On Column-restricted and Priority Covering Integer Programs *

Deeparnab Chakrabarty, Elyot Grant, and Jochen Könemann

Department of Combinatorics and Optimization
University of Waterloo, Waterloo, ON, Canada N2L 3G1

Abstract. In a column-restricted covering integer program (CCIP), all the non-zero entries of any column of the constraint matrix are equal. Such programs capture capacitated versions of covering problems. In this paper, we study the approximability of CCIPs, in particular, their relation to the integrality gaps of the underlying 0,1-CIP.

If the underlying 0,1-CIP has an integrality gap $O(\gamma)$, and assuming that the integrality gap of the *priority version* of the 0,1-CIP is $O(\omega)$, we give a factor $O(\gamma + \omega)$ approximation algorithm for the CCIP. Priority versions of 0,1-CIPs (PCIPs) naturally capture *quality of service* type constraints in a covering problem.

We investigate priority versions of the line (PLC) and the (rooted) tree cover (PTC) problems. Apart from being natural objects to study, these problems fall in a class of fundamental geometric covering problems. We bound the integrality of certain classes of this PCIP by a constant. Algorithmically, we give a polytime exact algorithm for PLC, show that the PTC problem is APX-hard, and give a factor 2-approximation algorithm for it.

1 Introduction

In a *0,1-covering integer program* (0,1-CIP, in short), we are given a constraint matrix $A \in \{0,1\}^{m \times n}$, demands $b \in \mathbb{Z}_+^m$, non-negative costs $c \in \mathbb{Z}_+^n$, and upper bounds $d \in \mathbb{Z}_+^n$, and the goal is to solve the following integer linear program (which we denote by $\texttt{Cov}(A, b, c, d)$).

$$\min\{c^T x \,:\, Ax \geq b, 0 \leq x \leq d, x \text{ integer}\}.$$

Problems that can be expressed as 0,1-CIPs are essentially equivalent to set multi-cover problems, where sets correspond to columns and elements correspond to rows. This directly implies that 0,1-CIPs are rather well understood in terms of approximability: the class admits efficient $O(\log n)$ approximation algorithms and this is best possible unless NP = P. Nevertheless, in many cases one can get better approximations by exploiting the structure of matrix A. For example, it is well known that whenever A is *totally unimodular* (TU)(e.g., see [18]), the

* Supported by NSERC grant no. 288340 and by an Early Research Award. Emails: deeparnab@gmail.com, elyot@uwaterloo.ca, jochen@uwaterloo.ca

canonical LP relaxation of a 0,1-CIP is integral; hence, the existence of efficient algorithms for solving linear programs immediately yields fast exact algorithms for such 0,1-CIPs as well.

While a number of general techniques have been developed for obtaining improved approximation algorithms for structured $0,1$-CIPs, not much is known for structured non-$0,1$ CIP instances. In this paper, we attempt to mitigate this problem, by studying the class of *column-restricted covering integer programs* (CCIPs), where all the non-zero entries of any column of the constraint matrix are equal. Such CIPs arise naturally out of $0,1$-CIPs, and the main focus of this paper is to understand how the structure of the underlying 0,1-CIP can be used to derive improved approximation algorithms for CCIPs.

Column-Restricted Covering IPs (CCIPs): Given a 0,1-covering problem $\texttt{Cov}(A, b, c, d)$ and a supply vector $s \in \mathbb{Z}^n_+$, the corresponding CCIP is obtained as follows. Let $A[s]$ be the matrix obtained by replacing all the 1's in the jth column by s_j; that is, $A[s]_{ij} = A_{ij}s_j$ for all $1 \leq i \leq m, 1 \leq j \leq n$. The column-restricted covering problem is given by the following integer program.

$$\min\{c^T x \,:\, A[s]x \geq b, 0 \leq x \leq d, x \text{ integer}\}. \qquad (\texttt{Cov}(A[s], b, c, d))$$

CCIPs naturally capture *capacitated* versions of 0,1-covering problems. To illustrate this we use the following 0,1-covering problem called the tree covering problem. The input is a tree $T = (V, E)$ rooted at a vertex $r \in V$, a set of *segments* $\mathcal{S} \subseteq \{(u, v) : u \text{ is a child of } v\}$, non-negative costs c_j for all $j \in \mathcal{S}$, and demands $b_e \in \mathbb{Z}_+$ for all $e \in E$. An edge e is contained in a segment $j = (u, v)$ if e lies on the unique u, v-path in T. The goal is to find a minimum-cost subset C of segments such that each edge $e \in E$ is contained in at least b_e segments of C. When T is just a line, we call the above problem, the *line cover* (LC) problem. In this example, the constraint matrix A has a row for each edge of the tree and a column for each segment in $\mathcal{S}$. It is not too hard to show that this matrix is TU and thus these can be solved exactly in polynomial time.

In the above tree cover problem, suppose each segment $j \in \mathcal{S}$ also has a capacity supply s_j associated with it, and call an edge e covered by a collection of segments C iff the total supply of the segments containing e exceeds the demand of e. The problem of finding the minimum cost subset of segments covering every edge is precisely the column-restricted tree cover problem. The column-restricted line cover problem encodes the minimum knapsack problem and is thus NP-hard.

For general CIPs, the best known approximation algorithm, due to Kolliopoulos and Young [15], has a performance guarantee of $O(1 + \log \alpha)$, where α, called the *dilation* of the instance, denotes the maximum number of non-zero entries in any column of the constraint matrix. Nothing better is known for the special case of CCIPs unless one aims for *bicriteria* results where solutions violate the upper bound constraints $x \leq d$ (see Section 1.1 for more details).

In this paper, our main aim is to understand how the approximability of a given CCIP instance is determined by the structure of the underlying $0,1$-CIP. In

particular, if a $0,1$-CIP has a constant integrality gap, under what circumstances can one get a constant factor approximation for the corresponding CCIP? We make some steps toward finding an answer to this question.

In our main result, we show that there is a constant factor approximation algorithm for CCIP if *two* induced $0,1$-CIPs have constant integrality gap. The first is the underlying original 0,1-CIP. The second is a *priority* version of the 0,1-CIP (PCIP, in short), whose constraint matrix is derived from that of the 0,1-CIP as follows.

Priority versions of Covering IPs (PCIPs): Given a 0,1-covering problem $\texttt{Cov}(A, b, c, d)$, a priority supply vector $s \in \mathbb{Z}^n_+$, and a priority demand vector $\pi \in \mathbb{Z}^m_+$, the corresponding PCIP is as follows. Define $A[s, \pi]$ to be the following 0,1 matrix

$$A[s,\pi]_{ij} = \begin{cases} 1 & : \quad A_{ij} = 1 \text{ and } s_j \geq \pi_i \\ 0 & : \quad \text{otherwise,} \end{cases} \tag{1}$$

Thus, a column j covers row i, only if its priority supply is higher than the priority demand of row i. The priority covering problem is now as follows.

$$\min\{c^T x \,:\, A[s,\pi]x \geq \mathbb{1}, 0 \leq x \leq d, x \text{ integer}\}. \qquad (\texttt{Cov}(A[s,\pi], \mathbb{1}, c))$$

We believe that priority covering problems are interesting in their own right, and they arise quite naturally in covering applications where one wants to model *quality of service* (QoS) or priority restrictions. For instance, in the tree cover problem defined above, suppose each segment j has a *quality of service* (QoS) or priority supply s_j associated with it and suppose each edge e has a QoS or priority demand π_e associated with it. We say that a segment j covers e iff j contains e *and* the priority supply of j exceeds the priority demand of e. The goal is to find a minimum cost subset of segments that covers every edge. This is the priority tree cover problem.

Besides being a natural covering problem to study, we show that the priority tree cover problem is a special case of a classical geometric covering problem: that of finding a minimum cost cover of points by axis-parallel rectangles in 3 dimensions. Finding a constant factor approximation algorithm for this problem, even when the rectangles have uniform cost, is a long standing open problem.

We show that although the tree cover is polynomial time solvable, the priority tree cover problem is APX-hard. We complement this with a factor 2 approximation for the problem. Furthermore, we present constant upper bounds for the integrality gap of this PCIP in a number of special cases, implying constant upper bounds on the corresponding CCIPs in these special cases. We refer the reader to Section 1.2 for a formal statement of our results, which we give after summarizing works related to our paper.

1.1 Related work

There is a rich and long line of work ([9, 11, 17, 19, 20]) on approximation algorithms for CIPs, of which we state the most relevant to our work. Assum-

ing no upper bounds on the variables, Srinivasan [19] gave a $O(1 + \log \alpha)$-approximation to the problem (where α is the dilation as before). Later on, Kolliopoulos and Young [15] obtained the same approximation factor, respecting the upper bounds. However, these algorithms didn't give any better results when special structure of the constraint matrix was known. On the hardness side, Trevisan [21] showed that it is NP-hard to obtain a $(\log \alpha - O(\log \log \alpha))$-approximation algorithm even for 0,1-CIPs.

The most relevant work to this paper is that of Kolliopoulos [12]. The author studies CCIPs which satisfy a rather strong assumption, called the *no bottleneck assumption*, that the supply of any column is smaller than the demand of any row. Kolliopoulos [12] shows that if one is allowed to violate the upper bounds by a multiplicative constant, then the integrality gap of the CCIP is within a constant factor of that of the original 0,1-CIP[1]. As the author notes such a violation is necessary; otherwise the CCIP has unbounded integrality gap. If one is not allowed to violated upper bounds, nothing better than the result of [15] is known for the special case of CCIPs.

Our work on CCIPs parallels a large body of work on column-restricted *packing* integer programs (CPIPs). Assuming the *no-bottleneck assumption*, Kolliopoulos and Stein [14] show that CPIPs can be approximated asymptotically as well as the corresponding 0,1-PIPs. Chekuri et al. [7] subsequently improve the constants in the result from [14]. These results imply constant factor approximations for the column-restricted tree *packing* problem under the no-bottleneck assumption. Without the no-bottleneck assumption, however, only polylogarithmic approximation is known for the problem [6].

The only work on priority versions of covering problems that we are aware of is due to Charikar, Naor and Schieber [5] who studied the priority Steiner tree and forest problems in the context of QoS management in a network multicasting application. Charikar et al. present a $O(\log n)$-approximation algorithm for the problem, and Chuzhoy et al. [8] later show that no efficient $o(\log \log n)$ approximation algorithm can exist unless NP $\subseteq$ DTIME($n^{\log \log \log n}$) (n is the number of vertices).

To the best of our knowledge, the column-restricted or priority versions of the line and tree cover problem have not been studied. The best known approximation algorithm known for both is the $O(\log n)$ factor implied by the results of [15] stated above. However, upon completion of our work, Nitish Korula [16] pointed out to us that a 4-approximation for column-restricted line cover is implicit in a result of Bar-Noy et al. [2]. We remark that their algorithm is not LP-based, although our general result on CCIPs is.

1.2 Technical Contributions and Formal Statement of Results

Given a 0,1-CIP $\texttt{Cov}(A, b, c, d)$, we obtain its *canonical LP relaxation* by removing the integrality constraint. The *integrality gap* of the CIP is defined as the

[1] Such a result is implicit in the paper; the author only states a $O(\log \alpha)$ integrality gap.

supremum of the ratio of optimal IP value to optimal LP value, taken over all non-negative integral vectors b, c, and d. The integrality gap of an IP captures how much the integrality constraint affects the optimum, and is an indicator of the *strength* of a linear programming formulation.

CCIPs: Suppose the CCIP is $\texttt{Cov}(A[s], b, c, d)$. We make the following two assumptions about the integrality gaps of the 0,1 covering programs, both the original 0,1-CIP and the priority version of the 0,1-CIP.

Assumption 1 *The integrality gap of the original 0,1-CIP is $\gamma \geq 1$. Specifically, for any non-negative integral vectors b, c, and d, if the canonical LP relaxation to the CIP has a fractional solution x, then one can find in polynomial time an integral feasible solution to the CIP of cost at most $\gamma \cdot c^T x$. We stress here that the entries of b, c, d could be 0 as well as ∞.*

Assumption 2 *The integrality gap of the PCIP is $\omega \geq 1$. Specifically, for any non-negative integral vectors s, π, c, if the canonical LP relaxation to the PCIP has a fractional solution x, then one can find in polynomial time, an integral feasible solution to the PCIP of cost at most $\omega \cdot c^T x$.*

We give an LP-based approximation algorithm for solving CCIPs. Since the canonical LP relaxation of a CCIP can have unbounded integrality gap, we strengthen it by adding a set of valid constraints called the *knapsack cover constraints*. We show that the integrality gap of this strengthened LP is $O(\gamma+\omega)$, and can be used to give a polynomial time approximation algorithm.

Theorem 1. *Under Assumptions 1 and 2, there is a $(24\gamma + 8\omega)$-approximation algorithm for column-restricted CIPs.*

Knapsack cover constraints to strengthen LP relaxations were introduced in [1, 10, 22]; Carr et al. [3] were the first to employ them in the design approximation algorithms. The paper of Kolliopoulos and Young [15] also use these to get their result on general CIPs.

The main technique in the design of algorithms for column-restricted problems is *grouping-and-scaling* developed by Kolliopoulos and Stein [13, 14] for packing problems, and later used by Kolliopoulos [12] in the covering context. In this technique, the *columns* of the matrix are divided into groups of 'close-by' supply values; in a single group, the supply values are then scaled to be the same; for a single group, the integrality gap of the original 0,1-CIP is invoked to get an integral solution for that group; the final solution is a 'union' of the solutions over all groups.

There are two issues in applying the technique to the new strengthened LP relaxation of our problem. Firstly, although the original constraint matrix is column-restricted, the new constraint matrix with the knapsack cover constraints is not. Secondly, unless additional assumptions are made, the current grouping-and-scaling analysis doesn't give a handle on the degree of violation of the upper

bound constraints. This is the reason why Kolliopoulos [12] needs the strong no-bottleneck assumption.

We get around the first difficulty by grouping the *rows* as well, into those that get most of their coverage from columns not affected by the knapsack constraints, and the remainder. On the first group of rows, we apply a subtle modification to the vanilla grouping-and-scaling analysis and obtain a $O(\gamma)$ approximate feasible solution satisfying these rows; we then show that one can treat the remainder of the rows as a PCIP and get a $O(\omega)$ approximate feasible solution satisfying them, using Assumption 2. Combining the two gives the $O(\gamma + \omega)$ factor. The full details are given in Section 2.

We stress here that apart from the integrality gap assumptions on the 0,1-CIPs, we do not make any other assumption (like the no-bottleneck assumption). In fact, we can use the modified analysis of the grouping-and-scaling technique to get a similar result as [12] for approximating CCIPs violating the upper-bound constraints, under a *weaker* assumption than the no-bottleneck assumption. The no-bottleneck assumption states that the supply of *any* column is less than the demand of *any* row. In particular, even though a column has entry 0 on a certain row, its supply needs to be less than the demand of that row. We show that if we weaken the no-bottleneck assumption to assuming that the supply of a column j is less than the demand of any row i only if $A[s]_{ij}$ is positive, a similar result can be obtained via our modified analysis.

Theorem 2. *Under assumption 1 and assuming $A_{ij}s_j \leq b_i$, for all i, j, given a fractional solution x to the canonical LP relaxation of $\texttt{Cov}(A[s], b, c, d)$, one can find an integral solution $x^{\texttt{int}}$ whose cost $c \cdot x^{\texttt{int}} \leq 10\gamma(c \cdot x)$ and $x^{\texttt{int}} \leq 10d$.*

Priority Covering Problems. In the following, we use PLC and PTC to refer to the priority versions of the line cover and tree cover problems, respectively. Recall that the constraint matrices for line and tree cover problems are totally unimodular, and the integrality of the corresponding 0,1-covering problems is therefore 1 in both case. It is interesting to note that the 0,1-coefficient matrices for PLC and PTC are not totally unimodular in general. The following integrality gap bound is obtained via a primal-dual algorithm.

Theorem 3. *The canonical LP for priority line cover has an integrality gap of at least* $3/2$ *and at most* 2.

In the case of tree cover, we obtain constant upper bounds on the integrality gap for the case $c = \mathbb{1}$, that is, for the minimum cardinality version of the problem. We believe that the PCIP for the tree cover problem with general costs also has a constant integrality gap. On the negative side, we can show an integrality gap of at least $\frac{e}{e-1}$.

Theorem 4. *The canonical LP for* unweighted *PTC has an integrality gap of at most* 6.

We obtain the upper bound by taking a given PTC instance and a fractional solution to its canonical LP, and decomposing it into a collection of PLC

instances with corresponding fractional solutions, with the following two properties. First, the total cost of the fractional solutions of the PLC instances is within a constant of the cost of the fractional solution of the PTC instance. Second, union of integral solutions to the PLC instances gives an integral solution to the PTC instance. The upper bound follows from Theorem 3. Using Theorem 1, we get the following as an immediate corollary.

Corollary 1. *There are $O(1)$-approximation algorithms for column-restricted line cover and the cardinality version of the column-restricted tree cover.*

We also obtain the following combinatorial results.

Theorem 5. *There is a polynomial-time exact algorithm for PLC.*

Theorem 6. *PTC is APX-hard, even when all the costs are unit.*

Theorem 7. *There is an efficient 2-approximation algorithm for PTC.*

The algorithm for PLC is a non-trivial dynamic programming approach that makes use of various structural observations about the optimal solution. The approximation algorithm for PTC is obtained via a similar decomposition used to prove Theorem 4.

We end by noting some interesting connections between the priority tree covering problem and set covering problems in computational geometry. The *rectangle cover* problem in 3-dimensions is the following: given a collection of points P in $\mathbb{R}^3$, and a collection C of axis-parallel rectangles with costs, find a minimum cost collection of rectangles that covers every point. We believe studying the PTC problem could give new insights into the rectangle cover problem.

Theorem 8. *The priority tree covering problem is a special case of the rectangle cover problem in 3-dimensions.*

Due to space restrictions, we omit many proofs. A full version of the paper is available [4].

2 General Framework for Column Restricted CIPs

In this section we prove Theorem 1. Our goal is to round a solution to a LP relaxation of $\texttt{Cov}(A[s], b, c, d)$ into an approximate integral solution. We strengthen the following canonical LP relaxation of the CCIP

$$\min\{c^T x \ : \ A[s]x \geq b, 0 \leq x \leq d, x \geq 0\}$$

by adding valid *knapsack cover* constraints. In the following we use $\mathcal{C}$ for the set of columns and $\mathcal{R}$ for the set of rows of A.

2.1 Strengthening the canonical LP Relaxation

Let $F \subset \mathcal{C}$ be a subset of the columns in the column restricted CIP $\texttt{Cov}(A[s], b, c, d)$. For all rows $i \in \mathcal{R}$, define $b_i^F = \max\{0, b_i - \sum_{j \in F} A[s]_{ij} d_j\}$ to be the residual demand of row i w.r.t. F. Define matrix $A^F[s]$ by letting

$$A^F[s]_{ij} = \begin{cases} \min\{A[s]_{ij}, b_i^F\} & : \quad j \in \mathcal{C} \setminus F \\ 0 & : \quad j \in F, \end{cases} \tag{2}$$

for all $i \in \mathcal{C}$ and for all $j \in \mathcal{R}$. The following *Knapsack-Cover* (KC) inequality

$$\sum_{j \in \mathcal{C}} A^F[s]_{ij} x_j \geq b_i^F$$

is valid for the set of all integer solutions x for $\texttt{Cov}(A[s], b, c, d)$. Adding the set of all KC inequalities yields the following stronger LP formulation CIP. We note that the LP is not column-restricted, in that, different values appear on the same column of the new constraint matrix.

$$\begin{aligned} \texttt{opt}_P := \min \quad & \sum_{j \in \mathcal{C}} c_j x_j & & \text{(P)} \\ \text{s.t.} \quad & \sum_{j \in \mathcal{C}} A^F[s]_{ij} x_j \geq b_i^F & \forall F \subseteq \mathcal{C}, \forall i \in \mathcal{R} \quad & (3) \\ & 0 \leq x_j \leq d_j & \forall j \in \mathcal{C} & \end{aligned}$$

It is not known whether (P) can be solved in polynomial time. For $\alpha \in (0, 1)$, call a vector x^* α-relaxed if its cost is at most $\texttt{opt}_P$, and if it satisfies (3) for $F = \{j \in \mathcal{C} : x_j^* \geq \alpha d_j\}$. An α-relaxed solution to (P) can be computed efficiently for any α. To see this note that one can check whether a candidate solution satisfies (3) for a set F; we are done if it does, and otherwise we have found an inequality of (P) that is violated, and we can make progress via the ellipsoid method. Details can be found in [3] and [15].

We fix an $\alpha \in (0, 1)$, specifying its precise value later. Compute an α-relaxed solution, x^*, for (P), and let $F = \{j \in \mathcal{C} : x_j^* \geq \alpha d_j\}$. Define $\bar{x}$ as, $\bar{x}_j = x_j^*$ if $j \in \mathcal{C} \setminus F$, and $\bar{x}_j = 0$, otherwise. Since x^* is an α-relaxed solution, we get that $\bar{x}$ is a feasible fractional solution to the *residual* CIP, $\texttt{Cov}(A^F[s], b^F, c, \alpha d)$. In the next subsection, our goal will be to obtain an *integral* feasible solution to the covering problem $\texttt{Cov}(A^F[s], b^F, c, d)$ using $\bar{x}$. The next lemma shows how this implies an approximation to our original CIP.

Lemma 1. *If there exists an integral feasible solution, $x^{\texttt{int}}$, to $\texttt{Cov}(A^F[s], b^F, c, d)$ with $c^T x^{\texttt{int}} \leq \beta \cdot c^T \bar{x}$, then there exists a $\max\{1/\alpha, \beta\}$-factor approximation to $\texttt{Cov}(A[s], b, c, d)$.*

2.2 Solving the Residual Problem

In this section we use a feasible fractional solution $\bar{x}$ of $\texttt{Cov}(A^F[s], b^F, c, \alpha d)$, to obtain an *integral* feasible solution $x^{\texttt{int}}$ to the covering problem $\texttt{Cov}(A^F[s], b^F, c, d)$, with $c^T x^{\texttt{int}} \leq \beta c^T \bar{x}$ for $\beta = 24\gamma + 8\omega$. Fix $\alpha = 1/24$.

Converting to Powers of 2**.** For ease of exposition, we first modify the input to the residual problem $\texttt{Cov}(A^F[s], b^F, c, d)$ so that all entries of are powers of 2. For every $i \in \mathcal{R}$, let $\bar{b}_i$ denote the smallest power of 2 larger than b_i^F. For every column $j \in \mathcal{C}$, let $\bar{s}_j$ denote the largest power of 2 smaller than s_j.

Lemma 2. *$y = 4\bar{x}$ is feasible for $\texttt{Cov}(A^F[\bar{s}], \bar{b}, c, 4\alpha d)$.*

Partitioning the rows. We call $\bar{b}_i$ the residual demand of row i. For a row i, a column $j \in \mathcal{C}$ is *i-large* if the supply of j is at least the residual demand of row i; it is *i-small* otherwise. Formally,

$$\mathcal{L}_i = \{j \in \mathcal{C} \,:\, A_{ij} = 1, \bar{s}_j \geq \bar{b}_i\} \quad \text{is the set of } i\text{-large columns}$$
$$\mathcal{S}_i = \{j \in \mathcal{C} \,:\, A_{ij} = 1, \bar{s}_j < \bar{b}_i\} \quad \text{is the set of } i\text{-small columns}$$

Recall the definition from (2), $A^F[\bar{s}]_{ij} = \min(A[\bar{s}]_{ij}, b_i^F)$. Therefore, $A^F[\bar{s}]_{ij} = A_{ij} b_i^F$ for all $j \in \mathcal{L}_i$ since $\bar{s}_j \geq \bar{b}_i \geq b_i^F$; and $A^F[\bar{s}]_{ij} = A_{ij}\bar{s}_j$ for all $j \in \mathcal{S}_i$, since being powers of 2, $\bar{s}_j < \bar{b}_i$ implies, $\bar{s}_j \leq \bar{b}_i/2 \leq b_i^F$.

We now partition the rows into large and small depending on which columns most of their coverage comes from. Formally, call a row $i \in \mathcal{R}$ *large* if

$$\sum_{j \in \mathcal{S}_i} A^F[\bar{s}]_{ij} y_j \leq \sum_{j \in \mathcal{L}_i} A^F[\bar{s}]_{ij} y_j,$$

and small otherwise. Note that Lemma 2 together with the fact that each column in row i's support is either small or large implies,

$$\sum_{j \in \mathcal{L}_i} A^F[\bar{s}]_{ij} y_j \geq \bar{b}_i/2, \text{ for all large rows } i, \text{ and}$$
$$\sum_{j \in \mathcal{S}_i} A^F[\bar{s}]_{ij} y_j \geq \bar{b}_i/2, \text{ for all small rows } i.$$

Let $\mathcal{R}_L$ and $\mathcal{R}_S$ be the set of large and small rows.

In the following, we address small and large rows separately. We compute a pair of integral solutions $x^{\texttt{int},\mathcal{S}}$ and $x^{\texttt{int},\mathcal{L}}$ that are feasible for the small and large rows, respectively. We then obtain $x^{\texttt{int}}$ by letting

$$x_j^{\texttt{int}} = \max\{x_j^{\texttt{int},\mathcal{S}}, x_j^{\texttt{int},\mathcal{L}}\}, \tag{4}$$

for all $j \in \mathcal{C}$.

Small rows. For these rows we use the grouping-and-scaling technique a la [7, 12–14]. However, as mentioned in the introduction, we use a modified analysis that bypasses the no-bottleneck assumptions made by earlier works.

Lemma 3. *We can find an integral solution* $x^{\mathtt{int},\mathcal{S}}$ *such that*
a) $x_j^{\mathtt{int},\mathcal{S}} \leq d_j$ *for all* j*,*
b) $\sum_{j\in\mathcal{C}} c_j x_j^{\mathtt{int},\mathcal{S}} \leq 24\gamma \sum_{j\in\mathcal{C}} c_j \bar{x}_j$*, and*
c) for every small row $i \in \mathcal{R}_S$*,* $\sum_{j\in\mathcal{C}} A^F[s]_{ij} x_j^{\mathtt{int},\mathcal{S}} \geq b_i^F$*.*

Proof. (Sketch) Since the rows are small, for any row i, we can zero out the entries that are larger than $\bar{b}_i$, and still $2y$ will be a feasible solution. Note that, now in each row, the entries are $< \bar{b}_i$, and thus are at most $\bar{b}_i/2$ (everything being powers of 2). We stress that it could be that $\bar{b}_i$ of some row is less than the entry in some other row, that is, we don't have the no-bottleneck assumption. However, when a particular row i is fixed, $\bar{b}_i$ is at least any entry of the matrix in the ith row. Our modified analysis of grouping and scaling then makes the proof go through.

We *group* the columns into classes that have s_j as the same power of 2, and for each row i we let $\bar{b}_i^{(t)}$ be the contribution of the class t columns towards the demand of row i. The columns of class t, the small rows, and the demands $\bar{b}_i^{(t)}$ form a CIP where all non-zero entries of the matrix are the same power of 2. We scale both the constraint matrix and $\bar{b}_i^{(t)}$ down by that power of 2 to get a 0,1-CIP, and using assumption 1, we get an integral solution to this 0,1-CIP. Our final integral solution is obtained by concatenating all these integral solutions over all classes.

Till now the algorithm is the standard grouping-and-scaling algorithm. The difference lies in our analysis in proving that this integral solution is feasible for the original CCIP. Originally the no-bottleneck assumption was used to prove this. However, we show that, since the column values in different classes are geometrically decreasing, the weaker assumption of $\bar{b}_i$ being at least any entry in the ith row is enough to make the analysis go through.

This completes the sketch of the proof.

Large rows. The large rows can be showed to be a PCIP problem and thus Assumption 2 can be invoked to get an analogous lemma to Lemma 3.

Lemma 4. *We can find an integral solution* $x^{\mathtt{int},\mathcal{L}}$ *such that*
a) $x_j^{\mathtt{int},\mathcal{L}} \leq 1$ *for all* j*,*
b) $\sum_{j\in\mathcal{C}} c_j x_j^{\mathtt{int},\mathcal{S}} \leq 8\omega \sum_{j\in\mathcal{C}} c_j \bar{x}_j$*, and*
c) for every large row $i \in \mathcal{R}_L$*,* $\sum_{j\in\mathcal{C}} A^F[s]_{ij} x_j^{\mathtt{int},\mathcal{S}} \geq b_i^F$*.*

Define $x^{\mathtt{int}}$ as $x_j^{\mathtt{int}} = \max\{x_j^{\mathtt{int},\mathcal{S}}, x_j^{\mathtt{int},\mathcal{L}}\}$ for all j; using the previous two lemmas and Lemma 1, this integral solution proves Theorem 1.

3 Priority line cover

In this extended abstract, we show that the integrality gap of the canonical linear programming relaxation of PLC is at most 2. Subsequently, we sketch an exact combinatorial algorithm for the problem.

3.1 Canonical LP relaxation: Integrality gap

We start with the canonical LP relaxation for PLC and its dual in Figure 1.

$$\min \Big\{ \sum_{j \in \mathcal{S}} c_j x_j : \quad x \in R_+^{\mathcal{S}} \qquad \text{(Primal)} \qquad \sum_{j \in \mathcal{S}: j \text{ covers } e} x_j \geq 1, \quad \forall e \in E \Big\}$$

$$\max \Big\{ \sum_{e \in E} y_e : \quad y \in R_+^{E} \ \text{(Dual)} \qquad \sum_{e \in E: j \text{ covers } e} y_e \leq c_j, \quad \forall j \in \mathcal{S} \Big\}$$

Fig. 1. The PLC canonical LP relaxation and its dual.

We use the terminology an edge e is larger than f, if $\pi_e \geq \pi_f$. The algorithm maintains a set of segments Q initially empty. Call an edge e *unsatisfied* if no segment in Q covers e and let U be the set of unsatisfied edges. The algorithm picks the largest edge in U and raises the dual value y_e till some segments becomes tight. The segments with the farthest left-end point and the farthest right-end point are picked in Q, and all edges contained in any of them are removed from U. Note that since we choose the largest in U, all such edges are covered. The algorithm repeats this process till U becomes $\emptyset$, that is, all edges are covered. The final set of segments is obtained by a reverse delete step, where a segment is deleted if its deletion doesn't make any edge uncovered.

The algorithm is a factor 2 approximation algorithm. To show this it suffices by a standard argument for analysing primal-dual algorithms, that any edge with a positive dual y_e is contained in at most two segments in Q. These two segments correspond to the left-most and the right-most segments that cover e; it is not too hard to show if something else covers e, then either e has zero dual, or the third segment is removed in the reverse delete step.

3.2 An Exact Algorithm for PLC

We sketch the exact algorithm for PLC. A segment j covers only a subset of edges it contains. We call a contiguous interval of edges covered by j, a *valley* of j. The uncovered edges form *mountains*. Thus a segment can be thought of as forming a series of valleys and mountains.

Given a solution $S \subseteq \mathcal{S}$ to the PLC (or even a PTC) instance, we say that segment $j \in S$ is *needed* for edge e if j is the unique segment in S that covers

e. We let $E_{S,j}$ be the set of edges that need segment j. We say a solution is *valley-minimal* if it satisfies the following two properties: (a) If a segment j is needed for edge e that lies in the valley v of j, then no higher supply segment of S intersects this valley v, and (b) every segment j is needed for its last and first edges. We show that an optimum solution can be assumed to be valley-minimal, and thus it suffices to find the minimum cost valley-minimal solution.

The crucial observation follows from properties (a) and (b) above. The valley-minimality of solution S implies that there is a unique segment $j \in S$ that covers the first edge of the line. At a very high level, we may now use j to decompose the given instance into a set of *smaller* instances. For this we first observe that each of the remaining segments in $S \setminus \{j\}$ is either fully contained in the strict interior of segment j, or it is disjoint from j, and lies to the right of it. The set of all segments that are disjoint from j form a feasible solution for the smaller PLC instance induced by the portion of the original line instance to the right of j. On the other hand, we show how to reduce the problem of finding an optimal solution for the part of the line contained in j to a single shortest-path computation in an auxiliary digraph. Each of the arcs in this digraph once again corresponds to a smaller sub-instance of the original PLC instance, and its cost is that of its optimal solution. The algorithm follows by dynamic programming.

4 Priority tree cover

In this extended abstract, we sketch a factor 2 approximation for the PTC problem, and show how the PTC problem is a special case of the 3 dimensional rectangle cover problem. For the APX hardness and the integrality gap of the unweighted PTC LP, we refer the reader to the full version.

4.1 An approximation algorithm for PTC

We use the exact algorithm for PLC to get the factor 2 algorithm for PTC. The crucial idea is the following. Given an optimum solution $S^* \subseteq \mathcal{S}$, we can partition the edge-set E of T into disjoint sets $E_1, \ldots, E_p$, and partition two copies of S^* into $S_1, \ldots, S_p$, such that E_i is a path in T for each i, and S_i is a priority line cover for the path E_i. Using this, we describe the 2-approximation algorithm which proves Theorem 7.

Proof of Theorem 7: For any two vertices t (top) and b (bottom) of the tree T, such that t is an ancestor of b, let P_{tb} be the unique path from b to t. Note that P_{tb}, together with the restrictions of the segments in $\mathcal{S}$to P_{tb}, defines an instance of PLC. Therefore, for each pair t and b, we can compute the optimal solution to the corresponding PLC instance using the exact algorithm; let the cost of this solution be c'_{tb}. Create an instance of the 0,1-tree cover problem with T and segments $\mathcal{S}' := \{(t, b) : t \text{ is an ancestor of } b\}$ with costs c'_{tb}. Solve the 0,1-tree cover instance exactly (recall we are in the rooted version) and for the segments

(t, b) in $\mathcal{S}'$ returned, return the solution of the corresponding PLC instance of cost c'_{tb}.

One now uses the decomposition above to obtain a solution to the 0,1-tree cover problem $(T, \mathcal{S}')$ of cost at most 2 times the cost of S^*. This proves the theorem. The segments in $\mathcal{S}'$ picked are precisely the segments corresponding to paths E_i, $i = 1, \ldots, p$ and each S_i is a solution to the PLC instance. Since we find the optimum PLC, there is a solution to $(T, \mathcal{S}')$ with costs c' of cost less than total cost of segments in $S_1 \cup \cdots \cup S_p$. But that cost is at most twice the cost of S^* since each segment of S^* is in at most two S_i's.

4.2 Priority Tree Cover and Geometric Covering Problems

We sketch how the PTC problem can be encoded as a rectangle cover problem. To do so, an auxiliary problem is defined, which we call 2-PLC.

2-Priority Line Cover (2-PLC) The input is similar to PLC, except each segment and edge has now an ordered pair of priorities, and a segment covers an edge it contains iff each of the priorities of the segment exceeds the corresponding priority of the edge. The goal, as in PLC, is to find a minimum cost cover.

It is not too hard to show 2-PLC is a special case of rectangle cover. The edges correspond to points in 3 dimension and segments correspond to rectangles in 3-dimension; dimensions encoded by the linear coordinates on the line, and the two priority values. In general, p-PLC can be shown to be a special case of $(p + 1)$-dimensional rectangle cover.

What is more involved is to show PTC is a special case of 2-PLC. To do so, we run two DFS orderings on the tree, where the order in which children of a node are visited is completely opposite in the two DFS orderings. The first ordering gives the order in which these edges must be placed on a line. The second gives one of the priorities for the edges. The second priority of the edges comes from the original priority in PTC. It can be shown that the segments priorities can be so set that the feasible solutions are precisely the same in both the instances proving Theorem 8.

5 Concluding Remarks

In this paper we studied column restricted covering integer programs. In particular, we studied the relationship between CCIPs and the underlying 0,1-CIPs. We conjecture that the approximability of a CCIP should be asymptotically within a constant factor of the integrality gap of the original 0,1-CIP. We couldn't show this; however, if the integrality gap of a PCIP is shown to be within a constant of the integrality gap of the 0,1-CIP, then we will be done. At this point, we don't even know how to prove that PCIPs of special 0,1-CIPS, those whose constraint matrices are totally unimodular, have constant integrality gap. Resolving the case of PTC is an important step in this direction, and hopefully in resolving our conjecture regarding CCIPs.

References

1. E. Balas. Facets of the knapsack polytope. *Math. Programming*, 8:146–164, 1975.
2. Amotz Bar-Noy, Reuven Bar-Yehuda, Ari Freund, Joseph Naor, and Baruch Schieber. A unified approach to approximating resource allocation and scheduling. *J. ACM*, 48(5):1069–1090, 2001.
3. R. D. Carr, L. K. Fleischer, V. J. Leung, and C. A. Phillips. Strengthening integrality gaps for capacitated network design and covering problems. In *Proceedings, ACM-SIAM Symposium on Discrete Algorithms*, pages 106–115, 2000.
4. D. Chakrabarty, E. Grant, and J. Könemann. On column-restricted and priority covering integer programs. *arXiv eprint*, 2010.
5. M. Charikar, J. Naor, and B. Schieber. Resource optimization in qos multicast routing of real-time multimedia. *IEEE/ACM Trans. Netw.*, 12(2):340–348, 2004.
6. C. Chekuri, A. Ene, and N. Korula. Unsplittable flow in paths and trees and column-restricted packing integer programs. In *Proceedings, International Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, page (*to appear*), 2009.
7. C. Chekuri, M. Mydlarz, and F. B. Shepherd. Multicommodity demand flow in a tree and packing integer programs. *ACM Trans. Alg.*, 3(3), 2007.
8. J. Chuzhoy, A. Gupta, J. Naor, and A. Sinha. On the approximability of some network design problems. *ACM Trans. Alg.*, 4(2), 2008.
9. G. Dobson. Worst-case analysis of greedy heuristics for integer programming with non-negative data. *Math. Oper. Res.*, 7(4):515–531, 1982.
10. P. Hammer, E. Johnson, and U. Peled. Facets of regular 0-1 polytopes. *Math. Programming*, 8:179–206, 1975.
11. D. S. Hochbaum. Approximation algorithms for the set covering and vertex cover problems. *SIAM Journal on Computing*, 11(3):555–556, 1982.
12. S. G. Kolliopoulos. Approximating covering integer programs with multiplicity constraints. *Discrete Appl. Math.*, 129(2-3):461–473, 2003.
13. S. G. Kolliopoulos and C. Stein. Approximation algorithms for single-source unsplittable flow. *SIAM Journal on Computing*, 31(3):919–946, 2001.
14. S. G. Kolliopoulos and C. Stein. Approximating disjoint-path problems using packing integer programs. *Math. Programming*, 99(1):63–87, 2004.
15. S. G. Kolliopoulos and N. E. Young. Approximation algorithms for covering/packing integer programs. *J. Comput. System Sci.*, 71(4):495–505, 2005.
16. Nitish Korula. private communication, 2009.
17. S. Rajagopalan and V. V. Vazirani. Primal-dual RNC approximation algorithms for (multi)set (multi)cover and covering integer programs. In *Proceedings, IEEE Symposium on Foundations of Computer Science*, 1993.
18. A. Schrijver. *Combinatorial optimization*. Springer, New York, 2003.
19. A. Srinivasan. Improved approximation guarantees for packing and covering integer programs. *SIAM Journal on Computing*, 29(2):648–670, 1999.
20. A. Srinivasan. An extension of the lovász local lemma, and its applications to integer programming. *SIAM Journal on Computing*, 36(3):609–634, 2006.
21. L. Trevisan. Non-approximability results for optimization problems on bounded degree instances. In *Proceedings, ACM Symposium on Theory of Computing*, pages 453–461, 2001.
22. L. Wolsey. Facets for a linear inequality in 0-1 variables. *Math. Programming*, 8:168–175, 1975.