Faster Algorithms for Integer Programs with Block Structure

Friedrich Eisenbrand
EPFL, 1015 Lausanne, Switzerland
friedrich.eisenbrand@epfl.ch

Christoph Hunkenschröder
EPFL, 1015 Lausanne, Switzerland
christoph.hunkenschroder@epfl.ch

Kim-Manuel Klein
EPFL, 1015 Lausanne, Switzerland
kim-manuel.klein@epfl.ch

Abstract

We consider integer programming problems $\max\{c^T x : \mathcal{A}x = b, l \leq x \leq u, x \in \mathbb{Z}^{nt}\}$ where $\mathcal{A}$ has a (recursive) block-structure generalizing *n-fold* integer programs which recently received considerable attention in the literature. An n-fold IP is an integer program where $\mathcal{A}$ consists of n repetitions of submatrices $A \in \mathbb{Z}^{r \times t}$ on the top horizontal part and n repetitions of a matrix $B \in \mathbb{Z}^{s \times t}$ on the diagonal below the top part. Instead of allowing only two types of block matrices, one for the horizontal line and one for the diagonal, we generalize the n-fold setting to allow for arbitrary matrices in every block. We show that such an integer program can be solved in time $n^2 t^2 \varphi \cdot (r\, s\, \Delta)^{\mathcal{O}(rs^2+sr^2)}$ (ignoring logarithmic factors). Here Δ is an upper bound on the largest absolute value of an entry of $\mathcal{A}$ and φ is the largest binary encoding length of a coefficient of c. This improves upon the previously best algorithm of Hemmecke, Onn and Romanchuk that runs in time $n^3 t^3 \varphi \cdot \Delta^{\mathcal{O}(st(r+t))}$. In particular, our algorithm is not exponential in the number t of columns of A and B.

Our algorithm is based on a new upper bound on the ℓ_1-norm of an element of the *Graver basis* of an integer matrix and on a proximity bound between the LP and IP optimal solutions tailored for IPs with block structure. These new bounds rely on the *Steinitz Lemma*.

Furthermore, we extend our techniques to the recently introduced *tree-fold IPs*, where we again present a more efficient algorithm in a generalized setting.

2012 ACM Subject Classification Theory of computation → Integer programming

Keywords and phrases n-fold, Tree-fold, Integer Programming

Digital Object Identifier 10.4230/LIPIcs.ICALP.2018.49

Related Version A full version of the paper is available at `https://arxiv.org/abs/1802.06289`.

Funding This work was supported by the Swiss National Science Foundation (SNSF) within the project *Convexity, geometry of numbers, and the complexity of integer programming (Nr. 163071)*.

45th International Colloquium on Automata, Languages, and Programming (ICALP 2018).
Editors: Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella;
Article No. 49; pp. 49:1–49:13

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

An *integer program (IP)* is an optimization problem of the form

$$\max\{c^T x : Ax = b,\, l \leq x \leq u,\, x \in \mathbb{Z}^n\} \tag{1}$$

which is described by a *constraint matrix* $A \in \mathbb{Z}^{m\times n}$, an *objective function* vector $c \in \mathbb{Z}^n$ a *right-hand side* vector $b \in \mathbb{Z}^m$ and *lower* and *upper bounds* $l \leq x \leq u$. Integer programming is one of the most important paradigms in the field of algorithms as a breadth of combinatorial optimization problems have an IP-model, see, e.g. [17, 20]. Since integer programming is NP-hard, there is a strong interest in restricted versions of integer programs that can be solved in polynomial time, while still capturing interesting classes of combinatorial optimization problems. A famous example is the class of integer programs with *totally unimodular* constraint matrix, capturing flow, bipartite matching, and shortest path problems for example. This setting has been extended to *bimodular* integer programming recently [1].

Another such polynomial-time solvable restriction is n-fold integer programming [6]. Given two matrices $A \in \mathbb{Z}^{r\times t}$ and $B \in \mathbb{Z}^{s\times t}$ and a vector $b \in \mathbb{Z}^{r+ns}$ for some $r, s, t, n \in \mathbb{Z}_+$. An n-fold Integer Program (n-fold IP) is an integer program (1) with constraint matrix

$$\mathcal{A} = \begin{pmatrix} A & A & \dots & A \\ B & 0 & \dots & 0 \\ 0 & B & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & B \end{pmatrix} \tag{2}$$

Clearly, one can assume that $t \geq r$ and $t \geq s$ holds, as linearly dependent equations can be removed. Notice that the number of variables of an n-fold integer program is $t \cdot n$. The best known algorithm to solve an n-fold IP is due to Hemmecke, Onn and Romanchuk [10] with a running time of $\mathcal{O}(n^3t^3\varphi) \cdot \Delta^{\mathcal{O}(st(r+t))}$, where Δ is the absolute value of the largest entry in $\mathcal{A}$ and φ is the logarithm of the largest absolute value of a component of c. For fixed Δ, r, s and t, the running time depends only polynomially (cubic) on the number of variables and is therefore more efficient than applying algorithms for general IPs based on lattice-basis reduction [12, 16] or dynamic programming [7, 19].

The n-fold setting has gained strong momentum in the last years, especially in the fields of *parameterized complexity* and *approximation algorithms*. An algorithm is *fixed parameter tractable (fpt)* with respect to a parameter k derived from the input, if its running time is of the form $f(k) \cdot n^{O(1)}$ for some computable function f. The result of Hemmecke et al. [10] shows that integer programming is fixed parameter tractable with respect to Δ, s, r and t.

This opens the possibility to model combinatorial optimization problems with a fixed parameter as an n-fold integer program, see for instance [3, 13] and thereby obtain novel new results in the area of parameterized complixity. Very recently Jansen, Klein, Maack and Rau [11] used n-fold IPs to formulate an enhanced *configuration IP*, that is capable to track additional properties of configurations. With this enhanced IP they were able to develop approximation algorithms for several scheduling problems that involve setups. Not only for the scheduling problems, but also in the design of efficient algorithms for string and social choice problems, n-fold IPs have been successfully applied [14, 15].

A generalization of the classical n-fold IP, called *tree-fold* IP, was very recently introduced by Chen and Marx [2]. A matrix A is of tree-fold structure, if it is of recursive n-fold

structure, i.e. the matrices $B^{(i)}$ in IP (2) are of n'-fold structure themselves, and so on. Chen and Marx presented an algorithm to solve tree-fold IPs which runs in time $f(L) \cdot n^3 \varphi$, where φ is the encoding length and L involves parameters of the tree like the height of the tree and the number of variables and rows of the involved sub-matrices. They applied the tree-fold IP to a special case of the traveling salesman problem, where m clients have to visit every node of a weighted tree and the objective is to minimize the longest tour over all clients. Using the framework of tree-fold IPs, they obtained an fpt algorithm with a running time of $f(K) \cdot |V|^{\mathcal{O}(1)}$, where K is the longest tour of a client in the optimal solution and V is the set of vertices of the tree. However, the function f involves a term with a tower of K exponents.

1.1 Graver Bases and Augmentation Algorithms

Before we discuss our contributions, we have to review the core concepts of the algorithm of Hemmecke, Onn and Romanchuk [10] in a nutshell.

Suppose we are solving a general integer program (1) with constraint matrix $A \in \mathbb{Z}^{m \times n}$ and that we have a feasible solution z_0 at hand. Let $z^\star$ be an optimal solution. The vector $z^\star - z_0$ lies in the kernel of A, i.e., $A(z^\star - z_0) = 0$. An integer vector $y \in \ker(A)$ is called a *cycle* of A. Two vectors $u, v \in \mathbb{R}^n$ are said to be *sign compatible* if $u_i \cdot v_i \geq 0$ for each i. A cycle $y \in \ker(A)$ is *indecomposable* if it is not the sum of two sign-compatible and non-zero cycles of A. The set of indecomposable and integral elements from the kernel of A is called the *Graver basis* of A, [8], see also [18, 5].

A result of Cook, Fonlupt and Schrijver [4] implies that there exist $2n$ Graver-basis elements $g_1, \ldots, g_{2n} \in \ker(A)$ each sign compatible with $z^\star - z_0$ such that

$$z^\star - z_0 = \sum_{i=1}^{2n} \lambda_i g_i$$

holds for $\lambda_i \in \mathbb{N}_0$. For each i one has that $z_0 + \lambda_i g_i$ is a feasible integer solution of (1). Furthermore, there exists one i with $c^T(z^\star - z_0)/(2\,n) \leq \lambda_i c^T g_i$. Thus there exists an element g of the Graver basis of A and a positive integer $\lambda \in \mathbb{N}$ such that $z_0 + \lambda\, g$ is feasible and the gap to the optimum value has been reduced by a factor of $1 - 1/(2\,n)$.

Why should it be any simpler to find such an *augmenting vector* g as above? The crucial ingredient that is behind the power of this approach are *bounds* on the ℓ_1-norm of elements of the Graver basis of A. In some cases, these bounds are much more restrictive than the original lower and upper bounds $l \leq x \leq u$ and thus help in dynamic programming. In fact, each element g of the Graver basis of A has ℓ_1-norm bounded by $\|g\|_1 \leq \delta \cdot (n - m)$ where δ is the largest absolute value of a sub-determinant of A, see [18]. Applying the Hadamard bound, this means that

$$\|g\|_1 \leq m^{m/2} \Delta^m \cdot (n - m), \tag{3}$$

where Δ is a largest absolute value of an entry of A. Let us denote $m^{m/2}\Delta^m \cdot (n-m)$ by G_A. In order to find an augmenting solution which reduces the optimality gap by a factor of (roughly) $1 - 1/n$ one solves the following *augmentation integer program* with a *suitable* λ,

$$\max\{c^T y : Ay = 0,\, l - z_0 \leq \lambda \cdot y \leq u - z_0,\, \|y\|_1 \leq G_A, y \in \mathbb{Z}^n\}. \tag{4}$$

and replaces z_0 by $z_0 + \lambda \cdot y^\star$, where $y^\star$ is the optimal solution of (4). The number of augmenting steps can be bounded by $\mathcal{O}(n \log(c^T(z^\star - z_0)))$.

At first sight, it seems that one has not gained much with this approach, except that the right-hand side vector b has disappeared. In the case of n-fold integer programming however,

the ℓ_1-norm of an element of the Graver basis of $\mathcal{A}$ is bounded by a function in r, s, t and Δ and thus much smaller than the bound (3). This can be exploited in dynamic programming approaches.

Contributions of this paper. We present several elementary observations that, together, result in a much faster algorithm for integer programs with block structure including n-fold and tree-fold integer programs. We start with the following.

i) The ℓ_1-norm of an element of the Graver basis of a given matrix $A \in \mathbb{Z}^{m\times n}$ is bounded by $(2\,m\cdot\Delta+1)^m$, where Δ is an upper bound on the absolute value of each entry of A. This is shown with the Steinitz lemma and uses similar ideas as in [7]. Compared to the previous best bound (3), this new bound is independent on the number of columns n of A.

We then turn our attention to integer programming problems

$$\max\{c^T x : \mathcal{A}\,x = b,\, l \le x \le u,\, x \in \mathbb{Z}^{n\times t}\} \tag{5}$$

with constraint matrix of the form

$$\mathcal{A} = \begin{pmatrix} A^{(1)} & A^{(2)} & \dots & A^{(n)} \\ B^{(1)} & 0 & \dots & 0 \\ 0 & B^{(2)} & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & B^{(n)} \end{pmatrix},$$

where $A^{(1)},\dots,A^{(n)} \in \mathbb{Z}^{r\times t}$ and $B^{(1)},\dots,B^{(n)} \in \mathbb{Z}^{s\times t}$ are arbitrary matrices. This is a more general setting than n-fold integer programming, since the matrices on the top line and on the diagonal respectively do not have to repeat. In this setting, we obtain the following results.

ii) The ℓ_1-norm of an element of the Graver basis of $\mathcal{A}$ is bounded by $\mathcal{O}(r\,s\,\Delta)^{(r+1)(s+1)}$ which is independent on the number of columns t of the $A^{(i)}$ and $B^{(i)}$.

iii) We next provide a special proximity bound for integer programs with block structure (5). Let $x^\star$ be an arbitrary optimal solution of the linear programming relaxation of (5). We show that there exists an optimal solution $z^\star$ of (5) with

$$\|x^\star - z^\star\|_1 \le n\,t\,(r\,s\,\Delta)^{\mathcal{O}(r\,s)}.$$

iv) We then exploit the bounds ii) and iii) in a new dynamic program to solve (5). Its running time is bounded by

$$n^2t^2\varphi\log^2 nt\cdot(rs\Delta)^{\mathcal{O}(r^2s+rs^2)} + \mathrm{LP}$$

where φ denotes the largest binary encoding length of c, and LP denotes the time needed to solve the LP relaxation of (5).

The main advantage of the running time of our algorithm is the improved dependency on the parameter t. In contrast, the previous best known algorithm by Hemmecke, Onn and Romanchuk [10] for classical n-fold IPs invovles a term $\Delta^{\mathcal{O}(st^2)}$ and therefore has an exponential dependency on t. Recall that we can assume that $t \ge r, s$ holds. The number of columns t can be very large. Even if we do not allow column-repetitions, t can be as large as Δ^{r+s} and in applications involving *configuration IPs* this is often the order of magnitude one is dealing with. Knop, Koutecky and Mnich [14] improved the dependency of t in a special

setting of n-fold to a factor $t^{\mathcal{O}(r)}$. In their setting, the matrix B on the diagonal consists of one line of ones only. Our running time is an improvement of their result also in this case.

Next, we generalize the notion of tree-fold IPs of [2] where we allow for arbitrary matrices at each node. This yields a rather natural description of a generalized tree-fold IP. We refer to Section 4 for the precise definition.

In this setting we obtain the following result.

v) We present an algorithm for generalized tree-fold IPs with a running time that is roughly doubly exponential in the height of the tree (for a precise running time we refer to Lemma 10). With this algorithm we improve upon the algorithm by Chen and Marx [2], which has a running time involving a term that has a tower of τ exponents, where τ is the height of the tree.

vi) Using the tree-fold IP formulation of [2], this implies an fpt algorithm for the the traveling salesman problem on trees with m clients with running time $2^{2^{poly(K)}} \cdot |V|^{O(1)}$, where K is the longest tour of an optimal solution over all clients.

Notation. We use the following notation throughout this paper. For positive numbers $n, r, s, t \in \mathbb{N}$ and index $i = 1, \ldots, n$, let $A^{(i)} \in \mathbb{Z}^{r \times t}$, $B^{(i)} \in \mathbb{Z}^{s \times t}$ with $\left|\left|A^{(i)}\right|\right|_\infty, \left|\left|B^{(i)}\right|\right|_\infty \leq \Delta$ for some constant Δ. Columns of matrices are denoted with a lower index, i.e. the j-th column of the matrix $A^{(i)}$ is denoted by $A^{(i)}_j$, and so on. With $\log x$, we denote the logarithm to the basis 2 of some number x.

We will often subdivide the set of entries in a vector $y \in \mathbb{R}^{nt}$ or a vector $\mathcal{A}y \in \mathbb{R}^{r+ns}$ into *bricks*. A vector $y \in \mathbb{R}^{nt}$ will consist of n bricks with t variables each, i.e.

$$y^T = \left((y^{(1)})^T, \quad (y^{(2)})^T, \quad \ldots, \quad (y^{(n)})^T\right)$$

with the brick $y^{(i)} \in \mathbb{R}^t$ corresponding to the block $B^{(i)}$. A vector $g = \mathcal{A}y \in \mathbb{R}^{r+ns}$ will consist of $n+1$ bricks,

$$(\mathcal{A}y)^T = \left((g^{(0)})^T, \quad (g^{(1)})^T, \quad \ldots, \quad (g^{(n)})^T\right),$$

where the first brick $g^{(0)} \in \mathbb{R}^r$ consists of the first r entries and corresponds to the block row $(A^{(1)}, \ldots, A^{(n)})$ of $\mathcal{A}$, and every other block $g^{(i)}$, $i \geq 1$, consists of s entries and corresponds to the block $B^{(i)}$. We will always use upper indices with brackets when referring to the bricks, and the indices will coincide with the index of the block $B^{(i)}$ they correspond to (except brick $g^{(0)}$). A simple but crucial observation we will use several times is the following. If y is a cycle of $\mathcal{A}$, then each brick $y^{(i)}$ is already a cycle of the matrix $B^{(i)}$.

2 The norm of a Graver-basis element

In this section, we provide the details of the contributions i) and ii). We will make use of the following lemma of Steinitz [9, 21]. Here $\|\cdot\|$ denotes an arbitrary norm.

▶ **Lemma 1** (Steinitz Lemma). *Let $v_1, \ldots, v_n \in \mathbb{R}^m$ be vectors with $||v_i|| \leq \Delta$ for $i = 1, \ldots, n$. If $\sum_{i=1}^n v_i = 0$, then there is a reordering $\pi \in S_n$ such that for each $k \in \{1, \ldots, n\}$ the partial sum $p_k := \sum_{i=1}^k v_{\pi(i)}$ satisfies $||p_k|| \leq m\Delta$ (for the same norm $\|\cdot\|$).*

▶ **Lemma 2.** *Let $A \in \mathbb{Z}^{m \times n}$ be an integer matrix, let Δ be an upper bound on the absolute value of each component of A, and let $y \in \mathbb{Z}^n$ be an element of the Graver basis of A. Then $\|y\|_1 \leq (2m\Delta + 1)^m$.*

Proof. We define a sequence of vectors $v_1, \dots, v_{\|y\|_1} \in \mathbb{Z}^m$ in the following manner. If $y_j \geq 0$, we add y_j copies of the j-th column of A to the sequence, if $y_j < 0$ we add $|y_j|$ copies of the negative of column j to the sequence.

Clearly, the v_i sum up to zero and their ℓ_∞-norm is bounded by Δ. Using Steinitz, there is a reordering $u_{\pi(1)}, \dots, u_{\pi(\|y\|_1)}$ of this sequence s.t. each partial sum $p_k := \sum_{j=1}^{k} u_{\pi(j)}$ is bounded by $m\Delta$ in the l_∞-norm. Clearly,

$$|\{x \in \mathbb{Z}^m : \|x\|_\infty \leq m\Delta\}| = (2m\Delta + 1)^m .$$

Thus, if $\|y\|_1 > (2m\Delta + 1)^m$, then two of these partial sums are the same and we have a sequence $u_{\pi(k)} + \dots + u_{\pi(k+\ell)} = 0$. But then we can decompose y into two vectors corresponding to this sequence and the remaining vectors $u_{\pi(i)}$. This shows the claim. ◀

We will now apply the Steinitz lemma to bound the ℓ_1-norm of an element of the Graver basis of

$$\mathcal{A} = \begin{pmatrix} A^{(1)} & A^{(2)} & \dots & A^{(n)} \\ B^{(1)} & 0 & \dots & 0 \\ 0 & B^{(2)} & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & B^{(n)} \end{pmatrix},$$

where $A^{(1)}, \dots, A^{(n)} \in \mathbb{Z}^{r \times t}$ and $B^{(1)}, \dots, B^{(n)} \in \mathbb{Z}^{s \times t}$ are arbitrary matrices. Lemma 2 shows that the ℓ_1-norm of an element of the Graver basis of a matrix $B^{(i)}$ is bounded by $(2\,s\Delta + 1)^s =: L_B$.

▶ **Lemma 3.** *Let y be a Graver-basis element of $\mathcal{A}$, then*

$$||y||_1 \leq L_B\,(2r\Delta L_B + 1)^r =: L_{\mathcal{A}}.$$

Proof. Let g be a Graver basis element of $B^{(i)}$. Note that as $||g||_1 \leq L_B$ and $\left|\left|A^{(i)}\right|\right|_\infty \leq \Delta$, the infinity-norm of the vector $A^{(i)}g$ is bounded by

$$\left|\left|A^{(i)}g\right|\right|_\infty \leq \Delta L_B. \tag{6}$$

Now consider a Graver basis element $y \in \mathbb{Z}^{nt}$ of $\mathcal{A}$ and split it according to the matrices $B^{(i)}$ into bricks, i.e. $y^T = ((y^{(1)})^T, \dots, (y^{(n)})^T)$ with each $y^{(i)} \in \mathbb{Z}^t$ being a cycle of $B^{(i)}$. Hence, each $y^{(i)}$ can be decomposed into the sum of Graver basis elements $y_j^{(i)}$ of $B^{(i)}$, i.e. $y^{(i)} = y_1^{(i)} + \dots + y_{N_i}^{(i)}$. Thus, we have a decomposition

$$\begin{aligned} 0 &= (A^{(1)}, \dots, A^{(n)})y \\ &= A^{(1)}y^{(1)} + \dots + A^{(n)}y^{(n)} \\ &= A^{(1)}y_1^{(1)} + \dots + A^{(1)}y_{N_1}^{(1)} + \dots + A^{(n)}y_1^{(n)} + \dots + A^{(n)}y_{N_n}^{(n)} \\ &=: v_1 + \dots + v_N \in \mathbb{Z}^r \end{aligned}$$

for some $N = \sum_{i=1}^{n} N_i$ and $||v_i||_\infty \leq \Delta L_B$ for $i = 1, \dots, N$, using (6). Now we apply Steinitz to reorder the v_i s.t. each partial sum is bounded by $r\Delta L_B$ in the l_∞-norm. Again, if two partial sums were the same, we could decompose y, thus the number N of vectors

v_i is bounded by $(2r\Delta L_B + 1)^r$. Each v_i is of the form $A^{(j)}y_k^{(j)}$ for some j and k with $\|y_k^{(j)}\|_1 \leq L_B$, hence

$$\begin{aligned} ||y||_1 &\leq L_B \left(2r\Delta L_B + 1\right)^r \\ &= L_{\mathcal{A}}, \end{aligned}$$

finishing the proof. ◀

As $L_B \in \mathcal{O}(s\Delta)^s$ in our case, this shows $L_{\mathcal{A}} \in \mathcal{O}(rs\Delta)^{(r+1)(s+1)}$, as stated in point *ii*) in the previous chapter.

3 Solving the Generalized n-fold IP

Given a feasible solution x of the IP (5) we now follow the principle that we outlined in Section 1.1. There exists an element y of the Graver basis of $\mathcal{A}$ and a positive integer $\lambda \in \mathbb{N}$ such that $x + \lambda y$ is feasible and reduces the gap to the optimum value by a factor of $1 - 1/(2n)$. Suppose that we know λ. With our bound on $\|y\|_1 \leq L_A$ we will find an augmenting vector of at least this quality by solving (a relaxation of) the following *augmentation IP*:

$$\begin{aligned} \max\, & c^T y \\ & \mathcal{A}y = 0 \\ & ||y||_1 \leq L_{\mathcal{A}} \\ l - z \leq\, & \lambda y \leq u - z \\ & y \in \mathbb{Z}^{nt} \end{aligned} \tag{7}$$

The vector y we compute might violate the condition $||y||_1 \leq L_{\mathcal{A}}$, but we will have that $c^T y$ is at least as big as the optimum value of (7).

▶ **Lemma 4.** *Let λ be a fixed positive integer. In time $nt\,(rs\Delta)^{\mathcal{O}(r^2s+rs^2)}$, we can find an integral vector y with $Ay = 0$, $l - z \leq \lambda y \leq u - z$ and $c^T y \geq c^T y^\star$, where $y^\star$ is an optimum solution for (7).*

Proof. As λ is fixed, it will be convenient to rewrite the bounds on the variables as

$$\begin{aligned} & l^\star \leq y \leq u^\star \quad \text{with} \\ & l_i^\star = \max\left\{ \left\lceil \frac{l_i - z_i}{\lambda} \right\rceil, -L_{\mathcal{A}} \right\} \\ & u_i^\star = \min\left\{ \left\lfloor \frac{u_i - z_i}{\lambda} \right\rfloor, L_{\mathcal{A}} \right\}. \end{aligned} \tag{8}$$

In particular, $u^\star < \infty$. First observe that for each $y \in \mathbb{Z}^{nt}$ with $||y||_1 \leq L_{\mathcal{A}}$, one has

$$||\mathcal{A}y||_\infty \leq \Delta L_{\mathcal{A}}. \tag{9}$$

We can decompose $y = (y^{(1)}, \ldots, y^{(n)})$ into bricks according to the matrices $B^{(i)}$, and $B^{(k)}y^{(k)} = 0$ has to hold independently of the other variables. Let $U \subseteq \mathbb{Z}^{r+s}$ be the set of integer vectors of infinity norm at most $\Delta L_{\mathcal{A}}$. To find an optimal $y^\star$ for the augmentation IP (7) we construct the following acyclic digraph. There are two nodes 0_{start} and 0_{target}, together with nt copies of the set U, arranged in n blocks of t layers as

$$U_1^{(1)}, \ldots, U_t^{(1)}, U_1^{(2)}, \ldots, U_t^{(2)}, \ldots, U_1^{(n)} \ldots U_t^{(n)},$$

where the k-th block will correspond to the matrix

$$M^{(k)} := \begin{pmatrix} A^{(k)} \\ B^{(k)} \end{pmatrix}$$

(and thus to the brick $y^{(k)}$ of y). Writing $M^{(k)}_j$ for the j-th column of the matrix $M^{(k)}$, the arcs are given as follows. There is an arc from 0_{start} to $v \in U^{(1)}_1$ if there is an integer y_1 such that

$$v = y_1 M^{(1)}_1 \quad \text{and} \quad l^\star_1 \leq y_1 \leq u^\star_1$$

holds. The weight of this arc is $c_1 y_1$.

For two nodes $u \in U^k_{i-1}$ and $v \in U^k_i$ of two consecutive layers in the same block, we add an arc (u, v) if there is an integer $y_{(k-1)t+i}$ such that

$$v - u = y_{(k-1)t+i} M^{(k)}_i \quad \text{and} \quad l^\star_{(k-1)t+i} \leq y_{(k-1)t+i} \leq u^\star_{(k-1)t+i}$$

holds, i.e. if we can get from u to v by adding the i-th column of $\binom{A^{(k)}}{B^{(k)}}$ multiple times. The weight is $c_{(k-1)t+i} \cdot y_{(k-1)t+i}$. It remains to define the arcs between two blocks. If we fix a path through the whole block $U^{(k)}_1, \ldots, U^{(k)}_t$, this corresponds to fixing a brick $y^{(k)}$. Note that $M^{(k)} y^{(k)}$ has to be zero in the last s components, since continuing with this path in the next block will not change the entries of $\mathcal{A}y$ corresponding to $B^{(k)}$ any more. Thus, for placing an arc between two nodes $u \in U^k_t$ and $v \in U^{k+1}_1$ in two consecutive layers of different blocks, also the constraints $u_{r+1} = \cdots = u_{r+s} = 0$ have to be fulfilled.

Finally, we add arcs from $u \in U^{(n)}_t$ to 0_{target} if there exists an integer y_{nt} such that

$$-u = y_{nt} M^{(n)}_t \quad \text{and} \quad l^\star_{nt} \leq y_{nt} \leq u^\star_{nt}$$

holds. Again, the weight is $c_{nt} y_{nt}$.

Clearly, a longest $(0_{start} - 0_{target})$-path corresponds to an optimum solution of the augmentation IP (7), hence it is left to limit the time needed to find such a path.

The out-degree of each node is bounded by $u^\star_i - l^\star_i \leq 2L_{\mathcal{A}} + 1$ using (8). Therefore, the number of arcs is bounded by

$$\begin{aligned} nt \cdot |U| \cdot (2L_{\mathcal{A}} + 1) &= nt \left(2\Delta L_{\mathcal{A}} + 1\right)^{r+s} (2L_{\mathcal{A}} + 1) \\ &\leq nt \left(2\Delta L_{\mathcal{A}} + 1\right)^{r+s+1} \\ &\leq nt \left(2\Delta L_B \left(2r\Delta L_B + 1\right)^r + 1\right)^{r+s+1} \\ &= nt \cdot \mathcal{O}(\Delta r)^{r^2 s + r s^2 + o(r^2 s + r s^2)} \mathcal{O}(s)^{r^2 + rs + r}. \end{aligned}$$

We can find a shortest path by a Breadth-First Search in time linear in the number of edges. ◀

In the following lemma we consider the value $\Gamma := \max_i(u_i - l_i)$. In the case $u < \infty$, we can estimate $\Gamma \leq 2^\varphi$ and obtain a fixed running time in combination with Lemma 4. However, if there are variables present that are not bounded from above, we will combine this lemma with the proximity result of the next Section 3.1 which allows us to introduce artificial upper bounds $u' < \infty$.

▶ **Lemma 5.** *Consider the n-fold IP (5) with $u < \infty$. Let $\Gamma := \max_i(u_i - l_i)$. Given an initial feasible solution, we can find an optimum solution of the IP by solving the augmentation IP (7) for a constant $\lambda \in \mathbb{Z}_+$ at most*

$$\mathcal{O}\left(nt \log(\Gamma) \left(\log(nt\Gamma) + \varphi\right)\right)$$

times, where φ is the logarithm of the largest number occurring in the objective function c.

Proof (sketch). As previously discussed, for every feasible z there exists a pair (λ, y) s.t. $z + \lambda y$ is feasible and reducing the gap to the optimum value by $1 - 1/(2\,nt)$. This leads to roughly $nt \log(nt\,||c||_\infty \Gamma)$ iterations. If we only guess values for λ that are a power of 2, we only lose a constant factor but are able to limit the number of guesses. We refer to the full version of the paper for details. ◀

3.1 Proximity for n-fold IPs

If no explicit upper bounds are given (i.e. $u_i = \infty$ for some indices i), we cannot bound the number of necessary augmentation steps directly. To overcome this difficulty, we will present a proximity result in this section, stating that for an optimum rational solution $x^\star$, there exists an optimum integral solution $z^\star$ with $||x^\star - z^\star||_1 \leq ntL_{\mathcal{A}}$.

With this proximity result, we can first compute an optimum LP solution $x^\star$, and then introduce artificial box constraints $l(x^\star) \leq z \leq u(x^\star)$, depending on $x^\star$, knowing that at least one optimum IP solution lies within the introduced bounds.

▶ **Lemma 6.** *Let $x^\star$ be an optimum solution to the LP relaxation of (5). There exists an optimum integral solution $z^\star$ to (5) with*

$$||x^\star - z^\star||_1 \leq ntL_{\mathcal{A}} = nt(rs\Delta)^{\mathcal{O}(rs)}.$$

Proof. Let $x^\star$ be an optimum vertex solution of the LP relaxation of (5) and $z^\star$ be an optimum (integral) solution of (5) that minimizes the l_1-distance to $x^\star$.

We say a vector y *dominates* a cycle y' if they are sign-compatible and $|y'_i| \leq |y_i|$ for each i. The idea is to show that if the l_1-distance is too large, we can find a cycle dominated by $z^\star - x^\star$ and either add it to $x^\star$ or subtract it from $z^\star$ leading to a contradiction in both cases. However, as $z^\star - x^\star$ is fractional, we cannot decompose it directly but have to work around the fractionality.

To this end, denote with $\lfloor x^\star \rceil$ the vector $x^\star$ rounded towards $z^\star$ i.e. $\lfloor x^\star_i \rceil = \lfloor x^\star_i \rfloor$ if $z^\star_i \leq x^\star_i$ and $\lfloor x^\star_i \rceil = \lceil x^\star_i \rceil$ otherwise. Denote with $\{x^\star\}$ the fractional rest i.e. $\{x^\star\} = x^\star - \lfloor x^\star \rceil$. Consider the equation

$$\mathcal{A}\,(z^\star - x^\star) = \mathcal{A}\,(z^\star - \lfloor x^\star \rceil) - \mathcal{A}\{x^\star\} = 0.$$

Consider the integral vector $\mathcal{A}\{x^\star\}$. For each index i, we will obtain an integral vector w_i out of $\{x^\star\}_i \mathcal{A}_i$ by rounding the entries suitably such that

$$\mathcal{A}\{x^\star\} = \sum_{i=1}^{nt} (\{x^\star\}_i \mathcal{A}_i) = w_1 + \cdots + w_{nt}.$$

To be more formal, fix an index j and let $a_1, \ldots, a_{nt}$ denote the j-th entry of the vectors $\{x^\star\}_i \mathcal{A}_i$. Define $f := \left(\sum_{i=1}^{nt} a_i - \lfloor a_i \rfloor\right) \in \mathbb{Z}_+$ as the sum of the fractional parts. We round up f of the fractional entries a_i, and we round down all other fractional entries. If some a_i is integral already, it remains unchanged. After doing this for each component j, we obtain the vectors w_i as claimed. As $||\{x^\star\}||_\infty \leq 1$, each vector w_i is dominated by either $\mathcal{A}_i$ or $-\mathcal{A}_i$, in particular it inherits the zero entries.

Define the matrix

$$\mathcal{A}' := (w_1, \ldots, w_{nt})\,.$$

After permuting the columns, the matrix $(\mathcal{A}, -\mathcal{A}')$ has n-fold structure with parameters $r, s, 2t$. As Lemma 3 does not depend on t, the Graver basis elements of $(\mathcal{A}, -\mathcal{A}')$ are bounded

by $L_{\mathcal{A}}$ as well. We can now identify

$$\mathcal{A}(z^\star - x^\star) = (\mathcal{A}, -\mathcal{A}') \begin{pmatrix} z^\star - \lfloor x^\star \rceil \\ \mathbf{1}_{nt} \end{pmatrix} = 0,$$

and decompose the integral vector $\binom{z^\star - \lfloor x^\star \rceil}{\mathbf{1}_{nt}}$ into Graver basis elements of l_1-norm at most $L_{\mathcal{A}}$. But if

$$ntL_{\mathcal{A}} < ||z^\star - x^\star||_1 \leq \left|\left| \begin{pmatrix} z^\star - \lfloor x^\star \rceil \\ \mathbf{1}_{nt} \end{pmatrix} \right|\right|_1,$$

we obtain at least $nt + 1$ cycles. As $||\mathbf{1}_{nt}||_1 = nt$, this grants a cycle $\binom{\bar{y}}{\mathbf{0}_{nt}}$ and hence a cycle $\bar{y}$ of $\mathcal{A}$.

Case 1: $c^T\bar{y} \leq 0$: As $\bar{y}$ is dominated by $z^\star - \lfloor x^\star \rceil$, removing cycle $\bar{y}$ from the solution gives a new solution $\bar{z} = z^\star - \bar{y}$ with $c^T\bar{z} \geq c^T z^\star$, which is closer to the fractional solution $x^\star$. However, this contradicts the fact that $z^\star$ was chosen to be a solution with minimal distance $||x^\star - z^\star||_1$.

Case 2: $c^T\bar{y} > 0$: As we rounded $x^\star$ towards $z^\star$ and $\bar{y}$ is dominated by $z^\star - \lfloor x^\star \rceil$, we can add $\bar{y}$ to $x^\star$ and obtain a better solution, contradicting its optimality. ◀

We are now able to state our main theorem.

▶ **Theorem 7.** *The generalized n-fold IP (5) can be solved in time*

$$n^2t^2\varphi \log^2 nt \cdot (rs\Delta)^{\mathcal{O}(r^2s+rs^2)} + \boldsymbol{LP}$$

where φ denotes the logarithm of the largest number occurring in the input, and $\boldsymbol{LP}$ denotes the time needed to solve the LP relaxation of (5).

We refer to the full paper for a detailed analysis of the running time.

4 Tree-Fold IPs

Given matrices $A^{(i)} \in \mathbb{Z}^{m_i \times n}$ and vectors $b^{(i)} \in \mathbb{Z}^{m_i}$ for $i = 1, \ldots, N$ and $c, l, u \in \mathbb{Z}^n$ for some $n, N \in \mathbb{Z}_+$, $m_1, \ldots, m_N \in \mathbb{Z}^+$. We consider the following IP consisting of a system of (systems of) linear equations

$$\begin{aligned} \max\, & c^T x \\ & A^{(1)}x = b^{(1)} \\ & \qquad \vdots \\ & A^{(N)}x = b^{(N)} \\ & l \leq x \leq u \\ & x \in \mathbb{Z}^n. \end{aligned} \tag{10}$$

We call (10) a *tree-fold IP*, if for every matrix $A^{(i)}$ there is an index set $S(i)$ containing all indices of the non-zero columns of $A^{(i)}$, i.e.

$$S(i) \supseteq \left\{ j \mid A^{(i)}_j \neq 0 \right\},$$

such that the following two conditions hold. For all i, j, the sets $S(i), S(j)$ are either disjoint, or one of the sets is contained in the other. There is a matrix $A^{(i_0)}$ for which $S(i_0)$ contains all column indices.

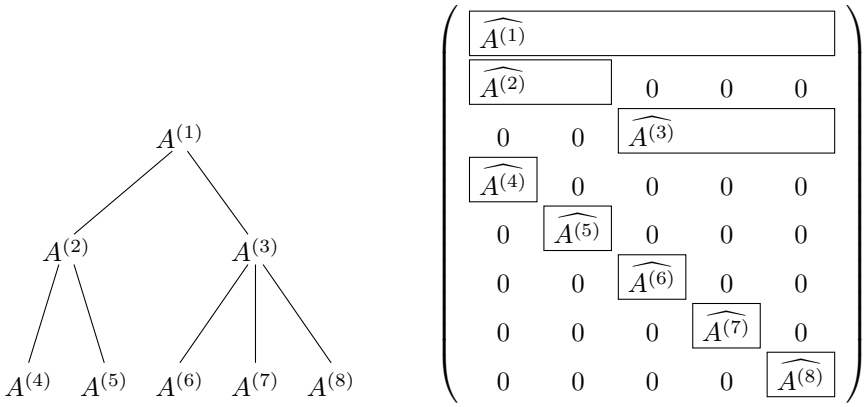

Figure 1 A matrix tree T and the induced tree-fold matrix $\mathcal{T}$, where $\widehat{A^{(i)}}$ denotes the part of $A^{(i)}$ that consists of the columns with index in $S(i)$. Note that if $A^{(1)}$ was not present, the set of columns of $\mathcal{T}$ could be bipartitioned into two sets orthogonal to each other.

Intuitively, the partial ordering induced by the sets $S(i)$ forms a tree T on the matrices $A^{(i)}$ (if the arcs stemming from transitivity are omitted). The root of this tree is the matrix $A^{(i_0)}$ with the largest set $S(i_0)$.

Analogously to our n-fold results, we will provide an upper bound on the l_1-norm of Graver basis elements of tree-fold matrices, together with a proximity result for optimum solutions. This will be sufficient to obtain an algorithm with a comparable running time.

Throughout this section, T will denote a tree as in Figure 1, we will denote the depth by τ and enumerate the layers starting at the deepest leaves (the leaves are not necessarily all in the same layer). The whole matrix induced by a tree-fold IP will be denoted by $\mathcal{T}$. This is, the IP (10) can be rewritten as

$$\begin{aligned} \max\, & c^T x \\ & \mathcal{T}x = b \\ & l \le x \le u \\ & x \in \mathbb{Z}^n \end{aligned} \tag{11}$$

▶ **Lemma 8.** *Let $\mathcal{T}$ be a tree-fold matrix where the corresponding matrix tree T has τ layers. Let the matrices of layer i have at most s_i rows and define $s = \prod_{i=1}^{\tau}(s_i+1)$ and $\Delta := ||\mathcal{T}||_\infty$. Then the Graver basis elements of $\mathcal{T}$ are bounded in their l_1-norm by*

$$L_\tau \le (3s\Delta)^{s-1}\,.$$

Proof (sketch). We enumerate the layers of T starting at the layer with the deepest leaves. We prove the claim by induction on the number τ of layers in the tree T. First observe that for $\tau = 1$, the claim follows by Lemma 2, as

$$L_1 \le (2s_1\Delta + 1)^{s_1} \le (3s\Delta)^{s-1}.$$

For the induction step, note that every child matrix $A^{(i)}$ of the root in T can be seen as the root matrix of a subtree T_i in T of depth $\tau - 1$ with at most $s_1, \dots, s_{\tau-1}$ rows in the corresponding layers. More formal, delete the root $A^{(1)}$ in T and let T_i be the connected component $A^{(i)}$ is in. Write

$$\widetilde{s} = \prod_{i=1}^{\tau-1}(s_i+1),$$

i.e. $s = \widetilde{s}(s_\tau + 1)$. By induction, we know that all Graver basis elements of the subtree-fold IPs $\mathcal{T}_i$ induced by T_i are bounded by

$$L_{\tau-1} \leq (3\widetilde{s}\Delta)^{\widetilde{s}-1} \leq (3s\Delta)^{\widetilde{s}-1}. \tag{12}$$

The rest of the induction step works similar to the proof of Lemma 3. We pick a cycle y of $\mathcal{T}$, decompose it into Graver basis elements for the subtree-fold matrices $\mathcal{T}_i$ and obtain a Steinitz sequence of vectors bounded by the induction hypothesis. However, due to space constraints, we omit the remaining part of this proof and refer to the full version of the paper. ◀

The following lemma states a proximity result for tree-fold IPs in the flavour of Lemma 6 for n-fold IPs. The proof uses that the bound in Lemma 8 only depends on the shape of the matrix but is independent of the number of columns in each block $A^{(i)}$, precisely as in Lemma 6. We refer to the full version of the paper for details on the proof.

▶ **Lemma 9.** *Let $\mathcal{T}$ be a matrix of tree-fold structure corresponding to the IP (10) and let $x^\star$ be an optimum solution to the LP relaxation of (10). There exists an optimum integral solution $z^\star$ to (10) with*

$$||x^\star - z^\star||_1 \leq nL_\tau.$$

We conclude with the following theorem that states the running time of our algorithm to solve a tree-fold IP. For a detailed analysis of the running time, we refer to the full version of the paper.

▶ **Theorem 10.** *Let $\mathcal{T}$ be of tree-fold structure with infinity-norm Δ and corresponding tree T. Let τ denote the number of layers of T and let the matrices of layer i have at most s_i rows.*

Define $s = \prod_{i=1}^{\tau}(s_i + 1)$ and $\sigma = \sum_{i=1}^{\tau} s_i$. Let n denote the number of columns of $\mathcal{T}$ and $l, u \in (\mathbb{Z} \cup \{\infty\})^n$. We can solve the IP (11),

$$\begin{aligned} \max\, & c^T x \\ & \mathcal{T}x = b \\ & l \leq x \leq u \\ & x \in \mathbb{Z}^n \end{aligned}$$

in time

$$n^2\varphi \log^2 n (s\Delta)^{\mathcal{O}(\sigma s)} + \boldsymbol{LP}$$

where φ denotes the logarithm of the largest number occurring in the input, and $\boldsymbol{LP}$ denotes the time needed to solve the LP relaxation of (11).

References

1 Stephan Artmann, Robert Weismantel, and Rico Zenklusen. A strongly polynomial algorithm for bimodular integer linear programming. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1206–1219. ACM, 2017.

2 Lin Chen and Dániel Marx. Covering a tree with rooted subtrees–parameterized and approximation algorithms. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2801–2820. SIAM, 2018.

3 Lin Chen, Dániel Marx, Deshi Ye, and Guochuan Zhang. Parameterized and approximation results for scheduling with a low rank processing time matrix. In *34th Symposium on Theoretical Aspects of Computer Science, STACS 2017, March 8-11, 2017, Hannover, Germany*, pages 22:1–22:14, 2017.

4 William Cook, Jean Fonlupt, and Alexander Schrijver. An integer analogue of caratheodory's theorem. *Journal of Combinatorial Theory, Series B*, 40(1):63–70, 1986.

5 Jesús A De Loera, Raymond Hemmecke, and Matthias Köppe. *Algebraic and geometric ideas in the theory of discrete optimization*. SIAM, 2012.

6 Jesús A De Loera, Raymond Hemmecke, Shmuel Onn, and Robert Weismantel. N-fold integer programming. *Discrete Optimization*, 5(2):231–241, 2008.

7 Friedrich Eisenbrand and Robert Weismantel. Proximity results and faster algorithms for integer programming using the Steinitz lemma. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 808–816. SIAM, 2018.

8 Jack E Graver. On the foundations of linear and integer linear programming i. *Mathematical Programming*, 9(1):207–226, 1975.

9 Victor S Grinberg and Sergey V Sevast'yanov. Value of the Steinitz constant. *Functional Analysis and Its Applications*, 14(2):125–126, 1980.

10 Raymond Hemmecke, Shmuel Onn, and Lyubov Romanchuk. N-fold integer programming in cubic time. *Mathematical Programming*, pages 1–17, 2013.

11 Klaus Jansen, Kim-Manuel Klein, Marten Maack, and Malin Rau. Empowering the configuration-ip - new PTAS results for scheduling with setups times. *CoRR*, abs/1801.06460, 2018. URL: `http://arxiv.org/abs/1801.06460`.

12 Ravi Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12(3):415–440, 1987.

13 Dušan Knop and Martin Koutecký. Scheduling meets n-fold integer programming. *Journal of Scheduling*, pages 1–11, 2017.

14 Dušan Knop, Martin Koutecký, and Matthias Mnich. Combinatorial n-fold Integer Programming and Applications. In Kirk Pruhs and Christian Sohler, editors, *25th Annual European Symposium on Algorithms (ESA 2017)*, volume 87 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 54:1–54:14, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

15 Dušan Knop, Martin Koutecký, and Matthias Mnich. Voting and bribing in single-exponential time. In *34th Symposium on Theoretical Aspects of Computer Science, STACS 2017, March 8-11, 2017, Hannover, Germany*, pages 46:1–46:14, 2017.

16 Hendrik W Lenstra Jr. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983.

17 George L Nemhauser and Laurence A Wolsey. Integer and combinatorial optimization. interscience series in discrete mathematics and optimization. *ed: John Wiley & Sons*, 1988.

18 Shmuel Onn. Nonlinear discrete optimization. *Zurich Lectures in Advanced Mathematics, European Mathematical Society*, 2010.

19 Christos H Papadimitriou. On the complexity of integer programming. *Journal of the ACM*, 28(4):765–768, 1981.

20 Alexander Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1998.

21 Ernst Steinitz. Bedingt konvergente reihen und konvexe systeme. *Journal für die reine und angewandte Mathematik*, 143:128–176, 1913.