Complexity and Approximability of k–Splittable Flows

Ronald Koch *

Universität Dortmund, Fachbereich Mathematik, 44221 Dortmund, Germany, ronald.koch@math.uni-dortmund.de, tel. 0049 231 7557210, fax 49 231 7557215

Ines Spenke *[,1]

Technische Universität Berlin, Institut für Mathematik, 10623 Berlin, Germany, spenke@math.tu-berlin.de, tel. 0049 30 31425739, fax 0049 30 31425191

Abstract

Let $G = (V, E)$ be a graph with a source node s and a sink node t, $|V| = n$, $|E| = m$. For a given number k, the *Maximum k–Splittable Flow Problem* (MkSF) is to find an s, t–flow of maximum value with a flow decomposition using at most k paths. In the multicommodity case this problem generalizes disjoint paths problems and unsplittable flow problems.

We provide a comprehensive overview of the complexity and approximability landscape of MkSF on directed and undirected graphs. We consider constant values of k and k depending on graph parameters. For arbitrary constant values of k, we prove that the problem is strongly NP–hard on directed and undirected graphs already for $k = 2$. This extends a known NP–hardness result for directed graphs that could not be applied to undirected graphs. Furthermore, we show that MkSF cannot be approximated with a performance ratio better than 5/6. This is the first constant bound given for this value. For non constant values of k, the polynomial solvability was known before for all $k \geq m$, but open for smaller k. We prove that MkSF is NP–hard for all k fulfilling $2 \leq k \leq m - n + 1$ (for $n \geq 3$). For all other values of k the problem is shown to be polynomially solvable.

Key words: s, t–flow; k–splittable flow; complexity; approximation.

* Corresponding author.

[1] Supported by the DFG Research Center MATHEON "Mathematics for key technologies".

1 Introduction

In classic flow theory, flow is sent through a network from sources to sinks respecting edge capacities. It does not matter how many paths the flow uses. It can split into small flow portions along a large number of paths. Many applications in transport, telecommunication, production or traffic are modeled as flow problems but they do not allow to split into an unbounded number of paths with possibly tiny flow portions. In logistics for example, paths often mean vehicles that are used to transport goods. Usually, this transport has to be done with a limited number of vehicles. The problem that we consider here limits the number of paths that are used in a flow by a given integer k.

Problem Description. Let $G = (V, E)$ be a connected undirected or directed graph with n nodes and m edges, capacities $u : E \to \mathbb{Q}_{\geq 0}$, a source $s \in V$ and a sink $t \in V$. Furthermore, a number $k \in \mathbb{N}$ is given. A flow is called *k–splittable* if it can be decomposed using at most k paths. The paths are not required to be disjoint, not even different. This k–splittability has been introduced recently by Baier, Köhler, and Skutella [4]. In the *Maximum k–Splittable Flow* problem (MkSF) we ask for a k–splittable s, t–flow of maximum value. Of course, k–splittability can also be considered in the more general multi–commodity setting. Then it generalizes unsplittable flow problems and disjoint paths problems. In this paper we consider the single commodity case.

Results from the Literature. Many publications consider s, t-flows with no bound on the number of used paths, see e.g. Ford and Fulkerson [5]. It is well known that a maximum s, t–flow can be computed in polynomial time, for example, by augmenting path algorithms. Another classical result states that any s, t–flow can be decomposed into flow along at most m paths and cycles. For further details see the book by Ahuja, Magnanti, and Orlin [1].

Kleinberg [6] studies *unsplittable flows*. These multicommodity flows send the entire demand for each commodity along one path. This concept generalizes edge–disjoint paths. Kleinberg analyses complexity and approximation algorithms for different unsplittable flow problems, e.g. for minimizing the congestion on edges or equivalently maximizing the throughput, for the problem of minimizing the number of rounds needed to satisfy all demands, and for the problem of maximizing the total demand which can be routed simultaneously. In the multicommodity setting, k–splittable flows constitute a generalization of unsplittable flows.

Baier, Köhler, and Skutella [4] (see also [3]) investigate k–splittable flows in the single- and in the multicommodity setting. They prove NP–hardness of

MkSF in directed graphs for all constant $k \geq 2$. For the special case of the *uniform* MkSF, where all k paths must carry the same amount of flow, they give a max flow – min cut result as well as an $O(k\ m \log n)$ algorithm for an optimal solution. Based on these insights, they present 1/2–approximation algorithms for the general MkSF problem. Bagchi, Chaudhary, Scheideler, and Kolman [2] consider fault tolerant routing in networks and define notions similar to k–splittable flows. To ensure connection for each commodity for up to $k-1$ edge failures in the network, they require edge disjoint flow–paths per commodity. Martens and Skutella [9] consider a new variant of k–splittable multicommodity flows with upper bounds on the amount of flow sent along each path. The objective is to minimize the congestion on arcs. They prove that any ρ–approximation for the unsplittable flow problem gives a 2ρ–approximation for two different variants of the considered problem.

Krysta, Sanders, and Vöcking [8] consider related problems in the area of machine scheduling problems by imposing a bound on the number of preemptions of each task. In their k–splittable scheduling problem, each task can be split into at most $k \geq 2$ pieces that are assigned to different machines. They describe a polynomial time algorithm for finding an exact solution for the k–splittable scheduling problem and a slightly more general problem. This algorithm has a running time which is exponential in the number of machines but linear in the number of tasks.

Koch, Skutella, and Spenke [7] decouple the problem MkSF into two steps. A first step called *Packing* determines a set of path flow value candidates for solving MkSF to optimality for constant k and near-optimal for k being part of the input. A second step called *Routing* finds out which path flow value tuples can be routed. The packing procedure is described for general graphs, the routing for graphs of bounded treewidth. Finally, they get a polynomial algorithm for MkSF on graphs of bounded treewidth if k is constant and a PTAS if k is part of the input.

Our Paper. In this paper, we investigate the complexity and approximability of MkSF for different values of k.

The well-known maximum s,t–flow problem is polynomially solvable. Our problem MkSF differs from it by allowing at most k paths. We show that this additional requirement results in a strongly NP–hard problem on directed and undirected graphs for arbitrary constant k, even for $k = 2$. From this result we derive a constant bound for the approximability of MkSF. It cannot be approximated better than with a guarantee of 5/6, unless $P = NP$.

It is a known result that each s,t–flow can be decomposed into flow along at most m paths and cycles. Thus, for large k in relation to m MkSF is obviously polynomially solvable. Until now it was not clear what happens with e.g. $k = m/2$. We prove that the problem is NP-hard for all k within the range $2 \leq k \leq m - n + 1$ and polynomially solvable for $k \geq m - n + 2$ for $n \geq 3$. We add some results for simple graphs. These are graphs without parallel edges and loops.

For the sake of simplicity we restrict ourselves to undirected graphs, but any result can be applied to the directed case by minor modifications in the proofs. To the best of our knowledge, all publications on k–splittable flows so far deal with directed graphs.

2 Constant Number of Paths

In this section we consider constant values of $k \geq 2$. MkSF is shown to be strongly NP–hard. We show that there is no approximation algorithm with performance ratio better than 5/6. To the best of our knowledge, there was no constant bound given before.

In [4] the NP–hardness of MkSF is proven for constant $k \geq 2$ in directed graphs. The construction given there cannot be applied to undirected graphs. Theorem 1 and Corollary 2 prove NP–hardness for both. To simplify notation, we denote the problem MkSF with $k = 2$ by M2SF, as well for other k.

Theorem 1 M2SF *is strongly NP–hard and cannot be approximated in polynomial time with a guarantee better than* $2/3$*, unless* $P = NP$*.*

PROOF. We reduce 3SAT to M2SF and show that a satisfiable instance of 3SAT yields an optimum solution of value 3, whereas an unsatisfiable instance yields an optimum of value 2 for the corresponding M2SF–instance.

Consider an instance of 3SAT with variables $x_1, ..., x_r$ and clauses $C_1, ..., C_q$. We construct the corresponding M2SF–instance in two steps illustrated in Fig. 1 and Fig. 2. The entire construction is shown in Fig. 3.

Step 1, Fig. 1: The graph constructed in this step represents the clauses of the 3SAT–instance. Introduce two nodes s and t and for every clause C_j two nodes a_j, b_j. Introduce three parallel edges between a_j and b_j for each j. Each of these edges belongs to a literal that occurs in the clause C_j. Later in Step 2, we will insert additional nodes into these edges such that they finally build a_j, b_j–paths. The nodes a_j, b_j together with the three edges among them are said to "represent" the clause C_j. Connect the representations of the clauses by the $q+1$ edges $\{s, a_1\}, \{b_1, a_2\}, \{b_2, a_3\}, ..., \{b_q, t\}$. All edges created in this step get capacity 1. The construction so far allows s, t–paths traversing each clause along one path representing one literal of the clause. To control that such s,t–paths do not use paths that belong to contrary literals we introduce a blocking construction in Step 2.

Step 2, Fig. 2: We want to avoid that an s,t–path traverses the graph using an edge representing a literal x_i and also an edge representing $\bar{x}_i$. Assume that there are h pairs of contrary literals x_i and $\bar{x}_i$. Consider the l-th pair and assume that x_i appears in a clause C and $\bar{x}_i$ in a clause C'. Insert one edge $\{y_l, z_l\}$ into an edge $\{u, v\}$ of unit capacity of the path representing x_i. The new edges $\{u, y_l\}$ and $\{z_l, v\}$ get a capacity of 1 and the edge $\{y_l, z_l\}$ gets a capacity of 2. Analogously, insert an edge $\{y'_l, z'_l\}$ into an edge $\{u', v'\}$ of unit capacity of the path representing $\bar{x}_i$. Introduce two nodes c_l and d_l and edges $\{c_l, y_l\}, \{d_l, z_l\}, \{c_l, y'_l), \{d_l, z'_l\}$ with capacities 2 to get a blocking construction for the l-th pair of contrary literals. This has to be done for all pairs of contrary literals for all pairs of clauses C, C'. To complete the construction we add edges $\{s, c_1\}, \{d_1, c_2\}, \{d_2, c_3\}, ..., \{d_{h-1}, c_h\}, \{d_h, t\}$, also with capacities 2.

Figure 3 shows the entire construction for an instance. This reduction is of polynomial size: The number of nodes is at most quadratic in the number q of clauses and the number of edges is bounded due to a maximum node degree of 4. Furthermore, any s,t–flow has a value no more than 3, because the capacity of edges incident to s is 3. Next we show that any 2–splittable flow with a value greater than 2 implies the satisfiability of the 3SAT–instance.

Assume to have two s,t–paths carrying in sum a flow value greater than 2. Then there has to be a path P_1 with flow value greater than 1. P_1 can only use edges with capacity 2. Such edges only occur in the blocking constructions of contrary literals. According to the graph structure, P_1 has to traverse all these blocking constructions. A second path P_2 has to be disjoint from P_1 because edge capacities never exceed 2 and both path values are assumed to be greater than 2 in sum. Thus, P_2 has to traverse all clause representations from Step 1. While traversing the clauses, P_2 never sends flow along paths representing contrary literals because P_1 blocks at least one of them. Referring to the 3SAT instance, set $x_i := 1$ if P_2 traverses an a_j, b_j–path representing x_i in one arbitrary clause C_j. Otherwise, set $x_i := 0$. Thus, every variable is set to 0 or 1 and every clause has to contain one true literal. We have described a satisfying assignment for the 3SAT–instance. The paths being disjoint we could have sent 2 flow units along P_1 and one unit along P_2.

As well, every satisfiable 3SAT instance implies a maximum 2–splittable flow of value 3: Choose one satisfied literal for each clause in a satisfying assignment. Route one flow unit along a path P_2 traversing the clause representations, always along one fulfilled literal. Send 2 flow units through the blocking constructions using an s,t–path P_1. This is possible because P_2 never traverses contrary literals simultaneously. We get a 2–splittable flow of value 3.

Thus, 3SAT can be reduced to M2SF such that a 3SAT–instance is satisfiable if and only if a maximum 2–splittable flow has value 3 and not satisfiable if and only if the maximum value is 2. □

Corollary 2 MkSF, *$k \geq 2$ constant, cannot be approximated with a performance guarantee better than* $5/6$, *unless* $P = NP$.

PROOF. We show this bound by a reduction from 3SAT to MkSF. Given k, write k as $2q + r$ with $q \in \mathbb{N}$ odd and $r \in \{0, 1, 2, 3\}$. Consider an instance of 3SAT. Construct a graph G by using q times the graph constructed in the proof of Theorem 1 (see Fig. 3) and linking all these identical graphs by common nodes s and t. Moreover, add r edges from s to t with capacity 1. We get a graph polynomial in the input size of the instance, which is sketched in Fig. 4. Note, that a maximum s, t–flow has a value of $3q+r$ independent from the satisfiability of the 3SAT instance and therefore this is an upper bound for the value of any k–splittable flow.

Solve MkSF in the constructed graph. If the 3SAT instance is satisfiable, then MkSF results in a flow of value 3 along two paths in each of the q subgraphs. On the r additional edges we get a flow of value r on r paths. Thus, MkSF gives a total flow of value $3q + r$ on $2q + r = k$ paths.

If the 3SAT instance is not satisfiable, then in each of the q subgraphs we can use $0, 1, 2$ or 3 and more paths to send flow of value at most $0, 2, 2$ or 3, respectively. With the aim to maximize the flow on a limited number of paths we will not use two or more than three paths in any subgraph. Consider a solution of MkSF in G. Let a_1 be the number of subgraphs with a flow of value 2 on one path, a_2 be the number of subgraphs with a flow of value 3 on three paths and a_3 be the number of s, t–edges carrying one unit of flow. The following integer program gives the maximum value of a k–splittable s, t–flow:

$$\begin{aligned} \max\ & 2a_1 + 3a_2 + a_3 \\ s.t.\quad & a_1 + a_2 \leq q && (1) \\ & a_3 \leq r && (2) \\ & a_1 + 3a_2 + a_3 \leq 2q + r && (3) \\ & a_1, a_2, a_3 \in \mathbb{N} && (4) \end{aligned}$$

The following setting gives a feasible solution $a_1 = (q + 1)/2$, $a_2 = (q - 1)/2$, $a_3 = r$ with the value $2a_1 + 3a_2 + a_3 = (5q - 1)/2 + r$. Since q is odd, the optimal value is bounded by the same value. To show this, we take a linear combination of the inequalities (1), (2), (3) with coefficients 3/2, 1/2 and 1/2, respectively, and get:

$$2a_1 + 3a_2 + a_3 \leq 5q/2 + r \quad \Rightarrow \quad 2a_1 + 3a_2 + a_3 \leq (5q-1)/2 + r.$$

Thus, MkSF gets the optimal value $(5q-1)/2 + r$ if the instance is unsatisfiable. We cannot approximate better than with a guarantee of 5/6 because:

$$\frac{\frac{5q-1}{2} + r}{3q + r} = \frac{5}{6} - \frac{1 - \frac{r}{3}}{6q + 2r} \leq \frac{5}{6} \quad \forall q \in \mathbb{N}, r \in \{0, 1, 2, 3\}.$$

□

Remark 3 *By a more detailed consideration of the last estimation in the proof of Corollary 2 we get that MkSF with constant $k \geq 2$ cannot be approximated better than with*

$$\begin{cases} \frac{5}{6} - \frac{1}{6(2l-1)} & \text{for } k = 4l-2, l \in \mathbb{N}_{\geq 1} \\ \frac{5}{6} - \frac{1}{6(3l-1)} & \text{for } k = 4l-1, l \in \mathbb{N}_{\geq 1} \\ \frac{5}{6} - \frac{1}{6(6l-1)} & \text{for } k = 4l, l \in \mathbb{N}_{\geq 1} \\ \frac{5}{6} & \text{for } k = 4l+1, l \in \mathbb{N}_{\geq 1} \end{cases}$$

unless $P = NP$.

3 Number of Paths Depending on Network Parameters

We consider two kinds of problems. In the first one, we assume that the number of paths k is a function $k(m,n)$ on the number of edges m and nodes n. That means, that for a given instance an algorithm knows k. In the second one, we allow that an algorithm can first choose k in a certain interval and then computes a maximal splittable s,t–flow for this chosen k. The interval also depends on n and m.

Note, that in both cases k and the interval are not seen as a part of the input, but as a property of the problem. Thus, for different functions $k(m,n)$ or intervals we consider different problems. For the first type of problems some functions cause polynomial solvability. We write $k(m)$ to emphasize that k depends on m only.

3.1 Polynomially Solvable Cases

Theorem 4 MkSF *with $k(m,n) \geq m - n + 2$ is polynomially solvable.*

PROOF. We show that any maximum s,t–flow f in G can be decomposed into at most $m - n + 2$ paths and cycles in polynomial time. Consider an orientation of the edges of G such that f is still a feasible flow and add an edge (t, s) of infinite capacity to obtain a directed graph G'. Setting the flow on the edge (t, s) to the value of f results in a circulation f' in G'. Each decomposition of f' in cycles easily yields a decomposition of f in paths and cycles with the same number of elements.

We compute a decomposition of f' with the standard decomposition algorithm of Fulkerson: Start with a flow carrying edge and go through G' only along edges with a positive amount of flow until a cycle is closed. Assign the maximal possible flow value to this cycle with respect to f' and reduce f' by the cycle flow. Repeat the procedure until $f' = 0$. Since in any iteration the flow on at least one edge is set to 0 the incidence vectors of these cycles are linearly independent. Furthermore, the cycle space of G' has a dimension of $m + 1 - n + 1 = m - n + 2$ such that the computed decomposition of f' contains at most $m - n + 2$ cycles. □

Corollary 5 MkSF *with* $k(m) = m - 1$ *is polynomially solvable.*

PROOF. If $n \geq 3$, this is implied by the previous lemma. If $n = 2$, the graph consists of two nodes and m parallel edges. Sending flow along the $m-1$ edges with highest capacity gives an optimal solution to MkSF which can obviously be found in polynomial time. □

3.1.1 Simple Graphs

Theorem 6 *On simple graphs,* MkSF *can be solved in polynomial time for* $k(m) = m - c$*, where* $c \in \mathbb{N}_{\geq 2}$ *is an arbitrary constant.*

PROOF. Theorem 4 shows that MkSF is polynomially solvable in the case that $m - c \geq m - n + 2$ (resp. $n \geq c + 2$). Therefore, for all graphs fulfilling $n \geq c + 2$ the assertion holds. Assume to work on a graph with $n < c + 2$. Consider an instance of MkSF. If we specify which paths are used in a solution (P_1, ..., P_k), then it takes polynomial time to assign optimal flow values to the paths by solving the linear program:

$$\begin{aligned} \max \quad & f_1 + f_2 + ... + f_k \\ s.t. \quad & \sum_{i \in \{1,...,k\}: e \in P_i} f_i \leq u_e \; \forall e \in E \\ & \qquad f_i \geq 0 \qquad\quad \forall i \in \{1, ..., k\} \; . \end{aligned}$$

Having a constantly bounded number of nodes clearly also the number of edges and the number of paths are bounded by a constant and with them the number of possibilities to choose $m-c$ paths. Requiring more precise bounds later, we evaluate the number of possibilities to choose $m-c$ paths in a simple graph G with less than $c+2$ nodes more in detail. Such a bound effects that the problem MkSF can be solved in polynomial time.

Paths can contain nodes more than once. Obviously, there is always a solution of MkSF on simple paths without node repetition. So we can restrict our considerations to simple paths. We want to calculate the number of different simple s,t–paths in a simple graph with at most $c+1$ nodes. This number is bounded from above by the number of simple s,t–paths in a complete graph with $c+1$ nodes. It holds that a complete graph has $(c-1)(c-2)...(c-i+1) = (c-1)!/(c-i)!$ different simple s,t–paths using i edges (1 for $i=1$) and thus in total not more than $c!$ different paths:

$$\sum_{i=1}^{c}(c-1)!/(c-i)! = (c-1)!\sum_{i=0}^{c-1} 1/i! \leq c!.$$

A simple undirected graph with at most $c+1$ nodes fulfills $m \leq c(c+1)/2$. Thus, $m-c \leq c(c-1)/2$. There are no more than $\binom{c!}{m-c} \leq \binom{c!}{c(c-1)/2}$ possibilities to choose $m-c$ simple s,t–paths in G. The inequality holds because $m-c \leq c(c-1)/2 \leq c!/2$ for all $c \in \mathbb{N}$ and the binomial coefficients increase monotonically in that range. Thus, the number of simple paths in G is bounded by a constant, which implies that MkSF can be solved in polynomial time for $k = m-c$.

The argument also holds for directed graphs with little modifications concerning the bounds. Our arguments consider exactly $m-c$ different paths. If an optimal solution of MkSF contains less than $m-c$ different paths we would choose $m-c$ paths and the solution of the linear program would assign the value 0 to some f_i. □

Theorem 7 *On simple graphs,* MkSF *can be solved in polynomial time for all* $k(m,n) \geq m - (\log p(m,n))^{\epsilon}$*, where* p *is a polynomial in* m *and* n *and* $\epsilon \in (0, 1/3)$.

PROOF. We refer to Theorem 6 and describe how to bound the possibilities to choose $k = m-c$ simple s,t–paths. Notice, that c is not a constant here. For this number it holds:

$$\binom{c!}{\frac{c(c-1)}{2}} \leq c!^{c(c-1)/2} \leq c!^{c^2} \leq (c^c)^{c^2} = \left((2^{\log c})^c\right)^{c^2} = 2^{\log c\ c^3} \quad \forall c \in \mathbb{N}.$$

If $2^{\log c\ c^3}$ could be bounded from above by a polynomial p in m and n, then we only had to check a polynomial number of path combinations to solve MkSF.

Thus, MkSF would be polynomially solvable. Here we can, in fact, do this. Fix a polynomial $p(m,n)$ and an $\epsilon \in (0, 1/3)$. Let be $c \leq (\log p(m,n))^{\epsilon}$ and define $\delta := 1/\epsilon - 3 > 0$. We see the following:

$$\begin{aligned} & c && \leq (\log p(m,n))^{1/(3+\delta)} \\ \Rightarrow\ & c^{3+\delta} && \leq \log p(m,n) \\ \Rightarrow\ & c^{\delta} c^{3} && \leq \log p(m,n) \end{aligned}$$

There exists a c_δ such that for all $c > c_\delta$ we have $\log c \leq c^{\delta}$. Note that c_δ is a constant and so Theorem 6 gives polynomial solvability for these $k \geq m - c_\delta$. For $c > c_\delta$ it follows:

$$\begin{aligned} & (\log c)\ c^{3} && \leq \log p(m,n) \\ \Rightarrow\ & 2^{(\log c)\ c^{3}} && \leq p(m,n) \end{aligned}$$

Thus, MkSF, $k \geq m - (\log p(m,n))^{\epsilon}$, is polynomially solvable when p is a polynomial in m and n and $\epsilon \in (0, 1/3)$. □

3.2 NP–Hardness Proofs

We show that the problem MkSF is NP–hard for all functions $k(m,n)$ with $2 \leq k(m,n) \leq m - n + 1$. This is done in two steps. We prove the NP–hardness for MkSF where k can be chosen in the range $2 \leq k \leq m - m^{\epsilon}$ by a reduction from 3SAT and then for $m^{\epsilon} \leq k \leq m-n+1$ by a reduction from SubsetSum. In both cases $\epsilon \in (0, 1)$.

Theorem 8 *Let $\epsilon \in (0,1)$. The problem* MkSF *allowing an algorithm to choose k in the range $2 \leq k \leq m - m^{\epsilon}$ is strongly NP–hard and cannot be approximated with a guarantee better than $(m+3)/(m+4)$, unless $P = NP$.*

PROOF. Given $\epsilon \in (0, 1)$ we reduce 3SAT to MkSF where k can be chosen in the range $2 \leq k \leq m - m^{\epsilon}$. We extend the instance constructed in Theorem 1. For a 3SAT–instance let m_0 be the number of edges of this graph G_0 and set $m := \lceil m_0^{1/\epsilon} \rceil$. We add $m - m_0$ edges from s to t with capacity $\frac{1}{2}$ to G_0 and obtain a graph G with m edges. Note, that the size of G is polynomial in the 3SAT–instance because this holds for m_0 and thus also for m and G.

Let k be in the range $2 \leq k \leq m - m^{\epsilon}$ the number of paths chosen by the algorithm. For the number $m - m_0$ of additional s,t–edges we have:

$$2 \leq k \leq m - m^{\epsilon} \leq m - (m_0^{1/\epsilon})^{\epsilon} = m - m_0$$

Because of Theorem 1 the considered 3SAT instance is satisfiable if and only if a k–splittable flow in G has a maximum value of $3+\frac{1}{2}(k-2)=\frac{1}{2}(k+4)$ and is not satisfiable if and only if the maximum value is $3+\frac{1}{2}(k-3)=\frac{1}{2}(k+3)$.

So this MkSF–problem is strongly NP–hard (because of the NP–hardness of 3SAT and the constantly bounded capacities in the reduction) and cannot be approximated with performance guarantee better than

$$\frac{k+3}{k+4} \leq \frac{m-m^{\epsilon}+3}{m-m^{\epsilon}+4} \leq \frac{m+3}{m+4},$$

unless $P=NP$. $\square$

Theorem 9 MkSF *with* $k(m)=m-2$ *is* NP*–hard.*

PROOF. We reduce Subset Sum to MkSF with $k=m-2$. Consider a Subset Sum decision problem: Given q positive integers $u_1,...,u_q$ and a number M, is there a subset $S \subseteq \{1,...,q\}$ such that $\sum_{i\in S} u_i = M$? This problem is known to be NP–complete. Figure 5 shows a transformation of this problem to a k–splittable flow problem on a graph with three nodes s, v, t. The nodes s and v are connected by q parallel edges with capacities $u_1,...,u_q$. The nodes v and t are connected by two parallel edges with capacities M and $U-M$, where $U=u_1+...+u_q$.

Solve MkSF with $k=m-2=q$ for this special graph. The optimal value cannot exceed U. If a flow of value U is realized, then all $q+2$ edges have to be filled up to their capacity. If we compose such a flow using q paths, then every path has to consist of one s,v–edge and has to continue either via the edge with capacity M or with capacity $(U-M)$ and has to carry a flow of value u_i, one path for each u_i. This means that the flow of value M on the edge with capacity M is formed by path flows with value u_i for some values i. Thus, the number M is the sum of some u_i and the Subset Sum instance is satisfiable. On the other hand, if Subset Sum is satisfiable, then MkSF, $k=m-2$, results in a flow of value U. For all $i \in S$ we send a flow of value u_i along the s,v–edge with capacity u_i and continue on the edge with capacity M. For all $i \notin S$ we send a flow of value u_i along the s,v–edge with capacity u_i and continue on the edge with capacity $U-M$.

Thus, Subset Sum can be reduced polynomially to the MkSF problem with $k=m-2$. $\square$

Theorem 10 *Let* $\epsilon \in (0,1)$*. The problem* MkSF *allowing an algorithm to choose* k *in the range* $m^{\epsilon} \leq k \leq m-n+1$ *is* NP*–hard for every given* $n>2$*.*

PROOF. Again, we use a reduction from Subset Sum. Fix $\epsilon \in (0,1)$. We transform an instance of Subset Sum to an instance of MkSF where k can be chosen in the range $m^\epsilon \leq k \leq m-n+1$.

Given a Subset Sum instance construct a graph G' with $m' = q+2$ edges as in the proof of Theorem 9 (see Fig. 5) and add $n-3$ nodes $w_4, w_5, ..., w_n$ and $n-3$ edges $\{s, w_i\}$, $i \in \{4, ..., n\}$. Note, that these additional nodes and edges do not affect the argumentation given in the proof of Theorem 9, because they cannot appear in any s,t–path. To complete the construction add $\lceil q^{1/\epsilon} \rceil$ s,t–edges each with capacity 1/2 and get a graph G with m edges (see Fig. 6).

The hardness of SubsetSum gives insight for MkSF here, if $k \geq q$ and $k \leq q + \lceil q^{1/\epsilon} \rceil$. It holds:

$$k \geq m^\epsilon \geq (m' + \lceil q^{\frac{1}{\epsilon}} \rceil)^\epsilon \geq (q^{\frac{1}{\epsilon}})^\epsilon = q$$
$$k \leq m - n + 1 = (m' + n - 3 + \lceil q^{\frac{1}{\epsilon}} \rceil) - n + 1 = q + \lceil q^{\frac{1}{\epsilon}} \rceil$$

Solve MkSF in G. We show that Subset Sum is satisfiable if and only if MkSF has value $U + 1/2(k-q)$.

If Subset Sum is satisfiable, then we get a flow of value U using q paths in the smaller graph G' and additionally a value $1/2(k-q)$ on the remaining $k-q$ paths each formed by an s,t–edge with capacity 1/2. On the other hand, assume that MkSF has the optimal value $U+1/2(k-q)$ on k paths. According to the capacities of edges incident with s, k paths carry at most a flow of value $u_1, u_2, ..., u_q$ (first q paths) and $1/2, ..., 1/2$ (last $k-q$ paths) regarding that $u_i \in \mathbb{N}_{\geq 1}$. Thus, a k–splittable flow with value $u_1 + u_2 + ... + u_q + 1/2(k-q)$ is only possible if all q edges from s to v carry a flow of their capacity and if additionally $k-q$ edges from s to t transport each 1/2. It follows that these $k-q$ edges from s to t form $k-q$ paths from s to t and exactly q paths are left to transport U along s,v,t–paths. We have seen before that this implies a satisfiable Subset Sum instance. □

The next corollary follows essentially from Theorem 8 and Theorem 10 (choose $\epsilon = 1/3$).

Corollary 11 MkSF *with* $2 \leq k(m,n) \leq m-n+1$ *is NP–hard for* $n > 2$.

Remark 12 *Corollary 5 and Theorem 9 imply that* MkSF *is NP–hard for* $2 \leq k(m) \leq m-2$ *and polynomially solvable for other* k.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows. Theory, Algorithms, and Applications.* Prentice Hall, Englewood Cliffs, NJ, 1993.

[2] A. Bagchi, A. Chaudhary, P. Kolman, and C. Scheideler. Algorithms for faulttolerant routing in circuit-switched networks. In *Proceedings of the 14th Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 265–274. ACM Press, 2002.

[3] G. Baier. *Flows with Path Restrictions.* PhD thesis, TU Berlin, 2003.

[4] G. Baier, E. Köhler, and M. Skutella. On the k-splittable flow problem. *Algorithmica*, 42:231–248, 2005.

[5] L. R. Ford and D. R. Fulkerson. *Flows in Networks.* Princeton University Press, 1962.

[6] J. M. Kleinberg. *Approximation algorithms for disjoint paths problems.* PhD thesis, M.I.T., 1996.

[7] R. Koch, M. Skutella, and I. Spenke. Approximation and complexity of k-splittable flows. In T. Erlebach and P. Persiano, editors, *WAOA*, Lecture Notes in Computer Science, pages 244–257. Springer, 2005.

[8] P. Krysta, P. Sanders, and B. Vöcking. Scheduling and traffic allocation for tasks with bounded splittability. In *Proceedings of the 28th International Symposium on Mathematical Foundations of Computer Science*, volume 2747, pages 500–510. Springer, Berlin, 2003.

[9] M. Martens and M. Skutella. Flows on few paths: Algorithms and lower bounds. In S. Albers and T. Radzik, editors, *Algorithms — ESA '04*, volume 3221 of *Lecture Notes in Computer Science*, pages 520–531. Springer, Berlin, 2004.