Approximation Algorithms for BalancedCC Multiwinner Rules

Markus Brill
TU Berlin
Berlin, Germany
brill@tu-berlin.de

Piotr Faliszewski
AGH University
Krakow, Poland
faliszew@agh.edu.pl

Frank Sommer
Philipps-Universität Marburg
Marburg, Germany
fsommer@mathematik.uni-marburg.de

Nimrod Talmon
Ben-Gurion University
Be'er Sheva, Israel
talmonn@bgu.ac.il

ABSTRACT

X-BalancedCC multiwinner voting rules constitute an attractive but computationally intractable compromise between the proportionality provided by the Monroe rule and the diversity provided by the Chamberlin–Courant rule. We show how to use the Greedy-Monroe algorithm to get improved approximation results for the X-BalancedCC rules and for the Chamberlin–Courant rule, by appropriately setting a "schedule" for the sizes of virtual districts. We describe a polynomial-time algorithm for computing a schedule that guarantees high approximation ratio, but show that finding the best possible schedule for a given election is NP-hard. We further evaluate our algorithms experimentally and show that they perform very well in practice.

KEYWORDS

multiwinner elections; approximation algorithms; Monroe rule; Chamberlin–Courant rule; greedy algorithms

ACM Reference Format:
Markus Brill, Piotr Faliszewski, Frank Sommer, and Nimrod Talmon. 2019. Approximation Algorithms for BalancedCC Multiwinner Rules. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019*, IFAAMAS, 9 pages.

1 INTRODUCTION

Multiwinner elections model settings where a group of agents (i.e., voters) wishes to select a group of candidates (i.e., a committee) based on their preferences [12]. Classic examples of multiwinner elections include (1) choosing a parliament or (2) selecting a set of movies to put on the entertainment system of an airplane. In the former case, a rule that guarantees *proportional representation* is appropriate. In the latter case, a rule that selects a *diverse* set of movies is appropriate, so that as many voters (i.e., passengers) as possible can find a good movie to watch (disregarding the fact that a large fraction of passengers would watch the same Hollywood blockbuster movie and all the other movies would be watched by small minorities only).

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

Two well-known multiwinner voting rules that are often discussed in the literature, and seem particularly suitable for the example situations mentioned above, are the *Monroe* (M) and *Chamberlin–Courant* (CC) rules [6, 19]. In this paper, we focus on a class of rules that lie "between" the M and CC rules: *X-BalancedCC rules*, recently introduced by Faliszewski and Talmon [14], which strike a compromise between representing voters proportionally, as M does, and selecting a diverse set of candidates, as CC does.

The M and CC rules indeed achieve proportional representation and committee diversity, respectively. Interestingly, even though their goals are quite different, their definitions are very similar (we use the ordinal election model, where each voter ranks the candidates from the most to the least appreciated one):

(1) Under CC, the procedure for selecting a winning committee is as follows. For each committee of a given size k, each voter is assigned to the committee member that he or she ranks highest; if a voter v is assigned to candidate c then we say that c is v's *representative*. The set of voters assigned to a given committee member form a *virtual district*. The rule outputs the committee where, on the average, each voter ranks his or her representative highest.
(2) The M rule proceeds similarly, but requires all virtual districts to be of same size (give or take one voter). Thus, under M we do not necessarily assign each voter to the committee member he or she appreciates most, but find an assignment fulfilling the condition on the virtual districts' sizes while optimizing the average position of a voter's representative.

Faliszewski and Talmon [14] pointed out that the approaches taken by M and CC are quite extreme in that in M, we require the sizes of all virtual districts to be (nearly) identical, while in CC, we do not put any constraints on their sizes. To provide some middle ground, they introduced X-BalancedCC rules, for $X \geq 1$, which require that the largest virtual district can be at most X times larger than the smallest one. Faliszewski and Talmon [14] argued that X-BalancedCC rules still achieve some form of (degressive) proportionality,[1] but are more open to selecting a diverse set of candidates than the pure M rule. For example, when choosing movies for an airplane, an X-BalancedCC rule (e.g., for $X \geq 2$) would not choose just a single blockbuster movie for all the passengers that want to see a movie of this type (like CC might), but rather find a larger selection of such movies, while still providing enough different ones for the remaining passengers.

[1] See the work of Koriyama et al. [15] for a discussion of degressive proportionality.

Unfortunately, X-BalancedCC rules—just like M and CC—are NP-hard to compute [4, 14, 18, 22]. To alleviate this, Faliszewski and Talmon described ILP formulations to be used for medium-sized elections, and adapted an approximation algorithm of Skowron et al. [24] (Algorithm P), originally designed for CC. Their algorithm, however, was quite cumbersome and achieved relatively poor approximation guarantees. Indeed, in some cases their algorithm gave worse approximation guarantees for X-BalancedCC than that of the currently best-known approximation algorithm for M (the GreedyMonroe algorithm of Skowron et al. [24]), even though the M rule is much more constrained than the X-BalancedCC rules.[2]

Our Contribution. Our main contribution is adapting the GreedyMonroe algorithm to X-BalancedCC rules, evaluating its approximation guarantees for this case (which turns out to be significantly better than those of Faliszewski and Talmon [14]; sometimes even better than the best-known approximation guarantees for CC), and demonstrating how to use the algorithm in practice.

Briefly put, to select a committee of size k based on preferences of n voters, GreedyMonroe proceeds in k iterations where in each iteration it selects a group of n/k voters and finds them a representative. We modify and generalize this algorithm by providing it with a vector $(s_1, \ldots, s_k)$ of positive integers, which we call a *schedule*, so that in the i-th iteration it selects s_i voters. By choosing the schedule appropriately, we ensure that GreedyMonroe finds an assignment of committee members to voters where the largest virtual district is at most X times larger than the smallest one (for a given X). On the technical side, we show the following main results:

(1) We describe a polynomial-time algorithm that, for a given number n of voters and committee size k, computes a schedule that guarantees high-quality committees for X-BalancedCC rules, irrespective of the voter preference orders.
(2) We show that the problem of finding a schedule that leads to the best possible results for a given election is NP-hard even if we put no restrictions on the sizes of the virtual districts.
(3) We show experimentally that even without the ability to find perfect schedules for specific elections, it is feasible to prepare a small set of schedules so that after trying GreedyMonroe with each of them and choosing the best committee computed, we obtain very high quality results.

The main message of our paper is that the X-BalancedCC rules (or, rather, their approximate variants) are perfectly usable in practical settings, even for elections whose sizes preclude the use of the ILP formulations of Faliszewski and Talmon [14].

Related Work. Our work relates to the notion of (degressive) proportional representation, which has been studied in quite some depth, albeit often for approval-based elections, where instead of ranking candidates, voters indicate which candidates they approve of. (The M and CC rules can also be adapted to this setting [1, 23].) For example, in this setting Aziz et al. [1] introduced and studied the notions of justified representation (JR) and extended justified representation (EJR), which capture the intuitive notion of proportional representation; later, Sánchez-Fernández [23] introduced proportional justified representation as a middle-ground between JR and EJR. In the same setting, Peters [21] has shown that, in general, proportional representation and strategy-proofness are mutual exclusive (see also the work of Lackner and Skowron [16]). Lackner and Skowron [17], among other topics, discussed proportionality aspects of a class of voting rules which they called Thiele methods. Brill et al. [5] discussed how multiwinner rules relate to apportionment methods (including a discussion of the M rule).

The ordinal setting received somewhat less attention in the recent literature. Aziz et al. [2] adapted the notion of justified representation (and related ones) to the world of ordinal elections, and Elkind et al. [9] discussed a proportionality-related axiom inspired by Dummett [7] (however, only recently Aziz et al. [3] expressed Dummett's ideas as axiomatic properties in a convincing way). For more detailed discussions of multiwinner voting in general, and of proportionality and diversity specifically, we point the readers to the overview of Faliszewski et al. [12].

2 PRELIMINARIES

We mostly follow the notations of Faliszewski and Talmon [14]. In particular, we consider ordinal elections and focus on rules based on the Borda scoring function. For an integer t, we denote $\{1, \ldots, t\}$ by $[t]$. An *election* $E = (C, V)$ consists of a set of candidates $C = \{c_1, \ldots, c_m\}$ and a collection of voters $V = (v_1, \ldots, v_n)$. Each voter v_i is endowed with a *preference order* $\succ_{v_i}$ ranking all candidates from the most to the least preferred one (from the voter perspective). We write $\mathrm{pos}_{v_i}(c_j)$ to denote the position of candidate c_j in the preference order of voter v_i (the most preferred candidate has position 1, the next one has position 2, and so on).

Scoring Functions. Consider an election $E = (C, V)$ with m candidates. A *single-winner scoring function* $\gamma_m : [m] \to \mathbb{R}$ associates a score value with each position in a preference order. For candidate $c \in C$, his or her γ_m-score is defined as:

$$\gamma_m\text{-score}_E(c) = \sum_{v \in V} \gamma_m(\mathrm{pos}_v(c)).$$

We are almost exclusively interested in the family of Borda scoring functions, defined as $\beta_m(i) = m - i$.

Multiwinner Voting Rules. A *multiwinner voting rule* $\mathcal{R}$ is a function that, given an election $E = (C, V)$ and committee size $k \in [|C|]$, outputs a family of winning size-k subsets of candidates (the winning committees). There is quite a large variety of multiwinner voting rules (see, e.g., the overview of Faliszewski et al. [12]), but for our purposes it suffices to consider only a few. For example, the k-Borda rule outputs all size-k committees consisting of candidates with the highest Borda scores. To define the X-BalancedCC rules, on which we focus, we introduce some additional definitions.

Assignment Functions. Let $E = (C, V)$ be an election and k the desired committee size. A k-CC-assignment function is a function $\Phi : V \to C$ which associates each voter with one out of at most k candidates (in other words, we require that $|\Phi(V)| \leq k$). We write $\Psi_{\mathrm{CC}}(k, C, V)$ to denote the set of all k-CC-assignment functions for candidate set C and voter collection V. For a voter v, we refer to the candidate $\Phi(v)$ as v's *representative* (under assignment Φ). A

[2] A careful reader could point out that a less constrained problem does not necessarily need to have more effective approximation algorithms. However, in this case the committees computed by GreedyMonroe could be directly used as X-BalancedCC committees, achieving the same approximation ratios as for M.

k-CC-assignment function Φ induces a size-k committee $W \subseteq C$ if W contains the representatives of all the voters. If W is an induced committee (for function Φ) then for each candidate $c \in W$ we refer to the voters in the set $\Phi^{-1}(c)$ as the *virtual district* of c. Formally, it is possible that some virtual districts are empty.

Let $X \in \mathbb{R}$ be some number, $X \geq 1$ (we refer to it as the *balancedness ratio*). Then, a given k-CC-assignment function is *X-balanced* if for each committee W that it induces the following hold:

(1) $\Phi^{-1}(c) \neq \emptyset$ for each candidate $c \in W$ (i.e., there are no empty virtual districts).
(2) $\frac{\max_{c \in W} |\Phi^{-1}(c)|}{\min_{d \in W} |\Phi^{-1}(d)|} \leq X$ (i.e., the largest virtual district is at most X times larger than the smallest one).

A k-CC-assignment function $\Phi\colon V \to C$ is a *k-Monroe-assignment* function if $|\Phi^{-1}(c)| \in \{0, \lfloor |V|/k \rfloor, \lceil |V|/k \rceil\}$ holds for each $c \in C$. Thus, whenever k divides $|V|$, k-Monroe-assignment functions are exactly 1-balanced (formally, if k does not divide $|V|$, then no 1-balanced assignments exist). For an election $E = (C, V)$ and size-k committee W, we define the *grab-your-best* assignment function $\Phi_{\text{gyb}}(W)\colon V \to C$ so that each voter v is assigned to the member of W they rank highest.

Voter Satisfaction (Score). Consider a k-CC-assignment function $\Phi\colon V \to C$ and a single-winner scoring function γ_m (where $m = |C|$). We say that voter v's *(γ_m-based) satisfaction* from his or her representative is $\gamma_m(\text{pos}_v(\Phi(v)))$. The total satisfaction associated with Φ, denoted $\gamma_m(\Phi)$, is the sum of the voters' satisfactions:

$$\gamma_m(\Phi) = \sum_{v \in V} \gamma_m(\text{pos}_v(\Phi(v))).$$

Here, we focus on Borda scoring functions for defining voter satisfaction. We will typically omit the subscript m in $\beta_m(\Phi)$ as the number of candidates is implicit in the definition of Φ. Often, instead of using the term voter satisfaction we will simply speak of a given assignment's *score*.

Monroe, CC, and X-BalancedCC. We define the Chamberlin–Courant (CC) rule $\mathcal{R}_{\text{CC}}$ [6] as follows. Given an election $E = (C, V)$ and committee size k, the rule outputs all size-k committees W that are induced by k-CC-assignment functions with the highest possible total voter satisfaction according to the Borda scoring function. The Monroe and X-BalancedCC rules (where $X \geq 1$ is a given number), denoted $\mathcal{R}_{\text{Monroe}}$ and $\mathcal{R}_{X\text{-CC}}$, are defined analogously, except that we are restricted to k-Monroe assignment functions and X-balanced k-CC-assignment functions, respectively.

Example 2.1. Consider the following election $E = (C, V)$ with six candidates $C = \{a, b, c, d, e, f\}$, and the following six voters:

$$v_1\colon a \succ d \succ f \succ b \succ e \succ c, \quad v_2\colon a \succ d \succ e \succ f \succ b \succ c,$$
$$v_3\colon a \succ d \succ e \succ f \succ b \succ c, \quad v_4\colon a \succ e \succ d \succ f \succ c \succ b,$$
$$v_5\colon a \succ c \succ e \succ d \succ f \succ b, \quad v_6\colon b \succ c \succ e \succ f \succ d \succ a.$$

We are looking for a committee of size 2. The unique 2-Borda committee is $\{a, d\}$ with total Borda score 43. Under CC the unique winning committee is $\{a, b\}$, with $\{v_1, \ldots, v_5\}$ being a's virtual district and $\{v_6\}$ as b's virtual district, and total voter satisfaction of 30 (highest possible). The unique Monroe winning committee is $\{a, e\}$, with $\{v_1, v_2, v_3\}$ as a's virtual district and $\{v_4, v_5, v_6\}$ being e's virtual district, and total voter satisfaction of 25. The unique 2-CC winning committee is $\{a, c\}$ with $\{v_1, \ldots, v_4\}$ as a's virtual district and $\{v_5, v_6\}$ as c's virtual district, and total voter satisfaction of 28.

Informally, we can think of the CC rule as the ∞-BalancedCC rule. For the cases where the number of voters is a multiple of the committee size (as often will be the case in this paper), the Monroe rule is equivalent to the 1-BalancedCC rule.

Approximation Algorithms. We say that an algorithm is an α-approximation algorithm for the CC rule if it outputs an assignment that achieves at least an α fraction of the highest possible total voter satisfaction, and we refer to the value α as the algorithm's approximation ratio. We define α-approximation algorithms for the M and X-BalancedCC rules analogously.

3 APPROXIMATION ALGORITHMS

In this section we first describe two approximation algorithms for the CC rule—GreedyCC of Lu and Boutilier [18] and Algorithm P of Skowron et al. [24]—and discuss how Faliszewski and Talmon [14] extended the latter to the case of X-BalancedCC rule. We mention several drawbacks of their approach and describe how the GreedyMonroe algorithm—originally designed as an approximation algorithm for the Monroe rule [24]—can be adapted to the case of X-BalancedCC rules. Throughout this section we consider an election $E = (C, V)$ with m candidates and n voters, and committee size k. We start by considering the GreedyCC algorithm.

GreedyCC. The algorithm starts with an empty committee $W_0 = \emptyset$ and performs k iterations. For each $i \in [k]$, in the i-th iteration it chooses a candidate $c \in C \setminus W_{i-1}$ that maximizes the value $\beta(\Phi_{\text{gyb}}(W \cup \{c\}))$ and forms $W_i = W_{i-1} \cup \{c\}$ (if there are several candidates c maximizing the score, one of them is arbitrarily chosen; for the sake of concreteness, let us assume lexicographic tie-breaking).

GreedyCC was introduced by Lu and Boutilier [18], who observed that its approximation ratio is at least $1 - 1/e$ (in essence, this follows directly from the fact that the algorithm greedily approximates a monotone, submodular function [20]). While, in principle, it is possible to adapt GreedyCC to the case of X-BalancedCC rules (e.g., by replacing the grab-your-best assignments used throughout the algorithm with some more involved ones), it appears to be rather cumbersome, and showing that the algorithm preserves the approximation ratio may be involved (for example, such analysis for the case of Monroe, due to Skowron et al. [24], is far more complicated than that for the case of the CC rule).

Algorithm P. Skowron et al. [24] improved upon GreedyCC by providing a polynomial-time approximation scheme (PTAS) for the CC rule (i.e., an algorithm that for any given $\varepsilon > 0$ finds a $(1 - \varepsilon)$-approximate solution in time that depends polynomially on the election size, but which may depend superpolynomially on $1/\varepsilon$). Their approach was largely based on using Algorithm P (which they also introduced). Algorithm P can be seen as a variant of the GreedyCC algorithm and proceeds as follows:

(1) It computes a value $\lambda = \text{W}(k)/k$, where $\text{W}(k)$ is Lambert's W function (which, roughly speaking, grows more slowly than $\log k$). Then it replaces each voter's preference order with a set of λm candidates that the voter ranks on top.

(2) The algorithm forms an empty committee $W = \emptyset$ and executes k iterations, where in each iteration it extends W with a candidate c that appears in the top sets of the largest number of voters; then, all the voters that rank c among top λm positions are removed from further considerations.
(3) The algorithm outputs the assignment $\Phi_{\text{gyb}}(W)$.

Skowron et al. [24] have shown that this algorithm has approximation ratio $1 - 2\text{W}(k)/k$. Faliszewski and Talmon [14] observed that this algorithm (or, rather, its analysis) can be adapted to the case of X-BalancedCC rules by choosing different values of λ. Specifically, they have observed that for a given value λ, one can guarantee that the largest virtual district will have size at most $n\lambda$ and the smallest one will have size at least $n\lambda(1-\lambda)^{k-1}$. Thus, by choosing appropriate λ they were able to control the ratio between the sizes of the largest and smallest virtual district.

Their approach had, however, one serious drawback. The reason why Skowron et al. [24] were able to get a very good approximation bound for their algorithm was that they could assume that the candidate selected in each iteration does not appear among the top λm positions of the remaining voters, which allowed them to use the pigeonhole principle very effectively. Faliszewski and Talmon [14] could not make this assumption because they had to bound the sizes of the virtual districts created and they could not assume that when they add a given candidate c to the committee then they also assign to c and remove all voters that rank c among their top λm positions. In consequence, the theoretical approximation ratio they could guarantee was not great (specifically, it was $\approx (\sqrt[k-1]{1/X} - \frac{k-1}{m})(1 - 1/X(\sqrt[k-1]{1/X}))$). In consequence, even with a number of tweaks they applied, the actual guarantees they could provide (for particular numbers of candidates and voters and for particular committee sizes) were either worse or only marginally better than those achieved by Skowron et al. [24] for the Monroe rule using the GreedyMonroe algorithm.

GreedyMonroe. As GreedyMonroe often obtains better approximation ratios than those achieved by the analysis of Faliszewski and Talmon [14], yet it produces far more constrained committees than many X-BalancedCC rules require (recall that Monroe is very close to being a 1-BalancedCC rule), in this paper we consider how we can design a variant of the GreedyMonroe algorithm that, on the one hand, will take advantage of the full power of X-BalancedCC rules (for values $X > 1$) and, on the other hand, will still maintain high approximation ratio. Below we describe the GreedyMonroe algorithm adapted to our purposes.

In addition to election $E = (C, V)$ with m candidates and n voters, and the committee size k, the algorithm also receives a vector $s = (s_1, \ldots, s_k) \in \mathbb{N}^k$, such that $s_1 + \cdots + s_k \leq n$.[3] We refer to this vector as the *schedule*. The algorithm proceeds as follows:

(1) It starts with an empty assignment function Φ (i.e., with a partial assignment function which does not assign candidates to any of the voters) and with an empty committee $W_0 = \emptyset$.
(2) It performs k iterations, where for each $i \in [k]$, the i-th iteration is as follows. For each candidate $c \in C \setminus W_{i-1}$ it finds a subcollection V_c of s_i voters (that have not been removed from consideration yet) that maximizes the value $s(c) = \beta\text{-score}_{(C,V_c)}(c)$. (In other words, for each not-yet-selected candidate we find a group of s_i voters that jointly assign to him or her the highest total Borda score). We choose the candidate c for whom the value $s(c)$ is highest (breaking ties arbitrarily; in our case, lexicographically), set $W_i = W_{i-1} \cup \{c\}$, extend Φ so that for each voter v from V_c we have $\Phi(v) = c$, and remove the voters of V_c from further consideration.
(3) If $s_1 + \cdots + s_k < n$, then some voters do not have assigned representatives. In this case, we use the following *filling procedure*: We iteratively consider each of these voters and assign him to a committee member $c \in W$ that the voter ranks as highly as possible but whose virtual district is not the largest, unless all virtual districts are of the same size[4]).
(4) We output Φ.

Skowron et al. [24] considered this algorithm for the case of the Monroe rule, that is, for the schedule $(\lceil n/k \rceil, \ldots, \lfloor n/k \rfloor)$. They have shown that in this case the algorithm achieves approximation ratio $\approx 1 - \frac{k-1}{2(m-1)} - \frac{H_k}{k}$, where H_k is the k-th harmonic number.

By extending the GreedyMonroe algorithm with arbitrary schedules, as we do here, and choosing such schedules appropriately, we can ensure that GreedyMonroe finds a valid X-balanced assignment function for a given X. Specifically, if we use a schedule $(s_1, \ldots, s_k)$ such that $\frac{\max_{i \in [k]} s_i}{\min_{j \in [k]} s_j} \leq X$ (we refer to such schedules as *X-balanced*), then GreedyMonroe will output an X-balanced assignment (due to the filling procedure, this holds even if $s_1 + \cdots + s_k < n$). The problem, however, lies in choosing schedules ensuring a high voter satisfaction. The rest of the paper is dedicated to identifying such schedules.

Useful notations. We will use the following notation. For a given election $E = (C, V)$, committee size k, and schedule $(s_1, \ldots, s_k)$, we write $\text{gm}(E, s)$ to denote the total (Borda) satisfaction of the voters from an assignment computed using GreedyMonroe with schedule s (note that the schedule implicitly gives the number of candidates in the committee). For a number n of voters, committee size k, and balancedness ratio X, we write $\text{sched}(n, k, X)$ to denote the set of all schedules $(s_1, \ldots, s_k)$ such that $s_1 + \cdots + s_k \leq n$ and $\frac{\max_{i \in [k]} s_i}{\min_{j \in [k]} s_j} \leq X$ (i.e., the set of all schedules that achieve balancedness ratio X with n voters and k committee members). We define $\text{sched}(n, k)$ analogously, by dropping the constraint on X.

4 APPROXIMATION GUARANTEES

In this section we analyze the worst-case approximation ratios that can be achieved for a given X-BalancedCC rule by using GreedyMonroe with an appropriately chosen schedule. It turns out that in some cases we obtain even better approximation ratios for X-BalancedCC rules than Skowron et al. [24] obtained via Algorithm P for the less constrained CC rule.

Before we proceed with our analysis, in the following example we show that, even though it is natural to expect high-quality schedules to be nonincreasing, this does not need to be the case.

[3] One might wonder why we do not require that $s_1 + \cdots + s_k = n$. Indeed, we find that sometimes it is useful to consider schedules that take into account fewer than n voters.

[4] This procedure may seem a bit strange but its purpose will soon become clear. Alternatively, we could compute an optimal X-balanced assignment for a given committee, but this would require using a somewhat complicated algorithm and we wanted to keep our solution as simple as possible.

Example 4.1. Consider the 4-BalancedCC rule and election $E = (C, V)$ with candidates $C = \{a, b, c, d, e\}$ and the voters:

$$v_1 : a \succ e \succ c \succ d \succ b, \qquad v_2 : a \succ e \succ b \succ d \succ c,$$
$$v_3 : c \succ b \succ e \succ d \succ a, \qquad v_4 : e \succ b \succ c \succ d \succ a,$$
$$v_5 : d \succ b \succ e \succ c \succ a.$$

We seek a committee of size 2. The unique optimal committee is $\{a, b\}$ with total voter satisfaction of 17 with $\{v_1, v_2\}$ as a's virtual district and $\{v_3, v_4, v_5\}$ as b's virtual district. For schedule $(2, 3)$, GreedyMonroe first chooses candidate a (for voters v_1 and v_2) and then candidate b (for the remaining voters). For schedule $(3, 2)$, it first chooses e and then b. Schedules $(1, 4)$ and $(4, 1)$ lead to committees (e, c) and (e, d); all these committee have total satisfaction 16 and, hence, we see that the optimal schedule can be increasing.

Knowing that high-quality schedules may possibly be quite peculiar, we move on to search for schedules that guarantee as high approximation ratios as possible. We borrow our main technical tool from Skowron et al.'s [24] analysis of GreedyMonroe. Let $E = (C, V)$ be an election, let k be the committee size, and let $(s_1, \ldots, s_k)$ be a given schedule. Let $m = |C|$ be the number of candidates and $n = |V|$ be the number of voters. In the first iteration, GreedyMonroe finds s_1 voters and a candidate $c \in C$ such that these s_1 voters assign the highest Borda score to c. By the pigeonhole principle, we note that each of these s_1 voters have to rank c at least on position $t_1 = \lceil \frac{s_1 m}{n} \rceil$ or higher (the reason is that the n voters have altogether $t_1 n$ slots on top t_1 positions and, in the worst case, each of the m candidates has to appear in s_1 of these slots; by requiring that $t_1 n = s_1 m$ we compute the value t_1 (and take its ceiling, for rounding). Thus, GreedyMonroe obtains a total score of at least:

$$s_1 \left(m - \lceil \tfrac{s_1 m}{n} \rceil\right)$$

in the first iteration. By the same reasoning, in the i-th iteration the algorithm finds s_i voters that rank their representative at least on position $t_i = \lceil \frac{s_2 m}{n - (s_1 + \cdots + s_{i-1})} \rceil + i - 1$ (the $i - 1$ addition comes from the fact that the candidates selected in the previous $i - 1$ iterations could be ranked on the top $i - 1$ positions by all the voters) and the algorithm obtains score of at least:

$$s_i \left(m - \left\lceil \frac{s_1 m}{n - (s_1 + \cdots + s_{i-1})} \right\rceil - (i - 1)\right) .$$

Since in the best possible solution every voter ranks his or her representative on the top position, in total we get that GreedyMonroe for m candidates, n voters, and with schedule $(s_1, \ldots, s_k)$ achieves an approximation ratio of at least:[5]

$$\frac{1}{n(m-1)} \cdot \sum_{i=1}^{k} s_i \left(m - \left\lceil \frac{s_1 m}{n - (s_1 + \cdots + s_{i-1})} \right\rceil - (i - 1)\right) . \qquad (1)$$

In consequence, to find an approximation guarantee for a given X-BalancedCC rule it suffices to find an X-balanced schedule that maximizes the value (1). Fortunately, it is possible to compute such schedules in polynomial time.

Remark 1. *Notice that what we do next, in essence, is using an algorithm (specifically, an algorithm based on dynamic programming) in order to design an approximation algorithm for X-BalancedCC, particularly by optimizing the approximation guarantee.*

Proposition 4.2. *There is an algorithm that, given a number n of voters, number m of candidates, committee size k, and two integers L and U, computes a schedule $(s_1, \ldots, s_k)$ that maximizes the value (1), while ensuring that $L \le s_i \le U$ for each $i \in [k]$. The algorithm runs in polynomial time with respect to $n + m$.*

Proof. To specify our algorithm we provide a function that it computes, express this function recursively, and then, by standard dynamic programming techniques, we conclude that it is possible to compute this function in polynomial time with respect to $n + m$.

Let $\text{sched}(n, k, L, U)$ denote the set of schedules $(s_1, \ldots, s_k)$ such that $\sum_{i=1}^{k} s_i \le n$ and for each $i \in [k]$ it holds that $L \le s_i \le U$. For integers N and j and a sequence $(s_j, \ldots, s_k)$ we define:

$$g_j(N, s_j, \ldots, s_k) = \sum_{i=j}^{k} s_i \left(m - \left\lceil \frac{s_i m}{N - (s_j + \cdots + s_{i-1})} \right\rceil - (i - 1)\right).$$

Intuitively, $g_j(N, s_1, \ldots, s_k)$ computes the part of expression (1) that corresponds to the latter $k - j + 1$ elements of the schedule (ignoring the multiplicative constant $\frac{1}{n(m-1)}$), where we interpret N as $n - (s_1 + \cdots + s_{j-1})$. For each pair (N, j) we define:

$$f(N, j) = \max_{(s_j, \ldots, s_k) \in \text{sched}(N, k-j+1, L, U)} g_j(N, s_j, \ldots, s_k).$$

(We implicitly assume that the max operator, when applied to an empty set, returns $-\infty$.) Note that $f(n, 1)$ gives the highest value (1) possible to obtain by any schedule. We express $f(N, j)$ recursively:

$$f(N, j) = \max_{L \le s_j \le U} g_j(N, s_j) + f(N - s_j, j + 1).$$

(We use here $g_j(N, s_j)$ to compute the lower bound on the number of points GreedyMonroe would obtain in the j-th iteration, not counting the following iterations.) Using this formula and standard dynamic programming techniques we can compute $f(n, 1)$ and recover the schedule which leads to its value. □

Using the algorithm from Proposition 4.2 we can compute a schedule that maximizes the value (1) for an X-balanced schedule. Specifically, for an election with n voters, m candidates, committee size k, and for a given value X, we run the algorithm from Proposition 4.2 for all values of L between 1 and n/k, matched with values $U = X \cdot L$. We have ran this procedure for values of $X \in \{1.5, 2, 3, 5, 10\}$ and with the following parameters:

(1) 100 candidates, 100 voters, 10 committee members;
(2) 500 candidates, 500 voters, 20 committee members; and
(3) 1000 candidates, 1000 voters, 100 committee members.

We have compared the achieved approximation ratios against those of Faliszewski and Talmon [14]. We present the results in Table 1. We see that our approach always achieves better guarantees than that of Faliszewski and Talmon [14]; furthermore, the smaller X, the larger the improvement. Interestingly, in some cases (specifically, for $m = n = 500$ and $k = 20$) our approach even gives better approximation guarantees than Algorithm P of Skowron et al. [24] for the Chamberlin–Courant rule. Indeed, this is the first case of achieving a *guarantee* better than that of Algorithm P (although, of course, there were heuristic algorithms that in practice performed better than Algorithm P [13]).

[5] Skowron et al. [24] carried out such analysis for schedules of the form $(n/k, \ldots, n/k)$, obtaining a closed-form approximation ratio for Monroe; their analysis of Algorithm P also is with respect to the score $n(m - 1)$.

Table 1: Approximation guarantees for GreedyMonroe with optimal schedules for a given balancedness ratio X (typeset in bold) and the previously best results achieved by Faliszewski and Talmon [14]; the column marked GM shows the approximation ratios for the classic GreedyMonroe algorithm for $X = 1$, and the column marked Alg. P shows the approximation ratios guaranteed by Algorithm P. In the rows denoted "imp. ratio" we report the ratio of the approximation guarantees of our algorithm and the algorithm of Faliszewski and Talmon.

	GM	$X = 1.5$	$X = 2$	$X = 3$	$X = 5$	$X = 10$	Alg. P
$m = n = 100$		**0.714**	**0.718**	**0.721**	**0.721**	**0.721**	
$k = 10$	0.687	0.658	0.685	0.710	0.716	0.716	0.726
imp. ratio		1.085	1.048	1.015	1.006	1.006	
$m = n = 500$		**0.826**	**0.831**	**0.835**	**0.836**	**0.836**	
$k = 20$	0.808	0.704	0.748	0.787	0.809	0.815	0.813
imp. ratio		1.173	1.11	1.06	1.033	1.025	
$m = n = 1000$		**0.906**	**0.917**	**0.925**	**0.930**	**0.932**	
$k = 100$	0.903	0.718	0.763	0.840	0.878	0.909	0.948
imp. ratio		1.261	1.201	1.101	1.059	1.025	

It is also interesting to consider the actual schedules that our algorithm computed. Below we give the schedules computed for the case of 100 candidates, 100 voters, and committee size 10:

$$\begin{aligned} X = 1.5\colon &\quad (12, 11, 11, 12, 9, 9, 8, 8, 8, 8),\\ X = 2\colon &\quad (12, 12, 12, 11, 11, 9, 8, 8, 6, 6),\\ X = 3\colon &\quad (15, 14, 13, 10, 10, 9, 8, 7, 5, 5),\\ X = 5\colon &\quad (16, 14, 12, 10, 10, 9, 8, 7, 5, 4),\\ X = 10\colon &\quad (16, 14, 12, 10, 10, 9, 8, 7, 5, 4). \end{aligned}$$

These schedules have some interesting features. The first observation is that the schedule for $X = 1.5$ is not monotone (Example 4.1 indeed suggested that we might expect this effect), whereas the other ones are. Our schedules for other parameters were typically not monotone (usually just a few entries were breaking the monotonicity; when we sorted the schedules to be nonincreasing, they gave only slightly worse guarantees). The second observation is that neither of our schedules uses all the 100 voters (the above schedules use either 95 or 96 voters). This seems to be a general feature of all the computed schedules for all the settings (intuitively, not using all the voters means that we cannot get the score values from the omitted ones, but for the used ones we get better bounds on the positions of the representatives). The third observation is that the schedules for $X = 5$ and $X = 10$ are identical and, in fact, achieve balancedness ratio 4. Indeed, for every election size there is an X value beyond which it is impossible to improve the approximation guarantee using our technique (this certainly happens for $X = n$, but typically this value of X is much lower than n).

5 SCHEDULES FOR SPECIFIC ELECTIONS

In the previous section we have shown that our technique is promising and that there are universal schedules that lead to very high approximation ratios. In this section we explore the computational complexity of computing an X-balanced schedule that achieves the highest possible voter satisfaction for a *given* election. Formally, we consider the following computational problem.

Definition 5.1. In the X-Optimal Schedule problem we are given an election $E = (C, V)$, a committee size k, and score value B. We ask if there exists a schedule $s = (s_1, \ldots, s_k) \in \mathrm{sched}(|V|, k, X)$ such that $\mathrm{gm}(E, s) \geq B$. We define the Optimal Schedule problem analogously, except that we do not put constraints on the balancedness ratio of the schedule that we consider.

Unfortunately, for some elections and committee sizes no schedule leads to an optimal solution.

Example 5.2. Consider an election $E = (C, V)$ with five candidates $C = \{a, b, c, d, e\}$ and the following four voters:

$$v_1\colon a \succ d \succ e \succ c \succ b, \quad v_2\colon c \succ a \succ d \succ e \succ b,$$
$$v_3\colon c \succ b \succ d \succ e \succ a, \quad v_4\colon b \succ e \succ d \succ c \succ a.$$

We search for a committee of size 2. The unique optimal committee is $\{a, b\}$ with a total voter satisfaction of 14. Any schedule $s = (s_1, s_2)$ with $s_1 + s_2 \leq 3$ achieves a maximal voter satisfaction of $3(m - 1) = 12$ and thus does not lead to an optimal solution. Schedule $(3, 1)$ leads to committees $\{c, a\}$ or $\{c, b\}$ with a total voter satisfaction of 13; schedule $(2, 2)$ leads to committees $\{c, d\}$ or $\{c, e\}$ with a total voter satisfaction of 13; and schedule $(1, 3)$ leads to committees $\{a, c\}$ and $\{b, c\}$ with a total voter satisfaction of at most 13. Hence, in this example, no schedule is optimal.

Naturally, the above example is not an argument against using the GreedyMonroe algorithm. By their very nature, sometimes approximation algorithms are unable to find optimal solutions and the fact that this holds true also for GreedyMonroe is marginally disappointing, but not surprising. It is, however, a bit more worrying is that the Optimal Schedule problem is intractable (in the next section, however, we describe a workaround for this problem).

Theorem 5.3. *The Optimal Schedule problem is* NP*-hard, even when restricted to decreasing schedules.*

Proof. We describe a reduction from the X3C problem to Optimal Schedule. As input, we are given a universe $U = \{u_1, \ldots, u_{3k}\}$ of elements and a family $\mathcal{T} = \{T_1, \ldots, T_{3k}\}$ of three-element subsets of U, where each element u_i belongs to exactly three different sets. The question is if there is a set $I \subseteq \{1, \ldots, 3k\}$ of k indices such that $\bigcup_{i \in I} T_i = U$. We form an instance of Optimal Schedule with election $E = (C, V)$, committee size k (this is the same k as in the X3C instance), and score requirement B. Below we describe the exact construction of E and B.

Let L be $(10 \cdot k)^6$ (intuitively, L is a large value, relative to the size of the X3C instance). We form candidate set $C = T \cup D$, where $T = \{t_1, \ldots, t_{3k}\}$ contains candidates that correspond to the sets from the X3C instance and D contains $7k(1 + 2 + \cdots + 3k)$ dummy candidates. We have two groups of voters:

(1) In the first group, for each element $u_i \in U$, we have L triples of voters with preference orders of the following form (let T_x, T_y, and T_z be the three sets to which u_i belongs, with $x < y < z$; by putting a set in the description of a preference

order, we mean listing members of this set in some arbitrary but fixed order):

$$u^i_{xyz}: t_x \succ t_y \succ t_z \succ D \succ T \setminus \{t_x, t_y, t_z\},$$
$$u^i_{zxy}: t_z \succ t_x \succ t_y \succ D \succ T \setminus \{t_x, t_y, t_z\},$$
$$u^i_{yzx}: t_y \succ t_z \succ t_x \succ D \succ T \setminus \{t_x, t_y, t_z\}.$$

(2) In the second group, for each set T_i, we have i voters such that each of them (a) ranks t_i on position $m-i$, (b) ranks dummy candidates on the top $7k$ positions (except for the $(m-i)$-th position), and (c) ranks the remaining candidates in some arbitrary but fixed order. Furthermore, each voter in this group ranks different dummy candidates on his top $7k$ positions (this is possible due to the number of dummy candidates in the election).

Altogehter, there are $n = 3k \cdot 3L + (1+2+\cdots+3k) = 9kL + \frac{1}{2}3k(3k+1)$ voters. Intuitively, each voter from the first group corresponds to some member of u_i and ranks highly (among top 3 positions) the sets to which he or she belongs. The voters from the second group are used to connect the schedule that we seek with the sets from $\mathcal{T}$.

Let m be the number of candidates in the election (i.e., $m = |C| = 3k + 7k(1+2+\cdots+3k)$). We set $B = 3kL(m-1) + 3kL(m-2) + 3kL(m-3) = 9kL(m-2)$. Roughly put, this score requirement corresponds to the situation where each voter from the first group is assigned to a candidate that he or she ranks among top 3 positions.

We claim that the answer for the constructed instance of Optimal Schedule is *yes* if and only if the answer for the input X3C instance is *yes*. To show that this is the case, we first assume that there is a set $I = \{i_1, \ldots, i_k\}$ of k indices such that $i_1 < \cdots < i_k$ and $\bigcup_{i \in I} T_i = U$, and we will show that GreedyMonroe for schedule $(9L + i_k, 9L + i_{k-1}, \ldots, 9L + i_1)$ achieves score $\mathrm{gm}(E, s) \geq B$.[6]

Given this schedule, in the first iteration GreedyMonroe considers each candidate and finds $9L + i_k$ voters that rank this candidate as highly as possible. Let us focus on some candidate t_j. There are $3L$ voters in the first group that rank t_j on the first position, $3L$ voters that rank t_j on the second position, and $3L$ voters that rank t_j on the third position. GreedyMonroe also selects $\min(j, i_k)$ voters from the second group that rank t_j on position j, and $\max(i_k - j, 0)$ voters (possibly from either of the groups) that rank t_j at position $7k$ or lower. Altogether, candidate t_j is associated with score at most:

$$9L(m-2) + \min(j, i_k)(m-j) + \max(i_k - j, 0)(m - 7k)\,.$$

Indeed, for $j \geq i_k$ this is exactly the score that the candidate achieves. For $j = i_k$, the score is $9L(m-2) + i_k(m-i_k)$, and for $j > i_k$ it is $9L(m-2) + i_k(m-j)$. Thus, we see that t_{i_k} obtains higher score than each candidate t_j with $j > i_k$. On the other hand, if $j < i_k$, then t_j gets score:

$$9L(m-2) + j(m-j) + (i_k - j)(m - 7k)\,,$$

which, as shown by simple calculations, also is lower than that of t_{i_k}. Finally, each dummy candidate is associated, at best, with a single voter (from the second group) that ranks him or her on the top position and $9L + i_k - 1$ voters (from the first group) that rank him or her on the fourth position, giving total score $(9L + i_k)(m-4) + 3$, which is far below that of t_{i_k}. Thus in the first iteration GreedyMonroe selects i_k.

By a similar analysis—and noting that the set I gives an exact cover of U—we see that in the next iteration GreedyMonroe chooses $t_{i_{k-1}}$, then $t_{i_{k-2}}$, and so on, until t_{i_1}. Each of these candidates contributes at least $9L(m-2)$ to the score of the created assignment and, so, the total score of the assignment is at least $9kL(m-2) = B$.

Now, assume that there is a schedule $s = (s_1, \ldots, s_k)$ such that $\mathrm{gm}(E, s) \geq B$. We will show that this implies that there are k sets in $\mathcal{T}$ whose union is U. Let us consider the assignment Φ computed by GreedyMonroe for our election with schedule s and, for the sake of contradiction, let us assume that there is at least one element $u_\ell \in U$ such that u_ℓ belongs to sets T_x, T_y, and T_z but there is no voter that is assigned to any of t_x, t_y, and t_z (note that if such an element u_i did not exist, then this would mean that there is an exact cover of elements from U with k sets from $\mathcal{T}$).

We now compute the upper bound for $\beta(\Phi)$, that is, the score that is associated with the assignment β. By definition, assignment Φ involves at most k different candidates. In the first group of voters, each candidate is ranked first by at most $3L$ voters. Similarly, in this group, each candidate is ranked second by at most $3L$ voters, and is ranked third by at most $3L$ voters. As there are $9kL$ voters in the first group, the highest score that is possible by assigning k candidates is $3kL(m-1) + 3kL(m-2) + 3kL(m-3) = 9kL(m-2)$. However, due to our assumption regarding element u_ℓ, there are $3L$ voters that to whom Φ can, at best, assign a candidate that they rank on the fourth position. Thus the score that Φ receives from voters in the first group is at most $9kL(m-2) - 3L(m-3) + 3L(m-4) = 9kL(m-2) - 3L$. Since the voters in the second group can provide at most score $(1 + \cdots + 3k)(m-1) \leq (3k)^2(m-1)$, Φ has score at most:

$$9kL(m-2) - 3L + (3k^2)(m-1)\,,$$

which is less than B (because $m < (10k)^4$ and $L = (10k)^6$). This contradicts the assumption that Φ has score at least B and, so, our assumption regarding element u_ℓ must have been false. In consequence, we know that if there is a schedule that lead GreedyMonroe to find an assignment with score at least B then there must be an exact cover of elements from U with k sets from T. □

A similar proof also shows that the problem is W[1]-hard when parametrized by the committee size. On the positive side, our problems are fixed-parameter tractable when parametrized either by the number of voters or by the number of candidates.

Theorem 5.4. *The Optimal Schedule problem is* W[1]*-hard when parametrized by the committee size, but is fixed-parameter tractable with respect to the number of voters and with respect to the number of candidates.*

To conclude, it is intractable to compute optimal schedules for given elections. While one might try to form integer linear programs (ILPs) to compute such schedules using off-the-shelf ILP solvers, we expect that it would be more effective to simply compute optimal committees using ILP solvers. Thus in the next section we suggest a simple workaround for the problem and evaluate its effectiveness.

[6] Note that, for this schedule, we have $(9L + i_k) + \ldots + (9L + i_1) = 9kL + i_1 + i_2 + \ldots + i_k < 9kL + 3k(3k+1)/2$. In other words, under this schedule we need to use the fill-in procedure to assign a few of the voters; we make use of the property that the set $sched(n, k)$ of all possible schedules contains all vectors $(s_1, \ldots, s_k) \in \mathbb{N}$ such that $s_1 + \ldots + s_k \leq n$.

Table 2: Ratios of the average position of a voter's representative in committees computed using the Multischedule GreedyMonroe algorithm and in the optimal X-BalancedCC committee, for elections with 100 candidates and 100 voters, generated using the Pólya-Eggenberger urn model, with a given value of α, for $X \in \{1.5, 2, 3, 5, 10\}$. Numbers in brackets give the same values, but for GreedyMonroe with schedules from Section 4.

	$\alpha = 0$	$\alpha = 0.1$	$\alpha = 0.25$	$\alpha = 0.5$
$X = 1.5$	1.38 (1.30)	1.46 (1.32)	1.36 (1.27)	1.23 (1.19)
$X = 2$	1.33 (1.26)	1.44 (1.36)	1.35 (1.30)	1.23 (1.23)
$X = 3$	1.31 (1.22)	1.44 (1.31)	1.36 (1.29)	1.23 (1.25)
$X = 5$	1.29 (1.20)	1.43 (1.36)	1.38 (1.42)	1.21 (1.39)
$X = 10$	1.29 (1.20)	1.43 (1.43)	1.44 (1.66)	1.30 (1.69)

6 EXPERIMENTAL EVALUATION

In Section 4 we have shown a way of computing schedules that guarantees high voter satisfactions; indeed, one could simply use GreedyMonroe with those schedules to compute good X-BalancedCC committees, but such a variant of the algorithm would not adapt to specific elections. However, Faliszewski and Talmon [14] in their experimental evaluation of X-BalancedCC rules observed that for a large class of elections (they considered the Pólya-Eggenberger urn model [8] and Euclidean elections [10, 11]) the vectors of the sizes of the virtual districts (sorted from largest to smallest) in optimal solutions can be categorized to the following types:

(1) "Sigmoidal" vectors, consisting of a sequence of large districts followed by a sequence of small districts.
(2) Vectors which are linearly decreasing.
(3) Vectors which are exponentially decreasing.

Thus we have hand-picked a set of schedules of the above types for the case of 100 candidates, 100 voters, committee size 10, and $X \in \{1.5, 2, 3, 5, 10\}$. Below we show such schedules for $X = 3$ (for each X we prepared three "sigmoid" type schedules, with different numbers of large and small districts, one linear schedule and one exponential schedule; since the entries of the schedules need to be integer and we chose ones that sum up to the number of voters, the linear and exponential ones are not perfectly linear/exponential):

$$\text{Sig}_1\colon (18, 18, 18, 7, 7, 7, 7, 6, 6, 6)$$
$$\text{Sig}_2\colon (15, 15, 15, 15, 15, 5, 5, 5, 5, 5)$$
$$\text{Sig}_3\colon (13, 12, 12, 12, 12, 12, 12, 5, 5, 5)$$
$$\text{Linear}\colon (15, 14, 13, 12, 11, 9, 8, 7, 6, 5)$$
$$\text{Exponential}\colon (17, 15, 13, 11, 10, 8, 7, 7, 6, 6)$$

Our algorithm now proceeds as follows. To compute an X-BalancedCC committee for a given election E (with 100 candidates and 100 voters), and committee size 10, we run GreedyMonroe for all such schedules with balancedness ratio up to the given X (so for $X = 3$ we would try 15 schedules: five for balancedness ratio 1.5, five for balancedness ratio 2, and five for balancedness ratio 3) and we choose the committee with the highest score. We refer to this algorithm as the *Multischedule GreedyMonroe* algorithm.

We have tested our algorithm on elections generated according to the Pólya-Eggenberger urn model. In this model we assume some parameter $\alpha > 0$ (known as the parameter of contagion). To generate an election with n voters and m candidates, we proceed as follows. Initially, we have an urn with all $m!$ possible preference orders. To generate a vote, we draw a random preference order from the urn, add it to the election, and return it to the urn together with $\alpha m!$ copies (so the larger the parameter α, the more likely it is that many identical votes appear in the election). For $\alpha = 0$ the urn model is equivalent to the Impartial Culture model, where each preference order is equally likely.

We have considered values for α from the set $\{0, 0.1, 0.25, 0.5\}$. For each value of α and for each $X \in \{1.5, 2, 3, 5, 10\}$, we have generated 150 elections and computed optimal X-BalancedCC committees for them using the ILP formulations of Faliszewski and Talmon [14]. Then for each of these elections, we have run our GreedyMonroe algorithm with the optimal schedule from Section 4 and the variant of the algorithm described in this section (which uses multiple hand-picked schedules and chooses the best outcome seen). In all our experiments, we observed average approximation ratios between 97% and 99%. Thus, following Faliszewski et al. [13], instead of reporting these approximation ratios, we report ratios x/y, where x is the average position of a voter's representative in a solution computed using a given algorithm and y is the average position of a voter's representative in the optimal committee. We report the results in Table 2.

It turns out that GreedyMonroe with our schedules from Section 4 performs very well for most of the test cases, except for the situations where α is either 0.25 or 0.5 and, at the same time, X is 5 or 10. This suggests that the intractability results from Section 5 do not have a strong bite and in practice one can either use the schedules from Section 4 (if one wants high quality and low running time) or one can design a set of a few varied schedules to try within Multischedule GreedyMonroe (but one should then include the schedules from Section 4 in the mix).

7 CONCLUSIONS

We have adapted the GreedyMonroe algorithm to the case of X-BalancedCC rules. We have shown that our variant of the algorithm provides better approximation guarantees than the previous algorithm of Faliszewski and Talmon [14] and, indeed, in some settings it even achieves higher approximation ratios than Algorithm P of Skowron et al. [24], designed for the Chamberlin–Courant rule. While it is computationally intractable to find parameters for which our variant of GreedyMonroe achieves best performance for a given election, we have shown experimental evidence that it is possible to prepare a handful of parameter settings to try, to ensure that the algorithm achieves good results.

ACKNOWLEDGMENTS

This work was initiated at the 2018 research retreat of the Algorithmics and Computational Complexity (AKT) group of TU Berlin. This work was supported by the Deutsche Forschungsgemeinschaft under grants BR 4744/2-1 (Markus Brill) and KO 3669/4-1 (Frank Sommer), and by Poland's National Science Centre (NCN) under project 2016/21/B/ST6/01509 (Piotr Faliszewski).

REFERENCES

[1] H. Aziz, M. Brill, V. Conitzer, E. Elkind, R. Freeman, and T. Walsh. 2017. Justified Representation in Approval-Based Committee Voting. *Social Choice and Welfare* 48, 2 (2017), 461–485.

[2] H. Aziz, E. Elkind, P. Faliszewski, M. Lackner, and P. Skowron. 2017. The Condorcet Principle for Multiwinner Elections: From Shortlisting to Proportionality. In *Proceedings of IJCAI-2017*. 84–90.

[3] H. Aziz and B. Lee. 2018. *The Expanding Approvals Rule: Improving Proportional Representation and Monotonicity*. Technical Report arXiv:1708.07580v2 [cs.GT]. arXiv.org.

[4] N. Betzler, A. Slinko, and J. Uhlmann. 2013. On the Computation of Fully Proportional Representation. *Journal of Artificial Intelligence Research* 47 (2013), 475–519.

[5] M. Brill, J.-F. Laslier, and P. Skowron. 2018. Multiwinner Approval Rules as Apportionment Methods. *Journal of Theoretical Politics* 30, 3 (2018), 358–382.

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

[7] M. Dummett. 1984. *Voting Procedures*. Oxford University Press.

[8] F. Eggenberger and G. Pólya. 1923. Über die statistik verketteter vorgänge. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik* 3, 4 (1923), 279–289.

[9] E. Elkind, P. Faliszewski, P. Skowron, and A. Slinko. 2017. Properties of Multiwinner Voting Rules. *Social Choice and Welfare* 48, 3 (2017), 599–632.

[10] J. M. Enelow and M. J. Hinich. 1984. *The spatial theory of voting: An introduction*. Cambridge University Press.

[11] J. M. Enelow and M. J. Hinich. 1990. *Advances in the spatial theory of voting*. Cambridge University Press.

[12] P. Faliszewski, P. Skowron, A. Slinko, and N. Talmon. 2017. Multiwinner Voting: A New Challenge for Social Choice Theory. In *Trends in Computational Social Choice*, U. Endriss (Ed.). AI Access Foundation.

[13] P. Faliszewski, A. Slinko, K. Stahl, and N. Talmon. 2018. Achieving fully proportional representation by clustering voters. *Journal of Heuristics* 24, 5 (2018), 725–756.

[14] P. Faliszewski and N. Talmon. 2018. Between Proportionality and Diversity: Balancing District Sizes under the Chamberlin–Courant Rule. In *Proceedings of AAMAS-2018*. IFAAMAS, 14–22.

[15] Y. Koriyama, J. Laslier, A. Macé, and R. Treibich. 2013. Optimal Apportionment. *Journal of Political Economy* 121, 3 (2013), 584–608.

[16] M. Lackner and P. Skowron. 2018. Approval-Based Multi-Winner Rules and Strategic Voting. In *Proceedings of IJCAI-2018*. 340–346.

[17] M. Lackner and P. Skowron. 2018. Consistent Approval-Based Multi-Winner Rules. In *Proceedings of ACM EC-2018*. 47–48.

[18] T. Lu and C. Boutilier. 2011. Budgeted Social Choice: From Consensus to Personalized Decision Making. In *Proceedings of IJCAI-2011*. 280–286.

[19] B. Monroe. 1995. Fully proportional representation. *American Political Science Review* 89, 4 (1995), 925–940.

[20] G. Nemhauser, L. Wolsey, and M. Fisher. 1978. An Analysis of Approximations for Maximizing Submodular Set Functions. *Mathematical Programming* 14, 1 (Dec. 1978), 265–294.

[21] D. Peters. 2018. Proportionality and Strategyproofness in Multiwinner Elections. In *Proceedings of AAMAS-2018*. 1549–1557.

[22] A. Procaccia, J. Rosenschein, and A. Zohar. 2008. On the Complexity of Achieving Proportional Representation. *Social Choice and Welfare* 30, 3 (2008), 353–362.

[23] L. Sánchez-Fernández, E. Elkind, M. Lackner, N. Fernández, J. A. Fisteus, P. Basanta Val, and P. Skowron. 2017. Proportional Justified Representation. In *Proceedings of AAAI-2017*. 670–676.

[24] P. Skowron, P. Faliszewski, and A. Slinko. 2015. Achieving Fully Proportional Representation: Approximability Result. *Artificial Intelligence* 222 (2015), 67–103.