November 2003

Near-Optimal Hardness Results and Approximation Algorithms for Edge-Disjoint Paths and Related Problems

Venkatesan Guruswami
Massachusetts Institute of Technology

Sanjeev Khanna
University of Pennsylvania, sanjeev@cis.upenn.edu

Rajmohan Rajaraman
Northeastern University

F. Bruce Shepherd
Bell Labs

Mihalis Yannakakis
Bell Labs

Postprint version. Published in *Journal of Computer and System Sciences*, Volume 67, Issue 3, November 2003, pages 473-496.
Publisher URL: http://dx.doi.org/10.1016/S0022-0000(03)00066-7

Near-Optimal Hardness Results and Approximation Algorithms for Edge-Disjoint Paths and Related Problems

Abstract

We study the approximability of edge-disjoint paths and related problems. In the edge-disjoint paths problem (EDP), we are given a network *G* with source-sink pairs (s_i, t_i), $1 \leq i \leq k$, and the goal is to find a largest subset of source-sink pairs that can be simultanoeusly connected in an edge-disjoint manner. We show that in *directed* networks, for any $\varepsilon > 0$, EDP is NP-hard to approximate within $m^{1/2-\varepsilon}$. We also design simple approximation algorithms that achieve essentially matching approximation guarantees for some generalizations of EDP. Another related class of routing problems that we study concerns EDP with the additional constraint that the routing paths be of bounded length. We show that, for any $\varepsilon > 0$, bounded length EDP is hard to approximate within $m^{1/2-\varepsilon}$ even in undirected networks, and give an $O(\sqrt{m})$-approximation algorithm for it. For directed networks, we show that even the single source-sink pair case (i.e. find the maximum number of paths of bounded length between a given source-sink pair) is hard to approximate within $m^{1/2-\varepsilon}$, for any $\varepsilon > 0$.

Keywords

approximation algorithms, bounded length edge-disjoint paths, edge-disjoint paths, hardness of approximation, multicommodity flow, network routing, unsplittable flow, vertex-disjoint paths.

Comments

Postprint version. Published in *Journal of Computer and System Sciences*, Volume 67, Issue 3, November 2003, pages 473-496.
Publisher URL: http://dx.doi.org/10.1016/S0022-0000(03)00066-7

Near-Optimal Hardness Results and Approximation Algorithms for Edge-Disjoint Paths and Related Problems*

Venkatesan Guruswami[†] Sanjeev Khanna[‡] Rajmohan Rajaraman[§]
Bruce Shepherd[¶] Mihalis Yannakakis[||]

August 17, 2001

Abstract

We study the approximability of edge-disjoint paths and related problems. In the edge-disjoint paths problem (EDP), we are given a network G with source-sink pairs (s_i, t_i), $1 \leq i \leq k$, and the goal is to find a largest subset of source-sink pairs that can be simultaneously connected in an edge-disjoint manner. We show that in *directed* networks, for any $\epsilon > 0$, EDP is NP-hard to approximate within $m^{1/2-\epsilon}$. We also design simple approximation algorithms that achieve essentially matching approximation guarantees for some generalizations of EDP. Another related class of routing problems that we study concerns EDP with the additional constraint that the routing paths be of bounded length. We show that, for any $\epsilon > 0$, bounded length EDP is hard to approximate within $m^{1/2-\epsilon}$ even in undirected networks, and give an $O(\sqrt{m})$-approximation algorithm for it. For directed networks, we show that even the single source-sink pair case (i.e. find the maximum number of paths of bounded length between a given source-sink pair) is hard to approximate within $m^{1/2-\epsilon}$, for any $\epsilon > 0$.

Keywords: approximation algorithms, bounded length edge-disjoint paths, edge-disjoint paths, hardness of approximation, multicommodity flow, network routing, unsplittable flow, vertex-disjoint paths.

*A preliminary version of this paper appeared in the *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, 1999, pp. 19–28.

[†]MIT Laboratory for Computer Science, 545 Technology Square, Cambridge, MA 02139. Email: `venkat@theory.lcs.mit.edu`.

[‡]Dept. of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104. E-mail: `sanjeev@cis.upenn.edu`.

[§]College of Computer Science, Northeastern University, Boston MA 02115. Email: `rraj@ccs.neu.edu`. Part of this work was done when the author was visiting Bell Labs during his DIMACS postdoctoral fellowship 1997-1998.

[¶]Bell Labs, 700 Mountain Avenue, Murray Hill, NJ 07974. Email: `bshep@research.bell-labs.com`.

[||]Bell Labs, 700 Mountain Avenue, Murray Hill, NJ 07974. Email: `mihalis@research.bell-labs.com`.

1 Introduction

In the *edge-disjoint paths* problem, denoted EDP, we are given a (possibly directed) graph G and a set $\mathcal{T} = \{(s_i, t_i) : 1 \leq i \leq k\}$ of k source-sink pairs, and the objective is to connect a maximum number of these pairs via edge-disjoint paths. EDP turns out to be a fundamental, extensively studied problem in the fields of combinatorial optimization, algorithmic graph theory and operations research, and is one of the classical NP-hard problems [9]. This paper investigates the approximability of EDP and of two related classes of network routing problems (on undirected as well as directed graphs) that are natural generalizations and variants of EDP. We now define these classes of problems for the directed graph case; we omit defining their standard restrictions to the undirected case.

Multicommodity Flow Problems: In multicommodity problems, we are given a directed graph $G = (V, A)$ and a set $\mathcal{T}$ of k source-sink pairs as above – we let $m = |A|$ and $n = |V|$ throughout. In addition, we are also given an integer capacity function $u : A \to \mathbf{Z}$ on the arcs, and a positive integer demand d_i for each (s_i, t_i) pair in $\mathcal{T}$, $1 \leq i \leq k$; d_i represents the bandwidth requested for flow from source s_i to sink t_i. We use the notation u_{min} and d_{max} to denote the minimum capacity and maximum demand value respectively. In all versions discussed, we assume that the bandwidth assigned (or reserved) for a pair (s_i, t_i) induces a standard network flow between s_i and t_i of value d_i. Depending on the version addressed, we may require extra conditions on these flows, e.g., for *unsplittable flow problems*, we require this flow to be sent on a single path.

We denote by $\mathcal{P}_i$ the set of all (simple) directed paths in D from s_i to t_i. A *routing* (of $\mathcal{T}$) in D is an assignment $x : \mathcal{P} \to \mathbf{R}_+$ of weights to directed paths in D, where $\mathcal{P} = \cup_{1 \leq i \leq k} \mathcal{P}_i$. A routing is said to *fulfill* the demand d_i for pair i, if $\sum(x(P) : P \in \mathcal{P}_i) = d_i$, i.e, if the following demand constraint is satisfied with equality:

$$\sum(x(P) : P \in \mathcal{P}_i) \leq d_i. \tag{1}$$

A routing x *satisfies the arc capacities* if

$$\sum(x(P) : P \in \mathcal{P}, e \text{ is an arc of } P) \leq u(e) \tag{2}$$

holds for every arc $e \in A$.

We consider several versions of these multicommodity routing problems. In each case, there is a profit $r_i \geq 0$ associated with each demand pair (s_i, t_i) and we wish to maximize our total profit. The common thread is that we only gain profit for the pair (s_i, t_i) if our routing has *fulfilled* its demand d_i.

We can consider two basic feasibility models: (i) *unsplittable flow* (USF), where we only gain profit for the demand pair (s_i, t_i) if our routing has assigned a flow of value d_i to a single path in $\mathcal{P}_i$, and, (ii) *splittable flow* (SF), where we gain the profit r_i as long as our routing has assigned a total weight of d_i to the paths in $\mathcal{P}_i$. In the splittable version, we further require our routings to be integral – this is referred to as *integral splittable flow* (ISF). Throughout, we let O_u, O_{is}, O_s respectively denote the maximum earnable profit from a feasible routing for the *unsplittable, integral-splittable* and *fractional-splittable* versions. One natural assumption is made throughout for all versions of the problem:

$$\text{The supply graph always has enough capacity to satisfy any single demand.} \tag{3}$$

Note that in the special case when all demands, capacities and profits equal one, both USF and ISF reduce to EDP. Hence these problems are NP-hard themselves, which motivates the investigation of their efficient approximability, which is one of the focuses of this paper.

Bounded Length Edge Disjoint Paths (BLEDP) Problems: A second set of problems that we study here is that of routing a maximum number of edge-disjoint paths between specified source-sink pairs in a network such that each of the paths used is of length at most L, for some length-bound L that is also part of the problem instance. We refer to this problem as BLEDP. As observed in Kleinberg [16], this length constraint, which arises quite naturally in practical routing problems, can transform tractable disjoint paths problems into NP-hard variants. For instance, even the single source-single sink case of this problem, referred to as (s,t)-BLEDP, is NP-hard.

1.1 Overview of Our Results

EDP, USF **and** ISF: Approximation algorithms for USF and EDP have been extensively studied in prior works [23, 10, 1, 18, 19, 15, 16, 17, 28, 2, 21, 22, 6]; [16] provides a comprehensive background on these and related problems. The best known approximation factor for EDP is $O(\sqrt{m})$ [16] (an $O(\sqrt{m})$ approximation for weighted EDP, where profits are not necessarily all one, is presented in [28]). In recent work, an $O(\sqrt{m})$ approximation algorithm has been obtained for the more general USF problem [2], under the assumption that $d_{\max} \leq u_{\min}$. The preceding result improves upon the $O(\sqrt{m}\log m)$ approximation algorithm of [22] for USF. All these approximation bounds are rather weak and reflect the generally appreciated hardness of these problems. Yet no better hardness than MAX SNP-hardness is known for any of these problems. We prove that even EDP on directed graphs is NP-hard to approximate within a factor of $m^{1/2-\epsilon}$ for any $\epsilon > 0$; our proof is surprisingly simple and does not rely on the PCP theorem. Recently, Ma and Wang [24] have independently shown a weaker hardness result, namely, EDP on directed graphs is *Quasi*-NP-hard to approximate within $2^{O(\log^{1-\epsilon} m)}$ for any $\epsilon > 0$. Their proof uses the hardness of approximating LABEL COVER, and hence relies on the PCP theorem.

On the algorithmic side, we present a simple randomized $O(\sqrt{m}\log^{3/2} m)$ approximation algorithm for USF with polynomially bounded demands, without making the assumption that $d_{max} \leq u_{min}$. With the assumption that $d_{max} \leq u_{min}$, the approximation guarantee of our algorithm improves to $O(\sqrt{m\log m}\log\log m)$. While the preceding approximation guarantee is weaker than the $O(\sqrt{m})$ bound achieved recently by [2], the significance of our result lies in the fact that our randomized algorithm uses, perhaps, the most basic rounding scheme introduced by Raghavan and Thompson [26] and our analysis relies on elementary combinatorial arguments and straightforward Chernoff-type bounds. We also achieve an $O(\sqrt{m}\log^2 m)$ approximation for USF using a simple greedy algorithm; the main contribution here is extending the analysis of [22] for EDP to handle general capacities. While our strong inapproximability results above apply only for the directed case, we can also prove the (once again tight up to polylogarithmic factors) result that *undirected* USF is NP-hard to approximate within $n^{1/2-\epsilon}$, for any $\epsilon > 0$, if we consider the *node-capacitated* version [17] instead.

The integral splittable version of the problem was shown to be NP-hard on directed as well as undirected graphs (even with just two sources and sinks) in [7] and also for trees in [10], and to our knowledge no explicit results on its approximability appear in the literature. Our hardness result for EDP trivially implies a similar hardness bound for approximating ISF on directed graphs. In fact, an easy reduction from Independent Set (where the demand pairs play the role of the nodes) shows that the same hardness bound of $m^{1/2-\epsilon}$ applies for the undirected case as well. This same reduction shows that the "fractional" splittable version of the problem remains as hard to approximate. On the algorithmic side, we present a simple greedy algorithm, again generalizing the one in [22], that achieves an approximation guarantee of $O(\sqrt{md_{max}}\log^2 m)$.

BLEDP: We show that BLEDP can be approximated in polynomial time within a factor of $O(\sqrt{m})$. We prove a matching hardness result of $m^{1/2-\epsilon}$ for any $\epsilon > 0$ that works for undirected graphs as well. For (s,t)-BLEDP we prove an inapproximability result of $m^{1/2-\epsilon}$ for directed graphs, and MAX SNP-hardness for undirected (and directed) graphs. The MAX SNP-hardness applies even when the length bound is a (small) constant. We also present a simple greedy algorithm for BLEDP that achieves an $O(\sqrt{m})$ approximation.

REMARK: In general, directed versions of these problems appear to be harder than their undirected counterparts. Accordingly, all our algorithms are described for the directed case, but they all work for the undirected case as well. Regarding hardness results, unless mentioned otherwise the result applies only to the directed case – but if a hardness result is stated or proved specifically for the undirected case, we stress that a similar result will hold for the directed case as well.

1.2 Organization

A portion of our algorithmic work follows a linear programming based approach, and hence we begin by describing the relaxations we use, bound their integrality gaps, and note a useful property about the structure of basic feasible solutions in Section 2. In Section 3, we present the hardness results for EDP, USF and ISF, and present an LP-based approximation for USF. We study the hardness of BLEDP problems in Section 4. Finally, in Section 5, we present simple greedy algorithms for all our problems that almost match the corresponding hardness bounds.

2 LP Formulations and Rounding

LP Formulations: A natural relaxation (in the sense of the objective function) of our network routing problems is the following linear program (LP) LP-BASIC.

$$\max\{\sum_{i=1}^{k}\frac{r_i}{d_i}(\sum_{P\in\mathcal{P}_i}x(P)) : \; x\geq 0 \;\text{ and satisfies the constraints (1), (2)}\} \tag{4}$$

Let LP_s denote the optimum value of this LP. Of course, in solving such an LP, one would resort to the well-known compact formulation which uses variables f_e^i for the flow for the demand pair (s_i,t_i) through edge e, for each $i\in[k]$ and $e\in A$. For the purposes of exposition, however, we view our solutions as vectors in $\mathbf{R}^{\mathcal{P}}$. For any such vector x, we denote by $supp(x)$ the set of paths P for which $x(P)>0$.
One sees immediately that LP_s may be much more than our desired optimum.

Example 2.1 *Let G be a cycle with unit capacities on the arcs. Also between each pair of vertices consider a demand of size 2 and a profit of 1. Then clearly $LP_s = m/2$, however, $O_{is} = O_s = 1$ (N.B. $O_u = 0$ and would hence violate (3)).*

For many versions of our problems, we may easily amend the LP formulation to get a tighter relaxation. For instance, in the unsplittable version we define $\mathcal{P}_i^*$ to be those paths in $\mathcal{P}_i$ for which each arc has capacity at least d_i (and let $\mathcal{P}^*$ be the union of these sets). Clearly, any feasible solution to the unsplittable problem may only use the paths in $\mathcal{P}^*$ in its support. Thus we define the linear program LP-UNSPLIT:

$$\max\{\sum_{i=1}^{k} \frac{r_i}{d_i}(\sum_{P \in \mathcal{P}_i^*} x(P)) : \ x \geq 0 \ \text{ and satisfies the constraints } (1), (2)\} \tag{5}$$

We use LP_u to denote the optimum value of this LP. Clearly we have $O_u \leq LP_u \leq LP_s$ and $O_u \leq O_{is} \leq O_s \leq LP_s$. Again, we view solutions as vectors in $\mathbf{R}^{\mathcal{P}^*}$. It turns out that this LP is a better approximation for the unsplittable flow problem than (4) was for the splittable flow problem. We will see that: $O_u = \Omega(LP_u/\sqrt{m}\text{poly}(\log m))$.

For a solution x to any of our LP's, we use $G(x)$ to denote the set of (*good*) demand pairs (s_i, t_i) for which (1) is satisfied with equality. $B(x)$ denotes the set of demand pairs (s_i, t_i) which are not good but for which the left hand side of (1) is positive. Finally, let $S(x) = B(x) \cup G(x)$ be the set of demand pairs which are at least partially satisfied.
There seems to be very little known about the structure of basic solutions for such multi-commodity LP's. One elementary result we can prove is the following.

Proposition 2.1 *If x is a basic optimal solution to (4) or (5), then $|supp(x)| \leq |G(x)| + m$.*

Proof: We argue this for (4) but the same holds for (5). One easily sees that each unit vector in $\mathbf{R}^{\mathcal{P}}$ is feasible for (4) and hence the solution space is full-dimensional. It follows that any basic solution must satisfy some linearly independent subsystem of $|\mathcal{P}|$ constraints. The result now follows. ∎

Proposition 2.1 gives the following weak lower bound on the number of pairs which are satisfied by an unsplit flow.

Proposition 2.2 *If x is a basic optimal solution to (4), then at least $|S(x)| - m$ pairs are satisfied by an unsplit flow under x.*

Proof: Let x be a basic optimal solution to (4) and for each demand pair (s_i, t_i), let n_i denote $|supp(x) \cap \mathcal{P}_i|$. Hence (s_i, t_i) is satisfied by an unsplit flow under x if and only if $n_i = 1$ and $(s_i, t_i) \in G(x)$. Let $U(x)$ denote the set of such pairs. Then, by Proposition 2.1, we have:

$$\begin{aligned} |G(x)| + m &\geq |supp(x)| = \sum_i n_i \\ &\geq |U(x)| + |B(x)| + 2|G(x) - U(x)| \end{aligned}$$

and hence $|U(x)| \geq |B(x)| + |G(x)| - m = |S(x)| - m$, as required. ∎

If our demands set $\mathcal{T}$ consists of every pair of vertices in G so that for each pair of vertices $i \neq j$ there is a positive integral demand $d(ij)$, then we call this an *all-pairs* instance. The preceding immediately implies:

Corollary 2.1 *If all profits equal one, i.e if* $r \equiv \bar{1}$*, then* $O_{is} \geq O_u \geq LP_s - m$*, and in particular for any all-pairs instance, the LP (4) gives a factor 2 approximation algorithm as long as* $d(ij) \leq u(ij)$ *for each* $(i,j) \in A$*.* ■

Another corollary, using Example 2.1, is a tightness result for the formulation (4) – the proof is easy and is omitted.

Corollary 2.2 $O_s \geq O_{is} \geq O_u \geq \frac{1}{2m} LP_s$. ■

Standard Rounding and Deviation Bounds: We will later show how the LP formulation for USF can be rounded to obtain an approximate solution with performance ratios almost matching our hardness result. We now review a standard rounding technique and develop some bounds on its performance that we will use in our analysis later.

Consider a solution to our LP formulation for USF. Let z_i denote the fraction of the demand between the pair (s_i, t_i) that is satisfied by this solution. Decompose the flow of value $z_i d_i$ into a set of flow paths $\{\Gamma_{i,1}, \Gamma_{i,2}, ..., \Gamma_{i,q_i}\}$ where flow on path $\Gamma_{i,j}$ is given by $f_{i,j}$. Now consider the following rounding procedure, introduced by Raghavan and Thompson [26] in a classic paper. Each (s_i, t_i) pair is routed independently with probability z_i and once a pair is chosen, we toss a q_i-sided dice with the property that the jth face shows up with probability $f_{i,j}/(z_i d_i)$. We choose to route the pair along the path $\Gamma_{i,j}$ if the jth face turns up. This rounding procedure is referred to as the *standard rounding* from here on. In what follows, we develop some properties of this rounding procedure. We start by stating some well-known deviation bounds [4, 11].

Proposition 2.3 *(Chernoff-Hoeffding Bounds) Let* $X_1, X_2, \ldots, X_\ell$ *be a set of* k *independent random variables in* $[0,1]$ *and let* $X = \sum_{i=1}^{\ell} X_i$*.*

1. *For any* $\delta \geq 0$*, we have:*

$$\Pr[X > (1+\delta)E(X)] \leq \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}} \right)^{E(X)}$$

2. *For* $0 \leq \delta < 1$*, we have:*

$$\Pr[X < (1-\delta)E(X)] \leq e^{-\delta^2 E(X)/2}.$$

Proposition 2.4 *Consider an LP solution to an instance of the* USF *problem with each demand being at least* 1*. Let* S *be a set of arcs and let* f_e *denote the flow through an arc* $e \in S$ *in an LP solution. Then if* X *is the random variable indicating the total number of paths in the standard rounding solution that use at least one arc in the set* S*,* $\Pr[X > c \cdot \max\{\sum_{e \in S} f_e, \log m\}] < 1/m^2$ *for some suitably large constant* c*.*

Proof: For each demand pair (s_i, t_i), let Γ_i denote the set of s_i-t_i paths in the flow decomposition (of the LP solution) that use at least one arc in S. Let $f(\Gamma_i)$ denote the total flow on all flow paths in the set Γ_i. Finally, let X_i be a 0/1 random variable that indicates whether or not any path in Γ_i is chosen in the rounded solution. Clearly, $E(X_i) = f(\Gamma_i)/d_i$ and $X = \sum_i X_i$.

Since X_i's are independent random variables, we can use the bound in Proposition 2.3(1) with a sufficiently large value of c to conclude that $\Pr[X = \sum_i X_i > c \cdot \max\{\sum_i (f(\Gamma_i)/d_i), \log m\}] < 1/m^2$. The proposition now follows from the fact that $\sum_{e \in S} f_e \geq \sum_i f(\Gamma_i) \geq \sum_i f(\Gamma_i)/d_i$ since each d_i is at least 1. ∎

Corollary 2.3 *If $u_{\min}/d_{\max} \geq c \log m$ for some suitably large constant c, then* USF *can be approximated to within a constant factor.*

Proof: Solve LP-UNSPLIT, scale the flow down by a factor of $1/c$, where c is a positive constant, and perform the standard rounding. If c is chosen sufficiently large, then it follows from Proposition 2.3(1) that w.h.p., no arc capacity is violated in the rounding procedure. On the other hand, the expected value of the solution obtained is $\Omega(LP_u)$. The result follows. ∎

3 EDP, USF and ISF

3.1 Hardness of Approximating EDP (USF and ISF)

Recall that in the *edge-disjoint paths problem* (EDP), we are given a graph G and a set $\mathcal{T}$ of k source-sink pairs $(s_1, t_1), \ldots, (s_k, t_k)$, and the goal is to find a subset $\mathcal{S} \subseteq \mathcal{T}$ of maximum cardinality such that all (s_i, t_i) pairs in $\mathcal{S}$ can be connected by edge-disjoint paths. Only an $O(\sqrt{m})$-approximation is known for EDP [16]. We now prove an essentially matching hardness result on directed graphs.

Theorem 1 *For any $\epsilon > 0$, it is NP-hard to distinguish given a directed instance $[G = (V, A), \mathcal{T} = \{(s_i, t_i) : i \in [k], s_i, t_i \in V\}]$ with $|A| = m$, whether all k pairs in $\mathcal{T}$ can be connected by edge-disjoint paths or at most a fraction $1/m^{1/2-\epsilon}$ of the k pairs can be connected.*

Remark: Note that the theorem says that the difficulty of the problem is not due to determining which subset of pairs to "route", but lies in determining the routing itself.

Proof of Theorem 1: The proof is by a reduction from the following well-known NP-hard problem [8]:

PROBLEM: 2DIRPATH:
INSTANCE: A *directed* graph $H = (V, A)$, *distinct* vertices $x_1, x_2, y_1, y_2 \in V$.
QUESTION: Are there two edge-disjoint directed paths, one from x_1 to y_1 and the other from x_2 to y_2, in H?

Given an $\epsilon > 0$, we construct a directed graph G (which will be the directed graph underlying our EDP instance) from H as follows. The skeleton of G will be the graph G' whose basic structure is as described below. G' will comprise of vertices s_i, t_i for $1 \leq i \leq N$, where $N = |A|^{\lceil 1/\epsilon \rceil}$, together with vertices $\{h_{ij}^{(1)}, h_{ij}^{(2)}, v_{ij}^{(1)}, v_{ij}^{(2)} : 1 \leq j < i \leq N\}$ and the "diagonal" vertices $\{d_{ii} : 1 \leq i \leq N\}$ connected in a grid-like fashion. Each s_i is connected by a directed path $\mathcal{P}_i$ to t_i where

$$\begin{aligned} \mathcal{P}_i &= \Big[s_i, v_{i1}^{(1)}, v_{i1}^{(2)}, v_{i2}^{(1)}, v_{i2}^{(2)}, \ldots, v_{i,i-1}^{(1)}, v_{i,i-1}^{(2)}, d_{ii}, \\ &\quad h_{i+1,i}^{(1)}, h_{i+1,i}^{(2)}, \ldots, h_{n,i}^{(1)}, h_{n,i}^{(2)}, t_i\Big] \end{aligned}$$

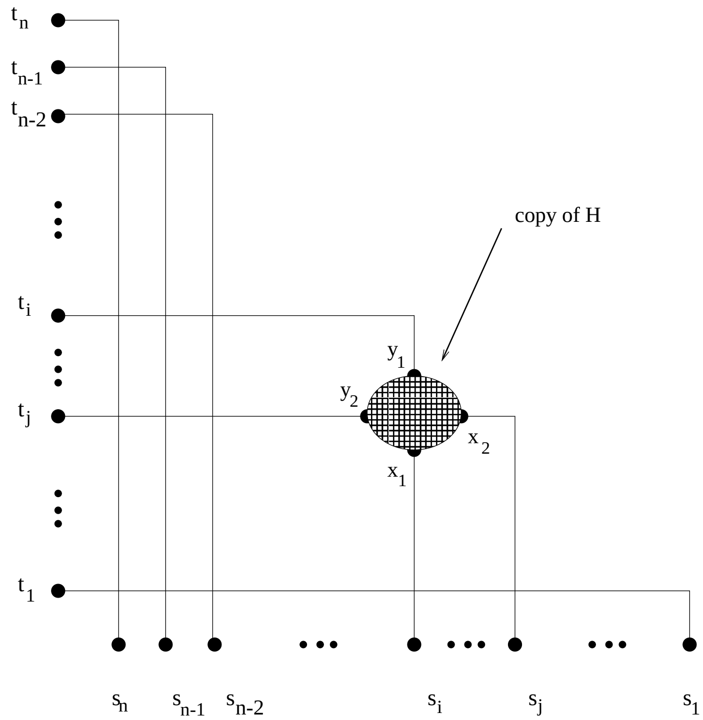

Figure 1: *The reduction from* 2DIRPATH *to* EDP

The edge set of G' will be the (disjoint) union of the edges in the paths $\mathcal{P}_i$ for $1 \leq i \leq N$. (For geometric intuition, one can identify s_i, t_i with the points $(N-i+1, 0)$ and $(0, i)$ on the 2-dimensional grid, and visualize $\mathcal{P}_i$ as the path connecting $(N-i+1, 0)$ with $(0, i)$ on the 2-dimensional grid that first goes "up" the y-direction, makes a *single bend*, and then goes "left" along the x-direction. The "ij intersection" can be thought of as located at the point $(N-i+1, j)$ of the 2-dimensional grid.)

The graph G is constructed by making the following modification at each "ij intersection" for $1 \leq j < i \leq N$: (i) remove the edges $(h_{ij}^{(1)}, h_{ij}^{(2)})$ and $(v_{ij}^{(1)}, v_{ij}^{(2)})$, and (ii) place a copy of the graph H at the ij intersection while identifying the x_1, y_1, x_2, y_2 of the copy of H with the vertices $h_{ij}^{(1)}, h_{ij}^{(2)}, v_{ij}^{(1)}, v_{ij}^{(2)}$, respectively, of the ij intersection. The instance of EDP will now comprise of the graph G together with the N source-sinks pairs $\mathcal{T} = \{(s_i, t_i) : i \in [N]\}$.

Claim 1 *If there are edge-disjoint paths from x_1 to y_1 and x_2 to y_2 in H, then there are N edge-disjoint paths in G, one connecting s_i to t_i, for each $i \in [N]$.*

Proof: Suppose A_1 and A_2 are two edge-disjoint paths in H connecting x_1 to y_1 and x_2 to y_2 respectively. For $r \in [N]$, define the path $\mathcal{Q}_r$ in G to be the s_r-t_r path that is the same as the path $\mathcal{P}_r$ of the skeleton G' except that instead of using the edges $(h_{ij}^{(1)}, h_{ij}^{(2)})$ and $(v_{ij}^{(1)}, v_{ij}^{(2)})$ (for the relevant values of i, j for the path $\mathcal{P}_r$), it uses the edge-disjoint paths A_1 and A_2, respectively, of the local copy of H at the ij intersection. The paths $\mathcal{Q}_r$ thus defined are clearly edge-disjoint. ∎

Claim 2 *If there exist edge-disjoint paths $\mathcal{Q}_{i_1}$ and $\mathcal{Q}_{i_2}$ in G connecting s_{i_1} to t_{i_1} and s_{i_2} to t_{i_2}, respectively, for any $1 \leq i_1 \neq i_2 \leq N$, then there must be two edge-disjoint paths in H from x_1 to y_1 and x_2 to y_2.*

Proof: Identify G with its embedding in the plane. Clearly, one can extend $\mathcal{Q}_{i_1}$ to a closed contour where s_{i_2}, t_{i_2} are on the outside and inside respectively. It follows (cf. [25]) that $\mathcal{Q}_{i_2}$ must cross this contour, but this implies that there exists p, q such that $\mathcal{Q}_{i_1}, \mathcal{Q}_{i_2}$ each uses one of the edges associated with the pq intersection.

Now going back to the graph G, the two paths $\mathcal{Q}_{i_1}$ and $\mathcal{Q}_{i_2}$ will intersect at a point as guaranteed by the above fact, i.e $\mathcal{Q}_{i_1}$ enters at x_1 and leaves at y_1 in the copy of H at this intersection, while $\mathcal{Q}_{i_2}$ enters at x_2 and leaves at y_2, and the edge-disjointness of $\mathcal{Q}_{i_1}$ and $\mathcal{Q}_{i_2}$ implies that there must exist two edge-disjoint paths in H from x_1 to y_1 and x_2 to y_2. ■

The above two claims imply that YES instances of 2DIRPATH are mapped to instances of EDP where all N pairs can be satisfied, while NO instances are mapped to instances of EDP where at most one pair is satisfied. This creates a gap of N, and since the number of arcs in G equals $m = O(N^2|A|) = O(N^{2+\epsilon})$ (recall that $N = |A|^{\lceil 1/\epsilon \rceil}$), the gap equals $\Omega(m^{1/(2+\epsilon)})$, and since $\epsilon > 0$ was arbitrary, we are done. ■

Corollary 3.1 *The* USF *and* ISF *problems on directed graphs are NP-hard to approximate within $m^{1/2-\epsilon}$ for any $\epsilon > 0$.*

Extending these $m^{1/2-\epsilon}$-hardness results for EDP, USF and ISF to undirected graphs pose problems of widely varying degrees of difficulty. For ISF the task is straightforward as there is an easy approximation-preserving reduction from the independent set problem in a graph G. Namely, one creates a new graph G' obtained by creating a non-adjacent copy v' of each node in G. Each edge in G' is assigned a capacity of one, and we include a demand between each v and v' for a flow of size d_v, the degree of v. This together with Håstad's inapproximability result for independent set [12], gives

Fact 3.1 *Unless* NP = ZPP, ISF *on undirected graphs cannot be approximated to within $m^{1/2-\epsilon}$ of the optimum in polynomial time, for any $\epsilon > 0$.*

At the other end, the hardness of undirected EDP remains an interesting open question. Indeed, even the hardness of edge-capacitated USF remains open. We initiate some progress in the situation for undirected graphs by considering the node-capacitated version of USF as was also considered, for instance, by Kleinberg [17] under a *boundedness* assumption. In this case, we can obtain an inapproximability bound of $n^{1/2-\epsilon}$, for any $\epsilon > 0$, where n is the number of nodes, even on **undirected** graphs; and this result is once again essentially tight.

We first define the undirected node-capacitated USF problem, denoted Undir-Node-USF, below: We are given an undirected graph $G = (V, E)$ with positive integral capacities on the **nodes**, and k source-sink pairs (s_i, t_i) with a positive integer demand d_i and a "profit" r_i for $i \in [k]$ — the objective is to find a subset of the pairs that can be routed feasibly (i.e all node capacities are obeyed) and that maximizes the total profit (as before we only get the profit for a pair (s_i, t_i) when its demand d_i is fully routed on a single path). When all profits are equal, we refer to the version as being "unweighted". Note that unweighted Undir-Node-USF is simply the classical vertex-disjoint paths problem when all demands and capacities are equal to 1.

Theorem 2 *It is NP-hard to approximate unweighted* Undir-Node-USF *within a factor of* $n^{1/2-\epsilon}$ *for any* $\epsilon > 0$.

We will see that the above theorem is nearly tight since LP-based (or greedy) algorithms like those presented in the following section for the edge-capacity version, can achieve an approximation ratio of $O(\sqrt{n}\ \mathrm{poly}(\log n))$ for Undir-Node-USF.

The proof of Theorem 2 uses the following theorem. The proof is based on a reduction from SAT along the lines of the NP-hardness proof for two-commodity integral flow presented in [7].

Theorem 3 *Given an instance of* Undir-Node-USF *with two source-sink pairs, it is NP-hard to decide if both pairs can be feasibly routed. Moreover, the hardness holds even if all node capacities are* 1 *or* 2, *and the two demands are* 1 *and* 2.

Proof: Given an instance of SAT, we will create an undirected graph $G = (V, E)$ with four distinct nodes s_1, s_2, t_1, t_2 and node capacities $c : V \to \{1, 2\}$ with the property that the instance of SAT is a yes-instance if and only if G contains a pair of node-disjoint paths P_1, P_2 such that P_i is a path from s_i to t_i and P_2 only uses nodes of capacity two.

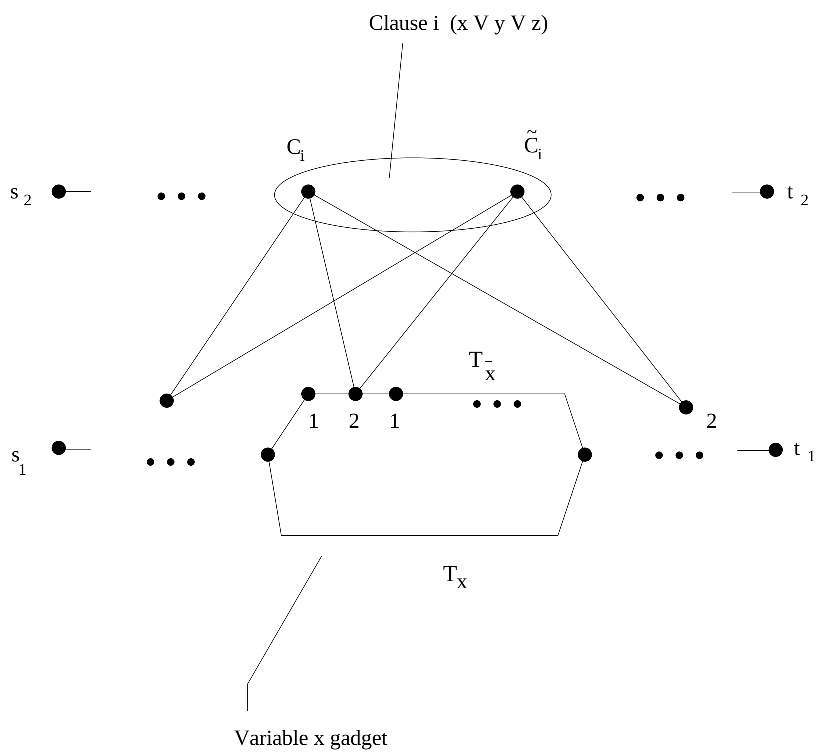

Figure 2: *The reduction from* SAT *to* Undir-Node-USF

The graph G is obtained by stringing together n cycle gadgets, one for each variable x in the SAT instance. These cycles are the union of two paths T_x and $T_{\bar{x}}$ each of which alternates between capacity 2 and capacity 1 nodes. We also create two capacity 2 nodes $C_i, \tilde{C}_i$ for each clause i, $1 \le i \le m$, and connect $\bar{C}_i$ to C_{i+1} for $1 \le i < m$, and also include edges from these to nodes in cycle gadgets corresponding to the literals in clause i — see Figure 2. In addition we add vertices s_1, s_2, t_1, t_2 to G and join s_1 (resp. t_1) to the obvious vertex of the cycle T_{x_1} (resp. T_{x_n}) where x_1 (x_n) is the first (last) variable in the SAT instance as per some ordering. The vertex s_2 is connected to C_1 and t_2 to $\bar{C}_m$.

Now consider a pair of paths P_1, P_2 satisfying our Undir-Node-USF problem. The path P_2 must have the form $s_2 C_1 v_1 \bar{C}_1 C_2 v_2 \bar{C}_2 \ldots \bar{C}_m t_2$, where each v_i is a node corresponding to some literal x in clause i. One thus deduces that the path P_1 must have the form $s_1 Q_1 Q_2 \ldots Q_n t_1$ where each Q_j is either T_{x_j} or $T_{\bar{x}_j}$. Since each Q_j is disjoint from each v_i, the path P_1 determines a satisfying truth assignment for the instance of SAT in the natural way. Conversely, one easily checks that a solution to the node-capacitated unsplittable flow problem can be obtained from a satisfying truth assignment. ■

We remark that the preceding proof (and hence Theorem 2) could be extended to showing hardness of USF with only the extra condition that flow paths must be node-disjoint (and not arbitrary node capacities). We now return to our task of proving hardness of Undir-Node-USF.

Proof of Theorem 2: We use a construction similar to the one used in Theorem 1 – the main difference is that at each "ij intersection" we now place a hard instance $\mathcal{I}$ of 2PAIR-Undir-Node-USF above, instead of the 2DIRPATH instance we used in Theorem 1. We set the demand of the pair (s_i, t_i) to $n - i$, for $i \in [N]$. For the nodes in the copy of $\mathcal{I}$ placed at the "ij intersection" (for $1 \leq j < i \leq N$) we change the capacities of nodes with capacity 1 (recall that all nodes in instance $\mathcal{I}$ have capacity 1 or 2) to $n - i$, and those of nodes with capacity 2 to $n - j$.

Omitting the details, we now claim that if the instance $\mathcal{I}$ was feasible, then the demands of all the N (s_i, t_i) pairs can be fully met by a feasible routing, while if $\mathcal{I}$ was not feasible, then at most one pair's demand can be met by any feasible routing in the Undir-Node-USF instance constructed above. As before, this gives us a gap of $n^{1/2-\epsilon}$ where n is the number of nodes in the Undir-Node-USF instance created. ■

In the next section we give an (essentially tight) approximation algorithm for USF based on rounding an optimal solution to the LP (5). Standard rounding techniques are not as easily applied to ISF; we defer the positive results for the latter problem to Section 5.

3.2 An LP-Based Approximation for USF

We now give a simple algorithm to approximate USF to within $O(\sqrt{m \log m} \log \log m)$ when $u_{\min} \geq d_{\max}$, and an $O(\sqrt{m}(\log m)^{3/2})$-approximation without this assumption, but requiring that $d_{\max}$ is polynomially bounded.

Theorem 4 *If all demands are between 1 and 2, then* USF *can be approximated to within a factor of $O(\sqrt{m \log m})$.*

Proof: We assume w.l.o.g. that the maximum profit, $\max_i r_i$, is 1; otherwise, we can scale the profits appropriately. We first solve LP-UNSPLIT. If the LP value LP_u is less than $\sqrt{m}$, then routing the demand with the highest profit yields a $\sqrt{m}$ approximation. Therefore, in the remainder of the proof, we assume that $LP_u \geq \sqrt{m}$. We scale down the LP flow by a factor of 1/6, and perform standard rounding. In the rounding procedure, each demand is selected independently with a probability equal to one-sixth of the fraction routed in the LP. Thus, the expected total profit of the rounded solution is $\Omega(LP_u)$. Since $LP_u \geq \sqrt{m}$ and the profit of each demand is in $[0, 1]$, it follows from Proposition 2.3(2) that w.h.p. the total profit of the rounded solution is $\Omega(LP_u)$. The rounded solution, however, may violate the capacity of many arcs. We now show that we can obtain a solution which w.h.p. has value $\Omega(LP_u/\sqrt{m}\log m)$ and does not violate any arc capacities.

Heavy and Light Edges. Let c_0 denote the quantity $12 \ln m$. Call an arc e *heavy* if $u(e) > c_0$ and *light* otherwise. Our first claim is that with probability at least $(1 - 1/m)$, the rounded solution does not violate the capacity of an heavy arc. To see this, define X_e to be a random variable that indicates the *total number of demands* routed through an arc e. By Proposition 2.3(1),

$$\Pr[X_e > \frac{u(e)}{2}] \leq \Pr[X_e > (1 + 2)2 \ln m] \leq \frac{1}{m^2}.$$

Thus with probability at least $1 - 1/m$, at most $u(e)/2$ paths are chosen to go through an arc of capacity at least $12 \ln m$. Since all demands are between 1 and 2, it follows that, with probability at least $1 - 1/m$, no heavy arc is violated.

Processing the Rounded Solution. Now let $\mathcal{R}$ denote the set of paths in a solution obtained by standard rounding. A path $P \in \mathcal{R}$ is said to be *α-light* if the total capacity associated with the light arcs that lie on this path is exactly α. Partition $\mathcal{R}$ into two sets $\mathcal{R}_1$ and $\mathcal{R}_2$ such that $\mathcal{R}_1$ is the set of all α-light paths with $\alpha \geq \sqrt{mc_0}$, and $\mathcal{R}_2 = \mathcal{R} \setminus \mathcal{R}_1$. It is easy to see that $\mathcal{R}$ contains at most mc_0/α paths which are α-light. Thus, $|\mathcal{R}_1|$ is at most $\sqrt{mc_0}$.

Initialize S_1 to be the single-element set consisting of a path of largest profit in $\mathcal{R}_1$. For a given set S of paths, let $\text{PROFIT}(S)$ denote the sum of the profits of the source-sink pairs routed in S. Clearly, $\text{PROFIT}(S_1) \geq \text{PROFIT}(\mathcal{R}_1)/\sqrt{mc_0}$. Construct also another set S_2 of paths as follows: Repeatedly pick a path P of largest profit from $\mathcal{R}_2$, add P to S_2, and delete P along with all paths in $\mathcal{R}_2$ that share a light arc with P. Since the total capacity of light arcs is at most $\sqrt{mc_0}$, it follows from Proposition 2.4 that for any path P in $\mathcal{R}_2$, the total number of paths in $\mathcal{R}$ that share a light arc with P is $O(\sqrt{mc_0})$ with probability at least $1 - 1/m^2$. Therefore, with probability at least $1 - 1/m$, for every path added to S_2, only $O(\sqrt{mc_0})$ other paths in $\mathcal{R}_2$ are thrown away. Thus, upon termination (i.e. when $\mathcal{R}_2 = \emptyset$), $\text{PROFIT}(S_2)$ is $\Omega(\text{PROFIT}(\mathcal{R}_2)/\sqrt{mc_0})$ w.h.p. Output the better among the solutions S_1 and S_2.

Approximation Ratio and Correctness. It is clear from the preceding description that we are guaranteed a profit of $\Omega(\text{PROFIT}(\mathcal{R})/\sqrt{mc_0})$; this yields the claimed approximation guarantee as $\text{PROFIT}(\mathcal{R})$ is $\Omega(LP_u)$ w.h.p. To see that with w.h.p., the set of paths in S_2 does not violate the capacity of any arc, observe that: (a) no two paths in S_2 share a light arc, and (b) with probability $1 - 1/m$, the set of paths in $\mathcal{R}$ does not violate the capacity of a heavy arc. ■

Theorem 5 USF *with polynomially bounded demands can be approximated to within a factor of $O(\sqrt{m}\log^{3/2} m)$.*

Proof: Let $d_{\max} = O(m^{O(1)})$ denote the largest demand. Partition the demands into $O(\log m)$ classes, say $D_0, D_1, \ldots$ where the class D_i contains all demands in the interval $[2^i, 2^{i+1})$. Find a solution for each demand class using the approach of Theorem 4. Output the solution with the largest profit. By pigeonhole principle, one of these classes contains $\Omega(1/\log m)$-fraction of the optimum solution's total profit. The theorem follows. ■

Theorem 6 USF *with arbitrary demands can be approximated to within a* $O(\sqrt{m \log m} \log \log m)$ *factor if the minimum arc capacity is at least as large as the largest demand.*

Proof: Without loss of generality, we may assume that $d_{min} = 1$. As before, partition the demands into classes $D_0, D_1, ..., D_q$ where the class D_i contains all demands in the interval $[2^i, 2^{i+1})$. Set $p = q - (\log \log m + \log c)$ where c is the constant of Corollary 2.3, and define $\mathcal{D}_1 = \bigcup_{i=1}^{p} D_i$ and $\mathcal{D}_2 = \bigcup_{i=p}^{q} D_i$. Also, for some optimal solution, let OPT_1 and OPT_2 denote the profit generated by commodities in the demand sets $\mathcal{D}_1$ and $\mathcal{D}_2$ respectively.

Since the maximum demand in $\mathcal{D}_1$ is at most

$$(d_{max}/c \log m) \leq (u_{min}/c \log m),$$

by Corollary 2.3, it follows that OPT_1 can be approximated to within a constant factor. On the other hand, we can use Theorem 4 to find an approximate solution for each demand class D_i with $p \leq i \leq q$. By pigeonhole principle, one of these classes contains $\Omega(1/\log \log m)$-fraction of OPT_2. Choosing the better of the two solutions, we get the desired approximation ratio. ■

4 Bounded Length Edge-Disjoint Paths

We are given a (possibly directed) graph $G = (V, E)$, in which each edge e has a given nonnegative length. We are also given an integer L, which is referred to as the *length bound*. In BLEDP, we are given a set $\mathcal{T}$ of k source-sink pairs $(s_1, t_1), \ldots, (s_k, t_k)$, and the goal is to find a subset $\mathcal{S} \subseteq \mathcal{T}$ of maximum cardinality such that all (s_i, t_i) pairs in $\mathcal{S}$ can be connected by edge-disjoint paths, each path of "length" at most L. In the (s,t)-BLEDP problem, we are given a *single* pair (s,t), the goal is to find a maximum number of edge-disjoint s–t paths each of "length" at most L.

We consider (α, β)-approximations for the BLEDP problems. An (α, β)-approximation algorithm is one that obtains for each instance $\mathcal{I}$ with length bound L, $\alpha \cdot \text{OPT}(\mathcal{I})$ edge-disjoint paths each of length at most βL, where $\text{OPT}(\mathcal{I})$ is the size of the optimal solution. For convenience, we refer to an $(\alpha, 1)$-approximation as an α-approximation.

4.1 (α, β)-Approximating (s,t)-BLEDP

4.1.1 Inapproximability results

It is easy to see, by a simple reduction from 2DIRPATH, that the directed version of (s,t)-BLEDP does not have a polynomial time $(\frac{1}{2} + \epsilon, \frac{4}{3} - \epsilon)$-approximation algorithm unless P = NP. This reduction, however, is inexorably tied to the directed case; indeed the undirected version of 2DIRPATH is solvable in polynomial time by the work of Robertson and Seymour [27]. One could possibly attack the undirected case, however, by considering the related NP-hard problem (see [7]) INTEGER2COMMODITY where in we are given an *undirected* graph $G = (V, E)$ and *distinct* vertices $x_1, x_2, y_1, y_2 \in V$, and the objective is to find a maximum collection of edge-disjoint paths, each joining x_i to y_i for $i = 1$ or 2. We note that the half-integral version of INTEGER2COMMODITY flow was shown to be polynomially solvable in a classical paper of Hu [13]. We now give a reduction which shows MAX SNP-hardness of both undirected (s,t)-BLEDP and INTEGER2COMMODITY; a similar result for undirected (s,t)-BLEDP has recently been obtained independently in [3].

Theorem 7 *For both undirected as well as directed graphs, (s,t)-BLEDP with a length bound of six is MAX SNP-hard. Hence, there exist constants α and β, $\alpha < 1$ and $\beta > 1$, such that there is no polynomial-time (α, β)-approximation for (s,t)-BLEDP unless P=NP.*

Proof: We prove the desired claim for undirected graphs only. The proof can be easily amended to apply to directed graphs.

The reduction is from Bounded Occurrence 3-DIMENSIONAL MATCHING (3DM) and follows the reduction presented in [21] which in turn is motivated by ideas from [10]. In an instance $\mathcal{I}$ of 3DM, we are given three disjoint sets $A = \{a_1, a_2, \ldots, a_n\}$, $B = \{b_1, b_2, \ldots, b_n\}$ and $C = \{c_1, c_2, \ldots, c_n\}$, and a set $\mathcal{T}$ of m triples $T_\mu \in A \times B \times C$, $\mu \in [m]$. It is shown in [14] that there exists an $\epsilon_0 > 0$ such it is NP-hard to distinguish between instances $\mathcal{I}$ where there exist n *disjoint* triples in $\mathcal{T}$ (call them "satisfiable" instances) and those where there are at most $(1-\epsilon_0)n$ disjoint triples in $\mathcal{T}$ (call these "ϵ_0-unsatisfied" instances), even if we assume that each element of A, B, C is in the same constant number M of triples in $\mathcal{T}$.[1] We denote the μ^{th} triple T_μ as $(a_{p_\mu}, b_{q_\mu}, c_{r_\mu})$ for some $1 \le p_\mu, q_\mu, r_\mu \le n$.

Create an undirected graph $H = (V, E)$ as follows:

$$\begin{aligned}
V &= \{b_i, c_i : i \in [n]\} \bigcup \{x_\mu, y_\mu : \mu \in [m]\} \\
&\quad \bigcup \{a_{il} : i \in [n], l \in [M-1]\} \bigcup \{s, t\} \\
E &= \{(s, b_i), (c_i, t), (a_{il}, t) : i \in [n], l \in [M-1]\} \\
&\quad \bigcup \{(s, x_\mu), (y_\mu, a_{p_\mu, l}) : \mu \in [m], l \in [M-1]\} \\
&\quad \bigcup \{(b_{q_\mu}, x_\mu), (x_\mu, y_\mu), (y_\mu, c_{r_\mu}) : \mu \in [m]\}
\end{aligned}$$

We define positive integer lengths on every edge of H and consider the problem of finding the maximum number of edge-disjoint s-t paths of total length at most 6 in H. The edges (s, x_μ) for $\mu \in [m]$ have a "length" of 3 each, and the edges (y_μ, c_{r_μ}) have a length of 2, while all other edges have a length equal to 1.

We now claim that (a) [completeness] if the instance $\mathcal{I}$ of 3DM is *satisfiable*, then we can find Mn edge-disjoint paths from s to t of length at most 6, and (b) [soundness] if $\mathcal{I}$ is ϵ_0-unsatisfied, then there are at most $(M - \epsilon_0/2)n$ such s-t paths in H.

Let us first verify the completeness. Suppose $T_{\mu_1}, T_{\mu_2}, \ldots, T_{\mu_n}$ are n *disjoint* triples. Denote $f_i = q_{\mu_i}$ and $g_i = r_{\mu_i}$ for $i \in [n]$. Define the paths $P_i = [s, b_{f_i}, x_{\mu_i}, y_{\mu_i}, c_{g_i}, t]$ for $i \in [n]$, and the paths $Q_\mu = [s, x_\mu, y_\mu, a_{p_\mu, l_\mu}, t]$ for $\mu \in [m] - \{\mu_1, \mu_2, \ldots, \mu_n\}$ and an $l_\mu \in [M-1]$ defined such that no two μ's whose corresponding triples "share" an element of A, have the same value of l_μ. (This is possible to achieve since the fact that each element of A is present in exactly one triple among $T_{\mu_1}, \ldots, T_{\mu_n}$ implies that, for each element $a_i \in A$, there are $M-1$ such μ's in the set $[m] - \{\mu_1, \ldots, \mu_n\}$.) It is now easy to check that the paths P_i and Q_μ are all edge-disjoint and that each has length exactly 6.

To prove soundness, suppose there is a collection $\mathcal{C}$ of $(Mn - \delta n)$ edge-disjoint s-t paths of length at most 6. Clearly at most $(M-1)n$ of these paths use a final edge of the form $a_{p_\mu, l}t$ and hence at least $(1-\delta)n$ of the paths in $\mathcal{C}$ use a final edge of type $c_i t$. Any such

[1] Actually we only need the fact that each element in A occurs in some constant number of triples.

path of length at most 6 is easily seen to be of the form P_i above. The b_i's and c_j's used in these paths are all distinct and yield a set $\mathcal{S}$ of $(1-\delta)n$ triples which have distinct B and C "coordinates". If some a_i is present in say $t > 1$ triples in $\mathcal{S}$, we can only take 1 of these in a 3-matching. But then we also "lost" at least $t-1$ potential paths of the form Q_μ for our collection $\mathcal{C}$. Thus a_i was used to get at most $(M-t)$ paths of the form Q_μ (as opposed to the $M-1$ paths in the completeness case). We can clearly lose at most δn such paths in all, and hence by retaining in $\mathcal{S}$ at most one triple containing any a_i, for each $i \in [n]$, we will be left with a set of at least $(1-2\delta)n$ **disjoint** triples.

We have thus proved that the undirected (s,t)-BLEDP problem (even with a length bound of 6) is MAX SNP-hard, and in fact for any $\epsilon > 0$, there is no polynomial-time $(1-\frac{\epsilon_0}{2M}, \frac{7}{6}-\epsilon)$-approximation for (s,t)-BLEDP unless P=NP. ■

One can easily amend the previous reduction to obtain the following claim.

Corollary 4.1 INTEGER2COMMODITY *is MAX SNP-hard.*

Proof: We create an instance of INTEGER2COMMODITY by taking the graph H in the preceding proof, and splitting s and t into s_1, s_2, t_1, t_2 as follows. The node s_1 will be adjacent only to the nodes b_i and t_1 will be adjacent only to the c_i. Similarly, s_2 is adjacent to the x_μ's and t_2 is adjacent only to the a_{il}'s. One now sees that the $s_i - t_i$ paths are one of the two types prescribed in the original proof. ■

4.1.2 An $(1-\epsilon, 1/\epsilon)$-approximation algorithm for (s,t)-BLEDP

We are given an undirected graph G, length bound L, a source s and a sink t in G. Let p be the optimal number of edge-disjoint paths from s to t, each path of length at most L. In this section, we present a simple algorithm that obtains, for any positive real ϵ, at least $(1-\epsilon)k$ paths, each of length at most L/ϵ.

We first define a minimum-cost flow problem in G from s to t, where the cost of an edge is defined to be its length and the value of the desired flow is p. Since there exists a flow of value p and cost at most pL, the minimum-cost flow algorithm returns a set S of edge-disjoint paths from s to t such that the sum of all of the path lengths is at most pL. By averaging, it follows that at least $(1-\epsilon)p$ s-t paths in S have length at most L/ϵ, thus yielding the desired approximation.

4.2 Hardness of α-approximating BLEDP

4.2.1 Hardness of (s,t)-BLEDP

Theorem 8 *For directed graphs, (s,t)-BLEDP is NP-hard to approximate within a factor of $m^{1/2-\epsilon}$ for any $\epsilon > 0$.*

Proof: The reduction is from 2DIRPATH. Given an instance $[H; x_1, x_2, y_1, y_2]$ of 2DIRPATH, we construct an instance $[G = (V, A'); s, t; L]$ where G is the same directed graph as the one constructed in the proof of Theorem 1 together with two new vertices s, t and arcs joining s to $s_1, s_2, \ldots, s_N$ and joining $t_1, t_2, \ldots, t_N$ to t, and L is a suitable length bound to be mentioned later in the proof.

We now define non-negative integer lengths on the arcs and consider the problem of finding edge-disjoint paths of total length at most L between s and t. In the directed graph

G, the arcs at the bend (the ones incident to d_{ii} for $i \in [N]$) and the arcs which lie wholly within the various copies of H, get a length of zero. The arc (s, s_i) gets a length of i, and the arc (t_i, t) gets a length of $N - i$, for $i \in [N]$, and all other arcs have a length of 1.

We now claim that any s, t path P with total length at most $2N-1$ must use a "bend", i.e must go through d_{rr} for some $r \in [N]$. Indeed, let the path P go through s_i and t_j. Then, if it does not use a bend, then all arcs it uses in the underlying grid of (the skeleton of) G have a length 1, and at least j such "vertical" arcs and $N - i + 1$ such "horizontal" arcs are required for any directed path through s_i and t_j. This, together with the lengths i and $N - j$ on (s, s_i) and (t_j, t) respectively, implies that P has total length at least $i + (N - j) + j + (N - i + 1) = 2N + 1$. Hence, using the same *geometric argument* as in the proof of Theorem 1, we can prove that in the case when there are no edge-disjoint paths between x_1, y_1 and x_2, y_2 in H, the maximum number of edge-disjoint (s, t)-paths in G of length at most $2N - 1$ is one.

Also, when we start from a YES instance of 2DIRPATH, there are N edge-disjoint (s, t)-directed paths of length $2N - 1$ in G, namely the N paths $\mathcal{P}_i$ where $\mathcal{P}_i$ is the (s, t) path that goes through s_i and t_i via the "bend" vertex d_{ii} (i.e it follows i vertical arcs from s_i up to d_{ii} and then $N-i+1$ horizontal arcs to t_i, for an overall length of $i+(i-1)+(N-i)+(N-i) = 2N - 1$).

Thus the gap between the optimum values of the instances created from the YES and NO instances of 2DIRPATH is N, which as before will prove an inapproximability bound of $m^{1/2-\epsilon}$ by choosing N suitably. ∎

4.2.2 Hardness of BLEDP

We now employ an approximation preserving reduction from the independent set problem to establish that BLEDP, even on undirected graphs, is hard to approximate within $m^{1/2-\epsilon}$.

Theorem 9 *Unless* NP = ZPP, BLEDP *on undirected graphs cannot be approximated in polynomial time within a factor of* $m^{1/2-\epsilon}$ *for any* $\epsilon > 0$.

Proof: The reduction is from the independent set problem. Assume we are given a graph $G = (V, E)$ with $V = \{1, 2, \ldots, n\}$. It is known that unless NP = ZPP, it is not possible to distinguish in polynomial time between the cases when $\alpha(G) \geq n^{1-\epsilon_0}$ and when $\alpha(G) \leq n^{\epsilon_0}$ for any fixed $\epsilon_0 > 0$ [12].

Starting from G, we construct a "grid graph" instance of BLEDP $[H; \{(s_i, t_i) : i \in [n]\}]$ as follows. The n sources s_i correspond to the vertices $(i, 0)$ respectively, while the n sinks t_i correspond to $(0, n - i + 1)$ respectively. Each s_i is connected to t_i through a "canonical" path $\mathcal{Q}_i = [s_i, a[i, 1], a[i, 2], \ldots, a[i, n-i], b[i-1, n-i+1], b[i-2, n-i+1], \ldots, b[1, n-i+1], t_i]$ – here each $a[i, j]$ and $b[i, j]$ is to be thought of as *an edge* (with a designated entry and exit point) placed at vertex (i, j) of the plane grid (for $1 \leq i, j \leq n$ with $i + j \leq n$). The adjacency information of G is encoded in H by *identifying* the edges $a[i, n - j + 1]$ and $b[i, n - j + 1]$ whenever vertices i and j (with $i < j$) are adjacent in G – this serves the purpose that whenever i and j are adjacent in G, the canonical paths $\mathcal{Q}_i$ and $\mathcal{Q}_j$ will share an edge (viz. the "identified" copy of $a[i, n - j + 1]$ and $b[i, n - j + 1]$), and thus cannot both be simultaneously routed.

We assign a length of 2 to the following edges: (i) $(s_i, a[i, 1])$ for $1 \leq i < n$, and $(b[1, n - i + 1], t_i)$ for $1 < i \leq n$; (ii) $a[i, j]$ and $b[i, j]$ for i, j such that $i + j \leq n$; and (iii) edges joining the "exit" of $a[i,j]$ (respectively $b[i,j]$) to the "entry" of $a[i,j+1]$ (respectively $b[i-1,j]$) for i,j such that $i>1$ and $i+j<n$. We assign a length of 1 to all other edges.

We now claim that the only s_i-t_i path in H of total length at most $4n-3$ is the canonical path $\mathcal{Q}_i$. Indeed let P_i be any shortest s_i-t_i path. Then, arguing informally, it can be shown that if P_i "leaves" a column (or a row) at a vertex v_1 and comes back to meet it again at v_2, then the path which for the v_1-v_2 leg of P_i uses the v_1-v_2 segment of that column (or row as the case may be), will have strictly smaller length than P_i, a contradiction. Thus the shortest s_i-t_i path lies wholly within the portion of the grid $\{(p,q) : p \leq i, q \leq n-i+1\}$, and now it is easy to check that P_i is the unique s_i-t_i shortest path and that P_i has a total length of $4n-3$.

Thus, starting from G, we can, in polynomial time, construct a graph H with n source-sink pairs (s_i,t_i) such that the maximum number of pairs which can be joined by edge-disjoint paths of length at most $4n-3$, equals $\alpha(G)$. The gap of $n^{1-2\epsilon_0}$ in $\alpha(G)$ thus clearly translates into a gap of $m^{1/2-\epsilon_0}$ for the BLEDP instance we create, and the proof is completed. ■

5 Greedy Algorithms for USF, ISF and BLEDP

We now show that the hardness bounds for all problems considered can be matched (ignoring certain polylogarithmic factors) by suitable adaptations of a greedy algorithm. We give the "core" algorithm in Figure 3, although subroutines must be tailored for the different problems considered. In the code description, a network refers to a directed graph together with integer capacities on the arcs. The operation of *decrementation* of a network by a flow f, results in a new network for which the capacity on an arc e is precisely f_e units less than before.

```
GREEDY MULTICOMMODITY ROUTING (GREED)
{
/* Input is a directed network where some arcs have length 0
and others have length 1 */
    S = ∅
    While (There is an unsatisfied demand)
        For each demand s in D − S
            Find a shortest flow f for s in the residual network
        EndFor
        If no routings are found, then Output Current Solution
        Let s be such a demand with an overall minimum routing f
        Decrement the residual network by the flow f
        Add s to S
    EndWhile
    Output Current Solution
}
```

Figure 3: The "core" greedy algorithm

The flows (or routings) f found by GREED depend (e.g., split or unsplit) on the problem version being addressed. In any case, "shortest" will always refer to minimizing the measure $\sum_e f_e l_e$ where l_e is either 0 or 1. In the cases of USF and ISF, the problem of finding such an f reduces to solving a shortest path or mincost flow problem respectively. For BLEDP with length bound L, the "shortest flow" f is a source-sink path with the fewest number of edges among all of the source-sink paths of length L. In the following, we briefly describe a simple subroutine that computes such a path. It is sufficient to provide an algorithm that given a directed graph $D = (V, A)$, a source s, a sink t, and a positive integer k, determines a path that has length at most L and has less than k edges, if such a path exists. We now define such an algorithm.

We construct a layered directed graph $D' = (V', A')$ in which the set V' consists of k copies of each vertex in V. We denote the ith copy, $1 \leq i \leq k$ of a vertex v in V by v_i. For every arc (u, v) in E, we have the following $k-1$ arcs in A', each of which has the same length as that of (u, v): (u_i, v_{i+1}) for $1 \leq i < k$. In addition, for each v in V, we have the following $k-1$ arcs in A', each of length zero: (u_i, u_{i+1}) for $1 \leq i < k$. It is now easy to see that there is a path in D from s to t that has length at most L and has at most k arcs if and only if there is a path in D' from s_1 to t_k that has length at most L. Moreover, given a path P' in D' from s_1 to t_k that has length at most L, we can construct in polynomial time a path in D from s to t that has length at most L and has at most k arcs. To complete the description of the algorithm, we note that we can determine the existence of path P' by finding the shortest path between s_1 and t_k.

In [16], [20], and [22], the greedy algorithm is analyzed to show that it satisfies at least $O_u/\sqrt{m}$ demands for an instance of EDP (also see [5]). (Recall that O_u denotes the optimum value for a given USF instance.) Their arguments may be extended easily to give $\sqrt{u(A)}$-approximations for unsplittable problems with general demands and capacitated networks (where $u(A) = \sum_e u(e)$). We improve this bound for capacitated networks by invoking the greedy algorithm several times.

Theorem 10 *Consider a unit-profit instance of* USF *in a directed graph D. If $d_{max} \leq d_{min}\Delta$ for some integer Δ, then the greedy algorithm may be used to find a solution which satisfies at least $\frac{O_u}{2\Delta\sqrt{m}}$ demands.*

Proof: Consider a guess O for O_u and set $u_e^O = \min\{u_e, \frac{d_{min} O}{\sqrt{m}}\}$. Call an arc *clipped* if $u_e^O \neq u_e$. We associate a length l_e with each arc in the following manner: e is assigned a length of 0 if it is clipped and 1 otherwise. Run the greedy algorithm on the graph D^O obtained by reducing the capacity of each arc e to u_e^O. Let $P_1, P_2, \ldots, P_t$ be the flow paths chosen by the greedy algorithm in D^O.

Let $\mathcal{Q}$ be those demands which are satisfied in some optimal solution but not by the greedy algorithm. Let Q be a path in the optimal solution satisfying some demand in $\mathcal{Q}$. It follows that there is an arc e in D such that (i) Q uses the arc e, and (ii) the greedy algorithm is using at least a capacity of $u_e^O - d_{max}$ on the arc e. Otherwise, the greedy algorithm could have satisfied this demand along the flow path Q. Now if e were a clipped arc then the greedy algorithm must have routed $(\frac{O d_{min}}{\sqrt{m}} - d_{max})/d_{max} \geq \frac{O}{2\Delta\sqrt{m}}$ demands and we are done. So we may assume this is not the case and hence each such Q must contain an arc of length 1 from D^O that is in common with some P_i.

We now build on the ideas used in [20, 22] and analyze how many demands in the set $\mathcal{Q}$ are not satisfied due to a greedily selected flow path P_i. Let n_i denote the number of arcs of length 1 in the greedy flow path P_i. Consider any arc e and the greedy flow paths routed through it. We can imagine e as a bin of capacity u_e^O such that each successive greedy flow path is allocated a contiguous block of capacity on e. In an analogous manner, the optimal flow paths corresponding to the demands in Q can be viewed to have been allocated successive contiguous blocks of capacity on e. A greedy flow path P_i is said to *block* an optimal flow path Q if i is the least index such that (a) P_i and Q share a common arc e of length 1, and (b) the capacity block of P_i on e overlaps with the capacity block of path Q. Since each blocked optimal path Q uses up a capacity of d_{min} at least and each greedy path P_i uses a capacity of d_{max} at most, we deduce that any P_i may block at most $n_i\lceil d_{max}/d_{min}\rceil \leq n_i\Delta$ flow paths in $\mathcal{Q}$. Let k_i denote the number of optimal flow paths (corresponding to demands in $\mathcal{Q}$) that are blocked by P_i. It follows that $k_i \leq n_i\Delta$. But also by the definition of the greedy algorithm, we have that each such blocked flow must have used at least $n_i d_{min} \geq (k_i/\Delta)d_{min}$ units of length 1 capacity. Hence the total length 1 capacity used by the unrouted paths from the optimal solution is at least $\frac{d_{min}}{\Delta}\sum_i k_i^2$. Combining this with the observations that all length 1 arcs are unclipped and that the total available length 1 capacity is at most $\sum_{e\in A(D)} u_e^O$, we obtain

$$\frac{d_{min}}{\Delta}\cdot\frac{(\sum_{i=1}^t k_i)^2}{t} \leq \frac{d_{min}}{\Delta}\sum_{i=1}^t k_i^2 \leq \sum_{e\in A(D)} u_e^O \leq \sqrt{m}Od_{min}. \tag{6}$$

Since $\sum_i k_i = |\mathcal{Q}| = O_u - t$, we obtain $\frac{(O_u-t)^2}{t} \leq \Delta\sqrt{m}O$. Running the above procedure for various possible values of O (guessed within a factor of 2 by doubling), we may deduce that $t \geq \frac{O_u}{2\Delta\sqrt{m}}$ as required. ■

If $\Delta > 2$, we can divide the demands into $O(\log\Delta)$ classes and do the above analysis for the class that has the largest number of pairs satisfied by an optimal solution. This would give us a solution that satisfies at least $O(\frac{O_u}{\log\Delta\sqrt{m}})$ demands. The preceding argument applies equally well (with $\Delta = 1$ of course) to BLEDP and so we obtain the following result.

Theorem 11 *There is a polynomial-time $O(\sqrt{m})$ approximation algorithm for* BLEDP. ■

At first glance, it may appear that the proof of Theorem 10 also gives a similar bound for the integral splittable problem. There is a snag, however; we may only deduce the following.

Theorem 12 *Consider a unit-profit instance of the* ISF *problem. Then by applying the greedy algorithm $O(\log n)$ times, we may find a solution which satisfies at least $\frac{O_{is}}{2\sqrt{md_{max}}}$ demands.*

Proof: Mimic the proof of Theorem 10 directly, except clip each capacity at $\frac{O\sqrt{d_{max}}}{\sqrt{m}}$. The proof breaks down only because we cannot guarantee that the flow P_i blocks at most $n_i\Delta$ flow systems from an integral splittable optimum. Indeed, we can only deduce that P_i must use at least k_i units of length 1 capacity, and hence so do each of the flows which it blocks. Equation (6) becomes $\sum_i k_i^2 \leq \sqrt{md_{max}}O$, and the rest then follows. ■

Returning to the general versions with profits and arbitrary demands, the repeated greedy algorithm, together with techniques from the proof of Theorem 6, gives us an upper bound on the approximation factor that nearly matches the hardness result of Corollary 3.1.

Corollary 5.1 *The general* USF *problem with arbitrary polynomially bounded demands can be approximated to within a factor of* $O(\sqrt{m}\log^2 m)$. *The general* ISF *problem with arbitrary polynomially bounded demands can be approximated to within a factor of* $O(\sqrt{m d_{max}}\log^2 m)$.
■

Acknowledgements

We are grateful to Jon Kleinberg, Stavros Kolliopoulos, and Éva Tardos for guiding us through the extensive literature on this subject.

References

[1] Y. Aumann and Y. Rabani. Improved Bounds for All Optical Routing. *Proceedings of 6th ACM-SIAM Symposium on Discrete Algorithms*, 1995, pp. 567–576.

[2] A. Baveja and A. Srinivasan. Approximation algorithms for disjoint paths and related routing and packing problems. *Submitted*, January 1998.

[3] A. Bley. On the complexity of vertex-disjoint length-restricted path problems. *Konrad-Zuse-Zentrum für Informationstechnik Berlin Tech. Report SC-98-20*, 1998.

[4] H. Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23:493–507, 1952.

[5] E. A. Dinitz. Algorithm for solution of a problem of maximum flow in networks with power estimation. *Soviet Math. Dokl.*, Vol. 11 (1970), pp. 1277-1280.

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

[7] S. Even, A. Itai and A. Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM Journal on Computing*, Vol. 5, No. 4 (1976), pp. 691-703.

[8] S. Fortune, J. Hopcroft and J. Wyllie. The directed subgraph homeomorphism problem. *Theoretical Computer Science*, Vol. 10, No. 2 (1980), pp. 111–121.

[9] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness.* Freeman, 1979.

[10] N. Garg, V. Vazirani and M. Yannakakis. Primal-dual approximation algorithms for integral flow and multicut in trees. *Algorithmica*, 18 (1997), pp. 3–20. (Preliminary version in *Proceedings of 20th International Colloquium on Automata, Languages, and Programming*, 1993, pp. 64–75.)

[11] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

[12] J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *ECCC Technical Report TR97-038.* (Preliminary version in *Proceedings of the 37th Symposium on the Foundations of Computer Science*, 1996, pp. 627–636.)

[13] T. C. Hu. Multi-commodity Network Flows. *Operations Research*, 11(1963), pp. 344–360.

[14] V. Kann. Maximum bounded 3-dimensional matching is MAX SNP-complete. *Information Processing Letters*, 37(1991), pp. 27–35.

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

[16] J. M. Kleinberg. *Approximation algorithms for disjoint paths problems.* PhD thesis, MIT, Cambridge, MA, May 1996.

[17] J. M. Kleinberg. Decision algorithms for unsplittable flow and the half-disjoint paths problem. *Proc. of STOC '98*, pp. 530–539.

[18] J. M. Kleinberg and É. Tardos. Approximations for the disjoint paths problem in high-diameter planar networks. *Journal of Computer and System Sciences*, 57, pp. 61-73, 1998. (Preliminary version in the *Proceedings of the 27th Symposium on the Theory of Computing*, 1995, pp. 26–35.)

[19] J. M. Kleinberg and É. Tardos. Disjoint Paths in Densely Embedded Graphs. *Proceedings of the 36th Symposium on the Foundations of Computer Science*, 1995, pp. 52–61.

[20] S. G. Kolliopoulos. Exact and approximation algorithms for network flow and disjoint-path problems. *PhD Thesis*, Dartmouth College, Hanover, NH, August 1998.

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

[22] S. G. Kolliopoulos and C. Stein. Approximating disjoint-path problems using greedy algorithms and Packing Integer Programs. *Integer programming and Combinatorial Optimization*, 1998.

[23] F. T. Leighton and S. B. Rao. An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms. *Proceedings of the 29th Symposium on the Foundations of Computer Science*, 1988, pp. 422–431.

[24] B. Ma and L. Wang. On the inapproximability of disjoint paths and minimum steiner forest with bandwidth constraints. *Journal of Computer and Systems Sciences*, to appear.

[25] W. S. Massey. *Algebraic Topology: An Introduction.* Graduate texts in mathematics 56, Springer-Verlag, 1967.

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

[27] N. ROBERTSON AND P. D. SEYMOUR. Outline of a disjoint paths algorithm. In B. Korte, L. Lovász, H. J. Prömel, and A. Schrijver, Eds., *Paths, Flows and VLSI-Layout*. Springer-Verlag, Berlin, 1990.

[28] A. SRINIVASAN. Improved approximations for edge-disjoint paths, unsplittable flow, and related routing problems. *Proceedings of the 38th Symposium on the Foundations of Computer Science*, 1997, pp. 416–425.