Social Choice and Welfare manuscript No.
(will be inserted by the editor)

On the Complexity of Achieving Proportional Representation

Ariel D. Procaccia, Jeffrey S. Rosenschein, Aviv Zohar

School of Engineering and Computer Science, The Hebrew University of Jerusalem
Jerusalem 91904, Israel (e-mail: {arielpro,jeff,avivz}@cs.huji.ac.il)

Received: date / Revised version: date

Abstract We demonstrate that winner selection in two prominent proportional representation voting systems is a computationally intractable (namely, $\mathcal{NP}$-complete) problem — implying that these systems are impractical when the assembly is large. On a different note, in settings where the size of the assembly is constant, we show that the problem can be solved in polynomial time.

1 Introduction

Andrew Carnegie, the American industrialist, famously remarked: "The first man gets the oyster, the second man gets the shell." Nevertheless, in many settings there may be multiple winners; the second, third and fourth men (or women) may also savor the oyster. When elections are held for a legislature, for instance, there are often hundreds of winners. Although many single-winner voting rules can be generalized to elect multiple winners [4], when multiple winners are involved complications arise — which are not necessarily shared by single-winner scenarios.

One of the basic properties one expects from a multi-winner voting rule is *proportional representation* (PR). Intuitively, the requirement is that the proportional support enjoyed by different factions be accurately reflected in the election's results. In practice, this usually means that the percentage of votes secured by a party is roughly proportional to the number of seats it is awarded.

Monroe [9] proposes an intriguing multi-winner voting scheme which strives towards proportional representation. In what Monroe refers to as the "pure" version of his scheme, the elected set of candidates minimizes voters'

Correspondence to: arielpro@cs.huji.ac.il

sum of "misrepresentation" values. More concrete versions of the system specify the source of these values. Monroe's scheme attempts to realize the following principle: if there are k winners, voters should be partitioned into k equally-sized groups, and each group assigned to a candidate, in a way that minimizes dissatisfaction of voters from their associated candidates. Chamberlin and Courant [5] preceded Monroe with a similar voting scheme. The main difference is that they eschew the abovementioned principle, but attempt to achieve a comparable outcome using weighted voting.

Unfortunately, otherwise appealing theoretical voting schemes may have a fatal flaw: the problem of determining who the winners are may be computationally intractable. A problem is considered tractable (in this context) if its solution can be calculated using a number of computational steps which is polynomial in the size of the input; problems which may require exponential time are considered intractable. Indeed, for all practical purposes, it would take an absurdly long time to solve a problem which demands a number of steps which is exponential in the size of the input — when the input is large. Computational complexity theory [11] classifies problems into complexity classes, such as $\mathcal{NP}$ (see Section 2 for details). The class of $\mathcal{NP}$-complete problems contains problems which, intuitively, are as hard as any other problem in $\mathcal{NP}$. Such problems are almost unanimously believed to be intractable.

Bartholdi et al. [3] establish computational intractability as a major consideration against the adoption of a voting system. Specifically, they show that in the voting schemes devised by Charles Dodgson (a.k.a. Lewis Carroll, author of "Alice's Adventures in Wonderland") and Kemeny, it is $\mathcal{NP}$-hard to pinpoint the winners. The importance of such results is not at all diminished by the fact that the complexity analysis involved is worst-case. There is no arguing that a situation where the outcome of an election takes centuries to compute must be avoided at all costs — even if such occasions are relatively rare. This stands in stark contrast to work on the computational complexity of manipulating elections [1,2,6,7]; even if a manipulation problem is $\mathcal{NP}$-hard, it may still be the case that manipulation is often possible. $\mathcal{NP}$-hardness is thus not a strong enough guarantee of resistance to manipulation, although it can be a mighty objection against a voting system.

Potthoff and Brams [12] show that in Monroe's and the Chamberlin-Courant voting schemes, the winners can be determined using integer programming. Although this provides a method to safely and accurately compute results of elections when the numbers involved are small, it is by no means an efficient method, as finding the solution of a general integer program is an $\mathcal{NP}$-complete problem [10]. Potthoff and Brams acknowledge this obstacle, and formulate an improved integer program for settings where the electorate is large. Nevertheless, the modified integer program is still of gargantuan size when the number of candidates is large.

In this paper we prove that in general, selecting the winners in an election is an $\mathcal{NP}$-complete problem in both the Monroe scheme and the

Chamberlin-Courant scheme. This result holds even if the misrepresentation values are based on simple approval ballots.

We also consider a scenario where the number of winners is small, although the electorate and the number of candidates are large. Using a new algorithm, we show that in this setting winner selection can be accomplished efficiently.

The paper proceeds as follows. In Section 2 we briefly discuss computational complexity analysis. In Section 3 we formally define the voting schemes devised by Monroe, and by Chamberlin and Courant. In Section 4, we prove that the winner selection problem is hard when the number of winners is large, and in Section 5, we demonstrate that the problem can be efficiently solved when the number of winners is small. Finally, our conclusions appear in Section 6.

2 Computational Complexity

Computational complexity is a branch of the theory of computation, which deals with the resources required to solve a problem (in our context the resource is *time*, but there are other scarce resources, such as memory). Established in the second half of the 20th century, it has since become a standard tool in many disciplines, including operations research and economics. In this section, we briefly discuss a tiny fragment of the theory. Readers are urged to consult [13] for an especially readable overview; a more detailed presentation can be found in [11].

The *time complexity* of a problem is the worst-case number of computational steps required to solve the problem, as a function of the size of the input, using the best algorithm. The problems considered are often *decision problems*: questions to which the answer is either "yes" or "no". A decision problem is formally described as a set: instances which are contained in the set are "yes" instances, while those that are not are "no" instances.

Decision problems are classified into complexity classes; the two best-known are $\mathcal{P}$ and $\mathcal{NP}$. $\mathcal{P}$ is the class of all decision problems whose time complexity is polynomial in the size of the input. In order to show that a problem is in $\mathcal{P}$, one has to devise an algorithm that solves the problem in polynomial time for any input; such an algorithm is considered formally *efficient*.

$\mathcal{NP}$ is the class of decision problems whose positive solutions can be verified in polynomial time, given the right information. In other words, for any "yes" instance of an $\mathcal{NP}$ problem, there exists a *witness* which makes it possible to verify in polynomial time that the given instance is indeed a "yes" instance. Consider the following problem, which will be used throughout the paper.

Definition 1 *In the* Max k-Cover *problem, we are given a set $\mathcal{U}$ of n points, a collection of subsets $\mathcal{F} = \{F_1, \ldots, F_m\}$, and integers $k, t \in \mathbb{N}$. We*

are asked whether it is possible to find k subsets in $\mathcal{F}$ such that their union covers at least t points in $\mathcal{U}$.

An instance is a specific choice of $\langle \mathcal{U}, \mathcal{F}, k, t \rangle$. If the given instance is a "yes" instance, a witness is a choice of k subsets from $\mathcal{F}$. One can easily check in polynomial time whether the k subsets indeed cover at least t points.

Some of the problems in $\mathcal{NP}$ are known as $\mathcal{NP}$-complete problems. Informally, these are problems which are as hard as any problem in $\mathcal{NP}$: if one $\mathcal{NP}$-complete problem can be solved efficiently, then any problem in $\mathcal{NP}$ can be solved efficiently. Known algorithms that solve $\mathcal{NP}$-complete problems precisely might require a number of computational steps which is exponential in the size of the input. Unfortunately, exponential functions tend to grow at an alarming rate. For example, consider an algorithm that executes 2^n computational steps, where n is the size of the input; when $n = 400$, this algorithm might require more than 10^{100} operations, which is significantly more than the number of particles in the known universe. Even with computers that are a thousand-fold faster than today's best supercomputers, such a computation would run longer than the the current age of the universe. Furthermore, this is a trifle compared to the time required to solve the problem when the size of the input is 500...

To prove that a problem is $\mathcal{NP}$-complete, one must show that any problem in $\mathcal{NP}$ can be reformulated as this problem. Formally:

Definition 2 *Let A and B be two sets, which are associated with decision problems. A function g from instances of A to instances of B is a* polynomial reduction *if and only if g can always be calculated in polynomial time and for all instances x of the problem A:*

$$x \in A \Leftrightarrow g(x) \in B.$$

If there exists such a function, we say that A reduces *to B, and write $A \leq_p B$.*

Given that one knows how to reduce problem A into problem B, one can solve A by translating it to B, and then solving B.

Definition 3 *A problem B is $\mathcal{NP}$-complete if and only if:*

1. *$B \in \mathcal{NP}$.*
2. *$\mathcal{NP}$-*hardness*: For all $A \in \mathcal{NP}$, $A \leq_p B$.*

As literally thousands of problems are currently known to be $\mathcal{NP}$-complete, the most straightforward way to show that a problem is $\mathcal{NP}$-complete is to provide a reduction from some problem already known to be $\mathcal{NP}$-complete; this is sufficient, since a composition of reductions is also a reduction.

The problem Max k-Cover is intuitively difficult: there is no obvious way to avoid checking many choices of k subsets from $\mathcal{F}$ — and there are an exponential number of possibilities. In this case, the intuition does not fail us; the following result is known [8]:

Lemma 1 MAX k-COVER *is* $\mathcal{NP}$*-complete.*

3 Proportional Representation Systems

Let the set of voters be V, and denote $|V| = n$; let the set of candidates be C, $|C| = m$. We use i to index the voters, and j to index the candidates. Furthermore, assume that k candidates are to be elected.

We begin by specifying Monroe's pure scheme [9]. For each voter i and candidate j, a *misrepresentation value* μ_{ij} is known; this value characterizes the degree to which candidate j misrepresents voter i.

Let $\mathcal{S} = \{S \subseteq C : |S| = k\}$. Let $S \in \mathcal{S}$, and let $f_S : V \to S$ be a function which assigns voters to candidates in S. The misrepresentation score of voter i under f_S is $\mu_{if_S(i)}$. The total misrepresentation of assignment f_S is $\sum_{i \in V} \mu_{if_S(i)}$.

Monroe requires that f_S be restricted so that a similar number of voters is assigned to each candidate in S. In other words, each candidate in S must be assigned at least $\lfloor n/k \rfloor$ voters; we say that such an assignment is *balanced*. The misrepresentation score of S is the misrepresentation score of f_S, where $f_S : V \to S$ is the assignment with the minimal misrepresentation, subject to the above restriction. The k winners are the set $S \in \mathcal{S}$ with the lowest misrepresentation score.

Chamberlin and Courant [5] adopt a similar approach; as before, one considers sets $S \in \mathcal{S}$ and assignments f_S. However, Chamberlin and Courant impose no restrictions on the assignments. Therefore, each set S is associated with the assignment $f_S : V \to S$ which minimizes misrepresentation among all possible assignments. In order to maintain proportionality, Chamberlin and Courant compensate by using weighted voting in the assembly: if S is the winning set of candidates, and the function f_S is the one which minimizes misrepresentation, a candidate $j \in S$ is given weight $|f_S^{-1}(j)|$.

The misrepresentation values μ_{ij} may be naturally derived from ballots cast by the electorate. Assume voters cast ordinal ballots, i.e., each voter ranks all candidates, and denote by r_{ij} the rank of candidate j on the ballot of voter i. In this setting, it is reasonable to define $\mu_{ij} = r_{ij} - 1$. Another possibility, which Monroe recommends, is approval balloting: voters either approve or disapprove candidates, and may approve as many candidates as they wish. The misrepresentation value μ_{ij} is 0 if i approves j, and 1 otherwise.

Remark 1 In any case, it is logical to assume that $\mu_{ij} \in \{0, 1, \ldots, m\}$, and we make this assumption throughout the paper. This point is very important, since the results in Section 5 depend on this assumption.

4 Winner Selection is Intractable When the Assembly is Large

Before proceeding with a rigorous computational complexity analysis, we must formulate the decision problem that we face.

Definition 4 *In the* Winner-Selection *problem, we are given the set of voters V, the set of candidates C, integers $k, t \in \mathbb{N}$, and integral values $\mu_{ij} \in \{0, 1, \ldots, m\}$. We are asked whether there exists a subset $S \subseteq C$ such that $|S| = k$, with misrepresentation at most t.*

Keeping in mind the discussion in Section 3, this formulation is valid with respect to both approaches mentioned: in Chamberlin and Courant's scheme the misrepresentation of a subset is the minimum over all assignments, while in Monroe's scheme it is the minimum over all balanced assignments.

Remark 2 Selecting winners is clearly harder than the above decision problem: once winners are selected, one can easily determine whether their misrepresentation is at most t. Since the set of winners minimizes misrepresentation, this procedure correctly decides the problem.

Remark 3 For ease of exposition, we shall assume that n/k is an integer. This does not limit the generality of our results, as otherwise it is possible to pad the electorate with voters i such that $\mu_{ij} = 0$ for all $j \in C$.

We shall presently prove that the Winner-Selection problem is computationally intractable. We strengthen the result by limiting voters' ballots to approval — misrepresentation values are either 0 or 1.

Theorem 1 *The* Winner-Selection *problem in the Chamberlin-Courant scheme is $\mathcal{NP}$-complete, even with approval ballots.*

Proof We must first show that the problem is in $\mathcal{NP}$. Given a "yes" instance $\langle V, C, \{\mu_{ij}\}, k, t\rangle$, it can be verified in polynomial time when a specific choice S of k candidates is produced as a witness. One simply verifies that the misrepresentation of S is at most t. The function f_S which minimizes misrepresentation can be trivially obtained by assigning each voter i to $\operatorname{argmin}_{j \in S} \mu_{ij}$.

We must show that any problem in $\mathcal{NP}$ can be reduced to Winner-Selection in the Chamberlin-Courant scheme with approval ballots. It is sufficient to exhibit a polynomial reduction from some $\mathcal{NP}$-complete problem; we use Max k-Cover. We begin by showing how to transform an instance of the the latter problem to an instance of the former; this transformation is the reduction function.

We are given an instance $\langle \mathcal{U}, \mathcal{F}, k, t\rangle$ of Max k-Cover. We associate each point $i \in \mathcal{U}$ with a voter $i \in V$, and associate each set $F_j \in \mathcal{F}$ with a candidate $j \in C$; $\mu_{ij} = 0$ (candidate j is approved by voter i) if in the given instance of Max k-Cover it holds that $i \in F_j$; otherwise $\mu_{ij} = 1$. This should make the association between sets in $\mathcal{F}$ and candidates apparent: a candidate j is affiliated with the set of all voters i such that $\mu_{ij} = 0$. We ask whether there is $S \in \mathcal{S}$ with a misrepresentation score of at most $n - t$. Notice that this reduction between problems can be carried out in polynomial time.

It remains to show that an instance of MAX k-COVER is a "yes" instance if and only if the transformed instance is a "yes" instance of our problem. Assume an instance of the former problem is a "yes" instance. Hence, there is a k-cover $\{F_{j_1}, \ldots, F_{j_k}\}$ which covers at least t points in $\mathcal{U}$. Observe the set of candidates $S = \{j_1, \ldots, j_k\}$; each candidate j_l is approved by all the voters associated with points in F_{j_l}. These voters can be assigned to the candidate j_l; it follows that their misrepresentation score is 0. There are at least t voters that approve one of the candidates in S, as the union of the sets F_{j_l} covers at least t points in $\mathcal{U}$. Note that a nonempty intersection of sets F_{j_l} is equivalent to voters approving several candidates in S; such voters are simply assigned to one of the candidates they approve, so their misrepresentation score is still 0. Therefore, there are at most $n - t$ voters with misrepresentation 1, so the total misrepresentation of S cannot exceed $n - t$.

In the other direction, suppose that the transformed instance is a "yes" instance of WINNER-SELECTION in the Chamberlin-Courant scheme. There must exist a set of candidates $S = \{j_1, \ldots, j_k\}$ with misrepresentation at most $n - t$. By similar reasoning as before, at least t voters $\{i_1, \ldots, i_t\}$ approve the candidates in S. Thus it holds that $\{i_1, \ldots, i_t\} \subseteq \bigcup_{l=1}^{k} F_{j_l}$; this entails that $|\bigcup_{l=1}^{k} F_{j_l}| \geq t$. □

WINNER-SELECTION in the Monroe scheme is intuitively harder than in the Chamberlin-Courant Scheme, since it involves an extra stage of balancing assignments, instead of assigning each voter to a favorite candidate. Indeed, the hardness of the Monroe scheme follows almost directly from the last theorem.

Theorem 2 *The* WINNER-SELECTION *problem in Monroe's scheme is $\mathcal{NP}$-complete, even with approval ballots.*

Proof In order to show that WINNER-SELECTION in Monroe's scheme is in $\mathcal{NP}$, we note that given a specific set S and an assignment f_S, it can be verified in polynomial time that f_S is balanced and that the misrepresentation of S with respect to f_S is at most t.

Next we prove that WINNER-SELECTION in Monroe's scheme is at least as hard as solving the problem in the Chamberlin-Courant scheme. Given an instance $\langle V, C, \{\mu_{ij}\}, k, t\rangle$ of the latter, we transform it by building an instance $\langle V', C, \{\mu_{ij}\}, k, t\rangle$ of the former which is identical in all respects, with the exception of the set of voters V'. The transformed electorate contains kn voters: the n original voters V with the given misrepresentation values μ_{ij}, and another $(k-1)n$ voters i with $\mu_{ij} = 0$ for all candidates j. Clearly the abovementioned transformation constitutes a polynomial time reduction.

We claim that for all $S \subseteq C$ such that $|S| = k$, S's misrepresentation with respect to the given Chamberlin-Courant version is equal to its misrepresentation in the transformed Monroe version. Indeed, the misrepresentation in the Chamberlin-Courant instance is at most that of the Monroe instance:

there are less voters, and the functions f_S are unconstrained. In the other direction, for any function f_S in the Chamberlin-Courant instance, we can construct a function f'_S in the Monroe instance with equal misrepresentation. For all $i \in V$, $f_S = f'_S$. The other $(k-1)n$ voters in V' are assigned to candidates in S arbitrarily, under the restriction that each candidate is assigned exactly n voters. This is possible as any candidate was assigned at most n voters by f_S. The assignment f'_S is balanced, and its misrepresentation is clearly equal to that of f_S, since for all $i \in V' \setminus V$ and all candidates j, $\mu_{ij} = 0$.

It follows directly that there is a set of k candidates with misrepresentation at most t in the Chamberlin-Courant instance if and only if there is such a set in the Monroe instance. □

5 Winner Selection is Tractable When the Assembly is Small

Our hardness results in the previous section relied on the implicit assumption that the number of winners k grows with the number of voters and candidates. In this section, we explore the scenario where the number of winners is constant — this is an additional property of the problem which is implicitly or explicitly assumed throughout this section. In fact, we prove:

Theorem 3 *When k is constant,* Winner-Selection *is in $\mathcal{P}$, both in the Chamberlin-Courant scheme and in Monroe's scheme.*

It is quite straightforward that Winner-Selection in the Chamberlin-Courant scheme can be solved efficiently. In this scheme, the misrepresentation value for each $S \in \mathcal{S}$ can be easily calculated by assigning each voter i to $\text{argmin}_{j \in S} \mu_{ij}$. Moreover, it holds that

$$|\mathcal{S}| = \binom{m}{k} = \frac{m!}{k!(m-k)!}.$$

This is a polynomial in m when k is constant. For example, if $k = 3$,

$$|\mathcal{S}| = \binom{m}{3} = \frac{m(m-1)(m-2)}{6} \leq m^3,$$

and in general $|\mathcal{S}| \leq m^k$. To conclude the point, one can compute the misrepresentation for every set in $\mathcal{S}$ using a number of operations which is polynomial in n and m — assuming k is constant.

Solving Winner-Selection efficiently in Monroe's scheme is far trickier. Naturally, it still holds that $\mathcal{S}$ is polynomial in m. An algorithm that efficiently computes the misrepresentation score of every $S \in \mathcal{S}$ entails, by similar reasoning as before, an efficient method to select winners: one simply computes the misrepresentation for all subsets in $\mathcal{S}$ and minimizes.

Unfortunately, computing the misrepresentation for a given $S \in \mathcal{S}$ in Monroe's scheme is not straightforward. It can be accomplished using integer programming, but the solution suggested by [12] might be exponential in

the number of candidates. Monroe [9] himself mentions a simple algorithm: at first, each voter i is assigned to $\text{argmin}_{j \in S} \mu_{ij}$. Then, the assignment is balanced by shifting voters that "will suffer the least increase in misrepresentation from the shift." Monroe adds that "determining the proper order of shifting is more tedious" when $k > 2$, "but the process is still straightforward."

Although this algorithm is indeed suitable when $k = 2$, a simple interpretation is not even guaranteed to terminate when $k \geq 3$. For example, let $S = \{1, 2, 3\}$, and let there be 3 voters. The misrepresentation values are: $\mu_{i1} = 1$, $\mu_{i2} = \mu_{i3} = 0$ for all i. Suppose that the initial assignment assigns two voters to candidate 2 and one voter to candidate 3. One of the voters assigned to candidate 2 must be shifted, and the total misrepresentation remains the same if a voter is shifted to candidate 3. Now candidate 3 is assigned 2 voters instead of one, and a voter is shifted back to candidate 2. This sequence of events is repeated infinitely.

We propose an efficient but somewhat more complicated algorithm. The algorithm maintains at each stage a balanced assignment, and iteratively decreases misrepresentation. Changes in the assignment are introduced by *cyclically right-shifting* sets of voters: each voter in a set $A = \{i_1, i_2, \ldots, i_l\}$ is shifted to the candidate which is assigned to his successor; the assignment remains balanced as the last voter is assigned to the first candidate. In more detail, if the current assignment is f_S, the algorithm singles out a set of voters $A = \{i_1, i_2, \ldots, i_l\}$, $l \leq k$, and modifies the assignment by defining the next assignment f'_S as follows:

$$f'_S(i) = \begin{cases} f_S(i_{d+1}) & i = i_d \in A \text{ and } d < l \\ f_S(i_1) & i = i_l \in A \\ f_S(i) & i \notin A \end{cases}$$

The algorithm can be described as follows:

Input: n voters V, m candidates C, a subset S of k candidates, misrepresentation values μ_{ij}.

Output: An assignment f_S of voters to candidates in S such that n/k voters are assigned to each candidate, and the total misrepresentation is minimized.

Initialization: Arbitrarily assign n/k voters to each candidate in S.

Iterative Step: Determine whether there is $A \subseteq V$, $|A| = l \leq k$, such that cyclically right-shifting the voters in A strictly decreases the total misrepresentation. If so, perform the shift. Otherwise, terminate and return the current assignment.

Lemma 2 *The algorithm terminates after at most $n \cdot m$ repetitions of the iterative step.*

Proof At each iteration, the total misrepresentation decreases by at least 1, since the μ_{ij} are integers. On the other hand, the total misrepresentation cannot decrease below 0, and is initially at most $n \cdot \max_{i,j} \mu_{ij} \leq n \cdot m$. □

The iterative step of the algorithm can be calculated efficiently: since k is constant, the number of possible cycles of length at most k is polynomial in n. By Lemma 2, the iterative step is repeated polynomially in n and m. We have that the time complexity of Winner-Selection in Monroe's scheme is polynomial — provided we are able to show that the algorithm works!

Lemma 3 *The algorithm returns an optimal assignment.*

Proof Consider a scenario where the algorithm reaches the iterative step, but the current assignment is not optimal. We must show that the algorithm does not terminate at this point. Indeed, let $f_S^* : V \to S$ be a fixed optimal assignment. We consider the voters i such that $f_S(i) = f_S^*(i)$ to be *placed*, and the other voters to be *misplaced*. Assume without loss of generality that f_S^* minimizes the number of misplaced voters among all optimal assignments.

We claim that there is a set of $l \leq k$ voters which can be cyclically right-shifted in a way that places all l voters. Let i_1 be a misplaced voter. In order to place it, it has to be assigned to the candidate $f_S^*(i_1)$. Thus, one of the voters that f_S assigns to $f_S^*(i_1)$ must be misplaced, otherwise f_S is not balanced; call this voter i_2. i_2 can be placed by uprooting a candidate i_3 assigned to $f_S^*(i_2)$. Iteratively repeating this line of reasoning, there must at some stage be a voter i_{d_1}, $d_1 \leq k$, such that $f_S^*(i_{d_1}) = f_S(i_{d_0})$ for some $d_0 < d_1$; this is true, since there are only k distinct candidates in S. Hence, the voters $\{i_{d_0}, i_{d_0+1}, \ldots, i_{d_1}\}$ can be cyclically right-shifted in a way that places all $d_1 - d_0 + 1 = l \leq k$ voters.

For any set of voters that can be placed by cyclic right-shifting, the shift must strictly decrease misrepresentation. Otherwise, by cyclically left-shifting the same set in f_S^*, we can obtain a new optimal and balanced assignment, in which more voters are placed compared to f_S^*; this is a contradiction to our assumption that f_S^* minimizes the number of misplaced voters.

It follows that there must be a set of at most k voters such that cyclically right-shifting the set strictly decreases the misrepresentation. Therefore, the algorithm does not terminate prematurely. □

The proof of Theorem 3 is completed.

6 Conclusion

The fact that the Winner-Selection problem in the Monroe and the Chamberlin-Courant schemes is $\mathcal{NP}$-complete strongly suggests that these

schemes cannot be used in practical settings. In the worst-case, computing the election results could take eons! Viewed positively, this result implies that the integer programming formulation proposed by Potthoff and Brams [12] is in a sense optimal: one cannot hope to find a polynomial-time solution, and different methods can be used to tackle integer programs with reasonable efficiency on average (although, of course, the solution might still take exponential time in the worst-case). Another important aspect of this result is that the type of ballots cast does not affect the complexity: the problem is hard even when misrepresentation values are only 0 or 1.

Our hardness results depend on the number of winners growing with the number of voters and candidates. When the number of winners is small, it is possible to select winners efficiently. This is straightforward in the Chamberlin-Courant scheme, but is more difficult in the Monroe scheme, as the misrepresentation of sets is based on balanced assignments. Our algorithm finds a balanced assignment that minimizes misrepresentation in polynomial time. The analysis also relies (in the proof of Lemma 2) on the misrepresentation values being in $\{0, 1, \ldots, m\}$. That said, our results are quite general, since this assumption holds when approval, ordinal, or any method of reasonable balloting is used. We stress that integer programming cannot be efficiently used in this setting. In fact, in the worst case the solution of an appropriate integer program (of the type proposed by Potthoff and Brams for large electorates) could run exponentially long in the number of candidates, even if the number of winners is constant.

References

1. J. Bartholdi and J. Orlin. Single transferable vote resists strategic voting. *Social Choice and Welfare*, 8:341–354, 1991.
2. J. Bartholdi, C. A. Tovey, and M. A. Trick. The computational difficulty of manipulating an election. *Social Choice and Welfare*, 6:227–241, 1989.
3. J. Bartholdi, C. A. Tovey, and M. A. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare*, 6:157–165, 1989.
4. S. J. Brams and P. C. Fishburn. Voting procedures. In K. J. Arrow, A. K. Sen, and K. Suzumura, editors, *Handbook of Social Choice and Welfare*, chapter 4. North-Holland, 2002.
5. J. R. Chamberlin and P. N. Courant. Representative deliberations and representative decisions: Proportional representation and the Borda rule. *American Political Science Review*, 77(3):718–733, 1983.
6. V. Conitzer, J. Lang, and T. Sandholm. How many candidates are needed to make elections hard to manipulate? In *Proceedings of the International Conference on Theoretical Aspects of Reasoning about Knowledge*, pages 201–214, Bloomington, Indiana, 2003.
7. V. Conitzer and T. Sandholm. Complexity of manipulating elections with few candidates. In *Proceedings of the National Conference on Artificial Intelligence*, pages 314–319, Edmonton, Canada, July 2002.
8. U. Feige. A threshold of ln$n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

9. B. L. Monroe. Fully proportional representation. *American Political Science Review*, 89(4):925–940, 1995.
10. C. H. Papadimitriou. On the complexity of integer programming. *Journal of the Association for Computing Machinery*, 28(4), 1981.
11. C. H. Papadimitriou. *Computational Complexity*. Addison Wesley, 1994.
12. R. F. Potthoff and S. J. Brams. Proportional representation: Broadening the options. *Journal of Theoretical Politics*, 10(2), 1998.
13. M. Sipser. *Introduction to the Theory of Computation*. Course Technology, 1996.