Algorithmica (2011) 61:75–93
DOI 10.1007/s00453-010-9431-z

Approximability of Sparse Integer Programs

David Pritchard · Deeparnab Chakrabarty

Received: 9 February 2010 / Accepted: 6 July 2010 / Published online: 21 July 2010

Abstract The main focus of this paper is a pair of new approximation algorithms for certain integer programs. First, for covering integer programs $\{\min cx : Ax \geq b, \mathbf{0} \leq x \leq d\}$ where A has at most k nonzeroes per row, we give a k-approximation algorithm. (We assume A, b, c, d are nonnegative.) For any $k \geq 2$ and $\varepsilon > 0$, if $\mathsf{P} \neq \mathsf{NP}$ this ratio cannot be improved to $k - 1 - \varepsilon$, and under the unique games conjecture this ratio cannot be improved to $k - \varepsilon$. One key idea is to replace individual constraints by others that have better rounding properties but the same nonnegative integral solutions; another critical ingredient is knapsack-cover inequalities. Second, for packing integer programs $\{\max cx : Ax \leq b, \mathbf{0} \leq x \leq d\}$ where A has at most k nonzeroes per column, we give a $(2k^2 + 2)$-approximation algorithm. Our approach builds on the iterated LP relaxation framework. In addition, we obtain improved approximations for the second problem when $k = 2$, and for both problems when every A_{ij} is small compared to b_i. Finally, we demonstrate a $17/16$-inapproximability for covering integer programs with at most two nonzeroes per column.

Keywords Integer programming · Approximation algorithms · LP rounding

1 Introduction

We investigate the following problem: what is the best possible approximation ratio for integer programs where the constraint matrix is sparse? To put this in context we

The first author is partially supported by an NSERC post-doctoral fellowship.

D. Pritchard (✉)
Institute of Mathematics, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
e-mail: david.pritchard@epfl.ch

D. Chakrabarty
Department of Computer and Information Science, University of Pennsylvania, Philadelphia, USA
e-mail: deepc@seas.upenn.edu

recall a famous result of Lenstra [31]: integer programs with a constant number of variables or a constant number of constraints can be solved in polynomial time. Our investigations analogously ask what is possible if each constraint involves at most k variables, or if each variable appears in at most k constraints.

Rather than consider all integer programs, we consider only packing and covering problems. Such programs have only positive quantities in their parameters. One reason for this is that *every* integer program can be rewritten (possibly with additional variables) in such a way that each constraint contains at most 3 variables and each variable appears in at most 3 constraints, if both positive and negative coefficients are allowed. Aside from this, packing programs and covering programs capture a substantial number of combinatorial optimization problems and are interesting in their own right.

A *covering* (resp. *packing*) *integer program*, shorthanded as CIP (resp. PIP) henceforth, is an integer program of the form $\{\min cx : Ax \geq b, \mathbf{0} \leq x \leq d\}$ (resp. $\{\max cx : Ax \leq b, \mathbf{0} \leq x \leq d\}$) with A, b, c, d nonnegative and rational. Note that CIPs are sometimes called *multiset multicover* when A and b are integral. We call constraints $x \leq d$ *multiplicity constraints* (also known as *capacity constraints*). We allow for entries of d to be infinite, and without loss of generality, all finite entries of d are integral. An integer program with constraint matrix A is *k-row-sparse*, or *k-RS*, if each row of A has at most k entries; we define *k-column-sparse* (*k-CS*) similarly. As a rule of thumb we ignore the case $k = 1$, since such problems trivially admit fully polynomial-time approximation schemes (FPTAS's) or poly-time algorithms. The symbol $\mathbf{0}$ denotes the all-zero vector, and similarly $\mathbf{1}$ denotes the all-ones vector. For covering problems an *α-approximation algorithm* returns a feasible solution with objective value at most α times optimal; for packing, the algorithm returns a feasible solution with objective value is at least $1/\alpha$ times optimal. We use n to denote the number of variables and m the number of constraints (i.e. the number of columns and rows of A, respectively). Throughout the paper, A will be used as a matrix. We let A_j denote the jth column of A, and let a_i denote the ith row of A.

1.1 k-Row-Sparse Covering IPs

The special case of 2-RS CIP where A, b, c, d are 0-1 is the same as Min Vertex Cover, which is APX-hard. More generally, 0-1 k-RS CIP is the same as k-Bounded Hypergraph Min Vertex Cover (a.k.a. Set Cover with maximum frequency k) which is not approximable to $k - 1 - \varepsilon$ for any fixed $\varepsilon > 0$ unless $\mathsf{P} = \mathsf{NP}$ [9] ($k - \varepsilon$ under the unique games conjecture [24]). This special case is known to admit a matching positive result: set cover with maximum frequency k can be k-approximated by direct rounding of the naive LP [17] or local ratio/primal-dual methods [2].

The following results are known for other special cases of k-RS CIP with multiplicity constraints: Hochbaum [14] gave a k-approximation in the special case that A is 0-1; Hochbaum et al. [19] and Bar-Yehuda & Rawitz [3] gave pseudopolynomial 2-approximation algorithms for the case that $k = 2$ and d is finite. For the special case $d = \mathbf{1}$, Carr et al. [5, §2.6] gave a k-approximation, and Fujito & Yabuta [10] gave a primal-dual k-approximation. Moreover [5, 10] claim a k-approximation for

general d, however, the papers do not give a proof and we do not see a straightforward method of extending their techniques to the general d case. Our first main result, given in Sect. 2, is a simple proof of the same claim.

Theorem 1 *There is a polynomial time k-approximation algorithm for k-RS CIPs with multiplicity constraints.*

Our approach is to first consider the special case that there are no multiplicity constraints (i.e. $d_j = +\infty$ for all j); we then extend to the case of finite d via *knapsack-cover inequalities*, using linear programming (LP) techniques from Carr et al. [5]. A $(k+1)$-approximation algorithm is relatively easy to obtain using LP rounding; in order to get the tighter ratio k, we replace constraints by other "$\mathbb{Z}_+$-equivalent" constraints (see Definition 8) with better rounding properties. We require use of the ellipsoid algorithm, so the degree of the polynomial time complexity is a large constant.

Independent simultaneous work of Koufogiannakis & Young [28–30] also gives a full and correct proof of Theorem 1. Their approach works for a broad generalization of k-RS CIPs and runs in strongly polynomial (near-linear) time. Our approach has the generic advantage of giving new ideas that can be used in conjunction with other LP-based methods, and the specific advantage of giving integrality gap bounds (see Sect. 2.2).

1.2 k-Column-Sparse Packing IPs

Before 2009, no constant-factor approximation was known for k-CS PIPs, except in special cases. If every entry of b is $\Omega(\log m)$ then randomized rounding provides a constant-factor approximation. *Demand matching* is the special case of 2-CS PIP where (i) in each column of A all nonzero values in that column are equal to one another and (ii) no two columns have their nonzeroes in the same two rows. Shepherd & Vetta [36] showed demand matching is APX-hard but admits a $(\frac{11}{2} - \sqrt{5})$-approximation algorithm when $d = \mathbf{1}$; their approach also gives a $\frac{7}{2}$-approximation for 2-CS PIP instances satisfying (i). Results of Chekuri et al. [7] yield a $11.542k$-approximation algorithm for k-CS PIP instances satisfying (i) and such that the maximum entry of A is less than the minimum entry of b.

The special case of k-CS PIP where A, b are 0-1 is the same as *min-weight k-set packing*, *hypergraph matching with edges of size $\leq k$*, and *strong independent sets in hypergraphs with degree at most k*. The best approximation ratio known for this problem is $(k+1)/2 + \varepsilon$ [4] for general weights, and $k/2 + \varepsilon$ when $c = \mathbf{1}$ [20]. The best lower bound is due to Hazan et al. [16], who showed $\Omega(k/\ln k)$-inapproximability unless P = NP, even for $c = \mathbf{1}$.

Our second main result, given in Sect. 3, is the following result.

Theorem 2 *There is a polynomial time $(2k^2+2)$-approximation algorithm for k-CS PIPs with multiplicity constraints.*

We use the *iterated LP relaxation* [37] technique to find an integral solution whose objective value is larger than the optimum, but violates some constraints. However

the violation can be bounded. Then we use a colouring argument to decompose the violating solution into $O(k^2)$ feasible solutions giving us the $O(k^2)$-factor algorithm. We require a linear programming subroutine, so the degree of the polynomial time complexity is a large constant.

The original arXiv eprint and conference version [34] of this work gave a $O(k^2 2^k)$-approximation for k-CS PIP using iterated relaxation plus a randomized decomposition approach; that was the first approximation algorithm for this problem with ratio that depends only on k. Subsequently in April 2009, C. Chekuri, A. Ene and N. Korula (personal communication) obtained an $O(k2^k)$ algorithm using randomized rounding, and an $O(k^2)$-approximation in May 2009. The latter method was independently re-derived by the authors, which appears in this version. Finally, Bansal et al. [1], in August 2009, gave a simple and elegant $O(k)$-approximation algorithm based on randomized rounding with a careful alteration argument. A special case of this result, for "column-restricted" programs (where all nonzeroes in each column are equal) appeared in Chekuri et al. [8]. They also give a simple greedy k-approximation for column-restricted unit-profit ($c = \mathbf{1}$) k-CS PIPs.

1.3 k-Column-Sparse Covering IPs

Srinivasan [38, 39] showed that k-CS CIPs admit a $O(\log k)$-approximation. Kolliopoulos and Young [26] extended this result to handle multiplicity constraints. There is a matching hardness result: it is NP-hard to approximate k-Set Cover, which is the special case where A, b, c are 0-1, better than $\ln k - O(\ln \ln k)$ for any $k \geq 3$ [40]. Hence for k-CS CIP the best possible approximation ratio is $\Theta(\log k)$. A $(k + \varepsilon)$-approximation algorithm can be obtained by separately applying an approximation scheme to the knapsack problem corresponding to each constraint. Although 0-1 2-CS CIP is Edge Cover which lies in P, general 2-CS CIP is NP-hard due to Hochbaum [18], who also gave a bicriteria approximation algorithm. Here, we give a stronger inapproximability result.

Theorem 3 *For every $\varepsilon > 0$ it is* NP*-hard to approximate* 2*-CS CIPs of the form* $\{\min c \cdot x \mid Ax \geq b, x \text{ is } 0\text{-}1\}$ *and* $\{\min c \cdot x \mid Ax \geq b, x \geq 0, x \text{ integral}\}$ *within ratio* $17/16 - \varepsilon$ *even if the nonzeroes of every column of A are equal and A is of the block form* $\begin{bmatrix} A_1 \\ A_2 \end{bmatrix}$ *where each A_i is* 1*-CS.*

Our proof modifies a construction of [6]; we also note a construction of [36] can be modified to prove APX-hardness for the problem.

1.4 Other Work

The special case of 2-RS PIP where A, b, c are 0-1 is the same as Max Independent Set, which is not approximable within $n/2^{\log^{3/4+\varepsilon} n}$ unless $\mathsf{NP} \subset \mathsf{BPTIME}(2^{\log^{O(1)} n})$ [23]. On the other hand, n-approximation of any packing problem is easy to accomplish by looking at the best singleton-support solution. A slightly better n/t-approximation, for any fixed t, can be accomplished by exhaustively guessing the t most profitable variables in the optimal solution, and then solving the resulting t-dimensional integer program to optimality via Lenstra's result [31].

Table 1 The landscape of approximability of sparse integer programs. Our main results are in boldface

	k-Column-Sparse		k-Row-Sparse	
	lower bound	upper bound	lower bound	upper bound
Packing	$\Omega(k/\ln k)$	$\mathbf{2k^2+2}$, $O(k)$	$n^{1-o(1)}$	εn
Covering	$\ln k - O(\ln\ln k)$	$O(\ln k)$	$k-\varepsilon$	$\mathbf{k}$

A closely related problem is k-Dimensional Knapsack, which are PIPs or CIPs with at most k constraints (in addition to nonnegativity and multiplicity constraints). For fixed k, such problems admit a PTAS and pseudo-polynomial time algorithms, but are weakly NP-hard; see [22] and [35, Chap. 9] for detailed references.

When $d = \mathbf{1}$, a natural way to generalize CIP/PIPs is to allow the objective function to be submodular (rather than linear). For minimizing a nondecreasing nonnegative submodular objective subject to k-row sparse covering constraints, the framework of Koufogiannakis & Young [28–30] gives a k-approximation; we show at the end of Sect. 2.1 that our Theorem 1 also extends to this case, based on helpful referee feedback. If also A, b are 0-1 (i.e. submodular set cover) Iwata and Nagano [21] give a k-approximation for all k and Goel et al. [12] give a 2-approximation for $k = 2$. For maximizing a monotone submodular function subject to k-column sparse packing constraints, the algorithm of Bansal et al. [1] gives a $O(k)$-approximation.

1.5 Summary

We summarize our results and the best others in Table 1. Note that in all four cases, the strongest known lower bounds are obtained even in the special case that A, b, c, d are 0-1.

2 k-Approximation for k-Row-Sparse CIPs

By scaling rows suitably and clipping coefficients that are too high (i.e. setting $A_{ij} = \min\{1, A_{ij}\}$), we may make the following assumption without loss of generality.

Definition 4 A *k-RS CIP* is an integer program $\{\min c \cdot x : Ax \geq \mathbf{1}, \mathbf{0} \leq x \leq d, x \in \mathbb{Z}\}$ where A is k-RS and all entries of A are at most 1.

To begin with, we focus on the case $d_j = +\infty$ for all j, which we call the *unbounded k-RS CIP*, since it illustrates the essence of our new technique. Let x be a n-dimensional vector of variables and α is a vector of real coefficients. Throughout, we assume coefficients are nonnegative. When we apply $\lfloor\cdot\rfloor$ to vectors we mean the component-wise floor. That is, the jth coordinate of $\lfloor\alpha\rfloor$ is $\lfloor\alpha_j\rfloor$.

Definition 5 A constraint $\alpha \cdot x \geq 1$ is *ρ-roundable* for some $\rho > 1$ if for all nonnegative real x, $(\alpha \cdot x \geq 1)$ implies $(\alpha \cdot \lfloor \rho x \rfloor \geq 1)$.

Note that ρ-roundability implies ρ'-roundability for $\rho' > \rho$. The relevance of this property is explained by the following proposition.

Proposition 6 *If every constraint in an unbounded covering integer program is ρ-roundable, then there is a ρ-approximation algorithm for the program.*

Proof Let x^* be an optimal solution to the program's linear relaxation. Then $c \cdot x^*$ is a lower bound on the cost of any optimal solution. Thus, $\lfloor \rho x^* \rfloor$ is a feasible integral solution with cost at most ρ times optimal. □

We make another simple observation.

Proposition 7 *The constraint $\alpha \cdot x \geq 1$ is $(1 + \sum_i \alpha_i)$-roundable.*

Proof Let $\rho = (1 + \sum_i \alpha_i)$. Since $\lfloor t \rfloor > t - 1$ for any t, if $\alpha \cdot x \geq 1$ for a nonnegative x, then

$$\alpha \cdot \lfloor \rho x \rfloor \geq \sum_i \alpha_i (\rho x_i - 1) = \rho \sum_i \alpha_i x_i - \sum_i \alpha_i \geq \rho - (\rho - 1) = 1,$$

as needed. □

Now consider an unbounded k-RS CIP. Since each constraint has at most k coefficients, each less than 1, it follows from Proposition 7 that every constraint in these programs is $(k+1)$-roundable, and so such programs admit a $(k+1)$-approximation algorithm by Proposition 6. It is also clear that we can tighten the approximation ratio to k for programs where the sum of the coefficients in every constraint (row) is at most $k - 1$. We now show that rows with sum in $(k-1, k]$ can be replaced by other rows which are k-roundable.

Definition 8 Two constraints $\alpha \cdot x \geq 1$ and $\alpha' \cdot x \geq 1$ are $\mathbb{Z}_+$-*equivalent* if for all nonnegative integral x, $(\alpha \cdot x \geq 1) \Leftrightarrow (\alpha' \cdot x \geq 1)$.

In other words, replacing a constraint by an $\mathbb{Z}_+$-equivalent constraint doesn't affect the value of the CIP.

Proposition 9 *Every constraint $\alpha \cdot x \geq 1$ with at most k nonzero coefficients is $\mathbb{Z}_+$-equivalent to a k-roundable constraint.*

Before proving Proposition 9, let us illustrate its use.

Theorem 10 *There is a polynomial time k-approximation algorithm for unbounded k-RS CIPs.*

Proof Using Proposition 9 we replace each constraint with a $\mathbb{Z}_+$-equivalent k-roundable one. The resulting IP has the same set of feasible solutions and the same

objective function. Therefore, Proposition 6 yields a k-approximately optimal solution. □

With the framework set up, we begin the technical part: a lemma, then the proof of Proposition 9.

Lemma 11 *For any positive integers k and v, the constraint $\sum_{i=1}^{k-1} x_i + \frac{1}{v}x_k \geq 1$ is k-roundable.*

Proof Let $\alpha \cdot x \geq 1$ denote the constraint, i.e. $\alpha_k = \frac{1}{v}$, $\alpha_i = 1$ for $1 \leq i < k$. If x satisfies the constraint, then the maximum of $x_1, x_2, \ldots, x_{k-1}$ and $\frac{1}{v}x_k$ must be at least $1/k$. If $x_i \geq 1/k$ for some $i \neq k$ then $\lfloor kx_i \rfloor \geq 1$ and so $\alpha \cdot \lfloor kx \rfloor \geq 1$ as needed. Otherwise x_k must be at least v/k and so $\lfloor kx_k \rfloor \geq v$ which implies $\alpha \cdot \lfloor kx \rfloor \geq 1$ as needed. □

Proof of Proposition 9 If the sum of coefficients in the constraint is $k-1$ or less, we are done by Proposition 7, hence we assume the sum is strictly greater than $k-1$. Without loss of generality (by renaming) such a constraint is of the form

$$\sum_{i=1}^{k} x_i \alpha_i \geq 1 \tag{1}$$

where $\mathbf{0} < \alpha \leq \mathbf{1}$, $k-1 < \sum_i \alpha_i \leq k$, and the α_i's are nonincreasing in i.

Define the *support* of x to be $\mathtt{supp}(x) := \{i \mid x_i > 0\}$. We claim that for any two distinct j, ℓ, $\alpha_j + \alpha_\ell > 1$. Otherwise, the $\sum_i \alpha_i \leq (k-2)+1 = k-1$. Thus, for any feasible integral x with $|\mathtt{supp}(x)| \geq 2$, we have $\alpha \cdot x \geq 1$. To express the set of *all* feasible integral solutions, let t be the maximum i for which $\alpha_i = 1$ (or $t = 0$ if no such i exists), let e_i denote the ith unit basis vector, and let $v = \lceil 1/\alpha_k \rceil$. Then it is not hard to see that the nonnegative integral solution set to constraint (1) is the disjoint union

$$\begin{aligned} &\{x \mid x \geq 0, |\mathtt{supp}(x)| \geq 2\} \uplus \{ze_i \mid 1 \leq i \leq t, z \geq 1, z \in \mathbb{Z}\} \\ &\quad \uplus \{ze_i \mid t < i < k, z \geq 2, z \in \mathbb{Z}\} \uplus \{ze_k \mid z \geq v, z \in \mathbb{Z}\}. \end{aligned} \tag{2}$$

The special case $t = k$ (i.e. $\alpha_1 = \alpha_2 = \cdots = \alpha_k = 1$) is already k-roundable by Lemma 11, so assume $t < k$. Consider the constraint

$$\sum_{i=1}^{t} x_i + \sum_{i=t+1}^{k-1} \frac{v-1}{v} x_i + \frac{1}{v} x_k \geq 1. \tag{3}$$

Every integral $x \geq 0$ with $|\mathtt{supp}(x)| \geq 2$ satisfies constraint (3). By also considering the cases $|\mathtt{supp}(x)| \in \{0, 1\}$, it is easy to check that constraint (3) has precisely (2) as its set of feasible solutions, i.e. constraint (3) is $\mathbb{Z}_+$-equivalent to $\alpha x \geq 1$. If $t < k-1$, the sum of the coefficients of constraint (3) is $k-1$ or less, so it is k-roundable by Proposition 7. If $t = k-1$, constraint (3) is k-roundable by Lemma 11. Thus in either case we have what we wanted. □

2.1 Multiplicity Constraints

We next obtain approximation guarantee k even with multiplicity constraints $x \le d$. For this we use *knapsack-cover inequalities*. These inequalities represent residual covering problems when a set of variables is taken at maximum multiplicity. Wolsey [41] studied inequalities like this for 0-1 problems to get a primal-dual approximation algorithm for submodular set cover. The LP we use is similar to what appears in Carr et al. [5] and Kolliopoulos & Young [26], but we first replace each row with a k-roundable one.

Specifically, given a CIP $\{\min c \cdot x \mid Ax \ge \mathbf{1}, \mathbf{0} \le x \le d, x \in \mathbb{Z}\}$ with A, d non-negative, we now define the knapsack cover LP. Note that we allow d to contain some entries equal to $+\infty$; if $d_j = +\infty$ and some i has $A_{ij} = 0$ our convention is that $A_{ij}d_j = 0$. Recall, a_i is the ith row of A and $\mathtt{supp}(a_i)$ denotes the set $\{j : A_{ij} > 0\}$. For a subset F of $\mathtt{supp}(a_i)$ such that $\sum_{j\in F} A_{ij}d_j < 1$, define $A_{ij}^{(F)} = \min\{A_{ij}, 1 - \sum_{j\in F} A_{ij}d_j\}$. Following [5, 26] we define the *knapsack cover LP* for our problem to be

$$\text{KC-LP} = \Big\{\min c \cdot x : \mathbf{0} \le x \le d; \quad \forall i, \forall F \subset \mathtt{supp}(a_i) \text{ s.t. } \sum_{j\in F} A_{ij}d_j < 1\text{: } \sum_{j\notin F} A_{ij}^{(F)} x_j \ge 1 - \sum_{j\in F} A_{ij}d_j\Big\}.$$

It is not too hard to check that any integral solution to the CIP satisfies the constraints of KC-LP, and thus the solution to the latter is a lower bound on the value of the CIP.

Theorem 1 *There is a polynomial time k-approximation algorithm for k-RS CIPs.*

Proof Using Proposition 9, we assume all rows of A are k-roundable. Let x^* be the optimal solution to KC-LP. Define $\widehat{x} = \min\{d, \lfloor kx^* \rfloor\}$, where min denotes the component-wise minimum. We claim that $\widehat{x}$ is a feasible solution to the CIP, which will complete the proof since the objective value of $\widehat{x}$ is at most k times the objective value of KC-LP. In other words, we want to show for each row i that $a_i \cdot \widehat{x} \ge 1$.

Fix any row i and define $F = \{j \in \mathtt{supp}(a_i) \mid x_j^* \ge d_j/k\}$, i.e. F is those variables in the constraint that were rounded to their maximum multiplicity. If $F = \varnothing$ then, by the k-roundability of $a_i \cdot x \ge 1$, we have that $a_i \cdot \widehat{x} = a_i \cdot \lfloor kx^* \rfloor \ge 1$ as needed. So assume $F \ne \varnothing$. Note that for $j \in F$, we have $\widehat{x}_j = d_j$ and for $j \notin F$, we have $\widehat{x}_j = \lfloor kx_j^* \rfloor$.

If $\sum_{j\in F} A_{ij}d_j \ge 1$ then the constraint $a_i \cdot \widehat{x} \ge 1$ is satisfied; consider otherwise. Since $\lfloor kx_j^* \rfloor > kx_j^* - 1$ for $j \notin F$, since x^* satisfies the knapsack cover constraint for i and F, and since $A_{ij}^{(F)} \le 1 - \sum_{j\in F} A_{ij}d_j$ for each j, we have

$$\begin{aligned}\sum_{j\notin F} A_{ij}^{(F)} \widehat{x}_j &= \sum_{j\notin F} A_{ij}^{(F)} \lfloor kx_j^* \rfloor \\ &\ge k \sum_{j\notin F} A_{ij}^{(F)} x_j^* - \sum_{j\notin F} A_{ij}^{(F)}\end{aligned}$$

$$\geq k\Big(1-\sum_{j\in F} A_{ij}d_j\Big) - \Big|\{j : j \in \mathtt{supp}(a_i)\backslash F\}\Big|\Big(1-\sum_{j\in F} A_{ij}d_j\Big)$$

$$= k\Big(1-\sum_{j\in F} A_{ij}\widehat{x}_j\Big) - \Big|\{j : j \in \mathtt{supp}(a_i)\backslash F\}\Big|\Big(1-\sum_{j\in F} A_{ij}\widehat{x}_j\Big).$$

Since $F \neq \varnothing$ and $|\mathtt{supp}(a_i)| \leq k$, this gives $\sum_{j\notin F} A_{ij}^{(F)}\widehat{x}_j \geq 1 - \sum_{j\in F} A_{ij}\widehat{x}_j$. Rearranging, and using the fact $(\forall j : A_{ij} \geq A_{ij}^{(F)})$, we deduce $a_i \cdot \widehat{x} \geq 1$, as needed.

For fixed k, we may solve KC-LP explicitly, since it has polynomially many constraints. For general k, no method is currently known to solve KC-LP in polynomial time. However, one can use the ellipsoid method to find a solution x^* whose objective is lower than that of KC-LP, and which satisfies the knapsack-cover constraints corresponding to the set $F = \{j : x_j^* \geq d_j/k\}$. Note that this is all we need for the above analysis. Details of how the ellipsoid method finds such a solution are given in [5, 26]. □

Submodular Optimization Theorem 1 can be generalized to non-decreasing non-negative submodular objective functions when $d = \mathbf{1}$. The Koufogiannakis-Young approach [30] gives a much faster result of the same approximation quality, and thus we only sketch the details. Let $f(x)$ denote the objective function, then the key is to use the Lovász extension [32] of f, which is a convex piecewise linear function $\widetilde{f} : [0,1]^n \to \mathbb{R}$ that agrees with f at integral inputs. The ellipsoid algorithm-based approach may again be applied for this objective function [13, Thm. 6.5.19]. Our new algorithm minimizes $\widetilde{f}(x)$ subject to knapsack-cover constraints generated like before. We round the optimal fractional point x to an integral $\widehat{x}$ by rounding coordinates at least $1/k$ to 1 and others to 0, as before. The analysis revolves around the fact $f(\widehat{x}) \leq k\widetilde{f}(x)$ which is straightforward from the definition of the Lovász extension.

2.2 Integrality Gap Bounds

In discussing integrality gaps for k-RS CIP problems, we say that the *naive LP relaxation* of $\{\min c \cdot x \mid Ax \geq b, \mathbf{0} \leq x \leq d, x \in \mathbb{Z}\}$ is the LP obtained by removing the restriction of integrality. Earlier, we made the assumption that $A_{ij} \leq b_i$ for all i, j; let us call this the *clipping assumption*. The clipping assumption is without loss of generality for the purposes of approximation guarantees, however, it affects the integrality gap of the naive LP for unbounded k-RS CIP, as we now illustrate. Without the clipping assumption, the integrality gap of k-RS CIP problems can be unbounded as a function of k; indeed for any integer $M \geq 1$ the well-known covering problem $\{\min x_1 \mid [M]x_1 \geq 1, 0 \leq x_1\}$ has integrality gap M. In instances with the clipping assumption and without multiplicity constraints, the previous methods in this section establish that the integrality gap of the naive LP is at most $k+1$.

Even under the clipping assumption, it is well-known that k-RS CIPs with *multiplicity constraints* can have large integrality gaps—e.g. $\{\min x_2 \mid [\begin{smallmatrix} M \\ M \end{smallmatrix}]x \geq M+1,\ \mathbf{0} \leq x,\ x_1 \leq 1\}$ has integrality gap M. For bounded instances, the knapsack-cover inequalities represent a natural generalization of the clipping assumption, namely, we perform a sort of clipping even considering that any subset of the variables are chosen to their maximum extent.

We have seen that KC-LP has integrality gap at most $k+1$ on k-RS CIP instances. Our methods also show that if we replace each row with a k-roundable one (Proposition 9), then the corresponding KC-LP has integrality gap at most k. We are actually unaware of any k-RS CIP instance with $k > 1$ where the integrality gap of KC-LP (without applying Proposition 9) is greater than k; resolving whether such an instance exists would be interesting. Some special cases are understood, e.g. Koufogiannakis and Young [29] give a primal-dual k-approximation for k-CS PIP in the case A is 0-1, also known as hypergraph b-matching.

3 Column-Sparse Packing Integer Programs

In this section we give an approximation algorithm for k-column-sparse packing integer programs with approximation ratio $2k^2 + 2$. We better results for $k = 2$, and for programs with high width (we defer the definition to a later subsection). The results hold even in the presence of multiplicity constraints $x \leq d$. Broadly speaking, our approach is rooted in the demand matching algorithm of Shepherd & Vetta [36]; their path-augmenting algorithm can be viewed as a restricted form of *iterated relaxation*, which is the main tool in our new approach. Iterated relaxation yields a solution whose objective value is *larger* than the optimum, however, the solution violates some constraints. We then decompose this infeasible solution to a collection of feasible solutions while retaining at least a constant fraction of the objective value.

For a k-CS PIP $\mathcal{P}$ let $\mathcal{L}(\mathcal{P})$ denote its linear relaxation $\{\max c \cdot x \mid Ax \leq b, \mathbf{0} \leq x \leq d\}$. We use the set I to index the constraints and J to index the variables in our program. We note a simple assumption that is without loss of generality for the purposes of obtaining an approximation algorithm: $A_{ij} \leq b_i$ for all i, j. To see this, note that if $A_{ij} > b_i$, then every feasible solution has $x_j = 0$ and we can simply delete x_j from the instance.

Now we give our iterated rounding method. Let the term *entry* mean a pair $(i, j) \in I \times J$ such that $A_{ij} > 0$. Our iterated rounding algorithm computes a set S of *special* entries; for such a set we let $A_{S \to 0}$ denote the matrix obtained from A by zeroing out the special entries.

Lemma 12 *Given a k-CS PIP $\mathcal{P}$, we can, in polynomial time, find S and nonnegative integral vectors x^0, x^1 with $x^0 + x^1 \leq d$ and $x^1 \leq \mathbf{1}$ such that*

(a) $c \cdot (x^0 + x^1) \geq \mathrm{OPT}(\mathcal{L}(\mathcal{P}))$
(b) $\forall i \in I$, *we have* $|\{j : (i, j) \in S\}| \leq k$
(c) $Ax^0 + A_{S \to 0} x^1 \leq b$.

In particular, since x^1 is 0-1, $(x^0 + x^1)$ is a solution such that for each row i, we have $a_i \cdot (x^0 + x^1) \leq b_i + k \max_j A_{ij}$. We now give the proof of the above lemma.

Proof of Lemma 12 First, we give a sketch. Recall that A_j denote the jth column of A and a_i denotes the ith row of A. Let $\mathtt{supp}(A_j) := \{i \in I \mid A_{ij} > 0\}$, which has size at most k, and similarly $\mathtt{supp}(a_i) := \{j \in J \mid A_{ij} > 0\}$. Let x^* be an extreme

ITERATEDSOLVER(A, b, c, d)

1: Let x^* be an extreme optimum of $\{\max cx \mid x \in \mathbf{R}^J; \mathbf{0} \le x \le d; Ax \le b\}$
2: Let $x^0 = \lfloor x^* \rfloor, x^1 = \mathbf{0}, J' = \{j \in J \mid x^*_j \notin \mathbb{Z}\}, I' = I, S = \varnothing$.
3: **loop**
4: Let x^* be an extreme optimum of $\{\max cx \mid x \in [0,1]^{J'}; Ax^0 + A_{S\to 0}(x + x^1) \le b\}$
5: For each $j \in J'$ with $x^*_j = 0$, delete j from J'
6: For each $j \in J'$ with $x^*_j = 1$, set $x^1_j = 1$ and delete j from J'
7: If $J' = \varnothing$, terminate and return S, x^0, x^1
8: **for** each $i \in I'$ with $|\,\mathtt{supp}(a_i) \cap J'| \le k$ **do**
9: Mark each entry $\{(i, j) \mid j \in \mathtt{supp}(a_i) \cap J'\}$ special and add it in S and delete i from I'
10: **end for**
11: **end loop**

Fig. 1 Algorithm for k-CS PIP

optimal solution to $\mathcal{L}(\mathcal{P})$. The crux of our approach is as follows: if x^* has integral values we have made progress. If not, x^* is a *basic feasible solution* so there is a set of $\mathtt{supp}(x^*) = |J|$ linearly independent tight constraints for x^*, so the total number of constraints $|I|$ satisfies $|I| \ge |J|$. By double-counting there is some $i \in I$ with $|\,\mathtt{supp}(a_i)| \le k$, which is what permits iterated relaxation: we discard the constraint for i and go back to the start.

Figure 1 contains pseudocode for our iterated rounding algorithm, ITERATED-SOLVER.

Now we explain the pseudocode. The x^0 term can be thought of as a preprocessing step which effectively reduces the general case to the special case that $d = \mathbf{1}$. The term $x^1 \in \{0,1\}^J$ grows over time. The set J' represents all j that could be added to x^1 in the future, but have not been added yet. The set I' keeps track of constraints that have not been dropped from the linear program so far.

Since x^* is a basic feasible solution we have $|I'| \ge |J'|$ in Step 8. Being k-CS, each set $|\,\mathtt{supp}(A_j) \cap I'|$ for $j \in J'$ has size at most k. By double-counting, $\sum_{i\in I'} |\,\mathtt{supp}(a_i) \cap J'| \le k|J'| \le k|I'|$ and so some $i \in I'$ has $|\,\mathtt{supp}(a_i) \cap J'| \le k$. Thus $|I'|$ decreases in each iteration, and the algorithm has polynomial running time. (In fact, it is not hard to show that there are at most $O(k \log |I|)$ iterations.)

The algorithm has the property that $c \cdot (x^0 + x^1 + x^*)$ does not decrease from one iteration to the next, which implies property (a). Properties (b) and (c) can be seen immediately from the definition of the algorithm. □

Now we give the proof of the main result in this section. Here and later we abuse notation and identify vectors in $\{0,1\}^J$ with subsets of J, with 1 representing containment. That is, if we have two 0, 1 vectors y and x we let $y \subset x$ denote the fact that $y_i = 1$ implies $x_i = 1$.

Theorem 2 *There is a polynomial time $(2k^2 + 2)$-approximation algorithm for k-CS PIPs with multiplicity constraints.*

Proof We use Lemma 12 to obtain x^0 and x^1. The main idea in the proof is to partition the set x^1 into $2k^2+1$ sets which are all feasible (i.e., we get $x^1 = \sum_{j=1}^{2k^2+1} y^j$ for 0-1 vectors y^j each with $Ay^j \le b$). If we can establish the existence of such a partition, then we are done as follows: the total profit of the $2k^2+2$ feasible solutions $x^0, y^1, \ldots, y^{2k^2+1}$ is $c \cdot (x^0 + x^1) \ge \mathrm{OPT}$, so the most profitable is a $(2k^2+2)$-approximately optimal solution.

Call $j, j' \in x^1$ *in conflict at* i if $A_{ij} > 0$, $A_{ij'} > 0$ and at least one of (i, j) or (i, j') is special. We claim that if $y \subset x^1$ and no two elements of y are in conflict, then y is feasible; this follows from Lemma 12(c) together with the fact that $A_{ij} \le b_i$ for all i, j. (Explicitly, for each constraint we either just load it with a single special entry, or all non-special entries, both of which are feasible.) In the remainder of the proof, we find a $(2k^2+1)$-colouring of the set x^1 such that similarly-coloured items are never in conflict; then the colour classes give the needed sets y^j and we are done.

To find our desired colouring, we create a *conflict digraph* which has node set x^1 and an arc (directed edge) from j to j' whenever j, j' are in conflict at i and (i, j) is special. Rewording, there is an arc (j, j') iff some $(i, j) \in S$ and $A_{ij'} > 0$. (If (i, j') is also special, this also implies an arc (j', j).) The key observation is that each node $j \in x^1$ has indegree bounded by k^2, i.e. there are at most k^2 choices of j such that (j, j') is an arc: to see this note $\#\{i \mid A_{ij'} > 0\} \le k$, and each i in this set has $\#\{j \mid (i, j) \in S\} \le k$. Now we use the following lemma, which completes the proof.

Lemma 13 *A digraph with maximum indegree d has a $2d+1$-colouring.*

Proof We use induction on the number of nodes in the graph, with the base case being the empty graph. Now suppose the graph is nonempty. The average indegree is at most d, and the average indegree equals the average outdegree. Hence some node n has outdegree at most the average, which is d. In total, this node has at most $2d$ neighbours. By induction there is a $(2d+1)$-colouring when we delete n, then we can extend it to the whole digraph by assigning n any colour not used by its neighbours. □

(We remark that Lemma 13 is tight, e.g. arrange $2d+1$ vertices on a circle and include an arc from each vertex to its d clockwise-next neighbours; this directed K_{2d+1} cannot be $2d$-coloured.) This ends the proof of Theorem 2. □

3.1 Improvements for $k = 2$

We give some small improvements for the case $k = 2$, using some insights due to Shepherd & Vetta [36]. A 2-CS PIP is *non-simple* if there exist distinct j, j' with $\mathtt{supp}(A_j) = \mathtt{supp}(A_{j'})$ and $|\mathtt{supp}(A_j)| = 2$. Otherwise, it is simple. Shepherd and Vetta consider the case when all non-zero entries of a column are equal. Under this assumption, they get a 3.5 approximation for 2-CS PIPs, and a $\frac{11}{2} - \sqrt{5} \approx 3.26$ approximation for simple 2-CS PIPs, when $d = \mathbf{1}$. We extend their theorem as follows.

Theorem 14 *There is a deterministic 4-approximation algorithm for 2-CS PIPs. There is also a randomized $6 - \sqrt{5} \approx 3.764$-approximation algorithm for simple 2-CS PIPs with $d = \mathbf{1}$.*

(Sketch) Since we are dealing with a 2-CS PIP, each $\texttt{supp}(A_j)$ is an edge or a loop on vertex set I; we abuse notation and directly associate j with an edge/loop. Consider the initial value of J', i.e. after executing Step 2. Then we claim that the graph (I, J') has at most one cycle per connected component; to see this, note that any connected component with two cycles would have more edges than vertices, which contradicts the linear independence of the tight constraints for the initial basic solution x^*.

We modify ITERATEDSOLVER slightly. Immediately after Step 2, let $M \subset J'$ consist of one edge from each cycle in (I, J'), and set $J' := J' \backslash M$. Then M is a matching (hence a feasible 0-1 solution) and the new J' is acyclic. Modify the cardinality condition in Step 8 to $|\texttt{supp}(a_i) \cap J'| \leq 1$ (instead of ≤ 2); since J' is acyclic, it is not hard to show the algorithm will still terminate, and $\forall i \in I$, we have $|\{j : (i, j) \in S\}| \leq 1$.

To get the first result, we use a colouring argument from [36, Thm. 4.1] which shows that x^1 can be decomposed into two feasible solutions $x^1 = y^1 + y^2$. We find that the most profitable of x^0, M, y^1, y^2 is a 4-approximately optimal solution.

For the second result, we instead apply a probabilistic technique from [36, §4.3]. They define a distribution over subsets of the forest x^1; let z be the random variable indicating the subset. Let $p = \frac{1}{20}(5+\sqrt{5})$. Say that an edge ii' is *compatible* with z if z neither contains an edge with a special endpoint at i, nor at i'. The distribution has the properties that z is always feasible for the PIP, $\Pr[j \in z] = p$ for all $j \in x^1$, and $\Pr[\texttt{supp}(A_j)$ compatible with $z] \geq p$ for all $j \in x^0$. (Simplicity implies that x^0 and x^1 have no edge in common, except possibly loops, which is needed here.) Finally, let w denote the subset of x^0 compatible with z. Then $z + w$ is a feasible solution, and $\mathrm{E}[c(z + w)] \geq pc(x^1 + x^0)$. Hence the better solution of $z + w$ and M is a $1 + 1/p = (6 - \sqrt{5})$-approximately optimal solution. □

3.2 Improvements for High Width

The *width* W of an integer program is $\min_{ij} b_i/A_{ij}$, taking the inner term to be $+\infty$ when $A_{ij} = 0$. Note that without loss of generality, $W \geq 1$. From now on let us normalize each constraint so that $b_i = 1$; then a program has width $\geq W$ iff every entry of A is at most $1/W$.

In many settings better approximation can be obtained as W increases. For example in k-RS CIPs with $b = \mathbf{1}$, the sum of each row of A is at most k/W, so Propositions 6 and 7 give a $(1 + k/W)$-approximation algorithm. Srinivasan [38, 39] gave a $(1+\ln(1+k)/W)$-approximation algorithm for unbounded k-CS CIPs. Using *grouping and scaling* techniques introduced by Kolliopoulos and Stein [25], Chekuri et al. [7] showed that no-bottleneck demand multicommodity flow in a tree, and certain other problems, admit approximation ratio $1 + O(1/\sqrt{W})$. Multicommodity flow in a tree (without demands) admits approximation ratio $1 + O(1/W)$ [27]. Motivated by these results, we will prove the following theorem.

Theorem 15 *There is a polynomial time* $1 + \frac{2k}{W-k}$*-approximation algorithm to solve* k*-column-sparse PIPs with* $W > k$.

For $W \geq 2k$, Theorem 15 implies a $1 + O(k/W)$-approximation. For fixed $k \geq 4$ and large W this is asymptotically tight since $1 + o(1/W)$-approximation is NP-

hard, by results from [11, 27] on multicommodity flows in trees. After the initial publication of Theorem 15 [34], Bansal et al. [1] gave an algorithm with ratio $16e \cdot k^{1/\lfloor W \rfloor}$, where $e = 2.718\ldots$. Note that Theorem 15 is still the best ratio known for some choices of k and W, for example when $W \geq 3k$, or when $W \geq 1.1 \cdot k$ and $k \geq 8$.

Proof of Theorem 15 Run ITERATEDSOLVER. From Lemma 12 we see that $c \cdot (x^0 + x^1) \geq \text{OPT}$ and, using the width bound,

$$A(x^0 + x^1) \leq (1 + k/W)\mathbf{1}. \tag{4}$$

Define $\mathcal{V}(x)$ by $\mathcal{V}(x) := \{i \in I \mid a_i \cdot x > 1\}$, e.g. the set of violated constraints in $Ax \leq \mathbf{1}$.

We want to reduce $(x^0 + x^1)$ so that no constraints are violated. In order to do this we employ a linear program. Let $\chi(\cdot)$ denote the characteristic vector. Our LP, which takes a parameter $\widehat{x}$, is

$$\mathcal{R}(\widehat{x}) : \max\left\{cx \mid \mathbf{0} \leq x \leq \widehat{x}, Ax \leq \mathbf{1} - \frac{k}{W}\chi(\mathcal{V}(\widehat{x}))\right\}. \tag{5}$$

We can utilize this LP in an iterated rounding approach, described by the following pseudocode.

ITERATEDREDUCER
1: Let $\widehat{x} := x^0 + x^1$
2: **while** $\mathcal{V}(\widehat{x}) \neq \varnothing$ **do**
3: Let x^* be an extreme optimum of $\mathcal{R}(\widehat{x})$
4: Let $\widehat{x} = \lceil x^* \rceil$
5: **end while**

We claim that this algorithm terminates, and that the value of $c\widehat{x}$ upon termination is at least

$$\frac{1 - k/W}{1 + k/W} c \cdot (x^0 + x^1) \geq \frac{1 - k/W}{1 + k/W} \text{OPT}.$$

Once we show these facts, we are done, since the for the final $\widehat{x}$, $\mathcal{V}(\widehat{x}) = \varnothing$ implies $\widehat{x}$ is feasible. As an initial remark, note that each coordinate of $\widehat{x}$ is monotonically nonincreasing, and so $\mathcal{V}(\widehat{x})$ is also monotonically nonincreasing.

Observe that $\mathcal{R}$ in the first iteration has $\frac{1-k/W}{1+k/W}(x^0 + x^1)$ as a feasible solution, by (4). Next, note that x which is feasible for $\mathcal{R}$ in one iteration is also feasible for $\mathcal{R}$ in the next iteration since $\mathcal{V}(\widehat{x})$ is monotonically nonincreasing; hence the value of $c \cdot x^*$ does not decrease between iterations.

To show the algorithm terminates, we will show that $\mathcal{V}(\widehat{x})$ becomes strictly smaller in each iteration. Note first that if $i \notin \mathcal{V}(\widehat{x})$, the constraint $a_i \cdot x \leq 1$ is already implied by the constraint $x \leq \widehat{x}$. Hence $\mathcal{R}(\widehat{x})$ may be viewed as having only $|\mathcal{V}(\widehat{x})|$ many constraints other than the box constraints $0 \leq x \leq \widehat{x}$. Then x, a basic feasible solution to $\mathcal{R}(\widehat{x})$, must have at most $|\mathcal{V}(\widehat{x})|$ non-integral variables. In particular, using the fact that the program is k-CS, by double counting, there exists some $i \in \mathcal{V}(\widehat{x})$ such that $\#\{j \mid x^*_j \notin \mathbb{Z}, A_{ij} > 0\} \leq k$. Thus (using the fact that all entries of A are at

most $1/W$) we have $a_i \cdot \lceil x^* \rceil < a_i \cdot x^* + k(1/W) \leq 1$: so $i \notin \mathcal{V}(\lceil x^* \rceil)$, and $\mathcal{V}(\widehat{x})$ is strictly smaller in the next iteration, as needed. □

4 Hardness of Column-Restricted 2-CS CIP

Theorem 3 *It is* NP*-hard to approximate* 2*-CS CIPs of the form* $\{\min cx \mid Ax \geq b, x \text{ is } 0\text{-}1\}$ *and* $\{\min cx \mid Ax \geq b, x \geq 0, x \text{ integral}\}$ *within ratio* $17/16 - \varepsilon$ *even if the nonzeroes of every column of* A *are equal and* A *is of the block form* $\left[\begin{smallmatrix} A_1 \\ A_2 \end{smallmatrix}\right]$ *where each* A_i *is* 1*-CS.*

Proof Our proof is a modification of a hardness proof from [6] for a budgeted allocation problem. We focus on the version where x is 0-1; the other version follows similarly with only minor modifications to the proof. The specific problem described in the statement of the theorem is easily seen equivalent to the following problem, which we call *demand edge cover in bipartite multigraphs*: given a bipartite multigraph (V, E) where each vertex v has a demand b_v and each edge e has a cost c_e and value d_e, find a minimum-cost set E' of edges so that for each vertex v its demand is satisfied, meaning that $\sum_{e \in E' \cap \delta(v)} d_e \geq b_v$. Our construction also has the property that $c_e = d_e$ for each edge—so from now on we denote both d_e.

The proof uses a reduction from Max-3-Lin(2), which is the following optimization problem: given a collection $\{x_i\}_i$ of 0-1 variables and a family of three-variable modulo-2 equalities called *clauses* (for example, $x_1 + x_2 + x_3 \equiv 1 \pmod 2$), find an assignment of values to the variables which satisfies the maximum number of clauses. Håstad [15] showed that for any $\varepsilon > 0$, it is NP-hard to distinguish between the two cases that (1) a $(1 - \varepsilon)$ fraction of clauses can be satisfied and (2) at most a $(1/2 + \varepsilon)$ fraction of clauses can be satisfied.

Given an instance of Max-3-Lin(2) we construct an instance of demand edge cover as follows. For each variable x_i there are three vertices "x_i", "$x_i = 0$" and "$x_i = 1$"; these vertices have b-value $4\deg(x_i)$ where $\deg(x_i)$ denotes the number of clauses containing x_i. For each clause there are four vertices labelled by the four assignments to its variables that do *not* satisfy it; for example for the clause $x_1 + x_2 + x_3 \equiv 1 \pmod 2$ we would introduce four vertices, one of which would be named "$x_1 = 0, x_2 = 0, x_3 = 0$." These vertices have b-value equal to 3. Each vertex "$x_i = C$" is connected to "x_i" by an edge with d-value $4\deg(x_i)$; each vertex v of the form "$x_{i_1} = C_1, x_{i_2} = C_2, x_{i_3} = C_3$" is incident to a total of nine edges each with d-value 1: three of these edges go to "$x_{i_j} = C_j$" for each $j = 1, 2, 3$. The construction is illustrated in Fig. 2.

Let m denote the total number of clauses; so $\sum_i \deg(x_i) = 3m$. We claim that the optimal solution to this demand edge cover instance has cost $24m + 3t$ where t is the least possible number of unsatisfied clauses for the underlying Max-3-Lin(2) instance. If we can show this then we are done since Håstad's result shows we cannot distinguish whether the optimal cost is $\geq 24m + 3m(1/2 - \varepsilon)$ or $\leq 24m + 3(\varepsilon m)$; this gives an inapproximability ratio of $\frac{24+3/2-3\varepsilon}{24+3\varepsilon} = 17/16 - \varepsilon'$ for some $\varepsilon' > 0$ such that $\varepsilon' \to 0$ as $\varepsilon \to 0$, which will complete the proof.

Let x^* denote a solution to the Max-3-Lin(2) instance with t unsatisfied clauses; we show how to obtain a demand edge cover E' of cost $24m + 3t$. We include in E' the

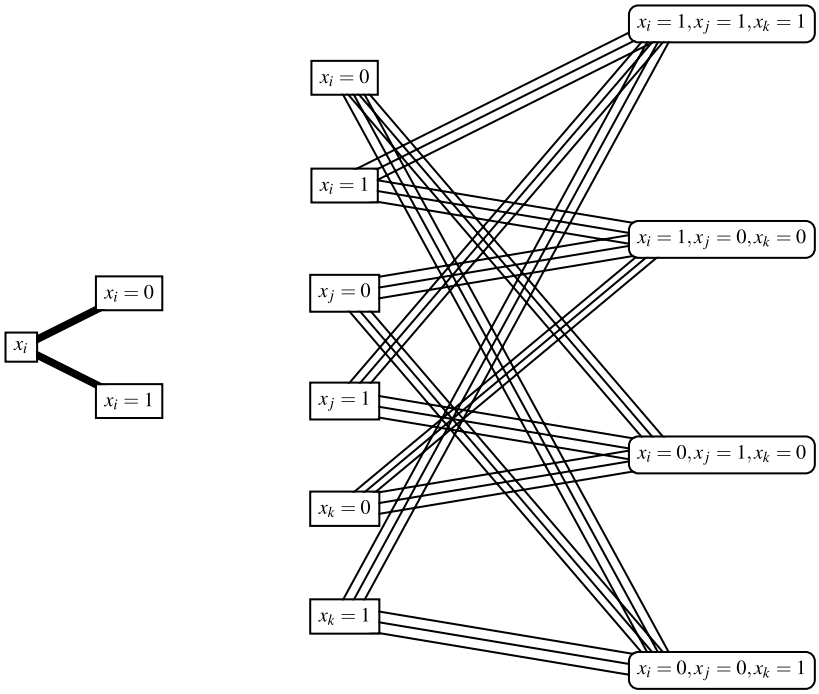

Fig. 2 *Left*: the gadget constructed for each variable x_i. The vertices shown as rectangles have b-value $4\deg(x_i)$; the thick edges have d-value and cost $4\deg(x_i)$. *Right*: the gadget constructed for the clause $x_i + x_j + x_k \equiv 0 \pmod 2$. The vertices shown as rounded boxes have b-value 3; the thin edges each have unit d-value and cost

edge between "x_i" and "$x_i = x_i^*$" for each i; this has total cost $\sum_i 4\deg(x_i) = 12m$. For each satisfied clause $x_i + x_j + x_k \equiv C \pmod 2$, we include in E' all three edges between "$x_i = 1 - x_i^*$" and "$x_i = 1 - x_i^*, x_j = x_j^*, x_k = x_k^*$" and similarly for j, k, and one of each of the parallel triples incident to "$x_i = 1 - x_i^*, x_j = 1 - x_j^*, x_k = 1 - x_k^*$"; this has cost 12 for that clause. For each unsatisfied clause $x_i + x_j + x_k \equiv C \pmod 2$, we include in E' any three unit-cost edges incident to "$x_i = x_i^*, x_j = x_j^*, x_k = x_k^*$," as well as twelve more unit-cost edges: namely in the six nodes consisting of "$x_i = 1 - x_i^*$," "$x_i = 1 - x_i^*, x_j = 1 - x_j^*, x_k = x_k^*$" and their images under swapping i with j and k, the induced subgraph is a 6-cycle of parallel triples, and we take two edges out of each triple. Thus the chosen edges have total cost 15 for that clause. It is not hard to see that this solution is feasible—e.g. vertices of the form "$x_i = 1 - x_i^*$" are covered by 4 edges for each clause containing them. The total cost is $c(E') = 12m + 12(m - t) + 15t = 24m + 3t$.

To finish the proof we show the following.

Claim 16 *Given a feasible demand edge cover E', we can find a solution x^* such that t, the number of unsatisfied clauses for x^*, satisfies $24m + 3t \le c(E')$.*

Proof First we claim it is without loss of generality that for each i, E' contains exactly one of the edges incident to "x_i". Clearly at least one of these two edges lies in E'; if both do, then remove one (say, the edge between "x_i" and "$x_i = 0$") and add to E' any subset of the other $6\deg(x_i)$ edges incident to "$x_i = 0$" so that the total number of edges incident on "$x_i = 0$" in E' becomes at least $4\deg(x_i)$. The removed edge has d-value $4\deg(x_i)$ and all other incident edges have d-value 1, so clearly the solution is still feasible and the cost has not increased.

Define x^* so that for each i, E' contains the edge between "x_i" and "$x_i = x_i^*$." Let E'' denote the edges of E' incident on clause vertices (i.e. the edges of E' with unit d-value). For $F \subset E''$ their *left-contribution*, denoted $\ell(F)$, is the number of them incident on vertices of the form "$x_i = 1 - x_i^*$." Note that $\ell(F) \leq |F|$ for any F. Furthermore for each unsatisfied clause, all edges incident on its vertex "$x_i = x_i^*, x_j = x_j^*, x_k = x_k^*$" have zero left-contribution, but E' contains at least three of these edges. Thus the edges of E'' incident on that clause's vertices have $\ell(F) \leq |F| - 3$. Finally, consider $\ell(E'')$. Each edge of E'' is in the gadget for a particular clause, and it follows that $\ell(E'') \leq |E''| - 3t$ where t is the number of unsatisfied clauses for x^*. However, E'' needs to have $4\deg(x_i)$ edges incident on each "$x_i = 1 - x_i^*$" so $\ell(E'') \geq \sum_i 4\deg(x_i) = 12m$. Thus $|E''| \geq 12m + 3t$ and considering the edges incident on the vertices "x_i" we see that $c(E') \geq 24m + 3t$. □

This completes the proof of the reduction. □

5 Open Problems

It is natural to conjecture that k-CS CIP with a submodular objective admits an approximation ratio depending only on k, perhaps $O(\ln k)$ matching the best ratio known for linear objectives.

Although 2-RS IPs are very hard to optimize (at least as hard as Max Independent Set), the problem of finding a *feasible* solution to a 2-RS IP is still interesting. Hochbaum et al. [19] gave a pseudopolynomial-time 2-SAT-based feasibility algorithm for 2-RS IPs with finite upper and lower bounds on variables. They asked if there is a pseudopolynomial-time feasibility algorithm when the bounds are replaced by just the requirement of nonnegativity, which is still open as far as we know. It is strongly NP-hard to determine if IPs of the form $\{x \geq 0 \mid Ax = b\}$ are feasible when A is 2-CS [18], e.g. by a reduction from 3-Partition; but for IPs where each variable appears at most twice *including* in upper/lower bounds, it appears all that is known is weak NP-hardness (for example, via the *unbounded knapsack problem* [33]).

Theorem 15 shows that we can $(1 + \varepsilon)$-approximate a k-CS PIP when $W \geq k(2/\varepsilon + 1)$. It is an interesting open problem to find what other k-W regimes allow approximation ratio very close to 1.

Acknowledgement We would like to thank Glencora Borradaile, Christina Boucher, Stephane Durocher, Jochen Könemann and Christos Koufogiannakis for helpful discussions, and the ESA and Algorithmica referees for useful feedback.

References

1. Bansal, N., Korula, N., Nagarajan, V., Srinivasan, A.: On k-column sparse packing programs. In: Proc. 14th IPCO, pp. 369–382 (2010). Preliminary version at arXiv:0908.2256
2. Bar-Yehuda, R., Even, S.: A linear time approximation algorithm for the weighted vertex cover problem. J. Algorithms **2**, 198–203 (1981)
3. Bar-Yehuda, R., Rawitz, D.: Efficient algorithms for integer programs with two variables per constraint. Algorithmica **29**(4), 595–609 (2001)
4. Berman, P.: A $d/2$ approximation for maximum weight independent set in d-claw free graphs. Nord. J. Comput. **7**(3), 178–184 (2000). Preliminary version in Proc. 7th SWAT, pp. 214–219 (2000)
5. Carr, R.D., Fleischer, L., Leung, V.J., Phillips, C.A.: Strengthening integrality gaps for capacitated network design and covering problems. In: Proc. 11th SODA, pp. 106–115 (2000)
6. Chakrabarty, D., Goel, G.: On the approximability of budgeted allocations and improved lower bounds for submodular welfare maximization and GAP. In: Proc. 49th FOCS, pp. 687–696 (2008)
7. Chekuri, C., Mydlarz, M., Shepherd, F.B.: Multicommodity demand flow in a tree and packing integer programs. ACM Trans. Algorithms **3**(3), 27 (2007). Preliminary version in Proc. 30th ICALP, pp. 410–425 (2003)
8. Chekuri, C., Ene, A., Korula, N.: Unsplittable flow in paths and trees and column-restricted packing integer programs. In: Proc. 12th APPROX, pp. 42–55 (2009)
9. Dinur, I., Guruswami, V., Khot, S., Regev, O.: A new multilayered PCP and the hardness of hypergraph vertex cover. SIAM J. Comput. **34**(5), 1129–1146 (2005). Preliminary version in Proc. 35th STOC, pp. 595–601 (2003)
10. Fujito, T., Yabuta, T.: Submodular integer cover and its application to production planning. In: Proc. 2nd WAOA, pp. 154–166 (2004)
11. Garg, N., Vazirani, V.V., Yannakakis, M.: Primal-dual approximation algorithms for integral flow and multicut in trees. Algorithmica **18**(1), 3–20 (1997). Preliminary version in Proc. 20th ICALP, pp. 64–75 (1993)
12. Goel, G., Karande, C., Tripathi, P., Wang, L.: Approximability of combinatorial problems with multi-agent submodular cost functions. In: Proc. 50th FOCS, pp. 755–764 (2009)
13. Grötschel, M., Lovász, L., Schrijver, A.: Geometric Algorithms and Combinatorial Optimization. Springer, Berlin (1988)
14. Hall, N.G., Hochbaum, D.S.: A fast approximation algorithm for the multicovering problem. Discrete Appl. Math. **15**(1), 35–40 (1986)
15. Håstad, J.: Some optimal inapproximability results. J. ACM **48**(4), 798–859 (2001)
16. Hazan, E., Safra, S., Schwartz, O.: On the complexity of approximating k-set packing. Comput. Complex. **15**(1), 20–39 (2006). Preliminary versions in Proc. 6th APPROX, pp. 83–97 (2003)
17. Hochbaum, D.S.: Approximation algorithms for set covering and vertex cover problems. SIAM J. Comput. **11**, 555–556 (1982)
18. Hochbaum, D.S.: Monotonizing linear programs with up to two nonzeroes per column. Oper. Res. Lett. **32**(1), 49–58 (2004)
19. Hochbaum, D.S., Megiddo, N., Naor, J.S., Tamir, A.: Tight bounds and 2-approximation algorithms for integer programs with two variables per inequality. Math. Program. **62**(1), 69–83 (1993)
20. Hurkens, C.A.J., Schrijver, A.: On the size of systems of sets every t of which have an SDR, with an application to the worst-case ratio of heuristics for packing problems. SIAM J. Discrete Math. **2**(1), 68–72 (1989)
21. Iwata, S., Nagano, K.: Submodular function minimization under covering constraints. In: Proc. 50th FOCS, pp. 671–680 (2009)
22. Kellerer, H., Pferschy, U., Pisinger, D.: Knapsack Problems. Springer, Berlin (2004)
23. Khot, S., Ponnuswami, A.K.: Better inapproximability results for MaxClique, Chromatic Number and Min-3Lin-Deletion. In: Proc. 33rd ICALP, pp. 226–237 (2006)
24. Khot, S., Regev, O.: Vertex cover might be hard to approximate to within $2-\varepsilon$. J. Comput. Syst. Sci. **74**(3), 335–349 (2008). Preliminary version in Proc. 18th CCC, pp. 379–386 (2003)
25. Kolliopoulos, S.G., Stein, C.: Improved approximation algorithms for unsplittable flow problems. In: Proc. 38th FOCS, pp. 426–436 (1997)
26. Kolliopoulos, S.G., Young, N.E.: Approximation algorithms for covering/packing integer programs. J. Comput. Syst. Sci. **71**(4), 495–505 (2005)
27. Könemann, J., Parekh, O., Pritchard, D.: Max-weight integral multicommodity flow in spiders and high-capacity trees. In: Proc. 6th WAOA, pp. 1–14 (2008)

28. Koufogiannakis, C., Young, N.E.: Distributed and parallel algorithms for weighted vertex cover and other covering problems. In: Proc. 28th PODC, pp. 171–179 (2009)
29. Koufogiannakis, C., Young, N.E.: Distributed fractional packing and maximum weighted b-matching via tail-recursive duality. In: Proc. 23rd DISC, pp. 221–238 (2009)
30. Koufogiannakis, C., Young, N.E.: Greedy δ-approximation algorithm for covering with arbitrary constraints and submodular cost. In: Proc. 36th ICALP, pp. 634–652 (2009)
31. Lenstra, H.: Integer programming with a fixed number of variables. Math. Oper. Res. **8**, 538–548 (1983)
32. Lovász, L.: Submodular functions and convexity. In: Mathematical Programming: The State of the Art, pp. 235–257. Springer, Berlin (1983)
33. Lueker, G.S.: Two NP-complete problems in nonnegative integer programming. Tech. Rep. 178, Computer Science Laboratory, Princeton University (1975)
34. Pritchard, D.: Approximability of sparse integer programs. In: Proc. 17th ESA, pp. 83–94 (2009). Preliminary version at arXiv:0904.0859
35. Pritchard, D.: Linear programming tools & approximation algorithms for combinatorial optimization. Ph.D. thesis, University of Waterloo (2009)
36. Shepherd, F.B., Vetta, A.: The demand-matching problem. Math. Oper. Res. **32**(3), 563–578 (2007). Preliminary version in Proc. 9th IPCO, pp. 457–474 (2002)
37. Singh, M.: Iterative methods in combinatorial optimization. Ph.D. thesis, Carnegie Mellon University (2008)
38. Srinivasan, A.: Improved approximation guarantees for packing and covering integer programs. SIAM J. Comput. **29**(2), 648–670 (1999). Preliminary version in Proc. 27th STOC, pp. 268–276 (1995)
39. Srinivasan, A.: An extension of the Lovász Local Lemma, and its applications to integer programming. SIAM J. Comput. **36**(3), 609–634 (2006). Preliminary version in Proc. 7th SODA, pp. 6–15 (1996)
40. Trevisan, L.: Non-approximability results for optimization problems on bounded degree instances. In: Proc. 33rd STOC, pp. 453–461 (2001)
41. Wolsey, L.: An analysis of the greedy algorithm for the submodular set covering problem. Combinatorica **2**(4), 385–393 (1982)