Improved Bounds for the Unsplittable Flow Problem*

Petr Kolman
Inst. for Theoretical Computer Science†
Charles University
Malostranské nám. 25
118 00 Prague, Czech Republic
kolman@kam.mff.cuni.cz

Christian Scheideler
Dept. of Computer Science
Johns Hopkins University
3400 N. Charles Street
Baltimore, MD 21218, USA
scheideler@cs.jhu.edu

September 10, 2004

Abstract

In this paper we consider the *unsplittable flow problem* (UFP): given a directed or undirected network $G = (V, E)$ with edge capacities and a set of terminal pairs (or requests) with associated demands, find a subset of the pairs of maximum total demand for which a single flow path can be chosen for each pair so that for every edge, the sum of the demands of the paths crossing the edge does not exceed its capacity. We present a collection of new results for the UFP both in the offline (all requests are given from the beginning) and the online (requests arrive at the system one after the other) setting. A fundamental ingredient of our analysis is the introduction of a new graph parameter, the *flow number*, that aims to capture global communication properties of the network. With the help of the flow number we develop a general method for transforming arbitrary multicommodity flow solutions into solutions that use *short* paths only. This generalizes a well-known theorem of Leighton and Rao [16] that applies to uniform flows only. Both the parameter and the method may therefore be of independent interest.

*A preliminary version of this work appeared in the ACM-SIAM *Symposium on Discrete Algorithms*, 2002

†Supported by the Ministry of Education of the Czech Republic as project LN00A056.

1 Introduction

In the *unsplittable flow problem*, denoted by UFP, we are given a directed or undirected network $G = (V, E)$, $n = |V|$, $m = |E|$, with edge capacities prescribed by $c : E \to \mathbb{R}_+$ and a set $T = \{(s_i, t_i) : 1 \leq i \leq k\}$ of k terminal pairs (or *requests*) with demands $d_i \in [0, 1]$. (The requirement that $d_i \in [0, 1]$ is not a restriction, since this can always be achieved by suitably scaling the demands and capacities.) A feasible solution is a subset $S \subseteq T$ of the requests such that the demand of each request in S is satisfied by a single flow path and the capacity constraints are fulfilled. The objective is to maximize the total demand of the satisfied requests. The UFP is a natural generalization of the classical edge disjoint paths problem.

One of the motivations for the UFP is the problem of allocating bandwidth for traffic with different bandwidth requirements in heterogeneous networks. Unfortunately, the UFP is MAXSNP-hard [7]. Therefore, the best one can hope for (unless $P = NP$) is to find good approximate solutions. Approximation algorithms for the UFP and related problems have been presented in several prior works [9, 18, 11, 12, 6, 7, 3, 14]. Kleinberg [9] provides a comprehensive background on these problems. Many of these algorithms begin with a linear programming relaxation of the problem (i.e., instead of using a single path, a commodity is shipped along multiple paths) and then round the solution in a suitable way to obtain an approximate solution for the UFP.

Under the assumption that the maximum demand of a commodity, d_{max}, does not exceed the minimum edge capacity, called *no-bottleneck assumption* in the following, Kleinberg [9] presented an $O(\sqrt{m} \cdot \frac{\max c(e)}{\min d_i})$ approximation algorithm for the UFP. This was subsequently improved by Baveja and Srinivasan [3] to $O(\sqrt{m})$, by an LP-based algorithm. Azar and Regev [2] gave a simpler, combinatorial algorithm with the same approximation guarantee. Kolliopoulos and Stein [11] presented the first nontrivial approximation, $O(\log m\sqrt{m})$, for a more general version of the UFP in which each request has an associated *profit* p_i and the goal is to maximize the total profit of accepted requests (*UFP with profits*). They also showed that the total profit of a solution computed by their algorithm is $\Omega\left(OPT^2/(m \log^2 m \log\log m)\right)$ where OPT denotes the total profit of the optimal solution; the above mentioned algorithm of Baveja and Srinivasan [3] guarantees a solution of total profit $\Omega(OPT^2/m)$.

Without the no-bottleneck assumption, Guruswami et al. [7] gave a randomized algorithm with an approximation ratio of $O(\sqrt{m} \log^{3/2} m)$. The algorithm is based on a suitable rounding of an LP relaxation of the UFP. Recently, Azar and Regev [2] described a deterministic algorithm with an approximation ratio of $O(\sqrt{m} \log m)$. Both of these results require the ratio between the largest edge capacity and minimum demand to be polynomially bounded. We present an algorithm with an essentially optimal approximation ratio of $O(\sqrt{m})$, unless $P = NP$, without using any assumption about the ratio of the edge capacities and the demands. The bound is achieved with a simple greedy algorithm.

On the lower bound side, it was shown by Guruswami et al. [7] that on

directed networks it is NP-hard to approximate the UFP within a factor of $m^{1/2-\epsilon}$, for any $\epsilon > 0$. Azar and Regev proved that without the no-bottleneck assumption, it is NP-hard to approximate the UFP with profits within $\Omega(m^{1-\epsilon})$, for any $\epsilon > 0$. This paper does not deal with the UFP with profits.

The $O(\sqrt{m})$ approximation and the matching $m^{1/2-\epsilon}$ lower bound [7] are not very satisfying. However, it is important to correctly understand the meaning of the lower bound: for any $\epsilon > 0$ there *exists* a directed network for which it is NP-hard to approximate the UFP within a factor of $m^{1/2-\epsilon}$. The lower bound does not imply anything about the possibility to approximate the UFP for most other networks (and, in fact, anything for any undirected network). However, still if we think about general algorithms for arbitrary networks and m is the *only* known parameter, we cannot get (much) better results than $O(\sqrt{m})$. Thus, it is necessary to use other graph parameters, for example the edge expansion, to be able to get below the $\sqrt{m}$ approximation ratio for many classes of networks. We will consider this approach for the rest of this section. All the time we will assume that the no-bottleneck assumption holds.

For a special case of the UFP, the unit-capacity UFP (that is, all edges have a capacity of one), Baveja and Srinivasan [3] gave an algorithm with an $O(\Delta^2\alpha^{-2}\log^3 n)$ approximation ratio, where Δ is the maximum degree and α is the edge expansion. A crucial building block of the paper is an observation of Kleinberg and Rubinfeld [10] that for a multicommodity flow problem an almost optimal solution can always be obtained by flows of length at most $O(\Delta^2\alpha^{-2}\log^3 n)$. The algorithm itself is quite involved and the proof heavily uses probabilistic tools like the FKG inequality. Combining the results of Kleinberg and Rubinfeld [10] and Leighton and Rao [16], the approximation ratio can be decreased to $O(\Delta^2\alpha^{-2}\log^2 n)$. Recently, Kolman and Scheideler [14] improved this ratio to $O(\Delta^2\alpha^{-1}\log n)$, and for certain classes of graphs in which the diameter is much lower than α^{-1}, they even prove stronger results, using a so-called *Shrewd algorithm*. Using a new parameter called the *flow number* F of a network, we improve the general approximation ratio further to $O(F)$ with $F = O(\Delta\ \alpha^{-1}\log n)$. Redefining Δ and α with respect to the different capacities (i.e., Δ as the maximum total capacity leaving or leading to a node, and α respecting the capacities) the bound can be extended to arbitrary undirected networks. In contrast to the results of Bajeva and Srinivasan [3] both our algorithm and its analysis are very simple and, especially, they do not need tools from probabilistic analysis. Moreover, it is possible to make the algorithm online.

Imposing a stronger restriction on the maximum demand than just the no-bottleneck assumption (i.e., $d_{\max} \leq c_{\min}$), namely, having the ratio $c_{\min}/d_{\max}$ lower bounded by a constant larger than 1, allows one to get much better approximation ratios. This setting is sometimes called UFP with *small* or *bounded* demands [2, 3]. In our case (recall the $d_i \in [0, 1]$ assumption), the minimum edge capacity $c_{\min}$ itself serves as the lower bound, and it will be used as an additional parameter. Based on techniques of Awerbuch et al. [1], Azar and Regev [2] recently obtained an algorithm with an approximation ratio of $O(c_{\min} \cdot n^{1/c_{\min}})$, for any $2 \leq c_{\min} \leq \log n$. They note that the n in the bound can be replaced by

an upper bound d on the longest path in an optimal solution, however, without giving any better bound on d other than the trivial n. Bajeva and Srinivasan [3] again use the $d = O(\Delta^2 \alpha^{-2} \log^3 n)$ bound by Kleinberg and Rubinfeld [10] and they present an algorithm with an approximation ratio of $O(d^{1/(c_{\min}-1)})$, for any $c_{\min} \geq 2$. The latter result gives a better bound for large $c_{\min}$. For example, for $c_{\min} = \Omega(\log d)$ it implies an $O(1)$-approximation algorithm. We improve the bound of Azar and Regev by presenting an algorithm with an approximation ratio of $O(c_{\min} \cdot (F^{1/c_{\min}} - 1))$ for all integral $c_{\min} \geq 1$ (which is $O(\log F)$ if $c_{\min} \geq \log F$).

The UFP has also been considered in the online setting where the requests arrive one by one and decisions have to be made without any knowledge about the future requests. For $c_{\min} = \Omega(\log n)$, Awerbuch, Azar and Plotkin [1] describe an optimal online algorithm with a competitive ratio of $\Theta(\log n)$. Azar and Regev [2] present a randomized algorithm with a competitive ratio of $O(c_{\min} \cdot (n^{1/c_{\min}} - 1))$ for any $c_{\min} \geq 2$, and they show that no deterministic online algorithm can achieve a better competitive ratio than this. This online lower bound is of the same nature as the $m^{1/2-\epsilon}$ offline lower bound by Guruswami et al. [7] that was explained earlier. It is still possible to get better bounds for many networks if other parameters are used. We present a deterministic online algorithm with a competitive ratio of $O(c_{\min} \cdot (F^{1/(c_{\min}-1)} - 1))$ for all $c_{\min} \geq 2$ and show that any deterministic online algorithm has a competitive ratio of $\Omega(c_{\min} \cdot (F^{1/(c_{\min}-1)} - 1))$. Using randomization, it is possible to get an $O(c_{\min} \cdot (F^{1/c_{\min}} - 1))$ competitive ratio. We also show that if it is allowed to cancel previously established paths, then there is a deterministic online algorithm with a competitive ratio of $O(c_{\min} \cdot (F^{1/c_{\min}} - 1))$, for all $c_{\min} \geq 1$. This demonstrates that the ability to cancel paths is an important and powerful feature. It has not been investigated prior to our work.

Our key technique to obtain most of our results is a technique to shorten flow paths, captured in the *Shortening Lemma*, which may be of independent interest. It allows to transform any feasible solution of a multicommodity flow problem into a solution in which the maximal path length is only $O(F)$ and the edge capacities are overloaded by only a very small constant factor. This generalizes a well-known result about short flow solutions for *uniform* multicommodity flow problems (there is a commodity with demand one for every pair of nodes) by Leighton and Rao [16, Theorem 18] to a result about short flow solutions for *arbitrary* multicommodity flow problems, and also substantially improves and generalizes the above mentioned bound $O(\log^3 n)$ of Kleinberg and Rubinfeld for bounded degree expanders [10, Theorem 6].

There have been several new results since the first publication of this work. Chekuri and Khanna [5] improved the analysis of the old greedy algorithm for the edge disjoint paths problem. They observed that in terms of the number of vertices, the best lower bound is $\Omega(n^{1/2-\epsilon})$ while the best upper bound is $O(n)$ only. They proved that the greedy algorithm gives an $O(n^{2/3})$-approximation in undirected graphs and $O(n^{4/5})$-approximation in directed graphs; on the other hand, there are directed and undirected instances in which the approximation ratio of the greedy algorithm is $\Omega(n^{2/3})$. They showed that the bounds apply

also for the unit-capacity UFP. Based on their work, Kolman [13] proved the same bounds for the general UFP with arbitrary demands and capacities. Hajiaghayi and Leighton [8] improved the upper bound on the approximation ratio of the greedy algorithm on directed graphs to $O(n^{3/4})$ and Varadarajan and Venkataraman further to $O((n \log n)^{2/3})$, almost matching the $\Omega(n^{2/3})$ lower bound for the greedy.

The concept of the flow number was subsequently used by Chakrabarti et al. [4] to design other, LP-based approximation algorithms for the UFP. In contrast to our work, to measure the performance of an algorithm on a network G, they use the flow number of the underlying *graph* G, denoted by F_G, (i.e., F_G depends only on the structure of the underlying graph, not on the edge capacities). They present an $O(F_G \log n)$ approximation for the UFP under the no-bottleneck assumption, an $O(F_G) = O(F)$ approximation for the unit-capacity UFP, an $O((F_G \log n)^{1/c_{\min}})$ approximation for the UFP with small demands and an $O(F_G^{1/c_{\min}})$ approximation for the unit-capacity UFP with small demands. All algorithms are based on randomized rounding of a fractional flow along short paths. In general, the two parameters F and F_G are incomparable. There are networks with $F > F_G$ but also networks with $F \leq F_G$. Thus, for some instances of the UFP, the results of Chakrabarti et al. are an improvement, but they are not an improvement for all.

1.1 Organization of the paper

We start in Section 2 with defining the key new parameter flow number and we compare it with the expansion of a network. In Section 3, the Shortening lemma is given. Section 4 deals with offline algorithms for the UFP, and in Section 5 we present online algorithms for the UFP. The paper ends with a conclusion and open problems.

2 A New Network Measure

Many of the previous techniques have problems proving strong upper bounds on approximation or competitive ratios of algorithms due to the use of inappropriate parameters. As can be seen from the lower bound of Guruswami et al. [7], if m is the only parameter used, an upper bound of $O(\sqrt{m})$ is essentially the best possible. Much better ratios can be shown if the expansion or the routing number [17] of a network are used. These measures give very good bounds for low-degree networks with uniform edge capacities, but are usually very poor when applied to networks of high degree or highly nonuniform edge capacities. For instance, when applying the previously known general bounds to the hypercube on n nodes, then the best approximation ratio is $O(\log^2 n)$. However, it is possible to reduce it to $O(\log n)$. For the purpose of getting more precise bounds for the approximation and competitive ratios of algorithms (that allow, for example, the $O(\log n)$ bound for the hypercube) we introduce a new network measure, the *flow number* F. Apart from allowing more precise results,

the flow number has the advantage that, in contrast to the expansion or the routing number, it can be computed exactly in polynomial time. After defining the flow number, we will compare it in this section with the expansion α of a network and show that $F = O(\Delta\ \alpha^{-1} \log n)$.

Once the flow number is defined it is easy to prove the Shortening Lemma which in turn makes it possible to significantly improve the previous upper bounds on the approximation and competitive ratio for the UFP. To give an example of its usefulness, for networks with flow number $\Theta(\log n)$ like the hypercube, butterfly or expanders, when all capacities are equal to $\log \log n$, the previous best bound on the competitive ratio was $O(\log \log n \cdot n^{1/\log \log n})$ whereas we achieve a bound of $O(\log \log n)$ only.

2.1 Basic notation

A *network* is a graph $G = (V, E)$ with a function $c : E \to \mathbb{R}_+$ denoting the *capacities* of the edges. By $c_{\min} = \min_{e \in E} c(e)$ we denote the *minimum edge capacity* of G. Unless explicitly mentioned, we will assume that G is undirected. (However, all of our results hold up to constant factors also for directed graphs satisfying $\sum_{(v,w) \in E} c(v, w) = \Theta(\sum_{(w,v) \in E} c(w, v))$, for every $v \in V$; for the purpose of presentation we restrict ourselves to undirected graphs.) If we simply talk about a *graph* (and not about a network), we assume that all edges have capacity one. The number of nodes in G will always be denoted by n and the number of edges by m. For any node v, let $c(v) = \sum_{e=\{v,w\} \in E} c(e)$ denote the *capacity of* v and let $\Delta = \max_{v \in V} c(v)$. Given any set of nodes U, let $c(U) = \sum_{v \in U} c(v)$, and given any set of edges H, let $c(H) = \sum_{e \in H} c(e)$. Given a network $G = (V, E)$, we call $c(V)$ the *capacity of* G.

For any set of nodes U, let $\bar{U} = V \setminus U$ denote its complement, let $|U|$ denote its size, and let $(U, \bar{U})$ denote the set of all edges connecting U and $\bar{U}$. The *edge expansion* of a network G (or simply *expansion*) is defined as

$$\alpha = \min_{U \subset V} \frac{c(U, \bar{U})}{\min\{|U|, |\bar{U}|\}} .$$

In a *concurrent multicommodity flow problem* there are k commodities, each with a pair of terminal nodes (s_i, t_i) and demand d_i. We say that the node s_i is the *source* and the node t_i is the *destination*; the commodity *originates* in s_i and *terminates* in t_i. A *feasible* solution is a set of flow paths for the commodities that obey the capacity constraints but need not meet the specified demands. The *flow value of a feasible solution* is the maximum value f such that at least $f \cdot d_i$ units of commodity i are simultaneously routed for each i. The *max-flow* for a multicommodity flow problem is defined as the maximum flow value over all feasible solutions. In contrast to the UFP problem, the commodity i between s_i and t_i is allowed to be sent along multiple paths. For a path p in a solution, the *flow size of the path* is the number of units routed along it. We will need two special classes of multicommodity flow problems in the paper. A *balanced multicommodity flow problem* (BMFP) is a multicommodity

flow problem such that for every node $v \in V$, the sum of the demands of the commodities originating in v is equal to $c(v)$ and the sum of the demands of the commodities terminating in v is also equal to $c(v)$. In a *product multicommodity flow problem* (PMFP) [16], a nonnegative weight $\pi(u)$ is associated with each node $u \in V$. There is a commodity for every ordered pair of nodes and the demand for the pair (u, v) is equal to $\pi(u) \cdot \pi(v)$.

2.2 The flow number

In research about network communication properties, permutation routing has often been used as a benchmark for comparing different networks. This reflects the idea that permutation routing represents the communication behavior of an ideal parallel program: the communication is evenly balanced among the processors. Both the expansion and the routing number [17] are able to describe quite accurately the ability of a network to route arbitrary permutations. However, to achieve an even balance of the communication is only desirable in homogeneous network systems (e.g., parallel computers) but may not be desirable in heterogeneous networks. Therefore, we suggest another benchmark, which is a generalization of the routing number.

Suppose we have a network $G = (V, E)$ with arbitrary non-negative edge capacities. Given a concurrent multicommodity flow problem with feasible solution $\mathcal{S}$, let the *dilation* $D(\mathcal{S})$ of $\mathcal{S}$ be defined as the length of the longest flow path in $\mathcal{S}$ and the *congestion* $C(\mathcal{S})$ of $\mathcal{S}$ be defined as the inverse of its flow value (i.e., the congestion says how many times the edge capacities would have to be increased in order to satisfy the demands of all commodities when using the same set of paths). Let I_0 be the PMFP in which $\pi(v) = c(v)/\sqrt{c(V)}$ for every node v, that is, each ordered pair of nodes (v, w) has a commodity of demand $c(v) \cdot c(w)/c(V)$. The *flow number* $F(G)$ of a network G is defined as the minimum over all feasible solutions $\mathcal{S}$ of I_0 of $\max\{C(\mathcal{S}), D(\mathcal{S})\}$. In the case that there is no risk of confusion, we will simply write F instead of $F(G)$. Note that the flow number of a network is invariant to scaling of the capacities.

The smaller the flow number, the better are the communication properties of the network. For example, $F(\text{line}) = \Theta(n)$, $F(\text{mesh}) = \Theta(\sqrt{n})$, $F(\text{hypercube}) = \Theta(\log n)$ and $F(\text{expander}) = \Theta(\log n)$. (This can be derived from results of Leighton [15] and Scheideler [17].) The following result shows that F can be computed exactly in polynomial time. This seems not to be possible for the routing number or the expansion.

Claim 2.1 *There is an algorithm that computes the exact value of the flow number for every network in polynomial time.*

Proof. Consider any network $G = (V, E)$ with capacities given by c. Let $V = \{v_1, \ldots, v_n\}$ and F be its flow number. The following strategy will serve as a basic building block for our algorithm.

For any $L \in \mathbb{N}$, let $G_L = (V', E')$ denote a directed leveled graph of depth L. Each level has n nodes, and the node set in level $i \in \{0, \ldots, L\}$ is given

by $V_i = \{v_{i,1}, \ldots, v_{i,n}\}$. The set E' consists of all directed edges $(v_{i,k}, v_{j,\ell})$ with $j = i + 1$ and either $k = \ell$ or $\{v_k, v_\ell\} \in E$. For any k and ℓ with $\{v_k, v_\ell\} \in E$ let $E_{k,\ell} = \{(v_{i,k}, v_{i+1,\ell}) : \ i \in \{0, \ldots, L-1\}\}$. Consider now the multicommodity flow problem for G_L in which for each pair of nodes $(v_{0,k}, v_{L,\ell})$ there is a commodity of demand $c(v_k) \cdot c(v_\ell)/c(V)$. Let $\mathcal{S}$ be any solution to this problem. $\mathcal{S}$ is called *feasible* if for every $E_{k,\ell}$ the sum of all the flows traversing the edges in $E_{k,\ell} \cup E_{\ell,k}$ is at most $c(\{v_k, v_\ell\})$. If we allow fractional flows, then it is possible to compute via linear programming a solution $\mathcal{S}$ in polynomial time that minimizes $F_L = \max\{C(\mathcal{S}), L\}$, where $C(\mathcal{S})$ is the congestion of $\mathcal{S}$.

Having such an algorithm for G_L, it is easy to see that $F = \min_L F_L$. Since the function $g : \{1, \ldots, n\} \to \mathbb{R}_+$ with $g(x) = F_x$ is first nonincreasing and then nondecreasing, simple binary search can be applied to find F. □

The following claim shows that the flow number does not only characterize the ability of a network to handle balanced product multicommodity flows but also to handle any balanced multicommodity flow.

Claim 2.2 *For any network G with flow number F and any instance I of the BMFP for G, there is a feasible solution for I with congestion and dilation at most $2F$.*

Proof. The proof uses a strategy similar to the technique of Leighton and Rao [16] for transforming a permutation into an instance of the uniform multicommodity flow problem. The idea is to decompose I into two multicommodity flow problems: for every commodity i with source s_i and destination t_i, the first problem I_1 has commodities i_u from s_i to u for all $u \in V$ with demands $d_{i_u} = d_i \cdot c(u)/c(V)$, and the second problem I_2 has commodities i'_u from u to t_i for all $u \in V$ with demands $d_{i'_u} = d_i \cdot c(u)/c(V)$. For every commodity i from the original problem, the total demand of corresponding commodities in I_1 is d_i and is d_i in I_2 as well. Moreover, for every node $u \in V$, the amount of the commodity i_u terminating in u in I_1 is equal to the amount of the commodity $i_{u'}$ originating in u in I_2.

Both of the flow problems I_1 and I_2 are PMFPs with $\pi(v) = c(v)/\sqrt{c(V)}$ for every node v, because for any pair $v, w \in V$, the total demand of the commodities with source v and destination w in I_1 is equal to

$$\sum_{i:\ s_i = v} \frac{d_i \cdot c(w)}{c(V)} = \frac{c(v) \cdot c(w)}{c(V)} = \pi(v) \cdot \pi(w) \ ,$$

and in I_2 is equal to

$$\sum_{i:\ t_i = w} \frac{d_i \cdot c(v)}{c(V)} = \frac{c(v) \cdot c(w)}{c(V)} = \pi(v) \cdot \pi(w) \ .$$

Thus, according to the definition of the flow number, both I_1 and I_2 have a feasible solution with congestion and dilation at most F. Hence, the original problem I has a feasible solution with congestion and dilation at most $2F$, which proves the claim. □

2.3 Flow number vs. expansion

Next we compare the flow number with the expansion. Consider a PMFP with weights $\pi(u)$ for all nodes u, and let p denote the number of nodes with nonzero weight. Without loss of generality, we assume that $p = \sum_{u \in V} \pi(u)$ (otherwise scale c and π appropriately). The *min-cut of a PMFP* is defined as

$$S = \min_{U \subset V} \frac{c(U, \bar{U})}{\pi(U)\pi(\bar{U})} \quad \text{, where} \quad \pi(U) = \sum_{u \in U} \pi(u) \ .$$

Leighton and Rao [16, Theorem 18] proved the following theorem about the relationship between the min-cut, max-flow, and the length of the flow paths for a PMFP:

Theorem 2.3 (Leighton, Rao, 1999) *Given any PMFP for which the min-cut has size S, there is a flow of size $f = \Omega(S/\log p)$ for which every flow path has a length of at most*

$$L = O\left(\frac{\max\{\hat{\gamma}, c(V)/p\} \log p}{p \cdot S}\right) \quad \text{, where} \quad \hat{\gamma} = \max_{u \in V : \pi(u) > 0} \frac{c(u)}{\pi(u)} \ .$$

The following definition will turn out to be useful. The *weighted expansion* of a network G is defined as

$$\beta = \min_{U \subset V} \frac{c(U, \bar{U})}{\min\{c(U), c(\bar{U})\}} \ .$$

Using Theorem 2.3, we prove the following result.

Theorem 2.4 *For any network G with expansion α and $c_{\min} \geq 1$ it holds for its flow number F that*

$$F = \Omega(\alpha^{-1}) \quad \textit{and} \quad F = O(\Delta \cdot \alpha^{-1} \log n)$$

where $\Delta = \max_{v \in V} c(v)$. Furthermore, there are families of networks that match the upper and lower bounds.

Proof. We start with the following lemma:

Lemma 2.5 *For any network G with weighted expansion β and flow number F it holds that*

$$F = \Omega(\beta^{-1}) \quad \textit{and} \quad F = O(\beta^{-1} \log n)$$

Proof. First we prove that $F \geq \beta^{-1}/2$. Let f be the max-flow of the problem I_0 used for the definition of F. Then it holds that for any set U,

$$f \leq \frac{c(U, \bar{U})}{\pi(U) \cdot \pi(\bar{U})} \ ,$$

where $\pi(U) \cdot \pi(\bar{U}) = c(U) \cdot c(\bar{U})/c(V)$. We distinguish between two cases. If $c(U) \geq c(V)/2$, then $c(U) \cdot c(\bar{U})/c(V) \geq c(\bar{U})/2$. Thus,

$$f \leq \frac{c(U,\bar{U})}{c(\bar{U})/2} = 2\frac{c(U,\bar{U})}{\min\{c(U), c(\bar{U})\}} .$$

If $c(\bar{U}) \geq c(V)/2$, then $c(U) \cdot c(\bar{U})/c(V) \geq c(U)/2$ and therefore

$$f \leq \frac{c(U,\bar{U})}{c(U)/2} = 2\frac{c(U,\bar{U})}{\min\{c(U), c(\bar{U})\}} .$$

Hence, in both cases,

$$f \leq 2\frac{c(U,\bar{U})}{\min\{c(U), c(\bar{U})\}}$$

and therefore $f \leq 2 \cdot \beta$ or $2/f \geq \beta^{-1}$. Using the fact that $F \geq 1/f$ it follows that $F = \Omega(\beta^{-1})$.

Next we show that $F = O(\beta^{-1} \log n)$. Consider the PMFP I_0 and let S be the min-cut for it. Recall the notation used in Theorem 2.3. We require there that $p = \sum_{u \in V} \pi(u)$. In our case, $\sum_{u \in V} \pi(u) = \sum_{u \in V} c(u)/\sqrt{c(V)} = \sqrt{c(V)}$, but since F is invariant to scaling we can scale the capacities so that $\sqrt{c(V)} = n$ without changing F. Using the definitions of the minimum cut-ratio and the weighted edge expansion it holds that

$$S = \min_{U \subseteq V} \frac{c(U,\bar{U})}{\pi(U) \cdot \pi(\bar{U})} = \min_{U \subseteq V} \frac{c(U,\bar{U})}{c(U) \cdot c(\bar{U})/c(V)} \geq \min_{U \subseteq V} \frac{c(U,\bar{U})}{\min\{c(U), c(\bar{U})\}} = \beta$$

because $c(U) \cdot c(\bar{U})/c(V) \leq \min\{c(U), c(\bar{U})\}$. Furthermore, we have that for any $u \in V$, $\hat{\gamma} = c(u)/\pi(u) = \sqrt{c(V)}$ and $c(V)/p = \sqrt{c(V)}$. Thus, according to Theorem 2.3, there is a solution to the PMFP I_0 such that $L = O((\sqrt{c(V)} \log n)/(nS)) = O((\log n)/S) = O(\beta^{-1} \log n)$ and $f = \Omega(S/\log n) = \Omega(\beta/\log n)$, which implies the desired $F = O(\beta^{-1} \log n)$. □

Next we prove a lemma that together with the previous lemma implies that $F = \Omega(\alpha^{-1})$ and $F = O(\Delta\alpha^{-1} \log n)$. Recall that $\Delta = \max_{v \in V} c(v)$.

Lemma 2.6 *For any network G with expansion α and weighted expansion β and $c_{\min} \geq 1$ it holds that $\alpha^{-1} = \Omega(\beta^{-1})$ and $\alpha^{-1} = O(\Delta\beta^{-1})$.*

Proof. Since for any set of nodes $U \subseteq V$, $|U| \leq c(U) \leq \Delta|U|$, the lemma directly follows from the definitions of α and β. Note, that the requirement $c_{\min} \geq 1$ is needed only for the lower bound part of the lemma. □

It remains to show that the upper and lower bound for F are in general best possible.

Lemma 2.7 *For any α, $1/n \leq \alpha \leq 1/\log n$, there exists a constant degree graph G with n vertices, expansion $\Theta(\alpha)$ and flow number $\Theta(\alpha^{-1})$.*

Proof. We distinguish between two cases. First, $1/n^{1/2} \leq \alpha \leq 1/\log n$. In this case, consider a d-dimensional Butterfly on n' nodes for some n' specified later. We note that $d = \Theta(\log n')$. It is known that this graph has an expansion of $\Theta(1/d)$ and a flow number of $\Theta(d)$ (e.g., [17]). The expansion is $O(1/d)$ due to the fact that its two $(d-1)$-dimensional sub-butterflies have $d \cdot 2^{d-1}$ vertices each but only 2^d edges leaving them. If we replace now every edge by a path of length ℓ, then the number of nodes of the new graph G increases to $n = \ell \cdot n'$ and the expansion decreases to $\alpha = \Theta(1/(d \cdot \ell))$. Furthermore, the flow number increases to $\Theta(d \cdot \ell)$. Hence, for any desired $\alpha, 1/n^{1/2} \leq \alpha \leq 1/\log n$, the graph G with an expansion α can be obtained by setting $\ell = \lfloor \alpha^{-1}/\log n \rfloor$ and $n' = n/\ell$ in the construction above.

Second, $1/n \leq \alpha \leq 1/n^{1/2}$. In this case, consider the grid network with $\lfloor \alpha^{-1} \rfloor$ nodes in one dimension and $n/\lfloor \alpha^{-1} \rfloor$ nodes in the other dimension. It is easy to check that this graph has an expansion of $\Theta(\alpha)$ and a flow number of $\Theta(\alpha^{-1})$. □

Lemma 2.8 *For any $\log n/n^{1-\epsilon} \leq \alpha \leq 1$ where ϵ is an arbitrary positive constant, and any $\Delta \geq 0$, there exists a constant degree network G with n vertices, expansion $\Theta(\alpha)$ and flow number $F = \Theta(\Delta\alpha^{-1}\log n)$.*

Proof. We start with showing this for $\alpha = 1$. Let G' be a constant degree expander, that is, the expansion of G' is constant. Construct out of G' a graph G in which each edge in G' is replaced by a path of length 3. Each middle edge of a path is assigned a capacity of 1, and the border edges have a capacity of Δ. Had all edges been of capacity Δ, the flow number would have been $\Theta(\log n)$. However, since the middle edges have capacity 1, the flow number increases to $\Theta(\Delta \log n) = \Theta(\Delta\alpha^{-1}\log n)$. This result can now be generalized to other values of α by using the same construction as for the butterfly in the proof of Lemma 2.7. □

Combining the lemmata yields Theorem 2.4. □

Previous best results about the approximability of the UFP gave upper bounds of $O(\Delta^2 \cdot \alpha^{-1}\log n)$ [14], and it seems difficult to improve them when using the expansion as a parameter. If an approximation or competitive ratio of $O(F)$ can be proved (and we will indeed prove it), Theorem 2.4 implies much better results, in particular for networks with $F = \Theta(\alpha^{-1})$. Many of the standard networks (e.g., meshes, butterfly, De Bruijn) actually have this property. Hence, the flow number F seems to be more suitable parameter for the UFP than the expansion.

3 Flow Shortening

The main contribution of this section is the following lemma, which proves that for every multicommodity flow solution there is an almost optimal solution consisting of short paths only. Moreover, this short solution can be efficiently

computed using linear programming. For the rest of the paper, we will use the Shortening lemma with $\epsilon = 1$. Most of our upper bounds rely heavily on it.

Lemma 3.1 (Shortening Lemma) *Suppose we are given a network with flow number F. Then, for any $\epsilon \in (0, 1]$ and any feasible solution $\mathcal{S}$ to an instance of the concurrent multicommodity flow problem with a flow value of f, there exists a feasible solution with flow value $f/(1+\epsilon)$ that uses paths of length at most $2 \cdot F(1 + 1/\epsilon)$. Moreover, the flow through any edge e not used by $\mathcal{S}$ is at most $\epsilon \cdot c(e)/(1+\epsilon)$.*

Proof. Let $\mathcal{O}$ denote the set of paths in the solution $\mathcal{S}$ with flow value f and let $\mathcal{O}' \subseteq \mathcal{O}$ consist of all paths from $\mathcal{O}$ that are longer than L, for $L = 2 \cdot F/\epsilon$. We are going to shorten the paths in $\mathcal{O}'$ at the cost of slightly decreasing the satisfied demand of each commodity.

For a path $p \in \mathcal{O}'$ between s_p and t_p, let $a_{p,1} = s_p, a_{p,2}, \cdots, a_{p,L}$ denote its first L nodes and $b_{p,1}, \cdots, b_{p,L-1}, b_{p,L} = t_p$ its last L nodes and let f_p be the size of the flow along p. Then the set $\mathcal{U} = \bigcup_{p \in \mathcal{O}'} \bigcup_{i=1}^{L} \{a_{p,i}, b_{p,i}, f_p\}$ is (a subset of) an instance of the BMFP. By Claim 2.2, there exists a feasible solution $\mathcal{P}$ to $\mathcal{U}$ with flow value at least $1/(2F)$ consisting of paths of length at most $2F$. We are going to combine the initial and final parts of the long paths in $\mathcal{O}'$ with these "shortcuts" in $\mathcal{P}$ to obtain the desired short solution.

First, decrease the flows along all paths $p \in \mathcal{O}$ by a factor of $1/(1+\epsilon)$ so that we have room to accommodate new, short paths for the paths in $\mathcal{O}'$. These short paths are constructed in the following way:

For every path $p \in \mathcal{O}'$, we replace p by L flow systems $S_{p,i}$, $i = 1, \cdots, L$. Each flow system $S_{p,i}$ consists of two parts:

1. the flow paths between $a_{p,i}$ and $b_{p,i}$ in $\mathcal{P}$ corresponding to the request $\{a_{p,i}, b_{p,i}, f_p\}$ from $\mathcal{U}$, now with a flow of $f_p/(L(1+\epsilon))$, and
2. $f_p/(L(1+\epsilon))$ units of flow between $a_{p,1}$ and $a_{p,i}$ along p, and $f_p/(L(1+\epsilon))$ units of flow between $b_{p,i}$ and $b_{p,L}$ along p.

For each i, the length of each path in the subsystem $S_{p,i}$ is at most $L + 2 \cdot F$, and $f_P/(L(1+\epsilon))$ units of flow are shipped along each path system $S_{p,i}$. Summed over all $i = 1 \ldots L$, we have $f_p/(1+\epsilon)$ units of flow between $s_p = a_{p,1}$ and $t_p = b_{p,L}$, which is as high as the original flow through p reduced by $1/(1+\epsilon)$. Hence, we can replace p by the systems $S_{p,i}$ without changing the amount of flow from s_p to t_p.

Now, it holds for every edge e that the flow traversing e due to the paths in $\mathcal{O}$ is at most $c(e)/(1+\epsilon)$, and due to the shortcuts in $\mathcal{P}$ is at most

$$\sum_{p \in \mathcal{P}:\ e \in p} \frac{f_p}{L(1+\epsilon)} \leq \frac{2F}{L(1+\epsilon)} \cdot c(e) = \frac{\epsilon \cdot c(e)}{1+\epsilon} ,$$

since

$$\sum_{p \in \mathcal{P}:\ e \in p} \frac{f_p}{2F} \leq c(e) .$$

Thus, the flows in $\mathcal{O}$ and $\mathcal{P}$ sum up to at most $c(e)$ for an edge e. Therefore, the modification yields a feasible solution satisfying the desired properties. □

4 Offline Algorithms for the UFP

The UFP seems to be much easier with the no-bottleneck assumption. Therefore, we will assume for most of this section that the no-bottleneck assumption is fulfilled (i.e., $c_{\min} \geq d_{max}$; remember also the assumption $d_i \in [0, 1]$), and only in the last subsection we will deal with the UFP without this assumption. We start with an elementary bounded greedy algorithm (elementary BGA) with an approximation ratio of $O(F)$ and show that for directed networks this is essentially the best possible (if nothing is known about $c_{\min}$ apart from $c_{\min} \geq 1$). Then we present a weighted BGA with an approximation ratio of $O(c_{\min}(F^{1/c_{\min}} - 1))$, which is $O(\log F)$ if $c_{\min} \geq \log F$. Finally, we present a simple greedy algorithm for the UFP without the no-bottleneck assumption with an approximation ratio of $O(\sqrt{m})$.

For a request r let $d(r)$ denote the demand of r and for a flow path p let $d(p)$ denote the demand of the request associated with p and f_p be the flow value of p. Note that $d(p)$ might be different from f_p if p belongs to a fractional multicommodity flow solution.

4.1 The elementary BGA

Consider the following *elementary bounded greedy algorithm* (or in short, *elementary BGA*) [9]: Let L be a suitably chosen parameter. Given a request, reject it if there is no feasible flow path of length at most L between its terminal nodes. Otherwise accept it and select any such path for it.

Theorem 4.1 *For any network G with flow number F, the approximation ratio of the elementary BGA with parameter $L = 4 \cdot F$, when run on requests ordered according to their demands starting with the largest, is at most $O(F)$.*

Proof. Let $\mathcal{B}$ denote the set of paths for the requests accepted by the BGA and $\mathcal{O}$ be the set of paths in the optimal (integral) solution of the UFP. By the Shortening Lemma it is possible to modify the optimal solution $\mathcal{O}$ into a (fractional) solution $\mathcal{O}'$ of the same flow value consisting of paths of length at most $4F$ only, at the cost of overloading the capacity of edges by a factor of two.

For a set of paths Q let $||Q|| = \sum_{p \in Q} f_p$, where f_p is the flow value of p. Furthermore, for any $e \in E$ and $p \in \mathcal{O}'$ let $D(e, p) = ||\{q \,:\, q \in \mathcal{B},\ e \in q,\ d(q) \geq d(p)\ \}||$ and let $D(e)$ denote the total capacity of edge e used by paths in $\mathcal{B}$. A path $q \in \mathcal{B}$ is a *witness* for a path $p \in \mathcal{O}'$ if $d(q) \geq d(p)$ and q and p intersect in at least one edge e such that $D(e, p) + d(p) > c(e)$. A key element in our proof will be the following fact.

Fact 4.2 *Every path $p \in \mathcal{O}'$ associated with a request that was not accepted by the BGA must have a witness in $\mathcal{B}$.*

The fact holds, since otherwise the BGA would have been able to accept the request. Now, let $\mathcal{O}_1 \subseteq \mathcal{O}'$ consist of all the paths corresponding to requests accepted by the BGA and let $\mathcal{O}_2 = \mathcal{O}' \setminus \mathcal{O}_1$. Clearly, $||\mathcal{O}_1|| \leq ||\mathcal{B}||$. Let $E' \subseteq E$ denote the set of all edges on which some path from $\mathcal{O}_2$ has a witness in $\mathcal{B}$. By Fact 4.2 each $p \in \mathcal{O}_2$ must have a witness in $\mathcal{B}$. Since the total demand of the paths in $\mathcal{O}_2$ traversing an edge exceeds the capacity of that edge by a factor of at most two, we have $||\mathcal{O}_2|| \leq 2\sum_{e \in E'} c(e)$. For every path $p \in \mathcal{O}_2$ that has a witness $q \in \mathcal{B}$ at edge e it must hold by definition that $D(e,p) + d(p) > c(e)$ and further $D(e,p) \geq d(q) \geq d(p)$. Hence, $2D(e,p) > c(e)$ and therefore $D(e) \geq D(e,p) > c(e)/2$. Thus, $||\mathcal{O}_2|| \leq 4\sum_{e \in E'} D(e) \leq 16F \cdot ||\mathcal{B}||$, because all paths in $\mathcal{B}$ are of length at most $4F$. Thus, altogether $||\mathcal{O}|| \leq (16F+1)||\mathcal{B}||$, which proves the theorem. □

We note that if it is guaranteed that all requests have demands at most 1/2 or all requests have demands at least 1/2, than the algorithm works even without the reordering (which is vital for Fact 4.2 to hold). This will be used for constructions of the online algorithms.

4.2 A general lower bound for directed networks

Next we show that for directed graphs, the approximation ratio obtained by the elementary BGA is essentially the best possible. The proof is by reduction from the NP-hard problem 2DIRPATH: Given a directed graph H and four distinct vertices s_1, t_1, s_2, t_2, are there two edge-disjoint directed paths, one from s_1 to t_1 and the other from s_2 to t_2?

Before we state and prove the result, we first have to adapt the definition of the flow number to directed graphs. For a node u let $c_{out}(u)$ denote the sum of capacities of outgoing edges and $c_{in}(u)$ the sum of capacities of incoming edges. The instance I_0 of the concurrent multicommodity flow we consider in the definition of the flow number has for each oriented pair of nodes (v,w) a single commodity with demand $c_{out}(v) \cdot c_{in}(w)/c(V)$. The flow number of the directed network is the minimum over all feasible solutions $\mathcal{S}$ of I_0 of $\max\{C(\mathcal{S}), D(\mathcal{S})\}$.

Theorem 4.3 *For any $\epsilon > 0$, it is NP-hard to approximate the UFP on directed graphs with n vertices and flow number $F = n^\gamma$, $0 < \gamma \leq 1/2$, within $F^{1-\epsilon}$.*

Proof. The proof is by reduction from the NP-hard problem 2DIRPATH, using the ideas of a construction by Guruswami et al. [7]. Consider any fixed $\epsilon > 0$. Let k denote the number of vertices in a given directed graph H for which the 2DIRPATH problem is to be decided. Furthermore, let G be any constant degree directed graph with $n = k^{1/2\gamma\epsilon}$ vertices with in-degree(v) =out-degree(v) for every vertex v and flow number $F = n^\gamma$ (such a graph certainly exists). We construct out of G and copies of H a directed graph G' with $\Theta(n)$ vertices and flow number $F' = \Theta(n^\gamma)$ in the following way.

Let M be an $l \times l$-mesh with $l = F^{1-\epsilon}$ in which all horizontal edges are oriented to the right and all vertical edges are oriented downwards. Now, replace

every internal node v in M by a copy of H in such a way that the left incoming edge to v is connected to s_1, the right outgoing edge to t_1, and the upper incoming edge to s_2, the lower outgoing edge to t_2. This results in a directed graph M' on $F^{1-\epsilon} \cdot F^{1-\epsilon} \cdot k = F^2$ nodes. Let $a_1, \ldots, a_{l/2}$ denote the first $l/2$ nodes on the highest row of M' and $b_1, \ldots, b_{l/2}$ denote the last $l/2$ nodes on the lowest row of M'. In order to obtain the graph G', we connect G to M' via $l/2$ edges leading to $a_1, \ldots, a_{l/2}$ and $l/2$ edges leaving $b_1, \ldots, b_{l/2}$. The endpoints of these edges in G are chosen in the following way (Figure 1).

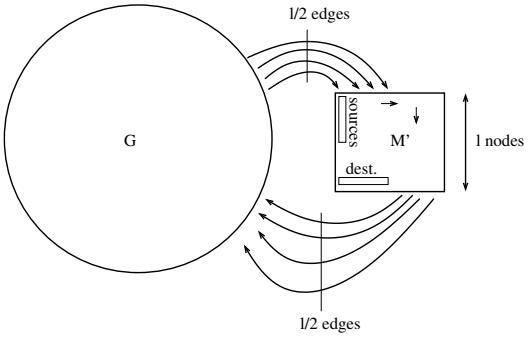

Figure 1: The construction of the graph G'.

Let T be any spanning tree in G. Start at some node v in T and follow the edges of T in an Euler tour. For every $i \in \{1, \ldots, l\}$ let S_i denote the set of vertices visited between $(i-1)F + 1$ and $i \cdot F$ steps of the Euler tour. Since every node is visited at most twice in an Euler tour, a node can appear in at most two different sets S_i. For every $i \in \{1, \ldots, l/2\}$, connect the first node of set S_i to node a_i and the first node of set $S_{l/2+i}$ to node b_i. This results in a directed graph G' with the following property.

Lemma 4.4 *The graph G' has $\Theta(n)$ nodes and a flow number F' with $F' = \Omega(F^{1-\epsilon})$ and $F' = O(F^{1+\epsilon})$.*

Proof. Since the diameter of M' is at least $F^{1-\epsilon}$, the flow number of G' is $\Omega(F^{1-\epsilon})$. It remains to prove the upper bound. Consider the instance I_0 of the BMFP on G'. To show which paths to use for routing of the commodities, we distinguish between three types of commodities:

1. commodities (u, v) with $u, v \in G$,
2. commodities (u, v) with $u, v \in M'$,
3. commodities (u, v) with $u \in G$ and $v \in M'$ or with $u \in M'$ and $v \in G$.

Commodities of type 1 can be easily connected via paths of congestion and dilation at most F, since G itself has a flow number F.

Paths of commodities of type 2 will consist of several parts. First, we connect all of them to the nodes $b_1, \ldots, b_{l/2}$ in such a way that the congestion of these parts in M' is $O(F^{1+\epsilon})$ and for each $i \in \{1, 2, \ldots, l/2\}$, the total demand of paths ending in b_i is $O(F^{1+\epsilon})$. Similarly for the last part of the paths: for each node in M', a path is chosen from one of the nodes $a_1, \ldots, a_l$ in such a way that the congestion of these parts in M' is $O(F^{1+\epsilon})$ and for each $i \in \{1, 2, \ldots, l/2\}$, the total demand of paths going from a_i is $O(F^{1+\epsilon})$. We still have to describe the intermediate part. The point is that after extending every initial part from b_i to the corresponding node in G and from there evenly to the nodes in S_i, and similarly extending backwards each last part from a_i to the corresponding node in G and from there evenly to the nodes in S_i, we are left with an almost balanced instance of the multicommodity flow problem: if we consider the matching pairs of the initial and final parts as separate commodities with their demand equalling the demand of the matching parts, then the demand originating and terminating in any vertex of G is at most $O(F^{\epsilon})$. Thus, by a directed version of Claim 2.2, it is possible to route the intermediate parts with congestion and dilation $O(F^{1+\epsilon})$.

The commodities of type 3 can be dealt with in an analogous way. Putting everything together, we have a way of routing the instance I_0 with congestion and dilation $O(F^{1+\epsilon})$, which completes the proof. □

Since $\Theta(n) = poly(k)$, the size of G' is polynomial in the size of the graph H. Consider the following instance of the UFP: let $s_1, \ldots, s_{l/2}$ be the first $l/2$ nodes of the leftmost column of M' and $t_1, \ldots, t_{l/2}$ be the first $l/2$ nodes of the lowest row of M'. The set of pairs to connect via a path of capacity 1 is given by $\{(s_i, t_i) : 1 \leq i \leq l/2\}$. Since connecting more than two of these pairs would mean to solve the 2DIRPATH problem (cf. [7]), it is NP-hard to distinguish whether there are $l/2 = F^{1-\epsilon}/2$ disjoint paths or just a single one. □

4.3 The weighted BGA

In order to get below the lower bound above for specific instances of the problem (e.g., for high capacity networks), additional parameters apart from F are needed. Recall the definition of the minimum capacity, $c_{\min} = \min_e c(e)$. Since we deal with the no-bottleneck assumption, we have $c_{\min} \geq 1$. We will assume in this subsection that $c_{\min}$ is an integral value. Like other variants of the BGA, also the weighted BGA will process the requests one after the other without any later rearrangements. For an edge e and a request r let $D(e, r)$ denote the load of e after processing all requests before r. Furthermore, let $f_e(x) = F^{\lceil x \rceil / c(e)}$ for any edge e. The *weight* of an edge e before processing request r is defined as $w(e, r) = f_e(D(e, r))$, and the *weight* of a path p for a request r is defined as $w(p, r) = \sum_{e \in p} w(e, r)$. Now we are ready to describe the weighted BGA, which is related to the AAP algorithm by Awerbuch, Azar and Plotkin [1] but uses a simpler cost function that allows it to be implemented in a much more efficient way.

Suppose we have a network G with minimum capacity $c_{\min}$ and flow number

F. The *weighted BGA* works as follows: Let L and W be suitably chosen parameters. Given a request, reject it if there is no flow path p available for it of length at most L and weight at most W. Otherwise, accept it and select any such path for it.

The following theorem shows that the weighted BGA improves exponentially with an increasing $c_{\min}$. When reading the theorem, note that for the special case of $c_{\min} \geq \log F$ it holds $c_{\min} \cdot (F^{1/c_{\min}} - 1) = O(\log F)$. We believe that a similar result can also be shown when using the Shortening Lemma in the analysis of Azar and Regev [2].

Theorem 4.5 *For any network G with minimum capacity $c_{\min}$ and flow number F, the approximation ratio of the weighted BGA with parameters $L = 4 \cdot F$ and $W = 5 \cdot F$, when run on requests in non-increasing order of demands, is $O(c_{\min} \cdot (F^{1/c_{\min}} - 1))$.*

Proof. Let $\mathcal{B}$ denote the set of paths for the requests accepted by the weighted BGA and $\mathcal{O}$ be the set of paths in the optimal (integral) solution to the UFP. By the Shortening Lemma it is possible to modify the solution $\mathcal{O}$ into a (fractional) solution $\mathcal{O}'$ of the same flow value consisting of paths of length at most $4F$ edges only, at the cost of overloading the capacity of edges by a factor of at most 2. Let $\mathcal{O}'_1 \subseteq \mathcal{O}'$ consist of all flow paths whose requests were rejected by the weighted BGA.

We will need a few more definitions. Let the *normalized weight* of an edge e before processing a request r be defined as $\bar{w}(e, r) = d(r) \cdot w(e, r)$ and the *normalized weight of a path* p as $\bar{w}(p, r) = \sum_{e \in p} \bar{w}(e, r)$. For an edge e let $r^e_1, \ldots, r^e_{k_e}$ be all the requests that were accepted by the weighted BGA and routed through e in this order, and let $D(e)$ denote the final load of e, that is, $D(e) = \sum_{i=1}^{k_e} d(r^e_i)$. If there is no danger of confusion we will omit the upper index e. Recall the definition of $D(e, r)$ at the beginning of this subsection.

Consider now any flow path $p \in \mathcal{O}'_1$ and let r be the request associated with it. Then either (a) one of the edges along p, say e, has the property that $D(e, r) + d(r) > c(e)$, or (b) $w(p, r) > W$. This has the following consequences.

(a) Recall that $d(r') \in [0, 1]$ for every request r'. Thus, it must hold at the end that $D(e) > c(e) - 1$. Let the requests contributing to $D(e)$ before r was processed be denoted by $r_1, \ldots, r_k$. We distinguish between two cases. If $d(r) \leq 1/2$, then $D(e, r) > c(e) - 1/2$ and the sum of the normalized weights at e after all requests have been processed is at least

$$\sum_{i=1}^{k} \bar{w}(e, r_i) = \sum_{i=1}^{k} d(r_i) \cdot f_e \left(\sum_{j=1}^{i-1} d(r_j) \right) \geq \sum_{i=0}^{c(e)-2} F^{i/c(e)} + \tfrac{1}{2} F^{(c(e)-1)/c(e)}$$

(using the fact that $f_e \left(\sum_{j=1}^{k-1} d(r_j) \right) = F^{(c(e)-1)/c(e)}$). This is at least

$$\frac{1}{2} \sum_{i=0}^{c(e)-1} F^{i/c(e)} = \frac{F - 1}{2(F^{1/c(e)} - 1)} \, .$$

If $1/2 < d(r) \leq 1$, then only $D(e,r) > c(e) - 1$. However, it is not difficult to check that for every $i \in \{1, \ldots, c(e)-1\}$ there must be a request r_j, $j \in \{1, \ldots, k\}$, such that $i - 1 < D(e, r_j) \leq i$, and for request r, $c(e) - 1 < D(e,r) \leq c(e)$. Since $d(r_j) > 1/2$ for every j (recall that the weighted BGA is run on requests sorted in non-increasing order of their demands), the sum of the normalized weights of e after all requests have been processed is again at least

$$\sum_{i=1}^{k} \bar{w}(e, r_i) \geq \sum_{i=0}^{c(e)-1} \frac{1}{2} \cdot F^{i/c(e)} = \frac{F-1}{2(F^{1/c(e)} - 1)} \, .$$

In both cases, we will use this to assign to p a *relative normalized weight* of

$$\frac{1}{2c(e)} \cdot \frac{F-1}{2(F^{1/c(e)} - 1)} \, .$$

This is at least

$$\frac{F-1}{4c_{\min}(F^{1/c_{\min}} - 1)} \, ,$$

since for all e, $c(e) \geq c_{\min}$ and therefore $c(e) \cdot (F^{1/c(e)} - 1) \leq c_{\min} \cdot (F^{1/c_{\min}} - 1)$. (This follows from the fact that the derivation of the function $f(x) = x \cdot (F^{1/x} - 1)$ is $f'(x) = (1 - \ln F/x)F^{1/x} - 1$ and that $f'(x) < 0$ for all $x > 0$: for $x \leq \ln F$, $f'(x) < 0$ is obviously true; for $x > \ln F$, $F^{1/x} = e^{\ln F/x} = \sum_{i \geq 0} \frac{(\ln F/x)^i}{i!}$ and $\frac{1}{1 - \ln F/x} = \sum_{i \geq 0} (\ln F/x)^i$, therefore $F^{1/x} < \frac{1}{1 - \ln F/x}$.) In the following, let $\gamma(x) = x \cdot (F^{1/x} - 1)$.

(b) Suppose that $p = (e_1, \ldots, e_l)$. Since $w(p,r) > W$, it holds that $\sum_{i=1}^{l} f_{e_i}(D(e_i, r)) > W$. Analogously to the case (a), for every edge e_i, the sum of the normalized weights of the edge e_i after all requests have been processed is at least

$$\sum_{j=0}^{\lceil D(e_i)-1 \rceil} F^{j/c(e_i)} = \frac{F^{\lceil D(e_i) \rceil / c(e_i)} - 1}{F^{1/c(e_i)} - 1} = \frac{f_{e_i}(D(e_i)) - 1}{F^{1/c(e_i)} - 1} \, .$$

Similarly to (a) we assign to p a relative normalized weight (now we sum over all the edges e_i)

$$\sum_{e_i \in p} \frac{1}{2c(e_i)} \cdot \frac{f(D(e_i)) - 1}{2 \cdot (F^{1/c(e_i)} - 1)} \, ,$$

which is again at least (by the definition of $w(p)$)

$$\frac{w(p) - l}{4 \cdot \gamma(c_{\min})} \geq \frac{W - L}{4 \cdot \gamma(c_{\min})} \geq \frac{F}{4 \cdot \gamma(c_{\min})} \, .$$

Recalling where the relative normalized weight of a path comes from (roughly, it is a lower bound on the sum, over all edges e on p, of the sum of the normalized weights on the edge e over all requests accepted on e, multiplied by $\frac{1}{2 \cdot c(e)}$), it follows for every path $p \in \mathcal{O}'_1$ that

$$\frac{F-1}{4 \cdot \gamma(c_{\min})} \leq \sum_{e \in p} \frac{1}{2c(e)} \sum_{i=1}^{k_e} \bar{w}(e, r_i^e)$$

Hence, we get

$$\begin{aligned}
||\mathcal{O}'_1|| = \sum_{p \in \mathcal{O}'_1} f_p &= \frac{4 \cdot \gamma(c_{\min})}{F-1} \sum_{p \in \mathcal{O}'_1} f_p \cdot \frac{F-1}{4 \cdot \gamma(c_{\min})} \\
&\leq \frac{4 \cdot \gamma(c_{\min})}{F-1} \sum_{p \in \mathcal{O}'_1} f_p \cdot \sum_{e \in p} \frac{1}{2c(e)} \sum_{i=1}^{k_e} \bar{w}(e, r_i^e) \\
&\leq \frac{4 \cdot \gamma(c_{\min})}{F-1} \sum_{e \in E} \sum_{p \in \mathcal{O}' : e \in p} \frac{f_p}{2c(e)} \cdot \sum_{i=1}^{k_e} \bar{w}(e, r_i^e) \\
&\leq \frac{4 \cdot \gamma(c_{\min})}{F-1} \sum_{e \in E} \sum_{i=1}^{k_e} \bar{w}(e, r_i^e) \\
&= \frac{4 \cdot \gamma(c_{\min})}{F-1} \cdot \bar{w}(\mathcal{B}) \ ,
\end{aligned}$$

where $\bar{w}(\mathcal{B}) = \sum_{p \in \mathcal{B}} \bar{w}(p, r_p)$ and r_p is the request that was accepted and routed along p by the weighted BGA. From

$$\bar{w}(\mathcal{B}) = \sum_{p \in \mathcal{B}} d(p) \cdot w(p, r_p) \leq \sum_{p \in \mathcal{B}} d(p) \cdot W = W \cdot ||\mathcal{B}|| = 5F \cdot ||\mathcal{B}||$$

it follows that $||\mathcal{O}'_1|| \leq O(\gamma(c_{\min}) \cdot ||\mathcal{B}||)$. Since $||\mathcal{O}' - \mathcal{O}'_1|| \leq ||\mathcal{B}||$, also $||\mathcal{O}'|| \leq O(\gamma(c_{\min})||\mathcal{B}||)$, which concludes the proof. □

4.4 The UFP without the no-bottleneck assumption

In contrast to the previous subsections here we allow $c_{\min} < 1$, that is, demands may be larger than the minimal edge capacity.

Consider the following *careful* BGA: Order the requests according to their demands starting with the largest. Accept a request r if there exists a feasible path p for it such that after routing r the total flow on at most $\sqrt{m}$ edges of p is larger than half of their capacity. We say that the request r uses these edges in their *upper half*. Let $\mathcal{B}_1$ denote the solution we get. Let $\mathcal{B}_2$ denote the solution consisting of only the largest request connected by any path. As our solution we take the maximal of these two, that is $\mathcal{B} = \max(\mathcal{B}_1, \mathcal{B}_2)$.

Theorem 4.6 *The solution of the careful BGA is a* $(6\sqrt{m}+1)$*–approximation.*

Proof. Let $\mathcal{O}$ denote the optimal unsplittable flow and $\mathcal{O}' \subseteq \mathcal{O}$ its subset consisting of requests rejected by both runs of the careful BGA algorithm, that is of requests neither in $\mathcal{B}_1$ nor in $\mathcal{B}_2$. Obviously $||\mathcal{O} - \mathcal{O}'|| \leq 2 \cdot ||\mathcal{B}||$. Consider a path $p \in \mathcal{O}'$. There are two possible reasons why the request r corresponding to p was not routed along p by the careful BGA: either p was infeasible, which means the existence of an edge $e \in p$ where r did not fit in, or there are (at least) $\sqrt{m}$ edges $e_1, \cdots, e_{\sqrt{m}}$ on p that would be used by r in their upper half, that is for each e_i the sum of $d(p)$ and the flow on e_i in the moment of deciding about p was larger than half of their capacity $c(e_i)/2$.

Let us think about the first rejection reason. Since the requests were processed according to their demands, the flow on e in the moment of rejecting p was more than $c(e)/2$. Consider the paths from $\mathcal{B}_1$ participating on this flow that use the edge e in the upper half. Again, due to processing the requests according to their demands, the sum of flow values of these paths is at least $c(e)/4$ (if $d(p) \leq c(e)/4$, then there is less than $c(e)/4$ capacity available in e and therefore the sum of flow values of paths that use e in the upper half is at least $c(e)/4$; if $d(p) > c(e)/4$, then the smallest request routed though e uses e in the upper half, and due to the ordering of the requests, its demand is at least $d(p) > c(e)/4$). Each of these paths q is called a *type I witness* of p and its weight for p is defined as $d(q) \cdot d(p)/c(e)$. Note that the total weight of each path $q \in \mathcal{B}_1$ as a type I witness for paths in $\mathcal{O}'$ is at most $d(q) \cdot \sqrt{m}$ (q serves as a type I witness only on edges that are used by it in the upper half and the number of these is upper bounded by $\sqrt{m}$), and, on the other hand, each path $p \in \mathcal{O}'$ rejected for the first reason has witnesses in $\mathcal{B}_1$ with total weight at least $d(p)/4$. Thus, the total demand of paths rejected for the first reason is at most $4\sqrt{m}||\mathcal{B}_1||$.

If the path $p \in \mathcal{O}'$ was rejected for the second reason then either (a) there are more than $\sqrt{m}/2$ edges on p such that $d(p) > c(e)/2$ for each of them, or (b) there are $\sqrt{m}/2$ edges each with a flow at least $d(p)$. In the former case, the total number of paths in the optimal solution that use more than a half of capacity of more than $\sqrt{m}/2$ edges is less than $2\sqrt{m}$ and their total demand is at most $2\sqrt{m}||\mathcal{B}_2||$. In the latter case, the paths on the $\sqrt{m}/2$ edges are called *type II witnesses* and the weight of q which meets on e_i with p is defined as $d(q) \cdot d(p)/c(e_i)$. The weight of each path $q \in \mathcal{B}_1$ as a type II witness is at most $d(q) \cdot m$ (q can serve as a type II witness on each edge $e \in q$, and the length of q is upper bounded by the number of edges m) and, on the other hand, the total weight of type II witnesses for each path in $\mathcal{O}'$ rejected for the '2b' reason is at least $d(p) \cdot \sqrt{m}/2$. Therefore, by double counting, total demand of paths rejected for the '2b' reason is at most $2\sqrt{m}||\mathcal{B}_1||$. □

Note that the flow number is useless without the no-bottleneck assumption. For example, think about an expander network $G_1(V, E_1)$ on n nodes with all edge capacities equal to $1 - \epsilon$, for some $\epsilon > 0$, and about a mesh $G_2(V, E_2)$ on n nodes with all edge capacities equal to one. The flow number of a network $G(V, E_1 \cup E_2)$ is $O(\log n)$ but if all requests have demands larger than $1 - \epsilon$, they can only make use of the mesh subnetwork with flow number $\Theta(\sqrt{n})$. Thus, the

flow number does not help to get better algorithms in this setting.

5 Online algorithms for the UFP

So far we only presented offline algorithms. Since in real systems requests usually arrive in a continuous fashion, it is important to find also efficient online algorithms. Throughout the section we will assume that the no-bottleneck assumption is true, i.e. $c_{\min} \geq 1$.

Our aim will be to ensure that at the end of any input sequence of requests, the total demand of the connected paths is close to the best possible total demand. That is, we search for algorithms with a competitive ratio that is as small as possible. As a reminder, the competitive ratio of a deterministic online algorithm is defined as

$$c = \sup_{\sigma} \frac{OPT(\sigma)}{ON(\sigma)} ,$$

where the supremum is taken over all possible sequences of requests σ, $ON(\sigma)$ is the profit of the online algorithm on σ, and $OPT(\sigma)$ is the profit of an optimal offline algorithm. In our case, the profit is the sum of all satisfied demands. Similarly, the competitive ratio of a randomized online algorithm is defined as

$$c = \sup_{\sigma} \frac{OPT(\sigma)}{\mathrm{E}[ON(\sigma)]} .$$

5.1 Online algorithms that do not cancel paths

If $c_{\min}$ is an integral capacity of at least 2, then we can use the weighted BGA presented in Section 4.3 to obtain the following result.

Theorem 5.1 *For any network G with capacity $c_{\min} \geq 2$ and flow number F, the competitive ratio of the weighted BGA with parameters $L = 4 \cdot F$ and $W = 5 \cdot F$ and cost function $f_e(x) = F^{\lceil x \rceil/(c(e)-1)}$ is $O(c_{\min} \cdot (F^{1/(c_{\min}-1)} - 1))$.*

Proof. The proof works in the same way as the proof of Theorem 4.5. The difference is that here the requests are not processed according to their demands, which results in a weaker bound (i.e., the power $1/(c_{\min} - 1)$ in the on-line case versus $1/c_{\min}$ in the off-line case). □

The next theorem shows that this upper bound is the best possible by providing a matching lower bound. The proof follows the arguments given by Azar and Regev [2] in their $\Omega(c_{\min} \cdot n^{1/(c_{\min})})$ lower bound. However, since the proof does not appear in the published version of their paper, we provide for the sake of completeness a full description of its improved version here. The trick to improve their lower bound to the lower bound below is to offer an additional request of demand ϵ at the beginning, where $\epsilon > 0$ is sufficiently small. This reduces the usable capacity for the following requests from $c_{\min}$ to $c_{\min} - 1$.

Theorem 5.2 *For all F and all integral $2 \leq c_{\min} \leq \log F$ there is a network G of minimum edge capacity $c_{\min}$ and flow number $\Theta(F)$ such that the competitive ratio of any deterministic online algorithm on G is $\Omega(c_{\min} \cdot F^{1/(c_{\min}-1)})$.*

Proof. We will restrict ourselves to considering a linear array with edge capacities $c_{\min}$ consisting of $n+1$ nodes numbered from 0 to n. Obviously, in this case $F = \Theta(n)$. The general case can simply be obtained by attaching a linear array of F nodes and edge capacities $c_{\min}$ to a network with flow number F and minimal edge capacity $c_{\min}$. Let $k = c_{\min} - 1$. For simplicity we also assume that $n = r^k$ for some integer r.

For any deterministic online algorithm we are going to describe an input sequence on which the given algorithm is $\Omega((k+1) \cdot n^{1/k})$ times worse than the optimal one. For any algorithm, the first request in the sequence will always be a request between 0 and n with a small demand $\epsilon > 0$. Any algorithm with a bounded competitive ratio has to accept it. The rest of the sequence will be described with a help of a complete r-ary tree T of height k: the root is at level 0 and the leaves are at level k. Given any drawing of the tree in the plane, we number the nodes in each level from left to right with integers starting from 0. Now we associate each node of T with a segment of our line graph. A node j in level i, $0 \leq i \leq k$, corresponds to the interval between nodes $j \cdot r^{k-i}$ and $(j+1) \cdot r^{k-i}$. Note that for any non-leaf node of T its children's segments are disjoint (up to the border nodes) parts of its own segment.

The rest of the input sequence (till now we have just the first request) is constructed as follows. All further request will be of demand one. We traverse the tree in a depth first search manner, starting from the root. In each node u we present to the algorithm a sequence of identical requests between end-nodes of the interval corresponding to u until either i) the algorithm accepts one, or ii) we have already presented $k+1$ such requests in this node. In the former case we keep traversing the tree in the DFS order, in the later case we skip all nodes in the subtree of u and then continue in the DFS traversal of the tree. Note that if we happen to arrive to a leaf of the tree during the DFS traversal, the algorithm cannot accept any of the requests presented here since there are already k other accepted requests overlapping with the leaf's corresponding interval plus the ϵ request.

Let $T' \subseteq T$ be a tree consisting of all nodes of T really visited by our traversal of the tree. The profit of the algorithm is equal to the number of inner nodes of T' plus ϵ. On the other hand, a better solution is to accept the $k+1$ requests for each leaf of T'. Since the number of the leaves of T' is $r-1$ times larger than the number of the inner nodes (can be proved by induction) the lower bound follows. □

The performance guarantee of the weighted BGA in the online setting (Theorem 5.1) is asymptotically weaker than its performance guarantee in the off-line setting (Theorem 4.5). However, there are still ways to get better algorithms even in the online setting:

1. To use randomized algorithms. Both the elementary BGA and weighted

BGA can be easily modified into algorithms with the same performance guarantee as their off-line counterparts. This is considered in Theorems 5.3 and 5.4.

2. To allow the on-line algorithm to cancel previously established paths. Subsection 5.2 deals with this setting.

For $c_{\min} = 1$, the *randomized elementary BGA* works as follows, using the same trick as Azar and Regev [2]: With probability 1/2 either consider only requests of demand less than 1/2 or consider only requests of demand at least 1/2. Use the elementary BGA with parameter $L = 4F$ to decide whether to accept or reject requests in the chosen group.

Theorem 5.3 *The randomized elementary BGA has a competitive ratio of* $O(F)$.

Proof. Let $\mathcal{O}$ be the set of paths accepted by the optimal solution. Furthermore, let $\mathcal{O}'$ be the set of paths with demands less than 1/2 and $\mathcal{O}''$ be the set of paths with demands at least 1/2. The result easily follows from the remark after Theorem 4.1. With probability 1/2 the algorithm considers only those requests (either smaller or larger than 1/2) that compose most of the optimal profit and on this subsequence the performance is guaranteed by Theorem 4.1. □

The same separation trick also works when applied to the weighted BGA.

Theorem 5.4 *The randomized weighted BGA has a competitive ratio of* $O(c_{\min} \cdot (F^{1/c_{\min}} - 1))$.

5.2 Online algorithms that cancel paths

In this section we will present online algorithms that are allowed to cancel paths. However, any request whose path has been canceled is not allowed to be reestablished later. Hence, the online algorithms we will consider are still very restricted: the requests arrive one after the other, and for each of them the algorithm has to decide before knowing the next requests in the input sequence whether to accept it or not. If the request is accepted, a flow path has to be provided for it that, in addition to the already established paths, does not exceed the capacity of any edge. To achieve this, the algorithm is allowed to cancel previously connected requests but cannot reconnect them later.

Consider the following online algorithm, called *rude BGA* with parameter L: Given a request of demand d, check whether there is a flow path of value d and length at most L available for it after canceling previously established paths of total flow value at most $d/2$. If so, establish the new request along any of these paths and cancel the corresponding old requests (if necessary). Otherwise, reject the request.

We call paths that get canceled due to a request r *victims* of r. The rude BGA has the following performance.

Theorem 5.5 *The rude BGA with parameter $4F$ has a competitive ratio of $O(F)$.*

Proof. Let $\mathcal{B}$ be the paths used at the end by the rude BGA and $\mathcal{O}$ be the paths used by an optimal offline strategy. For any path p let f_p be its flow value and $d(p)$ be the demand of the request associated with it. For any set of paths Q let $||Q|| = \sum_{p \in Q} f_p$. Let $\mathcal{B}'$ be the set of all flow paths ever selected by the rude BGA, even if they were canceled later on.

Lemma 5.6

$$||\mathcal{B}'|| \leq 2 \cdot ||\mathcal{B}||.$$

Proof. Our strategy for proving the lemma will be to distribute the flow values of the paths in $\mathcal{B}$ in a suitable way among the paths in $\mathcal{B}'$. Suppose that in a first step every path $p \in \mathcal{B}$ moves f_q units of its flow to each of its victims q. This is possible, since the flow value of p exceeds the flow values of its victims by a factor of at least 2. Next, each victim q that got a value of f_q moves a value of $f_{q'}$ to each of its victims q'. This is continued until all paths in $\mathcal{B}'$ have received a flow value. Since the rude BGA ensures that the sum of the flow values of the victims of a path p is at most $d(p)/2$, it is easy to see that the values of the paths in $\mathcal{B}$ are distributed by the above process among the paths in $\mathcal{B}'$ so that every path $q \in \mathcal{B}'$ has a value of at least $d(q)/2$. Thus, $||\mathcal{B}'|| \leq 2 \cdot ||\mathcal{B}||$. □

For an edge $e \in E$ let $D(e)$ denote the sum of flow values of the paths in $\mathcal{B}'$ passing through edge e. A set of flow paths $\{q_1, \ldots, q_k\} \subseteq \mathcal{B}'$ is a *set of witnesses* for a flow path $p \in \mathcal{O}$ if $\sum_i d(q_i) \geq d(p)/2$ and for every i, q_i and p share at least one edge. As in the previous proofs the goal is to show that the requests rejected by the rude BGA but accepted by OPT have enough witnesses in $\mathcal{B}'$ without using each path in $\mathcal{B}'$ too often as a witness.

According to Lemma 3.1 we can assume that all paths in $\mathcal{O}$ have a length of at most $4 \cdot F$ and for every edge e the sum of the demands of paths in $\mathcal{O}$ crossing e is at most $2c(e)$. Let $\mathcal{O}'$ be the set of all paths in $\mathcal{O}$ that do not correspond to requests accepted by $\mathcal{B}'$. Since $||\mathcal{O} \setminus \mathcal{O}'|| \leq ||\mathcal{B}'||$ it remains to bound $||\mathcal{O}'||$.

First note that each $p \in \mathcal{O}'$ must have a set of witnesses in $\mathcal{B}'$ since otherwise the rude BGA would have been able to accept the corresponding request. Let $E' \subseteq E$ denote the set of all edges e on which some path from $\mathcal{O}'$ has a witness in $\mathcal{B}'$ and for which $D(e) \geq c(e)/2$. Let $\mathcal{O}'' \subseteq \mathcal{O}'$ be the set of paths that contain at least one edge from E'. Then

$$||\mathcal{O}''|| \leq \sum_{e \in E'} 2c(e) \leq 4 \sum_{e \in E'} D(e) \leq 16 \cdot F \cdot ||\mathcal{B}'|| \,,$$

because all paths in $\mathcal{B}'$ are of length at most $4 \cdot F$.

For each of the remaining paths $p \in \mathcal{O}' \setminus \mathcal{O}''$ it holds that there must be a set of edges E_p with $d(p) < 2 \sum_{e \in E_p} D(e)$ and $d(p) > c(e) - D(e)$ for all $e \in E_p$ since otherwise the rude BGA would have been able to accept the request corresponding to p. Let $E'' = \bigcup_{p \in \mathcal{O}' \setminus \mathcal{O}''} E_p$ be the set of all of these

edges. Since for every $p \in \mathcal{O}' \setminus \mathcal{O}''$ we have $D(e) < c(e)/2$ for all $e \in E_p$ it holds that $d(p) > c(e)/2$ for all of these edges. Thus,

$$\begin{aligned} ||\mathcal{O}' \setminus \mathcal{O}''|| &= \sum_{p \in \mathcal{O}' \setminus \mathcal{O}''} f_p = \sum_{p \in \mathcal{O}' \setminus \mathcal{O}''} \frac{f_p}{d(p)} \cdot d(p) \\ &< \sum_{p \in \mathcal{O}' \setminus \mathcal{O}''} \frac{f_p}{d(p)} \cdot 2 \sum_{e \in E_p} D(e) = 2 \sum_{e \in E''} D(e) \sum_{p \in \mathcal{O}' \setminus \mathcal{O}'':\, e \in E_p} \frac{f_p}{d(p)} \\ &< 4 \sum_{e \in E''} D(e) \sum_{p \in \mathcal{O}' \setminus \mathcal{O}'':\, e \in E_p} \frac{f_p}{c(e)} \leq 8 \sum_{e \in E''} D(e) \leq 32 \cdot F \cdot ||\mathcal{B}'|| \,. \end{aligned}$$

Therefore, $||\mathcal{O}'|| \leq 48 \cdot F \cdot ||\mathcal{B}||$, which completes the proof. □

Next we show that for the case that $c_{\min}$ is known and $c_{\min} > 1$, a better competitive ratio can be achieved when using the following *weighted rude BGA*: Let L and W be suitably chosen parameters. Given a request r, accept it there is a flow path for r of length at most L and weight at most W, with a possible cancellation of old paths with total weight at most $W/2$. Otherwise, reject it.

Theorem 5.7 *For any network G with $c_{\min} > 1$ and flow number F, the competitive ratio of the weighted rude BGA with parameters $L = 4 \cdot F$ and $W = 5 \cdot F$ is $O(c_{\min} \cdot (F^{1/c_{\min}} - 1))$.*

Proof. The proof is basically a combination of the proofs of Theorem 4.5 and Theorem 5.5: First, it is shown that the weighted rude BGA ensures that $\bar{w}(\mathcal{B}') \leq 2\bar{w}(\mathcal{B})$. Then $\mathcal{B}'$ (or actually the highest total demand each edge ever reaches during the algorithm; all other requests can be neglected) is compared with $\mathcal{O}$ and it is shown that $||\mathcal{O}'|| \leq \frac{4 \cdot \gamma(c_{\min})}{F-1} \cdot \bar{w}(\mathcal{B}')$.

Given a flow path p associated with a request r that was accepted by the optimal algorithm but not by the weighted rude BGA, cases (a.1) and (b) from the proof of Theorem 4.5 go through as before. The only problematic case is (a.2), i.e. $1/2 \leq d(r) \leq 1$, namely the situation when the minimum weight of paths that have to be canceled in order to be able to route the request r along the path p, exceeds half of the weight of p while the weight of p is still at most W. Recall that the weight of a path q is $w(q) = \sum_{e \in q} w(e)$. Let $v(e)$ denote the total weight of the paths passing through an edge $e \in p$ that would have had to be canceled in order to accept r along p. Since r was rejected, it must hold that $w(p) \leq 2 \sum_{e \in p} v(e)$. Hence, there must exist an $e \in p$ with $v(e) \geq w(e)/2$. Let Q be the set of paths corresponding to $v(e)$. Since $d(r) \leq 1$, all paths $q \in Q$ must have a normalized weight of at least $d(q) \cdot f(c(e) - 1)$. Hence, all paths $q \in Q$ together must have a normalized weight of at least $v(e) \cdot f(c(e)-1) \geq (w(e)/2) \cdot f(c(e)-1) \geq f(c(e)-1)/4$ (recall that $w(e) \geq 1/2$). This allows to show in a similar way to (a).2 that the total normalized weight of all paths in e is at least $\frac{F-1}{4(F^{1/c(e)}-1)}$. Thus, the analysis goes through as before. □

6 Open Problems

In this paper we have made a significant advance in proving better bounds on the approximation ratio and the competitive ratio of algorithms for the UFP. However, many problems remain open. For instance, are there lower bounds on the approximation ratio for undirected graphs that are close to those for directed graphs? Is the Shortening Lemma essentially best possible in a sense that any rearrangement to short paths does cause a decrease in the flow value? Can constant factor approximation schemes also be found for $c_{\min} = o(\log F)$? Also, although the presented algorithms substantially improve the previous upper bounds, they still do not make use of the fact that all the paths in the optimal solution for the UFP have to be *unsplittable*. In fact, they only compare the offline or online solution with an optimal fractional solution (and the fractional solution may be significantly larger - by a $\sqrt{m}$ factor on the brick wall). Can the unsplittability be exploited in the analysis to obtain better bounds?

Acknowledgements. We would like to thank the anonymous referees for valuable comments and suggestions.

References

[1] B. Awerbuch, Y. Azar, and S. Plotkin. Throughput-competitive on-line routing. In *Proceedings of the 34th Annual IEEE Symposium on Foundations of Computer Science*, pages 32–40, 1993.

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

[3] A. Baveja and A. Srinivasan. Approximation algorithms for disjoint paths and related routing and packing problems. *Mathematics of Operations Research*, 25, 2000.

[4] A. Chakrabarti, C. Chekuri, A. Gupta, and A. Kumar. Approximation algorithms for the unsplittable flow problem. In *Proceedings of the 5th International Workshop on Approximation Algorithms for Combinatorial Optimization*, 2002.

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

[6] Y. Dinitz, N. Garg, and M. Goemans. On the single source unsplittable flow problem. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, pages 290–299, 1998.

[7] 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. In *Proceedings of the 31st ACM Symposium on Theory of Computing*, pages 19–28, 1999.

[8] M. Hajiaghayi and F. T. Leighton. On the max-flow min-cut ratio for directed multicommodity flows. Technical Report MIT-LCS-TR-910, MIT Laboratory for Computer Science, 2003. Submitted to Theoretical Computer science.

[9] J. Kleinberg. *Approximation Algorithms for Disjoint Paths Problems*. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1996.

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

[11] S. G. Kolliopoulos and C. Stein. Improved approximation algorithms for unsplittable flow problems. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science*, pages 426–435, 1997.

[12] S. G. Kolliopoulos and C. Stein. Approximating disjoint-path problems using greedy algorithms and packing integer programs. In *Proceedings of the 6th Integer Programming and Combinatorial Optimization Conference*, volume 1412 of *Lecture Notes in Computer Science*, pages 153–162, 1998.

[13] P. Kolman. A note on the greedy algorithm for the unsplittable flow problem. *Information Processing Letters*, 88(3):101–105, 2003.

[14] P. Kolman and C. Scheideler. Simple, routing-based on-line algorithms for maximum disjoint paths problem. In *Proceedings of the 13th Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 38–47, 2001.

[15] F. T. Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays - Trees - Hypercubes*. Morgan Kaufmann, San Mateo, 1992.

[16] T. Leighton and S. Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM*, 46(6):787–832, Nov. 1999.

[17] C. Scheideler. *Universal Routing Strategies for Interconnection Networks*. Lecture Notes in Computer Science 1390, Springer Verlag, 1998.

[18] A. Srinivasan. Improved approximations for edge-disjoint paths, unsplittable flow, and related routing problems. In *38th Annual Symposium on Foundations of Computer Science*, pages 416–425, 20–22 Oct. 1997.