Preference Elicitation for Single Crossing Domain

Palash Dey
Indian Institute of Science, Bangalore
palash@csa.iisc.ernet.in

Neeldhara Misra
Indian Institute of Technology, Gandhinagar
mail@neeldhara.com

Abstract

Eliciting the preferences of a set of agents over a set of alternatives is a problem of fundamental importance in social choice theory. Prior work on this problem has studied the query complexity of preference elicitation for the unrestricted domain and for the domain of single peaked preferences. In this paper, we consider the domain of single crossing preference profiles and study the query complexity of preference elicitation under various settings. We consider two distinct situations: when an ordering of the voters with respect to which the profile is single crossing is *known* versus when it is *unknown*. We also consider random and sequential access models. The main contribution of our work is to provide polynomial time algorithms with low query complexity for preference elicitation in all the above cases. Further, we show that the query complexities of our algorithms are optimal up to constant factors for all but one of the above cases.

1 Introduction

Agents in multiagent systems often have individual preferences which are complete orders over a set of candidates and one would like to find an aggregate ranking or choose the *"best"* candidate. Classic examples where such a scenario appears are collaborative filtering [Pennock *et al.*, 2000] etc. In a typical such setting, we have a set of agents or voters; each of them has a preference, called a vote, over a set of candidates; and a voting rule (respectively an aggregation function) which finds a candidate (respectively an aggregated preference) called winner.

However, eliciting the preferences of the agents is a nontrivial task since we often have a large number of candidates (ranking restaurants for example) and it may often be infeasible for the agents to rank all of them. Hence it becomes important to elicit the preferences of the agents by asking them (hopefully a small number of) comparison queries only - ask an agent i to compare two candidates x and y.

Unfortunately, it turns out that one would need ask every voter $\Omega(m \log m)$ queries to know her preference, and this argument is based on the decision-tree based lower bound on the number of comparisons for sorting an array. However, if the preferences are not completely arbitrary, but admit additional structure, then possibly we can do better. Indeed, an affirmation of this thought comes from the work of [Conitzer, 2009], who showed that we can elicit preferences using only $O(m)$ queries per voter for what are called *single peaked preferences* (which is a well-studied restriction on preferences, and we will define the notion formally later in this section).

There are two reasons for restricting the domain of preferences. The first is that in several application scenarios commonly considered, it is rare that votes are ad-hoc, demonstrating no patterns whatsoever. For example, the notion of single-peaked preferences forms the basis of several studies in the analytical political sciences [Hinich and Munger, 1997].

The second motivation for studying restricted preferences comes from the fact that they are very well-behaved from a theoretical perspective as well. The axiomatic approach of social choice involves defining certain "properties" that formally capture the quality of a voting rule. For example, we would not want a voting rule to be, informally speaking, a *dictatorship*, which would essentially mean that it discards all but one voter's input. As it happens, a series of cornerstone results establish that it is impossible to devise voting rules which respect some of the simplest desirable properties. Celebrated results in social choice theory [Arrow, 1950; Gibbard, 1973; Satterthwaite, 1975] show that it is impossible to have an aggregation function or a voting rule that simultaneously satisfies a desirable set of properties, for example, strategy-proofness, ontoness, non-dictatorship etc. We refer the reader to [Moulin, 1991] for an excellent exposition of all key issues that arise in this context. Adding to the difficulties is the fact that many important voting rules such as Kemeny [Kemeny, 1959; Levenglick, 1975], Dodgson [Dodgson, 1876; Black *et al.*, 1958], and Young [Young, 1977] are computationally intractable [Bartholdi III *et al.*, 1989; Hemaspaandra *et al.*, 2005; Procaccia *et al.*, 2008].

This brings us to the second important reason for considering structured preferences — they provide a very elegant workaround to the difficulties that we outlined above. The domains of single peaked and single crossing profiles are arguably the most important and well-studied domains among such restricted domains [Saporiti and Tohmé, 2006; Ballester and Haeringer, 2011; Cornaz *et al.*, 2013; Skowron *et al.*, 2013; Magiera and Faliszewski, 2014; Faliszewski *et* *al.*, 2014; Brandt *et al.*, 2015, and references therein]. A profile is called *single peaked* if the candidates can be arranged in a complete order $\succ$ such that every preference $\succ'$ in the profile *respects* the order $\succ$ in the sense that, for every two candidates x and y, we have $x \succ' y$ whenever we have either $c \succ x \succ y$ or $y \succ x \succ c$, where c is the most preferred candidate in $\succ'$ [Black, 1948]. On the other hand a profile is called *single crossing* if the voters can be arranged in a complete order $\succ$ such that for every two candidates x and y, all the voters who prefer x over y appear consecutively in $\succ$ [Mirrlees, 1971; Roberts, 1977].

Elicitation on Restricted Domains. Conitzer and Sandholm show that determining whether we have enough information at any point of the elicitation process for finding a winner under some common voting rules is computationally intractable [Conitzer and Sandholm, 2002]. They also prove in their classic paper [Conitzer and Sandholm, 2005] that one would need to make $\Omega(mn\log m)$ queries even to decide the winner for many commonly used voting rules which matches with the trivial $\mathcal{O}(mn\log m)$ upper bound (based on sorting) for preference elicitation in unrestricted domain. Dey and Misra proved tight query complexity bounds for preferences single peaked on trees with respect to various tree parameters [Dey and Misra, 2016a].

A natural question at this point is if these restricted domains allow for better elicitation algorithms as well. The answer to this is in the affirmative, and one can indeed elicit the preferences of the voters using only $\mathcal{O}(mn)$ many queries for the domain of single peaked preference profiles [Conitzer, 2009]. Our work belongs to this kind of research– we investigate the number of queries one has to ask for preference elicitation in single crossing domains. When some partial information is available about the preferences, Ding and Lin prove interesting properties of what they call a deciding set of queries [Ding and Lin, 2013]. Lu and Boutilier empirically show that several heuristics often work well [Lu and Boutilier, 2011b; 2011a].

Contributions. In this paper we present novel algorithms for preference elicitation for the domain of single crossing profiles in various settings. We consider two distinct situations: when an ordering of the voters with respect to which the profile is single crossing is *known* versus when it is *unknown*. We also consider different access models: when the votes can be accessed at random, as opposed to when they are coming in a pre-defined sequence. In the sequential access model, we distinguish two cases when the ordering is known: the first is that sequence in which the votes appear is also a single-crossing order, versus when it is not. We also prove lower bounds on the query complexity of preference elicitation for the domain of single crossing profiles; these bounds match the upper bounds up to constant factors (for a large number of voters) for all the six scenarios above except the case when we know a single crossing ordering of the voters and we have a random access to the voters; in this case, the upper and lower bounds match up to a factor of $\mathcal{O}(m)$. We summarize our results in Table 1.

2 Preliminaries

For a positive integer ℓ, we denote the set $\{1,\ldots,\ell\}$ by $[\ell]$. Let $\mathcal{V} = \{v_i : i \in [n]\}$ be a set of n *voters* and $\mathcal{C} = \{c_j : j \in [m]\}$ be a set of m *candidates*. If not mentioned otherwise, we denote the set of candidates, the set of voters, the number of candidates, and the voters by $\mathcal{C}$, $\mathcal{V}$, m, and n respectively. Every voter v_i has a *preference* $\succ_i$ which is a complete order over the set $\mathcal{C}$ of candidates. We say voter v_i prefers a candidate $x \in \mathcal{C}$ over another candidate $y \in \mathcal{C}$ if $x \succ_i y$. We denote the set of all preferences over $\mathcal{C}$ by $\mathcal{L}(\mathcal{C})$. The n-tuple $(\succ_i)_{i\in[n]} \in \mathcal{L}(\mathcal{C})^n$ of the preferences of all the voters is called a *profile*. We say a candidate x is *placed at the k^{th} position* of a preference $\succ$ if x is preferred over all but exactly $(k-1)$ other candidates in $\succ$. Let $\mathbb{S}_n$ denote the set of all permutations over $[n]$ and id_n be the identity permutation of $[n]$. Let an ordering σ be $x_1 \succ x_2 \succ \cdots \succ x_\ell$. Then by $\overleftarrow{\sigma}$ we denote the ordering $x_\ell \succ \cdots \succ x_2 \succ x_1$. All the logarithms in this paper are base 2 unless specified otherwise.

Single Crossing Domain A profile $\mathcal{P} = (\succ_1,\ldots,\succ_n)$ of n voters over a set $\mathcal{C}$ of candidates is called a single crossing profile if there exists a permutation $\sigma \in \mathbb{S}_n$ of $[n]$ such that, for every two distinct candidates $x,y \in \mathcal{C}$, whenever we have $x \succ_{\sigma(i)} y$ and $x \succ_{\sigma(j)} y$ for two integers i and j with $1 \leqslant i < j \leqslant n$, we have $x \succ_{\sigma(k)} y$ for every $i \leqslant k \leqslant j$. The following observation is immediate from the definition of single crossing profiles.

Observation 1 *Suppose a profile $\mathcal{P}$ is single crossing with respect to an ordering $\sigma \in \mathbb{S}_n$ of votes. Then $\mathcal{P}$ is single crossing with respect to the ordering $\overleftarrow{\sigma}$ too.*

Problem Formulation Suppose we have a profile $\mathcal{P}$ with n voters and m candidates. Let us define a function $\textsc{Query}(x \succ_\ell y)$ for a voter ℓ and two different candidates x and y to be TRUE if the voter ℓ prefers the candidate x over the candidate y and FALSE otherwise. We now formally define the problem.

Definition 1 PREFERENCE ELICITATION
Given an oracle access to $\textsc{Query}(\cdot)$ *for a single crossing profile $\mathcal{P}$, find $\mathcal{P}$.*

For two distinct candidates $x,y \in \mathcal{C}$ and a voter ℓ , we say a PREFERENCE ELICITATION algorithm $\mathcal{A}$ *compares* candidates x and y for voter ℓ , if $\mathcal{A}$ makes a call to either $\textsc{Query}(x \succ_\ell y)$ or $\textsc{Query}(y \succ_\ell x)$. We define the number of queries made by the algorithm $\mathcal{A}$, called the *query complexity* of $\mathcal{A}$, to be the number of distinct tuples $(\ell,x,y) \in \mathcal{V}\times\mathcal{C}\times\mathcal{C}$ with $x \neq y$ such that the algorithm $\mathcal{A}$ compares the candidates x and y for voter ℓ . Notice that, even if the algorithm $\mathcal{A}$ makes multiple calls to $\textsc{Query}(\cdot)$ with same tuple (ℓ,x,y), we count it only once in the query complexity of $\mathcal{A}$. This is without loss of generality since we can always implement a wrapper around the oracle which memorizes all the calls made to the oracle so far and whenever it receives a duplicate call, it replies from its memory without "actually" making a call to the oracle. We say two query complexities $\mathsf{q}(m,n)$ and $\mathsf{q}'(m,n)$ are tight up to a factor of ℓ *for a large number of voters* if $1/\ell \leqslant \lim_{n\to\infty} \mathsf{q}(m,n)/\mathsf{q}'(m,n) \leqslant \ell$.

Ordering	Access model	Query Complexity	
		Upper Bound	Lower Bound
Known	Random	$\mathcal{O}(m^2 \log n)$ Lemma 1	$\Omega(m \log m + m \log n)$ Theorem 1
	Sequential single crossing order	$\mathcal{O}(mn + m^2)$ Theorem 2	$\Omega(m \log m + mn)$ Theorem 3
	Sequential any order	$\mathcal{O}(mn + m^2 \log n)$ Theorem 4	
Unknown	Sequential any order	$\mathcal{O}(mn + m^3 \log m)$ Theorem 5	
	Random	$\mathcal{O}(mn + m^3 \log m)$ Corollary 1	$\Omega(m \log m + mn)$ Theorem 6

Table 1: Summary of Results for preference elicitation for single crossing profiles.

Note that by using a standard sorting routine like merge sort, we can fully elicit an unknown preference using $\mathcal{O}(m \log m)$ queries. We state this explicitly below, as it will be useful in our subsequent discussions.

Observation 2 *There is a* PREFERENCE ELICITATION *algorithm for eliciting a single preference with query complexity $\mathcal{O}(m \log m)$.*

Model of Input We study two models of input for PREFERENCE ELICITATION for single crossing profiles.

- **Random access to voters:** In this model, we have a set of voters and we are allowed to ask any voter to compare any two candidates at any point of time. Moreover, we are also allowed to interleave the queries to different voters. Random access to voters is the model of input for elections within an organization where every voter belongs to the organization and can be queried any time.
- **Sequential access to voters:** In this model, voters are arriving in a sequential manner one after another to the system. Once a voter ℓ arrives, we can query voter ℓ as many times as we like and then we "release" the voter ℓ from the system to grab the next voter in the queue. Once voter ℓ is released, it can never be queried again. Sequential access to voters is indeed the model of input in many practical elections scenarios such as political elections, restaurant ranking etc.

3 Results

In this section, we present our technical results. In the interest of space, we defer the proofs of a few results to the full version of the paper [Dey and Misra, 2016b]. We first consider the (simpler) situation when the single crossing order is known, and then turn to the case when the order is unknown. In both cases, we explore all the relevant access models.

3.1 Results: Known Single Crossing Order

We begin with a simple PREFERENCE ELICITATION algorithm when we are given a random access to the voters and a single crossing ordering is known.

Lemma 1 *Suppose a profile $\mathcal{P}$ is single crossing with respect to a known permutation of the voters. Given a random access to voters, there is a* PREFERENCE ELICITATION *algorithm with query complexity $\mathcal{O}(m^2 \log n)$.*

Proof: By renaming, we assume that the profile is single crossing with respect to the identity permutation of the votes. Now, for every $\binom{m}{2}$ pair of candidates $\{x, y\} \subset \mathcal{C}$, we perform a binary search over the votes to find the index $i(\{x, y\})$ where the ordering of x and y changes. We now know how any voter j orders any two candidates x and y from $i(\{x, y\})$ and thus we have found $\mathcal{P}$. □

Interestingly, the simple algorithm in Lemma 1 turns out to be optimal up to a multiplicative factor of $\mathcal{O}(m)$ as we prove next. The idea is to "pair up" the candidates and design an oracle which "hides" the vote where the ordering of the two candidates in any pair (x, y) changes unless it receives at least $(\log m - 1)$ queries involving only these two candidates x and y. We formalize this idea below.

Theorem 1 *Suppose a profile $\mathcal{P}$ is single crossing with respect to the identity permutation of votes. Given random access to voters, any* PREFERENCE ELICITATION *algorithm has query complexity $\Omega(m \log m + m \log n)$.*

Proof: The $\Omega(m \log m)$ bound follows from the query complexity lower bound of sorting and the fact that any profile consisting of only one preference $\succ \in \mathcal{L}(\mathcal{C})$ is single crossing. Let $\mathcal{C} = \{c_1, \ldots, c_m\}$ be the set of m candidates where m is an even integer. Consider the ordering $Q = c_1 \succ c_2 \succ \cdots \succ c_m \in \mathcal{L}(\mathcal{C})$ and the following pairing of the candidates: $\{c_1, c_2\}, \{c_3, c_4\}, \ldots, \{c_{m-1}, c_m\}$. Our oracle answers QUERY$(\cdot)$ as follows. The oracle fixes the preferences of the voters one and n to be Q and $\overleftarrow{Q}$ respectively. For every odd integer $i \in [m]$, the oracle maintains θ_i (respectively β_i) which corresponds to the largest (respectively smallest) index of the voter for whom (c_i, c_{i+1}) has already been queried and the oracle answered that the voter prefers c_i over c_{i+1} (c_{i+1} over c_i respectively). The oracle initially sets $\theta_i = 1$ and $\beta_i = n$ for every odd integer $i \in [m]$. Suppose oracle receives a query to compare candidates c_i and c_j for $i, j \in [m]$ with $i < j$ for a voter ℓ. If i is an even integer or $j - i \geqslant 2$ (that is, c_i and c_j belong to different pairs), then the oracle answers that the voter ℓ prefers c_i over c_j. Otherwise we have $j = i + 1$ and i is an odd integer. The oracle answers the query to be $c_i \succ c_{i+1}$ and updates θ_i to ℓ keeping β_i fixed if $|\ell - \theta_i| \leqslant |\ell - \beta_i|$ and otherwise answers $c_{i+1} \succ c_i$ and updates β_i to ℓ keeping θ_i fixed (that is, the oracle answers according to the vote which is closer to the voter ℓ between θ_i and β_i and updates θ_i or β_i accordingly). If the pair (c_i, c_{i+1}) is queried less than $(\log n - 2)$ times, then we have $\beta_i - \theta_i \geqslant 2$ at the end of the algorithm since every query for the pair (c_i, c_{i+1}) decreases $\beta_i - \theta_i$ by at most a factor of two and we started with $\beta_i - \theta_i = n - 1$. Consider a voter κ with $\theta_i < \kappa < \beta_i$. If the elicitation algorithm outputs that the voter κ prefers c_i over c_{i+1} (respectively c_{i+1} over c_i), then the oracle sets all the voters κ' with $\theta_i < \kappa' < \beta_i$ to prefer c_{i+1} over c_i (respectively c_i over c_{i+1}). Clearly, the algorithm does not elicit the preference of the voter κ correctly. Also, the profile is single crossing with respect to the identity permutation of the voters and consistent with the answers of all the queries made by the algorithm. Hence, for every odd integer $i \in [m]$, the algorithm must make at least $(\log n - 1)$ queries for the pair (c_i, c_{i+1}) thereby making $\Omega(m \log n)$ queries in total. □

We now present our PREFERENCE ELICITATION algorithm when we have a sequential access to the voters according to a single crossing order. We elicit the preference of the first voter using Observation 2. From second vote onwards, we simply use the idea of *insertion sort* relative to the previously elicited vote [Cormen, 2009]. Since we are using insertion sort, any particular voter may be queried $\mathcal{O}(m^2)$ times. However, we are able to bound the query complexity of our algorithm due to two fundamental reasons: (i) consecutive preferences will often be almost similar in a single crossing ordering, (ii) our algorithm takes only $\mathcal{O}(m)$ queries to elicit the preference of the current voter if its preference is indeed the same as the preference of the voter preceding it. In other words, every time we have to "pay" for shifting a candidate further back in the current vote, the relative ordering of that candidate with all the candidates that it jumped over is now fixed, because for these pairs, the one permitted crossing is now used up. We begin with presenting an important subroutine called Elicit(·) which finds the preference of a voter ℓ given another preference $\mathcal{R}$ by performing an insertion sort using $\mathcal{R}$ as the order of insertion.

Algorithm 1 Elicit($\mathcal{C}$, $\mathcal{R}$, ℓ)

```
Input: A set of candidates $\mathcal{C} = \{c_i : i \in [m]\}$, an ordering
    $\mathcal{R} = c_1 \succ \cdots \succ c_m$ of $\mathcal{C}$ , a voter $\ell$
Output: Preference ordering $\succ_\ell$ of voter $\ell$ on $\mathcal{C}$
1: $\mathcal{Q} \leftarrow c_1$        ▷ $\mathcal{Q}$ will be the preference of the voter $\ell$
2: for $i \leftarrow 2$ to $m$ do        ▷ $c_i$ is inserted in the $i^{th}$ iteration
3:     Scan $\mathcal{Q}$ linearly from index $i-1$ to $1$ to find the index
       $j$ where $c_i$ should be inserted according to the preference
       of voter $\ell$ and insert $c_i$ in $\mathcal{Q}$ at $j$
4: end for
5: return $\mathcal{Q}$
```

For the sake of the analysis of our algorithm, let us to introduce a few terminologies. Given two preferences $\succ_1$ and $\succ_2$, we call a pair of candidates $(x, y) \in \mathcal{C} \times \mathcal{C}, x \neq y$, *good* if both $\succ_1$ and $\succ_2$ order them in a same way; a pair of candidates is called *bad* if it is not good. We divide the number of queries made by our algorithm into two parts: goodCost(·) and badCost(·) which are the number of queries made between good and respectively bad pair of candidates. In what follows, we show that goodCost(·) for Elicit(·) is small and the total badCost(·) across all the runs of Elicit(·) is small.

Lemma 2 *The goodCost(Elicit($\mathcal{C}, \mathcal{R}, \ell$)) of Elicit($\mathcal{C}, \mathcal{R}, \ell$) is $\mathcal{O}(m)$ (good is with respect to the preferences $\mathcal{R}$ and $\succ_\ell$).*

Proof: Follows immediately from the observation that in any iteration of the for loop at line 2 in Algorithm 1, only one good pair of candidates are compared. □

We now use Algorithm 1 iteratively to find the profile. We present the pseudocode in Algorithm 2 which works for the more general setting where a single crossing ordering is known but the voters are arriving in any arbitrary order π. We next compute the query complexity of Algorithm 2 when voters are arriving in a single crossing order.

Algorithm 2 PreferenceElicit(π)

```
Input: $\pi \in \mathbb{S}_n$
Output: Profile of all the voters
1: $\mathcal{Q}[\pi(1)] \leftarrow$ Elicit $\succ_{\pi(1)}$ using Observation 2  ▷ $\mathcal{Q}$ stores
   the profile
2: $\mathcal{X} \leftarrow \{\pi(1)\}$     ▷ Set of voters' whose preferences have
   already been elicited
3: for $i \leftarrow 2$ to $n$ do ▷ Elicit the preference of voter $\pi(i)$ in
   iteration $i$
4:     $k \leftarrow \min_{j \in \mathcal{X}} |j - i|$          ▷ Find the closest known
       preference
5:     $\mathcal{R} \leftarrow \mathcal{Q}[k], \mathcal{X} \leftarrow \mathcal{X} \cup \{\pi(i)\}$
6:     $\mathcal{Q}[\pi(i)] \leftarrow$ Elicit($\mathcal{C}, \mathcal{R}, \pi(i)$)
7: end for
8: return $\mathcal{Q}$
```

Theorem 2 *Assume that the voters are arriving sequentially according to an order with respect to which a profile $\mathcal{P}$ is single crossing. Then there is a* PREFERENCE ELICITATION *algorithm with query complexity $\mathcal{O}(mn + m^2)$.*

Proof: By renaming, let us assume, without loss of generality, that the voters are arriving according to the identity permutation id_n of the voters and the profile $\mathcal{P}$ is single crossing with respect to id_n. Let the profile $\mathcal{P}$ be $(P_1, P_2, \ldots, P_n) \in \mathcal{L}(\mathcal{C})^n$. For two candidates $x, y \in \mathcal{C}$ and a voter $i \in \{2, \ldots, n\}$, let us define a variable $b(x, y, i)$ to be one if x and y are compared for the voter i by Elicit($\mathcal{C}$,P_{i-1}, i) and (x, y) is a bad pair of candidates with respect to the preferences of voter i and $i-1$; otherwise $b(x, y, i)$ is defined to be zero. Then we have the following.

$$\begin{aligned}
& \text{CostPreferenceElicit}(id_n) \\
= \;& \mathcal{O}(m \log m) + \sum_{i=2}^{n} \Big(\text{goodCost}(\text{QUERY}(\mathcal{C}, P_{i-1}, i)) + \\
& \text{badCost}(\text{QUERY}(\mathcal{C}, P_{i-1}, i))\Big) \\
\leqslant \;& \mathcal{O}(m \log m + mn) + \sum_{i=2}^{n} \text{badCost}(\text{QUERY}(\mathcal{C}, P_{i-1}, i)) \\
= \;& \mathcal{O}(m \log m + mn) + \sum_{(x,y) \in \mathcal{C} \times \mathcal{C}} \left(\sum_{i=2}^{n} b(x, y, i) \right) \\
\leqslant \;& \mathcal{O}(m \log m + mn) + \sum_{(x,y) \in \mathcal{C} \times \mathcal{C}} 1 \\
= \;& \mathcal{O}(mn + m^2)
\end{aligned}$$

The first inequality follows from Lemma 2, the second equality follows from the definition of $b(x, y, i)$, and the second inequality follows from the fact that $\sum_{i=2}^{n} b(x, y, i) \leqslant 1$ for every pair of candidates $(x, y) \in \mathcal{C}$ since the profile $\mathcal{P}$ is single crossing. □

We show next that, when the voters are arriving in a single crossing order, the query complexity upper bound in Theorem 3 is tight for a large number of voters up to constant factors. The idea is to pair up the candidates in a certain way and argue that the algorithm must compare the candidates in every pair for every voter thereby proving a $\Omega(mn)$ lower bound on query complexity.

Theorem 3 *Assume that the voters are arriving sequentially according to an order with respect to which a profile $\mathcal{P}$ is single crossing. Then any* PREFERENCE ELICITATION *algorithm has query complexity $\Omega(m \log m + mn)$.*

Proof: The $\Omega(m \log m)$ bound follows from the fact that any profile consisting of only one preference $P \in \mathcal{L}(\mathcal{C})$ is single crossing. By renaming, let us assume without loss of generality that the profile $\mathcal{P}$ is single crossing with respect to the identity permutation of the voters. Suppose we have an even number of candidates and $\mathcal{C} = \{c_1, \dots, c_m\}$. Consider the order $\mathcal{Q} = c_1 \succ c_2 \succ \cdots \succ c_m$ and the pairing of the candidates $\{c_1, c_2\}, \{c_3, c_4\}, \dots, \{c_{m-1}, c_m\}$. The oracle answers all the query requests consistently according to the order Q till the first voter κ for which there exists at least one odd integer $i \in [m]$ such that the pair (c_i, c_{i+1}) is not queried. If there does not exist any such κ, then the algorithm makes at least $mn/2$ queries thereby proving the statement. Otherwise, let κ be the first vote such that the algorithm does not compare c_i and c_{i+1} for some odd integer $i \in [m]$. The oracle answers the queries for the rest of the voters $\{\kappa + 1, \dots, n\}$ according to the order $Q' = c_1 \succ c_2 \succ \cdots \succ c_{i-1} \succ c_{i+1} \succ c_i \succ c_{i+2} \succ \cdots \succ c_m$. If the algorithm orders $c_i \succ_\kappa c_{i+1}$ in the preference of the voter κ, then the oracle sets the preference of the voter κ to be $\mathcal{Q}'$. On the other hand, if the algorithm orders $c_{i+1} \succ_\kappa c_i$ in the preference of voter κ, then the oracle sets the preference of voter κ to be $\mathcal{Q}$. Clearly, the elicitation algorithm fails to correctly elicit the preference of the voter κ. However, the profiles for both the cases are single crossing with respect to the identity permutation of the voters and are consistent with the answers given to all the queries made by the algorithm. Hence, the algorithm must make at least $mn/2$ queries. □

We next move on to the case when we know a single crossing order of the voters; however, the voters arrive in an arbitrary order $\pi \in \mathbb{S}_n$. The idea is to call the function Elicit($\mathcal{C}, \mathcal{R}, i$) where the current voter is the voter i and $\mathcal{R}$ is the preference of the voter which is closest to i according to a single crossing ordering and whose preference has already been elicited by the algorithm.

Theorem 4 *Assume that a profile $\mathcal{P}$ is known to be single crossing with respect to a known ordering of voters $\sigma \in \mathbb{S}_n$. However, the voters are arriving sequentially according to an arbitrary order $\pi \in \mathbb{S}_n$ which may be different from σ. Then there is a* PREFERENCE ELICITATION *algorithm with query complexity $\mathcal{O}(mn + m^2 \log n)$.*

Proof: By renaming, let us assume, without loss of generality, that the profile $\mathcal{P}$ is single peaked with respect to the identity permutation of the voters. Let the profile $\mathcal{P}$ be $(P_1, P_2, \dots, P_n) \in \mathcal{L}(\mathcal{C})^n$. Let $f : [n] \longrightarrow [n]$ be the function such that $f(i)$ is the k corresponding to the i at line 4 in Algorithm 2. For candidates $x, y \in \mathcal{C}$ and voter ℓ, we define $b(x, y, \ell)$ analogously as in the proof of Theorem 2. We claim that $B(x, y) = \sum_{i=2}^{n} b(x, y, i) \leqslant \log n$. To see this, we consider any arbitrary pair $(x, y) \in \mathcal{C} \times \mathcal{C}$. Let the set of indices of the voters that have arrived immediately after the first time (x, y) contributes to $B(x, y)$ be $\{i_1, i_2, \dots, i_t\}$. Without loss of generality, let us assume $i_1 < i_2 < \cdots < i_t$. Again, without loss of generality, let us assume that voters $i_1, i_2, \dots, i_j$ prefer x over y and voters $i_{j+1}, \dots, i_t$ prefer y over x. Let us define Δ to be the difference between smallest index of the voter who prefers y over x and the largest index of the voter who prefers x over y. Hence, we currently have $\Delta = i_{j+1} - i_j$. A crucial observation is that if a new voter ℓ contributes to $B(x, y)$ then we must necessarily have $i_j < \ell < i_{j+1}$. Another crucial observation is that whenever a new voter contributes to $B(x, y)$, the value of Δ gets reduced at least by a factor of two by the choice of k at line 4 in Algorithm 2. Hence, the pair (x, y) can contribute at most $(1 + \log \Delta) = \mathcal{O}(\log n)$ to $B(x, y)$ since we have $\Delta \leqslant n$ to begin with. The rest of the proof is along the same line of the proof of Theorem 2. □

3.2 Results: Unknown Single Crossing Order

We now turn our attention to PREFERENCE ELICITATION for single crossing profiles when no single crossing ordering is known. Before we present our PREFERENCE ELICITATION algorithm for this setting, let us first prove a few structural results about single crossing profiles which we will use crucially later. We begin with showing an upper bound on the number of distinct preferences in any single crossing profile.

Lemma 3 *Let $\mathcal{P}$ be a profile on a set $\mathcal{C}$ of candidates which is single crossing. Then the number of distinct preferences in $\mathcal{P}$ is at most $\binom{m}{2} + 1$.*

Proof: By renaming, let us assume, without loss of generality, that the profile $\mathcal{P}$ is single crossing with respect to the identity permutation of the voters. We now observe that whenever the i^{th} vote is different from the $(i+1)^{th}$ vote for some $i \in [n-1]$, there must exist a pair of candidates $(x, y) \in \mathcal{C} \times \mathcal{C}$ whom the i^{th} vote and the $(i+1)^{th}$ vote order differently. Now the statement follows from the fact that, for every pair of candidates $(a, b) \in \mathcal{C} \times \mathcal{C}$, there can exist at most one $i \in [n-1]$ such that the i^{th} vote and the $(i+1)^{th}$ vote order a and b differently. □

We show next that in every single crossing profile $\mathcal{P}$ where all the preferences are *distinct*, there exists a pair of candidates $(x, y) \in \mathcal{C} \times \mathcal{C}$ such that nearly half of the voters in $\mathcal{P}$ prefer x over y and the other voters prefer y over x.

Lemma 4 *Let $\mathcal{P}$ be a profile of n voters such that all the preferences are distinct. Then there exists a pair of candidates $(x, y) \in \mathcal{C}$ such that x is preferred over y in at least $\lfloor n/2 \rfloor$ preferences and y is preferred over x in at least $\lfloor n/2 \rfloor$ preferences in $\mathcal{P}$.*

Proof: Without loss of generality, by renaming, let us assume that the profile $\mathcal{P}$ is single crossing with respect to the identity permutation of the voters. Since all the preferences in $\mathcal{P}$ are distinct, there exists a pair of candidates

$(x, y) \in \mathcal{C} \times \mathcal{C}$ such that the voter $\lfloor n/2 \rfloor$ and the voter $\lfloor n/2 \rfloor + 1$ order x and y differently. Let us assume, without loss of generality, that the voter $\lfloor n/2 \rfloor$ prefers x over y. Now, since the profile $\mathcal{P}$ is single crossing, every voter in $[\lfloor n/2 \rfloor]$ prefer x over y and every voter in $\{\lfloor n/2 \rfloor + 1, \dots, n\}$ prefer y over x. □

Using Lemma 3 and 4 we now design a PREFERENCE ELICITATION algorithm when no single crossing ordering of the voters is known. The overview of the algorithm is as follows. At any point of time in the elicitation process, we have the set $\mathcal{Q}$ of all the distinct preferences that we have already elicited completely and we have to elicit the preference of a voter ℓ. We first search the set of votes $\mathcal{Q}$ for a preference which is *possibly* same as the preference $\succ_\ell$ of the voter ℓ. It turns out that we can find a possible match $\succ \in \mathcal{Q}$ using $\mathcal{O}(\log |\mathcal{Q}|)$ queries due to Lemma 4 which is $\mathcal{O}(\log m)$ due to Lemma 3. We then check whether the preference of the voter ℓ is indeed the same as $\succ$ or not using $\mathcal{O}(m)$ queries. If $\succ$ is the same as $\succ_\ell$, then we have elicited $\succ_\ell$ using $\mathcal{O}(m)$ queries. Otherwise, we elicit $\succ_\ell$ using $\mathcal{O}(m \log m)$ queries using Observation 2. Fortunately, Lemma 3 tells us that we would use the algorithm in Observation 2 at most $\mathcal{O}(m^2)$ times. We present the pseudocode of our PREFERENCE ELICITATION algorithm in this setting in Algorithm 4.

Algorithm 3 Same($\mathcal{R}$, ℓ)

```
Input: $\mathcal{R} = c_1 \succ c_2 \succ \cdots \succ c_m \in \mathcal{L}(\mathcal{C}), \ell \in [n]$
Output: TRUE if the preference of the $\ell^{th}$ voter is $\mathcal{R}$ ; FALSE otherwise
1: for $i \leftarrow 1$ to $m - 1$ do
2:     if QUERY($c_i \succ_\ell c_{i+1}$) = FALSE then
3:         return FALSE                ▷ We have found a mismatch.
4:     end if
5: end for
6: return TRUE
```

Theorem 5 *Assume that a profile $\mathcal{P}$ is known to be single crossing. However, no ordering of the voters with respect to which $\mathcal{P}$ is single crossing is known. The voters are arriving sequentially according to an arbitrary order $\pi \in \mathbb{S}_n$. Then there is a* PREFERENCE ELICITATION *algorithm with query complexity $\mathcal{O}(mn + m^3 \log m)$.*

Proof: We present the pesudocode in Algorithm 4. We maintain two arrays in the algorithm. The array $\mathcal{R}$ is of length n and the j^{th} entry stores the preference of voter j. The other array $\mathcal{Q}$ stores all the votes seen so far after removing duplicate votes; more specifically, if some specific preference $\succ$ has been seen ℓ many times for any $\ell > 0$, $\mathcal{Q}$ stores only one copy of $\succ$. Upon arrival of voter i, we first check whether there is a preference in $\mathcal{Q}$ which is "potentially" same as the preference of voter i. At the beginning of the search, our search space $\mathcal{Q}' = \mathcal{Q}$ for a potential match in $\mathcal{Q}$ is of size $|\mathcal{Q}|$. We next iteratively keep halving the search space as follows. We find a pair of candidates $(x, y) \in \mathcal{C} \times \mathcal{C}$ such that at least $\lfloor |\mathcal{Q}'|/2 \rfloor$ preferences in $\mathcal{Q}'$ prefer x over y and at least $\lfloor |\mathcal{Q}'|/2 \rfloor$ preferences prefer y over x. The existence of such a pair of candidates is guaranteed by Lemma 4 and can be found in $\mathcal{O}(m^2)$ time by simply going over all possible pairs of candidates. By querying how voter i orders x and y, we reduce the search space $\mathcal{Q}'$

Algorithm 4 PreferenceElicitUnknownSCOrdering(π)

```
Input: $\pi \in \mathbb{S}_n$
Output: Profile of all the voters
1: $\mathcal{R}, \mathcal{Q} \leftarrow \emptyset$      ▷ $\mathcal{Q}$ stores all the votes seen so far without duplicate. $\mathcal{R}$ stores the profile.
2: for $i \leftarrow 1$ to $n$ do      ▷ Elicit preference of the $i^{th}$ voter in $i^{th}$ iteration of this for loop.
3:     $\mathcal{Q}' \leftarrow \mathcal{Q}$
4:     while $|\mathcal{Q}'| > 1$ do      ▷ Search $\mathcal{Q}$ to find a vote potentially same as the preference of $\pi(i)$
5:         Let $x, y \in \mathcal{C}$ be two candidates such that at least $\lfloor |\mathcal{Q}'|/2 \rfloor$ votes in $\mathcal{Q}'$ prefer $x$ over $y$ and at least $\lfloor |\mathcal{Q}'|/2 \rfloor$ votes in $\mathcal{Q}'$ prefer $y$ over $x$.
6:         if QUERY($x \succ_{\pi(i)} y$) = TRUE then
7:             $\mathcal{Q}' \leftarrow \{v \in \mathcal{Q}' : v$ prefers $x$ over $y\}$
8:         else
9:             $\mathcal{Q}' \leftarrow \{v \in \mathcal{Q}' : v$ prefers $y$ oer $x\}$
10:        end if
11:    end while
12:    Let $w$ be the only vote in $\mathcal{Q}'$  ▷ $w$ is potentially same as the preference of $\pi(i)$
13:    if Same($w, \pi(i)$) = TRUE then   ▷ Check whether the vote $\pi(i)$ is potentially same as $w$
14:        $\mathcal{R}[\pi(i)] \leftarrow w$
15:    else
16:        $\mathcal{R}[\pi(i)] \leftarrow$ Elicit using Observation 2
17:        $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{\mathcal{R}[\pi(i)]\}$
18:    end if
19: end for
20: return $\mathcal{R}$
```

for a potential match in $\mathcal{Q}$ to a set of size at most $\lfloor |\mathcal{Q}'|/2 \rfloor + 1$. Hence, in $\mathcal{O}(\log m)$ queries, the search space reduces to only one preference since we have $|\mathcal{Q}| \leqslant m^2$ by Lemma 3. Once we find a potential match w in $\mathcal{Q}$ (line 12 in Algorithm 4), we check whether the preference of voter i is the same as w or not using $\mathcal{O}(m)$ queries. If the preference of voter i is indeed same as w, then we output w as the preference of voter i. Otherwise, we use Observation 2 to elicit the preference of voter i using $\mathcal{O}(m \log m)$ queries and put the preference of voter i in $\mathcal{Q}$. Since the number of times we need to use the algorithm in Observation 2 is at most the number of distinct votes in $\mathcal{P}$ which is known to be at most m^2 by Lemma 3, we get the statement. □
Theorem 5 immediately gives us the following corollary in the random access to voters model when no single crossing ordering is known.

Corollary 1 *Assume that a profile $\mathcal{P}$ is known to be single crossing. However, no ordering of the voters with respect to which $\mathcal{P}$ is single crossing is known. Given a random access to voters, there is an* PREFERENCE ELICITATION *algorithm with query complexity $\mathcal{O}(mn + m^3 \log m)$.*

We now show that the query complexity upper bound of Corollary 1 is tight up to constant factors for large number of voters.

Theorem 6 *Given a random access to voters, any* PREFERENCE ELICITATION *algorithm which do not know any ordering of the voters with respect to which the input profile is single crossing has query complexity $\Omega(m \log m + mn)$.*

Acknowledgments. Palash Dey wishes to gratefully acknowledge support from Google India for providing him with a special fellowship for carrying out his doctoral work. Neeldhara Misra acknowledges support by the INSPIRE Faculty Scheme, DST India (project IFA12-ENG-31).

References

[Arrow, 1950] Kenneth J Arrow. A difficulty in the concept of social welfare. *The Journal of Political Economy*, pages 328–346, 1950.

[Ballester and Haeringer, 2011] Miguel A Ballester and Guillaume Haeringer. A characterization of the single-peaked domain. *Social Choice and Welfare*, 36(2):305–322, 2011.

[Bartholdi III *et al.*, 1989] John Bartholdi III, Craig A Tovey, and Michael A Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and welfare*, 6(2):157–165, 1989.

[Black *et al.*, 1958] Duncan Black, Robert Albert Newing, Iain McLean, Alistair McMillan, and Burt L Monroe. *The theory of committees and elections*. Springer, 1958.

[Black, 1948] Duncan Black. On the rationale of group decision-making. *The Journal of Political Economy*, pages 23–34, 1948.

[Brandt *et al.*, 2015] Felix Brandt, Markus Brill, Edith Hemaspaandra, and Lane A Hemaspaandra. Bypassing combinatorial protections: Polynomial-time algorithms for single-peaked electorates. *Journal of Artificial Intelligence Research (JAIR)*, pages 439–496, 2015.

[Conitzer and Sandholm, 2002] Vincent Conitzer and Tuomas Sandholm. Vote elicitation: Complexity and strategy-proofness. In *Eighteenth National Conference on Artificial Intelligence (AAAI)*, pages 392–397, 2002.

[Conitzer and Sandholm, 2005] Vincent Conitzer and Tuomas Sandholm. Communication complexity of common voting rules. In *Proceedings of the 6th ACM conference on Electronic Commerce (EC)*, pages 78–87. ACM, 2005.

[Conitzer, 2009] Vincent Conitzer. Eliciting single-peaked preferences using comparison queries. *J. Artif. Intell. Res. (JAIR)*, 35:161–191, 2009.

[Cormen, 2009] Thomas H Cormen. *Introduction to algorithms*. MIT press, 2009.

[Cornaz *et al.*, 2013] Denis Cornaz, Lucie Galand, and Olivier Spanjaard. Kemeny elections with bounded single-peaked or single-crossing width. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence (IJCAI)*, pages 76–82. AAAI Press, 2013.

[Dey and Misra, 2016a] Palash Dey and Neeldhara Misra. Elicitation for preferences single peaked on trees. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI)*. AAAI Press, 2016.

[Dey and Misra, 2016b] Palash Dey and Neeldhara Misra. Preference elicitation for single crossing domain. https://arxiv.org/abs/1604.05194, 2016.

[Ding and Lin, 2013] Ning Ding and Fangzhen Lin. Voting with partial information: what questions to ask? In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1237–1238. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

[Dodgson, 1876] Charles Lutwidge Dodgson. *A method of taking votes on more than two issues*. 1876.

[Faliszewski *et al.*, 2014] Piotr Faliszewski, Edith Hemaspaandra, and Lane A Hemaspaandra. The complexity of manipulative attacks in nearly single-peaked electorates. *Artificial Intelligence*, 207:69–99, 2014.

[Gibbard, 1973] Allan Gibbard. Manipulation of voting schemes: a general result. *Econometrica: Journal of the Econometric Society*, pages 587–601, 1973.

[Hemaspaandra *et al.*, 2005] Edith Hemaspaandra, Holger Spakowski, and Jörg Vogel. The complexity of kemeny elections. *Theoretical Computer Science*, 349(3):382–391, 2005.

[Hinich and Munger, 1997] Melvin J Hinich and Michael C Munger. *Analytical politics*. Cambridge University Press, 1997.

[Kemeny, 1959] John G Kemeny. Mathematics without numbers. *Daedalus*, 88(4):577–591, 1959.

[Levenglick, 1975] Arthur Levenglick. Fair and reasonable election systems. *Behavioral Science*, 20(1):34–46, 1975.

[Lu and Boutilier, 2011a] Tyler Lu and Craig Boutilier. Robust approximation and incremental elicitation in voting protocols. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*, pages 287–293, 2011.

[Lu and Boutilier, 2011b] Tyler Lu and Craig Boutilier. Vote elicitation with probabilistic preference models: Empirical estimation and cost tradeoffs. In *Algorithmic Decision Theory*, pages 135–149. 2011.

[Magiera and Faliszewski, 2014] Krzysztof Magiera and Piotr Faliszewski. How hard is control in single-crossing elections. *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI)*, 2014.

[Mirrlees, 1971] James A Mirrlees. An exploration in the theory of optimum income taxation. *The review of economic studies*, pages 175–208, 1971.

[Moulin, 1991] Hervi Moulin. *Axioms of cooperative decision making*. Number 15. Cambridge University Press, 1991.

[Pennock *et al.*, 2000] David M. Pennock, Eric Horvitz, and C. Lee Giles. Social choice theory and recommender systems: Analysis of the axiomatic foundations of collaborative filtering. In *Proceedings of the 17th International Conference on Artificial Intelligence (AAAI)*, 2000.

[Procaccia *et al.*, 2008] Ariel D Procaccia, Jeffrey S Rosenschein, and Aviv Zohar. On the complexity of achieving proportional representation. *Social Choice and Welfare*, 30(3):353–362, 2008.

[Roberts, 1977] Kevin WS Roberts. Voting over income tax schedules. *Journal of Public Economics*, 8(3):329–340, 1977.

[Saporiti and Tohmé, 2006] Alejandro Saporiti and Fernando Tohmé. Single-crossing, strategic voting and the median choice rule. *Social Choice and Welfare*, 26(2):363–383, 2006.

[Satterthwaite, 1975] Mark Allen Satterthwaite. Strategy-proofness and arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of Economic Theory*, 10(2):187–217, 1975.

[Skowron *et al.*, 2013] Piotr Skowron, Lan Yu, Piotr Faliszewski, and Edith Elkind. The complexity of fully proportional representation for single-crossing electorates. In *Symposium Algorithmic Game Theory (SAGT)*, pages 1–12. Springer, 2013.

[Young, 1977] H Peyton Young. Extending condorcet's rule. *Journal of Economic Theory*, 16(2):335–353, 1977.