Profile Sequence Formation in the Social Choice Logic Programming

Kenryo Indo

Abstract

The SCLP (Social Choice Logic Programming) approach aims to model the axioms of collective decision making and to prove the classical results on preference aggregation computationally, such as Arrow's social welfare function and its variants, by using PROLOG, albeit for the smallest case. This paper aims to extend and improve the SCLP approach in two directions. First, the Cube-First Method (CFM) is proposed to improve the computational efficiency. By using the minimal set of profiles which is sufficient to prove the dictatorial result earlier in the recursion over the set of all profiles, the computation is prominently accelerated. Second, the Profile Sequence Formation (PSF) based on decisiveness-strengthen connection is argued and it is shown that it can partly simulate acceleration under the CFM.

1 Introduction

K. J. Arrow [1]'s general possibility theorem tells us that only dictatorial aggregation is possible when transitivity (or linear), weak Pareto optimality, and IIA are simultaneously satisfied on the aggregation of individuals' orderings (i.e., the social welfare function).

Practically it is difficult to verify for every logically possible combination of the individual preference orderings (i.e., profile) and the aggregated ordering. In other words, Arrow's proof shows this complexity is superficial and it can be reduced to the set of a single individual's orderings if we assumes a few number of axioms each of them seems no to violate democracy, at least, intuitively.

The SCLP (Social Choice Logic Programming) [2, 3, 4] is an alternative computational modeling and simulation method which verifies or substitutes the traditional axiomatic modeling of the social choice theory. For example, Arrow [1]'s social welfare function and its variants can be proved easily, however, only for the smallest case, i.e., two agents and three alternatives, within a second (see [2]).

This approach also has been generalized to prove the variants of the theorem [5, 6], to derive the strategy-proof voting procedures, to handle other types of orderings and domain conditions, and to recast them in game theory (see [3, 4]). For example, in order to prove Arrow's dictatorial result, both the permissible individual orderings (and their profile) and the aggregated ordering are linear (`l`) or transitive (`t`), satisfying IIA and Pareto conditions. Oligarchy theorem (see Sen [6]) is a variant of it where quasi-transitive(`q`)-valued function is used alternatively.

Whereas it is possible to extend a SCLP model so as to include more than three agent and more than four alternatives, the computation might be awkward. For example, proving Arrow's theorem even on the transitive domain of three agents and three alternatives might take a day or more on your PC (see Table 1).

This is because of the implementation of social welfare functions by using a naive recursion over the set of possible ordering profiles (see Section 2).

This paper aims to extend and improve the SCLP approach in two directions. First, the Cube-First Method (CFM), which uses the Cube Representation [4] in order to realign the list of all possible profiles as the second argument in `swf3`, is proposed. The cube is a minimal set of profiles of individual orderings, which is sufficient to prove the Arrow's

Table 1: The time consumption to prove the Arrow-Sen-type results of ordering aggregation by using standard recursion (*memory allocation error after around 22 hours and two dictatorial SWFs are generated).

agents	alternatives	domain	value	conditions	resulting rule	time (rough)
two	three	`l`	`l`	`iia, p`	dictatorial	0.8 sec
two	three	`l`	`q`	`iia, p`	oligarchical	4 sec
two	three	`t`	`t`	`iia, p`	dictatorial	20 sec
two	three	`t`	`q`	`iia, p`	oligarchical	70 sec
three	three	`l`	`l`	`iia, p`	dictatorial	3260 sec
three	three	`l`	`q`	`iia, p`	oligarchical	6900 sec
three	three	`t`	`t`	`iia, p`	dictatorial	a day*
three	three	`t`	`q`	`iia, p`	oligarchical	unknown

dictatorial result.

In both linear and transitive domain, using a subset of profiles which consists of the subsidiary cubes consists of all two-agent subgroup unanimous profiles earlier turns out to be an effective way.

Second, the Profile Sequence Formation (PSF), which outputs the list of profiles as the domain of the aggregation function and on which the SCLP recursive computing works, is tested experimentally. And this also is a social-oriented computing where each profile is considered as a virtual agent and their coalition formation based on decisiveness-strengthen connection is simulated.

Similar to the CFM acceleration, in fact, at a point of departure, the PSF selects a pair of an element in the cube and the counterpart within its complementary cube is connected, it nevertheless will be grown up to a complete (or maximal, at least) list of profiles irrelevant to the order of the remaining elements in the cube.

These method both can reduce the time the system spends over computing all social welfare functions for above mentioned case in twenty minutes. But the latter is considered more favorable from the viewpoint of parallel computing.

The remaining part of this paper is organized as follows: Section 2 introduces the original SCLP approach. In Section 3 we will extend the original approach to faster computation by using the Cube-First Method. The process of Profile Sequence Formation is simulated and compared to the method in Section 4. Lastly, Section 5 concludes with some remarks.

2 the SCLP approach

The SCLP approach models and simulates the logical conditions (i.e., axioms) of the preference modeling based on several types of binary relations on the set of alternatives (i.e., the domain type); for example, linear ordering (`l`), transitive ordering (`t`), quasi-transitive ordering (`q`), acyclic relation (`a`), and complete and reflexive relation (`o`).

Definition 1 *(SCLP Model). A conceptual model in the SCLP approach, or simply a model, is a four-tuple (m, n, r, v), where m: a number of alternatives (or states of society), n: the number of individual agents, r: the domain, i.e., which type of relation is permissible to individual preference, and v: the type of relation as the aggregated preference, or the aggregation rule per se.*

For the simplicity of analysis, the number of alternatives (or states) is assumed to be three throughout this paper. Let us denote a model, therefore, a three-tuple (n, r, v).

```
% domain_type/1: To inspect which ordering type is currently used.
?- domain_type(I).
I = l:linear
Yes
% chdom/1: To change the domain type.
?- chdom(I).
I = l:linear->t:transitive
Yes
% model/1: To inspect the alternatives and agents.
?- model(A,B).
A = states:[a, b, c]
B = agents:[1, 2]
Yes
% make_n_agents/1: to increase or decrease the number of agents.
?- make_n_agents(3).
Yes.
?- agents(N).
N = [1, 2, 3]
Yes
% chdpm/1: To change the sign-based representation.
?- chdpm(I).
I = 1->2
Yes
```

Figure 1: Selecting the model and representation in the SCLP system: domain type, the number of agents, the sign-based representation.

As the default, the SCLP system uses two-agent domain of linear ordering, and the same valued aggregation function, namely, ($n = 2$, $r =$l, $v =$1).

As for the computational counterpart of the SCLP modeling, the user can alter the current domain type (i.e., which ordering `r/1` is used to construct each profile), the number of agents, and the sign-based representation of the relations, by using `chdom`, `make_n_agents` and `chdpm`, respectively. See Figure 1.

And the rules of collective decision making rules, such as weak Pareto optimality (i.e., unanimity), IIA (Independence of Irrelevant Alternatives), monotonicity, strategy-proofness, and so on naturally.

```
pareto_rule(w,RR->Q):- \+ (dop(XY),agree(s,XY,RR),\+ p(XY,Q)).
iia(RR->R,F):- \+ (member(QQ->Q,F),dop(XY),
  is_same_profile_for_dop(XY,RR,QQ),opposite(_,XY,[R,Q])).
```

Each ordering profile (`rr`) and the collective decision rules defined over the set of profiles (`swfs`) are coded into PROLOG clauses in standard manner. And the above mentioned axioms are accumulated along the profile recursion.

```
rr([],[]).
rr(QQ):- model(_,_:Agents),rr(QQ,Agents).
rr([R|Q],[_|N]):- rr(Q,N),r(R).
all_rr(L):- findall(QQ,rr(QQ),L).
```

The PROLOG model of those aggregation rules which satisfy IIA and Pareto condition in the SCLP approach is as follows.

```
swf([],[],_).
swf([RR->Q|F],[RR|L],X):- swf(F,L,X),axiom_swf(X,RR->Q,F).
swf(F,X):- all_rr(L),swf(F,L,X).
axiom_swf(iia,RR->Q,F):- r(Q),iia(RR->Q,F).
axiom_swf(pareto,RR->Q,_):- r(Q), pareto(w,RR->Q).
axiom_swf(arrow,RR->Q,F):- r(Q),t(Q),pareto(w,RR->Q),iia(RR->Q,F).
axiom_swf(sen,RR->Q,F):- q(Q),pareto(w,RR->Q),iia(RR->Q,F).
```

Then PROLOG is also used to simulate the theorems of these axiomatic agent society based on resolution principle a basic automated theorem proving. It can produce all the logically possible social welfare function(SWF)s given a domain-and-values.

Figure 2 demonstrates the automated proof by PROLOG for a sort of the Arrow-Sen-type aggregation on the domain where linear ordering for each individual agent and quasi-transitive for the aggregated ordering is permissible respectively.

Note that in the literature the word SWF means that of the transitive-valued (i.e., Arrow-type) aggregated ordering defined over the set of the transitive (or linear) individual ordering profiles. However, in this paper it is intended that `swf` denotes a code of any type of aggregation function over any type of domain and value which are specified by the first argument.

Further, the SCLP system enables two alternative sign-based representation of binary relations. Under the second mode, `dp_mode`(2), switched by a flip-flop command `chdpm`(I) as shown in Figure 1. However it might be slightly slower than the default mode (the first mode), it benefits the user of SCLP to see that special two relations which are excluded either from `l, t, q`, and `a` because of their cyclicity.

```
% inspect all cyclical relations using r/4.
?- r(A,B,C,c),nl,write(A;B;C),fail.
[+, +, +];[ (a, b): +, (b, c): +, (c, a): +];[a>b, b>c, c>a]
[-, -, -];[ (a, b): -, (b, c): -, (c, a): -];[b>a, c>b, a>c]
No
```

3 the Cube-First Method

In this section we will modify the SCLP approach in the preceding section into a faster computation based on the Cube Representation of social welfare function ([4] contains the slides). The cube as a set of minimal profiles has been used, at least implicitly, in the literature (for example, see the proof by [1, 5]), but such a graphical interpretation is unprecedented as far as I know.

For n=2, the cube consists of the following profiles (dpm = 2, domain = l: linear).

```
?- dp_mode(2),domain_type(l:_).
Yes
?- rr_cube(dp(2,l),K,B),nl,write(K:B),fail.
%    ab bc ca   ab bc ca
1: [[-, +, +], [+, +, -]]
3: [[-, -, +], [-, +, -]]
5: [[+, -, +], [-, +, +]]
7: [[+, -, -], [-, -, +]]
9: [[+, +, -], [+, -, +]]
11: [[-, +, -], [+, -, -]]
No
```

```
?- model(A,B), display_domain.
current domain: CGITVZ
[base domain=l:linear]
A = states:[a, b, c]
B = agents:[1, 2]
Yes
?- stopwatch((swf(F,sen),display_swf_t2(F),fail;true),T).
swf:row  col |CGITVZ ab|-+-+-+ bc|+--++- ca|+++---
--------------------------------------------------
[-, +, +]=C  |CCCCCC  -|------  +|++++++  +|++++++
[+, -, +]=G  |GGGGGG  +|++++++  -|------  +|++++++
[-, -, +]=I  |IIIIII  -|------  -|------  +|++++++
[+, +, -]=T  |TTTTTT  +|++++++  +|++++++  -|------
[-, +, -]=V  |VVVVVV  -|------  +|++++++  -|------
[+, -, -]=Z  |ZZZZZZ  +|++++++  -|------  -|------
swf:row  col |CGITVZ ab|-+-+-+ bc|+--++- ca|+++---
--------------------------------------------------
[-, +, +]=C  |CEFKLO  -|-0-0-0  +|+00++0  +|+++000
[+, -, +]=G  |EGHMOQ  +|0+0+0+  -|0--00-  +|+++000
[-, -, +]=I  |FHIOPR  -|-0-0-0  -|0--00-  +|+++000
[+, +, -]=T  |KMOTUW  +|0+0+0+  +|+00++0  -|000---
[-, +, -]=V  |LOPUVX  -|-0-0-0  +|+00++0  -|000---
[+, -, -]=Z  |OQRWXZ  +|0+0+0+  -|0--00-  -|000---
swf:row  col |CGITVZ ab|-+-+-+ bc|+--++- ca|+++---
--------------------------------------------------
[-, +, +]=C  |CGITVZ  -|-+-+-+  +|+--++-  +|+++---
[+, -, +]=G  |CGITVZ  +|-+-+-+  -|+--++-  +|+++---
[-, -, +]=I  |CGITVZ  -|-+-+-+  -|+--++-  +|+++---
[+, +, -]=T  |CGITVZ  +|-+-+-+  +|+--++-  -|+++---
[-, +, -]=V  |CGITVZ  -|-+-+-+  +|+--++-  -|+++---
[+, -, -]=Z  |CGITVZ  +|-+-+-+  -|+--++-  -|+++---
% time elapsed (sec): 1.172
F = _G157
T = 1.172
Yes
?-
```

Figure 2: Two-agent, three-alternatives, and quasi-transitive-valued aggregated rule (the oligarchy theorem). The above figure shows that PROLOG system has proved that two dictatorial rules, one oligarchical rule, and no other rules can be generated. Each rule has been displayed as a cross table of two agents' orderings, augmented with its binary decomposed sign-patterns for the three directed pairs, (a, b), (b, c), and (c, a). This experimentation used the Cube-First Method. If the original version is used instead, it will take around 4 or 5 seconds.

These profiles corresponds to the six of eight vertices of the cube. They can be depicted as six positions of a ball in the cube [4], where three faces of it perpendicular to X-Y-Z axis, and intersecting at the all-'+' vertex. The ball position projected on each face represents the decomposed binary relation profiles for the ternary pairs the above mentioned. Two special diagonal vertices, all-'+'s and all-'−'s, have to be excluded from the cube because these are cyclical profiles not allowed in either type of domain.

Note that each profile (i.e., vertex) in the cube agrees its adjacent profiles on only one pair. It can also be observed that those profiles are generated by reversing the sign for a pair. The even numbering is preserved for their intermediate positions which will be explained below.

With this cube representation we can see the decisiveness for each pair of alternatives, which is implied by IIA and Pareto conditions, is propagating along the adjacent vertices. And it can be considered that this is possible because of the social welfare function constrained by these conditions for three (or more) individuals is one-dimensional essentially as proved by Arrow.

Computationally, this is done simply by realignment of profiles so as to move the cube profiles into the last position. Let us call this alternative is the Cube-First Method (CFM), where `all_rr`/1 is modified with `all_rr_cube`/1 as follows:

```
all_rr(A):- all_rr_cube(C), findall(D, rr(D), E),
  subtract(E, C, F), append(F, C, A).
all_rr_cube(C):- domain_type(Y:_), dp_mode(I),
   setof(K:X,rr_cube(dp(I,Y),K,X),L), findall(X,member(_:X,L),C).
```

This realignment of profiles can reduce the cost of checking IIA condition. In fact, it is sufficient and minimal in the sense below.

Proposition 1 *For linear two-(or three-)agent domain, the cube profiles are sufficient to derive the dictatorial result of the SWF. It is also minimal in that any proper subset of it cannot derive that result.*

When n is greater than two, by replacing the agent positions, the cube profiles can be extended to a set of $(n-1)$-agent subgroup unanimity profiles, which is the above two-agent cube profiles if the unanimous components of it are reduced. This extended cube representation called the Hyper Plane Cube. The difference set is as follows.

```
?- hyper_plane_cube(C),member(B,C),\+ rr_cube(dp(2,l),K,B),
nl,write(B),fail.
[[+, +, -], [-, +, +]]
[[-, +, -], [-, -, +]]
[[-, +, +], [+, -, +]]
[[-, -, +], [+, -, -]]
[[+, -, +], [+, +, -]]
[[+, -, -], [-, +, -]]
No
```

As for the transitive ordering domain, the intermediate positions are required in addition to them (see [4]). The following query shows the zero profile of total indifference, two adjacent vertices of the cube numbered 1 and 3 with the intermediate profile, numbered 2, between them.

```
?- rr_cube(dp(2,t),K,B), K<4, nl,write(K:X),fail.
0:[[0, 0, 0], [0, 0, 0]]
```

Table 2: The time consumption to prove the Arrow-Sen-type results of ordering aggregation by using the Cube-First Method and `dp_mode`(1).

agents	alternatives	domain	value	conditions	resulting rule	time (rough)
two	three	`l`	`l`	`iia, p`	dictatorial	0.4 sec
two	three	`l`	`q`	`iia, p`	oligarchical	1 sec
two	three	`t`	`t`	`iia, p`	dictatorial	8 sec
two	three	`t`	`q`	`iia, p`	oligarchical	12 sec
three	three	`l`	`l`	`iia, p`	dictatorial	18 sec
three	three	`l`	`q`	`iia, p`	oligarchical	57 sec
three	three	`t`	`t`	`iia, p`	dictatorial	1650 sec
three	three	`t`	`q`	`iia, p`	oligarchical	4200 sec

```
1:[[-, +, +], [+, +, -]]
2:[[-, 0, +], [0, +, -]]
3:[[-, -, +], [-, +, -]]
No
```

Proposition 2 *For the transitive domain, it is needed to include the intermediate positions, in addition to those vertices for linear domain and the pair of total indifference relations in order to prove the similar result to Proposition 1.*

Here again, the SCLP provides us automated proof instead of mathematical proof. Figure 3 shows the Arrow-type aggregation function confined for the cube profiles in the binary form (see Figure 3). And we consider that this proves the first half part of Proposition 2 (and 1). The minimality would be proved by the fact that the following query to fail.

```
?- all_rr_cube(C),nth0(K,C,B),subtract(C,[B],D),swf(H,D,arrow),
\+ (d_pair(XY),decompose_swf(XY,H,G),length(G,N),N<9).
No
```

The automated proof by PROLOG can be prominently accelerated by using the cube profiles at earlier steps in the recursion. See Table 2 for the statistics summary and compare it with Table 1. For example, as shown in Figure 4, three-agent linear ordering domain case the automated proof is demonstrated within less than 20 seconds, whereas the original version took near one hour as we seen in Table 1. However addition of the intermediate positions could not have a clear improvement with respect to the computation time. Therefore we will not use the intermediate positions as default, but reserve it as `cube_mode`(4) and use it by `chcube`/1 if needed.

4 Profile Sequence Formation

CFM in the preceding section apparently seems to be no more than an artificial realignment of the profile list. In this section, however, rather than cutting the cost of computing, we will focus on evolving process of such a list. In words, we will answer the following question experimentally: Is a sequence (optimal) for the list of profiles (`all_rr`) over which the aggregation function defined, and the recursive computing works efficiently, can be evolved under some simple rules? More straightforwardly, is it possible to consider the CFM as the result of the following evolutive process?

Definition 2 *(Profile Sequence Formation). The Profile Sequence Formation (PSF) is an experimental method where each profile is considered as an virtual agent and their coalition*

```
?- model(A,B), display_domain, chcube(_->4).
current domain: CFGHILOQTUVWZ
[base domain=t:transitive]
A = states:[a, b, c]
B = agents:[1, 2]
Yes
?- all_rr_cube(C), stopwatch((swf(F,C,arrow),display_swf_t6(F,s), fail
); true, Time), !,fail.
[0, 0]->[ (a, b):[0], (b, c):[0], (c, a):[0]]
[0, +]->[ (a, b):[0], (b, c):[0], (c, a):[0]]
[0, -]->[ (a, b):[0], (b, c):[0], (c, a):[0]]
[+, 0]->[ (a, b):[+], (b, c):[+], (c, a):[+]]
[+, +]->[ (a, b):[+], (b, c):[+], (c, a):[+]]
[+, -]->[ (a, b):[+], (b, c):[+], (c, a):[+]]
[-, 0]->[ (a, b):[-], (b, c):[-], (c, a):[-]]
[-, +]->[ (a, b):[-], (b, c):[-], (c, a):[-]]
[-, -]->[ (a, b):[-], (b, c):[-], (c, a):[-]]
---end of swf ---
[0, 0]->[ (a, b):[0], (b, c):[0], (c, a):[0]]
[0, +]->[ (a, b):[+], (b, c):[+], (c, a):[+]]
[0, -]->[ (a, b):[-], (b, c):[-], (c, a):[-]]
[+, 0]->[ (a, b):[0], (b, c):[0], (c, a):[0]]
[+, +]->[ (a, b):[+], (b, c):[+], (c, a):[+]]
[+, -]->[ (a, b):[-], (b, c):[-], (c, a):[-]]
[-, 0]->[ (a, b):[0], (b, c):[0], (c, a):[0]]
[-, +]->[ (a, b):[+], (b, c):[+], (c, a):[+]]
[-, -]->[ (a, b):[-], (b, c):[-], (c, a):[-]]
---end of swf ---
% time elapsed (sec): 0.516
No
```

Figure 3: The above PROLOG code provide an automated proof of that the cube profiles are sufficient to derive the dictatorial result of the Arrow-type SWF and it is minimal. In the above figure, the SCLP system proved that two dictatorial SWFs which is transitive and confined to the cube profiles for two-agent domain, shown in binary decomposed form.

```
?- make_n_agents(3).
Yes
?- model(A,B), display_domain.
current domain: CGITVZ
[base domain=l:linear]
A = states:[a, b, c]
B = agents:[1, 2, 3]
Yes
?- chcube(_->1).
Yes
?- stopwatch((swf(F,arrow),display_swf_t6(F,s),fail;true),T).
[+, +, +]->[ (a, b):[+], (a, c):[+], (b, c):[+]]
[+, +, -]->[ (a, b):[+], (a, c):[+], (b, c):[+]]
[+, -, +]->[ (a, b):[+], (a, c):[+], (b, c):[+]]
[+, -, -]->[ (a, b):[+], (a, c):[+], (b, c):[+]]
[-, +, +]->[ (a, b):[-], (a, c):[-], (b, c):[-]]
[-, +, -]->[ (a, b):[-], (a, c):[-], (b, c):[-]]
[-, -, +]->[ (a, b):[-], (a, c):[-], (b, c):[-]]
[-, -, -]->[ (a, b):[-], (a, c):[-], (b, c):[-]]
---end of swf ---
[+, +, +]->[ (a, b):[+], (a, c):[+], (b, c):[+]]
[+, +, -]->[ (a, b):[+], (a, c):[+], (b, c):[+]]
[+, -, +]->[ (a, b):[-], (a, c):[-], (b, c):[-]]
[+, -, -]->[ (a, b):[-], (a, c):[-], (b, c):[-]]
[-, +, +]->[ (a, b):[+], (a, c):[+], (b, c):[+]]
[-, +, -]->[ (a, b):[+], (a, c):[+], (b, c):[+]]
[-, -, +]->[ (a, b):[-], (a, c):[-], (b, c):[-]]
[-, -, -]->[ (a, b):[-], (a, c):[-], (b, c):[-]]
---end of swf ---
[+, +, +]->[ (a, b):[+], (a, c):[+], (b, c):[+]]
[+, +, -]->[ (a, b):[-], (a, c):[-], (b, c):[-]]
[+, -, +]->[ (a, b):[+], (a, c):[+], (b, c):[+]]
[+, -, -]->[ (a, b):[-], (a, c):[-], (b, c):[-]]
[-, +, +]->[ (a, b):[+], (a, c):[+], (b, c):[+]]
[-, +, -]->[ (a, b):[-], (a, c):[-], (b, c):[-]]
[-, -, +]->[ (a, b):[+], (a, c):[+], (b, c):[+]]
[-, -, -]->[ (a, b):[-], (a, c):[-], (b, c):[-]]
---end of swf ---
% time elapsed (sec): 18.484
F = _G16
T = 18.484
Yes
```

Figure 4: Automated proof of the three-agent social welfare function using recursion with the cube-first profile sequence. The left-hand side of an arrow represents the binary pattern of the profiles and the right-hand side the SWF values on each pair.

Table 3: Distribution of the degree of self-(in)decisiveness (sdd) values for each domain.

l: linear	n = 2	n = 3	t: transitive	n = 2	n = 3
sd(1)	6	6	sd(1)	6	6
sd(2)	12	36	sd(3)	42	150
sd(3)	12	72	sd(5)	48	432
sd(6)	6	102	sd(13)	73	1609
total:	36	216	total:	169	2197

```
?- sdd(P,K), sdd(Q,L), pairwise_sdd(P,Q,J),K>J, L>J,
 nl,write(P:K;Q:L->J),fail.
[[+, -, +], [-, +, +]]:3;[[+, -, -], [-, +, -]]:3->2
[[+, +, -], [-, +, +]]:3;[[+, -, -], [-, -, +]]:3->2
[[-, +, +], [+, -, +]]:3;[[-, +, -], [+, -, -]]:3->2
[[+, +, -], [+, -, +]]:3;[[-, +, -], [-, -, +]]:3->2
[[-, +, -], [-, -, +]]:3;[[+, +, -], [+, -, +]]:3->2
[[+, -, -], [-, -, +]]:3;[[+, +, -], [-, +, +]]:3->2
[[-, +, +], [+, +, -]]:3;[[-, -, +], [+, -, -]]:3->2
[[+, -, +], [+, +, -]]:3;[[-, -, +], [-, +, -]]:3->2
[[-, -, +], [-, +, -]]:3;[[+, -, +], [+, +, -]]:3->2
[[+, -, -], [-, +, -]]:3;[[+, -, +], [-, +, +]]:3->2
[[-, -, +], [+, -, -]]:3;[[-, +, +], [+, +, -]]:3->2
[[-, +, -], [+, -, -]]:3;[[-, +, +], [+, -, +]]:3->2
No
?- sdd(P,K),sdd(Q,L),pairwise_sdd(P,Q,J),J=1,K>1,L>1.
No
```

Figure 5: Automated proof of the Proposition 3.

formation (i.e., the clustering of patially defined SWFs) based on decisiveness-strengthen connection is simulated.

For each subset of profiles, the possible patterns of value-assignment, or equally the number of branching, in the partial aggregation function confined to these profiles is called the *degree of self-(in)decisiveness.* Let sdd(P) denotes the degree of self-decisiveness for a coalition, a subset of profiles, P. And the coalition consists of all profiles P such that sdd(P) $= K$ is denoted by sd(K), $K = 1, 2,$ ..., which are experimentaly verified (see Table 3). For example, for every member P of sd(1), P has a sdd value 1, namely there is no room for choice. If P has a sdd value 2, bifurcation is possible.

A coalition of those profiles may be considered mutually beneficial, or *reciprocal* if you will, if both the participants simultaneously can reduce sdd(P). Actually, there are a very limited number of such two-element reciprocal coalitions.

Proposition 3 *There are only pairs of sd(3) profiles for the linear domain, and of sd(5) profiles for the transitive domain, which can reciprocally reduce their indecisiveness.*

Figure 5 shows a small experimentation which proves the above proposition for the two-agent linear domain under dp_mode(2).

The following proposition is the cololary of Proposition 1 and Proposition 2.

Proposition 4 *The sd(3) profiles, and the sd(5) profiles, is sufficient to prove the Arrow-type dictatorial result in the binary decomposed form of the aggregation for linear domain,* *and for transitive domain, respectively.*

Next, we will define the PSF procedures. These procedures also simulate the cube-first acceleration method partially. In fact, in the early step of the PSF selects a pair of vertices, i.e., a single-pair-agreed profiles in the cube accompanied with another vertex in its complementary (virtual) cube. The outline of the common procedure are as follows:

Procedure 1 *(PSF-Base). Based on the above reciprocal pairs, remaining profiles are sequentially connected to one of the previous coalitions and grown up (or merged) to a complete (or maximal) list of profiles.*

Two types of the additional PSF procedure have been experimented according to the SCLP approach.

Procedure 2 *(PSF-FER). In the first procedure takes a surefooted fork. First, merge two disjoint pairs (or coalitions). Repeat this until there is no disjoint partnership. Then a sort of* Satisficing criteria a la Herb Simon, which governs evolving the set of profiles and the partial SWF defined over them, is adopted. That is, at first the reservation level `sdd` value starts from 1 and gradually increase to select a newcomer at the first contact. We call this the First Encounter Rule (FER).

Procedure 3 *(PSF-MCR). The second procedure goes more greedy. While being similar to the FER it starts from the reciprocal pairs as the evolutive bases, it departs from the FER and grows up faster under the Merge and Clustering Rule (MCR). That is to repeat merge process consistently for a pair of parties such that they have non empty intersection for the partial SWF without any contradictory assignment of the SWF values.*

The computational counterpart for the PSF-Base, PSF-FER, PSF-MCR is `psf_base`, `psf_fer`, and `psf_mcr` respectively.

As already mentioned, the time improvement is unclear compared to using the CFM profiles, our concern might be focused on the measurement of how these profile sequence evolve and are differ from the one which CFM computes.

For example, a PSF-FER experiment of the n=2, transitive domain has been evoled up to 169 rofiles at last. And this process is a survival. The resulting profile list can produce (complete) dictatorial SWFs in about 12 seconds. For n=3, linear domain, this PSF-FER brought about similar process and stopped in 216 steps remains a single complete profile list. It takes about 7 seconds to prove the theorem.

As for PSF-MCR, evolution is not complete and many clusters remain during the experimentation. For n=2, transitive domain, starting from the 13 clusters of `sd`(5) members with each partner, two steps to merge these clusters. The first merge step produces 13 new clusters consist of twelve 22-profile and one 8-profile. The result second merge step is diversed from the min =22 to max=102, however the total number of clusters is same as the previous step.

Table 4 summarizes the experimental results on the correlation coefficients across these procedures. The sequence of numbers in the PSF-FER profile list is negatively coorelated (coefficient -0.5 - -0.6) with the naive `all_rr`, and has no correlation (almost 0 coefficient) with the list of the CFM. The performance distribution of the resulting clusters of the PSF-MCR profile list, for n=3, transitive domain, which appropriately expanded into the complete SWF domain, are almost same as the other PSF procedures and the CFM, but the premature list is often better than the last products about one second. The sequence of numbers in the PSF-MCR profile list has almost no correlation with the CFM, and has weak negative coorelation (coefficient -0.25) with the naive `all_rr`.

Table 4: Summary of the PSF experimentation and the correlation coefficients between the profiles generated by different procedures.

`procedure`	model	max length	corr CFM	corr original SCLP
PSF-FER	($n = 2$, `l`, `l`)	36	0.00	-0.54
PSF-FER	($n = 2$, `t`, `t`)	169	0.00	-0.61
PSF-MCR	($n = 2$, `t`, `t`)	102	0.00	-.249

5 Conclusion

As the result of experimenation shown in Table 4 shows, these two method both can bring about profile lists which have the great computational advantage to us to compute SWFs than the original SCLP.

However, their advantage is not clear when they are compared with the CFM introduced in the preceding section. A list resulted in by the PSF profile evolving process is not superior to the CFM, with respect to the computation time as a whole.

This may be caused by using a list in the predicate `swf`. In other words, the PSF can be considered more favorable from the viewpoint of parallel computing in order to prove the Arrow-type thorem that `swf` is always dictatorial (or oligarchical).

Lastly, the experimental environment is as follows: machine: VGN-FS (Sony Corp.); OS: MS Windows XP Home Edition 2002 SP3; processor: Intel Celeron M, 1.40 GHz; memory: 1.99 GB RAM; language: PROLOG (SWI-Prolog Version 5.0.9); program name: [gprf06,cswf08,fcswf0].

References

[1] K. J. Arrow. Social Choice and Individual Values., 2nd edition. Yale University Press, 1963.

[2] K. Indo. Proving Arrow's theorem by PROLOG. *Computational Economics*, 30(1), 57-63, 2007. (doi: 10.1007/s10614-007-9086-2)

[3] K. Indo. Modeling A Small Agent Society Based on Social Choice Logic Programming. In *Proc. The Fifth International Workshop on Agent-based Approaches in Economic and Social Complex Systems (AESCS-2007)*, Waseda University, Tokyo, Japan, 2007.

[4] K. Indo. Logic Programming for Modeling Social Choice. URL: `http://www.xkindo.net/cog_dec/wp/mplsc.html`. Accessed 1 Jun 2008.

[5] E. Kalai, and E. Muller. Characterization of Domains Admitting Nondictatorial Social Welfare Functions and Nonmanipulable Voting Procedures. *Journal of Economic Theory*, 16, 457-469, 1977.

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

Kenryo Indo
Department of Management, Faculty of Economics
Kanto Gakuen University
200 Fujiagu, Ota, Gunma 373-8515, Japan
Email: `kindo@kanto-gakuen.ac.jp`