Parameterized graph separation problems

Dániel Marx[*]

26th March 2005

Abstract

We consider parameterized problems where some separation property has to be achieved by deleting as few vertices as possible. The following five problems are studied: delete k vertices such that (a) each of the given ℓ terminals is separated from the others, (b) each of the given ℓ pairs of terminals is separated, (c) exactly ℓ vertices are cut away from the graph, (d) exactly ℓ connected vertices are cut away from the graph, (e) the graph is separated into at least ℓ components. We show that if both k and ℓ are parameters, then (a), (b) and (d) are fixed-parameter tractable, while (c) and (e) are W[1]-hard.

1 Introduction

In this paper we study five problems where we have to delete at most k vertices from a graph to achieve a certain goal. In all five cases, the goal is related to making the graph disconnected: either certain vertices have to be separated from each other, or certain components have to be created.

Before defining the five problems, let us discuss our research methodology. The separation problems we study are known to be NP-hard, but here we investigate them from the parameterized complexity point of view. Since the solution is a set of k vertices and it is easy to verify whether a solution is correct, these problems can be solved by enumerating and verifying all the $O(n^k)$ sets of size k. Therefore, for every fixed value of k, the problems can be solved in polynomial time. However, this way of solving the problems is not practical even for, say, $k = 10$. In parameterized complexity, we are interested in the question whether a uniformly polynomial-time algorithm can be given for the problem, that is, an algorithm where in the running time the parameter k does not appear in the exponent of n (e.g., $O(2^k \cdot n)$). In this case we say that the problem is fixed-parameter tractable. There is a large number of standard techniques in the literature for designing uniformly polynomial-time algorithms. On the negative side, the theory of W[1]-hardness gives us methods to show that a problem is not fixed-parameter tractable (under some standard complexity-theoretic assumptions). This means that every algorithm has to search essentially the whole n^k search

[*]Dept. of Computer Science and Information Theory, Budapest University of Technology and Economics (dmarx@cs.bme.hu). Research is supported in part by grants OTKA 44733, 42559, and 42706 of the Hungarian National Science Fund.

Problem	Parameter(s)		
	k	ℓ	k and ℓ
MINIMUM NODE MULTIWAY CUT	FPT (Theorem 3.7)	NP-hard for $\ell \geq 3$ [5]	FPT (Theorem 3.7)
MINIMUM NODE MULTICUT	Open	NP-hard for $\ell \geq 3$ [5]	FPT (Theorem 3.9)
CUTTING ℓ VERTICES	W[1]-hard (Theorem 4.1)	W[1]-hard (Theorem 4.1)	W[1]-hard (Theorem 4.1)
CUTTING ℓ CONNECTED VERTICES	W[1]-hard (Theorem 4.5)	W[1]-hard (Theorem 4.4)	FPT (Theorem 4.3)
CUTTING INTO ℓ COMPONENTS	W[1]-hard (Theorem 4.6)	W[1]-hard (Theorem 4.6)	W[1]-hard (Theorem 4.6)

Table 1: Complexity of the problems with different parameterizations.

space. The most important notions of parameterized complexity are summarized in Section 2. For further background, the reader is referred to the monograph of Downey and Fellows [8].

Classical flow theory gives us a way of deciding in polynomial time whether two vertices t_1 and t_2 can be disconnected by deleting at most k vertices. However, for every $\ell \geq 3$, if we have ℓ terminals $t_1, t_2, \ldots, t_\ell$, then it is NP-hard to find k vertices such that no two terminals are in the same component after deleting these vertices [5]. In [10] a $(2 - 2/\ell)$-approximation algorithm was presented for this problem. Here we give an algorithm that is efficient if k is small: in Section 3 it is shown that the MINIMUM NODE MULTIWAY CUT problem is fixed-parameter tractable with parameter k. We also consider the more general MINIMUM NODE MULTICUT problem where ℓ pairs $(s_1, t_1), \ldots, (s_\ell, t_\ell)$ are given, and it has to be decided whether there is a set of k vertices whose deletion separates each of the ℓ pairs. We show that this problem is fixed-parameter tractable if both k and ℓ are parameters. Our results go through for the edge deletion versions of these problems as well.

In Section 4 we consider two separation problems without terminals. In the CUTTING ℓ VERTICES problem exactly ℓ vertices have to be separated from the rest of the graph by deleting at most k vertices. In CUTTING INTO ℓ COMPONENTS problem k vertices have to be deleted such that the remaining graph has at least ℓ connected components. The edge deletion variants of these problems were considered in [7], where it is shown that these problems are W[1]-hard with parameter ℓ. Here we show that the vertex deletion versions of both problems are W[1]-hard even if both k and ℓ are parameters. However, in the case of CUTTING ℓ VERTICES if we restrict the problem to bounded-degree graphs, then it becomes fixed-parameter tractable if both k and ℓ are parameters. Moreover, we also consider the variant CUTTING ℓ CONNECTED VERTICES, where it is also required that the separated vertices form a connected subgraph. It turns out that this problems is fixed-parameter tractable if both k and ℓ are parameters, but W[1]-hard if only one of them is parameter.

The results of the paper are summarized in Table 1.

2 Parameterized Complexity

In parameterized complexity we are dealing with *parameterized problems,* where every input instance (x, k) has a distinguished part k called the *parameter.* For example, in the Maximum Clique problem the parameter k is the size of the clique to be found. Problems such as Maximum Clique, Minimum Vertex Cover, and Longest Path can be solved trivially by trying all the $O(n^k)$ possibilities for the solution. However, such an algorithm is not really practical: n^k can be huge even for moderate values of n and small values of k. Therefore, we are interested in the question whether there is an algorithm where k does not appear in the exponent of n. We say that a parameterized problem is *fixed-parameter tractable (FPT)* if it has an algorithm with running time $f(k)n^c$, where c is a constant independent of k and n, and f depends only on k. Such an algorithm can be useful even for large values of n, provided that $f(k)$ is relatively small and c is a small constant. It turns out that several NP-hard problems, e.g., Minimum Vertex Cover, Longest Path, k-Disjoint Triangles, etc. are fixed-parameter tractable. There is a standard toolbox of techniques for designing FPT algorithms: kernelization, bounded search trees, color coding, well-quasi-ordering, just to name some of the more important ones (see [8] and [14]).

The theory of NP-completeness can be used to show that certain problems are unlikely to be polynomial-time solvable. In parameterized complexity, W[1]-hardness plays an analogous role: by showing that a problem is W[1]-hard, we can give strong evidence that the problem is not fixed-parameter tractable. We omit the somewhat technical definition of the complexity class W[1], see [8] for details. Here it will be sufficient to know that there are several problems, including Maximum Clique, that were proved to be W[1]-hard. Furthermore, we also expect that there is no $O(n^{o(k)})$ algorithm for Maximum Clique: recently it was shown that there exists an $O(n^{o(k)})$ algorithm for Maximum Clique if and only if there are subexponential-time algorithms for 3-Sat (see [4] and [11]).

To prove that a parameterized problem Q' is W[1]-hard, we have to present a parameterized reduction from a known W[1]-hard problem Q to Q'. A *parameterized reduction* from problem Q to problem Q' is a function that transforms a problem instance (x, k) of Q into a problem instance (x', k') of Q' such that

- $(x', k') \in Q'$ if and only if $(x, k) \in Q$,
- k' is a function of k independent of x, and
- the transformation can be computed in time $f(k) \cdot |x|^c$ for some constant c and function $f(k)$.

It is easy to see that if there is a parameterized reduction from Q to Q', and Q' is fixed-parameter tractable, then it follows that Q is fixed-parameter tractable as well.

3 Separating Terminals

The parameterized terminal separation problem studied in this section is formally defined as follows:

Minimum Node Multiway Cut	
Input:	A graph $G(V, E)$, a set of terminals $T \subseteq V$, and an integer k.
Parameter 1:	k
Parameter 2:	$\ell := |T|$, the number of terminals
Question:	Is there a set of vertices $S \subseteq V$ of size at most k such that no two vertices of T belong to the same connected component of $G \setminus S$?

Note that S and T do not have to be disjoint, which means that it is allowed to delete terminals. A deleted terminal is considered to be separated from all the other terminals (later we will argue that our results remain valid for the slightly different problem where the terminals cannot be deleted).

The graph minor theory of Robertson and Seymour gives a quick way of showing that Minimum Node Multiway Cut is fixed-parameter tractable. Here we briefly sketch the main idea, we refer the reader to [8, Chapter 7] for more background on the connections between graph minors and parameterized complexity.

The celebrated result of Robertson and Seymour states that graphs are well-quasi-ordered with respect to the minor relation. Moreover, the same holds for graphs where the edges are colored with a fixed number of colors. For every terminal $v \in T$, we add a new vertex v' and a red edge vv' (the original edges have color black). Now separating the terminals and separating the red edges are the same problem. Consider the set $\mathscr{G}_k$ that contains those red-black graphs where the red edges can be separated by deleting at most k vertices. It is easy to see that $\mathscr{G}_k$ is closed with respect to taking minors. Therefore, by the Graph Minor Theorem, $\mathscr{G}_k$ has a finite set of forbidden minors. Another result of Robertson and Seymour states that for every graph H there is an $O(|V|^3)$ algorithm deciding whether a graph $G(V, E)$ has a H-minor; therefore, membership in $\mathscr{G}_k$ can be tested in $O(|V|^3)$ time. This means that for every k, Minimum Node Multiway Cut can be solved in $O(|V|^3)$ time, thus the problem is (non-uniformly) fixed-parameter tractable.

The constants given by this non-constructive method are incredibly large. In this section we give a direct combinatorial algorithm for the problem, which is more efficient.

The notion of *important separator* is the most important definition in this section:

Definition 3.1. *Let $G(V, E)$ be a graph. For subsets $X, S \subseteq V$, the set of vertices reachable from $X \setminus S$ in $G \setminus S$ is denoted by $R(X, S)$. For $X, Y \subseteq V$, the set S is called an (X, Y)-*separator *if $Y \cap R(X, S) = \emptyset$. An (X, Y)-separator is* minimal *if none of its proper subsets is an (X, Y)-separator. An (X, Y)-separator S'* dominates *an (X, Y)-separator S if $|S'| \leq |S|$ and $R(X, S) \subset R(X, S')$ (proper subset). A subset S is an* important (X, Y)-separator *if it is minimal, and there is no (X, Y)-separator S' that dominates S.*

Abusing notations, the one element set $\{v\}$ will be often denoted by v. We note that X and Y can have non-empty intersection, but in this case every (X, Y)-separator has to contain $X \cap Y$.

We use Figure 1 to demonstrate the notion of important separator. Let $X = \{x\}$ and $Y = \{y_1, y_2, y_3, y_4, y_5\}$, we want to separate these two sets. X and Y can be separated by deleting x, this is the only separator of size 1. There are several separators of size 2,

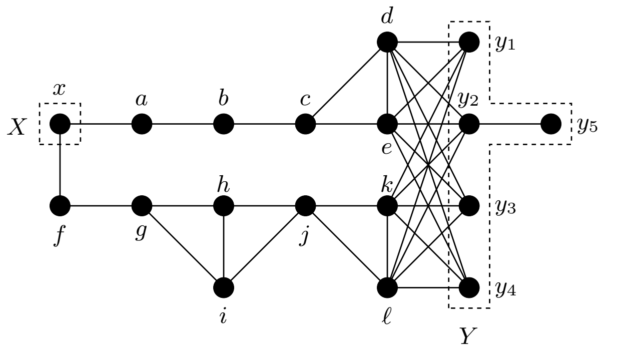


Figure 1: The important (X, Y)-separators in this graph are $\{x\}$, $\{c, j\}$, $\{c, k, \ell\}$, $\{d, e, j\}$, and $\{y_1, y_2, y_3, y_4\}$.

for example $\{a, f\}$, $\{b, g\}$, $\{b, j\}$, $\{c, j\}$. However, $\{c, j\}$ is the only important separator of size 2: $R(x, \{c, j\}) = \{x, a, b, f, g, h, i\}$ and the set of vertices reachable from x is smaller for every other separator of size 2. There are two important separators of size 3: $\{c, k, \ell\}$ and $\{d, e, j\}$. Separator $\{c, h, i\}$ is not important, since it is dominated by both $\{c, j\}$ and $\{c, k, \ell\}$. Finally, there is only one important separator of size 4: the set $\{y_1, y_2, y_3, y_4\}$. The separator $\{y_1, y_2, y_3, y_4, y_5\}$ is not important, since is not minimal.

Before we go into the technical details, let us have an intuitive overview of our algorithm for Minimum Node Multiway Cut and the motivation behind the definition of important separators. The algorithm first selects a set S_1 that separates t_1 from the rest of the terminals; next it selects a set S_2 that separates t_2 from the rest of the terminals in $G \setminus S_1$, etc. The solution is obtained as $S := S_1 \cup S_2 \cup \dots$. Of course, there are many different $(\{t_1\}, \{t_2, \dots, t_\ell\})$-separators, and it can matter a lot which separator is selected as S_1: if S_1 is chosen carefully, then besides separating t_1 and $\{t_2, \dots, t_\ell\}$, it can also help us in separating the terminals $\{t_2, \dots, t_\ell\}$ from each other. Intuitively, the closer S_1 is to $\{t_2, \dots, t_\ell\}$ and the farther it is from t_1, the more it can help in separating $\{t_2, \dots, t_\ell\}$. This motivates the definition of important separators: a separator is important, if there is no separator strictly farther from t_1. Lemma 3.6 shows that if there is a solution, then the algorithm can obtain a solution by selecting S_1, S_2, etc. to be important separators. Furthermore, Lemma 3.4 shows that there are at most 4^{k^2} important separators of size at most k. The proof of this lemma is the technically most demanding part of the algorithm. Therefore, when the algorithm chooses the separator S_i, then it has to branch into only a constant number of different directions, and the running time is uniformly polynomial.

Testing whether a given (X, Y)-separator S is important can be done using standard network flow techniques:

Lemma 3.2. *It can be checked in $O(|V|^4)$ time whether a set S is an important (X, Y)-separator in $G(V, E)$. Furthermore, if S is a minimal separator that is not important, then we can find in $O(|V|^5)$ time an important (X, Y)-separator that dominates S.*

Proof. The minimality of S can be easily checked by testing for each vertex $s \in S$ whether $S \setminus s$ remains a separator. If S is minimal, then for every vertex $s \in S$, we test whether there is an (X, Y)-separator S' of size at most $|S|$ that does not contain any vertex of

$R(X,S) \cup s$. This separator can be found in $O(|V|^3)$ time using network flow techniques (see [1]). If there is such a separator S', then S is not important: $R(X,S)$ is a proper subset of $R(X,S') \supseteq R(X,S) \cup s$.

If S is not important, then this method can be used to find an important separator that dominates S. In the previous paragraph, we have obtained a separator S' that dominates S. The test can be repeated for S', and if it is not important, then we get another separator S'' that dominates S'. We repeat this as many times as necessary. Since the set of vertices reachable from X increases at each step, eventually we arrive to an important separator. We have to repeat the test at most $O(|V|)$ times, hence the total running time is $O(|V|^5)$. ■

Let X and Y be two sets of vertices. Our algorithm for the Minimum Node Multiway Cut problem is based on the observation that the number of important (X,Y)-separators can be bounded by a function of k (Lemma 3.4). This is easy to see for $k = 1$: there is at most one important (X,Y)-separator of size 1. A separator of size 1 has to be a cut vertex (here we ignore the special cases where $|X| = 1$ or $|Y| = 1$). If there are multiple cut vertices that separate X and Y, then there is a unique cut vertex that is farthest from X and closest to Y. This vertex will be the only important (X,Y)-separator.

This observation on the number of size 1 important separators also follows from the following more general result stating that there is a unique important separator of minimum size. This result will not be needed, but we present it here for completeness.

Lemma 3.3. *If every (X,Y)-separator has size at least k, then there is at most one important (X,Y)-separator of size k.*

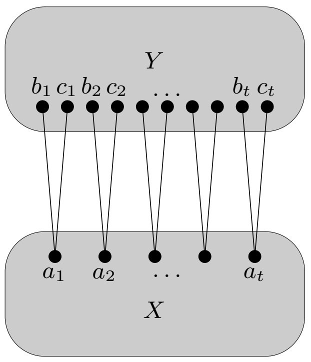
Proof. The proof uses standard techniques from the theory of network flows. For a set of vertices A, define the set $d(A)$ to contain those vertices *outside* A that either belong to X or are adjacent to a vertex in A. If S is an (X,Y)-separator, then $d(R(X,S)) \subseteq S$: the separator has to ensure that no edge leads out of $R(X,S)$ and the separator has to contain every vertex of X outside $R(X,S)$. Moreover, if S is a minimal separator, then $d(R(X,S)) = S$: there is no point in deleting any other vertex.

If vertex set W is disjoint from Y, then $d(W)$ is an (X,Y)-separator. To see this, assume that there is a path P connecting $x \in X$ and $y \in Y$ in $G \setminus d(W)$. Since $d(W)$ contains $X \setminus W$, vertex x has to be in W. The path P cannot leave W: the set $d(W)$ contains all the neighbors of W. Therefore, y also has to be in W, which is not possible.

It is a routine exercise to verify that the function d satisfies the submodular inequality

$$|d(A)| + |d(B)| \geq |d(A \cap B)| + |d(A \cup B)|. \tag{1}$$

Assume that there are two size k important (X,Y)-separators S_1 and S_2. Let $A := R(X,S_1)$ and $B := R(X,S_2)$. Notice that it is not possible that $|d(A \cap B)| < k$: as we have seen above, $d(A \cap B)$ would be an (X,Y)-separator of size less than k. The left hand side of (1) is $2k$ and $|d(A \cap B)| \geq k$, thus it follows that $|d(A \cup B)| \leq k$. The set $A \cup B$ is disjoint from Y, hence, as we have seen above, $S := d(A \cup B)$ is an (X,Y)-separator of size k. Furthermore, all the vertices of $A \cup B$ can be reached from $X \setminus S$ in $G \setminus S$ (this follows from the way A and B was defined and from the fact that S contains no vertex from $A \cup B$.) Therefore, $R(X,S) = A \cup B = R(X,S_1) \cup R(X,S_2)$, contradicting the assumption that S_1 and S_2 are important. ■



Y
b_1 c_1 b_2 c_2 $\dots$ b_t c_t
a_1 a_2 $\dots$ a_t
X

Figure 2: A graph where there is an exponential number of important separators that separate the large cliques X and Y.

For larger sizes, there can be many important (X, Y)-separators of a given size. Let us consider the example in Figure 2. To separate the two large cliques X and Y, for each $1 \le i \le t$, either a_i, or both b_i and c_i have to be deleted. If we choose to delete both b_i and c_i, then we have to delete two vertices instead of one, but the set of vertices reachable from X increases, it includes a_i. Therefore, there are $\binom{t}{t/2}$ important (X, Y)-separators of size $3t/2$: for $t/2$ of the i's we delete a_i, and for the remaining $t/2$ we delete b_i and c_i. All these separators are important, since $R(X, S')$ and $R(X, S'')$ are incomparable for two such separators S' and S''. Thus the number of important separators of a given size k can be exponential in k. However, we show that the maximum number is independent of the size of the graph:

Lemma 3.4. *If X and Y are arbitrary sets of vertices, then there are at most 4^{k^2} important (X, Y)-separators of size at most k. Moreover, these separators can be enumerated in uniformly polynomial time.*

Proof. The proof is by induction on k. We have seen above that the statement holds for $k = 1$. Let S be an important (X, Y)-separator of size at most k in G. We count how many other important separators can be in G. If H is another important (X, Y)-separator of size at most k, then we consider two cases depending on whether $Z = S \cap H$ is empty or not. If Z is not empty, then it is easy to see that $H \setminus Z$ is an important $(X \setminus Z, Y \setminus Z)$-separator in $G \setminus Z$. Since $|H \setminus Z| < k$, by the induction hypothesis the number of such separators is at most $4^{(k-1)^2}$. There are not more than 2^k possibilities for the set Z, and for each set Z there are at most $4^{(k-1)^2}$ possibilities for the set H, hence the total number of different H that intersect S is at most $2^k 4^{(k-1)^2}$.

Next we count those separators that do not intersect S. Assume that H contains ℓ vertices from $R(X, S)$ and at most $k - \ell$ vertices from $R(Y, S)$. It is not possible that $\ell = 0$: that would imply that $R(X, S) \cup S \subseteq R(X, H)$ and S would not be an important separator. Here we used the minimality of S: if no vertex of $R(X, S)$ and S is deleted, then every vertex of S can be reached from X. Similarly, it is not possible that $\ell = k$ because H would not be an important separator in that case. To see this, notice that by the minimality of S, from every vertex of S a vertex of Y can be reached using only the vertices in $R(Y, S)$. Therefore, no vertex of S can be reached from $X \setminus H$ in $G \setminus H$, otherwise H would not be an (X, Y)-separator. Since S is an (X, Y)-separator, thus this also means that no vertex of $R(Y, S)$ can be reached. Therefore, $R(X, H)$ is contained in $R(X, S)$, and the containment is proper because of $\ell > 0$. Henceforth, it is assumed that $0 < \ell < k$.

We divide H into two parts: let $H_1 = H \cap R(X, S)$ and $H_2 = H \cap R(Y, S)$ (see Figure 3). The separator S is also divided into two parts: $S_1 = S \cap R(X, H)$ contains those vertices that can be reached from $X \setminus H$ in $G \setminus H$, while $S_2 = S \setminus S_1$ contains those that cannot be reached. Let G_1 (resp., G_2) be the subgraph of G induced by $R(X, S) \cup S$ (resp., $R(Y, S) \cup S$).[1] It is clear that H_1 is an $(X \cup S_1, S_2)$-separator in G_1, and H_2 is an $(S_1, Y \cup S_2)$-separator in G_2. Moreover, we claim that they are important separators:

Claim 3.5. *H_1 is an important $(X \cup S_1, S_2)$-separator in G_1 and H_2 is an important $(S_1, Y \cup S_2)$-separator in G_2.*

Proof. First, if H_1 is not minimal, i.e., it remains an $(X \cup S_1, S_2)$-separator in G_1 without some vertex $v \in H_1$, then H would be an (X, Y)-separator without v as well. To see this, assume that there is a path P from X to Y in $G \setminus (H \setminus v)$. Let u_1 be the last vertex on P that is from $X \cup S_1$, and let u_2 be the first vertex after u_1 that is from $Y \cup S_2$. The subpath P' between u_1 and u_2 is completely contained either in G_1 or G_2, since the interior points of P' cannot contain vertices from S_1 and S_2. If P' is completely contained in G_1, then it connects a vertex of $X \cup S_1$ with a vertex of S_2, which contradicts the assumption that $H_1 \setminus v$ is an $(X \cup S_1, S_2)$-separator in G_1. Similarly, if P' is in G_2, then it connects a vertex of S_1 with a vertex of $Y \cup S_2$, which is not possible if H_2 is an $(S_1, Y \cup S_2)$-separator.

Assume now that the minimal $(X \cup S_1, S_2)$-separator H_1 is not important, because some $(X \cup S_1, S_2)$-separator H_1^* dominates H_1 in G_1. In this case $H_1^* \cup H_2$ is an (X, Y)-separator in G. This can be shown by an argument similar to the previous paragraph: if there were a path P connecting X and Y, then P would have a subpath connecting $X \cup S_1$ and S_2 in G_1 (impossible because of H_1^*) or a subpath connecting S_1 and $Y \cup S_2$ (impossible because of H_2).

Furthermore, we show that $H_1^* \cup H_2$ dominates H, contradicting the assumption that H is important. To see that $R(X, H) \subset R(X, H_1^* \cup H_2)$ holds, assume on the contrary that some vertex $u \in H_1^* \cup H_2$ is in $R(X, H)$. Clearly, u has to be in H_1^*. Let $Q \subseteq R(X, H)$ be a path connecting a vertex of X and u, this path is disjoint from H. Without loss of generality, we can assume that u is the first vertex from H_1^* on Q. Let z be the last vertex on Q from $X \cup S_1$. The subpath Q' between z and u is completely contained in G_1, since $Q \subseteq R(X, H)$ cannot go through S_2. This means that u is reachable from $X \cup S_1$ in $G_1 \setminus H_1$, but (trivially) $u \in H_1^*$ is not reachable from $X \cup S_1$ in $G_1 \setminus H_1^*$. This means that H_1^* does not dominate H_1, a contradiction.

A similar argument shows that H_2 is an important $(S_1, Y \cup S_2)$-separator in G_2. ■

By the induction hypothesis, we have a bound on the number of possible important separators H_1 and H_2. For a given partition (S_1, S_2) of S and a given value of ℓ, there are at most $4^{\ell^2} 4^{(k-\ell)^2}$ possibilities for H_1 and H_2. There are 2^k possible partitions (S_1, S_2), and

[1] As shown in Figure 4, the situation can be more complicated than what is suggested by Figure 3. It is possible that some vertex $v \in S_1$ is not reachable from X in $G_1 \setminus H_1$, but can be reached in $G \setminus H$ by going through G_2 first. Therefore in the proof it is important that we formally verify every claim and do not rely on the intuition gained from Figure 3.

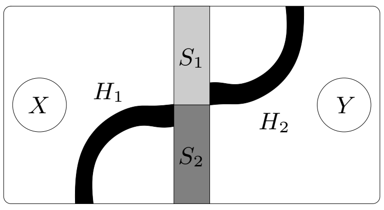


Figure 3: Separators in the proof of Lemma 3.4.

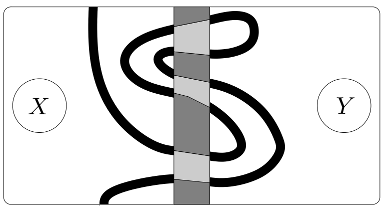


Figure 4: A more complicated example for the separators in the proof of Lemma 3.4.

the value of ℓ is between 1 and $k-1$. Therefore, the total number of different separators (including S itself and the at most $2^k 4^{(k-1)^2}$ sets from the first case) is at most

$$\begin{aligned} &1+2^k4^{(k-1)^2}+\sum_{\ell=1}^{k-1}2^k4^{\ell^2}4^{(k-\ell)^2}\le 1+2^k4^{(k-1)^2}+(k-1)2^k4^{(k-1)^2+1}\\ &\le k2^k4^{(k-1)^2+1}\le 4^k4^{(k-1)^2+1}=4^{k+k^2-2k+2}\le 4^{k^2},\end{aligned}$$

which we had to show (in the first inequality we used $\ell^2+(k-\ell)^2\le(k-1)^2+1$, which follows from $1\le\ell\le k-1$).

The proof also gives an algorithm for finding all the important separators. First, we use standard network flow techniques to find an arbitrary (X,Y)-separator S_0 of size at most k. Next we use Lemma 3.2 to find an important separator S that dominates S_0. Let us enumerate the other important separators besides S. To handle the first case of the proof, we take every subset Z of S, and recursively find all the important size $k-|Z|$ separators in $G\setminus S$. In the second case, we consider every $1\le\ell\le k-1$ and every division (S_1,S_2) of S. We recursively enumerate every important $(X\cup S_1,S_2)$-separator H_1 in G_1 that has size at most ℓ and every important $(S_1,Y\cup S_2)$-separator H_2 in G_2 that has size at most $k-\ell$. We have seen above that if an important (X,Y)-separator is disjoint from S, then it is the union of some H_1 and some H_2. For each H_1, H_2, it has to be checked whether $H_1\cup H_2$ is an important (X,Y)-separator. Checking whether H is important can be done with the algorithm of Lemma 3.2.

Our algorithm makes a constant number of recursive calls with smaller k; therefore, the running time is uniformly polynomial. ■

What makes important separators important is that a separator in a solution can be always replaced by an important separator:

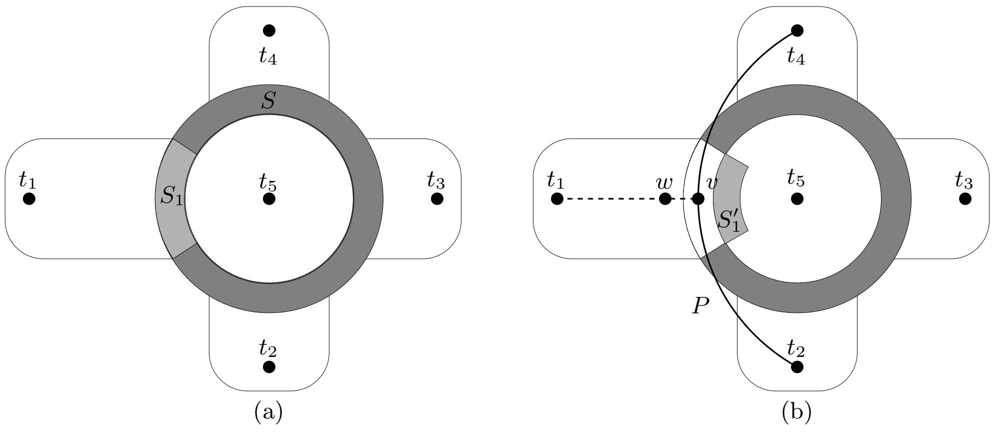


Figure 5: The proof of Lemma 3.6.

Lemma 3.6. *If there is a set S of vertices that separates the terminals $t_1, \dots, t_r$, then there is a set H with $|H| \leq |S|$ that also separates the terminals and contains an important $(\{t_1\}, \{t_2, t_3, \dots, t_r\})$-separator.*

Proof. Let $S_0 \subseteq S$ be those vertices of S that can be reached from t_1 without going through other vertices of S. Clearly, S_0 is a $(\{t_1\}, \{t_2, t_3, \dots, t_r\})$-separator, and it contains a minimal separator S_1 (see Figure 5a). If S_1 is important, then S contains the important separator S_1, and we are done. Otherwise, there is an important $(\{t_1\}, \{t_2, t_3, \dots, t_r\})$-separator S_1' that dominates S_1. We claim that $S' = (S \setminus S_1) \cup S_1'$ also separates the terminals. If this is true, then $|S_1'| \leq |S_1|$ implies $|S'| \leq |S|$, proving the lemma.

Since S_1' is a $(\{t_1\}, \{t_2, t_3, \dots, t_r\})$-separator, S' separates t_1 from all the other vertices. Assume therefore that there is a path P in $G \setminus S'$ connecting terminals t_i and t_j (see Figure 5b). Since S separates t_i and t_j, this is only possible if P goes through a vertex v of S_1. Every vertex of $S_1 \subseteq S_0$ has a neighbor in $R(t_1, S)$, let w be such a neighbor of v. Since $R(t_1, S) \subseteq R(t_1, S')$, vertex w can be reached from t_1 in $G \setminus S'$. Therefore, t_i can be reached from t_1 via w and v, which is a contradiction, since S' is a $(\{t_1\}, \{t_2, t_3, \dots, t_r\})$-separator. ∎

Lemma 3.4 and Lemma 3.6 allow us to use the method of bounded search trees to solve the Minimum Node Multiway Cut problem:

Theorem 3.7. Minimum Node Multiway Cut *is fixed-parameter tractable with parameter k.*

Proof. We choose an arbitrary terminal t that is not already separated from every other terminal. By Lemma 3.6, there is a solution that contains an important $(t, T \setminus t)$-separator. Using Lemma 3.4, we enumerate all the at most 4^{k^2} important separators of size at most k, and select a separator S from this list. We delete S from G, and recursively solve the problem for $G \setminus S$ with problem parameter $k - |S|$. At each step we can branch into at most 4^{k^2} directions, and the problem parameter is decreased by at least one, hence the search tree has height at most k and has at most 4^{k^3} leaves. The work to be done is uniformly polynomial at each step, hence the algorithm is uniformly polynomial. ∎

A natural way to generalize Minimum Node Multiway Cut is to have a more complicated restriction on which terminals should be separated. Instead of a set of terminals where every terminal has to be separated from every other terminal, in the following problem there are pairs (s_i, t_i) of terminals, and each s_i has to be separated only from the corresponding t_i:

Minimum Node Multicut	
Input:	A graph $G(V, E)$, pairs of vertices (s_1, t_1), (s_2, t_2), $\dots$, (s_ℓ, t_ℓ), and an integer k.
Parameter 1:	k
Parameter 2:	ℓ
Question:	Is there a set of vertices $S \subseteq V$ of size at most k such that for every $1 \le i \le \ell$, vertices s_i and t_i are in different components of $G \setminus S$?

Let $T = \bigcup_{i=1}^{\ell} \{s_i, t_i\}$ be the set of terminals. We can prove an analog of Lemma 3.6: there is an optimal solution containing an important separator.

Lemma 3.8. *If there is a set S of vertices that separates every pair, then there is a set S' with $|S'| \le |S|$ that also separates the pairs and S' contains an important $(\{s_1\}, T')$-separator for some nonempty subset $T' \subseteq T$.*

Proof. We proceed similarly as in the proof of Lemma 3.6. Let T' be the set of those terminals that are separated from s_1 in $G \setminus S$. Let $S_0 \subseteq S$ be the vertices reachable from s_1 without going through other vertices of S. Clearly, S_0 is an (s_1, T')-separator, and it contains a minimal (s_1, T')-separator S_1. If S_1 is not important, then there is an important (s_1, T')-separator S_1' that dominates S_1. We claim that $S' = (S \setminus S_1) \cup S_1'$ also separates the pairs. Clearly, $t_1 \in T'$, hence s_1 and t_1 are separated in S'. Assume therefore that s_i and t_i are connected by a path P in $G \setminus S'$. As in Lemma 3.6, path P goes through a vertex of S_1, and it follows that both s_i and t_i are connected to s_1 in $G \setminus S'$. Therefore, $s_i, t_i \notin T'$. However, this implies that s_1 is connected to s_i and t_i in $G \setminus S$, hence S does not separate s_i from t_i, a contradiction.

■

To find k vertices that separate the pairs, we use the same method as in Theorem 3.7. In Lemma 3.8, there are 2^ℓ different possibilities for the set T', and by Lemma 3.4, for each T' there are at most 4^{k^2} important $(\{s_1\}, T'\}$-separators of size at most k. Therefore, we can generate $2^\ell \cdot 2^{k^2}$ separators such that one of them is contained in an optimum solution. This results in a search tree with at most $2^{k\ell} \cdot 4^{k^3}$ leaves.

Theorem 3.9. *The* Minimum Node Multicut *problem is fixed-parameter tractable with parameters k and ℓ.* ■

Separating the terminals in T can be expressed as separating $\binom{|T|}{2}$ pairs, hence Minimum Node Multiway Cut is a special case of Minimum Node Multicut. However, Theorem 3.9 does not imply Theorem 3.7. In Theorem 3.9 the number of pairs is a parameter, while the size of T can be unbounded in Theorem 3.7. The complexity of Minimum Node

MULTICUT if only k is the parameter remains an interesting open question. We remark that similarly to Theorem 3.7, a non-constructive proof of Theorem 3.9 follows from the graph minor theorems. However, this technique does not seem to work for MINIMUM NODE MULTICUT if ℓ is not a parameter.

As noted at the beginning of the section, in the separation problems we assume that any vertex can be deleted, even the terminals themselves. However, we can consider the slightly more general problem when the input contains a set V^* of distinguished vertices, and these vertices cannot be deleted. All the results in this section hold for this variant of the problem as well. In all of the proofs, when a new separator is constructed, then it is constructed from vertices that were contained in some other separator.

We can consider the variants MINIMUM MULTIWAY CUT and MINIMUM MULTICUT where the terminals have to be separated by deleting at most k edges. The edge deletion problems received more attention in the literature [6, 5, 9]. As noted in [10], it is easy to reduce the edge deletion problem to vertex deletion; therefore, our algorithms can be used for these edge deletion problems as well. For completeness, we briefly describe a possible reduction. The edge deletion problem can be solved by considering the line graph (in the line graph $L(G)$ of G the vertices correspond to the edges of G, and two vertices are connected if the corresponding two edges have a common vertex.) However, we have to do some tinkering before we can define the terminals in the line graph. For each terminal v_i of G, add a new vertex v'_i and a new edge $v_iv'_i$. Let v'_i be the terminal instead of v_i. If edge $v_iv'_i$ is marked as unremovable, then this modification does not change the solvability of the instance. Now the problem can be solved by using the vertex separation algorithms (Theorem 3.7 and 3.9) on the line graph $L(G)$. The terminals in the line graph are the vertices corresponding to the edges $v_iv'_i$. These edges were marked as unremovable in G, hence the corresponding vertices have to be included in the set V^* of distinguished vertices in $L(G)$.

Theorem 3.10. *The edge deletion versions of* MINIMUM MULTIWAY CUT *(with parameter k) and* MINIMUM MULTICUT *(with parameters k and ℓ) are fixed-parameter tractable.* ■

4 Cutting up a Graph

Finding a good separator that splits a graph into two parts of approximately equal size is a useful algorithmic technique (see [12, 13] for classic examples). This motivates the study of the following problem, where a given number of vertices has to be separated from the rest of the graph:

CUTTING ℓ VERTICES	
Input:	A graph $G(V, E)$, integers k and ℓ.
Parameter 1:	k
Parameter 2:	ℓ
Question:	Is there a partition $V = X \cup S \cup Y$ such that $\lvert X\rvert = \ell$, $\lvert S\rvert \leq k$ and there is no edge between X and Y?

It follows from [3] that the problem is NP-hard in general. Moreover, it is not difficult to show that the parameterized version of the problem is hard as well, even with both parameters:

Theorem 4.1. Cutting ℓ Vertices *is* W[1]*-hard with parameters k and ℓ.*

Proof. The proof is by reduction from the parameterized Maximum Clique problem. Let G be a graph with n vertices and m edges, it has to be determined whether G has a clique of size k. We construct G' as follows. In G' there are n vertices $v_1, \dots, v_n$ that correspond to the vertices of G, they form a clique in G'. Furthermore, G' has m vertices $e_1, \dots, e_m$ that correspond to the edges of G. If the end points of edge e_j in G are vertices v_{j_1} and v_{j_2}, then connect vertex e_j with vertices v_{j_1} and v_{j_2} in G'. Set $\ell' := \binom{k}{2}$ and $k' := k$.

If there is a clique of size k in G, then we can cut away ℓ' vertices in G' by removing k' vertices. From $v_1, \dots, v_n$ remove those k vertices that correspond to the clique. Now the $\binom{k}{2}$ vertices of G' that correspond to the edges of the clique are isolated vertices. On the other hand, assume that in G' it is possible to cut away ℓ' vertices by deleting k' vertices. The remaining vertices of $v_1, \dots, v_n$ form a clique of size greater than ℓ' (assuming $n > \binom{k}{2} + k$), hence the ℓ' separated vertices correspond to ℓ' edges of G. These vertices have to be isolated, since they cannot be connected to the large clique formed by the remaining v_i's. This means that the end vertices of the corresponding edges were all deleted. Therefore, these $\ell' = \binom{k}{2}$ edges can have at most $k' = k$ end points, which is only possible if the end points induce a clique of size k in G. ∎

If we restrict the problem to bounded-degree graphs, then Cutting ℓ Vertices becomes fixed-parameter tractable:

Theorem 4.2. Cutting ℓ Vertices *is fixed-parameter tractable with parameters k, ℓ, and d, where d is the maximum degree of the graph.*

Proof. Consider a solution $V = X \cup S \cup Y$, and consider the subgraph induced by $X \cup S$. This subgraph consists of some number of connected components, let $X_i \cup S_i$ be the vertex set of the i-th component. For each i, the pair (S_i, X_i) has the following two properties:

(1) in graph G the set S_i separates X_i from the rest of the graph, and

(2) $X_i \cup S_i$ induces a connected graph.

On the other hand, assume that the pairs $(X_1, S_1), \dots, (X_t, S_t)$ satisfy (1), (2), and the sets $X_1, \dots, X_t, S_1, \dots, S_t$ are pairwise disjoint. In this case $S = S_1 \cup \cdots \cup S_t$ separates $X = X_1 \cup \cdots \cup X_t$ from the rest of G. Furthermore, if X has size exactly ℓ and S has size at most k, then they form a solution. Therefore, to solve the problem we generate all the pairs that satisfy these requirements, and use color coding to decide whether there are disjoint pairs with the required total size (see below for details). If there is a solution, then this method will find one.

By requirement (2) a pair (X_i, S_i) induces a connected subgraph of size at most $k+\ell$. We enumerate every such connected subgraph. If a vertex v is contained in a connected subgraph of size at most $k+\ell$, then every vertex of the subgraph is at distance less than $k+\ell$ from v. The maximum degree of the graph is d, thus there are at most $d^{k+\ell}$ vertices at distance less than $k+\ell$ from v. Therefore, the number of connected subgraphs that contain v and have size at most $k+\ell$ is a constant (depending only k, ℓ, and d), which means that there is a linear number of such subgraphs in the whole graph. We can enumerate these subgraphs in linear time. Each subgraph can be divided into a pair (X_i, S_i) in at most $\binom{k+\ell}{k} = O(2^{k+\ell})$ different ways. From these pairs we retain only those that satisfy requirement (1).

Having generated all the possible pairs (X_1, S_1), ..., (X_p, S_p), a solution can be found as follows. We consider a random coloring of the vertices with a set C of $c := k + \ell$ colors. Using dynamic programming, we try to find a solution where every vertex of $X \cup S$ has a distinct color. Subproblem (C', j, k', ℓ') asks whether it is possible to select some pairs from the first j pairs such that

- the selected pairs are pairwise disjoint,
- the selected pairs use only vertices with colors in C' and every color of C' is used at most once,
- the union of the S_i's has size at most k', and
- the union of the X_i's has size ℓ'.

Clearly, there is a solution where $X \cup S$ has distinct colors if and only if the answer to subproblem (C, p, k, ℓ) is true. For $j = 0$, the subproblems are trivial. If the subproblems for $j-1$ are solved, then the problem can be solved for j using the following two recurrence relations. First, if subproblem $(C', j-1, k', \ell')$ is true, then clearly (C', j, k', ℓ') is true as well. Moreover, if every vertex of $X_j \cup S_j$ has distinct color (denote by C_j these colors), and subproblem $(C' \setminus C_j, j-1, k' - |S_j|, \ell' - |X_j|)$ is true, then a solution for this subproblem can be extended by the pair (X_j, S_j) to obtain a solution for (C', j, k', ℓ'). Using these two rules, all the subproblems can be solved.

If there is a solution $X \cup S$, then with probability at least c^{-c} (where $c = k + \ell$ is the number of colors) these vertices receive distinct colors, and the algorithm described above finds a solution. Therefore, if there is a solution, then on average we have to repeat the method at most c^c (constant) times to find a solution. The algorithm can be derandomized using the standard method of k-perfect hash functions, see [8, Section 8.3] and [2]. ■

A variant of Cutting ℓ Vertices is the Cutting ℓ Connected Vertices problem where we also require that X induces a connected subgraph of G. This problem is fixed-parameter tractable:

Theorem 4.3. *The* Cutting ℓ Connected Vertices *problem is fixed-parameter tractable with parameters k and ℓ.*

Proof. A vertex with degree at most $k + \ell$ will be called a *low degree* vertex, let G_0 be the subgraph induced by these vertices. A vertex v with degree more than $k + \ell$ cannot be part of X: at most k neighbors of v can be in S, hence v would have more than ℓ neighbors in X, which is impossible if $|X| = \ell$. Therefore, X is a connected subgraph of G_0. As in the proof of Theorem 4.2, a bounded-degree graph has a linear number of connected subgraphs of size ℓ. For each such subgraph, it has to be checked whether it can be separated from the rest of the graph by deleting at most k vertices, i.e., whether its neighborhood has size at most k. ■

However, if only k is chosen as the parameter, then the problem is W[1]-hard. This follows from the proof of Theorem 4.1. We construct the $n+m$ vertex graph as before, but instead of asking whether it is possible to separate $\binom{k}{2}$ vertices by deleting k vertices, we ask whether it is possible to separate $\ell := n + m - \binom{k}{2} - k$ connected vertices by deleting k vertices. (Notice that ℓ is not a function of k, but this is not a problem, as now ℓ is not a parameter.) It is easy to see that the two questions have the same answer, thus

Theorem 4.4. CUTTING ℓ CONNECTED VERTICES *is* W[1]*-hard with parameter* k. ■

Similarly, the problem is W[1]-hard if only ℓ is the parameter.

Theorem 4.5. CUTTING ℓ CONNECTED VERTICES *is* W[1]*-hard with parameter* ℓ.

Proof. The reduction is from MAXIMUM CLIQUE. It is not difficult to show that MAXIMUM CLIQUE remains W[1]-hard for regular graphs (by attaching appropriate gadgets, we can make the graph regular without modifying the maximum clique size). Assume that we are given an r-regular graph G, and it has to be decided whether there is a clique of size k. If $r \leq k^4$, then the problem is fixed-parameter tractable: for every vertex v, we select $k-1$ neighbors of v in at most $\binom{k^4}{k-1}$ possible ways, and test whether these k vertices form a clique. Thus in the following it will be assumed that $r > k^4$.

Consider the line graph $L(G)$ of G, i.e., the vertices of $L(G)$ correspond to the edges of G. Set $\ell' := \binom{k}{2}$ and $k' := k(r-k+1)$ (r can be up to $n-1$, but this is not a problem, since k' is not a parameter). We claim that ℓ' connected vertices can be separated from $L(G)$ by deleting k' vertices if and only if G has a size k clique. If G has a size k clique, then the ℓ' edges induced by the clique can be separated from the rest of the line graph: for each vertex of the clique, we have to delete the $r-k+1$ edges leaving the clique. On the other hand, assume that ℓ' vertices of $L(G)$ can be separated by deleting k vertices. The corresponding ℓ' edges in G span a set T of vertices of size $t \leq 2\ell'$. We show that $t = k$, thus T is a clique of size k in G. Assume that $t > k$. Each vertex of T has at least $r-t+1$ edges that leave T. The corresponding $t(r-t+1)$ vertices have to be deleted from the line graph of G, hence $k' \geq t(r-t+1)$. However, this is not possible since

$$t(r-t+1) - k' = (t-k)r - t(t-1) + k(k-1) \geq (t-k)r - 4\ell'^2 \geq r - k^4 > 0$$

(in the first inequality we use $t^2 \leq 4\ell'^2$, in the second $t > k$ and $\ell' < k^2/2$). ■

The vertex connectivity of a graph is the minimum number of vertices that has to be deleted to make the graph disconnected. Using network flow techniques, vertex connectivity can be determined in polynomial time. However, if we want to disconnect the graph into at least $\ell \geq 3$ components by deleting as few vertices as possible, then the problem becomes hard. By essentially the same proof as in Theorem 4.1, we can show hardness for this problem as well:

CUTTING INTO $\boldsymbol{\ell}$ COMPONENTS

Input:	A graph $G(V,E)$, integers k and ℓ
Parameter 1:	k
Parameter 2:	ℓ
Question:	Is there a set S of k vertices such that $G \setminus S$ has at least ℓ connected components?

Theorem 4.6. CUTTING INTO ℓ COMPONENTS *is* W[1]*-hard with parameters* k *and* ℓ.

Proof. The construction is the same as in Theorem 4.1, but this time we set $\ell' = \binom{k}{2} + 1$ and $k' = k$. By deleting the vertices corresponding to a clique of size k the graph is separated into ℓ' components. The converse is also easy to see, the argument is the same as in Theorem 4.1. ■

5 Conclusions

In this paper we have studied a bunch of loosely related graph separation problems. The common theme in all these problems is that vertices have to be deleted to make the graph disconnected in a certain way. We have shown that the Minimum Node Multiway Cut problem is fixed-parameter tractable if the number of nodes to be deleted is chosen as parameter. For the more general Minimum Node Multicut problem, we were only able to show that it is fixed-parameter tractable if both the number of pairs and the number of vertices to be deleted are parameters. It remains an intriguing open question to determine the complexity of Minimum Node Multicut in the case when only k is the parameter. This question restricted to planar graphs would be also worth studying.

Another natural generalization is to consider directed graphs. Lemma 3.4 and 3.6 go through for the directed case, giving a uniformly polynomial algorithm for the directed version of Minimum Node Multiway Cut. However, Lemma 3.8 breaks down in the directed case. Hence the complexity of the directed version of Minimum Node Multicut remains open. The problem would be interesting to study even on acyclic graphs.

In the second part of the paper, we have considered problems where "something" has to be cut away from the rest of the graph by deleting vertices. Edge-deletion variants of some of these problems were studied in [7]. However, parameterizing by the number of edges to be deleted is not investigated in [7]. It would be interesting to determine the complexity of all the possible parameterizations of the edge-deletion variants.

Acknowledgments

I'm grateful to Tibor Jordán for useful remarks, and to Ildi Schlotter for her careful reading of the manuscript.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows: theory, algorithms, and applications.* Prentice Hall Inc., Englewood Cliffs, NJ, 1993.

[2] N. Alon, R. Yuster, and U. Zwick. Finding and counting given length cycles. *Algorithmica*, 17(3):209–223, 1997.

[3] T. N. Bui and C. Jones. Finding good approximate vertex and edge partitions is NP-hard. *Inform. Process. Lett.*, 42(3):153–159, 1992.

[4] J. Chen, B. Chor, M. Fellows, X. Huang, D. Juedes, I. Kanj, and G. Xia. Tight lower bounds for certain parameterized NP-hard problems. In *Proceedings of 19th Annual IEEE Conference on Computational Complexity*, pages 150–160, 2004.

[5] W. H. Cunningham. The optimal multiterminal cut problem. In *Reliability of computer and communication networks (New Brunswick, NJ, 1989)*, volume 5 of *DIMACS Ser. Discrete Math. Theoret. Comput. Sci.*, pages 105–120. Amer. Math. Soc., Providence, RI, 1991.

[6] E. Dahlhaus, D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, and M. Yannakakis. The complexity of multiterminal cuts. *SIAM J. Comput.*, 23(4):864–894, 1994.

[7] R. Downey, V. Estivill-Castro, M. Fellows, E. Prieto, and F. Rosamund. Cutting up is hard to do. In J. Harland, editor, *Electronic Notes in Theoretical Computer Science*, volume 78. Elsevier, 2003.

[8] R. G. Downey and M. R. Fellows. *Parameterized complexity.* Monographs in Computer Science. Springer-Verlag, New York, 1999.

[9] N. Garg, V. V. Vazirani, and M. Yannakakis. Primal-dual approximation algorithms for integral flow and multicut in trees. *Algorithmica*, 18(1):3–20, 1997.

[10] N. Garg, V. V. Vazirani, and M. Yannakakis. Multiway cuts in node weighted graphs. *J. Algorithms*, 50(1):49–61, 2004.

[11] M. Grohe and J. Flum. Parameterized complexity and subexponential time. *Bull. Eur. Assoc. Theor. Comput. Sci. EATCS*, (84):71–100, 2004.

[12] R. J. Lipton and R. E. Tarjan. A separator theorem for planar graphs. *SIAM J. Appl. Math.*, 36(2):177–189, 1979.

[13] R. J. Lipton and R. E. Tarjan. Applications of a planar separator theorem. *SIAM J. Comput.*, 9(3):615–627, 1980.

[14] R. Niedermeier. Invitation to fixed-parameter algorithms, 2002. Habilitation thesis, Universität Tübingen.