Exact Algorithms for Weighted and Unweighted Borda Manipulation Problems

(Extended Abstract)

Yongjie Yang*
Universität des Saarlandes
Saarbrücken, Germany
yyongjie@mmci.uni-saarland.de

Jiong Guo†
Universität des Saarlandes
Saarbrücken, Germany
jguo@mmci.uni-saarland.de

ABSTRACT

Both weighted and unweighted Borda manipulation problems have been proved $\mathcal{NP}$-hard. However, there is no exact combinatorial algorithm known for these problems. In this paper, we initiate the study of exact combinatorial algorithms for both weighted and unweighted Borda manipulation problems. More precisely, we propose $O^*((m \cdot 2^m)^{t+1})$ - time and $O^*(t^{2m})$ - time[1] combinatorial algorithms for weighted and unweighted Borda manipulation problems, respectively, where t is the number of manipulators and m is the number of candidates. Thus, for $t = 2$ we solve one of the open problems posted by Betzler et al. [IJCAI 2011]. As a byproduct of our results, we show that the unweighted Borda manipulation problem admits an algorithm of running time $O^*(2^{9m^2 \log m})$, based on an integer linear programming technique. Finally, we study the unweighted Borda manipulation problem under single-peaked elections and present polynomial-time algorithms for the problem in the case of two manipulators, in contrast to the $\mathcal{NP}$-hardness of this case in general settings.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity; G.2.1 [**Combinatorics**]: Combinatorial algorithms; J.4 [**Computer Applications**]: Social Choice and Behavioral Sciences

General Terms

Algorithms

Keywords

voting systems, Borda manipulation, exact combinatorial algorithm, single-peaked election

*Supported by the DFG Cluster of Excellence (MMCI) and the China Scholarship Council (CSC).

†Supported by the DFG Cluster of Excellence (MMCI).

[1] $O^*()$ is the $O()$ notation with suppressed factors polynomial in the size of the input.

Appears in: *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2013)*, Ito, Jonker, Gini, and Shehory (eds.), May, 6–10, 2013, Saint Paul, Minnesota, USA.

1. PRELIMINARIES

This paper studies combinatorial algorithms for Borda manipulation problems.

In Weighted Borda Manipulation (WBM for short), we are given a set $\mathcal{C} \cup \{p\}$ of *candidates*, a multiset $\Pi_{\mathcal{V}} = \{\pi_{v_1}, \pi_{v_2}, ..., \pi_{v_n}\}$ of *votes* casted by a corresponding set $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ of *voters* (π_{v_i} is casted by v_i), a set $\mathcal{V}'$ of t manipulators and weight functions $f_1 : \mathcal{V} \to \mathbb{N}$ and $f_2 : \mathcal{V}' \to \mathbb{N}$, and asked whether the manipulators can cast their votes $\Pi_{\mathcal{V}'}$ in such a way that p uniquely wins the weighted election $(\mathcal{C} \cup \{p\}, \Pi_{\mathcal{V}} \cup \Pi'_{\mathcal{V}'}, \mathcal{V} \cup \mathcal{V}', f : \mathcal{V} \cup \mathcal{V}' \to \mathbb{N})$, where $f(v) = f_1(v)$ if $v \in \mathcal{V}$ and $f(v) = f_2(v)$ otherwise. Here, each vote π_v is defined as a bijection $\pi_v : \mathcal{C} \cup \{p\} \to [|\mathcal{C} \cup \{p\}|]$ and contributes $f(v) \cdot (pos(c) - 1)$ score to the candidate c, where the *position* of c in v is defined as $pos(c) = \pi_v(c)$. The unique winner is the candidate who has the most total score.

Unweighted Borda Manipualtion (UBM for short) is a special case of WBM where all voters and manipulators have the same unit weight, that is, $f_1 : \mathcal{V} \to \{1\}$ and $f_2 : \mathcal{V}' \to \{1\}$.

For a candidate c and a voter v, we use $SC_v(c)$ to denote the score of c which is contributed by v, that is, $SC_v(c) = f(v) \cdot (\pi_v(c) - 1)$. Let $SC_{\mathcal{V}}(c)$ denote the total score of c contributed by voters in $\mathcal{V}$, that is, $SC_{\mathcal{V}}(c) = \sum_{v \in \mathcal{V}} SC_v(c)$.

2. ALGORITHM FOR WEIGHTED CASE

Let $((\mathcal{C} \cup \{p\}, \Pi_{\mathcal{V}}, \mathcal{V}, f_1), \mathcal{V}', f_2, t)$ be the given instance. It is clear that any true-instance has a solution where every manipulator places p in the highest position, that is, a solution $\Pi_{\mathcal{V}'}$ with $SC_{\mathcal{V} \cup \mathcal{V}'}(p) = SC_{\mathcal{V}}(p) + \sum_{v' \in \mathcal{V}'} f(v') \cdot |\mathcal{C}|$. Therefore, to make p the winner, $SC_{\mathcal{V}'}(c) \leq g(c)$ should be satisfied for all $c \in \mathcal{C}$, where $g(c) = SC_{\mathcal{V}}(p) + \sum_{v' \in \mathcal{V}'} f(v') \cdot |\mathcal{C}| - SC_{\mathcal{V}}(c) - 1$. The value of $g(c)$ is called the *capacity* of c. Meanwhile, if in the given instance there is a candidate c with $g(c) < 0$, then the given instance must be a false-instance. Therefore, we assume that there is no candidate c with $g(c) < 0$. We reformulate WBM as follows:

Reformulation of WBM
Input: A set $\mathcal{C}$ of candidates associated with a capacity function $g : \mathcal{C} \to \mathbb{N}$, and a multiset $F = \{f_1, f_2, ..., f_t\}$ of nonnegative integers.
Question: Is there a multiset $\Pi = \{\pi_1, \pi_2, ..., \pi_t\}$ of bijections mapping from $\mathcal{C}$ to $[|\mathcal{C}|]$ such that $\sum_{i=1}^{t} f_i \cdot (\pi_i(c) - 1) \leq g(c)$ holds for all $c \in \mathcal{C}$?

Our algorithm is based on a dynamic programming method which is associated with a boolean dynamic table defined as $DT(C, Z_1, Z_2, ..., Z_t)$, where $C \subseteq \mathcal{C}$ is a subset of candidates, $Z_i \subseteq [|\mathcal{C}|]$ and $|C| = |Z_i|$ for all $i \in [t]$. Here, each Z_i encodes the positions that are occupied by the candidates of C in the vote casted by the i-th manipulator. The entry $DT(C, Z_1, Z_2, ..., Z_t) = 1$ means that there is a multiset $\Pi = \{\pi_1, \pi_2, ..., \pi_t\}$ of bijections mapping from $\mathcal{C}$ to $[|\mathcal{C}|]$ such that for each $i \in [t]$, $\bigcup_{c \in C}\{\pi_i(c)\} = Z_i$ and, for every candidate $c \in C$, c is "safe" under Π. Here, we say a candidate c is safe under Π, if $\sum_{i=1}^{t} f_i \cdot (\pi_i(c) - 1) \leq g(c)$. Intuitively, $DT(C, Z_1, Z_2, ..., Z_t) = 1$ means that we can place all candidates of C in the positions encoded by Z_i for all $i \in [t]$ without exceeding the capacity of any $c \in C$. Clearly, a given instance of WBM is a true-instance if and only if $DT(\mathcal{C}, Z_1 = [|\mathcal{C}|], Z_2 = [|\mathcal{C}|], ..., Z_t = [|\mathcal{C}|]) = 1$. We update the entries $DT(C, Z_1, Z_2, ..., Z_t)$ with $|C| = |Z_1| = |Z_2| = ... = |Z_t| = l$ as follows: if $\exists c \in C$ and $\exists z_i \in Z_i$ for all $i \in [t]$ such that $DT(C \setminus \{c\}, Z_1 \setminus \{z_1\}, Z_2 \setminus \{z_2\}, ..., Z_t \setminus \{z_t\}) = 1$ and $DT(\{c\}, \{z_1\}, \{z_2\}, ..., \{z_t\}) = 1$, then $DT(C, Z_1, Z_2, ..., Z_t) = 1$, otherwise, $DT(C, Z_1, Z_2, ..., Z_t) = 0$.

THEOREM 1. *WBM is solvable in* $O^*((|\mathcal{C}| \cdot 2^{|\mathcal{C}|})^{t+1})$ *time.*

In [1], Betzler et al. posed as an open question whether UBM in the case of two manipulators can be solved in less than $O^*(|\mathcal{C}|!)$ time. By Theorem 1, we can answer this question affirmatively.

COROLLARY 2. *WBM (UBM is a special case of WBM) in the case of two manipulators can be solved in* $O^*(8^{|\mathcal{C}|})$ *time.*

3. ALGORITHM FOR UNWEIGHTED CASE

Recall that UBM is a special case of WBM where all voters have the same unit weight. However, compared to the weighted version, when we compute $SC_{\mathcal{V}'}(c)$ for a candidate c, it is irrelevant which manipulators placed c in the j-th positions. The decisive factor is the number of manipulators placing c in the j-th positions. This leads to the following approach where we firstly reduce UBM to a matrix problem and then solve this matrix problem by a dynamic programming technique, resulting in a better running time than in Corollary 2.

Filling Magic Matrix (FMM)
Input: A multiset $g = \{g_1, g_2, ..., g_m\}$ of non-negative integers and an integer $t > 0$.
Question: Is there an $m \times m$ matrix M with non-negative integers such that $\forall i \in [m]$, $\sum_{j=1}^{m} (j-1) \cdot M[i][j] \leq g_i$ and $\sum_{j=1}^{m} M[i][j] = t$, and $\forall j \in [m]$, $\sum_{i=1}^{m} M[i][j] = t$?

The algorithm for FMM is based on a dynamic programming method associated with a boolean dynamic table $DT(l, T)$, where $l \in [m]$ and $T = \{T_j \in \mathbb{N} \mid j \in [m], T_j \leq t\}$ is a multiset. The entry $DT(l, T) = 1$ means that there is an $m \times m$ matrix M such that: (1) $\sum_{j=1}^{m} M[i][j] = t$ for all $i \in [l]$; (2) $\sum_{i=1}^{l} M[i][j] = T_j$ for all $j \in [m]$; and (3) $\sum_{j=1}^{m}(j-1) \cdot M[i][j] \leq g_i$ for all $i \in [l]$. It is clear that a given instance of FMM is a true-instance if and only if $DT(m, T_{[m]}) = 1$, where $T_{[m]}$ is the multiset containing m copies of t. We update $DT(l, T)$ for $2 \leq l \leq t$ and all possible multiset $T = \{T_j \in \mathbb{N} \mid j \in [m], T_j \leq t\}$ as follows: if there is a multiset $T' = \{T'_j \in \mathbb{N} \mid j \in [m], T'_j \leq T_j\}$ such that $DT(l-1, T') = 1$, $\sum_{j=1}^{m} (T_j - T'_j) = t$ and $\sum_{j=1}^{m} (j-1) \cdot (T_j - T'_j) \leq g_l$, then set $DT(l, T) = 1$; otherwise, set $DT(l, T) = 0$.

LEMMA 3. *FMM is solvable in* $O^*(t^{2m})$ *time.*

LEMMA 4. *UBM can be reduced to FMM in polynomial time.*

THEOREM 5. *UBM can be solved in* $O^*(t^{2|\mathcal{C}|})$ *time.*

Next we show that FMM can be solved by an integer linear programming (ILP) based algorithm. The ILP contains m^2 variables x_{ij} for $i, j \in [m]$ and, subject to the following four kinds of restrictions: (1) $\sum_{i=1}^{m} x_{ij} = t$ for all $j \in [m]$; (2) $\sum_{j=1}^{m} x_{ij} = t$ for all $i \in [m]$; (3) $\sum_{j=1}^{m} (j-1) \cdot x_{ij} \leq g_i$ for all $i \in [m]$; (4) $x_{ij} \geq 0$ for all $i, j \in [m]$; where $t \in \mathbb{N}$ and $g = \{g_1, g_2, \ldots, g_m\}$ with $g_i \in \mathbb{N}$ for all $i \in [m]$ are input.

LEMMA 6. *[4] An ILP problem with* ζ *variables can be solved in* $O^*(\zeta^{4.5\zeta})$ *time.*

Due to Lemmas 4 and 6, we have the following theorem.

THEOREM 7. *UBM admits an algorithm with running time* $O^*(2^{9|\mathcal{C}|^2 \log |\mathcal{C}|})$.

4. SINGLE-PEAKED ELECTIONS

It is known that UNWEIGHTED BORDA MANIPULATION is polynomial-time solvable with one manipulator [3] but becomes $\mathcal{NP}$-hard with two manipulators [1, 2]. Here, we show that this problem with two manipulators can be solved in polynomial time in single-peaked elections.

Let $\mathcal{L}$ be a linear order over the candidates $\mathcal{C}$. We say that a vote $\pi_v : \mathcal{C} \rightarrow [|\mathcal{C}|]$ is *coincident* with $\mathcal{L}$ if and only if for any three distinct candidates $a, b, c \in \mathcal{C}$ with $a \mathcal{L} b \mathcal{L} c$ or $c \mathcal{L} b \mathcal{L} a$, $\pi_v(c) > \pi_v(b)$ implies $\pi_v(b) > \pi_v(a)$. An election is a *single-peaked* election if there exists a linear order $\mathcal{L}$ over the candidates such that all votes of the election are coincident with $\mathcal{L}$.

THEOREM 8. UNWEIGHTED BORDA MANIPULATION *with two manipulators under single-peaked elections is polynomial-time solvable.*

5. REFERENCES

[1] N. Betzler, R. Niedermeier, and G. J. Woeginger. Unweighted coalitional manipulation under the Borda rule is NP-hard. In *IJCAI*, pages 55–60, 2011.

[2] J. Davies, G. Katsirelos, N. Narodytska, and T. Walsh. Complexity of and algorithms for Borda manipulation. In *AAAI*, pages 657–662, 2011.

[3] J. J. Bartholdi III, C.A. Tovey, and M. Trick. The computational difficulty of manipulating an election. *Social Choice and Welfare*, 6(3):227–241, 1989.

[4] R. Kannan. Minkowski's convex body theory and integer programming. *Mathematics of Operations Research*, 12:415–440, 1987.