Path-Contractions, Edge Deletions and Connectivity Preservation*†

Gregory Gutin[1], M. S. Ramanujan[2], Felix Reidl[3], and Magnus Wahlström[4]

1 Royal Holloway University, University of London, Egham, UK
G.Gutin@rhul.ac.uk
2 Algorithms and Complexity Group, TU Wien, Vienna, Austria
ramanujan@ac.tuwien.ac.at
3 Royal Holloway University, University of London, Egham, UK
felix.reidl@gmail.com
4 Royal Holloway University, University of London, Egham, UK
Magnus.Wahlstrom@rhul.ac.uk

Abstract

We study several problems related to graph modification problems under connectivity constraints from the perspective of parameterized complexity: (Weighted) Biconnectivity Deletion, where we are tasked with deleting k edges while preserving biconnectivity in an undirected graph, Vertex-deletion Preserving Strong Connectivity, where we want to maintain strong connectivity of a digraph while deleting exactly k vertices, and Path-contraction Preserving Strong Connectivity, in which the operation of path contraction on arcs is used instead. The parameterized tractability of this last problem was posed in [Bang-Jensen and Yeo, Discrete Applied Math 2008] as an open question and we answer it here in the negative: both variants of preserving strong connectivity are W[1]-hard. Preserving biconnectivity, on the other hand, turns out to be fixed parameter tractable (FPT) and we provide an FPT algorithm that solves Weighted Biconnectivity Deletion. Further, we show that the unweighted case even admits a randomized polynomial kernel. All our results provide further interesting data points for the systematic study of connectivity-preservation constraints in the parameterized setting.

1998 ACM Subject Classification G.2.2 Graph algorithms

Keywords and phrases connectivity, strong connectivity, vertex deletion, arc contraction

Digital Object Identifier 10.4230/LIPIcs.ESA.2017.47

1 Introduction

Some of the most well studied classes of network design problems involve starting with a given network and making modifications to it so that the resulting network satisfies certain connectivity requirements, for instance a prescribed edge- or vertex-connectivity. This class of problems has a long and rich history (see *e.g.* [1, 8]) and has recently started to be examined through the lens of parameterized complexity. Under this paradigm, we ask whether a (hard) problem admits an algorithm with a running time $f(k)n^{O(1)}$, where n is the size of the input,

* The research of Gregory Gutin was partially supported by Royal Society Wolfson Research Merit Award. M. S. Ramanujan acknowledges support from Austrian Science Fund (FWF, project P26696). Felix Reidl and Magnus Wahlström were supported by EPSRC grant EP/P007228/1.
† A full version of the paper is available at `https://arxiv.org/abs/1704.06622`.

k the *parameter*, and f some computable function. A natural parameter to consider in this context is the number of editing operations allowed and we can reasonably assume that this number is small compared to the size of the graph.

To approach this line of research systematically, let us identify the 'moving parts' of the broader question of editing under connectivity constraints: first and foremost, the network in question might best be modelled as either a directed or undirected graph, potentially with edge- or vertex-weights. This, in turn, informs the type of connectivity we restrict, *e.g.* strong connectivity or fixed value of edge-/vertex-connectivity. Additionally, the connectivity requirement might be non-uniform, *i.e.* it might be specified for individual vertex-pairs. The constraint one operates under might either be to *preserve*, to *augment*, or to *decrease* said connectivity. Finally, we need to fix a suitable editing operation; besides the obvious vertex- and edge-removal, more intricate operations like edge contractions are possible.

While not all possible combinations of these factors might result in a problem that currently has an immediate real-world application, they are nonetheless important data points in the systematic study of algorithmic tractability. For example, if we fix the editing operation to be the addition of edges (often called 'links' in this context) and our goal is to increase connectivity, then the resulting class of *connectivity augmentation problems* has been thoroughly researched. We refer to the monograph by Frank [8] for further results on polynomial-time solvable cases and approximation algorithms. Under the parameterized complexity paradigm, Nagamochi [16] and Guo and Uhlmann [10] studied the problem of augmenting a 1-edge- connected graph with k links to a 2-edge-connected graph. Nagamochi obtained an FPT algorithm for this problem while Guo and Uhlmann showed that this problem, alongside its vertex-connectivity variant, admits a quadratic kernel. Marx and Végh [14] studied the more general problem of augmenting the edge-connectivity of an undirected graph from $\lambda - 1$ to λ, via a minimum set of links that has a total *cost* of at most k, and obtained an FPT algorithm as well as a polynomial kernel for this problem. Basavaraju *et al.* [3] improved the running time of their algorithm and further showed the fixed-parameter tractability of a dual parameterization of this problem.

A second large body of work can be found in the antithetical class of problems, where we ask to *delete* edges from a network while *preserving* connectivity. Probably the most studied member of these *connectivity preservation problems* is the MINIMUM STRONG SPANNING SPANNING SUBGRAPH (MSSS) problem: given a strongly connected digraph we are asked to find a strongly connected subgraph with a minimum number of arcs. The problem is NP-complete (an easy reduction from the HAMILTONIAN CYCLE problem) and there exist a number of approximation algorithms for it (see the monograph by Bang-Jensen and Gutin for details and references [1]). Bang-Jensen and Yeo [2] were the first to study MSSS from the parameterized complexity perspective. They presented an algorithm that runs in time $2^{O(k \log k)} n^{O(1)}$ and decides whether a given strongly connected digraph D on n vertices and m arcs has a strongly connected subgraph with at most $m - k$ arcs provided $m \geqslant 2n - 2$. Basavaraju *et al.* [4] extended this result not only to arbitrary number m of arcs but also to λ-arc-strong connectivity for an arbitrary integer λ, and they further extended it to λ-edge-connected *undirected* graphs.

We consider the undirected variant of this problem, however, we aim to preserve the *vertex-connectivity* instead of edge-connectivity. As noted by Marx and Végh [14], vertex-connectivity variants of parameterized connectivity problems seem to be much harder to approach than their edge-connectivity counterparts.[1] Moreover, even the complexity of the

[1] Marx and Végh [14] compare [18] and [7] to [9] and [17] with respect to polynomial-time exact and

problem of augmenting the vertex-connectivity of an undirected graph from 2 to 3, via a minimum set of up to k new links remains open [14]. Our main result in this direction is the first FPT algorithm for the following problem[2]:

WEIGHTED BICONNECTIVITY DELETION parametrised by k

Input: A biconnected graph G, $k \in \mathbb{N}$, $w^* \in \mathbb{R}_{\geqslant 0}$ and a function $w : E(G) \to \mathbb{R}_{\geqslant 0}$.
Problem: Is there a set $S \subseteq E(G)$ of size at most k such that $G - S$ is biconnected and $w(S) \geqslant w^*$?

▶ **Theorem 1.** WEIGHTED BICONNECTIVITY DELETION *can be solved in time* $2^{O(k \log k)} n^{O(1)}$.

We further show that this problem has a randomized polynomial kernelization when the edges are required to have only unit weights. To be precise, all inputs for the unweighted variant UNWEIGHTED BICONNECTIVITY DELETION (UBD) are of the form (G, k, w^*, w), where $w^* = k$ and $w(e) = 1$ for every $e \in E(G)$.

▶ **Theorem 2.** UBD *has a randomized kernel with* $O(k^9)$ *vertices.*

Along with arc-additions and arc-deletions, a third interesting operation on digraphs is the *path-contraction* operation which has been used to obtain structural results on paths in digraphs [1]. To *path-contract* an arc (x, y) in a digraph D, we remove it from D, identify x and y and keep the in-arcs of x and the out-arcs of y for the combined vertex. The resulting digraph is denoted by $D \mathbin{/\!\!/} (x, y)$. It is useful to extend this notation to sequences of contractions: let $S = (a_1, a_2, \dots, a_p)$ be a sequence of arcs of a digraph D. Then $D \mathbin{/\!\!/} S$ is defined as $(\dots((D \mathbin{/\!\!/} a_1) \mathbin{/\!\!/} a_2) \mathbin{/\!\!/} \dots) \mathbin{/\!\!/} a_p$. Since the resulting digraph does not depend on the order of the arcs [1], this notation can equivalently be used for arc-sets.

Bang-Jensen and Yeo [2] asked whether the problem of path-contracting at least k arcs to maintain strong connectivity of a given digraph D is fixed-parameter tractable. Formally, the problem is stated as follows:

PATH-CONTRACTION PRESERVING STRONG CONNECTIVITY parametrised by k

Input: A strongly connected digraph D and an integer k.
Problem: Is there a sequence $S = (a_1, \dots, a_k)$ of arcs of D such that $D \mathbin{/\!\!/} S$ is also strongly connected?

Our first result is a negative answer to the question of Bang-Jensen and Yeo. That is, we show that this problem is unlikely to be FPT.

▶ **Theorem 3.** PATH-CONTRACTION PRESERVING STRONG CONNECTIVITY *is W[1]-hard.*

We follow up this result by considering a natural vertex-deletion variant of the problem and extending our W[1]-hardness result to this problem as well. In this variant, the objective is to check for the existence of a set of *exactly*[3] k vertices such that on deleting these vertices from the given digraph, the digraph stays strongly connected.

▶ **Theorem 4.** VERTEX-DELETION PRESERVING STRONG CONNECTIVITY *is W[1]-hard.*

approximation algorithms.

[2] Note that since 1-vertex-connectivity is trivially equivalent to 1-edge-connectivity, the 1-vertex-connectivity case was proved to be FPT by Basavaraju *et al.* [4].

[3] We require *exactly* k vertices rather than *at least* k vertices to be deleted since the one-vertex digraph is strongly connected [1].

Our Methodology. Our algorithm for Weighted Biconnectivity Deletion builds upon the recent approach introduced by Basavaraju *et al.* [4] to handle connectivity preservation problems, in particular the p-λ-Edge Connected Subgraph (p-λ-ECS) problem where the objective is to delete k edges while keeping the graph λ-edge connected. Call an edge *deletable* (we refer to it as *non-critical* in the case of vertex-connectivity) if deleting it keeps the given (di)graph λ-edge connected, *undeletable* (*critical*) otherwise, and call an edge *irrelevant* if there is a solution disjoint from the edge.

For an even value of λ and a λ-edge-connected undirected graph G, Basavaraju *et al.* [4] proved that unless the total number of deletable edges is bounded by $O(\lambda k^2)$, it is possible in polynomial time to obtain a set F of k edges such that $G - F$ is still λ-edge-connected. This result does not hold for odd values of λ as can be seen, *e.g.*, when $\lambda = 1$ and G is a cycle. In this much more involved case, unless the total number of deletable edges is bounded by $O(\lambda k^3)$, it is possible in polynomial time to obtain either a set F of k edges such that $G - F$ is still λ-edge-connected or to identify an irrelevant edge.

Weighted Biconnectivity Deletion is similar to the case of odd λ as we find either a solution or an irrelevant edge. The main difference between our FPT algorithm and the one presented by Basavaraju *et al.* is the deep structural analysis necessitated by the shift from edge-connectivity to vertex-connectivity: While in the former case the failure to find a solution means that G can be decomposed into a 'cycle-like' structure as shown in [4], in our case no such simple structure arises. Instead, we perform a careful examination of mixed cuts in the graph, each of which comprises precisely one critical edge e and a vertex w which we call the *partner* of e. We show that either a large number of critical edges share a common partner or there is a large number of critical edges with pairwise distinct partners. In the former case, we prove the existence of an irrelevant edge while in the latter case we are able to construct a solution. Our result is based on a non-trivial combination of several new structural properties of biconnected graphs and critical edges which we believe is of independent interest and useful in the study of other connectivity-constrained problems.

The kernel stated in Theorem 2 relies on the powerful *cut-covering lemma* of Kratsch and Wahlström [13] which has been central to the development of several recent kernelization algorithms [12]. While Basavaraju *et al.* obtained a randomized compression for the p-λ-ECS problem using sketching techniques from dynamic graph algorithms, we provide an alternative approach and show that when dealing with biconnectivity it is also possible to obtain a (randomized) polynomial *kernel*. We believe that this approach could be applicable for higher values of vertex- connectivity and for other connectivity deletion problems, as long as one is able to bound the number of critical or undeletable edges in the given instance by an appropriate function of the parameter.

Further related work. In the Minimum Equivalent Digraph problem, given a digraph D, the aim is to find a spanning subgraph H of D with minimum number of arcs such that if there is an x-y directed path in D then there is such a path in H for every pair x, y of vertices of D. Since it is not hard to solve Minimum Equivalent Digraph for acyclic digraphs, Minimum Equivalent Digraph for general digraphs can be reduced to MSSS in polynomial time. Chapter 12 of the monograph of Bang- Jensen and Gutin [1] surveys pre-2009 results on Minimum Equivalent Digraph. The first exact algorithm for the Mnimum Equivalent Digraph problem, running in time $2^{O(m)}$, was given by Moyles and Thompson [15] in 1969, where m is the number of arcs in the graph. More recently, Fomin, Lokshtanov, and Saurabh [6] gave the first vertex-exponential algorithm for this problem, *i.e.* an algorithm with a running time of $2^{O(n)}$.

Paper organization. This paper is a shortened version of the full paper [11]. Due to the space limit, we omitted several results, proofs, and other material.

2 Preliminaries

Graphs. For an undirected graph G and vertex set $S \subseteq V(G)$, we denote by $E(S)$ the set of edges of G with both endpoints in S. For a vertex set $X \subseteq V(G)$, we denote by $N_G(X)$ the set of vertices of $V(G) \setminus X$ which are adjacent to a vertex in X. A vertex in a connected undirected graph is a *cut-vertex* if deleting this vertex disconnects the graph. A *biconnected graph* is a connected graph on two or more vertices having no cut-vertices.

▶ **Definition 5.** Let G be a graph and $x, y \in V(G)$ two vertices. An *x-y separator* (an *x-y cut*) is a set $S \subseteq V(G) \setminus \{x, y\}$ (respectively $S \subseteq E(G)$) such that there is no x-y path in $G - S$. A *mixed x-y cut* is a set $S \subseteq V(G) \cup E(G)$ such that $|S \cap E(G)| = 1$ and there is no x-y path in $G - S$.

▶ **Definition 6.** Let G be a graph and $x, y \in V(G)$. Let $\mathcal{P}$ be a set of internally vertex-disjoint x-y paths in G. Then, we call $\mathcal{P}$ an *x-y flow.* The *value* of this flow is $|\mathcal{P}|$. We say that an edge e *participates* in the x-y flow $\mathcal{P}$ if $e \in \bigcup_{P \in \mathcal{P}} P$.

We denote by $\kappa_G(x, y)$ the value of the maximum x-y flow in G with the reference to G dropped when clear from the context.

Let $\mathcal{P}$ be a set of paths in G which have an endpoint in Y and intersect only in x. Then, we refer to $\mathcal{P}$ as an x-Y flow, with the value of this flow defined as $|\mathcal{P}|$.

Directed graphs. We will refer to edges in a digraph as *arcs*. For a vertex x in a digraph D we write $N_D^-(x)$ and $N_D^+(x)$ to denote its in- and out-neighbours, respectively. A *sink* is a vertex with no out-neighbours and a *source* is a vertex with no in-neighbours.

Parameterized Complexity. An instance of a parameterized problem Π is a pair (I, k) where I is the *main part* and k is the *parameter*; the latter is usually a non-negative integer. A parameterized problem is *fixed-parameter tractable* if there exists a computable function f such that instances (I, k) can be solved in time $O(f(k)|I|^c)$ where $|I|$ denotes the size of I. The class of all fixed-parameter tractable decision problems is called FPT and algorithms which run in the time specified above are called FPT algorithms.

To establish that a problem under a specific parameterization is not in FPT (under common complexity-theoretic assumptions) we provide *parameter-preserving reductions* from problems known to lie in intractable classes like W[1] or W[2]. In such a reduction, an instance (I_1, k_1) is reduced in FPT time to an instance (I_2, k_2) where $k_2 \leqslant f(k_1)$ for some function f. In the context of this paper we will use that Independent Set under its natural parameterization (the size of the independent set) is W[1]-hard [5].

A *reduction rule* for a parameterized problem Π is an algorithm that given an instance (I, k) of a problem Π returns an instance (I', k') of the *same* problem. The reduction rule is said to be *sound* if it holds that $(I, k) \in \Pi$ if and only if $(I', k') \in \Pi$. A *kernelization* is a polynomial-time algorithm that given any instance (I, k) returns an instance (I', k') such that $(I, k) \in \Pi$ if and only if $(I', k') \in \Pi$ *and* $|I'| + k' \leqslant f(k)$ for some computable function f. The function f is called the *size* of the kernelization, and we have a polynomial kernelization if $f(k)$ is polynomially bounded in k. A *randomized kernelization* is an algorithm which is allowed to err with certain probability. That is, the returned instance will be equivalent to the input instance only with a certain probability.

3 Preserving strong connectivity

In this section, we prove Theorem 3.

▶ **Theorem 3.** Path-contraction Preserving Strong Connectivity *is W[1]-hard.*

Proof. We reduce Independent Set to Path-contraction Preserving Strong Connectivity.

Construction. Let (G, k) be an instance of Independent Set. We now define a digraph D as follows. We begin with the vertex set of D. For every vertex $v \in V(G)$, D has two vertices v^-, v^+. For every edge $e = (u, v) \in E(G)$, the digraph D has $k+2$ vertices $\hat{e}, \hat{e}_1, \ldots, \hat{e}_{k+1}$. Finally, there are $2k+4$ special vertices $x, y, x^1, \ldots, x^{k+1}, y^1, \ldots, y^{k+1}$. This completes the definition of $V(D)$. We now define the arc set of D (see Figure 1).

- For every $v \in V(G)$, we add the arc (v^-, v^+) in D .
- For every $i \in [k+1]$, we add the arcs $\{(x, x^i), (x^i, x), (y, y^i), (y^i, y), (y, x)\}$.
- For every edge $e = (u, v) \in E(G)$ and $i \in [k+1]$, we add the arcs $\{(\hat{e}, \hat{e}_i), (\hat{e}_i, \hat{e}), (v^-, \hat{e}), (\hat{e}, v^+), (u^-, \hat{e}), (\hat{e}, u^+)\}$ in D .
- For every $v \in V(G)$, we add the arc (x, v^-) and the arc (v^+, y).

This completes the construction of the digraph D. Clearly, D is strongly-connected.

For an edge $e = (u, v) \in E(G)$, we denote by $\mathcal{B}_e$ the set of arcs $\{(v^-, \hat{e}), (\hat{e}, v^+), (u^-, \hat{e}), (\hat{e}, u^+)\}$ and by $\mathcal{F}_e$, the set of arcs

$$\mathcal{B}_e \cup \{(\hat{e}, \hat{e}_i), (\hat{e}_i, \hat{e}) | i \in [k+1]\} \cup \{(u^-, u^+), (v^-, v^+), (x, v^-), (v^+, y), (x, u^-), (u^+, y), (y, x)\}.$$

We refer to the subgraph of D induced by $\mathcal{F}_e$ as the *edge-selection gadget* in D corresponding to e (see Figure 1). The intuition here is that, as we will prove formally, any solution in D will contain at most one of the two arcs $(u^-, u^+), (v^-, v^+)$.

Proof of correctness. We now argue that (G, k) is a yes-instance of Independent Set if and only if (D, k) is a yes-instance of Path-contraction Preserving Strong Connectivity. In the forward direction, suppose that (G, k) is a yes-instance of Independent Set and let $X \subseteq V(G)$ be a solution. Observe that $S = \{(v^-, v^+) \mid v \in X\}$ is a pairwise vertex-disjoint set of arcs. We claim that S is a solution for the instance (D, k). That is, $|S| \geqslant k$ and $D /\!\!/ S$ is strongly connected. The former is true by definition. We prove the latter, as the claim below, in the full version of the paper.

▶ **Claim 7.** $D' = D /\!\!/ S$ *is strongly connected.*

We now consider the converse direction. Suppose that (D, k) is a yes-instance of Path-contraction Preserving Strong Connectivity and let $S = \{a_1, \ldots, a_k\}$ be a solution for this instance. We require the following claim whose proof can be found in the full version of the paper.

▶ **Claim 8.** *For every edge* $e = (u, v) \in E(G)$, $|S \cap \{(u^-, u^+), (v^-, v^+)\}| \leqslant 1$. *Furthermore,* $S \subseteq \{(v^-, v^+) \mid v \in V(G)\}$.

The claim above implies that if X is a solution for the reduced instance of Path-contraction Preserving Strong Connectivity, then the set S of arcs corresponds independent set in G. In other words, (G, k) is a yes-instance of Independent Set. This proves the correctness of the reduction and completes the proof of the theorem. ◀

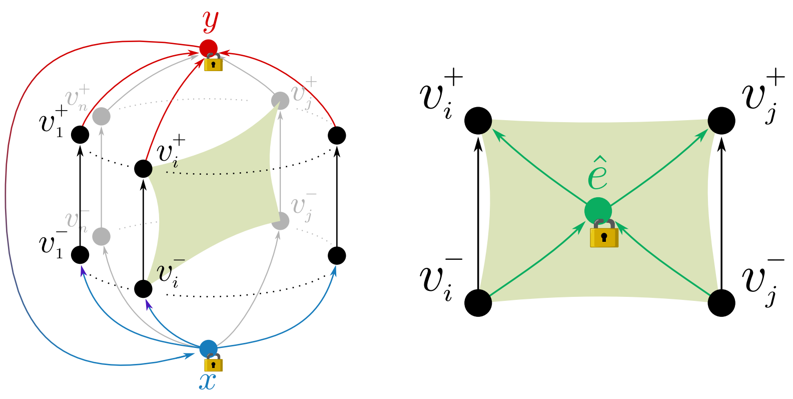

Figure 1 An illustration of the arcs in the reduced instance of Path-contraction Preserving Strong Connectivity. The second figure only contains the arcs of the edge-selection gadget corresponding to the edge $e = (v_i, v_j) \in E(G)$. Vertices with a padlock have additional $k+1$ pendant vertices with arcs in both directions.

4 Edge deletion to biconnected graphs

In this section, we consider the Weighted Biconnectivity Deletion problem on undirected graphs. Recall that the problem is defined as follows:

Weighted Biconnectivity Deletion parametrised by k

Input:	A biconnected graph G, $k \in \mathbb{N}$, $w^* \in \mathbb{R}_{\geqslant 0}$ and a function $w : E(G) \to \mathbb{R}_{\geqslant 0}$.
Problem:	Is there a set $S \subseteq E(G)$ of size at most k such that $G - S$ is biconnected and $w(S) \geqslant w^*$?

We refer to a set $S \subseteq E(G)$ such that $G - S$ is biconnected as a *biconnectivity deletion set* of G. For an instance (G, k, w^*, w) of Weighted Biconnectivity Deletion and a biconnectivity deletion set S of G, we say that S is a *solution* if $|S| \leqslant k$ and $w(S) \geqslant w^*$. The main result of this section is the following.

▶ **Theorem 1.** Weighted Biconnectivity Deletion *can be solved in time* $2^{O(k \log k)} n^{O(1)}$.

We denote by $\kappa(G)$ the vertex-connectivity of a graph G. Let G be a biconnected graph. An edge $e \in E(G)$ is called *critical* if $\kappa(G - e) < 2$. We denote by $\mathsf{Critical}_\mathsf{G}(\mathsf{e})$ the subset of $E(G)$ comprising edges which are critical in $G - e$ but not in G. We denote by $\mathsf{Critical}_G(\emptyset)$ the set of edges which are already critical in G. In all notations, we ignore the explicit reference to G when it is clear from the context. We say that e is *critical for* a pair of vertices u, v in G if u and v are non-adjacent and e participates in every u-v flow of value two in G.

4.1 The FPT algorithm for Weighted Biconnectivity Deletion

To prove Theorem 1 we consider a more general version of the Weighted Biconnectivity Deletion problem where the input also includes a set $E^\infty \subseteq E(G)$ and the objective is to decide whether there is a solution disjoint from this set. Henceforth, instances of Weighted Biconnectivity Deletion will be of the form (G, k, w^*, w, E^∞) and any solution S is required to be disjoint from E^∞. We will refer to edges of $E(G) \setminus E^\infty$ as *potential solution edges*. We say that a potential solution edge e is *irrelevant* if either the instance has no solution, or has a solution that does not contain e. For an instance $I = (G, k, w^*, w, E^\infty)$ and $r \in \mathbb{N}$, we denote by $\mathsf{Heavy}_I(r)$ the *heaviest* r potential solution edges of G with respect to the function w. If I is clear from the context, we simply write $\mathsf{Heavy}(r)$ when referring to $\mathsf{Heavy}_I(r)$.

Observe that no edge from the set $\mathsf{Critical}_G(\emptyset)$ can be part of a solution. As a result, we assume without loss of generality that for any instance (G, k, w^*, w, E^∞), the set $\mathsf{Critical}_G(\emptyset)$ is contained in E^∞. Furthermore, since the edges in E^∞ can never be part of a solution, we assume without loss of generality that for every edge $e \in E^\infty$, $w(e) = 0$. The proof of Theorem 1 is based on the following lemma which states that either a) the number of potential solution edges in the instance is already bounded polynomially in k, or b) a 'small' set of the heaviest edges in the instance must intersect a solution, or c) there is an irrelevant edge which can be found in polynomial time. For ease of presentation, let use define the polynomial $\mu(x) := 20x^3 + 46x^2 + x$ for the rest of this section.

▶ **Lemma 9.** *Let $I = (G, k, w^*, w, E^\infty)$ be an instance of* Weighted Biconnectivity Deletion. *If $|E(G) \setminus E^\infty| > \mu(k)$, then the set $\mathsf{Heavy}(\mu(k))$ contains either a solution edge or an irrelevant edge which can be computed in polynomial time.*

Proof. We give a brief proof sketch for the lemma. Note that the individual claims below do not map directly to claims in the full paper, but are present to illustrate the important ideas.

The general strategy of our result, following Basavaraju *et al.* [4], is a greedy algorithm that iteratively selects a non-critical edge $e \in \mathsf{Heavy}(\mu(k))$ for inclusion into a solution, computes which other edges become critical by the deletion of e, and either proceeds to select another edge for inclusion or halt, if either all edges of the remaining graph are critical or if the solution already contains k edges. Let us first cover the latter case.

▶ **Claim 10.** *If there exists a biconnectivity deletion set $S \subseteq \mathsf{Heavy}(\mu(k))$ with $|S| = k$, then any solution to the instance must intersect $\mathsf{Heavy}(\mu(k))$.*

Recall that this is one of the positive outcomes of Lemma 9. Therefore, we henceforth assume that there are more than $\mu(k)$ potential solution edges, but that the greedy algorithm terminates after fewer than k steps due to all edges of the remaining graph being critical. Let $f_1, \ldots, f_t$, $t < k$ be the sequence of edges selected by the greedy algorithm. By a pigeonhole argument, there must then be some index $r \in [t]$ such that $|\mathsf{Critical}_{G-\{f_1,\ldots,f_{r-1}\}}(f_r)| > 20k^2 + 46k$. Let $S = \{f_1, \ldots, f_{r-1}\}$ be the edges deleted until this point, let $e = f_r = (x, y)$, and let $G' = G - S$. We proceed to analyse the structure implied by the edges in $\mathsf{Critical}_{G'}(e)$.

▶ **Claim 11.** *The following hold.*
1. *The maximum value of an x-y flow in G' is 2.*
2. *For any x-y flow $\mathcal{P} = \{P_1, P_2\}$ in G', every edge of $\mathsf{Critical}_{G'}(e)$ participates in $\mathcal{P}$.*
3. *For every edge $e' \in \mathsf{Critical}_{G'}(e)$, say $e' \in E(P_1)$, there is a mixed x-y cut $\{e', w\}$ in $G' - e$, and for every such mixed cut we have $w \in V(P_2)$.*

In the latter case, we refer to w as a *partner vertex* of e', and to the set of all partner vertices of e' as the *partner set* of e'. The crux of the remainder of the proof lies in analysing the interaction between different mixed cuts and partner sets in G'. For convenience, we fix an order on P_1 and P_2 where we traverse both paths from x to y. We denote $\hat{E} = \mathsf{Critical}_{G'}(e) \cap E(P_1)$, and assume without loss of generality that $|\hat{E}| > 10k^2 + 23k$. The following is the most important structural observation on which our proof is based. For each edge $e_i \in \hat{E}$, let V_i denote the partner set of e_i.

▶ **Claim 12.** *For every pair of edges e_i, e_j in $\hat{E}$, where e_i lies before e_j on P_1, $|V_i \cap V_j| \leqslant 1$, and if $V_i \cap V_j$ contains such a common vertex w, then w is the last vertex of V_i and the first vertex of V_j on P_2.*

Our proof now splits into two fundamentally different cases. Either the flow $\mathcal{P}$ contains a long sequence of pairwise essentially non-interacting mixed cuts, or there are many edges of $\hat{E}$ with pairwise identical partner sets. We cover the first case now.

▶ **Claim 13.** *Let $e_1, \ldots, e_{3k+1}$ be a sequence of edges of $\hat{E}$, traversed in this order from x to y, such that for every $i \in [3k]$ the edges e_i and e_{i+1} have distinct partner sets. Then the set $F = \{e_1, e_4, \ldots, e_{3k-2}\}$ is a biconnectivity deletion set for G.*

By Claim 10, this would imply Lemma 9, so it remains to consider the case when Claim 13 fails to apply. In fact, Claim 13 applies whenever there is a sufficiently large number of distinct partner sets for edges of $\hat{E}$; hence we may assume that there is a large number of distinct edges of $\hat{E}$ with identical partner sets. Let $\hat{E}' \subseteq \hat{E}$ be a set of $\Omega(k)$ edges with pairwise identical partner sets. Then by Claim 12, there is a single vertex $w \in V(P_2)$ such that for any $e' \in \hat{E}'$ the partner set of e' is simply $\{w\}$. We show that this case implies an irrelevant edge rule. For simplicity, we illustrate the rule for the case that $S = \emptyset$.

▶ **Claim 14.** *Assume that $G = G'$ and $|\hat{E}'| \geqslant 2k + 4$. Let $e' = \arg\min_{e' \in \hat{E}'} w(e')$. Then for any biconnectivity deletion set S' of size k in G we have $|S' \cap \hat{E}'| \leqslant 1$, and if $e' \in S$ then there exists an edge $e'' \in \hat{E}' \setminus \{e'\}$ such that the set $S'' = (S' \setminus \{e'\}) \cup \{e''\}$ is a biconnectivity deletion set with $|S''| = k$ and $w(S'') \geqslant w(S')$. Hence e' is an irrelevant edge in G.*

By additional arguments omitted from this sketch, a similar result also holds for the general case of $S \neq \emptyset$, and under the assumption that $|\hat{E}| > 10k^2 + 23k$ we can show that either Claim 13 or Claim 14 applies. Hence in every case we find either an irrelevant edge or a large biconnectivity deletion set, and Lemma 9 follows. ◀

Given Lemma 9, Theorem 1 is proved as follows. Let $I = (G, k, w, w^*, E^\infty)$ be an instance of Weighted Biconnectivity Deletion. If the number of potential solution edges in this instance is already bounded by $\mu(k)$, then we simply enumerate all k-sized subsets of this set (there are $2^{O(k \log k)}$ choices) and check in polynomial time whether one of these subsets is a solution. Otherwise, we invoke Lemma 9 and either correctly conclude that the set $\mathsf{Heavy}(\mu(k))$ contains a solution edge, or we compute an irrelevant edge e in polynomial time. In the first case we branch on the set $\mathsf{Heavy}(\mu(k))$, reduce the budget k by 1 and the target weight w^* accordingly and recursively solve the resulting instance. In the second case, we add the edge e to the set E^∞ (thus decreasing the set of potential solution edges) and repeat.

4.2 A randomized kernel for Unweighted Biconnectivity Deletion

We now present our randomized kernel for the Weighted Biconnectivity Deletion problem where instances are of the form (G, k, w^*, w, E^∞) where $w(e) = 1$ for every $e \in E(G) \setminus E^\infty$, $w(e) = 0$ for every $e \in E^\infty$, and $w^* = k$. This version of the problem will be referred to as Unweighted Biconnectivity Deletion and instances of this problem will henceforth be of the form (G, k, E^∞) where a solution is a biconnectivity deletion set of size k contained in $E(G) \setminus E^\infty$. We continue to refer to the set $E(G) \setminus E^\infty$ as the set of potential solution edges and assume without loss of generality that at any point, any edge in the set $\mathsf{Critical}_G(\emptyset)$ is already part of E^∞. Finally, recall that a *linkage* from A to B in a digraph

D, where A and B are vertex sets, is a collection of $|A| = |B|$ pairwise vertex-disjoint paths originating in A and terminating in B.

Our kernelization relies on a result of Kratsch and Wahlström [13]. Before we are able to state it formally, we need the following definitions. Let us define a *potentially overlapping A-B vertex cut* in a digraph D to be a set of vertices $C \subseteq V(D)$ such that $D - C$ contains no directed path from $A \setminus C$ to $B \setminus C$. For any digraph D and set $X \subseteq V(D)$, a set $Z \subseteq V(D)$ is called a *cut-covering set* for (D, X) if for any $A, B, R \subseteq X$, there is a minimum-cardinality potentially overlapping A-B vertex cut C in $D - R$ such that $C \subseteq Z$.

▶ **Lemma 15** (Corollary 3, [13]). *Let D be a directed graph and let $X \subseteq V(D)$. We can identify a cut-covering set Z for (D, X) of size $O(|X|^3)$ in polynomial time with failure probability $O(2^{-|V(D)|})$.*

We first give a randomized kernelization that outputs an instance whose size is bounded polynomially in the number of the potential solution edges in the input instance.

▶ **Lemma 16.** Unweighted Biconnectivity Deletion *has a randomized kernel with number of vertices bounded by $O(|E(G) \setminus E^\infty|^3)$.*

Proof. Let $F = E(G) \setminus E^\infty$ be the set of potential solution edges. Now, the kernelization task essentially consists of retaining enough information from the input graph G to verify for any set $S \subseteq F$, whether S is a biconnectivity deletion set for G. Observe that this is equivalent to verifying whether there exists an edge $e = (u, v) \in S$, such that the maximum value of a u-v flow in $G - S$ is less than 2. We show an equivalent formulation of this as a question about the existence of linkages in an auxiliary digraph, followed by an application of Lemma 15.

For the formulation, we create a digraph $D_{G,F}$ from G and F. We refer to this digraph as D when G and F are clear from the context. In the first step, subdivide every edge $e \in F$ with a new vertex x_e. That is, for an edge $e = (u, v) \in F$, we create a new vertex x_e, remove the edge e and add edges (u, x_e) and (v, x_e). Let G_1 be the resulting undirected graph. In the second step, replace every edge (u, v) in $E(G_1)$ by a pair of arcs (u, v), (v, u). Finally, for every vertex v incident to any edge of F in G, add vertices v^+, v^- and add arcs from v^+ to all vertices in $N_{G_1}(v)$ and from all vertices in $N_{G_1}(v)$ to v^-. Let D be the resulting digraph. Note that $N^+_D(v^-) = \emptyset$ and $N^-_D(v^+) = \emptyset$. Let $X_E = \{x_e \mid e \in F\}$, $X_V = \{v^+, v^-, v \mid e \in F, e = (u, v)\}$ and $X = X_E \cup X_V$. We now relate solutions for the given instance and linkages in D. The following assertion is proved in the full version of the paper.

▶ **Claim 17.** *For any $S \subseteq F$, S is a biconnectivity deletion set for G if and only if for every edge $(u, v) \in S$ there is a linkage from $\{u^+, u\}$ to $\{v^-, v\}$ in $D - \{x_e \mid e \in S\}$.*

Let $Z \subseteq V(D)$ be the cut-covering set for (D, X), as computed by the algorithm of Lemma 15. Having in hand the set Z, we define the set $Y = (Z \cap V(G)) \cup V(F)$. Note that Z could contain vertices from X_V, but we want Y to be a subset of $V(G)$. Therefore, we first add to Y those vertices in Z which are also vertices in G and then add the vertices of $V(F)$. Our objective now is to reduce G down to what is commonly known as the *torso* graph of G defined by Y (see [13]). We now make this precise in the form of reduction rules. In the rest of the proof of the lemma, we fix Z to be a set computed using Lemma 15 and let Y be as defined above. We now state three reduction rules which will be applied on the given instance in the order in which they are presented.

▶ **Reduction Rule 18.** *If $k = 0$, then return an arbitrary yes-instance of constant size.*

▶ **Reduction Rule 19.** *Suppose that Reduction Rule 18 has been applied on the given instance. If there is an edge $(u,v) \in F$ such that G contains a u-v path avoiding all edges of F and all vertices of $Y \setminus \{u,v\}$, then delete (u,v) from G and reduce the budget k by 1. That is, return the instance $(G - \{(u,v)\}, k-1, E^\infty)$.*

▶ **Reduction Rule 20.** *Suppose that Reduction Rule 18 and Reduction Rule 19 have been applied exhaustively on the given instance. For every pair $u, v \in Y$ such that $(u,v) \notin E(G)$ and there is a u-v-path in G that is internally vertex-disjoint from Y, we add the edge (u,v). Finally, return the instance (G', k, E'^∞), where $G' = G[Y]$ and $E'^\infty = (E^\infty \cap E(G')) \cup (E(G') \setminus E(G))$.*

The soundness of Rule 18 is trivial and we move on to prove the soundness of the remaining two rules.

▶ **Claim 21.** *Reduction Rules 19 and 20 are sound.*

Proof. Let $e = (p,q) \in F$ be an edge which is deleted in an application of Reduction Rule 19. Observe that in order to argue the soundness of this reduction rule, it suffices to argue that e is part of some solution for the given instance (if there exist any). Let S be an arbitrary subset of F containing e such that $S \setminus \{e\}$ is a solution. If S itself is a biconnectivity deletion set then we may correctly conclude that e is part of some solution for the given instance. Suppose that this is not the case.

Recall that by the previous claim, S is a biconnectivity deletion set for G if and only if there is a linkage from $\{u^+, u\}$ to $\{v^-, v\}$ in $D - \{x_e \mid e \in S\}$ for every $(u,v) \in S$. Since we are in the case that S is *not* a biconnectivity deletion set, there is a $(u,v) \in S$, with $A = \{u^+, u\}$, $B = \{v^-, v\}$, and $R = \{x_e \mid e \in S\}$ such that there is no linkage from A to B in $D - R$. Since $S \setminus \{e\}$ is a biconnectivity deletion set, we may assume without loss of generality that $u = p$ and $v = q$ and furthermore, $\kappa_{G-S}(p,q) = 1$. In addition, the fact that Z is a cut-covering set for (D, X) implies that Z contains a vertex w such that $C = \{w\}$ is a minimum-cardinality potentially overlapping A-B vertex cut in $D - R$. It is straightforward to see that $w \notin \{p, q, p^+, q^-\}$ since otherwise, there will be at least one path from A to B which is disjoint from w. Finally, since $\kappa_{G-S}(p,q) = 1$, it follows that every p-q path in $G - S$ intersects w. If $w \in X_E$ then we know that it corresponds to an edge in F. Otherwise, it corresponds to a vertex in Y. In either case, we obtain a contradiction to the applicability of Reduction Rule 19 on the edge (p,q), completing the proof of soundness for this rule.

We now argue the soundness of Reduction Rule 20. To do so, we prove that $S \subseteq F$ is a solution for (G, k, E^∞) if and only if it is a solution for (G', k, E'^∞). Let $D_1 = D_{G,F}$ and let $D_2 = D_{G',F}$.

In the forward direction, suppose that S is a solution for (G, k, E^∞). By Claim, 17, it follows that for every edge $(u,v) \in S$, there is a linkage from $\{u^+, u\}$ to $\{v^-, v\}$ in $D_1 - \{x_e \mid e \in S\}$. Fix such an edge (u,v) and let the paths in the linkage be P_1, P_2. If we demonstrate such a linkage in D_2, then we are done. This can be achieved as follows. Let $i \in \{1,2\}$ and consider a pair of vertices $x_i, y_i \in V(P_i) \cap Y$ such that the subpath of P_i from x_i to y_i has all its internal vertices disjoint from Y. Then, we know that the graph G' contains the edge (x_i, y_i) and hence the digraph D_2 contains the arc (x_i, y_i). We replace the subpath from x_i to y_i with the arc (x_i, y_i) and we do this for every such subpath of P_i. It is straightforward to see that what results is indeed a linkage from $\{u^+, u\}$ to $\{v^-, v\}$ in $D_2 - \{x_e \mid e \in S\}$. Hence, we conclude that S is a solution for (G', k, E'^∞).

The same argument can be reversed for the converse direction in order to convert, for any $(u,v) \in S$, a linkage from $\{u^+, u\}$ to $\{v^-, v\}$ in $D_2 - \{x_e \mid e \in S\}$ to a a linkage from

$\{u^+, u\}$ to $\{v^-, v\}$ in $D_1 - \{x_e \mid e \in S\}$. This completes the proof of soundness of Reduction Rule 20. ◀

The above claim implies that if $(G', k', E(G') \setminus F')$ is the instance obtained by exhaustively applying the three reduction rules above, then $(G', k', E(G') \setminus F')$ is indeed equivalent to (G, k, E^∞). Furthermore, the size $|V(G')| = O(|F|^3)$ and the randomized polynomial running time follow from Lemma 15. This completes the proof of the lemma. ◀

▶ **Theorem 2.** UBD *has a randomized kernel with* $O(k^9)$ *vertices.*

Proof sketch. Let (G, k, E^∞) be the given instance and let $F = E(G) \setminus E^\infty$ be the set of potential solution edges in this instance. We present reduction rules which reduce F (while maintaining equivalence) to size $O(k^3)$; the result then follows from Lemma 16.

If $|F| = O(k^3)$, we are done. Otherwise, following the approach described in Subsection 4.1 in the full version [11], we greedily construct a biconnectivity deletion set in G, at each step keeping track of the edges that become critical. That is, we let $\hat{S} = \{f_1, \ldots, f_r\} \subseteq F$ be a set greedily constructed as follows. The edge f_1 is an arbitrary edge in F and for each $2 \leqslant i \leqslant r$, f_i is an arbitrary edge which is *not* critical in $G - \{f_1, \ldots, f_{i-1}\}$. As earlier, we terminate this procedure after k steps if we manage to find edges $\{f_1, \ldots, f_k\}$ or earlier if for some $r < k$, every remaining edge of F is critical in $G - \{f_1, \ldots, f_r\}$.

If $r = k$, then we identify the instance as a yes-instance and return an arbitrary yes-instance of constant size. Otherwise, if there is an $i \in [r]$ such that $G - \{f_1, \ldots, f_i\}$ is biconnected and $|\mathsf{Critical}_{G-\{f_1,\ldots,f_{i-1}\}}(f_i)| \geqslant 20k^2 + 46k$, then we execute the case analysis, as in Subsection 4.1 in the full version [11], and in polynomial time, either find $3k+1$ distinct partner sets or an irrelevant edge. In the latter case, we simply remove this irrelevant edge from F (add it to the set E^∞). Finally, if we reach a case with at least $3k+1$ distinct partner sets, we can find a biconnectivity deletion set $S \subseteq F$ with $|S| \geqslant k$ in polynomial time (see the full version), and since we are dealing with the unweighted case, we can simply identify the instance as a yes-instance and return an arbitrary yes-instance of constant size.

The only remaining case is that this greedy algorithm fails to produce a large enough solution yet never marks too many edges as critical at once. That is, it terminates in $r < k$ steps and never marks more than $20k^2 + 46k$ edges as critical in step i for any $i \in [r]$. This implies that $|F| \leqslant 20k^3 + 46k^2 + k = O(k^3)$, completing the proof of the theorem. ◀

5 Conclusions

Our results on Path-contraction Preserving Strong Connectivity and Weighted Biconnectivity Deletion provide additional data points for the algorithmic landscape of graph editing problems under connectivity constraints and its application in network design.

Since we established that Path-contraction Preserving Strong Connectivity is W[1]-hard for general digraphs, we ask whether the problem becomes FPT when restricted to planar digraphs or other structurally sparse classes.

Concerning the parameterized algorithm for Weighted Biconnectivity Deletion, we ask whether the dependence of $2^{O(k \log k)}$ can be improved to single-exponential or proven to be optimal. Naturally, we would further like to know whether we can reach beyond *bi*connectivity and extend our algorithm to higher values of vertex-connectivity. Is it possible to obtain a similar algorithm on digraphs?

Finally, regarding our polynomial kernel for Unweighted Biconnectivity Deletion, we ask whether it is possible to obtain a deterministic kernel. It is also left open whether the weighted case admits a polynomial kernel.

The results presented in this paper raise more questions than they answer, a clear indication that connectivity constraints are far from properly explored under the paradigm of parameterized complexity. As such, the topic offers exciting but challenging opportunities for further research.

References

1 Jørgen Bang-Jensen and Gregory Z Gutin. *Digraphs: theory, algorithms and applications*. Springer Science & Business Media, 2008.

2 Jørgen Bang-Jensen and Anders Yeo. The minimum spanning strong subdigraph problem is fixed parameter tractable. *Discrete Applied Mathematics*, 156(15):2924–2929, 2008.

3 Manu Basavaraju, Fedor V Fomin, Petr Golovach, Pranabendu Misra, MS Ramanujan, and Saket Saurabh. Parameterized algorithms to preserve connectivity. In *Proceedings of the 41st International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 800–811. Springer, 2014.

4 Manu Basavaraju, Pranabendu Misra, M. S. Ramanujan, and Saket Saurabh. On finding highly connected spanning subgraphs. *CoRR*, abs/1701.02853, 2017.

5 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

6 Fedor V. Fomin, Daniel Lokshtanov, Fahad Panolan, and Saket Saurabh. Efficient computation of representative families with applications in parameterized and exact algorithms. *Journal of the ACM (JACM)*, 63(4):29:1–29:60, September 2016.

7 András Frank. Augmenting graphs to meet edge-connectivity requirements. *SIAM J. Discrete Math.*, 5(1):25–53, 1992.

8 András Frank. *Connections in Combinatorial Optimization*. Oxford Univ. Press, 2011.

9 András Frank and Tibor Jordán. Minimal edge-coverings of pairs of sets. *J. Comb. Theory, Ser. B*, 65(1):73–110, 1995.

10 Jiong Guo and Johannes Uhlmann. Kernelization and complexity results for connectivity augmentation problems. *Networks*, 56(2):131–142, 2010.

11 Gregory Gutin, M. S. Ramanujan, Felix Reidl, and Magnus Wahlström. Path-contractions, edge deletions and connectivity preservation. *CoRR*, abs/1704.06622, 2017. URL: `https://arxiv.org/abs/1704.06622`.

12 Stefan Kratsch. Recent developments in kernelization: A survey. *Bulletin of the EATCS*, 113, 2014.

13 Stefan Kratsch and Magnus Wahlström. Representative sets and irrelevant vertices: New tools for kernelization. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 450–459. IEEE Computer Society, 2012.

14 Dániel Marx and László A Végh. Fixed-parameter algorithms for minimum-cost edge-connectivity augmentation. *ACM Transactions on Algorithms (TALG)*, 11(4):27, 2015.

15 Dennis M Moyles and Gerald L Thompson. An algorithm for finding a minimum equivalent graph of a digraph. *Journal of the ACM (JACM)*, 16(3):455–460, 1969.

16 Hiroshi Nagamochi. An approximation for finding a smallest 2-edge-connected subgraph containing a specified spanning tree. *Discrete Applied Mathematics*, 126(1):83–113, 2003. 5th Annual International Computing and combinatorics Conference.

17 László A. Végh. Augmenting undirected node-connectivity by one. *SIAM J. Discrete Math.*, 25(2):695–718, 2011.

18 T. Watanabe and A. Nakamura. Edge-connectivity augmentation problems. *J. Comput. System Sci.*, 35:96 – 144, 1987.