# Unsplittable Flow in Paths and Trees and Column-Restricted Packing Integer Programs

Chandra Chekuri*, Alina Ene, and Nitish Korula**

Dept. of Computer Science, University of Illinois, Urbana, IL 61801.
{chekuri, ene1, nkorula2}@illinois.edu

**Abstract.** We consider the unsplittable flow problem (UFP) and the closely related column-restricted packing integer programs (CPIPs). In UFP we are given an edge-capacitated graph $G = (V, E)$ and $k$ request pairs $R_1, \ldots, R_k$, where each $R_i$ consists of a source-destination pair $(s_i, t_i)$, a demand $d_i$ and a weight $w_i$. The goal is to find a maximum weight subset of requests that can be routed unsplittably in $G$. Most previous work on UFP has focused on the *no-bottleneck* case in which the maximum demand of the requests is at most the smallest edge capacity. Inspired by the recent work of Bansal *et al.* [3] on UFP on a path without the above assumption, we consider UFP on paths as well as trees. We give a simple $O(\log n)$ approximation for UFP on trees when all weights are identical; this yields an $O(\log^2 n)$ approximation for the weighted case. These are the first non-trivial approximations for UFP on trees. We develop an LP relaxation for UFP on paths that has an integrality gap of $O(\log^2 n)$; previously there was no relaxation with $o(n)$ gap. We also consider UFP in general graphs and CPIPs without the no-bottleneck assumption and obtain new and useful results.

## 1 Introduction

In the Unsplittable Flow Problem (hereafter, UFP), the input is a graph $G(V, E)$ (directed or undirected; in this paper, we chiefly focus on the latter case) with a capacity $c_e$ on each edge $e \in E$, and a set $\mathcal{R} = \{R_1, R_2, \ldots R_k\}$ of *requests*. Each request $R_i$ consists of a pair of vertices $(s_i, t_i)$, a demand $d_i$, and a weight/profit $w_i$. To route a request $R_i$ is to send $d_i$ units of flow along a *single* path (hence the name *unsplittable flow*) in $G$ from $s_i$ to $t_i$. The goal is to find a maximum-profit set of requests that can be simultaneously routed without violating the capacity constraints; that is, the total flow on an edge $e$ should be at most $c_e$. A special case of UFP when $d_i = 1$ for all $i$ and $c_e = 1$ for all $e$ is the classical maximum edge-disjoint path problem (MEDP). MEDP has been extensively studied, and its approximability in directed graphs is better understood — the best approximation ratio known is is $O(\min\{\sqrt{m}, n^{2/3} \log^{1/3} n)\})$ [20, 33], while it is NP-Hard to approximate to within a factor better than $n^{1/2-\varepsilon}$ [18];

* Partially supported by NSF grants CCF 07-28782 and CNS-0721899.
** Partially supported by NSF grant CCF 07-28782.

here $n$ and $m$ are the number of vertices and edges respectively in the input graph. For undirected graphs there is a large gap between the known upper and lower bounds on the approximation ratio: there is an $O(\sqrt{n})$-approximation [13] while the best known hardness factor is $\Omega(\log^{\frac{1}{2}-\varepsilon} n)$ under the assumption that $NP \not\subseteq ZPTIME(n^{O(\text{polylog}(n))})$ [1]. Thus UFP is difficult in general graphs even without the packing constraints imposed by varying demand values; one could ask if UFP is harder to approximate than MEDP. Most of the work on UFP has been on two special cases. One is the uniform capacity UFP (UCUFP) in which $c_e = C$ for all $e$ and the other is UFP with the no-bottleneck assumption (UFP-NBA) where one assumes that $\max_i d_i \leq \min_e c_e$. Note that UCUFP is a special case of UFP-NBA. Kolliopoulos and Stein [22] showed, via grouping and scaling techniques, that certain linear programming based approximation algorithms for MEDP can be extended with only an extra constant factor loss to UFP-NBA. This reduction holds even when one considers restricted families of instances, say, those induced by planar graphs. See [15] for a precise definition of when the reduction applies. In [7, 30], a different randomized rounding approach was used for UFP-NBA.

In this paper we are primarily interested in UFP instances that *do not* necessarily satisfy the no-bottleneck assumption. UCUFP and UFP-NBA have many applications and are of interest in themselves. However, the general UFP, due to algorithmic difficulties, has received less attention. One can extend some results for MEDP and UFP-NBA to UFP by separately considering requests that are within say a factor of 2 of each other; this geometric grouping incurs an additional factor of $\log d_{\max}/d_{\min}$ in the approximation ratio, which could be as large as a factor of $n$ [18]. Azar and Regev showed that UFP in directed graphs is $\Omega(n^{1-\varepsilon})$-hard unless $P = NP$; note that the hardness for UFP-NBA is $\Omega(n^{1/2-\varepsilon})$ [18]. Chakrabarti *et al.* [12] observed that the natural LP relaxation has $\Omega(n)$ integrality gap even when $G$ is a path. In contrast, the integrality gap for the path is $O(1)$ for UFP-NBA [12, 15]. One could argue that the integrality gap of the natural LP has been the main bottleneck in addressing UFP.

This paper is inspired by the recent work of Bansal *et al.* [3] who gave an $O(\log n)$ approximation for UFP on a path. Interestingly, this was the first non-trivial approximation for this problem; previously there was a quasi-polynomial time approximation scheme [4], provided the capacities and demands are quasi-polynomially bounded in $n$. We note that UFP even on a single edge is NP-Hard, since it is equivalent to the knapsack problem. UFP on a path has received considerable attention, not only as an interesting special case of UFP, but also as a problem that has direct applications to resource allocation where one can view the path as modeling the availability of a resource over time. See [5, 6, 9, 12, 4, 3] for previous work related to UFP on a path. The algorithm in [3] is combinatorial and bypasses the $\Omega(n)$ lower bound on the integrality gap of the natural LP. An open problem raised in [3] is whether UFP on trees also has a poly-logarithmic approximation. The difficulty of UFP on paths and trees is not because of routing (there is a unique path between any two nodes) but entirely due to the difficulty of choosing the subset to route. We note that this

subset selection problem is easy on a path if $d_i = 1$ for all $i$ (the natural LP is integral since the incidence matrix is totally unimodular) while this special case is already NP-Hard (and APX-Hard to approximate) on capacitated trees [17]. A constant factor approximation is known for UFP-NBA on trees [15]. We prove the following theorem, answering positively the question raised in [3].

**Theorem 1.** *There is an $O(\log n)$ approximation for UFP on $n$-vertex trees when all weights are equal. There is an $O(\log n \cdot \min\{\log n, \log k\})$ approximation for arbitrary non-negative weights.*

We borrow a crucial high-level idea from [3] of decomposing the given instance into one in which the demands all intersect. We, however, deviate from their approach of using dynamic programming for "large" demands which does not (seem to) generalize from paths to trees; our algorithm for trees is significantly simpler than the complex dynamic programming for the path used by [3]. We show that for the unit-weight case, a greedy algorithm is a 2-approximation if all requests go through a common vertex in the tree. This insight into the performance of the greedy algorithm allows us to develop a new linear programming relaxation for paths.

**Theorem 2.** *There is a linear programming relaxation for UFP on the path that has an integrality gap of $O(\log n \cdot \min\{\log n, \log k\})$ and there is a polynomial time algorithm that obtains a feasible $O(1)$-approximate solution to the relaxation.*

The separation oracle for the exponential-sized relaxation we develop is non-trivial. The integrality gap of the relaxation may very well be $O(1)$; resolving this is an interesting open problem. We underscore the novelty of our relaxation by showing that some reasonable approaches to strengthening the natural relaxation fail to improve the gap. In particular we show that the relaxation obtained after applying $t$ rounds of the Sherali-Adams lift-and-project scheme [29] to the natural relaxation has a gap of $\Omega(n/t)$.

**Column-Restricted Packing Integer Programs:** UFP on paths and trees are special cases of column-restricted packing integer programming problems (CPIP). A packing integer program (PIP) is an optimization problem of the form $\max\{wx \mid Ax \leq b, x \in \{0,1\}^n\}$ where $A$ is a non-negative matrix; we use $(A, w, b)$ to define a PIP. A CPIP has the additional restriction that all the non-zero entries in each column of $A$ are identical. It is easy to write UFP on a tree as a CPIP (see Section 5 for formal details). The common coefficient of each column is the "demand" of that column. UFP on general graphs can also be related to CPIPs by using the path formulation and additional constraints [22]. A 0-1 PIP is one in which all entries of $A$ are in $\{0,1\}$; note that it is also a CPIP. 0-1 PIPs capture the maximum independent set problem (MIS) as a special case and the strong inapproximability results for MIS [19] imply that no $n^{1-\varepsilon}$-approximation is possible for 0-1 PIPs unless $P = NP$; here $n$ is the number of columns of $A$. However, an interesting question is the following. Suppose a 0-1 PIP has a small integrality gap because $A$ has some structural

properties. For example, if $A$ is totally unimodular, then the integrality gap is 1. What can be said about a CPIP that is derived from $A$? In other words, one is asking how the "demand version" of a CPIP is related to its "unit-demand" version (see [22, 28, 15]). A CPIP satisfies the no-bottleneck assumption (NBA) if $\max_{i,j} A_{ij} \leq \min_i b_i$. Kolliopoulos and Stein [22] showed that for CPIPs that satisfy the NBA, one can relate the integrality gap of a CPIP to the gap of its underlying 0-1 PIP; there is only an extra constant factor. These ideas are what allows one to relate UFP-NBA to MEDP.

As with UFP, we are interested in this paper in CPIPs where we do not make the NBA assumption. As above, one could ask whether the integrality gap for the demand version of CPIP can be related to its unit-demand version. (Here, we refer to the "natural" relaxation in which one simply relaxes the integrality constraints.) However, the gap example for UFP on the path shows that unlike the no-bottleneck case, such a relationship is not possible. The unit-demand version of UFP on the path has integrality gap 1 while the demand-version has a gap of $\Omega(n)$. It is therefore natural to look for an intermediate case. In particular, suppose we have a CPIP $(A, w, b)$ such that $\max_j A_{ij} \leq (1-\delta)b_i$ for each $i$; this corresponds to the assumption that each demand is at most $(1-\delta)$ times the *bottleneck* capacity for that demand. We call such a CPIP a $\delta$-bounded CPIP. We informally state below a result that we obtain; the formal statement can be found in Section 5.

*The integrality gap of a $\delta$-bounded CPIP is at most $O(\log(1/\delta)/\delta^3)$ times the integrality gap of its unit-demand version.*

The proof of the above is not difficult and is based on the grouping and scaling ideas of [22] with an additional trick. However, this has not been observed or stated before and the corollary below was not known previously.

**Corollary 1.** *For each fixed $\delta > 0$, there is an $O(\log(1/\delta)/\delta^3)$ approximation for UFP on paths and trees if the demand of each request is at most $(1-\delta)$ times the capacity of the edges on the unique path of the request.*

One class of CPIPs that have been studied before are those in which the maximum number of non-zero entries in any column is at most $L$. Baveja and Srinivasan [7] showed that the integrality gap of such CPIPs is $O(L)$ if $A$ satisfies the no-bottleneck assumption. In recent and independent work, Pritchard [25] considered PIPs that have at most $L$ non-zero entries per column, calling them $L$-column-sparse PIPs, and gave an $O(2^L L^2)$ approximation for them. We follow his notation, but obtain a tighter bound by restricting our attention to $L$-sparse CPIPs.

**Theorem 3.** *There is an $O(L)$-approximation for $L$-sparse CPIPs via the natural LP relaxation, even without the no-bottleneck assumption. If $w$ is the all 1's vector then a simple greedy algorithm gives an $L$-approximation to the integral optimum (not necessarily with respect to the LP optimum).*

As corollaries we obtain the following results. We refer to UFP in which the paths for the routed requests have to contain at most $L$ edges as $L$-bounded-UFP.

**Corollary 2.** *There is an $O(L)$-approximation for $L$-bounded-UFP in* directed *graphs.*

The demand-matching problem considered by Shepherd and Vetta [28] is an instance of a 2-bounded CPIP and therefore we have.

**Corollary 3.** *There is an $O(1)$-approximation for the demand-matching problem. Moreover, there is a 2-approximation for the cardinality version.*

Note that [28] gives a 3.264 approximation for general graphs and a 3-approximation for the cardinality version, both with respect to the LP optimum. Our $O(L)$ bound for $L$-bounded CPIPs has a larger constant factor since it does not take the structure of the particular problem into account, however the algorithm is quite simple. On the other hand, the greedy 2-approximation for the cardinality case was not noticed in [28].

Due to space constraints, we omit most of the proofs. A full version of the paper will be available on the authors' websites.

**Other Related Work and Discussion:** UFP and MEDP are extensively studied and we refer the reader to [1, 13, 14, 16, 20, 21] for various pointers on approximation algorithms and hardness results. Schrijver [27] discusses known results on exact algorithms in great detail. We focus on UFP on paths and trees and have already pointed to the relevant literature. We mention some results on UFP for the special case when $w_i = d_i$. Kolman and Schiedeler [24] considered this special case in directed graphs and obtained an $O(\sqrt{m})$-approximation. Kolman [23] extended the results in [33] for UCUFP to this special case. We note that the $\Omega(n)$ integrality gap for the path [12] does not hold if $w_i = d_i$. In a technical sense, one can reduce a UFP instance with $w_i = d_i$ to an instance in which the ratio $d_{\max}/d_{\min}$ is polynomially bounded. Two approximation techniques for UFP-NBA are greedy algorithms [20, 22, 2] and randomized rounding of the multi-commodity flow based LP relaxation [30, 7, 9, 12]. These methods when dealing with UFP-NBA classify demands as "large" ($d_i \geq d_{\max}/2$) and "small". Large demands can be reduced to uniform demands and handled by MEDP algorithms (since $d_{\max} \leq \min_e c_e$) and small demands behave well for randomized rounding. This classification does not apply for UFP. A simple observation we make is that if we are interested in the cardinality problem then it is natural to consider the greedy algorithm that gives preference to smaller demands; under various conditions this gives a provably good algorithm. Another insight is that the randomized rounding algorithm followed by alteration [31, 12] has good behaviour if we sort the demands in decreasing order of their size — this observation was made in [12] but its implication for general UFP was not noticed. Finally, the modification of the grouping and scaling ideas to handle $\delta$-bounded demands is again simple but has not been noticed before. Moreover, for UFP on paths and trees one obtains constant factor algorithms for any fixed $\delta$. We remark that this result is not possible to derive from the randomized rounding and alteration approach for paths (or trees) because the alteration approach needs to insert requests based on left end point to take advantage of the path

structure while one needs the requests to be sorted in decreasing demand value order to handle the fact that we cannot separate small and large demands any more.

Strengthening LP relaxations by adding valid inequalities is a standard methodology in mathematical programming. There are various generic as well as problem specific approaches known. The knapsack problem plays an important role since each linear constraint in a relaxation can be thought of inducing a separate knapsack constraint. Knapsack cover inequalities [10] have been found to be very useful in reducing the integrality gap of covering problems [10, 25]. However, it is only recently that Bienstock [8], answering a question of Van Vyve and Wolsey [32], developed an explicit system of inequalities for the knapsack packing problem (the standard maximization problem) that yields an approximation scheme. Wolsey (as reported in [15]) raises the question of how multiple knapsack constraints implied by the different linear constraints of a relaxation interact since that is what ultimately determines the strength of the relaxation. UFP on a path is perhaps a good test case for examining this question. The $\Omega(n)$ gap example shows the need to consider multiple constraints simultaneously — we hope that our formulation and its analysis is a step forward in tackling other problems.

## 2 UFP on Trees

Recall that each request $R_i$ consists of a pair of vertices $s_i, t_i$, a demand $d_i$ and a profit/weight $w_i$, and if selected, the entire $d_i$ units of demand for this request must be sent along a single path. When the input graph is a tree, there is a *unique* path between each $s_i$ and $t_i$. For such instances, we refer to this unique path $P_i$ as being the request path for $R_i$.

The following flow-based LP relaxation is natural for UFP on trees: Here, $x_i$ indicates whether flow is routed from $s_i$ to $t_i$.

$$\begin{array}{rcll} \max & \sum_{i=1}^{k} w_i x_i & \text{s.t.} & \\ \sum_{i:\ P_i \ni e} d_i x_i & \leq & c_e & (\forall e \in E(G)) \\ x_i & \in & [0,1] & (\forall i \in \{1,\ldots,k\}) \end{array}$$

This relaxation has an $O(1)$ integrality gap for UFP-NBA on trees [15]. Unfortunately, without NBA, the gap can be as large as $\Omega(n)$ even when the input graph is a path, as shown in [12] (see Section 3). No relaxations with gap $o(n)$ were previously known, even for UFP on paths. The difficulty appeared to lie in dealing with requests for which the demands are very close to the capacity constraints; we confirm this intuition by proving Corollary 1 in Section 5: For UFP on trees, if each $d_i \leq (1-\delta)\min_{e\in P_i} c_e$, the natural LP relaxation has an integrality gap of $O(\text{poly}(1/\delta))$. In Section 4, we show how to handle large demands for UFP on paths by giving a new relaxation with an integrality gap of $O(\log n \cdot \min\{\log n, \log k\})$.

In this section, we give a simple combinatorial algorithm that achieves an $O(\log n \cdot \min\{\log n, \log k\})$-approximation for UFP on trees. We first obtain an

$O(\log n)$ approximation for unit-profit instances of UFP on trees with $n$ vertices. To do this, we note that if all the request paths must pass through a common vertex, a simple greedy algorithm achieves a 2-approximation.

**Lemma 1.** *Consider unit profit instances of UFP on trees, for which there exists a vertex $v$ such that all request paths pass through $v$. There exists a 2-approximation algorithm for such instances.*

**Proof Sketch:** We order the requests in increasing order according to their demands. We consider the requests in this order and, if adding the current request maintains feasibility, we add the request to our set. □

**Lemma 2.** *There exists an $O(\log n)$-approximation algorithm for unit profit instances of UFP on trees.*

**Proof Sketch:** It is well known that any $n$-vertex tree $T$ has a vertex $v$, called a *center*, such that each component of $T \backslash v$ has at most $n/2$ vertices. If many request paths pass through the center $v$, we use Lemma 1 and are done; if not, most paths are entirely contained in the subtrees (each of size at most $n/2$) obtained after deleting $v$ from $T$, and we can recurse. □

Theorem 1 now follows from Lemma 2 and Lemma 3 below, which is proved using standard profit-scaling.

**Lemma 3.** *Suppose there exists an $r$-approximation algorithm for unit profit instances of UFP on a given graph. Then there exists an $O(r \min\{\log n, \log k\})$-approximation algorithm for arbitrary instances of UFP on the graph, where $k$ is the number of requests.*

## 3 LP Relaxations for UFP on Paths

The following Linear Programming relaxation is natural for UFP on paths. There is a variable $x_i$ for each request $R_i$ to indicate whether it is selected, and the constraints enforce that the total demand of selected requests on each edge is at most its capacity.

$$\textbf{Standard LP} \quad \max \textstyle\sum_i w_i x_i$$

$$\begin{aligned} \textstyle\sum_{i:\, e \in P_i} d_i x_i &\le c_e && (\forall e \in E(G)) \\ x_i &\in [0,1] && (\forall i \in \{1, \ldots, k\}) \end{aligned}$$

It is shown in [12] that the integrality gap of this LP relaxation is $\Theta(\log \frac{d_{\max}}{d_{\min}})$ where $d_{\max}$ and $d_{\min}$ are $\max_i d_i$ and $\min_i d_i$ respectively. Unfortunately, this gap can be as bad as $\Omega(n)$, as shown in the following example from [12]: the input path has $n$ edges with edge $i$ having capacity $2^i$; request $R_i$ is for $2^i$ units of capacity on edges $i$ through $n$, and has profit 1. (See Fig. 1.) An integral solution can only route a single request, for a profit of 1; however, setting $x_i = 1/2$ for each $i$ is a feasible fractional solution to the LP, for a total profit of $n/2$. We refer to this instance as the *canonical integrality gap example.*

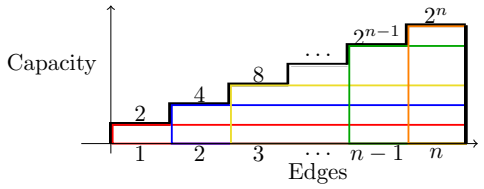


Fig. 1: An instance of UFP on paths with large integrality gap.

Though an $O(\log n)$-approximation algorithm for UFP on paths was given in [3], no LP with an integrality gap of $o(n)$ was known for this problem, and obtaining such an LP has been an interesting open question. One could attempt to write a configuration LP for the problem, or to consider strengthening the natural LP, for instance, via the Sherali-Adams hierarchy of relaxations. We remark that these relaxations also have feasible fractional solutions of profit $\Omega(n)$ for the canonical integrality gap example. For both of the relaxations below, we use $\mathcal{R}_e$ to denote the set of requests passing through edge $e$.

*A Configuration LP:* In the configuration LP below, there is a variable $x_{S,e}$ for each set $S \subseteq \mathcal{R}_e$ if the total demand $d_S$ of the requests in $S$ is at most the capacity $c_e$. Though this LP has an exponential number of variables, we can separate over its dual, which has a polynomial number of variables and constraints that are essentially equivalent to the knapsack problem (with polynomially bounded profits, since we assume that the profits of the original instance are integers in $\{1, \ldots, k^2\}$). However, the integrality gap of the configuration LP is also $n/2$, as shown by the canonical example; set $x_i = 1/2$ for each $i$, and for the $j$th edge $e_j$, set $x_{\{R_j\},e_j} = 1/2$, and $x_{S_j,e_j} = 1/2$, where $S_j = \{1, \ldots, j-1\}$. (On edge $e_1$, set $x_{\emptyset,e_1} = 1/2$.)

**Config LP** $\quad \max \sum_i w_i x_i$

$$\begin{aligned} \textstyle\sum_{S:\, S \subseteq \mathcal{R}_e} x_{S,e} &= 1 && (\forall e \in E(G)) \\ x_i &\leq \textstyle\sum_{S:\, S \subseteq \mathcal{R}_e} x_{S,e} && (\forall i \in \{1, \ldots, k\}, e \in P_i) \\ x_{S,e} &\geq 0 && (\forall e \in E(G), S \subseteq \mathcal{R}_e, d_S \leq c_e) \end{aligned}$$

*The Sherali-Adams hierarchy for the* ***Standard LP****:* For a zero-one programming problem, let $P$ denote the feasible integer polytope, and $P_0$ denote the convex polytope of an LP relaxation for $P$. The Sherali-Adams Hierarchy [29] is a sequence $P_0, P_1, P_2 \ldots P_n = P$ of (successively tighter) relaxations of $P$. We refer the reader to [29] for a more complete description of the Sherali-Adams Hierarchy; here, we simply note that the integrality gap of $P_t$ is $\Omega(n/t)$.

**Theorem 4.** *After applying $t$ rounds of the Sherali-Adams hierarchy to the relaxation* ***Standard LP****, the integrality gap of the LP obtained is $\Omega(n/t)$.*

The two preceding examples show that it is difficult to write an LP relaxation with small integrality gap by only considering "local" constraints, which bound the capacity used on each edge in isolation. A stronger LP needs to introduce constraints that are more global in nature, taking into account that different edges may prevent different subsets of requests from being routed.

## 4 A New Relaxation

We now describe a Linear Programming Relaxation for the UFP on paths with an $O(\log^2 n)$ integrality gap. Corollary 1 implies that **Standard LP** has small integrality gap if the demand of each request is small compared to the capacity constraints; recall that in the canonical example with integrality gap $n/2$, every request, if routed, uses the *entire* capacity of the leftmost edge on its path. In the new LP relaxation, we keep the previous constraints to handle "small" requests, and introduce new rank constraints to deal with "big" requests.

For each request $R_i$, let the *bottleneck* for $R_i$ be the edge in $P_i$ with least capacity. (If multiple edges have the same minimum capacity, let the bottleneck be the leftmost edge.) Let $\mathcal{S} \subseteq \mathcal{R}$ be the set of all requests $R$ such that the demand of $R$ is smaller than $(3/4) \cdot c(e)$ where $e$ is the bottleneck edge for $R$. Let $\mathcal{B} = \mathcal{R} \backslash \mathcal{S}$ denote the remaining ("big") requests, and let $\mathcal{B}_e$ denote the set of requests $R$ in $\mathcal{B}$ such that the path for $R$ passes through edge $e$. For each request $R_i$, we have a variable $x_i$ denoting whether this request is selected or not. For each set $B \subseteq \mathcal{B}$ of big requests, let $f(B)$ denote the maximum *number* of requests in $B$ that can be simultaneously routed without violating the capacity constraints. For each set $B$ of "big" requests that pass through a common edge, we introduce a *rank* constraint which requires that the total extent to which requests in $B$ are selected by the LP must be at most the number of requests in $B$ that can be routed integrally.

**UFP-LP** $\quad \max \sum_i w_i x_i$

$$\begin{array}{rll} \sum_{i:\, e \in P_i} d_i x_i \leq c_e & (\forall e \in E(G)) & \text{[capacity constraints]} \\ \sum_{R_i \in B} x_i \leq f(B) & (\forall e \in E(G), B \subseteq \mathcal{B}_e) & \text{[rank constraints]} \\ x_i \in [0,1] & (\forall i \in \{1, \ldots, k\}) & \end{array}$$

The new constraints enforce a small integrality gap; we prove Theorem 5 in Section 4.2. The upper bound on the integrality gap is not known to be tight; the integrality gap could be $O(\log n)$ or even $O(1)$.

**Theorem 5.** *The LP relaxation **UFP-LP** has integrality gap $O(\log n \cdot \min\{\log n, \log k\})$ for instances of UFP on paths, where $n$ is the length of the path and $k$ is the number of requests.*

An interesting question is obtaining a separation oracle for **UFP-LP**, which has an an exponential number of constraints. We describe an algorithm Separation Oracle and prove the following theorem in Section 4.1 below; together, Theorems 5 and 6 imply Theorem 2.

**Theorem 6.** *Let $x \in [0,1]^n$ and suppose there exists a set $B \subseteq \mathcal{B}_e$ such that $\sum_{R_i \in B} x_i > 18f(B)$. Then the algorithm* Separation Oracle($e$) *returns a violated constraint.*

An approximate separation oracle such as the one guaranteed by Theorem 6 can be used to find an approximate solution to **UFP-LP**; this follows for a large class of packing and covering problems (see [11]); we omit details.

One can write a relaxation similar to **UFP-LP** for UFP on trees; though it has small integrality gap, we do not know a separation oracle as in Theorem 6.

### 4.1 A Separation Oracle

We now describe an approximate separation oracle for **UFP-LP**. We can obviously check in polynomial time whether there exists a capacity constraint that is violated (and return such a constraint if one exists). Therefore we may assume that all the capacity constraints are satisfied and hence we can safely ignore the requests in $\mathcal{S}$. We give an algorithm to detect a violated rank constraint at edge $e$ if some rank constraint at $e$ is violated by a factor of at least 18. We first introduce some notation:

We define $x(S) = \sum_{R_i \in S} x_i$. Let $\mathcal{B}_{\text{left}}(e) \subseteq \mathcal{B}_e$ be the set of requests $R_i$ such that the bottleneck for $R_i$ is to the *left* of edge $e$, and $\mathcal{B}_{\text{right}}(e)$ be the set of requests $R_i$ with bottleneck to the right of $e$. (If the bottleneck for $R_i \in \mathcal{B}_e$ is edge $e$, $R_i$ can be added to either $\mathcal{B}_{\text{left}}(e)$ or $\mathcal{B}_{\text{right}}(e)$.)

Let *left*($e$) denote the set of edges to the left of $e$, together with edge $e$, and let *right*($e$) be the set of edges to the right of $e$ (again including $e$). For requests $R_i, R_j$ both in $\mathcal{B}_{\text{left}}(e)$ (respectively, both in $\mathcal{B}_{\text{right}}(e)$) we say that $R_i$ blocks $R_j$ if there is an edge $e' \in$ *left*($e$) (respectively, *right*($e$)) such that $d_i + d_j > c_{e'}$ and both $P_i$ and $P_j$ pass through $e'$.

```
Separation Oracle(edge e):
  for each request R_i ∈ B_left(e)
      let S = {R_i} ∪ {R_j | R_j ∈ B_left(e), d_j > d_i, R_i blocks R_j}
      if x(S) > 1
          return S                          ⟨⟨f(S) = 1 by construction⟩⟩

  for each request R_i ∈ B_right(e)
      let S = {R_i} ∪ {R_j | R_j ∈ B_right(e), d_j > d_i, R_i blocks R_j}
      if x(S) > 1
          return S                          ⟨⟨f(S) = 1 by construction⟩⟩
```

**Lemma 4.** *For any set $S$ returned by* Separation Oracle, *$f(S) = 1$.*

Thus, if this algorithm returns a set $S \subseteq \mathcal{B}_e$, the constraint corresponding to $S$ and $e$ is violated, as $x(S) > 1$. To prove Theorem 6, it remains only to show that if constraints are sufficiently violated, the algorithm will always return some set $S$ corresponding to a violated constraint. The proof is somewhat involved, though the outline is simple: Given a set $B \subseteq \mathcal{B}_e$ such that $x(B) > 18f(B)$, we

show the existence of a set $S \subseteq B$ with "simpler" structure, such that $f(S) = 1$ and $x(S) > 1$; that is, $S$ is a violated set. The structure of $S$ is such that the algorithm SEPARATION ORACLE($e$) can find it.

Given an edge $e$ and a set $S' \subseteq \mathcal{B}_e$, we say that $S'$ is *feasible on the left* (respectively, on the right), if all requests in $S'$ can be routed simultaneously without exceeding the capacity of any edge in *left*($e$) (respectively, *right*($e$)). For any set $S \subseteq \mathcal{B}_e$, let $f_\ell(S)$ denote the maximum size subset of $S$ that is feasible on the left and $f_r(S)$ denote the maximum size subset of $S$ that is feasible on the right. (Equivalently, $f_\ell(S)$ is $f(S)$ in the instance obtained by truncating all requests at the right endpoint of $e$.)

**Lemma 5.** *If there exists a set $B \subseteq \mathcal{B}_e$ such that $x(B) > \alpha f(B)$, there exists a set $B' \subseteq \mathcal{B}_{left}(e)$ such that $x(B') > \frac{\alpha}{2} f_\ell(B')$ or a set $B'' \subseteq \mathcal{B}_{right}(e)$ such that $x(B'') > \frac{\alpha}{2} f_r(B'')$.*

By symmetry, we assume w.l.o.g. that there exists $B' \subseteq \mathcal{B}_{\text{left}}(e)$ such that $x(B') > (\alpha/2) f_\ell(B')$. For brevity, we complete the proof of Theorem 6 by only stating the remaining lemmas for the "left" side.

**Lemma 6.** *If there exists a set $S' \subseteq \mathcal{B}_{left}(e)$ such that $f_\ell(S') = 1$ and $x(S') > 1$, the algorithm* SEPARATION ORACLE *returns such a set.*

The previous two lemmas show that: (a) If there is a constraint violated by a factor $\alpha$, there is one violated by a factor of $\alpha/2$ either "on the left" or "on the right", and (b) If there is a constraint corresponding to set $S'$ violated on the left (or on the right) such that $f_\ell(S')$ (or $f_r(S')$) $= 1$, the algorithm detects it. To complete the proof of Theorem 6, our final lemma shows that if there is a constraint violated on the left by a large factor, there is a violated constraint corresponding to a set $S'$ such that $f_\ell(S') = 1$.

**Lemma 7.** *If there exists a set $B \subseteq \mathcal{B}_{left}(e)$ such that $x(B) > \beta f_\ell(B)$ for some $\beta > 9$, there exists a set $S' \subseteq B$ such that $f_\ell(S') = 1$ and $x(S') > 1$.*

### 4.2 Bounding the Integrality Gap

Given an fractional solution to the LP of profit OPT, we show how to round it to obtain an integral solution of comparable profit. For any set $S$ of requests, we define profit($S$) as $\sum_{R_i \in S} w_i x_i$. We round "small" and "big" jobs separately; note that one of profit($\mathcal{S}$) or profit($\mathcal{B}$) is at least OPT/2. If profit($\mathcal{S}$) $\geq$ OPT/2 then one obtains from Corollary 1 that there is an integral solution of value $\Omega(\text{OPT})$; recall that for each request $R_i \in \mathcal{S}$, we have $d_i \leq (3/4) \min_{e \in P_i} c_e$.

The difficulty in bounding the integrality gap of LPs has been in dealing with the "big" requests. However, the new rank constraints allow one to overcome this. The proof essentially follows the combinatorial algorithm from Section 2. We apply the same arguments as in Section 2, now with respect to the LP solution instead of an integral optimum solution, to reduce the problem to an intersecting instance with unit weights; this loses an $O(\log n \min\{\log n, \log k\})$ factor. For an

intersecting instance with unit weights, the rank constraints trivially show that the LP optimum is equal to the integral optimum. We omit further details. This completes the proof of Theorem 5.

## 5 UFP and Column-Restricted Packing Integer Programs

In this section, we consider a class of packing problems, so-called *Column-Restricted* Packing Integer Programs (hereafter, CPIPs), introduced by Kolliopoulos and Stein [22]. Let $A$ be an arbitrary $m \times n$ $\{0,1\}$ matrix, and $d$ be an $n$-element vector with $d_j$ denoting the $j$th entry in $d$. Let $A[d]$ denote the matrix obtained by multiplying every entry of column $j$ in $A$ by $d_j$. A CPIP is a problem of the form $\max wx$, subject to $A[d]x \leq b, x \in \{0,1\}^n$, for some integer vectors $w, d, b$.[1] (Intuitively, a CPIP is a 0-1 packing program in which all non-zero coefficients of a variable $x_j$ are the same.) It is easy to see that the natural LP for UFP in paths and trees is a CPIP.

CPIPs were studied in [22, 15], and it was shown that the integrality gap of a CPIP with $\max_j d_j \leq \min_i b_i$ is at most a constant factor more than the integrality gap of the corresponding "unit-demand" version; we explain this more formally below, using the notation introduced by [15].

Let $P$ be a convex body in $[0,1]^n$ and $w \in \mathbf{R}^n$ be an objective vector; for any choice of $P, w$, we obtain a maximization problem $\max\{wx : x \in P\}$. Let $\gamma$ denote the fractional optimum value of this program, and $\gamma^*$ denote the optimum *integral* value, which is given by $\max wx$ over all integer vectors $x \in P$. The *integrality gap* of $P$ is $\gamma/\gamma^*$, the ratio between the value of the optimal fractional and integral solutions. A class $\mathcal{P}$ of integer programs is given by problems induced by pairs $P, w$ as above; the integrality gap for a class of problems $\mathcal{P}$ is the supremum of integrality gaps for each problem in $\mathcal{P}$.

We say that a collection of vectors $W \subseteq \mathbf{Z}^n$ is *closed* if for each $w \in W$, replacing any entry $w_i$ with 0 gives a vector $w' \in W$. Subsequently, for each $m \times n$ matrix $A$ and closed collection of vectors $W$ in $Z^n$, we use $\mathcal{P}(A, W)$ to denote the class of problems of the form $\max\{wx : Ax \leq b, x \in [0,1]^n\}$, where $w \in W$ and $b$ is a vector in $\mathbf{Z}_+^m$. We let $\mathcal{P}^{dem}(A, W)$ denote the class of problems of the form $\max\{wx : A[d]x \leq b, x \in [0,1]^n\}$ where $w \in W, b \in \mathbf{Z}_+^m, d \in \mathbf{Z}_+^n$. Finally, we use $\mathcal{P}_{nba}^{dem}(A, W)$ to denote the class of problems of the same form that satisfy $\max_j d_j \leq \min_i b_i$. For UFP, the condition $\max_j d_j \leq \min_i b_i$ corresponds to the no-nottleneck assumption.

Using techniques introduced in [22], the following theorem was proved in [15]:

**Theorem 7 ([15]).** *Let $A$ be a $\{0,1\}$ matrix and $W$ be a closed collection of vectors. If the integrality gap for the collection of problems $\mathcal{P}(A, W)$ is at most $\Gamma$, then the integrality gap for the collection of problems $\mathcal{P}_{nba}^{dem}(A, W)$ is at most $11.542\Gamma \leq 12\Gamma$.*

The above theorem is used in [15] to give an $O(1)$-approximation for UFP-NBA on trees. Unfortunately, the analogous theorem is not true for $\mathcal{P}^{dem}(A, W)$,

[1] If vectors $w, d, b$ are rational, we can scale them as necessary.

as shown by the canonical integrality gap example for the UFP linear program **Standard LP**. In this section, we note that if there exists $\delta < 1$ such that for each $i$, we have $\max_j A_{ij} d_j \leq (1-\delta) b_i$, we *can* obtain an analogous theorem, with integrality gap depending on $\delta$. More precisely, let $\mathcal{P}_\delta^{dem}(A, W)$ denote the class of problems of the form $\max\{wx : A[d]x \leq b, x \in [0,1]^n\}$ where $w \in W, b \in \mathbf{Z}_+^m, d \in \mathbf{Z}_+^n$, and $\forall i$, $\max_j A_{ij} d_j \leq (1-\delta) b_i$.

**Theorem 8.** *Let $A$ be a $\{0,1\}$ matrix and $W$ be a closed collection of vectors. If the integrality gap for $\mathcal{P}(A, W)$ is at most $\Gamma$, the integrality gap for $\mathcal{P}_\delta^{dem}(A, W)$ is at most $O(\frac{\log(1/\delta)}{\delta^3} \cdot \Gamma)$.*

Thus, we obtain Corollary 1 as a special case of Theorem 8.

## Concluding Remarks

Is there an $O(1)$-approximation for UFP on paths, and more generally on trees? Is the integrality gap of **UFP-LP** $O(1)$? Is there an LP relaxation for UFP on trees with poly-logarithmic integrality gap?

We recently obtained an $O(L^2)$-approximation ratio and integrality gap bound for $L$-sparse PIPs using the iterated rounding idea of Pritchard [25]; this improves his bound of $O(2^L L^2)$. Can the bound be improved to $O(L)$, matching the lower bound on the integrality gap?

## References


1. M. Andrews, J. Chuzhoy, S. Khanna, and L. Zhang. Hardness of the undirected edge-disjoint paths problem with congestion. *Proc. of IEEE FOCS*, 226–241, 2005.
2. Y. Azar, O. Regev. Combinatorial algorithms for the unsplittable flow problem. *Algorithmica*, 441(1):49–66, 2006.
3. N. Bansal, Z. Friggstad, R. Khandekar, M. R. Salavatipour. A logarithmic approximation for unsplittable flow on line graphs In *Proc. of ACM-SIAM SODA*, 702–709, 2009.
4. N. Bansal, A. Chakrabarti, A. Epstein, B. Schieber. A Quasi-PTAS for unsplittable flow on line graphs In *Proc. of ACM STOC*, 721–729, 2006.
5. A. Bar-Noy, R. Bar-Yehuda, A. Freund, J. Naor, B. Schieber. A unified approach to approximating resource allocation and scheduling. *JACM* 48(5): 1069–1090, 2001.
6. A. Bar-Noy, S. Guha, J. Naor, B. Schieber. Approximating the Throughput of Multiple Machines in Real-Time Scheduling. *SICOMP.* 31(2):331–352, 2001.
7. A. Baveja, A. Srinivasan. Approximation algorithms for disjoint paths and related routing and packing problems. *Math. Oper. Res.*, 25(2):255–280, 2000.
8. D. Bienstock. Approximate formulations for 0-1 knapsack sets. *Oper. Res. Lett.* 36(3): 317–320, 2008.
9. G. Calinescu, A. Chakrabarti, H. Karloff, Y. Rabani. Improved approximation algorithms for resource allocation. In *Proc. of IPCO*, 439–456, 2001.
10. R.D. Carr, L. Fleischer, V.J. Leung, C.A. Phillips. Strengthening integrality gaps for capacitated network design and covering problems. *Proc. of ACM-SIAM SODA*, 106–115, 2000.

11. R. Carr and S. Vempala. Randomized meta-rounding. *Random Structures and Algorithms*, 20(3):343–352, 2002.
12. A. Chakrabarti, C. Chekuri, A. Gupta, A. Kumar. Approximation Algorithms for the Unsplittable Flow Problem *Algorithmica* 47(1):53–78, 2007.
13. C. Chekuri, S. Khanna, F. B. Shepherd. An $O(\sqrt{n})$ Approximation and Integrality Gap for Disjoint Paths and Unsplittable Flow. *Theory of Computing*, Vol 2, 137–146, 2006.
14. C. Chekuri, S. Khanna, F. B. Shepherd. Edge-Disjoint Paths in Planar Graphs with Constant Congestion. *Proc. of ACM STOC*, 757–766, 2006.
15. C. Chekuri, M. Mydlarz, F. B. Shepherd. Multicommodity Demand Flow in a Tree and Packing Integer Programs. *ACM Trans. on Algorithms* 3(3), 2007.
16. J. Chuzhoy, V. Guruswami, S. Khanna, K. Talwar. Hardness of Routing with Congestion in Directed Graphs. *Proc of ACM STOC*, 165–178, 2007.
17. N. Garg. V. V. Vazirani, M. Yannakakis. Primal-dual approximation algorithms for integral flow and multicut in trees. *Algorithmica* 18(1):3–20, 1997.
18. V. Guruswami, S. Khanna, B. Shepherd, R. Rajaraman, M. Yannakakis. Near-Optimal Hardness Results and Approximation Algorithms for Edge-Disjoint Paths and Related Problems. *J. of Computer and System Sciences*, 67(3): 473–496, 2003.
19. J. Håstad. Clique is Hard to Approximate within $n^{1-\varepsilon}$. *Acta Mathematica*, 182:105–142, 1999.
20. J. M. Kleinberg. Approximation algorithms for disjoint paths problems. PhD thesis, MIT EECS, 1996.
21. S.G. Kolliopoulos. Edge-disjoint Paths and Unsplittable Flow. In Handbook of Approximation Algorithms and Metaheuristics, ed. T. F. Gonzalez, Chapman & Hall/CRC, 2007.
22. S. G. Kolliopoulos, C. Stein. Approximating disjoint-path problems using greedy algorithms and Packing Integer Programs. *Math. Prog. A*, (99):63–87, 2004.
23. P. Kolman. A Note on the Greedy Algorithm for the Unsplittable Flow Problem. *Information Processing Letters*, 88(3):101–105, 2003.
24. P. Kolman., C. Scheideler. Improved bounds for the unsplittable flow problem. *J. of Algorithms* 61(1): 20–44, 2006.
25. D. Pritchard. Approximability of Sparse Integer Programs. *Proc. of ESA*, 2009, to appear. arXiv.org preprint. `arXiv:0904.0859v1`. http://arxiv.org/abs/0904.0859.
26. P. Raghavan. Probabilistic Construction of Deterministic Algorithms: Approximating Packing Integer Programs. *JCSS* 37(2): 130–143, 1988.
27. A. Schrijver. Combinatorial Optimization: Polyhedra and Efficiency. Springer-Verlag, 2003.
28. B. Shepherd, A. Vetta. The Demand Matching Problem. *Math. of Operations Research*, 32(3): 563–578, 2007.
29. H. Sherali, W. P. Adams. A Hierarchy of Relaxations and Convex Hull Characterizations for Mixed-Integer Zero-One Programming Problems. *Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science*, Vol. 52, 1994.
30. A. Srinivasan. Improved approximations for edge-disjoint paths, unsplittable flow, and related routing problems. *Proc. of IEEE FOCS*, 416–425, 1997.
31. A. Srinivasan. New Approaches to Covering and Packing Problems. *Proc. of ACM-SIAM SODA*, 567–576, 2001.
32. M. Van Vyve, L.A. Wolsey. Approximate Extended Formulations. *Math. Prog.* 105: 501–522, 2006.
33. K. Varadarajan, G. Venkataraman. Graph Decomposition and a Greedy Algorithm for Edge-disjoint Paths. *Proc. of ACM-SIAM SODA*, 379–380, 2004.