Necessary President in Elections with Parties

Katarína Cechlárová
P.J. Šafárik University
Košice, Slovakia
katarina.cechlarova@upjs.sk

Ildikó Schlotter
ELTE Centre for Economic and Regional Studies
Budapest, Hungary
Budapest University of Technology and Economics
Budapest, Hungary
schlotter.ildiko@krtk.elte.hu

ABSTRACT

Consider an election where the set of candidates is partitioned into parties, and each party must choose exactly one candidate to nominate for the election held over all nominees. The Necessary President problem asks whether a candidate, if nominated, becomes the winner of the election for all possible nominations from other parties.

We study the computational complexity of Necessary President for several voting rules. We show that while this problem is solvable in polynomial time for Borda, Maximin, and Copeland$^{\alpha}$ for every $\alpha \in [0, 1]$, it is coNP-complete for general classes of positional scoring rules that include ℓ-Approval and ℓ-Veto, even when the maximum size of a party is two. For such positional scoring rules, we show that Necessary President is W[2]-hard when parameterized by the number of parties, but fixed-parameter tractable with respect to the number of voter types. Additionally, we prove that Necessary President for Ranked Pairs is coNP-complete even for maximum party size two, and W[1]-hard with respect to the number of parties; remarkably, both of these results hold even for constant number of voters.

KEYWORDS

elections, parties, candidate nomination, necessary president, computational complexity, fixed-parameter tractability

ACM Reference Format:
Katarína Cechlárová and Ildikó Schlotter. 2026. Necessary President in Elections with Parties. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/RYRS8869

1 INTRODUCTION

Most political elections are preceded by a turbulent and intense period when parties want to decide which candidate to nominate for the election. The nomination process may take the form of primaries, or may involve more complex, strategic decisions that are not only based on the candidates' traits as viewed by the party members but also on the preferences of voters. Indeed, as the election approaches, parties may realize that the candidate previously picked by the party—e.g., the winner of a primary—has a low support in the polls when compared to the nominees of other parties, and thus needs to be replaced.

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/RYRS8869

Among recent examples of such strategic nominations, the most famous one is perhaps the replacement of Joe Biden by Kamala Harris before the 2024 US presidential election [32]. In Poland, the largest opposition group Civic Platform replaced her primary election winner Małgorzata Kidawa-Błońska by Rafał Trzaskowski after a significant drop in her support before the 2020 presidential elections [20]. Notably, in neither case was the substitute candidate able to ensure victory.

Sometimes the main purpose of parties is not victory itself, but rather to prevent a certain candidate from winning, leading to various forms of strategic nomination. Such a process was witnessed in the 2022 Hungarian elections, where opposition parties decided to cooperate and create a joint list of nominees in order to be able to defeat the ruling party Fidesz [33]. A similar cooperation was carried out in the 2023 Turkish elections where the opposition formed a six-party alliance to nominate Kemal Kılıçdaroğlu in the hope of defeating the ruling president Recep Tayyip Erdoğan [25].

Several interesting questions arise in such situations. In this paper, we address one of the most basic ones: Can a given nominee p participating in an election be defeated with a judicious choice of nominations from all remaining parties? Or will p necessarily win, irrespective of the nominees chosen by all other parties?

To study this topic, we use the formal model of candidate nomination as introduced by Faliszewski et al. [12] where parties are interpreted as sets of candidates, and each party has to nominate exactly one of its candidates for the upcoming election. Faliszewski et al. assumed that parties know the preferences of *all* voters over *all* potential candidates, and studied two problems in this setting: The Possible President problem asks whether a given party can nominate one of its candidates in such a way that he or she becomes a winner of the election for *some* nominations from other parties, while the Necessary President problem asks if a given nominee of the party will be a winner *irrespective of all other nominations.*

Faliszewski et al. [12] concentrated on Plurality. For this voting rule they proved that the Possible President problem is NP-complete and the Necessary President problem is coNP-complete if voters' preferences are unrestricted. Motivated by these hardness results, they focused on structured preferences and showed that Necessary President admits a polynomial-time algorithm for single-peaked profiles. By contrast, Possible President remains NP-complete even on 1D-Euclidean profiles but admits a polynomial-time algorithm if the elections are restricted to single-peaked profiles where the candidates of each party appear consecutively on the societal axis.

In contrast to the steadily growing research concerning the classical and parameterized computational complexity of the Possible President problem for a whole range of different voting

Table 1: Our results for Necessary President. The column 'Param./Const.' contains the considered parameters or their restriction to a constant; 'eff. comp.' stands for voting rules where the winners of an election can be computed efficiently, i.e., in polynomial time. For Copeland$^\alpha$, α is in $[0, 1]$.

Voting rule	Param./Const.	Complexity	Reference
Borda	–	in P	Thm. 3.1
Short	$s = 2$	coNP-complete	Thm. 3.2
	t	W[2]-hard, in XP	Thm. 3.4, Obs. 2
	τ	in FPT	Thm. 3.6
Veto-like	$s = 2$	coNP-complete	Thm. 3.3
	t	W[2]-hard, in XP	Thm. 3.5, Obs. 2
	τ	in FPT	Thm. 3.7
Copeland$^\alpha$	–	in P	Thm. 4.1
Maximin	–	in P	Thm. 4.2
Ranked Pairs	$s = 2$, $\|V\| = 12$	coNP-complete	Thm. 4.3
	t, $\|V\| = 20$	W[1]-hard, in XP	Thm. 4.3, Obs. 2
eff. comp.	s, t	in FPT	Obs. 2

rules [4, 23, 24, 30, 31], not much is known about the Necessary President problem. Cechlárová et al. [4] generalized the intractability result of Faliszewski et al. [12] by showing that Necessary President is coNP-complete for Plurality with run-off, as well as for the voting rules ℓ-Approval and ℓ-Veto for every positive integer ℓ; all these results hold in the restricted setting where each party has at most two candidates. The authors also provided integer programs for Necessary President for a wide range of voting rules and performed computational experiments based on real and synthetic election data. Complementing the tractability result by Faliszewski et al. [12] for Necessary President for Plurality on single-peaked preference profiles, Misra [24] showed that the same problem is polynomial-time solvable on single-crossing profiles.

1.1 Our Contribution

We study the computational complexity of the Necessary President problem for several positional scoring rules and a variety of Condorcet-consistent voting rules; our results are summarized in Table 1. The positional scoring rules we consider include the Borda rule as well as the set of all so-called *short* and *Veto-like* voting rules, introduced by Schlotter et al. [31]. Additionally, we consider some of the most well-known Condorcet-consistent voting rules: Copeland$^\alpha$ for all $\alpha \in [0, 1]$, Maximin, and Ranked Pairs.

We classify the complexity of Necessary President for each of these voting rules as either polynomial-time solvable or coNP-complete. Furthermore, we apply the framework of parameterized complexity to deal with the computationally intractable cases: we examine how certain natural parameters of a given instance influence the computational complexity of the Necessary President problem. The parameters we consider are the following:

- t: the number of parties;
- s: the maximum size of a party;
- τ: the number of *voter types*, where two voters are of the same type if they have the same preferences over the candidates;
- $|V|$: the number of voters.

For each of the voting rules for which Necessary President turns out to be coNP-complete, we settle its parameterized complexity for every possible combination of the above four parameters as either (i) fixed-parameter tractable (FPT), (ii) W[1]- or W[2]-hard and in XP, or (iii) para-NP-hard.

1.2 Related Work

The line of research most closely related to this paper—the problem of strategic candidate nomination by parties preceding an election—was initiated by Faliszewski et al. [12] and, apart from the papers already mentioned [4, 12], has focused on the Possible President problem; see Section 1.2.1. Different models describing elections with parties from an algorithmic viewpoint are briefly discussed in Section 1.2.2. Another related topic is *strategic candidacy* where candidates are independent and have full power over deciding whether to run for the election or not; see Section 1.2.3.

Candidate nomination can also be seen as part of the broader topic of elections where the set of candidates is not fixed; such a scenario appeared already in the seminal paper by Bartholdi et al. [19] in the form of control by adding or deleting candidates by an election chair. For an overview of various other results dealing with this type of election control, see for example Faliszewski and Rothe [14], Chen et al. [5] or Erdélyi et al. [11].

1.2.1 The Possible President Problem. The results of Faliszewski et al. [12] for Possible President for Plurality voting have been extended to other voting rules. Lisowski [23] dealt with tournament solutions and showed that Possible President for Condorcet rule (which selects the Condorcet winner if it exists, and the empty set otherwise) can be solved in polynomial time but is NP-complete for the Uncovered Set rule. Cechlárová et al. [4] studied the problem for positional scoring voting rules, among them ℓ-Approval, ℓ-Veto, and Borda, and Condorcet-consistent rules such as Copeland, Llull, and Maximin. In addition, this paper provides integer programs for the Possible President as well as Necessary President problem for all the studied voting rules and computational experiments with these integer programs applied to real and synthetic elections.

The parameterized complexity of Possible President was first studied by Misra [24] who strengthened the results of Faliszewski et al. [12] by showing that Possible President for Plurality is para-NP-hard when parameterized by the size of the largest party even in profiles that are both single-peaked and single-crossing; she also proved that when parameterized by the number of parties, the problem is W[1]-hard and in XP for general preferences but becomes FPT on the 1D-Euclidean domain.

A detailed multivariate complexity analysis of the Possible President problem for several classes of positional scoring rules has been provided by Schlotter et al. [31] who studied the parameterized complexity of these problems with respect to the four parameters studied in this paper. The same multivariate approach was taken by Schlotter and Cechlárová [30] for two types of Condorcet-consistent voting rules: Copeland$^\alpha$ for every $\alpha \in [0, 1]$ and Maximin.

1.2.2 Elections Involving Parties. Harrenstein et al. [16] introduced a model where voters and candidates are both described by their position on the real line (modeling the political spectrum), and each party has to choose its nominee from among its potential candidates under the assumption that each voter votes for the closest nominee. The authors showed that a Nash equilibrium (NE) is not guaranteed to exist even in a two-party game, and finding a NE is NP-complete in general but can be computed in linear time for two parties.

The above model was extended by Deligkas et al. [7]: they associated each candidate with a cost and studied the parameterized complexity of the equilibrium computation problem under several natural parameters such as the number of different positions of the candidates, the so-called discrepancy and span of the nominees, and the maximum overlap of the parties.

Harrenstein and Turrini [17] considered district-based elections. In each district, voters rank the nominated candidates and elect the Plurality winners, and parties have to strategically place their candidates in districts so as to maximize the number of their nominees that get elected. The authors showed that deciding the existence of pure NE for these games is NP-complete if the party size is bounded by a constant and Σ_2^P-complete in general.

Perek et al. [27] introduced a model where voters, *not* candidates, are partitioned into parties, with voters of the same party voting in the same way. The authors proposed to measure the threat to the so-called leading party P—the party whose favored candidate is the expected winner of the election—by the maximum number of voters who can abandon P for another party without changing the winner of the election, and by the minimal number of voters that must leave P to ensure that the winner changes. Perek at al. [27] and in a follow-up paper Guo et al. [15] studied the computational complexity of these problems for several different voting rules.

1.2.3 *Strategic Candidacy.*

In strategic candidacy games as introduced by Dutta et al. [10], there is a finite set of voters and candidates, and some of the candidates may also be voters. Both voters and candidates have preferences over the set of all candidates and it is assumed that every candidate prefers herself to all other candidates. Candidates then strategically choose whether to join or leave the election to obtain a favored outcome. Brill and Conitzer [2] extended the analysis to the case when also voters may act strategically, defining several stability notions.

Lang et al. [22] illustrated the phenomenon of strategic candidacy with the 2017 French presidential election where the centrist candidate Bayrou withdrew to help Macron qualify to the second round (successfully), and green candidate Jadot withdrew to help the socialist candidate Hamon qualify (not successfully). The authors presented an analysis of such games for a list of common voting procedures, and examined the question of whether such games possess a pure strategy NE in which the outcome is the same as if all candidates run (called *genuine equilibria*). They also established a strong relationship between equilibria of candidacy games and a form of voting control by adding or removing candidates, where candidates must consent to addition or deletion.

Other works take into account also the monetary and reputational costs of running an electoral campaign. The model by Obraztsova et al. [26] assumes that running an electoral campaign incurs some costs, whereas the one proposed by Lang et al. [21] assumes that such a campaign yields incidental benefits. Obraztsova et al. [26] and Polukarov et al. [29] studied equilibrium dynamics in candidacy games in an iterative scenario. For a more detailed description of these works, see the full version of our paper [3].

2 PRELIMINARIES

We use the notation $[i] = \{1, 2, \dots, i\}$ for each positive integer i.

An election $\mathcal{E} = (C, V, \{\succ_v\}_{v \in V})$ consists of a finite set C of candidates, a finite set V of voters, and the preferences of voters over the set C of candidates. We assume that the preferences of voter v are represented by a strict linear order $\succ_v$ over C where $c \succ_v c'$ means that voter v *prefers* candidate c to candidate c'. If two voters have the same preferences, they are said to be of the same *type*; the number of voter types in V will be denoted by τ.

We denote the set of all elections over a set C of candidates by $\mathbb{E}_C$. A *voting rule* $\mathcal{R} : \mathbb{E}_C \to 2^C$ chooses a set of *winners* of the election.

We shall also assume that a partition $\mathcal{P} = \{P_1, \dots, P_t\}$ of the set C of candidates is given; each set P_j is interpreted as a *party* that has to decide about whom among its potential candidates to nominate for the election. Formally, a *reduced election* arises after each party has nominated a unique candidate, leading to a set $C' \subseteq C$ of nominees such that $|C' \cap P_j| = 1$ for each $j \in [t]$. In the reduced election $\mathcal{E}_{C'} = (C', V, \{\succ'_v\}_{v \in V})$ each voter $v \in V$ restricts her original preference relation $\succ_v$ over C to C', yielding $\succ'_v$.

Now we formulate the problem studied in this paper.

> **Problem** NECESSARY PRESIDENT for voting rule $\mathcal{R}$.
> **Instance:** A tuple $I = (\mathcal{E}, \mathcal{P}, p)$ where $\mathcal{E} = (C, V, \{\succ_v\}_{v \in V})$ is an election with candidate set C and voter set V, a partition $\mathcal{P}$ of C into parties, and a *distinguished candidate* $p \in C$.
> **Question:** Is p a *necessary president*, that is, is it true that for all possible nominations from parties not containing p, leading to a set C' of nominees with $p \in C'$, the distinguished candidate p is a winner of the reduced election $\mathcal{E}_{C'}$ over C'?

Notice that we consider the *non-unique winner model*, so we define p to be a necessary president if it is *among the winners* in all possible reduced elections that contain p. While the party containing the distinguished candidate p may contain additional candidates, those are irrelevant in the context of NECESSARY PRESIDENT.

In order to verify that a given candidate p is *not* a necessary president, it suffices to present a reduced election containing p in which p is not a winner. Thus, we have the following fact.

OBSERVATION 1. *NECESSARY PRESIDENT is in* coNP *for each voting rule where winner determination can be done in polynomial time.*

As observed for POSSIBLE PRESIDENT by, e.g., Schlotter et al. [31], there are at most s^t possible nominations by the parties where s and t are the maximum size and the number of parties, respectively. Thus, a simple brute force approach yields the following:

OBSERVATION 2. *NECESSARY PRESIDENT is in* XP *when parameterized by the number t of parties and in* FPT *when parameterized by both t and the maximum size s of a party for each voting rule where winner determination can be done in polynomial time.*

We assume familiarity with the framework of parameterized complexity; see the books [6, 9] for an introduction.

2.1 Voting Rules

In this paper we shall deal with two classes of voting rules, *positional scoring rules* and *Condorcet-consistent rules.* For all considered voting rules the winners can be computed efficiently (that is, in polynomial time) for any election, so by Observation 1 we know that Necessary President is in coNP for all the voting rules studied in this paper.[1]

2.1.1 Positional Scoring Rules. A positional scoring rule for elections involving t candidates is associated with a *scoring vector* $(a_1, a_2, \dots, a_t)$ where $a_1 \geq a_2 \geq \cdots \geq a_t$ and at least one inequality is strict. For each candidate c, the rule assigns a_i points to c for each voter that ranks c on the i^{th} position of her preference list. The *winners* of the election are the candidates with the highest *score*, that is, the total number of points obtained. We write $\mathrm{scr}_{\mathcal{E}}(c)$ for the score of candidate c in an election $\mathcal{E}$.

We deal with the following (classes of) positional scoring rules.

Short scoring rules, introduced by Schlotter et al. in [31], are defined by scoring vectors with only a constant number of non-zero positions, i.e., having the form $(a_1, a_2, \dots, a_\ell, 0, \dots, 0)$ for some constant ℓ. In other words, voters in such elections only allocate points to their ℓ most preferred candidates. This class of voting rules contains the well-known scoring rule *ℓ-Approval* for fixed ℓ, corresponding to the scoring vector with ones in their first ℓ positions and zeros afterwards. The case $\ell = 1$ is *Plurality* where voters only allocate a single point to their most preferred candidate.

Veto-like scoring rules have scoring vectors that contain some value a on every position except for the last ℓ positions for some constant ℓ, i.e., they have the form $(a, \dots, a, a_1, a_2, \dots, a_\ell)$ for some constant $\ell \geq 1$ and $a > a_1$. In other words, voters in such elections distinguish only their ℓ least favored candidates. Veto-like scoring rules include *ℓ-Veto*, whose scoring vector is $(1, 1, \dots, 1, 0, 0, \dots, 0)$ with exactly ℓ zeros; the case $\ell = 1$ is called *Veto*.

Finally, we also consider the *Borda voting rule* which is described by the scoring vector $(t-1, t-2, \dots, 1, 0)$. Thus the number of points that a candidate c receives from a voter $v \in V$ is the number of candidates ranked worse than c in the preference list of v.

2.1.2 Condorcet-Consistent Rules. For two candidates $c, c' \in C$, let $N_{\mathcal{E}}(c, c')$ denote the number of voters who prefer candidate c to candidate c' in election $\mathcal{E}$; we shall omit the subscript when $\mathcal{E}$ is clear from the context. If $N_{\mathcal{E}}(c, c') > N_{\mathcal{E}}(c', c)$ we say that candidate c *defeats* candidate c' in $\mathcal{E}$; if $N_{\mathcal{E}}(c, c') = N_{\mathcal{E}}(c', c)$ and $c \neq c'$, then candidates c and c' are *tied* in $\mathcal{E}$. The *Condorcet winner* is a candidate that defeats all other candidates; a voting rule is *Condorcet-consistent* if it always selects the Condorcet winner whenever it exists.

The *Copeland$^\alpha$ voting rule* was defined by Faliszewski et al. [13] for some constant $\alpha \in [0, 1]$. This voting rule takes all pairs (c, c') of distinct candidates. Considering their head-to-head comparisons, it allocates 1 point to the candidate defeating the other and allocates 0 points to the defeated one; being tied earns α points to both candidates. Formally, the score received by c on the basis of the head-to-head comparison of c with c' in $\mathcal{E}$ is

$$\mathrm{Cpl}^{\alpha}_{\mathcal{E}}(c, c') = \begin{cases} 1 & \text{if } c \text{ defeats } c' \text{ in } \mathcal{E}; \\ \alpha & \text{if } c \text{ and } c' \text{ are tied in } \mathcal{E}; \\ 0 & \text{if } c \text{ is defeated by } c' \text{ in } \mathcal{E}. \end{cases} \quad (1)$$

Then the Copeland$^\alpha$ score of candidate c is computed as the sum $\mathrm{Cpl}^{\alpha}_{\mathcal{E}}(c) = \sum_{c' \in C \setminus \{c\}} \mathrm{Cpl}^{\alpha}_{\mathcal{E}}(c, c')$. The winners of $\mathcal{E}$ are all candidates with the maximum score. Copeland$^\alpha$ for $\alpha = 1$ is called the *Llull* rule, and we refer to the case $\alpha = 0$ as the *Copeland* rule.[2]

In the *Maximin voting rule*, the Maximin score of candidate c in election $\mathcal{E}$ is $\mathrm{MM}_{\mathcal{E}}(c) = \min_{c' \in C \setminus \{c\}} N_{\mathcal{E}}(c, c')$. In other words, the Maximin score of a candidate c is the largest integer r such that for every other candidate c', there exist r voters who prefer c to c'. Again, the winners of $\mathcal{E}$ are the candidates with maximum score.

The *Ranked Pairs voting rule* uses the so-called majority graph of the election: a directed graph $D_{\mathcal{E}} = (C, A)$ where vertices are candidates and (c, c') is an arc for two distinct candidates c and c' if and only if c defeats c' in pairwise comparison in the election $\mathcal{E}$. The winner determination process for Ranked Pairs builds an acyclic[3] arc set by considering the set of candidate pairs in A, examined in non-increasing order of their *weight*, where the weight of $(c, c') \in A$ is $N_{\mathcal{E}}(c, c')$. Starting from an empty arc set F, this process checks whether the currently examined arc can be added to F without creating any cycles, and if so, adds it to F; the process then proceeds with the next arc. For simplicity, we only consider arcs present in the majority graph $D_{\mathcal{E}}$ and, hence, no arcs between tied candidates. Nonetheless, this process usually necessitates some tie-breaking which determines the ordering of arcs with the same weight. The winners are all candidates with no incoming arcs in F. We remark that our results are not dependent on any particular tie-breaking method, but hold for arbitrary tie-breaking methods.

3 RESULTS FOR POSITIONAL SCORING RULES

In this section we present our results on the parameterized complexity of the Necessary President problem for three types of positional scoring rules. In Section 3.1 we deal with the Borda rule, while in Section 3.2 we study short and Veto-like rules.

3.1 The Borda Rule

We start by showing that Necessary President for the Borda voting rule is polynomial-time solvable. This tractability result is somewhat surprising in view of the fact that the closely related Possible President problem is computationally hard for Borda [31].

Theorem 3.1. Necessary President *for Borda is polynomial-time solvable.*

Proof. We propose a polynomial-time algorithm that solves Necessary President for Borda voting; see Algorithm NP-Borda for a pseudocode. Let $I = (\mathcal{E}, \mathcal{P}, p)$ be our input instance.

Assume that there exists a set $C' \ni p$ of nominated candidates for which p is *not* a winner in the reduced election $\mathcal{E}_{C'}$ over C'. Let P be the party containing p. Algorithm NP-Borda first guesses some candidate $w \in P_w \in \mathcal{P} \setminus \{P\}$ whose score in $\mathcal{E}_{C'}$ exceeds the score of p, and then greedily nominates a candidate from every remaining party. Note that by "guessing" w we mean iterating over all possibilities for choosing it.

[1] We remark that in the case of Ranked Pairs, efficient winner determination assumes some tie-breaking method.

[2] Notice that some papers, in particular [13], use the term Copeland rule for Copeland$^{0.5}$.

[3] The arc set F is *acyclic* if there is no directed cycle in the spanned digraph (C, F); see the textbook by Diestel [8] for basic graph terminology.

Algorithm NP-Borda proceeds by computing the following value for each candidate $c \in C \setminus (P \cup P_w)$:

$$\Delta(c) = |\{v : v \in V, w \succ_v c \succ_v p\}| - |\{v : v \in V, p \succ_v c \succ_v w\}|. \quad (2)$$

Notice that the quantity $\Delta(c)$ captures the excess of the score of w over the score of p that results from nominating candidate c.

Next, for each party $\widetilde{P} \in \mathcal{P} \setminus \{P, P_w\}$, Algorithm NP-Borda nominates a candidate $c_{\widetilde{P}} \in \widetilde{P}$ maximizing $\Delta(c_{\widetilde{P}})$ and checks whether p is not a winner in the resulting election; if so, it outputs "no." If the algorithm has explored all possible guesses for w but has not returned "no", then it returns "yes."

Algorithm NP-Borda Solving Necessary President for Borda.

Input: An instance $(\mathcal{E}, \mathcal{P}, p)$ of Necessary President with candidate set C and $p \in P \in \mathcal{P}$.

```
1: for all w ∈ C \ P do
2:     Let P_w be the party in 𝒫 containing w.
3:     for all c' ∈ C \ (P ∪ P_w) do
4:         Compute Δ(c) as in (2).
5:     Set C̃ = {p, w}.
6:     for all P̃ ∈ 𝒫 \ {P, P_w} do
7:         Add some candidate c_P̃ ∈ arg max_{c∈P̃} Δ(c) to C̃.
8:     if p is not a winner in ℰ_C̃ then return "no".
9: return "yes".
```

Let us prove the correctness of Algorithm NP-Borda. It is clear that whenever Algorithm NP-Borda returns "no", then it does so correctly, because for a set $\widetilde{C}$ of nominated candidates (containing exactly one candidate from each party in $\mathcal{P}$), candidate p nominated by party P is *not* a winner in the reduced election $\mathcal{E}_{\widetilde{C}}$ over $\widetilde{C}$.

Hence, it remains to prove that whenever the input is a "no"-instance of Necessary President, Algorithm NP-Borda returns "no." Let C' be the set of nominees in some reduced election $\mathcal{E}_{C'}$ where p is a nominee but not a winner. Then there exists some candidate $w \in P_w \in \mathcal{P} \setminus \{P\}$ with $\mathrm{scr}_{\mathcal{E}_{C'}}(w) > \mathrm{scr}_{\mathcal{E}_{C'}}(p)$. Let us denote by $V^{p \succ w}$ the set of those voters who prefer p to w, and let $V^{w \succ p} = V \setminus V^{p \succ w}$ denote the rest of the voters.

Our key observation is the following: if for some voter $v \in V^{w \succ p}$ there are exactly i nominees in $\mathcal{E}_{C'}$ that v prefers to p but not to w, then the Borda rule allocates $i + 1$ points more to w than to p due to voter v in $\mathcal{E}_{C'}$. Similarly, if for some voter $v \in V^{p \succ w}$ there are exactly i nominees in $\mathcal{E}_{C'}$ that v prefers to w but not to p, then Borda allocates $i + 1$ points more to p than to w due to voter v. Therefore, we get that

$$\begin{aligned}
0 &< \mathrm{scr}_{\mathcal{E}_{C'}}(w) - \mathrm{scr}_{\mathcal{E}_{C'}}(p) = \\
&= \sum_{v \in V^{w \succ p}} \left(\Big| \bigcup_{\substack{c \in C', \\ w \succ_v c \succ_v p}} \{c\} \Big| + 1 \right) - \sum_{v \in V^{p \succ w}} \left(\Big| \bigcup_{\substack{c \in C', \\ p \succ_v c \succ_v w}} \{c\} \Big| + 1 \right) \\
&= \sum_{c \in C' \setminus \{p, w\}} \Delta(c) + |\{v : v \in V, w \succ_v p\}| - |\{v : v \in V, p \succ_v w\}| \\
&\leq \sum_{c \in \widetilde{C} \setminus \{p, w\}} \Delta(c) + |\{v : v \in V, w \succ_v p\}| - |\{v : v \in V, p \succ_v w\}| \\
&= \sum_{v \in V^{w \succ p}} \left(\Big| \bigcup_{\substack{c \in \widetilde{C}, \\ w \succ_v c \succ_v p}} \{c\} \Big| + 1 \right) - \sum_{v \in V^{p \succ w}} \left(\Big| \bigcup_{\substack{c \in \widetilde{C}, \\ p \succ_v c \succ_v w}} \{c\} \Big| + 1 \right) \\
&= \mathrm{scr}_{\mathcal{E}_{\widetilde{C}}}(w) - \mathrm{scr}_{\mathcal{E}_{\widetilde{C}}}(p).
\end{aligned}$$

where $\widetilde{C}$ is the set computed by Algorithm NP-Borda on lines 5–7 during the iteration where w is picked on line 3. Note that the inequality follows from the fact that each nominee $c \in \widetilde{C}$ maximizes $\Delta(c)$ among all candidates within its own party, as ensured by line 7. It follows that Algorithm NP-Borda will find on line 8 that p is not a winner of the election $\mathcal{E}_{\widetilde{C}}$ because $\mathrm{scr}_{\mathcal{E}_{\widetilde{C}}}(w) > \mathrm{scr}_{\mathcal{E}_{\widetilde{C}}}(p)$, and hence outputs "no" as required.

Note that there are less than $|C|$ possibilities to choose w. The values $\Delta(c)$ can be computed in time $O(|V| \cdot |C|)$ and then $\widetilde{C}$ can also constructed in time $O(|C|)$. Finally, the score of p and w in the reduced election over $\widetilde{C}$ can also be computed in $O(|V| \cdot |C|)$ time, so the total running time of Algorithm NP-Borda is $O(|C|^2 \cdot |V|)$. ◻

3.2 Short and Veto-Like Scoring Rules

In this section we examine the computational complexity of the Necessary President problem in detail for all short and Veto-like scoring rules. We begin by proving that Necessary President for such voting rules is coNP-complete even if the maximum party size is $s = 2$. Thus, for short scoring rules we generalize a result by Faliszewski et al. [12] who showed that Necessary President for Plurality is coNP-complete.

The proofs of Theorems 3.2 and 3.3 are based on a polynomial reduction from (2,2)-E3-SAT, the problem of deciding whether a given 3-CNF formula where each variable occurs twice as a positive and twice as a negative literal is satisfiable; this problem was proved to be NP-complete by Berman et al. [1]. These proofs, as well as the proofs of all statements marked with an asterisk are deferred to the full version [3].

Theorem 3.2 (★). *Let $\mathcal{R}$ be a short voting rule based on a positional scoring vector that has the form $(a_1, a_2, \ldots, a_\ell, 0, \ldots, 0)$ for some constant $\ell \geq 1$ such that $a_\ell > 0$. Then Necessary President for $\mathcal{R}$ is* coNP*-complete even if the maximum party size is $s = 2$.*

Theorem 3.3 (★). *Let $\mathcal{R}$ be a Veto-like scoring voting rule, based on a positional scoring vector that has the form $(a, \ldots, a, a_1, a_2, \ldots, a_\ell)$ for some constant $\ell \geq 1$ such that $a > a_\ell$. Then Necessary President for $\mathcal{R}$ is* coNP*-complete even if the maximum party size is $s = 2$.*

Given the intractability results in Theorems 3.2 and 3.3, we investigate the parameterized complexity of Necessary President for short and Veto-like scoring rules. In particular, we consider the number t of parties and the number τ of voter types as parameters.

3.2.1 Parameterizing by the Number of Parties. Here we show that Necessary President is W[2]-hard for both short and Veto-like scoring rules with parameter t. Our proofs use similar ideas as the proofs of Theorems 1 and 6 in [31], respectively. For both classes of scoring rules, we provide a parameterized reduction from the classic Hitting Set problem which is W[2]-hard when parameterized by the size of the desired hitting set (see Cygan et al. [6]).

Theorem 3.4 (★). *Let $\mathcal{R}$ be a short voting rule, based on a positional scoring vector of the form $(a_1, a_2, \ldots, a_\ell, 0, \ldots, 0)$ for some* *constant $\ell \geq 1$ such that $a_\ell > 0$. Then* Necessary President *for* $\mathcal{R}$ *is* W[2]*-hard when parameterized by t, the number of parties.*

Theorem 3.5 (★). *Let $\mathcal{R}$ be a Veto-like voting rule, based on a scoring vector of the form $(a, \dots, a, a_1, a_2, \dots, a_\ell)$ for some constant $\ell \geq 1$ such that $a > a_1$. Then* Necessary President *for* $\mathcal{R}$ *is* W[2]*-hard when parameterized by t, the number of parties.*

3.2.2 Parameterizing by the Number of Voter Types. Next, we show that Necessary President becomes fixed-parameter tractable for both short and Veto-like positional scoring rules when parameterized by the number τ of voter types. This contrasts sharply our intractability results for parameterizing by the number or maximum size of parties, as presented in Theorems 3.2–3.5.

Theorem 3.6. *Let $\mathcal{R}$ be a short voting rule, based on a scoring vector of the form $(a_1, a_2, \dots, a_\ell, 0, \dots, 0)$ for some $\ell \geq 1$ such that $a_\ell > 0$. Then* Necessary President *for* $\mathcal{R}$ *is* FPT *when parameterized by τ, the number of voter types.*

Proof sketch. We present Algorithm NP-Short to solve an instance $I = (\mathcal{E}, \mathcal{P}, p)$ of Necessary President for $\mathcal{R}$ in FPT time with parameter τ, the number of different voter types in $\mathcal{E}$. Let $V = V_1 \cup \dots \cup V_\tau$ be the partitioning of the voters by their types, i.e., all voters in V_i for some $i \in [\tau]$ have the same preferences over C.

Assume that there exists a set $C' \ni p$ of nominated candidates for which p is *not* a winner in the reduced election $\mathcal{E}_{C'}$ over C'. Let P be the party containing p. Algorithm NP-Short guesses the following information about $\mathcal{E}_{C'}$:

- A candidate $w \in C'$ whose score in $\mathcal{E}_{C'}$ exceeds that of p. There are $|C \setminus P|$ possibilities to choose w.
- The *structure* of $\mathcal{E}_{C'}$, defined as follows. We first define a function $\mathrm{prt} : [\tau] \times [\ell] \to \mathcal{P}$ as follows. For some voter type $i \in [\tau]$ and index $j \in [\ell]$, let $\mathrm{prt}(i, j)$ denote the party containing the candidate at the j^{th} position of the votes from V_i in the reduced election $\mathcal{E}_{C'}$. For some pairs (i, j) and (i', j') in $[\tau] \times [\ell]$, we write $(i, j) \sim (i', j')$ if $\mathrm{prt}(i, j) = \mathrm{prt}(i', j')$. The *structure of* $\mathcal{E}_{C'}$ is the family $\mathcal{Q}$ of equivalence classes of the relation $\sim$, which is a partitioning of $[\tau] \times [\ell]$. Note that there are at most $(\tau\ell)^{\tau\ell}$ possibilities to choose $\mathcal{Q}$.
- The equivalence class $Q_w \in \mathcal{Q}$ containing all pairs (i, j) for which $\mathrm{prt}(i, j) = P_w$. There are $|\mathcal{Q}| \leq \tau\ell$ possibilities to choose Q_w.
- The set Q_p containing all pairs (i, j) for which $\mathrm{prt}(i, j) = P$. Note that either $Q_p \in \mathcal{Q}$ or $Q_p = \emptyset$. Therefore, there are $|\mathcal{Q} \setminus \{Q_w\}| + 1 \leq \tau\ell$ possibilities to choose Q_p.

After guessing the above described information about $\mathcal{E}_{C'}$ (where by "guessing" we mean trying all possibilities), Algorithm NP-Short proceeds by creating an auxiliary bipartite graph G as follows. The vertex set of G is $\hat{\mathcal{P}} \cup \hat{\mathcal{Q}}$ where $\hat{\mathcal{P}} = \mathcal{P} \setminus \{P, P_w\}$ and $\hat{\mathcal{Q}} = \mathcal{Q} \setminus \{Q_p, Q_w\}$. For each $Q \in \mathcal{Q}$, let us define

$$Q^+ = Q \cup \{(i, \ell+1) : i \in [\tau], \nexists j \in [\ell] \text{ such that } (i, j) \in Q\}.$$

Intuitively Q^+ represents the situation of a party whose nominee (i) obtains the j^{th} position in the votes from voters of V_i in $\mathcal{E}_{C'}$ for each $(i, j) \in Q$ with $j \in [\ell]$, and (ii) obtains a position after the ℓ^{th} position in the votes from voters of V_i in $\mathcal{E}_{C'}$ for each $(i, \ell+1) \in Q^+$.

A candidate $c \in C \setminus (P \cup P_w)$ is *well placed (with respect to p and w) for* $Q \in \hat{\mathcal{Q}}$ if for each $i \in [\tau]$ the following hold:

(i) if $(i, j) \in Q^+$ and $(i, j') \in Q_p^+$ for some j and j', then either $j = j' = \ell + 1$ or c precedes p in the preferences of voters from V_i if and only if $j < j'$;
(ii) if $(i, j) \in Q^+$ and $(i, j'') \in Q_w^+$ for some j and j'', then either $j = j'' = \ell + 1$ or c precedes w in the preferences of voters from V_i if and only if $j < j''$.

We can now define the auxiliary graph G: some party $\widetilde{P} \in \hat{\mathcal{P}}$ is connected by an edge with some $Q \in \hat{\mathcal{Q}}$ in G if and only if there exists a candidate $c \in \widetilde{P}$ that is well placed with respect to p and w for Q. Given candidate c and some $Q \in \hat{\mathcal{Q}}$, it can be checked in $O(\tau\ell)$ time whether c is well placed for Q, so the graph G can be computed in $O(|C|\tau^2\ell^2)$ time.

After computing the graph G, Algorithm NP-Short also computes a set $\mathcal{S} \subseteq \mathcal{P}$ of *secure* parties which are those parties S that contain at least one candidate c_S that is *safe*, meaning that for each $i \in [\tau]$ where $(i, \ell + 1) \notin Q_w^+$, voters in V_i prefer w to c_S; we fix one such candidate c_S for each secure party S. Intuitively, such candidates can be safely nominated in the sense that they will not prevent w from obtaining the required points in $\mathcal{E}$.

Next, Algorithm NP-Short computes a matching M in G covering all non-secure parties in $\hat{\mathcal{P}} \setminus \mathcal{S}$ and all equivalence classes in $\hat{\mathcal{Q}}$; if no such matching exists, it discards the current set of guesses. Each party $\widetilde{P} \in \hat{\mathcal{P}}$ that is covered by an edge $(\widetilde{P}, Q) \in M$ nominates a candidate that is well placed for Q, whereas each party S not covered by an edge of M (which is necessarily a secure party) nominates the candidate c_S. Finally, the algorithm checks whether these nominations together with p and w yield a reduced election $\mathcal{E}^\star$ in which p is not a winner; if so, it returns "no." If the algorithm has explored all possible guesses but has not returned "no", then it returns "yes."

Note that there are at most $|C| \cdot (\tau\ell)^{\tau\ell+2}$ possibilities for picking $w, \mathcal{Q}, Q_p$, and Q_w. Once these guesses are fixed, the bipartite auxiliary graph G can be computed in $O(|C|\tau^2\ell^2)$ time as we have already argued. The bottleneck in each iteration is the computation of the matching on line 12 of Algorithm NP-Short. Since G has at most $|\mathcal{P}| + \tau\ell \leq |C| + \tau\ell$ vertices, we can compute a matching in G that covers the required set of vertices in $O((|C| + \tau\ell)^{2.5})$ time by, e.g., the Hopcroft–Karp algorithm [18]. This yields a total running time of $O(|C|^{3.5} \cdot (\tau\ell)^{\tau\ell+4.5})$ which is fixed-parameter tractable with respect to τ, as ℓ is a constant.

For the correctness of our algorithm, see the full version [3]. ⊿

We next present an algorithm that solves Necessary President for Veto-like scoring rules in FPT time when parameterized by τ.

Theorem 3.7. *Let $\mathcal{R}$ be a Veto-like voting rule, based on a scoring vector of the form $(a, \dots, a, a_1, a_2, \dots, a_\ell)$ for some constant $\ell \geq 1$ where $a > a_1$. Then* Necessary President *for* $\mathcal{R}$ *is* FPT *when parameterized by τ, the number of voter types.*

Proof. Let $(\mathcal{E}, \mathcal{P}, p)$ be the input instance of Necessary President. First, if $|\mathcal{P}| \leq \ell\tau$, then we consider $\mathcal{R}$ as an ℓ'-short voting rule for $\ell' = |\mathcal{P}| \leq \tau\ell$ and apply Algorithm NP-Short. As shown in the proof of Theorem 3.6, the running time is $O(|C|^{3.5} \cdot (\tau^2\ell)^{\tau^2\ell+4.5})$ which is fixed-parameter tractable for τ because ℓ is a constant.

Algorithm NP-Short Solving Necessary President for short voting rules.

Input: An instance $(\mathcal{E}, \mathcal{P}, p)$ of Necessary President over candidate set C and $p \in P \in \mathcal{P}$.

```
for all w ∈ C \ P do
    for all partitioning Q of [τ] × [ℓ] do    ▷ Q: structure of ℰ
        for all Q_w ∈ Q and Q_p ∈ (Q \ {Q_w}) ∪ {∅} do
            Let P_w be the party in 𝒫 containing w.
            Let 𝒫̂ = 𝒫 \ {P, P_w} and Q̂ = Q \ {Q_p, Q_w}.
            Let E = ∅ and 𝒮 = ∅.
            for all c ∈ P_c ∈ 𝒫̂ and Q ∈ Q̂ do
                if c is well placed for Q then add (P_c, Q) to E.
            Create the graph G = (𝒫̂ ∪ Q̂, E).
            for all S ∈ 𝒫̂ do
                if ∃ a safe candidate c_S ∈ S then add S to 𝒮.
            if ∃ a matching M in G covering Q̂ ∪ (𝒫̂ \ 𝒮) then
                Set C_M = {p, w}.
                for all (P̃, Q) ∈ M do
                    add to C_M a candidate of P̃ well placed for Q.
                for all S ∈ 𝒮 not covered by M do add c_S to C_M.
                if p is not a winner in ℰ_{C_M} then return "no".
return "yes".
```

Second, if $|\mathcal{P}| > \ell\tau$, then there will be at least one nominated candidate in each reduced election $\mathcal{E}'$ that achieves the maximum possible score of $|V| \cdot a$. This means that p is not a winner in $\mathcal{E}'$ if and only if p has score less than $|V| \cdot a$, i.e., it is ranked among the ℓ least favorite nominees for some voter. To decide whether this is possible, it suffices the check whether

$$|\{\widetilde{P} \in \mathcal{P} : c \succ_v p \text{ for some } c \in \widetilde{P}\}| \geq |\mathcal{P}| - \ell \tag{3}$$

for some voter $v \in V$. Clearly, this can be checked in polynomial time. If Inequality (3) holds for some $v \in V$, then we return "no", otherwise we return "yes".

To see the correctness of this algorithm, assume that Inequality (3) holds for some voter $v \in V$. Then nominating p and nominating a candidate preferred to p for every other party where this is possible, we get that p will be preceded by at least $|\mathcal{P}| - \ell$ nominees in the preferences of v in the resulting election $\mathcal{E}'$. Hence, v allocates less than a points to p, yielding $\mathrm{scr}_{\mathcal{E}'}(p) < |V| \cdot a$, which in turn implies that p is not a winner in $\mathcal{E}'$. Assume now that Inequality (3) does not hold for any voter $v \in V$. Then irrespective of the nominations from the parties, for each voter $v \in V$ there will exist less than $|\mathcal{P}| - \ell$ parties whose nominee is preferred by v to p. Thus, p receives a points from each voter, yielding a total score of $|V| \cdot a$, the maximum obtainable score, which ensures that p is a winner in the resulting election. □

4 RESULTS FOR CONDORCET-CONSISTENT VOTING RULES

In this section we deal with three Condorcet-consistent voting rules. We show that Necessary President is polynomial-time solvable for Copeland$^\alpha$ as well as for Maximin. By contrast, Necessary President for Ranked Pairs is computationally hard even for a constant number of voters.

Theorem 4.1. *For each $\alpha \in [0, 1]$, Necessary President for Copeland$^\alpha$ is polynomial-time solvable.*

Proof. We propose Algorithm NP-Copeland that solves Necessary President for Copeland$^\alpha$ voting in polynomial time; see the full version [3] for a pseudocode. Let $(\mathcal{E}, \mathcal{P}, p)$ be our input instance with election $\mathcal{E} = (C, V, \{\succ_v\}_{v \in V})$ and party P containing p.

Assume that there exists a set $C' \ni p$ of nominated candidates for which p is *not* a winner in the reduced election $\mathcal{E}_{C'}$ over C'. Algorithm NP-Copeland first guesses a candidate $w \in C' \setminus P$ such that $\mathrm{Cpl}^\alpha_{\mathcal{E}_{C'}}(w) > \mathrm{Cpl}^\alpha_{\mathcal{E}_{C'}}(p)$. Let P_w be the party containing w.

Recall that $\mathrm{Cpl}^\alpha_{\mathcal{E}_{C'}}(c, c')$ for two candidates c and c' is the score received by c resulting from the head-to-head comparison of c with c' in $\mathcal{E}_{C'}$, see its definition in Equation (1). Let us now define $\Delta(w, p) = \mathrm{Cpl}^\alpha_{\mathcal{E}_{C'}}(w, p) - \mathrm{Cpl}^\alpha_{\mathcal{E}_{C'}}(p, w)$; note that this value is the same for all reduced elections $\mathcal{E}$ containing both p and w, and can be computed easily from the election $\mathcal{E}$. Then the difference in the Copeland$^\alpha$ score of w and p in $\mathcal{E}_{C'}$ can be expressed as

$$\mathrm{Cpl}^\alpha_{\mathcal{E}_{C'}}(w) - \mathrm{Cpl}^\alpha_{\mathcal{E}_{C'}}(p) = \Delta(w, p) + \sum_{c \in C' \setminus \{p, w\}} \Delta_c(w, p) \tag{4}$$

where $\Delta_c(w, p) = \mathrm{Cpl}^\alpha_{\mathcal{E}_{C'}}(w, c) - \mathrm{Cpl}^\alpha_{\mathcal{E}_{C'}}(p, c)$ for each candidate $c \in C' \setminus \{p, w\}$ that is nominated in $\mathcal{E}$. Hence, the value $\Delta_c(w, p)$ reflects the difference resulting in the score of w and p from their comparison with some candidate $c \in C' \setminus \{p, w\}$. Notice that $\Delta_c(w, p)$ is the same for all reduced elections of $\mathcal{E}$ containing p, w, and c, and can be calculated according to the following cases:

- (i) if w defeats c and c defeats p, then $\Delta_c(w, p) = 1$;
- (ii) if w defeats c and p is tied with c, then $\Delta_c(w, p) = 1 - \alpha$;
- (iii) if w is tied with c and p is defeated by c, then $\Delta_c(w, p) = \alpha$;
- (iv) if w is tied with c and p defeats c, then $\Delta_c(w, p) = \alpha - 1$;
- (v) if w is defeated by c and p is tied with c, then $\Delta_c(w, p) = -\alpha$;
- (vi) if w is defeated by c and p defeats c, then $\Delta_c(w, p) = -1$;
- (vii) if both p and w defeat c, or are both tied with c, or are both defeated by c, then $\Delta_c(w, p) = 0$.

After guessing w, Algorithm NP-Copeland nominates a candidate $c_{\widetilde{P}} \in \widetilde{P}$ for each party $\widetilde{P} \in \mathcal{P} \setminus \{P, P_w\}$ maximizing $\Delta_c(w, p)$ over $\widetilde{P}$. Finally, the algorithm computes $\mathrm{Cpl}^\alpha_{\mathcal{E}_{\widetilde{C}}}(w) - \mathrm{Cpl}^\alpha_{\mathcal{E}_{\widetilde{C}}}(p)$ according to Equation (4) for the obtained reduced election $\mathcal{E}_{\widetilde{C}}$; if this value is positive, it returns "no." If the algorithm has explored all possible guesses for w without returning "no", then it returns "yes."

Let us prove the correctness of Algorithm NP-Copeland. It is clear that whenever Algorithm NP-Copeland returns "no", then it does so correctly, because $\mathrm{Cpl}^\alpha_{\mathcal{E}_{\widetilde{C}}}(w) > \mathrm{Cpl}^\alpha_{\mathcal{E}_{\widetilde{C}}}(p)$ and so candidate p is not a winner of the reduced election $\mathcal{E}_{\widetilde{C}}$ over the candidate set $\widetilde{C}$ constructed by the algorithm.

Conversely, assume that a candidate w is a winner of some reduced election $\mathcal{E}_{C'}$ over a set $C' \ni p$ of nominated candidates in which p is not a winner. Let P_w be the party containing w, and let $c_{\widetilde{P}}$ denote the nominee of some party $\widetilde{P}$ other than P or P_w in $\mathcal{E}_{C'}$. Since w is a winner in $\mathcal{E}_{C'}$ but p is not, we have $\mathrm{Cpl}^\alpha_{\mathcal{E}_{C'}}(w) > \mathrm{Cpl}^\alpha_{\mathcal{E}_{C'}}(p)$. Using Equation (4) and the algorithm's choice for the nominated candidates, we get that

$$\mathsf{Cpl}^{\alpha}_{\mathcal{E}_{\widetilde{C}}}(w) - \mathsf{Cpl}^{\alpha}_{\mathcal{E}_{\widetilde{C}}}(p) = \Delta(w,p) + \sum_{\widetilde{P}\in\widetilde{\mathcal{P}}} \max\{\Delta_c(w,p) : c \in \widetilde{P}\}$$
$$\geq \Delta(w,p) + \sum_{\widetilde{P}\in\widetilde{\mathcal{P}}} \Delta_{c_{\widetilde{P}}}(w,p) = \mathsf{Cpl}^{\alpha}_{\mathcal{E}_{C'}}(w) - \mathsf{Cpl}^{\alpha}_{\mathcal{E}_{C'}}(p) > 0$$

for $\widetilde{\mathcal{P}} = \mathcal{P} \setminus \{P, P_w\}$. Thus, Algorithm NP-Copeland returns "no."

To evaluate the computational complexity of Algorithm NP-Copeland, note that there are at most $|C|$ possibilities to choose w. The values $\Delta_c(w,p)$ can be computed in time $O(|V| \cdot |C|)$ which also suffices to find the nominated candidates and compute the resulting Copeland$^{\alpha}$ scores of p and w. Hence the total running time of Algorithm NP-Copeland is $O(|C|^2 \cdot |V|)$. ◻

Theorem 4.2. Necessary President *for Maximin is polynomial-time solvable.*

Proof. We propose an algorithm that solves Necessary President for Maximin in polynomial time; see Algorithm NP-Maximin for a pseudocode. Let $(\mathcal{E}, \mathcal{P}, p)$ be our input instance with election $\mathcal{E} = (C, V, \{\succ_v\}_{v\in V})$ and party P containing p.

Assume that there exists a reduced election $\mathcal{E}'$ where p is nominated but is not a winner. Then there exists a nominee w in $\mathcal{E}'$ whose Maximin score exceeds the Maximin score of p. Let P_w be the party containing w. Algorithm NP-Maximin first guesses candidate w as well as a candidate $\hat{c} \in C \setminus (P \cup P_w)$ that determines the Maximin score of p in $\mathcal{E}'$, i.e., for which $\mathsf{MM}_{\mathcal{E}'}(p) = N(p, \hat{c})$. Let $\hat{s} = N(p, \hat{c})$ and $\hat{P}$ be the party containing $\hat{c}$.[4]

First, the algorithm checks whether its guesses are valid in the sense that hold $N(w, \hat{c}) > \hat{s}$ and $N(w, p) > \hat{s}$ (conditions necessary for $\mathsf{MM}_{\mathcal{E}}(w) > \hat{s}$). Next, the algorithm searches for a suitable nominee $c_{\widetilde{P}}$ for each party $\widetilde{P} \in \mathcal{P} \setminus \{P, P_w, \hat{P}\}$ that satisfies $N(w, c_{\widetilde{P}}) > \hat{s}$. If such a candidate is found for each party other than P, P_w, and $\hat{P}$, then NP-Maximin returns "no." If all guesses are exhausted but the algorithm has not output "no", then it returns "yes."

Algorithm NP-Maximin Solving Necessary President for Maximin.

Input: An instance $(\mathcal{E}, \mathcal{P}, p)$ of Necessary President with candidate set C.

```
1: for all w ∈ C \ P do
2:     Let P_w be the party in 𝒫 containing w.
3:     for all ĉ ∈ C \ (P ∪ P_w) do
4:         Let ŝ = N(p, ĉ) and let P̂ be the party containing ĉ.
5:         if N(w, ĉ) > ŝ and N(w, p) > ŝ then
6:             if ∀P̃ ∈ 𝒫 \ {P, P_w, P̂} ∃c ∈ P̃ : N(w, c) > ŝ then
7:                 return "no".
8: return "yes".
```

To show the correctness of NP-Maximin, observe first that assuming correct guesses, the conditions checked on line 5 must hold for p, w, and $\hat{c}$, because by our definitions we have

$$\hat{s} = N(p, \hat{c}) = \mathsf{MM}_{\mathcal{E}'}(p) < \mathsf{MM}_{\mathcal{E}'}(w) \leq \min\{N(w,p), N(w,\hat{c})\}.$$

[4]Notice that the value $N_{\mathcal{E}'}(c, c')$ for any pair of candidates c, c' is the same irrespective of other nominations and so also of the reduced election, therefore in the rest of this proof we shall omit the index $\mathcal{E}'$.

Similarly, for each nominee c in $\mathcal{E}'$ other than these three candidates, we know $\hat{s} = \mathsf{MM}_{\mathcal{E}'}(p) < \mathsf{MM}_{\mathcal{E}'}(w) \leq N(w, c)$. Therefore, the algorithm will find on line 6 that there exists a candidate c satisfying the requirement $N(w, c) > \hat{s}$ in each party not containing p, w, or $\hat{c}$. Hence, NP-Maximin will return "no" on line 7.

For the other direction, assume that NP-Maximin returns "no" in some iteration. Consider the reduced election obtained where P, P_w, and $\hat{P}$ nominate the candidates p, w, and $\hat{c}$, respectively, guessed in this iteration, while each remaining party $\widetilde{P}$ nominates a candidate $c_{\widetilde{P}}$ that satisfies $N(w, c_{\widetilde{P}}) > \hat{s}$. Note that such candidates exist because the algorithm found the condition on line 6 to hold in the iteration when it returned "no." Hence, this method indeed yields a reduced election $\mathcal{E}_{C'}$ over some candidate set C'. Clearly, our assumptions on the nominees ensure $\mathsf{MM}_{\mathcal{E}_{C'}}(w) = \min_{c\in C'\setminus\{w\}} N(w, c) > \hat{s} = N(p, \hat{c}) \geq \mathsf{MM}_{\mathcal{E}_{C'}}(p)$, and therefore p is not a winner in $\mathcal{E}_{C'}$.

To estimate the computational complexity of the algorithm we first note that there are at most $|C|^2$ possible guesses for candidates w and $\hat{c}$. To compute the scores $N_{\mathcal{E}}(c, c')$ for each candidate pair c and c' in C requires $O(|C|^2 \cdot |V|)$ steps. To check the condition on line 6 for some fixed guess takes $O(|C|)$ time. Hence, the total running time of NP-Maximin is $O(|C|^2(|C| + |V|))$. ◻

Finally, we show that, unlike for Copeland$^{\alpha}$ and Maximin, Necessary President is computationally hard for Ranked Pairs, even for a constant number of voters. The reduction showing coNP-hardness in Theorem 4.3 is from 3-SAT, while W[1]-hardness is obtained by a reduction from the Multicolored Clique [28] problem.

Theorem 4.3 (★). *Necessary President for Ranked Pairs is* coNP*-complete even if the maximum party size is* $s = 2$ *and the number of voters is* $|V| = 12$*, and it is* W[1]*-hard with respect to parameter* t *denoting the number of parties, even if* $|V| = 20$.

5 CONCLUSIONS AND OUTLOOK

We explored the computational complexity of the Necessary President problem for several popular voting rules; together with previous results in [4], [31], and [30], our study offers a detailed picture of the computational tractability of problems faced by parties in the candidate nomination process preceding an election.

A possible direction for future research is to extend the existing tractability results for Necessary President for Plurality under single-peaked [12] or single-crossing [24] preferences to different voting rules or other structured domains.

As an interesting new topic, we propose to study candidate nomination problems for multiwinner elections. Suppose that the goal of the election is to choose a committee consisting of k members. In this case, it is natural to assume that a party may nominate more than one candidate. What will its optimal strategy be if it wants to have at least one of its nominees in the committee or if it wishes to maximize the number of its nominees in the committee?

ACKNOWLEDGMENTS

Ildikó Schlotter is supported by the Hungarian Academy of Sciences under its Momentum Programme (LP2021-2) and its János Bolyai Research Scholarship. Katarína Cechlárová is supported by VEGA 1/0585/24 and APVV-21-0369.

REFERENCES

[1] Piotr Berman, Marek Karpinski, and Alexander D. Scott. 2003. *Approximation Hardness of Short Symmetric Instances of MAX-3SAT.* Technical Report TR03-049. Electronic Colloquium on Computational Complexity.

[2] Markus Brill and Vincent Conitzer. 2015. Strategic voting and strategic candidacy. In *Proc. of the 29th AAAI Conference on Artificial Intelligence (AAAI 2015).* AAAI Press, 819–826. https://dl.acm.org/doi/10.5555/2887007.2887121

[3] Katarína Cechlárová and Ildikó Schlotter. 2026. Necessary President in Elections with Parties. *CoRR* arXiv:2602.10601 [cs.GT] (2026). https://doi.org/10.48550/arXiv.2602.10601

[4] Katarína Cechlárová, Julien Lesca, Diana Trellová, Martina Hančová, and Jozef Hanč. 2023. Hardness of candidate nomination. *Autonomous Agents and Multi-Agent Systems* 37 (2023), 37. https://doi.org/10.1007/s10458-023-09622-9

[5] Jiehua Chen, Piotr Faliszewski, Rolf Niedermeier, and Nimrod Talmon. 2017. Elections with few voters: Candidate control can be easy. *Journal of Artificial Intelligence Research* 60 (2017), 937–1002. https://doi.org/10.1613/jair.5515

[6] Marek Cygan, Fedor V Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. 2015. *Parameterized algorithms.* Springer.

[7] Argyrios Deligkas, Eduard Eiben, and Tiger-Lily Goldsmith. 2022. Parameterized Complexity of Hotelling-Downs with Party Nominees. In *Proc. of the 31st International Joint Conference on Artificial Intelligence (IJCAI 2022).* 244–250. https://doi.org/10.24963/ijcai.2022/35

[8] Reinhard Diestel. 2005. *Graph Theory.* Graduate Texts in Mathematics, Vol. 173. Springer-Verlag, Berlin, Heidelberg.

[9] Rodney G. Downey and Michael R. Fellows. 2013. *Fundamentals of Parameterized Complexity.* Springer, London.

[10] Bhaskar Dutta, Matthew O. Jackson, and Michel Le Breton. 2001. Strategic candidacy and voting procedures. *Econometrica* 69, 4 (2001), 1013–1037. https://doi.org/10.1111/1468-0262.00228

[11] Gábor Erdélyi, Marc Neveling, Christian Reger, Jörg Rothe, Yongjie Yang, and Roman Zorn. 2021. Towards completing the puzzle: complexity of control by replacing, adding, and deleting candidates or voters. *Autonomous Agents and Multi-Agent Systems* 35:41 (2021). https://doi.org/10.1007/s10458-021-09523-9

[12] Piotr Faliszewski, Laurent Gourvès, Jérôme Lang, Julien Lesca, and Jérôme Monnot. 2016. How hard is it for a party to nominate an election winner?. In *Proc. of the 25th International Joint Conference on Artificial Intelligence (IJCAI 2016).* AAAI Press, 257–263. https://doi.org/10.5555/3060621.3060658

[13] Piotr Faliszewski, Edith Hemaspaandra, Lane A Hemaspaandra, and Jörg Rothe. 2009. Llull and Copeland voting computationally resist bribery and constructive control. *Journal of Artificial Intelligence Research* 35 (2009), 275–341. https://doi.org/10.1613/jair.2697

[14] Piotr Faliszewski and Jörg Rothe. 2016. Control and bribery in voting. In *Handbook of Computational Social Choice*, Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia (Eds.). Cambridge University Press, 146–168.

[15] Jiong Guo, Yash Raj Shrestha, and Yongjie Yang. 2015. How credible is the prediction of a party-based election?. In *Proc. of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015).* IFAAMAS, 1431–1439. https://dl.acm.org/doi/abs/10.5555/2772879.2773335

[16] Paul Harrenstein, Grzegorz Lisowski, Ramanujan Sridharan, and Paolo Turrini. 2021. A Hotelling-Downs Framework for Party Nominees. In *Proc. of the 20th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2021).* 593–601. https://dl.acm.org/doi/10.5555/3463952.3464025

[17] Paul Harrenstein and Paolo Turrini. 2022. Computing Nash Equilibria for District-based Nominations. In *Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022).* IFAAMAS, 588–596. https://dl.acm.org/doi/abs/10.5555/3535850.3535917

[18] John E. Hopcroft and Richard M. Karp. 1973. An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs. *SIAM J. Comput.* 2, 4 (1973), 225–231. https://doi.org/10.1137/0202019

[19] John J. Bartholdi III, Craig A. Tovey, and Michael A. Trick. 1992. How hard is it to control an election? *Mathematical and Computer Modeling* 16, 8-9 (1992), 27–40. https://doi.org/10.1016/0895-7177(92)90085-Y

[20] TVN24 News in English. 2020. Civic Coalition candidate Małgorzata Kidawa-Błońska quits presidential run. https://tvn24.pl/tvn24-news-in-english/civic-coalition-candidate-malgorzata-kidawa-blonska-quits-presidential-run-st4584889. Accessed: 2025-09-30.

[21] Jérôme Lang, Vangelis Markakis, Nicolas Maudet, Svetlana Obraztsova, Maria Polukarov, and Zinovi Rabinovich. 2019. *Stategic Candidacy with Keen Candidates.* Technical Report hal-02294985. HAL. https://hal.science/hal-02294985v1

[22] Jérôme Lang, Nicolas Maudet, Maria Polukarov, and Alice Cohen-Hadria. 2025. Strategic Candidacy Equilibria for Common Voting Rules. *Theory of Computing Systems* 69 (2025), 23. https://doi.org/10.1007/s00224-025-10220-3

[23] Grzegorz Lisowski. 2022. Strategic Nominee Selection in Tournament Solutions. In *Proc. of the 19th European Conference on Multi-Agent Systems (EUMAS 2022).* 239–256. https://doi.org/10.1007/978-3-031-20614-6_14

[24] Neeldhara Misra. 2019. On the parameterized complexity of party nominations. In *Proc. of the 6th International Conference on Algorithmic Decision Theory (ADT 2019) (Lecture Notes in Computer Science, Vol. 11834).* Springer, 112–125. https://doi.org/10.1007/978-3-030-31489-7_8

[25] Le Monde. 2023. Turkey's opposition picks Kemal Kilicdaroglu to take on Erdogan in May election. https://www.lemonde.fr/en/elections/article/2023/03/07/turkey-s-opposition-picks-candidate-to-take-on-erdogan-in-may-election_6018420_84.html. Accessed: 2025-09-30.

[26] Svetlana Obraztsova, Maria Polukarov, Edith Elkind, and Zinovi Rabinovich. 2015. Strategic Candidacy Games with Lazy Candidates. In *Proc. of the 24th International Joint Conference on Artificial Intelligence (IJCAI 2015).* 610–616. https://dl.acm.org/doi/10.5555/2832249.2832334

[27] Tomasz Perek, Piotr Faliszewski, Maria Silvia Pini, and Francesca Rossi. 2013. The Complexity of Losing Voters. In *Proc. of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2013).* 407–414. https://dl.acm.org/doi/abs/10.5555/2484920.2484986

[28] Krzysztof Pietrzak. 2003. On the parameterized complexity of the fixed alphabet shortest common supersequence and longest common subsequence problems. *J. Comput. System Sci.* 67, 4 (2003), 757–771. https://doi.org/10.1016/S0022-0000(03)00078-3

[29] Maria Polukarov, Svetlana Obraztsova, Zinovi Rabinovich, Alexander Kruglyi, and Nicholas R. Jennings. 2015. Convergence to Equilibria in Strategic Candidacy. In *Proc. of the 24th International Joint Conference on Artificial Intelligence (IJCAI 2015).* 624–630. https://dl.acm.org/doi/10.5555/2832249.2832336

[30] Ildikó Schlotter and Katarína Cechlárová. 2025. Candidate nomination for Condorcet-consistent voting rules. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025).* 1858–1866. https://dl.acm.org/doi/10.5555/3709347.3743822

[31] Ildikó Schlotter, Katarína Cechlárová, and Diana Trellová. 2024. Parameterized complexity of candidate nomination for elections based on positional scoring rules. *Autonomous Agents and Multi-Agent Systems* 38 (2024), 28. https://doi.org/10.1007/s10458-024-09658-5

[32] Wikipedia. 2024. 2024 United States presidential election. https://en.wikipedia.org/wiki/2024_United_States_presidential_election. Accessed: 2025-09-30.

[33] Wikipedia. 2025. 2022 Hungarian parliamentary election. https://en.wikipedia.org/wiki/2022_Hungarian_parliamentary_election. Accessed: 2025-09-30.