Efficient Sequential and Parallel Algorithms for Multistage Stochastic Integer Programming Using Proximity

Jana Cslovjecsek
EPFL, Lausanne, Switzerland

Friedrich Eisenbrand
EPFL, Lausanne, Switzerland

Michał Pilipczuk
University of Warsaw, Poland

Moritz Venzin
EPFL, Lausanne, Switzerland

Robert Weismantel
ETH, Zürich, Switzerland

Abstract

We consider the problem of solving integer programs of the form $\min\{c^\intercal x \ : \ Ax = b, x \in \mathbb{Z}_{\geqslant 0}\}$, where A is a multistage stochastic matrix in the following sense: the primal treedepth of A is bounded by a parameter d, which means that the columns of A can be organized into a rooted forest of depth at most d so that columns not bound by the ancestor/descendant relation do not have non-zero entries in the same row. We give an algorithm that solves this problem in fixed-parameter time $f(d, \|A\|_\infty) \cdot n \log^{\mathcal{O}(2^d)} n$, where f is a computable function and n is the number of rows of A. The algorithm works in the strong model, where the running time only measures unit arithmetic operations on the input numbers and does not depend on their bitlength. This is the first fpt algorithm for multistage stochastic integer programming to achieve almost linear running time in the strong sense. For two-stage stochastic integer programs, our algorithm works in time $2^{((r+s)\|A\|_\infty)^{\mathcal{O}(r(r+s))}} \cdot n \log^{\mathcal{O}(rs)} n$, which improves over previous methods both in terms of the polynomial factor and in terms of the dependence on r and s. In fact, for $r = 1$ the dependence on s is asymptotically almost tight assuming the Exponential Time Hypothesis. Our algorithm can be also parallelized: we give an implementation in the PRAM model that achieves running time $f(d, \|A\|_\infty) \cdot \log^{\mathcal{O}(2^d)} n$ using n processors.

The main conceptual ingredient in our algorithms is a new proximity result for multistage stochastic integer programs. We prove that if we consider an integer program P, say with a constraint matrix A, then for every optimum solution to the linear relaxation of P there exists an optimum (integral) solution to P that lies, in the ℓ_∞-norm, within distance bounded by a function of $\|A\|_\infty$ and the primal treedepth of A. On the way to achieve this result, we prove a generalization and considerable improvement of a structural result of Klein for multistage stochastic integer programs. Once the proximity results are established, this allows us to apply a treedepth-based branching strategy guided by an optimum solution to the linear relaxation.

2012 ACM Subject Classification Theory of computation → Fixed parameter tractability

Keywords and phrases parameterized algorithm, multistage stochastic programming, proximity

Digital Object Identifier 10.4230/LIPIcs.ESA.2021.33

Related Version *Full Version*: `https://arxiv.org/abs/2012.11742`

Funding *Friedrich Eisenbrand*: received funding from the Swiss National Science Foundation within the project Lattice Algorithms and Integer Programming (Nr. 200021-185030).
Michal Pilipczuk: received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 677651).
Robert Weismantel: received support from the Einstein Foundation Berlin.

29th Annual European Symposium on Algorithms (ESA 2021).
Editors: Petra Mutzel, Rasmus Pagh, and Grzegorz Herman; Article No. 33; pp. 33:1–33:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

We consider *integer linear programming* problems

$$\min\{\, c^{\intercal} x \;:\; Ax = b, x \in \mathbb{Z}_{\geqslant 0} \,\}, \tag{$\spadesuit$}$$

that are described by the *constraint matrix* $A \in \mathbb{Z}^{n\times m}$, the *linear objective goal* $c \in \mathbb{Z}^m$ and the *right-hand side* vector $b \in \mathbb{Z}^n$. As in this work we only consider linear objective functions, for brevity we use term *integer programming* instead of *integer linear programming.*

While *integer programming* is NP-hard, there are various natural assumptions on the constraint matrix A for which $(\spadesuit)$ is solvable in polynomial time. Famous examples include *totally unimodular* and *bimodular integer programming* [15, 1], integer programs with a *constant number of variables* [20, 17] or *bipartite matching* and *shortest path* problems, see, e.g. [25]. Another example are *block-structured* integer programs in which the constraint matrix exhibits a (recursive) block structure. For instance, in the case of N*-fold* integer programming [7], the removal of a small number of constraints (rows of A) results in decomposing the instance into a large number of small independent subproblems.

We focus on the case where the removal of a few *columns* of A results in a large number of independent subproblems, i.e., on the case where A is *two-stage* or *multistage stochastic.*

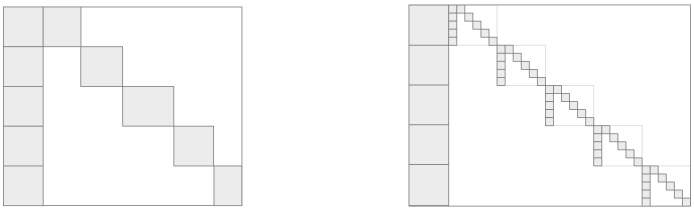

Figure 1 A schematic view of a two-stage stochastic matrix (left panel) and a multistage stochastic matrix (right panel). All non-zero entries are contained in the blocks depicted in grey.

Formally, A is two-stage or (r, s)*-stochastic* (Figure 1, left panel) if after deleting the first r columns the matrix can be decomposed into blocks with at most s columns each. The terminology is borrowed from the field of stochastic integer optimization, a model for discrete optimization under uncertainty. Here, the r "global" variables correspond to a decision made in the first stage, whereas the $\Omega(n)$ blocks involving s variables represent a usually *large* number different *scenarios* that arise in the second stage of stochastic optimization. Two-stage stochastic integer programming has found multiple applications and is a classical topic in optimization, see the survey of Schultz et al. [26] for some examples and algorithms. *Multistage stochastic* integer programming is a generalization of the two-stage variant above obtained by allowing further recursive levels in the block structure (Figure 1, right panel).

The recursive structure in multistage stochastic integer programs can be explained through the notion of the *primal treedepth* of a matrix. The primal treedepth of A, denoted $\mathsf{td}_{\mathsf{P}}(A)$, is the least integer d such that the columns of A can be organized into a rooted forest of depth at most d (called an *elimination forest*) with the following property: for every pair of columns that are not independent – they have non-zero entries in the same row – these columns have to be in the ancestor/descendant relation in the forest. The form presented in Figure 1 can be obtained by ordering the columns as in the top-down depth-first traversal of the elimination forest, and applying a permutation to the rows to form the blocks.

The primal treedepth is a structural parameter that is useful in the design of efficient integer programming solvers. By this, we mean the existence of *fixed-parameter algorithms* for the parameterization by $\mathsf{td}_{\mathsf{P}}(A)$ and $\|A\|_\infty$. For this parameterization, fixed-parameterized tractability can be understood in two ways. *Weak fpt algorithms* have running time of the form $f(\mathsf{td}_{\mathsf{P}}(A), \|A\|_\infty) \cdot |P|^{\mathcal{O}(1)}$, where f is a computable function and $|P|$ is the total bitlength of the encoding of the input. From *strong fpt algorithms* we require time complexity of the form $f(\mathsf{td}_{\mathsf{P}}(A), \|A\|_\infty) \cdot n^{\mathcal{O}(1)}$, where f is computable and n is the number of rows of the input matrix. Such algorithms work in the model where input numbers occupy single memory cells on which unit-cost arithmetic operations are allowed. Note that thus, the running time is not allowed to depend on the bitlength of the input numbers.

A weak fixed-parameter algorithm for the considered parameterization follows implicitly from the work of Aschenbrenner and Hemmecke [2]. The first to explicitly observe the applicability of primal treedepth to the design of fpt algorithms for integer programming were Ganian and Ordyniak [13], although their algorithm also treats $\|b\|_\infty$ as a parameter besides $\mathsf{td}_{\mathsf{P}}(A)$ and $\|A\|_\infty$. A major development was brought by Koutecký et al. [19], who gave the first strong fpt algorithm, with running time $f(\mathsf{td}_{\mathsf{P}}(A), \|A\|_\infty) \cdot n^3 \log^2 n$. We refer the reader to the joint manuscript of Eisenbrand et al. [10], which comprehensively presents the recent developments in the theory of block-structured integer programming. Corollaries 93 and 96 there discuss the cases of two-stage and multistage stochastic integer programming.

Our contribution. We advance the state-of-the-art of fpt algorithms for two-stage and multistage stochastic integer programming problems by proving the following. Here, n and d respectively denote the number of rows and the primal treedepth of the constraint matrix A.

A) We give an $f(d, \|A\|_\infty) \cdot n \log^{\mathcal{O}(2^d)} n$-time algorithm for integer programming $(\spadesuit)$ in the strong sense, where f is a computable function (Theorem 8). This improves upon the currently fastest strong fpt algorithm by Koutecký et al. [19] that is nearly cubic in n.

B) We provide a $2^{((r+s)\|A\|_\infty)^{\mathcal{O}(r(r+s))}} \cdot n \log^{\mathcal{O}(rs)} n$-time algorithm for (r, s)-stochastic integer programming, again in the strong sense. This improves upon the currently fastest algorithm that runs in time $2^{(2\|A\|_\infty)^{\mathcal{O}(r^2 s + rs^2)}} \cdot n^{\mathcal{O}(1)}$ [10, 18], both in terms of the parametric dependence and in terms of the polynomial factor in the running time.

The algorithmic contributions A and B rely on the following proximity result for integer programs with low primal treedepth. This result can be regarded as the core contribution of this paper, and we believe that it uncovers an important connection between the primal treedepth of A and the solution space of $(\spadesuit)$.

C) (Proximity) For each optimal solution $x^\star$ to the linear relaxation of $(\spadesuit)$ there is an optimal (integral) solution $x^\diamond$ such that $\|x^\diamond - x^\star\|_\infty$ is bounded by a computable function of $\mathsf{td}_{\mathsf{P}}(A)$ and $\|A\|_\infty$. (This is proved in Lemma 3.)

This proximity result provides a very simple template for designing fpt algorithms for multistage integer programming. Let us explain it for the case of (r, s)-stochastic IPs. After one has found an optimal fractional solution $x^\star$ of the linear relaxation of $(\spadesuit)$, one only has to enumerate the $(2 \cdot f(d, \|A\|_\infty) + 1)^r$ many possible integer assignments for the r stage 1 variables that are within the allowed distance, where $f(d, \|A\|_\infty)$ is the proximity bound provided by Item C. For each of these assignments, the integer program $(\spadesuit)$ decomposes into $\mathcal{O}(n)$ independent sub-problems, each with at most s variables. This results in a $f(r, s, \Delta) \cdot n$-time algorithm (excluding the time needed for solving the linear relaxation). For multistage-stochastic integer programming, this argument has to be applied recursively.

As for solving the linear relaxation, note that to obtain results A and B we need to be able to solve linear programs with low primal treedepth in near-linear fpt time in the strong sense. This is a non-trivial task. Here we rely on a recent paper, Cslovjecsek et al. [5] have shown that the dual of the linear programming relaxation of $(\spadesuit)$ can be solved in time $n \log^{\mathcal{O}(2^d)} n$.

By linear programming duality, this provides an algorithm for finding the *optimum value* of the linear relaxation of (♠) within the required complexity, but for applying the approach presented above, we need to actually compute an optimum fractional solution to (♠). While it is likely that the approach of Cslovjecsek et al. [5] can be modified so that it outputs such a solution as well, we give a self-contained argument using complementary slackness that applies the results of [5] only as a black-box.

The approach of Cslovjecsek et al. [5] is parallelizable, in the sense that the algorithm for solving the linear relaxation of (♠) can be implemented on a PRAM with n processors so that the running time is $\log^{\mathcal{O}(2^d)} n$, assuming the constraint matrix A is suitably organized on input. As the simple enumeration technique sketched above also can be easily applied in parallel, we obtain the following PRAM counterpart of A and B.

D) In both cases A and B, we provide algorithms that run in time $f(d, \|A\|_\infty) \cdot \log^{\mathcal{O}(2^d)} n$ and $2^{((r+s)\|A\|_\infty)^{\mathcal{O}(r(r+s))}} \cdot \log^{\mathcal{O}(rs)} n$, respectively, on a PRAM with n processors. For A we assume that the constraint matrix is suitably organized on input.

The proof of Item C relies on a structural lemma of Klein [18], which allows us to bound the ℓ_∞-norm of the projections of Graver-basis elements to the space of stage 1 variables. In the language of convex geometry, the lemma says the following: if the intersection of integer cones $C_1, \ldots, C_m \subseteq \mathbb{Z}^d$ is non-empty, where each generator of each C_i has ℓ_∞-norm at most Δ, then there is an integer vector $b \in \bigcap_{i=1}^m C_i$ that satisfies $\|b\|_\infty \leqslant 2^{\mathcal{O}(d\Delta)^d}$. In fact, the original bound of Klein [18] is doubly exponential in d^2, while we provide a new proof that improves this to a doubly exponential dependence on $d \log d$ only. A direct implication of this is the improvement in the parametric factor reported in B. We also consider some further relaxations of the statement that appear to be important in the proof of Item C.

Related work. The algorithm proposed by Koutecký et al. [19] for multistage stochastic programming relies on *iterative augmentation* using elements of the *Graver basis*, see also [7, 23, 14]. The Graver basis of a matrix A consists of all minimal integer solutions of $Ax = 0$. Here, minimal is w.r.t. the partial order $\sqsubseteq$ of $\mathbb{R}^n$ where $x \sqsubseteq y$ if for each $i \in \{1, \ldots, n\}$ we have $|x_i| \leqslant |y_i|$ and $x_i y_i \geqslant 0$. Intuitively, a Graver basis element comprises of "single steps" in the lattice of points x satisfying $Ax = 0$. The augmentation framework is to iteratively improve the current solution along directions in the Graver basis. It turns out that in the case of multistage stochastic programs, the ℓ_∞-norms of the elements of the Graver basis of the constraint matrix A can be bounded by $g(\mathsf{td}_\mathsf{P}(A), \|A\|_\infty)$ for some computable function g. This makes the iterative augmentation technique applicable in this setting. It seems that the augmentation framework is however inherently sequential.

Let us note that Koutecký et al. [19] relied on bounds on the function g above due to Aschenbrenner and Hemmecke [2], which only guaranteed computability. Better and explicit bounds on g were later given by Klein [18], see also [10]. Roughly speaking, the proof of Klein [18] shows that $g(d, a)$ is at most d-fold exponential, and it is open whether this bound can be improved to an elementary function.

On a related note, Jansen et al. [16] have very recently given a $2^{2^{o(s)}} \cdot n^{\mathcal{O}(1)}$ lower bound for $(1, s)$-stochastic IPs in which all coefficients of the constraint matrix are bounded by a constant in absolute values. This is assuming the Exponential Time Hypothesis. Thus, for $(1, s)$-stochastic integer programming with bounded coefficients, our result B is almost tight.

While robust and elegant, iterative augmentation requires further arguments to accelerate the convergence to an optimal solution in order to guarantee a good running time. As presented in [10], to overcome this issue one can either involve the bitlength of the input numbers in measuring the complexity, thus resorting to weak fpt algorithms, or reduce this bitlength using arguments originating in the work of Frank and Tardos [12]. For instance, integer program (♠) can be solved in time $f(d, \|A\|_\infty) \cdot n^{1+o(1)} \cdot \log^d \|c\|_\infty$, where $d = \mathsf{td}_\mathsf{P}(A)$. However, to the best of our knowledge, before this work there was no strong fpt algorithm that would achieve a subquadratic running time dependence on n, even in the setting of two-stage stochastic integer programming.

The setting of N-fold and tree-fold integer programming, which is dual to the setting considered in this work, has received a lot of attention in the literature, see e.g. [3, 5, 7, 10, 14, 24]. Here, we mostly rely on the recent results of Cslovjecsek et al. [5]. They obtained nearly linear-time strong fpt algorithms using an approach quite different from iterative augmentation, which served as a loose inspiration for our work. The key component is a proximity result for integer programs with bounded dual treedepth: they show that if P is an integer program with constraint matrix A, then for every optimal solution $x^\star$ to a suitable linear relaxation of P there exists an optimal (integral) solution $x^\diamond$ to P such that $\|x^\diamond - x^\star\|_1$ is bounded by a function of $\|A\|_\infty$ and the *dual treedepth* of A (i.e. primal treedepth of $A^\intercal$). It follows that if a solution $x^\star$ is available, then an optimal integral solution $x^\diamond$ can be found in linear fpt time using dynamic programming, where the bound on $\|x^\diamond - x^\star\|$ is used to limit the number of relevant states. This approach requires devising an auxiliary algorithm for solving linear relaxations with bounded dual treedepth in strong fpt time. This is achieved through recursive Laplace dualization using ideas from Norton et al. [22].

Let us stress that our proximity bound provided by Item C requires a different proof using completely different tools than the one obtained for tree-fold integer programs by Cslovjecsek et al. [5]. Note also that our proximity result concerns the standard linear relaxation, whereas the one in [5] holds for the strengthened relaxation, where the blocks are replaced by their integer hulls.

Very recently, Dong et al. [8] proposed a sophisticated interior-point algorithm to approximately solve linear programs whose constraint matrices have primal treewidth t in time $\tilde{\mathcal{O}}(nt^2 \cdot \log(1/\varepsilon))$, where ε is an accuracy parameter. Note here that the primal treewidth is bounded by the primal treedepth, so this algorithm in principle could be applied to the linear relaxation of (♠). There are two caveats: the algorithm of [8] provides only an approximate solution, and it is unclear whether it can be parallelized. For these reasons we rely on the algorithm of Cslovjecsek et al. [5] through dualization, but exploring the applicability of the work of Dong et al. [8] in our context is an exciting perspective for future work.

Organization. In this paper we focus on presenting the proximity result Item C and deriving algorithmic corollaries. Discussion of solving the linear relaxation as well as full proofs of statements marked with (✂), can be found in the full version of this paper, which is available on ArXiv [6].

2 Preliminaries

Model of computation. We assume a real RAM model of computation, where each memory cell stores a real number (of arbitrary bitlength and precision) and arithmetic operations (including rounding) are assumed to be of unit cost. For parallel computation we assume the CRCW PRAM model. As we will be working with sparse matrices, we assume that a matrix is specified on input by a list of its non-zero entries.

(Integer) linear programming. We consider integer programs of the form (♠). When replacing the integrality constraint $x \in \mathbb{Z}_{\geqslant 0}$ by $x \in \mathbb{R}_{\geqslant 0}$ yields the linear relaxation of (♠). We represent a program P as a quadruple $P = (x, A, b, c)$, where x, A, b, c are as in (♠). We denote by $\mathsf{Sol}^{\mathbb{Z}}(P)$ (resp. $\mathsf{Sol}^{\mathbb{R}}(P)$) the set of feasible integral solutions to P (resp. the set of fractional solutions to the linear relaxation to P). Analogously, we denote the set of optimal solutions (with respect to the objective function) by $\mathsf{opt}^{\mathbb{Z}}(P)$ and $\mathsf{opt}^{\mathbb{R}}(P)$ respectively.

Stochastic matrices. We say that a matrix M is *block-decomposable* if there are non-zero block matrices $M_1, \cdots, M_t$ such that M can be written as $M = \mathrm{diag}(M_1, \cdots, M_t)$. The matrices $M, M_1, \cdots, M_t$ need not be square. The *block decomposition* of M is then the unique presentation of M as $M = \mathrm{diag}(M_1, \cdots, M_t)$, where the blocks $M_1, \ldots, M_t$ are non empty and not block-decomposable.

For nonnegative integers r and s, a matrix A is *(r, s)-stochastic* if the following condition holds: if A' is A with the first r columns removed, then each block in the block decomposition of A' has at most s columns. Equivalently, an (r, s)-stochastic matrix can be written as

$$A = \begin{pmatrix} A_1 & B_1 & & & \\ A_2 & & B_2 & & \\ \vdots & & & \ddots & \\ A_t & & & & B_t \end{pmatrix}, \tag{$\diamondsuit$}$$

where the blocks $A_1, \ldots, A_t$ have r columns and each block B_i has at most s columns. As usual, in ($\diamondsuit$) and throughout the paper, empty spaces denote blocks filled with zeros. In general, a presentation of matrix A as in ($\diamondsuit$) is called the *stochastic decomposition* of A. To define the *primal treedepth* $\mathsf{td}_{\mathsf{P}}(A)$ of a matrix A, we first recursively define the *depth* of A:

- if A has no columns, then its depth is 0;
- if A is block-decomposable, then its depth is equal to the maximum among the depths of the blocks in its block decomposition; and
- if A has at least one column and is not block-decomposable, then the depth of A is one larger than the depth of the matrix obtained from A by removing its first column.

The primal treedepth of A is then the smallest integer d, such that the rows and columns of A can be permuted so that the resulting matrix has depth d.

For the remainder of this paper we will assume that matrices of bounded primal treedepth are suitably organized on input. That is, rows and columns are permuted so that the matrix has primal treedepth at most d and is in the block form depicted above. Finding such a permutation can be done in linear fpt time, we discuss this in the full version of the paper.

Graver bases. We collect some basic facts about Graver bases, for a thorough introduction to the theory and its applications we refer to [23, 21]. For an integer matrix A, we write $\ker^{\mathbb{Z}}(A)$ for the set of all integer vectors from $\ker(A)$. The *Graver basis* of A, denoted $\mathcal{G}(A)$, consists of all $\sqsubseteq$-minimal vectors of $\ker^{\mathbb{Z}}(A)$. We will use the following known bounds on $g_\infty(A) := \max_{v \in \mathcal{G}(A)} \|v\|_\infty$, the maximum norm of Graver basis elements:

▶ **Theorem 1** ([9], ✂). *For every integer matrix A with n rows and m columns, we have $g_\infty(A) \leqslant (2n\|A\|_\infty + 1)^n$ and $g_\infty(A) \leqslant (2m\|A\|_\infty + 1)^m$.*

We will also use the more general bounds for matrices with bounded primal treedepth.

▶ **Theorem 2** (Lemma 26 of [10]). *There is a computable function $f \colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ such that for every integer matrix A, $g_\infty(A) \leqslant f(\mathsf{td}_{\mathsf{P}}(A), \|A\|_\infty)$.*

We note that the proof of Theorem 2 given by Eisenbrand et al. [10] shows that, roughly speaking, $g_\infty(A)$ is bounded by a d-fold exponential function of $\|A\|_\infty$, where d is the primal treedepth of A.

3 Algorithms

As discussed, our algorithms use two ingredients: proximity results for stochastic integer programs, and algorithms for solving their linear relaxations. In this section we state those ingredients formally and argue how the results claimed in Section 1 follow.

As for proximity, we show that in stochastic integer programs, for every optimal fractional solution there is always an optimal integral solution that is not far, in terms of the ℓ_∞-norm. Precisely, the following results will be proved in Section 5.

▶ **Lemma 3.** *There exists a computable function $f\colon \mathbb{N}\times\mathbb{N}\to\mathbb{N}$ with the following property. Suppose $P=(x,A,b,c)$ is a linear program in the form (♠). Then for every optimal fractional solution $x^\star \in \mathsf{Sol}^{\mathbb{R}}(P)$ there exists an optimal integral solution $x^\diamond \in \mathsf{Sol}^{\mathbb{Z}}(P)$ satisfying*

$$\|x^\diamond - x^\star\|_\infty \leqslant f(\mathsf{depth}(A), \|A\|_\infty).$$

▶ **Lemma 4.** *Suppose $P=(x,A,b,c)$ is a linear program in the form (♠), where A is (r,s)-stochastic for some positive integers r,s. Then for every optimal fractional solution $x^\star \in \mathsf{Sol}^{\mathbb{R}}(P)$ there exists an optimal integral solution $x^\diamond \in \mathsf{Sol}^{\mathbb{Z}}(P)$ satisfying*

$$\|x^\diamond - x^\star\|_\infty \leqslant 2^{\mathcal{O}(r(r+s)\|A\|_\infty)^{r(r+s)}}.$$

A (r,s)-stochastic matrix has depth at most $r+s$, so Lemma 4 can be seen as a special case of Lemma 3, but provides an explicit bound. For solving the linear relaxation we obtain:

▶ **Lemma 5** (✂). *Suppose we are given a linear program $P=(x,A,b,c)$ in the form (♠), where A has n rows. Then, in the PRAM model, one can, using n processors and in time $\log^{\mathcal{O}(2^{\mathsf{depth}(A)})} n$, compute an optimal fractional solution to P.*

▶ **Lemma 6** (✂). *Suppose we are given an (r,s)-stochastic linear program $P=(x,A,b,c)$ in the form (♠), where A has n rows. Then, in the PRAM model, one can, using n processors and in time $2^{\mathcal{O}(r^2+rs^2)}\cdot\log^{\mathcal{O}(rs)} n$, compute an optimal fractional solution to P.*

Again, Lemma 6 differs from Lemma 5 by considering a more restricted class of matrices (i.e., (r,s)-stochastic), but providing better complexity bounds.

We now combine the tools presented above to show the following theorems.

▶ **Theorem 7** (✂). *Suppose we are given an (r,s)-stochastic linear program $P=(x,A,b,c)$ in the form (♠), where A has n rows. Then, in the PRAM model, one can, using n processors and in time $2^{((r+s)\|A\|_\infty)^{\mathcal{O}(r(r+s))}}\cdot\log^{\mathcal{O}(rs)} n$, compute an optimal integral solution to P.*

Sketch of proof. Apply Lemma 5 to find an optimal fractional solution $x^\star$. By Lemma 4, there is an optimal integral solution $x^\diamond$ satisfying $\|x^\diamond - x^\star\|_\infty \leqslant \rho$, where $\rho \in 2^{\mathcal{O}(r(r+s)\|A\|_\infty)^{r(r+s)}}$. In particular, if $x_0^\diamond$ and $x_0^\star$ are the projections of $x^\diamond$ and $x^\star$ onto the first r coordinates, respectively, then $\|x_0^\diamond - x_0^\star\|_\infty \leqslant \rho$.

Assume the stochastic decomposition (◇) of A. For all $\xi \in \mathbb{Z}^r_{\geqslant 0}$ satisfying $\|\xi - x^\star\|_\infty \leqslant \rho$ and all $i\in\{1,\ldots,t\}$, let us consider the integer program $P_i(\xi)$ defined as:

$$\min\{\, c_i^\mathsf{T} x_i \;:\; B_i x_i = b_i - A_i\xi,\; x_i \geqslant 0\,\},$$

where b_i, c_i, x_i are suitable restrictions of b,c,x to entries corresponding to rows or columns of B_i. It follows that

$$\mathsf{opt}^{\mathbb{Z}}(P) = \min\left\{ c_0^\mathsf{T}\xi + \sum_{i=1}^{t} \mathsf{opt}^{\mathbb{Z}}(P_i(\xi)) \;:\; \xi \geqslant 0 \text{ is integral and } \|\xi - x_1^\star\|_\infty \leqslant \rho \right\},$$

where c_0 is the projection of c onto the first r coordinates. Therefore, it suffices to iterate through all integral vectors $\xi \geqslant 0$ satisfying $\|\xi - x_1^\star\| \leqslant \rho$ one by one – of which there are at most $(2\rho+1)^r$ many – and for each of them solve all the programs $P_i(\xi)$ in parallel, by assigning to $P_i(\xi)$ as many processors as the number of rows of B_i.

It remains to argue how each of the programs $P_i(\xi)$ is going to be solved efficiently. For this, we may apply the same approach. Namely, we use Lemma 5 to find an optimal fractional solution $x_i^\star$ of $P_i(\xi)$, and using Lemma 4 again we can argue that there exists an optimal integral solution $x_i^\diamond$ of $P_i(\xi)$ that satisfies $\|x_i^\diamond - x_i^\star\|_\infty \leqslant \rho$. Now B_i has at most s columns, so there are only at most $(2\rho+1)^s$ candidates for an integral vector $x_i^\diamond$ satisfying the above, and they can be checked one by one. The overall running time analysis follows easily from the bounds provided by Lemma 4 and Lemma 5; we leave the details to the reader. ◀

The same basic idea, but applied recursively, yields the following.

▶ **Theorem 8** (✂)**.** *There is a computable function $f\colon \mathbb{N}\times\mathbb{N}\to\mathbb{N}$ such that the following holds. Suppose we are given a linear program $P=(x,A,b,c)$ in the form (♠), where A has n rows. Then, in the PRAM model, one can using n processors and in time $f(\mathsf{depth}(A),\|A\|_\infty)\cdot \log^{\mathcal{O}(2^{\mathsf{depth}(A)})} n$ compute an optimal integral solution to P.*

4 A stronger Klein bound

In this section we discuss a stronger variant of a structural result of Klein [18] which we will need for our proximity bounds in the next section.

▶ **Theorem 9** (Stronger Klein bound, ✂)**.** *Let $T_1,\ldots,T_n \subseteq \mathbb{Z}^d$ be multisets of integer vectors of ℓ_∞-norm at most Δ such that their respective sums are almost the same in the following sense: there is some $b\in\mathbb{Z}^d$ and a positive integer ϵ such that*

$$\Big\| \sum_{v\in T_i} v - b \Big\|_\infty < \epsilon \qquad \text{for all } i \in \{1,\ldots,n\}.$$

There exists a function $f(d,\Delta)\in 2^{\mathcal{O}(d\Delta)^d}$ such that the following holds. Assuming $\|b\|_\infty > \epsilon\cdot f(d,\Delta)$, one can find nonempty submultisets $S_i\subseteq T_i$ for all $i\in\{1,\ldots,n\}$, and a vector $b'\in\mathbb{Z}^d$ satisfying $\|b'\|_\infty \leqslant f(d,\Delta)$, such that

$$\sum_{v\in S_i} v = b' \qquad \text{for all } i \in \{1,\ldots,n\}.$$

Theorem 9 strengthens the original formulation of Klein [18, Lemma 2] in various aspects. First, the formulation of Klein required all the vectors to be nonnegative. Second, the argument of Klein yields a bound on $f(d,\Delta)$ that is doubly exponential in d^2, our proof improves this dependence to doubly exponential in $d\log d$. Finally, we allow the sums of the respective multisets to differ by some slack parameter ϵ, while in the original setting of Klein all sums need to be exactly equal. This last aspect will prove essential in the proof of our proximity bound, while the second is primarily used for improving the parametric factor in the running time.

The full proof of Theorem 9 relies on polyhedral techniques and is rather lengthy. However, using only the original formulation due to Klein and the pigeonhole principle, there is a short proof that achieves the third aspect of our improvement, i.e. that we may allow the sums of respective multisets to differ by some slack parameter. As this is central for the next section, we prove only this part and defer the full proof of Theorem 9 to the full version of the paper.

Proof of a simpler variant of Theorem 9. Specifically, we show the following statement:

> *Let $T_1, \ldots, T_n$ be multisets of vectors in $\mathbb{Z}^d_{\geqslant 0}$ of ℓ_∞ norm at most Δ such that there is some $b \in \mathbb{Z}^d$ and $\epsilon \in \mathbb{N}$ such that*
>
> $$\Big\| \sum_{v \in T_i} v - b \Big\|_\infty < \epsilon \qquad \text{for all } i \in \{1, \ldots, n\}.$$
>
> *Then there is a function $g(d, \Delta)$ such that provided $\|b\|_\infty > \epsilon \cdot g(d, \Delta)$, there exist nonempty submultisets $S_i \subseteq T_i$ for $i \in \{1, \ldots, n\}$ and a vector $b' \in \mathbb{Z}^d_{\geqslant 0}$ satisfying $\|b'\|_\infty \leqslant g(d, \Delta)$ such that*
>
> $$\sum_{v \in S_i} v = b' \qquad \text{for all } i \in \{1, \ldots, n\}.$$
>
> *In fact, one can take $g(d, \Delta) = 2(f(d, \Delta) + 1)^{d+1}$ where $f(d, \Delta)$ is the bound from [18].*

To prove this, we first add vectors belonging to $\{0, 1\}^d$ to each multiset T_i so that

$$\sum_{v \in \widetilde{T}_i} v = b + \epsilon \cdot \mathbf{1} \qquad \text{for all } i \in \{1, \ldots, n\},$$

where the $\widetilde{T}_i$ are the resulting multisets and $\mathbf{1}$ is the all-ones vector. Clearly, this can be achieved by adding to each multiset T_i at most 2ϵ vectors from $\{0, 1\}^d$.

Assume that $\|b\|_\infty > 2\epsilon \cdot (f(d, \Delta) + 1)^{d+1}$ where $f(d, \Delta)$ is the original bound from [18]. Since the multisets $\widetilde{T}_i$ all sum up to $b + \epsilon \cdot \mathbf{1}$ exactly, we can use the original formulation of [18] to infer that there are submultisets $S^1_i \subseteq \widetilde{T}_i$, for all $\{1, \ldots, n\}$, and a vector $b_1 \in \mathbb{Z}^d_{\geqslant 0}$ with $\|b_1\|_\infty \leqslant f(d, \Delta)$ such that

$$\sum_{v \in S^1_i} v = b_1 \qquad \text{for all } i \in \{1, \ldots, n\}.$$

Since the multisets $\widetilde{T}_i - S^1_i$ sum up to $b + \epsilon \cdot \mathbf{1} - b_1$, we can iteratively find nonempty submultisets $S^2_i \subseteq \widetilde{T}_i - S^1_i, \ldots, S^k_i \subseteq \widetilde{T}_i - (S^1_i \cup \ldots \cup S^k_i)$ and vectors $b_1, \ldots, b_k$ of ℓ_∞-norm bounded by $f(d, \Delta)$ such that

$$\sum_{v \in S^j_i} v = b_j \qquad \text{for all } j \in \{2, \ldots, k\} \text{ and } i \in \{1, \ldots, n\}.$$

Since we assumed that $\|b\|_\infty > 2\epsilon \cdot (f(d, \Delta) + 1)^{d+1}$, we can continue the above procedure until $k > 2\epsilon \cdot (f(d, \Delta) + 1)^d$. Note that there are at most $(f(d, \Delta) + 1)^d$ integral and nonnegative vectors of ℓ_∞ norm at most $f(d, \Delta)$. Therefore, by pigeonhole principle there exists $b' \in \mathbb{Z}^d_{\geqslant 0}$ with $\|b'\|_\infty \leqslant f(d, \Delta)$ and a set of indices J of size $2\epsilon + 1$ such that $\sum_{v \in S^j_i} v = b'$ for all $j \in J$ and $i \in \{1, \ldots, n\}$. For each $i \in \{1, \ldots, n\}$, one of these multisets $S^j_i \subseteq \widetilde{T}_i$ for $j \in J$ does not contain any of the (at most) 2ϵ vectors we have added to T_i to obtain $\widetilde{T}_i$. Thus, for each $i \in \{1, \ldots, n\}$ we can find a nonempty submultisets $S_i \subseteq T_i$ satisfying

$$\sum_{v \in S_i} v = b'.$$

Since $\|b'\|_\infty \leqslant f(d, \Delta) \leqslant g(d, \Delta)$, this concludes the proof. ◀

5 Proximity

The goal of this section to prove a very general theorem, Theorem 12, that will imply Lemma 3 and Lemma 4, see Corollaries 14 and 15. To facilitate the discussion of proximity, let us introduce the following definition.

▶ **Definition 10.** *Let $P = (x, A, b, c)$ be a linear program in the form $(\spadesuit)$. The* proximity *of P, denoted* $\mathsf{proximity}_\infty(P)$*, is the infimum of reals $\rho \geqslant 0$ satisfying the following: for every fractional solution $x^\star \in \mathsf{Sol}^{\mathbb{R}}(P)$ and integral solution $x^\square \in \mathsf{Sol}^{\mathbb{Z}}(P)$, there is an integral solution $x^\diamond \in \mathsf{Sol}^{\mathbb{Z}}(P)$ such that*

$$\|x^\diamond - x^\star\|_\infty \leqslant \rho \qquad \text{and} \qquad x^\diamond - x^\star \sqsubseteq x^\square - x^\star.$$

The condition $x^\diamond - x^\star \sqsubseteq x^\square - x^\star$ is equivalent to saying that $x^\diamond$ is contained in the axis parallel box spanned by $x^\star$ and $x^\square$, see Figure 2.

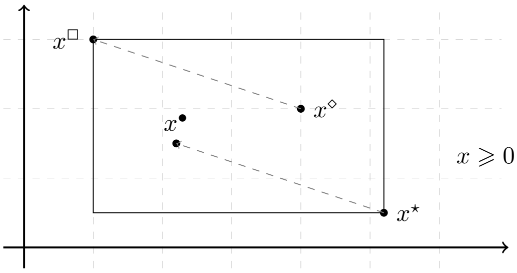

Figure 2 $x^\diamond - x^\star \sqsubseteq x^\square - x^\star$, $x^\diamond$ is in the rectangle spanned by $x^\square$ and $x^\star$.

Comparing to earlier work, for instance [4, 11], this notion of proximity is independent of the optimization goal. However, it can also be used to bound the distance of optimal fractional solutions to optimal integral solutions.

▶ **Lemma 11** (✂). *Suppose $P = (x, A, b, c)$ is a linear program in the form $(\spadesuit)$. Then for every optimal fractional solution $x^\star$ to P there is an optimal integral solution $x^\diamond$ to P satisfying*

$$\|x^\diamond - x^\star\|_\infty \leqslant \mathsf{proximity}_\infty(P).$$

Sketch of proof, see Figure 2. Let $x^\square$ any optimal integral solution. By our definition of proximity there is $x^\diamond$ with $\|x^\diamond - x^\star\|_\infty \leqslant \mathsf{proximity}_\infty(P)$ and $x^\diamond - x^\star \sqsubseteq x^\square - x^\star$. It can be easily checked that if $c^\intercal x^\square < c^\intercal x^\diamond$, then $x^\bullet := x^\star + x^\square - x^\diamond$ is feasible and $c^\intercal x^\bullet < c^\intercal x^\star$, contradicting the optimality of $x^\star$. Thus, $x^\diamond$ is an optimal integral solution. ◀

For the remainder of this section we adopt the following notation. Suppose that A has a stochastic decomposition $(\diamondsuit)$. Let $x_0, x_1, \ldots, x_t$ be the partition of the vector of variables x so that $x_0 \in \mathbb{R}^r$ corresponds to the columns of matrices $A_1, \ldots, A_t$, while $x_i \in \mathbb{R}^s$ corresponds to the columns of B_i, for each $i \in \{1, \ldots, t\}$. Finally, partition b into $b_1, \ldots, b_t$ so that b_i corresponds to the rows of A_i and B_i respectively. Thus, $\mathsf{Sol}^{\mathbb{R}}(P)$ takes the form:

$$\begin{aligned} A_i x_0 + B_i x_i &= b_i \qquad \text{for all } i \in \{1, \ldots, t\}, \\ x_i &\geqslant 0 \qquad \text{for all } i \in \{0, 1, \ldots, t\}. \end{aligned}$$

For each $i \in \{1, \ldots, t\}$, we define

$$P_i = \left(\begin{pmatrix} x_0 \\ x_i \end{pmatrix}, D_i, b_i, 0 \right) \quad \text{with} \quad D_i \coloneqq (A_i \; B_i). \tag{1}$$

We observe that

$$\begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_t \end{pmatrix} \in \mathsf{Sol}^{\mathbb{R}}(P) \qquad \text{if and only if} \qquad \begin{pmatrix} x_0 \\ x_i \end{pmatrix} \in \mathsf{Sol}^{\mathbb{R}}(P_i) \quad \text{for all } i \in \{1, \ldots, t\}.$$

This decomposition of the constraint matrix is the key to our main technical result.

▶ **Theorem 12** (Composition Theorem). *Suppose $P = (x, A, b, c)$ is a linear program in the form (♠), where A admits a stochastic decomposition (♢). Adopt the notation presented above and let k be the number of columns of each of the matrices $A_1, \ldots, A_t$. Further, let*

$$\gamma \coloneqq \max_{1 \leqslant i \leqslant t} g_\infty(D_i) \qquad \textit{and} \qquad \rho \coloneqq \max_{1 \leqslant i \leqslant t} \mathsf{proximity}_\infty(P_i).$$

Then

$$\mathsf{proximity}_\infty(P) \leqslant 3k\gamma\rho \cdot f(k, \gamma)$$

where $f(k, \gamma)$ is the bound provided by Theorem 9.

By substituting $f(k, \gamma)$ with the bound of Theorem 9, we get $\mathsf{proximity}_\infty(P) \leqslant \rho \cdot 2^{\mathcal{O}(k\gamma)^k}$. To derive the promised bounds on the proximity we use the following simple lemma.

▶ **Lemma 13** (✂). *Let $P = (x, A, b, c)$ be a linear program in the form (♠) where A has m columns. Then*

$$\mathsf{proximity}_\infty(P) \leqslant (m\|A\|_\infty)^{m+1}.$$

Sketch of proof. Given a feasible fractional solution $x^\star$ and an integral vector $x^\square$, we can consider the ILP on m variables defined by constraints $Ax = b$ and $x - x^\star \sqsubseteq x^\square - x^\star$. By the classic theorem of Cook et al. [4], there is an integral solution whose ℓ_∞ distance from $x^\star$ is at most m times the largest sub-determinant. It remains to apply the Hadamard bound. ◀

▶ **Corollary 14.** *Let $P = (x, A, b, c)$ be a linear program in the form (♠), where A is (r, s)-stochastic. Then*

$$\mathsf{proximity}_\infty(P) \leqslant 2^{\mathcal{O}(r(r+s)\|A\|_\infty)^{r(r+s)}}.$$

Proof. Recalling the definition of P_i in (1) and using that the matrix A is (r, s)-stochastic, we see that the constraint matrix D_i of P_i has at most $r + s$ columns with entries bounded by $\|A\|_\infty$. Using Lemma 13 and Theorem 1 respectively, we get

$$g_\infty(D_i) \leqslant (2(r + s)\|A\|_\infty + 1)^{r+s} \quad \text{and} \quad \mathsf{proximity}_\infty(P_i) \leqslant ((r + s)\|A\|_\infty)^{r+s+1}.$$

By Theorem 12 we obtain the claimed bound on $\mathsf{proximity}_\infty(A)$. ◀

Applying the same idea recursively yields the following.

▶ **Corollary 15** (✂). *There is a computable function $h\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ such that for every linear program $P = (x, A, b, c)$ in the form (♠), we have*

$$\mathsf{proximity}_\infty(P) \leqslant h(\mathsf{td}_{\mathsf{P}}(A), \|A\|_\infty).$$

5.1 Proof of Theorem 12

We now use our strengthening of the lemma of Klein [18], Theorem 9, to prove Theorem 12.

Proof of Theorem 12. Consider any $x^\star \in \mathsf{Sol}^{\mathbb{R}}(P)$ and $x^\square \in \mathsf{Sol}^{\mathbb{Z}}(P)$. Let $x^\diamond \in \mathsf{Sol}^{\mathbb{Z}}(P)$ be an integral solution such that $x^\diamond - x^\star \sqsubseteq x^\square - x^\star$ and subject to the condition that $\|x^\diamond - x^\star\|_1$ is minimized. Our goal is to show that then $\|x^\diamond - x^\star\|_\infty \leqslant 3k\gamma\rho \cdot f(k,\gamma)$, where $f(\cdot,\cdot)$ is the function given by Theorem 9.

Observe that if there existed a non-zero vector $u \in \ker^{\mathbb{Z}}(A)$ such that $u \sqsubseteq x^\star - x^\diamond$, then we would have that $x^\diamond + u \in \mathsf{Sol}^{\mathbb{Z}}(P)$, $(x^\diamond + u) - x^\star \sqsubseteq x^\diamond - x^\star \sqsubseteq x^\square - x^\star$, and the ℓ_1 distance from $x^\star$ to $x^\diamond + u$ would be strictly smaller than to $x^\diamond$. This would contradict the choice of $x^\diamond$. Therefore, it is sufficient to show the following: if $\|x^\diamond - x^\star\|_\infty$ is larger than $3k\gamma\rho \cdot f(k,\gamma)$, then there exists a non-zero vector $u \in \ker^{\mathbb{Z}}(A)$ such that $u \sqsubseteq x^\star - x^\diamond$.

To this end, for all $i \in \{1,\dots,t\}$, restrict $x^\star$ and $x^\diamond$ to the variables of P_i as follows:

$$\widetilde{x}_i^\star \coloneqq \begin{pmatrix} x_0^\star \\ x_i^\star \end{pmatrix} \in \mathsf{Sol}^{\mathbb{R}}(P_i) \qquad \text{and} \qquad \widetilde{x}_i^\diamond \coloneqq \begin{pmatrix} x_0^\diamond \\ x_i^\diamond \end{pmatrix} \in \mathsf{Sol}^{\mathbb{Z}}(P_i).$$

By the definition of proximity, for all $i \in \{1,\dots,t\}$ there are integral solutions

$$\widetilde{x}_i \in \mathsf{Sol}^{\mathbb{Z}}(P_i) \quad \text{with} \quad \|\widetilde{x}_i - \widetilde{x}_i^\star\|_\infty \leqslant \mathsf{proximity}_\infty(P_i) \leqslant \rho \quad \text{and} \quad \widetilde{x}_i - \widetilde{x}_i^\star \sqsubseteq \widetilde{x}_i^\diamond - \widetilde{x}_i^\star.$$

Since $\widetilde{x}_i$ and $\widetilde{x}_i^\diamond$ are both integral solutions to P_i, we have $\widetilde{x}_i - \widetilde{x}_i^\diamond \in \ker^{\mathbb{Z}}(A_i\ B_i)$ and we can decompose this vector into a multiset G_i of Graver elements. That is, G_i is a multiset consisting of sign compatible (i.e., belonging to the same orthant) elements of $\mathcal{G}(D_i)$ with $\widetilde{x}_i - \widetilde{x}_i^\diamond = \sum_{g \in G_i} g$. Note that the first k entries of vectors $\widetilde{x}_1,\dots,\widetilde{x}_t$ correspond to the same k variables of P, but they may differ for different $i \in \{1,\dots,t\}$. For a vector w, let $\pi(w)$ be the projection onto the first k entries of w.

Let $\pi(G_i)$ be the multiset that includes a copy of $\pi(g)$ for each $g \in G_i$. By the definition of $\widetilde{x}_i^\star$ and $\widetilde{x}_i^\diamond$, we have $\pi(\widetilde{x}_i^\star) = \pi(\widetilde{x}_j^\star)$ and $\pi(\widetilde{x}_i^\diamond) = \pi(\widetilde{x}_j^\diamond)$ for all $i,j \in \{1,\dots,t\}$. From this, for all $i \in \{1,\dots,t\}$,

$$\Big\| \sum_{x \in \pi(G_i)} x \ - \pi(\widetilde{x}_1^\star - \widetilde{x}_1^\diamond) \Big\|_\infty = \|\pi(\widetilde{x}_i) - \underbrace{\pi(\widetilde{x}_1^\star)}_{=\pi(\widetilde{x}_i^\star)} + \underbrace{\pi(\widetilde{x}_1^\diamond) - \pi(\widetilde{x}_i^\diamond)}_{=0}\|_\infty \leqslant \|\widetilde{x}_i - \widetilde{x}_i^\star\|_\infty \leqslant \rho.$$

Thus, Theorem 9 is applicable for $d = k$, $\Delta = \gamma$, $\epsilon = \rho$ and $b = \pi(\widetilde{x}_1^\star - \widetilde{x}_1^\diamond)$. Note that for each $i \in \{1,\dots,t\}$ and $g \in G_i$, we have $\|g\|_\infty \leqslant \gamma$. We now distinguish between two cases:
Case 1: $\|\pi(\widetilde{x}_1^\star - \widetilde{x}_1^\diamond)\|_\infty > \rho \cdot f(k,\gamma)$.
By Theorem 9, there exist nonempty submultisets $S_1 \subseteq \pi(G_1),\dots,S_t \subseteq \pi(G_t)$ such that

$$\sum_{x \in S_i} x = \sum_{x \in S_j} x \qquad \text{for all } i,j \in \{1,\dots,t\}.$$

Define a vector u in the following way. For all $i \in \{1,\dots,t\}$, let $\widehat{G}_i \subseteq G_i$ be submultisets with $\pi(\widehat{G}_i) = S_i$ and set $\widetilde{u}_i \coloneqq \sum_{g \in \widehat{G}_i} g \in \ker^{\mathbb{Z}}(D_i)$. Observe that vectors $\pi(\widetilde{u}_i)$ are equal for all $i \in \{1,\dots,t\}$. This allows us to define u as the vector obtained by combining all the $\widetilde{u}_i$, so that projecting u to the variables of P_i yields $\widetilde{u}_i$, for each $i \in \{1,\dots,t\}$. Note that since multisets $\widehat{G}_i$ are nonempty, u is a non-zero vector. Also $u \in \ker^{\mathbb{Z}}(A)$, since $\widetilde{u}_i \in \ker^{\mathbb{Z}}(D_i)$ for all $i \in \{1,\dots,t\}$. Further, we have $u \sqsubseteq x^\star - x^\diamond$, because for all $i \in \{1,\dots,t\}$,

$$\widetilde{u}_i = \sum_{g \in \widehat{G}_i} g \sqsubseteq \widetilde{x}_i - \widetilde{x}_i^\diamond \sqsubseteq \widetilde{x}_i^\star - \widetilde{x}_i^\diamond.$$

Thus, u contradicts the minimality of $\|x^\diamond - x^\star\|_1$. We move to the second case.
Case 2: $\|\pi(\widetilde{x}_1^\star - \widetilde{x}_1^\diamond)\|_\infty \leqslant \rho \cdot f(k, \gamma)$.
Since we have $\|\pi(\widetilde{x}_i - \widetilde{x}_i^\diamond) - \pi(\widetilde{x}_1^\star - \widetilde{x}_1^\diamond)\|_\infty \leqslant \rho$ for all $i \in \{1, \ldots, t\}$, we have

$$\|\pi(\widetilde{x}_i - \widetilde{x}_i^\diamond)\|_\infty \leqslant \rho \cdot f(k,\gamma) + \rho \leqslant 2\rho \cdot f(k,\gamma) \qquad \text{for all } i \in \{1,\ldots,t\}.$$

Suppose for a moment that for some $i \in \{1, \ldots, t\}$, there exists an element $g \in G_i$ with $\pi(g) = 0$. Then by putting zeros on all the other coordinates, we can extend g to a vector $u \in \ker^{\mathbb{Z}}(A)$ which satisfies $u \sqsubseteq x^\star - x^\diamond$. As g is non-zero, so is u, hence u satisfies all the requested properties. Hence, from now on we may assume that no multiset G_i contains an element g with $\pi(g) = 0$. It follows that

$$|G_i| = |\pi(G_i)| \leqslant \Big\| \sum_{x \in \pi(G_i)} x \Big\|_1 \leqslant k \Big\| \sum_{x \in \pi(G_i)} x \Big\|_\infty = k\|\pi(\widetilde{x}_i - \widetilde{x}_i^\diamond)\|_\infty \leqslant 2k\rho \cdot f(k,\gamma).$$

Since $\|g\|_\infty \leqslant \gamma$ for every element $g \in G_i$, we infer that

$$\|\widetilde{x}_i - \widetilde{x}_i^\diamond\|_\infty \leqslant \Big\| \sum_{g \in G_i} g \Big\|_\infty \leqslant \gamma |G_i| \leqslant 2k\gamma\rho \cdot f(k,\gamma).$$

By combining this with $\|\widetilde{x}_i - \widetilde{x}_i^\star\|_\infty \leqslant \rho$, we get

$$\|\widetilde{x}_i^\diamond - \widetilde{x}_i^\star\|_\infty \leqslant \|\widetilde{x}_i^\diamond - \widetilde{x}_i\|_\infty + \|\widetilde{x}_i - \widetilde{x}_i^\star\|_\infty \leqslant 2k\gamma\rho \cdot f(k,\gamma) + \rho \leqslant 3k\gamma\rho \cdot f(k,\gamma).$$

This implies that $\|x^\diamond - x^\star\| \leqslant 3k\gamma\rho \cdot f(k,\gamma)$ and concludes the proof. ◀

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*, STOC 2017, page 1206â€“1219, New York, NY, USA, 2017. Association for Computing Machinery. doi:10.1145/3055399.3055473.

2 Matthias Aschenbrenner and Raymond Hemmecke. Finiteness theorems in stochastic integer programming. *Found. Comput. Math.*, 7(2):183–227, 2007. doi:10.1007/s10208-005-0174-1.

3 Lin Chen and Daniel 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 ’18, page 2801â€“2820, USA, 2018. Society for Industrial and Applied Mathematics.

4 W. Cook, A. M. H. Gerards, A. Schrijver, and É. Tardos. Sensitivity theorems in integer linear programming. *Mathematical Programming*, 34(3):251–264, 1986. doi:10.1007/BF01582230.

5 Jana Cslovjecsek, Friedrich Eisenbrand, Christoph Hunkenschröder, Lars Rohwedder, and Robert Weismantel. Block-structured integer and linear programming in strongly polynomial and near linear time. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021*, pages 1666–1681. SIAM, 2021. doi:10.1137/1.9781611976465.101.

6 Jana Cslovjecsek, Friedrich Eisenbrand, MichaÅ, Pilipczuk, Moritz Venzin, and Robert Weismantel. Efficient sequential and parallel algorithms for multistage stochastic integer programming using proximity, 2020. arXiv:2012.11742.

7 JesÃºs A. De Loera, Raymond Hemmecke, Shmuel Onn, and Robert Weismantel. N-fold integer programming. *Discrete Optimization*, 5(2):231–241, 2008. In Memory of George B. Dantzig. doi:10.1016/j.disopt.2006.06.006.

8 Sally Dong, Yin Tat Lee, and Guanghao Ye. A nearly-linear time algorithm for linear programs with small treewidth: A multiscale representation of robust central path. *CoRR*, abs/2011.05365, 2020. arXiv:2011.05365.

9 Friedrich Eisenbrand, Christoph Hunkenschröder, and Kim-Manuel Klein. Faster algorithms for integer programs with block structure. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018*, volume 107 of *LIPIcs*, pages 49:1–49:13. Schloss Dagstuhl — Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.49.

10 Friedrich Eisenbrand, Christoph Hunkenschröder, Kim-Manuel Klein, Martin Koutecký, Asaf Levin, and Shmuel Onn. An algorithmic theory of integer programming. *CoRR*, abs/1904.01361, 2019. arXiv:1904.01361.

11 Friedrich Eisenbrand and Robert Weismantel. Proximity results and faster algorithms for integer programming using the steinitz lemma. *ACM Trans. Algorithms*, 16(1), 2019. doi:10.1145/3340322.

12 András Frank and Éva Tardos. An application of simultaneous diophantine approximation in combinatorial optimization. *Combinatorica*, 7(1):49–65, 1987. doi:10.1007/BF02579200.

13 Robert Ganian and Sebastian Ordyniak. The complexity landscape of decompositional parameters for ILP. In *30th Conference on Artificial Intelligence, AAAI 2016*, pages 710–716. AAAI Press, 2016. URL: http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12432.

14 Raymond Hemmecke, Shmuel Onn, and Lyubov Romanchuk. N-fold integer programming in cubic time. *Math. Program.*, 137(1-2):325–341, 2013. doi:10.1007/s10107-011-0490-y.

15 Alan J Hoffman and Joseph B Kruskal. Integral boundary points of convex polyhedra. In *Linear Inequalities and Related Systems.(AM-38), Volume 38*, pages 223–246. Princeton University Press, 2016.

16 Klaus Jansen, Kim-Manuel Klein, and Alexandra Lassota. The double exponential runtime is tight for 2-stage stochastic ILPs. *CoRR*, abs/2008.12928, 2020. arXiv:2008.12928.

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

18 Kim-Manuel Klein. About the complexity of two-stage stochastic IPs. In *Proceedings of the 21st International Conference, IPCO 2020*, volume 12125 of *Lecture Notes in Computer Science*, pages 252–265. Springer, 2020. doi:10.1007/978-3-030-45771-6_20.

19 Martin Koutecký, Asaf Levin, and Shmuel Onn. A parameterized strongly polynomial algorithm for block structured integer programs. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018*, volume 107 of *LIPIcs*, pages 85:1–85:14. Schloss Dagstuhl — Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.85.

20 H. W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983. URL: http://www.jstor.org/stable/3689168.

21 Jesús De Loera, Raymond Hemmecke, and Matthias Kppe. *Algebraic and Geometric Ideas in the Theory of Discrete Optimization.* Society for Industrial and Applied Mathematics, USA, 2012.

22 Carolyn Haibt Norton, Serge A. Plotkin, and Éva Tardos. Using separation algorithms in fixed dimension. *J. Algorithms*, 13(1):79–98, 1992. doi:10.1016/0196-6774(92)90006-X.

23 S. Onn. *Nonlinear Discrete Optimization: An Algorithmic Theory.* European Mathematical Society Publishing House, 2010. URL: https://books.google.ch/books?id=wzAGMQAACAAJ.

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

25 Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science, 2003.

26 R. Schultz, L. Stougie, and M. H. van der Vlerk. Two-stage stochastic integer programming: a survey. *Statistica Neerlandica*, 50(3):404–416, 1996. doi:10.1111/j.1467-9574.1996.tb01506.x.