On Problem Kernels for Possible Winner Determination Under the k-Approval Protocol

Nadja Betzler*

Institut für Informatik, Friedrich-Schiller-Universität Jena
Ernst-Abbe-Platz 2, D-07743 Jena, Germany
nadja.betzler@uni-jena.de

Abstract. The POSSIBLE WINNER problem asks whether some distinguished candidate may become the winner of an election when the given incomplete votes (partial orders) are extended into complete ones (linear orders) in a favorable way. Under the k-approval protocol, for every voter, the best k candidates of his or her preference order get one point. A candidate with maximum total number of points wins. The POSSIBLE WINNER problem for k-approval is NP-complete even if there are only two votes (and k is part of the input). In addition, it is NP-complete for every fixed $k \in \{2, \ldots, m-2\}$ with m denoting the number of candidates if the number of votes is unbounded. We investigate the parameterized complexity with respect to the combined parameter k and "number of incomplete votes" t, and with respect to the combined parameter $k' := m - k$ and t. For both cases, we use kernelization to show fixed-parameter tractability. However, we show that whereas there is a polynomial-size problem kernel with respect to (t, k'), it is very unlikely that there is a polynomial-size kernel for (t, k). We provide additional fixed-parameter algorithms for some special cases.

1 Introduction

Voting situations arise in political elections, multi-agent systems, human resource departments, etc. This includes scenarios in which one is interested in finding a small group of winners (or losers), such as awarding a small number of grants, picking out a limited number of students for a graduate school, or voting for a committee with few members. Such situations are naturally reflected by a variant of approval voting, the *k-approval* voting system, where every voter gives one point to each of the k alternatives/candidates which he or she likes best and the candidates having the most points in total win. On the one side, k-approval extends *plurality* where a voter gives one point to one candidate, that is $k = 1$, and, on the other side, it extends *veto* where a voter gives one point to all but one candidate, that is, $k' = 1$ for $k' := m - k$ and m candidates.

At a certain point in the decision making process one might face the situation that the voters have made up their minds "partially". For example, for the decision about the Nobel prize for peace in 2009, a committee member might have already known that he (or she) prefers Obama and Bono to Berlusconi, but might have not decided on the order of Obama and Bono yet. This immediately leads to the question whether, given a

* Supported by the DFG, research project PAWS, NI 369/10.

set of "partial preferences", a certain candidate may still win. The formalization of this question leads to the POSSIBLE WINNER problem.

The POSSIBLE WINNER problem has been introduced by Konczak and Lang [15] and since then its computational complexity has been studied for several voting systems [2, 3, 5, 17, 18]. Even for the comparatively simple k-approval voting, it turned out that POSSIBLE WINNER is NP-complete except for the special cases of plurality and veto [3], that is, for any k greater than one and smaller than the number of candidates minus one. A multivariate complexity study showed that it is NP-complete if there are only two voters when k is part of the input but fixed-parameter tractable with respect to the "number of candidates" [5]. In contrast, for the approval voting variant where each voter can assign a point to *up to* k candidates, it can easily be seen that POSSIBLE WINNER can be solved in polynomial-time. A prominent special case of POSSIBLE WINNER is the MANIPULATION problem, where the input consists of a set of linear orders and a set of completely unspecified votes. For k-approval, it is easy to see that MANIPULATION is solvable in polynomial time for unweighted votes but for weighted votes it is NP-complete for all fixed $k \neq 1$ [14].

The above described hardness results motivate a multivariate analysis with respect to the combined parameter "number of voters" *and* "number of candidates to which a voter gives one/zero points" for k-approval. Can we efficiently solve POSSIBLE WINNER in the case that these parameters are both small? Directly related questions are whether we can ignore or delete candidates which are not relevant for the decision process and how to identify such candidates. In this context, parameterized algorithmics [10, 16] provides the concept of kernelization by means of polynomial-time data reduction rules that "preprocess" an instance such that the size of the "reduced" instance only depends on the parameters [6, 13].

In this work, we use kernelization to show the fixed-parameter tractability of POSSIBLE WINNER for k-approval in two "symmetric" scenarios. First, we consider the combined parameter "number of incomplete votes" t *and* "number of candidates to which every voter gives zero points" $k' := m - k$ for m candidates (directly extending the veto voting system with $k' = 1$). Making use of a simple observation we show that POSSIBLE WINNER admits a polynomial-size problem kernel with respect to (t, k') and provide two algorithms: one with exponential running time factor $2^{O(k')}$ in case of constant t and one with exponential running time factor $2^{O(t)}$ in case of constant k'. Second, we consider the combined parameter t and k, where k denotes the "number of candidates to which a voter gives a point". We observe that here one cannot argue symmetrically to the first scenario. Using other arguments, we give a superexponential-size problem kernel showing the fixed-parameter tractability of POSSIBLE WINNER with respect to (t, k). For the special case of 2-approval, we give an improved polynomial-size kernel with $O(t^2)$ candidates. Using a methodology due to Bodlaender et al. [7], our main technical result shows that POSSIBLE WINNER is very unlikely to admit a polynomial-size problem kernel with respect to (t, k).

Preliminaries. A *linear vote* is a transitive, antisymmetric, and total relation on a set C of candidates and *partial vote* a transitive and antisymmetric relation on a set C of candidates. We use $>$ to denote the relation between candidates in a linear vote and $\succ$ to denote the relation between candidates in a partial vote. We often specify a subset $D \subseteq$

C of candidates instead of single candidates in a partial vote; for a candidate $e \in C \setminus D$ and $D = \{d_1, \ldots, d_s\}$, the meaning of "$e \succ D$" is "$\{e \succ d_1, e \succ d_2, \ldots, e \succ d_s\}$". A linear vote v^l *extends* a partial vote v^p if $v^p \subseteq v^l$, that is, for every $i, j \leq m$, from $c_i \succ c_j$ in v^p it follows that $c_i > \cdots > c_j$ in v^l. An extension E of a set of partial votes $V^p = \{v^p_1, \ldots, v^p_n\}$ is a mapping from V^p to a set of linear votes $V^l := \{v^l_1, \ldots, v^l_n\}$ such that v^l_i extends v^p_i for every i. Given a set of partial votes V^p on C, a candidate $c \in C$ is a *possible winner* if there exists a *winning extension* E, that is, c wins in E with respect to a considered voting system. For any voting system R, the underlying decision problem is defined as follows.

POSSIBLE WINNER
Given: A set of candidates C, a set of partial votes V on C, and a distinguished candidate $c \in C$.
Question: Is there an extension E of V such that c wins with respect to R in E?

We focus on the voting system k-*approval* where, given a set V of linear votes on a set C of candidates, the first k candidates within a vote get one point and the remaining candidates get zero points. For every candidate $c' \in C$, one sums up the points over all votes from V to obtain its *score* $s(c')$ and the candidates with maximum score win. We call the first k positions of a vote *one-positions* and the remaining positions *zero-positions*. All results are given for the *unique winner* case, that is, looking for a single candidate with maximum score, but can be adapted easily to hold for the "co-winner" case where several candidates may get the maximum score and all of them win.

A parameterized problem L is a subset of $\Sigma^* \times \Sigma^*$ for some finite alphabet Σ [10, 16]. An instance of a parameterized problem consists of (x, p) where p is called the parameter. We mainly consider "combined" parameters which are tuples of positive integers. A parameterized problem is *fixed-parameter tractable* if it can be solved in time $f(|p|) \cdot \mathrm{poly}(|x|)$ for a computable function f. A kernelization algorithm consists of a set of *(data) reduction rules* working as follows [6, 13, 16]. Given an instance $(x, p) \in \Sigma^* \times \Sigma^*$, they output in time polynomial in $|x| + |p|$ an instance $(x', p') \in \Sigma^* \times \Sigma^*$ such that the following two conditions hold. First, (x, p) is a yes-instance if and only if (x', p') is a yes-instance (termed *soundness*). Second, $|x'| + |p'| \leq g(|p|)$ where g is a computable function. If g is a polynomial function, then we say that the parameterized problem admits a *polynomial kernel*.

Some of the reduction rules given in this work will not directly decrease the instance size by removing candidates or votes but instead only decrease the number of possible extensions of a vote, for example, by "fixing" candidates. To *fix* a candidate at a certain position means to specify its relation to all other candidates. Clearly, a candidate may not be fixed at every position in a specific partial vote. To take this into account, an important concept is the notation of *shifting* a candidate. More precisely, we say a candidate c' can shift a candidate c'' to the left (right) in a partial vote v if $c'' \succ c'$ ($c' \succ c''$) in v, that is, setting c' to a one-position (zero-position) implies setting c'' to a one-position (zero-position) as well. For every candidate $c' \in C$ and a partial vote $v \in V$, let $L(v, c') := \{c'' \in C \mid c'' \succ c' \text{ in } v\}$ and $R(v, c') := \{c'' \in C \mid c' \succ c'' \text{ in } v\}$. Then, fixing a candidate $c' \in C$ *as good as possible* means to add $L(v, c') \succ c' \succ C \setminus (L(v, c') \cup \{c'\})$ to v. Analogously, fixing a candidate *as bad as possible* is realized

by adding $C \setminus (R(v,c') \cup \{c'\}) \succ c' \succ R(v,c')$ to v. If a candidate $c' \in C$ is fixed in all partial votes, this implies that also its score $s(c')$ is fixed.

The votes of an input instance of POSSIBLE WINNER can be partitioned into a (possibly empty) set of linear votes, called V^l, and a set of proper (non-linear) partial votes, called V^p. We state all our results for the parameter $t := |V^p|$. All positive results also hold for the parameter number of total votes $n := |V^l| + |V^p|$. Due to the space restrictions, several (parts of) proofs are deferred to a full version of this work.

2 Fixed number of zero-positions

For $(m-k')$-approval with $k' < m$, k' denotes the number of zero-positions. We give a polynomial kernel with respect to (t, k') for POSSIBLE WINNER where t is the number of partial votes. In addition, we provide two parameterized algorithms for special cases.

2.1 Problem kernel

Consider a POSSIBLE WINNER instance with candidate set C, vote set $V = V^l \cup V^p$, and distinguished candidate $c \in C$ for $(m-k')$-approval. We start with a simple reduction rule that is a crucial first step for all kernelization results in this work.

Rule 1 *For every vote $v_i \in V^p$, if $|L(v_i, c)| < m - k'$, fix c as good as possible in v_i.*

The soundness and polynomial-time running time of Rule 1 is easy to verify. The condition $|L(v_i,c)| < m-k'$ is crucial since otherwise c might shift a candidate c' to a one-position whereas c is assigned to a zero position and this could cause c' to beat c. After applying Rule 1, the score of c is fixed at the maximum possible value since it makes one point in all votes in which this is possible. Now, for every candidate $c' \in C \setminus \{c\}$, by counting the points that c' makes within the linear votes V^l, compute the number of zero positions that c' must assume within the partial votes V^p such that it is beaten by c. Let this number be $z(c')$ and $Z_+ := \{c' \in C \setminus \{c\} \mid z(c') > 0\}$. Since there are only tk' zero positions in V^p, one can observe the following.

Observation 1 *In a yes-instance, $\sum_{c' \in C \setminus \{c\}} z(c') \leq tk'$ and $|Z_+| \leq tk'$.*

Observation 1 provides a simple upper bound for the number of candidates in Z_+. By formulating a data reduction rule that bounds the number of remaining candidates and replacing the linear votes V^l by a bounded number of "equivalent votes" we can show the following theorem. The basic idea is that since a remaining candidate from $C \setminus (Z_+ \cup \{c\})$ can be set arbitrarily in every vote without beating c, it is possible to replace the set of all remaining candidates by tk'^2 "representative candidates".

Theorem 1. *For $(m-k')$-approval,* POSSIBLE WINNER *with t partial votes admits a polynomial kernel with at most $tk'^2 + tk' + 1$ candidates.*

Initialization:
For every $D' \in \mathcal{D} \setminus \{(d_1, \ldots, d_p)\}$, set $T(0, D') = 0$.
Set $T(0, (d_1, \ldots, d_p)) = 1$.

Update:
For $0 \leq i \leq t-1$,
 for every $D' = (d'_1, \ldots, d'_p) \in \mathcal{D}$,
  $T(i+1, D') = 1$ if there are two candidates z_g, z_h that can take the zero-positions in v_{i+1}
  and $T(i, D'') = 1$ with $D'' := \{d''_1, \ldots, d''_p\}$ and
  $d''_j = d'_j$ for $j \in \{1, \ldots, q\} \setminus \{g, h\}$, $d''_g \leq d'_g + 1$, and $d''_h \leq d'_h + 1$.

Output:
"yes" if $T(t, (0, \ldots, 0)) = 1$, "no" otherwise

Fig. 1. Dynamic programming algorithm for $(m-2)$-approval.

2.2 Parameterized algorithms

We give algorithms running in $2^{O(p)} \cdot \mathrm{poly}(n, m)$ time with p denoting either k' or t where the other parameter is of constant value. Note that the kernelization from the previous subsection does not imply such running times.

Constant number of partial votes. For two partial votes, there can be at most $2k'$ candidates that must take a zero-position in a yes-instance (see Observation 1). Branching into the two possibilities of taking the zero-position in the first or in the second vote for every such candidate, results in a search tree of size $2^{2k'} = 4^{k'}$. For every "leaf" of the search tree it is easy to check if there is a corresponding extension. Using similar arguments, one arrives at the following.

Proposition 1 *For a constant number t of partial votes,* POSSIBLE WINNER *for $(m - k')$-approval can be solved in $2^{t^2 k'} \cdot \mathrm{poly}(n, m)$ time.*

Constant number of zero-positions. For constant k' the existence of an algorithm with running time $2^{O(t)} \cdot \mathrm{poly}(n, m)$ seems to be less obvious than for the case of constant t. We start by giving a dynamic programming algorithm for $(m-2)$-approval. Employing an idea used in [4, Lemma 2], we show that it runs in $4^t \cdot \mathrm{poly}(n, m)$ time and space.

As in the previous subsection, fix c according to Rule 1 such that it makes the maximum possible score and let $Z_+ := \{z_1, \ldots, z_p\}$ denote the set of candidates that take at least one zero-position in a winning extension. Let $d_1, \ldots, d_p$ denote the corresponding number of zero-positions that must be assumed and let $\mathcal{D} := \{(d'_1, \ldots, d'_p) \mid 0 \leq d'_j \leq d_j \text{ for } 0 \leq j \leq p\}$. Then, the dynamic programming table T is defined by $T(i, D')$ for $1 \leq i \leq t$ and $D' = (d'_1, \ldots, d'_p) \in \mathcal{D}$. Herein, $T(i, D') = 1$ if the partial votes from $\{v_1, \ldots, v_i\}$ can be extended such that candidate z_j takes at least $d_j - d'_j$ zero-positions for $1 \leq j \leq p$; otherwise $T(i, D') = 0$. Intuitively, d'_j stands for the number of zero-positions which z_j must still take in the remaining votes $\{v_{i+1}, \ldots, v_t\}$. Clearly, if $T(t, (0, \ldots, 0)) = 1$ for an instance, then it is a yes-instance. The dynamic programming algorithm is given in Figure 1. By further extending it to work for any constant k' we can show the following.

Theorem 2. *For $(m-2)$-approval with t partial votes,* POSSIBLE WINNER *can be solved in $4^t \cdot \mathrm{poly}(n,m)$ time and $O(t \cdot 4^t)$ space. For $(m-k')$-approval with t partial votes,* POSSIBLE WINNER *can be solved in $2^{O(t)} \cdot \mathrm{poly}(n,m)$ time for constant k'.*

3 Fixed number of one-positions

We study POSSIBLE WINNER for k-approval with respect to the combined parameter k and number t of partial votes. The problem can be considered as "filling" tk one-positions such that no candidate beats c. In the previous section, we exploited that the number of candidates that must take a zero-position is already bounded by the combined parameter t and "number of zero-positions" in a yes-instance (Observation 1). Here, we cannot argue analogously: Our combined parameter (t,k) only bounds the number of one-positions but there can be an unbounded number of candidates that may take a one-position in different winning extensions of the partial votes. Hence, we argue that if there are too many candidates that can take a one-position, then there must be several choices that lead to a valid extension. We show that it is sufficient to keep a set of "representative candidates" that can take the required one-positions if and only if this is possible for the whole set of candidates. This results in a problem kernel of super-exponential size showing fixed-parameter tractability with respect to (t,k). We complement this result by showing that it is very unlikely that there is a kernel of polynomial size. In addition, we give a polynomial kernel with $O(t^2)$ candidates for 2-approval.

3.1 Problem kernels

We first describe a kernelization approach for POSSIBLE WINNER for k-approval in general and then show how to obtain a better bound on the kernel size for 2-approval.

Problem kernel for k-approval. In order to describe more complicated reduction rules, we assume that a considered instance is exhaustively reduced with respect to some simple rules. To this end, we fix the distinguished candidate c as good as possible by Rule 1 (using that $m-k'=k$). Afterwards, we apply a simple reduction rule to get rid of "irrelevant" candidates and check whether an instance is a trivial no-instance:

Rule 2 *First, for every candidate $c' \in C \setminus \{c\}$, if making one point in the partial votes causes c' not to be beaten by c, then fix c' as bad as possible in every vote. Second, compute the set D of candidates that can be deleted: For every candidate $c' \in C \setminus \{c\}$ with $|L(v,c')| > k$ for all $v \in V^p$, if the score $s(c')$ is at least $s(c)$, then output "no solution", otherwise add c' to D. Delete D and replace V^l by an equivalent set.*

The soundness of Rule 2 is easy to see: Every candidate fixed by the first part cannot be assigned to a one-position in any winning extension. For the second part, every winning extension of an unreduced instance can easily be transformed into a winning extension for the reduced one by deleting the candidates specified by Rule 2 and *vice versa*. A set of equivalent linear votes can be found according to [3, Lemma 1][1].

[1] Herein, it might be necessary to add one new candidate. However, this will not affect the following analysis and will be discussed in more detail in the full version of this work.

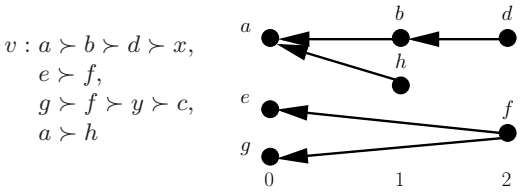

Fig. 2. Example for 3-approval: Partial vote v (left-hand side) and corresponding digraph with levels 0, 1, and 2. Arcs following by transitivity are omitted. Note that x, y, and c do not appear in the digraph since they are irrelevant for v.

In the following, we assume that Rule 2 has been applied, that is, all remaining candidates can make at least one point in an extension without beating c. To state further reduction rules, a partial vote v is represented as a digraph with vertex set $\{c' \mid c' \in C \setminus \{c\}$ and $|L(v, c')| < k\}$. All other candidates are considered as "irrelevant" for this vote since they cannot take a one-position. The vertices are organized into k levels. For $0 \leq j \leq k-1$, let $L_j(v) := \{c' \mid c' \in C \setminus \{c\}$ and $|L(v, c')| = j\}$ containing all candidates that shift exactly j candidates to a one-position if they are assigned to the best possible position. There is a directed arc from c' to c'' if and only if $c'' \in L(v, c')$. Figure 2 displays an example for the representation of a partial vote for 3-approval.

In general, the number of candidates per level is unbounded. However, for some cases it is easy to see that one can "delete" all but some representative candidates. The following reduction rule provides such an example using the fact that in any vote a candidate from the first level can be set to an arbitrary one-position without shifting any other candidate.

Rule 3 *For $v \in V^p$ with $|L_0(v)| \geq tk$, consider any subset $L' \subseteq L_0(v)$ with $|L'| = tk$. Add $L' \succ C \setminus L'$ to v.*

To see the soundness of Rule 3 consider a winning extension E for a non-reduced instance and a vote $v \in V^p$ with $|L_0(v)| \geq tk$. Since there are tk one-positions in the partial votes, there must be at least k candidates from L' not having assumed a one-position within the other $t-1$ votes. Setting these k candidates to the one-positions in v leads to a winning extension of the reduced instance. The other direction is obvious.

If Rule 3 applies to all partial votes, then in a reduced instance at most t^2k candidates are not fixed at zero-positions in V^p and the remaining candidates can be deleted by Rule 2. Hence, we consider the situation that there is a partial vote v with $|L_0(v)| < tk$. Then, we cannot ignore the candidates from the other levels but replace them by a bounded number of representatives. We first discuss how to find a set of representatives for 2-approval and then extend the underlying idea to work for general k.

For 2-approval, for a vote v with $|L_0(v)| < 2t$, it remains to bound the size of $L_1(v)$. This is achieved by the following reduction rule: Fix all but $2t$ in-neighbors of every candidate from $L_0(v)$ at zero-positions. To see the soundness, we show, given a winning extension E for the non-reduced instance, how to obtain a winning extension E' for v after the reduction (the other direction is obvious). Clearly, in $E(v)$ the first position must be assigned to a candidate c' from $L_0(v)$ and c' can also be assigned to

the first position in $E'(v)$. If there is another candidate from $L_0(v)$ that takes the second position in $E(v)$, one can do the same in $E'(v)$. Otherwise, distinguish two cases. First, c' has less than $2t$ in-neighbors, then the reduction rule has not fixed any candidate that shifts c' to the first position and thus v can be extended in the same way as in E. Second, c' has at least $2t$ in-neighbors. Since there are only $2t$ one-positions and $2t$ non-fixed in-neighbors, the second position of v can be assigned to a candidate that does not take a one-position in any other vote of E.

Altogether, for 2-approval, one ends up with up to $4t^2$ non-fixed candidates per vote and hence with $O(t^3)$ non-reduced candidates in total. For general k, extend this approach iteratively by bounding the number of candidates for every level:

Rule 4 *Consider a partial vote $v \in V^p$ with $|L_0(v)| < tk$. Start with $i = 1$ and repeat until $i = k$.*
- For every candidate $c' \in L_i(v)$, if there are more than tk candidates in $L_i(v)$ which have the same neighborhood as c' in $L_0(v) \cup L_1(v) \cup \cdots \cup L_{i-1}(v)$, fix all but tk of them as bad as possible.
- Set $i := i + 1$.

Using Rule 4 one can show the following.

Theorem 3. *For k-approval,* POSSIBLE WINNER *admits a problem kernel with size bounded by a computable function in k and the number of partial votes t.*

Improved problem kernel for 2-approval. As discussed above, the kernelization as stated for k-approval in general leads to a polynomial kernel with $O(t^3)$ candidates for 2-approval. To give a kernel with $O(t^2)$ candidates, we use some properties of bipartite graphs. For a bipartite graph $(G \cup H, E)$ with vertex set $G \cup H$ and edge set $E \subseteq \{\{g, h\} \mid g \in G \text{ and } h \in H\}$, a *matching* denotes a subset $M \subseteq E$ such that for all $e, e' \in M$, $e \cap e' = \emptyset$. A vertex contained in e for an $e \in M$ is called *matching vertex* and, for $\{g, h\} \in M$, g and h are *matching neighbors*. A *maximum matching* is a matching with maximum cardinality. The *open neighborhood* of a vertex $g \in G$ is denoted by $N(g) := \{h \mid \{g, h\} \in E\}$ and, for $G' \subseteq G$, $N(G') := \bigcup_{g \in G'} N(g)$.

Lemma 1. *For a bipartite graph $(G \cup H, E)$ with maximum matching M, there is a partition of G into $G_1 \uplus G_2$, such that the following holds. First, all neighbors of G_1 are part of M. Second, every vertex from G_2 has a matching neighbor outside $N(G_1)$.*

Now, we employ Lemma 1 to design a reduction rule. Note that similar arguments are used in several works, see [8, 16]. In the following, we assume that Rule 1 and Rule 2 have been applied. We define a bipartite graph $(G \cup H, E)$ as follows. For a partial profile with partial votes V^p and candidate set C, let $V' := \{v' \in V^p \mid |L_0(v')| < 2t\}$. For every $v'_i \in V'$, for $1 \leq j \leq |L_0(v'_i)|$, add a vertex g_i^j to G. Intuitively, for every candidate that can take a first position in v'_i there is a corresponding vertex in G. If a candidate can take the first position in several votes, then there are several vertices corresponding to this candidate. The vertex set H contains one vertex for every candidate from $(\bigcup_{v' \in V'} L_1(v')) \setminus (\bigcup_{v' \in V'} L_0(v'))$. There is an edge between $g_i^j \in G$ and $h \in H$ if setting the candidate corresponding to h to the second position in v'_i shifts the candidate corresponding to g_i^j to the first position. Now, we can state the following.

Rule 5 *Compute a maximum matching M in $(G \cup H, E)$. Fix every candidate corresponding to a non-matched vertex in H as bad as possible in every vote from V'.*

Lemma 2. *Rule 5 is sound and can be carried out in $O(|E| \cdot |G \cup H| + |V| \cdot |C|)$ time.*

Proof. A winning extension for an instance reduced with respect to Rule 5 is also a winning extension for an unreduced instance. Now, we show the other direction. Given a winning extension E for an unreduced instance, we construct a winning extension E_r for a reduced instance. Since Rule 5 does not fix any candidate which can take the first position in at least one vote, the first positions in E_r can be assumed by the same candidates as in E. It remains to fix the second positions without beating c. For every vote v_i, let g_i^e denote the candidate that takes the first position in v_i in E. For the corresponding vertex g_i^e one can distinguish two cases: First, $g_i^e \in G_1$. In this case, none of the neighbors of g_i^e have been fixed and, thus, the candidate which takes the second position in v_i in E can also take the second position E_r. Second, $g_i^e \in G_2$. In this case, set the candidate corresponding to the matching neighbor from g_i^e to the second position. Now, it is not to hard to see that c wins in E_r: The only candidates that possibly make more points in E_r than in E are the candidates corresponding to the matching neighbors of vertices from G_2. Due to the matching property, every such candidate makes at most one point in V'. By definition, G only contains vertices that can make at least one point and for all votes from $V^p \setminus V'$ one can easily find a winning extension which does not assign the "matching-candidates" to one-positions (see Rule 2). It follows that c also wins in the extension E_r. The claimed running time follows since a maximum bipartite matching can be found in $O(|E| \cdot |G \cup H|)$ time. □

Bounding the size of candidates in level 0 by Rule 3 and the (remaining) candidates in level 1 by Rule 5 one arrives at the following.

Theorem 4. *For 2-approval with t partial votes,* POSSIBLE WINNER *admits a polynomial kernel with less than $4t^2$ candidates.*

3.2 Kernel lower bound

We use a method introduced by Bodlaender et al. [7] and Fortnow and Santhanam [12] to show that, for k-approval, POSSIBLE WINNER cannot have a polynomial kernel with respect to (t, k). They provide a general scheme to show the non-existence of polynomial kernels under some reasonable assumptions from classical complexity theory.

Definition 1. *[7] A composition algorithm for a parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is an algorithm that receives as input a sequence $((x_1, p), \ldots, (x_q, p))$ with $(x_i, p) \in \Sigma^* \times \mathbb{N}$ for each $1 \leq i \leq q$, uses time polynomial in $\sum_{i=1}^{q} |x_i| + p$, and outputs $(y, p') \in \Sigma^* \times \mathbb{N}$ with (a) $(y, p') \in L \Leftrightarrow (x_i, p) \in L$ for some $1 \leq i \leq q$ and (b) p' is polynomial in p. A parameterized problem is compositional if there is a composition algorithm for it.*

Theorem 5. *[7, 12] Let L be a compositional parameterized problem whose unparameterized version is NP-complete. Then, unless $\mathrm{coNP} \subseteq \mathrm{NP} / \mathrm{poly}$, there is no polynomial kernel for L.*

Dom et al. [9] provide a framework to build composition algorithms by using "identifiers". One of the necessary conditions is the existence of an algorithm running in $2^{p^\gamma} \cdot \mathrm{poly}$ time for the considered parameter p and a fixed constant γ. Considering the combined parameter "number of ones" k and "number of partial votes" t for POSSIBLE WINNER under k-approval, there is no known algorithm running in $2^{(tk)^\gamma} \cdot \mathrm{poly}(|X|)$ time. Hence, we apply the following overall strategy (which might be also useful for other problems).

Overall strategy. We employ a proof by contradiction. Assume that there is a polynomial kernel with respect to (t, k). Then, since for POSSIBLE WINNER there is an obvious brute-force algorithm running in $m^{tk} \cdot \mathrm{poly}(n, m)$ time for m candidates and n votes, there must be an algorithm A with running time $\mathrm{poly}(t, k)^{tk} \cdot \mathrm{poly}(n, m) < 2^{(tk)^\gamma} \cdot \mathrm{poly}(n, m)$ for an appropriate constant γ. In the next paragraph, we use the existence of algorithm A to design a composition algorithm for the combined parameter (t, k). Since it is easy to verify that the unparameterized version[2] of POSSIBLE WINNER is NP-complete, it follows from Theorem 5 that there is no polynomial kernel with respect to (t, k), a contradiction unless $\mathrm{coNP} \subseteq \mathrm{NP} / \mathrm{poly}$. Altogether, it remains to give a composition algorithm.

Composition algorithm. Consider a sequence $((x_1, (t, k)), \ldots, (x_q, (t, k)))$ of q POSSIBLE WINNER instances for k-approval. To simplify the construction, we make two assumptions. First, we assume that there is no "obvious no-instance", that is, an instance in which a candidate c' beats c even if c' makes zero points in all of the partial votes. This does not constitute any restriction since such instances can be found and removed in time polynomial in $\sum_{i=1}^{q} |x_i|$. Second, we assume that for $x_j, 1 \leq j \leq q$, within the partial votes the distinguished candidate makes zero points in every extension. Since it follows from known constructions [3, 5] that the unparameterized version of the problem remains NP-complete for this case, this assumption leads to a non-existence result for this special case and thus also for the general case.

Now, we give the composition algorithm. If $q > 2^{(tk)^\gamma}$ for γ as given by algorithm A, the composition algorithm applies A to every instance. This can be done within the running time bound required by Definition 1 and, in the following, we can assume that the number of instance is at most $2^{(tk)^\gamma}$. This can be used to assign an "identifier" of sufficiently small size to every instance. Basically, the identifiers, which will be realized by specific sets of candidates, rely on the binary representation of the numbers from $\{1, \ldots, q\}$. The size of an identifier will be linear in $s := \lceil \log q \rceil$ which is polynomial in the combined parameter (t, k) since $q \leq 2^{(tk)^\gamma}$.

Compose the sequence of instances to one big instance $(X, (3s + 4, 2t))$ with $X = (C, V^l \cup V^p, c)$ as follows. For $1 \leq i \leq q$, let x_i be $(C_i, V_i^l \cup V_i^p, c_i)$. Then,

$$C := \biguplus_{1 \leq i \leq q} (C_i \setminus \{c_i\}) \uplus \{c\} \uplus D \uplus Z \uplus A \uplus B$$

with $D := \{d_0^0, \ldots, d_s^0\} \cup \{d_0^1, \ldots, d_s^1\}$, $Z := \bigcup_{1 \leq j \leq t} Z_j$ with $Z_j := \{z_{h,j}^0 \mid 0 \leq h \leq s\} \cup \{z_{h,j}^1 \mid 0 \leq h \leq s\}$, $A := \{a_1, \ldots, a_q\}$, and a set B with $|B| := 2s + 3 - k$. The candidates from D and Z will be used as identifiers for the different instances: Every

[2] See [7] for a formal definition.

$$
\begin{array}{lllll}
V_1^p : Z_{w,1} > \overline{D_w} > \overline{Z_{w,t}} & > a_w & > C \setminus (Z_{w,1} \cup \overline{D_w} \cup \overline{Z_{w,t}}) & & \\
\quad\ Z_{w,j} > \overline{D_w} > \overline{Z_{w,j-1}} & > a_w & > C \setminus (Z_{w,j} \cup \overline{D_w} \cup \overline{Z_{w,j-1}}) & & \text{for } 2 \leq j \leq t \\
V_2^p : B \quad > D_w > w_j & & > C \setminus (B \cup D_w \cup (C_w \setminus \{c_w\})) & & \text{for } 1 \leq j \leq t
\end{array}
$$

Fig. 3. Extension for X in which c wins. For a winning extension $E(x_w) = w'_1, \ldots, w'_t$ of x_w, let w_j denotes the linear order given by w'_j restricted to the candidates from $C_w \setminus \{c\}$.

instance x_i is uniquely identified by the binary code of the integer $i = b_0 \cdot 2^0 + b_1 \cdot 2^1 + \cdots + b_s \cdot 2^s$ with $b_h \in \{0, 1\}$. Then, a subset $D_i \subset D$ identifies x_i when $d_h^1 \in D_i$ if and only if $b_h = 1$ and $d_h^0 \in D_i$ if and only if $b_h = 0$. Let $\overline{D_i} := D \setminus D_i$. Similarly, for every $1 \leq j \leq t$, the set $Z_{i,j}$ denotes the candidates from Z_j that identify i, that is,

$$Z_{i,j} := \{z_{h,j}^0 \mid h \in \{0, \ldots, s\} \text{ and } b_h = 0\} \cup \{z_{h,j}^1 \mid h \in \{0, \ldots, s\} \text{ and } b_h = 1\}.$$

Let $\overline{Z_{i,j}} := Z_j \setminus Z_{i,j}$ denote the remaining candidates from Z_j.

The set of partial votes V^p consists of two subsets V_1^p and V_2^p, both containing t partial votes. The basic idea is that a winning extension of V_1^p "selects" an (arbitrary) instance x_i and there is a winning extension for x_i if and only if V_2^p can be extended such that c wins. The set V_1^p contains the vote

$$\{Z_{i,1} \cup \overline{D_i} \cup \overline{Z_{i,t}} \succ a_i \mid 1 \leq i \leq q\},\ D \cup Z \cup A \succ C \setminus (D \cup Z \cup A),$$

meaning that the vote contains the condition $Z_{i,1} \cup \overline{D_i} \cup \overline{Z_{i,t}} \succ a_i$ for every i. Furthermore, for every $j \in \{2, \ldots, t\}$, the set V_1^p contains the vote

$$\{Z_{i,j} \cup \overline{D_i} \cup \overline{Z_{i,j-1}} \succ a_i \mid 1 \leq i \leq q\},\ D \cup Z \cup A \succ C \setminus (D \cup Z \cup A).$$

The set V_2^p consists of the partial votes $v_1, \ldots, v_t$. Every vote $v_j \in V_2^p$ "composes" the votes v_i^j for $1 \leq i \leq q$ where v_i^j denotes the jth vote from instance x_i after deleting c_i. Then, for $1 \leq j \leq t$, the vote v_j is

$$B \succ (C \setminus B),\ \{v_i^j \mid 1 \leq i \leq q\},\ \{D_i \succ C_i \setminus \{c_i\} \mid 1 \leq i \leq q\},\ C \setminus (A \cup Z \cup \{c\}) \succ A \cup Z \cup \{c\}.$$

Using [3, Lemma 1], one can construct a set V_l of linear votes polynomial in $|C|$ and $|V^p|$ such that in every winning extension, within V^p,
(a) for $i \in \{1, \ldots, q\}$, the number of points a candidate $c' \in C_i \setminus \{c_i\}$ makes is at most the maximum number of points which c' makes in a winning extension within the partial votes from x_i,
(b) every candidate from $A \cup D \cup B$ makes at most t points, and
(c) every candidate from Z makes at most one point.

Fig. 3 sketches a winning extension of the composed instance X making use of a winning extension of an instance x_w. We omit to show that the constructed instanced X is a yes-instance for $(3s+4)$-approval if and only if there is an $i \in \{1, \ldots, q\}$ such that x_i is a yes-instance for k-approval.

Theorem 6. *For k-approval,* POSSIBLE WINNER *with t partial votes does not admit a polynomial problem kernel with respect to (t, k) unless* $\mathrm{NP} \subseteq \mathrm{coNP} \,/\, \mathrm{poly}$.

4 Outlook

Can similar results as in this paper be obtained for "more general" problems such as SWAP BRIBERY [11] or the counting version of POSSIBLE WINNER[1]? This comprises the development of reduction rules preserving all winning extensions. Another interesting scenario might be as follows. Given a number s of winners in the input, for example, the size of a committee, one is interested in the s candidates such that each of them has more points than the remaining candidates. For this scenario, the negative results for POSSIBLE WINNER for k-approval as given in this work and related work [3, 5] can be adapted by adding $s - 1$ fixed candidates that always win, but as to the algorithmic results, it is open whether they extend to this scenario.

References

1. Y. Bachrach, N. Betzler, and P. Faliszewski. Probabilistic possible winner determination. In *Proc. of 24th AAAI*, 2010. To appear.
2. D. Baumeister and J. Rothe. Taking the final step to a full dichotomy of the Possible Winner problem in pure scoring rules. In *Proc. of 19th ECAI*, 2010. Short paper.
3. N. Betzler and B. Dorn. Towards a complexity dichotomy of finding possible winners in elections based on scoring rules. In *Proc. of 34th MFCS*, volume 5734 of *LNCS*, pages 124–136. Springer, 2009. Longversion to appear in *J. Comput. Syst. Sci.*
4. N. Betzler, J. Guo, and R. Niedermeier. Parameterized computational complexity of Dodgson and Young elections. *Inform. Comput.*, 208(2):165–177, 2010.
5. N. Betzler, S. Hemmann, and R. Niedermeier. A multivariate complexity analysis of determining possible winners given incomplete votes. In *Proc. of 21st IJCAI*, pages 53–58, 2009.
6. H. L. Bodlaender. Kernelization: New upper and lower bound techniques. In *Proc. of 4th IWPEC*, volume 5917 of *LNCS*, pages 17–37. Springer, 2009.
7. H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels. *J. Comput. Syst. Sci.*, 75(8):423–434, 2009.
8. B. Chor, M. Fellows, and D. W. Juedes. Linear kernels in linear time, or how to save k colors in $o(n^2)$ steps. In *Proc. of 30th WG*, volume 3353 of *LNCS*, pages 257–269. Springer, 2004.
9. M. Dom, D. Lokshtanov, and S. Saurabh. Incompressibility through colors and IDs. In *Proc. of 36th ICALP*, volume 5555 of *LNCS*, pages 378–389. Springer, 2009.
10. R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.
11. E. Elkind, P. Faliszewski, and A. Slinko. Swap bribery. In *Proc. of 2nd SAGT*, volume 5814 of *LNCS*, pages 299–310. Springer, 2009.
12. L. Fortnow and R. Santhanam. Infeasibility of instance compression and succinct PCPs for NP. In *Proc. of 40th STOC*, pages 133–142. ACM, 2008.
13. J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38(1):31–45, 2007.
14. E. Hemaspaandra and L. A. Hemaspaandra. Dichotomy for voting systems. *J. Comput. Syst. Sci.*, 73(1):73–83, 2007.
15. K. Konczak and J. Lang. Voting procedures with incomplete preferences. In *Proc. of IJCAI-2005 Multidisciplinary Workshop on Advances in Preference Handling*, 2005.
16. R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.
17. M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Incompleteness and incomparability in preference aggregation. In *Proc. of 20th IJCAI*, pages 1464–1469, 2007.
18. L. Xia and V. Conitzer. Determining possible and necessary winners under common voting rules given partial orders. In *Proc. of 23rd AAAI*, pages 196–201. AAAI Press, 2008.