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

October 22, 2001

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

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 study 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. For this we introduce a new graph parameter, the *flow number*. 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, generalizing a well-known theorem of Leighton and Rao [16]. Both the parameter and the method may therefore be of independent interest. They allow us to prove upper bounds on the approximation ratio and competitive ratio of algorithms for the UFP that are significantly below all previous upper bounds. For example, if $c_{min} \geq 1$ is the minimum capacity of an edge and the maximum demand of a request is at most 1, then we achieve an approximation ratio of $O(c_{\min}((\Delta\alpha^{-1}\log n)^{1/c_{\min}} - 1))$, where Δ is the maximum degree, α is the expansion, and n is the number of nodes in the network, whereas the best previous result was an approximation ratio of $O(c_{\min} \cdot (n^{1/c_{\min}} - 1))$. For networks with high expansion (such as the hypercube, the butterfly, or any expander), this is an exponential improvement. We also show how to transfer these results to the online setting.

1 Introduction

In the *unsplittable flow problem*, denoted by UFP, we are given a directed or undirected network $G = (V, E)$ with edge capacities prescribed by $c : E \rightarrow \mathbb{R}_+$ 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.

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 [6]. 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 [15, 7, 21, 11, 12, 5, 6, 16, 3, 13]. Kleinberg [7] provides a comprehensive background on these problems. Most 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 presented an $O(\sqrt{m} \cdot \frac{\max c(e)}{\min d_i})$ approximation algorithm for the UFP [7], where m is the number of edges. This was subsequently improved by Baveja and Srinivasan [3] to $O(\sqrt{m})$. On the lower bound side, it was shown by Guruswami et al. [6] that on directed networks the UFP is NP-hard to approximate within a factor of $m^{1/2-\epsilon}$ for any $\epsilon > 0$. For a special case of the UFP, the unit-capacity UFP (that is, all edges have a capacity 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, α is the expansion, and n is the number of nodes in the network. Using the techniques of Kleinberg and Rubinfeld [9] and results of Leighton and Rao [16] this can be decreased to $O(\Delta^2\alpha^{-2}\log^2 n)$. Recently, Kolman and Scheideler [13] improved this ratio to $O(\Delta^2\alpha^{-1}\log n)$. Using a new parameter called the *flow number* F of a network, we improve the ratio further to $O(F)$ with $F = O(\Delta\alpha^{-1}\log n)$. For this we use a simple greedy algorithm. Redefining Δ as the maximum total capacity leaving or leading to a node, the bound can be extended to arbitrary undirected networks that fulfill the no-bottleneck assumption.

Without the no-bottleneck assumption, Guruswami et al. [6] 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 $O(\sqrt{m})$ without using any assumption about the ratio of the edge capacities and the demands. The bound is again achieved with a simple greedy algorithm.

In several papers also the minimum edge capacity (denoted here by $c_{\min}$) was used as a parameter for measuring the performance of algorithms [1, 3, 4, 8, 10, 18, 21]. Based on techniques of Awerbuch et al. [1], Azar and Regev [2] recently obtained a polynomial time algorithm with an approximation ratio of $O(c_{\min}\cdot n^{1/c_{\min}})$ for any $2 \le c_{\min} \le \log n$. They note that the n in the bound can be replaced by an upper bound on the longest path in an optimal solution, however, without giving any better bound on this value other than the trivial n. We significantly 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 $c_{\min}\ge 1$ (which is $O(\log F)$ if $c_{\min}\ge \log F$). Since $F = O(\Delta\alpha^{-1}\log n)$, this also implies a competitive ratio of $O(c_{\min}\cdot((\Delta\alpha^{-1}\log n)^{1/c_{\min}}-1))$. Our key technique to obtain this result, which may be of general interest, is the so-called *Shortening Lemma*. 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 by Leighton and Rao [16, Theorem 18], to a result about short flow solutions for *arbitrary* multicommodity flow problems. For $c_{\min} \geq \log n$, Raghavan and Thompson [18] described a constant factor approximation. With the help of advanced rounding techniques [20] we show that already for $c_{\min} \geq \gamma \log F$ for a sufficiently large constant γ there are constant factor approximation algorithms for the UFP.

The UFP has also been considered in the online setting where the requests arrive one by one and decisions have to be made without knowing 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. 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 that does not cancel paths has a competitive ratio of $\Omega(c_{\min} \cdot (F^{1/(c_{\min}-1)} - 1))$. We also show that if it is allowed to cancel 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.

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. [6], 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 [19] 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 $F = O(\Delta \cdot \alpha^{-1} \log n)$.

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 approximation and 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 cost function $c : E \to \mathbb{R}_+$ denoting the *capacities* of the edges and $c_{\min} = \min_{e \in E} c(e)$ is called the *minimum edge capacity* of G. Unless explicitly mentioned, we will assume that G is undirected. (However, all of our results hold also for directed graphs satisfying $\sum_{(v,w) \in E} c(v, w) = \sum_{(w,v) \in E} c(w, v)$ for every $v \in V$, but 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. 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 $\Gamma = C(V)$ 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

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

In a *concurrent multicommodity flow problem* there are k commodities, each with two terminal nodes s_i and t_i and demand d_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 value 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 in which the sum of the demands of the commodities originating and the commodities terminating in a node v is equal to $c(v)$ for every $v \in V$. 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 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 [19] 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 $\pi(v) = c(v)/\sqrt{\Gamma}$ for every node v, that is, each pair of nodes (v, w) has a commodity of demand $c(v) \cdot c(w)/\Gamma$. 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 [14] and Scheideler [19].) 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 levelled 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 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 $c(v_k) \cdot c(v_\ell)/\Gamma(G)$. 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 $E_{k,\ell}$ is at most $c(\{v_k, v_\ell\})$. Let the *congestion* $C(\mathcal{S})$ of $\mathcal{S}$ be defined as the inverse of its flow value. If we allow fractional flows, then it is easy to compute via linear programming a solution $\mathcal{S}$ in polynomial time that minimizes $F_L = \max\{C(\mathcal{S}), L\}$.

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

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 $d_{i_u} = d_i \cdot c(u)/\Gamma$, and the second problem I_2 has commodities i'_u from u to t_i for all $u \in V$ with demands $d_{i'_u} = d_i \cdot c(u)/\Gamma$. 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 commodity i shipped to u in I_1 is equal to the amount of commodity i shipped from u in I_2.

Both of the flow problems I_1 and I_2 are PMFPs with $\pi(v) = c(v)/\sqrt{\Gamma}$ 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

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

and in I_2 is equal to

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

Also, both I_1 and I_2 are BMFPs because for every node $v \in V$ its total demand is $c(v)$. 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 $p = \sum_{u \in V} \pi(u)$. The *min-cut of a PMFP* is defined as

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

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 length at most $L = O(\hat{\gamma} \log p / pS)$. Here, $\hat{\gamma}$ is the maximum value over the nodes with nonzero weight of the capacity of the node divided by the weight of the node, and the path length is defined as the number of nodes of nonzero weight in the path.*

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

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

Without loss of generality we consider only networks with a minimum edge capacity of $c_{min} = 1$. Using Theorem 2.3, we prove the following result.

Theorem 2.4 *For any network G with expansion α 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, in general the upper and lower bounds are the best possible.

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. The fact that $F = \Omega(\beta^{-1})$ follows directly from the definition of F and β. Hence, it only remains to show that $F = O(\beta^{-1} \log n)$.

Consider the PMFP I_0 and let S be the min-cut for it. Recall the assumption $p = \sum_{u \in V} \pi(u)$ which implies $n = \sum_{u \in V} \pi(u) = \sum_{u \in V} c(u)/\sqrt{\Gamma} = \sqrt{\Gamma}$ in the case of I_0. From the definitions of the weighted expansion β of G and the min-cut S for the I_0 we infer $\beta \leq S$. For the PMFP I_0 we also have $\bar{\gamma} = c(u)/\pi(u) = \sqrt{\Gamma}$.

According to the Theorem 2.3 there is a solution to the PMFP I_0 such that $L = O(\sqrt{\Gamma} \frac{\log n}{nS}) = O(\beta^{-1} \cdot \log n)$ and $f = \Omega(\frac{S}{\log n}) = \Omega(\frac{\beta}{\log n})$ 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 we assume $c_{\min} = 1$.

Lemma 2.6 *For any network G with expansion α and weighted expansion β 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. $\square$

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 of size n with 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., [19]). The expansion is $O(1/d)$ due to the fact that its two $(d-1)$-dimensional subbutterflies have a size of $d \cdot 2^{d-1}$ 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 $\Theta(\alpha^{-1})$. $\square$

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 of size n with 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 $O(\Delta^2 \cdot \alpha^{-1} \log n)$ [13], 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 $2 \cdot F(1 + 1/\epsilon)$.*

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 $i = 1 \dots L$, 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 $c_{\min} \geq \log F$. After this we show that when using advanced LP rounding techniques, for $c_{\min} = \Omega(\log F)$ even a constant factor approximation ratio can be achieved. 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*) [7]: 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 best possible. For this 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 $c_{out}(v) \cdot c_{in}(w)/\Gamma$. 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 flow number $F = n^\gamma$, $0 < \gamma \leq 1/2$, within $F^{1-\epsilon}$.*

Proof. Basically, the proof is by reduction from the NP-hard problem 2DIRPATH, using the ideas of a construction by Guruswami et al. [6]. Consider any fixed $\epsilon > 0$. Let k denote the size of a given directed graph H for which the 2DIRPATH problem is to be decided. Furthermore, let G be any directed graph of size $n = k^{1/2\gamma\epsilon}$ with flow number $F = n^\gamma$. We construct out of G and copies of H a directed graph G' with 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_2, 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 (see also Figure 1). 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.

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' of $\Theta(F)$.*

Proof. Since the flow number of G is F, the flow number of G' is $\Omega(F)$. The task is to prove the upper bound. Consider the instance I_0 of the BMFP on G'. All commodities that have both endpoints in G can be connected via paths of congestion and dilation at most F. For all pairs that have both endpoints in M', we basically distribute the starting parts of the paths evenly among the nodes $b_1, \ldots, b_{l/2}$ and from there evenly among the nodes in each set $S_{l/2+1}, \ldots, S_l$. Furthermore, the ending parts of the paths are evenly distributed among the nodes $a_1, \ldots, a_{l/2}$ and from there evenly among the nodes in each set $S_1, \ldots, S_{l/2}$. At this stage we are left with routing another instance of BMFP in G in order to connect the starting parts with the ending parts of the paths which can be done by Claim 2.2 with congestion and dilation $2F$. The remaining pairs can be handled similarly. □

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. [6]), it is NP-hard to distinguish whether there are $l/2 = F^{1-\epsilon}/2$ disjoint paths or just a single one. □

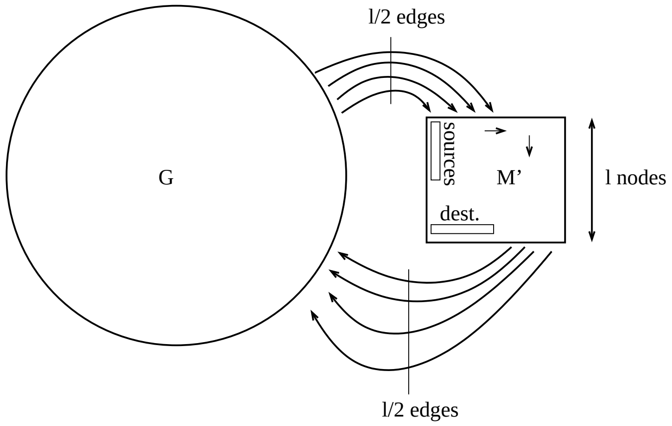

Figure 1: The construction of the graph G'.

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 the following 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 ordered according to their 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 $r_1^e, \ldots, r_{k_e}^e$ 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, $D(e) = \sum_{i=1}^{k_e} d(r_i^e)$. 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) We distinguish between two more cases. If $d(r) \leq 1/2$, then $D(e) > c(e) - 1/2$ and the sum of the normalized weights $\bar{w}(e, r_i)$ of e after all requests have been processed is equal to

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

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

If $1/2 < d(r) \leq 1$, then only $D(e) > c(e) - 1$. It is easy to check that for every $i \in \{1, \dots, c(e) - 1\}$ there must be a request r_j, $j \in \{1, \dots, k\}$, such that $i - 1 < D(e, r_j) \leq i$. Since $d(r_j) > 1/2$ for every j (recall that the weighted BGA is run on requests ordered according to 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 $c(e) \cdot (F^{1/c(e)} - 1) \leq c_{\min} \cdot (F^{1/c_{\min}} - 1)$. In the following, let $\gamma(x) = x \cdot (F^{1/x} - 1)$.

(b) Suppose that $p = (e_1, \dots, e_l)$. 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

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

Similar 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

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

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. $\square$

4.4 A constant factor approximation algorithm

Next we present a randomized polynomial time algorithm that achieves a constant factor approximation ratio for the UFP if $c_{\min} = \Theta(\log F)$ or more. Our basic approach will be to first find an optimal (fractional) multicommodity flow solution to the UFP, i.e. a solution that maximizes the sum of the satisfied parts of the demands. Afterwards, our aim will be to round it to an integral solution while losing only a constant in the total satisfied demand and obeying the capacities of all edges. Our techniques to achieve this will be based on an improved version [20, Theorem 3.84] of an algorithm by Molloy and Reed [17] for the Lovász Local Lemma.

Theorem 4.6 *There is an algorithm that provides in polynomial time a constant factor approximation for every UFP on a network with flow number F and minimum capacity $c_{\min} \geq \gamma \log F$, where γ is a sufficiently large constant.*

Proof. Given a UFP for some network $G = (V, E)$, our strategy to obtain a constant factor approximation algorithm for the UFP works in four steps: finding an optimal fractional solution, initial rounding, intermediate rounding, and final rounding. We will assume here that $\log F = o(\log m)$, since the case that $\log F = \Theta(\log m)$ can be solved by using standard randomized rounding techniques (see [18]). Furthermore, we will assume that F is beyond some sufficiently large constant (if F is smaller, we just have to increase γ in a sufficient way).

1. Optimal fractional solution. First, we obtain an optimal fractional solution by computing a (fractional) multicommodity flow that maximizes the sum of the satisfied fractional demands of the requests, i.e. $\sum_i d'_i$, where $d'_i \le d_i$ is the amount of the demand satisfied for request i. Afterwards, we reduce all d'_i by a fixed constant δ to ensure that the constant increase in the flow caused by the rounding process afterwards does not violate any edge capacities. In the following, d'_i will always mean the current demand routed for request i.

2. Initial rounding. Next we transform the solution into a solution with paths of length at most $4F$ using the Shortening Lemma. Afterwards, for each edge e, $D(e) \le c(e)/\delta'$ for some constant $\delta' \ge \delta/2$ (recall that $D(e)$ is the total flow crossing e). Then, cut each edge $e \in E$ with capacity more than $2c_{\min}$ into a set of edges E_e of capacities in $[c_{\min}, 2c_{\min}]$ with $\sum_{f \in E_e} c(f) = c(e)$. Since every request has a demand of at most 1, we only have to consider edges of capacity up to the total number of requests. Hence, $\sum_e |E_e|$ is still polynomial in the size of the UFP instance. We move the demands from e to E_e so that for every $f \in E_e$ the same constant fraction of $c(f)$ is used.

Now, let $v_0 = \log m$ and let $v_{j+1} = (\log v_j)^3$ for all $v_j > 0$. For simplicity we assume for the rest of the proof that the following condition is true:

(*) for every v_j and d'_i with $1/v_j \le d'_i$, d'_i is a multiple of $1/v_j$

We can assume this w.l.o.g., since rounding down each d'_i to the nearest multiple of $1/v_j$ for $j = 0$, then $j = 1$, then $j = 2$ and so on as long as $1/v_j \le d'_i$, only decreases d'_i by some constant factor and therefore will not affect our rounding strategies below.

Next we use standard randomized rounding [18] to obtain a multicommodity flow with the property that for all requests i with $d_i \le 1/\log m$ either a single flow path of value d_i is used or the request is rejected, and for all other requests i, fractional flow paths with value $1/\log m$ are used while keeping a total flow of approximately d'_i. More precisely, the random experiments are the following:

1. For each request i with $d'_i > 0$ and $d_i \leq 1/\log m$, decide with probability d'_i/d_i whether to increase its demand to d_i and if so, select exactly one of its fractional paths, giving a flow path of value f a probability of f/d'_i of being selected. Otherwise, remove all of its flow paths from the system.

2. For every other request i with $d'_i > 0$, we distinguish between two cases.

   (a) If $d'_i < 1/\log m$, round it to a value of 0 or $1/\log m$ in the same way as in (1).

   (b) Otherwise, cut the fractional path system of the request into systems with flow value exactly $1/\log m$ (here we use condition $(*)$) and round each of them independently to a single path of value $1/\log m$, giving a flow path of value f a probability of $f \log m$ of being selected.

Simple Chernoff bounds (see [20] for a whole collection of them) suffice to show that after this random experiment we have the following situation with high probability:

- every request i with a remaining flow path either has a single flow path of value d_i if $d_i < 1/\log m$ or otherwise flow paths of value $1/\log m$,
- all flow paths have a length of at most $4F$,
- the total flow over an edge has increased by at most some additive $O(\log F)$, but is still at least some $\Theta(\log F)$ value below the capacity of the edge (if γ and δ are sufficiently large), and
- the total amount of flow in the system has decreased by at most some constant factor.

All requests i with a single flow path of value d_i will stick to their paths for the rest of the rounding and therefore will not be considered in the following.

3. Intermediate rounding. The intermediate rounding now works as follows:

$v_0 = \log m$
$i = 0$
while $v_i > F$ do
    $v_{i+1} = (\log v_i)^3$
    $\textsc{Round}(v_{i+1})$
    $i = i + 1$

Algorithm $\textsc{Round}(v)$ has the task to round the solution so that afterwards

- every request with $d_i < 1/v$ either has a single path of flow value d_i or no path any more,
- every request with $d_i \geq 1/v$ has only flow paths of value exactly $1/v$ (here we will use the condition $(*)$),
- the flow value in each edge increased by at most some additional $O((\log F)/\sqrt[3]{v})$, and
- the total flow value in the system decreased by at most some factor of $1 - O(1/v)$.

Suppose that these requirements can be fulfilled. Then, at the end of the intermediate rounding,

- every request with $d_i < 1/F$ has a single path of flow value d_i or no path any more,
- every request with $d_i \geq 1/F$ has only flow paths of value at least $1/F$,
- the flow value in each edge increased by at most an additional $\Theta(\log F)$ factor throughout the whole rounding process, and
- the total flow value in the system decreased by at most some constant factor compared to the situation before the intermediate rounding.

This would allow us to perform some final rounding. But first we explain how $\textsc{Round}(v)$ can fulfill its task.

We start with some additional notation. Let $\mathcal{T} = \{t_1, \ldots, t_n\}$ be a set of independent random trials, and let $\mathcal{A} = \{A_1, \ldots, A_m\}$ be a set of events such that each A_i is determined by the outcome of the trials in $T_i \subseteq \mathcal{T}$. We say that T_i *intersects* T_j and A_i *intersects* A_j if $T_i \cap T_j \neq \emptyset$.

For any $t_{j_1}, \ldots, t_{j_k} \in T_i$ and any $w_{j_1}, \ldots, w_{j_k}$ in the domains of $t_{j_1}, \ldots, t_{j_k}$, we define $\Pr^*[A_i \mid t_{j_1} = w_{j_1}, \ldots, t_{j_k} = w_{j_k}]$ to be the probability of A_i conditional on the event that the outcomes of $t_{j_1}, \ldots, t_{j_k}$ are $w_{j_1}, \ldots, w_{j_k}$. We sometimes just say $\Pr^*[A_i]$ if it causes no ambiguity, always meaning $t_{j_1}, \ldots, t_{j_k}$ to be the set of trials already carried out and $w_{j_1}, \ldots, w_{j_k}$ to be their outcomes. For $k = 0$, we have $\Pr^*[A_i] = \Pr[A_i]$. The following theorem will be applied (see [20, Theorem 3.84]):

Theorem 4.7 ([20]) *If we have the following:*

1. *for each* $1 \leq i \leq m$, $\Pr[A_i] \leq p$;
2. *each* T_i *intersects less than* d *other* T_j*'s;*
3. *for each* $1 \leq i \leq m$, $|T_i| \leq w$;
4. $p \cdot d^9 \leq (1/2\mathrm{e})^3$;
5. $p \cdot w \leq 1$;
6. *for each* $1 \leq j \leq n$, *we can carry out the random trial in time* τ_1;
7. *for each* $1 \leq i \leq m$, $t_{j_1}, \ldots t_{j_k} \in T_i$ *and* $w_{j_1}, \ldots, w_{j_k}$ *in the domains of* $t_{j_1}, \ldots t_{j_k}$, *we can compute* $\Pr^*[A_i]$ *in time* τ_2;

then there is a randomized $O(n \cdot d \cdot (\tau_1 + \tau_2) + n(\tau_1 \cdot \log^{O(1)} m + 2^{(\log\log m)^{O(1)}}))$*-time algorithm which will find outcomes for* $t_1, \ldots, t_n$ *such that none of the events in* $\mathcal{A}$ *holds.*

We say that a request i *participates* in $\textsc{Round}(v)$ if $d'_i > 0$ but it still does not have a single path of flow value d_i. For $\textsc{Round}(v)$ we will use the following random experiment:

1. For each participating request i with $d_i \leq 1/v$, decide with probability d'_i/d_i whether to increase its demand to d_i and if so, select exactly one of its fractional paths, giving a flow path of value f a probability of f/d'_i of being selected. Otherwise, remove all of its paths from the system.

2. For every other participating request i, we combine its flow paths into sets with flow value of exactly $1/v$ (this is ensured by condition $(*)$). Each of the sets is rounded independently of the other. For each set, exactly one of its paths is selected and its flow value increased to $1/v$, giving a flow path of value f a probability of $v \cdot f$ of being selected.

Expressing this in the framework of trials and events, each trial t_i represents the outcome for a single set of flow paths of a participating request and each event A_i represents the bad event that the flow value in an edge increases by more then some additional $O((\log F)/\sqrt[3]{v})$. In addition, we use events B_i to ensure that after the rounding the total flow in the system decreased by at most a factor of $1 - O(1/v)$. To achieve this, we group the flow paths into sets S_i with a total value in $[v^3(1/v), 2v^3(1/v)]$. If this is not possible, then there can be at most $v^3 \cdot 2^{\sqrt[3]{v}}$ flow paths left at the beginning of $\textsc{Round}(v)$ (note that for $v_{i+1} = v$ it holds that $v_i = 2^{\sqrt[3]{v}}$). In this case, a single performance of the random experiment described above suffices to achieve the objectives with sufficiently high probability (as one can see below when we determine the probabilities). Hence, we can assume w.l.o.g. that there are at least $v^3 \cdot 2^{\sqrt[3]{v}}$ flow paths left at the beginning of $\textsc{Round}(v)$. For each S_i, the expected amount of flow that will remain in S_i is in the interval $[v^3(1/v), 2v^3(1/v)]$. B_i now represents the bad event that the total amount of flow left in S_i after performing the random experiment above is less than $(1 - O(1/v))$ times the expected value. If it can be ensured that none of the events A_i and B_i hold, then it is easy to check that all requirements for $\textsc{Round}(v)$ are fulfilled. Hence, it remains to show with the help of Theorem 4.7 that it is possible to find outcomes for the random experiments in polynomial time so that none of the events is fulfilled.

We start with bounding the probability for the events A_i and B_i to occur. One can easily show via Chernoff-Hoeffding bounds [20] that

$$\Pr[A_i] \le \mathrm{e}^{-(\alpha/\sqrt[3]{v})^2 \cdot \log F/(3/v)} \le \mathrm{e}^{-\alpha^2 \sqrt[3]{v} \log F/3}$$

for a constant $\alpha > 0$ that can be made as large as required (as long as γ and δ are sufficiently large). Furthermore, one can show that

$$\Pr[B_i] \le \mathrm{e}^{-(\beta/v)^2 \cdot v^3/2} = \mathrm{e}^{-\beta^2 \cdot v/2}$$

for a constant $\beta > 0$ that can be made sufficiently large. Since $\sqrt[3]{v} \ge \log F$, we can therefore set $p = \mathrm{e}^{-\alpha^2 \sqrt[3]{v} \log F}$ as an upper bound for both $\Pr[A_i]$ and $\Pr[B_i]$.

Next we bound the dependencies between the events. For this we consider three cases:

1. Dependencies among A_i's: Since the edge associated with A_i is used by at most $O(\log F \cdot 2^{\sqrt[3]{v}})$ flow paths, each of these flow paths belongs to a rounding set of at most $2^{\sqrt[3]{v}}$ flow paths, and every flow path has a length of at most $4F$, an A_i can intersect at most $O(F \log F \cdot 2^{\sqrt[3]{v}})$ many other A_j's.

2. Dependencies among B_i's: Since every B_i contains at most $O(v^3 \cdot 2^{\sqrt[3]{v}})$ flow paths before the random experiment and each of these flow paths belongs to a rounding set of at most $2^{\sqrt[3]{v}}$ flow paths, a B_i can intersect at most $O(v^3 \cdot 2^{2\sqrt[3]{v}})$ many other B_j's.

3. Dependencies among A_i's and B_i's: Since every edge associated with an A_i contains at most $O(\log F \cdot 2^{\sqrt[3]{v}})$ flow paths and each of these flow paths belongs to a rounding set of at most $2^{\sqrt[3]{v}}$ flow paths, an A_i can intersect at most $O(\log F \cdot 2^{2\sqrt[3]{v}})$ many other B_j's. On the other hand, since every B_i contains at most $O(v^3 \cdot 2^{\sqrt[3]{v}})$ flow paths, each of these flow paths belongs to a rounding set of at most $2^{\sqrt[3]{v}}$ flow paths, and every flow path has a length of at most $4F$, a B_i can intersect at most $O(F \log F \cdot 2^{\sqrt[3]{v}})$ many other A_j's.

Hence, every event intersects at most $d = O((F \log F + v^3) \cdot 2^{2\sqrt[3]{v}})$ other events. Since α can be chosen so that $p \cdot d^9 \leq (1/2\mathrm{e})^3$ and $p \cdot |T_i| \leq 1$ for every T_i, it follows from Theorem 4.7 that the task specified for Round(v) can be achieved in polynomial time.

4. Final Rounding. Finally, we want to round the flow paths of the remaining requests so that afterwards,

- every remaining request i has a single flow path of value d_i,
- the flow along every edge increases by at most an additive $\Theta(\log F)$, and
- the total flow of the system decreased by at most some constant factor.

This would result in the theorem. In order to achieve this, consider the following random experiment:

Each participating request i decides with probability d'_i/d_i to increase its demand to d_i and if so, selects exactly one of its flow paths for this, giving a flow path of value f a probability of f/d'_i.

Let each event A_i represent the bad event that the flow value of an edge increases by more than an additive $\Theta(\log F)$. In addition, we will again use events B_i. For this we group the flow paths into sets S_i with a total value in $[\beta \log F, 2\beta \log F]$ for some constant $\beta > 0$. Obviously, the expected amount of flow in S_i after the random experiment is also in $[\beta \log F, 2\beta \log F]$. B_i represents the bad event that the total amount of flow in S_i is by more than a constant factor smaller than its expected value. If it can be ensured that none of the events A_i and B_i are true, then the requirements for the final rounding would be fulfilled. We again want to use Theorem 4.7 to show that this can be achieved in polynomial time.

We start with bounding the probability for the events A_i and B_i to occur. One can easily show via Chernoff bounds that

$$\Pr[A_i] \leq \mathrm{e}^{-\alpha \log F}$$

for a constant $\alpha > 0$ that can be made as large as required (as long as γ and δ are sufficiently large). Furthermore, one can show that

$$\Pr[B_i] \leq \mathrm{e}^{-\beta' \log F}$$

for a constant $\beta' > 0$ that can be made sufficiently large. We can therefore set $p = \mathrm{e}^{-\alpha \log F}$ as an upper bound for both $\Pr[A_i]$ and $\Pr[B_i]$.

Since the edge associated with A_i is only used by at most $O(F \log F)$ flow paths and each B_i contains at most $O(F \log F)$ flow paths before the random experiment and every flow path has a length of at most $4F$, each trial set T_i intersects at most $d = O(F^2 \log F)$ other T_j's. Since α can be chosen so that $p \cdot d^9 \leq (1/2\mathrm{e})^3$ and $p \cdot |T_i| \leq 1$ for every T_i, it follows from Theorem 4.7 that the task specified for ROUND(v) can be achieved in polynomial time.

Hence, altogether we obtain a feasible solution for the UFP (if γ and δ are sufficiently large) with asymptotically the same total demand as the optimal fractional solution for the UFP, which concludes the proof. $\square$

4.5 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 heaviest. 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 we get when running the greedy algorithm (without any restriction) on the requests sorted in decreasing order of their demands. As our solution we take the maximal of these two, that is $\mathcal{B} = \max(\mathcal{B}_1, \mathcal{B}_2)$.

Theorem 4.8 *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 weighted 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 $c(e)/4$ (hint: distinguish between $d(p) \leq c(e)/4$ and $d(p) > c(e)/4$). Each of these paths q is a *type I witness* of p and its weight for p is $d(q) \cdot d(p)/c(e)$. 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 $d(q) \cdot \sqrt{m}$, 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 $d(q) \cdot d(p)/c(e_i)$. Clearly, the weight of each path $q \in \mathcal{B}$ as a type II witness is at most $d(q) \cdot m$ 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 $\sqrt{m} \cdot d(p)/2$. The 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 $c_{\min} > 1$, 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 $c_{\min} > 1$ 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 best possible by providing a matching lower bound. The proof follows almost exactly the arguments given by Azar and Regev [2]. 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 $\Omega(c_{\min} \cdot (F^{1/(c_{\min}-1)} - 1))$.*

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 counter parts. 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 cancelled 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 cancelled 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 cancelled later on.

Lemma 5.6

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

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 Theorem 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 cancelation of old paths with total weight at most $W/2$. Otherwise, reject it.

Theorem 5.7 *For any network G with capacity C 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 cancelled 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 cancelled 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. $\square$

6 Open Problems

In this paper we 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). How can the unsplittability be exploited in the analysis to obtain better bounds?

References

[1] B. Awerbuch, Y. Azar, and S. Plotkin. Throughput-competitive on-line routing. In *Proceedings of the 34th Annual Symposium on Foundations of Computer Science*, pages 32–40, Palo Alto, California, 3–5 Nov. 1993. IEEE.

[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 (IPCO)*, 2001.

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

[4] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis.* Cambridge University Press, 1998.

[5] 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.

[6] 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 *Proc. 31st ACM Symp. on Theory of Computing*, pages 19–28, 1999.

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

[8] J. Kleinberg. Decision algorithms for unsplittable flow and the half-disjoint paths problem. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 530–539, 1998.

[9] 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.

[10] J. Kleinberg and É. Tardos. Disjoint paths in densely embedded graphs. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, pages 52–61, Los Alamitos, Oct. 1995. IEEE Computer Society Press.

[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 *IPCO: 6th Integer Programming and Combinatorial Optimization Conference*, volume 1412 of *Lecture Notes in Computer Science*, pages 153–162, 1998.

[13] 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, Crete Island, Greece, July 3–6, 2001. SIGACT/SIGARCH and EATCS.

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

[15] T. Leighton and S. Rao. An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms. In *29th Annual Symposium on Foundations of Computer Science*, pages 422–431, White Plains, New York, 24–26 Oct. 1988. IEEE.

[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] M. Molloy and B. Reed. Further algorithmic aspects of the local lemma. In *Proceedings of the 30th ACM Symposium on Theory of Computing (STOC)*, pages 524–529, 1998.

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

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

[20] C. Scheideler. *Probabilistic Methods for Coordination Problems.* Habilitation thesis, University of Paderborn, July 2000.

[21] 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.