Uncertainty in preference elicitation and aggregation*

Toby Walsh
NICTA and UNSW
Sydney, Australia
tw@cse.unsw.edu.au

Abstract

Uncertainty arises in preference aggregation in several ways. There may, for example, be uncertainty in the votes or the voting rule. Such uncertainty can introduce computational complexity in determining which candidate or candidates can or must win the election. In this paper, we survey recent work in this area and give some new results. We argue, for example, that the set of possible winners can be computationally harder to compute than the necessary winner. As a second example, we show that, even if the unknown votes are assumed to be single-peaked, it remains computationally hard to compute the possible and necessary winners, or to manipulate the election.

Introduction

A common mechanism for aggregating preferences is to apply a voting rule. Each agent expresses a preference ordering over a set of candidates, and an election is held to compute the winner. Going back to at least the Marquis de Condorcet in 1785, and continuing with Arrow, Sen, Gibbard, Satterthwaite and others from the 1950s onwards, social choice theory has identified fundamental issues that arise in running such elections. For instance, Arrow's famous impossibility theorem shows that there is no fair way to run such an election if we have more than two candidates. More recently, researchers have started to consider computational issues surrounding such elections. For example, it has been shown that it can be computationally difficult to manipulate an election (Bartholdi, Tovey, & Trick 1989; Bartholdi & Orlin 1991).

One important consideration is the impact of uncertainty on voting. One source of uncertainty is in the votes. For example, during preference elicitation, not all voters may have expressed their preferences. Even if all voters have expressed their preferences, a new candidates might be introduced. To deal with such situations, Konczak and Lang considered how to reason about voting when preferences are incompletely specified (Konczak & Lang 2005). For instance, how do we compute if a certain candidate can still win? Can we compute when to stop eliciting preferences? Another source of uncertainty is in the voting rule itself. For example, uncertainty may be deliberately introduced into the voting rule to make manipulation computationally difficult (Conitzer & Sandholm 2002a). There are other forms of uncertainty which we shall not consider here. For example, preferences may be certain but the state of the world uncertain (Gajdos *et al.* 2006). As a second example, we may have a probabilistic model of the user's preference that is used to direct preference elicitation (Boutilier 2002).

In this paper, we consider how uncertainty in votes and voting rules impacts upon the computation of who wins. We will survey some recent work in this area, as well as giving some new results. We will argue, for example, that it can be computationally harder to compute who can win as opposed to who must win. Some of the results described here were first mentioned in (Konczak & Lang 2005; Pini *et al.* 2007; Lang *et al.* 2007). However, all theorems and proofs appear for the first time.

Voting rules

A *profile* is a set of n total orders over m candidates. A *voting rule* is a function mapping a profile onto one candidate, the *winner*. We shall normally assume that any rule takes polynomial time to apply. We let $N(i, j)$ be the number of voters preferring i to j. We shall consider some of the most common voting rules on profiles. To reduce the impact of ties, we will assume an odd number of voters (or total weight of votes). In the case of a tie, the winner is chosen at random between the tied candidates unless otherwise indicated.

Scoring rules: $(w_1, \dots, w_m)$ is a vector of weights, the ith candidate in a total order scores w_i, and the winner is the candidate with highest total score. The *plurality* rule has the weight vector $(1, 0, \dots, 0)$, the *veto* rule has the vector $(1, 1, \dots, 1, 0)$, whilst the *Borda* rule has the vector $(m - 1, m - 2, \dots, 0)$.

Cup: The winner is the result of a series of pairwise majority elections between candidates.

Copeland (aka tournament): The candidate with the highest Copeland score wins. The Copeland score of candidate i is $\sum_{i \neq j}(N(i, j) > \frac{n}{2}) - (N(i, j) < \frac{n}{2})$. The Copeland winner is the candidate that wins the most

*NICTA is funded by the Australian Government's Department of Communications, Information Technology and the Arts, and the Australian Research Council. Thanks to Jerome Lang, Maria Pini, Francesca Rossi and Brent Venable for their contributions.

pairwise elections. In the second order Copeland rule, if there is a tie, the winner is the candidate whose defeated competitors have the largest sum of Copeland scores.

Simpson (aka maximin): The candidate with the highest maximin score wins. The maximin score of candidate i is $\min_{i \neq j} N(i, j)$. The Simpson winner is the candidate whose worst performance in pairwise elections is best.

Plurality with runoff: If one candidate has a majority, they win. Otherwise everyone but the two candidates with the most votes are eliminated and the winner is chosen using the majority rule. A variation of this (used in the French National Assembly) is to eliminate all candidates with less than a given threshold and then run a plurality election.

STV: This rule requires up to $m - 1$ rounds. In each round, the candidate with the least number of voters ranking them first is eliminated until one of the remaining candidates has a majority.

We also consider some other common voting rules in which votes are not give by a total preference ordering but by some sort of score.

Approval: Each voter labels candidates as approved or not. The candidate with the most number of votes wins.

Range: Each voter gives a score from a given range to each candidate. The candidate with the highest total score wins. Approval voting is an example of range voting where only scores of 0 or 1 are allowed.

Cumulative: Voters have a number of points to distribute between candidates. The candidate with the highest total score wins.

All these rules can be easily modified to work with *weighted* votes. A vote of integer weight k can be viewed as k voters who vote identically. Finally, a *Condorcet winner* is a candidate who beats all others in pairwise elections. Not all elections have a Condorcet winner.

Incomplete profiles

The first form of uncertainty we consider is when the votes are incompletely specified. For example, during preference elicitation, we may have only an incomplete profile. An incomplete profile is one in which some of the votes are incomplete orders. To reason about such a situation, Konczak and Lang define the *possible* winners (candidates that are winners in at least one completion of the profile) and the *necessary* winner (a candidate that wins in all completions) (Konczak & Lang 2005). We will let POSSIBLEWINNERS be the problem of deciding if a given set of candidates is the set of possible winners, and NECESSARYWINNER be the problem of deciding if a given candidate is the necessary winner. Similarly POSSIBLECONDORCETWINNERS is the problem of deciding if a set of candidates are Condorcet winners in at least one completion of the profile, and NECESSARYCONDORCETWINNER is the problem of deciding if a candidate is the Condorcet winner in all completions (Konczak & Lang 2005). Konczak and Lang argue that possible and necessary winners can be used to terminate preference elicitation. When the possible winners narrows down and equals the necessary winner, preference elicitation can stop. In (Pini *et al.* 2007), we show that possible and necessary winners can also direct preference elicitation. We need just resolve the ordering between candidates who are possible winners and can ignore candidates who are outside this set.

Weighted votes

As in (Conitzer & Sandholm 2002a), we will consider both weighted and unweighted votes. Although human elections are often unweighted, the addition of weights makes voting schemes more general. Weighted voting systems are used in a number of real-world settings like shareholder meetings, and elected assemblies. Weights are useful in multiagent systems where we have different types of agents. Weights are also interesting from a computational perspective. First, weights can increase computational complexity. For example, computing the possible winners for the Borda rule is polynomial with unweighted votes (Konczak & Lang 2005), but NP-hard with weighted votes (Pini *et al.* 2007). Second, as we argue in detail later, the weighted case informs us about the unweighted case when we have probabilistic information about the votes. For instance, if computing possible winners is NP-hard with weighted votes, then it is NP-hard to compute the probability of winning when there is uncertainty about how the unweighted votes have been cast. A similar argument has been advanced for considering manipulation with weighted votes (Conitzer & Sandholm 2002a).

Computing Possible and Necessary Winners

We first consider the computational complexity of computing possible and necessary winners. Our analysis is along two dimensions: weighted or unweighted votes, and a bounded or unbounded number of candidates.

Unweighted votes

If the number of candidates is bounded, there are only a polynomial number of effectively different votes. We can thus enumerate and evaluate all different votes in polynomial time. Hence computing POSSIBLEWINNERS and NECESSARYWINNER are both polynomial. A similar argument is made to show that manipulation by a coalition is polynomial when the number of candidates is bounded (Conitzer & Sandholm 2002a; Conitzer, Lang, & Sandholm 2003).

Suppose now that the number of candidates is not necessarily bounded. Konczak and Lang observed that, in this case, POSSIBLEWINNERS is in NP and NECESSARYWINNER is in coNP (Konczak & Lang 2005). They also showed that for any scoring rule, POSSIBLEWINNERS and NECESSARYWINNER are polynomial to compute. Finally, they proved that possible and necessary Condorcet winners are polynomial to compute. In (Pini *et al.* 2007), we proved that POSSIBLEWINNERS for STV is NP-complete whilst NECESSARYWINNER is coNP-complete. In fact, we proved it was even NP-hard to approximate the set of possible winner to within some constant factor in size.

We now argue that POSSIBLEWINNERS can be harder to compute than NECESSARYWINNER (assuming $P \neq NP$).

Theorem 1 *Deciding* POSSIBLEWINNERS *for the second order Copeland rule with unweighted votes and an unbounded number of candidates is NP-complete, but* NECESSARYWINNER *is polynomial.*

Proof: Consider CONSTRUCTIVEMANIPULATION, the problem of deciding if a final vote can be cast to make a given candidate win. This is NP-hard for the second order Copeland rule (Bartholdi, Tovey, & Trick 1989). As POSSIBLEWINNERS is a more general problem, it also is NP-hard. On the other hand, consider the following algorithm for computing if a is the necessary winner under the second order Copeland rule. For every other candidate b, we complete the profile so that b is as high as possible, and a is as low as possible in each vote, and other candidates are in any ordering. Then a is the necessary winner iff a wins each of these elections. Since there are at most $m-1$ such completions to consider, this takes polynomial time to compute. $\diamond$

Finally, it is not hard to see that POSSIBLEWINNERS and NECESSARYWINNER are polynomial for the Approval, Range and Cumulative rules.

Weighted votes

In (Pini *et al.* 2007), we argued that for the Borda, Copeland, Simpson and STV rules, computing POSSIBLEWINNERS with weighted votes is NP-hard. This was based on the NP-hardness of constructive manipulation for these rules with weighted votes (Conitzer & Sandholm 2002a). We now give a more precise result where we bound the number of candidates, and also consider the computational complexity of computing the necessary winner when votes are weighted but the number of candidates is bounded.

Theorem 2 *For 3 or more candidates and weighted votes,* POSSIBLEWINNERS *is NP-complete, and* NECESSARYWINNER *is coNP-complete.*

Proof: Immediate from the proofs that both constructive and destructive manipulation of STV and plurality with runoff are NP-hard for 3 or more candidates (Conitzer 2006). $\diamond$

In fact, POSSIBLEWINNERS for weighted votes is NP-complete with 3 or more candidates for Borda, veto, STV and plurality with runoff, and with 4 or more candidates for Copeland and Simpson. Note that POSSIBLEWINNERS is NP-complete for Borda with weighted votes and 3 or more candidates, but polynomial for unweighted votes even with an unbounded number of candidates.

Probabilistic information

Another form of uncertainty is when we have a probability distribution over the votes (Conitzer & Sandholm 2002a). We first show that the weighted case informs us about the unweighted case when we have probabilistic information about the votes. Given a probability distribution over the votes, a number $r \in [0, 1]$ and a candidate, EVALUATION is the problem of deciding if the probability of the candidate winning is strictly greater than r (Conitzer & Sandholm 2002a).

Theorem 3 POSSIBLEWINNERS *is NP-complete for a voting rule on weighted votes implies* EVALUATION *with unweighted votes is also NP-complete.*

Proof: We reduce POSSIBLEWINNERS to EVALUATION. Each agent of weight k is replaced by k agents of weight 1 whose votes are perfectly correlated. We then construct a joint probability distribution over the votes so that each completion is drawn with the correct frequency. We set $r = 0$. EVALUATION then decides POSSIBLEWINNERS. $\diamond$

Dually, if EVALUATION with unweighted votes is polynomial for a particular voting rule then POSSIBLEWINNERS is also polynomial even with weighted votes. Note that these results hold even if we bound the number of candidates. Note also that such implications do not reverse.

Theorem 4 *For the approval rule,* EVALUATION *is NP-complete, but* POSSIBLEWINNERS *on weighted votes is polynomial.*

Proof: Theorem 84 in (Conitzer 2006) shows that EVALUATION for weighted votes is NP-hard for the approval rule. Theorem 85 in (Conitzer 2006) shows EVALUATION for weighted votes is NP-hard implies EVALUATION for unweighted votes is also NP-hard. For POSSIBLEWINNERS, we just assume that all unknown votes are in favour of the candidate. $\diamond$

Single peaked preferences

An interesting special case is when preferences are single peaked. That is, candidates can be placed in a left to right order and a voter's preference for a candidate decreases with distance from their peak. Single peaked preferences are interesting from several perspectives. First, single peaked preferences are likely to occur in a number of domains. For example, if you are buying a house, you might have an optimal price in mind and your preference for a house decreases as the distance from this price increases. Second, single peaked preferences are easy to elicit. For instance, we might simply ask you for your optimal house price. As a second example, Conitzer gives a strategy for eliciting any single peaked preference ordering with a linear number of pairwise ranking questions (Conitzer 2007). Third, single peaked preferences are easy to aggregate. In particular, there is always a Condorcet winner (the *median* candidate who beats all others in pairwise comparisons) (Black 1948).

Suppose we assume that agents' preferences will be single peaked. Does this make it easier to decide if elicitation can be terminated? We might, for example, stop eliciting preferences if a candidate is already guaranteed to be the Condorcet winner. We suppose we know in advance the ordering of the candidates which make agents' preferences single peaked. For instance, if the feature is price in dollars, we might expect preferences to be single peaked over the standard ordering of integers. However, an interesting extension is when this ordering is not known. Given a total ordering of candidates, an incomplete profile, weights for each agent and a candidate, NECESSARYCONDORCETWINNER for single peaked profiles is the problem of deciding if the candidate is the Condorcet winner in all completions of the incomplete profile which are single peaked with respect to the given ordering. POSSIBLECONDORCETWINNERS for single peaked profiles is the problem of deciding if the candidate is the Condorcet winner in at least one completion of the incomplete profile which is single peaked with respect to the given ordering.

Theorem 5 NECESSARYCONDORCETWINNER *and* POSSIBLECONDORCETWINNERS *for single peaked profiles are polynomial to decide.*

Proof: We assume candidates lie on a left-right scale. For each agent, there is a leftmost and rightmost peak candidate consistent with their declared preferences. The profile can be completed so that any candidate between this leftmost and rightmost candidate is the peak of the agent's preference. We take the leftmost peak candidate for each agent and compute the median candidate for this completion. Similarly, we take the rightmost peak candidate for each agent and compute the median candidate for this completion. If the two completions have the same median candidates, this is necessarily the Condorcet winner. A possible Condorcet winner is any candidate between these two median candidates. $\diamond$

Even though we are assuming agents' preferences are single peaked, there are a number of reasons why we might not use the Condorcet winner. First, the Condorcet winner does not take into account the agents' intensity of preferences. Second, we may not know each agent's most preferred outcome. For example, many web search mechanisms permit users to specify just an approved range of prices (e.g. upper and lower bound on price). In such a situation, it might be more appropriate to use approval voting (which is not Condorcet consistent) to aggregate the possibly single peaked preferences. Third, we might prefer voting systems which are not Condorcet consistent but which have other desirable properties. For example, we might want to take into account voters' lower ranked candidates using a method like Borda. Fourth, we might have hard constraints as well as preferences. As a result, the Condorcet winner might be infeasible. We might therefore consider a voting system which gives a total ranking over all candidates so that we can combine it with constraint solving.

Given a total ordering of candidates, an incomplete profile, and a candidate, POSSIBLEWINNERS for single peaked profiles is the problem of deciding if the candidate is the winner of the election in at least one completion of the incomplete profile which is single peaked with respect to the given ordering. NECESSARYWINNER for single peaked profiles is the problem of deciding if the candidate is the winner in all completions of the incomplete profile which are single peaked with respect to the given ordering.

Theorem 6 *For 3 or more candidates and weighted votes,* POSSIBLEWINNERS *for single peaked profiles is NP-complete for STV.*

Proof: Reduction from number partition. Suppose we have a bag of n numbers, $\{a_i\}$ where $\sum_{i=1}^{n} a_i = 2k$. The 3 candidates are a, b and c. Agents' preferences are single peaked when candidates are ordered alphabetically. We construct an incomplete profile as follows. One agent with weight $6k-1$ votes $b > c > a$, a second agent with weight $4k$ votes $a > b > c$, and a third agent also with weight $4k$ votes $c > b > a$. There are n other agents, each with a weight $2a_i$ and unspecified preferences. Suppose there is a perfect partition. Then, we can have $2k$ weight of votes putting a at the peak, and the other $2k$ weight of votes putting c at the peak. In this case, the STV rule eliminates b in the first round (as b has just $6k-1$ weight of votes, and the other two candidates have $6k$), and then elects c. Hence, there is a completion in which b is a winner if there is a perfect partition. Suppose there is not a perfect partition. Then either a, b or c will receive less than $2k$ weight of votes from the final n agents. In the first case, a is eliminated by the first round of STV and b goes on to win. In the second case, either a or c is eliminated by the first round. If a is eliminated, b then wins. If c is eliminated, b also wins. Finally, in the third case, c is eliminated and b wins. Hence c is a winner iff there is a perfect partition. $\diamond$

Theorem 7 *For 3 or more candidates and weighted votes,* NECESSARYWINNER *for single peaked profiles is coNP-complete for STV.*

Proof: We consider the complement problem, is there a single peaked completion in which the candidate does not win? We use the same reduction from number partitioning used in the last proof. There is a single peaked completion in which b is not a winner iff there is a perfect partition. $\diamond$

Note that plurality with runoff for 3 candidates is equivalent to STV. It follows therefore that POSSIBLEWINNERS and NECESSARYWINNER for single peaked profiles are NP-hard for plurality with runoff.

Manipulation

Uncertainty is a factor when we consider how to manipulate a vote. In this situation, computational complexity may be desirable as a barrier to manipulation. For example, both the second order Copeland rule and STV are NP-hard to manipulate if the number of candidates and voters is unbounded (Bartholdi, Tovey, & Trick 1989; Bartholdi & Orlin 1991). In fact, STV is NP-hard to manipulate even if the number of candidates is bounded provided the voters are weighted (Conitzer & Sandholm 2002a). Here, we show that such results continue to hold when votes are single peaked.

Given a subset of single peaked votes, weights for each agent and a candidate, CONSTRUCTIVEMANIPULATION for single peaked profiles is the problem of deciding if the remaining agents can cast single peaked votes so that the candidate wins the election. DESTRUCTIVEMANIPULATION for single peaked profiles is the problem of deciding if the remaining agents can cast single peaked votes so that the candidate does not win the election. Note we assume the ordering of candidates determining single peakedness is given. However, it would also be interesting to consider when this is not fixed in advance.

Theorem 8 *For 3 or more candidates and weighted votes,* CONSTRUCTIVEMANIPULATION *and* DESTRUCTIVEMANIPULATION *for single peaked profiles are NP-hard for STV.*

Proof: We use the same reduction from number partitioning used in the last two proofs. ◇

Again, it immediately follows that CONSTRUCTIVEMANIPULATION and DESTRUCTIVEMANIPULATION for single peaked profiles are NP-hard for plurality with runoff.

Multiple elections

Agents may be expressing preferences over several related issues. For example, suppose agents are deciding their position with respect to three topical issues. For simplicity, we will consider three agents and three binary issues. The issues are: is global warming happening, does this have catastrophic consequences, and should we act now. A position on the three issues can be expressed as a triple. For instance, N Y Y represents that global warming is not happening, global warming would have catastrophic consequences, and that we do need to act now. The agents' preferences over the three issues are as follows:

	Agent1	Agent2	Agent3
1	Y Y Y	Y N N	N Y N
2	Y N N	Y Y Y	N Y Y
3	N Y N	N N N	N N N
4	N N N	N Y N	N N Y
5	Y N Y	Y N Y	Y N Y
6	N Y Y	N Y Y	Y Y Y
7	N N Y	N N Y	Y N N
8	Y Y N	Y Y N	Y Y N

All agents believe that if global warming is happening and this is causing catastrophic consequences, we must act now. Therefore they all place Y Y N last in their preference rankings. As there are an exponential number of outcomes, it may be unrealistic for agents to provide such complete rankings when voting. One possibility is for the agents just to declare their most preferred outcomes. As issues are binary, we can apply the majority rule to each issue. In this case, this gives Y Y N. Unfortunately, this is everyone's least favourite option. Lacy and Niou show that such "paradoxical" results are a consequence of the agents' preferences not being separable (Lacy & Niou 2000). We can define the possible and necessary winners for such an election. Both are polynomial to compute. Even though there can be an exponential number of possible winners, the set of possible winners can always be represented in linear space as the issues are decided independently. For example, if Agent1 and Agent2 have voted, but Agent3 has not, the possible winners are Y* *.

One way around such paradoxical results is to vote sequentially, issue by issue. Suppose the agents vote sincerely (that is, declaring their most preferred option at each point consistent with the current decisions). Here, the agents would decide Y for global warming, then N for catastrophic consequences (as both Agent2 and Agent3 prefer this given Y for global warming), and then N for acting now. Lacy and Niou prove that if agents vote sincerely, such sequential voting will not return an outcome dominated by all others (Lacy & Niou 2000). However, such sequential voting does not necessarily return the Condorcet winner. Here, for example, it does not return the Condorcet winner, Y Y Y. This outcome beats all others in pairwise elections. We can define possible and necessary winners for such sequential voting. However, it is not at all obvious if the possible or necessary winners of such a sequential vote can be computed in polynomial time, nor even if the set of possible winners can be represented in polynomial space.

Related work

To deal with uncertainty in the votes, Konczak and Lang introduced the notions of possible and necessary winners (Konczak & Lang 2005). They proved that for any scoring rule, possible and necessary winners are polynomial to compute, as are possible and necessary Condorcet winners. They also argue that when computing possible/necessary winners is polynomial, so is constructive/destructive manipulation. In (Pini *et al.* 2007), we proved that possible and necessary winners are NP-hard to compute for STV, and NP-hard even to approximate.

Conitzer, Lang and Sandholm have studied the computational complexity of manipulating an election (Conitzer & Sandholm 2002a; Conitzer, Lang, & Sandholm 2003; Conitzer 2006). They proved that constructive manipulation of Borda, veto, STV, plurality with runoff, Copeland and Simpson are all NP-hard for weighted votes with a small (bounded) number of candidates. Similarly, they proved that destructive manipulation of STV and plurality with runoff are NP-hard for weighted votes with a small (bounded) number of candidates. Finally, they proved that deciding when elicitation can be terminated is NP-hard for STV but polynomial for many other rules, whilst deciding which votes to elicit is NP-hard for approval, Borda, Copeland and Simpson (Conitzer & Sandholm 2002b).

Another source of uncertainty can be in the voting rule itself. For instance, we have considered uncertainty in the application of the Cup rule (Lang *et al.* 2007). This rule performs a series of pairwise majority elections between candidates. The order of these elections, the so-called *agenda*, may not be fixed or known to the voters. We can again introduce upper and lower bounds on who can/must win depending on whether the candidate wins in at least one agenda or in all possible agendas.

Conclusion

Uncertainty arises in preference aggregation in several ways. There may, for example, be uncertainty in the votes or the voting rule. Such uncertainty can introduce computational complexity in determining which candidate or candidates can or must win the election. In this paper, we have surveyed some recent work in this area, and given some new results. We argued, for example, that the set of possible winners can be computationally harder to compute than the necessary winner. As a second example, we have shown that, even if the unknown votes are assumed to be single-peaked, it may remain computationally hard to compute the possible and necessary winners, or to manipulate the election.

Other forms of uncertainty have yet to be studied in detail. For instance, there may be uncertainty in the weight vector used by a scoring rule. The Chair may want to manipulate the election by collecting all the votes and then deciding on the weights in such a way that ensures that their preferred candidate wins. As a second example, the Chair may want to hinder strategic voting by not announcing the weights in advance or by choosing weights randomly. Possible and necessary winners can be defined in a similar way as before to reason about such a situation.

Many open questions remain. For example, what is the complexity of manipulating an election by adding, deleting or partitioning candidates, or by adding, deleting or partitioning voters when preferences are guaranteed to be single peaked? What is the complexity of computing possible and necessary winners, or of constructive and destructive manipulation for other voting rules like Borda when profiles are single peaked? Finally, what is the complexity of computing the possible and necessary winners when the candidates (or the voters) can be partitioned?

References

Bartholdi, J., and Orlin, J. 1991. Single transferable vote resists strategic voting. *Social Choice and Welfare* 8(4):341–354.

Bartholdi, J.; Tovey, C.; and Trick, M. 1989. The computational difficulty of manipulating and election. *Social Choice and Welfare* 6(3):227–241.

Black, D. 1948. On the rationale of group decision-making. *Journal of Political Economy* 56(1):23–34.

Boutilier, C. 2002. A POMDP formulation of preference elicitation problems. In *Proceedings of the 18th National Conference on AI*, 239–246. American Association for Artificial Intelligence.

Conitzer, V., and Sandholm, T. 2002a. Complexity of manipulating elections with few candidates. In *Proceedings of the 18th National Conference on AI*. American Association for Artificial Intelligence.

Conitzer, V., and Sandholm, T. 2002b. Vote elicitation: Complexity and strategy-proofness. In *Proceedings of the 18th National Conference on AI*. American Association for Artificial Intelligence.

Conitzer, V.; Lang, J.; and Sandholm, T. 2003. How many candidates are needed to make elections hard to manipulate. In *Proceedings of the Conference on Theoretical aspects of reasoning about Knowledge*.

Conitzer, V. 2006. *Computational Aspects of Preference Aggregation*. Ph.D. Dissertation, Computer Science Department, Carnegie Mellon University.

Conitzer, V. 2007. Eliciting single-peaked preferences using comparison queries. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*.

Gajdos, T.; Hayashi, T.; Tallon, J.; and Vergnaud, J. 2006. On the impossibility of preference aggregation under uncertainty. Working Paper, Centre d'Economie de la Sorbonne, Universite Paris 1-Pantheon-Sorbonne.

Konczak, K., and Lang, J. 2005. Voting procedures with incomplete preferences. In *Proceedings of the IJCAI-2005 workshop on Advances in Preference Handling*.

Lacy, D., and Niou, E. 2000. A problem with referenda. *Journal of Theoretical Politics* 12(1):5–31.

Lang, J.; Pini, M.; Rossi, F.; Venable, B.; and Walsh, T. 2007. Winner determination in sequential majority voting. In *Proceedings of 20th IJCAI*. International Joint Conference on Artificial Intelligence.

Pini, M.; Rossi, F.; Venable, B.; and Walsh, T. 2007. Incompleteness and incomparability in preference aggregation. In *Proceedings of 20th IJCAI*. International Joint Conference on Artificial Intelligence.