J Comb Optim (2009) 18: 272–293
DOI 10.1007/s10878-009-9225-x

Flows with unit path capacities and related packing and covering problems

Maren Martens · Martin Skutella

Published online: 21 May 2009

Abstract Since the seminal work of Ford and Fulkerson in the 1950s, network flow theory is one of the most important and most active areas of research in combinatorial optimization. Coming from the classical maximum flow problem, we introduce and study an apparently basic but new flow problem that features a couple of interesting peculiarities. We derive several results on the complexity and approximability of the new problem. On the way we also discover two closely related basic covering and packing problems that are of independent interest.

Starting from an LP formulation of the maximum s-t-flow problem in path variables, we introduce unit upper bounds on the amount of flow being sent along each path. The resulting (fractional) flow problem is NP-hard; its integral version is strongly NP-hard already on very simple classes of graphs. For the fractional problem we present an FPTAS that is based on solving the k shortest paths problem iteratively. We show that the integral problem is hard to approximate and give an interesting $O(\log m)$-approximation algorithm, where m is the number of arcs in the considered graph. For the multicommodity version of the problem there is an $O(\sqrt{m})$-approximation algorithm. We argue that this performance guarantee is best possible, unless P = NP.

This work was partially supported by the DFG Research Center Matheon in Berlin, by the Graduate School of Production Engineering and Logistics, North Rhine-Westphalia, by the DFG Focus Program 1126 and the DFG grants SK 58/4-1 and SK 58/5-3, and by an NSERC Operating Grant.

Part of this work was done while M. Martens was at Universität Dortmund and at the Sauder School of Business, University of British Columbia.

Part of this work was done while M. Skutella was at Universität Dortmund.

M. Martens (✉)
Zuse Institute Berlin, Takustr. 7, 14195 Berlin, Germany
e-mail: martens@zib.de

M. Skutella
TU Berlin, Institut für Mathematik, MA 5–2, Str. des 17. Juni 136, 10623 Berlin, Germany
e-mail: skutella@math.tu-berlin.de

Keywords Network flows · Approximation algorithms · Packing problems · Covering problems

1 Introduction

Problem definition and notation The classical maximum s-t-flow problem has been studied from many different points of view. Numerous algorithms are known to solve the problem in polynomial time (see, e.g., Ahuja et al. 1993; Schrijver 2003). Ford and Fulkerson (1956) proved already in the 1950s that there always exists an integral optimal solution to the maximum s-t-flow problem provided that all arc capacities are integral. This follows for example from the fact that the constraint matrix of the natural LP formulation in arc variables is totally unimodular. It is also well known that any s-t-flow can be decomposed into flow along paths and cycles. Omitting flow along cycles (which does not contribute to the flow value) yields an alternative LP formulation of the problem in path variables.

In this paper we study a new network flow problem in which the flow on any path is bounded by 1. In other words, we add box constraints to the LP formulation of the maximum flow problem in path variables. We call the resulting problem the *maximum one-flow problem*. Our motivation for studying this problem is mainly academic but, from a more practical point of view, the problem is well motivated when we think of applications in transportation or communication networks where every single path might be unreliable. In such situations it is reasonable to diversify a commodity or information among several different paths. This can be accomplished by forbidding to send more than a fixed amount of flow along a single path.

A more formal definition of the problem is as follows: We are given a network (digraph) $D = (V, A)$ with arc capacities $u : A \to \mathbb{R}^+$ and two distinguished nodes $s,\ t \in V$. We assume that $u(a) \geq 1$, for all $a \in A$. If not stated otherwise, $m := |A|$ denotes the number of arcs in the network. Let $\mathcal{P}$ be the set of simple directed s-t-paths in D. Then the *maximum one-flow problem* (*max-1FP*) can be formulated as follows, where the path variable x_P denotes the amount of flow sent along path $P \in \mathcal{P}$:

$$\max \quad \sum_{P \in \mathcal{P}} x_P$$

$$\text{s.t.} \quad \sum_{P \ni a} x_P \leq u(a) \quad \forall a \in A \tag{1}$$

$$0 \leq x_P \leq 1 \quad \forall P \in \mathcal{P} \tag{2}$$

Notice that omitting the constraints $x_P \leq 1$ yields the classical maximum s-t-flow problem. An s-t-flow fulfilling (1) and (2) is called a *one-flow*. In an *integral one-flow* each s-t-path sends either 0 or 1 unit of flow. To emphasize that a certain one-flow is not necessarily integral, we sometimes call it *fractional*. Note that in general the encoding size of a maximum one-flow is not polynomial in the input size of the problem since one might want to send flow along exponentially many s-t-paths. Therefore, the best one can expect in terms of complexity are algorithms with running time polynomially bounded in the input plus output size.

We also consider the dual of the max-1FP which is given as follows:

$$\begin{aligned} \min \quad & \sum_{a\in A} u(a)y_a + \sum_{P\in\mathcal{P}} z_P \\ \text{s.t.} \quad & z_P + \sum_{a\in P} y_a \geq 1 \quad \forall P \in \mathcal{P} \\ & z_P, y_a \geq 0 \quad \forall a \in A,\ P \in \mathcal{P} \end{aligned}$$

The integer version of the dual can be interpreted as a special minimum cut problem, where each s-t-path must be destroyed and this can be done by deleting either a single arc on the path or the path itself. The deletion of an arc a is in general more expensive than that of a whole path ($u(a)$ instead of 1), but can also destroy more than one path simultaneously. The dual separation problem of the classical maximum s-t-flow problem is a shortest path problem. It is not difficult to observe that the dual separation problem of the max-1FP can be solved by computing the k shortest s-t-paths with respect to the dual arc lengths y_a, where k is the number of paths $P \in \mathcal{P}$ with $z_P > 0$ plus 1. The k shortest paths problem also plays an important role in solving the Lagrange relaxation of the max-1FP that is obtained by penalizing the violation of capacity constraints (1) in the objective function. We discuss this issue in more detail later on.

Related results from the literature To the best of our knowledge, the one-flow problem (1FP) is studied for the first time in this paper. But there is some literature dealing with problems related to it. The problem to compute the number of different (simple) s-t-paths in a network is a special case of the max-1FP. (Consider the case when all arc capacities are infinite.) Valiant (1979) shows that this problem is #P-complete under polynomial-time reductions. Some work has been done on counting the number of paths in grids. E.g., Lucas (1983) shows the relationship between Pascal's triangle and the "space" grid or Gessel (1993) counts paths in the so called Young's lattice. More general results were, e.g., obtained by Bartholdi (1999) and Stanley (1996).

Similar to the problem of counting paths in a graph is the edge-disjoint paths problem (EDP) which is the same as the integral multicommodity 1FP when we fix all capacities in the considered network to 1. Kleinberg (1996) gives an extensive overview of the EDP. Further interesting results were, e.g., obtained by Guruswami et al. (1999). They prove that the maximum EDP is hard to approximate within a factor $O(m^{1/2-\epsilon})$, for any $\epsilon > 0$, and give an $O(\sqrt{m})$-approximation algorithm. More recent results for the EDP were, e.g., found by Andrews and Zhang (2007) or Chekuri and Khanna (2007). Considering the EDP as a special packing problem, there are more results by Baveja and Srinivasan (2000) and Kolliopoulos and Stein (2004). Recently, Kolliopoulos (2007) presented an extensive survey on edge-disjoint paths.

Also for more general packing problems many results have been obtained during the last years. Among the most interesting results for the present paper are the ones obtained by Plotkin et al. (1995) who introduce new approximation algorithms for the problem. Together with Grigoriadis and Khachiyan (1995a, 1995b, 1996a, 1996b), they adapt techniques for solving multicommodity flow problems (see, e.g., Shahrokhi and Matula 1990; Klein et al. 1994; Leighton et al. 1995; Garg and Könemann 1998; Fleischer 2000) to a general class of packing and covering problems.

Other results on mixed packing and covering problems were, e.g., obtained by Young (2001).

In Martens and Skutella (2006) the authors have already considered a type of network flow problems with path capacities. They study the k-splittable flow problem with path capacities. Here the flow of any commodity is restricted to at most k paths whose flow values may not exceed given bounds (capacities).

As mentioned above, the k shortest paths problem (for a single source and a single sink) is of great interest for the problem considered in this paper. To solve the Lagrange relaxation of the max-1FP that is obtained by penalizing the violation of arc capacities in the objective function, we make use of a result by Lawler (1972) who shows how to compute k shortest (simple) paths in $O(kn^3)$ time. This improves the runtime of an algorithm by Yen (1971) by a factor n. Eppstein (1998) considers the problem to compute k shortest s-t-paths allowing cycles and presents an algorithm running in $O(m + n \log n + k)$ time. Fox (1975) presents a method for the problem allowing cycles that is based on Dijkstra's (1959) algorithm and runs in $O(m + kn \log n)$ time. Using improvements in Dijkstra's (1959) algorithm the runtime in Lawler (1972) can be decreased to $O(kn(m + n \log n))$. Dreyfus (1969) and Eppstein (1998) also consider the problem to compute k shortest paths (allowing cycles) from a given source to each other vertex. The running times they obtain are $O(kn^2)$ and $O(m + n \log n + kn)$ respectively. Roditty (2007) presented an approximation algorithm that computes k simple paths from a source s to a sink t such that the length of the ith path is at most $3/2$ times the length of the ith shortest simple s-t-path. This algorithm runs in $O(k\sqrt{n}(m + n \log n))$ time.

Contribution of this paper As mentioned above, the problem of computing the number of different simple s-t-paths in a network is #P-complete and a special case of the max-1FP. It therefore follows immediately that computing the maximum one-flow value is NP-hard. This holds for the fractional as well as for the integral one-flow problem. We prove that the integral max-1FP is strongly NP-hard already on very simple acyclic networks consisting of a chain of parallel arcs. Even worse, the integral max-1FP is APX-hard, even in networks where the number of s-t-paths is polynomially bounded in the size of the network. One interesting consequence of these hardness results is the following: It is NP-hard to decide whether a given integral s-t-flow has an integral path decomposition such that each path carries at most one unit of flow.

In Sect. 2 we establish a close relation between two interesting new combinatorial problems and the special case of the max-1FP on networks consisting of a chain of parallel arcs. The first problem is to cover the edges of a complete graph by cuts of bounded size where the size of a cut is the cardinality of the smaller of the two vertex subsets. The second problem is a packing problem: Consider a set where each element has a given integral weight and find a pre-specified number of different subsets such that the number of subsets containing an element is bounded by the element's weight. The two problems are equivalent and, maybe surprisingly, strongly NP-hard. This also yields the strong NP-hardness of the integral max-1FP on chains of parallel arcs. To keep Sect. 2 well arranged, we present all NP-hardness results in Sect. 3. In Sect. 2,

however, we also show that already on a chain of parallel arcs of length 3 the max-1FP has an integrality gap. We also prove that it might happen that each arc in a network carries an integral amount of flow in a maximum one-flow but no maximum one-flow is integral.

In Sect. 4 we show that the approach of Plotkin et al. (1995) yields an FPTAS for the fractional max-1FP. The core of the algorithm consists of iteratively solving k shortest paths problems on the given network with varying arc lengths. In Sect. 5 we derive several approximation algorithms for the integral max-1FP. Our main result is a randomized approximation algorithm with performance ratio $O(\log m)$.

Finally, in Sect. 6 we study multicommodity versions of the one-flow problem. We show that the FPTAS from Sect. 4 can be generalized to the fractional multicommodity one-flow problem. For the integral maximum multicommodity one-flow problem we present a randomized $O(\sqrt{m})$-approximation algorithm and show that, unless P = NP, no better approximation is possible. Moreover, we present an $O(\log m)$-approximation algorithm for the problem to find an integral multicommodity one-flow with minimum congestion.

2 Interesting related problems

In this section we study the max-1FP on a restricted class of networks that are given by chains of parallel arcs. In order to obtain a better understanding of the max-1FP on this particular class of networks we consider two equivalent combinatorial optimization problems, one of which is a covering and the other a packing problem. Although these two problems are easy to formulate and seem quite natural, they have not appeared in the literature before to the best of our knowledge.

We consider networks that consist of $n+1$ vertices $v_0, v_1, \ldots, v_n$ and $2n$ arcs ($n \in \mathbb{N}$) such that there are two parallel arcs from v_{i-1} to v_i, for $i = 1, \ldots, n$. Vertex v_0 is the source and v_n is the sink. We call one arc of each pair of parallel arcs the *upper* and the other one the *lower arc*. All lower arcs have infinite capacity. The capacity of the ith upper arc is $c_i \in \mathbb{N}$, for $i = 1, \ldots, n$. We call such a capacitated network a *chain of parallel arcs*; see Fig. 1 for an example.

An integral one-flow is given by a set of s-t-paths that are pairwise distinct. Notice that two s-t-paths in the considered network are different if and only if there is a pair of arcs where one path uses the upper arc and the other path uses the lower arc. In particular, the ith pair of arcs can distinguish a subset of at most c_i paths from all other paths. This motivates the following problem.

Bounded Cut Cover Problem
GIVEN: k numbers $c_1, \ldots, c_k \in \mathbb{N}$ and a number $q \in \mathbb{N}$.
TASK: Find k subsets $M_1, \ldots, M_k \subseteq \{1, \ldots, q\}$ with $|M_i| \leq c_i$, for $i = 1, \ldots, k$, such that for any pair $j, \ell \in \{1, \ldots, q\}$ with $j \neq \ell$ there is some $i \in \{1, \ldots, k\}$ with $|M_i \cap \{j, \ell\}| = 1$; or decide that no such family of subsets exists.

The name that we choose for this problem stems from the following graph-theoretic interpretation: Consider a complete undirected graph with vertex set $\{1, \ldots, q\}$. The

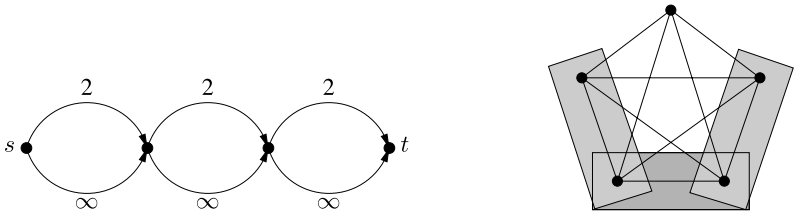

Fig. 1 On the *left*: A chain of parallel arcs. All upper arcs have capacity 2 while the lower arcs have infinite capacity. On the *right*: A solution to the bounded cut cover problem for $q = 5$, $k = 3$, and $c_1 = c_2 = c_3 = 2$. Every vertex of the bounded cut cover instance can be interpreted as an s-t-path in the chain of parallel arcs carrying one unit of flow. The three cuts on the right, each of size 2, guarantee that on the left three pairs of parallel arcs, with the upper arcs having capacity 2, suffice to route five units of flow feasibly: Since every two vertices on the right are separated by cut(s), one can choose flow carrying paths on the left such that every two paths differ at exactly those pairs of parallel arcs that correspond to the separating cuts

question is whether the edges of the complete graph can be covered by k cuts where, for $i = 1, \ldots, k$, the ith cut partitions the vertex set into two subsets the smaller of which has cardinality at most c_i. In Fig. 1 we give an example. The solution to the instance of the bounded cut cover problem depicted there represents a one-flow of value 5 in the corresponding chain of parallel arcs (also shown in Fig. 1).

Observation 2.1 *The bounded cut cover problem has a solution if and only if there exists an integral one-flow of value q in a chain of parallel arcs of length k where the capacities of the upper arcs are $c_1, \ldots, c_k$.*

Next we consider the following fractional relaxation of the bounded cut cover problem.

Fractional Bounded Cut Cover Problem
GIVEN: k numbers $c_1, \ldots, c_k \in \mathbb{N}$ and a number $q \in \mathbb{R}^+$.
TASK: For some $r \geq q$, find weights $x_1, \ldots, x_r \in [0, 1]$ with $\sum_{j=1}^{r} x_j = q$ and determine k subsets $M_1, \ldots, M_k \subseteq \{1, \ldots, r\}$ with $\sum_{j \in M_i} x_j \leq c_i$, for $i = 1, \ldots, k$, such that for any pair $j, \ell \in \{1, \ldots, r\}$ with $j \neq \ell$ there is some $i \in \{1, \ldots, k\}$ with $|M_i \cap \{j, \ell\}| = 1$; or decide that this is not possible.

There is again a graph-theoretic interpretation of the problem. The task is to find a complete graph with weights on the vertices such that the weight x_i of every vertex i is between 0 and 1 and the weights sum up to q. Moreover, the edges of the complete graph must be covered by k cuts such that the ith cut partitions the vertex set into two subsets the lighter of which has total weight at most c_i. Associating the weighted nodes of the complete graph with s-t-paths of corresponding flow value yields the following observation.

Observation 2.2 *The fractional bounded cut cover problem has a solution if and only if there exists a (fractional) one-flow of value q in a chain of parallel arcs of length k where the capacities of the upper arcs are $c_1, \ldots, c_k$.*

It is natural to ask whether the fractional bounded cut cover problem allows for larger values of q with feasible solutions than the non-fractional version. By Observations 2.1 and 2.2, this is equivalent to the question whether for chains of parallel arcs there always exists a maximum one-flow that is integral. In the fractional bounded cut cover problem the price of the additional degree of freedom given by the possibility to assign fractional weights to the nodes is an increase in the number of nodes (since the node weights still have to sum up to q). On the one hand, a larger number of nodes makes it more difficult to cover all edges of the complete graph. On the other hand, fractional weights on the vertices allow for more balanced cuts that contain more edges. We show below that there exist instances with a larger feasible value of q in the fractional version of the problem than in the integral version. Before we discuss this issue in more detail, we present another equivalent packing problem.

In a chain of parallel arcs, every s-t-path is uniquely determined by the subset of upper arcs contained in the path. Therefore, computing an integral one-flow of value q corresponds to finding a family of q pairwise distinct subsets of $\{1, \ldots, k\}$ such that $i \in \{1, \ldots, k\}$ is contained in at most c_i of these subsets.

Capacitated Set Packing Problem
GIVEN: k numbers $c_1, \ldots, c_k \in \mathbb{N}$ and a number $q \in \mathbb{N}$.
TASK: Find q pairwise distinct subsets of $\{1, \ldots, k\}$ such that element $i \in \{1, \ldots, k\}$ is contained in at most c_i of these subsets, for $i = 1, \ldots, k$; or decide that no such family of subsets exists.

We also consider the following fractional relaxation of the capacitated set packing problem.

Fractional Capacitated Set Packing Problem
GIVEN: k numbers $c_1, \ldots, c_k \in \mathbb{N}$ and a number $q \in \mathbb{R}^+$.
TASK: For some $r \geq q$, find pairwise distinct subsets $N_1, \ldots, N_r$ of $\{1, \ldots, k\}$ with weights $x_1, \ldots, x_r \in [0, 1]$ such that $\sum_{j=1}^{r} x_j = q$ and $\sum_{j:i \in N_j} x_j \leq c_i$, for $i = 1, \ldots, k$; or decide that this is not possible.

Observation 2.3 *The (fractional) capacitated set packing problem has a solution if and only if there exists an integral (fractional) one-flow of value q in a chain of parallel arcs of length k where the capacities of the upper arcs are $c_1, \ldots, c_k$. In particular, the (fractional) capacitated set packing problem is equivalent to the (fractional) bounded cut cover problem.*

The following instance shows that the fractional capacitated set packing problem in general allows for strictly larger values of q with feasible solutions than the non-fractional version. Due to Observation 2.3, the same holds for the bounded cut cover problem. Let $k = 3$ and $c_1 = c_2 = c_3 = 2$. It is not difficult to check that $q = 5$ is the largest value of q with a feasible solution to the non-fractional version of the problem: Choose for example the subsets $\emptyset$, $\{1\}$, $\{2\}$, $\{3\}$, and $\{1, 2, 3\}$. However, there is a solution to the fractional version of the problem with $q = 5.5$: Choose subsets $\emptyset$, $\{1\}$, $\{2\}$, and $\{3\}$ all with weight 1. In addition choose subsets $\{1, 2\}$, $\{1, 3\}$, and $\{2, 3\}$ all with weight $1/2$. The corresponding instance of the max-1FP is depicted in Fig. 1.

Observation 2.4 *The instance of the max-1FP depicted in Fig. 1 has an integrality gap of* $11/10$.

The instance depicted in Fig. 1 also shows that in general there is no integral optimal solution to the dual problem of the max-1FP. An optimal dual solution destroys the path using all lower arcs and half of each path that uses exactly one upper arc; further, one half of each upper arc is deleted. The following result underlines the discrepancy between the fractional and the integral one-flow problem even more.

Proposition 2.5 *The existence of a maximum one-flow where the flow value on each arc is integral does in general not imply the existence of an integral path decomposition where each path carries at most one unit of flow.*

Proof We show that a one-flow sending an integral amount of flow along each arc is not necessarily integral in the sense that each path routes either 0 or 1 unit of flow. Figure 2 shows an instance of the max-1FP in which the maximum flow yields an integral flow value on all arcs, although it is fractional. A similar network is used in Baier et al. (2006) to prove the discrepancy of arc-wise and path-wise integrality for the length-bounded flow problem.

The idea is to double the network in Fig. 1. Then we can send 5.5 units from s to v along different paths on either chain of parallel arcs (see also the discussion before Observation 2.4). These 11 units of flow are routed from v to t along the arc of capacity 11. Another unit of flow can be gained by sending half a unit of flow along each s-t-path using only arcs of capacity 4 from s to v plus the arc (v, t) of capacity 1.

It remains to show that there does not exist an integral path decomposition of this flow of value 12. In an integral path decomposition, all paths but one use the upper arc (v, t) of capacity 11. By symmetry we can assume that the path containing the lower arc (v, t) of capacity 1 uses the lower chain of parallel arcs from s to v. Thus there are 6 paths using the upper chain of parallel arcs from s to v and the upper arc (v, t) of capacity 11. This yields a contradiction to the discussion before Observation 2.4. □

We conclude this section with an open problem. The instance of the capacitated set packing problem discussed after Observation 2.3 has an additive integrality gap of $1/2$. Do there exist instances with additive integrality gap 1 or larger? This question is also interesting in view of the Bounded Cut Cover Problem.

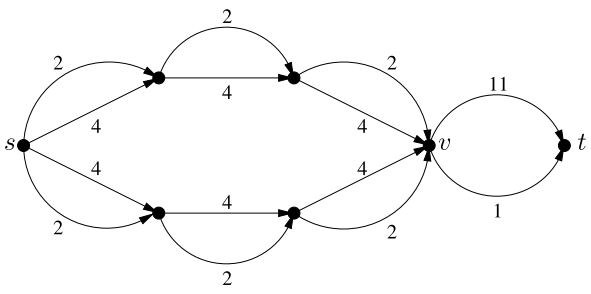

Fig. 2 A network with a maximum one-flow that is arcwise integral, but pathwise fractional

3 Hardness results

We first return to the capacitated set packing problem. In contrast to the classical set packing problem and many similar problems known from the literature, the capacitated set packing problem allows to choose arbitrary subsets, i.e., they do not have to belong to a given family of subsets. This might make the problem seem to be easier. However, we can prove the following somewhat surprising theorem.

Theorem 3.1 *The capacitated set packing problem is strongly NP-hard. If q is polynomially bounded in k, the problem is strongly NP-complete.*

Proof We use the notion r-subset to describe a subset of cardinality $r \in \mathbb{N}$ of a given ground set.

To prove the NP-hardness of the capacitated set packing problem, we use a reduction of the strongly NP-complete 3-PARTITION problem.

3-PARTITION
GIVEN: An integer $B \in \mathbb{N}$ and $k = 3\ell$ weights $s_1, \ldots, s_k \in \mathbb{N} \cap (B/4, B/2)$, for some $\ell \in \mathbb{N}$, such that $\sum_{i=1}^{k} s_i = \ell B$.
TASK: Find a partition $A_1, \ldots, A_\ell$ of $\{1, \ldots, k\}$ such that $\sum_{i \in A_j} s_i = B$ for all $j = 1, \ldots, \ell$; or decide that no such partition exists. (Note that $|A_j| = 3$, for all $j = 1, \ldots, \ell$, since $s_i \in (B/4, B/2)$ for $i = 1, \ldots, k$.)

Consider any instance of 3-PARTITION. We construct an instance of the capacitated set packing problem as follows. Let $B_3^<$ contain exactly the 3-subsets of $\{1, \ldots, k\}$ for which the elements' weights sum up to less than B, i.e., $B_3^< := \{\mathcal{M} \subseteq \{1, \ldots, k\} \mid |\mathcal{M}| = 3, \sum_{i \in \mathcal{M}} s_i < B\}$. For $i = 1, \ldots, k$, let $B_3^<(i)$ be the sets in $B_3^<$ that contain i. Then the capacity of i is given as $c_i := |B_3^<(i)| + k + 1$. The integer q is set to $q := |B_3^<| + \binom{k}{2} + k + \ell + 1$.

Note that all numbers here are computable in time polynomial in the size of the 3-PARTITION instance, since there are only $\binom{k}{3} = O(k^3)$ different 3-subsets of $\{1, \ldots, k\}$. Further, all numbers c_i and q are polynomial in k.

Now we show that a feasible partition for the instance of 3-PARTITION exists if and only if there is a feasible family of subsets for the instance of the capacitated set packing problem.

Let us start with the forward implication and assume that for the considered instance of 3-PARTITION there is a feasible partition. Then we can choose q subsets of $\{1, \ldots, k\}$ as follows. First choose the empty set, all 1-subsets, all 2-subsets, and all sets in $B_3^<$. Further, choose the sets $A_1, \ldots, A_\ell$ of a 3-Partition. Then we have a total number of $1 + k + \binom{k}{2} + |B_3^<| + \ell = q$ subsets. Any $i \in \{1, \ldots, k\}$ appears in $1 + (k-1) + |B_3^<(i)| + 1 = c_i$ of the chosen subsets. Thus, we have a feasible family of subsets for the considered instance of the capacitated set packing problem.

Now let us turn to the backward implication and assume that for the constructed instance of the capacitated set packing problem there exists a feasible family of subsets. Then there exists a family S of q pairwise distinct subsets of $\{1, \ldots, k\}$ such that each $i \in \{1, \ldots, k\}$ appears in at most c_i of the subsets in S. We show by contradiction that the empty set as well as all 1-subsets of $\{1, \ldots, k\}$, all 2-subsets of $\{1, \ldots, k\}$, and all sets in $B_3^<$ must be in S.

1. $\emptyset \in S$.
   Assume that $\emptyset \notin S$. Since we have q subsets in S, it holds that
   $$\sum_{\mathcal{M}\in S} |\mathcal{M}| \geq k + 2\binom{k}{2} + 3(|B_3^<| + \ell + 1),$$
   because the first sum is smallest, if we use all k of the 1-subsets of $\{1,\ldots,k\}$, all $\binom{k}{2}$ of the 2-subsets of $\{1,\ldots,k\}$, and no subsets of $\{1,\ldots,k\}$ with more than three elements. From the group of 3-subsets we have to pick at least $|B_3^<| + \ell + 1$ to end up with a total number of q subsets.
   The total capacity of elements in $\{1,\ldots,k\}$ is $\sum_{i=1}^{k} c_i = 3|B_3^<| + 2\binom{k}{2} + 2k$ which is less than $\sum_{\mathcal{M}\in S} |\mathcal{M}|$—a contradiction.
2. All 1-subsets have to be in S.
   Assume that there is a 1-subset which is not in S. By an argument analogous to the one above, it follows that
   $$\sum_{\mathcal{M}\in S} |\mathcal{M}| \geq (k-1) + 2\binom{k}{2} + 3(|B_3^<| + \ell + 1),$$
   because the first sum is smallest, if we use the empty set, which does not appear in the latter sum, the $k-1$ remaining 1-subsets of $\{1,\ldots,k\}$, all $\binom{k}{2}$ of the 2-subsets of $\{1,\ldots,k\}$, and no subsets of $\{1,\ldots,k\}$ with more than three elements.
   Again the sum of elements used in total is larger than the total capacity of elements in $\{1,\ldots,k\}$.
3. All 2-subsets have to be in S.
   Assume that there is a 2-subset which is not in S. Here it holds that
   $$\sum_{\mathcal{M}\in S} |\mathcal{M}| \geq k + 2\left(\binom{k}{2} - 1\right) + 3(|B_3^<| + \ell + 1),$$
   which is again larger than the total capacity of elements in $\{1,\ldots,k\}$.
4. All sets in $B_3^<$ have to be in S.
   Assume that there are $r \geq 1$ subsets in $B_3^< \setminus S$. To obtain a total number of q subsets in S, the selection must contain $r + \ell$ subsets of $\{1,\ldots,k\}$ whose respective sum of elements is greater or equal to B. Let the family of these subsets be denoted by T.
   Since we know that all 1-subsets of $\{1,\ldots,k\}$ and all 2-subsets of $\{1,\ldots,k\}$ are in S, an element $i \in \{1,\ldots,k\}$ may be used by at most $c_i - 1 - (k-1) = |B_3^<(i)| + 1$ other subsets. The total weigth of elements that may be used is therefore at most
   $$\sum_{i=1}^{k} s_i(|B_3^<(i)| + 1) = \ell B + \sum_{i=1}^{k} s_i |B_3^<(i)| = \ell B + \sum_{\mathcal{M}\in B_3^<} \sum_{i\in\mathcal{M}} s_i.$$
   The total weigth of elements that is needed by T and the $|B_3^<| - r$ sets from $B_3^< \cap S$ is given by
   $$\sum_{\mathcal{M}\in T} \sum_{i\in\mathcal{M}} s_i + \sum_{\mathcal{M}\in B_3^<\cap S} \sum_{i\in\mathcal{M}} s_i.$$

It follows that

$$\sum_{\mathcal{M}\in T}\sum_{i\in\mathcal{M}} s_i + \sum_{\mathcal{M}\in B_3^<\cap S}\sum_{i\in\mathcal{M}} s_i \le \ell B + \sum_{\mathcal{M}\in B_3^<}\sum_{i\in\mathcal{M}} s_i.$$

This implies

$$(\ell + r)B \le \sum_{\mathcal{M}\in T}\sum_{i\in\mathcal{M}} s_i \le \ell B + \sum_{\mathcal{M}\in B_3^<\setminus S}\sum_{i\in\mathcal{M}} s_i < \ell B + rB,$$

which yields a contradiction.

Hence, we know that the empty set, all 1-subsets of $\{1,\dots,k\}$, all 2-subsets of $\{1,\dots,k\}$, and all sets in $B_3^<$ must be in S. Then, for each $i \in \{1,\dots,k\}$, we have a remaining capacity of $c_i - 1 - (k-1) - |B_3^<(i)| = 1$. Since S contains q subsets, it uses $q - 1 - k - \binom{k}{2} - |B_3^<| = \ell$ subsets in addition to the ones mentioned. These subsets must be 3-subsets, since we have only 3ℓ elements and all subsets with less than three elements are already used. Thus, they form a partition of $\{1,\dots,k\}$. Further, the respective sum of weigths of the elements in the subsets must be greater or equal to B, since all sets in $B_3^<$ are also already used. But now it follows immediately that for each of these subsets its total sum equals B, because $\sum_{i=1}^k s_i = \ell B$ and if any subset had a total weigth of more than B, another subset would have to have one of less than B.

These arguments show that for the considered instance of 3-PARTITION a feasible partition exists if there is a feasible family of subsets for the constructed instance of the capacitated set packing problem.

As already mentioned, in this reduction all numbers (in particular q) are polynomial in k. This proves that the capacitated set packing problem is strongly NP-hard even if q is polynomially bounded in k. For any instance to this restricted problem, it can be checked in time polynomial in its encoding size whether a given family of subsets is a solution for this instance. □

The following is an immediate implication of the reduction shown in the preceding proof.

Corollary 3.2 *For an integral one-flow given in arc variables, it is NP-hard to compute an integral path decomposition.*

A reduction to 3-SAT proves that the situation is even worse: The integral max-1FP is APX-hard, even in networks where the number of s-t-paths is polynomially bounded in the size of the network.

Theorem 3.3 *The integral max-1FP is APX-hard, even if we restrict it to networks in which the number of s-t-paths is polynomially bounded in the size of the network.* (*Thus, unless P = NP, there does not exist a PTAS that runs in time polynomial in input plus output size.*)

Proof We use a reduction of MAX 3SAT-3—a special version of MAX 3SAT. In contrast to the usual 3SAT problem, MAX 3SAT-3 has the additional constraint on the input that each variable appears in at most three clauses. MAX 3SAT-3 is APX-hard (Papadimitriou 1994).

Let the considered 3SAT instance consist of clauses C_i, for $i = 1, \ldots, m$. We say that a variable *appears purely* if it turns up unnegated. We assume that no variable appears only in exactly one clause or appears either only purely or only negated. This can be assumed without loss of generality, because the values of such variables can be set in a preprocessing step.

The corresponding instance of the max-1FP is constructed as follows. First insert a source s, a sink t, and nodes c_i with arcs (s, c_i) for all clauses C_i $(i = 1, \ldots, m)$. We need to distinguish three types of variables:

1. Variables that appear exactly twice, once purely and once negated.
2. Variables that appear purely once and negated twice.
3. Variables that appear purely twice and negated once.

For each variable of the first type we insert a node v, an arc (v, t) and arcs (c_i, v) for both $i \in \{1, \ldots, m\}$ for which C_i contains the considered variable. In the following let V_1 be the set of nodes in the constructed graph resulting from these variables.

For each variable x of the second type we insert the following gadget (see also Fig. 3): The gadget consists of six nodes v, v_1, v_2, v_3, v_4, and w and the arcs (v_1, v_2), (v_2, v_3), (v_3, v_4), (v, v_1), (v_2, v), (v_4, v), (w, v_1), (w, v_3), and (v_4, w). It is connected to s and t by the arcs (w, t) and (v, t), an arc (c_i, v) for $i \in \{1, \ldots, m\}$ for which C_i contains x, and arcs (c_j, w) for both $j \in \{1, \ldots, m\}$ for which C_j contains $\neg x$. The described gadget is also denoted by the shortcut given in Fig. 3. The important property of this gadget is that—given arc capacities equal to 1—it can either route one unit of flow from v to w or one or two units of flow from w to v, but not both.

For each variable x of the third type we insert the same gadget as above, but connect it to the rest of the network the other way around. For the clause C_i which contains $\neg x$ we insert an arc (c_i, v), for the clauses C_j and C_k containing x we add arcs (c_j, w) and (c_k, w). Additionally, we add again arcs (w, t) and (v, t).

The arc capacities are given as follows: Each arc in a gadget and all arcs (v, t), (c_i, v) resulting from variables of the first type have capacity 1. All other arcs (c_i, v), (c_i, w) have capacity 2 and all other arcs (v, t), (w, t) capacity 3. The capacity of an arc (s, c_i) equals the number of arcs of capacity 2 leaving c_i plus 1. (See Fig. 4 for an example of the reduction.)

Now let $D = (V, A)$ with capacities $u : A \to \mathbb{R}^+$ be the reduction network for the considered instance $C = \bigwedge_{i=1}^{m} C_i$ of MAX 3SAT-3. Let C^* be the set of clauses

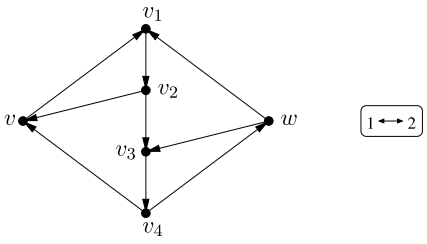

Fig. 3 Gadget for the reduction of 3SAT to max-1FP with shortcut notation on the *right*

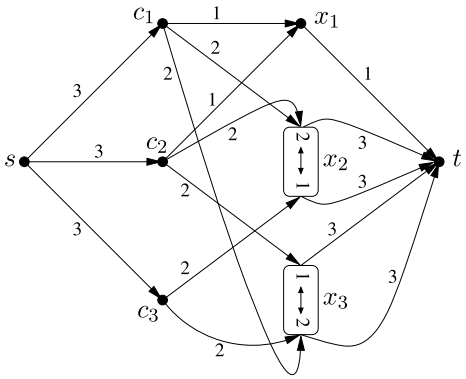

Fig. 4 Reduction network for the SAT-instance $(\neg x_1 \vee \neg x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_2 \vee x_3)$. All arcs entering or leaving a gadget are incident to the corresponding v or w respectively

that are satisfied in an optimal solution to this instance. Further, let L_3 be the set of variables that appear in C exactly three times.

Claim The value of a maximum integral one-flow in D with capacity function u is exactly $3|L_3| + |C^*|$.

We first show that there exists an integral one-flow in D having value $3|L_3| + |C^*|$. Assume that we have an assignment for the variables in C that satisfies the clauses in C^*. Then send one unit of flow along each s-t-path not using arcs in a gadget or a node in V_1. This gives us $3|L_3|$ units of flow, because for each variable x in L_3 there are exactly three s-t-paths using v and w in the gadget of x, but no arc of it. Due to our construction of the gadgets we can additionally send one unit of flow for each satisfied clause through the gadget (or the node from V_1 respectively) that belongs to one of the variables by which the clause is satisfied. Since this gives us another $|C^*|$ units of flow, it follows that there exists an integral one-flow of value $3|L_3| + |C^*|$.

It remains to show that there cannot be an integral one-flow of value greater than $3|L_3| + |C^*|$. Assume that we have an integral one-flow f of value $3|L_3| + k$, for some $k \in \mathbb{N}$. We show that then there must exist an assignment for the variables in C such that at least k clauses of C are satisfied. We know that f can send at most $3|L_3|$ units of flow that traverse neither a gadget nor a node from V_1. This implies immediately that at least k units of flow must be routed through gadgets or V_1. It remains to show that there are at least k clauses for which a unit of flow is routed from its corresponding node to t by traversing a gadget or a node in V_1. Then the flow defines an assignment for the variables in C as follows: A variable is set TRUE if the inflow in the corresponding gadget (or in the node from V_1) comes from a clause in which it appears purely and FALSE otherwise. The number of satisfied clauses is then k.

If f routes at most one unit of flow for any clause from its corresponding node to t by traversing a gadget or a node in V_1, we are done. Thus, let us assume that there is a clause for which f routes $\ell \geq 2$ units of flow in such a way. Due to the capacities of the arcs (s, c_i), for $i = 1, \ldots, m$, it follows that at most $3|L_3| - (\ell - 1)$ units of flow can be sent without using gadgets or V_1, which means for the remaining clauses that they must send $k - 1$ units of flow through gadgets or V_1. By induction it follows that

there must be at least k clauses for which at least one unit of flow is routed through a gadget or V_1.

Thus, we showed that the value of a maximum integral one-flow in D is at least $3|L_3| + |C^*|$ and that for any integral one-flow of value $3|L_3| + k$ we can find an assignment for the variables in C such that k clauses in C are satisfied. This implies that $3|L_3| + |C^*|$ is the value of any maximum integral one-flow and thus proves the claim.

Since the assignment of values to the variables in C can immediately be taken from the computed one-flow (see above), it only remains to prove that the number of paths in D is polynomially bounded in the size of D. Let X be the number of variables in C, then this follows immediately from the fact that $10X$ yields an upper bound for the number of paths in D. We can bound the number of paths in D by $10X$, because at each structure representing a clause (either a gadget or a node in V_1) we have at most three entering arcs. An arc entering a node in V_1 results in only one path to t. Arcs entering a node v of a gadget result in two paths to t—one through the gadget and a "direct" one. And arcs entering a node w of a gadget result in four paths to t—three through the gadget and a "direct" one. Since there is only one possibility to reach either of the considered arcs from s and three arcs (only two of the same type) enter a gadget, we have at most ten paths per variable. □

Note that it follows from the strong NP-hardness of the decision problem 3SAT-3 that it is even strongly NP-hard to compute only the value of a maximum integral one-flow in networks. This result even holds if we restrict to networks in which the number of s-t-paths is polynomially bounded in the size of the network.

As an immediate consequence of Theorem 3.1 we can state the following hardness results.

Theorem 3.4

(i) *The bounded cut cover problem is strongly NP-hard. If q is polynomially bounded in k, then the problem is strongly NP-complete.*

(ii) *The problem of finding an integral one-flow of maximum value for a chain of parallel arcs is strongly NP-hard, even if the maximum flow value is polynomially bounded in the size of the network* (*i.e., number of vertices*).

It follows that, in contrast to the problem to count the number of s-t-paths in a digraph, the integral max-1FP is already strongly NP-hard in acyclic networks. Further, the strong NP-hardness of the capacitated set packing problem immediately implies that it is even strongly NP-hard to compute only the value of a maximum integral one-flow in those networks. Note that in the integral one-flow problem the flow value bounds the number of paths that are used to route a flow. Thus, we can derive the following from Theorem 3.4(ii).

Corollary 3.5 *Unless* P = NP, *there is no algorithm for the integral max-1FP on chains of parallel arcs whose runtime is pseudo-polynomial in input plus output size.*

Another consequence of our considerations made so far is the following.

Theorem 3.6 *It is NP-hard to decide whether a given (integral) s-t-flow can be decomposed into integral flows along paths and cycles such that no path carries more than one unit of flow.*

4 An FPTAS for the fractional max-1FP

Theorem 4.1 *For any $\epsilon > 0$ and any instance of the max-1FP with maximum flow value F^*, it is possible to compute a one-flow of value $(1-\epsilon)F^*$ in time polynomial in the input size, ϵ^{-1}, and F^*.*

First we show how to compute a one-flow of a given value F that does not violate any arc capacity by more than a factor $(1+\epsilon)$, for some $\epsilon > 0$, or decide that no valid flow of value F exists. A flow violating all arc capacities by at most a factor $(1+\epsilon)$ is called $(1+\epsilon)$-*approximate*.

Plotkin et al. (1995) developed an appropriate algorithm for the general fractional set packing problem. In that problem sets of capacitated elements are given and the task is to search for a packing, i.e., a selection of sets such that each element is contained in at most as many sets as its capacity permits. To compute $(1+\epsilon)$-approximate packings of a given size, Plotkin et al. use a Lagrange relaxation that penalizes the violation of the capacity constraints in the objective function. Iteratively, they choose reasonable Lagrange multipliers, compute a solution to the relaxed problem, and combine this with the current solution. This algorithm runs in time polynomial in the input size and ϵ^{-1}.

The approach in Plotkin et al. (1995) can be adapted to the fractional 1FP. Details about this can be found in Martens (2007). However, we also want to provide the main ideas of the methods here: For given Lagrange multipliers $\lambda : A \to \mathbb{R}^+$ and $\lambda_P := \sum_{a\in P} \lambda(a)$ for all s-t-paths $P \in \mathcal{P}$, the adjusted Lagrange relaxation is the following.

$$\begin{aligned} \min \quad & \sum_{P\in\mathcal{P}} \lambda_P x_P \\ \text{s.t.} \quad & \textstyle\sum_{P\in\mathcal{P}} x_P \geq F \\ & 0 \leq x_P \leq 1 \quad \forall P \in \mathcal{P} \end{aligned}$$

This problem can be solved in time polynomial in the input size and $\lceil F \rceil$ by computing the $\lceil F \rceil$ shortest paths according to the length function λ (see, e.g., Lawler 1972). The $\lfloor F \rfloor$ shortest paths must carry one unit of flow ($x_P = 1$) and the $(\lfloor F \rfloor + 1)$-shortest path gets a flow value of $F - \lfloor F \rfloor$.

We assume without loss of generality that $\epsilon < 1$. To obtain a one-flow that obeys all arc capacities and approximates the maximum flow value F^* within a factor $(1-\epsilon)$, we embed the algorithm by Plotkin et al. in a binary search to find a value $\bar{F}$ with $\bar{F} \leq F^* \leq (1+\tilde{\epsilon})\bar{F}$, for $\tilde{\epsilon} = (1-\epsilon)^{-1/2} - 1$. Then the desired flow is obtained by computing a $(1+\tilde{\epsilon})$-approximate one-flow of value $\bar{F}$ and dividing all its flow values by $(1+\tilde{\epsilon})$. The resulting flow has value $(1+\tilde{\epsilon})^{-1}\bar{F} \geq (1+\tilde{\epsilon})^{-2}F^* = (1-\epsilon)F^*$.

The binary search works as follows. For increasing values $\ell \in \mathbb{N}$ (starting with $\ell = 0$) we use the algorithm by Plotkin et al. to compute a 2-approximate one-flow of value 2^ℓ or decide that no feasible one-flow of this value exists. We stop with the first value $\bar{\ell}$ for ℓ for which the answer is that no feasible one-flow of value $\bar{F} := 2^{\bar{\ell}}$ exists. This takes $O(\log F^*)$ steps. It follows that $\bar{F}/4 \leq F^* < \bar{F}$. Using geometric binary search now, we obtain $\tilde{F}$ with $\tilde{F} \leq F^* \leq (1+\tilde{\epsilon})\tilde{F}$ within $O(\log(1/\tilde{\epsilon})) = O(\log(1/\epsilon))$ steps. Thus, we have $O(\log F^* + \log(1/\epsilon))$ steps in total.

The algorithm is polynomial in input plus output size and ϵ^{-1}. Note that there are $O(\log F^* + \log(1/\epsilon))$ steps in which we compute an approximate one-flow. Computing $\lceil 2^\ell \rceil$ shortest paths takes $O(F^*)$ time in each step. And we need at least $(1-\epsilon)F^*$ paths to present a one-flow of value $(1-\epsilon)F^*$.

5 Approximating the integral max-1FP

We assume that all arc capacities are integral and start by proving that, for the integral max-1FP, the additive integrality gap is at most m. This result follows from basic linear programming theory.

Proposition 5.1 *The difference of the value F_F^* of a maximum fractional one-flow and the value F_I^* of a maximum integral one-flow is less than m.*

Proof Consider an optimal *basic* solution x to the maximum one-flow LP given in Sect. 1. Then x is a maximum fractional one-flow of value F_F^*. In a basic solution to an LP, the number of constraints that are tight is at least equal to the dimension, i.e., the number of variables. Since there are m capacity constraints (1), at least $|\mathcal{P}| - m$ constraints in (2) are tight. In particular, all but at most m paths $P \in \mathcal{P}$ carry zero or one unit of flow. Removing the flow on those $\leq m$ paths decreases the flow value by less than m and thus yields an integral one-flow of value greater than $F_F^* - m$. □

Combining this result with the FPTAS from the previous section we can prove the following in a similar manner.

Corollary 5.2 *For any $\epsilon > 0$, an integral one-flow of value at least $(1-\epsilon)F_F^* - m$ can be computed in time polynomial in the input size and F_I^*.*

Proof By Theorem 4.1 we can compute a fractional one-flow $(\bar{x}_P)_{P\in\mathcal{P}}$ of value at least $(1-\epsilon)F_F^*$ for some constant $\epsilon > 0$ in time polynomial in the input size and F_F^*. Using this flow we compute another one-flow of the same or larger value as follows. Let $\bar{\mathcal{P}} := \{P \in \mathcal{P} \mid \bar{x}_P > 0\}$ be the set of paths with positive flow values in $\bar{x}$. Then

the size of the system

$$\sum_{P\in\bar{\mathcal{P}}} x_P \geq \sum_{P\in\bar{\mathcal{P}}} \bar{x}_P$$

$$\sum_{\substack{P\in\bar{\mathcal{P}}:\\ a\in P}} x_P \leq u(a) \quad \forall a \in A$$

$$0 \leq x_P \leq 1 \quad \forall P \in \bar{\mathcal{P}}$$

is polynomial in the input size of the original problem and F_F^*. It follows that a basic solution to this system can be computed in polynomial time. The flow value of this solution is at least $(1-\epsilon)F_F^*$. For the fractional part of this basic solution, we can argue like in the proof of Proposition 5.1 that there are at most m paths carrying a fractional amount of flow. (Note that all vectors now have an additional 1 in the first entry.) Thus, we obtain an integral one-flow of value at least $(1-\epsilon)F_F^* - m$ when dropping the fractional part of the considered basic solution. Since $F_F^* \leq F_I^* + m$, the described procedure can be implemented to run in time polynomial in the input size and F_I^*. □

As an immediate consequence of Corollary 5.2, we obtain the following approximation result.

Theorem 5.3 *There exists a constant factor approximation algorithm for the integral max-1FP with runtime polynomial in input plus output size when we restrict to instances with maximum fractional flow value greater than or equal to some constant $c > 1$ times its number of arcs.*

In the remainder of this section we develop a randomized $O(\log m)$-approximation algorithm for the integral max-1FP that works for arbitrary instances. We start with a simple observation.

Observation 5.4 *Applying Raghavan and Thompson's* (1987) *randomized rounding method to a fractional solution computed by the FPTAS from Sect.* 4*, we obtain a randomized constant factor approximation algorithm for the integral max-1FP if the minimum arc capacity is at least $\Omega(\log m)$.*

The algorithm starts with an approximate fractional one-flow x and selects a path $P \in \mathcal{P}$ for the integral one-flow with probability $\gamma \cdot x_P$, for some constant $\gamma < 1$. The resulting one-flow obeys arc capacities with some constant positive probability that depends on the choice of γ. (The analysis is similar to the one given by Kleinberg 1996, more details can be found in Martens 2007.)

In order to obtain a randomized $O(\log m)$-approximation algorithm for arbitrary instances of the integral max-1FP, we compute an approximate maximum integral one-flow in a modified network by giving a special treatment to arcs whose capacity is smaller than $\log m$. We call such arcs *thin*, whereas an arc with capacity at least $\log m$ is called *thick*. A path is called *thick* if all its arcs are thick; otherwise it is called *thin*.

For a given instance of the max-1FP, we compute an approximate solution to the fractional problem using the FPTAS from Sect. 4 for some constant $\epsilon > 0$. If the total flow value along thick paths is at least half of the total flow value (and thus at least a constant fraction of the maximum integral flow value), we can use randomized rounding as explained above in order to obtain a constant factor approximation. Otherwise we can use the algorithm described in the following which computes an $O(\log m)$-approximation from the flow that is routed along thin paths.

The algorithm works as follows. First we delete the flow routed along thick paths. From now on we consider only the part of the underlying graph which is used by thin paths. For each thin arc (v, w) insert a new node $\tilde{v}$, delete the arc (v, w), insert the arcs $(v, \tilde{v})$ and $(\tilde{v}, w)$ and assign the capacity of (v, w) to either one. (The flow is adjusted adequately using arcs $(v, \tilde{v})$ and $(\tilde{v}, w)$ instead of (v, w).) The resulting network is denoted by $D = (V, A)$, the set of newly inserted nodes by $U \subseteq V$. Next, we make a copy $D' = (V', A')$ of D. From each node in U we insert an arc to its copy in D'. The resulting graph is denoted by $\bar{D} = (\bar{V}, \bar{A})$.

We define v' to be the clone of $v \in V$ in V' and a' to be the clone of $a \in A$ in A'. An arc connecting a node $u \in U$ with $u' \in V'$ is denoted by a_u. For any $u \in U$ that was inserted to divide an arc a of the original digraph, the capacity of a_u is set to be the same as that of a.

We modify the considered fractional flow by rerouting all its paths from D to D' along the last thin arc at which a rerouting is possible. More precisely, this works as follows. Consider any path P that is used in the original fractional flow and let $(v, w) \in P$ be the last thin arc on P. (This arc does not exist in D.) Then the analogon to P in $\bar{D}$ uses the adjusted path P in D until it reaches v, then uses $(v, \tilde{v})$ and is rerouted to D' along $a_{\tilde{v}}$. In D' the new path uses the arc from $\tilde{v}'$ to w' and then the arcs corresponding to the ones P used in D after (v, w). (See Fig. 5 for an illustration.)

Let the resulting s-t'-flow be denoted by $\bar{x}$. Note that the value $|\bar{x}|$ of $\bar{x}$ is still only some constant factor smaller than the value of a maximum one-flow in the original network. We choose integral capacities $u(a)$ and $u(a')$ for a thick arc $a \in A$ and its clone a' as follows. If $\lceil \bar{x}(a) \rceil + \lceil \bar{x}(a') \rceil$ is not larger than the original capacity of a, we set $u(a) = \lceil \bar{x}(a) \rceil$ and $u(a') = \lceil \bar{x}(a') \rceil$. Otherwise, we choose the capacities by rounding the smaller flow value up and the larger one down. (The sum of the resulting

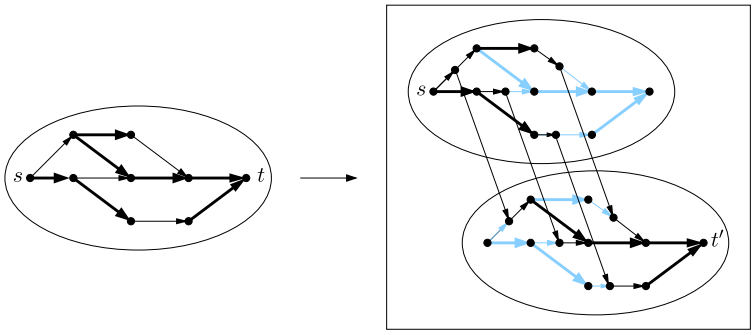

Fig. 5 Rerouting of the original flow on the *left* in the 2-layer graph on the *right*. Gray arcs are not used by the final flow

values is not larger than the original capacity of a, because this was assumed to be integral.) For all thin arcs $a \in \bar{A}$, the capacity $u(a)$ is set to 1. It is easy to prove that $u(a) > \bar{x}(a)/\log m$, for all $a \in \bar{A}$.

Lemma 5.5 *For each arc $a \in \bar{A}$, it holds that $u(a) > \bar{x}(a)/\log m$.*

Proof First let $a \in A$ be a thick arc. If $\lceil \bar{x}(a) \rceil + \lceil \bar{x}(a') \rceil$ is not larger than the original capacity of a, it obviously holds that $u(a) \geq \bar{x}(a)$ and $u(a') \geq \bar{x}(a')$. For the other case, we assume without loss of generality that $\bar{x}(a) \leq \bar{x}(a')$. Then, $u(a) = \lceil \bar{x}(a) \rceil \geq \bar{x}(a)$ and $u(a') = \lfloor \bar{x}(a') \rfloor$ with $\bar{x}(a') \geq 1$, because otherwise $\lceil \bar{x}(a) \rceil + \lceil \bar{x}(a') \rceil < 2$ which is less than the original capacity of a. It follows that $u(a') > \bar{x}(a')/2$.

Now let $a \in \bar{A}$ be a thin arc. Then, $u(a) = 1 > \bar{x}(a)/\log m$, because the original capacity of a was less than $\log m$. □

It follows immediately that a(n ordinary) maximum s-t'-flow in $\bar{D}$ with capacities u has flow value at least $|\bar{x}|/\log m$, because $\bar{x}/\log m$ is a feasible s-t'-flow in that network. By network flow theory, an integral maximum s-t'-flow can be computed in polynomial time. Since the value of this flow is at least $|\bar{x}|/\log m$, it is only by a factor $O(\log m)$ smaller than the value of a maximum one-flow in the original network. If we reroute this flow to D, i.e., do not let it pass over from D to D' and let it use the corresponding arcs in D instead, it is still feasible, because the sum of arc capacities $u(a)$ and $u(a')$ is at most the original capacity of $a \in A$. Further, the resulting flow does not send more than one unit of flow along each path, because each path uses at least one thin arc (otherwise it would not have been able to get from D to D') whose capacity is now 1. This gives us the following result.

Theorem 5.6 *There exists a randomized $O(\log m)$-approximation algorithm for the integral max-1FP with runtime polynomial in input plus output size.*

It follows immediately that the (multiplicative) integrality gap of the max-1FP is $O(\log m)$, because we have always compared the value of our integral solution to that of an optimal fractional solution.

6 Multicommodity one-flows

In this section we consider the multicommodity version of the one-flow problem in that we have several source-sink-pairs. As before we have a digraph $D = (V, A)$ with arc capacities $u : A \to \mathbb{R}^+$. Instead of a single source-sink-pair we now have *requests* $(s_i, t_i) \in V \times V$ for $i = 1, \ldots, K$, where $K \in \mathbb{N}$ denotes the total number of such pairs. We also call (s_i, t_i) the ith *commodity*.

Different optimization problems for multicommodity flows have been considered in the literature. Among them are the maximization of the total flow sent through a network and the minimization of the congestion of a flow that satisfies a given demand for each commodity. (The congestion measures the relative overload on an arc—a detailed definition follows later.) We consider both such optimization problems in the context of one-flows and show for either one that the FPTAS from Sect. 4

can be generalized to its needs. For the integral maximization problem we present an $O(\sqrt{m})$-approximation algorithm in Sect. 6.1. A result by Guruswami et al. (1999) shows that it is NP-hard to approximate the problem within a factor $O(m^{1/2-\epsilon})$, for any $\epsilon > 0$. For the integral minimum congestion problem we present an $O(\log m)$-algorithm in Sect. 6.2.

In the following we use $\mathcal{P}_i$ to denote the set of s_i-t_i-paths in D, for all $i = 1, \ldots, K$, and $\mathcal{P} := \bigcup_{i=1}^{K} \mathcal{P}_i$.

6.1 Maximum multicommodity one-flows

With the definitions given above we can describe the *maximum multicommodity one-flow problem* (*max-mc-1FP*) by the linear program in Sect. 1. Since solving the min-cost LP from Sect. 4 can be adapted to the new situation easily by computing the $\lceil F \rceil$ shortest paths for each commodity and then choosing the $\lceil F \rceil$ overall shortest paths, the FPTAS can be used for the max-mc-1FP without any changes.

Theorem 6.1 *For any $\epsilon > 0$ and any instance of the max-mc-1FP with maximum flow value F^*, it is possible to compute a multicommodity one-flow of value $(1-\epsilon)F^*$ in time polynomial in the input size, ϵ^{-1}, and F^*.*

We turn to integral multicommodity one-flows. A result by Guruswami et al. (1999) shows that it is NP-hard to approximate the problem within a factor $O(m^{1/2-\epsilon})$, for any $\epsilon > 0$. On the other hand, randomized rounding yields an $O(\sqrt{m})$-approximation algorithm for the considered problem. This can be proven using a more detailed analysis of the randomized rounding method given by Baveja and Srinivasan (2000).

Theorem 6.2 *There exists a randomized $O(\sqrt{m})$-approximation algorithm for the integral max-mc-1FP with runtime polynomial in input plus output size.*

Even in the multicommodity case Proposition 5.1 holds. This gives us a constant factor approximation algorithm with runtime polynomial in input plus output size when we restrict the problem to instances with maximum fractional flow value greater than or equal to some constant $\gamma > 1$ times the number of arcs.

6.2 Minimizing congestion

Here, each request (s_i, t_i) $(i = 1, \ldots, K)$ has a corresponding positive demand d_i that has to be satisfied by a one-flow. We look for a solution of minimum congestion. The *congestion* of a flow $(x_P)_{P \in \mathcal{P}}$ is the minimum μ such that $\sum_{P \ni a} x_P \leq \mu u(a)$, for all $a \in A$.

Again, the FPTAS from Sect. 4 can be adapted to the new situation. We obtain the following result.

Theorem 6.3 *For any $\epsilon > 0$ and any instance of the min-cong-1FP with minimum congestion μ^*, it is possible to compute a multicommodity one-flow of congestion at most $(1+\epsilon)\mu^*$ in time polynomial in the input size, ϵ^{-1}, and $d_{\max} := \max_i d_i$.*

It remains to give an approximation for the integral min-cong-1FP. For this case we again apply Raghavan and Thompson's (1987) randomized rounding method to an approximate optimal fractional solution and obtain an $O(\log m)$-approximation algorithm. In the congestion case we do not need any restrictions on the arc capacities and the rounding can even be derandomized using the method of conditional probabilities.

Theorem 6.4 *Applying randomized rounding to a nearly optimal fractional one-flow yields an* $O(\log m)$*-approximation to the integral min-cong-*1*FP.*

References

Ahuja RK, Magnanti TL, Orlin JB (1993) Network flows. Prentice-Hall, Englewood Cliffs

Andrews M, Zhang L (2007) Hardness of the undirected congestion minimization problem. SIAM J Comput 37(1):112–131

Baier G, Erlebach T, Hall A, Köhler E, Schilling H, Skutella M (2006) Length-bounded cuts and flows. In: Proceedings of the 33rd international colloquium on automata, languages and programming, pp 679–690

Bartholdi L (1999) Counting paths in graphs. Enseign Math 45:83–131

Baveja A, Srinivasan A (2000) Approximation algorithms for disjoint paths and related routing and packing problems. Math Oper Res 25:255–280

Chekuri C, Khanna S (2007). Edge-disjoint paths revisited. ACM Trans Algorithms (TALG), 3(4)

Dijkstra EW (1959) A note on two problems in connection with graphs. Numer Math 1:269–271

Dreyfus SE (1969) An appraisal of some shortest path algorithms. Oper Res 17:395–412

Eppstein D (1998) Finding the k shortest paths. SIAM J Comput 28:652–673

Fleischer LK (2000) Approximating fractional multicommodity flow independent of the number of commodities. SIAM J Discrete Math 13(4):505–520

Ford LR, Fulkerson DR (1956) Maximal flow through a network. Can J Math 8:399–404

Fox BL (1975) k-th shortest paths and applications to the probabilistic networks. In: ORSA/TIMS joint national meeting, 23:B263

Garg N, Könemann J (1998) Faster and simpler algorithms for multicommodity flow and other fractional packing problems. In: Proceedings of the 39th annual IEEE symposium on foundations of computer science, pp 300–309

Gessel I (1993) Counting paths in Young's lattice. J Stat Plan Inference 34:125–134

Grigoriadis M, Khachiyan LG (1995a) An exponential-function reduction method for block-angular convex programs. Networks 26(1.2):59–68

Grigoriadis M, Khachiyan LG (1995b) A sublinear-time randomized approximation algorithm for matrix games. Oper Res Lett 18(2):53–58

Grigoriadis M, Khachiyan LG (1996a) Approximate minimum-cost multicommodity flows in $o(\epsilon^{-2}knm)$ time. Math Program 75:477–482

Grigoriadis M, Khachiyan LG (1996b) Coordination complexity of parallel price-directive decomposition. Math Oper Res 21:321–340

Guruswami V, Khanna S, Rajaraman R, Shepherd B, Yannakakis M (1999) Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems. In: Proceedings of the 31st annual ACM symposium on theory of computing, pp 19–28

Klein P, Plotkin SA, Shmoys DB, Tardos E (1994) Faster approximation algorithms for the unit capacity concurrent flow problem with applications to routing and finding sparse cuts. SIAM J Comput 23(3):466–487

Kleinberg JM (May 1996) Approximation algorithms for disjoint path problems. PhD thesis, Massachusetts Institute of Technology

Kolliopoulos SG (2007) Edge-disjoint paths and unsplittable flow. In: Gonzalez TF (ed) Handbook of approximation algorithms and metaheuristics. Chapman & Hall/CRC, London

Kolliopoulos SG, Stein C (2004) Approximating disjoint-path problems using packing integer programs. Math Program Ser A 99:63–87

Lawler EL (1972) A procedure for computing the K best solutions to discrete optimization problems and its application to the shortest path problem. Manag Sci 18:401–405
Leighton T, Makedon F, Plotkin SA, Stein C, Tardos E (1995) Fast approximation algorithms for multicommodity flow problems. J Comput Syst Sci 50(2):228–243
Lucas JF (1983) Paths and Pascal numbers. Two-Year College Math J 14(4):329–341
Martens M (2007) Path-constrained network flows. PhD thesis, Universität Dortmund
Martens M, Skutella M (2006) Flows on few paths: Algorithms and lower bounds. Networks 48(2):68–76
Papadimitriou CH (1994) Computational complexity. Addison-Wesley, Reading
Plotkin SA, Shmoys DB, Tardos E (1995) Fast approximation algorithms for fractional packing and covering problems. Math Oper Res 20:257–301
Raghavan P, Thompson CD (1987) Randomized rounding: A technique for provably good algorithms and algorithmic proofs. Combinatorica 7:365–374
Roditty L (2007) On the k-simple shortest paths problem in weighted directed graphs. In: Proceedings of the 18th annual ACM-SIAM symposium on discrete algorithms, pp 920–928
Schrijver A (2003) Combinatorial optimization. Polyhedra and efficiency. Springer, Berlin
Shahrokhi F, Matula DW (1990) The maximum concurrent flow problem. J ACM 37:318–334
Stanley RP (1996) A matrix for counting paths in acyclic digraphs. J Comb Theory Ser A 74(1):169–172
Valiant LG (1979) The complexity of enumeration and reliability problems. SIAM J Comput 8(3):410–421
Yen JY (1971) Finding the k shortest loopless paths in a network. Manag Sci 17(11):712–716
Young NE (2001) Sequential and parallel algorithms for mixed packing and covering. In: IEEE symposium on foundations of computer science, pp 538–546