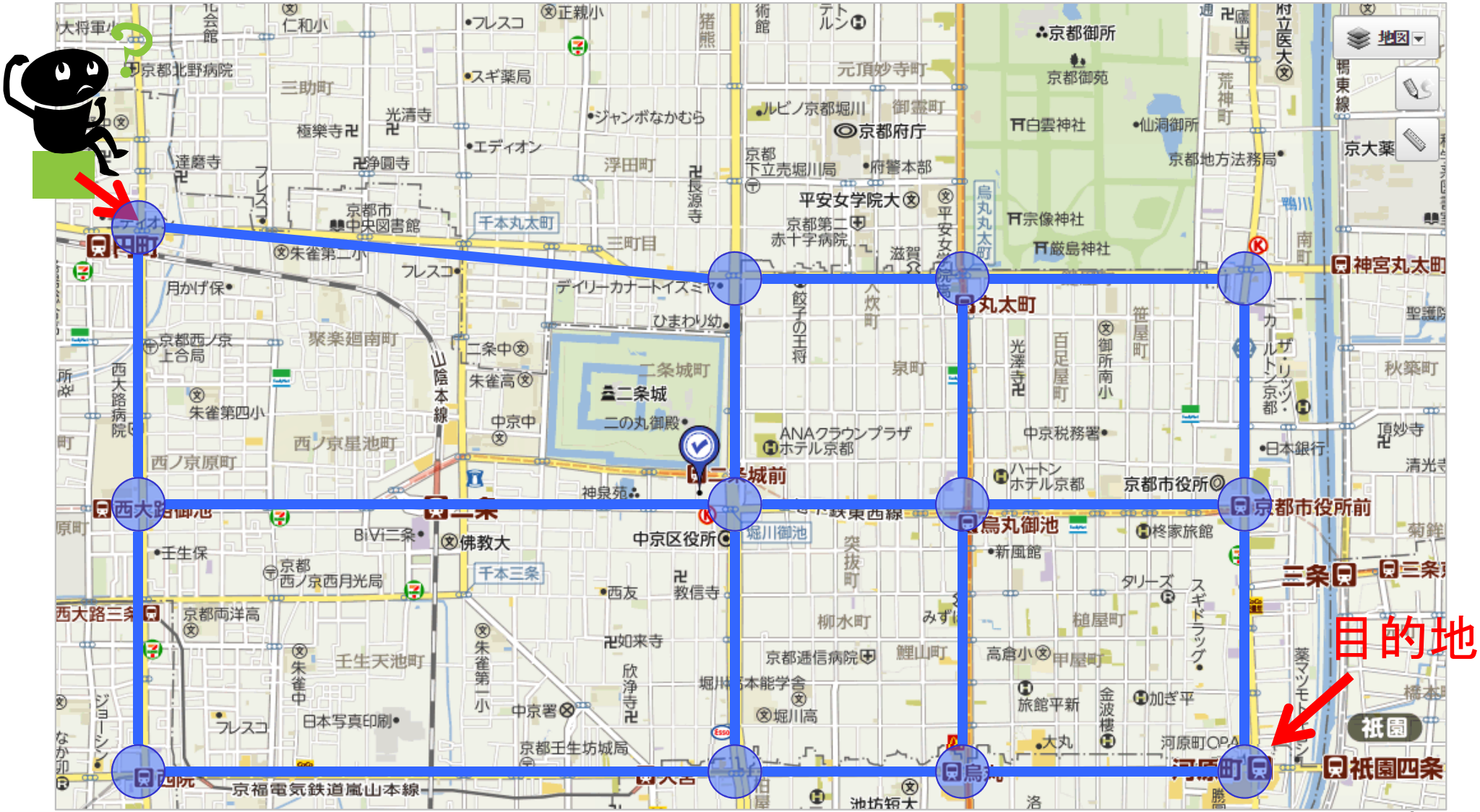
意思決定

問題の定義



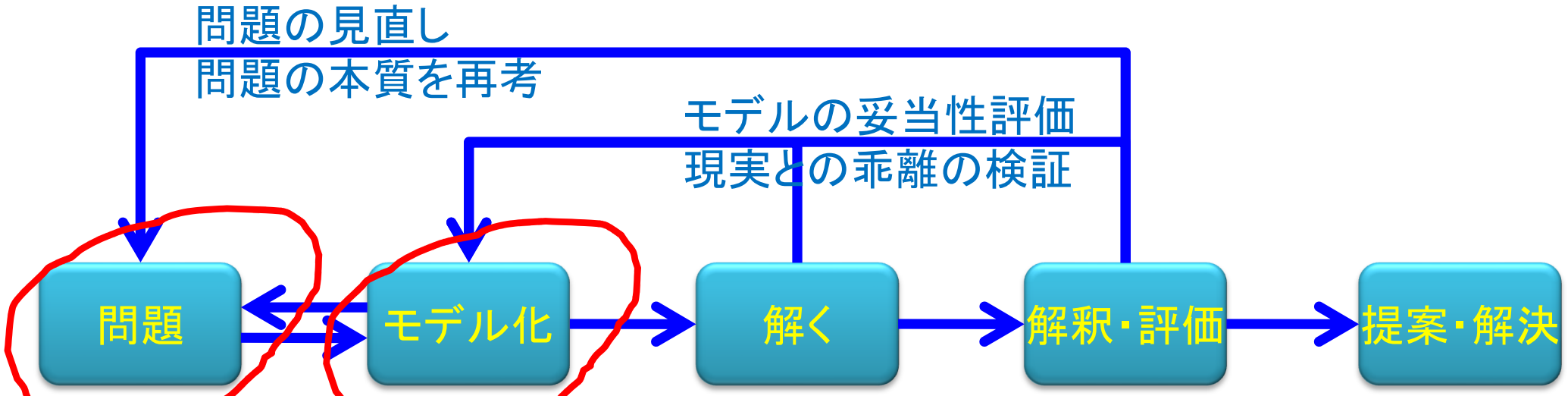
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

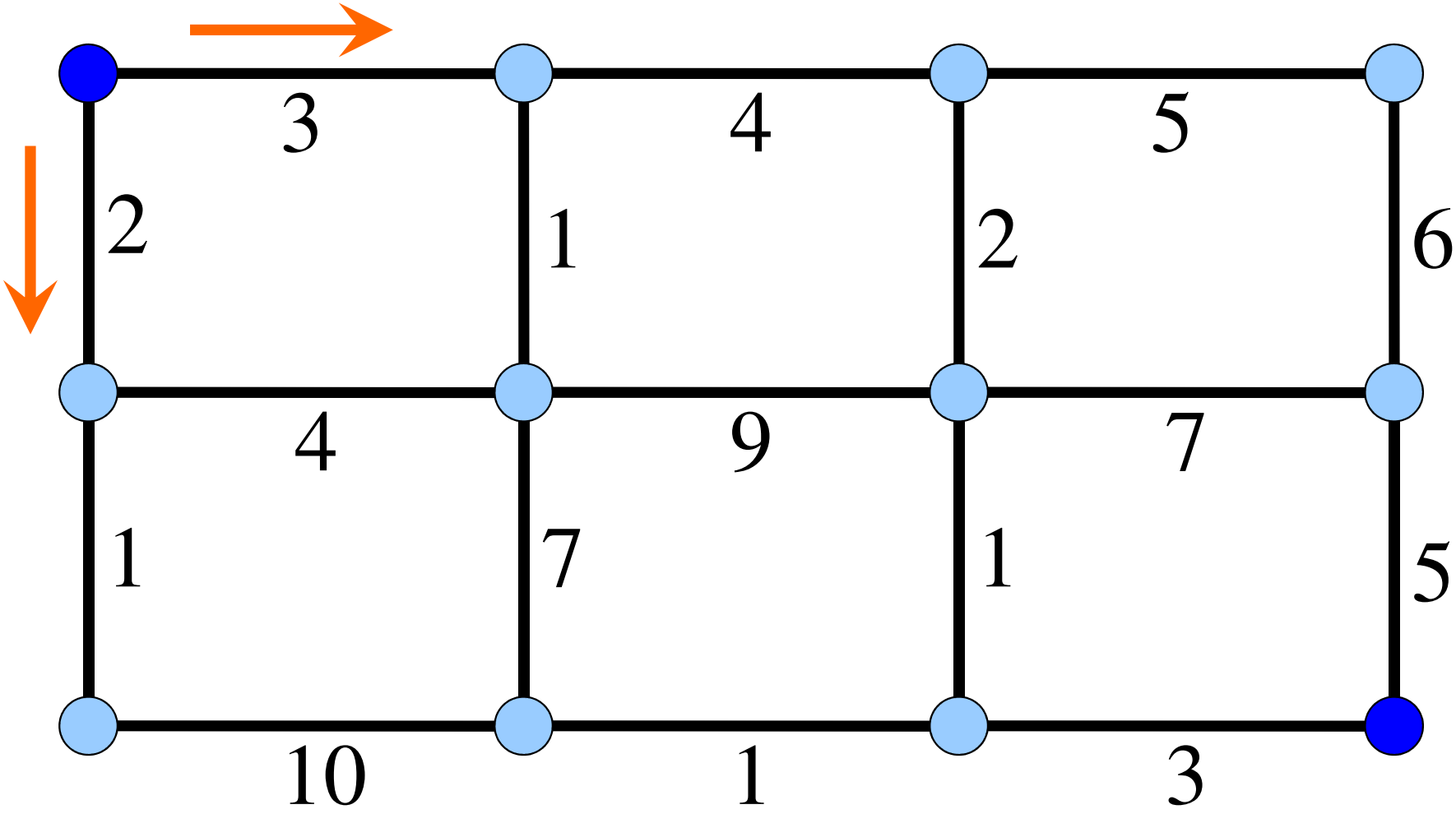
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

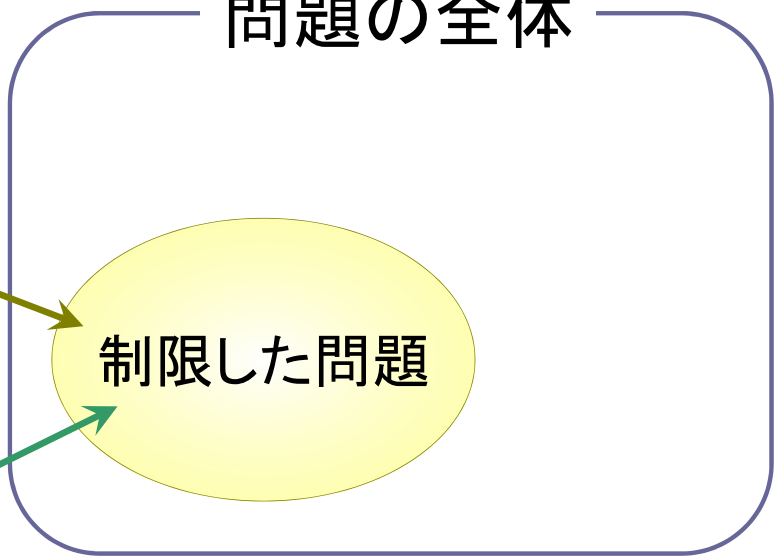


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

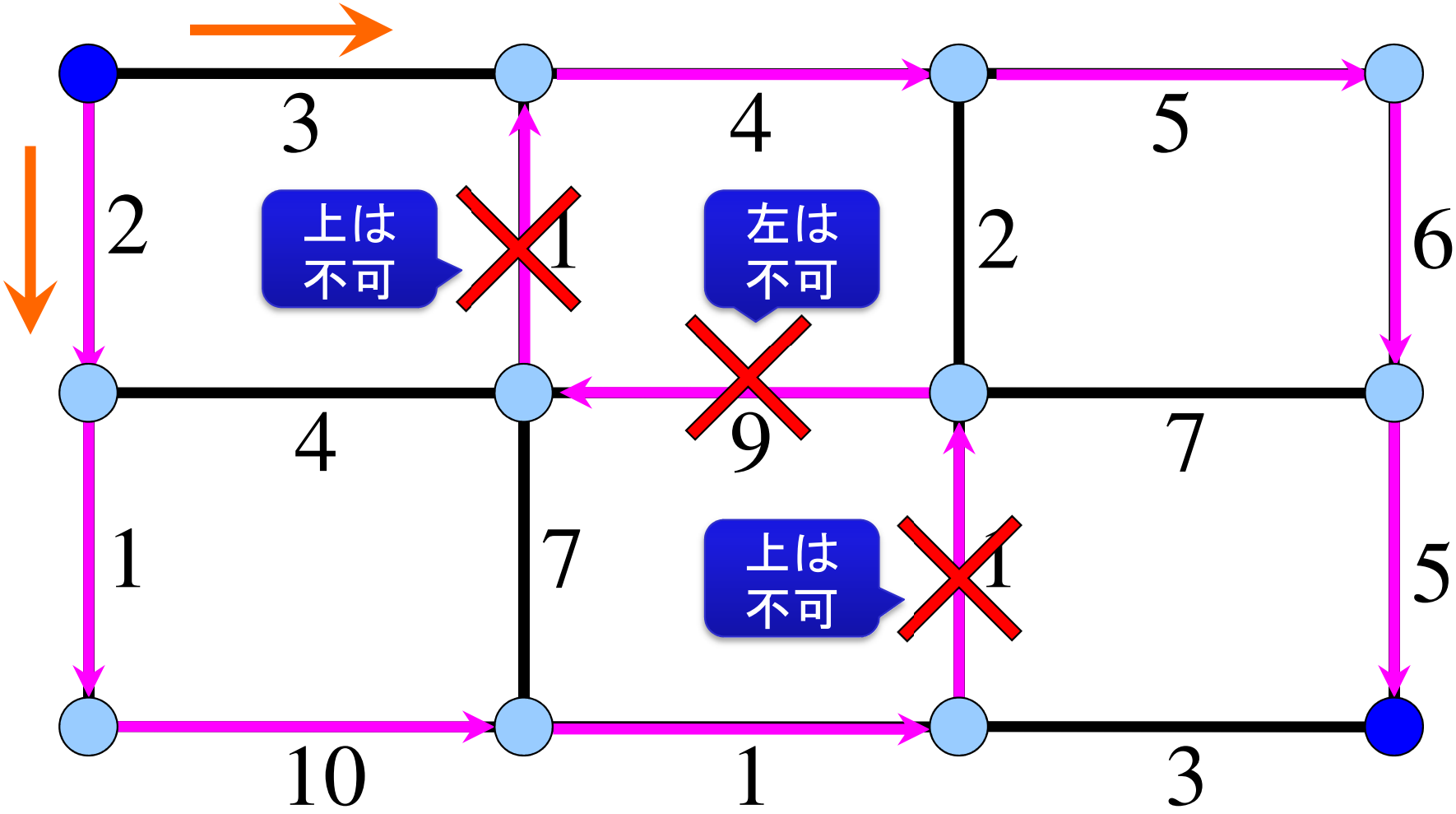
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

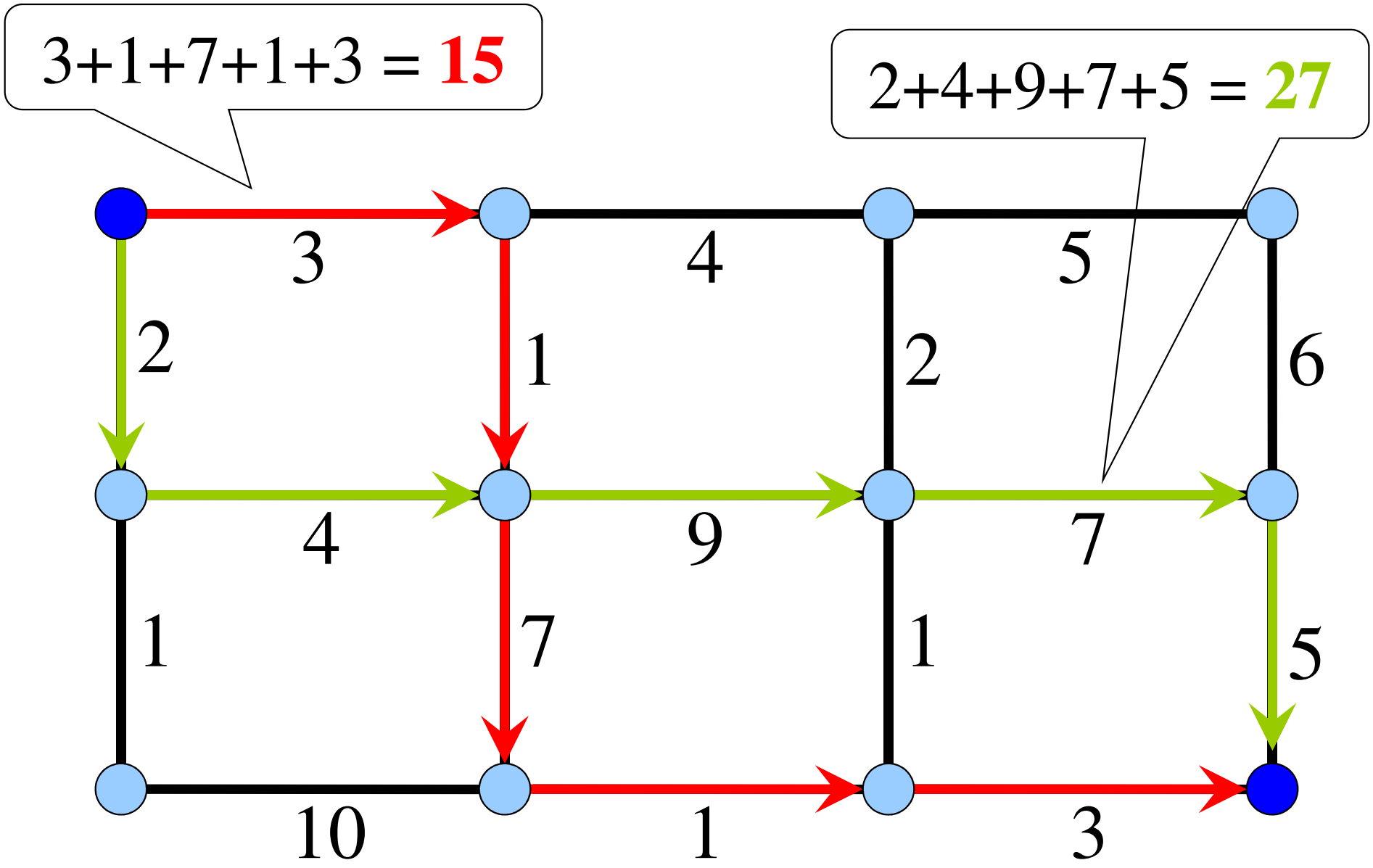
難しいなら易しくすればいいのさ！

制限した問題

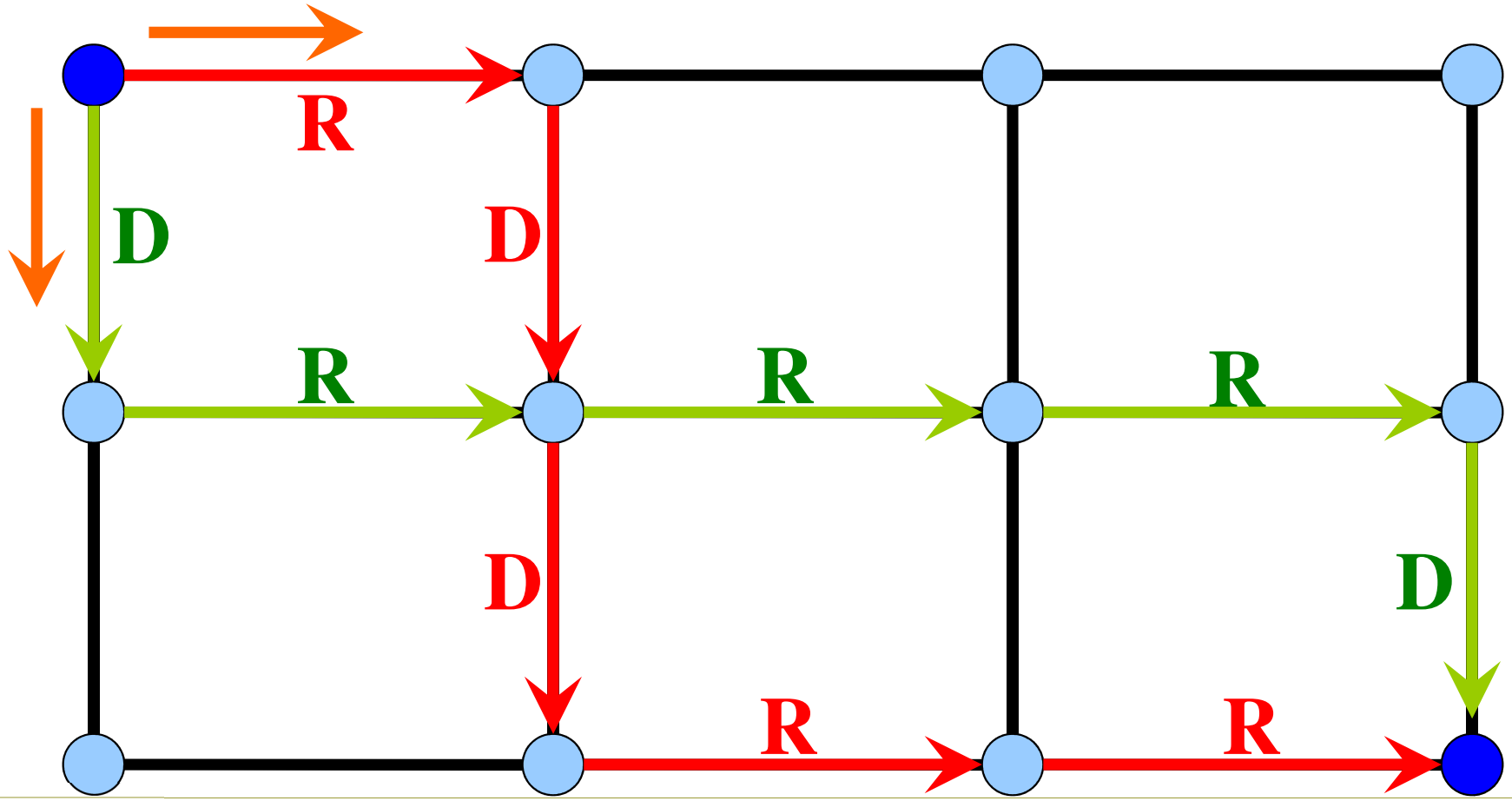
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



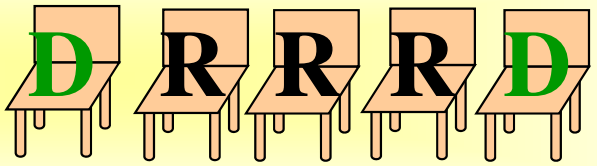
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



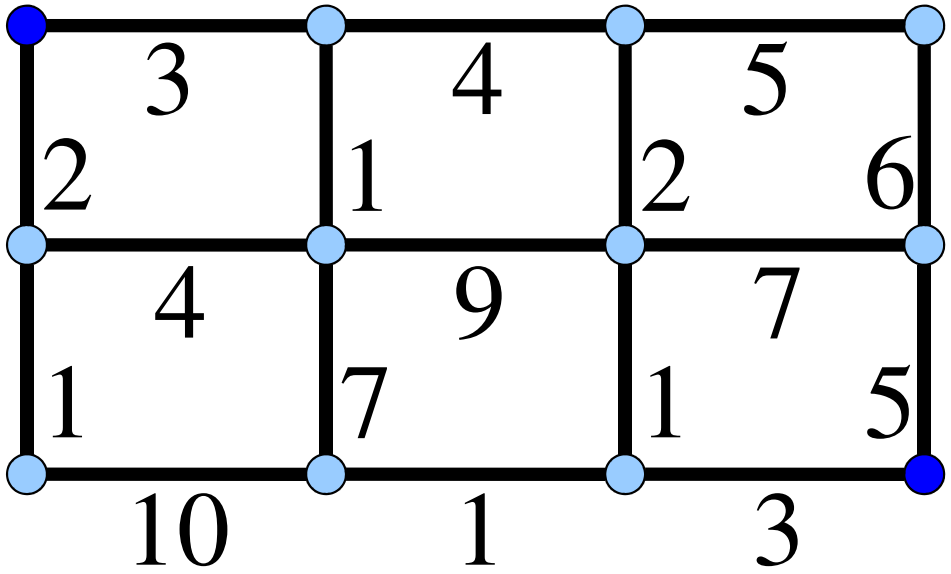
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

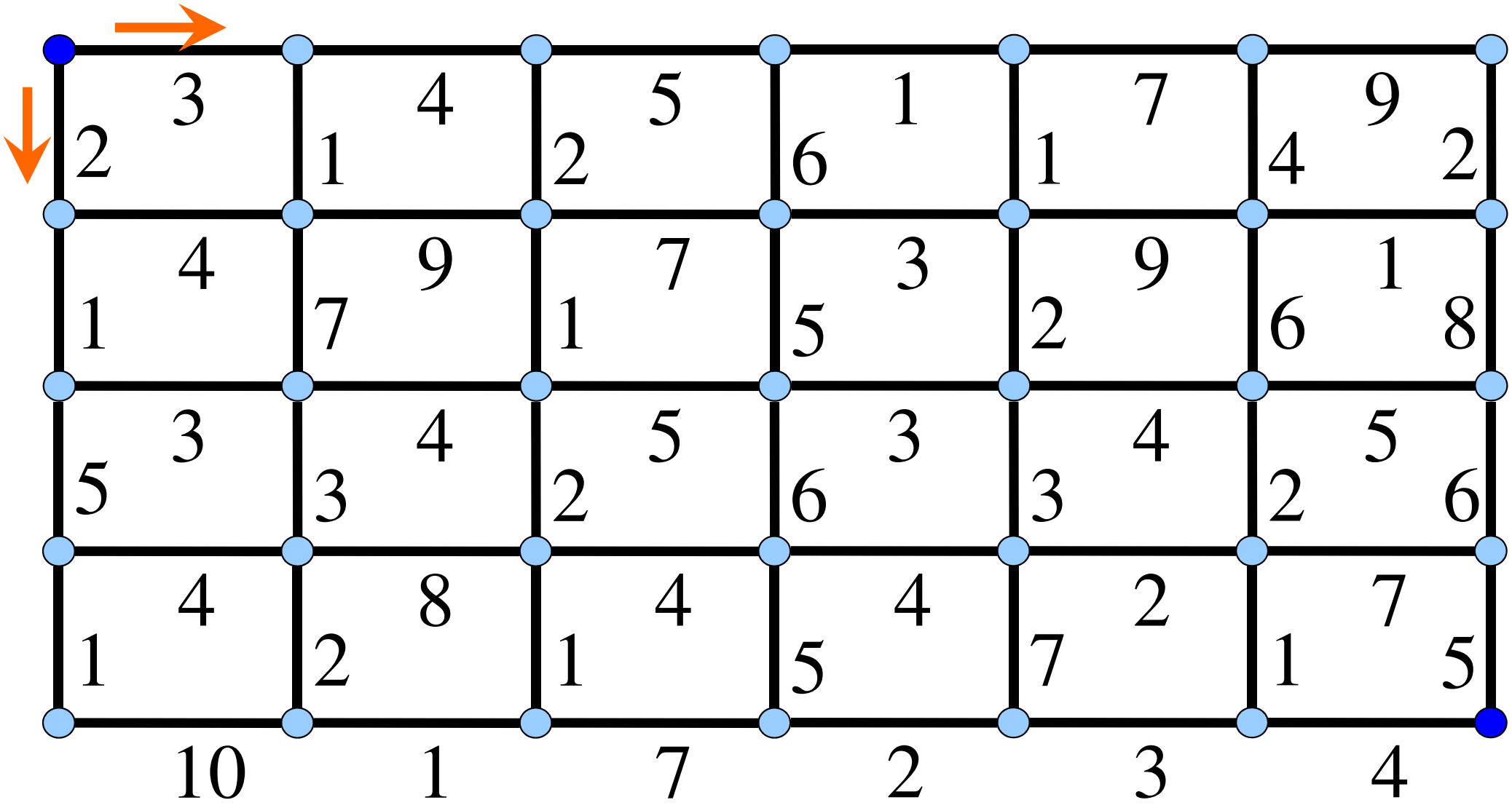
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

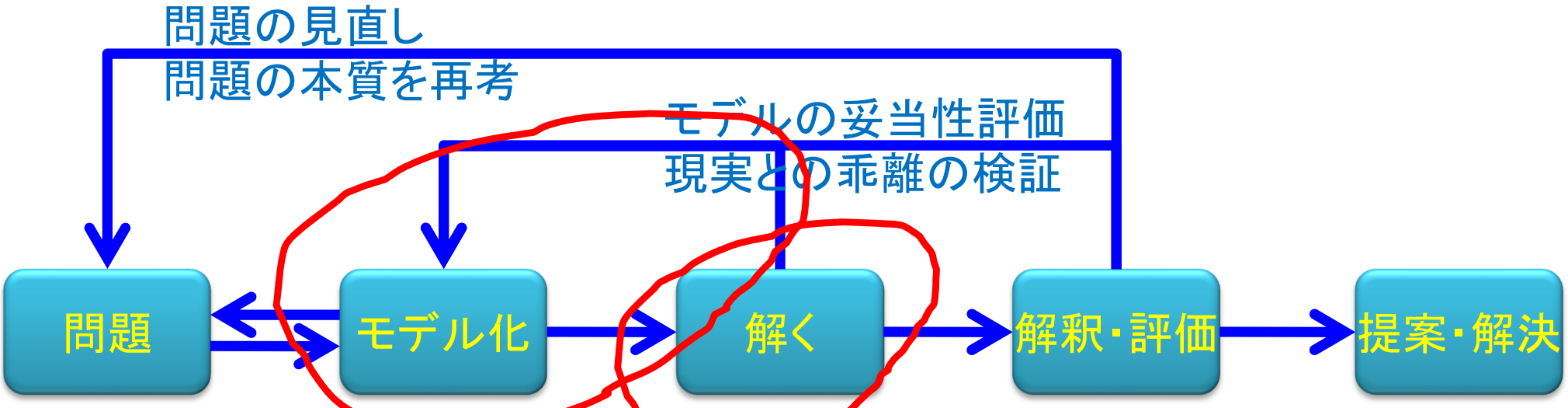
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

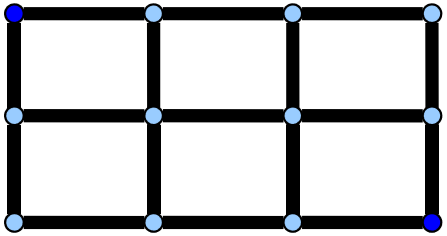
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

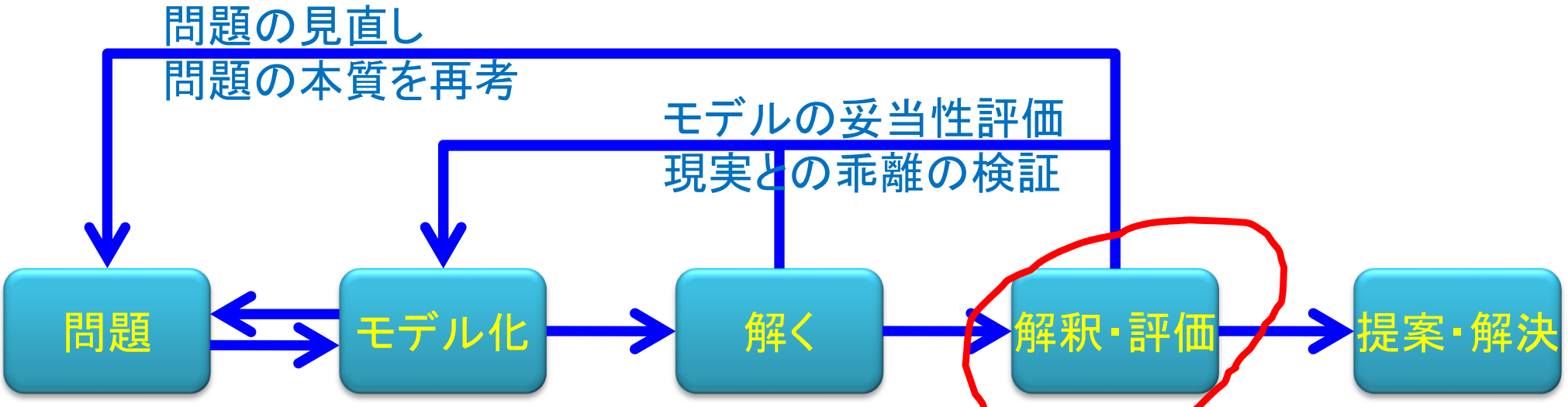
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

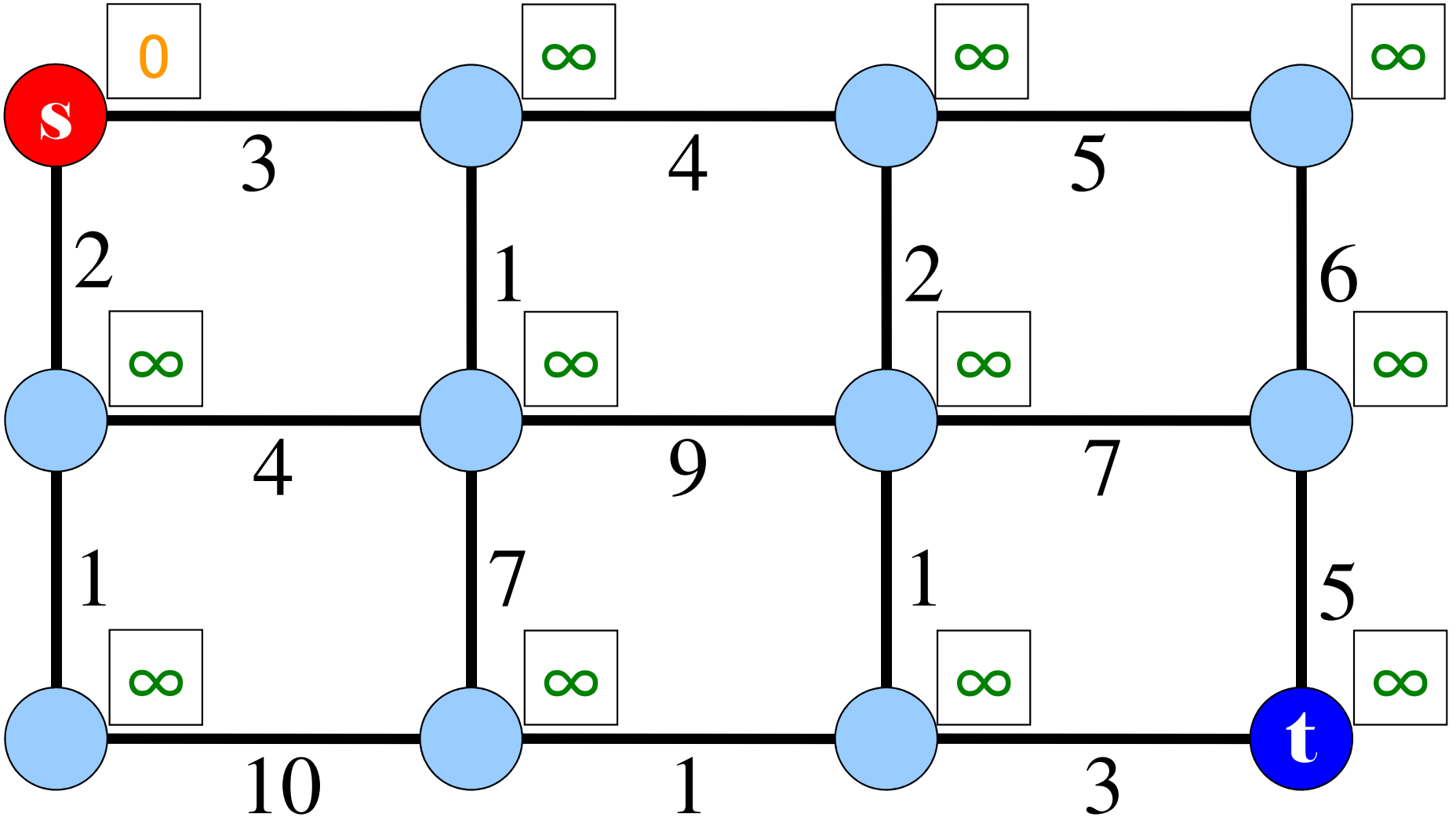
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

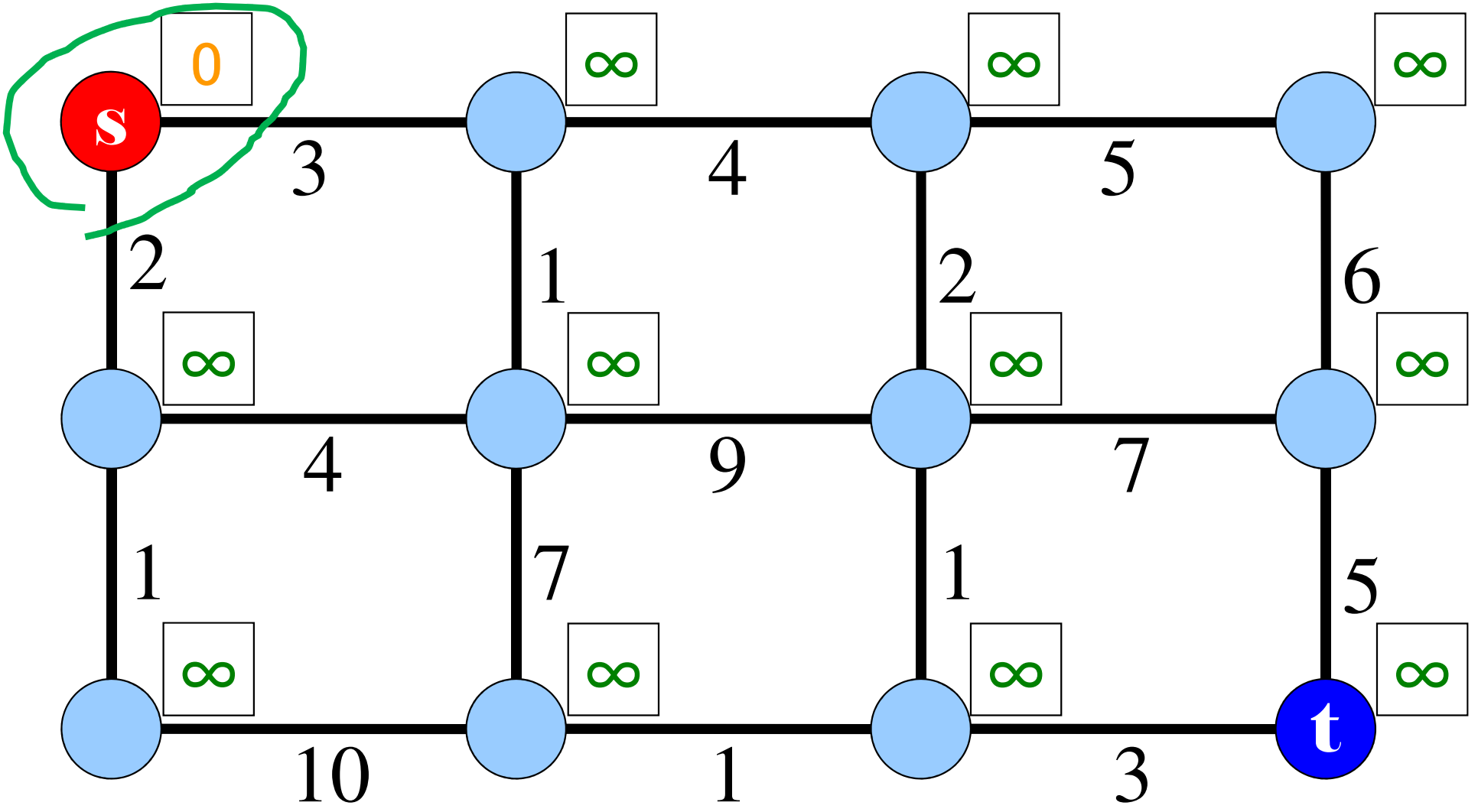
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



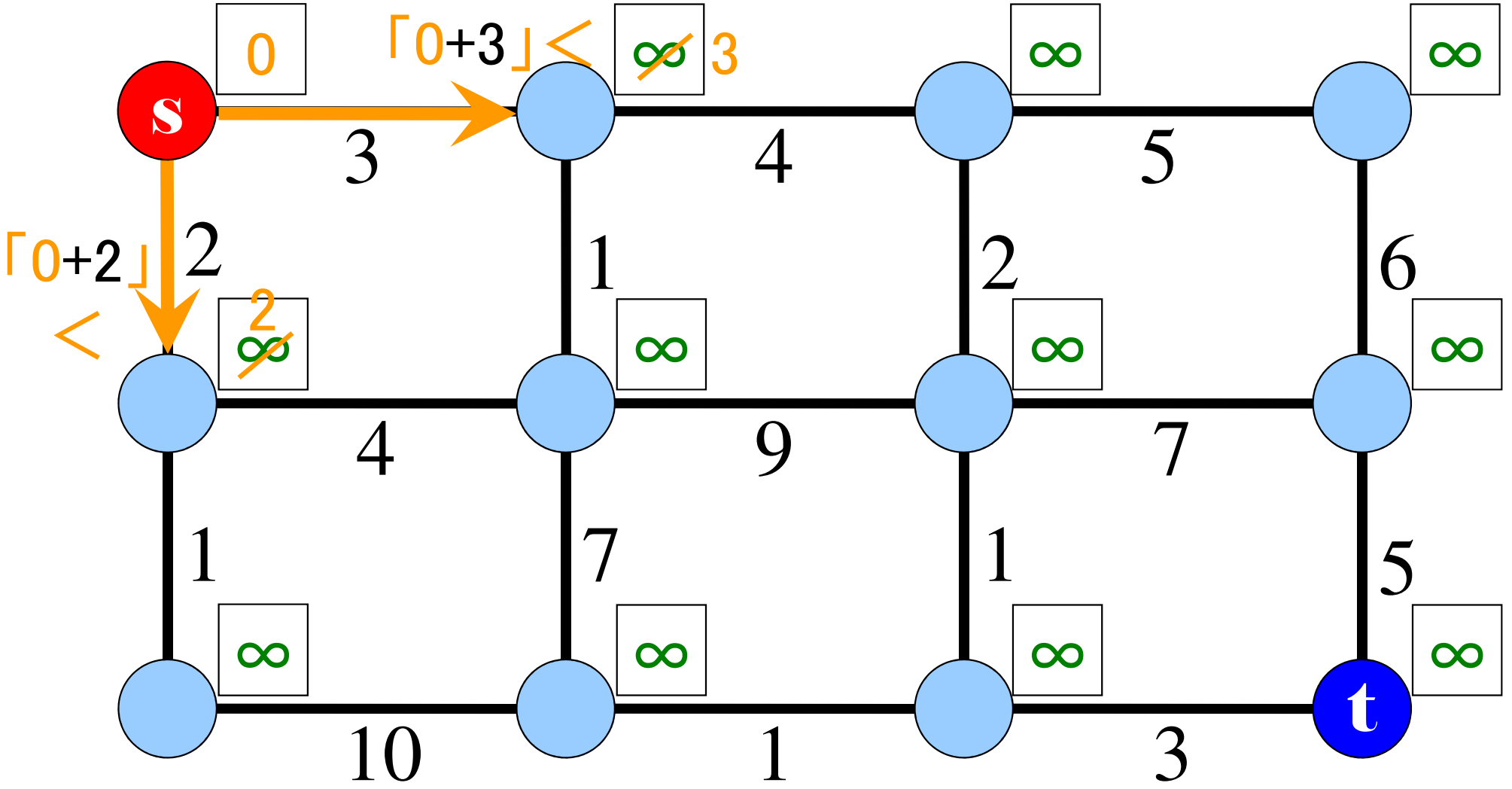
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

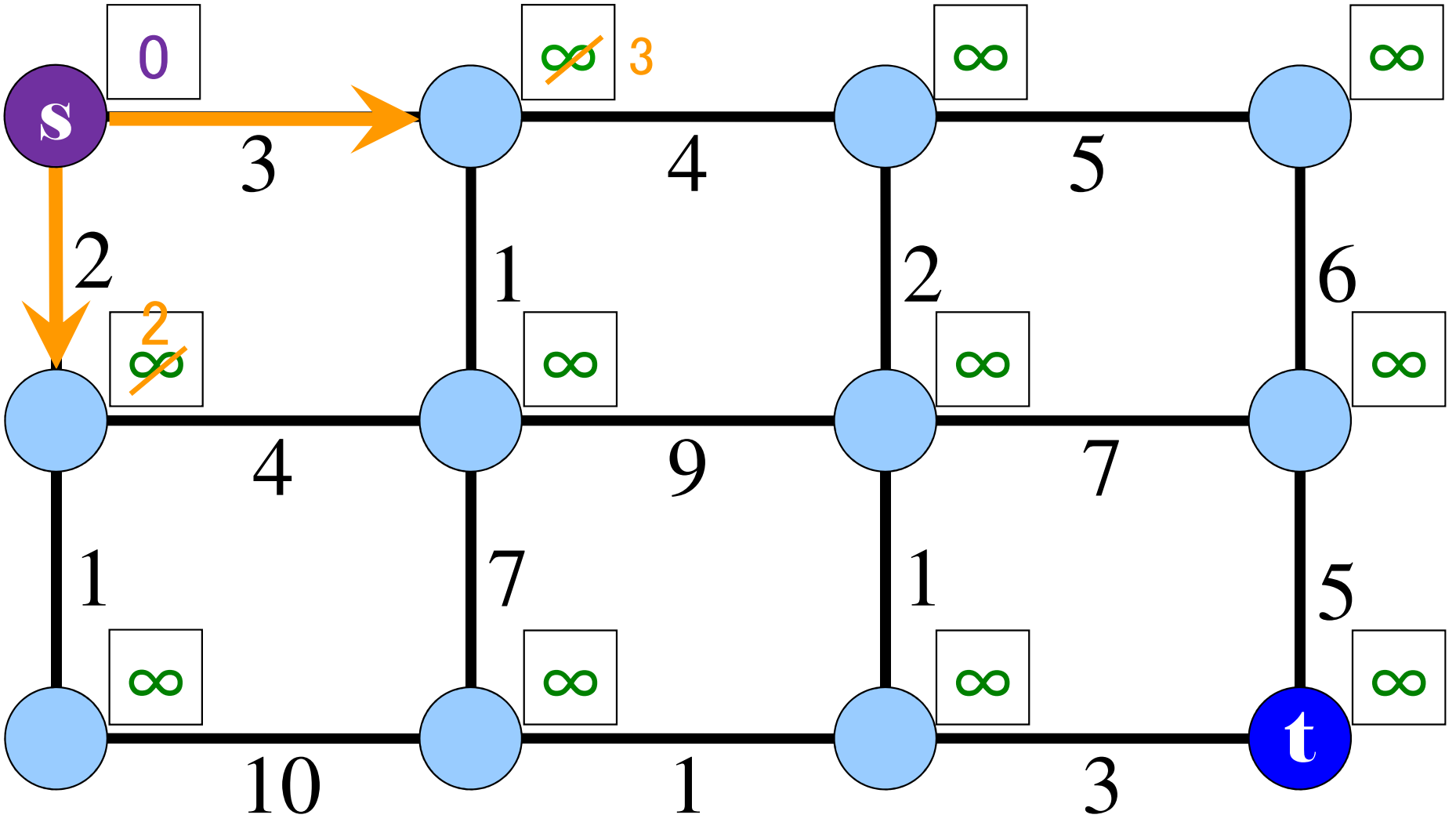


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

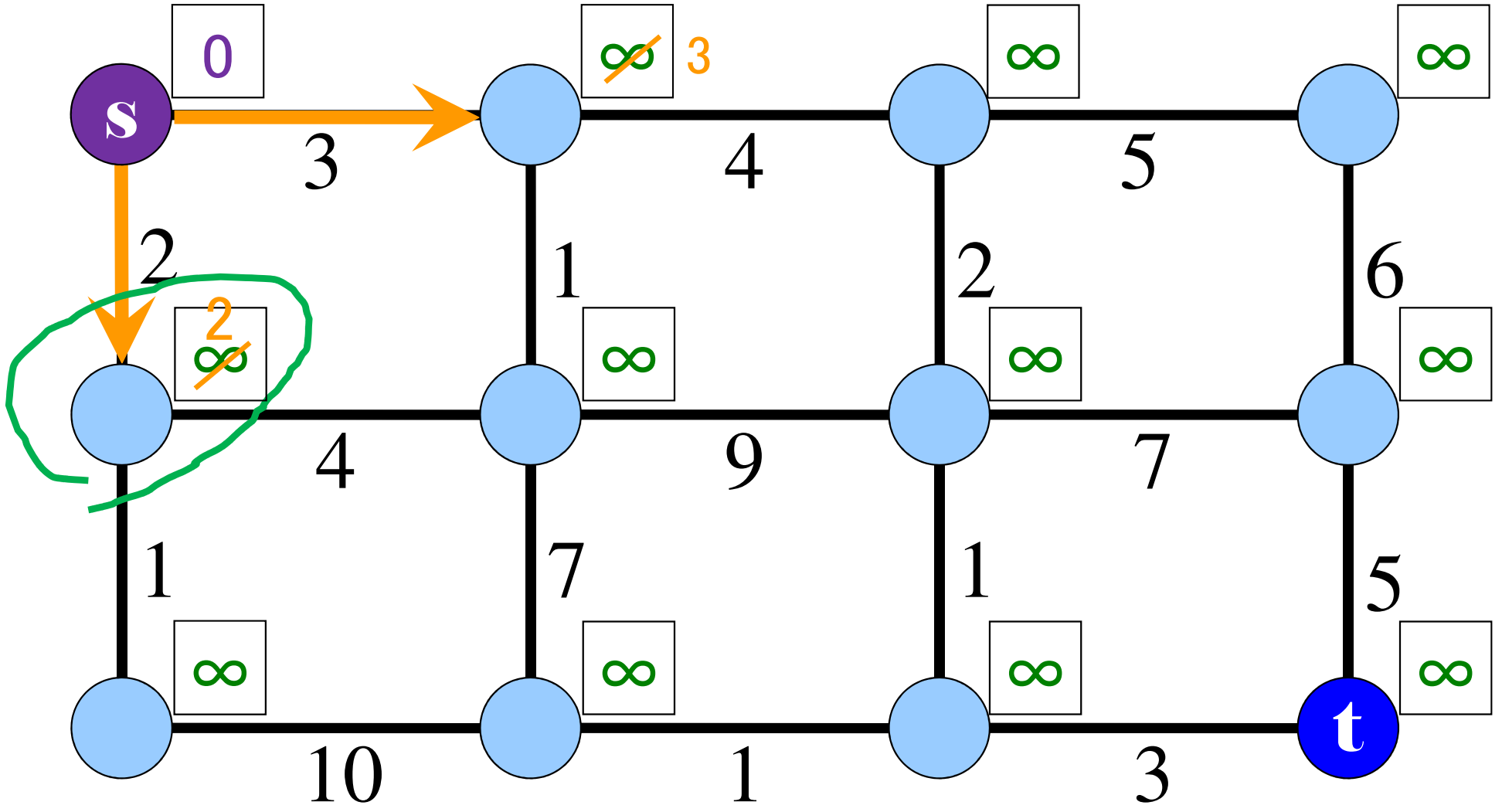
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



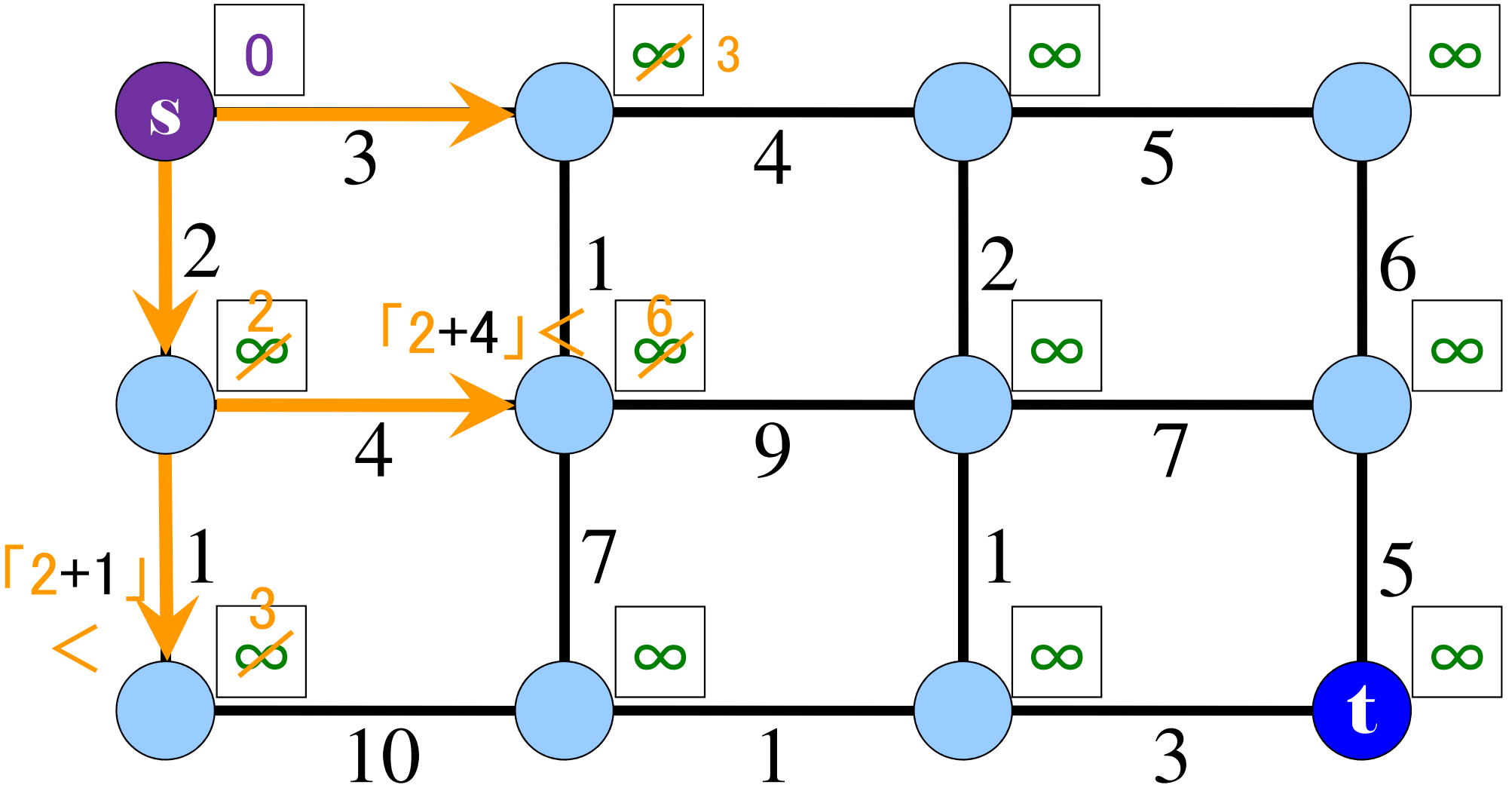
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

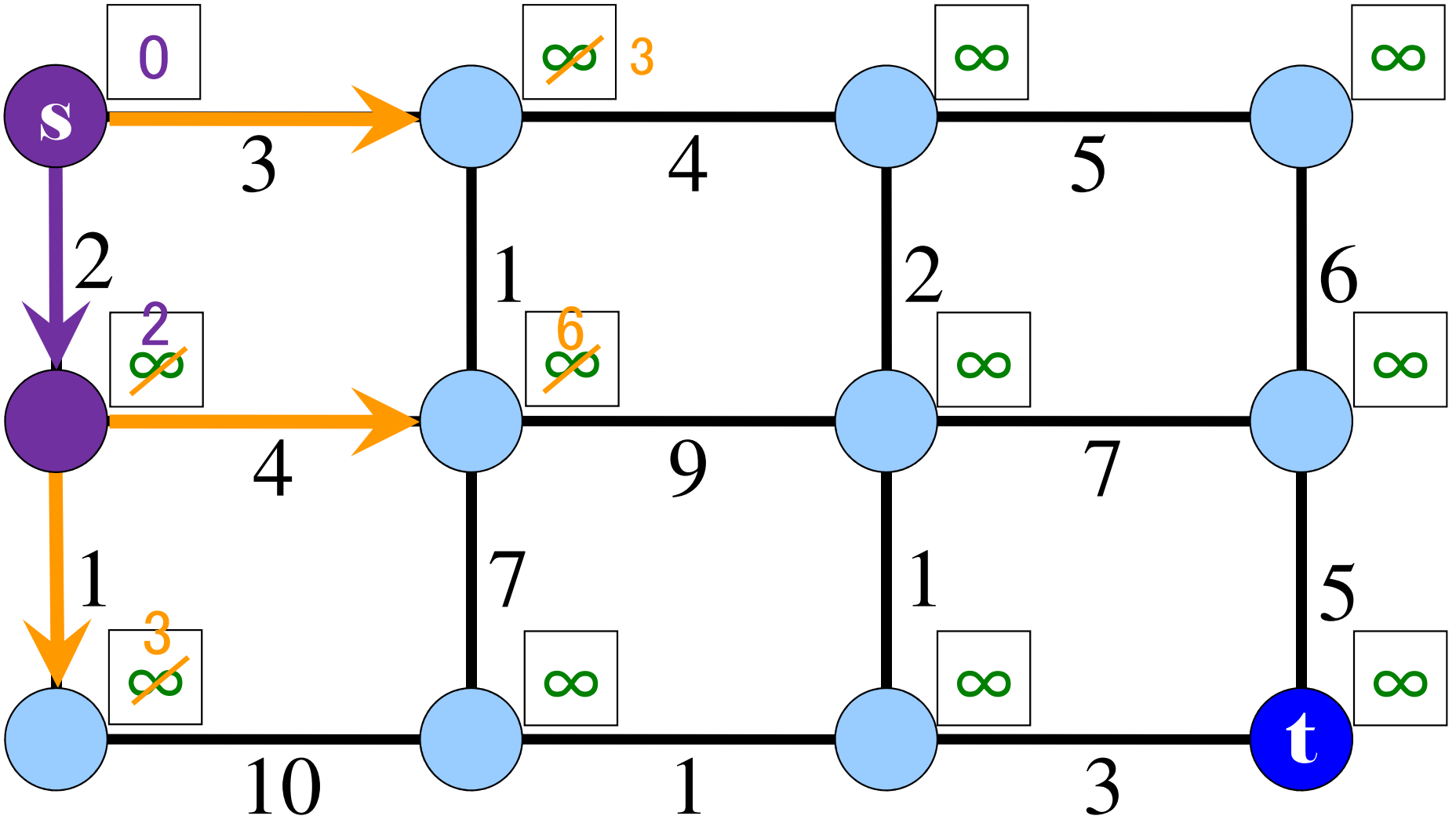


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

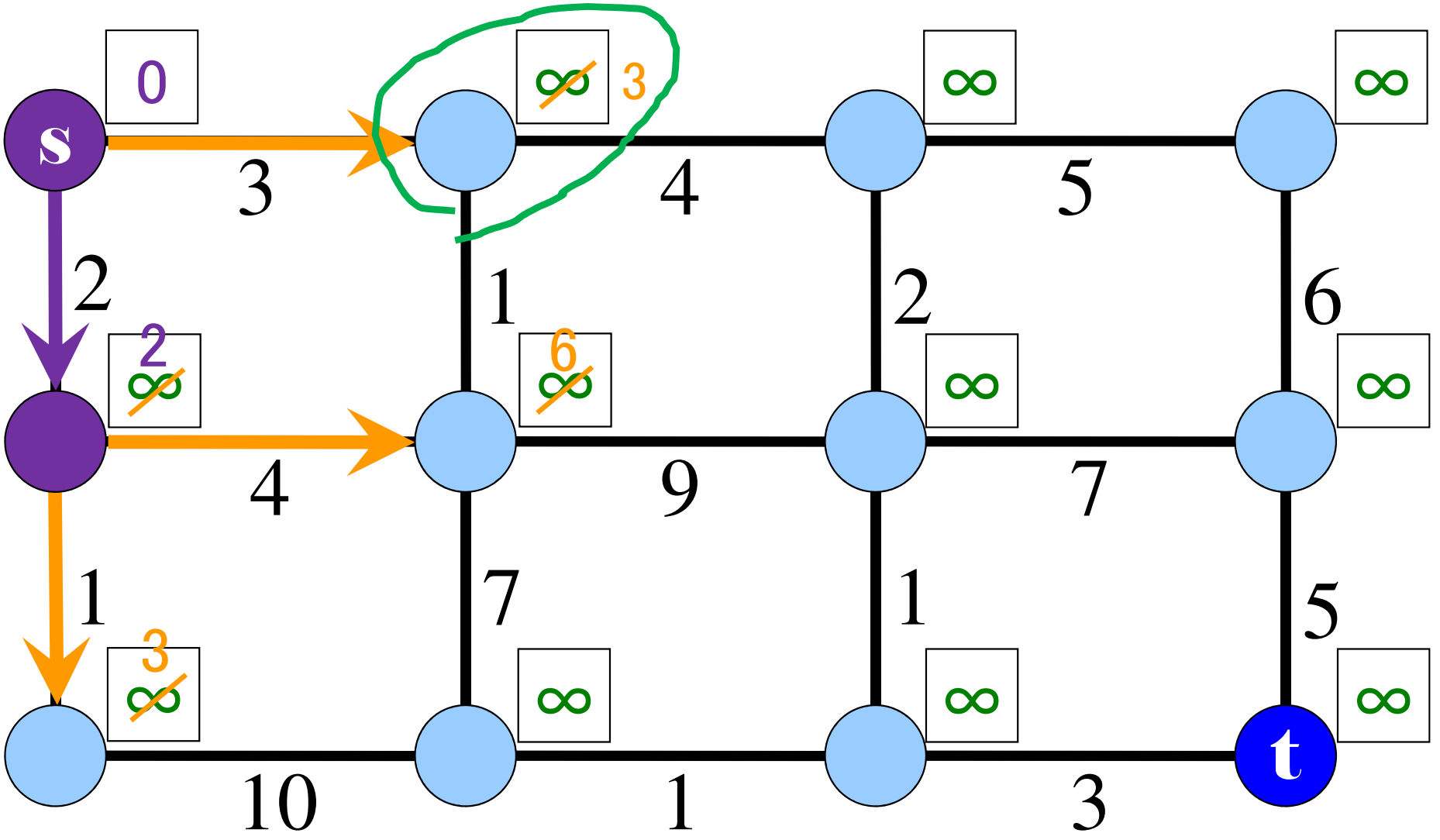
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



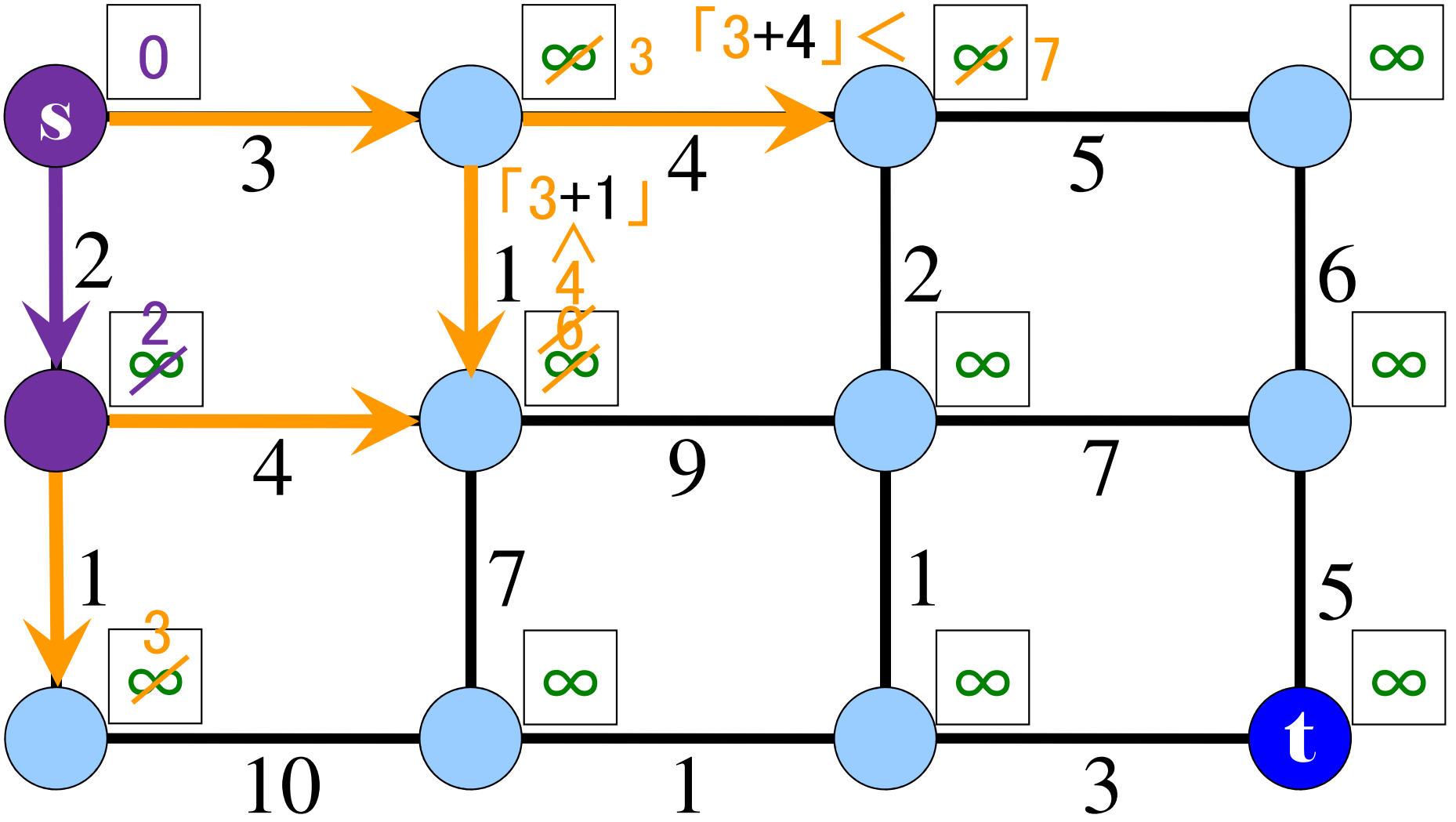
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

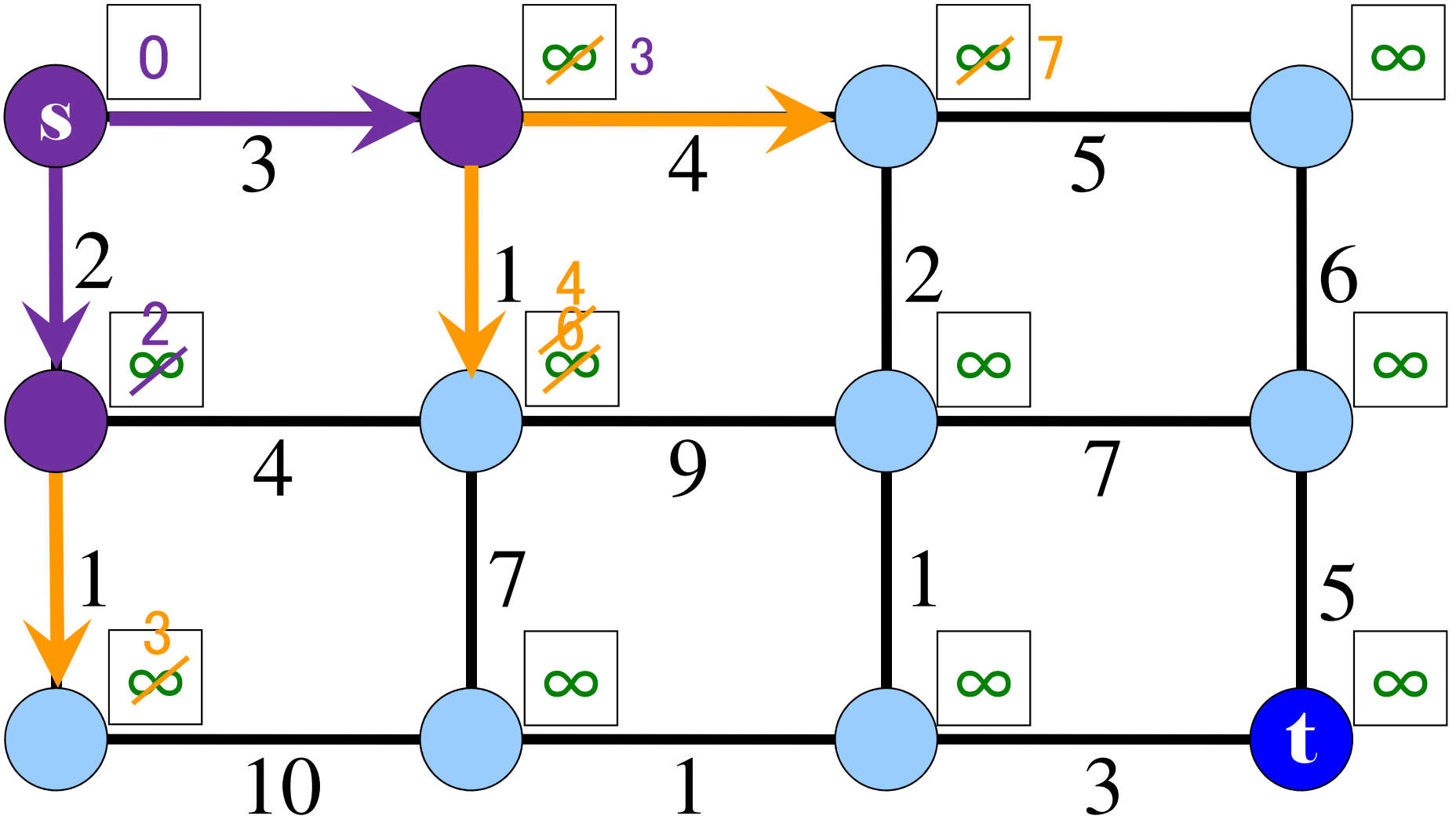


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

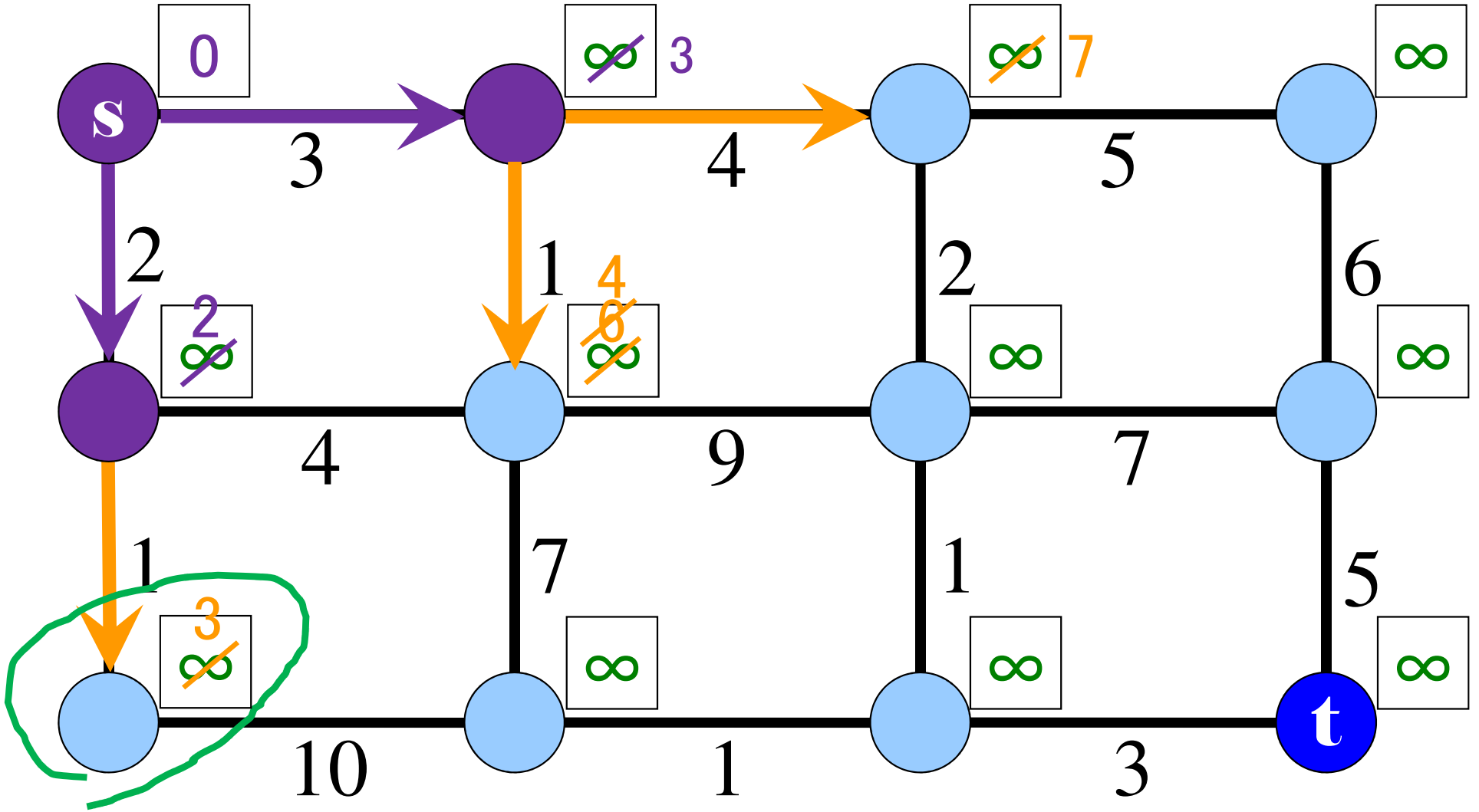
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



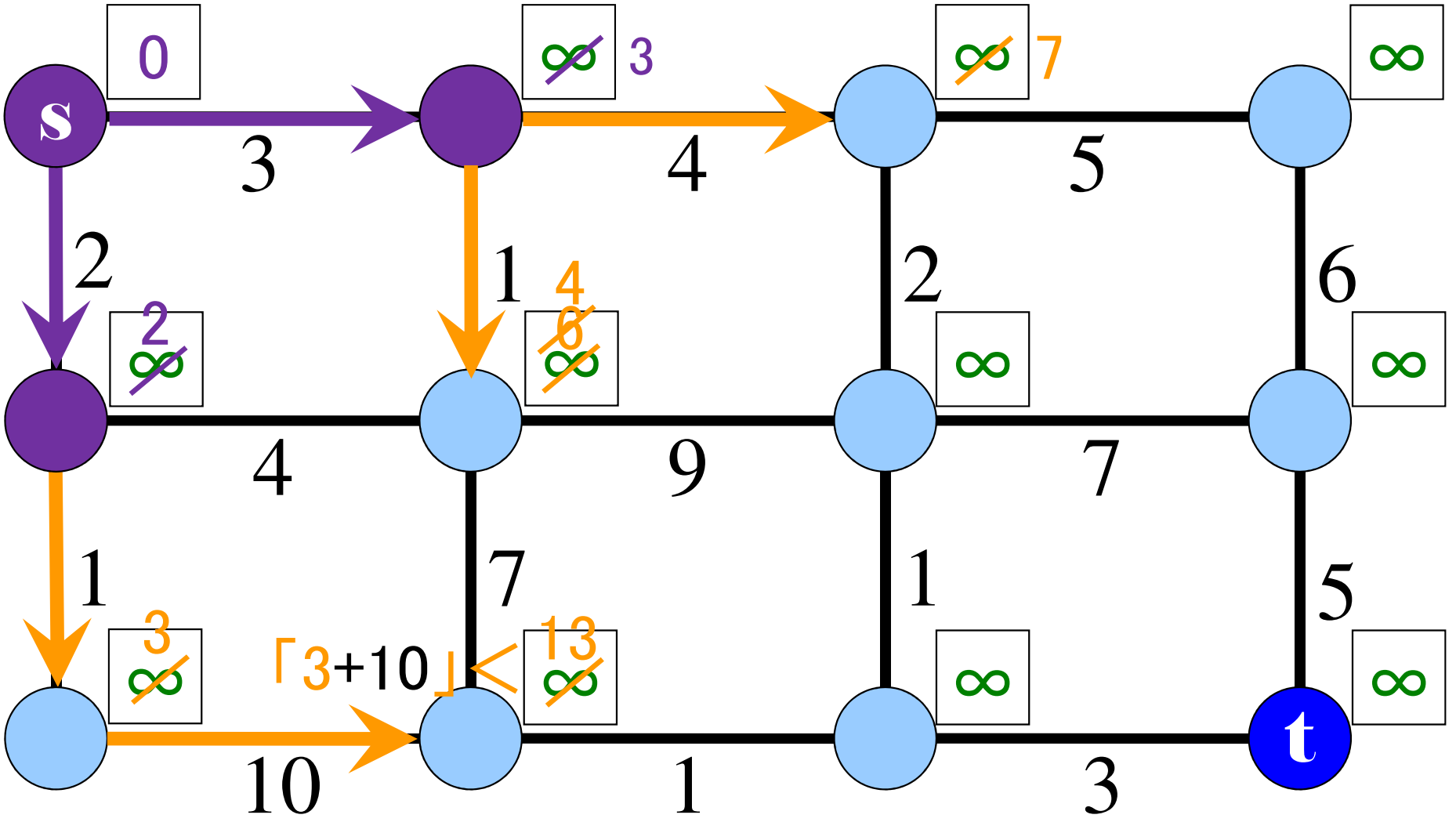
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

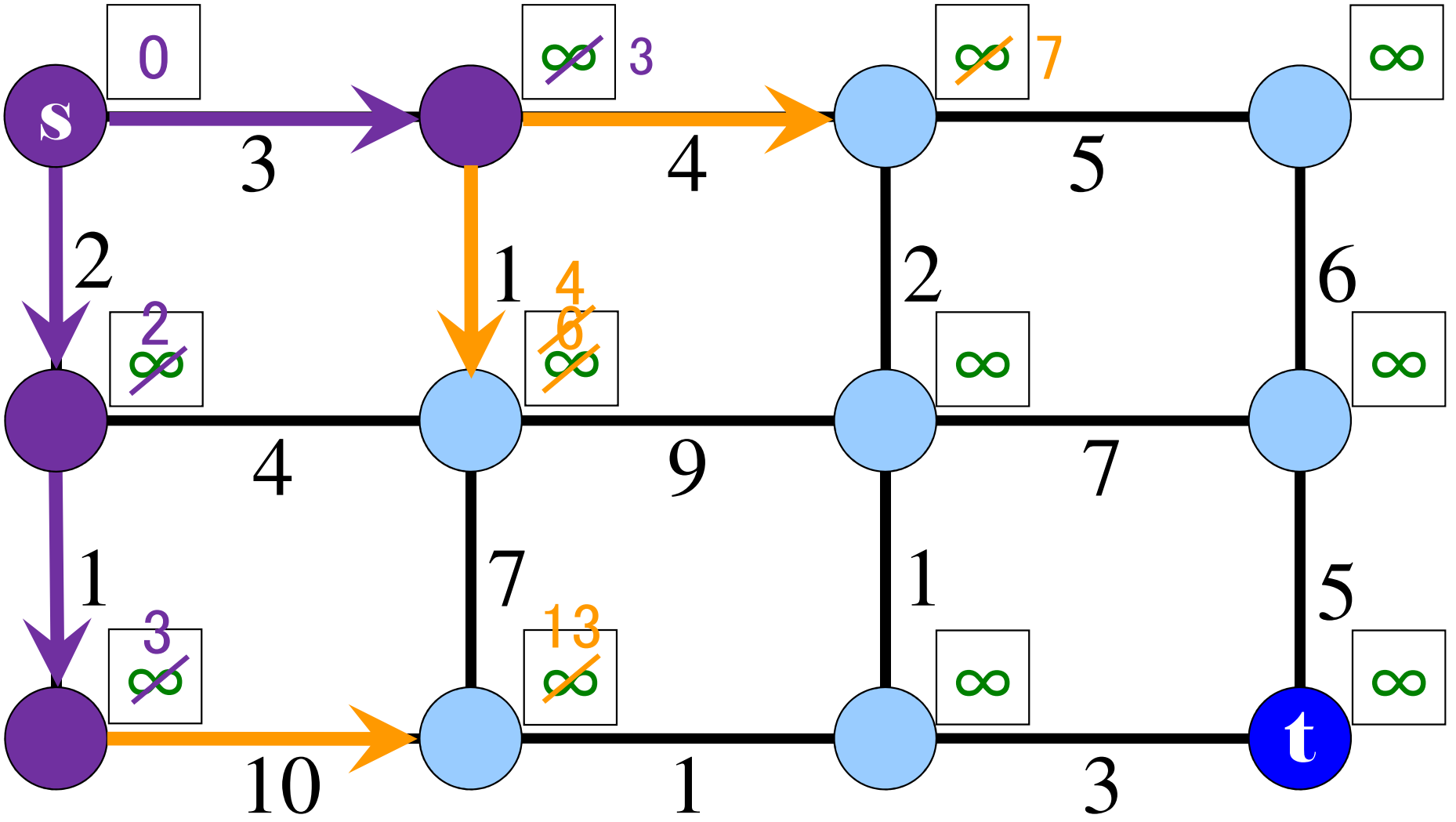


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

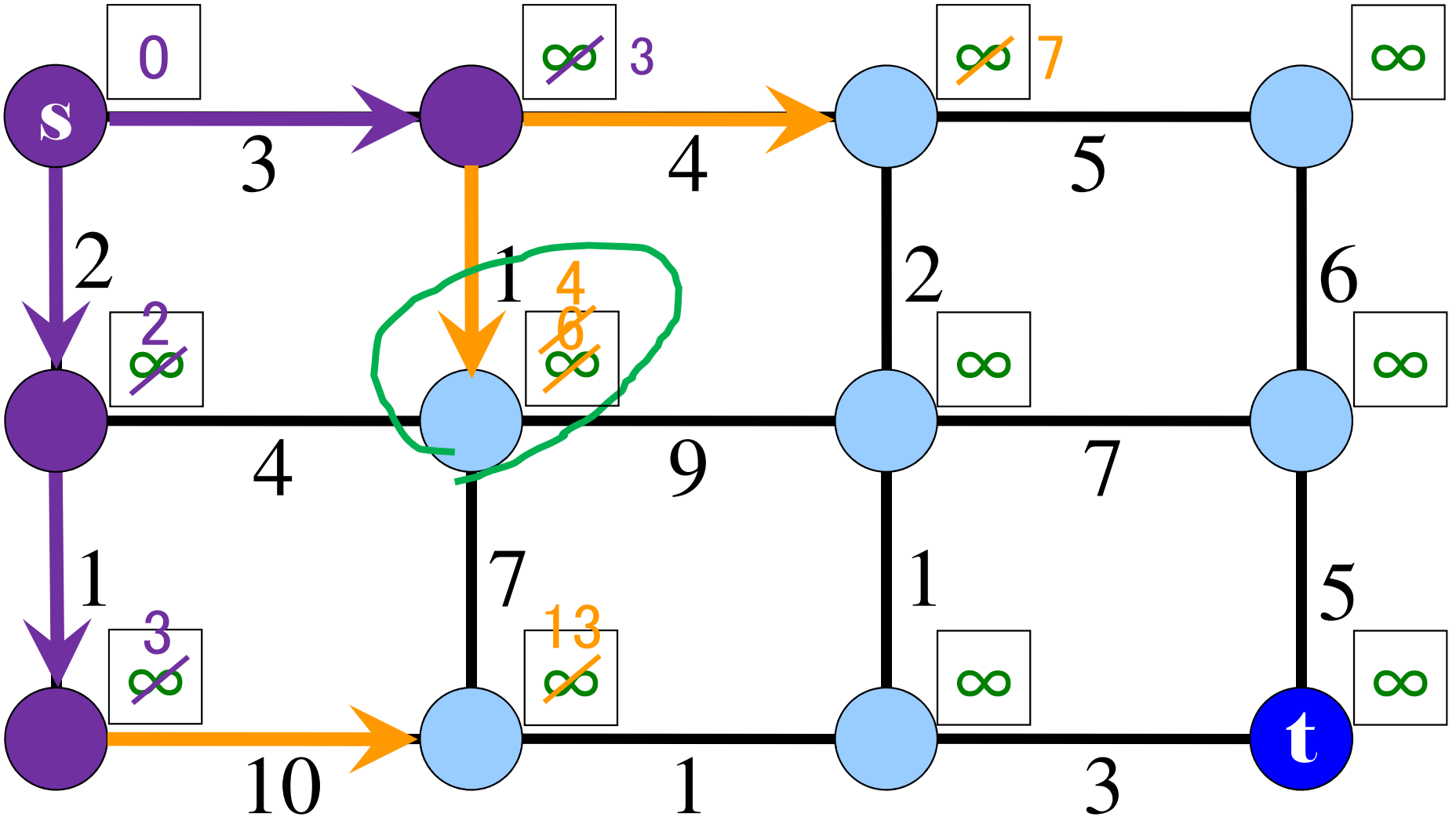
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



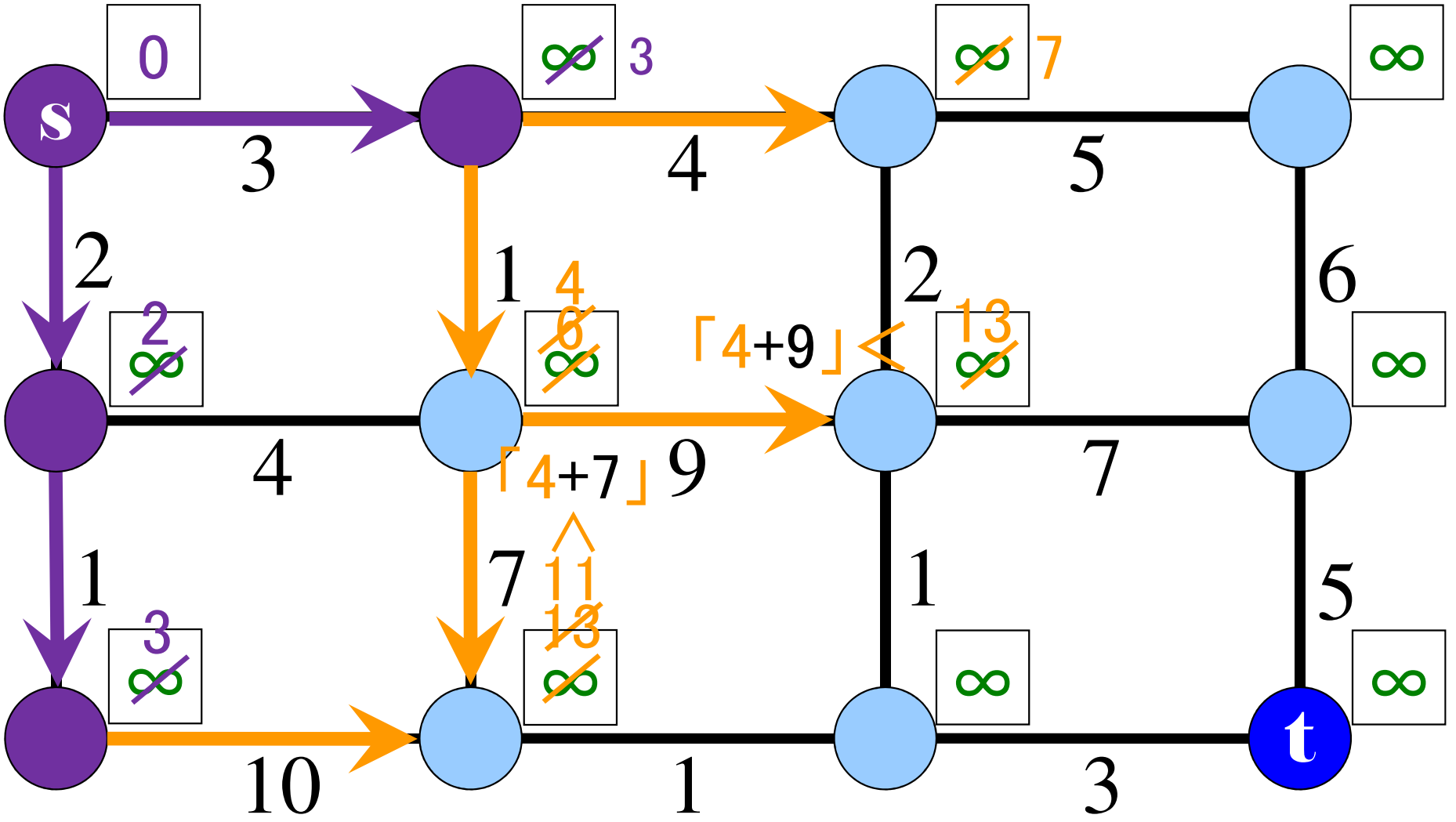
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

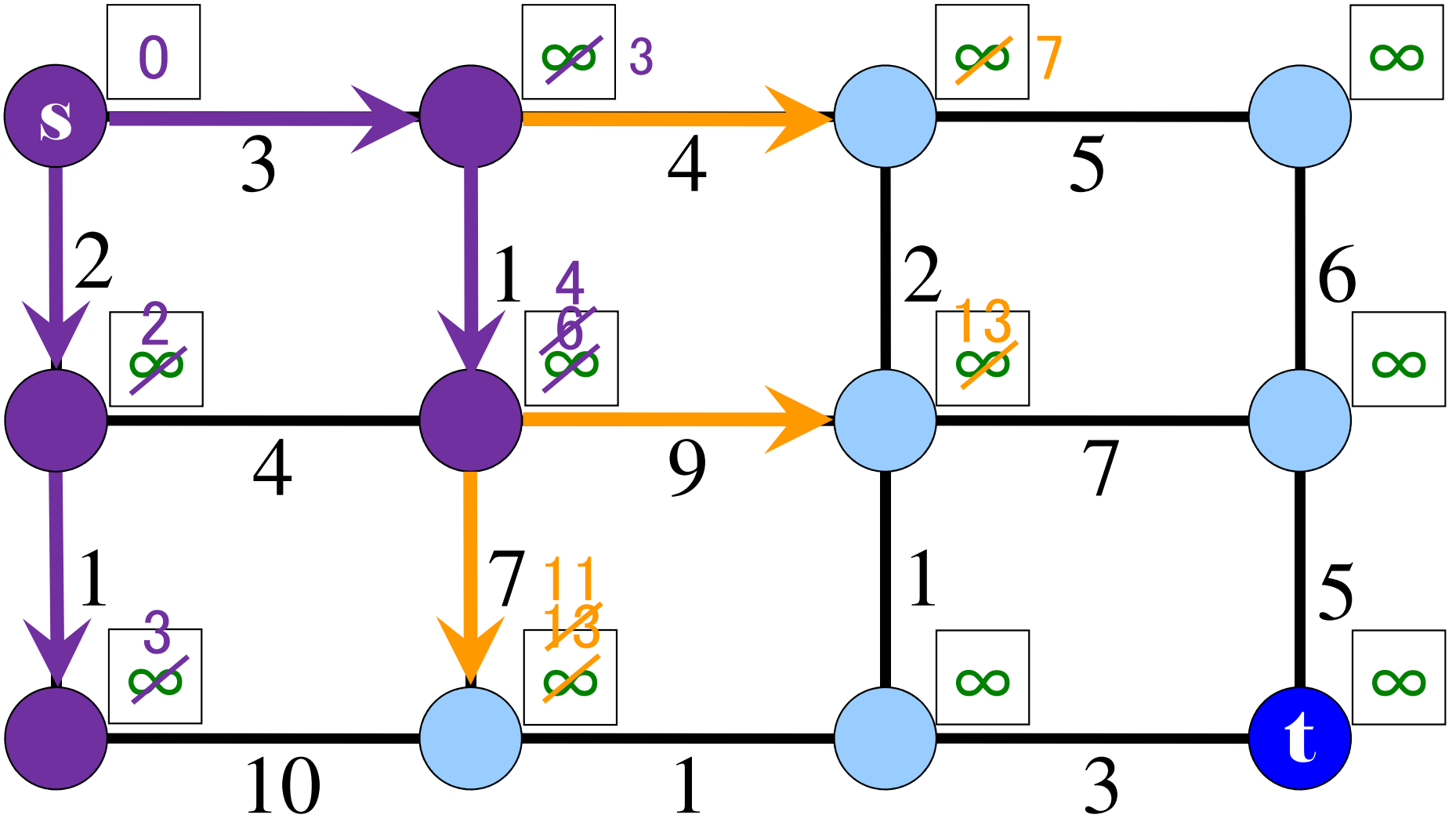


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

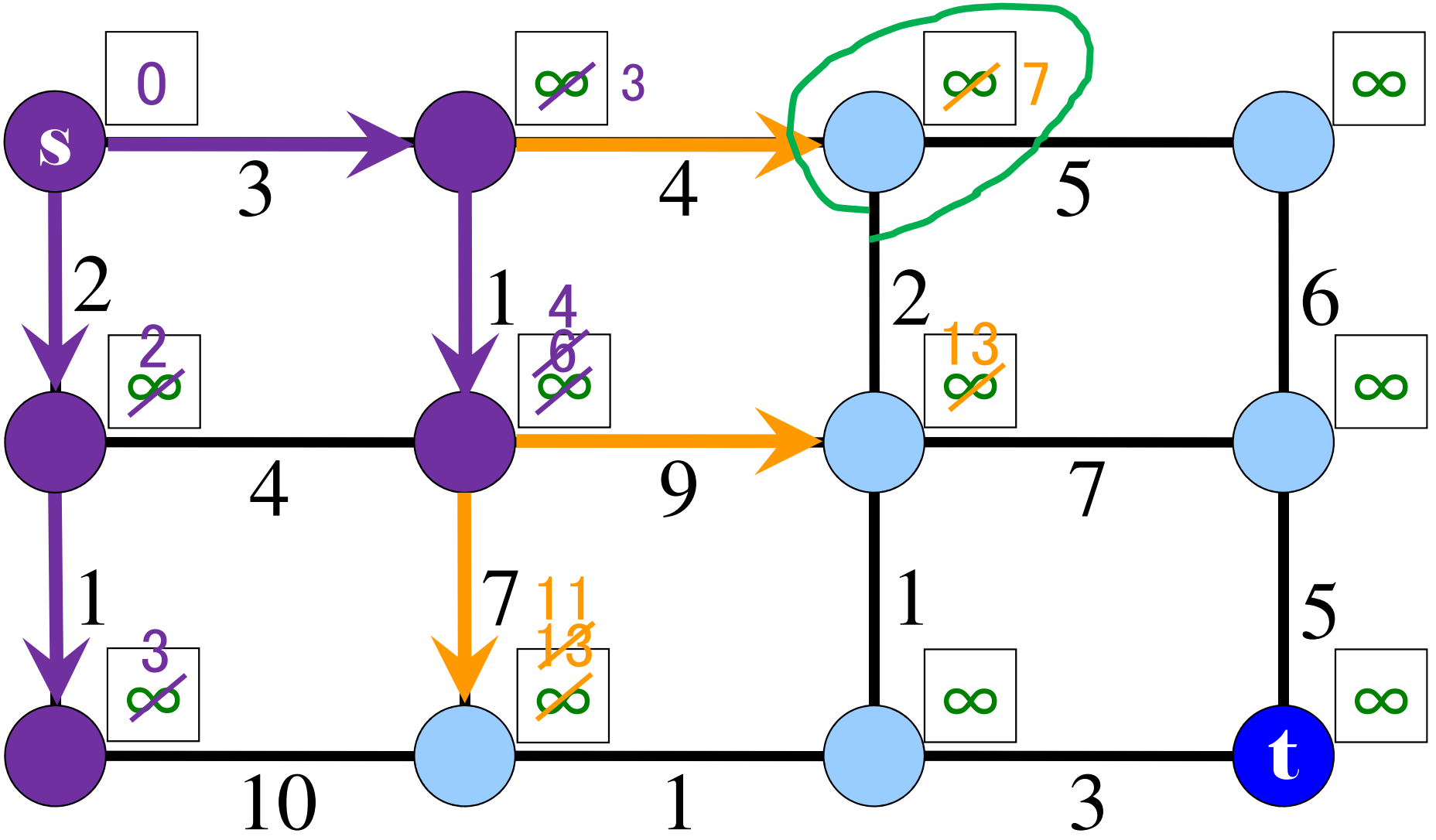
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



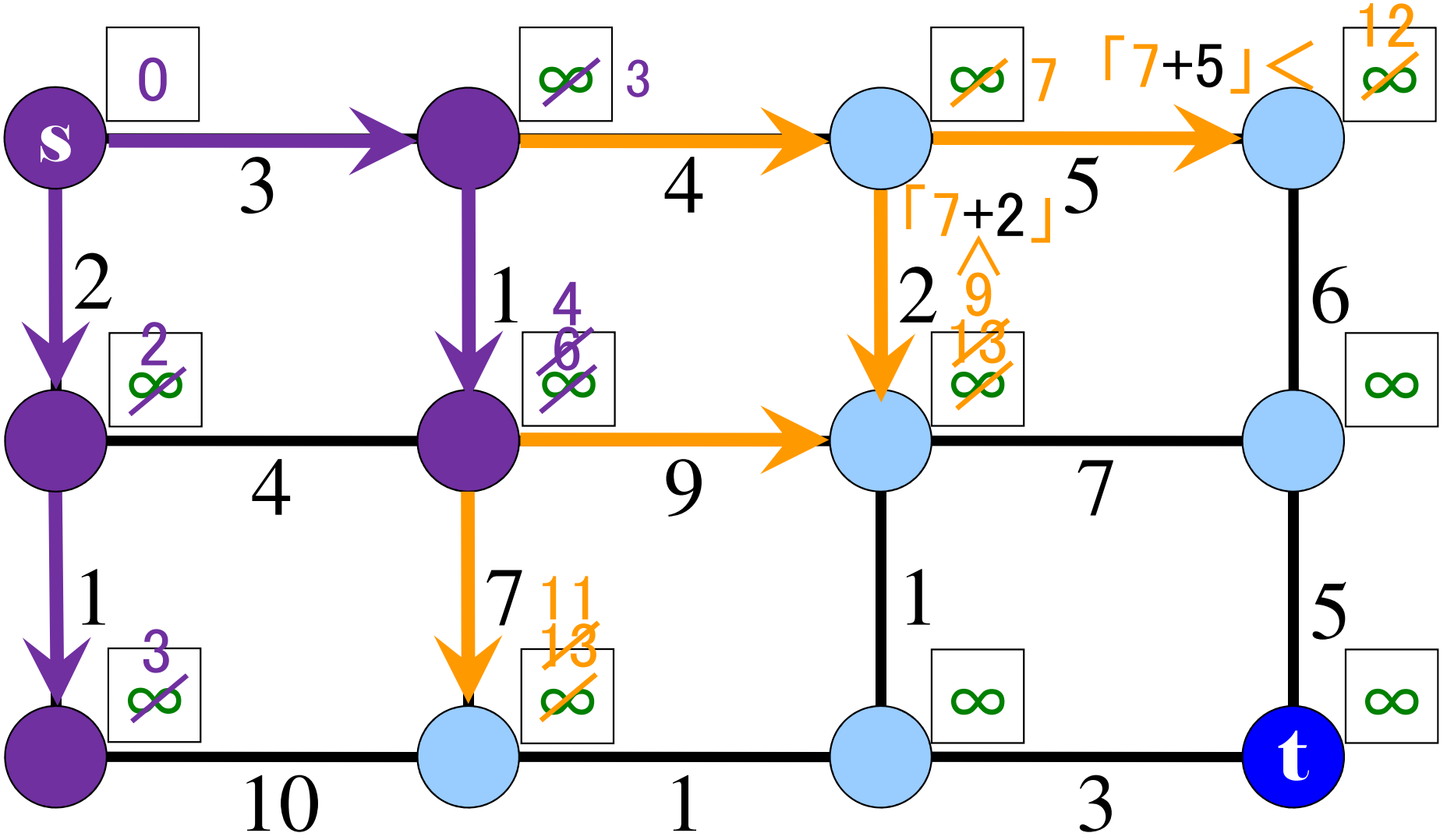
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

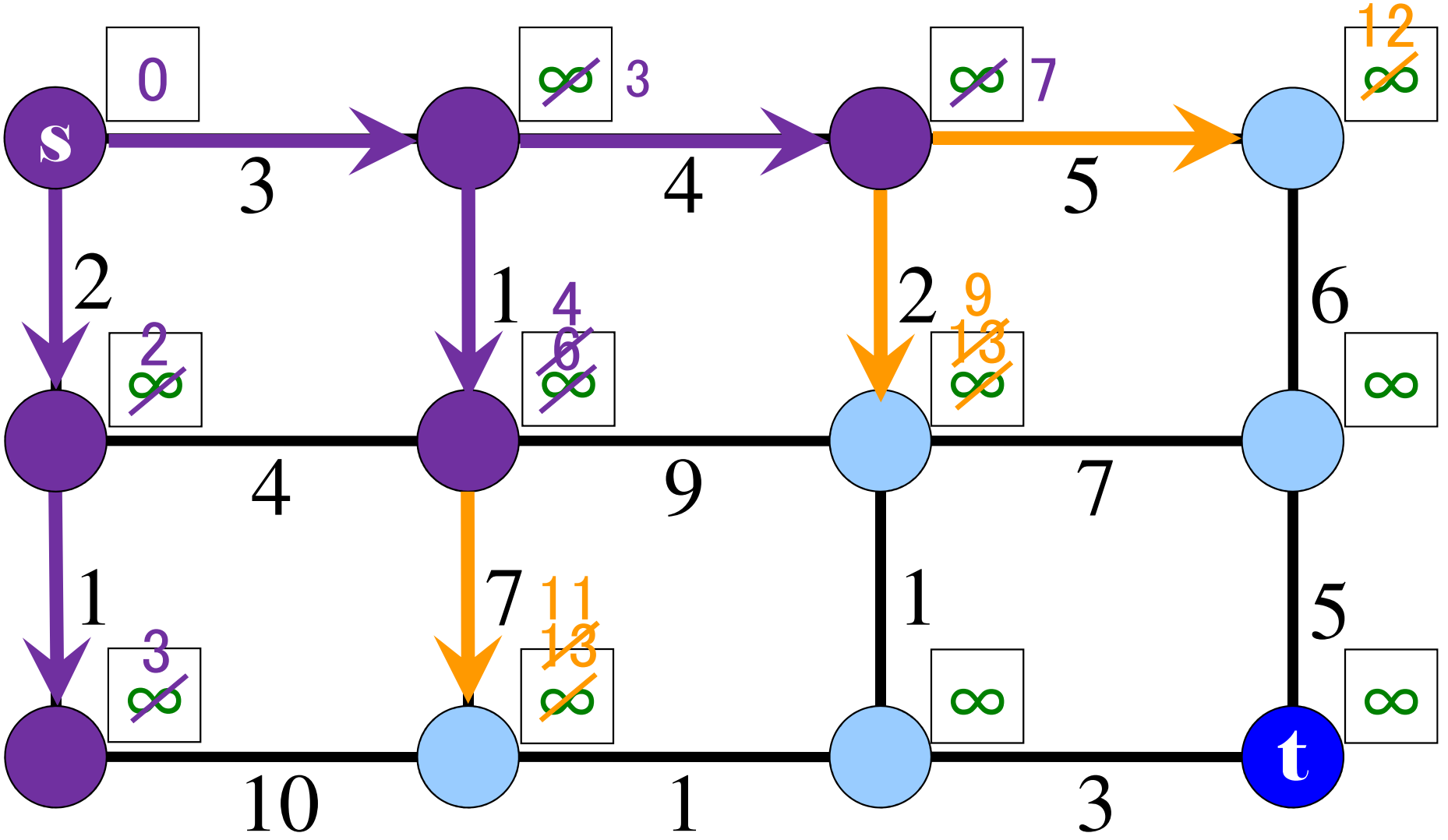


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

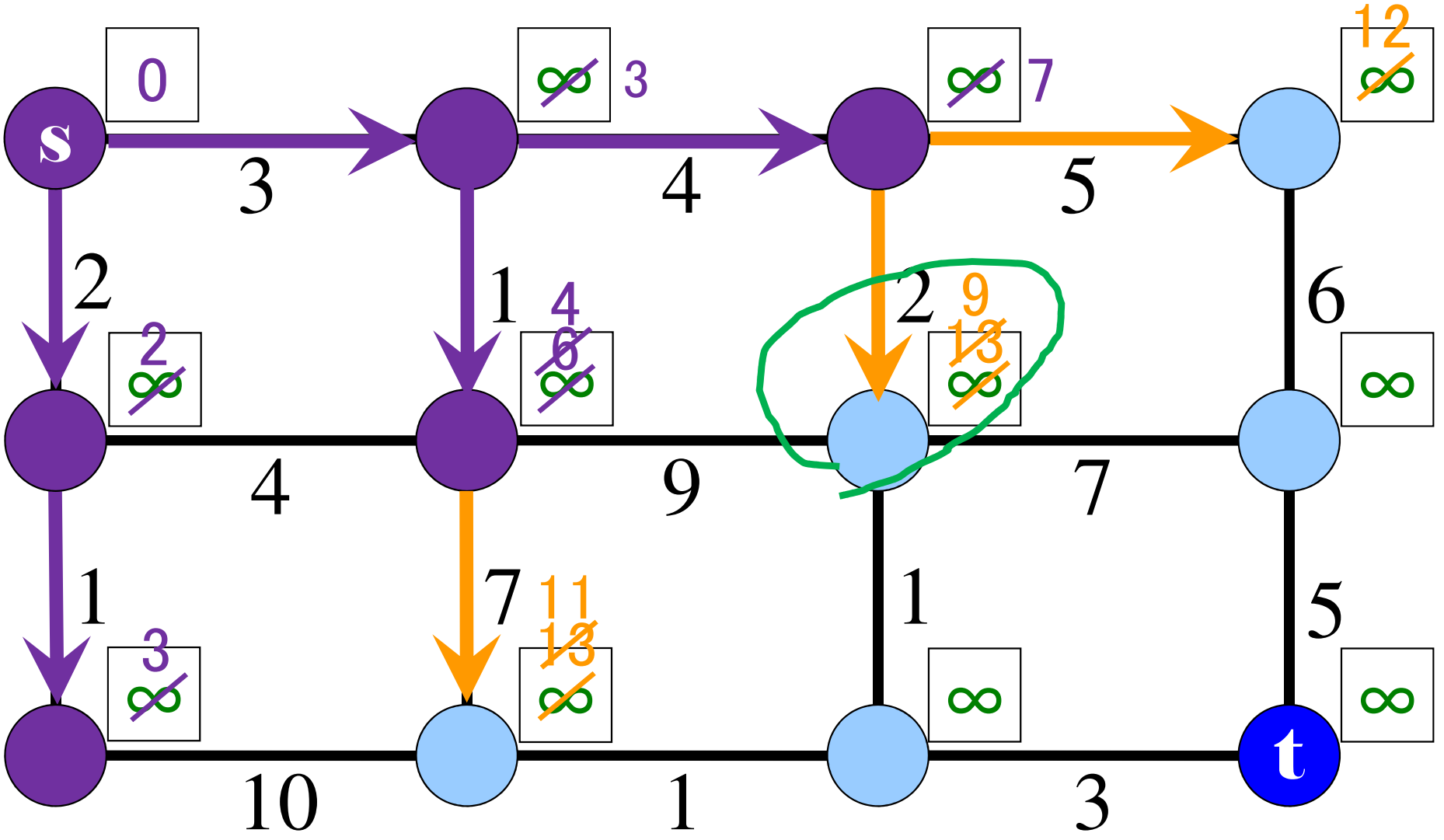
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



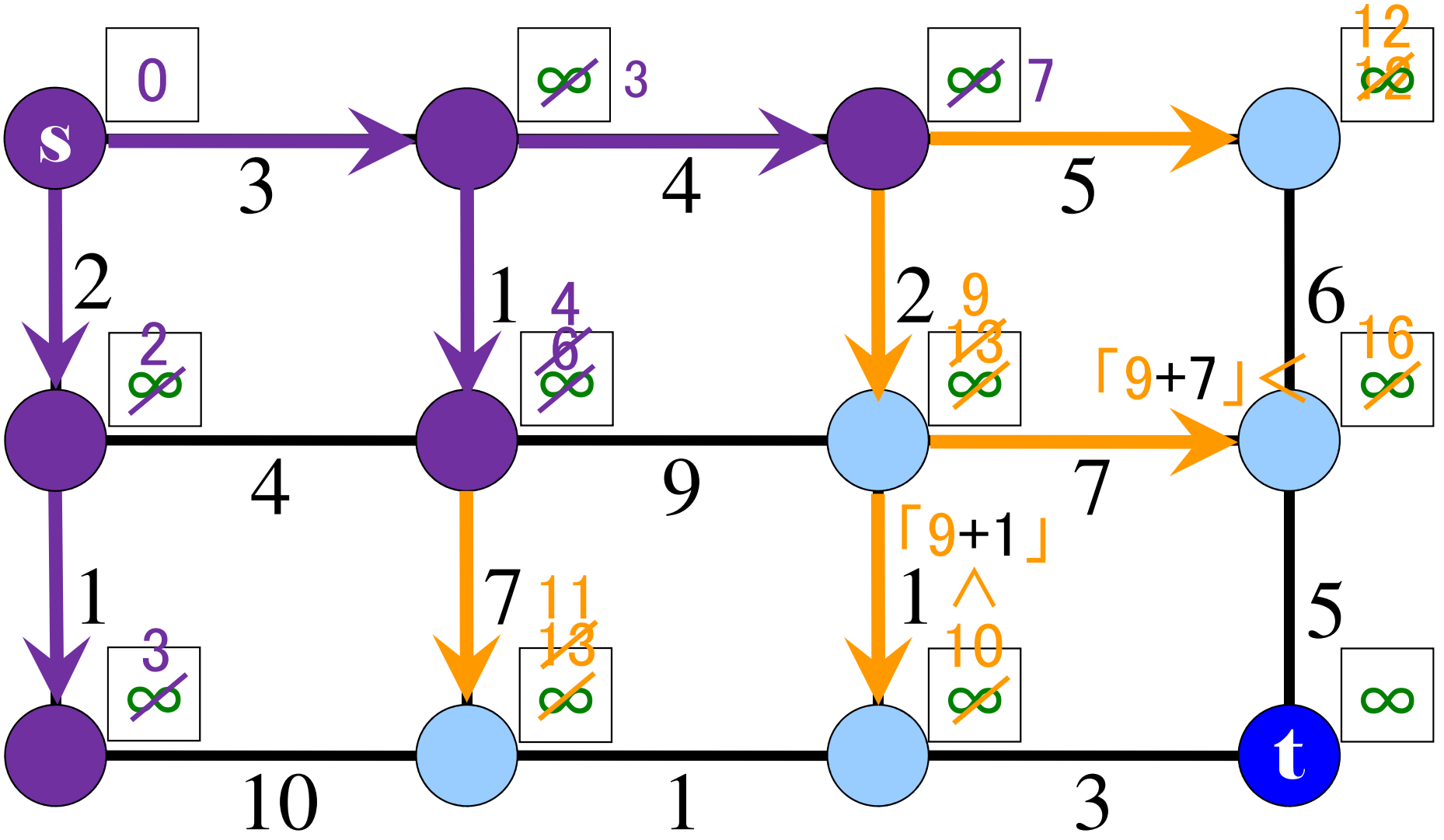
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

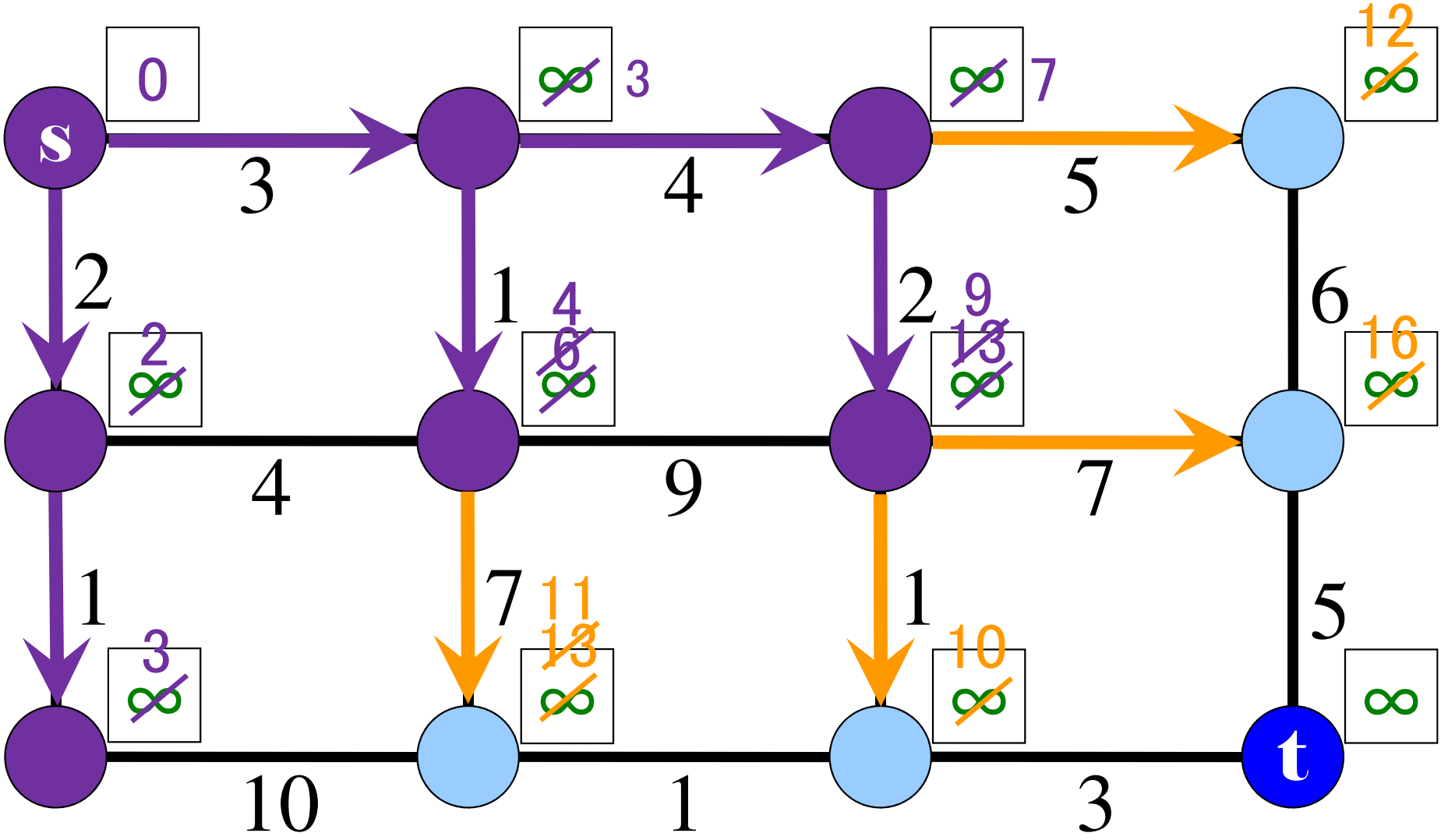


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

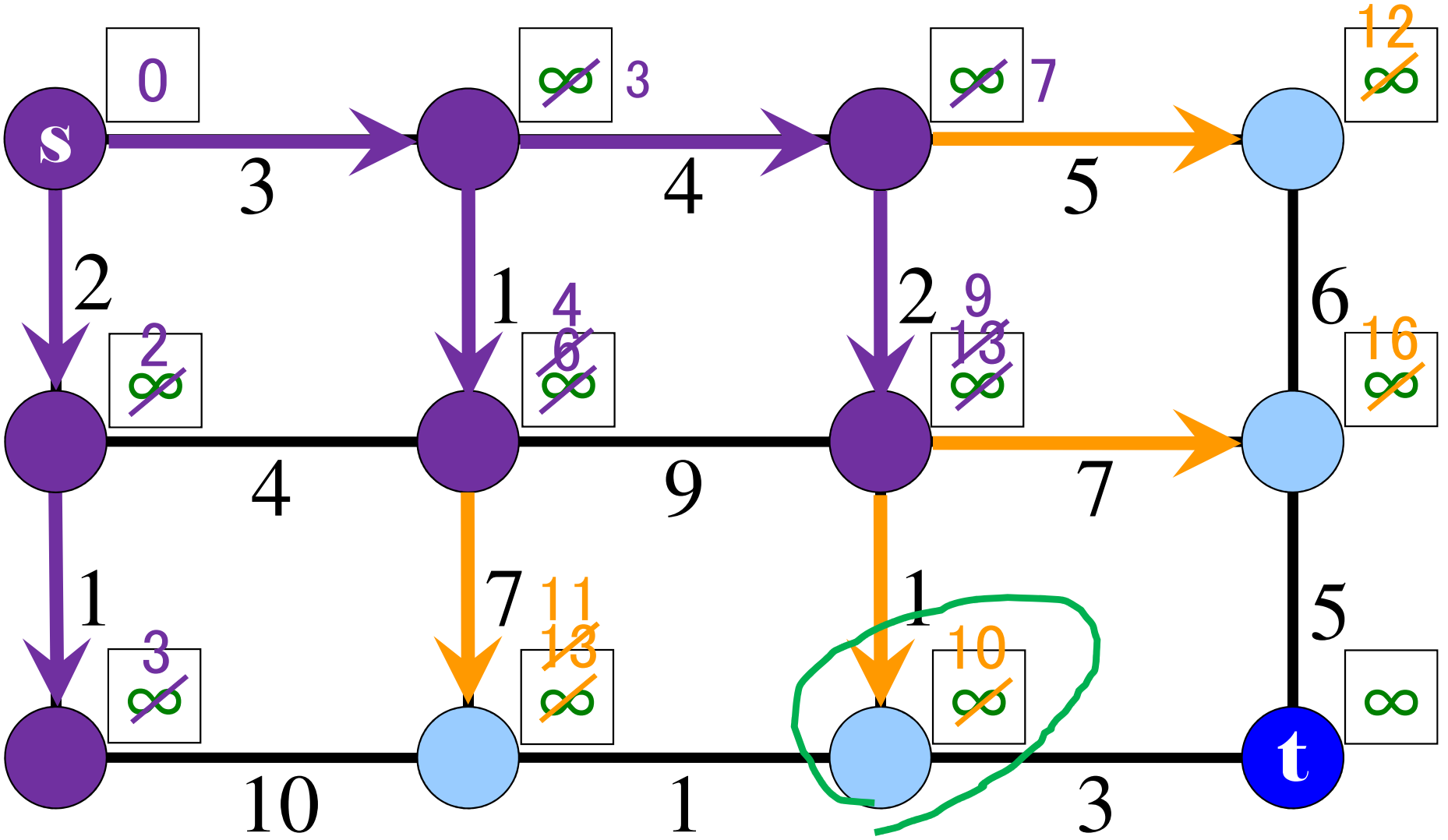
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



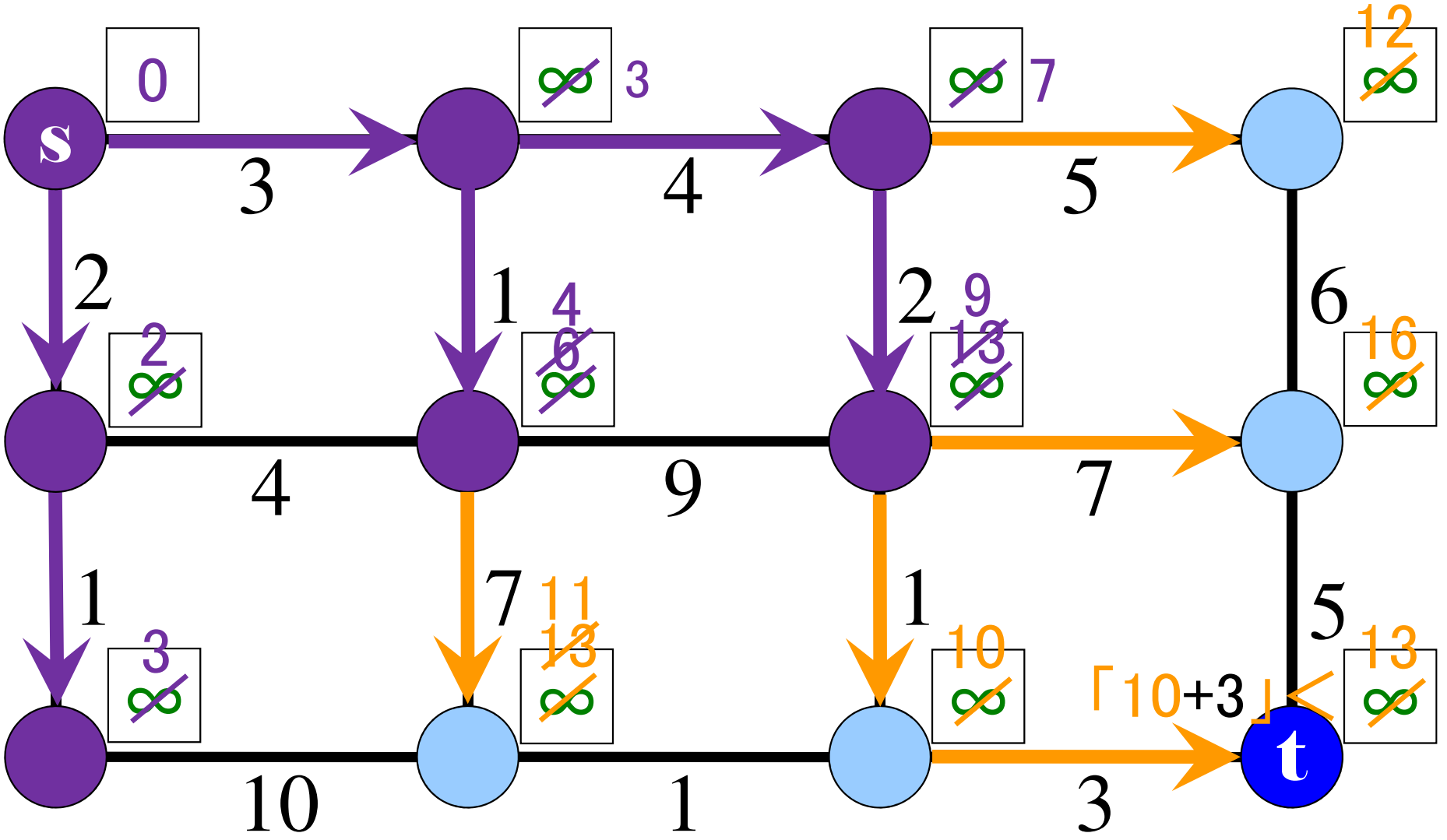
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

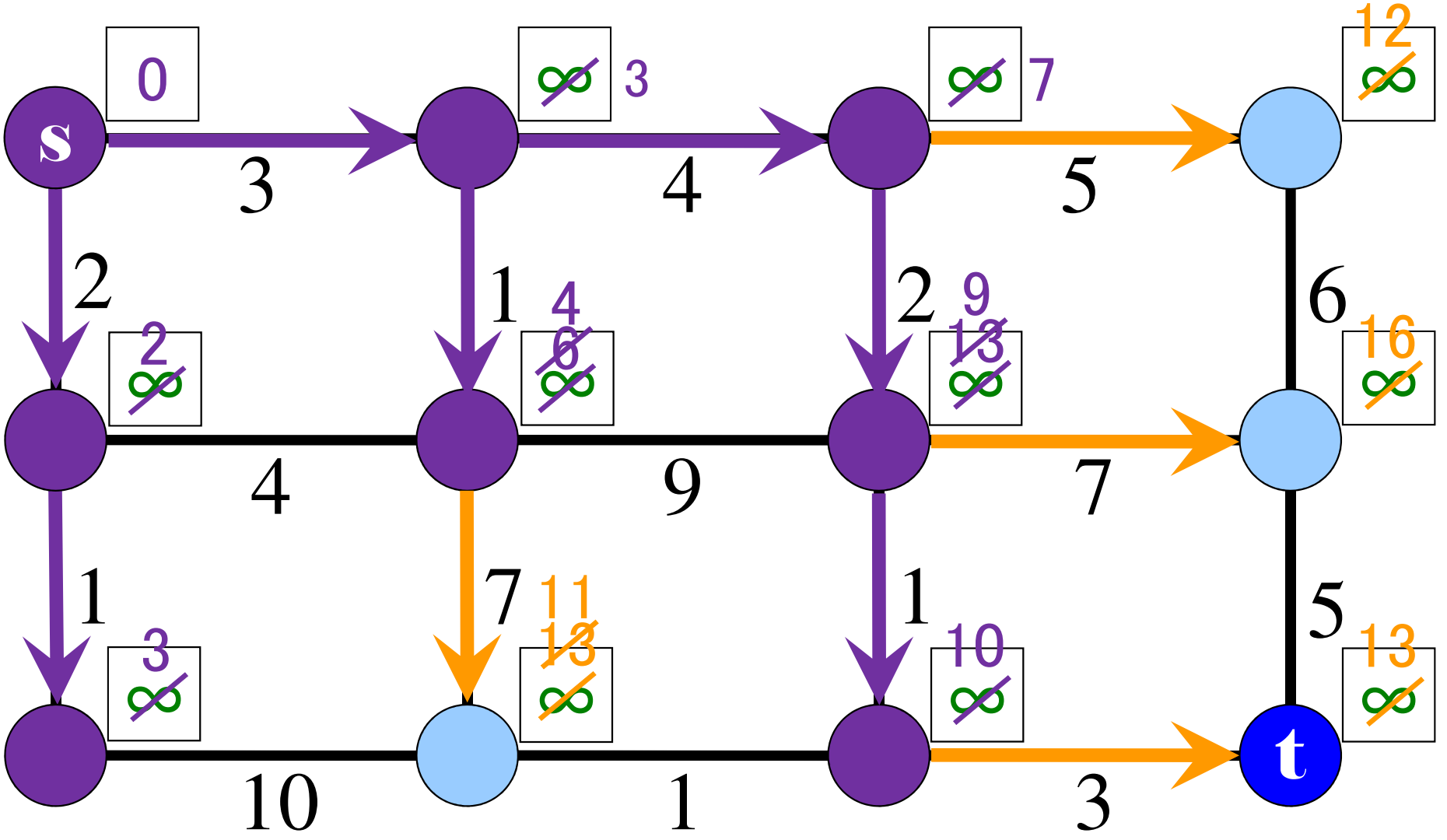


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

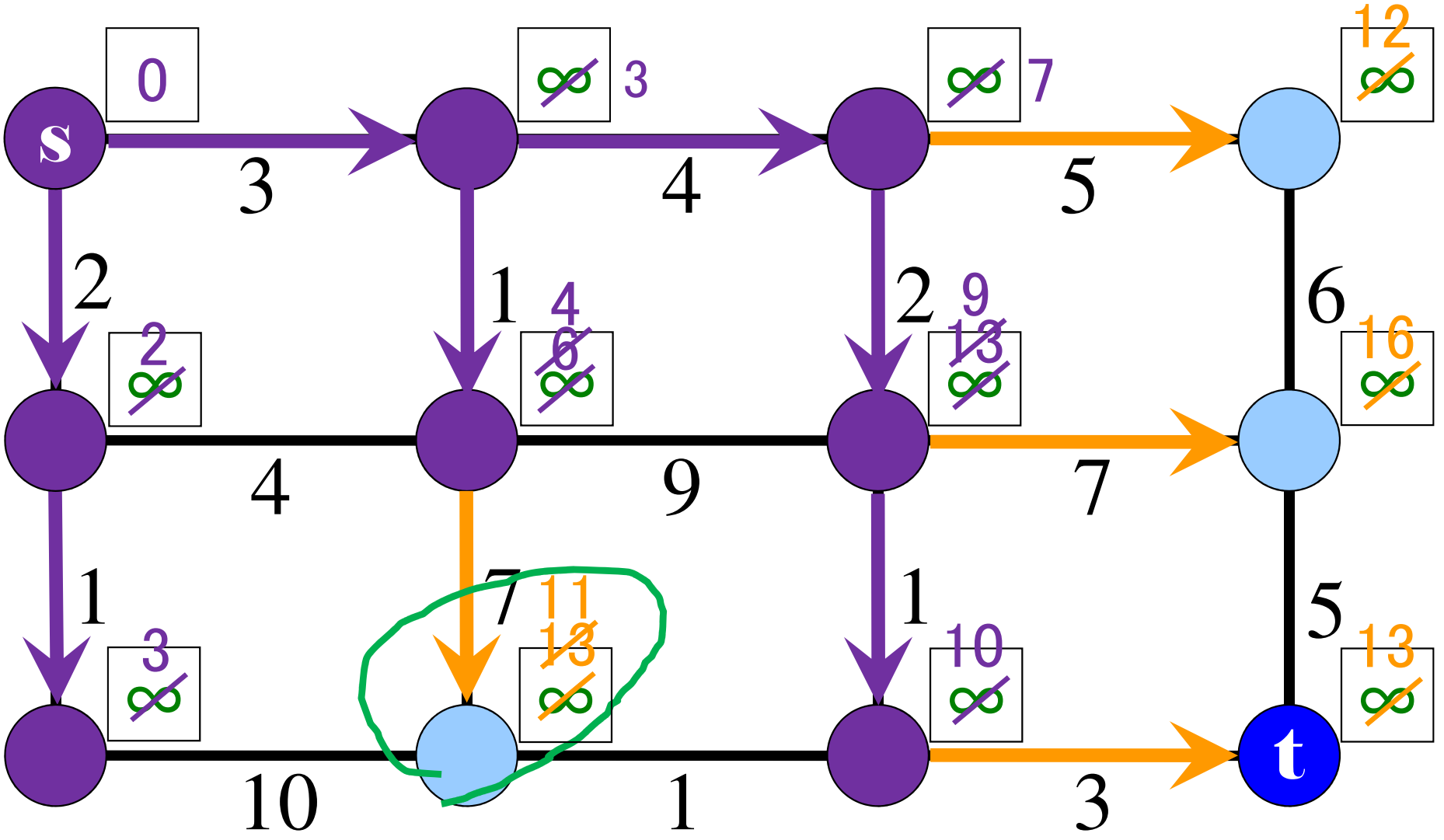
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



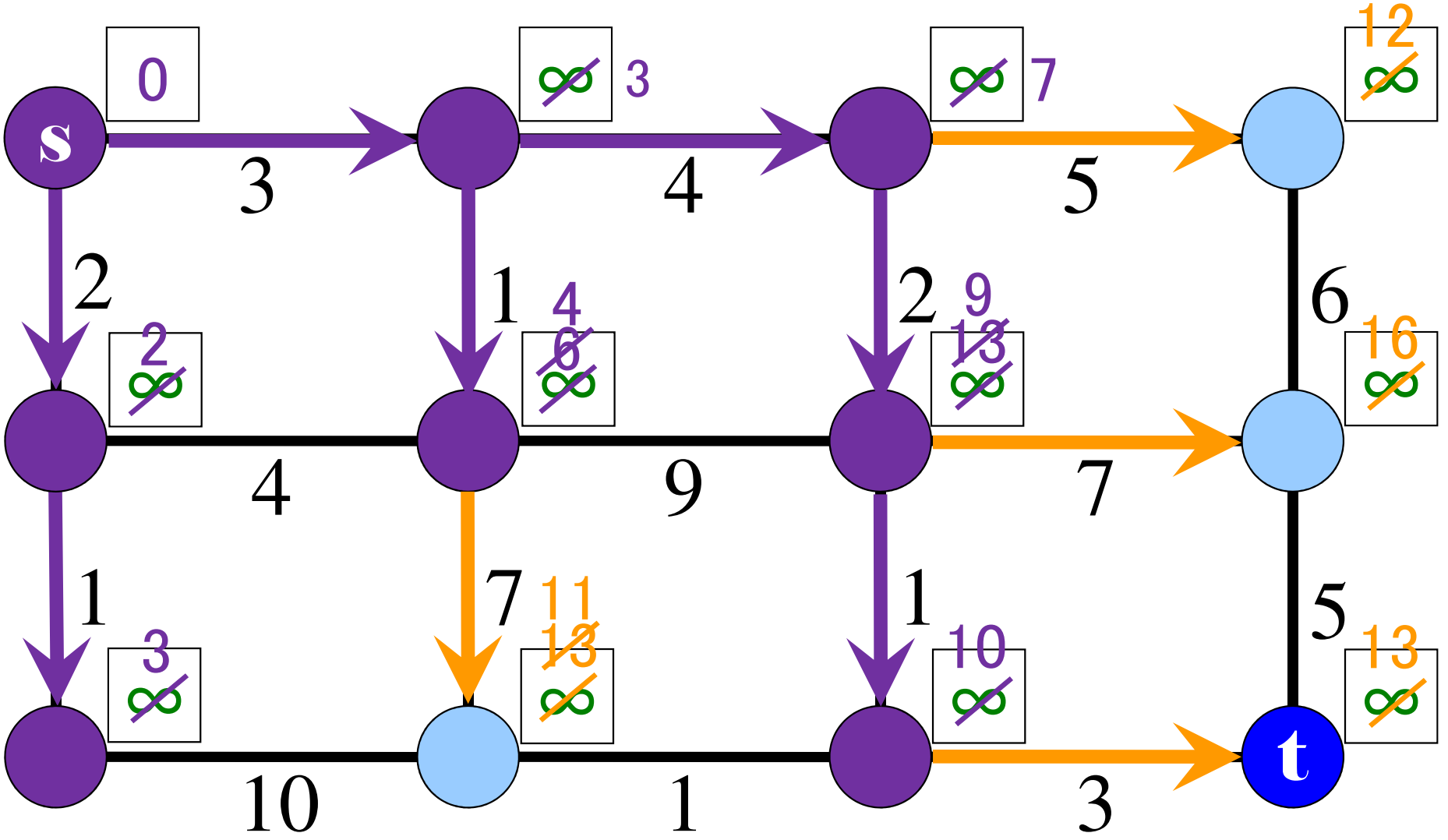
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

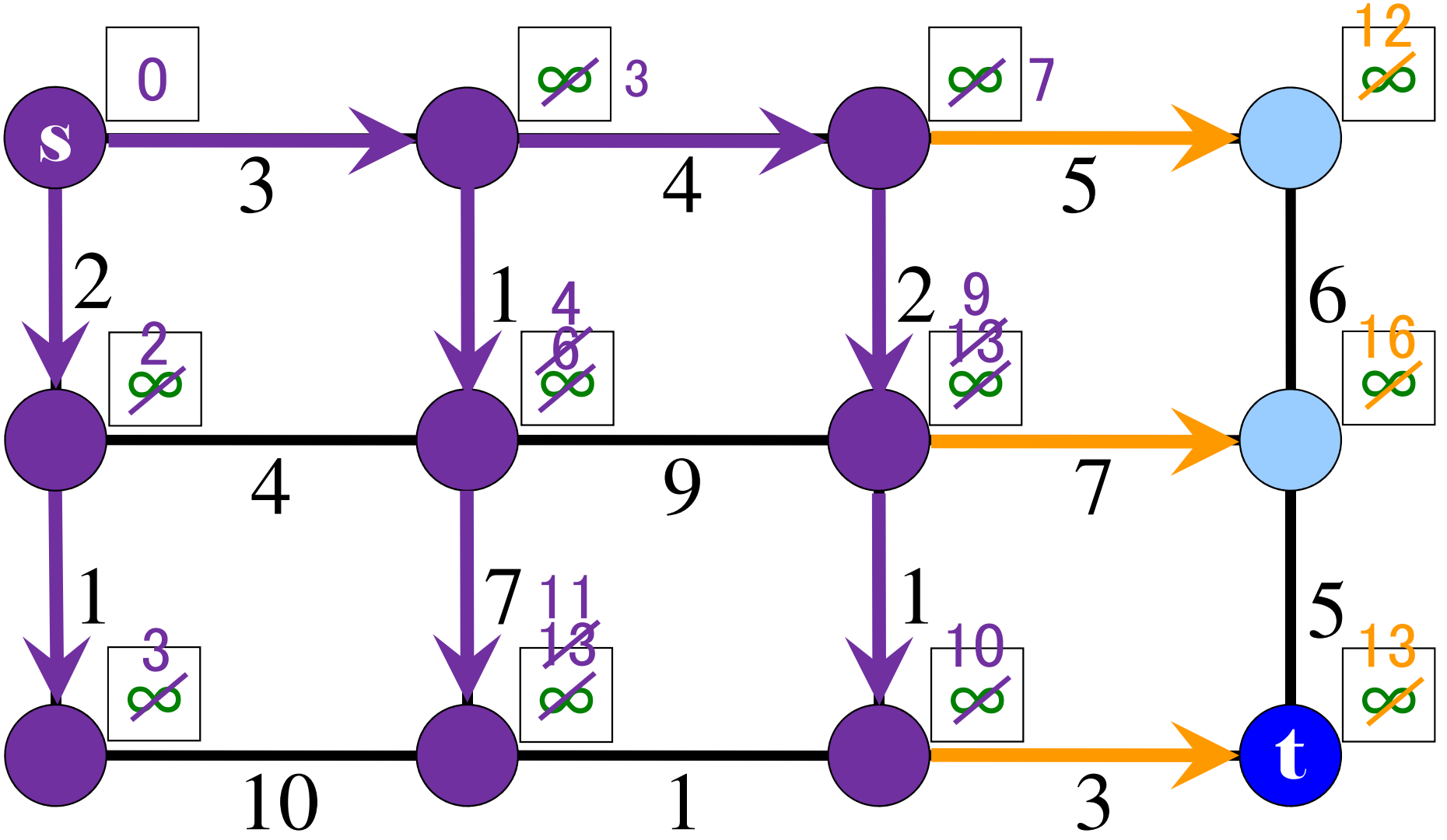


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

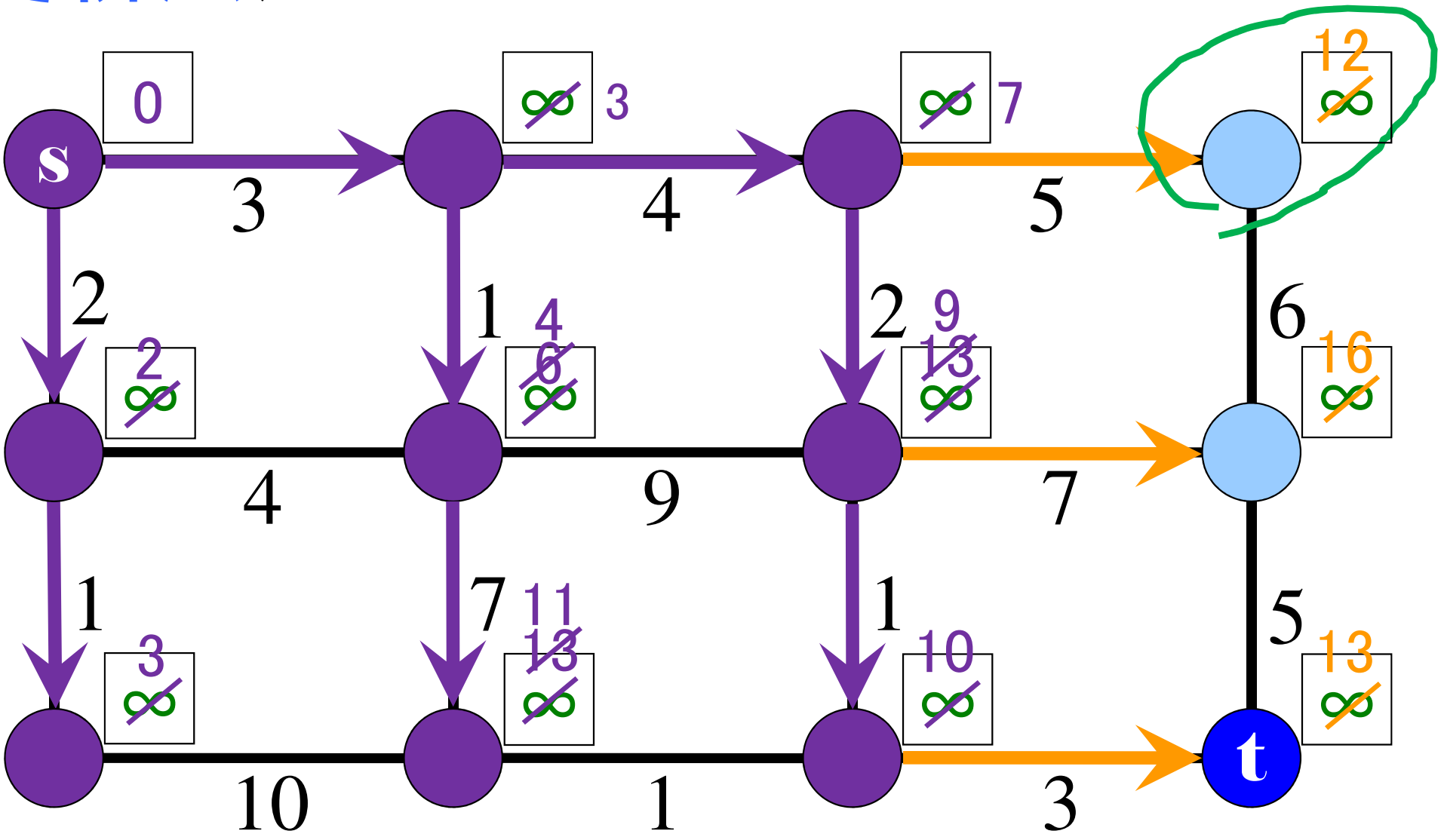
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



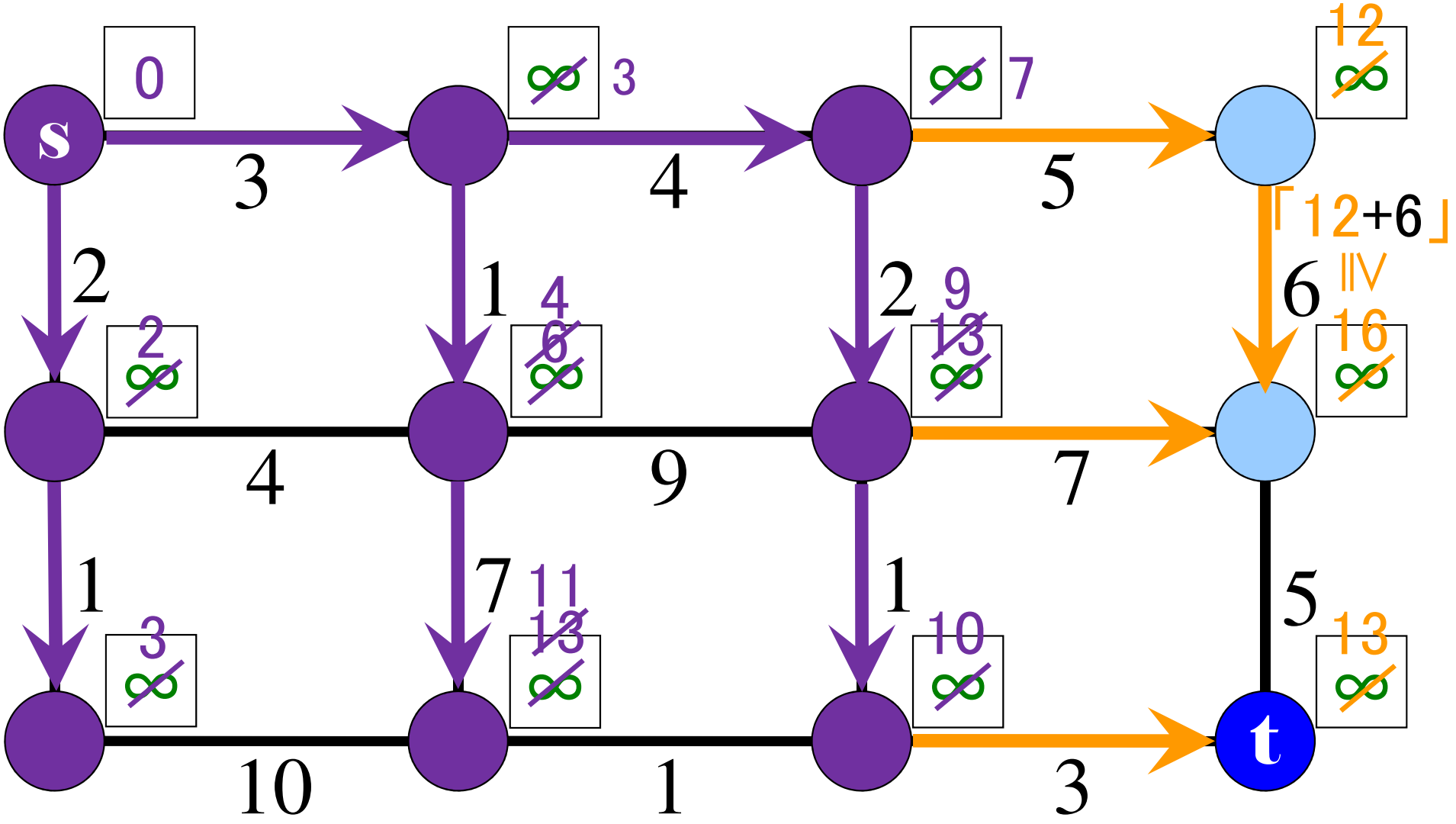
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

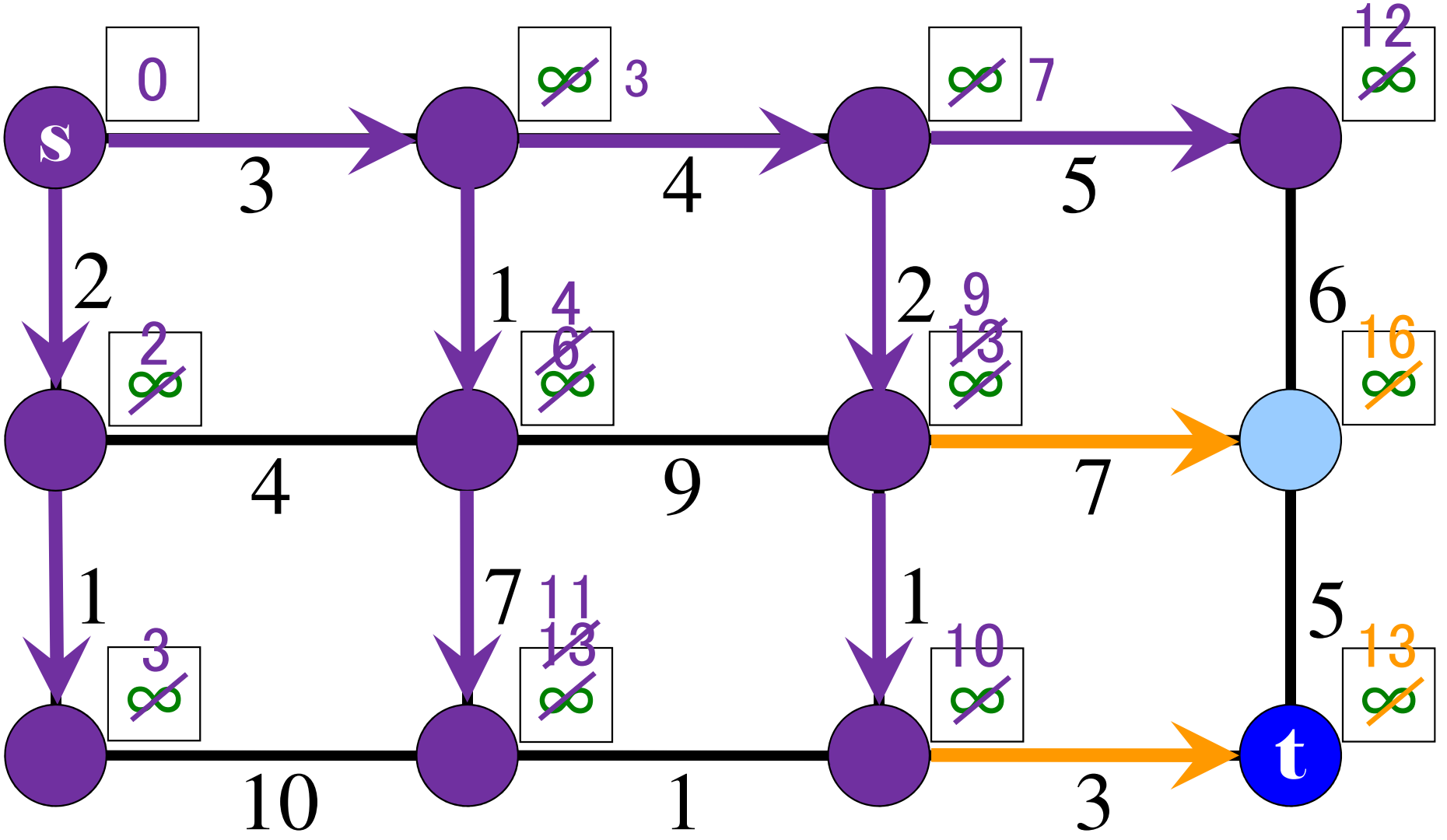


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

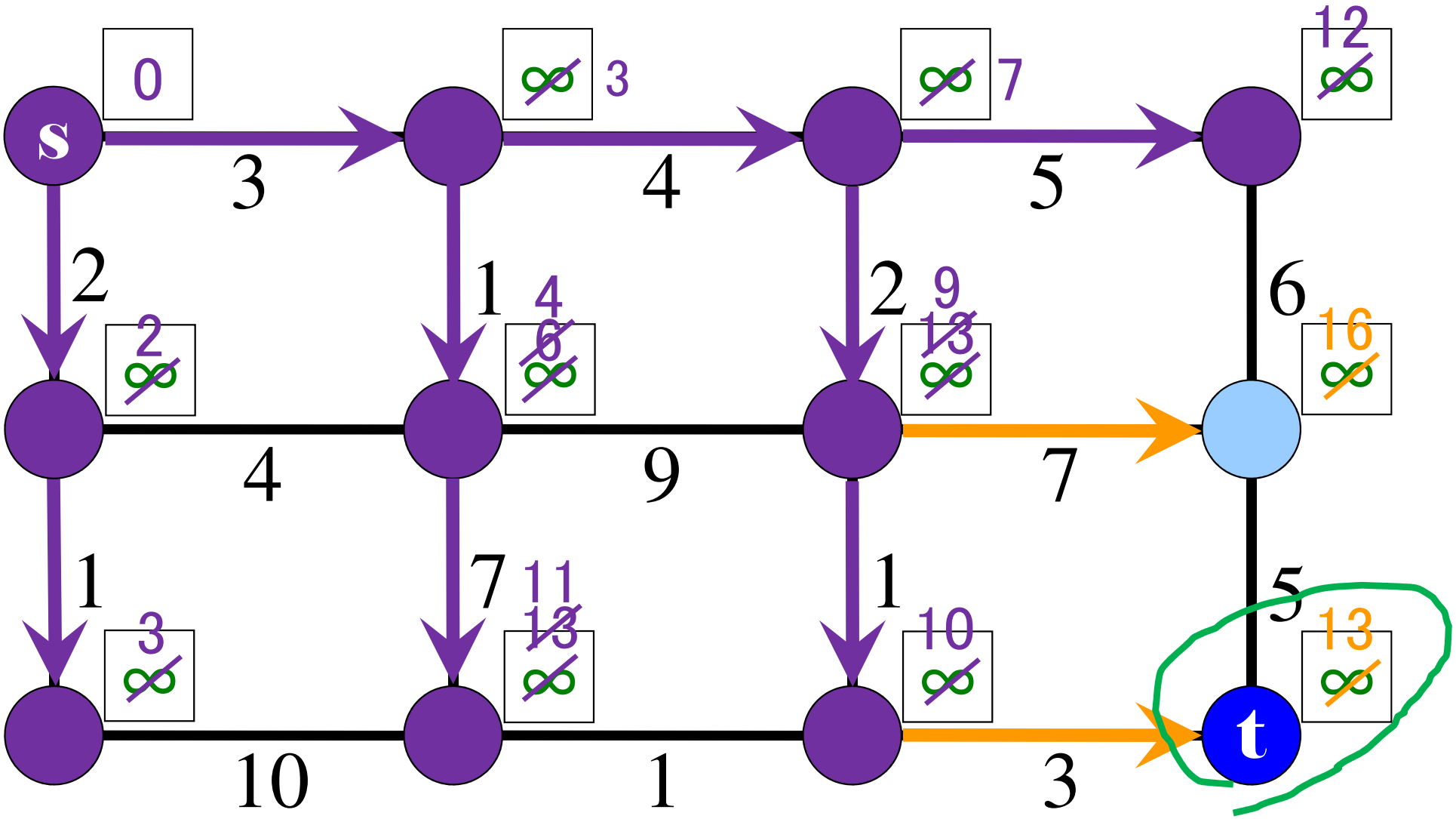
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



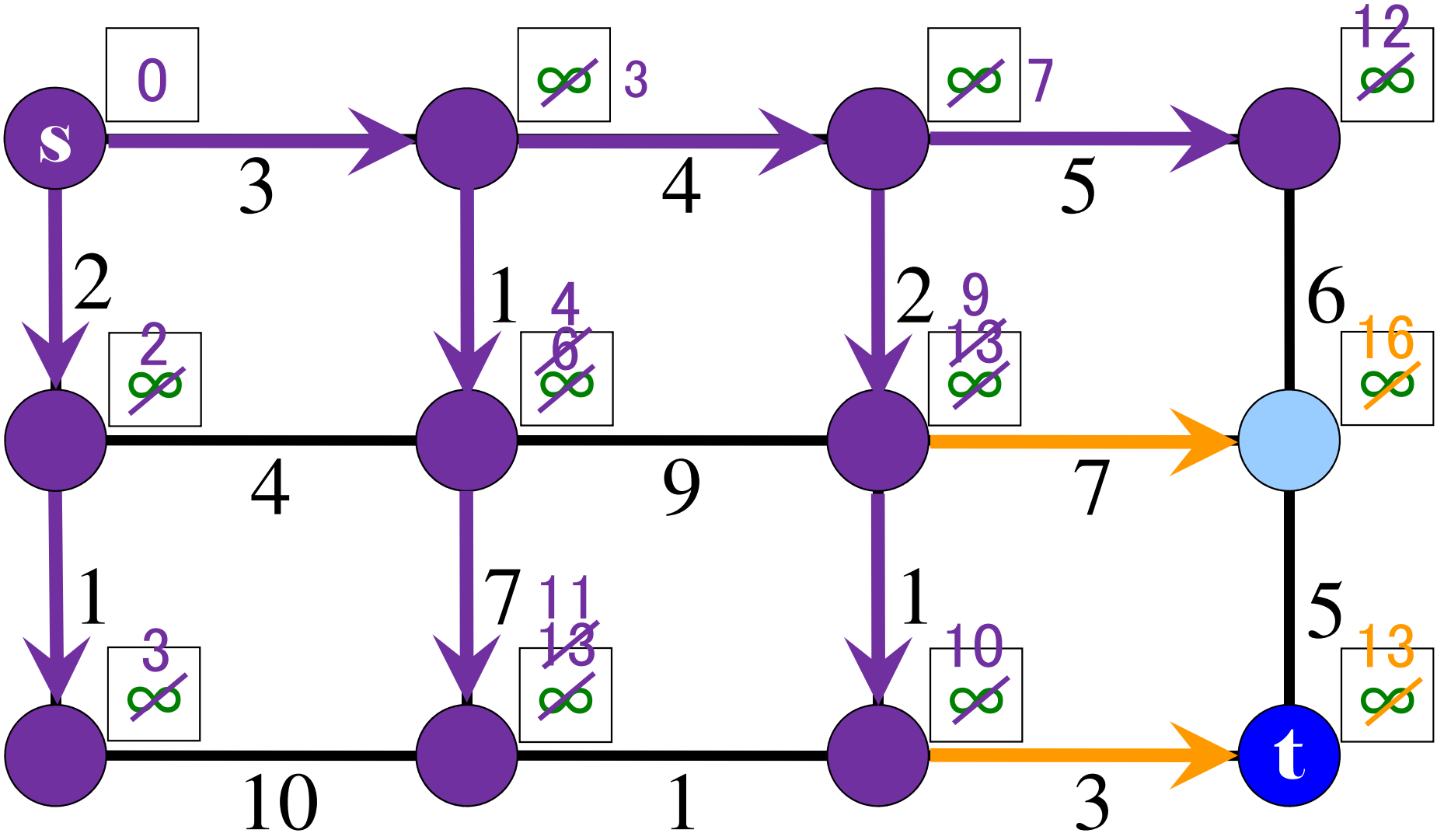
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

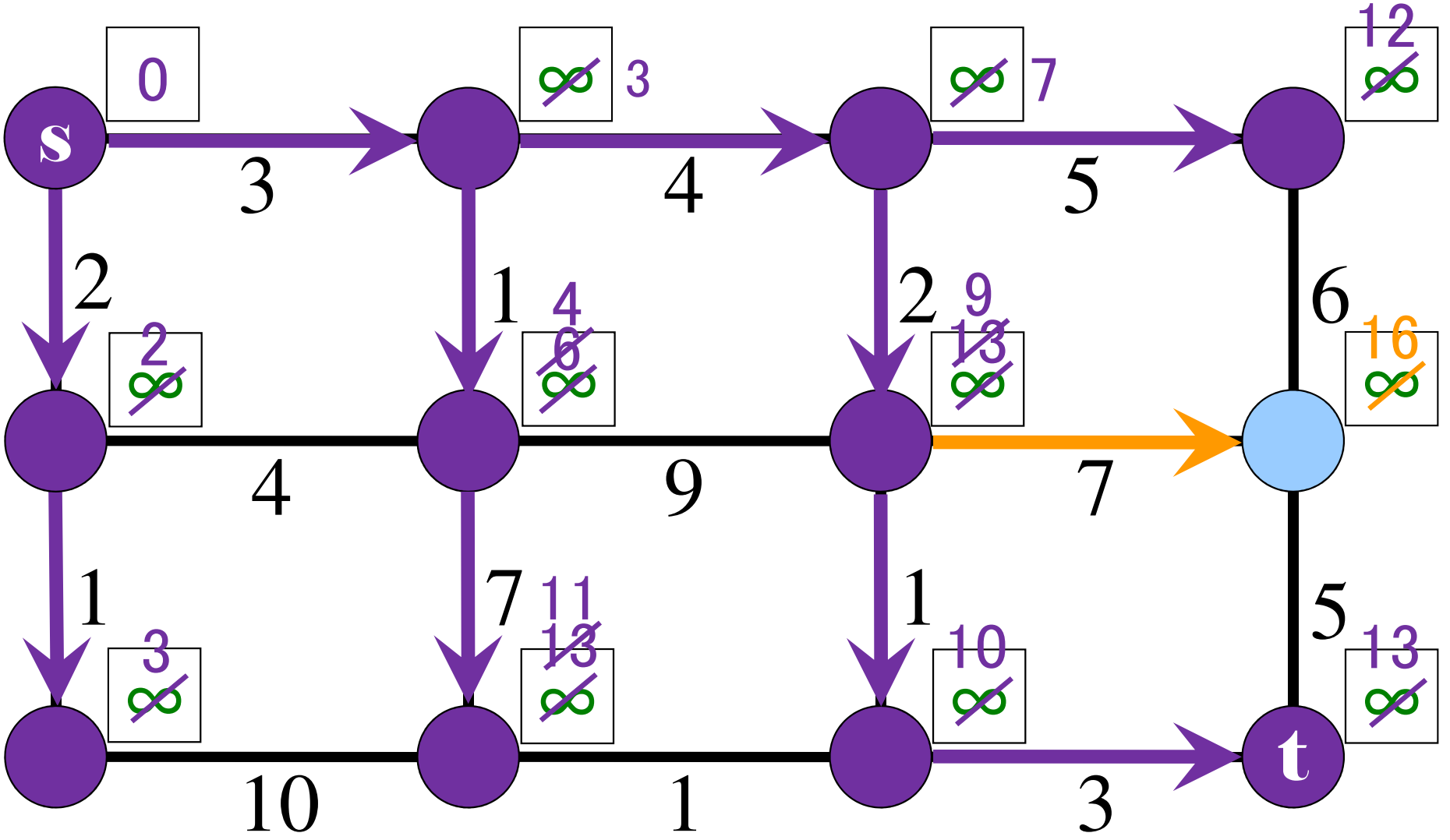


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

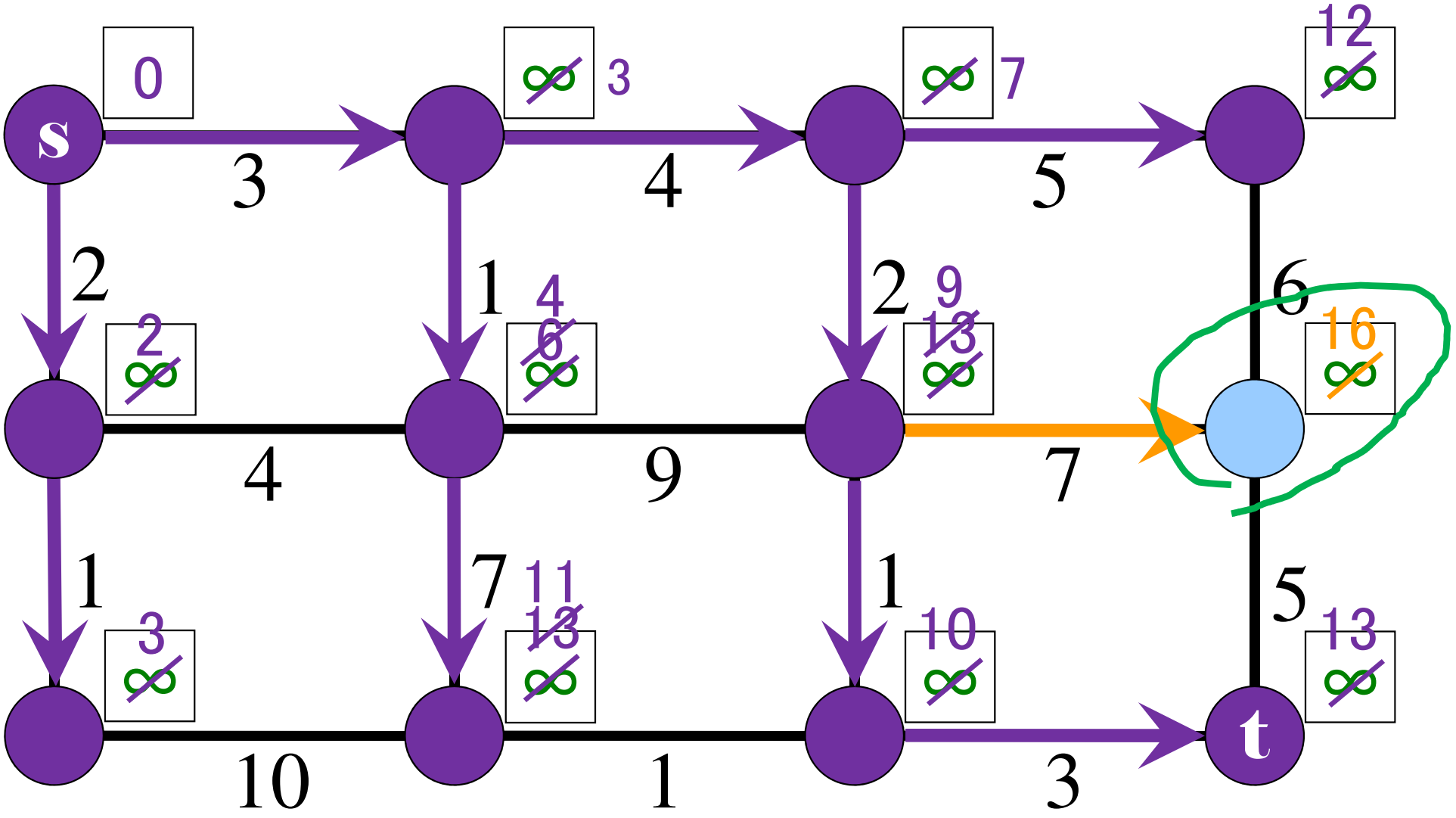
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



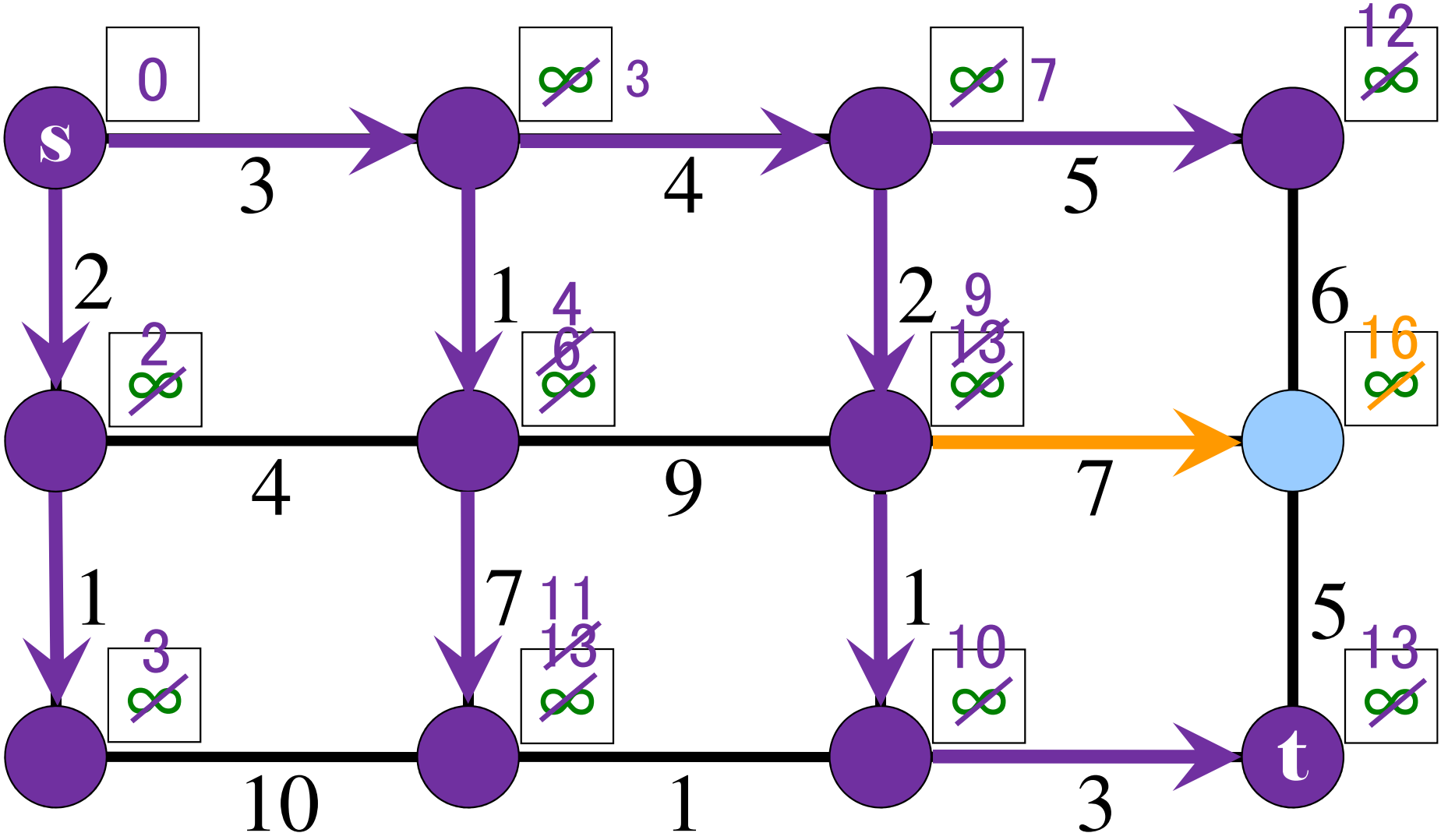
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

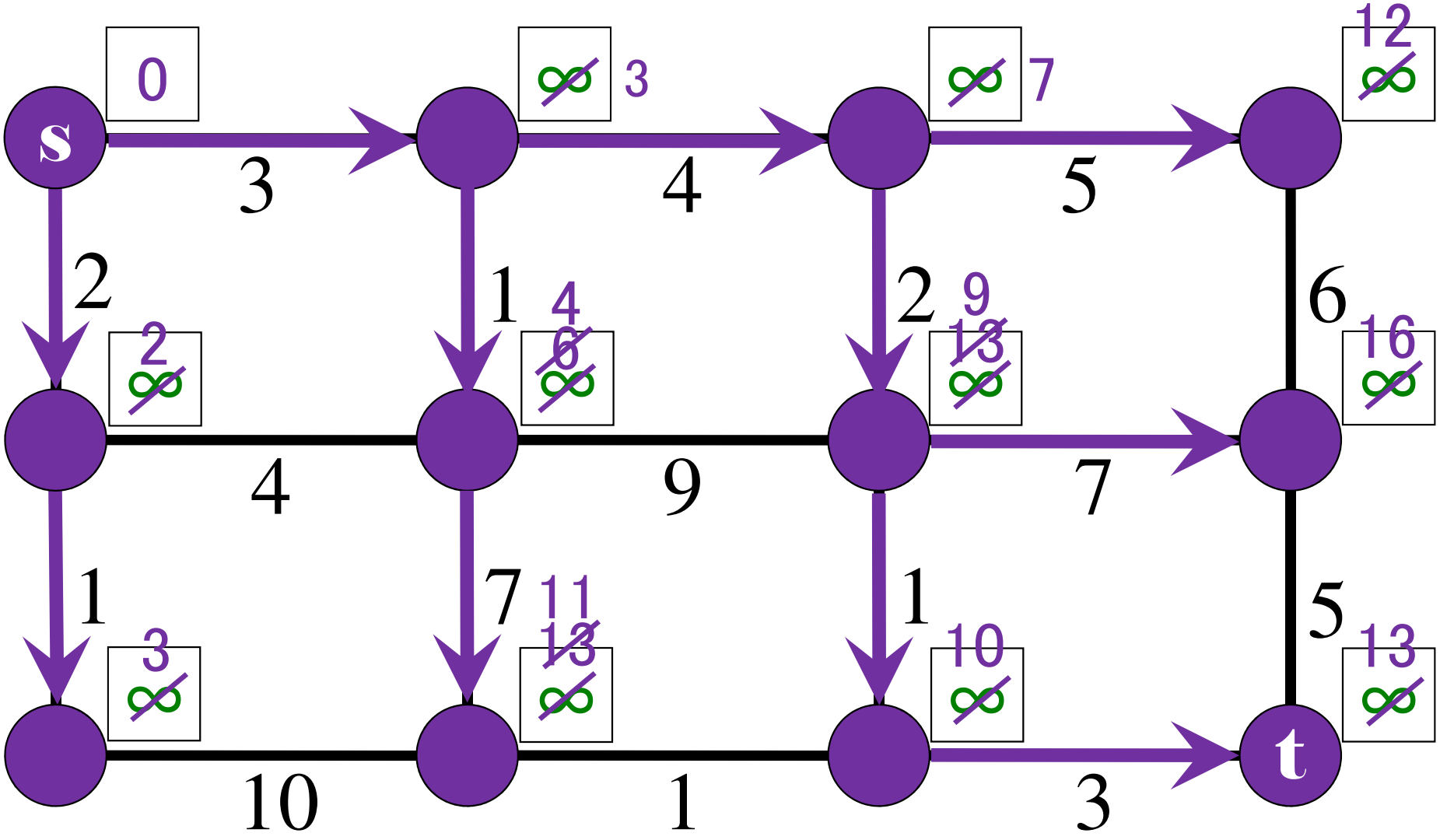


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

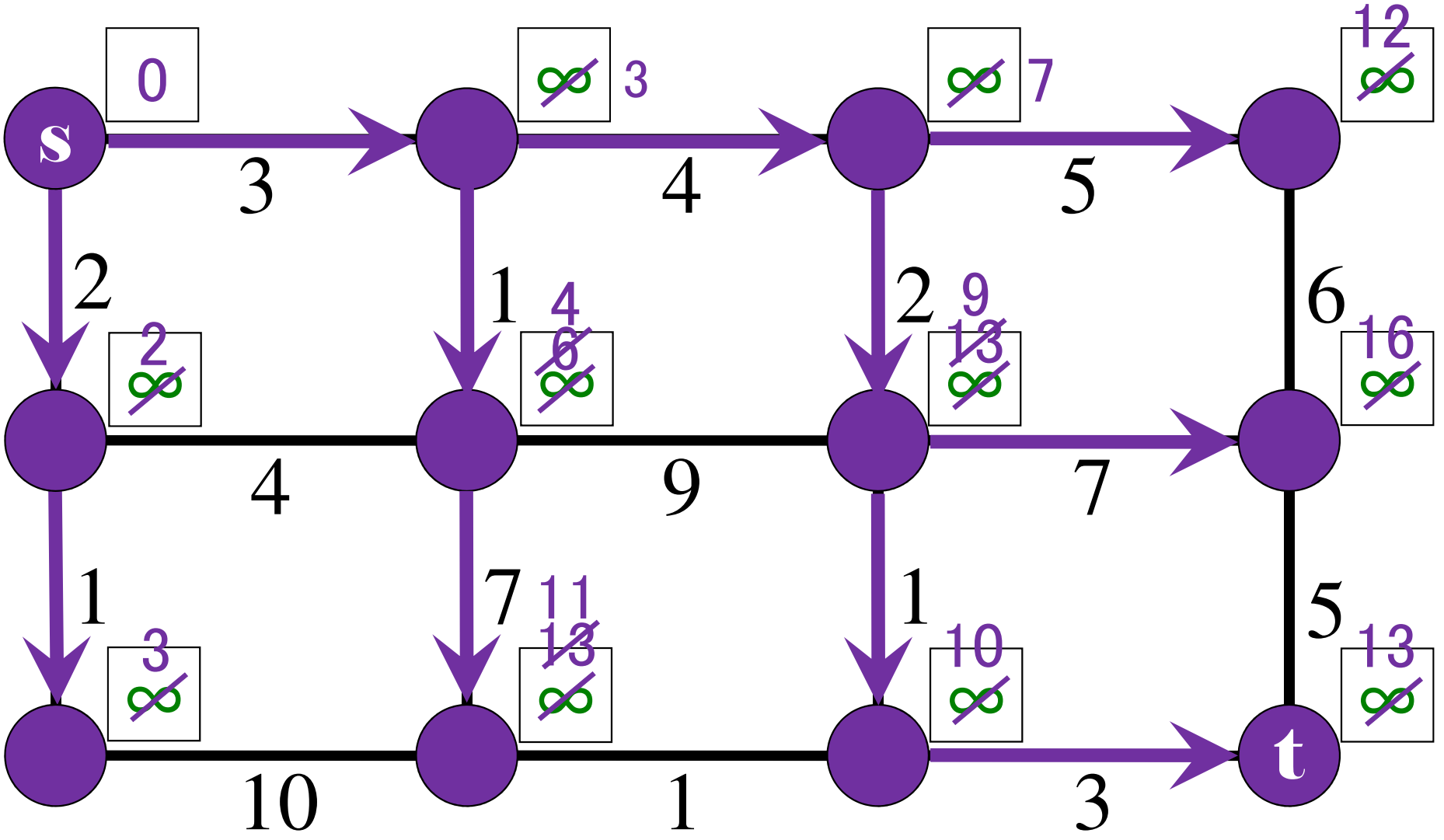
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



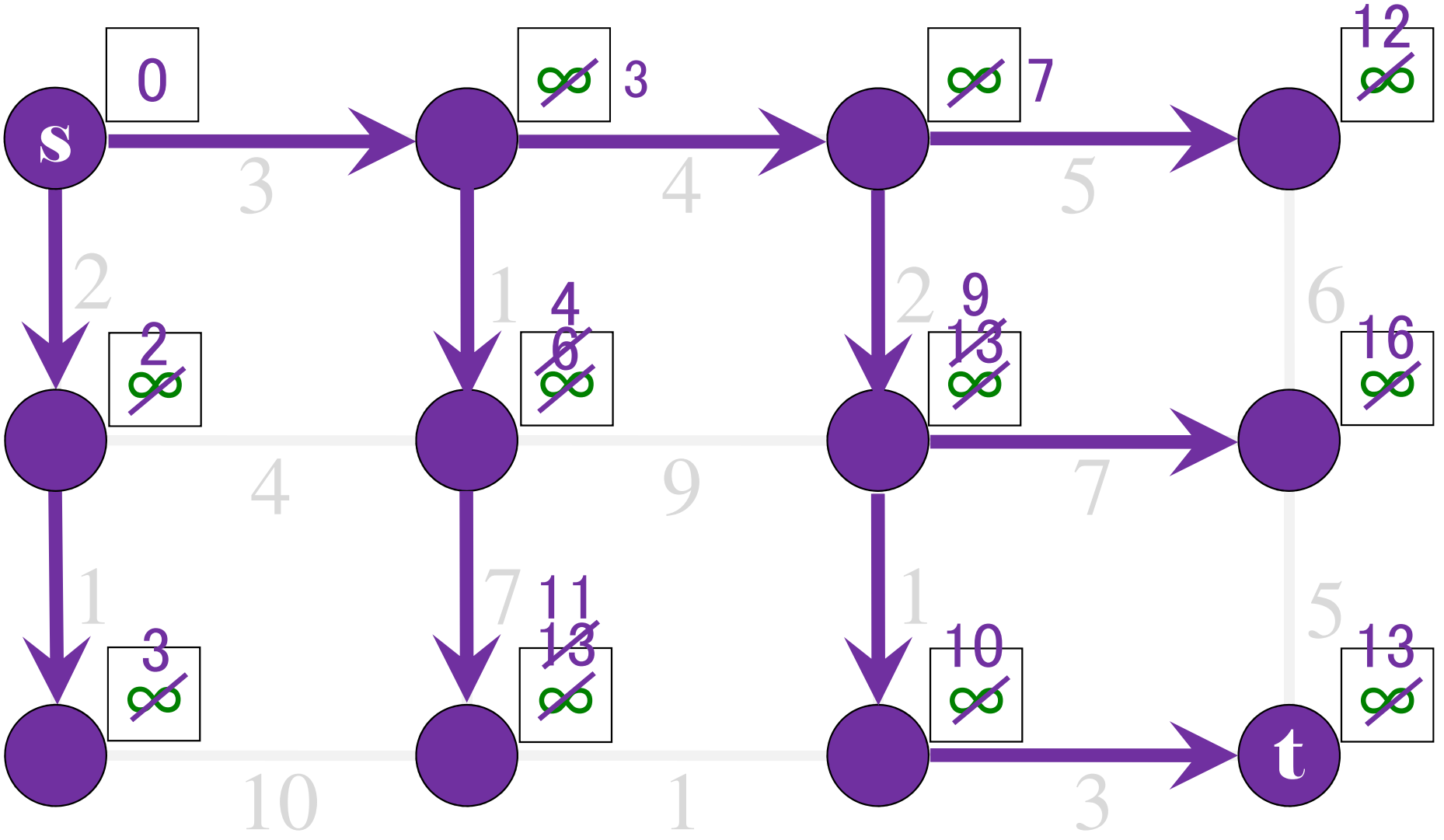
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



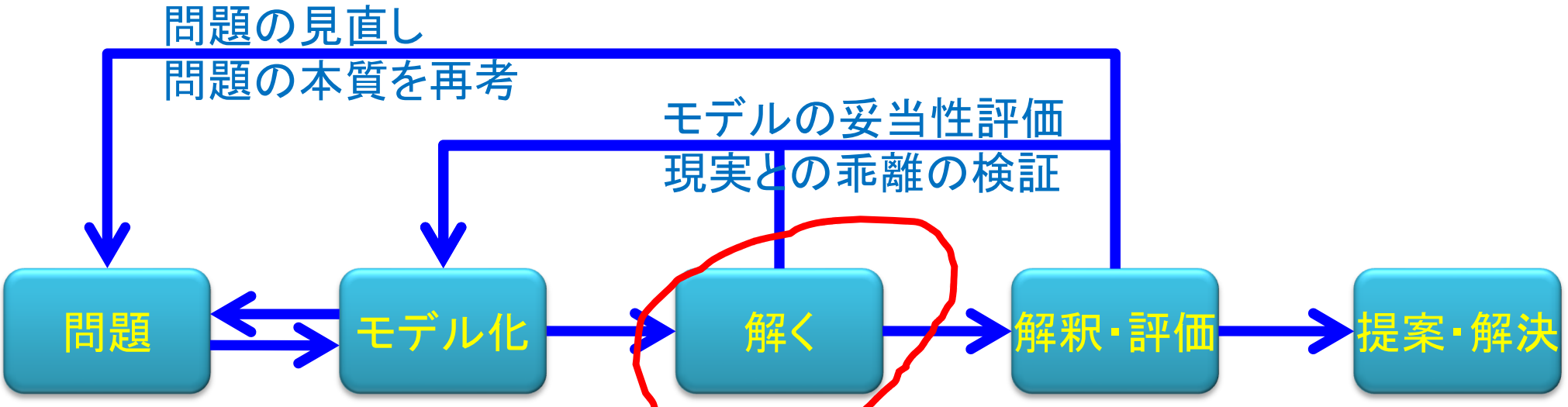
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

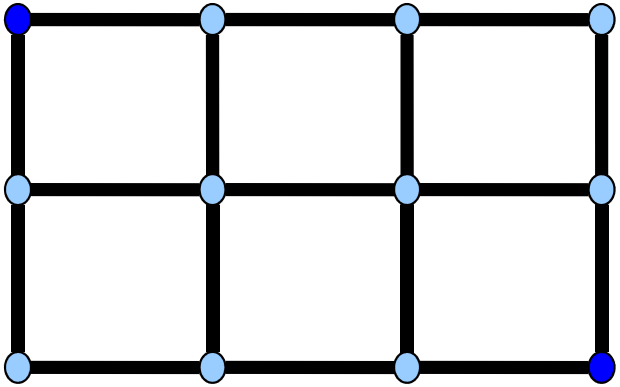
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

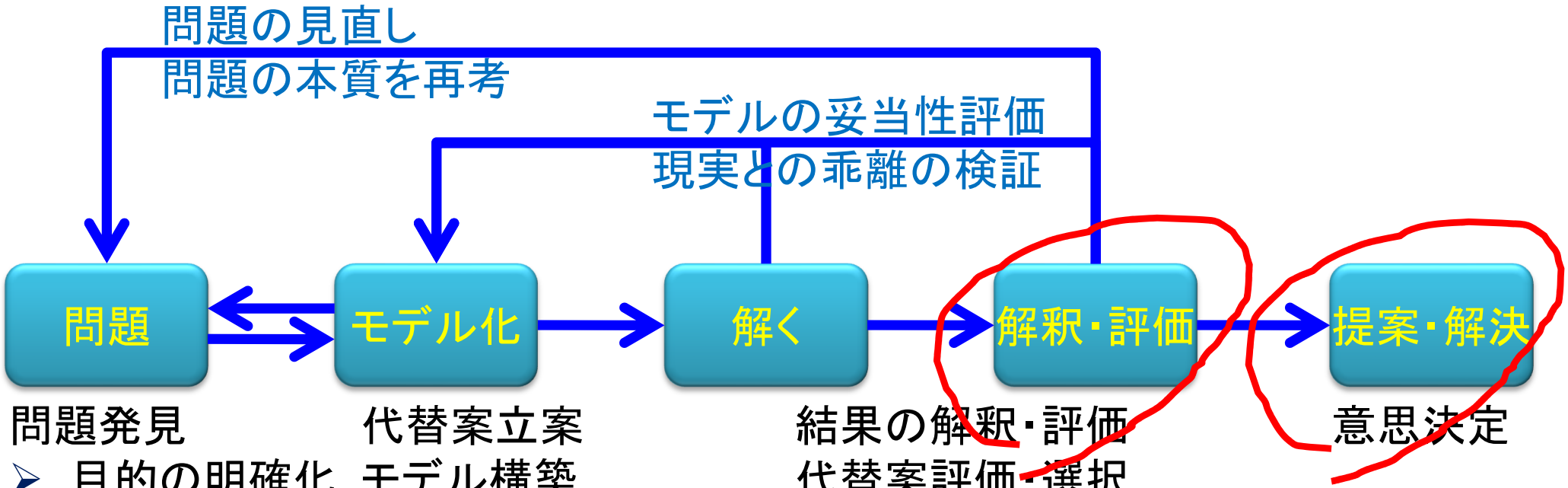
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

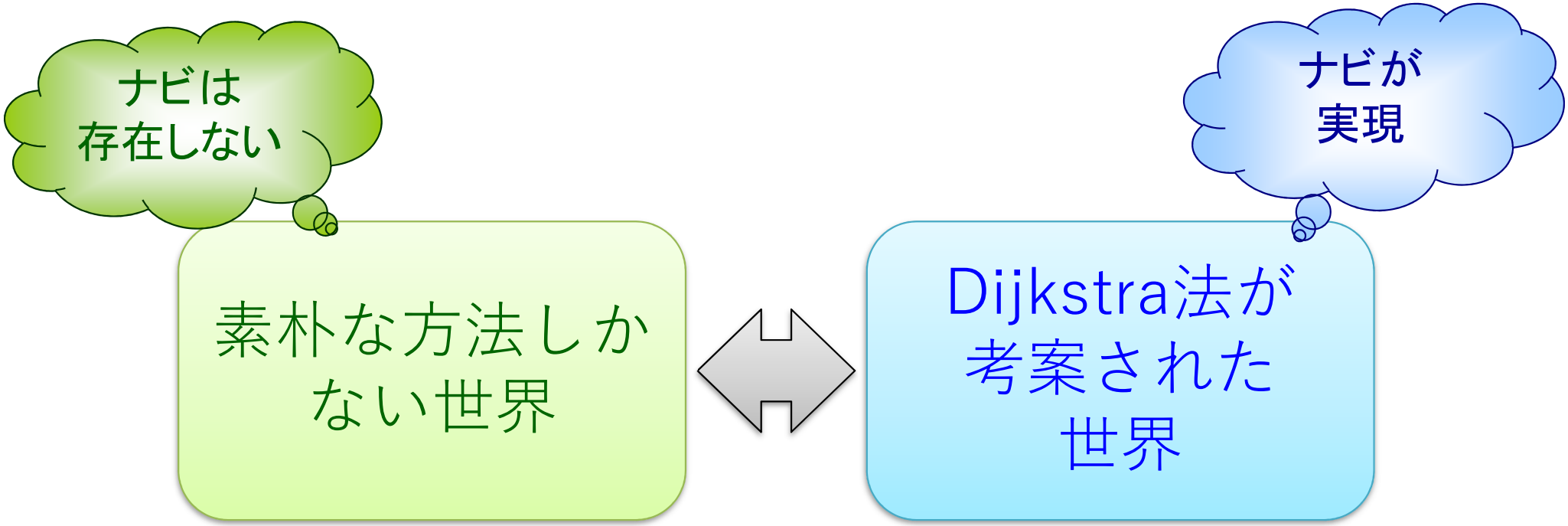
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…

2022/10/26, Wed.

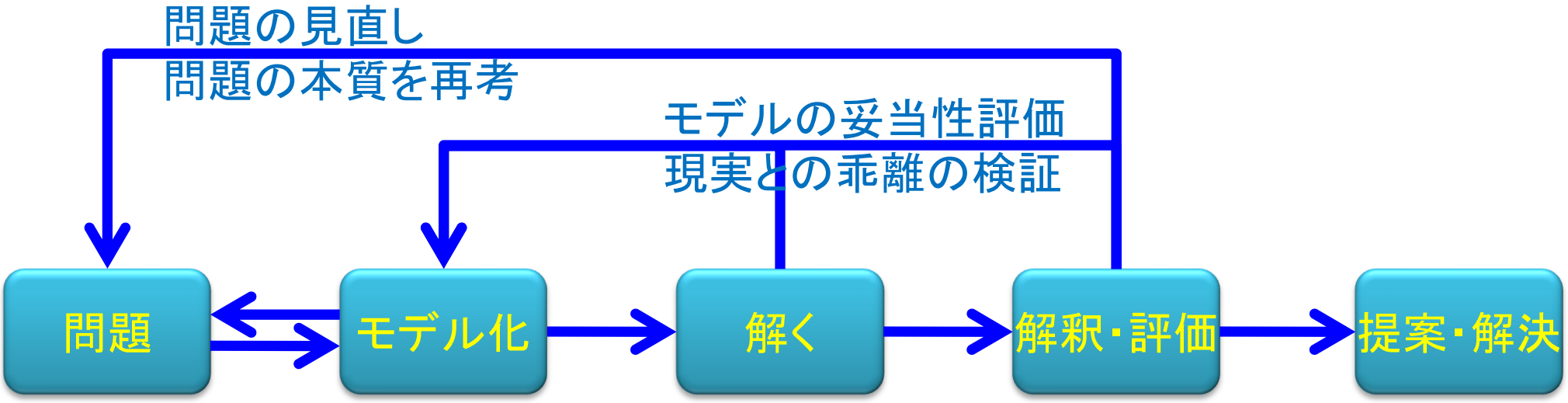
問題解決技法入門

2. Graph / Optimization
3. Shortest Path Problem

堀田 敬介

問題解決とは？

- 問題発見・問題解決から意思決定まで



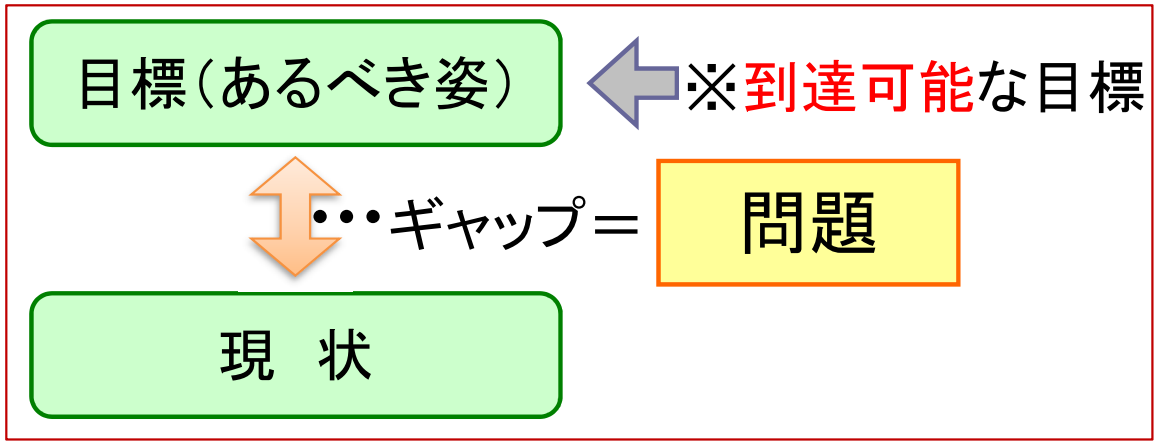
問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

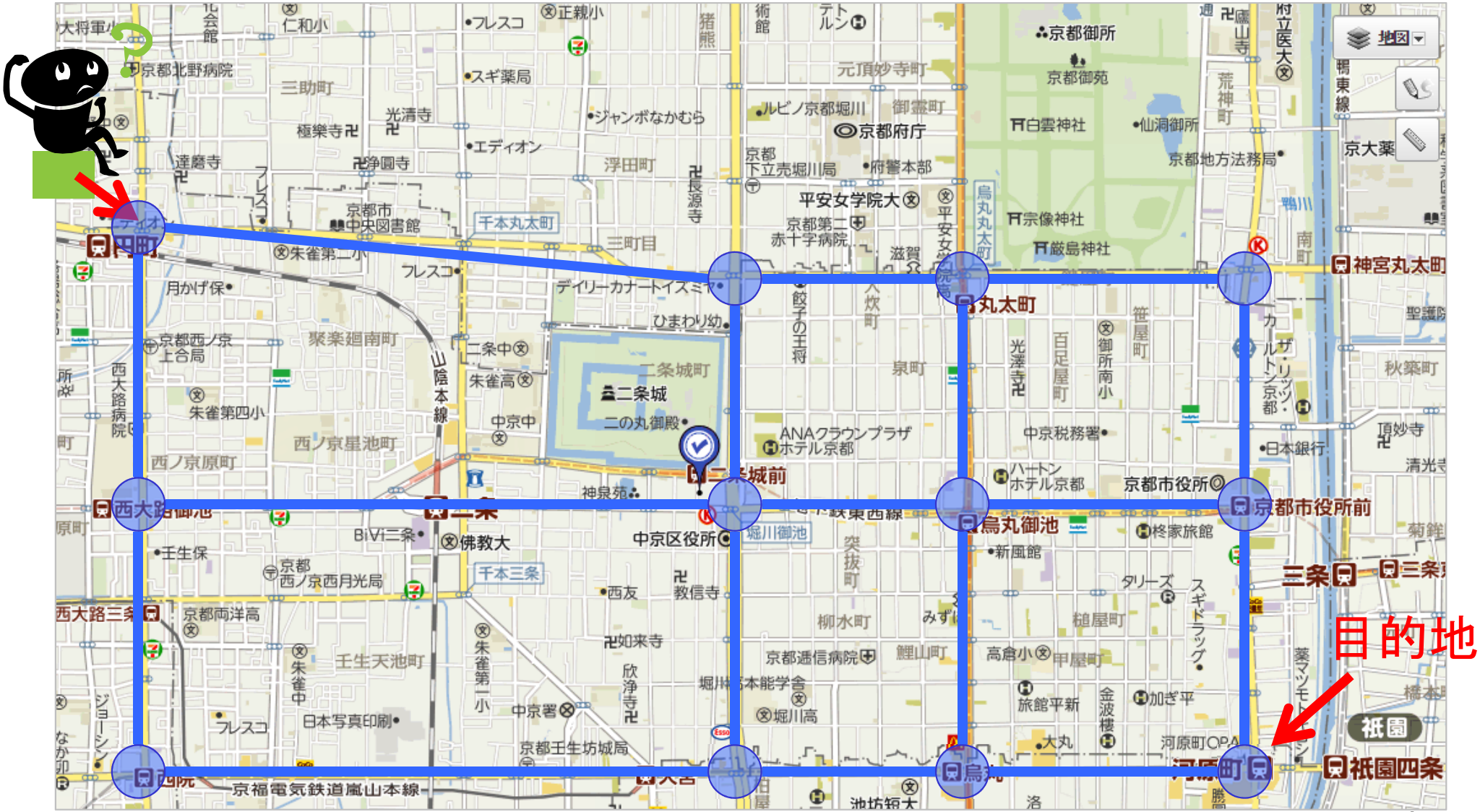
意思決定

問題の定義



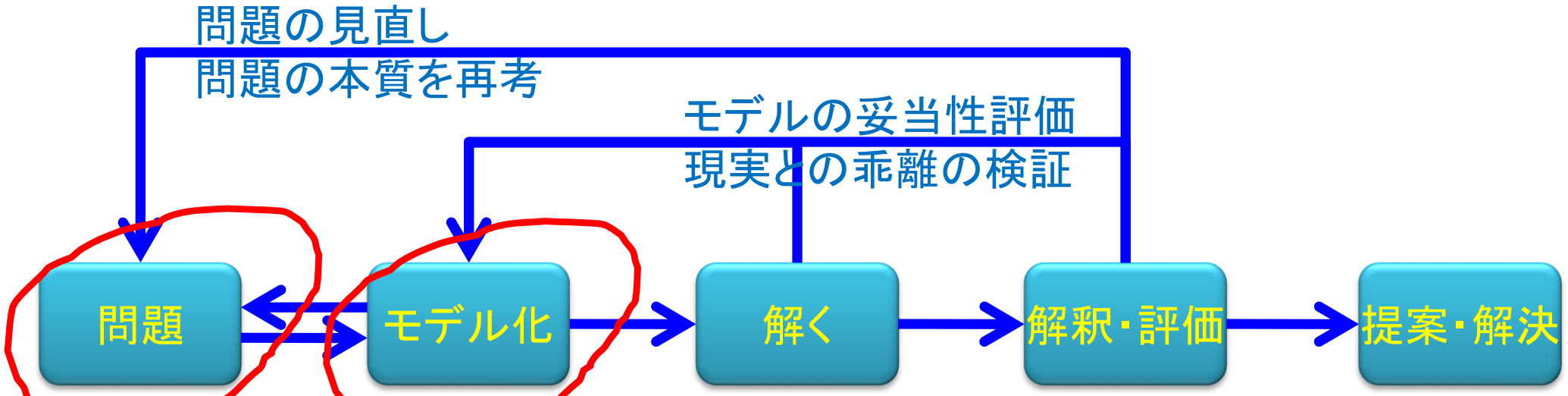
ルート探索

今，円町の交差点にいる　河原町の交差点まで，車で大通りのみを選んで通り，目的地までたどり着きたい　どの経路（ルート）を通るのがよいか？


目的地
円町
丸太町
烏丸御池
京都市役所前
二条城前
西大路御池
三条
祇園
祇園四条
河原町
京都御所
京都府庁
京都市役所
中京区役所
中京税務署
ANAクラウンプラザホテル京都
ハートンホテル京都
京都御苑
二条城
神宮丸太町
京都北野病院
京都市中央図書館
平安女学院大
京都第二赤十字病院
京都地方法務局
京大薬
京都西ノ京上合局
京都西ノ京病院
京都信用金庫
京都信用金庫
京都信用金庫

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

①
目的：最短経路を求める

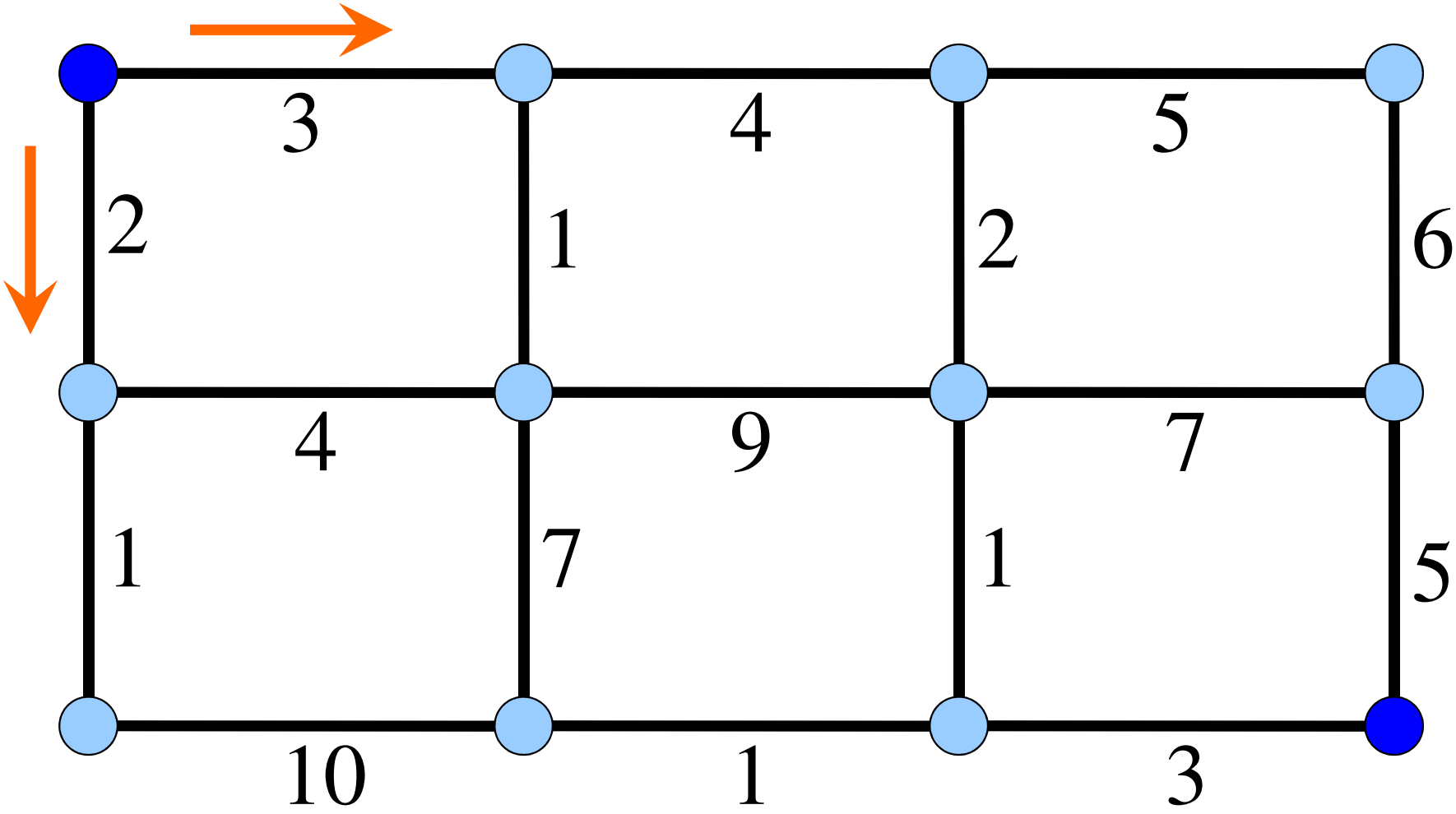
現状：データが与えられている
最短経路が求まっていない

解く:どうやって解くか？

youtube
[erato お姉さん]で検索

全ての経路を調べ，
その中から最も短い経路を選べば良い！
〔素朴で素直な方法＝全列挙，しらみつぶし〕

何通りあるか
お姉さんに聞
いてみよう！

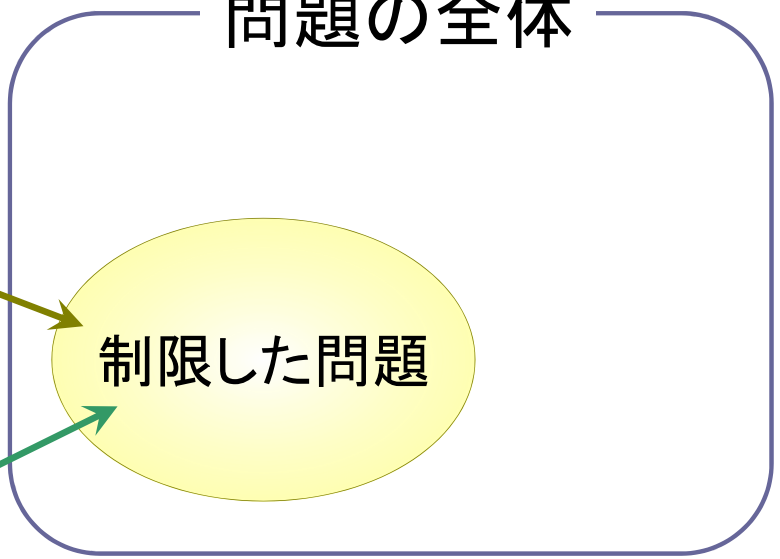


難しいなら易しくすればいいのさ！

OR的問題解決のヒント

問題を簡単にする！

- 問題の一部だけを考える
- 条件を付加して易しくする



ここだけで考えて上手くいけば，
全体に広げられるかも！

全てのネットワーク上の最短路問題

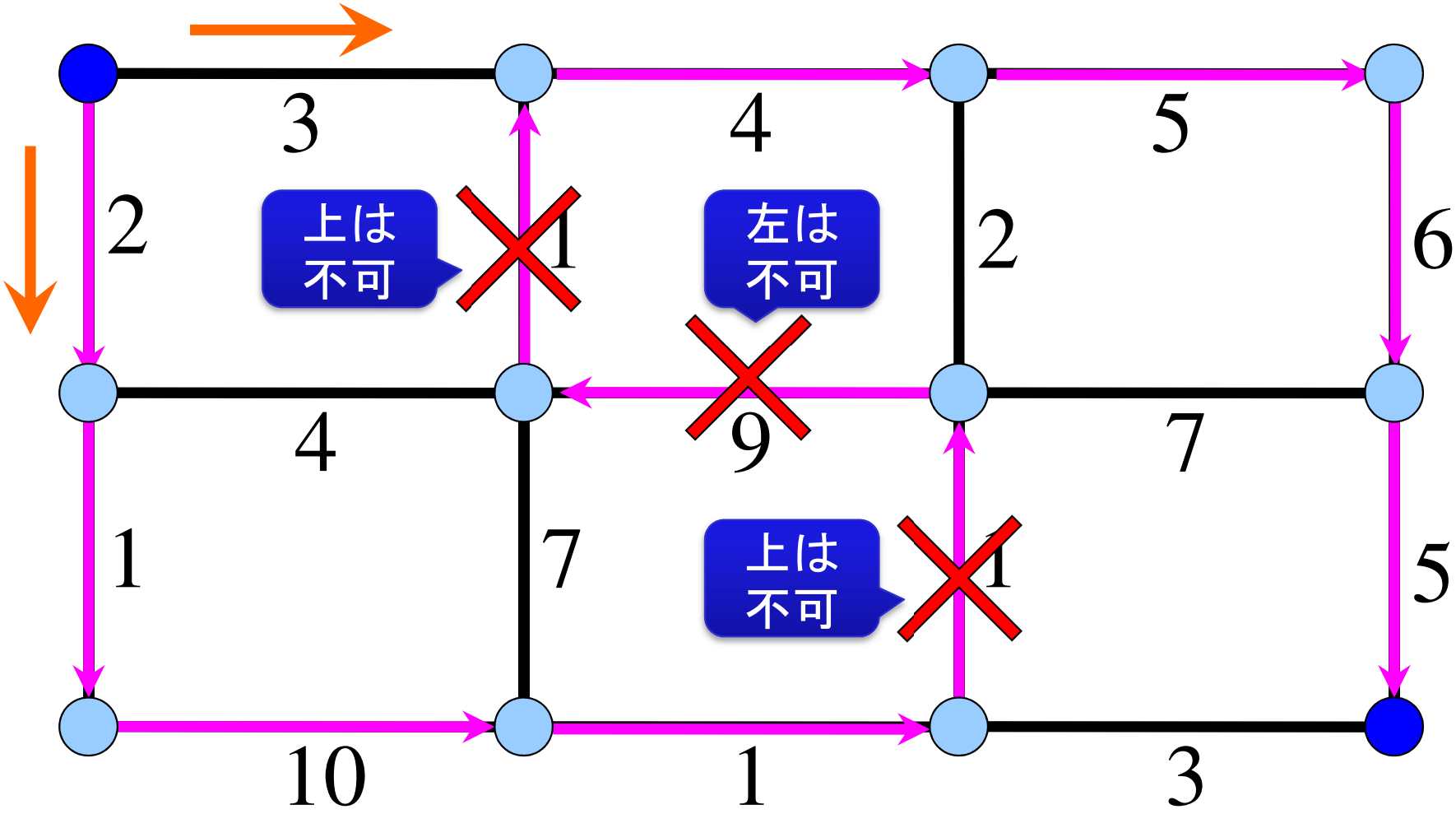
制限した問題

- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ

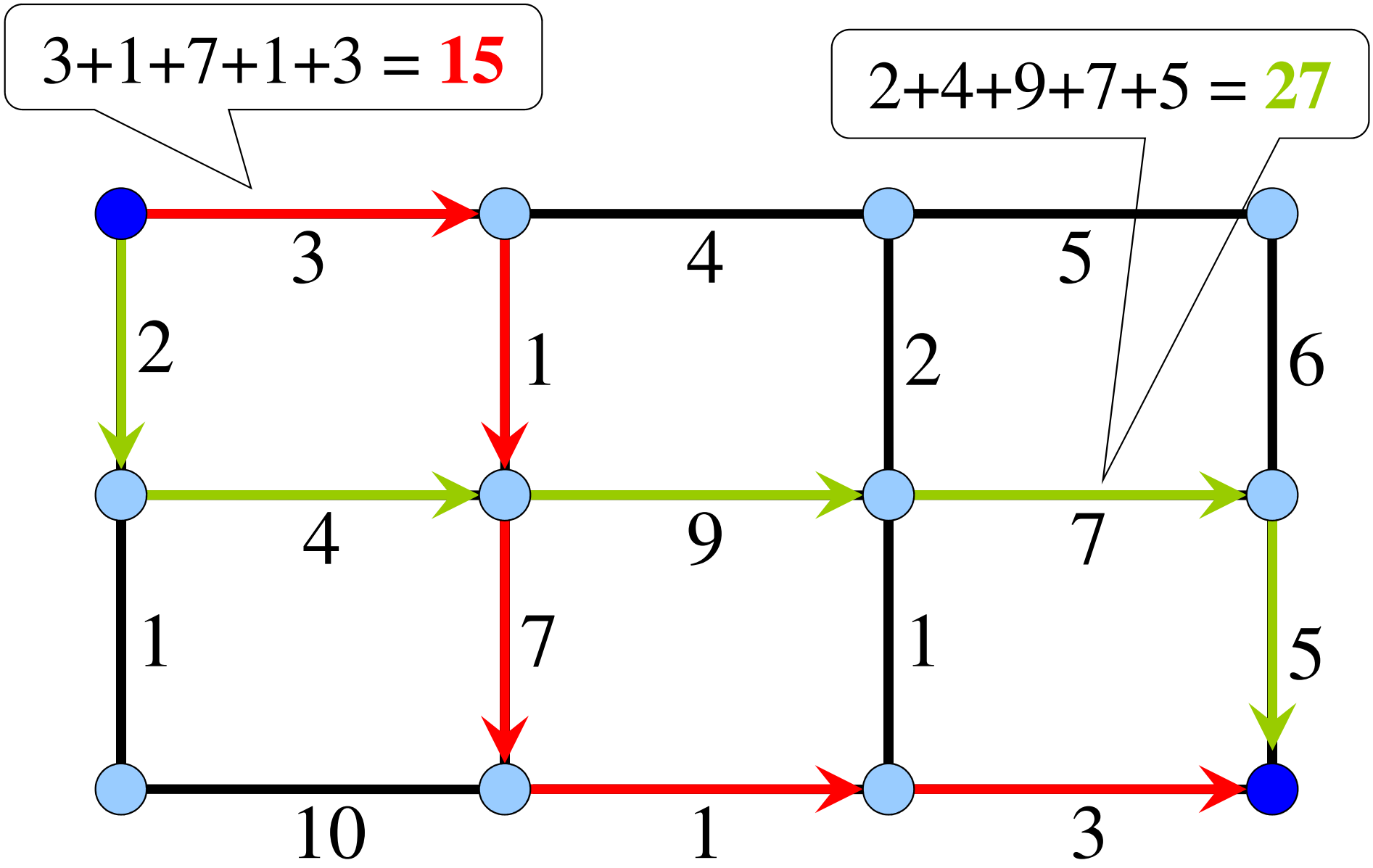
難しいなら易しくすればいいのさ！

制限した問題

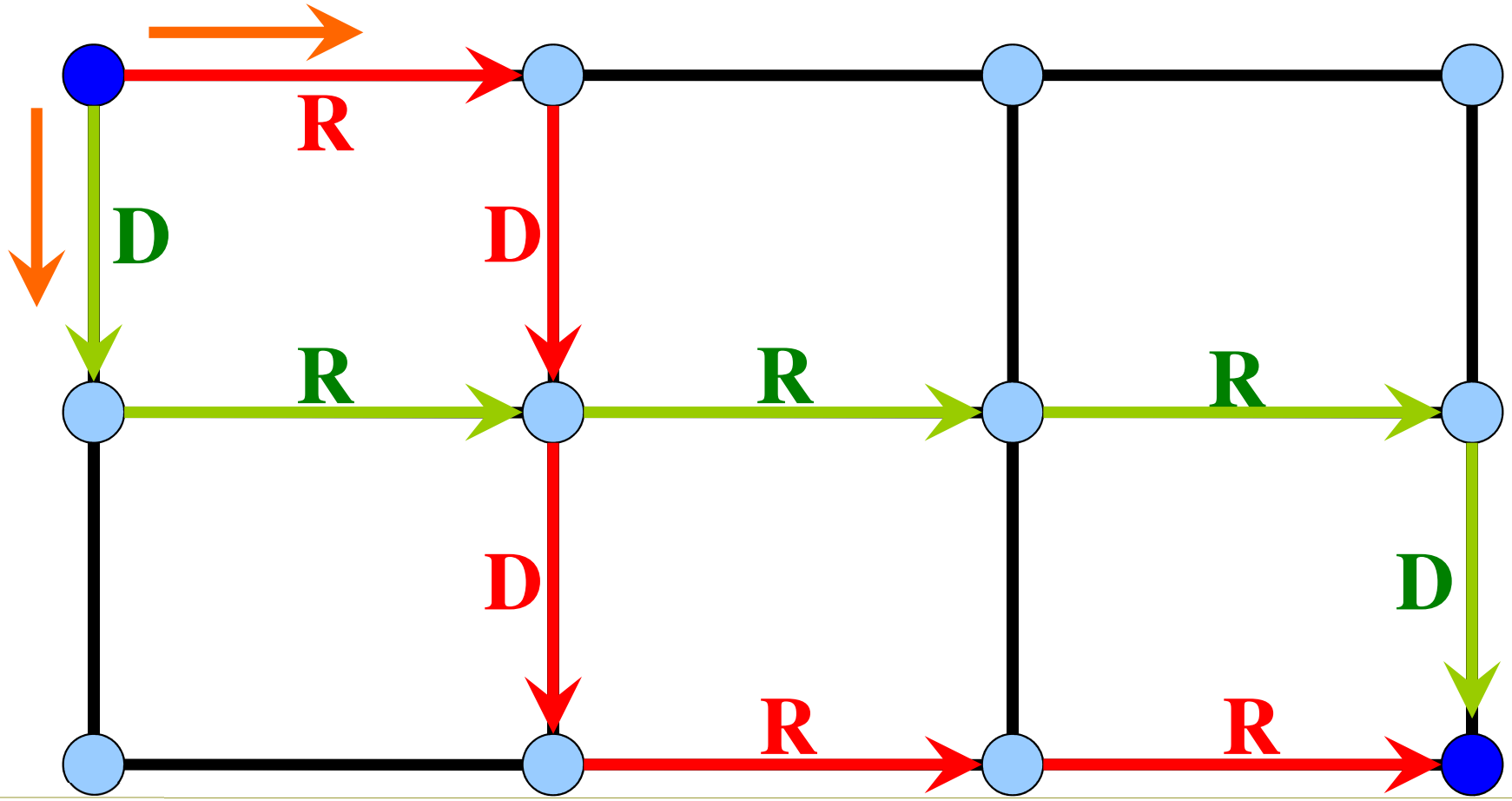
- 格子状のネットワーク
- 出発地：左上点，目的地：右下点
- 移動は右・下方向へのみ



難しいなら易しくすればいいのさ！



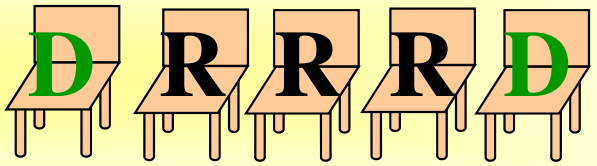
さて，経路は全部で幾つあるのか？



Point: どんな経路も，順番を無視すれば，R=3回，D=2回使う

緑の経路＝DRRRD

赤の経路＝RDDRR



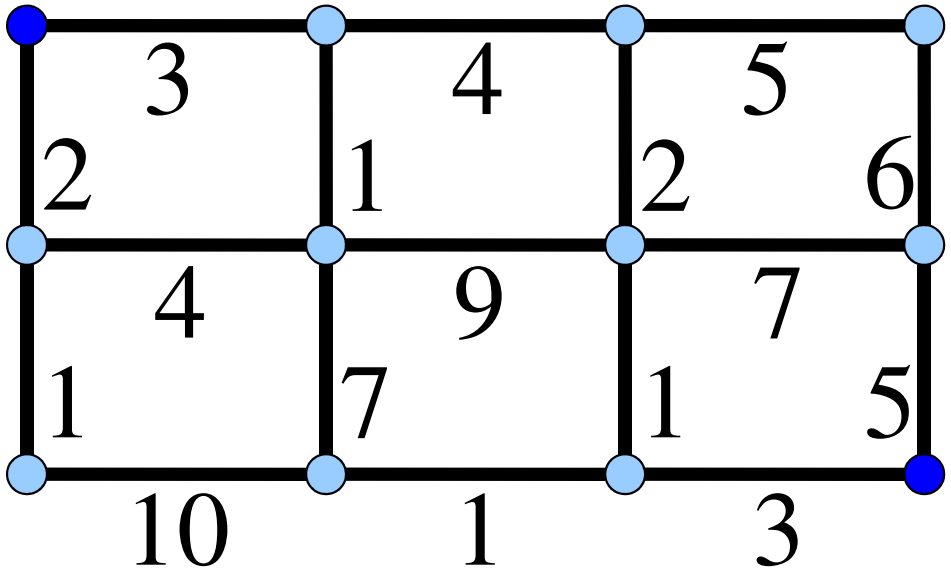
$$= \frac{(R+D)!}{R!D!} \text{通り}$$

i.e., (R+D)の椅子へのDの座らせ方を決めれば良い→ ${}_{R+D}C_D$

例では ${}_{3+2}C_2 =$ 10 通り

演習：やってみよう！全列挙

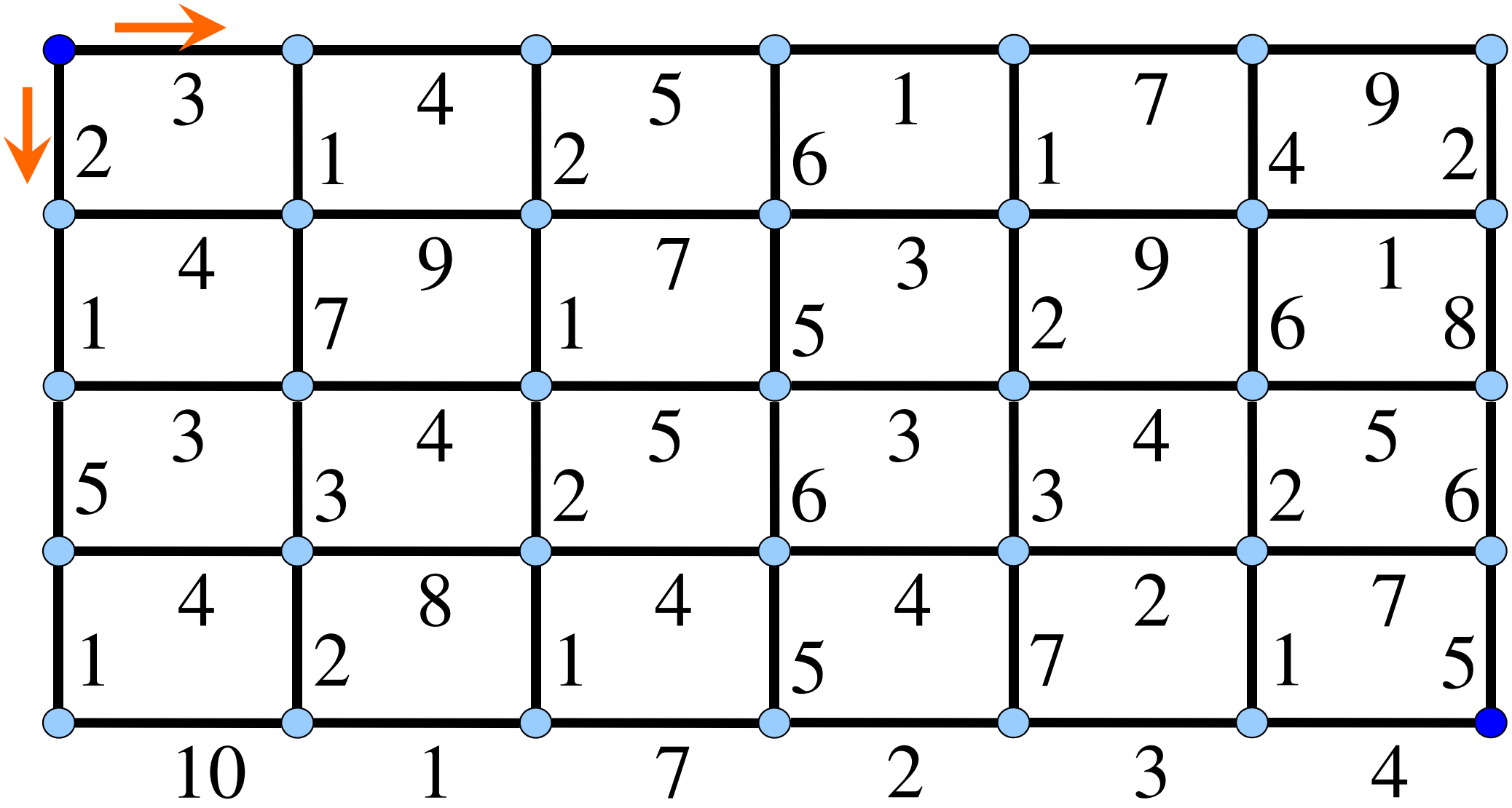
- **Q**: スタート（左上）からゴール（右下）へと至る最短経路を求めなさい．そしてそれが最短だと示しなさい



① **DDRRR:** 2+1+10+1+3=17
② **DRDRR:** 2+4+7+1+3=17
③ **DRRDR:** 2+4+9+1+3=19
④ **DRRRD:** 2+4+9+7+5=27
⑤ **RDDRR:** 3+1+7+1+3=15
⑥ **RDRDR:** 3+1+9+1+3=17
⑦ **RDRRD:** 3+1+9+7+5=25
⑧ **RRDDR:** 3+4+2+1+3=**13**
⑨ **RRDRD:** 3+4+2+7+5=21
⑩ **RRRDD:** 3+4+5+6+5=23

- **A**: 全列挙したよ　①～⑩の10通り計算し⑧が最短だ！

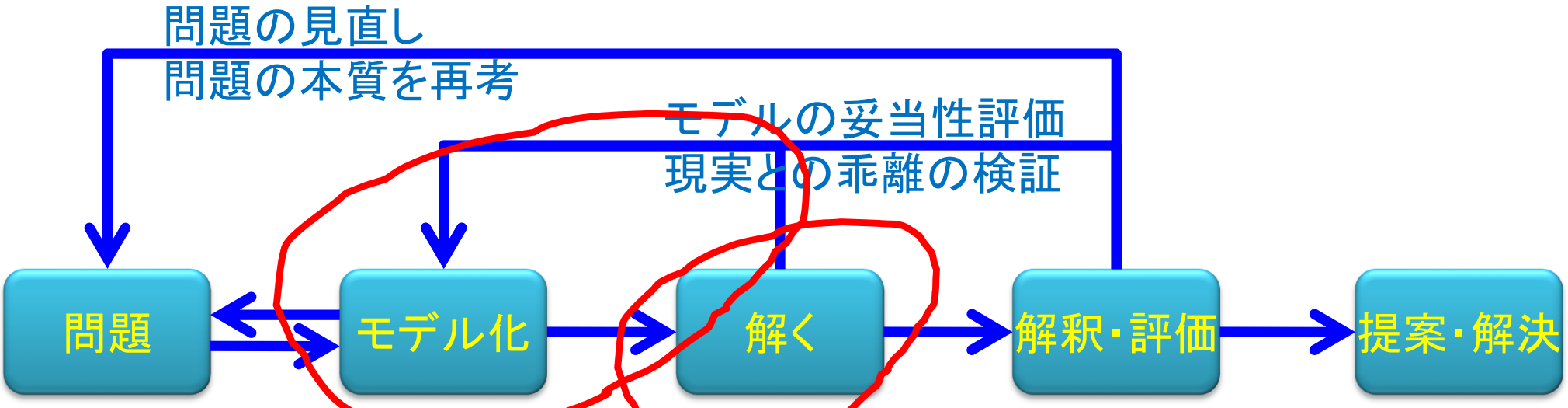
経路は全部で幾つ？



R=6, D=4なので, $_{6+4}C_4 = \frac{10 \cdot 9 \cdot 8 \cdot 7}{4 \cdot 3 \cdot 2 \cdot 1} = 210$ 通り

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

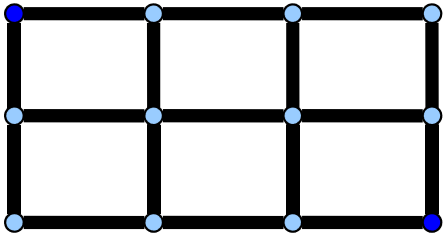
④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

経路は全部で幾つ？【全列挙】

R（横）	D（縦）	全経路
3	2	10
6	4	210
10	5	3,003
20	10	30,045,015
50	50	1.0×10^{29}
100	100	9.1×10^{58}
500	500	2.7×10^{299}
1000	1000	#NUM!



【格子道路の街】
cf.京都市,札幌市
R, D幾つぐらい？

経路は全部で幾つ？【全列挙】

経路がとてもたくさんあるとは言っても，今のコンピュータは
かなりの速さで計算できるんでしょ？　だから大丈夫だよね！

- 代表的なCPU, Game機, super computer の浮動小数点演算回数
  - Intel Core i7(3.2GHz) : **51.2 GFLOPS**　…1秒間に**512億**回
  - PS3 : **218 GFLOPS**　…1秒間に**2180億**回
  - PS4 : **1.84 TFLOPS**　…1秒間に**1兆8400億**回
  - PS5 : **10.28 TFLOPS**　…1秒間に**10兆2800億**回
  - 京 : **10.51 PFLOPS**　…1秒間に**1京510兆**回
  - 富岳 : **415.53 PFLOPS**　…1秒間に**41京5530兆**回

※**FLOPS** = ***FL**oating-point **O**perations **P**er **S**econd*

（※京：2011年6,11月 世界最速 [Top500.org]）
（※富岳：2020年6月 世界最速 [Top500.org]）

K（キロ）≃×10^3=千倍
M（メガ）≃×10^6=百万倍
G（ギガ）≃×10^9=10億倍
T（テラ）≃×10^{12}=1兆倍
P（ペタ）≃×10^{15}=千兆倍
E（エクサ）≃×10^{18}=百京倍

1つの経路を見つけ，その総コストを計算するのに，たどる経路枝数の浮動小数点演算でできると仮定する．例えば，R=10, D=5の経路なら，10+5回の演算で計算可とするということ

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}		
50	50	1.0×10^{29}		
100	100	9.1×10^{58}		
500	500	2.7×10^{299}		

経路は全部で幾つ？【全列挙】

10.28 TFLOPS　　415.53 PFLOPS

R（横）	D（縦）	全経路	PS5	富岳
3	2	10	0.000000000 秒	0.000000000 秒
6	4	210	0.000000000 秒	0.000000000 秒
10	5	3,003	0.000000004 秒	0.000000000 秒
20	10	30,045,015	0.000087680 秒	0.000000002 秒
25	25	1.3×10^{14}	10.25 分	0.015210768 秒
30	30	1.2×10^{17}	7.99 日	17.076684942 秒
40	40	1.1×10^{23}	26,529.43 年	239.56 日
50	50	1.0×10^{29}	2.26 宙齢	769,918.79 年
100	100	9.1×10^{58}	4.05×10^{30} 宙齢	1.00×10^{26} 宙齢
500	500	2.7×10^{299}	6.04×10^{271} 宙齢	1.49×10^{267} 宙齢

圧倒的な計算力をもつコンピュータですら，全列挙（しらみつぶし）では答えを求めることが出来ない！

＃1宙齢＝138億年

参考：大きい数を表す接頭辞

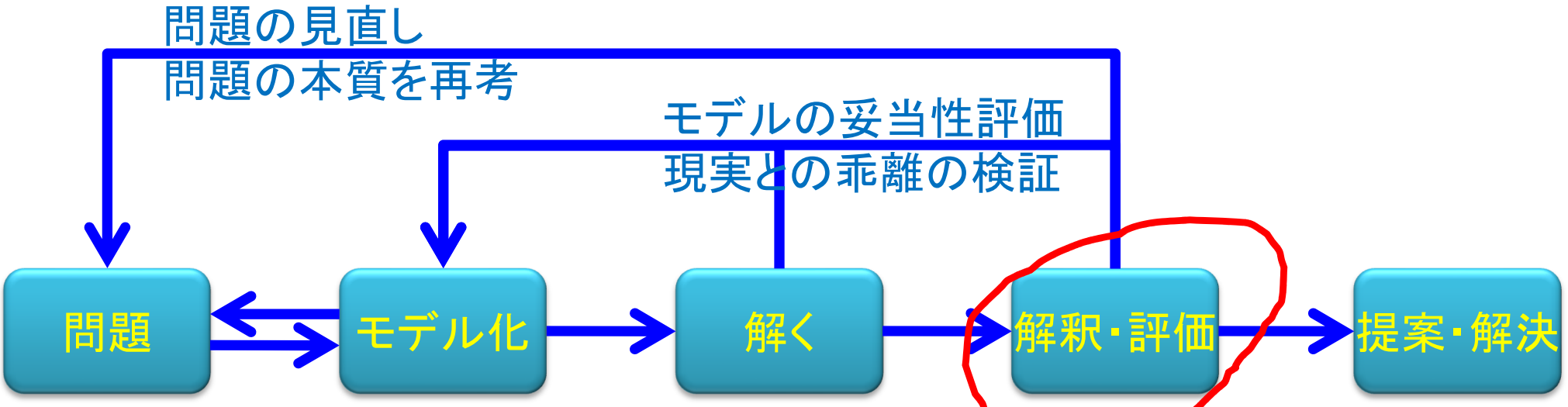
- 万（まん） $\times 10^{4}$
- 億（おく） $\times 10^{8}$
- 兆（ちょう） $\times 10^{12}$
- 京（けい） $\times 10^{16}$
- 垓（がい） $\times 10^{20}$
- 杼（じょ） $\times 10^{24}$
- 穣（じょう） $\times 10^{28}$
- 溝（こう） $\times 10^{32}$
- 澗（かん） $\times 10^{36}$
- 正（せい） $\times 10^{40}$
- 載（さい） $\times 10^{44}$
- 極（ごく） $\times 10^{48}$

- 恒河沙（ごうがしゃ） $\times 10^{52}$
- 阿僧祇（あそうぎ） $\times 10^{56}$
- 那由他（なゆた） $\times 10^{60}$
- 不可思議（ふかしぎ） $\times 10^{64}$
- 無量大数（むりょうたいすう） $\times 10^{68}$

【注】「杼」は正しくは「のぎへん」（らしい）
【注】「無量大数」は「無限大∞」とは違う

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ×

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ×

ではどうする？

- 素朴で素直な方法〔列挙法〕
  - 全経路をしらみつぶしに調べて，最も短い経路を見つける方法

時間が掛かり過ぎ

全経路をしらみつぶしに調べずに，
最も短い経路を，現実的時間で
見つける方法があるか？

Dijkstra法
（ダイクストラ法）

人間の創造的な仕事！

Dijkstra法

（初期設定）

step0: start点(s)のラベルを0にし，その他のラベルを∞に設定する．未確定点集合{S,1,2,...,T}(=V)とする

（更新法）

step1-1: 未確定点中，ラベル値が最小の点を見つける

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

step1-3: その点から出る全枝の作業が終了したら，その点を未確定点集合から除去
（確定点のラベル値＝その点までの最短距離）

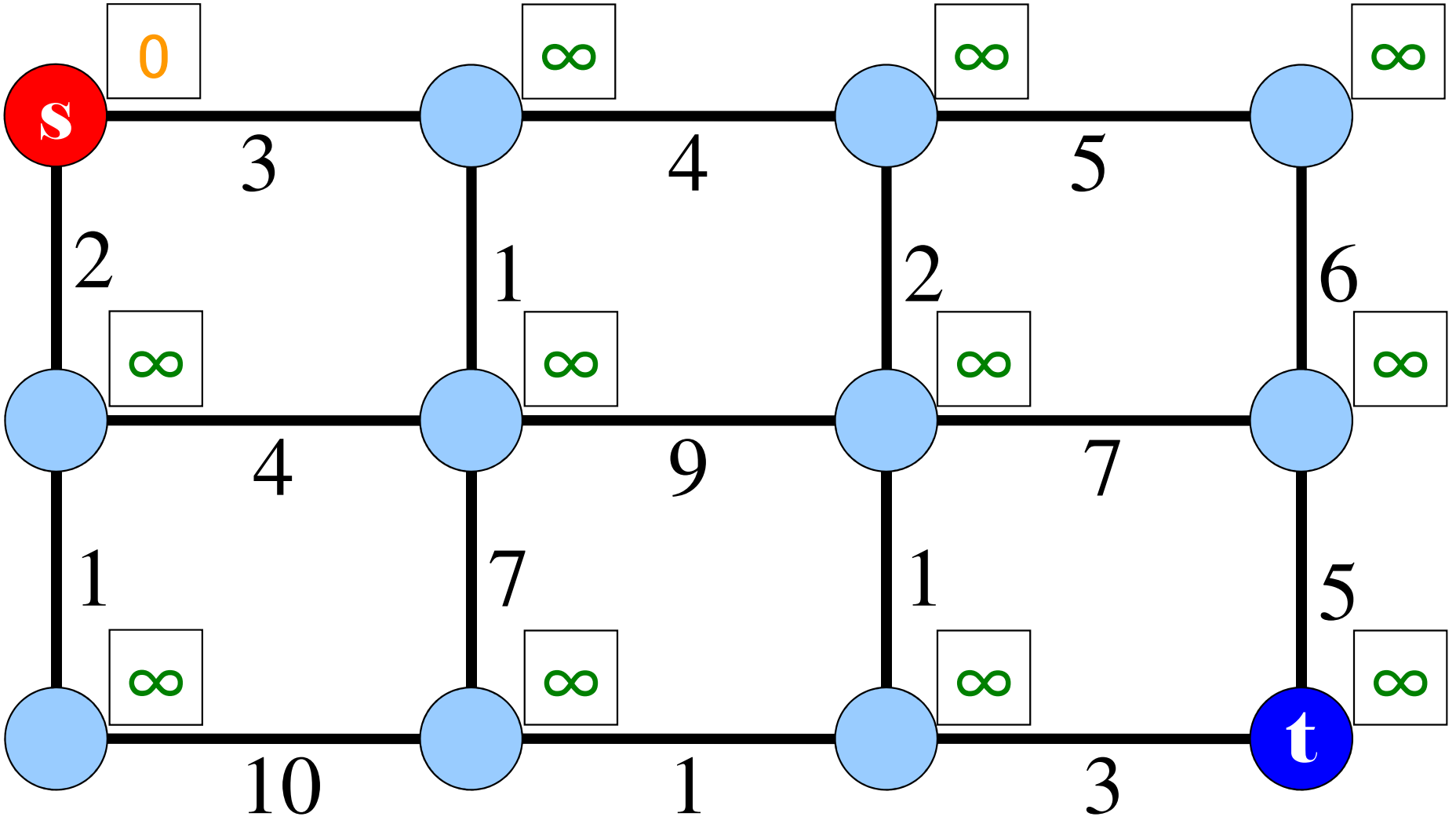
step1-1～**step1-3**を終了条件を満たすまで繰り返す

（終了判定）

step2: 未確定点集合が空になったら終了

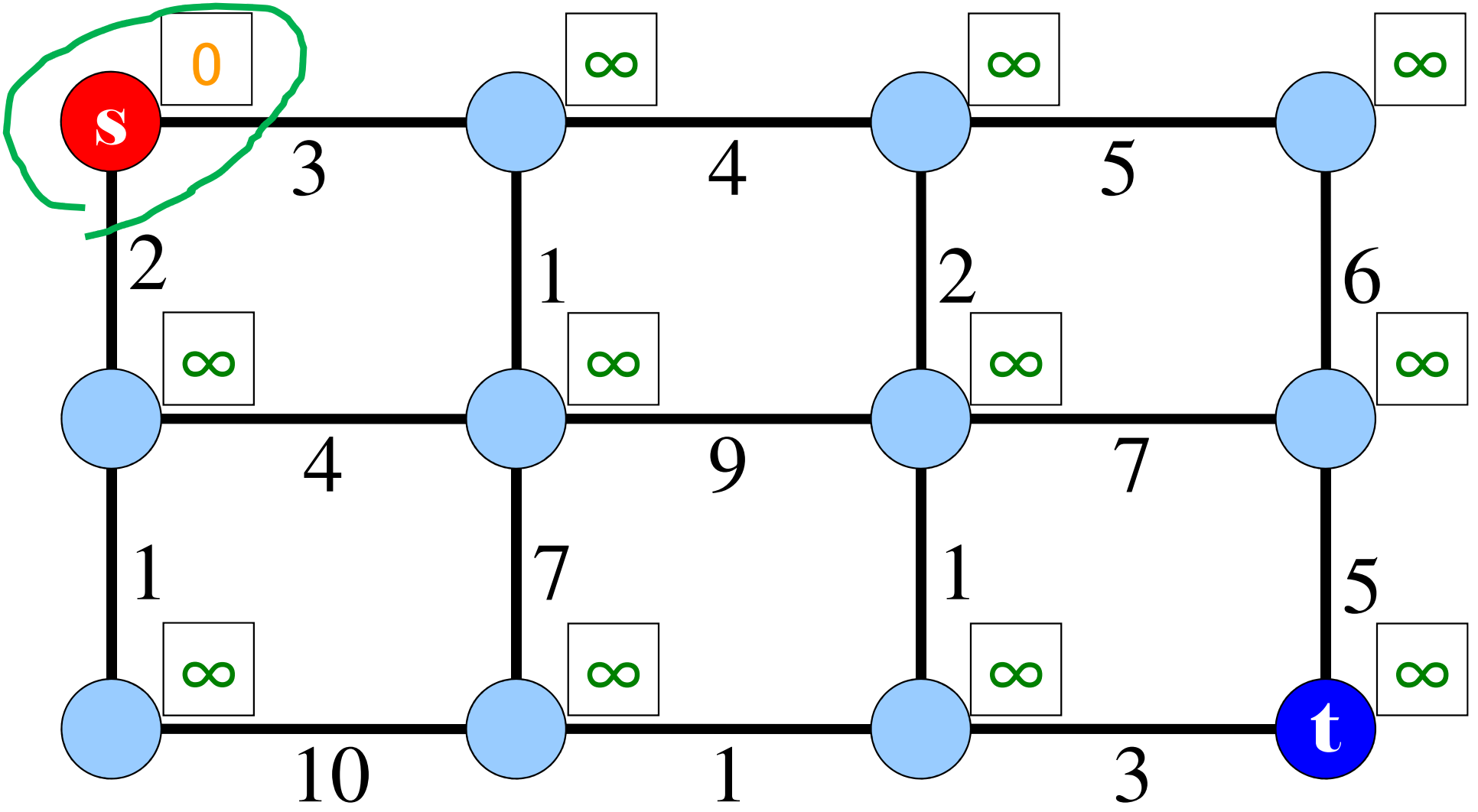
Dijkstra法（初期設定）

step0: start点 s のラベルを0にし，その他のラベルを∞に設定する．未確定点集合 {s,1,2,...,t}(=V) とする



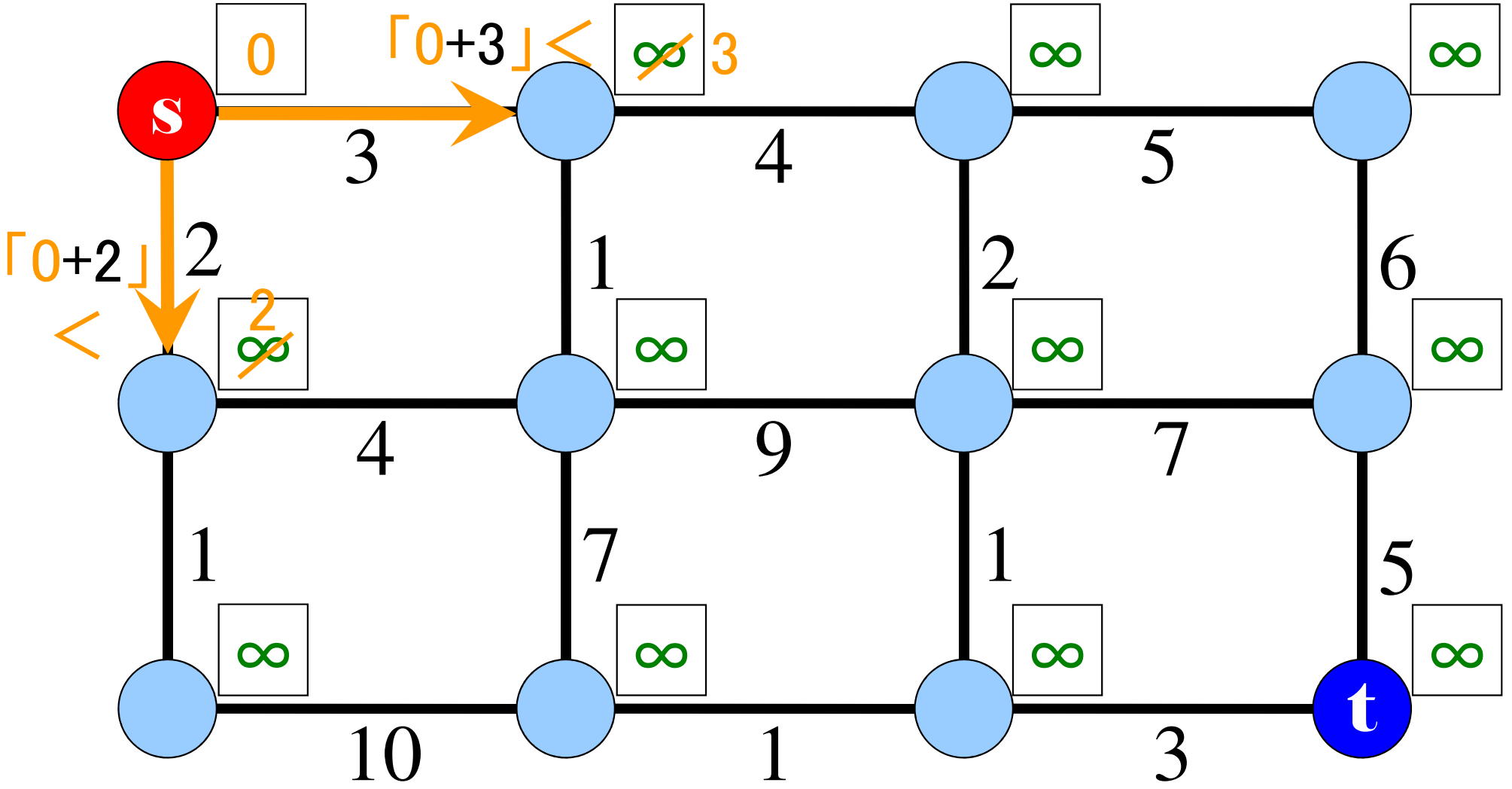
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

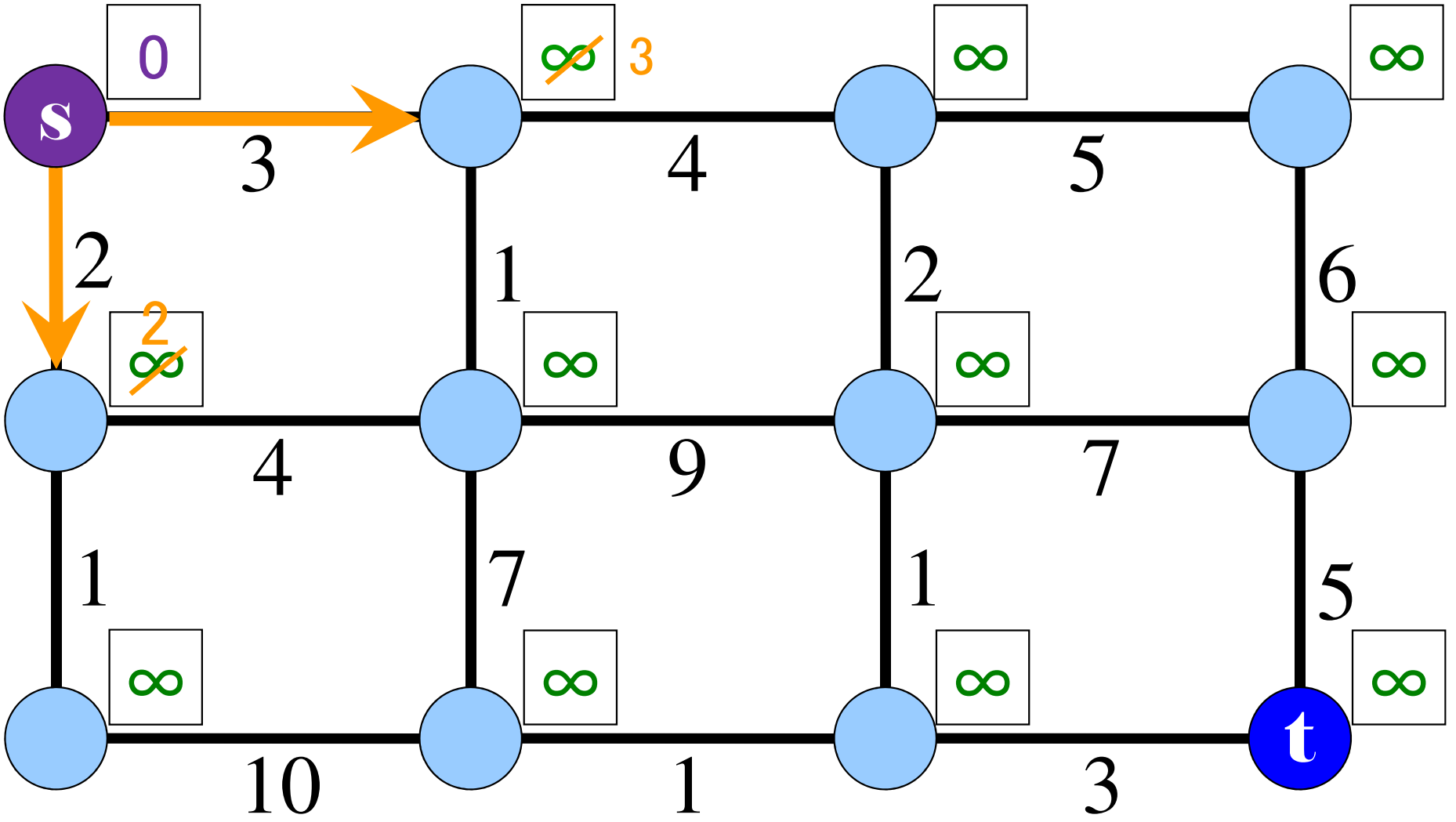


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

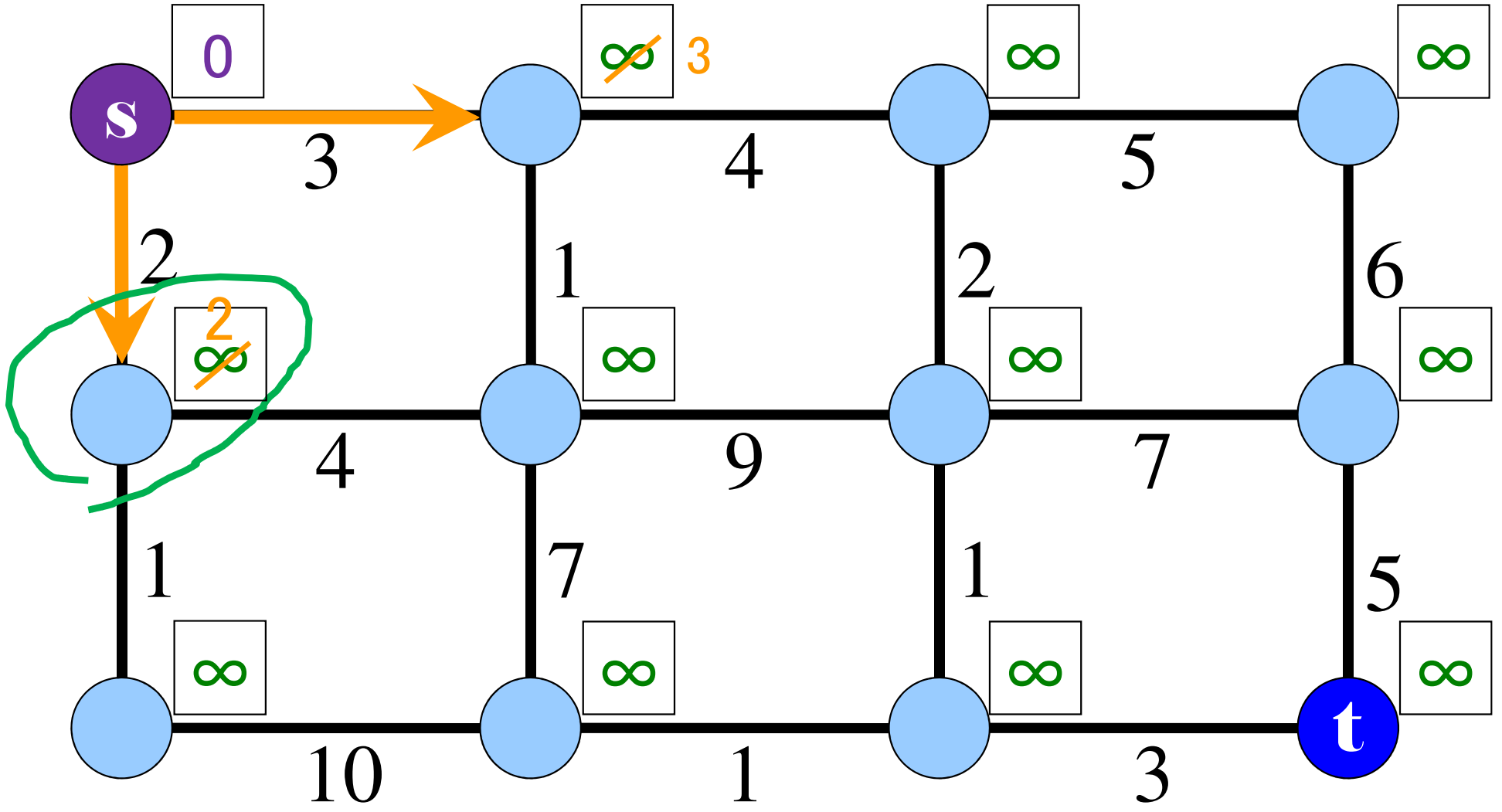
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



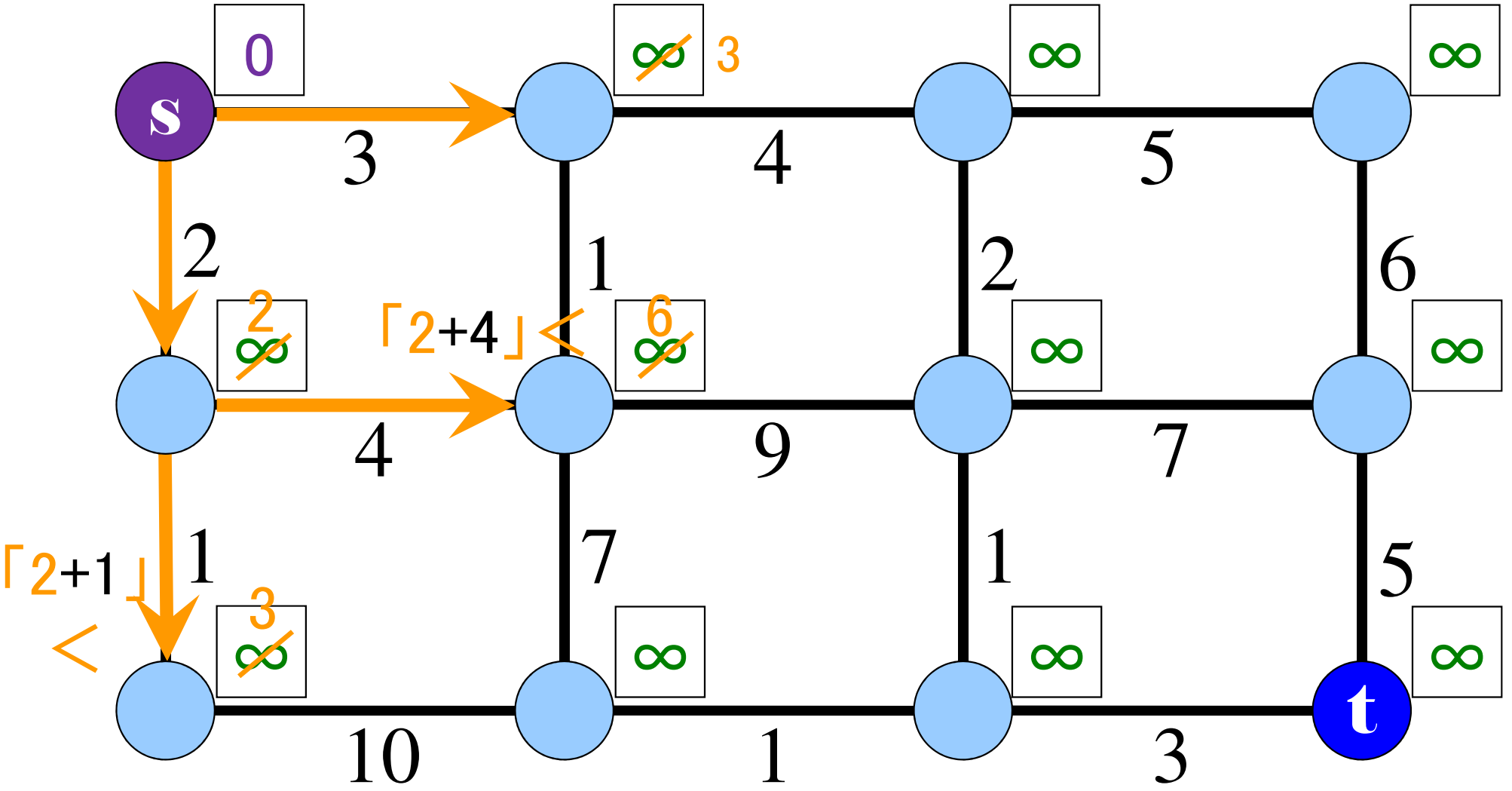
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

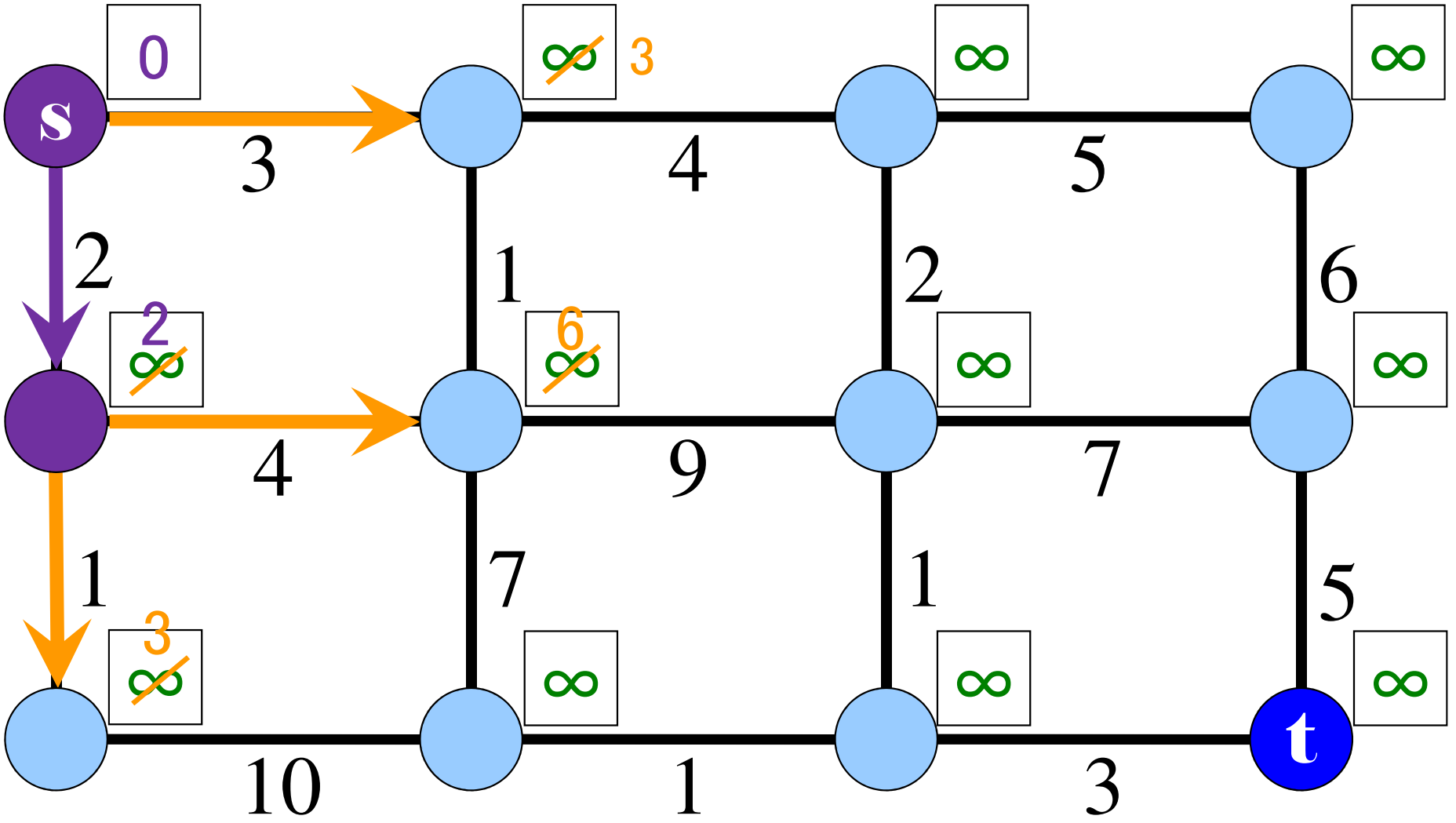


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

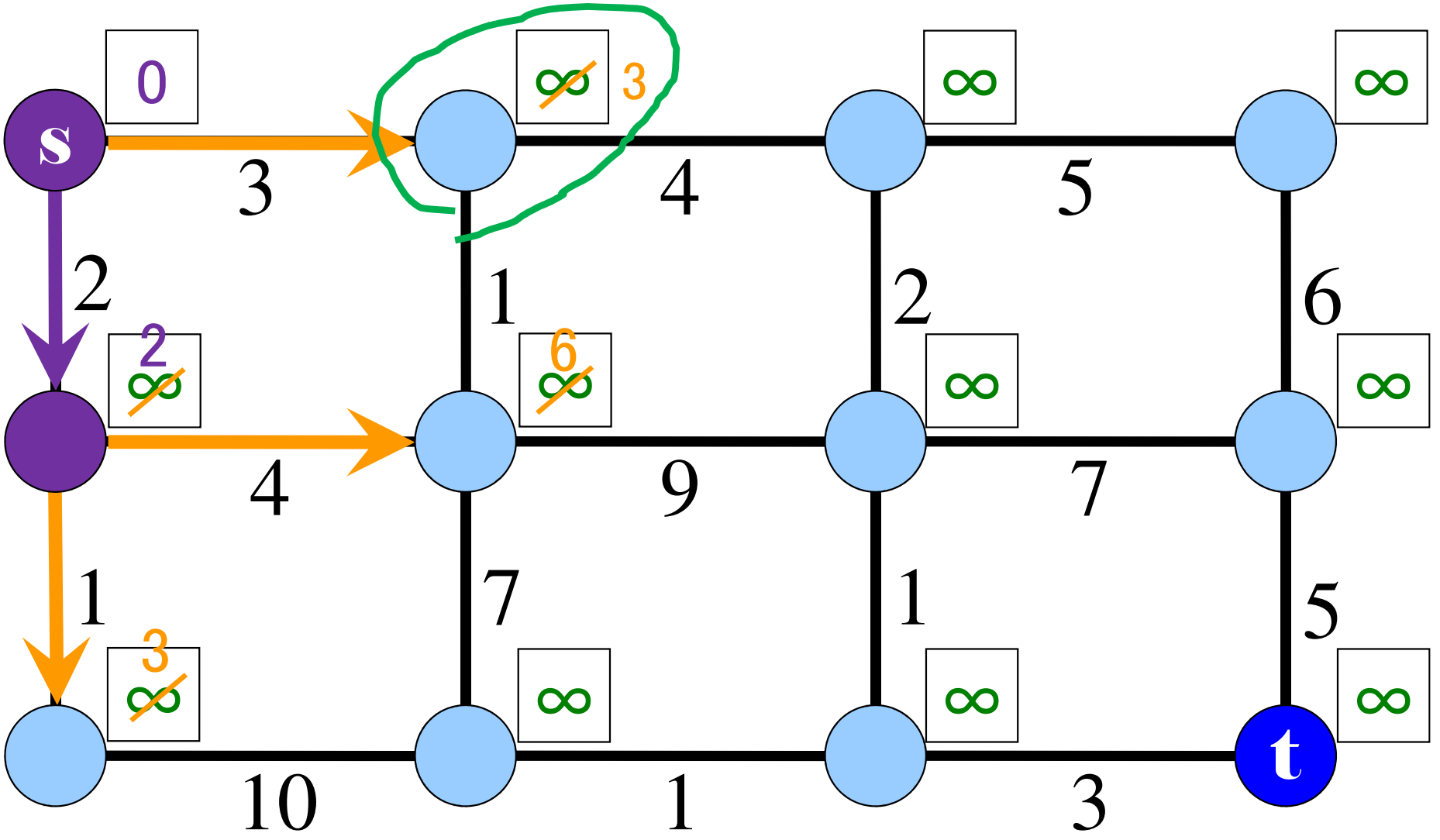
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



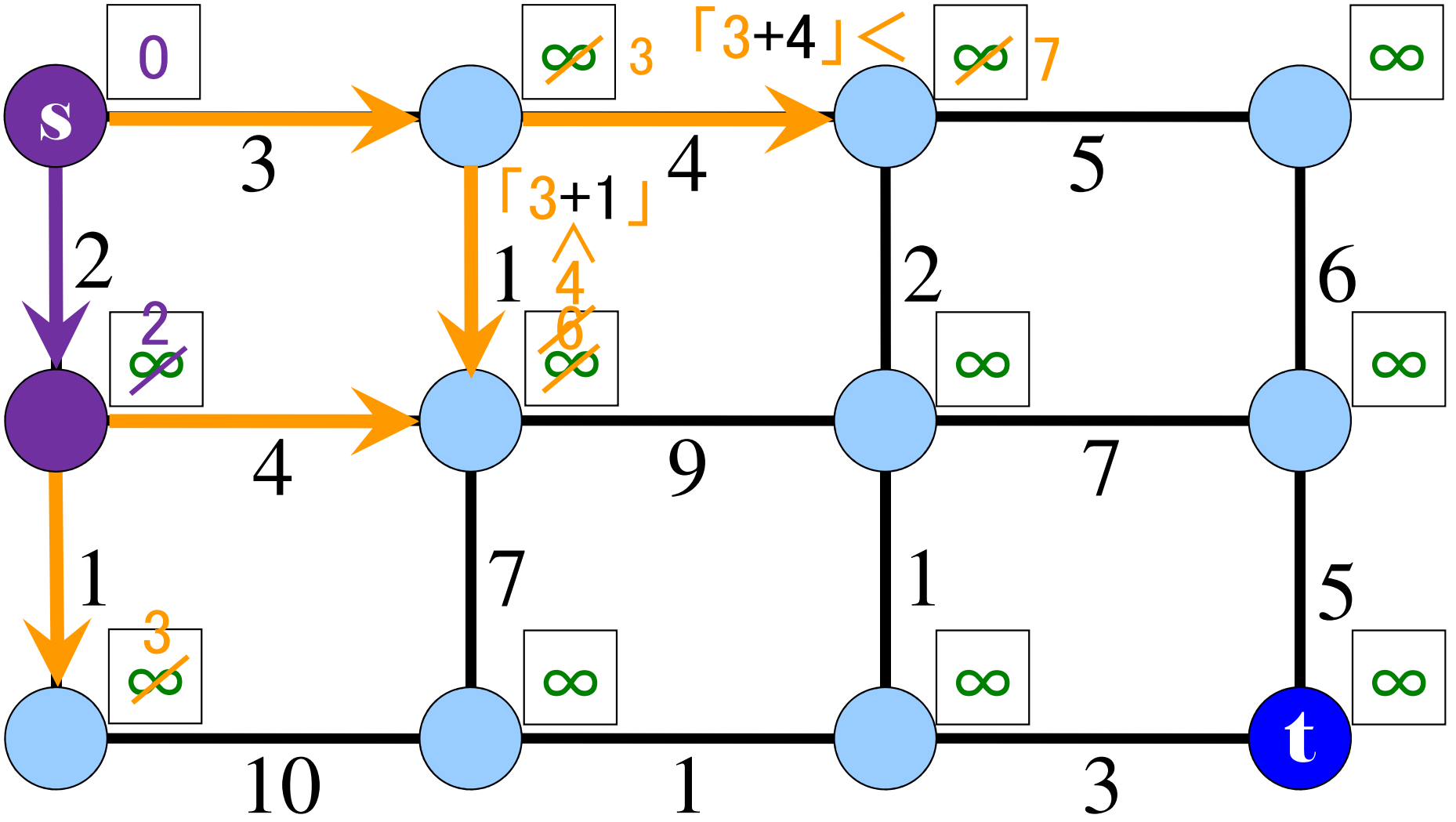
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

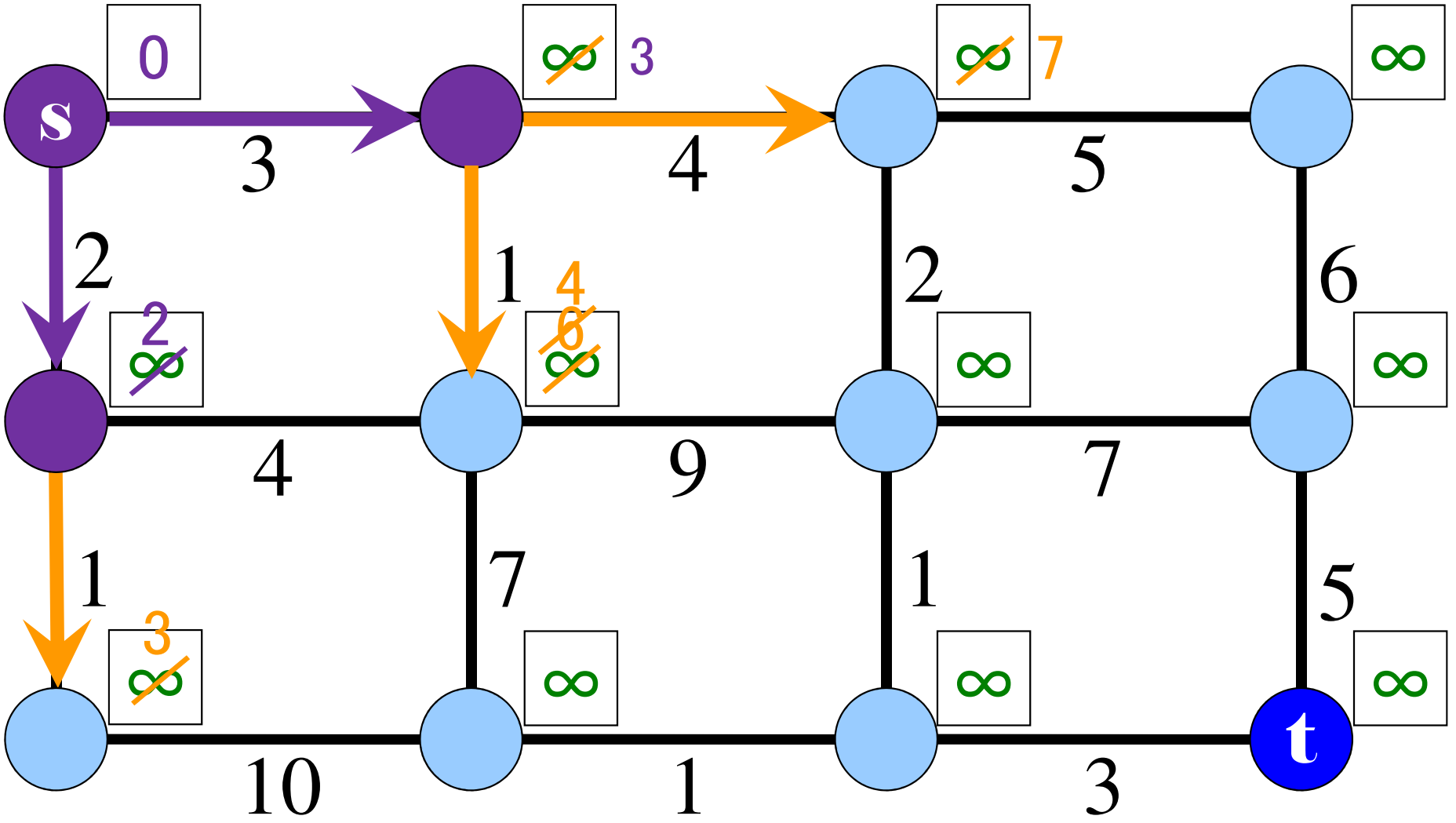


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

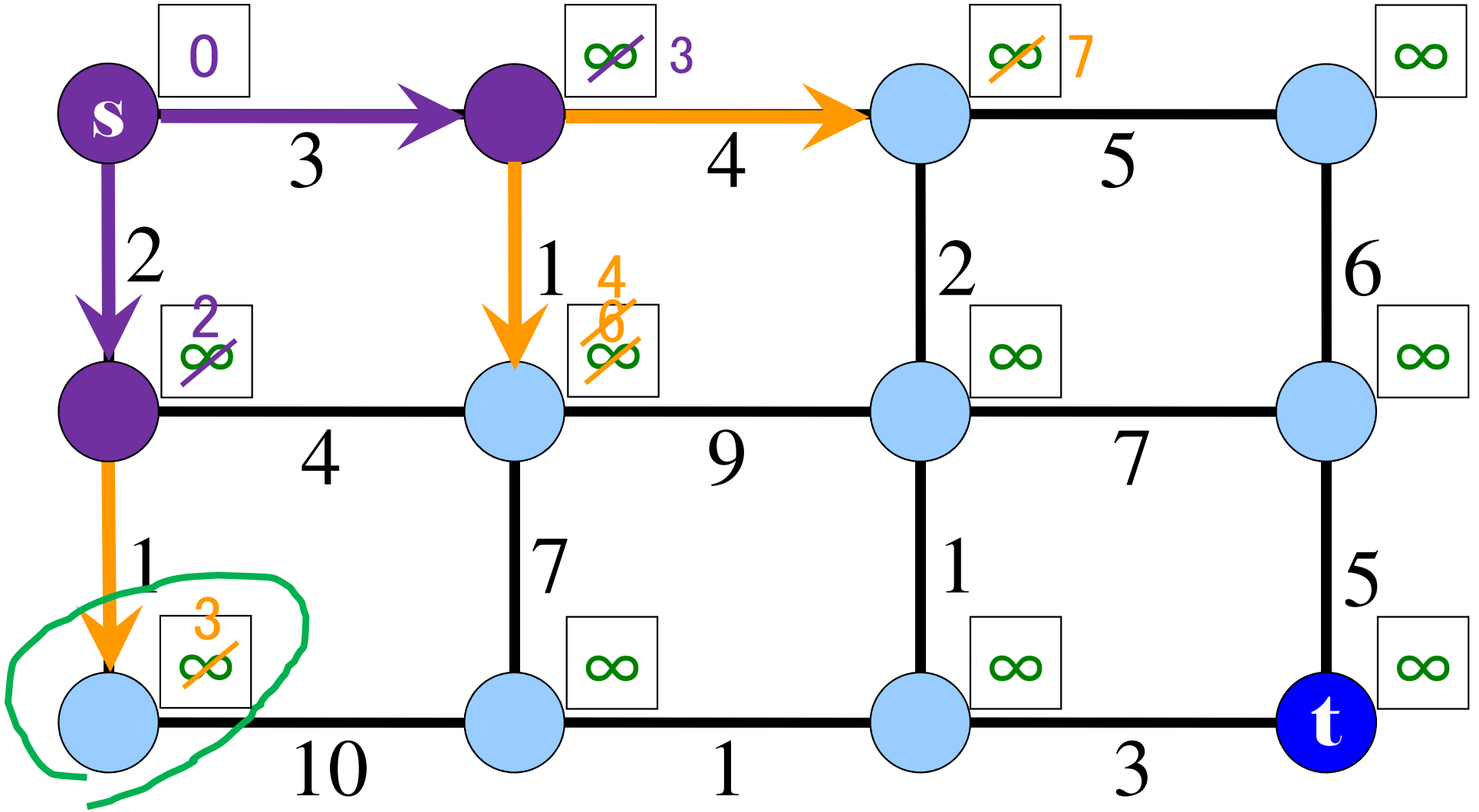
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



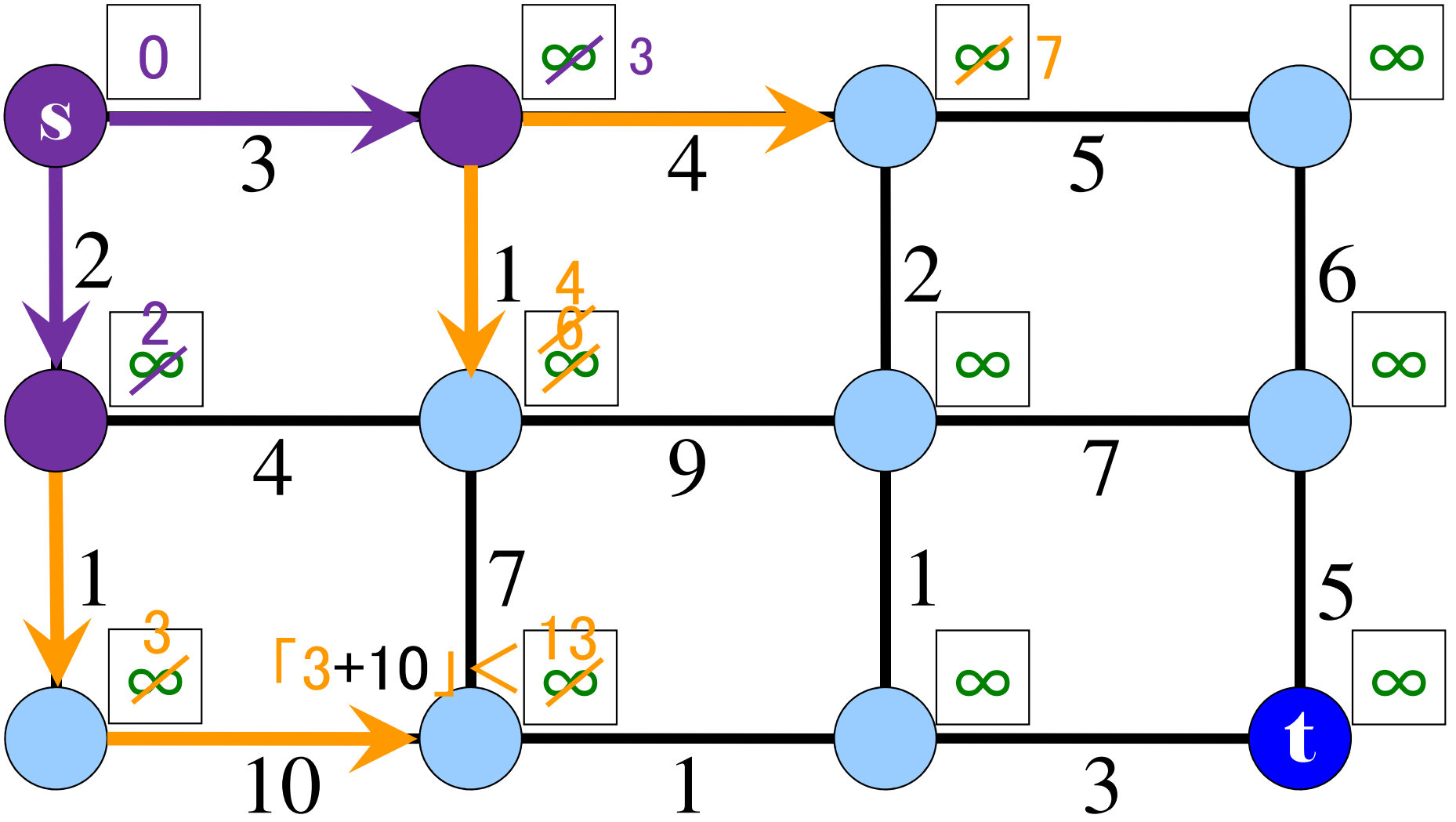
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

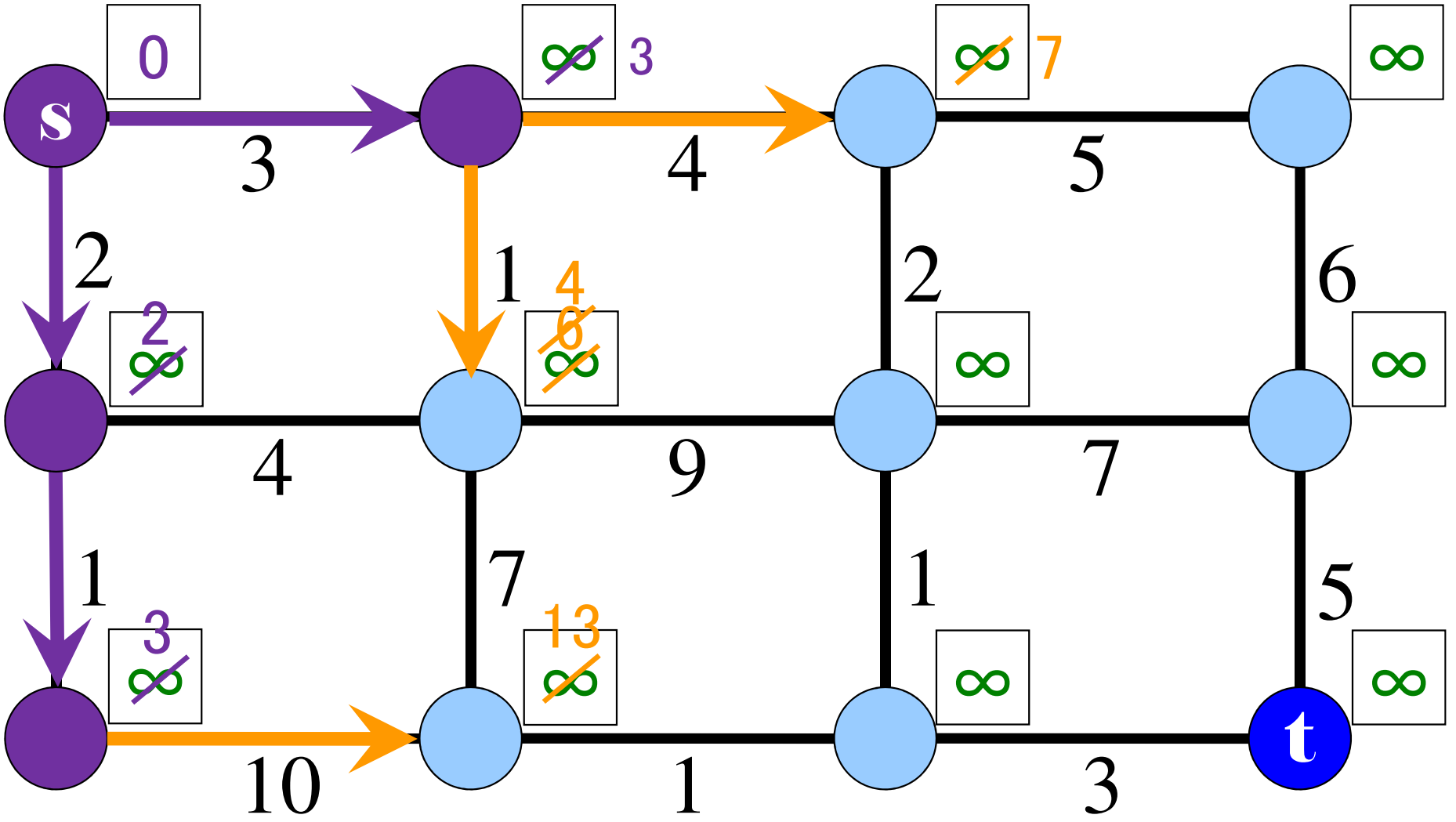


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

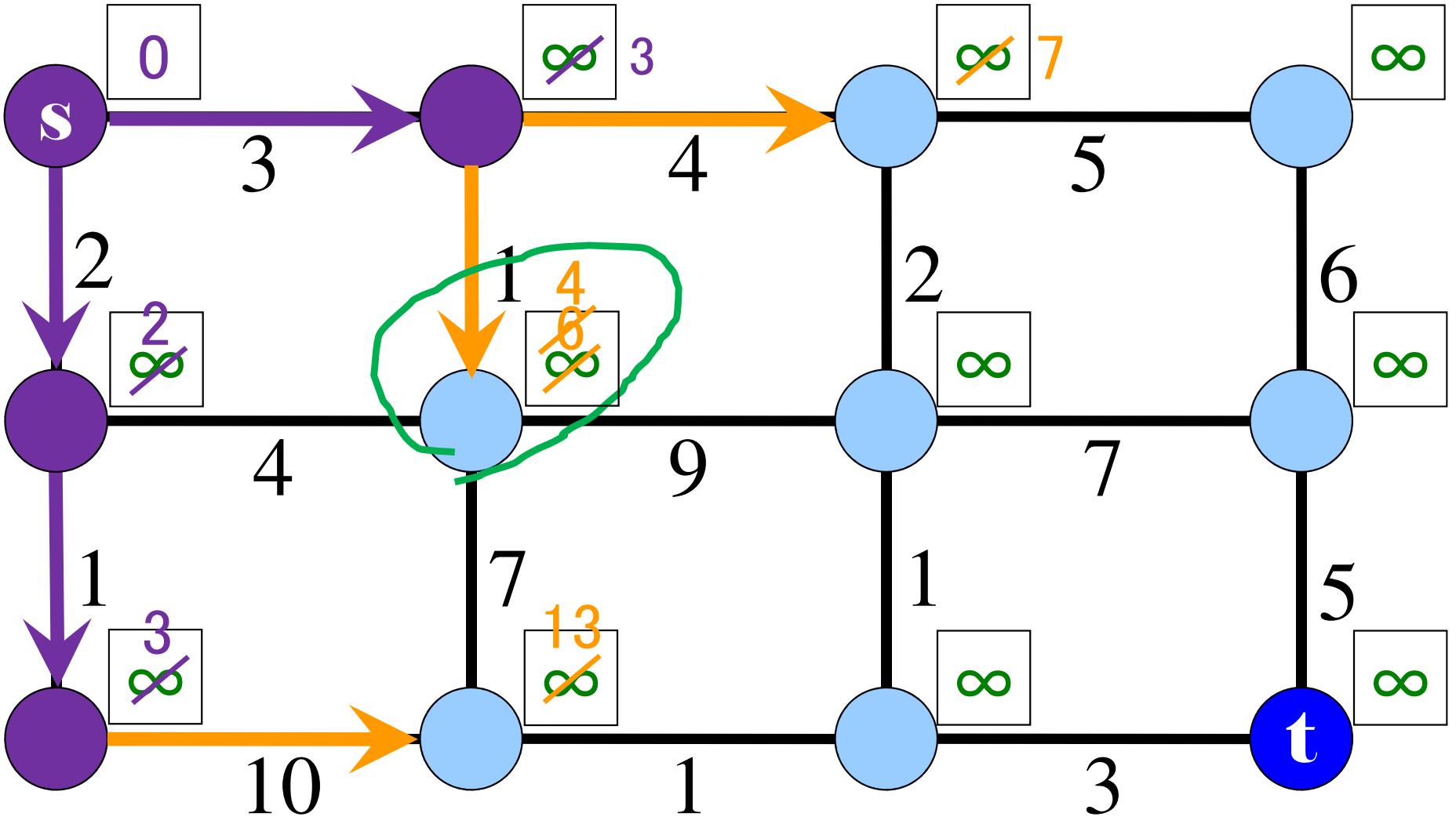
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



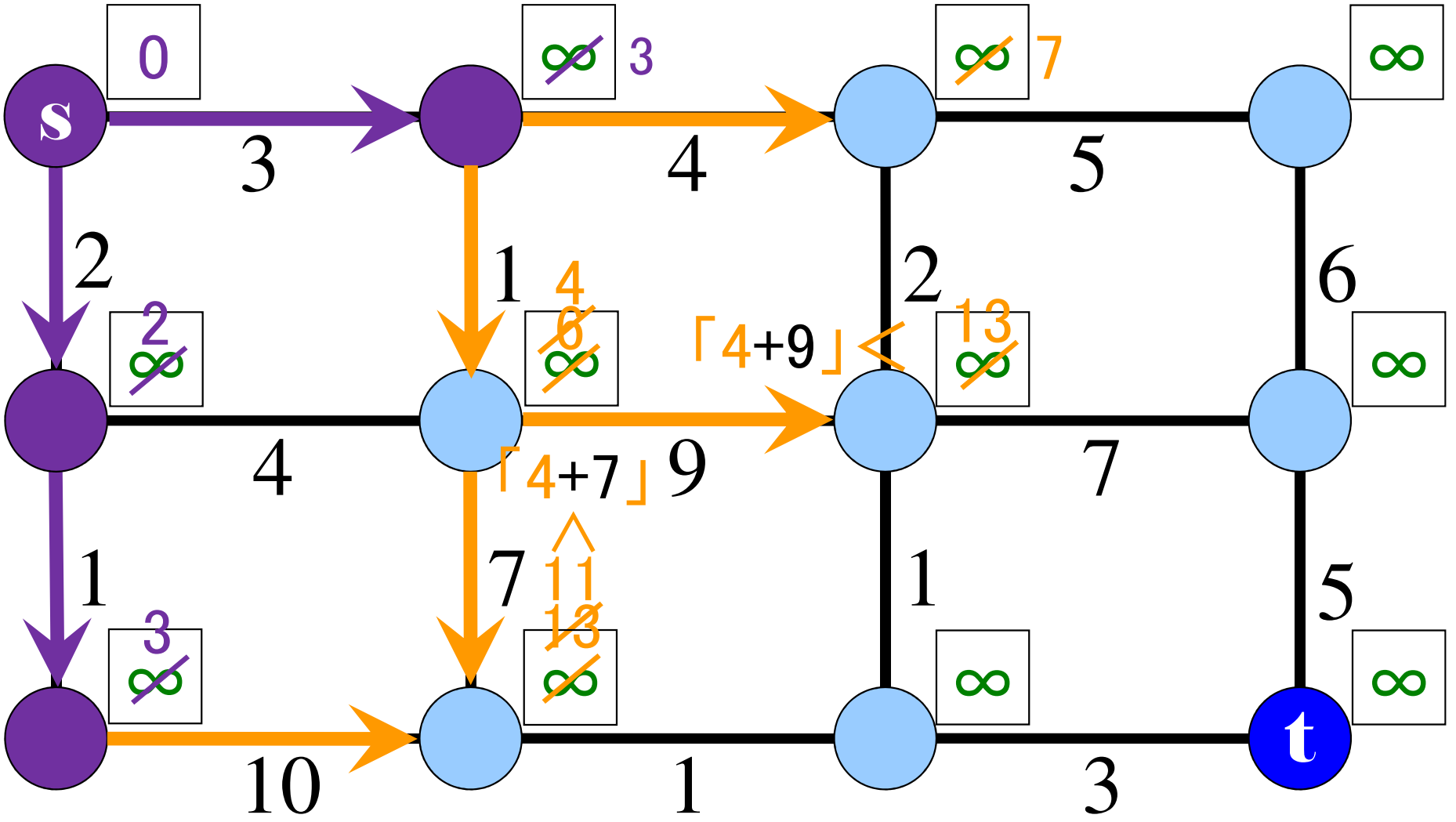
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

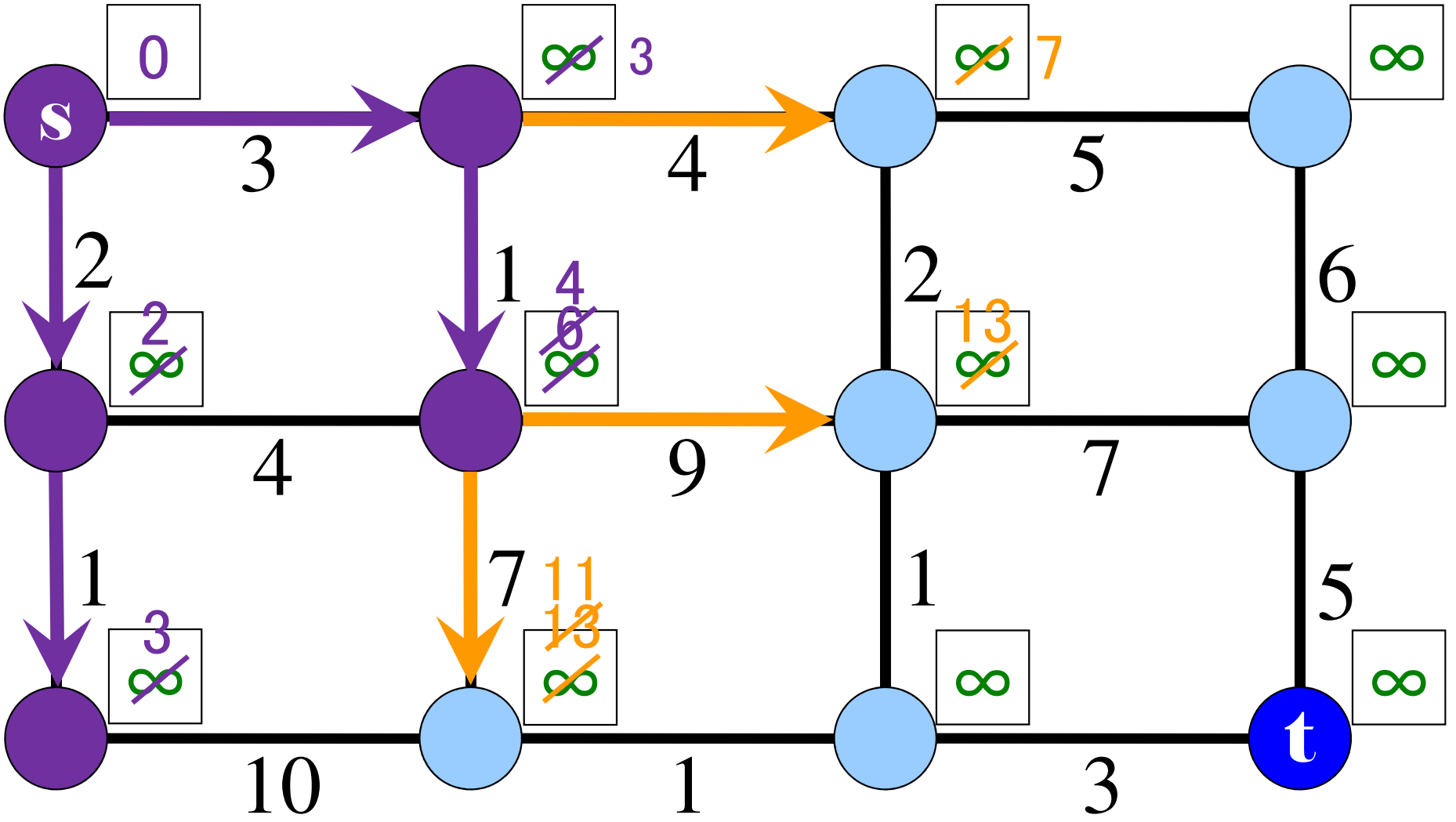


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

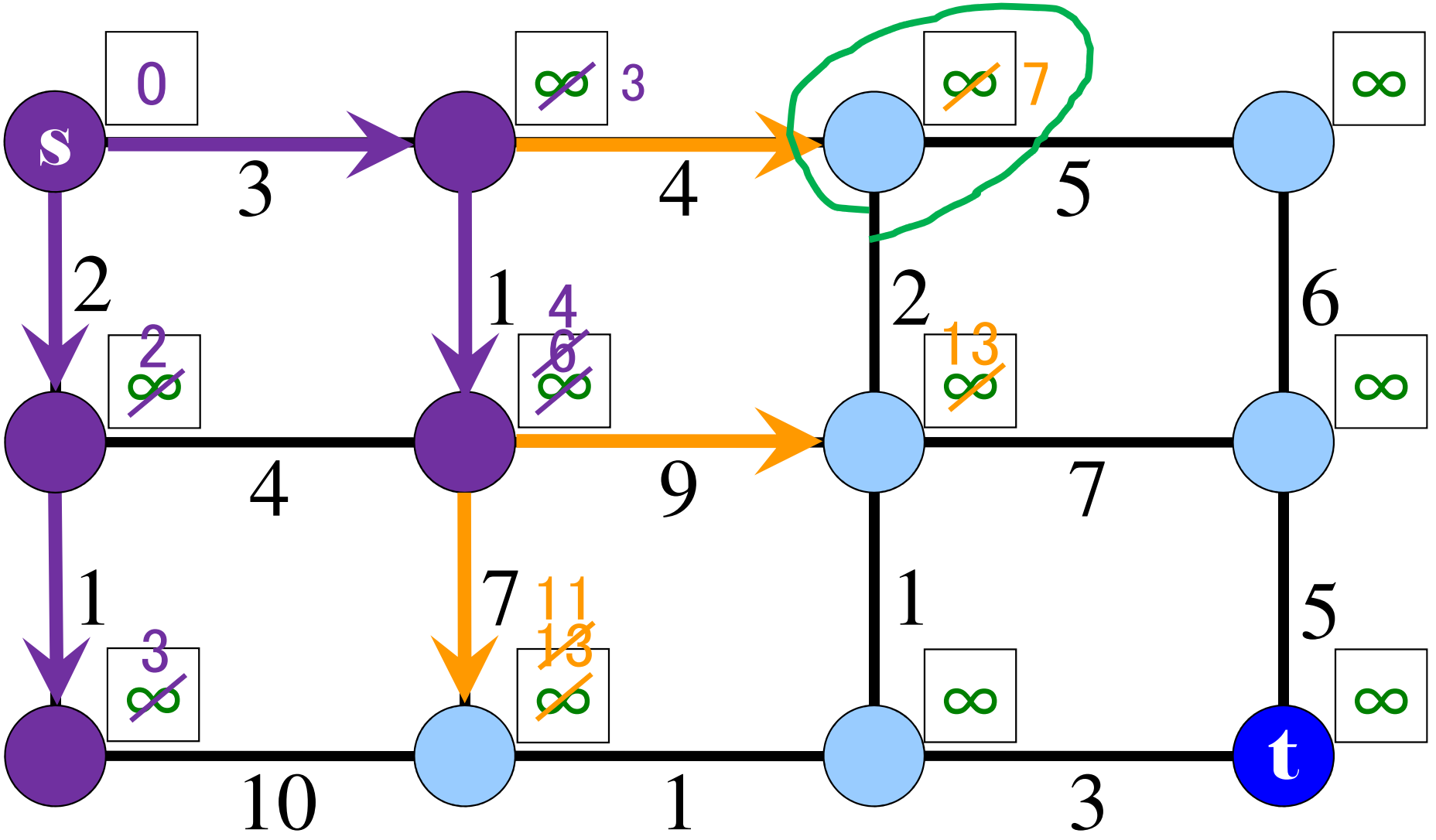
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



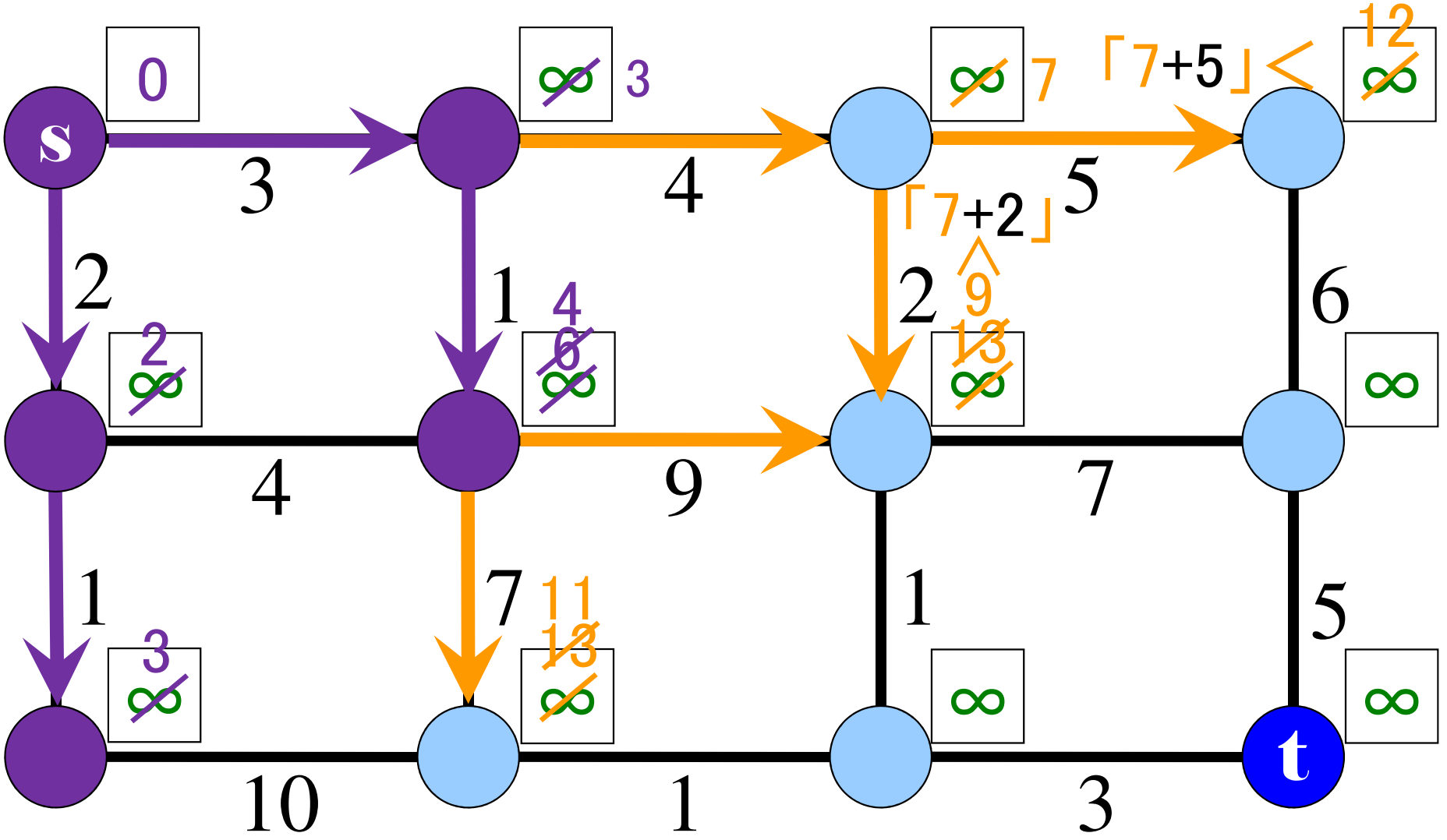
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

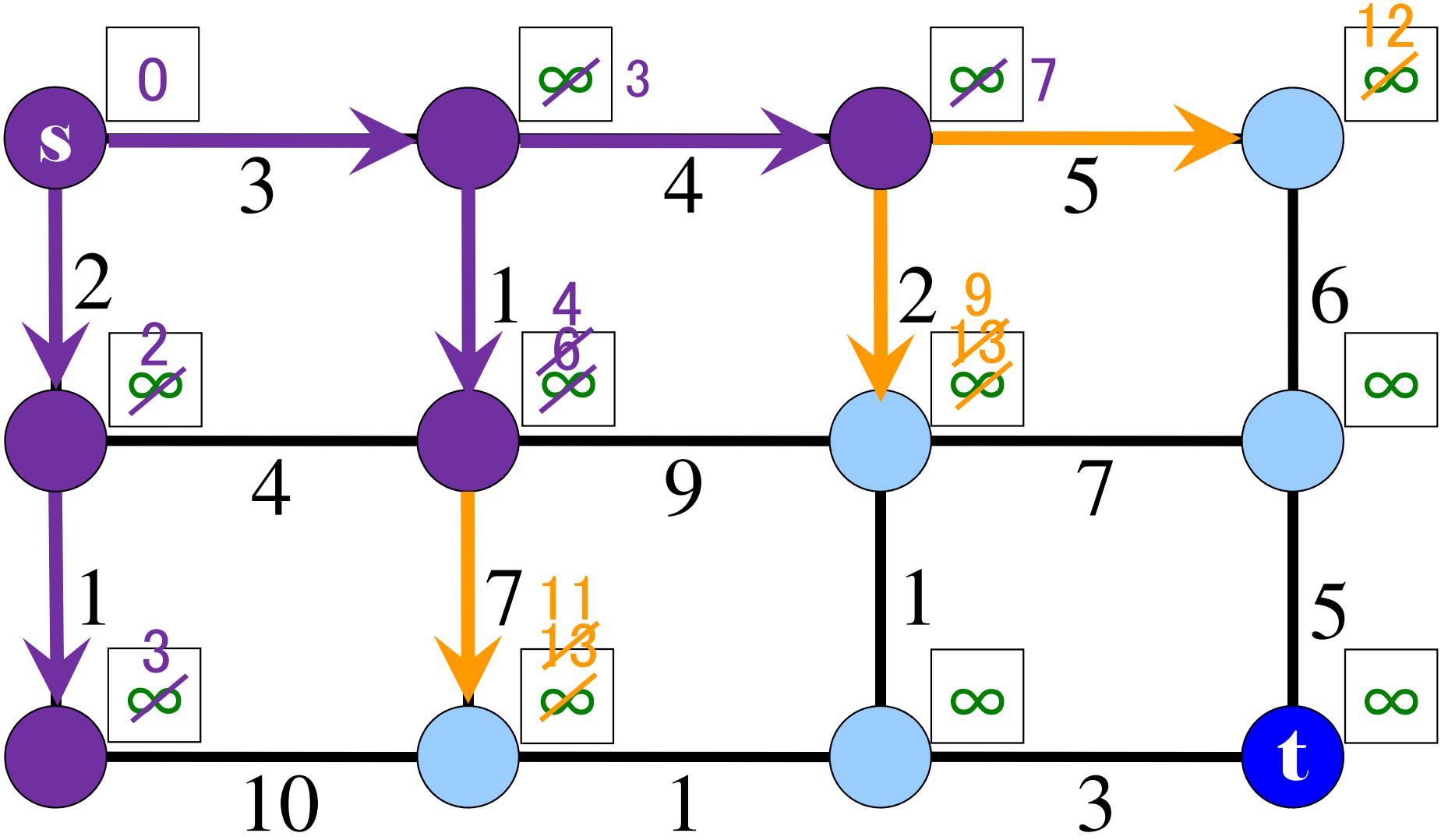


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

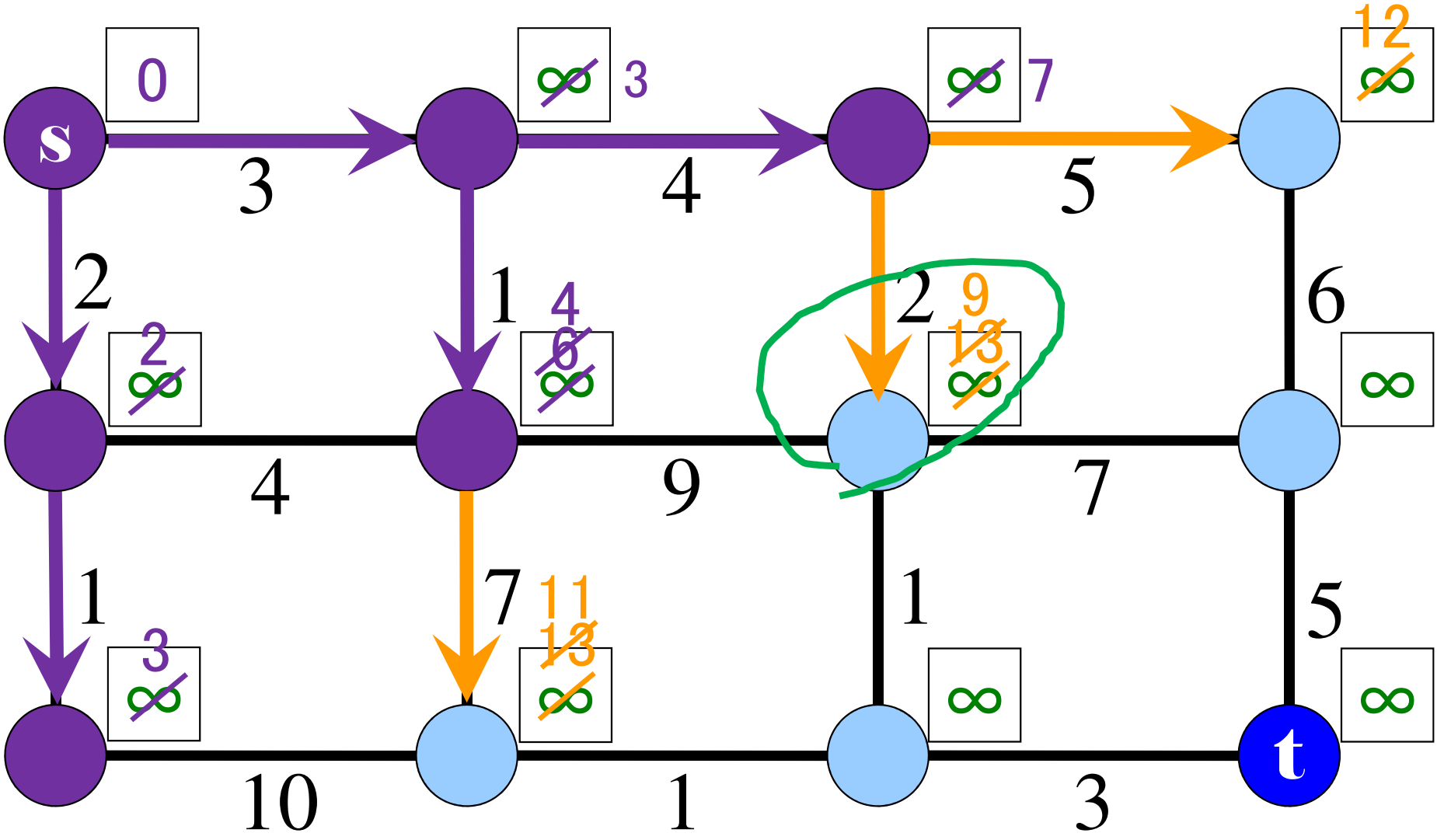
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



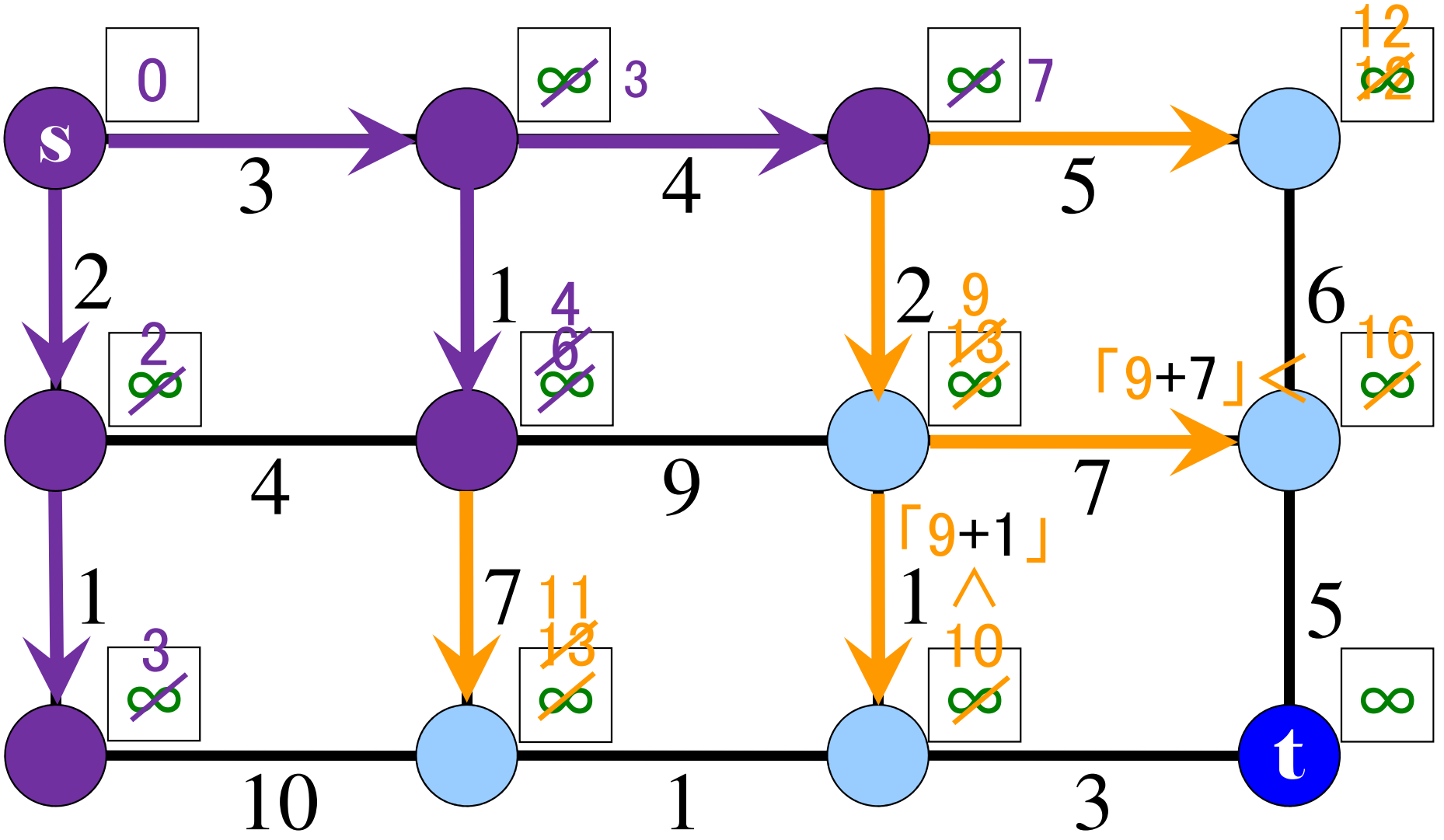
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

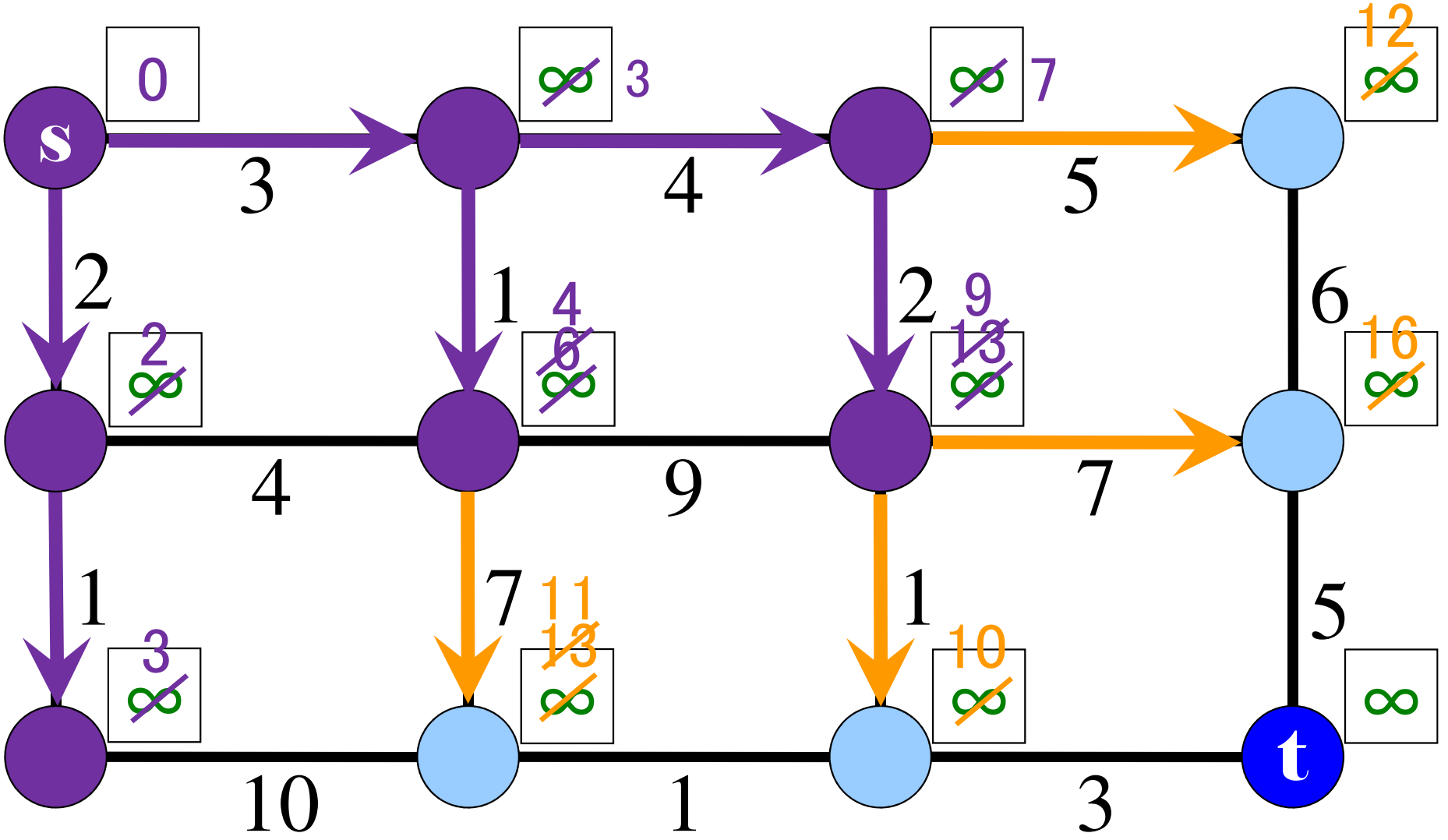


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

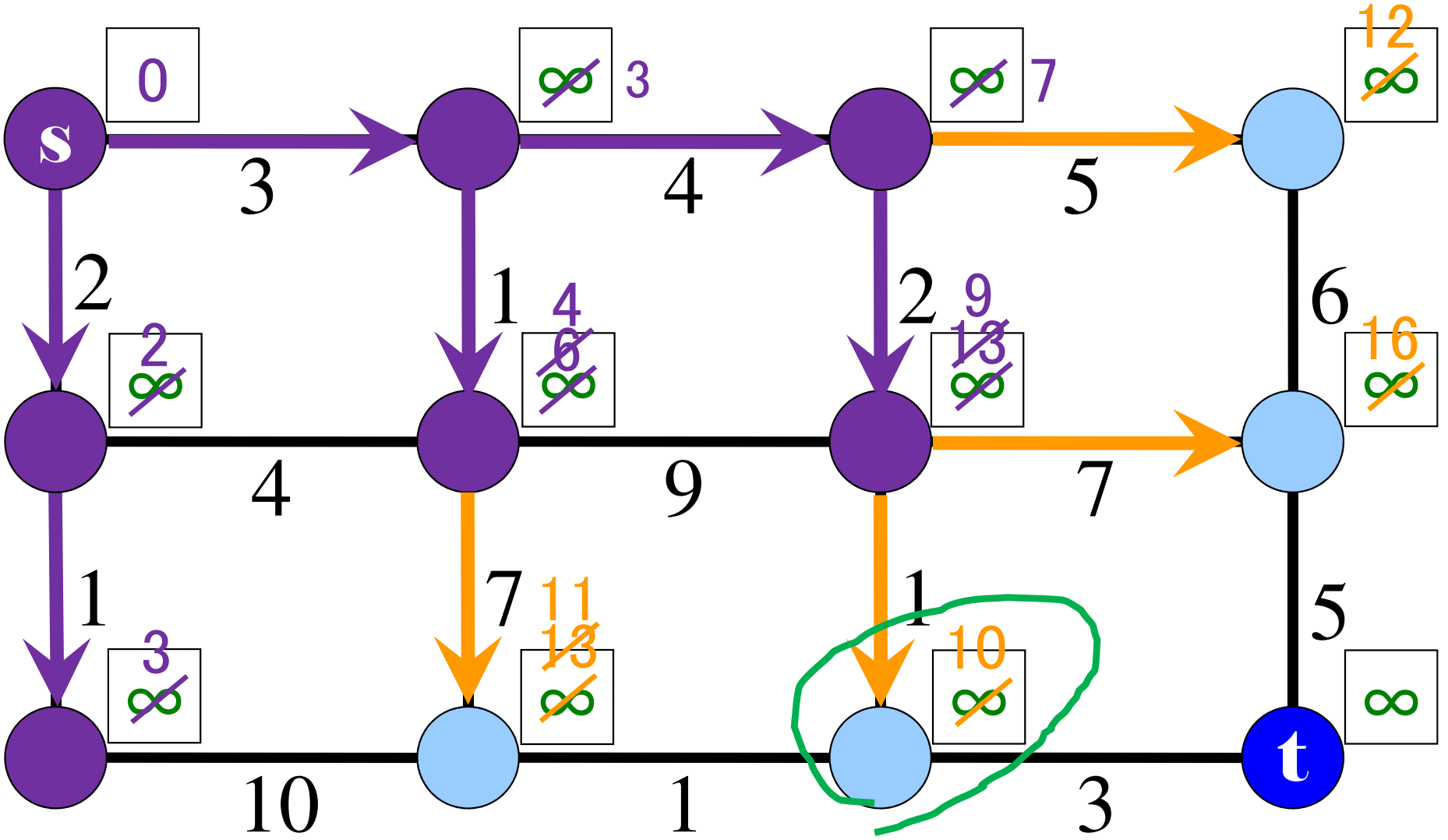
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



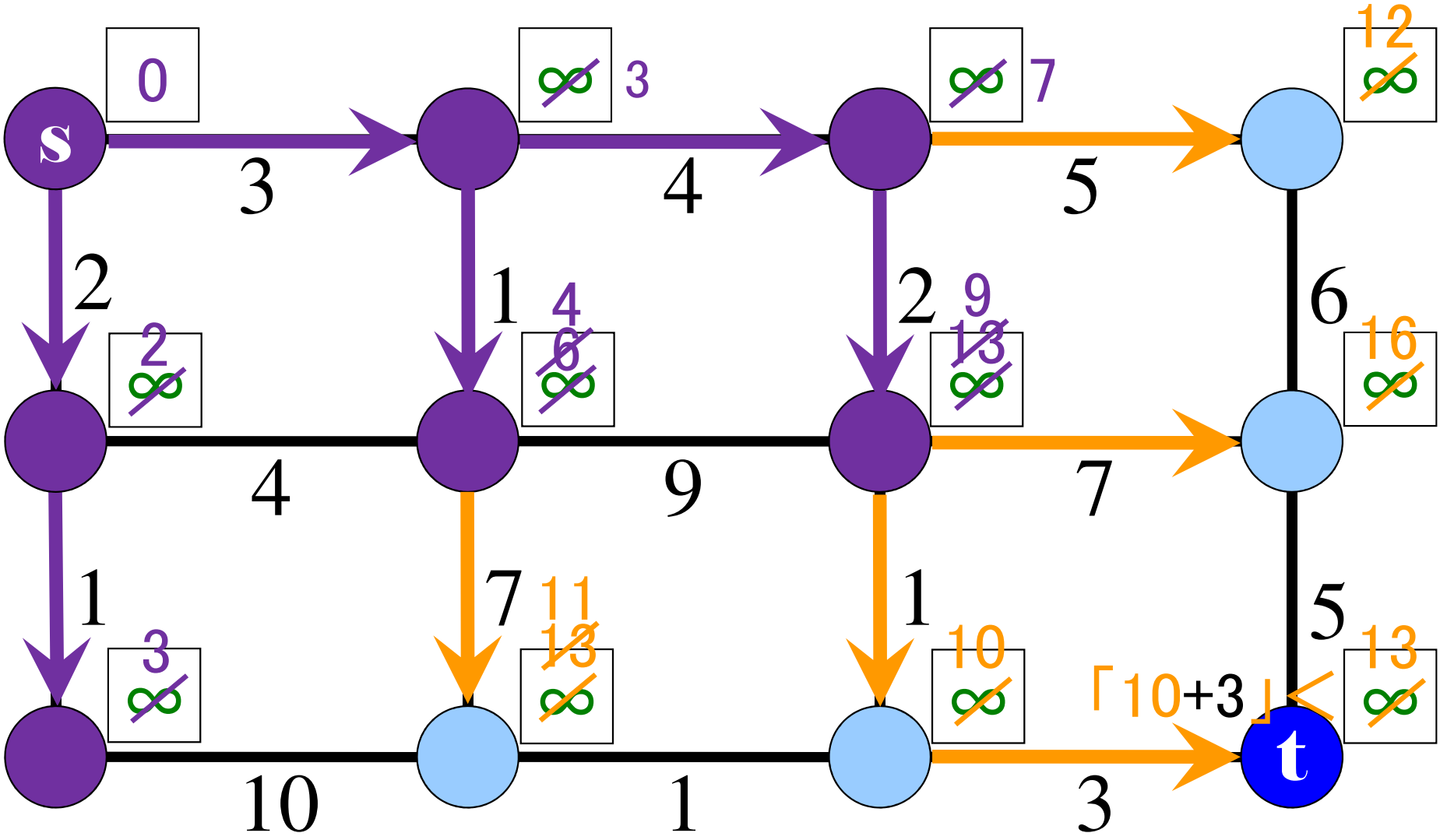
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

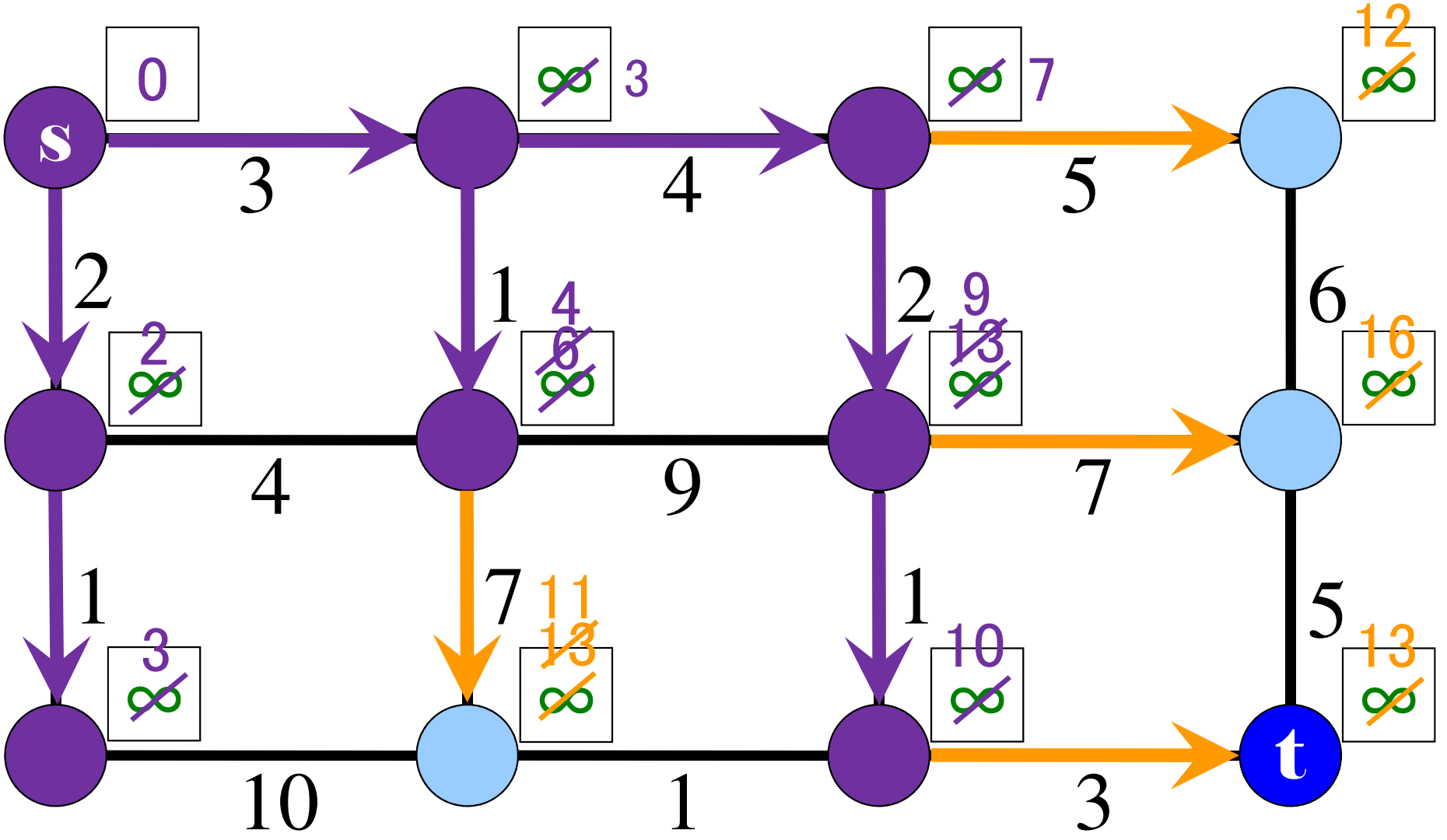


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

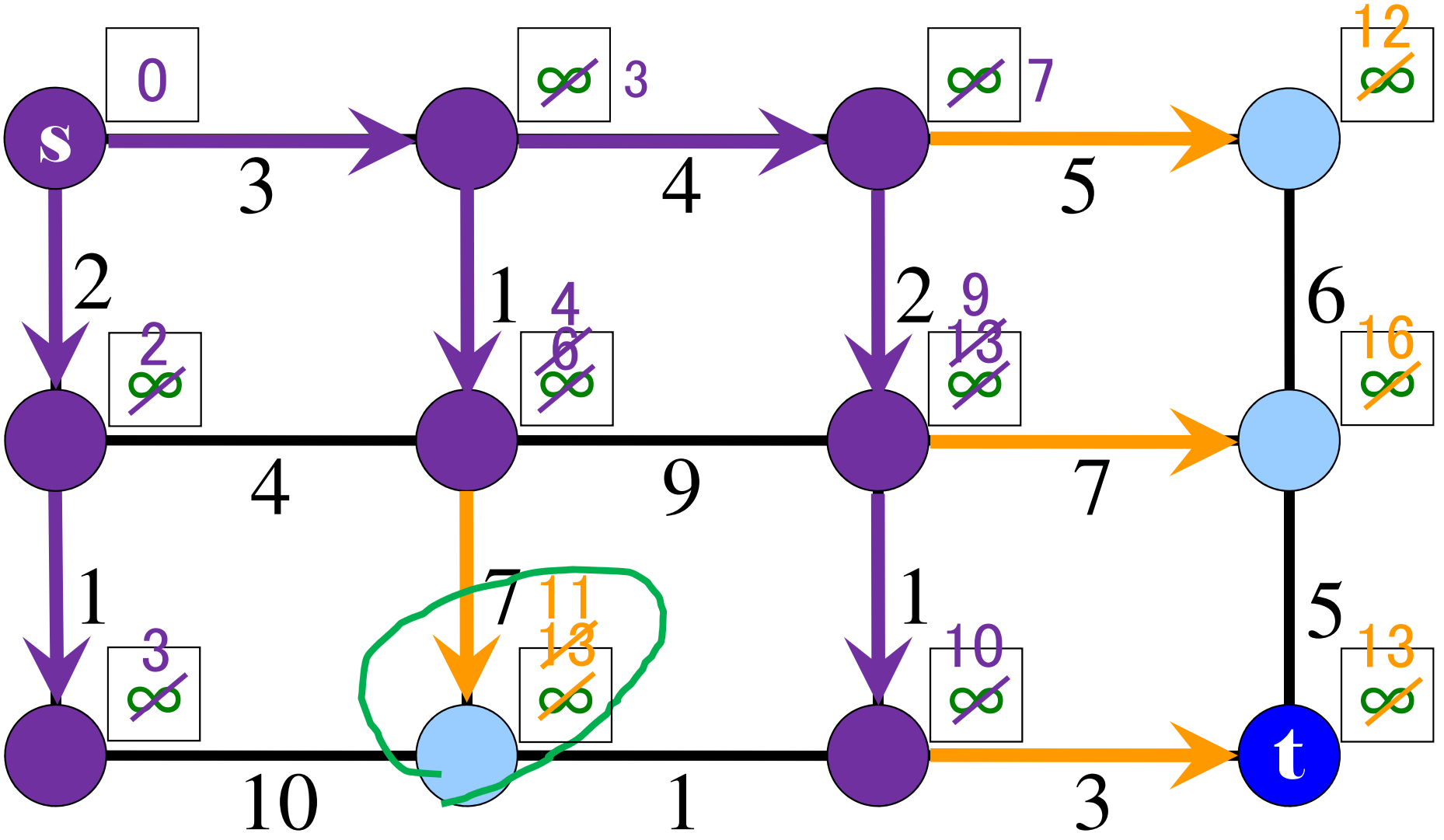
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



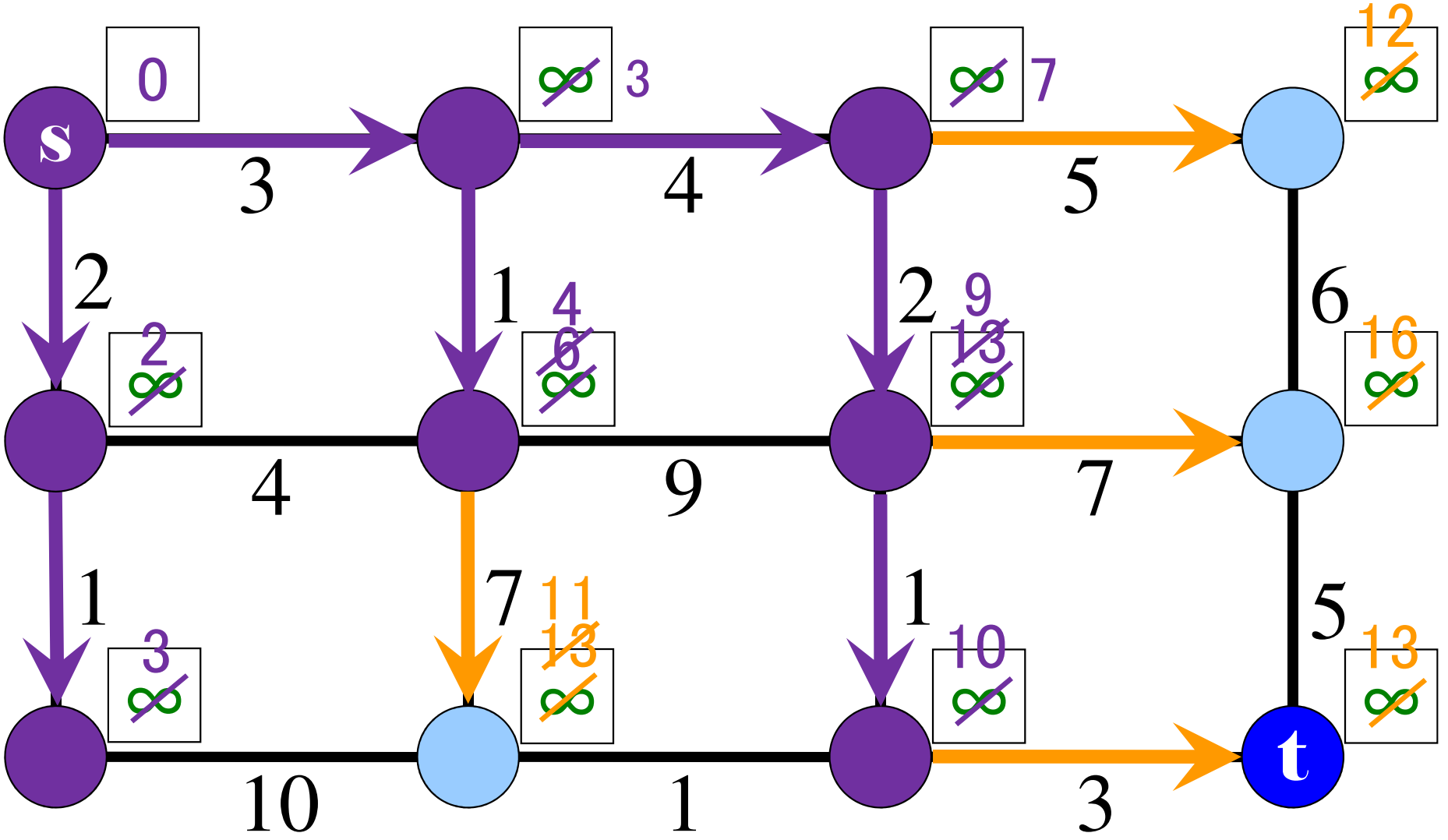
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

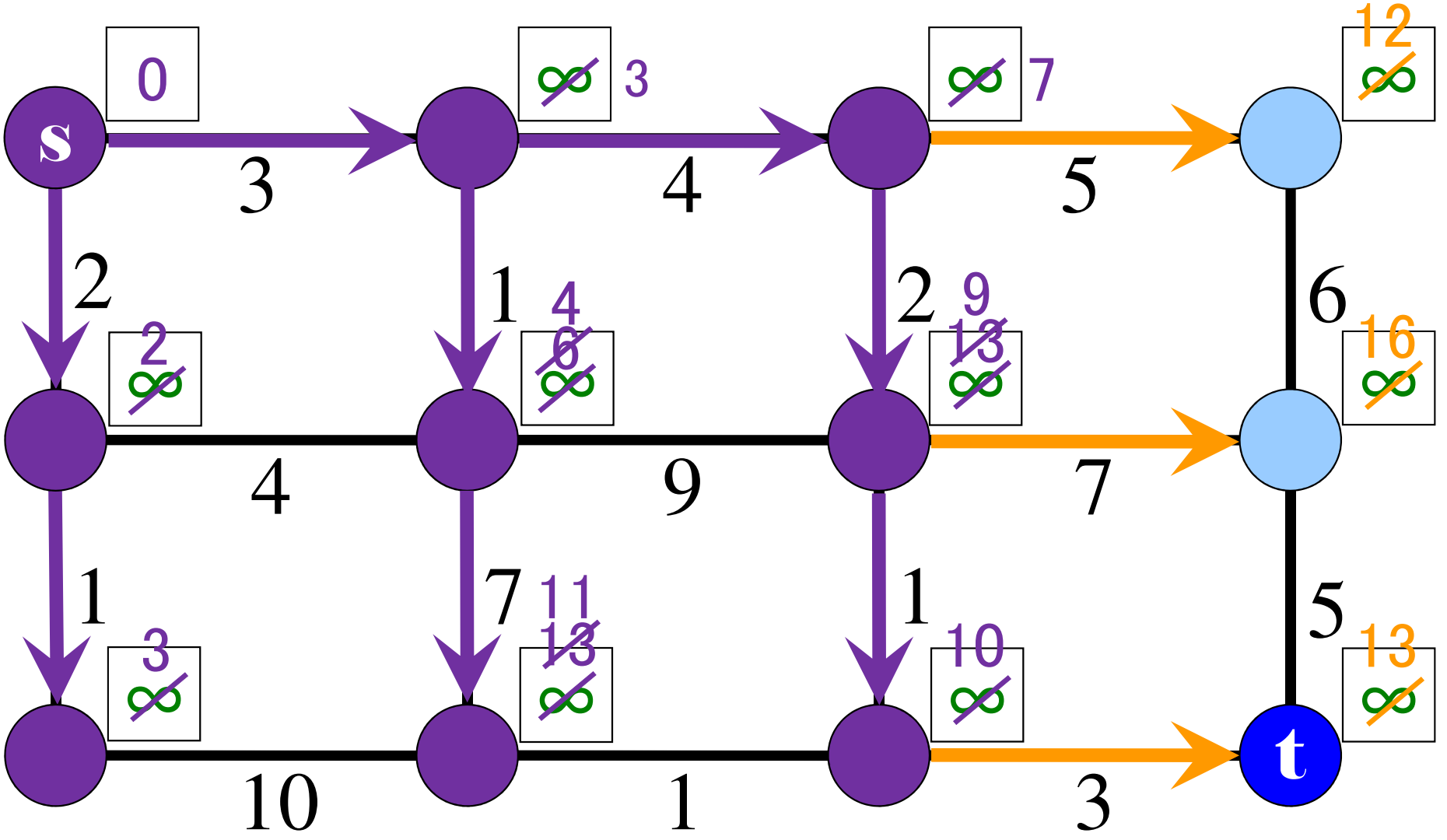


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

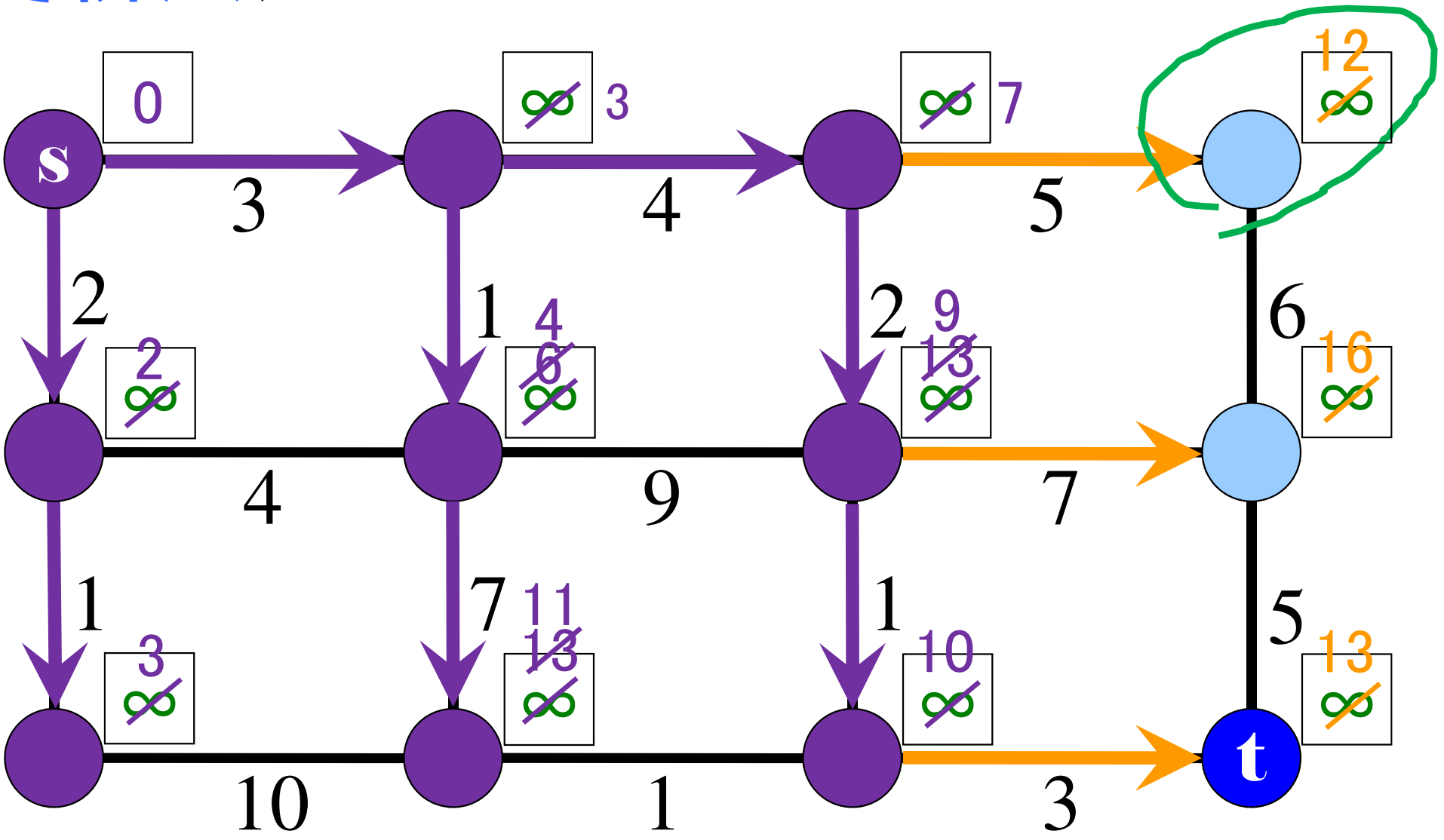
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



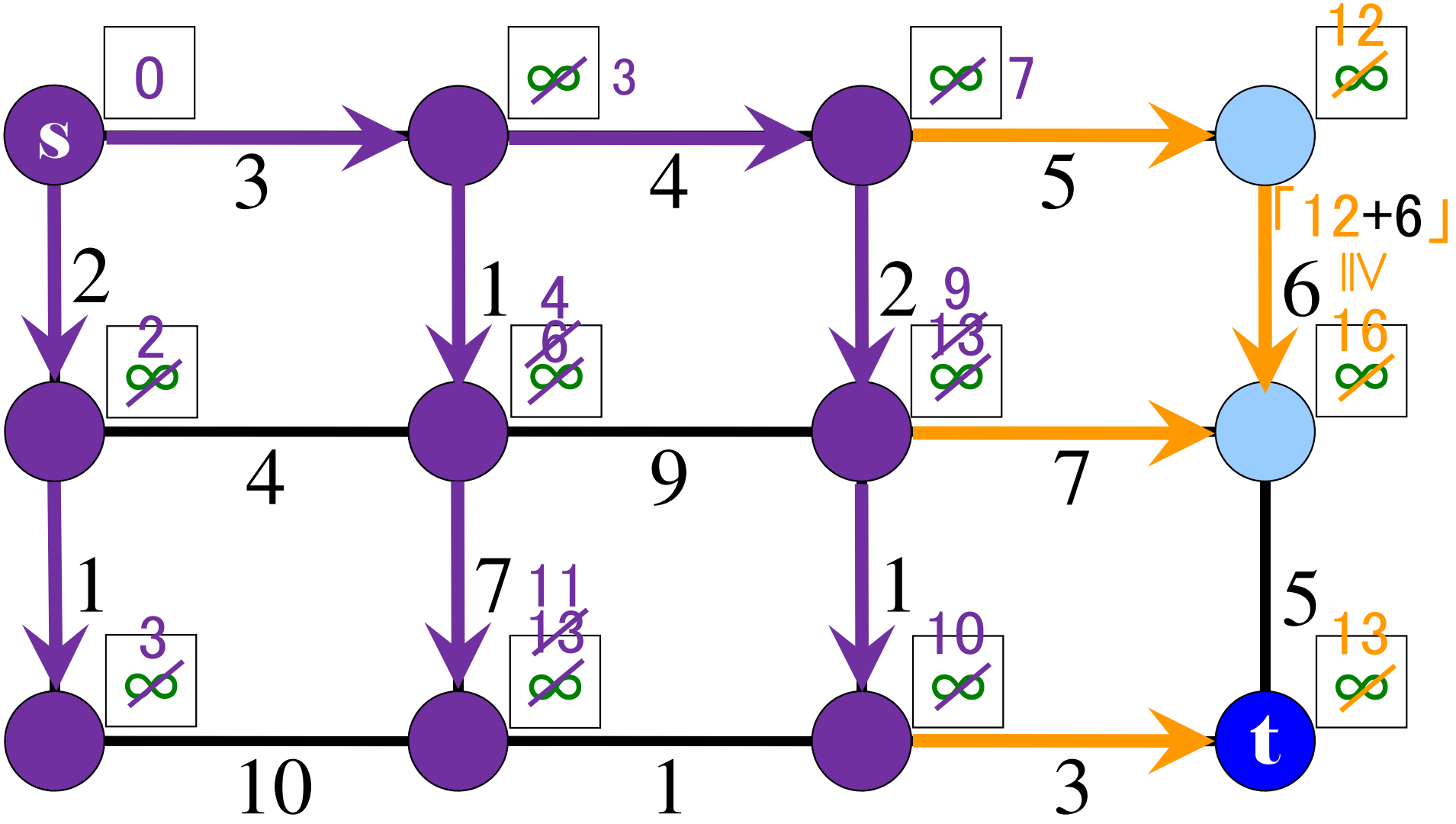
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

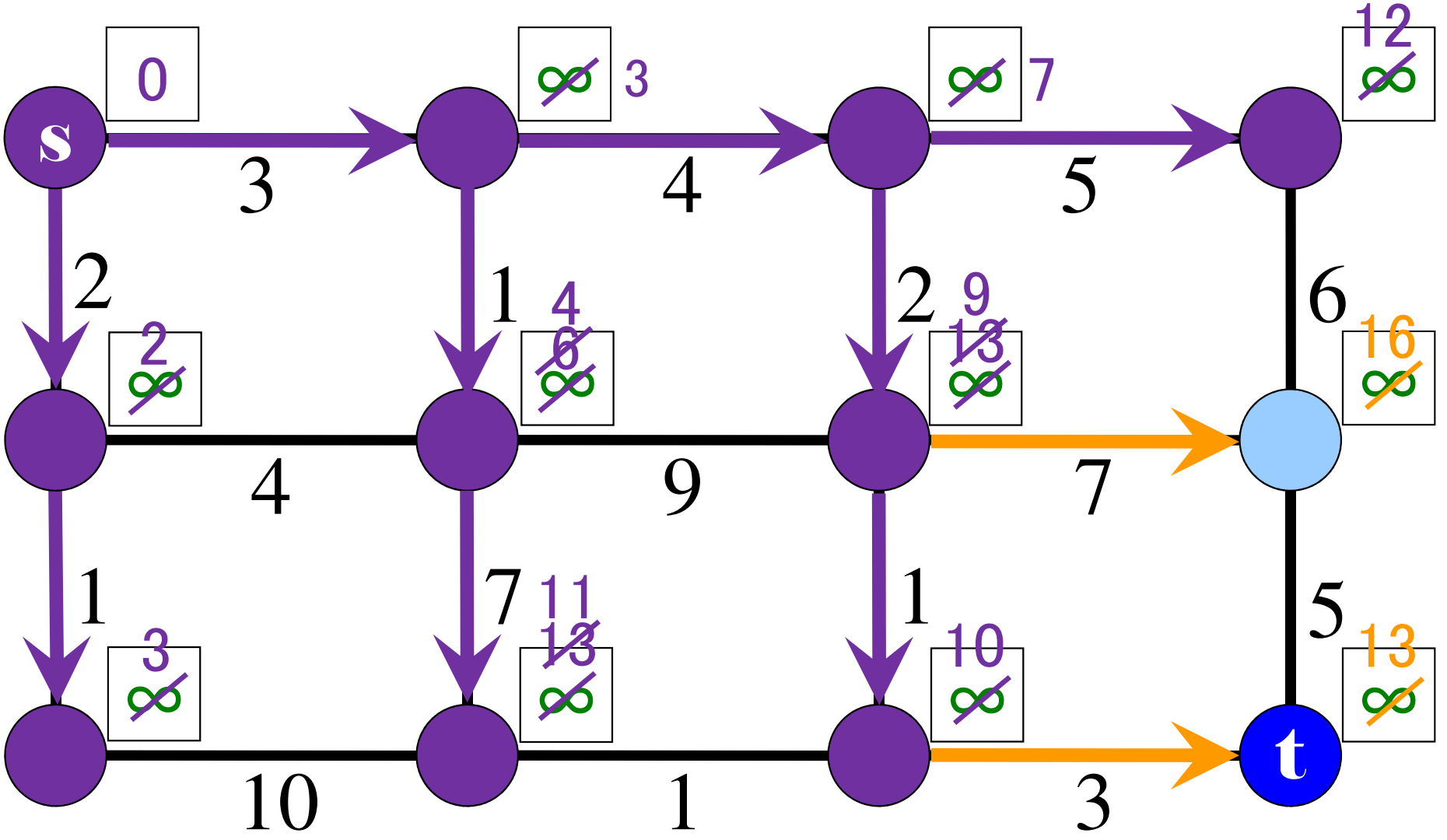


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

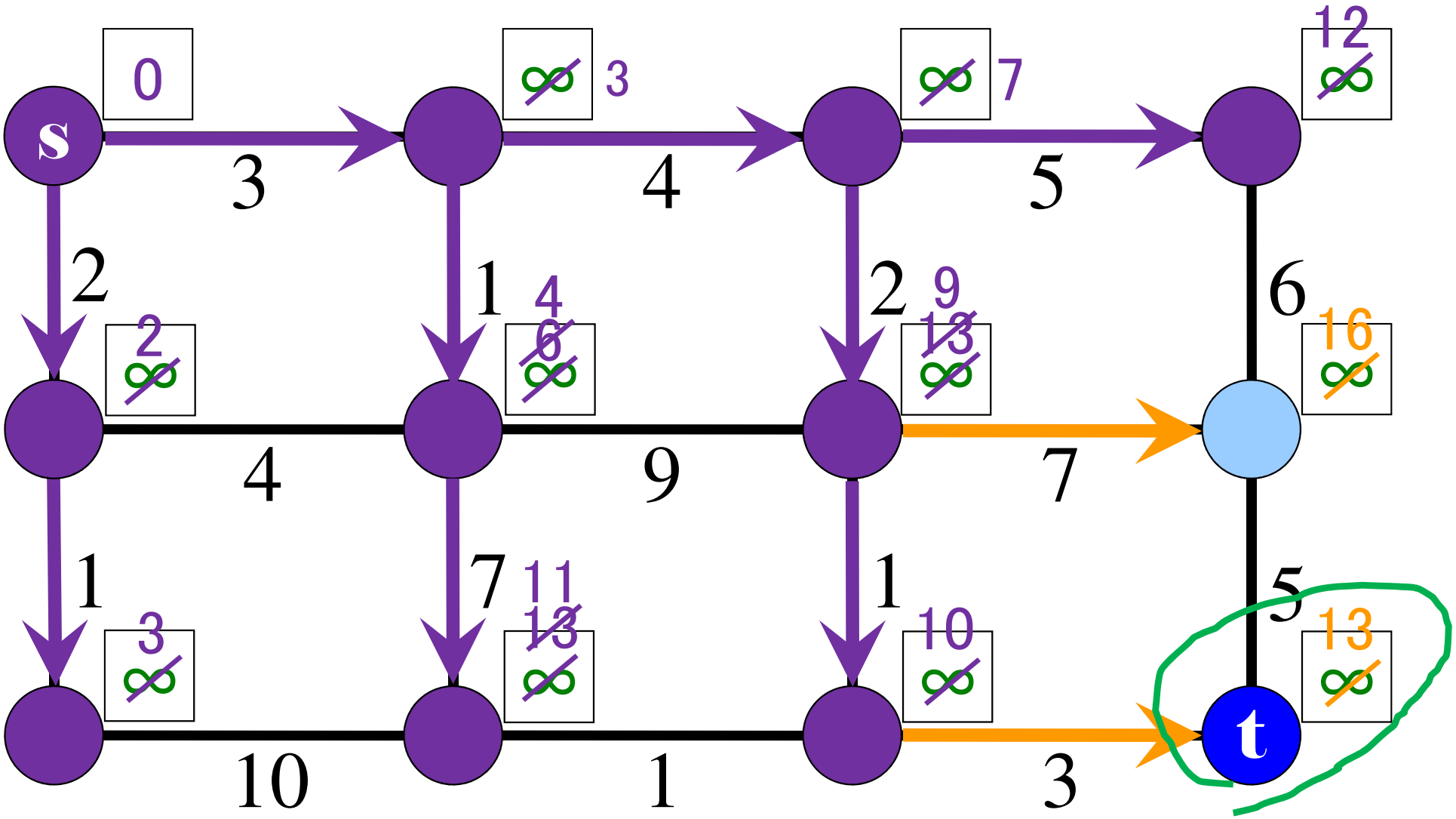
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



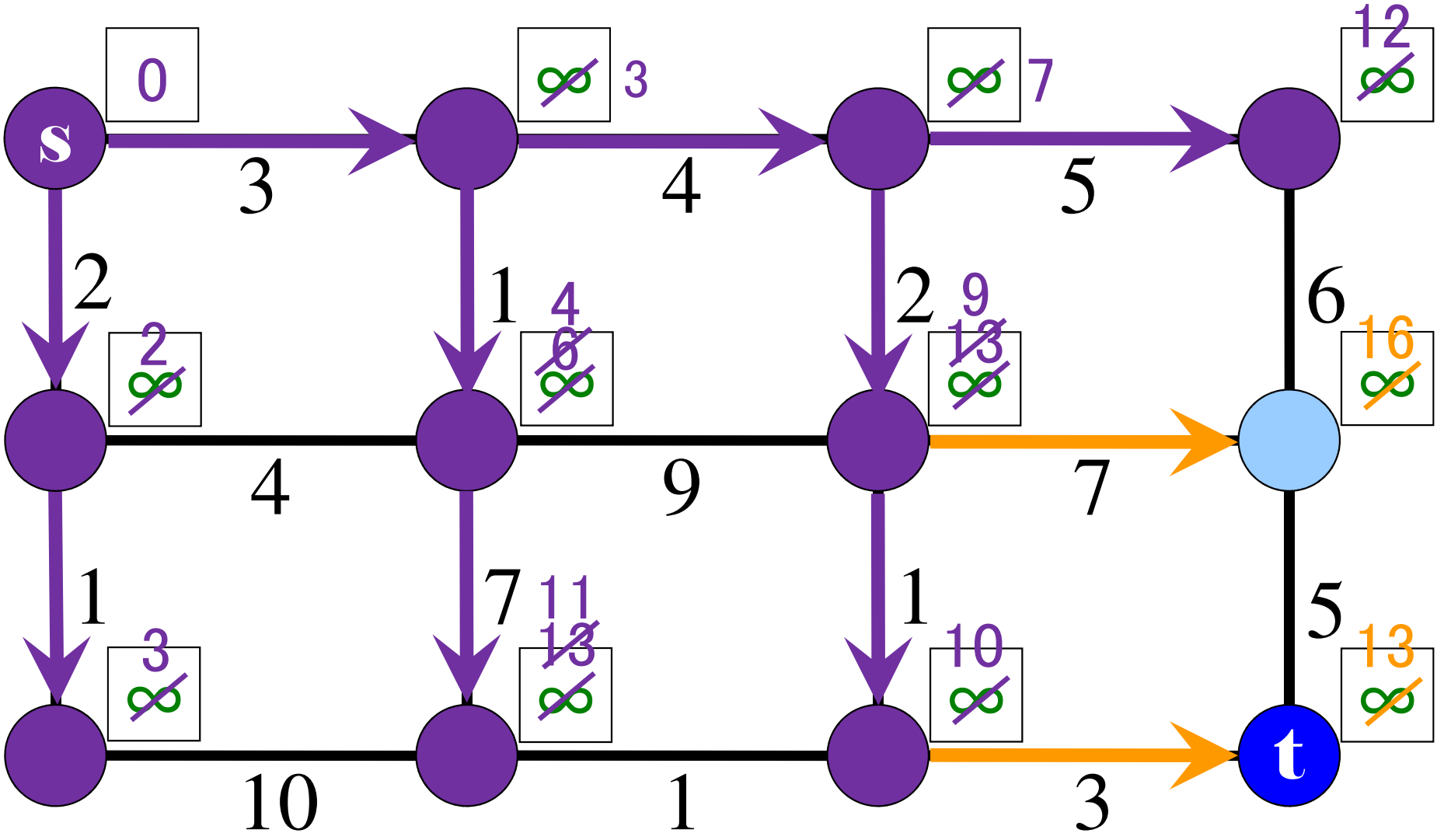
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

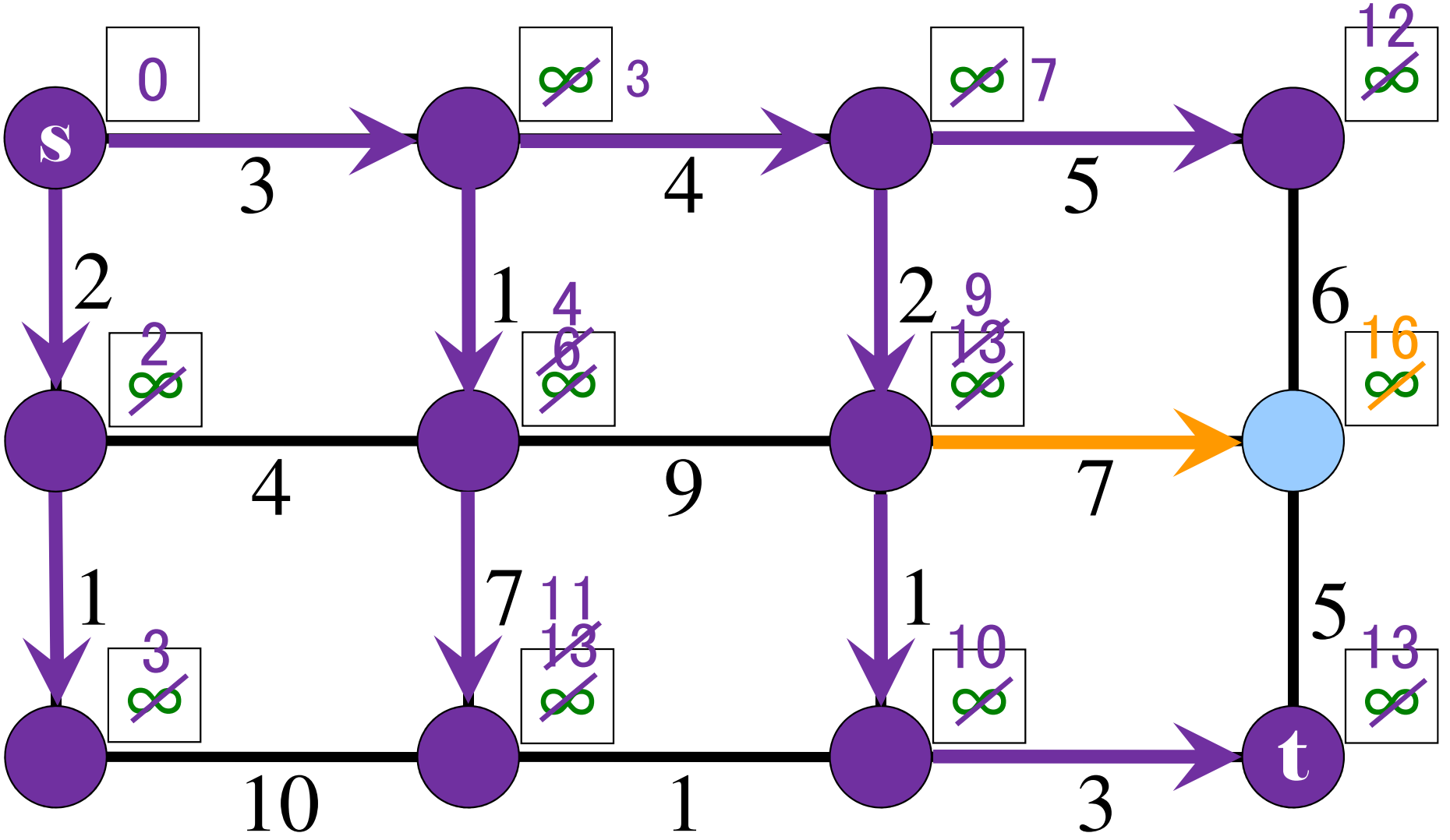


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

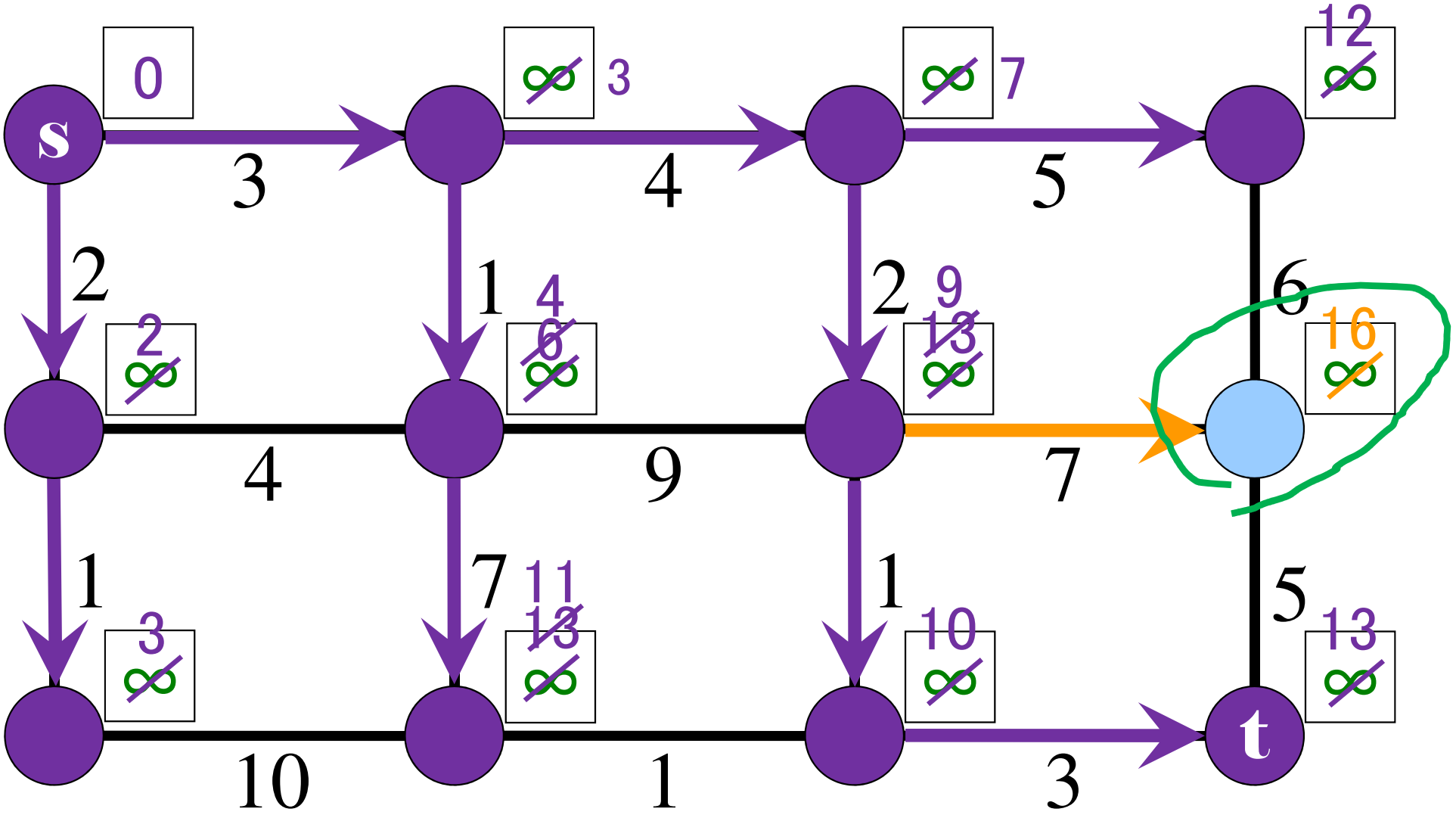
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



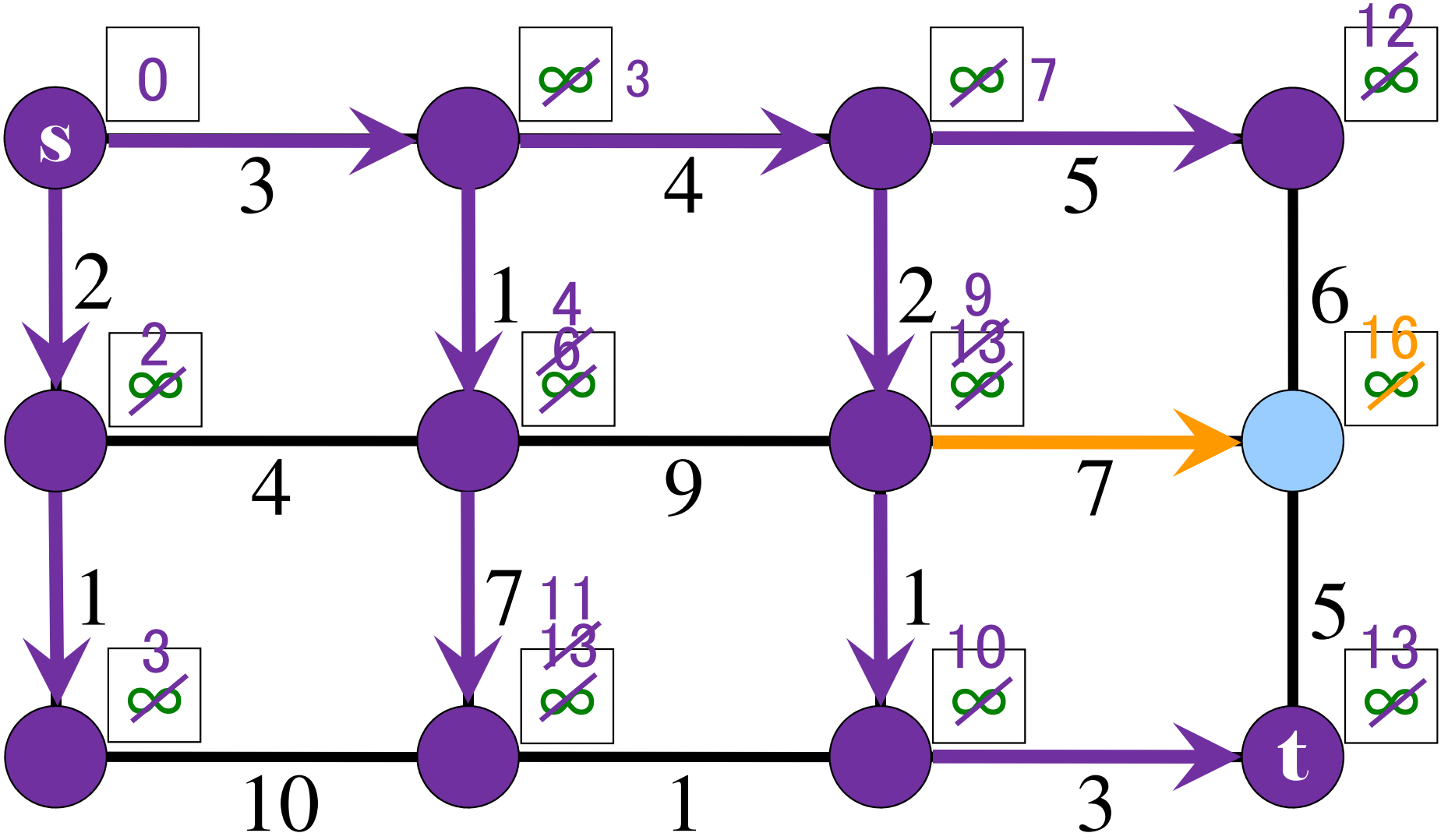
Dijkstra法（更新法）

step1-1:未確定点中，ラベル値が最小の点を見つける



Dijkstra法（更新法）

step1-2: その点から出る全枝について「ラベル＋枝コスト」を計算し，枝先点のラベル値と比較し，
もし，小さい（＜）→枝先点のラベル更新（暫定最短路）し，
そうでないなら何もしない

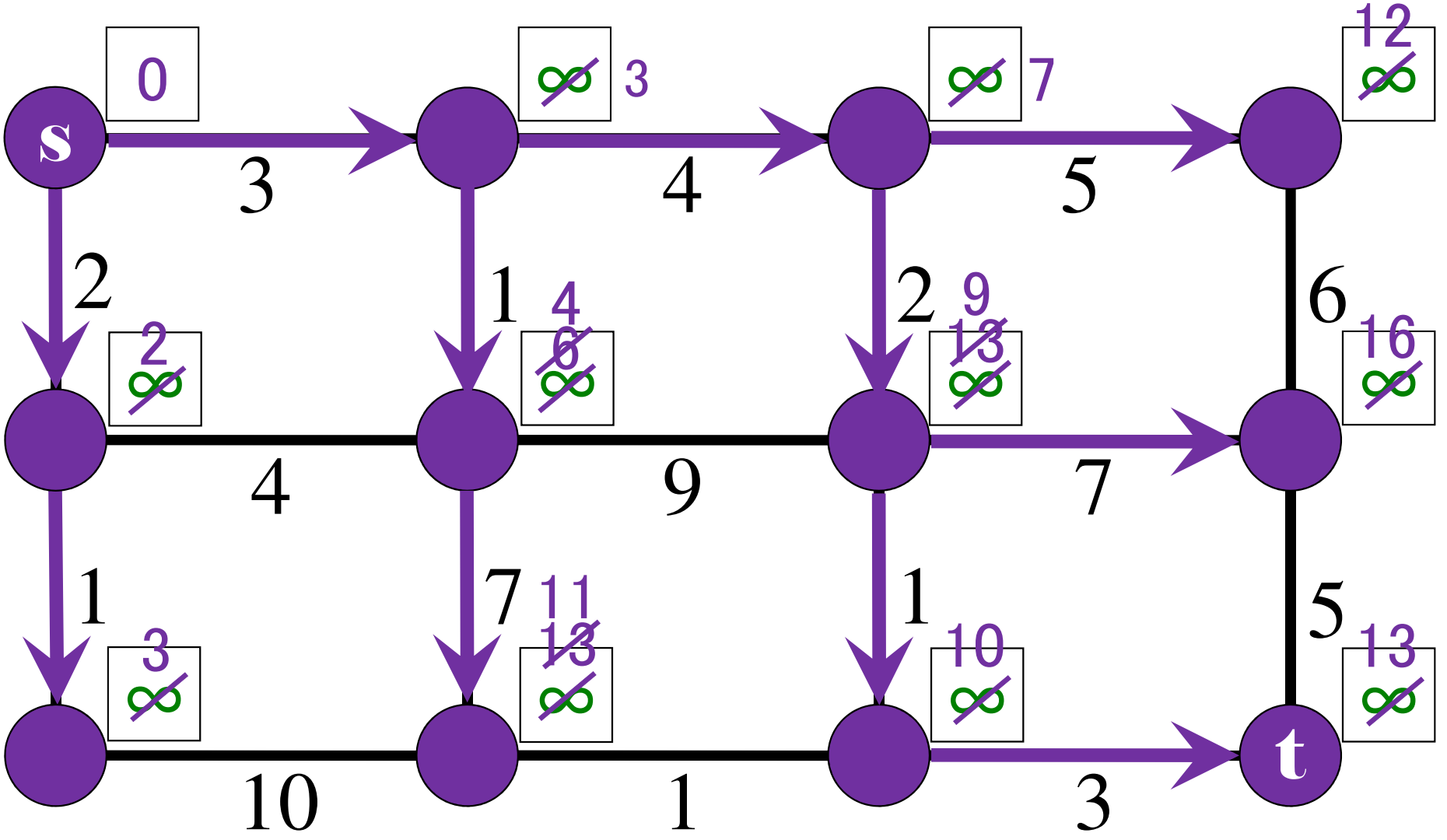


Dijkstra法（更新法）

step1-3:その点から出る全枝の作業が終了したら，その点を未確定点集合から除去

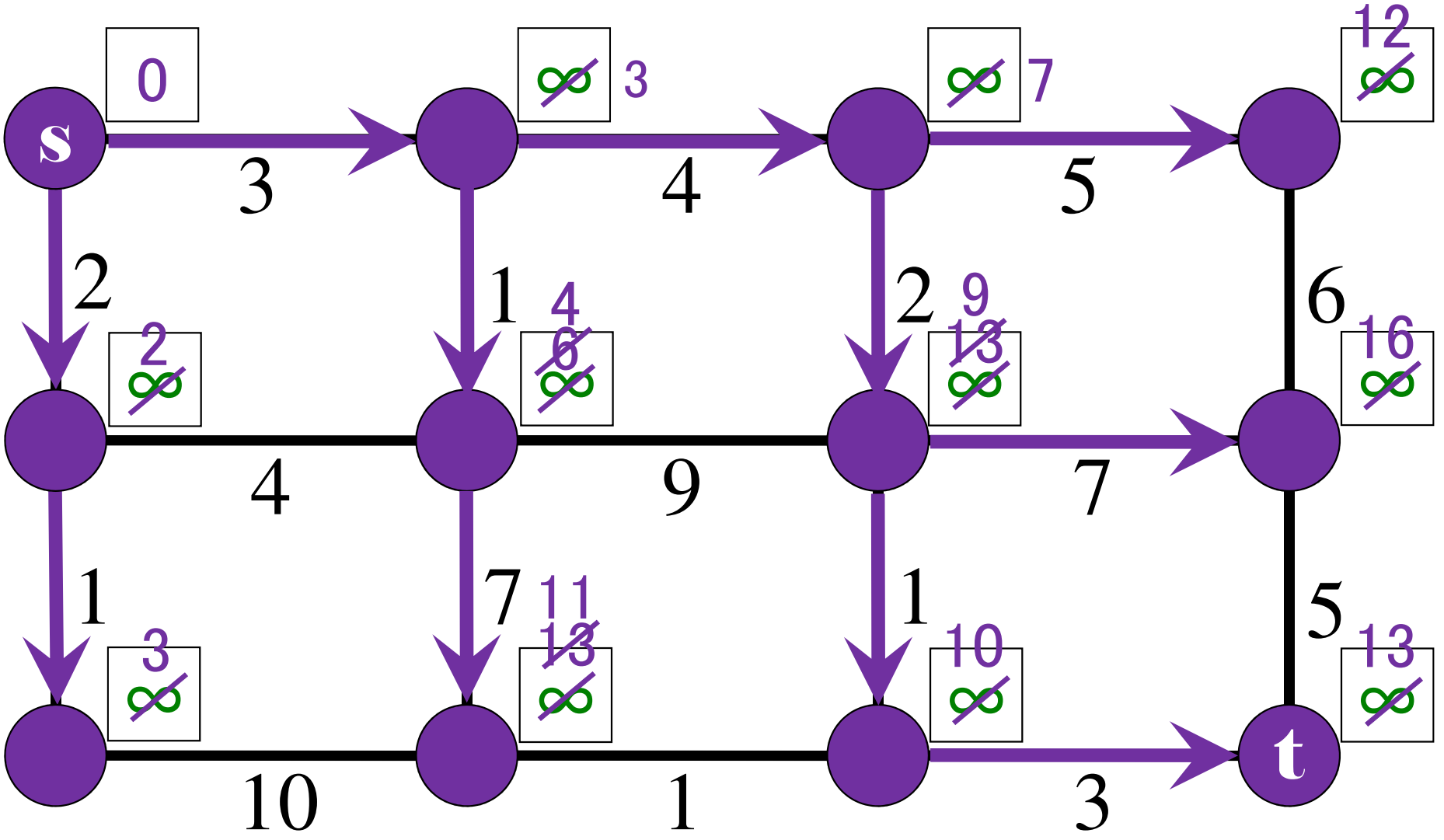
（確定点のラベル値＝その点までの最短距離）

以降**step1-1**～**step1-3**を終了条件を満たすまで繰り返す



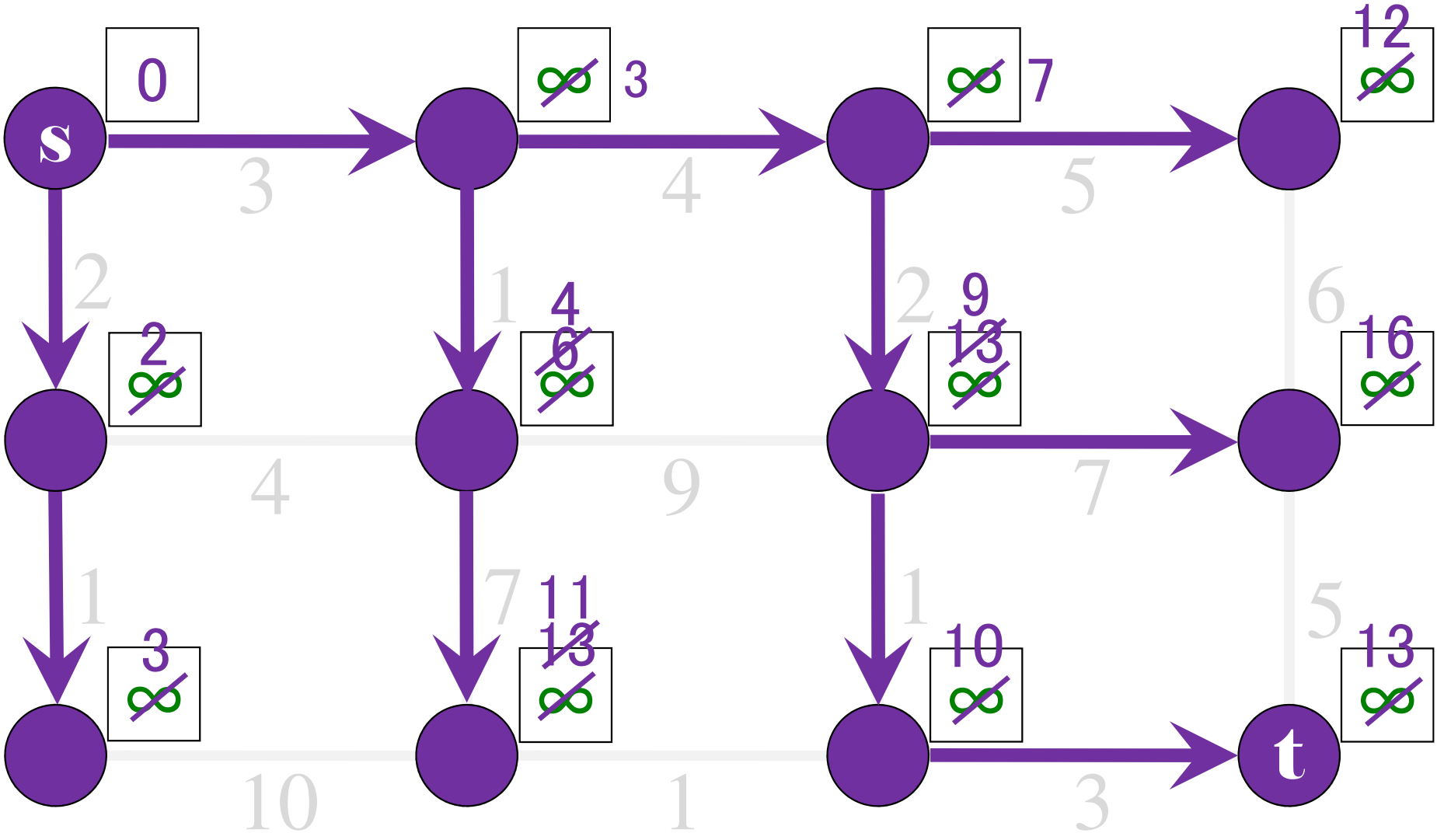
Dijkstra法（終了判定）

step2:未確定点集合が空になったら終了



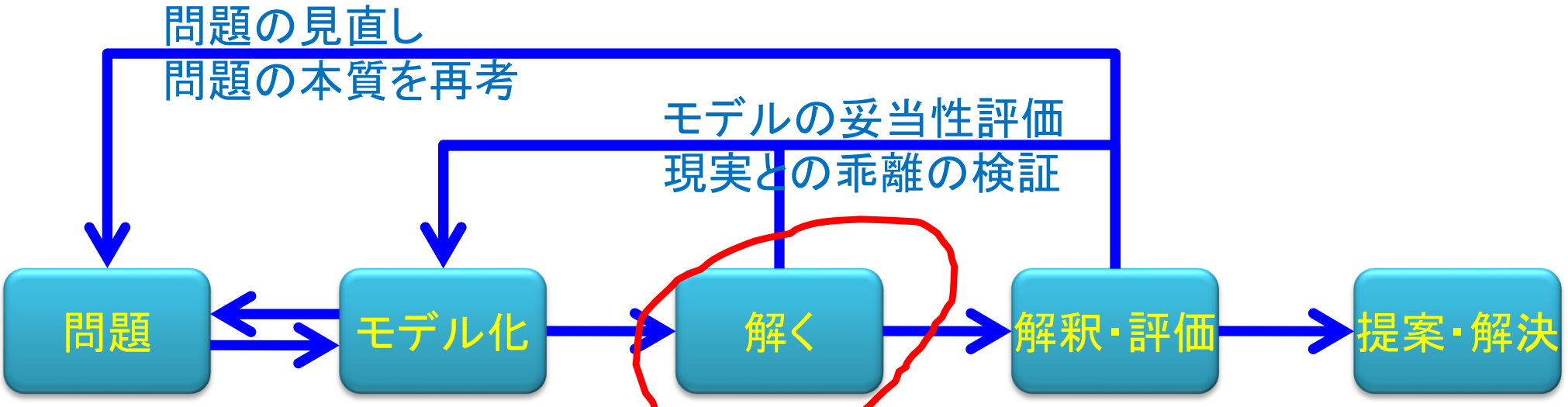
Dijkstra法終了後

【注】「スタートから全頂点への最短路が求まっている」ことに注意



問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

① 目的：最短経路を求める
現状：データが与えられている
最短経路が求まっていない

② グラフによるモデル
現実問題の抽象化

③ 全列挙 ✗

④ 格子グラフに制限
進行方向を制限

⑤ 全列挙

⑥ 計算量の見積り ✗

⑦ Dijkstra法

評価：Dijkstra法って速いのか？

- 点の数を n とすると，大雑把な見積もりで，

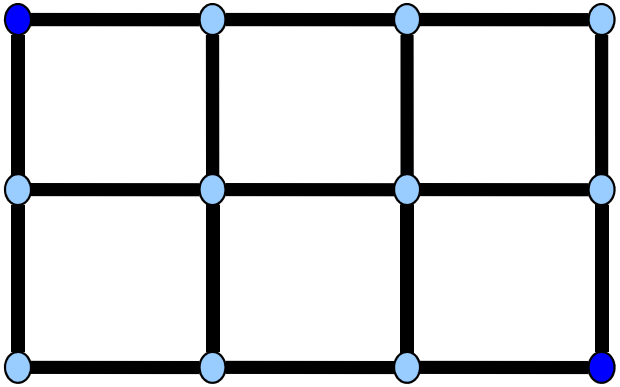
$$O(n^2)$$

多項式オーダー

$O(m+n\log n)$

- 点の数 n を右向枝数R，下向枝数Dで表すと

$$n=(R+1)\times(D+1)$$



$$n=(3+1)\times(2+1)=12$$

$$n^2=12^2=144$$

コンピュータに計算させてみよう！

簡単のため n^2 の5倍の浮動小数点演算回数で計算できると仮定.

評価：Dijkstra法って速いのか？

			415.53 PFLOPS	51.2GFLOPS
R（横）	D（縦）	全経路	富岳&しらみつぶし	Core i7 & Dijkstra
3	2	10	0.000000000秒	0.000000014秒
6	4	210	0.000000000秒	0.000000120秒
10	5	3,003	0.000000000秒	0.000000425秒
20	10	30,045,015	0.000000002秒	0.000005211秒
25	25	1.3×10^{14}	0.015210768秒	0.000044627秒
30	30	1.2×10^{17}	17.076684942秒	0.000090188秒
40	40	1.1×10^{23}	239.56日	0.000275953秒
50	50	1.0×10^{29}	769,918.79年	0.000660664秒
100	100	9.1×10^{58}	1.00E+26宙齢	0.010162149秒
500	500	2.7×10^{299}	1.49E+267宙齢	6.152490430秒

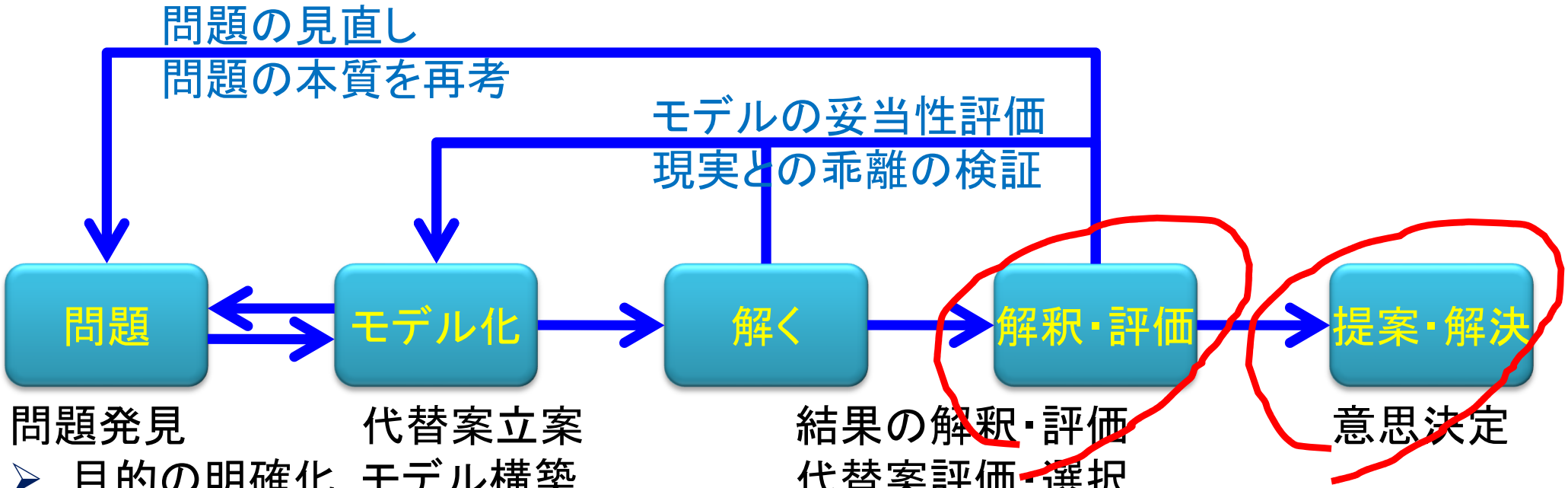
世界最速SuperComp
＋力技（しょぼい方法）

<<<

そこらのPC
＋人間の知恵

問題解決とは？

- 問題発見・問題解決から意思決定まで



問題発見
- 目的の明確化
- 現状の把握

代替案立案
モデル構築

結果の解釈・評価
代替案評価・選択

意思決定

①
目的：最短経路を求める
現状：データが与えられている
　　最短経路が求まっていない

②
グラフによるモデル
現実問題の抽象化

③
全列挙 ×

④
~~格子グラフに制限~~
~~進行方向を制限~~

⑤
全列挙

⑥
計算量の見積り ×

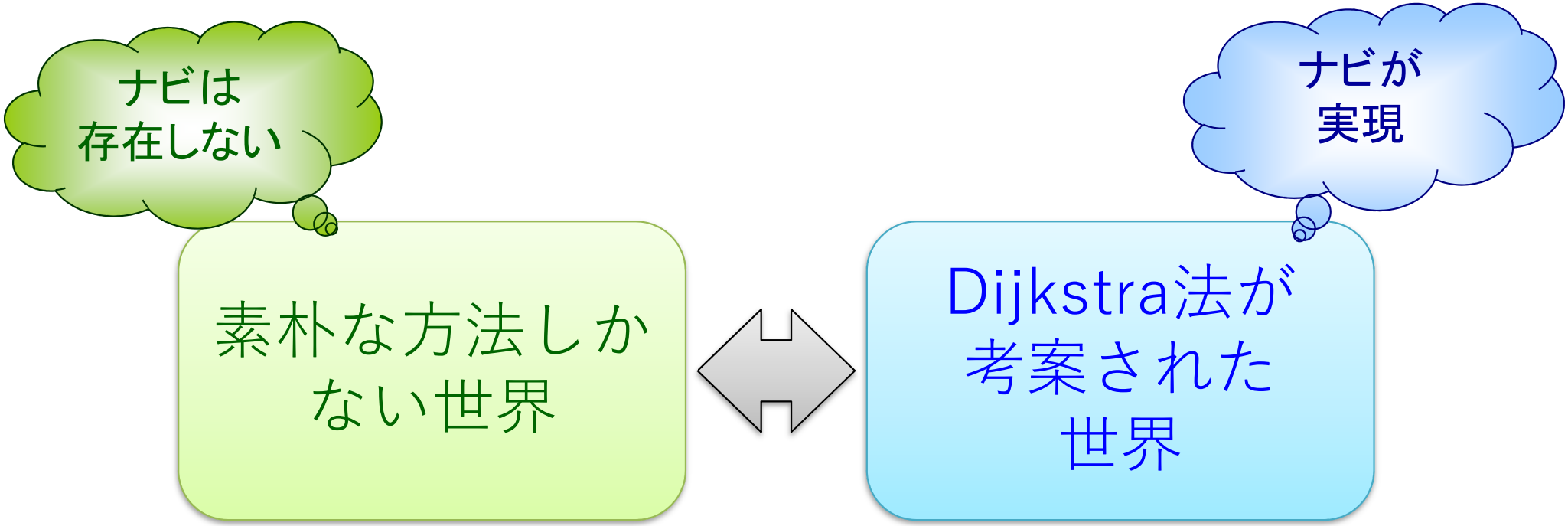
⑦
Dijkstra法

⑧
計算量の見積り

⑨

⑩
提案・解決

意思決定支援･ビジネスサポート



人類の創造的な仕事！

参考文献

コンピュータに仕事を奪われつつある人類…

[1] 新井紀子
「コンピュータが仕事を奪う」日経新聞社(2010)

[2] E. Brynjolfsson, A. McAfee, 村井章子訳
「機械との競争」日経BP社(2013)

もっと知りたい人へ

- 参考文献
  - グリッツマン，ブランデンベルク**「最短経路の本」** シュプリンガー（2008）
  - W.J.クック**「驚きの数学 巡回セールスマン問題」** 青土社（2013）
  - 山本，久保**「巡回セールスマン問題への招待」** 朝倉書店（1997）
  - 久保，松井**「組合せ最適化『短編集』」** 朝倉書店（1999）
  - 松井，根本，宇野**「入門オペレーションズ・リサーチ」**東海大出版（2008）

- 関連する授業
  - 「**ネットワークモデル分析**」（4セメ）
  - 「**最適化モデル分析**」（5セメ）　　etc…