

# 最適化・シミュレーション演習

## 第2回 AMPLの定式化

- ・ 授業サポートページ  
<http://www.shiina.mgmt.waseda.ac.jp/optsim/>
- ・ 数理計画による最適化と(離散事象型)シミュレーションに関する演習を行う。使用するソフトウェアは、AMPL(+ GurobiまたはCPLEX), および, Simul8を想定している。
- ・ 演習では, 数理計画による最適化やシミュレーションの実践的能力を身につけることを目指す。履修者は, Cを使用できる環境を有するPCを持参すること。受講者は, 実験室にて, 演習で使用する C, AMPL, Simul8をダウンロードできる。

## ・ 典型的な数理計画問題の定式化と演習

- 輸送問題
- 施設配置問題
- 集合被覆／分割問題
- ・ **輸送問題**: 工場 $m=2$ か所から倉庫 $n=3$ か所への輸送問題を考える。工場 $i$ から倉庫 $j$ への単位輸送量あたりの輸送コスト $c_{ij}$  ( $i=1,...,2, j=1,...,3$ ), 工場 $i$ からの供給量 $S_i$  ( $i=1,...,2$ ), 倉庫 $j$ の需要量 $D_j$  ( $j=1,...,3$ )は表のように与えられている。
- ・ すべての需要を満たし、総費用(輸送費)を最小にする輸送量を求めたい。この問題を数理計画問題として定式化せよ。

倉庫	倉庫1	倉庫2	倉庫3	供給量
工場				
工場1	6	5	4	35
工場2	8	9	5	15
需要量	25	15	10	

2

## 輸送問題の定式化

$$\begin{aligned} \min & \sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij} \\ \text{subject to} & \sum_{i=1}^m x_{ij} = D_j, j=1,...,n \\ & \sum_{j=1}^n x_{ij} = S_i, i=1,...,m \\ & x_{ij} \geq 0, i=1,...,m, j=1,...,n \end{aligned}$$

- ・ 変数  $x_{ij}$ : 工場  $i$  から倉庫  $j$  への輸送量
- ・ 目的関数: 総費用
- ・ 各倉庫での需要制約
- ・ 各工場での供給制約
- ・ 変数の非負制約
- ・ これをAMPLを用いて定式化する。

3

## AMPLによる輸送問題の定式化

#tp.mod

```
param NPLANT ;
param NDEMAND ;
param c {i in 1..NPLANT, j in 1..NDEMAND} >= 0;
param S {i in 1..NPLANT} >= 0;
param D {j in 1..NDEMAND} >= 0;
```

```
var x {i in 1..NPLANT, j in 1..NDEMAND} >= 0;
```

```
minimize totalcost:
sum {i in 1..NPLANT, j in 1..NDEMAND} c[i,j]*x[i,j];
```

```
subject to supply {i in 1..NPLANT}:
sum {j in 1..NDEMAND} x[i,j] = S[i];
```

```
subject to demand {j in 1..NDEMAND}:
sum {i in 1..NPLANT} x[i,j] = D[j];
```

- ・ パラメータ定義
- ・ パラメータ名 {添字 in 集合}
  - ・ 工場数
  - ・ 倉庫数
  - ・ 輸送費用
  - ・ 供給量
  - ・ 需要量
- ・ 決定変数定義
- ・ 決定変数名 {添字 in 集合}
  - ・ 決定変数 x
- ・ 目的関数定義
- ・ 目的関数名: 数式
  - ・ 最小化 totalcost: 目的関数記述
- ・ 制約条件定義
- ・ 制約条件名: 数式
  - ・ supply-const: 制約条件記述
  - ・ demand-const: 制約条件記述

4

## 輸送問題: データ、コマンドスクリプト

```
#tp.dat
param NPLANT := 2;
param NDEMAND := 3;
param varcost :=
[1,*] 1 6 2 5 3 4
[2,*] 1 8 2 9 3 5;
param S :=
1 35 2 15;
param D :=
1 25 2 15 3 10;

#tp.run;
model tp.mod;
data tp.dat;
option solver cplexamp;
solve;
option display_round 6;
expand > to.sol;
display totalcost > tp.sol;
display supply.dual > tp.sol;
display demand.dual > tp.sol;
display x > tp.sol;
exit;
```

- データの記述はかなり融通効く
- モデルファイル指定
- データファイル指定
- ソルバ指定
- 求解

- 展開して表示
- 総費用表示
- 双対変数表示
- 双対変数表示

5

## 施設配置問題

- 関東地方を中心に営業を行ってきた輸入品販売業のある会社は、関西地方に活動を拡大するため、京阪神地方に倉庫の賃借を行う計画を立てている。
- 賃借の候補となる倉庫は  $m$  力所にあつて、第  $i$  地点の倉庫 ( $i=1, \dots, m$ ) の月間処理能力は  $a_i$  (トン/月) で、その経費 (賃借料や維持費など毎月の固定費用) は  $d_i$  (千円/月) である。
- また、関西一円に広がる消費地  $j$  ( $j=1, \dots, n$ ) での輸入品の需要量  $b_j$  (トン/月) と、倉庫  $i$  から需要地  $j$  へのトン当たり輸送費  $c_{ij}$  (千円) が与えられている。
- すべての需要を満たし、毎月の総費用 (倉庫経費 + 輸送費) を最小にする倉庫配置と輸送計画を求めたい。この問題を数理計画問題として定式化せよ。

6

## 施設配置問題の定式化

$$\min \sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij} + \sum_{i=1}^m d_i y_i$$

$$\text{subject to } \sum_{i=1}^m x_{ij} \geq b_j, j = 1, \dots, n$$

$$\sum_{j=1}^n x_{ij} \leq a_i y_i, i = 1, \dots, m$$

$$x_{ij} \geq 0, i = 1, \dots, m, j = 1, \dots, n$$

$$y_i \in \{0, 1\}, i = 1, \dots, m$$

強い定式化

$$x_{ij} \leq b_j y_i, i = 1, \dots, m, j = 1, \dots, n$$

- 変数  $x_{ij}$ : 倉庫  $i$  から消費地  $j$  への輸送量
- 変数  $y_i$  は倉庫  $i$  を賃借するかどうか
- $y_i = 1$  ならば、倉庫  $i$  を借り供給能力は  $a_i$  となる。
- $y_i = 0$  ならば、倉庫  $i$  を借りずに供給能力は 0 となる。
- 実際に問題を扱う場合、このような条件のみで十分である。例えば、倉庫を借りながらも、 $\sum_{j=1}^n x_{ij} = 0$  となる解が得られることはない。なぜなら、各倉庫の固定費  $d_i > 0$  であるから、供給が行われない倉庫を借りるという選択よりも、その倉庫を借りないという解の方が目的関数値が小さく、解として選択されるためである。

7

## AMPLによる施設配置問題

```
#fl.mod;
param NPLANT;
param NDEMAND;

var y {i in 1..NPLANT, j in 1..NDEMAND} >= 0;
var x {i in 1..NPLANT} binary;

param varcost {i in 1..NPLANT, j in 1..NDEMAND} >= 0;
param fixcost {i in 1..NPLANT} >= 0;
param caplimit {i in 1..NPLANT} >= 0;
param demand {j in 1..NDEMAND} >= 0;

minimize cost:
sum {i in 1..NPLANT} fixcost[i]*x[i]
+sum {i in 1..NPLANT, j in 1..NDEMAND} varcost[i,j]*y[i,j];

subject to capacity {i in 1..NPLANT}:
sum {j in 1..NDEMAND} y[i,j] <= caplimit[i]*x[i];

subject to demconst {j in 1..NDEMAND}:
sum {i in 1..NPLANT} y[i,j] = demand[j];

#fl.dat
param NPLANT := 2;
param NDEMAND := 3;
param varcost :=
[1,*] 1 72 2 9 3 34
[2,*] 1 85 2 78 3 79;
;
param fixcost :=
1 1500 2 1400;
param caplimit :=
1 96 2 74;
param demand :=
1 10 2 28 3 21;
```

8

## AMPLによる輸送問題の結果

```
#fl.run;
model fl.mod ;
data fl.dat ;
option solver cplexamp;
option display_round 6;
solve;

display cost > fl.sol;
display sum {i in 1..NPLANT} fixcost[i]*x[i] >
fl.sol;
display sum {i in 1..NPLANT, j in 1..NDEMAND}
varcost[i,j]*y[i,j] > fl.sol;

display x > fl.sol;
display y > fl.sol;
quit;

#fl.sol
cost = 3186.000000
sum {i in 1..NPLANT}
fixcost[i]*x[i] = 1500.000000

sum {i in 1..NPLANT, j in 1..
NDEMAND} varcost[i,j]*y[i,j] =
1686.000000

x [*] :=
1 1.000000
2 0.000000;

y :=
1 1 10.000000
1 2 28.000000
1 3 21.000000
2 1 0.000000
2 2 0.000000
2 3 0.000000
;
```

9

## 集合被覆問題の応用

- あるコンサートをアリーナで開催することになった。そのコンサートは熱狂的なファンが殺到することで知られており、アリーナのコンサートも事故防止のために厳重な警備が必要と考えられる。
- アリーナでは、どこに警備員を配置したらよいかを検討することにした。会場担当者は、会場全体を小さなブロック（たとえば、ブロック1,2,...,m）に分け、仮に警備員をある候補地点（たとえば、候補地点1,2,...,n）に配置したときに警備員がどのブロックを同時にみることができるかを調べた。（例：m=5, n=5の場合、地点1よりブロック1,2を監視可能）
- この情報をもとに、会場全体を最小人数の警備員で警備するための配置とそのときの警備要員の必要人数を求めたい。

候補地点	1	2	3	4	5
ブロック1	○		○		
ブロック2	○	○		○	○
ブロック3		○			○
ブロック4			○	○	
ブロック5					○

10

## 集合被覆問題の定式化

- 問題データ**: 配置候補地点  $(1,2,\dots,n)$ 、警備すべきブロック  $(1,2,\dots,m)$ ; ブロック  $i$  が候補地点  $j$  から監視可能なら  $a_{ij}=1$  (さもなければ0) とする制約係数行列  $A=[a_{ij}]$
- 変数(=列)の定義**:  $x_j$ =警備員を地点  $j$  に配置するとき1、さもなければ0
- 目的関数(警備員を配置する地点数最小)**:  
最小化 
$$z = \sum_{j=1}^n c_j x_j$$
- 制約条件(各ブロックは監視可能)**:  
制約 
$$\sum_{j=1}^n a_{ij} x_j \geq 1, \quad i = 1, \dots, m$$
  
$$x_j \in \{0,1\}$$

```
min x1 + x2 + x3 + x4 + x5
subject to x1 + x3 ≥ 1
           x1 + x2 + x4 + x5 ≥ 1
           x2 + x5 ≥ 1
           x3 + x4 ≥ 1
           x5 ≥ 1
           x1, x2, x3, x4, x5 ∈ {0,1}
```

### 集合被覆問題

11

## AMPLによる集合被覆問題

```
#sc.mod
param NBLOCK ;
param NPOINT ;
param a {i in 1..NBLOCK, j in 1..NPOINT} binary;
var x {j in 1..NPOINT} binary;

minimize totalcost: sum {j in 1..NPOINT} x[j];

subject to watch {i in 1..NBLOCK}:
sum {j in 1..NPOINT} a[i,j]*x[j] >= 1;

#sc.dat
param NPOINT := 5;
param NBLOCK := 5;
param a :=
1 1 1 2 0 1 3 1 1 4 0 1 5 0
2 1 1 2 2 1 2 3 0 2 4 1 2 5 1
3 1 0 3 2 1 3 3 0 3 4 0 3 5 1
4 1 0 4 2 0 4 3 1 4 4 1 4 5 0
5 1 0 5 2 0 5 3 0 5 4 0 5 5 1;
```

```
#sc.run
model sc.mod ;
data sc.dat ;
option solver cplexamp;
option display_round 6;
solve;

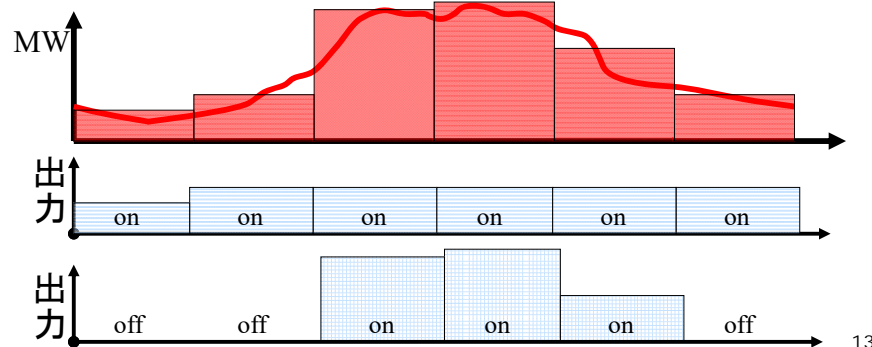
expand > sc.sol;
display totalcost > sc.sol;

display x > sc.sol;
display _ampl_time > sc.sol;
display _total_solve_time
> sc.sol;
exit;
```

12

## 発電機起動停止問題

- unit commitment: 電力システムにおけるスケジューリング問題
- 各時間帯に与えられた電力需要を満足するように発電機のオン/オフおよび発電量を決定



13

## 起動停止問題モデル

- 問題(UC):  $I$  台の発電機による  $T$  時間にわたる電力供給
- 変数  $u_{it}$  は発電機  $i$  の時間  $t$  における状態を表す 0-1 変数
- 変数  $x_{it}$  は発電機  $i$  の時間  $t$  の出力
- 関数  $f_i(x_{it})$  は発電機の燃料費を表すの凸2次関数
- 関数  $g_i(u_{i,t-1}, u_{it})$  は発電機の起動費用を表し  $g_i(0,1) > 0$  となり、それ以外の場合には 0 となる関数

## 起動停止問題—目的関数

- 総費用の最小化
- 燃料費  $f_i(x_{it})$ : 凸2次関数
- 起動費用  $g_i(u_{i,t-1}, u_{it})$ :  $g_i(0,1) > 0$

$$\min \sum_{i=1}^I \sum_{t=1}^T g_i(u_{i,t-1}, u_{it}) + \sum_{i=1}^I \sum_{t=1}^T f_i(x_{it}) u_{it}$$

0-1変数  $u_{it}$ : 発電機の状態(稼動/停止)

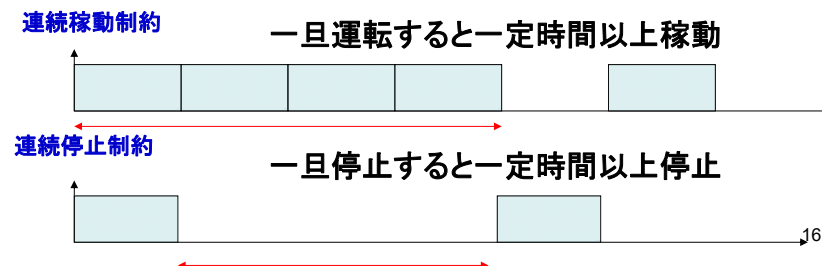
非負変数  $x_{it}$ : 発電機の出力

$I, T$ : 発電機数、計画期間数

15

## 起動停止問題—制約

- 需要制約  $\sum_{i=1}^I x_{it} \geq d_t, \forall t$
- 運転出力の上下限制約  $q_i u_{it} \leq x_{it} \leq Q_i u_{it}, u_{it} \in \{0,1\}, \forall i, \forall t$
- 連続稼動制約:  $L_i$  期間  $u_{it} - u_{i,t-1} \leq u_{i\tau}, \tau = t+1, \dots, \min\{t+L_i-1, T\}, \forall i, t=2, \dots, T$
- 連続停止制約:  $l_i$  期間  $u_{i,t-1} - u_{it} \leq 1 - u_{i\tau}, \tau = t+1, \dots, \min\{t+l_i-1, T\}, \forall i, t=2, \dots, T$



16