Robust Approximation and Incremental Elicitation in Voting Protocols

Tyler Lu
University of Toronto
Department of Computer Science
tl@cs.toronto.edu

Craig Boutilier
University of Toronto
Department of Computer Science
cebly@cs.toronto.edu

Abstract

While voting schemes provide an effective means for aggregating preferences, methods for the effective elicitation of voter preferences have received little attention. We address this problem by first considering *approximate winner determination* when incomplete voter preferences are provided. Exploiting natural scoring metrics, we use *max regret* to measure the quality or *robustness* of proposed winners, and develop polynomial time algorithms for computing the alternative with *minimax regret* for several popular voting rules. We then show how minimax regret can be used to effectively drive incremental preference/vote elicitation and devise several heuristics for this process. Despite worst-case theoretical results showing that most voting protocols require nearly complete voter preferences to determine winners, we demonstrate the practical effectiveness of regret-based elicitation for determining both approximate and exact winners on several real-world data sets.

1 Introduction

Effective schemes for the aggregation of user preferences is critical in settings where a single *consensus* decision or recommendation must be made for a group of users. While social choice theorists have studied such problems for decades, the availability of data about the preferences of millions of individuals afforded by search engines, recommender systems, and related artifacts, has made the practical, computational solution to such problems all the more relevant.

Incremental elicitation of preferences is critical to easing cognitive and communication demands on users and mitigating privacy concerns. However, the use of *voting schemes* to support consensus decision making rarely takes this into account. In most schemes, users or *voters* express their preferences over the space of options or *alternatives*. The voting scheme then selects a consensus option, or *winner*. Requiring voters to express full preference orderings can be onerous (especially for large sets of alternatives) and often captures more information than is needed to determine the winner. While winners can't be determined in many voting schemes without a large amount of information in the worst case [5; 6], this does not diminish the practical imperative to minimize the amount of information elicited from voters.

To address this, we consider two key issues. First, given *partial* information about voter preferences, how does one choose a suitable *approximate winner*? Note that we are discussing *informational approximation* (cf. existing work on *computational approximation* of voting schemes focus under conditions of full information). In this sense, our approach shares much with research on *possible and necessary winner determination* [11; 12; 19]; however, we propose a general technique by which one actually *chooses a winner* in settings with incomplete preferences. Using *maximum regret* to quantify the worst-case error, the alternative that minimizes this error is selected, providing us with a form of *robust optimization*. We develop methods for computing the minimax-optimal alternative for a selection of voting rules.

The second issue we address is incremental *vote elicitation*. If the available voter preference information is too limited, the potential error associated with the robust winner (i.e., its max regret) may be unacceptable. We use the solution to the robust winner determination problem to choose *which queries to ask of which voters*. We discuss distribution-free heuristics that allow us to determine queries that quickly allow one to reduce minimax regret. Our experimental results on randomly-generated and real-world data sets show that winners can be determined even when voters express a relatively small proportion of their preferences. While our elicitation schemes do not exploit preference distribution information, they can be readily extended to do so (as we discuss in Sec. 6).

The paper is organized as follows. We provide brief background on social choice and voting rules in Sec. 2. We define minimax regret for voting rules, and discuss its relationship with possible and necessary winners, in Sec. 3, deferring discussion of algorithms and complexity to Sec. 4. Sec. 5 presents heuristics for elicitation along with empirical results demonstrating their effectiveness on several data sets. Sec. 6 concludes with discussion of future research directions.

2 The Social Choice Problem

We begin with a review of basic concepts from social choice and several common voting schemes (see [7; 14] for further background). We assume a set of *agents* (or *voters*) $N = \{1, \ldots, n\}$ and a set of *alternatives* $A = \{a_1, \ldots, a_m\}$. Alternatives can represent any outcome space over which the voters have preferences (e.g., product configurations, restaurant dishes, candidates for office, public projects, etc.) and for

which a single collective choice must be made. Let Γ_A be the set of *rankings* (or *votes*) over A (i.e., permutations over A). Voter k's preferences are represented by a ranking $v_k \in \Gamma_A$. Let $v_k(a)$ denote the *rank* of a in v_k. Then k prefers a_i to a_j, written $a_i \succ_k a_j$ (or $a_i \succ a_j$ when k is clear from context), if $v_k(a_i) < v_k(a_j)$. We refer to a collection of votes $\mathbf{v} = \langle v_1, \ldots, v_n \rangle \in \Gamma_A^n$ as a *preference profile*. Let V be the set of all such profiles.

Given a preference profile, we consider the problem of selecting a *consensus alternative*, requiring the design of a *social choice function* or *voting rule* $r : V \rightarrow A$ which selects a "winner" given voter rankings/votes. *Plurality* is one of the most common rules: the alternative with the greatest number of "first place votes" wins (various tie-breaking schemes can be adopted). Plurality does not require that voters provide rankings; however, this "elicitation advantage" means that it fails to account for relative voter preferences for any alternative other than its top choice. Other schemes produce winners that are more sensitive to relative preferences, among them:

- *Positional scoring rules*: Let $\alpha(1), \ldots, \alpha(m)$ score each rank position with $\alpha(i) \geq \alpha(i+1)$. The *score* of a is $s_\alpha(a, \mathbf{v}) = \sum_k \alpha(v_k(a))$. The winner is the a with the greatest score. The well-known *Borda count* is a positional rule with $\alpha(i) = m - i$. Plurality (with $\alpha(1) = 1$ and $\alpha(i) = 0$ for $i > 1$), k-approval (with $\alpha(i) = 1$ if $i \leq k$, $\alpha(i) = 0$ o.w.), k-veto ($\alpha(i) = 1$ if $i \leq m - k$, $\alpha(i) = 0$ o.w.) are also positional rules.
- *Maxmin fairness*: Let $s_f(a, \mathbf{v}) = \min\{m - v_k(a) : k \in N\}$. The winner is the a with maximum score (i.e., the alternative whose worst rank among all voters is highest).
- *Copeland*: Define $W(a_i, a_j; \mathbf{v})$ to be 1 if more voters prefer a_i to a_j, 0.5 if tied, and 0 otherwise. Let $s_c(a_i) = \sum_{j \neq i} W(a_i, a_j; \mathbf{v})$. The a with highest score $s_c(a)$ wins.
- *Maximin*: Let $N(a_i, a_j; \mathbf{v}) = |\{v_k : v_k(a_i) < v_k(a_j)\}|$ be the number of voters who prefer a_i to a_j. Let $s_m(a_i, \mathbf{v}) = \min_{j \neq i} N(a_i, a_j; \mathbf{v})$. The a with highest score $s_m(a, \mathbf{v})$ wins.
- *Bucklin*: The Bucklin score $s_B(a, \mathbf{v})$ is the smallest $k \in \{1, \ldots, m\}$ such that more than half of all voters rank a above position k. The winner is the a with smallest Bucklin score.

We will mention other voting rules below without defining them; see [19] for brief descriptions. Notice that these voting schemes explicitly *score* alternatives, implicitly defining "societal utility" for each alternative. This is true of many (though not all) voting schemes.

One obstacle to the widespread use of voting schemes that require full rankings is the informational and cognitive burden imposed on voters, and concomitant ballot complexity. (Political factors play a role as well). This partly explains the popularity of plurality voting in many jurisdictions [14], and the advocacy of more expressive methods (e.g., approval voting [3]) that fall short of full ranking. Elicitation of sufficient, but still *partial* information about voter rankings could alleviate some of these concerns.

The *elicitation* question has been studied from a theoretical perspective, addressing whether winners for some voting rules can be determined with incomplete voter preferences (rankings). Unfortunately, worst-case results are generally discouraging. Conitzer and Sandholm demonstrate that the communication complexity of several common voting protocols, such as Borda and Copeland, is $O(nm \log m)$, essentially requiring communication of full voter preferences in the worst-case [6]. Indeed, determining which votes to elicit to determine a winner is NP-hard in many schemes (e.g., Borda) [5; 18]. However, almost no work has addressed the question of practical vote elicitation—we discuss one exception, the recent work of Kalech et al. [9], in Sec. 5.

A question somewhat related to vote elicitation pertains to determining *necessary* and *possible* winners [11; 12; 19]. We define these problems formally below, but intuitively, given partial voter preferences, one asks whether alternative a *must* win under any completion of the preferences, or whether a *could* win under some completion. Both concepts are tightly related to questions of elicitation (see below).

Despite the theoretical complexity of partial elicitation, practical means of eliciting partial rankings and making decisions with incomplete preferences are vital. We first address winner determination under partial preferences, then turn our attention to elicitation.

3 Robust Winner Selection and Partial Profiles

Suppose we have only partial information about the preferences of some or all of the voters. We assume the *partial ranking* of a voter k takes a very general form, namely, a partially ordered set over domain A, or equivalently (the transitive closure of) a collection of *pairwise comparisons* of the form $a_i \succ a_j$. Most natural constraints on preferences, including the responses to natural queries (e.g., pairwise comparison, positional, top-t, and other queries) can be represented in this way.[1] Let p_k be the partial ranking of voter k. A *completion* of p_k is any vote v_k that extends p_k. Let $C(p_k)$ denote the set of *completions* of p_k, that is, the set of all (complete) votes v_k that extend p_k. We introduce the following notation: $Nec_k(a \succ b)$ iff $a \succ b$ in all completions of p_k; $Pos_k(a \succ b)$ iff $a \succ b$ in some completion of p_k; and $Inc_k(a, b)$ iff $Pos_k(a \succ b)$ and $Pos_k(b \succ a)$. An incomplete profile is a collection of partial votes $\mathbf{p} = \langle p_1, \ldots, p_n \rangle$. Let $C(\mathbf{p}) = C(p_1) \times \ldots \times C(p_n)$ be the set of *completions* of $\mathbf{p}$.

Let r be a voting rule. We define possible and necessary winners as follows [11; 12]: a is a *possible winner* under $\mathbf{p}$ iff there is some $\mathbf{v} \in C(\mathbf{p})$ such that $r(\mathbf{v}) = a$; and a is a *necessary winner* under $\mathbf{p}$ iff $r(\mathbf{v}) = a$ for all $\mathbf{v} \in C(\mathbf{p})$. Computationally, the possible winner question is typically hard (NP-complete), while necessary winner computation can be easy (polynomial time) or difficult (coNP-complete) depending on the rule [11; 19].

In the sequel, we assume the existence of a *scoring function* $s(a, \mathbf{v})$ that measures the quality of any candidate given a preference profile $\mathbf{v}$. Specifically, consider some voting rule $r : V \mapsto A$. We say a scoring function s is *consistent* with rule r iff $r(\mathbf{v}) \in \text{argmax}_{a \in A}\, s(a, \mathbf{v})$ for all $\mathbf{v} \in V$. This is, of course a minimal requirement, since any voting rule can be defined using an indicator function as the score. Many voting rules are, however, defined using a "natural" score. Indeed,

[1] One exception involves constraints that are naturally disjunctive, e.g., a response to the question "What alternative is ranked t^{th}?" cannot be mapped to a set of pairwise preferences unless the positions t are queried in ascending or descending order.

all rules discussed above have natural scoring functions (this fact is exploited in the algorithms for necessary winner determination developed in [19]). The notions of possible and necessary winners can be generalized when scores are used to allow for ties: a *necessary co-winner* is a candidate who has a maximum score in all possible completions of a partial profile (hence, even if the tie-breaking rule used by r goes against it, a is still "as good" a candidate as any winner); possible co-winners are defined similarly [19].

Suppose we have a partial profile $\mathbf{p}$ and we are forced to make a decision in the face of this incomplete information.[2] Necessary and possible winners do not resolve this issue satisfactorily: necessary winners often do not exist, and possible winners can only be used to narrow the set of options. We are unaware of any work that attempts to address the question of suggesting winners regardless of the degree of incompleteness of the votes. Here we propose the use of the *minimax regret* solution concept. This concept has been used for robust decision making, and for driving preference elicitation, in a variety of single-agent domains [1; 2; 4] and in mechanism design [8]; but our work is the first application of the notion to voting and (rank-based) social choice.

Intuitively, we measure the quality of any proposed winner a given $\mathbf{p}$ by considering how far from optimal a could be in the worst case (i.e., given any completion of $\mathbf{p}$). The minimax optimal solution is any alternative that is nearest to optimal in the worst case. More formally:

$$\begin{aligned} Regret(a, \mathbf{v}) &= max_{a' \in A} s(a', \mathbf{v}) - s(a, \mathbf{v}) && (1)\\ &= s(r(\mathbf{v}), \mathbf{v}) - s(a, \mathbf{v}) \\ PMR(a, a', \mathbf{p}) &= max_{\mathbf{v} \in C(\mathbf{p})} s(a', \mathbf{v}) - s(a, \mathbf{v}) && (2)\\ MR(a, \mathbf{p}) &= max_{\mathbf{v} \in C(\mathbf{p})} Regret(a, \mathbf{v}) \\ &= max_{a' \in A} PMR(a, a', \mathbf{p}) && (3)\\ MMR(\mathbf{p}) &= min_{a \in A} MR(a, \mathbf{p}) && (4)\\ a^*_{\mathbf{p}} &\in \operatorname*{argmin}_{a \in A} MR(a, \mathbf{p}) && (5) \end{aligned}$$

$Regret(a, \mathbf{v})$ is the loss (or regret) of selecting a as a winner instead of choosing the optimal alternative under rule r or score s.[3] $PMR(a, a', \mathbf{p})$ denotes the *pairwise max regret* of a relative to a' given partial profile $\mathbf{p}$, namely, the worst-case loss—under all possible realizations of the full profile—of selecting alternative a rather than a'. Notice that pairwise max regret can be negative. Max regret $MR(a, \mathbf{p})$ is the worst-case loss associated with a. We can view this as *adversarial selection* of a complete profile $\mathbf{v}$ to maximize the loss between our chosen alternative a and the true winner under $\mathbf{v}$. Our aim is to chose the a with *minimal max regret*: $MMR(\mathbf{p})$ denotes minimax regret under partial profile $\mathbf{p}$, while $a^*_{\mathbf{p}}$ denotes the minimax optimal alternative.[4] This gives us a form of robustness in the face of vote uncertainty: every alternative has worst-case error at least as great as that of $a^*_{\mathbf{p}}$.

[2] We distinguish our *partial information* setting from the question of aggregating preferences of voters whose preferences reflect genuine *incomparability* (see, e.g., [15]).

[3] See Smith [17] who uses score-based regret to measure the performance of various voting rules, including range voting.

[4] We informally write as if the optimal candidate is unique, but there can be several a that minimize max regret.

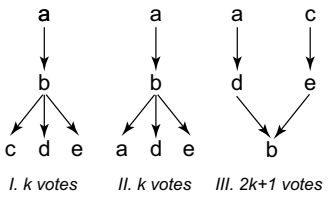

Fig. 1: A partial profile $\mathbf{p}$ where the minimax alternative is not a possible winner (under 2-approval). Alternative b has score $2k$ in every completion. Either a or c must be at the top of every vote in set III, so one must get at least $k+1$ points from set III. Hence $\max(s(a), s(c)) \geq 2k+1$, and a, c are possible winners, while b is not. Now, $MR(b, \mathbf{p}) = k+1$ (a completion that puts a at the top of all votes in set III would give a a score of $3k+1$, the maximum possible). But $MR(a, \mathbf{p}) = 2k+1$: if we select a, the adversary will place c and e above a in each vote in set III, setting $s(a) = k$ and $s(c) = 3k+1$. $MR(c, \mathbf{p}) = 2k+1$ by similar reasoning.

Notice that if $MMR(\mathbf{p}) = 0$, then the minimax winner $a^*_{\mathbf{p}}$ has the same score or utility as the winner in any completion $\mathbf{v} \in C(\mathbf{p})$; i.e., $a^*_{\mathbf{p}}$ is guaranteed to be optimal. While this does not imply there is a necessary winner under $\mathbf{p}$ (due to tie-breaking), $MMR(\mathbf{p}) = 0$ iff there is a necessary co-winner. Thus for any rule r we have, by setting $\varepsilon = 0$:

Observation 1. *The max regret decision problem (i.e., does alternative a have $MR(a, \mathbf{p}) \leq \varepsilon$) is at least as computationally hard as the necessary co-winner problem.*

This implies minimax regret is coNP-hard for, say, the Copeland and ranked pairs voting schemes [19]. It does *not* imply the easiness of max/minimax regret when the necessary co-winner problem is easy; but we describe polynomial time algorithms for minimax regret below. The relationship with possible winners is more complicated. For certain scoring rules (e.g., plurality) the minimax winner $a^*_{\mathbf{p}}$ must be a possible winner under $\mathbf{p}$. However, in general, we have:

Observation 2. *The regret-minimizing alternative may not be a possible winner for some voting rules.*

Fig. 1 shows this for the 2-approval scoring rule (where the top two candidates in each ranking receive a point): both possible winners have a poor score under *some* completion of the votes, while a compromise candidate that cannot win has a much higher guaranteed score (i.e., lower max regret) than either possible winner. This suggests that using the notion of possible winners to select winners with partial votes can be problematic for some voting rules. Indeed, there would appear to be no general way to ensure a possible winner isn't *far from being optimal* without using max regret to quantify this risk. The fact that the minimax winner $a^*_{\mathbf{p}}$ is not a possible winner is not problematic in our view, but if one insists on selecting from the set of possible winners, max regret could at least be used to aid in that selection (i.e., to choose the possible winner with least max regret). Still we take max regret to be the more fundamental notion for winner determination with partial information.

4 Computing Minimax Regret

Minimax regret can often be solved as a mixed integer program (MIP) [1; 2]. In our voting context, a MIP formulation

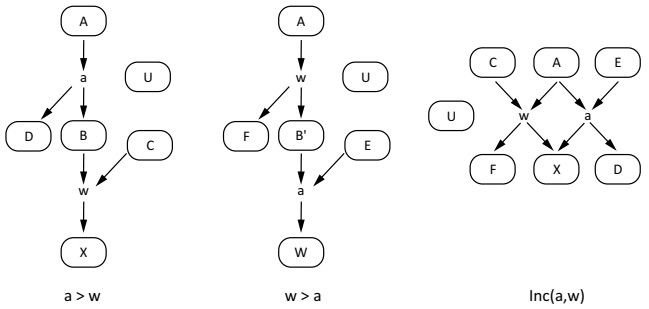

Fig. 2: An illustration of the three possible relations between alternative a and adversarial alternative/witness w in a partial vote p. To maximize a partial vote's contribution to pairwise max regret $PMR(a, w, \mathbf{p})$, linearizations of p require placing the groups of candidates indicated by rectangles in specific positions relative to a and w in a way that depends on the scoring rule.

(with variables capturing rank placement in specific votes) would be prohibitively expensive to solve. However, for certain voting rules and preference constraints, we can greatly simplify minimax regret computation by directly considering *properties* of worse-case completions of voter profiles without directly computing them. Our constructions are tightly related to those used by Xia and Conitzer [19] to demonstrate polynomial time algorithms for necessary winners for the positional scoring, maximin, and Bucklin rules. Indeed, their constructions can be viewed as attempting to maximize the difference in score between a proposed winner and an "adversarially chosen" alternative. We adapt these ideas to minimax regret, and extend the analysis to maxmin fairness.

To demonstrate the polynomial time computability of minimax regret, we explicitly compute the pairwise max regret $PMR(a, w, \mathbf{p})$ of all m^2 pairs of alternatives (a, w) (where a is a proposed winner and w is an adversarial witness). With PMR in hand, we can readily determine minimax regret using Eqs. 3 and 4. We then need only show that PMR can be computed in polynomial time.

A scoring rule is *(additively) decomposable* if $s(a, \mathbf{v}) = \sum_i s(a, v_i)$; i.e., it is the sum of votewise scores. This implies that regret is decomposable, since

$$\begin{aligned} \mathit{Regret}(a, w, \mathbf{v}) &= s(w, \mathbf{v}) - s(a, \mathbf{v}) && (6)\\ &= \sum_i s(w, v_i) - \sum_i s(a, v_i) && (7)\\ &= \sum_i [s(w, v_i) - s(a, v_i)]. && (8) \end{aligned}$$

Given a set of partial votes p_i, their completions by an adversary can be undertaken independently, so we can compute PMR by independently choosing completions v_i of each p_i that maximize v_i's local regret:

$$\begin{aligned} PMR(a, w, \mathbf{p}) &= \max_{\mathbf{v} \in C(\mathbf{p})} s(w, \mathbf{v}) - s(a, \mathbf{v}) && (9)\\ &= \sum_i \max_{v_i \in C(p_i)} s(w, v_i) - s(a, v_i). && (10) \end{aligned}$$

All positional scoring rules are decomposable in this way.

We illustrate our constructions by first examining the simple case of $PMR(a, w, \mathbf{p})$ for a *linear* positional scoring rule.[5] Since PMR is decomposable, we determine, for any partial vote p_i, the completion v_i with maximum contribution to PMR. Fig. 2 illustrates three different cases. In the first case, we have $Nec_i(a \succ w)$, so p_i's contribution to PMR must be negative: we maximize regret with a completion v_i that minimizes the positional gap between a and w (i.e., maximize the adversary's (negative) advantage). Note that all alternatives bear one of six distinct relationships to a and w.[6] To minimize the gap, it suffices to order set D below w (arbitrarily), set C above a, and U either above a or below w. The (negative) contribution to PMR is exactly $-(|B| + 1)$: we needn't compute an actual linearization of p_i, but simply determine the cardinality of set B. The case of $Nec_i(w \succ a)$ proceeds similarly (see figure), but instead we maximize the positional gap between w and a by placing sets F, E and U (arbitrarily) between w and a. Hence, the contribution to PMR by p_i is $|B' \cup F \cup E \cup U| + 1 = m - |A \cup W| - 1$. Finally, in the third case of $Inc_i(a, w)$, (positive) advantage is maximized by ordering w over a and placing sets E, F and U between the two.

Computing PMR thus requires, for each partial vote, categorizing all alternatives as belonging to the sets in Fig. 2. This is a simple matter: one compares each alternative a' to a and w, classifying it into the appropriate set, which takes $O(m)$ time, making $PMR(a, w, \mathbf{p})$ computable in $O(nm)$ time for linear (and nonlinear, see below) scoring rules. With m^2 pairs, computing $MMR(\mathbf{p})$ (and the optimal $a^*_{\mathbf{p}}$ and its witness) takes $O(nm^3)$ time. In many practical settings, m can be treated as a small constant relative to n, in which case our algorithms scale linearly in n.

With linear positional rules, arbitrary placement of alternatives that do not influence the positional gap between w and a (i.e., set U when $Nec_i(a \succ w)$) is allowed. For nonlinear rules, the size of the gap *and the position* of a, w can influence the advantage. However, the required placement can be found by simply examining splits of set U of different cardinalities to determine how many to place above a and below w to minimize a's advantage over w—again, this can be accomplished in $O(m)$ time. Certain special cases can be treated more efficiently; e.g., if a positional rule is *monotonic nonincreasing* (i.e., $s_i - s_{i+1} \geq s_{i+1} - s_{i+2}$) then U is placed above a (and if non-decreasing, below w). In any case, the minimax regret computation remains $O(nm^3)$.

PMR and minimax regret can be computed using independent completion of partial votes for non-decomposable scoring rules in some cases. Consider maxmin fairness, where $s(a, \mathbf{v}) = \min_i\{m - v_i(a)\}$. While minimizing or maximizing the score of a candidate in partial vote p_i is straightforward, the way in which *adversarial advantage* is maximized in p_i can depend on other votes. In the cases $Nec_i(w \succ a)$ and $Inc_i(a, w)$, there is only one way to maximize the local

[5] Linear implies that an alternative's score is a linear function of its rank in v, hence the *difference* in two rank positions uniquely determines their difference in score. Veto, approval, and plurality are not linear, but Borda is (linear rules are all "Borda-like").

[6] A is the set of alternatives (if any) preferred to both a and w; D are those less preferred than a but with unknown relation to w; U have unknown relation to both a and w; etc.

advantage of w over a (see above). But when $Nec_i(a \succ w)$, the placement of U either above a or below w influences the maximin score of a and w in a way that depends on other votes. However, one can show that unless $PMR(a, w, \mathbf{p})$ is negative, then advantage is maximized by ordering U *below* w. Informally, placing U below w can improve the min score of both a and w. However, this placement can only improve the min score of a if vote v_i gives a its min score over all p_i, in which case the min score of w is strictly less than that of a, and $PMR(a, w, \mathbf{p})$ is negative. Since max regret can never be negative, the pair (a, w) cannot define a's max regret. This lets us prove that, unless PMR is negative, $PMR(a, w, \mathbf{p})$ is maximized by ordering U below w in any p_i where $Nec_i(a \succ w)$. The running time for PMR is $O(nm)$ and for MMR is $O(nm^3)$, as in the case of positional scoring rules, since we only need to identify the relevant sets in Fig. 2.

We consider to other *nondecomposable rules*, maximin and Bucklin, which require more intricate computation. To compute pairwise max regret for the maximin rule, one must also consider possible alternatives $a' \in A \backslash \{a, w\}$. For any partial vote p_i, the adversarial construction is the same as for positional scoring rules in the cases $Nec_i(w \succ a)$ and $Inc_i(a, w)$. The case $Nec_i(a \succ w)$ is different: one must consider, for each a', the adversarial placement of a' over a, if possible, and everything else below w. This gives $O(nm^2)$ time for computing PMR and consequently $O(nm^4)$ time for computing minimax regret. For Bucklin, to compute pairwise max regret one asks questions of the form "Is there a completion where at least half of the voters rank a below position j *and* more than half of the voters rank w at or above rank j'?" Such questions can be answered in polynomial time, and again the adversarial constructions can be done vote-wise, with the most involved case being $Nec_i(a \succ w)$, where one must check whether (and which of) a or w can be ranked above/below j or j'. These constructions are again similar to those of Xia and Conitzer [19]. We defer details to a longer version of this paper.

5 Vote Elicitation

We now turn to vote elicitation. As discussed above, while elicitation can be difficult (w.r.t. computation and communication complexity) in the worst case, minimax regret can be used effectively to guide the elicitation process. As a valuable measure of solution quality, it can be used to terminate elicitation whenever regret falls to some suitable threshold (including zero if optimality is desired).[7] More importantly, the solution to the minimax optimization can guide the selection of queries (and voters to ask) so that an (approximate or exact) optimal solution can be found quickly. In this section, we describe a simple heuristic strategy to do just this. We focus on linear positional rules (Borda-like) and two specific query types; but these ideas generalize to other rules and other forms of queries. (These generalizations are described in a longer version of this paper.)

[7] If determining optimal termination is hard [5; 18], then so is minimax regret; or equivalently, if computing minimax regret is easy (as demonstrated for certain rules above), so is termination.

The Current Solution Strategy (CSS). We consider two forms of queries. A *comparison query* asks a voter k to compare two alternatives: "Is $a \succ_k b$?". A *top-t query* asks voter k to state which alternative is t^{th} in their ranking (we assume that the first $t-1$ alternatives have already been articulated by k). We describe our key heuristic below using comparison queries, but the intuitions are easily adapted to the selection of top-t queries (see our experiments).

The *current solution strategy* generates queries by considering the current solution to the minimax optimization—i.e., the minimax optimal alternative a and adversarial witness w—and using this to choose a voter-query pair with greatest potential to reduce minimax regret. Notice that if the advantage of w over a is not reduced in some partial vote p_k in response to a query, $PMR(a, w)$ will not change, thus, unless the response changes the minimax optimal solution, MMR will not change. So CSS selects queries that tackle this gap directly. We determine the value of posing a query to voter k, by considering the three cases in Fig. 2, in each case determining the query with the largest potential reduction given a positive response by k:

Case 1: $a \succ w$: We can reduce $PMR(a, w)$ by asking two different types of queries: $d \succ w$ for some $d \in D$ or $a \succ c$ for $c \in C$. In each case, a positive response will position alternatives between a and w, thus reducing $PMR(a, w)$ by increasing the (worst-case) position of a relative to w in p_k. We pick the alternative in $C \cup D$ (and voter) with greatest potential to reduce PMR (i.e., in the case of a positive response). For linear rules, this potential is measured by the number of ancestors of d in set D (all of which will be positioned between a and w if d is), and the number of descendents of c in set C. If $C \cup D = \emptyset$, we can ask two other query types, $u \succ w$ or $a \succ u$ for some $u \in U$. These do not reduce PMR directly, but move u and its ancestors in U to set C (for query $u \succ w$) or u and its descendents in U to set D (for query $a \succ u$). The u that can move the greatest number of items within U to some other set is chosen.

Case 2: $w \succ a$: We can reduce $PMR(a, w)$ by asking four different types of queries: $a \succ f$ for some $f \in F$; $a \succ u$ for $u \in U$; $e \succ w$ for some $e \in E$; or $u \succ w$ for some $u \in U$. A positive response to any such query will reduce PMR by increasing the (worst-case) score of a in p_k or reducing that of w. Selection is again made by picking the alternative, query and voter that will have the greatest potential (i.e. in the case a positive response is received) reduction in PMR.

Case 3: $Inc(a, w)$: We can reduce $PMR(a, w)$ by asking several different queries, however, heuristically, we always choose to ask if $a \succ w$, since a positive response reverses p_k's contribution to PMR from positive to negative. *Any* response will move partial vote p_k into either case 1 or case 2.

CSS must eventually terminate with an optimal solution:

Proposition 3. *Unless $MMR = 0$, CSS will always select a voter k and comparison query $a_i \succ_k a_j$ s.t. $Inc_k(a_i, a_j)$.*

If partial vote p_k falls into Case 3, this is obvious. In cases 1, 2, we can show that at least one of the designated sets for some voter must be nonempty if $MMR > 0$.

CSS can be adapted, using similar intuitions, to generate top-t queries (see a longer version of the paper). Top-t queries

are asked of any voter in order: so no voter is asked for the second-ranked candidate before their first, their third before their second, etc. Note that CSS need only select a voter at any stage, not a query.

As benchmarks in our experiments, we use two other strategies as well. The *random strategy (Rand)* randomly chooses a voter k and a comparison query such that $Inc_k(a_i, a_j)$ (so the query response always bears information). With top-t queries, Rand only needs to choose voter k at random. The *volumetric strategy (Vol)* selects a voter k and query $a_i \succ a_j$ that maximizes the number of *new* pairwise preferences revealed (given the worst response):

$$Vol(p_k) = \max_{a_i,a_j} \min \left\{ \begin{array}{l} |tc(p_k \cup \{a_i \succ a_j\})|, \\ |tc(p_k \cup \{a_j \succ a_i\})| \end{array} \right\},$$

where tc denotes transitive closure. This strategy reduces preference uncertainty maximally, without regard for "relevance" to winner determination (much like volumetric strategies for single-agent settings). Its application to top-t queries involves selecting the voter whose next-ranked candidate reduces uncertainty the most: since this voter *must* be one who has ranked the fewest candidates, the strategy reduces to a simple sequential iteration where each voter is asked for their top-ranked candidate, then each is asked for their second-ranked, and so on. We dub Vol in this case SequentialTop.

In very recent work, Kalech et al. [9] developed two elicitation algorithms for winner determination with scoring-based rules (e.g., Borda, Range voting). This seems to be the first significant investigation of practical incremental elicitation. Their first is essentially the SequentialTop method, which proceeds in rounds in which each voter is queried for their next most preferred choice. It uses necessary winner computation for termination. This contrasts with our CSS approach, which is much more subtle and incremental: we identify a *particular voter* to query at each stage by evaluating its potential to reduce minimax. We see in our experiments that this can reduce the number of required queries substantially. Furthermore, our elicitation methods are anytime: querying can terminate when minimax regret is sufficiently small, and we show below that this further reduces the number of queries significantly.

Their second algorithm proceeds for a predetermined number of rounds, asking each voter at each stage for fixed number of positional rankings. Since termination is predetermined, necessary winners may not result (instead possible winners are returned), and interesting tradeoffs between the number of rounds and amount of information per round are explored. One attractive feature of this model is the batching of queries (voters are only queried a fixed, ideally small, number of times, though each query may request a lot of information), thus minimizing interruption, waiting time, etc. As the authors acknowledge, this scheme provides no guarantee of winner optimality or any bounds on quality. A key advantage of our minimax regret-based schemes is that a natural, precise objective is being minimized, and anytime quality guarantees are provided.[8]

[8]We also note that elicitation of pairwise preferences is not considered in [9]; such queries are extremely valuable and arise naturally in many domains such as search, IR, consumer product comparisons, etc.

Experiments We test CSS on three datasets: (a) *Sushi*, [10], with 5000 preference rankings over 10 varieties of sushi; (b) *Irish*, with 2002 voting data from the Dublin North constituency, comprising 3662 rankings over 12 candidates;[9] and (c) *Mallows*, 100 random rankings over 20 items generated from the *Mallows preference model* [13].[10] These datasets were used to generate responses to elicitation queries, and span both political voting and recommender systems.

We tested CSS on each data set, using both paired and top-t queries, assuming Borda voting (similar results hold for other rules), and compared it to the random and volumetric elicitation strategies on the two real-world sets. Fig. 3 shows MMR as a function of the number of queries asked (both paired and top-t). On both Sushi and Irish, CSS offers superior elicitation performance with both paired and top-t queries. With Sushi CSS reaches the optimal solution (i.e., the provable winner with $MMR = 0$) after an average of only 11.82 paired queries per voter (cf. 20.64 for Vol, 20.63 for Rand, and 25 queries required by the theoretically optimal MergeSort to determine full voter rankings), and after 3.40 top-t queries per voter (cf. 4.18 for Seq, 5.50 for Rand). With Irish, results are similar: CSS reaches optimality with 18.57 paired and 5.47 top-t queries per user (cf. 31.82, 6.91 for Vol/Seq; 31.22, 8.38 for Rand, 33 for MergeSort). Note that top-t queries are "information rich" as they provide many pairwise comparisons per response. Thus, while CSSTop's advantage is somewhat less (though RandomTop still asks over 50% more queries to reach $MMR = 0$ in Irish), the fact that there is an advantage is of greater significance. Critically, if one is interested in approximate solutions, we see that CSS reduces MMR very quickly. For example, with Irish, CSS reduces MMR to 18% of its initial value (with no voter preference data) after only 5.82 paired queries per voter (cf. 25.77 for Vol, 24.03 for Rand), a small fraction of the queries required to elicit full rankings. CSS took a few milliseconds on average (wall clock time) to find the best agent/comparison query (including time needed to recompute the MMR-solution).

On the synthetic Mallows data set, we sampled 10 complete voter profiles for each value of ϕ and run CSS. With larger ϕ, more queries are clearly needed to reach the same level of regret, which conforms to our intuitions that intelligent elicitation schemes can take significant advantage of less uniform preferences to minimize queries and voter effort (and conversely, that with almost uniformly random preferences, nearly full rankings must be obtained). Work in behavioral social choice strongly suggests that real-world preferences are not uniformly random [16], and CSS seems to perform especially well in this case; indeed our results on Sushi and Irish suggest that real preferences are not uniform, and contain regularities that can be readily exploited to reduce the informational complexity of voting.

[9]The data has $43,942$ top-t ballots; 3662 are complete (i.e., $t = 12$). See `www.dublincountyreturningofficer.com`.

[10]Mallows is a distribution over rankings given by a modal ranking σ and dispersion $\phi \in (0, 1]$ with $\Pr(r|\sigma, \phi) \propto \phi^{d(r,\sigma)}$, where d is Kendall's τ-distance. Smaller ϕ concentrates mass around σ, while $\phi = 1$ gives the uniform distribution.

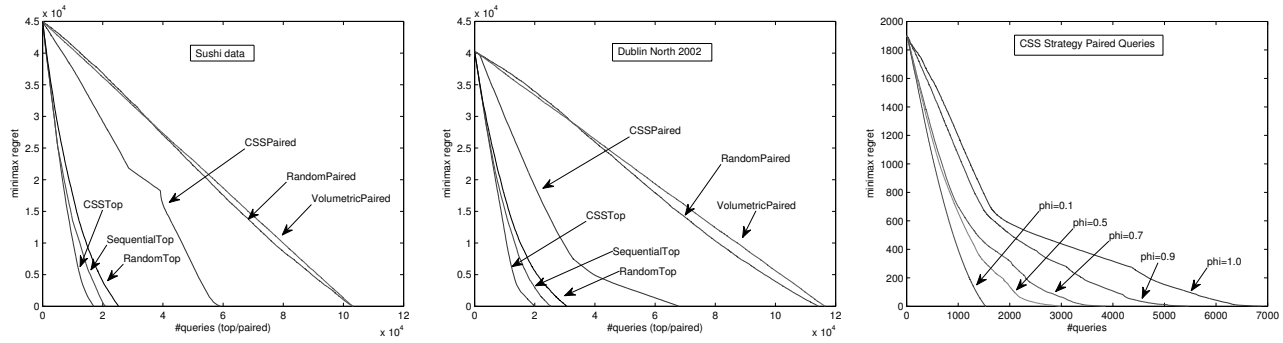

Fig. 3: Performance of elicitation algorithms (paired and top queries) on Sushi, Dublin and Mallows data.

6 Concluding Remarks

We have proposed the use of minimax regret as a means of robust winner determination to support the informational approximation of voting rules, as well as to guide the process of incremental elicitation of voter preferences. We demonstrated the tractability of regret computation for a collection of common voting rules, and demonstrated the power of regret-based elicitation on two real-world data sets and on synthetic data. Specifically, regret-based elicitation allows one to determine both approximate and exact winners using only a small fraction of (pairwise) voter preferences.

While our results suggest that incremental elicitation is viable in many practical domains, a number of interesting avenues for future research remain. Apart from developing computational and elicitation schemes for additional voting rules, we are currently extending our approach to multiattribute domains, which are especially relevant in recommender systems and product configuration. We are also developing formalisms, analytical techniques, and elicitation schemes that are tuned to the demands of particular population preference distributions (e.g., Mallows, Plackett-Luce, etc.). Such a probabilistic framework would allow for more subtle analysis of elicitation performance and new elicitation heuristics. The models could be purely (Bayesian) decision-theoretic; but more interesting analyses would mix probabilistic and regret-based reasoning (e.g., the expected number of queries needed to determine a winner or to reduce minimax regret to some acceptable threshold). This is an important step in making incremental vote elicitation practical in real-world settings. There are also interesting questions connecting elicitation to vote manipulation, where the elicitation process may reveal preference information that a manipulating coalition can exploit (see also [5]).

Acknowledgements: Thanks to Lirong Xia and Vincent Conitzer for helpful discussions. We also thank the reviewers for their helpful comments. This work was supported by NSERC.

References

[1] C. Boutilier, R. Patrascu, P. Poupart, and D. Schuurmans. Constraint-based optimization and utility elicitation using the minimax decision criterion. *Art. Intel.*, 170:686–713, 2006.

[2] C. Boutilier, T. Sandholm, and R. Shields. Eliciting Bid Taker Non-price Preferences in (Combinatorial) Auctions. *AAAI-04*, pp.204–211, San Jose, 2004.

[3] S. Brams and P. Fishburn. Approval voting. *Amer. Pol. Sci. Rev.*, 72(3):831–847, 1978.

[4] D. Braziunas and C. Boutilier. Assessing regret-based preference elicitation with the UTPREF recommendation system. *ACM EC'10*, pp.219–228, Cambridge, MA, 2010.

[5] V. Conitzer and T. Sandholm. Vote elicitation: Complexity and strategy-proofness. *AAAI-02*, pp.392–397, Edmonton, 2002.

[6] V. Conitzer, T. Sandholm. Communication complexity of common voting rules. *ACM EC'05*, pp.78–87, Vancouver, 2005.

[7] W. Gaertner. *A Primer in Social Choice Theory*. Oxford, 2006.

[8] N. Hyafil and C. Boutilier. Mechanism design with partial revelation. *IJCAI-07*, pp.1333–1340, Hyderabad, 2007.

[9] M. Kalech, S. Kraus, G. Kaminka and C. Goldman Practical voting rules with partial information. *Journal of Autonomous Agents and Multi-Agent Systems*, 22:151–182, 2011.

[10] T. Kamishima, H. Kazawa, and S. Akaho. Supervised ordering: An empirical survey. *IEEE Intl. Conf. on Data Mining*, 673–676, 2005.

[11] K. Konczak and J. Lang. Voting procedures with incomplete preferences. *IJCAI-05 Workshop on Advances in Preference Handling*, pp.124–129, Edinburgh, 2005.

[12] J. Lang. Vote and aggregation in combinatorial domains with structured preferences. *IJCAI-07*, pp.1366–1371, Hyderabad, 2007.

[13] C. L. Mallows. Non-null ranking models. *Biometrika*, 44:114–130, 1957.

[14] H. Nurmi. *Voting Procedures Under Uncertainty*. Springer, Berlin, 2002.

[15] M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Aggregating partially ordered preferences. *J. Logic and Comp.*, 19:475–502, 2009.

[16] M. Regenwetter, B. Grofman, A. A. J. Marley, and I. Tsetlin. *Behavioral Social Choice*. Cambridge, 2006.

[17] W. Smith. Range voting. `http://www.math.temple.edu/~wds/homepage/rangevote.pdf`, 2000.

[18] T. Walsh. Complexity of terminating preference elicitation. *AAMAS-08)*, pp.967–974, Estoril, PT, 2008.

[19] L. Xia and V. Conitzer. Determining possible and necessary winners under common voting rules given partial orders. *AAAI-08*, pp.202–207, Chicago, 2008.