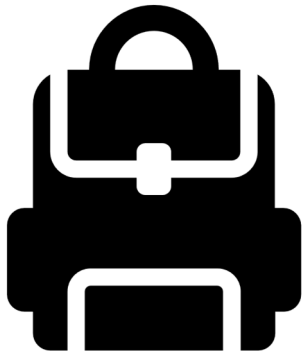
2022/4/25 Mon.

問題解決のための最適化

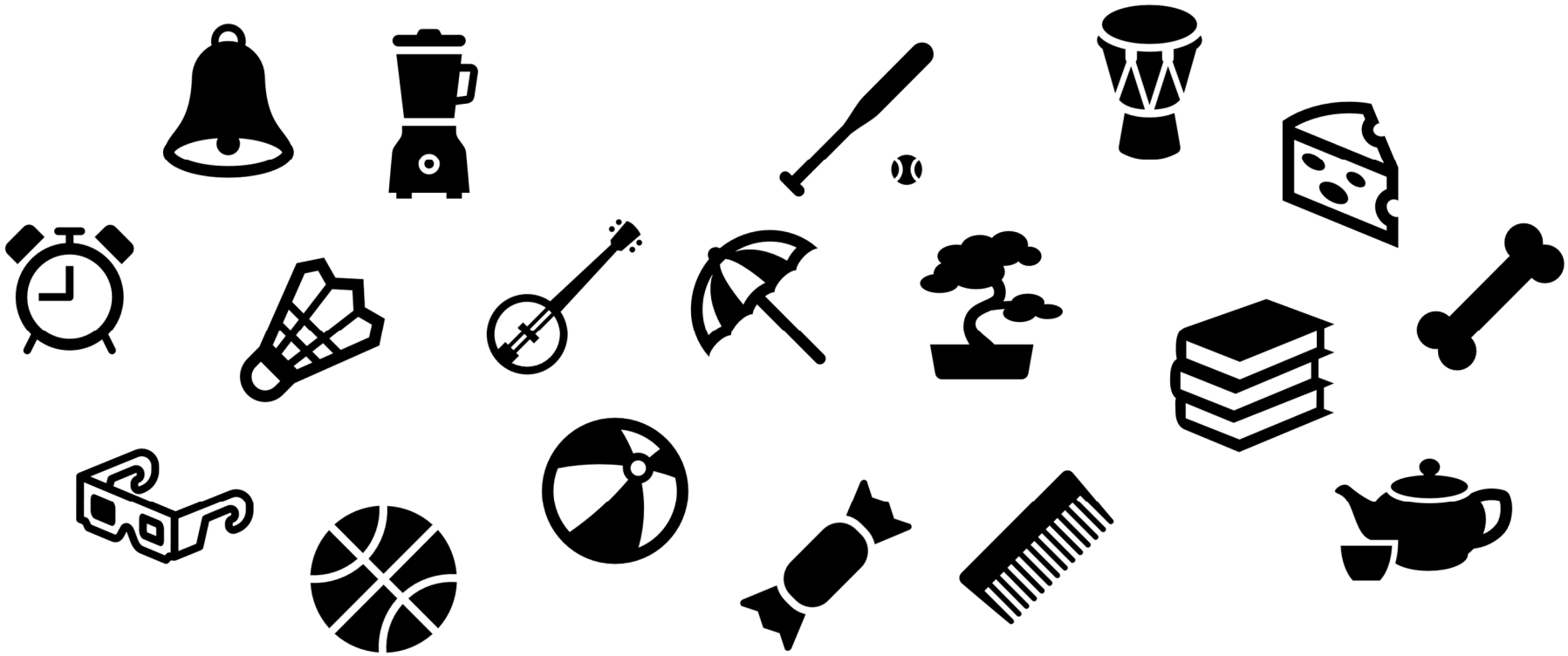
組合せ最適化と整数計画法
1. ナップサック問題

堀田 敬介

ナップサック問題の最適化



- ナップサック問題 knapsack problem
  - 容量 C のナップサックがある
  - n 個のアイテムがあり，それぞれコストと効用がある
  - アイテム i のコストを c_i ，効用を u_i とする（所与）
  - コスト和が容量を越えない範囲でアイテムを選び，ナップサックに詰める
  - 目的：効用の和が最大になるように，アイテムを選ぶ



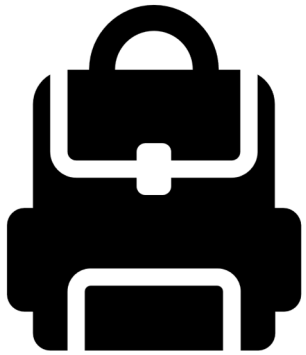
ナップサック問題の最適化

- 最適化問題の定式化（変数設定・係数表記）
  - 0-1変数 $x_i = \begin{cases} 1 & \ldots \text{アイテム } i \text{ を詰める} \\ 0 & \ldots \text{アイテム } i \text{ を詰めない} \end{cases}$
  - アイテム i のコスト c_i ，コストベクトル $\boldsymbol{c} = (c_1, \ldots, c_n)$
  - アイテム i の効用 u_i ，効用ベクトル $\boldsymbol{u} = (u_1, \ldots, u_n)$

- 最適化問題の定式化（Σ表記・ベタ表記）

$$\left| \begin{array}{l} \max. \sum_{i=1}^{n} u_i x_i \\ \text{s. t. } \sum_{i=1}^{n} w_i x_i \leq C \\ \qquad x_i \in \{0,1\} \ (i = 1, \ldots, n) \end{array} \right| \quad \left| \begin{array}{l} \max. \ u_1 x_1 + u_2 x_2 + \ldots + u_n x_n \\ \text{s. t. } \ c_1 x_1 + c_2 x_2 + \ldots + c_n x_n \leqq C \\ \qquad x_1, x_2, \ldots, x_n \in \{0,1\} \end{array} \right|$$

ナップサック問題の最適化



- ナップサック問題 knapsack problem
  - 容量 C のナップサックがある
  - n 個のアイテムがあり，それぞれコストと効用がある
  - アイテム i のコストを c_i ，効用を u_i とする（所与）
  - コスト和が容量を越えない範囲でアイテムを選び，ナップサックに詰める
  - 目的：効用の和が最大になるように，アイテムを選ぶ

- ナップサック問題でモデル化出来る例（ex1）
  - 容量C(=70)リットルのナップサックがある
  - 登山用具がn(=15)個あり，体積と効用は下表の通り
  - 効用最大化するよう用具をナップサックに詰めたい，どれを持って行くか？

登山用具	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
体積(l)	15	2	4	7	8	9	12	6	4	5	3	2	7	9	8
効用	10	3	2	6	5	7	10	5	7	3	5	1	6	9	7

  - アイテム集合 $\{1, 2, \ldots, 15\}$　※n=15

ナップサック問題をCPLEXで解く

- 新規プロジェクトの作成
  - ① [ファイル(F)]ー[新規(N)]ー[OPLプロジェクト]を選択
  - ② [プロジェクト名] を記入（例：Knapsack）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 [Knapsack.mod] → [ks.mod]
  - ✓ データファイル [Knapsack.dat] → [ksex1.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

ナップサック問題をCPLEXで解く

- モデルファイル(ks.mod)の中身の記述

```
int i_max = ...;        // アイテムの最大数
int Capacity = …;       // ナップサック容量

range I = 1..i_max;

int c[I] = …;  // 各アイテムのコスト
int u[I] = …;  // 各アイテムの効用

dvar int+ x[I] in 0..1;// 変数宣言:0-1変数ベクトル(size:I)

maximize
  sum(I in I) u[i]*x[i];
subject to {
  sum(I in I) c[i]*x[i] <= Capacity;
};
```

ナップサック問題をCPLEXで解く

- データファイル（ksex1.dat）の中身の記述

```
i_max = 15;// アイテムの最大数
Capapacity = 70;// ナップサック容量

c = [15,2,4,7,8,9,12,6,4,5,3,2,7,9,8];// 各アイテムのコスト
u = [10,3,2,6,5,7,10,5,7,3,5,1,6,9,7];// 各アイテムの効用
```

- 例1）
  - ナップサック容量C（=70）リットル
  - 登山用具（アイテム）がn（=15）個あり，体積と効用は下表

登山用具	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
体積（l）	15	2	4	7	8	9	12	6	4	5	3	2	7	9	8
効用	10	3	2	6	5	7	10	5	7	3	5	1	6	9	7

ナップサック問題をCPLEXで解く

- 計算結果の確認
  - CPLEX [解] タブの中身

最適値 = 66

```
// solution (optimal) with objective 66
// Quality Incumbent solution:
// MILP objective                                  6.6000000000e+01
// MILP solution norm |x| (Total, Max)             1.10000e+01  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
        1 0 1 0 1 1 1 1 0 1 1 1 1 1];
```

最適解

15個のアイテムのうち，
2,4,6,7,8,9,11,12,13,14,15番目のアイテムを
ナップサックに詰める　と言う答え

ナップサック問題を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 [ks.mod], データファイル [ksex1.dat]
    → 生成する lpファイル [ksex1.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

```
oplrun -e ksex1.lp ks.mod ksex1.dat  [Enter]
```

  - この結果, モデルファイル [ks.mod] とデータファイル [ksex1.dat] と同じフォルダ内に, lpファイル [ksex1.lp] が出来る(※確認すること)

ナップサック問題を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(kpex1.lp)を読み込み，model へセット

```
model = read("ksex1.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("ksex1.sol")  [Enter]
```

ナップサック問題を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内で実行する

    (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

ナップサック問題をgurobiで解く(2)

- 問題(ex1)を python & gurobi で記述(ks.py)

①

```
# coding: Shift_JIS
from gurobipy import *

# ####### 例題設定 #######
def make_data_ex1():
    cap = 70
    c = [15,2,4,7,8,9,12,6,4,5,3,2,7,9,8]
    u = [10,3,2,6,5,7,10,5,7,3,5,1,6,9,7]
    return cap,c,u
```

②

```
# ####### 定式化 #######
def mss(V,E):
    mod = Model("knapsack problem")

    # 変数設定
    x = {}
    for i in range(len(c)):
        x[i] = mod.addVar(vtype="B", name="x(%s)" % i)    mod.update()

    # 制約条件の設定
    mod.addConstr(quicksum(c[i]*x[i] for i in range(len(c))) <= cap)

    # 目的関数の設定
    mod.setObjective(quicksum(u[i]*x[i] for i in range(len(u))), GRB.MAXIMIZE)
    mod.update()
    mod.__data = x
    return mod
```

③

```
# ####### 実行 #######
if __name__=="__main__":
    cap,c,u = make_data_ex1()          # データの生成
    mod = kp(cap,c,u)                  # モデルの生成
    mod.write("kpex1.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("kpex1.sol")             # 最適解をsolファイルに出力
```

1つのファイル「ks.py」に
①②③の順に記述して保存

ナップサック問題をgurobiで解く(2)

- Pythonファイル(ks.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("ks.py").read() ) [Enter]
```

※python3系の場合

※python2系の場合の命令文は以下

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

ナップサック問題をgurobiで解く(2)

- 実行結果

```
gurobi> exec(open("ks.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 1 rows, 15 columns and 15 nonzeros
Model fingerprint: 0xd8ab50db
Variable types: 0 continuous, 15 integer (15 binary)
Coefficient statistics:
  Matrix range     [2e+00, 2e+01]
  Objective range  [1e+00, 1e+01]
  Bounds range     [1e+00, 1e+00]
  RHS range        [7e+01, 7e+01]
Found heuristic solution: objective 60.0000000
Presolve removed 1 rows and 15 columns
Presolve time: 0.00s
Presolve: All rows and columns removed

Explored 0 nodes (0 simplex iterations) in 0.00 seconds (0.00 work units)
Thread count was 1 (of 20 available processors)

Solution count 2: 66 60

Optimal solution found (tolerance 1.00e-04)
Best objective 6.600000000000e+01, best bound 6.600000000000e+01, gap 0.0000%

optimal value =  66.0

   Variable            X
-------------------------
        x(1)            1
        x(3)            1
        x(5)            1
        x(6)            1
        x(7)            1
        x(8)            1
       x(10)            1
       x(11)            1
       x(12)            1
       x(13)            1
       x(14)            1
gurobi>
```

【演習】ナップサック問題を解く

- ナップサック問題 knapsack problem
  - 容量 $C =$ [　　] のナップサックがある
  - n 個のアイテムがあり，それぞれコストと効用がある
  - アイテム i のコストを c_i ，効用を u_i とする（所与）
  - コスト和が容量を越えない範囲でアイテムを選び，ナップサックに詰める
  - 目的：効用の和が最大になるように，アイテムを選ぶ

アイテム	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
コスト																	
効用																	

- 問
  1. ナップサック容量 C と，アイテム(n=17) のコストと効用を適当に定めよ
  2. 例1と同様に0-1変数を設定し，定式化せよ
  3. 整数計画ソルバーを用いて，最適解を求めよ