Computing Voting Rules with Elicited Incomplete Votes

Daniel Halpern
Harvard University
dhalpern@g.harvard.edu

Safwan Hossain
Harvard University
shossain@g.harvard.edu

Jamie Tucker-Foltz
Harvard University
jtuckerfoltz@gmail.com

Abstract

Motivated by the difficulty of specifying complete ordinal preferences over a large set of m candidates, we study voting rules that are computable by querying voters about $t < m$ candidates. Generalizing prior works that focused on specific instances of this problem, our paper fully characterizes the set of positional scoring rules that can computed for any $1 \leq t < m$, which, notably, does not include plurality. We then extend this to show a similar impossibility result for single transferable vote (elimination voting). These negative results are information-theoretic and agnostic to the number of queries. Finally, for scoring rules that are computable with limited-sized queries, we give parameterized upper and lower bounds on the number of such queries a deterministic or randomized algorithm must make to determine the score-maximizing candidate. While there is no gap between our bounds for deterministic algorithms, identifying the exact query complexity for randomized algorithms is a challenging open problem, of which we solve one special case.

1 Introduction

Traditional social choice frameworks typically assume that voting rules have access to each voter's complete ordinal preferences over all candidates. Indeed, this is seen in practice as well with the widening adoption of ranked-choice voting systems, requiring voters to submit such information (FairVote, 2024). Whether such information can actually be reliably elicited depends significantly on the context. If the number of candidates is small and voters have strong opinions, it may indeed be reasonable for them to provide a complete ranking. However, these assumptions do not hold in many scenarios. Primary elections in the United States, for example, routinely field large numbers of candidates, with many being unfamiliar to voters (Hirano and Jr, 2019).

A classic line of work in behavioral economics and psychology supports the premise that individuals struggle in such scenarios. In his seminal work, Schwartz (2004) puts forth the *paradox of choice*: individuals incur increased anxiety when faced with too many alternatives, which often leads them to take a default action, defer, or not participate altogether (Iyengar and Lepper, 2000). Recent literature has shown this phenomenon to hold specifically in the voting and social choice setting. Cunow et al. (2021, 2023) experimentally show that even increasing the number of candidates from 3 to 6 leads voters to spend less effort learning candidates' policy positions and instead rely on arbitrary heuristics. This voter frustration is also evidenced in practice: incomplete ballots are quite common, which are often completely exhausted under elimination voting long before the final candidate is elected Burnett and Kogan (2015).

These cognitive challenges are further exacerbated in contexts where the "candidates" are not politicians but *opinions*, of which there may be very many. Prime examples of such contexts can

be found in online platforms like *Polis* Small et al. (2021), *Remesh*,[1] *All Our Ideas*,[2] and *Loomio*,[3] which facilitate deliberation, build consensus, and ultimately aggregate opinions on a specific topic. These platforms allow users to both submit opinions as free-form text and vote on submissions of others. Polis, for example, was deployed by the government of Taiwan to gauge sentiment on the regulation of ride-share apps, ultimately leading to new legislation (Horton, 2018). Here, asking voters' opinions across all submissions can be far too time-consuming or downright infeasible. Instead, Polis makes a natural simplification by only showing a subset of opinions to each user. But what meaningful conclusions can be drawn from querying users over such limited data? And how should the platform select such queries?

Recent work by Halpern et al. (2023) studies this question in the context of approval votes, where the goal is to select a representative "committee" of size k. Voters from the population arrive randomly and can be presented with at most $t < m$ candidates (opinions) at a time, over which they can express their approval or disapproval. With sufficient arrivals, a committee selection algorithm can estimate the distribution of the population's approvals of any set of at most t candidates. In an idealized query model, it is assumed that the algorithm exactly obtains this distribution in a single query. Halpern et al. (2023) gives adaptive query algorithms to find committees satisfying the standard axioms of *extended justified representation* and *proportionality*. On the other hand, they also show information-theoretic lower bounds on the number of queries non-adaptive algorithms must make to guarantee a representative committee.

Our paper extends this framework to consider ordinal preferences, with the more classic goal of selecting a single winner (rather than a committee) under a given rule. There is a distribution over rankings of the m candidates representing the underlying voter preferences that is unknown to the algorithm. It may, however, *query* randomly arriving individuals about any subset of t candidates, where $t < m$; with suitable samples, the algorithm can determine the corresponding ranking distribution over this subset.[4] This leads to the fundamental question: what is the set of voting rules that are implementable with such small-sized queries chosen by the platform? Conceptually, this question asks about the axiomatic implications of cognitive barriers to preference elicitation in social choice.

1.1 Contributions

We begin our investigation with the ubiquitous plurality rule. Here, we find a surprisingly negative result: determining a plurality winner cannot be done using queries of any size $t < m$ (Theorem 1). That is, even if one has access to the distribution of the population's rankings over every $m - 1$ subset of the m candidates, it is still impossible to correctly identify who received the most first-place votes. In fact, even a randomized algorithm can only correctly choose a plurality winner with probability $\frac{1}{m}$ in the worst case, i.e., there are instances where, no matter which queries an algorithm makes, it can do no better than picking a candidate uniformly at random. The proof follows from a novel construction of pairs of profiles that have different plurality winners but induce the same distribution on any subset of $m - 1$ candidates. Section 3 is dedicated to explaining this construction, which forms the basis for all of the impossibility results in this paper.

Plurality is just one example of a *positional scoring rule*, whereby candidates receive points

[1] `https://www.remesh.ai/`

[2] `https://allourideas.org/`

[3] `https://www.loomio.com/`

[4] Clearly, having access to the exact distribution is strictly more informative than having to approximate it through repeated samples. Our negative results hold for this idealized setting and thus immediately apply to the weaker and more realistic query model.

corresponding to their rank position in each ballot, with the winner being the candidate with the most aggregated points Young (1975). While plurality requires queries of size m, it is known that another positional scoring rule, the Borda count, only requires pairwise margins to determine a winner and, hence, can be computed with queries of size 2. For general t-sized queries, one straightforward algorithm is to query every subset of size t and give each candidate a certain number of points depending on which of the t positions they appear. Under a different query model, Bentert and Skowron (2020) give a family of scoring rules for each size t for which this algorithm works.[5] A similar argument for the same families of rules holds for our model as well, which we prove in Lemma 4 for completeness, along with a more detailed comparison between our work and theirs. We then proceed to our second main result (Theorem 2) that shows that these rules are, in fact, the *only* ones that can be implemented with queries of size t. We thus obtain a complete characterization of all computable positional scoring rules under a limited query model and also give visualizations of this space. Our analysis is then extended to the *single transferrable vote (STV)* rule (also known as *instant-runoff voting*) wherein we find a negative result akin to Plurality: algorithms with limited query size ($t < m$) cannot correctly implement this rule. All of these negative results again hold not only for deterministic algorithms but for randomized ones that are correct with probability strictly better than $\frac{1}{m}$ (i.e. better than randomly guessing).

We next use our characterization to study the rules that *are* computable with small-sized queries. Specifically, for any scoring rule requiring t^*-sized queries, we lower bound the number of t-sized queries ($t^* \leq t < m$) needed to compute the winner under this rule (Theorem 4). If t and t^* are treated as constants, then $\Theta(m^{t^*})$ queries are needed in the worst case. This asymptotic bound holds even for randomized algorithms that are correct with probability $\frac{1}{m} + \varepsilon$ for any constant $\varepsilon > 0$. If on the other hand an algorithm covers a δ fraction of all t^*-subsets, we give an upper bound on the success probability. Although this is not tight, we exactly determine the optimal success probability for Borda count with $m = 3$ (Theorem 5) with a surprisingly intricate construction, and leave open the general case.

1.2 Related Work

Our work contributes to a line of literature on the information-theoretic aspects of voting and elections, specifically on what can be accomplished with incomplete information. There are a variety of lenses through which to study this problem. One line of work considers the communication complexity of computing various voting rules Conitzer and Sandholm (2005). Another studies when using incomplete votes can guarantee a candidate must or cannot be the winner, regardless of the missing information Konczak and Lang (2005); Xia and Conitzer (2011). Others consider different "approximation" objectives such as minimax regret Lu and Boutilier (2011) and distortion Procaccia and Rosenschein (2006). For a more complete survey, see chapter 10 of (Brandt et al., 2016).

More specific to our work are models where the partial information given is t-wise comparisons. The special case of $t = 2$ corresponds to only being given pairwise comparisons. All information about pairwise comparisons can be summarized in a *weighted tournament graph*, a widely studied object in social choice theory Brandt et al. (2016). For example, there is a classification of common voting rules into those that can be computed using just the tournament graph (such as *Borda Count*, *Minimax*, *Kemeny*, and *Copeland*) and those that cannot (Fishburn, 1977). Any of these weighted tournament solutions can trivially be computed with queries of size $t = 2$. Beyond information-theoretic results, there are even bounds on query complexity, such as how many queries to the

[5] In their model, each voter submits a ranking over a *randomly* selected set of t candidates.

tournament graph are needed to compute Condorcet winners Procaccia (2008). Our paper is a natural extension of this literature to the realm of more powerful t-wise comparison queries for $t > 3$.

Related, but technically incomparable, is when voters reveal a ranking of their top t candidates for a fixed value of t Oren et al. (2013); Filmus and Oren (2014). This was one of the two models studied by Bentert and Skowron (2020). The other has voters revealing a ranking over a *random* set of t candidates. Positive results here translate to our model, although negative ones do not. We describe this connection and their results more in-depth in Section 4.1.

Finally, note that we largely study information-theoretic impossibilities and thus do not focus on the randomized arrivals aspect; other works do consider the sample complexity of computing various rules (Dey and Bhattacharyya, 2015); however, they assume randomly sampled voters reveal their complete preferences over all candidates.

2 Preliminaries

2.1 Voter preferences

For a positive integer s, let $[s] := \{1, \ldots, s\}$. A *ranking* or *preference* over a set C of m *candidates* is a bijection $\sigma : [m] \to C$, where $\sigma(j)$ represents the j'th most-preferred candidate according to σ. We use the standard notation $a \succ_\sigma b$ to denote that a is preferred to b under σ, i.e., $\sigma^{-1}(a) < \sigma^{-1}(b)$, where σ^{-1} is the inverse mapping from candidates to rankings. We write $\mathcal{L}(C)$ to denote the set of all $m!$ rankings over the candidates in C. For a subset of candidates $S \subseteq C$, we write $\sigma|_S$ to denote the ranking σ restricted to the candidates in S, i.e., $\sigma|_S \in \mathcal{L}(S)$, with $\sigma|_S(j)$ being the j'th most preferred among those in S according to σ.

For a permutation over the candidates $\pi : C \to C$, we will write $\pi \circ \sigma$ for the ranking σ permuted by π, i.e., $(\pi \circ \sigma)(j) = \pi(\sigma(j))$. Of particular interest will be permutations that swap a single pair of candidates. For this reason, for two candidates a and b, we define π^{ab} to be the (a, b)-*transposition*, the permutation that swaps a and b, i.e., $\pi^{ab}(a) = b$, $\pi^{ab}(b) = a$, and $\pi^{ab}(c) = c$ for all $c \neq a, b$. Further, we will write $\sigma^{a \leftrightarrow b}$ for $\pi^{ab} \circ \sigma$.

A *preference profile* (or simply a *profile*) is a distribution $\boldsymbol{\sigma}$ over preferences $\mathcal{L}(C)$, representing the proportion of voters in the population that have each ranking. For example,

$$\Pr_{\sigma \sim \boldsymbol{\sigma}}[\sigma = a \succ b \succ c] = \frac{1}{5}$$

means that $\frac{1}{5}$ of the voters have the ranking $a \succ b \succ c$.[6] We denote by $\Pi(C)$ the space of all possible preference profiles over a candidate set C. For a profile $\boldsymbol{\sigma} \in \Pi(C)$, $\sigma \sim \boldsymbol{\sigma}$ denotes sampling a preference σ from distribution $\boldsymbol{\sigma}$. For a set of rankings $R \subseteq \mathcal{L}(C)$, we will also use the notation $\sigma \sim \text{Unif}(R)$ to denote sampling a ranking σ uniformly from R. We extend the restriction $\sigma|_S$, permutation $\pi \circ \sigma$, and transposition $\sigma^{a \leftrightarrow b}$ operations from rankings to profiles in the natural way. More formally, for restrictions, we write $\boldsymbol{\sigma}|_S$ to denote the distribution $\boldsymbol{\sigma}$ when restricted to candidates in S, i.e., $\boldsymbol{\sigma}|_S$ is an element of $\Pi(S)$ induced by sampling $\sigma \sim \boldsymbol{\sigma}$ and outputting $\sigma|_S$. For permutations, $\pi \circ \boldsymbol{\sigma}$ is the distribution induced by sampling $\sigma \sim \boldsymbol{\sigma}$ and outputting $\pi \circ \boldsymbol{\sigma}$. For transpositions, $\boldsymbol{\sigma}^{a \leftrightarrow b} = \pi^{ab} \circ \boldsymbol{\sigma}$. For a set S, we also use the notation $\text{Unif}(S)$ to denote the uniform distribution over elements of S.

[6] More often in social choice, a profile is a ranking assignment for a finite number of n agents. The distributional definition is essentially equivalent insofar as voter identity is not important (as is the case for all rules we study) while having the additional benefit of making our query model and proof techniques easier to understand. However, we could have equivalently used the more traditional definitions, and all of our results would still hold.

2.2 Voting rules

A *voting rule* f maps preference profiles to a set of winning candidates. If a candidate c is amongst the winners for a voting rule f, we refer to this candidate as an f-winner.

We will primarily focus on *positional scoring rules* (or simply *scoring rules*), which are a very practical and well-studied class. These are parameterized by a *scoring vector* $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_m) \in \mathbb{R}^m$.[7] Intuitively, a voter with ranking σ gives α_1 points to their first place candidate $\sigma(1)$, α_2 points to their second place candidate $\sigma(2)$, and so on, with the winner of the profile being the candidate with the most aggregated points. Written in our distributional notation, the score of a candidate c on profile $\boldsymbol{\sigma}$ is $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(c) := \mathbb{E}_{\sigma\sim\boldsymbol{\sigma}}[\alpha_{\sigma^{-1}(c)}]$, and the winning candidates on a profile $\boldsymbol{\sigma}$ are those with maximal score. Some common scoring rules include *plurality*, parameterized by $(1, 0, \ldots, 0)$, *veto*, parameterized by $(0, \ldots, 0, -1)$, and *Borda count*, parameterized by $(m-1, m-2, \ldots, 0)$. Since the plurality score will come up quite frequently, we will write $\mathrm{plu}_{\boldsymbol{\sigma}}(c)$ instead of using the $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(c)$ notation, and note that this simplifies to $\mathrm{plu}_{\boldsymbol{\sigma}}(c) = \Pr_{\sigma\sim\boldsymbol{\sigma}}[\sigma(1) = c]$. For conciseness, we will use $\boldsymbol{\alpha}$-winner, plurality winner, veto winner, and Borda winner to refer to f-winners of the scoring rule induced by $\boldsymbol{\alpha}$, plurality, veto, and Borda count, respectively. Note that $\boldsymbol{\alpha}$-winners are invariant under modifying $\boldsymbol{\alpha}$ via translation or multiplication by a positive constant, e.g., veto-winners (with vector $(0, \ldots, 0, -1)$) coincide with both $(1, \ldots, 1, 0)$-winners and $(\frac{1}{m-1}, \ldots, \frac{1}{m-1}, 0)$-winners.

In addition to scoring rules, we will also consider the rule *Single Transferable Vote (STV)*, which is defined as follows. For a profile $\boldsymbol{\sigma}$, if there is a single candidate, it returns that candidate. Otherwise, it chooses a candidate c with *minimal* plurality score, deletes them from the profile, and recurses on the rest. More formally, it chooses $c \in \mathrm{argmin}_{c'} \mathrm{plu}_{\boldsymbol{\sigma}}(c')$, and runs STV on $\boldsymbol{\sigma}|_{C\setminus\{c\}}$. This must eventually terminate, as a candidate is removed at each iteration. Further, for $m = 2$, this coincides with plurality. Note that in some cases, there are ties for the minimal score candidate. Hence, we will say that c is an STV winner if some sequence of valid eliminations results in c being the winner.

2.3 Query model

We consider (possibly randomized) algorithms that are allowed to adaptively submit queries to the underlying distribution $\boldsymbol{\sigma}$. For a fixed parameter t, each query consists of a subset of candidates $Q \subseteq C$ with $|Q| \leq t$ (referred to as a *query of size* t, or a *t-query* for short) and returns the distribution $\boldsymbol{\sigma}|_Q$. As mentioned previously, having access to the exact distribution is strictly more informative than one approximated by a random voter arriving (a voter $\sigma \sim \boldsymbol{\sigma}$ arrives, and the algorithm learns $\sigma|_Q$). All our impossibility results hold in this idealized setting and thus immediately apply to the more realistic one.

Two profiles $\boldsymbol{\sigma}^1$ and $\boldsymbol{\sigma}^2$ are said to be *t-indistinguishable* if for all subsets $Q \subseteq C$ with $|Q| \leq t$, $\boldsymbol{\sigma}^1|_Q = \boldsymbol{\sigma}^2|_Q$. That is, regardless of whether the profile is $\boldsymbol{\sigma}^1$ or $\boldsymbol{\sigma}^2$, any query Q with $|Q| \leq t$ will have the same response. Importantly, if $\boldsymbol{\sigma}^1$ and $\boldsymbol{\sigma}^2$ are t-indistinguishable, but $f(\boldsymbol{\sigma}^1) \cap f(\boldsymbol{\sigma}^2) = \emptyset$, then no t-query algorithm can always output an f-winner. Note that to check whether two profiles are t-indistinguishable, it suffices to check only queries Q of size exactly t, as if $Q' \subseteq Q$, then $\boldsymbol{\sigma}|_{Q'} = (\boldsymbol{\sigma}|_Q)|_{Q'}$, so $\boldsymbol{\sigma}^1|_Q = \boldsymbol{\sigma}^2|_Q$ implies $\boldsymbol{\sigma}^1|_{Q'} = \boldsymbol{\sigma}^2|_{Q'}$. Finally, notice that the special case of 2-indistinguishable is equivalent to $\boldsymbol{\sigma}^1$ and $\boldsymbol{\sigma}^2$ having the same *weighted tournament graph*, the complete-directed graph, where the nodes are candidates, and the weight on edge (a, b) is the proportion of voters that prefer a to b (Brandt et al., 2016). In this sense, the collection of distributions $\{\boldsymbol{\sigma}|_Q\}_{Q\subseteq C:|Q|=t}$ are a generalization of the weighted tournament graph to arbitrary

[7] Often, scoring vectors are restricted to be nonnegative and nonincreasing, but, for our purposes, it will be more convenient to allow for arbitrary vectors.

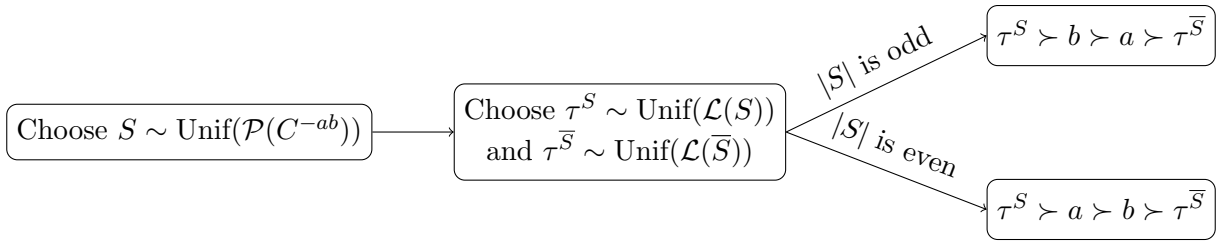

Figure 1: A process inducing the distribution over rankings for $\boldsymbol{\sigma}$.

$t \geq 2$ (e.g., for $t = 3$, rather than consisting of proportions of people that prefer a to b, it consists of the proportions of people that prefer a to b to c for all such combinations).

3 An Indistinguishable Construction

We begin with a construction of two indistinguishable profiles which will be used throughout our technical results.

Lemma 1. *For any m and pair of candidates $a, b \in C$, there is a profile $\boldsymbol{\sigma}$ such that (i) $\text{plu}_{\boldsymbol{\sigma}}(a) \neq \text{plu}_{\boldsymbol{\sigma}}(b)$ and (ii) $\boldsymbol{\sigma}$ and $\boldsymbol{\sigma}^{a \leftrightarrow b}$ are $(m-1)$-indistinguishable.*

Proof. Fix m, a, and b. Let $C^{-ab} = C \setminus \{a, b\}$ to be the set of remaining candidates. We have that $|C^{-ab}| = m - 2$. We define $\boldsymbol{\sigma}$ to be the distribution induced by the following random process. First, pick a set $S \subseteq C^{-ab}$ uniformly at random (i.e., each of the 2^{m-2} sets with equal probability). Formally, let $\mathcal{P}(C^{-ab})$ denote the power set of C^{-ab}, and we will sample $S \sim \text{Unif}(\mathcal{P}(C^{-ab}))$. Then choose a uniformly random ranking τ^S of the candidates in S and a uniformly random rankiong $\tau^{\overline{S}}$ of the candidates in $\overline{S} := C^{-ab} \setminus S$. Finally, if $|S|$ is even, output $\tau^S \succ a \succ b \succ \tau^{\overline{S}}$ and if $|S|$ is odd, output $\tau^S \succ b \succ a \succ \tau^{\overline{S}}$. This process is visually represented in Figure 1.

First, observe that it is indeed the case that $\text{plu}_{\boldsymbol{\sigma}}(a) \neq \text{plu}_{\boldsymbol{\sigma}}(b)$. The only way that either a or b could be ranked first is if the set S is empty. In that case, $|S|$ is even, so a will be ranked first. This does happen with positive probability ($1/2^{m-2}$), so the plurality score of a is positive. However, this can never happen for candidate b.

Next, we show that $\boldsymbol{\sigma}$ and $\boldsymbol{\sigma}^{a \leftrightarrow b}$ are $(m-1)$-indistinguishable. Note that a ranking from $\boldsymbol{\sigma}^{a \leftrightarrow b}$ can be sampled by running the process for $\boldsymbol{\sigma}$ but swapping the outcomes of the "$|S|$ is even" and "$|S|$ is odd" branches. Fix any $Q \subseteq C$ with $|Q| = m - 1$ and fix some ranking $\tau \in \mathcal{L}(Q)$. We want to show that observing τ is equiprobable under both $\boldsymbol{\sigma}|_Q$ and $\boldsymbol{\sigma}^{a \leftrightarrow b}|_Q$.

First, suppose Q contains only one of a or b. Without loss of generality, suppose it contains a. Since Q does not contain b, if we run the process of Figure 1 and pick τ^S and $\tau^{\overline{S}}$, regardless of whether we follow the "$|S|$ is even" or "$|S|$ is odd" branch the output when restricted to Q is the same. This is due to $\boldsymbol{\sigma}$ and $\boldsymbol{\sigma}^{a \leftrightarrow b}$ differing only in which branch to follow, and both branches are identical apart from the ordering of a and b, which occur consecutively in both. Thus, outputting τ is equiprobable in both restricted profiles.

Next, suppose Q contains both a and b. Again, since in any ranking of $\boldsymbol{\sigma}$ or $\boldsymbol{\sigma}^{a \leftrightarrow b}$, a and b always appear adjacent to each other, if a and b are not adjacent in τ, it occurs with probability 0 in both profiles. As such, suppose they are adjacent in τ, and without loss of generality, let $a \succ_\tau b$ (the other case is symmetric). Let T be the set of candidates ranked above a in τ, and L be the set of candidates ranked below b. Therefore, $Q = T \cup L \cup \{a, b\}$. Note to sample σ with $\sigma|_Q = \tau$ under either $\boldsymbol{\sigma}$ or $\boldsymbol{\sigma}^{a \leftrightarrow b}$, it must be the case that when selecting S, $S \cap Q = T$. Further, conditioned on

this, the probability of getting both T and L in the order matching τ is simply $\frac{1}{|T|!|L|!}$. Finally, the order of a and b will match τ exactly when $|S|$ is even. Putting this together, we have that the probability of sampling σ with $\sigma|_Q = \tau$ under $\boldsymbol{\sigma}$ is exactly

$$\frac{1}{|T|!|L|!} \Pr[(S \cap Q = T) \wedge (|S| \text{ is even})].$$

For $\boldsymbol{\sigma}^{a \leftrightarrow b}$, it is identical but with "even" switched with "odd." Hence, to show equality, it suffices to show that:

$$\Pr[(S \cap Q = T) \wedge (|S| \text{ is even})] = \Pr[(S \cap Q = T) \wedge (|S| \text{ is odd})].$$

This is equivalent to showing that

$$\Pr[|S| \text{ is even} \mid S \cap Q = T] = \Pr[|S| \text{ is odd} \mid S \cap Q = T].$$

Let us now consider how to sample S from the conditional distribution given $S \cap Q = T$. Note that S satisfies this exactly when $T \subseteq S$ and $L \cap S \neq \emptyset$. Hence, to sample such an S, we can sample S' uniformly from $C^{-ab} \setminus (T \cup L)$ and output $S' \cup T$. By the assumptions that $|Q| = m - 1 < |C|$ and $\{a, b\} \subseteq Q$, we have that $C^{-ab} \setminus (T \cup L)$ is nonempty. It is known that when sampling a subset uniformly at random from a non-empty set, it is equiprobable whether it is of even or odd carnality.[8] Hence, $|S'|$ is equally likely to be even or odd, which implies that $|S|$ conditioned on $S \cap Q = T$ is equally likely to be even or odd, as needed. □

4 Uncomputable Voting Rules

In this section, we prove that there is a family of voting rules that *cannot* be computed using limited query sizes. In particular, we give a complete characterization of which positional scoring rules can be computed in our query model for each choice of t. In addition, we analyze the widely adopted STV rule. We begin, however, with two lemmas, which together give sufficient conditions for a scoring rule to *not* be computable using limited queries. The second will also be helpful for the STV impossibility.

Lemma 2. *Fix a vector $\boldsymbol{\alpha}$, and suppose there exists a profile $\boldsymbol{\sigma}$ and two candidates a and b such that $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(a) \neq \mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(b)$ yet $\boldsymbol{\sigma}$ and $\boldsymbol{\sigma}^{a \leftrightarrow b}$ are t-indistinguishable. Then, there exists a family of profiles, $\{\boldsymbol{\sigma}^c\}_{c \in C}$ such that all are t-indistinguishable from one another, but each candidate $c \in C$ uniquely maximizes $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^c}$.*

Proof. Let $\boldsymbol{\sigma}$, a, and b be the profile and candidates satisfying the lemma conditions. Fix a candidate c. We describe the distribution $\boldsymbol{\sigma}^c$ as follows. Assume without loss of generality that $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(a) > \mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(b)$. We first sample a permutation over the candidates π uniformly at random. If $\pi(b) = c$, we return a ranking sampled from $\pi \circ \boldsymbol{\sigma}^{a \leftrightarrow b}$; otherwise, we return a ranking sampled from $\pi \circ \boldsymbol{\sigma}$. A visual representation can be found in Figure 2. We will compare this constructed profile with respect to another, which we call $\boldsymbol{\sigma}^{\mathrm{unif}}$. The profile $\boldsymbol{\sigma}^{\mathrm{unif}}$ is constructed by picking a permutation π uniformly at random and returning a sample $\sigma \sim \pi \circ \boldsymbol{\sigma}$ regardless of π, shown in Figure 3.

We first claim that $\boldsymbol{\sigma}^{\mathrm{unif}}$ is in fact the uniform distribution over $\mathcal{L}(C)$. Indeed, an equivalent way of sampling from $\boldsymbol{\sigma}^{\mathrm{unif}}$ is to sample in the reverse order, first sampling $\sigma \sim \boldsymbol{\sigma}$ and then applying

[8] Fix an element x. For any subset of the remaining elements S, it is equally likely to pick S and $S \cup \{x\}$. One of these has even parity and the other has odd.

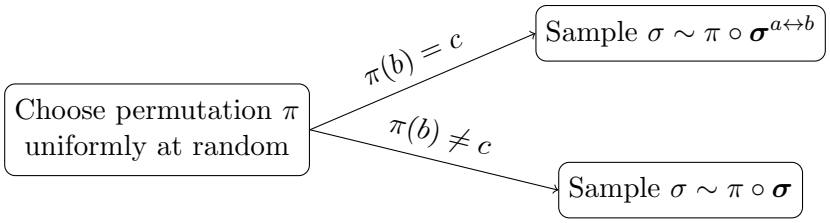

Figure 2: A process inducing the distribution over rankings for $\boldsymbol{\sigma}^c$.

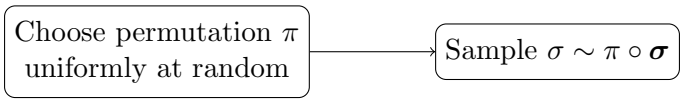

Figure 3: A process inducing the distribution over rankings for $\boldsymbol{\sigma}^{\text{unif}}$.

a randomly selected permutation π to σ. This makes it clear that $\boldsymbol{\sigma}^{\text{unif}}$ is a mixture over uniform distributions and is hence uniform.

We can think of both $\boldsymbol{\sigma}^c$ and $\boldsymbol{\sigma}^{\text{unif}}$ as a mixture of $m!$ different profiles, the one associated with each choice of π. Abusing notation slightly, we will write $\boldsymbol{\sigma}^c_\pi$ and $\boldsymbol{\sigma}^{\text{unif}}_\pi$ for the profile sampled from when we had the permutation π. More concretely,

$$\boldsymbol{\sigma}^c_\pi = \begin{cases} \pi \circ \boldsymbol{\sigma} & \text{if } \pi(b) \neq c \\ \pi \circ \boldsymbol{\sigma}^{a \leftrightarrow b} & \text{if } \pi(b) = c \end{cases},$$

while $\boldsymbol{\sigma}^{\text{unif}}_\pi = \pi \circ \boldsymbol{\sigma}$ for all π.

Note that the scores $\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\text{unif}}}(c')$ are equal for all c' by symmetry. We will show both (i) $\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^c}(c) > \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\text{unif}}}(c)$ while $\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^c}(c') \leq \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\text{unif}}}(c')$ for all $c' \neq c$ and (ii) $\boldsymbol{\sigma}^c$ is t-indistinguishable from $\boldsymbol{\sigma}^{\text{unif}}$. The first shows that the c is the unique score maximizer, and the second shows that all the constructed profiles are t-indistinguishable from each other since each is t-indistinguishable from $\boldsymbol{\sigma}^{\text{unif}}$.

For the first, consider the difference $\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^c}(c') - \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\text{unif}}}(c')$ for an arbitrary candidate c'. Because scores are linear, we can split them across our mixture definitions to get

$$\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^c}(c') = \mathbb{E}_\pi[\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^c_\pi}(c')] \text{ and } \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\text{unif}}}(c') = \mathbb{E}_\pi[\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\text{unif}}_\pi}(c')].$$

Plugging this into the difference, by linearity of expectation, we have that

$$\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^c}(c') - \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\text{unif}}}(c') = \mathbb{E}_\pi[\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^c_\pi}(c') - \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\text{unif}}_\pi}(c')].$$

Now, for any π with $\pi(b) \neq c$, the difference inside the expectation is 0 because $\boldsymbol{\sigma}^c_\pi = \boldsymbol{\sigma}^{\text{unif}}_\pi$. Fix some π such that $\pi(b) = c$. In this case, $\boldsymbol{\sigma}^c_\pi = \pi \circ \boldsymbol{\sigma}^{a \leftrightarrow b}$ while $\boldsymbol{\sigma}^{\text{unif}}_\pi = \pi \circ \boldsymbol{\sigma}$. Note that

$$\text{sc}^{\boldsymbol{\alpha}}_{\pi \circ \boldsymbol{\sigma}^{a \leftrightarrow b}}(c') = \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{a \leftrightarrow b}}(\pi^{-1}(c')) = \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(\pi^{ab}(\pi^{-1}(c')))$$

where the first equality moves the application of π and the second does the same, using the fact that $\boldsymbol{\sigma}^{a \leftrightarrow b}$ is simply achieved by the π^{ab} permutation (which is its own inverse). We also have:

$$\text{sc}^{\boldsymbol{\alpha}}_{\pi \circ \boldsymbol{\sigma}}(c') = \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(\pi^{-1}(c)).$$

Hence, this difference is only nonzero if $\pi^{ab}(\pi^{-1}(c')) \neq \pi^{-1}(c')$, i.e., $\pi^{-1}(c') \in \{a, b\}$. If $\pi^{-1}(c') = a$, then the difference is $\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(b) - \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(a)$, while if $\pi^{-1}(c') = b$, then the difference is $\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(a) - \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(b)$.

By the lemma assumptions, $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(a) > \mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(b)$; hence, the difference is positive in the first case and negative in the second. To summarize, $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^c_\pi}(c') - \mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\mathrm{unif}}_\pi}(c')$ is nonzero only when $\pi(b) = c$ and either $\pi(b) = c'$ (in which case it is positive) or $\pi(a) = c'$ (in which case it is negative). From this description, we see that when $c' = c$, this can *only* take on positive values (and does whenever $\pi(b) = c$), and when $c' \neq c$, this can *only* take on negative values (and does whenever $\pi(b) = c$ and $\pi(a) = c'$). Hence, $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^c}(c') - \mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\mathrm{unif}}}(c')$ is positive when $c' = c$ and negative when $c' \neq c$, as needed.

To complete the proof, we show that $\boldsymbol{\sigma}^c$ and $\boldsymbol{\sigma}^{\mathrm{unif}}$ are t-indistinguishable. To that end, fix $Q \subseteq C$ of size t. The key fact we will use is that if $\boldsymbol{\sigma}^1$ and $\boldsymbol{\sigma}^2$ are t-indistinguishable, then $\pi \circ \boldsymbol{\sigma}^1$ and $\pi \circ \boldsymbol{\sigma}^2$ are t-indistinguishable for all π. This implies that $(\pi \circ \boldsymbol{\sigma}^{a \leftrightarrow b})|_Q = (\pi \circ \boldsymbol{\sigma})|_Q$ for all π. Therefore, both $\boldsymbol{\sigma}^c|_Q$ and $\boldsymbol{\sigma}^{\mathrm{unif}}|_Q$ are mixtures over the exact same $m!$ distributions, and are hence equal, as needed. □

Lemma 3. *Fix a voting rule f and suppose there are m profiles that are all t-indistinguishable, but each has a distinct singleton f-winner. Then, for all (possibly randomized) algorithms A which on input profile $\boldsymbol{\sigma}$ can make queries of size at most t to $\boldsymbol{\sigma}$ and output a candidate, there is a profile $\boldsymbol{\sigma}^*$ with a unique f-winning candidate c, such that the probability A outputs c on $\boldsymbol{\sigma}^*$ is at most $1/m$.*

Proof. Let $\{\boldsymbol{\sigma}^c\}_{c \in C}$ denote the set of m profiles that are all t-indistinguishable, and let c be the f-winner on profile $\boldsymbol{\sigma}^c$. Note that an algorithm run on any of these profiles will receive the exact same responses to queries. Hence, its output must be identical for all of them. There must be some candidate a^* which it outputs with probability at most $1/m$, and hence, $\boldsymbol{\sigma}^{a^*}$ satisfies the desired properties. □

Taken together, these two lemmas outline sufficient conditions wherein limited query algorithms cannot compute the winner. Combined with the construction presented in Lemma 1, it allows us to immediately conclude the following result about the impossibility of computing a plurality winner with any restricted query size.

Theorem 1. *For any number of candidates $m \geq 2$, for all $t < m$, no randomized t-query algorithm can always output a plurality winner with probability more than $1/m$.*

Proof. In Lemma 1, we proved the existence of a profile $\boldsymbol{\sigma}$ that has distinct plurality scores for two candidates a, b, and is $m - 1$ indistinguishable from its transposed profile $\boldsymbol{\sigma}^{a \leftrightarrow b}$. As such, we can apply Lemma 2 and show the existence of m profiles with distinct plurality winners that are all $m - 1$ indistinguishable. Applying Lemma 3 on these m profiles directly gives us the desired result for plurality. □

While the result for plurality follows immediately, the question of computing arbitrary scoring rules with a limited query size requires a more involved approach. This is tackled next.

4.1 Characterization of scoring rules

We now consider an arbitrary scoring vector $\boldsymbol{\alpha}$ and determine the exact query size needed to compute an $\boldsymbol{\alpha}$-winner. Fix $1 \leq t \leq m$, and fix a candidate set C of size m. The following notation will be convenient for discussing arbitrary positional scoring rules. For any preference profile $\boldsymbol{\sigma} \in \Pi(C)$ and candidate $c \in C$, we define $\mathrm{pos}(\boldsymbol{\sigma}, c) \in \mathbb{R}^m$ to be the vector of positional occurrences of candidate c across all rankings in profile $\boldsymbol{\sigma}$. More formally, for each $j \in [m]$,

$$\mathrm{pos}(\boldsymbol{\sigma}, c)_j := \Pr_{\sigma \sim \boldsymbol{\sigma}}[\sigma(j) = c].$$

Thus, a positional scoring rule is a voting rule parameterized by scoring vector $\boldsymbol{\alpha}$ which selects a candidate c maximizing $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(c) = \boldsymbol{\alpha} \cdot \mathrm{pos}(\boldsymbol{\sigma}, c)$. Now for each $k \in [t]$, consider the scoring vector $\boldsymbol{\alpha}^k \in \mathbb{R}^m$ given by:

$$\alpha^k_j = \binom{j-1}{k-1}\binom{m-j}{t-k}.$$

We define the subspace spanned by these k vectors as $R_{m,t}$. Formally,

$$R_{m,t} := \mathrm{span}(\boldsymbol{\alpha}^1, \boldsymbol{\alpha}^2, \ldots, \boldsymbol{\alpha}^t) \subseteq \mathbb{R}^m.$$

To build some intuition for this space, it can be shown that for $t \geq 2$, $R_{m,t}$ contains the commonly used Borda score, corresponding to $\boldsymbol{\alpha} = (m-1, m-2, \ldots, 0)$ (see Figure 5 for a visualization). We next show that any scoring rule in this space can be computed with t-sized queries and give a constructive algorithm. This is essentially shown in Theorem 1 of Bentert and Skowron (2020) but under a different model. For completeness, we present the proof for our setting below:

Lemma 4 (Theorem 1 of Bentert and Skowron (2020))**.** *For any $t \leq m$ and $\boldsymbol{\alpha} \in R_{m,t}$, given $\boldsymbol{\sigma} \in \Pi(C)$ and a candidate $c \in C$, it is possible to compute $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(c)$ with queries of size t.*

Proof. We will show that for each $k \leq t$, it is possible to compute $\mathrm{sc}^{\boldsymbol{\alpha}^k}_{\boldsymbol{\sigma}}(c)$. For any $\boldsymbol{\alpha} \in R_{m,t}$, since $\boldsymbol{\alpha} = \lambda_1\boldsymbol{\alpha}^1 + \cdots + \lambda_t\boldsymbol{\alpha}^t$ for some scalars $\lambda_1, \ldots, \lambda_t$, by linearity, $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(c) = \lambda_1 \mathrm{sc}^{\boldsymbol{\alpha}^1}_{\boldsymbol{\sigma}}(c) + \cdots + \lambda_t \mathrm{sc}^{\boldsymbol{\alpha}^t}_{\boldsymbol{\sigma}}(c)$. Hence, as long as we can compute the score for each of the basis vectors, we can do so for any vector in the span.

Fix a candidate c and index $j \in [m]$, and let σ be any ranking such that $\sigma(j) = c$. Since there are m candidates in C, there are $\binom{m}{t}$ possible sets $S \subseteq C$ of size t. The number of such subsets S for which the restricted ranking $\sigma|_S$ puts c in a position k is given by $\binom{j-1}{k-1}\binom{m-j}{t-k}$, since S must contain the special candidate c, along with $k-1$ of the $j-1$ candidates ranked above c in σ, and $t-k$ of the $m-j$ candidates ranked below c in σ. Thus, the probability that $\sigma|_S(k) = c$ for a uniformly random subset S of size t is

$$\frac{\binom{j-1}{k-1}\binom{m-j}{t-k}}{\binom{m}{t}}.$$

Consider the following algorithm. For any input preference profile $\boldsymbol{\sigma} \in \Pi(C)$ and candidate $c \in C$, we first compute the probability that, if we draw a set S of t distinct candidates uniformly at random from C and draw a preference $\sigma \sim \boldsymbol{\sigma}$, candidate c will be at position k in the ranking σ restricted to S - i.e the event $\sigma|_s(k) = c$. Clearly, this probability can be determined with queries of size t by brute-forcing over all subsets S of size t. We then output this probability multiplied by $\binom{m}{t}$. Observe that we may write the output of this algorithm as

$$\begin{aligned}
\binom{m}{t} \Pr_{\substack{S \subseteq C,\ |S|=t \\ \sigma \sim \boldsymbol{\sigma}}}[\sigma|_S(k) = c] &= \binom{m}{t} \sum_{j=1}^{t} \Pr_{\sigma \sim \boldsymbol{\sigma}}[\sigma(j) = c] \Pr_{\substack{S \subseteq C,\ |S|=t \\ \sigma \sim \boldsymbol{\sigma}}}[\sigma|_S(k) = c \mid \sigma(j) = c] \\
&= \binom{m}{t} \sum_{j=1}^{t} \mathrm{pos}(\boldsymbol{\sigma}, c)_j \frac{\binom{j-1}{k-1}\binom{m-j}{t-k}}{\binom{m}{t}} \qquad \text{(by the calculation above)} \\
&= \sum_{j=1}^{t} \boldsymbol{\alpha}^k_j \mathrm{pos}(\boldsymbol{\sigma}, c)_j = \boldsymbol{\alpha}^k \cdot \mathrm{pos}(\boldsymbol{\sigma}, c) = \mathrm{sc}^{\boldsymbol{\alpha}^k}_{\boldsymbol{\sigma}}(c). \qquad \square
\end{aligned}$$

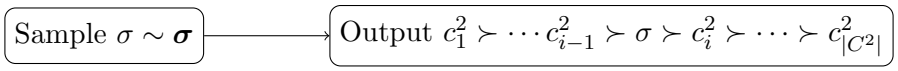

Figure 4: A process inducing $\boldsymbol{\sigma}^i$. This is parameterized by two disjoint sets, C^1 and C^2 and two candidates $a, b \in C^1$. The profile $\boldsymbol{\sigma} \in \Pi(C^1)$ satisfies the conditions of Lemma 1 with C_1, a, and b. The indexing $c_1^2 \succ \cdots \succ c_{|C^2|}^2$ is an arbitrary order of the candidates in C^2.

We now move to our main result which generalizes Theorem 1 by proving that $R_{m,t}$is exactly the space of all scoring rules computable with limited queries of size t, thus giving a complete characterization.

Theorem 2. *For any number of candidates $m \geq 2$, any $1 \leq t \leq m$, and any vector $\boldsymbol{\alpha} \in \mathbb{R}^m$*

1. *If $\boldsymbol{\alpha} \in R_{m,t}$ then there a t-query algorithm that always outputs a candidate c maximizing $\mathrm{sc}_{\boldsymbol{\sigma}}^{\boldsymbol{\alpha}}(c)$ on any input profile $\boldsymbol{\sigma}$.*
2. *If $\boldsymbol{\alpha} \notin R_{m,t}$, then no randomized t-query algorithm can always output an $\boldsymbol{\alpha}$-winner with probability more than $\frac{1}{m}$.*

Statement (1) is the easier part and follows immediately from Lemma 4. Our main focus here is statement (2), which leverages the construction from Section 3. We define a sequence of $m - t$ profiles as follows. We partition the candidate set C into two disjoint pieces, a set C_1 of size $t + 1$ with two distinguished candidates $a, b \in C_1$, and a set C_2 of size $m - t - 1$. Let $\boldsymbol{\sigma} \in \Pi(C_1)$ be a profile satisfying the conditions of Lemma 1 with C_1, a and b, i.e., $\mathrm{plu}_{\boldsymbol{\sigma}}(a) \neq \mathrm{plu}_{\boldsymbol{\sigma}}(b)$ and $\boldsymbol{\sigma}$ and $\boldsymbol{\sigma}^{a \leftrightarrow b}$ are t-industinguishable. Let $c_1^2 \succ \cdots \succ c_{|C_2|}^2$ be an arbitrary order of the candidates in C^2. For each $i \in [m - t]$, we extend $\boldsymbol{\sigma}$ to a profile on all m candidates $\boldsymbol{\sigma}^i \in \Pi(C)$ by inserting it in the C^2 order after the first $i - 1$ candidates. A visual representation of this can be found in Figure 4. Note that, for each $i \in [m - t]$, we clearly have that $\boldsymbol{\sigma}^i$ is t-indistinguishable from $(\boldsymbol{\sigma}^i)^{a \leftrightarrow b}$ since $\boldsymbol{\sigma}$ is t-indistinguishable from $\boldsymbol{\sigma}^{a \leftrightarrow b}$, as for any t-query query Q, the candidates from C_2 are all in the same order, and $\boldsymbol{\sigma}|_{Q \cap C_1} = \boldsymbol{\sigma}^{a \leftrightarrow b}|_{Q \cap C_1}$ because $|Q \cap C_1| \leq t$. We now show for any vector not in the span, the score for a and b on one of these t-indistinguishable profiles is not the same.

Lemma 5. *For any $\boldsymbol{\alpha} \notin R_{m,t}$, there exists some $i \in [m - t]$ such that $\mathrm{sc}_{\boldsymbol{\sigma}^i}^{\boldsymbol{\alpha}}(a) \neq \mathrm{sc}_{\boldsymbol{\sigma}^i}^{\boldsymbol{\alpha}}(b)$.*

Proof. For each $i \in [m - t]$, we define the vector

$$s^i := \mathrm{pos}(\boldsymbol{\sigma}^i, a) - \mathrm{pos}(\boldsymbol{\sigma}^i, b) \in \mathbb{R}^m.$$

Suppose toward a contradiction that candidates a and b have the same scores in each $\boldsymbol{\sigma}^i$ according to $\boldsymbol{\alpha}$. This implies that

$$\boldsymbol{\alpha} \cdot \mathrm{pos}(\boldsymbol{\sigma}^i, a) = \boldsymbol{\alpha} \cdot \mathrm{pos}(\boldsymbol{\sigma}^i, b).$$

Since the score under each basis vector $\mathrm{sc}_{\boldsymbol{\sigma}_i}^{\boldsymbol{\alpha}^k}$ for $k \in [t]$ can be computed with queries of size t by Lemma 4 and each $\boldsymbol{\sigma}^i$ is t-indistinguishable from $(\boldsymbol{\sigma}^i)^{a \leftrightarrow b}$, we know that a and b must have the same scores in $\mathrm{sc}_{\boldsymbol{\sigma}^i}^{\boldsymbol{\alpha}^k}$ for each $\boldsymbol{\sigma}^i$ and $k \in [t]$ as well. In other words, for each $0 \leq k \leq t$, and for each $i \in [m - t]$, we have

$$\boldsymbol{\alpha}^k \cdot \mathrm{pos}(\boldsymbol{\sigma}^i, a) = \boldsymbol{\alpha}^k \cdot \mathrm{pos}(\boldsymbol{\sigma}^i, b).$$

We can therefore equivalently write for all $\boldsymbol{\alpha}' \in \{\boldsymbol{\alpha}, \boldsymbol{\alpha}^1, \boldsymbol{\alpha}^2, \ldots, \boldsymbol{\alpha}^t\}$ and all $i \in [m - t]$,

$$\boldsymbol{\alpha}' \cdot s^i = 0.$$

Consider the following two subspaces of $\mathbb{R}^m$:

$$R := \text{span}(\boldsymbol{\alpha}, \boldsymbol{\alpha}^1, \boldsymbol{\alpha}^2, \ldots, \boldsymbol{\alpha}^t)$$
$$S := \text{span}(s^1, s^2, \ldots, s^{m-t})$$

What we have just shown is that the vectors generating R are each orthogonal to the vectors generating S, so the spaces are orthogonal to each other. Therefore, $\dim(R) + \dim(S) \leq m$. To obtain a contradiction, we will show that $\dim(R) = t + 1$ and $\dim(S) = m - t$.

First consider R. For each $k \in [t]$, observe that the j^{th} entry of $\boldsymbol{\alpha}^k$ is zero for all $j < k$, since $\binom{j-1}{k-1} = 0$. On the other hand, the k^{th} entry of $\boldsymbol{\alpha}^k$ is

$$\binom{k-1}{k-1}\binom{m-k}{t-k} = \binom{m-k}{t-k} > 0$$

because $k \leq t \leq m$. Thus, arranging the vectors $\boldsymbol{\alpha}^1, \boldsymbol{\alpha}^2, \ldots, \boldsymbol{\alpha}^t$ as the rows of a matrix, we have a triangle of zeros below nonzero diagonal entries of the form

$$M = \begin{pmatrix} \boldsymbol{\alpha}^1 \\ \boldsymbol{\alpha}^2 \\ \vdots \\ \boldsymbol{\alpha}^t \end{pmatrix} = \begin{pmatrix} \boldsymbol{\alpha}^1_1 > 0 & \boldsymbol{\alpha}^1_2 & \cdots & \boldsymbol{\alpha}^1_t \\ 0 & \boldsymbol{\alpha}^2_2 > 0 & \cdots & \boldsymbol{\alpha}^2_t \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{\alpha}^t_t > 0 \end{pmatrix}$$

Clearly, M has full rank, so the $\boldsymbol{\alpha}^k$ vectors are all linearly independent. Furthermore, $\boldsymbol{\alpha}$ is independent of all of these t basis vectors, since we are assuming $\boldsymbol{\alpha} \notin R_{m,t}$. Hence, the $t + 1$ vectors generating R are independent, so R has dimension $t + 1$.

Now consider S. For each $i \in [m - t]$, there is no way that either of the two special candidates a and b could be ranked above position i in any preference σ in the support of $\boldsymbol{\sigma}^i$, since a and b belong to C_1, not C_2. Thus, for each $j < i$, we have

$$s^i_j = \text{pos}(\boldsymbol{\sigma}^i, a)_j - \text{pos}(\boldsymbol{\sigma}^i, b)_j = 0 - 0 = 0.$$

On the other hand, a and b occur exactly at position i in $\boldsymbol{\sigma}^i$ with the same probability that they occur first in $\boldsymbol{\sigma}$ satisfying Lemma 1. By Lemma 1, the probability of a occurring at the first position is different than the probability of b occurring at the first position. In other words,

$$s^i_i = \text{pos}(\boldsymbol{\sigma}^i, a)_i - \text{pos}(\boldsymbol{\sigma}^i, b)_i = \text{pos}(\boldsymbol{\sigma}, a)_1 - \text{pos}(\boldsymbol{\sigma}, b)_1 = \text{plu}_{\boldsymbol{\sigma}}(a) - \text{plu}_{\boldsymbol{\sigma}}(b) \neq 0.$$

Thus, by the same triangle-of-zeros argument as before, we conclude that the s^i vectors are independent, so $\dim(S) = m - t$. □

Proof of Theorem 2. First suppose $\boldsymbol{\alpha} \in R_{m,t}$. By Lemma 4, on input $\boldsymbol{\sigma}$, we can compute $\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}(c)$ for each candidate c using queries of size t. Therefore, we can output a candidate maximizing this score.

On the other hand, suppose $\boldsymbol{\alpha} \notin R_{m,t}$. Then let $\boldsymbol{\sigma}^i$ be as in Lemma 5. Since $\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^i}(a) \neq \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^i}(b)$, yet $\boldsymbol{\sigma}^i$ is indistinguishable from $(\boldsymbol{\sigma}^i)^{a \leftrightarrow b}$, the conclusion follows immediately from Lemmas 2 and 3. □

To make this space of computable scoring rules interpretable, we visualize the subspaces $R_{m,t}$ for $m = 4$. Since translating a scoring vector by a constant and scaling by a positive value do not affect the induced rule, all rules in $\mathbb{R}^4$ can be normalized such that they are contained within

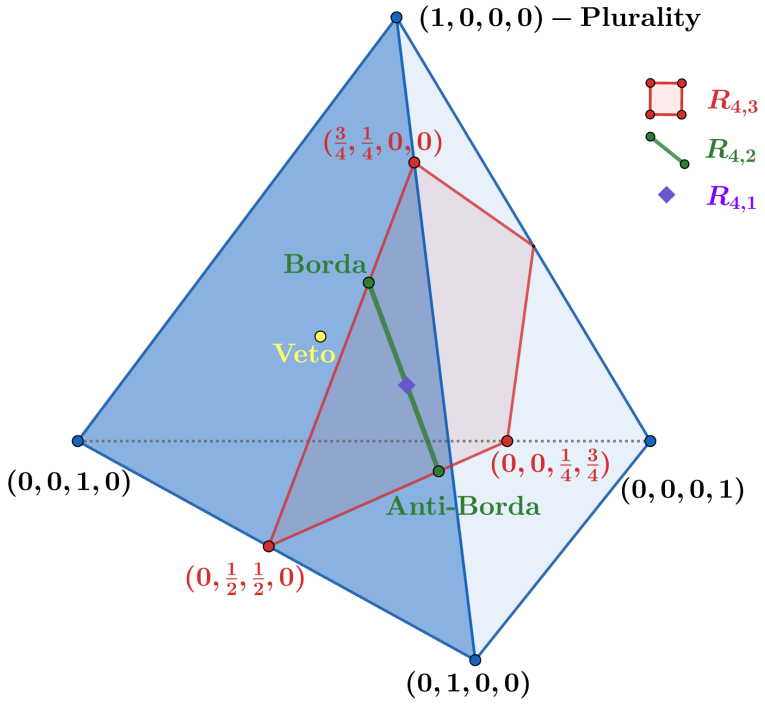

Figure 5: The space of positional scoring rules for $m = 4$ candidates. The corners represent rules that give all weight to a single position, with the top being Plurality. The red kite-shaped region is the 2-dimensional subspace $R_{4,3}$ spanned by the vectors $\boldsymbol{\alpha}^1 = (3, 1, 0, 0)$, $\boldsymbol{\alpha}^2 = (0, 1, 1, 0)$, and $\boldsymbol{\alpha}^3 = (0, 0, 1, 3)$, which are normalized in the simplex as the three red points at the corners of the kite. As we have shown, this subspace does not contain Plurality. Nested within this subspace is $R_{4,2}$, the green 1-dimensional subspace spanned by the Borda and Anti-Borda scoring vectors. The purple point at the middle of this line is the trivial voting rule that gives every candidate the same score, which is the only element of the 0-dimensional subspace $R_{4,1}$.

the 3-dimensional simplex (a tetrahedron). For instance, the scoring vector $(3, 2, 1, 0)$ for Borda is equivalent to $(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}, 0)$ while the one for veto $(0, 0, 0, -1)$ corresponds to $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, 0)$. Figure 5 depicts this 3-simplex of scoring rules for $m = 4$ candidates and highlights its intersection with the subspaces $R_{4,3}$, $R_{4,2}$, and $R_{4,1}$ along with other rules of interest.[9] Note that $R_{4,4}$ corresponds to the whole simplex.

4.2 Single Transferrable Vote

Next, we consider the Single Transferrable Vote (STV) which cannot be parameterized by a scoring vector. Nonetheless, similar to plurality, we find a strong negative result about its computability with any limited-sized queries.

Theorem 3. *For any number of candidates $m \geq 2$, for all $t < m$, no randomized t-query algorithm can always output an STV winner with probability more than $\frac{1}{m}$.*

Proof. Observe that when $m = 2$, the STV winner is equivalent to the plurality winner, so this is directly implied by Theorem 1. Fix $m \geq 3$. We would like to apply Lemma 3, and to do so, we will construct m profiles $\{\boldsymbol{\sigma}^{c_1}_{STV}, \ldots, \boldsymbol{\sigma}^{c_m}_{STV}\}$ that are all $(m-1)$-indistinguishable, but the STV winner on profile $\boldsymbol{\sigma}^{c_i}_{STV}$ is candidate c_i. The construction of such profiles is as follows. Let $\alpha = (-1, 0, \ldots, 0)$, the negation of the plurality score vector. Since on any profile $\boldsymbol{\sigma}$ and any candidate pair a, b, we have that $\text{plu}_{\boldsymbol{\sigma}}(a) \neq \text{plu}_{\boldsymbol{\sigma}}(b) \iff sc^{\alpha}_{a}(\boldsymbol{\sigma}) \neq sc^{\alpha}_{b}(\boldsymbol{\sigma})$, we can use Lemmas 1

[9] Although as a subspace of $\mathbb{R}^m$, $R_{m,t}$ is t-dimensional, when we restrict to the simplex, we lose a dimension. Hence $R_{4,4}$ becomes 3-dimensional, $R_{4,3}$ becomes 2-dimensional and so on.

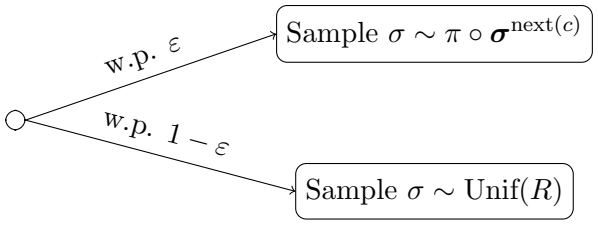

Figure 6: A process inducing $\boldsymbol{\sigma}^c_{STV}$.

and 2 to obtain a set of $(m-1)$-indistinguishable profiles $\{\boldsymbol{\sigma}^c\}_{c\in M}$, where candidate c is the unique α score maximizer on profile $\boldsymbol{\sigma}^c$; correspondingly, it is the unique plurality minimizer on profile $\boldsymbol{\sigma}^c$ due to the choice of α.

Next, fix a directed cycle of the candidates $D := c_1 \to c_2 \to \cdots \to c_m \to c_1$. Let R be the set of all rankings such that the first and last candidates are consecutive in the cycle, i.e., $R = \{\sigma \in \mathcal{L}(C) \mid (\sigma(1), \sigma(m)) \in D\}$. Choose $\varepsilon > 0$ such that $\varepsilon < \frac{1-\varepsilon}{m(m-1)}$ ($\varepsilon = \frac{1}{m^2}$ will do). Fix a candidate c, and let $\text{next}(c)$ be the subsequent candidate in the cycle, i.e., the unique candidate c' such that $(c, c') \in D$. Define $\boldsymbol{\sigma}^c_{STV}$ as follows: with probability ε, output a sample σ from $\boldsymbol{\sigma}^{\text{next}(c)}$, with remaining probability $1-\varepsilon$, select σ from $\text{Unif}(R)$. A visual representation of this can be found in Figure 6.

Observe that for any pair c, c', the profiles $\boldsymbol{\sigma}^c_{STV}$ and $\boldsymbol{\sigma}^{c'}_{STV}$ are $(m-1)$-indistinguishable. Indeed, their generating processes are the same with probability $1-\varepsilon$ (when we sample uniformly from R), and with probability ε they differ due to $\boldsymbol{\sigma}^c$ and $\boldsymbol{\sigma}^{c'}$ which are themselves $(m-1)$-indistinguishable.

We next show that on each profile $\boldsymbol{\sigma}^c_{STV}$, the unique STV winner is c. Without loss of generality, consider the candidate $c = c_m$ (so $\text{next}(c) = c_1$, $\text{next}(c_1) = c_2$, and so on) as the argument holds symmetrically for any other c. We will show by strong induction that the k'th candidate to be eliminated is c_k. This implies that c_m will be the final candidate remaining, and thus the STV winner.

We begin with the base case, that c_1 is the first to be eliminated. Note that when sampling uniformly from R, by symmetry, each candidate has the same plurality score. On the other hand, in $\boldsymbol{\sigma}^{\text{next}(c)}$, candidate $\text{next}(c) = c_1$ is the unique plurality minimizer. Thus, in the mixed profile $\boldsymbol{\sigma}^c_{STV}$, c_1 has the lowest plurality score and is eliminated first.

Next, suppose candidates $c_1, \ldots, c_{k-1}$ for $k \geq 2$ (and $k \leq m-1$) have been eliminated. We will show that the next to be eliminated is c_k. Let $C^k = C \setminus \{c_1, \ldots, c_{k-1}\}$ be the set of uneliminated candidates. We first consider the proportion of first-place votes each candidate $c \in C^k$ gets in $\text{Unif}(R)|_{C^k}$. Let $R = R^1 \sqcup \cdots \sqcup R^m$ be a partition of R such that R^i contains the rankings with c_i ranked first. By symmetry, these are each the same size. Note that for $i \geq k$, $c_i \in C_k$, so rankings in R^i will continue to rank c_i first, each accounting for a $1/m$ proportion of the rankings. For $i \leq k-2$, since $\text{next}(c_i) \notin C^k$, by symmetry, the first place votes of R^i will be distributed equally among all candidates in C^k. For R^{k-1} however, every $\sigma \in R^{k-1}$ ranks c_k last, and, since $|C^k| \geq 2$, no votes will go to c_k; instead, they will be spread equally among $C^k \setminus \{c_k\}$. Hence, the plurality score of c_k on $\text{Unif}(R)|_{C^k}$ will be $\frac{1}{m\cdot(|C^k|-1)} \geq \frac{1}{m(m-1)}$ smaller than all other candidates. By the choice of ε, $(1-\varepsilon)\frac{1}{m(m-1)} > \varepsilon$, so no matter how many first place votes c_k gets on $\boldsymbol{\sigma}^{\text{next}(c)}$, c_k has the smallest plurality score on $\boldsymbol{\sigma}^c_{STV}|_{C^k}$. Therefore, it is the next to be eliminated.

Finally, we apply Lemma 3 to this set of m profiles to conclude that there exists a profile where no $(m-1)$-query algorithm can determine the STV winner with probability greater than $\frac{1}{m}$. $\square$

5 Query Complexity

In the previous section, we characterized when it was information-theoretically possible to find winning candidates using a certain query size. We turn now to focusing on those cases when it *is* possible and prove bounds on the *query complexity*. In other words, when it is possible to determine the winner with limited-sized query size, how many such queries are needed?

For an integer $k \leq m$, we use the notation $\binom{S}{k}$ to denote the set of all subsets of S of size k. Fix a scoring vector $\boldsymbol{\alpha}$ and let t^* be the minimal value such that $\boldsymbol{\alpha} \in R_{m,t^*}$. Suppose we can make queries of size $t \geq t^*$ and wish to find a candidate maximizing $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}$. As a benchmark, note that if we make enough queries to be able to deduce $\boldsymbol{\sigma}|_S$ for all possible $S \in \binom{C}{t^*}$, then it is information-theoretically possible to find this winning candidate. Indeed, one could simulate any t^*-query algorithm using this (say the one from Lemma 4) as they would have the responses for all t-sized query. Let $\mathrm{cov}(m, t, t^*)$ be the minimum number of subsets of size t needed to cover all subsets of size t^* out of a set of size m. This value is referred to as a *covering number*; computing such covering numbers and optimal subset structures that induce them is a canonical problem in combinatorics with a rich history (see, e.g., Mills and Mullin (1995)). For our purposes, a reasonable (and nearly tight) lower bound on covering numbers is

$$\mathrm{cov}(m, t, t^*) \geq \frac{\binom{m}{t^*}}{\binom{t}{t^*}}.$$

This follows from a simple argument: There are $\binom{m}{t^*}$ subsets of size t^*, and each subset of size t can cover at most $\binom{t}{t^*}$ subsets them. When t and t^* are treated as constant, then this value is $\Omega(m^{t^*})$.

Our primary question is whether we can cleverly choose queries to use fewer than $\mathrm{cov}(m, t, t^*)$ queries. As a motivating example, consider instead finding a *Condorcet winner* under our model. A Condorcet winner on profile $\boldsymbol{\sigma}$ is a candidate a that beats all others in a pairwise competition. More formally, for all $b \neq a$, $\Pr_{\sigma \sim \boldsymbol{\sigma}}[a \succ_\sigma b] > 1/2$. Note that Condorcet winners need not exist,[10] however when they do, they are unique. From the definition, we can see that using queries of size $t^* = 2$ is sufficient to determine whether a Condorcet winner exists and, if so, determine this candidate as this only depends on pairwise margins. One option is to make all possible queries of size 2; this requires $\mathrm{cov}(m, 2, 2) = \binom{m}{2} = \Theta(m^2)$ queries. However, as shown by Procaccia (2008), there is a more clever way, requiring only $O(m)$ queries to compute this winner.[11]

We now ask whether such improvements can be found for scoring rules. Unfortunately, we show that for both deterministic and randomized algorithms, they cannot.

Theorem 4. *Fix $\boldsymbol{\alpha} \in \mathbb{R}^m$ and let t^* be the minimal value such that $\boldsymbol{\alpha} \in R_{m,t^*}$. For $t^* \geq 2$, any deterministic t-query algorithm that always outputs a candidate maximizing $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}}$ must make at least $cov(m, t, t^*)$ queries in the worst case[12]. Further, any randomized algorithm making at most $\delta \frac{\binom{m}{t^*}}{\binom{t}{t^*}}$ queries outputs an $\boldsymbol{\alpha}$-winner with probability at most $\min(\delta + \frac{1}{m}, \delta + (1-\delta)\frac{1}{t^*})$ in the worst case.*

Proof. Fix $\boldsymbol{\alpha}$ and t^*. Let C_1 be a set of candidates of size t^* with two distinguished candidates $a, b \in C_1$, and let $C_2 = \overline{C_1}$ be the remaining candidates. We construct $\boldsymbol{\sigma}^1, \ldots, \boldsymbol{\sigma}^{m-t^*}$ just as

[10] The classic example is when a third of the voters have each of the rankings $a \succ b \succ c$, $c \succ a \succ b$, and $b \succ c \succ a$.

[11] The algorithm runs in two phases. First, run a knockout tournament among the candidates, where the candidate receiving more pairwise votes makes it on to the next round. If there is a Condorcet winner, then that candidate must be the winner of the knockout tournament. In the second phase, compare this winner to all other candidates they did not play. If they win all of these comparisons, they are the Condorcet winner, if not, there is no winner. This requires $2m - \lfloor \log m \rfloor - 2$ queries.

[12] It is only when $\boldsymbol{\alpha}$ is a constant vector does $t^* = 1$, a degenerate case that we ignore since any candidate can be considered the winner.

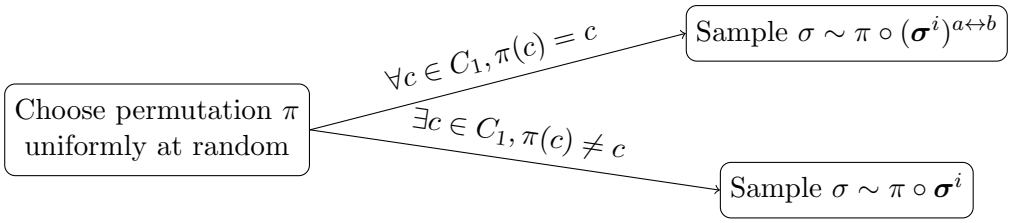

Figure 7: A process inducing $\boldsymbol{\sigma}^*$.

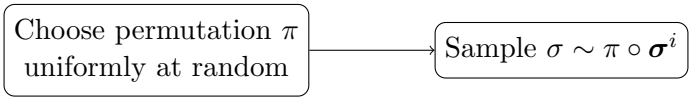

Figure 8: A process inducing $\boldsymbol{\sigma}^{\text{unif}}$.

in Figure 4 from Section 4.1. Recall that each profile $\boldsymbol{\sigma}^i$ has a profile $\boldsymbol{\sigma} \in \Pi(C_1)$ satisfying the conditions of Lemma 1 with a and b "contained" in it. In addition, it has $i-1$ of the C_2 candidates ranked in a fixed order before $\boldsymbol{\sigma}$, and the rest are in a fixed order after. After describing these profiles in Section 4.1, we observed that each $\boldsymbol{\sigma}^i$ and $(\boldsymbol{\sigma}^i)^{a \leftrightarrow b}$ are (t^*-1)-indistinguishable. However, we claim that an even stronger property is true: For any t sized query Q that does not contain C_1, $\boldsymbol{\sigma}^i$ and $(\boldsymbol{\sigma}^i)^{a \leftrightarrow b}$ are indistinguishable (note t may be much larger than t^*). More formally, for all Q such that $C_1 \not\subseteq Q$, $\boldsymbol{\sigma}^i|_Q = (\boldsymbol{\sigma}^i)^{a \leftrightarrow b}|_Q$. Indeed, the candidates of $C_2 \cap Q$ are always in the same order, either before or after the candidates of C_1. The candidates in C_1 will follow the distribution according to $\boldsymbol{\sigma}|_{Q \cap C_1}$ and $\boldsymbol{\sigma}^{a \leftrightarrow b}|_{Q \cap C_1}$. By Lemma 1, since $Q \cap C_1 \subsetneq C_1$, these are identical.

By definition of t^*, $\boldsymbol{\alpha} \notin R_{m,t^*-1}$. Hence, Lemma 5 implies that for one of these profiles, $\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^i}(a) \neq \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^i}(b)$. Fix such an i, and without loss of generality, assume $\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^i}(a) > \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^i}(b)$. Consider the profile $\boldsymbol{\sigma}^*$ induced by sampling a permutation π uniformly at random, and, if $\pi(c) = c$ for all $c \in C_1$, sample $\sigma \sim \pi \circ (\boldsymbol{\sigma}^i)^{a \leftrightarrow b}$, otherwise, sample $\sigma \sim \pi \circ \boldsymbol{\sigma}^i$. This is shown in Figure 7.

Next, we will compare $\boldsymbol{\sigma}^*$ to another profile, $\boldsymbol{\sigma}^{\text{unif}}$, defined in Figure 8, where we sample from $\pi \circ \boldsymbol{\sigma}^i$ regardless of i. Note that $\boldsymbol{\sigma}^{\text{unif}}$ is the uniform distribution over all rankings which can be equivalently achieved by first sampling $\sigma \sim \boldsymbol{\sigma}^i$, and then outputting $\pi \circ \sigma$ for a π that is uniformly selected.[13] We will show two things (i) $\boldsymbol{\sigma}^*|_Q = \boldsymbol{\sigma}^{\text{unif}}$ unless $C_1 \subseteq Q$, and (ii) b is the unique $\boldsymbol{\alpha}$-winner on $\boldsymbol{\sigma}^*$. Note that (i) implies that on any query Q with $C_1 \not\subseteq Q$, $\boldsymbol{\sigma}^*|_Q$ is the uniform distribution over rankings in $\mathcal{L}(Q)$.

For (i), fix a query Q with $C_1 \not\subseteq Q$. Since the $\pi(c) \neq c$ for some $c \in C_1$ branch of $\boldsymbol{\sigma}^*$ is identical to $\boldsymbol{\sigma}^{\text{unif}}$, it suffices to focus on π such that $\pi(c) = c$ for all $c \in C_1$. When this holds, π can only reorder the candidates of C_2, leaving candidates in C_1 unchanged. Therefore, $(\pi \circ (\boldsymbol{\sigma}^i)^{a \leftrightarrow b})|_Q = (\pi \circ \boldsymbol{\sigma}^i)|_Q$. Hence, sampling from $\boldsymbol{\sigma}^*|_Q$ is equivalent to sampling π uniformly at random and sampling from $(\pi \circ \boldsymbol{\sigma}^i)|_Q$, equivalent to sampling from $\boldsymbol{\sigma}^{\text{unif}}|_Q$.

For (ii), we will show that $\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^*}(b) > \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\text{unif}}}(b)$ while $\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^*}(c) \leq \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\text{unif}}}(c)$ for all $c \neq b$. By symmetry, the scores of all candidates in $\boldsymbol{\sigma}^{\text{unif}}$ are the same, hence, this shows that b is the unique winner. To that end, note that the only time in sampling $\boldsymbol{\sigma}^*$ and $\boldsymbol{\sigma}^{\text{unif}}$ that the scores will differ, is if we take the top branch, in which case $\pi(c) = c$ for all $c \in C_1$. By assumption, $\pi(a) = a$ and $\pi(b) = b$ as $a, b \in C_1$, so we are simply swapping a and b. Since $\text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^i}(a) > \text{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^i}(b)$, this strictly

[13] Note that although the generating process for $\boldsymbol{\sigma}^{\text{unif}}$ here is different than the one used in Lemma 2 and Figure 3, the resulting distribution is still the same.

increases the score of b on average, and strictly decreases the score of a. Hence, $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^*}(b) > \mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\mathrm{unif}}}(b)$, $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^*}(a) < \mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\mathrm{unif}}}(a)$, and $\mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^*}(c) = \mathrm{sc}^{\boldsymbol{\alpha}}_{\boldsymbol{\sigma}^{\mathrm{unif}}}(c)$ for all $c \neq a, b$, as needed.

Fix a deterministic algorithm t-query algorithm $\mathcal{A}$ that outputs a candidate after making strictly fewer than $\mathrm{cov}(m, t, t^*)$ queries. We will show that it cannot always output an $\boldsymbol{\alpha}$-winner. Consider a run of the algorithm where on every query Q, it receives in response the uniform distribution over $\mathcal{L}(Q)$. Suppose on this run, it outputs candidate c. Now, by the definition of the covering number, there must be a set C' with $|C'| = t^*$ such that C' was not contained in any query made by the algorithm. Since $t^* \geq 2$, $C' \setminus \{c\}$ is not empty. Let $c^* \in C' \setminus \{c\}$. Let π be a permutation such that $\pi(b) = c^*$ and π maps $C_1 \setminus \{b\}$ to $C' \setminus \{c^*\}$. Consider the running $\mathcal{A}$ on $\pi \circ \boldsymbol{\sigma}^*$. Note that on every query Q not containing C_1, the response will be indistinguishable from $\boldsymbol{\sigma}^{\mathrm{unif}}$, and hence, the uniform distribution over Q. Therefore, by above, $\mathcal{A}$ will return candidate c on this instance. However, by construction, c^* is the unique $\boldsymbol{\alpha}$-winner, and $c^* \neq c$, a contradiction.

Next, we will show that a randomized algorithm making at most $\delta \frac{\binom{m}{t^*}}{\binom{t}{t^*}}$ queries will output an $\boldsymbol{\alpha}$-winner with probability at most $\min(\delta + \frac{1}{m}, \delta + (1-\delta)\frac{1}{t^*})$. We will make use of Yao's Minimax Principle (Yao, 1977). More specifically, will show that there is a distribution over profiles such that no deterministic algorithm can be correct with larger probability. This implies that no randomized algorithm can achieve a larger probability on a worst-case profile.

The distribution over instances we will choose is simply uniform over $\pi \circ \boldsymbol{\sigma}^*$ for all permutations π. Fix an arbitrary deterministic algorithm $\mathcal{A}$ that always outputs a candidate after at most $\delta\binom{m}{t^*}/\binom{t}{t^*}$ queries. Consider a run of this algorithm where the response to every query Q is the uniform distribution over $\mathcal{L}(Q)$, and suppose on this run, the output candidate is c. Let $\mathcal{Q}$ be the set of queries asked on this run. We have that $|\mathcal{Q}| \leq \delta\binom{m}{t^*}/\binom{t}{t^*}$ by assumption. Observe that since each query of size t can cover $\binom{t}{t^*}$ sets of size t^*, at most a δ-fraction of the $\binom{m}{t^*}$ t^*-sets are covered by $\mathcal{Q}$.

We claim that the algorithm must be incorrect for all $\pi \circ \boldsymbol{\sigma}^*$ such that both (i) $\pi(C_1) \not\subseteq Q$ for all $Q \in \mathcal{Q}$ and (ii) $\pi(b) \neq c$. Indeed, (i) ensures that the run of the algorithm will always lead to uniform responses, meaning $\mathcal{A}$ *must* output c, and (ii) ensures this is the incorrect choice. More formally, let

$$\mathcal{E}_1 = \{\pi \mid \pi(C_1) \not\subseteq Q \text{ for all } Q \in \mathcal{Q}\}$$

be the event that the first property holds, and

$$\mathcal{E}_2 = \{\pi \mid \pi(b) \neq c\}$$

be the event that the second does. The probability of success is at most $1 - \Pr[\mathcal{E}_1 \cap \mathcal{E}_2]$. We will upper bound this in two ways. First,

$$1 - \Pr[\mathcal{E}_1 \cap \mathcal{E}_2] = \Pr[\overline{\mathcal{E}_1} \cup \overline{\mathcal{E}_2}] \leq \Pr[\overline{\mathcal{E}_1}] + \Pr[\overline{\mathcal{E}_2}] \leq \delta + \frac{1}{m},$$

where the first inequality holds by the union bound, $\Pr[\overline{\mathcal{E}_1}] \leq \delta$ because at most a δ-fraction of all t^*-subsets are covered, and $\Pr[\overline{\mathcal{E}_2}] \leq \frac{1}{m}$ by symmetry. Second,

$$\begin{aligned}
1 - \Pr[\mathcal{E}_1 \cap \mathcal{E}_2] &= 1 - \Pr[\mathcal{E}_2 \mid \mathcal{E}_1] \cdot \Pr[\mathcal{E}_1] \\
&\leq 1 - \left(1 - \frac{1}{t^*}\right) \cdot (1 - \delta) \\
&= 1 - (1 - \delta) + (1 - \delta) \cdot \frac{1}{t^*} \\
&= \delta + (1 - \delta) \cdot \frac{1}{t^*}.
\end{aligned}$$

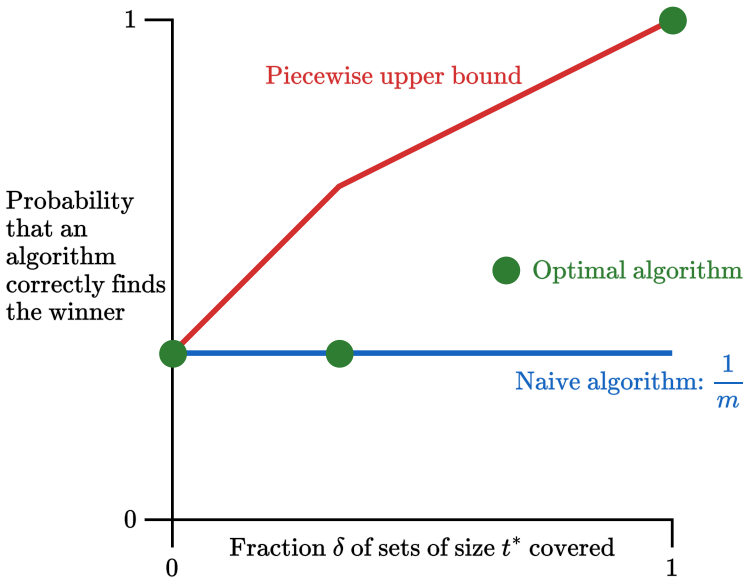

Figure 9: Success probabilities for randomized algorithms computing the Borda winner on $m = 3$ candidates making queries of size $t = t^* = 2$. The red piecewise-linear curve, which has similar shapes for larger parameters and other computable scoring rules, represents the upper bound on the success probability of any algorithm from Theorem 4. The blue curve is the best-known general lower bound obtained by the algorithm that makes no queries and simply guesses a candidate at random. The green points are the true success probabilities of the optimal algorithm given by Theorem 5.

Again, $\Pr[\mathcal{E}_1] \geq 1 - \delta$ holds because at most a δ-fraction are covered. The other term $\Pr[\mathcal{E}_2 \mid \mathcal{E}_1] \cdot \Pr[\mathcal{E}_1]$ holds because conditioned on $\pi(C_1) = C'$ for *any* uncovered C', the probability that $\pi(b) = c$ is at most $1 - \frac{1}{t^*}$. Indeed, this holds exactly if $c \in C'$, and is 0 otherwise. Together, these two bounds imply that the probaility of success is at most $\min(\delta + \frac{1}{m}, \delta + (1-\delta)\frac{1}{t^*})$. □

Theorem 4 completely settles the query complexity of deterministic algorithms for computing all positional scoring rules. However, for randomized algorithms, the story is not quite complete. On the one hand, if t^* and t are constants, and an algorithm would like to be correct with probability $\frac{1}{m} + c$ for a constant c, then $\Omega(m^{t^*})$ queries are needed (hiding constants depending on t^*, t, and c), and $O(m^{t^*})$ clearly suffice, as regardless of t, $\binom{m}{t^*}$ are certainly enough to cover all sets. On the other hand, if an algorithm can make $\delta\frac{\binom{m}{t^*}}{\binom{t}{t^*}}$ queries for a fixed δ, we do not know the exact probability with which it can be correct. Figure 9 depicts the gap between the upper bound and the best general lower bound as functions of the parameter δ. This lower bound is essentially the naive algorithm achieving $\frac{1}{m}$ by simply picking a candidate at random. However, as the following result (Theorem 5) shows, even for the simplest nontrivial case of $m = 3$ and $t = t^* = 2$, the true query complexity of computing the (essentially unique) scoring rule in $R_{3,2}$ is strictly between our general bounds when $\delta = \frac{2}{3}$.

Theorem 5. *With $m = 3$ candidates, the optimal randomized algorithm making queries of size $t = 2$ to compute the Borda count winner succeeds with*

1. *worst-case success probability $\frac{1}{3}$ when allowed to make exactly one query, and*
2. *worst-case success probability $\frac{1}{2}$ when allowed to make exactly two queries.*

Before giving the proof, we note that the measure of worst-case optimality here is a bit finicky. On the negative side, we show that for all algorithms and any $\varepsilon > 0$, there are instances where they do not succeed with probability more than $\frac{1}{3} + \varepsilon$ for (1), and $\frac{1}{2} + \varepsilon$ for (2). At least for (1), this relaxation is necessary. Consider the algorithm that makes a single query to a uniformly random

pair of candidates and selects between them with the probabilities given by the query response (i.e., if the algorithm learns that $\Pr_{\sigma\sim\boldsymbol{\sigma}}[a \succ_\sigma b] = p$, it picks a with probability p and b with probability $1-p$). It can be shown that this process selects each candidate with probability proportional to its Borda score.[14] Given any fixed profile, unless all three candidates have the same score, a maximal one will be selected with probability strictly greater than $\frac{1}{3}$ (and if they are all the same, then all are Borda winners, and hence the algorithm succeeds with probability 1). However, there are instances where the best Borda score is arbitrarily close to the others, resulting in a success probability no constant greater than $\frac{1}{3}$. Thus, by saying that "the optimal" randomized algorithm that makes a single query achieves a worst-case success probability of $\frac{1}{3}$, we really mean that it is not possible to surpass $\frac{1}{3}$ by any constant. In the proof of Theorem 5, we must construct a family of increasingly more difficult instances that bring the success probabilities closer to $\frac{1}{3}$ and $\frac{1}{2}$. This becomes quite complicated, involving a construction based on Fibonacci numbers to ensure query responses do not leak cardinal information about the relative strengths of candidates.

Proof of Theorem 5: Fix a set of candidates $C = \{a, b, c\}$. Throughout the proof, we use the scoring vector $(1, 0, -1)$ to compute Borda scores (which is equivalent to the more traditional choice of $(2, 1, 0)$ by translation). When writing scores, we drop the $(1, 0, -1)$ superscript in the score notation, using $\mathrm{sc}_{\boldsymbol{\sigma}}(c')$ to refer to $\mathrm{sc}^{(1,0,-1)}_{\boldsymbol{\sigma}}(c')$. We also use the convention that when $\boldsymbol{\sigma}$ is queried on $\{a, b\}$, the algorithm learns $\Pr_{\sigma\sim\boldsymbol{\sigma}}[a \succ_\sigma b]$, on $\{b, c\}$, the algorithm learns $\Pr_{\sigma\sim\boldsymbol{\sigma}}[b \succ_\sigma c]$, and on $\{a, c\}$, the algorithm learns $\Pr_{\sigma\sim\boldsymbol{\sigma}}[c \succ_\sigma a]$. These single numbers completely parameterize the distribution $\boldsymbol{\sigma}|_Q$ for each Q of size 2. One can check that the following equalities hold for scores

$$\begin{aligned}
\mathrm{sc}_{\boldsymbol{\sigma}}(a) &= \Pr_{\sigma\sim\boldsymbol{\sigma}}[a \succ_\sigma b] - \Pr_{\sigma\sim\boldsymbol{\sigma}}[b \succ_\sigma c],\\
\mathrm{sc}_{\boldsymbol{\sigma}}(b) &= \Pr_{\sigma\sim\boldsymbol{\sigma}}[b \succ_\sigma c] - \Pr_{\sigma\sim\boldsymbol{\sigma}}[c \succ_\sigma a],\\
\mathrm{sc}_{\boldsymbol{\sigma}}(c) &= \Pr_{\sigma\sim\boldsymbol{\sigma}}[c \succ_\sigma a] - \Pr_{\sigma\sim\boldsymbol{\sigma}}[a \succ_\sigma b].
\end{aligned}$$

We begin with the lower bounds on the probabilities. First, note that it is always possible to succeed with probability $\frac{1}{3}$ by just picking a random one of the three candidates. This establishes the lower bound on (1).

For (2), consider the following algorithm. We pick a random candidate c' and query both sets of size 2 containing that candidate. From this information, we are able to learn the Borda score of candidate c'. If the score is positive, we return c'. Otherwise, we randomly return one of the other two candidates. Observe that, with our choice of scoring vector $(1, 0, -1)$, the sum of all three Borda scores must be zero, so at most two are strictly positive. If none of them are positive, then they must all be zero, in which case every candidate is a Borda winner, and the algorithm succeeds with probability 1. If one Borda score is positive, then if that candidate is chosen as c, the algorithm succeeds with probability 1, and otherwise the algorithm succeeds with probability $\frac{1}{2}$. Since the former case happens with probability $\frac{1}{3}$, the total expected success probability is $\frac{2}{3}$. Finally, if two Borda scores are positive, then if the true Borda winner is chosen as c', the algorithm succeeds with probability 1; if the other candidate with positive Borda score is chosen as c', the algorithm incorrectly returns it, succeeding with probability 0; and if the candidate with negative Borda score is chosen as c, the algorithm succeeds with probability $\frac{1}{2}$. In total, the success probability is $\frac{1}{3}\cdot 1 + \frac{1}{3}\cdot 0 + \frac{1}{3}\cdot\frac{1}{2} = \frac{1}{2}$. Thus, the worst-case success probability is $\frac{1}{2}$.

For the upper bounds, we again use Yao's Minimax principle. That is, we will show that there are distributions over instances where no deterministic algorithm can output a Borda winner with

[14] The probability it picks a candidate c is equal to $\frac{1}{\binom{m}{2}}\sum_{c'\neq c}\Pr_{\sigma\sim\boldsymbol{\sigma}}[c \succ c']$. It is well known that the Borda score of c is equal to $\sum_{c'\neq c}\Pr_{\sigma\sim\boldsymbol{\sigma}}[c \succ c']$ Brandt et al. (2016).

probability more than $\frac{1}{3} + \varepsilon$ for any $\varepsilon > 0$. This implies that no randomized algorithm can do so on every instance.

Define a family of distributions over profile $D_1, D_2, D_3, \ldots$ as follows. To generate D_n, we first sample three values, $p_1, p_2, p_3 \in [\frac{1}{3}, \frac{2}{3}]$ and return an arbitrary profile $\boldsymbol{\sigma}$ where

$$p_1 := \Pr_{\sigma \sim \boldsymbol{\sigma}}[a \succ_\sigma b],$$
$$p_2 := \Pr_{\sigma \sim \boldsymbol{\sigma}}[b \succ_\sigma c],$$
$$p_3 := \Pr_{\sigma \sim \boldsymbol{\sigma}}[c \succ_\sigma a].$$

Before showing how to sample p_1, p_2, p_3, we first show that such a profile $\boldsymbol{\sigma}$ satisfying the pairwise margins always exists. We claim that the following preference profile suffices.

Ranking	Probability
$a \succ b \succ c$	$p_2 - \frac{1}{3}$
$a \succ c \succ b$	$\frac{2}{3} - p_2$
$b \succ c \succ a$	$p_3 - \frac{1}{3}$
$b \succ a \succ c$	$\frac{2}{3} - p_3$
$c \succ a \succ b$	$p_1 - \frac{1}{3}$
$c \succ b \succ a$	$\frac{2}{3} - p_1$

The reader may verify that:

- All probabilities lie in $[0, 1]$ for $p_1, p_2, p_3 \in [\frac{1}{3}, \frac{2}{3}]$.
- The sum of all six probabilities is 1.
- The pairwise ranking probabilities are indeed given by p_1, p_2, and p_3.

As shown above, the p_i values contain all relevant information for computing the Borda scores of each candidate: $\mathrm{sc}_{\boldsymbol{\sigma}}(a) = p_1 - p_3$, $\mathrm{sc}_{\boldsymbol{\sigma}}(b) = p_2 - p_1$, and $\mathrm{sc}_{\boldsymbol{\sigma}}(c) = p_3 - p_2$, along with the responses for all the queries. Hence, the exact construction of $\boldsymbol{\sigma}$ will not be important for the remainder of the proof.

Let $F_1, F_2, F_3, \ldots$ be the Fibonacci sequence shifted to the left by one, beginning with $F_1 := 1$, $F_2 := 2$, $F_3 := 3$, $F_4 := 5$, and so on. For any positive integer n, we generate p_1, p_2, and p_3 for D_n using the following process. First, sample i uniformly from $\{1, 2, \ldots, n\}$, sample s uniformly from $\{0, 1, 2, \ldots, nF_{n+2}\}$ (all integers from 0 to nF_{n+2}), and sample r uniformly from $\{1, 2, 3, 4, 5, 6\}$. Then output the profile $\boldsymbol{\sigma}(i, s, r)$ defined by the table below, where p_1, p_2, and p_3 are defined from the auxiliary values $\hat{p}_1$, $\hat{p}_2$ and $\hat{p}_3$ by the correspondence

$$p_j := \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{((n+1)F_{n+2})} \cdot \hat{p}_j.$$

Profile	True Borda winner	Scaled probabilities $(\hat{p}_1, \hat{p}_2, \hat{p}_3)$
$\boldsymbol{\sigma}(i, s, 1)$	a	$(s + F_{i+2}, s, s + F_i)$
$\boldsymbol{\sigma}(i, s, 2)$	c	$(s + F_{i+2}, s, s + F_{i+1})$
$\boldsymbol{\sigma}(i, s, 3)$	b	$(s + F_i, s + F_{i+2}, s)$
$\boldsymbol{\sigma}(i, s, 4)$	a	$(s + F_{i+1}, s + F_{i+2}, s)$
$\boldsymbol{\sigma}(i, s, 5)$	c	$(s, s + F_i, s + F_{i+2})$
$\boldsymbol{\sigma}(i, s, 6)$	b	$(s, s + F_{i+1}, s + F_{i+2})$

Note that each $p_j \in [\frac{1}{3}, \frac{2}{3}]$ if and only if $\hat{p}_j \in [0, (n+1)F_{n+2}]$, so any pair (i, s) is valid. We leave the reader to verify that the Borda winners are as stated in the table. We explicitly compute winners for the final profile as an example:

$$\begin{aligned}
\mathrm{sc}_{\boldsymbol{\sigma}(s,i,6)}(a) &= p_1 - p_3 = \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{((n+1)F_{n+2})} \cdot (\hat{p}_1 - \hat{p}_3) \\
&= \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{((n+1)F_{n+2})} \cdot ((s) - (s + F_{i+2})) = \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{((n+1)F_{n+2})} \cdot (-F_{i+2}) \\
\mathrm{sc}_{\boldsymbol{\sigma}(s,i,6)}(b) &= p_2 - p_1 = \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{((n+1)F_{n+2})} \cdot (\hat{p}_2 - \hat{p}_1) \\
&= \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{((n+1)F_{n+2})} \cdot ((s + F_{i+1}) - (s)) = \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{((n+1)F_{n+2})} \cdot (F_{i+1}) \\
\mathrm{sc}_{\boldsymbol{\sigma}(s,i,6)}(c) &= p_3 - p_2 = \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{((n+1)F_{n+2})} \cdot (\hat{p}_3 - \hat{p}_2) \\
&= \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{((n+1)F_{n+2})} \cdot ((s + F_{i+2}) - (s + F_{i+1})) = \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{((n+1)F_{n+2})} \cdot (F_i)
\end{aligned}$$

Thus, the winner is b, since F_{i+1} is the largest out of $\{-F_{i+2}, F_{i+1}, F_i\}$.

Fix a deterministic algorithm $\mathcal{A}$ making at most one query of size 2. Observe that D_n is completely symmetric with respect to p_1, p_2, and p_3 (in the sense that permuting $p_1 \mapsto p_2$, $p_2 \mapsto p_3$, and $p_3 \mapsto p_1$ gives the same distribution on preference profiles). Thus, we may assume without loss of generality that an algorithm queries $\{a, b\}$ and learns $\hat{p}_1 \in \{0, 1, 2, \ldots, (n+1)F_{n+2}\}$. Let $\mathcal{E}$ be the event that $i \in \{3, 4, 5, \ldots, n-2\}$ and $s \in [F_{n+2}, (n-2)F_{n+2}]$. By the union bound, the probability that $\mathcal{E}$ does not occur is at most

$$\begin{aligned}
\Pr[\overline{\mathcal{E}}] &\leq \Pr[i \in \{1, 2, n-1, n\}] + \Pr[s \in [0, F_{n+2}] \cup [(n-2)F_{n+2}, nF_{n+2}]] \\
&\leq \frac{4}{n} + \frac{F_{n+2} + 2F_{n+2}}{nF_{n+2}} \\
&= \frac{7}{n}.
\end{aligned}$$

When $\mathcal{E}$ occurs, we will have $\hat{p}_j \in [F_{n+2}, (n-1)F_{n+2}]$ for each j, so in particular this holds for $\hat{p}_1$. Suppose that the algorithm additionally learns i, which only makes it stronger. Then there are exactly six possible choices of the parameters s and c that could have led to the specific realization $\hat{p}_1$:

- $r = 1$ and $s = \hat{p}_1 - F_{i+2} \implies a$ is the winner.
- $r = 2$ and $s = \hat{p}_1 - F_{i+2} \implies c$ is the winner.
- $r = 3$ and $s = \hat{p}_1 - F_i \implies b$ is the winner.
- $r = 4$ and $s = \hat{p}_1 - F_{i+1} \implies a$ is the winner.
- $r = 5$ and $s = \hat{p}_1 \implies c$ is the winner.
- $r = 6$ and $s = \hat{p}_1 \implies b$ is the winner.

Since each possibility is equally likely, no matter which candidate the algorithm picks it succeeds with probability $\frac{1}{3}$. Thus, overall,

$$\begin{aligned}\Pr[\text{success}] &= \Pr[\mathcal{E}]\Pr[\text{success} \mid \mathcal{E}] + \Pr[\overline{\mathcal{E}}]\Pr[\text{success} \mid \overline{\mathcal{E}}] \\ &\leq \Pr[\text{success} \mid \mathcal{E}] + \Pr[\overline{\mathcal{E}}] \\ &\leq \frac{1}{3} + \frac{7}{n}.\end{aligned}$$

By picking n sufficiently large, we see that no algorithm that makes a single query can achieve a worst-case success probability of $\frac{1}{3} + \varepsilon$ for any $\varepsilon > 0$.

For an algorithm that makes two queries, we similarly assume without loss of generality that the queries yielded the values of $\hat{p}_1$ and $\hat{p}_2$ in some order. Regardless of whether the second query was made adaptively or nonadaptively, we will argue that, from these responses the algorithm cannot determine the winner with probability greater than $\frac{1}{2} + \frac{7}{n}$. As before, it suffices to show that the algorithm succeeds with probability at most $\frac{1}{2}$ when $\hat{p}_1, \hat{p}_2 \in [F_{n+2}, (n-1)F_{n+2}]$. Here there are two cases to consider, depending on which observed value is larger.

First suppose $\hat{p}_1 > \hat{p}_2$, and let $j \in \{3, 4, 5, \ldots, n\}$ be such that $\hat{p}_1 - \hat{p}_2 = F_j$. Then there are exactly two possible choices of the parameters s, i, and c that could have led to the specific realizations $\hat{p}_1$ and $\hat{p}_2$:

- $r = 1$, $i = j - 2$, and $s = \hat{p}_1 - F_{i+2} \implies a$ is the winner.
- $r = 2$, $i = j - 2$, and $s = \hat{p}_1 - F_{i+2} \implies c$ is the winner.

Thus, a and c are equally likely to be the winner, so no matter which candidate the algorithm returns, it will be correct with probability at most $\frac{1}{2}$.

Now suppose $\hat{p}_1 < \hat{p}_2$, and let $j \in \{3, 4, 5, \ldots, n\}$ be such that $\hat{p}_2 - \hat{p}_1 = F_j$. Then there are exactly four possibilities for s, i, and c:

- $r = 3$, $i = j - 1$, and $s = \hat{p}_1 - F_{i+i} \implies b$ is the winner. Note that this is the first place we make use of the Fibonacci recurrence: $\hat{p}_2 - \hat{p}_1 = (s + F_{i+2}) - (s + F_i) = F_{i+1} = F_j$.
- $r = 4$, $i = j$, and $s = \hat{p}_1 - F_{i+1} \implies a$ is the winner (again using the Fibonacci recurrence).
- $r = 5$, $i = j$, and $s = \hat{p}_1 \implies c$ is the winner.
- $r = 6$, $i = j - 1$, and $s = \hat{p}_1 \implies b$ is the winner.

Clearly, b is the best guess here, but it is still only correct with probability $\frac{1}{2}$.

Thus, picking n sufficiently large as before, we conclude that no algorithm making only two queries can achieve a worst-case success probability of $\frac{1}{2} + \varepsilon$ for any $\varepsilon > 0$. □

6 Discussion

Voting rules are increasingly applied to aggregate preferences across a large range of candidates, from primary elections to online opinions. This can, however, be at odds with the cognitive and implementation challenges that exist when requiring individuals to specify preferences across a large selection. Naturally in such scenarios, voters end up specifying preferences over a limited set, whether by explicit design or implicitly by submitting incomplete votes. Our work studies the implications of this phenomenon on the computability of voting rules.

For the large class positional scoring rules, we provide an exact information-theoretic characterization of what can and cannot be correctly computed under this model. Specifically, a decrease

in query size equivalently diminishes the dimension of the computable scoring vector space. We explicitly characterize these spaces, finding that, while the Borda count is included for $t \geq 2$, Plurality is not for *any* limited-sized query. We also extend this strong impossibility to STV. From a practical perspective, these results demonstrate the pitfalls of common rules like Plurality and STV within the setting incomplete votes and point to the space of alternative rules. Future work could go beyond voting and further investigate this question of computability with limited-sized queries for other social choice rules or other more general functions, such as committee selection with rankings.

For rules computable with at least t^*-sized queries, we also give bounds on the query complexity of any deterministic or randomized algorithm making $t \geq t^*$ sized queries. While we show that deterministic algorithms must cover the space of all t^* sized queries in the worst-case, thus giving a tight bound, the picture for randomized algorithms is far less clear. We give an upper bound on the success probability when using a given number of queries, yet no known general-purpose algorithm achieves anything close to it. In Theorem 5, we close this gap for a special case of the Borda rule by constructing surprisingly intricate hard instances. Closing the general query complexity gap in our randomized setting is an intriguing open problem whose technical depth is illustrated by this result.

References

M. Bentert and P. Skowron. 2020. Comparing election methods where each voter ranks only few candidates. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI)*. 2218–2225.

F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. D. Procaccia (Eds.). 2016. *Handbook of Computational Social Choice*. Cambridge University Press.

C. M. Burnett and V. Kogan. 2015. Ballot (and voter) "exhaustion" under Instant Runoff Voting: An examination of four ranked-choice elections. *Electoral Studies* 37 (2015), 41–49.

V. Conitzer and T. Sandholm. 2005. Communication Complexity of Common Voting Rules. In *Proceedings of the 6th ACM Conference on Economics and Computation (EC)*. 78–87.

S. Cunow, S. Desposato, A. Janusz, and C. Sells. 2021. Less is more: The paradox of choice in voting behavior. *Electoral Studies* 69 (2021), 102230.

S. Cunow, S. Desposato, A. Janusz, and C. Sells. 2023. Too much of a good thing? Longer ballots reduce voter participation. *Journal of Elections, Public Opinion and Parties* (2023), 1–18.

P. Dey and A. Bhattacharyya. 2015. Sample complexity for winner prediction in elections. In *Proceedings of the 14th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. 1421–1430.

FairVote. 2024. *Research and data on RCV in practice.* `https://fairvote.org/resources/data-on-rcv/`

Y. Filmus and J. Oren. 2014. Efficient Voting via the Top-k Elicitation Scheme: A Probabilistic Approach. In *Proceedings of the 15th ACM Conference on Economics and Computation (EC)*. 295–312.

P. C. Fishburn. 1977. Condorcet Social Choice Functions. *SIAM J. Appl. Math.* 33, 3 (1977), 469–487.

D. Halpern, G. Kehne, A. D. Procaccia, J. Tucker-Foltz, and M. Wüthrich. 2023. Representation with Incomplete Votes. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence (AAAI)*. 5657–5664.

S. Hirano and J. M. Snyder Jr. 2019. *Primary elections in the United States*. Cambridge University Press.

C. Horton. 2018. The simple but ingenious system Taiwan uses to crowdsource its laws. *MIT Technology Review* (2018).

S. S. Iyengar and M. R. Lepper. 2000. When choice is demotivating: Can one desire too much of a good thing? *Journal of personality and social psychology* 79, 6 (2000), 995.

K. Konczak and J. Lang. 2005. Voting Procedures with Incomplete Preferences. In *Proceedings of the 2nd Multidisciplinary Workshop on Advances in Preference Handling (M-PREF)*.

T. Lu and C. Boutilier. 2011. Robust Approximation and Incremental Elicitation in Voting Protocols. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*. 287–293.

W. H. Mills and R. C. Mullin. 1995. Coverings and packings. In *Contemporary Design Theory: A Collection of Surveys*, J. H. Dinitz and D. R. Stinson (Eds.). Wiley, Chapter 9.

J. Oren, Y. Filmus, and C. Boutilier. 2013. Efficient vote elicitation under candidate uncertainty. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI)*. 309–316.

A. D. Procaccia. 2008. A Note on the Query Complexity of the Condorcet Winner Problem. *Inform. Process. Lett.* 108, 6 (2008), 390–393.

A. D. Procaccia and J. S. Rosenschein. 2006. The Distortion of Cardinal Preferences in Voting. In *Proceedings of the 10th International Workshop on Cooperative Information Agents (CIA)*. 317–331.

B. Schwartz. 2004. *The paradox of choice: Why more is less*. Harper Perennial.

C. Small, M. Bjorkegren, T. Erkkilä, L. Shaw, and C. Megill. 2021. Polis: Scaling Deliberation by Mapping High Dimensional Opinion Spaces. *Revista De Pensament I Anàlisi* 26, 2 (2021).

L. Xia and V. Conitzer. 2011. Determining Possible and Necessary Winners Given Partial Orders. *Journal of Artificial Intelligence Research* 41 (2011), 25–67.

A. C. Yao. 1977. Probabilistic Computations: Towards a Unified Measure of Complexity. In *Proceedings of the 17th Symposium on Foundations of Computer Science (FOCS)*. 222–227.

H. P. Young. 1975. Social Choice Scoring Functions. *SIAM Journal of Applied Mathematics* 28, 4 (1975), 824–838.