Paths of Bounded Length and Their Cuts: Parameterized Complexity and Algorithms

Petr A. Golovach[1] and Dimitrios M. Thilikos[2]

[1] Department of Informatics, University of Bergen, PB 7803, 5020 Bergen, Norway.
[2] Department of Mathematics, National and Kapodistrian University of Athens, Panepistimioupolis, GR15784 Athens, Greece.[⋆⋆⋆]

Abstract. We study the parameterized complexity of two families of problems: the *bounded length disjoint paths* problem and the *bounded length cut* problem. From Menger's theorem both problems are equivalent (and computationally easy) in the unbounded case for single source, single target paths. However, in the bounded case, they are combinatorially distinct and are both NP-hard, even to approximate. Our results indicate that a more refined landscape appears when we study these problems with respect to their parameterized complexity. For this, we consider several parameterizations (with respect to the maximum length l of paths, the number k of paths or the size of a cut, and the treewidth of the input graph) of all variants of both problems (edge/vertex-disjoint paths or cuts, directed/undirected). We provide FPT-algorithms (for all variants) when parameterized by both k and l and hardness results when the parameter is only one of k and l. Our results indicate that the bounded length disjoint-path variants are structurally harder than their bounded length cut counterparts. Also, it appears that the edge variants are harder than their vertex-disjoint counterparts when parameterized by the treewidth of the input graph.

Keywords: Bounded length disjoint paths, Bounded length cuts, Parameterized Complexity, Parameterized Algorithms.

1 Introduction

We consider finite (directed and undirected) graphs without loops or multiple edges. The vertex set of a graph G is denoted by $V(G)$ and its edge set by $E(G)$. We denote undirected edges by $\{u, v\}$, and directed edges by (u, v). Given a graph G and a set $F \subseteq E(G)$ (resp. $X \subseteq V(G)$), we denote by $G \setminus F$ (resp. $G \setminus X$) the graph obtained by G if we remove from it all edges in F (resp. vertices in X).

One of the most celebrated problems in discrete algorithms and combinatorial optimization is the disjoint paths problem. Its algorithmic study dates back to

[⋆⋆⋆] Supported by the project "Kapodistrias" (AΠ 02839/28.07.2008) of the National and Kapodistrian University of Athens (project code: 70/4/8757).

Menger's Theorem [25] (see also [9]), was extended by the work of Ford and Fulkerson [15] on network flows, and now constitutes (along with its variants) a central algorithmic problem in algorithm design.

According to Menger's theorem, given a graph G and two terminals $s, t \in V(G)$, the maximum number of vertex-disjoint (s,t)-paths in G is *equal* to the minimum cardinality of a set of vertices in $V(G) \setminus \{s,t\}$ meeting all (s,t)-paths of G. Interestingly, it appears that such a min-max equality does not hold if we restrict paths to be of bounded length. This was observed for the first time by Adámek and Koubek in [1]. Lovász, Neumann Lara, and Plummer proved in [24] that a similar min-max relation holds only for path lengths equal to 2,3, or 4. Analogous results were provided for the case where the paths are edge-disjoint in [11] and [27]. We present below the main decision versions of the problems generated by the bounded length restriction.

We need some definitions. Let G be a graph, $s, t \in V(G)$, and let l be a positive integer. We call set $F \subseteq E(G)$ (resp. $X \subseteq V(G) \setminus \{s,t\}$) (s,t)-*edge* (resp. *vertex*) l-*bounded cut* if $G \setminus F$ (resp. $G \setminus X$) contains no (s,t)-path of length at most l.

Bounded Edge Directed (s,t)-disjoint Paths (BEDP)

Input: A directed graph G, two positive integers k, l and two distinct vertices s, t of G.
Question: Are there k edge-disjoint (s,t)-paths each of length at most l in G?

Bounded Edge Directed (s,t)-Cut (BEDC)

Input: A directed graph G, two positive integers k, l and two distinct vertices s, t of G.
Question: Is there a (s,t)-edge l-bounded cut $F \subseteq E(G)$ of size at most k?

Also, the first of the above problems has been extended to its multi-terminal version as follows.

Bounded Edge Directed Multi-terminal disjoint Paths (BEDMP)

Input: A directed graph $G = (V, E)$, two positive integers k, l, and two sequences $S = (s_1, \ldots, s_k)$ (sources), $T = (t_1, \ldots, t_k)$ (targets) of vertices in G.
Question: Are there k edge-disjoint (s_i, t_i)-paths of length at most l in G for $i = 1, \ldots, k$?

Similarly to the above, we can define numerous variants depending whether the graph is directed or undirected, and whether the paths are edge-disjoint or internally vertex-disjoint. All variants and the corresponding notation are depicted in Table 1.

For all multi-terminal disjoint paths problems we can assume that all terminals are pair-wise distinct, since otherwise we can apply the following rule:

Rule (1): for every vertex v that corresponds to r terminals we first subdivide all its incident edges and then replace v by r vertices (each with one of the

	Multi-terminal Disjoint Paths		(s,t)-disjoint Paths		(s,t)-Cut	
	Directed	Undirected	Directed	Undirected	Directed	Undirected
Edge	BEDMP	BEUMP	BEDP	BEUP	BEDC	BEUC
Vertex	BVDMP	BVUMP	BVDP	BVUP	BVDC	BVUC

Table 1. Bounded length variants of the edge-disjoint paths problem and the edge cut problem.

terminals corresponding to v) that have the same neighborhood as v (in the directed case, replacement edges maintain their original directions). The new graph contains k edge-disjoint paths of length at most $l+2$ iff the original one contains k vertex-disjoint paths of length at most l.

The first algorithmic results for the above problems appeared by Itai, Perl and Shiloach in [19] where they proved that BVUP and BEUP are polynomially solvable for path lengths 2, 3 or 4, while they become NP-complete for length values bigger than 4. In the same paper they proved that if, instead of fixing the length l, we fix the number k of paths, the problem is still NP-complete even for 2 paths. Results on the fractional versions of these problems were given in [13, 21, 22]. The approximability of these problems was studied in [6, 17] and [4, 5]. Finally, for some applications of the above problems, see [29, 31] and [18].

Some results for the multi-terminal variants of the bounded-length disjoint paths problem were given in [17]. We just stress that, when there is no restriction to the length of the paths, BVUMP is NP-complete in general [20] and polynomially solvable for fixed k [28], while BVDMP is NP-complete even when $k=2$ [16].

In this paper, we provide a detailed study of the parameterized complexity of all the bounded length variants of the problems in Table 1. In a parameterized problem we distinguish some part of the input to be its parameter. Typically, a parameter is an integer, k, related to the problem input and the question is whether the problem can be solved by an algorithm (called FPT-*algorithm*) of time complexity $f(k)\cdot n^{O(1)}$ where n is the size of the input and f is a (super-polynomial) function depending only on the parameter (instead of worst time complexities such as $O(n^{f(k)})$ or $O(k^{f(n)})$). When a parameterized problem admits an FPT-algorithm, then it belongs in the parameterized complexity class FPT. Not all parameterized problems belong in FPT. There are several parameterized complexity classes, such as W[1], W[2], para-NP and analogous notions of hardness with respect to parameter-preserving reductions, able to prove that membership in FPT is rather non-possible (for more details, see the monographs [10, 14, 26]). Briefly, if a parameterized problem is W[1]-hard, this means that a complexity of type $O(n^{f(k)})$ is the best we may expected, while if a parameterized problem is para-NP-hard, then we cannot even hope for something better than a $k^{f(n)}$-algorithm.

Table 2 indicates the existence of reductions between all considered problems. Here $\Pi_1 \leq^{(i)} \Pi_2$ means that the problem Π_1 can be reduced to the problem Π_2

by the reduction rule i. The edge undirected versions are reduced to the vertex undirected ones by the following rule:

Rule (2): Take the line graph L_G of G and for every clique K of L_G, corresponding to the edges incident to a terminal v of G, add a new terminal vertex v' and connect it with all the vertices of the clique. Vertex-disjoint paths of length $l+1$ in the new graph correspond to edge-disjoint paths of length l in the original graph, while it trivially follows that edge cuts become vertex cuts.

Certainly, vertex undirected versions are reduced to vertex directed ones by the following obvious rule:

Rule (3): Replace every edge by two opposite direction edges.

The following rule reduces all vertex directed versions to their edge directed counterparts:

Rule (4): Replace every non-terminal vertex v by a directed edge (v_t, v_h) (we call such edges *new edges*) and make v_t the head of all previous edges whose head was v and v_h the tail of all previous edges whose tail was v. Notice that every path of length at most $2l-1$ in the new graph corresponds to a path of length at most l in the original graph and edge-disjoint paths in the new graph correspond to vertex-disjoint paths in the original graph and vice versa. This proves the correctness of Rule (4) for disjoint path problems. For cut problems we additionally observe that every vertex cut of the original graph correspond to an edge cut in the new graph. For the inverse direction, take an edge cut of the new graph and replace each non-new edges e in it with some new edge that has a common endpoint with e. This makes every edge cut in the new graph to correspond to a vertex cut in the original graph.

Notice that all rules are parameter-preserving when the parameter is k, l, or both.

BEUMP	$\leq^{(2)}$	BVUMP	$\leq^{(3)}$	BVDMP	$\leq^{(4)}$	BEDMP
$\vee\!\vert^{(1)}$		$\vee\!\vert^{(1)}$		$\vee\!\vert^{(1)}$		$\vee\!\vert^{(1)}$
BEUP	$\leq^{(2)}$	BVUP	$\leq^{(3)}$	BVDP	$\leq^{(4)}$	BEDP
BEUC	$\leq^{(2)}$	BVUC	$\leq^{(3)}$	BVDC	$\leq^{(4)}$	BEDC

Table 2. Reductions between problems

All problems in Table 1 have two possible parameters k and l in their inputs. Therefore, we consider parameterizations of them with respect to l, k, or both, indicating which parameterization we pick in each problem. For example, the BEUP problem is denote as BEUP(k) when parameterized by the number of paths k, BEUP(l) when parameterized by the maximum length l of a path and BEUP(k, l) when parameterized by both these quantities. We follow the same notation for all problems in Table 1.

We prove that all variants of our problems are in FPT when parameterizing on both k and l.

All problems we consider are NP-hard for fixed values of l, bigger than some constant [4, 19]. Using standard terminology from [14], this means that all of them, parameterized by l are para-NP-complete (i.e. they are NP-hard even for fixed values of the parameter). Moreover, the problem asking for the existence of two edge-disjoint paths between two terminals of an undirected graph is also NP-complete even for two paths, because of the results [30, 23]. This implies the para-NP-completeness of all the disjoint paths variants when parameterized by k. However, no similar result can be expected (unless P=NP) for bounded cut problems, as they trivially admit an $n^{O(k)}$-step algorithm (just check all possible cuts of size at most k). It appears that this running time substantially cannot become better: we prove that these four variants are W[1]-hard (Theorem 4) and that for the directed graph variants, this holds even for DAGs (Theorem 3). This indicates that, apart from the combinatorial discrepancy between problems on paths and problems on cuts, there is also a discrepancy on the parameterized complexities of the corresponding problems. We stress that this distinction cannot be made clear by studying the classic complexity of the two families of problems (they are all NP-complete in general). Our results are depicted in Table 1.

	l	k, l	k
BEDMP BVDMP BVUMP BEUMP	para-NP-c [19]	FPT $O(2^{O(kl)} \cdot m \cdot \log n)$ (Th. 1)	para-NP-c [30, 23]
BEDP BVDP BVUP BEUP	para-NP-c [19]	FPT $O(2^{O(kl)} \cdot m \cdot \log n)$ (Th. 1)	para-NP-c [30, 23]
BEDC BVDC BVUC BEUC	para-NP-c [4]	FPT $O(l^k \cdot m)$ (Th. 2)	W[1]-h for DAGs (Th. 3) W[1]-h for DAGs (Th. 3) W[1]-h (Th. 4) W[1]-h (Th. 4)

Table 3. Summary of our results when parameterizing by l, k, l and k.

Our next step is to study the (in general para-NP-complete) parameterized problems BVDP(l) and BVUP(l) for the special case where their input graphs are sparse. We prove (Theorem 5) that both problems admit FPT-algorithms for classes of graphs that have bounded local treewidth (typical graph class with bounded local treewidth are planar graphs or bounded-degree graphs). Moreover, this result can be extended for classes of graphs where the removal of at most one vertex includes them in some bounded local treewidth class. On the other side, we prove that this sparsity criterion cannot be relaxed: BVDP(l)

(BVUP(l)) remains para-NP-complete (Theorem 6) for $l \geq 6$, on undirected (directed acyclic) graphs that can be made planar after removing 2 vertices (we call these graphs *2-apex-graphs*). We also prove that the same holds for the edge variants of the same problems (Theorem 7) for $l \geq 7$. Our results suggest a rapid change on the problem complexity with respect to the minor-exclusion sparsity criterion.

	l, [bounded **ltw**]	l, [two-apex]	**tw**, l
BVUP	FPT (Th. 5)	para-NP-c (Th. 6), $l \geq 6$	FPT (Th. 5)
BVDP	FPT (Th. 5)	para-NP-c for DAGs (Th. 6), $l \geq 6$	FPT (Th. 5)
BEUP	open	para-NP-c (Th. 7), $l \geq 7$	W[1]-h for fixed $l \geq 10$ (Th. 8)
BEDP	open	para-NP-c for DAGs (Th. 7), $l \geq 7$	W[1]-h for DAGs for fixed $l \geq 10$ (Th. 8)

Table 4. Summary of our results for sparse graph families.

Our last result concerns the case where BEUP and BEDP are parameterized by the treewidth of their input graphs. We prove that BEUP is W[1]-hard when parameterized by the treewidth of the input graph and that BEDP is W[1]-hard when parameterized by the treewidth of the underlying graph of its input graph even when the input graph is acyclic (Theorem 8). This last result indicates that the edge-disjoint variants are harder than the vertex-disjoint ones (the same parameterization leads to an FPT-algorithm for BVUP and BVDP – Theorem 5). Our results on sparse graph classes are summarized in Table 4.

2 Parameterized algorithms

2.1 An FPT-algorithm for BEDMP($\boldsymbol{k}, \boldsymbol{l}$)

Our algorithm for the BEDMP(k, l) is based on the color-coding technique introduced by Alon, Yuster and Swick in [2]. In particular, we consider a family $\mathcal{F}$ of hash functions, each mapping $\{1, \ldots, m\}$ to a set of colors $\{1, \ldots, k \cdot l\}$, such that for every $S \subseteq \{1, \ldots, m\}$, where $|S| \leq k \cdot l$, there is a $f \in \mathcal{F}$ such that its restriction to S is a bijection. As mentioned in [2], such a family where $|\mathcal{F}| = 2^{O(k \cdot l)} \cdot \log m$ can be constructed in $2^{O(k \cdot l)} \cdot m \cdot \log m$ steps.

Let $\mathcal{F}$ be a family of hash functions as above where $\{1, \ldots, m\}$ represent the edges of G. Let also $\chi \in \mathcal{F}$. Given an integer $i \in \{1, \ldots, k\}$, we define a Boolean function B_i^{χ} such that, for every set of colors $X \subseteq \{1, \ldots, k \cdot l\}$, $B_i^{\chi}(X)$ is true if and only if there exists a collection of i paths $P_1, \ldots, P_i$ of length at most l where, for $j \in \{1, \ldots, i\}$, the endpoints of P_j are s_j and t_j and such that the set of the colors assigned to the edges of these paths is a subset of X (i.e. $\chi^{-1}(\cup_{j \in \{1, \ldots, i\}} E(P_j)) \subseteq X$). Notice that an instance of BEDMP(k, l) is a YES-instance if and only if there is a $\chi \in \mathcal{F}$ such that $B_k^{\chi}(\{1, \ldots, k \cdot l\}) = \mathsf{true}$.

In general, to compute $B_i^{\chi}(X)$ for some $X \subseteq \{1, \ldots, k \cdot l\}$, we observe that

$$B_i^{\chi}(X) = \bigvee_{Y \subseteq X} (B_{i-1}^{\chi}(Y) \wedge C_i^{\chi}(X \setminus Y))$$

where C_i^{χ} is a Boolean function such that if $S \subseteq \{1, \ldots, k \cdot l\}$ the value of $C_i^{\chi}(S)$ is true if and only if the subgraph of G induced by the edges colored by colors in S contains a path between s_i and t_i of length at most l. Notice that C_i^{χ} can be computed in $O(m)$ steps. Moreover, computing $B_i^{\chi}(X)$ for all $X \subseteq \{1, \ldots, l \cdot k\}$ requires $O(3^{k \cdot l} \cdot m)$ steps. Therefore, the above dynamic programming requires in total $O(3^{k \cdot l} \cdot m \cdot k)$ steps to compute $B_k^{\chi}(\{1, \ldots, k \cdot l\})$. Concluding BEDMP$(k, l)$ can be solved in $O(2^{O(k \cdot l)} \log m \cdot m \cdot k)$ steps.

It is easy to observe that the above algorithm can be modified so that it also would return the requested paths when exist. We conclude to the following.

Theorem 1. *The* BEDMP(k, l) *problem (as well as* BVDMP(k, l)*,* BVUMP(k, l)*, and*
BEUMP(k, l)*) can be solved by an* FPT*-algorithm that runs in* $O(2^{O(k \cdot l)} \log n \cdot m \cdot k)$ *steps where* $n = |V(G)|$ *and* $m = |E(G)|$*.*

2.2 An FPT-algorithm for BEDC$(\boldsymbol{k}, \boldsymbol{l})$

The proof of the following theorem is based on the simple observation that for any (s, t)-path of length at most l, at least one edge of it has to be included to any (s, t)-edge l-bounded cut.

Theorem 2. *The* BEDC(k, l) *problem can be solved by an* FPT*-algorithm that runs in* $O(l^k \cdot m)$ *time where* $n = |V(G)|$ *and* $m = |E(G)|$*.*

3 Hardness results for $(\boldsymbol{s}, \boldsymbol{t})$-cuts

In this section we prove W[1]-hardness of BVDC(k) and BEUC(k). It can be noted that by the reduction rules (see Table 2) W[1]-hardness of BVDC(k) follows from a similar result for BEUC(k), but we prove here a stronger result.

Theorem 3. BVDC(k) *problem is* W[1]*-hard even for acyclic digraphs.*

Proof. We present a reduction from the Multicolored Clique problem:
Multicolored Clique

Input: A graph G with a proper k-coloring of G.
Question: Is there a clique of size k in G containing exactly one vertex from each color class?

The Multicolored Clique problem, parameterized by k, was proved to be W[1]-hard by Fellows et al. [12].

Let G be an n-vertex undirected graph. Denote by X_i the i-th color class in the given k-coloring of G. Assume without loss of a generality that $k \geq 4$. We assume also that for any pair of sets X_i, X_j, $i \neq j$, vertices of these sets are connected by the same number of edges denoted by m, and $m > 0$ (otherwise it is possible to add pairs of adjacent vertices to the graph to ensure this condition). Denote by $e_1^{(i,j)}, e_2^{(i,j)}, \ldots, e_m^{(i,j)}$ the edges which join sets X_i and X_j. Let $l = 5m + 4$.

Now we consider auxiliary constructions. For every $i, j \in \{1, 2, \ldots, k\}$, $i \neq j$, a directed graph $F_{i,j}$ is constructed as follows.

1. Two vertices s and t are created.
2. For every $r \in \{1, 2, \ldots, m\}$, vertices u_r, $a_r^{(1)}, a_r^{(2)}, a_r^{(3)}$ and $b_r^{(1)}, b_r^{(2)}, b_r^{(3)}$ are constructed, and for every $r \in \{0, 1, \ldots, m\}$, vertices v_r are introduced. It is assumed, for convenience, that $s = a_0^{(1)} = a_0^{(2)} = a_0^{(3)}$, $b_0^{(1)} = a_m^{(1)}$, $b_0^{(2)} = a_m^{(2)}$, $b_0^{(3)} = a_m^{(3)}$ and $t = b_{m+1}^{(3)} = b_{m+1}^{(3)} = b_{m+1}^{(3)}$.
3. For each vertex u_r, edges $(a_{r-1}^{(1)}, u_r)$, $(a_{r-1}^{(2)}, u_r)$, $(a_{r-1}^{(3)}, u_r)$ and $(u_r, a_r^{(1)})$, $(u_r, a_r^{(2)})$, $(u_r, a_r^{(3)})$ are added.
4. For each vertex v_r, edges $(b_r^{(1)}, v_r)$, $(b_r^{(2)}, v_r)$, $(b_r^{(3)}, v_r)$ and $(v_r, b_{r+1}^{(1)})$, $(v_r, b_{r+1}^{(2)})$, $(v_r, b_{r+1}^{(3)})$ are added.
5. Pairs of vertices $a_{r-1}^{(f)}, a_r^{(f)}$ are joined by paths of length $r + 2$ for $f = 1, 2, 3$ and $r \in \{1, 2, \ldots, m\}$.
6. Pairs of vertices $a_{r-1}^{(f)}, b_r^{(f)}$, $f = 1, 2, 3$, are joined by paths of length $3m + 4$ for $r \in \{1, 2, \ldots, m + 1\}$.
7. Add vertices $w_1^{(i,j)}, w_2^{(i,j)}, \ldots, w_m^{(i,j)}$, and join every vertex v_{r-1} with $w_r^{(i,j)}$ by a path of length $3(m + 1 - r)$.

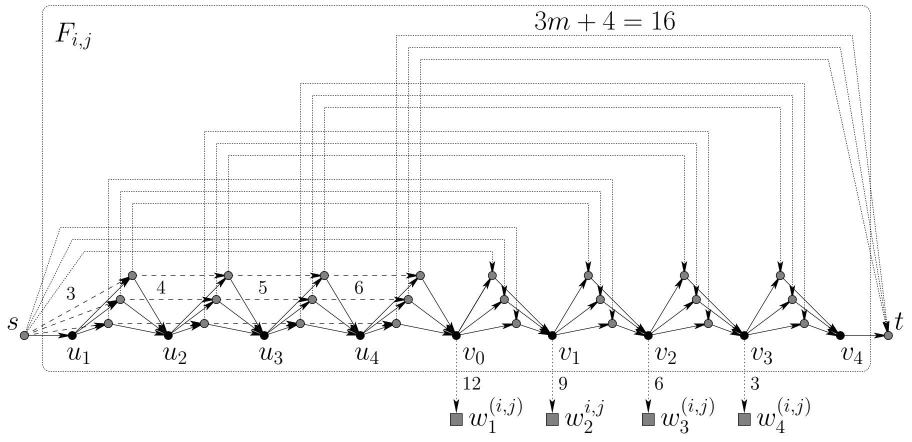

Fig. 1. Construction of $F_{i,j}$ for $m = 4$. Paths are shown by dash lines.

The graph $F_{i,j}$ is shown in Figure 1 for $m = 4$. Using these gadgets $F_{i,j}$ we construct a directed graph H from G as follows.

8. For all pairs $\{i, j\}$, $i, j \in \{1, \dots, k\}$, $i \neq j$, graphs $F_{i,j}$ with common vertices s and t are constructed.

9. Every edge $e_f^{(i,j)} = \{x, y\}$ of G is replaced by two directed edges $(w_f^{(i,j)}, x)$ and $(w_f^{(i,j)}, y)$.

10. For each vertex $x \in V(G)$, an edge (x, t) is added.

It is easy to see that H is a directed acyclic graph. The next claim concludes the proof of the theorem.

Claim. *Graph G has a clique of size k which contains exactly one vertex from any color class if and only if there is (s,t)-vertex l-bounded cut in H with at most $k' = k^2$ vertices.*

Notice that the reduction 4 from BVDC(k) to BEDC(k) transforms a directed acyclic graph into another directed acyclic graph. So, W[1]-hardness of BEDC(k) for DAGs follows immediately. What remains to prove is the W[1]-hardness for the undirected case. The proof of the following theorem is a similar reduction from the Multicolored Clique problem and is omitted here.

Theorem 4. BEUC(k) *is* W[1]*-hard.*

4 (s,t)-paths of bounded length for sparse graphs

4.1 FPT-algorithms for sparse graph classes

A *tree decomposition* of a graph G is a pair (X, T) where T is a tree and $X = \{X_i \mid i \in V(T)\}$ is a collection of subsets (called *bags*) of $V(G)$ such that: 1. $\bigcup_{i \in V(T)} X_i = V(G)$, 2. for each edge $\{x, y\} \in E(G)$, $x, y \in X_i$ for some $i \in V(T)$, and 3. for each $x \in V(G)$ the set $\{i \mid x \in X_i\}$ induces a connected subtree of T. The *width* of a tree decomposition $(\{X_i \mid i \in V(T)\}, T)$ is $\max_{i \in V(T)} \{|X_i| - 1\}$. The *treewidth* of a graph G (denoted as $\mathbf{tw}(G)$) is the minimum width over all tree decompositions of G. For a directed graph G, $\mathbf{tw}(G)$ is the treewidth of the underlying graph.

We say that a graph class $\mathcal{G}$ has *bounded local treewidth with bounding function f* if there is a function $f : \mathbb{N} \to \mathbb{N}$ such that for every graph $G \in \mathcal{G}$, every $v \in V(G)$, and every positive integer i it holds that $\mathbf{tw}(G[N_G^i[v]]) \leq f(i)$ where $N_G^i[v] = \{u \in V(G) \colon \mathrm{dist}_G(u, v) \leq i\}$.

It appears that many sparse classes have bounded local treewidth. Examples are planar graphs and graphs of bounded genus, bounded max-degree graphs, and graphs excluding an apex graph as a minor (an apex graph is a graph that can become planar after the removal of one vertex). The purpose of this subsection is to construct an FPT algorithm for the BVDP(l) and the BVUP(l)

problems when their inputs are restricted to directed graphs whose underlying graphs belong to some (almost) bounded local treewidth graph class.

Theorem 5. *The* BVDP(l) *problems (and therefore, also* BVUP(l)*) can be solved by an* FPT*-algorithm for graph classes that have bounded local treewidth. Moreover, let $\mathcal{G}$ be a bounded local treewidth graph class, and let $\mathcal{G}'$ be a set of all graphs G such that there is a set $X \subseteq V(G)$, $|X| \leq 1$, for which $G \setminus X \in \mathcal{G}$. Then the* BVDP($l$) *can be solved by an* FPT*-algorithm in $\mathcal{G}'$.*

4.2 Vertex-disjoint (s,t)-paths of bounded length for H-minor-free graphs

In this section we show that the restrictions of Theorem 5 are somehow tight. We call a graph G a *two-apex graph* if there is a set X of at most two vertices such that $G \setminus X$ is a planar graph. Due the space restrictions the proof of the following theorem is omitted here.

Theorem 6. BVUP(l) *is* NP*-complete and* BVDP(l) *is* NP*-complete for directed acyclic graphs for any fixed $l \geq 6$ for two-apex graphs.*

Consider the class of K_k-minor free graphs (i.e. none of the graphs in this class contains a subgraph that can be contracted to K_k). Notice that K_5-free graphs have bounded local treewidth. However this is not correct for K_r-minor-free graphs for $r \geq 6$. Since two-apex graphs are K_7-minor-free, Theorem 5 provides a nearly optimal estimation on the tractability of BVUP(l) and BVDP(l) on K_r-minor-free graphs. Actually, the same Theorem argues that not even a $n^{f(k)}$ step algorithm can be found for $r \geq 7$.

We can also prove the following (the proof is similar to the one of Theorem 6 and is omitted here). We consider these theorem separately instead applying reduction rules since these rules do not preserve exact value of the parameter l.

Theorem 7. *The* BEUP(l) *is* NP*-complete and* BEDP(l) *is* NP*-complete for directed acyclic graphs for any fixed $l \geq 7$ for two-apex graphs.*

It is interesting to note that BEUP(l) and BEDP(l) are more difficult than their vertex disjoint counterparts for graphs of bounded treewidth (see Theorem 5). The fact the edge-disjoint variants are harder is also indicated by the results of the following section.

4.3 Edge-disjoint (s,t)-paths of bounded length for graphs of bounded treewidth

By reduction rules (see Table 2) BEUP can be reduced to BVUP, but the reduction 2 does not preserve the treewidth of the graph. The following theorem (the proof of which is omitted here due the space restrictions) shows that vertex-disjoint and edge-disjoint path problems behave very differently when parameterized by the treewidth.

Theorem 8. *For every fixed $l \geq 10$,* BEUP *is* W[1]*-hard, when parameterized by treewidth and* BEDP *is* W[1]*-hard for directed acyclic graphs when parameterized by treewidth of the underlying graph.*

5 Conclusions

A natural question about the parameterized complexity of the variants of the bounded length disjoint path and the bounded length cut problems parameterized by k and l is whether they admit polynomial kernels. In fact, using techniques from [7], we can prove that this is not the case for all the disjoint path variants. We believe that the existence of polynomial kernels for the edge cut variants as well as the planar restrictions of the disjoint path variants is an interesting open problem.

References

1. J. Adámek and V. Koubek, *Remarks on flows in network with short paths*, Commentationes Mathematicae Universitatis Carolinae, 12(4) (1971), pp. 661–667.
2. N. Alon, R. Yuster, and U. Zwick, *Color-coding*, J. Assoc. Comput. Mach., 42 (1995), pp. 844–856.
3. S. Arnborg, J. Lagergren, and D. Seese, *Easy problems for tree-decomposable graphs*, Journal of Algorithms, 12 (1991), pp. 308–340.
4. G. Baier, T. Erlebach, A. Hall, E. Köhler, H. Schilling, and M. Skutella, *Length-bounded cuts and flows*, in Automata, languages and programming. Part I, vol. 4051 of Lecture Notes in Comput. Sci., Springer, Berlin, 2006, pp. 679–690.
5. G. Baier, T. Erlebach, A. Hall, E. Köhler, P. Kolman, O. Pangrác, H. Schilling, and M. Skutella, *Length-bounded cuts and flows*, ACM Transactions in Algorithms, to appear.
6. A. Bley, *On the complexity of vertex-disjoint length-restricted path problems*, Comput. Complexity, 12 (2003), pp. 131–149.
7. H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin, *On problems without polynomial kernels (extended abstract)*, in Automata, languages and programming. Part I, vol. 5125 of Lecture Notes in Comput. Sci., Springer, 2008, pp. 563–574.
8. R. B. Borie, *Generation of polynomial-time algorithms for some optimization problems on tree-decomposable graphs*, Algorithmica, 14 (1995), pp. 123–137.
9. G. B. Dantzig and D. R. Fulkerson, *On the max-flow min-cut theorem of networks*, in Linear inequalities and related systems, Annals of Mathematics Studies, no. 38, Princeton University Press, Princeton, N. J., 1956, pp. 215–221.
10. R. G. Downey and M. R. Fellows, *Parameterized complexity*, Monographs in Computer Science, Springer-Verlag, New York, 1999.
11. G. Exoo, *On line disjoint paths of bounded length*, Discrete Math., 44 (1983), pp. 317–318.
12. M. Fellows, D. Hermelin, F. Rosamond andS. Vialette, *On the parameterized complexity of multiple-interval graph problems*, Theor. Comput. Sci., 410 (2009), pp. 53–61.

13. L. Fleischer and M. Skutella, *The quickest multicommodity flow problem*, in Integer programming and combinatorial optimization, vol. 2337 of Lecture Notes in Comput. Sci., Springer, Berlin, 2002, pp. 36–53.
14. J. Flum and M. Grohe, *Parameterized complexity theory*, Texts in Theoretical Computer Science. An EATCS Series, Springer-Verlag, Berlin, 2006.
15. L. R. Ford, Jr. and D. R. Fulkerson, *Maximal flow through a network*, Canad. J. Math., 8 (1956), pp. 399–404.
16. S. Fortune, J. Hopcroft, and J. Wyllie, *The directed subgraph homeomorphism problem*, Theoret. Comput. Sci., 10 (1980), pp. 111–121.
17. V. Guruswami, S. Khanna, R. Rajaraman, B. Shepherd, and M. Yannakakis, *Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems*, J. Comput. System Sci., 67 (2003), pp. 473–496.
18. D. HSU, *On container width and length in graphs, groups, and networks)*, IEICE transactions on fundamentals of electronics, communications and computer sciences, 77 (19940425), pp. 668–680. Dedicated to Professor Paul Erdős on the occasion of his 80th birthday (Special Section on Discrete Mathematics and Its Applications).
19. A. Itai, Y. Perl, and Y. Shiloach, *The complexity of finding maximum disjoint paths with length constraints*, Networks, 12 (1982), pp. 277–286.
20. R. M. Karp, *On the computational complexity of combinatorial problems*, Networks, 5(1) (1975), pp. 45–68. (Proceedings of the Symposium on Large-Scale Networks, Evanston, IL, USA, 18-19 April 1974).
21. P. Kolman and C. Scheideler, *Improved bounds for the unsplittable flow problem*, in Proceedings of the Symposium on Discrete Algorithms, ACM, 2002, pp. 184–193.
22. P. Kolman and C. Scheideler, *Improved bounds for the unsplittable flow problem*, Journal of Algorithms, 61(1) (2006), pp. 20–44.
23. C.-L. Li, T. McCormick, and D. Simchi-Levi, *The complexity of finding two disjoint paths with min-max objective function*, Discrete Appl. Math., 26 (1990), pp. 105–115.
24. L. Lovász, V. Neumann Lara, and M. Plummer, *Mengerian theorems for paths of bounded length*, Period. Math. Hungar., 9 (1978), pp. 269–276.
25. K. Menger, *Über reguläre Baumkurven*, Math. Ann., 96 (1927), pp. 572–582.
26. R. Niedermeier, *Invitation to fixed-parameter algorithms*, vol. 31 of Oxford Lecture Series in Mathematics and its Applications, Oxford University Press, Oxford, 2006.
27. L. Niepel and D. Šafaříková, *On a generalization of Menger's theorem*, Acta Math. Univ. Comenian., 42/43 (1983), pp. 275–284 (1984).
28. N. Robertson and P. D. Seymour, *Graph minors. XIII. The disjoint paths problem*, Journal of Combinatorial Theory. Series B, 63 (1995), pp. 65–110.
29. D. Ronen and Y. Perl, *Heuristics for finding a maximum number of disjoint bounded paths*, Networks, 14 (1984), pp. 531–544.
30. S. Tragoudas and Y. L. Varol, *Computing disjoint paths with length constraints*, in Graph-theoretic concepts in computer science (Cadenabbia, 1996), vol. 1197 of Lecture Notes in Comput. Sci., Springer, Berlin, 1997, pp. 375–389.
31. D. Wagner and K. Weihe, *A linear-time algorithm for edge-disjoint paths in planar graphs*, Combinatorica, 15 (1995), pp. 135–150.