Computing possible and necessary winners from incomplete partially-ordered preferences

M. S. Pini*, **F. Rossi***, **K. B. Venable***, **T. Walsh**** [1]

1 INTRODUCTION

We consider how to combine the preferences of multiple agents in the presence of incompleteness and incomparability in their preference orderings. An agent's preference ordering may be incomplete because, for example, we are in the middle of eliciting their preferences. It may also contain incomparability since, for example, we might have multiple criteria we wish to optimize.

To combine preferences, we use *social welfare functions*, which map a profile, that is, a sequence of partial orders (one for each agent), to a partial order (the result). For example, the Pareto social welfare function orders A before B iff every agent orders A before B, else if there is some disagreement between agents declares A and B to incomparable.

Since agents' preferences may be incomplete, we need to complete them to perform preference aggregation. In each possible completion, we may obtain different optimal elements (or *winners*). This leads to the idea of *possible winners* (those outcomes which are winners in at least one possible completion) and *necessary winners* (those outcomes which are winners in all possible completions) [5].

Possible and necessary winners are useful in many scenarios including preference elicitation [3]. In fact, elicitation is over when the set of possible winners coincides with that of the necessary winners [4]. In addition, preference elicitation can focus just on the incompleteness concerning those outcomes which are possible and necessary winners. We can ignore completely all other outcomes.

Whilst computing the sets of possible and necessary winners is in general a difficult problem, we identify sufficient conditions where we can obtain the necessary winners and an upper approximation of the set of possible winners in polynomial time. Such conditions concern either the language for stating preferences, or general properties of the preference aggregation function.

2 FROM THE COMBINED RESULT TO WINNERS

We would like to compute efficiently the set of possible and necessary winners, as well as to determine whether a given outcome is a possible or a necessary winner. In general, even if the social welfare function is polynomially computable, incompleteness in the profile may require us to consider an exponential number of completions. As observed in [5], determining the possible winners is in NP, and the necessary winners is in coNP.

We consider a compact representation of all the completions that is polynomial in size. This necessarily throws away information by compacting together results into a single combined result. Given a social welfare function f and a possibly incomplete profile ip, we consider a complete binary graph, whose nodes are the outcomes, and whose arcs are labeled by non-empty subsets of $\{<, >, =, \bowtie\}$, where $\bowtie$ represents incomparability. Label l is on the arc between outcomes A and B if there exists a completion in which A and B are related by l in the result. We call this structure the *combined result* of f and ip and we denote it with $cr(f, ip)$.

We first consider how to compute the possible and necessary winners given the combined result. We will then consider how to compute the combined result.

Consider the arc between an outcome A and an outcome B in the combined result. Then, if this arc has the label $A < B$, A is not a necessary winner, since there is an outcome B which is better than A in some result. If this arc *only* has the label $A < B$, then A is not a possible winner since we must have $A < B$ in all results. Moreover, consider all the arcs between A and every other outcome C. Then, if no such arc has label $A < C$, then A is a necessary winner. Notice, however, that, even if none of the arcs connecting A have just a single label $A < C$, then we cannot be sure that A is a possible winner: A could be better than some outcomes in every completion, but there might be no completion where it is better than all of them. Following the above considerations, it is thus possible to define the following algorithm to compute the necessary winners and a superset of the possible winners in time quadratic in the number of outcomes.

Algorithm 1: Necessary and possible winners

Input: Ω: set of outcomes; f: preference aggregation function; ip: incomplete profile;
Output: P, N: sets of outcomes;
$P \leftarrow \Omega$;
$N \leftarrow \Omega$;
foreach $O \in \Omega$ **do**
  if $\exists\, O' \in \Omega$ *such that* $(O < O') \in$ *cr(f,ip)* **then**
    $N \leftarrow N - O$;
  if $\exists\, O' \in Omega$ *such that* $(O < O') \in$ *cr(f,ip) and* $(OrO') \notin cr(f, ip)$ *for* $r \in \{=, >, \bowtie\}$ **then** $P \leftarrow P - O$;
return P, N;

If NW is the set of necessary winners and PW is the set of possible winners, Algorithm 1 obtains $N = NW$ and $P = PW^*$, which is a superset of the set of possible winners, in time quadratic in the number of outcomes. PW^* can be different from the set of possible winners for two reasons. First, since we considers one arc at a time, we could not able to recognize global inconsistencies due to violation of the transitivity property. Second, we starts from the combined result where we have already thrown away some information.

[1] *: Department of Pure and Applied Mathematics, University of Padova, Italy. Email: {mpini,frossi,kvenable}@math.unipd.it. **: NICTA and UNSW, Sydney, Australia. Email: tw@cse.unsw.edu.au

The first reason for approximation (that is, non-transitivity) can be eliminated. In fact, given an outcome O, we can eliminate $O < O'$ from the label of each arc connecting O in the combined result, and test whether the new structure, which we call the *possibility structure* of outcome O (or $poss(O)$) is consistent with transitivity. This test is equivalent to testing the consistency of a set of branching temporal constraints [2], which is NP-hard. Fortunately, however, there are many classes of branching temporal constraint problems which are tractable [2], that are likely to occur in our setting. For example, one of the tractable classes is defined by restricting the labels to the set $\{<, >, =\}$. That is, we do not permit incomparability (⋈) in the result. Another tractable case is when we use the Pareto social welfare function, since we fall in one of the tractable classes defined in [2].

Unfortunately, the computation of the combined result requires applying the social welfare function to an exponential number of completions.

3 TRACTABLE COMPUTATION OF POSSIBLE AND NECESSARY WINNERS

We identify some properties of preference aggregation functions which allow us to compute an upper approximation to the combined result in polynomial time, assuming that the social welfare function is polynomially computable. This can then be used to compute possible and necessary winners again in polynomial time.

Let us denote the set of labels of an arc between A and B in the combined result as $rel(A, B)$.

The first property we consider is *independence to irrelevant alternatives (IIA)*. A social welfare function is said to be IIA when, for any pair of outcomes A and B, the ordering between A and B in the result depends only on the relation between A and B given by the agents. Many preference aggregation functions are IIA, and this is a desirable property which is related to the notion of fairness in voting theory [1].

Given a function which is IIA, to compute the set $rel(A, B)$, we just need to ask each agent its preference over the pair A and B, and then use f to compute all possible results between A and B. However, if agents have incompleteness between A and B, f has to consider all the possible completions, which is exponential in the number of such agents.

Assume now that f is also *monotonic*. We say that an outcome B improves with respect to another outcome A if the relationship between A and B does not move left along the following sequence: $>, \geq, (\bowtie \text{ or } =), \leq, <$. A social welfare function f is monotonic if, given any two profiles p and p' and any two outcomes A and B, if passing from p to p' B improves with respect to A in one agent i and $p_j = p'_j$ for all $j \neq i$, then in passing from $f(p)$ to $f(p')$ B improves with respect to A.

To compute $rel(A, B)$ under IIA and monotonicity, again, since f is IIA, we just need to consider the agents' preferences over the pair A and B. However, now we don't need to consider all possible completions for all agents with incompleteness between A and B, but just two completions: $A < B$ and $B > A$. Function f will return a result for each of these two completions, say AxB and AyB, where $x, y \in \{<, >, =, \bowtie\}$. Since f is monotonic, the results of all the other completions will necessarily be between x and y in the ordering $>, \geq, (\bowtie \text{ or } =), \leq, <$.

By taking all such relations, we obtain a superset of $rel(A, B)$, that we call $rel^*(A, B)$. In fact, monotonicity of f assures that, if we consider profile $A < B$ and we get a certain result, then considering profiles where A is in a better position w.r.t. B (that is, $A > B$, $A = B$, or $A \bowtie B$), will give an equal or better situation for A in the result.

Notice that we have obtained set $rel^*(A, B)$ in time polynomial in the number of agents as we only needed to consider two completions. Under the IIA and monotonicity assumptions, we can thus obtain in polynomial time a labeled graph similar to the combined result, but with possibly more labels on the arcs. Then, we can apply the same reasoning as in the previous section to this labeled graph. It is important to notice that the additional labels do not change the necessary and possible winners computed by the algorithm. So we can obtain NW and PW^* in polynomial time.

4 PREFERENCE ELICITATION

At each stage in eliciting agents' preferences, there is a set of possible and necessary winners. When $NW = PW$, preference elicitation can be stopped since we have enough information to declare the winners, no matter how the remaining incompleteness is resolved [4]. At the beginning, NW is empty and PW contains all outcomes. As preferences are declared, NW grows and PW shrinks. At each step, an outcome in PW can either pass to NW or become a loser.

In those steps where PW is still larger than NW, we can use these two sets to guide preference elicitation and avoid useless work. In fact, to determine if an outcome $A \in PW - NW$ is a loser or a necessary winner, it is enough to ask agents to declare their preferences over all pairs involving A and another outcome, say B, in PW. In fact, any outcome outside PW is a loser, and thus is dominated by at least one possible winner.

If the preference aggregation function is IIA, then all those pairs (A, B) with a defined preference for all agents can be avoided, since they will not help in determining the status of outcome A. Moreover, IIA allows us to consider just one profile when computing the relations between A and B in the result, and assures that the result is a precise relation, that is, either $<$, or $>$, or $=$, or $\bowtie$. In the worst case, we need to consider all such pairs. To determine all the winners, we thus need to know the relations between A and B for all $A \in PW - NW$ and $B \in PW$.

During preference elicitation, we can also use the consistency test defined in the previous section to test the consistency of the preferences of each agent. In particular, if the agent declares outcomes to be ordered or incomparable, testing the consistency of the agents' preferences is tractable. If the consistency test is successful, we can exploit the information deduced by the consistency enforcement to avoid asking for preferences which are implied by previously elicited ones. If instead we detect inconsistency, then we can help the agent to make their preferences consistent by providing one or more triangles where consistency fails.

REFERENCES

[1] K. J. Arrow, A. K. Sen, and K. Suzumara, *Handbook of Social Choice and Welfare.*, North-Holland, Elsevier, 2002.

[2] M. Broxvall and P. Jonsson, 'Point algebras for temporal reasoning: Algorithms and complexity', *Artifcial Intelligence*, **149(2)**, 179–220, (2003).

[3] L. Chen and P. Pu, 'Survey of preference elicitation methods', Technical Report IC/200467, Swiss Federal Institute of Technology in Lausanne (EPFL), (2004).

[4] V. Conitzer and T. Sandholm, 'Vote elicitation: Complexity and strategy-proofness', in *Proc. AAAI/IAAI 2002*, pp. 392–397, (2002).

[5] K. Konczak and J. Lang, 'Voting procedures with incomplete preferences', in *Proc. IJCAI-05 Multidisciplinary Workshop on Advances in Preference Handling*, (2005).