The Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21)

Proportional Representation under Single-Crossing Preferences Revisited

Andrei Costin Constantinescu, Edith Elkind
University of Oxford
andrei.costin.constantinescu@gmail.com, elkind@cs.ox.ac.uk

Abstract

We study the complexity of determining a winning committee under the Chamberlin–Courant voting rule when voters' preferences are single-crossing on a line, or, more generally, on a tree. For the line, Skowron et al. (2015) describe an $O(n^2mk)$ algorithm (where n, m, k are the number of voters, the number of candidates and the committee size, respectively); we show that a simple tweak improves the time complexity to $O(nmk)$. We then improve this bound for $k = \Omega(\log n)$ by reducing our problem to the k-link path problem for DAGs with concave Monge weights, obtaining a $nm2^{O(\sqrt{\log k \log\log n})}$ algorithm for the general case and a nearly linear algorithm for the Borda misrepresentation function. For trees, we point out an issue with the algorithm proposed by Clearwater, Puppe, and Slinko (2015), and develop a $O(nmk)$ algorithm for this case as well.

1 Introduction

The problem of computing election winners under various voting rules is perhaps the most fundamental research challenge in computational social choice (Brandt et al. 2016). While this problem is typically easy for single-winner voting rules (with a few notable exceptions (Hemaspaandra, Hemaspaandra, and Rothe 1997; Rothe, Spakowski, and Vogel 2003)), for many rules that are supposed to return a *set* of winners, the winner determination problem is computationally demanding. In particular, this is the case for one of the most prominent and well-studied multiwinner voting rules, namely, the Chamberlin–Courant rule (Chamberlin and Courant 1983). Under this rule, each voter is assumed to assign a numerical disutility to each candidate; these disutilities are then lifted to sets of candidates, so that a voter's disutility for a set of candidates S is his minimum disutility for a member of S, and the goal is to identify a committee that minimizes the sum of voters' disutilities given an upper bound on the committee size (see Section 2 for formal definitions). It has been argued that this rule is well-suited for a variety of tasks, ranging from selecting a representative student assembly to deciding which movies to show on a plane (Faliszewski et al. 2017).

Decision problems related to winner determination under the Chamberlin–Courant rule have been shown to be NP-hard even when the disutility function takes a very simple form (Procaccia, Rosenschein, and Zohar 2008; Lu and Boutilier 2011). Accordingly, there is substantial body of work that focuses on identifying special classes of voters' preferences for which a winning committee can be determined efficiently. In particular, polynomial-time algorithms have been obtained for various structured preference domains, such as single-peaked preferences (Betzler, Slinko, and Uhlmann 2011), single-crossing preferences (Skowron et al. 2015), and preferences that are single-peaked on trees, as long as the underlying trees have few leaves or few internal vertices (Yu, Chan, and Elkind 2013; Peters and Elkind 2016) (see also (Peters et al. 2020)). These results extend to preferences that are nearly single-peaked or nearly single-crossing, for a suitable choice of distance measure (Cornaz, Galand, and Spanjaard 2012; Skowron et al. 2015; Misra, Sonar, and Vaidyanathan 2017); see also the survey by Elkind, Lackner, and Peters (2017) for a summary of results for restricted domains and the survey by Faliszewski et al. (2017) for a discussion of other approaches to circumventing hardness results for the Chamberlin–Courant rule.

Recently, Kung (2015) and, independently, Clearwater, Puppe, and Slinko (2015) introduced the domain of preferences that are singe-crossing on trees, which considerably extends the domain of single-crossing preferences, while sharing some if its desirable properties, such as existence of (weak) Condorcet winners. Clearwater, Puppe, and Slinko (2015) also proposed an algorithm for computing the Chamberlin–Courant winners when voters' preferences belong to this domain. Unfortunately, a close inspection of this algorithm shows that its running time scales approximately with the number of subtrees of the underlying tree, which may be exponential in the number of voters; we discuss this issue in Section 4.

Our Contribution In this paper, we revisit the problem of computing the winners under the Chamberlin–Courant rule when the voters' preferences are single-crossing, or, more broadly, single-crossing on a tree. For the former setting, we observe that a simple tweak of the algorithm of Skowron et al. (2015) improves the running time from $O(n^2mk)$ to $O(nmk)$. We then reduce the Chamberlin–Courant winner determination problem to the well-studied DAG k-LINK PATH problem, and show that the instances of the latter problem that are produced by our reduction have the *con-*

cave Monge property. This can be used to show that for $k = \Omega(\log n)$ our problem admits an algorithm that runs in time $nm2^{O(\sqrt{\log k \log\log n})}$; also, for the Borda disutility function (see Section 2), we obtain an algorithm that runs in time $O(nm\log(nm))$, i.e., nearly linear in the input size. This improvement is significant, as in some of the applications we discussed (such as movie selection) k can be quite large. For preferences single-crossing on trees, we design a polynomial-time algorithm that is based on dynamic programming. Interestingly, we can achieve a running time of $O(nmk)$ for this case as well.

Full version The full version of the paper is available on arXiv (Constantinescu and Elkind 2020).

2 Preliminaries

For a positive integer n, we write $[n]$ to denote the set $\{1,\dots,n\}$; given two non-negative integers n, n', we write $[n : n']$ to denote the set $\{n,\dots,n'\}$. Given a tree T, we write $|T|$ to denote the number of vertices of T.

We consider a setting with a set of voters V, where $|V| = n$, and a set of candidates $C = [m]$. Voters rank candidates from best to worst, so that the preferences of a voter v are given by a linear order $\succ_v$: given two distinct candidates $i, j \in C$ we write $i \succ_v j$ when v prefers i to j. We write $\mathcal{P} = (\succ_v)_{v\in V}$; the list $\mathcal{P}$ is referred to as a *preference profile*. We assume that we are also given a *misrepresentation function* $\rho : V \times C \to \mathbb{Q}$; we say that ρ is *consistent* with $\mathcal{P}$ if $c \succ_v c'$ implies $\rho(v,c) \le \rho(v,c')$ for each $v \in V$ and all $c, c' \in C$. Intuitively, the value $\rho(v,c)$ indicates to what extent candidate c misrepresents voter v. An example of a misrepresentation function is the *Borda misrepresentation function* ρ_B given by $\rho_B(v,c) = |\{c' \in C : c' \succ_v c\}|$: this function assigns value 0 to voter's top choice, value 1 to his second choice, and value $m-1$ to his last choice.

Multiwinner Rules A *multiwinner voting rule* maps a profile $\mathcal{P}$ over a candidate set C and a positive integer k, $k \le |C|$, to a non-empty collection of subsets of C of size at most k; the elements of this collection are called the *winning committees*[1]. In this paper, we focus on a family of multiwinner voting rules known as *Chamberlin–Courant* rules (Chamberlin and Courant 1983; Faliszewski et al. 2017).

An *assignment function* is a mapping $w : V \to C$; for each $V' \subseteq V$ we write $w(V') = \{w(v) : v \in V'\}$. If $|w(V)| \le k$, then w is called a *k-assignment function*. Given a misrepresentation function ρ and a profile $\mathcal{P} = (\succ_v)_{v\in V}$, the *total dissatisfaction of voters in V under a k-assignment* w is given by $\Phi_\rho(\mathcal{P}, w) = \sum_{v\in V} \rho(v, w(v))$. Intuitively, $w(v)$ is the representative of voter v in the committee $w(V)$, and $\Phi_\rho(\mathcal{P}, w)$ measures to what extent the voters are dissatisfied with their representatives. An *optimal k-assignment function* for ρ and $\mathcal{P}$ is a k-assignment function that minimizes $\Phi_\rho(\mathcal{P}, w)$ among all k-assignment functions for $\mathcal{P}$.

The *Chamberlin–Courant multiwinner voting rule* takes as input a preference profile $\mathcal{P} = (\succ_v)_{v\in V}$ over a candidate set C, a misrepresentation function $\rho : V \times C \to \mathbb{Q}$ that is consistent with $\mathcal{P}$ and a positive integer $k \le |C|$, and outputs all sets W such that $W = w(V)$ for some k-assignment function w that is optimal for ρ and $\mathcal{P}$. In the CC-WINNER problem the goal is to find some set W in the output of this rule. This problem is known to be NP-complete (Procaccia, Rosenschein, and Zohar 2008), even if ρ is the Borda misrepresentation function (Lu and Boutilier 2011). We assume that operations on values of $\rho(v,c)$ (such as, e.g., addition) can be performed in unit time; this assumption is realistic as the values of ρ are usually small integers.

We say that a k-assignment w for a profile $\mathcal{P}$ and a misrepresentation function ρ is *canonical* if w is optimal for $\mathcal{P}$ and ρ, and for each voter $v \in V$ the candidate $w(v)$ is v's most preferred candidate in $w(V)$. If $\rho(v,a) \ne \rho(v,b)$ for all $v \in V$ and all pairs of distinct candidates $(a,b) \in C \times C$, then every optimal assignment is canonical; however, if it may happen that $\rho(v,a) = \rho(v,b)$ for $a \ne b$, this need not be the case. An optimal k-assignment w can be transformed into a canonical assignment $\widehat{w}$ by setting $\widehat{w}(v)$ to be v's most preferred candidate in $w(V)$; note that this transformation weakly decreases misrepresentation and the committee size.

Single-Crossing Preferences A profile $\mathcal{P} = (\succ_v)_{v\in V}$ over C is *single-crossing (on a line)* if there is a linear order $\lhd$ on V such that for any triple of voters v_1, v_2, v_3 with $v_1 \lhd v_2 \lhd v_3$ and every pair of distinct candidates $(c, c') \in C \times C$ it is not the case that $c \succ_{v_1} c'$, $c' \succ_{v_2} c$, and $c \succ_{v_3} c'$. That is, if we order the voters in V according to $\lhd$ and traverse the list of voters from left to right, each pair of candidates 'crosses' at most once. A profile $\mathcal{P} = (\succ_v)_{v\in V}$ over C is *single-crossing on a tree* if there exists a tree T with vertex set V that has the following property: for any triple of voters v_1, v_2, v_3 such that v_2 lies on the path from v_1 to v_3 in T and every pair of distinct candidates $(c, c') \in C \times C$ it is not the case that $c \succ_{v_1} c'$, $c' \succ_{v_2} c$, and $c \succ_{v_3} c'$. Note that if a profile $\mathcal{P}$ is single-crossing on a tree T that is a path, then $\mathcal{P}$ is single-crossing on a line.

We say that an assignment function w for a profile $\mathcal{P}$ over C that is single-crossing on a tree T is *connected* if for every candidate $c \in C$ it holds that the inverse image $w^{-1}(c)$ defines a subtree of T. The following lemma shows that, when considering profiles single-crossing on trees, we can focus on connected assignment functions.

Lemma 1. *For every profile $\mathcal{P}$ over C that is single-crossing on a tree T and every $k \le |C|$ every canonical k-assignment for $\mathcal{P}$ is connected.*

Proof. Let w be a canonical k-assignment for $\mathcal{P}$. To see that w is connected, fix a candidate $c \in C$ and let T' be the smallest subtree of T that contains the set $w^{-1}(c)$. If w is not connected, then there is a voter v in T' such that $w(v) = c'$, $c' \ne c$, and deleting v would disconnect T'. Then there are two voters $x, y \in T' \cap w^{-1}(c)$ for which the unique simple x–y path contains v. Since w is a canonical assignment, we have $c \succ_x c', c \succ_y c'$, but $c' \succ_v c$, a contradiction with $\mathcal{P}$ being single-crossing on T. □

The next lemma establishes a monotonicity property of canonical assignments that will be useful for our analysis.

[1] Note that we allow committees of size less than k, as this simplifies the presentation.

Lemma 2. *Consider a profile $\mathcal{P} = (\succ_v)_{v \in V}$ that is single-crossing on a tree T, and suppose that voter v ranks the candidates as $1 \succ_v \ldots \succ_v m$. Then, every canonical k-assignment for $\mathcal{P}$ is non-decreasing along every simple path in T starting at v.*

Proof. Consider a canonical k-assignment w. Let P be a simple path starting at voter v and let x, y be two voters on P such that x precedes y on P. Suppose $w(x) = a$, $w(y) = b$ with $a > b$. Then $b \succ_v a$, $a \succ_x b$ and $b \succ_y a$, a contradiction with $\mathcal{P}$ being single-crossing on T. □

We will be interested in solving CC-WINNER if voters' preferences are single-crossing on a line or on a tree; we denote these special cases of our problem by CC-WINNER-SC and CC-WINNER-SCT, respectively. We assume that the respective ordering/tree is given as part of the input; this assumption is without loss of generality as such an ordering/tree can be computed from the input profile in polynomial time (Doignon and Falmagne 1994; Kung 2015; Clearwater, Puppe, and Slinko 2015).

Rooted Trees and DAGs A *rooted tree* is a finite tree with a designated root vertex r. We say that a vertex u is a *parent* of v (and v is a *child* of u) if u and v are adjacent and u lies on the path from v to r. A vertex with no children is called a *leaf*. We denote the number of children of vertex v by n_v, and represent the set of children of v as an array $ch[v, 1], \ldots, ch[v, n_v]$. We write T_v to denote the subtree of T with vertex set $\{u :$ the path from u to r goes through $v\}$. Similarly, for each $v \in V$ and $i \in [1 : n_v + 1]$, let $T_{v,i}$ be the subtree obtained by starting with T_v and removing the subtrees $T_{ch[v,1]}, \ldots, T_{ch[v,i-1]}$. Observe that $T_{v,1} = T_v$ and that T_{v,n_v+1} is just the singleton vertex v.

A *directed acyclic graph* (DAG) is a directed graph whose vertices can be totally ordered so that the tail of each arc precedes its head in the ordering. All DAGs we consider have the set $[0 : n]$ as their set of vertices (ordered according to the natural ordering $<$), and are complete, i.e., contain an edge (i, j) for each pair $i, j \in [0 : n]$ with $i < j$. A DAG is *weighted* if its arcs are given real values by a function ω. A weighted DAG on vertex set $[0 : n]$ has the *concave Monge* property if for all vertices i, j such that $0 < i+1 < j < n$ it holds that $\omega(i, j)+\omega(i+1, j+1) \leq \omega(i, j+1)+\omega(i+1, j)$. We refer to the weight function ω itself as being *concave Monge* in this case.

3 Improved Algorithms for Single-Crossing Preferences

We start by considering the setting where the voters' preferences are single-crossing on a line. We assume without loss of generality that the voter order $\lhd$ is given by $v_1 \lhd \ldots \lhd v_n$ and that the first voter ranks the candidates in $C = [m]$ as $1 \succ_{v_1} \ldots \succ_{v_1} m$. The following lemma is implicit in the work of Skowron et al. (2015), and can be seen as an instantiation of Lemmas 1 and 2.

Lemma 3. *For every canonical assignment w_{opt} for* CC-WINNER-SC *and every pair of voters v_i, v_j with $i < j$ it holds that $w_{opt}(v_i) \leq w_{opt}(v_j)$.*

Skowron et al. (2015) use this lemma to develop a dynamic programming algorithm for CC-WINNER-SC that runs in time $O(n^2mk)$. We will now present a faster dynamic programming algorithm that uses auxiliary variables.

Theorem 1. *Given an instance of* CC-WINNER-SC *with n voters, m candidates and committee size k, we can compute an optimal solution in time $O(nmk)$.*

Proof. We will explain how to compute the minimum dissatisfaction; a winning committee can then be computed using standard dynamic programming techniques.

We define the following subproblems for each $i \in [n]$, $c \in [m]$ and each $\ell = 1, \ldots, \min\{k, m - c + 1, n - i + 1\}$:

- let $dp_0[i, \ell, c]$ be the minimum dissatisfaction of voters in $\{v_i, \ldots, v_n\}$ for a size-ℓ committee that is contained in $[c : m]$;
- let $dp_1[i, \ell, c]$ be the minimum dissatisfaction of voters in $\{v_i, \ldots, v_n\}$ for a size-ℓ committee that is contained in $[c : m]$ and represents v_i by c.

To simplify presentation, we assume $dp_f[-, -, -] = \infty$ for $f \in \{0, 1\}$ and i, c, ℓ not satisfying the conditions.

We have $dp_1[n, 1, c] = \rho(v_n, c)$ for each $c \in C$. Also, $dp_0[n, 1, m] = \rho(v_n, m)$, and for $c < m$ we have $dp_0[n, 1, c] = \min\{dp_1[n, 1, c], dp_0[n, 1, c + 1]\}$.
For $i = n - 1, \ldots, 1$ we have the following recurrence:

$$dp_1[i, \ell, c] = \rho(v_i, c) + \min\{dp_1[i + 1, \ell, c], dp_0[i + 1, \ell - 1, c + 1]\};$$
$$dp_0[i, \ell, c] = \min\{dp_1[i, \ell, c], dp_0[i, \ell, c + 1]\}.$$

This recurrence enables us to compute all values $dp_f[-, -, -]$ for $f \in \{0, 1\}$; the minimum dissatisfaction in our instance is given by $\min_{1 \leq \ell \leq k} dp_0[1, \ell, 1]$. The dynamic program has $O(nmk)$ entries; each entry can be computed in time $O(1)$ given the already-computed entries. □

To improve over the $O(nmk)$ bound, we will reduce CC-WINNER-SC to the well-studied DAG k-LINK PATH problem with Monge concave weights (see, e.g., Bein, Larmore, and Park (1992)), and use the powerful machinery developed for it to obtain faster algorithms for our setting.

Definition 1. *Given a DAG with an arc weight function ω and two designated vertices s and t, the* k-LINK PATH *problem (k-LPP) asks for a minimum total weight path starting at s, ending at t and consisting of exactly k arcs.*

There are a number of algorithmic results for the k-LINK PATH problem assuming a concave Monge weight function (Bein, Larmore, and Park 1992; Aggarwal, Schieber, and Tokuyama 1994; Schieber 1995). We will first present our reduction, and then discuss the implications for CC-WINNER-SC.

Given an instance of CC-WINNER-SC with a preference profile $\mathcal{P} = (\succ_v)_{v \in V}$ over $C = [m]$, we construct a DAG G with vertex set $[0 : n]$ and the weight function ω given by

$$\omega(i, j) = \min_{c \in C} (\rho(v_{i+1}, c) + \ldots + \rho(v_j, c)). \quad (1)$$

That is, $\omega(i,j)$ represents the minimum total dissatisfaction that voters in $\{v_{i+1},\dots,v_j\}$ derive from being represented by a single candidate c. Let $cand(i,j)$ be some candidate in $\arg\min_{c\in C}(\rho(v_{i+1},c)+\ldots+\rho(v_j,c))$.

First, we observe that an optimal solution to CC-WINNER-SC corresponds to a minimum cost path in k-LPP.

Theorem 2. *The minimum cost of a k-link 0–n path in G with respect to ω is equal to the minimum total dissatisfaction for $\mathcal{P}$ and k.*

Proof. Let $P = 0 \to \ell_1 \to \ldots \to \ell_{k-1} \to n$ be a minimum cost k-link path in G. Then P induces an assignment of candidates to voters: if P contains an arc (i,j) we assign candidate $cand(i,j)$ to voters $v_{i+1},\dots,v_j$. The total dissatisfaction under this assignment equals to the weight of P.

Conversely, let w_{opt} be a canonical k-assignment. By Lemma 3 we know that w_{opt} partitions the voters into contiguous subsequences. To build a path P in G, we proceed as follows. For every maximal contiguous subsequence of voters $v_{i+1},\dots,v_j$ represented by the same candidate in w_{opt}, we add the arc $i \to j$ to P. By construction, the resulting set of arcs forms a k-link path from 0 to n, and its total weight is at most $\Phi_\rho(\mathcal{P}, w_{opt})$. □

Note, however, that Theorem 2 is insufficient for our purposes: the efficient algorithms for k-LPP require the weight function ω to have the concave Monge property, so we need to prove that the reduction in Theorem 2 always produces such instances of k-LPP.

We say that an instance of CC-WINNER-SC is *concave Monge* if the reduction in Theorem 2 maps it to an instance of k-LPP with the concave Monge property. Thus, we need to prove that each instance of CC-WINNER-SC is concave Monge. To this end, we will first argue that if there is an instance of CC-WINNER-SC that is not concave Monge, then there exists such an instance with three voters. Then we prove that every three-voter instance is concave Monge.

Lemma 4. *If there exists a non-concave Monge instance of* CC-WINNER-SC*, then there exists a non-concave Monge instance of* CC-WINNER-SC *with three voters.*

Proof. Consider a non-concave Monge instance of CC-WINNER-SC with $n \neq 3$ voters. Note that $n \geq 4$: indeed, for $n < 3$ there is no pair of vertices i,j that satisfies $0 < i+1 < j < n$. We can assume that the (i,j) pair that violates the concave Monge property is $(0,n-1)$: otherwise we could just remove all voters before v_{i+1} and all voters after v_j. Thus, we have $\omega(0,n-1)+\omega(1,n) > \omega(0,n)+\omega(1,n-1)$. Recall that

$$\omega(0,n-1) = \min_{c\in C}(\rho(v_1,c)+\ldots+\rho(v_{n-1},c)); \quad (2)$$

$$\omega(1,n) = \min_{c\in C}(\rho(v_2,c)+\ldots+\rho(v_n,c)); \quad (3)$$

$$\omega(0,n) = \min_{c\in C}(\rho(v_1,c)+\ldots+\rho(v_n,c)); \quad (4)$$

$$\omega(1,n-1) = \min_{c\in C}(\rho(v_2,c)+\ldots+\rho(v_{n-1},c)). \quad (5)$$

Now, consider a three-voter profile $(\succ_x,\succ_y,\succ_z)$ with misrepresentation function ρ' constructed as follows. Set $\succ_x = \succ_{v_1}$, $\succ_z = \succ_{v_n}$ and $\rho'(x,c) = \rho(v_1,c)$, $\rho'(z,c) = \rho(v_n,c)$ for all $c \in C$. Further, set $\rho'(y,c) = \rho(v_2,c) + \rho(v_3,c) + \ldots + \rho(v_{n-1},c)$ and let $a \succ_y b$ if and only if $\rho'(y,a) < \rho'(y,b)$ or $\rho'(y,a) = \rho'(y,b)$ and $a \succ_{v_1} b$. One can verify that $\succ_y$ is a linear order. Moreover, we claim that the profile $(\succ_x,\succ_y,\succ_z)$ is single-crossing with respect to the voter order $x \lhd y \lhd z$. Indeed, consider two distinct candidates a,b. If x and z disagree on (a,b), then in $(\succ_x,\succ_y,\succ_z)$ candidates a and b cross at most once, irrespective of y's preferences. Now suppose that x and z agree on (a,b); for concreteness, suppose that $a \succ_x b$, $a \succ_z b$. As the input profile is single-crossing, all other voters also prefer a to b and hence $\rho(v_2,a)+\rho(v_3,a)+\ldots+\rho(v_{n-1},a) \leq \rho(v_2,b)+\rho(v_3,b)+\ldots+\rho(v_{n-1},b)$, in which case $a \succ_y b$. Hence, $(\succ_x,\succ_y,\succ_z)$ is indeed single-crossing.

Now, we can rewrite equations (2)–(5) as

$$\omega(0,n-1) = \min_{c\in C}(\rho'(x,c)+\rho'(y,c));$$
$$\omega(1,n) = \min_{c\in C}(\rho'(y,c)+\rho'(z,c));$$
$$\omega(0,n) = \min_{c\in C}(\rho'(x,c)+\rho'(y,c)+\rho'(z,c));$$
$$\omega(1,n-1) = \min_{c\in C}\rho'(y,c).$$

This shows that the instance of CC-WINNER-SC formed by x, y and z together with ρ' is also non-concave Monge. □

Proposition 1. *Every instance of* CC-WINNER-SC *is concave Monge.*

Proof. By Lemma 4, it suffices to consider instances with three voters. Thus, consider a three-voter profile that is single-crossing with respect to the voter order $v_1 \lhd v_2 \lhd v_3$. We need to argue that

$$\omega(0,2)+\omega(1,3) \leq \omega(0,3)+\omega(1,2).$$

Let $a = cand(1,2)$; we can assume that a is the top candidate of the second voter. Also, let $b = cand(0,3)$, $c_1 = cand(0,2)$, $c_2 = cand(1,3)$.

Suppose first that $b = a$. Then

$$\omega(0,3)+\omega(1,2) = \rho(v_1,a)+2\rho(v_2,a)+\rho(v_3,a).$$

On the other hand, $c_1 = cand(0,2)$ implies that

$$\omega(0,2) = \rho(v_1,c_1)+\rho(v_2,c_1) \leq \rho(v_1,a)+\rho(v_2,a),$$

and $c_2 = cand(1,3)$ implies that

$$\omega(1,3) = \rho(v_2,c_2)+\rho(v_3,c_2) \leq \rho(v_2,a)+\rho(v_3,a).$$

Adding up these inequalities, we obtain the desired result.

Now, suppose that $b \neq a$. Then

$$\omega(0,3)+\omega(1,2) = \rho(v_1,b)+\rho(v_2,b)+\rho(v_2,a)+\rho(v_3,b).$$

As the second voter ranks a first, she ranks b below a. Hence, by the single-crossing property, at least one other voter prefers a to b; we can assume without loss of generality that $a \succ_{v_3} b$. Thus, $\rho(v_3,b) \geq \rho(v_3,a)$ and hence

$$\omega(0,3)+\omega(1,2) \geq \rho(v_1,b)+\rho(v_2,b)+\rho(v_2,a)+\rho(v_3,a).$$

Now, $c_1 = cand(0, 2)$ implies that

$$\omega(0, 2) = \rho(v_1, c_1) + \rho(v_2, c_1) \leq \rho(v_1, b) + \rho(v_2, b),$$

and $c_2 = cand(1, 3)$ implies that

$$\omega(1, 3) = \rho(v_2, c_2) + \rho(v_3, c_2) \leq \rho(v_2, a) + \rho(v_3, a).$$

Again, adding up these inequalities, we obtain the desired result. □

It now follows that any fast algorithm for k-LPP with the concave Monge property translates into an algorithm for CC-WINNER-SC, slowed down by a factor of $O(m)$ required for computing arc weights[2].

Now, if individual dissatisfactions are non-negative integers in range $[0 : U]$ (e.g. $U = m$ for the Borda misrepresentation function), then the weakly-polynomial algorithm of Bein, Larmore, and Park (1992) and Aggarwal, Schieber, and Tokuyama (1994) leads to an $O(nm \log(nU))$ algorithm for CC-WINNER-SC. Alternatively, we can use the strongly-polynomial time algorithm of Schieber (1995) to get a runtime of $nm2^{O(\sqrt{\log k \log\log n})}$, which improves on our earlier bound of $O(nmk)$ for $k = \omega(\log n)$. We summarize these results in the following theorem.

Theorem 3. *Given an instance of* CC-WINNER-SC *with n voters, m candidates and committee size k, where $k = \Omega(\log n)$, we can compute an optimal solution in time $nm2^{O(\sqrt{\log k \log\log n})}$. Moreover, if ρ is the Borda misrepresentation function, we can compute an optimal solution in time $O(nm \log(nm))$.*

4 Preferences Single-Crossing on a Tree

Clearwater, Puppe, and Slinko [2015] present an algorithm for CC-WINNER-SCT that proceeds by dynamic programming, building a solution for the entire tree from solutions for various subtrees. Entries of their dynamic program are indexed by subtrees of the input tree, and on some instances the algorithm may need to consider all subtrees containing the root; a tree on n vertices can have $2^{\Omega(n)}$ such subtrees. We present a detailed example in the full version.[3]

4.1 A Dynamic Programming Solution

We will now present a different dynamic programming algorithm, which provably runs in polynomial time. This algorithm, too, builds a solution iteratively by considering subtrees of the original tree, but it proceeds in such a way that it only needs to consider polynomially many subtrees.

Fix a misrepresentation function ρ, and consider a profile $\mathcal{P} = (\succ_v)_{v \in V}$, $|V| = n$, over $C = [m]$ that is single-crossing on a tree T. We will view T as a rooted tree with v_1 as its root, and assume without loss of generality that $1 \succ_{v_1} \ldots \succ_{v_1} m$.

We first reformulate our problem as a tree partition problem using Lemma 1.

Definition 2. *A* p-partition *of T is a partition of T into p subtrees $\mathcal{F} = \{F_1, \ldots, F_p\}$. A p-assignment $w : V \to C$ for a profile $\mathcal{P} = (\succ_v)_{v \in V}$ that is single-crossing on a tree T is a* p-tree assignment *if there is a p-partition $\{F_1, \ldots, F_p\}$ of T such that $w(v) = w(v')$ for each $\ell \in [p]$ and $v, v' \in F_\ell$. In the* CHAMBERLIN–COURANT TREE PARTITION *(CCTP) problem the goal is to find a value $p \in [k]$ and a p-tree assignment w_{opt}, together with the associated p-partition $\mathcal{F}_{opt}$, such that w_{opt} minimizes $\Phi_\rho(\mathcal{P}, w)$ over all $p \in [k]$ and all p-tree assignments for $\mathcal{P}$.*

By Lemma 1, an optimal assignment for CCTP is an optimal assignment for the associated instance of CC-WINNER-SCT.

We start by presenting a dynamic programming algorithm for CCTP and proving a bound of $O(nmk^2)$ on its running time; later, we will improve this bound to $O(nmk)$.

Theorem 4. *Given an instance of CCTP with n voters, m candidates and committee size k, we can compute an optimal solution in time $O(nmk^2)$.*

Proof. We will explain how to find the value of an optimal solution in time $O(nmk^2)$; an optimal solution can then be recovered using standard dynamic programming techniques.

We define the following subproblems for each $v \in V$ and $c \in [m]$.

- For each $\ell = 1, \ldots, \min\{k, |T_v|\}$, let $dp_0[v, \ell, c]$ be the minimal dissatisfaction of voters in T_v that can be achieved by partitioning T_v into ℓ subtrees using only candidates in $[c : m]$ as representatives.
- For each $\ell = 1, \ldots, \min\{k, |T_v|\}$, let $dp_1[v, \ell, c]$ be the minimal dissatisfaction of voters in T_v that can be achieved by partitioning T_v into ℓ subtrees using only candidates in $[c : m]$ as representatives, with voter v represented by candidate c.
- For each $i \in [n_v + 1]$, and each $\ell = 1, \ldots, \min\{k, |T_{v,i}|\}$, let $dp_2[v, i, \ell, c]$ be the minimal dissatisfaction of voters in $T_{v,i}$ that can be achieved by partitioning $T_{v,i}$ into ℓ subtrees using only candidates in $[c : m]$ as representatives, with voter v represented by candidate c.

To simplify presentation, we assume these quantities to take value ∞ for v, i, ℓ, c not satisfying the conditions.

Clearly, the value of an optimal solution to our instance of CCTP is $\min_{\ell \in [k]} dp_0[v_1, \ell, 1]$. It remains to explain how to compute the intermediate quantities.

The following observations are immediate from the definitions of dp_0, dp_1, dp_2:

$$dp_2[v, n_v + 1, 1, c] = \rho(v, c), \tag{6}$$

$$dp_1[v, \ell, c] = dp_2[v, 1, \ell, c], \tag{7}$$

$$dp_0[v, \ell, c] = \min\{dp_1[v, \ell, c], dp_0[v, \ell, c + 1]\}. \tag{8}$$

The next lemma explains how to compute dp_2.

[2] Computing all arc weights in advance would be too expensive. Instead, we precompute the values $\sum_{\ell=1}^{j} \rho(v_\ell, c)$ for all $j \in [n]$, $c \in C$ in time $O(nm)$; then, when the algorithm needs to know $\omega(i, j)$, we compute $\sum_{\ell=i+1}^{j} \rho(v_\ell, c)$ for each c in time $O(1)$ as a difference of two precomputed quantities, and then compute the minimum over C in time $O(m)$.

[3] We have contacted the authors of the paper, and they have acknowledged this issue.

Lemma 5. *Let u be the i-th child of v, and let $s = |T_{v,i+1}|$. Then $dp_2[v, i, \ell, c] = \min\{DIFF, SAME\}$, where*

$$DIFF = \min\{dp_0[u, t, c+1] + dp_2[v, i+1, \ell - t, c] : 1 \le t \le \min\{\ell, |T_u|\},\ 1 \le \ell - t \le \min\{\ell, s\}\},$$
$$SAME = \min\{dp_1[u, t, c] + dp_2[v, i+1, \ell - t + 1, c] : 1 \le t \le \min\{\ell, |T_u|\},\ 1 \le \ell - t + 1 \le \min\{\ell, s\}\}.$$

Proof. By Lemma 2 we can assume that candidates with higher indices are placed further down in the tree. Let $(\mathcal{F}_{opt}, w_{opt})$ be an optimal connected ℓ-tree partition of $T_{v,i}$ where voter v is assigned candidate c. There are two cases:

- Voter u is represented by a candidate $c' > c$. Then each subtree in $\mathcal{F}_{opt}$ is either fully contained in T_u or fully contained in $T_{v,i+1}$, so there exists $t \in [\ell]$ such that $\mathcal{F}_{opt}$ partitions T_u into t subtrees and $T_{v,i+1}$ into $\ell - t$ subtrees. Hence, to minimize dissatisfaction, we take the minimum over all $t \in [\ell]$. For a fixed $t \in [\ell]$ we choose (i) an optimal t-partition of T_u that uses candidates in $[c+1 : m]$ only and (ii) an optimal $(\ell - t)$-partition of $T_{v,i+1}$ that uses candidates in $[c : m]$ only and assigns c to v. The optimal values of the former and the latter are given by $dp_0[u, t, c+1]$ and $dp_2[v, i+1, \ell - t, c]$, respectively.
- Voter u is represented by candidate c. The analysis is similar to the previous case, except that the subtree in $\mathcal{F}_{opt}$ that contains v may partly reside in both T_u and $T_{v,i+1}$, so there exists $t \in [\ell]$ such that $\mathcal{F}_{opt}$ partitions T_u into t subtrees and $T_{v,i+1}$ into $\ell - t + 1$ subtrees.

□

Note that the assignment implicitly computed by our dynamic programming algorithm is not necessarily connected; however, this is not required for optimality.

Our dynamic program proceeds from the leaves to the root of T, computing the quantities dp_0, dp_1 and dp_2; we process a vertex after its children have been processed. Computing all these quantities is trivial if v is a leaf; if v is not a leaf, we first compute $dp_2[v, i, \ell, c]$ for all $i = |n_v| + 1, \ldots, 1$ and all relevant values of ℓ and c using (6) and Lemma 5, and then compute $dp_1[v, \ell, c]$ (using (7)) and $dp_0[v, \ell, c]$ (using (8)) for $c = m, \ldots, 1$.

To bound the running time, note that (1) there are $O(nmk)$ subproblems of the form $dp_0[-,-,-]$ and $dp_1[-,-,-]$, each requiring constant time to solve; (2) there are $O(nmk)$ subproblems of the form $dp_2[-,-,-,-]$. This is because pairs of the form $(v,\ i)$ such that $1 \le i \le n_v$ correspond to edges of the tree and there are precisely $n - 1$ of them. Each of these subproblems can be solved in time $O(k)$ by Lemma 5; (3) The tree can be traversed in time $O(n)$.

Altogether, we get a time bound of $O(nmk^2)$. □

4.2 Tighter Analysis of the Running Time

We will now show how to improve the bound on the running time of our algorithm to $O(nmk)$. To do so, it suffices to establish that all subproblems of the form $dp_2[-,-,-,-]$ can be solved in time $O(nmk)$.

The following technical lemma is an important building block in our analysis (see the full version for the proof).

Lemma 6. *Consider a voter v and a candidate c, and let u be the i-th child of v for some $i \in [n_v]$. Then all subproblems of the form $dp_2[v, i, -, c]$ can be solved in time $O(\min\{k, |T_u|) \cdot \min\{k, |T_{v,i+1}|\})$.*

Before proving the stronger $O(nmk)$ bound, we first show an easier bound of $O(n^2m)$, which is tight when $k = n$ and better than $O(nmk^2)$ whenever $k = \omega(n^{1/2})$. Proving this is both informative in itself, helping to build intuition, and will also provide us with a tool useful for the general argument. The $O(n^2m)$ bound is immediate from the following lemma (inspired by Cygan (2012)).

Lemma 7. *For each candidate $c \in C$ solving all subproblems of the form $dp_2[-,-,-,c]$ using Lemma 5 takes time $O(n^2)$.*

Proof. By Lemma 6, the time required to solve all subproblems of the form $dp_2[-,-,-,c]$ is asymptotically bounded by

$$\sum_{v \in V, 1 \le i \le n_v} \left(|T_{ch[v,i]}| \cdot |T_{v,i+1}|\right).$$

The quantity $|T_{ch[v,i]}| \cdot |T_{v,i+1}|$ can be interpreted as the number of pairs of vertices (v_1, v_2) such that $v_1 \in T_{ch[v,i]}$ and $v_2 \in T_{v,i+1}$. Note that for all such pairs, the lowest common ancestor of v_1 and v_2 is v. Thus, if we sum this quantity over all $i \in [n_v]$, we get precisely the number of unordered pairs of distinct vertices whose lowest common ancestor is v. It is now immediate that, if we further sum this quantity over all $v \in V$, we get precisely $\binom{n}{2}$, which is the number of unordered pairs of distinct nodes in a tree with n vertices, completing the proof. □

By summing up the $O(n^2)$ terms from Lemma 7 over all $c \in C$, and observing that CCTP becomes trivial if $k \ge n$ (we can then afford to include the top choice of each voter), we obtain the following bound.

Theorem 5. *Solving all subproblems of the form $dp_2[-,-,-,-]$ using Lemma 5 takes time $O(n^2m)$. Hence, CCTP can be solved in time $O(n^2m)$.*

We are now ready to prove the $O(nmk)$ time bound. Just as in Lemma 7, it suffices to bound the time required to solve all subproblems of the form $dp_2[-,-,-,c]$ for each $c \in C$.

Lemma 8. *For each candidate $c \in C$ solving all subproblems of the form $dp_2[-,-,-,c]$ takes time $O(nk)$.*

Proof. Let us revisit the expression for the time S needed to solve all subproblems of this form:

$$S = \sum_{v \in V, 1 \le i \le n_v} (\min\{k, |T_{ch[v,i]}|\} \cdot \min\{k, |T_{v,i+1}|\}). \tag{9}$$

Note that the pairs (v, i) in the summation index correspond to the edges of the tree. This observation suggests a new way of computing S based on incrementally building the tree starting from n singleton vertices. Namely, we start with $S = 0$ and an empty graph G consisting of n disconnected

singleton vertices, and repeat the next two steps until G becomes isomorphic to T:

1. Pick an edge $\{v, v'\}$ of T that has not been chosen before (where v' is a child of v in T) and connect v and v' in G. This edge corresponds to a pair (v, i) such that v' is the i-th child of v. We call this operation a (v, i)-*join*. A (v, i)-join can only take place if all (v, i')-joins with $i' > i$ have already been performed and the connected component of v' in G is isomorphic to $T_{v'}$.
2. Increase S by $\min\{k, |T_{v'}|\} \cdot \min\{k, |T_{v,i+1}|\}$.

We observe that at each step of this procedure the graph G is a forest, and each component tree of G is of the form $T_{v,i}$ for some $v \in V$ and $1 \leq i \leq n_v + 1$. Moreover, valid orders of joining the connected components of G correspond to valid orders of solving all the subproblems of the form $dp_2[-, -, -, c]$, and the final value of S (given in equation (9)) does not depend on the order selected. In particular, for the purposes of our analysis it will be convenient to split the process into two phases: in the first phase, we will only join two connected components if each of them has at most k vertices, and in the second phase we will perform the remaining joins. Accordingly, let S_1 and S_2 denote the amounts added to S in the first and the second phase, respectively, so that $S = S_1 + S_2$. We will now argue that $S_1 = O(nk)$ and $S_2 = O(nk)$.

Claim 1. $S_1 = O(nk)$.

Proof. At the end of the first phase, the graph G is a forest consisting of p trees $T_{u_1,i_1}, T_{u_2,i_2}, \ldots, T_{u_p,i_p}$. This state of G corresponds to having solved all subproblems of the form $dp_2[v, i, -, c]$ on which $dp_2[u_1, i_1, -, c], \ldots, dp_2[u_p, i_p, -, c]$ depend (possibly indirectly, and including the problems themselves), and no others. This is the same as performing the complete algorithm restricted to each of $T_{u_1,i_1}, T_{u_2,i_2}, \ldots, T_{u_p,i_p}$ individually. Thus, we can bound S_1 by applying Lemma 7 to each connected component and summing up the results:

$$S_1 \leq |T_{u_1,i_1}|^2 + |T_{u_2,i_2}|^2 + \ldots + |T_{u_p,i_p}|^2.$$

Since each such connected component has been generated by joining two connected components of size at most k, we can bound their individual sizes by $2k$. It follows that

$$S_1 \leq 2k \cdot (|T_{u_1,i_1}| + |T_{u_2,i_2}| + \ldots + |T_{u_p,i_p}|) \leq 2nk.$$

□

Claim 2. $S_2 = O(nk)$.

Proof. Given a sequence of p integers $(a_1, \ldots, a_p)$, let

$$\lambda(a_1, \ldots, a_p) = \min\{k, a_1\} + \min\{k, a_2\} + \ldots + \min\{k, a_p\}.$$

Suppose that at the start of the second phase the graph G has s components, and let $(b_1, \ldots, b_s)$ be the list of sizes of these components. Note that $b_1 + \cdots + b_s = n$ and hence $\lambda(b_1, \ldots, b_s) \leq n$. Further, suppose that at some point during the second phase the list of sizes of the components of G is given by $(f_1, \ldots, f_q)$, and consider a join operation merging together two connected components of sizes f_i and f_j. At least one of these components has size greater than k; without loss of generality, assume that $f_i > k$. This operation removes f_i and f_j from the list $(f_1, \ldots, f_q)$. This changes the value of λ by removing a $\min\{k, f_i\} + \min\{k, f_j\} = k + \min\{k, f_j\}$ term and adding back a $\min\{k, f_i + f_j\} = k$ term, thus decreasing λ by $\min\{k, f_j\}$. On the other hand, this operation increases S_2 by $\min\{k, f_i\} \cdot \min\{k, f_j\} = k \cdot \min\{k, f_j\}$. Therefore, whenever λ decreases by Δ, S_2 increases by $k\Delta$. Since λ can only ever decrease, starts off bounded from above by n and never becomes negative, S_2 is bounded from above by nk, completing the proof. □

Now Lemma 8 follows by combining Claims 1 and 2. □

As argued earlier in the paper, Lemma 8 immediately implies the desired bound on the performance of our algorithm.

Theorem 6. CC-WINNER-SCT *can be solved in time* $O(nmk)$.

5 Conclusions and Future Work

We have significantly improved the state of the art concerning the algorithmic complexity of the Chamberlin–Courant rule, both for preferences single-crossing on a line and for preferences single-crossing on a tree. For the former setting, the performance of our algorithms makes them suitable for a broad range of practical applications; for the latter setting, we identify an issue in prior work and present the first polytime algorithm. It is instructive to contrast the algorithmic results for preferences single-crossing on trees and preferences single-peaked on trees: for the latter domain, positive results hold only if the underlying tree has a special structure, and the problem remains hard for general trees (Peters et al. 2020), whereas our positive result holds for all trees.

In our paper, we focused on the utilitarian version of the Chamberlin–Courant rule, where the goal is to minimize the sum of voters' dissatisfactions; however, both of our $O(nmk)$ algorithms can be modified to compute winners under the *egalitarian* version of this rule, where the goal is to minimize the dissatisfaction of the most misrepresented voter, simply by replacing '+' with $\max$ in the respective dynamic programs. This is no longer the case for our reduction to the k-LPP problem; however, by using binary search to reduce the egalitarian problem to the utilitarian problem, we can nevertheless find solutions for the former in time $O(nm \log(n) \log(nm))$ using this approach.

Now, the concept of single-crossing preferences can be extended beyond trees to the much broader class of *median graphs*, which includes, e.g., grid graphs (Puppe and Slinko 2019). It would be very interesting to extend our algorithmic results to median graphs, and in particular to grids. While an analogue of the connectivity lemma (Lemma 1) holds for grids, we need a stronger geometric condition on the structure of optimal partitions for a dynamic programming approach to work (see the full version). We have empirically verified that this condition holds for small instances; proving this for the general case (and thus designing a polynomial-time algorithm for CC-WINNER on grids) remains a challenge for future work.

Acknowledgements

We thank Arkadii Slinko and Clemens Puppe for answering our questions about the algorithm for CC-WINNER-SCT proposed by Clearwater, Puppe, and Slinko (2015). We also thank the anonymous AAAI-2021 reviewers for their useful suggestions. This work was supported by an ERC Starting Grant ACCORD under Grant Agreement 639945 (Elkind).

References

Aggarwal, A.; Schieber, B.; and Tokuyama, T. 1994. Finding a minimum-weight k-link path in graphs with the concave Monge property and applications. *Discrete and Computational Geometry* 12: 263–280.

Bein, W.; Larmore, L.; and Park, J. 1992. The d-edge shortest-path problem for a Monge graph. *UNT Digital Library* .

Betzler, N.; Slinko, A.; and Uhlmann, J. 2011. On the Computation of Fully Proportional Representation. *Journal of Artificial Intelligence Research* 47.

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

Chamberlin, J.; and Courant, P. 1983. Representative Deliberations and Representative Decisions: Proportional Representation and the Borda Rule. *American Political Science Review* 77(3): 718–733.

Clearwater, A.; Puppe, C.; and Slinko, A. 2015. Generalizing the Single-Crossing Property on Lines and Trees to Intermediate Preferences on Median Graphs. In *IJCAI'15*, 32–38.

Constantinescu, A.; and Elkind, E. 2020. Proportional Representation under Single-Crossing Preferences Revisited. *arXiv preprint* arXiv:2010.08637.

Cornaz, D.; Galand, L.; and Spanjaard, O. 2012. Bounded single-peaked width and proportional representation. In *ECAI'12*, 270–275.

Cygan, M. 2012. Barricades. In Diks, K.; Idziaszek, T.; Lacki, J.; and Radoszewski, J., eds., *Looking for a Challenge?*, 63–67.

Doignon, J.-P.; and Falmagne, J.-C. 1994. A polynomial time algorithm for unidimensional unfolding representations. *Journal of Algorithms* 16(2): 218–233.

Elkind, E.; Lackner, M.; and Peters, D. 2017. Structured Preferences. In Endriss, U., ed., *Trends in Computational Social Choice*, chapter 10, 187–207. AI Access.

Faliszewski, P.; Skowron, P.; Slinko, A.; and Talmon, N. 2017. Multiwinner Voting: A New Challenge for Social Choice Theory. In Endriss, U., ed., *Trends in Computational Social Choice*, chapter 2, 27–47. AI Access.

Hemaspaandra, E.; Hemaspaandra, L. A.; and Rothe, J. 1997. Exact analysis of Dodgson elections: Lewis Carroll's 1876 voting system is complete for parallel access to NP. *Journal of the ACM* 44(6): 806–825.

Kung, F.-C. 2015. Sorting out single-crossing preferences on networks. *Social Choice and Welfare* 44(3): 663–672.

Lu, T.; and Boutilier, C. 2011. Budgeted Social Choice: From Consensus to Personalized Decision Making. In *Proceedings of IJCAI'11*, 280–286.

Misra, N.; Sonar, C.; and Vaidyanathan, P. 2017. On the complexity of Chamberlin-Courant on almost structured profiles. In *ADT'17*, 124–138. Springer.

Peters, D.; and Elkind, E. 2016. Preferences Single-Peaked on Nice Trees. In *AAAI'16*, 594–600.

Peters, D.; Yu, L.; Chan, H.; and Elkind, E. 2020. Preferences Single-Peaked on a Tree: Multiwinner Elections and Structural Results. *CoRR* abs/2007.06549.

Procaccia, A.; Rosenschein, J.; and Zohar, A. 2008. On the complexity of achieving proportional representation. *Social Choice and Welfare* 30: 353–362.

Puppe, C.; and Slinko, A. 2019. Condorcet domains, median graphs and the single-crossing property. *Economic Theory* 67(1): 285–318.

Rothe, J.; Spakowski, H.; and Vogel, J. 2003. Exact complexity of the winner problem for Young elections. *Theory of Computing Systems* 36(4): 375–386.

Schieber, B. 1995. Computing a minimum-weight k-link path in graphs with the concave Monge property. *J. Algorithms* 29: 204–222.

Skowron, P.; Yu, L.; Faliszewski, P.; and Elkind, E. 2015. The complexity of fully proportional representation for single-crossing electorates. *Theoretical Computer Science* 569: 43–57.

Yu, L.; Chan, H.; and Elkind, E. 2013. Multiwinner Elections Under Preferences That Are Single-Peaked on a Tree. In *IJCAI'13*, 425–431.