

Unweighted Coalitional Manipulation Under the Borda Rule is NP-Hard

Nadja Betzler

TU Berlin

Berlin, Germany

nadja.betzler@campus.tu-berlin.de

Rolf Niedermeier

TU Berlin

Berlin, Germany

rolf.niedermeier@tu-berlin.de

Gerhard J. Woeginger

TU Eindhoven

Eindhoven, Netherlands

gwoegi@win.tue.nl

Abstract

The Borda voting rule is a positional scoring rule where, for m candidates, for every vote the first candidate receives $m - 1$ points, the second $m - 2$ points and so on. A Borda winner is a candidate with highest total score. It has been a prominent open problem to determine the computational complexity of UNWEIGHTED COALITIONAL MANIPULATION UNDER BORDA: Can one add a certain number of additional votes (called manipulators) to an election such that a distinguished candidate becomes a winner? We settle this open problem by showing NP-hardness even for two manipulators and three input votes. Moreover, we discuss extensions and limitations of this hardness result.

1 Introduction

In their recent overview on “AI’s war on manipulation” Faliszewski and Procaccia [2010] write “An enigmatic open problem is the complexity of Unweighted Coalitional Manipulation under Borda.” Here, we settle this open problem by showing NP-hardness for UNWEIGHTED COALITIONAL MANIPULATION UNDER BORDA,¹ which we subsequently refer to as BORDA MANIPULATION. Informally speaking, BORDA MANIPULATION asks whether under the Borda rule (see Section 2 for a formal definition) a distinguished candidate can be made a winner by adding a certain number of manipulative votes (called manipulators).

Previous work. There is a large amount of work for over two decades concerning the study of the computational complexity of manipulation in elections [Faliszewski and Procaccia, 2010; Faliszewski *et al.*, 2010]. Hence, here we only highlight few previous publications related to BORDA MANIPULATION. For one manipulator, the problem can be easily solved in polynomial time [Bartholdi III *et al.*, 1989]. Zuckerman *et al.* [2009] showed that for BORDA MANIPULATION a greedy algorithm can always find a set of x manipulators if the given input allows $x - 1$ manipulators to make

¹The same result was independently announced by Davies *et al.* [2011].

a distinguished candidate win. In other words, this means that the optimization version of BORDA MANIPULATION is polynomial-time approximable with an additive error one. Regarding the computational complexity of UNWEIGHTED COALITIONAL MANIPULATION for scoring rules in general, there is only one NP-hardness result for an artificially constructed scoring rule [Xia *et al.*, 2010]. Similarly to our NP-hardness result it is based on an NP-hard scheduling problem. By way of contrast, WEIGHTED COALITIONAL MANIPULATION UNDER BORDA is known to be NP-hard even for (at least) three candidates [Conitzer *et al.*, 2007; Hemaspaandra and Hemaspaandra, 2007].

Regarding practical relevance, there is evidence that BORDA MANIPULATION “usually” is an easy-to-solve problem. This is justified by experimental work [Davies *et al.*, 2010] as well as by considering some forms of average case analysis [Procaccia and Rosenschein, 2007; Xia and Conitzer, 2008].

Finally, we mention that BORDA MANIPULATION is *fixed-parameter tractable* with respect to the number of candidates, that is, it can be solved in a running time whose exponential part only depends on a computable function in the number of candidates. This is a direct consequence of some integer linear programming formulation in combination with a result of Lenstra [1983] for more general problems (see e.g. [Betzler *et al.*, 2009; Dorn and Schlotter, 2010]).

Our results. Our central result is to show that BORDA MANIPULATION is NP-hard even in case of three input votes and two manipulators. The key to prove this result is to devise a polynomial-time many-one reduction from a “close” NP-hard problem from scheduling theory. The problem is called 2-NUMERICAL MATCHING WITH TARGET SUMS and its NP-hardness has been proven by Yu *et al.* [2004]. Notably, Xia *et al.* [2010] used a general version of this problem for their already mentioned NP-hardness result for an “artificial” scoring rule. We also show that BORDA MANIPULATION remains NP-hard not only for three input votes but also for any other number greater than three. Moreover, we provide a close analysis of our reduction and reveal that it requires very special “settings” in order to work, partially explaining why attempts to prove NP-hardness have failed so far. In particular, our findings also make clear that the (worst-case) NP-hardness of BORDA MANIPULATION has little to

say concerning the practical feasibility of manipulating Borda elections (also see Walsh [2010] for making this point in a more general way).

2 Preliminaries and basic observations

An *election* (V, C) consists of a multiset V of votes and a set C of candidates (or alternatives). A *vote* is a linear order (that is, a transitive, antisymmetric, and total relation) on C . The *Borda voting rule* determines a winner as follows. In every vote v , the best candidate is at position 1 and the least-liked candidate is at position $|C|$. For every vote v , the candidate at position i is assigned $|C| - i$ points, that is, the score of c in v is

$$s(v, c) := |C| - \text{“position of } c \text{ in } v\text{”}.$$

Each candidate with the highest total score is a *Borda winner*, that is, we mainly focus on the case that there may be several co-winners. Moreover, let $s(V, c) := \sum_{v \in V} s(v, c)$ denote the score of candidate c under all votes in V .

The UNWEIGHTED COALITIONAL MANIPULATION problem for Borda (BORDA MANIPULATION for short) is defined as follows.

Input: An election (V, C) , a distinguished candidate $c^* \in C$, and a positive integer t .

Question: Is there a multiset W (called *coalition*) consisting of t votes over C such that c^* is a Borda winner of $(V \cup W, C)$?

We denote the votes from V as *nonmanipulative* and the votes from W as *manipulative* votes. Without loss of generality, we assume that c^* takes the first position in every vote of W . Regarding the nonmanipulative votes, one is mainly interested in the scores of the candidates. To this end, we introduce the following notion (analogously to [Davies *et al.*, 2010, Definition 1]):

For an election $E = (V, C)$ with $C := \{c^*, c_1, \dots, c_m\}$ and coalition size t , the *gap* of candidate c_i , $1 \leq i \leq m$, is

$$g_{E,t}(c_i) := s(V, c^*) + t \cdot m - s(V, c_i).$$

If the context is clear, we refer to the gap of candidate c_i simply by g_i . Intuitively, g_i denotes the number of points that c_i can make within W such that c^* is still a winner. Throughout the paper, we assume that $g_1 \leq g_2 \leq \dots \leq g_m$. Then, the following necessary condition for a yes-instance is easy to see [Davies *et al.*, 2010, Observation 1]. In a yes-instance, for $j \in \{1, \dots, m\}$,

$$\sum_{i=1}^j g_i \geq t \cdot j(j-1)/2 \quad (1)$$

since the candidates assigned to the last j positions of a vote make together $j(j-1)/2$ points.

A crucial concept used for our NP-hardness proof regards *tightness* with respect to an index j , that is, one has $\sum_{i=1}^j g_i = t \cdot j(j-1)/2$. This leads to the following easy-to-verify observation.

Observation 2.1 *If an instance is tight with respect to an index $j \in \{1, \dots, m\}$, then, in every solution, the candidates c_1, \dots, c_j take the last j positions in every manipulative vote and $s(W, c_i) = g_i$ for every $i \in \{1, \dots, j\}$.*

3 The NP-hardness proof

In the first subsection, we show that BORDA MANIPULATION is NP-hard with two manipulative and three nonmanipulative votes. In the second subsection, we then discuss to which other settings this hardness proof can be extended.

3.1 Two manipulators and three input votes

Yu, Hoogeveen & Lenstra [2004, Theorem 23] provided a very sophisticated proof that the following special case of NUMERICAL MATCHING WITH TARGET SUMS is NP-complete.²

2-NUMERICAL MATCHING WITH TARGET SUMS (2NMTS)

Input: A sequence a_1, \dots, a_k of positive integers with $\sum_{i=1}^k a_i = k(k+1)$ and $1 \leq a_i \leq 2k$ for $1 \leq i \leq k$.

Question: Are there two permutations ψ_1 and ψ_2 of the integers $1, \dots, k$ such that $\psi_1(i) + \psi_2(i) = a_i$ for $1 \leq i \leq k$?

Throughout the paper, we assume that $a_1 \leq \dots \leq a_k$. We devise a polynomial-time many-one reduction from 2NMTS to show that BORDA MANIPULATION is NP-hard in case of two manipulative and three nonmanipulative votes. We first describe the main idea based on specific gap values and the manipulative votes and then show how these gap values can be obtained by setting the nonmanipulative votes accordingly.

Consider an arbitrary instance a_1, \dots, a_k of 2NMTS. Assume that one can construct a BORDA MANIPULATION instance $(V, C, c^*, 2)$ with $C = \{c^*, c_1, \dots, c_k\}$ such that gap $g_i = 2k - a_i$ for every i , $1 \leq i \leq k$. Then, the constructed instance is a yes-instance of BORDA MANIPULATION if and only if the 2NMTS instance is a yes-instance: Given a solution for the 2NMTS instance, a solution for the BORDA MANIPULATION instance can be obtained as follows. Let c^* take the first positions in the manipulative votes. For every integer i , set the corresponding candidate c_i to position $\psi_1(i) + 1$ in the first manipulative vote and to position $\psi_2(i) + 1$ in the second manipulative vote. In this way, every candidate c_i makes $2k + 2 - a_i - 2 = g_i$ points in W and c^* wins.

To see the reverse direction, first note that

$$\sum_{i=1}^k g_i = \sum_{i=1}^k (2k - a_i) = 2k^2 - k(k+1) = k(k-1).$$

Hence, the BORDA MANIPULATION instance is tight with respect to k , implying that every candidate c_i makes exactly g_i points in a solution (see Observation 2.1). Let $p_1(i)$ and $p_2(i)$ denote the positions of c_i in the two manipulative votes, respectively. Since c_i makes exactly g_i points, $p_1(i) + p_2(i) = a_i + 2$ and hence setting $\psi_1(i) := p_1(i) - 1$ and $\psi_2(i) := p_2(i) - 1$ results in a solution for 2NMTS.

²Yu *et al.* used a slightly different problem notion which requires $\psi_1(i) + \psi_2(i) + u_i = e$ while we rephrase this by setting $a_i := e - u_i$ (resulting in $1 \leq a_i \leq 2k$ due to some side constraints in the definition of Yu *et al.*). Moreover, they denote the problem as RN3DM.

It remains to construct a set of nonmanipulative votes such that the gap g_i for every candidate c_i is realized, that is, the difference between the scores of c^* and c_i in the nonmanipulative votes is set such that c_i can make at most g_i points in the manipulative votes without beating c^* . We show NP-hardness even in case of having three nonmanipulative and two manipulative votes. To realize this, we will introduce a further set D of “dummy” candidates (described later) such that we end up with a new candidate set $C \uplus D$ with cardinality $m + 1$ (which will be specified later in the proof). The tightness with respect to k will ensure that the candidates from $C \setminus \{c^*\}$ must be at the last k positions in the manipulative votes in every solution.

To define the three nonmanipulative votes we will first fix the positions of the candidates from C and then fill the remaining positions with dummy candidates from D . To assign positions to the candidates from C , we rename the candidates c_1, \dots, c_k as follows. In an instance, several candidates might have the same gap value. Let G_1, \dots, G_h denote the *different* gap values. Consider a gap value G_j , $1 \leq j \leq h$, that occurs s_j times; that is, there are s_j indices such that the corresponding gap values equal G_j . Denote the s_j corresponding candidates by $c_j^1, \dots, c_j^{s_j}$. Let z be the maximum over all s_j , that is, the maximum number of occurrences of one gap value. Then, let D consist of $8zk$ new candidates. This means that

$$m := 8zk + k$$

and we end up with $m + 1$ candidates in total. To ease the representation, we assume that m is divisible by four³. Then, the distinguished candidate c^* is assigned to position $3m/4$ in every nonmanipulative vote.

Moreover, for candidate c_j^x , $1 \leq j \leq h$, $1 \leq x \leq s_j$, we compute the number $b(c_j^x)$ of candidates with “smaller” indices as follows

$$b(c_j^x) := |\{c_{j'}^{x'} \in C \text{ with } j' < j \text{ or } (j' = j \text{ and } x' < x)\}|.$$

Then, the positions of every candidate c_j^x in the three nonmanipulative votes are as follows.

- First vote: c_j^x is assigned to position $m/4 - 2xk + G_j$.
- Second vote: c_j^x is assigned to position $b(c_j^x) + 1$.
- Third vote: c_j^x is assigned to position $2xk - (b(c_j^x) + 1)$.

We first show that every assigned position is within the range of 1 and $m + 1$. Let p denote the position assigned to c_j^x

³This is no restriction. Since in the following we will also need that $|D| = 8zk$ is divisible by three, we sketch how to construct an equivalent instance of 2NMTS such that k (and thus also m and $|D|$) are divisible by 12 as follows. Let $p := k \bmod 12$. Add the target sums a_{k+1}, \dots, a_{k+p} with $a_{k+i} = 2(k+i)$. Then, it is easy to verify that this results in a valid new instance. Moreover, the new instance is a yes-instance if and only if the old instance is a yes-instance. To establish the equivalence the crucial idea is that in solutions for the new and the old instance the old target sums correspond to the same positions while every new a_{k+i} must correspond to position $k+i$ in both permutations of the new instance. This is easy to observe for a_{k+p} since the only possibility to build the target sum of $2(k+p)$ is to choose the position $k+p$ twice and this can be inductively shown for every other a_{k+i} with $p > i \geq 1$.

in the first vote. Since $x \leq z$ and $m = 8zk + k$, one has $p > 0$. Since there are at most k different gap values, one has $G_j \leq k$, and hence $x \geq 1$ implies $p < m/4$. Regarding the second and the third vote, the required range follows directly from the conditions $0 \leq b(c_j^x) \leq k - 1$ and $x \geq 1$. More specifically, all assigned positions are even smaller than $m/4$.

Furthermore, we show that in one vote every position is assigned to at most one candidate from C . For the second and the third vote two candidates from $C \setminus \{c^*\}$ do not coincide because of the order induced by the function b . Moreover, in all three votes, c^* appears at position $3m/4$ while the other candidates take a position smaller than $m/4$ (see above). In the first vote, if two candidates from C have the same gap value, then they must have different x -indices and hence different positions. Two candidates with different gap values do clearly assume different positions when having the same x -index and every candidate with other x -index is more than $2k > G_j$ positions away (for all j). The score of c_j^x in the three nonmanipulative votes is

$$\begin{aligned} 3(m+1) - m/4 - G_j + 2xk - 2xk - b(c_j^x) - 1 + b(c_j^x) + 1 \\ = (2 + 3/4) \cdot m - G_j + 3. \end{aligned}$$

Since c^* makes $2m$ points in the manipulative votes and $3 \cdot m/4 + 3$ points in the nonmanipulative votes, the gap of candidate c_j^x is G_j , the gap value required for every candidate c_j^x .

Finally, we describe how to fill the remaining positions, that is, positions that are not already assigned to candidates from C , with the dummy candidates from D . To this end, we partition D into three subsets D_1, D_2 , and D_3 of equal size (assuming that $|D|$ divisible by three; see second footnote). Then, in the first vote one has $D_1 > D_2 > D_3$, in the second vote $D_2 > D_3 > D_1$, and in the third vote $D_3 > D_1 > D_2$, where $D_i > D_j$ means that every candidate $d_i \in D_i$ has a smaller position than every candidate $d_j \in D_j$. Regarding the internal order of the candidates from $D_1 := \{d_1, \dots, d_l\}$, we assume that for every $i \in \{1, \dots, l-1\}$, we have $d_i > d_{i+1}$ in the first vote and $d_{i+1} > d_i$ in the second vote, and an arbitrary order in the third vote. Since every d_i from D_1 in the first and the second vote together makes at most $m+1$ points (due to the reverse orders) and in the third vote the $|D|/3$ candidates from D_3 have smaller positions, every candidate from D_1 makes at most

$$m + 1 + (m + 1 - |D|/3) = (1 + 2/3) \cdot m - k/3 + 2$$

points, using that $|D| = m - k$. The internal order of the candidates from D_2 and D_3 can be fixed analogously, resulting in the same upper bound for their scores. We show that every candidate $d \in D$ can make at least $m+k$ points in the manipulative votes and is still beaten by c^* :

$$\begin{aligned} s(V \cup W, c^*) - s(V, d) - (m+k) &\geq \\ (2 + 3/4) \cdot m + 3 - (1 + 2/3) \cdot m + k/3 - 2 - m - k &= \\ m/12 - 2k/3 + 1 &> 0 \end{aligned}$$

since $k = m/(8z+1) \leq m/9$ and hence $2k/3 \leq 2m/27 < m/12$. It follows that in the manipulative votes the candidates from D can assume all positions from 2 to $|D| + 1$

without beating c^* (by putting them in arbitrary order in the first vote and in the reverse order in the second vote). Moreover, because of the tightness with respect to k , the candidates c_1, \dots, c_k must be assigned to the last k positions in every possible solution (see Observation 2.1).

Altogether, since BORDA MANIPULATION clearly is in NP, one arrives at the following.

Theorem 3.1 *BORDA MANIPULATION is NP-complete for three nonmanipulative and two manipulative votes.*

Note that for the unique-winner case, that is, a candidate is a Borda winner only if it makes strictly more points than every other candidate, the construction can easily be modified by setting c^* to position $3m/4 - 1$ in the first nonmanipulative vote. Then, since c^* makes one point more in the nonmanipulative votes, the gap values remain the same and one can argue in complete analogy.

3.2 Other NP-hard cases

In the previous subsection, we showed NP-hardness for BORDA MANIPULATION in case of three nonmanipulative and two manipulative votes. In this section, we discuss further settings to which this result can be extended.

First, the NP-hardness reduction described in Subsection 3.1 can be extended to any number of nonmanipulative votes greater than three.

Proposition 3.2 *For two manipulative and more than three nonmanipulative votes BORDA MANIPULATION remains NP-hard.*

Proof. (Sketch) For four nonmanipulative votes one can modify the construction from Subsection 3.1 roughly as follows.

- In the second vote assign candidate c_j^x to position $2b(c_j^x) + 2$ while c^* remains at the old position.
- In the (new) fourth vote let c^* be at position $k + 1$ and c_j^x at position $k - b(c_j^x)$.

Then, the gap values for c_j^x remains the same since it “loses” $b(c_j^x) + 1$ points against c^* in the second vote but wins $b(c_j^x) + 1$ points against c^* in the fourth vote. The dummy candidates can be adapted to this case appropriately.

Moreover, every other number of manipulative votes can be achieved by adding pairs of any vote and its “reversal” without changing the relative scores. \square

Second, we briefly discuss the case a coalition size greater than two. Yu [1996] was convinced to have a proof for the conjecture that the construction of Yu et al. [2004] can be adapted so that it yields NP-hardness of d -NUMERICAL MATCHING WITH TARGET SUMS for any fixed $d \geq 3$ where d denotes the number of permutations.⁴ If the conjecture holds, arguments analogous to the one in Subsection 3.1 would imply that for any fixed coalition size $d \geq 2$, BORDA MANIPULATION is NP-hard.

⁴Yu passed away in 2002 and so this conjecture was not proven until now. Since the construction used for 2NMTS is already very sophisticated, this seems to be a demanding task.

4 A more refined look at the reduction

The NP-hardness of BORDA MANIPULATION stands in sharp contrast to the problem being easy to solve in practice [Davies et al., 2010]. Moreover, in probabilistic settings, it is provably often polynomial-time solvable [Procaccia and Rosenschein, 2007; Xia and Conitzer, 2008]. In contrast to study ways of assessing “average hardness”, in the following we pursue the approach of “deconstructing intractability” [Niedermeier, 2010]. To this end, we investigate the structure of instances resulting from the NP-hardness reduction (going back to the NP-hardness proof of 2NMTS). We reveal that the instance resulting from the reduction does not resemble realistic settings. Clearly, this does not directly provide any information about instances that are not obtained by the reduction but nevertheless it helps on the way to identify and characterize “easy” instances. Moreover, this also helps to understand what makes the problem difficult and leads the way to interesting questions for future research.

The NP-hardness proof for BORDA MANIPULATION relies on a “series” of polynomial-time many-one reductions starting from the “classical” strongly NP-complete 3-PARTITION problem.

Input: A multiset of $3q$ positive integers $X = \{x_1, \dots, x_{3q}\}$ and a positive integer b such that $b < x_j < 2b$ for $j \in \{1, \dots, 3q\}$ and $\sum_{j=1}^{3q} x_j = 4qb$.
Question: Is there a partition of X into k disjoint subsets X_1, \dots, X_q such that $\sum_{x_j \in X_i} x_j = 4b$ for $i \in \{1, \dots, q\}$?

More specifically, 3-PARTITION is reduced to a scheduling problem which in turn can be reduced to 2NMTS [Yu et al., 2004] and which then can be reduced to BORDA MANIPULATION (see the proof of Theorem 3.1).

A BORDA MANIPULATION instance resulting from the reductions is as follows. The number k of nondistinguished candidates is $4q^2b + 4qb$ in the construction resulting from [Yu et al., 2004]. Herein, following Subsection 3.1, k denotes the number of integers of the 2NMTS instance. We omit the dummy candidates from further consideration (since they are only an auxiliary tool to show NP-hardness for a constant number of nonmanipulative votes). Moreover, the corresponding gap values are as follows. There are

- $4q^2b$ candidates with gap $k + 4b + 1$,
- $4qb - 3q$ candidates with gap $k - 4qb - 2$, and
- one candidate with gap $k - 4qb - 2 + x_j$ for every integer x_j from the 3-PARTITION instance with $x_j < 2b$.

In the constructed instance the gaps imply that the distinguished candidate has a strictly smaller score than every other candidate in the nonmanipulative votes. Note that, although very restrictive, this “requirement” in general does not lead to NP-hardness: An instance with $2k > g_i \geq k + 1$ for every $i \in \{1, \dots, k\}$ allows for a trivial solution although the distinguished candidate makes less points than every other candidate in the nonmanipulative votes (by setting the candidates in an arbitrary order in the first manipulative vote and the reverse order in the second vote). This observation leads to the following question. Call a candidate c_i a *strong opponent* if

Table 1: Two manipulative votes v_1 and v_2 illustrating the strategy used in the proof of Proposition 4.2.

pos.	1	2	$m - j + 2$...	$m + 1$
score	m	$m - 1$	$j - 1$...	0
v_1 :	c^*	c_m	c_{m-1}	...	c_j	...	c_1
v_2 :	c^*	c_j	c_{j+1}	...	c_m

$g_i < k + 1$. Moreover, let n_s denote the number of strong opponents. Now, the question arises whether BORDA MANIPULATION is fixed-parameter tractable with respect to n_s .

Note that in the given construction the number of strong opponents exceeds the number of elements from the 3-PARTITION instance and hence the reduction does not show “fixed-parameter intractability” with respect to n_s .

In the remainder of this section, we discuss another property of the reduction that might be unlikely to hold in realistic settings and whose “relaxation” leads to a sufficient condition for an instance being a yes-instance.

Relaxing Tightness. One crucial property for showing the hardness of 2NMTS and BORDA MANIPULATION was that there is an index j of “unbounded” size such that $\sum_{i=1}^j g_i = j(j - 1)$, while for almost all indices $j' < j$ it holds that $\sum_{i=1}^{j'} g_i > j'(j' - 1)$. An instance getting tight (only) for a large number of candidates seems unlikely for realistic settings. In the following, we discuss aspects of “relaxing” or “strengthening” the tightness requirements, ending up with cases for which BORDA MANIPULATION for two manipulative votes is easy to solve.

First, note that when an instance is tight for every index i , then it can be solved by simply putting c_i to position $|C| - i$ in every vote. This observation can be extended as follows. Let $t_1 \leq \dots \leq t_x$ denote the x indices for which tightness holds.

Observation 4.1 BORDA MANIPULATION is fixed-parameter tractable with respect to the parameter $\max_{1 \leq i < x} \{t_{i+1} - t_i\}$.

Observation 4.1 follows directly from the fact that the candidates corresponding to the gaps between two tight indices must also be in the corresponding position range in every solution and for every such “range” one can apply a simple ILP-formulation from which fixed-parameter tractability follows via a result from Lenstra [1983].

Second, we describe a condition relaxing the tightness for *all* indices that leads to a sufficient condition for a yes-instance of BORDA MANIPULATION. Recall that for the gaps we assume that $g_i \leq g_{i+1}$ for $i \in \{1, \dots, m - 1\}$ and non-distinguished candidates c_1, \dots, c_m .

Proposition 4.2 For a BORDA MANIPULATION instance with $\sum_{i=1}^j g_i \geq 1.5 \cdot j(j - 1)$ for all $1 \leq j \leq m$, two manipulative voters can always make the distinguished candidate win.

Proof. We show that under the given condition a simple greedy strategy always leads to a solution: In the first manipulative vote, order the candidates according to their gap values

in decreasing order (ties are broken arbitrarily). More specifically, the first position is assigned to c^* , the second position to c_m , and so on (see Table 1).⁵

We show by contradiction that there is a candidate c_j such that c_j can take the second position (that is, the next position after c^*) in the second manipulative vote without beating c^* ; in other words, $g_j \geq m + j - 2$. Now, assume that such an index j does not exist. This implies $g_j < m + j - 2$ for every j . Then,

$$\begin{aligned} \sum_{i=1}^m g_i &< \sum_{i=1}^m (m + i - 2) \\ &= m^2 - 2m + 0.5 \cdot m(m + 1) \\ &= 1.5 \cdot m^2 - 1.5 \cdot m \\ &= 1.5 \cdot m(m - 1), \end{aligned}$$

a contradiction to the condition of the proposition.

Now, the second manipulative vote can be “filled” as follows. Put a candidate c_j with $g_j \geq m + j - 2$ at position 2. Moreover, all positions up to position j in the second vote can be assigned to the candidates c_{j+1}, \dots, c_m in this order. This is true since $g_j \leq g_i$ for $j < i$ and by this assignment every such candidate makes exactly the same score as g_j . To fill the positions greater than $m - j + 2$, we apply the described strategy again, that is, find a $j' < j$ such that $c_{j'}$ can take position $m - j + 3$ and so on. \square

Summarizing, the condition $\sum_{i=1}^j g_i \geq j(j - 1)$ (see Inequality 1 in Section 2) provides a necessary condition for an instance being a yes-instance of BORDA MANIPULATION while $\sum_{i=1}^j g_i \geq 1.5 \cdot j(j - 1)$ provides a sufficient condition (from Proposition 4.2). It is interesting to find stronger bounds for both cases, for example, what happens when $\sum_{i=1}^j g_i \geq j^2$ for every j ?

Finally, note that Proposition 4.2 may also apply to instances with strong opponents, that is, candidates with gap at most m . For example, there can be one candidate c_i with $g_i = 0$ or $m/4$ candidates each with $g_i = m$ (which are allowed to make $m^2/4 > 1.5 \cdot m/4 \cdot (m/4 - 1)$ points).

5 Conclusion

We showed NP-hardness for BORDA MANIPULATION even for very restricted settings such as having constant numbers of input votes and manipulators. Our NP-hardness proof is of theoretical nature in the sense that it is a purely worst-case result with little impact on practical aspects of solving BORDA MANIPULATION. This also motivates the issue of parameterizing NP-hard problems such as BORDA MANIPULATION in the spirit of multivariate algorithmics [Niedermeier, 2010]. In this context, our results still leave a number of interesting challenges for future work. For instance, we still miss a combinatorial algorithm (not based on integer linear pro-

⁵Iteratively applying this greedy algorithm provides an approximation algorithm with an additive error of one for the optimization problem of minimizing the size of a “winning coalition” [Zuckerman *et al.*, 2009].

gramming) to solve BORDA MANIPULATION efficiently⁶ in case of few candidates and an “unbounded” coalition size. It is also of interest whether in case of two manipulators one can solve the problem in less than $O(|C|!)$ time.

Moreover, with Inequality 1 (see Section 2) and Proposition 4.2 one now has one necessary and one sufficient condition for a BORDA MANIPULATION instance being a yes-instance. Besides improving the “gap” between the two corresponding bounds, it might be also interesting to “evaluate” the instances so far used for experimental studies or further real-world instances by answering the following question. What percentage of typically tested instances can not be decided based on the necessary or sufficient condition?

Furthermore, all our NP-hardness results rely on having a constant number of manipulative votes and an *unbounded* number of candidates. However, in many realistic voting scenarios one has only a small number of candidates but a large number of votes. Hence, it might be interesting to investigate whether BORDA MANIPULATION becomes “easy” when the coalition size is much larger than the number of candidates.

Finally, this paper explores a close connection between BORDA MANIPULATION and 2NMTS as well as the corresponding scheduling problem from [Yu *et al.*, 2004] to show computational hardness. On the positive side, it seems interesting to investigate whether some of the algorithmic results from the scheduling or matching literature can also be applied to BORDA MANIPULATION or can be helpful to design algorithms for manipulation problems in general. Some results in this direction have already been obtained by Xia *et al.* [2010] but this still might be a fruitful field.

Acknowledgements. NB was supported by the DFG project “PAWS”, NI 369/10. We are very grateful to the anonymous referees whose detailed and constructive feedback helped to significantly improve our presentation.

References

- [Bartholdi III *et al.*, 1989] J. J. Bartholdi III, C. A. Tovey, and M. A. Trick. The computational difficulty of manipulating an election. *Social Choice and Welfare*, 6:227–241, 1989.
- [Betzler *et al.*, 2009] N. Betzler, S. Hemmann, and R. Niedermeier. A multivariate complexity analysis of determining possible winners given incomplete votes. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 53–58, 2009.
- [Conitzer *et al.*, 2007] V. Conitzer, T. Sandholm, and J. Lang. When are elections with few candidates hard to manipulate? *Journal of the ACM*, 54(3):1–33, 2007.
- [Davies *et al.*, 2010] J. Davies, G. Katsirelos, N. Narodytska, and T. Walsh. An empirical study of Borda manipulation. In *Proceedings of the 3rd International Workshop on Computational Social Choice*, pages 91–102, 2010.
- [Davies *et al.*, 2011] J. Davies, G. Katsirelos, N. Narodytska, and T. Walsh. Complexity of and algorithms for Borda manipulation. Under review for *AAAI 2011*, 2011.
- [Dorn and Schlotter, 2010] B. Dorn and I. Schlotter. Multivariate complexity analysis of swap bribery. In *Proceedings of the 5th International Symposium on Parameterized and Exact Computation (IPEC)*, volume 6478 of *LNCS*, pages 107–122. Springer, 2010.
- [Faliszewski and Procaccia, 2010] P. Faliszewski and A. Procaccia. AI’s war on manipulation: Are we winning? *AI Magazine*, 31(4):53–64, 2010.
- [Faliszewski *et al.*, 2010] P. Faliszewski, E. Hemaspaandra, and L. A. Hemaspaandra. Using complexity to protect elections. *Communications of the ACM*, 53(1):74–82, 2010.
- [Hemaspaandra and Hemaspaandra, 2007] E. Hemaspaandra and L. A. Hemaspaandra. Dichotomy for voting systems. *Journal of Computer and System Sciences*, 73(1):73–83, 2007.
- [Lenstra, 1983] H. W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8:538–548, 1983.
- [Niedermeier, 2010] R. Niedermeier. Reflections on multivariate algorithmics and problem parameterization. In *Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science (STACS’10)*, volume 5 of *LIPICs*, pages 17–32, 2010.
- [Procaccia and Rosenschein, 2007] A. Procaccia and J. S. Rosenschein. Junta distributions and the average-case complexity of manipulating elections. *Journal of Artificial Intelligence Research*, 28:157–181, 2007.
- [Walsh, 2010] T. Walsh. Is computational complexity a barrier to manipulation? In *Proceedings of the 11th International Workshop on Computational Logic in Multi-Agent Systems (CLIMA)*, volume 6245 of *LNCS*, pages 1–7. Springer, 2010.
- [Xia and Conitzer, 2008] L. Xia and V. Conitzer. Generalized scoring rules and the frequency of coalitional manipulability. In *Proceedings of the 9th ACM Conference on Electronic Commerce (EC)*, pages 109–118, 2008.
- [Xia *et al.*, 2010] L. Xia, V. Conitzer, and A. D. Procaccia. A scheduling approach to coalitional manipulation. In *Proceedings of the 11th ACM Conference on Electronic Commerce (EC)*, pages 275–284. ACM, 2010.
- [Yu *et al.*, 2004] W. Yu, H. Hoogeveen, and J. K. Lenstra. Minimizing makespan in a two-machine flow shop with delays and unit-time operations is NP-hard. *Journal of Scheduling*, 7:333–348, 2004.
- [Yu, 1996] W. Yu. Personal communication. 1996.
- [Zuckerman *et al.*, 2009] M. Zuckerman, A. D. Procaccia, and J. S. Rosenschein. Algorithms for the coalitional manipulation problem. *Artificial Intelligence*, 173(2):392–412, 2009.

⁶The key issue here is showing fixed-parameter tractability, that is, an algorithm with running time $f(|C|) \cdot |V \cup W|^{O(1)}$ where, importantly, the degree of the polynomial does not depend on $|C|$.