


A Parameterized Strongly Polynomial Algorithm for Block Structured Integer Programs

Martin Koutecký¹

Technion – Israel Institute of Technology, Haifa, Israel, and

Charles University, Prague, Czech Republic

koutecky@technion.ac.il

 <https://orcid.org/0000-0002-7846-0053>

Asaf Levin²

Technion – Israel Institute of Technology, Haifa, Israel

levinas@ie.technion.ac.il

Shmuel Onn³

Technion – Israel Institute of Technology, Haifa, Israel

onn@ie.technion.ac.il

Abstract

The theory of n -fold integer programming has been recently emerging as an important tool in parameterized complexity. The input to an n -fold integer program (IP) consists of parameter A , dimension n , and numerical data of binary encoding length L . It was known for some time that such programs can be solved in polynomial time using $O(n^{g(A)}L)$ arithmetic operations where g is an exponential function of the parameter. In 2013 it was shown that it can be solved in fixed-parameter tractable time using $O(f(A)n^3L)$ arithmetic operations for a single-exponential function f . This, and a faster algorithm for a special case of *combinatorial* n -fold IP, have led to several very recent breakthroughs in the parameterized complexity of scheduling, stringology, and computational social choice. In 2015 it was shown that it can be solved in strongly polynomial time using $O(n^{g(A)})$ arithmetic operations.

Here we establish a result which subsumes all three of the above results by showing that n -fold IP can be solved in strongly polynomial fixed-parameter tractable time using $O(f(A)n^6 \log n)$ arithmetic operations. In fact, our results are much more general, briefly outlined as follows.

- There is a strongly polynomial algorithm for integer linear programming (ILP) whenever a so-called Graver-best oracle is realizable for it.
- Graver-best oracles for the large classes of multi-stage stochastic and tree-fold ILPs can be realized in fixed-parameter tractable time. Together with the previous oracle algorithm, this newly shows two large classes of ILP to be strongly polynomial; in contrast, only few classes of ILP were previously known to be strongly polynomial.
- We show that ILP is fixed-parameter tractable parameterized by the largest coefficient $\|A\|_\infty$ and the primal or dual treedepth of A , and that this parameterization cannot be relaxed, signifying substantial progress in understanding the parameterized complexity of ILP.

2012 ACM Subject Classification Theory of computation → Fixed parameter tractability

Keywords and phrases integer programming, parameterized complexity, Graver basis, n -fold integer programming

Digital Object Identifier 10.4230/LIPIcs.ICALP.2018.85

¹ Supported by a Technion postdoctoral fellowship and the project 17-09142S of GA ČR.

² Supported by a grant from the GIF, the German-Israeli Foundation for Scientific Research and Development (grant number I-1366-407.6/2016).

³ Supported by the Dresner chair.



© Martin Koutecký, Asaf Levin, and Shmuel Onn;
licensed under Creative Commons License CC-BY

45th International Colloquium on Automata, Languages, and Programming (ICALP 2018).

Editors: Ioannis Chatzigiannakis, Christos Kaklamani, Dániel Marx, and Donald Sannella;
Article No. 85; pp. 85:1–85:14



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



Related Version A full version of the paper is available at <https://arxiv.org/abs/1802.05859>.

1 Introduction

In this article we consider the general linear integer programming (ILP) problem in standard form,

$$\min \{ \mathbf{w}\mathbf{x} \mid A\mathbf{x} = \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}, \mathbf{x} \in \mathbb{Z}^n \}. \quad (\text{ILP})$$

with A an integer $m \times n$ matrix, $\mathbf{b} \in \mathbb{Z}^m$, $\mathbf{w} \in \mathbb{Z}^n$, $\mathbf{l}, \mathbf{u} \in (\mathbb{Z} \cup \{\pm\infty\})^n$. It is well known to be strongly NP-hard, which motivates the search for tractable special cases.

The first important special case is ILP in fixed dimension. In the '80s it was shown by Lenstra and Kannan [17, 20] that (ILP) can be solved in time $n^{O(n)}L$, where L is the length of the binary encoding of the input. Secondly, it is known that if the matrix A is totally unimodular (all subdeterminants between -1 and 1), all vertices of the feasible region are integral and thus applying any polynomial algorithm for linear programming (LP) suffices. Later, Veselov and Chirkov [25] have shown that the more general class of bimodular ILP is also polynomial-time solvable. Other results exploit certain structural properties of A . These include the large classes of n -fold [13], tree-fold [4], 2-stage and multi-stage stochastic [3], and 4-block n -fold [12] ILPs, as well as algorithms for ILPs with bounded treewidth [11], treedepth [10] and fracture number [7] of certain graphs related to the matrix A .

A fundamental question regarding problems involving large numbers is whether there exists an algorithm whose number of arithmetic operations does not depend on the length of the numbers involved; if this number is polynomial, this is a *strongly polynomial algorithm* [24]. For example, the ellipsoid method or the interior-point method which solve LP take time which does depend on the encoding length, and the existence of a strongly polynomial algorithm for LP remains a major open problem. So far, the only strongly polynomial ILP algorithms we are aware of exist for totally unimodular ILP [14], bimodular ILP [2], so-called binet ILP [1], and n -fold IP with constant block dimensions [6]. All remaining results, such as Lenstra's famous algorithm or the fixed-parameter tractable algorithm for n -fold IP which has recently led to several breakthroughs [4, 16, 18, 19], are not strongly polynomial.

1.1 Our Contributions

To clearly state our results we introduce the following terminology. The input to a problem will be partitioned into three parts (α, β, γ) , where α is the *parametric input*, β is the *arithmetic input*, and γ is the *numeric input*. A *strongly fixed-parameter tractable (FPT) algorithm* for the problem is one that solves it using $f(\alpha)\text{poly}(\beta)$ arithmetic operations and $g(\alpha)\text{poly}(\beta, \gamma)$ time, where f, g are some computable functions. If such an algorithm exists, we say that the problem is *strongly fixed-parameter tractable (FPT) parameterized by α* . Thus, such an algorithm both demonstrates that the problem is FPT *parameterized by α* because it runs in FPT time $g(\alpha)\text{poly}(\beta, \gamma)$, and provides a strongly polynomial algorithm for each fixed α . Having multiple parameters $\alpha_1, \dots, \alpha_k$ simultaneously is understood as taking the *aggregate parameter* $\alpha = \alpha_1 + \dots + \alpha_k$. If the algorithm involves oracles then the oracle queries are also counted as arithmetic operations and the answers to oracle queries should be polynomial in (β, γ) . Each part of the input may have several entities, which may be presented in unary or binary, where $\langle e \rangle$ denotes the encoding length of an entity e presented in binary. For the parametric input the distinction between unary and binary is irrelevant.

ILP abounds in natural parameters: the dimension n , number of inequalities m , largest coefficient $\|A\|_\infty$, largest right-hand side $\|\mathbf{b}\|_\infty$, various structural parameters of A , etc. Here, we are interested in algorithms which are both strongly polynomial and FPT.

Recently it was shown that, if we have access to the so-called *Graver basis* of A , the problem (ILP) is polynomial time solvable even for various nonlinear objective functions [5, 21]. We show that all of these results can be extended to be strongly polynomial with only $\langle A \rangle$ as the arithmetic input.

► **Theorem 1.** *The problem (ILP) with arithmetic input $\langle A \rangle$ and numeric input $\langle \mathbf{w}, \mathbf{b}, \mathbf{l}, \mathbf{u} \rangle$, endowed with a Graver-best oracle for A , is solvable by a strongly polynomial oracle algorithm.*

The existence of Graver-best oracles is thus of prime interest. We show such oracles for the wide classes of multi-stage stochastic and tree-fold ILPs; for precise definitions of these classes cf. Section 3.1.2. See Table 1 for a summary of improvements over the current state of the art.

► **Theorem 2.** *Multi-stage stochastic ILP with blocks B_1, \dots, B_τ , $B_i \in \mathbb{Z}^{l \times n_i}$, is strongly FPT parameterized by $l + n_1 + \dots + n_\tau$ and $\|A\|_\infty$.*

► **Theorem 3.** *Tree-fold ILP with blocks A_1, \dots, A_τ , $A_i \in \mathbb{Z}^{r_i \times t}$, is strongly FPT parameterized by $r_1 + \dots + r_\tau$ and $\|A\|_\infty$.*

This improves on the algorithm for tree-fold ILP [4] not only by making it strongly FPT, but also by leaving the block length t out of the parameter. Similarly, the following algorithm for the special case of n -fold ILP greatly improves both on the previous results of Hemmecke et al. [13] and Knop et al. [18] and is the currently fastest algorithm for this problem:

► **Theorem 4.** *n -fold ILP with blocks $A_1 \in \mathbb{Z}^{r \times t}$ and $A_2 \in \mathbb{Z}^{s \times t}$ can be solved in time $a^{O(r^2s+rs^2)}(nt)^6 \log(nt) + \mathcal{L}(\langle A \rangle)$, where $\mathcal{L}(\langle A \rangle)$ is the runtime of a strongly polynomial LP algorithm.*

Next, we turn our attention to structural parameters of the constraint matrix A . We focus on two graphs which can be associated with A :

- the *primal graph* $G_P(A)$, which has a vertex for each column and two vertices are connected if there exists a row such that both columns are non-zero, and,
- the *dual graph* $G_D(A) = G_P(A^T)$, which is the above with rows and columns swapped.

Two standard parameters of structural sparsity are the *treewidth* (measuring the “tree-likeness” of a graph) and the more restrictive *treedepth* (measuring its “star-likeness”). We denote the treewidth of $G_P(A)$ and $G_D(A)$ by $\text{tw}_P(A)$ and $\text{tw}_D(A)$; for treedepth we have $\text{td}_P(A)$ and $\text{td}_D(A)$. Note that bounded treedepth implies bounded treewidth but not vice versa.

We show that ILP parameterized by $\text{td}_P(A) + \|A\|_\infty$ and $\text{td}_D(A) + \|A\|_\infty$ can be reduced to the previously mentioned classes, respectively, implying (ILP) with these parameters is strongly FPT.

► **Theorem 5.** *(ILP) is strongly FPT parameterized by $\text{td}_P(A)$ and $\|A\|_\infty$.*

This improves in two ways upon the result of Ganian and Ordyniak [10] who show that (ILP) with $\mathbf{w} \equiv \mathbf{0}$ (i.e. deciding the feasibility) is FPT parameterized by $\text{td}_P(A) + \|A, \mathbf{b}\|_\infty$ [10]. First, we use the smaller parameter $\|A\|_\infty$ instead of $\|A, \mathbf{b}\|_\infty$, and second, we solve not only the feasibility but also the optimization problem. An analogous result holds for the parameter $\text{td}_D(A)$, for which previously nothing was known at all.

► **Theorem 6.** *(ILP) is strongly FPT parameterized by $\text{td}_D(A)$ and $\|A\|_\infty$.*

■ **Table 1** Run time improvements implied by this paper. We denote by L the binary length of the numeric input $\mathbf{b}, \mathbf{l}, \mathbf{u}, \mathbf{w}$, i.e., $L = \langle \mathbf{b}, \mathbf{l}, \mathbf{u}, \mathbf{w} \rangle$, and consider $\langle A \rangle$ to be part of the arithmetic input. We denote by $a = \max\{2, \|A\|_\infty\}$, by r, s, t the relevant block dimensions (cf. Section 3.1), and by $\mathcal{L}(\langle A \rangle)$ the runtime of a strongly polynomial LP algorithm [24].

Type of instance	Previous best run time	Our result
n -fold ILP	$a^{O(rst+st^2)}n^3L$ [13]	$a^{O(r^2s+sr^2)}(nt)^6 \log(nt) + \mathcal{L}(\langle A \rangle)$ Thm 4
n -fold ILP	$n^{f_1(a,r,s,t)}$ [6]	
n -fold ILP	$t^{O(r)}(ar)^{r^2}n^3L$ if $A_2 = (1 \ 1 \ \dots \ 1)$ [18]	$a^{O(r^2)}(nt)^6 \log(nt) + \mathcal{L}(\langle A \rangle)$ Thm 4
tree-fold ILP	$f_{\text{tf}}(a, n_1, \dots, n_\tau, t)n^3L$ [4]	$f_{\text{tf}}(a, n_1, \dots, n_\tau)(nt)^3 + \mathcal{L}(\langle A \rangle)$ Thm 3
Multi-stage stochastic ILP	$f_{\text{mss}}(a, n_1, \dots, n_\tau, l)n^3L$ [3]	$f_{\text{mss}}(a, n_1, \dots, n_\tau, l)n^3 + \mathcal{L}(\langle A \rangle)$ Thm 2
Bounded dual treedepth	Open whether fixed-parameter tractable	$f_D(a, \text{td}_D(A))(nt)^3 + \mathcal{L}(\langle A \rangle)$ Thm 6
Bounded primal treedepth	$f_{P'}(a, \ \mathbf{b}\ _\infty, \text{td}_P(A))nL$ [10]	$f_P(a, \text{td}_P(A))n^3 + \mathcal{L}(\langle A \rangle)$ Thm 5

We emphasize that the parameterizations cannot be relaxed neither from treedepth to treewidth, nor by removing the parameter $\|A\|_\infty$: (ILP) is NP-hard already on instances with $\text{tw}_P(A) = 3$ and $\|A\|_\infty = 2$ [10, Thm 12], and it is strongly W[1]-hard parameterized by $\text{td}_P(A)$ alone [10, Thm 11]; the fact that a problem is W[1]-hard is strong evidence that it is not FPT. Similarly, deciding feasibility is NP-hard on instances with $\text{tw}_D(A) = 3$ and $\|A\|_\infty = 2$ (Lemma 18) and strongly W[1]-hard parameterized by $\text{td}_D(A)$ alone [19, Thm 5].

1.2 Interpretation of Results

We believe our approach also leads to several novel insights. First, we make it clear that the central question is finding Graver-best oracles; provided these oracles, Theorem 1 shows that tasks such as optimization and finding initial solutions can be handled under very mild assumptions. Even though we show these tasks are routine, they have been reimplemented repeatedly [4, 12, 13, 18].

Second, we show that the special classes of highly uniform block structured ILPs, namely multi-stage stochastic and tree-fold ILPs, are in some sense *universal* for all ILPs of bounded primal or dual treedepth, respectively. Specifically, we show that any ILP with bounded primal or dual treedepth can be embedded in an equivalent multi-stage stochastic or tree-fold ILP, respectively (Lemmas 25 and 26).

Third, we show that, besides bounded primal or dual treedepth, the crucial property for efficiency is the existence of augmenting steps with bounded ℓ_∞ - or ℓ_1 -norms, respectively (Lemmas 19 and 21). This suggests that for ILPs whose primal or dual graph is somehow “sparse” and “shallow”, finding augmenting steps of bounded ℓ_∞ - or ℓ_1 -norm might be both sufficient for reaching the optimum and computationally efficient.

1.3 Related Work

We have already covered all relevant work regarding strongly polynomial algorithms for ILP.

Let us focus on structural parameterizations. It follows from Freuder’s algorithm [9] and was reproven by Jansen and Kratsch [15] that (ILP) is FPT parameterized by $\text{tw}_P(A)$ and the largest domain $\|\mathbf{u} - \mathbf{l}\|_\infty$. Regarding the dual graph $G_D(A)$, the parameters $\text{td}_D(A)$ and $\text{tw}_D(A)$ were only recently considered by Ganian et al. [11]. They show that even deciding feasibility of (ILP) is NP-hard on instances with $\text{tw}_I(A) = 3$ ($\text{tw}_I(A)$ denotes the treewidth of the *incidence graph*; $\text{tw}_I(A) \leq \text{tw}_D(A) + 1$ always holds) and $\|A\|_\infty = 2$ [11, Theorem 12].

Furthermore, they show that (ILP) is FPT parameterized by $\text{tw}_I(A)$ and parameter Γ , which is an upper bound on any prefix sum of $A\mathbf{x}$ for any feasible solution \mathbf{x} .

Dvořák et al [7] introduce the parameter fracture number; having a bounded *variable fracture number* $\mathfrak{p}^V(A)$ implies that deleting a few columns of A breaks it into independent blocks of small size; similarly for *constraint fracture number* $\mathfrak{p}^C(A)$ and deleting a few rows. Because bounded $\mathfrak{p}^V(A)$ implies bounded $\text{td}_P(A)$ and bounded $\mathfrak{p}^C(A)$ implies bounded $\text{td}_D(A)$, our results generalize theirs. The remaining case of *mixed fracture number* $\mathfrak{p}(A)$, where deleting both rows and columns is allowed, reduces to the 4-block n -fold ILP problem, which is not known to be either FPT or $W[1]$ -hard. Because bounded $\mathfrak{p}(A)$ implies bounded $\text{td}_I(A)$, ILP parameterized by $\text{td}_I(A) + \|A\|_\infty$ is at least as hard as 4-block n -fold ILP, highlighting its status as an important open problem.

Organization. The paper contains three main parts. In Section 2, we provide the proof of Theorem 1, showing the existence of a strongly polynomial algorithm whenever a Graver-best oracle is provided. Then, in Section 3, we provide Graver-best oracles for multi-stage stochastic and tree-fold ILPs and discuss n -fold ILP, and prove Theorems 2, 3 and 4. Finally, in Section 4 we show how to embed any instance of bounded primal or dual treedepth into a multi-stage stochastic or tree-fold ILP without increasing the relevant parameters, proving Theorems 5 and 6. Due to space restrictions the proofs of our technical statement and other supplementary material are moved to the full version available at <https://arxiv.org/abs/1802.05859>; the statements whose proofs are presented there are marked with (*).

2 The Graver-best Oracle Algorithm

2.1 Preliminaries

For positive integers m, n , $m \leq n$, we set $[m, n] = \{m, \dots, n\}$ and $[n] = [1, n]$. We write vectors in boldface (e.g., \mathbf{x}, \mathbf{y}) and their entries in normal font (e.g., the i -th entry of \mathbf{x} is x_i). If A is a matrix, A_r denotes its r -th column. For an integer $a \in \mathbb{Z}$, we denote by $\langle a \rangle = 1 + \log_2 a$ the binary encoding length of a ; we extend this notation to vectors, matrices and tuples of these objects. For example, $\langle A, \mathbf{b} \rangle = \langle A \rangle + \langle \mathbf{b} \rangle$, and $\langle A \rangle = \sum_{i,j} \langle a_{ij} \rangle$. For a graph G we denote by $V(G)$ its set of vertices.

Graver bases and augmentation. Let us now introduce Graver bases and discuss how they are used for optimization. We define a partial order \sqsubseteq on \mathbb{R}^n as follows: for $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ we write $\mathbf{x} \sqsubseteq \mathbf{y}$ and say that \mathbf{x} is *conformal* to \mathbf{y} if $x_i y_i \geq 0$ (that is, \mathbf{x} and \mathbf{y} lie in the same orthant) and $|x_i| \leq |y_i|$ for $i \in [n]$. It is well known that every subset of \mathbb{Z}^n has finitely many \sqsubseteq -minimal elements.

► **Definition 7** (Graver basis). The *Graver basis* of an integer $m \times n$ matrix A is the finite set $\mathcal{G}(A) \subset \mathbb{Z}^n$ of \sqsubseteq -minimal elements in $\{\mathbf{x} \in \mathbb{Z}^n : A\mathbf{x} = 0, \mathbf{x} \neq 0\}$.

We say that \mathbf{x} is *feasible* for (ILP) if $A\mathbf{x} = \mathbf{b}$ and $\mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$. Let \mathbf{x} be a feasible solution for (ILP). We call \mathbf{g} a *feasible step* if $\mathbf{x} + \mathbf{g}$ is feasible for (ILP). Further, call a feasible step \mathbf{g} *augmenting* if $\mathbf{w}(\mathbf{x} + \mathbf{g}) < \mathbf{w}(\mathbf{x})$. An augmenting step \mathbf{g} and a *step length* $\alpha \in \mathbb{Z}$ form an \mathbf{x} -feasible *step pair* with respect to a feasible solution \mathbf{x} if $\mathbf{l} \leq \mathbf{x} + \alpha \mathbf{g} \leq \mathbf{u}$. An augmenting step \mathbf{h} is a *Graver-best step* for \mathbf{x} if $\mathbf{w}(\mathbf{x} + \mathbf{h}) \leq \mathbf{w}(\mathbf{x} + \lambda \mathbf{g})$ for all \mathbf{x} -feasible step pairs $(\mathbf{g}, \lambda) \in \mathcal{G}(A) \times \mathbb{Z}$. The *Graver-best augmentation procedure* for (ILP) with a given feasible solution \mathbf{x}_0 works as follows:

1. If there is no Graver-best step for \mathbf{x}_0 , return it as optimal.
2. If a Graver-best step \mathbf{h} for \mathbf{x}_0 exists, set $\mathbf{x}_0 := \mathbf{x}_0 + \mathbf{h}$ and go to 1.

► **Proposition 8** ([21, Lemma 3.10]). *Given a feasible solution \mathbf{x}_0 for (ILP), the Graver-best augmentation procedure finds an optimum of (ILP) in at most $(2n - 2) \log F$ steps, where $F = \mathbf{w}\mathbf{x}_0 - \mathbf{w}\mathbf{x}^*$ and \mathbf{x}^* is any minimizer of $\mathbf{w}\mathbf{x}$.*

► **Definition 9** (Graver-best oracle). *A Graver-best oracle for an integer matrix A is one that, queried on $\mathbf{w}, \mathbf{b}, \mathbf{l}, \mathbf{u}$ and \mathbf{x} feasible to (ILP), returns a Graver-best step \mathbf{h} for \mathbf{x} .*

2.2 The Algorithm

It follows from Proposition 8 that given a Graver-best oracle, problem (ILP) can be solved in time which is polynomial in the binary encoding length $\langle A, \mathbf{w}, \mathbf{b}, \mathbf{l}, \mathbf{u} \rangle$ of the input. We now show that, in fact, given such an oracle, the problem admits a *strongly* polynomial algorithm. In the next theorem the input has only arithmetic and numeric parts and no parametric part.

► **Theorem 1.** *The problem (ILP) with arithmetic input $\langle A \rangle$ and numeric input $\langle \mathbf{w}, \mathbf{b}, \mathbf{l}, \mathbf{u} \rangle$, endowed with a Graver-best oracle for A , is solvable by a strongly polynomial oracle algorithm.*

► **Remark.** The partition of the input to the arithmetic input $\langle A \rangle$ and the numeric input $\langle \mathbf{w}, \mathbf{b}, \mathbf{l}, \mathbf{u} \rangle$ is the same as in the classical results for linear programming [8, 24].

Proof. The algorithm which demonstrates the theorem consists of several steps as follows.

Step 1: Reducing $\mathbf{b}, \mathbf{l}, \mathbf{u}$. Apply the strongly polynomial algorithm of Tardos [24] to the linear programming relaxation $\min \{ \mathbf{w}\mathbf{y} \mid \mathbf{y} \in \mathbb{R}^n, A\mathbf{y} = \mathbf{b}, \mathbf{l} \leq \mathbf{y} \leq \mathbf{u} \}$; the algorithm performs $\mathcal{L}(\langle A \rangle) = \text{poly}(\langle A \rangle)$ arithmetic operations. If the relaxation is infeasible then so is (ILP) and we are done. If it is unbounded then (ILP) is either infeasible or unbounded too, and in this case we set $\mathbf{w} := \mathbf{0}$ so that all solutions are optimal, and we proceed as below and terminate at the end of step 3. Suppose then that we obtain an optimal solution $\mathbf{y}^* \in \mathbb{R}^n$ to the relaxation, with round down $\lfloor \mathbf{y}^* \rfloor \in \mathbb{Z}^n$. Let $a := \max\{2, \|A\|_\infty\}$. Let $\mathcal{C}(A) \subseteq \mathcal{G}(A)$ be the set of *circuits* of A , which are those $\mathbf{c} \in \mathcal{G}(A)$ with support which is a circuit of the linear matroid of A . Let $c_\infty := \max_{\mathbf{c} \in \mathcal{C}(A)} \|\mathbf{c}\|_\infty$. We have $c_\infty \leq n^{\frac{n}{2}} a^n$ [21, Lemma 3.18].

We now use the proximity results of [12, 14] which assert that either (ILP) is infeasible or it has an optimal solution \mathbf{x}^* with $\|\mathbf{x}^* - \mathbf{y}^*\|_\infty \leq nc_\infty$ and hence $\|\mathbf{x}^* - \lfloor \mathbf{y}^* \rfloor\|_\infty \leq n^{\frac{n}{2}+1} a^n + 1$. Thus, making the variable transformation $\mathbf{x} = \mathbf{z} + \lfloor \mathbf{y}^* \rfloor$, problem (ILP) reduces to following,

$$\min \{ \mathbf{w}(\mathbf{z} + \lfloor \mathbf{y}^* \rfloor) \mid \mathbf{z} \in \mathbb{Z}^n, A(\mathbf{z} + \lfloor \mathbf{y}^* \rfloor) = \mathbf{b}, \mathbf{l} \leq \mathbf{z} + \lfloor \mathbf{y}^* \rfloor \leq \mathbf{u}, \|\mathbf{z}\|_\infty \leq n^{\frac{n}{2}+1} a^n + 1 \},$$

which is equivalent to the program

$$\min \{ \mathbf{w}\mathbf{z} \mid \mathbf{z} \in \mathbb{Z}^n, A\mathbf{z} = \bar{\mathbf{b}}, \bar{\mathbf{l}} \leq \mathbf{z} \leq \bar{\mathbf{u}} \} \quad (1)$$

where

$$\bar{\mathbf{b}} := \mathbf{b} - A\lfloor \mathbf{y}^* \rfloor, \quad \bar{l}_i := \max\{l_i - \lfloor y_i^* \rfloor, -(n^{\frac{n}{2}+1} a^n + 1)\}, \quad \bar{u}_i := \min\{u_i - \lfloor y_i^* \rfloor, n^{\frac{n}{2}+1} a^n + 1\} .$$

If some $\bar{l}_i > \bar{u}_i$ then (1) is infeasible and hence so is (ILP), so we may assume that

$$-(n^{\frac{n}{2}+1} a^n + 1) \leq \bar{l}_i \leq \bar{u}_i \leq n^{\frac{n}{2}+1} a^n + 1, \quad \text{for all } i .$$

This implies that if \mathbf{z} is any feasible point in (1) then $\|A\mathbf{z}\|_\infty \leq na(n^{\frac{n}{2}+1} a^n + 1)$ and so we may assume that $\|\bar{\mathbf{b}}\|_\infty \leq na(n^{\frac{n}{2}+1} a^n + 1)$ else there is no feasible solution. So we have

$$\|\bar{\mathbf{b}}\|_\infty, \|\bar{\mathbf{l}}\|_\infty, \|\bar{\mathbf{u}}\|_\infty \leq 2^{O(n \log n)} a^{O(n)} \text{ and hence } \langle \bar{\mathbf{b}}, \bar{\mathbf{l}}, \bar{\mathbf{u}} \rangle \text{ is polynomial in } \langle A \rangle .$$

Step 2: Solving the system of equations. We first search for an integer solution to the system of equations $A\mathbf{z} = \bar{\mathbf{b}}$. This can be done by computing the Hermite normal form of A , see [23], using a number of arithmetic operations polynomial in $\langle A \rangle$ and time polynomial in $\langle A, \bar{\mathbf{b}} \rangle$ which is polynomial in $\langle A \rangle$, and hence strongly polynomially in our original input. Then either we conclude that there is no integer solution to $A\mathbf{z} = \bar{\mathbf{b}}$ and hence (1) is infeasible, or we find a solution $\mathbf{z} \in \mathbb{Z}^n$ with $\langle \mathbf{z} \rangle$ polynomially bounded in $\langle A, \bar{\mathbf{b}} \rangle$ and hence also in $\langle A \rangle$.

Step 3: Finding a feasible point. Define relaxed bounds by

$$\hat{l}_i := \min\{\bar{l}_i, z_i\}, \quad \hat{u}_i := \max\{\bar{u}_i, z_i\}, \quad i \in [n].$$

Now for $i \in [n]$ iterate the following. If $\bar{l}_i \leq z_i \leq \bar{u}_i$ then simply increment i and repeat. If $z_i < \bar{l}_i$ (and hence $\hat{l}_i = z_i$ and $\hat{u}_i = \bar{u}_i$) then consider the following auxiliary integer program,

$$\max \left\{ x_i \mid \mathbf{x} \in \mathbb{Z}^n, A\mathbf{x} = \bar{\mathbf{b}}, \hat{\mathbf{l}} \leq \mathbf{x} \leq \hat{\mathbf{u}} \right\}. \quad (2)$$

Starting from the point \mathbf{z} feasible in (2), and using the Graver-best oracle for A , we can solve program (2) using Proposition 8 in polynomial time and in a number of arithmetic operations and oracle queries which is polynomial in n and $\log F$ (recall $F = z_i^* - z_i$ for some minimizer z_i^*), which is bounded by $\log(\hat{u}_i - \hat{l}_i) = \log(\bar{u}_i - z_i)$, thus polynomial in $\langle A \rangle$.

Let \mathbf{x} be an optimal solution of (2). If $x_i < \bar{l}_i$ then (1) is infeasible and we are done. Otherwise (in which case $\bar{l}_i \leq x_i \leq \bar{u}_i$) we update $\hat{l}_i := \bar{l}_i$ and $\mathbf{z} := \mathbf{x}$, increment i and repeat. The last case $z_i > \bar{u}_i$ is treated similarly where in (2) we minimize rather than maximize x_i .

Thus, strongly polynomially we either conclude at some iteration i that program (1) is infeasible or complete all iterations and obtain $\hat{\mathbf{l}} = \bar{\mathbf{l}}$, $\hat{\mathbf{u}} = \bar{\mathbf{u}}$, and a point \mathbf{z} feasible in (1).

Step 4: Reducing \mathbf{w} . Let $N := 2n(n^{\frac{n}{2}+1}a^n + 1) + 1$. Now apply the strongly polynomial algorithm of Frank and Tardos [8], which on arithmetic input $n, \langle N \rangle$ and numeric input $\langle \mathbf{w} \rangle$, outputs $\bar{\mathbf{w}} \in \mathbb{Z}^n$ with $\|\bar{\mathbf{w}}\|_\infty \leq 2^{O(n^3)} N^{O(n^2)}$ such that $\text{sign}(\mathbf{w}\mathbf{x}) = \text{sign}(\bar{\mathbf{w}}\mathbf{x})$ for all $\mathbf{x} \in \mathbb{Z}^n$ with $\|\mathbf{x}\|_1 < N$. Since $\langle N \rangle = O(\log N) = O(n \log n + n \log a)$ is polynomial in $\langle A \rangle$, this algorithm is also strongly polynomial in our original input. Now, for every two points \mathbf{x}, \mathbf{z} feasible in (1) we have $\|\mathbf{x} - \mathbf{z}\|_1 < 2n(n^{\frac{n}{2}+1}a^n + 1) + 1 = N$, so that for any two such points we have $\mathbf{w}\mathbf{x} \leq \mathbf{w}\mathbf{z}$ if and only if $\bar{\mathbf{w}}\mathbf{x} \leq \bar{\mathbf{w}}\mathbf{z}$, and therefore we can replace (1) by the equivalent program

$$\min \left\{ \bar{\mathbf{w}}\mathbf{z} : \mathbf{z} \in \mathbb{Z}^n, A\mathbf{z} = \bar{\mathbf{b}}, \bar{\mathbf{l}} \leq \mathbf{z} \leq \bar{\mathbf{u}} \right\}, \quad (3)$$

where

$$\|\bar{\mathbf{w}}\|_\infty = 2^{O(n^3 \log n)} a^{O(n^3)} \text{ and hence } \langle \bar{\mathbf{w}}, \bar{\mathbf{b}}, \bar{\mathbf{l}}, \bar{\mathbf{u}} \rangle \text{ is polynomial in } \langle A \rangle.$$

Step 5: Finding an optimal solution. Starting from the point \mathbf{z} which is feasible in (3), and using the Graver-best oracle for A , we can solve program (3) using again Proposition 8 in polynomial time and in a number of arithmetic operations and oracle queries which is polynomial in n and $\log F$, which is bounded by $\log(n\|\bar{\mathbf{w}}\|_\infty\|\bar{\mathbf{u}} - \bar{\mathbf{l}}\|_\infty)$, which is polynomial in $\langle A \rangle$, and hence strongly polynomially. ◀

► **Remark.** In fact, the reduced objective $\bar{\mathbf{w}}$ in step 4 need not be constructed: already its *existence* implies that (1) is solved in the same number of iterations as (3).

3 Multi-stage Stochastic and Tree-fold ILP

In this section we prove Theorems 2 and 3. We first formalize a common construction for a Graver-best oracle: one constructs a set of relevant step lengths Λ and then for each $\lambda \in \Lambda$ finds a λ -Graver-best step. A step with the best improvement among these is then guaranteed to be a Graver-best step. Thus, we reduce our task to constructing a Λ -Graver-best oracle.

Both algorithms for multi-stage stochastic ILP and tree-fold ILP follow the same pattern:

1. show that all elements of $\mathcal{G}(A)$ have bounded norms (ℓ_∞ and ℓ_1 , respectively),
2. show that A has bounded treewidth (primal and dual, respectively),
3. apply existing algorithms for (ILP) which are FPT parameterized by $\|A\|_\infty$, $\max \|\mathbf{x}\|_\infty$ and $\max \|\mathbf{x}\|_1$, and $\text{tw}_P(A)$ and $\text{tw}_D(A)$, respectively.

3.1 Preliminaries

3.1.1 Relevant Step Lengths

We say that $\mathbf{h} \in \{\mathbf{x} \in \mathbb{Z}^n \mid A\mathbf{x} = \mathbf{0}\}$ is a λ -Graver-best step if $\lambda\mathbf{h}$ is a feasible step and $\lambda\mathbf{w}\mathbf{h} \leq \lambda\mathbf{w}\mathbf{g}$ for any $\mathbf{g} \in \mathcal{G}(A)$ such that $\lambda\mathbf{g}$ is a feasible step. We denote by $g_1(A) = \max_{\mathbf{g} \in \mathcal{G}(A)} \|\mathbf{g}\|_1$ and $g_\infty(A) = \max_{\mathbf{g} \in \mathcal{G}(A)} \|\mathbf{g}\|_\infty$. The following lemma states that provided a bound on $g_\infty(A)$, in order to find a Graver-best step, it is sufficient to find a λ -Graver-best step for all $\lambda \in \Lambda$ for some not too large set Λ .

► **Definition 10** (Graver-best step-lengths). Let \mathbf{x} be a feasible solution to (ILP). We say that $\lambda \in \mathbb{N}$ is a *Graver-best step-length* for \mathbf{x} if there exists $\mathbf{g} \in \mathcal{G}(A)$ with $\mathbf{x} + \lambda\mathbf{g}$ feasible, such that $\forall \lambda' \in \mathbb{N}$ and $\forall \mathbf{g}' \in \mathcal{G}(A)$, $\mathbf{x} + \lambda'\mathbf{g}'$ is either infeasible or $\mathbf{w}(\mathbf{x} + \lambda\mathbf{g}) \leq \mathbf{w}(\mathbf{x} + \lambda'\mathbf{g}')$. We denote by $\Lambda(\mathbf{x}) \subseteq \mathbb{N}$ the set of Graver-best step-lengths for \mathbf{x} .

► **Lemma 11** (Polynomial $\Lambda \supseteq \Lambda(\mathbf{x})$). (*) Let \mathbf{x} be a feasible solution to (ILP), let $M \in \mathbb{N}$ be such that $g_\infty(A) \leq M$. Then it is possible to construct in time $O(Mn)$ a set $\Lambda \subseteq \mathbb{N}$ of size at most $2Mn$ such that $\Lambda(\mathbf{x}) \subseteq \Lambda$.

With this Λ at hand, in order to realize a Graver-best oracle, it suffices to realize an oracle which finds a λ -Graver-best step for a given λ :

► **Definition 12** (Λ -Graver-best oracle). A Λ -Graver-best oracle for an integer matrix A is one that, queried on $\mathbf{w}, \mathbf{b}, \mathbf{l}, \mathbf{u}, \mathbf{x}$ feasible to (ILP), and an integer $\lambda \in \mathbb{N}$, returns a λ -Graver-best step \mathbf{h} for \mathbf{x} .

► **Lemma 13** (Λ -Graver-best oracle \Rightarrow Graver-best oracle). (*) Let A be an integer matrix and let $M \in \mathbb{N}$ satisfy $g_\infty(A) \leq M$. Then a Graver-best oracle for A can be realized with $2Mn$ calls to a Λ -Graver-best oracle for A .

3.1.2 Multi-stage Stochastic and Tree-fold Matrices

Let the *height* of a rooted tree or forest be the maximum root-to-leaf distance in it (i.e., the number of edges along the root-to-leaf path). In the following we let T be a rooted tree of height $\tau - 1 \in \mathbb{N}$ whose all leaves are at *depth* $\tau - 1$, that is, the length of every root-leaf path is exactly $\tau - 1$. For a vertex $v \in T$, let T_v be the subtree of T rooted in v and let $\ell(v)$ denote the number of leaves of T contained in T_v . Let B_1, B_2, \dots, B_τ be a sequence of integer matrices with each B_s having $l \in \mathbb{N}$ rows and n_s columns, where $n_s \in \mathbb{N}$, $n_s \geq 1$. We shall define a *multi-stage stochastic* matrix $T^P(B_1, \dots, B_\tau)$ inductively; the superscript P

refers to the fact that $T^P(B_1, \dots, B_\tau)$ has bounded *primal* treedepth td_P , as we will later see.

For a leaf $v \in T$, $T_v^P(B_\tau) := B_\tau$. Let $d \in \mathbb{N}$, $0 \leq d \leq \tau - 2$, and assume that for all vertices $v \in T$ at depth $d + 1$, matrices $T_v^P(B_{\tau-d}, \dots, B_\tau)$ have been defined. For $s \in \mathbb{N}$, $1 \leq s \leq \tau$, we set $T_v^P(B_{[s:\tau]}) = T_v^P(B_s, \dots, B_\tau)$. Let $v \in T$ be a vertex at depth d with δ children v_1, \dots, v_δ . We set

$$T_v^P(B_{[\tau-d-1:\tau]}) := \begin{pmatrix} B_{\tau-d-1, \ell(v_1)} & T_{v_1}^P(B_{[\tau-d:\tau]}) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ B_{\tau-d-1, \ell(v_\delta)} & 0 & \cdots & T_{v_\delta}^P(B_{[\tau-d:\tau]}) \end{pmatrix}$$

where, for $N \in \mathbb{N}$, $B_{s,N} = \begin{pmatrix} B_s \\ \vdots \\ B_s \end{pmatrix}$ consists of N copies of the matrix B_s .

The structure of a multi-stage stochastic matrix makes it natural to partition any solution of a multi-stage stochastic ILP into *bricks*. Bricks are defined inductively: for $T_v^P(B_\tau)$ there is only one brick consisting of all coordinates; for $T_v^P(B_{[s:\tau]})$ the set of bricks is composed of all bricks for all descendants of v , plus the first n_s coordinates form an additional brick.

► **Example 14.** For $\tau = 3$ and T with root r of degree 2 and its children u and v of degree 2 and 3, we have $T_u^P(B_2, B_3) = \begin{pmatrix} B_2 & B_3 \\ B_2 & B_3 \end{pmatrix}$, $T_v^P(B_2, B_3) = \begin{pmatrix} B_2 & B_3 \\ B_2 & B_3 \\ B_2 & B_3 \end{pmatrix}$, and

$$T^P(B_1, B_2, B_2) = T_r^P(B_1, B_2, B_2) = \begin{pmatrix} B_1 & B_2 & B_3 \\ B_1 & B_2 & B_3 \\ B_1 & B_2 & B_3 \\ B_1 & B_2 & B_3 \end{pmatrix}, \text{ with a total of 8 bricks.}$$

Tree-fold matrices are essentially transposes of multi-stage stochastic ILP matrices. Let T be as before and A_1, \dots, A_τ be a sequence of integer matrices with each $A_s \in \mathbb{Z}^{r_s \times t}$, where $t \in \mathbb{N}$, $r_s \in \mathbb{N}$, $r_s \geq 1$. We shall define $T^D(A_1, \dots, A_\tau)$ inductively; the superscript D refers to the fact that $T^D(A_1, \dots, A_\tau)$ has bounded *dual* treedepth. The inductive definition is the same as before except that, for a vertex $v \in T$ at depth d with δ children v_1, \dots, v_δ , we set

$$T_v^D(A_{[\tau-d-1:\tau]}) := \begin{pmatrix} A_{\tau-d-1, \ell(v_1)} & A_{\tau-d-1, \ell(v_2)} & \cdots & A_{\tau-d-1, \ell(v_\delta)} \\ T_{v_1}^D(A_{[\tau-d:\tau]}) & 0 & \cdots & 0 \\ 0 & T_{v_2}^D(A_{[\tau-d:\tau]}) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & T_{v_\delta}^D(A_{[\tau-d:\tau]}) \end{pmatrix}$$

where, for $N \in \mathbb{N}$, $A_{s,N} = (A_s \cdots A_s)$ consists of N copies of the matrix A_s . A solution \mathbf{x} of a tree-fold ILP is partitioned into bricks $(\mathbf{x}^1, \dots, \mathbf{x}^n)$ where n is the number of leaves of T , and each \mathbf{x}^i is a t -dimensional vector.

3.1.3 Structural Parameters

We consider two graph parameters, namely *treewidth* $\text{tw}(G)$ and *treedepth* $\text{td}(G)$. We postpone the definition of treewidth to the full version as it is not central for us.

► **Definition 15 (Treedepth).** The *closure* $\text{cl}(F)$ of a rooted forest F is the graph obtained from F by making every vertex adjacent to all of its ancestors. The *treedepth* $\text{td}(G)$ of a graph G is one more than the minimum height of a forest F such that $G \subseteq \text{cl}(F)$.

It is known that $\text{tw}(G) \leq \text{td}(G)$. The treedepth $\text{td}(G)$ of a graph G with a witness forest F can be computed in time $f_{\text{td}}(\text{td}(G)) \cdot |V(G)|$ for some computable function f_{td} [22].

► **Definition 16** (Primal and dual graph). Given a matrix $A \in \mathbb{Z}^{m \times n}$, its *primal graph* $G_P(A) = (V, E)$ is defined as $V = [n]$ and $E = \{\{i, j\} \in \binom{[n]}{2} \mid \exists k \in [m] : A_{k,i}, A_{k,j} \neq 0\}$. In other words, its vertices are the columns of A and two vertices are connected if there is a row with non-zero entries at the corresponding columns. The *dual graph of A* is defined as $G_D(A) = G_P(A^\top)$, that is, the primal graph of the transpose of A .

► **Definition 17** (Matrix treewidth). Given a matrix A , its *primal treewidth* $\text{tw}_P(A)$ is defined as the treewidth of its primal graph, i.e., $\text{tw}(G_P(A))$, and its *dual treewidth* $\text{tw}_D(A)$ is $\text{tw}(G_D(A))$. Similarly, we define the *primal* and *dual treedepth* as $\text{td}_P(A) = \text{td}(G_P(A))$ and $\text{td}_D(A) = \text{td}(G_D(A))$, respectively.

Using a proof of Ganian et al. [11, Theorem 12] we show that we cannot hope to relax the parameter $\text{td}_D(A)$ to $\text{tw}_D(A)$, even if $\|A\|_\infty$ was a constant.

► **Lemma 18.** (*) (ILP) is NP-hard already when $\text{tw}_D(A) = 3$, $\|A\|_\infty = 2$, and $\mathbf{w} = \mathbf{0}$.

3.2 Multi-stage Stochastic ILP is strongly FPT

To prove Theorem 2, we need two ingredients: a bound on $g_\infty(A)$, and an algorithm for (ILP) with bounded $\text{tw}_P(A)$ and $\max \|\mathbf{x}\|_\infty$.

► **Lemma 19** (Multi-stage stochastic \Rightarrow bounded $g_\infty(A)$). (*) Let $A = T^P(B_1, \dots, B_\tau)$. Then $g_\infty(A) \leq f_{\text{mss-norm}}(a, n_1, \dots, n_\tau, l)$ for some computable function $f_{\text{mss-norm}}$.

► **Lemma 20.** (*) Let $X \in \mathbb{N}$. Problem (ILP) with the additional constraint $\|\mathbf{x}\|_\infty \leq X$ can be solved in time $(X + 1)^{O(\text{tw}_P(A))} \cdot (n + m)$.

Proof of Theorem 2. Let $A = T^P(B_1, \dots, B_\tau)$ be a multi-stage stochastic matrix. By Lemma 19, $g_\infty(A)$ is bounded by $M = f_{\text{mss-norm}}(a, n_1, \dots, n_\tau, l)$. We show how to construct a λ -Graver-best oracle. Given an integer $\lambda \in \mathbb{N}$, use Lemma 20 to solve

$$\min\{\lambda \mathbf{w} \mathbf{h} \mid \mathbf{A} \mathbf{h} = \mathbf{0}, \mathbf{1} \leq \mathbf{x} + \lambda \mathbf{h} \leq \mathbf{u}, \|\mathbf{h}\|_\infty \leq M\} .$$

This returns a λ -Graver-best step, because any optimal solution satisfies $\lambda \mathbf{h} \leq \lambda \mathbf{g}$ for all $\mathbf{g} \in \mathcal{G}(A)$. Using a simple induction and the inductive construction of A , one gets that A has $\text{tw}_P(A) \leq \text{td}_P(A) \leq n_1 + \dots + n_\tau + 1$ and thus the oracle is realized in FPT time. Lemma 13 then yields a Graver-best oracle, which, combined with Theorem 1, finishes the proof. ◀

3.3 Tree-fold ILP is strongly FPT

As before, to prove Theorem 3, we need two ingredients: a bound on $g_1(A)$, and an algorithm for (ILP) with bounded $\text{tw}_D(A)$ and $\max \|\mathbf{x}\|_1$.

► **Lemma 21** (Tree-fold \Rightarrow bounded $g_1(A)$). (*) Let $A_i \in \mathbb{Z}^{r_i \times t}$ for $i \in [\tau]$ with $a = \max\{2, \max_{i \in [\tau]} \|A_i\|_\infty\}$, $r = \sum_{i=1}^\tau r_i$. Let $A = T^D(A_1, \dots, A_\tau)$. There exists a computable function $f_{\text{tf-norm}}(a, r_1, \dots, r_\tau)$ such that $g_1(A) \leq f_{\text{tf-norm}}(a, r_1, \dots, r_\tau)$.

Proof sketch. Chen and Marx [4] prove a similar result under the assumption that t is also a parameter; thus, the remaining problem are essentially duplicitous columns. However, De Loera et al. [5] show that repeating columns of any matrix A' does not increase $g_1(A')$, and thus we can take A , delete duplicitous columns, apply the result of Chen and Marx, and our Lemma follows.

► **Lemma 22.** (*) Let $X \in \mathbb{N}$. Problem (ILP) with the additional constraint $\|\mathbf{x}\|_1 \leq X$ can be solved in time $(aX)^{O(\text{tw}_D(A))} \cdot n$, where $a = \max\{2, \|A\|_\infty\}$.

Proof sketch. Lemma 22 is proved by reformulating the nonlinear constraint $\|\mathbf{x}\|_1 \leq X$ by “splitting” each variable x_i into two non-negative variables $x_i = x_i^+ - x_i^-$, imposing the constraint $\sum_{i=1}^n (x_i^+ + x_i^-) \leq X$, and showing that this does not increase $\text{tw}_D(A)$ much; then, a recent dynamic programming algorithm of Ganian et al. [11, Theorem 6] does the job.

Proof of Theorem 3. Let $A = T^D(A_1, \dots, A_\tau)$ be a tree-fold matrix. By Lemma 21 we have that $g_1(A) \leq f_{\text{tf-norm}}(a, r_1, \dots, r_\tau) =: M$. We show how to construct a λ -Graver-best oracle. Given an integer $\lambda \in \mathbb{N}$, solve $\min\{\lambda \mathbf{w} \mathbf{h} \mid \mathbf{A} \mathbf{h} = 0, \mathbf{l} \leq \mathbf{x} + \lambda \mathbf{h} \leq \mathbf{u}, \|\mathbf{h}\|_1 \leq M\}$ using Lemma 22; clearly the result is a λ -Graver-best step. Using a simple induction and the inductive construction of A , one gets that A has $\text{tw}_D(A) \leq \text{td}_D(A) \leq r_1 + \dots + r_\tau + 1$ and thus the oracle is realized in FPT time. Lemma 13 then yields a Graver-best oracle, which, combined with Theorem 1, finishes the proof. \blacktriangleleft

n -fold ILP. A special case of tree-fold ILP is n -fold ILP, obtained by taking T to be the star with n leaves and $A = T^D(A_1, A_2)$, where $A_1 \in \mathbb{Z}^{r \times t}$ and $A_2 \in \mathbb{Z}^{s \times t}$.

Proof of Theorem 4. Before we apply Lemma 22, we need to bound $g_1(A)$. It follows from the proof of [13, Lemma 6.1] that there is a number $g(A) = \max_{\mathbf{v} \in \mathcal{G}(A_1 \mathcal{G}(A_2))} \|\mathbf{v}\|_1$ such that $g_1(A) \leq g(A) \cdot g_1(A_2)$.

Let $d_2 \leq (2a + 1)^s$ be the number of distinct columns of A_2 . De Loera et al. [5] give a bound on $g_1(A)$ in terms of the number of distinct columns of a matrix A :

► **Lemma 23** ([5, Corollary 3.7.4]). *Let $A \in \mathbb{Z}^{m \times n}$ be a matrix of rank r , let d be the number of different columns in A , and let $a = \max\{2, \|A\|_\infty\}$. Then $g_1(A) \leq (d - r)(r + 1)(\sqrt{ma})^m$.*

Thus, $g_1(A_2) \leq (d_2 - s)(s + 1)(\sqrt{sa})^s \leq (as)^{O(s)}$. Let G_2 be a matrix whose columns are elements of $\mathcal{G}(A_2)$. We have that $\|A_1 G_2\|_\infty \leq a \cdot (as)^{O(s)} \leq (as)^{O(s)}$. Moreover, since $A_1 G_2$ has r rows, it has at most $d_1 = ((as)^{O(s)})^r = (as)^{O(rs)}$ distinct columns. Again, by Lemma 23 we have that $g(A) = g_1(A_1 G_2) \leq (d_1 - r)(r + 1)(\sqrt{ras})^{O(s)r} \leq (ars)^{O(rs)}$. Combining, we get $g_1(A) \leq (ars)^{O(rs)} \cdot (as)^{O(s)} \leq (ars)^{O(rs)} =: M$.

We have $\text{tw}_D(A) \leq r + s + 1$ and thus running the algorithm of Lemma 22 once takes time $((ars)^{O(rs)})^{r+s} nt \leq (ars)^{O(r^2s+rs^2)} nt$ and finds the λ -Graver-best step. Lemma 13 then yields a Graver-best oracle. The reduced objective function $\bar{\mathbf{w}}$ in Step 4 of the proof of Theorem 1 satisfies $\|\bar{\mathbf{w}}\|_\infty \leq (ant)^{O((nt)^3)}$ and thus the number of calls to the Graver-best oracle is bounded by $(nt)^3 \log(nt)$, concluding the proof. \blacktriangleleft

4 Primal and Dual Treedepth

We prove Theorems 5 and 6 by showing that an ILP with bounded primal (dual) treedepth can be embedded into a multi-stage stochastic (tree-fold) ILP without increasing the parameters too much. The precise notion of how one ILP is embedded in another is captured as follows.

► **Definition 24** (Extended formulation). Let $n' \geq n$, $m' \in \mathbb{N}$, $A \in \mathbb{Z}^{m \times n}$, $\mathbf{b} \in \mathbb{Z}^m$, $\mathbf{l}, \mathbf{u} \in (\mathbb{Z} \cup \{\pm\infty\})^n$ and $A' \in \mathbb{Z}^{m' \times n'}$, $\mathbf{b}' \in \mathbb{Z}^{m'}$, $\mathbf{l}', \mathbf{u}' \in (\mathbb{Z} \cup \{\pm\infty\})^{n'}$. We say that $A'(\mathbf{x}, \mathbf{y}) = \mathbf{b}', \mathbf{l}' \leq (\mathbf{x}, \mathbf{y}) \leq \mathbf{u}'$ is an *extended formulation* of $A\mathbf{x} = \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$ if $\{\mathbf{x} \mid A\mathbf{x} = \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}\} = \{\mathbf{x} \mid \exists \mathbf{y} : A'(\mathbf{x}, \mathbf{y}) = \mathbf{b}', \mathbf{l}' \leq (\mathbf{x}, \mathbf{y}) \leq \mathbf{u}'\}$.

We note that from here on we always assume that if $\text{td}_P(A) = k$ or $\text{td}_D(A) = k$, then there is a *tree* (not a forest) F of height $k - 1$ such that $G_P(A) \subseteq \text{cl}(F)$ or $G_D(A) \subseteq \text{cl}(F)$, respectively. Otherwise $G_P(A)$ is not connected and each component corresponds to a subset of variables which defines an ILP that can be solved independently; similarly for $G_D(A)$.

4.1 Primal Treedepth

► **Lemma 25** (Bounded primal treedepth \Rightarrow multi-stage stochastic). *Let $A, \mathbf{b}, \mathbf{l}$ and \mathbf{u} as in (ILP) be given, let $a = \max\{2, \|A\|_\infty\}$ and $\tau + 1 = \text{td}_P(A)$. Then there exists $C \in \mathbb{Z}^{m' \times n'}$, $\mathbf{b}' \in \mathbb{Z}^{m'}$ and $\mathbf{l}', \mathbf{u}' \in (\mathbb{Z} \cup \{\pm\infty\})^{n'}$, $n' \leq n\tau$, $m' \leq (2a+1)^{\tau^2}$, which define an integer program $C(\mathbf{x}, \mathbf{y}) = \mathbf{b}', \mathbf{l}' \leq (\mathbf{x}, \mathbf{y}) \leq \mathbf{u}'$, which is an extended formulation of $A\mathbf{x} = \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$. Moreover, there exist matrices $B_1, \dots, B_{\tau-1} \in \mathbb{Z}^{(2a+1)^{\tau^2} \times \tau}$ and $B_\tau \in \mathbb{Z}^{(2a+1)^{\tau^2} \times (\tau + (2a+1)^{\tau^2})}$ and a tree T such that $C = T^P(B_1, \dots, B_\tau)$ is a multi-stage stochastic constraint matrix, and all can be computed in time $f_{P\text{-embed}}(a, \text{td}_P(A)) \cdot n^2$ for some computable function $f_{P\text{-embed}}$.*

Proof. Let F be a rooted tree of height τ such that $G_P(A) \subseteq \text{cl}(F)$ (recall that it can be computed in time $f_{\text{td}}(\text{td}_P(A)) \cdot |V(G_P(A))|$).

Step 1: Dummy columns. We make F structured by adding dummy columns. Observe that every root-leaf path is of length at most τ and thus contains at most τ branching vertices. Unless F is a path, obtain a matrix A' from A by inserting zero columns into A in order to make the path between any two branching vertices of length τ ; a special case is the root which we force to be in distance $\tau - 1$ from the closest branching vertex. Set lower and upper bounds on the corresponding new variables to 0. A zero column is an isolated vertex in the primal graph and thus can be inserted to an arbitrary path of the tree F . Moreover, if any leaf is at depth less than $\tau^2 - 1$, insert zero columns in the same way to make it be at depth exactly $\tau^2 - 1$. Now there exists a rooted tree F' of height $\tau^2 - 1$ such that $G_P(A') \subseteq \text{cl}(F')$, all branching vertices are in distances $0, \tau - 1, 2\tau - 1, \dots, (\tau - 1)\tau - 1$ from the root, and all leaves are at depth exactly $\tau^2 - 1$. We call all vertices at depth 0 or $i\tau - 1$, for $i \in [\tau]$, *frets*, including the root and leaves.

Step 2: Multi-stage stochastic extended formulation. Consider a root-leaf path P in F' : its vertex set $V(P)$ corresponds to a certain subset of the columns of A' , with $|V(P)| \leq \tau^2$. Furthermore, any row \mathbf{a} of A' with $\text{supp}(\mathbf{a}) \subseteq V(P)$ can be written as a vector $(\mathbf{a}^1, \dots, \mathbf{a}^\tau) \in \mathbb{Z}^{\tau^2}$ with $\mathbf{a}^i \in \mathbb{Z}^\tau$ for each $i \in [\tau]$, and with $\|\mathbf{a}\|_\infty \leq a$. The *bricks* \mathbf{a}^i correspond to segments between frets (including the end fret, i.e., the fret farthest from the root; segments adjacent to the root also contain the root). Also, for any row \mathbf{a} of A' , there exists some root-leaf path P such that $\text{supp}(\mathbf{a}) \subseteq V(P)$.

This inspires the following construction: let $B \in \mathbb{Z}^{(2a+1)^{\tau^2} \times \tau^2}$ be the matrix whose columns are all the possible vectors $\mathbf{a} \in \mathbb{Z}^{\tau^2}$ with $\|\mathbf{a}\|_\infty \leq \|A\|_\infty$. Let $B_i \in \mathbb{Z}^{(2a+1)^{\tau^2} \times \tau}$, for $i \in [\tau]$, be the submatrix of B formed by rows $(i-1)\tau + 1, \dots, i\tau$ and modify the last such submatrix B_τ by putting $B_\tau := (B_\tau \mid I)$ where $I \in \mathbb{Z}^{(2a+1)^{\tau^2} \times (2a+1)^{\tau^2}}$ is the identity matrix; the variables corresponding to columns of I will play the role of slack variables. Let T be the tree of height τ obtained from F' by contracting all paths between frets.

Now, let $C = T^P(B_1, \dots, B_\tau)$. Obtain \tilde{F} from F' by appending a leaf to every leaf, and observe that $G_P(T^P(B_1, \dots, B_\tau)) \subseteq \text{cl}(\tilde{F})$; the new leaves correspond to the slack variables in B_τ . Our goal now is to construct a right hand side vector \mathbf{b}' and lower and upper bounds \mathbf{l}', \mathbf{u}' to enforce exactly the constraints present in $A\mathbf{x} = \mathbf{b}$. For every root-leaf path P in F' there is a corresponding root-leaf path \tilde{P} in \tilde{F} such that \tilde{P} is P with an additional leaf. Fix a root-leaf path P in F' . For every row \mathbf{a} of A' with $\text{supp}(\mathbf{a}) \subseteq V(P)$ and right hand side β , there exists a unique row \mathbf{c} of C with $\text{supp}(\mathbf{c}) \subseteq V(\tilde{P})$ such that $\mathbf{c} = (\mathbf{a}, 1)$, and we set the right hand side of row \mathbf{c} to β .

For every row \mathbf{c} of C which was not considered in the previous paragraph, set the right hand side to 0 and for the slack variable of this row set the lower bound to $-\infty$ and the

upper bound to ∞ . Let us remark in passing that we are not limited by using the standard equality form of ILP: transforming an instance $A\mathbf{x} \leq \mathbf{b}$ into the standard equality form by adding slack variables only possibly increases the treedepth by 1. ◀

4.2 Dual Treedepth

► **Lemma 26** (Bounded dual treedepth \Rightarrow tree-fold). (*) Let $A, \mathbf{b}, \mathbf{l}$ and \mathbf{u} be as in (ILP), $a = \max\{2, \|A\|_\infty\}$ and $\tau + 1 = \text{td}_D(A)$. Then there exists $D \in \mathbb{Z}^{m' \times n'}$, $\mathbf{b}' \in \mathbb{Z}^{m'}$ and $\mathbf{l}', \mathbf{u}' \in (\mathbb{Z} \cup \{\pm\infty\})^{n'}$, $n' \leq nt$, $t \leq n$, $m' \leq m \cdot \tau$, which define an extended formulation of $A\mathbf{x} = \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$. Moreover, there exist matrices $A_1, \dots, A_\tau \in \mathbb{Z}^{\tau \times t}$ and a tree T such that $D = T^D(A_1, \dots, A_\tau)$ is a tree-fold constraint matrix, and all can be computed in time $f_{D\text{-embed}}(a, \text{td}_D(A)) \cdot n^2$ for some computable function $f_{D\text{-embed}}$.

References

- 1 Gautam Appa, Balázs Kotnyek, Konstantinos Papalamprou, and Leonidas Pitsoulis. Optimization with binet matrices. *Operations research letters*, 35(3):345–352, 2007.
- 2 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, STOC 2017*, pages 1206–1219. ACM, 2017.
- 3 Matthias Aschenbrenner and Raymond Hemmecke. Finiteness theorems in stochastic integer programming. *Foundations of Computational Mathematics*, 7(2):183–227, 2007.
- 4 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, SODA 2018*, pages 2801–2820. SIAM, 2018.
- 5 Jesús A. De Loera, Raymond Hemmecke, and Matthias Köppe. *Algebraic and Geometric Ideas in the Theory of Discrete Optimization*, volume 14 of *MOS-SIAM Series on Optimization*. SIAM, 2013.
- 6 Jesús A. De Loera, Raymond Hemmecke, and Jon Lee. On augmentation algorithms for linear and integer-linear programming: From Edmonds–Karp to Bland and beyond. *SIAM Journal on Optimization*, 25(4):2494–2511, 2015.
- 7 Pavel Dvořák, Eduard Eiben, Robert Galian, Dušan Knop, and Sebastian Ordyniak. Solving integer linear programs with a small number of global variables and constraints. *arXiv preprint arXiv:1706.06084*, 2017.
- 8 András Frank and Éva Tardos. An application of simultaneous diophantine approximation in combinatorial optimization. *Combinatorica*, 7(1):49–65, 1987.
- 9 Eugene C. Freuder. Complexity of K -tree structured constraint satisfaction problems. In *Proceedings of the 8th National Conference on Artificial Intelligence*, pages 4–9, 1990.
- 10 Robert Galian and Sebastian Ordyniak. The complexity landscape of decompositional parameters for ILP. *Artificial Intelligence*, 2018.
- 11 Robert Galian, Sebastian Ordyniak, and M. S. Ramanujan. Going beyond primal treewidth for (M)ILP. In Satinder P. Singh and Shaul Markovitch, editors, *AAAI*, pages 815–821. AAAI Press, 2017.
- 12 Raymond Hemmecke, Matthias Köppe, and Robert Weismantel. Graver basis and proximity techniques for block-structured separable convex integer minimization problems. *Mathematical Programming*, 145(1-2, Ser. A):1–18, 2014.
- 13 Raymond Hemmecke, Shmuel Onn, and Lyubov Romanchuk. N -fold integer programming in cubic time. *Mathematical Programming*, pages 1–17, 2013.
- 14 Dorit S. Hochbaum and J. George Shanthikumar. Convex separable optimization is not much harder than linear optimization. *Journal of the ACM*, 37(4):843–862, 1990.

- 15 Bart M. P. Jansen and Stefan Kratsch. A structural approach to kernels for ILPs: Treewidth and total unimodularity. In Nikhil Bansal and Irene Finocchi, editors, *ESA*, volume 9294 of *Lecture Notes in Computer Science*, pages 779–791. Springer, 2015.
- 16 Klaus Jansen, Kim-Manuel Klein, Marten Maack, and Malin Rau. Empowering the configuration-IP – new PTAS results for scheduling with setups times. *arXiv preprint arXiv:1801.06460*, 2018.
- 17 Ravi Kannan. Minkowski’s convex body theorem and integer programming. *Mathematics of Operations Research*, 12(3):415–440, 1987.
- 18 Dušan Knop, Martin Koutecký, and Matthias Mnich. Combinatorial n-fold integer programming and applications. In *ESA*, volume 87 of *LIPICs*, pages 54:1–54:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. Extended version available at <https://arxiv.org/abs/1705.08657>.
- 19 Dušan Knop, Martin Koutecký, and Matthias Mnich. Voting and bribing in single-exponential time. In *STACS*, volume 66 of *LIPICs*, pages 46:1–46:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.
- 20 Hendrik W. Lenstra, Jr. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983.
- 21 Shmuel Onn. Nonlinear discrete optimization. *Zurich Lectures in Advanced Mathematics, European Mathematical Society*, 2010. <http://ie.technion.ac.il/~onn/Book/ND0.pdf>.
- 22 Felix Reidl, Peter Rossmanith, Fernando Sánchez Villaamil, and Somnath Sikdar. A faster parameterized algorithm for treedepth. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *ICALP (1)*, volume 8572 of *Lecture Notes in Computer Science*, pages 931–942. Springer, 2014.
- 23 Alexander Shrijver. *Theory of linear and integer programming*. Interscience Series in Discrete Mathematics and Optimization. Wiley, 1986.
- 24 Éva Tardos. A strongly polynomial algorithm to solve combinatorial linear programs. *Operations Research*, 34(2):250–256, 1986.
- 25 Sergey I. Veselov and Aleksandr J. Chirkov. Integer program with bimodular matrix. *Discrete Optimization*, 6(2):220–222, 2009.