working paper 2006.4.8. (revised 2006.4.14.)

社会的選択問題のための論理プログラミング
A logic programming for social choice problems

犬童健良 †
Kenryo INDO †
† 関東学園大学 経済学部経営学科
† Department of Management, Faculty of Economics, Kanto Gakuen Univ.

要旨:
本論文は論理プログラミング言語による社会的選択問題のモデリング手法（LPMSC）を提案した．また実装に PROLOG を用い，2 人 3 代替案の場合のギッバード＝サタスウェイトの定理を自動証明した．さらにこの場合の制限領域で，操作不可能な社会的選択関数が存在する非独裁的許容領域を自動設計した．

1. はじめに

社会的選択理論(Arrow, 1963; Sen, 1982)は，投票や市場経済などの社会的意思決定のメカニズムを抽象化し，診断･設計するための基礎である．今日，その応用は，情報技術にも及ぶ．[1]

より具体的には，個人の選好順序や社会的な意思決定ルールをモデリングし，その特性を診断したり、あるいは所定の要求を満たす意思決定ルールを設計したりする．

しかし，例えば操作不可能な意思決定ルールは，非制限領域において独裁性を含意してしまう（ギッバード＝サタスウェイトの定理による）ので，望ましい社会的選択を行うには，いずれかの条件を緩和しなければならないが，許容範囲内でどのように変更すればよいだろうか．

意思決定支援システムの研究では，多基準モデルを用い，GUI を備えた完成度の高いものがあるが，しかし筆者の知る限り，そのような実装やシミュレーションは殆どなされていない．そこで本論文では実際に PROLOG によって社会的選択問題をモデリングし，まず上記の不可能性定理を自動証明する．例外的に犬童(2002)がナッシュ遂行に PROLOG を応用しているが，そこで用いられた素朴な生成検査法では，これは事実上不可能であった．

また論理プログラミングによって社会的選択関数を設計するための LPMSC 手法を提案し，そのシステム実装例を示す．まず次節で LPMSC 手法について概略し，続く 3 節で社会的選択に対する制約条件をモデリングする．第 4 節はギッバード＝サタスウェイトの定理を自動証明する．また第 5 節において，非独裁的な制限領域を自動設計する．最後にまとめとする．

2. 社会的選択の論理モデリング

本論文では，緩やかに，その実装を伴った社会的選択問題の実行可能な論理モデリングのことを LPMSC 法と呼ぶ．その特色は以下のように述べられる．

●個人の選好順序，社会全体の意思決定ルール，およびそれらが満たすべき，あるいは避けるべき諸条件（制約）を，論理モデルないし論理プログラミングによって表す．

●上記の論理モデル，とくに意思決定ルールを，シミュレーション実行できるプログラミング言語に翻訳する（間接的 LPMSC 法）か，あるいは直接実行できる（直接的 LPMSC 法）．

●直接的 LPMSC の簡便な方法として，PROLOG による実装を推奨する．またルール生成の際，制約を利用し，計算効率を高める．

なお，標準的な PROLOG 言語については Clocksin and Mellish (2003)などを参照されたい．また最適化問題への論理と制約に基づくアプローチにかんしては，Hooker(2000)が詳しい．

LPMSC 法のユーザーは，まず以下に述べる基本要素，次いで諸条件をモデリングする．またシミュレーション（自動定理証明）を用い，意思決定ルールの要求仕様を定める．さらに第 4 節に述べる不可能性定理を避けるため，一部の条件を緩和し，現実的な落としどころを見出すだろう．

[1] より一般的には，メカニズムデザインと呼ばれ，しばしばゲーム理論を伴う．そのより新しい応用分野はマルチエージェントシステムやグループ意思決定支援である．例えば，オークションのプロトコール(Sandholm, 1999), 会議のスケジューラー(Ephratiet *et al*., 1994)などである．自律分散環境では，利己的なエージェントたちの計算資源利用が，システム全体の目的に反しないように調整する必要がある．しかし従来の研究では，真の選好を聞き出すため，VCG メカニズム（クラーク税）に頼っていた．そのような（準線形の）制限的領域では選好変化への対処も必要だ．例えば，協調的知識ベースシステム(Wong, 1994)では信念更新モデルが流用されている．さらに伝統的な自発的メカニズムと同じく，結局，それらを使ってもらえない可能性を排除できない．

```
set_of_alternatives([a,b,c]).
set_of_agents([1,2]).
alternative(X):-
  set_of_alternatives(A),
  member(X,A).
agent(J):-
  set_of_agents(N),
  member(J,N).
possible_ranking( r(1), [a,b,c]).
possible_ranking( r(2), [a,c,b]).
possible_ranking( r(3), [b,a,c]).
possible_ranking( r(4), [b,c,a]).
possible_ranking( r(5), [c,a,b]).
possible_ranking( r(6), [c,b,a]).
prefer_x_to_y(A,B, R):-
   possible_ranking( R, O),
   append( _,[A|C],O),
   member(B,C).
possible_ranking_of( J, R, O):-
   agent(J),
   possible_ranking(R, O).
```

図 1　代替案，エージェント，選好順序

```
% the domain
all_profile_of_rankings(L):-
  findall((R1,R2),
    profile_of_rankings( [R1,R2]),
  L).
% a generic constructor
auto_scf( FL,Property):-
  all_profile_of_rankings( L),
  auto_scf( FL, L, Property).
% initial stage of recursions
auto_scf( [],[], _).
% scf without constraints
auto_scf( [R-> X | H], [R|Q], free):-
  auto_scf( H,Q,free),
  alternative( X).
```

図 2　社会的意思決定ルール（SCF の場合）

```
% no taboo alternative
citizens_sovereignty(Scf):-
  forall(
    alternative(A),
    member( _->A, Scf)
  ).
%  Pareto principle
is_Pareto_efficient_at_profile( (R1, R2)->Y):-
  ¥+ (
    prefer_x_to_y( X, Y, R1),
    prefer_x_to_y( X, Y, R2)
  ).
is_Pareto_efficient_scf( Scf):-
  forall(
    member( V, Scf),
    is_Pareto_efficient_at_profile( V)
  ).
%  dictatorship
is_dictatorial_scf( Scf,J):-
  is_ranking_of(J,L,R),
  forall(
    member(L->X, Scf),
    ¥+ prefer_x_to_y( _,X, R)
  ).
is_non_dictatorial_scf( Scf):-
  ¥+ is_dictatorial_scf( Scf, _).
is_ranking_of(1,(R,_),R).
is_ranking_of(2,(_,R),R).
```

図 3　市民主権，パレート原理，（非）独裁性

```
%  manipulability
is_manipulable_at_profile( Scf,(R,R2)->X,(Q,R2)->Y,1):-
  member( (Q,R2)->Y, Scf),
  prefer_x_to_y( Y, X, R).
is_manipulable_at_profile( Scf,(R1,R)->X,(R1,Q)->Y,2):-
  member( (R1,Q)->Y, Scf),
  prefer_x_to_y( Y, X, R).
is_strategy_proof_scf( Scf):-
  forall(
    member( V, Scf),
    ¥+ is_manipulable_at_profile( Scf, V,_,_)
  ).
% (Maskin) monotonicity
preference_reversal( (R1,R2)->X, (Q1,Q2), (J,W)):-
  is_ranking_of(J,(R1,R2),R),
  is_ranking_of(J,(Q1,Q2),Q),
  prefer_x_to_y( X, W, R),
  prefer_x_to_y( W, X, Q).
is_non_monotonic_at_profile( Scf, R->X, Q->Y):-
  member( Q->Y, Scf),
  Y ¥= X,
  ¥+ preference_reversal( R->X, Q, _).
is_monotonic_at_profile( Scf, R->X):-
  ¥+ is_non_monotonic_at_profile( Scf, R->X, _).
is_monotonic_scf( Scf):-
  forall(
    member( V, Scf),
    is_monotonic_at_profile( Scf, V)
  ).
```

図 4　操作（不）可能性，（非）単調性

```
% strategy-proof scf

auto_scf( [R-> X | H], [R|Q], sp):-
   auto_scf( H,Q,sp),
   alternative(X),
   ¥+ is_manipulable_at_profile( H, R->X,_,_),
   ¥+ (
     member(V, H),
     is_manipulable_at_profile( [R->X|H], V,R->X,_)
   ).
```

図 5　操作不可能な SCF を生成するコード

```
?- auto_scf( F,sp),citizens_sovereignty(F),
show_scf(F),nl,fail.

col=r(#) [1, 2, 3, 4, 5, 6]
---------------------------
row=r(1) [a, a, b, b, c, c]
row=r(2) [a, a, b, b, c, c]
row=r(3) [a, a, b, b, c, c]
row=r(4) [a, a, b, b, c, c]
row=r(5) [a, a, b, b, c, c]
row=r(6) [a, a, b, b, c, c]

col=r(#) [1, 2, 3, 4, 5, 6]
---------------------------
row=r(1) [a, a, a, a, a, a]
row=r(2) [a, a, a, a, a, a]
row=r(3) [b, b, b, b, b, b]
row=r(4) [b, b, b, b, b, b]
row=r(5) [c, c, c, c, c, c]
row=r(6) [c, c, c, c, c, c]

No
?- auto_scf( F,sp),¥+ is_monotonic_scf(F).

No
?- auto_scf( F,monotonic),member(U,F),(F).
is_manipulable_at_profile(F,U,V,J).

No
?- auto_scf( F,monotonic),citizens_sovereignty(F),
¥+ is_Pareto_efficient_scf(F).

No
?-
```

図 6　SP⇒D，SP⇔M および M&CS⇒P の証明

さて，具体的にモデリング対象となる社会的選択問題の基本要素は以下のようである．

●有限な代替案の集合 A={a, b, …}，エージェントの集合 N＝｛1, 2, …, n｝.

●選好. 各エージェントのペア比較 $r^i \subseteq A \times A$, $i \in N$ ないしランキング．すべての完備的，推移的，非反射的（弱選好なら反射的）な 2 項関係を Σ と書く．許容的選好順序は $R \subseteq \Sigma$，$r^i \in R$.

また社会的意思決定ルールには以下の 2 種類がある．

●社会厚生関数（SWF）の場合，全員の選好順序からなる任意のプロフィール $r^N=(r^1, r^2, \ldots, r^n) \in R^N = \times_{i \in N} (R^i)$ に対し，完備的かつ推移的な選好順序を定義する．$SWF \subseteq R^N \times R$.

●社会的選択関数（SCF）の場合，同じく任意の選好順序プロフィールに対して，ある代替案社会的選択対応(SCC)ではその部分集合）を定義する．$SCF \subseteq R^N \times A$.

SCF の方が，投票の手続きを抽象化したものとして解釈しやすいだろう．本論文で SCF の場合を論じる. なお SWF の方は Indo(2006)で論じる．

図 1 と図 2 に，2 人 3 代替案の非制限領域，すなわち A={a, b, c}，N={1, 2}の場合の選好順序とその SCF を，なんら明示的な制約なしに生成するコードと実行例をそれぞれ示した．

3. 社会的選択に対する制約条件

社会的意思決定ルールとしての SWF や SCF の特性ないし基準として代表的なものを以下に挙げる．

●非制限的領域 (条件 U)．定義域である可能な選好順序を制限しない．つまり $R = \Sigma$.

●市民主権（条件 CS)．値域が制限されない．

●弱いパレート原理（条件 P)．全員一致ならば，社会全体もそれを選ぶ．

●無関係な代替案からの独立性（条件 IIA)．2 者択一は当該の代替案についての選好パタンだけで決定される．

●非独裁性（条件¬D)．社会の決定がつねにある一人の欲すままにならない．

●操作不可能性（条件 SP)．あるメンバーが自分の選好を偽ることで，社会全体の決定を有利に導くことができない．

●単調性（条件 M)．あるプロフィールで選ばれた案は，誰かにとってランク落ちしないかぎり，プロフィール変更後も選ばれる．

図 2 の方法で生成した SCF は，条件 U を満たす．また図 3 と図 4 に上記の制約に対応するコードを抜粋して示した．その他．多数決，公平性，簡潔性，計算可能性など，さまざまな基準があるが，省略する．また強選好のとき，次の性質が知られる．

$$M \Leftrightarrow SP. \quad (1)$$

$$CS\&M \Rightarrow P. \quad (2)$$

式(1)は Muller and Satterthwaite (1977)による．

本論文のコードにより，読者は実験を通じて，これらの制約のはたらきを容易に確かめられる．とくに条件 SP または M（定理 A の場合 IIA）は，これらはそれ自体で特定の値を定めないが，制約としては強力である．実際，合理的な選好順序を前提として，プロフィール間で決定性を伝播する．

4. 不可能性定理の自動証明

社会的選択理論の初期の成果は，望ましい社会的意思決定ルールとしての SCF の困難性を示す以下の定理に集約される．

定理 GS. (Gibbard, 1973; Satterthwaite, 1975)　3 つ以上の代替案があり，SCF が条件 U および条件 CS を満たすとき，SP⇒D.

また SCW についての次の定理は，いわゆるアローの一般不可能性定理として知られる．

定理 A.　(Arrow, 1963)　3 つ以上の代替案があるとき，U&P&IIA&¬D を満たす SWF はない．

なお次節に示すように両定理は論理的に同値である．また類似の方法によって証明できる（例えば Reny (2001)を見よ）．

一般の場合の定理の解説や証明については，文献に譲る．以下では２人３代替案かつ強選好（線形順序）の場合の GS 定理の自動証明を示す．

原理的には，前節で示したコードを用い，SCF を生成してから，操作可能性を事後的にチェックできるが，この方法で所定の要求を満たす SCF や SWF を作るのは，効率的でないか，あるいは事実上不可能である．

そこで, 図 5 のように再帰的に SCF を生成する過程で，条件 SP を用いて割当てできる代替案を制限する．ただし，図 4 で示したものと異なり，双方向の制約を課していることに注意する．すなわち，１）現時点でのプロフィールから見て，すでに割当てたプロフィールへと操作可能なものがないか，２）また現時点のプロフィールが，すでに値を割当てたプロフィールから見て操作可能な変更先になっていないか，両方チェックする．

主張 1．図 5 に示すプログラムは，図 1～4 のプログラムと共に用いることによって，3 代替案 2 人の場合の操作不可能な SCF を枚挙し，かつ定理 GS を正しく模擬する．

論証 : すべてのプロフィールをチェックし終えた時点では，条件 SP が満たされている(図 5 の is_strategy_proof/1 で確かめられる)．また条件 SP に反しない割当て可能性が，この方法により排除されることはない．よって主張 1 の前半は満たされる．実際，図 6 上に示すとおり，２種類の独裁的ルールが導出され，かつそれ以外の SCF は作れない．[2] よって主張内容は満たされた．

[2] ちなみにその実行時間は PC で SWF-Prolog 5.0.9 を用い 0.125 秒（表なしのとき 0.078 秒）であった．また条件 SP を

図 7　非独裁的な制限領域

```
?- auto_restricted_domain(L),¥+ ¥+ (auto_scf( F,sp),
citizens_sovereignty(F),is_non_dictatorial_scf(F)),
nl,write(L),fail.

[r(1), r(2), r(3), r(4), r(5)]
[r(1), r(2), r(3), r(4), r(6)]
[r(1), r(2), r(3), r(5), r(6)]
[r(1), r(2), r(3), r(5)]
[r(1), r(2), r(3), r(6)]
[r(1), r(2), r(4), r(5), r(6)]
[r(1), r(2), r(4), r(5)]
[r(1), r(2), r(4), r(6)]
[r(1), r(2), r(4)]
[r(1), r(2), r(6)]
[r(1), r(3), r(4), r(5), r(6)]
[r(1), r(3), r(4), r(5)]
[r(1), r(3), r(4), r(6)]
[r(1), r(3), r(5), r(6)]
[r(1), r(3), r(5)]
[r(1), r(3), r(6)]
[r(1), r(4), r(5), r(6)]
[r(1), r(4), r(5)]
[r(1), r(4), r(6)]
[r(1), r(5), r(6)]
[r(1), r(6)]
[r(2), r(3), r(4), r(5), r(6)]
[r(2), r(3), r(4), r(5)]
[r(2), r(3), r(4), r(6)]
[r(2), r(3), r(4)]
[r(2), r(3), r(5), r(6)]
[r(2), r(3), r(5)]
[r(2), r(3), r(6)]
[r(2), r(4), r(5), r(6)]
[r(2), r(4), r(5)]
[r(2), r(4), r(6)]
[r(2), r(4)]
[r(3), r(4), r(5)]
[r(3), r(5), r(6)]
[r(3), r(5)]

No
?-
```

また図 6 下側は式(1)および式(2)を証明したものである．よって条件 M を用いても同じ結果を得る．表出力については，読者の練習問題とする．

5. 非独裁的制限領域の設計

前節の定理 A における条件 U は，非自明な 3 代替案の存在に緩められる(Arrow, 1963)．また以下の定理は，制限領域における 2 種類のモデリングを，2 人 3 代替案の場合に結びつける．

定理 KM. (Kalai and Muller, 1977)　許容領域 R⊆Σ上で，2 人以上，ある SWF が非独裁性¬D を満たす⇔2 人のとき，ある SWF が¬D を満たす⇔2 人以上のとき，ある操作不可能な SCF が¬D を満たす．

またその必要十分条件（領域の分解可能性[3]）は 3 代替案の関係として述べられるが，コードは紙幅の都合により省く．

本節では，前節のコードを用い，SP&CS&¬D を満たす許容領域 R⊆{r(k)|k=1, 2, …, 6}を実際に生成してみよう．図 7 はその実行結果である．若干のコード変更をしているが，読者に委ねる．

図 7 に列挙された領域は，もちろん非制限的¬U であり，かつ自明なペア比較（つまり全員の判断が常に一致）の集合が空である．

満たす SCF は，条件 CS を満たさない 15 個を含め，各プロフィールに対して暫定的に値割当てを 490 回成功させており，またその後のバックトラックによる棄却は 11 回である．これは 17 個分のプロフィール総数より少ない．

[3] ただし各代替案がトップランクになる 3 つのケースでは SCF で選べないので CS を満たさないが分解可能となる．また強選好（線形順序）なので循環は除去する必要がある．

6. まとめ

SWF や SCF の性質は社会的意思決定システムの論理的仕様に対して，厳密な限界線を引く．本論文では，素朴な生成検査でできなかった不可能性定理の証明は，累積的制約の方法により解決できた．また制限領域での自動設計についても可能になった．必ずしも社会的選択問題を深く洞察できたといえないが，操作不可能性や権限設定の必要を伴うさまざまな現実的なシステムへの応用が考えられる．今後の課題としたい．

参考文献

[1] Arrow, K., *Social Choice and Individual Values*, Yale University Press, 1963.

[2] Clocksin, W. F. and Mellish, C. S., *Programming in Prolog: Using the ISO Standard*, 5th edition, Springer, 2003.

[3] Ephrati, E., Zlotkin, G., and Rosenschein, J. S., “Meet your density: A non-manipulable meeting scheduler,” *CSCW’94*, pp. 22-26, 1994.

[4] Gibbard, A., “Manipulation of voting schemes: A general result,” *Econometrica*, Vol. 41: 587-602, 1973.

[5] Kalai, E. and Muller, E., “Characterization of domains admitting nondictatorial social welfare functions and nonmanipulable voting procedures,” *Journal of Economic Theory*, Vol. 16: 457-469, 1977.

[6] Hooker, J., *Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction*, John Wiley & Sonsm, 2000.

[7] 犬童健良, “PROLOG でナッシュ遂行理論を学ぶ,” 経営情報学会秋季全国発表大会予稿集, pp.224-227, 2002.

[8] Indo, K., “An automated proof of the Arrow’s theorem,” mimeo, 2006.

[9] Muller, E. and Satterthwaite, M. A., “The equivalence of strong positive association and strategy-proofness,” *Journal of Economic Theory*, Vol. 14: 412-418, 1977.

[10] Reny, P. J., “Arrow's theorem and the Gibbard-Satterthwaite theorem: A unified approach,” *Economic Letters*, Vol. 70: 99-105, 2001.

[11] Satterthwaite, M. A., “Strategy-proofness and Arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions,” *Journal of Economic Theory*, Vol. 10: 187-217, 1975.

[12] Sandholm, T. W., “Distributed rational decision making,” in G. Weiss (ed.), *Multiagent Systems*, The MIT Press, pp. 201-258, 1997.

[13] Sen, A., *Choice, Welfare and Measurement*, The MIT Press, 1982.

[14] Wong, S. T. C., “Preference-based decision making for cooperative knowledge-based systems,” *ACM trans. Information Systems*, Vol. 12, No. 4: 407-435, 1994.