THEORY OF COMPUTING, Volume 2 (2006), pp. 137–146
http://theoryofcomputing.org

An $O(\sqrt{n})$ Approximation and Integrality Gap for Disjoint Paths and Unsplittable Flow

Chandra Chekuri Sanjeev Khanna F. Bruce Shepherd

Received: December 8, 2005; published: May 4, 2006.

Abstract: We consider the maximization version of the edge-disjoint path problem (EDP). In undirected graphs and directed acyclic graphs, we obtain an $O(\sqrt{n})$ upper bound on the approximation ratio where n is the number of nodes in the graph. We show this by establishing the upper bound on the integrality gap of the natural relaxation based on multicommodity flows. Our upper bound matches within a constant factor a lower bound of $\Omega(\sqrt{n})$ that is known for both undirected and directed acyclic graphs. The best previous upper bounds on the integrality gaps were $O(\min\{n^{2/3}, \sqrt{m}\})$ for undirected graphs and $O(\min\{\sqrt{n \log n}, \sqrt{m}\})$ for directed acyclic graphs; here m is the number of edges in the graph. These bounds are also the best known approximation ratios for these problems. Our bound also extends to the unsplittable flow problem (UFP) when the maximum demand is at most the minimum capacity.

ACM Classification: C.2.0, F.2.2, G.1.6, G.3

AMS Classification: 68W20, 68W25, 90C59

Key words and phrases: edge-disjoint paths, unsplittable flow, routing, approximation algorithm, integrality gap, undirected graphs, directed acyclic graphs

1 Introduction

The *edge-disjoint path problem* (EDP) in undirected graphs is defined as follows. We are given an undirected graph $G = (V,E)$ and k node pairs $s_1t_1, s_2t_2, \ldots, s_kt_k$ (a pair can occur multiple times). The decision version asks if there is a collection of edge-disjoint paths $P_1, P_2, \ldots, P_k$ such that for $1 \le i \le k$,

 DOI: 10.4086/toc.2006.v002a007

P_i is a path from s_i to t_i. We consider the maximization version where we seek to find the largest subset of the k pairs that can be connected by edge-disjoint paths. In the weighted case, each pair s_it_i has a non-negative weight w_i and we seek to find the largest weight subset of pairs to connect. In the directed version, the *arc-disjoint path problem*, each demand consists of an ordered pair (s_i, t_i) and we require a directed path from s_i to t_i. For expediency, however, we refer to all versions as EDP. We also consider a generalization of EDP called the *unsplittable flow problem* (UFP). In this problem the edges of the graph G have non-negative integer capacities given by a function $c : E \rightarrow Z^+$ and each of the k pairs has an integer demand, d_i for pair i. A subset S of the pairs is *routed* if there is a path collection $P_i, i \in S$ such that P_i connects s_i to t_i and for each edge $e \in E$, $\sum_{i \in S : e \in P_i} d_i \leq c(e)$. We say that an instance of UFP satisfies the *no-bottleneck* assumption if $d_{\max} = \max_i d_i \leq \min_e c(e) = c_{\min}$. Note that EDP is a special case of UFP with $d_i = 1$, $1 \leq i \leq k$ and $c(e) = 1$, $e \in E$. We refer to the special case of UFP when only the d_i's are 1 as *capacitated EDP*.

EDP and UFP are fundamental problems in combinatorial optimization and also arise in a number of applications. These problems are strongly NP-hard even in very restricted settings and also hard to approximate. Consequently there is a large body of literature on special cases and variants. The natural multicommodity flow relaxation (see Section 2 for more details) plays an important role in providing an upper bound on the optimum value. In this paper we focus on approximation algorithms and integrality gaps of the flow relaxation for arbitrary instances.

We briefly review known results for general graphs. For a more comprehensive view, including results for special classes of EDP, see [7, 10, 17, 11]. In discussing results for EDP we normally assume that the underlying graph is a simple graph and use n and m to refer to the number of nodes and edges (arcs) respectively. For EDP it is known that the integrality gap of the flow LP is $\Omega(\sqrt{n})$ even in undirected planar graphs [8]. A first upper bound on the approximation ratio for EDP was $O(\sqrt{m})$ [10]. This also holds for capacitated EDP and in fact can be obtained using a simple greedy algorithm that iteratively picks an arbitrary unconnected pair and picks a shortest feasible path for it [12, 9, 14]. The best upper bound on the integrality gap for EDP is $O(\min\{n^{2/3}, \sqrt{m}\})$ in undirected graphs [4] and $O(\min\{n^{2/3}\log^{1/3} n, \sqrt{m}\})$ in directed graphs [18]. It has been an interesting open problem to bridge the gap between the upper and lower bounds on the integrality gap for EDP. In [4], the greedy algorithm was combined with an algorithm based on single source flow (for a special class of instances called *v-separable instances*) to obtain a bound of $O(\sqrt{n \log n})$ in directed acyclic graphs (DAGs). Guruswami et al. [9] showed that EDP in directed graphs is hard to approximate within a factor of $O(m^{1/2-\varepsilon})$ unless $P = NP$. Their result applies to sparse graphs, and hence as a function of n, it establishes a hardness factor of $\Omega(n^{1/2-\varepsilon})$. Ma and Wang [15] showed that EDP in DAGs is hard to approximate within a factor of $\Omega(2^{\log^{1-\varepsilon} n})$ unless $NP \subseteq DTIME(n^{\text{polylog}(n)})$. For undirected graphs, Andrews et al. [1] showed that EDP is hard to approximate within a factor of $\Omega(\log^{1/2-\varepsilon} n)$ unless $NP \subseteq ZPTIME(n^{\text{polylog}(n)})$. In [4] it was conjectured that that the approximation threshold for EDP in directed graphs and the integrality gap of EDP in undirected and directed graphs is $\Theta(\sqrt{n})$. Here we improve the upper bound to $O(\sqrt{n})$ for undirected graphs and DAGs. In fact we prove the following stronger result.

Theorem 1.1. *The integrality gap of the relaxation based on multicommodity flows is $\Theta(\sqrt{n})$ for capacitated EDP in undirected graphs and DAGs.*

Algorithms for UFP have typically been based on those for EDP. We distinguish between instances that satisfy the no-bottleneck assumption and those that do not. First we discuss the general case where

$d_{\max}$ can be larger than $c_{\min}$. The best upper bound known in this case is $\tilde{O}(\sqrt{m}\log d_{\max}/c_{\min})$ [9] even in directed graphs. Azar and Regev [2] showed that unless $P = NP$, UFP is hard to approximate in directed graphs within a factor of $\Omega(m^{1-\varepsilon})$ on sparse instances which translates to a hardness factor of $\Omega(n^{1-\varepsilon})$ as a function of n. Note that the hard instances constructed in [2] have the property that $d_{\max}/c_{\min}$ is exponential in m. For instances satisfying the no-bottleneck assumption, Baveja and Srinivasan [3] obtained an $O(\sqrt{m})$ approximation. Kolman and Schiedeler [14] showed that if $w_i = d_i$ then an $O(\sqrt{m})$ approximation can be obtained even if $d_{\max} \geq c_{\min}$. Under the same assumption on the weights, Kolman [13] extended the EDP bounds in [4, 18] to UFP. It is known from the work of Kolliopoulos and Stein [12] that, for a certain class of packing integer programming problems (PIPs) that they call column restricted packing integer programs (CPIPs), the integrality gap for instances that satisfy the no-bottleneck assumption is within a constant factor of the integrality gap for unit-demand instances (see [6] for some refinements and extensions). From this we immediately obtain the following corollary to Theorem 1.1.

Corollary 1.2. *The integrality gap of the relaxation based on multicommodity flows is $\Theta(\sqrt{n})$ for no-bottleneck instances of UFP in undirected graphs and DAGs.*

We prove Theorem 1.1 in Section 3, after setting up some notation and proving a lemma on single source flows in Section 2. We remark that Thanh Nguyen [16] has independently obtained an approximation ratio of $O(\sqrt{n})$ for DAGs, and subsequent to our work, obtained an alternative $O(\sqrt{n})$ approximation in undirected graphs.

2 Preliminaries

An instance of capacitated EDP consists of a graph $G = (V,E)$, integer edge capacities specified by $c : E \to Z^+$, and k node pairs $s_1t_1, s_2t_2, \ldots, s_kt_k$. Each pair s_it_i has a non-negative weight w_i and demand $d_i = 1$. In a directed graph instance, the node pairs are ordered: $(s_1,t_1),(s_2,t_2),\ldots,(s_k,t_k)$. For convenience of notation we assume that any pair of nodes occurs at most once in the given instance. A *terminal* is a node that is the end point of some pair in the given instance.

Multicommodity flow LP formulation: For the given instance, we let $\mathcal{P}_i$ denote the set of all paths joining s_i and t_i in G, and let $\mathcal{P} = \cup_i \mathcal{P}_i$. The following multicommodity flow relaxation is used to obtain an upper bound on an optimal solution to the given instance. For each path $P \in \mathcal{P}$ we have a variable $f(P)$ which is the amount of flow sent on P. We let x_i denote the total flow sent on paths for pair i. The LP relaxation is the following.

$$
\begin{aligned}
&\max \sum_{i=1}^{k} w_i x_i \quad \text{s.t} \\
&x_i - \sum_{P \in \mathcal{P}_i} f(P) \;=\; 0 \qquad 1 \leq i \leq k \\
&\sum_{P : e \in P} f(P) \;\leq\; c(e) \qquad \forall e \in E \\
&x_i, f(P) \;\in\; [0,1] \qquad 1 \leq i \leq k, P \in \mathcal{P}
\end{aligned}
$$

We work with the above exponential size path formulation for discussion's sake, but there is an equivalent compact formulation that can be used for computational purposes. We also remark that for any fixed $\varepsilon > 0$, efficient combinatorial algorithms are known for obtaining a $(1+\varepsilon)$-approximation to the above LP. Let OPT denote the optimal value of the LP on a given instance. In a solution, we call a path P fractional if $f(P) \in (0,1)$, otherwise $f(P) \in \{0,1\}$ and P is integral. If the total weight of flow on integrally routed paths is more than OPT$/2$, then we already obtain a 2-approximation. The interesting and difficult case is when the fractionally routed paths contribute most of OPT and we focus on this case. From standard polyhedral theory, the number of fractionally routed paths in a basic solution to the LP is at most m. Therefore we can assume that $c(e) \leq m$ for all edges. By making parallel copies of edges, in the following, we assume that all edges in G have unit capacity.

2.1 Incremental augmentation of directed flow

Let $G = (V, A, c)$ be a directed graph with integer arc capacities given by c. For $S \subset V$, we denote by $\delta_G^+(S)$, or simply $\delta^+(S)$ if G is clear from the context, the set of arcs (u,v) such that $u \in S$ and $v \in V \setminus S$. Similarly $\delta_G^-(S)$ denotes the set of arcs (u,v) with $u \in V \setminus S$ and $v \in S$. Let $s_1, s_2, \ldots, s_k$ be distinct nodes (terminals) that seek to send flow to a sink node t. A non-negative vector $(b_1, b_2, \ldots, b_k)$ is a *feasible* flow vector if the terminals can route $\sum_i b_i$ flow to t with b_i flow originating at s_i for $1 \leq i \leq k$. Let B be the set of all feasible flow vectors. For a vector $b \in B$, let $F(b) = \sum_i b_i$ denote the *total flow*. For $b \in B$ let $I(b)$ be the index set of terminals that have integer flow, that is, $i \in I(b)$ iff b_i is an integer.

Theorem 2.1. *Given $b \in B$ and $j \notin I(b)$ with $b_j > 0$, we can compute $b' \in B$ in polynomial time with $b'_j = \lceil b_j \rceil$ and $F(b') \geq F(b)$ such that*

- *$b'_\ell \leq \lceil b_\ell \rceil$ for $1 \leq \ell \leq k$, and*
- *$b'_i = b_i$, for $i \in I(b)$.*

Proof. Let f be a flow that demonstrates the feasibility of b. We obtain an auxiliary graph G' from G and f in the usual way: for each arc $a \in A$ with positive flow, we retain the arc in G' if $f(a) < c(a)$ and assign it capacity $c(a) - f(a)$. If $f(a) > 0$ we add a reverse arc $\bar{a}$ in G' of capacity $f(a)$. In G' we look for a directed path from s_j to either t or some terminal s_h with $h \notin I(b) \cup \{j\}$. If we can find such a directed path, we can increase b_j while not changing the net flow out of any terminal in $I(b)$, or decreasing the total flow into t. If the augmenting path allows us to increase b_j to $\lceil b_j \rceil$ we are done. Otherwise we find another augmenting path and repeat. We need only to show that if b_j is not integral, then we can always find an augmenting path of the above type. Suppose not. Let S be the set of all nodes reachable from s_j in G'. It follows that t and no terminal s_h with $h \notin I(b) \cup \{j\}$ belongs to S. Since no arcs leave S in G', any $a \in \delta_G^-(S)$ has zero flow, in other words $f(a) = 0$. Further, any arc $a \in \delta_G^+(S)$ is at capacity, $f(a) = c(a)$. This implies that $\sum_{a \in \delta_G^+(S)} c(a) = \sum_{i \in S} b_i$. Note that the only terminals in S are s_j and some terminals from $I(b)$. Thus the expression $\sum_{i \in S} b_i$ cannot be integral since b_j is not integral while the rest of the summands are. This is a contradiction since $\sum_{a \in \delta_G^+(S)} c(a)$ is integral. □

3 Proof of Theorem 1.1

In this section, we prove Theorem 1.1. The proof for undirected graphs differs from that for DAGs but they share a common initial phase. For simplicity of description, we focus on the cardinality case ($w_i = 1$ for $1 \le i \le k$). We discuss the extension to the weighted case later.

Consider an optimal multicommodity flow solution for a given instance of EDP and let $f = \sum_i x_i$ be the total flow in the solution. For each pair i we decompose its flow into flow paths (alternatively we can simply use the flow paths given by the solution). Suppose more than $f/2$ flow is routed on flow paths of length at most $\sqrt{n}$. Then we can greedily build a solution of value $\Omega(f/\sqrt{n})$ as follows. First, we remove all the flow on paths of length more than $\sqrt{n}$, by assumption we have at least $f/2$ flow left. Pick a flow path P corresponding to some unrouted pair, say pair i. Route the pair along the path P and remove the edges of P from the graph. In this process we discard all flow on paths that use any edge of P as well as any flow on paths other than P that carry flow for the pair i. Since the length of the path P is at most $\sqrt{n}$, the total flow that we discard is at most $\sqrt{n}+1$. Repeating this process until there is no more flow ensures that we route $\Omega(f/\sqrt{n})$ pairs.

We now focus on the case when more than $f/2$ flow is routed along paths of length greater than $\sqrt{n}$. Let $\alpha = f/\sqrt{n}$. If $\alpha \le 1$, it suffices to route any one pair in the EDP instance to obtain an $O(\sqrt{n})$ integrality gap. Otherwise, since each flow path is of length at least $\sqrt{n}+1$, by the pigeonhole principle, there exists a node v such that at least $f\sqrt{n}/n = \alpha$ flow is routed through v . Discard all flow that does not go through v; we show how to route $\Omega(\alpha)$ pairs using the flow that goes through v. The algorithms for the undirected and directed acyclic graphs differ in this second phase and we describe them separately.

The above scheme that combines the greedy analysis with the analysis for the case when a large amount of flow is routed through a single node v is from [4] where it was used to obtain an $O(\sqrt{n \log n})$ approximation for the DAG case.

In the rest of the section, for convenience of notation, we will assume that a node in the graph is a terminal of at most one of the pairs $s_1t_1, s_2t_2, \ldots, s_kt_k$. Otherwise, we can add dummy nodes to ensure this property; note that we are aiming for a constant factor approximation in this second stage and hence the increase in the number of nodes does not contradict our overall goal of an $O(\sqrt{n})$ approximation. With this assumption, we can specify the pairs by a matching M on the terminals.

3.1 Undirected graphs

Let x_i denote the flow for pair i that is routed through v. This implies that s_i and t_i each send x_i amount of flow to v. For a terminal $a \in V$ we let $y(a)$ denote its flow through v, thus $y(s_i) = y(t_i) = x_i$. We have that $\sum_i x_i = \alpha$. Given any $0 < \varepsilon < 1/2$, we either transform x into another feasible solution x' with several additional properties or we can directly find an edge-disjoint routing for $\Omega(\varepsilon\alpha)$ pairs. In the former case, we perform an additional step to recover an edge-disjoint routing for $\Omega(\varepsilon\alpha)$ pairs from the solution x'. The solution x' satisfies the following properties: (i) for $1 \le i \le k$, $x'_i = 0$ or $x'_i = 1-\varepsilon$, (ii) $\sum_i x'_i = \Omega(\varepsilon\alpha)$, and (iii) x' is a feasible multicommodity flow for the pairs such that all flow still goes through v. Suppose we can obtain a solution x' with the properties described above. Let $M' \subseteq M$ be the matching on pairs i such that $x'_i > 0$. Let X be the set of end points of M'. We have that $\sum_i x'_i = (1-\varepsilon)|M'| = (1-\varepsilon)|X|/2$. We solve a single source flow problem in a modified graph obtained by adding a super-sink t connected to each terminal a with an end point in M' by a directed arc of capacity one; we find a maximum flow from

v to t. By construction, there is a total flow of at least $2\sum_i x'_i$ between v and t. Further, since the graph has integer capacities, we can choose this maximum flow to be integral which induces a collection of edge-disjoint paths from v to a subset of terminals $X' \subset X$ where $|X'| \geq (1-\varepsilon)|X|$. Let M'' be the largest sub-matching of M' induced by the terminals in X'. Simple calculations show that $|M''| \geq (1-2\varepsilon)|M'|$. Hence, if we choose $\varepsilon = 1/4$, we obtain a matching M'' of size $\Omega(\alpha)$ such that the end-points of M'' are connected to v be edge-disjoint paths and hence the pairs corresponding to M'' are routed in G.

The transformation: We accomplish the transformation of x to x' using a clustering scheme from [5]. For a graph H and $\gamma \geq 0$, let γH denote the graph H with capacities of the edges set to γ. We find edge-disjoint and connected subgraphs called *clusters* $G_1 = (V_1, E_1), G_2 = (V_2, E_2), \ldots, G_p = (V_p, E_p)$ with the following properties. Each terminal a is assigned to exactly one of the clusters with $l(a)$ denoting the index of the cluster that it is assigned to. Further for each cluster G_i we ensure that $1/\varepsilon \leq \sum_{a:l(a)=i} y(a) \leq 2/\varepsilon$. It follows that $p = \Omega(\varepsilon\alpha)$. We give a sketch of the clustering scheme and refer the reader to [5] for more details. Consider an arbitrary rooted spanning tree T of G. Let u be a deepest node in T such that $\sum_{a\in T_u} y(a) \geq 1/\varepsilon$; here T_u is the subtree of T rooted at u. Let $u_1, u_2, \ldots, u_h$ be the children of u in T and let T_u^i denote the subtree obtained from T_u by removing the children $u_{i+1}, \ldots, u_h$ and their subtrees from T_u. Let j be the smallest index such that $\sum_{a\in T_u^j} y(a) \geq 1/\varepsilon$. We let G_1 be the graph induced by the nodes in T_u^j and set $\ell(a) = 1$ for all terminals $a \in V(G_1)$. By the choice of u and j it follows that $\sum_{a\in V(G_1)} y(a) \leq 2/\varepsilon$. We remove all nodes in G_1 from T except u, reduce $y(u)$ to 0 if u is a terminal, and apply the procedure above iteratively to create the required clusters. Note that the clusters are not necessarily node-disjoint but they are edge-disjoint and connected. Once the clusters are formed, we create a multi-graph C that has one node per cluster. For each pair $s_i t_i$ we add an edge between cluster $l(s_i)$ and $l(t_i)$ – this might be a self-loop if $l(s_i) = l(t_i)$. We find a maximal independent set of edges F in this cluster graph – we note that we are allowed to pick a self-loop in the set. It is relatively easy to argue that the cardinality of this set is $\Omega(p)$ using the fact that $1/\varepsilon \leq \sum_{a:l(a)=i} y(a) \leq 2/\varepsilon$ for each G_i; each edge that is picked can eliminate other edges of total flow of at most $4/\varepsilon$. Let F' be the set of self-loops in F. We claim that the pairs associated with the edges in F' can be routed in G via edge-disjoint paths. This follows because a self-loop corresponds to a pair with both end points in the same cluster; recall that each cluster is connected and the clusters are edge-disjoint. Thus, if $|F'| \geq |F|/2$ we can route $\Omega(p)$ pairs. Otherwise we consider the pairs corresponding to the edges in $F \setminus F'$. Let Z be the terminals that are the end points of the pairs corresponding to these $\Omega(p)$ edges.

The terminal from Z with $l(a) = i$ is termed the *representative* for cluster G_i. We obtain x' by setting $x'_i = 1-\varepsilon$ for each pair with an end point in Z and $x'_i = 0$ otherwise. To argue that x' satisfies the desired properties, we observe that the terminals in Z can each simultaneously send a flow of $(1-\varepsilon)$ to v, in the following manner. The representative terminal a in G_i can send one unit of flow to other terminals in G_i such that a terminal $a' \in G_i$ receives flow of at most $\varepsilon y(a')$. This follows from the fact that G_i is connected and $\sum_{a':l(a')=i} y(a') \geq 1/\varepsilon$. By scaling down the flow, a can send $(1-\varepsilon)$ units of flow to these same terminals in the graph $(1-\varepsilon)G_i$. Now each terminal a' in G_i that received $\varepsilon y(a')$ flow can send it to v. Note that initially each terminal a' in G could send $y(a')$ flow to v, hence it can send $\varepsilon y(a')$ flow to v in the graph εG. Since the G_i's are edge-disjoint and $\varepsilon < 1/2$, from the above description, we have that the cluster representatives can each send $(1-\varepsilon)$ flow each to v in the graph G. This finishes the proof of the transformation.

3.2 Directed acyclic graphs

The clustering scheme that was at the heart of the proof for the undirected graph case does not apply in the directed graph setting. When the graph is acyclic, however, we can reduce the problem to a highly structured instance called a v-separable instance [4] as follows. Let $G_1 = G[V_1]$ where V_1 is the set of nodes that have a directed path to v in G. Let $G_2 = G[V_2]$ where V_2 is the set of nodes that have a directed path from v in G. Since G is acyclic, $V_1 \cap V_2 = \{v\}$ and for any pair (s_i, t_i) that sends flow through v, $s_i \in V_1$ and $t_i \in V_2$. Let $H = G_1 \cup G_2$. Note that all the flow for a pair (s_i, t_i) that goes through v in G is routed entirely in H.

An $O(\log n)$ approximation for v-separable instances of EDP was shown in [4]. Here we provide a constant factor approximation by iteratively applying the incremental flow augmentation procedure of Section 2.1 to the graphs G_1 and G_2. Independent of our work, Nguyen [16] has shown that v-separable instances can in fact be solved exactly if G_1 and G_2 are acyclic.

Let $X = \{(s_i, t_i) \mid 1 \leq i \leq p\}$ denote the set of source-sink pairs with non-zero flow that goes through v and let x_i denote the flow sent by (s_i, t_i) through v. Hence $\sum_i x_i = \alpha$. Let I be the subset of pairs i such that $x_i = 1$. We describe an iterative procedure that modifies the original multicommodity flow through v, f^0, to obtain flows $f^1, f^2, \ldots, f^h$. In f^j, we let x_i^j denote the flow from s_i to t_i. We ensure that x_i^h is integral for $1 \leq i \leq k$ and $\sum_i x_i^h \geq \lfloor \alpha/2 \rfloor$. To accomplish this, in the jth iteration we pick an arbitrary pair ℓ such that $x_\ell^j \in (0,1)$. We stop if there is no such pair. We apply the incremental flow augmentation algorithm from Theorem 2.1 twice. We apply it once to the graph G_1 with v as the sink to augment the flow of s_ℓ up to 1 without decreasing the flow of any s_i with $x_i^j = 1$. We apply it another time to the graph G_2 with v as the source, to augment the flow of t_ℓ up to 1 without decreasing the flow of any t_i with $x_i^j = 1$. These two flow augmentations do not interfere with each other since G_1 and G_2 are disjoint. In the augmenting procedure in G_1 we might reduce the flow from some s_i's to v in G_1. Note, however, that the total such lost flow in G_1 is at most $1 - x_\ell^j < 1$. Similarly we might reduce the flow from v to some t_i's in the procedure in G_2. For any demand i affected in either augmentation, we set $x_i^{j+1} = \min\{b_i, b_i'\} \leq x_i^j$, where b_i, b_i' are the new flow values for s_i, t_i respectively after the augmentations. Thus the total loss of flow to other pairs in phase j is at most $2(1 - x_\ell^j)$. Since we started with α units of flow, it follows that we end up with at least $\lfloor \alpha/2 \rfloor$ pairs with unit flow each. Since G_1 and G_2 are disjoint, the sources of the chosen pairs can route their flow integrally to v and similarly v can route flow to the sinks of the pairs integrally. This yields the desired disjoint paths.

Weighted case: For both undirected and the DAG case, it is straightforward to modify the algorithms to handle weights on the pairs. In each step where we have a choice of choosing an arbitrary pair, we choose the pair with the largest weight. The analysis extends directly and we omit the details.

4 Conclusions

We showed an $O(\sqrt{n})$ upper bound on the integrality gap for EDP and UFP in undirected graphs and DAGs. This matches the known lower bound within a constant factor. It is conjectured in [4] that the integrality gap for directed graphs is also $O(\sqrt{n})$ and this remains an interesting open problem. Proving this in the affirmative would essentially settle the approximability of EDP in directed graphs.

Acknowledgments. We thank the reviewers for several comments that improved the presentation of the paper. Chandra Chekuri and F. Bruce Shepherd acknowledge support from ONR basic research grant N00014-05-1-0256 to Lucent Bell Labs. Sanjeev Khanna acknowledges support from an NSF Career Award CCR-0093117.

References

[1] * MATTHEW ANDREWS, JULIA CHUZHOY, SANJEEV KHANNA, AND LISA ZHANG: Hardness of the undirected edge-disjoint paths problem with congestion. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 226–244, 2005. [FOCS:10.1109/SFCS.2005.41]. 1

[2] * YOSSI AZAR AND ODED REGEV: Combinatorial algorithms for the unsplittable flow problem. *Algorithmica*, 44(1):49–66, 2006. Preliminary version in *Proc. of IPCO*, 2001. [Algorithmica:yl21u55g402h8033, IPCO:eq2xun7jtdm8udue]. 1

[3] * ALOK BAVEJA AND ARAVIND SRINIVASAN: Approximation algorithms for disjoint paths and related routing and packing problems. *Math. Oper. Res.*, 25(2):255–280, 2000. 1

[4] * CHANDRA CHEKURI AND SANJEEV KHANNA: Edge disjoint paths revisited. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 628–637, 2003. [SODA:644108.644212]. 1, 1, 3, 3.2, 4

[5] * CHANDRA CHEKURI, SANJEEV KHANNA, AND F. BRUCE SHEPHERD: The all-or-nothing multicommodity flow problem. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 156–165, 2004. [STOC:1007352.1007383]. 3.1

[6] * CHANDRA CHEKURI, MARCELO MYDLARZ, AND F. BRUCE SHEPHERD: Multicommodity demand flow in a tree. In *Proceedings of 30th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 2719 of *Lecture Notes in Computer Science*, pp. 410–425. Springer, 2003. [ICALP:du648d9x5721adll]. 1

[7] * ANDRÁS FRANK: Packing paths, cuts, and circuits - a survey. In B. Korte, L. Lovász, H. J. Prömel, and A. Schrijver, editors, *Paths, Flows and VLSI-Layout*, pp. 49–100. Springer Verlag, 1990. 1

[8] * NAVEEN GARG, VIJAY V. VAZIRANI, AND MIHALIS YANNAKAKIS: Primal-dual approximation algorithms for integral flow and multicut in trees. *Algorithmica*, 18(1):3–20, 1997. [Algorithmica:x1dykd3030ggac4w]. 1

[9] * VENKATESAN GURUSWAMI, SANJEEV KHANNA, RAJMOHAN RAJARAMAN, F. BRUCE SHEPHERD, AND MIHALIS YANNAKAKIS: Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems. *J. Comput. Syst. Sci.*, 67(3):473–496, 2003. Preliminary version in *Proc. of ACM STOC*, 1999. [JCSS:10.1016/S0022-0000(03)00066-7, STOC:301250.301262]. 1, 1

[10] * JON KLEINBERG: *Approximation algorithms for disjoint paths problems*. PhD thesis, MIT, Cambridge, MA, May 1996. 1

[11] * STAVROS KOLLIOPOULOS: Edge disjoint paths and unsplittable flow. In Teofilo F. Gonzalez, editor, *Handbook on Approximation Algorithms and Metaheuristics*, CRC Computer and Information Science. Chapman and Hall, 2006. To appear. 1

[12] * STAVROS G. KOLLIOPOULOS AND CLIFFORD STEIN: Approximating disjoint-path problems using packing integer programs. *Math. Program.*, 99(1):63–87, 2004. Preliminary version in *Proc. of IPCO, 1998*. [MathProg:lh5yrd3peyp3kb3h, IPCO:upy56mcmrvfqdm84]. 1, 1

[13] * PETR KOLMAN: A note on the greedy algorithm for the unsplittable flow problem. *Inf. Process. Lett.*, 88(3):101–105, 2003. [IPL:10.1016/S0020-0190(03)00351-X]. 1

[14] * PETR KOLMAN AND CHRISTIAN SCHEIDELER: Improved bounds for the unsplittable flow problem. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithm (SODA)*, pp. 184–193, 2002. [SODA:545381.545404]. 1, 1

[15] * BIN MA AND LUSHENG WANG: On the inapproximability of disjoint paths and minimum steiner forest with bandwidth constraints. *J. Comput. Syst. Sci.*, 60(1):1–12, 2000. [JCSS:10.1006/jcss.1999.1661]. 1

[16] * THANH NGUYEN: On disjoint paths. Personal communication, August 2005. 1, 3.2

[17] * ALEXANDER SCHRIJVER: *Combinatorial Optimization: Polyhedra and Efficiency*. Springer-Verlag, Berlin Hiedelberg, 2003. 1

[18] * KASTURI R. VARADARAJAN AND GANESH VENKATARAMAN: Graph decomposition and a greedy algorithm for edge-disjoint paths. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithm (SODA)*, pp. 379–380, 2004. [SODA:982792.982846]. 1, 1

AUTHORS

Chandra Chekuri
Lucent Bell Labs
600 Mountain Avenue
Murray Hill, NJ 07974, USA
chekuri@research.bell-labs.com
http://cm.bell-labs.com/cm/cs/who/chekuri/

Sanjeev Khanna
Dept. of Computer and Information Science
University of Pennsylvania
Philadelphia PA 19104, USA
sanjeev@cis.upenn.edu
http://www.cis.upenn.edu/~sanjeev/

F. Bruce Shepherd
Lucent Bell Labs
600 Mountain Avenue
Murray Hill, NJ 07974, USA
bshep@research.bell-labs.com
http://cm.bell-labs.com/cm/ms/who/bshep/

ABOUT THE AUTHORS

CHANDRA CHEKURI is a Member of Technical Staff (MTS) at Bell Labs of Lucent Technologies, Murray Hill, New Jersey. He joined the labs after finishing his Ph. D. in Computer Science from Stanford University under the supervision of Rajeev Motwani. Before that he obtained his B. Tech. degree in Computer Science and Engineering from the Indian Institute of Technology, Madras (now Chennai). He is primarily interested in algorithms for discrete optimization problems with current research focussing on approximation algorithms. He has worked on various real world problems at Bell Labs and appreciates the difficulty in finding the right balance between theory and practice. He grew up for the most part in the town of Rajahmundry on the banks of the beautiful river Godavari.

SANJEEV KHANNA is a Professor in the Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA. He obtained his Ph. D. from the Computer Science Department of Stanford University under the supervision of Rajeev Motwani. He obtained an undergraduate degree in Computer Science and Economics from Birla Institute of Technology (BITS), Pilani, India. His research interests are in algorithms and complexity with a focus on approximation algorithms and hardness of approximation. He spends his free time playing games with his son Nalin.

BRUCE SHEPHERD is a Member of Technical Staff (MTS) at Bell Labs of Lucent Technologies, Murray Hill, New Jersey. Prior to that he was a Reader at the London School of Economics. He obtained his Ph. D. from the University of Waterloo in the Department of Combinatorics and Optimization under the supervision of Bill Pulleyblank. He obtained an undergraduate degree joint in Mathematics and Computer Science from the University of Victoria. His research interests are in graphs and networks, polyhedral combinatorics, mathematical programming, algorithms and optimization.