Parameterized Computational Complexity of Dodgson and Young Elections

Nadja Betzler*, Jiong Guo**, and Rolf Niedermeier

Institut für Informatik, Friedrich-Schiller-Universität Jena, Ernst-Abbe-Platz 2, D-07743 Jena, Germany
{betzler,guo,niedermr}@minet.uni-jena.de

Abstract. We show that, other than for standard complexity theory with known NP-completeness results, the computational complexity of the Dodgson and Young election systems is completely different from a parameterized complexity point of view. That is, on the one hand, we present an efficient fixed-parameter algorithm for determining a Condorcet winner in Dodgson elections by a minimum number of switches in the votes. On the other hand, we prove that the corresponding problem for Young elections, where one has to delete votes instead of performing switches, is W[2]-complete. In addition, we study Dodgson elections that allow ties between the candidates and give fixed-parameter tractability as well as W[2]-hardness results depending on the cost model for switching ties.

1 Introduction

Computational social choice and, more specifically, the computational complexity of election systems has become an increasingly important field of interdisciplinary research [3, 7]. Besides the obvious application in computational politics where the fact that people have varying preferences concerning whom to elect leads to preference aggregation demands via some election system, it also has become very important in multiagent systems: In groups of software agents often a common decision has to be found, again taking into account different preferences about which decision is to be made. Thus, election systems for instance play a central role in planning (artificial intelligence in general) or page ranking systems for Internet search engines.

We study the following classic scenario for election systems: There is a set of candidates and a set of vote(r)s and each voter chooses an order of preference (total order) among the candidates. The well-known Condorcet principle from 1785 then requires that a winner of an election is the candidate who is preferred to each other candidate in more than half of the votes. Unfortunately, a Condorcet winner does not always exist. Hence, several voting systems have been proposed which always choose the Condorcet winner if it exists, and, otherwise,

* Supported by the DFG, research project DARE, GU 1023/1.
** Supported by the DFG, Emmy Noether research group PIAF, NI 369/4.

pick a candidate that is in some sense closest to being a Condorcet winner. In other words, these election systems deal with certain "editing problems". In this work, we focus on two classic editing problems from social choice theory [15], that is, the one due to C. L. Dodgson[1] from 1876 and the one due to H. P. Young from 1977. In Dodgson elections, the editing operation is to switch neighboring candidates in the voters' preference lists and the goal is to minimize the overall number of switches needed in order to end up with a Condorcet winner. In Young elections, the editing operation is to remove a vote, again trying to minimize the number of removals in order to end up with a Condorcet winner.

In their seminal work, Bartholdi et al. [1] initiated the study of the computational complexity of election systems. They showed that to decide whether a distinguished candidate can be made a Condorcet winner by performing no more than a given number of editing operations is NP-complete for both Dodgson and Young elections. In a further breakthrough, for Dodgson elections Hemaspaandra et al. [11] and later for Young elections Rothe et al. [18] showed that the corresponding winner and ranking problems are even complete for Θ_2^p, the class of problems that can be solved via parallel access to NP. Thus, Faliszewski et al. [7] concluded that "since checking whether a given candidate has won should be in polynomial time in any system to be put into actual use, these results show that Dodgson and Young elections are unlikely to be useful in practice". This is the point of view of classical, "one-dimensional" computational complexity analysis.[2] By way of contrast, we propose the framework of parameterized computational complexity theory [6, 9, 16] for studying election systems.[3]

For Dodgson and Young elections, we consider the question whether the NP-hard problems become fixed-parameter tractable with respect to the parameter number of editing operations. We choose this standard parameterization[4] as a natural first step towards a systematic (future) study using further parameterizations. As we can show, other than in the classical context, the parameterized complexity of Dodgson and Young elections completely differs. For n votes and m candidates, for Dodgson elections we can determine in $O(2^k \cdot nk + nm)$ time whether a distinguished candidate can be made a Condorcet winner by performing at most k switches, that is, the problem is fixed-parameter tractable with respect to the parameter k. In contrast, for Young elections the corresponding problem with the parameter denoting either the number of deleted votes or the number of remaining votes becomes W[2]-complete. Our results imply that Dodgson elections can be put in actual use whenever the input instances are close to having a Condorcet winner. This answers an open question of Christian et al. [4][5] and refutes a parameterized hardness conjecture of McCabe-Dansted [13].

[1] Also known as the writer Lewis Carroll.

[2] For more classical complexity results w.r.t. election systems we refer to [5, 10].

[3] In companion work [2], we also do so for Kemeny elections.

[4] In parameterized algorithmics [6, 9, 16] the solution size typically is the "standard parameter".

[5] In the meantime, Fellows et al. independently showed that Dodgson Score is fixed-parameter tractable with a worse running time [8].

Moreover, this complements recent work on a simple greedy heuristic for finding Dodgson winners with a guaranteed frequency of success [12] and some work on the approximability of Dodgson and Young elections [14, 17]. In particular, Procaccia et al. [17] gave (randomized) approximation algorithms for Dodgson elections and show that it is hard to approximate Young elections by any factor. Moreover, for Dodgson elections we can show that allowing ties (that is, votes may remain undecided between certain candidates), depending on the choice between two switching mechanisms, we either obtain fixed-parameter tractability or W[2]-completeness.

Due to lack of space, several details had to be deferred to the full version of this paper.

2 Preliminaries

Throughout this work, an *election* (V, C) consists of a set V of n votes and a set C of m candidates.[6] A vote is a *preference list* of the candidates, that is, for each voter the candidates are ordered by preference. For instance, in case of three candidates a, b, c, the ordering $a < b < c$ would mean that candidate c is the best liked one and candidate a is the least liked one for this voter. We also consider the case where *ties* are allowed in the votes, that is, instead that a vote consists of a totally ordered list of candidates it then may only be partially ordered. In an election (V, C), a candidate $c \in C$ is called *Condorcet winner* if c wins against every other candidate from C, that is, for each $d \in C \backslash \{c\}$, candidate c is better liked than d in at least $\lfloor n/2 \rfloor + 1$ votes, or, having ties, the number of votes in which c is better liked than d is higher than the number of votes in which d is better liked than c. Observe that a Condorcet winner does not always exist. As a consequence, several specific election systems have been introduced to remedy this situation. Here, we study two popular ones, that is, Dodgson and Young elections. To this end, define as a *switch* the swapping of the two positions of two neighboring candidates in a vote. Based on this, we now can introduce the basic computational problems of this work:

> Dodgson Score:
> *Given*: An election (V, C), a distinguished candidate $c \in C$, and an integer $k \geq 0$.
> *Question*: Can c be made a Condorcet winner by at most k switches?

In other words, for Dodgson Score, we ask whether the *Dodgson score* of c is at most k. The *Young score* is defined by the number of remaining votes:

> Young Score:
> *Given*: An election (V, C), a distinguished candidate $c \in C$, and an integer $l \geq 0$.
> *Question*: Is there a subset $V' \subseteq V$ of size at least l such that (V', C) has the Condorcet winner c?

[6] Note that we identify votes and voters.

The *dual Young score* is defined by the number of removed votes:

> Dual Young Score:
> *Given*: An election (V, C), a distinguished candidate $c \in C$, and an integer $k \geq 0$.
> *Question*: Is there a subset $V' \subseteq V$ of size at most k such that $(V \backslash V', C)$ has the Condorcet winner c?

Note that all three problems are NP-complete [1, 18].

To present our results, an important concept is the one of the *deficit* of a candidate $d \in C \backslash \{c\}$ against the distinguished candidate c: Let N_d denote the number of votes from V in which d *defeats* c, that is, in which d is better positioned than c. Then, the *Dodgson deficit* of d is $\lfloor (N_d - (n - N_d))/2 \rfloor + 1$, that is, the minimum number of votes in which the relative order of c and d has to be reversed such that c defeats d in strictly more than half of the votes. Analogously, the *Young deficit* is defined as $N_d - (n - N_d)$. Moreover, we call a candidate with a positive Dodgson deficit *dirty*.

Finally, we briefly introduce the relevant notions of parameterized complexity theory [6, 9, 16]. Parameterized algorithmics aims at a multivariate (at least two-dimensional) complexity analysis of problems. This is done by studying relevant problem parameters and their influence on the computational complexity of problems. The hope lies in accepting the seemingly inevitable combinatorial explosion for NP-hard problems, but to confine it to the parameter. Hence, the decisive question is whether a given parameterized problem is *fixed-parameter tractable (FPT)* with respect to the parameter, say k. In other words, here we ask for the existence of a solving algorithm with running time $f(k) \cdot \text{poly}(n, m)$ for some computable function f. Unfortunately, not all parameterized problems are fixed-parameter tractable. Downey and Fellows [6] developed a theory of parameterized intractability by means of devising a completeness program with complexity classes. The first two levels of (presumable) parameterized intractability are captured by the complexity classes W[1] and W[2]. We will show several W[2]-completeness results. There is good reason to believe that the corresponding problems thus are not fixed-parameter tractable. To this end, a reduction concept is needed. A *parameterized reduction* reduces a problem instance (I, k) in $f(k) \cdot \text{poly}(|I|)$ time to an instance (I', k') such that (I, k) is a yes-instance iff (I', k') is a yes-instance and k' only depends on k but not on $|I|$.

3 Dodgson Score

In this section, we describe an efficient fixed-parameter algorithm based on dynamic programming for the Dodgson Score problem with respect to the parameter score, answering an open question of Christian et al. [4]. The algorithm can not only decide whether a given Dodgson Score instance is a "yes"-instance, but for a "yes"-instance also constructs a set of at most k switches which lead to a modified input instance where the distinguished candidate c

becomes a Condorcet winner. The following two observations are used for the design of the algorithm:

First, it is easy to see (McCabe-Dansted [13, Lemma 2.19]) that there is always an optimal solution that considers only switches that move c in a vote to better positions (Observation 1). Making use of this, our algorithm only considers switches of such kind. Second, since a switch never increases any deficit, we only consider candidates with positive deficit (dirty candidates). With one switch, we can decrease the deficit of exactly one candidate by one. Therefore, with at most k switches allowed, in a yes-instance, the sum of the deficits of the dirty candidates is upper-bounded by k (Observation 2). This fact is crucial for the analysis of the algorithm when bounding the size of the dynamic programming table.

The basic idea of the algorithm is that a solution can be decomposed into subsolutions. In each subsolution the deficit of each dirty candidate is decreased by a certain amount, the *partial decrement*. More precisely, our dynamic programming considers a linear number of subsets of votes, beginning with the subset that contains only one vote and then extending it by adding the other votes one by one. For each of these vote subsets, we consider all possible combinations of partial decrements of deficits.

Definitions for the Algorithm. Let c be the distinguished candidate and let $C_d = (c_1, c_2, \ldots, c_p)$ denote the list of candidates with positive deficit in an arbitrary but fixed order. Let $D = (d_1, d_2, \ldots, d_p)$ be the corresponding *deficit list*.

The dynamic programming table is denoted by T, each row corresponding to a vote v_i for $i = 1, \ldots, n$ and each column corresponding to a *partial deficit list* $(d'_1, d'_2, \ldots, d'_p)$ with $0 \leq d'_j \leq d_j$ for $1 \leq j \leq p$. The entry $T(v_i, (d'_1, d'_2, \ldots, d'_p))$ stores the minimum number of switches within the votes $\{v_j \mid 1 \leq j \leq i\}$ that result in a new instance where the deficits of the p dirty candidates are *at most* $d'_1, d'_2, \ldots, d'_p$, respectively.[7] If a deficit list $(d'_1, d'_2, \ldots, d'_p)$ cannot be achieved by switching within the set of votes $\{v_j \mid 0 \leq j \leq i\}$, we set $T(v_i, (d'_1, d'_2, \ldots, d'_p)) := +\infty$.

Let $\text{switch}(v_i, c_j)$ denote the minimum number of switches needed such that in vote v_i candidate c defeats candidate c_j. If c already defeats c_j in v_i, then $\text{switch}(v_i, c_j) := 0$. For a deficit list $D' = (d'_1, d'_2, \ldots, d'_p)$ and a subset of indices $S \subseteq \{1, \ldots, p\}$, we use $D' + S$ to denote a deficit list $(e_1, \ldots, e_p)$ where $e_i := d'_i + 1$ for $i \in S$ and $d'_i < d_i$, and $e_i := d'_i$, otherwise. Analogously, for the original deficit list $D = (d_1, \ldots, d_p)$, $D - S$ denotes the list $(f_1, \ldots, f_p)$ where $f_i := d_i - 1$ if $i \in S$ and $f_i := d_i$, otherwise. Let $\text{best}(S, i)$ denote the candidate c_j with $j \in S$ that is best in vote v_i.

Algorithm. The dynamic programming algorithm for Dodgson Score is given in Figure 1. We assume that we already have the deficits of the candidates

[7] Note that with this definition of table entries, we do not have to consider deficit lists $(d'_1, \ldots, d'_p)$ where $d'_i < 0$ for some i. In this way, the case that an optimal solution may decrease the deficit of a dirty candidate to a negative value is also covered.

Algorithm *DodScore*
Input: Set of votes $V = \{v_1, \ldots, v_n\}$, set of candidates C, set of dirty candidates $C_d = \{c_1, \ldots, c_p\} \subseteq C$, distinguished candidate c, positive integer k with $p \leq k$, deficit list $D = (d_1, \ldots, d_p)$ of dirty candidates
Output: Yes, if c can become a Condorcet winner with at most k switches

Initialization:
01 for all $D' = (d'_1, \ldots, d'_p)$ with $0 \leq d'_j \leq d_j$ for $0 \leq j \leq p$
02 for $i = 1, \ldots, n$
03 $T(v_i, D') := +\infty$
04 for all $S \subseteq \{1, \ldots, p\}$
05 if c_j defeats c in v_1 for all $j \in S$ then
06 $T(v_1, D - S) := \text{switch}(v_1, \text{best}(S, 1))$
Recursion:
07 for $i = 2, \ldots, n$
08 for all $D' = (d'_1, \ldots, d'_p)$ with $0 \leq d'_j \leq d_j$ for $0 \leq j \leq p$
09 for all $S \subseteq \{1, \ldots, p\}$
10 if c_j defeats c in v_i for all $j \in S$ then
11 $T(v_i, D') := \min\{T(v_i, D'),\ T(v_{i-1}, D' + S) + \text{switch}(v_i, \text{best}(S, i))\}$

Output:
12 if $T(v_n, (0, 0, \ldots, 0)) \leq k$ then
13 return "Yes"

Fig. 1. Algorithm for Dodgson Score

and the sum of the deficits of the dirty candidates is at most k as argued in Observation 2. In the initialization of the first row of the dynamic programming table (Figure 1, lines 4–6), the algorithm considers all possible combinations of deficit decrements that can be achieved by switches within the first vote, and stores the minimum number of switches needed for each of them. In the recursion (lines 7–11), the subset of votes is extended by a new vote v_i and for the new subset $\{v_1, \ldots, v_i\}$ a solution for all partial deficit lists is computed by combining a number of switches within the new vote v_i with information already stored in the table.

Lemma 1. *The algorithm DodScore (Figure 1) is correct.*

Lemma 2. *The algorithm DodScore (Figure 1) runs in $O(4^k \cdot nk + nm)$ time.*

Proof. (Sketch) It is easy to see that the deficit list D can be computed in $O(nm)$ time by iterating over all votes and counting the deficits for all candidates. Now, we consider the size of the dynamic programming table.

A deficit d'_i can have values ranging from 0 to d_i. Hence, the number of partial deficit lists, that is, the number of columns in the table, is $\prod_{i=1}^{p}(d_i + 1)$. Clearly, for a potential "yes"-instance, we have the constraints $p \leq k$ and $\sum_{i=1}^{p} d_i \leq k$ (see Observations 1 and 2). It is not hard to see that 2^k is a tight upper bound on $\prod_{i=1}^{p}(d_i + 1)$. Thus, the overall table size is $n \cdot 2^k$.

For computing an entry $T(v_i, D')$, the algorithm iterates over all 2^p subsets of $\{1, \ldots, p\}$. For each such subset S, it computes the "distance" in v_i between

the best of the dirty candidates with indices in S and c, that is, the number of switches needed to make c better than this best dirty candidate. This distance can be computed in $O(k)$ time and, hence, the computation of $T(v_i, D')$ can be done in $O(2^k \cdot k)$ time. The initialization clearly needs $O(2^k \cdot n)$ time. Hence, table T can be computed in $O(2^k \cdot n \cdot 2^k \cdot k + 2^k \cdot n) = O(4^k \cdot nk)$ time. □

By making use of a "monotonicity property" of the table, we can improve the running time of *DodScore* as shown in the following theorem.

Theorem 1. Dodgson Score *can be solved in* $O(2^k \cdot nk + nm)$ *time.*

Proof. The improvement is achieved by replacing the innermost for-loop (lines 9–11) of the recursion which computes a table entry and needs $O(2^k \cdot k)$ time as shown in Lemma 2 by an instruction running in time linear in k.

Let $S_i(d)$ denote the set of the dirty candidates that are better than the distinguished candidate c but not better than the candidate d in vote v_i. This set is empty if d is worse than c in v_i. We replace lines 9–11 in Figure 1 by the following recurrence:

$$T(v_i, D') := \min_{1 \le r \le p} \{T(v_{i-1}, D' + S_i(c_r)) + \text{switch}(v_i, c_r)\}.$$

To prove the correctness of the recurrence, on the one hand, observe that, for every r with $1 \le r \le p$, there exists a subset $S \subseteq \{1, \ldots, p\}$ satisfying the if-condition in line 10 of *DodScore* such that $S = S_i(c_r)$ and $\text{best}(S, i) = c_r$. Thus,

$$\begin{aligned} &\min_{S \subseteq \{1,\ldots,p\}} \{T(v_{i-1}, D' + S) + \text{switch}(v_i, \text{best}(S, i))\} \\ \le &\min_{1 \le r \le p} \{T(v_{i-1}, D' + S_i(c_r)) + \text{switch}(v_i, c_r)\}. \end{aligned}$$

On the other hand, for every $S \subseteq \{1, \ldots, p\}$ satisfying the if-condition in line 10, there exists an r with $1 \le r \le p$ such that $S \subseteq S_i(c_r)$. For instance, let r be the index of the candidate in S that is the best in v_i; we then have $\text{best}(S, i) = c_r$ and, thus, $\text{switch}(v_i, \text{best}(S, i)) = \text{switch}(v_i, c_r)$. Moreover, from the definition of table entries, the following monotonicity of the table is easy to verify:

$$T(v_i, (d_1, \ldots, d_i, \ldots, d_p)) \ge T(v_i, (d_1, \ldots, d_i + 1, \ldots, d_p))$$

Thus, from $S \subseteq S_i(c_r)$, we conclude that $T(v_i, D' + S) \ge T(v_i, D' + S_i(c_r))$. Since $S_i(c_r) \subseteq \{1, \ldots, p\}$ and, by the definition of $S_i(c_r)$, this subset satisfies the if-condition in line 10, we then have

$$\begin{aligned} &\min_{1 \le r \le p} \{T(v_{i-1}, D' + S_i(c_r)) + \text{switch}(v_i, c_r)\} \\ \le &\min_{S \subseteq \{1,\ldots,p\}} \{T(v_{i-1}, D' + S) + \text{switch}(v_i, \text{best}(S, i))\}. \end{aligned}$$

The time for computing a table entry in the improved version is clearly $O(k)$: Before looking for the minimum, we can compute $S_i(c_r)$ for all $1 \le r \le p$ by iterating one time over v_i. Then, based on Lemma 2, the overall running time becomes $O(2^k \cdot nk + nm)$. □

Allowing Ties. As noted by Hemaspaandra et al. [11], there are two natural ways going from total to partial orders. In the first model, transforming $c < a = b$ into $a = b < c$ costs one switch and in the second model it costs two switches. We denote the problem based on the first model by Dodgson Tie Score 1 (DTS1) and the problem based on the second one by Dodgson Tie Score 2 (DTS2). In the second model we allow c to improve upon an arbitrary subset of equally ranked candidates in order to prevent for "paying" for all equally ranked candidates in a vote even if some of the candidates have already negative deficits.[8] Note that, in the case with ties, the deficit of a candidate $d \neq c$ is defined as $\lfloor (N_d - N_{\overline{d}})/2 \rfloor + 1$, where N_d is the number of votes in which d defeats c and $N_{\overline{d}}$ is the number of votes in which c defeats d. Further, one can choose upon which candidate one likes to improve c with the first switch. Then, the order within a set of equally ranked candidates including c does matter, for example, starting with $c < a = b < d$, by one switch we can either achieve $a < c < b < d$ or $b < c < a < d$. To improve c upon d, we need two additional switches in both cases. Moreover, with ties, we have now two sorts of switches: The first sort swaps the positions of c and other candidates, as in the case without ties, and the second sort breaks and builds ties between c and other candidates. Since these two sorts of switches decrease the deficits of candidates by different values, we introduce the notion of "switch cost". We associate a switch of the first sort with cost 2, while a switch of the second sort has cost 1. For example, given $c < a < b$, by one switch between c and a, we can get $c = a < b$ or $a < c < b$. However, the switch in the first case has cost 1 but the second case needs a cost-2 switch. Finally, the parameter we consider in the following is the total cost of the switches needed to make c a Condorcet winner.

Whereas the both variants DTS1 and DTS2 remain NP-complete (which easily follows from the NP-completeness of the case without ties [1]), their parameterized complexity differs. The problem DTS2 is fixed-parameter tractable, but DTS1 becomes W[2]-complete.[9] Note that compared to Theorem 1 (without ties) we obtain a slightly worse running time for DTS2. More precisely, due to a slight modification of algorithm *DodScore* from Figure 1 we can state the following.

Theorem 2. Dodgson Tie Score 2 *can be solved in* $O(4^k \cdot nk + nm)$ *time.*

Theorem 3. Dodgson Tie Score 1 *is* $W[2]$*-complete with respect to the parameter* k*.*

[8] If we only allow to improve upon whole sets of equally ranked candidates, we can use the improved version of the algorithm *DodScore* by treating the whole set as one possibility. In this way, we achieve a running time of $O(2^k \cdot nk + nm)$.

[9] Interestingly, Hemaspaandra et al. [11] have shown that the ranking and the winner versions remain Θ_2^p-complete in both cases considering ties—no differentiation in the classical complexity can be observed.

4 Young Score

In this section, we show that Young Score and Dual Young Score are W[2]-complete with respect to their corresponding solution size bounds l and k, respectively. We start with the proof of the W[2]-hardness of Dual Young Score where two parameterized reductions are given, the first from the W[2]-hard Red Blue Dominating Set (RBDS) [6] to an intermediate problem, a variant of Red Blue Dominating Set, and then the second one from the intermediate problem to Dual Young Score. Red Blue Dominating Set (RBDS) is defined as follows: Given a bipartite graph $G = (R \cup B, E)$ with R and B being the two disjoint vertex sets, $E \subseteq R \times B$ and an integer $k \geq 0$, the question is whether there is a subset $D \subseteq R$ of size at most k such that every vertex in B has at least one neighbor in D. The $k/2$-Red Blue Dominating Set ($k/2$-RBDS) problem has a bipartite graph $G = (R \cup B, E)$ with R and B being the two disjoint vertex sets, $E \subseteq R \times B$, and an integer $k \geq 0$ as input, and asks whether there is a subset $D \subseteq R$ of size at most k such that every vertex in B has at least $\lfloor k/2 \rfloor + 1$ neighbors in D.

Lemma 3. *$k/2$-RBDS is W[2]-hard.*

Next, we give a parameterized reduction from $k/2$-RBDS to Dual Young Score.

Lemma 4. Dual Young Score *is W[2]-hard.*

Proof. Given a $k/2$-RBDS-instance $(G = (B \cup R, E), k)$ with $B = \{b_1, ..., b_m\}$ and $R = \{r_1, ..., r_n\}$, we first consider the case that k is odd. The corresponding Dual Young Score instance is constructed as follows. For each blue vertex we add a candidate to the candidate set C. After that, three additional candidates a, b, and c are added to C, where c is the distinguished candidate of the Dual Young Score instance. Thus, $C = \{c_i \mid b_i \in B\} \cup \{a, b, c\}$.

Let $N_C(r_i) := \{c_j \in C \mid \{r_i, b_j\} \in E\}$ and $\overline{N_C(r_i)} := C \setminus (\{a, b, c\} \cup N_C(r_i))$, that is, the elements in $N_C(r_i)$ correspond to the neighbors of r_i in G and $\overline{N_C(r_i)}$ corresponds to the rest of the vertices in B. We construct three disjoint subsets of votes, V_1, V_2, and V_3.

- The votes in V_1 correspond to the red vertices in R. For every red vertex r_i, we add a vote v_i to V_1 by placing the candidates in $N_C(r_i) \cup \{a, b\}$ better than c and the candidates in $\overline{N_C(r_i)}$ worse than c, that is,

$$V_1 := \{\overline{N_C(r_i)} < c < N_C(r_i) < a < b \mid i = 1, \ldots, n\}.$$

  Note that, here and in the following, if we write a set in a vote, then the order of the elements in the set is irrelevant and can be fixed arbitrarily.
- The set V_2 also contains n votes which guarantee that the total deficit of each candidate is zero in $V_1 \cup V_2$. However, there is an exception with b which has deficit $2k - 2$ in $V_1 \cup V_2$.

$$V_2 := \{a < b < N_C(r_i) < c < \overline{N_C(r_i)} \mid i = 1, \ldots, n-k+1\}$$
$$\cup \{a < N_C(r_i) < c < \overline{N_C(r_i)} < b \mid i = n-k+2, \ldots, n\}.$$

Later, it will become clear that the $(2k-2)$-deficit of b will be used to argue that all votes in a solution of a Dual Young Score instance have to come from V_1.

– The set V_3 consists of $k-1$ votes to adjust the deficits of a and b so that in $V_1 \cup V_2 \cup V_3$ both a and b have a deficit of $k-1$ and all other candidates have a deficit of 0. Let $C_R := C \setminus \{a, b, c\}$. The set V_3 consists of $\lfloor k/2 \rfloor$ votes with $b < c < C_R < a$ and $\lfloor k/2 \rfloor$ votes with $b < C_R < c < a$.

Finally, the set V of votes is set to $V_1 \cup V_2 \cup V_3$ and the upper bound for the solution size of the Dual Young Score instance is equal to k. The key idea behind the above construction is that, to reduce the $(k-1)$-deficits of a and b by deleting at most k votes, all solutions of the Dual Young Score instance actually contain *exactly* k votes from V_1.

In the following, we show that the candidate c can become a Condorcet winner by deleting at most k votes iff there is a solution of size at most k for the $k/2$-RBDS-instance.

"⇒": We know that every solution V' of Dual Young Score contains exactly k votes in V_1 and, by the above construction, each vote in V_1 corresponds to a vertex in R. Denote the corresponding subset of R as D. Since V' is a solution, it must hold that every candidate $c_i \in (C \setminus \{a, b, c\})$ must be better than c in at least $\lfloor k/2 \rfloor + 1$ of the votes in V'. Therefore, choosing the corresponding red vertices to form a dominating set achieves that every blue vertex is dominated at least $\lfloor k/2 \rfloor + 1$ times.

"⇐": Since every dominating set $D \subseteq R$ of size at most k dominates each blue vertex at least $\lfloor k/2 \rfloor + 1$ times, we can easily extend D to a dominating set D' of size exactly k by adding $k - |D|$ arbitrary red vertices to D. Since every red vertex corresponds to a vote in V_1, we thus obtain a size-k subset V' of V corresponding to D'. According to the above construction of V_1, the removal of V' results in a new vote set where the deficits of a and b are both -1 and the deficits of all other candidates are at most -1. Therefore, c can become Condorcet winner by deleting exactly k votes.

Finally, we consider the case that k is even and give a reduction from Dual Young Score with an odd k to Dual Young Score with an even k. Given a Dual Young Score instance (V, C, c, k) with k being odd, we add a new vote v to V that has the form: $c < C \setminus \{c\}$ to get the new vote set V'. Then $(V', C, c, k' := k+1)$ is a Dual Young Score instance with k' being even. The correspondence between the solutions is easy to achieve. □

Next, we show that Dual Young Score is in W[2]. To this end, we can construct a parameterized reduction from Dual Young Score to the W[2]-complete Optimal Lobbying problem [4] also arising in the context of election

systems, which is defined as follows: Given an $n \times m$ 0/1-matrix M, a length-m 0/1-vector x, and an integer $k \geq 0$, the question is whether there is a choice of at most k rows from the rows of M such that the selected rows can be *edited* such that, if x has a 0 in its ith entry, then there are more 0's than 1's in the ith column of the matrix resulting after editing the selected rows; otherwise, there are more 1's than 0's in the ith column. By *editing* a row, we mean to change some 1's in the row to 0's and/or to change some 0's to 1's. We call x the *target* vector.

Lemma 5. Dual Young Score *is in W[2].*

Combining Lemmas 4 and 5, we arrive at the main result of this section.

Theorem 4. Dual Young Score *is W[2]-complete.*

Using a similar reduction as the one in the proof of Lemma 5 (containment in W[2]) and a slightly modified version of the reduction from the W[2]-hard Set Packing problem presented by Rothe et al. [18, Theorem 2.3] (W[2]-hardness), we can also show the following theorem.

Theorem 5. Young Score *is W[2]-complete.*

5 Conclusion and Outlook

Probably the most important general observation deriving from our work is that Dodgson and Young elections behave differently with respect to the parameter "number of editing operations". Whereas for Dodgson elections we achieve fixed-parameter tractability, we experience parameterized intractability in case of Young elections. This stands in sharp contrast to traditional complexity analysis, where both election systems appear as equally hard [1, 11, 18] and complements the results on polynomial-time approximability of Procaccia et al. [17]. Furthermore, we found that the complexities of Dodgson elections allowing ties between the candidates are much different (fixed-parameter tractability vs W[2]-hardness) depending on the cost model for switching ties. Again, in the standard complexity framework these two cases cannot be differentiated because both lead to NP-completeness.

We conclude with few specific open questions. Bartholdi et al. [1] gave an integer linear program which implies the fixed-parameter tractability of Dodgson Score with respect to the number of candidates (also see [13] for further results in this direction). Unfortunately, the corresponding running times are extremely high and an efficient combinatorial algorithm would be desirable. The parameterized complexity with respect to the parameter "number of votes" remains open. By way of contrast, there is a trivial $O(2^n \cdot \text{poly}(m, n))$-time algorithm for Young elections with $n := |V|$ and $m := |C|$. For the parameter "number of candidates", however, only an integer linear program implying (presumably impractical) fixed-parameter tractability is known [19].

Acknowledgement. We thank Jörg Vogel for various valuable pointers to the literature and inspiring discussions.

References

1. J. Bartholdi III, C. A. Tovey, and M. A. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare*, 6:157–165, 1989.
2. N. Betzler, M. R. Fellows, J. Guo, R. Niedermeier, and F. A. Rosamond. Fixed-parameter algorithms for Kemeny scores. In *Proc. of 4th AAIM*, volume 5034 of *LNCS*, pages 60–71. Springer, 2008.
3. Y. Chevaleyre, U. Endriss, J. Lang, and N. Maudet. A short introduction to computational social choice (invited paper). In *Proc. 33rd SOFSEM*, volume 4362 of *LNCS*, pages 51–69. Springer, 2007.
4. R. Christian, M. R. Fellows, F. A. Rosamond, and A. M. Slinko. On complexity of lobbying in multiple referenda. *Review of Economic Design*, 11(3):217–224, 2007.
5. 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.
6. R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.
7. P. Faliszewski, E. Hemaspaandra, L. A. Hemaspaandra, and J. Rothe. A richer understanding of the complexity of election systems. In S. Ravi and S. Shukla, editors, *Fundamental Problems in Computing: Essays in Honor of Professor Daniel J. Rosenkrantz*. Springer, 2008. To appear.
8. M. R. Fellows. Personal communication, October 2007.
9. J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.
10. E. Hemaspaandra and L. A. Hemaspaandra. Dichotomy for voting systems. *Journal of Computer and System Sciences*, 73(1):73–83, 2007.
11. E. Hemaspaandra, L. A. Hemaspaandra, and J. Rothe. Exact analysis of Dodgson elections: Lewis Caroll's 1876 voting system is complete for parallel access to NP. *Journal of the ACM*, 44(6):806–825, 1997.
12. C. M. Homan and L. A. Hemaspaandra. Guarantees for the success frequency of an algorithm for finding Dodgson-election winners. *Journal of Heuristics*, 2007.
13. J. C. McCabe-Dansted. Approximability and computational feasibility of Dodgson's rule. Master's thesis, University of Auckland, 2006.
14. J. C. McCabe-Dansted, G. Pritchard, and A. Slinko. Approximability of Dodgson's rule. *Social Choice and Welfare*, 2007.
15. I. McLean and A. Urken. *Classics of Social Choice*. University of Michigan Press, Ann Arbor, Michigan, 1995.
16. R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.
17. A. D. Procaccia, M. Feldman, and J. S. Rosenschein. Approximability and inapproximability of Dodgson and Young elections. Technical Report Discussion paper 466, Center for the Study of Rationality, Hebrew University, October 2007.
18. J. Rothe, H. Spakowski, and J. Vogel. Exact complexity of the winner problem for Young elections. *Theory of Computing Systems*, 36:375–386, 2003.
19. H. P. Young. Extending Condorcet's rule. *Journal of Economic Theory*, 16:335–353, 1977.