Parameterized Algorithms for Survivable Network Design with Uniform Demands*

Jørgen Bang-Jensen† Manu Basavaraju‡ Kristine Vitting Klinkby§
Pranabendu Misra¶ M. S. Ramanujan‖ Saket Saurabh** Meirav Zehavi††

Abstract

In the Survivable Network Design Problem (SNDP), the input is an edge-weighted (di)graph G and an integer r_{uv} for every pair of vertices $u, v \in V(G)$. The objective is to construct a subgraph H of minimum weight which contains r_{uv} edge-disjoint (or node-disjoint) u-v paths. This is a fundamental problem in combinatorial optimization that captures numerous well-studied problems in graph theory and graph algorithms. Consequently, there is a long line of research into exact-polynomial time algorithms as well as approximation algorithms for various restrictions of this problem.

An important restriction of this problem is one where the connectivity demands are the same for every pair of vertices. In this paper, we first consider the edge-connectivity version of this problem which we call λ-Edge Connected Subgraph (λ-ECS). In this problem, the input is a λ-edge connected (di)graph G and an integer k and the objective is to check whether G contains a spanning subgraph H that is also λ-edge connected and H excludes *at least* k edges of G. In other words, we are asked to compute a maximum subset of edges, of cardinality at least k, which may be safely deleted from G without affecting its connectivity. If we replace λ-edge connectivity with λ-vertex connectivity we get the λ-Vertex Connected Subgraph (λ-VCS) problem.

We show that λ-ECS is fixed-parameter tractable (FPT) for both graphs and digraphs even if the (di)graph has non-negative real weights on the edges and the objective is to exclude from H, some edges of G whose total weight exceeds a prescribed value. In particular, we design an algorithm for the weighted variant of the problem with running time $2^{\mathcal{O}(k \log k)}|V(G)|^{\mathcal{O}(1)}$. We follow up on this result and obtain a polynomial compression for λ-ECS on unweighted graphs. As a direct consequence of our results, we obtain the first FPT algorithm for the parameterized version of the classical Minimum Equivalent Graph (MEG) problem. We also show that λ-VCS is FPT on digraphs; however the problem on undirected graphs remains open. Finally, we complement our algorithmic findings by showing that SNDP is W[1]-hard for both arc and vertex connectivity versions on digraphs. The core of our algorithms is composed of new combinatorial results on connectivity in digraphs and undirected graphs.

1 Introduction

Network design problems, and survivable network design in particular, are fundamental research topics in combinatorial optimization and algorithms with widespread applications. The survivable network design problem involves designing a cost effective communication network that can survive equipment failures. The failure may be caused by any number of things such as a hardware or software breakage, human error, or a broken link between two network components. Such problems are often modeled as graphs, with the nodes representing the network components (such as computers, routers, etc.), and edges representing the communication links between the components. Thus, a generic network design problem corresponds to a problem of finding a subgraph satisfying certain connectivity constraints, or augmenting a given network to certain connectivity requirements. In particular, the input is an undirected graph (or digraphs) with weights on the edges or nodes and prescribed demands on connectivity between nodes in the graph with the objective being the computation a subgraph of minimum weight that satisfies the connectivity demands.

In the most general variant of these problems, called the Survivable Network Design Problem (SNDP), the input is an edge-weighted graph G and an integer r_{uv} for every pair of vertices $u, v \in V(G)$. The objective is to construct a subgraph H of minimum weight which contains r_{uv} edge-disjoint (or node-

*This work is supported by PARAPPROX, ERC starting grant no. 306992.

†University of Southern Denmark, Odense, Denmark, jbj@imada.sdu.dk.

‡Department of Computer Science and Engineering, NITK Surathkal, India, manub@nitk.ac.in.

§University of Bergen, Bergen, Norway, and University of Southern Denmark, Odense, Denmark, Kristine.Knudsen@uib.no.

¶University of Bergen, Norway, Pranabendu.Misra@uib.no.

‖University of Warwick, UK, R.Maadapuzhi-Sridharan@warwick.ac.uk.

**University of Bergen, Norway, Institute of Mathematical Sciences, HBNI, India, and UMI ReLax saket@imsc.res.in.

††Ben-Gurion University, Israel, meiravze@bgu.ac.il.

disjoint) u-v paths for every pair of vertices u, v. Depending on the type of demands or weights allowed there are numerous variants of SNDP, and there is a long line of research into the design of polynomial-time exact algorithms as well as approximation algorithms for these problems. A highlight of this line of research is the 2-approximation algorithm of Jain [20] for the *edge-connectivity* version of SNDP. This work introduced the iterative rounding technique which has subsequently become an important part of the approximation algorithms toolkit. Kortsarz et al. [22] were the first to prove a lower bound for the *node-connectivity* of SNDP and showed that this problem cannot be approximated within a factor of $2^{\log^{1-\epsilon} n}$ for any $\epsilon > 0$. Subsequently, Chakraborty et al. [9] improved this lower bound to p^{ϵ}, where p is the minimum connectivity demand and it exceeds p_0, with p_0 and ϵ being fixed constants. More recently, Chuzhoy and Khanna [10] gave an $\mathcal{O}(q^3 \log n)$-factor approximation algorithm for this problem, where q is the maximum of the connectivity demands. There is also a significant amount of literature on the *directed* versions of SNDP. In this case, there is an integer r_{uv} for every *ordered pair* $(u, v) \in V(G) \times V(G)$. We direct the reader to the surveys [21, 23] on the topic of survivable network design.

One of the most well-studied restrictions of SNDP is the version where the demands are *uniform*. That is, for some λ, $r_{uv} = \lambda$ for every pair $u, v \in V(G)$. This restriction is termed λ-SNDP with Uniform Demands and the edge-connectivity version is called λ-Edge Connected Subgraph (λ-ECS). This problem generalizes many other well studied problems such as Hamiltonian Cycle, Minimum Strongly Connected Spanning Subgraph (MSCSS), 2-Edge Connected Spanning Subgraph etc. If we replace λ-edge connectivity with κ-vertex connectivity we get the κ-Vertex Connected Subgraph (κ-VCS) problem. We again direct the reader to the surveys [21, 23] for more details.

In this paper, we study the *parameterized complexity* of λ-ECS and λ-VCS. In parameterized complexity each problem instance is of the form (x, k) where x is the problem instance, and k is a positive integer called the *parameter*. The notion of tractability in parameterized complexity is called *fixed parameter tractability* (FPT). This entails solvability of any instance (x, k) in time $\tau(k) \cdot |x|^{\mathcal{O}(1)}$, where τ is an arbitrary computable function. We refer to textbooks [11, 13] for an introduction to parameterized complexity.

The most natural parameterization when studying an NP-complete problem is the *size* of the solution which in this case would be the number of edges in H. However, observe that since H is required to be λ-connected, every vertex in H has degree at least λ, implying that H has at least $\frac{\lambda n}{2}$ edges. Hence, if we are looking for a λ-connected subgraph of G containing at most k edges, then either $\frac{\lambda n}{2} > k$, in which case there is no such subgraph, or $n \leq \frac{2k}{\lambda}$ in which case we can just go over all edge subsets of G of size at most k, resulting in a trivial FPT algorithm. So, perhaps a more meaningful question is whether there is a subgraph H on at most $\frac{\lambda n}{2} + k$ edges, where k is the parameter (such parameterizations are called above/below guarantee parameterization, see [18, 24] for an introduction to this topic). However, this problem cannot even have an algorithm of the form $\mathcal{O}(n^{g(k)})$ unless P = NP, for any arbitrary function g. The reason is as follows. Any 2-edge connected graph G has a hamiltonian cycle if and only if it has a 2-edge connected spanning subgraph with $n = \frac{2 \cdot n}{2} + 0$ edges. That is, we set $\lambda = 2$ and $k = 0$. Then, an algorithm for this problem that runs in time $\mathcal{O}(n^{g(k)})$ in fact runs in polynomial time and solves the Hamiltonian Cycle problem. Hence, a more meaningful parameterization is the 'dual' parameter, which is the number of edges of G *not* in H. We study both the weighted and unweighted versions of λ-ECS with this parameterization. Formally, the parameterized (unweighted) λ-ECS problem, denoted by p-λ-ECS, is defined as follows.

p-λ-ECS

Input:	A graph or digraph G which is λ-edge connected and an integer k.
Parameter:	k
Problem:	Is there a set $F \subseteq E(G)$ of size at least k such that $H = G - F$ is also λ-edge connected?

Our Results and Methods. In this paper we show that the p-λ-ECS problem is FPT by proving the following theorem.

Theorem 1.1. *p-λ-ECS can be solved in time $2^{\mathcal{O}(k(\log \lambda + \log k))} n^{\mathcal{O}(1)}$.*

We point out that the exponent of n in the polynomial component of the running time is in fact independent of λ. We also extend our result to the *weighted* version of this problem. In this version, which we call p-Weighted λ-ECS, the input also contains a function $w : E(G) \to \mathbb{R}_{\geq 0}$ and a target weight $\alpha \in \mathbb{R}$ and the objective is to find a set F of at most k edges such that $w(F) \geq \alpha$ and $G - F$ is λ-edge connected. We build on structural insights gained during the design of the algorithm for the unweighted version, to obtain an algorithm with a similar running time for p-Weighted λ-ECS.

Theorem 1.2. *p-Weighted λ-ECS can be solved in time $2^{\mathcal{O}(k(\log \lambda + \log k))} n^{\mathcal{O}(1)}$.*

p-WEIGHTED λ-ECS

Input:	A graph or digraph G which is λ-edge connected, $w : E(G) \to \mathbb{R}_{\geq 0}$, a target weight $\alpha \in \mathbb{R}$ and an integer k.
Parameter:	k
Problem:	Is there a set $F \subseteq E(G)$ of size at most k such that $H = G - F$ is also λ-edge connected and $w(F) \geq \alpha$?

Then we turn to polynomial kernelization (compression) (see [11, 13]). A parameterized problem $\Pi \subseteq \Sigma^* \times \mathbb{N}$ is said to admit a *polynomial kernel*, if there is a polynomial time algorithm which given an instance $(x, k) \in \Pi$ returns an instance $(x', k') \in \Pi$ such that $(x, k) \in \Pi$ if and only if $(x', k') \in \Pi$ and $|x'|, k' \leq k^{\mathcal{O}(1)}$. A *polynomial compression* is a relaxation of polynomial kernelization where the output may be an instance of a (fixed) different language than the input language. That is, polynomial kernelization can be viewed as polynomial time self-reduction while polynomial compression is a polynomial time reduction to a different language. Our next main result is a randomized polynomial compression for the *p*-λ-ECS problem.

THEOREM 1.3. *For any $\delta > 0$, there exists a randomized compression for p-λ-ECS of size $\mathcal{O}(k^{18}\lambda^6(\log k\lambda + \log(1/\delta))$, such that the error probability is upper bounded by $1 - \delta$.*

If we replace λ-edge connectivity with κ-vertex connectivity, then we get the following problem.

p-WEIGHTED κ-VCS

Input:	A digraph D which is κ-vertex connected, $w : A(D) \to \mathbb{R}_{\geq 0}$, a target weight $\alpha \in \mathbb{R}$ and an integer k.
Parameter:	k
Problem:	Is there a set $F \subseteq A(D)$ of size at most k such that $H = D - F$ is also κ-vertex connected and $w(F) \geq \alpha$?

THEOREM 1.4. *p*-WEIGHTED κ-VCS *can be solved in time* $2^{\mathcal{O}(k(\log\kappa + \log k))}n^{\mathcal{O}(1)}$.

However, the *p*-WEIGHTED κ-VCS problem on undirected graphs remains open. Finally, we complement our algorithmic findings by showing that SNDP is W[1]-hard for both arc and vertex connectivity versions on digraphs via a *parameter preserving reduction* (see [11, 13]) from INDEPENDENT SET.

At a high level, Theorems 1.1, 1.2 and 1.4 are based on a greedy strategy and the submodularity of the 'cut' function. An edge in a (di)graph is *deletable* if removing it does not alter the connectivity of the input (di)graph. At the heart of our algorithm for digraphs is the following combinatorial result. If a digraph contains $\Omega(\lambda k^2)$ deletable edges then there is a set of k edges which can be removed from the graph without altering the connectivity of the input digraph. To prove this result we consider the structure of a maximal set of edges whose removal leaves the connectivity of the input digraph unaltered. We delete the edges of such a set iteratively and note how, in each step, deletable edges in the graph turn into undeletable edges. The crux of our argument is a lemma that states that if at any step a large number of edges turn undeletable, then we may remove this large set of edges from the graph without altering the connectivity of the input digraph. Next we move to undirected graphs where, surprisingly, the arguments turned out to be much more difficult than in the case for digraphs. While the arguments for undirected graphs are similar in spirit, they are much more involved than the digraphs. Especially, when G is an undirected graph and λ is an odd integer, the arguments involve a lot of subtleties and we require a considerable strengthening of the structural results proved for digraphs and for undirected graphs when λ is an even integer. All our proofs are based on carefully applying arguments based on submodularity. In a nutshell, for an unweighted graph G and a positive integer k, we give a polynomial time algorithm that either finds a set of k edges whose deletion does not alter the connectivity of the input (di)graph, or computes a set $\mathsf{del}(G) \subseteq E(G)$ of size $\mathcal{O}(\lambda k^3)$ such that, if there exists a set $F \subseteq E(G)$ of size at least k such that $H = G - F$ is also λ-connected then there exists a desired set of size at least k contained in $\mathsf{del}(G)$. For the weighted version of the problem, we have a similar polynomial time algorithm that finds a set $W \subseteq E(G)$ of size $\mathcal{O}(\lambda k^3)$ such that, if there exists a set $F \subseteq E(G)$ of size at most k such that $H = G - F$ is λ-connected and $w(F) \geq \alpha$ then there exists a set $F^\star$ of size at most k such that $H = G - F^\star$ is λ-connected, $w(F^\star) \geq \alpha$ and $|F \cap W| \geq 1$. This leads to a branching algorithm with the desired running time. Finally, we design a polynomial compression for the *p*-λ-ECS problem. Towards this we utilize the ideas and methods developed for dynamic graph optimization problems. In particular, we use the results proved in [1] regarding dynamic sketches for *s*-*t*-edge connectivity problems.

An immediate corollary of our fixed-parameter tractability result for *p*-λ-ECS is the first fixed-parameter algorithm for the parameterized version of the classical MINIMUM EQUIVALENT GRAPH (MEG)

problem. In this problem, the goal is to find a minimum spanning subgraph which is "equivalent" to the input graph. Two graphs G and H are said to be *equivalent* if for any two vertices u, v, the vertex v is reachable from u in G,if and only if, v is reachable from u in H. This problem is easily seen to be NP-complete, by a reduction from the Hamiltonian Cycle problem [16]. The natural *parameterized* version of this problem asks, given a graph G and integer k, whether there is a subgraph H on at most $m - k$ edges which is equivalent to G. It is well known that Minimum Equivalent Graph can be reduced to an input G' which is strongly connected (that is, there is directed path between every pair of vertices in G'). The following proposition due to Moyles and Thompson [26], see also [2, Sections 2.3], reduces the problem of finding a minimum equivalent sub-digraph of an arbitrary G to a strong digraph.

Proposition 1.1. *Let G be a digraph on n vertices with strongly connected components $C_1, \ldots, C_r$. Given a minimum equivalent subdigraph C_i' for each C_i, $i \in [r]$, one can obtain a minimum equivalent subdigraph G' of G containing each of C_i' in $\mathcal{O}(n^\omega)$ time. Here, ω is the exponent of the fastest known matrix multiplication algorithm and ω is currently upper bounded by* 2.376.

Proposition 1.1 allows us to reduce an instance of Minimum Equivalent Graph on a general digraph to instances where the graph is strongly connected, in polynomial time. We then solve Minimum Equivalent Graph by executing the algorithm of Theorem 1.1 with $\lambda = 1$ for each strongly connected component of the input digraph.

Related work. It is important to point out another parameterization of p-λ-ECS on digraphs when $\lambda = 1$. Bang-Jensen and Yeo [4] used the fact that every strongly-connected graph has an equivalent subdigraph containing at most $2n - 2$ arcs to study the parameterization below $2n - 2$ instead of m and obtained an algorithm that runs in time $2^{\mathcal{O}(k \log k)} n^{\mathcal{O}(1)}$ and decides whether a given strongly connected digraph has an equivalent digraph with at most $2n - 2 - k$ edges. However, observe that when $m \leq 2n-2-k$ this parameterization is not useful since the entire edge set of G is already a solution. Marx and Végh gave a FPT algorithm for computing a minimum cost set of at most k links, which augments the connectivity of an undirected graph from $\lambda-1$ to λ [25]. Basavaraju et. al. [5] improved the running time this algorithm and, also gave an algorithm for another variant of this problem. The first exact algorithm for MEG and MSCSS, running in time $2^{\mathcal{O}(m)}$ time, where m is the number of edges in the graph, was given in by Moyles and Thompson [26] in 1969. Only recently, Fomin et. al. gave the first single-exponential algorithm for MEG and MSCSS, i.e. with a running time of $2^{\mathcal{O}(n)}$ [14]. For the special case of Hamiltonian Cycle, a $\mathcal{O}(2^n)$ algorithm is known [19, 7] for general di-graphs from 1960s. It was recently improved to $\mathcal{O}(1.657^n)$ for graphs [8], and to $\mathcal{O}(1.888^n)$ for bipartite di-graphs [12]. For other results and more details we refer to Chapter 12 of [2].

In the rest of the paper we give an overview of our results and methods, and also give brief sketches of the proofs of several key results. The complete proofs along with all other relevant details will appear in the full version of the paper (also see [6]). Let us also note that, in the following, whenever we talk of λ-connectivity of a graph we mean λ-edge connectivity unless stated otherwise.

2 An Overview of Our Results

A crucial notion we will use repeatedly is that of deletable edges, and their relation among themselves. An edge in a graph is *deletable*, if removing it does not violate the required connectivity constraints, and otherwise it is *undeletable*. We denote by $\mathsf{del}(G)$ the set of deletable edges in G, and by $\mathsf{undel}(G)$ the set of undeletable edges in G. It is clear that any subset of edges F (called a *deletion set*) of cardinality k, such that $G - F$ is λ-connected, is always a subset of the collection of deletable edges. Let us note that, we can additionally specify a set of forbidden edges $S \subseteq \mathsf{del}(G)$ such that $F \cap S = \emptyset$. In this case, we update the set of deletable edges in G to $\mathsf{del}(G) - S$. We show the following structural result, relating the number of deletable edges to the cardinality of a deletion set.

> ($\star$) If a graph contains $\Omega(\lambda k^3)$ deletable edges then there is a set of k edges which can be removed from the graph without violating the connectivity constraints.

At a first glance, this is obviously false. For example, set $\lambda = 1$ and consider an arbitrarily long cycle. Then every edge of the cycle is a deletable edge but no more than one edge may be deleted without disconnecting the graph. Note that, this example can be generalized to any *odd* value of λ. However, we show that the statement does indeed hold for digraphs (for any value of λ), and for undirected graphs whenever λ is even. Hence, we prove the following lemmas for undirected graphs and digraphs.

Lemma 2.1. *Let G be an undirected graph and k be an integer, such that G is λ-connected where λ is an even integer. Then in polynomial time, we can either find a set F of cardinality k such that $G - F$ is λ-connected, or conclude that G has at most $2\lambda k^2$ deletable edges in total.*

LEMMA 2.2. *Let G be a digraph and k be an integer such that, G is λ-connected for some integer λ. Then in polynomial time, we can either find a set F of cardinality k such that $G - F$ is λ-connected, or conclude that G has at most λk^2 deletable edges in total.*

The case of undirected graphs and an odd value of λ is much more involved. In this case, we show that the statement ($\star$) is essentially true if we restrict our deletion set to a well chosen subset of the deletable edges, which can be computed in polynomial time. As can be observed in the above example of a cycle, it is possible to identify certain deletable edges as being disjoint from *some* deletion set of cardinality k in the given graph. We call an edge satisfying this property, an *irrelevant* edge. We give a polynomial time procedure that identifies certain edges as irrelevant in the given graph. We use this procedure to iteratively grow the set of irrelevant edges, always ensuring that if there is a deletion set of k edges then there is one that is disjoint from this set. Finally, by excluding these irrelevant edges from the set of deletable edges, we show that the proposed statement holds true.

LEMMA 2.3. *Let $\lambda \in \mathbb{N}$ be odd. Let G be an undirected graph such that G is λ connected, k be an integer and let $\mathcal{R}$ a subset of edges of G. Then there is a polynomial time algorithm that, either either computes a subset of edges $F \subseteq E(G) \setminus R$ of cardinality k such that $G - F$ is λ-connected, or finds an edge e in $E(G) \setminus \mathcal{R}$ such that the given graph has a deletion set of cardinality k that is disjoint from $\mathcal{R}$ if and only if it has such a set disjoint from $\mathcal{R} \cup \{e\}$, or concludes that there are at most $\lambda(6k^3 + 9k^2 + k)$ deletable edges in $E(G) \setminus \mathcal{R}$.*

Our proof of these statements is built upon a close examination of a greedily constructed maximal set deletion set F. Let us first discuss the case of digraphs, and undirected graphs with an even value of λ. We may assume that the graph has more than $\mathcal{O}(\lambda k^2)$ deletable edges to begin with, as otherwise the claims are trivially true. Now, if the greedy deletion set has k or more edges, then we are done. Otherwise, we delete the edges of the greedy deletion set in an arbitrary but fixed sequence and examine its effect on the other deletable edges of the graph. Each time an edge in the greedy deletion set is deleted several other deletable edges in the graph will become undeletable in the remaining graph. Since at the end of this deletion sequence, all the remaining edges are undeletable, there must be a step where $\Omega(\lambda k)$ edges turn undeletable after having been deletable prior to this step. We show that we can extract another set X of k edges from this collection of edges such that $G - X$ is λ-connected. The process of extracting this set X from the above collection of edges is as follows. We show that there is a subset of $\Theta(k)$ edges in this collection such that there are no λ-cuts in the current graph which separate the endpoints of more than one edge in this subset. We then show a way to pick k edges from this subset such that these edges forms a deletion set.

For the case of undirected graphs with an odd value of λ, the algorithm, and the corresponding analysis, becomes much more involved. We obtain a novel structural result on such graphs, which could be of independent interest. We show that if a graph has a large number of deletable edges but only admits a small deletion set, then it can be decomposed into a "cyclic structure". More precisely, we obtain a partition of the vertex set of the graph such that, the sets in the partition can be arranged in a cycle with each subset being "adjacent" only to two neighboring subsets. We give a polynomial time algorithm that either finds a deletion set of size k, or computes such a decomposition of the graph. This algorithm proceeds in the same manner as the previous algorithm, and computes a candidate deletion set X of k edges. However, it could be the case that $G - X$ is not λ-connected. In this case, by examining the relations between the edges in set X, the algorithm gradually builds a cyclic-decomposition of the graph. Essentially we show that $G - X$ is not λ-connected implies that the edges in X are distributed across a "cyclic partition" of the graph. Then this cyclic decomposition allows us to identify a new irrelevant edge in the graph. By an iterative application of this algorithm, we arrive at the aforementioned lemma.

While our algorithms are quite simple, the analysis is fairly involved, which examines the structure of the graph by applying the *submodularity of edge-cuts*. Interestingly, the analysis is much simpler in the case of digraphs as compared to the case of undirected graphs.

The above results imply our FPT algorithm for p-λ-ECS in any unweighted (di)graph. This is because in polynomial time, we can either compute a solution or conclude that the set of deletable edges (which are not irrelevant) is bounded by a polynomial in k. This implies a branching algorithm for p-λ-ECS with the claimed running time. The above results also form the starting point of our *polynomial compression* for the p-λ-ECS problem. This is because, we have proved that unless the number of deletable edges in the instance is bounded, we can always compute a solution in polynomial time. Hence, we may assume that the instance has $\mathcal{O}(\lambda k^3)$ deletable edges and we use the results of Assadi et.al. [1] to give a randomized polynomial compression for such instances. Assadi et.al. [1] give a dynamic sketching scheme for finding

min-cuts between a fixed pair of vertices in a dynamic graph, where the dynamic edge set is restricted to the edges between a fixed subset of vertices. We obtained the claimed compression by treating the deletable edges of the graph as the aforementioned set of dynamic edges and using certain structural properties of a solution.

Finally, we turn to p-WEIGHTED λ-ECS. First, note that our results can be used to solve a more general version of p-λ-ECS. In this generalization, there is an additional requirement that the solution *must* be contained in a given subset W of the edges of the graph. To be precise we give a polynomial time algorithm that, given a set W containing $7\lambda k^3$ deletable edges of the graph (λk^2 deletable edges for digraphs), finds a deletion set of cardinality k(if one exists) that is additionally a subset of W. While the set W is not explicitly mentioned in the statements of our lemmas and theorems, we always assume that the set of deletable edges is restricted to be a subset of W. This fact comes in handy for designing an FPT algorithm for p-WEIGHTED λ-ECS, where we must find a deletion set of maximum total weight which contains up to k edges. We use the following simple observation which leads us to the algorithm for weighted instances.

> Let W be the set of the 'heaviest' $7\lambda k^3$ deletable edges, (only λk^2 edges for digraphs). Then there is a polynomial time algorithm that either correctly concludes that there is a solution (of the required kind) which intersects this set W (and this is the only possibility for digraphs), or computes an edge $e \in W$ which can be safely added to the set of irrelevant edges.

For digraphs and undirected graphs with an even value of λ, this result follows easily from the arguments in the unweighted case as there are no irrelevant edges to deal with. For undirected graphs with an odd value of λ, we have to be more careful while marking an edge as irrelevant, lest it affect the weight of the required solutions. For this, we use a modification of our scheme for finding irrelevant edges in the unweighted odd λ case.

For the vertex connectivity version of our problem on digraphs, we translate the vertex connectivity as a "special kind of edge connectivity" on a related digraph. This allows us to use the results and methods for edge connectivity to solve the vertex connectivity problem on digraphs.

3 Preliminaries

A *cut* (X, Y) in a graph or digraph G is an ordered partition of $V(G)$. Therefore, for any subset X of $V(G)$ we have a cut $(X, \overline{X})$. We also use the term "the cut X" to denote $(X, \overline{X})$. In undirected graphs $(X, \overline{X})$ and $(\overline{X}, X)$ denote the same cut. We say that $(X, \overline{X})$ *separates a pair of vertices* $\{u, v\}$ if exactly one of these vertices is in X. We say that an edge(arc) $e = (u, v)$ *crosses* $(X, \overline{X})$, if the cut separates $\{u, v\}$. In directed graphs we distinguish between the cuts $(X, \overline{X})$ and $(\overline{X}, X)$. We say that $(X, \overline{X})$ *separates an ordered pair of vertices* $\{u, v\}$ only if $u \in X, v \in \overline{X}$. We say that an edge(arc) $e = (u, v)$ crosses $(X, \overline{X})$, if the cut separates the ordered pair $\{u, v\}$. For a subgraph H of G and a cut $(X, \overline{X})$, we define $\partial_H(X)$ as the set of edges(arcs) in H which cross this cut. We use $\delta_H(X)$ to denote the *number* of edges(arcs) in H which cross this cut, that is $\delta_H(X) = |\partial_H(X)|$. We also say that an edge(arc) e *is part of the cut* $(X, \overline{X})$ if e crosses $(X, \overline{X})$. For a number λ and a graph H, we say that $(X, \overline{X})$ is a λ-cut in H if $\delta_H(X) = \lambda$.

The key tool in our arguments is the submodularity of graph cuts. A function $f : 2^V \to \mathbb{R}$, where V is a finite set, is called *submodular* if for any $X, Y \subseteq V, f(X \cap Y) + f(X \cup Y) \leq f(X) + f(Y)$. It is well known that cuts are submodular [15, 27].

PROPOSITION 3.1. *Let G be a (di)graph. Let $(X, \overline{X})$ and $(Y, \overline{Y})$ be two cuts in G. Then $\delta(X \cap Y) + \delta(X \cup Y) \leq \delta(X) + \delta(Y)$. Furthermore, if $e \in \partial(X \cap Y) \cup \partial(X \cup Y)$, then $e \in \partial(X) \cup \partial(Y)$.*

The submodularity of cuts implies that the λ-cuts in a λ-connected graph where λ is *odd*, form a laminar family and this fact is a crucial component of our algorithm when λ is odd.

PROPOSITION 3.2. *Let $\lambda \in \mathbb{N}$ be odd and let G be a λ-connected graph. Let $(X, \overline{X})$ and $(Y, \overline{Y})$ be two λ-cuts in G such that $X \cup Y \neq V(G)$. Then, $(X, \overline{X})$ and $(Y, \overline{Y})$ do not cross and we have that, either $X \subseteq Y$ or $Y \subseteq X$.*

For a digraph $D = (V, A)$ a *vertex splitting* of a vertex $v \in A$ consist of dividing the vertex into two vertices v^- and v^+ such that there is a arc from v^- to v^+ and every in-arc to v is instead going into v^- and every out-arc of v is going out of v^+.

4 Edge Connectivity in Digraphs

In this section, we present our results for p-λ-ECS on digraphs. Let $G = (V, A)$ be a digraph, and for any arc $e = (u, v)$ let $\mathcal{D}(e)$ denote the set of deletable arcs of G which are undeletable in $G - e$. We will deal with a *fixed* deletable arc $e^* = (u^*, v^*)$ in G such that $\mathcal{D}(e^*)$ has at least $k\lambda$ arcs. The main lemma we require for our algorithm is the following.

LEMMA 4.1. *Let $G = (V, A)$ be a digraph and $\lambda \in \mathbb{N}$ such that G is a λ-connected digraph. If there is a deletable arc $e^* \in A(G)$ such that $|\mathcal{D}(e^*)| \geq k\lambda$ then*

there is a set $\mathcal{Z} \subseteq \mathcal{D}(e^*)$ of k arcs such that $G - \mathcal{Z}$ is λ-connected.

We denote by G^* the graph $G - e^*$. Since e^* is by definition, deletable in G, it follows that G^* is a λ-connected digraph. Furthermore, for the fixed arc e^*, we denote by $\mathcal{Z}(e^*)$ a subset $\{e_1, \ldots, e_k\}$ of $\mathcal{D}(e^*)$ which has the property that for any λ-cut $(X, \overline{X})$ in G^* that separates the pair $\{u^*, v^*\}$, the intersection of the arcs of this cut with $\mathcal{Z}(e^*)$ is at most 1. We note that the fact that such a set exists is non-trivial and requires a proof. For every $j \in [k]$, we let $e_j = (u_j, v_j) \in \mathcal{Z}(e^*)$. Finally, for every $i \in [k]$, we denote by $\mathcal{Z}_i$ the set $\{e_1, e_2, \ldots, e_i\} \subseteq \mathcal{Z}(e^*)$ and by G_i^* the subgraph $G^* - Z_i$. Note that $\mathcal{Z}_k = \mathcal{Z}(e^*)$.

We will prove that the digraph $G - \mathcal{Z}(e^*)$ is λ-connected. From this, together with the fact that $|\mathcal{Z}(e^*)| = k$, Lemma 4.1 follows. Hence, in the remaining part of this section we sketch a proof of the claim that $G - \mathcal{Z}(e^*)$ is λ-connected. Before we proceed, we need a final definition.

DEFINITION 4.1. *A cut $(X, \overline{X})$ in G_i^* (for any $i \in [k]$) is called a cut of* **Type 1** *if it separates the ordered pair $\{u^*, v^*\}$ and a cut of* **Type 2** *otherwise. We call $(X, \overline{X})$ a* **violating cut** *if $(X, \overline{X})$ is a cut of Type 1 and $\delta_{G_i^*}(X) \leq \lambda - 2$ or $(X, \overline{X})$ is a cut of Type 2 and $\delta_{G_i^*}(X) \leq \lambda - 1$.*

Observe that if G_k^* excludes every violating cut then $G - \mathcal{Z}(e^*)$ will be λ-edge connected. Hence it will be enough to show the following lemma.

LEMMA 4.2. *For every $i \in [k]$, the digraph G_i^* has no violating cuts.*

Using Lemma 4.2, we obtain Lemma 4.1 which provides a way to compute a deletion set from $\mathcal{D}(e^*)$. We will then use this lemma to prove Lemma 2.2.

Proof of Lemma 2.2. Let $F = \{f_1, f_2, \ldots, f_p\}$ be an arbitrary maximal set of arcs such that $G - F$ is λ-connected. If $|F| = p \geq k$, then we already have the required deletion set. Therefore, we may assume that $p \leq k - 1$. Now, consider the graphs $G_0, \ldots, G_p$ with $G_0 = G$ and G_i defined as $G_i = G - \{f_1, \ldots f_i\}$ for all $i \in [p]$. Note that $G_{i+1} = G_i - f_{i+1}$ and $G_p = G - F$. Observe that each G_i is λ-connected, by the definition of F. Let $\mathcal{D}_i$ be the set of deletable arcs in G_i which are undeletable in G_{i+1}. Observe that $\mathcal{D}_i$ is the set of arcs that turn undeletable when f_i is deleted.

Now consider any deletable arc of G. It is either contained in F, or there is some $i \in \{0, \ldots, p-1\}$ such that it is deletable in G_i but undeletable in G_{i+1}. In other words, the set $F \cup \mathcal{D}_1 \cup \mathcal{D}_2 \ldots \cup \mathcal{D}_p$ covers all the deletable arcs of G. Since $p \leq k - 1$ and the number of deletable arcs in G is at least $k^2\lambda$, it follows that for some $i \in [p]$, the set $\mathcal{D}_i$ has size at least $k \cdot \lambda$. Let $\mathcal{Z}_i$ be the set of at least k arcs corresponding to $\mathcal{D}_i$ guaranteed by Lemma 4.1. We know that $G_i - \mathcal{Z}_i$ is λ-connected. Since G_i is a subgraph of G on the same set of vertices, it follows that $G - \mathcal{Z}_i$ is also λ-connected, which gives us a required deletion set. □

5 Edge Connectivity in Undirected Graphs

In this section, we consider the structural properties of undirected graphs. As mentioned earlier, we need to handle even-connectivity and odd-connectivity separately. When λ is even, we closely follow the strategy used for digraphs, albeit with a more involved analysis. The details will appear in the full version of the paper (also see [6]).

However, the case when λ is odd is much more involved and is sketched in this section. In this case, it is possible that the number of deletable edges is unbounded in k despite the presence of a deletion set of size k. Indeed, recall the following example. Let G be a cycle on n vertices, $\lambda = 1$ and $k = 2$. Clearly, every edge in G is deletable, but there is no deletion set of cardinality 2. In order to overcome this obstacle, we design a subroutine that either find a required deletion set, or detects an edge which is disjoint from *some* deletion set of cardinality k in the graph. Before we formally state the corresponding lemma, we additionally define a subset of irrelevant edges and a deletion set is now defined to be a subset F of $E(G) \setminus \mathcal{R}$ of size k such that $G - F$ is λ-connected. Finally, we note that the set $\mathcal{R}$ contains all the undeletable edges of G.

LEMMA 2.3. *Let $\lambda \in \mathbb{N}$ be odd. Let G be an undirected graph such that G is λ connected, k be an integer and let $\mathcal{R}$ a subset of edges of G. Then there is a polynomial time algorithm that, either either computes a subset of edges $F \subseteq E(G) \setminus R$ of cardinality k such that $G - F$ is λ-connected, or finds an edge e in $E(G) \setminus \mathcal{R}$ such that the given graph has a deletion set of cardinality k that is disjoint from $\mathcal{R}$ if and only if it has such a set disjoint from $\mathcal{R} \cup \{e\}$, or concludes that there are at most $\lambda(6k^3 + 9k^2 + k)$ deletable edges in $E(G) \setminus \mathcal{R}$.*

We can then iteratively execute the algorithm of this lemma to either find a required deletion set or grow the set of irrelevant edges. From now onward, we represent the input to our algorithm as $(G, k, \mathcal{R})$, and assume that λ is an odd integer. We begin by proving the following lemma which says that if the graph admits a "cycle-like" decomposition, then certain deletable edges may be safely added to the set $\mathcal{R}$ without affecting the existence of a deletion set.

LEMMA 5.1. *Let $(G, k, \mathcal{R})$ be an input, where G is λ-connected, and let $X_1, X_2, \ldots X_{2k+2}$ be a partition of $V(G)$ into non-empty subsets such that the following properties hold in the graph G.*

1. *$\delta_G(X_1, X_2) = \delta_G(X_2, X_3) \ldots = \delta_G(X_{2k+2}, X_1) = \frac{\lambda+1}{2}$.*
2. *Every edge of the graph either has both endpoints in some X_i for $i \in [2k+2]$, or contained in one of the edge sets mentioned above.*
3. *There are deletable edges $e_1, e_2, \ldots, e_{2k+2}$ in $E(G) \setminus \mathcal{R}$ such that $e_i \in \partial(X_i, X_{i+1})$ for $i \in [2k+2]$. (Here X_{2k+3} denotes the set X_1.)*

Then $(G, k, \mathcal{R})$ has a deletion set of cardinality k if and only if $(G, k, \mathcal{R} \cup \{e_1\}$ has a deletion set of cardinality k.

Before we proceed with the rest of the section, we set up some notation which will be used in subsequent lemmas. Let S^* denote a fixed subset of $E(G) \setminus \mathcal{R}$ of at most $k-1$ edges such that the graph $G_{S^*} = G - S^*$ is λ-connected. We let $e^* \notin \mathcal{R}$ denote a deletable edge in G_{S^*} such that $\mathcal{D}(e^*) = (\mathsf{del}(G_{S^*}) \cap \mathsf{undel}(G_{S^*} - \{e^*\})) \setminus \mathcal{R}$ has at least $\eta\lambda$ edges where $\eta = 3k(2k+3)+1$. We denote by G^* the graph $G_{S^*} - \{e^*\}$. Let $\mathcal{Z}(e^*) = \{e_1, \ldots, e_\eta\}$ be a collection of edges in $\mathcal{D}(e^*)$ as before in the case of directed graphs. Furthermore, let $\mathcal{C}(e^*) = \{C_1, \ldots, C_\eta\}$ be a collection of η λ-cuts in G^* such that, for each $e_i \in \mathcal{Z}(e^*)$ there is a unique cut $C_i \in \mathcal{C}(e^*)$ which separates the endpoints of e_i and, for every $i \in [\eta - 1]$, $C_i \subset C_{i+1}$. Again, the existence of such a collection requires a proof. Furthermore, we may assume that both these collections are known to us. We remark that *computing* these collections was not particularly important in the case of digraphs or the case of even λ in undirected graphs. This is because the main structural lemmas we proved were only required to be *existential*. However, in the odd case, it is crucial that we are able to *compute* these collections when given the graph G_{S^*} and the edge e^*. For every $i \in [\eta]$, we let (u_i, v_i) denote the endpoints of the edge e_i.

Let $\widehat{\mathcal{Z}} = \{e_{(2k+3)i+1} \in \mathcal{Z}(e^*) \mid 0 \leq i \leq 3k\}$ and observe that $|\widehat{\mathcal{Z}}| = 3k+1$. Let $\widehat{\mathcal{C}}$ be the subcollection of $\mathcal{C}(e^*)$ corresponding to $\widehat{\mathcal{Z}}$. Let $\mathcal{C}$ be defined as the set $\{C_i \in \widehat{\mathcal{C}} \mid (C_i \setminus C_{i-(2k+3)}) \cap V(S^*) = \emptyset\}$ where $V(S^*)$ denotes the set of endpoints of edges in S^*. Since $|S^*| \leq k-1$ at most $2(k-1)$ cuts of $\widehat{\mathcal{C}}$ are excluded from $\mathcal{C}$ and hence, $|\mathcal{C}| \geq k$. Let $\mathcal{Z}$ be the subcollection of $\widehat{\mathcal{Z}}$ corresponding to $\mathcal{C}$. For any $i \in [\eta]$ such that $e_i \in \mathcal{Z}$, we define $\mathcal{Z}_i = \{e_j \in \mathcal{Z} \mid j \leq i\}$ and $G_i^* = G^* - \mathcal{Z}_i$. In the rest of the section, whenever we talk about the set Z_i and graph G_i, we assume that the corresponding edge $e_i \in \mathcal{Z}$ and hence these are well-defined.

DEFINITION 5.1. *Let $i \in [\eta]$ such that $e_i \in \mathcal{Z}$. A cut $(X, \overline{X})$ in G_i^* (for any $i \in [k]$) is called a cut of* **Type 1** *if it separates the pair $\{u^*, v^*\}$ and a cut of* **Type 2** *otherwise. We call $(X, \overline{X})$ a* **violating cut** *if $(X, \overline{X})$ is a cut of Type 1 and $\delta_{G_i^*}(X) \leq \lambda - 2$ or $(X, \overline{X})$ is a cut of Type 2 and $\delta_{G_i^*}(X) \leq \lambda - 1$.*

As before, we have the following lemma for handling Type 1 cuts.

LEMMA 5.2. *For any $i \in [\eta]$ such that $e_i \in \mathcal{Z}$, the graph G_i^* has no violating cuts of Type 1.*

To handle the violating cuts of Type 2, we define a violating triple (X, i, j) and we prove several structural lemmas based on this definition.

DEFINITION 5.2. *Let $i \in [\eta]$ such that $e_i \in \mathcal{Z}$. Let $(X, \overline{X})$ be a violating cut of Type 2 in G_i^* such that $u^*, v^* \notin X$, e_i crosses $(X, \overline{X})$ and X is inclusion-wise minimal. Let $j < i$ be such that $e_j \in \mathcal{Z}$, e_j crosses the cut $(X, \overline{X})$ in G^* and there is no r such that r satisfies these properties and $j < r < i$. Then we call the tuple (X, i, j) a* **violating triple**.

Observe that for any violating triple (X, i, j), it holds that $j \leq i - (2k+3)$ and hence, there are cuts $C_j \subset C_{i-(2k+2)} \subset C_{i-(2k+1)} \ldots \subset C_{i-1} \subset C_i$ such that they are all λ-cuts in G^* and all but C_j and C_i are λ-cuts in G_i^* as well. For the sake of convinience, let us rename these cuts as follows. Let $C_j \subset C_{2k+2} \subset C_{2k+1} \ldots \subset C_1 \subset C_i$ denote the sets $C_j \subset C_{i-(2k+2)} \subset C_{i-(2k+2)} \ldots \subset C_{i-1} \subset C_i$ respectively, and let $\mathcal{C}_{ij}$ denote this ordered collection. Additionally, in some of our arguments we may refer to the cuts C_0 and C_{2k+3}, which denote the cuts C_i and C_j respectively. The following lemma ties the existence of violating cuts of Type 2 to the existence of violating triples.

LEMMA 5.3. *Let $i \in [\eta]$ such that $e_i \in \mathcal{Z}$ and let $(X, \overline{X})$ be a violating cut of Type 2 in G_i^* such that G_{i-1}^* has no such violating cut, $u^*, v^* \notin X$ and X is inclusion-wise minimal. Then, there is a $j < i$ such that (X, i, j) is a violating triple. Furthermore given G, i, X, we can compute j in polynomial time. Finally, the following properties hold with regards to the triple (X, i, j).* **(1)** *$\delta_{G^*}(X) \geq \lambda + 1$,* **(2)** *$X \subseteq C_i \setminus C_j$,* **(3)** *$e_i$ and e_j are the only edges of $\mathcal{Z}$ which cross the cut $(X, \overline{X})$ in G^*,* **(4)** *$\delta_{G_i^*}(X) = \lambda - 1$.*

Let C_a and C_b be two consecutive cuts in $\mathcal{C}_{ij}$ such that $b = a + 1$ and observe that $C_j = C_j \subset C_b \subset C_a \subset C_i = C_i$. Let $X_1 = X \cap (C_i \setminus C_a)$, $X_2 = X \cap (C_a \setminus C_b)$ and $X_3 = C_b \setminus C_j$. We will show that these sets are non-empty and more interestingly, X_2 is in fact *all* of $C_a \setminus C_b$.

LEMMA 5.4. *Let $i \in [\eta]$ such that $e_i \in \mathcal{Z}$ and let (X, i, j) be a violating triple. Let $X_1 \uplus X_2 \uplus X_3$ be the partition of X as defined above. The sets X_1, X_2, X_3 are all non-empty and furthermore, $X_2 = C_a \setminus C_b$.*

Moving forward, when dealing with a violating triple (X, i, j), we continue to use the notation defined earlier. That is, the sets X_1, X_2, X_3 are defined to be the intersections of X with the sets $C_i \setminus C_a$, $C_a \setminus C_b$ and C_b respectively with $X_2 = C_a \setminus C_b$. Furthermore, we may assume that $\delta_{G_i^*}(X_1) = \delta_{G_i^*}(X_3) = \delta_{G_i^*}(X_1 \cup X_2) = \delta_{G_i^*}(X_3 \cup X_2) = \lambda$. The justification for this can be found in the proof of Lemma 5.4. Finally, $\delta_{G_i^*}(X_2) = \lambda + 1$.

Recall that our main objective in the rest of the section is to show that the sets X_1, X_2, X_3 satisfy the premises of Lemma 5.1. For this, we begin by showing that these sets satisfy similar properties with respect to the graph G^* instead of the graph G (which is what is required for Lemma 5.1). Following this, we show how to 'lift' the required properties to the graph G (Lemma 5.6), which will allow us to satisfy the premises of Lemma 5.1.

LEMMA 5.5. *Let $i \in [\eta]$ such that $e_i \in \mathcal{Z}$ and let (X, i, j) be a violating triple. Let $X_1 \uplus X_2 \uplus X_3$ be the partition of X as defined above. Let $W = V(G) \setminus X$. Then, $\delta_{G_i^*}(W, X_1) = \delta_{G_i^*}(X_3, W) = \frac{\lambda-1}{2}$, $\delta_{G_i^*}(X_1, X_2) = \delta_{G_i^*}(X_2, X_3) = \frac{\lambda+1}{2}$. Furthermore, $\delta_{G_i^*}(X_2, W) = \delta_{G_i^*}(X_1, X_3) = 0$.*

LEMMA 5.6. *Let $i \in [\eta]$ such that $e_i \in \mathcal{Z}$ and let (X, i, j) be a violating triple. Let $X_1 \uplus X_2 \uplus X_3$ be the partition of X as defined above. Let $W = V(G) \setminus X$. Then, $\delta_G(W, X_1) = \delta_G(X_1, X_2) = \delta_G(X_2, X_3) = \frac{\lambda+1}{2}$. Furthermore, $\delta_G(X_2, W) = \delta_G(X_1, X_3) = 0$.*

Proof of Lemma 2.3. Let $F = \{f_1, f_2, \ldots, f_p\}$ be an arbitrary maximal set of edges disjoint from $\mathcal{R}$ such that $G - F$ is λ-connected. If $|F| = p \geq k$, then we already have the required deletion set in G. Therefore, we may assume that $p \leq k - 1$.

Now, consider the graphs $G_0, \ldots, G_p$ with $G_0 = G$ and G_i defined as $G_i = G - \{f_1, \ldots f_i\}$ for all $i \in [p]$. Note that $G_{i+1} = G_i - f_{i+1}$ and $G_p = G - F$. Observe that each G_i is λ-connected by the definition of F. Let $\mathcal{D}_i$ be the set of deletable edges in G_i which are undeletable in G_{i+1}. Observe that $\mathcal{D}_i = \mathcal{D}(f_i)$ in the graph G_i.

Now consider any deletable edge of G. It is either contained in F, or there is some $r \in \{0, \ldots, p-1\}$ such that it is deletable in G_i but undeletable in G_{r+1}. In other words, the set $F \cup \mathcal{D}_1 \cup \mathcal{D}_2 \ldots \cup \mathcal{D}_p$ covers all the deletable edges of G. Since $p \leq k - 1$ and the number of deletable edges in G is greater than $\eta\lambda$, it follows that for some $r \in [p]$, the set $\mathcal{D}_r$ has size more than $\eta\lambda$. We fix one such $r \in [p]$ and if $r > 1$, then we define $S^* = \{f_1, \ldots, f_{r-1}\}$ and $S^* = \emptyset$ otherwise. We define $e^* = e_r$.

We then construct the sets $\mathcal{Z}(e^*), \mathcal{C}(e^*), \widehat{\mathcal{Z}}, \widehat{\mathcal{C}}$. Then we construct the cut-collection $\mathcal{C}$ and the corresponding edge set $\mathcal{Z}$ by using the set S^*. Recall that $\mathcal{Z}$ contains at least k edges and is by definition disjoint from $\mathcal{R}$. Consider the graph $G^* = G - S^* \cup \{e^*\} = G - \{f_1, \ldots, f_r\}$ and note that G^* is λ-connected.

We check whether $G^* - \mathcal{Z}$ is λ-connected. If so, then we are done since $\mathcal{Z}$ is a deletion set for the graph. Otherwise, we know that $G^* - \mathcal{Z}$ contains a violating cut. Lemma 5.2 implies that such a violating cut cannot be of Type 1. Hence, we compute in polynomial time (see Lemma 5.3) a violating triple (X, i, j) in the graph G_i^* for some $i \in [\eta]$. We now invoke Lemma 5.1 with the resulting decomposition to compute an irrelevant edge $e \in E(G) \setminus \mathcal{R}$ in polynomial time and return it. This completes the proof of the lemma. □

6 Vertex Connectivity in Digraphs

In this section we consider the vertex-connectivity variant of the survivable network design problem with uniform demands in digraphs. Our proof strategy for this problem will be to transform the input digraph G into another digraph $G^\star$ such that vertex cuts in G correspond to "special edge cuts" in $G^\star$. Then we follow the same strategy as for edge connectivity versions in graphs and digraphs, but concentrate only on the special edge cuts in $G^\star$, which are a subset of all the edge cuts in the graph. We start by defining the transformation that we will do on the input digraph to transform vertex cuts to special kind of edge cuts.

DEFINITION 6.1. *For a digraph G, the **splitted digraph** $G^\star$ is the digraph obtained from G by using the vertex splitting procedure on all vertices.*

DEFINITION 6.2. *For a digraph G, a **special-cut** $(X, \overline{X})$ is a cut in the splitted digraph $G^\star$, where there exist a vertex $u^+ \in X$ and a vertex $v^- \in \overline{X}$ such that $u \neq v$ and there is no edge from u to v in G.*

From Mengers theorem [3] the following lemma holds.

LEMMA 6.1. *For a digraph G the two following statements are equivalent:*

- *G is κ-vertex connected for $\kappa < n - 1$*
- *Every special-cut in $G^\star$ has least κ directed edges crossing it.*

By Lemma 6.1 it follows that a minimum special-cut corresponding to a κ-vertex connected digraph is of value at least κ.

COROLLARY 6.1. *A digraph G is κ-vertex connected if and only if the minimum value of a special-cut in $G^\star$ is at least κ.*

We now define deletable and undeletable sets, in the same way as we defined for the edge-connectivity-problems.

DEFINITION 6.3. *For a κ-vertex connected digraph G, an arc $e \in A(G)$ is **deletable** if $G - e$ is κ-vertex connected, **undeletable** otherwise. Let $\mathsf{del}(G) = \{e \in E(G) | \, e$ is deletable$\}$ and $\mathsf{undel}(G) = \{e \in E(G) | \, e$ is undeletable$\}$.*

With the above definitions and results in hand, we may now follow an approach similar to the edge-connectivity version of the problem. Consider an instance (G, k) of p-κ-VCS. Clearly a solution F to (G, k) must be a subset of $\mathsf{del}(G)$. The following lemma states that if $|\mathsf{del}(G)| \geq \kappa k^2$ then we can compute a solution of cardinality at least k in polynomial time.

LEMMA 6.2. *Let (G, k) be an instance of p-κ-VCS. Then in polynomial time we can either find a solution of cardinality k or conclude that $|\mathsf{del}(G)| < \kappa k^2$.*

It follows that if $|\mathsf{del}(G)| < \kappa k^2$, then we can test whether (G, k) is a yes-instance of p-κ-VCS by enumerating all subsets $S \subseteq \mathsf{del}(G)$ of size k and testing whether $G - S$ is κ-vertex connected or not. This leads to the main result of this section.

THEOREM 6.1. p-WEIGHTED κ-VCS *can be solved in time $2^{\mathcal{O}(k(\log\kappa+\log k))}n^{\mathcal{O}(1)}$.*

We can extend this algorithm to the weighted version of the problem, in the same manner as in edge-connectivity version of the problem.

7 Intractability of Survivable Network Design Problem

In this section a digraph $D = (V, A)$ with edge-connectivity (vertex-connectivity) function $R : V \times V \to \mathbb{N}$ is said to be $\mathcal{R}$-edge connected ($\mathcal{R}$-strong) if for every pair $u, v \in V(D)$ there is $r(u, v)$ arc-disjoint (vertex-disjoint) paths from u to v in D. Recall that, in SNDP, given a $\mathcal{R}$-edge connected ($\mathcal{R}$-strong) weighted digraph the objective is to find a minimum weighed digraph H, which is $\mathcal{R}$-edge connected ($\mathcal{R}$-strong). A parameterization of the problems is as follows:

(V)SNDP

Input:	A digraph $D = (V, A)$ which is $\mathcal{R}$-edge connected ($\mathcal{R}$-strong) with $w : A \to \mathbb{R}$, $\alpha \in \mathbb{R}$ and an integer k.
Parameter:	k, α
Problem:	Does there exist a set $F \subseteq A$ such that $\lvert F\rvert \leq k$, $w(F) \geq \alpha$ and $D - F$ is $\mathcal{R}$-edge connected ($\mathcal{R}$-strong)?

To show that SNDP is W[1]-hard we first show that the following similar problem on unweighted digraphs is W[1]-hard.

(VERTEX) UNWEIGHTED SNDP ((V)USNDP)

Input:	A digraph $D = (V, A)$ which is $\mathcal{R}$-edge connected ($\mathcal{R}$-strong) and an integer k.
Parameter:	k, α
Problem:	Does there exist a set $F \subseteq A$ such that $\lvert F\rvert \geq k$ and $D - F$ is $\mathcal{R}$-edge connected ($\mathcal{R}$-strong)?

To prove that USNDP is W[1]-hard we give a parameter-preserving reduction from INDEPENDENT SET to USNDP. We first define the INDEPENDENT SET problem.

INDEPENDENT SET

Input:	An undirected graph G and an integer k.
Parameter:	k
Problem:	Does there exist a set $I \subseteq V(G)$ such that $\lvert I\rvert \geq k$ and I is an independent set?

It is well known that INDEPENDENT SET is W[1]-hard [11]. We will use $[q]$ to denote the set $\{1, \ldots, q\}$.

THEOREM 7.1. USNDP *is* W[1]-hard.

Proof. Let (G, k) be an instance of INDEPENDENT SET. Without loss of generality, let $v_1, v_2, \ldots, v_n$ denote the vertices in $V(G)$ and $e_1, \ldots, e_m$ be the arcs in $E(G)$. We make an instance (D, R, k) of USNDP as follows.

1. Set $A(D) = \emptyset$. Make a vertex s and set $V(D) = \{s\}$.
2. For each $v_i \in V(G)$ add a new vertex u_i to $V(D)$ and add an arc $a_i = (s, u_i)$ to $A(D)$.
3. For each $e_j = (v_q, v_p) \in E(G)$ add a new vertex t_j to $V(D)$ and add two arcs $b_{q,j} = (u_q, t_j)$ and $b_{p,j} = (u_p, t_j)$ to $A(D)$.

4. Make a function $R : V(D) \times V(D) \to \mathbb{N}$ with the following values:
   (a) Set $R(s, t_j) = 1$ for all $j \in [m]$.
   (b) Set $R(u_i, t_j) = 1$ for all $i \in [n]$, $j \in [m]$ where $b_{i,j} \in A(D)$
   (c) Set all remaining values of R to 0.

In the digraph D it will be possible to obtain the following observation using the fact that $R(u_i, t_j) = 1$ if $b_{i,j} \in A(D)$ and this arc will be part of every path from u_i to t_j.

Observation 7.1. *The only arcs that can be deleted from D such that D remains $\mathcal{R}$-edge connected belong to a_i for $i \in [n]$.*

For every t_j, $j \in [m]$, there are exactly two paths from s to t_j, $P_1 = [s, u_q, t_j]$ and $P_2 = [s, u_p, t_j]$ for some u_q and u_p. We know that $R(s, t_j) = 1$ and therefore, at most one of the arcs $a_q = (s, u_q)$ and $a_p = (s, u_p)$ can be deleted from D such that D remains $\mathcal{R}$-edge connected.

Let T_i denote the set of vertices, t_j, such that u_i has an arc to it. If a_i is removed from D then every a_p for which u_p has an arc in T_i cannot be deleted. Let those arcs be denoted A_i. Observe that if none of the arcs in A_i are deleted from D, then it is possible to delete a_i from D such that even after this deletion D is $\mathcal{R}$-edge connected.

Observation 7.2. *If a_i is deleted from D, then none of the arcs in A_i can be deleted from D. Furthermore, if none of the arcs in A_i is deleted from D, then removing a_i leaves D $\mathcal{R}$-edge connected.*

The digraph D is build in such a way that there is a one-to-one corresponds between the vertex $v_i \in V(G)$ and the arc $a_i \in A(D)$ and the vertex $u_i \in V(D)$. Furthermore, for every vertex $v_i \in V(D)$, the vertex $u_i \in V(D)$ have an outgoing arc to $t_j \in V(D)$ exactly when v_i is endpoint of $e_j \in E(G)$. This leads to the following observation:

Observation 7.3. *There is an arc $(v_p, v_i) \in E(G)$ if and only if $a_p \in A_i$.*

From Observation 7.1, 7.2 and 7.3 it is possible to show the following claim.

Claim 1. *(G, k) is a yes-instance of* Independent Set *if and only if (D, R, k) is a yes-instance of* USNDP.

Proof. First assume that there exist a solution to the instance (D, R, k) and let F be a solution. Let F' consist of k arcs from F. By Observation 7.1 F' is a subset of the arcs a_i for $i \in [n]$. Set $S = \{v_i \in V(D) \mid a_i \in F'\}$, then for all $a_i \in F'$ we have that $A_i \cap F = \emptyset$ by Observation 7.2. By Observation 7.3 there cannot exists a pair of vertices $v_p, v_q \in S$ such that $(v_p, v_q) \in E(G)$. It means that S is a solution to (G, k).

In the other direction let S be an independent set of size at least k of G. Set $F = \{a_i \in A(D) \mid v_i \in S)$. By Observation 7.3 and the fact that S is an independent set, a pair $a_i, a_j \in F$ such that $a_j \in A_i$ or $a_i \in A_j$ cannot exist. By Observation 7.2 D is $\mathcal{R}$-edge connected if a_i is removed but all the arcs in A_i remains in D. It means that $D \setminus F$ is $\mathcal{R}$-edge connected. Hence F is a solution to (D, R, k). □

From Claim 1 the proof of the theorem follows. □

With small modifications to the construction of D in the proof of Theorem 7.1 it is possible to show the following result.

Theorem 7.2. VUSNDP, SNDP *and* VSNDP *are* W[1]-hard.

8 Conclusion

In this paper, we studied the edge and vertex connectivity version of SNDP with Uniform Demands. We obtain new structural results on λ-connected graphs and digraphs, which could be of independent interest. These results lead to FPT algorithms for these problems with general weights, and a polynomial compression of the unweighted version.

In a recent work, Gutin et al. [17] have applied the techniques of this paper to show that given an undirected graph G and a positive integer k, in FPT time, one can decide whether one can delete at least k edges from G such that even after their removal, the graph G is 2-vertex connected. This provides promising evidence that the techniques developed in this paper might have applications for other variants of the Survivable Network Design Problem.

Our paper opens up several new avenues of research, especially in parameterized complexity. We conclude with a few open problems and future research directions.

(a) Is there an algorithm for Survivable Network Design Problem with Uniform Demands running in time $c^k n^{\mathcal{O}(1)}$ for any fixed value of λ?

(b) What is the parameterized complexity of the problem when we are only interested in the connectivity of a given subset of terminals (say T)? This generalizes well studied problems such as Steiner Tree and Strongly Connected Steiner Subgraph.

(c) In the context of the above problem, is there a relation between the total number of deletable edges and the cardinality of the largest deletion set?

(d) The same questions may be asked with respect to vertex connectivity of the graph.

(e) Finally, what is the parameterized complexity of the deletion version of the Survivable Network Design problem in its full generality on undirected graphs. This problem is likely to be quite difficult and resolving its complexity could very well require the development of new algorithmic tools and techniques.

References

[1] S. Assadi, S. Khanna, Y. Li, and V. Tannen, *Dynamic sketching for graph optimization problems with applications to cut-preserving sketches*, in 35th IARCS Annual Conference on Foundation of Software Technology and Theoretical Computer Science (FSTTCS), 2015, pp. 52–68.

[2] J. Bang-Jensen and G. Z. Gutin, *Digraphs: theory, algorithms and applications*, Springer Science & Business Media, 2008.

[3] J. Bang-Jensen and G. Z. Gutin, *Digraphs: Theory, Algorithms and Applications*, Springer Publishing Company, Incorporated, 2nd ed., 2008.

[4] J. Bang-Jensen and A. Yeo, *The minimum spanning strong subdigraph problem is fixed parameter tractable*, Discrete Applied Mathematics, 156 (2008), pp. 2924–2929.

[5] M. Basavaraju, F. V. Fomin, P. Golovach, P. Misra, M. Ramanujan, and S. Saurabh, *Parameterized algorithms to preserve connectivity*, in Proceedings of the 41st International Colloquium on Automata, Languages, and Programming (ICALP), Springer, 2014, pp. 800–811.

[6] M. Basavaraju, P. Misra, M. S. Ramanujan, and S. Saurabh, *On finding highly connected spanning subgraphs*, CoRR, abs/1701.02853 (2017).

[7] R. Bellman, *Dynamic programming treatment of the travelling salesman problem*, Journal of the ACM (JACM), 9 (1962), pp. 61–63.

[8] A. Bjorklund, *Determinant sums for undirected hamiltonicity*, SIAM Journal on Computing, 43 (2014), pp. 280–299.

[9] T. Chakraborty, J. Chuzhoy, and S. Khanna, *Network design for vertex connectivity*, in Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC), 2008, pp. 167–176.

[10] J. Chuzhoy and S. Khanna, *An $O(k^3 \log n)$-approximation algorithm for vertex-connectivity survivable network design*, Theory of Computing, 8 (2012), pp. 401–413.

[11] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh, *Parameterized Algorithms*, Springer Science & Business Media, 2015.

[12] M. Cygan, S. Kratsch, and J. Nederlof, *Fast hamiltonicity checking via bases of perfect matchings*, in Proceedings of the 45th annual ACM symposium on Theory of computing (STOC), 2013, pp. 301–310.

[13] R. G. Downey and M. R. Fellows, *Parameterized complexity*, Springer Science & Business Media, 2012.

[14] F. V. Fomin, D. Lokshtanov, and S. Saurabh, *Efficient Computation of Representative Sets with Applications in Parameterized and Exact Algorithms*, 2014, ch. 10, pp. 142–151.

[15] A. Frank, *Connections in Combinatorial Optimization*, Oxford Lecture Series in Mathematics and Its Applications, Oxford University Press, Oxford, 2011.

[16] M. R. Garey and D. S. Johnson, *Computers and intractability: a guide to NP-completeness*, WH Freeman New York, 1979.

[17] G. Gutin, M. S. Ramanujan, F. Reidl, and M. Wahlström, *Path-contractions, edge deletions and connectivity preservation*, to appear in ESA 2017, also see abs/1704.06622 (2017).

[18] G. Gutin and A. Yeo, *Constraint satisfaction problems parameterized above or below tight bounds: A survey*, in The Multivariate Algorithmic Revolution and Beyond - Essays Dedicated to Michael R. Fellows on the Occasion of His 60th Birthday, vol. 7370 of Lecture Notes in Computer Science, Springer, 2012, pp. 257–286.

[19] M. Held and R. M. Karp, *A dynamic programming approach to sequencing problems*, Journal of the Society for Industrial and Applied Mathematics, 10 (1962), pp. 196–210.

[20] K. Jain, *A factor 2 approximation algorithm for the generalized steiner network problem*, Combinatorica, 21 (2001), pp. 39–60.

[21] S. Khuller, *Approximation algorithms for finding highly connected subgraphs*, in Approximation algorithms for NP-hard problems, PWS Publishing Co., 1996, pp. 236–265.

[22] G. Kortsarz, R. Krauthgamer, and J. R. Lee, *Hardness of approximation for vertex-connectivity network design problems*, SIAM Journal on Computing, 33 (2004), pp. 704–720.

[23] G. Kortsarz and Z. Nutov, *Approximating minimum cost connectivity problems*, in Dagstuhl Seminar Proceedings, Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2010.

[24] M. Mahajan, V. Raman, and S. Sikdar, *Parameterizing above or below guaranteed values*, Journal of Computer and System Sciences, 75 (2009), pp. 137–153.

[25] D. Marx and L. A. Végh, *Fixed-parameter algorithms for minimum-cost edge-connectivity augmentation*, ACM Transactions on Algorithms (TALG), 11 (2015), p. 27.

[26] D. M. Moyles and G. L. Thompson, *An algorithm for finding a minimum equivalent graph of a digraph*, Journal of the ACM (JACM), 16 (1969), pp. 455–460.
[27] H. Nagamochi and T. Ibaraki, *Algorithmic aspects of graph connectivity*, vol. 123, Cambridge University Press, New York, 2008.