Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 22, no. 2, pp. 177–191 (2018)
DOI: 10.7155/jgaa.00462

On Algorithms Employing Treewidth for L-bounded Cut Problems

Petr Kolman

Department of Applied Mathematics
Faculty of Mathematics and Physics
Charles University, Prague, Czech Republic

Abstract

Given a graph $G = (V, E)$ with two distinguished vertices $s, t \in V$ and an integer parameter $L > 0$, an L-*bounded cut* is a subset F of edges (vertices) such that the every path between s and t in $G \backslash F$ has length more than L. The task is to find an L-bounded cut of minimum cardinality.

Though the problem is very simple to state and has been studied since the beginning of the 70's, it is not much understood yet. The problem is known to be $\mathcal{NP}$-hard to approximate within a small constant factor even for $L \geq 4$ (for $L \geq 5$ for the vertex–deletion version). On the other hand, the best known approximation algorithm for general graphs has approximation ratio only $\mathcal{O}(n^{2/3})$ in the edge case, and $\mathcal{O}(\sqrt{n})$ in the vertex case, where n denotes the number of vertices.

We show that for planar graphs, it is possible to solve both the edge– and the vertex–deletion version of the problem optimally in $\mathcal{O}((L+2)^{3L}n)$ time. That is, the problem is fixed-parameter tractable (FPT) with respect to L on planar graphs. Furthermore, we show that the problem remains FPT even for bounded genus graphs, a super class of planar graphs.

Our second contribution deals with approximations of the vertex–deletion version of the problem. We describe an algorithm that for a given graph G, its tree decomposition of width τ and vertices s and t computes a τ-approximation of the minimum L-bounded $s-t$ vertex cut; if the decomposition is not given, then the approximation ratio is $\mathcal{O}(\tau\sqrt{\log \tau})$. For graphs with treewidth bounded by $\mathcal{O}(n^{1/2-\epsilon})$ for any $\epsilon > 0$, but not by a constant, this is the best approximation in terms of n that we are aware of.

Submitted:	Reviewed:	Revised:	Reviewed:	Revised:
September 2017	November 2017	December 2017	December 2017	January 2018
	Accepted:	Final:	Published:	
	January 2018	Februray 2018	February 2018	
	Article type:		Communicated by:	
	Regular paper		D. Wagner	

This research was partially supported by project GA15-11559S of GA ČR

E-mail address: petr.kolman@mff.cuni.cz (Petr Kolman)

1 Introduction

The subject of this paper is a variation of the classical $s-t$ cut problem, namely the *minimum L-bounded edge (vertex) cut problem*: given a graph $G = (V, E)$ with two distinguished vertices $s, t \in V$ and an integer parameter $L > 0$, find a subset F of edges (vertices) of minimum cardinality such that every path between s and t in $G \setminus F$ has length more than L. The problem has been studied in various contexts since the beginning of the 70's (e.g., [1, 26, 2]) and occasionally it appears also under the name the *short paths interdiction problem* [19].

Closely related is the *shortest path most vital edges and vertices problem* (e.g. [3, 4, 5]): given a graph G, two distinguished vertices s and t and an integer k, the task is to find a subset F of k edges (vertices) whose removal maximizes the increase in the length of the shortest path between s and t. If we introduce an additional parameter – the desired minimum distance of s and t – we obtain a parameterized version of the L-bounded cut problem: given a graph G, two distinguished vertices s and t and integers k and L, does there exist a subset F of at most k edges (vertices) such that every path between s and t in $G \setminus F$ has length more than L? We also note that $\mathcal{NP}$-hardness of the shortest path most vital edges (vertices) problem immediately implies $\mathcal{NP}$-hardness of the L-bounded edge (vertex) cut problem, and vice versa.

In contrast to many other cut problems on graphs (e.g., multiway cut, multicut, sparsest cut, balanced cut, maximum cut, multiroute cut), the known approximations of the minimum L-bounded cut problem are substantially weaker. In this work we focus on algorithms for restricted graph classes, namely planar graphs, bounded genus graphs and graphs with bounded, yet not constant, treewidth, and provide new results for the L-bounded cut problem on them; the results for planar graphs solve one of the open problems suggested by Bazgan et al. [5]. We also remark that the L-bounded cut problem does not fit into the framework of Czumaj et al. [9] that is applicable for some $\mathcal{NP}$-hard problems in graphs with superlogarithmic treewidth.

Related Results. $\mathcal{NP}$-hardness of the shortest path most vital edges problem (and, thus, as noted above, also of the L-bounded cut problem) was proved by Bar-Noy et al. [4]. The best known approximation algorithm for the minimum L-bounded cut problem on general graphs has approximation ratio only $\mathcal{O}(\min\{L, n/L\}) \subseteq \mathcal{O}(\sqrt{n})$ for the vertex case and $\mathcal{O}(\min\{L, n^2/L^2, \sqrt{m}\}) \subseteq \mathcal{O}(n^{2/3})$ for the edge case, where m denotes the number of edges and n the number of vertices [2]. On the lower bound side, the edge–deletion version of the problem is known to be $\mathcal{NP}$-hard to approximate within a factor of 1.1377 for $L \geq 4$, and the vertex–deletion version for $L \geq 5$ [2]; for smaller values of L the problem is solvable in polynomial time [26, 27]. Independently, Khachiyan et al. [19] proved that a version of the problem with edge lengths is $\mathcal{NP}$-hard to approximate within a factor smaller than 1.36. Recently, assuming the Unique Games Conjecture, Lee [23] proved that the problem is $\mathcal{NP}$-hard to approximate within *any* constant factor.

An instance of the L-bounded edge (vertex, resp.) cut problem on a graph $G = (V, E)$ of treewidth τ can be cast as an instance of constraint satisfaction problem (CSP) with $|V|$ variables, domain of size $L + 2$ ($L + 3$, resp.) and treewidth τ.[1] As CSP instances with n variables, treewidth bounded by τ and domain by D can be solved in $O(D^\tau n)$ time [15] (when a tree decomposition of width τ of the constraint graph is given), the problem is fixed-parameter tractable with respect τ. Dvořák and Knop [10] provide a direct proof of the same result with a slightly worse dependance on L and τ; they also prove that the problem is $W[1]$-hard when parameterized by the treewidth only.

From the point of view of parameterized complexity, the problem was also studied by Golovach and Thilikos [16], Bazgan et al. [5] and by Fluschnik et al. [14].

For planar graphs, the problem is known to be $\mathcal{NP}$-hard [13, 30], too, and the edge–deletion version of the problem has no polynomial-size kernel when parameterized by the combination of L and the size of the optimal solution [14].

For more detailed overview of other related results and applications, we refer to the papers [19, 2, 27]. For more background on parameterized algorithms, we refer to the textbook by Cygan et al. [8].

Our Contribution. We show that on planar graphs, both the edge– and the vertex–deletion version of the problem are solvable in $\mathcal{O}((L+2)^{3L}n)$ time. That is, we show that on planar graphs the minimum L-bounded cut problem is fixed-parameter tractable (FPT) with respect to L. Furthermore, we show that the problem remains FPT even for bounded genus graphs, a super class of planar graphs. This is in contrast with the situation for general graphs – the problem is NP-hard even for $L = 4$ and $L = 5$, for the edge– and vertex–deletion versions, respectively.

Our second contribution is a τ-approximation algorithm for the vertex–deletion version of the problem, if a tree decomposition of width τ is given. If the decomposition is not given, then using the best known algorithm to compute a tree decomposition of a given graph, we obtain an $\mathcal{O}(\tau\sqrt{\log \tau})$-approximation for general graphs with treewidth τ, and an $\mathcal{O}(\tau)$-approximation for planar graphs, graphs excluding a fixed minor and graphs with treewidth bounded by $\mathcal{O}(\log n)$. For graphs with treewidth bounded by $\tau = \mathcal{O}(n^{1/2-\epsilon})$ for any $\epsilon > 0$, but not by a constant, in terms of n, this is the best approximation we are aware of.

Our results are based on a combination of observations about the structure of L-bounded cuts and various known results. The proofs are straightforward but apparently non-obvious, considering the attention given to the problem in recent years.

2 Preliminaries

Throughout the paper, given a graph $G = (V, E)$, we use m to denote the number of edges in G, that is, $m = |E|$, and for $u, v \in V$, we use $d(u, v)$ to

[1]For the sake of completeness, in Appendix A we provide details about this reduction.

denote the shortest path distance between u and v, that is, the number of edges on a shortest path. For a graph $G = (V, E)$ and a subset of vertices $W \subset V$, *a subgraph of G induced by W* is the graph (W, F) where F is the subset of edges with both vertices in W, that is, $F = \{\{u, v\} \in E \mid u, v \in W\}$. For a graph $G = (V, E)$ and a subset of edge $F \subset E$, we use $G \setminus F$ to denote the graph $(V, E \setminus F)$, and for a subset of vertices $W \subset V$, we use $G \setminus W$ to denote the subgraph of G induced by $V \setminus W$.

Given a graph $G = (V, E)$ with two distinguished vertices s and t, a subset of vertices $W \subset V$ is an *$s - t$ vertex cut* if s and t are in different connected components in $G \setminus W$. A subset of vertices $W \subset V$ is a *vertex cut* if the removal of W disconnects the graph, that is, if the graph $G \setminus W$ is not connected.

For notions related to the treewidth of a graph and tree decomposition we stick to the standard terminology as given in the book by Kloks [21]. A *tree decomposition* of a graph $G = (V, E)$ is a tree T with a node set $V(T)$ in which each *node* $a \in V(T)$ has an assigned set of vertices $B(a) \subseteq V$, called a *bag*, such that $\bigcup_{a \in T} B(a) = V$ with the following properties:

- for any $\{u, v\} \in E$, there exists a node $a \in V(T)$ such that $u, v \in B(a)$,
- if $v \in B(a)$ and $v \in B(b)$, then $v \in B(c)$ for all nodes c on the path between a and b in T.

The tree decomposition is *rooted* if one of the nodes in the tree T is specified as the root. The *treewidth of a tree decomposition T* is the size of the largest bag of T minus one. The *treewidth of a graph G* is the minimum treewidth over all possible tree decompositions of G. To distinguish vertices of a graph G and of a tree decomposition T of G, we call the vertices of the tree decomposition *nodes*. A tree decomposition satisfies the *non-containment condition* if no bag is contained in any other bag.

A simple yet important property of tree decompositions is stated in the following lemma.

Lemma 1 (Folklore) *Let G be a graph and T a tree decomposition of G satisfying the non-containment condition. Then*

- *For any node $a \in V(T)$ that is not a leaf, $B(a)$ is a vertex cut in G.*
- *For any two adjacent nodes $a, b \in V(T)$ such that none of the two bags $B(a)$ and $B(b)$ is contained in the other, $B(a) \cap B(b)$ is a vertex cut in G.*

Note that the size of the cut $B(a) \cap B(b)$ in Lemma 1 is at most the width of the tree decomposition T.

In a rooted tree, the *parent* of a node is the node connected to it on the path to the root; every node except the root has a unique parent. A *child* of a node v is a node of which v is the parent. A *descendant* of any node v is any node which is either the child of v or is (recursively) the descendant of any of the children of v. A *leaf* is a vertex having no child.

3 Fixed-parameter Tractability on Planar and Bounded Genus Graphs

Our main tools are the following two well-known results.

Theorem 1 (Robertson and Seymour [28], Bodlaender [6]) *The treewidth of a planar graph with radius d is at most $3d$.*

Theorem 2 (Freuder [15]) *CSP instances with n variables, treewidth bounded by τ and domain by D are solvable in $\mathcal{O}(D^\tau n)$ time.*

Since the minimum L-bounded edge (vertex, resp.) cut problem on a graph $G = (V, E)$ of treewidth τ can be cast as a CSP instance with $|V|$ variables, treewidth τ and domain of size $L+2$ ($L+3$, resp.), the problem is solvable in $\mathcal{O}((L+2)^\tau n)$ time ($\mathcal{O}((L+3)^\tau n)$ time, resp.), as already stated in the introduction and explained in Appendix A.

The main result of this section says that the L-bounded cut problem on planar graphs is fixed-parameter tractable, with respect to the parameter L.

Theorem 3 *The minimum L-bounded edge (vertex, resp.) cut problem on planar graphs is solvable in $\mathcal{O}((L+2)^{3L}n)$ time ($\mathcal{O}((L+3)^{3L}n)$ time, resp.).*

Proof: We prove the theorem for the edge–deletion version; the proof for the vertex–deletion version is analogous.

Given a graph $G = (V, E)$, $s, t \in V$ and an integer L, let $V' = \{v \in V \mid d(s, v) + d(t, v) \leq L\}$. In words, V' is the subset of vertices lying on paths of length at most L between s and t. Without loss of generality we assume that $d(s, t) \leq L$ – otherwise the problem is trivial. Let G' be the subgraph of G induced by V'. Note that the radius of G' is at most L as, by definition, $d(s, v) \leq L$ for every $v \in V'$.

The set V' (and, thus, the subgraph G') can be computed using the $\mathcal{O}(n)$-time algorithm for single-source shortest paths on planar graphs by Klein et al. [20] that we run twice, once for s and once for t. Note that both s and t belong to V'.

Obviously, G' is a planar graph, and by Theorem 1, its treewidth is at most $3L$. We solve the L-bounded problem for G' and s and t by Theorem 2 in $\mathcal{O}((L+2)^{3L}n)$ time. Let F be the optimal solution for G'. We claim that F is an optimal solution for the original instance of the problem on G as well. To prove *feasibility* of F, assume, for contradiction, that there exists an $s-t$-path p of length at most L in $(V, E \setminus F)$. As there is no such path in $G' \setminus F$, p has to use at least one vertex v from $V \setminus V'$. However, this yields a contradiction: on the one hand, $d(s, v) + d(v, t) \leq L$ as v is on an $s-t$-path of length at most L, on the other hand, $d(s, v) + d(v, t) > L$ as v is not in V'. Concerning the *optimality* of F, it is sufficient to note that the size of an optimal solution for the subgraph G' is a lower bound on the size of an optimal solution for G. $\square$

Theorem 1 was generalized by Eppstein [11] to graphs of bounded genus and this result makes it possible to generalize Theorem 3 also to graphs of bounded genus.

Theorem 4 (Eppstein [11]) *There exists a constant $\hat{c}$ such that the treewidth of every graph with radius d and genus g is at most $\hat{c}dg$.*

In the same way as we used Theorem 1 to prove fixed-parameter tractability for the L-bounded cut problem on planar graphs (Theorem 3), we can use Theorem 4 to prove fixed-parameter tractability of the L-bounded cut problem on graphs of bounded genus. The only other change is that instead of the $\mathcal{O}(n)$-time single-source shortest path algorithm for planar graphs [20] we use the $\mathcal{O}(n + m)$-time single-source shortest path algorithm for general graphs [29]. Considering the fact that by Euler's formula, genus g graphs have $\mathcal{O}(n + g)$ edges [17], we obtain the following theorem.

Theorem 5 *The minimum L-bounded edge (vertex, resp.) cut problem on graphs with genus g is solvable in $\mathcal{O}((L + 2)^{\hat{c}gL}n)$ time ($\mathcal{O}((L + 3)^{\hat{c}gL}n)$ time, resp.).*

4 τ-Approximation for L-bounded Vertex Cuts

In this section we describe an algorithm for the L-bounded $s - t$ vertex cut problem whose approximation ratio is parameterized by the width τ of a tree decomposition T of the input graph G. Throughout this section we assume that the vertices s and t are not connected by an edge – in such a case there is no L-bounded $s - t$ vertex cut in G. Without loss of generality we also assume that tree decompositions in this section satisfy the *non-containment condition*.

Consider a graph G and a rooted tree decomposition T of G of width τ. By $d(G, s, t)$ we denote the distance between s and t in G. Given a subset $R \subseteq V(T)$ of nodes inducing a connected subtree of T, a *deepest node* in R is a node in R with no child in R. Given a node b of the rooted tree decomposition T, we denote by T_b the subtree of T consisting of b and of all its descendants, and by G_b the subgraph of G induced by vertices in bags of T_b; similarly, we denote by $\bar{T}_b$ the subtree of T consisting of all nodes in T including b and excluding the descendants of b and by $\bar{G}_b$ the subgraph of G induced by vertices in bags of $\bar{T}_b$. Note that b is the only node of the tree T that appears in both subtrees T_b and $\bar{T}_b$.

The following simple observation captures the main properties of G and T that make the algorithm of this section work. For notational simplicity, in the rest of the section we use the term *L-bounded path* for an $s - t$ path of length at most L.

Claim 2 *If b is a deepest node in the set $R = \{a \in V(T) \mid d(G_a, s, t) \le L\}$ and $G' = \bar{G}_b \setminus (B(b) \setminus \{s, t\})$, then the following holds:*

1. *There is at least one L-bounded path in G_b.*
2. *There is no L-bounded path in $G_b \setminus (B(b) \setminus \{s, t\})$.*
3. *The L-bounded paths in G_b are internally vertex disjoint with the L-bounded paths in G'.*

Proof: The first point follows from the membership of b in the set R.

The second point is obvious if b has none or exactly one child. Assume that b is a node with two or more children and that there is an L-bounded path p between s and t in $G_b \setminus (B(b) \setminus \{s,t\})$. Then, by the choice of b (i.e., a deepest node in R), there exist children c and c' of b and vertices x, x' on the path p such that $x \in V(G_c) \setminus V(G_{c'})$ and $x' \in V(G_{c'}) \setminus V(G_c)$. Consider the vertex cut $B(b)$ (cf. Lemma 1) and note that x and x' belong to different components of connectivity of $G_b \setminus B(b)$. Thus, the sub-path of p between x and x' has to contain as an inner vertex a vertex from $B(b) \setminus \{s,t\}$, a contradiction. We conclude that there is no L-bounded path in $G_b \setminus (B(b) \setminus \{s,t\})$.

For the third point, note that any L-bounded path in G either intersects the set $B(b) \setminus \{s,t\}$ or appears in $G \setminus (B(b) \setminus \{s,t\})$. As every L-bounded path in G_b intersects, by the second part of this claim, the set $B(b) \setminus \{s,t\}$, and as G' is a subgraph of $G \setminus (B(b) \setminus \{s,t\})$, the third part of the Claim follows. □

The L-bounded cut is computed using the recursive procedure L-bounded_cut(G,T,s,t,L) described in Algorithm 1. In step 12, prune(G,T,C) is a procedure that for a graph $G=(V,E)$, a tree decomposition T and a vertex set $C \subset V$, deletes from G the vertices in C and all adjacent edges, and modifies the tree decomposition T by deleting the vertices in C from all bags.

Algorithm 1 L-bounded_cut(G,T,s,t,L)

```
1:  if $d(G,s,t) > L$ then                                # no need to remove anything
2:      return $\emptyset$
3:  else $R \leftarrow \{a \in V(T) \mid d(G_a,s,t) \le L\}$                       # set up
4:      $b \leftarrow$ a deepest node in $R$
5:      if $|B(b) \cap \{s,t\}| \le 1$ then               # simple cases - no need for
    recursion
6:          if $|B(b) \cap \{s,t\}| = 1$ then # $s \in B(b)$, $t \notin B(b)$, or vice versa
7:              return $B(b) \setminus \{s,t\}$
8:          else                                                   # $s,t \notin B(b)$
9:              $c \leftarrow$ child of $b$ s.t. $s \in G_c$
10:             return $B(b) \cap B(c)$
11:     else                                           # recursion, $s,t \in B(b)$
12:         $(G',T') \leftarrow$ prune$(\bar{G}_b, \bar{T}_b, B(b) \setminus \{s,t\})$
13:         $S' \leftarrow L$-bounded_cut$(G',T',s,t,L)$
14:         return $S' \cup B(b) \setminus \{s,t\}$
```

The main result of this section is obtained from Lemmas 3 and 4.

Lemma 3 *Given a graph $G=(V,E)$, two vertices $s,t \in V$ and a tree decomposition T of G of width τ, Algorithm 1 finds in polynomial time an L-bounded $s-t$ vertex cut.*

Proof: To prove the correctness of Algorithm 1, we proceed by induction on the recursion depth. We start by showing the correctness of the final recursive calls. To this end we distinguish the following three cases dealt with in the algorithm:

Case 1. $d(G, s, t) > L$. As there is no need to remove anything in this case, the correctness is obvious from the description of the algorithm.

Case 2. $|B(b) \cap \{s, t\}| = 1$ (where b is the node selected in step 4). As there exists at least one L-bounded path in G_b, both vertices s and t appear in G_b, and as $|B(b) \cap \{s, t\}| = 1$, one of the vertices s and t appears in $G_b \setminus B(b)$. By the second point of Claim 2, $B(b) \setminus \{s, t\}$ is an L-bounded cut in G_b, and every L-bounded path in G disjoint with $B(b) \setminus \{s, t\}$ has to use a vertex that does not appear in G_b. However, as $B(b)$ is a vertex cut in G separating $G_b \setminus B(b)$ from the rest of the graph, there is no L-bounded path in G disjoint with $B(b) \setminus \{s, t\}$. We conclude that $G \setminus (B(b) \setminus \{s, t\})$ is an L-bounded $s - t$ vertex cut in G.

Case 3. $B(b) \cap \{s, t\} = \emptyset$ (where b is the node selected in step 4). The argument is similar as in the previous case. On one hand, as there exists at least one L-bounded path in G_b, both vertices s and t appear in G_b, and as none of them belongs to the set $B(b)$, there must be a child c of b such that $s \in G_c$. On the other hand, the second point of Claim 2 implies that every L-bounded path in G disjoint with $B(b) \setminus \{s, t\}$ has to use a vertex that does not appear in G_b. As $B(b) \cap B(c)$ is a vertex cut in G separating G_c from the rest of the graph, we conclude that there is no L-bounded path in $G \setminus (B(b) \cap B(c))$.

Inductive step. Consider a run of the procedure with a graph G and its tree decomposition T, and let R and b be the objects defined by the procedure in steps 3 and 4. Note that the set R induces a connected subgraph of T.

The inductive assumption (i.e., S' is an L-bounded cut in G') combined with the second point of Claim 2 implies that the set $S' \cup B(b) \setminus \{s, t\}$ is an L-bounded $s - t$ cut in G, completing the inductive step in the proof of the correctness.

Concerning the running time, the second point of Claim 2 implies that the vertex b selected as a deepest node from R in some iteration will not belong to the set R in any of the future recursive calls. Thus, the size of the set R decreases by at least one with each new recursive call, yielding an upper bound $V(T)$ on the number of recursive calls. Apart from the recursive call, each level of recursion can be implemented in time $\mathcal{O}(\tau \cdot |V(T)|)$, yielding an upper bound $\mathcal{O}(\tau \cdot |V(T)|^2)$ on the total running time. $\square$

Let $cost(G, T)$ be the size of the solution computed by Algorithm 1 for a graph G and a tree decomposition T of G, and let $opt(G)$ be the size of an optimal solution for the graph G.

Lemma 4 *Given a graph $G = (V, E)$, two vertices $s, t \in V$ and a tree decomposition T of G of width τ, then*

$$cost(G, T) \leq \tau \cdot opt(G) \ .$$

Proof: Similarly as in the proof of Lemma 3, we proceed by induction on the recursion depth. We start by showing the correctness of the bound for the final recursive calls and, as before, we distinguish the following three cases:

Case 1. $d(G, s, t) > L$. For graphs with no L-bounded $s - t$ path, the claim is obvious as $cost(G, T) = 0$ in this case.

Case 2. $|B(b) \cap \{s,t\}| = 1$. It suffices to note that $|B(b) \setminus \{s,t\}| \leq \tau$ and that $opt(G) \geq 1$.

Case 3. $B(b) \cap \{s,t\} = \emptyset$. As in the previous case, it suffices to note, using the non-containment condition, that $|B(b) \cap B(c)| \leq \tau$ and that $opt(G) \geq 1$.

Inductive step. From the description of the algorithm we know that $cost(G,T) \leq \tau + cost(G',T')$. Points 1 and 3 of Claim 2 imply $opt(G) \geq 1 + opt(G')$. Combining these observations with the inductive assumption $cost(G',T') \leq \tau \cdot opt(G')$, we obtain $cost(G,T) \leq \tau \cdot opt(G)$. □

Putting Lemmas 3 and 4 together, we get the main result of this section.

Theorem 6 *Given a graph G, a rooted tree decomposition T of G of width τ, vertices s and t and an integer L, Algorithm 1 finds in polynomial time a τ-approximation of the minimum L-bounded $s-t$ vertex cut.*

Remark. At the cost of increasing the approximation ratio to $\tau+1$, the steps 5-10 of the algorithm can be simplified as follows:

if $|B(b) \cap \{s,t\}| \leq 1$ **then return** $B(b) \setminus \{s,t\}$

By this change, if $|B(b) \cap \{s,t\}| = 1$, the output of the algorithm will not change. If $B(b) \cap \{s,t\} = \emptyset$, the modified algorithm outputs $B(b) = B(b) \setminus \{s,t\}$ instead of the original output $B(b) \cap B(c)$; obviously, this will not break the correctness of the algorithm but the bound in Lemma 4 will change to $cost(G,T) \leq (\tau+1) \cdot opt(G)$ as $B(b)$ may be of size $\tau+1$.

In the case that we are not given a tree decomposition on input, we start by constructing it using one of the known algorithms: Feige et al. [12] describe a polynomial time algorithm that yields, for a given graph of treewidth τ, a tree decomposition of width $\mathcal{O}(\tau\sqrt{\log \tau})$; for planar graphs and for graphs excluding a fixed minor, the width is in $\mathcal{O}(\tau)$. Similarly, for graphs with treewidth bounded by $\mathcal{O}(\log n)$, Bodlaender et al. [7] describe how to find in polynomial time a tree decomposition of width $\mathcal{O}(\tau)$. Depending on the input graph, one of these algorithms is used to obtain a desired tree decomposition. Thus, we obtain the following corollary.

Corollary 7 *There exists an $\mathcal{O}(\tau\sqrt{\log \tau})$-approximation algorithm for the minimum L-bounded vertex cut on graphs with treewidth τ; for planar graphs, graphs excluding a fixed minor and graphs with treewidth bounded by $\mathcal{O}(\log n)$, there exists an $\mathcal{O}(\tau)$-approximation algorithm.*

5 Open problems

Having shown fixed-parameter tractability of the L-bounded cut problem on planar and bounded genus graphs by giving $L^{\mathcal{O}(L)}n$ time algorithms, the question arises whether the presented bounds are optimal. Could the dependence on the parameter L be improved to $2^{\mathcal{O}(L)}$? As our proofs of fixed-parameter tractability rely on the existence of the algorithm for CSP, a much more general class of problems, on graphs of bounded treewidth, it is conceivable that a better bound is possible; on the other hand, under the Strong Exponential Time Hypothesis [18], matching lower bounds for some problems expressible as CSP (e.g., q-coloring) do exist [25].

A natural open problem for planar graphs is whether the shortest path most vital edges (vertices) problem is fixed-parameter tractable on them, with respect to the number k of deleted edges (vertices). Despite the close relation of the L-bounded cut problem and the shortest path most vital edges (vertices) problem, fixed-parameter tractability of one of them does not seem to easily imply fixed-parameter tractability of the other problem.

The τ-approximation for L-bounded vertex cuts is based on the fact that bags in a tree decomposition yield *vertex* cuts of size at most equal the width of the decomposition. Unfortunately, this is not the case for *edge* cuts – one can easily construct bounded treewidth graphs with no small balanced edge cuts. Thus, another open problem is to look for better approximation algorithms for minimum L-bounded *edge* cuts, for graphs with treewidth bounded by τ.

Yet another challenging and more general open problem is to narrow the gap between the upper and lower bounds on the approximation ratio of algorithms for the L-bounded cut for general graphs: the best upper bound for the edge– and vertex–deletion version of the problem is $\mathcal{O}(n^{2/3})$ and $\mathcal{O}(\sqrt{n})$, resp., while the best lower bound is constant.

Finally, we note that the edge–deletion version of the L-bounded cut problem in a graph $G = (V, E)$ is a kind of a *vertex ordering* problem. We are looking for a mapping ℓ from the vertex set V to the set $\{0, 1, \ldots, L, L + 1\}$ such that $\ell(s) = 0$, $\ell(t) = L + 1$ and the objective is to minimize the number of edges $\{u, v\} \in E$ for which $|\ell(u) - \ell(v)| > 1$; given a solution $F \subseteq E$, the lengths of the shortest paths from s to all other vertices in $G \setminus F$ yield such a mapping of cost $|F|$. There are plenty of results dealing with *linear* vertex ordering problems where one is looking for a bijective mapping from the vertex set V to the set $\{1, 2, \ldots, n\}$ minimizing some objective function (e.g., the minimum cut linear arrangement problem, the minimum feedback arc set problem [24]). However, the requirement that the mapping is a bijection to a set of size n seems crucial in the design and analysis of approximation algorithms for these problems. The question is whether it is possible to obtain good approximations for some nontrivial *non-linear* vertex ordering problems.

Acknowledgements

We thank Martin Koutecký and Hans Raj Tiwary for stimulating discussions and anonymous referees for numerous helpful comments and for a suggestion yielding an improvement by a factor $\log n$ in the approximation ratio in Theorem 6.

References

[1] J. Adámek and V. Koubek. Remarks on flows in network with short paths. *Commentationes Mathematicae Universitatis Carolinae*, 12(4):661–667, 1971. URL: `http://www.dml.cz/dmlcz/105376`.

[2] G. Baier, T. Erlebach, A. Hall, E. Köhler, P. Kolman, O. Pangrác, H. Schilling, and M. Skutella. Length-bounded cuts and flows. *ACM Trans. Algorithms*, 7(1):4:1–4:27, 2010. Preliminary version in *Proc. of 33rd International Colloquium on Automata, Languages, and Programming (ICALP)*, 2006. `doi:10.1145/1868237.1868241`.

[3] M. O. Ball, B. L. Golden, and R. V. Vohra. Finding the most vital arcs in a network. *Operations Research Letters*, 8(2):73 – 76, 1989. `doi:10.1016/0167-6377(89)90003-5`.

[4] A. Bar-Noy, S. Khuller, and B. Schieber. The complexity of finding most vital arcs and nodes. Technical Report CS-TR-3539, Univ. of Maryland, Dept. of Computer Science, Nov. 1995. URL: `ftp://ftp.cs.umd.edu/pub/papers/papers/3539/3539.ps.Z`.

[5] C. Bazgan, A. Nichterlein, and R. Niedermeier. A refined complexity analysis of finding the most vital edges for undirected shortest paths. In *Proc. of Algorithms and Complexity - 9th International Conference (CIAC)*, pages 47–60, 2015. `doi:10.1007/978-3-319-18173-8_3`.

[6] H. L. Bodlaender. Planar graphs with bounded treewidth. Technical Report RUU-CS-88-14, Univ. Utrecht, Dept. of Computer Science, 1988.

[7] H. L. Bodlaender, P. G. Drange, M. S. Dregi, F. V. Fomin, D. Lokshtanov, and M. Pilipczuk. A $c^k n$ 5-approximation algorithm for treewidth. *SIAM J. Comput*, 45(2):317–378, 2016. `doi:10.1137/130947374`.

[8] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer, 2015.

[9] A. Czumaj, M. M. Halldórsson, A. Lingas, and J. Nilsson. Approximation algorithms for optimization problems in graphs with superlogarithmic treewidth. *Information Processing Letters*, 94(2):49–53, 2005. `doi:10.1016/j.ipl.2004.12.017`.

[10] P. Dvořák and D. Knop. Parametrized complexity of length-bounded cuts and multi-cuts. In *Proc. of 12 Annual Conference on Theory and Applications of Models of Computation (TAMC)*, pages 441–452, 2015. `doi:10.1007/978-3-319-17142-5_37`.

[11] D. Eppstein. Diameter and treewidth in minor-closed graph families. *Algorithmica*, 27(3):275–291, 2000. `doi:10.1007/s004530010020`.

[12] U. Feige, M. T. Hajiaghayi, and J. R. Lee. Improved approximation algorithms for minimum weight vertex separators. *SIAM J. Comput*, 38(2):629–657, 2008. Preliminary version in Proc. of STOC 2005. `doi:10.1137/05064299X`.

[13] T. Fluschnik, D. Hermelin, A. Nichterlein, and R. Niedermeier. Fractals for kernelization lower bounds, with an application to length-bounded cut problems. *CoRR*, abs/1512.00333, 2015. `arXiv:arXiv:1512.00333`.

[14] T. Fluschnik, D. Hermelin, A. Nichterlein, and R. Niedermeier. Fractals for kernelization lower bounds, with an application to length-bounded cut problems. In *Proc. of 43rd International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 25:1–25:14, 2016. `doi:10.4230/LIPIcs.ICALP.2016.25`.

[15] E. C. Freuder. Complexity of K-tree structured constraint satisfaction problems. In *Proc. of the 8th National Conference on Artificial Intelligence*, pages 4–9, 1990.

[16] P. A. Golovach and D. M. Thilikos. Paths of bounded length and their cuts: Parameterized complexity and algorithms. *Discrete Optimization*, 8(1):72–86, 2011. `doi:10.1016/j.disopt.2010.09.009`.

[17] F. Harary. *Graph Theory.* Addison-Wesley, 1969.

[18] R. Impagliazzo and R. Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci*, 62(2):367–375, 2001. `doi:10.1006/jcss.2000.1727`.

[19] L. Khachiyan, E. Boros, K. Borys, K. M. Elbassioni, V. Gurvich, G. Rudolf, and J. Zhao. On short paths interdiction problems: Total and node-wise limited interdiction. *Theory Comput. Syst*, 43(2):204–233, 2008. `doi:10.1007/s00224-007-9025-6`.

[20] P. Klein, S. Rao, M. Rauch, and S. Subramanian. Faster shortest-path algorithms for planar graphs. In *Proc. of the 26th ACM Symposium on Theory of Computing (STOC)*, pages 27–37, 1994. `doi:10.1145/195058.195092`.

[21] T. Kloks. *Treewidth: Computations and Approximations*, volume 842 of *Lecture Notes in Computer Science*. Springer, 1994.

[22] P. Kolman and M. Koutecký. Extended formulation for CSP that is compact for instances of bounded treewidth. *Electr. J. Comb*, 22(4):P4.30, 2015.

[23] E. Lee. Improved hardness for cut, interdiction, and firefighter problems. In *Proc. of 44rd International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 92:1–92:14, 2017. `doi:10.4230/LIPIcs.ICALP.2017.92`.

[24] F. T. Leighton and S. Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *J. ACM*, 46(6):787–832, 1999. `doi:10.1145/331524.331526`.

[25] D. Lokshtanov, D. Marx, and S. Saurabh. Known algorithms on graphs on bounded treewidth are probably optimal. In *Proc. of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 777–789, 2011. `doi:10.1137/1.9781611973082`.

[26] L. Lovász, V. Neumann-Lara, and M. D. Plummer. Mengerian theorems for paths of bounded length. *Periodica Mathematica Hungarica*, 9:269–276, 1978. `doi:10.1007/BF02019432`.

[27] A. R. Mahjoub and S. T. McCormick. Max flow and min cut with bounded-length paths: complexity, algorithms, and approximation. *Math. Program*, 124(1-2):271–284, 2010. `doi:10.1007/s10107-010-0366-6`.

[28] N. Robertson and P. D. Seymour. Graph minors. III. planar tree-width. *J. Comb. Theory, Ser. B*, 36(1):49–64, 1984. `doi:10.1016/0095-8956(84)90013-3`.

[29] M. Thorup. Undirected single-source shortest paths with positive integer weights in linear time. *J. ACM*, 46(3):362–394, 1999. `doi:10.1145/316542.316548`.

[30] P. Zschoche, T. Fluschnik, H. Molter, and R. Niedermeier. The Computational Complexity of Finding Separators in Temporal Graphs. *ArXiv e-prints*, Nov. 2017. `arXiv:1711.00963`.

A Appendix L-bounded Cut as a CSP Instance

An instance $Q = (V, \mathcal{D}, \mathcal{H}, \mathcal{C})$ of CSP [22] consists of

- a set of *variables* z_v, one for each $v \in V$; without loss of generality we assume that $V = \{1, \dots, n\}$,
- a set $\mathcal{D}$ of finite *domains* D_v (also denoted $D(v)$), one for each $v \in V$,
- a set of *hard constraints* $\mathcal{H} \subseteq \{C_U \mid U \subseteq V\}$ and a set of *soft constraints* $\mathcal{C} \subseteq \{C_U \mid U \subseteq V\}$ where each constraint $C_U \in \mathcal{C} \cup \mathcal{H}$ with $U = \{i_1, i_2, \dots, i_k\}$ and $i_1 < i_2 < \cdots < i_k$, is a $|U|$-ary relation $C_U \subseteq D_{i_1} \times D_{i_2} \times \cdots \times D_{i_k}$.

For a vector $z = (z_1, z_2, \dots, z_n)$ and $U = \{i_1, i_2, \dots, i_k\} \subseteq V$ with $i_1 < i_2 < \cdots < i_k$, we define the *projection of* z on U as $z|_U = (z_{i_1}, z_{i_2}, \dots, z_{i_k})$. A vector $z = (z_1, z_2, \dots, z_n)$ *satisfies the constraint* $C_U \in \mathcal{C} \cup \mathcal{H}$ if and only if $z|_U \in C_U$. We say that a vector $z^\star = (z_1^\star, \dots, z_n^\star)$ is *a feasible solution* for Q if $z^\star \in D_1 \times D_2 \times \ldots \times D_n$ and $z^\star$ satisfies every hard constraint $C \in \mathcal{H}$. In the *maximization* (*minimization*, resp.) version of CSP, the task is to find a feasible solution that maximizes (minimizes, resp.) the number of *satisfied* (unsatisfied, resp.) soft constraints; the *cost* of a feasible solution is the number of satisfied (unsatisfied, resp.) soft constraints.

The *constraint graph* of Q is defined as $H = (V, E)$ where $E = \{\{u, v\} \mid \exists C_U \in \mathcal{C} \cup \mathcal{H} \text{ s.t. } \{u, v\} \subseteq U\}$. We say that a *CSP instance* Q *has bounded treewidth* if the constraint graph of Q has bounded treewidth.

Given an edge–deletion version of the L-bounded cut instance $G = (V, E)$ with $s, t \in V$ and an integer L, we construct the corresponding minimization CSP instance $Q = (V, \mathcal{D}, \mathcal{H}, \mathcal{C})$ as follows. The set of variables of Q coincides with the set V of vertices of G and for each $v \in V$, the corresponding domain D_v is $\{0, 1, \dots, L, L+1\}$. The set of hard constraints $\mathcal{H}$ consists of two constraints, $C_{\{s\}} = \{0\}$ and $C_{\{t\}} = \{L+1\}$. The set of soft constraints $\mathcal{C}$ contains a constraint

$$C_{\{i,j\}} = \{(\ell, \ell') \mid 0 \le \ell, \ell' \le L+1,\ |\ell - \ell'| \le 1\}$$

for each edge $\{i, j\} \in E$ of the graph G.

To see that a feasible solution for the constructed instance Q of CSP corresponds to a feasible solution of the L-bounded cut problem of the same cost, and vice versa, we observe the following.

Given an optimal solution $F \subset E$ of the edge–deletion version of the L-bounded cut problem, we distinguish two cases. If s and t belong to the same component of connectivity in $(V, E \setminus F)$, then the vector of shortest path distances from s to all other vertices in $(V, E \setminus F)$ yields a feasible solution for the CSP instance Q (to be more precise, if some of the distances are larger than $L+1$, we replace in the vector every such value by $L+1$); if s and t do not belong to the same component of connectivity in $(V, E \setminus F)$, we obtain a feasible solution for Q by assigning the value 0 to every vertex in the s–component and the value $L+1$ to every vertex in the t–component. Note that in both cases the cost of the L-bounded cut and the cost of the CSP instance Q are the same. We also note that for every feasible solution $(z_1, \dots, z_n)$ of the instance Q, the set $F = \{\{u,v\} \in E \mid |z_u - z_v| > 1\}$ is an L-bounded cut of the same cost. Finally, we note that the constraint graph of Q coincides with the original graph G.

For the vertex–deletion version of the L-bounded cut problem in $G = (V, E)$, the corresponding minimization CSP instance $Q = (V, \mathcal{D}, \mathcal{H}, \mathcal{C})$ is defined similarly. For each $v \in V$, we have $D_v = \{-1, 0, \dots, L, L+1\}$ – the domain of every vertex is extended by an extra element -1 representing the fact that v belongs to the L-bounded cut. The set of hard constraints $\mathcal{H}$ contains constraints $C_{\{s\}} = \{0\}$ and $C_{\{t\}} = \{L+1\}$, and for each edge $\{i,j\} \in E$ also a constraint

$$\begin{aligned} H_{\{i,j\}} = &\{(\ell, \ell') \mid 0 \le \ell, \ell' \le L+1,\ |\ell - \ell'| \le 1\} \\ &\cup \{(\ell, -1) \mid 0 \le \ell \le L+1\} \cup \{(-1, \ell) \mid 0 \le \ell \le L+1\}\ . \end{aligned}$$

The set of soft constraints contains for each vertex u other than s and t a constraint

$$C_{\{u\}} = \{0, 1 \dots, L, L+1\}\ .$$

Given an optimal solution $U \subset V$ of the vertex–deletion version of the L-bounded cut problem, we distinguish two cases. If s and t belong to the same component of connectivity of $G' = G \setminus U$, then assigning to every $v \in U$ the value -1 and assigning to every $v \in V \setminus U$ its distance from s in G' yields a feasible solution for the CSP instance Q (to be more precise, if some of the distances are larger than $L+1$, we replace in the vector every such value by $L+1$); if s and t do not belong to the same component of connectivity in G', we obtain a feasible solution for Q by assigning the value 0 to every vertex in the s–component, the value $L+1$ to every vertex in the t–component and the value -1 to every $v \in U$. Note that in both cases the size of the L-bounded cut and the cost of the CSP instance Q are the same. We also note that for every feasible solution $(z_1, \dots, z_n)$ of the instance Q, the set $U = \{v \in V \mid z_v = -1\}$ is an L-bounded cut of size equal the cost of Q.