

Linear Time Parameterized Algorithms via Skew-Symmetric Multicuts*

M. S. RAMANUJAN[†]SAKET SAURABH[‡]

Abstract

A skew-symmetric graph $(D = (V, A), \sigma)$ is a directed graph D with an involution σ on the set of vertices and arcs. Flows on skew-symmetric graphs have been used to generalize maximum flow and maximum matching problems on graphs, initially by Tutte [1967], and later by Goldberg and Karzanov [1994, 1995]. In this paper, we introduce a separation problem, d -SKEW-SYMMETRIC MULTICUT, where we are given a skew-symmetric graph D , a family of \mathcal{T} of d -sized subsets of vertices and an integer k . The objective is to decide if there is a set $X \subseteq A$ of k arcs such that every set J in the family has a vertex v such that v and $\sigma(v)$ are in different connected components of $D' = (V, A \setminus (X \cup \sigma(X)))$. In this paper, we give an algorithm for d -SKEW-SYMMETRIC MULTICUT which runs in time $\mathcal{O}((4d)^k(m+n+\ell))$, where m is the number of arcs in the graph, n the number of vertices and ℓ the length of the family given in the input.

This problem, apart from being independently interesting, also abstracts out and captures the main combinatorial obstacles towards solving numerous classical problems. Our algorithm for d -SKEW-SYMMETRIC MULTICUT paves the way for the first *linear time parameterized algorithms* for several problems. We demonstrate its utility by obtaining following linear time parameterized algorithms.

- We show that ALMOST 2-SAT is a special case of 1-SKEW-SYMMETRIC MULTICUT, resulting in an algorithm for ALMOST 2-SAT which runs in time $\mathcal{O}(4^k k^4 \ell)^1$ where k is the size of the solution and ℓ is the length of the input formula. Then, using linear time parameter preserving reductions to ALMOST 2-SAT, we obtain algorithms for ODD CYCLE TRANSVERSAL and EDGE BIPARTIZATION which run in time $\mathcal{O}(4^k k^4(m+n))$ and $\mathcal{O}(4^k k^5(m+n))$ respectively where k is size of the solution, m and n are the number of edges and vertices respectively. This resolves an open problem posed by Reed, Smith and Vetta [Operations Research Letters, 2003] and improves upon the earlier almost linear time algorithm of Kawarabayashi and Reed [SODA, 2010].
- We show that DELETION q -HORN BACKDOOR SET DETECTION is a special case of 3-SKEW-SYMMETRIC MULTICUT, giving us an algorithm for DELETION q -HORN BACKDOOR SET DETECTION which runs in time $\mathcal{O}(12^k k^5 \ell)$ where k is the size of the solution and ℓ is the length of the input formula. This gives the first fixed-parameter tractable algorithm for this problem answering a question posed in a paper by a superset of the authors [STACS, 2013]. Using this result, we get an algorithm for SATISFIABILITY which runs in time $\mathcal{O}(12^k k^5 \ell)$ where k is the size of the smallest q -Horn deletion backdoor set, with ℓ being the length of the input formula.

*Supported by Parameterized Approximation, ERC Starting Grant 306992.

[†]Institute of Mathematical Sciences, India. {msramanujan|saket}@imsc.res.in

[‡]University of Bergen, Norway.

¹We have recently learnt that independently to our work and using very different techniques, Iwata, Oka and Yoshida have also given a $\mathcal{O}(4^k \ell)$ parameterized algorithm for ALMOST 2-SAT.

1 Introduction

A skew-symmetric graph is a digraph $D = (V, A)$ along with an involution $\sigma : V \cup A \rightarrow V \cup A$ where for every $x \in V \cup A$, $\sigma(x) \neq x$ and $\sigma(\sigma(x)) = x$ and for every $x \in V$ ($x \in A$), $\sigma(x) \in V$ (respectively $\sigma(x) \in A$). Skew-symmetric graphs were introduced under the name of *antisymmetrical digraphs* by Tutte [40] along with a notion of self-conjugate flows as a generalization of maximum flows in networks and matchings in graphs and subsequently by Zelinka [44] and Zaslavsky [43]. Goldberg and Karzanov [15, 16] revisited the work of Tutte and gave unified proofs for the analogues of the flow-decomposition and max-flow min-cut theorems on these graphs.

In this paper we use skew-symmetric graphs and an appropriate notion of separators on them as a model to abstract out “cut properties” underlying several problems in parameterized complexity. In parameterized complexity each problem instance comes with a parameter k and a central notion in parameterized complexity is *fixed parameter tractability* (FPT). This means, for a given instance (x, k) , solvability in time $f(k) \cdot p(|x|)$, where f is an arbitrary function of k and p is a polynomial in the input size.

We now introduce the main problem studied in this paper – a variant of the MULTICUT problem on skew-symmetric graphs.

<p>d-SKEW-SYMMETRIC MULTICUT</p> <p>Input: A skew-symmetric graph $D = ((V, A), \sigma)$, a family \mathcal{T} of d-sets of vertices, integer k.</p> <p>Question: Is there a set $S \subseteq A$ such that $S = \sigma(S)$, $S \leq 2k$, and for any d-set $\{v_1, \dots, v_d\}$ in \mathcal{T}, there is a vertex v_i such v_i and $\sigma(v_i)$ lie in distinct strongly connected components of $D \setminus S$?</p>	<p>Parameter: k</p>
--	---

The set S in the above definition is called a *skew-symmetric multicut* for the given instance. Our main result is a FPT algorithm for the above problem where the dependence of the running time of the algorithm on the input size is linear. Formally,

Theorem 1. *There is an algorithm that, given an instance $(D = (V, A, \sigma), \mathcal{T}, k)$ of d -SKEW-SYMMETRIC MULTICUT, runs in time $\mathcal{O}((4d)^k k^4 (\ell + m + n))$ and either returns a skew-symmetric multicut of size at most $2k$ or correctly concludes that no such set exists, where $m = |A|$ and $n = |V|$, and ℓ , the length of the family \mathcal{T} , is defined as $d \cdot |\mathcal{T}|$.*

Overview of our algorithm. The main obstacle to applying existing digraph algorithms on skew-symmetric graphs comes from the fact that standard arguments heavily based on sub-modularity of cuts break down, allowing only approximations, (see for example [28, 13]). Our first contribution is a reduction rule which overcomes this obstacle by allowing us to essentially (and correctly) think of *local* parts of an irreducible instance as a normal digraph. The reduction rule is essentially the following.

“Given two vertex sets X and Y which satisfy certain properties, if there is a minimum X - Y separator which contains an arc a and its image $\sigma(a)$, then some arc (*not necessarily* a) in this separator is part of an optimal solution and therefore, we find this arc, delete it from the instance, reduce the budget and continue.”

Though simple to state, the soundness of this reduction rule is far from obvious, requires certain structural observations specific to separators in skew-symmetric graphs. Observe

that this is a *parameter decreasing rule* which ensures that the number of applications of this rule is bounded linearly in the parameter. We believe that this rule could prove to be of independent interest for data reduction (or kernelization) for this as well as other similar problems.

Given this reduction rule, the next obstacle we need to overcome is that of applying this rule in linear time. For this, we start from the Ford-Fulkerson algorithm for computing maximum flows and by coupling it carefully with structural properties of skew-symmetric graphs, show that in linear time, we can either apply the reduction rule or locate a part of the graph which is already “reduced” for the next step of our algorithm. In the final step of our algorithm, having found a reduced part of the graph, we show that it is sufficient for us to consider arcs emanating from this part whose number is linearly bounded in the parameter and move ahead by performing an exhaustive branching on this bounded set of arcs. Finally, by a combination of these three subroutines, we obtain a linear time FPT algorithm for *d-SKEW-SYMMETRIC MULTICUT*.

An additional feature of the algorithm we present is that it does not require the family \mathcal{T} to be explicitly given as part of the input. It is sufficient for our algorithm to have access to a linear time violation oracle for \mathcal{T} , i.e, an algorithm which, in linear time returns a violated set in \mathcal{T} . This feature increases the utility of this algorithm substantially and we demonstrate this in the case of *DELETION q-HORN BACKDOOR SET DETECTION* where even though a direct reduction does not run in linear time, we can use a linear time violation oracle to obtain a linear time FPT algorithm for *DELETION q-HORN BACKDOOR SET DETECTION*.

Applications. Here we provide the list of main applications (see Figure 1) that can be derived from our algorithm (Theorem 1) together with a short overview of previous work on each application.

ODD CYCLE TRANSVERSAL, ALMOST 2-SAT and related problems. Graph bipartization is a classical NP-hard problem with several applications [5, 38, 33]. In the field of parameterized complexity, this problem is better known as *ODD CYCLE TRANSVERSAL*, which is formally defined as follows.

<p>ODD CYCLE TRANSVERSAL(OCT) Input: A graph G, positive integer k Question: Does there exist a set S of at most k vertices such that $G \setminus S$ is a bipartite graph?</p>	<p>Parameter: k</p>
---	---

The parameterized complexity of *ODD CYCLE TRANSVERSAL* was a well known open problem for a long time. In 2003, in a breakthrough paper, Reed et al. [37] showed that *OCT* is FPT by developing an algorithm for the problem running in time $\mathcal{O}(3^k mn)$. We use n and m to denote the number of vertices and edges of the input graph respectively. In fact this was the first time that the iterative compression technique was used. This technique has been useful in resolving several other open problems in the area of parameterized complexity, including *DIRECTED FEEDBACK VERTEX SET*, *ALMOST 2-SAT*, *MULTICUT* [4, 36, 29]. However, the algorithm for *OCT* had seen no further improvements in the last 9 years, though reinterpretations of the algorithm have been published [17, 23]. Only recently, Lokshantov et al. [21] obtained an algorithm with an improved dependence on the parameter k . This algorithm is based on a branching guided by linear programming and runs in time $\mathcal{O}(2.32^k n^{\mathcal{O}(1)})$. In a parallel line of research, Fiorini et al. [10] showed that when the input is restricted to planar graphs there is an $\mathcal{O}(f(k)n)$ time algorithm— a linear time algorithm, for *OCT*. Continuing this line of research, recently, Kawarabayashi and Reed [18] obtained an

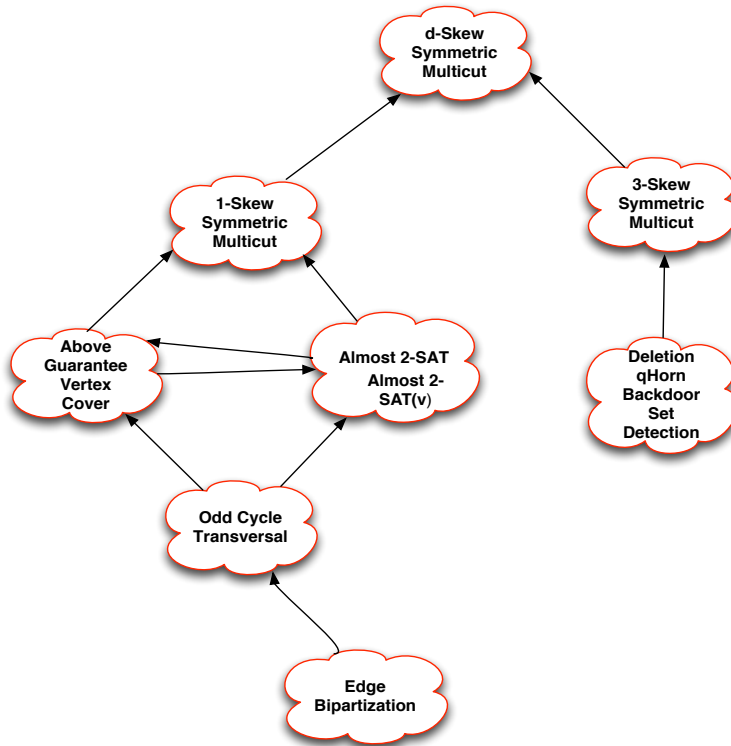


Figure 1: Problems with a linear time parameter preserving reduction to d -SKEW-SYMMETRIC MULTICUT

algorithm for OCT on general graphs with an improved dependence on the input size. This algorithm uses tools from graph minors and odd variants of graph minors and runs in time $\mathcal{O}(f(k)m \cdot \alpha(m, n))$. Here the function $\alpha(m, n)$ is the inverse of the Ackermann function (see by Tarjan [39]) and $f(k)$ is at least a triple exponential function. However, an algorithm on general graphs with a linear dependence on the input size has so far proved elusive. In this work, we obtain the first linear time algorithm for OCT running in time $\mathcal{O}(4^k k^4(m + n))$. This resolves an open problem posed by Reed et al. in 2003 [37].

In the *edge* version of OCT, namely EDGE BIPARTIZATION, the objective is to test if there is a set of at most k edges whose deletion makes the input graph bipartite. Using a known linear time parameter preserving reduction from EDGE BIPARTIZATION to OCT, we also get a similar result for EDGE BIPARTIZATION. In fact, both these problems have linear time parameter preserving reductions to the more general problem of ALMOST 2-SAT. In fact, our algorithms for EDGE BIPARTIZATION and OCT are obtained via reductions to ALMOST 2-SAT, which in turn is solved using our algorithm for d -SKEW-SYMMETRIC MULTICUT (Theorem 1).

The ALMOST 2-SAT problem is formally defined as follows.

ALMOST 2-SAT	Parameter: k
Input: A 2-CNF formula F , positive integer k	
Question: Does there exist a set S_c of at most k clauses of F such that $F \setminus S_c$ is satisfiable?	

It was introduced by Mahajan and Raman [24] in 1999 and its parameterized complexity status remained open until 2008 when Razgon and O’Sullivan [35] gave an algorithm running in time $\mathcal{O}(15^k m^3)$ on formulas with m clauses. More recently, there have been a series of improved algorithms ($\mathcal{O}(9^k n^{\mathcal{O}(1)})$ [34], $\mathcal{O}(4^k n^{\mathcal{O}(1)})$ [7], $\mathcal{O}(2.618^k n^{\mathcal{O}(1)})$ [30]) with the current best algorithm running in time $\mathcal{O}(2.32^k n^{\mathcal{O}(1)})$ [21]. However, none of these algorithms have linear dependence on the input size. We show that ALMOST 2-SAT is a special case of 1-SKEW-SYMMETRIC MULTICUT, resulting in an algorithm for ALMOST 2-SAT which runs in time $\mathcal{O}(4^k k^4 \ell)$ where k is the size of the solution and ℓ is the length of the input formula.

Another problem related to ALMOST 2-SAT is the ABOVE GUARANTEE VERTEX COVER (AGVC) which is defined as follows.

<p>ABOVE GUARANTEE VERTEX COVER (AGVC)</p> <p>Input: $(G = (V, E), M, k)$, where G is an undirected graph, M is a maximum matching for G, k a positive integer</p> <p>Question: Does G have a vertex cover of size at most k?</p>	<p>Parameter: $k - M$</p>
--	---

This problem is linear time equivalent to ALMOST 2-SAT in a parameter preserving way and hence, our results imply an algorithm for AGVC with running time $\mathcal{O}(4^{k-|M|}(k-|M|)^4(m+n))$. This equivalence has already proved useful as a linear programming based branching algorithm for AGVC led to algorithms for ALMOST 2-SAT and several other problems around it. However, the linear time reduction from AGVC to ALMOST 2-SAT crucially utilizes the fact that a maximum matching of the graph is also part of the input. Therefore, a natural goal would be to obtain an algorithm running in time $f(k-|M|)(m+n)$ for AGVC even when a maximum matching is not given as part of the input.

DELETION q -Horn BACKDOOR SET DETECTION and related problems. A *strong \mathcal{C} -backdoor set* of a CNF formula F is a set B of variables such that $F[\tau] \in \mathcal{C}$ for each assignment $\tau : B \rightarrow \{0, 1\}$, that is, for every instantiation of the variables in B , the reduced formula is in the class \mathcal{C} . A *deletion \mathcal{C} -backdoor set* of F is a set B of variables such that $F - B \in \mathcal{C}$. Backdoor sets were independently introduced by Crama et al. [6] and by Williams et al. [42], the latter authors coining the term “backdoor”. If we know a strong \mathcal{C} -backdoor set of F of size k , we can reduce the satisfiability of F to the satisfiability of 2^k formulas in \mathcal{C} . If \mathcal{C} is clause-induced, every deletion \mathcal{C} -backdoor set of F is a strong \mathcal{C} -backdoor set of F . For several base classes, deletion backdoor sets are of interest because they are easier to detect than strong backdoor sets.

The parameterized complexity of finding small backdoor sets was initiated by Nishimura et al. [32] who showed that for the base classes of Horn formulas and Krom formulas, the detection of strong backdoor sets is fixed-parameter tractable. For base classes other than Horn and Krom, strong backdoor sets can be much smaller than deletion backdoor sets, and their detection is more difficult. For more recent results, the reader is referred to a survey on the parameterized complexity of backdoor sets [14].

The class q -Horn, introduced by Boros, Crama and Hammer [2], is one of the largest known classes of propositional CNF formulas for which satisfiability can be decided in polynomial time. This class properly contains the fundamental classes of Horn and Krom formulas as well as the class of renamable (or disguised) Horn formulas. The parameterized complexity of finding small q -Horn backdoor sets was studied by Gaspers et al. [13] who showed that the DELETION q -Horn BACKDOOR SET DETECTION problem is fixed-parameter approximable. Formally, the DELETION q -Horn BACKDOOR SET DETECTION problem is the following.

DELETION q -HORN BACKDOOR SET DETECTION

Parameter: k

Input: A CNF formula F and a positive integer k

Question: Does F have a deletion q -Horn backdoor set of size at most k ?

We show that DELETION q -HORN BACKDOOR SET DETECTION is a special case of 3-SKEW-SYMMETRIC MULTICUT, giving us an algorithm for DELETION q -HORN BACKDOOR SET DETECTION which runs in time $\mathcal{O}(12^k k^5 \ell)$ where k is the size of the solution and ℓ is the length of the input formula. This gives the first fixed-parameter tractable algorithm for this problem. Using this result, we get an algorithm for SATISFIABILITY which runs in time $\mathcal{O}(12^k k^5 \ell)$ where k is the size of the smallest q -Horn deletion backdoor set, with ℓ being the length of the input formula.

Related Results on Cut problems. d -SKEW-SYMMETRIC MULTICUT is neither the first nor will it be the last cut problem studied in parameterized complexity. Marx [25] was the first to consider graph separation problems in the context of parameterized complexity. He gave an algorithm for MULTIWAY CUT with a running time of $\mathcal{O}(4^{k^3} n^{\mathcal{O}(1)})$ which was later improved to $\mathcal{O}(4^k n^{\mathcal{O}(1)})$ by Chen et al. [3]. The current fastest algorithm for MULTIWAY CUT runs in time $\mathcal{O}(2^k n^{\mathcal{O}(1)})$ [8]. The notions used in the paper of Marx have been useful in settling the parameterized complexity, as well as obtaining improved FPT algorithms for a wide variety of problems including DIRECTED FEEDBACK VERTEX SET [4], ALMOST 2-SAT [36] and ABOVE GUARANTEE VERTEX COVER [36, 34]. These sequence of results have led to an entirely new and very active subarea dealing with parameterized graph separation problems both due to independent interest in the problems themselves, as well as due to the fact that these problems seem to be able to capture the underlying properties of a large variety of seemingly unrelated problems.

Organization of the paper. In Section 3, we define the notions of separators in skew-symmetric graphs, followed by structural results on separators in skew-symmetric graphs and the notions of (L, k) -components whose computation is at the core of our algorithm. In Section 4, we prove an observation regarding the structure of optimal solutions, followed by a description and proof of correctness of our algorithm. In Section 5, we give linear time parameterized algorithms for a number of problems using our result.

2 Preliminaries

In this section we give some basic definitions and set up the notations for the paper.

Parameterized Complexity. Parameterized complexity is one of the ways to cope up with intractability of problems. The goal of parameterized complexity is to find ways of solving NP-hard problems more efficiently than by brute force. Here, the aim is to restrict the combinatorial explosion of computational difficulty to a parameter that is hopefully much smaller than the input size. Formally, a *parameterization* of a problem is the assignment of an integer k to each input instance and we say that a parameterized problem is *fixed-parameter tractable* (FPT) if there is an algorithm that solves the problem in time $f(k) \cdot |I|^{\mathcal{O}(1)}$, where $|I|$ is the size of the input instance and f is an arbitrary computable function depending only on the parameter k . For more background, the reader is referred to the monographs [9, 11, 31].

Digraphs. Let $D = (V, A)$ be a directed graph. For an arc $(u, v) \in A$, we refer to u as the *tail* of this arc and denote it by $\mathbf{Tail}(u, v)$ and we refer to v as the *head* of this arc and denote it by $\mathbf{Head}(u, v)$. For a set of arcs P , we denote by $\mathbf{Tail}(P)$, the set $\bigcup_{(u,v) \in P} \{\mathbf{Tail}(u, v)\}$

and we denote by $\mathbf{Head}(P)$ the set $\bigcup_{(u,v) \in P} \{\mathbf{Head}(u, v)\}$. For a set of vertices V' , we let $A[V']$ denote the set of arcs with both end points in the set V' . For a set of vertices V' , we let $\delta^+(V')$ denote the set of arcs which have their tail in V' and their head in $V \setminus V'$. Similarly, we let $\delta^-(V')$ denote the set of arcs which have their head in V' and their tail in $V \setminus V'$. We also use $N^+(V')$ to denote the set $\mathbf{Head}(\delta^+(V'))$ and $N^-(V')$ to denote the set $\mathbf{Tail}(\delta^-(V'))$. Given two disjoint vertex sets X and Y , we define an X - Y path as a directed path from a vertex $x \in X$ to a vertex $y \in Y$ whose internal vertices are disjoint from $X \cup Y$.

Skew-Symmetric Graphs. The notation is from [16]. A *skew-symmetric graph* is a digraph $D = (V, A)$ and an involution $\sigma : V \cup A \rightarrow V \cup A$ such that:

1. for each $x \in V \cup A$, $\sigma(x) \neq x$ and $\sigma(\sigma(x)) = x$.
2. for each $v \in V$, $\sigma(v) \in V$
3. for each $a = (v, w) \in A$, $\sigma(a) = (\sigma(w), \sigma(v))$

We call $\sigma(x)$ *symmetric* to x and also refer to x and $\sigma(x)$ as *conjugates*. For ease of description, we let x' denote the conjugate of an element x and we let S' denote the set of conjugates of the elements in the set S . We say that a set S is *regular* if $S \cap S' = \emptyset$ and *irregular* otherwise. A set S is called *self-conjugate* if $S = S'$.

CNF Formulas and Satisfiability. A *literal* is a variable x or a negated variable \bar{x} ; if $y = x$ or $y = \bar{x}$ is a literal for some variable x , then we write \bar{y} to denote \bar{x} or x , respectively. A *clause* is a finite set of literals and a finite set of clauses is a *CNF formula* where the clauses are considered as a disjunction of its literals and the CNF formula is considered a conjunction of its clauses. By $\mathcal{C}(F)$ we denote the set of clauses of a CNF formula F . A formula is *Horn* if each of its clauses contains at most one positive literal, a formula is *Krom* (or *2CNF*, or *quadratic*) if each clause contains at most two literals. The *length* of a CNF formula F is defined as $\sum_{C \in F} |C|$. If F is a formula and X a set of variables, then we denote by $F - X$ the formula obtained from F after removing all literals with a variable in X from the clauses in F . Let F be a formula and $X \subseteq \text{var}(F)$. A *truth assignment* is a mapping $\tau : X \rightarrow \{0, 1\}$ defined on some set X of variables. A truth assignment τ *satisfies* a clause C if C contains some literal x with $\tau(x) = 1$; τ satisfies a formula F if it satisfies all clauses of F . A formula is *satisfiable* if it is satisfied by some truth assignment; otherwise it is *unsatisfiable*.

3 Skew-symmetric graphs, separators and components

The following observation is a direct consequence of the definition of a skew-symmetric graph.

Observation 3.1. *Let $D = ((V, A), \sigma)$ be a skew-symmetric graph and let $u, v \in V$. There is a path from v to u in D iff there is a path from u' to v' .*

Definition 3.1. *Let $D = (V, A)$ be a directed graph and let X, Y be disjoint subsets of V . A set $S \subseteq A$ is an X - Y separator if there is no directed path from X to Y in the graph $D \setminus S$. We say that S is a minimal X - Y separator if no proper subset of S is an X - Y separator.*

Definition 3.2. *Let $D = ((V, A), \sigma)$ be a skew-symmetric graph and let L be a regular set of vertices. Let $X \subseteq A$ be a self-conjugate set of arcs of D . We call X an L - L' **self-conjugate separator** if X is a (not necessarily minimal) L - L' separator. We call X a minimal L - L' self-conjugate separator if there is no self-conjugate strict subset of X which is also an L - L' separator.*

Definition 3.3. Let $D = ((V, A), \sigma)$ be a skew-symmetric graph and let L be a regular set of vertices. Let X be an L - L' self-conjugate separator. We denote by $R(L, X)$ the set of vertices of D that can be reached from L via directed paths in $D \setminus X$, and we denote by $\bar{R}(L, X)$ the set of vertices of D which have a directed path to can reach L in $D \setminus X$.

Observation 3.2. Let $D = ((V, A), \sigma)$ be a skew-symmetric graph and let L be a regular set of vertices. Let X be an L - L' self-conjugate separator. Then, the sets $R(L, X)$ and $\bar{R}(L', X)$ are also regular and $\sigma(R(L, X)) = \bar{R}(L', X)$.

Proof. Since deleting a self-conjugate set of arcs from a skew-symmetric graph results in a skew-symmetric graph, we know that there is a path from u to v in $D \setminus X$ iff there is a path from v' to u' in $D \setminus X$. Therefore, if $R(L, X)$ is irregular, then there is a path from L to y and y' for some vertex y , which is disjoint from X , which implies a path from L to L' in $D \setminus X$, which is a contradiction. Therefore, $R(L, X)$ and $\bar{R}(L', X)$ are regular and since $D \setminus X$ is a skew-symmetric graph, they are conjugates. \square

3.1 Minimum separators in skew-symmetric graphs

Lemma 3.1. Let $D = ((V, A), \sigma)$ be a skew-symmetric graph, L be a regular set of vertices.

1. Suppose that there is an L - L' path in D and let X be a minimum L - L' separator and let $Z = R(L, X \cup X')$. Then, $\delta^+(Z)$ is also a minimum L - L' separator.
2. An arc is part of a minimum L - L' separator if and only if its conjugate is also part of a minimum L - L' separator.

Proof. Recall that Z is regular (Observation 3.2). Since L is in Z and L' is disjoint from Z , $\delta^+(Z)$ is an L - L' separator. It remains to show that it is a minimum such separator. Clearly, $\delta^+(Z) \subseteq X \cup X'$. Let $A = \delta^+(Z) \cap X$ and $B = \delta^+(Z) \setminus X$.

Since B is disjoint from X , it must be the case that $B' \subseteq X$. We now claim that A and B' are disjoint. Suppose that this is not the case and let $x \in B$ such that $x' \in A$. Since $x' \in \delta^+(Z)$, it must be the case that $x \in \delta^-(Z')$. Since there is a path from Z to Z' via x and X is disjoint from $A[Z \cup Z'] \cup \{x\}$, there is a path from L to L' disjoint from X , a contradiction. Therefore, we conclude that $A \cap B' = \emptyset$. We now have that $|\delta^+(Z)| = |A \cup B| = |A| + |B| \leq |X|$ where the last inequality used the fact that A and B' are disjoint. Therefore, we conclude that $\delta^+(Z)$ is indeed a minimum L - L' separator. Consequently, $\delta^-(Z)$ is also a minimum L - L' separator. This concludes the proof of the first part of the lemma.

We claim that if X is an L - L' separator, then X' is an L - L' separator as well. Suppose that this is not the case and let there be a path v_1, \dots, v_r in $D \setminus X'$ where $v_1 = l$ and $v_r = l'$. However, this implies that X is disjoint from the path $\sigma(v_r), \dots, \sigma(v_1)$, which is a contradiction. This concludes the proof of the lemma. \square

Lemma 3.2. (*Crossing-Uncrossing*) Let $D = ((V, A), \sigma)$ be a skew-symmetric graph and let $B \subseteq V$. If $\delta^+(B) \cap \delta^+(B') = \emptyset$ then $|\delta^+(B \setminus B')| = |\delta^+(B)|$ and $|\delta^+(B \setminus B')| < |\delta^+(B)|$ otherwise.

Proof. Let $Q = B \setminus B'$. We partition $\delta^+(Q)$ into the following sets (see Figure 2).

1. $Q_1^o = \delta^+(Q) \cap \delta^-(V \setminus (B \cup B'))$, that is, those arcs with the tail in Q and the head in $V \setminus (B \cup B')$.
2. $Q_2^o = \delta^+(Q) \cap \delta^-(B \setminus B')$, that is, those arcs with the tail in Q and the head in $B' \setminus B$.

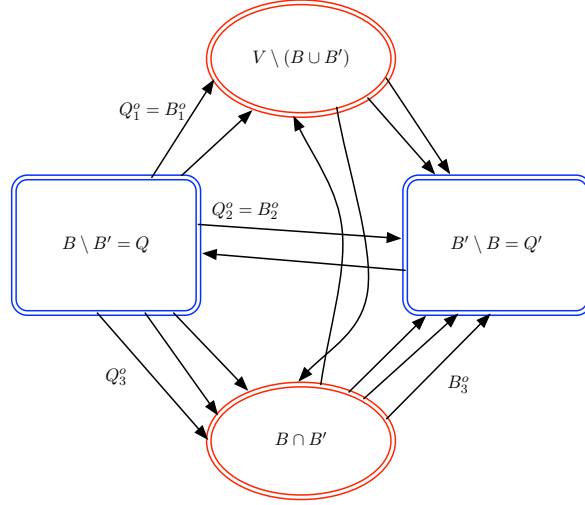


Figure 2: An illustration of the partitions described in the proof of Lemma 3.2

3. $Q_3^o = \delta^+(Q) \cap \delta^-(B \cap B')$, that is, those arcs with the tail in Q and the head in $B \cap B'$.

Similarly, we partition $\delta^+(B)$ as follows.

1. $B_1^o = (\delta^+(B \setminus B') \cap \delta^-(V \setminus (B \cup B')))$, that is, those arcs with the tail in $B \setminus B'$ and the head in $V \setminus (B \cup B')$.
2. $B_2^o = (\delta^+(B) \cap \delta^-(B' \setminus B))$, that is, those arcs with the tail in $B \setminus B'$ and the head in $B' \setminus B$.
3. $B_3^o = \delta^+(B) \cap \delta^+(B' \setminus B)$, that is, those arcs with the tail in $B \cap B'$ and the head in $B \setminus B'$.
4. $B_4^o = \delta^+(B) \cap \delta^+(B')$

Observe that $Q_1^o = B_1^o$, $Q_2^o = B_2^o$ and $Q_3^o = (B_3^o)'$. Therefore, $|\delta^+(Q)| = |\delta^+(B)|$ if B_4^o is empty and $|\delta^+(Q)| < |\delta^+(B)|$ otherwise. This completes the proof of the lemma. \square

3.2 (L, k) -Components

Definition 3.4. Let $D = ((V, A), \sigma)$ be a skew-symmetric graph and $k \in \mathbb{N}$. Let $L \subseteq V$ be a regular set of vertices. A set of vertices $Z \subseteq V$ is called an (L, k) -component if it satisfies the following properties.

1. $L \subseteq Z$
2. Z is regular
3. Z is reachable from L in $D[Z]$
4. The size of a minimum Z - Z' separator is equal to the size of a minimum L - L' separator and this size is at most $2k$.
5. Z is inclusion-wise maximal among the sets satisfying the above properties.

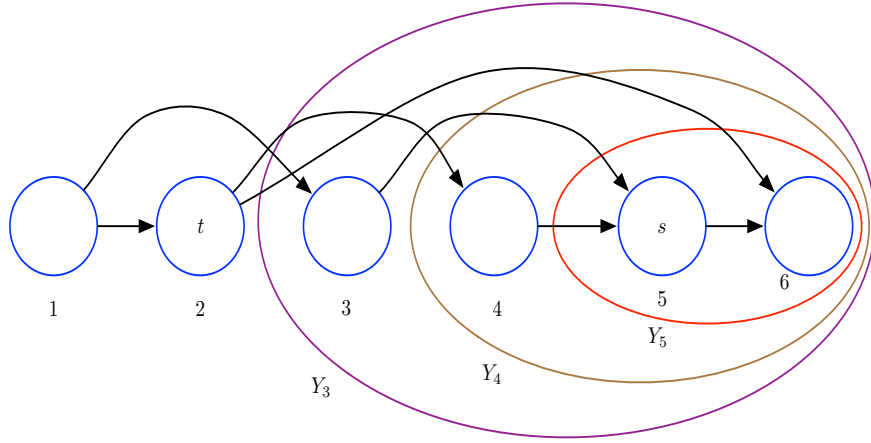


Figure 3: An illustration of the sets in the proof of Lemma 3.3. The blue circles are the strongly connected components of D_1 and $\alpha(s) = 5$ and $\alpha(t) = 2$.

Lemma 3.4 gives an algorithm that in linear time either computes an (L, k) -component or finds a minimum L - L' separator of a particular kind, which we later show can be used to reduce the instance. We first require the arc version of Lemma 2.4 from [27].

Lemma 3.3. *Let s, t be two vertices in a digraph $D = (V, A)$ such that the minimum size of an s - t separator is $\ell > 0$. Then, there is a collection $\mathcal{X} = \{X_1, \dots, X_q\}$ of sets where $\{s\} \subseteq X_i \subseteq V \setminus \{t\}$ such that*

1. $X_1 \subset X_2 \subset \dots \subset X_q$,
2. X_i is reachable from s in $D[X_i]$,
3. $|\delta^+(X_i)| = \ell$ for every $1 \leq i \leq q$ and
4. every s - t separator of size ℓ is fully contained in $\bigcup_{i=1}^q \delta^+(X_i)$.

Furthermore, there is an $\mathcal{O}(\ell(|V| + |A|))$ time algorithm that produces the sets $X_1, X_2 \setminus X_1, \dots, X_q \setminus X_{q-1}$ corresponding to such a collection \mathcal{X} .

Proof. This proof is from [27]. However, the proof in [27] does not need that X_i is reachable from s in $D[X_i]$. Thus, we need to do a bit more extra work for the version presented here. We first run ℓ iterations of the Ford-Fulkerson algorithm on the graph with unit capacities on all arcs to find a maximum s - t flow. Let D_1 be the residual graph. Let C_1, \dots, C_q be a topological ordering of the strongly connected components of D_1 such that $i < j$ if there is a path from C_i to C_j . Recall that there is a t - s path in D_1 . Let C_x and C_y be the strongly connected components of D_1 containing t and s respectively. Since there is a path from t to s in D_1 , $x < y$. For each $x < i \leq y$, let $Y_i = \bigcup_{j=i}^q C_j$ (see Figure 3). We first show that $|\delta^+(Y_i)| = \ell$. Since no arcs leave Y_i in the graph D_1 , no flow enters Y_i and every arc in $\delta^+(Y_i)$ is saturated by the maximum flow. Therefore, $|\delta^+(Y_i)| = \ell$.

We now show that every arc which is part of a minimum s - t separator is contained in $\bigcup_{i=1}^q \delta^+(Y_i)$. Consider a minimum s - t separator S and an arc $(a, b) \in S$. Let Y be the set of vertices reachable from s in $D \setminus S$. Since F is a minimum s - t separator, it must be the case that $\delta^+(Y) = F$ and therefore, $\delta^+(Y)$ is saturated by the maximum flow. Therefore, we have that (b, a) is an arc in D_1 . Since no flow enters the set Y , there is no cycle in D_1

containing the arc (b, a) and therefore, if the strongly connected component containing b is C_{i_b} and that containing a is C_{i_a} , then $i_b < i_a$. Furthermore, since there is flow from s to a from b to t , it must be the case that $x < i_b < i_a < y$ and hence the arc (a, b) appears in the set $\delta^+(Y_{i_a})$.

Finally, we define the set $R(Y_i)$ to be the set of vertices of Y_i which are reachable from s in the graph $D[Y_i]$. Then, the sets $R(Y_y) \subset R(Y_{y-1}) \subset \dots \subset R(Y_{x+1})$ form a collection of the kind described in the statement of the lemma.

In order to compute these sets, we first need to run the Ford-Fulkerson algorithm for ℓ iterations and perform a topological sort of the strongly connected components of D_1 . This takes time $\mathcal{O}(\ell(|V| + |A|))$. During this procedure, we also assign indices to the strongly connected components in the manner described above, that is, $i < j$ if C_i occurs before C_j in the topological ordering. In $\mathcal{O}(\ell(|V| + |A|))$ time, we can assign indices to vertices such that the index of a vertex v (denoted by $\alpha(v)$) is the index of the strongly connected component containing v . We then perform the following preprocessing for every vertex v such that $\alpha(v) < \alpha(s)$. We go through the list of in-neighbors of v and find

$$\beta(v) = \max_{u \in N^-(v)} \left\{ \alpha(u) \mid \alpha(v) < \alpha(u) \right\} \text{ and}$$

$$\gamma(v) = \min_{u \in N^-(v)} \left\{ \alpha(u) \mid \alpha(v) < \alpha(u) \right\}$$

and set $\beta'(v) = \min\{\beta(v), \alpha(s)\}$ and $\gamma'(v) = \max\{\gamma(v), \alpha(t) + 1\}$.

The meaning of these numbers is that the vertex v occurs in each of the sets $N^+(Y_{\beta'(v)})$, $N^+(Y_{\beta'(v)-1})$, \dots , $N^+(Y_{\gamma'(v)})$. This preprocessing can be done in time $\mathcal{O}(m + n)$ since we only compute the maximum and minimum in the adjacency list of each vertex. A vertex v is said to be i -forbidden for all $\gamma'(v) \leq i \leq \beta'(v)$. We now describe the algorithm to compute the sets in the collection.

Computing the collection. We do a modified (directed) breadth first search (BFS) starting from s by using only *out-going* arcs. Along with the standard BFS queue, we also maintain an additional *forbidden* queue.

We begin by setting $i = \alpha(s)$ and start the BFS by considering the out-neighbors of s . We add a vertex to the BFS queue only if it is both unvisited and not i -forbidden. If a vertex is found to be i -forbidden (and it is not already in the forbidden queue), we add this vertex to the forbidden queue. Finally, when the BFS queue is empty and every unvisited out-neighbor of every vertex in this tree is in the forbidden queue, we return the set of vertices *added* to the BFS tree in the current iteration as $R(Y_i) \setminus R(Y_{i+1})$. Following this, the vertices in the forbidden queue which are not $(i - 1)$ -forbidden are removed and added to the BFS queue and the algorithm continues after decreasing i by 1. The algorithm finally stops when $i = \alpha(t)$.

We claim that this algorithm returns each of the sets $R(Y_{\alpha(s)})$, $R(Y_{\alpha(s)-1}) \setminus R(Y_{\alpha(s)})$, \dots , $R(Y_{\alpha(t)+1}) \setminus R(Y_{\alpha(t)+2})$ and runs in time $\mathcal{O}(\ell(|V| + |A|))$. In order to bound the running time, first observe that the vertices which are i -forbidden are exactly the vertices in the set $N^+(R(Y_i))$ and therefore the number of i -forbidden vertices for each i is at most ℓ . This implies that the number of vertices in the forbidden queue at any time is at most ℓ . Hence, testing if a vertex is i -forbidden or already in the forbidden queue for a fixed i can be done in time $\mathcal{O}(\ell)$. Therefore, the time taken by the algorithm is $\mathcal{O}(\ell)$ times the time required for a BFS in D , which implies a bound of $\mathcal{O}(\ell(|V| + |A|))$.

For the correctness, we prove the following invariant for each iteration. Whenever a set is returned in an iteration,

- the set of vertices currently in the forbidden queue are exactly the i -forbidden vertices
- the vertices in the current BFS tree are exactly the vertices in the set $R(Y_i)$.

For the first iteration, this is clearly true. We assume that the invariant holds at the end of iteration $j \geq 1$ (where $i = i'$) and consider the $(j + 1)$ -th iteration (where i is now set as $i' - 1$).

Let P_j be the vertices present in the BFS tree at the end of the j -th iteration and P_{j+1} be the vertices present in the BFS tree at the end of the $(j + 1)$ -th iteration. We claim that the set $P_{j+1} = R(Y_{i'-1})$.

Since we never add a vertex to P_{j+1} if it is $(i' - 1)$ -forbidden, the vertices in $P_{j+1} \setminus P_j$ are precisely those vertices which are reachable from P_j via a path disjoint from $(i' - 1)$ -forbidden vertices. Since the invariant holds for the preceding iteration, we know that $P_j = R(Y_{i'})$ and by our observation about $P_{j+1} \setminus P_j$, we have that P_{j+1} is the set of vertices reachable from $R(Y_{i'})$ via paths disjoint from $(i' - 1)$ -forbidden vertices, which implies that $P_{j+1} = R(Y_{i'-1})$ since $R(Y_{i'-1})$ is precisely the set of vertices reachable from $R(Y_{i'})$ via paths disjoint from $(i' - 1)$ -forbidden vertices. We now show that the vertices in the forbidden queue are exactly the $(i' - 1)$ -forbidden vertices. Since the BFS tree in iteration $j + 1$ could not be grown any further, every out-neighbor of every vertex in the tree is in the forbidden queue. Since we have already shown that the vertices in the BFS tree, that is in P_{j+1} , are precisely the vertices in $R(Y_{i'-1})$, we have that every $(i' - 1)$ -forbidden vertex is already in the forbidden queue. This proves that the invariant holds in this iteration as well and completes the proof of correctness of the algorithm. \square

The main lemma of this section is the following.

Lemma 3.4. *Let $D = ((V, A), \sigma)$ be a skew-symmetric graph and $k \in \mathbb{N}$. Let $L \subseteq V$ be a regular set of vertices such that there is an L - L' path in D . There is an algorithm which runs in time $\mathcal{O}(k^3(m + n))$ and*

- *correctly concludes that no (L, k) -component exists or*
- *returns an (L, k) -component or*
- *returns an irregular minimum L - L' separator*

where $m = |A|$ and $n = |V|$.

Proof. The main idea of the algorithm is to start with the collection coming from Lemma 3.3 and then use this to either find an (L, k) -component or to return an irregular minimum L - L' separator. In what follows we describe possible situations that could arise and how they could be handled. Finally, we use all this to describe the algorithm and prove its correctness.

We can, in $\mathcal{O}(k(m + n))$ time check if the size of the minimum L - L' separator is at most $2k$ by running $2k$ iterations of the Ford-Fulkerson algorithm [12]. If the size of the minimum separator exceeds $2k$, then we can conclude that no (L, k) -component exists. Therefore, we assume in the rest of this proof that the size of the minimum L - L' separator is at most $2k$. Let $\mathcal{X} = \{X_1, \dots, X_q\}$ be a collection with the properties mentioned in Lemma 3.3 where “ L acts as s and L' acts as t ”. We make this formal when we describe the algorithm later.

We begin by showing that not all X_i 's can be irregular.

Claim 3.1. *There is an index $i \geq 1$ such that for all $j \leq i$, the set X_j is regular.*

Proof. We first show that X_1 is regular. Suppose that this is not the case and there are vertices $y, y' \in X_1$. Since no arc in $A[X_1]$ is part of a minimum L - L' separator (by property

4 of the collection), Lemma 3.1 implies that no arc in $A[X_1]$ is the conjugate of an arc in $S = \delta^+(X_1)$. Therefore, there is a path from L to y and a path from L to y' disjoint from $S \cup S'$. However, this implies the presence of a path from L to L' in $D \setminus (S \cup S')$, which is a contradiction. Therefore, X_1 is regular. Since every subset of a regular set is regular, there is an index $i \geq 1$ such that for all $j \leq i$, X_j is regular. This completes the proof of the claim. \square

Given the above claim, we first consider the case when X_i is regular for every $1 \leq i \leq q$. This brings us to the following claim.

Claim 3.2. *If X_i is regular for every $1 \leq i \leq q$ then X_q itself is an (L, k) -component.*

Proof. Observe that X_q satisfies the first four properties of an (L, k) -component and thus it suffices to prove that X_q is inclusion-wise maximal with respect to these 4 properties. Suppose that X_q is not maximal with respect to these properties and let $Z \supset X_q$ have the required properties and let Y be a minimum Z - Z' separator. Clearly, Y is a minimum L - L' separator. Since $Z \supset X_q$, there is an arc $y \in Y$ which is not in $\delta^+(X_q)$. Since X_q strictly contains all other X_i 's, $y \notin \delta^+(X_i)$ for any i , which contradicts property 4 of the collection. \square

Claims 3.1 and 3.2 act like base cases of our algorithm that we describe later.

We now suppose that there exists $i \leq q$ such that X_i is irregular. Let a be the highest index such that X_a is regular and X_{a+1} is irregular. Let $A = X_a$ and $B = X_{a+1}$. Since $\delta^+(B)$ is a minimum L - L' separator, by the crossing-uncrossing lemma (Lemma 3.2), we have that $|\delta^+(B \setminus B')| = |\delta^+(B)|$ and $\delta^+(B) \cap \delta^+(B') = \emptyset$. That is, there is no arc which enters $V \setminus (B \cup B')$ from $B \cap B'$ and thus there is no arc which enters $B \cap B'$ from $V \setminus (B \cup B')$. Furthermore, if there is an arc $x \in \delta^+(B \setminus B') \cap \delta^-(B' \setminus B)$, then $x' \in \delta^+(B \setminus B')$, which implies that x, x' are contained in a minimum L - L' separator. Therefore, from this point on, we may assume that there is no arc in $\delta^+(B \setminus B') \cap \delta^-(B' \setminus B)$. Before we go further we summarize the sets and the various intersections they have. From now onwards the sets B and $Q = B \setminus B'$ will always have the following intersection properties.

1. $Q = B \setminus B'$
2. $|\delta^+(B \setminus B')| = |\delta^+(Q)| = |\delta^+(B)|$
3. $\delta^+(B) \cap \delta^+(B') = \emptyset$
4. $\delta^-(B) \cap \delta^-(B') = \emptyset$
5. $\delta^+(B \setminus B') \cap \delta^-(B' \setminus B) = \delta^+(Q) \cap \delta^-(Q') = \emptyset$

The proof of the next observation follows from the fact that there is neither an arc entering $B \cap B'$ from $V \setminus (B \cup B')$ nor an arc leaving $B \cap B'$ to $V \setminus (B \cup B')$ (see Figure 4).

Observation 3.3. *Any path from Q to Q' with the internal vertices disjoint from $Q \cup Q'$ is contained entirely in either $D \setminus (B \cap B')$ or $D[B \cup B']$.*

The next claim describes certain properties of paths exiting Q and entering $B \cap B'$. This will be used later in some of the arguments.

Claim 3.3. *Let B and Q be defined as above. Then for every $q \in N^+(Q) \cap (B \cap B')$ there is a path from q to $N^-(Q') \cap (B \cap B')$ which completely lies in the graph $D[B \cap B']$.*

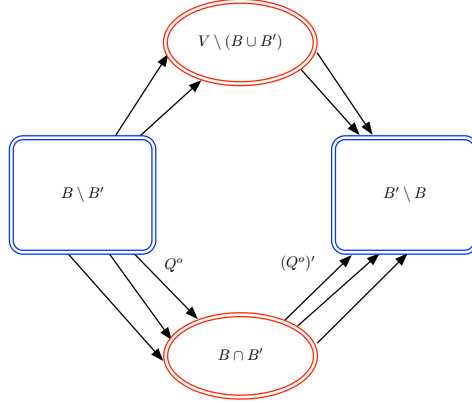


Figure 4: An illustration of the sets defined in Lemma 3.4. Observe that there are no arcs between $B \setminus B'$ and $B' \setminus B$ and between $B \cap B'$ and $V \setminus (B \cup B')$

Proof. Suppose this is not the case. That is, there exists an edge $x = (p, q) \in \delta^+(Q)$ such that $q \in (B \cap B')$ and there is no path from q to $N^-(Q') \cap (B \cap B')$ which completely lies in the graph $D[B \cap B']$. Consider the graph $D_1 = D \setminus (\delta^+(Q) \setminus \{x\})$. Since $\delta^+(Q)$ is a minimum L - L' separator, there is path from L to L' in D_1 . Since this path contains the arc x , there is a subpath, say W , from q to $N^-(Q')$ that does not intersect Q' . Furthermore, in D_1 the only arc that emanates from Q is x and thus we have that W is disjoint from Q as well. However, by Observation 3.3 every path from Q to Q' with the internal vertices disjoint from Q and Q' is contained in $D[B \cup B']$ or $D[V \setminus (B \cap B')]$. Since $q \in B \cap B'$, we conclude that there is a path from q to $N^-(Q') \cap (B \cap B')$ in $D[B \cap B']$. This is a contradiction to our assumption and thus concludes the proof. \square

We are now ready to describe the cases that occur when we *have* an irregular set. This case is divided into following three exhaustive subcases.

Case I: $A \cap B' = \emptyset$.

Case II: $A \cap B' \neq \emptyset$ and $A \setminus B' \neq \emptyset$.

Case III: $A \subseteq B \cap B'$.

We now consider each case one by one and show how we will handle it algorithmically (later).

Case I: $A \cap B' = \emptyset$. We start by defining the set $Q^\circ = \delta^+(Q) \cap \delta^-(B \cap B')$.

The next claim proves an interesting structural property that is crucial to the correctness of the algorithm.

Claim 3.4. *Either every (L, k) -component $Z \supseteq Q$ is such that $Q^\circ \subseteq \delta^+(Z)$ or there is an arc $y \in Q^\circ$ such that there is a minimum L - L' separator containing y and y' .*

Proof. Let $Z \supseteq Q$ be an (L, k) -component and $x \in Q^\circ$ an arc such that $x = (p, q) \notin \delta^+(Z)$. Note that this implies that $x \in A[Z]$. By Claim 3.3, we have that there is a path from q to $N^-(Q') \cap (B \cap B')$ which lies in the graph $D[B \cap B']$. Observe that $N^-(Q') \cap (B \cap B') =$

Tail $((Q^o)')$. Let $r \in B \cap B'$ be such that q has a path in $D[B \cap B']$ from q to r , where $(r, s) \in (Q^o)'$. We claim that there is a minimum L - L' separator containing (r, s) and (s', r') . Consider a minimum Z - Z' separator S . Since every arc in $A[B \cap B']$ has both endpoints inside B and both endpoints outside A , none of these arcs appear in the set $\bigcup_{i=1}^q \delta^+(X_i)$, we conclude that S is disjoint from $A[B \cap B']$ (by property 4 of the collection). Since S is disjoint from $A[B \cap B']$, we have that $r \in Z$. Furthermore, since $s \in Q$, we have that $s \in Z'$. Consequently, we have that $r' \in Z'$ and $s' \in Z$. Therefore, both the arcs (r, s) and (s', r') are in both the sets $\delta^+(Z)$ and $\delta^-(Z')$, which implies that they are also present in S . Since S is also a minimum L - L' separator, this completes the proof of the claim. \square

The following claims allow our algorithm to use the above observations recursively in the graph $D \setminus (B \cap B')$ in the absence of $y \in Q^o$ such that there is no minimum L - L' separator containing y and its conjugate y' .

Claim 3.5. *Assume that every (L, k) -component $Z \supseteq Q$ is such that $Q^o \subseteq \delta^+(Z)$. (Recall $Q^o = \delta^+(Q) \cap \delta^-(B \cap B')$.)*

1. *If S is a minimum Q - Q' separator in $D_1 = D[V \setminus (B \cap B')]$, then $S \cup Q^o$ is a minimum Q - Q' separator in D .*
2. *If Z is an (L, k) -component such that $Q^o \subseteq \delta^+(Z)$, then Z is also an $(L, k - |Q^o|)$ -component in $D_1 = D[V \setminus (B \cap B')]$ and furthermore, any $(L, k - |Q^o|)$ -component in D_1 is an (L, k) -component in D .*

Proof. Recall that $\delta^+(B) \cap \delta^+(B') = \emptyset$ and $\delta^-(B) \cap \delta^-(B') = \emptyset$. Hence every Q - Q' path in D is contained entirely in the graph D_1 or $D_2 = D[B \cup B']$. The last assertion implies that the size of the minimum Q - Q' separator in D is the sum of the sizes of the minimum Q - Q' separator in the graphs D_1 and D_2 . Since $\delta^+(Q) \setminus Q^o$ is a minimum Q - Q' separator in D_1 , S is no larger than $\delta^+(Q) \setminus Q^o$. Therefore, $S \cup Q^o$ is no larger than $\delta^+(Q)$. Since $S \cup Q^o$ is clearly a Q - Q' separator in D , this completes the proof of this statement.

Now we show the second part of the claim. We first show that Z satisfies the first 4 properties of an $(L, k - |Q^o|)$ -component in D_1 . Recall that the size of a minimum Q - Q' separator in the graph D_1 is $|\delta^+(Q)| - |Q^o|$, which implies that Z indeed satisfies the first 4 properties of an $(L, k - |Q^o|)$ -component in D_1 .

Therefore, if Z were not an $(L, k - |Q^o|)$ -component in D_1 , then there is a set $W \supset Z$ which satisfies these 4 properties and let S be a minimum W - W' separator in D_1 . We claim that W also satisfies the first 4 properties of an (L, k) -component in D , which contradicts our assumption of Z as an (L, k) -component. From the first part of the claim, we have that $S \cup Q^o$ is a minimum Q - Q' separator in D , which implies that W indeed satisfies the first 4 properties of an (L, k) -component in D .

In the converse direction, let Z be an $(L, k - |Q^o|)$ -component in D_1 . Since $S \cup Q^o$ is a minimum Q - Q' separator in D , we have that Z satisfies the first 4 properties of an (L, k) -component in D . For contradiction assume that Z is not an (L, k) -component in D . Then there exists $W \supset Z$ that is a (L, k) component in D . Since every (L, k) -component W is such that $Q^o \subseteq \delta^+(W)$, we have that $W \cap \mathbf{Head}(Q^o) = \emptyset$. But then it implies that W also satisfies the first 4 properties of an $(L, k - |Q^o|)$ -component in D_1 . However this is a contradiction to our assumption that Z is an $(L, k - |Q^o|)$ -component in D_1 . This completes the proof of the claim. \square

This completes the study of the case when $A \cap B' = \emptyset$.

Case II : $A \cap B' \neq \emptyset$ and $A \setminus B' \neq \emptyset$.

Here, for a subcase we construct a new collection \mathcal{X}' where this case is avoided. Let $Q = B \setminus B'$ and $P = A \cup (B \setminus B')$. We have already observed that $|\delta^+(Q)| = |\delta^+(B)|$ and that the set $\delta^+(B) \cap \delta^+(B')$ is empty. We now show that $|\delta^+(Q)| = |\delta^+(P)|$. Let $P_1^o = \delta^+(P) \setminus \delta^+(Q)$ and $Q_1^o = \delta^+(Q) \setminus \delta^+(P)$. We claim that $|P_1^o| = |Q_1^o|$. But this is true since $|P_1^o| < |Q_1^o|$ implies that $|\delta^+(P)| < |\delta^+(Q)|$, which is a contradiction since $\delta^+(P)$ is an L - L' separator smaller than the minimum one and $|Q_1^o| < |P_1^o|$ implies that $|\delta^+(A \setminus B')| < |\delta^+(A)|$, which is a contradiction since $\delta^+(A \setminus B)$ is an L - L' separator smaller than the minimum one. Therefore, we conclude that $|P_1^o| = |Q_1^o|$ and hence $|\delta^+(Q)| = |\delta^+(P)|$.

By combining this with an application of the crossing-uncrossing lemma (Lemma 3.2) on the set $K = B \setminus A'$, we have that $|\delta^+(P)| = |\delta^+(K \setminus K')| = |\delta^+(K)|$. Furthermore, $A \subset K \subset B$, and $A \cap K' = \emptyset$.

If K is irregular, then consider the collection \mathcal{X}' obtained by inserting the set K between X_a and X_{a+1} in the collection \mathcal{X} . Clearly, \mathcal{X}' is also a collection which satisfies the properties 1, 2, and 4 of Lemma 3.3. It also satisfies property 3 since $\delta^+(K)$ is a minimum L - L' separator. However, if we consider the collection \mathcal{X}' , A would still be the regular set with the highest index, while K would be the irregular set with the least index. Since A is disjoint from K' , we fall back into the previous case when we consider this collection.

If K is regular, then $(B \cap B') \setminus (A \cup A') = \emptyset$. Observe that $N^+(Q) \cap (B \cap B') \subseteq A \cap B'$ and $N^-(Q') \cap (B \cap B') \subseteq A' \cap B$. Since $A \cap B' \neq \emptyset$ and every vertex in A is reachable from L , we have that $N^+(Q) \cap (B \cap B') \neq \emptyset$ and thus $N^-(Q') \cap (B \cap B') \neq \emptyset$. This together with Claim 3.3 implies that there is a path, say W , from $A \cap B'$ to $A' \cap B$ which lies in the graph $D[B \cap B']$. Let p be the last vertex on W from $A \cap B'$. Since $A \cap B'$ and $A' \cap B$ partition $B \cap B'$, either there is an arc (p, q) such that $p \in A \cap B'$ and $q \in A' \cap B$ or $p \in A \cap B'$ and $q \in B'$. In either case this implies that there is an arc (q', p') where $q' \in A \cap B'$ in the former case and $q \in B'$ in the latter case and $p' \in A' \cap B$. Since both (p, q) and (q', p') are in $\delta^+(K) \cap \delta^-(K')$, and $\delta^+(K)$ is a minimum L - L' separator, we have that $\delta^+(K)$ is a minimum L - L' separator containing arcs y and y' where $y = (p, q)$.

Case III: $A \subseteq B \cap B'$. This case is non-existent since $L \subseteq A$ and $L \cap B' = \emptyset$.

The Algorithm We begin by applying the Ford Fulkerson algorithm to compute a minimum L - L' separator in the graph. If we require more than $2k$ iterations of the Ford Fulkerson algorithm, then we return that there is no (L, k) -component. We then apply the algorithm of Lemma 3.3 to compute the collection $\mathcal{X} = \{X_1, \dots, X_q\}$ which can be computed in time $\mathcal{O}(k(m+n))$. In order to apply Lemma 3.3, we need vertices s and t . Therefore, we add vertices s and t to the graph, add $2k+1$ arcs from s to each vertex in L and $2k+1$ arcs from each vertex in L to t . It is clear from this construction that no s - t separator of size at most $2k$ will contain any of these newly added arcs and therefore, the s - t separators of size at most $2k$ are in one to one correspondence with the L - L' separators of size at most $2k$. This allows us to use Lemma 3.3 in the form it is stated in.

We then simply need to examine each $X_{i+1} \setminus X_i$ once to compute the index a such that X_a is the highest regular set and X_{a+1} is the least irregular set. After computing the index a , in $\mathcal{O}(m)$ time, we can compute the case we are currently in by computing the intersection of the sets A and B . If we are in case (b), in time $\mathcal{O}(k^2m)$, we can compute K and either find an irregular minimum L - L' separator by testing if there is a minimum L - L' separator containing y, y' for some $y \in \delta^+(K)$ or move to case (a) where we already have computed the required

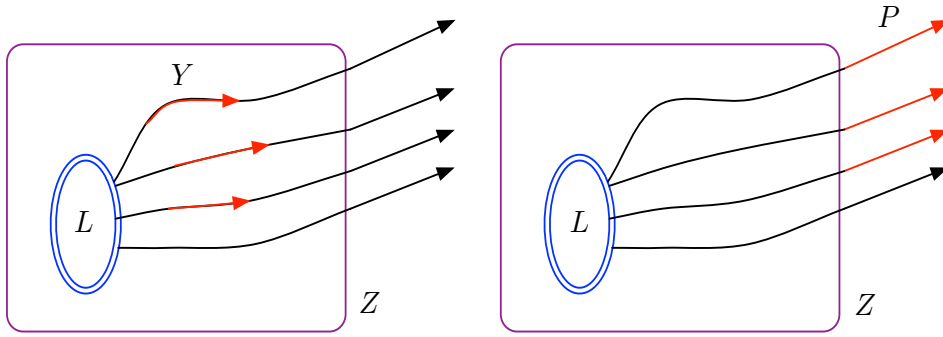


Figure 5: An illustration of the sets Z , Y and P in the proof of Lemma 4.1.

sets— the regular set with the highest index A and the irregular set with the least index K . Finally, if we are in case (a), in $\mathcal{O}(m)$ time, we compute the set Q and iteratively compute a (Q, k') -component in $D \setminus (B \cap B')$ (which we have already shown is an (L, k) -component) where $k' < k$ or an irregular minimum Q - Q' separator. In the latter case, we add Q^o to this minimum separator to get an irregular minimum Q - Q' separator in D . The correctness of our algorithm follows from the structural claims preceding the description. Furthermore, each time we iterate, we attempt to compute an (L, k') -component where $k' < k$. Therefore, we can have at most $2k$ such iterations and hence, the running time of our algorithm is bounded by $\mathcal{O}(k^3(m+n))$. This completes the proof of the lemma. \square

4 Algorithm for d -SKEW-SYMMETRIC MULTICUT

In this section we design our linear time parameterized algorithm for d -SKEW-SYMMETRIC MULTICUT. We first give a lemma which allows us to find a solution that is disjoint from some part of the solution. More formally we show the following.

Lemma 4.1. *Let $(D = (V, A), \sigma, \mathcal{T}, k)$ be a YES instance of d -SKEW-SYMMETRIC MULTICUT and let L be a regular set of vertices such that there is an L - L' path in D . If there is a solution for the given instance which is an L - L' self-conjugate separator in D , then the following hold.*

1. *An (L, k) -component exists.*
2. *Let $Z \subseteq V$ be a regular set of vertices containing L such that the size of the minimum Z - Z' separator is the same as the size of the minimum L - L' separator. Then, there is a solution for the given instance disjoint from $A[Z]$.*

Proof. For the first statement, observe that since there is a solution for the given instance which is an L - L' self-conjugate separator, the size of the minimum L - L' separator is at most $2k$. Therefore the set L itself satisfies the first 4 properties of an (L, k) -component and therefore an (L, k) -component exists. This completes the proof for the first statement.

Now we prove the second part of the lemma. Let X be the solution defined above, that is, X is an L - L' self-conjugate separator. If X is disjoint from $A[Z]$, then we are done. Therefore, suppose that X intersects $A[Z]$ and let Y be the set of arcs in $A[Z]$ such that $Y \cup Y' \subseteq X$. Let $P \subseteq \delta^+(Z)$ be such that $\mathbf{Tail}(P)$ is not reachable from L in $D[Z] \setminus Y$. That is, Y is an L - $\mathbf{Tail}(P)$ separator in the digraph $D[Z]$. It might be possible that $\mathbf{Tail}(P) = \emptyset$.

We now claim that the set $\widehat{X} = (X \setminus (Y \cup Y')) \cup (P \cup P')$ (see Figure 5) is also a solution for the given instance.

Claim 4.1. $\widehat{X} = (X \setminus (Y \cup Y')) \cup (P \cup P')$ is a solution for the given instance and $|\widehat{X}| \leq |X|$.

Proof. We first show that $|\widehat{X}| \leq |X|$. Since $\delta^+(Z)$ is a minimum L - L' separator, there are $|\delta^+(Z)|$ arc disjoint paths from L to $\mathbf{Tail}(\delta^+(Z))$ in $D[Z]$. Therefore, $|Y| \geq |P|$. Clearly, \widehat{X} is no larger than X . Therefore, it remains to show that \widehat{X} is a skew-symmetric multicut for D . If this were not the case, then there is a closed walk in the graph $D \setminus \widehat{X}$ intersecting an arc $y \in Y$ and containing vertices t and t' . Here $t \in J$ where $J \in \mathcal{T}$ and for every vertex $v_i \in J$, v_i and $\sigma(v_i)$ lie in the same strongly connected components of $D \setminus \widehat{X}$. That is, J is a violated constraint.

Since $\mathbf{Tail}(y)$ is reachable from L in the graph $D \setminus \widehat{X}$, both t and t' are reachable from L in the graph $D \setminus \widehat{X}$, which implies the presence of a path, say W , from L to L' in $D \setminus \widehat{X}$. Now using this path we construct another path W' from L to L' in $D \setminus X$. This will contradict our assumption that X is an L - L' self-conjugate separator in D .

Observe that W must also intersect an arc in $\delta^+(Z)$ since $L \subseteq Z$ and L' is disjoint from Z . However, since $P \cup P' \subseteq \widehat{X}$, this arc, say (p, z_1) , is in $\delta^+(Z) \setminus P$. Furthermore, this path also contains a subpath from a vertex $z_1 \in \mathbf{Head}(\delta^+(Z) \setminus P)$ to a vertex $z_2 \in \mathbf{Tail}(\delta^-(Z') \setminus P')$ whose arcs are disjoint from $A[Z \cup Z'] \cup \delta^+(Z) \cup \delta^-(Z')$. We call this path $W_{z_1 z_2}$. Let (z_2, q) be an arc in $\delta^-(Z') \setminus P'$ such that the arc (z_2, q) is on W . Observe that p, q are in Z and there are paths from L to both p and q that avoids arcs of Y . The last assertion follows from the fact that the vertices of $\delta^+(Z) \setminus P$ are reachable from L in $D[Z] \setminus Y$. Let these paths avoiding arcs of Y be called W_{Lp} and W_{Lq} . Then observe that $W_{Lp}W_{z_1 z_2}(W_{Lq})'$ forms a path in $D \setminus X$ – a contradiction. Here, $(W_{Lq})'$ is the path that is conjugate to W_{Lq} . This completes the proof of the claim. \square

The above claim completes the proof of the lemma. \square

From this point on, we assume that an instance of d -SKEW-SYMMETRIC MULTICUT is of the form $(D = (V, A), \sigma, T, k, L)$ where L is a regular set of vertices and the question is to check if there is a solution for the given instance which is an L - L' self-conjugate separator. To solve the problem on the given input instance, we simply solve it on the instance $(D = (V, A), \sigma, \mathcal{T}, k, \emptyset)$.

Lemma 4.2. Let $(D = (V, A), \sigma, \mathcal{T}, k, L)$ be an instance of d -SKEW-SYMMETRIC MULTICUT and let S be an irregular minimum L - L' separator. Then, there are arcs y, y' such that $y, y' \in \delta^+(R(L, S \cup S'))$ and there is a solution for the given instance containing y, y' .

Proof. Let $Z = R(L, S \cup S')$. By Lemma 3.1, $\delta^+(Z)$ is a minimum L - L' separator and hence $|\delta^+(Z)| = |S|$. Since S is irregular, $S \cup S'$ contains arcs from at most $|S| - 1$ conjugate pairs. Thus there are arcs $y, y' \in \delta^+(Z)$.

Let X be a solution for the given instance. Since there is no path from L - L' in $D \setminus X$, it must be the case that $D \setminus X$ cannot contain paths from L to both $\mathbf{Tail}(y)$ and $\mathbf{Tail}(y')$. Therefore, it must be the case that X intersects $A[Z]$ and intersects all paths from L to $\mathbf{Tail}(y)$ or $\mathbf{Tail}(y')$. However, by Lemma 4.1 we know that there exists a solution that does not intersect $A[Z]$. This implies that there is also a solution containing the arcs y, y' . \square

The above lemma gives us the following reduction rule.

Reduction Rule 1. Let $(D = (V, A), \sigma, T, k, L)$ be an instance of d -SKEW-SYMMETRIC MULTICUT and let S be an irregular minimum L - L' separator. Let y, y' be the arcs given by Lemma 4.2. Then return the instance $(D = (V, A \setminus \{y, y'\}), \sigma, \mathcal{T}, k - 1)$.

Therefore, by combining this reduction rule with the algorithm of Lemma 3.4, we can, in linear time either reduce the parameter or compute an (L, k) -component with a regular neighborhood. We are now ready to prove Theorem 1 by giving an algorithm for d -SKEW-SYMMETRIC MULTICUT.

Description of Algorithm. The input to our algorithm for d -SKEW-SYMMETRIC MULTICUT is an instance $(D = (V, A), \sigma, \mathcal{T} = \{J_1, \dots, J_r\}, k, L)$ where $J_i = \{v_{i_1}, \dots, v_{i_d}\}$ and the algorithm either returns a skew-symmetric multicut of size at most $2k$ which is an L - L' self-conjugate separator in D or concludes correctly that no such set exists. In order to solve the problem on the given instance of d -SKEW-SYMMETRIC MULTICUT, the algorithm is invoked on the input $(D = (V, A), \sigma, \mathcal{T}, k, \emptyset)$.

1. If $L = \emptyset$ or if there is no path from L to L' in D , then the algorithm checks if there is a set $J_i \in \mathcal{T}$ such that for all $1 \leq s \leq d$, v_{i_s} and v'_{i_s} lie in the same strongly connected component in D , that is, a violated set. If there is no such set, then the algorithm returns the emptyset. Otherwise, the algorithm picks such a set J_i , and branches in $2d$ ways. In the first d branches, it recurses on the instances $\{(D = (V, A), \sigma, \mathcal{T}, k, \{v_{i_j}\})\}_{1 \leq j \leq d}$ and in the next d branches, it recurses on the instances $\{(D = (V, A), \sigma, \mathcal{T}, k, \{v'_{i_j}\})\}_{1 \leq j \leq d}$.

2. Suppose $L \neq \emptyset$ and there is an L - L' path in D . Then, the algorithm of Lemma 3.4 is first used on the instance $(D = (V, A), \sigma, \mathcal{T}, k, L)$ to either compute an (L, k) -component (if it exists) or an irregular minimum L - L' separator. If an (L, k) -component does not exist, then we return NO. If an irregular minimum L - L' separator is returned, then we apply Reduction Rule 1. Suppose that an (L, k) -component Z is returned. We check if Reduction Rule 1 applies on any arc in $\delta^+(Z)$ and if it does, apply the rule. Therefore, at this point, we may assume that an (L, k) -component Z is returned and that the rule is not applicable on any arc in $\delta^+(Z)$. Observe that $\delta^+(Z) \neq \emptyset$ since there is an L - L' path in D . The algorithm then picks an arc $a \in \delta^+(Z)$ and branches in 2 ways as follows. In the first branch, the algorithm deletes $\{a, a'\}$ and recurses on the resulting instances, that is, the algorithm recurses on the instance $(D = (V, A \setminus \{a, a'\}), \sigma, \mathcal{T}, k - 1, L)$. In the next branch, the algorithm recurses on the instance assuming that a is in $A[R(L, X)]$ where X is the hypothetical solution, that is, the algorithm recurses on the instance $(D = (V, A), \sigma, \mathcal{T}, k, L \cup \{\text{Head}(a)\})$.

Correctness. The correctness of the algorithm is proved by induction on a measure defined on the instance I . Let this measure be denoted by $\mu(I) = \mu((D, k), L) = 2k - \lambda(L, L')$ where $\lambda(L, L')$ is the size of the smallest L - L' separator. In the base case, if $\lambda(L, L') > 2k$, then the algorithm of Lemma 3.4 returns NO on input $(D = (V, A), \sigma, k, L)$, and hence this algorithm returns NO as well, which is correct since the solution we require contains an L - L' separator of size at most $2k$. Similarly, the case when $k < 0$ is also clearly correct. We now assume as induction hypothesis that the algorithm is correct on all instances I such that $\mu(I) \leq \mu - 1$ and consider an instance $I = (D = (V, A), \sigma, \mathcal{T}, k, L)$ such that $\mu(I) = \mu$ and $k \geq 0$.

We first show that an application of Reduction Rule 1 does not decrease this measure. Since deleting an arc and its conjugate from an irregular minimum L - L' separator reduces the size of the minimum size L - L' separator by 2 and the budget k by 1, the measure $2k - \lambda(L, L')$ remains unchanged.

We now consider the branching rules. If $L = \emptyset$ or there is no L - L' path in D , then $\lambda(L, L') = 0$. Consider an instance $I' = (D = (V, A), \sigma, \mathcal{T}, k, \{v\})$ resulting from a branch here. Although the parameter has not decreased here, since there is a path from v to v' , $\lambda(v, v') > 0$, which implies that $\mu(I') < \mu(I)$. Therefore, by combining the exhaustiveness of the branching along with the induction hypothesis, we obtain the correctness of the algorithm on the instance I as well.

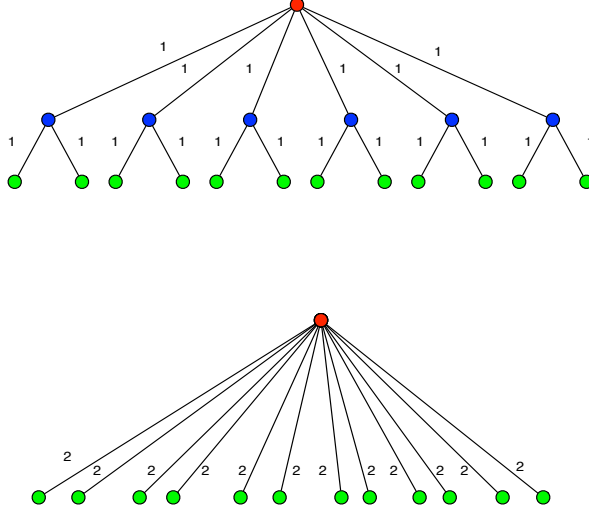


Figure 6: An illustration of the tighter analysis of the search tree in the algorithm.

We now suppose that $L \neq \emptyset$ and there is an L - L' path in D . The branching is exhaustive due to Lemma 4.1. We now show that for each of the resulting instances I' from any of the branches, $\mu(I') \leq \mu - 1$ in which case we can apply the induction hypothesis on these instances, thus proving the correctness of the algorithm.

(a) We begin with the branch where we recurse on the instance $I' = (D = (V, A), \sigma, \mathcal{T}, k, Z \cup \{\mathbf{Head}(a)\})$. Since Z is an (L, k) -component, by definition, the size of the smallest separator from $Z \cup \{\mathbf{Head}(a)\}$ to $Z' \cup \{(\mathbf{Head}(a))'\}$ is strictly larger than $\lambda(L, L')$, which implies that $\mu(I') < \mu(I)$.

(b) We now consider the instance resulting from the other branch, that is the instance $I' = (D = (V, A \setminus \{a, a'\}), \sigma, \mathcal{T}, k - 1, L)$ where $a \in \delta^+(Z)$. In this case, the parameter has decreased by 1. Since Reduction Rule 1 is not applicable on $\delta^+(Z)$, removing $\{a, a'\}$ from the graph reduces $\lambda(L, L')$ by at most 1 and therefore, we have that $\mu(I') < \mu$. This completes the proof of correctness of the algorithm.

Running time. We prove that on an instance $I = (D = (V, A), \sigma, \mathcal{T}, k, L)$, the algorithm computes a search tree with at most $(2\sqrt{d})^{\mu(I)}$ leaves. We have already proved that in each branch, the measure $\mu(I)$ decreases by at least 1 and since we only have $2d$ -way branchings, the number of nodes of the search tree is clearly bounded by $(2d)^{\mu(I)}$. However, we analyze more closely a branch which occurs in a $2d$ -way branching where $\mu(I)$ decreases by exactly 1. Suppose that J was the violating set computed in this step and suppose $v \in J$ be such that $\lambda(v, v') = 1$. Consider the branch where we recurse on the instance $(D = (V, A), \sigma, \mathcal{T}, k, \{v\})$. In this recursion, we first observe that the reduction rule will not be applied. This is because the reduction rule requires a minimum $\{v\}$ - $\{v'\}$ separator of size at least 2 while $\lambda(v, v') = 1$. Therefore, the algorithm of Lemma 3.4 will find a $(\{v\}, k)$ -component Z .

Claim 4.2. $|\delta^+(Z)| = 1$.

Proof. Suppose not and let $(a, b), (p, q) \in \delta^+(Z)$. Furthermore, let S be a minimum Z - Z' separator. Then, since $|S| = 1$ by our assumption, either $(a, b) \notin S$ or $(p, q) \notin S$. Suppose

that $(a, b) \notin S$. Then, S is also a minimum $Z \cup \{b\}$ - $Z' \cup \{b'\}$ separator, which contradicts the maximality of Z as a $(\{v\}, k)$ -component. \square

Consider the branching performed on the single arc in $\delta^+(Z)$. We have already shown that the measure decreases by 1 in each of these branchings. Therefore, we combine these 2 branches with the branch where we decided to recurse on the instance $(D = (V, A), \sigma, \mathcal{T}, k, \{v\})$. This leads to 2 branches where the measure decreases by 2 in each. For each branch in the $2d$ -way branching where the measure decreases by 1, we can do the same to obtain at most $4d$ branches in each of which the measure decreases by 2 (see Figure 6). Therefore, our worst case branching is a $4d$ -way branching, where the measure drops by 2 in each branch. Thus, we obtain a bound of $(2\sqrt{d})^{\mu(I)} \leq (4d)^k$ on the number of leaves of the search tree.

Since finding a violating set can be done in time $\mathcal{O}(m + n + \ell)$, computing an (L, k) -component or an irregular minimum L - L' separator can be done in time $\mathcal{O}(k^3(m + n))$ and there can be at most k applications of Reduction Rule 1 along any root to leaf path of this search tree, we have the claimed bound of $\mathcal{O}((4d)^k k^4 (m + n + \ell))$ on the running time, completing the proof of Theorem 1. \square

We observe here that ℓ occurs in the running time simply because the time taken to compute a violating set is $\mathcal{O}(\ell + m)$. However, in some cases, the set \mathcal{T} may be given in the form of a violation oracle, in which case, the running time bound remains the same if violation oracle runs in time $\mathcal{O}(\ell)$. Therefore, the theorem below follows from the proof of Theorem 1.

Theorem 2. *There is an algorithm for d -SKEW-SYMMETRIC MULTICUT that, given a tuple $(D = (V, A), \sigma, k)$ along with a violation oracle for a family \mathcal{T} , runs in time $\mathcal{O}((4d)^k k^4 (\ell + m + n))$ and either returns a skew-symmetric multicut of size at most $2k$ or correctly concludes that no such set exists, where ℓ is the time required for the violation oracle to compute a violated set in the family \mathcal{T} , $m = |A|$ and $n = |V|$.*

5 Applications

In this section we use the algorithm developed for d -SKEW-SYMMETRIC MULTICUT to obtain linear time parameterized algorithms for several other problems.

5.1 Linear time algorithm for ALMOST 2-SAT

Algorithms for the ALMOST 2-SAT problem (defined in the introduction) together with its variable version have turned out to be extremely useful as a subroutine in several parameterized algorithms (see [28, 29, 22]). The variable version of the problem is formally defined as follows.

ALMOST 2-SAT(v)	Parameter: k
Input: A 2-CNF formula F , integer k	
Question: Does there exist a set S_v of at most k variables such that by deleting all the clauses which they occur in from F we get a satisfiable formula?	

It is known that the *variable* version ALMOST 2-SAT(v) can be reduced to the *clause* version ALMOST 2-SAT via a linear time reductions [29]. Therefore, it suffices for us to give a reduction from the clause version of ALMOST 2-SAT to d -SKEW-SYMMETRIC MULTICUT. In order to give this reduction, we begin by recalling the notion of implication graphs of a 2-CNF formula.

Definition 5.1. Given a 2-CNF formula F , the **implication graph** of F is denoted by $D(F)$ and is defined as follows. The vertex set of the graph is the set of literals of F and for every clause $\{l_1, l_2\}$ in F , we have arcs (\bar{l}_1, l_2) and (l_2, \bar{l}_1) .

Clearly, implication graphs are skew-symmetric where the involution σ is defined as $\sigma(l) = \bar{l}$ and $\sigma(l_1, l_2) = (\bar{l}_2, \bar{l}_1)$.

Theorem 3. [1] A 2-CNF formula F is satisfiable iff no literal and its complement are contained in the same strongly connected component of $D(F)$.

Observation 5.1. Given a 2-CNF formula F , let D be the implication graph of this formula and let C be a subset of the clauses of F . Let C_D be the corresponding set of arcs in the graph D . Then, the implication graph of $F - C$ is the same as the graph $D \setminus C_D$.

Lemma 5.1. Let F be a 2-CNF formula on n variables x_1, \dots, x_n . Then, (F, k) is a YES instance of ALMOST 2-SAT iff $(D(F), \mathcal{T} = \{\{x_1\}, \dots, \{x_n\}\}, k)$ is a YES instance of 1-SKEW-SYMMETRIC MULTICUT.

Proof. Suppose that C is a set of clauses such that $|C| \leq k$ and $F - C$ is satisfiable and let C_D be the corresponding set of arcs in D . Then, by Theorem 3, no literal of F appears in the same strongly connected component as its complement in the implication graph of $F - C$. However, by Observation 5.1, we have that C_D is a set of arcs such that no vertex and its conjugate lie in the same strongly connected component of $D \setminus C_D$, which implies that C_D is a solution for the instance of 1-SKEW-SYMMETRIC MULTICUT.

Conversely, let C_D be a self-conjugate set of arcs such that $|C_D| \leq 2k$ and no vertex in \mathcal{T} lies in the same strongly connected component as its conjugate in $D \setminus C_D$. Let C be the set of clauses of F corresponding to C_D . Since C_D is self-conjugate, we have that $|C| \leq k$. Then, by Observation 5.1 and Theorem 3, the formula $F - C$ is satisfiable, which implies that C is indeed a solution for the instance of ALMOST 2-SAT. This completes the proof of the lemma. \square

Since the graph $D(F)$ can be constructed in time $\mathcal{O}(|F|)$ and has $\mathcal{O}(|F|)$ arcs, we have the following theorem.

Theorem 4. There is an algorithm that, given an instance (F, k) of ALMOST 2-SAT (ALMOST 2-SAT(v)), runs in time $\mathcal{O}(4^k k^4 \ell)$ and either returns an assignment satisfying all but at most k clauses of F or correctly concludes that no such assignment exists.

Furthermore, there are known linear time parameter preserving reductions from EDGE BIPARTIZATION to OCT and from OCT to ALMOST 2-SAT (see [41, Page Number – 72] and [19]). The reduction from EDGE BIPARTIZATION to OCT increases the the number of edges and vertices in the graph by a factor of $\mathcal{O}(k)$ and the reduction from OCT to ALMOST 2-SAT is both parameter as well as size preserving. Therefore, have the following corollaries.

Theorem 5. EDGE BIPARTIZATION and ODD CYCLE TRANSVERSAL can be solved in time $\mathcal{O}(4^k k^5 (m + n))$ and $\mathcal{O}(4^k k^4 (m + n))$ respectively where m and n are the number of edges and vertices in the input graph respectively.

5.2 Linear time algorithm for DELETION q -Horn BACKDOOR SET DETECTION

A CNF formula F is q -Horn if there is a *certifying function* $\beta : \text{var}(F) \cup \overline{\text{var}(F)} \rightarrow \{0, \frac{1}{2}, 1\}$ with $\beta(x) = 1 - \beta(\bar{x})$ for every $x \in \text{var}(F)$ such that $\sum_{l \in C} \beta(l) \leq 1$ for every clause C of F . In this subsection, we prove Theorem 6. We begin by recalling the notion of a quadratic cover given by Boros et al. [2].

Definition 5.2. Given a CNF formula F , the **quadratic cover** of F , is a Krom formula denoted by F_2 and is defined as follows. Let x_1, \dots, x_n be the variables of F . For every clause C , we have $|C| - 1$ new variables $y_1^C, \dots, y_{|C|-1}^C$. We order the literals in each clause according to their variables, that is, a literal of x_i will occur before a literal of x_j if $i < j$. Let $l_1^C, \dots, l_{|C|}^C$ be the literals of the clause C in this order. The quadratic cover is defined as.

$$F_2 = \bigcup_{C \in F} \bigcup_{1 \leq i \leq |C|-1} \{ \{l_i^C, y_i^C\}, \{\bar{y}_i^C, l_{i+1}^C\} \} \cup \bigcup_{C \in F} \bigcup_{1 \leq i \leq |C|-2} \{ \{\bar{y}_i^C, y_{i+1}^C\} \}.$$

Observation 5.2. If F is a CNF formula of length ℓ , then the number of arcs in $D(F_2)$ is $\mathcal{O}(\ell)$.

We require the following characterization of q -Horn formulas.

Lemma 5.2 ([2]). A CNF formula F is q -Horn iff no clause of F has three literals l_1, l_2, l_3 such that each l_i and \bar{l}_i are in the same strongly connected component of $D(F_2)$.

Recall that a *deletion \mathcal{C} -backdoor set* of F is a set B of variables such that $F - B \in \mathcal{C}$. These characterizations allow us to give a linear time reduction from DELETION q -HORN BACKDOOR SET DETECTION to 3-SKEW-SYMMETRIC MULTICUT.

Theorem 6. There is an algorithm that, given an instance (F, k) of DELETION q -HORN BACKDOOR SET DETECTION, runs in time $\mathcal{O}(12^k k^5 \ell)$ and either returns a deletion q -Horn-backdoor set of F of size at most k or correctly concludes that no such set exists, where ℓ is the length of F .

Proof. The proof is by a reduction to 3-SKEW-SYMMETRIC MULTICUT. We first construct the graph $D(F_2)$. We now define a graph D_1 which is a modification of $D(F_2)$ as follows. For every vertex l_i in $D(F_2)$ corresponding to a positive literal in F , we have two vertices l_i^+ and l_i^- and an arc (l_i^-, l_i^+) and for every vertex l_i in $D(F_2)$ corresponding to a negative literal we have two vertices \bar{l}_i^+ and \bar{l}_i^- and an arc $(\bar{l}_i^+, \bar{l}_i^-)$. We say that an arc (l_i^-, l_i^+) corresponds to a (positive) literal l_i and an arc $(\bar{l}_i^+, \bar{l}_i^-)$ corresponds to a (negative) literal l_i .

Now, for every vertex y in $D(F_2)$ which does not correspond to a literal of F , we add vertices y_1, \dots, y_{k+1} and for every arc (y, l_i) in $D(F_2)$, if l_i is a positive literal, then we add arcs $(y_1, l_i^-), \dots, (y_{k+1}, l_i^-)$ and if l_i is a negative literal, then we add arcs $(y_1, \bar{l}_i^-), \dots, (y_{k+1}, \bar{l}_i^-)$. For every arc (l_i, y) in $D(F_2)$, if l_i is a positive literal then we add arcs $(l_i^+, y_1), \dots, (l_i^+, y_{k+1})$ and if l_i is a negative literal then we add arcs $(\bar{l}_i^+, y_1), \dots, (\bar{l}_i^+, y_{k+1})$. This completes the construction of D_1 . Clearly, D_1 is also skew-symmetric. The purpose of modifying the graph $D(F_2)$ is simply to map literals of the input formula to arcs in the skew-symmetric graph and conversely to ensure that arcs which do not correspond to literals of the formula F are unlikely to participate in skew-symmetric multicuts of size at most k . We note that $\{l_1^+, l_2^+\}$ are contained in the same strongly connected component of D_1 if and only if $\{l_1, l_2\}$ are in the same strongly connected component of $D(F_2)$.

We now claim that (F, k) is a YES instance of DELETION q -HORN BACKDOOR SET DETECTION iff $(D_1, \mathcal{T}, k, \emptyset)$ is a YES instance of 3-SKEW-SYMMETRIC MULTICUT where \mathcal{T} is the set of all triples of literals $\{l_1^+, l_2^+, l_3^+\}$ in F such that l_1, l_2, l_3 occur in a clause in F .

Consider a solution S for the instance of DELETION q -HORN BACKDOOR SET DETECTION and let S_D be the set of those arcs in D_1 which correspond to the literals of the variables in S . Clearly, S_D is self-conjugate and $|S_D| \leq 2k$. We claim that S_D is a skew-symmetric multicut for the given instance. If this were not the case, then there is a clause $C \in \mathcal{C}(F)$ and literals $l_1, l_2, l_3 \in C$ such that l_1^+, l_2^+, l_3^+ each lie in the same strongly connected component of $D_1 \setminus S_D$ as their complements. Recall that by $\mathcal{C}(F)$ we denote the set of clauses of a CNF formula F .

However, by Lemma 5.2, there is no violating triple in the graph $D(F_2) \setminus \text{lit}(S)$ and therefore, there cannot be a violated set in the graph $D_1 \setminus S_D$.

Conversely, consider a solution S_D for the instance of 3-SKEW-SYMMETRIC MULTICUT. It is easy to see from the construction of D_1 that S_D is disjoint from arcs incident on any y_i^C . Therefore, the arcs in S_D correspond to literals and hence variables in F . Let S be the this set of variables. We claim that S is a q -Horn deletion backdoor set. If this were not the case, then by Lemma 5.2, there is a clause $C \in \mathcal{C}(F)$ and literals $l_1, l_2, l_3 \in C$ such that l_1, l_2, l_3 each lie in the same strongly connected component of $D(F_2) \setminus \text{lit}(S)$ as their complements. However, this implies that l_1^+, l_2^+, l_3^+ each lie in the same strongly connected component of $D_1 \setminus S_D$ as their complements, which is a contradiction. This completes the proof of correctness of the reduction.

Though this reduction is parameter preserving, it is not a linear time reduction since the number of triples we need to give as input to the 3-SKEW-SYMMETRIC MULTICUT instance could be super linear in the length of the formula. However, by using an algorithm by Boros et al. [2] that runs in time $\mathcal{O}(\ell)$ and returns a violated triple, we can use the above reduction which runs in time $\mathcal{O}(k\ell)$ and returns a skew-symmetric graph with $\mathcal{O}(k\ell)$ arcs, along with our algorithm for 3-SKEW-SYMMETRIC MULTICUT to get an algorithm which runs in time $\mathcal{O}(12^k k^5 \ell)$. This completes the proof of the theorem. \square

Since every deletion q -Horn backdoor set is also a strong q -Horn backdoor set, Theorem 6 has the following corollary.

Corollary 1. *There is an algorithm for SATISFIABILITY that runs in time $\mathcal{O}(12^k k^5 \ell)$, where k is the size of the smallest q -Horn deletion backdoor set of the input formula.*

6 Conclusions

We introduced the d -SKEW-SYMMETRIC MULTICUT problem, a general graph separation problem which generalizes a large number of well-studied problems, and described an FPT algorithm for this problem with a linear dependence on the input size and a moderate dependence on the parameter. This result gives the first linear time FPT algorithms for OCT, ALMOST 2-SAT and DELETION q -HORN BACKDOOR SET DETECTION. We believe that there are more graph separation problems which can be reduced to d -SKEW-SYMMETRIC MULTICUT and that our algorithm can be used as a “tool” to give (linear time) FPT algorithms for other problems which have graph separation at their core. We would like to remark that, to keep our analysis simple, we have not optimized the polynomial dependence of the running times on k .

We would also like to point out that the algorithms for variants of EDGE BIPARTIZATION and OCT studied in the paper of Marx et al. [26] use the almost linear time algorithm for OCT of Kawarabayashi and Reed or the quadratic time algorithm of Reed et al. [37]. Therefore, using our algorithm instead of these algorithms results in linear time FPT algorithms for these variants studied by Marx et al.

Finally, we leave open the kernelization complexity of this problem. Given that a (randomized) polynomial kernel for ALMOST 2-SAT exists [20], it would be a natural goal to see if such a result extends to this much more general problem.

References

- [1] B. ASPVALL, M. F. PLASS, AND R. E. TARJAN, *A linear-time algorithm for testing the truth of certain quantified boolean formulas*, Inf. Process. Lett., 8 (1979), pp. 121–123.
- [2] E. BOROS, P. L. HAMMER, AND X. SUN, *Recognition of q -Horn formulae in linear time*, Discr. Appl. Math., 55 (1994), pp. 1–13.
- [3] J. CHEN, Y. LIU, AND S. LU, *An improved parameterized algorithm for the minimum node multiway cut problem*, Algorithmica, 55 (2009), pp. 1–13.
- [4] J. CHEN, Y. LIU, S. LU, B. O’SULLIVAN, AND I. RAZGON, *A fixed-parameter algorithm for the directed feedback vertex set problem*, J. ACM, 55 (2008).
- [5] H.-A. CHOI, K. NAKAJIMA, AND C. S. RIM, *Graph bipartization and via minimization*, SIAM Journal on Discrete Mathematics, 2 (1989), pp. 38–47.
- [6] Y. CRAMA, O. EKIN, AND P. L. HAMMER, *Variable and term removal from Boolean formulae*, Discr. Appl. Math., 75 (1997), pp. 217–230.
- [7] M. CYGAN, M. PILIPCZUK, M. PILIPCZUK, AND J. O. WOJTASZCZYK, *Subset feedback vertex set is fixed-parameter tractable*, in ICALP (1), 2011, pp. 449–461.
- [8] —, *On multiway cut parameterized above lower bounds*, IPEC, 2011.
- [9] R. G. DOWNEY AND M. R. FELLOWS, *Parameterized Complexity*, Springer-Verlag, New York, 1999.
- [10] S. FIORINI, N. HARDY, B. A. REED, AND A. VETTA, *Planar graph bipartization in linear time*, Discrete Applied Mathematics, 156 (2008), pp. 1175–1180.
- [11] J. FLUM AND M. GROHE, *Parameterized Complexity Theory*, Texts in Theoretical Computer Science. An EATCS Series, Springer-Verlag, Berlin, 2006.
- [12] L. R. FORD, JR. AND D. R. FULKERSON, *Maximal flow through a network*, Canadian J. Math., 8 (1956), pp. 399–404.
- [13] S. GASPERS, S. ORDYNIK, M. S. RAMANUJAN, S. SAURABH, AND S. SZEIDER, *Backdoors to q -horn*, in STACS, 2013, pp. 67–79.
- [14] S. GASPERS AND S. SZEIDER, *Backdoors to satisfaction*, in The Multivariate Algorithmic Revolution and Beyond - Essays Dedicated to Michael R. Fellows on the Occasion of His 60th Birthday, vol. 7370 of Lecture Notes in Computer Science, Springer Verlag, 2012, pp. 287–317.
- [15] A. V. GOLDBERG AND A. V. KARZANOV, *Path problems in skew-symmetric graphs*, Combinatorica, 16 (1996), pp. 353–382.
- [16] —, *Maximum skew-symmetric flows and matchings*, Math. Program., 100 (2004), pp. 537–568.
- [17] F. HÜFFNER, *Algorithm engineering for optimal graph bipartization*, J. Graph Algorithms Appl., 13 (2009), pp. 77–98.
- [18] K. ICHI KAWARABAYASHI AND B. A. REED, *An (almost) linear time algorithm for odd cycles transversal*, in SODA, 2010, pp. 365–378.

- [19] S. KHOT AND V. RAMAN, *Parameterized complexity of finding subgraphs with hereditary properties*, Theor. Comput. Sci., 289 (2002), pp. 997–1008.
- [20] S. KRATSCH AND M. WAHLSTRÖM, *Representative sets and irrelevant vertices: New tools for kernelization*, in FOCS, 2012, pp. 450–459.
- [21] D. LOKSHTANOV, N. S. NARAYANASWAMY, V. RAMAN, M. S. RAMANUJAN, AND S. SAURABH, *Faster parameterized algorithms using linear programming*, CoRR, abs/1203.0833 (2012).
- [22] D. LOKSHTANOV AND M. S. RAMANUJAN, *Parameterized tractability of multiway cut with parity constraints*, in ICALP (1), 2012, pp. 750–761.
- [23] D. LOKSHTANOV, S. SAURABH, AND S. SIKDAR, *Simpler parameterized algorithm for oct*, in IWOCA, 2009, pp. 380–384.
- [24] M. MAHAJAN AND V. RAMAN, *Parameterizing above guaranteed values: MaxSat and MaxCut*, J. Algorithms, 31 (1999), pp. 335–354.
- [25] D. MARX, *Parameterized graph separation problems*, Theoret. Comput. Sci., 351 (2006), pp. 394–406.
- [26] D. MARX, B. O’SULLIVAN, AND I. RAZGON, *Finding small separators in linear time via treewidth reduction*, Transactions on Algorithms, To Appear.
- [27] ———, *Treewidth reduction for constrained separation and bipartization problems*, in STACS, 2010, pp. 561–572.
- [28] D. MARX AND I. RAZGON, *Constant ratio fixed-parameter approximation of the edge multicut problem*, Inf. Process. Lett., 109 (2009), pp. 1161–1166.
- [29] ———, *Fixed-parameter tractability of multicut parameterized by the size of the cutset*, in STOC, 2011, pp. 469–478.
- [30] N. S. NARAYANASWAMY, V. RAMAN, M. S. RAMANUJAN, AND S. SAURABH, *Lp can be a cure for parameterized problems*, in STACS, 2012, pp. 338–349.
- [31] R. NIEDERMEIER, *Invitation to Fixed-Parameter Algorithms*, vol. 31 of Oxford Lecture Series in Mathematics and its Applications, Oxford University Press, Oxford, 2006.
- [32] N. NISHIMURA, P. RAGDE, AND S. SZEIDER, *Detecting backdoor sets with respect to Horn and binary clauses*, in SAT, 2004, pp. 96–103.
- [33] A. PANCONESI AND M. SOZIO, *Fast hare: A fast heuristic for single individual snp haplotype reconstruction*, in Algorithms in Bioinformatics, Springer, 2004, pp. 266–277.
- [34] V. RAMAN, M. S. RAMANUJAN, AND S. SAURABH, *Paths, flowers and vertex cover*, in ESA, vol. 6942 of Lecture Notes in Computer Science, 2011, pp. 382–393.
- [35] I. RAZGON AND B. O’SULLIVAN, *Almost 2-sat is fixed-parameter tractable (extended abstract)*, in ICALP(1), 2008, pp. 551–562.
- [36] ———, *Almost 2-sat is fixed-parameter tractable.*, J. Comput. Syst. Sci., 75 (2009), pp. 435–450.

- [37] B. A. REED, K. SMITH, AND A. VETTA, *Finding odd cycle transversals*, Oper. Res. Lett., 32 (2004), pp. 299–301.
- [38] R. RIZZI, V. BAFNA, S. ISTRAIL, AND G. LANCIA, *Practical algorithms and fixed-parameter tractability for the single individual snp haplotyping problem*, in Algorithms in Bioinformatics, Springer, 2002, pp. 29–43.
- [39] R. E. TARJAN, *Efficiency of a good but not linear set union algorithm*, J. ACM, 22 (1975), pp. 215–225.
- [40] W. T. TUTTE, *Antisymmetrical digraphs*, Canadian J. Math., 19 (1967), pp. 1101–1117.
- [41] S. WERNICKE, *On the algorithmic tractability of single nucleotide polymorphism (snp) analysis and related problems*, Master’s thesis, Wilhelm-Schickard-Institut für Informatik, Universität Tübingen, (2003).
- [42] R. WILLIAMS, C. GOMES, AND B. SELMAN, *Backdoors to typical case complexity*, in IJCAI, 2003, pp. 1173–1178.
- [43] T. ZASLAVSKY, *Orientation of signed graphs*, European Journal of Combinatorics, 12 (1991), pp. 361–375.
- [44] B. ZELINKA, *Analoga of Menger’s theorem for polar and polarized graphs*, Czechoslovak Math. J., 26(101) (1976), pp. 352–360.