

Social Choice Logic Programming I[†]

Kenryo Indo
Kanto Gakuen University



**Faculty of Economics, Kanto Gakuen University,
Faculty Seminar, 8 October 2008**

†This work was partially supported by the Academic Research Expense of Kanto Gakuen University, 2008. The author gratefully acknowledges the generous support of the chief director Junichi Matsudaira and the preseedent Sumio Hyugaji.

Introduction

□ **S**ocial **C**hoice **L**ogic **P**rogramming

- Social Choice Theory (Axiomatic approach)

- Logic Programming (Computational approach)

 - **Commonality**: logic-based modeling-and-proving

- SCLP toolkit (a collection of PROLOG programs)

 - Check my web:

 - http://www.xkindo.net/cog_dec/wp/mplsc.html

 - **Near-miss**: COMSOC (Computational Social Choice)

 - <http://www.csc.liv.ac.uk/~pwg/COMSOC-2008/>

- (Potential) applications

 - Social sciences: Economics, Managerial science, Political science, Jurisprudence

 - Decision/Cognitive sciences, AI, OR/MS: Multiagent systems, Intelligent Scheduling and SCM, multicriteria decision making, Bounded rationality, Multiple-self

Logic programming and PROLOG

□ PROLOG (**PRO**gramming in **LOGIC**)

- A PROLOG Program is a conjunction of **horn-clauses**. A horn-clause represents **if** relation, **Head <- Body**.
- PROLOG is a system of automated theorem proving. A 'query' typed after system prompt '?-' means the refutation of the goal clause.
- An example:

□ **(Proposition) The number of prime numbers less than 10 is 4.**

Proof. By using the following clauses asserted:

```
is_divisor_of(P,K,H):- member(P,H), 0 is mod(K,P).
prime([], [1]).
prime(L, [K|H]):- prime(L,H), \+ \+ is_divisor_of(_,K,L).
prime([K|L], [K|H]):- prime(L,H), \+ is_divisor_of(_,K,L).
```

This program uses a **recursion**. It can be seen as a sort of mathematical induction.

```
?- prime(I, [9,8,7,6,5,4,3,2,1]), length(I, N).
I = [7, 5, 3, 2],
N = 4 .
```



Social Choice Theory

□ **Some Classical Problems**

- Preference (ordering, or ranking) aggregation
- Strategy-proof voting procedure (and mechanism design)
- Domain conditions

□ **Some Classical Results**

- Arrow's general (im)possibility theorem
 - Gibbard-Satterthwaite theorem
 - Sen's possibility theorem, and so on
-

Preference Aggregation and Arrow's social welfare function

- SWF (social welfare function): for each profile of individual rankings a SWF should map it to an aggregated ranking of society, or a group, as a whole
- **Axioms of SWF:**
 - Linearity (or Transitivity) of rankings **(T)**
 - Unrestrictedness of rankings **(U)**
 - Pareto Condition **(P)**
if every individual prefers x over y , then so should society
 - Independence of Irrelevant Alternatives **(I)**
social ranking of x over y should only depend on individuals' ranking over x and y

□ A ranking, or (weak) ordering is a complete transitive binary relations over the set of alternatives. A linear ordering is an asymmetric ordering.

Arrow's general (im)possibility theorem

□ more ..

■ Nondictatorship (-D)

no single individual is able to impose her or his ranking as the aggregated ranking

Proposition 1 (Arrow, 1951) Let us assume that (T) and (U) are satisfied. Then, for more than two alternatives, there exists no SWF that satisfies all of (P), (I) and (-D). Equivalently, (P) & (I) implies dictatorship.

K.J. Arrow. Social Choice and Individual Values.
2nd edition, Wiley, 1963. (Reprint by Y.U.P.)

SCLP approach to Arrow's theorem (Indo, 2007)

- A SWF of two-individual and three-candidates can be modeled as a simple recursive program.
 - The impossibility result (its equivalent that the conjunction of U,T,P,I A implies D) can be proved automatically within a second on my PC.
 - But the time of proving depends on the sequence of the list of all the profiles, and there are some lists which can prove within 0.5 sec.

K. Indo. Proving Arrow's theorem by PROLOG. Computational Economics, Vol. 30, No. 1, 2007. (doi: 10.1007/s10614-007-9086-2)

SCLP application of SWF for linear, 2x3 model

□ The complete program based on Indo(2007):

```
/****** preference ordering and profile *****/
lx([a,b,c]). % list of all candidates
x(X):- lx(A), member(X,A). % a candidate
r(R):- lx(A), permutation(A,R). % ranking
pp([R, Q]):- r(R), r(Q). % a profile
r([X, Y], R):- r(R), append(_,[X|Z],R), member(Y,Z). % preference relation
/****** Arrovian swf and its axiom *****/
swf([], []).
swf([P-S|F], [P|L]):- swf(F, L), r(S), pareto(P-S), iia(P-S, F).
swf(F):- findall(R, pp(R), L), swf(F, L).
agree_on([X, Y], [R, Q]):- x(X), x(Y), r([X,Y],R), r([X,Y],Q).
pareto(P-S):- forall(agree_on(XY, P), r(XY,S)).
br(+, [X,Y], R):- X \= Y, r([X,Y], R). % sign-based relation
br(-, [X,Y], R):- X \= Y, \+ r([X,Y], R).
br(+, [X,X], _). % the reflexivity
iia([R,Q]-S, F):- \+ (x(X),x(Y),XY=[X,Y],br(A,XY,R), br(B,XY,Q), br(C,XY,S),
    member([P,W]-T,F),br(A,XY,P), br(B,XY,W), \+ br(C,XY,T)).
/****** table output *****/
fig(F):- findall(R,r(R),L),r(R),nl,forall(member([R,_]-I,F), (nth1(K,L,I), write(K))),fail.
fig(_).
```

Automated Proof of Arrow's theorem for linear, 2x3 model

```
SWI-Prolog -- d:/Prolog/lpmsc/swf_Oct5.pl
File Edit Settings Run Debug Help
% d:/Prolog/lpmsc/swf_Oct5.pl compiled 0.00 sec, 4,960 bytes
Welcome to SWI-Prolog (Multi-threaded, 32 bits, Version 5.6.52)
Copyright (c) 1990-2008 University of Amsterdam.
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.


For help, use ?- help(Topic). or ?- apropos(Word).

1 ?- swf(F), fig(F), nl, write('--'), fail.

123456
123456
123456
123456
123456
123456
123456
--
111111
222222
333333
444444
555555
666666
--
fail.

2 ?- █
```

This run can be seen as the proof of the impossibility theorem because the only two SWFs are proved to be true, and they are both dictatorial. In words, each aggregated ranking displayed as a number for each in $6 \times 6 = 36$ profiles in a cross table which is same as either the number (of ranking, see right) of row or of the column.



```
?-
findall(R,r(R),L),nth1(K,L
,R),nl,write(K:R),fail.

1:[a, b, c]
2:[b, a, c]
3:[b, c, a]
4:[a, c, b]
5:[c, a, b]
6:[c, b, a]
fail.
```

Voting rule and strategy-proofness

- A voting rule (SCF; social choice function) is a function mapping each profile of individual rankings over candidates to a winning candidate.
 - **Possible Properties of SCF:**
 - A voting rule is **dictatorial** if the winner is always the top candidate of a particular voter (the dictator).
 - A voting rule is **manipulable** if it may give a voter an incentive to misrepresent their preferences.
 - A voting rule is **strategy-proof** if it cannot be manipulable.
-

The Gibbard-Satterthwaite Theorem

Proposition 2 (Gibbard, 1973; Satterthwaite, 1975) Under T and U, for more than two candidates, every voting rule is either dictatorial or manipulable. And this result and Proposition 1 are equivalent.

A. Gibbard. Manipulation of Voting Schemes. *Econometrica*, 1973.

M.A. Satterthwaite. Strategy-proofness and Arrow's Conditions. *JET*, 1975.

SCLP application of strategy-proof voting rule for linear, 2x3 model

□ A version of Indo (2008):

```
/****** preference modeling (same as SWF) *****/
lx([a,b,c]). % list of all candidates
x(X):- lx(A), member(X,A). % a candidate
r(R):- lx(A), permutation(A,R). % ranking
pp([R, Q]):- r(R), r(Q). % a profile
r([X, Y], R):- r(R), append(_,[X|Z],R), member(Y,Z). % preference relation
/****** Strategy-proof scf (voting procedure) *****/
scf([], []).
scf([P-Z|F], [P|L]):- scf(F, L), x(Z), \+ manipulable(P-Z, F).
scf(F):- findall(R, pp(R), L), scf(F, L), cs(F).
cs(F):- \+ (x(X), \+ member(_-X, F)).
manipulable([R,Q]-S, F):- member([P,Q]-T,F), (r([T,S],R); r([S,T],P)).
manipulable([R,Q]-S, F):- member([R,W]-T,F), (r([T,S],Q); r([S,T],W)).
/****** table output *****/
fig(scf,F):- r(R),nl,forall(member([R,_]-X,F), write(X)),fail; true.
```

proof

```
?- scf(F), fig(scf,F), nl,
write('**'), fail.
```

```
abbacc
abbacc
abbacc
abbacc
abbacc
**
aaaaaa
bbbbbb
bbbbbb
aaaaaa
cccccc
cccccc
**
fail.
```



K. Indo. Modeling a small agent society based on the social choice logic programming. In T. Terano et al. (eds.), Agent-based Approaches in Economic and Social Complex Systems V: Post-proceedings of the Aescs International Workshop 2007, Springer, 2008

Additionally, some other classical theorems

- The following two results says, No Latin-square profiles (free-triples) is the possible domain for Arrovian SWF and nondictatorial strategy-proof SCF exist.

Proposition 3 (Sen, 1966) Under T , and the value restricted rankings, the aggregation using pairwise majority satisfies all conditions in Proposition 1.

Proposition 4 (Kalai and Muller, 1979) Under T , and the common admissible domain, there exists nondictatorial SWF (and nondictatorial strategy-proof SCF) if and only if the domain is decomposable.

A.K. Sen. A possibility theorem on majority decisions. *Econometrica*, 1966.

E. Kalai and E. Muller. Characterization of domains admitting nondictatorial social welfare functions and nonmanipulable voting procedures. *JET*, 1977.

A finding: minimal profiles to prove impossibility

- However, we can find the minimal set of profiles, by carefully reading Arrow's original proof.

Proposition 5 For two individuals and three alternatives, under conditions T and U, there exists a set of profiles which is minimal in that while it can prove the impossibility, i.e., a single individual is decisive for all pair of alternatives, either any subset of it or the complement set cannot.

Proof. Let $M = \{1, 2, 3, 4, 5, 6\}$, where $k \in M$ represents the index of k -th ranking, $r(1) = acb$, $r(2) = abc$, $r(3) = bac$, $r(4) = bca$, $r(5) = cba$, $r(6) = cab$. Let C denote a set of profiles, $C = \{(r(k), r(j)) \mid (k, j) = (1, 5), (2, 6), (3, 1), (4, 2), (5, 3), (6, 4)\}$. That is, these are profiles of Has-A-Single-Disagree-Pair (HASDP).

(next slide)

- Partitions $\{1, 3, 5\}$ and $\{2, 4, 6\}$ are Latin squares!

A finding: minimal profiles to prove impossibility

- Proof (continued) by using SCLP. It needs only five additional lines to the SWF program as follows:

```
cvr([ 1:[a,c,b], 2:[a,b,c], 3:[b,a,c], 4:[b,c,a], 5:[c,b,a], 6:[c,a,b] ]).  
rc(K,R):- cvr(L), member(K:R,L).  
jv([5,6,1,2,3,4]).  
ppc([K,J],[R,Q]):- jv(I),nth1(K,I,J),rc(K,R),rc(J,Q).  
m(C):- findall(P,ppc(_,P),C).
```

- And the automated proof, the PROLOG system's output after prompt ?- shows the only two dictatorial SWFs restricted to that profiles are satisfied all conditions except (-D).

```
?- m(C),swf(F,C),nl,forall(member([P,Q]-S,F),  
(rc(K,R),rc(J,Q),rc(W,S),write([K,J]-W:' '))),fail.
```

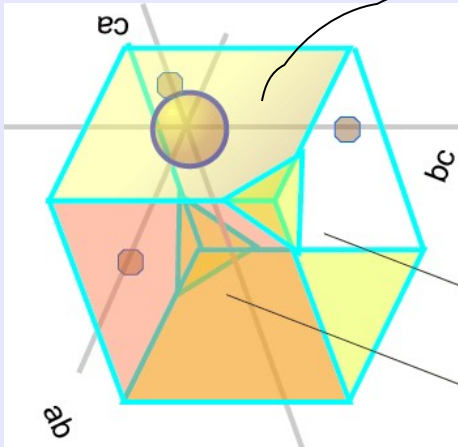
```
[1, 5]-5: [1, 6]-6: [1, 1]-1: [1, 2]-2: [1, 3]-3: [1, 4]-4:  
[1, 5]-1: [1, 6]-2: [1, 1]-3: [1, 2]-4: [1, 3]-5: [1, 4]-6:  
fail.
```

Appendix. Graphical Representation of SWF

- K. Indo. Graphical representation of Arrow's theorem. http://www.xkindo.net/cog_dec/wp/arrow_theorem
- K. Indo. A cube representation of social welfare function. http://www.xkindo.net/cog_dec/wp/cube_swf

Pareto condition (unanimity)
for any pair (x, y)

1 \ 2	>	<
>	>	
<		<



Profiles and the transitivity

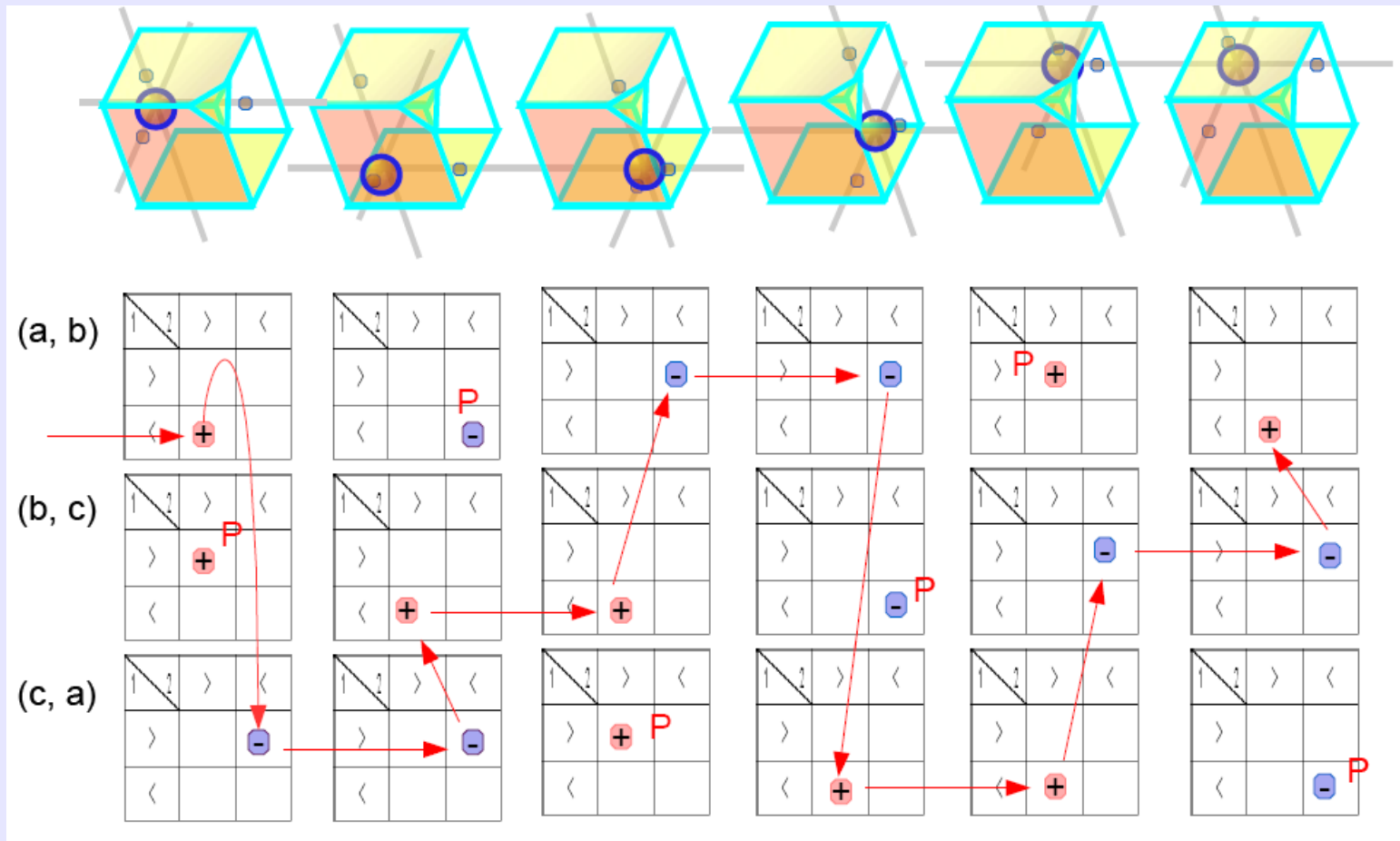
	(a, b)	(b, c)	(c, a)
1 \ 2	>	<	<
>		>	
<	😊	✘	😊

😊 ∈ { >, < }

Which the cell, for (b, c), can you assign me to without violating the transitivity?

An aggregated ranking is transitive *iff* the three smiles don't have a same direction.

Propagating transitivity and decisiveness



Summary, and Future Works

- **Social Choice Logic Programming (SCLP)**
 - Logic Programming and PROLOG
 - Social Choice Theory
 - The SCLP approach combines the above two

 - Beyond classical theory and show some extended results
 - Characterizing the SWFs under the complement of the minimal profiles
 - Relating them to the Strategy-Proof voting procedures
 - Relating them to the (generalized) Single-Peakedness
 - Relating them to the Without-Pareto condition
 - Economic environment
 - Mechanism Design (with Game Theory)
-

Thank you very much for
your kind attention!
