ON WEIGHTED GRAPH SEPARATION PROBLEMS AND FLOW-AUGMENTATION[*]

EUN JUNG KIM[†], TOMÁŠ MASAŘÍK[‡], MARCIN PILIPCZUK[‡], ROOHANI SHARMA[§],
AND MAGNUS WAHLSTRÖM[¶]

Abstract. One of the first applications of the recently introduced technique of *flow-augmentation* [Kim et al., STOC 2022] is a fixed-parameter algorithm for the weighted version of Directed Feedback Vertex Set, a landmark problem in parameterized complexity. In this article, we explore the applicability of flow-augmentation to other weighted graph separation problems parameterized by the size of the cutset. We show the following.

- In weighted undirected graphs Multicut is FPT, both in the edge- and vertex-deletion version.
- The weighted version of Group Feedback Vertex Set is FPT, even with oracle access to group operations.
- The weighted version of Directed Subset Feedback Vertex Set is FPT.

Our study reveals Directed Symmetric Multicut as the next important graph separation problem whose parameterized complexity remains unknown, even in the unweighted setting.

Key words. Weighted Multicut, Weighted Group Feedback Vertex Set, Weighted Directed Subset Feedback Vertex Set, Parameterized Complexity, Directed Flow Augmentation

MSC codes. 05C85, 68Q25, 68W20

1. Introduction. The family of graph separation problems includes a wide range of combinatorial problems where the goal is to remove a small part of the input graph to obtain some separation properties. For example, in the Multicut problem, the input graph G is equipped with a set of terminal pairs $\mathcal{T} \subseteq V(G) \times V(G)$ and the separation objective is to destroy, for every $(s, t) \in \mathcal{T}$, all paths from s to t. In the Subset Feedback Edge/Vertex Set problems, the input graph G is equipped with a set $R \subseteq E(G)$ of *red edges* and the goal is to destroy all cycles that contain at least one red edge by deleting edges (resp. vertices).[1] We remark that in directed graphs, one can equivalently require to destroy all closed walks containing at least one red edge. Even though the later does not explicitly look like a separation problem, at the core of it lies a special variant of the Multicut problem [3].

Both these problems (and many others) can be considered in multiple variants: graphs can be undirected or directed, we are allowed to delete edges or vertices, weights can be present, etc. In this paper, we consider both edge- and vertex-deletion variants and both cardinality and weight budget for the solution. That is, the input graph G is equipped with a weight function ω that assigns positive integral weights to deletable objects (i.e., edges or vertices), and we are given two integers: k, the maximum

---

[†]Université Paris-Dauphine, PSL Research University, CNRS, UMR 7243, LAMSADE, 75016, Paris, France (eun-jung.kim@dauphine.fr).

[‡]University of Warsaw, Warsaw, Poland (masarik@mimuw.edu.pl, malcin@mimuw.edu.pl).

[§]Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany (rsharma@mpi-inf.mpg.de).

[¶]Royal Holloway, University of London, UK (magnus.wahlstrom@rhul.ac.uk).

[1]In the literature, sometimes one considers red vertices instead of red edges. Since there are simple reductions between the variants (cf. [11]), we prefer to work with red edges.

number of deleted objects, and W, the maximum total weight of the deleted objects in the sought solution.

The study of parameterized complexity of graph separation problems has been a vivid line for the past two decades, and resulted in many tractability results and a wide range of algorithmic techniques: important separators and shadow removal [3, 4, 7, 11, 22, 26, 30, 32], branching guided by an LP relaxation [10, 14, 16], matroid-based techniques [23, 24], treewidth reduction [27], randomized contractions [5, 8], and, most recent, flow-augmentation [19, 20]. However, the vast majority of these works considered only the unweighted versions of the problems for a very simple reason: we did not know how to handle their weighted counterparts. In particular, one of the most fundamental notions — important separators, introduced by Marx in 2004 [26] — relies on a greedy argument that breaks down in the presence of weights. The quest to understand the weighted counterparts of studied graph separation problems, with a specific goal to resolve the parameterized complexity of the weighted version of DIRECTED FEEDBACK VERTEX SET — the landmark problem in parameterized complexity [4] — was raised by Saurabh in 2017 [34] (see also [25]).

This question has been resolved recently by Kim et al. [20] with a new algorithmic technique called *flow-augmentation*. Apart from proving fixed-parameter tractability of the weighted version of DIRECTED FEEDBACK VERTEX SET, they also showed fixed-parameter tractability of the weighted CHAIN SAT, resolving another long-standing open problem [6]. Both the aforementioned results are, in fact, the same relatively simple algorithm for a more general problem WEIGHTED BUNDLED CUT WITH ORDER.

Very recently, Galby et al. [13] used the flow-augmentation technique to design an FPT algorithm for weighted MULTICUT on trees. Our results thus extend theirs by generalizing the input graphs from trees to arbitrary undirected graphs.

Our results. The goal of this paper is to explore for which other graph separation problems the flow-augmentation technique helps in getting fixed-parameter algorithms for weighted graph separation problems. (All algorithms below are randomized; all randomization comes from the flow-augmentation technique.)

We start with the MULTICUT problem in undirected graphs, whose parameterized complexity — in the unweighted setting — had been a long-standing open problem until being settled in the affirmative by two independent groups of researchers in 2011 [2, 30].

THEOREM 1.1. WEIGHTED MULTICUT, *parameterized by the cardinality of the cutset, is randomized FPT, both in the edge- and vertex-deletion variants.*

Theorem 1.1 follows from a combination of two arguments. First, we revisit the reduction of Marx and Razgon from MULTICUT to a *bipedal variant*, presented in the conference version of their paper [29] and show how to replace one greedy step based on important separators with a different, weights-resilient step. Then, a folklore reduction to a graph separation problem called COUPLED MIN-CUT, spelled out in [18], does the job: the fixed-parameter tractability of a wide generalization of COUPLED MIN-CUT, including its weighted variant, is one of the main applications of flow-augmentation [19, 20, 21].

MULTIWAY CUT is a special case of MULTICUT where the input graph G is equipped with a set $T \subseteq V(G)$ of terminals and $\mathcal{T} = \{(s,t) \mid s,t \in T, s \neq t\}$, that is, we are to destroy all paths between distinct terminals. Thus, Theorem 1.1 implies the following.

COROLLARY 1.2. WEIGHTED MULTIWAY CUT, *parameterized by the cardinality*

of the cutset, is randomized FPT, both in the edge- and vertex-deletion variants.

We remark that in directed graphs the parameterized complexity of MULTICUT is fully understood: without weights, it is W[1]-hard for 4 terminal pairs [31] and FPT for 3 terminal pairs [15], but with weights, it is already W[1]-hard for 2 terminal pairs [15], while for 1 terminal pair it is known under the name of BI-OBJECTIVE st-CUT and its fixed-parameter tractability follows easily via flow-augmentation [20]. Furthermore, while MULTIWAY CUT on directed graphs is FPT in the unweighted setting [7], on directed graphs MULTICUT with 2 terminal pairs reduces to MULTIWAY CUT with two terminals [7]. Hence, MULTIWAY CUT with weights is W[1]-hard and without weights is FPT on directed graphs.

Then we turn our attention to GROUP FEEDBACK EDGE/VERTEX SET. Here, the input graph G is equipped with a group Γ, not necessarily Abelian, and an assignment ψ, called the *group labels*, that assigns to every $e \in E(G)$ and $v \in e$ an element $\psi(e, v) \in \Gamma$ such that for $e = uv$ we have $\psi(e, u) + \psi(e, v) = 0$.[2] With a walk $C = (v_1, e_1, v_2, e_2, \ldots, v_\ell, e_\ell, v_{\ell+1})$ we associate a sum $\psi(C) = \sum_{i=1}^{\ell} \psi(e_i, v_i)$. A walk C is a *null walk* in (G, ψ) if $\psi(C) = 0$ and *non-null* otherwise. This is well-defined even for non-Abelian groups, i.e., a cycle being null or non-null does not depend on the direction of traversal or the choice of starting vertex v_1 [9]. The separation goal is to destroy all non-null cycles (equivalently, all non-null closed walks) by edge or vertex deletions.

THEOREM 1.3. WEIGHTED GROUP FEEDBACK EDGE SET *and* WEIGHTED GROUP FEEDBACK VERTEX SET, *parameterized by the cardinality of the cutset, are randomized FPT.*

Since WEIGHTED SUBSET FEEDBACK EDGE/VERTEX SET can be modeled as WEIGTED GROUP FEEDBACK EDGE/VERTEX SET with group $\Gamma = \mathbb{Z}_2^{|R|}$ (cf. [9]), we immediately have the following corollary.

COROLLARY 1.4. WEIGHTED SUBSET FEEDBACK EDGE SET *and* WEIGHTED SUBSET FEEDBACK VERTEX SET, *parameterized by the cardinality of the cutset, are randomized FPT.*

The currently fastest FPT algorithm for (unweighted) GROUP FEEDBACK VERTEX SET is due to Iwata, Wahlström, and Yoshida [16] and uses sophisticated branching guided by an LP relaxation. To prove Theorem 1.3, we revisit an older (and less efficient) FPT algorithm due to Cygan, Pilipczuk, and Pilipczuk [9] that performs some branching steps to reduce the problem to multiple instances of MULTIWAY CUT. We observe that the branching easily adapts to the weighted setting, and the algorithm for WEIGHTED MULTIWAY CUT is provided by Corollary 1.2.

We now move to directed graphs. As already mentioned, the parameterized complexity of both weighted and unweighted DIRECTED MULTICUT (and DIRECTED MULTIWAY CUT) is already fully understood [15, 31]. Our main result here is fixed-parameter tractability of WEIGHTED DIRECTED SUBSET FEEDBACK EDGE/VERTEX SET.

THEOREM 1.5. WEIGHTED DIRECTED SUBSET FEEDBACK EDGE SET *and* WEIGHTED DIRECTED SUBSET FEEDBACK VERTEX SET, *parameterized by the car-*

---

[2]Thorough this paper, we use $+$ for the group operation, 0 for the neutral element in the group, and $-$ for the group inverse, to conform with the standard terminology of *null cycles* in GFVS. We note that this is in tension with the convention that a group operation written as $+$ tends to imply an Abelian group.

dinality of the cutset, are randomized FPT.

Theorem 1.5 follows from a surprisingly delicate reduction to WEIGHTED BUNDLED CUT WITH ORDER, known to be FPT via flow-augmentation [20]. SKEW MULTICUT is a special case of DIRECTED MULTICUT where the set $\mathcal{T}$ has the form

$$\{(s_i, t_j) \mid 1 \le i \le j \le \ell\}$$

for some terminals $s_1, \ldots, s_\ell, t_1, \ldots, t_\ell \in V(G)$. SKEW MULTICUT naturally arises in the context of DIRECTED FEEDBACK VERTEX SET if one applies the iterative compression technique. In the unweighted setting, SKEW MULTICUT is long known to be FPT parameterized by the size of the cutset [3]. With weights, [20] showed that SKEW MULTICUT is FPT when parameterized by $k+\ell$. We observe a simple reduction to WEIGHTED DIRECTED SUBSET FEEDBACK VERTEX SET, yielding fixed-parameter tractability when parameterizing by k only.

COROLLARY 1.6. WEIGHTED SKEW MULTICUT, *parameterized by the cardinality of the cutset, is randomized FPT, both in the edge- and vertex-deletion variants.*

Proof. Let $(G, (s_i, t_i)_{i=1}^\ell, \omega, k, W)$ be a WEIGHTED SKEW MULTICUT instance (in the edge- or vertex-deletion setting) where ω is the weight function (on the edges or vertices, respectively) and W is the weight budget of the solution. Construct a graph G' and a set of red edges R as follows: start with $G' = G$, $R = \emptyset$ and, for every $1 \le i \le j \le \ell$, introduce a red edge (t_j, s_i) and add it to G' (in the edge-deletion setting, the new edge has weight $W+1$, that is, it is effectively undeletable). It is easy to see that the resulting WEIGHTED DIRECTED SUBSET FEEDBACK EDGE/VERTEX SET instance (G', R, ω, k, W) is equivalent to the input WEIGHTED SKEW MULTICUT instance: any closed walk in G' involving a red edge contains a subpath from s_i to t_j for some $1 \le i \le j \le \ell$ without any red edge, and any path in G from s_i to t_j for $1 \le i \le j \le \ell$ closes up to a cycle with a red edge (t_j, s_i) in G'. Therefore the corollary follows from Theorem 1.5.   □

The running time bounds of all our algorithms are of the form $2^{\mathrm{poly}(k)}\mathrm{poly}(|V(G)|)$, where both polynomial dependencies have an unspecified large degree coming from the use of involved flow-augmentation-based algorithms of [20, 21].

Summary of our results. In this paper we show that the following problems are fixed-parameter tractable with respect to the solution size k: the vertex- and edge-deletion variants of WEIGHTED MULTIWAY CUT, WEIGHTED MULTICUT and WEIGHTED SKEW MULTICUT, WEIGHTED GROUP FEEDBACK VERTEX/EDGE SET, WEIGHTED SUBSET FEEDBACK VERTEX/EDGE SET and WEIGHTED DIRECTED SUBSET FEEDBACK VERTEX/ARC SET.

Organization. We introduce the necessary tools, in particular the used corollaries of the flow-augmentation technique, in Section 2. Theorem 1.1 is proven in Section 3, Theorem 1.3 is proven in Section 4, and Theorem 1.5 is proven in Section 5. Section 6 concludes the paper and identifies DIRECTED SYMMETRIC MULTICUT as a next problem whose parameterized complexity remains open.

2. Preliminaries.

2.1. Edge- and vertex-deletion variants. For directed graphs, we use edges and arcs interchangeably. In directed graphs, there is a simple reduction from the vertex-deletion setting to the edge-deletion one: replace every vertex v with two vertices v^+ and v^- and an edge (v^-, v^+); every previous arc (u, v) becomes an arc (u^+, v^-). Now, the deletion of the vertex v corresponds to the deletion of the arc

(v^-, v^+). Hence, in Section 5 we will consider only the edge-deletion variant, that is, DIRECTED SUBSET FEEDBACK EDGE SET. No such simple reduction is available in undirected graphs and, in fact, in some cases, the vertex-deletion variant turns out to be significantly more difficult (cf. the k-WAY CUT problem [5, 17, 26]).

In the presence of weights, there is a simple reduction from the edge-deletion variant to the vertex-deletion variant: subdivide every edge with a new vertex that inherits the weight of the edge it is placed on, and set the weight of the original vertices to $+\infty$, making them undeletable. (For clarity, we allow the weight function ω to attain the value $+\infty$, which is equivalent to any weight larger than W and models an undeletable edge or vertex.) Thus, both in Section 3 and in Section 4 we consider the vertex-deletion variants.

2.2. Iterative Compression. All problems considered in this paper are monotone in the sense that the deletion of an edge or a vertex from the input graph cannot turn a YES-instance into a NO-instance. This allows to use of the standard technique of *iterative compression* [33]: We enumerate $V(G) = \{v_1, v_2, \ldots, v_n\}$ for $n = |V(G)|$, denote $G_i = G[\{v_1, \ldots, v_i\}]$ for $0 \le i \le n$ and iteratively solve the problem on graphs $G_0, G_1, \ldots, G_n = G$. If the instance for G_i turns out to be a NO-instance, we deduce that the input instance is a NO-instance, too. Otherwise, the computed solution for G_i allows us to infer a set $X' \subseteq V(G_i)$ of size at most k such that in $G_i - X'$ already has the desired separation (i.e., induces a YES-instance with parameter $k = 0$). We set $X = X' \cup \{v_{i+1}\}$ and observe that $G_{i+1} - X = G_i - X'$ and $|X| \le k + 1$.

Furthermore, in all considered problems, using self-reducibility it is immediate to turn an algorithm that only gives a yes/no answer into an algorithm that, in case of a positive answer, returns a cutset that is a solution.

Hence, in all our algorithmic results, we can solve a compression version of the problem. That is, we can assume that our algorithm is additionally given on input a set $X \subseteq V(G)$ of size at most $k + 1$ such that $G - X$ already satisfies the desired separation (i.e., has no cycle with a red edge in case of SUBSET FEEDBACK EDGE SET etc.).

Furthermore, in the problems that involve vertex deletions (i.e., Sections 3 and 4), we can additionally branch on the set X into $2^{|X|}$ options, guessing a set $Y \subseteq X$ of vertices that are included in the sought solution. In each branch, we delete Y from the graph and the set X, decrease k by $|Y|$ and decrease W by the weight of Y. Furthermore, we set the weight of the remaining vertices of $X \setminus Y$ to $+\infty$, so they become undeletable. In other words, in Sections 3 and 4, we solve a disjoint compression variant of the problem, where the sought solution is supposed to be disjoint with the set X.

2.3. Generalized Digraph Pair Cut. We will not need flow-augmentation in its raw form, but only one algorithmic corollary of this technique.

An instance of GENERALIZED DIGRAPH PAIR CUT (GDPC for short) consists of:

- a directed multigraph G with two distinguished vertices $s, t \in V(G)$;
- a multiset $\mathcal{C}$ of (unordered) pairs of vertices of G, called *clauses*;
- a family $\mathcal{B}$ of pairwise disjoint subsets of $E(G) \cup \mathcal{C}$ called *bundles* such that no bundle contains two copies of the same arc or two copies of the same clause;
- a weight function $\omega : \mathcal{B} \to \mathbb{Z}_+$;
- two integers k and W.

We now make a few definitions that help us understand the desired solution of the GDPC problem.

- A set $Z \subseteq E(G)$ is a *cut* in a GDPC instance $\mathcal{I} = (G, s, t, \mathcal{C}, \mathcal{B}, \omega, k, W)$ if $Z \subseteq E(G) \cap \bigcup_{B \in \mathcal{B}} B$ (i.e., Z contains only edges of bundles) and there is no path from s to t in $G - Z$.
- A cut Z *violates* an edge $e \in E(G)$ if $e \in Z$ and *violates* a clause $uv \in \mathcal{C}$ if both u and v are reachable from s in $G - Z$.
- A bundle is *violated* by Z if it contains an edge or a clause violated by Z.
- An edge, a clause, or a bundle not violated by Z is *satisfied* by Z.
- A cut Z is a *solution* if every clause violated by Z is part of a bundle, Z violates at most k bundles, and the total weight of violated bundles is at most W. (Recall that a cut is required to contain only edges of bundles, that is, it satisfies all edges outside bundles.) Note that k here is not the number of edges deleted.

The GDPC problem asks for an existence of a solution. GDPC, parameterized by k, is W[1]-hard even in the unweighted setting and without clauses: it suffices to have bundles consisting of two edges for the hardness [28]. However, flow-augmentation yields fixed-parameter tractability of some specific useful restrictions of GDPC.

For a bundle $B \in \mathcal{B}$, let $V(B)$ be the set of vertices that are involved in an arc or a clause of B and let G_B be an undirected graph with $V(G_B) = V(B) \setminus \{s, t\}$ and $uv \in E(G_B)$ if B contains an arc (u, v), an arc (v, u), or a clause uv. A bundle B is $2K_2$-free if G_B is $2K_2$-free, that is, it does not contain $2K_2$ (the four-vertex graph consisting of two independent edges) as an induced subgraph. An instance $\mathcal{I}$ of GDPC is $2K_2$-free if every bundle of $\mathcal{I}$ is $2K_2$-free. Finally, an instance $\mathcal{I}$ is b-bounded if for every $B \in \mathcal{B}$ we have $|V(B)| \leq b$.

One of the main algorithmic corollaries of the flow-augmentation technique is the tractability of $2K_2$-free b-bounded instances of GDPC.

THEOREM 2.1 ([21], Theorem 3.3).   *There exists a randomized polynomial-time algorithm for* GENERALIZED DIGRAPH PAIR CUT *restricted to $2K_2$-free b-bounded instances that never accepts a* NO*-instance and accepts a* YES*-instance with probability $2^{-\mathrm{poly}(k,b)}$.*

For DIRECTED SUBSET FEEDBACK EDGE SET it will be more convenient to look at a different restriction of GDPC. Let $\mathcal{I} = (G, s, t, \emptyset, \mathcal{B}, \omega, k, W)$ be a GDPC instance without clauses. An arc $e \in E(G)$ is *crisp* if it is not contained in any bundle, and *soft* otherwise. An arc $e \in E(G)$ is *deletable* if it is soft and there is no copy of e in G that is crisp. Note that a cut needs to contain soft arcs only, and in fact, we can restrict our attention to cuts containing only deletable arcs. A bundle $B \in \mathcal{B}$ has *pairwise linked deletable edges* if for every two deletable arcs $e_1, e_2 \in B$ that are not incident with either s or t, there is a path from an endpoint of one of the edges to an endpoint of the other that does not use an edge of another bundle (i.e., uses only edges of B and crisp edges).

In [20], a notion of BUNDLED CUT WITH ORDER has been introduced as one variant of GDPC without clauses that is tractable. In [21], it was observed that the notion of *pairwise linked deletable edges* is slightly more general than the "with order" assumption and is more handy.

THEOREM 2.2 ([21], Theorem 3.21).   *There exists a randomized polynomial-time algorithm that, given a GDPC instance $\mathcal{I} = (G, s, t, \emptyset, \mathcal{B}, \omega, k, W)$ with no clauses and whose every bundle has pairwise linked deletable edges, never accepts a* NO*-instance and accepts a* YES*-instance with probability $2^{-\mathcal{O}(k^4 d^4 \log(kd))}$ where d is the maximum number of deletable arcs in a single bundle.*

Note that if $\mathcal{I}$ is b-bounded, then $d \leq b^2$.

3. Multicut. This section is devoted to the proof of Theorem 1.1.

As discussed in Subsection 2.1, we can restrict ourselves to the vertex-deletion variant. Let $\mathcal{I} = (G, \mathcal{T}, \omega, k, W)$ be an instance of WEIGHTED MULTICUT. Let $T = \bigcup_{(s,t)\in\mathcal{T}}\{s, t\}$ be the set of all terminals. By a simple reduction, we can assume that all terminals have weight $+\infty$ and form an independent set: for every $(s, t) \in \mathcal{T}$, add a new vertex s' adjacent to s, add a new vertex t' adjacent to t, set $\omega(s') = \omega(t') = +\infty$ and replace (s, t) with (s', t') in $\mathcal{T}$.

We also use iterative compression, but in the ordering $v_1, \ldots, v_n$ of $V(G)$ we start with terminals. Note that the subgraph of G induced by the terminals is edgeless and thus admits a solution being the emptyset. As a result, using the standard iterative compression step discussed in Subsection 2.2 we can assume that the algorithm is given access to a set $X \subseteq V(G) \setminus T$ of size $|X| \leq k + 1$ such that for every $(s, t) \in \mathcal{T}$ there is no path from s to t in $G - X$ and we are to check if there is a solution disjoint with X. We can set $\omega(x) = +\infty$ for every $x \in X$.

We closely follow the steps in Section 5 of [29], reengineering only one branching step that originally uses important separators.

Fix a hypothetical solution Z. We first guess how the vertices of X are partitioned between connected components of $G - Z$. If two vertices of X are guessed to be in the same connected component of $G - Z$, we can merge them into a single vertex (recall that the solution Z is disjoint with X). This results in $2^{\mathcal{O}(k \log k)}$ subcases. After this step, we can assume that every connected component of $G - Z$ contains at most one vertex of X and X is an independent set. For brevity, we say that $Y \subseteq V(G) \setminus (X \cup T)$ is a *multiway cut* if every connected component of $G - Y$ contains at most one vertex of X. Thus, it suffices to develop a randomized FPT algorithm that (a) accepts with constant probability an instance that admits a solution that is a multiway cut of X; (b) never accepts a NO-instance.

An instance of WEIGHTED MULTICUT with X given by iterative compression step is *bipedal* if X is an independent set and for every connected component C of $G - X$, we have $|N_G(C)| \leq 2$, that is, C is adjacent to at most two vertices of X. In Section 3.2 we show how to reduce a bipedal instance to a GDPC instance handled by Theorem 2.1. We emphasize that we do not claim authorship of this reduction: while there is no citeable source of this reduction, it has been floating around in the community in the last years. The reduction, in the edge-deletion setting (and leading to an undirected analog of GDPC) has been spelled out in [18]. We include it here for completeness of the argument.

Section 3.1 describes a branching algorithm, closely following the arguments of [29], whose goal is to break connected components C of $G - X$ with $|N_G(C)| > 2$. In the leaves of the branching process, we obtain bipedal instances that are passed to the algorithm of Section 3.2.

3.1. Branching on a multilegged component. The algorithm is a recursive branching routine on an instance $(G, \mathcal{T}, \omega, k, W, X)$ where X is an independent set and a multicut for $\mathcal{T}$, and the hypothetical solution is also a multiway cut for X. In the beginning $|X| \leq k + 1$ as discussed earlier. During the branching algorithm, one may delete vertices, merge vertices or grow the set X while maintaining that the hypothetical solution is also a multiway cut for (the new) X.

In a recursive call, we start with a few cleaning steps. At every moment, apply the first applicable reduction step.

1. If $\emptyset$ is a solution, return YES.

2. If $k \le 0$, $W \le 0$, or X is not an independent set, return No.
3. If the number of connected components C of $G - X$ with $|N_G(C)| > 1$ is more than k, return No. (Note that every such component needs to contain at least one vertex of every multiway cut for X.)
4. If there exists $x \in X$ such that the cardinality of the minimum-cardinality vertex cut between x and $X \setminus x$ is of size larger than k, return No. (Recall that the solution is also a multiway cut for X.)
5. If there exists a vertex v that admits a family $\mathcal{P}$ of $k + 2$ paths that start in v, end in distinct vertices of X, and are vertex-disjoint except for v, delete v, decrease k by one, decrease W by $\omega(v)$, and recurse. (Note that every such vertex v needs to be included in any multiway cut for X of size at most k.)
6. If there exists a connected component C of G that do not contain both vertices of any terminal pair $(s, t) \in \mathcal{T}$ and contains at most one vertex of X, delete it and all terminal pairs involving a vertex of C. (Recall that for every $(s, t) \in \mathcal{T}$, the terminals s and t lie in different connected components of $G - X$. Hence, this rule applies to any component C that contains no vertex of X and to any isolated vertex of X.)
7. If $|X| > k(k+1)$, return No. (Since Step 5 is inapplicable, for every multiway cut Z, every $z \in Z$ is adjacent to at most $k + 1$ connected components of $G - Z$ that contain a vertex of X. Also, since Z is a multiway cut for X, every vertex of X is in a distinct connected component of $G - Z$. Further, since the previous rule (RR 6) is not applicable, there does not exist a connected component of $G - Z$ that has no neighbor in Z. Indeed, as such an isolated component will have at most one vertex from each terminal pair in $\mathcal{T}$ because Z is a solution and at most one vertex of X since Z is a multiway cut of Z. Therefore, $|X|$ is at most the number of connected components of $G - Z$ that intersect X, which is upper bounded by $|Z|(k + 1) \le k(k+1)$.)
8. If the current instance is bipedal, pass it to the algorithm of Section 3.2.

A component C of $G - X$ is *nontrivial* if $|N_G(C)| > 1$. If neither of the reduction steps is applicable, we have at most k nontrivial connected components (due to Rule 3) and the size of the neighbourhood of each component of $G - X$ is at most $|X| \le k(k+1)$ (due to Rule 7).

At every branching step, we will ensure that one of the following progresses happen in any recursive call:

- the instance will be resolved immediately by Reduction Rules 1–4, 7, or 8, or
- the parameter k decreases, or
- the parameter k stays the same, but the number of nontrivial connected components plus the number of vertices of X adjacent to a nontrivial component increases.

We observe that the reduction rules do not reverse the above progress. That is, Rule 5 can decrease the number of nontrivial connected components or the number of vertices of X incident with a nontrivial connected component, but at the same time decreases k by one, while Rule 6 cannot delete a nontrivial connected component.

After the application of the described reduction rules, the number of nontrivial components is at most k and the size of X is at most $k(k + 1)$. Thus, the depth of the recursion is bounded by $\mathcal{O}(k^3)$.

Let C be a component of $G - X$ with $|N_G(C)| > 2$. (It exists as the instance is not bipedal.) For a subset $B \subseteq C$ and a function $f : B \to N_G(C)$, we construct an instance $\mathcal{I}_f$ as follows: for every $v \in B$, we merge v onto the vertex $f(v)$ (we use $f(v)$ as the name of the resulting vertex and the resulting vertex still belongs to X).

We say that B is a *shattering set* if for every $f : B \to N_G(C)$, the instance $\mathcal{I}_f$ either contains strictly more nontrivial components than the current instance, or recursing on $\mathcal{I}_f$ will result in returning an immediate answer by one of the first four reduction rules.

The main technical contribution of Section 5 of [29] is the following statement.

LEMMA 3.1 ([29, Lemma 5.3]).   *Given an instance $(G, \mathcal{T}, \omega, k, W)$ together with a set $X \subseteq V(G) \setminus T$ such that in $G - X$ there is no path from s to t for any $(s, t) \in \mathcal{T}$, and a component C of $G - X$ with $|N_G(C)| > 2$, one can find a shattering set $B \subseteq C$ of size at most $3k$ in polynomial time.*

We apply Lemma 3.1 to C, obtaining a set B of size at most $3k$. We branch, guessing the first of the following options that happens with regards to a hypothetical solution Z:

1. There is a vertex $v \in B \cap Z$. We guess v, delete v from the graph, decrease k by one, decrease W by $\omega(v)$, and recurse. This gives $|B| \leq 3k$ subcases and in each subcase k drops.
2. For every $v \in B$, the connected component of $G - Z$ that contains v also contains a vertex of X. For every $v \in B$, we guess a vertex $f(v) \in N_G(C)$ that is in the same connected component of $G - Z$ as v. As $|N_G(C)| \leq |X| \leq k(k + 1)$ and $|B| \leq 3k$, there are $2^{\mathcal{O}(k \log k)}$ options for $f : B \to N_G(C)$. We recurse on $\mathcal{I}_f$. To see that we obtain progress, observe that:
   - the parameter k stays the same;
   - if X is not an independent set, the recursive call returns NO immediately;
   - otherwise, the fact that B is a shattering set implies that in each instance $\mathcal{I}_f$, the number of nontrivial components increases, while the connectivity of C implies that every vertex of $N_G(C)$ remains adjacent to a nontrivial connected component, so the set of vertices of X adjacent to a nontrivial connected component does not change.
3. There exists $v \in B$ such that the connected component of $G - Z$ that contains v is disjoint with X. Here, [29] branches on an important separator separating v from X. This does not work in the presence of weights, so we need to proceed differently. We insert v into X, set its weight to $+\infty$, and recurse. Clearly, the hypothetical solution Z remains a solution and, if the guess is correct, Z remains a multiway cut (with regards to the enlarged set X). To see that we obtain progress, observe that:
   - the parameter k stays the same;
   - if X is not an independent set, the recursive call returns NO immediately;
   - otherwise, first observe that in the right guess v has no neighbors in the set X; therefore, for every $y \in N_G(C)$, there exists a connected component C_y of $C - \{v\}$ with $y \in N_G(C_y)$ and as $v \in N_G(C_y)$ due to connectivity of C, C_y is a new nontrivial component; hence the number of vertices of X that are incident with a nontrivial connected component increases as both v and the whole $N_G(C)$ are now adjacent to nontrivial connected components; furthermore, the number of nontrivial connected components does not decrease as at least one new nontrivial component is created in the place of C since $N_G(C) \neq \emptyset$.

Hence, the recursive step invokes $2^{\mathcal{O}(k \log k)}$ recursive subcalls, in each obtaining the promised progress. Every single recursive call takes polynomial time. Consequently, the branching algorithm takes $2^{\mathcal{O}(k^4 \log k)} n^{\mathcal{O}(1)}$ time and results in $2^{\mathcal{O}(k^4 \log k)}$ leaves of the recursion trees that give either an immediate answer or a bipedal in-

stance, which is passed to Section 3.2.

3.2. Solving a bipedal instance. We now show how to reduce a bipedal instance to a GDPC instance where every bundle consists of at most two arcs and a single clause containing the heads of these two arcs. These bundles are $2K_2$-free and 4-bounded and hence can be solved by Theorem 2.1 in randomized FPT time $2^{k^{\mathcal{O}(1)}} n^{\mathcal{O}(1)}$. This is essentially repeating the arguments of Lemma 7.1 of [18], adjusted for the vertex-deletion setting and GDPC.

We start with a graph H consisting of vertices s and t. For every component C of $G - X$, proceed as follows. Recall that $|N_G(C)| \in \{1, 2\}$. Denote one of the elements of $N_G(C)$ as s_C and the other as t_C, if present. For every $v \in V(G)$, create four vertices v_s^+, v_s^-, v_t^+, v_t^-, arcs (v_s^-, v_s^+), (v_t^-, v_t^+), and a clause $v_s^+ v_t^+$. The two constructed arcs and the constructed clause form a bundle B_v of weight $\omega(v)$. These are all the bundles that we will construct; all subsequent arcs and clauses will not be in any bundle and thus will be undeletable. For every connected component C of $G - X$ and $uv \in E(G[C])$, add arcs (u_s^+, v_s^-), (v_s^+, u_s^-) (u_t^+, v_t^-), and (v_t^+, u_t^-). For every $vs_C \in E(G)$ with $v \in C$, add arcs (s, v_s^-) and (v_t^+, t). For every $vt_C \in E(G)$ with $v \in C$, add arcs (s, v_t^-) and (v_s^+, t). Observe that this creates an s to t path in H whenever there is an s_C to t_C path whose internal vertices are contained in C (this is formalized later in Lemmas 3.2 and 3.3). Since our solution is also a multiway cut for X, it needs to kill all s_C to t_C paths. In the new graph H, this will amount to deleting the s to t paths.

Finally, for every $(u, v) \in \mathcal{T}$ we proceed as follows. Note that u and v are in distinct connected components of $G - X$, say C_u and C_v. For every $x \in N_G(C_u) \cap N_G(C_v)$ we proceed as follows. Say $x = \alpha_{C_u}$ and $x = \beta_{C_v}$ for $\alpha, \beta \in \{s, t\}$. Add a clause $u_\alpha^- v_\beta^-$. This is done to ensure that the paths between the endpoints of any terminal pair (that definitely lie in two distinct components of $G - X$) are hit by the GDPC solution. This finishes the description of the GDPC instance $\mathcal{I}' = (H, s, t, \mathcal{C}, \mathcal{B}, \omega, k, W)$; see Figure 1 for a simple example of the construction. It is immediate that the instance satisfies the prerequisites of Theorem 2.1 with $b = 4$.

It remains to check the equivalence of the instance $\mathcal{I}'$ of GDPC with the input instance $\mathcal{I} = (G, \mathcal{T}, \omega, k, W)$ together with the set X. We do it in the next two lemmata, completing the proof of Theorem 1.1. Recall T is the set of all terminal vertices.

LEMMA 3.2. *If $Z \subseteq V(G) \setminus (X \cup T)$ is a solution that is also a multiway cut for X, then $Z' = \bigcup_{v \in Z} B_v \cap E(H)$ is a cut in $\mathcal{I}'$ that satisfies all clauses outside B_v for $v \in Z$.*

Proof. Assume first that $H - Z'$ contains a path P' from s to t. Observe that there exists a component C of $G - X$ and $\alpha \in \{s, t\}$ such that all internal vertices of P' are of the form v_α^+ or v_α^- for $v \in C$. Then, the path P' induces a path from s_C to t_C via C in $G - Z$, a contradiction to the assumption that Z is a multiway cut for X.

Assume now that Z' violates a clause $v_s^+ v_t^+$ in B_v. Then first observe that $v \in C$, for a component C of $G - X$. Let P'_s be a path from s to v_s^+ in $H - Z'$ and let P'_t be a path from s to v_t^+ in $H - Z'$. In $G - Z$, the path P'_s yields a path P_s from s_C to v and the path P'_t (reversed) yields a path P_t from v to t_C. Together, P_s and P_t yield a path from s_C to t_C in $G - Z$, a contradiction to the assumption that Z is a multiway cut.

Finally, assume that Z' violates a clause $u_\alpha^- v_\beta^-$ for some $(u, v) \in \mathcal{T}$, where C_u and C_v are the components of $G - X$ containing u and v, respectively, $x \in N_G(C_u) \cap$
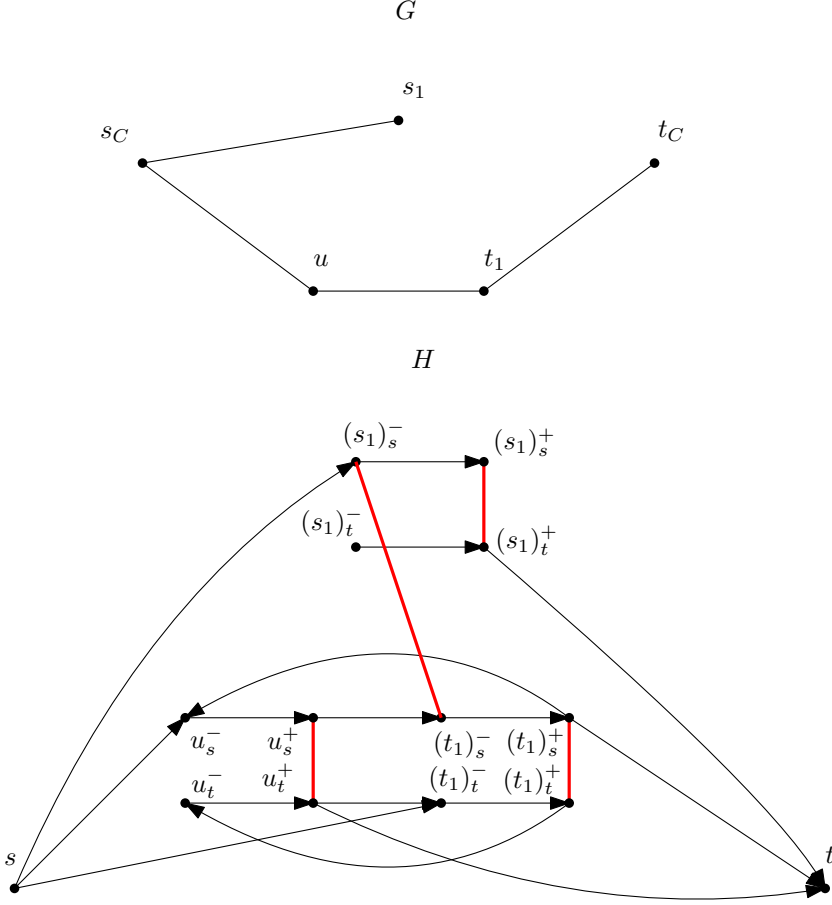
FIG. 1. *Illustration of the construction of Subsection 3.2. On the top, there is graph G with $(s_1, t_2) \in \mathcal{T}$ and $s_C, t_C \in X$. On the bottom, there is the corresponding constructed graph H with clauses depicted in red.*

$N_G(C_v)$, and $x = \alpha_{C_u}$, $x = \beta_{C_v}$ for $\alpha, \beta \in \{s, t\}$. Let P'_u be a path from s to u_α^- in $H - Z'$ and let P'_v be a path from s to v_β^- in $H - Z'$. In $G - Z$, P'_u yields a path P_u from u to $x = \alpha_{C_u}$ and P'_v yields a path P_v from v to $x = \beta_{C_v}$. Together, P_u and P_v yield a path from u to v in $G - Z$, a contradiction to the assumption that Z is a solution. □

LEMMA 3.3. *If Z' is a cut in $\mathcal{I}'$ that satisfies all clauses that are not in bundles and Z consists of those v such that Z' violates B_v, then Z is a solution to $\mathcal{I}$ that is also a multiway cut for X.*

Proof. We first show that Z is a multiway cut for X. By contradiction, assume that there exists a component C of $G - X$ and a path P from s_C to t_C via C that avoids Z. Let v be an arbitrary vertex of P in C. Then, the prefix of P from s_C to v lifts to a path P'_s in $H - Z'$ from s to v_s^+. Similarly, the suffix of P from v to t_C, reversed, lifts to a path P'_t in $H - Z'$ from s to v_t^+. Hence, Z' violates the clause $v_s^+ v_t^+$ and hence the bundle B_v, which is a contradiction.

Consider now $(u, v) \in \mathcal{T}$ and assume there is a path P from u to v in $G - Z$. Since Z is a multiway cut for X, P contains at most one vertex of X. Since u and

v are in distinct connected components of $G - X$ (say, C_u and C_v, respectively), P contains at least one vertex of X. That is, P starts in u, continues via C_u to a vertex $x \in N_G(C_u) \cap N_G(C_v)$, and then continues via C_v to v. The prefix of P from u to x (reversed) lifts to a path P'_u in $H - Z'$ from s to u_α^- where $x = \alpha_{C_u}$, $\alpha \in \{s, t\}$. The suffix of P from x to v lifts to a path P'_v in $H - Z'$ from s to v_β^- where $x = \beta_{C_v}$, $\beta \in \{s, t\}$. Hence, the clause $u_\alpha^- v_\beta^-$ is violated by Z', a contradiction. This finishes the proof of Lemma 3.3.   □

With the discussion above, Lemmata 3.2 and 3.3 conclude the proof of Theorem 1.1.

4. Group Feedback Edge/Vertex Set. This section is devoted to the proof of Theorem 1.3. In fact, we just closely follow the arguments of [9] and verify that they also work in the weighted setting. The algorithm reduces the problem to multiple instances of MULTIWAY CUT. Here, in the presence of weights, we apply the algorithm of Theorem 1.1 to solve WEIGHTED MULTIWAY CUT (in particular, we use Corollary 1.2).

As discussed in Subsection 2.1, we can focus on the vertex-deletion variant GROUP FEEDBACK VERTEX SET Using iterative compression (Subsection 2.2) we assume that, apart from the input instance (G, ψ, ω, k, W), we are given a set $X \subseteq V(G)$ of size at most $k + 1$ such that $G - X$ has no non-null cycles and the goal is to find a solution disjoint from X. We set $\omega(x) = +\infty$ for every $x \in X$. Recall that in this problem, the input graph G is equipped with a group Γ.

For a graph H with group labels ψ, a *consistent labeling* is a function $\phi : V(H) \to \Gamma$ such that $\phi(v) = \phi(u) + \psi(e, u)$ for every $e = uv \in E(H)$. It is easy to see that (H, ψ) has no non-null cycle if and only it admits a consistent labeling.

Untangling. By standard relabelling process, we can assume that $\psi(e, v) = 0$ for every $e \in E(G - X)$ and $v \in e$; we call such an instance *untangled*. Since $G - X$ has no non-null cycles, there exists $\phi : V(G) \setminus X \to \Gamma$ such that for every $e = uv \in E(G - X)$ we have $\phi(v) = \phi(u) + \psi(e, u)$. For every $e = uv \in E(G - X)$ we relabel $\psi(e, u) := \phi(u) + \psi(e, u) - \phi(v)$ and $\psi(e, v) := \phi(v) + \psi(e, v) - \phi(u)$. Furthermore, for every $e = uv \in E(G)$ with $u \in X$ but $v \notin X$, we relabel $\psi(e, u) := \psi(e, u) - \phi(v)$ and $\psi(e, v) = \phi(v) + \psi(e, v)$. It is easy to check that, after the above relabeling, for every closed walk C it does not change whether $\psi(C) = 0$ or not, while $\psi(e, v) = 0$ for every $e \in E(G - X)$ and $v \in e$.

Extending a labeling of X. We now observe that, given a labeling $\phi_0 : X \to \Gamma$, finding a set $Z \subseteq V(G) \setminus X$ such that ϕ_0 extends to a consistent labeling of $G - Z$ reduces to MULTIWAY CUT.

LEMMA 4.1. *There exists a randomized FPT algorithm with running time bound $2^{k^{O(1)}} n^{O(1)}$ that, given an untangled instance $(G, \psi, \omega, k, W, X)$ and a function $\phi_0 : X \to \Gamma$, checks if there is a set $Z \subseteq V(G) \setminus X$ of cardinality at most k and weight at most W such that $G - Z$ admits a consistent labeling extending ϕ_0.*

Proof. First, we check if for every $e = uv \in E(G[X])$ we indeed have $\phi_0(v) = \phi_0(u) + \psi(e, u)$, as otherwise, the answer is No. We construct a MULTIWAY CUT instance as follows. Let T be the set of those elements $g \in \Gamma$ such that there exists $uv \in E(G)$, $u \in X$, $v \notin X$, and $g = \phi_0(u) + \psi(uv, u)$ (i.e., in a consistent labeling extending ϕ_0, we would need to assign g to v). Note that $|T| \leq |E(G)|$. Let H be the graph consisting of a copy of $G - X$ (with weights inherited), the set T as additional vertices, and for every $uv \in E(G)$, $u \in X$, $v \notin X$, an edge from $\phi_0(u) + \psi(uv, u) \in T$ to v. A direct check shows that it suffices to solve the obtained MULTIWAY CUT

instance (G, T, ω, k, W) and return the answer (the proof of the equivalence is spelled out in the proof of Lemma 7 in [9]). □

Enumerating reasonable labelings of X. Since Γ can be large, we cannot enumerate all labelings $\phi_0 : X \to \Gamma$. In [9], a procedure is presented that enumerates a family of $2^{\mathcal{O}(k \log k)}$ labelings such that, for every solution Z, there is a consistent labeling of $G - Z$ that extends one of the enumerated labelings.

The main trick lies in the following reduction step. For $v \in V(G) \setminus X$ and $x \in X$, we define a flow graph $F(v, x)$ as follows. Let Γ_x be the set of those $g \in \Gamma$ such that there exists $xu \in E(G)$, $u \notin X$ and $\psi(xu, u) = g$. Note that $|\Gamma_x| \leq |E(G)|$. The graph $F(v, x)$ consists of a copy of $G - X$, the set Γ_x as additional vertices and, for every $xu \in E(G)$ with $u \notin X$, an edge $u\psi(xu, u)$.

We have the following statement.

LEMMA 4.2 (Lemma 8 of [9]).   *If there are $k + 2$ paths in $F(v, x)$ from v to distinct elements of Γ_x that are vertex-disjoint except for v, then v is contained in every solution of cardinality at most k.*

The condition of Lemma 4.2 can be checked in polynomial time. If such a vertex v is discovered, we can delete it, decrease k by one, decrease W by $\omega(v)$, and repeat the analysis.

Fix $x, y \in X$, $x \neq y$. An *external path* from x to y is a path with endpoints x and y and all internal vertices in $G - X$; note that an edge xy is also an external path. Let $\Gamma(x, y)$ be the set of all elements $g \in \Gamma$ such that there exists an external path P from x to y with $\psi(P) = g$. We also have the following statement.

LEMMA 4.3 (Lemma 9 of [9]).   *If there is no vertex v as in Lemma 4.2, but for some $x, y \in X$, $x \neq y$ we have $|\Gamma(x, y)| \geq k^3(k+1)^2 + 2$, then there is no solution of cardinality at most k.*

The condition of Lemma 4.3 can be again checked in polynomial time and, if we find that $\Gamma(x, y)$ is too large for some $x, y \in X$, $x \neq y$, we return No.

Otherwise, we enumerate reasonable labelings $\phi_0 : X \to \Gamma$ as follows. First, we guess how X is partitioned into connected components of $G - Z$ for a hypothetical solution Z; in every connected component, we can set ϕ_0 independently. Let $Y \subseteq X$ be a set of vertices guessed to be in the same connected component of $G - Z$; note that necessarily Y needs to live in the same connected component of G, so $\Gamma(x, y) \neq \emptyset$ for every distinct $x, y \in Y$. Fix $y \in Y$ and set $\phi_0(y) = 0$. Note that in a consistent labeling of $G - Z$ that assigns the value of 0 to y, for $x \in Y \setminus \{y\}$ the value assigned to x needs to be in $\Gamma(y, x)$ as a path P from y to x in $G - Z$ has $\psi(P) \in \Gamma(y, x)$. By Lemma 4.3, there are only $\mathcal{O}(k^5)$ options for $\phi_0(x)$. Overall, this gives $2^{\mathcal{O}(k \log k)}$ options for ϕ_0, as desired.

This finishes the proof of Theorem 1.3.

5. Directed Subset Feedback Edge/Vertex Set. This section is devoted to the proof of Theorem 1.5. As discussed in Section 2.1, we can restrict ourselves to the edge-deletion version, that is, to the DIRECTED SUBSET FEEDBACK EDGE SET problem. Furthermore, we can assume that red edges are undeletable (of weight $+\infty$): for every $e = (u, v) \in R$, we subdivide e, replacing it with a path $u \to x_e \to v$; the edge (u, x_e) becomes red and of weight $+\infty$, and (x_e, v) is not red and inherits the weight of e.

Let $\mathcal{I} = (G, R, \omega, k, W)$ be the input instance. Using iterative compression, we can assume we are given access to a set $X \subseteq V(G)$ of size at most $k + 1$ such that

$G - X$ has no cycle involving a red edge.

Let $Z \subseteq E(G) \setminus R$. Observe that $G - Z$ has no cycle containing a red edge if and only if for every $(u, v) \in R$, there is no path from v to u in $G - Z$. The latter condition is equivalent to u and v being in different strong connected components of $G - Z$. We will use the above reformulations of the desired separation property interchangeably.

Let Z be a sought solution. We start with some branching steps. First, we guess how the vertices of X are partitioned between strong connected components of $G - Z$. We identify vertices of X that are guessed to be in the same connected components of $G - Z$; note that in the branch where the guess is correct, this does not change whether two vertices of $G - Z$ are in the same strong connected component or not. Henceforth, by somewhat abusing the notation, we can assume that the vertices of X lie in distinct strong connected components of $G - Z$. We guess the order of X in a topological ordering of the strong connected components of $G - Z$; that is, we guess an enumeration of X as $x_1, x_2, \ldots, x_{|X|}$ such that in $G - Z$ there is no path from x_j to x_i for $1 \leq i < j \leq |X|$. Since initially $|X| \leq k + 1$, there are $2^{\mathcal{O}(k \log k)}$ branches up to this point and we retain the property $|X| \leq k + 1$.

We now construct a GDPC instance $\mathcal{I}'$. We first construct a graph H as follows. We start from $2|X| + 1$ copies of the graph G, denoted G^a for $1 \leq a \leq 2|X| + 1$. For $u \in V(G)$, let u^a be the copy of u in the graph G^a. For every $1 \leq a < b \leq 2|X| + 1$ and every $u \in V(G)$ we add an arc (u^b, u^a). For every red arc $(u, v) \in R$ and every $1 \leq a \leq |X|$, we add an arc (u^{2a}, v^{2a+1}). Finally, we introduce two new vertices s and t and, for every $1 \leq a \leq X$ and $1 \leq b \leq 2|X| + 1$ an arc (s, x_a^b) if $2a \geq b$ and an arc (x_a^b, t) if $2a < b$.

For every $e = (u, v) \in E(G) \setminus R$, we make a bundle B_e consisting of all $2|X| + 1$ copies of the arc e. We set $\omega(B_e) = \omega(e)$. This finishes the description of a GDPC instance $\mathcal{I}' = (H, s, t, \emptyset, \mathcal{B}, \omega, k, W)$ with no clauses. See Figure 2.

We observe that the obtained instance has pairwise linked deletable edges, due to the existence of crisp arcs (u^b, u^a) for every $u \in V(G)$ and $1 \leq a < b \leq 2|X| + 1$. Furthermore, every bundle contains at most $2|X| + 1 \leq 2k + 3$ deletable edges. Thus, by Theorem 2.2, we can resolve it in randomized FPT time $2^{\mathcal{O}(k^8 \log k)} n^{\mathcal{O}(1)}$.

It remains to show that the answer to $\mathcal{I}'$ is actually meaningful. This is done in the next two lemmata that complete the proof of Theorem 1.5.

LEMMA 5.1. *Let $Z \subseteq E(G) \setminus R$ be such that $G - Z$ has no cycle containing a red edge, and, additionally, all vertices of X lie in distinct strong connected components of $G - Z$, and there is no path from x_j to x_i in $G - Z$ for every $1 \leq i < j \leq |X|$. Then $Z' = \bigcup_{e \in Z} B_e$ is a solution to $\mathcal{I}'$.*

Proof. By contradiction, assume that $G - Z'$ contains a path P' from s to t. Pick such a path P' that minimizes the number of indices a such that P' contains a vertex of G^a. Let $a \in \{1, \ldots, 2|X| + 1\}$ be the minimum index such that P' contains a vertex of G^a and let $u^a \in V(P')$ be the last vertex of P' in G^a. Observe that if (s, x_i^b) is the first edge of P', then H also contains crisp edges $(s, x_i^{b'})$ for every $1 \leq b' \leq b$. Hence, by the minimality of a, we can modify P' so that the entire prefix from s to u^a is contained in G^a: whenever P' traverses a vertex $v^{a'}$, we instead traverse the vertex v^a.

Symmetrically, by choosing a to be maximum such that P' contains a vertex of G^a and $u^a \in V(P')$ to be the first vertex of P' in G^a, we observe that we can replace the suffix of P' from u^a to t so that it is completely contained in G^a.

Observe that the only edges of H that lead from G^a to G^b for $a < b$ are edges of the form (u^{2a}, v^{2a+1}) for $1 \leq a \leq |X|$ and red arcs (u, v). Hence, we can assume that
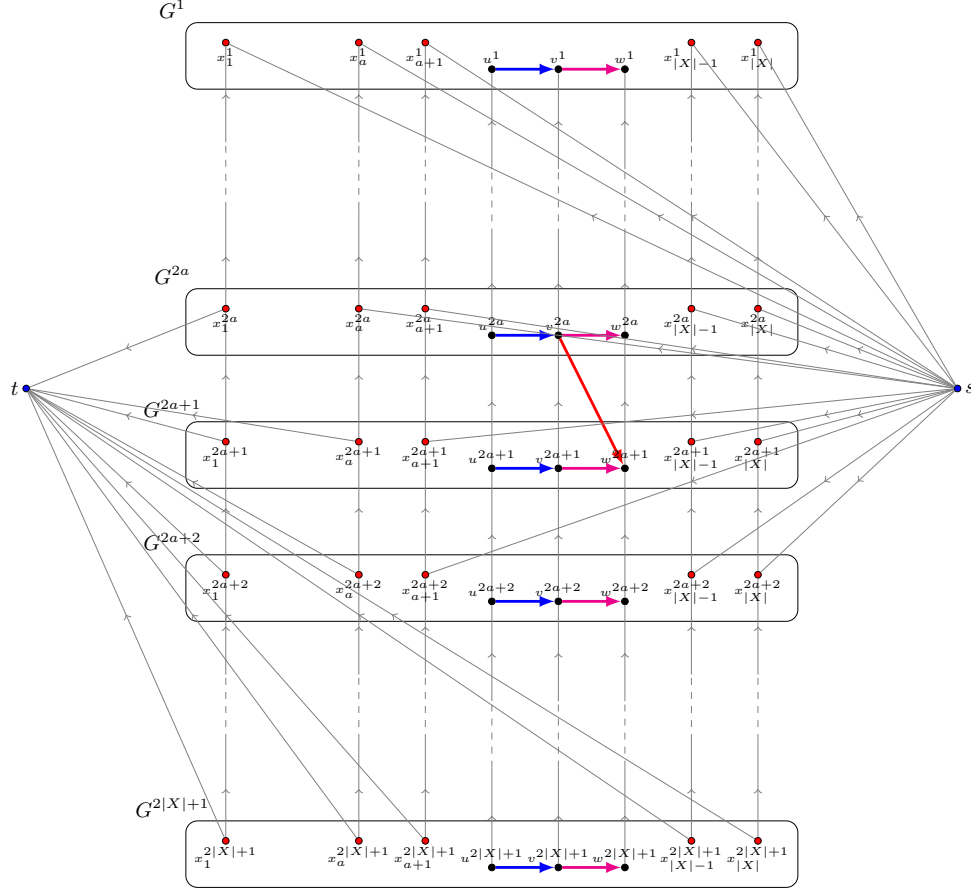
FIG. 2. *Illustration of the reduction of Section 5. All copies of an edge e of G form a bundle B_e: here, blue edges form one bundle for an edge (u, v) and magenta edges form another bundle for an edge (v, w). Furthermore, if (v, w) is red, then there is an extra arc (v^{2a}, w^{2a+1}) for every $1 \le a \le |X|$, depicted in red. Intuitively, this arc together with arcs (s, x_a^{2a}) and (x_a^{2a+1}, t) asks to destroy all closed walks that pass both through (v, w) and x_a.*

the path P' is of one of the following two types:

1. All internal vertices of P' lie in the same graph G^a.
2. For some $1 \le a \le |X|$, the path P' first goes from s via G^{2a}, then uses one edge (u^{2a}, v^{2a+1}) for some $(u, v) \in R$, and then continues via G^{2a+1} to t.

In the first case, let (s, x_j^a) be the first edge of P' and let (x_i^a, t) be the last edge of P'. By construction of H, we have $2j \ge a > 2i$, so $j > i$. Thus, P' without the first and the last edge gives a path in $G - Z$ from x_j to x_i for some $j > i$, a contradiction.

In the second case, let (s, x_j^{2a}) be the first edge of P' and let (x_i^{2a+1}, t) be the last edge of P'. By construction of H, we have $2j \ge 2a$ and $2a + 1 > 2i$, so $j \ge i$. If $j > i$, P' without the first and the last edge gives a path from x_j to x_i, again a contradiction as in the first case. If $j = i$, then the subpath of P' from v^{2a+1} to x_i^{2a+1} gives a path from v to x_i in $G - Z$ and the subpath of P' from x_j^{2a} to u^{2a} gives a path from x_j to u in $G - Z$. As $j = i$, this gives a path from v to u in $G - Z$, a contradiction as (u, v) is a red edge. $\qed$

LEMMA 5.2. *Let Z' be a cut in $\mathcal{I}'$ and let $Z = \{e \in E(G) \setminus R \mid B_e \cap Z' \neq \emptyset\}$. Then $G - Z$ contains no cycle containing a red edge.*

Proof. By contradiction, assume $G - Z$ contains a path P from v to u for some $(u, v) \in R$. Since $G - X$ contains no such path, P contains a vertex of X. Let $x_i \in V(P)$. Let P_v be the prefix of P from v to x_i and let P_u be the suffix of P from x_i to u. Consider the copy P_v^{2i+1} of P_v in G^{2i+1} and the copy P_u^{2i} of P_u in G^{2i}. Then, since Z' contains no edge of B_e for any $e \in E(P)$, P_v^{2i+1} and P_u^{2i} are disjoint with Z'. This is a contradiction, as a concatenation of (s, x_i^{2i}), P_u^{2i}, (u^{2i}, v^{2i+1}), P_v^{2i+1}, and (x_i^{2i+1}, t) is a path from s to t in $H - Z'$. □

6. Conclusions. We showed fixed-parameter tractability of a number of weighted graph separation problems. Our first result extends a recent result of Galby et al. [13], who considered the special case of weighted MULTICUT in trees. For all our algorithms, we revisited an old combinatorial approach to the problem, adjusted it to weights, and provided a reduction to GDPC in one of its tractable variants. The application of the technique of flow-augmentation is hidden in the algorithms for GDPC (Theorems 2.1 and 2.2).

We would like to highlight here one graph separation problem that resisted our attempts: DIRECTED SYMMETRIC MULTICUT. Here, the input consists of a directed graph G, weights $\omega : E(G) \to \mathbb{Z}_+$ (that is, we consider an edge-deletion variant, but, as we are working with directed graphs, it is straightforward to reduce between edge- and vertex-deletion variants), integers k and W, and a set $\mathcal{T} \subseteq \binom{V(G)}{2}$ of unordered pairs of vertices of G. The problem asks for an existence of a set $Z \subseteq E(G)$ of size at most k and total weight at most W such that for every $uv \in \mathcal{T}$, the vertices u and v are not in the same strong connected component of $G - Z$ (i.e., Z cuts all paths from u to v or cuts all paths from v to u). Eiben, Rambaud, and Wahlström [12] considered the parameterized complexity of DIRECTED SYMMETRIC MULTICUT and gave partial results, but the main problem of the parameterized complexity of DIRECTED SYMMETRIC MULTICUT parameterized by k remains open, even in the unweighted setting.

To motivate the DIRECTED SYMMETRIC MULTICUT problem further, we point out that it has a very natural reformulation in the context of *temporal CSPs*, that is, constraint satisfaction problems with domain $\mathbb{Q}$ and access to the order on $\mathbb{Q}$. More formally, a *temporal CSP relation* is an FO formula with a number of free variables that can be accessed via comparison predicates $x = y$, $x \neq y$, $x < y$, and $x \leq y$. A *temporal CSP language* is a set of temporal CSP relations. For a temporal CSP language Λ, an instance of CSP(Λ) consists of a set of variables X and a set $\mathcal{C}$ of constraints; each constraint is an application of a formula from Λ to a tuple of variables from X. The goal is to find an assignment $\alpha : X \to \mathbb{Q}$ that satisfies all constraints. In the MAX SAT(Λ) problem, we are additionally given an integer k and the goal is to satisfy all but k constraints (i.e., delete at most k constraints to get a satisfiable instance).

In various CSP contexts, the MAX SAT(Λ) problem is usually hard, yet the parameterized complexity landscape with k as a parameter is often rich; see, e.g., the recent dichotomy for the Boolean domain [21] and references therein. The P vs NP dichotomy for temporal CSP(Λ) is known since over a decade [1]. Can we establish parameterized complexity dichotomy for temporal MAX SAT(Λ) parameterized by k?

One of the most prominent examples of temporal CSP languages is

$$\Lambda = \{x = y, x \neq y, x < y, x \leq y\},$$

called a *point algebra*. Here, $\mathrm{CSP}(\Lambda)$ is known to be polynomial-time solvable. We observe that MAX SAT(Λ) is equivalent to the (unweighted) DIRECTED SYMMETRIC MULTICUT.

In one direction, given an unweighted DIRECTED SYMMETRIC MULTICUT instance $(G, \mathcal{T}, k)$, we set $X = V(G)$, model every arc $(u, v) \in E(G)$ as a constraint $u \leq v$ and each pair $uv \in \mathcal{T}$ as $k + 1$ copies of a constraint $u \neq v$. Intuitively, a desired assignment $\alpha : V(G) \to \mathbb{Q}$ maps all vertices of the same strong connected component to the same number, and otherwise sorts the strong connected components according to a topological ordering.

The other direction is slightly more involved due to some technicalities. First, we replace each constraint $x = y$ with a pair of constraints $x \leq y$ and $y \leq x$; note that we will never want to delete both such constraints. Similarly, we replace each constraint $x < y$ with $x \neq y$ and $x \leq y$; again we will never want to delete both resulting constraints. Thus, we can assume that the instance uses only $x \neq y$ and $x \leq y$ constraints. Then, for every constraint $x \neq y$, we introduce fresh copies x' and y' of x and y, introduce constraints $x \leq x'$, $x' \leq x$, $y \leq y'$, $y' \leq y$, and $k + 1$ copies of $x' \neq y'$, and delete $x \neq y$. Now deleting $x \neq y$ is equivalent to deleting one of the inequalities, say $x \leq x'$, and setting x' to some very small number different than y and y'. Thus, we end up in an instance where only $x \leq y$ and $x \neq y$ constraints are present, and the latter constraints are always undeletable (appear in batches of $k + 1$ copies). Now, we can directly model it as DIRECTED SYMMETRIC MULTICUT: we set $V(G) = X$, for every constraint $x \leq y$ we add an arc (x, y) and for every batch of $k + 1$ constraints $x \neq y$ we add a pair xy to $\mathcal{T}$.

With a very similar reduction we observe that for $\Lambda' = \{x < y, x \leq y\}$ the problem MAX SAT(Λ') is equivalent to (unweighted) DIRECTED SUBSET FEEDBACK EDGE SET: every constraint $x < y$ is equivalent to a red arc (x, y) and every constraint $x \leq y$ is equivalent to a non-red arc (x, y).

Therefore, the unresolved status of the parameterized complexity of DIRECTED SYMMETRIC MULTICUT stands as the main obstacle to obtain a dichotomy for parameterized complexity of MAX SAT(Λ) for temporal CSP languages Λ, parameterized by the deletion budget k.

REFERENCES

[1] M. BODIRSKY AND J. KÁRA, *The complexity of temporal constraint satisfaction problems*, J. ACM, 57 (2010), pp. 9:1–9:41, https://doi.org/10.1145/1667053.1667058.

[2] N. BOUSQUET, J. DALIGAULT, AND S. THOMASSÉ, *Multicut is FPT*, SIAM J. Comput., 47 (2018), pp. 166–207, https://doi.org/10.1137/140961808.

[3] 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), pp. 21:1–21:19, https://doi.org/10.1145/1411509.1411511.

[4] R. CHITNIS, M. CYGAN, M. HAJIAGHAYI, AND D. MARX, *Directed subset feedback vertex set is fixed-parameter tractable*, ACM Trans. Algorithms, 11 (2015), pp. 28:1–28:28, https://doi.org/10.1145/2700209.

[5] R. CHITNIS, M. CYGAN, M. HAJIAGHAYI, M. PILIPCZUK, AND M. PILIPCZUK, *Designing FPT algorithms for cut problems using randomized contractions*, SIAM J. Comput., 45 (2016), pp. 1171–1229, https://doi.org/10.1137/15M1032077.

[6] R. CHITNIS, L. EGRI, AND D. MARX, *List H-coloring a graph by removing few vertices*, Algorithmica, 78 (2017), pp. 110–146, https://doi.org/10.1007/s00453-016-0139-6.

[7] R. Chitnis, M. Hajiaghayi, and D. Marx, *Fixed-parameter tractability of directed multiway cut parameterized by the size of the cutset*, SIAM J. Comput., 42 (2013), pp. 1674–1696, https://doi.org/10.1137/12086217X.

[8] M. Cygan, P. Komosa, D. Lokshtanov, M. Pilipczuk, M. Pilipczuk, S. Saurabh, and M. Wahlström, *Randomized contractions meet lean decompositions*, ACM Trans. Algorithms, 17 (2021), pp. 6:1–6:30, https://doi.org/10.1145/3426738.

[9] M. Cygan, M. Pilipczuk, and M. Pilipczuk, *On group feedback vertex set parameterized by the size of the cutset*, Algorithmica, 74 (2016), pp. 630–642, https://doi.org/10.1007/s00453-014-9966-5.

[10] M. Cygan, M. Pilipczuk, M. Pilipczuk, and J. O. Wojtaszczyk, *On multiway cut parameterized above lower bounds*, ACM Trans. Comput. Theory, 5 (2013), pp. 3:1–3:11, https://doi.org/10.1145/2462896.2462899.

[11] M. Cygan, M. Pilipczuk, M. Pilipczuk, and J. O. Wojtaszczyk, *Subset feedback vertex set is fixed-parameter tractable*, SIAM J. Discrete Math., 27 (2013), pp. 290–309, https://doi.org/10.1137/110843071.

[12] E. Eiben, C. Rambaud, and M. Wahlström, *On the parameterized complexity of symmetric directed multicut*, in 17th International Symposium on Parameterized and Exact Computation, IPEC 2022, September 7-9, 2022, Potsdam, Germany, H. Dell and J. Nederlof, eds., vol. 249 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, pp. 11:1–11:17, https://doi.org/10.4230/LIPIcs.IPEC.2022.11, https://doi.org/10.4230/LIPIcs.IPEC.2022.11.

[13] E. Galby, D. Marx, P. Schepper, R. Sharma, and P. Tale, *Parameterized complexity of weighted multicut in trees*, in Graph-Theoretic Concepts in Computer Science - 48th International Workshop, WG 2022, Tübingen, Germany, June 22-24, 2022, Revised Selected Papers, M. A. Bekos and M. Kaufmann, eds., vol. 13453 of Lecture Notes in Computer Science, Springer, 2022, pp. 257–270, https://doi.org/10.1007/978-3-031-15914-5_19, https://doi.org/10.1007/978-3-031-15914-5_19.

[14] S. Guillemot, *FPT algorithms for path-transversal and cycle-transversal problems*, Discrete Optimization, 8 (2011), pp. 61–71, https://doi.org/10.1016/j.disopt.2010.05.003.

[15] M. Hatzel, L. Jaffke, P. T. Lima, T. Masařík, M. Pilipczuk, R. Sharma, and M. Sorge, *Fixed-parameter tractability of DIRECTED MULTICUT with three terminal pairs parameterized by the size of the cutset: twin-width meets flow-augmentation*, in Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023, N. Bansal and V. Nagarajan, eds., SIAM, 2023, pp. 3229–3244, https://doi.org/10.1137/1.9781611977554.ch123, https://doi.org/10.1137/1.9781611977554.ch123.

[16] Y. Iwata, M. Wahlström, and Y. Yoshida, *Half-integrality, LP-branching, and FPT algorithms*, SIAM J. Comput., 45 (2016), pp. 1377–1411, https://doi.org/10.1137/140962838.

[17] K. Kawarabayashi and M. Thorup, *The minimum k-way cut of bounded size is fixed-parameter tractable*, in IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011, R. Ostrovsky, ed., IEEE Computer Society, 2011, pp. 160–169, https://doi.org/10.1109/FOCS.2011.53.

[18] E. J. Kim, S. Kratsch, M. Pilipczuk, and M. Wahlström, *Solving hard cut problems via flow-augmentation*, CoRR, abs/2007.09018 (2020), https://doi.org/10.48550/arXiv.2007.09018.

[19] E. J. Kim, S. Kratsch, M. Pilipczuk, and M. Wahlström, *Solving hard cut problems via flow-augmentation*, in Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021, D. Marx, ed., SIAM, 2021, pp. 149–168, https://doi.org/10.1137/1.9781611976465.11.

[20] E. J. Kim, S. Kratsch, M. Pilipczuk, and M. Wahlström, *Directed flow-augmentation*, in STOC 2022: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022, S. Leonardi and A. Gupta, eds., ACM, 2022, pp. 938–947, https://doi.org/10.1145/3519935.3520018.

[21] E. J. Kim, S. Kratsch, M. Pilipczuk, and M. Wahlström, *Flow-augmentation III: complexity dichotomy for boolean csps parameterized by the number of unsatisfied constraints*, in Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023, N. Bansal and V. Nagarajan, eds., SIAM, 2023, pp. 3218–3228, https://doi.org/10.1137/1.9781611977554.ch122, https://doi.org/10.1137/1.9781611977554.ch122.

[22] S. Kratsch, M. Pilipczuk, M. Pilipczuk, and M. Wahlström, *Fixed-parameter tractability of multicut in directed acyclic graphs*, SIAM J. Discrete Math., 29 (2015), pp. 122–144, https://doi.org/10.1137/120904202.

[23] S. Kratsch and M. Wahlström, *Compression via matroids: A randomized polynomial kernel*

for odd cycle transversal, ACM Trans. Algorithms, 10 (2014), pp. 20:1–20:15, https://doi.org/10.1145/2635810.

[24] S. Kratsch and M. Wahlström, *Representative sets and irrelevant vertices: New tools for kernelization*, J. ACM, 67 (2020), pp. 16:1–16:50, https://doi.org/10.1145/3390887.

[25] D. Lokshtanov, M. S. Ramanujan, and S. Saurabh, *When recursion is better than iteration: A linear-time algorithm for acyclicity with few error vertices*, in Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018, A. Czumaj, ed., SIAM, 2018, pp. 1916–1933, https://doi.org/10.1137/1.9781611975031.125.

[26] D. Marx, *Parameterized graph separation problems*, Theor. Comput. Sci., 351 (2006), pp. 394–406, https://doi.org/10.1016/j.tcs.2005.10.007.

[27] D. Marx, B. O'Sullivan, and I. Razgon, *Finding small separators in linear time via treewidth reduction*, ACM Trans. Algorithms, 9 (2013), pp. 30:1–30:35, https://doi.org/10.1145/2500119.

[28] D. Marx and I. Razgon, *Constant ratio fixed-parameter approximation of the edge multicut problem*, Inf. Process. Lett., 109 (2009), pp. 1161–1166, https://doi.org/10.1016/j.ipl.2009.07.016.

[29] D. Marx and I. Razgon, *Fixed-parameter tractability of multicut parameterized by the size of the cutset*, in Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011, L. Fortnow and S. P. Vadhan, eds., ACM, 2011, pp. 469–478, https://doi.org/10.1145/1993636.1993699.

[30] D. Marx and I. Razgon, *Fixed-parameter tractability of multicut parameterized by the size of the cutset*, SIAM J. Comput., 43 (2014), pp. 355–388, https://doi.org/10.1137/110855247.

[31] M. Pilipczuk and M. Wahlström, *Directed multicut is W[1]-hard, even for four terminal pairs*, ACM Trans. Comput. Theory, 10 (2018), pp. 13:1–13:18, https://doi.org/10.1145/3201775.

[32] I. Razgon and B. O'Sullivan, *Almost 2-SAT is fixed-parameter tractable*, J. Comput. Syst. Sci., 75 (2009), pp. 435–450, https://doi.org/10.1016/j.jcss.2009.04.002.

[33] B. A. Reed, K. Smith, and A. Vetta, *Finding odd cycle transversals*, Oper. Res. Lett., 32 (2004), pp. 299–301, https://doi.org/10.1016/j.orl.2003.10.009.

[34] S. Saurabh, *What's next? future directions in parameterized complexity*, 2017, https://rapctelaviv.weebly.com/uploads/1/0/5/3/105379375/future.pdf. Recent Advances in Parameterized Complexity school, Tel Aviv, December 2017.

Appendix A. Problem Definitions.
In this section, we give formal descriptions of the problems discussed in this paper.

Weighted Multicut (vertex-/edge-deletion variant)   **Parameter:** k
Input: An undirected graph G, a set $\mathcal{T} \subseteq V(G) \times V(G)$, a weight function $\omega : V(G) \to \mathbb{N}$ (resp. $\omega : E(G) \to \mathbb{N}$), and positive integers k, W
Question: Does there exist $S \subseteq V(G)$ (resp. $S \subseteq E(G)$) such that $|S| \leq k$, $\omega(S) \leq W$ and, $G - S$ has no path from s to t for any $(s, t) \in \mathcal{T}$.

Weighted Multiway Cut (vertex-/edge-deletion variant)   **Parameter:** k
Input: An undirected graph G, a set $T \subseteq V(G)$, a weight function $\omega : V(G) \to \mathbb{N}$ (resp. $\omega : E(G) \to \mathbb{N}$), and positive integers k, W
Question: Does there exist $S \subseteq V(G)$ (resp. $S \subseteq E(G)$) such that $|S| \leq k$, $\omega(S) \leq W$ and, $G - S$ has no path from t to t' for any $t, t' \in T$, $t \neq t'$.

WEIGHTED GROUP FEEDBACK VERTEX/EDGE SET   **Parameter:** k
Input: An undirected graph G equipped with a group Γ which is not necessarily Abelian,
an assignment ψ, called the group labels, from a pair (e, v), where $e \in E(G)$ and $v \in e$, to an element $\psi(e, v) \in \Gamma$ such that for any $e = uv \in E(G)$ $\psi(e, u) + \psi(e, v) = 0$,
a weight function $\omega : V(G) \to \mathbb{N}$ (resp. $\omega : E(G) \to \mathbb{N}$), and positive integers k, W
Question: Does there exist $S \subseteq V(G)$ (resp. $S \subseteq E(G)$) such that $|S| \leq k$, $\omega(S) \leq W$ and, $G - S$ has no non-null cycle C, that is, if $C = (v_1, e_1, v_2, e_2, \ldots, v_\ell, e_\ell, v_{\ell+1})$, then $\psi(C) = \sum_{i=1}^{\ell} \psi(e_i, v_i) \neq 0$.

WEIGHTED DIRECTED FEEDBACK VERTEX/EDGE SET   **Parameter:** k
Input: A directed graph G, a weight function $\omega : V(G) \to \mathbb{N}$ (resp. $\omega : E(G) \to \mathbb{N}$), and positive integers k, W
Question: Does there exist a set $S \subseteq V(G)$ (resp. $S \subseteq E(G)$) such that $G - S$ is a directed acyclic graph.

WEIGHTED DIRECTED SUBSET FEEDBACK VERTEX/EDGE SET   **Parameter:** k
Input: A directed graph G, a set $R \subseteq E(G)$, a weight function $\omega : V(G) \to \mathbb{N}$ (resp. $\omega : E(G) \to \mathbb{N}$, and positive integers k, Ws)
Question: Does there exist $S \subseteq V(G)$ (resp. $S \subseteq E(G)$) such that $G - S$ has no directed cycle that contains at least one edge of R.

WEIGHTED SKEW MULTICUT (vertex-/edge-deletion variant)   **Parameter:** k
Input: A directed graph G, an ordered set $\{(s_1, t_1), \ldots, (s_\ell, t_\ell)\} \subseteq V(G) \times V(G)$, a weight function $\omega : V(G) \to \mathbb{N}$ (resp. $\omega : E(G) \to \mathbb{N}$), and positive integers k, W
Question: Does there exist $S \subseteq V(G)$ (resp. $S \subseteq E(G)$) such that $|S| \leq k$, $\omega(S) \leq W$ and, $G - S$ has no path from s_i to t_j for any $1 \leq i \leq j \leq \ell$.

DIRECTED SYMMETRIC MULTICUT   **Parameter:** k
Input: A directed graph G, a set $\mathcal{T} \subseteq V(G) \times V(G)$, a weight function $\omega : E(G) \to \mathbb{N}$, and positive integers k, W
Question: Does there exist $S \subseteq E(G)$ such that $|S| \leq k$, $\omega(S) \leq W$ and for each $(s, t) \in \mathcal{T}$, $G - S$ has either no path from s to t or no path from t to s.