Winner Determination in Voting Trees with Incompleteness*

J. Lang
LAMSADE

M. S. Pini
University of Padova

F. Rossi
University of Padova

K. B. Venable
University of Padova

T. Walsh
NICTA and UNSW

Abstract

In multiagent settings where agents have different preferences, preference aggregation can be an important issue. Voting is a general method to aggregate preferences. We consider the use of voting tree rules to aggregate agents' preferences. In a voting tree, decisions are taken by performing a sequence of pairwise comparisons in a binary tree where each comparison is a weighted majority vote among the agents. Incompleteness in the agents' preferences is common in many real-life settings due to privacy issues or an ongoing elicitation process. We study how to determine the winners when preferences may be incomplete, not only for voting tree rules (where the tree is assumed to be fixed), but also for the *Schwartz* rule (in which the winners are the candidates winning for at least one voting tree). In addition, we study how to determine the winners when only balanced trees are allowed. In each setting, we address the complexity of computing necessary (respectively, possible) winners, which are those candidates winning for all completions (respectively, at least one completion) of the incomplete profile. We show that many such winner determination problems are computationally intractable. However, in some cases, the exact complexity remains unknown. Since it is generally computationally difficult to find the exact set of winners for voting trees and the Schwartz rule, we propose several polynomial algorithms that find approximations of the sets of possible and necessary winners, based on the completions of the (incomplete) majority graph built from the incomplete profiles.

1 Introduction

In multiagent settings, agents generally have different preferences, and it can be important to aggregate these preferences, i.e., to select a collectively desirable candidate from a set of candidates. Candidates could be, for example, potential presidents, joint plans, allocations of goods or resources, etc. A general method for aggregating preferences in multi-agent systems, in order to take a collective decision, is to run an election among the different options using a voting rule.

*This paper is a revised and extended version of the conference papers [17, 23].

Eliciting preferences from agents, to be able to run such an election, is a typical problem in multiagent systems, such as combinatorial auctions and voting systems. However, preference elicitation is often difficult, time-consuming and costly. Agents may be unwilling to reveal all their preferences due to privacy reasons, or due to the set of alternatives being large. Fortunately, we can often determine the outcome before all the preferences have been revealed [7]. For example, it may be that one option has so many votes that it will win whatever happens with the remaining votes.

In this paper, we consider the class of voting tree rules (for short, we shall often say *voting trees*, identifying the tree with the associated rule). These are sometimes called "sequential majority voting" or "Cup". In addition, we consider the Schwartz rule, which can be characterized as a voting tree rule with an undefined tree since the Schwartz winners are exactly those who win for at least one voting tree. We also consider fair winners, i.e., those candidates who win when the voting tree is *balanced*. Fairness comes from the fact that both finalist will have faced the same number of competitions, or the same number plus or minus one.

We consider the computation of the above rules when voters' preferences are incomplete, i.e., represented by partial orders on the set of candidates. We focus on the problem of determining if a given candidate is the *necessary winner* (i.e., the candidate who wins for any completion of these partial preferences) and a *possible winner* (i.e., a candidate who wins for some completion of these partial preferences). Determining if a given candidate is the necessary or a possible winner for voting trees and the Schwartz rule are likely to be computationally hard. We show that they are indeed NP-complete when the input consists of weighted votes, and also when we require that the tree is balanced. We therefore introduce an approximation of the possible and necessary winners, where the set of completions of the incomplete preferences is replaced by the completions of the *incomplete majority graph* computed from the incomplete preferences. We discuss the links between possible (respectively, necessary) winners and their approximation from the incomplete majority graph, and we show that these approximations can be computed in polynomial time.

Our results concern several issues in computational social choice: voting trees, voting under incomplete knowledge, and incomplete tournaments. Voting trees have received increasing attention of late in the literature. For example, they have been considered by Trick [27], Procaccia *et al.* [24], Conitzer *et al.* [6, 8], Xia and Conitzer [31], Hazon *et al.* [14], Vu et al. [29], Fischer *et al.* [11], and Vassilevska [28]. The computational aspects of incomplete tournaments have been thoroughly investigated by Brandt *et al.* [4]. Voting under incomplete knowledge has received even more attention [7, 15, 22, 31, 30, 13, 21, 2, 5].

2 Voting trees, the Schwartz rule, and fair winners

We define some basic notions about voting trees. We first focus on unweighted profiles and then extend all the notions to weighted votes.

2.1 Preferences, profiles and majority graphs

We assume that each agent's *preferences* are specified by a strict total order (TO), that is, by an asymmetric, transitive and complete order, on a set of m candidates. The candidates are taken from a set Ω, and they represent the possible options over which agents vote.

Definition 1 (profile) *A* profile *P on Ω is a collection of n strict total orders over Ω, i.e., $P = (P_1, \ldots, P_n)$, where P_i is the preference relation (or* vote*) of agent (also called* voter*) i.*

For the sake of simplicity we assume throughout the paper that the number n of voters is odd. Our results can be extended to the case where n is even, but at the price of some complications that arise with the handling of ties.

Profiles are denoted using the following notation: $((A > B > C); (A > C > B); (C > A > B))$ means that voter 1 prefers A to B and B to C, etc.

Definition 2 (voting rule and correspondence) *A (deterministic) voting rule is a mapping from the set of profiles to the set of candidates. A voting correspondence is a mapping from the set of profiles to the set of nonempty sets of candidates.*

A voting rule computes a single winner. On the other hand, a voting correspondence can compute multiple winners. For the sake of simplicity, we will use the word "rule" even for correspondence where it does not cause any ambiguity.

Given a profile P, the induced majority graph $M(P)$ is defined as follows.

Definition 3 (majority graph) *Let P be a profile. The majority graph induced by P, denoted by $M(P)$, is the graph whose set of vertices is the set of candidates Ω and where, for all $A, B \in \Omega$, there is a directed edge from A to B (denoted by $A >_m^P B$, or by the abbreviated form $A >_m B$) iff a strict majority of voters prefers A to B.*

The majority graph $M(P)$ induced from any profile P is asymmetric, but it is not necessarily transitive. Moreover, since the number of voters is odd, $M(P)$ is complete: for each A and $B \neq A$, either $A >_m B$ or $B >_m A$ holds. Therefore, $M(P)$ is a complete and asymmetric graph, also called a *tournament* on Ω [18].

2.2 Voting trees

Given a set of candidates, a voting tree is a binary tree (also called an *agenda*) with one candidate per leaf, such that each candidate appears on at least one leaf. This assumption can be made without loss of generality: if a candidate does not appear anywhere in the tree then they can be removed from the set of candidates. In addition if every candidate appears on *exactly* one leaf then we say that the voting tree is a *simple voting tree*. Each internal node represents the candidate that wins the pairwise election between the node's children. The winner of each pairwise election is computed by the majority rule, where A beats B iff there is a majority of voters stating $A > B$. The candidate at the root of the agenda is the overall winner. Given a binary tree T, we will denote with $V(T)$ (and sometimes with $r_{V(T)}$) the voting tree rule induced by T, and given a profile P, we will denote by $V(P,T)$ the candidate that, given profile P, wins in tree T. Note that, because the number of voters is odd, ties never happen in the pairwise elections, so that $V(T)$ is a (deterministic) voting rule. We denote by V the class of all voting tree rules.

Example 1 *Let $\Omega = \{A, B, C, D\}$. Consider the agenda T shown in Figure 1 (a), where A first plays against D, the winner, called w_1, plays against C, and then the winner, called w_2, plays against B. The winner, called w_3, is the overall winner. Consider the profile $P = ((A > B > C > D); (D > A > B > C); (C > D > B > A))$. Its induced majority graph is shown in Figure 1 (b). Therefore $V(P,T) = B$.* □

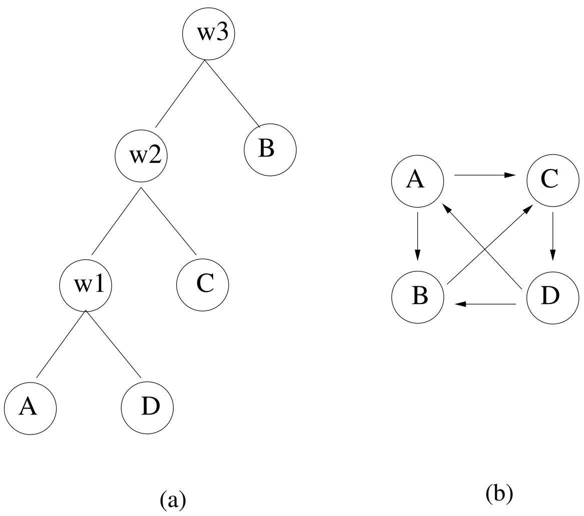

Figure 1: An agenda and a majority graph.

2.3 The Schwartz rule

Given a profile P, a Schwartz winner [25] is a candidate that wins, according to the preferences of P, in some agenda.

Definition 4 (Schwartz winner) *Given a profile P, a candidate A is a Schwartz winner iff there exists a binary tree T such that $V(P,T) = A$.*

The Schwartz rule (more precisely, correspondence) denoted by $r_{Schwartz}$ maps any profile P to the set of all Schwartz winners for P. The set of Schwartz winners coincides with the *top cycle* of the majority graph [20]. The top cycle of a majority graph G is the set of maximal elements of the reflexive and transitive closure G^* of G.

Example 2 *Consider the profile presented in Example 1, i.e.,* $P = ((A > B > C > D); (D > A > B > C); (C > D > B > A))$*. Then the set of Schwartz winners for* P *is* $\{A, B, C, D\}$*.* □

Thus an equivalent characterization of Schwartz winners, that we use later in this paper, is the following:

Theorem 1 (see e.g. [20, 18]) *Given a majority graph* G*, a candidate* A *is a Schwartz winner iff, for every other candidate* C*, there exists a path from* A *to* C*.*

From the characterization above, since finding a path from a single source to all other candidates is polynomial [10], computing the set of the Schwartz winners is polynomial.

Note that voting tree rules, as well as Schwartz, are rules that can be computed from the majority graph. Among the most well-known rules computable from the majority graph, the Schwartz rule is the most liberal. At the other extreme, the Condorcet set contains the Condorcet winner when there exists one, and is empty otherwise; we recall that the Condorcet winner is a candidate who beats every other candidate in the majority graph. In particular, the Schwartz set contains the set of Copeland winners[1], as well as the uncovered set[2], the Banks set [1], and the Slater set[3] [26]. Finally, all these rules are Condorcet-consistent: when a Condorcet winner A exists, all these sets coincide with the singleton $\{A\}$.

2.4 Fair winners

Schwartz winners are candidates who win in at least one binary tree. However, such a tree could be very unbalanced and the winner could compete against only a few candidates. This might be considered "unfair". In the following, we will consider a voting rule *fair* if it has a balanced agenda.

Definition 5 (balanced agenda) *A balanced agenda* T *is a binary tree in which each candidate must appear exactly once in leaves and the difference between the maximum and the minimum depth among the leaves (the level of imbalance, written* $d(T)$*) is less than or equal to 1.*

Assume that we have a profile of strict total orders. We will call fair winners the winners of a balanced agenda.

Definition 6 (fair winner) *Given a profile* P*, a candidate* A *is a fair winner iff there exists a balanced agenda* T*, such that* $V(P, T) = A$*.*

[1] Copeland winners are those candidates who maximize the number of outgoing edges in the majority graph [9].

[2] The uncovered set are those candidates that defeat every other alternative either directly or indirectly at one remove [19].

[3] We do not give formal definitions for Banks and Slater sets since they do not play any further role in the paper.

Example 3 *Consider the profile presented in Example 1, i.e., $P = ((A > B > C > D); (D > A > B > C); (C > D > B > A))$. The set of the fair winners for P is $\{A, D\}$. In fact, we have only three balanced trees, up to isomorphism: in the balanced tree where A plays against B, and C plays against D, the winner is A, while in the other two balanced trees the winner is D.* □

Notice that the Condorcet winner is a fair winner since they win in all trees, and thus also in balanced ones. Moreover, a fair winner is a Schwartz winner since there is a tree, in fact a balanced tree in which they win. Moreover, while there could be no Condorcet winner, there is always at least one fair winner, because there always exist at least one balanced tree. Notice also that the candidates that win in every balanced tree are Condorcet winners.

2.5 Weighted votes

Until now we have considered unweighted votes. However, weighted voting systems are used in a number of real-world settings like shareholder meetings and elected assemblies. An agent with integer weight k can be viewed as k agents voting identically. Although human elections are often unweighted, the addition of weights make voting schemes more general. Weights are useful in multiagent systems where we have different types of agents and some agents are more important than others when taking a decision. Moreover, the weighted case informs us about unweighted case when we have probabilistic information about votes [6]. For example, in [6] Conitzer and Sandholm show that, if coalitional manipulation is hard in the weighted case with complete information about the votes, evaluating a candidate's winning probability is hard in the unweighted case when there is uncertainty about the votes provided we permit perfect correlations between the votes (that is, some votes are uncertain but known to be identical).

We now generalize the notions presented above in this section assuming that each agent may have a weight.

Definition 7 (weighted profile) *A weighted profile is a profile where each agent has a given integer weight.*

Note that the unweighted case considered previously is a special case of weighted case where all the agents' weights are equal to 1.

For simplicity, we assume that the sum of the weights of the agents is odd. The assumption ensures that there cannot be a tied result. This simplification is not essential. However, with an even sum of weights, we would need to specify what to do with tied results.

Definition 8 (corresponding unweighted profile) *Given a weighted profile P, its corresponding unweighted profile $U(P)$ is the profile obtained from P by replacing every ordering, say O, expressed by an agent with weight k, by k agents with weight 1 all with the same ordering O.*

Example 4 *Consider the profile shown in Example 1, i.e.,* $P = ((A > B > C > D); (D > A > B > C); (C > D > B > A))$*. Assume that the first agent has weight* 1*, and that the second and third agents have weight* 2*. Then the corresponding unweighted profile is* $U(P) = ((A > B > C > D); (D > A > B > C); (D > A > B > C); (C > D > B > A); (C > D > B > A))$*.* □

Given a weighted profile P, the induced *majority graph* $M(P)$ is the majority graph of the corresponding unweighted profile of P, i.e., $M(P) = M(U(P))$.

Since the sum of weights is odd, $M(P)$ is complete: for each A and $B \neq A$, either $A >_m B$ or $B >_m A$ holds. Therefore, as for unweighted profiles, $M(P)$ is a tournament.

Example 5 *Consider the weighted profile of Example 4. The majority graph induced by such a profile is shown in Figure 1 (b).* □

Voting trees for weighted profiles work as in the case of unweighted profiles, except that the winner of every pairwise competition is computed by the *weighted* majority rule (and not by the classical majority rule), where A beats B iff there is a weighted majority of votes stating $A > B$.

3 From incomplete profiles to incomplete majority graphs

When votes are only partially known, applying a voting rule may result in some uncertainty about the winner. We first define what we mean by an incomplete profile; then we will make clear what we mean by applying a voting rule to an incomplete preference profile, via the definition of possible and necessary winners. Finally, in the specific case of voting trees, we define an approximation of these two sets of winners, computed directly from the incomplete majority graph induced from the incomplete profile.

3.1 Incomplete profiles, possible and necessary winners

Previously, we supposed agents gave complete preferences over candidates. We now consider the case where agents' preferences may only be partially known.

Definition 9 (incomplete profile) *Let* Ω *be a set of* candidates *and* $\{1, \dots, n\}$ *a set of* voters*. An* incomplete preference relation $>$ *on* Ω *is a strict order on* Ω*, that is, a transitive and irreflexive relation on* Ω*. An (*n*-voter)* incomplete profile *on* Ω *is a collection* $P = (P_1, \dots, P_n)$ *of incomplete preference relations on* Ω*.*

Example 6 *Assume there are three voters and three candidates* A*,* B*, and* C*, then a possible incomplete profile is ((*$A > B > C$*); (*$B > A$*); (*$A > B$*)).* □

This definition naturally extends to weighted voters: a weighted incomplete profile is a weighted collection of incomplete preference relations.

Definition 10 (complete extension (aka completion)) *Let $P = (P_1, \ldots, P_n)$ be an incomplete profile over a set of candidates Ω. A complete extension (or completion) R of P is a tuple $(R_1, \ldots, R_n)$ such that every R_i is a strict total order on Ω containing P_i. We will denote with $Ext(P)$ the set of all complete extensions of P.*

Definition 11 (possible and necessary winners) *Let r be a voting rule, P be an incomplete profile and A a candidate. A is a* possible winner *for P and r, denoted by $A \in Poss(P, r)$, if there exists a completion Q of P such that $r(Q) = A$. A is the* necessary winner *for P and r, denoted by $A \in Nec(P, r)$, if for any completion Q of P we have $r(Q) = A$.*

Possible and necessary winners were introduced in [15]. Xia and Conitzer showed that for most commonly used voting rules computing possible winners is NP-hard, whilst computing necessary winners is coNP-hard for some voting rules but polynomial for others [31].

These definitions apply to any voting rule, and thus to V(T) and Schwartz. For instance, A is a *possible Schwartz winner* for P if it is a possible winner with respect to the incomplete profile P and the Schwartz rule, i.e., $A \in Poss(P, r_{Schwartz})$.

We also define possible and necessary Condorcet winners [15], and possible and necessary fair winners:

Definition 12 (possible and necessary Condorcet winners) *Let P be an incomplete profile and A a candidate, A is a* possible Condorcet winner *for P if there exists a completion Q of P such that A is the Condorcet winner for Q, and the* necessary Condorcet winner *for P if for every completion Q of P, A is the Condorcet winner for Q.*

Definition 13 (possible and necessary fair winners) *Let P be an incomplete profile and A a candidate, A is a* possible fair winner *for P if there exists a completion Q of P such that A is a fair winner for Q, and the* necessary fair winner *for P if for every completion Q of P, A is the fair winner for Q.*

3.2 Winners defined from the majority graph

We now consider an abstraction of the incomplete profile: the incomplete majority graph.

Definition 14 (incomplete majority graph) *Given an incomplete (weighted) profile P with n voters, the incomplete majority graph $M(P)$ induced by P is the graph whose set of vertices is Ω and where there is an edge from A to B, denoted by $A >_m^P B$ (or by the abbreviated form $A >_m B$), if the number of voters (or the sum of the weights of the voters) who prefer A to B is greater than $n/2$ (half of the voters' total weight).*

$M(P)$ is an incomplete asymmetric graph (or *weak tournament*) over Ω.

Example 7 *Consider the incomplete profile shown in Example 6, i.e., (($A > B > C$); ($B > A$); ($A > B$)), and assume that the weight of the first agent is 1 and the weight of*

every other agent is 2*. The majority graph induced by this incomplete weighted profile, that we call Q, is the graph with three nodes A, B, and C and only one edge $A >_m^Q B$.*
$\square$

A *completion* of $M(P)$ is a (complete) tournament containing $M(P)$. Note that the set of all completions of $M(P)$ is a *superset* of the set of majority graphs induced by all possible completions of P. Formally, let $Ext(M(P))$ be the set of all complete extensions of $M(P)$ and $M(Ext(P)) = \{M(R) \mid R \in Ext(P)\}$ the set of all majority graphs induced from the extensions of P. Then, the following result holds.

Proposition 1 *$M(Ext(P)) \subseteq Ext(M(P))$.*

Proof: By Definition of $Ext(M(P))$, we have that $G \in Ext(M(P))$ if and only if for all (A, B) such that $A >_m^P B$, $(A, B) \in G$. Now, let $G \in M(Ext(P))$, which means there is a $Q \in Ext(P)$ such that $G = M(Q)$. Now, $A >_m^P B$ clearly implies $A >_m^Q B$, therefore, $M(Q) \in Ext(M(P))$. $\square$

This inclusion can be strict, which means that moving from P to $M(P)$ implies a loss of information. An incomplete majority graph throws away information about how individual agents have voted. For example, we lose information about correlations between votes. Such correlations may prevent a candidate from being able to win.

Example 8 *Consider an incomplete profile P with just one agent and $\Omega = \{A, B, C\}$, where the only vote is $A > B$ (the relations between A and C, and between B and C have not been specified). The induced majority graph $M(P)$ has only one edge from A to B. In this situation, there is a completion of $M(P)$ (with edges from B to C and from C to A) and an agenda where B wins (we first compare A with C, C wins, and then C with B, and B wins). However, there is no way to complete profile P and set up the agenda so that B wins. In fact, the possible completions of P are $A > B > C$, $A > C > B$, and $C > A > B$, and in all these cases B is always beaten at least by A. Hence, there is no agenda where B wins. Note that the completion of the majority graph which allows us to conclude that B can win, cannot be obtained from any possible completion of the agent's preferences of P since it violates transitivity.* $\square$

Clearly, the more "complete" P is, the more complete $Ext(P)$ is ($M(P)$ can even be complete without P being so) and the better $M(P)$ approximates $M(Ext(P))$.

For any rule r based on the majority graph, including V(T) and Schwartz, we can define notions of the possible (respectively, necessary) winner from the incomplete majority graph, which will approximate the possible and necessary winners (and may be much easier to compute).

Definition 15 (possible/necessary winners from the incomplete majority graph) *Let Q be a complete profile, A a candidate, and r a voting rule based on the majority graph, that is, there is a function f_r such that $r(Q) = f_r(M(Q))$. Let P be an incomplete profile, A is a* possible winner *for $M(P)$ and r if there exists a completion M' of $M(P)$ such that $A = f_r(M')$. A is the* necessary winner *for $M(P)$ and r if for every* *completion M' of $M(P)$ we have $A = f_r(M')$. We denote by $Poss(M(P), r)$ and $Nec(M(P), r)$*[4] *respectively the set of possible and necessary winners for $M(P)$ and r.*

Note that these notions of possible and necessary winners only apply to rules that are based on the majority graph. Note also that $Poss(M(P), r) \supseteq Nec(M(P), r)$, and that when the majority graph is complete, $Poss(M(P), r) = Nec(M(P), r) = r(P)$.

We also define the possible and necessary Condorcet winners from a majority graph in a similar way:

Definition 16 *Let P be a profile and A a candidate, A is a* possible Condorcet winner *for $M(P)$ if there exists a completion M' of $M(P)$ such that A beats every other candidate in M'. A is the* necessary Condorcet winner *for $M(P)$ if for every completion M' of $M(P)$, A beats every other candidate in M'.*

We will denote with $PossCond(P)$ and $NecCond(P)$ (respectively, $PossCond(M(P))$ and $NecCond(M(P))$) the possible and necessary Condorcet winners from the profile P (respectively, from the majority graph $M(P)$).

Example 9 *Consider the profile P given in Example 7. We have that $NecCond(P) = Nec(P, r_{Schwartz}) = \emptyset$, $PossCond(P) = \{A, C\}$, and $Poss(P, r_{Schwartz}) = \{A, B, C\}$. Notice that in this example the possible and necessary Schwartz and Condorcet winners obtained by considering the completions of P coincide with those obtained by considering the completions of the majority graph induced by P. However, as shown in Example 8, this is not true in general.*

3.2.1 Unweighted profiles

Given an incomplete unweighted profile P and the incomplete majority graph G induced by P, that is, $G = M(P)$, we already observed that the completions of G are a (possibly proper) superset of the set of complete majority graphs induced by all possible completions of P. This observation leads to the following results.

Proposition 2 *Let P be an incomplete profile and r be a rule based on the majority graph.*

1. *$Poss(M(P), r) \supseteq Poss(P, r)$.*
2. *$Nec(M(P), r) \subseteq Nec(P, r)$.*
3. *$PossCond(P) = PossCond(M(P))$.*
4. *$NecCond(P) = NecCond(M(P))$*

[4]Note that, given a voting rule r, $Poss(x, r)$ and $Nec(x, r)$ represent two different functions, depending on whether the first argument x is a profile or a majority graph.

Proof: Points 1 and 2 are straightforward consequences of Proposition 1.

For point 3, if a candidate A is a possible Condorcet winner for P then, by Proposition 1, it follows that it is also a possible Condorcet winner for $M(P)$. Also the converse holds. If a candidate A is a possible Condorcet winner for $M(P)$, then there must be one or more completions of the majority graph where A is the Condorcet winner, i.e., where there is an outgoing edge from A to all the other candidates. We now show that there is completion of P in which A is the Condorcet winner. For this we complete P by first adding, to every agent preference relation $>_i$ and every candidate C such that neither $A >_i C$ nor $C >_i A$, the relation $A >_i C$. Let $>'_i$ the relation obtained. First, we claim that $>'_i$ has no cycles. Assume that $>'_i$ has a cycle, then there exists an agent i and a cycle in $>'_i$ involving A (otherwise the cycle would already be in $>_i$, which is not possible because $>_i$ is a strict partial order). Without loss of generality, this cycle is $A >'_i C_1 >'_i C_2 >'_i \ldots >'_i C_m >'_i A$, for some $\{C_1, \ldots, C_m\} \subseteq \Omega \setminus \{A\}$, where $m \geq 1$. Because every C_i is different from A, $C_i > C_{i+1}$ was already in $>_i$, as well as $C_m > A$, that is, $C_1 >_i C_2 >_i \ldots >_i C_m >_i A$. Since $>_i$ is transitive, this implies that $C_1 >_i A$. This contradicts the fact that $A >_i C_1$ was added to $>_i$, because the relation between A and C_1 was *not* unspecified in $>_i$. Now, for every agent i, let $>^*_i$ be the transitive closure of $>'_i$. Because $>'_i$ has no cycles, $>^*_i$ has no cycles and is a partial order. Now, we complete $>^*_i$ by instantiating all remaining unspecified pairs in an arbitrary way that preserves transitivity. Let us denote with P' such a completion of P. According to Proposition 1 we have, $M(P') \in Ext(M(P))$, therefore in $M(P')$ there is an outgoing edge from A to all the other candidates and thus A is the Condorcet winner for P', and therefore a possible Condorcet winner for P.

Now for point 4, the fact that the necessary Condorcet winner for a majority graph $M(P)$ is the necessary Condorcet winner for the profile P is an obvious consequence of Proposition 1. For the converse inclusion, let us suppose A is not the necessary Condorcet winner for $M(P)$. Then there exists B such that $(A, B) \notin M(P)$, that is, in the profile we have $\#\{voter\ i \mid A >_i B\} < \frac{n}{2}$. Let us complete the profile by adding $B >_j A$ for all voters j for which the relation between A and B is unspecified: we get an extension of P in which a majority of voters prefers B to A, therefore A is not the necessary Condorcet winner for P. □

Notice that there are cases in which the subset relation $Poss(M(P), r_{Schwartz}) \supseteq Poss(P, r_{Schwartz})$ is strict. In fact, a candidate can be a Schwartz winner for a completion of $M(P)$ which is not induced by any completion of P, as shown previously in Example 8.

Note that if r is a Condorcet-consistent rule, then $NecCond(P) \subseteq Nec(P, r)$ and $PossCond(P) \subseteq Poss(P, r)$.

3.2.2 Weighted profiles

Proposition 2 can be easily extended to weighted profiles, that is, $Poss(M(P), r) \supseteq Poss(P, r)$, $Nec(M(P), r) \subseteq Nec(P, r)$, $NecCond(M(P)) = NecCond(P)$, and $PossCond(M(P)) = PossCond(P)$. In fact, obviously, $M(P) = M(U(P))$, and it is easy to show that $NecCond(P) = NecCond(U(P))$ and $PossCond(P) = PossCond(U(P))$. We recall that $U(P)$ is the corresponding unweighted profile of P

(see Definition 8).

4 Computing winners with incomplete preferences in voting trees

We now analyze the computational complexity of computing the various kinds of winners in voting trees when preferences are incomplete[5] . We will consider first winners from an incomplete profile and then winners from an incomplete majority graph.

4.1 Incomplete profiles

Assume agents' preferences have been specified by an incomplete weighted profile P. It is possible to show that, when we have 3 or more candidates, it is difficult to determine if a candidate is a possible winner for $V(T)$, even if T is a balanced tree.

Theorem 2 *Given a fixed number n of candidates with $n \geq 3$, an incomplete weighted profile P and any agenda T on these candidates, deciding if a candidate is in $Poss(P, r_{V(T)})$ is NP-complete, even if every candidate appears exactly once in T, and even if T is a balanced tree.*

Proof: We give a reduction from the number partitioning problem. Consider the tree T where A plays against B and the winner thus plays against C. We have a bag of integers, k_i with sum $2k$ and we wish to decide if they can be partitioned into two bags, each with sum k. We will show that we can build an election where we can complete the incomplete weighted profile so that C wins (i.e., C is a possible winner) iff such a partition exists. We suppose the following votes are given: 1 vote for $C > B > A$ of weight 1, 1 vote $C > A > B$ of weight $2k - 1$, and 1 vote $B > C > A$ of weight $2k - 1$. Hence, C is ahead of A by a weight of $4k - 1$, C is ahead of B by a weight of 1 and B is ahead of A by a weight of 1. For each k_i in the bag of integers, we have an incomplete vote of weight $2k_i$ in which $A > C$ is fixed, but the rest of the vote is incomplete. We are sure A beats C in the final result by a weight of 1 whatever completion takes place. We now show that the incomplete weighted profile can be completed so that B beats A and C beats B (and thus C is a possible winner) iff there is a partition of size k.

Assume that such a partition exists, and that votes in one partition have $A > C > B$ and the votes in the other have $B > A > C$. Thus, B beats A overall and C beats B. Thus C is the winner. On the other hand, suppose there is a way to complete the preferences so that C wins. This can only happen if B beats A and C then beats B. In fact, if A beats B in the first round, A will beat C in the second round, since we have shown before that A beats C overall, and then A will be the final winner. For C to beat B, at least half the weight of incomplete votes must rank C above B. Similarly, for B to beat A, at least half the weight of incomplete votes must rank B above A. Since all votes rank A above C, B cannot be both above A and below C. Thus precisely half

[5]For a complete treatment of the basic notions of complexity theory, see [12].

the weight of incomplete votes ranks B above A and half ranks C above B. Hence, we have a partition of equal weight. Therefore, we can complete the incomplete profile so that C wins iff there is a partition of size k. The result can also be extended to the trees with more than 3 candidates which extend T or a tree isomorphic to it, by placing any additional candidate at the bottom of every voters' preference ordering (it does not matter how).

We can use the same construction above when T is a balanced tree. Given the profile constructed here, the only possible balanced trees in which C wins are those in which A plays against B, and the winner then plays against C. All the additional candidates will be defeated by A, B and C, so that they can be placed anywhere in the balanced tree. □

We now show that, when we have 4 or more candidates, it is a coNP-complete problem to determine if a candidate is the necessary winner.

Theorem 3 *Given a fixed number n of candidates with $n \geq 4$, an incomplete weighted profile P and any agenda T on these candidates, deciding if a candidate is in $Nec(P, r_{V(T)})$ is coNP-complete, even if every candidate appears exactly once in T.*

Proof: To show that with more than 4 candidates it is coNP-complete to decide if a candidate wins for every completion of the preferences, we will consider the tree T where A plays against B, the winner then plays against C, and the winner of this match goes forward to the final match against D. We will reduce number partitioning to deciding if, given a particular incomplete weighted profile, we can complete such a profile to make a given candidate win.

We have a bag of integers, k_i with sum $2k$ and we wish to decide if they can be partitioned into two bags, each with sum k. We construct an incomplete profile where the following votes are given: 1 vote for $C > D > B > A$ of weight 1, 1 vote $C > D > A > B$ of weight $2k - 1$, and 1 vote $D > B > C > A$ of weight $2k - 1$. For the first number, k_1 in the bag of integers, we have a vote for $D > B > A > C$ of weight $2k_1$. For each other number, k_i where $i > 1$, we have an incomplete vote of weight $2k_i$ in which $A > C$ is fixed, but the rest of the vote is incomplete. We are sure that A beats C in the final result by 1 vote whatever completion takes place. Similarly, we are also sure that D beats A, and D beats B.

If we complete preferences so that in all incomplete votes D beats C in their pairwise election, then D will win overall. We now show that there is a completion that makes C win iff there is a partition of equal weight.

Suppose there is such a partition and that the complete votes in one partition have $B > A > C$ and the complete votes in the other have $A > C > B$. Thus, B beats A overall, C beats B. We put D in the bottom position in all such completions as far as possible, and so C beats D. Thus C is the overall winner.

On the other hand, suppose there is a way to complete the preferences so that C wins. This can only happen if B beats A, C then beats B and C finally beats D. If A beats B in the first round, A will beat C in the second round and then A will be beaten by D. For C to beat B, at least half the weight of incomplete votes must rank C above B. Similarly, for B to beat A, at least half the weight of incomplete votes must rank

B above A. Since all votes rank A above C, B cannot be both above A and below C. Thus precisely half the weight of incomplete votes ranks B above A and half ranks C above B, and we have a partition of equal weight. Hence, if there is a partition of equal weight then both C and D are possible winners; if not, then only D is a possible winner, therefore the necessary winner. Therefore, D is the necessary winner if and only if there is no partition in the initial problem. The proof can be extended to all the trees with more than 4 candidates, when they extend T or a tree isomorphic to it, by placing any additional candidate at the bottom of every voter's preference ordering (it does not matter how). $\square$

These results can be compared with Theorem 7 in [6] which states that weighted constructive manipulation for $V(T)$ is polynomial. A candidate can be made a winner by a coalition of weighted votes if they are a possible winner for a specific incomplete profile consisting of full votes and empty votes. By allowing any kind of incomplete profile, we increase the computational complexity.

The possible winner (respectively, necessary winner) problem for $V(T)$ was proved to be NP-complete (respectively, coNP-complete) for unweighted votes and an unbounded number of candidates even when T is balanced [32] and easy to compute for unweighted votes and a bounded number of candidates [30].

4.2 Incomplete majority graphs

In this Section (unlike the rest of the paper) we focus on *simple voting trees*, that is, voting trees where every candidate appears exactly once. We present an algorithm, called *Win*, for determining, given an simple voting tree T and an incomplete majority graph G, the set of possible winners (i.e., $Poss(G, r_{V(T)})$). We will represent a simple voting tree T with a binary tree with a root, that we call $root(T)$, a left subtree, called $left(T)$, and a right subtree, called $right(T)$.

Algorithm *Win* recursively takes in input a simple voting tree T, an incomplete majority graph G, and it returns a set of candidates W, which is the set of possible winners. If $root(T)$ is not empty, and both $left(T)$ and $right(T)$ are empty, then the algorithm returns $root(T)$. Otherwise, the set of winners at the root of T is the set of all candidates who are possible winners in the left (respectively, right) branch of T and who beat at least one candidate who is a possible winner in the right (respectively, left) branch of T. If a candidate is a possible winner both for the left and the right branch of T (which may happen because a candidate may appear in more than one terminal node of T), then it belongs to the possible winners for T.

To check whether there exists a necessary winner for an incomplete majority graph G, it suffices to run the algorithm Win on P; if it outputs a single candidate, then it is the necessary winner, otherwise there is no necessary winner. We will call *Necessary-Win* such a procedure.

Example 10 *We now show how to determine possible and necessary winners given a fixed simple voting tree by applying* Win. *Let* $\Omega = \{A, B, C, D, E, F, H, I\}$ *and consider the simple voting tree* T *over* Ω *with* $left(root(T)) = Win(Win(\{C\}, \{D\}), Win(\{E\}, \{F\}))$ *and* $right(root(T)) = Win(Win(\{A\}, \{B\}), Win(\{I\},$

Algorithm 1: *Win*

Input: T: a simple voting tree, G: an incomplete majority graph;
Output: W: set of candidates;
if *T contains only one node* **then**
  $W \leftarrow label(root(T))$
else
  $W_1 \leftarrow Win(left(T), G)$;
  $W_2 \leftarrow Win(right(T), G)$;
  $W \leftarrow \emptyset$;
  foreach $(s,t) \in W_1 \times W_2$ **do**
    if $s >_m t$ **then**
      $W \leftarrow W \cup \{s\}$
    else
      if $t >_m s$ **then**
        $W \leftarrow W \cup \{t\}$
      else
        $W \leftarrow W \cup \{s,t\}$
return W

$\{H\}$)). *Consider also the incomplete majority graph G with edges $A >_m B$, $A >_m C$, $A >_m D$, $A >_m E$, $A >_m I$, $E >_m F$, and $I >_m H$.*

The application of Win to the subtree of T that contains only C and D gives $Win(\{C\},\{D\}) = \{C, D\}$, because G contains no edge between C and D. Its application to the subtree that contains only C and D gives $Win(\{E\},\{F\}) = \{E\}$, because G contains an edge from E to F. Next, the application of Win to the left subtree of T gives $Win(Win(\{C\},\{D\}), Win(\{E\},\{F\}))$ = $Win(\{C, D\},\{E\}) = \{C, D, E\}$, because E is not beaten by both C and D, C is not beaten by E and D is not beaten by E. As for the right subtree of T, we get $Win(Win(\{A\},\{B\}), Win(\{I\},\{H\})) = Win(\{A\},\{I\}) = \{A\}$. Finally, the application of Win to the root of T gives $Win(\{C, D, E\},\{A\}) = \{A\}$, because A beats all of C, D and E in the graph G. Thus, Win *returns $\{A\}$. Since the set returned by* Win *contains only one candidate, A is also the necessary winner.* $\square$

Theorem 4 *Consider a set of candidates Ω, an agenda T over Ω, and a possibly incomplete majority graph G over Ω. Then,*

- *Algorithm* Win(T, G) *returns $Poss(G, r_{V(T)})$.*
- *Algorithm* NecessaryWin(T, G) *returns $Nec(G, r_{V(T)})$.*
- *Both algorithms run in polynomial time in the number of candidates.*

Proof: Let us first consider Algorithm *Win*. We prove by induction that the following induction hypothesis holds for any tree T:

$$H(T)\text{: } Poss(G, r_{V(T)}) = Win(T, G)$$

We omit G in $H(T)$ because the induction is over trees, and not on G.

If T is a tree with a single node, then $H(T)$ holds trivially, because T contains only one candidate, namely $label(root(T))$, and we have $Win(T, G) = Poss(G, r_{V(T)}) = \{label(root(T))\}$. Now we need to show that if T is a tree whose left subtree is T_1 and right subtree is T_2, then $H(T_1)$ and $H(T_2)$ imply $H(T)$. Assume now that $H(T_1)$ and $H(T_2)$ hold.

Assume that $A \in Win(T, G)$. Then either $A \in Win(T_1, G)$ or $A \in Win(T_2, G)$. Without loss of generality, assume $A \in Win(T_1, G)$. Then there exists a B in $Win(T_2, G)$ such that G either contains an edge from A to B or no edge between A and B.

By the induction hypothesis, $A \in Win(T_1, G)$ and $B \in Win(T_2, G)$ imply $A \in Poss(G, r_{V(T_1)})$ and $B \in Poss(G, r_{V(T_2)})$, which in turn imply that there exist two complete tournaments G_1 and G_2 extending G such that $A = r_{V(T_1)}(G_1)$ and $B = r_{V(T_2)}(G_2)$. Now, let G_3 be the tournament obtained from G_1 and G_2 in the following way: for any two candidates X and Y, (a) if both X and Y appear in T_1 then G_3 contains the edge $X >_m Y$ if and only if G_1 contains it; (b) if both X and Y appear in T_2 then G_3 contains the edge $X >_m Y$ if and only if G_2 contains it; (c) G_3 contains the edge $A >_m B$; (d) in all other cases, if G contains an edge $X >_m Y$ then G_3 contains it as well, and if G does not contain any edge between X and Y then we can decide arbitrarily whether there is an edge from X to Y or an edge from Y to X. By construction, G_3 is an extension of G, because both G_1 and G_2 are and because G either contains or does not contain the edge $B >_m A$.

Now, because no candidate appears in more than one leaf of T, the sets of candidates appearing in T_1 and T_2 are disjoint. Therefore no pair of candidates can be concerned by more than one of the cases (a), (b), and (c), and the definition of G_3 is well-founded. Now, because the way $r_{V(T_1)}(G)$ (respectively, $r_{V(T_2)}(G)$) is determined depends only on the restriction of the graph on the candidates appearing in T_1 (respectively, T_2), we have that $r_{V(T_1)}(G_3) = r_{V(T_1)}(G_1) = A$ and $r_{V(T_2)}(G_3) = r_{V(T_2)}(G_2) = B$. From this and the fact that G_3 contains the edge $A >_m B$, we get that $r_{V(T)}(G_3) = A$. Thus we have found a complete extension of G in which the winner of $r_{V(T)}$ is A, which shows that $A \in Poss(G, r_{V(T)})$.

Conversely, assume that $A \in Poss(G, r_{V(T)})$. Then there is a completion G' of G such that $r_{V(T)}(G') = A$. Obviously, A appears in T. Without loss of generality, assume it appears in T_1. Because T_1 is a subtree of T, $r_{V(T)}(G') = A$ implies $r_{V(T_1)}(G') = A$; therefore, since G' is an extension of G, we have $A \in Poss(G, r_{V(T_1)})$, and by the induction hypothesis, we have $A \in Win(T_1, G)$. Now, $r_{V(T)}(G') = A$ implies that A beats $r_{V(T_2)}(G')$ in G'. Denote $r_{V(T_2)}(G')$ by B. Again by the induction hypothesis, we have $B \in Poss(G, r_{V(T_2)})$. Finally, G does not contain the edge $B >_m A$, otherwise G' (which contains the edge $A >_m B$) would not be an extension of G. This implies that $Win(T, G)$ contains A. We now have proven that $H(T)$ holds, and therefore we have shown that the algorithm Win is sound and complete, *i.e.*, returns the set of possible winners for $r_{V(T)}$.

Finally, if we have m candidates, Algorithm *Win* performs at most $O(m)$ steps, where at each step it may consider $O(m^2)$ pairs of candidates. For each pair it performs a constant amount of work. Thus, the overall complexity is $O(m^3)$. The same complexity holds for Algorithm *NecessaryWin*. $\square$

Since $Nec(M(P), r_{V(T)}) \subseteq Nec(P, r_{V(T)})$ and $Poss(M(P), r_{V(T)}) \supseteq Poss(P, r_{V(T)})$, the set returned by Algorithm *Win* is an upper approximation of $Poss(P, r_{V(T)})$.

Notice that, in general, there is not guarantee on the quality of this approximation, as shown by the following example.

Example 11 *Assume we have an incomplete profile, say P, over $n+2$ candidates, say A, $B_1, \ldots, B_n$, and C, where there is only one voter that expresses his preferences as follows: $A > B_i$, $\forall i$. Assume also that the agenda T has A first play against C. For example, T may be the agenda where first A plays against C, the winner plays against B_1, and each new winner at step i plays against B_{i+1}. Then, $Poss(M(P), r_{V(T)}) = \{A, B_1, \ldots, B_n, C\}$, while $Poss(P, r_{V(T)}) = \{A, C\}$. This means that all candidates are possible winners for the incomplete majority graph $M(P)$, while only two of them, i.e., A and C, are possible winners for the incomplete profile P. In fact, a candidate B_i may win only if C beats A and B_i beats C. To achieve this, there must be a completion with a cycle $A > B_i > C > A$, which is allowed in the majority graph, but it is forbidden by transitivity in the profile.* $\square$

Since $Poss(P, r_{V(T)}) \subseteq Poss(M(P), r_{V(T)})$, the quality of the approximation can be measured by the ratio between the number of winners returned by $Poss(M(P), r_{V(T)})$ and $Poss(P, r_{V(T)})$. Example 11 shows that this ratio can be as bad as $\frac{m+2}{2}$.

However, if the number of the missing arcs in the incomplete majority graph is smaller than a certain value, then we are able to give an upper bound to the size of the set of possible winners from the incomplete majority graph.

Proposition 3 *Let P be an incomplete profile over m candidates, $M(P)$ its corresponding incomplete majority graph, and k the number of the missing arcs in $M(P)$. If $k < log_2 m$, then $Poss(M(P), r_{V(T)}) < m$.*

Proof: If there are k missing arcs in $M(P)$, then there are 2^k possible completions of $M(P)$. Therefore, there are at most 2^k results and so there are at most 2^k possible winners. Hence, if $k < log_2 m$, then $2^k < m$, i.e., the number of the possible winners for $M(P)$, given a certain agenda, is smaller than m. $\square$

Parametrizations based on the total number of undetermined candidate pairs have also been considered in [21] to analyze the complexity of the possible winner problem for some voting rules. However, these voting rules do not include the voting tree rule.

5 Computing winners with incomplete preferences in the Schwartz rule

We now analyze the complexity of determining Schwartz winners when we have uncertainty in the preferences.

We show that deciding whether a given candidate is a Schwartz winner from incomplete weighted profiles is intractable in general, but polynomial from the induced majority graphs. Hence, computing the winners for incomplete majority graphs can be a computationally efficient way to have an approximation of the winners for incomplete weighted profiles. We conjecture that computing the set of all the possible Schwartz winners for incomplete unweighted profiles is NP-hard as well, but so far have not been able to show this.

5.1 Incomplete weighted profiles

We prove that determining whether a given candidate is a possible Schwartz winner for weighted profiles is NP-hard for incomplete weighted profiles where there are 3 or more candidates.

Theorem 5 *Given a fixed number n of candidates with $n \geq 3$ and an incomplete weighted profile P on these candidates, deciding if a candidate is in $Poss(P, r_{Schwartz})$ is NP-complete.*

Proof: Clearly the problem is in NP as a polynomial witness is a completion and an agenda in which the candidate wins. To show it is NP-complete, we give a reduction from the number partitioning problem. The reduction is based on constructing a Condorcet cycle and is similar to those used in [6] to show that manipulation is computationally hard even with a small number of candidates when votes are weighted.

We have a bag of integers, k_i with sum $2k$ and we wish to decide if they can be partitioned into two bags, each with sum k. We want to show that a candidate B is a possible Schwartz winner if and only if such a partition exists. We construct an incomplete profile over three candidates (A, B, and C) as follows. We have 1 vote for $B > C > A$ of weight 1, 1 vote $B > A > C$ of weight $2k-1$, and 1 vote $C > B > A$ of weight $2k-1$. At this point, the total weight of votes with $B > A$ exceeds that of $A > B$ by $4k-1$, the total weight of votes with $B > C$ exceeds that of $C > B$ by 1, and the total weight of votes with $C > A$ exceeds that of $A > C$ by 1.

We also have, for each k_i, a partially specified vote of weight $2k_i$ in which we know just that $A > B$. As the total weight of these partially specified votes is $4k$, we are sure A beats B in the final result by 1 vote. The only agenda in which B can win is the one in which A plays against C and the winner then plays against B. In addition, for B to win, the partially specified votes need to be completed so that B beats C, and C beats A in the final result. We show that this is possible iff there is a partition of size k. Suppose there is such a partition. Then let the votes in one bag be $A > B > C$ and the votes in the other be $C > A > B$. Then, A beats B, B beats C and C beats A, all by 1 vote in the final result. On the other hand, suppose there is a way to cast the votes to give the result A beats B, B beats C, and C beats A. All the uncast votes rank A above B. In addition, at least half the weight of votes must rank B above C, and at least half the weight of votes must rank C above A. Since A is above B, C cannot be both above A and below B. Thus precisely half the weight of votes ranks C above A and half ranks B above C. Hence we have a partition of equal weight. We therefore conclude that B can win iff there is a partition of size k. That is, deciding if

B is a possible Schwartz winner is NP-complete. We can extend the reduction to more than 3 candidates by placing any additional candidate at the bottom of every voters' preference ordering in whatever way we wish. □

Computing possible Schwartz winners for weighted profiles seems to be intractable, however we will show that is easy to find an upper approximations of these winners considering the majority graph.

5.2 Incomplete majority graphs

We now give graph characterizations of possible and necessary Schwartz winners for incomplete majority graphs, which allow us to conclude that they can be computed in polynomial time.

Theorem 6 *Given an incomplete majority graph G and a candidate A, $A \in Nec(G, r_{Schwartz})$ if and only if, for every other candidate B, there is a path from A to B in G.*

Proof: ($\Leftarrow$) Suppose that for each $B \neq A$ there is a path from A to B in G. Then these paths remain in every completion of G. Therefore, using Theorem 1, A is a Schwartz winner in every completion of G, i.e., it is the necessary Schwartz winner.
($\Rightarrow$) Suppose there is no path from A to B in G. Let us define the following three subsets of the set of candidates Ω: $R(A)$ is the set of candidates reachable from A in G (including A); $R^{-1}(B)$ is the the set of candidates from which B is reachable in G (including B); and $Others = \Omega \setminus (R(A) \cup R^{-1}(B))$. Because there is no path from A to B in G, we have that $R(A) \cap R^{-1}(B) = \emptyset$ and therefore $\{R(A), R^{-1}(B), Others\}$ is a partition of Ω. Now, let us build the complete majority graph $\hat{G}$ as follows:

1. $\hat{G} := G$;
2. $\forall\, x \in R(A)$, $y \in R^{-1}(B)$, add (y, x) to $\hat{G}$;
3. $\forall\, x \in R(A)$, $y \in Others$ (i.e., $y \in \Omega \setminus (R(A) \cup R^{-1}(B))$), add (y, x) to $\hat{G}$;
4. $\forall\, x \in Others$ (i.e., $x \in \Omega \setminus (R(A) \cup R^{-1}(B))$), $y \in R^{-1}(B)$, add (y, x) to $\hat{G}$;
5. $\forall\, x, y$ belonging to the same element of the partition: if neither (x, y) nor (y, x) is in G then add one of them (arbitrarily) in $\hat{G}$.

Let us first show that $\hat{G}$ is a complete majority graph. If $x \in R(A)$ and $y \in R^{-1}(B)$, then $(x, y) \notin G$ (otherwise there would be a path from A to B in G). If $x \in R(A)$ and $y \in Others$, then $(x, y) \notin G$, otherwise y would be in $R(A)$. If $x \in Others$ and $y \in R^{-1}(B)$, then $(x, y) \notin G$, otherwise x would be in $R^{-1}(B)$. Therefore, whenever x and y belong to two distinct elements of the partition, $\hat{G}$ contains (y, x) or (x, y), but not both. Now, if x and y belong to the same element of the partition, by Step 5, $\hat{G}$ contains exactly one edge in $\{(x, y), (y, x)\}$. Therefore, $\hat{G}$ is a complete majority graph. Let us show now that there is no path from A to B in $\hat{G}$. Suppose there is one, that is, there exist $z_0 = A, z_1, \ldots, z_{m-1}, z_m = B$ such that

$\{(z_0, z_1), (z_1, z_2), \ldots, (z_{m-1}, z_m)\} \subseteq \hat{G}$. Now, for all $x \in R(A)$ and all y such that $(x, y) \in \hat{G}$, by construction of $\hat{G}$, we necessarily have $y \in R(A)$. Therefore, for all $i < m$, if $z_i \in R(A)$ then $z_{i+1} \in R(A)$. Now, since $z_0 = A \in R(A)$, by induction we have $z_i \in R(A)$ for all i, thus $B \in R(A)$, which is impossible. Therefore, there is no path from A to B in $\hat{G}$. Thus, $\hat{G}$ is a complete majority graph with no path from A to B, which implies that A is not a Schwartz winner with respect to $\hat{G}$. Lastly, by construction, $\hat{G}$ contains G. So $\hat{G}$ is a complete extension of G for which A is not a Schwartz winner. This shows that A is not the necessary Schwartz winner for G. □

A procedure based on the previous theorem gives us a polynomial time algorithm to find necessary Schwartz winners. Hence, we have a polynomial time approximation algorithm of necessary Schwartz winners for incomplete profiles.

Corollary 1 *Given an incomplete majority graph G, $Nec(G, r_{Schwartz})$ can be computed in polynomial time.*

Proof: By Theorem 6, we know that $A \in Nec(G, r_{Schwartz})$ if and only if, for every other candidate B, there is a path from A to B in G. Since finding a path from a single source to all other candidates is polynomial [10], then also computing $Nec(G, r_{Schwartz})$ is polynomial. □

We now consider computing possible Schwartz winners. Let G be an asymmetric graph, Ω the set of candidates, and $A \in \Omega$. Let us consider Algorithm 2. It tries to complete G in order to have a path from A to every other candidate Z by putting edges directed to Z, where it is possible, from the candidates that are reachable from A.

Algorithm 2: PossibleSchwartzWinners

```
Input: G: an asymmetric graph, A: a candidate,
Output: Σ: a set of candidates;
Σ ← {A} ∪ {X | there is a path from A to X in G};
G' ← G;
repeat
    foreach (Y, Z) ∈ Σ × (Ω \ Σ) do
        if (Z → Y) ∉ G' then
            add (Y → Z) to G';
    foreach Z ∈ Ω \ Σ do
        if there is a path from A to Z in G' then
            add Z to Σ;
until Σ = Ω or there is no (Y, Z) ∈ Σ × (Ω \ Σ) s. t. (Z → Y) ∉ G' ;
return Σ
```

Theorem 7 *Given an incomplete majority graph G and a candidate A, $A \in Poss(G, r_{Schwartz})$ if and only if* PossibleSchwartzWinners$(G, A) = \Omega$.

Proof: We start with the following observation: the graph G' obtained at the end of the algorithm is asymmetric and extends G. It is asymmetric because it is so at the start of the algorithm (since G is asymmetric) and it is still asymmetric when an edge $Y \rightarrow Z$ is added to G' when $Z \rightarrow Y$ is not already in G'.

Now, assume PossibleSchwartzWinners$(G, A) = \Omega$. Let G'' be a majority graph extending G' (and, a fortiori, G). Such a G'' exists (because G' is asymmetric). By construction of G', there is a path in G' from A to every node of PossibleSchwartzWinners$(G, A) \setminus \{A\}$, hence to every node of $\Omega \setminus \{A\}$; since G'' extends G', this holds a fortiori for G'', hence A is a Schwartz winner in G'' and therefore a possible Schwartz winner for G.

Conversely, assume PossibleSchwartzWinners$(G, A) = \Sigma \neq \Omega$. Denote $\Theta = \Omega \setminus \Sigma$. Then, for all $(Y, Z) \in \Sigma \times \Theta$ we have $Z \rightarrow Y \in G'$. Now, $Z \in \Theta$ means that no edge $Z \rightarrow Y$ (for $Z \in \Theta$ and $Y \in \Sigma$) was added to G'; hence, for every $Y \in \Sigma$ and $Z \in \Theta$, we have that $Z \rightarrow Y \in G'$ if and only if $Z \rightarrow Y \in G$. This implies that for all $(Y, Z) \in \Sigma \times \Theta$ we have $Z \rightarrow Y \in G$, therefore, in every majority graph G'' extending G, every candidate of Θ beats every candidate of Σ, and in particular A. Therefore, there cannot be a path in G'' from A to a candidate in Z, which implies that A is not a Schwartz winner in G''. Since the latter holds for every majority graph G'' extending G, A is not a possible Schwartz winner for G. □

Algorithm 2 is in fact a polynomial-time approximation algorithm for computing possible Schwartz winners for incomplete profiles.

Corollary 2 *Given an incomplete majority graph G, the set $Poss(G, r_{Schwartz})$ can be computed in polynomial time.*

Proof: It follows from Theorem 7 together with the fact that Algorithm PossibleSchwartzWinners (Algorithm 2) runs in polynomial time, since it performs at most m iterations, each iteration considering at most m^2 pairs of candidates. □

By Lemma 2 of [3], the set of the Smith winners for an incomplete tournament G [3] coincides with the set of possible Schwartz winners for G. Therefore, the above polynomiality result is also subsumed by Theorem 5 of [3].

Given an asymmetric graph $G = (\Omega, E)$ and $\Theta \subseteq \Omega$, Θ is said to be a *dominating set* of G if and only if, for every $Z \in \Theta$ and every $X \in \Omega \setminus \Theta$, we have $(Z, X) \in G$. Then we have an alternative characterization of possible Schwartz winners:

Theorem 8 *Given an incomplete majority graph G and a candidate A, $A \in Poss(G, r_{Schwartz})$ if and only if A belongs to all dominating sets of G.*

Proof: Suppose there exists a dominating set Θ of G such that $A \notin \Theta$. Then there is no extension of G in which there is a path from A to any candidate $Z \in \Theta$. Hence A is not a possible Schwartz winner for G.

Conversely, suppose that A is not a possible Schwartz winner for G. Then the algorithm for computing PossibleSchwartzWinners(G, A) stops with PossibleSchwartzWinners $(G, A) \neq \Omega$ and $\Omega \setminus$ PossibleSchwartzWinners(G, A) being a dominating set of G. Since $A \in$ PossibleSchwartzWinners(G, A), $\Omega \setminus$ PossibleSchwartzWinners(G, A)

is a dominating set of G to which A does not belong. □

To conclude, we have shown that it is easy to compute an upper approximation of the possible Schwartz winners from the incomplete profile by considering the majority graph. Regarding the quality of this approximation, similar results can be dervied as in Section 4.2 since Example 11 can be easily generalized to work for every agenda.

6 Computing winners over balanced agendas

We now turn our attention to fair winners, that is, candidates that win in some balanced agendas.

6.1 Complete profiles

For simplicity, we will assume there are 2^k candidates. We will show that a candidate, say A, is a fair winner iff the nodes of the majority graph can be covered by a spanning binomial arborescence [10] with root A. Spanning binomial arborescences are defined inductively as follows:

- A spanning binomial arborescence of order 0, written as T_0, is a tree with only one node.
- A spanning binomial arborescence of order k, with $k > 0$, written T_k, is the tree where the root has k children, and for $i = 1, \ldots, k$ the i-th child is a spanning binomial arborescence of order $k - i$.

It is easy to see that, in a spanning binomial arborescence of order k, there are 2^k nodes and the tree has height k.

Given a majority graph with 2^k nodes and a candidate A, it may be possible to find a covering of the nodes which is a spanning binomial arborescence of order k with root A. In this situation, we can build a balanced binary tree where A wins. Thus A is a fair winner.

Theorem 9 *Given a majority graph G with 2^k nodes and a candidate A, there is a spanning binomial arborescence T_k covering all nodes of G with root A, if and only if A is a fair winner.*

Proof: Assume there is a spanning binomial arborescence T_k satisfying the statement of the theorem. Then each node of T_k is associated to a candidate. For node n, we will write $cand(n)$ to denote such a candidate.

We will show that, starting from T_k, it is possible to define a balanced binary tree $b(T_k)$ where A wins. Each node of $b(T_k)$ is associated with a candidate, with the same notation as above. The definition of $b(T_k)$ is given by induction:

- If $k = 0$, $b(T_0) = T_0$;

- If $k > 0$, $b(T_k)$ is the balanced tree built from two instances of $b(T_{k-1})$, corresponding to the two instances of T_{k-1} which are part of T_k by definition; the roots of such trees are the children of the root of $b(T_k)$; the candidate of the root of $b(T_k)$ is the candidate of the root of T_k.

It is easy to see that $b(T_k)$ is a perfectly balanced binary tree, where the winner is A if A is the candidate associated to the root of the spanning binomial arborescence. The proof of the opposite direction is similar. $\square$

Example 12 *Consider the majority graph G over candidates A, B, C and D depicted in Figure 2. Since such a majority graph is covered by the spanning binomial arborescence T_2 with root A, we can conclude that A is a fair winner.* $\square$

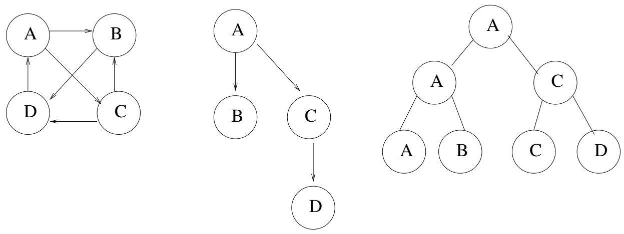

Figure 2: From a majority graph to a balanced binary tree via a spanning binomial arborescence.

The complexity of deciding if A is a fair winner is an open problem. However, if we restrict certain matches between players, then the problem is NP-complete [28]. Moreover, if a candidate satisfies certain conditions, he is a fair winner and it is possible to construct in polynomial time a balanced tree where he wins [28]. For example, if a candidate A has a number of outgoing edges in the majority graph G that is greater than or equal to the number of outgoing edges of every candidate B that beats A in G, then A is a fair winner and it is possible to compute in polynomial time a binomial spanning arborescence of G rooted in A [28].

On the other hand, if A is a Schwartz winner, it is easy to find a tree with a *bounded level of imbalance* in which A wins. We define $D(T, A)$ as the length of the path in the tree T from the root to the only leaf labelled A. We define $\Delta(A)$ as the maximum $D(T, A)$ over all competition trees T where A wins. Obviously, $\Delta(A) = m - 1$ *iff* A is the Condorcet winner. Computing $\Delta(A)$ is easy:

Proposition 4 *Given a majority graph G and a Schwartz winner A, $\Delta(A)$ is the number of outgoing edges from A in G.*

Proof: If A has k outgoing edges, it is possible to build a binary tree where A wins and appears at depth k. Let us first consider the linear tree T_1 in which A competes against the candidates $D_1, \ldots, D_k$ which it defeats directly in G. Clearly in such a tree, T_1, $D(T_1, A) = k$. We now consider, one by one, each candidate C such that $C \rightarrow A$ is an edge of G. For each such candidate we add a subtree to the current tree T_1. Let us consider the first C and the path, which we know exists, which connects A to C, say $A \rightarrow B_1 \rightarrow B_2 \ldots \rightarrow B_h \rightarrow B_{h+1} = C$. Let $j \in \{1, \ldots, h\}$ be such that B_j belongs to the current tree and $\forall i > j$, B_i does not belong to the current tree. Notice that such a candidate B_j always exists, since any path from A must start with an edge to one of $D_1, \ldots, D_k$. We then attach to the current tree the subtree corresponding to the path $B_j \rightarrow \ldots \rightarrow C$ at node B_j obtaining a new tree in which only new candidates have been added. After having considered all candidates defeating C in A, the tree obtained is a binary tree in which A wins and has depth exactly k. Thus $\Delta(A) = k$. □

Note that if A is not a Schwartz winner, this equality does not necessarily hold.

We now show that, if A is a Schwartz winner, there exists a binary tree where A wins with level of imbalance at most $m - \Delta(A) - 1$. Notice that there could also be more balanced trees in which A wins.

Proposition 5 *Given a majority graph G and a Schwartz winner A, there is a binary tree with level of imbalance smaller than or equal to $m - \Delta(A) - 1$, where A wins. This tree can be built in polynomial time.*

Proof: Since A beats $\Delta(A)$ candidates, we can easily build a balanced binary tree BT involving only A and the candidates beaten by A. Then we have to add the remaining $m - \Delta(A) - 1$ candidates to BT. Since A is a Schwartz winner, there must be a path from one of the candidates beaten by A to each of the candidates that are not beaten by A. In the worst case, this adds a subtree of depth $m - \Delta(A) - 1$ rooted at one of the nodes beaten by A in BT. □

If unfair tournaments are undesirable, we can consider those Schwartz winners for which there are binary trees which are as balanced as possible. Proposition 5 helps us in this respect: if A is a Schwartz winner, knowing $\Delta(A)$ we can compute an upper bound on the minimum imbalance of a tree in which A wins. In general, if $\Delta(A) \geq k$, then there is a tree with imbalance level smaller than or equal to $m - k - 1$, hence the higher $\Delta(A)$ is, the lower is the upper bound to the level of imbalance of a tree where A wins.

If a candidate has the maximum number of outgoing edges in the majority graph (i.e., it is a *Copeland winner* [9]), we can give a smaller upper bound on the amount of imbalance in the most balanced tree in which they win. Notice that every Copeland winner is a Schwartz winner [16].

Proposition 6 *If a candidate A is a Copeland winner, then the imbalance of a fairest/most balanced tree in which A wins is smaller than or equal to $\log(m - \Delta(A) - 1)$.*

Proof: If A is a Copeland winner, then every candidate beating A must be beaten by at least one candidate beaten by A. Consider the balanced tree BT defined in the proof of Proposition 5 rooted at A and involving precisely the candidates beaten by A. In the worst case all the remaining $m - \Delta(A) - 1$ candidates are beaten only by the same candidate in BT, say B'. In this case, however, we can add to BT a balanced subtree with depth $log(m - \Delta(A) - 1)$ rooted at B', involving all the candidates that are not beaten by A. □

Notice that Proposition 6 may be strengthened to hold for members of the uncovered set.

6.2 Incomplete weighted profiles

We now study the computational complexity of determining whether there is completion of the incomplete preferences such that a given candidate can win in a balanced tree, i.e., whether they are a possible fair winner.

We recall that the candidates that win in every balanced agenda coincide with Condorcet winners, thus the candidates that win in every balanced agenda for some completion (respectively, all completions) of the preferences coincide with possible (respectively, necessary) Condorcet winners, and thus they are polynomial to find both for weighted and unweighted profiles, and for majority graphs.

Given an incomplete weighted profile P, since every balanced agenda is also an agenda, we have that every possible fair winner is also a possible Schwartz winner. We already know from Theorem 5 that determining whether a given candidate is in $Poss(P, r_{Schwartz})$ is intractable. We will now show that this remains so also when we require that the agenda is balanced.

Theorem 10 *Given a fixed number n of candidates with $n \geq 3$ and an incomplete weighted profile P, deciding if a candidate is a possible fair winner for P is NP-complete.*

Proof: We use the same construction as in the proof of Theorem 5. Given the profile constructed there, the only possible fair agendas in which B wins are those in which A plays against C, and (at some later point) B then plays against the winner. All the additional candidates will be defeated by A, B, and C so can be placed anywhere in the fair agenda. □

The computational complexity of determining whether a given candidate is the necessary fair winner remains an open question, just as the complexity of computing necessary Schwartz winners. However, we know that, since every balanced agenda is also an agenda, every necessary fair winner is also a necessary Schwartz winner.

7 Related work

Since we have studied a number of different issues (fair winners with complete profiles, possible and necessary winners for voting trees and Schwartz, from incomplete profiles and from incomplete majority graphs), the related work spans several areas of research. We therefore structure this section according to the kind of topic studied. There are three related streams of work: voting rules with incomplete preferences, computational aspects of voting trees, and computing winners from incomplete tournaments.

7.1 Computing voting rules with incomplete preferences

We have given several results about voting trees and the Schwartz rule when voters' preferences are incomplete. The notions of possible and necessary winners for voting rules, as well as the notions of possible and necessary Condorcet winners, were introduced by Konczak and Lang [15]. The computational complexity of computing possible and necessary winners for many common voting rules was thoroughly studied by Xia and Conitzer [31]. We complete the picture by adding results concerning voting trees and Schwartz. Moreover, we consider both the standard case of incomplete profiles and the case of incomplete majority graphs, which lead to an approximation of the sets of possible winners which is easier to compute. The complexity of determining whether a given candidate is a possible or necessary winner has also been considered in Pini *et al.* [22] for the STV rule. Moreover, they give a preference elicitation procedure which focuses just on the set of possible winners. The problem of deciding if preference elicitation is over (that is, the problem of determining if the remaining votes can be cast so that a given candidate does not win) has been previously investigated also by Conitzer and Sandholm [7, 8, 30] .

7.2 Voting trees

We have considered the computational complexity of determining possible winners for voting trees. A special case of the possible winner problem is when a subset of voters provide complete votes and the other voters provide empty votes. In such situations, determining whether a given candidate is a possible winner is equivalent to deciding whether there is a constructive coalitional manipulation for this candidate. Conitzer and Sandholm [6] have investigated the complexity of weighted coalitional manipulation for $V(T)$ and found out that the latter problem is polynomial. We prove that the more general possible winner problem for $V(T)$ is NP-complete for weighted voters and 3 candidates or more.

Vu *et al.* [29] consider the following problem. The input consists of a set of candidates and a collection of pairwise winning probabilities (i, j) representing the probability that candidate i beats candidates j in a pairwise election; the problem (called "tournament schedule control") consists of designing a voting tree satisfying some given conditions (such as balancedness) that maximizes the winning probability of a target candidate. It is shown ([29], Theorems 1 and 3) that finding a balanced tree maximizing the probability that a given candidate wins is NP-hard, even if the pairwise winning probabilities are all in $\{0, 0.5, 1\}$. Our fair winners correspond to winners for which the maximum winning probability over all balanced trees is 1, when the probability matrix is composed only of 0's and 1's (for any two candidates, we know with certainty which of them beats the other in a pairwise comparison). Whether the NP-hardness result of [29] still holds when winning probabilities are 0/1, that is, whether computing fair winners is NP-hard, is still an open problem. However, Vassilevska [28] has shown that, if we allow only some of the matches between players, then the problem of determining if a candidate is a fair winner is NP-complete. Moreover, she has shown that if a candidate satisfies certain conditions, they are a fair winner and it is possible to construct in polynomial time a balanced tree in which this candidate wins. For example, if a candidate A has a number of outgoing edges in the majority graph G that is greater than or equal to the number of outgoing edges of every candidate B that beats A in G, then A is a fair winner and it is possible to compute in polynomial time a binomial spanning arborescence of G rooted in A [28].

We have given some slightly different complexity results to [29] and [32] for certain problems about voting trees with incomplete preferences. The differences concern mainly the nature of the incomplete preferences. In [29], the input is a probability matrix specifying, for every pair of candidates x, y, how likely it is that x beats y. In Sections 4.2 and 5.2, we start from an incomplete majority graph, therefore the 'qualitative' counterpart of [29]. In [32] and in Sections 4.1 and 5.1, the input is a collection of partial orders over the set of candidates.

Hazon *et al.* [14] also address the maximization over all voting trees of the probability that a given candidate wins the election, and show that a modified version of the problem of designing a tree such that computing the probability that a given candidate wins is NP-complete.

7.3 Incomplete tournaments

Brandt *et al.* [3, 4] generalize several tournament solutions to incomplete tournaments, and study their computational complexity. Among the concepts they study there are the Smith set [3] and the Schwartz set (which are known to coincide for complete tournaments). Therefore, their input is an incomplete tournament, just as we study in Section 3.2. By Lemma 2 of [3] and Definition 15, the set of the Smith winners for an incomplete tournament G defined as in [3] coincides with the set of our possible Schwartz winners for G. Therefore, our polynomiality result (Corollary 2) is subsumed by their Theorem 5. Note that these two works have been developed independently (both conference papers were published in 2007) with very different starting points and motivations.

Since the Schwartz set is contained in the Smith set even for incomplete tournaments [3], (Theorem 1 in [3]), we may wonder whether it coincides with the set of necessary Schwartz winners. However, this does not hold; for example, consider the graph G with four candidates A, B, C, D, and the edges $A >_m B$, $B >_m C$, and $C >_m A$: A is in the Smith set for G (from Lemma 4 of [3]), but from our Theorem 6, A is not a necessary Schwartz winner.

8 Conclusions and future work

We have considered multi-agent settings where agents' preferences may be incomplete and are aggregated using voting trees. We have addressed various computational issues of determining winners for complete and incomplete profiles, as well as majority graphs.

All our complexity results about determining the various types of winners are summarized in Table 1: in the first four rows we consider the voting tree rule associated with a fixed tree ($V(T)$) with incomplete preferences, as well as the case of balanced

	incomplete weighted profiles (number of candidates)	incomplete majority graphs
Possible winner for $V(T)$	NP-complete ($\geq$ 3) (Th. 2)	P (Th. 4)
Necessary winner for $V(T)$	coNP-complete ($\geq$ 4) (Th. 3)	P (Th. 4)
Possible winner for $V(T)$, T balanced	NP-complete ($\geq$ 3) (Th. 2)	P(Th. 4)
Necessary winner for $V(T)$, T balanced	?	P(Th. 4)
Necessary Schwartz winner	?	P (Cor. 1)
Possible Schwartz winner	NP-complete ($\geq$ 3) (Th. 5)	P (Cor. 2)
Possible fair winner	NP-complete ($\geq$ 3) (Th. 10)	?

Table 1: Complexity of winner determination. All the polynomial results for $V(T)$ hold for simple voting trees.

trees. The following three rows concern the Schwartz rule and the fair winner determination when the agents' preferences are incomplete.

We first considered voting trees when the tree is fixed, i.e., $V(T)$, when the agents' preferences are incomplete (rows 1, 2, 3, and 4 of Table 1). In this context, we have shown that, if we assume we know the agents' preferences in the form of incomplete majority graphs, it is polynomial to compute both the necessary and the possible winners. However, if we assume that agents' preferences are expressed by an incomplete weighted profile, then possible and necessary winners are computationally intractable to compute. We have also shown that possible winners are intractable to compute even when the agenda is balanced. The same results have been shown also in [32] in the case of incomplete unweighted profiles with an unbounded number of candidates. All these hardness results imply that it is intractable to manipulate the election by setting the agenda.

We then analyzed the Schwartz rule in the case of incomplete preferences (rows 5 and 6 of Table 1). When the preferences are expressed via an incomplete majority graph, it is easy to find possible and necessary Schwartz winners, i.e., those candidates that are Schwartz winners in some completion or in all the completions of the preferences, respectively. On the other hard, when the agents' preferences are expressed via an incomplete weighted profile, it is intractable to determine if a given candidate is a possible Schwartz winner. These two results imply that every candidate has an incentive to participate in the election which is desirable. If it were easy to find the set of possible Schwartz winners, then a candidate that was not in this set, i.e., that is a loser, might not chose to participate.

We also have defined fair winners, i.e., candidates that win in at least one balanced agenda. We have given a characterization of fair winners via spanning binomial arborescences. Moreover, we have presented a polynomial method to build an agenda with a bounded level of imbalance where a given candidate wins. The problem of determining whether a given candidate is a fair winner for a given complete profile is an open problem that we intend to investigate in the future. We have also analyzed the complexity of determining whether a given candidate is a possible fair winner when the agents' votes are incomplete and weighted, and we have shown that this problem is intractable to compute (row 7 of Table 1).

The computational complexity of determining necessary Schwartz winners is related to the problem of testing whether constructive control is possible. This problem remains open and needs to be studied further when we consider incomplete profiles. However, we have shown that it is polynomial if we consider incomplete majority graphs. Thus, it is polynomial to have an approximation of this set of winners.

An interesting direction for future work is deciding which candidates are most likely to win, which is related to probabilistic approaches to voting theory. We also plan to study other forms of uncertainty in the application of the voting rule, such as uncertain weights in a scoring rule and uncertainty about the voting rule.

References

[1] J. S. Banks. Sophisticated voting outcomes and agenda control. *Social Choice and Welfare*, 1(4):295–306, 1985.

[2] N. Betzler and B. Dorn. Towards a Dichotomy of Finding Possible Winners in Elections Based on Scoring Rules. In *MFCS*, volume 5734 of *LNCS*, pages 124–136. Springer, 2009.

[3] F. Brandt, F. Fischer, and P. Harrenstein. The computational complexity of choice sets. In *Proceedings of TARK'07*, pages 82–91, 2007.

[4] F. Brandt, F. Fischer, and P. Harrenstein. The computational complexity of choice sets. *Mathematical Logic Quarterly*, 55(4):444–459, 2009.

[5] Y. Chevaleyre, J. Lang, N. Maudet, and J. Monnot. Possible winners when new candidates are added: the case of scoring rules. In *AAAI'10*, 2010.

[6] V. Conitzer and T. Sandholm. Complexity of manipulating an election with few candidates. In *Proceedings of AAAI'02*, pages 314–319, 2002.

[7] V. Conitzer and T. Sandholm. Vote elicitation: complexity and strategy-proofness. In *Proceedings of AAAI'02*, pages 392–397, 2002.

[8] V. Conitzer, T. Sandholm, and J. Lang. When are elections with few candidates hard to manipulate. *Journal of the ACM*, 54(3), 2007.

[9] A. H. Copeland. A reasonable social welfare function. *University of Michigan Seminar on Applications of Mathematics to the Social Sciences*, 1951.

[10] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, 1990.

[11] F. A. Fischer, A. D. Procaccia, and A. Samorodnitsky. A new perspective on implementation by voting trees. In *Proceedings of EC'09*, pages 31–40. ACM, 2009.

[12] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: W.H. Freeman, 1979.

[13] N. Hazon, Y. Aumann, S. Kraus, and M. Wooldridge. Evaluation of election outcomes under uncertainty. In *Proceedings of AAMAS'08, Volume 2*, pages 959–966. IFAAMAS, 2008.

[14] N. Hazon, P. E. Dunne, S. Kraus, and M. Wooldridge. How to Rig Elections and Competitions. In *Proceedings of COMSOC'08*, 2008.

[15] K. Konczak and J. Lang. Voting procedures with incomplete preferences. In *Proceedings of IJCAI'05 Multidisciplinary Workshop on Advances in Preference Handling*, 2005.

[16] G. Laffond, J.-F. Laslier, and M. Le Breton. Condorcet choice correspondences: A set-theoretical comparison. *Mathematical Social Sciences*, pages 23–35.

[17] J. Lang, M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Winner determination in sequential majority voting. In *Proceedings of IJCAI'07*, pages 1372–1377. AAAI Press, 2007.

[18] J.-F. Laslier. *Tournament solutions and majority voting*. Springer-Verlag, 1997.

[19] N. Miller. A new solution set for tournaments and majority voting: further graph-theoretical approaches to the theory of voting. *American Journal of Political Science*, 24:68–9, 1980.

[20] H. Moulin. *Axioms of Cooperative Decision Making*. Cambridge University Press, 1988.

[21] R. Niedermeier N. Betzler, S. Hemmann. A Multivariate Complexity Analysis of Determining Possible Winners Given Incomplete Votes. In *Proceedings of IJCAI'09*, pages 53–58, 2009.

[22] M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Incompleteness and incomparability in preference aggregation. In *Proceedings of IJCAI'07*, pages 1464–1469. AAAI Press, 2007.

[23] M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Dealing with incomplete agents' preferences and an uncertain agenda in group decision making via sequential majority voting. In *Proceedings of KR'08*, pages 571–578. AAAI Press, 2008.

[24] A. D. Procaccia, A. Zohar, Y. Peleg, and J. S. Rosenschein. Learning Voting Trees. In *Proceedings of AAAI*, pages 110–115, 2007.

[25] T. Schwartz. Rationality and the myth of the maximum. *Nous*, 6(2):97–117, 1972.

[26] P. Slater. Inconsistencies in a schedule of paired comparisons. *Biometrika*, 48(3-4):303–312, 1961.

[27] M. Trick. Small binary voting trees. In *Proceedings of COMSOC'06*, pages 500–511, 2006.

[28] V. Vassilevska. Fixing a tournament. In *Proceedings of AAAI'10*, 2010.

[29] T. Vu, A. Altman, and Y. Shoham. On the complexity of schedule control problems for knockout tournaments. In *Proceedings of AAMAS'09, Volume 1*, pages 225–232. IFAAMAS, 2009.

[30] T. Walsh. Complexity of terminating preference elicitation. In *Proceedings of AAMAS'08*, pages 967–974, 2008.

[31] L. Xia and V. Conitzer. Determining possible and necessary winners under common voting rules given partial orders. In *Proceedings of AAAI'08*, pages 196–201. AAAI Press, 2008.

[32] L. Xia and V. Conitzer. Determining possible and necessary winners under common voting rules given partial orders. A longer unpublished version of [31], available at *http://www.cs.duke.edu/∼lxia*, 2010.