

問題解決

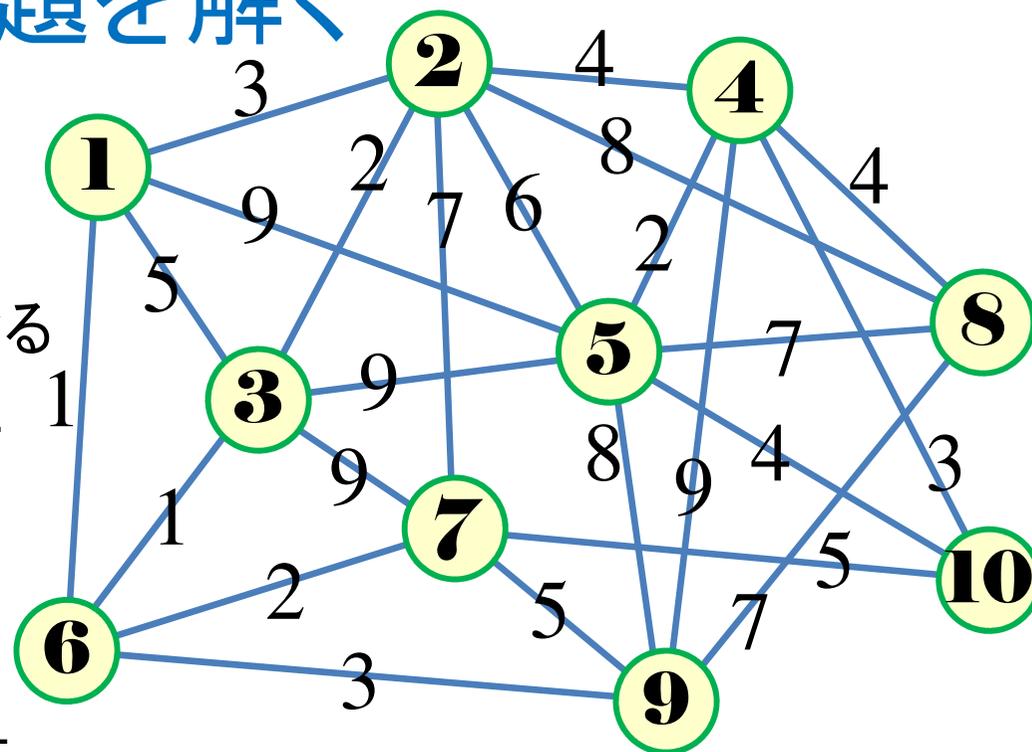
グラフ最適化と整数計画法 2. 最大重みマッチング問題

堀田 敬介

最大重みマッチング問題を解く

➤ 最大重みマッチング問題(例1)

- 10人の学生で複数のペアをつくる
- ペアになりたい親密度が各々異なる
- 親密度の総和が最大になるようにペアを組みたい



➤ 無向グラフ $G = (V, E)$

- 点集合 $V = \{1, 2, \dots, 10\}$ ※点=学生
- 枝集合 $E = \{(1, 2), (1, 3), (1, 5), (1, 6), (2, 3), (2, 4), (2, 5), (2, 7), (2, 8), (3, 5), (3, 6), (3, 7), (4, 5), (4, 8), (4, 9), (4, 10), (5, 8), (5, 9), (5, 10), (6, 7), (6, 9), (7, 9), (7, 10), (8, 9)\}$
- 各枝の重み $W_e = (3, 5, 9, 1, 2, 4, 6, 7, 8, 9, 1, 9, 2, 4, 9, 3, 7, 8, 4, 2, 3, 5, 5, 7)$
※枝=互いにペアになってもよい2人の学生を結ぶ線, 重み=親密度
- マッチング $M (\subseteq E)$: 端点を共有しない枝の集合

➤ 最大重みマッチング maximum weight matching

- 重みの総和が最大のマッチング

最大重みマッチング問題を解く

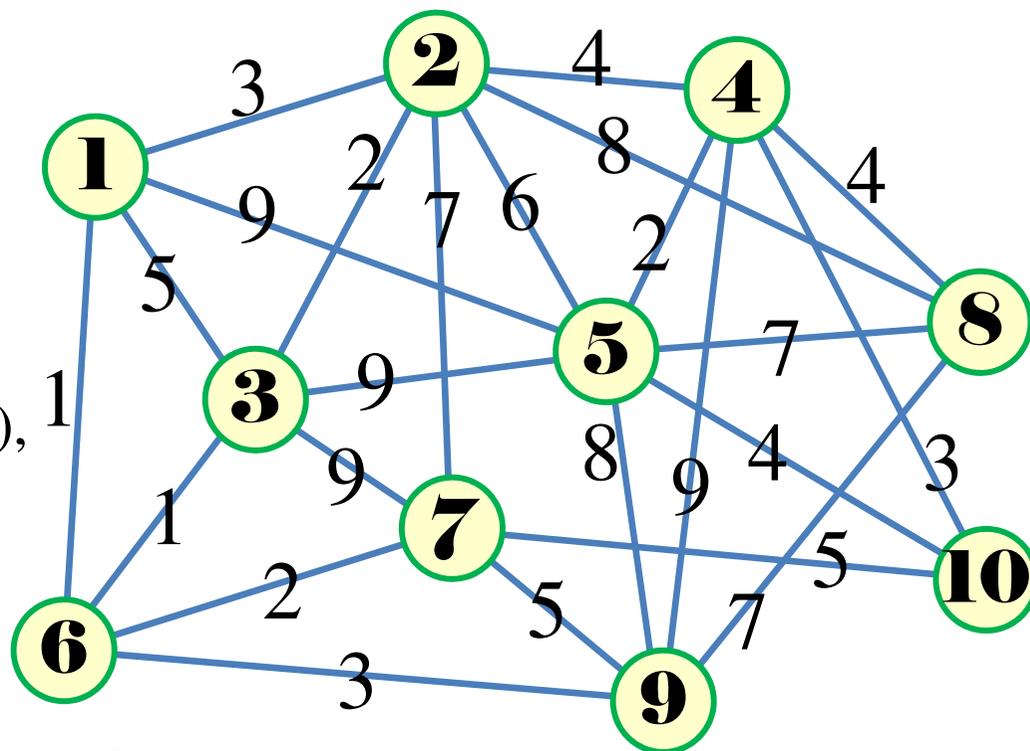
➤ 無向グラフ $G = (V, E)$

➤ 点集合 $V = \{1, 2, \dots, 10\}$

➤ 枝集合 $E = \{(1,2), (1,3), (1,5), (1,6), (2,3), (2,4), (2,5), (2,7), (2,8), (3,5), (3,6), (3,7), (4,5), (4,8), (4,9), (4,10), (5,8), (5,9), (5,10), (6,7), (6,9), (7,9), (7,10), (8,9)\}$

➤ $|V| = 10, |E| = 24$

➤ 各枝の重み $W_e = (3, 5, \dots, 7)$



➤ 接続行列 incident matrix $A = [a_{v,(i,j)}]$

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

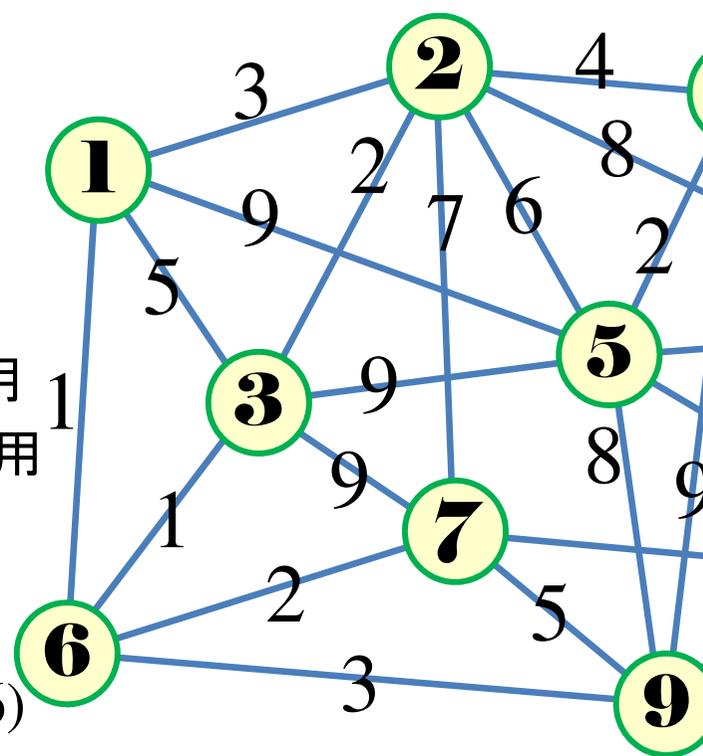
最大重みマッチング問題を解く

最適化問題の定式化 (変数設定)

- 0-1変数 $x_{(i,j)} = \begin{cases} 1 & \dots \text{枝}(i,j) \text{ をマッチングとして採用} \\ 0 & \dots \text{枝}(i,j) \text{ をマッチングとして不採用} \end{cases}$

制約条件

- 学生1とペアになって良い学生は4人 (学生2,3,5,6)
- 学生1とペアになれる学生は高々1人より $x_{(1,2)} + x_{(1,3)} + x_{(1,5)} + x_{(1,6)} \leq 1$
- 残りの学生 (2,3,...,10) についても同様



最適化問題の定式化 (ベタ表記)

$$\begin{aligned} \max. & w_{(1,2)}x_{(1,2)} + w_{(1,3)}x_{(1,3)} + \dots + w_{(8,9)}x_{(8,9)} \\ \text{s. t.} & x_{(1,2)} + x_{(1,3)} + x_{(1,5)} + x_{(1,6)} \leq 1 \\ & x_{(1,2)} + x_{(2,3)} + x_{(2,4)} + x_{(2,5)} + x_{(2,7)} + x_{(2,8)} \leq 1 \\ & x_{(1,3)} + x_{(2,3)} + x_{(3,5)} + x_{(3,6)} + x_{(3,7)} \leq 1 \\ & \dots \\ & x_{(4,10)} + x_{(5,10)} + x_{(7,10)} \leq 1 \\ & x_{(1,2)}, x_{(1,3)}, \dots, x_{(8,9)} \in \{0,1\} \end{aligned}$$

重みベクトル $W_e = (w_{(1,2)}, \dots, w_{(8,9)})$

接続行列 $A = [a_{v,(i,j)}]$

$$[a_{v,(i,j)}] = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & \dots \\ 1 & 0 & 0 & 0 & 1 & \dots \\ 0 & 1 & 0 & 0 & 1 & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & 1 & 0 & 0 & \dots \\ 0 & 0 & 0 & 1 & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots \end{pmatrix}$$

最大重みマッチング問題を解く

($|V|=10, |E|=24$)

最適化問題の定式化 (Σ 表記 \Leftrightarrow ベタ表記)

$\max. \sum_{(i,j) \in E} w_{(i,j)} x_{(i,j)}$ <p>s. t.</p> <div style="border: 1px solid black; background-color: #e0ffe0; padding: 5px; margin: 10px 0;"> $\sum_{(i,j) \in E} a_{v,(i,j)} x_{(i,j)} \leq 1 \quad (v \in V)$ </div> $x_{(i,j)} \in \{0,1\} \quad ((i,j) \in E)$	\longleftrightarrow	$\max. w_{(1,2)}x_{(1,2)} + w_{(1,3)}x_{(1,3)} + \dots + w_{(8,9)}x_{(8,9)}$ <p>s. t.</p> <div style="border: 1px solid black; background-color: #e0ffe0; padding: 5px; margin: 10px 0;"> $x_{(1,2)} + x_{(1,3)} + x_{(1,5)} + x_{(1,6)} \leq 1$ $x_{(1,2)} + x_{(2,3)} + x_{(2,4)} + x_{(2,5)} + x_{(2,7)} + x_{(2,8)} \leq 1$ $x_{(1,3)} + x_{(2,3)} + x_{(3,5)} + x_{(3,6)} + x_{(3,7)} \leq 1$ <p style="text-align: center;">...</p> $x_{(4,10)} + x_{(5,10)} + x_{(7,10)} \leq 1$ </div> $x_{(1,2)}, x_{(1,3)}, \dots, x_{(8,9)} \in \{0,1\}$
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

最適化問題の定式化 (行列表記)

$\max.$ <p>s. t.</p>	$(3 \ 5 \ 9 \ 1 \ 2 \ 4 \ 6 \ 7 \ 8 \ 9 \ \dots \ 7)$ <div style="border: 1px solid black; background-color: #e0ffe0; padding: 10px; display: inline-block;"> <table border="0" style="text-align: center; border-collapse: collapse;"> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>...</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>...</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>...</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>...</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>...</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>...</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>...</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>...</td><td>1</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>...</td><td>1</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>...</td><td>0</td></tr> </table> </div>	1	1	1	1	0	0	0	0	0	0	...	0	1	0	0	0	1	1	1	1	1	0	...	0	0	1	0	0	1	0	0	0	0	1	...	0	0	0	0	0	0	1	0	0	0	0	...	0	0	0	1	0	0	0	1	0	0	1	...	0	0	0	0	1	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	1	0	0	...	0	0	0	0	0	0	0	0	0	1	0	...	1	0	0	0	0	0	0	0	0	0	0	...	1	0	0	0	0	0	0	0	0	0	0	...	0	\leq	$\begin{pmatrix} x_{(1,2)} \\ x_{(1,3)} \\ x_{(1,5)} \\ x_{(1,6)} \\ x_{(2,3)} \\ x_{(2,4)} \\ \dots \\ x_{(8,9)} \end{pmatrix}$
1	1	1	1	0	0	0	0	0	0	...	0																																																																																																																
1	0	0	0	1	1	1	1	1	0	...	0																																																																																																																
0	1	0	0	1	0	0	0	0	1	...	0																																																																																																																
0	0	0	0	0	1	0	0	0	0	...	0																																																																																																																
0	0	1	0	0	0	1	0	0	1	...	0																																																																																																																
0	0	0	1	0	0	0	0	0	0	...	0																																																																																																																
0	0	0	0	0	0	0	1	0	0	...	0																																																																																																																
0	0	0	0	0	0	0	0	1	0	...	1																																																																																																																
0	0	0	0	0	0	0	0	0	0	...	1																																																																																																																
0	0	0	0	0	0	0	0	0	0	...	0																																																																																																																
			$\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ \dots \\ 1 \\ 1 \\ 1 \end{pmatrix}$																																																																																																																								
			$\begin{pmatrix} x_{(1,2)} \\ x_{(1,3)} \\ x_{(1,5)} \\ x_{(1,6)} \\ x_{(2,3)} \\ x_{(2,4)} \\ \dots \\ x_{(8,9)} \end{pmatrix} \in \{0,1\}^{ E }$																																																																																																																								

最大重みマッチング問題をCPLEXで解く

➤ 新規プロジェクトの作成

- ① [ファイル(F)]－[新規(N)]－[OPLプロジェクト]を選択
- ② [プロジェクト名]を記入(例: **MaxWeightMatching**)し, 3カ所にチェックする
 - デフォルトの実行構成の追加
 - モデルの作成
 - データの作成
- ③ [終了]をクリック

プロジェクト名は自由だが, **半角英数**で何の問題を解こうとしているのかが分かる名前が良い

➤ プロジェクト内のいくつかの名前を変更

- ✓ [構成1] → [**config1**] ※日本語を英語に変更しないと実行時エラーになる
- ✓ モデルファイル [MaxWeightMatching.mod] → [**mwm.mod**]
- ✓ データファイル [MaxWeightMatching.dat] → [**mwmex1.dat**]

➤ モデルファイル・データファイルを記述し保存(次ページ参照)

➤ [config1]にモデルファイルとデータファイルをセットし, 解く

最大重みマッチング問題をCPLEXで解く

➤ モデルファイル(mwm.mod)の中身の記述

```
int v_max = ...; // 点数|V|
int e_max = ...; // 枝数|E|

range V = 1..v_max;
range E = 1..e_max;

int a[V,E] = ...; // 接続行列
int w[E] = ...; // 重みベクトル

dvar int x[E] in 0..1;

maximize
    sum(e in E) w[e]*x[e];
subject to {
    forall (v in V) {
        sum(e in E) a[v,e]*x[e] <= 1;
    };
};
```

最大重みマッチング問題をCPLEXで解く

▶ データファイル(mwmex1.dat)の中身の記述

```
v_max = 10; // 点数|V|
e_max = 24; // 枝数|E|

w = [ 3 5 9 1 2 4 6 7 8 9 1 9 2 4 9 3 7 8 4 2 3 5 5 7 ];
a = [
[ 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ]
[ 1 0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 ]
[ 0 1 0 0 1 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 ]
[ 0 0 0 0 0 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 ]
[ 0 0 1 0 0 0 1 0 0 1 0 0 1 0 0 0 1 1 1 0 0 0 ]
[ 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1 0 0 ]
[ 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 1 0 1 1 ]
[ 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 ]
[ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 1 ]
[ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 ]
]; // 接続行列(サイズ:|V|×|E|)
```

最大重みマッチング問題をCPLEXで解く

➤ 結果([解]タブ)

最大重みは35 (= 9+8+9+9)

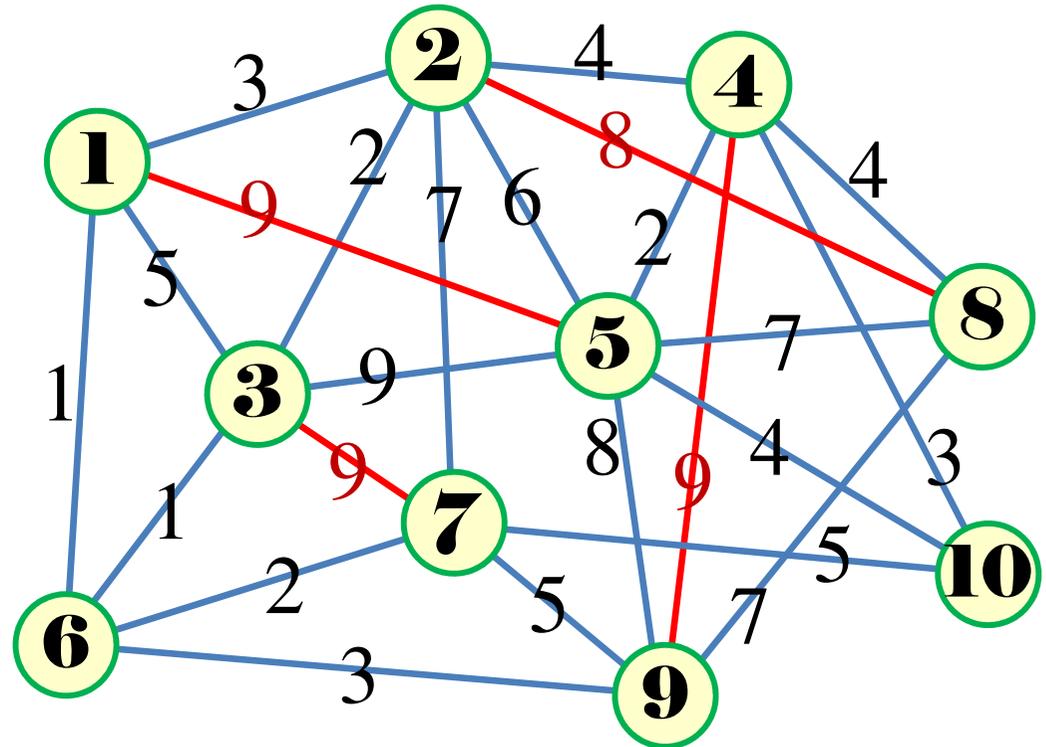
```
// solution (optimal) with objective 35 ← 最適値 = 35
// Quality Incumbent solution:
// MILP objective 3.5000000000e+01
// MILP solution norm |x| (Total, Max) 4.00000e+00 1.00000e+00
// MILP solution error (Ax=b) (Total, Max) 0.00000e+00 0.00000e+00
// MILP x bound error (Total, Max) 0.00000e+00 0.00000e+00
// MILP x integrality error (Total, Max) 0.00000e+00 0.00000e+00
// MILP slack bound error (Total, Max) 0.00000e+00 0.00000e+00
//
x = [0 0 1 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0];
```

最適解

使用枝数は4本

➤ 枝集合 $E = \{$

- (1,2), (1,3), (1,5), (1,6), (2,3),
- (2,4), (2,5), (2,7), (2,8), (3,5),
- (3,6), (3,7), (4,5), (4,8), (4,9),
- (4,10), (5,8), (5,9), (5,10), (6,7),
- (6,9), (7,9), (7,10), (8,9) $\}$



最大重みマッチング問題をgurobiで解く(1)

- cplexの「モデルファイル (*.mod)」と「データファイル (*.dat)」を使って「lpファイル (*.lp)」を生成する
 - 例) モデルファイル [mwm.mod], データファイル [mwmex1.dat]
→ 生成する lpファイル [mwmex1.lp]
 - [Win]+[R] キー で [ファイル名を指定して実行] d-boxを起動する
 - 枠内で `cmd [Enter]`
 - コマンドプロンプト command prompt のウィンドウ (黒い画面) が起動する
- 以降, コマンドプロンプト内でコマンド (命令文) を打って順次命令を実行する
 - (1) モデルファイルとデータファイルがあるフォルダに移動する
`cd [フォルダへのパス] [Enter]`
 - (2) 以下のコマンドを実行する
`oplrn -e mwmex1.lp mwm.mod mwmex1.dat [Enter]`
- この結果, モデルファイル [mwm.mod] とデータファイル [mwmex1.dat] と同じフォルダ内に, lpファイル [mwmex1.lp] が出来る (※確認すること)

最大重みマッチング問題をgurobiで解く(1)

➤ gurobi を起動して問題を解き，最適解を得る

➤ コマンドプロンプトで，以下の命令文を打って gurobi を起動する

```
gurobi [Enter]
```

➤ 起動した gurobi 内で，順次，以下の命令文を打って問題を解いていく

(1) 問題を記述してある lpファイル(mwmex1.lp)を読み込み，model へセット

```
model = read("mwmex1.lp") [Enter]
```

(2) 解く(最適化計算を開始する) ※読込に失敗しているとエラーとなる

```
model.optimize() [Enter]
```

(3) 最適解を表示する ※最適解が求まっていない場合はエラーとなる

```
model.printAttr('X') [Enter]
```

(4) 最適値(目的関数値)を表示する ※同上

```
model.ObjVal [Enter]
```

(5) 最適解をファイル(*.sol)に出力する ※ファイル名は好きに

```
model.write("mwmex1.sol") [Enter]
```

最大重みマッチング問題をgurobiで解く(1)

➤ gurobi のその他, 知っておくと便利な命令文

➤ いずれも gurobi を起動して, gurobi内で行う

(a) ヘルプを表示する

```
help() [Enter]
```

(b) 全ての最適解(値が0の解)を表示する

```
for v in model.getVar(): [Enter]  
    print( v.VarName, ":", v.X) [Enter]
```

- 最適解を表示する命令文「`m.printAttr('X')`」は, 値が0となる解は表示しない
- 2行目の print 文は, 必ず字下げ(インデント)して書くこと(Pythonの文法)
- 字下げは[Tab]キーを使うと良い(※面倒でなければ, 半角スペースでも可)
- `model.getVar()` でモデルから変数Var(variableの頭3文字)を get する命令
- getした各変数をインデックス v として, for文で繰り返す(2行目を繰り返す)
- `v.VarName` は, ゲットした各変数の「名称」を意味する予約語
- `v.X` は, ゲットした各変数の「値」を意味する予約語
- 以上より, 各変数を1つずつ「名称: 値」の形で画面に表示(print)する

最大重みマッチング問題をgurobiで解く(2)

➤ 問題(ex1)をpython & gurobi で記述(mwm.py)

```
# coding: Shift_JIS
from gurobipy import *
```

①

```
# ##### 例題設定 #####
```

```
def make_data_ex1():
    V = [1,2,3,4,5,6,7,8,9,10]
    E,w = multidict({(1,2):3,(1,3):5,(1,5):9,
(1,6):1,(2,3):2,(2,4):4,(2,5):6,(2,7):7,(2,8):8,
(3,5):9,(3,6):1,(3,7):9,(4,5):2,(4,8):4,(4,9):9,
(4,10):3,(5,8):7,(5,9):8,(5,10):4,(6,7):2,
(6,9):3,(7,9):5,(7,10):5,(8,9):7})
    return V,E,w
```

```
# ##### 定式化 #####
```

```
def mwm(V,E,w):
```

```
    mod = Model("maximum weight matching problem")
```

```
    # 変数設定
```

```
    x = {}
```

```
    for (i,j) in E:
```

```
        x[i,j] = mod.addVar(vtype="B", name="x(%s,%s)" % (i,j))
```

```
    mod.update()
```

```
    # 制約条件の設定
```

```
    for v in V:
```

```
        mod.addConstr(quicksum(x[i,j] for (i,j) in E if i==v or j==v) <= 1)
```

```
    # 目的関数の設定
```

```
    mod.setObjective(quicksum(w[i,j] * x[i,j] for (i,j) in E), GRB.MAXIMIZE)
```

```
    mod.update()
```

```
    mod.__data = x
```

```
    return mod
```

②

```
# ##### 実行 #####
```

```
if __name__ == "__main__":
```

```
    V,E = make_data_ex1() # データの生成
```

```
    mod = mwm(V,E,w) # モデルの生成
```

```
    mod.write("mwmex1.lp") # lpファイルを出力
```

```
    mod.optimize() # 最適化実行
```

```
    print("¥n optimal value = ", mod.ObjVal) # 最適値の表示
```

```
    mod.printAttr('X') # 最適解の表示
```

```
    mod.write("mwmex1.sol") # 最適解をsolファイルに出力
```

③

1つのファイル「mwm.py」に
①②③の順に記述して保存

最大重みマッチング問題をgurobiで解く(2)

➤ Pythonファイル(mwm.py)をgurobi上で実行し、解く

➤ [Win]+[R] キー で [ファイル名を指定して実行] d-boxを起動する

➤ 枠内で `cmd [Enter]`

➤ コマンドプロンプト command prompt のウィンドウ(黒い画面)が起動する

➤ コマンドプロンプト内でコマンド(命令文)を打って順次命令を実行する

(1) 実行ファイルがあるフォルダに移動する

```
cd [フォルダへのパス] [Enter]
```

(2) 以下の命令文を打って gurobi を起動する

```
gurobi [Enter]
```

➤ 起動した gurobi 内で、以下の命令文を打って問題を解く

```
gurobi> exec( open("mwm.py").read() ) [Enter]
```

※python3系の場合

※python2系の場合の命令文は以下

```
gurobi> execfile("mwm.py") [Enter]
```

最大重みマッチング問題をgurobiで解く(2)

▶ 実行結果

```
gurobi> exec(open("mwm.py").read())
Gurobi Optimizer version 9.5.2 build v9.5.2rc0 (win64)
Thread count: 10 physical cores, 20 logical processors, using up to 20 threads
Optimize a model with 10 rows, 24 columns and 48 nonzeros
Model fingerprint: 0xfb73ca57
Variable types: 0 continuous, 24 integer (24 binary)
Coefficient statistics:
  Matrix range    [1e+00, 1e+00]
  Objective range [1e+00, 9e+00]
  Bounds range    [1e+00, 1e+00]
  RHS range       [1e+00, 1e+00]
Found heuristic solution: objective 18.0000000
Presolve time: 0.00s
Presolved: 10 rows, 24 columns, 48 nonzeros
Variable types: 0 continuous, 24 integer (24 binary)
Found heuristic solution: objective 25.0000000

Root relaxation: objective 3.500000e+01, 5 iterations, 0.00 seconds (0.00 work units)

   Nodes          |   Current Node   |   Objective Bounds   |   Work
  Expl Unexpl |   Obj  Depth IntInf | Incumbent  BestBd  Gap | It/Node Time
*    0     0 |             0 | 35.0000000  35.00000  0.00% | -     0s

Explored 1 nodes (5 simplex iterations) in 0.02 seconds (0.00 work units)
Thread count was 20 (of 20 available processors)

Solution count 3: 35 25 18

Optimal solution found (tolerance 1.00e-04)
Best objective 3.5000000000000e+01, best bound 3.5000000000000e+01, gap 0.0000%

optimal value = 35.0

-----
Variable          X
-----
x(1,5)            1
x(2,8)            1
x(3,7)            1
x(4,9)            1
gurobi>
```

【演習】最大重みマッチング問題を解く

➤ ex2) 無向グラフ $G = (V, E)$

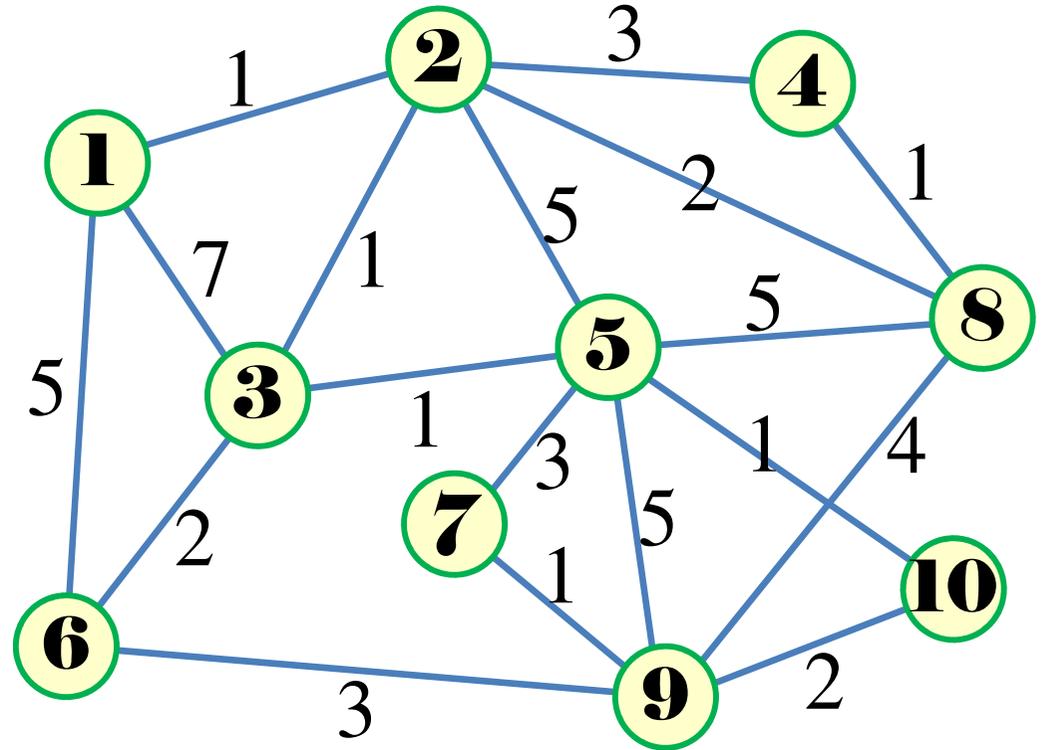
➤ 点集合 $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$,

➤ 枝集合 $E = \{(1, 2), (1, 3), (1, 6), (2, 3), (2, 4), (2, 5), (2, 8), (3, 5), (3, 6), (4, 8), (5, 7), (5, 8), (5, 9), (5, 10), (6, 9), (7, 9), (8, 9), (9, 10)\}$

➤ 枝重み $W_e = (1, 7, 5, 1, 3, \dots, 2)$

➤ 問

1. $|V| = ?$ $|E| = ?$
2. 接続行列 A をつくれ
3. 例1と同様に0-1変数 $x_{(i,j)}$ を設定し, 定式化せよ
4. 整数計画ソルバー(cplex)を用いて, 最大重みマッチングを求めよ
5. oplrun を使って, mod file / dat file から lp file を作れ
6. 整数計画ソルバー(gurobi)で5のlp file を解き, 最大重みマッチングを求めよ
7. 整数計画ソルバー(gurobi)とpython で解き, 最大重みマッチングを求めよ
8. 結果を networkx でグラフ描画せよ



【演習】最大重みマッチング問題を解く

➤ ex3) 無向グラフ $G = (V, E)$

➤ 点集合 $V = \{1, 2, \dots, 10\}$

➤ 枝集合 $E = \{(1, 3), (1, 5), (1, 6), (2, 4), (2, 5), (3, 5), (3, 7), (4, 5), (4, 8), (5, 8), (5, 9), (6, 7), (6, 9), (7, 9), (8, 9), (8, 10), (9, 10)\}$

➤ 枝重み $W_e = (3, 1, 4, 4, 8, \dots, 5)$

➤ 問

1. $|V| = ?$ $|E| = ?$
2. 接続行列 A をつくれ
3. 例1と同様に0-1変数 $x_{(i,j)}$ を設定し、定式化せよ
4. 整数計画ソルバー(cplex)を用いて、最大重みマッチングを求めよ
5. oplrun を使って、mod file / dat file から lp file を作れ
6. 整数計画ソルバー(gurobi)で5のlp file を解き、最大重みマッチングを求めよ
7. 整数計画ソルバー(gurobi)とpython で解き、最大重みマッチングを求めよ
8. 結果を networkx でグラフ描画せよ

