EDGE DISJOINT PATHS AND MULTICUT PROBLEMS IN GRAPHS GENERALIZING THE TREES*

Cédric Bentz†

5th October 2005

Abstract

We generalize all the results obtained for maximum integer multiflow and minimum multicut problems in trees by Garg et al. [Primal-dual approximation algorithms for integral flow and multicut in trees. Algorithmica 18 (1997) 3–20] to graphs with a fixed cyclomatic number, while this cannot be achieved for other classical generalizations of the trees. Moreover, we prove that the minimum multicut problem with a fixed number of source-sink pairs is polynomial-time solvable in planar and in bounded tree-width graphs. Eventually, we introduce the class of k-*edge-outerplanar* graphs and show that the integrality gap of the maximum edge-disjoint paths problem is bounded in these graphs. We also provide stronger results for cacti ($k = 1$).

Categories and Subject Descriptors: F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms; G.2.2 [Discrete Mathematics]: Graph Theory

General Terms: Algorithms, Theory

Additional Key Words: Edge disjoint paths, integer multiflows, multicuts, multiterminal cuts, approximation algorithms

1 Introduction

In this paper, we are interested in the study of the maximum edge-disjoint paths and the minimum multicut problems in undirected graphs (no directed version is considered), as well as some of their variants. These two fundamental problems have been extensively studied, and are well-known to be $\mathcal{NP}$-hard even in very restricted classes of graphs.

*A preliminary version of a part of this paper, entitled "Edge disjoint paths and max integral multiflow/min multicut theorems in planar graphs", appeared in Proceedings ICGT'05 (2005), to appear in Electronic Notes in Discrete Mathematics.

†CEDRIC-CNAM, 292, rue Saint-Martin, 75141 Paris Cedex 03, France.
Phone: +33 (0) 1 58 80 85 50. E-mail address: cedric.bentz@cnam.fr

Assume we are given a n-vertex m-edge undirected graph $G = (V, E)$, a *capacity function* $c : E \rightarrow \mathbb{Z}^+$ and a list $\mathcal{N}$ of pairs (source s_i, sink s'_i) of *terminal* vertices. Each pair (s_i, s'_i) defines a *net* or a *commodity*. The *maximum integer multiflow problem* (MaxIMF) consists in maximizing the sum of the integral flows of each commodity (from s_i to s'_i), subject to capacity and flow conservation requirements. When $c_e = 1$ for each $e \in E$, MaxIMF turns into the *maximum edge-disjoint paths problem* (MaxEDP). When each commodity is required to be routed along a single path, MaxIMF turns into the *maximum unsplittable flow problem* (MaxUSF).

The *minimum multicut problem* (MinMC) is to select a minimum weight set of edges whose removal separates s_i from s'_i for each i. The *minimum multiterminal cut problem* (MinMTC) is a particular minimum multicut problem in which, given a set of vertices $\mathcal{T} = \{t_1, \ldots, t_{|\mathcal{T}|}\}$, the nets are (t_i, t_j) for $1 \leq i < j \leq |\mathcal{T}|$.

For $|\mathcal{N}| = 1$, the powerful Ford-Fulkerson's theorem establishes that the value of the minimum cut is equal to the value of the maximum integral flow [17]. Unfortunately, this property does not hold for larger $|\mathcal{N}|$. However, MaxIMF and MinMC do have a fundamental relationship. Both can be expressed as integer linear programs, and the continuous relaxations of their linear programming formulations are dual. One consequence is that the value of any feasible multiflow cannot exceed the value of any multicut. This property explains why approximation results sometimes relate the value of an approximately optimal multiflow to the value of a well-suited feasible multicut, instead of relating it directly to the value of an optimal multiflow.

A lot of work has been done on these problems. Although the basic problems are known to be $\mathcal{NP}$-hard for a long time, much efforts have been done in two directions: first, identifying classes of graphs or special cases where the problems become tractable; second, obtaining good polynomial-time approximation algorithms for these problems, and, in particular, deriving good integer solutions from fractional solutions (i.e., finding solutions with a small integrality gap) or designing primal-dual schemes.

Both aspects are considered in this paper. Since we are looking for valuable cases, we begin by presenting the main known results. Given an optimization problem P and a real $\alpha > 0$, a *α-approximation algorithm for P* is a polynomial-time algorithm A that always outputs a feasible solution for P such that $\max_{I \text{ / } I \text{ is an instance of } P} \left\{ \frac{\text{OPT}_I}{\text{SOL}_A(I)}, \frac{\text{SOL}_A(I)}{\text{OPT}_I} \right\} \leq \alpha$, where OPT_I is the optimum value for the instance I of problem P and $\text{SOL}_A(I)$ is the value of the solution given by A for the instance I of problem P.

Prior to the study of MaxEDP, lots of results concerned a basic $\mathcal{NP}$-complete problem, the *edge-disjoint paths problem* (EDP). Given an undirected graph and a list of nets, the problem is to decide whether it is possible to route *all* the nets along edge-disjoint paths. Obviously, whenever this decision problem is $\mathcal{NP}$-complete, MaxEDP is $\mathcal{NP}$-hard. However, solving EDP in polynomial time does not necessarily help us for dealing with MaxEDP efficiently. See [19] for an extensive survey on EDP.

On the negative side, Pfeiffer and Middendorf show that EDP remains $\mathcal{NP}$-complete even if the graph obtained by adding the edges (s_i, s'_i), $i \in \{1, \ldots, |\mathcal{N}|\}$, to the initial graph G, is planar [34] (however, if, in addition, we restrict the terminals to lie on a *bounded number of faces* of G, they prove that the problem becomes tractable). Moreover, Marx shows that EDP is $\mathcal{NP}$-complete in eulerian planar graphs with a degree bounded by 4 (by showing that it is $\mathcal{NP}$-complete in eulerian grids [33]), and Nishizeki et al. show that it is also $\mathcal{NP}$-complete in series-parallel graphs (i.e., in graphs with tree-width 2) [35].

On the positive side, Robertson and Seymour show that, when $|\mathcal{N}|$ is fixed, EDP is polynomial-time solvable in unrestricted graphs [40]. Moreover, extending a result of Okamura and Seymour [37], Frank shows that EDP is polynomial-time solvable in planar graphs, if all the terminals lie on the outer face and all the vertices not on the outer face have even degrees [18]. Note that the above class of graphs includes the planar graphs with all their vertices on the outer face, i.e., the *outerplanar graphs* (a subclass of the series-parallel graphs).

We turn back to MaxEDP. In their seminal paper, Garg, Vazirani and Yannakakis show that MaxEDP is polynomial-time solvable in trees [23]. However, they also show that, in trees with capacities 1 and 2, MaxIMF is $\mathcal{NP}$-hard and APX-hard. By replacing each edge of capacity 2 by two parallel paths of length two, each containing only edges with capacity 1, this implies that MaxEDP is $\mathcal{NP}$-hard and APX-hard in outerplanar graphs having all their *edges* lying on the outer face. Moreover, Even, Itai and Shamir show that, even if $|\mathcal{N}| = 2$, MaxEDP is $\mathcal{NP}$-hard in unrestricted graphs [16]. It can be noticed that, if $|\mathcal{N}|$ is fixed and the degrees of the vertices are bounded by a constant, then MaxEDP can be solved in polynomial time by calling a constant number of times the algorithm of Robertson and Seymour [40]. This is also true if we consider the problem of linking by edge-disjoint paths as many nets as possible (i.e., if we consider MaxUSF with unit capacities). However, to the best of our knowledge, *in planar graphs*, MaxEDP *remains open if* $|\mathcal{N}|$ *is fixed* (although the variant where one requires vertex-disjoint instead of edge-disjoint paths is known to be tractable [24]). Note that the case where $|\mathcal{N}| = 2$ and adding the edges (s_i, s'_i), $i \in \{1, \ldots, |\mathcal{N}|\}$, does not destroy planarity, is tractable [31].

We now look at approximability results. Some important ones are known for MaxEDP, MaxIMF and MaxUSF. In general graphs, there is an $O(\min(\sqrt{m}, n^{2/3}))$-approximation algorithm for MaxEDP and MaxUSF [8, 30], although the stronger (and very recent) known inapproximability result is that both cannot be approximated within $(\log m)^{1/3-\epsilon}$ for every $\epsilon > 0$ [1]. For planar graphs, the approximation ratio is $O(\sqrt{n})$, while only APX-hardness is known. Both a greedy algorithm (denoted by SPF, for

Shortest Paths First) and a rounding based algorithm achieve this ratio [3]. Furthermore, it is important to note that there are families of planar graphs where the integrality gap is $\Theta(\sqrt{n})$ [23]. For more restricted classes of graphs, however, constant- or logarithmic-factor approximation algorithms are known: for MaxIMF, a 2-approximation algorithm in trees [23] and an $O(log(|\mathcal{N}|)/\epsilon)$-approximation (resp. an $O(1/\epsilon)$-approximation) in graphs (resp. in planar graphs) where any multicut has a value at least $\epsilon \sum_{e \in E} c(e)$ [36]; for MaxEDP, a 3-approximation in trees of rings [15], an $O(1)$-approximation (resp. an $O(\log n)$-approximation) in densely embedded (resp. in high-diameter) and nearly eulerian planar graphs (including the two-dimensional mesh) [26, 27], a 9-approximation in complete graphs [7] and an $O(F)$-approximation for graphs with *flow number* F (see [29] for details). Moreover, for high-capacitated networks (i.e., for graphs where all the capacities are $\Omega(\log n)$), an $O(1)$-approximation can be achieved for MaxIMF by randomized rounding techniques [38]. In expander graphs, a general result on the connectivity between pairs of vertices is given in [21]. In planar graphs where all the capacities are at most two, a recent paper of Chekuri et al. proposes an $O(\log n)$-approximation algorithm for MaxIMF based on a continuous relaxation [9, 10]. Still, it can be noticed that few approximation results are available, due to the noticeable difficulty to design good approximation algorithms for these problems.

Now, let us consider the MinMC problem. Garg, Vazirani and Yannakakis show that it is $\mathcal{NP}$-hard and APX-hard even in unweighted stars, but that it can be solved in polynomial time in trees if $|\mathcal{N}|$ is fixed, and approximated within a factor of 2 otherwise [23]. Moreover, Dahlhaus et al. show that MinMTC (and thus MinMC) is $\mathcal{NP}$-hard in unrestricted graphs, even if $|\mathcal{N}| = 3$ [12]; in planar graphs, MinMTC is polynomial-time solvable if $|\mathcal{N}|$ is fixed [12, 25, 42], and $\mathcal{NP}$-hard otherwise [12]. Nevertheless, the integrality gap for MinMC is $O(\log |\mathcal{N}|)$ in general graphs [22] and $O(1)$ in planar graphs [41], and there exist polynomial-time algorithms achieving these ratios. Furthermore, Călinescu et al. give a polynomial-time approximation scheme for MinMC in unweighted graphs of bounded tree-width and bounded degree, and show that dropping any of these three assumptions leads to APX-hardness (instead of $\mathcal{NP}$-hardness only) [6].

In [23], Garg et al. give a primal-dual scheme showing, in particular, that the integrality gap for MaxIMF is at most 2 in trees, and exhibit an example showing that, even in planar graphs, this gap can be quite large in general. This raises the question of finding classes of graphs where this gap is small. Actually, there are several motivations to our paper. First, trying to generalize the results of Garg et al., i.e., looking for classes of graphs that generalize the trees and where all (or a main part of) their results remain true, and trying to understand what makes these problems much easier on trees (is it a structural property? Or merely a key parameter that is small in trees?) Second, trying to identify a parameter (or some parameters) that makes MaxEDP tractable if we bound it (or them), and $\mathcal{NP}$-hard otherwise. And third, finding special cases generalizing the trees and specializing, in some sense, the example given in page 17 in [23], and where the integrality gap remains bounded for MaxEDP. The first and the third motivations have been strongly inspired by the work of Garg et al., and the second motivation has revealed to be closely related to the first one.

A natural way of generalizing the trees is to consider graphs with bounded tree-width [39]. Another generalization is to consider planar graphs where the terminals lie on a fixed number of faces [34]. However, as mentioned above, Garg et al. show that MaxEDP remains $\mathcal{NP}$-hard and APX-hard in outerplanar graphs, which have tree-width at most 2 [5], and in which the terminals all lie on one face (the outer one). In addition, their polynomial reduction remains valid even if we restrict ourselves to graphs having a bounded degree inside each 2-vertex-connected component.

Our first result is that *all* the results presented in [23] can be generalized, in some sense, to graphs with a fixed cyclomatic number (a tree being a graph with cyclomatic number 0). In particular, we prove that MaxEDP is polynomial-time solvable in such graphs, and that the integrality gap for MaxIMF is bounded by two times one plus the cyclomatic number. We also show that, for fixed $|\mathcal{N}|$, MinMC is polynomial-time solvable in planar and in bounded tree-width graphs. Although bounding the maximum degree and having all the terminals lying on one face do not lead to a bounded integrality gap for MaxEDP [23], our second result is that the integrality gap for MaxEDP is bounded in k-outerplanar graphs having a bounded degree inside each 2-vertex-connected component. Such graphs obviously generalize the trees, but also specialize the example given in page 17 in [23], where each degree is bounded by 3 and the graph is planar but not k-outerplanar. To prove this second result, we introduce the *k-edge-outerplanar* graphs, which form a subclass of the k-outerplanar graphs, and then we apply on a particular spanning tree the approximation algorithm given in [23]. We also consider the cacti, a class of graphs that generalize the trees of rings, and show that, in this case, we can bound the integrality gap for MaxIMF.

The paper is organized as follows. In Section 2, we give or recall some definitions, notions and preliminary results that will be useful in the next sections. In Section 3, we detail our results concerning graphs with a fixed cyclomatic number, showing how to generalize the work of Garg, Vazirani and Yannakakis. We also detail our results concerning MinMC. Then, Section 4 deals with the integrality gap of MaxEDP in k-edge-outerplanar graphs. Finally, in Section 5, we formulate a conjecture about the integrality gap of MaxIMF in the graphs studied in [18].

2 Preliminaries

2.1 Definitions

Given $k \geq 1$, a k-outerplanar graph is a planar graph having an embedding with at most k *layers* of vertices, i.e., such that, after removing iteratively the vertices (and their adjacent edges) lying on the outer face at most k times, we obtain the empty graph [2]. The class of k-outerplanar graphs is very well-known to be an important class of planar graphs with bounded tree-width [5].

In this paper, we introduce the class of k-*edge-outerplanar* graph, which has been inspired by the above class of graphs. Given $k \geq 1$, a k-edge-outerplanar graph is a planar graph having an embedding with at most k layers of edges, i.e., such that, after removing iteratively the edges lying on the outer face at most k times, we obtain a graph with no edge. We will detail in Section 4.1 the relationships between k-outerplanar and k-edge-outerplanar graphs. Note that the $2k \times N$ planar mesh ($N > 2k$) is both k-outerplanar and k-edge-outerplanar.

In particular, an *edge-outerplanar* graph (resp. an *outerplanar* graph) is a planar graph containing at least one edge (resp. one vertex) and having an embedding with all its *edges* (resp. *vertices*) lying on the outer face.

A graph (or one of its components) is called 2-*vertex-connected* (resp. 2-*edge-connected*) iff for any two of its vertices there are at least two paths between them that do not share any vertices (resp. any edges). A *block* is an inclusionwise maximal 2-vertex-connected component of a graph.

We also need to define formally the notion of *inside degrees*. Recall that the *degree* of a vertex is the number of vertices adjacent to it. Given a graph, one of its 2-vertex-connected components $2VCC$, and a vertex v lying in a block, the degree of v *inside* $2VCC$, denoted by $deg_{2VCC}(v)$, is the number of vertices lying in $2VCC$ that are adjacent to v. Note that a vertex can have a bounded inside degree and an unbounded degree (the converse being obviously false).

Now, let us define two other classes of graphs. Given two integers $k \geq 1$ and $d \geq 2$, the class of k-outerplanar connected graphs having a degree bounded by d inside each block will be denoted by $OPBID_{k,d}$. Given an integer $\gamma \geq 0$, the class of connected graphs $G = (V, E)$ with a cyclomatic number $\nu(G) = |E| - |V| + 1$ smaller than or equal to γ will be denoted by S_γ (these graphs being very *sparse* since $|E| \leq |V| - 1 + \gamma$). Note that each connected planar graph with $\gamma' \leq \gamma$ internal faces is in S_γ (in particular, S_0 represents the trees).

Given a graph G and a list of nets $\{(s_1, s'_1), \ldots, (s_{|\mathcal{N}|}, s'_{|\mathcal{N}|})\}$ on its vertices, we denote by P_i the set of elementary paths linking s_i to s'_i in G, for $i \in \{1, \ldots, |\mathcal{N}|\}$. Moreover, let $P = \bigcup_{i \in \{1,\ldots,|\mathcal{N}|\}} P_i$. A *flow path* is a path carrying at least one unit of flow of any commodity. Note that all the graphs considered in this paper are *simple* (i.e., with no parallel edges), *loopless* and *connected* (if this is not the case, we consider each connected component independently).

Eventually, we need two simple notation rules. Given a multicut C and a multiflow F, we shall denote by $\|C\|$ and $\|F\|$ their respective *values*. Given a graph G and a subset R of the edge set of G, let $G \setminus R$ denote the graph obtained from G by removing all the edges in R from the edge set of G.

2.2 A simple approximation algorithm

Recall that the approximation ratio of the greedy algorithm SPF is $O(\min(\sqrt{m}, n^{2/3}))$, i.e., $O(\sqrt{n})$ in planar graphs. This simple algorithm iteratively routes the shortest available path in P (it was introduced in [28]). Moreover, there exist families of trees where this bound is reached. We give one family here. Start with a path $v_1, v_2, \ldots, v_{p+2}$ of length $p+1$. Then, add a path of length $p+1$ from v_i to s_i for each $i \in \{2, \ldots, p+1\}$. Eventually, let s_1 lie on v_1, let s'_1 lie on v_{p+2} and let s'_i lie on v_{i+1} for each $i \in \{2, \ldots, p+1\}$. This graph has $\Theta(p^2)$ vertices (i.e., $p = \Theta(\sqrt{n})$) and $p+1$ nets (i.e., $|\mathcal{N}| = p+1$), and the path from s_i to s'_i has length $p+2$, for each $i \in \{2, \ldots, p+1\}$. SPF routes (s_1, s'_1) (which has length $p+1$), while the optimal solution is to route $(s_2, s'_2), \ldots, (s_{p+1}, s'_{p+1})$. Furthermore, the graph is a tree and the new graph obtained by adding the $p+1$ edges (s_i, s'_i) is planar. Note that this instance can easily be transformed into a non trivial one (i.e., such that the optimal value is neither $O(1)$ nor $\Theta(|\mathcal{N}|)$).

Hence, even for restricted classes of graphs, we have to look for better approximation algorithms. Given a connected graph G, several of our results use the same basic idea: computing a spanning tree of G in order to use the results given in [23] for trees. Since we shall use a new simple algorithm based on this idea several times, we give it here. It can be viewed as a primal-dual scheme containing three steps:

1. Compute a spanning tree T of G;
2. Use the primal-dual algorithm given in [23] which constructs an integer multiflow F_T and a multicut C_T for T such that $\|C_T\| \leq 2\|F_T\|$;
3. Build a multicut C_G for G satisfying $\|C_G\| \leq \alpha\|C_T\|$ for a fixed $\alpha > 0$.

At the end of this algorithm, we obtain an integer multiflow F_T and a multicut C_G such that $\|C_G\| \leq 2\alpha\|F_T\|$. Noticing that F_T is also feasible for G, this yields 2α-approximation algorithms for both MaxIMF and MinMC. Obviously, the first step may have to be done with some care, and the third one as well (even if our purpose is not to find the best possible α). Note that Step 3 is only relevant to prove the approximation ratio: if one is only interested in computing an approximately optimal flow, only the two first steps are necessary. Also note that, to the best of our knowledge, this is the first attempt to generalize the constant bound of the maximum integer multiflow / minimum multicut theorem of Garg et al. in trees [23]. Indeed, the family of trees given at the beginning of this section has an $\Omega(\sqrt{n})$ flow number (since any path has length $\Omega(\sqrt{n})$) and the minimum multicut uses $O(\sqrt{n})$ edges, hence neither the results in [29] nor the results in [36] provide a constant bound.

3 Graphs with a fixed cyclomatic number

In this section, we generalize the results of [23] from the trees to the graphs in S_γ. We prove that MaxEDP can be solved in polynomial time for the graphs in S_γ, by solving $O((2^\gamma|\mathcal{N}|+1)^\gamma)$ instances on a set of trees. We also show how to apply these ideas to get an approximation algorithm for MaxEDP in *quasi-complete graphs*. Then, given a graph G in S_γ, we show how to compute an integer multiflow F_G and a multicut C_G such that $\|C_G\| \leq 2(\gamma+1)\|F_G\|$, by using the algorithm given in Section 2.2. Finally, we show that MinMC can be solved in $O(m^{2^\gamma|\mathcal{N}|})$ time for the graphs in S_γ, which is polynomial in m if $|\mathcal{N}|$ is fixed. Using a totally different approach, we then show that much stronger results actually hold for MinMC. For the sake of simplicity, we do not systematically try to optimize the constants used in our analysis.

3.1 Solving MaxEDP

Garg et al. show that MaxEDP is polynomial-time solvable in trees. We use this result to design a polynomial-time algorithm solving MaxEDP in graphs of S_γ.

Let G be a graph in S_γ. We remove γ edges from G, so that the resulting graph is a spanning tree, by iteratively picking an edge from a block. Let these edges be $e_1, \ldots, e_\gamma$. The main idea is that, since γ is fixed, there is a bounded number of edges that has to be considered. For each one of these γ edges, we select either no path or one elementary path that crosses it, and remove this edge and all the other edges crossed by the (possibly) selected path. We have to be careful to select only compatible (i.e., edge-disjoint) paths: for instance, if we select a path p crossing e_i, we must also select p for e_j, $i \neq j$, if p crosses e_j. After we do this for the γ edges, we obtain a forest. We compute an optimal solution for MaxEDP in this forest by using the algorithm of Garg et al. [23]. Gathering the paths selected in this solution with the ones selected previously, we obtain a solution for MaxEDP in G. We repeat this procedure until each possible combination of the elementary paths crossing $e_1, \ldots, e_\gamma$ has been tried (recall that, in fact, for every of these γ edges, we also have to try the case where no path goes through it). Keeping the best of all these solutions, we obtain the optimal solution.

Our algorithm solves $O((|P|+1)^\gamma)$ instances of MaxEDP in a forest. Thus, γ being fixed, if $|P|$ is polynomial in n and $|\mathcal{N}|$, our algorithm runs in polynomial time. The following lemma gives a bound on $|P_i|$ for each i[1]:

Lemma 1. *Given a graph G in S_γ and two vertices s_i and s'_i, the number of elementary paths $|P_i|$ linking s_i to s'_i in G is at most 2^γ.*

Proof. We proceed by induction on γ. For $\gamma = 0$, we have $|P_i| = 1$ (G is a tree). Assume this holds for $\gamma - 1$, $\gamma \geq 1$, and let us show it holds for γ. If $|P_i| = 1$, we are done. Otherwise, let v be the first vertex encountered in any elementary path from s_i to s'_i that lies in a block. We can assume w.l.o.g. that $v = s_i$ (if this is not the case, this assumption does not modify $|P_i|$). Thus, there are at least two edges, e_1 and e_2, adjacent to s_i and lying in a block. No elementary path from s_i to s'_i crosses both e_1 and e_2, hence there is an edge $e \in \{e_1, e_2\}$ such that at least half of the paths in P_i do not cross e. Moreover, if we remove e, we obtain a graph $G' \in S_{\gamma-1}$, and we can apply the induction hypothesis: there are at most $2^{\gamma-1}$ elementary paths between s_i and s'_i in G'. Hence, $|P_i| \leq 2 \cdot 2^{\gamma-1} = 2^\gamma$. Lemma 1 follows. □

Note that this result is tight: consider a path of length γ with s_i and s'_i as endpoints. Then, replace each edge by a triangle. The obtained graph is in S_γ and satisfies $|P_i| = 2^\gamma$. Moreover, Lemma 1 implies that $|P| \leq 2^\gamma |\mathcal{N}|$, and thus the algorithm given above runs in polynomial time. Hence:

Theorem 1. MaxEDP *is polynomial-time solvable for graphs in S_γ.*

Recall that MaxEDP is $\mathcal{NP}$-hard even for $|\mathcal{N}| = 2$. Nevertheless, using the results in this section, one can state:

Theorem 2. *If $|\mathcal{N}|$ is fixed,* MaxEDP *is polynomial-time solvable in graphs whose cyclomatic number is $O(\sqrt{\log n})$.*

Proof. We use the above algorithm. Recall that we have to solve $O((2^\gamma|\mathcal{N}|+1)^\gamma)$ instances of MaxEDP in a forest, where γ is the cyclomatic number. Thus, if $\gamma = O(\sqrt{\log n})$ and $|\mathcal{N}|$ is fixed, we have to solve $O(n^{O(1)})$ instances, which is polynomial in n. □

Note that Theorems 1 and 2 (and their analyses) also hold for the variant where one requires vertex-disjoint instead of edge-disjoint paths, since this problem is also polynomial-time solvable in trees [6]. Using similar ideas (i.e., adding or removing a constant number of edges), one can also derive from [7] and from [9, 10] the following results (only the first proof is given):

Proposition 1. *There is a $9(\gamma+1)$-approximation algorithm for* MaxEDP *in graphs $G = (V, E)$ with $|E| = \frac{|V|(|V|-1)}{2} - \gamma$ (or* quasi-complete graphs*).*

[1] We have not been able to determine whether this result is already known, but we give a short proof anyway for the sake of completeness.

Proof. Simply add γ edges to G to transform it into a complete graph G' and compute a solution for MAXEDP in G' using the algorithm in [7]. Then, remove the γ edges that have been added to G and the paths that cross them (there are at most γ such paths). If there were at least $\gamma + 1$ paths routed in G', we are done. Otherwise, we route any path in G and loose at most a factor γ in the value of the solution. In both cases, we obtain the desired ratio since the algorithm in [7] is a 9-approximation algorithm for MAXEDP in complete graphs. □

Proposition 2 (using [9, 10])**.** *The integrality gap for* MAXIMF *is* $O(\log n)$ *in planar graphs having a fixed number of edges with capacity* 1*.*

3.2 Bounding the integrality gap for MaxIMF

MAXIMF is $\mathcal{NP}$-hard and APX-hard for trees, and hence for graphs in S_γ. However, Garg et al. show that, given a tree T, one can compute in polynomial time an integer multiflow F_T and a multicut C_T such that $\|C_T\| \leq 2\|F_T\|$. In this section, we prove that, given a graph G in S_γ, one can compute in polynomial time an integer multiflow F_G and a multicut C_G such that $\|C_G\| \leq 2(\gamma+1)\|F_G\|$.

We use the algorithm given in Section 2.2. All we have to do is to detail how to construct a spanning tree T for G (Step 1), and then, how to construct a multicut C_G such that $\|C_G\| \leq (\gamma+1)\|C_T\|$ (Step 3).

Step 1 proceeds as follows: we construct a maximum weight spanning tree of G, using a variant of Kruskal's algorithm [32]. Hence, we iteratively pick an edge e_i having the minimum capacity among all the edges lying in blocks of $G \setminus \{e_1, \ldots, e_{i-1}\}$. This gives us a set of γ edges satisfying $c(e_1) \leq c(e_2) \leq \cdots \leq c(e_\gamma)$, and the graph $G \setminus \bigcup_{i\in\{1,\ldots,\gamma\}} e_i$ is a tree, T.

In Step 2, we compute for T an integral multiflow $F_T = F_G$ and a multicut C_T such that $\|C_T\| \leq 2\|F_T\|$. Eventually, in Step 3, we use C_T to construct a multicut C_G for G. For each edge f_j in C_T that lies in a block of G, let $\lambda(f_j)$ be the larger i such that, before the edge e_i was removed from $G \setminus \{e_1, \ldots, e_{i-1}\}$, f_j still lied in a block. Moreover, let $\lambda^* = \max_{f_j \in C_T} \lambda(f_j)$ and $f_{j^*} \in C_T$ such that $\lambda(f_{j^*}) = \lambda^*$. Then, let $C_G = C_T \bigcup_{i\in\{1,\ldots,\lambda^*\}} \{e_i\}$.

First, let us prove that $\|C_G\| \leq (\gamma+1)\|C_T\|$. We have $\|C_G\| = \|C_T\| + \sum_{i=1}^{\lambda^*} c(e_i) \leq \|C_T\| + \lambda^* c(e_{\lambda^*}) \leq \|C_T\| + \lambda^* c(f_{j^*}) \leq \|C_T\| + \lambda^*\|C_T\| = (\lambda^*+1)\|C_T\|$. Note that the inequality $c(e_{\lambda^*}) \leq c(f_{j^*})$ comes from the definitions of λ^* and f_{j^*}, and the way e_{λ^*} has been chosen.

Second, let us show that C_G is indeed a multicut for G. In fact, all we have to prove is that, given an edge belonging to $\{e_{\lambda^*+1}, \ldots, e_\gamma\}$, there is no need to pick it in C_G, i.e., there exists another path in $T \setminus C_T$ linking its two endpoints. Let (a, b) be an edge belonging to $\{e_{\lambda^*+1}, \ldots, e_\gamma\}$. There exists a path between a and b in T, since T is a spanning tree of G. Thus, if there exists no path from a to b in $T \setminus C_T$, then necessarily the path from a

to b in T contains an edge f belonging to C_T. This implies that, just before (a, b) was removed, f was lying in a block. Since $(a, b) \in \{e_{\lambda^*+1}, \dots, e_\gamma\}$, we have a contradiction. We have $\lambda^* \leq \gamma$, and hence:

Theorem 3. *The gap between the optima of* MinMC *and* MaxIMF *is bounded by* $2(\gamma + 1)$ *for the graphs in* S_γ. *Moreover, solutions for* MinMC *and* MaxIMF *achieving this ratio can be computed in polynomial time.*

Corollary 1. *The integrality gap of* MaxIMF *is bounded by* $2(\gamma + 1)$ *for the graphs in* S_γ. *Moreover, a solution for* MaxIMF *achieving this ratio can be computed in polynomial time.*

Note that Theorem 3 applies to MaxUSF as well, since the solution computed by our method is feasible for MaxUSF. Also note that, in the analysis of Theorem 3, explicitly knowing that the spanning tree constructed in Step 1 is actually maximum weighted is not necessary (and knowing how it is constructed is sufficient). Moreover, we do not know whether the bound $2(\gamma + 1)$ is tight or not (obviously, this is the case for $\gamma = 0$ [23]). Figure 1 shows an example where a weaker bound holds.

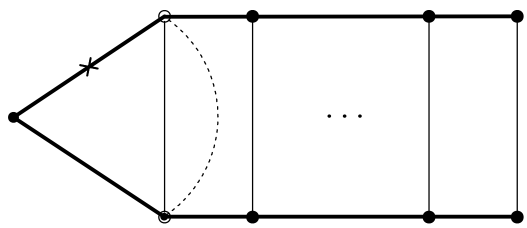


Figure 1: An example for Theorem 3 with one net (in dashed lines). The edges in bold lines (i.e., the edges forming the spanning tree) have capacity $N+1$ for some $N > 0$, while all the other edges have capacity N. So, $\|F_T\| = N + 1$, $\|C_T\| = N + 1$ (the edge denoted by $\not\!\!-$) and $\|C_G\| = \gamma N + (N + 1)$

3.3 Solving MinMC

In this section, we detail results concerning MinMC. Recall that, in trees, Garg et al. show how to compute a multicut within twice the optimum (and even within twice the value of an integral multiflow). If we consider a graph G in S_γ and assume that all its edges have capacities bounded by a small integer β, we can construct a spanning tree T as in Section 3.1, compute a multicut C_T and an integer multiflow F_T, and build a multicut C_G for G by picking all the edges in C_T and the γ edges removed from G to obtain T. Obviously, C_G satisfies $\|C_G\| \leq \|C_T\| + \gamma\beta \leq 2\|F_T\| + \gamma\beta = (2+o(1))\|F_T\|$. This gives another generalization of the approximation result obtained in [23] for trees, which is different from the one given in Section 3.2, but which only applies to graphs with small capacities.

Moreover, Garg et al. show that, in trees, MinMC can be solved in polynomial time if $|\mathcal{N}|$ is fixed. The idea is that a multicut contains at most $|\mathcal{N}|$ edges, since there is one path from s_i to s'_i, for $i \in \{1, \ldots, |\mathcal{N}|\}$: thus, MinMC can be solved in $O(m^{|\mathcal{N}|})$. For a graph G in S_γ, from Lemma 1, there are at most 2^γ paths between s_i and s'_i, for $i \in \{1, \ldots, |\mathcal{N}|\}$. Hence, MinMC can be solved in $O(m^{2^\gamma|\mathcal{N}|})$ for the graphs in S_γ, which is polynomial in m if $|\mathcal{N}|$ is fixed.

In fact, one can prove, using ideas from [12] and [42], much stronger results concerning MinMC: if $|\mathcal{N}|$ is fixed, it is polynomial-time solvable both in graphs with bounded tree-width and in planar graphs. Note that, in each case, dropping any of the two assumptions leads to $\mathcal{NP}$-hardness and even to APX-hardness. In the proof of the two following theorems, we use an idea from [12]: MinMC can be solved by solving at most $\frac{(\sqrt{2|\mathcal{N}|}+1)^{2|\mathcal{N}|}}{(\sqrt{2|\mathcal{N}|}+1)!}$ instances of MinMTC. Indeed, in any optimal solution for MinMC, the $2|\mathcal{N}|$ terminals are clustered in $q \leq \sqrt{2|\mathcal{N}|} + 1$ sets (or *clusters*), such that each one does not contain both s_i and s'_i for each i, and, for each pair of clusters, there is an i such that s_i is in one cluster and s'_i is in the other (otherwise, we can merge the two clusters and still have a valid clustering). Given a clustering, for each cluster, we add one new vertex (called a *cluster vertex*) linked by a new edge (called a *cluster edge*) to every terminal contained in this cluster in order to transform the instance of MinMC into an instance of MinMTC (where the terminals are the cluster vertices). Hence, whenever MinMTC is polynomial-time solvable, one can try all the possible clusterings and solve each one of them as a MinMTC instance (in the following, we call this idea *the clustering technique*).

Theorem 4. *If $|\mathcal{N}|$ is fixed,* MinMC *is polynomial-time solvable in graphs with bounded tree-width.*

Proof. We use the clustering technique and the fact that, according to [12], MinMTC can be solved in polynomial time in graphs with bounded tree-width, by standard dynamic programming techniques. Before adding the cluster vertices and edges, the graph has bounded tree-width by assumption, but we still have to prove that this remains true after they have been added. We use the following easy lemma:

Lemma 2. *Assume we are given a connected graph $G = (V, E)$ with tree-width $tw(G)$. Let G' be the graph obtained from G by adding a new vertex $\tilde{v}$ and edges between $\tilde{v}$ and some vertices of G. Then, $tw(G') \leq tw(G) + 1$, where $tw(G')$ denotes the tree-width of G'.*

Proof. Let $\Theta = (T, (X_w)_{w \text{ is a vertex of T}})$ be a tree decomposition of G, consisting of a tree T and a multiset whose elements X_w (called *bags*), indexed by the vertices of T, are subsets of V. Recall that Θ satisfies:

1. $\forall (u, v) \in E, \exists w$ vertex of T s. t. $u \in X_w$ and $v \in X_w$;
2. for every $v \in V$, the vertices w such that $v \in X_w$ induce a connected subgraph of T.

Recall that the *width* of Θ is equal to $\max_{w \text{ is a vertex of T}} |X_w| - 1$ and that the tree-width of G is the minimum of the widths of all tree decompositions of G. Given the decomposition Θ, we construct a tree decomposition Θ' for G' as follows: we transform each bag X_w into a new bag $X'_w = X_w \cup \{\tilde{v}\}$. The width of Θ' is equal to $\max_{w \text{ is a vertex of T}} |X'_w| - 1 = \max_{w \text{ is a vertex of T}} |X_w|$, hence it is equal to the width of Θ plus one. Moreover, Θ' satisfies 1. (because G is connected and $\tilde{v}$ is in every bag) and 2. (since the subgraph of T induced by $\tilde{v}$ is T), with respect to G'. Lemma 2 follows. $\square$

Lemma 2 implies that, if G is connected, the tree-width of the graph obtained from G by adding the cluster vertices and edges is at most the tree-width of G plus the number of cluster vertices, q. Since $q \leq \sqrt{2|\mathcal{N}|} + 1$ and $|\mathcal{N}|$ is fixed, Theorem 4 follows. If G is not connected, we consider each connected component independently, and we are done. $\square$

Theorem 5. *When $|\mathcal{N}|$ is fixed,* MinMC *is polynomial-time solvable in planar graph.*

Proof. When using the clustering technique, we need to add cluster edges, and this operation can destroy planarity. So, we cannot directly use this technique and reduce our problem to MinMTC. However, we do use the fact that, in any optimal solution for MinMC, the $2|\mathcal{N}|$ terminals are clustered in q clusters, such that each one does not contain both s_i and s'_i for each i. Hence, we try all the possible clusterings (we shall see that, in this case, a clustering can contain more than $\sqrt{2|\mathcal{N}|} + 1$ clusters) and focus on solving a subproblem of MinMC (which is equivalent to MinMTC in general graphs, but not in planar graphs): given a graph and one particular clustering of the terminals, find a minimum *multi-cluster cut*, i.e., a minimum weight set of edges whose removal separates $\mathcal{T}_i$ from $\mathcal{T}_j$ for $i \neq j$, where $\mathcal{T}_i$ is the set of terminals belonging to the i^{th} cluster. This problem (MinMCC), introduced in [14], is defined as the Colored Multiterminal Cut problem in [12], where it is proved to be $\mathcal{NP}$-hard in planar graphs, even if the number of clusters is fixed (note that the proof of Theorem 4 shows that, for a fixed number of clusters, MinMCC is polynomial-time solvable in bounded tree-width graphs). However, we shall see that the clusterings we have to deal with can be assumed to have interesting properties, and solve MinMCC in polynomial time in this case. Theorem 5 will follow.

Given a solution for MinMCC, we say that the i^{th} cluster *induces a connected component* if, after removing the edges in this solution, any vertex (including terminals) is linked to no terminal in $\mathcal{T}_i$ or to all the terminals in $\mathcal{T}_i$ (any terminal being linked to itself). The main point is the following: given an instance of MINMC, once the edges of any optimal solution $\mathcal{S}$ for this instance have been removed, each vertex is linked to at least one terminal. Therefore, the obtained graph has $q(\mathcal{S}) \leq 2|\mathcal{N}|$ connected components, each one containing at least one terminal. For each solution, one can then define a clustering as follows: for each i, the i^{th} cluster contains the terminals in the i^{th} connected component. Let $\tilde{\mathcal{S}}$ be an optimal solution for MINMC such that $q(\tilde{\mathcal{S}}) = \max_{\mathcal{S} \text{ / } \mathcal{S} \text{ is an optimal solution for MinMC}} q(\mathcal{S})$. In the following, we will focus on finding the clustering associated with $\tilde{\mathcal{S}}$. The simple example of an unweighted path $s_1, s'_1, s_2, s'_2, s_3, s'_3, \ldots, s_{|\mathcal{N}|}, s'_{|\mathcal{N}|}$ shows that $q(\tilde{\mathcal{S}})$ can be as large as $|\mathcal{N}|+1$ (we do not know whether this bound is tight or not), so we try all the possible clusterings containing at most $2|\mathcal{N}|$ clusters to make sure that we find the one associated with $\tilde{\mathcal{S}}$ (this implies that we have to consider more clusterings than we did in Theorem 4). Given this clustering, we know that every optimal solution for the instance $\mathcal{I}_{\tilde{\mathcal{S}}}$ of MINMCC defined on it will induce exactly $q(\tilde{\mathcal{S}})$ connected components (the optimal solutions for $\mathcal{I}_{\tilde{\mathcal{S}}}$ being in fact the optimal solutions for the initial instance of MINMC having the same clustering as $\tilde{\mathcal{S}}$). Now, we show that the ideas used in [42] can be applied to MINMCC in this case. Two of the key results of this paper are [42, Theorem 2] and [42, Theorem 3]. Before presenting these results, we need a few more definitions. Given a graph $G = (V, E)$ and a multiterminal cut (resp. a multi-cluster cut) separating a set of terminals $\mathcal{T} = \{t_1, \ldots, t_{|\mathcal{T}|}\}$ (resp. a set of terminal sets $\mathcal{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_{|\mathcal{T}|}\}$), let V_i be the set of vertices that remain connected to t_i (resp. to $\mathcal{T}_i$). Moreover, let C_i be the set of edges having exactly one endpoint in V_i and let D_i be its dual set of edges (i.e., the set of edges corresponding to C_i in G^D, the dual planar of G). A *dual-joint* w is a vertex of G^D adjacent to at least 3 edges of $\bigcup_i D_i$; the associated *joint-cycle* is the face corresponding to w in G. A set D_i *contains* a dual-joint w if w is adjacent to at least one edge of D_i. Yeh proved:

Theorem (implicit in [42])**.** *Let C be a minimum multiterminal cut in a graph $G = (V, E)$ and let D_i, $i \in \{1, \ldots, |\mathcal{T}|\}$, contain p dual-joints $w_1, \ldots, w_p$, i.e., there exist p vertices $\{v^*_{i1}, \ldots, v^*_{ip}\}$ such that C_i is a cut separating $\{t_i, v^*_{i1}, \ldots, v^*_{ip}\}$ from $\mathcal{T} - \{t_i\}$ and $v^*_{ij} \in V$ is a vertex in both V_i and the joint-cycle of w_j, $j = 1, \ldots, p$. If $C'_i \neq C_i$ is a cut separating $\{t_i, v^*_{i1}, \ldots, v^*_{ip}\}$ from $\mathcal{T} - \{t_i\}$, then $C' = (C - C_i) \bigcup C'_i$ is also a multiterminal cut. Moreover, this implies that C_i is a min cut separating $\{t_i, v^*_{i1}, \ldots, v^*_{ip}\}$ from $\mathcal{T} - \{t_i\}$.*

We use a suitably modified version of this theorem, but the proof is fairly similar to the one given in [42]. Namely, one can prove the following (recall that the index i refers to $\mathcal{T}_i$, the set of terminals of the i^{th} cluster, not to t_i):

Lemma 3. *Let C be a minimum multi-cluster cut in a graph $G = (V, E)$ and let D_i, $i \in \{1, \ldots, |\mathcal{T}|\}$, contain p dual-joints $w_1, \ldots, w_p$, i.e., there exist p vertices $\{v_{i1}^*, \ldots, v_{ip}^*\}$ such that C_i is a cut separating $\mathcal{T}_i \bigcup \{v_{i1}^*, \ldots, v_{ip}^*\}$ from $\mathcal{T} - \mathcal{T}_i$ and $v_{ij}^* \in V$ is a vertex in both V_i and the joint-cycle of w_j, $j = 1, \ldots, p$. If $C_i' \neq C_i$ is a cut separating $\mathcal{T}_i \bigcup \{v_{i1}^*, \ldots, v_{ip}^*\}$ from $\mathcal{T} - \mathcal{T}_i$, then $C' = (C - C_i) \bigcup C_i'$ is also a multi-cluster cut. Moreover, this implies that C_i is a min cut separating $\mathcal{T}_i \bigcup \{v_{i1}^*, \ldots, v_{ip}^*\}$ from $\mathcal{T} - \mathcal{T}_i$.*

All the remaining results proved in [42] can then be adapted for our case in a straightforward way, implying that, in this case, MinMCC can be solved in polynomial time. Simply note that we have to consider at most $2q - 4 \leq 2(2|\mathcal{N}|) - 4 = 4(|\mathcal{N}| - 1)$ dual-joints, because we know that, given the clustering associated with $\tilde{\mathcal{S}}$, every cluster necessarily induces only one connected component in any optimal solution (implicitly, this point is crucial in [42, Corollary 2] to prove the bound on the number of dual-joints, since this bound is in fact equal to 2 times the number of connected components minus 4). Moreover, our approach can be used to solve MinMCC in polynomial time when the graph is planar and the sum of the sizes of the clusters (i.e., the total number of terminals) is fixed. In general, however, an optimal solution for MinMCC can induce more than q (the number of clusters) connected components, and thus the number of dual-joints cannot be bounded by a function of q: this explains why MinMCC cannot be solved by this technique when the graph is planar and the number of clusters (but not the number of terminals) is fixed (recall that MinMCC was proved $\mathcal{NP}$-hard in this case [12]). In fact, the gap between the optimum values for MinMCC with and without the constraint that the optimal solution must contain exactly q connected components is unbounded. Indeed, consider the following MinMCC instance: the graph is a 3×3 mesh, containing 9 vertices ($t_1, t_2, t_3, t_4, t_5, t_6, a, b$ and c), 6 horizontal edges ($(t_1, a), (a, t_2), (t_3, b), (b, t_4), (t_5, c)$ and (c, t_6)) and 6 vertical edges ($(t_1, t_3), (a, b), (t_2, t_4), (t_3, t_5), (b, c)$ and (t_4, t_6)). There are three clusters: $\{t_1, t_2\}$, $\{t_3, t_4\}$ and $\{t_5, t_6\}$. The edge (b, c) and the 3 edges adjacent to a have a capacity $N \geq 5$, and the 8 other edges a capacity 1. The only cut inducing 3 connected components is formed by the 6 vertical edges, has weight $2N + 4$ and contains 1 dual-joint, while the unique minimum cut is formed by the 8 edges of capacity 1, induces 5 connected components and contains $3 > (2 * 3 - 4)$ dual-joints. Moreover, if we apply a variant of Yeh's algorithm (suggested by Lemma 3) on this graph and consider the cluster $\{t_1, t_2\}$ first, the only solution that is explored is the minimum cut with 8 edges. Therefore, the solution inducing 3 connected components cannot be found this way, implying that this algorithm cannot be used to solve MinMCC when the graph is planar, the number of clusters is fixed and we require that every cluster induces only one connected component (this problem remaining open). Also note that, if we merge $\{t_3, t_4\}$ and $\{t_5, t_6\}$

into a single cluster, we obtain an instance of MinMCC with two clusters where the optimum solution (computed, for example, by a min cut algorithm) induces 3 connected components ($\{t_1, a, t_2, b, c\}$, $\{t_3, t_5\}$ and $\{t_4, t_6\}$) and has weight 6, while the best solution inducing 2 connected components has weight $N + 2$. Therefore, even for 2 clusters, we cannot solve MinMCC by a simple min cut algorithm if only 2 connected components are required.

We make a last remark: if, during the running of our algorithm, there are two terminals of the same cluster (given the current clustering) that are no longer linked by a path, then the combination of the current clustering and of the non terminal vertices already chosen is not associated with the optimal solution we are looking for. Hence, we ignore it (without choosing non terminal vertices for the remaining clusters) and try another one. □

Using a slightly different analysis of the algorithm of Yeh [42] and a result from [11], one can also prove the following theorem for MinMTC, providing an improved running time to solve the problem:

Theorem 6. *When G is a n-vertex planar graph and the terminals lie on $\phi \leq |\mathcal{T}|$ faces of G,* MinMTC *can be solved in* $O((|\mathcal{T}|-\frac{3}{2})^{\phi-1}\cdot(n-|\mathcal{T}|)^{2|\mathcal{T}|-4}\cdot(n|\mathcal{T}|-\frac{3}{2}|\mathcal{T}|^2+\frac{1}{2}|\mathcal{T}|)\cdot\log(n-|\mathcal{T}|)+n|\mathcal{T}|^2\cdot(|\mathcal{T}|+\log n))$.

Proof. The idea behind this result is that, once all the terminals lying on one particular face of G are isolated from the rest of the terminals, we just have, to solve the problem, to compute a minimum multiterminal cut on a planar graph where all the vertices lie on the same face, which can be done by using the algorithm presented in [11]. However, the problem of identifying the set of vertices linked to the terminals on this face in an optimal solution remains. Using the work of Yeh [42], one can prove another variant of [42, Theorem 2]. We use the notations defined in the proof of Theorem 5, as well as some new ones. Let $\Phi_1, \ldots, \Phi_\phi$ be the ϕ faces where the terminals lie, and let $\mathcal{T}_{\Phi_h}$ be the set of terminals lying on Φ_h. Let V_{Φ_h} be the set of vertices that remain connected to any terminal in $\mathcal{T}_{\Phi_h}$ in a solution for MinMTC, and let C_{Φ_h} be the set of edges with exactly one endpoint in V_{Φ_h}. Let D_{Φ_h} be the dual set of edges of C_{Φ_h} (it is important to note that the dual-joints are still defined with respect to the D_i, not the D_{Φ_h}). One can prove:

Lemma 4. *Let C be a minimum multiterminal cut in a graph $G = (V, E)$ and let D_{Φ_h}, $h \in \{1, \ldots, \phi\}$, contain p dual-joints $w_1, \ldots, w_p$, i.e., there exists $2p$ vertices $\{v^*_{h1}, \ldots, v^*_{hp}, u^*_{h1}, \ldots, u^*_{hp}\}$ such that C_{Φ_h} is a cut separating $\mathcal{T}_{\Phi_h} \bigcup \{v^*_{h1}, \ldots, v^*_{hp}\}$ from $(\mathcal{T} - \mathcal{T}_{\Phi_h}) \bigcup \{u^*_{h1}, \ldots, u^*_{hp}\}$ and $v^*_{hj} \in V$ is both in V_{Φ_h} and on the joint-cycle of w_j, while u^*_{hj} is the clockwise first vertex after v^*_{hj} on the joint-cycle of w_j, $j = 1, \ldots, p$. If $C'_{\Phi_h} \neq C_{\Phi_h}$ is a cut separating $\mathcal{T}_{\Phi_h} \bigcup \{v^*_{h1}, \ldots, v^*_{hp}\}$ from $(\mathcal{T} - \mathcal{T}_{\Phi_h}) \bigcup \{u^*_{h1}, \ldots, u^*_{hp}\}$, then $C' = (C - C_{\Phi_h}) \bigcup C'_{\Phi_h}$ is also a multiterminal cut. Moreover, this implies that C_{Φ_h} is a min cut separating $\mathcal{T}_{\Phi_h} \bigcup \{v^*_{h1}, \ldots, v^*_{hp}\}$ from $(\mathcal{T} - \mathcal{T}_{\Phi_h}) \bigcup \{u^*_{h1}, \ldots, u^*_{hp}\}$.*

The proof of this lemma is rather similar to the proof of [42, Theorem 2], but note that, here, we use a "stronger" version: the p dual-joints $w_1, \ldots, w_p$ are *on* the cycle D'_{Φ_h} (the dual set of edges of C'_{Φ_h}) and not *inside* it. On the one hand, this implies that each v^*_{hj} is uniquely defined (although in Lemma 3 one may have had the choice); on the other hand, we need this definition of v^*_{hj} in order to make sure that C' is indeed a multiterminal cut (this was not the case in [42, Theorem 2]). A consequence of Lemma 4 is that, instead of applying the algorithm of Yeh [42] on each one of the $|\mathcal{T}|$ terminals, one can identify the C_{Φ_h}'s by applying this algorithm on ϕ new terminals: for each face Φ_h, we add a new vertex "inside" Φ_h and link it, by an edge of capacity $+\infty$ (i.e., sufficiently large), to every terminal in $\mathcal{T}_{\Phi_h}$ (this does not destroy planarity). Note that *the number of dual-joints still depends on* $|\mathcal{T}|$ (and not on ϕ), and that an optimal solution for the initial instance does *not* necessarily induce an optimal solution for the MinMTC instance defined on the ϕ new terminals. Once a combination of C_{Φ_h}'s has been obtained, we compute the best possible multiterminal cut associated with this combination by solving ϕ independent instances of MinMTC where all the terminals lie on the same face (for each h, the instance on Φ_h can be solved in $O(|V_{\Phi_h}| \cdot |\mathcal{T}_{\Phi_h}|^2 \cdot (|\mathcal{T}_{\Phi_h}| + \log |V_{\Phi_h}|)) = O(|V_{\Phi_h}| \cdot |\mathcal{T}|^2 \cdot (|\mathcal{T}| + \log |V|))$ [11], with $\sum_{h=1}^{\phi} |V_{\Phi_h}| = |V| = n$). It should be clear from our analysis that the limited enumeration of the possible combinations leads to an optimal solution. Note that the running time of our first phase is the running time of Yeh's algorithm reduced by a factor of $O((|\mathcal{T}| - \frac{3}{2})^{|\mathcal{T}|-\phi})$. □

Note that, although the problem: *given a planar graph with a set of terminals, find a planar embedding of the graph such that the terminals are covered by as few faces as possible* is $\mathcal{NP}$-hard [4], it can be approximated within a ratio of 2 in linear time, provided that an embedding of the input planar graph is already given [20]. An interesting open problem is to determine the complexity of planar MinMTC when the terminals lie on a bounded number of faces ϕ (the *number of terminals* being *unbounded*). Note that the analysis of Theorem 6 is then irrelevant, because, even in this case, *the number of dual-joints cannot be bounded by a function of* ϕ.

Another question to consider is whether there exists a polynomial-time approximation scheme (PTAS) for planar MinMTC: the problem is known to be $\mathcal{NP}$-hard [12], but not APX-hard. However, although MinMTC is polynomial-time solvable in k-outerplanar graphs (since, according to [12], it is polynomial-time solvable in bounded tree-width graphs), the general framework for designing PTASs developed in [2] and based on a decomposition of the planar graph into a set of k-outerplanar graphs (one gets optimal solutions for the k-outerplanar graphs, and then combines them into a single solution) cannot be used for MinMTC, since gluing the pieces together does not always yield a multiterminal cut for the whole graph.

4 Integrality gap in k-edge-outerplanar graphs

In this section, we study the case of k-edge-outerplanar graphs. We first show that k'-outerplanar graphs having a degree bounded by d inside each block are closely related to these graphs.

4.1 Relationship between k-outerplanar graphs and k-edge-outerplanar graphs

The main result of this section is given in Theorem 7:

Theorem 7. *Any k-outerplanar graph such that the degree of each vertex is bounded by $d \geq 2$ inside each block is $(\lceil \frac{d}{2} \rceil + (k-1)\lfloor \frac{d}{2} \rfloor)$-edge-outerplanar. Moreover, any k-edge-outerplanar graph is k-outerplanar.*

Proof. The second part of Theorem 7 is obvious. We prove the first part by induction. Let G be a graph in $OPBID_{k,d}$, $k \geq 2$. For the proof, we can consider each block of G independently. Let $2VCC$ be an inclusionwise maximal 2-vertex-connected component of G. Each vertex of $2VCC$ lying on the outer face is adjacent to exactly two edges of $2VCC$ lying on the outer face. For each such vertex, we remove the corresponding two edges. We also remove any edge that does not lie in a block. We repeat this until each vertex of $2VCC$ lying on the outer face of G has at most one neighbor among the vertices lying in $2VCC$. At each iteration, for each vertex v lying on the outer face and still having at least two neighbors among the vertices in $2VCC$, we remove two edges adjacent to v, so we have to do it at most $\frac{d}{2}$ times if d is even. If d is odd, then we stop when the residual $deg_{2VCC}(v)$ is at most one, so we have to do it at most $\frac{d-1}{2}$ times, i.e., at most $\lfloor \frac{d}{2} \rfloor$ times. After that, we obtain a component in $OPBID_{k-1,d}$. Eventually, for a graph in $OPBID_{1,d}$, we use the same technique. If d is even, then the analysis is similar. If d is odd, then we have to make the residual $deg_{2VCC}(v)$ of each vertex v equal to 0, so we have to remove edges on the outer face $\lceil \frac{d}{2} \rceil$ times. Finally, any graph in $OPBID_{k,d}$ is $((k-1)\lfloor \frac{d}{2} \rfloor + \lceil \frac{d}{2} \rceil)$-edge-outerplanar. □

This theorem shows that, in order to be k-edge-outerplanar for some k, it is sufficient for a graph to be in $OPBID_{k',d}$ for some k' and d. However, it is not a necessary condition (every *Halin* graph, i.e., every planar graph with no vertex of degree 2 and whose edges are the disjoint union of a tree and a cycle connecting the leaves of this tree, is 2-outerplanar *and* 2-edge-outerplanar), and being only k'-outerplanar is not sufficient in general (for any $p > 2$, the complete bipartite graph $K_{2,p}$ is $\lceil \frac{p}{2} \rceil$-edge-outerplanar and 2-outerplanar, the first layer having 4 vertices and the second one $p-2$).

Moreover, Figure 2 shows that the bound of Theorem 7 is tight. In Section 4.2, we consider only k-edge-outerplanar graphs. Theorem 7 shows that our results will apply, in particular, to the graphs in $OPBID_{k',d}$.

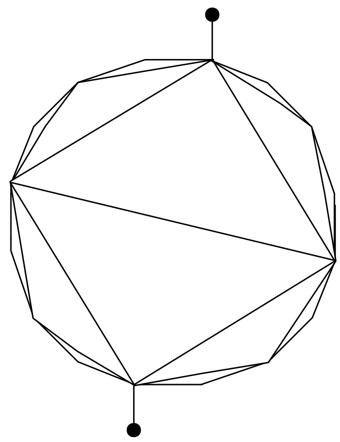


Figure 2: G_1, the skeleton of a family of tight graphs for Theorem 7 (d odd). Each graph G_i, $i \geq 2$, is actually obtained from G_1 by replacing each edge by a copy of G_{i-1}, the big two vertices corresponding to the endpoints of this edge. Given $k > 0$, the graph G_k is both k-outerplanar and $((k-1)\lfloor \frac{d}{2} \rfloor + \lceil \frac{d}{2} \rceil)$-edge-outerplanar ($d = 7$ here)

4.2 Bounding the gap for MaxEDP

Recall that MaxEDP is $\mathcal{NP}$-hard and APX-hard in edge-outerplanar graphs [23]. The main result of this section is that we can bound the integrality gap for MaxEDP in k-edge-outerplanar graphs:

Theorem 8. *The integrality gap for* MaxEDP *is bounded by* $4k$ *in* k*-edge-outerplanar graphs. Moreover, a solution for* MaxEDP *achieving this ratio can be computed in polynomial time.*

Proof. We use the algorithm given in Section 2.2. Let us describe Steps 1, 2 and 3. Given a k-edge-outerplanar connected graph $G = (V, E)$, Step 1 proceeds as follows: (i) for each one of the k layers of edges L of G, (ii) for each internal face Φ, if there exist edges lying both on L and on the border of Φ, remove exactly one such edge. After part (ii) ends for the i^{th} layer ($i \leq k-1$), we obtain a $(k-i)$-edge-outerplanar connected graph. Hence, at the end of Step 1 (i.e., when part (i) ends), we obtain a spanning tree T of G. Then, F_T and C_T are obtained in Step 2. Eventually, we use C_T to construct C_G in Step 3.

For each edge in C_T, C_G will contain at most $2k$ edges, and hence $\|C_G\| \leq 2k\|C_T\| \leq 4k\|F_T\|$. The removal of any edge $(u, v) \in C_T$ separates the vertices of T (and hence the vertices of G) in exactly two connected components, V_u and $V_v = V \setminus V_u$. Let $\delta_G(u, v)$ be the set of edges between V_u and V_v in G, and let the *circuit boundary* of a block of G be the cycle delimiting this block (the circuit boundary of G is defined as the disjoint union of the circuit boundaries of all its blocks). We need the following lemma, showing that $|\delta_G(u, v)| \leq 2k$:

Lemma 5. *Given V_u and V_v in a k-edge-outerplanar graph G, $\delta_G(u, v)$ contains at most 2 edges on each one of the k layers. Moreover, it contains exactly 2 edges on the k^{th} layer iff they are on the circuit boundary of G.*

Proof. We proceed by induction on k. For $k = 1$, we have an edge-outerplanar graph. If (u, v) does not lie in a ring, then $\delta_G(u, v)$ contains only (u, v) and we are done. Otherwise, by the way we construct T, there is a ring of G containing both (u, v) and an edge not in T: $\delta_G(u, v)$ contains these 2 edges. This completes the case $k = 1$.

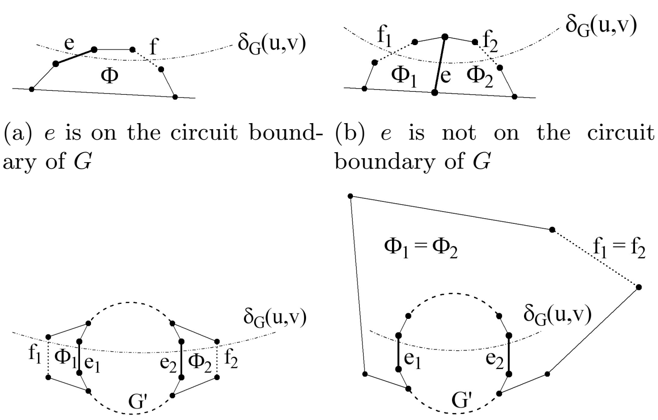
Assume now that Lemma 5 holds for $k - 1$, $k \geq 2$, and consider a k-edge-outerplanar connected graph. For each internal face having edges in common with the outer face, one such edge has been removed during Step 1; let us remove it again from G. The resulting connected graph G' is $(k-1)$-edge-outerplanar, hence we can apply the induction hypothesis. Moreover, $\delta_{G'}(u, v) \subseteq \delta_G(u, v)$. We have to distinguish between three cases:

- If there is no edge on the $(k-1)^{st}$ layer that belongs to $\delta_{G'}(u, v)$, then, obviously, for each edge e on the circuit boundary of G, there is a path linking its two endpoints and using only edges on the $(k-1)^{st}$ layer of G'. Hence, e does not belong to $\delta_G(u, v)$.

- If there is one edge (say, e) on the $(k-1)^{st}$ layer that belongs to $\delta_{G'}(u, v)$, then, by assumption, e is not on the circuit boundary of G'. Let us first assume that e lies in a block of G. If e is on the circuit boundary of G, then there is an internal face Φ of G (adjacent to the outer face of G) whose border contains both e and an edge f not in G', both lying in $\delta_G(u, v)$ (see Figure 3(a)). Otherwise, e belongs to the border of two internal faces of G, Φ_1 and Φ_2. Φ_1 (resp. Φ_2) is adjacent to the outer face and its border contains one edge f_1 (resp. f_2) not in G', that belongs to $\delta_G(u, v)$ (see Figure 3(b)). Note that if e does not lie in a block of G, then no edge from the circuit boundary of G belongs to $\delta_G(u, v)$ (i.e., $\delta_{G'}(u, v) = \delta_G(u, v)$).

- If there are two edges e_1 and e_2 on the $(k-1)^{st}$ layer belonging to $\delta_{G'}(u, v)$, then, by assumption, they belong to the circuit boundary of G'. Hence, e_1 (resp. e_2) belongs to the border of an internal face Φ_1 (resp. Φ_2) of G, adjacent to the outer face of G and containing one edge f_1 (resp. f_2) not in G'. f_1 and f_2 are distinct (and in $\delta_G(u, v)$) iff Φ_1 and Φ_2 are distinct (see Figures 3(c) and 3(d)).

The proof of Lemma 5 is now complete. □



(a) e is on the circuit boundary of G

(b) e is not on the circuit boundary of G

(c) Φ_1 and Φ_2 are distinct

(d) Φ_1 and Φ_2 are not distinct

Figure 3: Illustrating the 4 main cases of Lemma 5

We apply Lemma 5 for each edge in C_T, and this immediately implies $\|C_G\| \leq 2k\|C_T\| \leq 4k\|F_T\|$, as claimed. Note that the reason for processing Step 1 carefully is that, if T is not constructed as indicated, we will not be able to bound $|\delta_G(u,v)|$. For example, in Figure 1, assume that all the edges are valued by one and that the spanning tree constructed in Step 1 is the one in bold lines. Then, $\|C_T\| = 1$ although $\delta_G(u,v)$ contains all the edges in thin lines, so $\|C_G\|$ is unbounded. □

Note that, in graphs where all the edges have the same capacity, Theorem 8 applies to MaxIMF and MaxUSF as well (since at most one flow path is associated with each net and only edge-disjoint flow paths are used). More generally, we have the following corollary:

Corollary 2. *The integrality gap for* MaxIMF *and* MaxUSF *is bounded by* $4\beta k$ *in* k*-edge-outerplanar graphs* $G = (V, E)$ *satisfying* $\max_{e\in E} c(e) \leq \beta \min_{e\in E} c(e)$*. Moreover, solutions achieving this ratio can be computed in polynomial time.*

4.3 MaxIMF and MinMC in edge-outerplanar graphs

In this section, we consider the class of graphs where the degree of each vertex is bounded by two (i.e., is equal to 0 or 2) inside each block. Note that this is exactly the class of graphs where two arbitrary 2-vertex-connected components share at most one vertex, i.e., the class of graphs where each block is restricted to be a ring: hence, this is the class of edge-outerplanar graphs (or cacti). Obviously, such graphs generalize the trees of rings: a tree of rings is a graph obtained from a tree by replacing each vertex by a ring, two rings sharing a vertex if and only if the corresponding vertices of the tree are adjacent. Another definition is that a tree of rings is a 2-edge-connected edge-outerplanar graph.

The polynomial reduction given in [23] shows that MaxEDP (and thus MaxIMF) is $\mathcal{NP}$-hard and APX-hard in edge-outerplanar graphs. Moreover, Erlebach shows that this also holds in trees of rings and gives a 3-approximation algorithm for MaxEDP in these graphs [15]. We now show how to obtain 4-approximation algorithms for both MaxIMF and MinMC in edge-outerplanar graphs. The idea is to use the algorithm given in Section 2.2. Given an edge-outerplanar connected graph G, we denote by R_i, $i \in \{1, \ldots, \rho\}$, its i^{th} ring. Then, for each i, we remove the edge e_i in R_i having the smallest capacity among all the edges in R_i. This way, we obtain a maximum weight spanning tree T of G, and we can compute an integer multiflow F_T and a multicut C_T for T such that $\|C_T\| \leq 2\|F_T\|$ by using the algorithm given in [23]. Eventually, we construct a multicut C_G for G: for each ring R_i, we select e_i in C_G if and only if there is another edge of R_i in C_T. Moreover, we add in C_G all the edges of C_T. We have $\|C_G\| = \|C_T\| + \sum_{e_i \text{ / there is an edge of } R_i \text{ in } C_T} c(e_i) \leq 2\|C_T\| \leq 4\|F_T\|$. It is easily seen that C_G is indeed a multicut for G, since, for each edge $e_i = (a_i, b_i)$ not selected in C_G, there exists a path from a_i to b_i in T (i.e., a path in R_i that does not cross e_i). This implies:

Theorem 9. *In edge-outerplanar graphs, the integrality gap for* MaxIMF *(resp.* MinMC*) is at most* 4*. Moreover, a solution for* MaxIMF *(resp.* MinMC*) achieving this ratio can be computed in polynomial time.*

Figure 4 shows that our analysis of this algorithm is tight, since there exist families of instances where $\|C_G\|$ is equal to $4\|F_T\|$. However, this does not necessarily imply that the gaps for MaxIMF and MinMC are tight. Note that Theorem 9 also holds for MaxUSF. Moreover, this theorem shows that the integrality gap for MaxIMF shrinks to a factor of 4 when the maximum inside degree is at most 2, while it can be as large as $\sqrt{n}$ when the maximum degree is 3 [23, p. 17].

5 A conjecture about another class of graphs

We make the following conjecture about the graphs studied in [18]:

Conjecture 1. *The integrality gap for* MaxIMF *is* $O(1)$ *in planar graphs where all the terminals lie on the outer face and the sum of the capacities of the edges adjacent to each vertex not on the outer face is even.*

This conjecture has been suggested, in particular, by the recent results presented in [9, 10] and by the noticeable importance of both the evenness condition and the localization of the terminals on the outer face [18] (recall

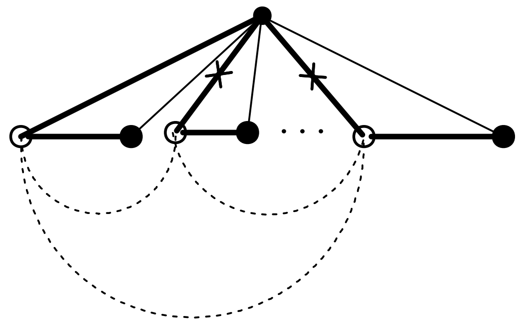


Figure 4: A family of $(2p+1)$-vertex graphs (p odd), tight for our algorithm. There exists a net (in dashed lines) between any two of the p white vertices, and the edges of the spanning tree T are in bold lines. In these graphs, $\|F_T\| = \frac{p-1}{2}$, $\|C_T\| = p-1$ (edges denoted by $\not\!-$) and $\|C_G\| = 2(p-1)$

that Frank proves that EDP is polynomial-time solvable in this case, and gives necessary and sufficient conditions for the existence of a solution).

Proving Conjecture 1 would imply, in particular, that the integrality gap for MaxIMF is $O(1)$ ($O(\log n)$ being already known [9, 10]) in planar graphs where all the terminals lie on the outer face, if all the capacities are at most 2 (otherwise, recall that the integrality gap can be $\Theta(\sqrt{n})$ [23]). Indeed, given such a graph G, one can decrease every odd capacity by one and obtain a graph G' having only even capacities, and thus satisfying the assumptions of Conjecture 1. It is easy to see that the integrality gap in G is at most $\frac{3}{2}$ times the integrality gap in G' (and hence is also $O(1)$ if Conjecture 1 is true), since the value of the minimum fractional multicut (and hence of the maximum fractional multiflow) in G is at most $\frac{3}{2}$ times the value in G' (the worst case occurring when all the capacities are decreased from 3 to 2).

6 Conclusion

In this paper, we have generalized all the results obtained for the trees by Garg, Vazirani and Yannakakis to graphs with a fixed cyclomatic number. In particular, this implies that, in these graphs, MaxEDP is polynomial-time solvable and MaxIMF has an integrality gap bounded by two times one plus the cyclomatic number. It is worth mentioning that our algorithmic approaches are simple and directly rely on algorithms for trees, so any improvement for these algorithms (improved running times, parallelization, online versions, etc.) can immediately be used for ours. Moreover, we have shown that other classical generalizations do not lead to results such as ours, and, for a fixed number of nets, we have solved in polynomial time MinMC in planar and in bounded tree-width graphs. We have also introduced a new class of planar graphs, the k-edge-outerplanar graphs. We have proved that the integrality gap for MaxEDP is bounded in these graphs and have shown how they are related to k-outerplanar graphs. Furthermore, we have shown that the integrality gap for MaxIMF is bounded in edge-outerplanar graphs (or cacti), a class of graphs that generalizes the trees of rings.

However, there are still interesting open problems for which no significant progress has been made: can we improve the $O(\sqrt{n})$ approximation ratio for MaxEDP in planar graphs, or can an inapproximability result stronger than APX-hardness be proved for this problem? And what about the general graphs? Turning back to our results, one may also explore further the fixed-parameter tractability of MaxEDP [13]. Furthermore, is the integrality gap for MaxEDP or MaxIMF bounded by a constant in k-outerplanar graphs, or even in bounded tree-width graphs? How can Conjecture 1 be proved?

References

[1] M. Andrews and L. Zhang. Hardness of the undirected edge-disjoint paths problem. Proceedings STOC 2005.

[2] B.S. Baker. Approximation Algorithms for NP-complete Problems on Planar Graphs. J. ACM 41 (1994) 153–180.

[3] A. Baveja and A. Srinivasan. Approximation algorithms for disjoint paths and related routing and packing problems. Math. Oper. Res. 25 (2000).

[4] D. Bienstock and C.L. Monma. On the complexity of covering vertices by faces in a planar graph. SIAM Journal On Computing 17 (1988), 53–76.

[5] H.L. Bodlaender. Planar graphs with bounded treewidth. Technical Report RUU-CS-88-14, Utrecht University, The Netherlands (1988).

[6] G. Călinescu, C.G. Fernandes and B. Reed. Multicuts in unweighted graphs and digraphs with bounded degree and bounded tree-width. Journal of Algorithms 48 (2003) 333–359.

[7] P. Carmi, T. Erlebach and Y. Okamoto. Greedy edge-disjoint paths in complete graphs. Proceedings 29th International Workshop on Graph Theoretic Concepts in Computer Science. LNCS 2880 (2003) 143–155.

[8] C.Chekuri and S.Khanna. Edge Disjoint Paths Revisited. Proceedings SODA 03 (2003).

[9] C. Chekuri, S. Khanna and B. Shepherd. Edge-disjoint paths in planar graphs. Proceedings 45th IEEE FOCS (2004).

[10] C. Chekuri, S. Khanna and B. Shepherd. Multicommodity Flow, Well-linked Terminals, and Routing Problems. Proceedings STOC 05 (2005).

[11] D.Z. Chen and X. Wu. Efficient algorithms for k-terminal cuts on planar graphs. Algorithmica 38 (2004) 299–316.

[12] E. Dahlhaus, D.S. Johnson, C.H. Papadimitriou, P.D. Seymour and M. Yannakakis. The complexity of multiterminal cuts. SIAM Journal on Computing 23 (1994) 864–894.

[13] R.G. Downey and M.R. Fellows. Parameterized Complexity (1999). Springer-Verlag. New York.

[14] P.L. Erdös and L.A. Székely. Algorithms and Min-max Theorems for Certain Multiway Cuts. Proceedings IPCO 92 (1992) 334–345.

[15] T. Erlebach. Approximation algorithms and complexity results for path problems in trees of rings. Proceedings 26th International Symposium on Mathematical Foundations of Computer Science. LNCS 2136 (2001) 351–362.

[16] S. Even, A. Itai and A. Shamir. On the complexity of timetable and multicommodity flow problems. SIAM J. Comput. 5 (1976) 691–703.

[17] L.R. Ford and D.R. Fulkerson. Maximal Flow Through a Network. Canadian Journal of Mathematics 8 (1956) 339–404.

[18] A. Frank. Edge-disjoint paths in planar graphs. Journal of Combinatorial Theory, Series B 39 (1985) 164–178.

[19] A. Frank. Packing Paths, Circuits and Cuts - A Survey. In B. Korte, L. Lovász, H. J. Prömel and A. Schrijver. *Paths, Flows and VLSI-Layout.* Algorithms and combinatorics 9 (1990) 47–100. Springer-Verlag. Berlin.

[20] G. Frederickson. Planar graph decomposition and all pairs shortest paths. Journal of the ACM 38 (1991), 162–204.

[21] A. Frieze. Edge-disjoint paths in expander graphs. SIAM Journal On Computing 30 (2000) 1790–1801.

[22] N. Garg, V.V. Vazirani and M. Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. SIAM Journal on Computing 25 (1996) 235–251.

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

[24] A.M.H. Gerards and B. Shepherd. Preselecting homotopies for the weighted disjoint paths problem. Manuscript (1993).

[25] D. Hartvigsen. The planar multiterminal cut problem. Discrete Applied Mathematics 85 (1998) 203–222.

[26] J. Kleinberg and É. Tardos. Disjoint paths in densely embedded graphs. Proceedings 36th IEEE FOCS (1995) 52–61.

[27] J. Kleinberg and É. Tardos. Approximations for the disjoint paths problem in high-diameter planar networks. Journal of Computer and System Sciences 57 (1998) 61–73.

[28] S. Kolliopoulos and C. Stein. Approximating disjoint-paths using greedy algorithms and packing integer programs. Proceedings IPCO 98 (1998).

[29] P. Kolman and C. Scheideler. Improved bounds for the unsplittable flow problem. Proceedings SODA 02 (2002), 184–193.

[30] P. Kolman. A note on the greedy algorithm for the unsplittable flow problem. Information Processing Letters 88 (2003) 101–105.

[31] E. Korach and M. Penn. A fast algorithm for maximum integral two-commodity flow in planar graphs. Discrete Applied Mathematics 47 (1993) 77 – 83.

[32] J.B. Kruskal. On the shortest spanning subtree of a graph and traveling salesman problem. Proc. Amer. Math. Soc. 7 (1956) 48–50.

[33] D. Marx. Eulerian disjoint paths problem in grid graphs is NP-complete. Discrete Applied Mathematics 143 (2004) 336–341.

[34] M. Middendorf and F. Pfeiffer. On the complexity of the disjoint paths problem. Combinatorica 13 (1993) 97–107.

[35] T. Nishizeki, J. Vygen and X. Zhou. The edge-disjoint paths problem is NP-complete for series-parallel graphs. Discrete Applied Mathematics 115 (2001) 177–186.

[36] K. Obata. Approximate max-integral-flow/min-multicut theorems. Proceedings STOC 04 (2004).

[37] H. Okamura and P.D. Seymour. Multicommodity flows in planar graphs. Journal of Combinatorial Theory, Series B 31 (1981) 75–81.

[38] P. Raghavan. Probabilistic construction of deterministic algorithms: approximating packing integer programs. Journal of Computer and System Sciences 37 (1988) 130–143.

[39] N. Robertson and P.D. Seymour. Graph minors II: Algorithmic aspects of tree-width. Journal of Algorithms 7 (1986) 309–322.

[40] N. Robertson and P.D. Seymour. Graphs minors XIII: The disjoint paths problem. J. Combin. Theory, Ser. B, 63 (1995) 65–110.

[41] É. Tardos and V.V. Vazirani. Improved bounds for the max-flow min-multicut ratio for planar and $K_{r,r}$-free graphs. Inform. Process. Lett. 47 (1993) 77–80.

[42] W.-C. Yeh. A simple algorithm for the planar multiway cut problem. Journal of Algorithms 39 (2001) 68–77.