Approximating Disjoint-Path Problems Using Packing Integer Programs

Stavros G. Kolliopoulos[1] Clifford Stein[2]

Abstract

In a *packing integer program,* we are given a matrix A and column vectors b, c with nonnegative entries. We seek a vector x of nonnegative integers, which maximizes $c^T x$, subject to $Ax \leq b$. The *edge* and *vertex-disjoint path* problems together with their *unsplittable flow* generalization are NP-hard problems with a multitude of applications in areas such as routing, scheduling and bin packing. These two categories of problems are known to be conceptually related, but this connection has largely been ignored in terms of approximation algorithms. We explore the topic of approximating disjoint-path problems using polynomial-size packing integer programs. Motivated by the disjoint paths applications, we introduce the study of a class of packing integer programs, called *column-restricted.* We develop improved approximation algorithms for column-restricted programs, a result that we believe is of independent interest. Additional approximation algorithms for disjoint-paths are presented that are simple to implement and achieve good performance when the input has a special structure.

Keywords: approximation algorithms, packing integer programs, edge-disjoint paths.

Corresponding author: Stavros Kolliopoulos

[1]Department of Computing and Software, Faculty of Engineering, McMaster University, 1280 Main St. West, Hamilton, Ontario, L8S 4K1, Canada. Email: `stavros@mcmaster.ca.` Fax: +1-905-524-0340. Work partially supported by NSERC OG 227809-00 and a CFI New Opportunities Award. Part of this work was done while at the Department of Computer Science, Dartmouth College and partiallly supported by NSF Award CCR-9308701 and NSF Career Award CCR-9624828.

[2]Department of Computer Science, Dartmouth College, Hanover, NH 03755-3510, USA. E-mail: `cliff@cs.dartmouth.edu`. Research partially supported by NSF Award CCR-9308701 and NSF Career Award CCR-9624828.

1 Introduction

In a *packing integer program* (abbr. PIP), we are given a matrix A and column vectors b, c with nonnegative entries. We seek a vector x of nonnegative integers, which maximizes the inner product $c^T x$, subject to $Ax \leq b$. Packing integer programs are a well-studied class of integer programs, and model several NP-complete problems, including maximum independent set, hypergraph k-matching [31, 1], job-shop scheduling [46, 40, 35, 32]. A great deal of effort has gone into the design of approximation algorithms for these problems with worst-case performance guarantees or more recently the derivation of inapproximability results [2]. For a minimization (resp. maximization) problem, a *ρ-approximation algorithm,* $\rho > 1$, ($\rho < 1$), outputs in polynomial time a solution with objective value at most (at least) ρ times the optimum. Following standard practice, we occasionally refer to a ρ-approximation, $\rho < 1$, for a maximization problem as a $1/\rho$-approximation.

A large category of problems which involve path selection in a graph under capacity constraints, such as finding edge-disjoint paths between selected pairs of terminals, and more generally integral multicommodity flow, have natural packing formulations. In this setting, each entry of the x vector corresponds to a path between a source-sink pair of the underlying flow network, while rows correspond to the capacity constraints on the edges. Relaxing the integrality constraint on x, yields a fractional solution to the underlying flow problem. The given formulation has a variable for each of an exponential number of paths. See [35] for an alternative packing formulation and how despite the exponential size a fractional solution can be efficiently computed. In spite of the close connection between packing and disjoint paths, no approximation algorithms exploiting it have been developed. We explore the topic of approximating disjoint-path problems using polynomial-size packing integer programs. Motivated by disjoint-paths applications we introduce the study of a class of packing integer programs, called *column-restricted.* We develop improved approximation algorithms for column-restricted programs, a result that we believe is of independent interest. Additional approximation algorithms for disjoint-paths are presented that are simple to implement and achieve good performance when the input has special structure. An central theme of our work is the efficient rounding of a solution to a fractional relaxation of a problem, such as PIP or disjoint paths, to an integral feasible solution while incurring bounded degradation to the objective value.

Disjoint-Path Problems. In the *edge(vertex)-disjoint path* problem, we are given a graph $G = (V, E)$ and a set $\mathcal{T}$ of K *connection requests,* also called *commodities.* We assume throughout the paper that G is directed unless explicitly specified otherwise. However all our results hold for undirected graphs as well. Every connection request in $\mathcal{T}$ is a vertex pair (s_i, t_i), $1 \leq i \leq K$. The objective is to connect a maximum number of the pairs via edge(vertex)-disjoint paths. For the vertex-disjoint paths problem, the connection requests are assumed to be disjoint. We call a set of feasibly connected pairs *realizable.* Disjoint-path problems are instances of *admission control* optimization; a subset of requests are bound to be rejected. A generalization of the edge-disjoint paths problem is *multiple-source unsplittable flow.* In this problem every commodity k in the set $\mathcal{T}$

has an associated demand ρ_k, and every edge e has a capacity u_e. The demand ρ_k must be routed on a single path from s_k to t_k. The objective is to maximize the sum of the demands that can be fully routed while respecting the capacity constraints. Without loss of generality, we assume that $\max_k \rho_k = 1$, and following the standard definition of the problem in the literature, $u_e \geq 1$, $\forall e \in E$. Unsplittable flow models the general situation of heterogeneous requirements for different commodities, which have to be routed using the same underlying network. When all demands and capacities are 1 in the multiple-source unsplittable flow problem we obtain the edge-disjoint path problem as a special case. (See [20, 27] for further applications and motivation for unsplittable flow.) In all the above problems one can assign a weight $w_i \leq 1$ to each connection request and define the weight of a realizable set $\mathcal{I}$ as $\sum_{i \in \mathcal{I}} w_i \rho_i$. The objective is to find a realizable set of maximum total weight.

Both the edge- and vertex-disjoint path problems are fundamental, extensively studied (see e.g. [38, 13, 39, 33, 20, 23, 6]), NP-hard problems [18], which combine two basic combinatorial notions: packing and routing. They have applications in diverse areas such as telecommunications, VLSI and scheduling. Disjoint-path problems have been brought to further prominence due to the emergence of high-speed, large-bandwidth networks. It is hoped that a better theoretical understanding will lead to good heuristics for practical admission control and virtual-circuit routing problems. See Kleinberg's thesis for relevant background [21]. In dealing with these computationally hard problems, we focus on ρ-approximation algorithms; we will also give and refer to algorithms, which output solutions whose size is a non-linear function of the optimum value OPT, such as $OPT^2/|E|$. Despite the attention they have received, disjoint-path problems on general graphs remain notoriously hard in terms of approximation; even for edge-disjoint paths, no algorithm is known[1] which can find even an $\omega(1/\sqrt{|E|})$ fraction of the realizable paths.

Overview of previous work. Two main approaches have been followed for approximation.

(i) The first approach, which we call the rounding approach, consists of solving a fractional relaxation and then use rounding techniques to obtain an integral solution. The fractional relaxation is typically multicommodity flow and the rounding techniques used to date involved sophisticated and non-standard use of randomized rounding [43]. The objective value of the resulting solution is compared to the *fractional optimum* y_*, which is an upper bound on the *integral optimum, OPT*. This approach has been the more successful one and recently yielded the first approximation algorithm for *uniform unsplittable flow* [43] which is the special case of unsplittable flow where all the capacities have the same value. Let d denote the *dilation* of the fractional solution, i.e. the maximum length of a flow path in the fractional relaxation. Bounds that rely on the dilation are particularly appealing for expander graphs, where it is known that $d = O(polylog(n))$ [28, 22]. The rounding approach yields, for unweighted uniform unsplittable flow (and thus for unweighted edge-disjoint paths as well) a realizable

[1] Following the first publication of this work some harndess results were shown in [15]. Cf. Section 5.

	routing approach	rounding approach
unweighted EDP	$\frac{OPT}{\sqrt{\lvert E\rvert}}$ [20], $\frac{\mathbf{OPT}}{\sqrt{\lvert\mathbf{E_o}\rvert}}$, $\frac{\mathbf{OPT^2}}{\lvert\mathbf{E_o}\rvert}$, $\frac{\mathbf{OPT}}{\mathbf{d_o}}$	$\frac{y_*}{\sqrt{\lvert E\rvert}}$ [43], $\frac{(y_*)^2}{\lvert E\rvert}$ [43], $\frac{y_*}{d}$ [43]
weighted EDP	—	$\frac{\mathbf{y_*}}{\sqrt{\lvert\mathbf{E}\rvert}}$, $\frac{(y_*)^2}{\lvert E\rvert}$ [43], $\frac{y_*}{d}$ [43]
weighted UCUFP	—	$\frac{(y_*)^2}{\lvert E\rvert}$ [43], $\frac{y_*}{d}$ [43]
weighted UFP	—	$\frac{\mathbf{y_*}}{\log\lvert\mathbf{E}\rvert\sqrt{\lvert\mathbf{E}\rvert}}$, $\frac{(\mathbf{y_*})^{\mathbf{2}}}{\lvert\mathbf{E}\rvert\log^{\mathbf{2}}\lvert\mathbf{E}\rvert\log\log\lvert\mathbf{E}\rvert}$, $\frac{\mathbf{y_*}}{\mathbf{d}}$

Table 1: Known approximation bounds for edge-disjoint paths (EDP), uniform capacity unsplittable flow (UCUFP), and general unsplittable flow (UFP), Ω-notation omitted. E_o denotes the set of edges used by some path in an integral optimal solution and d_o the average length of the paths in the same solution. Results with no citation come from the present paper. Our $y^*/\sqrt{|E|}$ bound for the weighted EDP problem holds under the assumption that the number of connection requests $K = O(|E|)$.

set of size $\Omega(\max\{(y_*)^2/|E|, y_*/\sqrt{|E|}, y_*/d\})$ and an $\Omega(\max\{(y_*)^2/|E|, y_*/d\})$ bound for the weighted version [43] . This approach is known to have limitations, e.g. it is known that a gap of $\Omega(\sqrt{|V|})$ exists between the fractional and integral optima for both the edge- and vertex-disjoint path problems on a graph with $|E| = \Theta(|V|)$ [14].

(ii) Under the second approach, which we call the routing approach, a commodity is never split, i.e. routed fractionally along more than one path during the course of the algorithm. In the analysis, the objective value of the solution is compared to an estimated upper bound on OPT. This approach has found very limited applicability so far, one reason being the perceived hardness of deriving upper bounds on OPT without resorting to a fractional relaxation. The only example of this method we are aware of is the on-line Bounded Greedy Algorithm in [20] whose approximation guarantee depends also on the diameter of the graph. The algorithm can be easily modified into an off-line procedure, which outputs realizable sets of size $\Omega(OPT/\sqrt{|E|})$ ($\Omega(OPT/\sqrt{|V|})$) for edge(vertex)-disjoint paths. The $\Omega(OPT/\sqrt{|V|})$ bound is the best known bound to date for vertex-disjoint paths.

Summary of our contribution. In this paper we provide techniques for approximating disjoint-path problems that bear on both of the above approaches. Tables 1 and 2 summarize previous and new bounds for edge-, vertex-disjoint path and unsplittable flow problems.

We turn to the rounding approach (approach **(i)**) to handle the weighted disjoint path and unsplittable flow problems. We propose the use of *packing integer programs* as a unifying framework that abstracts away the need for customized and complex randomized rounding schemes. We develop, as part of our tools, improved approximation algorithms for a class of packing integer programs, called *column restricted,* that are relevant to unsplittable flow problems. Armed with both this new algorithm and existing algorithms for general packing integer programs, we show how packing

	routing approach	rounding approach
unweighted	$\frac{OPT}{\sqrt{\lvert V\rvert}}$ [20], $\mathbf{\frac{OPT}{\sqrt{\lvert V_o\rvert}}}$, $\mathbf{\frac{OPT^2}{\lvert V_o\rvert}}$, $\mathbf{\frac{OPT}{d_o}}$	$\mathbf{\frac{y^*}{\sqrt{\lvert V\rvert}}}$, $\mathbf{\frac{(y^*)^2}{\lvert V\rvert}}$, $\mathbf{\frac{y^*}{d}}$
weighted	—	$\mathbf{\frac{y^*}{\sqrt{\lvert V\rvert}}}$, $\mathbf{\frac{(y^*)^2}{\lvert V\rvert}}$, $\mathbf{\frac{y^*}{d}}$

Table 2: Known approximation bounds for vertex-disjoint paths, Ω-notation omitted. V_o denotes the set of vertices used by some path in an integral optimal solution and d_o the average length of the paths in the same solution. Results with no citation come from the present paper.

formulations both provide a unified and simplified derivation of many results from [43] and lead to new ones. In particular, we obtain the first approximation algorithm for weighted multiple-source unsplittable flow on networks with arbitrary demands and capacities, and the first approximation algorithm for weighted vertex-disjoint paths. Further, we believe that both our new algorithm for column-restricted packing integer programs and the technique associated with its derivation are of independent interest. Finally, under the routing approach (approach **(ii)**) we provide a greedy algorithm for edge and vertex disjoint paths. This latter algorithm is particularly simple to implement and provides good approximations when the optimal solution has particular structure.

We now elaborate on our results providing further background as necessary. Our presentation is divided into three main parts: packing integer programs; applications of packing to approximating disjoint-path and other problems; the greedy algorithm for disjoint paths.

1.1 Packing Integer Programs

Following [42] a packing integer program (PIP) is defined as follows.

Definition 1.1 *Given $A \in [0,1]^{m\times n}$, $b \in [1,\infty)^m$ and $c \in [0,1]^n$ with $\max_j c_j = 1$, a PIP $\mathcal{P} = (A,b,c)$ seeks to maximize $c^T \cdot x$ subject to $x \in Z_+^n$ and $Ax \leq b$. Constraints of the form $0 \leq x_j \leq d_j$ are also allowed. If $A \in \{0,1\}^{m\times n}$, each entry of b is assumed integral. Let B and α denote respectively $\min_i b_i$, and the maximum number of non-zero entries in any column of A.*

The parameters B and α in the definition above appear in the approximation bounds. For convenience we call b_i the *capacity* of row i. The restrictions on the values of the entries of A, b, c are without loss of generality ; the values in an arbitrary packing program can be scaled to satisfy the above requirements [44]. We will state explicitly when some packing program in this paper deviates from these requirements. When $A \in \{0,1\}^{m\times n}$, we say that we have a $(0,1)$-*PIP*.

Previous Work on Packing Programs. The basic techniques for approximating packing integer programs have been the *randomized rounding* technique of Raghavan and Thompson [36, 37] and the work of Plotkin, Shmoys and Tardos [35]. Randomized rounding schemes in the packing setting can be typically derandomized through the use of suitable pessimistic estimators, resulting in deterministic algorithms. Let y_* denote the optimum value of the linear relaxation. Standard randomized rounding yields integral solutions of value $\Omega(y_*/m^{1/B})$ for general PIPs and $\Omega(y_*/m^{1/(\lfloor B\rfloor+1)})$ for $(0,1)$-PIPs [37] (see also [44].) Observe that for a $(0,1)$-PIP, the integral optimum is not affected if B is rounded down to the nearest integer. Therefore it is not unnatural to work with $\lfloor B\rfloor$. Srinivasan [44, 42] improved on the exponent of m in the above bounds and introduced new ones, depending on α. His work examines more closely the correlation between the packing constraints and introduces probabilistic tools of independent interest. For future reference, we codify previous work in the following theorem.

Theorem 1.1 *[37, 44, 42] Let $\mathcal{P}$ be a PIP conforming to Definition 1.1, and y_* be the optimum of the linear relaxation of $\mathcal{P}$. One can compute in deterministic polynomial time a feasible solution to $\mathcal{P}$ of value*

$$\Omega\left(\max\left\{y_*/m^{1/B}, y_*(y_*/m)^{1/(B-1)}, y_*/\alpha^{1/(B-1)}\right\}\right).$$

If $\mathcal{P}$ is a $(0,1)$-PIP the guarantee on the solution value is

$$\Omega\left(\max\left\{y_*/m^{1/(\lfloor B\rfloor+1)}, y_*(y_*/m)^{1/\lfloor B\rfloor}, y_*/\alpha^{1/\lfloor B\rfloor}\right\}\right).$$

New results for column-restricted PIPs. The above results show that for various combinations of values for y_*, m and B, the bounds obtained for a $(0,1)$-PIP are significantly better than those for general PIPs. In fact they are always better when $y_* < m$. As another example, the approximation ratio $\Omega(1/m^{1/(\lfloor B\rfloor+1)})$ obtained for a $(0,1)$-PIP is polynomially better than the approximation ratio of a PIP with the same parameters. Thus it is natural to ask whether we can bridge the approximability gap between general and $(0,1)$ PIPs. We make progress in this direction by defining a *column-restricted* PIP $\mathcal{P}_r$ as one in which all non-zero entries of the j-th column of A have the same value $\rho_j \leq 1$. Column-restricted PIPs form a natural class in their own right and also arise in applications such as unsplittable flow problems (see next section). We show how to obtain approximation guarantees for column-restricted PIPs that are similar to the ones obtained for $(0,1)$-PIPs. For a column-restricted PIP, we obtain an integral solution of value

$$\Omega\left(\max\left\{\frac{y_*}{m^{1/(\lfloor B\rfloor+1)}},\ \frac{y_*}{\alpha^{1/\lfloor B\rfloor}},\ y_*\left(\frac{y_*}{m\log\log m}\right)^{1/\lfloor B\rfloor}\right\}\right).$$

To solve column-restricted PIPs, we employ the following *grouping-and-scaling* technique. First we find an optimum solution x^* to the linear relaxation of the column-restricted PIP $\mathcal{P}_r$. We partition the ρ_j's into a fixed number of intervals according to their values and generate a packing subproblem for each range. In a packing subproblem $\mathcal{P}^L$ corresponding to range L, we only include the columns of A with $\rho_j \in L$ and to each component of the b^L-vector we allocate only a fraction of the original b_i value, a fraction that is determined based on information from x^*. Next we find approximate solutions to each subproblem and combine them to obtain a solution to the original problem. Perhaps the key idea is in using the solution x^* to define the capacity allocation to the b^L-vector for subproblem $\mathcal{P}^L$. The other key idea is that each subproblem can be approximated almost as well as a $(0,1)$-PIP. This set of ideas generalizes previous work of the authors [27] on single-source unsplittable flow. We believe that the grouping-and-scaling technique will find further applications.

1.2 Applications of packing to approximation

We introduce an approach for applying packing techniques to disjoint-path problems. First, we formulate an integer program (which is not necessarily a PIP) and solve a linear relaxation of this integer program to obtain a solution x. Typically this is a multicommodity flow problem. We then explicitly use the solution x to guide the formation of a column-restricted or $(0,1)$-PIP. A related usage of a solution to the linear relaxation of integer programs in a different context can be found in [19, 45]. An integral approximate solution to the created PIP will be an approximate solution to the original disjoint path problem (with possibly some small degradation in the approximation factor). This integral solution can be found using algorithms for approximating column-restricted or $(0,1)$-PIPs as a black box; we thus abstract away the complications arising from sophisticated randomized rounding schemes and their derandomization [44, 42, 43]. Our algorithms apply to the case when there are weights on the commodities, and thus generalize those of Srinivasan for edge-disjoint paths. This general approach yields four applications which we explain below.

Application 1: weighted unsplittable flow. Let $\alpha^f(\mathcal{T})$ optimum of a fractional relaxation with dilation d. We obtain a realizable set of weight

$$\Omega\left(\max\left\{\frac{\alpha^f(\mathcal{T})}{\log|E|\sqrt{|E|}},\ \frac{(\alpha^f(\mathcal{T}))^2}{|E|\log^2|E|\log\log|E|},\ \frac{\alpha^f(\mathcal{T})}{d}\right\}\right)$$

for unsplittable flow with arbitrary demands and capacities. We provide improved bounds when the maximum demand is bounded away from 1 (cf. Theorem 3.4). In the case where the number K of commodities is $O(|E|)$ we also show how to obtain an unsplittable flow of value

$$\Omega\left(\max\left\{\frac{\alpha^f(\mathcal{T})}{\sqrt{|E|}},\ \frac{(\alpha^f(\mathcal{T}))^2}{|E|\log\log|E|}\right\}\right).$$

For the edge-disjoint path problem this is a natural assumption since at most $|E|$ connection requests can be feasibly routed. We also note that a ρ-approximation for the maximization problem entails an $O(\rho\log|E|)$ approximation for the problem of *routing in rounds* [3, 20]. We do not pursue the latter problem any further in this paper.

Application 2: weighted vertex-disjoint paths. We give an algorithm that outputs a solution of value

$$\Omega\left(\max\left\{\frac{\alpha^f(\mathcal{T})}{\sqrt{|V|}},\ \frac{(\alpha^f(\mathcal{T}))^2}{|V|},\ \frac{\alpha^f(\mathcal{T})}{d}\right\}\right).$$

The algorithm relies on the observation that, after solving a fractional relaxation, the problem of rounding is essentially an instance of hypergraph matching; thus it can be formulated as a packing program with $|V|$ constraints. The algorithm is surprisingly simple but the performance guarantee matches the integrality gap known for the problem [14].

Straightforward modifications of our algorithms give the same guarantees to a slightly different formulation of Applications 1 and 2: find a realizable set $\mathcal{I}$, which maximizes $\sum_{i \in \mathcal{I}} w_i$.

Application 3: routing with low congestion. A problem that has received a lot of attention in the literature on routing problems (e.g. [37, 29, 35, 34, 20, 27]) is that of minimizing *congestion*, i.e. the factor by which one is allowed to scale up capacities in order to achieve an optimal (or near-optimal) realizable set. In our usage of packing in the rounding algorithms we have assumed that the parameter B of the packing program is equal to 1. Allowing $B > 1$ is equivalent to allowing congestion B in the corresponding disjoint-path problem. Thus another advantage of the packing approach is that tradeoffs with the allowed congestion B can be obtained immediately by plugging in B in the packing algorithms that we use as a black box. For example the approximation for edge-disjoint paths becomes $\Omega(\max\{y^*(y^*/|E| \log\log |E|)^{1/B}, y^*/|E|^{1/(B+1)}, y^*/d^{1/B}\})$. Our congestion tradeoffs generalize previous work by Srinivasan [43] who showed the $\Omega(y^*/d^{1/B})$ tradeoff for uniform capacity unsplittable flow. We state explicitly the tradeoff with congestion only for the most general problem we examine, i.e. weighted unsplittable flow, in Theorem 3.4.

Application 4: Independent Set in the square of a graph. Given an undirected graph $G = (V, E)$ the *k-th power* $G^k = (V, E^k)$ of G is a graph where two vertices are adjacent if and only if they are at distance at most k in G. We further demonstrate the power of packing formulations by providing an $O(\sqrt{|V|})$ approximation algorithm for finding a maximum independent set in the square of a graph. We also give results that depend on the maximum vertex degree Δ in G. Our approximation ratio cannot be polynomially improved in the sense that no $n^{1/2-\varepsilon}$ approximation, for any fixed $\varepsilon > 0$, can be obtained in polynomial time unless $NP \subseteq ZPP$. Studying NP-hard problems in powers of graphs is a topic that has received some attention in the literature [12, 47, 30, 8]. Finally we note that an $O(\sqrt{|V|})$-approximation for maximum independent set in even powers of a graph has been derived independently by Halldórsson, Kratochvíl, and Telle [16].

1.3 A simple greedy algorithm

In addition to our packing-based results we give also a simple greedy algorithm Greedy_Path for edge- and vertex-disjoint paths that follows the routing approach (approach **(ii)**). Greedy_Path has the advantage that it is particularly simple to implement, performing only a series of breadth-first search computations. On the other hand, its performance guarantees are comparable to those obtained by the much more sophisticated multicommodity flow based algorithms and even existentially better depending on the input. It could therefore be of interest as a basis for practical heuristics.

GREEDY_PATH outputs a realizable set of size $\Omega(\max\{OPT^2/|E_o|, OPT/\sqrt{|E_o|}\})$ for the edge-disjoint path problem. Here $E_o \subseteq E$ is the set of edges used by the paths in an optimal solution. Note that $OPT^2/|E_o|$ always dominates $OPT/\sqrt{|E_o|}$ in the unweighted case that we consider; we give both bounds to facilitate comparison with existing work and to conform to the traditional notion of a ρ-approximation algorithm. Our approximation existentially improves upon the multicommodity-flow based results when $|E_o| = o(|E|)$, i.e. when the optimal solution uses a small portion of the edges of the graph. Another bound can be obtained by noticing that $OPT^2/|E_o| = OPT/d_o$, where d_o denotes the average length of the paths in an optimal solution.

Essentially the same algorithm, GREEDY_VPATH, obtains for the vertex-disjoint path problem a realizable set of size $\Omega(\max\{OPT^2/|V_o|, OPT/\sqrt{|V_o|}\})$, where $V_o \subseteq V$ is the set of vertices used by the paths in an optimal solution. Recall that the best known bound to date is $t = \Omega(OPT/\sqrt{|V|})$. The realizable set output by our algorithm has size $\Omega(t^2)$ and potentially better than this when $|V_o| = o(|V|)$. This is a significant improvement when $OPT = \omega(\sqrt{|V|})$. For example, when $OPT = \Omega(|V|)$, we obtain a constant-factor approximation. Again an $\Omega(OPT/d_o)$ guarantee follows immediately.

An outline of the paper is as follows. In Section 2 we present approximation algorithms for packing integer programs. In Section 3 we show how unsplitable flow and related problems can be approximated by resorting to packing. In Section 4 we present the greedy algorithm. We conclude with a general discussion in Section 5. This paper is a revised version of [26]. Independently of our work, Baveja and Srinivasan [5] have obtained results similar to ours for approximating vertex-disjoint paths under the rounding approach, unsplittable flow and column-restricted packing integer programs. Their work follows a different approach and builds on the methods in [43].

After the first publication of this work in IPCO 98 [26] new results for column-restricted PIPs appeared in [4]. Kolliopoulos recently extended the algorithmic techniques from the present paper to problems related to column-restricted Covering Integer Programs [24]. Hardness results for disjoint path problems and approximation algorithms partly inspired from our greedy approach appeared in the paper by Guruswami et al. [15]. In particular Guruswami et al. show that our bounds for disjoint paths and hence unsplittable flow are close to best possible. We elaborate on their bounds in our conclusion in Section 5.

2 Approximating a column-restricted PIP

2.1 Notation and basic facts

All logarithms in this paper are base 2 unless otherwise noted. Let $\mathcal{P} = (A, b, c)$ be a column-restricted PIP. We call $\rho_j \leq 1$, the value of the non-zero entries of the j-th column, $1 \leq j \leq n$, the *value* of column j. Throughout this section we use y_* to denote the optimum of the linear relaxation of the PIP under consideration. We will also refer to y_* as the *fractional optimum* of $\mathcal{P}$ and to a solution of the linear relaxation of $\mathcal{P}$ as a *fractional solution.* We now introduce some notation.

Given α_i, α_j, let J^{α_i,α_j} be the set of column indices k for which $\alpha_i < \rho_k \leq \alpha_j$. We then define, A^{α_i,α_j} to be the $m \times |J^{\alpha_i,\alpha_j}|$ submatrix of A consisting of the columns in J^{α_i,α_j}, and for any vector x, x^{α_i,α_j} to be the $|J^{\alpha_i,\alpha_j}|$-entry subvector x consisting of the entries whose indices are in J^{α_i,α_j}. We will also need to combine back together the various subvectors, and define $x^{\alpha_1,\alpha_2} \cup \cdots \cup x^{\alpha_{k-1},\alpha_k}$ for $\alpha_1 < \alpha_2 < \ldots \alpha_k$, to be the n-entry vector x' in which the entries of subvector $x^{\alpha_{i-1},\alpha_i}$, $1 < i \leq k$, are mapped back into the position indexed by $J^{\alpha_{i-1},\alpha_i}$. Any positions in x' which are not indexed in $\cup_{1<i\leq k} J^{\alpha_{i-1},\alpha_i}$ are set to 0. Let x_* be the optimal solution of the linear relaxation of $\mathcal{P}$, i.e., $y_* = c^T x_*$. We use $y_*^{r_1,r_2}$ to denote $c^{r_1,r_2} x_*^{r_1,r_2}$.

When column values are much smaller than B, one can obtain a constant-factor approximation. In particular, the randomized rounding scheme of Raghavan and Thompson [37] together with a straightforward application of the Chernoff bound [7] yields the well-known result we present in Theorem 2.2. We give first the relevant Chernoff-type bound.

Theorem 2.1 *[7, 36] If Y is the sum of independent random variables each in $[0, U]$, with $E(Y) \leq \mu$, then, for any $\epsilon > 0$,*

$$Pr[Y \geq (1+\epsilon)\mu] \leq e^{-\min\{\epsilon,\epsilon^2\}\mu/3U}.$$

For the sake of completeness we sketch the proof of the following basic result.

Theorem 2.2 *Let $\mathcal{P} = (A, b, c)$ be a column-restricted PIP such that the maximum column value $\rho_{\max}$ is at most $1/(\beta \log m)$ for appropriate constant β. Then one can obtain in deterministic polynomial time a feasible solution to $\mathcal{P}$ of value at least γy_* for a constant $0 < \gamma < 1$.*

Proof sketch. For simplicity, we assume that the entries of a fesible solution are restricted to lie in $\{0, 1\}$. Let x_* be the n-vector, which solves optimally the linear relaxation of $\mathcal{P}$. Obtain an integral vector $\bar{x}$ by the following random experiment. Set $\bar{x}_j$ to 1 with probability $(1/2)x_{*j}$ and to 0 with probability $1 - (1/2)x_{*j}$. Consider now the m packing constraints, and let $Z_i = \sum_{j=1}^{n} A_{ij}\bar{x}_j$, $1 \leq i \leq m$. Z_i is the sum of at most n independent random variables each lying in $[0, (\beta \log m)^{-1}]$; The expectation $E(Z_i)$ is at most $(1/2)b_i$. The event $\{Z_i > b_i\}$ corresponds to the i-th constraint being violated by solution $\bar{x}$. By the Chernoff bound with $\epsilon = 1$,

$$Pr[Z_i > b_i] = Pr[Z_i > 2(1/2)b_i] \leq e^{-(1/2)b_i/3(\beta \log m)^{-1}}.$$

Using the fact that $b_i \geq 1$, and setting $\beta = 12$, we obtain that

$$Pr[Z_i > b_i] < \frac{1}{m^2}.$$

Invoking the union bound, the probability that any of the m constraints is violated is at most $1/m$. The expected value of the objective $c^T\bar{x}$ is $(1/2)$ times the fractional optimum. It can be seen that with probability at least $1/3$, $c^T\bar{x}$ has value at least $(1/4)$ times the fractional optimum. Therefore, with probability at least $1/3 - 1/m$, the random experiment yields a feasible solution to

$\mathcal{P}$ of value at least $(1/4)y_*$. The outlined algorithm can be derandomized by applying the method of conditional probabilities [10, 41, 36]. ∎

Theorem 2.2 combined with the linearity of the objective function has interesting consequences. If $y_*^{0,(\beta\log m)^{-1}} \geq y_*/2$, a vector $\bar{x}'$ yielding a constant-factor approximation to the fractional optimum of $\mathcal{P}' = (A^{0,(\beta\log m)^{-1}}, b, c^{0,(\beta\log m)^{-1}})$, can be trivially extended with zeros to a feasible solution to $\mathcal{P}$, yielding a constant-factor approximation to y_*. Similarly if $y_*^{(\beta\log n)^{-1},1} \geq y_*/2$, a vector $\bar{x}''$ yielding a ρ-approximation to the fractional optimum of $\mathcal{P}'' = (A^{(\beta\log n)^{-1},1}, b, c^{(\beta\log n)^{-1},1})$, can be extended to a $(\rho/2)$-approximation to y_* for $\mathcal{P}$. Therefore we assume in the remainder of this section that the minimum column value in A is $\Omega(1/\log m)$.

Our goal is to demonstrate that column-restricted PIPs admit approximations of similar quality to those known for $(0,1)$-PIPs. Accordingly, we will give many of our results in terms of a generic bound for a $(0,1)$-PIP. Throughout this section we assume that there is a polynomial-time algorithm $\mathcal{A}$, which given a $(0,1)$-PIP with fractional optimum y_*, outputs an integral solution of value at least $\sigma(m, \lfloor B\rfloor, \alpha, y_*)$; m, B, α are the parameters of the packing program under consideration. The known functions σ with this property are given in Theorem 1.1. For example a known σ is $\Omega(y_*/m^{1+\lfloor B\rfloor})$.

2.2 The approximation algorithm

At a high level our algorithm partitions $\mathcal{P}$ into subproblems, each containing a subset of the columns of matrix A. The column values in each subproblem will be close in range. We start by providing a subroutine for solving a column-restricted PIP when the column values are close in range.

Theorem 2.3 *Let $\mathcal{P} = (A, b, c)$ be a column-restricted PIP, in which all column values ρ_j are equal to ρ and each b_i equals $k_i\Gamma\rho$, $\Gamma \geq 1$, k_i positive integer, $1 \leq i \leq m$. Here* $\min_i b_i$ *is not necessarily greater than* 1. *Let y_* denotes the optimum of the linear relaxation of $\mathcal{P}$ and $k = min_i k_i$. Then we can find in polynomial time a feasible solution for $\mathcal{P}$ of value at least $\sigma(m, \lfloor k\Gamma\rfloor, \alpha, y_*)$.*

Proof. Transform the given program $\mathcal{P}$ to a $(0,1)$-PIP $\mathcal{P}' = (A', b', c)$, in which $b'_i = k_i\Gamma$, and $A'_{ij} = A_{ij}/\rho$. Every feasible solution (either fractional or integral) $\bar{x}$ to $\mathcal{P}'$ is a feasible solution to $\mathcal{P}$ and vice versa. Therefore the fractional optimum y_* is the same for both programs. Also the maximum number of non-zero entries on any column is the same for A and A'. Thus we can unambiguously use α for both. Invoking algorithm $\mathcal{A}$ for $\mathcal{P}'$ returns an integral solution of objective value $\sigma(m, \lfloor k\Gamma\rfloor, \alpha, y_*)$. ∎

The proof of the following lemma generalizes that of Lemma 4.1 in [21].

Lemma 2.1 *Let $\mathcal{P} = (A, b, c)$, be a column-restricted PIP with column values in the interval $(a_1, a_2]$, and $b_i \geq \Gamma a_2$, $\forall i$ and some number $\Gamma \geq 1$. Here* $\min_i b_i$ *is not necessarily greater than* 1. *Let y_* denote the optimum of the linear relaxation of $\mathcal{P}$. There is a polynomial-time algorithm* s_Packing, *which finds a solution g to $\mathcal{P}$ of value at least $\sigma(m, \lfloor\Gamma\rfloor, \alpha, \frac{a_1}{2a_2}y_*)$.*

Proof. We sketch the algorithm s_PACKING. Obtain a PIP $\mathcal{P}' = (A', b', c)$ from $\mathcal{P}$ as follows. Round down b_i to the nearest multiple of Γa_2 and then multiply it by a_1/a_2. Set b'_i equal to the resulting value. Every b'_i is now between $a_1/2a_2$ and a_1/a_2 times the corresponding b_i. Set A'_{ij} to a_1 if $A_{ij} \neq 0$ and to 0 otherwise. If x_* is an optimal fractional solution to $\mathcal{P}$, $(a_1/2a_2)x_*$ is a feasible fractional solution to $\mathcal{P}'$ of value at least $(a_1/2a_2)c^T x_* = (a_1/2a_2)y_*$.

All column values in $\mathcal{P}'$ are equal to a_1 and every b'_i is a multiple of Γa_1. Thus we can invoke Theorem 2.3 and find a feasible solution g' to $\mathcal{P}'$ of value at least $v = \sigma(m, \lfloor \Gamma \rfloor, \alpha, (a_1/2a_2)y_*)$. Observe that g' is also feasible for $\mathcal{P}$. The value is again v since $\mathcal{P}'$ and $\mathcal{P}$ have the same objective. ∎

Remark 2.1 *The statement of Lemma 2.1 holds also when the column values of $\mathcal{P}$ lie in $[a_1, a_2]$.*

The following lemma is a specialized version of Lemma 2.1.

Lemma 2.2 *Let $\mathcal{P} = (A, b, c)$, be a column-restricted PIP with column values in the interval $(a_1, a_2]$, and b_i a multiple of Γa_2, $\forall i$ and some number $\Gamma \geq 1$. Here $\min_i b_i$ is not necessarily greater than 1. Let y_* denote the optimum of the linear relaxation of $\mathcal{P}$. There is a polynomial-time algorithm* INTERVAL_PACKING, *which finds a solution g to $\mathcal{P}$ of value at least $\sigma(m, \lfloor \Gamma \rfloor, \alpha, \frac{a_1}{a_2} y_*)$.*

Proof. Similar to that of Lemma 2.1. Since all b_i's are already multiples of Γa_2, we don't pay the 1/2 factor for the initial rounding. ∎

We now outline in some detail the grouping-and-scaling method employed in the full algorithm. The technique generalizes earlier work of the authors on single-source unsplittable flow [27]. Let x_* denote the optimum solution to the linear relaxation of $\mathcal{P}$. We are going to create packing subproblems $P^\lambda = (A^\lambda, b^\lambda, c^\lambda)$ such that A^λ contains only the columns of A with values in some fixed range $(\alpha_{\lambda-1}, \alpha_\lambda]$. The subproblems will be amenable to good approximations because of the fixed range of the column values. We will obtain our integral solution to $\mathcal{P}$ by combining these approximate solutions to the subproblems. The crucial step is capacity allocation to subproblems. Consider a candidate for the b^λ-vector that we call the λ-th *fractional capacity vector.* In the fractional capacity vector the i-th entry is equal to $\sum_{j|\rho_j \in (\alpha_{\lambda-1}, \alpha_\lambda]} A_{ij} x_{*j}$. In other words, we allocate capacity to the λ-th subproblem by "pulling" out of b the amount of capacity used up in the solution x^* by the columns with values in $(\alpha_{\lambda-1}, \alpha_\lambda]$. The benefit of such a scheme would be that by using the relevant entries of x^* we could obtain a feasible solution to the linear relaxation of P^λ. However, to be able to benefit from Lemma 2.1 when finding an approximate integral solution to each P^λ, we need all entries of b^λ to be larger than $B\alpha_\lambda$. This increases the required capacity for each subproblem potentially above the fractional capacity. We resort to more sophisticated capacity allocation to ensure that

(i) there is enough total capacity in the b-vector of $\mathcal{P}$ to share among the subproblems

(ii) the subproblems can benefit from the subroutines in Lemmata 2.1, 2.2 and

ALGORITHM COLUMN_PARTITION($\mathcal{P}$)

Step 1. Find the n-vector x_* that yields the optimal solution to the linear relaxation of $\mathcal{P}$.

Step 2a. Define a partition of the $[(\beta \log m)^{-1}, 1]$ interval into $\xi = O(\log \log m)$ consecutive subintervals $[(\beta \log m)^{-1}, 4^{-\lfloor \log_4(\beta \log m) \rfloor}], \ldots, (4^{-\lambda}, 4^{-\lambda+1}], \ldots, (4^{-2}, 4^{-1}], (4^{-1}, 1]$. For $\lambda = 1, \ldots, \xi - 1$ form subproblem $P^\lambda = (A^\lambda, b^\lambda, c^\lambda)$. A^λ and c^λ are the restrictions defined by $A^\lambda = A^{4^{-\lambda}, 4^{-\lambda+1}}$ and $c^\lambda = c^{4^{-\lambda}, 4^{-\lambda+1}}$. Define similarly $P^\xi = (A^\xi, b^\xi, c^\xi)$ such that $A^\xi = A^{(\beta \log m)^{-1}, 4^{-\lfloor \log_4(\beta \log m) \rfloor}}$, $c^\xi = c^{(\beta \log m)^{-1}, 4^{-\lfloor \log_4(\beta \log m) \rfloor}}$.

Step 2b. Let d_i^λ be the ith entry of $(A^\lambda \cdot x_*^\lambda)$. Define $b^\lambda = b$ when λ is 1, and otherwise $b_i^\lambda = B(1/4)^{\lambda-1} + (1/2)d_i^\lambda$ when $\lambda = 2, 3, \ldots, \xi$.

Step 3. On each P^λ, $2 \leq \lambda \leq \xi$, invoke S_PACKING to obtain a solution vector x^λ. Combine the solutions to subproblems 2 through ξ to form n-vector $\hat{x} = \cup_{2 \leq \lambda \leq \xi} x^\lambda$.

Step 4. Invoke INTERVAL_PACKING on P^1 to obtain a solution vector x^1. Let $\bar{x} = \cup x^1$.

Step 5. Of the two vectors $\hat{x}$ and $\bar{x}$, output the one, call it $\dot{x}$, which maximizes $c^T \cdot \dot{x}$.

Figure 1: Algorithm COLUMN_PARTITION.

(iii) the fractional optimum of the λ-th subproblem, for all λ, is within a constant factor of the subproblem's contribution to y_*, i.e., $y_*^{\alpha_{\lambda-1}, \alpha_\lambda}$.

In particular, we initially assign to each subproblem λ only 1/2 of the fractional capacity vector; this has the asymptotically negligible effect of scaling down the fractional optimum for each subproblem by at most 2. We exploit the unused $(1/2)b$ vector of capacity to add an extra $B\alpha_\lambda$ units to the entries of b^λ, for all λ.

For a vector t, if $r_1 = (1/4)^\lambda$ and $r_2 = (1/4)^{\lambda-1}$, then we abbreviate t^{r_1, r_2} as t^λ. Similarly for $y_*^{r_1, r_2}$. We present the full algorithm in Figure 1. Recall from Section 2.1 that the minimum column value is assumed to be $1/(\beta \log m)$ for some constant β.

Two tasks remain. First, we must show that the vector $\dot{x}$ output by the algorithm is a feasible solution to the original packing problem $\mathcal{P}$. Second, we must lower bound $c^T \cdot \dot{x}$ in terms of the optimum $y_* = c^T \cdot x_*$ of the linear relaxation of $\mathcal{P}$. We examine first the vector $\hat{x}$.

Lemma 2.3 *Algorithm* Column_Partition *runs in polynomial time and the n-vector $\hat{x}$ it outputs is a feasible solution to $\mathcal{P}$ of value at least*

$$\sum_{\lambda=2}^{\xi} \sigma(m, \lfloor B \rfloor, \alpha, (1/16) y_*^\lambda), \quad \xi = O(\log \log m).$$

Proof. Let $\hat{y}^\lambda$ be the optimum of the linear relaxation of P^λ. By Lemma 2.1 the value of the solution x^λ, $2 \le \lambda \le \xi$, found at Step 3 is at least $\sigma(m, \lfloor B \rfloor, \alpha, (1/2)(1/4)\hat{y}^\lambda)$. By the definition of b^λ in P^λ, $(1/2)x_*^\lambda$,i.e., the restriction of x_* scaled by 1/2, is a feasible fractional solution for P^λ. Thus $\hat{y}^\lambda \ge (1/2)y_*^\lambda$. Hence, the value of x^λ is at least $\sigma(m, \lfloor B \rfloor, \alpha, (1/16)y_*^\lambda)$. The claim of the theorem on the value follows.

For the feasibility, we observe that the aggregate capacity used by $\hat{x}$ on row i of A is the sum of the capacities used by x^λ, $2 \le \lambda \le \xi$, on each subproblem. This sum is by Step 2b at most

$$(1/2) \sum_{\lambda=2}^{\xi} d_i^\lambda + B \sum_{\lambda=2}^{\xi} (1/4)^{\lambda-1}.$$

But

$$B \sum_{\lambda=2}^{\xi} (1/4)^{\lambda-1} < (1/2)B \le (1/2)b_i$$

and

$$\sum_{\lambda=2}^{\xi} d_i^\lambda \le b_i.$$

Thus the aggregate capacity used by $\hat{x}$ at row i is at most

$$(1/2)b_i + (1/2)b_i = b_i.$$

Therefore $\hat{x}$ is a feasible solution to $\mathcal{P}$. ∎

It remains to account for $\bar{x}$. The following theorem is the main result of this section.

Theorem 2.4 *Let $\mathcal{P} = (A, b, c)$ be a column-restricted PIP and y_* be the optimum of the linear relaxation of $\mathcal{P}$. Algorithm* Column_Partition *finds in polynomial time a solution g to $\mathcal{P}$ of value*

$$\Omega\left(\max\left\{ \frac{y_*}{m^{1/(\lfloor B \rfloor + 1)}}, \frac{y_*}{\alpha^{1/\lfloor B \rfloor}}, y_* \left(\frac{y_*}{m \log \log m} \right)^{1/\lfloor B \rfloor} \right\} \right).$$

Proof. Each of the two vectors $\hat{x}, \bar{x}$ solves a packing problem with column values lying in $((\beta \log m)^{-1}, 1/4]$ and $(1/4, 1]$ respectively. Let $\hat{\mathcal{P}}, \bar{\mathcal{P}}$ be the two induced packing problems. The optimal solution to the linear relaxation of at least one of them will have value at least 1/2 of the optimum y_* of the linear relaxation of $\mathcal{P}$. It remains to lower bound the approximation achieved by each of the two vectors on its corresponding domain of column values. Since m, B, and α are the same for all subproblems, note that σ is a function of one variable, y_*.

By Lemma 2.1, vector $\bar{x}$ outputs a solution to $\bar{\mathcal{P}}$ of value at least $\sigma(m, \lfloor B \rfloor, \alpha, (1/4)y_*^{1/4,1})$. Therefore if $y_*^{1/4,1} \geq y_*/2$ the theorem follows trivially. We assume from now on that $y_*^{(\beta \log m)^{-1},1/4} \geq y_*/2$ and focus on the value of $c^T \cdot \hat{x}$.

By Lemma 2.3, the value of the solution output for $\hat{\mathcal{P}}$, is lower bounded by the sum

$$A \equiv \sum_{\lambda=2}^{\xi} \sigma(m, \lfloor B \rfloor, \alpha, (1/16)y_*^{\lambda}).$$

We distinguish two cases reflecting the two different types of bounds given in Theorem 1.1.
Case A: σ is a function linear in y_*. Then

$$A \geq \sigma(m, \lfloor B \rfloor, \alpha, (1/16)y_*^{(\beta \log m)^{-1},1/4}) \geq \sigma(m, \lfloor B \rfloor, \alpha, (1/32)y_*).$$

By Theorem 1.1 we can instantiate $\sigma(z_*)$ with a function $\Omega(\max\{z_*/m^{1/(\lfloor B \rfloor+1)}, z_*/\alpha^{1/\lfloor B \rfloor}\})$ and obtain the two linear bounds of the theorem.
Case B: σ is a function convex in y_*. The sum A has $\Theta(\log\log m)$ terms, hence A is minimized when all the terms y_*^{λ} are equal to $\Theta(y_*^{(\beta \log m)^{-1},1/4}/\log\log m) = \Theta(y_*/\log\log m)$. Therefore

$$A \geq A' \equiv \Omega(\sum_{\lambda=2}^{\xi} \sigma(m, \lfloor B \rfloor, \alpha, y_*/\log\log m)).$$

By Theorem 1.1 we can substitute a function $\Omega(z_*(z_*/m)^{1/\lfloor B \rfloor})$ for $\sigma(z_*)$ in A' and obtain the convex bound of the the theorem. ∎

The approximation ratios achieved for column-restricted and general $(0,1)$-PIPs are admittedly small, but there is evidence they cannot be polynomially improved in the general case. Finding a maximum independent set in an undirected graph $G = (V, E)$ can be formulated as a $(0,1)$ packing integer program $\mathcal{P}_G$ with $|E|$ constraints and $B = 1$. No ρ-approximation, $\rho = |V|^{1-\varepsilon}$, $\varepsilon > 0$, can be obtained in polynomial time for maximum independent set unless $NP \subseteq ZPP$ [17]. Thus no $1/|E|^{1/2-\varepsilon}$ approximation can be achieved for $\mathcal{P}_G$, unless $NP \subseteq ZPP$. The above discussion leads to the following theorem, which we consider part of the folklore.

Theorem 2.5 *There exists a family of column-restricted PIPs, for which no approximation better than $1/m^{1/(\lfloor B \rfloor+1)-\varepsilon}$, $\varepsilon > 0$, can be obtained in polynomial time, unless $NP \subseteq ZPP$.*

Theorem 2.2 demonstrates that when the maximum column value is bounded by $O(1/\log m)$, one can obtain a constant-factor approximation. This does not contrast Theorem 2.5 in any way. When the maximum column value is bounded by $\zeta < 1$, one can multiply the column values and the b-vector by $1/\zeta$ and obtain an equivalent integer program. Improved approximations are obtained by substituting B/ζ for B in the bounds of Theorem 2.4.

Corollary 2.1 *Let $\mathcal{P} = (A, b, c)$ be a column-restricted PIP with maximum column value $\zeta < 1$ and y_* be the optimum of the linear relaxation of $\mathcal{P}$. Algorithm* COLUMN_PARTITION *finds in polynomial time a solution g to $\mathcal{P}$ of value*

$$\Omega\left(\max\left\{\frac{y_*}{m^{1/(\lfloor (B/\zeta) \rfloor+1)}}, \frac{y_*}{\alpha^{1/\lfloor (B/\zeta) \rfloor}}, y_*\left(\frac{y_*}{m\log\log m}\right)^{1/\lfloor (B/\zeta) \rfloor}\right\}\right).$$

3 Applications of PIPs to approximation

3.1 Weighted multiple-source unsplittable flow

Our approach for weighted unsplittable flow consists of finding first the optimum of the fractional relaxation, i.e. weighted multicommodity flow, which can be solved in polynomial time via linear programming. For the unweighted version of the problem, a fast combinatorial algorithm can be used instead [46]. The relaxation consists of allowing commodity k to be shipped along more than one path. Call these paths the *fractional paths.* At this point, a simple rounding scheme for the fractional solution would be to pack as many of the fractional paths as possible, each carrying flow equal to some demand ρ_i. The problem with this naive approach is that we may end up sending for some commodity k, $l\rho_k$ units of flow along $l > 1$ distinct paths. Therefore we round in two stages. In the first stage we select at most one of the fractional paths for each commodity, at the expense of congestion, i.e. some capacities may be violated. In addition, some commodities may not be routed at all. The first stage is implemented by resorting to a well-known result, given in Theorem 3.1 below. In the second stage, among the commodities routed during the first stage, we select those that will ultimately be routed while respecting the capacity constraints. It is in this last stage that a column-restricted PIP is used.

A well-known analogue of Theorem 2.2, follows using standard randomized rounding [37, 36] and the Chernoff bound of Theorem 2.1. The proof is quite similar in spirit to that of Theorem 2.2. See for example, [20] for a randomized algorithm.

Theorem 3.1 *Let $\Pi = (G = (V, E), \mathcal{T})$ be a weighted multiple-source unsplittable flow problem such that the maximum demand value $\rho_{\max}$ is at most $1/(\beta \log |E|)$ times the minimum capacity, for appropriate constant β. Let $\alpha^f(\mathcal{T})$ be the value of a fractional routing f. Then one can obtain in deterministic polynomial time a feasible solution to Π of value at least $\gamma\alpha^f(\mathcal{T})$ for a constant $0 < \gamma < 1$.*

In light of Theorem 3.1, we will focus solely on instances with minimum demand value $1/(\beta \log |E|)$; the rationale behind this decision is the same as for column-restricted PIPs in Section 2.1. We introduce some terminology before giving the algorithm. A *routing* is a set of s_{k_i}-t_{k_i} paths P_{k_i}, used to route ρ_{k_i} amount of flow from s_{k_i} to t_{k_i} for each $(s_{k_i}, t_{k_i}) \in I \subseteq \mathcal{T}$. Given a routing g, the flow g_e through edge e is equal to $\sum_{P_i \in g, P_i \ni e} \rho_i$. A routing g for which $g_e \leq u_e$ for every edge e is an *unsplittable flow.* A *fractional routing* is a routing, in which for commodity k (i) the flow is split along potentially many paths (ii) total flow $f_k \leq \rho_k$ is routed. A fractional routing corresponds thus to standard multicommodity flow. A *fractional single-path routing* is a routing, in which the flow for a commodity is shipped on one path if at all, but only a fraction $f_k \leq \rho_k$ of the demand is shipped for commodity k. The *value* of a routing g (fractional or otherwise) is the weighted sum of the flow routed in g for each commodity. Recall that by the definition of the problem, the maximum demand and the minimum capacity values are respectively at most and at least 1. The full algorithm is given in Figure 2.

ALGORITHM MAX_ROUTING($G = (V, E, u), \mathcal{T}$)

Step 1. Find an optimum fractional routing f by invoking a weighted multicommodity flow algorithm. Denote by $\alpha^f(\mathcal{T})$ the value of f.

Step 2. Scale up all capacities by a factor of $\beta \log |E|$ to obtain network G' with capacity function u'. Invoke the algorithm from Theorem 3.1 on G' to round f to a routing g'.

Step 3. Scale down the flow on every path of g' by a factor of at most $\beta \log |E|$ to obtain a fractional single-path routing g, which is feasible for G.

Step 4. Construct a column-restricted PIP $\mathcal{P} = (A, b, c)$ as follows. Let $\{k_1, k_2, \ldots, k_\nu\}$ be the set of commodities shipped in g, $\nu \leq K$. A has ν columns, one for each commodity in g, and $|E|$ rows, one for each edge of G. $A_{ij} = \rho_{k_j}$ if the path P_{k_j} in g for commodity k_j goes through edge e_i and 0 otherwise. The cost vector c has entry c_j set to $w_{k_j}\rho_{k_j}$ for each commodity k_j shipped in g. Finally, b_i is set to u_{e_i}, the capacity of edge e_i, $1 \leq i \leq |E|$.

Step 5. Invoke algorithm COLUMN_PARTITION to find an integral solution $\hat{g}$ to $\mathcal{P}$. Construct an unsplittable flow g'' by routing commodity k_j on path P_{k_j} if and only if $\hat{g}_j = 1$.

Figure 2: Algorithm MAX_ROUTING.

Theorem 3.2 *Let $(G = (V,E), \mathcal{T})$ be a weighted multiple-source unsplittable flow problem, and $\alpha^f(\mathcal{T})$ be the value of an optimum fractional routing f. Algorithm* Max_Routing *finds in polynomial time an unsplittable flow of value*

$$\Omega\left(\max\left\{\frac{\alpha^f(\mathcal{T})}{\log|E|\sqrt{|E|}}, \frac{(\alpha^f(\mathcal{T}))^2}{|E|\log^2|E|\log\log|E|}\right\}\right).$$

Proof. First note that since routing g' is feasible for G', after scaling down the flow at Step 3, routing g is feasible for G. The approximate solution to $\mathcal{P}$ maintains feasibility.

We now examine the claim of the theorem on the value of $\hat{g}$. By Theorem 3.1, the value of g' is a constant fraction of $\alpha^f(\mathcal{T})$. Set z_{k_j} to be equal to the amount of flow that is routed in g for commodity k_j divided by ρ_{k_j}. The ν-vector z is a feasible fractional solution to $\mathcal{P}$ of value $\Omega(\alpha^f(\mathcal{T})/\log|E|)$. The theorem follows by using Theorem 2.4 to lower bound the value of the integral solution $\hat{g}$ to $\mathcal{P}$. ∎

We now give a bound that depends on the dilation d of the fractional routing f.

Theorem 3.3 *Let $(G = (V,E), \mathcal{T})$ be a weighted multiple-source unsplittable flow problem, and $\alpha^f(\mathcal{T})$ be the value of an optimum fractional routing f with dilation d. There is a polynomial-time algorithm, which finds an unsplittable flow of value $\Omega(\alpha^f(\mathcal{T})/d)$.*

Proof. The algorithm is similar to Max_Routing but omits the Steps 2 and 3 that find the single-path fractional routing. Step 4 is modified as follows. Let $1, \ldots, M$ be an arbitrary ordering of the paths in the flow decomposition of f. Matrix A of $\mathcal{P}$ has $M + |\mathcal{T}|$ total columns, one for each path in the decomposition and an additional $|\mathcal{T}|$ special columns. The number of rows is $|E| + |\mathcal{T}|$. The first $|E|$ rows correspond to capacity constraints in G and their entries are filled in the manner described in Algorithm Max_Routing. The remaining rows correspond to constraints that enforce that only one path is chosen per commodity in the final integral solution. In particular row $|E| + i$, $1 \le i \le |\mathcal{T}|$, has ρ_i in column j, $1 \le j \le M$, if the j-th path in the flow decomposition carries flow for commodity i; otherwise the entry is zero. The entries of column $M + l$, $1 \le l \le |\mathcal{T}|$, are all zero except for the $(|E| + l)$-th row where the entry is equal to $1 - \rho_l$. Finally, each entry $b_{|E|+i}$, $1 \le i \le |\mathcal{T}|$, of the b-vector is set to 1. We set c_i, $1 \le i \le M$, to $w_{k(i)}\rho_{k(i)}$ where $k(i)$ is the commodity that flows along the i-th path in the decomposition. The entries in the c-vector past the M-th position are set to 0.

The fractional solution $\bar{x}$ we shall feed into our rounding algorithm Column_Partition has the $\bar{x}_i$-variables, $1 \le i \le M$ defined by the flow decomposition and $\bar{x}_i$ set to 1 for $i = M + l$, $1 \le l \le |\mathcal{T}|$. Clearly $\bar{x}$ is feasible for $\mathcal{P}$ with value $\alpha^f(\mathcal{T})$. Moreover $\mathcal{P}$ is column-restricted with maximum number of non-zero entries in a column equal to $d+1$. Therefore algorithm Column_Partition will return an integral solution of value as given by the theorem. It is straightforward to verify that the above construction enforces an integral solution where at most one of the fractional paths is chosen for each commodity. The theorem follows. ∎

The construction in the proof of Theorem 3.3 can also be used to obtain improved bounds for the case, in which the number of commodities $|\mathcal{T}| = O(|E|)$. For example this is true in the case of the edge-disjoint path problem. More generally, the approximation bounds can be expressed as a function of $|\mathcal{T}|$:

Corollary 3.1 *Let $(G=(V,E),\mathcal{T})$ be a weighted multiple-source unsplittable flow problem, and $\alpha^f(\mathcal{T})$ be the value of an optimum fractional routing f. Algorithm* MAX_ROUTING *finds in polynomial time an unsplittable flow of value*

$$\Omega(\max\{\frac{\alpha^f(\mathcal{T})}{\sqrt{|E|+|\mathcal{T}|}}, \frac{(\alpha^f(\mathcal{T}))^2}{(|E|+|\mathcal{T}|)\log\log|E|}\}).$$

If the maximum demand is bounded by $\zeta<1$, at Step 2 of MAX_ROUTING it is enough to scale by $\zeta\beta\log|E|$. We can then use Corollary 2.1 instead of Theorem 2.4. Generalizing further, we can alllow some congestion $B>1$, which has already been taken into account in the aforementioned corollary. The following theorem expresses the improvements obtained in this general case on Theorems 3.2 and 3.3.

Theorem 3.4 *Let $(G=(V,E),\mathcal{T})$ be a weighted multiple-source unsplittable flow problem, and $\alpha^f(\mathcal{T})$ be the value of an optimum fractional routing f of dilation d. Let $\zeta<1$, be the value of the maximum demand and assume a congestion of $B>1$ is allowed in the solution. There is a polynomial-time algorithm, which finds an unsplittable flow of value*

$$\Omega(\max\{\frac{\alpha^f(\mathcal{T})}{\zeta\log|E|(|E|)^{1/(\lfloor(B/\zeta)\rfloor+1)}}, \frac{\alpha^f(\mathcal{T})}{\zeta\log|E|}(\frac{\alpha^f(\mathcal{T})}{(|E|\zeta\log|E|\log\log|E|)})^{1/\lfloor(B/\zeta)\rfloor}, \frac{\alpha^f(\mathcal{T})}{d^{1/(\lfloor(B/\zeta)\rfloor+1)}}\}).$$

3.2 Weighted vertex-disjoint paths

A relaxation of the weighted vertex-disjoint path problem is integral multicommodity flow, in which every commodity has an associated demand of 1. We can express vertex-disjoint paths as an integer program $\mathcal{I}$ that has the same constraints as multicommodity flow together with additional "bandwidth" constraints on the vertices. Let f_{ij}^k denote in $\mathcal{I}$ the flow through edge $(i,j)\in E$ corresponding to the k-th commodity, i.e. to the terminal pair (s_k,t_k).

$$\text{maximize} \sum_{k=1}^{K} w_k(\sum_{(i,t_k)\in E} f_{it_k}^k - \sum_{(t_k,i)\in E} f_{t_k i}^k) \quad \text{subject to}$$

$$\begin{array}{llll}
\displaystyle\sum_{(j,i)\in E} f_{ji}^k - \sum_{(i,j)\in E} f_{ij}^k & & \begin{cases} \leq 1 & \text{if } i=t_k \\ \geq -1 & \text{if } i=s_k \\ 0 & \text{otherwise} \end{cases} & \forall i\in V, 1\leq k\leq K \\
\displaystyle\sum_{k=1}^{K}\sum_{(j,i)\in E} f_{ji}^k & \leq & 1, & \forall i\in V \\
\displaystyle\sum_{k=1}^{K}\sum_{(i,j)\in E} f_{ij}^k & \leq & 1 & \forall i\in V \\
f_{ij}^k & \geq & 0 & \forall (i,j)\in E, 1\leq k\leq K
\end{array}$$

ALGORITHM PATH_PACKING($G = (V, E), \mathcal{T}$)

Step 1. Formulate the linear relaxation $\mathcal{LP}$ and find an optimal solution f to it. Using flow decomposition express f as a set of paths $P_1, P_2, \ldots, P_\nu$, each connecting a pair of terminals and carrying $z_i \leq 1$, $1 \leq i \leq \nu$, units of flow.

Step 2. Construct a $(0,1)$-PIP $\mathcal{P} = (A, b, c)$ as follows. A is a $|V| \times \nu$ matrix; A_{ij} is 1 if path P_j includes vertex i and 0 otherwise. b is a vector of ones. c_j is equal to w_k such that path P_j connects terminals s_k and t_k.

Step 3. Find an integral solution g to $\mathcal{P}$ by using the algorithms of Theorem 1.1 and output the corresponding set of paths $P(g)$.

Figure 3: Algorithm PATH_PACKING.

Let $\mathcal{LP}$ be the linear relaxation of $\mathcal{I}$. The optimal solution f to $\mathcal{LP}$ consists of a set of *fractional flow paths*. Our algorithm relies on the observation that f gives rise (and at the same time is a fractional solution) to a PIP. The particular PIP models a 1-matching problem on a hypergraph H with vertex set V and a hyperedge (subset of V) for every path in the fractional solution. In other words, the paths in f may be viewed as sets of vertices *without any regard for the flow through the edges of G*. We give the full algorithm in Figure 3.

Theorem 3.5 *Let $(G = (V, E), \mathcal{T})$ be a weighted vertex-disjoint paths problem, and $\alpha^f(\mathcal{T})$ be the value of an optimum solution to the linear relaxation $\mathcal{LP}$. Let d be the dilation of the solution. Algorithm* PATH_PACKING *finds in polynomial time a solution of value*

$$\Omega\left(\max\left\{\frac{\alpha^f(\mathcal{T})}{\sqrt{|V|}}, \frac{(\alpha^f(\mathcal{T}))^2}{|V|}, \frac{\alpha^f(\mathcal{T})}{d}\right\}\right).$$

Proof. We show first that $P(g)$ is a feasible solution to the problem. Clearly the constraints of the packing program $\mathcal{P}$, constructed at Step 2 ensure that the paths in $P(g)$ are vertex disjoint. This excludes the possibility that more than one (s_k, t_k)-path is present in $P(g)$ for some k. The optimal value of the linear relaxation of $\mathcal{P}$ is at least y_* since setting x_j equal to z_j, $1 \leq j \leq \nu$, yields a feasible fractional solution to $\mathcal{P}$. The claim on the value of g follows from the bounds in Theorem 1.1. ∎

3.3 An Application to Maximum Independent Set

In this section we show how a packing formulation leads to an $O(\sqrt{|V|})$ approximation for the following problem: find a maximum weight independent set in the square of the graph $G = (V, E)$.

Theorem 3.6 *Given an undirected graph $G = (V, E)$ and $c \in [0,1]^{|V|}$ a weight vector on the vertices, there exists a polynomial-time algorithm, which outputs an independent set in the square $G^2 = (V, E^2)$ of G, of weight*

$$\Omega(\max\{y_*/\sqrt{|V|}, (y_*)^2/|V|, y_*/\Delta\}).$$

Here y_* denotes the optimum of a fractional relaxation and Δ is the maximum vertex degree in G.

Proof. Assume an ordering $v_1, \ldots, v_{|V|}$ on the vertices in V. We form a $(0,1)$-PIP $\mathcal{P} = (A, b, c)$ as follows. A is a $|V| \times |V|$ matrix. Row i corresponds to the neighborhood of v_i. $A_{ij} = 1$ if and only if $j = i$ or $\{v_i, v_j\} \in E$. All the b-entries are set to 1.

We now show that any feasible integral solution $\bar{x}$ to $\mathcal{P}$ is an independent set in G^2. Let i, j be such that $\bar{x}_i = \bar{x}_j = 1$. If edge $\{v_i, v_j\}$ belongs to E^2 either $\{v_i, v_j\} \in E$ or v_i and v_j share a common neighbor v_k in G. In the first case the i-th constraint of $\mathcal{P}$ is violated; in the second case the k-th constraint of $\mathcal{P}$ is violated. A contradiction. Conversely, consider an independent set I of G^2. We will show that its characteristic vector x_I is a feasible solution of $\mathcal{P}$. Assume this is not the case and that the inner product of x_I with the i-th row of A exceeds 1. Then there exist two vertices v_k, v_l that are adjacent to v_i (or one of them identical to v_i) such that both v_k, v_l belong to I. But edge $\{v_k, v_l\}$ must belong to E^2 by the definition of G^2, a contradiction.

Let y_* be the optimum of the linear relaxation of $\mathcal{P}$. Notice that the maximum number of non-zero entries in a column is $\Delta + 1$. The bounds of the theorem follow by Theorem 1.1. ∎

A hardness of approximation result for the problem of finding a maximum independent set in the k-th power of a graph follows.

Theorem 3.7 *For the problem of finding a maximum independent set in the k-th power $G^k = (V, E^k)$ of a graph $G = (V, E)$ for any fixed integer $k > 0$, there is no ρ-approximation with $\rho = (\frac{|V|}{k-1})^{1/2-\varepsilon}$, for any fixed $\varepsilon > 0$, unless $NP \subseteq ZPP$.*

Proof. Lin and Skiena [30] give an approximation-preserving reduction from finding an independent set in an arbitrary graph $G_0 = (V_0, E_0)$ to finding an independent set in the k-th power $(G')^k$ of a graph $G' = (V', E')$ such that $|V'| \geq (k-1)|E_0|$. By Håstad's result [17] there is no $|V_0|^{1-\varepsilon}$-approximation algorithm, $\varepsilon > 0$, for maximum independent set in G_0 unless $NP \subseteq ZPP$. This implies that no $|E_0|^{1/2-\varepsilon/2}$-approximation algorithm exists unless $NP \subseteq ZPP$.

Therefore no $(|V'|/(k-1))^{1/2-\varepsilon/2}$ approximation exists for maximum independent set in $(G')^k$ under the same complexity-theoretic assumption. ∎

ALGORITHM GREEDY_PATH$(G, \mathcal{T})$

Step 1. Set $\mathcal{A}$ to $\emptyset$.

Step 2. Let (s_*, t_*) be the commodity in $\mathcal{T}$ such that the shortest path P_* in G from s_* to t_* has minimum length. If no such path exists halt and output $\mathcal{A}$.

Step 3. Add P_* to $\mathcal{A}$ and remove the edges of P_* from G. Remove (s_*, t_*) from $\mathcal{T}$. Goto Step 2.

Figure 4: Algorithm GREEDY_PATH.

4 Greedy Algorithms for Disjoint Paths

In this section we turn to the routing approach for the unweighted edge- and vertex-disjoint path problems. Our algorithm is motivated by the blocking flow method for the s-t maximum flow problem ([9], see also [11]). The latter method proceeds by iterations on the current residual graph. On every iteration a flow is found, which greedily saturates at least one edge on (i.e. blocks) all shortest paths from the source to the sink. Our initial motivation came from the following question: what amount of the residual flow has survived if one is not allowed to ever reroute the blocking flow found at a given iteration? GREEDY_PATH, given in Figure 4 does not use rerouting. The underlying idea is that if one keeps routing commodities along sufficiently short paths the final number of commodities routed is lowerbounded with respect to the optimum.

We begin by lowerbounding the approximation ratio of the algorithm. The proof will provide the intuition behind GREEDY_PATH. We say that a path P_1 *hits* a path P_2, if P_1 and P_2 share an edge.

Theorem 4.1 *Algorithm* GREEDY_PATH *runs in polynomial time and outputs a solution to an edge-disjoint path problem* $(G = (V, E), \mathcal{T})$ *of size at least* $1/(\sqrt{|E_o|} + 1)$ *times the optimum, where* $E_o \subseteq E$ *is the set of edges used by the paths in an optimal solution.*

Proof. Let t be the total number of iterations of GREEDY_PATH and $\mathcal{A}_i$ be the set $\mathcal{A}$, G_i the updated graph and $\mathcal{T}_i$ the current set $\mathcal{T}$ at the end of the i-th iteration. Let $\mathcal{O}$ be an optimal set of paths. We define the set $\mathcal{O} \ominus \mathcal{A}$ as the paths in $\mathcal{O}$ that correspond to commodities not routed in $\mathcal{A}$. We claim that after every iteration i the path P_i in $\mathcal{A}_i - \mathcal{A}_{i-1}$ hits at most $\sqrt{|E_o|}$ paths in $\mathcal{O} \ominus \mathcal{A}_i$ that are not hit by a path in $\mathcal{A}_{i-1}$. Then, since upon termination of GREEDY_PATH all paths in

$\mathcal{O} \ominus \mathcal{A}_t$ must hit some path in $\mathcal{A}_t$ we have that

$$|\mathcal{O} \ominus \mathcal{A}_t| \leq \sqrt{|E_o|}|\mathcal{A}_t|.$$

But $|\mathcal{O}| - |\mathcal{A}_t| \leq |\mathcal{O} \ominus \mathcal{A}_t|$ and the theorem follows.

We prove our claim by induction on the number of iterations. After the first iteration assume that the path P_1 in $\mathcal{A}_1$ hits k paths in $\mathcal{O} \ominus \mathcal{A}_1$. Note that an edge of P_1 can be used to hit only at most one other path in $\mathcal{O}$ so P_1 has length at least k. Since P_1 is the shortest path among all paths between terminal pairs of $\mathcal{T}$ in G, all k paths in $\mathcal{O} \ominus \mathcal{A}_1$ have length at least k. But $k^2 \leq |E_o|$, so $k \leq \sqrt{|E_o|}$.

For the inductive step assume the claim holds for $i \leq j$. Let P_{j+1} hit k paths in $\mathcal{O} \ominus \mathcal{A}_{j+1}$ that are not hit by a path in $\mathcal{A}_j$. The length of P_{j+1} is at least k. Assume that a path P' of these k paths hit by P_{j+1} has length less than k. Then the shortest path between the endpoints of P' in G_j has length less than k since no edge belonging to P' has been deleted so far. Moreover, the endpoints of P' are in $\mathcal{T}_j$ since P' belongs to $\mathcal{O} \ominus \mathcal{A}_{j+1}$. Thus P' would have been selected instead of P_{j+1} during the $(j+1)$-th iteration, a contradiction. We conclude that all k paths in $\mathcal{O} \ominus \mathcal{A}_{j+1}$ hit by P_{j+1} have length at least k and since $k^2 \leq |E_o|$ the inductive step is complete. ∎

Algorithm Greedy_Path gives a solution to vertex-disjoint paths with the following modification at Step 3: remove the vertices of P_* from G. Call the resulting algorithm Greedy_VPath. The analysis above can be easily modified to yield the following.

Theorem 4.2 *Algorithm* Greedy_VPath *runs in polynomial time and outputs a solution to a vertex-disjoint paths problem* $(G = (V, E), \mathcal{T})$ *of size at least* $1/(\sqrt{|V_o|} + 1)$ *times the optimum, where* $V_o \subseteq V$ *is the set of vertices used by the paths in an optimal solution.*

We now show improved bounds on the size of realizable set output by Greedy_Path. Recall that OPT denotes the optimal size of a realizable set.

Theorem 4.3 *Algorithm* Greedy_Path *outputs a solution to an edge-disjoint path problem* $(G = (V, E), \mathcal{T})$ *of size* $\Omega(OPT^2/|E_o|)$, *where* $E_o \subseteq E$ *is the set of edges used by the paths in an optimal solution.*

Proof. We maintain the notation from the proof of Theorem 4.1. Let k_i, $1 \leq i \leq t$ be the number of paths in $\mathcal{O} \ominus \mathcal{A}_i$ that are not hit by a path in $\mathcal{A}_{i-1}$ and are hit by the path P_i in $\mathcal{A}_i - \mathcal{A}_{i-1}$. It was established in the proof of Theorem 4.1 that all these k_i paths have length at least k_i. Furthermore all paths in $\mathcal{O}$ are edge-disjoint with total number of edges $|E_o|$. Therefore

$$\sum_{i=1}^{t} k_i^2 \leq |E_o|.$$

Applying the Cauchy-Schwartz inequality on the left-hand side we obtain that $(\sum_{i=1}^{t} k_i)^2/t \leq |E_o|$. But $\sum_{i=1}^{t} k_i = |\mathcal{O} \ominus \mathcal{A}_t|$ thus we obtain

$$\frac{|\mathcal{O} \ominus \mathcal{A}_t|^2}{t} \leq |E_o|.$$

Without loss of generality we can assume that $|\mathcal{A}_t| \leq |\mathcal{O}|/3$, since otherwise GREEDY_PATH obtains a constant-factor approximation. It follows that $t = \Omega(|\mathcal{O}|^2/|E_o|) = \Omega(OPT^2/|E_o|)$. ∎

Similarly we obtain the analoguous result for GREEDY_VPATH.

Theorem 4.4 *Algorithm* GREEDY_VPATH *outputs a solution to a vertex-disjoint path problem* $(G = (V, E), \mathcal{T})$ *of size* $\Omega(OPT^2/|V_o|)$*, where* $V_o \subseteq V$ *is the set of vertices used by the paths in an optimal solution.*

It is straightforward to obtain the ensuing corollaries from Theorems 4.3 and 4.4.

Corollary 4.1 *Algorithm* GREEDY_PATH *outputs a solution to an edge-disjoint path problem* $(G = (V, E), \mathcal{T})$ *of size* $\Omega(OPT/d_o)$*, where* d_o *is the average length of the paths in an optimal solution.*

Corollary 4.2 *Algorithm* GREEDY_VPATH *outputs a solution to a vertex-disjoint path problem* $(G = (V, E), \mathcal{T})$ *of size* $\Omega(OPT/d_o)$*, where* d_o *is the average length of the paths in an optimal solution.*

5 Concluding remarks

In this paper we introduced the study of approximation algorithms for column-restricted Packing Integer Programs. We used them to obtain the first approximation algorithm for the general unsplittable flow problem. Some further applications of PIPs to the design of approximation algorithms were also given. PIPs can model a comprehensive category of optimization problems; the use of PIP-solvers as subroutines can result in elegant and powerful approximation algorithms.

In addition to the packing approach, we gave a greedy algorithm with competitive worst-case performance that should be significantly simpler to implement. The work of Young on oblivious rounding [48] showed that for a variety of packing and covering problems, greedy algorithms can provide good approximations without solving first the fractional relaxations. Therefore there seems to be a point of convergence between the rounding and routing approaches. As noted in [25] it would be interesting to extend the greedy algorithm into two directions: (i) use it to round the solution to a fractional relaxation and (ii) modify it to handle general demands and capacities. After the first publication of our results [26], progress was reported on these two questions by Young [49] and Guruswami et al. [15]. In [15] it was also shown that edge-disjoint paths on directed graphs cannot be approximated within $O(|E|^{1/2-\varepsilon})$ for any $\varepsilon > 0$, unless $P = NP$. For vertex-disjoint paths Guruswami et al. showed that on undirected graphs the optimum cannot be approximated within $O(|V|^{1/2-\varepsilon})$ for any $\varepsilon > 0$, unless $P = NP$. These results partially settle an important open question and demonstrate that the approximation ratios we achieve for unsplittable flow and vertex-disjoint paths are asymptotically near-optimal.

Acknowledgments. We wish to thank Alok Baveja, Jon Kleinberg, Aravind Srinivasan and Neal Young for valuable discussions. Thanks to Alok and Aravind for pointing out the relevance of

Theorem 2.2. Thanks also to Javed Aslam for helpful discussions. Finally we thank the anonymous referees for useful suggestions.

References

[1] R. Aharoni, P. Erdős, and N. Linial. Optima of dual integer linear programs. *Combinatorica*, 8:13–20, 1988.

[2] S. Arora and C. Lund. Hardness of approximations. In D. S. Hochbaum, editor, *Approximation Algorithms for NP-hard problems*, pages 399–446. PWS, Boston, 1997.

[3] Y. Aumann and Y. Rabani. Improved bounds for all-optical routing. In *Proceedings of the 6th ACM-SIAM Symposium on Discrete Algorithms*, pages 567–576, 1995.

[4] A. Baveja and A. Srinivasan. Approximating low-congestion routing and column-restricted packing problems. *Information Processing Letters*, 74:19–25, 2000.

[5] A. Baveja and A. Srinivasan. Approximation algorithms for disjoint paths and related routing and packing problems. *Mathematics of Operations Research*, 25:255–280, 2000.

[6] A. Z. Broder, A. M. Frieze, and E. Upfal. Static and dynamic path selection on expander graphs: a random walk approach. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 531–539, 1997.

[7] H. Chernoff. A measure of the asymptotic efficiency for tests of a hypothesis based on sum of observations. *Ann. Math. Stat.*, 23:493–509, 1952.

[8] C. Cooper. The thresold of Hamilton cycles in the square of a random graph. *Random Structures and Algorithms*, 5:25–31, 1994.

[9] E. A. Dinitz. Algorithm for solution of a problem of maximum flow in networks with power estimation. *Soviet Math. Dokl.*, 11:1277–1280, 1970.

[10] P. Erdős and J. L. Selfridge. On a combinatorial game. *Journal of Combinatorial Theory A*, 14:298–301, 1973.

[11] S. Even and R. E. Tarjan. Network flow and testing graph connectivity. *SIAM Journal on Computing*, 4:507–518, 1975.

[12] H. Fleischner. The square of every two-connected graph is Hamiltonian. *Journal of Combinatorial Theory B*, 16:29–34, 1974.

[13] A. 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*, pages 49–100. Springer-Verlag, Berlin, 1990.

[14] N. Garg, V. Vazirani, and M. Yannakakis. Primal-dual approximation algorithms for integral flow and multicut in trees. *Algorithmica*, 18:3–20, 1997.

[15] V. Guruswami, S. Khanna, R. Rajaraman, B. Sheperd, and M. Yannakakis. Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, 1999.

[16] M. Halldórsson, J. Kratochvíl, and J. A. Telle. Independent sets with domination constraints. In *Proceedings of the 25th ICALP*, July 1998. Full version in Discrete Applied Mathematics, 99(1-3), 39-54, 1999.

[17] J. Håstad. Clique is hard to approximate within $n^{1-\varepsilon}$. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, pages 627–636, 1996. To appear in Acta Mathematica.

[18] R. M. Karp. On the computational complexity of combinatorial problems. *Networks*, 5:45–68, 1975.

[19] R. M. Karp, F. T. Leighton, R. L. Rivest, C. D. Thompson, U. V. Vazirani, and V. V. Vazirani. Global wire routing in two-dimensional arrays. *Algorithmica*, 2:113–129, 1987.

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

[21] J. M. Kleinberg. Single-source unsplittable flow. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, pages 68–77, October 1996.

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

[23] J. M. Kleinberg and É. Tardos. Disjoint paths in densely-embedded graphs. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, pages 52–61, 1995.

[24] S. G. Kolliopoulos. Approximating covering integer programs with multiset constraints. Submitted for journal publication, September 2000.

[25] S. G. Kolliopoulos. *Exact and Approximation Algorithms for Network Flow and Disjoint-Path Problems.* PhD thesis, Dartmouth College, Hanover, NH, August 1998.

[26] S. G. Kolliopoulos and C. Stein. Approximating disjoint-path problems using greedy algorithms and packing integer programs. In R. E. Bixby, E. A. Boyd, and R. Z. Rios-Mercado, editors, *Proceedings of the 6th Conference on Integer Programming and Combinatorial Optimization*, volume 1412 of *Lecture Notes in Computer Science*, pages 153–168. Springer-Verlag, June 1998.

[27] S. G. Kolliopoulos and C. Stein. Approximation algorithms for single-source unsplittable flow. 2001. To appear in SIAM J. Comp. Preliminary version in Proc. FOCS 1997.

[28] F. T. Leighton and S. B. Rao. Circuit switching: a multi-commodity flow approach. In *Workshop on Randomized Parallel Computing*, 1996.

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

[30] Y.-L. Lin and S. E. Skiena. Algorithms for square roots of graphs. *SIAM Journal on Discrete Mathematics*, 8(1):99–118, February 1995.

[31] L. Lovász. On the ratio of optimal and fractional covers. *Discrete Mathematics*, 13:383–390, 1975.

[32] P. Martin and D. B. Shmoys. A new approach to computing optimal schedules for the job-shop scheduling problem. In *Proceedings of the 5th Conference on Integer Programming and Combinatorial Optimization*, pages 389–403, 1996.

[33] D. Peleg and E. Upfal. Disjoint paths on expander graphs. *Combinatorica*, 9:289–313, 1989.

[34] S. Plotkin. Competitive routing of virtual circuits in ATM networks. *IEEE J. Selected Areas in Comm.*, pages 1128–1136, 1995. Special issue on Advances in the Fundamentals of Networking.

[35] S. Plotkin, D. B. Shmoys, and É. Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research*, 20:257–301, 1995.

[36] P. Raghavan. Probabilistic construction of deterministic algorithms: approximating packing integer programs. *Journal of Computer and System Sciences*, 37:130–143, 1988.

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

[38] N. Robertson and P. D. Seymour. Outline of a disjoint paths algorithm. In B. Korte, L. Lovász, H. J. Prömel, and A. Schrijver, editors, *Paths, Flows and VLSI-Layout*. Springer Verlag, Berlin, 1990.

[39] A. Schrijver. Homotopic routing methods. In B. Korte, L. Lovász, H. J. Prömel, and A. Schrijver, editors, *Paths, Flows and VLSI-Layout*. Springer Verlag, Berlin, 1990.

[40] D. B. Shmoys, C. Stein, and J. Wein. Improved approximation algorithms for shop scheduling problems. *SIAM Journal on Computing*, 23(3):617–632, June 1994.

[41] J. Spencer. *Ten Lectures on the Probabilistic Method*. SIAM, Philadelphia, 1987.

[42] A. Srinivasan. An extension of the Lovász Local Lemma and its applications to integer programming. In *Proceedings of the 7th ACM-SIAM Symposium on Discrete Algorithms*, pages 6–15, 1996.

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

[44] A. Srinivasan. Improved approximations guarantees for packing and covering integer programs. *SIAM Journal on Computing*, 29:648–670, 1999. Preliminary version in Proc. STOC 95.

[45] A. Srinivasan and C.-P. Teo. A constant-factor approximation algorithm for packet routing and balancing local vs. global criteria. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 636–643, 1997.

[46] C. Stein. *Approximation algorithms for multicommodity flow and shop scheduling problems*. PhD thesis, MIT, Cambridge, MA, August 1992. Also appears as MIT/LCS/TR-550.

[47] P. Underground. On graphs with Hamiltonian squares. *Discrete Mathematics*, 21:323, 1978.

[48] N. E. Young. Randomized rounding without solving the linear program. In *Proceedings of the 6th ACM-SIAM Symposium on Discrete Algorithms*, pages 170–178, 1995.

[49] Neal E. Young. Personal communication, 1998.