Parameterized Computational Complexity of Dodgson and Young Elections [1]

Nadja Betzler, Jiong Guo, and Rolf Niedermeier

Institut für Informatik, Friedrich-Schiller-Universität Jena, Ernst-Abbe-Platz 2, D-07743 Jena, Germany

Abstract

We show that the two NP-complete problems of Dodgson Score and Young Score have differing computational complexities when the winner is close to being a Condorcet winner. 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]-completeness results depending on the cost model for switching ties.

Key words: computational social choice, voting systems, winner determination, fixed-parameter tractability, W[2]-completeness

1 Introduction

Computational social choice and, more specifically, the computational complexity of election systems has become an increasingly important field of interdisciplinary research [4,11]. The analysis of election systems has applications

Email address: {nadja.betzler,jiong.guo,rolf.niedermeier}@uni-jena.de (Nadja Betzler, Jiong Guo, and Rolf Niedermeier).

[1] A preliminary version of this paper appears in *Proceedings of the 11th Scandinavian Workshop on Algorithm Theory (SWAT'08)*, Springer LNCS 5124, pp. 403-412. Supported by the Deutsche Forschungsgemeinschaft, Emmy Noether research group PIAF (fixed-parameter algorithms, NI 369/4), project DARE (data reduction and problem kernels, GU 1023/1), and project PAWS (parameterized algorithmics for voting systems, NI 369-10).

in computational politics; for example, preference aggregation via various election systems, and in multiagent systems when groups of software agents have to make a joint decision. Election systems play a central role in planning (artificial intelligence in general) and page ranking systems for Internet search engines.

We study the following classic scenario for election systems: a classic election system consists of a set of candidates and a set of vote(r)s. Each voter chooses an order of preference (total order) among the candidates. The well-known Condorcet principle from 1785 [6] 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 one exists, and, otherwise, 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 [22], one due to C. L. Dodgson [2] from 1876 [8] and one due to H. P. Young from 1977 [26]. 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 result in a Condorcet winner. In Young elections, the editing operation is to remove a vote, 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. [3] 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. [16] and later for Young elections Rothe et al. [25] 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. Faliszewski et al. [11] 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 conclusion is valid if analysis terminates once a problem has been shown to be NP-complete. However, classical analysis is only the beginning step for important computational problems. We propose the framework of parameterized computational complexity theory [10,13,24] for studying election systems, offering the next step towards finding an optimal

[2] Also known as the writer Lewis Carroll.

[3] In the meantime, there are many publications presenting classical complexity results for election systems, for example, see [7,15,17,23,27].

Table 1
Parameterized complexity of DODGSON SCORE and (DUAL) YOUNG SCORE with respect to different parameters. In case of fixed-parameter tractability we also give information about the (exponential terms of the) corresponding running times. Herein, "ILP" means that fixed-parameter tractability follows by an integer linear program and a result of Lenstra [19] implying impractical running times. Bold-faced results are new, the FPT-results for the parameter "number of candidates" can be directly obtained from [1,26], and the FPT-results for the Young elections with respect to the number of votes are trivial. Note that whereas the parameterized complexity of DODGSON SCORE with respect to the number of votes is open, it is solvable in polynomial time for a constant number of votes [1].

Parameter	DODGSON SCORE	DUAL YOUNG SCORE	YOUNG SCORE
# votes n	?	FPT (2^n)	FPT (2^n)
# candidates m	FPT (ILP)	FPT (ILP)	FPT (ILP)
# steps k	**FPT (2^k)**	$W[2]$**-complete**	$W[2]$**-complete**

solution. [4]

For Dodgson and Young elections, we consider the question of whether the NP-hard problems become fixed-parameter tractable (FPT) with respect to the parameter "number of editing operations". In parameterized complexity [10,13,24] the solution size typically is the standard parameter. Thus, we choose this standard parameterization 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. Having 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. Indeed, this "parameterized tractability gap" between Dodgson and Young elections is not completely surprising in the sense that in the case of Young the allowed editing operation is much more powerful than in the case of Dodgson. Our results imply that Dodgson elections can be put into actual use whenever the input instances are close to having a Condorcet winner. This answers an open question of Christian et al. [5] [5] and refutes a parameterized hardness conjecture of McCabe-

[4] We remark that in parallel work a parameterized complexity study has been initiated for Kemeny elections as well [2]. Therein, a number of positive algorithmic results are achieved.

[5] Fellows et al. independently showed that DODGSON SCORE is fixed-parameter tractable, but with a higher running time [12].

Dansted [20]. Other natural parameters in the context of voting systems are the "number of votes n" and the "number of candidates m". Regarding the parameter m, for both voting systems there are integer linear programs that imply fixed-parameter tractability [1,26]. Concerning the parameter n there is a trivial $2^n \cdot \text{poly}(m, n)$-time algorithm for Young elections. An overview of the parameterized complexity with respect to the different parameters is given in Table 1. Our results complement recent work on a simple greedy heuristic for finding Dodgson winners with a guaranteed frequency of success [18] and some work on the polynomial-time approximability of Dodgson and Young elections [3,21]. In particular, Caragiannis et al. [3] gave (randomized) approximation algorithms for Dodgson elections and showed 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.

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 three candidates a, b, c, the ordering $c > b > a$ means that candidate c is liked best and candidate a is liked least by this voter. In an election (V, C), a candidate $c \in C$ is called the *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. Observe that a Condorcet winner does not always exist. A *switch* is defined to be the swapping of two neighboring candidates in a vote. We now 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

[6] Note that we identify votes and voters.

the Condorcet winner c?

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?

All three problems are NP-complete [1,25].

Finally, we briefly introduce the relevant notions of parameterized complexity theory [10,13,24]. 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 a specific parameter. In our context, the studied parameters will be the numbers k and l of allowed edit operations. 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 [10] 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. It is commonly believed 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 if and only if (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 problem DODGSON SCORE parameterized by the score. This answers an open question of Christian et al. [5]. The algorithm not only decides 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.

An important tool to state the algorithms is the concept 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 *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. We call a candidate with a positive deficit *dirty*.

The following two observations are used for the design of the algorithm:

Observation 1. It is easy to see (McCabe-Dansted [20, Lemma 2.19]) that there is always an optimal solution that considers only switches such that every switch moves the distinguished c in a vote to a better position. Making use of this, our algorithm only considers switches of such kind.

Observation 2. 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. 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. For each such combination, the computation of an optimal solution achieving the partial decrements is based on the optimal solutions for the previously considered smaller subset. At the last vote, we can then construct an overall optimal solution based on the "partial" optimal solutions computed before.

In the following, we first describe a general version of the algorithm, which we also use to solve a generalized version of Dodgson Score (see Subsection 3.3). Then we show how to further improve the running time of this algorithm for Dodgson Score.

3.1 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 an integer equal to the minimum number of switches within the votes $\{v_j \mid 1 \leq j \leq i\}$ such that in a resulting instance 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 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 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 best(S, v_i) denote the candidate c_j with $j \in S$ such that c_j is liked better than each other candidate in $\{c_r \mid r \in S, r \neq j\}$ in vote v_i.

3.2 Algorithm

The dynamic programming algorithm for Dodgson Score is given in Figure 1. We assume that we already have the deficits of the candidates and that 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 an integer equal to the minimum number of switches needed for each of them. In the update (lines 7–11), the subset of votes $\{v_1, \ldots, v_{i-1}\}$ 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 T.

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

PROOF. Concerning the correctness of the initialization, note that the first for-loop (lines 1–3) merely sets all table entries to "$+\infty$". Hence it suffices to show that *DodScore* assigns the correct number of switches to all entries of

[7] Using "at most" in the 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, deficit list $D = (d_1, \ldots, d_p)$ of dirty candidates, positive integer k with $\sum_{i=1}^{p} d_i \leq k$
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 for each $j \in S$ candidate c_j defeats c in v_1 then
06 $T(v_1, D - S) := \text{switch}(v_1, \text{best}(S, v_1))$
Update:
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 for each $j \in S$ candidate c_j defeats c in v_i then
11 $T(v_i, D') := \min\{T(v_i, D'), \ T(v_{i-1}, D' + S) + \text{switch}(v_i, \text{best}(S, v_i))\}$
Output:
12 if $T(v_n, (0, 0, \ldots, 0)) \leq k$ then
13 return "Yes"

Fig. 1. Algorithm for Dodgson Score

the first row with partial deficit lists that can be achieved by switching within the first vote v_1 (lines 4–6). Since in one vote the deficit of every candidate can be reduced by at most one, it is sufficient to iterate over all possible subsets S of $\{1, \ldots, p\}$ and to reduce the original deficits of the corresponding candidates by one. Thereby, an entry can only become less than $+\infty$ if c can be improved upon all candidates with indices in S (line 5). Moreover, the minimum number of switches is obviously the number of switches needed to improve c upon the candidate that is best in vote v_1 among the candidates c_j with $j \in S$, which is given by $\text{switch}(v_1, \text{best}(S, v_1))$.

The computation of an entry $T(v_i, D')$ with $i \geq 2$ is based on the fact that the decrement from D to D' can be split into two parts. One part needs to be achieved by switches in vote v_i and the other one by switches in votes $v_1, \ldots, v_{i-1}$. The minimum number of switches needed for the corresponding splitting possibilities is stored in $T(v_i, D')$. Moreover, since the switches in v_i can decrease the deficit of one dirty candidate by at most one, every possible way of splitting the deficit decrement can be represented by a subset S of $\{1, \ldots, p\}$. Each subset S has the meaning that, by the switches in v_i, the deficits of the dirty candidates with indices in S should be decreased by exactly one; the rest of the decrement from D to D' has to be achieved by switches in $v_1, \ldots, v_{i-1}$. According to the definition of the table T, the minimum number of switches to achieve the latter is stored in the already

computed $(i-1)$th row of T, namely, in $T(v_{i-1}, D' + S)$. As argued for the initialization, switch$(v_i, \text{best}(S, v_i))$ returns the minimum number of switches to decrease the deficit of the candidates with indices in S. Therefore, lines 9-11 of *DodScore* compute $T(v_i, D')$ correctly.

Since *DodScore* computes the table T correctly, we can conclude that a given instance is a yes-instance if and only if $T(v_n, (0, \ldots, 0)) \leq k$ (lines 12 and 13). □

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

PROOF. 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 of T 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 compared to Lemma 2 is achieved by replacing the innermost for-loop (lines 9–11 in Figure 1) of the update step which computes a table entry and needs $O(2^k \cdot k)$ time by an instruction running in time linear in k.

For $d \in C \backslash \{c\}$, 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. Clearly, $S_i(d)$ is empty if d is worse than c in v_i and, otherwise, $S_i(d)$ contains d. We replace lines 9–11 in Figure 1 by the 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 $\text{best}(S, v_i) = c_r$. Thus,

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

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 $\text{best}(S, v_i) = c_r$ and, thus, $\text{switch}(v_i, \text{best}(S, v_i)) = \text{switch}(v_i, c_r)$. Moreover, from the definition of table entries, the following monotonicity of the table T 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 $T(v_i, D' + S) \ge T(v_i, D' + S_i(c_r))$. Clearly, $S_i(c_r) \subseteq \{1, \ldots, p\}$ and, by definition, $S_i(c_r)$ satisfies the if-condition in line 10. It follows that

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

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 $O(2^k \cdot nk + nm)$. □

3.3 Allowing Ties

Sometimes it might be desirable to allow a voter to rank two or more candidates equally. This leads to an election based on votes with ties. As noted by

Hemaspaandra et al. [16], there are (at least) two different natural models on how to generalize Dodgson Score to the case with ties. The models differ in the "power" of one switch. In the first model, transforming $a = b > c$ into $c > a = b$ requires just one switch and in the second model this requires two separate switches. The ranking and the winner versions remain Θ_2^p-complete in both cases [16].

Formally, a vote with ties can be considered as a total order of disjoint sets of candidates. To ease the presentation, we often write just "$> c >$" instead of "$> \{c\} >$".

Recall that in the case of ties a candidate c is a Condorcet winner if for every other candidate d the number of votes in which c is strictly preferred to d is higher than the number of votes in which d is strictly preferred to c. Hence, the deficit of a candidate $d \neq c$ is defined as $N_d - N_{\overline{d}} + 1$, 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. In the following, we describe two switch operations, one for each model. In both models a switch can now either break or build ties between the distinguished candidate c and other candidates.

For computing the Dodgson score only the relative order between the distinguished candidate c and the other candidates is relevant. Hence, to keep the models easy, we restrict them to the interesting case where each switch involves the distinguished candidate.

In the first model the distinguished candidate can improve upon a whole subset of candidates by one switch. More precisely, for an appropriate subset $B \subseteq C \backslash \{c\}$, we have one of the following two situations:

- "$\ldots > B > c > \ldots$": Such a vote can be transformed to "$\ldots > B \cup \{c\} > \ldots$" by applying one switch.
- "$\ldots > B \cup \{c\} > \ldots$": Such a vote can be transformed to "$\ldots > c > B > \ldots$" by applying one switch.

The problem of computing the Dodgson score for this model is denoted as Dodgson Tie Score 1 (DTS1).

In the second model, the switch operation becomes less powerful, that is, the distinguished candidate can only improve upon one candidate by one switch. Here, one has to consider the following situations:

- "$\ldots > B > c > \ldots$": Such a vote can be transformed to "$\ldots > B \backslash B' > B' \cup \{c\} > \ldots$" by $|B'|$ switches for any $B' \subseteq B$.
- "$\ldots > B \cup \{c\} > \ldots$": Such a vote can be transformed to "$\ldots > (B \backslash B') \cup \{c\} > B' \ldots$" by $|B'|$ switches for any $B' \subseteq B$.

The problem of computing the Dodgson score for this model is denoted as Dodgson Tie Score 2 (DTS2). The considered model is very general in the sense that it allows to choose to improve the distinguished canidate only upon a subset of equally ranked candidates and thus it is only "charged" to pay for this subset. A reasonable special case of this model is to restrict B' to be idential with B, that is, to allow only to switch the distinguished candidate with the whole subset. For this case, we can directly use the improved version of the algorithm *DodScore* as described in the proof of Theorem 1 by treating the whole set of tied candidates as one possibility. This yields an algorithm with running time $O(2^k \cdot nk + nm)$.

Note that for both models, a switch as defined for the case without ties can be simulated by two switches.

Whereas 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 while DTS1 is W[2]-complete.

To show the fixed-parameter tractability of DTS2 one can use a slight modification of algorithm *DodScore* from Figure 1. However, for DTS2 we obtain a slightly worse running time than for Dodgson Score without ties as given in Theorem 1. Since we do not have a total ordering of candidates in the votes, we cannot make use of the monotonicity property employed in the proof of Theorem 1, thus, returning to the algorithm used for Lemma 1.

Theorem 2 Dodgson Tie Score 2 *can be solved in* $O(6^k \cdot nk + nm)$ *time.*

PROOF. We use a slight modification of algorithm *DodScore* (Figure 1) to solve DTS2. In the DTS2 model, one switch can decrease the deficit of at most one candidate by at most one. Thus, it holds that the sum of all deficits is bounded by the number of total switches. In lines *05* and *10* of Figure 1, we replace the phrase "c_j defeats c" by "c does not defeat c_j" to include the possibility that c and c_j are tied. Next, we redefine the switch function for the update and initialization steps (used in lines *06* and *11* of Figure 1). Let $v_i = S_1 > S_2 > \cdots > S_{r-1} > S_r \cup \{c\} > \ldots$ denote an arbitrary vote from V. Consider a non-empty subset $S \subseteq \{1, \ldots, p\}$ in the computation of the entry $T(v_i, D')$. The set S has non-empty intersections with some of the candidate subsets $S_j, j \in \{1, \ldots, r\}$. Let S_b denote the subset $S_j, j \in \{1, \ldots, r\}$, with smallest index such that $U := S_j \cap S$ is not empty. We distinguish two cases. First, $b = r$. In this case, we only have to break the ties of c with all candidates from U. Hence, in this case, we can define

$$\text{switch}(v_i, S) := |U|$$

and do not need any further modifications. The second case is $b < r$. Here, we also need to decrease the deficit of all candidates of S by at least one. The way to achieve this with a minimum number of switches is to switch c so far that it is tied with all candidates from U. This requires one switch for all candidates in $U \cup S_r$ and two switches for all candidates from $\bigcup_{b<j<r} S_j$. Hence, in this case, we set

$$\text{switch}(v_i, S) := |U| + |S_r| + 2 \cdot |\bigcup_{b<j<r} S_j|.$$

Further, in this case we have to adapt the definition of $D + S$ used in line *10* such that one adds two for all candidates from $\bigcup_{b<j<r} S_j$. For the initialization, one has to adapt the definition of $D - S$ appropriately as well (used in line *06*). To have a sufficient initialization of the dynamic programming table, here we initialize all table entries corresponding to candidates from $\bigcup_{b<j<r} S_j$ with decrease of the initial deficit by all combinations of one and two. It may happen that in an optimal solution the distinguished candidate is further improved upon a subset of S_r in v_i. This can be realized by trying all subsets of S_r and update the corresponding table entry. Altoghether, the computation of an table entry $T(v_i, D)$ can be done in $O(3^k \dot{k})$ time. Trying out all possible subsets $S \subseteq \{1, \ldots, p\}$ and appropiate subsets S_r, we obviously consider all possible positionings of c after some switches in a vote. Since we take the minimum number of switches over all these possibilities, the modified *DodScore* algorithm works correctly. Since Observation 1 and Observation 2 still hold, the running time bound follows similar to the proof of Lemma 2; taking into account that an update step needs $3^k \cdot k$ time. □

In contrast to the fixed-parameter tractability of DTS2, DTS1 is W[2]-complete. Intuitively, this may be explained by the fact that in case of DTS1 a single edit operation can improve the distinguished candidate c upon, in principle, *all* other candidates.

Lemma 3 Dodgson Tie Score 1 *is W[2]-hard with respect to the parameter k.*

PROOF. We employ a parameterized reduction from the W[2]-complete Dominating Set problem [10]: Given an undirected graph $G = (W, E)$ and a positive integer k, the task is to decide whether there exists a dominating set of size k, that is, a subset of vertices $W' \subseteq W$ with $|W'| \leq k$ such that every vertex of W either belongs to W' or has at least one neighbor in it.

The basic idea for the reduction is as follows. We associate a candidate with every vertex of the graph. In a first set of votes, for every vertex i of the graph we construct a vote such that the candidates that correspond to the vertices of the closed neighborhood of i are positioned better than the distinguished

candidate c; these vertices are ranked equally such that one can build a tie between c and all of them by one switch. Then, we add a second set of votes of same size in which c cannot improve upon any of the "vertex candidates" with at most k switches. This is done by inserting dummy candidates. That is, in total we achieve that for every vertex candidate the number of votes in which it is worse than c equals the number of votes in which it is better than c. Then, the votes of the first set that are affected by switches correspond to the vertices of a dominating set.

In the following, we give the formal construction. Given a Dominating Set instance $(G = (W, E), k)$, we construct a DTS1-instance (V, C, c, k) as follows: The set of candidates consists of one candidate for every vertex, the distinguished candidate c, and $2k$ additional dummy candidates, that is, $C := \{c_i \mid 1 \leq i \leq |W|\} \cup \{c\} \cup \{f_j \mid 1 \leq j \leq 2k\}$. We denote the subset of candidates that corresponds to the closed neighborhood of a vertex i by

$$N_C[i] := \{c_j \mid \{i, j\} \in E\} \cup \{c_i\},$$

and, the set of remaining vertex candidates by

$$\overline{N_C[i]} := C \setminus (N_C[i] \cup \{c\} \cup \{f_j \mid 1 \leq j \leq 2k\}).$$

We define the vote set $V := V_1 \cup V_2$ as follows:

$$\begin{aligned}
V_1 :=& \{N_C[i] > c > \overline{N_C[i]} > f_{2k} > \cdots > f_1 \mid 1 \leq i \leq |W|\} \text{ and} \\
V_2 :=& \{\overline{\overline{N_C[i]} > f_k > \cdots > f_1 > c > f_{2k} > \cdots > f_{k+1} > N_C[i]} \mid \\
& \quad 1 \leq i \leq |W| - 1\} \\
& \quad \cup \{\overline{N_C[|W|]} > f_{2k} > \cdots > f_{k+1} > c > f_k > \cdots > f_1 > N_C[|W|]\}.
\end{aligned}$$

Clearly, c already defeats all of the dummy candidates $f_1, \ldots, f_{2k}$. In addition, consider an arbitrary vertex candidate c_i. For every vote from V_1 in which c_i is better (worse) than c there is a vote from V_2 in which c_i is worse (better) than c. That is, we have to decrease the deficit of every vertex candidate by at least one.

We show that the distinguished candidate c can become the Condorcet winner by k switches if and only if the Dominating Set instance has a solution of size k.

"$\Leftarrow$": Given a dominating set W' of size k, we claim that applying exactly one switch to each vote of V_1 that has its corresponding vertex in W' is a solution for the DTS1 instance. This claim is correct since W' is a dominating set and one switch in a vote $v_i \in V_1$ leads to a tie with c and the whole closed neighborhood of vertex i. Therefore, the deficits of all candidates are decreased to at most zero after these switches.

"⇒": Given a solution S using k switches for the DTS1 instance, we can assume that the votes in V_2 are not affected by the switches in S. This assumption is justified due to the fact that, by switching at most k times even within one vote in V_2, one can only improve upon dummy candidates. Furthermore, one switch within a vote in V_1 decreases the deficits of all candidates that are better than c in this vote by one, since all these candidates are ranked equally. Since in every vote from V_1 one can apply at most one switch, S affects exactly k votes from V_1. Since all vertex candidates have deficit 1, each of them appears at least once better than c in the votes affected by S. Due to the above construction, the set of vertices $\{i \mid v_i \in S\}$ is obviously a dominating set of size k in G. □

So far, we have proven the W[2]-hardness of DTS1. It remains to show its containment in W[2].

Lemma 4 Dodgson Tie Score *is contained in W[2] with respect to the parameter k.*

PROOF. By the fact that the number of switches is a lower bound for the total switch cost, it suffices to show containment in W[2] with respect to the number of switches as the parameter. More precisely, we show containment in W[2] by giving a parameterized reduction to a variant of the k-Weighted Circuit Satisfiability (k-WCS) problem that defines W[2] [9]. The k-WCS problem has as input a circuit and a positive integer k, and asks whether it has a weight-k satisfying assignment (an assignment setting the values of exactly k input gates to 1). Here, we use that a parameterized problem is in W[2] if it is parameterized reducible to k-WCS restricted to circuits satisfying the following two conditions [9]:

(1) On every input-output path, the number of gates with unbounded fan-in is at most two.
(2) The length of the longest input-output path is bounded by a function only depending on the parameter.

Given an election (V, C), a distinguished candidate c, and a positive integer k, we construct a corresponding Boolean expression E as follows. Intuitively, the variables represent the positions that candidate c can take by switching within the set of votes. Let e_i denote the number of switches needed such that c becomes the best ranked candidate in vote v_i. Having a set of equally ranked candidates, c can improve with one switch upon all of them, and, in this case, the "position" of c is increased by one. In addition, we have k copies of every variable that corresponds to a position in a vote. Thus, the set of variables for E is

$$P = \{p[i,j,s] \mid 1 \leq i \leq n,\ 1 \leq j \leq e_i,\ \text{and } 1 \leq s \leq k\}$$

A variable $p[i,j,s]$ is true means that the information that in vote v_i one has switched to position j is stored in copy s. For a vote v_i, we use $C(i,j)$ to denote the set of candidates upon which c improves with the jth switch in v_i.

We construct $E = E_1 \wedge E_2 \wedge E_3 \wedge E_4$ as follows:

(1) The expression E_1 ensures that in each of the k copies of the positions variables exactly one position variable is true in a satisfying assignment:

$$E_1 := \bigwedge_{s=1}^{k} \bigwedge_{\substack{i \neq i' \text{ or} \\ j \neq j'}} \Big((\neg p[i,j,s] \wedge p[i',j',s]) \vee (p[i,j,s] \wedge \neg p[i',j',s]) \Big).$$

Herein, the second "$\bigwedge$" quantifies over all $i, i' \in \{1,\ldots,n\}$ and the corresponding $j, j' \in \{1,\ldots,e_i\}$.

(2) The expression E_2 ensures that at most one copy of a variable can be true in a satisfying assignment:

$$E_2 := \bigwedge_{i,j} \bigwedge_{s' \neq s} \neg(p[i,j,s] \wedge p[i,j,s'])$$

(3) To switch the distinguished candidate c for the j-th time in vote v_i, we must have switched it $j-1$ times in v_i before. This is simulated by the expression E_3 as follows:

$$E_3 := \bigwedge_{\substack{i,j,s \\ j>1}} (\neg p[i,j,s] \vee \bigvee_{s' \in \{1,\ldots,k\}} p[i,j-1,s'])$$

(4) The most important part of the construction is to ensure that the deficit of every candidate is decreased to zero or below zero. For any candidate $c_x \in C$, let d_x denote its deficit. Then, expression E_4 ensures that at least d_x variables $p[i,j,s]$ with $c_x \in C(i,j)$ must be set to 1. Let $S(k,r)$ denote the set of all size-r subsets of $\{1,\ldots,k\}$.

$$E_4 := \bigwedge_{c_x \in C} \bigvee_{S' \in S(k,d_x)} \bigwedge_{s \in S'} \bigvee_{\substack{i,j \text{ with} \\ c_x \in C(i,j)}} p[i,j,s].$$

In E_4, the two innermost quantifications range over sets whose sizes are bounded by functions depending only on k. Therefore, the Boolean expression E can be easily transformed into conjunctive normal form (as required for the W[2]-characterization [9]), making use of the fact that the middle Or- and And-quantifiers in E_4 can be equivalently replaced by a Or- and And-quantifiers with changed order.

Let the weight of a truth assignment denote the number of variables set to 1. The following claim completes the proof.

Claim: Formula E is satisfiable by a weight-k satisfying truth assignment if and only if c can be made a Condorcet winner by applying k switches.

"$\Leftarrow$": Given a set of k switches that make c a Condorcet winner, construct a weight-k truth assignment as follows.

Number the switches from 1 to k and then set the "corresponding" variable to true. That is, if the switch with number q brings candidate c in vote i into position j, then set $p[i, j, q] :=$ true. The remaining variables with $s = q$ are set to false.

As for every q with $1 \leq q \leq k$ we set exactly one variable $p[i, j, q]$ to true, the expression E_1 is true. Since every two distinct switches operate on distinct votes or they move the distinguished candidate to distinct positions in the same vote, the expression E_2 is true. As the switches within one vote have to be contiguous, E_3 is true. Finally, for candidate c_x we must have that c improves upon c_x in at least d_x votes. For this reason, there must be a size-d_x subset of switches with $c_x \in C(i, j)$, and E_4 is true.

"$\Rightarrow$": Given a weight-k satisfying truth assignment for E, construct a solution for DTS1 by choosing the corresponding switches, that is, for every variable $p[i, j, s]$ that is true move c in vote v_i to position j. The expression E_1 ensures that we apply exactly k switches, E_2 ensures that every switch is made only once, and E_3 ensures that in one vote all switches are contiguous. Finally, E_4 makes sure that for every candidate the deficit is sufficiently reduced. □

Combining Lemmas 3 and 4, one arrives at the following.

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

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. From a parameterized perspective Dual Young Score appears to be more natural than Young Score because for Dual Young Score one may expect smaller parameter values.

For both problems, similar to Dodgson Score, it is helpful to consider a deficit concept for 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 *Young deficit* is defined as $N_d - (n - N_d)$.

We start with a W[2]-hardness-proof for Dual Young Score, giving two parameterized reductions: The first reduction is from the W[2]-hard Red Blue Dominating Set (RBDS) [10] to an intermediate problem, which is a variant of Red Blue Dominating Set, and then the second one is from the intermediate problem to Dual Young Score.

Red Blue Dominating Set (RBDS)
Given: A bipartite graph $G = (R \cup B, E)$, with R and B being the two disjoint vertex sets, and an integer $k \geq 0$.
Question: Is there a subset $D \subseteq R$ of size at most k such that every vertex in B has at least one neighbor in D?

The intermediate problem is defined as follows:

$k/2$-Red Blue Dominating Set ($k/2$-RBDS)
Given: A bipartite graph $G = (R \cup B, E)$, with R and B being the two disjoint vertex sets, and an integer $k \geq 0$.
Question: Is there 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 5 *$k/2$-Red Blue Dominating Set is W[2]-hard.*

PROOF. We give a parameterized reduction from RBDS. For an RBDS instance $(G = (B \cup R, E), k)$, the corresponding instance $(G' = (B' \cup R', E'), k')$ of $k/2$-RBDS is constructed as follows:

$$
\begin{aligned}
B' &:= B \cup \{b_x\},\\
R' &:= R \cup \{r_j^{\text{new}} \mid 1 \leq j \leq k\} \cup \{r_x\},\\
E' &:= E \cup \{\{b, r_j^{\text{new}}\} \mid b \in B \text{ and } 1 \leq j \leq k\}\\
&\quad \cup \{\{b_x, r_x\}\} \cup \{\{b_x, r_j^{\text{new}}\} \mid 1 \leq j \leq k\}, \text{ and}\\
k' &:= 2k + 1.
\end{aligned}
$$

The following claim finishes the proof.

Claim: The considered RBDS-instance is a yes-instance if and only if the $k/2$-RBDS-instance is a yes-instance.

"$\Rightarrow$": One can easily construct a solution for the $k/2$-RBDS-instance by choos-

ing the corresponding vertices of the size-$\leq k$ RBDS-solution D and additionally the $k+1$ new red vertices. The size of the new solution then is at most $2k+1$ and every blue vertex in B is dominated k times by the new red vertices and at least once by a vertex from D. The new blue vertex b_x is dominated by the $k+1$ new red vertices. Therefore, every vertex is dominated at least $k+1 = \lfloor k'/2 \rfloor + 1$ times.

"$\Leftarrow$": Consider a size-$\leq k'$ solution D of the $k/2$-RBDS-instance. Obviously, D must contain r_x and the other k new red vertices $r_1^{\text{new}}, \ldots, r_k^{\text{new}}$ to dominate b_x. Therefore, all of the other blue vertices are dominated exactly k times by $r_1^{\text{new}}, \ldots, r_k^{\text{new}}$. Since every blue vertex has to be dominated at least $\lfloor k'/2 \rfloor + 1 = k+1$ times, the vertices in $D \setminus \{r_x, r_1^{\text{new}}, \ldots, r_k^{\text{new}}\}$ dominate all other blue vertices in $B' \setminus \{b_x\} = B$ and, thus, the subset of R corresponding to $D \setminus \{r_x, r_1^{\text{new}}, \ldots, r_k^{\text{new}}\}$ is a size-$\leq k$ solution of the RBDS-instance. □

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

Lemma 6 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. We set $C := \{c_i \mid b_i \in B\} \cup \{a, b, c\}$.

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 candidates 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. 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, add a vote v_i to V_1 in which the candidates in $N_C(r_i) \cup \{a, b\}$ are better than c and the candidates in $\overline{N_C(r_i)}$ are worse than c. More precisely,

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

  Note that, here and in the following, if there is 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. These votes guarantee that in $V_1 \cup V_2$ the deficit of b is $2k-2$ whereas the deficit of each other candidate is zero.

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

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 $a > C_R > c > b$ and $\lfloor k/2 \rfloor$ votes with $a > c > C_R > b$.

Finally, the overall set V of votes is $V_1 \cup V_2 \cup V_3$ and the upper bound for the solution size of the Dual Young Score instance is set 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. The reason for this is that the votes in V_1 are the only votes whose deletion simultaneously reduces the deficits of a and b against c.

In the following, we show that c can become the Condorcet winner by deleting at most k votes if and only if there is a dominating set of size at most k for the (G, k).

"$\Rightarrow$": Every solution V' of Dual Young Score must contain exactly k votes from V_1 and, by the above construction, each vote in V_1 corresponds to a vertex in R. Denote the corresponding subset of R by D. Since V' is a solution, 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 implies that every blue vertex is dominated at least $\lfloor k/2 \rfloor + 1$ times.

"$\Leftarrow$": 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 from 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 ≤ -1. Therefore, c can become the Condorcet winner by deleting exactly k votes.

Recall that in the definition of V_3 it is decisive that k is odd. Now, 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: "$C \setminus \{c\} > c$" 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. □

To prove that DUAL YOUNG SCORE is W[2]-complete, it remains to show its containment in W[2]. To this end, we construct a parameterized reduction from DUAL YOUNG SCORE to the following W[2]-complete problem [5] also arising in the context of election systems:

OPTIMAL LOBBYING
Given: An $n \times m$ 0/1-matrix M, a length-m 0/1-vector x, and an integer $k \geq 0$.
Question: Is there a choice of at most k rows from M such that the selected rows can be *edited* in a way that, in the resulting matrix, it holds that if x has a 0 in its ith entry, then there are more 0's than 1's in the ith column, and if x has a 1 in its ith entry, then 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 7 DUAL YOUNG SCORE *is in W[2].*

PROOF. We give a parameterized reduction to OPTIMAL LOBBYING. We focus on instances (V, C, c, k) of DUAL YOUNG SCORE with $n - 3k > 0$ where $n := |V|$. The instances with $n - 3k \leq 0$ can be trivially solved by enumerating all size-$\leq k$ subsets of V, which can be done in $O(\binom{3k}{k} \cdot |C|)$ time. We rename the candidates such that $C = (c_1, c_2, \ldots, c_{m-1}, c_m)$ with $c_m = c$.

The matrix of the OPTIMAL LOBBYING instance consists of two submatrices. First, observe that in the DUAL YOUNG SCORE instance the exact ordering of the candidates in a vote is irrelevant, the key point here concerns the relative positions of the candidates compared to the distinguished candidate c. Therefore, one can easily transform a vote into a length-$(m-1)$ 0/1-vector where each candidate from $C \setminus \{c\}$ has an entry. If a candidate c_i for $1 \leq i \leq m-1$ is better than c in the vote, then set the ith entry of the vector to 0; otherwise, set it to 1. Putting all these binary vectors together, one obtains an $n \times (m-1)$ 0/1-matrix. Adding an all-0 column to this matrix as the mth column results in the first submatrix M_1 of the OPTIMAL LOBBYING instance to be constructed.

Second, construct a size-$(n-2k+1) \times m$ matrix M_2 as follows. Set all entries of the mth column of M_2 to 1's. In each of the other $m-1$ columns, set the first $\lceil (n-3k)/2 \rceil$ entries to 1's and the rest to 0's.

Finally, combine M_1 and M_2 by putting them on top of each other. The target vector x is set to an all-1 vector. Moreover, set the solution size bound of the Optimal Lobbying instance equal to k. It remains to show the equivalence between the solutions of the original Dual Young Score instance and the constructed Optimal Lobbying instance.

"$\Rightarrow$": Let S be a size-k solution of Dual Young Score. Choose the set S' of corresponding rows from M_1. The claim is that S' is a solution of the Optimal Lobbying instance. This means that, after editing all 0's in the rows from S' to 1's, every column of the resulting matrix has more 1's than 0's, "justifying" the target vector. Obviously, the claim is true for the mth column, since the mth column has $n-2k+1$ 1-entries in M_2 and only 0-entries in M_1. In particular, the rows from S' have 0's in the mth column. Therefore, after editing the 0's in the rows from S', one gets exactly $n-k+1$ 1's and $n-k$ 0's in the mth column.

Next, consider the ith column with $1 \leq i \leq m-1$. For a submatrix M' of M let $\#_1(M'[i])$ denote the number of 1's in the ith column of M' and let $\#_0(M'[i])$ denote the number of 0's in the ith column of M'. Let $M_1 \backslash S'$ denote the submatrix M_1 without the rows of S'.

As the distinguished candidate c is a Condorcet winner after deleting S, it must be better than c_i in more than half of the votes of $V \backslash S$. Thus, by construction, we have

$$\#_1(M_1 \backslash S'[i]) \geq \left\lfloor \frac{n-k}{2} \right\rfloor + 1.$$

Moreover, we additionally get k 1-entries by the edited entries of S' and in M_2 we have $\lceil (n-3k/2) \rceil$ 1-entries. Thus, for the total number of 1's in the matrix M'' resulting from editing S' in M, one gets that

$$\#_1(M''[i]) \geq \left\lfloor \frac{n-k}{2} \right\rfloor + 1 + k + \left\lceil \frac{n-3k}{2} \right\rceil = n-k+1.$$

Since the number of rows of the matrix M'' is $2n-2k+1$, one gets that

$$\#_0(M''[i]) = 2n-2k+1-(n-k+1) = n-k < \#_1(M[i]).$$

Hence, the target vector $(1, \ldots, 1)$ is realized and editing S' in the considered way gives a solution for the Optimal Lobbying instance.

"$\Leftarrow$": In order to validate the all-1 target vector for the mth column, all size-k solutions S' of the Optimal Lobbying instance contain only rows from M_1. Hence, one can easily construct a size-k subset $S \subseteq V$ by choosing the corresponding votes. To show that S is a solution of the Dual Young

SCORE instance, one needs to show that for every candidate c_i with $1 \leq i \leq m-1$, in $V' := V \backslash S$ the distinguished candidate c is better than c_i in at least $\lfloor (n-k)/2 \rfloor + 1$ votes, that is, in $M_1 \backslash S'$ there must be at least $\lfloor (n-k)/2 \rfloor + 1$ entries of the i'th column that are set to 1.

Assume that $\#_1(M_1 \backslash S'[i]) \leq \lfloor (n-k)/2 \rfloor$. By editing the entries of S', the number of rows which contain 1's can be increased at most by k. Thus, in total,

$$\#_1(M''[i]) \leq \left\lfloor \frac{n-k}{2} \right\rfloor + k + \left\lceil \frac{n-3k}{2} \right\rceil = n-k,$$

where M'' is the matrix resulting from editing S' in M.

Since, there are $2n-2k+1$ rows in total, $\#_0(M''[i])$ must be at least $n-k+1$, and, therefore, $\#_0(M''[i]) > \#_1(M''[i])$. As a consequence, S' cannot be a solution for the OPTIMAL LOBBYING instance with target vector $(1, \ldots, 1)$. □

Combining Lemmas 6 and 7, 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 7 (containment in W[2]) and a parameterized version of the non-parameterized reduction from the W[2]-hard SET PACKING problem to YOUNG SCORE as presented by Rothe et al. [25, Theorem 2.3] (W[2]-hardness), we can also derive 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 [1,16,25], and complements results on polynomial-time approximability [3]. Furthermore, we found that the complexities of Dodgson elections allowing ties between the candidates strongly vary (fixed-parameter tractability vs W[2]-completeness) 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 some specific open questions directly arising from our work. Regarding Dodgson Score, it would be interesting to investigate the power of polynomial-time data reduction and the existence of (small) problem kernels (see [14] for a survey on problem kernels which play a key role in parameterized algorithmics). Moreover, there are open questions regarding other parameterizations (see also Table 1): Bartholdi et al. [1] gave an integer linear program which implies the fixed-parameter tractability of Dodgson Score with respect to the parameter "number of candidates" (also see [20] for further results in this direction). Unfortunately, the corresponding running times are extremely high and a more efficient combinatorial algorithm would be desirable; the same holds for Young Score. The parameterized complexity of Dodgson Score with respect to the parameter "number of votes" remains open—we conjecture W[1]-hardness based on some preliminary evidence. Finally, it would be interesting to study typical values of Dodgson scores under reasonable distributions for preference profiles. Thus, a subject of study could be to find out what the probability of having a candidate with "low" Dodgson score is. This would help to better assess the practical potential of the described fixed-parameter algorithm for Dodgson Score.

Acknowledgement. We thank Jörg Vogel (Jena) for various valuable pointers to the literature and inspiring discussions. Moreover, we are grateful to three anonymous referees of *Information and Computation* whose careful and insightful reports helped to significantly improve the presentation and clarify inconsistencies.

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, 3, 4, 5, 12, 23, 24

[2] N. Betzler, M. R. Fellows, J. Guo, R. Niedermeier, and F. A. Rosamond. Fixed-parameter algorithms for Kemeny rankings. *Theoretical Computer Science*, 2009. To appear. 3

[3] I. Caragiannis, J. A. Covey, M. Feldmann, C. M. Homan, C. Kaklamanis, N. Kramikolas, A. D. Procaccia, and J. S. Rosenschein. On the approximability of Dodgson and Young elections. In *Proc. of 20th SODA*, pages 1058–1067. SIAM, 2009. 4, 23

[4] 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. 1

[5] 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. 3, 5, 21

[6] M. Condorcet. Essai sur l'application de l'analyse á la probabilité des décisions rendues à la pluraliste des voix. Facsimile reprint of the original published in Paris, 1785, by the Imprimerie Royale, 1972. 2

[7] 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. 2

[8] C. Dodgson. A method of taking votes on more than two issues. Pamphlet printed by the Clarendon Press, Oxford, and headed "not yet published", 1876. 2

[9] R. G. Downey and M. R. Fellows. Threshold dominating sets and an improved version of W[2]. *Theoretical Computer Science*, 209:123–140, 1998. 15, 16

[10] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999. 2, 3, 5, 13, 18

[11] 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*, chapter 14, pages 375–406. Springer, 2008. 1, 2

[12] M. R. Fellows. Personal communication, October 2007. 3

[13] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006. 2, 3, 5

[14] J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38(1):31–45, 2007. 24

[15] E. Hemaspaandra and L. A. Hemaspaandra. Dichotomy for voting systems. *Journal of Computer and System Sciences*, 73(1):73–83, 2007. 2

[16] 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. 2, 10, 11, 23

[17] E. Hemaspaandra, L. A. Hemaspaandra, and J. Rothe. Anyone but him: The complexity of precluding an alternative. *Artificial Intelligence*, 171(5-6):255–285, 2007. 2

[18] C. M. Homan and L. A. Hemaspaandra. Guarantees for the success frequency of an algorithm for finding Dodgson-election winners. *Journal of Heuristics*, 2007. 4

[19] H. W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8:538–548, 1983. 3

[20] J. C. McCabe-Dansted. Approximability and computational feasibility of Dodgson's rule. Master's thesis, University of Auckland, 2006. 3, 6, 24

[21] J. C. McCabe-Dansted, G. Pritchard, and A. Slinko. Approximability of Dodgson's rule. *Social Choice and Welfare*, 31(2):311–330, 2008. 4

[22] I. McLean and A. Urken. *Classics of Social Choice*. University of Michigan Press, Ann Arbor, Michigan, 1995. 2

[23] R. Meir, A. D. Procaccia, J. S. Rosenschein, and A. Zohar. The complexity of strategic behavior in multi-winner elections. *Journal of Artificial Intelligence Research*, 33:149–178, 2008. 2

[24] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006. 2, 3, 5

[25] J. Rothe, H. Spakowski, and J. Vogel. Exact complexity of the winner problem for Young elections. *Theory of Computing Systems*, 36:375–386, 2003. 2, 5, 23

[26] H. P. Young. Extending Condorcet's rule. *Journal of Economic Theory*, 16:335–353, 1977. 2, 3, 4

[27] M. Zuckerman, A. D. Procaccia, and J. S. Rosenschein. Algorithms for the coalitional manipulation problem. *Artificial Intelligence*, 173(2):392–412, 2009. 2