Most Expected Winner: An Interpretation of Winners over Uncertain Voter Preferences

ABSTRACT

It remains an open question how to determine the winner of an election given incomplete or uncertain voter preferences. One solution is to assume some probability space for the voting profile and declare that the candidates having the best chance of winning are the (co-)winners. We refer to this interpretation as the *Most Probable Winner* (MPW). In this paper, we focus on elections that use positional scoring rules, and propose an alternative winner interpretation, the *Most Expected Winner* (MEW), according to the expected performance of the candidates.

We separate the uncertainty in voter preferences into the generation step and the observation step, which gives rise to a unified voting profile combining both incomplete and probabilistic voting profiles. We use this framework to establish the theoretical hardness of MEW over incomplete voter preferences, and then identify a collection of tractable cases for a variety of voting profiles, including those based on the popular Repeated Insertion Model (RIM) and its special case, the Mallows model. We develop solvers customized for various voter preference types to quantify the candidate performance for the individual voters, and propose a pruning strategy that optimizes computation. The performance of the proposed solvers and pruning strategy is evaluated extensively on real and synthetic benchmarks, showing that our methods are practical.

CCS CONCEPTS

• **Theory of computation → Incomplete, inconsistent, and uncertain databases**; • **Applied computing → Voting / election technologies**.

KEYWORDS

Computational Social Choice, Positional Scoring Rules, Uncertain Preferences

1 INTRODUCTION

Voting is a mechanism to determine winners among the candidates in an election by aggregating voter preferences. In classical voting theory, each voter gives a complete preference (most frequently, a ranking) of all candidates. How voter preferences are aggregated is determined by a voting rule [8, 27, 29]. A prominent class of voting rules, which assign scores to candidates based on their positions in the rankings and then sum up the scores for each candidate, are positional scoring rules, on which we focus in this paper.

In practice, voter preferences may well be incomplete and represented by partial orders. Since voting rules are defined over (complete) rankings, the solution is to replace each partial order with all of its linear extensions, each of which is a *completion* or a *possible world*[1] of the partial order. The preferences of all voters in an election are referred to as a *voting profile*. A voting profile of complete rankings is a *complete voting profile*, while that of incomplete preferences is an *incomplete voting profile*. Voters are assumed to cast their preferences independently. A *completion* of an incomplete voting profile is a complete voting profile obtained by replacing each voter's partial order with one of its completions.

There have been various interpretations of winners proposed for this setting. For example, Konczak *et al.* [16] proposed the necessary winners (NWs) and possible winners (PWs): the NWs are the candidates winning in all possible worlds, while the PWs are the ones winning in at least one possible world. Bachrach *et al.* [2] assumed that the partial orders correspond to the uniform distribution over their completions, and evaluated the candidates by the number of possible worlds in which they win. We refer to this winner semantics as the *Most Probable Winner* (MPW).

In this paper, we propose the *Most Expected Winner* (MEW) as an alternative winner interpretation for incomplete voter preferences under positional scoring rules. Like MPW, it adopts the possible world semantics of incomplete voting profiles. However, in contrast to MPW that determines a winner by a (weighted) count of the possible worlds in which she wins, MEW follows the principle of score-based rules that high-scoring candidates should be favored. Specifically, *an MEW is the candidate who has the highest expected score in a random possible world.*

MEW and MPW are similar in that they both aggregate election results over all possible worlds and give a balanced evaluation of the candidates. Their difference lies in the aggregation methods, which will be discussed in detail in Section 10. In practice, MEW and MPW may yield the same result, but this is not always the case.

EXAMPLE 1. *Figure 1 gives an example where MEW and MPW select different winners in an election with two voters and three candidates, under the plurality rule. In this election, each voter produces a*

---

[1]We use *completion* and *possible world* interchangeably.

Voter	$\tau_1=\langle a,b,c\rangle$	$\tau_2=\langle b,a,c\rangle$	$\tau_3=\langle b,c,a\rangle$	$\tau_4=\langle c,b,a\rangle$
x	0.7	0.3	0	0
y	0	0	0.5	0.5

(a) A probabilistic voting profile of voters x and y.

Possible world	(Co-)winners by Plurality	$\Pr(\tau_x, \tau_y)$
x casts τ_1, y casts τ_3	$\{a, b\}$	$0.7 \times 0.5 = 0.35$
x casts τ_1, y casts τ_4	$\{a, c\}$	$0.7 \times 0.5 = 0.35$
x casts τ_2, y casts τ_3	$\{b\}$	$0.3 \times 0.5 = 0.15$
x casts τ_2, y casts τ_4	$\{b, c\}$	$0.3 \times 0.5 = 0.15$

(b) Each voter casts her vote independently of the other, leading to 4 possible worlds and 4 possible outcomes for (co-)winners under the plurality rule. The probability of a possible world is $\Pr(\tau_x, \tau_y)$. Under the plurality rule, candidate a is the MPW with a winning probability of $\Pr(\tau_1, \tau_3)+\Pr(\tau_1, \tau_4)$=0.7, while candidate b is the MEW with expected score $\Pr(\tau_2 \mid x)+\Pr(\tau_3 \mid y)$=0.3+0.5=0.8.

Figure 1: A probabilistic voting profile of 2 voters.

full ranking drawn from a probability distribution over $\tau_1 = \langle a, b, c\rangle$, $\tau_2 = \langle b, a, c\rangle$, $\tau_3 = \langle b, c, a\rangle$, $\tau_4 = \langle c, b, a\rangle$, *with probabilities of each ranking for each voter given in Figure 1a. Since voter preferences are probability distributions over rankings, the corresponding voting profile is named a* probabilistic voting profile.

Let $\Pr(\tau_x, \tau_y)$ *denote the probability that voters x and y cast rankings τ_x and τ_y, respectively. If we assume that x and y cast their votes independently, then* $\Pr(\tau_x, \tau_y) = \Pr(\tau_x \mid x) \cdot \Pr(\tau_y \mid y)$. *For example,* $\Pr(\tau_1, \tau_3) = 0.7 \cdot 0.5 = 0.35$. *The voting profile in Figure 1a generates 4 possible worlds in Figure 1b. The plurality rule rewards a candidate with 1 point every time she is ranked at the top of a ranking in the profile. So, in the possible world of* $\Pr(\tau_1, \tau_3)$, *candidate a obtains 1 point from* τ_1, *candidate b obtains 1 point from* τ_3, *and both of them become (co-)winners in this possible world. After enumerating all 4 possible worlds, we find that candidate a is the MPW with probability 0.7 to be a (co-)winner, while candidate b is the MEW with an expected score of 0.8 points.*

If the Borda rule, which rewards a candidate with the number of candidates ranked below her, is used, then, in the possible world of $\Pr(\tau_1, \tau_3)$, *candidate a would obtain 2 points from* τ_1, *while candidate b would obtain 1 point from* τ_1 *and 2 points from* τ_3. *Candidate b would then be the only NW of this profile, and she would also be the MEW with an expected score of 2.8 points.*

Contributions and Roadmap. We present related work in Section 2, and provide the necessary background on preferences and voting in Section 3.

Then, in Section 4, we present our *first contribution: a unified framework for representing uncertainty in voter preferences*, where we distinguish between the uncertainty from voters themselves (e.g., their preferences are actually probability distributions [22]) and that due to the voting mechanism (e.g., approval ballots do not allow the voters to fully reveal their preferences). We classify voting profiles into probabilistic profiles, where uncertainty is due to the voters themselves, incomplete profiles, where uncertainty is due to the voting mechanism, and combined profiles, with both kinds of uncertainty. This classification gives us a framework within

which to study the complexity of identifying MEW, by computing expected scores of the candidates.

In Section 5, we discuss several alternative interpretations of MEW and formally state the MEW problem. Then, we present our *second contribution: an investigation of the computational complexity of MEW, and its related Expected Score Computation (ESC) problem*, in Section 6. The MEW problem turns out to be $\mathrm{FP}^{\#P}$-complete under plurality, veto, and k-approval rules for the general case of uncertain profiles.[2]

In Sections 7 and 8, we present our *third contribution: exact solvers for MEW computation over different voting profiles*. We first give a solver with exponential complexity for the partial voting profiles studied in Section 6. We then identify many interesting cases, such as the RIM [7] voting profiles and the combined profiles of Mallows [21] and partitioned preferences, where it is tractable to compute the expected scores and determine the MEW (Section 7). In Section 8, we present performance optimizations to speed up the computation of MEW for large voting profiles.

In Section 9 we present our *fourth contribution: an extensive experimental evaluation of the proposed solvers*, demonstrating that our proposed methods are practical.

In Section 10 we present a thorough comparison between MEW and MPW, describing cases where MEW treats candidates differently from MPW, and highlighting the computational advantages of MEW that can lead to more efficient computation in practice.

We conclude and propose future directions in Section 11.

2 RELATED WORK

Winner semantics under incompleteness. Among the winner interpretations for incomplete preferences, the most thoroughly studied are the necessary and possible winners [16]. A candidate is a necessary winner (NW) if she wins in every possible world; she is a possible winner (PW) if she wins in at least one possible world. Chakraborty *et al.* [5] recently developed practical techniques for NW and PW computation. NW and PW have substantial shortcomings: The requirement for NW is so strict that there are often no winners available in a voting profile under this interpretation, while the requirement for PW does not differentiate between a candidate who only wins in one possible world and another candidate who only loses in one possible world. To address these limitations, alternative winner semantics for uncertain preferences have been proposed in the literature, discussed next.

Bachrach *et al.* [2] assume that an incomplete voting profile of partial orders represents a uniform distribution over its completions, and prefer the candidates who enjoy victory in more possible worlds. This winner semantics is named the *Most Probable Winner* (MPW). While this semantics is well defined under any voting rule, and while it can be extended in a straight-forward way to incorporate the probability of a completion of a voting profile, computing a winner under MPW is known to be intractable already under plurality [2]. Kenig and Kimelfeld [14] study the probability of the complement event, namely, losing an election, and devise

---

[2]Recall that $\mathrm{FP}^{\#P}$ is a class of functions efficiently solvable with an oracle to some #P problem. A function f is $\mathrm{FP}^{\#P}$-hard if there is a polynomial-time Turing reduction from any $\mathrm{FP}^{\#P}$ function to f.

an approach based on the Karp-Luby-Madras algorithm [12] for multiplicative polynomial-time approximations.

Hazon *et al.* [9] also investigate MPW but in a different setting, where voter preferences are specified explicitly by rankings and their associated probabilities. They prove that it is #P-hard to compute the winning probabilities under plurality, k-approval, Borda, Copeland, and Bucklin, and provide an approximation algorithm.

Imber and Kimelfeld [10] investigate the minimal and maximal possible ranks of candidates after rank aggregation of partial voting profiles, and prove intractability for every positional scoring rule.

Preference models. In this paper we model the uncertainty in voter preferences using a distance-based model known as the Mallows [21], and its more general forms the Repeated Insertion Model [7] and the Repeated Selection Model [5]. Others have considered uncertainty in voter preferences under the Random Utility Models (RUMs) such as the Thurstone-Mosteller (TM) [23, 31] and the Plackett-Luce (PL) [20, 26]. RUMs quantify the preferences over each item with a modal utility randomized with noise (e.g., Gaussian noise for TM and Gumbel distributions for PL), and the modal utilities are regarded as the ground truth.

Two recent papers proposed preference aggregation semantics for PL and TM models that are similar to MEW, in that they evaluate the candidates in an election based on their expected utility. Noothigattu *et al.* [24] first aggregate preferences over the TM or PL models that correspond to each voter into a summary model to represent the entire voter base (without explicit use of a scoring rule at this step), and then select the highest modal utility candidate. The authors show that their aggregation method is equivalent to Borda and Copeland. Zhao *et al.* [32] apply randomized voting rules that sample a winner from the candidates with a probability proportional to their expected scores. They demonstrate that the expected utilities of the candidates can be determined efficiently under plurality and Borda, for PL models. These papers share motivation with our work, but they do not consider distance-based preference models such as the Mallows, or their popular generalization like RIM, and do not study the complexity of winner determination for specific kinds of uncertain voting profiles. Understanding the complexity of evaluation of MEW for TM and PL models for different kinds of incomplete voting profiles and voting rules is an interesting direction for future work.

Querying probabilistic preferences. Voter preferences are a special case of preference data that has been studied in the database community [11]. Kenig *et al.* [15] propose RIMPPD, a database framework to incorporate probabilistic preferences represented by RIM models, and identify a class of tractable queries. Then Kenig *et al.* [13] further optimize the query engine with lifted inference. For a more general class of queries that are intractable in RIMPPD, Ping *et al.* [25] develop a number of exact solvers, as well as approximate techniques based on Multiple Importance Sampling. In this work, we build on some of the technical insights of these papers, and develop solvers for MEW.

3 PRELIMINARIES

3.1 Preferences

We denote by $C = \{c_1, \ldots, c_m\}$ a set of *items* or *candidates* (used interchangeably.) For any $a, b, c \in C$, preference is a binary relation

Table 1: Examples of positional scoring rules.

Voting rule	r_m
plurality	$(1, 0, \ldots, 0, 0)$
veto	$(1, 1, \ldots, 1, 0)$
2-approval	$(1, 1, 0, \ldots, 0, 0)$
Borda	$(m-1, m-2, \ldots, 1, 0)$

$>$ that is transitive ($a > b$ and $b > c$ imply $a > c$), irreflexive ($a \not> a$), and asymmetric ($a > b$ implies $b \not> a$). A *preference pair* is an instance of this relation.

A (strict) *partial order* or *poset* is a set of preference pairs that corresponds to a directed acyclic graph with edges being the preference pairs. For example, $v_0 = \{c_1 > c_2, c_1 > c_3\}$ states that candidate c_1 is preferred to both c_2 and c_3.

A *ranking* or *permutation* is a list $\tau = \langle \tau_1, \ldots, \tau_m \rangle$ such that $\forall i < j, \tau_i > \tau_j$. It defines a bijection between the candidates and the ranks: $\tau(i) = \tau_i$ and $\tau^{-1}(\tau_i) = i$.

The *linear extensions* of a partial order v, denoted by $\Omega(v)$, is a set of rankings consistent with v. For example, $C_0 = \{c_1, c_2, c_3\}$ and $v_0 = \{c_1 > c_2, c_1 > c_3\}$, then $\Omega(v_0) = \{\langle c_1, c_2, c_3 \rangle, \langle c_1, c_3, c_2 \rangle\}$.

3.2 Voting and Winners

We denote by $V = \{v_1, \ldots, v_n\}$ a set of voters and by $P = (\tau_1, \ldots, \tau_n)$ a *(complete) voting profile* where τ_i is a ranking over C by voter v_i.

Among various voting rules, the *positional scoring rules* are arguably the most thoroughly studied. Let $r_m = (r_m(1), \ldots, r_m(m))$ denote a positional scoring rule where $\forall 1 \le i < j \le m, r_m(i) \ge r_m(j)$ and $r_m(1) > r_m(m)$. It assigns a score $r_m(i)$ to the candidate at rank i. The performance of a candidate c is the sum of her scores across the voting profile P: $s(c, P) = \sum_{\tau \in P} s(c, \tau)$ where $s(c, \tau) = r_m(\tau^{-1}(c))$. Candidate w is a (co-)winner if her score is not less than that of any other candidate. We use $W(r_m, P)$ to denote the set of co-winners.

EXAMPLE 2. *Given the voting profile in Table 2, under the Borda rule,* Biden *obtains 3 points from* Ann *and* Dave*, and 0 point from* Bob*, which makes him the winner with a total score of 6.*

Most Probable Winner. Voters may only partially express their preferences. Let $P^{PO} = (v_1, \ldots, v_n)$ denote a *partial voting profile* of n partial orders. Recall that $\Omega(v)$ is the set of *linear extensions* of v, which are also called the *completions* of v. A complete voting profile $P = (\tau_1, \ldots, \tau_n)$ is a *completion* or a *possible world* of a partial voting profile $P^{PO} = (v_1, \ldots, v_n)$ if $\forall \tau_i \in P, \tau_i \in \Omega(v_i)$. Let $\Omega(P^{PO}) = \{P_1, \ldots, P_z\}$ denote the set of completions of P^{PO}. Assume that P^{PO} represents a uniform distribution of its possible worlds. The Most Probable Winner (MPW) [2, 9] is defined as follows.

voter	ranking
Ann	$\langle$ Biden, Sanders, Weld, Trump $\rangle$
Bob	$\langle$ Trump, Weld, Sanders, Biden $\rangle$
Dave	$\langle$ Biden, Sanders, Weld, Trump $\rangle$

Table 2: A voting profile of 3 voters over 4 candidates.

voter	partial order
Ann	{Biden > Weld, Sanders > Weld, Weld > Trump}
Bob	⟨Trump, Weld, Sanders, Biden⟩
Dave	⟨Biden, Sanders, Weld, Trump⟩

Table 3: A partial voting profile of 3 voters over 4 candidates. Ann gives a partial order, while Bob and Dave give rankings.

completion	voter	ranking
	Ann	⟨Biden, Sanders, Weld, Trump⟩
No. 1	Bob	⟨Trump, Weld, Sanders, Biden⟩
	Dave	⟨Biden, Sanders, Weld, Trump⟩
	Ann	⟨Sanders, Biden, Weld, Trump⟩
No. 2	Bob	⟨Trump, Weld, Sanders, Biden⟩
	Dave	⟨Biden, Sanders, Weld, Trump⟩

Table 4: The two completions of the voting profile in Table 3.

Algorithm 1 Repeated Insertion Model

Input: $\text{RIM}(\sigma, \Pi)$ where $|\sigma| = m$
Output: Ranking τ
1: Initialize an empty ranking $\tau = \langle \rangle$.
2: **for** $i = 1, \ldots, m$ **do**
3:    Insert $\sigma(i)$ into τ at $j \in [1, i]$ with probability $\Pi(i, j)$.
4: **end for**
5: **return** τ

DEFINITION 1 (MPW). *Given a partial voting profile P^{PO} and a positional scoring rule r_m, candidate $w \in C$ is the Most Probable Winner (MPW), if and only if, $\Pr(w \mid P^{PO}) = \max_{c \in C} \Pr(c \mid P^{PO})$ where $\Pr(c \mid P^{PO}) = \sum_{P \in \Omega(P^{PO})} \mathbb{1}(c \in W(r_m, P)) \cdot \Pr(P \mid P^{PO})$.*

A candidate is the MPW if her chance of winning in a random possible world is no less than any other candidate.

EXAMPLE 3. *Given the partial voting profile in Table 3, we first construct its completions listed in Table 4. Under the plurality rule, Biden is the only winner in completion No.1, while completion No.2 has winners Biden, Sanders and Trump. Both completions have the probability 0.5. Thus, Biden is the MPW with winning probability 1, while the winning probability for Sanders and Trump is 0.5.*

3.3 Preference Models

Repeated Insertion Model (RIM). Doignon *et al.* [7] proposed a generative model $\text{RIM}(\sigma, \Pi)$ that defines a probability distribution over permutations. It is parameterized by a reference ranking $\sigma = \langle \sigma_1, \ldots, \sigma_m \rangle$ and a probability function Π where $\Pi(i, j)$ is the probability of inserting σ_i at position j. Algorithm 1 presents the RIM generative procedure. It starts with an empty ranking, inserts items in the order of σ, and places σ_i at the jth position of the incomplete ranking τ with probability $\Pi(i, j)$.

Algorithm 2 Repeated Selection Model

Input: $\text{RSM}(\sigma, \Pi, p)$ where $|\sigma| = m$
Output: Partial order ν
1: Initialize an empty partial order $\nu = \{\}$.
2: **for** $i = 1, \ldots, m - 1$ **do**
3:    Sample $j \in [1, m - i + 1]$ with probability $\Pi(i, j)$.
4:    Obtain item $\sigma = \sigma(j)$ and remove it from σ.
5:    **for** $k = 1, \ldots, m - i$ **do**
6:       Add $\sigma > \sigma(k)$ into ν with probability $p(i)$.
7:    **end for**
8: **end for**
9: **return** ν

EXAMPLE 4. $\text{RIM}(\langle a, b, c \rangle, \Pi)$ *generates $\tau' = \langle c, a, b \rangle$ as follows. Initialize $\tau_0 = \langle \rangle$. When $i = 1$, $\tau_1 = \langle a \rangle$ by inserting a into τ_0 with probability $\Pi(1, 1)$. When $i = 2$, $\tau_2 = \langle a, b \rangle$ by inserting b into τ_1 at position 2 with probability $\Pi(2, 2)$. When $i = 3$, $\tau' = \langle c, a, b \rangle$ by inserting c into τ_2 at position 1 with probability $\Pi(3, 1)$. Overall, $\Pr(\tau' \mid \langle a, b, c \rangle, \Pi) = \Pi(1, 1) \cdot \Pi(2, 2) \cdot \Pi(3, 1)$.*

RIM Inference and Cover Width. Given $\text{RIM}(\sigma, \Pi)$ and a partial order ν, RIM inference is used to calculate the marginal probability of ν over $\text{RIM}(\sigma, \Pi)$, or, equivalently, the probability that a sample from $\text{RIM}(\sigma, \Pi)$ is a linear extension of ν.

RIMDP [13] is a RIM inference solver that uses Dynamic Programming (DP). It runs the RIM insertion procedure (Algorithm 1) to generate all possible rankings that satisfy the partial order. Given m items, there are potentially $m!$ permutations to generate. RIMDP prunes the space based on an insight that it is only necessary to track the positions of items directly related to some not-yet-inserted item in the partial order. Two items $\{a, b\}$ are *directly related* and a is a *cover* of b, if and only if $a > b$ and $\nexists c \in C, a > c > b$. As a DP approach, the states of RIMDP are mappings from tracked items to their positions. The maximum number of tracked items during RIM procedure is called the *cover width*, denoted $\text{cw}(\sigma, \nu)$. The complexity of RIMDP is $O(m^{\text{cw}(\sigma, \nu)+2})$. RIMDP runs in polynomial time for partial orders whose cover widths are bounded.

EXAMPLE 5. *Let $\sigma = \langle \sigma_1, \ldots, \sigma_9 \rangle$ and $\nu = \{\sigma_3 > \sigma_5, \sigma_5 > \sigma_8\}$. RIMDP does not track the positions of σ_1 and σ_2 since they are not directly related to any item in ν. The insertion of σ_3 generates 3 states: $\{\sigma_3 \to 1\}$, $\{\sigma_3 \to 2\}$, and $\{\sigma_3 \to 3\}$. The position of σ_3 is the boundary for σ_5. After inserting σ_5, RIMDP will stop tracking σ_3 by merging all states sharing the same position of σ_5, e.g., merging $\{\sigma_3 \to 1, \sigma_5 \to 3\}$ and $\{\sigma_3 \to 2, \sigma_5 \to 3\}$ into $\{\sigma_5 \to 3\}$. RIMDP will keep tracking σ_5 until σ_8 is inserted. The cover width in this example is 1, since RIMDP tracks at most 1 item for each insertion.*

Repeated Selection Model (RSM). Algorithm 2 presents a generative model for partial orders called $\text{RSM}(\sigma, \Pi, p)$ [5]. The model is parameterized by a reference ranking σ, a probability function Π where $\Pi(i, j)$ is the probability of the jth item selected at step i, and a probability function $p : \{1, \ldots, m - 1\} \to [0, 1]$ determining whether the selected item is preferred to each remaining item. In contrast to RIM that randomizes insertion positions, RSM randomizes the insertion order. In this paper, we will use RSM with $p \equiv 1$, such that it only outputs rankings, and will denote it $\text{rRSM}(\sigma, \Pi)$.

Mallows Model. Denoted by $\text{MAL}(\sigma, \phi)$, where $0 \leq \phi \leq 1$, the Mallows model [21] defines a probability distribution over rankings: reference ranking σ at the center and other rankings closer to σ having higher probabilities. For a given ranking τ, $\Pr(\tau | \sigma, \phi) \propto \phi^{D(\sigma, \tau)}$, where $D(\sigma, \tau) = |(a, a')| a >_\sigma a', a' >_\tau a|$ is the Kendall-tau distance counting the number of disagreed preference pairs. If $\phi = 1$, the Mallows becomes a uniform distribution. The $\text{MAL}(\sigma, \phi)$ is a special case for both $\text{RIM}(\sigma, \Pi)$ by $\Pi(i, j) = \frac{\phi^{i-j}}{1 + \phi + \ldots + \phi^{i-1}}$ and $\text{rRSM}(\sigma, \Pi)$ by $\Pi(i, j) = \frac{\phi^{j-1}}{1 + \phi + \ldots + \phi^{m-i}}$.

4 UNCERTAIN VOTING PROFILES

In classical voting theory, voters give complete rankings over candidates. However, in practice only partial preferences may be observed, due to the voting mechanism (e.g., when approval ballots or a ranking of at most $k < m$ candidates are elicited), the uncertainty in preferences themselves [22], or both. Figure 2 represents uncertainty as two distinct steps: preference generation (Figure 2a) and preference observation (Figure 2b). Important special cases of voting profiles are discussed next.

4.1 Uncertainty in profile generation

The most general form of voter preferences over rankings is a non-parametric probability distribution such as that given in Figure 1a. Let $P^M = (M_1, \ldots, M_n)$ denote a *probabilistic voting profile* where M_i is the ranking model of voter v_i. A *possible world* of P^M is a complete voting profile $P = (\tau_1, \ldots, \tau_n)$ where each τ_i is sampled from M_i. It is assumed that the voters cast their ballots independently, i.e., $\Pr(P \mid P^M) = \prod_{i=1}^{n} \Pr(\tau_i \mid M_i)$. So P^M represents a probability distribution of its possible worlds $\Omega(P^M) = \{P_1, \ldots, P_z\}$.

Let $s(c, M)$ and $s(c, P^M)$ denote the scores assigned to candidate c by model M and probabilistic voting profile P^M, respectively. (Note that both are random variables.) Partial voting profiles are a special case of probabilistic voting profiles, since they are based on the assumption that all completions are equally likely. Below are a few other cases.

In a *uniform voting profile*, denoted by P^U, no voter has any preference between any two candidates. Voter preferences are sampled from the uniform distribution over the rankings of all candidates.

There are specific important ranking models that we will describe in Section 7. The Mallows model [21] is the best known, and it is generalized by the Repeated Insertion Model (RIM) [7] and the ranking version of the Repeated Selection Model (RSM) [5]. Voting profiles consisting of Mallows, RIMs, and RSMs (ranking version) are denoted by P^{MAL}, P^{RIM}, and P^{rRSM}, respectively. We will see that, when the generation step is based on these models, we have a computational advantage for computing the MEW.

4.2 Uncertainty in profile observation

A *partial voting profile*, denoted by P^{PO}, consists of partial orders and represents a uniform distribution of its completions. What follows are important special cases.

A *(fully) partitioned voting profile*, denoted by P^{FP}, records preferences in the form of partitions: linear orders of item buckets, with no preference among the items within a bucket.

(a) Generation step   (b) Observation step
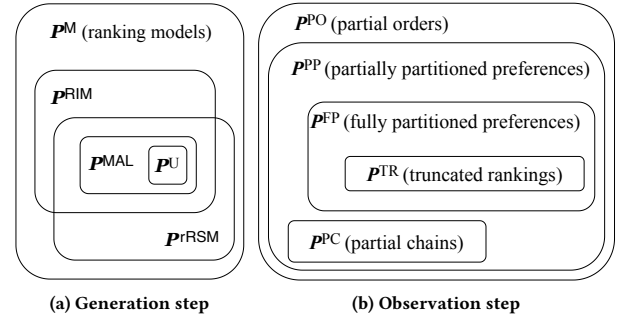
Figure 2: Uncertain voting profiles. (a) Uncertainty in the preferences by the voters themselves is represented by the ranking models in the generation step. (b) Uncertainty introduced by the preference elicitation mechanism is represented by the incomplete voting profiles in the observation step.

A *partially partitioned voting profile*, denoted by P^{PP}, generalizes the fully partitioned preferences by only considering a subset of items, with no preference information for the missing ones.

A *partial chain voting profile*, denoted by P^{PC}, records preferences in the form of partial chains. A partial chain is a linear order of a subset of items, with no preference over the remaining items. The partial chains are a special case of the partially partitioned preferences, where each partition only consists of one item.

A *truncated voting profile*, denoted by P^{TR}, records preferences in the form of truncated rankings. Let $\tau^{(t,b)}$ denote a truncated ranking with t items at the top and b items at the bottom, with no preference specified for the middle part of the ranking. The $\tau^{(t,b)}$ is a special case of the partitioned preferences, with $t + b + 1$ partitions.

Combined voting profiles. Most research works have assumed that a partial voting profile represents a uniform distribution over its completions. However, the assumption that all completions are equally likely may not hold in practice. We propose *combined voting profiles* P^{M+P}, where each voter is associated with both her original incomplete preferences P and a ranking model M. The ranking model M is her prior preference distribution, which may be obtained from historical data. But after observing new evidence P, the probabilities of rankings that violate P collapse to zero, while the remaining rankings that satisfy P have the same relative probabilities among each other. Formally speaking, the preferences of this voter become the posterior distribution of M conditioned on P.

In another sense, the combined voting profiles are also the most general form of voting profiles (same as P^M) that unify all voting profiles so far. For example, a partitioned voting profile P^{FP} is essentially P^{U+FP}, and a RIM voting profile P^{RIM} is essentially $P^{RIM+\emptyset}$ where $\emptyset$ means that the partial orders are the empty ones.

Among the tractable cases in Section 7, there are also combined voting profiles, which means that applying combined voting profiles both brings the benefit of more customized ranking distributions, and can also be practical.

5 MOST EXPECTED WINNER

For a general voting profile and a positional scoring rule, the performance of a candidate can be quantified by the expectation of her score in a random possible world. We define this formally below.

DEFINITION 2 (MEW). *Given a general voting profile P^M and a positional scoring rule r_m, candidate w is a Most Expected Winner, if and only if, $\mathbb{E}(s(w, P^M)) = \max_{c \in C} \mathbb{E}(s(c, P^M))$.*

We denote the set of Most Expected Winners by $\text{MEW}(r_m, P^M)$.

5.1 Alternative Interpretations

To gain an intuition for Most Expected Winner (MEW), we will now give two winner definitions that are equivalent to MEW. (See supplementary materials [1] for theorem proofs in this section.)

Least Expected Regret Winner. The MEW can also be regarded as the candidate who minimizes the expected regret in a random possible world. The concept of *regret* is borrowed from [19]. Let $\text{Regret}(w, P)$ denote the regret value of choosing $w \in C$ as the winner given a complete voting profile P.

$$\text{Regret}(w, P) = \max_{c \in C} s(c, P) - s(w, P)$$

Accordingly, the regret value $\text{Regret}(w, P^M)$ over a probabilistic voting profile becomes a random variable.

$$\mathbb{E}(\text{Regret}(w, P^M)) = \sum_{P \in \Omega(P^M)} \text{Regret}(c, P) \cdot \Pr(P \mid P^M)$$

Candidate w is a *Least Expected Regret Winner*, if and only if, she minimizes the expected regret $\mathbb{E}(\text{Regret}(w, P^M))$.

THEOREM 1. *Least Expected Regret Winner is equivalent to MEW.*

Meta-Election Winner. Recall that a voting profile P^M represents a probability distribution of possible worlds $\Omega(P^M) = \{P_1, \ldots, P_z\}$. The Meta-Election Winner is defined as the candidate who wins a meta election with a large meta profile $P_M = (P_1, \ldots, P_z)$ where rankings in P_i are weighted by $\Pr(P_i \mid P^M)$.

THEOREM 2. *Meta-Election Winner is equivalent to MEW.*

5.2 Problem Statement

The MEW is determined based on the expected performance of the candidates. Thus, the winner determination problem of MEW can be reduced to the problem of Expected Score Computation (ESC), stated below and addressed in the remainder of the paper.

DEFINITION 3 (ESC). *Given a general voting profile P^M, a positional scoring rule r_m, and a candidate $c \in C$, compute $\mathbb{E}(s(c, P^M))$ the expected score of the candidate c.*

6 HARDNESS OF ESC

This section investigates the complexity of the ESC problem. We first prove the hardness of two closely related problems, the Fixed-rank Counting Problem and the Rank Estimation Problem, and then demonstrate that the ESC problem is hard as well. See supplementary materials [1] for all proofs.

6.1 Fixed-rank Counting Problem

Counting the number of linear extensions of a partial order is well-known to be #P-complete [3]. The Fixed-rank Counting Problem (FCP) is interested in the number of linear extensions where an item is placed at a specific rank. Let $\Omega(v)$ denote the set of linear extensions of v, and $N(c \rightarrow j \mid v)$ denote the number of linear extensions in $\Omega(v)$ where item c is placed at rank j.

DEFINITION 4 (FCP). *Given a partial order v over m items, an item c, and an integer $j \in [1, m]$, calculate $N(c \rightarrow j \mid v)$, the number of linear extensions of v where item c is placed at rank j.*

De Loof's doctoral dissertation [18] discusses this problem (Section 4.2.1), proposing exact and approximate algorithms, but does not provide proof of complexity.

THEOREM 3. *The FCP is #P-complete.*

The hardness of FCP facilitates the hardness proofs for the Rank Estimation Problem.

6.2 Rank Estimation Problem

Now we move on to the Rank Estimation Problem (REP). This problem can be regarded as the probabilistic version of the FCP. It calculates the probability that a given item is placed at a specific rank. But the REP is generalized from partial orders to arbitrary ranking distributions.

DEFINITION 5 (REP). *Given a ranking model M over m items, an item c, and an integer $j \in [1, m]$, calculate $\Pr(c \rightarrow j \mid M)$, the probability that item c is placed at rank j, by M.*

For the convenience of discussions in the rest of this Section, we also define two special cases of the REP where items are placed at the top and bottom of the linear extensions.

DEFINITION 6 (REP-t). *Given a ranking model M of m items and an item c, calculate $\Pr(c \rightarrow 1 \mid M)$, the probability that item c is placed at the top of a linear extension, by M.*

DEFINITION 7 (REP-b). *Given a ranking model M of m items and an item c, calculate $\Pr(c \rightarrow m \mid M)$, the probability that item c is placed at the bottom of a linear extension, by M.*

Lerche and Sørensen [17] proposed an approximation for the REP over partial orders, but did not provide a formal complexity proof. Bruggemann and Annoni [4] and De Loof *et al.* [6] considered a related problem, calculating the expected rank of an item in the linear extensions of a partial order. These works focused on approximation, lacking complexity proofs as well.

With the help of Theorem 3, it turns out that the REP and its two variants are all FP$^{\#P}$-complete over partial orders.

LEMMA 1. *If ranking model M is a partial order v of m items representing a uniform distribution of $\Omega(v)$, the REP-t is FP$^{\#P}$-complete.*

LEMMA 2. *If ranking model M is a partial order v of m items representing a uniform distribution of $\Omega(v)$, the REP-b is FP$^{\#P}$-complete.*

THEOREM 4. *If ranking model M is a partial order v of m items representing a uniform distribution of $\Omega(v)$, the REP is FP$^{\#P}$-complete.*

6.3 Complexity of ESC

ESC is closely related to REP. Firstly, ESC is no harder than REP over general voting profiles (Theorem 5), which lays the foundation for the identification of tractable cases in Section 7.

THEOREM 5. *Given a general voting profile P^M and a positional scoring rule r_m, the ESC problem can be reduced to the REP.*

THEOREM 6. *The REP for rank k is equivalent to the ESC problem over either one or both of the $(k-1)$-approval and k-approval rules.*

Theorem 6 provides an insight into the relation between REP and ESC in terms of computational complexity. If a solver is available for the ESC problem over a collection of k-approval votes, this solver is computationally equivalent to the REP solver. Note that Theorem 6 is not limited to partial voting profiles.

THEOREM 7. *Given a partial voting profile P^{PO}, a distinguished candidate c, and plurality rule r_m, the ESC problem of calculating $\mathbb{E}(s(c \mid P^{PO}, r_m))$ is $FP^{\#P}$-complete.*

THEOREM 8. *Given a partial voting profile P^{PO}, a distinguished candidate c, and veto rule r_m, the ESC problem of calculating $\mathbb{E}(s(c \mid P^{PO}, r_m))$ is $FP^{\#P}$-complete.*

THEOREM 9. *Given a partial voting profile P^{PO}, a distinguished candidate c, and k-approval rule r_m, the ESC problem of calculating $\mathbb{E}(s(c \mid P^{PO}, r_m))$ is $FP^{\#P}$-complete.*

The three theorems above demonstrate the hardness of the ESC problem over partial voting profiles, under plurality, veto, and k-approval, respectively. In particular, ESC is $FP^{\#P}$-complete even under plurality (Theorem 7).

7 MOST EXPECTED WINNER SOLVERS

The problem of determining MEW can be reduced to the ESC problem (Definition 3), then further reduced to the REP by Theorem 5. This section will solve the MEW problem by solving the REP. See supplementary materials [1] for all theorem proofs.

7.1 Solver for RIM-based partial orders

We adopt the RIMDP algorithm [13], discussed in Section 3.3, to develop an exact solver for the REP over posets, and more generally, for RIM combined with posets. The exact solver only modifies the RIMDP by always tracking the target candidate, such that when RIM insertions finish, RIMDP obtains the probabilities of the target candidate being placed at each rank.

The RIMDP has exponential complexity $O(m^{cw(\sigma,\nu)+2})$, where $cw(\sigma,\nu)$ is the cover width, given the reference ranking σ and the partial order ν [13]. It takes $O(nm^{cw(\sigma,\nu)+3})$ to calculate MEW by executing RIMDP for all m candidates and n voters.

7.2 Solvers for Partitioned Preferences, Truncated Rankings, and Partial Chains

It turns out that the MEW problem can be solved efficiently for all special cases of partial voting profiles in Figure 2b, i.e., the partitioned voting profile,

THEOREM 10. *Given a positional scoring rule r_m, a fully partitioned voting profile $P^{FP} = (\nu_1^{FP}, \ldots, \nu_n^{FP})$, and candidate w, determining $w \in MEW(r_m, P^{FP})$ is in $O(nm^2)$.*

Algorithm 3 REP solver for RIM

Input: Item c, $RIM(\sigma, \Pi)$ where $|\sigma| = m$
Output: $\{k \to \Pr(c \to k \mid \sigma, \Pi) \mid k \in [1, m]\}$
1: $\delta_0 := \emptyset$, $\mathcal{P}_0 := \{\delta_0\}$ and $q_0(\delta_0) := 1$
2: **for** $i = 1, \ldots, m$ **do**
3:    $\mathcal{P}_i := \{\}$
4:    **for** $\delta \in \mathcal{P}_{i-1}$ **do**
5:      **for** $j = 1, \ldots, i$ **do**
6:        **if** $\sigma(i)$ is c **then**
7:          $\delta' := \{c \to j\}$
8:        **else if** $\delta = \{c \to k\}$ and $j \le k$ **then**
9:          $\delta' := \{c \to k+1\}$
10:        **else**
11:          $\delta' := \delta$
12:        **end if**
13:        **if** $\delta' \notin \mathcal{P}_i$ **then**
14:          $\mathcal{P}_i.add(\delta')$
15:          $q_i(\delta') := 0$
16:        **end if**
17:        $q_i(\delta') += q_{i-1}(\delta) \cdot \Pi(i, j)$
18:      **end for**
19:    **end for**
20: **end for**
21: $\forall k \in [1, m], \Pr(c \to k \mid \sigma, \Pi) = q_m(\{c \to k\})$.
22: **return** $\{k \to \Pr(c \to k \mid \sigma, \Pi) \mid k \in [1, m]\}$

The truncated voting profiles can be solved by the same approach above, since they are a special case of the fully partitioned profile.

THEOREM 11. *Given a positional scoring rule r_m, a partial chain voting profile $P^{PC} = (\nu_1^{PC}, \ldots, \nu_n^{PC})$, and candidate w, determining $w \in MEW(r_m, P^{PC})$ is in $O(nm^2)$.*

The MEW over partially partitioned preferences can be solved by extending the above approach for partial chain voting profiles.

THEOREM 12. *Given a positional scoring rule r_m, a partially partitioned voting profile $P^{PP} = (\nu_1^{PP}, \ldots, \nu_n^{PP})$, and candidate w, determining $w \in MEW(r_m, P^{PP})$ is in $O(nm^2)$.*

7.3 Solver for Probabilistic Voting Profiles

While the problem of MEW determination is hard in general, it is tractable over specific ranking models such as the Mallows [21] and its generalizations RIM [7] and RSM [5].

THEOREM 13. *Given positional scoring rule r_m, a RIM voting profile $P^{RIM} = (RIM_1, \ldots, RIM_n)$, and candidate w, determining $w \in MEW(r_m, P^{RIM})$ is in $O(nm^4)$.*

The complexity of MEW determination over RSM voting profiles is $O(nm^4)$ as well. Supplementary proofs for details [1].

7.4 Solver for Combined Voting Profiles

It is usually harder to compute the expected scores over combined voting profiles. Below are some cases where this problem is tractable. The first case is the RIMs combined with truncated rankings.

THEOREM 14. *Given a positional scoring rule r_m, a voting profile $P^{RIM+TR} = ((RIM_1, \tau_1^{(t_1,b_1)}), \ldots, (RIM_n, \tau_n^{(t_n,b_n)}))$, and candidate w, determining $w \in MEW(r_m, P^{RIM+TR})$ is in $O(nm^4)$.*

Table 5: Complexity of exact MEW solvers under general positional scoring rules for various voting profiles, including partial orders (PO), partially partitioned preferences (PP), fully partitioned preferences (FP), partial chains (PC), RIMs, rRSMs, Mallows, and combined profiles. A dagger (†) refers to results presented in supplementary materials [1]. An asterisk (*) means that the RIM and rRSM models are effectively the Mallows models in the experimental benchmarks.

Profile	Complexity	Solver	Experiments
P^{PO}	$O(nm^{cw+3})$	RIMDP	Figures 5 and 6
P^{PP}	$O(nm^2)$	Theorem 12	Figure 7
P^{FP}	$O(nm^2)$	Theorem 10	Figure 8
P^{PC}	$O(nm^2)$	Theorem 11	Figure 9
P^{TR}	$O(nm^2)$	Theorem 10, as P^{FP}	Figure 10
P^{RIM}	$O(nm^4)$	Theorem 13	
P^{MAL}	$O(nm^4)$	Theorem 13, as P^{RIM}	Figure 11
P^{rRSM}	$O(nm^4)$	Theorem 16†	Figure 12*
P^{RIM+PO}	$O(nm^{cw+3})$	RIMDP	Figure 13c
P^{RIM+PP}	$O(nm^{cw+3})$	RIMDP, as P^{RIM+PO}	
P^{RIM+PC}	$O(nm^{cw+3})$	RIMDP, as P^{RIM+PO}	
P^{RIM+FP}	$O(nm^{cw+3})$	RIMDP, as P^{RIM+PO}	
P^{RIM+TR}	$O(nm^4)$	Theorem 14	Figure 13b*
P^{MAL+FP}	$O(nm^4)$	Theorem 15	Figure 13a
P^{MAL+TR}	$O(nm^4)$	Theorem 15, as P^{MAL+FP}	
P^{MAL+PP}	$O(nm^{cw+3})$	RIMDP, as P^{RIM+PP}	
P^{MAL+PC}	$O(nm^{cw+3})$	RIMDP, as P^{RIM+PC}	
P^{MAL+PO}	$O(nm^{cw+3})$	RIMDP, as P^{RIM+PO}	

Mallows combined with partitioned preferences is also tractable.

THEOREM 15. *Given a positional scoring rule r_m, a voting profile $P^{MAL+FP} = \left((MAL_1, v_1^{FP}), \ldots, (MAL_n, v_n^{FP}) \right)$, and candidate w, determining $w \in MEW(r_m, P^{MAL+FP})$ is in $O(nm^4)$.*

7.5 Summary

Table 5 summarizes all solvers in this Section, including additional conclusions for a large number of specialized voting profiles. For example, the complexity over Mallows voting profiles is bounded by $O(nm^4)$, since they are a special case of P^{RIM}. An interesting observation is that the MEW complexity over P^{RIM+FP} is not tractable, but its special case P^{MAL+FP} has polynomial complexity. Although this table demonstrates that evaluating MEW over probabilistic voting profiles has higher complexity than that over incomplete voting profiles (e.g., $O(nm^4)$ for P^{MAL+FP} but only $O(nm^2)$ for P^{FP}), the tractability results over a collection of probabilistic and combined voting profiles give the MEW a computation advantage in practice. Note that the combined voting profiles with RSM models are not covered by Table 5, since the complexity of determining the MEW for them is still an open question.

8 OPTIMIZATION STRATEGIES

The algorithms in Section 7 find the MEW by computing candidate scores for a single voter at a time, and then aggregating the scores across the voting profile. We now present two optimization strategies for a voting profile solver in Algorithm 4.

Algorithm 4 Solver for voting profiles

Input: general voting profile $P^M = (M_1, \ldots, M_n)$, voting rule r_m
Output: MEW(r_m, P^M)

1: Group identical votes in P^M, sort them by the group size in descending order, to generate a weighted voting profile P_w where each vote M_i is associated with a weight w_i
2: Obtain candidates $C = \{c_1, \ldots, c_m\}$ from P_w
3: $\forall c \in C, UB(c) := 0, LB(c) := 0$
4: **for** $(M, w) \in P_w$ **do**
5:     **for** $c \in C$ **do**
6:         Compute the best possible rank a of c over M
7:         Compute the worst possible rank b of c over M
8:         $UB(c) += w \cdot r_m(a), LB(c) += w \cdot r_m(b)$
9:     **end for**
10: **end for**
11: Compute $max(LB)$, prune any candidate c if $UB(c) < max(LB)$
12: **for** $(M, w) \in P_w$ **do**
13:     Let C_t denote the candidates currently tracked by UB and LB
14:     **for** $c \in C_t$ **do**
15:         Compute the exact score of c over M
16:         Refine $UB(c)$ and $LB(c)$ with its exact score and the weight w
17:     **end for**
18:     Compute $max(LB)$, prune any candidate c if $UB(c) < max(LB)$
19:     **if** Only one candidate remains in UB and LB **then**
20:         Let c' denote the only remaining candidate
21:         **return** MEW(r_m, P^M) = c'
22:     **end if**
23: **end for**
24: Let C' denote the set of remaining candidates in UB and LB
25: **return** MEW(r_m, P^M) = C'

Candidate pruning. We can save computation by iteratively pruning candidates who cannot be the MEW. The key idea is to quickly compute score upper- and lower-bounds for all candidates across the voting profile, progressively refine these scores during per-voter computation, and prune candidates who cannot be the MEW based on the upper- and lower-bounds computed so far.

Algorithm 4 presents the candidate pruning procedure. The first step is to compute $UB(c)$ and $LB(c)$, the upper- and lower-bounds of the score of each candidate c by quickly computing their best possible and worst possible ranks assigned by each voter (lines 3 - 10). For example, in a given partial order M (voter) in a partial voting profile, the best possible rank of candidate c is bounded by the number of its ancestors in M, while its worst possible rank is bounded by the number of its descendants. Combing this information with the voting rule, we can efficiently obtain score upper- and lower-bounds of each candidate for each voter.

EXAMPLE 6. *Let $v = \{\sigma_3 > \sigma_5, \sigma_3 > \sigma_8\}$ be a partial order, over $m = 10$ items, and assume the voting rule is 2-approval. Item σ_5 has 1 ancestor σ_3 and no descendant in v. So its best possible rank is 2 and its worst possible rank is m, which correspond to scores 1 and 0, respectively, under 2-approval. Thus, 1 is the score upper-bound of σ_5, and 0 the score lower-bound, for v.*

Per-voter score upper-bounds and lower-bounds can be aggregated (summed up) across all voters in the profile (line 8 of Algorithm 4). The pruning strategy can then confidently determine the

(a) candidate pruning　　　　**(b) voter grouping**
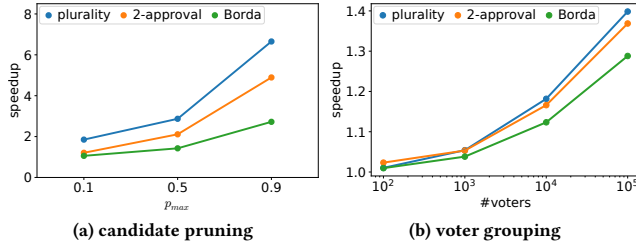
Figure 3: Performance of optimization strategies over partial voting profiles, with 10 cand., $\phi = 0.5$. Candidate pruning with 10,000 voters; effectiveness improves with p_{max}. Voter grouping with $p_{max} = 0.1$; effectiveness improves with #voters.

(a) running time　　　　**(b) speedup**

Figure 4: Parallel implementation for MEW over partial voting profiles of 10 candidates, 1 million voters, $\phi = 0.5$, and $p_{max} = 0.1$. The speedup due to parallelism is nearly linear in the number of worker processes.

losing candidates as those whose score upper-bounds are lower than the best observed score lower-bound (line 11 of Algorithm 4).

Score upper- and lower-bounds are refined as the algorithm iterates over the voters (lines 12- 23), computing the exact scores of viable (tracked) candidates, refining $UB(c)$ and $LB(c)$, and pruning any candidates that are no longer viable (line 18).

If, at any point during the iteration, only a single tracked candidate w remains, then it is returned as the MEW (line 21). Once all voters M have been iterated over, return all remaining tracked candidates as the winners. (Note that if after all voters are iterated, and there are still multiple tracked items, then their upper- and lower-bounds are precisely their exact scores, and they all share the same exact score, otherwise the candidates with lower exact scores would have been pruned during voter iteration.)

EXAMPLE 7. *Assume $UB = \{\sigma_1 \rightarrow 15, \sigma_2 \rightarrow 6\}$ and $LB = \{\sigma_1 \rightarrow 5, \sigma_2 \rightarrow 2\}$ at a time point during refining the UB and LB. Assume that after computing the exact scores of σ_1 and σ_2 over the latest voter, the refined $UB = \{\sigma_1 \rightarrow 12, \sigma_2 \rightarrow 5\}$ and the refined $LB = \{\sigma_1 \rightarrow 7, \sigma_2 \rightarrow 3\}$. Now, candidate σ_2 can be pruned, since $UB(\sigma_2) < LB(\sigma_1)$. After pruning σ_2, only σ_1 is tracked by UB and LB, thus we can confidently declare σ_1 as the winner.*

Voter grouping. As another optimization, we observe that when multiple voters cast the same vote, we only need to compute the expected candidate scores for this vote once. When computing the MEW over a voting profile, the first step is to group identical votes and sort them by their frequencies in descending order, which generates a new voting profile that is weighted and sorted.

9 EXPERIMENTS

We systematically tested the proposed algorithms and optimization strategies with both synthetic and real datasets, described below. The experiment code is available in the supplementary materials [1]. All experiments were conducted on a Linux machine with two Intel Xeon Platinum 8268 24-core 2.9GHz Processors and 384GB RAM.

Synthetic data generators. The synthetic partial orders are generated by RSMs. Let $RSM(\boldsymbol{\sigma}, \Pi, p)$ denote the preference model of a voter where $\boldsymbol{\sigma} = \langle \sigma_1, \dots, \sigma_m \rangle$ and $\Pi(i, j) = \frac{\phi^{j-1}}{1+\phi+\dots+\phi^{m-i}}$, which
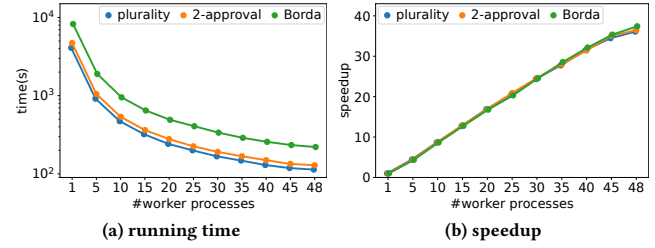
makes its item selection order equivalent to $MAL(\boldsymbol{\sigma}, \phi)$. Then uniformly draw probabilities $p : \{1, \dots, m-1\} \rightarrow \text{Uniform}(0, p_{max})$ where p_{max} is part of the parameter setup in our experiments.

Partially partitioned preferences are generated with specified numbers of partitions. First, create an ordered list of partitions, and randomly choose an item for each partition to guarantee that each partition has at least one item. Then, create an additional partition for the missing items. Finally, assign each of the remaining items to a random partition, including the "missing items" partition. Fully partitioned preferences are generated in a similar fashion, except that there is no partition for the missing items. Partial chains also use this generator, except that it stops at the first step where each partition has been assigned a single random item.

Truncated rankings are generated by first drawing a ranking from the uniform ranking distribution, and keeping only the top and the bottom parts of the ranking.

*Real datasets.* We applied the proposed solvers to three real datasets of different voting profile types.

CrowdRank [30] is a dataset of partial chains over movies collected from Amazon Mechanical Turk. It consists of 50 human intelligence tasks (HITs), each of which asked 100 users to rank 20 movies that they were familiar with.

MovieLens is a dataset of movie ratings collected by GroupLens (www.grouplens.org). We used the most frequently rated movies and converted the movie ratings by the same user into partially partitioned preferences, given that each user only rated a subset of the movies. We then obtained a partially partitioned voting profile with 200 movies and 6040 users.

Travel [28] is a dataset of ratings of European attractions that belong to 24 categories in Google Reviews. The ratings are average values given by each user for each attraction category. We converted these average ratings into a fully partitioned voting profile with 24 attraction categories and 5456 users.
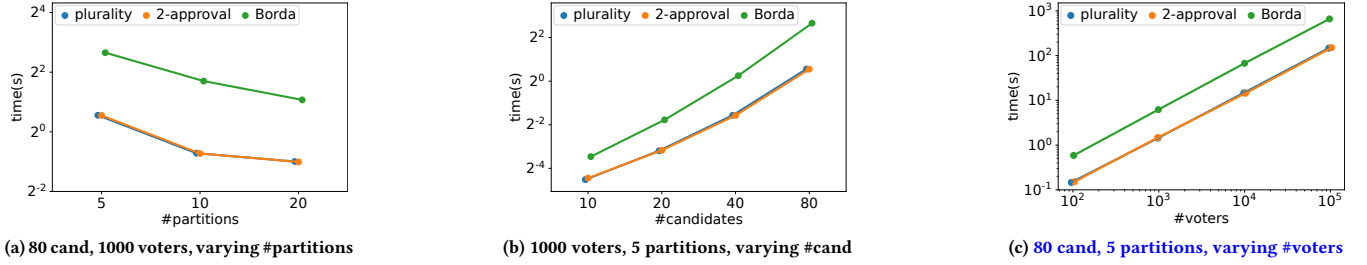
9.1 Performance optimizations

We tested the performance of optimization strategies over synthetic partial voting profiles, with 10 candidates under three voting rules. The partial orders are generated by $RSM(\boldsymbol{\sigma}, \Pi, p)$ and p_{max} as described above. In each voting profile, all voters share the same reference ranking $\boldsymbol{\sigma}$ and insertion probabilities Π, and their item

(a) 1000 voters

(b) $\phi = 0.5$
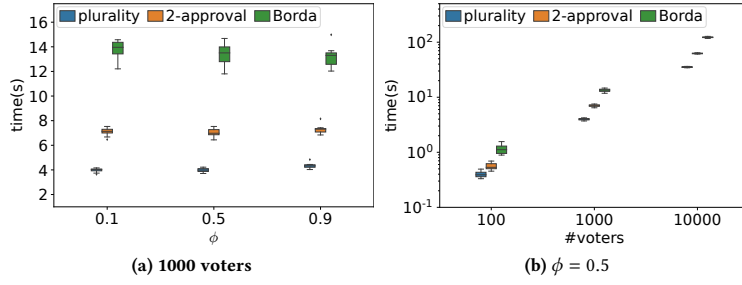
Figure 5: Running time over partial voting profiles, fixing 10 candidates and $p_{max} = 0.1$. MEW is determined fastest under plurality, slowest under Borda. Running time is insensitive to changes in ϕ (voter preference diversity). Running time increases linearly with the number of voters.
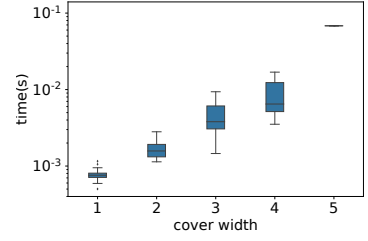
Figure 6: Running time over partial orders increases exponentially with cover widths. 100 partial orders generated with RSM over 10 candidates, $p_{max} = 0.1$, and insertion probabilities equivalent to Mallows with $\phi = 0.5$.

(a) 80 cand, 1000 voters, varying #partitions

(b) 1000 voters, 5 partitions, varying #cand

(c) 80 cand, 5 partitions, varying #voters

Figure 7: Average time over partially partitioned profiles. It declines with more partitions in the voter preferences, increases with more candidates in the voting profile, and increases linearly with the number of voters.

selection orders are equivalent to MAL($\boldsymbol{\sigma}, \phi$). For the convenience of discussion, we use ϕ as the setting that describes Π.

Candidate pruning. We turned off voter grouping to test the performance of candidate pruning. For each parameter setting, we generated 10 voting profiles, fixing 10,000 voters and $\phi = 0.5$, varying p_{max}, and computed MEW with and without candidate pruning. We present speedup as a function of p_{max} for 10,000 voters in Figure 3a, for plurality, 2-approval and Borda. We observe that pruning never hurts performance, and that it is more effective for higher values of p_{max}, the parameter that controls the density (number of preference pairs) in a partial order. Pruning is most effective for plurality, followed by 2-approval and then by Borda. We thus conclude that these performance optimizations are effective at reducing the running time of MEW computation, and we use both optimizations in all remaining experiments.

Voter grouping. In this experiment, we turned off candidate pruning, and tested the voter grouping optimization. For each parameter setting, we generated 10 voting profiles, fixing $\phi = 0.5$ and $p_{max} = 0.1$, varying the number of voters, and computed MEW with and without voter grouping. The speedup of voter grouping is the running time without voter grouping divided by the running time with voter grouping. Figure 3b demonstrates that voter grouping never hurts performance, and that the speedup realized by this optimization increases with the number of voters.
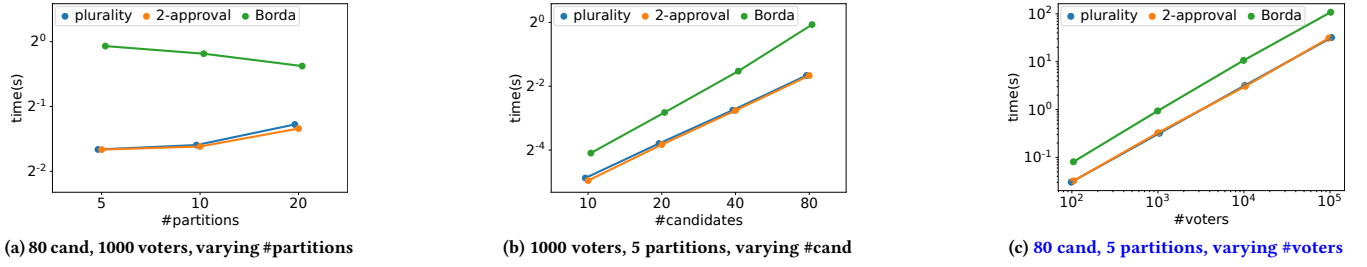
Parallel computation of MEW. Algorithm 4 gives a sequential implementation of MEW, with linear complexity in the number of

voters. To handle a large volume of voters in real-world settings, the computation of MEW can be easily parallelized by turning off the candidate pruning strategy and allocating all votes to different CPU cores. We tested the parallel implementation of MEW over 10 synthetic partial voting profiles of 10 candidates and 1 million voters. The partial orders are generated with $\phi = 0.5$ and $p_{max} = 0.1$, using the same methodology as in Section 9.1. Figure 4a demonstrates that the solver runs faster with more worker processes. In Figure 4b, the speedup of the parallel solver relative to the sequential solver increases linearly with the number of worker processes.
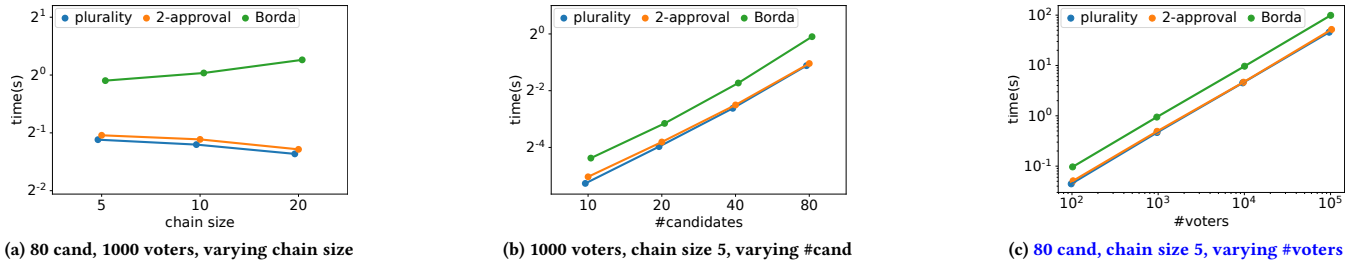
9.2 Incomplete voting profiles

Partial voting profiles. A collection of synthetic profiles of 10 candidates are generated to test the scalability of the solver in Section 7.1. Synthetic partial orders are generated in the same way as in Section 9.1. We generated 10 voting profiles for each parameter setting, and calculated the MEW under three voting rules, with performance optimizations discussed in Section 8.

Figure 5 demonstrates the impact of voting rules on the running time. MEW is determined fastest under plurality, followed by 2-approval, and finally, by Borda. In Figure 5a, the ϕ parameter of the Mallows model is varied to test the impact of voter consensus level on the running time, which turns out to be minor. Figure 5b shows the linearly increasing running time with the number of voters.

(a) 80 cand, 1000 voters, varying #partitions

(b) 1000 voters, 5 partitions, varying #cand

(c) 80 cand, 5 partitions, varying #voters

Figure 8: Average time over fully partitioned profiles. It increases under plurality and 2-approval, but declines under Borda, with more partitions in the voter preferences. It increases linearly when adding more candidates or voters.

(a) 80 cand, 1000 voters, varying chain size

(b) 1000 voters, chain size 5, varying #cand

(c) 80 cand, chain size 5, varying #voters

Figure 9: Average time over partial chain profiles is more sensitive to the Borda rule when increasing the chain size, and increases linearly with the number of candidates or voters for all rules.

(a) 80 candidates, 1000 voters

(b) 80 candidates, top / bottom size 5
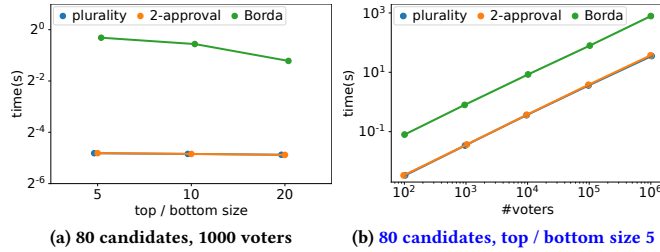
Figure 10: Average time over truncated voting profiles. It stays the same for plurality and 2-approval, but steadily decreases, when increasing the top and bottom sizes of the truncated rankings. It increases linearly with the number of voters.

Partial orders and cover width. A benchmark of partial orders is prepared to investigate the impact of the cover width parameter, discussed in Section 3.3, on performance. We fixed the reference ranking of the RSM over 10 candidates, and made its insertion probabilities equivalent to $\phi = 0.5$ in Mallows. For each partial order, we sample a new vector of edge construction probabilities with $p_{max} = 0.1$, generating 100 partial orders. Figure 6 shows that the exact solver for partial orders has exponential complexity with the cover width, which is consistent with our complexity analysis.

Partially partitioned voting profiles. Synthetic profiles are generated with the parameter settings in Figure 7. For each setup, we generated 10 profiles and calculated the MEW with the solver in Theorem 12 and Algorithm 4. Figure 7 shows their average running

time to demonstrate the scalability w.r.t. the number of partitions, candidates, and voters. In Figure 7a, the running time decreased with greater number of partitions. Figures 7b and 7c show that the running time increases linearly with more candidates and voters.

Fully partitioned voting profiles. The setup of this experiment is identical to that of the partially partitioned preferences above. While Figure 8b also gives the linear growth of running time with increasing number of candidates, Figure 8a shows opposite trends between Borda and the other two voting rules. It turns out that under Borda rule, the pruning strategy benefits from the increasing number of partitions. In comparison, after turning off the pruning strategy, the running time of computing MEW under the Borda rule increased from 0.76 seconds to 0.79 seconds.

Partial chain voting profiles. The setup of this experiment is almost identical to the experiments of partitioned voting profiles, except that the number of partitions is now the chain size. Figure 9 presents the average running time to compute the MEW with candidate pruning under various parameter settings. In Figure 9a, running time increases mildly with chain size, especially under plurality and 2-approval. In Figure 9b, the running time increases nearly linearly with more candidates and linearly with more voters.

Truncated voting profiles. In this experiment, we fixed the number of candidates to be 80, and varied the top and bottom sizes of the truncated rankings, and the number of voters. For each parameter setting, we generated 10 voting profiles and computed the average running time of MEW with the candidate pruning strategy.

Figure 10a demonstrates that the top and bottom size has no impact on the running time over plurality and 2-approval, which is expected since these voting rules only need information about

the top-1 or 2 ranked candidates. Under Borda, MEW computation is faster for higher top and bottom size, also as expected, since uncertainty in the truncated rankings decreases accordingly. Finally, Figure 10b demonstrates that the running time over truncated voting profiles increases linearly with the number of voters.

9.3 Probabilistic voting profiles

Mallows voting profiles. We generate synthetic profiles to test the performance of the RIM solver over its special case, the Mallows model. We vary the number of candidates from 10 to 80, and the dispersion parameter ϕ from 0.1 to 0.9, generate 10 profiles for each parameter setting, and report the average running time, with pruning, in Figure 11. Figure 11a shows that the solver scales well with the number of candidates, while Figure 11b shows that the time to compute MEW is insensitive to the Mallows parameter ϕ.

RSM voting profiles. This experiment uses exactly the same voting profiles as the experiment above for Mallows models. Since rRSM generalizes the Mallows model, we converted the Mallows models into rRSM instances and invoked the RSM solver for these converted voting profiles. Compared to Figure 11a, Figure 12a shows better scalability with regards to increasing number of candidates, especially for the plurality and 2-approval rules. Figure 12b gives a similar conclusion as Figure 11b that the dispersion of the preferences has little impact over the running time.

9.4 Combined voting profiles

Mallows with fully partitioned preferences. In this experiment, we investigate the impact of the Mallows ϕ on the running time. All voting profiles have 10 candidates, 1000 voters, and 2 partitions for the partitioned preferences. Figure 13a shows that the Mallows ϕ has little impact on the running time, and that the running time under plurality and 2-approval rules are almost identical. Similar trends were observed for more than 2 partitions, and we do not report these results here.

Mallows with truncated rankings. The setup of this experiment is similar to the above, for Mallows combined with fully partitioned preferences, except that the top and bottom sizes of the truncated rankings are fixed to be 3, instead of fixing the number of partitions. Figure 13b demonstrates that the dispersion ϕ has little impact on the running time. Similar trends were observed for other truncated ranking sizes, and we do not report these results here.

Mallows with partial orders. The setup of this experiment is similar to the previous two experiments for combined voting profiles. All voters in a profile have the same Mallows model. Their partial orders are generated by RSMs whose selection probabilities are derived from dispersion ϕ of their Mallows, and each voter has an independent edge construction probability vector sampled from $p_{max} = 0.9$. Figure 13c shows that ϕ substantially impacts the running time: The computation of MEW is faster with a small ϕ value, especially for Borda. This is expected, because the voters exhibit stronger consensus for lower values of ϕ, and so the pruning strategy is more effective.

9.5 Real datasets

Figure 14 compares the running time of computing MEW under 3 voting rules for the 50 HITs in CrowdRank. Recall that each HIT has

Table 6: Computing MEW for real datasets

Dataset	#cand	#voters	Running time (in seconds)		
			plurality	2-approval	Borda
MovieLens	200	6040	137	140	344
Travel	24	5456	0.57	0.49	0.35

its own set of 20 movies and 100 partial chains over these movies. The experiment result shows that the voting rule has a significant impact on the running time. It took much longer to compute MEW under the Borda rule, compared to plurality and 2-approval.

Table 6 presents the running time of MEW over MovieLens and Travel, with number of items m (i.e., movies and attraction categories, respectively), and number of voters n. MovieLens is a partially partitioned voting profile of 200 movies. It took about 38 minutes to compute the MEW under both plurality and 2-approval rules, and the winner is *American Beauty* in both cases. In comparison, *Star Wars: Episode IV - A New Hope* is the winner under Borda, computed in around 100 minutes. Travel is a fully partitioned voting profile with 24 attraction categories. It took less than 1 second to compute the MEW under all three voting rules, which yields *museums* as the winner.

10 COMPARING MEW AND MPW

We now conduct a thorough comparison between MEW and MPW to better demonstrate their differences. MEW and MPW can be interpreted as different aggregation approaches across possible worlds. Recall that given a general voting profile P^M, $\Omega(P^M) = \{P_1, \ldots, P_z\}$ is the set of its possible worlds of P^M, and each P_i is associated with a probability $p_i = \Pr(P_i \mid P^M)$ and $\sum_{i=1}^{z} p_i = 1$. Now let's see how the performance of a candidate c is aggregated across possible worlds. Let $\mathbb{1}()$ be the indicator function.

- MEW: $\mathbb{E}(s(c, P^M)) = \sum_{i=1}^{z} s(c, P_i) \cdot p_i$
- MPW: $\Pr(c \text{ wins} \mid P^M) = \sum_{i=1}^{z} \mathbb{1}(c \text{ wins} \mid P_i) \cdot p_i$

MEW estimates the average performance of a candidate, while MPW estimates the probability that she wins. As a result, MPW ignores the possible worlds in which the candidate cannot win, putting certain candidates at a disadvantage.

EXAMPLE 8. *Let $P^M = \{M_0\}$ be a single-voter voting profile over 4 candidates $\{a, b, c, d\}$. Assume M_0 is a ranking model where $\Pr(\langle b, a, c, d \rangle) = \Pr(\langle c, a, b, d \rangle) = \Pr(\langle d, a, b, c \rangle) = 1/3$. Under Borda rule, MEW favors candidate a who, despite losing in every possible world, enjoys the highest expected score of 2, while the MPWs are $\{b, c, d\}$, each winning in one possible world.*

MEW also quantifies the performance of candidates more granularly than MPW, as illustrated in the next example.

EXAMPLE 9. *Let $P^{PO} = \{v_0\}$ be a single-voter voting profile over 4 candidates $\{a, b, c, d\}$. Assume $v_0 = \{a > b, b > c, b > d\}$. Under the Borda rule, MEW and MPW agree that candidate a is the winner, but they disagree on the performance of candidate b. MPW cannot differentiate between b, c, and d, since none of them wins in any possible world, but MEW believes that b outperforms c and d, since b has a higher expected score than c and d.*
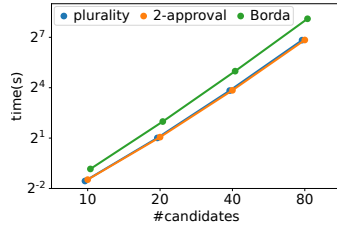
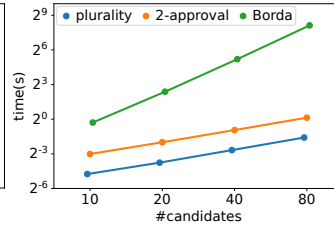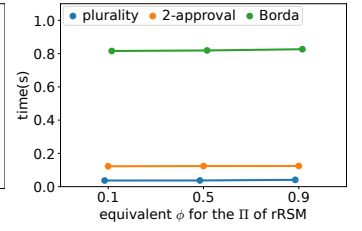**(a)** $\phi = 0.5$, varying #candidates
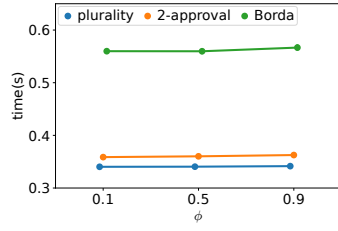
(b) 10 candidates, varying ϕ

Figure 11: Average time over Mallows voting profiles, fixing 1000 voters. It increases linearly with the number of candidates, while the ϕ has little impact over the running time.

(a) $\phi = 0.5$, varying #candidates
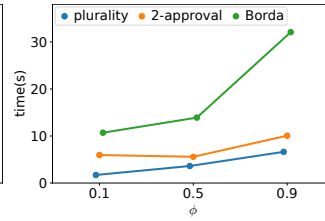
(b) 10 candidates, varying ϕ

Figure 12: Average time over RSM voting profiles, fixing 1000 voters. It increases linearly with the number of candidates, while the equivalent ϕ has little impact over the running time.
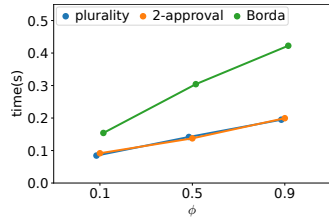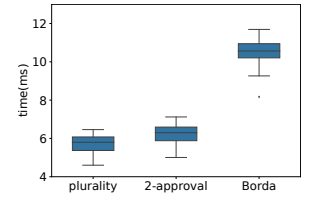

(a) $P^{\text{MAL+FP}}$
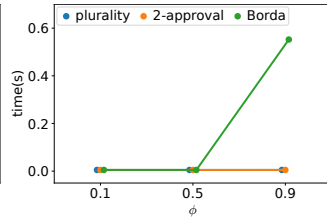
(b) $P^{\text{MAL+TR}}$

(c) $P^{\text{MAL+PO}}$
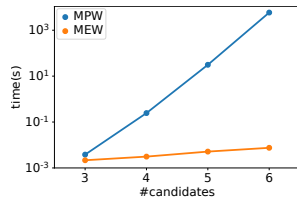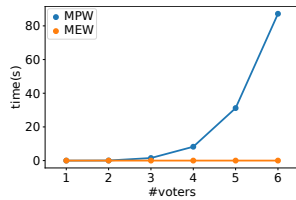
Figure 13: Average time over combined voting profiles. The ϕ has little impact over the running time when the Mallows models are combined with fully partitioned preferences or truncated rankings, while the running time increases with ϕ for voting profiles of the Mallows combined with partial orders, especially under the Borda rule.

Figure 14: **Running time on CrowdRank. Determining MEW under Borda is more demanding than under plurality or 2-approval.**

(a) 5 voters, varying #candidates

(b) 5 candidates, varying #voters

Figure 15: **Average time over partial voting profiles, fixing $\phi = 0.5$ and $p_{max} = 0.1$. MEW scales much better than MPW, with both #candidates and # voters.**

MEW also has a computational advantage over MPW, making it practical for real-world applications: Although both are intractable in the general case, MEW enjoys linear complexity in the number of voters. Figure 15 compares the running times of MEW and MPW over synthetic partial voting profiles that are generated by the same methodology as in Figure 5. We implemented the MPW solver based on the *VotingResult* algorithm by Hazon *et al.* [9], with the slight modification that the solver iterates over the completions of each voter's partial order, while the original algorithm assumes that each voter gives an explicit probability distribution over the rankings. The running time of the MPW solver grows exponentially in the number of voters, forcing us to use very small voting profiles in

this experiment. We fixed the number of voters to 5 and varied the number of candidates from 3 to 6, then fixed the number of candidates to 5 and varied the number of voters from 1 to 6. In Figures 15a and 15b, MEW scales much better than MPW.

11 CONCLUDING REMARKS

In this paper we modeled uncertainty in voter preferences with the help of a framework that distinguishes between uncertainty in preference generation and uncertainty in preference observation, unifying incomplete and probabilistic voting profiles. We then proposed the Most Expected Winner (MEW) semantics for positional scoring rules and established the theoretical hardness of this problem. We identified tractable cases with the help of the uncertainty framework for voting profiles, and developed solvers.

Much exciting future work remains. For example, the hardness of ESC is proved over only plurality, veto, and k-approval, which calls for investigation of other positional scoring rules such as Borda. When MEW is intractable, it may be necessary to develop approximate solvers. MEW can also be extended to other score-based rules, such as Simpson and Copeland. Another direction is to consider voter preferences represented by additional ranking models [22] such as the Plackett-Luce (PL) [20, 26] and Thurstone-Mosteller (TM) [23, 31]. For example, others [24, 32] have studied a preference aggregation method over PL models that is closely related to the MEW over PL models, and we plan to investigate this connection further in the future.

REFERENCES

[1] Anonymous. 2022. Most Expected Winner: An Interpretation of Winners over Uncertain Voter Preferences (Supplementary material). https://anonymous.4open.science/r/mew-supplementals-45F7/supp.pdf

[2] Yoram Bachrach, Nadja Betzler, and Piotr Faliszewski. 2010. Probabilistic Possible Winner Determination. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010.* http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/view/1692

[3] Graham R. Brightwell and Peter Winkler. 1991. Counting Linear Extensions is #P-Complete. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA.* 175–181. https://doi.org/10.1145/103418.103441

[4] Rainer Bruggemann and Paola Annoni. 2014. Average heights in partially ordered sets. *MATCH Commun Math Comput Chem* 71 (2014), 117–142.

[5] Vishal Chakraborty, Theo Delemazure, Benny Kimelfeld, Phokion G. Kolaitis, Kunal Relia, and Julia Stoyanovich. 2021. Algorithmic Techniques for Necessary and Possible Winners. *Trans. Data Sci.* 2, 3 (2021), 22:1–22:23. https://doi.org/10.1145/3458472

[6] Karel De Loof, Bernard De Baets, and Hans De Meyer. 2011. Approximation of average ranks in posets. *Match Commun Math Comput Chem* 66 (2011), 219–229.

[7] Jean-Paul Doignon, Aleksandar Pekeč, and Michel Regenwetter. 2004. The repeated insertion model for rankings: Missing link between two subset choice models. *Psychometrika* 69, 1 (2004), 33–54.

[8] Judy Goldsmith, Jérôme Lang, Nicholas Mattei, and Patrice Perny. 2014. Voting with Rank Dependent Scoring Rules. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada,* Carla E. Brodley and Peter Stone (Eds.). AAAI Press, 698–704. http://www.aaai.org/ocs/index.php/AAAI/AAAI14/paper/view/8549

[9] Noam Hazon, Yonatan Aumann, Sarit Kraus, and Michael J. Wooldridge. 2012. On the evaluation of election outcomes under uncertainty. *Artif. Intell.* 189 (2012), 1–18. https://doi.org/10.1016/j.artint.2012.04.009

[10] Aviram Imber and Benny Kimelfeld. 2021. Probabilistic Inference of Winners in Elections by Independent Random Voters. In *AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021,* Frank Dignum, Alessio Lomuscio, Ulle Endriss, and Ann Nowé (Eds.). ACM, 647–655. https://dl.acm.org/doi/10.5555/3463952.3464031

[11] Marie Jacob, Benny Kimelfeld, and Julia Stoyanovich. 2014. A System for Management and Analysis of Preference Data. *Proc. VLDB Endow.* 7, 12 (2014), 1255–1258. https://doi.org/10.14778/2732977.2732998

[12] Richard M. Karp, Michael Luby, and Neal Madras. 1989. Monte-Carlo Approximation Algorithms for Enumeration Problems. *J. Algorithms* 10, 3 (1989), 429–448. https://doi.org/10.1016/0196-6774(89)90038-2

[13] Batya Kenig, Lovro Ilijasic, Haoyue Ping, Benny Kimelfeld, and Julia Stoyanovich. 2018. Probabilistic Inference Over Repeated Insertion Models. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018.* 1897–1904. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16252

[14] Batya Kenig and Benny Kimelfeld. 2019. Approximate Inference of Outcomes in Probabilistic Elections. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019.* AAAI Press, 2061–2068. https://doi.org/10.1609/aaai.v33i01.33012061

[15] Batya Kenig, Benny Kimelfeld, Haoyue Ping, and Julia Stoyanovich. 2017. Querying Probabilistic Preferences in Databases. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2017, Chicago, IL, USA, May 14-19, 2017,* Emanuel Sallinger, Jan Van den Bussche, and Floris Geerts (Eds.). ACM, 21–36. https://doi.org/10.1145/3034786.3056111

[16] Kathrin Konczak and Jérôme Lang. 2005. Voting procedures with incomplete preferences. In *Proc. IJCAI-05 Multidisciplinary Workshop on Advances in Preference Handling,* Vol. 20.

[17] Dorte Lerche and Peter B Sørensen. 2003. Evaluation of the ranking probabilities for partial orders based on random linear extensions. *Chemosphere* 53, 8 (2003), 981–992.

[18] Karel De Loof. 2009. *Efficient computation of rank probabilities in posets.* Ph. D. Dissertation. Ghent University.

[19] Tyler Lu and Craig Boutilier. 2011. Robust Approximation and Incremental Elicitation in Voting Protocols. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011.* 287–293. https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-058

[20] R.D. Luce. 1959. *Individual Choice Behavior: A Theoretical Analysis.* Wiley. https://books.google.com/books?id=a80DAQAAIAAJ

[21] C. L. Mallows. 1957. Non-Null Ranking Models. *Biometrika* 44 (1957), 114–130.

[22] John I Marden. 1995. *Analyzing and modeling rank data.* CRC Press.

[23] Frederick Mosteller. 1951. Remarks on the method of paired comparisons: I. The least squares solution assuming equal standard deviations and equal correlations. *Psychometrika* 16, 1 (March 1951), 3–9. https://doi.org/10.1007/BF02313422

[24] Ritesh Noothigattu, Snehalkumar (Neil) S. Gaikwad, Edmond Awad, Sohan Dsouza, Iyad Rahwan, Pradeep Ravikumar, and Ariel D. Procaccia. 2018. A Voting-Based System for Ethical Decision Making. In *Proceedings of AAAI, New Orleans, Louisiana, USA.* 1587–1594. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17052

[25] Haoyue Ping, Julia Stoyanovich, and Benny Kimelfeld. 2020. Supporting Hard Queries over Probabilistic Preferences. *Proc. VLDB Endow.* 13, 7 (2020), 1134–1146. https://doi.org/10.14778/3384345.3384359

[26] Robin L Plackett. 1975. The analysis of permutations. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 24, 2 (1975), 193–202.

[27] Geoffrey Pritchard and Mark C. Wilson. 2007. Exact results on manipulability of positional voting rules. *Soc. Choice Welf.* 29, 3 (2007), 487–513. https://doi.org/10.1007/s00355-007-0216-5

[28] Shini Renjith, A. Sreekumar, and M. Jathavedan. 2018. Evaluation of Partitioning Clustering Algorithms for Processing Social Media Data in Tourism Domain. In *2018 IEEE Recent Advances in Intelligent Computational Systems (RAICS).* 127–131. https://doi.org/10.1109/RAICS.2018.8635080

[29] Roman Seidl. 2018. Handbook of Computational Social Choice *by Brandt Felix, Vincent Conitzer, Ulle Endriss, Jerome Lang, Ariel Procaccia. J. Artif. Soc. Soc. Simul.* 21, 2 (2018). http://jasss.soc.surrey.ac.uk/21/2/reviews/4.html

[30] Julia Stoyanovich, Marie Jacob, and Xuemei Gong. 2015. Analyzing Crowd Rankings. In *Proceedings of the 18th International Workshop on Web and Databases, Melbourne, VIC, Australia, May 31, 2015,* Julia Stoyanovich and Fabian M. Suchanek (Eds.). ACM, 41–47. https://doi.org/10.1145/2767109.2767110

[31] L. L. Thurstone. 1927. A Law of Comparative Judgment. *Psychological Review* 34, 4 (1927), 273–286. https://doi.org/10.1037/h0070288

[32] Zhibing Zhao, Haoming Li, Junming Wang, Jeffrey O. Kephart, Nicholas Mattei, Hui Su, and Lirong Xia. 2018. A Cost-Effective Framework for Preference Elicitation and Aggregation. In *Proceedings of UAI, Monterey, California, USA.* 446–456. http://auai.org/uai2018/proceedings/papers/179.pdf

A ADDITIONAL PROOFS

THEOREM 1. *Least Expected Regret Winner is equivalent to MEW.*

PROOF. Let P^{M} denote a general voting profile with $\Omega(P^{\mathsf{M}}) = \{P_1, \ldots, P_z\}$. The expected regret of a candidate w can be rewritten as follows.

$$
\begin{aligned}
&\mathbb{E}(\text{Regret}(w, P^{\mathsf{M}})) \\
&= \sum_{i=1}^{z} \text{Regret}(c, P_i) \cdot \Pr(P_i \mid P^{\mathsf{M}}) \\
&= \sum_{i=1}^{z} \left(\max_{c \in C} s(c, P_i) - s(w, P_i) \right) \cdot \Pr(P_i \mid P^{\mathsf{M}}) \\
&= \sum_{i=1}^{z} \max_{c \in C} s(c, P_i) \cdot \Pr(P_i \mid P^{\mathsf{M}}) - \mathbb{E}(s(w, P^{\mathsf{M}}))
\end{aligned}
$$

The first term $\sum_{i=1}^{z} \max_{c \in C} s(c, P_i) \cdot \Pr(P_i \mid P^{\mathsf{M}})$ is a constant value, when P^{M} and the voting rule are fixed. Thus, $\mathbb{E}(\text{Regret}(w, P^{\mathsf{M}}))$ is minimized by maximizing $\mathbb{E}(s(w, P^{\mathsf{M}}))$, the expected score of the candidate w. □

THEOREM 2. *Meta-Election Winner is equivalent to MEW.*

PROOF. Let P^{M} denote a general voting profile with $\Omega(P^{\mathsf{M}}) = \{P_1, \ldots, P_z\}$, and $P_{meta} = (P_1, \ldots, P_z)$ denote the large meta profile where rankings in P_i are weighted by $\Pr(P_i \mid P^{\mathsf{M}})$. According to the definition of the Meta-Election Winner, $s(w, P_{meta}) = \max_{c \in C} s(c, P_{meta})$. As a result, for any candidate c,

$$
\mathbb{E}(s(c, P^{\mathsf{M}})) = \sum_{P \in \Omega(P^{\mathsf{M}})} s(c, P) \cdot \Pr(P \mid P^{\mathsf{M}}) = s(c, P_{meta})
$$

Her expected score in P^{M} is precisely her score in P_{meta}. The two winner definitions are optimizing the same metric. □

THEOREM 3. *The FCP is #P-complete.*

PROOF. First, we prove its membership in #P. The FCP is the counting version of the following decision problem: given a partial order ν, an item c, and an integer j, determine whether ν has a linear extension $\tau \in \Omega(\nu)$ where c is ranked at j. This decision problem is obviously in NP, meaning that the FCP is in #P.

Then, we prove that the FCP is #P-hard by reduction. Recall that counting $|\Omega(\nu)|$, the number of linear extensions of a partial order ν, is #P-complete [3]. This problem can be reduced to the FCP by $|\Omega(\nu)| = \sum_{j=1}^{m} N(c@j \mid \nu)$.

In conclusion, the FCP is #P-complete. □

LEMMA 1. *If ranking model M is a partial order ν of m items representing a uniform distribution of $\Omega(\nu)$, the REP-t is $FP^{\#P}$-complete.*

PROOF. First, we prove that the REP-t is in $FP^{\#P}$. Recall that $\Omega(\nu)$ is the linear extensions of a partial order ν, and $N(c@1|\nu)$ is the number of linear extensions in $\Omega(\nu)$ where candidate c is

at rank 1. Then $\Pr(c@1|\nu) = N(c@1|\nu)/|\Omega(\nu)|$. Consider that counting $N(c@1 \mid \nu)$ is in #P (Theorem 3) and counting $|\Omega(\nu)|$ is #P-complete [3], so $\Pr(c@1|\nu)$ is in $FP^{\#P}$.

In the rest of this proof, we prove that the REP-t is #P-hard by reduction from the #P-complete problem of counting $|\Omega(\nu)|$.

Let c^* denote an item that has no parent in ν. Let ν_{-c^*} denote the partial order of ν with item c^* removed. If we are interested in the probability that c^* is placed at rank 1, we can write $\Pr(c^*@1|\nu) = N(c^*@1 \mid \nu)/|\Omega(\nu)|$. The item c^* has been fixed at rank 1, so any placement of the rest items will definitely satisfy any relative order involving c^*. That is to say, the placement of the rest items just needs to satisfy ν_{-c^*}, which leads to $N(c^*@1 \mid \nu) = |\Omega(\nu_{-c^*})|$.

For example, let $\nu' = \{c_1 > c_4, c_2 > c_4, c_3 > c_4\}$. Then $N(c_1@1 \mid \nu') = |\Omega(\nu'_{-c_1})| = |\Omega(\{c_2 > c_4, c_3 > c_4\})|$.

Then we re-write $\Pr(c^*@1|\nu) = N(c^*@1|\nu)/|\Omega(\nu)| = |\Omega(\nu_{-c^*})|/|\Omega(\nu)|$. The oracle for $\Pr(c^*@1|\nu)$ manages to reduce the size of the counting problem from $|\Omega(\nu)|$ to $|\Omega(\nu_{-c^*})|$. This oracle should be as hard as counting $|\Omega(\nu)|$. Thus calculating $\Pr(c^*@1|\nu)$ is $FP^{\#P}$-hard.

In conclusion, the REP-t is $FP^{\#P}$-complete. □

LEMMA 2. *If ranking model M is a partial order ν of m items representing a uniform distribution of $\Omega(\nu)$, the REP-b is $FP^{\#P}$-complete.*

PROOF. This proof adopts the same approach as the proof of Lemma 1.

Let m be the number of items in the ranking model M. For the membership proof that the REP-b is in $FP^{\#P}$, let $N(c@m|\nu)$ denote the number of linear extensions in $\Omega(\nu)$ where candidate c is at the bottom rank m. Then $\Pr(c@m|\nu) = N(c@m|\nu)/|\Omega(\nu)|$. Consider that counting $N(c@m|\nu)$ is in #P (Theorem 3) and counting $|\Omega(\nu)|$ is #P-complete [3], so $\Pr(c@m \mid \nu)$ is in $FP^{\#P}$.

In the proof of Lemma 1, item c^* is an item with no parent in the partial order ν. In the current proof, item c^* is set to be an item with no child in ν. The ν_{-c^*} still denotes the partial order of ν but with item c^* removed. Then the probability that item c^* at the bottom rank m is $\Pr(c^*@m \mid \nu) = N(c^*@m \mid \nu)/|\Omega(\nu)| = |\Omega(\nu_{-c^*})|/|\Omega(\nu)|$. The oracle for $\Pr(c^*@m \mid \nu)$ manages to reduce the size of the counting problem again from $|\Omega(\nu)|$ to $|\Omega(\nu_{-c^*})|$. Thus, this oracle is #P-hard, and calculating $\Pr(c^*@m \mid \nu)$ is $FP^{\#P}$-hard.

In conclusion, the REP-b is $FP^{\#P}$-complete. □

THEOREM 4. *If ranking model M is a partial order ν of m items representing a uniform distribution of $\Omega(\nu)$, the REP is $FP^{\#P}$-complete.*

PROOF. First, we prove that the REP is in $FP^{\#P}$. Recall that $\Omega(\nu)$ is the linear extensions of a partial order ν, and $N(c@j|\nu)$ is the number of linear extensions in $\Omega(\nu)$ where candidate c is at rank j. Then $\Pr(c@j|\nu) = N(c@j|\nu)/|\Omega(\nu)|$. Consider that counting $N(c@j|\nu)$ is #P-complete (Theorem 3) and counting $|\Omega(\nu)|$ is #P-complete [3] as well. So $\Pr(c@j|\nu)$ is in $FP^{\#P}$.

Lemma 1 demonstrates that REP-t, a special case of REP, is $FP^{\#P}$-hard. Thus REP is #P-hard as well.

In conclusion, REP is $FP^{\#P}$-complete. □

THEOREM 5. *Given a general voting profile P^{M} and a positional scoring rule r_m, the ESC problem can be reduced to the REP.*

PROOF. Recall that the MEW w maximizes the expected score, i.e.,

$$s(w, P^M) = \max_{c \in C} \mathbb{E}(s(c, P^M))$$

The voting profile P^M contains n ranking distributions $\{M_1, \ldots, M_n\}$, so

$$\mathbb{E}(s(c, P^M)) = \sum_{i=1}^{n} \mathbb{E}(s(c, M_i))$$

where $\mathbb{E}(s(c, M_i))$ is the expected score of c from voter v_i.

$$\mathbb{E}(s(c, M_i)) = \sum_{j=1}^{m} Pr(c@j \mid M_i) \cdot r_m(j)$$

where $c@j$ denotes candidate c at rank j, and $r_m(j)$ is the score of rank j.

Let $\mathbb{T}$ denote the complexity of calculating $Pr(c@j|M_i)$. The original MEW problem can be solved by calculating $Pr(c@j|M_i)$ for all m candidates, m ranks and n voters, which leads to the complexity of $O(n \cdot m^2 \cdot \mathbb{T})$. □

THEOREM 6. *The REP for rank k is equivalent to the ESC problem over either one or both of the $(k-1)$-approval and k-approval rules.*

PROOF. The ESC problem has been reduced to the REP (Theorem 5). This proof will focus on the other direction, i.e., reducing the REP to the ESC problem.

Let $Pr(c@j \mid M)$ denote the probability of placing candidate c at rank j over a ranking distribution M. Let P^M denote a single-voter profile consisting of only this ranking distribution M.

When $k = 1$, the REP can be reduced to solving the ESC problem under plurality or 1-approval rule.

$$Pr(c@1 \mid M) = \mathbb{E}(s(c \mid P^M, 1\text{-approval}))$$

When $k = m$, the REP can be reduced to solving the ESC problem under veto or $(m-1)$-approval rule.

$$Pr(c@m \mid M) = 1 - \mathbb{E}(s(c \mid P^M, (m-1)\text{-approval}))$$

When $2 \leq k \leq m$, the REP can be reduced to solving the ESC problem twice under k-approval and $(k-1)$-approval rules.

$$Pr(c@k \mid M) = \mathbb{E}(s(c \mid P^M, k\text{-approval}))$$
$$- \mathbb{E}(s(c \mid P^M, (k-1)\text{-approval}))$$

□

THEOREM 7. *Given a partial voting profile P^{PO}, a distinguished candidate c, and plurality rule r_m, the ESC problem of calculating $\mathbb{E}(s(c \mid P^{PO}, r_m))$ is $FP^{\#P}$-complete.*

PROOF. Firstly, we prove the membership of the ESC problem as an $FP^{\#P}$ problem. Consider that the REP is $FP^{\#P}$-complete over partial orders (Theorem 4), and the ESC problem can be reduced to the REP (Theorem 5) So the ESC problem is in $FP^{\#P}$ for partial voting profiles.

Secondly, we prove that the ESC problem is $FP^{\#P}$-hard, even for plurality rule, by reduction from the REP-t that is $FP^{\#P}$-hard (Lemma 1).

Let ν denote the partial order of the REP-t problem. Recall that the REP-t problem aims to calculate $Pr(c@1 \mid \nu)$ for a given item c.

Let P^ν denote a voting profile consisting of just this partial order ν. The answer to the REP-t problem is the same as the answer to the corresponding ESC problem, i.e., $Pr(c@1 \mid \nu) = \mathbb{E}(s(c \mid P^\nu, \text{plurality}))$. So the ESC problem is $FP^{\#P}$-hard, even for plurality voting rule.

In conclusion, the ESC problem is $FP^{\#P}$-complete, under plurality rule. □

THEOREM 8. *Given a partial voting profile P^{PO}, a distinguished candidate c, and veto rule r_m, the ESC problem of calculating $\mathbb{E}(s(c \mid P^{PO}, r_m))$ is $FP^{\#P}$-complete.*

PROOF. This proof adopts the same approach as the proof of Theorem 7.

Firstly, the membership proof that the ESC is in $FP^{\#P}$ is based on the conclusions that the REP is $FP^{\#P}$-complete over partial orders (Theorem 4), and that the ESC can be reduced to the REP (Theorem 5) So the ESC is in $FP^{\#P}$ for partial voting profiles.

Secondly, we prove that the ESC is $FP^{\#P}$-hard, under veto voting rule, by reduction from the REP-b that is $FP^{\#P}$-hard (Lemma 2).

Let ν denote the partial order of the REP-b problem. Recall that the REP-b problem aims to calculate $Pr(c@m \mid \nu)$ for a given item c. Let P^ν denote a voting profile consisting of just this partial order ν. The answer to the ESC indirectly solves the REP-b, i.e., $Pr(c@m \mid \nu) = 1 - \mathbb{E}(s(c \mid P^\nu, \text{veto}))$. So the ESC problem is $FP^{\#P}$-hard under veto rule.

In conclusion, the ESC is $FP^{\#P}$-complete, under veto rule. □

THEOREM 9. *Given a partial voting profile P^{PO}, a distinguished candidate c, and k-approval rule r_m, the ESC problem of calculating $\mathbb{E}(s(c \mid P^{PO}, r_m))$ is $FP^{\#P}$-complete.*

PROOF. Firstly, the proof that the Expected Score Computation (ESC) is in $FP^{\#P}$ is the same as the proof of Theorem 7. Now we prove that the ESC problem is $FP^{\#P}$-hard, under k-approval rule r_m, by reduction from the REP-t problem that is $FP^{\#P}$-hard (Lemma 1).

Let ν denote the partial order of the REP-t problem. Recall that the REP-t problem aims to calculate $Pr(c@1 \mid \nu)$ for a given item c. Let ν_+ denote a new partial order by inserting $(k-1)$ ordered items $d_1 > \ldots > d_{k-1}$ into ν such that item d_{k-1} is preferred to every item in ν. Such placement of items $\{d_1, \ldots, d_{k-1}\}$ is to guarantee that all linear extensions of ν_+ start with $d_1 > \ldots > d_{k-1}$ and these linear extensions will be precisely the linear extensions of ν after removing $\{d_1, \ldots, d_{k-1}\}$.

Let P^{ν_+} denote a voting profile consisting of just this partial order ν_+. The answer to the ESC problem for item c is $\mathbb{E}(s(c \mid P^{\nu_+}, k\text{-approval}))$. Since there is only one partial order ν_+ in the voting profile, $\mathbb{E}(s(c \mid P^{\nu_+}, k\text{-approval})) = \sum_{j=1}^{k} Pr(c@j \mid \nu_+)$. Recall that any linear extension of ν_+ always starts with $d_1 > \ldots > d_{k-1}$, so $\forall 1 \leq j \leq (k-1), Pr(c@j \mid \nu_+) = 0$, which leads to $\mathbb{E}(s(c \mid P^{\nu_+}, k\text{-approval})) = Pr(c@k \mid \nu_+)$. Since ν_+ is constructed by inserting $(k-1)$ items before items in ν, $Pr(c@k \mid \nu_+) = Pr(c@1 \mid \nu)$. So $\mathbb{E}(s(c \mid P^{\nu_+}, k\text{-approval})) = Pr(c@1 \mid \nu)$. The answer to the REP-t problem has been reduced to the ESC problem. So the ESC problem is $FP^{\#P}$-hard, under k-approval rule.

In conclusion, the ESC problem is $FP^{\#P}$-complete, under k-approval rule. □

THEOREM 10. *Given a positional scoring rule r_m, a fully partitioned voting profile $P^{FP} = (v_1^{FP}, \ldots, v_n^{FP})$, and candidate w, determining $w \in MEW(r_m, P^{FP})$ is in $O(nm^2)$.*

PROOF. Any $v^{\text{FP}} \in P^{\text{FP}}$ defines a set of consecutive ranks in the linear extensions of v^{FP} for each of its partitions of candidates. Any candidate is equally likely to be positioned at these ranks. So the REP can be solved in $O(1)$ for any candidate. Thus, the MEW problem can be solved in $O(nm^2)$ by calculating the expected scores of all candidates. □

THEOREM 11. *Given a positional scoring rule r_m, a partial chain voting profile $P^{PC} = (v_1^{PC}, \ldots, v_n^{PC})$, and candidate w, determining $w \in MEW(r_m, P^{PC})$ is in $O(nm^2)$.*

PROOF. For any $v^{\text{PC}} \in P^{\text{PC}}$ and any candidate c, the $\Pr(c \rightarrow j \mid v^{\text{PC}})$ is proportional to the degree of freedom to place the rest of the candidates, after fixing c at rank j.

- If $c \notin v^{\text{PC}}$, this is a trivial case where c is equally likely to be placed at any rank, thus $\forall 1 \le j \le m, \Pr(c \rightarrow j \mid v^{\text{PC}}) = 1/m$.
- If $c \in v^{\text{PC}}$, let $K_l = |\{c' \mid c' >_{v^{\text{PC}}} c\}|$ be the number of items preferred to c by v^{PC} and $K_r = |\{c' \mid c >_{v^{\text{PC}}} c'\}|$ be the number of items less preferred to c by v^{PC}, then $\Pr(c \rightarrow j \mid v^{\text{PC}}) \propto \binom{j-1}{K_l} \cdot \binom{m-j}{K_r}$ where .

It takes $O(nm^2)$ to obtain the expected scores of all candidates and to determine whether w is a MEW. □

THEOREM 12. *Given a positional scoring rule r_m, a partially partitioned voting profile $P^{PP} = (v_1^{PP}, \ldots, v_n^{PP})$, and candidate w, determining $w \in MEW(r_m, P^{PP})$ is in $O(nm^2)$.*

PROOF. For any $v^{\text{PP}} \in P^{\text{PP}}$ and any candidate c, the $\Pr(c \rightarrow j \mid v^{\text{PP}})$ is proportional to the degree of freedom to place the rest of the candidates, after fixing c at rank j.

- If $c \notin v^{\text{PP}}$, this is a trivial case where c is equally likely to be placed at any rank, thus $\forall 1 \le j \le m, \Pr(c \rightarrow j \mid v^{\text{PC}}) = 1/m$.
- If $c \in v^{\text{PP}}$, let $K_l = |\{c' \mid c' >_{v^{\text{PP}}} c\}|$ be the number of items preferred to c by v^{PP}, $K_r = |\{c' \mid c >_{v^{\text{PP}}} c'\}|$ be the number of items less preferred to c by v^{PP}, and K_c be the number of items in the partition of c, then $\Pr(c \rightarrow j \mid v^{\text{PC}}) \propto \sum_{x=0}^{K_c-1} \binom{j-1}{K_l+x} \cdot \binom{m-j}{K_r+K_c-1-x}$ where x is the number of items from the same partition as c and placed to the left of c.

It takes $O(nm^2)$ to obtain the expected scores of all candidates and to determine whether w is a MEW. □

THEOREM 13. *Given positional scoring rule r_m, a RIM voting profile $P^{RIM} = (RIM_1, \ldots, RIM_n)$, and candidate w, determining $w \in MEW(r_m, P^{RIM})$ is in $O(nm^4)$.*

PROOF. Given any $RIM \in P^{\text{RIM}}$ and any candidate c, the $Pr(c \rightarrow j \mid RIM)$ for $j = 1, \ldots, m$ can be calculated by Algorithm 3 in $O(m^3)$. Algorithm 3 is a variant of RIMDP [13]. RIMDP calculates the marginal probability of a partial order over RIM via Dynamic Programming (DP). Algorithm 3 is simplified RIMDP in the sense that Algorithm 3 only tracks a particular item c, while RIMDP tracks multiple items to calculate the insertion ranges of items that satisfy the partial

order. Note that Algorithm 3 calculates all m different values of j simultaneously. So it takes $O(nm \cdot m^3) = O(nm^4)$ to obtain the expected scores of m candidates over n RIMs to determine MEW. □

THEOREM 14. *Given a positional scoring rule r_m, a voting profile $P^{RIM+TR} = \big((RIM_1, \tau_1^{(t_1,b_1)}), \ldots, (RIM_n, \tau_n^{(t_n,b_n)})\big)$, and candidate w, determining $w \in MEW(r_m, P^{RIM+TR})$ is in $O(nm^4)$.*

PROOF. Given any $(RIM, \tau^{(t,b)}) \in P^{\text{RIM+TR}}$, candidate c, and rank j, if c is in the top or bottom part of $\tau^{(t,b)}$, its rank has been fixed, which is a trivial case; If c is in the middle part of $\tau^{(t,b)}$, we just need to slightly modify Algorithm 3 to calculate $\Pr(c \rightarrow j \mid RIM, \tau^{(t,b)})$. Line 5 in Algorithm 3 enumerates values for j from 1 to i. The constraints made by $\tau^{(t,b)}$ limits this insertion range of item $\sigma(i)$. If $\sigma(i)$ is in the top or bottom part of $\tau^{(t,b)}$, its insertion position has been fixed by $\tau^{(t,b)}$ and the inserted items of the top and bottom parts of $\tau^{(t,b)}$ should be recorded as well by the state δ'; If $\sigma(i)$ is in the middle part of $\tau^{(t,b)}$, $\sigma(i)$ can be inserted into any position between the inserted top and bottom items.

Theoretically, the algorithm needs to track as many as $(t + b + 1)$ items. But $(t + b)$ items are fixed, which makes c the only item leading to multiple DP states. The complexity of calculating $\Pr(c \rightarrow j \mid RIM, \tau^{(t,b)})$ for all j values is $O(m^3)$. It takes $O(nm^4)$ to calculate the expected scores of all candidates across all voters to determine the MEW. □

THEOREM 15. *Given a positional scoring rule r_m, a voting profile $P^{MAL+FP} = \big((MAL_1, v_1^{FP}), \ldots, (MAL_n, v_n^{FP})\big)$, and candidate w, determining $w \in MEW(r_m, P^{MAL+FP})$ is in $O(nm^4)$.*

PROOF. Given any $(MAL(\sigma, \phi), v^{\text{FP}}) \in P^{\text{MAL+FP}}$, candidate c, and rank j, consider calculating $Pr(c \rightarrow j \mid \sigma, \phi, v^{\text{FP}})$. Let C_P denote the set of candidates in the same partition with c in v^{FP}. The relative orders between c and items out of C_P are already determined by v^{FP}. That is to say, for a non-trivial j value, $Pr(c \rightarrow j \mid \sigma, \phi, v^{\text{FP}})$ is proportional to the exponential of the number of disagreed pairs within C_P. So we can construct a new Mallows model $MAL'(\sigma', \phi)$ over C_P. It has the same ϕ as MAL and its reference ranking σ' is shorter than but consistent with σ. The $Pr(c \rightarrow j \mid MAL', v^{\text{FP}})$ for all non-trivial j values can be calculated in $O(|C_P|^3) < O(m^3)$ by Algorithm 3.

The MEW problem can be solved in $O(nm^4)$ by calculating the expected scores of all candidates across all voters to determine whether w is a MEW. □

B TRACTABILITY OVER RSM PROFILES

RSM [5] denoted by $RSM(\sigma, \Pi, p)$ is another generalization of the Mallows. It is parameterized by a reference ranking σ, a probability function Π where $\Pi(i, j)$ is the probability of the jth item selected at step i, and a probability function $p : \{1, \ldots, m-1\} \rightarrow [0, 1]$ where $p(i)$ is the probability that the ith selected item preferred to the remaining items. In contrast to the RIM that randomizes the item insertion position, the RSM randomized the item insertion order. In this paper, we use RSM as a ranking model, i.e., $p \equiv 1$ such that it only outputs rankings. This ranking version is named rRSM and denoted by $rRSM(\sigma, \Pi)$.

Algorithm 5 REP solver for rRSM

Input: Item c, rank k, rRSM(σ, Π)

Output: $\Pr(c@k \mid \sigma, \Pi)$

1: $\alpha_0 := |\{\sigma_i | \sigma_i >_\sigma c\}|$, $\beta_0 := |\{\sigma_i \mid c >_\sigma \sigma_i\}|$
2: $\mathcal{P}_0 := \{\langle \alpha_0, \beta_0 \rangle\}$ and $q_0(\langle \alpha_0, \beta_0 \rangle) := 1$
3: **for** $i = 1, \ldots, (k-1)$ **do**
4:    $\mathcal{P}_i := \{\}$
5:    **for** $\langle \alpha, \beta \rangle \in \mathcal{P}_{i-1}$ **do**
6:      **if** $\alpha > 0$ **then**
7:        Generate a new state $\langle \alpha', \beta' \rangle = \langle \alpha - 1, \beta \rangle$.
8:        **if** $\langle \alpha', \beta' \rangle \notin \mathcal{P}_i$ **then**
9:          $\mathcal{P}_i.add(\langle \alpha', \beta' \rangle)$
10:          $q_i(\langle \alpha', \beta' \rangle) := 0$
11:        **end if**
12:        $q_i(\langle \alpha', \beta' \rangle) \mathrel{+}= q_{i-1}(\langle \alpha, \beta \rangle) \cdot \sum_{j=1}^{\alpha} \Pi(i, j)$
13:      **end if**
14:      **if** $\beta > 0$ **then**
15:        Generate a new state $\langle \alpha', \beta' \rangle = \langle \alpha, \beta - 1 \rangle$.
16:        **if** $\langle \alpha', \beta' \rangle \notin \mathcal{P}_i$ **then**
17:          $\mathcal{P}_i.add(\langle \alpha', \beta' \rangle)$
18:          $q_i(\langle \alpha', \beta' \rangle) := 0$
19:        **end if**
20:        $q_i(\langle \alpha', \beta' \rangle) \mathrel{+}= q_{i-1}(\langle \alpha, \beta \rangle) \cdot \sum_{j=\alpha+2}^{\alpha+1+\beta} \Pi(i, j)$
21:      **end if**
22:    **end for**
23: **end for**
24: **return** $\sum_{\langle \alpha, \beta \rangle \in \mathcal{P}_{k-1}} q_{k-1}(\langle \alpha, \beta \rangle) \cdot \Pi(k, \alpha + 1)$

EXAMPLE 10. rRSM(σ, Π) with $\sigma = \langle a, b, c \rangle$ generates $\tau = \langle c, a, b \rangle$ as follows. Initialize $\tau_0 = \langle \rangle$. When $i = 1$, $\tau_1 = \langle c \rangle$ by selecting c with probability $\Pi(1, 3)$, making the remaining $\sigma = \langle a, b \rangle$. When $i = 2$, $\tau_2 = \langle c, a \rangle$ by selecting a with probability $\Pi(2, 1)$, making the remaining $\sigma = \langle b \rangle$. When $i = 3$, $\tau = \langle c, a, b \rangle$ by selecting b with probability $\Pi(3, 1)$. Overall, $\Pr(\tau \mid \sigma, \Pi) = \Pi(1, 3) \cdot \Pi(2, 1) \cdot \Pi(3, 1)$.

THEOREM 16. *Given a positional scoring rule r_m, an RSM voting profile $P^{rRSM} = (rRSM_1, \ldots, rRSM_n)$, and candidate w, determining $w \in MEW(r_m, P^{rRSM})$ is in $O(nm^4)$.*

PROOF. Given any rRSM $\in P^{rRSM}$, candidate c, and rank j, the $Pr(c@j \mid rRSM)$ is computed by Algorithm 5 in a fashion that is similar to Algorithm 3. This is also a Dynamic Programming (DP) approach. The states are in the form of $\langle \alpha, \beta \rangle$, where α is the number of items before c, and β is that after c in the remaining σ. For state $\langle \alpha, \beta \rangle$, there are $(\alpha + 1 + \beta)$ items in the remaining σ. Algorithm 5 only runs up to $i = (k - 1)$ (in line 3), since item c must be selected at step k and the rest steps do not change the rank of c anymore. Each step i generates at most $(i + 1)$ states, corresponding to $[0, \ldots, i]$ items are selected from items before c in the original σ. The complexity of Algorithm 5 is bounded by $O(m^2)$. It takes $O(nm^4)$ to obtain the expected scores of all candidates and to determine the MEW.

□

EXAMPLE 11. *Let* rRSM(σ, Π) *denote a RSM where* $\sigma = \langle \sigma_1, \sigma_2, \sigma_3, \sigma_4 \rangle$, *and* $\Pi = [[0.1, 0.3, 0.4, 0.2], [0.2, 0.5, 0.3], [0.3, 0.7], [1]]$. *Assume we are interested in* $\Pr(\sigma_2@3 \mid \sigma, \Pi)$, *the probability of* rRSM(σ, Π) *placing* σ_2 *at rank* 3.

- *Before running RSM, there is* $\alpha_0 = 1$ *item before* σ_2 *and* $\beta_0 = 2$ *items after* σ_2 *in* σ. *So the initial state is* $\langle \alpha_0, \beta_0 \rangle = \langle 1, 2 \rangle$, *and* $q_0(\langle 1, 2 \rangle) = 1$.
- *At step* $i = 1$, *the selected item can be either from* $\{\sigma_1\}$ *or* $\{\sigma_3, \sigma_4\}$. *So two new states are generated here.*
  - *The* σ_1 *is selected with probability* $\Pi(1, 1) = 0.1$, *which generates a new state* $\langle 0, 2 \rangle$, *and* $q_1(\langle 0, 2 \rangle) = q_0(\langle 1, 2 \rangle) \cdot \Pi(1, 1) = 0.1$.
  - *An item* $\sigma \in \{\sigma_3, \sigma_4\}$ *is selected with probability* $\Pi(1, 3) + \Pi(1, 4) = 0.6$, *which generates a new state* $\langle 1, 1 \rangle$, *and* $q_1(\langle 1, 1 \rangle) = q_0(\langle 1, 2 \rangle) \cdot 0.6 = 0.6$.

  So $\mathcal{P}_1 = \{\langle 0, 2 \rangle, \langle 1, 1 \rangle\}$, $q_1 = \{\langle 0, 2 \rangle \mapsto 0.1, \langle 1, 1 \rangle \mapsto 0.6\}$.
- *At step* $i = 2$, *iterate states in* $\mathcal{P}_1$.
  - *For state* $\langle 0, 2 \rangle$, *the selected item must be from the last two items in the remaining reference ranking. A new state* $\langle 0, 1 \rangle$ *is generated with probability* $\Pi(2, 2) + \Pi(2, 3) = 0.8$.
  - *For state* $\langle 1, 1 \rangle$, *the selected item is either the first or last item in remaining reference ranking. A new state* $\langle 0, 1 \rangle$ *is generated with probability* $\Pi(2, 1) = 0.1$, *and another state* $\langle 1, 0 \rangle$ *is generated with probability* $\Pi(2, 3) = 0.3$.

  So $\mathcal{P}_2 = \{\langle 0, 1 \rangle, \langle 1, 0 \rangle\}$ *and*
  □ $q_2(\langle 0, 1 \rangle) = q_1(\langle 0, 2 \rangle) \cdot 0.8 + q_1(\langle 1, 1 \rangle) \cdot 0.1 = 0.1 \cdot 0.8 + 0.6 \cdot 0.1 = 0.14$
  □ $q_2(\langle 1, 0 \rangle) = q_1(\langle 1, 1 \rangle) \cdot 0.3 = 0.6 \cdot 0.3 = 0.18$
- *At step* $i = 3$, *item* σ_2 *must be selected to meet the requirement. For each state* $\langle \alpha, \beta \rangle \in \mathcal{P}_2$, *the rank of* σ_2 *is* $(\alpha + 1)$ *in the corresponding remaining ranking. So* $\Pr(\sigma_2@3 | \sigma, \Pi) = q_2(\langle 0, 1 \rangle) \cdot \Pi(3, 1) + q_2(\langle 1, 0 \rangle) \cdot \Pi(3, 2) = 0.14 \cdot 0.3 + 0.18 \cdot 0.7 = 0.168$.