Approximating covering integer programs with multiplicity constraints

Stavros G. Kolliopoulos

Abstract

In a covering integer program (CIP), we seek an n-vector x of nonnegative integers, which minimizes $c^T \cdot x$, subject to $Ax \geq b$, where all entries of $A, b, c$ are nonnegative. In their most general form, CIPs include also multiplicity constraints of the type $x \leq d$, i.e., arbitrarily large integers are not acceptable in the solution. The multiplicity constraints incur a dichotomy with respect to approximation between (0,1)-CIPs whose matrix $A$ contains only zeros and ones and the general case. Let $m$ denote the number of rows of $A$. The well known $O(\log m)$ cost approximation with respect to the optimum of the linear relaxation is valid for general CIPs, but multiplicity constraints can be dealt with effectively only in the (0,1) case. In the general case, existing algorithms that match the integrality gap for the cost objective violate the multiplicity constraints by a multiplicative $O(\log m)$ factor. We make progress by defining column-restricted CIPs, a strict superclass of (0,1)-CIPs, and showing how to find for them integral solutions of cost $O(\log m)$ times the LP optimum while violating the multiplicity constraints by a multiplicative $O(1)$ factor.

Keywords: Approximation algorithms; Covering integer programs; Integrality gap; Set multicovery

1. Introduction

In a covering integer program (CIP), we seek an n-vector x of nonnegative integers, which minimizes $c^T \cdot x$, subject to $Ax \geq b$, where all entries of $A, b, c$ are nonnegative.

1 Research partially supported by NSERC Grant 227809-00 and a CFI New Opportunities Award.

E-mail address: stavros@mcmaster.ca (S.G. Kolliopoulos).
Let the dimension of $A$ be $m \times n$. In their most general form, CIPs include also *multiplicity constraints* of the type $x \leq d$, i.e., arbitrarily large integers are not acceptable in the solution. Covering integer programs form a large subclass of integer programs (IPs) encompassing such classical problems as Minimum Knapsack and Set (multi)Cover. In particular, *Set Cover (Multicover)* is obtained from the general formulation when $A \in \{0,1\}^{m \times n}$ and $b^T = [1,1,\ldots,1]$ ($b \in \{1,2,\ldots\}^m$). In this problem, the set of rows of $A$ corresponds to a universe $M$ of $m$ elements. The set of columns corresponds to a collection $N$ of $n$ subsets of $M$. Element $i$ belongs to set $j$ iff $A_{ij} = 1$. The goal is to find a minimum cost subcollection $N' \subseteq N$ that covers all the elements. For Set Multicover, the constraint $x \leq d$, would imply that at most $d_j$ copies of set $j$ are available for potential inclusion in $N'$. In a general CIP multiplicity constraints express in a natural way a resource limitation: a fixed number of copies is available from each covering object, thus imposing an upper bound on the multiplicity of the latter in the final cover.

Solving CIPs to optimality is NP-hard, therefore, we are interested in efficient approximation algorithms which output a solution of near-optimal cost. In particular, we investigate how one can deal effectively with multiplicity constraints while finding a solution to a CIP of value as close as possible to the optimum of the corresponding linear relaxation. We give first a formal definition of CIPs after Srinivasan [26].

**Definition 1.** Given $A \in [0,1]^{m \times n}$, $b \in [1,\infty)^m$, $c \in [0,1]^n$ with $\max_j c_j = 1$ and $d \in \{1,2,\ldots\}^n$, a CIP $\mathcal{P} = (A,b,c,d)$ seeks to minimize $c^T \cdot x$ subject to $x \in \mathbb{Z}^n_+$, $x \leq d$ and $Ax \geq b$. If $A \in \{0,1\}^{m \times n}$, each entry of $b$ is assumed integral and the CIP is called $(0,1)$. Let $B$ and $\alpha$ denote, respectively, min, $b_i$, and the maximum number of nonzero entries in any column of $A$.

We will use the term *dilation* for the $\alpha$ parameter above. A *column-restricted* CIP (CCIP) is one where all nonzero entries of the $j$th column of $A$ have the same value $\rho_j$. Observe that $(0,1)$-CIPs are a special case of the column-restricted ones. We will use the term *general* CIP to emphasize the fact that the CIP in question is not $(0,1)$. The requirements on the actual numerical values in Definition 1 are without loss of generality. It is easy to see that all the coefficients can be scaled to lie between 0 and 1 without affecting the optimum solution [26]. Our results rely on the property that $B$ is at least as large as any column value. We note that if scaling is required to achieve this property, two values $A_{kj}$, $A_{lj}$ that were equal in the original input may end up different; hence, the original CCIP instance might not be column-restricted anymore.

1.1. Previous work

A $\rho$-approximate solution, $\rho > 1$, to an integer program is a feasible integral solution with objective value at most $\rho$ times the optimum. Let CIP$_\infty$ denote the problem of solving a covering integer program without multiplicity constraints. A CIP$_\infty$ instance is hence defined as a triple $(A,b,c)$ with $A,b,c$ as in Definition 1. There is a great amount of previous work focusing on approximating CIP$_\infty$ some of which is relevant for CIPs as well. Most of this work uses the value of the linear relaxation LP$_\infty$ as a lower bound.
on the optimum such as the early work of Lovász [19] and Chvátal [6] on Set Cover. In other work the error is analyzed directly with respect to some alternative estimate of the integral optimum [14,7,9]. Among more recent work that uses the value of LP_∞ as a lower bound the randomized rounding technique of Raghavan and Thompson [22] has proved particularly fruitful and relates to work in [21,29,26,25]. Closer to Chvátal’s dual-fitting technique [6] is the work of Rajagopalan and Vazirani in [23]. The reader is referred to the survey in [12] for a thorough discussion of the extensive literature on covering problems. Currently the best bounds for the CIP_∞ problem are due to Srinivasan [26,25].

From the body of work above, it is well known that the integrality gap of LP_∞ is Θ(1 + log z) and matching approximation algorithms exist. Moreover, it is unlikely that any other LP-relaxation with an asymptotically better integrality gap can be found. Starting with the work of Lund and Yannakakis [20] a series of papers established progressively stronger hardness of approximation results for Set Cover [3,8]. Raz and Safra [24] showed that it is NP-hard to obtain an o(ln m) approximation algorithm.

The hardness of approximation results for Set Cover apply also to a CIP as given in Definition 1. We now turn to examine positive results for the latter problem. There is work that provides approximation bounds that are functions of n, the number of columns of the A matrix [4,5]. The logarithmic approximation bounds we are interested in are given as functions of m, the number of rows. The currently best approximation algorithm is due to Dobson [7]. Dobson’s algorithm outputs a solution of cost O(max_{1≤i,j≤n} {log(∑_{i=1}^m A_{ij})}) times the integral optimum. The bound holds under the assumption that the entries of A have been scaled so that the minimum nonzero entry on each row is at least 1. For a CIP that conforms to Definition 1, if the minimum nonzero entry of A is 1/D, Dobson’s bound becomes O(log(Dz)) worst case.

The focus of our paper is on approximation guarantees with respect to the optimum of the linear relaxation of a CIP. This optimum is potentially much smaller than the integral optimum. We denote by y^* the value of this linear relaxation. The integer program we refer to each time will be clear from the context.

Simple as they appear, multiplicity constraints make the linear relaxation of a covering problem much weaker. The recent paper of Carr et al. [5] gives a simple instance of a minimum Knapsack problem (trivially a CCIP), for which the integrality gap can be made arbitrarily large.\(^2\) The CCIP below has an integrality gap of at least M > 0:

\[
\begin{align*}
\text{minimize} & \quad 0x_1 + x_2 \\
& \quad \frac{M - 1}{M} x_1 + x_2 \geq 1 \\
& \quad 0 \leq x_1, x_2 \leq 1.
\end{align*}
\]

However, if one multiplies the right-hand side of the multiplicity constraints by 2, the gap disappears from the resulting instance. This example demonstrates that, for any finite ρ, a ρ-approximate solution with respect to the LP optimum is impossible for a CCIP. With this motivation we define a \textit{k-relaxed solution}, \(k\) a nonnegative scalar, to

\(^2\) In [23] an O(log z) integrality gap was erroneously claimed for general CIPs.
be a vector \( x \) that satisfies the covering constraints \( Ax \geq b \) and the relaxed constraint \( 0 \leq x \leq kd \). In this terminology, solutions that satisfy the multiplicity constraints are 1-relaxed. Given a CIP, the question we are interested in is for how small a \( k \) can one find an integral \( k \)-relaxed solution of cost \( O(y_* \log m) \).

Currently the best approximations with respect to the LP optimum for general CIPs are due to Srinivasan \[26,25\]. Srinivasan’s cost guarantee is \( O(\min\{E_1, E_2\}) \) where the two expressions \( E_1 \) and \( E_2 \) are, respectively,

\[
y_*(1 + \max\{\ln(mB/y_*)/B, \sqrt{\ln(mB/y_*)/B}\}) \]
\[
y_*(1 + \max\{\ln(z + 1)/B, \sqrt{\ln(z + 1)/B}\})
\]

Although Srinivasan’s algorithms were given for CIP\(_\infty\) it is easy to see from the papers \[26,25\] that his solutions guarantee an upper bound on the violation of the multiplicity constraints: a multiplicative factor asymptotically equal to the approximation ratio attained for the cost. Hence until now, the best known integer solution of cost \( O(y_* \log m) \) is \( O(\log m) \)-relaxed. Better results are known for two special cases. Set Multicover, which is equivalent to a \((0,1)\)-CIP, is currently the most general formulation for which a 1-relaxed solution with a logarithmic cost guarantee can be obtained \[23\]. This result of Rajagopalan and Vazirani is obtained by a dual-fitting type analysis \[6\] of a simple greedy algorithm. We codify it in a theorem for future reference:

**Theorem 2** (Rajagopalan and Vazirani \[23\]). Given an instance of a \((0,1)\)-CIP, one can find in polynomial time a feasible integral solution of cost at most \( O(y_* \log z) \), where \( y_* \) is the optimum of the linear relaxation of the corresponding CIP.

A second special case was explored by Srinivasan and Teo \[27\]. Given a CIP where \( c_j = 1, j = 1, \ldots, n \) they showed through randomized-rounding techniques the following result: for any \( \varepsilon > 0 \), a vector \( \hat{x} \) of cost \( O(y_*(1 + \log z)) \) can be computed such that \( Ax \geq b \) and \( \hat{x} \leq [(1 + \varepsilon)d_j], j = 1, \ldots, n \). The cost guarantee in this result depends on \( 1/\varepsilon \) and becomes better than given if some conditions on \( m, B, \) and \( y_* \) are met \[27\].

1.2. **Our results**

Previous work suggests a dichotomy between \((0,1)\) and general CIPs as far as multiplicity constraints are concerned. Given the centrality of CIPs in combinatorial optimization, it is important to investigate how this dichotomy can be bridged. In this paper we introduce the separate study of column-restricted CIPs, a strict superclass of \((0,1)\)-CIPs. We design an approximation algorithm for CCIPs which outputs an \( O(1) \)-relaxed solution of cost \( O(y_* \log z) \).

As mentioned above, relaxing by some amount the multiplicity constraints is unavoidable if one wants to obtain a good cost approximation with respect to \( y_* \). Moreover, we show a second, negative, result: for any \( k > 1 \), a \( k \)-relaxed solution with cost \( o(y_* \log m) \) is impossible. Therefore, our cost approximation is asymptotically best possible, despite the extra liberty we take with the variable values.
1.3. Significance of the contribution

CCIPs present intrinsic theoretical interest as a strict generalization of (0,1)-CIPs. Moreover, the notion is particularly relevant in a network design setting, where a column of the matrix corresponds to a network edge and the column value stands for the edge capacity. Network design problems typically have an exponential number of constraints, one for each cut in the input graph, while the results in this paper assume that the constraint matrix $A$ is given explicitly. We elaborate on the connection to network design and the resulting open problems in Section 5. We believe that the concept of a $\mu$-relaxed solution, where $\mu$ is as small as possible for a given cost approximation, presents considerable theoretical interest: the integrality gap suggests that this is the only kind of solution that is possible if one wishes to stay close to the optimum cost of the linear relaxation. The notion of a relaxed solution could also be relevant for practice: we show that it suffices to supply a few extra copies of the covering objects to achieve a cost approximation which would have otherwise been impossible to obtain.

1.4. Algorithmic techniques

The algorithmic techniques we employ build on the grouping-and-scaling method introduced originally by Kolliopoulos and Stein [17] for the single-source unsplittable flow problem [15]. This technique was extended by the same authors in [16] to bridge the approximability gap between (0,1) and column-restricted Packing Integer Programs (CPIPs) thus culminating in the first nontrivial approximation for general multisource unsplittable flow. A Packing Integer Program (PIP) is of the form: maximize $c^T \cdot x$, subject to $Ax \leq b$, $0 \leq x \leq d$, where all coefficients are nonnegative. Grouping and scaling was also used later by Baveja and Srinivasan [2,1] in their examination of CPIPs. The main idea behind the technique is to decompose the problem $\mathcal{P}$ at hand into subproblems which are solved independently. For an IP, each subproblem groups in its constraint matrix only those columns of the original matrix $A$ whose values lie in a fixed range, say within a factor of 4 of each other. Apart from the column values, the definition of each subproblem is also based on the optimal fractional solution to $\mathcal{P}$. Information from the latter solution is used to define the covering requirements (i.e., $b$-vectors). By scaling the values we can transform each subproblem into an instance of Set Multicover; then one can use as a black box the algorithm suggested in Theorem 2. (In the packing setting the goal of the decomposition was to turn each subproblem into a (0,1)-PIP.) It is important to maintain the projection property: the sum of the fractional optima over all the subproblems is $O(y_*)$ ($\Omega(y_*)$ in the case of a PIP). An approximate integral solution is computed for each subproblem in isolation. If the decomposition is done carefully, the concatenation property should also hold: the partial integral solutions can be combined to form a feasible solution to the original problem. The above high-level description of the technique did not really distinguish between CCIPs and CPIPs, the latter being the original testing ground of the method. In a covering setting we have to cope with essentially the inverse requirements from those of a packing problem. Hence, the grouping-and-scaling technique was not known.
to apply to CIPs. In Section 3, we outline in detail the concrete difficulties arising when attempting to implement the decomposition. We find it rather surprising that the same high-level approach eventually applies to both packing and covering problems.

The outline of this paper is as follows. In Section 2 we give definitions. In Section 3 the algorithm for CCIPs and its analysis are presented. In Section 4, we prove a lower bound on the integrality gap of a CIP with arbitrarily large upper bounds on the variables. In Section 5 we present conclusions and open questions.

2. Preliminaries

All logarithms in this paper are base 2 unless otherwise noted. Let \( \mathcal{P} = (A, b, c, d) \) be a column-restricted CIP. We call \( \rho_j \leq 1 \), the value of the nonzero entries of the \( j \)th column, \( 1 \leq j \leq n \), the value of column \( j \). We use throughout the paper \( y_* \) to denote the optimum of the linear relaxation of the CIP 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. It should be clear from the context when a letter symbol denotes a vector. A number in boldface denotes a vector whose entries are all equal to the number. E.g., \( \mathbf{0} \) denotes a vector of zeros. The dimension of these vectors will also be clear from the context.

Given \( J_{\mathcal{VT}_i; \mathcal{VT}_j} \), let \( J_{\mathcal{VT}_i; \mathcal{VT}_j} \) be the set of column indices \( k \) for which \( z_i < \rho_k \leq z_j \). We then define, \( A_{\mathcal{VT}_i; \mathcal{VT}_j} \) to be the \( m \times |J_{\mathcal{VT}_i; \mathcal{VT}_j}| \) submatrix of \( A \) consisting of the columns in \( J_{\mathcal{VT}_i; \mathcal{VT}_j} \), and for any vector \( x \), \( x_{\mathcal{VT}_i; \mathcal{VT}_j} \) to be the \( |J_{\mathcal{VT}_i; \mathcal{VT}_j}| \)-entry subvector \( x \) consisting of the entries whose indices are in \( J_{\mathcal{VT}_i; \mathcal{VT}_j} \). We will also need to combine back together the various subvectors, and define \( x_{\mathcal{VT}_i; \mathcal{VT}_j} \cup \cdots \cup x_{\mathcal{VT}_i; \mathcal{VT}_j} \) to be the \( n \)-entry vector \( x' \) in which the entries of subvector \( x_{\mathcal{VT}_i; \mathcal{VT}_j} \), \( 1 < i \leq k \), are mapped back into the positions indexed by \( J_{\mathcal{VT}_i; \mathcal{VT}_i} \). Any positions in \( x' \) which are not indexed in \( \bigcup_{1 < i \leq k} J_{\mathcal{VT}_i; \mathcal{VT}_i} \) 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 \( y_{\mathcal{VT}_i; \mathcal{VT}_j} \) to denote \( (c_{\mathcal{VT}_i; \mathcal{VT}_j})^T x_{\mathcal{VT}_i; \mathcal{VT}_j} \). A \( \mu \)-relaxed integral (fractional) solution \( t \) to \( \mathcal{P} \), with \( \mu > 1 \) a scalar, will be an integral (fractional) vector satisfying \( A t \geq b \), \( t \leq \mu d \).

Our goal is to demonstrate that CCIPs admit approximations of similar quality to those known for (0,1)-CIPs with multiplicity constraints. Accordingly, we will give many of our results in terms of a generic bound for a (0,1)-CIP. Throughout Section 3 we assume that there is a polynomial-time algorithm \( \mathcal{A} \), which given a (0,1)-CIP with multiplicity constraints and fractional optimum \( y_* \), outputs an integral solution of value at most \( \sigma(m, z, y_*) \); \( m, z \) are the parameters of the (0,1)-CIP under consideration. Our only assumptions on \( \sigma \) are that it is an increasing function of its arguments and that it is linear in \( y_* \). By Theorem 2 the best-known \( \sigma \) is \( O(y_* \log z) \).

3. The algorithm for CCIPs

The main result of this section is a polynomial-time algorithm to obtain for a CCIP \( \mathcal{P} \) an \( O(1) \)-relaxed integral solution of cost \( O(\log z) \) times the fractional optimum of
This result matches the approximation ratio known for (0,1)-CIPs at the expense of a constant factor “congestion” on the multiplicity constraints.

The grouping-and-scaling technique we follow consists of decomposing the problem into subproblems of nice form. Every subproblem will then be transformed to an instance of Set Multicover with multiplicity constraints, i.e., a (0,1)-CIP. The solution to the original CCIP $\mathcal{P}$ will be formed by essentially concatenating the solution vectors of the subproblems. In the process, we must make sure that the solutions to the subproblems, when put together, are near-optimal for the original problem. We start by giving two lemmata that will be useful for reducing a subproblem to Set Multicover.

**Lemma 3.** Let $\mathcal{P} = (A, b, c, d)$ be a column-restricted CIP, in which all column values $\rho_i$ are equal to $\rho$ and each $b_i$ equals $k_i \rho$, a positive integer, $1 \leq i \leq m$. Here $\min b_i$ is not necessarily greater than 1. Let $y_*$ denote the optimum of the linear relaxation of $\mathcal{P}$. Then we can find in polynomial time a feasible solution to $\mathcal{P}$ of value at most $\sigma(m, x, y_*)$.

**Proof.** Transform the given program $\mathcal{P}$ to a (0,1)-CIP $\mathcal{P}' = (A', b', c, d)$, in which $b'_i = k_i$, 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 nonzero entries on any column is the same for $A$ and $A'$. Thus we can unambiguously use $x$ for both. By Theorem 2 we can find for $\mathcal{P}'$ an integral solution of value at most $\sigma(m, x, y_*)$. □

Now we show how the result from Lemma 3 can be extended to a more general form of CCIPs. For technical reasons we will allow constraints with all coefficients equal to zero. Call these constraints trivial.

**Lemma 4.** Let $\mathcal{P} = (A, b, c, d)$, be a column-restricted CIP with column values in the interval $(a_1, a_2]$, and $b_i \geq a_1$, for each nontrivial constraint $i$. Here $\min b_i$ is not necessarily greater than 1. Let $y_*$ denote the optimum of the linear relaxation of $\mathcal{P}$. There is a polynomial-time algorithm RANGE_COVERING, which finds an integral $[2a_2/a_1]$-relaxed solution $g$ to $\mathcal{P}$ of value $\sigma(m, x, (2a_2/a_1)y_*)$.

**Proof.** We sketch the algorithm RANGE_COVERING. Assume first that all the rows are nontrivial. Obtain a CIP $\mathcal{P}'=(A', b', c, d')$ from $\mathcal{P}$ as follows. Round up $b_i$ to the nearest integral multiple of $a_1$. Set $b'_i$ equal to the resulting value. Every $b'_i$ is now at most $2b_i$. Set $A'_{ij}$ to $a_1$ if $A_{ij} \neq 0$ and to 0 otherwise. If $x_*$ is an optimal fractional solution to $\mathcal{P}$, $(2a_2/a_1)x_*$ is a fractional solution to $\mathcal{P}'$ of value at most $(2a_2/a_1)c^T x_*$ = $(2a_2/a_1)y_*$. In order for the scaled $x_*$ to be feasible for $\mathcal{P}'$ we set $d' = \lceil 2a_2/a_1 \rceil d$.

All column values in $\mathcal{P}'$ are equal to $a_1$ and every $b'_i$ is an integral multiple of $a_1$. Thus, we can invoke Lemma 3 and find a feasible solution $g'$ to $\mathcal{P}$ of value at most $\sigma(m, x, (2a_2/a_1)y_*)$. Vector $g'$ is a $[2a_2/a_1]$-relaxed solution for $\mathcal{P}$. In case the original $\mathcal{P}$ contains trivial constraints, remove them and apply the same steps as above. The number of rows and the dilation can only decrease in the resulting IP. □
We now outline the ideas behind the decomposition part of our method and the difficulties of implementing them. We are going to decompose \( \mathcal{P} \) into covering subproblems \( P^i = (A^i, b^i, c^i, d^i) \) such that \( A^i \) contains only the columns of \( A \) with values in some fixed range \( (x_{j-1}, x_j] \). The subproblems will be amenable to good approximations as indicated by Lemma 4. We will obtain our integral solution to \( \mathcal{P} \) by combining these approximate solutions to the subproblems. A crucial step is the allocation of covering requirements on each \( b^i \) vector. The decomposition idea was originally formulated for packing problems [17] where as long as we do not overlap in the subproblems, we should be able to obtain a feasible solution for the original PIP. In a covering setting we face the inverse requirement: (a) We want each subproblem \( P^i \) to contribute to the \( i \)th constraint, an amount close to \( \sum_{j|p_j \in (x_{j-1}, x_j]} A_{ij}x_{sj} \). However, this quantity may be very small so setting \( b_i \) to it does not fulfill the hypothesis of Lemma 4 for \( P^i \). A second requirement is thus suggested by this lemma: (b) \( b_i \geq x_{j-1} \) for all \( i \). Moreover, in order to obtain a good approximation for cost, we want to meet a third requirement: (c) the fractional optimum of each \( P^i \) should be \( O(y^i_{x_i-1, x_i}) \). We would like to define the decomposition in a way that reconciles requirements (a)–(c). These requirements are potentially too strong so we settle for a decomposition which meets \( Ar \geq b - p \) for an appropriate constant \( p > 0 \). Algorithm \( \text{COVER}\_\text{PARTITION} \) does exactly this. Scaling up further \( t \) by a small constant will achieve full coverage, while incurring an asymptotically negligible increase on the cost.

We now present the algorithm \( \text{COVER}\_\text{PARTITION} \). Each subproblem will contain column values within a factor of \( 1/r \) of each other. The quantity \( r > 1 \) is a parameter passed to the algorithm and will be determined during the analysis of the performance guarantee. Not all subproblems will be required to contribute for a given constraint. For any \( 1 \leq i \leq m \), call an interval \( (x_{j-1}, x_j] \) \( i^\text{-weak} \) if \( \sum_{j|p_j \in (x_{j-1}, x_j]} A_{ij}x_{sj} < x_{j-1} \), else the interval is called \( i^\text{-strong} \). By our choice of the interval endpoints, the sum of the contribution of all \( i^\text{-weak} \) subproblems to the \( i \)th covering constraint will be shown to be bounded by \( 1/(r - 1) \). Hence, even if we throw the corresponding columns away as far as the \( i \)th row is concerned, we will be able to meet the \( b_i - 1/(r - 1) \) covering requirement. We use \( a(\rho) \) to denote the minimum column value.

**Algorithm \( \text{COVER}\_\text{PARTITION} \) \( (\mathcal{P}, r) \)**

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

**Step 2a:** Define a partition of the \( [a(\rho), 1] \) interval into \( \zeta = O(\log(1/a(\rho))) \) consecutive subintervals \( [a(\rho), r^{-[\log(1/a(\rho))]}, \ldots, (r^{-1}, r^{-1} - 1)] \). For \( \lambda = 1, \ldots, \zeta \) form subproblem \( P^\lambda = (A^\lambda, b^\lambda, c^\lambda, d^\lambda) \). \( A^\lambda, c^\lambda \) and \( d^\lambda \) are the restrictions defined by \( A^\lambda = A^{\lambda - \frac{1}{r^{-\lambda - 1}}}, c^\lambda = c^{\lambda - \frac{1}{r^{-\lambda - 1}}} \) and \( d^\lambda = d^{\lambda - \frac{1}{r^{-\lambda - 1}}} \). Matrix \( A^\lambda \), defined at Step 2b, is a modification of \( A^\lambda \).

**Step 2b:** Define matrix \( A^\lambda_2 \) equal to \( A^\lambda \). For each \( \lambda = 1, \ldots, \zeta \) and each \( i = 1, \ldots, m \) do: if the interval \( (r^{-i}, r^{-i + 1}] \) is \( i^\text{-weak} \), set all the entries of the \( i \)th row of \( A^\lambda_2 \) to 0.
Step 2c: For each \( \lambda = 1, \ldots, \zeta \) and each \( i = 1, \ldots, m \) do: set \( b_i^\lambda \) equal to the \( i \)th entry of \((A^\lambda_z \cdot x^\lambda_z)\).

Step 3: On each \( P^\lambda, 1 \leq \lambda \leq \zeta \), invoke RANGE_COVERING to obtain a solution vector \( \hat{x}^\lambda \). Combine the solutions to subproblems 1 through \( \zeta \) to form \( n \)-vector \( \hat{x} = \bigcup_{1 \leq \lambda \leq \zeta} \hat{x}^\lambda \). Output \( \hat{x} \).

We are faced with two tasks in the subsequent lemma. First we will show that the vector \( \hat{x} \) output by Algorithm COVER_PARTITION is an \( O(1) \)-relaxed solution to \( \mathcal{P}_1 = (A, b - 1/(r - 1), c, d) \). Second, we will upper bound \( c^T \cdot \hat{x} \) in terms of the fractional optimum \( y_\ast = c^T \cdot x_\ast \). We abbreviate \( y_\ast^{r^-\lambda} \) and \( x_\ast^{r^-\lambda} \) as \( y_\ast^\lambda \) and \( x_\ast^\lambda \), respectively.

**Lemma 5.** Given a CCIP \( \mathcal{P} = (A, b, c, d) \), and a fixed scalar \( r > 1 \), Algorithm COVER_PARTITION runs in polynomial time and the \( n \)-vector \( \hat{x} \) it outputs is a \([r^-\lambda] \)-relaxed solution to \( \mathcal{P}_1 = (A, b - 1/(r - 1), c, d) \). The value of \( \hat{x} \) is at most \( \sigma(m, z, 2ry_\ast^\lambda) \).

**Proof.** Let \( \hat{y}^\lambda \) be the optimum of the linear relaxation of \( P^\lambda \).

**Claim 6.** For each \( \lambda = 1, \ldots, \zeta \) and each \( i = 1, \ldots, m \), \((A^\lambda_z \cdot x^\lambda_z)_i = b_i^\lambda \). Moreover \( b_i^\lambda \in \{0\} \cup [r^-\lambda, +\infty) \).

**Proof of claim.** If all the entries of the \( i \)th row of \( A^\lambda_z \) are zero, \( b_i^\lambda \) is zero as well by Step 2c, so the claim holds. If there is a nonzero entry then by Step 2b, the interval is \( i^\ast \)-strong, so \((A^\lambda_z \cdot x^\lambda_z)_i \geq r^-\lambda \). Moreover, at Step 2c, \( b_i^\lambda \) is set to \((A^\lambda_z \cdot x^\lambda_z)_i \).

Claim 6 guarantees that \( x^\lambda_z \), i.e., the restriction of \( x_\ast \), is a feasible fractional solution for \( P^\lambda \). Therefore \( \hat{y}^\lambda \leq y^\lambda \). Moreover, the claim shows that each subproblem meets the hypothesis of Lemma 4. Hence, the value of \( x^\lambda \), \( 1 \leq \lambda \leq \zeta \), found at Step 3 via invocation of RANGE_COVERING is at most \( \sigma(m, z, 2ry_\ast^\lambda) \). By the assumption of linearity of \( \sigma \) the value of \( \hat{x} \) is

\[
\sum_{\lambda=1}^{\lambda=\zeta} \sigma(m, z, 2ry_\ast^\lambda) \leq \sigma(m, z, 2ry_\ast^\lambda). 
\]

The bound of this lemma on the value follows.

Denote as \( I_i \) the interval of column values corresponding to the subproblem \( P^\lambda \) in the decomposition. Also use \( f_i^\lambda \) to denote \( i \)th entry of \((A^\lambda_z \cdot x^\lambda_z)\). We emphasize that \( A^\lambda_z \) as the restriction of \( A \), is potentially different from \( A^\lambda_z \) as defined in Step 2b of the algorithm. Some of the rows of \( A^\lambda \) may have been zeroed out in \( A^\lambda_z \).

For the covering constraints, we observe that the aggregate covering requirement contributed by \( \hat{x} \) on row \( i \) of \( A \) is the sum of the covering requirements provided by \( \hat{x}^\lambda \), \( 1 \leq \lambda \leq \zeta \), on each subproblem. This sum is by Step 2c at least

\[
\sum_{I_i} b_i^\lambda = \sum_{i^\ast \text{strong } I_i} b_i^\lambda = \sum_{i^\ast \text{strong } I_i} f_i^\lambda.
\]
From the feasibility of $x_*$ for $\mathcal{P}$ and the definition of an $i*$-weak interval we obtain

$$
\sum_{i\text{-strong } l_i} f_i^* = \sum_{l_i} f_i^* - \sum_{i\text{-weak } l_i} f_i^* \geq b_i - \sum_{l \geq 1} r^{-l} \geq b_i - 1/(r - 1).
$$

It remains to estimate the amount by which the solution is relaxed. By Lemma 4 each of the vectors $\hat{x}$ is an integral $\left\lceil \frac{2}{r} \right\rceil$-relaxed solution to the corresponding subproblem. Therefore $\hat{x} \leq \left\lceil \frac{2}{r} \right\rceil d$.

For the running time, it suffices to observe that the decomposition generates $O(\log r(1/a(\rho)))$ subproblems, each of polynomial size. Clearly $\log n(1/a(\rho))$ is polynomial in the size of the original CCIP input.

We now work towards an $O(1)$-relaxed solution that actually satisfies the original covering constraints. The following theorem is the main result of this section.

**Theorem 7.** Given a CCIP $\mathcal{P} = (A, b, c, d)$, one can obtain in polynomial time an integer 12-relaxed solution of value $O(y_* \log z)$.

**Proof.** By Lemma 5 we can obtain, in polynomial time, for any fixed $r > 1$, a $\lceil 2r \rceil$-relaxed solution $\hat{x}$ to $\mathcal{P}_1 = (A, b - 1/(r - 1), c, d)$ of value at most $\sigma(m, x, 2r y_*)$. To obtain a relaxed solution for $\mathcal{P}$ we need to multiply $\hat{x}$ by a scalar $l$ such that

$$
l \geq \frac{b_i}{b_i - 1/(r - 1)} = \frac{r - 1}{r - 1 - 1/b_i}, \quad i = 1, \ldots, m.
$$

Since $B \geq 1$, $l$ must be at least $(r - 1)/(r - 2)$ for a total relaxation of $\lceil 2r \rceil (r - 1)/(r - 2)$. To minimize this quantity under the constraint that $l$ is an integer we choose $r = 3$. Thus, we obtain a vector of integers which is a 12-relaxed solution to $\mathcal{P}$ and has cost at most $\sigma(m, x, 12y_*)$. Instantiating $\sigma$ from Theorem 2 completes the proof.

Further improvements can be obtained if we know that $B \geq 2$. Then by choosing $r = 2$ the computation above yields an 8-relaxed solution of value at most $O(y_* \log z)$.

**4. A lower bound**

In this section we show that for any $k > 1$, a $k$-relaxed solution with cost $o(y_* \log m)$ is impossible. This demonstrates that the cost approximation we achieved in Theorem 7 is asymptotically optimal despite allowing ourselves to violate the upper bounds specified in the input. Theorem 8 is well known to hold for Set Cover instances, where $x \leq 1$. Our proof builds on the simple fact that increasing the value of a variable already set to 1 does not cover any additional elements.

**Theorem 8.** There is a $(0,1)$-CIP $\mathcal{P}$ with fractional optimum $y_*$ such that for any $k > 1$, a $k$-relaxed solution to $\mathcal{P}$ has value at least $y_* (\log m)/2$. 
Proof. Consider a Set Cover instance $S_f, f > 1$, with unit costs, which has the following two properties:

(1) the optimum $y_*$ of the linear relaxation is $O(1)$.

(2) for any $f - 1$ sets, there is an element that is not contained in their union.

Denote by $P$ the integer program associated with $S_f$. Let $P'$ be the (0,1)-CIP formulation for $S_f$ with arbitrarily large upper bounds on the variables. A solution to $P'$ corresponds to a $k$-relaxed solution to $P$ with arbitrarily large $k$. By Property 2, in any integral feasible solution to $P'$ at least $f$ variables must be set to at least 1. An instance with the two properties above that has $y_* = 2m/(m + 1)$ and $f = \log(m + 1)$ is given in [28, p. 111].

5. Discussion

In this paper, we presented an extension of the grouping-and-scaling technique, which was originally devised for packing problems, to the solution of covering problems with multiplicity constraints. We see as a basic feature of our decomposition method the way the optimal fractional solution is used: not for rounding per se but to illuminate the structure of the problem at hand and split the covering requirements among parts of the original problem. A second basic feature is the reduction of the problem at hand to the solution of a series of simpler (0,1) subproblems. For CCIPs, the subroutine to solve subproblems is a simple greedy algorithm, so the overall computation of our algorithm should be efficient in practice as well.

We see the solution of CCIPs in which the constraint matrix is only implicitly given as the main open question resulting from our work. This class of problems is of great interest in network design applications, where one wishes to select a minimum-cost subgraph of a given graph so that the capacity of every cut in the subgraph meets a specified covering requirement. Columns of the matrix correspond to edges and the value of a column to the edge capacity. Concretely, we are given a graph $G = (V,E)$ a nonnegative cost function $c : E \rightarrow \mathbb{R}^+$, a capacity function $u : E \rightarrow \mathbb{Z}^+$ and a requirement function $f$ defined on $2^V$. Let $\delta(S), S \subseteq V$, denote the set of edges with exactly one endpoint in $S$. The goal is to solve the following integer program:

$$\min \sum_{e \in E} c_e x_e$$

s.t. \quad \forall S \subseteq V \sum_{e \in \delta(S)} u_e x_e \geq f(S)$$

$$\forall e \in E \quad x_e \in \{0,1\}.$$-

The capacitated generalized Steiner network problem is obtained by defining a demand function $r : V \times V \rightarrow \mathbb{Z}^+$ and setting

$$f(S) = \max_{i \in S, j \notin S} r_{ij}, \quad \forall S \subset V.$$
The special case of the unit-capacity generalized Steiner network problem has been extensively studied (see the survey of Goemans and Williamson [11]) and recently Jain gave a two-approximation algorithm [13]. In contrast, the best-known algorithms for the capacitated problem have a worst-case performance guarantee of \( O(|E|) \) although they give better results for inputs with special structure [10,5]. Therefore even a \( \text{polylog} \)-cost approximate, \( O(1) \)-relaxed solution, would be of great interest. In this setting a \( \mu \)-relaxed solution is equivalent to the natural notion of increasing the edge capacities by the same \( \mu \)-factor. The network design problem does not necessarily meet the technical condition of Definition 1 that the minimum demand exceeds the maximum capacity. With a corresponding cost increase this can be easily overcome by solving \( O(\log(\max_{i,j \in V} r_{ij})) \) different problems, each defined on the entire graph \( G \). A direct extension of our technique would then reduce each of these capacitated problems to a solution of a logarithmic number of unit-capacity subproblems, conceivably using Jain’s algorithm as a subroutine to tackle the subproblems. However this approach apparently fails due to the condition imposed by existing algorithms for the unit-capacity case that the requirement function should be not only integer-valued but also weakly supermodular.

Finally we note that after the completion of this work, Kolliopoulos and Young gave new results on general CIPs [18]. In particular they showed how to obtain an \( O(\log m) \)-cost approximate, \( O(1) \)-relaxed solution for general CIPs by using a method based on randomized rounding. However, randomized rounding seems inapplicable to formulations with an exponential number of constraints. Hence, the technique in [18] is unlikely to extend to network design applications.

Acknowledgements

Thanks to Neal Young for valuable discussions. Thanks to Vijay Vazirani for a clarification on [23]. The comments of the two anonymous referees greatly improved the presentation.

References