交通工学における行動モデル
→モデルの解法にまつわる話

井料　隆雅（神戸大学）

27Sep2014

解法

- ネットワーク交通流理論においては，モデルの定式化が手続きではなく，「満たすべき条件」で記述されることがある（ex. 均衡配分）.
- 定式化できても解けないと意味が無いので，解法は重要

数理計画としての定式化

- パターン１：交通主体個々人の効用を最大化させる行動を見つける問題.
  ex: 経路探索問題
- パターン２：システム全体の最適解を求める問題．ex. 最適配分問題
- パターン３：均衡状態を等価な最適化問題に置き換える.
  - ex. ポテンシャルゲーム（対称な相互作用）

ポテンシャルゲーム？

F:利得（効用）

3.2. Potential games

We call the game F a *potential game* if it satisfies

$$\frac{\partial F_i^p}{\partial x_j^q}(x) = \frac{\partial F_j^q}{\partial x_i^p}(x) \quad \text{for all } i \in \mathcal{S}^p,\ j \in \mathcal{S}^q,\ p, q \in \mathcal{P},\ \text{and } x \in X.$$

This requirement is stated more concisely as

$$DF(x) \text{ is symmetric for all } x \in X.$$

Hofbauer, J., & Sandholm, W. H. (2007). Evolution in games with randomly disturbed payoffs. *Journal of Economic Theory*, *132*(1), 47–69.

ポテンシャルゲーム？

Since the derivative of F is symmetric, every potential game F admits a *potential function* $f\colon \bar{X} \to \mathbf{R}$: that is, a function that satisfies $\nabla f(x) = F(x)$ for all $x \in X$. Hofbauer [22] and Sandholm [35] show that this potential function serves as a Lyapunov function for a wide range of unperturbed evolutionary dynamics, and so can be used to establish global convergence results.

解釈すると:
ポテンシャル f を最大化できれば,
それが均衡解である.

Hofbauer, J., & Sandholm, W. H. (2007). Evolution in games with randomly disturbed payoffs. *Journal of Economic Theory*, *132*(1), 47–69.

バス問題

- x: バスを使う人，　y: 車を使う人
- バスの効用 F(x,y)=x
- 車の効用　　G(x,y)=y
- Fy=Gx=0, Fx=Gy=0 なので対称.
  ポテンシャルは　f(x,y)=½(x^2+y^2)
- 制約条件（x+y=nと非負制約）下で,
  fが最大化される答えは？？

問題の性質と解法

- 凸問題
  - 線形計画，線形制約つき非線形問題が典型的
  - ほぼ確実に解けるが計算速度が問題になりがち
- 動的計画法
  - 問題を入れ子に分割できる際に有効
- 組み合わせ最適化問題
  - 省略

基本的な解法・・・
どこかで習っていると思うので省略

解法の実装(オープンソース)

- 線形計画問題:glpk
- 非線形計画問題
  - Rのoptim:制約なし／矩形制約
    - 導関数を用いない方法を含め，さまざまな方法が実装されているのが強み.
    - 一般的な制約の扱いは面倒
      (ペナルティ関数:あまり効率がよくない.
      実行可能領域の境界部分で性質が悪い)
  - 線形制約つきであれば，Frank-Wolfeを自分で組むのはわりとお気軽(CからglpkをCall)

解法の実装(プロプライエタリ)

- エクセルのソルバ:簡単に試すにはよい.
- MATLABのoptimization toolbox
  - 信頼領域 Reflective 法
  - 有効制約法
  - SQP
  - 内点法

  http://www.mathworks.co.jp/jp/help/optim/ug/constrained-nonlinear-optimization-algorithms.html

glpk

- Gnu Linear Programming Kit
  https://www.gnu.org/software/glpk/
  - Cygwinに含まれるバイナリを使うのが楽
    glpsol.exe
- 単独でも，C, C++から呼び出しても使用可
- GNU MathProg という言語で問題を記述する.
  - AMPL modeling language（商用）のサブセット

GNU MathProg

- 例題
  - とんかつ弁当とさば味噌煮弁当を販売
    - とんかつ弁当の利益:300円／個
    - さば味噌煮弁当の利益:150円／個
  - 弁当を製造するために使える時間は300分
    - とんかつ弁当の製造時間:6分／個
    - さば味噌煮弁当の製造時間には2分／個
  - 客は「肉オンリー」か「魚オンリー」にわかれる
    人数はそれぞれ40人と60人.
    それ以上の客には売れない売れない.

GNU MathProg

$$\mathrm{Max.}\ z = 300x + 150y$$

sub. to.

$$x \le 40,\ y \le 60,\ 6x + 2y \le 300,\ x \ge 0,\ y \ge 0$$

とんかつ: x（個），さば: y（個）

GNU MathProg

```
var tonkatsu>=0;
var saba>=0;
maximize profit:2*tonkatsu+saba;
s.t. c1: tonkatsu<=40;
s.t. c2: saba<=60;
s.t. c3: 6*tonkatsu+2*saba<=300;
solve;
display tonkatsu,saba;
end;
```

```
iryo@Lets2013 ~
$ glpsol.exe -m tonkatu.glpk.txt
GLPSOL: GLPK LP/MIP Solver, v4.52
Parameter(s) specified in the command line:
 -m tonkatu.glpk.txt
Reading model section from tonkatu.glpk.txt...
tonkatu.glpk.txt:9: warning: final NL missing before end of file
9 lines were read
Generating profit...
Generating c1...
Generating c2...
Generating c3...
Model has been successfully generated
GLPK Simplex Optimizer, v4.52
4 rows, 2 columns, 6 non-zeros
Preprocessing...
1 row, 2 columns, 2 non-zeros
Scaling...
 A: min|aij| =  2.000e+00  max|aij| =  6.000e+00  ratio =  3.000e+00
Problem data seem to be well scaled
Constructing initial basis...
Size of triangular part is 1
*     0: obj =   0.000000000e+00  infeas =  0.000e+00 (0)
*     3: obj =   1.200000000e+02  infeas =  0.000e+00 (0)
OPTIMAL LP SOLUTION FOUND
Time used:   0.0 secs
Memory used: 0.1 Mb (96737 bytes)
Display statement at line 8
tonkatsu.val = 30
saba.val = 60
Model has been successfully processed
```

GNU MathProg

```
set Bento;
param maxCustomer {i in Bento};
param profit    {i in Bento};
param cookTime  {i in Bento};
param maxCookTime;
var kazu {i in Bento};

maximize totalProfit:
         sum{i in Bento} profit[i]*kazu[i];
s.t. customer {i in Bento}:
         kazu[i] <= maxCustomer[i];
s.t. time: sum{i in Bento}
           cookTime[i]*kazu[i]<=maxCookTime;
solve;
display kazu;
end;
```

GNU MathProg

```
data;
set Bento := Makunouchi, Shokado, Sushi, Tonkatsu;

param maxCustomer :=
Makunouchi  240          Shokado     120
Sushi       150          Tonkatsu    300;

param profit :=
Makunouchi  180          Shokado     300
Sushi       280          Tonkatsu    210;

param cookTime :=
Makunouchi  20           Shokado     30
Sushi       25           Tonkatsu    12;
param maxCookTime := 6000;
end;
```

```
iryo@Lets2013 ~
$ glpsol.exe -m bento.model -d bento.data
GLPSOL: GLPK LP/MIP Solver, v4.52
Parameter(s) specified in the command line:
 -m bento.model -d bento.data
Reading model section from bento.model...
14 lines were read
Reading data section from bento.data...
21 lines were read
Generating totalProfit...
Generating customer...
Generating time...
Model has been successfully generated
GLPK Simplex Optimizer, v4.52
6 rows, 4 columns, 12 non-zeros
Preprocessing...
1 row, 4 columns, 4 non-zeros
Scaling...
 A: min|aij| =  1.200e+01  max|aij| =  3.000e+01  ratio =  2.500e+00
GM: min|aij| =  1.000e+00  max|aij| =  1.000e+00  ratio =  1.000e+00
EQ: min|aij| =  1.000e+00  max|aij| =  1.000e+00  ratio =  1.000e+00
Constructing initial basis...
Size of triangular part is 1
*     0: obj =   0.000000000e+00  infeas =  0.000e+00 (0)
*     2: obj =   8.988000000e+04  infeas =  0.000e+00 (0)
OPTIMAL LP SOLUTION FOUND
Time used:   0.0 secs
Memory used: 0.1 Mb (120800 bytes)
Display statement at line 13
kazu[Makunouchi].val = 0
kazu[Shokado].val = 0
kazu[Sushi].val = 96
kazu[Tonkatsu].val = 300
Model has been successfully processed
```