Small Extended Formulation for Knapsack Cover Inequalities from Monotone Circuits

Abbas Bazzi
EPFL, Switzerland
abbas.bazzi@epfl.ch

Samuel Fiorini
Université libre de Bruxelles, Belgium
sfiorini@ulb.ac.be

Sangxia Huang
EPFL, Switzerland
huang.sangxia@gmail.com

Ola Svensson
EPFL, Switzerland
ola.svensson@epfl.ch

July 13, 2016

Abstract

Initially developed for the min-knapsack problem, the knapsack cover inequalities are used in the current best relaxations for numerous combinatorial optimization problems of covering type. In spite of their widespread use, these inequalities yield linear programming (LP) relaxations of exponential size, over which it is not known how to optimize exactly in polynomial time. In this paper we address this issue and obtain LP relaxations of quasi-polynomial size that are at least as strong as that given by the knapsack cover inequalities.

For the min-knapsack cover problem, our main result can be stated formally as follows: for any $\varepsilon > 0$, there is a $(1/\varepsilon)^{O(1)} n^{O(\log n)}$-size LP relaxation with an integrality gap of at most $2 + \varepsilon$, where n is the number of items. Prior to this work, there was no known relaxation of subexponential size with a constant upper bound on the integrality gap.

Our construction is inspired by a connection between extended formulations and monotone circuit complexity via Karchmer-Wigderson games. In particular, our LP is based on $O(\log^2 n)$-depth monotone circuits with fan-in 2 for evaluating weighted threshold functions with n inputs, as constructed by Beimel and Weinreb. We believe that a further understanding of this connection may lead to more positive results complementing the numerous lower bounds recently proved for extended formulations.

1 Introduction

Capacitated covering problems[1] play a central role in combinatorial optimization. These are the problems modeled by Integer Programs (IPs) of the form $\min\{\sum_{i=1}^n c_i x_i \mid Ax \geqslant b,\ x \in \{0,1\}^n\}$ where A is a size-$m \times n$ nonnegative matrix and b, c size-n nonnegative vectors. The *min-knapsack problem* is the special case arising when there is a single covering constraint, that is, when $m = 1$. This is arguably the simplest interesting capacitated covering problem.

In terms of complexity, the min-knapsack problem is well-understood: on the one hand it is weakly NP-hard [26] and on the other hand it admits an FPTAS [27, 32]. However, for its own sake and since it appears as a key substructure of numerous other IPs, improving our polyhedral understanding of the problem is important. By this, we mean finding "good" linear programming (LP) relaxations for the min-knapsack problem. Indeed, the polyhedral study of this problem has led to the development of important tools, such as the knapsack cover inequalities, for the strengthening of LP relaxations. These inequalities and generalizations thereof are now used in the current best known relaxations for several combinatorial optimization problems, such as single-machine scheduling [5] and capacitated facility location [1]. However, despite this important progress in the past, many fundamental questions remain open even in the most basic setting.

State of the Art. The feasible region of a min-knapsack instance is specified by positive *item sizes* $s_1, \ldots, s_n$ and a positive *demand* D. A vector $x \in \{0,1\}^n$ is *feasible* if $\sum_{i=1}^n s_i x_i \geqslant D$. To specify completely an instance of the min-knapsack problem, we are further given nonnegative *item costs* $c_1, \ldots, c_n$. Solving the resulting instance then amounts to solving the IP $\min\{\sum_{i=1}^n c_i x_i \mid \sum_{i=1}^n s_i x_i \geqslant D,\ x \in \{0,1\}^n\}$.

The *basic* LP relaxation $\min\{\sum_{i=1}^n c_i x_i \mid \sum_{i=1}^n s_i x_i \geqslant D,\ x \in [0,1]^n\}$ provides an estimate on the optimum value that can be quite bad. More precisely, defining the *integrality gap* as the supremum over all instances of the ratio of the optimum value of the IP to the optimum value of the LP relaxation, it is easy to see that the integrality gap is unbounded.

Several inequalities have been proposed for strengthening this basic LP relaxation. Already in the 70's, Balas [2], Hammer, Johnson and Peled [23] and Wolsey [35] independently proposed to add the *uncapacitated* knapsack cover inequalities: for every subset $A \subseteq [n]$ of the items such that $\sum_{i \in A} s_i < D$, add the inequality $\sum_{i \notin A} x_i \geq 1$ (saying that at least one item in $[n] \setminus A$ needs to be picked in order to satisfy the demand). Unfortunately, these (exponentially many) inequalities are not sufficient for bringing down the integrality gap to a constant. A strengthening of these inequalities was therefore proposed more recently by Carr, Fleischer, Leung and Philipps [13]. They defined the following valid inequalities: for every set of items $A \subseteq [n] := \{1, \ldots, n\}$ such that $\sum_{i \in A} s_i < D$, there is a corresponding (capacitated) *knapsack cover inequality*

$$\sum_{i \notin A} s'_i x_i \geqslant U \tag{1}$$

where $U = U(A) := D - \sum_{i \in A} s_i$ is the *residual demand* and $s'_i = s'_i(A) := \min\{s_i, U\}$. The validity of (1) is due to the fact that every feasible solution $x \in \{0,1\}^n$ has to contain some object $i \notin A$. This object can be *large*, that is, have $s_i \geqslant U$, and in this case the inequality is clearly satisfied. Otherwise, in case every object $i \notin A$ is *small*, the total size of the objects $i \notin A$ picked by x has to be at least the residual demand U.

Carr *et al.* [13] proved that whenever $x \in \mathbb{R}^n_{\geqslant 0}$ satisfies all knapsack cover inequalities, $2x$ dominates a convex combination of feasible solutions, that is, there exist feasible solutions $x^{(j)} \in \{0,1\}^n$ ($j \in [q]$) and coefficients $\lambda_j \geqslant 0$ summing up to 1 such that $2x \geqslant \sum_{j=1}^q \lambda_j x^{(j)}$. Given any nonnegative item costs,

[1]The term "capacitated" is used in the literature to emphasize that the entries of matrix A can take any non-negative value in contrast to the uncapacitated version where entries are Boolean.

one of the $x^{(j)}$ will have a cost that is at most 2 times that of x. This implies that the integrality gap of the corresponding LP relaxation is at most 2.

The LP relaxation defined by the knapsack cover inequalities is "good" in the sense that it has a constant integrality gap. However, it has exponential *size*, that is, exponentially many inequalities, over which it is not known how to optimize exactly in polynomial time.

In contrast, for the *max-knapsack problem*, Bienstock [9] proved that for all $\varepsilon > 0$ there exists a size-$n^{O(1/\varepsilon)}$ LP relaxation whose integrality gap[2] is at most $1+\varepsilon$. That LP is defined by an extended formulation that uses $n^{O(1/\varepsilon)}$ extra variables besides the x-variables. We remark that it is a notorious open problem to prove or disprove the existence of a $f(1/\varepsilon) \cdot n^{O(1)}$-size LP relaxation for max-knapsack with integrality gap at most $1+\varepsilon$, see e.g. the survey on extended formulations by Conforti, Cornuéjols and Zambelli [18]. Coming back to the min-knapsack problem, it is not known whether there exists a polynomial-size LP relaxation with constant integrality gap or not.[3]

Main Result. We come close to resolving the question and show that min-knapsack admits a quasi-polynomial-size LP relaxation with integrality gap at most $2+\varepsilon$. The upper bound on the integrality gap originates from the fact that our LP relaxation is at least as strong as that provided by a slightly weakened form of the knapsack cover inequalities. We point out that, under some conditions, we can bound the size of our relaxation by a polynomial, see Section 3.2. A more precise statement of our main result is as follows.

Theorem 1. *For all $\varepsilon \in (0,1)$, item sizes $s_1, \dots, s_n \in \mathbb{R}_+$ and demand $D \in \mathbb{R}_+$, there exists a size-$(1/\varepsilon)^{O(1)} n^{O(\log n)}$ extended formulation defining an LP relaxation of min-knapsack with integrality gap at most $2+\varepsilon$.*

As the result is obtained by giving quasi-polynomially many inequalities of roughly the same strength as the exponentially many knapsack cover inequalities, our techniques also lead to relaxations of quasi-polynomial size for the numerous applications of these inequalities. We mention some of these applications below when we discuss related works.

Beyond the result itself, the novelty of our approach lies in the concepts we rely on and the techniques we develop. Our starting point is a connection between monotone circuits and extended formulations that we explain below. This connection was instrumental in the recent *lower bounds* of Göös, Jain and Watson on the extension complexity of independent set polytopes [22], and can be traced back to a paper of Hrubeš [24]. Here we use it for the first time to prove an *upper bound*.

From Monotone Circuits to Extended Formulations. Each choice of item sizes and demand gives rise to a *weighted threshold function* $f : \{0,1\}^n \to \{0,1\}$ defined as

$$f(x) := \begin{cases} 1 & \text{if } \sum_{i=1}^n s_i x_i \geqslant D \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Since we assume that the item sizes and demand are nonnegative, f is *monotone* in the sense that $a \leqslant b$ implies $f(a) \leqslant f(b)$, for all $a, b \in \{0,1\}^n$.

[2]For maximization problems, one takes the supremum of the ratio of the optimum value of the LP relaxation to the optimum value of the IP.

[3]We remark that Bienstock and McClosky [10] considered the easier case when the relaxation is allowed to depend on the objective function to be optimized (i.e., on the cost of the items). In this case, using techniques similar to those developed for polynomial time approximation schemes, they obtained polynomial size relaxations with integrality gap at most $1+\varepsilon$, for any fixed $\varepsilon > 0$. This is, however, a very different setting and, as the developed inequalities depend on the objective function, they do not generalize to other problems.

Clearly, $x \in \{0,1\}^n$ is feasible if and only if $x \in f^{-1}(1)$. Furthermore, we can rewrite the uncapacitated knapsack cover inequalities as $\sum_{i:a_i=0} x_i \geqslant 1$ for $a \in f^{-1}(0)$. Consider the *slack matrix* $S_{a,b} := \sum_{i:a_i=0} b_i - 1$ indexed by pairs $(a,b) \in f^{-1}(0) \times f^{-1}(1)$. By Yannakakis' factorization theorem [36], the existence of a size-r LP relaxation of min-knapsack that is at least as strong as that given by the uncapacitated knapsack cover inequalities is equivalent to the existence of a decomposition of the slack matrix S as a sum of r nonnegative rank-1 matrices.

Now suppose that there exists a depth-t monotone circuit (that is, using only AND gates and OR gates) of fan-in 2 for computing $f(x)$. By a result of Karchmer and Wigderson [25], this implies a partition of the entries of S into at most 2^t rectangles[4] $R \subseteq f^{-1}(0) \times f^{-1}(1)$ such that in each one of these rectangles R, there exists some index $i^* = i^*(R)$ such that $a_{i^*} = 0$ and $b_{i^*} = 1$ for all $(a,b) \in R$. Then we may write, for $(a,b) \in R$,

$$S_{a,b} = \sum_{i:a_i=0} b_i - 1 = \sum_{i:a_i=0,\ i\neq i^*} b_i = \sum_{i\neq i^*} (1-a_i)b_i \tag{3}$$

so that S restricted to the entries of R can be expressed as a sum of at most $n-1$ nonnegative rank-1 matrices of the form $((1-a_i)b_i)_{(a,b)\in R}$, where i is a fixed index distinct from i^*. This implies a decomposition of the whole slack matrix S as a sum of at most $2^t(n-1)$ nonnegative rank-1 matrices, and thus the existence of a $2^t(n-1)$-size LP relaxation of min-knapsack that captures the uncapacitated knapsack cover inequalities. Since f is a weighted threshold function, we can take $t = O(\log^2 n)$, as proved by Beimel and Weinreb [8]. Therefore, we obtain a $n^{O(\log n)}$-size extended formulation for the uncapacitated knapsack cover inequalities. Unfortunately, these inequalities do not suffice to guarantee a bounded integrality gap.

For the full-fledged knapsack cover inequalities (1), the simple idea described above breaks down. If the special index $i^* = i^*(R)$ for some rectangle R corresponds to a large object, we can write

$$\sum_{i:a_i=0} s'_i b_i - U = \sum_{i:a_i=0,\ i\neq i^*} s'_i b_i = \sum_{i\neq i^*} s'_i (1-a_i) b_i$$

where each matrix $(s'_i(1-a_i)b_i)_{(a,b)\in R}$ has rank at most 1 because $s'_i(1-a_i)$ depends on a only. However, i^* may correspond to a small object, in which case we cannot decompose the slack matrix as above.

Nevertheless, we prove that it is possible to overcome this difficulty. Two key ideas we use to achieve this are to discretize some of the quantities (which explains why we lose an ε in the integrality gap) and resort to several weighted threshold functions instead of just one. If all these functions admit $O(\log n)$-depth monotone circuits of fan-in 2, then we obtain a size-$n^{O(1)}$ LP relaxation.

Related Works. Knapsack cover inequalities and their generalizations such as flow cover inequalities were used as a systematic way to *strengthen* LP formulations of other (seemingly unrelated) problems [13, 12, 29, 3, 4, 14, 5, 17, 19]. By strengthening we mean that one would start with a polynomial size LP formulation with a potentially unbounded integrality gap for some problem of interest, and then show that adding (adaptations) of knapsack cover inequalities reduces this integrality gap (we illustrate in Appendix A how this strengthening works for the Single Demand Facility Location problem, reducing the integrality gap down to 2). However, similar to the case of min-knapsack discussed above, the drawback of this approach is that the size of the resulting LP formulation becomes exponential. We can extend our result to show that it yields quasi-polynomial size LP formulation for many such applications. To name a few:

- Carr *et al.* [13] applied these inequalities to the Generalized Vertex Cover problem, Multi-color Network Design problem and the Fixed Charge Flow problem, and showed how these inequalities reduce the integrality gap of the starting LP formulations.

[4] A *rectangle* is the Cartesian product of a set of row indices and a set of column indices.

- Bansal and Pruhs [5] studied the Generalized Scheduling Problem (GSP) that captures many interesting scheduling problems such as Weighted Flow Time, Flow Time Squared and Weighted Tardiness. In particular, they showed a connection between GSP and a certain geometric covering problem, and designed an LP based approximation algorithm for the later that yields an approximate solution for the GSP. The LP formulation that they use for the intermediate geometric cover problem is strengthened using knapsack cover inequalities, and yields an $O(\log \log nP)$-approximation for the GSW where n is the number of jobs, and P is the maximum job size. In the special case of identical release time of the jobs, their LP formulation yields a 16-approximation algorithm. This constant factor approximation was later improved by Cheung and Shmoys [17] and Mestre and Verschae [30] to a $(4 + \varepsilon)$-approximation, where the authors added the knapsack cover inequalities directly to the LP formulation of the scheduling problem, i.e., without resorting to the intermediate geometric cover problem as in [5].[5]

- Efsandiari *et al.* [19] used a knapsack-cover-strengthened LP formulation to design an $O(\log k)$-approximation algorithm for Precedence-Constrained Single-Machine Deadline scheduling problem, where k is the number of distinct deadlines.

- Carnes and Shmoys [12] designed primal-dual algorithms for the Single-Demand Facility Location, where the primal LP formulation is strengthened by adding (generalizations) of knapsack cover inequalities.

Extended formulations have received a considerable amount of attention recently, mostly for proving impossibility results. Pokutta and Van Vyve [33] proved a worst-case $2^{\Omega(\sqrt{n})}$ size lower bound for extended formulations of the max-knapsack polytope, which directly implies a similar result for the min-knapsack polytope. Other recent works include [21, 11, 15, 34, 28, 6].

Outline. We prove our main result in Section 3, after giving some preliminaries in Section 2. Instead of explicitly constructing our extended formulation, we provide a nonnegative factorization of the appropriate slack matrix. For this, we use the language of communication complexity — we give an $O(\log^2 n + \log(1/\varepsilon))$-complexity two-party communication protocol with private randomness and nonnegative outputs whose expected output is the slack of a given feasible solution with respect to a given (weakened) knapsack cover inequality.

Next, in Appendix A, we extend our communication protocol to the flow cover inequalities for the Single-Demand Facility Location problem, and show how to approximate the exponentially many flow cover inequalities using a smaller LP formulation.

Finally, in Appendix B, we show that although we do not know how to write down our extended formulation for min-knapsack in quasi-polynomial time, we can at least compute a $(2 + \varepsilon)$-approximation of the optimum from the extended formulation in quasi-polynomial time, given any cost vector, *without* relying on the ellipsoid algorithm. This is done via a new cutting-plane algorithm that might be of independent interest.

2 Preliminaries

In this section, we introduce some key notions related to our problem. We review extended formulations and extension complexity of pairs of polyhedra in Section 2.1. Next, we define randomized communication protocols with non-negative outputs that compute entries of matrices in expectation. Finally, in Section 2.3,

[5] For both the GSP and its special case, our method yields an LP formulation whose size is quasi-polynomial in n, and polynomial in both $\log P$ and $\log W$, where W is the maximum increase in the cost function of a job at any point in time.

we review some constructions of low-depth monotone circuits, and the Karchmer-Wigderson game that relates circuit complexity and communication complexity.

2.1 Polyhedral Pairs, Extended Formulations and Slack Matrices

Let $P \subseteq \mathbb{R}^n$ be a polytope and $Q \subseteq \mathbb{R}^n$ be a polyhedron containing P. The complexity of the *polyhedral pair* (P, Q) can be measured by its extension complexity, which roughly measures how compactly we can represent a relaxation of P contained in Q. The formal definition is as follows.

Definition 1. *Given a polyhedral pair (P, Q) where $P \subseteq Q \subseteq \mathbb{R}^n$, we say that a system $E^{\leqslant}x + F^{\leqslant}y \leqslant g^{\leqslant}$, $E^{=}x + F^{=}y = g^{=}$ in $\mathbb{R}^{n+k}$ is an* extended formulation *of (P, Q) if the polyhedron $R := \{x \in \mathbb{R}^n \mid \exists y \in \mathbb{R}^k : E^{\leqslant}x + F^{\leqslant}y \leqslant g^{\leqslant}, E^{=}x + F^{=}y = g^{=}\}$ contains P and is contained in Q. The* size *of the extended formulation is the number of inequalities in the system. The* extension complexity *of (P, Q), denoted by $\mathrm{xc}(P, Q)$, is the minimum size of an extended formulation of (P, Q).*

Although the case $P = Q$ is probably the most frequent, we will need polyhedral pairs here. In a seminal paper, Yannakakis [36] showed that one can study the extension complexity of a polytope P through the non-negative rank of a matrix associated with P, namely, its slack matrix.

Definition 2. *Let (P, Q) be a polyhedral pair with $P \subseteq Q \subseteq \mathbb{R}^n$. Let $P = \mathrm{conv}(\{v_1, \ldots, v_p\})$ be an inner description of P and $Q = \{x \in \mathbb{R}^n \mid Ax \geqslant b\}$ be an outer description of Q, where $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. We now define the slack matrix S of the pair (P, Q) with respect to the given representations of P and Q. The ith row of S corresponds to the constraint $A_i x \geqslant b_i$, while the jth column of S corresponds to the point v_j. The value $S_{i,j}$ measures how close the constraint $A_i x \geqslant b_i$ is to being tight for point v_j. More specifically, the* slack matrix *$S \in \mathbb{R}^{m \times p}_{\geqslant 0}$ is defined as $S_{i,j} := A_i v_j - b_i$ for all $i \in [m], j \in [p]$.*

Note that the slack matrix is not unique as it depends on the choices of points $v_1, \ldots, v_p$ and linear description $Ax \geqslant b$.

Definition 3. *Given a non-negative matrix $M \in \mathbb{R}^{m \times n}_{\geqslant 0}$, we say that a pair of matrices T, U is a* rank-r non-negative factorization *of M if $T \in \mathbb{R}^{m \times r}_{\geqslant 0}$, $U \in \mathbb{R}^{r \times n}_{\geqslant 0}$, and $M = TU$. We define the* non-negative rank *of M as $\mathrm{rk}_+(M) := \min\{r : M$ has a rank-r non-negative factorization$\}$. Notice that a non-negative factorization of M of rank at most r is equivalent to a decomposition of M as a sum of at most r non-negative rank-1 matrices.*

Yannakakis [36] proved that for a polytope P of dimension at least 1 and any of its slack matrices S, the extension complexity of P is equal to the non-negative rank of S. Namely, $\mathrm{xc}(P) = \mathrm{rk}_+(S)$. In particular, all the slack matrices of P have the same nonnegative rank. This *factorization theorem* can be extended to polyhedral pairs: we have $\mathrm{xc}(P, Q) \in \{\mathrm{rk}_+(S), \mathrm{rk}_+(S) - 1\}$ whenever S is a slack matrix of (P, Q), see e.g. [11].

2.2 Randomized Communication Protocols

We now define a certain two-party communication problem and relate it to the non-negative rank discussed earlier, following the framework in Faenza, Fiorini, Grappe and Tiwary [20].

Definition 4. *Let $S \in \mathbb{R}^{\mathcal{A} \times \mathcal{B}}_{\geqslant 0}$ be a non-negative matrix whose rows and columns are indexed by $\mathcal{A}$ and $\mathcal{B}$, respectively. Let Π be a communication protocol with private randomness between two players Alice and Bob. Alice gets an input $a \in \mathcal{A}$ and Bob gets an input $b \in \mathcal{B}$. They exchange bits in a pre-specified way according to Π, and at the end either one of the players outputs some* non-negative *number $\xi \in \mathbb{R}_{\geqslant 0}$. We say that Π* computes *S* in expectation *if for every a and b, the expectation of the output ξ equals $S_{a,b}$.*

The communication complexity of a protocol Π *is the maximum of the number of bits exchanged* between *Alice and Bob, over all pairs* $(a, b) \in \mathcal{A} \times \mathcal{B}$ *and the private randomness of the players. The size of the final output does not count towards the communication complexity of a protocol. The* communication complexity of S*, denoted* $R^{\mathrm{cc}}_{\exp}(S)$ *is the minimum communication complexity of a randomized protocol* Π *computing* S *in expectation.*

Faenza *et al.* [20] relate the non-negative rank of a non-negative matrix S, to the communication complexity $R^{\mathrm{cc}}_{\exp}(S)$. They prove that $R^{\mathrm{cc}}_{\exp}(S) = \log_2 \mathrm{rk}_+(S) + \Theta(1)$, provided that $\mathrm{rk}_+(S) \neq 0$. Combining this with the factorization theorem, we get $R^{\mathrm{cc}}_{\exp}(S) = \log_2 \mathrm{xc}(P, Q) + \Theta(1)$ whenever (P, Q) is a polyhedral pair with slack matrix S, provided that $\mathrm{xc}(P, Q) \neq 0$.

2.3 Weighted Threshold Functions and Karchmer-Widgerson Game

An important part of our protocol depends on the communication complexity of (monotone) weighted threshold functions. We start with the following result from [7, 8] which gives low-depth circuits for such functions. Another construction was given in [16]. The circuits as stated in [7, 8, 16] have logarithmic depth, polynomial size and unbounded fan-in, thus it is straightforward to convert them into circuits with fan-in 2 with a logarithmic increase in depth. Below we state the result for circuits of fan-in 2 as will be used later. Recall that a circuit is *monotone* if it uses only AND and OR gates, but no NOT gates.

Theorem 2 ([7, 8])**.** *Let* $w_1, \ldots, w_n \in \mathbb{Z}_{>0}$ *be positive weights, and* $T \in \mathbb{Z}_{\geqslant 0}$ *be a threshold. Let* $f : \{0,1\}^n \to \{0,1\}$ *be the monotone function such that* $f(x_1, \ldots, x_n) = 1$ *if and only if* $\sum_{i=1}^n w_i x_i \geqslant T$*. Then there is a depth-*$O(\log^2 n)$ *monotone circuit of fan-in* 2 *that computes the function* f*.*

The well-known *Karchmer-Wigderson game* [25] connects the depth of monotone circuits and communication complexity. Given a monotone function $f : \{0,1\}^n \to \{0,1\}$, the *monotone Karchmer-Wigderson game* is the following: Alice receives $a \in f^{-1}(0)$, Bob receives $b \in f^{-1}(1)$, they communicate bits to each other, and the goal is to agree on a position $i \in \{1, \ldots, n\}$ such that $a_i = 0$ and $b_i = 1$. Let $D^{\mathrm{cc}}_{\mathrm{mon-KW}}(f)$ be the deterministic communication complexity of this game.

Theorem 3 ([25])**.** *Let* $f : \{0,1\}^n \to \{0,1\}$ *be a monotone function,* $D^{\mathrm{cc}}_{\mathrm{mon-KW}}(f)$ *be the deterministic communication complexity of the Karchmer-Wigerson game, and* $\mathrm{depth}(f)$ *be the minimum depth of a fan-in* 2 *monotone circuit that computes* f*. Then* $\mathrm{depth}(f) = D^{\mathrm{cc}}_{\mathrm{mon-KW}}(f)$*.*

Combining Theorems 2 and 3, we immediately get that $D^{\mathrm{cc}}_{\mathrm{mon-KW}}(f) = O(\log^2 n)$ for every weighted threshold function f on n inputs.

3 Small LP relaxation for Min-Knapsack

In this section, we show the existence of a $(1/\varepsilon)^{O(1)} n^{O(\log n)}$-size LP relaxation of min-knapsack with integrality gap $2 + \varepsilon$, proving Theorem 1. First, we give a high-level overview of the construction in Section 3.1. The actual protocol is described and analyzed in Section 3.2.

3.1 Overview

Consider the slack matrix S that has one row for each knapsack cover inequality and one column for each feasible solution of min-knapsack. More precisely, let $f : \{0,1\}^n \to \{0,1\}$ denote the weighted threshold function defined by the item sizes s_i ($i \in [n]$) and demand D as in (2). The rows and columns of S are indexed by $a \in f^{-1}(0)$ and $b \in f^{-1}(1)$ respectively. The entries of S are given by

$$S_{a,b} := \sum_{i : a_i = 0} s'_i b_i - U \,,$$

where as precedingly $U = U(a) := D - \sum_{i:a_i=1} s_i$, and $s'_i = s'_i(a) = \min\{s_i, U\}$. Geometrically, S is the slack matrix of the polyhedral pair (P, Q) in which P is the min-knapsack polytope and Q is the (unbounded) polyhedron defined by the knapsack cover inequalities.

Ideally, we would like to design a communication protocol for S, as those discussed in Section 2.2, with low communication complexity. This would imply a low-rank non-negative factorization of S. From the factorization theorem of Section 2.1, it would follow that there exists a small-size extended formulation yielding a polyhedron R containing the min-knapsack polytope P and contained in the knapsack-cover relaxation Q. Hence, we would get a small-size LP relaxation for min-knapsack that implies the exponentially many knapsack cover inequalities, and thus have integrality gap at most 2.

However, due to the fact that the quantities involved can be exponential in n, making them too expensive to communicate directly, we have to settle for showing the existence of small-size extended formulation that *approximately* implies the knapsack cover inequalities. Before discussing further these complications, we give an idealized version of the protocol to help with the intuition. Assume for now that all item sizes and the demand are polynomial in n. Thus Alice and Bob can communicate them with $O(\log n)$ bits.

The goal of the two players is to compute the slack $S_{a,b} = \sum_{i:a_i=0} s'_i b_i - U$, when Alice is given an infeasible $a \in \{0,1\}^n$ and Bob is given a feasible $b \in \{0,1\}^n$. That is, after several rounds of communication, either one of them outputs some non-negative value ξ, such that the expectation of ξ equals $S_{a,b}$.

We define for a set of items $J \subseteq [n]$ the quantity $s(J) := \sum_{j\in J} s_j$, and $s'(J) := \sum_{j\in J} s'_j$. Let A and B be the subsets of $[n]$ corresponding to Alice's input a and Bob's input b, respectively. The slack we want to compute thus becomes $s'(B \setminus A) - U$.

At the beginning, Alice computes the residual demand U and sends it to Bob. Now observe that if there is some $i^* \in B \setminus A$, such that $s_{i^*} \geqslant U$, then we have $s'_{i^*} = U$, and we can easily write the slack as $s'(B \setminus A \setminus \{i^*\}) + (s'_{i^*} - U) = s'(B \setminus A \setminus \{i^*\})$ (similarly to the uncapacitated case discussed in the introduction). Recall that we call an item i large if $s_i \geqslant U$ and small otherwise. Let I_{large} be the set of large items and I_{small} be the set of small items.

The rest of the protocol is divided into two cases as follows, depending on whether Alice and Bob can easily find a large item $i^* \in B \setminus A$. To this end, Alice sends $s(I_{\text{large}} \cap A)$ to Bob. Note that now Bob can compute $s(I_{\text{small}} \cap A) = D - U - s(I_{\text{large}} \cap A)$. Bob computes the contribution of large items in B, that is, $s(I_{\text{large}} \cap B)$.

If $s(I_{\text{large}} \cap B) > s(I_{\text{large}} \cap A)$, then we are guaranteed that there is some $i^* \in I_{\text{large}} \cap (B \setminus A)$. Moreover, defining the threshold function

$$g(x) := \begin{cases} 1 & \text{if } \sum_{i\in I_{\text{large}}} s_i x_i \geqslant s(I_{\text{large}} \cap B), \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

then $g(a) = 0$ and $g(b) = 1$. Hence, Alice and Bob can find such an item with $O(\log^2 n)$ bits of communication, see Section 2.3. With that, it is not hard to compute $s'(B \setminus A \setminus \{i^*\})$ with $O(\log n)$ bits of communication: Alice samples a uniformly random item i and sends the index to Bob, Bob replies with b_i, Alice outputs $s'_i \cdot n$ if $b_i = 1$, $i \neq i^*$ and $i \notin A$, and outputs 0 otherwise. All her outputs are non-negative and their expectation is exactly the slack.

In the other case, $s(I_{\text{large}} \cap B) \leqslant s(I_{\text{large}} \cap A)$. Note that $s(B) = s(I_{\text{large}} \cap B) + s(I_{\text{small}} \cap B) \geqslant D = s(I_{\text{large}} \cap A) + s(I_{\text{small}} \cap A) + U$, thus $s(I_{\text{small}} \cap B) - s(I_{\text{small}} \cap A) - U \geqslant s(I_{\text{large}} \cap A) - s(I_{\text{large}} \cap B) \geqslant 0$. We now write the slack as

$$\begin{aligned} s'(B \setminus A) - U &= s'(I_{\text{large}} \cap (B \setminus A)) + s(I_{\text{small}} \cap (B \setminus A)) - U \\ &= s'(I_{\text{large}} \cap (B \setminus A)) + s(I_{\text{small}} \cap B) - s(I_{\text{small}} \cap (A \cap B)) - U \\ &= s'(I_{\text{large}} \cap (B \setminus A)) + s(I_{\text{small}} \cap B) - s(I_{\text{small}} \cap A) + s(I_{\text{small}} \cap (A \setminus B)) - U \\ &= s'(I_{\text{large}} \cap (B \setminus A)) + (s(I_{\text{small}} \cap B) - s(I_{\text{small}} \cap A) - U) + s(I_{\text{small}} \cap (A \setminus B)) \,. \end{aligned}$$

Alice and Bob can compute the first and the last term in expectation using a protocol similar to that in the previous case. The term in the middle can be computed by Bob with all the information he has at this stage. To conclude, in both cases, Alice and Bob can compute the exact slack $S_{a,b}$ with $O(\log^2 n)$ bits of communication.

3.2 The Protocol

The actual slack matrix S^ε we work with is defined as

$$S^\varepsilon_{a,b} := \sum_{i:a_i=0} s'_i b_i - \frac{2}{2+\varepsilon} U \,, \tag{5}$$

where $\varepsilon > 0$ is any small constant, $a \in f^{-1}(0)$ and $b \in f^{-1}(1)$. S^ε is the slack matrix of the polyhedral pair (P, Q^ε) where P is the min-knapsack polytope and Q^ε is the polyhedron defined by a slight weakening of the knapsack cover inequalities obtained by replacing the right hand side of (1) by $\frac{2}{2+\varepsilon}U < U$. For every $x \in \mathbb{R}^n_{\geqslant 0}$ that satisfies all weakened knapsack cover inequalities, we have that $\frac{2+\varepsilon}{2}x$ satisfies all original knapsack cover inequalities, and thus $(2+\varepsilon)x$ dominates a convex combination of feasible solutions. Therefore the integrality gap of the resulting LP relaxation (obtained from a non-negative factorization of S^ε) is at most $2+\varepsilon$.

In order to refer to the "derived" weighted threshold functions g as in (4), we need a last bit of terminology. We say that $g : \{0,1\}^n \to \{0,1\}$ is a *truncation* of f if there exists $U, T \in \mathbb{Z}_{>0}$ with $T \leqslant D$ such that $g(x) = 1$ iff $\sum_{i=1}^n w_i x_i \geqslant T$, where $w_i = s_i$ if $s_i \geqslant U$ and $w_i = 0$ otherwise. We are now ready to state our main technical lemma.

Lemma 4. *For all constants $\varepsilon \in (0,1)$, item sizes $s_i \in \mathbb{Z}_{>0}$ ($i \in [n]$), all smaller than $2^{\lceil n \log n \rceil}$ and demand $D \in \mathbb{Z}_{>0}$ with $\max\{s_i \mid i \in [n]\} \leqslant D \leqslant \sum_{i=1}^n s_i$, such that all truncations of the corresponding weighted threshold function admit depth-t monotone circuits of fan-in 2, there is a $O(\log(1/\varepsilon) + \log n + t)$-complexity randomized communication protocol with non-negative outputs that computes the slack matrix S^ε in expectation. Since we may always take $t = O(\log^2 n)$, this gives a $O(\log(1/\varepsilon) + \log^2 n)$-complexity protocol, unconditionally.*

Before giving the proof, let us remark that Theorem 1 follows directly from this lemma. Indeed, the extra assumptions in the lemma are without loss of generality: the fact that we may assume without loss of generality that the item sizes s_i are positive integers that can be written down with at most $\lceil n \log n \rceil$ bits, is due to a classic result from [31]; and the fact that we may also assume that the demand D is a positive integer with $\max\{s_i \mid i \in [n]\} \leqslant D \leqslant \sum_{i=1}^n s_i$ should be clear.

Moreover, Lemma 4 implies that we can obtain a relaxation of polynomial size if all truncations of the weighted threshold function have monotone circuits of logarithmic depth. In particular, this is the case if all item sizes are polynomial in n. In that case the threshold function (and its truncations) can simply be written as the majority function on $O(\sum_i s_i)$ input bits and, as such functions have monotone circuits of fan-in 2 of logarithmic depth, i.e., depth $O(\log(\sum_i s_i))$, we obtain the following corollary.

Corollary 5. *For all $\varepsilon \in (0,1)$, $c > 0$, item sizes $s_1, \ldots, s_n \in \{0, 1, \ldots, n^c\}$ and demand $D \in \mathbb{N}$, there exists a size-$(1/\varepsilon)^{O(1)} n^{O(c)}$ extended formulation defining an LP relaxation of min-knapsack with integrality gap at most $2+\varepsilon$.*

We now proceed by proving our main technical result, i.e., Lemma 4.

Proof of Lemma 4. Let $\alpha = \alpha(\varepsilon) := 2/(2+\varepsilon)$ and $\delta > 0$ be such $(1-2\delta)/(1+\delta) = \alpha$. Thus $\delta = \varepsilon/(6+2\varepsilon) = \Theta(\varepsilon)$. As above, we denote by $a \in f^{-1}(0)$ the input of Alice and $b \in f^{-1}(1)$ that of Bob, and let A and B denote the corresponding subsets of $[n]$.

First, Alice tells Bob the identity of the set of large items $I_{\text{large}} = \{i \in [n] \mid s_i \geqslant U\}$ and its complement, the set of small items I_{small}. This costs $O(\log n)$ bits of communication. For instance, Alice can simply send the index of a smallest large item to Bob, or inform Bob that I_{large} is empty. Recall that

$$U = D - s(A) = D - s(I_{\text{large}} \cap A) - s(I_{\text{small}} \cap A)\,.$$

Then, Alice sends Bob the unique nonnegative integer k such that $(1+\delta)^k \leqslant U < (1+\delta)^{k+1}$. This sets the scale at which the protocol is operating. Since $U \leqslant n \cdot 2^{\lceil n \log n \rceil} \leqslant 2^{n^2}$, we have $(1+\delta)^k \leqslant 2^{n^2}$. This implies that $k = O((1/\varepsilon)n^2)$, thus k can be sent to Bob with $\log(1/\varepsilon) + 2\log n + O(1) = O(\log(1/\varepsilon) + \log n)$ bits. Let $\widetilde{U} := (1+\delta)^k$.

To efficiently communicate an approximate value of $s(I_{\text{large}} \cap A)$, Alice sends the unique nonnegative integer ℓ such that

$$(1+\ell\delta)\widetilde{U} < D - s(I_{\text{large}} \cap A) \leqslant (1+\ell\delta)\widetilde{U} + \delta\widetilde{U}.$$

Since small items have size at most U and we have at most n of them, we have $s(I_{\text{small}} \cap A) \leqslant Un$. Hence, $D - s(I_{\text{large}} \cap A) = U + s(I_{\text{small}} \cap A) \leqslant (n+1)U \leqslant (n+1)(1+\delta)\widetilde{U}$. Since $(1+\ell\delta)\widetilde{U} < (n+1)(1+\delta)\widetilde{U}$, we have $\ell = O((1/\varepsilon)n)$. This means that Alice can communicate ℓ to Bob with only $O(\log(1/\varepsilon) + \log n)$ bits. Let $\tilde{\Delta} = \tilde{\Delta}(\delta) := (1+\ell\delta)\widetilde{U}$. This is Bob's strict under-approximation of $D - s(I_{\text{large}} \cap A)$, so that $D - \tilde{\Delta}$ is a strict over-approximation of $s(I_{\text{large}} \cap A)$.

Bob checks if $s(I_{\text{large}} \cap B) \geqslant D - \tilde{\Delta}$. If this is the case, then the weighted threshold function g such that $g(x) = 1$ iff $\sum_{i \in I_{\text{large}}} s_i x_i \geqslant D - \tilde{\Delta}$ separates a from b in the sense that $g(a) = 0$ and $g(b) = 1$. Since g is a truncation of f, Alice and Bob can exchange t bits to find an index $i^* \in I_{\text{large}}$ such that $a_{i^*} = 0$ and $b_{i^*} = 1$.

We can rewrite the slack $S^{\varepsilon}_{a,b} = s'(B \smallsetminus A) - \alpha U$ as

$$s'(B \smallsetminus A \smallsetminus \{i^*\}) + s'_{i^*} - \alpha U = s'(B \smallsetminus A \smallsetminus \{i^*\}) + (U - \alpha U) = \sum_{i : a_i = 0,\ i \neq i^*} s'_i b_i + (U - \alpha U)\,. \quad (6)$$

With the knowledge of i^*, Alice and Bob can compute the slack as follows:

1. Alice samples a uniformly random number $i \in [n]$. If $i \notin A$, continue to the next step, otherwise Alice outputs 0 and terminates the communication.
2. If $i = i^*$, Alice outputs $n \cdot (U - \alpha U)$ and terminates the communication, otherwise continue.
3. Alice sends i to Bob using $\lceil \log n \rceil$ bits of communication, and Bob sends b_i back to Alice.
4. Alice outputs $n \cdot s'_i b_i$.

The above communication costs $O(\log n)$ bits, all outputs are non-negative and can be computed with the information available to each player, and by linearity of expectation, the expected output is exactly the slack (6). Together with the $O(\log(1/\varepsilon) + \log n + t)$ bits communicated previously, we conclude that in this case there is a protocol that computes the slack in expectation with $O(\log(1/\varepsilon) + \log n + t)$ bits of communication.

In the other case, we have $s(I_{\text{large}} \cap B) < D - \tilde{\Delta}$. Because $b \in \{0,1\}^n$ is feasible, we get

$$s(B) \geqslant D \iff \underbrace{s(I_{\text{large}} \cap B)}_{< D - \tilde{\Delta}} + s(I_{\text{small}} \cap B) \geqslant D\,,$$

therefore we can bound $s(I_{\text{small}} \cap B)$ as

$$s(I_{\text{small}} \cap B) > \tilde{\Delta} \geqslant D - s(I_{\text{large}} \cap A) - \delta\widetilde{U} = s(I_{\text{small}} \cap A) + U - \delta\widetilde{U} \geqslant \widetilde{\sigma} + (1-\delta)\widetilde{U}\,, \quad (7)$$

where $\widetilde{\sigma}$ is the unique integer multiple of $\delta\widetilde{U}$ such that

$$\widetilde{\sigma} \leqslant s(I_{\text{small}} \cap A) < \widetilde{\sigma} + \delta\widetilde{U}\,. \tag{8}$$

Since $\widetilde{\sigma} \leqslant s(I_{\text{small}} \cap A) \leqslant Un \leqslant (1+\delta)\widetilde{U}n$, Alice can communicate $\widetilde{\sigma}$ to Bob with $O(\log(1/\varepsilon)+\log n)$ bits.

This implies

$$s(I_{\text{small}} \cap (B \smallsetminus A)) = s(I_{\text{small}} \cap B) - s(I_{\text{small}} \cap (A \cap B)) > \widetilde{\sigma} + (1-\delta)\widetilde{U} - s(I_{\text{small}} \cap (A \cap B))\,.$$

Recall that by definition of $\widetilde{U}$, we have $(1+\delta)\widetilde{U} > U$, therefore

$$(1-2\delta)\widetilde{U} - \alpha U > (1-2\delta)\widetilde{U} - \alpha(1+\delta)\widetilde{U} = 0\,. \tag{9}$$

We now rewrite the slack as

$$s'(B \smallsetminus A) - \alpha U = \underbrace{s'(I_{\text{large}} \cap (B \smallsetminus A))}_{=\sum_{i \in I_{\text{large}} \smallsetminus A} s'_i b_i} + \underbrace{s(I_{\text{small}} \cap B) - \widetilde{\sigma} - (1-\delta)\widetilde{U}}_{\text{non-negative by (7)}} + \underbrace{s(I_{\text{small}} \cap (A \smallsetminus B))}_{\sum_{i \in I_{\text{small}} \cap A} s_i(1-b_i)}$$
$$+ \underbrace{\widetilde{\sigma} - s(I_{\text{small}} \cap A) + (1-\delta)\widetilde{U} - \alpha U}_{\text{non-negative by (8) and (9)}}\,.$$

Similar to the previous case, we design a protocol to compute the slack as follows:

1. Alice samples a uniformly random number $i \in [n+2]$. If $i = n+2$, Alice outputs the normalized value of the last term, i.e., $(n+2) \cdot (\widetilde{\sigma} - s(I_{\text{small}} \cap A) + (1-\delta)\widetilde{U} - \alpha U)$, and terminates the communication. Otherwise, she sends i to Bob using $O(\log n)$ bits.

2. If $i = n+1$, Bob outputs $(n+2) \cdot (s(I_{\text{small}} \cap B) - \widetilde{\sigma} - (1-\delta)\widetilde{U})$, and ends the communication. Otherwise, he replies to Alice with b_i.

3. If $i \in I_{\text{large}} \smallsetminus A$, Alice outputs $(n+2) \cdot s'_i b_i$; if $i \in I_{\text{small}} \cap A$, she outputs $(n+2) \cdot s_i(1-b_i)$; otherwise she outputs 0.

We can verify that the outputs of both players can be computed with information available to them, and that the outputs are non-negative due to Equation (7), (8) and (9), and the definition of the variables. □

4 Conclusion

After the recent series of strong negative results on extended formulations, we have presented a positive result inspired by a connection to monotone circuits. Namely, we obtain the first quasi-polynomial-size LP relaxation of min-knapsack with constant integrality gap from polylog-depth circuits for weighted threshold functions.

This result sheds new light on the approximability of min-knapsack via small LPs by connecting it to the complexity of monotone circuits. For instance, it follows from our results that proving that no $n^{O(1)}$-size LP relaxation for min-knapsack can have integrality gap at most α for some $\alpha > 2$ would rule out the existence of $O(\log n)$-depth monotone circuits with bounded fan-in for weighted threshold functions on n inputs, which is an open problem.

References

[1] Hyung-Chan An, Mohit Singh, and Ola Svensson. LP-based algorithms for capacitated facility location. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 256–265, 2014.

[2] E. Balas. Facets of the knapsack polytope. *Math. Program.*, 8:146–164, 1975.

[3] Nikhil Bansal, Niv Buchbinder, and Joseph Seffi Naor. Randomized competitive algorithms for generalized caching. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 235–244. ACM, 2008.

[4] Nikhil Bansal, Anupam Gupta, and Ravishankar Krishnaswamy. A constant factor approximation algorithm for generalized min-sum set cover. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 1539–1545. Society for Industrial and Applied Mathematics, 2010.

[5] Nikhil Bansal and Kirk Pruhs. The geometry of scheduling. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 407–414. IEEE, 2010.

[6] A. Bazzi, S. Fiorini, S. Pokutta, and O. Svensson. Small linear programs cannot approximate Vertex Cover within a factor of $2 - \varepsilon$. In *IEEE 56th Annual Symp. on Foundations of Computer Science (FOCS)*, pages 1123–1142, 2015.

[7] A. Beimel and E. Weinreb. Monotone circuits for weighted threshold functions. In *Proc. of the 20th Annual IEEE Conference on Computational Complexity*, pages 67–75, 2005.

[8] A. Beimel and E. Weinreb. Monotone circuits for monotone weighted threshold functions. *Information Processing Letters*, 97:12–18, 2006.

[9] D. Bienstock. Approximate formulations for 0-1 knapsack sets. *Operations Research Letters*, pages 317–320, 2008.

[10] Daniel Bienstock and Benjamin McClosky. Tightening simple mixed-integer sets with guaranteed bounds. *Mathematical Programming*, 133(1):337–363, 2012.

[11] G. Braun, S. Fiorini, S. Pokutta, and D. Steurer. Approximation limits of linear programs (beyond hierarchies). In *53rd IEEE Symp. on Foundations of Computer Science (FOCS 2012)*, pages 480–489, 2012.

[12] Tim Carnes and David Shmoys. Primal-dual schema for capacitated covering problems. In *Proceedings of the 13th International Conference on Integer Programming and Combinatorial Optimization*, IPCO'08, pages 288–302, Berlin, Heidelberg, 2008. Springer-Verlag.

[13] R.D. Carr, L.K. Fleischer, V.J. Leung, and C.A. Philipps. Strengthening integrality gaps for capacitated network design and covering problems. In *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms (SODA '00)*, pages 106–115, 2000.

[14] Deeparnab Chakrabarty, Elyot Grant, and Jochen Konemann. On column restricted and priority integer covering programs. In *Conference on Integer Programming and Combinatorial Optimization*, 2010.

[15] S.O. Chan, J.R. Lee, P. Raghavendra, and D. Steurer. Approximate constraint satisfaction requires large LP relaxations. In *IEEE 54th Annual Symp. on Foundations of Computer Science (FOCS 2013)*, pages 350–359, 2013.

[16] Xi Chen, Igor Carboni Oliveira, and Rocco A. Servedio. Addition is exponentially harder than counting for shallow monotone circuits. arXiv:1508.03061, 2015.

[17] Maurice Cheung and David B Shmoys. A primal-dual approximation algorithm for min-sum single-machine scheduling problems. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 135–146. Springer, 2011.

[18] M. Conforti, G. Cornuéjols, and G. Zambelli. Extended formulations in combinatorial optimization. *4OR*, 8:1–48, 2010.

[19] Hossein Efsandiari, MohammadTaghi Hajiaghyi, Jochen Könemann, Hamid Mahini, David Malec, and Laura Sanita. Approximate deadline-scheduling with precedence constraints. In *Algorithms-ESA 2015*, pages 483–495. Springer, 2015.

[20] Y. Faenza, S. Fiorini, R. Grappe, and H. R. Tiwary. Extended formulations, non-negative factorizations and randomized communication protocols. *Math. Programming*, 153:75–94, 2015.

[21] S. Fiorini, S. Massar, S. Pokutta, H. R. Tiwary, and R. de Wolf. Linear vs. semidefinite extended formulations: Exponential separation and strong lower bounds. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, 2012.

[22] M. Göös, R. Jain, and T. Watson. Extension complexity of independent set polytopes. arXiv:1604.07062, April 2016.

[23] P.L. Hammer, E.L. Johnson, and U.N. Peled. Facets of regular 0-1 polytopes. *Math. Program.*, 8:179–206, 1975.

[24] P. Hrubeš. On the nonnegative rank of distance matrices. *Information Processing Letters*, 112:457–461, 2012.

[25] M. Karchmer and A. Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM J. Discrete Math.*, 3:255–265, 1990.

[26] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[27] E. L. Lawler. Fast approximation algorithms for knapsack problems. *Mathematics of Operations Research*, 4:339–356, 1979.

[28] J.R. Lee, P. Raghavendra, and D. Steurer. Lower bounds on the size of semidefinite programming relaxations. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC)*, pages 567–576, 2015.

[29] Retsef Levi, Andrea Lodi, and Maxim Sviridenko. Approximation algorithms for the capacitated multi-item lot-sizing problem via flow-cover inequalities. *Mathematics of Operations Research*, 33(2):461–474, 2008.

[30] Julián Mestre and José Verschae. A 4-approximation for scheduling on a single machine with general cost function. *arXiv preprint arXiv:1403.0298*, 2014.

[31] S. Muroga. *Threshold Logic and Its Applications*. Wiley-Interscience, 1971.

[32] M. W. Padberg, T. J. Van Roy, and L. A. Wolsey. Valid inequalities for fixed charge problems. *Oper. Res.*, 33:842–861, 1985.

[33] S. Pokutta and M. Van Vyve. A note on the extension complexity of the knapsack polytope. *Operations Research Letters*, 41:347–350, 2013.

[34] T. Rothvoß. The matching polytope has exponential extension complexity. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 263–272, 2014.

[35] L.A. Wolsey. Faces for linear inequalities in 0-1 variables. *Math. Program.*, 8:165–178, 1975.

[36] M. Yannakakis. Expressing combinatorial optimization problems by linear programs. *J. Comput. System Sci.*, 43:441–466, 1991.

A Flow-cover inequalities

A variant of the knapsack cover inequalities, known as the *flow cover inequalities*, was also used to strengthen LPs for many problems such as the Fixed Charge Network Flow problem [13] and the Single-Demand Facility Location problem [12]. In this section, we describe the application of flow cover inequalities to the Single-Demand Facility Location problem as used in [12], and then give an $O(\log^2 n)$-bit two-party communication protocol that computes a weakened version of these inequalities.

In the Single-Demand Facility Location problem, we are given a set F of n facilities, such that each facility $i \in F$ has a capacity s_i, an opening cost f_i, and a per-unit cost c_i to serve the demand. The goal is to serve the demand D by opening a subset $S \subseteq F$ of facilities such that the combined cost of opening these facilities and serving the demand is minimized. The authors of [12] cast this problem as an Integer Program, and showed that its natural LP relaxation has an unbounded integrality gap. To reduce the integrality gap, they strengthened the relaxation by adding the so-called flow cover inequalities that we define shortly (See Section 3 in [12] for a more elaborate discussion).

A feasible solution (x, y) with $y \in \{0,1\}^n$ and $x \in [0,1]^n$ for the Single-Demand Facility Location LP can be thought of as follows: for each $i \in F$, $y_i \in \{0,1\}$ indicates if the i-th facility is open, and $x_i \in [0,1]$ indicates the fraction of the demand D being served by the i-th facility. A feasible solution (x, y) must then satisfy that

1. The demand is *met*, i.e., $\sum_i x_i = 1$.
2. No facility is supplying more than its capacity, i.e., $0 \le x_i D \le y_i s_i$ for all $i \in F$.

For a subset $J \subseteq F$ of facilities and a feasible solution (x, y), we denote by $B = \{i \in F : y_i = 1\} \subseteq [F]$ the set of *open* facilities according to y, and we define the quantity $x(J)$ to be the overall demand served by the facilities in J, i.e., $x(J) = \sum_{i \in J} x_i D$.[6] We also define the quantities $s(\cdot)$ and $s'(\cdot)$ as in Section 3.1.

Carnes and Shmoys [12] showed that the natural LP relaxation for the Single-Demand Facility Location has an unbounded integrality gap, and that adding the following flow cover inequalities (FCI) reduces the integrality gap down to 2: for any *infeasible* set $A \subseteq F$ (i.e., $A \subseteq F$ such that $s(A) < D$), and for all partitions of $F \setminus A = F_1 \sqcup F_2$, the following inequality holds for all feasible solutions (x, y):

$$s'(F_1 \cap B) + x(F_2 \cap B) \ge U\,, \tag{FCI}$$

where $U = D - s(A)$ is the residual demand and $s'_i = \min\{s_i, U\}$. For brevity, we refer to an infeasible set A along with some partition $F_1 \sqcup F_2 = F \setminus A$ as an *infeasible tuple* (A, F_1, F_2). Note that for $F_2 = \emptyset$, the flow-cover inequalities are the same as the knapsack cover inequalities.

Similar to the knapsack cover inequalities, the goal is to compute the slack of a *relaxed* version of (FCI) in expectation for any feasible solution (x, y) and any infeasible tuple (A, F_1, F_2). Namely, for any

[6] Note that since we are assuming that (x, y) is feasible, we get that $x(J) = x(J \cap B)$.

$\varepsilon \in (0,1)$, let $\alpha = 2/(2+\varepsilon)$, then our goal is to design an $O(\log^2 n + \log(1/\varepsilon))$-complexity two-party communication protocol with private randomness and nonnegative outputs whose expected output equals $s'(F_1 \cap B) + x(F_2 \cap B) - \alpha U$. That is, we want to compute the slack with respect to a given (weakened) flow-cover inequality $s'(F_1 \cap B) + x(F_2 \cap B) \geq \alpha U$, where the RHS of (FCI) is replaced by αU. This implies the existence of an LP of size $(1/\varepsilon)^{O(1)} n^{O(\log n)}$ with an integrality at most $2+\varepsilon$ for the Single-Demand Facility Location problem.

In Section A.1, we set up the notation and define a class of feasible solutions with a certain special structure which we refer to as *canonical* feasible solutions. We design the promised communication protocol restricted to canonical solutions in Section A.2, and extend it to arbitrary feasible solutions in Section A.3.

A.1 Preliminaries

Let (x, y) be a feasible solution for the flow-cover problem with demand D, and let $B = \{i \in F : y_i = 1\}$ denote the support of y. In this terminology, B only indicates which facilities are open, but it does not capture the *relative* demand being served through each of them. However this distinction will be essential for designing the protocol, hence we partition B into three disjoint sets $B = \widetilde{F}_1 \sqcup \widetilde{F}_2 \sqcup \widetilde{F}_3$ as follows:

$$\widetilde{F}_1 = \{i \in B : x_i D = s_i y_i\}, \qquad \widetilde{F}_2 = \{i \in B : 0 < x_i D < s_i y_i\}, \qquad \widetilde{F}_3 = \{i \in B : x_i D = 0\}.$$

As mentioned earlier, we will first focus on feasible solutions (x, y) that exhibit a certain structure, and then generalize to arbitrary solutions. Namely, we restrict our attention here and in Section A.2 to *canonical* feasible solutions defined as follows:

Definition 5. *A feasible solution (x, y) with associated sets $\widetilde{F}_1, \widetilde{F}_2, \widetilde{F}_3$ is* canonical *if $\widetilde{F}_2$ contains at most one facility, i.e., $|\widetilde{F}_2| \leq 1$. In other words, in a canonical feasible solution, there is at most one facility j that supplies a non-zero demand $x_j D > 0$ which is* not *equal to its full capacity s_j.*

Recall that we are interested in computing

$$s'(F_1 \cap B) + x(F_2 \cap B) - \alpha U \tag{10}$$

in expectation, which can be expanded as follows:

$$s'(F_1 \cap \widetilde{F}_1) + s'(F_1 \cap \widetilde{F}_2) + s'(F_1 \cap \widetilde{F}_3) + x(F_2 \cap \widetilde{F}_1) + x(F_2 \cap \widetilde{F}_2) + x(F_2 \cap \widetilde{F}_3) - \alpha U. \tag{11}$$

We get from the definition of the set $\widetilde{F}_3$ that the second to last term in the above equation is 0 when restricted to canonical feasible solutions.

Before we proceed, consider a canonical feasible solution (x, y) such that $\widetilde{F}_3 \neq \emptyset$, and let $(x, \bar{y})$ be the projection of (x, y) on $\widetilde{F}_1 \cup \widetilde{F}_2$ — that is, for all $i \in B \setminus \widetilde{F}_3$, set $\bar{y}_i = y_i$, and for all $i \in \widetilde{F}_3$, set $\bar{y}_i = 0$. It follows that $(x, \bar{y})$ is also a canonical feasible solution, as the items whose support is $\widetilde{F}_3$ do not contribute to the feasibility of the solution, and the cardinality of $\widetilde{F}_2$ does not change. Thus, for any infeasible tuple (A, F_1, F_2), Equation (11) applied to $(x, \bar{y})$ can be written as

$$s'(F_1 \cap \widetilde{F}_1) + s'(F_1 \cap \widetilde{F}_2) + x(F_2 \cap \widetilde{F}_1) + x(F_2 \cap \widetilde{F}_2) - \alpha U, \tag{12}$$

which is also non-negative, as it is the slack of $(x, \bar{y})$ and (A, F_1, F_2). Therefore, for any feasible solution (x, y), the slack as given by Equation (11) can be viewed as the summation of Equation (12) and $s'(F_1 \cap \widetilde{F}_3)$. The latter is easy to compute with a small communication protocol[7], thus if Alice and Bob can devise a communication protocol Π that computes (12) in expectation, they can then easily compute (11) in expectation. For example, Alice can generate a uniformly random bit $b \in \{0, 1\}$, and

[7] To compute $s'(F_1 \cap \widetilde{F}_3)$, Bob samples an index $i \in [n]$. If $i \notin \widetilde{F}_3$, he outputs 0 and terminates the protocol, otherwise he sends i to Alice. If $i \in F_1$, Alice outputs $n \cdot s'(i)$, otherwise, she outputs 0.

- if $b = 0$, then Alice and Bob run the protocol that computes $s'(F_1 \cap \widetilde{F}_3)$, and return *twice* its output.
- if $b = 1$, then Alice and Bob run the protocol Π that computes (12), and return *twice* its output.

We now focus on the case in which Bob's input is a canonical feasible solution (x, y) such that $\widetilde{F}_3 = \emptyset$. Moreover, since $|\widetilde{F}_2| \leq 1$, and using the fact that $x_i D = s_i y_i$ for $i \in \widetilde{F}_1$, we can further simplify Equation (11) as follows:

$$s'(F_1 \cap \widetilde{F}_1) + s(F_2 \cap \widetilde{F}_1) + \gamma(x, y, A, F_1, F_2) - \alpha U\,, \tag{13}$$

where the function $\gamma(x, y, A, F_1, F_2)$ is defined as

$$\gamma(x, y, A, F_1, F_2) = \begin{cases} s'_j y_j & \text{if } \widetilde{F}_2 = \{j\} \subseteq F_1 \\ x_j D & \text{if } \widetilde{F}_2 = \{j\} \subseteq F_2 \\ 0 & \text{if } \widetilde{F}_2 = \{j\} \subseteq A, \text{ or } \widetilde{F}_2 = \emptyset\,. \end{cases} \tag{14}$$

For simplicity of notation, we drop the parameters from $\gamma(x, y, A, F_1, F_2)$ when its is clear from the context and denote it by γ, i.e., $\gamma \coloneqq \gamma(x, y, A, F_1, F_2)$.

A.2 Randomized Protocol for Canonical Feasible Solutions

In what follows, we define a randomized communication protocol where Alice gets an infeasible tuple (A, F_1, F_2), and Bob gets a *canonical* feasible solution (x, y) with $\widetilde{F}_3 = \emptyset$, and the goal is to compute the value of (13) in expectation.

For a fixed $\varepsilon > 0$, we define $\alpha := \alpha(\varepsilon) = 2/(2+\varepsilon)$, $\delta := \delta(\varepsilon) = \varepsilon/(6+2\varepsilon)$ as in the min-knapsack case. Similar to the protocol for the knapsack cover inequalities, Alice sends Bob $O(\log n)$ bits at the beginning so that Bob knows $I_{\text{large}}, I_{\text{small}}, \widetilde{U}, \widetilde{\sigma}$ and $\widetilde{\Delta}$. Recall that I_{large} is the set of large items (i.e., $i \in F$ such that $s(i) \geq U$), I_{small} is the set of small items, $\widetilde{U}$ is an under-approximation of the residual demand U, $D - \widetilde{\Delta}$ is an over-approximation of $s(I_{\text{large}} \cap A)$, $\widetilde{\sigma}$ is an under-approximation of $s(I_{\text{small}} \cap A)$. Moreover, knowing his input (x, y), Bob can construct the sets $\widetilde{F}_1$ and $\widetilde{F}_2$. Thus, by exchanging an additional $O(\log n)$ bits, Alice and Bob can both figure out which condition is satisfied for Equation (14).

To compute the value of (13) in expectation, we distinguish between the following cases:

Case 1: Either $\widetilde{F}_2 = \emptyset$, or $\widetilde{F}_2 = \{j\}$ and $j \in A \cup F_1$. In this case, we have that the value γ is either 0 or $s'_j y_j$. Bob now checks if

$$s(I_{\text{large}} \cap (\widetilde{F}_1 \cup \widetilde{F}_2)) \geq D - \widetilde{\Delta}\,. \tag{15}$$

Equation (15) holds: In the same way as in the min-knapsack protocol, Alice and Bob exchange $O(\log^2 n)$ bits to identify an index $i^* \in I_{\text{large}}$ such that $i^* \in ((\widetilde{F}_1 \cup \widetilde{F}_2) \setminus A)$. Note that it may happen that $i^* = j$.

If $i^* \in F_2 \cap \widetilde{F}_1$, then Equation (13) can be rewritten as

$$s'(F_1 \cap \widetilde{F}_1) + s((F_2 \cap \widetilde{F}_1) \setminus \{i^*\}) + \gamma + (s_{i^*} - \alpha U)\,. \tag{16}$$

One can see that each of the above four terms is non-negative, and similar to the min-knapsack protocol, Alice and Bob can exchange $O(\log n)$ bits and compute the value of (16) as follows:

1. Bob sends Alice the bit y_j and the index j using $\lceil \log(n) \rceil + 1$ bits if and only if $\widetilde{F}_2 = \{j\}$, and he sends 0 if $\widetilde{F}_2 = \emptyset$.

2. Alice samples a uniformly random index $i \in [n+1]$. If $i = n+1$, Alice uses the knowledge of $\widetilde{F}_2$ (and thus γ) to compute the normalized value of the last terms, that is, she outputs $(n+1)\cdot(\gamma + s_{i^*} - \alpha U)$, and terminates the communication. Otherwise, she sends i to Bob using $\lceil \log(n) \rceil$ bits.
3. If $i \in \widetilde{F}_1$, Bob sends y_i to Alice; otherwise $i \notin \widetilde{F}_1$, Bob outputs 0 and terminates the communication.
4. If $i \in F_1$, Alice outputs $(n+1)\cdot s'_i y_i$; if $i \in F_2 \setminus \{i^*\}$, she outputs $(n+1)\cdot s_i y_i$; otherwise she outputs 0.

The above communication costs $O(\log n)$ bits, all outputs are non-negative and can be computed with the information available to each player, and by linearity of expectation, the expected output is exactly the slack (13) when $i^* \in F_2 \cap \widetilde{F}_1$.

The case where $i^* \in F_1 \cap \widetilde{F}_1$ is handled similarly.

In the remaining case, we have $\widetilde{F}_2 = \{j\}$ and $i^* = j \in F_1 \cap I_{\text{large}}$, and hence $\gamma = s'_j y_j > \alpha U$. The protocol described for the previous case can be changed to handle this case as well, where in the second step Alice outputs $(n+1)\cdot(s'_j - \alpha U)$ if $i = n+1$.

Equation (15) does not hold: Recall that since (x, y) is a feasible solution (and $\widetilde{F}_3 = \emptyset$), we have

$$\begin{aligned} D &\le x(\widetilde{F}_1) + x(\widetilde{F}_2) \\ &= x(I_{\text{small}} \cap \widetilde{F}_1) + x(I_{\text{small}} \cap \widetilde{F}_2) + x(I_{\text{large}} \cap \widetilde{F}_1) + x(I_{\text{large}} \cap \widetilde{F}_2) \\ &\le x(I_{\text{small}} \cap \widetilde{F}_1) + x(I_{\text{small}} \cap \widetilde{F}_2) + s(I_{\text{large}} \cap \widetilde{F}_1) + s(I_{\text{large}} \cap \widetilde{F}_2) \\ &= x(I_{\text{small}} \cap \widetilde{F}_1) + x(I_{\text{small}} \cap \widetilde{F}_2) + s(I_{\text{large}} \cap (\widetilde{F}_1 \cup \widetilde{F}_2)) \,. \end{aligned}$$

By the assumption that Equation (15) does not hold, together with the argument in Equation (7), we conclude that

$$x(I_{\text{small}} \cap \widetilde{F}_1) + x(I_{\text{small}} \cap \widetilde{F}_2) > \widetilde{\Delta} \ge \widetilde{\sigma} + (1-\delta)\widetilde{U} \,. \tag{17}$$

Since $|\widetilde{F}_2| \le 1$, the item $x(I_{\text{small}} \cap \widetilde{F}_2)$ must be in one of the following cases and we abbreviate it as $x_{\widetilde{F}_2}$:

$$x_{\widetilde{F}_2} = \begin{cases} 0 & \text{if } \widetilde{F}_2 = \emptyset \\ 0 & \text{if } \widetilde{F}_2 = \{j\} \text{ and } j \in I_{\text{large}} \\ x_j D & \text{if } \widetilde{F}_2 = \{j\} \text{ and } j \in I_{\text{small}} \,. \end{cases}$$

By the definition of $\widetilde{F}_1$, we also have $x(I_{\text{small}} \cap \widetilde{F}_1) = s(I_{\text{small}} \cap \widetilde{F}_1)$. Together this gives

$$s(I_{\text{small}} \cap \widetilde{F}_1) + x_{\widetilde{F}_2} > \widetilde{\sigma} + (1-\delta)\widetilde{U} \,. \tag{18}$$

We rewrite (13) as

$$\begin{aligned} & s'(F_1 \cap \widetilde{F}_1) + s(F_2 \cap \widetilde{F}_1) + \gamma - \alpha U \\ =& s'(I_{\text{large}} \cap F_1 \cap \widetilde{F}_1) + s(I_{\text{large}} \cap F_2 \cap \widetilde{F}_1) + s(I_{\text{small}} \cap (\widetilde{F}_1 \setminus A)) + \gamma - \alpha U \\ =& s'(I_{\text{large}} \cap F_1 \cap \widetilde{F}_1) + s(I_{\text{large}} \cap F_2 \cap \widetilde{F}_1) + s(I_{\text{small}} \cap (A \setminus B)) \\ & + s(I_{\text{small}} \cap A \cap \widetilde{F}_2) + s(I_{\text{small}} \cap \widetilde{F}_1) - s(I_{\text{small}} \cap A) + \gamma - \alpha U \,. \end{aligned} \tag{19}$$

The non-negativity of the first three terms is straightforward, and Alice and Bob can compute them easily by exchanging $O(\log n)$ bits [8]. By adding and subtracting $(\widetilde{\sigma} + (1-\delta)\widetilde{U} - x_{\widetilde{F}_2})$ to the remaining terms in (19), we can rearrange the terms and rewrite the rest as the sum of the following three non-negative terms that we can easily compute:

$$\left(s(I_{\text{small}} \cap \widetilde{F}_1) - \widetilde{\sigma} - (1-\delta)\widetilde{U} + x_{\widetilde{F}_2}\right) + \left(\widetilde{\sigma} + (1-\delta)\widetilde{U} - \alpha U - s(I_{\text{small}} \cap A)\right) + \left(s(I_{\text{small}} \cap A \cap \widetilde{F}_2) + \gamma - x_{\widetilde{F}_2}\right). \tag{20}$$

The non-negativity of the first part follows from (18), and Bob has all the information to compute it on his own. The non-negativity of the second part follows from our definition of $\widetilde{\sigma}$ and $\widetilde{U}$, and their relation to δ and α. Moreover, Alice has all the information to compute this part.

For the third part, if $x_{\widetilde{F}_2} = 0$, then clearly it is non-negative. In this case, Bob tells Alice $\widetilde{F}_2$ using $O(\log n)$ bits, and Alice can now compute the third part — by the assumption of this case, the value of γ is either 0 or $s'_j y_j$, depending on $\widetilde{F}_2$; the other term $s(I_{\text{small}} \cap A \cap \widetilde{F}_2)$ is also easy for her once $\widetilde{F}_2$ is known. If $x_{\widetilde{F}_2} \neq 0$, by definition this happens only if $\widetilde{F}_2 = \{j\}$ and $j \in I_{\text{small}}$. From our assumption of Case 1, we also have that $j \in A \cup F_1$. If $j \in A$, then $\gamma = 0$, else if $j \in F_1$, then $s(I_{\text{small}} \cap A \cap \widetilde{F}_2) = 0$. Thus the third part becomes

$$s(I_{\text{small}} \cap A \cap \widetilde{F}_2) + \gamma - x_{\widetilde{F}_2} = s_j y_j - x_j D \geq 0\,.$$

We conclude that the third part is non-negative in this case as well, and it is clear that Bob can compute $s_j y_j - x_j D$ on his own.

Case 2: $\widetilde{F}_2 = \{j\}$ and $j \in F_2$. In this case $\gamma = x_j D$. This case is quite similar to Case 1, with the difference being that Bob checks at the beginning if

$$s(I_{\text{large}} \cap \widetilde{F}_1) \geq D - \widetilde{\Delta}\,,$$

i.e., without including $\widetilde{F}_2$ compared to (15).

If the condition was indeed satisfied, then the same reasoning as the first part of Case 1 resolves this case. Otherwise, we get

$$s(I_{\text{small}} \cap \widetilde{F}_1) + x_j D > \widetilde{\sigma} + (1-\delta)\widetilde{U}\,, \tag{21}$$

and using Equation (19) from the second part of Case 1 yields that that *first four* terms in this case are non-negative and easy to compute. Similarly, adding and subtracting $(\widetilde{\sigma} + (1-\delta)\widetilde{U})$ to the *last four* terms of (19), and rearranging the terms we get

$$\left(s(I_{\text{small}} \cap \widetilde{F}_1) - \widetilde{\sigma} - (1-\delta)\widetilde{U} + x_j D\right) + \left(\widetilde{\sigma} + (1-\delta)\widetilde{U} - \alpha U - s(I_{\text{small}} \cap A)\right).$$

The first part of the summation is non-negative by Equation (21) and can be computed by Bob. The second part is the same as the second part in Equation (20). It is non-negative by definition and can be computed by Alice. This completes the proof.

[8] For instance, to compute $s'(I_{\text{large}} \cap F_1 \cap \widetilde{F}_1)$, Alice samples uniformly $i \in [n]$ and sends it to Bob, Bob responds with $b = 1$ if $i \in \widetilde{F}_1$ and $b = 0$ otherwise. Alice then outputs $n \cdot s'_i$ if $i \in I_{\text{large}} \cap F_1$ and $b = 1$, and 0 otherwise. The protocols for the second and the third term are very similar.

A.3 Randomized Protocol for Arbitrary Feasible Solutions

We now extend the communication protocol of canonical feasible solutions to arbitrary feasible solutions. Denote by $\mathcal{R} = \{(x^1, y^1), (x^2, y^2), \ldots, (x^r, y^r)\}$ the set of all canonical feasible solutions. Consider an infeasible input (A, F_1, F_2) to Alice as earlier, and an input (x, y) to Bob where in this case (x, y) is feasible but not necessarily canonical. Lemma 6 below shows that we can write (x, y) as a convex combination of canonical feasible solutions $(x^1, y^1), (x^2, y^2), \ldots, (x^r, y^r)$. In other words, there exists $\lambda_1, \lambda_2, \ldots, \lambda_r \geq 0$, $\sum_{k=1}^{r} \lambda_k = 1$, and

$$(x, y) = \sum_{k=1}^{r} \lambda_k (x^k, y^k)\,.$$

We write Equation (10) as

$$\begin{aligned}
&s'(F_1 \cap B) + x(F_2 \cap B) - \alpha U \\
&= \sum_{i \in F_1} s'_i \sum_{k=1}^{r} \lambda_k y_i^k + \sum_{i \in F_2} \sum_{k=1}^{r} \lambda_k x_i^k D - \alpha U \\
&= \sum_{k=1}^{r} \lambda_k \left(\sum_{i \in F_1} s'_i y_i^k + \sum_{i \in F_2} x_i^k D - \alpha U \right)
\end{aligned}$$

In order to compute the slack (10) in expectation given a feasible solution (x, y), Bob samples a canonical feasible solution $(x^k, y^k) \in \mathcal{R}$ with probability λ_k, then together with Alice, they compute the slack of

$$\sum_{i \in F_1} s'_i y_i^k + \sum_{i \in F_2} x_i^k D - \alpha U$$

as discussed in the previous section.

We now prove Lemma 6.

Lemma 6. *Let $\mathcal{R} = \{(x^1, y^1), (x^2, y^2), \ldots, (x^r, y^r)\}$ be the set of all the canonical feasible solutions for the flow cover problem, then any feasible solution (x, y) can be written as*

$$(x, y) = \sum_{k=1}^{r} \lambda_k (x^k, y^k)\,,$$

such that $\lambda_k \geq 0$ for all $1 \leq k \leq r$, and $\sum_k \lambda_k = 1$.

Proof. Given a feasible solution (x, y), define its support $\widetilde{F}^{x,y} = \{i : i \in F, \text{ and } y_i = 1\}$, and define the set $\mathcal{R}^{x,y}$ to be the set of all canonical feasible solutions whose support equals $\widetilde{F}^{x,y}$, i.e.,

$$\mathcal{R}^{x,y} = \{(x', y) : (x', y) \in \mathcal{R}\} \subseteq \mathcal{R}\,.$$

Without loss of generality, we assume that $\widetilde{F}^{x,y} = [n]$ to simplify the presentation.

We now consider the following polytope $P(y)$:

$$P(y) = \left\{ z \in [0,1]^n : (*)\ \sum_{i=1}^{n} z_i = 1, \quad (**)\ 0 \leq z_i \leq \frac{s_i y_i}{D} \text{ for all } 1 \leq i \leq n \right\}.$$

Note that for any feasible solution (x, y) to the flow cover problem, $x \in P(y)$. For a canonical feasible solution $(x', y) \in \mathcal{R}^{x,y}$, by definition for all except at most one item $i \in [n]$, we have that either $x'_i = 0$ or $x'_i D = s_i y_i$, thus x' satisfies at least $n-1$ linearly independent constraints of type $(**)$ with equality. Conversely, if a point $x \in P(y)$ satisfies at least $n-1$ constraints of type $(**)$ with equality, then $(x, y) \in \mathcal{R}^{x,y}$.

Recall that a point z is an extreme point solution of $P(y)$ iff there are n linearly independent constraints that are set to equality by z. Since constraint $(*)$ is an equality constraint and is linearly independent from any set of $n-1$ constraints from $(**)$, we conclude that $\{x' : (x', y) \in \mathcal{R}^{x,y}\}$ is the set of all extreme points of $P(y)$. This implies that for any $x \in P(y)$, there exists $\lambda_k \geq 0$ for each $1 \leq k \leq r$ such that $\sum_k \lambda_k = 1$ and

$$x = \sum_{k=1}^{r} \lambda_k x^k \,.$$

Since all these points have the same y-support, it follows that

$$(x, y) = \sum_{k=1}^{r} \lambda_k (x^k, y^k) \,.$$

□

B Algorithmic Aspects

Theorem 1 relies on the *existence* of a quasi-polynomial size extended formulation for the weakened knapsack cover inequalities. However, we do not know how to *construct* the full extended formulation in quasi-polynomial time. Nevertheless, there is a way to use the extended formulation algorithmically, which we describe here.

We adopt a more general point of view, since the findings of this section are applicable beyond the context of the knapsack cover inequalities. Consider any system of p inequalities $A_1 x \geqslant b_1, \ldots, A_p x \geqslant b_p$, and q solutions $x^{(1)}, \ldots, x^{(q)} \in \mathbb{R}^n$ of this system. In the context of the min-knapsack problem, the inequalities $A_i x \geqslant b_i$ ($i \in [p]$) are all the weakened knapsack cover inequalities and the solutions $x^{(j)}$ ($j \in [q]$) are all the feasible solutions $x \in \{0,1\}^n$. Typically, both p and q are exponentially large as functions of n.

To this data corresponds a slack matrix $S \in \mathbb{R}^{p \times q}_{\geqslant 0}$ defined by $S_{ij} := A_i x^{(j)} - b_i$. As observed by Yannakakis [36], every non-negative factorization $S = FV$ where $F \in \mathbb{R}^{p \times r}_{\geqslant 0}$ and $V \in \mathbb{R}^{r \times q}_{\geqslant 0}$ determines a system

$$\begin{aligned} A_i x - b_i = F_i y \quad \forall i \in [p] \\ y \geqslant 0 \end{aligned} \tag{22}$$

whose projection to the x-space gives a polyhedron $\{x \in \mathbb{R}^n \mid \exists y \in \mathbb{R}^r : Ax - b = Fy,\ y \geqslant 0\}$ containing each of the solutions $x^{(j)}$ and contained in each of the halfspaces $A_i x \geqslant b_i$.

Usually, the number p of equations in (22) is much bigger than both the number n of x-variables and rank r of the non-negative factorization. Thus the equation system is largely overdetermined and can be replaced by a smaller equivalent subsystem with at most $n + r$ equations. However, it is not obvious to tell efficiently what are the indices i for which the corresponding equation in (22) should be kept.

To avoid this difficulty, we assume that the way in which we want to use the extended formulation (shorthand: EF) $Ax - b = Fy$, $y \geqslant 0$ is to solve the LP $\min\{c^\intercal x \mid Ax \geqslant b\}$ for a *given* objective vector $c \in \mathbb{R}^n$, through the extended formulation.

For $I \subseteq [p]$, consider the linear program

$$\begin{aligned} \mathrm{LP}(I): \quad & \min c^\intercal x \\ & \text{s.t. } A_i x - b_i = F_i y \quad \forall i \in I \\ & y \geqslant 0\,. \end{aligned}$$

In fact, we will only need to consider sets I of size at most $n + r \ll p$.

Algorithm 1 solves the LP $\min\{c^\intercal x \mid Ax \geqslant b\}$ in several steps. In each step, it solves the smaller $\mathrm{LP}(I)$ where $I \subseteq [p]$ and calls a separation routine to check whether x^*, the x-part of the optimum solution found, satisfies $Ax \geqslant b$ or not. In the first case, it returns x^* and stops. In the second case, it adds the index i^* of any violated constraint to I and continues. At the beginning of the algorithm, I is initialized to $[n]$. To avoid technicalities, we assume that $\mathrm{LP}([n])$ is bounded. For the sake of concreteness, we assume furthermore that the n first inequalities of the system $Ax \geqslant b$ are the nonnegativity inequalities $x_1 \geqslant 0, \ldots, x_n \geqslant 0$, and that $c \in \mathbb{R}^n_{\geqslant 0}$.

Algorithm 1 Cutting-plane algorithm to solve $\min\{c^\intercal x \mid Ax \geqslant b\}$ through EF $Ax - b = Fy$, $y \geqslant 0$

```
initialize I ⟵ [n]
initialize feasible ⟵ false
repeat
    solve LP(I), get optimum solution (x*, y*)
    if there exists i* ∈ [p] such that A_{i*} x* < b_{i*} then
        add i* to I
    else
        set feasible ⟵ true
    end if
until feasible = true
return x*
```

To analyze the running time of the algorithm, we make the following assumptions:

- the size of each coefficient in (22) and each c_i is upper-bounded by $\Delta = \Delta(n)$;
- the separation problem (given $x^* \in \mathbb{R}^n$, find an index $i^* \in [p]$ such that $A_{i^*} x < b_{i^*}$ or report that no such index exists) can be solved in $T_{\mathrm{sep}}(n)$ time;
- each single equation in (22) can be written down in $T_{\mathrm{constr}}(n)$ time;
- $\mathrm{LP}(I)$ can be solved in time $T_{\mathrm{solve}}(n)$ for any set I of size at most $n + r$, where $r = r(n)$ is the rank of the nonnegative factorization giving rise to the extended formulation $Ax - b = Fy$, $y \geqslant 0$.

Notice that $T_{\mathrm{solve}}(n) = O(n^3(n + r)\Delta)$ if an interior point method is used to solve $\mathrm{LP}(I)$.

Lemma 7. *Under the above assumptions, the main loop of Algorithm 1 is executed at most $r + 1$ times. Thus the complexity of Algorithm 1 is $O(r \cdot (T_{\mathrm{solve}}(n) + T_{\mathrm{sep}}(n) + T_{\mathrm{constr}}(n)))$.*

Proof. The result follows directly from the simple observation that each time a new equation $A_{i^*} x - b_{i^*} = F_{i^*} y$ added to the system $A_i x - b_i = F_i y$ ($i \in I$), it is linearly independent from the current equations in the system. Notice that by assumption, the algorithm starts with n linearly independent constraints. By the above observation, we always have $|I| \leqslant n + r$. □

From now on, we assume that the non-negative factorization of the slack matrix S comes from a communication protocol with non-negative outputs computing S in expectation. The protocol is specified by a binary *protocol tree*, in which each internal node is owned either by Alice or Bob, and each leaf corresponds to an output of the protocol. At each internal node u owned by Alice, a *branching probability* $p_{\mathrm{branch}}(i,u) \in [0,1]$ is given for each input $i \in [p]$ of Alice. Similarly for each internal node v owned by Bob, we are given a branching probability $q_{\mathrm{branch}}(j,v) \in [0,1]$, where $j \in [q]$ is Bob's input. These branching probabilities specify the chance for the protocol of following the left branch. Finally, each leaf ℓ has a nonnegative number $\lambda(\ell) \in \mathbb{R}_{\geqslant 0}$ attached to it.

The corresponding extended formulation can be written as

$$\begin{aligned} A_i x - b_i &= \sum_{\ell \text{ leaf}} p_{\mathrm{reach}}(i,\ell) \cdot y_\ell \quad \forall i \in [p] \\ y_\ell &\geqslant 0 \quad \forall \ell \text{ leaf} \end{aligned} \tag{23}$$

where $p_{\mathrm{reach}}(i,u)$ denotes the probability of reaching node u of the protocol tree on any input pair of the form $(i,*)$.

Lemma 8. *Let Δ be any number that is at least $\max\{-\log(p_{\mathrm{reach}}(i,\ell)) \mid i \in [p],\ \ell$ leaf$,\ p_{\mathrm{reach}}(i,\ell) > 0\}$ and let h denote the height of the protocol tree. For any fixed $i \in [p]$, one can write down the right-hand side of the corresponding equation in* (23) *in $O(2^h \Delta \log \Delta \log\log \Delta)$ time and $O(2^h \Delta)$ space.*

Proof. Clearly, at the root of the protocol tree, we have $p_{\mathrm{reach}}(i,\mathrm{root}) = 1$. At an internal node u owned by Alice with left child v and right child w, we have $p_{\mathrm{reach}}(i,v) = p_{\mathrm{reach}}(i,u) \cdot p_{\mathrm{branch}}(i,u)$ and $p_{\mathrm{reach}}(i,w) = p_{\mathrm{reach}}(i,u) \cdot (1 - p_{\mathrm{branch}}(i,u)) = p_{\mathrm{reach}}(i,u) - p_{\mathrm{reach}}(i,v)$. In case u is owned by Bob, we simply have $p_{\mathrm{reach}}(i,v) = p_{\mathrm{reach}}(i,w) = p_{\mathrm{reach}}(i,u)$ since the behavior of the communication protocol at node u on input pair (i,j) is independent of i.

Using this, we can compute recursively $p_{\mathrm{reach}}(i,u)$ for all nodes u of the protocol tree, and thus for the leaves of the tree. All arithmetic operations are performed on numbers of at most $O(\Delta)$ bits. If we use the Schoolbook algorithm for subtraction and the Schönhage-Strassen algorithm for multiplication, we obtain the claimed bounds for the time- and space-complexity. □

Now, we discuss how Algorithm 1 and its analysis apply to the (weakened) knapsack cover inequalities and the corresponding slack matrix $(S^{\varepsilon}_{ab})_{a \in f^{-1}(0),\, b \in f^{-1}(1)}$ as in (5), where f is the weighted threshold function defining the knapsack. In order to do that, we first have to construct the protocol tree of the protocol described in the proof of Lemma 4. We claim that this can be done in time $(1/\varepsilon)^{O(1)} n^{O(\log n)}$.

The protocol has several deterministic parts (in which the branching probabilities are in $\{0,1\}$ locally). Each corresponds to the resolution of a Karchmer-Wigderson game. For writing down the corresponding subtrees of the protocol tree, we just need $\log^2(n)$-depth monotone circuits of fan-in 2 for computing certain truncations of the weighted threshold function f. The translation of the circuit into a protocol tree follows the standard construction of Karchmer and Wigderson [25]. For constructing the circuits, we rely either on the construction of Beimel and Weinreb [7, 8] or the simpler and more recent construction of Chen, Oliveira and Servedio [16]. Both constructions can be executed in $n^{O(1)}$ time.

The remaining parts of the protocol can be readily translated into the corresponding subtrees of the protocol tree.

Since the reaching probabilities in the protocol tree can be written down with $O(\log n)$ bits, each coefficient in the right-hand side of (23) can be written down in $O(\log n)$ bits. Assuming as before that all item sizes and demand can be written down with $O(n \log n)$ bits (which is without loss of generality), the coefficients of the left-hand side of (23) can be written down with $O(n \log n)$ bits. Therefore, we can take $\Delta = O(n \log n)$

From what precedes and Lemma 8, we have that $T_{\text{constr}}(n) = (1/\varepsilon)^{O(1)} n^{O(\log n)}$. Moreover, Lemma 4 gives $r(n) = (1/\varepsilon)^{O(1)} n^{O(\log n)}$.

For the separation routine, we deviate significantly from Algorithm 1: instead of using an exact separation routine (efficient exact separation of the knapsack cover inequalities is an open problem), we rely on a separation trick from Carr *et al.* [13]. That is, we simply check if the knapsack cover inequality for $A := \{i \in [n] \mid x_i^* \geqslant 1/2\}$ is satisfied. This is enough to guarantee that the modified Algorithm 1 computes a quantity that is within a $2 + \varepsilon$ factor of the integer optimum for that particular cost function c. Unfortunately, by relying on the pseudo-separation of Carr *et al.*, we cannot guarantee that the modified Algorithm 1 optimizes exactly over all weakened knapsack cover inequalities.

If we further assume that the coefficients of c can be written with $O(n \log n)$ bits, we conclude that one can find a $(2 + \varepsilon)$-approximation of $\min\{\sum_{i=1}^n c_i x_i \mid \sum_{i=1}^n s_i x_i \geqslant D,\ x \in \{0,1\}^n\}$ in time $(1/\varepsilon)^{O(1)} n^{O(\log n)}$, without relying on the ellipsoid algorithm, using our extended formulation.