Mathematical Programming manuscript No.
(will be inserted by the editor)

Martin Skutella

Approximating the single source unsplittable min-cost flow problem

Received: August 23, 2000 / Revised version: April 25, 2001

Abstract. In the single source unsplittable min-cost flow problem, commodities must be routed simultaneously from a common source vertex to certain destination vertices in a given graph with edge capacities and costs; the demand of each commodity must be routed along a single path so that the total flow through any edge is at most its capacity. Moreover, the total cost must not exceed a given budget. This problem has been introduced by Kleinberg [7] and generalizes several NP-complete problems from various areas in combinatorial optimization such as packing, partitioning, scheduling, load balancing, and virtual-circuit routing.

Kolliopoulos and Stein [9] and Dinitz, Garg, and Goemans [4] developed algorithms improving the first approximation results of Kleinberg for the problem of minimizing the violation of edge capacities and for other variants. However, known techniques do not seem to be capable of providing solutions without also violating the cost constraint. We give the first approximation results with hard cost constraints. Moreover, all our results dominate the best known bicriteria approximations. Finally, we provide results on the hardness of approximation for several variants of the problem.

Key words. approximation algorithm – multi-commodity flow – network flow – routing – unsplittable flow

1. Introduction

Problem definition, notation, and basics. An instance of the single source unsplittable flow problem is defined as follows. We are given a directed graph $G = (V, E)$ with edge capacities $u_e > 0$, $e \in E$, and a set of commodities $\{1, \ldots, k\}$ sharing a common source vertex $s \in V$. Moreover, together with each commodity i we are given a destination or sink $t_i \in V \setminus \{s\}$ and a demand value $d_i \in \mathbb{R}^+$; we say that the demand d_i is *located at vertex* t_i. The task is to route the demand of each commodity i *unsplittably*, i. e., on a single path, from the source s to its sink t_i such that the total flow through any edge e is at most its capacity u_e. In the version of the problem with costs, we are also given a cost function $c : E \to \mathbb{R}^+$ on the edges and the total cost of the unsplittable flow must not exceed a given budget $B \geqslant 0$.

Throughout the paper we use the following notation. A (*splittable*) flow on the graph G is a function $f : E \to \mathbb{R}^+$ satisfying the flow conservation constraints in each vertex $v \in V \setminus \{s, t_1, \ldots, t_k\}$ (i. e., the outflow is equal to the inflow at those vertices) and the source vertex s is the only vertex where the outflow may exceed the inflow. A flow is said to satisfy all demands, if the inflow minus the outflow at each vertex $v \in V \setminus \{s\}$ is equal to the sum of all demands located at v. A flow f is called *feasible* if it respects

M. Skutella: Technische Universität Berlin, Fachbereich Mathematik, MA 6–1, Straße des 17. Juni 136, D–10623 Berlin, Germany, e-mail: skutella@math.tu-berlin.de

Mathematics Subject Classification (1991): 90C27, 90B10, 90C35, 05C38, 05C85

the capacity constraints, i. e., if $f(e) \leqslant u_e$ for all $e \in E$; the cost $c(f)$ of flow f is given by $c(f) = \sum_{e\in E} f(e) \cdot c(e)$.

An unsplittable flow f is specified by a set of paths $\{P_1, \dots, P_k\}$, where P_i starts at the source s and ends at t_i, such that $f(e) = \sum_{i:\, e\in P_i} d_i$ for all edges $e \in E$. The cost $c(P_i)$ of an s-t_i-path P_i is defined as $c(P_i) = \sum_{e\in P_i} c(e)$ such that the cost of an unsplittable flow f given by paths $P_1, \dots, P_k$ can be written as $c(f) = \sum_{i=1}^{k} d_i \cdot c(P_i)$. Moreover, we set $d_{\max} := \max_{1\leqslant i\leqslant k} d_i$, $d_{\min} := \min_{1\leqslant i\leqslant k} d_i$, and $u_{\min} := \min_{e\in E} u_e$. Finally, for $a, b \in \mathbb{R}^+$ we write $a \mid b$ and say that b is *a-integral* if and only if $b \in a \cdot \mathbb{N}$.

We often refer to the following well-known results on splittable flows (see, e. g., [2]).

Theorem 1. *Let $G = (V, E)$ be a directed graph with capacities and costs on the edges. Moreover, there is a source vertex $s \in V$ and k sinks $t_1, \dots, t_k \in V$ with demands $d_1, \dots, d_k$.*

a) There exists a feasible (splittable) flow satisfying all demands if and only if, for any subset $T \subseteq V \setminus \{s\}$, the sum of capacities of edges in the directed cut $(V \setminus T, T)$ is at least $\sum_{i:\, t_i\in T} d_i$. We refer to the latter condition as cut condition.

b) If the cut condition is satisfied and all demands and capacities are a-integral for some $a \in \mathbb{R}^+$, then there exists a feasible (splittable) flow satisfying all demands with minimum cost such that the flow value on any edge is a-integral. Moreover, such a flow can be computed in polynomial time.

Complexity and optimization versions of the single source unsplittable flow problem. It is an easy observation that already the single source unsplittable flow problem without costs contains several well-known NP-complete problems as special cases, such as, for example, PARTITION, BIN PACKING, or even scheduling parallel machines with makespan objective [10]; we refer to [4,6,9] for more details and other special cases. If we consider the problem with costs, we obtain the KNAPSACK problem as a special case, see Figure 1. Moreover, an interesting special case of the generalized assignment problem considered by Shmoys and Tardos [12] can also be modeled as a single source unsplittable min-cost flow problem; we refer to [4,9] for a detailed discussion of the connection between the two problems for the case without costs.

Kleinberg [7] introduced the following optimization versions of the single source unsplittable flow problem:

Minimum congestion. Find the smallest value $\alpha \geqslant 1$ such that there exists an unsplittable flow that violates the capacity of any edge at most by a factor α.

Minimum number of rounds. Partition the set of commodities into a minimum number of subsets (rounds) and find a feasible unsplittable flow for each subset.

Maximum routable demand. Find a feasible unsplittable flow for a subset of demands maximizing the sum of demands in the subset.

For the more general setting with costs, we always add the requirement that the cost of the unsplittable flow is bounded by the given budget B (for the ‘minimum number of rounds’ problem, the collective cost of all rounds must not exceed B).

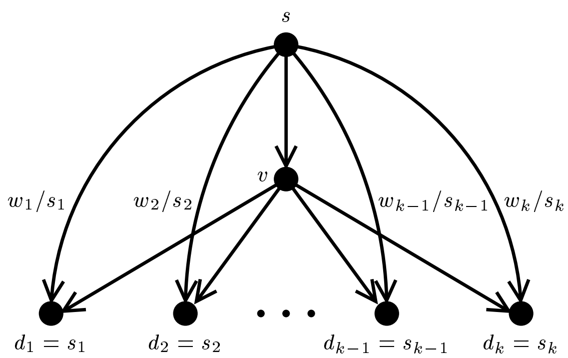

Fig. 1. Formulation of the KNAPSACK problem as a single source unsplittable min-cost flow problem. For each item, there is a corresponding commodity i whose demand d_i is equal to the size s_i of the i^{th} item; the cost of the direct edge from the source to the sink of commodity i is equal to the ratio w_i/s_i where w_i is the weight of the i^{th} item; all other edge costs are 0; the capacity of the edge from s to v is equal to the size of the knapsack; all other edges have infinite capacity. A feasible unsplittable flow satisfying all demands with minimum cost induces an optimal solution to the KNAPSACK problem and vice versa (items in the knapsack correspond to commodities routed via vertex v).

	congestion		number of rounds	routable demand
	$(d_{\max} \leqslant u_{\min})$	(arbitrary demands)	$(d_{\max} \leqslant u_{\min})$	$(d_{\max} \leqslant u_{\min})$
[7]	16	—	$O(1)$	$\Omega(1)^*$
[9]	$(3, 2)$ $(O(1/\varepsilon), 1+\varepsilon)$	$3 + 2\sqrt{2}$	13	$1/13^*$ $0.075 - \varepsilon$
[4]	2	5	5	0.226
this paper	$(3, 1)$	$(3 + 2\sqrt{2}, 1)$	$(8, 1)$	$(1/8, 1)^*$

Table 1. Summary of approximation bounds for variants of the single source unsplittable (min-cost) flow problem. Pairs of numbers (ρ, β) denote bicriteria approximations for the respective objective function and cost. All other results have been developed for the problem without costs. The results marked with a * have been developed for instances which satisfy the cut condition from Theorem 1 a).

Known results. Most results for the three optimization problems stated above were obtained under the assumption that all demands are at most as large as the minimum edge capacity, i. e., $d_{\max} \leqslant u_{\min}$, such that any commodity can be routed through any edge. Unless we emphasize in the following discussion that a result holds for the case of *arbitrary demands*, it always requires $d_{\max} \leqslant u_{\min}$. An account of the evolution of approximation results for the single source unsplittable (min-cost) flow problem is given in Table 1.

We first mention the results for the 'minimum congestion' problem. Kleinberg [7] gives a 16-approximation algorithm; for the corresponding problem on undirected graphs, he achieves performance guarantee $9/2 + \sqrt{14} \approx 8.25$. For the problem with costs on undirected graphs, he obtains a bicriteria $(6 + 2\sqrt{5}, 3 + 2\sqrt{5})$-approximation for congestion and cost. Kolliopoulos and Stein [9] (see also [8]) give a bicriteria approximation algorithm with performance guarantee 3 for congestion and 2 for cost; moreover, they can improve the performance ratio for cost to a constant arbitrarily close to 1 at the expense of an increase in the performance guarantee for congestion. For the case of arbitrary demands, they give a $(3 + 2\sqrt{2})$-approximation algorithm. Asano [3] generalizes the approach of [9] to the problem with k sources and gives a bicriteria $(k + 2, 2)$-approximation algorithm for congestion and cost. Finally, for the problem without costs, Dinitz, Garg, and Goemans [4] obtain an algorithm with performance guarantee 2; to be more precise, their basic result says that any splittable flow satisfying all demands can be turned into an unsplittable flow while increasing the total flow through any edge by less than the maximum demand. For the case of arbitrary demands, they give a 5-approximation algorithm for congestion.

For the 'minimum number of rounds' problem, the following results were previously known. Kleinberg [7] shows that there is a constant factor approximation for this problem. Kolliopoulos and Stein [9] give a 13-approximation algorithm. Dinitz, Garg, and Goemans [4] improved this result to performance ratio 5. Under the assumption that the cut condition is satisfied, they show how to route all commodities in 5 rounds; on the other hand, they give an example where 3 rounds are necessary.

Finally, we discuss the known results for the 'maximum routable demand' problem. Kleinberg [7] shows that when the cut condition is satisfied, then a constant fraction of the total demand can be routed unsplittably. Kolliopoulos and Stein [9] improve this result to a fraction of $1/13$. Moreover, they give a $(0.075 - \varepsilon)$-approximation algorithm for the general problem when the cut condition may be violated. For this problem, Dinitz, Garg, and Goemans [4] obtain a 0.226-approximation algorithm. They also give an instance for which the cut condition is satisfied but only a fraction 0.385 of the total demand can be routed unsplittably.

It follows from the work of Lenstra, Shmoys, and Tardos [10] that the 'minimum congestion' problem cannot be approximated with performance guarantee better than $3/2$, unless P=NP. It also follows that 2 is a lower bound on the approximability of the 'minimum number of rounds' problem. Finally, Kolliopoulos and Stein [9] show that for the unsplittable flow problem with two sources, it is NP-hard to obtain a ρ-approximation with $\rho < 2$ for congestion.

Contribution of this paper. As discussed above, all previously developed algorithms dealing with costs lead to bicriteria approximation results, i. e., they obtain a constant performance guarantee for congestion, number of rounds, or routed demand at the expense of an increase in cost. Moreover, in Figure 2 we give an example showing that the basic algorithm of Dinitz, Garg, and Goemans [4] (that was designed for the problem without costs) is not adapted for handling costs; the cost of the unsplittable flow computed by the algorithm can be arbitrarily large compared to the cost of the initial splittable flow.

We present approximation algorithms for all three versions of the problem introduced above without relaxing the cost constraints; in other words, we always achieve performance guarantee 1 for cost, see Table 1. Our results, as the results of Kolliopoulos and Stein [9] and Dinitz, Garg, and Goemans [4], hold for both directed and undirected graphs.

Our basic approach is closely related to the one taken by Kolliopoulos and Stein [9]. It is based on rounding demand values to integer multiples of each other; then, commodities are routed iteratively in non-decreasing order of demands by appropriately rounding edge capacities and making use of the integrality result in Theorem 1 b). The

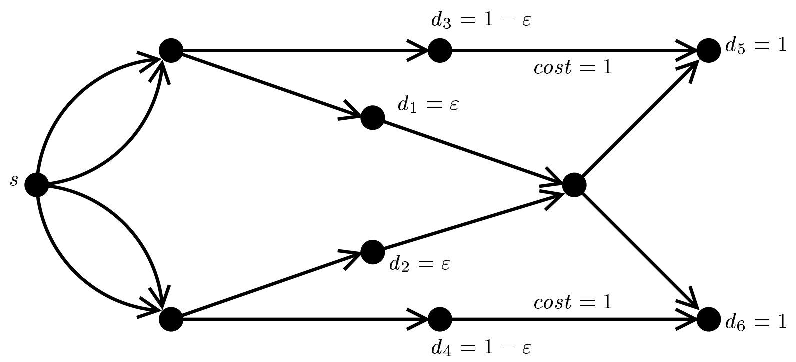

Fig. 2. An instance of the single source unsplittable min-cost flow problem with $d_{\max} \leqslant u_{\min}$; all capacities are 1 and the cost coefficients of all but the two labeled edges are 0. The cost of the unique feasible splittable flow is 2ϵ. However, the cost of any unsplittable flow found by the algorithm of Dinitz, Garg, and Goemans [4] is at least 1. The algorithm first routes commodities 1, 2, 3, and 4 along their (apart from the choice of the first edge) unique paths and decreases the flow value on the corresponding edges accordingly. In order to route the remaining commodities 5 and 6, the algorithm augments flow on cycles having a special property. The important observation is that, in the example, the only cycles with this property are the two 4-cycles with shapes of symmetric triangles in the figure. Moreover, the first augmentation always increases the flow on the edge with cost 1 in one of those cycles and the corresponding commodity is then routed across this edge. Thus, the cost of the unsplittable flow returned by the algorithm is at least 1. We refer to [4] for details.

critical part of this procedure is the rounding step. While Kolliopoulos and Stein loose a factor of 2 in cost, we apply a more sophisticated technique comprising the computation of most expensive paths from the source to the destination nodes or, alternatively, solving a max-cost flow problem.

We obtain the following results for problems with a given budget that must not be over-spent. There is a 3-approximation algorithm for congestion if $d_{\max} \leqslant u_{\min}$. For the case of arbitrary demands we obtain performance guarantee $3 + 2\sqrt{2}$. Moreover, we give improvements for several special cases and variants of the problem.

If the cut condition is satisfied, we show how to route all demands in 8 rounds such that the collective cost is bounded by the cost of a splittable min-cost flow satisfying all demands. In order to obtain this result, we use techniques similar to those developed by Dinitz, Garg, and Goemans [4] for the problem without costs. As a direct consequence of this result, we can route at least a fraction $1/8$ of the total demand unsplittably with cost bounded as above. It also leads to an 8-approximation algorithm for the general 'minimum number of rounds' problem (i. e., when the cut condition is not necessarily satisfied) with bounded collective cost.

Finally, we show that, unless P=NP, congestion cannot be approximated with performance guarantee better than $(-1+\sqrt{5})/2 \approx 1.618$ for the case of arbitrary demands. In the proof of this result we use a reduction from SAT which also yields a lower bound of $1/2$ on the approximability of the 'maximum routable demand' problem on directed graphs. As mentioned above, the best previously known lower bound for congestion is $3/2$ which follows from a reduction of the scheduling problem with makespan objective considered by Lenstra, Shmoys, and Tardos [10]. For this scheduling problem, it is a long standing open problem to close the gap between the lower bound $3/2$ and the currently best known 2-approximation algorithm. Also from this point of view, the non-approximability result presented in this paper might be of some interest.

Organization of the paper. In Section 2 we present the basic algorithm computing unsplittable flows for rounded demands. The rounding procedure for the case of arbitrary demands is discussed in Section 3. In Sections 4 and 5 we develop the approximation algorithms for minimizing congestion and number of rounds, respectively. Finally, in Section 6 we present the negative results on the existence of approximation algorithms.

2. The basic algorithm

In this section we consider the special case of the single source unsplittable flow problem where, for each pair of commodities i, j, the demands satisfy $d_i \mid d_j$ or $d_j \mid d_i$. We discuss a simple algorithm that turns an arbitrary (splittable) flow satisfying all demands into an unsplittable flow without increasing cost; moreover, we give a bound on the increase of the flow value on any edge. This result will turn out to be an important building block for deriving several approximation results for various problems and settings. In particular, in the next section, we will show how the algorithm can be extended to handle instances with arbitrary demand values.

Starting from a (splittable) flow f satisfying all demands, the algorithm constructs an unsplittable flow by considering the commodities i in non-decreasing order of demands, always routing the total demand d_i along a single path P_i from s to t_i in G. In general, even the commodities with minimum demand $d_{\min}$ cannot be routed unsplittably within the flow f since f might not be $d_{\min}$-integral. The idea is to carefully modify f such that all flow values are multiples of $d_{\min}$ and the increase of flow on any edge is bounded. Therefore, the problem is relaxed by setting the capacity of each edge to its current flow value rounded up to the nearest multiple of $d_{\min}$. Then, by Theorem 1 b), there exists a feasible $d_{\min}$-integral flow f' satisfying all demands and whose cost is bounded by the cost of f. All commodities i with demand $d_i = d_{\min}$ can be routed unsplittably within f': iteratively modify the flow f' by decreasing the flow values along an s-t_i-path P_i by $d_{\min}$ and remove commodity i from the instance; the existence of suitable paths P_i follows from Theorem 1. The whole procedure is then iterated until all commodities have been routed. Details are provided in Algorithm 1; it can obviously be implemented to run in polynomial time. We give a more detailed discussion of the running time at the end of this section.

Theorem 2. *Consider an instance of the single source unsplittable flow problem with demands satisfying $d_i \mid d_j$ or $d_j \mid d_i$ for each pair of commodities i, j. Given a (splittable) flow satisfying all demands, Algorithm 1 finds an unsplittable flow increasing the flow value on any edge by less than $d_{\max}$ and whose cost is bounded by the cost of the initial flow.*

Proof. The algorithm always implicitly maintains a flow satisfying all demands. At the end of iteration j, this *total flow* is given by the flow f_j plus the sum of flows of value d_i along paths P_i of commodities i that have already been routed. By construction of the

Algorithm 1:

Input : A directed graph $G = (V, E)$ with non-negative costs on the edges, a source vertex $s \in V$, k commodities $i = 1, \ldots, k$ with terminals $t_i \in V \setminus \{s\}$ and positive demands d_i such that $d_1 \mid d_2 \mid \cdots \mid d_k$, and a (splittable) flow f_0 on G satisfying all demands.

Output : An unsplittable flow given by a path P_i from s to each terminal t_i, $1 \leqslant i \leqslant k$.

```
i := 1; j := 0;
while i ⩽ k do
    j := j + 1; δ_j := d_i;
    for every edge e ∈ E, set its capacity u_e^j to f_{j−1}(e) rounded up to the nearest multiple of δ_j;
    compute a feasible δ_j-integral flow f_j satisfying all demands with c(f_j) ⩽ c(f_{j−1});
    remove all edges e with f_j(e) = 0 from G;
    while i ⩽ k and d_i = δ_j do
        determine an arbitrary path P_i from s to t_i in G;
        decrease f_j along P_i by d_i;
        remove all edges e with f_j(e) = 0 from G;
        i := i + 1;
return P_1, ..., P_k;
```

algorithm, the cost of the total flow never increases; in particular, the cost of the final unsplittable flow is bounded by the cost of the initial flow.

The total flow on edge e at the end of the j^{th} iteration of Algorithm 1 is given by the current flow value $f_j(e)$ plus the demands d_i of commodities i that have already been routed across edge e. The total flow is thus bounded by

$$f_j(e) + \sum_{i:\ \substack{d_i \leqslant \delta_j \\ e \in P_i}} d_i \;=\; f_j(e) + \sum_{i:\ \substack{d_i = \delta_j \\ e \in P_i}} d_i + \sum_{i:\ \substack{d_i < \delta_j \\ e \in P_i}} d_i \;\leqslant\; u_e^j \;+ \sum_{i:\ \substack{d_i < \delta_j \\ e \in P_i}} d_i \ .$$

By construction, the capacity u_e^j is obtained by rounding up $f_{j-1}(e)$ to the nearest multiple of δ_j. Since for $j > 1$ the flow f_{j-1} is δ_{j-1}-integral and $\delta_{j-1} \mid \delta_j$, we get

$$u_e^j \;\leqslant\; f_{j-1}(e) + \delta_j - \delta_{j-1} \ .$$

This bound also holds for $j = 1$ if we set $\delta_0 := \min(\{d_{\min}\} \cup \{f_0(e) \mod d_{\min} \mid e \in E,\ f_0(e) \text{ not } d_{\min}\text{-integral}\}) > 0$. Thus,

$$f_j(e) + \sum_{i:\ \substack{d_i \leqslant \delta_j \\ e \in P_i}} d_i \;\leqslant\; f_{j-1}(e) + \delta_j - \delta_{j-1} + \sum_{i:\ \substack{d_i < \delta_j \\ e \in P_i}} d_i \ .$$

Applying this inequality iteratively, we get that the total flow on edge e after iteration j is bounded by $f_0(e) + \delta_j - \delta_0$. In particular, after the last iteration the flow value is at most $f_0(e) + d_{\max} - \delta_0$. □

The result in Theorem 2 is tight in the following sense. Dinitz, Garg, and Goemans [4, remark after Theorem 3.7] give a class of instances together with a splittable flow satisfying all demands such that, in order to get an unsplittable flow, one has to increase the flow on one edge by an amount arbitrarily close to $d_{\max}$. Although those instances do not satisfy our assumption on the demand values ($d_i \mid d_j$ or $d_j \mid d_i$ for every pair of commodities i, j), they can easily be modified to fulfill this requirement by decomposing commodities into sub-commodities whose demand values sum up to the demands of the original commodities.

In the following corollary we state a slightly stronger version of Theorem 2 that has also been obtained by Dinitz, Garg, and Goemans [4, Theorem 3.7] for their algorithm.

Corollary 1. *In the unsplittable flow returned by Algorithm 1, the sum of all but one demand routed across any edge e is less than the initial flow on e.*

Proof. Let $d_i = \delta_j$ be the maximal demand routed across edge e. Then, the flow value on edge e is bounded by the total flow on e after the j^{th} iteration of the algorithm. It is stated in the proof of Theorem 2 that the latter amount is bounded by $f_0(e) + \delta_j - \delta_0$. □

Since all demands are multiples of $d_{\min}$, any unsplittable flow is $d_{\min}$-integral. Thus, it is reasonable to assume that the same is true for the capacities of all edges such that there exists a feasible $d_{\min}$-integral (splittable) min-cost flow satisfying all demands. For this case we can prove a slightly stronger result.

Corollary 2. *Given a feasible $d_{\min}$-integral (splittable) flow satisfying all demands, Algorithm 1 finds an unsplittable flow that violates the capacity of any edge by at most $d_{\max} - d_{\min}$ and whose cost is bounded by the cost of the initial flow.*

Proof. The result follows from the proof of Theorem 2; notice that δ_0 is equal to $d_{\min}$ in this case. □

For the case that $d_i = d_{\min} \cdot 2^{q_i}$, $q_i \in \mathbb{N}_0$, for every commodity i, a variant of Algorithm 1 and the results stated above have also been obtained by Kolliopoulos and Stein [9].

Running times. The bottleneck for the running time of Algorithm 1 is the computation of the δ_j-integral flow f_j in each iteration j. Given the flow f_{j-1}, this can be done in the following way (see, e. g., [2]). We consider the subgraph of the current graph G that is induced by all edges e whose flow value $f_{j-1}(e)$ is not δ_j-integral. Starting at an arbitrary vertex of this subgraph and ignoring directions of edges, we greedily determine a cycle C; this is possible since, due to flow conservation, the degree of every vertex is at least two. Then, we augment flow on C until the flow value on one of the edges becomes δ_j-integral; the orientation of the augmentation on C is chosen such that the cost of the flow is not increased. We delete all δ_j-integral edges and continue iteratively. This process terminates after at most m iterations and has thus running time $O(nm)$. Since the number of iterations of Algorithm 1 is bounded by k, its total running time is $O(knm)$.

We now discuss the special case when the ratios of all pairs of demands are powers of 2, i. e., $d_i = d_{\min} \cdot 2^{q_i}$, $q_i \in \mathbb{N}_0$, for $1 \leqslant i \leqslant k$. We can modify Algorithm 1 in the following way. Instead of only traversing the demand values d_i, the variable δ_j now adopts all values $d_{\min} \cdot 2^i$ between $d_{\min}$ and $d_{\max}$; details can be found in Algorithm 2. While, in doing so, the number of iterations is increased, the running time of an iteration can be decreased since half-integral flows can easier be turned into an integral flow than arbitrary fractional flows.

Algorithm 2:

Input : A directed graph $G = (V, E)$ with non-negative costs on the edges, a source vertex $s \in V$, k commodities $i = 1, \dots, k$ with terminals $t_i \in V \setminus \{s\}$ and positive demands $d_i = d_{\min} \cdot 2^{q}_{i}$, $q_i \in \mathbb{N}_0$, $q_1 \leqslant q_2 \leqslant \cdots \leqslant q_k$, and a (splittable) flow f_0 on G satisfying all demands.

Output : An unsplittable flow given by a path P_i from s to each terminal t_i, $1 \leqslant i \leqslant k$.

$i := 1; j := 0;$
while $d_{\min} \cdot 2^j \leqslant d_{\max}$ **do**
  $j := j + 1; \delta_j := d_{\min} \cdot 2^{j-1};$
  for every edge $e \in E$, set its capacity u_e^j to $f_{j-1}(e)$ rounded up to the nearest multiple of δ_j;
  compute a feasible δ_j-integral flow f_j satisfying all demands with $c(f_j) \leqslant c(f_{j-1})$;
  remove all edges e with $f_j(e) = 0$ from G;
  while $i \leqslant k$ *and* $d_i = \delta_j$ **do**
    determine an arbitrary path P_i from s to t_i in G;
    decrease f_j along P_i by d_i;
    remove all edges e with $f_j(e) = 0$ from G;
    $i := i + 1;$

return $P_1, \dots, P_k$;

To be more precise, the number of iterations in this variant of the algorithm is $1 + \log(d_{\max}/d_{\min})$. The running time of the first iteration is $O(nm)$ as discussed above. However, since f_{j-1} is $(d_{\min} \cdot 2^{j-2})$-integral in each further iteration $j \geqslant 2$, the amount of augmented flow along a cycle C in the procedure described above is $d_{\min} \cdot 2^{j-2}$ and all edges of C can thus be removed after the augmentation. In particular, the computation of f_j from f_{j-1} takes only $O(m)$ time. Moreover, the path P_i can be determined in $O(n)$ time for each commodity i and the total running time of Algorithm 2 is $O\big(kn + m\log(d_{\max}/d_{\min}) + nm\big)$.

3. The general case

In this section we discuss an algorithm that, given a splittable flow f satisfying all demands, constructs an unsplittable flow for the case of arbitrary demands. The basic idea of the algorithm is to round down the demand values such that the rounded demands satisfy the condition from Theorem 2. Then, Algorithm 1 (or Algorithm 2) is called to compute paths $P_1, \dots, P_k$. Finally, the original demand of commodity i, $1 \leqslant i \leqslant k$, is routed across path P_i.

In the following description of the method, which we call Algorithm 3, we assume that after removing all edges with flow value 0, the resulting graph is acyclic; otherwise, we iteratively reduce flow along directed cycles, which can be done in $O(nm)$ time. Notice that the cost of the flow is not increased since all edge costs are non-negative.

In the first step of Algorithm 3 we round all demands d_i to

$$\bar{d}_i \ := \ d_{\min} \cdot 2^{\lfloor \log(d_i/d_{\min}) \rfloor} \quad .$$

Then, in a second step, we modify the flow f such that it only satisfies the rounded demands $\bar{d}_i$, $1 \leqslant i \leqslant k$. This part of the algorithm is crucial for deriving a good bound on the cost of the final unsplittable flow. We consider the commodities i one after another and iteratively reduce the flow f along most expensive s-t_i-paths within

f (ignoring or removing edges with flow value zero) until the inflow in node t_i has been decreased by $d_i - \bar{d}_i$. Since the underlying graph has no directed cycles, a most expensive s-t_i-path can be computed in polynomial time. Notice that the resulting flow $\bar{f}$ satisfies all rounded demands. Thus, Algorithm 1 (or Algorithm 2) can be used to turn $\bar{f}$ into an unsplittable flow for the rounded instance. We construct an unsplittable flow for the original instance by routing, for each commodity i, the total demand d_i (instead of only $\bar{d}_i$) along the path P_i returned by Algorithm 1 (or Algorithm 2).

Theorem 3. *Algorithm 3 finds an unsplittable flow whose cost is bounded by the cost of the initial flow f and the flow value on any edge e is less than $2f(e) + d_{\max}$. More precisely, the sum of all but one demand routed across any edge e is less than twice the initial flow value on e.*

Proof. We first show that the cost of the unsplittable flow is bounded by the cost of the initial flow. By Theorem 2, the cost of the unsplittable flow for the rounded instance is bounded by the cost of $\bar{f}$, i. e.,

$$\sum_{i=1}^{k} \bar{d}_i \cdot c(P_i) \leqslant c(\bar{f}) \ . \tag{1}$$

The flow $\bar{f}$ was obtained from f by decreasing flow along most expensive paths within f from the source s to the terminals t_i (those paths exist since the underlying graph contains no directed cycles). In particular, since a positive amount of flow remained on path P_i in flow $\bar{f}$, its cost $c(P_i)$ is a lower bound on the cost of each s-t_i-path on which flow has been decreased during the construction of $\bar{f}$. This yields

$$\sum_{i=1}^{k} (d_i - \bar{d}_i) \cdot c(P_i) \leqslant c(f) - c(\bar{f}) \ . \tag{2}$$

Since the cost of the final unsplittable flow is $\sum_{i=1}^{k} d_i \cdot c(P_i)$, the result follows by taking the sum of inequalities (1) and (2).

In order to prove the result on the flow values, we consider a fixed edge e; let i_0 be a commodity with maximal demand that is routed across edge e. Notice that $d_i < 2\bar{d}_i$ for all commodities i. Together with Corollary 1 this yields the following bound on the flow value for edge e in the final unsplittable flow:

$$\sum_{i:\, e \in P_i} d_i \leqslant d_{i_0} + 2 \sum_{i:\substack{e \in P_i \\ i \neq i_0}} \bar{d}_i \leqslant d_{i_0} + 2f(e) \ .$$

Notice that the first inequality can only be tight if i_0 is the only commodity routed across edge e; however, the second inequality cannot be tight in this case and the result follows. $\square$

Kolliopoulos and Stein [9] give an algorithm similar to Algorithm 3. The seemingly small but crucial difference is that in their rounding step, demands are rounded *up* to powers of 2; thus, in the worst case, cost is increased by a factor 2 in this step and only a bicriteria performance guarantee can be given for the resulting unsplittable flow. In contrast to this, Algorithm 3 takes care of the problem by carefully decreasing the given flow on most expensive paths, thereby rounding down demands to $d_{\min}$ times powers of 2.

Running times. We now analyze the running time of Algorithm 3. The procedure for obtaining $\bar{f}$ from f can be implemented to run in $O(m^2)$ time; in each iteration of the procedure, computing most expensive paths from s to all vertices in the current acyclic network takes $O(m)$ time, and the number of iterations can be bounded by $O(m)$. Thus, the running time of Algorithm 3 is $O(m^2)$ plus the running time of Algorithm 1 (or Algorithm 2). For certain cases, the first term can be improved using the following variant of Algorithm 3.

Instead of computing $\bar{f}$ by iteratively reducing flow along most expensive paths, we use an arbitrary min-cost flow algorithm. We set the capacity of each edge $e \in E$ to $f(e)$ and the demand of each commodity i to $d_i - \bar{d}_i$. For the resulting instance we compute a feasible max-cost flow $\hat{f}$ satisfying all demands and set $\bar{f} := f - \hat{f}$. Notice that $\bar{f}$ satisfies the rounded demands $\bar{d}_i$. Moreover, if we decompose the max-cost flow $\hat{f}$ into flows on paths from the source to the terminals t_i, then any such s-t_i-path is at least as expensive as a most expensive s-t_i-path within the remaining flow $\bar{f}$. Therefore, inequality (2) also holds for this alternative definition of $\bar{f}$ and the proof of Theorem 3 is still correct.

Thus, the term $O(m^2)$ in the running time of Algorithm 3 can be replaced by the running time of an arbitrary min-cost flow algorithm. The running times of the currently best known min-cost flow algorithms are $O\big(nm\log(n^2/m)\log(nC)\big)$ [5], $O\big(nm(\log\log U)\log(nC)\big)$ [1], and $O\big((m\log n)(m + n\log n)\big)$ [11], see also [2].

Extensions. If the capacities of edges are large compared to the maximum demand value, the bound obtained in Theorem 3 can be improved through a modification of Algorithm 3. The idea is to decrease the multiplier 2 of $f(e)$ at the cost of increasing the additive part $d_{\max}$ of the bound. Since the multiplier 2 is due to the lack of accuracy of the rounding step, we improve it to $\sqrt{2}$ by employing a more precise rounding. However, this necessitates a partition of the rounded problem into two subproblems which are then solved independently, each causing some additive congestion.

In the first step of the modified algorithm, we round all demands d_i to

$$\tilde{d}_i := \begin{cases} d_{\max} \cdot 2^{\lfloor 2\log(d_i/d_{\max})\rfloor/2} & \text{if } d_i < d_{\max}, \\ d_{\max}/\sqrt{2} & \text{if } d_i = d_{\max}. \end{cases}$$

We partition the commodities into two subsets C_0 and C_1 with

$$C_0 := \{i \mid \log(d_{\max}/\tilde{d}_i) \in \mathbb{N}\} \qquad \text{and} \qquad C_1 := \{i \mid 1/2 + \log(d_{\max}/\tilde{d}_i) \in \mathbb{N}\} \ .$$

As above, the given flow f is reduced along most expensive paths; the reduced flow is then decomposed into the sum of two flows f_0 and f_1 satisfying the rounded demands of commodities in C_0 and C_1, respectively. Finally, by calling Algorithm 1 (or Algorithm 2) for the instance defined by the commodities in C_0 with input f_0 and then for the instance defined by the commodities in C_1 with input f_1, we obtain paths P_i for all commodities i and route the total demand d_i along path P_i.

Theorem 4. *The variant of Algorithm 3 described above computes an unsplittable flow whose cost is bounded by the cost of the initial flow f and the flow value on any edge e is less than $\sqrt{2}f(e) + (1 + 1/\sqrt{2})d_{\max}$.*

Proof. The bound on the cost of the unsplittable flow follows from the same argument as in the proof of Theorem 3. For a fixed edge e, choose commodities $i_0 \in C_0$ and $i_1 \in C_1$ with maximal demand that are routed across e; if no such commodity exists, we set $d_{i_0} = 0$ and/or $d_{i_1} = 0$ for the following argument. Notice that $d_i \leqslant \sqrt{2}\tilde{d}_i$, for $1 \leqslant i \leqslant k$, and $d_{i_0} < d_{\max}/\sqrt{2}$. This yields for each edge $e \in E$

$$\begin{aligned}\sum_{i:\, e\in P_i} d_i \;&\leqslant\; d_{i_0} + \sqrt{2}\sum_{\substack{i\in C_0:\, e\in P_i\\ i\neq i_0}} \tilde{d}_i + d_{i_1} + \sqrt{2}\sum_{\substack{i\in C_1:\, e\in P_i\\ i\neq i_1}} \tilde{d}_i\\ &\leqslant\; d_{i_0} + d_{i_1} + \sqrt{2}\big(f_0(e) + f_1(e)\big)\\ &<\; (1/\sqrt{2} + 1)d_{\max} + \sqrt{2}f(e)\ \ .\end{aligned}$$

This completes the proof. □

Using the same technique as described above but rounding the demands to $d_{\max}$ times half-integral powers of 3, one obtains an unsplittable flow whose cost is bounded by the cost of the initial flow and the flow value on any edge e is less than $\sqrt{3}f(e) + (1 + 1/\sqrt{3})d_{\max}$.

4. Minimizing congestion with bounded cost

In this section we make use of Algorithms 1 and 3 and the results in Theorems 2 and 3 in order to obtain approximation results for the problem of minimizing congestion with bounded cost. The underlying ideas of these implications have been introduced by Kolliopoulos and Stein [9] and have also been used by Dinitz, Garg, and Goemans [4] for the problem without costs.

Under the assumption that $d_{\max} \leqslant u_{\min}$, an unsplittable flow whose cost is bounded by a given budget B and whose congestion is less than a factor 3 away from the optimal congestion for that budget can be obtained in the following way. First we determine the smallest value $\alpha \geqslant 1$ (e. g., by binary search or by solving an LP formulation of the problem) such that there exists a (splittable) flow f of cost at most B satisfying all demands and the flow value on any edge is bounded by α times its capacity. Notice that α is a lower bound on the optimal congestion for unsplittable flows with budget B. Using Algorithm 3, f can be turned into an unsplittable flow whose value on any edge is bounded by 3α times its capacity without increasing cost, see Theorem 3. Since α is a lower bound on the minimal congestion, this procedure is a 3-approximation algorithm. It follows from Theorem 2 that the use of Algorithm 1 (or Algorithm 2) improves the performance guarantee to 2 if the demand values satisfy $d_i \mid d_j$ or $d_j \mid d_i$ for each pair of commodities i, j.

Theorem 5. *If $d_{\max} \leqslant u_{\min}$, there is a 3-approximation algorithm for the problem of minimizing congestion for a given budget that must not be over-spent. If $d_i \mid d_j$ or $d_j \mid d_i$ for each pair of commodities i, j, then the performance guarantee can be improved to 2.*

Without making the assumption $d_{\max} \leqslant u_{\min}$, we can give an approximation algorithm with performance guarantee $3+2\sqrt{2}$ by mimicking the approach of Kolliopoulos and Stein [9] for the problem without costs. We only give a brief description of the algorithm and its performance guarantee; for more details we refer to [9].

In contrast to the discussion above, we now restrict to flows where commodity i is sent only on edges with capacity at least d_i; notice that a (splittable) flow with this property respecting the budget and with minimum $\alpha \geqslant 1$ such that the flow on each edge is bounded by α times its capacity can be obtained, for example, from a linear programming formulation. Then, we partition the set of commodities into subsets C_q, $q = 0, 1, 2, \ldots$, with

$$C_q := \{i \mid d_{\max} \cdot \beta^{-(q+1)} < d_i \leqslant d_{\max} \cdot \beta^{-q}\}$$

where $\beta = 1 + 1/\sqrt{2}$. In the following, we only consider non-empty subsets C_q; their number is bounded by k. Using flow decomposition, f can be decomposed into a sum of flows f_q, $q = 0, 1, 2, \ldots$, satisfying the demands of commodities in C_q; moreover, f_q uses only edges of capacity at least $d_{\max} \cdot \beta^{-(q+1)}$. Next, we use a variant of Algorithm 3 to turn the flows f_q into unsplittable flows. In the first step of the modified algorithm, all demands in C_q are rounded to $d_{\max} \cdot \beta^{-(q+1)}$. The resulting unsplittable flows f'_q then satisfy

$$f'_q(e) \; < \; \beta f_q(e) + d_{\max} \cdot \beta^{-q} \qquad \text{and} \qquad c(f'_q) \; \leqslant \; c(f_q) \ .$$

A short computation shows that the congestion of the sum of the unsplittable flows f'_q, $q = 0, 1, 2, \ldots$, is bounded by $\big(1 + 2\beta + 1/(\beta - 1)\big) \cdot \alpha = (3 + 2\sqrt{2}) \cdot \alpha$.

Theorem 6. *For the case of arbitrary demands, the algorithm described above computes an unsplittable flow whose cost is within the budget B and whose congestion is bounded by $3 + 2\sqrt{2}$ times the minimum possible congestion for the budget B.*

For the special case of the problem treated in Theorem 2, we can improve the performance guarantee to 3 by grouping together commodities of equal demand.

Corollary 3. *For the case of arbitrary demands satisfying $d_i \mid d_j$ or $d_j \mid d_i$ for each pair of commodities i, j, there exists an approximation algorithm with performance guarantee 3 for the problem of minimizing congestion with bounded cost.*

By using an appropriate rounding scheme, we can generalize this result as follows.

Corollary 4. *Let $r \geqslant \delta \geqslant 1$; if there is a $d \leqslant d_{\min}$ such that $d_i/d \in \bigcup_{p=0}^{\infty} [r^p, \delta \cdot r^p]$ for all $1 \leqslant i \leqslant k$, then there exists an approximation algorithm with performance guarantee $\delta\big(2 + 1/(r - 1)\big)$ for the problem of minimizing congestion with bounded cost.*

Proof. Round each demand down to the nearest $d \cdot r^p$, $p \in \mathbb{N}_0$, and use a similar procedure as described above. □

In this context, we can also prove results of the following flavor.

Corollary 5. *Under the same assumptions as in Corollary 4, if there exists a feasible (splittable) flow of cost B satisfying all demands such that each commodity i uses only edges of capacity at least d_i, then we can compute a flow of cost at most B that routes a fraction $1/\delta$ of the demand of each commodity unsplittably such that the capacity of any edge is violated by a factor of at most $2 + 1/(r-1)$.*

5. Minimizing the number of rounds and maximum routable demand

We consider the problem of routing all commodities unsplittably without violating edge capacities in a minimum number of rounds such that the collective cost over all rounds must not exceed a given budget B. We restrict to the case $d_{\max} \leqslant u_{\min}$ in this section.

Our main result is an 8-approximation algorithm for this problem that is based on the algorithms for minimizing congestion presented above. In order to obtain this result, we first develop results on how to route demands within a certain range in few rounds. This approach and some of the basic ideas used in the following lemmas have been developed by Dinitz, Garg, and Goemans [4] for the problem without costs.

To give a compact formulation of the following lemma, we use the convention that $1/0 = \infty$ and $1/\infty = 0$.

Lemma 1. *Let f be a feasible (splittable) flow of cost at most B satisfying all demands. Moreover, let $q \geqslant 3$ and $k \in \mathbb{N}_0 \cup \{\infty\}$. Then if*

$$d_{\max} \leqslant \left(1 + \frac{1}{q-2} \cdot \frac{2}{1+1/k}\right)^{-1} u_{\min}$$

and

$$d_{\min} \geqslant \frac{1}{q-2} \cdot \frac{1}{k+1} \cdot d_{\max} \ ,$$

then the total demand can be routed unsplittably in q rounds with collective cost at most B.

Proof. Construct a new instance of the unsplittable flow problem as follows: make q copies of the graph $G = (V, E)$ and let the cost coefficient of each of the q copies of edge $e \in E$ be the original cost coefficient $c(e)$. Introduce a super source S and one edge of cost 0 from S to each copy of s. Moreover, for each commodity i, $1 \leqslant i \leqslant k$, add a super sink T_i and an edge of cost 0 from each copy of t_i to T_i. In the new instance, the original demand d_i of commodity i must be routed from S to T_i.

A (splittable) flow f' for the new instance satisfying all demands and with cost bounded by B can be constructed by assigning the flow f/q to each of the q copies of G and defining the flow on the additional edges leaving the super source S or arriving at a super sink T_i accordingly.

We use Algorithm 3 to turn f' into an unsplittable flow whose cost is at most B. If $k < \infty$, we set $d_{\min}$ to the lower bound

$$d_{\min} = \ell := \frac{1}{q-2} \frac{1}{k+1} d_{\max}$$

in the algorithm such that after the first step of the algorithm the rounded demands $\bar{d}_i$ are multiples of ℓ. Notice that this variant of Algorithm 3 works properly and the result on the quality of the computed solution in Theorem 3 remains true. The resulting unsplittable flow consists of one path from S to T_i for each commodity i. It is an easy observation that each path P_i lies entirely in one of the copies of G such that the unsplittable flow in G' can be interpreted as q unsplittable flows in G or, equivalently, as an unsplittable routing of all demands in q rounds; moreover, the collective cost is at most B.

It remains to be shown that the flow in each round (i. e., in each copy of G) respects the capacity constraints. If $k = \infty$, the first inequality in the lemma yields

$$d_{\max} \leqslant \frac{q-2}{q} u_{\min} \leqslant \frac{q-2}{q} u_e \ ,$$

for each edge e. In this case, by Theorem 3, the total flow on any copy of e (i. e., in any round) in the final unsplittable flow is bounded by

$$2f'(e) + d_{\max} \leqslant \frac{2}{q} f(e) + \frac{q-2}{q} u_e \leqslant u_e \ .$$

Thus, in the following we can assume that k is finite and

$$u_e < \frac{q}{q-2} d_{\max} \ .$$

Remember that Algorithm 3 calls Algorithm 1 to compute an unsplittable flow for a rounded instance where all demands $\bar{d}_i$ are multiples of ℓ. It follows from Corollary 1 that the sum of all but one demand routed across any copy of edge e in the unsplittable flow returned by Algorithm 1 for the rounded instance is at most

$$f'(e) \leqslant \frac{u_e}{q} < \frac{1}{q-2} d_{\max} = (k+1)\ell \ .$$

By ℓ-integrality of this flow, the considered value is thus bounded by $k\ell$. Since $d_i < 2\bar{d}_i$, the flow value in the final unsplittable flow is therefore at most

$$2k\ell + d_{\max} = \big(\frac{1}{q-2}\frac{2k}{k+1} + 1\big) d_{\max} \leqslant u_{\min} \leqslant u_e \ .$$

This completes the proof. □

In contrast to the last lemma, the following lemma also contains results on demands that can be routed in 2 rounds. However, for $q > 3$ rounds, the results in Lemma 1 turn out to be stronger.

Lemma 2. *Let f be a feasible (splittable) flow of cost at most B satisfying all demands. Moreover, let $q \geqslant 2$ and $k \in \mathbb{N}_0$. If*

$$d_{\max} \leqslant \frac{1}{k+1} u_{\min} \qquad \text{and} \qquad d_{\min} \geqslant \frac{1}{q}\frac{k+2}{k+1} d_{\max} \ ,$$

then the total demand can be routed unsplittably in q rounds with collective cost at most B.

Proof. The proof is similar to the proof of Lemma 1. In particular we use the same construction of the new instance on the graph G' and call Algorithm 3 to turn f' into an unsplittable flow on G' whose cost is bounded by B. However, this time we modify Algorithm 3 in the following way: In the first step, all demands d_i, $1 \leqslant i \leqslant k$, are rounded down to

$$\bar{d}_i \;=\; \ell \;:=\; \frac{1}{q}\frac{k+2}{k+1}d_{\max} \qquad \text{such that} \qquad d_i \;\leqslant\; q\frac{k+1}{k+2}\bar{d}_i \;\; .$$

If $d_{\max} \leqslant u_e/(k+2)$, then the final unsplittable flow on any of the q copies of edge e is bounded by

$$q\frac{k+1}{k+2}f'(e) + d_{\max} \;=\; \frac{k+1}{k+2}f(e) + \frac{1}{k+2}u_e \;\leqslant\; u_e \;\; .$$

Otherwise, if $u_e < (k+2)d_{\max}$, it follows from Corollary 1 that the sum of all but one demand routed across any copy of edge e in the unsplittable flow returned by Algorithm 1 for the rounded instance is at most

$$f'(e) \;\leqslant\; \frac{u_e}{q} \;<\; \frac{k+2}{q}d_{\max} \;=\; (k+1)\ell \;\; .$$

By ℓ-integrality of this flow, the considered value is thus bounded by $k\ell$. Since $d_i < q(k+1)\bar{d}_i/(k+2)$, the flow value in the final unsplittable flow is therefore at most

$$q\frac{k+1}{k+2}k\ell + d_{\max} \;=\; (k+1)d_{\max} \;\leqslant\; u_{\min} \;\leqslant\; u_e$$

and the proof is complete. □

Finally, we give a result on arbitrarily small demand values that can be routed unsplittably in 2 rounds making use of Theorem 4.

Lemma 3. *Let f be a feasible (splittable) flow of cost at most B satisfying all demands. If $d_{\max} \leqslant (3-2\sqrt{2})u_{\min}$, then the total demand can be routed unsplittably in 2 rounds with collective cost at most B.*

Proof. As in the proofs of the two lemmas above, we construct a new instance on the graph G' (with $q = 2$). By Theorem 4, the flow f' on G' can be turned into an unsplittable flow such that the flow on both copies of an edge $e \in E$ is bounded by

$$\sqrt{2}f'(e) + (1+1/\sqrt{2})d_{\max} \;\leqslant\; \frac{1}{\sqrt{2}}u_e + (1+1/\sqrt{2})(3-2\sqrt{2})u_{\min} \;\leqslant\; u_e \;\; .$$

This completes the proof. □

An overview of the results from Lemmas 1 to 3 is given in Table 2. Combining two of them, we can prove the following result.

Theorem 7. *If $d_{\max} \leqslant u_{\min}$ and there exists a feasible flow f with cost B satisfying all demands, then the total demand can be routed unsplittably in 8 rounds with collective cost at most B.*

a; b	$q=2$	$q=3$	$q=4$	$q=5$	$q=6$	$q=7$
$k=0$	$1; 1$ *	$1; 2/3$ *	$\mathbf{1; 1/2}$	$1; 1/3$	$1; 1/4$	$1; 1/5$
$k=1$	$1/2; 3/4$ *	$1/2; 1/2$	$2/3; 1/4$	$3/4; 1/6$	$4/5; 1/8$	$5/6; 1/10$
$k=2$	$1/3; 2/3$ *	$3/7; 1/3$	$3/5; 1/6$	$9/13; 1/9$	$3/4; 1/12$	$15/19; 1/15$
$k=3$	$1/4; 5/8$ *	$2/5; 1/4$	$4/7; 1/8$	$2/3; 1/12$	$8/11; 1/16$	$10/13; 1/20$
$k=\infty$	$3-2\sqrt{2}; 0$ **	$1/3; 0$	$\mathbf{1/2; 0}$	$3/5; 0$	$2/3; 0$	$5/7; 0$

Table 2. Results on the number of rounds from Lemmas 1 to 3. Each pair of numbers (a; b) in column q means that all demands can be routed unsplittably in q rounds without violating the budget constraint if $d_{\max} \leqslant a \cdot u_{\min}$ and $d_{\min} \geqslant b \cdot d_{\max}$. The results marked with a * follow from Lemma 2; the result marked with ** follows from Lemma 3; all other results follow from Lemma 1. The two results in bold face yield Theorem 7.

Proof. The flow f can be decomposed into the sum of two flows $f_1 + f_2$ such that f_1 satisfies all demands in the range $(0, u_{\min}/2]$ and f_2 satisfies all demands in the range $(u_{\min}/2, u_{\min}]$. By Lemma 1 (see also Table 2), the demands in the first range can be routed unsplittably in 4 rounds with collective cost at most $c(f_1)$ and the demands in the second range can be routed unsplittably in 4 rounds with collective cost at most $c(f_2)$. Since $c(f_1) + c(f_2) = c(f) \leqslant B$, the result follows.

An alternative proof partitions the demands according to the ranges $(0, u_{\min}/3]$ (3 rounds) and $(u_{\min}/3, u_{\min}]$ (5 rounds), see Table 2. □

Using the technique from the proof of Lemma 1, one can also prove various results of the following flavor.

Corollary 6. *If $d_{\max} \leqslant u_{\min}$ and there exists a feasible flow f with cost B satisfying all demands, then half of the demand of each commodity can be routed unsplittably in 2 rounds with collective cost at most $B/2$.*

In a similar way as in Section 4 for the problem of minimizing congestion, Theorem 7 can be turned into an approximation result by first determining the smallest value $\alpha \geqslant 1$ such that there exists a flow of cost at most B satisfying all demands and the flow value on each edge is bounded by α times its capacity. Notice that the number $\lceil\alpha\rceil$ is a lower bound on the minimum number of rounds; on the other hand, using essentially the same techniques as described above, we can route all demands unsplittably in $8\lceil\alpha\rceil$ rounds. We refer to [4] for a more detailed discussion (for the problem without costs).

Theorem 8. *If $d_{\max} \leqslant u_{\min}$, there is an 8-approximation algorithm for the problem of routing all demands unsplittably in a minimum number of rounds such that the collective cost over all rounds is bounded by a given budget.*

If the cut condition is satisfied, Theorem 7 also implies the following result on the maximum routable demand.

Corollary 7. *If $d_{\max} \leqslant u_{\min}$ and there exists a feasible (splittable) flow of cost at most B satisfying all demands, then a fraction of at least $1/8$ of the total demand can be routed unsplittably with cost at most B.*

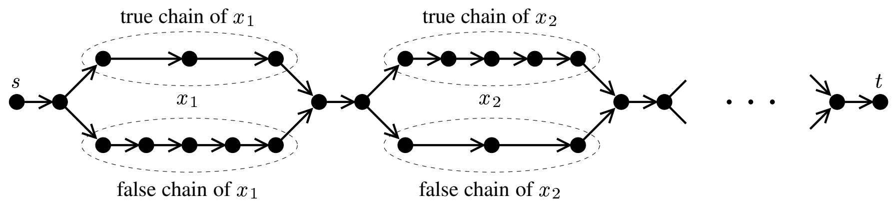

Fig. 3. The subgraph induced by all edges of capacity 1 in the reduction of SAT to the single source unsplittable flow problem. There is one forking for each variable x_i of the SAT instance. The reduction works for both directed and undirected graphs. For the latter case, ignore directions of edges in the figure.

6. Results on the hardness of approximation

The aim of this section is to prove negative results on the existence of approximation algorithms for several variants of the single source unsplittable flow problem. All results presented in this section already hold for the problem without costs on the edges.

Theorem 9. *For instances of the single source unsplittable flow problem on directed or undirected graphs, it is NP-hard to approximate congestion with performance guarantee strictly better than* $(1+\sqrt{5})/2 \approx 1.618$.

We prove this theorem by giving a reduction from the NP-complete SAT problem. Given an instance of SAT, we construct an instance of the single source unsplittable flow problem satisfying the following conditions. If the given SAT formula is satisfiable, then there is an unsplittable flow with congestion 1. However, if the formula is not satisfiable, then every unsplittable flow has congestion at least $(1+\sqrt{5})/2$.

Proof. Given an instance of SAT with variables $x_1, \dots, x_n$ and clauses $C_1, \dots, C_m$, we construct an instance of the single source unsplittable flow problem by introducing $m+1$ commodities $0, \dots, m$ with demands $d_0 = 1$ and $d_j = \delta := (-1+\sqrt{5})/2$, for $1 \leqslant j \leqslant m$. The edges of the (undirected) graph G have capacities 1 or δ. The subgraph of G induced by all edges of capacity 1 is depicted in Figure 3 (for the present, ignore directions of edges) with the source s on the left hand side and the common sink of all commodities t on the right hand side of the figure. There is one forking for each variable x_i, $1 \leqslant i \leqslant n$. Thus, each unsplittable routing of commodity 0 with congestion strictly less than $1/\delta = (1+\sqrt{5})/2$ uniquely determines a truth assignment to the variables $x_1, \dots, x_n$ and vice versa: x_i is set true (false) if commodity 0 is *not* routed across the true (false) chain corresponding to this variable. In other words, if x_i is set true (false), commodity 0 uses the whole capacity of x_i's false (true) chain and no other commodity can use an edge of this chain without raising congestion to at least $1+\delta = (1+\sqrt{5})/2$. The true (false) chain of x_i contains two consecutive edges for each clause C_j in which x_i occurs unnegated (negated). Thus, in the example given in Figure 3, the variable x_1 occurs in one clause unnegated and in two clauses negated. We label every second edge of the chain from left to right with the indices j of the corresponding clauses in decreasing order, see Figure 4 (again, for the present, ignore directions of edges).

We now describe the remaining part (edges of capacity δ) of the graph. The basic idea of the construction is that commodity j, $1 \leqslant j \leqslant m$, can be routed without raising

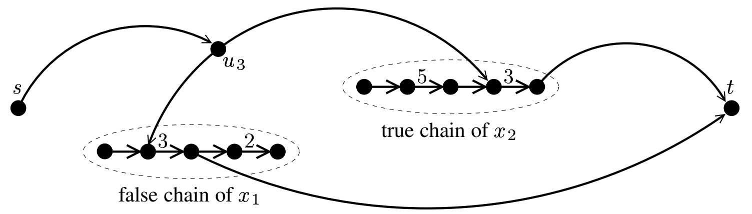

Fig. 4. The labeling of the edges in the chains from Figure 3: In our example, the variable x_1 occurs negated in clauses C_2 and C_3, while x_2 occurs unnegated in clauses C_3 and C_5. Clause C_3 is given by $(\neg x_1 \vee x_2)$.

congestion to at least $1 + \delta = (1 + \sqrt{5})/2$ if and only if the routing of commodity 0 corresponds to a truth assignment satisfying clause C_j. For each clause C_j, the graph contains one additional vertex u_j which is directly connected to the source s. Moreover, there is an edge between u_j and the left endpoint of any edge with label j in a true or false chain of some variable and another edge between the right endpoint of any such edge and the sink t, see Figure 4.

Since there are exactly m edges of capacity δ incident to s, we can assume without loss of generality that commodity j, $1 \leqslant j \leqslant m$, takes the edge from s to u_j. By construction, if the routing of commodity 0 corresponds to a truth assignment satisfying clause C_j, there is an edge labeled j which is not used by commodity 0 and commodity j can be routed from u_j to the left endpoint of this edge, across the edge, and then directly to the sink. On the other hand, if C_j is not satisfied by the truth assignment, all edges that can be reached from u_j carry commodity 0 and there is no path from u_j to the sink t with positive remaining capacity.

Summarizing, we have shown that all commodities can be routed unsplittably with congestion 1 if and only if there exists a satisfying truth assignment for the underlying instance of SAT. Otherwise, the congestion of any unsplittable flow is at least $(1 + \sqrt{5})/2$. Thus, a ρ-approximation algorithm with $\rho < (1 + \sqrt{5})/2$ would solve the NP-complete SAT problem. □

If the graph G is directed, we can show that Theorem 9 already holds for the case of only two commodities. Again, we give a reduction from SAT similar to the reduction presented above. Commodities $1, \dots, m$ are replaced by a single commodity 1 with demand δ and sink t. The definition of commodity 0 and the subgraph of all edges of capacity 1 remain unchanged; however, edges are now directed from left to right as shown in Figure 3. As discussed above, each unsplittable routing of commodity 0 with congestion strictly less than $1/\delta = (1 + \sqrt{5})/2$ uniquely determines a truth assignment to the variables $x_1, \dots, x_n$ and vice versa.

The remaining part of the directed graph is constructed such that commodity 1 can be routed unsplittably (without raising congestion to at least $1 + \delta = (1 + \sqrt{5})/2$) if and only if the routing of commodity 0 corresponds to a satisfying truth assignment. For each clause C_j, the graph contains two additional vertices u_j and v_j. We introduce a directed edge from v_j to u_{j+1} for $j = 0, \dots, m$, where $v_0 := s$ and $u_{m+1} := t$. Moreover, for each variable x_i occuring unnegated (negated) in C_j, there is a directed edge from u_j to the start vertex of the corresponding edge j in the true (false) chain of x_i (see Figure 4) and a directed edge from its end vertex to v_j.

Lemma 4. *The demands of the two commodities can be routed unsplittably without violating edge capacities if and only if there exists a satisfying truth assignment for the underlying instance of SAT. Otherwise, every unsplittable flow has congestion at least* $(1+\sqrt{5})/2$.

Proof. We first show how a satisfying truth assignment leads to an unsplittable flow with congestion 1. Route commodity 0 according to the given truth assignment as described above. Commodity 1 is routed across the edge connecting s to u_1 and then, for $j = 1, \ldots, m$, from u_j to u_{j+1} as follows. Since clause C_j is satisfied by the given truth assignment, there exists a variable i that either occurs unnegated and is set to true or occurs negated and is set to false. Commodity 1 is routed from u_j across the corresponding edge in the respective chain of variable x_i to v_j and then to u_{j+1}.

Conversely, we show that the routing of commodity 0 in an arbitrary unsplittable flow with congestion strictly less than $(1+\sqrt{5})/2$ corresponds to a satisfying truth assignment. Obviously, the routing of commodity 0 determines some truth assignment since otherwise congestion would be at least $1/\delta = (1+\sqrt{5})/2$. Let P be the s-t-path of commodity 1. If vertex u_j is contained in P, then clause C_j is satisfied for the following reason. After visiting u_j, commodity 1 must either use an edge in the true chain of a variable occuring unnegated in clause C_j or an edge in the false chain of a variable occuring negated in C_j. Since commodity 0 cannot be routed across the same edge, clause C_j is thus satisfied.

It remains to be shown that P contains u_j for all $j = 1, \ldots, m$. We claim that these vertices occur in P in order of increasing j. First notice that, due to the capacity of the only edge entering u_j (from v_{j-1}), each u_j occurs at most once in P. Obviously, u_1 is the first node after s in P. Assume that, for some fixed j, P visits the nodes $u_1, \ldots, u_j$ in this order. The next vertex after u_j in P is on the true or false chain of some variable x_i occuring in C_j. Moreover, P must leave this chain again arriving at some vertex v_h (otherwise, commodity 1 would get stuck at the vertex succeeding the true and the false chain of x_i in the subgraph depicted in Figure 3, since commodity 0 uses the only outgoing edge there). Since the edges in each chain are labeled from left to right in decreasing order, we get $h \leqslant j$. The only edge leaving v_h ends at u_{h+1}; since $u_1, \ldots, u_j$ have already been visited by P, we get $h = j$. The claim thus follows by induction which completes the proof. □

Corollary 8. *For the single source unsplittable flow problem on directed graphs, Theorem 9 already holds for instances with only two commodities.*

Using a variant of the reduction discussed before Lemma 4, we can prove the following result on the hardness of approximation for the maximum routable demand problem.

Theorem 10. *Unless P=NP, there is no approximation algorithm with performance guarantee* $\rho > 1/2$ *for the problem of routing the maximum possible demand in directed graphs.*

Proof. We employ the reduction from SAT with two commodities described above. However, we now choose $\delta = 1 - \varepsilon$ for an arbitrary $0 < \varepsilon < 1$. It follows from the proof of Lemma 4 that the two commodities can be routed unsplittably without violating edge capacities if and only if there exists a satisfying truth assignment for the underlying instance of SAT. Otherwise, at most a fraction of $1/(2-\varepsilon)$ of the total demand can be routed unsplittably. Since $\varepsilon > 0$ can be arbitrarily small, a ρ-approximation algorithm with $\rho > 1/2$ would decide the NP-complete SAT problem. □

As a direct consequence of the NP-completeness of the single source unsplittable flow problem, the 'minimum number of rounds' problem is NP-hard to approximate within a factor of $\rho < 2$. However, this statement is of little relevance since we can only show it to be tight for instances with optimal value 1. Thus, like for the edge-coloring problem, it could be the case that the minimum number of rounds problem can be approximated within an additive constant of 1.

7. Further remarks

The results presented in this paper are the best currently known for the single source unsplittable flow problem with costs. However, Dinitz, Garg, and Goemans [4] were able to give better bounds and performance guarantees for the problem without costs. The reason is that their basic algorithm turns a splittable flow into an unsplittable flow while increasing the flow value on any edge by less than $d_{\max}$. We achieve this result only for the case of rounded demands (Theorem 2) but have to endure an additional factor of 2 in the general case (Theorem 3). It is an interesting open problem to decide whether the unsplittable flow problem with costs allows the same strong results obtained for the problem without costs. Goemans (personal communication, January 2000) conjectures that the basic result of Dinitz, Garg, and Goemans [4] discussed above can be generalized to the problem with costs.

Acknowledgements. We wish to thank Michel Goemans for motivating this research by pointing out the conjecture mentioned in the concluding remarks. Thanks to Cliff Stein for sending us a copy of a preliminary version of [9]. We are grateful to two anonymous referees for many helpful comments.

References

1. Ahuja, R.K., Goldberg, A.V., Orlin, J.B., Tarjan, R.E. (1992): Finding minimum-cost flows by double scaling. Mathematical Programming **53**, 243–266
2. Ahuja, R.K., Magnanti, T.L., Orlin, J.B. (1993): Network Flows: Theory, Algorithms, and Applications. Prentice Hall, Englewood Cliffs, New Jersey
3. Asano, Y. (2000): Experimental evaluation of approximation algorithms for the minimum cost multiple-source unsplittable flow problem. Proceedings of the Workshop on Approximation and Randomized Algorithms in Communication Networks, Carleton Scientific Press, 111–122
4. Dinitz, Y., Garg, N., Goemans, M.X. (1999): On the single source unsplittable flow problem. Combinatorica **19**, 1–25
5. Goldberg, A.V., Tarjan, R.E. (1990): Solving minimum cost flow problems by successive approximation. Mathematics of Operations Research **15**, 430–466
6. Kleinberg, J.M. (1996): Approximation algorithms for disjoint paths problems. Ph.D. thesis, M.I.T.
7. Kleinberg, J.M. (1996): Single-source unsplittable flow. Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science, 68–77

8. Kolliopoulos, S.G. (1998): Exact and approximation algorithms for network flow and disjoint-path problems. Ph.D. thesis, Dartmouth College.
9. Kolliopoulos, S.G., Stein, C. (2001): Approximation algorithms for single-source unsplittable flow. SIAM Journal on Computing, to appear.
10. Lenstra, J.K., Shmoys, D.B., Tardos, E. (1990): Approximation algorithms for scheduling unrelated parallel machines. Mathematical Programming **46**, 259–271
11. Orlin, J.B. (1993): A faster strongly polynomial minimum cost flow algorithm. Operations Research **41**, 338–350
12. Shmoys, D.B., Tardos, E. (1993): An approximation algorithm for the generalized assignment problem. Mathematical Programming **62**, 461–474