Flows on Few Paths: Algorithms and Lower Bounds

Maren Martens, Martin Skutella
Universität Dortmund, Fachbereich Mathematik, Lehrstuhl V, Vogelpothsweg 87, 44227 Dortmund, Germany

The classical network flow theory allows decomposition of flow into several chunks of arbitrary sizes traveling through the network on different paths. In the first part of this article we consider the unsplittable flow problem where all flow traveling from a source to a destination must be sent on only one path. We prove a lower bound of $\Omega(\log m / \log\log m)$ on the performance of a general class of algorithms for minimizing congestion where m is the number of edges in a graph. These algorithms start with a solution for the classical multicommodity flow problem, compute a path decomposition, and select one of its paths for each commodity in order to obtain an unsplittable flow. Our result matches the best known upper bound for randomized rounding—an algorithm of this type introduced by Raghavan and Thompson. The k-splittable flow problem is a generalization of the unsplittable flow problem where the number of paths is bounded for each commodity. We study a new variant of this problem with additional constraints on the amount of flow being sent along each path. We present approximation results for two versions of this problem with the objective to minimize the congestion of the network. The key idea is to reduce the problem under consideration to an unsplittable flow problem while only losing a constant factor in the performance ratio. © 2006 Wiley Periodicals, Inc. NETWORKS, Vol. 48(2), 68–76 2006

Keywords: multicommodity flow; network flow; unsplittable flow; k-splittable flow; approximation algorithm; randomized rounding

1. INTRODUCTION

1.1. Problem Definition

In classical network flow theory, flow being sent from a source to a destination may be split into a large number of chunks of arbitrary sizes traveling on different paths through the network. This effect is undesired or even forbidden in many applications. For this reason we consider the unsplittable flow problem (UFP), which was introduced by Kleinberg [14]: given a network with capacities on the edges and several source-sink pairs (commodities) with associated demand values, route the demand of each commodity on exactly one path leading from its source to its sink without violating edge capacities. For the special case of unit capacities and unit demands, we get the edge-disjoint path problem, which is well-known to be NP-complete [11]. Kleinberg [14] introduced the following optimization versions of the unsplittable flow problem. *Minimum congestion:* find the smallest value $\alpha \geq 1$ 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. Here, we are mainly interested in the minimum congestion problem.

A natural generalization of the unsplittable flow problem is the k-splittable flow problem (k-SFP), which has recently been introduced by Baier et al. [5]. For each commodity, there is an upper bound on the number of paths that may be used to route its demand. Already for the single-commodity case, the k-SFP is NP-complete [5]. Of course, the optimization versions of the UFP discussed above naturally generalize to the k-SFP.

In this article we introduce a new variant of the k-splittable flow problem with additional upper bounds on the amount of flow being sent along each path. The motivation for these bounds comes from the following packing and routing problem: a commodity must be shipped using a given number of containers of given sizes. First, one has to make a decision on the fraction of the commodity packed into each container. Then, the containers must be routed through a network whose edges correspond, for example, to ships or trains. Each edge has a capacity bounding the total size or weight of containers that are being routed on it. Each container being used, must be routed along some path through the network. In particular, the size of the container induces an upper bound on the amount of flow being sent along the chosen path. It is therefore called *path capacity*.

Received March 2005; accepted May 2006
Correspondence to: M. Martens; e-mail: maren.martens@math.uni-dortmund.de
Contract grant sponsor: DFG Focus Program 1126 (Algorithmic Aspects of Large and Complex Networks); Contract grant number: SK 58/5-3
Contract grant sponsor: The German Research Foundation; Contract grant number: SFB 531
Contract grant sponsor: EU Thermatic Network APPOL II (Approximation and Online Algorithms); Contract grant number: IST-2001-30012
DOI 10.1002/net.20121
Published online in Wiley InterScience (www.interscience.wiley.com).

We consider two variants of the problem for which we provide an intuitive motivation:

1. When loading a ship, the *total weight* of all containers on the ship must not exceed the capacity of the ship. The weight of a container is determined by the actual amount of flow assigned to it. Thus, in the first variant with *weight capacities*, the capacity of an edge bounds the actual amount of flow being sent on paths containing this edge. This is the classical interpretation of edge capacities.
2. When loading a train, the *total size* of the containers is usually more important than their total weight. Therefore, in the model with *size capacities*, the capacity of an edge bounds the total path capacity of all paths being routed through that edge. Notice that these capacity constraints are more restrictive than the classical ones in the first model.

We are mainly interested in the corresponding NP-hard optimization problem to minimize the congestion. A precise and formal definition of the problem under consideration is given in Section 2.

From now on we use m to denote the number of edges and n to denote the number of nodes of the given graph.

1.2. Related Results from the Literature

The unsplittable flow problem has been well studied in the literature. Raghavan and Thompson [19, 20] introduce a randomized rounding technique that gives an $O(\log m/\log\log m)$-approximation algorithm for the problem of minimizing congestion for the UFP if the maximum demand is bounded from above by the minimum edge capacity (the *balance condition*). (Unless stated otherwise, the balance condition is always assumed to be met for the UFP.) For any instance of the problem of minimizing congestion for the UFP their technique solves the related (fractional) multicommodity flow problem first and then chooses exactly one of the flow paths from it for each commodity at random, such that each flow path is chosen with probability equal to its flow value divided by the demand of its commodity. Their procedure even yields a constant factor approximation, if either the ratio of the minimum edge capacity and the maximum demand or the congestion of an optimal fractional routing is at least $\Omega(\log m)$. For undirected graphs Leighton et al. [18] prove that the performance ratio of randomized rounding is in $\Omega(\log m/\log\log m)$. For the directed case they prove an integrality gap for the UFP of the same factor. Chuzhoy and Naor [9] show that the directed case of the UFP is $\Omega(\log\log m)$-hard to approximate unless $NP \subseteq DTIME(n^{O(\log\log\log n)})$. Before this result was found, only MaxSNP hardness for the UFP was known (see, e.g., Kleinberg [15]). For the special case that we have unit demands and unit edge capacities (the *edge-disjoint paths problem*) Andrews and Zhang [1] very recently proved that there is no $(\log\log m)^{1-\epsilon}$-approximation for the undirected congestion minimization problem, unless $NP \subseteq ZPTIME(n^{polylog\ n})$. ($ZPTIME(f(s))$ is the class of problems solvable by randomized algorithms that always return the correct answer and whose expected running time is in $O(f(s))$ where s is the input size of the respective problem.)

For the objective of maximizing the sum of satisfied demands while meeting all edge capacity constraints, Baveja and Srinivasan [6] give an approximation ratio of $O(\sqrt{m})$. Azar and Regev [2] present a strongly polynomial algorithm also with approximation ratio $O(\sqrt{m})$. Kolman and Scheideler [17] even give a strongly polynomial $O(\sqrt{m})$-approximation algorithm for the problem without the balance condition. On the other hand, Guruswami et al. [12] show that in the directed case there is no approximation algorithm with performance ratio better than $O(m^{1/2-\epsilon})$ for any $\epsilon > 0$, unless $P = NP$. To get better approximation results one has to incorporate additional graph parameters into the bound. Baveja and Srinivasan [6] develop a rounding technique to convert an arbitrary solution to an LP-relaxation into an unsplittable flow within a factor of $O(\sqrt{m})$ or $O(d)$ where d denotes the length of a longest path in the LP solution. Using a result of Kleinberg and Rubinfeld [13], showing that $d = O(\Delta^2\alpha^{-2}\log^3 n)$ for uniform capacity graphs with some expansion parameter α and maximum degree Δ, one can achieve a better bound. Kolman and Scheideler [17] use a new graph parameter, the "flow number" F, which provides information on the level of interconnectedness of a network, and improve the ratio further to $O(F)$ for undirected graphs; they show that $F = O(\Delta\alpha^{-1}\log n)$. Kolman and Scheideler also show that there is no better approximation in general, unless $P = NP$. Chekuri and Khanna [8] study the uniform capacity UFP for the case that the task is to satisfy as many requests as possible. Using results from [2] or [6] they find an $O(n^{2/3})$-approximation algorithm for the problem in undirected graphs, an $O(n^{4/5})$-approximation algorithm for the problem in directed graphs, and an $O(\sqrt{n}\log n)$-approximation algorithm for the problem in acyclic graphs. If all commodities share a common source vertex, the unsplittable flow problem gets considerably easier. Results for the single source unsplittable flow problem have, for example, been obtained in [10, 14, 16, 21].

As mentioned above the k-SFP has been introduced by Baier et al. [5]. They first consider a special form of the k-SFP in which the task is to find a feasible k-split routing that uses exactly k_i paths for commodity $i \in \{1, \ldots, K\}$ which all carry the same amount of flow. They call this problem the *uniform exactly-k-SFP*. Because this problem still contains the UFP as a special case, it is also NP-hard. But note that any instance of it can be solved by solving a special instance of the UFP on the same graph. Thus, it holds that any ρ-approximation algorithm for the problem of minimizing the congestion for the UFP provides a ρ-approximation algorithm for the problem of minimizing the congestion for this special k-SFP. To approximate an instance of the general k-SFP Baier et al. [5] use the fact that a uniform exactly-k-split routing of minimum congestion for any instance of the k-SFP is a k-split routing

that approximates the congestion of an optimal k-split routing for the same instance within a factor 2.

Bagchi et al. [4] introduce a problem that is quite similar to the k-SFP. They consider fault-tolerant routings in networks. To ensure connection for each commodity for up to $k-1$ edge failures in the network, they require $k \in \mathbb{N}$ edge disjoint flow paths per commodity. This problem is called the k-disjoint flow problem (k-DFP). Bagchi et al. [4] also consider the integral splittable flow problem (ISF). Bagchi [3] extends the considerations of [4] to the k-SFP. The aim of Bagchi et al. is always to maximize the sum of satisfied demands subject to meeting all edge capacity constraints. In the k-DFP a demand d for any request is to be satisfied by a flow on k disjoint paths, each path carrying d/k units of flow. In the ISF integral demands need to be satisfied by an arbitrary number of paths, where the flow value of any path has to be integral. In contrast to [5] and the present article, Bagchi does not admit different bounds on the numbers of paths for different commodities in the k-SFP. For all of the mentioned problems, Bagchi et al. introduce simple greedy algorithms in the style of greedy algorithms for the UFP given by Kolman and Scheideler [17]. With these algorithms they obtain approximation ratios of $O(k^3F\log(kF))$ for the k-DFP and $O(k^2F)$ for the k-SFP on the conditions, that they have unit capacities and the maximum demand is at most k times larger than the minimum edge capacity. Here, F is again the flow number of the considered instance of the corresponding problem. For the ISF, Bagchi et al. obtain an approximation ratio of $O(F)$ for any instance with uniform capacities in which the maximum demand is at most the capacity of the edges. The ISF has earlier been studied by Guruswami et al. [12], who obtain an approximation ratio of $O(\sqrt{m\Delta}\log^2 m)$ for it, where Δ is the maximum degree of a vertex in the considered graph.

1.3. Contribution of this Article

Randomized rounding has proved to be a very successful tool in the design of approximation algorithms for many NP-hard discrete optimization problems during the last decade. The general idea is as follows: formulate the problem as a 0/1-integer linear program, solve the linear programming relaxation, and randomly turn the resulting fractional solution into an integral solution by interpreting the fractional values as probabilities. This idea was originally introduced in 1987 by Raghavan and Thompson [20] for the edge-disjoint paths problem, which is a special case of the unsplittable flow problem. Although our general understanding of approximability has considerably improved since then, the $O(\log m/\log\log m)$-approximation for the minimum congestion version of the UFP is still best known. Even worse, it is not known whether it is NP-hard to obtain a constant factor approximation. It is shown by Leighton et al. [18] that for undirected graphs the performance ratio of randomized rounding is $\Omega(\log m/\log\log m)$ and for directed graphs there is an integrality gap of the same size. Notice that the integrality gap does not say anything about the gap between an optimal solution for an instance of the UFP and a solution obtained by randomized rounding.

In Section 3 we prove that the performance ratio of randomized rounding for the UFP is not better than $\Omega(\log m/\log\log m)$. This result even holds if all commodities share a single source and a single sink, which are connected by a chain of pairs of parallel edges. Our proof works for both cases, the directed and the undirected one, and more generally for all algorithms which start—like randomized rounding—with a solution to the classical multicommodity flow problem, compute a path decomposition, and select one of its paths for each commodity to obtain an unsplittable flow. Moreover, we show that starting with an optimal solution to the classical multicommodity flow problem, which uses a *minimal* number of paths, does not help to break the $\Omega(\log m/\log\log m)$ lower bound for randomized rounding.

In Section 2 we define the k-splittable flow problem with path capacities. For both variants of the problem discussed above, we prove that they have essentially the same approximability as the unsplittable flow problem. To be more precise, we show that any ρ-approximation algorithm for the UFP can be turned into a 2ρ-approximation for either variant of our problem. The underlying idea is to decompose the solution of the problem into two parts. First, a packing of flow into containers is obtained. Then, each container is routed along a path through the network. The latter step can be formulated as an unsplittable flow problem. In Sections 4 and 5, respectively, we present simple packing routines for both variants of the problem. We show that we do not lose more than a factor 2 with respect to congestion.

2. PROBLEM DEFINITION AND NOTATION

An instance of the *k-SFP with path capacities* consists of a digraph $G=(V,E)$ with edge capacities $c:E\to\mathbb{R}^+$ and a set $\mathcal{T}\subseteq V\times V$ of $K\in\mathbb{N}$ *requests* or *commodities*. The ith request is denoted by (s_i,t_i). Together with request i, we are given its nonnegative *demand* d_i and the number of paths k_i (containers) it can be routed on. The flow value on the jth path of commodity i must be bounded by the given *path capacity* $t^i_j\geq 0$. We always assume that the path capacities of every commodity suffice to route the entire demand, that is, $\sum_{j=1}^{k_i} t^i_j \geq d_i$, for all $i=1,\ldots,K$.

A feasible solution to the k-SFP with path capacities consists of s_i-t_i-paths $P^i_1,\ldots,P^i_{k_i}$ with corresponding nonnegative flow values $f^i_1,\ldots,f^i_{k_i}$ for each commodity i, such that the following requirements are met:

$$\sum_{j=1}^{k_i} f^i_j = d_i \qquad \text{for all } i\in\{1,\ldots,K\}, \tag{1}$$

that is, the whole demand of request i is routed. Moreover,

$$f^i_j \leq t^i_j \qquad \text{for all } i\in\{1,\ldots,K\} \text{ and } j\in\{1,\ldots,k_i\}, \tag{2}$$

that is, the path capacities are obeyed. Finally,

$$\sum_{i=1}^{K} \sum_{\substack{j=1,\ldots,k_i:\\ e\in P_j^i}} f_j^i \le c(e) \qquad \text{for all } e \in E, \tag{3}$$

or

$$\sum_{i=1}^{K} \sum_{\substack{j=1,\ldots,k_i:\\ f_j^i>0 \wedge e\in P_j^i}} t_j^i \le c(e) \qquad \text{for all } e \in E, \tag{4}$$

where (3) must hold if we consider the problem with *weight capacities*, and (4) must hold if we consider the problem with *size capacities*. That is, the corresponding inequalities ensure that no edge capacity constraint is violated. We do not require that the paths $P_1^i, \ldots, P_{k_i}^i$ are distinct, for $i \in \{1, \ldots, K\}$. Furthermore, we allow a path to have flow value 0. In other words, we may use less than k_i paths for commodity i.

In this article we consider constraints (1) and (2) as hard constraints, which must not be violated. Our objective is to violate constraint (3) or (4), respectively, only by a preferably small factor—the *congestion*—which will be defined in more detail later on.

In the following we use the notion *routing* for a pair $(\mathcal{R}, (f_P)_{P\in\mathcal{R}})$, where $\mathcal{R}$ is a set of paths in the considered graph and f_P is the flow value on path $P \in \mathcal{R}$. We will use the notion *k-split routing* for a routing of $\mathcal{T}$ that routes the whole demand of each commodity and uses at most k_i paths for commodity i. A *feasible k-split routing* shall be one that additionally satisfies all edge capacity constraints. Analogously we use the notion *(feasible) unsplit routing* for a routing of $\mathcal{T}$ that (obeys all edge capacities and) routes all demands using only one path per commodity.

Note, that the k-SFP with path capacities is strongly NP-hard, because it contains the unsplittable flow problem as a special case (set $k_i = 1$ and $t_1^i = d_i$, for all $i = 1, \ldots, K$). Even the single commodity version is strongly NP-hard because it is a generalization of the strongly NP-hard single source UFP [5]. Therefore, we are interested in approximate solutions to the problem. The *congestion* of a routing is the maximum factor by which the capacity of an edge in the graph is violated. Consider any k-split routing. For either variant of the k-SFP, we can simply define the congestion of this routing as the smallest value $\alpha \ge 1$ for which multiplying the right-hand side of (3) or (4), respectively, makes this inequality true. Our objective is always to minimize the congestion of the respective problem.

3. A LOWER BOUND FOR PATH-SELECTING ALGORITHMS

In this section we consider the unsplittable flow problem, that is, the special case $k_i = 1$ and $t_1^i = d_i$, for all $i = 1, \ldots, K$. We present a family of instances for which the congestion of an unsplit routing found by randomized rounding or any other path-selecting algorithm is always a factor $\Omega(\log m / \log\log m)$ away from the congestion of an optimal solution (one of minimum congestion).

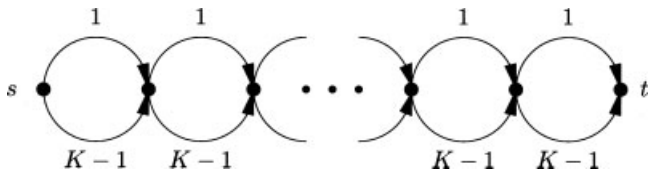

FIG. 1. An instance of the unsplittable flow problem consisting of K commodities with unit demands, sharing a common source s and a common sink t. There are K^K consecutive pairs of parallel edges. The numbers at the edges indicate capacities.

Theorem 1. *The performance ratio of randomized rounding for the minimum congestion version of the unsplittable flow problem is* $\Omega(\log m / \log\log m)$.

We present the proof for the lower bound on the performance ratio for the directed case, but it works for the undirected case as well. (For undirected graphs Theorem 1 can already be found in [18].) Moreover, our proof works for all "path-selecting" algorithms, which means for algorithms that start with a solution for the classical multicommodity flow problem, compute a path decomposition, and select one of its paths for each commodity to obtain an unsplit routing.

Proof. Consider an instance of the UFP consisting of K^K consecutive pairs of parallel edges and K commodities with unit demands, sharing a common source s (the "first" node of the consecution of edges) and a common sink t (the "last" node of the consecution of edges). Each "upper" edge in a pair has a capacity of 1 and the "lower" edges have capacities $K - 1$. This instance of the UFP is depicted in Figure 1. Obviously, there exists an unsplit routing with congestion 1 where the flow value on any edge is equal to its capacity. Of course, the latter property must also hold for any optimal fractional solution. Consider the following fractional path decomposition: each commodity $i = 1, \ldots, K$ is distributed among K paths $P_1^i, \ldots, P_K^i$ with flow values equal to $1/K$. The K^K upper edges of the graph are labeled by K-tuples $(j_1, \ldots, j_K) \in \{1, \ldots, K\}^K$ such that every edge gets a different label. For $i, j = 1, \ldots, K$, path P_j^i uses an upper edge labeled $(j_1, \ldots, j_K)$ if and only if $j_i = j$. The underlying intuition of this construction is that, for each subset of paths containing exactly one path of each commodity, there is an upper edge that is exactly used by the paths in the considered subset.

We argue that, for the described path decomposition, randomized rounding always yields an unsplit routing of congestion K: by construction of the fractional path decomposition, for every possible random choice, there exists an upper edge of unit capacity that is used by all K commodities. In particular, the flow on this edge exceeds its capacity by a factor K. Because $m = 2K^K$, we get $K = \Omega(\log m / \log\log m)$. ∎

The performance of randomized rounding depends on the particular choice of an optimal fractional solution, that is, on the paths along which flow is sent. At first sight, the fractional solution (path decomposition) used in the proof above seems rather artificial for the considered network. We show in the following that this solution is indeed an extreme point (i.e., a basic solution) for the path-based LP-formulation of the fractional flow problem.

Let $\mathcal{P}$ be the set of all s–t-paths in the network under consideration. We formulate the problem of finding an optimal fractional routing as a linear program with variables $f_{P,i} \geq 0$ denoting the amount of flow of commodity $i = 1, \ldots, K$ sent along path $P \in \mathcal{P}$. The constraints of the linear program are as follows:

$$\sum_{i=1}^{K} \sum_{P \in \mathcal{P}:\, e \in P} f_{P,i} = 1 \qquad \text{for all upper edges } e, \tag{5}$$

$$\sum_{P \in \mathcal{P}} f_{P,i} = 1 \qquad \text{for all } i \in \{1, \ldots, K\}, \tag{6}$$

Because the sum of capacities of every pair of parallel edges equals the total flow value K, constraints (5) suffice to enforce all capacity constraints. Let $\mathcal{P}_i = \{P_1^i, \ldots, P_K^i\}$ be the subset of paths used by commodity i in the path decomposition described above.

Lemma 1. *The columns of the matrix given by the left-hand sides of* (5) *and* (6) *corresponding to the variables $f_{P,i}$, with $i = 1, \ldots, K$ and $P \in \mathcal{P}_i$, are independent.*

Proof. We assume by contradiction that there exists a nontrivial linear combination of these columns that yields the zero vector. For $i, j = 1, \ldots, K$, the coefficient of the column corresponding to variable $f_{P,i}$ with $P = P_j^i$ in this linear combination is denoted by a_{ij}. By symmetry, we assume without loss of generality that $a_{11} > 0$. Considering the row corresponding to the ith constraint of (6), we observe that $\sum_{j=1}^{K} a_{ij} = 0$, for all $i = 1, \ldots, K$. Thus, for all $i = 2, \ldots, K$, there exists a nonnegative coefficient $a_{ij_i} \geq 0$. However, considering the constraint in (5) corresponding to the upper edge labeled $(1, j_2, \ldots, j_K)$, we get $a_{11} + \sum_{i=2}^{K} a_{ij_i} = 0$ which is a contradiction and therefore concludes the proof. ∎

Lemma 1 implies that the path decomposition from above yields a basic feasible solution to the linear program. Nevertheless, the chosen path decomposition seems odd since it uses K^2 paths, while there is an obvious path decomposition using only K paths (i.e., the optimal unsplittable flow solution). It is an interesting question whether the lower bound $\Omega(\log m / \log \log m)$ still holds if the randomized rounding algorithm starts with a path decomposition of an optimal fractional solution using a minimal number of paths. This is obviously not true for the instance discussed above.

In the following, we modify the instance such that an optimal fractional solution using a minimal number of paths still yields the lower bound $\Omega(\log m / \log \log m)$ for randomized rounding. We slightly perturb the right-hand sides of constraints (5) and (6). To enforce the balance condition, we perturb the upper edge capacities to values slightly above 1 and the demands to values slightly below 1. Furthermore, we also modify the capacities of the lower edges such that again the capacities of all pairs of edges are equal to the total demand.

By choosing a random perturbation, with probability 1 there are no degenerate basic solutions anymore, that is, all basic variables (paths) have a positive value (for more details on perturbations of linear programs we refer to the standard literature such as, e.g., [7]). This means that all basic solutions use a minimal number of paths. In particular, by Lemma 1, there is a perturbed version of our fractional solution considered above, which is a basic solution to the modified instance and thus uses a minimal number of paths. Because the perturbation can be arbitrarily small, this basic solution to the perturbed instance is arbitrarily close to the fractional solution considered above such that the expected congestion of an unsplit routing obtained by randomized rounding is in $\Omega(\log m / \log \log m)$. Moreover, the congestion of an optimal unsplit routing is still arbitrarily close to 1. This yields the following corollary.

Corollary 1. *The performance ratio of randomized rounding for the minimum congestion version of the unsplittable flow problem is $\Omega(\log m / \log \log m)$, even if the algorithm starts with an optimal fractional solution using a minimal number of paths.*

4. THE k-SPLITTABLE FLOW PROBLEM WITH WEIGHT CAPACITIES

In this section we consider the first variant of the k-SFP with path capacities, that is, the case of *weight capacities*. We present a simple packing strategy which allows us to reduce the k-SFP with path capacities to the UFP while losing at most a factor 2 with respect to minimum congestion. We decompose the demand d_i of commodity i into k_i pieces $\text{load}_1^i, \ldots, \text{load}_{k_i}^i$ and then replace request i by k_i copies of i, whose demands are set to the values of the demand pieces. To make the procedure more vivid, one may always think of packing k_i containers for each commodity i which are then routed along certain paths through the network. In the following we use Π_k and Π_u to denote the currently considered instance of the k-SFP with path capacities and its corresponding instance of the UFP that results from our packing.

After defining the load for each container, we consider an optimal solution to Π_k and interpret its flow values as capacities of bins. This interpretation comes from the idea that we can assign our loads to these resulting bins for the purpose of sending each load (container) along the same path as the corresponding bin is taking in the optimal solution. We show that one can find an assignment of loads to bins such that the bins are overloaded by a factor of at most 2. Consequently, there exists a solution to Π_k, which uses our

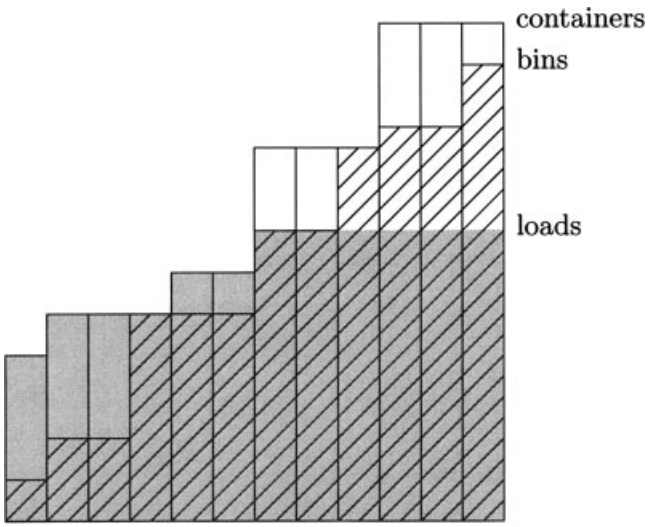

FIG. 2. An example for the loading of the containers. The beams represent the containers, the filled part the loading defined in (7), and the striped part the loads of an optimal solution.

packing of containers and has a congestion of at most twice the minimum congestion.

For the purpose of assigning the loads to the bins with a small overload factor, it appears to be reasonable to break the demand of any commodity into pieces which are as small as possible. In other words, we want to load the containers such that the maximal load is minimized. One can visualize this by picturing that all containers of the same commodity are lined up in a row and merged with each other and get filled up with water (see Fig. 2). To simplify notation, we assume without loss of generality that $t^i_1 \le t^i_2 \le \cdots \le t^i_{k_i}$, for all $i = 1, \ldots, K$. Then, we can assume that the loads load^i_j of commodity i are indexed in nondecreasing order also. A precise description of the loading strategy depicted in Figure 2 is as follows:

$$\text{load}^i_j = \min\left\{ t^i_j, \frac{1}{k_i - j + 1}\left(d_i - \sum_{j'=1}^{j-1} \text{load}^i_{j'} \right) \right\}. \quad (7)$$

For any $i \in \{1, \ldots, K\}$, we can simply start with $j = 1$ and then calculate the loads in increasing order. Note that two containers of same size get the same load. Thus, it even suffices to calculate the load for only one container per size.

Now consider any optimal solution F to the underlying problem of minimizing the congestion for Π_k. Denote its flow values by $\text{bin}^i_1, \ldots, \text{bin}^i_{k_i}$, for all $i = 1, \ldots, K$. If we arrange the bins in nondecreasing order, for all $i = 1, \ldots, K$, then we can prove the following lemma.

Lemma 2. *For all $i \in \{1, \ldots, K\}$ the following holds: If for some $j \in \{1, \ldots, k_i\}$ it holds that $\text{load}^i_j \ge \text{bin}^i_j$, then $\text{load}^i_{j'} \ge \text{bin}^i_{j'}$, for all $j' < j$.*

Proof. Because F is an optimal and thus feasible solution, no bin is larger than its path capacity. Hence, a violation of the claim would contradict our loading strategy (see also Fig. 2). ∎

Next we prove that we can pack k_i items of sizes $\text{load}^i_1, \ldots, \text{load}^i_{k_i}$ into k_i bins of sizes $\text{bin}^i_1, \ldots, \text{bin}^i_{k_i}$ without exceeding the bin sizes by more than a factor 2. This implies that we can simultaneously route all demands in Π_u, despite not allowing a congestion greater than twice the congestion of the optimal solution F. (Remember that Π_u resulted from replacing each request in Π_k by exactly k_i copies and giving the copies the demands $\text{load}^i_1, \ldots, \text{load}^i_{k_i}$.)

Lemma 3. *Let $a_1 \le a_2 \le \cdots \le a_n$ and $b_1 \le b_2 \le \cdots \le b_n$ be nonnegative real numbers with $\sum_{i=1}^n a_i = \sum_{i=1}^n b_i$ and satisfying the following: if $a_I \ge b_I$ for some $I \in \{1, \ldots, n\}$, then $a_i \ge b_i$ for all $i = 1, \ldots, I$. Then there exists an assignment $f : \{1, \ldots, n\} \to \{1, \ldots, n\}$ such that $\sum_{j \in f^{-1}(i)} a_j \le 2b_i$, for all $i = 1, \ldots, n$.*

Proof. If $a_i \le 2b_i$ for all $i \in \{1, \ldots, n\}$, we are done, because the identity function $f = id$ fulfills the requirement. Thus, we assume that there exists an i with $a_i > 2b_i$; let $I := \max\{i \mid a_i > 2b_i\}$. Notice that $a_i \ge b_i$, for all $i \in \{1, \ldots, I\}$. For all $i \in \{1, \ldots, n\}$ with $b_i \le a_i \le 2b_i$, we set $f(i) := i$ and exclude these indices from further consideration. The remaining indices are partitioned into a set of *small indices* $\mathcal{I}_s := \{i \mid i \le I \text{ and } a_i > 2b_i\}$ and a set of *large indices* $\mathcal{I}_\ell := \{i \mid i > I \text{ and } a_i < b_i\}$. Notice that $\mathcal{I}_\ell \ne \emptyset$, because otherwise $\sum_{i=1}^n a_i > \sum_{i=1}^n b_i$. Finally, define the *overload* as $\sum_{i \in \mathcal{I}_s}(a_i - b_i)$ and the *empty space* as $\sum_{i \in \mathcal{I}_\ell}(b_i - a_i)$; notice that the empty space is at least as large as the overload. Moreover, because $a_i > 2b_i$, for each $i \in \mathcal{I}_s$, it follows that $\sum_{i \in \mathcal{I}_s} a_i$ is bounded from above by twice the empty space.

For $i \in \mathcal{I}_\ell$, define $f(i) := i$ and set $u_i := 2b_i - a_i$. Consider the elements $i \in \mathcal{I}_s$ in decreasing order and assign i to the largest index in $\mathcal{I}_\ell$ for which the sum of the elements currently assigned to it does not exceed u_i, that is, $f(i) := \max\{i' \in \mathcal{I}_\ell \mid \sum_{j \in f^{-1}(i')} a_j \le u_{i'}\}$.

After this procedure, $\sum_{j \in f^{-1}(i)} a_j \le 2b_i$, for all $i \in \mathcal{I}_\ell$, because the value a_j, which was assigned to b_i last, is at most a_i. Because the values assigned to b_i before sum up to at most u_i, we obtain the desired result. On the other hand, we must have assigned each index in $\mathcal{I}_s$ to an index in $\mathcal{I}_\ell$ because $\sum_{j \in \mathcal{I}_s} a_j$ is at most twice the empty space. ∎

Note that for each $i \in \{1, \ldots, K\}$ the loads and the bins satisfy the requirements from Lemma 3. Thus, we can assign all loads to the bins without exceeding the bin capacities by a factor greater than 2. Now consider the mentioned Π_u: take Π_k. For all $i \in \{1, \ldots, K\}$, replace request i by k_i copies of i with demands equal to $\text{load}^i_1, \ldots, \text{load}^i_{k_i}$. Using the results from above, we obtain the following:

Theorem 2. *For the problem of minimizing congestion and each instance of the k-SFP with path capacities (weight capacities), there is a corresponding instance of the UFP*

such that a ρ-approximate solution to this instance yields a 2ρ-approximate solution to the underlying instance of the k-SFP with path capacities.

We conclude this section with some remarks on the running time of the resulting 2ρ-approximation algorithms for the k-SFP with path capacities. Notice that the size of Π_u is not necessarily polynomial in the input size of Π_k. The number of copies k_i of request i can be exponential, if in Π_k we are only given the different sizes of containers and the number of containers for each size. As already mentioned above, it will not be a problem to compute the loads in polynomial time. However, to achieve polynomial running time when solving the problem of minimizing congestion on Π_u, we must use a polynomial time algorithm for the UFP that is able to handle a compact input format where, for each request, the number of copies of the request is given. In fact, most approximation algorithms for the UFP fulfill this requirement or can easily be modified to fulfill it. As an example, we mention the randomized rounding procedure: in a fractional solution, all copies of a request can be treated as one commodity for which a path decomposition using at most m paths exists. Randomized rounding then assigns to each of these paths a certain number of copies of the corresponding request. This procedure can easily be implemented to run in polynomial time.

5. THE k-SPLITTABLE FLOW PROBLEM WITH SIZE CAPACITIES

In this section we consider the second variant of the k-SFP with path capacities, that is, the case of *size capacities*. If all path capacities are equal to some uniform value T_i, for all commodities i, it is easy to observe that there always exists an optimal solution using the minimum number of paths $\lceil d_i/T_i \rceil$ for commodity i. Therefore, this problem can be formulated as a "uniform exactly-k-SFP" (see [5] for details). In particular, all results for the latter problem obtained in [5] also hold for the k-SFP with uniform path capacities. The most important result of [5] in this context is that any instance of the considered problem can be solved by solving a special instance of the UFP on the same graph, and thus any ρ-approximation algorithm for the problem of minimizing the congestion for the UFP provides a ρ-approximation algorithm for the problem of minimizing the congestion for our problem here.

We turn to the general problem. As in the last section, we assume that $t_1^i \leq \cdots \leq t_{k_i}^i$, for all $i = 1, \ldots, K$. Our general approach is identical to the one presented in the last section. Again, we first give a simple strategy for the packing of containers. This reduces the problem to a UFP. We prove that at most a factor 2 in the performance is lost due to this packing by comparing it to the packing of an optimal solution ("bins"). Tying in with the previous section we use Π_k to denote the currently considered instance of the k-SFP with path capacities and Π_u to denote its corresponding instance of the UFP, which results from the packing introduced below.

In contrast to the last section, it seems to be reasonable to load each container that is used up to its size. Notice that only the size of a container (and not its actual load) matters for the routing. Among all containers, we try to use the smallest ones to be able to obtain reasonable assignments to bins later. The load load_j^i of the jth container of commodity i is determined as follows: Consider the containers of each commodity in order of decreasing size. If the whole (remaining) demand of the commodity fits into the remaining containers with smaller indices, then discard the container. Otherwise, the container is filled up to its size (unless the remaining demand is smaller than the size) and the remaining demand is decreased by this value. This loading strategy can be easily described by the following pseudocode. For all $i \in \{1, \ldots, K\}$, start with $j = k_i$ and then calculate the flow values in decreasing order as follows:

$$\alpha = \max \left\{ d_i - \sum_{l=j+1}^{k_i} \text{load}_l^i - \sum_{l=1}^{j-1} t_l^i, 0 \right\}$$

If $(\alpha > 0)$ then

$$\text{load}_j^i = \min \left\{ t_i^j, d_i - \sum_{l=j+1}^{k_i} \text{load}_l^i \right\}$$

else

$$\text{load}_j^i = 0$$

The strategy above finds the loads in polynomial time if the path capacity of every path is explicitly given in the input. If the encoding of the input is more compact and, for each commodity i, only specifies the different path capacities together with the number of paths for each capacity, the algorithm can be easily adapted to run still in polynomial time.

In the following we consider a fixed optimal solution to Π_k. For all $i \in \{1, \ldots, K\}$, let $\mathcal{O}_i \subseteq \{1, \ldots, k_i\}$ such that $j \in \mathcal{O}_i$ if and only if the jth container is used by the optimal solution. Similarly, let $\mathcal{B}_i \subseteq \{1, \ldots, k_i\}$ such that $j \in \mathcal{B}_i$ if and only if our loading uses the jth container, that is, $\text{load}_j^i > 0$. The following lemma says that the containers used by our loading can be packed into the containers (bins) used by the optimal solution such that the size of any container of the optimal solution is exceeded by at most a factor 2.

Lemma 4. *For all $i = 1, \ldots, K$, there exists an assignment $f : \mathcal{B}_i \to \mathcal{O}_i$ such that*

$$\sum_{j' \in f^{-1}(j)} t_{j'}^i \leq 2t_j^i \qquad \text{for all } j \in \mathcal{O}_i. \tag{8}$$

Proof. In the following, we keep i fixed; to simplify notation, we omit all indices and superscripts i. By construction of our packing strategy,

$$\sum_{j \in \mathcal{B}:\, j > j_0} t_j < \sum_{j \in \mathcal{O}:\, j \geq j_0} t_j \qquad \text{for all } j_0 \in \mathcal{B}. \tag{9}$$

In particular, it follows from (9) that the maximum index in $\mathcal{O}$ is at least as large as the maximum index in $\mathcal{B}$, and therefore, the largest container (bin) in the optimal solution is at least as large as the largest container in our packing. Although $\mathcal{B} \neq \emptyset$, we construct the assignment f iteratively as follows:

$$j_{\max} := \max\{j \mid j \in \mathcal{O}\}$$

$$\text{and} \qquad \bar{j} := \min\left\{ j \in \mathcal{B} \mid \sum_{j' \in \mathcal{B}:\, j' > j} t_{j'} \leq t_{j_{\max}} \right\}.$$

Set $f(j) := j_{\max}$ for all $j \in \mathcal{B}$ with $j \geq \bar{j}$; set $\mathcal{O} := \mathcal{O} \setminus \{j_{\max}\}$ and $\mathcal{B} := \mathcal{B} \setminus \{j \mid j \geq \bar{j}\}$. Notice that property (8) holds for $j_{\max}$ because, by (9), the container size $t_{j_{\max}}$ is an upper bound on t_j, for all $j \in \mathcal{B}$. Moreover, (9) is still fulfilled for the reduced sets $\mathcal{B}$ and $\mathcal{O}$. This concludes the proof. ∎

For all $i = 1, \ldots, K$ and $j = 1, \ldots, k_i$, set $\tilde{d}^i_j := t^i_j$ if $j \in \mathcal{B}_i$, and $\tilde{d}^i_j := 0$, otherwise. We define Π_u by replacing each request i by k_i copies with demands $\tilde{d}^i_1, \ldots, \tilde{d}^i_{k_i}$. According to Lemma 4, there exists a solution to Π_u whose congestion is at most twice the congestion of an optimal solution to Π_k.

Theorem 3. *For the problem of minimizing congestion and each instance of the k-SFP with path capacities (size capacities), there is a corresponding instance of the UFP such that a ρ-approximate solution to this instance yields a 2ρ-approximate solution to the underlying instance of the k-SFP with path capacities.*

With respect to the running time of the resulting 2ρ-approximation algorithms for the k-SFP with path capacities, the same remarks hold that we made in the last section after Theorem 2.

We conclude this section with the discussion of a special case of the k-SFP with path capacities where, for each request i, each of its path capacities is a multiple of all of its smaller path capacities. This property holds, for example, if all path capacities are powers of 2. In this case, we can use the same loading strategy as above to obtain a ρ-approximate solution to the underlying problem from any ρ-approximation algorithm for the UFP. This holds, because—with the same terms as used above—for all $i = 1, \ldots, K$, there exists an assignment $f : \mathcal{B}_i \to \mathcal{O}_i$ such that $\sum_{j' \in f^{-1}(j)} t^i_{j'} \leq t^i_j$ for all $j \in \mathcal{O}_i$. The argument is similar to that in the proof of Lemma 4: We can use the same strategy. If we choose $\bar{j} := \min\{ j \in \mathcal{B} \mid \sum_{j' \in \mathcal{B}:\, j' \geq j} t_{j'} \leq t_{j_{\max}} \}$ we obtain (except for the last step) that $\sum_{j' \in \mathcal{B}:\, j' \geq \bar{j}} t_{j'} = t_{j_{\max}}$. Thus, we get rid of the factor 2 in (8). (In the last step it might happen that $\sum_{j' \in \mathcal{B}:\, j' \geq \bar{j}} t_{j'} < t_{j_{\max}}$, but because all containers are assigned after this step, we do not need to take care of this.)

CONCLUSION

In Section 3 we proved that the performance ratio of all path-selecting algorithms for the UFP is $\Omega(\log m / \log\log m)$, no matter if we consider the directed or the undirected case. Leighton et al. [18] proved an integrality gap of the same size for directed graphs, but the integrality gap for the undirected case is still an open question.

For the k-SFP with path capacities we showed for both cases—the one with weight and the one with size capacities—that any ρ-approximation algorithm for the UFP can be turned into a 2ρ-approximation algorithm for the respective considered problem. It is not known yet, if this additional factor of 2 is indeed tight.

Acknowledgments

We thank two anonymous referees for their insightful comments that helped to improve the presentation of this article. An extended abstract appeared in *Proceedings of the 12th Annual European Symposium on Algorithms*, 2004, pp. 520–531.

REFERENCES

[1] M. Andrews and L. Zhang, "Hardness of the undirected congestion minimization problem," Proceedings of the 37th Annual ACM Symposium on Theory of Computing, 2005, pp. 284–293.

[2] Y. Azar and O. Regev, "Strongly polynomial algorithms for the unsplittable flow problem," Proceedings of the 8th Conference on Integer Programming and Combinatorial Optimization, 2001, pp. 15–29.

[3] A. Bagchi, Efficient strategies for topics in internet algorithmics, PhD thesis, The Johns Hopkins University, October 2002.

[4] A. Bagchi, A. Chaudary, C. Scheideler, and P. Kolman, "Algorithms for fault-tolerant routing in circuit switched networks," Fourteenth ACM Symposium on Parallel Algorithms and Architectures, 2002.

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

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

[7] D. Bertsimas and J. Tsitsiklis, Introduction to linear optimization, Athena Scientific, Belmont, MA, 1997.

[8] C. Chekuri and S. Khanna, "Edge disjoint paths revisited," Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms, 2003.

[9] J. Chuzhoy and J. Naor, "New hardness results for congestion minimization and machine scheduling," Proceedings of the 36th Annual ACM Symposium on Theory of Computing, 2004, pp. 28–34.

[10] Y. Dinitz, N. Garg, and M.X. Goemans, On the single source unsplittable flow problem, Combinatorica 19 (1999), 17–41.

[11] M.R. Garey and D.S. Johnson, Computers and intractability: A guide to the theory of NP–completeness, Freeman, San Francisco, 1979.

[12] V. Guruswami, S. Khanna, R. Rajaraman, B. Shepherd, and M. Yannakakis, "Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems," Proceedings of the 31st Annual ACM Symposium on Theory of Computing, 1999, pp. 19–28.

[13] J. Kleinberg and R. Rubinfeld, "Short paths in expander graphs," Proceedings of the 37th Annual Symposium on Foundations of Computer Science, 1996, pp. 86–95.

[14] J.M. Kleinberg, Approximation algorithms for disjoint path problems, PhD thesis, Massachusetts Institute of Technology, May 1996.

[15] J.M. Kleinberg, "Single-source unsplittable flow," Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science, 1996, pp. 68–77.

[16] S.G. Kolliopoulos and C. Stein, Approximation algorithms for single-source unsplittable flow, SIAM J Comput 31 (2002), 919–946.

[17] P. Kolman and C. Scheideler, "Improved bounds for the unsplittable flow problem," Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms, 2002, pp. 184–193.

[18] T. Leighton, S. Rao, and A. Srinivasan, "Multicommodity flow and circuit switching," Proceedings of the 31st Hawaii International Conference on System Sciences, 1998, pp. 459–465.

[19] P. Raghavan, Probabilistic construction of deterministic algorithms: Approximating packing integer programs, J Comput Syst Sci 37 (1988), 130–143.

[20] P. Raghavan and C.D. Thompson, Randomized rounding: A technique for provably good algorithms and algorithmic proofs, Combinatorica 7 (1987), 365–374.

[21] M. Skutella, Approximating the single source unsplittable min-cost flow problem, Math Program 91 (2002), 493–514.