Odd Multiway Cut in Directed Acyclic Graphs*

Karthekeyan Chandrasekaran[1] and Sahand Mozaffari[2]

1 **University of Illinois, Urbana-Champaign, USA**
karthe@illinois.edu
2 **University of Illinois, Urbana-Champaign, USA**
sahandm2@illinois.edu

Abstract

We investigate the odd multiway node (edge) cut problem where the input is a graph with a specified collection of terminal nodes and the goal is to find a smallest subset of non-terminal nodes (edges) to delete so that the terminal nodes do not have an odd length path between them. In an earlier work, Lokshtanov and Ramanujan showed that both odd multiway node cut and odd multiway edge cut are fixed-parameter tractable (FPT) when parameterized by the size of the solution in *undirected graphs*. In this work, we focus on directed acyclic graphs (DAGs) and design a fixed-parameter algorithm. Our main contribution is an extension of the shadow-removal framework for *parity problems in DAGs*. We complement our FPT results with tight approximability as well as polyhedral results for 2 terminals in DAGs. Additionally, we show inapproximability results for odd multiway *edge* cut in undirected graphs even for 2 terminals.

1998 ACM Subject Classification G.2.2 Graph Theory, I.1.2 Algorithms

Keywords and phrases Odd Multiway Cut, Fixed-Parameter Tractability, Approximation Algorithms

Digital Object Identifier 10.4230/LIPIcs.IPEC.2017.12

1 Introduction

In the classic $\{s,t\}$-cut problem, the goal is to delete the smallest number of edges so that the resulting graph has no path between s and t. A natural generalization of this problem is the multiway cut, where the input is a graph with a specified set of terminal nodes and the goal is to delete the smallest number of non-terminal nodes/edges so that the terminals cannot reach each other in the resulting graph. In this work, we consider a parity variant of this problem. A path[1] is an odd-path (even-path) if the number of edges in the path is odd (even). In the OddMultiwayNodeCut (similarly, OddMultiwayEdgeCut), the input is a graph with a collection of terminal nodes and the goal is to delete the smallest number of non-terminal nodes (edges) so that the resulting graph has no odd-path between the terminals. This is a generalization of $\{s,t\}$-OddPathNodeBlocker (and similarly, $\{s,t\}$-OddPathEdgeBlocker), which is the problem of finding a minimum number of nodes that are disjoint from s and t (edges) that cover all $s-t$ odd-paths.

Covering and packing paths has been a topic of intensive investigation in graph theory as well as polyhedral theory. Menger's theorem gives a perfect duality relation for min

* A full version of the paper is available at https://arxiv.org/abs/1708.02323.
[1] We emphasize that the term *paths* refers to simple paths and not walks. This distinction is particularly important in parity-constrained settings, because the existence of a walk with an odd number of edges between two nodes s and t does not imply the existence of an odd-path between s and t. This is in contrast to the non-parity-constrained settings where the existence of a walk between s and t implies the existence of a path between s and t.

$\{s,t\}$-cut: the minimum number of nodes (edges) that cover all $s-t$ paths is equal to the maximum number of node-disjoint (edge-disjoint) $s-t$ paths. However, packing paths of restricted kinds is a difficult problem. One special case is when the paths are required to be of odd-length for which many deep results exist [4, 17, 5]. In this work, we study the problem of covering $s-t$ odd-paths and more generally all odd-paths between a given collection of terminals.

Covering $s-t$ odd-paths in undirected graphs has been explored in the literature from the perspective of polyhedral theory—we refer to Chapter 29 in Schrijver's book [17]. Given an undirected graph $G=(V,E)$ with distinct nodes $s,t \in V$ and non-negative edge-lengths, we may find a shortest length $s-t$ odd-path in polynomial time. Edmonds gave a polynomial-time algorithm for this by reducing the shortest $s-t$ odd-path problem to the minimum-weight perfect matching problem [13, 7, 8]. However, as observed by Schrijver and Seymour [18], his approach of reducing to a matching problem does not extend to address other fundamental problems about $s-t$ odd-paths. One such fundamental problem is the $\{s,t\}$-OddPathEdgeBlocker problem. Towards investigating $\{s,t\}$-OddPathEdgeBlocker, Schrijver and Seymour [18] considered the following polyhedron:

$$\mathcal{P}^{\text{odd-cover}} := \left\{ x \in \mathbb{R}^E_+ : \sum_{e \in P} x_e \geq 1 \ \forall \ s-t \text{ odd-path } P \text{ in } G \right\}.$$

This leads to a natural integer programming formulation of $\{s,t\}$-OddPathEdgeBlocker: $\min \left\{ \sum_{e \in E} x_e : x \in \mathcal{P}^{\text{odd-cover}} \cap \mathbb{Z}^E \right\}$. By Edmonds' algorithm, we have an efficient separation oracle for $\mathcal{P}^{\text{odd-cover}}$ and hence there exists an efficient algorithm to optimize over $\mathcal{P}^{\text{odd-cover}}$ using the Ellipsoid algorithm [10]. It was known that the extreme points of $\mathcal{P}^{\text{odd-cover}}$ are not integral. Cook and Sebö conjectured that all extreme points of $\mathcal{P}^{\text{odd-cover}}$ are half-integral which was later shown by Schrijver and Seymour [18]. Schrijver and Seymour's work also gave a min-max relation for the max fractional packing of $s-t$ odd-paths. However their work does not provide algorithms or address the computational complexity of $\{s,t\}$-OddPathEdgeBlocker. In this work, we show NP-hardness and an inapproximability result for $\{s,t\}$-OddPathEdgeBlocker in undirected graphs.

The main focus of this work is OddMultiwayNodeCut in directed acyclic graphs (DAGs). Before describing the reason for focusing on the subfamily of DAGs among directed graphs, we mention that OddMultiwayNodeCut and OddMultiwayEdgeCut are equivalent in directed graphs by standard reductions. The reason we focus on the subfamily of DAGs and not all directed graphs is the following: consider the $(s \to t)$-OddPathEdgeBlocker problem where the input is a directed graph with nodes s,t and the goal is to find a minimum number of edges to delete so that the resulting graph has no odd-path from s to t. There is a stark contrast in the complexity of $\{s,t\}$-OddPathEdgeBlocker in undirected graphs and $(s \to t)$-OddPathEdgeBlocker in directed graphs: while there exists a polynomial time algorithm to verify if a given undirected graph has an $s-t$ odd-path (e.g., by Edmonds' reduction to a matching problem), it is NP-complete to verify if a given directed graph has an $s \to t$ odd-path (e.g., see Lapaugh-Papadimitriou [13]). Thus verifying feasibility of a solution to OddMultiwayEdgeCut is already NP-complete in directed graphs. However, there exists a polynomial time algorithm to verify if a given directed *acyclic* graph (DAG) has an $s \to t$ odd-path. For this reason, we restrict our focus to DAGs.

Our main contribution is a fixed-parameter algorithm for OddMultiwayNodeCut in DAGs. We complement the fixed-parameter algorithm by showing NP-hardness and tight approximability results for the two terminal variant, namely $(s \to t)$-OddPathNodeBlocker, in DAGs.

In addition to approximation algorithms, fixed-parameter algorithms have served as an alternative approach to address NP-hard problems [9]. A problem is said to be fixed-parameter tractable (FPT), if it can be solved in time $f(k)n^c$, where k is the parameter, f is a computable function, n is the size of the input and c is a universal constant. Fixed-parameter algorithms for cut problems have provided novel insights into the connectivity structure of graphs [6]. The notion of *important separators* and the *shadow-removal technique* have served as the main ingredients in the design of fixed-parameter algorithms for numerous cut problems [6]. Our work also builds upon the *shadow-removal technique* to design fixed-parameter algorithms but differs from known applications substantially owing to the parity constraint.

Related Work. We are not aware of any prior work on this problem in directed graphs. We describe the known results in undirected graphs. A simple reduction from vertex cover[2] shows that $\{s,t\}$-OddPathNodeBlocker in undirected graphs is NP-hard and does not admit a $(2-\epsilon)$-approximation for $\epsilon > 0$ assuming the Unique Games Conjecture [11]. These hardness results also hold for OddMultiwayNodeCut. The most relevant results to this work are that of Lokshtanov and Ramanujan [15, 16]. They showed a parameter-preserving reduction from OddMultiwayEdgeCut to OddMultiwayNodeCut and designed a fixed-parameter algorithm for OddMultiwayNodeCut. However, their algorithmic techniques work only for undirected graphs and do not extend immediately for OddMultiwayNodeCut in directed acyclic graphs.

Lokshtanov and Ramanujan also showed that OddMultiwayEdgeCut is NP-hard in undirected graphs for three terminals. However, their reduction is not an approximation-preserving reduction. Hence the approximability of OddMultiwayEdgeCut in undirected graphs merits careful investigation. In particular, the complexity of OddMultiwayEdgeCut in undirected graphs even for the case of two terminals is open in spite of existing polyhedral work in the literature [18] for this problem.

1.1 Results

Directed acyclic graphs. We recall that OddMultiwayNodeCut and OddMultiwayEdgeCut are equivalent in DAGs by standard reductions. Hence, all of the following results for DAGs hold for both problems. The following is our main result.

▶ **Theorem 1.** OddMultiwayNodeCut *in DAGs can be solved in* $2^{O(k^2)}poly(n)$ *time, where* k *is the size of the optimal solution and* n *is the number of nodes in the input graph.*

We briefly remark on the known techniques to illustrate the challenges in designing the fixed-parameter algorithm for OddMultiwayNodeCut in DAGs. To highlight the challenges, we will focus on the case of 2 terminals, namely $(s \to t)$-OddPathNodeBlocker in DAGs.

Remark 1. It is tempting to design a fixed-parameter algorithm for $(s \to t)$-OddPathNodeBlocker by suitably modifying the definition of *important separators* to account for parity and using the shadow-removal technique for directed graphs [3]. However, the main technical challenge lies in understanding and exploiting the acyclic property of the input directed graph.

[2] Given an instance G of vertex cover, introduce two new nodes s and t that are adjacent to all nodes in G to obtain a graph H. A set $S \subseteq V(G)$ is a feasible vertex cover in G if and only if S is a feasible solution to $\{s,t\}$-OddPathNodeBlocker in H.

Remark 2. The next natural attempt is to rely on the fixed-parameter algorithm for multicut in DAGs by Kratsch et al. [12]. However, their technique crucially relies on reducing the degrees of the source terminals by suitably branching to create a small number of instances. On the one hand, applying their branching rule directly to reduce the degree of s in $(s \to t)$-OddPathNodeBlocker will blow up the number of instances in the branching. On the other hand, it is unclear how to modify their branching rule to account for parity.

Given the difficulties mentioned in the above two remarks, our algorithm builds upon the shadow-removal technique and exploits the acyclic property of the input directed graph to reduce the instance to minimum odd cycle transversal (remove the smallest number of nodes to make an undirected graph bipartite) which in turn, has a fixed-parameter algorithm when parameterized by the number of removed nodes. Our technique is yet another illustration of the broad-applicability of the shadow-removal framework.

We complement our fixed-parameter algorithm in Theorem 1 with tight approximability results for 2 terminals. We refer the reader to Table 1 for a summary of the complexity and approximability results. Unlike the case of undirected graphs where there is still a gap in the approximability of both $\{s,t\}$-OddPathEdgeBlocker and $\{s,t\}$-OddPathNodeBlocker, we present tight approximability results for both $(s \to t)$-OddPathEdgeBlocker and $(s \to t)$-OddPathNodeBlocker.

▶ **Theorem 2.** *We have the following inapproximability and approximability results:*

(i) $(s \to t)$-OddPathNodeBlocker *in DAGs is NP-hard, and has no efficient* $(2-\epsilon)$*-approximation for any* $\epsilon > 0$ *assuming the Unique Games Conjecture.*

(ii) *There exists an efficient* 2*-approximation algorithm for* $(s \to t)$-OddPathNodeBlocker *in DAGs.*

We emphasize that our 2-approximation for $(s \to t)$-OddPathEdgeBlocker mentioned in Theorem 2 is a combinatorial algorithm and not LP-based. We note that Schrijver and Seymour's result [18] that all extreme points of $\mathcal{P}^{\text{odd-cover}}$ are half-integral holds only in undirected graphs and fails in DAGs—see Theorem 3 below. Consequently, we are unable to design a 2-approximation algorithm using the extreme point structure of the natural LP-relaxation of the path-blocking integer program. Instead, our approximation algorithm is combinatorial in nature. The correctness argument of our algorithm also shows that the integrality gap of the LP-relaxation of the path-blocking integer program is at most 2 in DAGs.

▶ **Theorem 3.** *The following odd path cover polyhedron is not necessarily half-integral:* $\mathcal{P}^{\text{odd-cover-dir}} := \left\{x \in \mathbb{R}_+^E : \sum_{e \in P} x_e \geq 1 \ \forall \ s \to t \text{ odd-path } P \text{ in } D\right\}$.

Undirected graphs. We next turn our attention to undirected graphs. As mentioned earlier, $\{s,t\}$-OddPathNodeBlocker is NP-hard and does not admit a $(2-\epsilon)$-approximation assuming the Unique Games Conjecture. We are unaware of a constant factor approximation for $\{s,t\}$-OddPathNodeBlocker. For $\{s,t\}$-OddPathEdgeBlocker, the results of Schrijver and Seymour [18] show that the LP-relaxation of a natural integer programming formulation of $\{s,t\}$-OddPathEdgeBlocker is half-integral and thus leads to an efficient 2-approximation for $\{s,t\}$-OddPathEdgeBlocker. However, the complexity of $\{s,t\}$-OddPathEdgeBlocker was open. We address this gap in complexity by showing the following NP-hardness and inapproximability results.

▶ **Theorem 4.** $\{s,t\}$-OddPathEdgeBlocker *is NP-hard and has no efficient* $(6/5-\epsilon)$*-approximation assuming the Unique Games Conjecture.*

Table 1 Complexity and Approximability. Text in gray refers to known results while text in black refers to the results from this work.

Problem	Undirected graphs	DAGs
$\{s,t\}$-OddPathNodeBlocker	$(2-\epsilon)$-inapprox	[Equiv. to edge-deletion]
$\{s,t\}$-OddPathEdgeBlocker	LP is half-integral [18] 2-approx [18] $(\frac{6}{5}-\epsilon)$-inapprox (Thm 4)	LP is NOT half-integral (Thm 3) 2-approx (Thm 2) $(2-\epsilon)$-inapprox (Thm 2)
OddMultiwayEdgeCut	NP-hard for 3 terminals [15] $(\frac{6}{5}-\epsilon)$-inapprox for 2 terminals (Thm 4)	

Organization. We summarize the preliminaries in Section 1.2. We prove the FPT for DAGs (Theorem 1) in Section 2. We refer the reader to the full version of the paper [1] for all missing proofs.

1.2 Preliminaries

For ease of notation, we will frequently use v instead of $\{v\}$. Let G be a (directed) graph and W be a subset of $V(G)$. A W-path in G is a path with both of its end-nodes in W. We restate the problem of OddMultiwayNodeCut in DAGs to set the notation.

▶ **Problem 5** (Minimum Odd Multiway Cut in DAGs). *Given a directed acyclic graph $G=(V,E)$ with sets T, $V^\infty \subseteq V$ where $T \subseteq V^\infty$, an* odd multiway cut *in G is a set $M \subseteq V(G) \setminus V^\infty$ of nodes that intersects every odd T-path in G. We refer to T as terminals, $V \setminus T$ as non-terminals and V^∞ as protected nodes. In* DagOddMultiwayNodeCut*, the input is specified as (G, V^∞, T, k), where $k \in \mathbb{Z}_+$ and the goal is to verify if there exists an odd multiway cut in G of size at most k.*

For subsets X and Y of $V(G)$ we say that $M \subseteq V(G) \setminus V^\infty$ is an $X \to Y$ *separator* in G when $G \setminus M$ has no path from X to Y. The set of nodes that can be reached from a node set X in G is denoted by $\mathcal{R}_G(X)$. We note that $\mathcal{R}_G(X)$ always includes X. We define the *forward shadow* of a node set M to be $f_G(M) := V(G \setminus M) \setminus \mathcal{R}_{G\setminus M}(T)$, i.e., the set of nodes v such that there is no $T \to v$ path in G disjoint from M. Similarly, the *reverse shadow* of M, denoted $r_G(M)$, is the set of nodes v from which there is no path to T in $G \setminus M$. Equivalently, the reverse shadow is $f_{G^{\text{rev}}}(M)$, where G^{rev} is the graph obtained from G by reversing all the edge orientations. We refer to the union of the forward and the reverse shadow of M in G, as *shadow* of M in G and denote it by $s_G(M)$. An $X \to Y$ separator M' is said to *dominate* another $X \to Y$ separator M, if $|M'| \leq |M|$ and $\mathcal{R}_{G\setminus M}(X) \subsetneq \mathcal{R}_{G\setminus M'}(X)$. A minimal $X \to Y$ separator that is not dominated by any other separator is called an *important $X \to Y$ separator*. A set $M \subseteq V(G)$ is *thin*, if every node $v \in M$ is not in $r_G(M \setminus \{v\})$.

For a directed graph G, we define the *underlying undirected graph* of G, denoted by G^*, as the undirected graph obtained from G by dropping the orientations. In an undirected graph H with protected nodes V^∞, an *odd cycle transversal* is a set $U \subseteq V(H) \setminus V^\infty$ of nodes such that $H \setminus U$ is bipartite. The problem of finding the minimum such set in a given instance of the problem is called the *minimum odd cycle transversal problem* and is denoted by MinOddCycleTransversal. Although this problem is NP-hard, it is fixed-parameter tractable when parameterized by the size of the solution. The current-best fixed-parameter algorithm

for MinOddCycleTransversal runs in time $O(2.32^k \text{poly}(n))$ [14]. The problem addressed in [14] does not allow for protected nodes, but MinOddCycleTransversal with protected nodes, can easily be reduced to MinOddCycleTransversal without protected nodes by iteratively replacing each protected node with $k+1$ nodes and connecting them to the same set of neighbors as the original node. We will use MinOddCycleTransversal(H, V^∞, k) to denote the procedure that implements this fixed-parameter algorithm for the input graph H with protected nodes V^∞ and parameter k.

2 FPT of OddMultiwayNodeCut in DAGs

We will use the shadow-removal technique introduced in [3]. We will reduce the problem to MinOddCycleTransversal problem in an undirected graph, which is a fixed-parameter tractable problem when parameterized by the solution size.

2.1 Easy instances

▶ **Theorem 6.** *Suppose the instance (G, V^∞, T, k) of* DagOddMultiwayNodeCut *has a solution M of size at most k with the following property: every node $v \in s_G(M)$ has total degree at most one in $G \setminus M$. There exists an algorithm that given one such instance (G, V^∞, T, k) as input, finds a solution of size at most k in time $O(2.32^k \text{poly}(n))$, where n is the number of nodes in the input graph G.*

Proof. Let (G, V^∞, T, k) be an instance of DagOddMultiwayNodeCut. We recall that G^* denotes the undirected graph obtained by dropping the orientations of the edges in G. We show the following equivalence: a set $M \subseteq V \setminus V^\infty$ with the property as in the statement is a solution if and only if $G^* \setminus M$ is bipartite with a bipartition (A, B) such that $T \subseteq A$.

Suppose $G^* \setminus M$ is bipartite with a bipartition (A, B) such that $T \subseteq A$. In a bipartite graph, every two end-nodes of any odd path are necessarily in different parts. Hence, there is no odd T-path in $G^* \setminus M$. Thus, there is no odd T-path in $G \setminus M$. Hence, M is a solution for the odd multiway cut instance (G, V^∞, T, k).

Suppose the solution M has the property mentioned in the statement of the theorem. Let $U := V(G \setminus M) \setminus s_G(M)$. Define $A := \{x \in U : \text{there is an even } T \to x \text{ path in } G \setminus M\}$ and $B := \{x \in U : \text{there is an odd } T \to x \text{ path in } G \setminus M\}$. It follows from the definition of the shadow that every node in U has a path P_1 from T in $G \setminus M$. Therefore, every node of U is in $A \cup B$. Also by definition, every node v in U has a path P_2 to T in $G \setminus M$. The parity of every $T \to v$ path has to be the same as the parity of P_2, because the concatenation of a $T \to v$ path and a $v \to T$ path in $G \setminus M$ is a T-path in $G \setminus M$ and therefore must be even. We note that such a concatenation cannot be a cycle since G is acyclic. Thus, no node of U is in both A and B. Hence, (A, B) is a partition of U.

We observe that there cannot be an edge from a node v in A to a node u in A, as otherwise the concatenation of the even $T \to v$ path Q_1 with the edge $v \to u$ is an odd $T \to u$ path in $G \setminus M$ which means $u \in B$. This contradicts our conclusion about A and B being disjoint. By a similar argument, there is no edge between any pair of nodes in B. Thus, the subgraph of G induced by A and B are independent sets respectively. Hence $G^*[A \cup B]$ is a bipartite graph. Furthermore, (A, B) is a bipartition of $G^*[A \cup B]$ with every node of T in A. By assumption, the degree of every node $x \in s_G(M)$ is at most one. Therefore, x has neighbors in at most one of A and B. Thus, we can extend the bipartition (A, B) of $G^*[A \cup B]$ to a bipartition (A', B') of $G^* \setminus M$ as follows: denote $H := G^*[A \cup B]$; repeatedly pick a node

Algorithm 1 SolveEasyInstance

Given: DAG G with terminal set T, a set V^∞ of protected nodes containing T, $k \in \mathbb{Z}_+$, where (G, V^∞, T, k) has the property specified in the statement of Theorem 6

1: $G_1 \leftarrow$ the underlying undirected graph of G.
2: Let G_2 be the graph obtained from G_1 by introducing a new node x and connecting it to every node in T.
3: $N \leftarrow \text{MinOddCycleTransversal}(G_2, V^\infty \cup \{x\}, k)$
4: **return** N

$x \in s_G(M) \setminus V(H)$ with a neighbor in H, include x in a part (A or B) in which x has no neighbor and update A, B and H.

Hence, if the given instance has a solution M of size at most k such that every node $v \in s_G(M)$ has total degree at most one, then such a solution can be found by the fixed-parameter algorithm for MinOddCycleTransversal. To ensure that the terminal nodes will be in the same part, we introduce a new protected node into the graph and connect it to every terminal node. This approach is described in Algorithm 1. All steps in Algorithm 1 can be implemented to run in polynomial time except Step 3. The running time of Step 3 is $O(2.32^k \text{poly}(n))$ [14]. ◀

We will use the name SolveEasyInstance to refer to the algorithm of Theorem 6. Our aim now is to reduce the given arbitrary instance (G, V^∞, T, k) to another instance that has a solution with the property mentioned in Theorem 6 or determine that no solution of size at most k exists. We need the notion of parity-preserving torso operation on DAGs.

2.2 Parity-Preserving Torso

The parity preserving torso operation was introduced by Lokshtanov and Ramanujan [15] for undirected graphs. We extend it in a natural fashion for DAGs.

▶ **Definition 7** (Parity-Preserving Torso)**.** Let G be a DAG and Z be a subset of $V(G)$. We define Parity-Torso(G, V^∞, Z) as (G', V'^∞), where G' is the DAG obtained from $G \setminus Z$ by adding an edge from node u to v, for every pair of nodes $u, v \in V(G) \setminus Z$ such that there is an odd-path from u to v in G all of whose internal nodes are in Z, and including a new node x_{uv} and edges $u \to x_{uv}$ and $x_{uv} \to v$ for every pair of nodes $u, v \in V(G) \setminus Z$ such that there is an even path from u to v in G all of whose internal nodes are in Z. The set V'^∞ is defined to be the union of $V^\infty \setminus Z$ and all the new nodes x_{uv}.

We emphasize that the acyclic nature of the input directed graph allows us to implement the parity-preserving torso operation in polynomial time. Moreover, applying parity-preserving torso on a DAG results in a DAG as well. In what follows, we state the properties of the Parity-Torso operation that are exploited by our algorithm. The parity-preserving torso operation, has the property that it maintains $u \to v$ paths along with their parities between any pair of nodes $u, v \in V(G) \setminus Z$. More precisely:

▶ **Lemma 8.** *Let G be a DAG and $Z, V^\infty \subseteq V(G)$. Define $(G', V'^\infty) :=$ Parity-Torso(G, V^∞, Z). Let u, v be nodes in $V(G) \setminus Z$. There is a $u \to v$ path P in G if and only if there is a $u \to v$ path Q of the same parity in G'. Moreover, the path Q can be chosen so that the nodes of P in $G \setminus Z$ are the same as the nodes of Q in $G \setminus Z$, i.e. $V(P) \cap (V(G) \setminus Z) = V(Q) \cap (V(G) \setminus Z)$.*

Algorithm 2 Minimum odd multiway cut in DAGs

Given: DAG G with terminal set T, a set V^∞ of protected nodes containing T, and $k \in \mathbb{Z}_+$

1: **for** $Z \in$ ShadowContainer(G, V^∞, k) **do**
2: $\quad (G_1, V_1^\infty) \leftarrow$ Parity-Torso(G, V^∞, Z)
3: $\quad N \leftarrow$ SolveEasyInstance(G_1, V_1^∞, T, k)
4: $\quad$ **if** N is a solution in G **then**
5: $\quad\quad$ **return** N
6: **return** No solution

▶ **Corollary 9.** *Let $\mathcal{I} = (G, V^\infty, T, k)$ be an instance of* DagOddMultiwayNodeCut *and let $Z \subseteq V(G) \setminus T$. Let $(G', V'^\infty) :=$ Parity-Torso(G, V^∞, Z) and denote the instance (G', V'^∞, T, k) by $\mathcal{I}'$. The instance $\mathcal{I}$ admits a solution S of size at most k that is disjoint from Z if and only if the instance $\mathcal{I}'$ admits a solution of size at most k.*

Therefore, we are interested in finding a set Z of nodes that is disjoint from some solution of size at most k, and moreover, the instance (Parity-Torso$(G, V^\infty, Z), T, k)$ satisfies the property mentioned in Theorem 6. The following lemma summarizes our key observation: it shows that it is sufficient to find a set Z that contains the shadow of a solution.

▶ **Lemma 10.** *Let G be a DAG and $M, Z, V^\infty \subseteq V(G)$. Suppose M intersects every odd T-path in G and $s_G(M) \subseteq Z \subseteq V(G) \setminus M$. Define $(G', V'^\infty) :=$ Parity-Torso(G, V^∞, Z). Then every node in $s_{G'}(M)$ has total degree at most one in $G' \setminus M$.*

2.3 Difficult instances

Corollary 9 and Lemma 10 show that if we find a set Z such that for some solution M, the set Z is disjoint from M and contains the shadow of M in G, then considering Parity-Torso(G, V^∞, Z) will give a new instance that satisfies the conditions of Theorem 6. Our goal now is to obtain such a set Z. We will show the following lemma. We emphasize that the lemma holds for arbitrary digraphs.

▶ **Lemma 11.** *There is an algorithm* ShadowContainer *that given an instance (G, V^∞, T, k) of* DagOddMultiwayNodeCut*, where G is a digraph, returns a family $\mathcal{Z}$ of subsets of $V(G)$ with $|\mathcal{Z}| = 2^{O(k^2)}$, with the property that if the problem has a solution of size at most k, then for some solution M of size at most k, there exists a set $Z \in \mathcal{Z}$ that is disjoint from M and contains $s_G(M)$. Moreover, the algorithm can be implemented to run in time $2^{O(k^2)}\,\mathrm{poly}(|V(G)|)$.*

We defer the proof of Lemma 11 to first see its implications.

▶ **Theorem 12.** *There exists an algorithm that given an instance (G, V^∞, T, k) of* DagOddMultiwayNodeCut*, runs in $2^{O(k^2)}\,\mathrm{poly}(|V(G)|)$ time and either finds a solution of size at most k or determines that no such solution exists.*

Proof. We use Algorithm 2. Let (G, V^∞, T, k) be an instance of DagOddMultiwayNodeCut, where G is a DAG. Suppose there exists a solution of size at most k. By Lemma 11, the procedure ShadowContainer(G, V^∞, k) in Line 1 returns a family $\mathcal{Z}$ of subsets of $V(G)$ with $|\mathcal{Z}| = 2^{O(k^2)}$ containing a set Z such that there is a solution M of size at most k that is disjoint from Z and Z contains $s_G(M)$. Let (G_1, V_1^∞) be the result of applying Parity-Torso operation to the set Z in G (i.e., the result of Step 2 in Algorithm 2). By Lemma 10, every node in $s_{G_1}(M)$ has total degree at most one in $G_1 \setminus M$. Therefore, by Theorem 6, the set

N returned in Line 3 is a solution to the instance (G_1, V_1^∞, T, k). By Corollary 9, the set N is also a solution to the original instance of the problem.

If there is no solution of size at most k, the algorithm will not find any. Therefore, the algorithm is correct. The runtime of the algorithm is dominated by Line 2 which can be implemented to run in $2^{O(k^2)}\text{poly}(|V(G)|)$ time by Lemma 11. ◀

In order to prove Lemma 11, we will use the following result.

▶ **Theorem 13** (Chitnis *et al.* [2]). *There is an algorithm that given a digraph G, a subset of protected nodes $V^\infty \subseteq V(G)$, terminal nodes $T \subseteq V^\infty$ and an integer k, returns a family $\mathcal{Z}$ of subsets of $V(G) \setminus V^\infty$ with $|\mathcal{Z}| = 2^{O(k^2)}$ such that for every $S, Y \subseteq V(G)$ satisfying*
(i) *S is a thin set with $|S| \leq k$, and*
(ii) *for every $v \in Y$, there exists an important $v \to T$ separator contained in S,*
there exists $Z \in \mathcal{Z}$ with $Y \subseteq Z \subseteq V(G) \setminus S$. Moreover, the algorithm can be implemented to run in time $2^{O(k^2)}\,\text{poly}(|V(G)|)$.

To invoke Theorem 13, we need to guarantee that there exists a solution S of size at most k such that S is thin and its reverse shadow Y in G has the property that for every $v \in Y$ there is an important $v \to T$ separator contained in S. Towards obtaining such a solution, we prove the following.

▶ **Lemma 14.** *Let (G, V^∞, T, k) be an instance of* DagOddMultiwayNodeCut*, where G is a DAG. Let M be a solution for this instance. If there exists $v \in r_G(M)$ such that M does not contain an important $v \to T$ separator, then there exists another solution M' of size at most $|M|$, such that $r_G(M) \cup f_G(M) \cup M \subseteq r_G(M') \cup f_G(M') \cup M'$, and $r_G(M) \subsetneq r_G(M')$.*

Proof. Let M_0 be the set of nodes $u \in M$ for which there is a $v \to u$ path in G that is internally disjoint from M. Since $v \in r_G(M)$, every $v \to T$ path intersects M. For a $v \to T$ path P, the first node $u \in P \cap M$ is in M_0. Hence, every $v \to T$ path intersects M_0. Therefore, the set M_0 is a $v \to T$ separator in G. Therefore, it contains a minimal separator M_1. Since we assumed that there is no important $v \to T$ separator contained in M, the set M_1 is not an important $v \to T$ separator. Suppose M_1 is dominated by another $v \to T$ separator and let M_2 be an important $v \to T$ separator that dominates M_1. Define M' as $(M \setminus M_1) \cup M_2$. We recall that a separator is by definition, disjoint from the protected node set. Therefore, $M' \cap V^\infty = \emptyset$. We will show that M' contradicts the choice of M. We need the following claims.

▶ **Claim 15.** *$M \setminus M' \subseteq r_G(M')$.*

Proof. We observe that $M \setminus M' = M_1 \setminus M_2$. Let u be an arbitrary node in $M_1 \setminus M_2$. Since $u \in M_1$ and M_1 is a minimal $v \to T$ separator, there is a $v \to u$ path P_1 that is internally disjoint from M_1. Since M_2 dominates M_1, therefore, $R_{G \setminus M_1}(v) \subseteq R_{G \setminus M_2}(v)$. Thus, $V(P_1) \subseteq R_{G \setminus M_2}(v)$. Hence, P_1 is disjoint from M_2. Suppose P_2 is an arbitrary $u \to T$ path in G. Concatenation of P_1 and P_2 is a $v \to T$ path in G and therefore, has to intersect M_2. Since P_1 is disjoint from M_2, the path P_2 has to intersect M_2. Hence, every $u \to T$ path in G intersects M_2 and in particular, intersects M'. Equivalently, $u \in r_G(M')$. ◀

We next show that M' is a feasible solution for the problem and is no larger than M.

▶ **Claim 16.** *The set M' intersects every odd T-path in G and $|M'| \leq |M|$.*

Proof. By assumption, every odd T-path P intersects M. If P intersects $M \cap M'$, then it also intersects M'. If P intersects $M \setminus M'$, then by Claim 15 it also intersects M'. Thus, every odd T-path in G intersects M'. Furthermore, by definition of M', we have

$$|M'| = |M| + (|M_2 \setminus M| - |M_1 \setminus M_2|) \leq |M| + (|M_2| - |M_1|) \leq |M|\,. \quad \blacktriangleleft$$

▶ **Claim 17.** *$r_G(M) \subseteq r_G(M')$.*

Proof. Let u be an arbitrary node in $r_G(M)$. The set M is a $u \to T$ separator. Therefore, every $u \to T$ path intersects M. We need to show that every $u \to T$ path also intersects M'. Let P be a $u \to T$ path. If P intersects $M \cap M'$, then it also intersects M'. If P does not intersect $M \cap M'$, then it has to intersect $M \setminus M'$. By Claim 15, every $M \setminus M' \to T$ path intersects M'. Therefore, $u \in r_G(M')$. ◀

▶ **Claim 18.** *$r_G(M) \cup f_G(M) \cup M \subseteq r_G(M') \cup f_G(M') \cup M'$.*

Proof. By Claim 15, we have $M \setminus M' \subseteq r_G(M')$ and by Claim 17, we have $r_G(M) \subseteq r_G(M')$. Thus, it remains to prove that $f_G(M) \subseteq r_G(M') \cup f_G(M') \cup M'$. Let u be an arbitrary node in $f_G(M) \setminus (r_G(M') \cup f_G(M') \cup M')$. Since $u \notin f_G(M')$, there is a $T \to u$ path P_1 in G that is disjoint from M'. But $u \in f_G(M)$. Thus P_1 has to intersect M, particularly it has to intersect $M \setminus M'$. Let P_2 be a subpath of P_1 from $M \setminus M'$ to u. Since $u \notin r_G(M')$, there is a $u \to T$ path P_3 in G that is disjoint from M'. The concatenation of P_2 and P_3 is a path from $M \setminus M'$ to T that is disjoint from M'. But by Claim 15, every $M \setminus M' \to T$ path in G must intersect M'. This contradiction shows that $f_G(M) \subseteq (r_G(M') \cup f_G(M') \cup M')$. ◀

▶ **Claim 19.** *$r_G(M) \subsetneq r_G(M')$.*

Proof. By Claim 17, $r_G(M) \subseteq r_G(M')$. We need to prove $r_G(M) \neq r_G(M')$. We recall that $M \setminus M' = M_1 \setminus M_2$. Since M_2 is an important $v \to T$ separator, it follows that the $v \to T$ separator M_1 is not contained in M_2. Therefore $M \setminus M'$ is non-empty. Furthermore, by definition of reverse shadow, $M \setminus M'$ is not contained in $r_G(M)$, but by Claim 15, it is contained in $r_G(M')$. ◀

By Claim 16, M' is a solution and is no larger than M. Therefore, the set M' has the properties claimed in Lemma 14. ◀

We recall that a set $M \subseteq V(G)$ is *thin*, if every node $v \in M$ is not in $r_G(M \setminus \{v\})$. The next result follows from Lemma 14.

▶ **Corollary 20.** *Let (G, V^∞, T, k) be an instance of* DagOddMultiwayNodeCut*, where G is a DAG. Let M^* be an optimal solution that maximizes the size of $|r_G(S) \cup f_G(S) \cup S|$ among all optimal solutions S. If more than one optimal solution maximizes this quantity, choose the one with largest $|r_G(S)|$. The set M^* is thin and for every node $v \in r_G(M^*)$ there is an important $v \to T$ separator in M^*.*

We will use Corollary 20 to prove Lemma 11.

Proof of Lemma 11. Let us use ReverseShadowContainer(G, V^∞, k) to denote the algorithm from Theorem 13. We will show that Algorithm 3 generates the desired set.

In Algorithm 3 we use procedure ReverseShadowContainer introduced in Theorem 13. By Theorem 13, the cardinality of $\mathcal{Z}$ returned by the algorithm is $2^{O(k^2)}$. The runtime of the algorithm follows from the runtime of the procedure ReverseShadowContainer in Theorem 13. To prove the correctness of this algorithm, we argue that at least one of the sets in the returned family $\mathcal{Z}$ has the desired properties.

Algorithm 3 ShadowContainer

Given: DAG G with terminal set T, a set V^∞ of protected nodes containing T, and $k \in \mathbb{Z}_+$

1: Let G^{rev} denote the graph obtained from G by reversing the orientation of all edges
2: $\mathcal{Z}_1 \leftarrow \text{ReverseShadowContainer}(G, V^\infty, k)$
3: **for** $Z_1 \in \mathcal{Z}_1$ **do**
4: $\quad \mathcal{Z}_2 \leftarrow \text{ReverseShadowContainer}(G^{\text{rev}}, V^\infty \cup Z_1, k)$
5: $\quad$ **for** $Z_2 \in \mathcal{Z}_2$ **do**
6: $\quad\quad \mathcal{Z} \leftarrow \mathcal{Z} \cup \{Z_1 \cup Z_2\}$
7: **return** $\mathcal{Z}$

Suppose there exists a solution of size at most k and let M^* be an optimal solution that maximizes the size of $|r_G(S) \cup f_G(S) \cup S|$ among all optimal solutions S. If more than one solution maximizes this quantity, choose the one with largest $|r_G(S)|$. By Corollary 20, the solution M^* is thin and has the property that every node v in the reverse shadow of M^* has an important $v \to T$ separator contained in M^*. By Theorem 13, the procedure $\text{ReverseShadowContainer}(G, V^\infty, k)$ in Line 2 will return a family $\mathcal{Z}_1$ of sets containing a set Z_1 that is disjoint from M^* and contains its reverse shadow. Let us fix such a Z_1.

We note that a solution for the DagOddMultiwayNodeCut instance $(G^{\text{rev}}, V^\infty \cup Z_1, T, k)$ is also a solution for the instance (G, V^∞, T, k). Conversely, a solution for the instance (G, V^∞, T, k) that is disjoint from Z_1 is also a solution for the instance $(G^{\text{rev}}, V^\infty \cup Z_1, T, k)$. Therefore, the set M^* is also an optimal solution to the instance $(G^{\text{rev}}, V^\infty \cup Z_1, T, k)$. We observe that $f_G(S) = r_{G^{\text{rev}}}(S)$ and $r_G(S) = f_{G^{\text{rev}}}(S)$ for all $S \subseteq V(G) \setminus V^\infty$. Therefore, M^* maximizes the size of $r_{G^{\text{rev}}}(S) \cup f_{G^{\text{rev}}}(S) \cup S$ among all optimal solutions S to $(G^{\text{rev}}, V^\infty \cup Z_1, T, k)$. We have the following claim.

▶ **Claim 21.** *If for an optimal solution M' for the instance $(G^{rev}, V^\infty \cup Z_1, T, k)$ of* DagOddMultiwayNodeCut*, we have $r_{G^{rev}}(M^*) \cup f_{G^{rev}}(M^*) \cup M^* \subseteq r_{G^{rev}}(M') \cup f_{G^{rev}}(M') \cup M'$ and $r_{G^{rev}}(M^*) \subseteq r_{G^{rev}}(M')$, then $M' = M^*$.*

Proof. As M^* maximizes $|r_{G^{\text{rev}}}(S) \cup f_{G^{\text{rev}}}(S) \cup S|$ among all the optimal solutions for the instance (G, V^∞, T, k) and as $r_{G^{\text{rev}}}(M^*) \cup f_{G^{\text{rev}}}(M^*) \cup M^* \subseteq r_{G^{\text{rev}}}(M') \cup f_{G^{\text{rev}}}(M') \cup M'$, hence, the two sets $r_{G^{\text{rev}}}(M^*) \cup f_{G^{\text{rev}}}(M^*) \cup M^*$ and $r_{G^{\text{rev}}}(M') \cup f_{G^{\text{rev}}}(M') \cup M'$ must be equal. Therefore, the set $M' \setminus M^*$ is contained inside $r_{G^{\text{rev}}}(M^*) \cup f_{G^{\text{rev}}}(M^*) \cup M^*$. Since nodes in $f_{G^{\text{rev}}}(M^*)$ are protected in G^{rev} by construction, the solution M' cannot contain any node from $f_{G^{\text{rev}}}(M^*)$. Since $r_{G^{\text{rev}}}(M^*) \subseteq r_{G^{\text{rev}}}(M')$ and by definition of reverse shadow, M' is disjoint from $r_{G^{\text{rev}}}(M^*)$. Thus, the set $M' \setminus M^*$ is disjoint from M^* and $r_{G^{\text{rev}}}(M^*)$ and $f_{G^{\text{rev}}}(M^*)$, while being contained in $r_{G^{\text{rev}}}(M^*) \cup f_{G^{\text{rev}}}(M^*) \cup M^*$. Hence, $M' \setminus M^* = \emptyset$ or equivalently $M' \subseteq M^*$. Therefore, $M' = M^*$, because $|M'| = |M^*|$. ◀

Suppose there is a node $v \in r_{G^{\text{rev}}}(M^*)$ such that no important $v \to T$ separator in G^{rev} is contained in M^*. Then by Lemma 14, there is another optimal solution M' such that $r_{G^{\text{rev}}}(M^*) \cup f_{G^{\text{rev}}}(M^*) \cup M^* \subseteq r_{G^{\text{rev}}}(M') \cup f_{G^{\text{rev}}}(M') \cup M'$ and $r_{G^{\text{rev}}}(M^*) \subsetneq r_{G^{\text{rev}}}(M')$. By Claim 21, the set $M' = M^*$, which contradicts $r_{G^{\text{rev}}}(M^*) \subsetneq r_{G^{\text{rev}}}(M')$. This contradiction shows that for every node $v \in r_{G^{\text{rev}}}(M^*)$, there is an important $v \to T$ separator in G^{rev} that is contained in M^*. Thus, by Theorem 13, the procedure $\text{ReverseShadowContainer}(G^{\text{rev}}, V^\infty \cup Z_1, k)$ from Line 4 will return a family $\mathcal{Z}_2$ of sets containing a set Z_2 that is disjoint from M^* and contains $r_{G^{\text{rev}}}(M^*) = f_G(M^*)$. Hence $Z_1 \cup Z_2$ is disjoint from M^* and contains $s_G(M^*)$. ◀

References

1. Karthekeyan Chandrasekaran and Sahand Mozaffari. Odd Multiway Cut in Directed Acyclic Graphs. https://arxiv.org/abs/????.????, 2017.
2. Rajesh Hemant Chitnis, Marek Cygan, Mohammad Taghi Hajiaghayi, and Dániel Marx. Directed subset feedback vertex set is fixed-parameter tractable. *ACM Trans. Algorithms*, 11(4):28:1–28:28, 2015. doi:10.1145/2700209.
3. Rajesh Hemant Chitnis, MohammadTaghi Hajiaghayi, and Dániel Marx. Fixed-parameter tractability of directed multiway cut parameterized by the size of the cutset. *SIAM J. Comput.*, 42(4):1674–1696, 2013. doi:10.1137/12086217X.
4. Maria Chudnovsky, Jim Geelen, Bert Gerards, Luis A. Goddyn, Michael Lohman, and Paul D. Seymour. Packing non-zero a-paths in group-labelled graphs. *Combinatorica*, 26(5):521–532, 2006. doi:10.1007/s00493-006-0030-1.
5. Ross Churchley, Bojan Mohar, and Hehui Wu. Weak duality for packing edge-disjoint odd (u, v)-trails. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '16, pages 2086–2094, 2016.
6. Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.
7. Jack Edmonds. Maximum matching and a polyhedron with 0, 1-vertices. *Journal of Research of the National Bureau of Standards*, 69(1-2):125–130, 1965.
8. Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17(3):449–467, 1965.
9. Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006. doi:10.1007/3-540-29953-X.
10. Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer, 1988. doi:10.1007/978-3-642-78240-4.
11. Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2-epsilon. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008. doi:10.1016/j.jcss.2007.06.019.
12. Stefan Kratsch, Marcin Pilipczuk, Michal Pilipczuk, and Magnus Wahlström. Fixed-parameter tractability of multicut in directed acyclic graphs. *SIAM J. Discrete Math.*, 29(1):122–144, 2015. doi:10.1137/120904202.
13. Andrea S. LaPaugh and Christos H. Papadimitriou. The even-path problem for graphs and digraphs. *Networks*, 14(4):507–513, 1984. doi:10.1002/net.3230140403.
14. Daniel Lokshtanov, N. S. Narayanaswamy, Venkatesh Raman, M. S. Ramanujan, and Saket Saurabh. Faster parameterized algorithms using linear programming. *ACM Trans. Algorithms*, 11(2):15:1–15:31, 2014. doi:10.1145/2566616.
15. Daniel Lokshtanov and M. S. Ramanujan. Parameterized tractability of multiway cut with parity constraints. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, volume 7391 of *Lecture Notes in Computer Science*, pages 750–761. Springer, 2012. doi:10.1007/978-3-642-31594-7_63.
16. Sridharan Ramanujan. *Parameterized Graph Separation Problems: New Techniques and Algorithms*. PhD thesis, The Institute of Mathematical Sciences, Chennai, 2013.
17. Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Algorithms and Combinatorics. Springer, 2003.
18. Alexander Schrijver and Paul D. Seymour. Packing odd paths. *J. Comb. Theory, Ser. B*, 62(2):280–288, 1994. doi:10.1006/jctb.1994.1070.