TheoretiCS

Parameterized algorithms for block-structured integer programs with large entries

Received Jan 21, 2024
Revised Dec 3, 2024
Accepted Feb 4, 2025
Published Jul 18, 2025

Key words and phrases
block-structured integer programming, parameterized complexity, Graver basis

Jana Cslovjecsek [a] ✉
Martin Koutecký [b] ✉
Alexandra Lassota [c] ✉
Michał Pilipczuk [d] ✉
Adam Polak [e] ✉

a École Polytechnique Fédérale de Lausanne (EPFL), Switzerland

b Computer Science Institute, Charles University, Prague, Czech Republic

c Eindhoven University of Technology, Eindhoven, The Netherlands

d Institute of Informatics, University of Warsaw, Poland

e Department of Computing Sciences, Bocconi University, Milan, Italy

ABSTRACT. We study two classic variants of block-structured integer programming. *Two-stage stochastic programs* are integer programs of the form $\{A_i\mathbf{x} + D_i\mathbf{y}_i = \mathbf{b}_i \text{ for all } i = 1, \ldots, n\}$, where A_i and D_i are bounded-size matrices. Intuitively, this form corresponds to the setting when after setting a small set of global variables $\mathbf{x}$, the program can be decomposed into a possibly large number of bounded-size subprograms. On the other hand, *n-fold programs* are integer programs of the form $\{\sum_{i=1}^{n} C_i\mathbf{y}_i = \mathbf{a} \text{ and } D_i\mathbf{y}_i = \mathbf{b}_i \text{ for all } i = 1, \ldots, n\}$, where again C_i and D_i are bounded-size matrices. This form is natural for knapsack-like problems, where we have a large number of variables partitioned into small-size groups, each group needs to obey some set of local constraints, and there are only a few global constraints that link together all the variables.

A line of recent work established that the optimization problem for both two-stage stochastic programs and n-fold programs is fixed-parameter tractable when parameterized by the

A preliminary version of this article appeared at SODA 2024 [**14**]. Martin Koutecký was partially supported by Charles University project UNCE/SCI/004 and by the project 22-22997S of GA ČR. The work of Michał Pilipczuk on this manuscript is a part of a project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, Grant Agreement No 948057 — BOBR. Part of this work was done when Adam Polak and Alexandra Lassota were at École Polytechnique Fédérale de Lausanne, supported by the Swiss National Science Foundation projects *Lattice Algorithms and Integer Programming* (185030) and *Complexity of Integer Programming* (CRFS-2_207365). Part of this work was carried out while Alexandra Lassota was affiliated with Max Planck Institute for Informatics, SIC, Saarbrücken, Germany.

Cite as Jana Cslovjecsek, Martin Koutecký, Alexandra Lassota, Michał Pilipczuk, Adam Polak. Parameterized algorithms for block-structured integer programs with large entries. TheoretiCS, Volume 4 (2025), Article 15, 1-49.

https://theoretics.episciences.org
DOI 10.46298/theoretics.25.15

dimensions of relevant matrices A_i, C_i, D_i and by the maximum absolute value of any entry appearing in the constraint matrix. A fundamental tool used in these advances is the notion of the *Graver basis* of a matrix, and this tool heavily relies on the assumption that all the entries of the constraint matrix are bounded.

In this work, we prove that the parameterized tractability results for two-stage stochastic and n-fold programs persist even when one allows large entries in the global part of the program. More precisely, we prove the following:

— The feasibility problem for two-stage stochastic programs is fixed-parameter tractable when parameterized by the dimensions of matrices A_i, D_i and by the maximum absolute value of the entries of matrices D_i. That is, we allow matrices A_i to have arbitrarily large entries.

— The linear optimization problem for n-fold integer programs that are uniform – all matrices C_i are equal – is fixed-parameter tractable when parameterized by the dimensions of matrices C_i and D_i and by the maximum absolute value of the entries of matrices D_i. That is, we require that $C_i = C$ for all $i = 1, \ldots, n$, but we allow C to have arbitrarily large entries.

In the second result, the uniformity assumption is necessary; otherwise the problem is NP-hard already when the parameters take constant values. Both our algorithms are weakly polynomial: the running time is measured in the total bitsize of the input.

In both results, we depart from the approach that relies purely on Graver bases. Instead, for two-stage stochastic programs, we devise a reduction to integer programming with a bounded number of variables using new insights about the combinatorics of integer cones. For n-fold programs, we reduce a given n-fold program to an exponential-size program with bounded right-hand sides, which we consequently solve using a reduction to mixed integer programming with a bounded number of integral variables. For n-fold programs, we reduce a given n-fold program to an exponential-size program with bounded right-hand sides, which we consequently solve using a reduction to mixed integer programming with a bounded number of integral variables.

1. Introduction

We study two variants of integer programming problems, where the specific structure of the constraint matrix can be exploited for the design of efficient parameterized algorithms. *Two-stage stochastic programs* are integer programs of the following form:

$$\begin{aligned} \mathbf{x} \in \mathbb{Z}^k_{\geqslant 0},\ \mathbf{y}_i \in \mathbb{Z}^k_{\geqslant 0}, \qquad & \text{and} \\ A_i\mathbf{x} + D_i\mathbf{y}_i = \mathbf{b}_i \qquad & \text{for all } i = 1, 2, \ldots, n. \end{aligned} \tag{♠}$$

Here, A_i, D_i are integer $k \times k$ matrices[1] and each $\mathbf{b}_i$ is an integer vector of length k. This form arises naturally when the given integer program can be decomposed into multiple independent subprograms on disjoint variable sets $\mathbf{y}_i$, except there are several global variables $\mathbf{x}$ that are involved in all the subprograms and thus link them. See the survey of Shultz et al. [48] as well as an exposition article by Gavenčiak et al. [23] for examples of applications.

We moreover study *n-fold programs* which are integer programs of the form

$$\begin{aligned} \mathbf{y}_i &\in \mathbb{Z}^k_{\geqslant 0}, \\ \sum_{i=1}^{n} C_i \mathbf{y}_i &= \mathbf{a}, \qquad \text{and} \\ D_i \mathbf{y}_i &= \mathbf{b}_i \qquad \text{for all } i = 1, 2, \ldots, n, \end{aligned} \tag{♣}$$

where again C_i, D_i are integer $k \times k$ matrices and $\mathbf{a}, \mathbf{b}_i$ are integer vectors of length k. This kind of programs appears for knapsack-like and scheduling problems, where blocks of variables $\mathbf{y}_i$ correspond to some independent local decisions (for instance, whether to pack an item into the knapsack or not), and there are only a few linear constraints that involve all variables (for instance, that the capacity of the knapsack is not exceeded). See [11, 18, 23, 27, 33, 35, 36, 37] for examples of n-fold programs appearing naturally "in the wild". Figure 1 depicts constraint matrices of two-stage stochastic and n-fold programs.

$$\begin{bmatrix} A_1 & D_1 & & & \\ A_2 & & D_2 & & \\ \vdots & & & \ddots & \\ A_n & & & & D_n \end{bmatrix} \qquad \begin{bmatrix} C_1 & C_2 & \ldots & C_n \\ D_1 & & & \\ & D_2 & & \\ & & \ddots & \\ & & & D_n \end{bmatrix} \qquad \begin{bmatrix} B & C_1 & C_2 & \ldots & C_n \\ A_1 & D_1 & & & \\ A_2 & & D_2 & & \\ \vdots & & & \ddots & \\ A_n & & & & D_n \end{bmatrix}$$

Figure 1. Constraint matrices in two-stage stochastic, n-fold, and 4-block integer programs, respectively. (The last kind will be discussed later.) Every block is a $k \times k$ matrix, where k is the parameter. Empty spaces denote blocks of zeroes.

Both for two-stage stochastic programs and for n-fold programs, we can consider two computational problems. The simpler *feasibility problem* just asks whether the given program has a *solution*: an evaluation of variables in nonnegative integers that satisfies all the constraints. In the harder *(linear) optimization problem*, we are additionally given an integer weight w_x for every variable x appearing in the program, and the task is to minimize $\sum_{x\colon \text{variable}} w_x \cdot x$ over all solutions.

1 Reliance on square matrices is just for convenience of presentation. The setting where blocks are rectangular matrices with dimensions bounded by k can be reduced to the setting of $k \times k$ square matrices by just padding with 0s.

Two-stage stochastic and n-fold programs have recently gathered significant interest in the theoretical community for two reasons. On one hand, it turns out that for multiple combinatorial problems, their natural integer programming formulations take either of the two forms. On the other hand, one can actually design highly non-trivial fixed-parameter algorithms for the optimization problem for both two-stage stochastic and n-fold programs; we will review them in a minute. Combining this two points yields a powerful algorithmic technique that allowed multiple new tractability results and running times improvements for various problems of combinatorial optimization; see [11, 23, 27, 33, 32, 35, 36, 37, 42] for examples.

Delving more into technical details, if by Δ we denote the maximum absolute value of any entry in the constraint matrix, then the optimization problem for

- two-stage stochastic programs (♠) can be solved in time $2^{\Delta^{O(k^2)}} \cdot n \log^{O(k^2)} n$ [13]; and
- n-fold programs (♣) can be solved in time $(k\Delta)^{O(k^3)} \cdot n \log^{O(k^2)} n$ [12].

The results above are in fact pinnacles of an over-a-decade-long sequence of developments, which gradually improved both the generality of the results and the running times [3, 12, 13, 18, 19, 20, 25, 28, 30, 39], as well as provided complexity lower bounds [26, 38]. We refer the interested reader to the monumental manuscript of Eisenbrand et al. [20] which provides a comprehensive perspective on this research area.

We remark that the tractability results presented above can be also extended to the setting where the global-local block structure present in two-stage stochastic and n-fold programs can be iterated further, roughly speaking to trees of bounded depth. This leads to the study of integer programs with bounded *primal* or *dual treedepth*, for which analogous tractability results have been established. Since these notions will not be of interest in this work, we refrain from providing further details and refer the interested reader to the works relevant for this direction [5, 6, 10, 12, 13, 19, 20, 30, 31, 38, 39].

All the above-mentioned works, be it on two-stage stochastic or n-fold programs, or on programs of bounded primal or dual treedepth, crucially rely on one tool: the notion of the *Graver basis* of a matrix. Formal definition can be found in Section 5.1, but intuitively speaking, the Graver basis of a matrix A consists of "minimal steps" within the lattice of integer points belonging to the kernel of A. Therefore, the maximum norm of an element of the Graver basis reflects the "granularity" of this lattice. And so, the two fundamental observations underlying all the discussed developments are the following:

- in two-stage stochastic matrices (or more generally, matrices of bounded primal treedepth), the ℓ_∞ norm of the elements of the Graver basis is bounded in terms of k (the dimension of every block) and the maximum absolute value of any entry appearing in the matrix (see [20, Lemma 28]); and
- an analogous result holds for n-fold matrices (or more generally, matrices of bounded dual treedepth) and the ℓ_1 norm (see [20, Lemma 30]).

Based on these observations, various algorithmic strategies, including augmentation frameworks [28, 39] and proximity arguments [12, 13, 19], can be employed to solve respective integer programs.

The drawback of the Graver-based approach is that it heavily relies on the assumption that all the entries of the input matrices are (parametrically) bounded. Indeed, the norms of the elements of the Graver basis are typically at least as large as the entries, so lacking any upper bound on the latter renders the methodology inapplicable. This is in stark contrast with the results on fixed-parameter tractability of integer programming parameterized by the number of variables [15, 16, 22, 29, 44, 47], which rely on different tools and for which no bound on the absolute values of the entries is required. In fact, two-stage stochastic programs generalize programs with a bounded number of variables (just do not use variables $\mathbf{y}_i$), yet the current results for two-stage stochastic programs do *not* generalize those for integer programs with few variables, because they additionally assume a bound on the absolute values of the entries.

The goal of this paper is to understand to what extent one can efficiently solve two-stage stochastic and n-fold programs while allowing large entries on input.

Our contribution. We prove that both the feasibility problem for two-stage stochastic programs and the optimization problem for uniform n-fold programs (that is, where $C_1 = C_2 = \ldots = C_n$) can be solved in fixed-parameter time when parameterized by the dimensions of the blocks and the maximum absolute value of any entry appearing in the diagonal blocks D_i. That is, we allow the entries of the global blocks (A_i and C_i, respectively) to be arbitrarily large, and in the case of n-fold programs, we require that all blocks C_i are equal. The statements below summarize our results. ($\|P\|$ denotes the total bitsize of a program P.)

THEOREM 1.1. *The feasibility problem for two-stage stochastic programs P of the form (♠) can be solved in time $f(k, \max_i \|D_i\|_\infty) \cdot \|P\|$ for a computable function f, where k is the dimension of all the blocks A_i, D_i.*

THEOREM 1.2. *The optimization problem for n-fold programs P of the form (♣) that are uniform (that is, satisfy $C_1 = \ldots = C_n$) can be solved in time $f(k, \max_i \|D_i\|_\infty) \cdot \|P\|^{O(1)}$ for a computable function f, where k is the dimension of all the blocks C_i, D_i.*

We remark that in Theorem 1.1 it is necessary to include $\max_i \|D_i\|_\infty$ among the parameters, because with no bound on the entries of the constraint matrix, the feasibility problem for two-stage stochastic programs becomes NP-hard already for a constant k. An easy way to see it is via a straightforward reduction from an NP-hard problem called Good Simultaneous Approximation [43], where the input consists of a rational vector $\mathbf{a} \in \mathbb{Q}^n$, a desired error bound $\epsilon \in \mathbb{Q}$, and an upper bound $N \in \mathbb{Z}_{\geqslant 0}$, and the goal is to decide if there exists an integer multiplier $Q \in \{1, 2, \ldots, N\}$ such that $Q\mathbf{a}$ is ϵ-close in the infinity norm to an integer vector, i.e., $\exists_{\mathbf{b} \in \mathbb{Z}^n} \|Q\mathbf{a} - \mathbf{b}\|_\infty \leqslant \epsilon$. In Appendix B we give a different argument, reducing directly from 3-SAT,

and showing that two-stage stochastic feasibility is actually *strongly* NP-hard for $k = 16$. This rules out also any running time of the form $f(k) \cdot (\|P\| + \max_i \|D_i\|_\infty)^{O(1)}$.

The uniformity condition in Theorem 1.2 is also necessary (unless P = NP), as one can very easily reduce Subset Sum to the feasibility problem for n-fold programs with $k = 2$ and D_i being $\{0, 1\}$-matrices. Indeed, given an instance of Subset Sum consisting of positive integers $a_1, \ldots, a_n$ and a target integer t, we can write the following n-fold program on variables $y_1, \ldots, y_n, y'_1, \ldots, y'_n \in \mathbb{Z}_{\geqslant 0}$: $y_i + y'_i = 1$ for all $i = 1, \ldots, n$ and $\sum_{i=1}^{n} a_i y_i = t$. We remark that uniform n-fold programs are actually of the highest importance, as this form typically arises in applications. In fact, many of the previous works named such problems "n-fold", while our formulation (♣) would be called "generalized n-fold".

We also remark that the algorithm of Theorem 1.2 does not use the assumption that the number of *rows* of matrix C is bounded by k (formally, in the precise, statement Theorem 5.1, we do not consider this number among parameters). However, we stress that the bound on the number of *columns* of C is heavily exploited, which sets our approach apart from many of the previous works [12, 19, 39].

Further, observe that Theorem 1.1 applies only to the feasibility problem for two-stage stochastic programs. We actually do not know whether Theorem 1.1 can be extended to the optimization problem as well, and we consider determining this an outstanding open problem. We will discuss its motivation in more details later on. Also, we remark that Theorem 1.1 seems to be the first algorithm for feasibility of two-stage stochastic programs that achieves truly linear dependence of the running time on the total input size; the earlier algorithms of [13, 39] had at least some additional polylogarithmic factors.

Finally, note that the algorithms of Theorems 1.1 and 1.2 are not strongly polynomial (i.e., the running time depends on the total bitsize of the input, rather than is counted in the number of arithmetic operations), while the previous algorithms of [12, 13, 19, 39] for the stronger parameterization are. At least in the case of Theorem 1.1, this is justified, as the problem considered there generalizes integer programming parameterized by the number of variables, for which strongly polynomial FPT algorithms are not known.

Not surprisingly, the proofs of Theorems 1.1 and 1.2 depart from the by now standard approach through Graver bases; they are based on entirely new techniques, with some key Graver-based insight needed in the case of Theorem 1.2. In both cases, the problem is ultimately reduced to (mixed) integer programming with a bounded number of (integral) variables, and this allows us to cope with large entries on input. We expand the discussion of our techniques in Section 2, which contains a technical overview of the proofs.

4-block programs. Finally, we would like to discuss another motivation for investigating two-stage stochastic and n-fold programs with large entries, namely the open question about the parameterized complexity of 4-*block integer programming*. 4-block programs are programs

in which the constraint matrix has the block-structured form depicted in the right panel of Figure 1; note that this form naturally generalizes both two-stage stochastic and n-fold programs. It is an important problem in the area to determine whether the feasibility problem for 4-block programs can be solved in fixed-parameter time when parameterized by the dimension of blocks k and the maximum absolute value of any entry in the input matrix. The question was asked by Hemmecke et al. [24], who proposed an XP algorithm for the problem. Improvements on the XP running time were reported by Chen et al. [9], and FPT algorithms for special cases were proposed by Chen et al. [8]; yet no FPT algorithm for the problem in full generality is known so far. We remark that recently, Chen et al. [7] studied the complexity of 4-block programming while allowing large entries in all the four blocks of the matrix. They showed that then the problem becomes NP-hard already for blocks of constant dimension, and they discussed a few special cases that lead to tractability.

We observe that in the context of the feasibility problem for uniform 4-block programs (i.e., with $A_i = A$ and $C_i = C$ for all $i = 1, \ldots, n$), it is possible to emulate large entries within the global blocks A, B, C using only small entries at the cost of adding a bounded number of auxiliary variables. This yields the following reduction, which we prove in Appendix A.

OBSERVATION 1.3. *Suppose the feasibility problem for uniform 4-block programs can be solved in time $f(k, \Delta) \cdot \|P\|^{O(1)}$ for some computable function f, where k is the dimension of every block and Δ is the maximum absolute value of any entry in the constraint matrix. Then the feasibility problem for uniform 4-block programs can be also solved in time $g(k, \max_i \|D_i\|_\infty) \cdot \|P\|^{O(1)}$ for some computable function g under the assumption that all the absolute values of the entries in matrices A, B, C are bounded by n.*

Consequently, to approach the problem of fixed-parameter tractability of 4-block integer programming, it is imperative to understand first the complexity of two-stage stochastic and n-fold programming with large entries allowed in the global blocks. And this is precisely what we do in this work.

We believe that the next natural step towards understanding the complexity of 4-block integer programming would be to extend Theorem 1.1 to the optimization problem; that is, to determine whether optimization of two-stage stochastic programs can be solved in fixed-parameter time when parameterized by k and $\max_i \|D_i\|_\infty$. Indeed, lifting the result from feasibility to the optimization problem roughly corresponds to adding a single constraint that links all the variables, and 4-block programs differ from two-stage stochastic programs precisely in that there may be up to k such additional linking constraints. Thus, we hope that the new approach to block-structured integer programming presented in this work may pave the way towards understanding the complexity of solving 4-block integer programs.

Acknowledgements. This research has been initiated during the trimester on Discrete Optimization at the Hausdorff Research Institute for Mathematics (HIM) in Bonn, Germany. We thank the organizers of the trimester for creating a friendly and motivating research environment. We also thank Eleonore Bach, Fritz Eisenbrand, and Robert Weismantel, for pointing us to the work of Aliev and Henk [2]. Finally, we thank an anonymous reviewer for suggesting a simpler proof of Lemma 5.6 as well as pointing out the possibility of obtaining the lower bound presented in Appendix B.

2. Overview

In this section, we provide a technical overview of our results aimed at presenting the main ideas and new conceptual contributions.

2.1 Two-stage stochastic programming

We start with an overview on the proof of Theorem 1.1. We will heavily rely on the combinatorics of integer and polyhedral cones, so let us recall basic definitions and properties.

Cones. Consider an integer matrix D with t columns and k rows. The *polyhedral cone* spanned by D is the set $\mathsf{cone}(D) := \{D\mathbf{y} \colon \mathbf{y} \in \mathbb{R}^t_{\geqslant 0}\} \subseteq \mathbb{R}^k_{\geqslant 0}$, or equivalently, the set of all vectors in $\mathbb{R}^k_{\geqslant 0}$ expressible as nonnegative combinations of the columns of D. Within the polyhedral cone, we have the *integer cone* where we restrict attention to nonnegative integer combinations: $\mathsf{intCone}(D) := \{D\mathbf{y} \colon \mathbf{y} \in \mathbb{Z}^t_{\geqslant 0}\} \subseteq \mathbb{Z}^k$. Finally, the *integer lattice* is the set $\mathsf{lattice}(D) := \{D\mathbf{y} \colon \mathbf{y} \in \mathbb{Z}^t\} \subseteq \mathbb{Z}^k$ which comprises all integer combinations of columns of D with possibly negative coefficients.

Clearly, not every integer vector in $\mathsf{cone}(D)$ has to belong to $\mathsf{intCone}(D)$. It is not even necessarily the case that $\mathsf{intCone}(D) = \mathsf{cone}(D) \cap \mathsf{lattice}(D)$, as there might be vectors that can be obtained both as a nonnegative combination and as an integer combination of columns of D, but every such integer combination must necessarily contain negative coefficients. To see an example, note that in dimension $k = 1$, this is the Frobenius problem: supposing all entries of D are positive integers, the elements of $\mathsf{intCone}(D)$ are essentially all nonnegative numbers divisible by the gcd (greatest common divisor) of the entries of D, except that for small numbers there might be some aberrations: a positive integer of order $O(\|D\|_\infty^2)$ may not be presentable as a nonnegative combination of the entries of D, even assuming it is divisible by the gcd of the entries of D.

However, the Frobenius example suggests that the equality $\mathsf{intCone}(D) = \mathsf{cone}(D) \cap \mathsf{lattice}(D)$ is almost true, except for aberrations near the boundary of $\mathsf{cone}(D)$. We forge this intuition into a formal statement presented below that says roughly the following: if one takes a look at $\mathsf{intCone}(D)$ at a large scale, by restricting attention to integer vectors $\mathbf{v} \in \mathbb{Z}^k$ with fixed

remainders of entries modulo some large integer B, then $\mathsf{intCone}(D)$ behaves like a polyhedron. In the following, for a positive integer B and a vector $\mathbf{r} \in \{0, 1, \ldots, B-1\}^k$, we let $\Lambda^B_{\mathbf{r}}$ be the set of all vectors $\mathbf{v} \in \mathbb{Z}^k$ such that $\mathbf{v} \equiv \mathbf{r} \bmod B$, which means $v_i \equiv r_i \bmod B$ for all $i \in \{1, \ldots, k\}$.

THEOREM 2.1 (Reduction to Polyhedral Constraints, see Theorem 4.9). *Let D be an integer matrix with t columns and k rows. Then there exists a positive integer B, computable from D, such that for every $\mathbf{r} \in \{0, 1, \ldots, B-1\}^k$, there exists a polyhedron $Q_{\mathbf{r}}$ such that*

$$\Lambda^B_{\mathbf{r}} \cap \mathsf{intCone}(D) = \Lambda^B_{\mathbf{r}} \cap Q_{\mathbf{r}}.$$

Moreover, a representation of such a polyhedron $Q_{\mathbf{r}}$ can be computed given D and $\mathbf{r}$.

In other words, Theorem 2.1 states that if one fixes the remainders of entries modulo B, then membership in the integer cone can be equivalently expressed through a finite system of linear inequalities. Before we sketch the proof of Theorem 2.1, let us discuss how to use this to solve two-stage stochastic programs.

The algorithm. Consider a two-stage stochastic program $P = (A_i, D_i, \mathbf{b}_i : i \in \{1, \ldots, n\})$ such that blocks A_i, D_i are integer $k \times k$ matrices and all entries of blocks D_i are bounded in absolute value by Δ. The feasibility problem for P can be understood as the question about satisfaction of the following sentence, where all quantifications range over $\mathbb{Z}^k_{\geqslant 0}$:

$$\exists_{\mathbf{x}} \left(\bigwedge_{i=1}^{n} \exists_{\mathbf{y}_i} A_i\mathbf{x} + D_i\mathbf{y}_i = \mathbf{b}_i \right), \quad \text{or equivalently,} \quad \exists_{\mathbf{x}} \left(\bigwedge_{i=1}^{n} \mathbf{b}_i - A_i\mathbf{x} \in \mathsf{intCone}(D_i) \right). \tag{1}$$

Applying Theorem 2.1 to each matrix D_i yields a positive integer B_i. Note that there are only at most $(2\Delta + 1)^{k^2}$ different matrices D_i appearing in P, which also bounds the number of different integers B_i. By replacing all B_is with their least common multiple, we may assume that $B_1 = B_2 = \ldots = B_n = B$. Note that B is bounded by a computable function of Δ and k.

Consider a hypothetical solution $\mathbf{x}$, $(\mathbf{y}_i : i \in \{1, \ldots, n\})$ to P. We guess, by branching into B^k possibilities, a vector $\mathbf{r} \in \{0, 1, \ldots, B-1\}^k$ such that $\mathbf{x} \equiv \mathbf{r} \bmod B$. Having fixed $\mathbf{r}$, we know how the vectors $\mathbf{b}_i - A_i\mathbf{x}$ look like modulo B, hence by Theorem 2.1, we may replace the assertion $\mathbf{b}_i - A_i\mathbf{x} \in \mathsf{intCone}(D_i)$ with the assertion $\mathbf{b}_i - A_i\mathbf{x} \in Q_{\mathbf{r}_i}$, where $\mathbf{r}_i \in \{0, 1, \ldots, B-1\}^k$ is the unique vector such that $\mathbf{b}_i - A_i\mathbf{r} \equiv \mathbf{r}_i \bmod B$. Thus, (1) can be rewritten to the sentence

$$\bigvee_{\mathbf{r} \in \{0,1,\ldots,B-1\}^k} \exists_{\mathbf{x}} \ (\mathbf{x} \equiv \mathbf{r} \bmod B) \wedge \left(\bigwedge_{i=1}^{n} \mathbf{b}_i - A_i\mathbf{x} \in Q_{\mathbf{r}_i} \right),$$

which is equivalent to

$$\bigvee_{\mathbf{r} \in \{0,1,\ldots,B-1\}^k} \exists_{\mathbf{x}} \exists_{\mathbf{z}} \ (\mathbf{x} = B \cdot \mathbf{z} + \mathbf{r}) \wedge \left(\bigwedge_{i=1}^{n} \mathbf{b}_i - A_i\mathbf{x} \in Q_{\mathbf{r}_i} \right). \tag{2}$$

Verifying satisfiability of (2) boils down to solving B^k integer programs on $2k$ variables $\mathbf{x}$ and $\mathbf{z}$ and linearly FPT many constraints, which can be done in linear fixed-parameter time using standard algorithms, for instance that of Kannan [29].

We remark that the explanation presented above highlights that Theorem 2.1 can be understood as a quantifier elimination result in the arithmetic theory of integers. This may be of independent interest, but we do not pursue this direction in this work.

Reduction to polyhedral constraints. We are left with sketching the proof of Theorem 2.1. Let $\mathcal{Z} := \Lambda^B_{\mathbf{r}} \cap \mathsf{intCone}(D)$. Our goal is to understand that $\mathcal{Z}$ can be expressed as the points of $\Lambda^B_{\mathbf{r}}$ that are contained in some polyhedron $Q = Q_{\mathbf{r}}$.

The first step is to understand $\mathsf{cone}(D)$ itself as a polyhedron. This understanding is provided by a classic theorem of Weyl [49]: given D, one can compute a set of integer vectors $\mathcal{F} \subseteq \mathbb{Z}^k$ such that

$$\mathsf{cone}(D) = \{\mathbf{v} \in \mathbb{R}^k \mid \langle \mathbf{f}, \mathbf{v} \rangle \geqslant 0 \text{ for all } \mathbf{f} \in \mathcal{F}\}.$$

Here, $\langle \cdot, \cdot \rangle$ denotes the scalar product in $\mathbb{R}^k$. We will identify vectors $\mathbf{f} \in \mathcal{F}$ with their associated linear functionals $\mathbf{v} \mapsto \langle \mathbf{f}, \mathbf{v} \rangle$. Thus, $\mathsf{cone}(D)$ comprises all vectors $\mathbf{v}$ that have nonnegative evaluations on all functionals in $\mathcal{F}$. It is instructive to also think of the elements of $\mathcal{F}$ as of the facets of $\mathsf{cone}(D)$ understood as a polyhedron, where the functional associated with $\mathbf{f} \in \mathcal{F}$ measures the distance from the corresponding facet.

Recall that in the context of Theorem 2.1, we consider vectors of $\Lambda^B_{\mathbf{r}}$, that is, vectors $\mathbf{v} \in \mathbb{Z}^k$ such that $\mathbf{v} \equiv \mathbf{r} \bmod B$. Then $\langle \mathbf{f}, \mathbf{v} \rangle \equiv \langle \mathbf{f}, \mathbf{r} \rangle \bmod B$ for every $\mathbf{f} \in \mathcal{F}$, hence we can find a unique integer $p_{\mathbf{f}} \in \{0, 1, \ldots, B-1\}$, $p_{\mathbf{f}} \equiv \langle \mathbf{f}, \mathbf{r} \rangle \bmod B$, such that $\langle \mathbf{f}, \mathbf{v} \rangle \equiv p_{\mathbf{f}} \bmod B$ for all $\mathbf{v} \in \Lambda^B_{\mathbf{r}}$. Now $\langle \mathbf{f}, \mathbf{v} \rangle$ is also nonnegative provided $\mathbf{v} \in \mathsf{cone}(D)$, hence

$$\langle \mathbf{f}, \mathbf{v} \rangle \in \{p_{\mathbf{f}}, p_{\mathbf{f}} + B, p_{\mathbf{f}} + 2B, \ldots\} \qquad \text{for all } \mathbf{f} \in \mathcal{F} \text{ and } \mathbf{v} \in \Lambda^B_{\mathbf{r}} \cap \mathsf{cone}(D).$$

Now comes the key distinction about the behavior of $\mathbf{v} \in \Lambda^B_{\mathbf{r}} \cap \mathsf{cone}(D)$ with respect to $\mathbf{f} \in \mathcal{F}$: we say that $\mathbf{f}$ is *tight* with respect to $\mathbf{v}$ if $\langle \mathbf{f}, \mathbf{v} \rangle = p_{\mathbf{f}}$, and is *not tight* otherwise, that is, if $\langle \mathbf{f}, \mathbf{v} \rangle \geqslant p_{\mathbf{f}} + B$. Recall that in the context of Theorem 2.1, we are eventually free to choose B to be large enough. Intuitively, this means that if $\mathbf{f}$ is not tight for $\mathbf{v}$, then $\mathbf{v}$ lies far from the facet corresponding to $\mathbf{f}$ and there is a very large slack in the constraint posed by $\mathbf{f}$ understood as a functional. On the other hand, if $\mathbf{f}$ is tight with respect to $\mathbf{v}$, then $\mathbf{v}$ is close to the boundary of $\mathsf{cone}(D)$ at the facet corresponding to $\mathbf{f}$, and there is a potential danger of observing Frobenius-like aberrations at $\mathbf{v}$.

Thus, the set $\mathcal{R} := \Lambda^B_{\mathbf{r}} \cap \mathsf{cone}(D)$ can be partitioned into subsets $\{\mathcal{R}_{\mathcal{G}} \colon \mathcal{G} \subseteq \mathcal{F}\}$ defined as follows: $\mathcal{R}_{\mathcal{G}}$ comprises all vectors $\mathbf{v} \in \mathcal{R}$ such that $\mathcal{G}$ is exactly the set of functionals $\mathbf{f} \in \mathcal{F}$ that are tight with respect to $\mathbf{v}$. Our goal is to prove that each set $\mathcal{R}_{\mathcal{G}}$ behaves uniformly with respect to $\mathcal{Z}$: it is either completely disjoint or completely contained in $\mathcal{Z}$. To start the discussion, let us look at the particular case of $\mathcal{R}_{\mathcal{G}}$ for $\mathcal{G} = \emptyset$. These are vectors that are deep inside $\mathsf{cone}(D)$, for

which no functional in $\mathcal{F}$ is tight. For these vectors, we use the following lemma, which is the cornerstone of our proof.

LEMMA 2.2 (Deep-in-the-Cone Lemma, simplified version of Lemma 4.6). *There exists a constant M, depending only on D, such that the following holds. Suppose $\mathbf{v} \in \mathsf{cone}(D) \cap \mathbb{Z}^k$ is such that $\langle \mathbf{f}, \mathbf{v} \rangle > M$ for all $\mathbf{f} \in \mathcal{F}$. Then $\mathbf{v} \in \mathsf{intCone}(D)$ if and only if $\mathbf{v} \in \mathsf{lattice}(D)$.*

PROOF. The left-to-right implication is obvious, hence let us focus on the right-to-left implication. Suppose then that $\mathbf{v} \in \mathsf{lattice}(D)$.

Let $\mathbf{w} = \sum_{\mathbf{d} \in D} L \cdot \mathbf{d}$, where the summation is over the columns of D and L is a positive integer to be fixed later. Observe that for every $\mathbf{f} \in \mathcal{F}$, we have $\langle \mathbf{f}, \mathbf{v} - \mathbf{w} \rangle > M - L \cdot \sum_{\mathbf{d} \in D} \langle \mathbf{f}, \mathbf{d} \rangle$. Therefore, if we choose M to be not smaller than $L \cdot \max_{\mathbf{f} \in \mathcal{F}} \|\mathbf{f}\|_1 \cdot \|D\|_\infty$, then we are certain that $\langle \mathbf{f}, \mathbf{v} - \mathbf{w} \rangle \geqslant 0$ for all $\mathbf{f} \in \mathcal{F}$, and hence $\mathbf{v} - \mathbf{w} \in \mathsf{cone}(D)$. Consequently, we can write $\mathbf{v} - \mathbf{w} = D\mathbf{y}$ for some $\mathbf{y} \in \mathbb{R}^t_{\geqslant 0}$. Let $\mathbf{y}' \in \mathbb{Z}^t_{\geqslant 0}$ be such that $y'_i = \lfloor y_i \rfloor$ for all $i \in \{1, \ldots, t\}$, and let $\mathbf{v}' = \mathbf{w} + D\mathbf{y}'$. Then

$$\|\mathbf{v} - \mathbf{v}'\|_\infty = \|D(\mathbf{y} - \mathbf{y}')\|_\infty \leqslant t \cdot \|D\|_\infty.$$

On the other hand, we clearly have $\mathbf{v}' \in \mathsf{intCone}(D)$ and by assumption, $\mathbf{v} \in \mathsf{lattice}(D)$. It follows that $\mathbf{v} - \mathbf{v}' \in \mathsf{lattice}(D)$. From standard bounds, see e.g. [45], it follows that there exists $\mathbf{z} \in \mathbb{Z}^t$ with $\mathbf{v} - \mathbf{v}' = D\mathbf{z}$ such that $\|\mathbf{z}\|_1$ is bounded by a function of D and $\|\mathbf{v} - \mathbf{v}'\|_\infty$, which in turn is again bounded by a function of D as explained above. (Note here that t is the number of columns of D, hence it also depends only on D.) This means that if we choose L large enough depending on D, we are certain that $\|\mathbf{z}\|_1 \leqslant L$. Now, it remains to observe that

$$\mathbf{v} = \mathbf{w} + D\mathbf{y}' + (\mathbf{v} - \mathbf{v}') = D(L \cdot \mathbf{1} + \mathbf{y}' + \mathbf{z}),$$

where $\mathbf{1}$ denotes the vector of t ones, and that all the entries of $L \cdot \mathbf{1} + \mathbf{y}' + \mathbf{z}$ are nonnegative integers. This proves that $\mathbf{v} \in \mathsf{intCone}(D)$. ■

We remark that the statement of Lemma 2.2 actually follows from results present in the literature, concerning the notion of *diagonal Frobenius numbers*. See the work of Aliev and Henk [2] for a broader discussion and pointers to earlier works, as well as the works of Aggrawal et al. [1] and of Bach et al. [4] for the newest developments on optimal bounds. As we will discuss in a moment, in this work we actually use a generalization of Lemma 2.2.

Consider any $\mathbf{u}, \mathbf{v} \in \mathcal{R}$. Since all the entries of $\mathbf{u} - \mathbf{v}$ are divisible by B, it is not hard to prove the following: if we choose B to be a large enough factorial, then $\mathbf{u} \in \mathsf{lattice}(D)$ if and only if $\mathbf{v} \in \mathsf{lattice}(D)$. Hence, from Lemma 2.2 it follows that $\mathcal{R}_\emptyset$ is either entirely disjoint or entirely contained in $\mathcal{Z}$.

A more involved reasoning based on the same fundamental ideas, but using a generalization of Lemma 2.2, yields the following lemma, which tackles also the case when some functionals of $\mathcal{F}$ are tight with respect to the considered vectors.

LEMMA 2.3 (see Claim 4.9.2). *Suppose* $\mathbf{u}, \mathbf{v} \in \mathcal{R}$ *are such that for every* $\mathbf{f} \in \mathcal{F}$*, if* $\mathbf{f}$ *is tight with respect to* $\mathbf{u}$*, then* $\mathbf{f}$ *is also tight with respect to* $\mathbf{v}$*. Then* $\mathbf{u} \in \mathcal{Z}$ *implies* $\mathbf{v} \in \mathcal{Z}$*.*

We remark that the proof of Lemma 2.3 actually requires more work and more ideas than those presented in the proof of Lemma 2.2. In essence, one needs to partition functionals that are tight with respect to $\mathbf{u}$ into those that are very tight (have very small $p_{\mathbf{f}}$) and those that are only slightly tight (have relatively large $p_{\mathbf{f}}$) in order to create a sufficient gap between very tight and slightly tight functionals. Having achieved this, a delicate variant of the reasoning from the proof of Lemma 2.2 can be applied. It is important that whenever a functional $\mathbf{f} \in \mathcal{F}$ is tight with respect to both $\mathbf{u}$ and $\mathbf{v}$, we actually know that $\langle \mathbf{f}, \mathbf{u} \rangle = \langle \mathbf{f}, \mathbf{v} \rangle = p_{\mathbf{f}}$. Note that this is exactly the benefit achieved by restricting attention to the vectors of $\Lambda^B_{\mathbf{r}}$.

Using Lemma 2.3, we can immediately describe how the structure of $\mathcal{Z}$ relates to that of $\mathcal{R}$.

COROLLARY 2.4 (see Claim 4.9.3). *For every* $\mathcal{G} \subseteq \mathcal{F}$*, either* $\mathcal{R}_{\mathcal{G}} \cap \mathcal{Z} = \emptyset$ *or* $\mathcal{R}_{\mathcal{G}} \subseteq \mathcal{Z}$*. Moreover, if* $\mathcal{R}_{\mathcal{G}} \subseteq \mathcal{Z}$ *and* $\mathcal{R}_{\mathcal{G}}$ *is non-empty, then* $\mathcal{R}_{\mathcal{G}'} \subseteq \mathcal{Z}$ *for all* $\mathcal{G}' \subseteq \mathcal{G}$*.*

Corollary 2.4 suggests now how to define the polyhedron $\mathcal{Q}$. Namely, $\mathcal{Q}$ is defined as the set of all $\mathbf{v} \in \mathbb{R}^k$ satisfying the following linear inequalities:

— inequalities $\langle \mathbf{f}, \mathbf{v} \rangle \geqslant 0$ for all $\mathbf{f} \in \mathcal{F}$ that define $\mathrm{cone}(D)$; and
— for every $\mathcal{G} \subseteq \mathcal{F}$ such that $\mathcal{R}_{\mathcal{G}}$ is nonempty and $\mathcal{R}_{\mathcal{G}} \cap \mathcal{Z} = \emptyset$, the inequality

$$\sum_{\mathbf{g} \in \mathcal{G}} \langle \mathbf{g}, \mathbf{v} \rangle \geqslant 1 + \sum_{\mathbf{g} \in \mathcal{G}} p_{\mathbf{g}}.$$

In essence, the inequalities from the second point "carve out" those parts $\mathcal{R}_{\mathcal{G}}$ that should not be included in $\mathcal{Z}$. We note that computing the inequalities defining $\mathcal{Q}$ requires solving several auxiliary integer programs to figure out for which $\mathcal{G} \subseteq \mathcal{F}$ the corresponding inequality should be included.

It is now straightforward to verify, using all the accumulated observations, that indeed $\mathcal{Z} = \mathcal{R} \cap \mathcal{Q}$ as required. This concludes a sketch of the proof of Theorem 2.1.

2.2 *n*-fold programming

We now give an overview of the proof of Theorem 1.2. For simplicity, we make the following assumptions.

— We focus on the feasibility problem instead of optimization. At the very end, we will remark on what additional ideas are needed to also tackle the optimization problem.
— We assume that all the diagonal blocks D_i are equal: $D_i = D$ for all $i \in \{1, \ldots, n\}$, where D is a $k \times k$ integer matrix with $\|D\|_\infty \leqslant \Delta$. This is only a minor simplification because there are only $(2\Delta + 1)^{k^2}$ different matrices D_i with $\|D_i\|_\infty \leqslant \Delta$, and in the general case, we

simply treat every such possible matrix "type" separately using the reasoning from the simplified case.

Breaking up bricks. Basic components of the given n-fold program $P = (C, D, \mathbf{a}, \mathbf{b}_i : i \in \{1, \ldots, n\})$ are *bricks*: programs $D\mathbf{y}_i = \mathbf{b}_i$ for $i \in \{1, \ldots, n\}$ that encode local constraints on the variables $\mathbf{y}_i$. While the entries of D are bounded in absolute values by the parameter Δ, we do not assume any bound on the entries of vectors $\mathbf{b}_i$. This poses an issue, as different bricks may have very different behaviors.

The key idea in our approach is to simplify the program P by iteratively breaking up every brick $D\mathbf{y} = \mathbf{b}$ into two bricks $D\mathbf{y} = \mathbf{b}'$ and $D\mathbf{y} = \mathbf{b}''$ with strictly smaller right-hand sides $\mathbf{b}', \mathbf{b}''$, until eventually, we obtain an equivalent n-fold program P' in which all right-hand sides have ℓ_∞-norms bounded in terms of the parameters. The following lemma is the crucial new piece of technology used in our proof. (Here, we use the *conformal order* on $\mathbb{Z}^k$: we write $\mathbf{u} \sqsubseteq \mathbf{v}$ if $|\mathbf{u}[i]| \leqslant |\mathbf{v}[i]|$ and $\mathbf{u}[i] \cdot \mathbf{v}[i] \geqslant 0$ for all $i \in \{1, \ldots, k\}$.)

LEMMA 2.5 (Brick Decomposition Lemma, see Lemma 5.4). *There exists a function $g(k, \Delta) \in 2^{(k\Delta)^{O(k)}}$ such that the following holds. Let D be an integer matrix with t columns and k rows and all absolute values of its entries bounded by Δ. Further, let $\mathbf{b} \in \mathbb{Z}^k$ be an integer vector such that $\|\mathbf{b}\|_\infty > g(k, \Delta)$. Then there are non-zero vectors $\mathbf{b}', \mathbf{b}'' \in \mathbb{Z}^k$ such that:*

- *$\mathbf{b}', \mathbf{b}'' \sqsubseteq \mathbf{b}$ and $\mathbf{b} = \mathbf{b}' + \mathbf{b}''$; and*
- *for every $\mathbf{v} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ satisfying $D\mathbf{v} = \mathbf{b}$, there exist $\mathbf{v}', \mathbf{v}'' \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ such that*

$$\mathbf{v} = \mathbf{v}' + \mathbf{v}'', \qquad D\mathbf{v}' = \mathbf{b}', \qquad \text{and} \qquad D\mathbf{v}'' = \mathbf{b}''.$$

In other words, Lemma 2.5 states that the brick $D\mathbf{y} = \mathbf{b}$ can be broken into two new bricks $D\mathbf{y}' = \mathbf{b}'$ and $D\mathbf{y}'' = \mathbf{b}''$ with conformally strictly smaller $\mathbf{b}', \mathbf{b}''$ so that *every* potential solution $\mathbf{v}$ to $D\mathbf{y} = \mathbf{b}$ can be decomposed into solutions $\mathbf{v}', \mathbf{v}''$ to the two new bricks. It is easy to see that this condition implies that in P, we may replace the brick $D\mathbf{y} = \mathbf{b}$ with $D\mathbf{y}' = \mathbf{b}'$ and $D\mathbf{y}'' = \mathbf{b}''$ without changing feasibility or, in the case of the optimization problem, the minimum value of the optimization goal. In the latter setting, both new bricks inherit the optimization vector $\mathbf{c}_i$ from the original brick.

Before we continue, let us comment on the proof of Lemma 2.5. We use two ingredients. The first one is the following fundamental result of Klein [30]. (Here, for a multiset of vectors A, by $\sum A$ we denote the sum of all the vectors in A.)

LEMMA 2.6 (Klein Lemma, variant from [13]). *Let $T_1, \ldots, T_n$ be non-empty multisets of vectors in $\mathbb{Z}^k$ such that $\sum T_1 = \sum T_2 = \ldots = \sum T_n$ and all vectors contained in all multisets $T_1, \ldots, T_n$ have*

ℓ_∞-norm bounded by Δ. Then there are non-empty multisets $S_1 \subseteq T_1, \ldots, S_n \subseteq T_n$, each of size at most $2^{O(k\Delta)^k}$, such that $\sum S_1 = \sum S_2 = \ldots = \sum S_n$.

In the context of the proof of Lemma 2.5, we apply Lemma 2.6 to the family of all multisets T that consist of columns of D and satisfy $\sum T = \mathbf{b}$. By encoding multiplicities, such multisets correspond to vectors $\mathbf{v} \in \mathbb{Z}^k_{\geqslant 0}$ satisfying $D\mathbf{v} = \mathbf{b}$. (We hide here some technicalities regarding the fact that this family is infinite.) By Lemma 2.6, from each such multiset T, we can extract a submultiset S of bounded size such that all the submultisets S sum up to the same vector $\mathbf{b}'$. Denoting $\mathbf{b}'' = \mathbf{b} - \mathbf{b}'$, this means that every vector $\mathbf{v} \in \mathbb{Z}^k_{\geqslant 0}$ satisfying $D\mathbf{v} = \mathbf{b}$ can be decomposed as $\mathbf{v} = \mathbf{v}' + \mathbf{v}''$ with $\mathbf{v}', \mathbf{v}'' \in \mathbb{Z}^k_{\geqslant 0}$ so that $D\mathbf{v}' = \mathbf{b}'$ and $D\mathbf{v}'' = \mathbf{b}''$. Namely, $\mathbf{v}'$ corresponds to the vectors contained in S and $\mathbf{v}''$ corresponds to the vectors contained in $T - S$, where T is the multiset corresponding to $\mathbf{v}$.

There is an issue in the above reasoning: we do not obtain the property $\mathbf{b}', \mathbf{b}'' \sqsubseteq \mathbf{b}$, which will be important in later applications of Lemma 2.5. To bridge this difficulty, we apply the argument above exhaustively to decompose $\mathbf{b}$ as $\mathbf{b}_1 + \ldots + \mathbf{b}_m$, for some integer m, so that every vector $\mathbf{b}_i$ has the ℓ_∞-norm bounded by $2^{O(k\Delta)^k}$ and every vector $\mathbf{v} \in \mathbb{Z}^k_{\geqslant 0}$ satisfying $D\mathbf{v} = \mathbf{b}$ can be decomposed as $\mathbf{v} = \mathbf{v}_1 + \ldots + \mathbf{v}_m$ where $\mathbf{v}_i \in \mathbb{Z}^k_{\geqslant 0}$ satisfies $D\mathbf{v}_i = \mathbf{b}_i$. Then, we treat vectors $\mathbf{b}_1, \ldots, \mathbf{b}_m$ with the following lemma.

LEMMA 2.7 (see Lemma 5.6). *Let $\mathbf{u}_1, \ldots, \mathbf{u}_m$ be vectors in $\mathbb{Z}^k$ of ℓ_∞-norm bounded by Ξ, and let $\mathbf{b} = \sum_{i=1}^m \mathbf{u}_i$. Then the vectors $\mathbf{u}_1, \ldots, \mathbf{u}_m$ can be grouped into non-empty groups $U_1, \ldots, U_\ell$, each of size at most $O(\Delta)^{2^{k-1}}$, so that $\sum U_i \sqsubseteq \mathbf{b}$ for all $i = 1, \ldots, \ell$.*

More precisely, Lemma 2.7 allows us to group vectors $\mathbf{b}_1, \ldots, \mathbf{b}_m$ into groups of bounded size so that the sum within each group is sign-compatible with $\mathbf{b}$. Assuming $\|\mathbf{b}\|_\infty$ is large enough, there will be at least two groups. Then, any non-trivial partition of the groups translates into a suitable decomposition $\mathbf{b} = \mathbf{b}' + \mathbf{b}''$ with $\mathbf{b}', \mathbf{b}'' \sqsubseteq \mathbf{b}$.

The proof of Lemma 2.7 is by induction on k and uses arguments similar to standard proofs of Steinitz Lemma. This concludes a sketch of the proof of Lemma 2.5.

Once Lemma 2.5 is established, it is natural to use it iteratively: break $\mathbf{b}$ into $\mathbf{b}', \mathbf{b}''$, then break $\mathbf{b}'$ into two even smaller vectors, and so on. By applying the argument exhaustively, eventually we obtain a collection of vectors $\mathbf{b}_1, \ldots, \mathbf{b}_m \sqsubseteq \mathbf{b}$ such that $\mathbf{b} = \mathbf{b}_1 + \ldots + \mathbf{b}_m$, $\|\mathbf{b}_i\|_\infty \leqslant 2^{(k\Delta)^{O(k)}}$ for all $i \in \{1, \ldots, m\}$, and every $\mathbf{v} \in \mathbb{Z}^k_{\geqslant 0}$ satisfying $D\mathbf{v} = \mathbf{b}$ can be decomposed as $\mathbf{v} = \mathbf{v}_1 + \ldots + \mathbf{v}_m$ with $\mathbf{v}_i \in \mathbb{Z}^k_{\geqslant 0}$ and $D\mathbf{v}_i = \mathbf{b}_i$ for all $i \in \{1, \ldots, m\}$. We call such a collection a *faithful decompostion* of $\mathbf{b}$ of *order* $2^{(k\Delta)^{O(k)}}$.

There is an important technical caveat here. Observe that the size m of a faithful decomposition of a right-hand side $\mathbf{b}$ can be as large as $\Omega(\|\mathbf{b}\|_1)$, which is *exponential* in the bitsize of the program P. So we cannot hope to compute a faithful decomposition explicitly within the target time complexity. However, observe that all vectors $\mathbf{b}_i$ in a faithful decomposition $\mathcal{B}$ are

bounded in ℓ_∞-norm by $\Xi := 2^{(k\Delta)^{O(k)}}$, and there are only at most $(2\Xi+1)^k$ different such vectors. Therefore, $\mathcal{B}$ can be encoded by storing, for each vector $\mathbf{b}'$ present in $\mathcal{B}$, the multiplicity of $\mathbf{b}'$ in $\mathcal{B}$. Thus, describing $\mathcal{B}$ takes $2^{(k\Delta)^{O(k)}} \cdot \log \|\mathbf{b}\|_\infty$ bits.

With this encoding scheme in mind, we show that a faithful decomposition $\mathcal{B}$ of a given vector $\mathbf{b}$ of order at most Ξ can be computed in fixed-parameter time $f(\Delta, k) \cdot (\log \|\mathbf{b}\|_\infty)^{O(1)}$, for a computable function f. For this, we show that one can extract parts of the decomposition in "larger chunks", at each step reducing the ℓ_1-norm of the decomposed vector by a constant fraction; this gives a total number of steps logarithmic in $\|\mathbf{b}\|_1$. In each step, to extract the next large chunk of the decomposition, we use the fixed-parameter algorithm for optimization problems definable in Presburger arithmetic, due to Koutecký and Talmon [41]. We remark that in our context, this tool could be also replaced by the fixed-parameter algorithm of Eisenbrand and Shmonin [21] for ∀∃ integer programming.

Reduction to (mixed) integer programming with few variables. With faithful decompositions understood, we can compute, for every right-hand side $\mathbf{b}_i$ part of P, a faithful decomposition $\{\mathbf{b}_i^1, \ldots, \mathbf{b}_i^{m_i}\}$ of $\mathbf{b}_i$. This allows us to construct an equivalent (in terms of feasibility and optimization) n-fold program P' by replacing each brick $D\mathbf{y}_i = \mathbf{b}_i$ with bricks $D\mathbf{y}_i^j = \mathbf{b}_i^j$ for $j \in \{1, \ldots, m_i\}$. Thus, the program P' has an exponential number of bricks, but can be computed and described concisely: all right-hand sides are bounded in the ℓ_∞-norm by at most Ξ, so for every potential right-hand side $\mathbf{b}$, we just write the multiplicity in which $\mathbf{b}$ appears in P'. We remark that such *high-multiplicity encoding* of n-fold integer programs has already been studied by Knop et al. [34].

For convenience, let $\mathsf{RHS} := \{-\Xi, \ldots, \Xi\}^k$ be the set of all possible right-hand sides, and for $\mathbf{b} \in \mathsf{RHS}$, by $\mathsf{count}[\mathbf{b}]$ we denote the multiplicity of $\mathbf{b}$ in P'.

It is now important to better understand the set of solutions to a single brick $D\mathbf{y} = \mathbf{b}$ present in P'. Here comes a key insight stemming from the theory of Graver bases: as (essentially) proved by Pottier [45], every solution $\mathbf{w} \in \mathbb{Z}^k_{\geqslant 0}$ to $D\mathbf{w} = \mathbf{b}$ can be decomposed as $\mathbf{w} = \widehat{\mathbf{w}} + \mathbf{g}_1 + \ldots + \mathbf{g}_\ell$, where

- $\widehat{\mathbf{w}} \in \mathbb{Z}^k_{\geqslant 0}$ is a *base solution* that also satisfies $D\widehat{\mathbf{w}} = \mathbf{b}$, but $\|\widehat{\mathbf{w}}\|_\infty$ is bounded by a function of Δ and $\|\mathbf{b}\|_\infty$, and
- $\mathbf{g}_1, \ldots, \mathbf{g}_\ell \in \mathbb{Z}^k_{\geqslant 0}$ are elements of the Graver basis of D.

Here, the *Graver basis* of D consists of all conformally-minimal non-zero vectors $\mathbf{g}$ satisfying $D\mathbf{g} = \mathbf{0}$. In particular, it is known that the Graver basis is always finite and consists of vectors of ℓ_∞ norm bounded by $(2k\Delta + 1)^k$ [19]. The decomposition explained above will be called a *Graver decomposition* of $\mathbf{w}$.

For $\mathbf{b} \in \mathsf{RHS}$, let $\mathsf{Base}[\mathbf{b}]$ be the set of all possible base solutions $\widehat{\mathbf{w}}$ to $D\mathbf{y} = \mathbf{b}$. As $\|\mathbf{b}\|_\infty \leqslant \Xi$ and Ξ is bounded by a function of the parameters under consideration, it follows that $\mathsf{Base}[\mathbf{b}]$ consists only of vectors of bounded ℓ_∞-norms, and therefore it can be efficiently constructed.

Having this understanding, we can write an integer program M with few variables that is equivalent to P'. The variables are as follows:

— For every $\mathbf{b} \in \mathsf{RHS}$ and $\widehat{\mathbf{w}} \in \mathsf{Base}[\mathbf{b}]$, we introduce a variable $\zeta^{\mathbf{b}}_{\widehat{\mathbf{w}}} \in \mathbb{Z}_{\geqslant 0}$ that signifies how many times in total $\widehat{\mathbf{w}}$ is used in the Graver decompositions of solutions to individual bricks.

— For every nonnegative vector $\mathbf{g}$ in the Graver basis of D, we introduce a variable $\delta_{\mathbf{g}} \in \mathbb{Z}_{\geqslant 0}$ signifying how many times in total $\mathbf{g}$ appears in the Graver decompositions of solutions to individual bricks.

Note that since program P' is uniform, the guessed base solutions and elements of the Graver basis can be assigned to *any* brick with the same effect on the linking constraints of P'. Hence, it suffices to verify the cardinalities and the total effect on the linking constrains of P', yielding the following constraints of M:

— the translated linking constraints: $\sum_{\mathbf{b}\in\mathsf{RHS}} \sum_{\widehat{\mathbf{w}}\in\mathsf{Base}[\mathbf{b}]} \zeta^{\mathbf{b}}_{\widehat{\mathbf{w}}} \cdot C\widehat{\mathbf{w}} + \sum_{\mathbf{g}\in\mathsf{Graver}(D),\mathbf{g}\geqslant 0} \delta_{\mathbf{g}} \cdot C\mathbf{g} = \mathbf{a}$.

— for every $\mathbf{b} \in \mathsf{RHS}$, the cardinality constraint $\sum_{\widehat{\mathbf{w}}\in\mathsf{Base}[\mathbf{b}]} \zeta^{\mathbf{b}}_{\widehat{\mathbf{w}}} = \mathsf{count}[\mathbf{b}]$.

Noting that the number of variables of M is bounded in terms of the parameters, we may apply any fixed-parameter algorithm for integer programming parameterized by the number of variables, for instance that of Kannan [29], to solve M. This concludes the description of the algorithm for the feasibility problem.

In the case of the optimization problem, there is an issue that the optimization vectors $\mathbf{c}_i$ may differ between different bricks, and there may be as many as n different such vectors. While the Graver basis elements can be always greedily assigned to bricks in which their contribution to the optimization goal is the smallest, this is not so easy for the base solutions, as every brick may accommodate only one base solution. We may enrich M by suitable assignment variables $\omega^{\mathbf{b},i}_{\widehat{\mathbf{w}}}$ to express how many base solutions of each type are assigned to bricks with different optimization vectors; but this yields as many as $\Omega(n)$ additional variables. Fortunately, we observe that in the enriched program M, if one fixes any integral valuation of variables $\zeta^{\mathbf{b}}_{\widehat{\mathbf{w}}}$ and $\delta_{\mathbf{g}}$, the remaining problem on variables $\omega^{\mathbf{b},i}_{\widehat{\mathbf{w}}}$ corresponds to a flow problem, and hence its constraint matrix is totally unimodular. Thus, we may solve M as a mixed integer program where variables $\omega^{\mathbf{b},i}_{\widehat{\mathbf{w}}}$ are allowed to be fractional. The number of integral variables is bounded in terms of parameters, so we may apply the fixed-parameter algorithm for mixed integer programming of Lenstra [44].

3. Preliminaries

We write $\mathbb{Z}_{\geqslant 0}$ and $\mathbb{R}_{\geqslant 0}$ for the sets of nonnegative integers and nonnegative reals, respectively.

Linear algebra. In the technical part of this paper we choose to use a somewhat unorthodox notation for linear algebra, which we introduce now. The correspondence between this notation and the standard notation used in the previous sections is straightforward.

A *tuple of variables* is just a finite set of variable names. We use the convention that tuples of variables are denoted $\mathbf{x}, \mathbf{y}, \mathbf{z}$, etc., while individual variable names are x, y, z, etc. For a set A and a tuple of variables $\mathbf{x}$, an $\mathbf{x}$-*vector* over A is a function $\mathbf{u}\colon \mathbf{x} \to A$. The value of $\mathbf{u}$ on a variable $x \in \mathbf{x}$ will be denoted by $\mathbf{u}[x]$, and we call it the x-*entry* of $\mathbf{u}$. Vectors will be typically denoted by $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{u}, \mathbf{v}, \mathbf{w}$ and so on. We write $A^{\mathbf{x}}$ for the set of all $\mathbf{x}$-vectors over A. We can also apply arithmetics to vectors coordinate-wise in the standard manner, whenever the set A is endowed with a structure of a ring. When $A = \mathbb{R}$, as usual we write $\|\mathbf{u}\|_\infty := \max_{x\in\mathbf{x}} |\mathbf{u}[x]|$ and $\|\mathbf{u}\|_1 := \sum_{x\in\mathbf{x}} |\mathbf{u}[x]|$.

For a ring R (typically $\mathbb{Z}$ or $\mathbb{R}$) and tuples of variables $\mathbf{x}$ and $\mathbf{y}$, an $\mathbf{x}\times\mathbf{y}$-*matrix* is an $\mathbf{x}$-vector of $\mathbf{y}$-vectors, that is, an element of $(R^{\mathbf{y}})^{\mathbf{x}}$; we denote the latter set as $R^{\mathbf{x}\times\mathbf{y}}$ for convenience. The reader should think of the individual $\mathbf{y}$-vectors comprised in a matrix as of the columns of the said matrix. We use capital letters A, B, C, etc. for matrices. For $x \in \mathbf{x}$ and $y \in \mathbf{y}$, we write $A[x, y] := (A[x])[y]$, that is, $A[x, y]$ is the y-entry of the column of A corresponding to variable x. As for vectors, by $\|A\|_\infty := \max_{x\in\mathbf{x}, y\in\mathbf{y}} |A[x, y]|$ we denote the maximum absolute value of an entry of A.

For a matrix $A \in R^{\mathbf{x}\times\mathbf{y}}$ and a vector $\mathbf{u} \in R^{\mathbf{x}}$, $A\mathbf{u}$ is the vector $\mathbf{v} \in R^{\mathbf{y}}$ satisfying

$$\mathbf{v}[y] = \sum_{x\in\mathbf{x}} A[x, y] \cdot \mathbf{u}[x] \qquad \text{for all } y \in \mathbf{y}.$$

For vectors $\mathbf{u}, \mathbf{v} \in R^{\mathbf{x}}$, the *inner product* of $\mathbf{u}$ and $\mathbf{v}$ is

$$\langle \mathbf{u}, \mathbf{v} \rangle := \sum_{x\in\mathbf{x}} \mathbf{u}[x] \cdot \mathbf{v}[x].$$

All-zero and all-one vectors are denoted by $\mathbf{0}$ and $\mathbf{1}$, respectively, and their domains will be always clear from the context.

Conformal order. Let $\mathbf{x}$ be a tuple of variables. Vectors $\mathbf{u}, \mathbf{v} \in \mathbb{Z}^{\mathbf{x}}$ are *sign-compatible* if $\mathbf{u}[x] \cdot \mathbf{v}[x] \geqslant 0$ for all $x \in \mathbf{x}$; that is, on every coordinate $\mathbf{u}$ and $\mathbf{v}$ must have the same sign, where 0 is assumed to be compatible with both signs. The *conformal order* $\sqsubseteq$ on vectors in $\mathbb{Z}^{\mathbf{x}}$ is defined as follows: For two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{Z}^{\mathbf{x}}$, we have $\mathbf{u} \sqsubseteq \mathbf{v}$ if $\mathbf{u}$ and $\mathbf{v}$ are sign-compatible and

$$|\mathbf{u}[x]| \leqslant |\mathbf{v}[x]| \qquad \text{for all } x \in \mathbf{x}.$$

Note that thus, $\sqsubseteq$ is a partial order on $\mathbb{Z}^{\mathbf{x}}$.

Collections. A *collection* over a set A is a function $C\colon I \to A$, where I is the *index set* of C. We often use notation $C = \langle c_i : i \in I \rangle$ to enumerate the elements of a collection with the elements of its index set, where formally c_i is the value of C on i. Note that syntactically, a collection

with index set I is just an I-vector, but we will use collections and vectors in different ways. Typically, when speaking about collections we are not really interested in the index set, and the reader can always assume it to be a prefix of natural numbers. However, in notation it will be convenient to consider various index sets.

Standard set theory notation can be used in the context of collections in the natural manner, by applying it to the index sets. In particular, a *subcollection* of a collection $C = \langle c_i : i \in I \rangle$ is any collection $C' = \langle c_i : i \in I' \rangle$ for $I' \subseteq I$.

Block-structured integer programming. We now introduce the two variants of block-structured integer programming problems that we are interested in: two-stage stochastic integer programming and n-fold integer programming.

A *two-stage stochastic integer program* consists of three collections:

— $\langle A_i : i \in I \rangle$ is a collection of matrices in $\mathbb{Z}^{\mathbf{x}\times\mathbf{t}}$;
— $\langle D_i : i \in I \rangle$ is a collection of matrices in $\mathbb{Z}^{\mathbf{y}\times\mathbf{t}}$; and
— $\langle \mathbf{b}_i : i \in I \rangle$ is a collection of vectors in $\mathbb{Z}^{\mathbf{t}}$.

Here, $\mathbf{x}, \mathbf{y}, \mathbf{t}$ are tuples of variables and I is the index set of the program. A *solution* to such a program consists of a vector $\mathbf{u} \in \mathbb{Z}^{\mathbf{x}}_{\geqslant 0}$ and vectors $\langle \mathbf{v}_i : i \in I \rangle$ in $\mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ such that

$$A_i\mathbf{u} + D_i\mathbf{v}_i = \mathbf{b}_i \quad \text{for all } i \in I.$$

In the Two-Stage Stochastic ILP Feasibility problem, we are given a two-stage stochastic integer program in the form described above, and the task is to decide whether this program has a solution.

An *n-fold integer program* consists of the following components:

— $\langle C_i : i \in I \rangle$ is a collection of matrices in $\mathbb{Z}^{\mathbf{y}\times\mathbf{s}}$;
— $\langle D_i : i \in I \rangle$ is a collection of matrices in $\mathbb{Z}^{\mathbf{y}\times\mathbf{t}}$;
— $\mathbf{a}$ is a vector in $\mathbb{Z}^{\mathbf{s}}$; and
— $\langle \mathbf{b}_i : i \in I \rangle$ is a collection of vectors in $\mathbb{Z}^{\mathbf{t}}$.

Again, $\mathbf{y}, \mathbf{s}, \mathbf{t}$ are tuples of variables and I is the index set of the program. A *uniform* program is one where $C_i = C_j$ for all $i, j \in I$. A *solution* to the program described above consists of vectors $\langle \mathbf{v}_i : i \in I \rangle$ in $\mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ such that

$$\sum_{i\in I} C_i\mathbf{v}_i = \mathbf{a} \qquad \text{and} \qquad D_i\mathbf{v}_i = \mathbf{b}_i \quad \text{for all } i \in I.$$

The n-fold ILP Feasibility problem asks for the existence of a solution to a given n-fold integer program, while in the n-fold ILP Optimization problem, the task is to minimize $\sum_{i\in I}\langle \mathbf{c}_i, \mathbf{v}_i \rangle$ among the solutions, for additionally given vectors $\langle \mathbf{c}_i : i \in I \rangle$ in $\mathbb{Z}^{\mathbf{y}}$. A prefix Uniform may be added to the problem name to specify that we speak about uniform programs. (Note that only the matrices C_i have to be equal, and not necessarily the optimization goal vectors $\mathbf{c}_i$.)

For an integer program P, be it a two-stage stochastic or an n-fold program, by $\|P\|$ we denote the total bitsize of the encoding of P, where all numbers are encoded in binary.

4. Two-stage stochastic integer programming

Our main result for two-stage stochastic integer programs is captured in the following statement, which is Theorem 1.1 with adjusted notation.

THEOREM 4.1. *An instance $P = \langle A_i, D_i, \mathbf{b}_i : i \in I \rangle$ of Two-Stage Stochastic ILP Feasability can be solved in time $f(\Delta, |\mathbf{x}|, |\mathbf{t}|) \cdot \|P\|$ for some computable function f, where $\Delta = \max_{i \in I} \|D_i\|_\infty$.*

Note that importantly, the absolute values of the entries of matrices A_i are *not* assumed to be bounded in terms of the considered parameters. Also, in Theorem 4.1 the parameters do not include $|\mathbf{y}|$, the number of columns of each block D_i. This is because by removing equal columns in those blocks (which does not affect the feasibility of the program), we may always assume $|\mathbf{y}| \leqslant (2\Delta + 1)^{|\mathbf{t}|}$.

The proof of Theorem 4.1 spans the entirety of this section.

4.1 Cones

In our proof we will rely on the geometry of polyhedral and integer cones. For this, we introduce the following notation. Let $\mathbf{t}$ be a tuple of variables and $\mathcal{D} \subseteq \mathbb{Z}^{\mathbf{t}}$ be a finite set of vectors. For $S \in \{\mathbb{R}, \mathbb{R}_{\geqslant 0}, \mathbb{Z}, \mathbb{Z}_{\geqslant 0}\}$, we define the *$S$-span of $\mathcal{D}$* as

$$\mathsf{span}^S(\mathcal{D}) := \left\{ \sum_{\mathbf{d} \in \mathcal{D}} \lambda_{\mathbf{d}} \cdot \mathbf{d} : \ \langle \lambda_{\mathbf{d}} : \mathbf{d} \in \mathcal{D} \rangle \text{ is a collection of elements of } S \right\} \subseteq \mathbb{R}^{\mathbf{t}}.$$

In other words, $\mathsf{span}^S(\mathcal{D})$ are all vectors that can be expressed as a linear combination of vectors in $\mathcal{D}$ with coefficients belonging to S. For clarity, we write

$$\mathsf{span}(\mathcal{D}) := \mathsf{span}^{\mathbb{R}}(\mathcal{D}), \qquad \mathsf{cone}(\mathcal{D}) := \mathsf{span}^{\mathbb{R}_{\geqslant 0}}(\mathcal{D}),$$
$$\mathsf{lattice}(\mathcal{D}) := \mathsf{span}^{\mathbb{Z}}(\mathcal{D}), \qquad \mathsf{intCone}(\mathcal{D}) := \mathsf{span}^{\mathbb{Z}_{\geqslant 0}}(\mathcal{D}).$$

Thus, $\mathsf{span}(\mathcal{D})$ is just the usual linear space spanned of $\mathcal{D}$. Next, $\mathsf{cone}(\mathcal{D})$ is the standard *polyhedral cone*: it consists of all vectors expressible as nonnegative linear combinations of vectors in $\mathcal{D}$. Similarly, $\mathsf{intCone}(\mathcal{D})$ is the *integer cone*, where we restrict attention to nonnegative integer combinations of vectors in $\mathcal{D}$. Finally, $\mathsf{lattice}(\mathcal{D})$ is the *integer lattice* consisting of all vectors that can be reached from $\mathbf{0}$ by adding and subtracting vectors of $\mathcal{D}$.

We remark that while the reader may think of $\mathcal{D}$ as of a matrix whose columns are the vectors of $\mathcal{D}$, ordered arbitrarily, in the notation we will consistently treat $\mathcal{D}$ as a set of vectors. Consequently, $|\mathcal{D}|$ denotes the number of vectors in $\mathcal{D}$ (or, the number of columns of $\mathcal{D}$ treated as a matrix), and the same also applies to all subsets of $\mathcal{D}$.

4.1.1 Dual representation of cones

We will use the classic result of Weyl about representing polyhedral cones as intersections of half-spaces.

THEOREM 4.2 (Weyl, [49]). *For every set of vectors $\mathcal{D} \subseteq \mathbb{Z}^{\mathbf{t}}$ there exists a finite set of vectors $\mathcal{F} \subseteq \mathbb{Z}^{\mathbf{t}}$ such that*

$$\mathsf{cone}(\mathcal{D}) = \{\mathbf{v} \in \mathbb{R}^{\mathbf{t}} \mid \langle \mathbf{f}, \mathbf{v} \rangle \geqslant 0 \text{ for all } \mathbf{f} \in \mathcal{F}\}.$$

Moreover, such a set $\mathcal{F}$ can be computed given $\mathcal{D}$.

A set $\mathcal{F}$ satisfying the outcome of Theorem 4.2 will be called a *dual representation* of $\mathsf{cone}(\mathcal{D})$. Here is a simple observation about how elements of the dual representation relate to the elements of $\mathcal{D}$.

OBSERVATION 4.3. *Suppose $\mathcal{F}$ is a dual representation of $\mathsf{cone}(\mathcal{D})$ for a set of vectors $\mathcal{D} \subseteq \mathbb{Z}^{\mathbf{t}}$. Then*

$$\langle \mathbf{f}, \mathbf{d} \rangle \geqslant 0 \quad \text{for all } \mathbf{f} \in \mathcal{F} \text{ and } \mathbf{d} \in \mathcal{D}.$$

PROOF. Clearly $\mathbf{d} \in \mathsf{cone}(\mathcal{D})$, so $\langle \mathbf{f}, \mathbf{d} \rangle \geqslant 0$ because $\mathcal{F}$ is a dual representation of $\mathsf{cone}(\mathcal{D})$. ■

For a subset $\mathcal{G} \subseteq \mathcal{F}$, we define

$$\mathcal{D}_{\mathcal{G}} \coloneqq \{\mathbf{d} \in \mathcal{D} \mid \langle \mathbf{g}, \mathbf{d} \rangle = 0 \text{ for all } \mathbf{g} \in \mathcal{G}\}.$$

In other words, $\mathcal{D}_{\mathcal{G}}$ consists of those vectors of $\mathcal{D}$ that are orthogonal to all vectors in $\mathcal{G}$. The next lemma expresses the following intuition: the facet of $\mathsf{cone}(\mathcal{D})$ corresponding to $\mathcal{G}$ is spanned by the vectors of $\mathcal{D}_{\mathcal{G}}$.

LEMMA 4.4. *Suppose $\mathcal{F}$ is a dual representation of $\mathsf{cone}(\mathcal{D})$ for a set of vectors $\mathcal{D} \subseteq \mathbb{Z}^{\mathbf{t}}$. Then for every subset $\mathcal{G} \subseteq \mathcal{F}$ we have*

$$\mathsf{cone}(\mathcal{D}_{\mathcal{G}}) = \mathsf{cone}(\mathcal{D}) \cap \{\mathbf{v} \in \mathbb{R}^{\mathbf{t}} \mid \langle \mathbf{g}, \mathbf{v} \rangle = 0 \text{ for all } \mathbf{g} \in \mathcal{G}\}.$$

PROOF. Note that $\mathcal{D}_{\mathcal{G}} \subseteq \mathcal{D}$ entails $\mathsf{cone}(\mathcal{D}_{\mathcal{G}}) \subseteq \mathsf{cone}(\mathcal{D})$. Further, since $\langle \mathbf{g}, \mathbf{d} \rangle = 0$ for all $\mathbf{g} \in \mathcal{G}$ and all $\mathbf{d} \in \mathcal{D}_{\mathcal{G}}$, the same also holds for all linear combinations of vectors in $\mathcal{D}_{\mathcal{G}}$, implying that $\langle \mathbf{g}, \mathbf{v} \rangle = 0$ for all $\mathbf{g} \in \mathcal{G}$ and $\mathbf{v} \in \mathsf{cone}(\mathcal{D}_{\mathcal{G}})$. This proves inclusion $\subseteq$.

For the converse inclusion, consider any $\mathbf{v} \in \mathsf{cone}(\mathcal{D})$ such that $\langle \mathbf{g}, \mathbf{v} \rangle = 0$ for all $\mathbf{g} \in \mathcal{G}$. As $\mathbf{v} \in \mathsf{cone}(\mathcal{D})$, there is a collection of nonnegative reals $\langle \lambda_{\mathbf{d}} \colon \mathbf{d} \in \mathcal{D} \rangle$ such that

$$\mathbf{v} = \sum_{\mathbf{d} \in \mathcal{D}} \lambda_{\mathbf{d}} \cdot \mathbf{d}.$$

Consider any $\mathbf{e} \in \mathcal{D} - \mathcal{D}_{\mathcal{G}}$. By the definition of $\mathcal{D}_{\mathcal{G}}$ and Observation 4.3, there exists $\mathbf{g} \in \mathcal{G}$ such that $\langle \mathbf{g}, \mathbf{e} \rangle > 0$. As all coefficients $\langle \lambda_{\mathbf{d}} \colon \mathbf{d} \in \mathcal{D} \rangle$ are nonnegative, by Observation 4.3 we have

$$0 = \langle \mathbf{g}, \mathbf{v} \rangle = \sum_{\mathbf{d} \in \mathcal{D}} \lambda_{\mathbf{d}} \cdot \langle \mathbf{g}, \mathbf{d} \rangle \geqslant \lambda_{\mathbf{e}} \cdot \langle \mathbf{g}, \mathbf{e} \rangle.$$

As $\langle \mathbf{g}, \mathbf{e} \rangle > 0$, we necessarily have $\lambda_{\mathbf{e}} = 0$; and this holds for every $\mathbf{e} \in \mathcal{D} - \mathcal{D}_{\mathcal{G}}$. So in fact $\mathbf{v}$ is a nonnegative linear combination of vectors in $\mathcal{D}_{\mathcal{G}}$, or equivalently $\mathbf{v} \in \mathsf{cone}(\mathcal{D}_{\mathcal{G}})$. This proves inclusion $\supseteq$. ■

4.1.2 Membership in integer cones and in integer lattices

First, we need a statement that membership in an integer lattice can be witnessed by a vector with small entries. The following lemma follows from standard bounds given by Pottier [45, Corollary 4] (c.f. Lemma 5.3).

LEMMA 4.5. *For every set of vectors $\mathcal{D} \subseteq \mathbb{Z}^{\mathbf{t}}$ and a vector $\mathbf{v} \in \mathsf{lattice}(\mathcal{D})$, there exist integer coefficients $\langle \lambda_{\mathbf{d}} \colon \mathbf{d} \in \mathcal{D} \rangle$ such that*

$$\mathbf{v} = \sum_{\mathbf{d} \in \mathcal{D}} \lambda_{\mathbf{d}} \cdot \mathbf{d} \qquad \text{and} \qquad \sum_{\mathbf{d} \in \mathcal{D}} |\lambda_{\mathbf{d}}| \leqslant \left(2 + \max_{\mathbf{d} \in \mathcal{D}} \|\mathbf{d}\|_\infty + \|\mathbf{v}\|_\infty \right)^{2|\mathbf{t}|}.$$

We now proceed to a technical statement that is crucial in our approach. Intuitively, it says that if a vector lies "deep" in the polyhedral cone, then its membership in the integer cone is equivalent to the membership in the integer lattice.

LEMMA 4.6 (Deep-in-the-Cone Lemma). *Let $\mathcal{D} \subseteq \mathbb{Z}^{\mathbf{t}}$ be a set of vectors and $\mathcal{F} \subseteq \mathbb{Z}^{\mathbf{t}}$ be a dual representation of $\mathsf{cone}(\mathcal{D})$. Then there is a positive integer M, computable from $\mathcal{D}$ and $\mathcal{F}$, such that for every $\mathcal{G} \subseteq \mathcal{F}$ and every $\mathbf{v} \in \mathsf{cone}(\mathcal{D}_{\mathcal{G}})$ satisfying $\langle \mathbf{f}, \mathbf{v} \rangle \geqslant M$ for all $\mathbf{f} \in \mathcal{F} - \mathcal{G}$, we have*

$$\mathbf{v} \in \mathsf{lattice}(\mathcal{D}_{\mathcal{G}}) \quad \textit{if and only if} \quad \mathbf{v} \in \mathsf{intCone}(\mathcal{D}_{\mathcal{G}}).$$

PROOF. We set

$$M \coloneqq L \cdot |\mathcal{D}| \cdot \max_{\mathbf{f} \in \mathcal{F}} \|\mathbf{f}\|_1 \cdot \max_{\mathbf{d} \in \mathcal{D}} \|\mathbf{d}\|_\infty, \qquad \text{where} \qquad L \coloneqq \left(2 + (|\mathcal{D}| + 1) \cdot \max_{\mathbf{d} \in \mathcal{D}} \|\mathbf{d}\|_\infty \right)^{2|\mathbf{t}|}$$

is the bound provided by Lemma 4.5 for vectors of ℓ_∞-norm bounded by $|\mathcal{D}| \cdot \max_{\mathbf{d} \in \mathcal{D}} \|\mathbf{d}\|_\infty$.

As $\mathsf{intCone}(\mathcal{D}_{\mathcal{G}}) \subseteq \mathsf{lattice}(\mathcal{D}_{\mathcal{G}})$, it suffices to prove the following: if $\mathbf{v} \in \mathsf{cone}(\mathcal{D}_{\mathcal{G}}) \cap \mathsf{lattice}(\mathcal{D}_{\mathcal{G}})$ satisfies $\langle \mathbf{f}, \mathbf{v} \rangle \geqslant M$ for all $\mathbf{f} \in \mathcal{F} - \mathcal{G}$, then in fact $\mathbf{v} \in \mathsf{intCone}(\mathcal{D}_{\mathcal{G}})$. Let

$$\mathbf{w} \coloneqq \sum_{\mathbf{d} \in \mathcal{D}_{\mathcal{G}}} L \cdot \mathbf{d}.$$

As both $\mathbf{v}$ and $\mathbf{w}$ are linear combinations of vectors in $\mathcal{D}_{\mathcal{G}}$, we have

$$\langle \mathbf{g}, \mathbf{v} - \mathbf{w} \rangle = 0 \qquad \text{for all } \mathbf{g} \in \mathcal{G}.$$

Further, for each $\mathbf{f} \in \mathcal{F} - \mathcal{G}$ we have

$$\langle \mathbf{f}, \mathbf{v} - \mathbf{w} \rangle = \langle \mathbf{f}, \mathbf{v} \rangle - \langle \mathbf{f}, \mathbf{w} \rangle \geqslant M - L \cdot \sum_{\mathbf{d} \in \mathcal{D}} |\langle \mathbf{f}, \mathbf{d} \rangle| \geqslant M - L \cdot |\mathcal{D}| \cdot \max_{\mathbf{f} \in \mathcal{F}} \|\mathbf{f}\|_1 \cdot \max_{\mathbf{d} \in \mathcal{D}} \|\mathbf{d}\|_\infty = 0.$$

As $\mathcal{F}$ is a dual representation of $\mathsf{cone}(\mathcal{D})$, the two assertions above together with Lemma 4.4 imply that $\mathbf{v} - \mathbf{w} \in \mathsf{cone}(\mathcal{D}_\mathcal{G})$. Consequently, there is a collection of nonnegative reals $\langle \lambda_\mathbf{d} \colon \mathbf{d} \in \mathcal{D}_\mathcal{G} \rangle$ such that

$$\mathbf{v} - \mathbf{w} = \sum_{\mathbf{d} \in \mathcal{D}_\mathcal{G}} \lambda_\mathbf{d} \cdot \mathbf{d}.$$

Now, consider the vector

$$\mathbf{v}' := \mathbf{w} + \sum_{\mathbf{d} \in \mathcal{D}_\mathcal{G}} \lfloor \lambda_\mathbf{d} \rfloor \cdot \mathbf{d} = \sum_{\mathbf{d} \in \mathcal{D}_\mathcal{G}} (L + \lfloor \lambda_\mathbf{d} \rfloor) \cdot \mathbf{d}.$$

Clearly $\mathbf{v}' \in \mathsf{lattice}(\mathcal{D}_\mathcal{G})$. As $\mathbf{v} \in \mathsf{lattice}(\mathcal{D}_\mathcal{G})$ by assumption, we have $\mathbf{v} - \mathbf{v}' \in \mathsf{lattice}(\mathcal{D}_\mathcal{G})$ as well. Furthermore,

$$\|\mathbf{v} - \mathbf{v}'\|_\infty = \left\| \sum_{\mathbf{d} \in \mathcal{D}_\mathcal{G}} (\lambda_\mathbf{d} - \lfloor \lambda_\mathbf{d} \rfloor) \cdot \mathbf{d} \right\|_\infty \leqslant |\mathcal{D}_\mathcal{G}| \cdot \max_{\mathbf{d} \in \mathcal{D}_\mathcal{G}} \|\mathbf{d}\|_\infty \leqslant |\mathcal{D}| \cdot \max_{\mathbf{d} \in \mathcal{D}} \|\mathbf{d}\|_\infty.$$

By Lemma 4.5 we conclude that there exist integer coefficients $\langle \mu_\mathbf{d} \colon \mathbf{d} \in \mathcal{D}_\mathcal{G} \rangle$ such that

$$\mathbf{v} - \mathbf{v}' = \sum_{\mathbf{d} \in \mathcal{D}_\mathcal{G}} \mu_\mathbf{d} \cdot \mathbf{d} \qquad \text{and} \qquad \sum_{\mathbf{d} \in \mathcal{D}_\mathcal{G}} |\mu_\mathbf{d}| \leqslant L.$$

Hence,

$$\mathbf{v} = \mathbf{v}' + (\mathbf{v} - \mathbf{v}') = \sum_{\mathbf{d} \in \mathcal{D}_\mathcal{G}} (L + \lfloor \lambda_\mathbf{d} \rfloor + \mu_\mathbf{d}) \cdot \mathbf{d}.$$

It now remains to observe that each coefficient $L + \lfloor \lambda_\mathbf{d} \rfloor + \mu_\mathbf{d}$ is a nonnegative integer, because $\lambda_\mathbf{d} \geqslant 0$ and $|\mu_\mathbf{d}| \leqslant L$. This shows that $\mathbf{v} \in \mathsf{intCone}(\mathcal{D}_\mathcal{G})$, thereby completing the proof. ■

4.2 Regular lattices

As mentioned in Section 1, the key idea in our approach is to guess the remainders of the entries of the sought solution modulo some large integer. To describe this idea formally, we consider regular lattices defined as follows.

Let K be a positive integer. For a vector of residues $\mathbf{r} \in \{0, 1, \ldots, K-1\}^\mathbf{t}$, where $\mathbf{t}$ is a tuple of variables, we define the *regular lattice* $\Lambda^K_\mathbf{r}$:

$$\Lambda^K_\mathbf{r} := \{\mathbf{v} \in \mathbb{Z}^\mathbf{t} \mid \mathbf{v}(t) \equiv \mathbf{r}(t) \bmod K \text{ for all } t \in \mathbf{t}\}.$$

In other words, $\Lambda^K_\mathbf{r}$ comprises all integer vectors in which the remainders mod K on all coordinates are exactly as specified in $\mathbf{r}$. Note that regular lattices are affine rather than linear. In

particular, $\mathbf{0} \in \Lambda^K_{\mathbf{r}}$ if and only if $\mathbf{r} = \mathbf{0}$, and hence a regular lattice $\Lambda^K_{\mathbf{r}}$ is not the form lattice$(\mathcal{D})$ for a multiset of vectors $\mathcal{D}$, unless $\mathbf{r} = \mathbf{0}$.

We will need the following simple claim about fractionality of solutions to systems of equations.

LEMMA 4.7. *Let $\mathcal{D} \subseteq \mathbb{Z}^{\mathbf{t}}$ be a set of vectors, where $\mathbf{t}$ is a tuple of variables. Then there is a positive integer C, computable from $\mathcal{D}$, satisfying the following: for every $\mathbf{v} \in \mathsf{span}(\mathcal{D}) \cap \mathbb{Z}^{\mathbf{t}}$ there exist coefficients $\langle \lambda_{\mathbf{d}} \colon \mathbf{d} \in \mathcal{D} \rangle$ such that*

$$\mathbf{v} = \sum_{\mathbf{d} \in \mathcal{D}} \lambda_{\mathbf{d}} \cdot \mathbf{d} \qquad \text{and} \qquad C\lambda_{\mathbf{d}} \text{ is an integer for every } \mathbf{d} \in \mathcal{D}.$$

PROOF. Let $\mathbf{z}$ be a tuple of variables containing one variable $z_{\mathbf{d}}$ for each $\mathbf{d} \in \mathcal{D}$, and let D be the $\mathbf{z} \times \mathbf{t}$ matrix such that the column of D corresponding to $z_{\mathbf{d}} \in \mathbf{z}$ is $\mathbf{d}$. Thus, collections of coefficients $\langle \lambda_{\mathbf{d}} \colon \mathbf{d} \in \mathcal{D} \rangle$ as in the lemma statement correspond to solutions $\boldsymbol{\lambda} \in \mathbb{Z}^{\mathbf{z}}$ of the system of equations $D\mathbf{z} = \mathbf{v}$. By first restricting the rows of D to a linearly independent set of rows, and then adding some $\{0, 1\}$ row vectors together with zeroes on the right hand side, we can obtain a system of equations $\widetilde{D}\mathbf{z} = \widetilde{\mathbf{v}}$ such that every solution to $\widetilde{D}\mathbf{z} = \widetilde{\mathbf{v}}$ is also a solution to $D\mathbf{z} = \mathbf{v}$, and $\widetilde{D}$ is a non-singular square matrix. It now follows from Cramer rules that if $\boldsymbol{\lambda} \in \mathbb{Z}^{\mathbf{z}}$ is the unique solution to $\widetilde{D}\mathbf{z} = \widetilde{\mathbf{v}}$, then $\det \widetilde{D} \cdot \boldsymbol{\lambda}$ is an integer. Therefore, we may set $C := \det \widetilde{D}$, where $\widetilde{D}$ is any non-singular square matrix that can be obtained from D by first restricting it to a linearly independent set of rows, and then extending this set of rows to a row basis using $\{0, 1\}$ row vectors. Note that C is clearly computable from D. ■

We now use Lemma 4.7 to prove the following statement that relates regular lattices with integer lattices generated by sets of vectors.

LEMMA 4.8. *Let $\mathcal{D} \subseteq \mathbb{Z}^{\mathbf{t}}$ be a set of vectors, where $\mathbf{t}$ is a tuple of variables. Then there is a positive integer K, computable from $\mathcal{D}$, satisfying the following: for every positive integer K' divisible by K and every vector of residues $\mathbf{r} \in \{0, 1, \ldots, K' - 1\}^{\mathbf{t}}$,*

$$\mathsf{span}(\mathcal{D}) \cap \Lambda^{K'}_{\mathbf{r}} \subseteq \mathsf{lattice}(\mathcal{D}) \qquad \text{or} \qquad \mathsf{lattice}(\mathcal{D}) \cap \Lambda^{K'}_{\mathbf{r}} = \emptyset.$$

PROOF. We set $K := C$, where C is the integer provided by Lemma 4.7 for $\mathcal{D}$. Fix any positive integer K' divisible by K and $\mathbf{r} \in \{0, 1, \ldots, K' - 1\}^{\mathbf{t}}$. It suffices to prove that if there exists $\mathbf{u} \in \mathsf{lattice}(\mathcal{D}) \cap \Lambda^{K'}_{\mathbf{r}}$, then in fact $\mathsf{span}(\mathcal{D}) \cap \Lambda^{K'}_{\mathbf{r}} \subseteq \mathsf{lattice}(\mathcal{D})$. Consider any $\mathbf{v} \in \mathsf{span}(\mathcal{D}) \cap \Lambda^{K'}_{\mathbf{r}}$. Since $\mathbf{u}, \mathbf{v} \in \Lambda^{K'}_{\mathbf{r}}$, every entry of the vector $\mathbf{u} - \mathbf{v}$ is divisible by K', so also by K. It follows that

$$\mathbf{u} - \mathbf{v} = K \cdot \mathbf{w} \quad \text{for some } \mathbf{w} \in \mathbb{Z}^{\mathbf{t}}.$$

Since $\mathbf{u}, \mathbf{v} \in \mathsf{span}(\mathcal{D})$, we also have $\mathbf{w} \in \mathsf{span}(\mathcal{D})$. By Lemma 4.7, there exist coefficients $\langle \lambda_{\mathbf{d}} \colon \mathbf{d} \in \mathcal{D} \rangle$ such that

$$\mathbf{w} = \sum_{\mathbf{d} \in \mathcal{D}} \lambda_{\mathbf{d}} \cdot \mathbf{d} \qquad \text{and} \qquad K\lambda_{\mathbf{d}} \text{ is an integer for all } \mathbf{d} \in \mathcal{D}.$$

Therefore, we have

$$\mathbf{u} - \mathbf{v} = K \cdot \mathbf{w} = \sum_{\mathbf{d} \in \mathcal{D}} (K\lambda_{\mathbf{d}}) \cdot \mathbf{d},$$

where coefficients $K\lambda_{\mathbf{d}}$ are integral. So $\mathbf{u} - \mathbf{v} \in \mathsf{lattice}(\mathcal{D})$. As $\mathbf{u} \in \mathsf{lattice}(\mathcal{D})$ by assumption, we conclude that $\mathbf{v} \in \mathsf{lattice}(\mathcal{D})$ as well; this concludes the proof. ■

4.3 Reduction to polyhedral constraints

We proceed to the cornerstone of our approach: the theorem stated below, which was presented in Section 2 as Theorem 2.1. Intuitively, it says that under fixing residues modulo a large integer, membership in an integer cone is equivalent to membership in a carefully crafted polyhedron.

THEOREM 4.9 (Reduction to Polyhedral Constraints). *Let $\mathcal{D} \subseteq \mathbb{Z}^{\mathbf{t}}$ be a set of vectors, where $\mathbf{t}$ is a tuple of variables. Then there exists a positive integer B, computable from $\mathcal{D}$, satisfying the following: for every vector of residues $\mathbf{r} \in \{0, 1, \ldots, B-1\}^{\mathbf{t}}$, there is a finite set $\mathcal{Q} \subseteq \mathbb{Z}^{\mathbf{t}} \times \mathbb{Z}$ such that*

$$\Lambda^B_{\mathbf{r}} \cap \mathsf{intCone}(\mathcal{D}) = \Lambda^B_{\mathbf{r}} \cap \left\{\mathbf{v} \in \mathbb{R}^{\mathbf{t}} \mid \langle \mathbf{q}, \mathbf{v} \rangle \geqslant a \text{ for all } (\mathbf{q}, a) \in \mathcal{Q}\right\}.$$

Moreover, there is an algorithm that given $\mathcal{D}$ and $\mathbf{r}$, computes $\mathcal{Q}$ satisfying the above.

PROOF. Let $\mathcal{F}$ be a dual representation of $\mathsf{cone}(\mathcal{D})$, computed from $\mathcal{D}$ using Theorem 4.2. Further, let

- K be the least common multiple of all integers $K_{\mathcal{G}}$ obtained by applying Lemma 4.8 to $\mathcal{D}_{\mathcal{G}}$ for every $\mathcal{G} \subseteq \mathcal{F}$, and
- M be the integer obtained by applying Lemma 4.6 to $\mathcal{D}$ and $\mathcal{F}$.

We define

$$\widehat{M} := M + |\mathcal{D}| \cdot \max_{\mathbf{f} \in \mathcal{F}} \|\mathbf{f}\|_1 \cdot \max_{\mathbf{d} \in \mathcal{D}} \|\mathbf{d}\|_\infty,$$

and set

$$B_0 := K, \qquad B_i := \widehat{M} \cdot B_{i-1} \quad \text{for } i = 1, 2, \ldots, |\mathcal{F}|, \qquad \text{and} \qquad B := 2 \cdot B_{|\mathcal{F}|}.$$

We are left with constructing a suitable set $\mathcal{Q}$ for a given $\mathbf{r} \in \{0, 1, \ldots, B-1\}^{\mathbf{t}}$.

For convenience, denote

$$\mathcal{R} := \Lambda^B_{\mathbf{r}} \cap \mathsf{cone}(\mathcal{D}) \qquad \text{and} \qquad \mathcal{Z} := \Lambda^B_{\mathbf{r}} \cap \mathsf{intCone}(\mathcal{D}).$$

For every $\mathbf{f} \in \mathcal{F}$, let $p_{\mathbf{f}}$ be the unique integer in $\{0, 1, \ldots, B-1\}$ such that

$$p_{\mathbf{f}} \equiv \langle \mathbf{f}, \mathbf{r} \rangle \mod B.$$

Observe the following.

CLAIM 4.9.1. For every $\mathbf{f} \in \mathcal{F}$ and $\mathbf{v} \in \mathcal{R}$, we have

$$\langle \mathbf{f}, \mathbf{v} \rangle \in \{p_{\mathbf{f}}, p_{\mathbf{f}} + B, p_{\mathbf{f}} + 2B, \ldots\}.$$

Proof. As $\mathbf{v} \in \mathrm{cone}(\mathcal{D})$ and $\mathcal{F}$ is a dual representation of $\mathrm{cone}(\mathcal{D})$, we have $\langle \mathbf{f}, \mathbf{v} \rangle \geqslant 0$. As $\mathbf{v} \in \Lambda^B_{\mathbf{r}}$, we also have $\langle \mathbf{f}, \mathbf{v} \rangle \equiv \langle \mathbf{f}, \mathbf{r} \rangle \equiv p_{\mathbf{f}} \bmod B$. The claim follows. ◆

For a given vector $\mathbf{v} \in \mathcal{R}$, call $\mathbf{f} \in \mathcal{F}$ *tight* for $\mathbf{v}$ if $\langle \mathbf{v}, \mathbf{f} \rangle = p_{\mathbf{f}}$. Importantly, Claim 4.9.1 implies the following: if $\mathbf{f}$ is not tight for $\mathbf{v}$, then $\langle \mathbf{f}, \mathbf{v} \rangle \geqslant p_{\mathbf{f}} + B$. Further, let

$$\mathrm{Tight}(\mathbf{v}) \coloneqq \{\mathbf{f} \in \mathcal{F} \mid \mathbf{f} \text{ is tight for } \mathbf{v}\}.$$

The next claim is the key step. Its proof relies on a carefully crafted application of the Deep-in-the-Cone Lemma (Lemma 4.6).

CLAIM 4.9.2. Suppose $\mathbf{u} \in \mathcal{Z}$ and $\mathbf{v} \in \mathcal{R}$ are such that $\mathrm{Tight}(\mathbf{u}) \supseteq \mathrm{Tight}(\mathbf{v})$. Then $\mathbf{v} \in \mathcal{Z}$ as well.

Proof. Denote $\ell \coloneqq |\mathcal{F}|$ for brevity. Enumerate $\mathcal{F}$ as $\{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_\ell\}$ so that

$$\langle \mathbf{f}_1, \mathbf{v} \rangle \leqslant \langle \mathbf{f}_2, \mathbf{v} \rangle \leqslant \langle \mathbf{f}_3, \mathbf{v} \rangle \leqslant \ldots \leqslant \langle \mathbf{f}_\ell, \mathbf{v} \rangle,$$

and recall that all these values are nonnegative integers due to $\mathbf{v}$ being an integer vector in $\mathrm{cone}(\mathcal{D})$ and $\mathcal{F}$ being the dual representation of $\mathrm{cone}(\mathcal{D})$. Let $k \in \{0, 1, \ldots, \ell\}$ be the largest index such that

$$\langle \mathbf{f}_i, \mathbf{v} \rangle \leqslant B_i \qquad \text{for all } i \in \{1, 2, \ldots, k\},$$

and denote

$$\mathcal{G} \coloneqq \{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_k\} \qquad \text{and} \qquad \widehat{B} \coloneqq B_k.$$

By the maximality of k it follows that either $k = \ell$, or $k < \ell$ and $\langle \mathbf{f}_{k+1}, \mathbf{v} \rangle > B_{k+1}$. In any case, by the choice of the sequence $B_0, B_1, \ldots, B_\ell$ we have

$$\langle \mathbf{g}, \mathbf{v} \rangle \leqslant \widehat{B} \quad \text{for all } \mathbf{g} \in \mathcal{G} \qquad \text{and} \qquad \langle \mathbf{f}, \mathbf{v} \rangle > \widehat{M}\widehat{B} \quad \text{for all } \mathbf{f} \in \mathcal{F} - \mathcal{G}. \tag{3}$$

Since $\widehat{B} < B$, all vectors of $\mathcal{G}$ are tight for $\mathbf{v}$; see Claim 4.9.1. As $\mathrm{Tight}(\mathbf{u}) \supseteq \mathrm{Tight}(\mathbf{v})$, these vectors are tight for $\mathbf{u}$ as well. We conclude that

$$\langle \mathbf{g}, \mathbf{u} \rangle = \langle \mathbf{g}, \mathbf{v} \rangle = p_{\mathbf{g}} \qquad \text{for all } \mathbf{g} \in \mathcal{G}. \tag{4}$$

Since $\mathbf{u} \in \mathcal{Z}$ by assumption, there is a collection of nonnegative integers $\langle \lambda_{\mathbf{d}} \colon \mathbf{d} \in \mathcal{D} \rangle$ such that

$$\mathbf{u} = \sum_{\mathbf{d} \in \mathcal{D}} \lambda_{\mathbf{d}} \cdot \mathbf{d}.$$

Define now a collection of nonnegative integers $\langle \mu_{\mathbf{d}} \colon \mathbf{d} \in \mathcal{D} \rangle$ as follows:

— if $\mathbf{d} \notin \mathcal{D}_{\mathcal{G}}$, then $\mu_{\mathbf{d}} = \lambda_{\mathbf{d}}$; and

— otherwise, if $\mathbf{d} \in \mathcal{D}_{\mathcal{G}}$, then $\mu_{\mathbf{d}}$ is the unique integer in $\{0, 1, \ldots, K-1\}$ such that $\lambda_{\mathbf{d}} \equiv \mu_{\mathbf{d}} \bmod K$.

Consider now any $\mathbf{e} \in \mathcal{D} - \mathcal{D}_{\mathcal{G}}$. By the definition of $\mathcal{D}_{\mathcal{G}}$, there exists $\mathbf{g} \in \mathcal{G}$ such that $\langle \mathbf{g}, \mathbf{e} \rangle > 0$, which by integrality entails $\langle \mathbf{g}, \mathbf{e} \rangle \geqslant 1$. By Observation 4.3, (3), and (4), we have

$$\widehat{B} \geqslant \langle \mathbf{g}, \mathbf{v} \rangle = \langle \mathbf{g}, \mathbf{u} \rangle = \sum_{\mathbf{d} \in \mathcal{D}} \lambda_{\mathbf{d}} \cdot \langle \mathbf{g}, \mathbf{d} \rangle \geqslant \lambda_{\mathbf{e}} \cdot \langle \mathbf{g}, \mathbf{e} \rangle \geqslant \lambda_{\mathbf{e}} = \mu_{\mathbf{e}}.$$

Together with the straightforward inequality $\mu_{\mathbf{d}} \leqslant K \leqslant \widehat{B}$ for all $\mathbf{d} \in \mathcal{D}_{\mathcal{G}}$, we conclude that

$$\mu_{\mathbf{d}} \leqslant \widehat{B} \qquad \text{for all } \mathbf{d} \in \mathcal{D}. \tag{5}$$

Let

$$\mathbf{u}' \coloneqq \sum_{\mathbf{d} \in \mathcal{D}} \mu_{\mathbf{d}} \cdot \mathbf{d}.$$

Clearly $\mathbf{u}' \in \mathsf{intCone}(\mathcal{D})$. Our goal is to show that Lemma 4.6 can be applied to $\mathbf{v} - \mathbf{u}'$.

First, consider any $\mathbf{f} \in \mathcal{F} - \mathcal{G}$. Then by (3) and (5), we have

$$\begin{aligned} \langle \mathbf{f}, \mathbf{v} - \mathbf{u}' \rangle &= \langle \mathbf{f}, \mathbf{v} \rangle - \langle \mathbf{f}, \mathbf{u}' \rangle > \widehat{M}\widehat{B} - \sum_{\mathbf{d} \in \mathcal{D}} \mu_{\mathbf{d}} \cdot \langle \mathbf{f}, \mathbf{d} \rangle \\ &\geqslant \widehat{M}\widehat{B} - |\mathcal{D}| \cdot \max_{\mathbf{f} \in \mathcal{F}} \|\mathbf{f}\|_1 \cdot \max_{\mathbf{d} \in \mathcal{D}} \|\mathbf{d}\|_\infty \cdot \widehat{B} = M\widehat{B} \geqslant M. \end{aligned} \tag{6}$$

Next, consider any $\mathbf{g} \in \mathcal{G}$. Then

$$\langle \mathbf{g}, \mathbf{v} - \mathbf{u}' \rangle = \sum_{\mathbf{d} \in \mathcal{D}} (\lambda_{\mathbf{d}} - \mu_{\mathbf{d}}) \cdot \langle \mathbf{g}, \mathbf{d} \rangle = \sum_{\mathbf{d} \in \mathcal{D}_{\mathcal{G}}} (\lambda_{\mathbf{d}} - \mu_{\mathbf{d}}) \cdot \langle \mathbf{g}, \mathbf{d} \rangle + \sum_{\mathbf{d} \in \mathcal{D} - \mathcal{D}_{\mathcal{G}}} (\lambda_{\mathbf{d}} - \mu_{\mathbf{d}}) \cdot \langle \mathbf{g}, \mathbf{d} \rangle. \tag{7}$$

Note that for $\mathbf{d} \in \mathcal{D} - \mathcal{D}_{\mathcal{G}}$ we have $\lambda_{\mathbf{d}} = \mu_{\mathbf{d}}$, while for $\mathbf{d} \in \mathcal{D}_{\mathcal{G}}$ we have $\langle \mathbf{g}, \mathbf{d} \rangle = 0$. So both summands on the right hand side of (7) are equal to 0, implying that

$$\langle \mathbf{g}, \mathbf{v} - \mathbf{u}' \rangle = 0 \qquad \text{for all } \mathbf{g} \in \mathcal{G}. \tag{8}$$

Observe that since $\mathcal{F}$ is the dual representation of $\mathsf{cone}(\mathcal{D})$, (6) and (8) together imply that

$$\mathbf{v} - \mathbf{u}' \in \mathsf{cone}(\mathcal{D}). \tag{9}$$

Then by combining (8), (9), and Lemma 4.4 we conclude that

$$\mathbf{v} - \mathbf{u}' \in \mathsf{cone}(\mathcal{D}_{\mathcal{G}}). \tag{10}$$

Finally, observe that for every $\mathbf{d} \in \mathcal{D}$, $\lambda_{\mathbf{d}} - \mu_{\mathbf{d}}$ is an integer divisible by K. Therefore, $\mathbf{u} - \mathbf{u}' = K \cdot \mathbf{w}$ for some integer vector $\mathbf{w}$, implying that $\mathbf{u} - \mathbf{u}' \in \Lambda_{\mathbf{0}}^K$. As $\mathbf{u}, \mathbf{v} \in \Lambda_{\mathbf{r}}^B$ and K divides B, we also have $\mathbf{v} - \mathbf{u} \in \Lambda_{\mathbf{0}}^K$. Combining these two observations yields

$$\mathbf{v} - \mathbf{u}' \in \Lambda_{\mathbf{0}}^K. \tag{11}$$

Noting that $\mathbf{0} \in \mathsf{lattice}(\mathcal{D}_\mathcal{G}) \cap \Lambda_\mathbf{0}^K$ and K is divisible by $K_\mathcal{G}$, by applying Lemma 4.8 to $\mathcal{D}_\mathcal{G}$ we infer that

$$\mathsf{span}(\mathcal{D}_\mathcal{G}) \cap \Lambda_\mathbf{0}^K \subseteq \mathsf{lattice}(\mathcal{D}_\mathcal{G}).$$

Combining this with (10) and (11) yields

$$\mathbf{v} - \mathbf{u}' \in \mathsf{lattice}(\mathcal{D}_\mathcal{G}). \tag{12}$$

Now, assertions (6), (10), and (12) show that we can use Lemma 4.6 to conclude that

$$\mathbf{v} - \mathbf{u}' \in \mathsf{intCone}(\mathcal{D}_\mathcal{G}).$$

Since $\mathbf{u}' = \sum_{\mathbf{d}\in\mathcal{D}} \mu_\mathbf{d} \cdot \mathbf{d}$ and $\langle \mu_\mathbf{d} \colon \mathbf{d} \in \mathcal{D}\rangle$ are nonnegative integers, we have $\mathbf{u}' \in \mathsf{intCone}(\mathcal{D}_\mathcal{G})$ as well. So as a sum of two vectors from $\mathsf{intCone}(\mathcal{D}_\mathcal{G})$, $\mathbf{v}$ also belongs to $\mathsf{intCone}(\mathcal{D}_\mathcal{G})$, and we are done. ◆

For every $\mathcal{G} \subseteq \mathcal{F}$, let

$$\mathcal{R}_\mathcal{G} := \{\mathbf{v} \in \mathcal{R} \mid \mathsf{Tight}(\mathbf{v}) = \mathcal{G}\}.$$

In other words, $\mathcal{R}_\mathcal{G}$ comprises all vectors in $\mathcal{R}$ for which $\mathcal{G}$ is exactly the set of tight vectors in $\mathcal{F}$. Thus, $\{\mathcal{R}_\mathcal{G} \colon \mathcal{G} \subseteq \mathcal{F}\}$ is a partition of $\mathcal{R}$.

Let $\mathscr{L}$ be the family of subsets of $\mathcal{F}$ consisting of all $\mathcal{G} \subseteq \mathcal{F}$ such that $\mathcal{R}_\mathcal{G} \cap \mathcal{Z} \neq \emptyset$. Further, let $\mathscr{L}^\downarrow$ be the downward closure of $\mathscr{L}$: a set $\mathcal{G} \subseteq \mathcal{F}$ belongs to $\mathscr{L}^\downarrow$ if and only if there exists $\mathcal{G}' \supseteq \mathcal{G}$ such that $\mathcal{G}' \in \mathscr{L}$. The following statement is an immediate corollary of Claim 4.9.2.

CLAIM 4.9.3. Suppose $\mathcal{G} \subseteq \mathcal{F}$. If $\mathcal{G} \in \mathscr{L}^\downarrow$ then $\mathcal{R}_\mathcal{G} \subseteq \mathcal{Z}$, and if $\mathcal{G} \notin \mathscr{L}^\downarrow$ then $\mathcal{R}_\mathcal{G} \cap \mathcal{Z} = \emptyset$. Consequently,

$$\mathcal{Z} = \bigcup_{\mathcal{G}\in\mathscr{L}^\downarrow} \mathcal{R}_\mathcal{G}.$$

Next, we show that the characterization of Claim 4.9.3 can be expressed through linear inequalities.

CLAIM 4.9.4. For every $\mathbf{v} \in \Lambda_\mathbf{r}^B$, the following conditions are equivalent:

(1) $\mathbf{v} \in \mathsf{intCone}(\mathcal{D})$; and

(2) $\mathbf{v} \in \mathsf{cone}(\mathcal{D})$ and

$$\sum_{\mathbf{g}\in\mathcal{G}} \langle \mathbf{g}, \mathbf{v}\rangle \geqslant 1 + \sum_{\mathbf{g}\in\mathcal{G}} p_\mathbf{g} \qquad \text{for all } \mathcal{G} \subseteq \mathcal{F} \text{ such that } \mathcal{G} \notin \mathscr{L}^\downarrow.$$

Proof. We first prove implication (1)⇒(2). Since $\mathbf{v} \in \Lambda_\mathbf{r}^B \cap \mathsf{intCone}(\mathcal{D}) = \mathcal{Z}$, by Claim 4.9.3 we have that $\mathbf{v} \in \mathcal{R}_\mathcal{G}$ for some $\mathcal{G} \in \mathscr{L}^\downarrow$. In particular, for every $\mathcal{G}' \subseteq \mathcal{F}$ with $\mathcal{G}' \notin \mathscr{L}^\downarrow$, $\mathcal{G}'$ is not a subset of $\mathcal{G}$, implying that there exists some $\mathbf{h} \in \mathcal{G}' - \mathcal{G}$. As $\mathcal{G} = \mathsf{Tight}(\mathbf{v})$, $\mathbf{h}$ is not tight for $\mathbf{v}$,

hence $\langle \mathbf{h}, \mathbf{v} \rangle \geqslant p_{\mathbf{h}} + B$ by Claim 4.9.1. Hence, by Claim 4.9.1 again,

$$\sum_{\mathbf{g} \in \mathcal{G}'} \langle \mathbf{g}, \mathbf{v} \rangle \geqslant B + \sum_{\mathbf{g} \in \mathcal{G}'} p_{\mathbf{g}} \geqslant 1 + \sum_{\mathbf{g} \in \mathcal{G}'} p_{\mathbf{g}};$$

and this holds for each $\mathcal{G}' \subseteq \mathcal{F}$ with $\mathcal{G}' \notin \mathscr{L}^{\downarrow}$. As $\mathbf{v} \in \mathsf{intCone}(\mathcal{D})$ entails $\mathbf{v} \in \mathsf{cone}(\mathcal{D})$, this proves (2).

We now move to the implication (2)⇒(1). Since $\mathbf{v} \in \Lambda_{\mathbf{r}}^{B} \cap \mathsf{cone}(\mathcal{D}) = \mathcal{R}$, by Claim 4.9.3 it suffices to show that the unique $\mathcal{G} \subseteq \mathcal{F}$ for which $\mathbf{v} \in \mathcal{R}_{\mathcal{G}}$ satisfies $\mathcal{G} \in \mathscr{L}^{\downarrow}$. Note $\mathbf{v} \in \mathcal{R}_{\mathcal{G}}$ is equivalent to $\mathcal{G} = \mathsf{Tight}(\mathbf{v})$. Therefore $\langle \mathbf{g}, \mathbf{v} \rangle = p_{\mathbf{g}}$ for all $\mathbf{g} \in \mathcal{G}$, implying that

$$\sum_{\mathbf{g} \in \mathcal{G}} \langle \mathbf{g}, \mathbf{v} \rangle = \sum_{\mathbf{g} \in \mathcal{G}} p_{\mathbf{g}}.$$

Hence we necessarily must have $\mathcal{G} \in \mathscr{L}^{\downarrow}$, for otherwise (2) would not be satisfied. ◆

We may now define $\mathcal{Q}$ to be the set of the following pairs:

— $(\mathbf{f}, 0)$ for all $\mathbf{f} \in \mathcal{F}$; and
— $\left(\sum_{\mathbf{g} \in \mathcal{G}} \mathbf{g}, 1 + \sum_{\mathbf{g} \in \mathcal{G}} p_{\mathbf{g}}\right)$ for all $\mathcal{G} \subseteq \mathcal{F}$ with $\mathcal{G} \notin \mathscr{L}^{\downarrow}$.

As $\mathcal{F}$ is a dual representation of $\mathsf{cone}(\mathcal{D})$, we have $\mathsf{cone}(\mathcal{D}) = \{\mathbf{v} \in \mathbb{R}^{\mathbf{t}} \mid \langle \mathbf{f}, \mathbf{v} \rangle \geqslant 0 \text{ for all } \mathbf{f} \in \mathcal{F}\}$. Hence, Claim 4.9.4 directly implies that

$$\Lambda_{\mathbf{r}}^{B} \cap \mathsf{intCone}(\mathcal{D}) = \Lambda_{\mathbf{r}}^{B} \cap \left\{\mathbf{v} \in \mathbb{R}^{\mathbf{t}} \mid \langle \mathbf{q}, \mathbf{v} \rangle \geqslant a \text{ for all } (\mathbf{q}, a) \in \mathcal{Q}\right\},$$

as required.

It remains to argue that the set $\mathcal{Q}$ can be computed given $\mathcal{D}$ and $\mathbf{r}$. For this, it suffices to distinguish which sets $\mathcal{G} \subseteq \mathcal{F}$ belong to $\mathscr{L}$ and which not, because based on this knowledge we may compute $\mathscr{L}^{\downarrow}$ and then construct $\mathcal{Q}$ right from the definition. (Recall here that by Theorem 4.2, $\mathcal{F}$ can be computed from $\mathcal{D}$.) Testing whether given $\mathcal{G} \subseteq \mathcal{F}$ belongs to $\mathscr{L}$ boils down to verifying whether there exists $\mathbf{v} \in \mathbb{Z}^{\mathbf{t}}$ such that

— $\mathbf{v} \in \Lambda_{\mathbf{r}}^{B}$, or equivalently, $\mathbf{v} = B \cdot \mathbf{w} + \mathbf{r}$ for some $\mathbf{w} \in \mathbb{Z}^{\mathbf{t}}$;
— $\langle \mathbf{g}, \mathbf{v} \rangle = p_{\mathbf{g}}$ for all $\mathbf{g} \in \mathcal{G}$; and
— $\langle \mathbf{g}, \mathbf{v} \rangle \geqslant p_{\mathbf{g}} + 1$ for all $\mathbf{g} \in \mathcal{F} - \mathcal{G}$; and
— $\mathbf{v} \in \mathsf{lattice}(\mathcal{D})$, or equivalently, there exist nonnegative integer coefficients $\langle \lambda_{\mathbf{d}} \colon \mathbf{d} \in \mathcal{D} \rangle$ such that $\mathbf{v} = \sum_{\mathbf{d} \in \mathcal{D}} \lambda_{\mathbf{d}} \cdot \mathbf{d}$.

These conditions form an integer program with $2|\mathbf{t}| + |\mathcal{D}|$ variables: $|\mathbf{t}|$ variables for $\mathbf{v}$, $|\mathbf{t}|$ variables for $\mathbf{w}$, and $|\mathcal{D}|$ variables for the coefficients $\langle \lambda_{\mathbf{d}} \colon \mathbf{d} \in \mathcal{D} \rangle$. Hence we may just check the feasibility of this program using any of the standard algorithms for integer programming, e.g., the algorithm of Kannan [29]. ■

We remark that while the statement of Theorem 4.9 does not specify any concrete upper bound on the value of B, it is not hard to trace all the estimates used throughout the reasoning to

see that B is bounded by an elementary function (that is, a constant-height tower of exponents) of the relevant parameters $|\mathbf{t}|$, $|\mathcal{D}|$, and $\max_{\mathbf{d}\in\mathcal{D}} \|\mathcal{D}\|_\infty$. We did not attempt to optimize the value of B in our proof, hence finding tighter estimates is left to future work.

4.4 Algorithm for two-stage stochastic integer programming

With all the tools prepared, we are ready to give a proof of Theorem 4.1.

PROOF OF THEOREM 4.1. For each matrix D_i, let $\mathcal{D}_i$ be the set of columns of D_i. Since every member of $\mathcal{D}_i$ is a vector in $\mathbb{Z}^{\mathbf{t}}$ with all entries of absolute value at most Δ, and there are $(2\Delta+1)^{|\mathbf{t}|}$ different such vectors, we have $|\mathcal{D}_i| \leqslant (2\Delta+1)^{|\mathbf{t}|}$. In particular, there are at most $2^{(2\Delta+1)^{|\mathbf{t}|}}$ distinct sets $\mathcal{D}_i$.

Now, the feasibility of the program P is equivalent to the following assertion: there exist $\mathbf{u} \in \mathbb{Z}^{\mathbf{x}}_{\geqslant 0}$ such that

$$\mathbf{b}_i - A_i\mathbf{u} \in \mathsf{intCone}(\mathcal{D}_i) \qquad \text{for all } i \in I. \tag{13}$$

For each $i \in I$ apply Theorem 4.9 to $\mathcal{D}_i$, yielding a positive integer B_i, computable from $\mathcal{D}_i$. Let B be the least common multiple of all integers B_i for $i \in I$. Since the number of distinct sets $\mathcal{D}_i$ is bounded by $2^{(2\Delta+1)^{|\mathbf{t}|}}$, it follows that B is bounded by a computable function of Δ and $|\mathbf{t}|$.

Towards verification of assertion (13), the algorithm guesses, by iterating through all possibilities, the vector $\mathbf{r} \in \{0, 1, \ldots, B-1\}^{\mathbf{x}}$ such that $\mathbf{u}[x] \equiv \mathbf{r}[x] \bmod B$ for each $x \in \mathbf{x}$; equivalently $\mathbf{u} \in \Lambda^B_{\mathbf{r}}$. This uniquely defines, for each $i \in I$, a vector $\mathbf{r}_i \in \{0, 1, \ldots, B_i - 1\}^{\mathbf{t}}$ such that

$$(\mathbf{b}_i - A_i\mathbf{u})[t] \equiv \mathbf{r}_i[t] \mod B_i \qquad \text{for all } t \in \mathbf{t}.$$

Or equivalently, $\mathbf{b}_i - A_i\mathbf{u} \in \Lambda^{B_i}_{\mathbf{r}_i}$.

Apply the algorithm of Theorem 4.9 to $\mathcal{D}_i$ and $\mathbf{r}_i$, yielding a set of pairs $Q_i \subseteq \mathbb{Z}^{\mathbf{t}} \times \mathbb{Z}$ such that

$$\Lambda^{B_i}_{\mathbf{r}_i} \cap \mathsf{intCone}(\mathcal{D}_i) = \Lambda^{B_i}_{\mathbf{r}_i} \cap \{\mathbf{v} \in \mathbb{R}^{\mathbf{t}} \mid \langle \mathbf{q}, \mathbf{v}\rangle \geqslant a \text{ for all } (\mathbf{q}, a) \in Q_i\}.$$

Hence, under the supposition that indeed $\mathbf{u} \in \Lambda^B_{\mathbf{r}}$, assertion (13) boils down to verifying the existence of $\mathbf{u} \in \mathbb{Z}^{\mathbf{x}}$ satisfying the following:

- $\mathbf{u}[x] \geqslant 0$ for all $x \in \mathbf{x}$;
- $\mathbf{u} \in \Lambda^B_{\mathbf{r}}$, or equivalently, $\mathbf{u} = B \cdot \mathbf{w} + \mathbf{r}$ for some $\mathbf{w} \in \mathbb{Z}^{\mathbf{x}}$;
- for each $i \in I$ and each $(\mathbf{q}, a) \in Q_i$, we have

$$\langle \mathbf{q}, \mathbf{b}_i - A_i\mathbf{u}\rangle \geqslant a.$$

The conditions above form an integer program with $2|\mathbf{x}|$ variables ($|\mathbf{x}|$ variables for $\mathbf{u}$ and $|\mathbf{x}|$ variables for $\mathbf{w}$) whose total bitsize is bounded by $g(\Delta, |\mathbf{t}|) \cdot \|P\|$ for some computable

function g. Hence, we can solve this program (and thus verify the existence of a suitable $\mathbf{u}$) in time $h(\Delta, |\mathbf{x}|, |\mathbf{t}|) \cdot \|P\|$ for a computable h using, for instance, the algorithm of Kannan [29]. Iterating through all $\mathbf{r} \in \{0, 1, \ldots, B-1\}^{\mathbf{x}}$ adds only a multiplicative factor that depends in a computable manner on $\Delta, |\mathbf{x}|, |\mathbf{t}|$, yielding in total a time complexity bound of $f(\Delta, |\mathbf{x}|, |\mathbf{t}|) \cdot \|P\|$ for a computable function f, as claimed. ■

5. n-fold integer programming

Our main result for n-fold integer programming is presented in the following statement, which is just Theorem 1.2 with adjusted notation.

THEOREM 5.1. *An instance $P = (C, \mathbf{a}, \langle D_i, \mathbf{b}_i, \mathbf{c}_i : i \in I\rangle)$ of Uniform n-fold ILP Optimization can be solved in time $f(\Delta, |\mathbf{y}|, |\mathbf{t}|) \cdot \|P\|^{O(1)}$ for some computable function f, where $\Delta = \max_{i\in I} \|D_i\|_\infty$.*

Again, importantly, we do not impose any upper bound on the absolute values of the entries of the matrix C. Note also that as mentioned in Section 1, the considered parameters *do not* include $|\mathbf{s}|$, the number of linking constraints, but *do* include $|\mathbf{y}|$, the number of local variables in each block.

The remainder of this section is devoted to the proof of Theorem 5.1.

5.1 Graver bases

One ingredient that we will repeatedly use in our proofs is the notion of the Graver basis of a matrix, which consists of $\sqsubseteq$-minimal integral elements of the kernel. Formally, the *integer kernel* of a matrix $D \in \mathbb{Z}^{\mathbf{y}\times\mathbf{t}}$, denoted $\ker^{\mathbb{Z}}(D)$, is the set of all integer vectors $\mathbf{u} \in \mathbb{Z}^{\mathbf{t}}$ such that $D\mathbf{u} = \mathbf{0}$. The *Graver basis* of D is the set $\mathsf{Graver}(D)$ of all $\sqsubseteq$-minimal non-zero elements of $\ker^{\mathbb{Z}}(D)$. In other words, a vector $\mathbf{u} \in \ker^{\mathbb{Z}}(D) - \{\mathbf{0}\}$ belongs to $\mathsf{Graver}(D)$ if there is no $\mathbf{u}' \sqsubseteq \mathbf{u}$, $\mathbf{u}' \notin \{\mathbf{0}, \mathbf{u}\}$, such that $\mathbf{u}' \in \ker^{\mathbb{Z}}(D)$ as well.

Note that since $\mathsf{Graver}(D)$ is an antichain in the $\sqsubseteq$ order, which is a well quasi-order on $\mathbb{Z}^{\mathbf{t}}$ by Dickson's Lemma, $\mathsf{Graver}(D)$ is always finite. There are actually many known bounds on the norms of the elements of the Graver basis under different assumptions about the matrix. We will use the following one tailored to matrices with few rows.

LEMMA 5.2 (Lemma 2 of [19]). *Let $D \in \mathbb{Z}^{\mathbf{y}\times\mathbf{t}}$ be an integer matrix, and let $\Delta = \|D\|_\infty$. Then for every $\mathbf{g} \in \mathsf{Graver}(D)$, it holds that*

$$\|\mathbf{g}\|_1 \leqslant (2|\mathbf{t}|\Delta + 1)^{|\mathbf{t}|}$$

We will also use the fact that every solution to an integer program can be decomposed into some solution of bounded norm and a multiset consisting of elements of the Graver basis. This fact was observed by Pottier in [45, Corollary 1], we recall his proof for completeness.

LEMMA 5.3. *Let $D \in \mathbb{Z}^{\mathbf{y}\times\mathbf{t}}$ be an integer matrix and $\mathbf{b} \in \mathbb{Z}^{\mathbf{t}}_{\geqslant 0}$ be an integer vector. Then for every $\mathbf{w} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ such that $D\mathbf{w} = \mathbf{b}$, there exists a vector $\widehat{\mathbf{w}} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ and a multiset $\mathcal{G}$ consisting of nonnegative elements of* $\mathsf{Graver}(D)$ *such that*

$$D\widehat{\mathbf{w}} = \mathbf{b}, \qquad \|\widehat{\mathbf{w}}\| \leqslant (2|\mathbf{t}|(\|D\|_\infty + \|\mathbf{b}\|_\infty) + 1)^{|\mathbf{t}|}, \qquad \text{and} \qquad \mathbf{w} = \widehat{\mathbf{w}} + \sum_{\mathbf{g}\in\mathcal{G}} \mathbf{g}.$$

PROOF. Let z be a fresh variable that does not belong to $\mathbf{y}$ and let $\mathbf{z} := \mathbf{y} \cup \{z\}$. Further, let $D' \in \mathbb{Z}^{\mathbf{z}\times\mathbf{t}}$ be the matrix obtained from D by adding column $D'[z] = -\mathbf{b}$, and $\mathbf{w}' \in \mathbb{Z}^{\mathbf{z}}_{\geqslant 0}$ be the vector obtained from $\mathbf{w}$ by adding the entry $\mathbf{w}'[z] = 1$. Observe that $D\mathbf{w} = \mathbf{b}$ entails $D'\mathbf{w}' = \mathbf{0}$, which means that $\mathbf{w}' \in \ker^{\mathbb{Z}}(D')$. Every element of the integer kernel can be decomposed into a sign-compatible sum of elements of the Graver basis; see e.g. [17, Lemma 3.2.3]. Therefore, there is a multiset $\mathcal{G}'$ consisting of nonnegative elements of $\mathsf{Graver}(D')$ such that $\sum_{\mathbf{g}'\in\mathcal{G}'} \mathbf{g}' = \mathbf{w}'$. Due to the sign-compatibility, exactly one element $\widehat{\mathbf{w}}' \in \mathcal{G}'$ has $\widehat{\mathbf{w}}[z] = 1$, and all the other elements $\mathbf{g}' \in \mathcal{G}' - \{\widehat{\mathbf{w}}'\}$ satisfy $\mathbf{g}'[z] = 0$. We can now obtain $\widehat{\mathbf{w}}$ and $\mathcal{G}$ from $\widehat{\mathbf{w}}'$ and $\mathcal{G}' - \{\widehat{\mathbf{w}}'\}$, respectively, by stripping every vector from the coordinate corresponding to variable z (formally, restricting the domain to $\mathbf{y}$). As $\mathbf{g}'[z] = 0$ for all $\mathbf{g}' \in \mathcal{G}' - \{\widehat{\mathbf{w}}'\}$, it follows that all elements of $\mathcal{G}$ belong to $\mathsf{Graver}(D)$. Finally, from Lemma 5.2 applied to D' we conclude that $\|\widehat{\mathbf{w}}\|_\infty \leqslant \|\widehat{\mathbf{w}}'\|_\infty \leqslant (2|\mathbf{t}|(\|D\|_\infty + \|\mathbf{b}\|_\infty) + 1)^{|\mathbf{t}|}$. ■

5.2 Decomposing bricks

As explained in Section 2, the key idea in the proof of Theorem 5.1 is to decompose right-hand sides of the program (that is, vectors $\mathbf{b}_i$) into smaller and smaller vectors while preserving the optimum value of the program. The next lemma is the key observation that provides a single step of the decomposition.

LEMMA 5.4 (Brick Decomposition Lemma). *There exists a function $g(k,\Delta) \in 2^{(k\Delta)^{O(k)}}$ such that the following holds. Let $D \in \mathbb{Z}^{\mathbf{y}\times\mathbf{t}}$ be a matrix and $\mathbf{b} \in \mathbb{Z}^{\mathbf{t}}$ be a vector such that $\|\mathbf{b}\|_\infty > g(k,\Delta)$, where $k = |\mathbf{t}|$ and $\Delta = \|D\|_\infty$. Then there are non-zero vectors $\mathbf{b}', \mathbf{b}'' \in \mathbb{Z}^{\mathbf{t}}$ such that the following conditions hold:*

- *$\mathbf{b}', \mathbf{b}'' \sqsubseteq \mathbf{b}$ and $\mathbf{b} = \mathbf{b}' + \mathbf{b}''$; and*
- *for every $\mathbf{v} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ satisfying $D\mathbf{v} = \mathbf{b}$, there exist $\mathbf{v}', \mathbf{v}'' \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ such that*

$$\mathbf{v} = \mathbf{v}' + \mathbf{v}'', \qquad D\mathbf{v}' = \mathbf{b}', \qquad \text{and} \qquad D\mathbf{v}'' = \mathbf{b}''.$$

Throughout this section, for a multiset V of vectors or numbers, we denote by $\sum V$ the sum of the elements in V.

The crucial ingredient to prove Lemma 5.4 is the following result of Klein [30]. We use the variant from Cslovjecsek et al. [13], which compared to the original version of Klein [30] has better bounds and applies to vectors with possibly negative coefficients.

LEMMA 5.5 (Klein Lemma, Theorem 4.1 of [13] for $\epsilon = 1$ and with adjusted notation). *Let $\mathbf{t}$ be a tuple of variables with $|\mathbf{t}| = k$ and let $T_1, \ldots, T_n$ be non-empty multisets of vectors in $\mathbb{Z}^{\mathbf{t}}$ such that for all $i \in \{1, \ldots, n\}$ and $\mathbf{u} \in T_i$, we have $\|\mathbf{u}\|_\infty \leqslant \Delta$, and the sum of all elements in each multiset is the same:*

$$\sum T_1 = \sum T_2 = \ldots = \sum T_n.$$

Then there exist non-empty submultisets $S_1 \subseteq T_1, S_2 \subseteq T_2, \ldots, S_n \subseteq T_n$, each of size bounded by $2^{O(k\Delta)^k}$, such that

$$\sum S_1 = \sum S_2 = \ldots = \sum S_n.$$

We also need the following lemma about partitioning a multiset of vectors into small submultisets with sign-compatible sums.

LEMMA 5.6. *Let U be a multiset of vectors in $\mathbb{Z}^{\mathbf{t}}$ with $\|\mathbf{u}\|_\infty \leqslant \Delta$ for all $\mathbf{u} \in U$, where $\mathbf{t}$ is a tuple of variables with $|\mathbf{t}| = k$. Further, let $\mathbf{b} = \sum U$. Then one can partition U into a collection of non-empty submultisets $\langle U_i : i \in I \rangle$ so that for every $i \in I$, we have $|U_i| \leqslant (2k\Delta + 1)^k$ and $\sum U_i \sqsubseteq \mathbf{b}$.*

PROOF. Let W be a multiset of vectors in $\mathbb{Z}^{\mathbf{t}}$ such that

- $\mathbf{w} \sqsubseteq -\mathbf{b}$ and $\|\mathbf{w}\|_\infty \leqslant 1$ for each $\mathbf{w} \in W$, and
- $\sum \mathbf{w} = -\mathbf{b}$.

Let $\mathbf{x}$ and $\mathbf{y}$ be tuples of variables enumerating the elements of U and W, respectively, so that we can set up a matrix $A \in \mathbb{Z}^{(\mathbf{x} \cup \mathbf{y}) \times \mathbf{t}}$ with the elements of U as the columns corresponding to $\mathbf{x}$ and the elements of W as the columns corresponding to $\mathbf{y}$. Thus, we have

$$A\mathbf{1} = \mathbf{0}.$$

By Lemma 5.2, there exists $\mathbf{g} \in \mathsf{Graver}(A)$ such that

$$\|\mathbf{g}\|_1 \leqslant (2k\Delta + 1)^k \qquad \text{and} \qquad \mathbf{g} \sqsubseteq \mathbf{1}.$$

Let U_1 be the multisets of those elements of U that correspond to the variables $x \in \mathbf{x}$ with $\mathbf{g}(x) = 1$. Similarly, let W_1 be the multiset of those elements of W that correspond to the variables $y \in \mathbf{y}$ with $\mathbf{g}(y) = 1$. Thus, we have $|U_1| + |W_1| = \|\mathbf{g}\|_1 \leqslant (2k\Delta + 1)^k$. Moreover, as $A\mathbf{g} = \mathbf{0}$, $\sum W = -\mathbf{b}$, and $\mathbf{w} \sqsubseteq -\mathbf{b}$ for each $\mathbf{w} \in W_1$, we have

$$\sum U_1 = -\sum W_1 \sqsubseteq \mathbf{b}.$$

Therefore, we can set U_1 to be the first multiset in the sought collection. It now remains to apply the same reasoning inductively to the multiset $U - U_1$ in order to extract the next elements of the collection, until the multiset under consideration becomes empty. ■

We are now ready to prove Lemma 5.4.

PROOF OF LEMMA 5.4. Let $\Xi \in 2^{O(k\Delta)^k}$ be the bound provided by Lemma 5.5 for parameters k and Δ.

Call a vector $\mathbf{v} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ a *solution* if $D\mathbf{v} = \mathbf{b}$. Further, a solution $\mathbf{v}$ is *minimal* if it is conformally minimal among the solutions: there is no solution $\mathbf{v}'$ such that $\mathbf{v}' \neq \mathbf{v}$ and $\mathbf{v}' \sqsubseteq \mathbf{v}$. Let V be the set of all minimal solutions. Note that V is an antichain in $\sqsubseteq$, so as $\sqsubseteq$ is a well quasi-order on $\mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ by Dickson's Lemma, it follows that V is finite. If V is empty, then there are no solutions and there is noting to prove, so assume otherwise.

For each $\mathbf{v} \in V$, construct a multiset of vectors $T_{\mathbf{v}}$ as follows: for each $y \in \mathbf{y}$, include $\mathbf{v}[y]$ copies of the column $D[y]$ (i.e., the column of D corresponding to variable y) in $T_{\mathbf{v}}$. Thus, we obtain a finite collection of multisets $\langle T_{\mathbf{v}} \colon \mathbf{v} \in V \rangle$. By construction, we have

$$\sum T_{\mathbf{v}} = D\mathbf{v} = \mathbf{b} \qquad \text{for all } \mathbf{v} \in V.$$

Note that all vectors in all multisets $T_{\mathbf{v}}$ have all entries bounded in absolute value by Δ. In the sequel, we will use the following claim a few times.

CLAIM 5.6.1. Suppose $\mathbf{v} \in V$ and $A \subseteq T_{\mathbf{v}}$ is a submultiset such that $\sum A = \mathbf{0}$. Then A is empty.

Proof. Let $\mathbf{w} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ be the vector of multiplicities in which the columns of D appear in A: for $y \in \mathbf{y}$, $\mathbf{w}[y]$ is the number of times $D[y]$ appears in A. Clearly, $\mathbf{w} \sqsubseteq \mathbf{v}$ and $\sum A = \mathbf{0}$ implies that $D\mathbf{w} = \mathbf{0}$. Then $D(\mathbf{v} - \mathbf{w}) = \mathbf{b}$ and $\mathbf{v} - \mathbf{w} \sqsubseteq \mathbf{v}$ as well. This is a contradiction with the minimality of $\mathbf{v}$ unless $\mathbf{w} = \mathbf{0}$, or equivalently, A is empty. ◆

As V is finite, we may apply Lemma 5.5 to multisets $\langle T_{\mathbf{v}} \colon \mathbf{v} \in V \rangle$. In this way, we obtain submultisets $\langle S_{\mathbf{v}} \subseteq T_{\mathbf{v}} \colon \mathbf{v} \in V \rangle$, each of size at most Ξ, such that

$$\sum S_{\mathbf{v}} = \sum S_{\mathbf{v}'} \qquad \text{for all } \mathbf{v}, \mathbf{v}' \in V.$$

Consider multisets $T'_{\mathbf{v}} := T_{\mathbf{v}} - S_{\mathbf{v}}$ for $\mathbf{v} \in V$. Observe that the multisets $T'_{\mathbf{v}}$ have again the same sums. Moreover, if some $T'_{\mathbf{v}}$ is empty, then $\sum T'_{\mathbf{v}'} = 0$ for every $\mathbf{v}' \in V$, which by Claim 5.6.1 implies that $T'_{\mathbf{v}'}$ is empty as well. It follows that either all multisets $\left\langle T'_{\mathbf{v}} \colon \mathbf{v} \in V \right\rangle$ are empty, or all of them are non-empty.

If all multisets $\left\langle T'_{\mathbf{v}} \colon \mathbf{v} \in V \right\rangle$ are non-empty, then we may apply the same argument to them again, and thus extract suitable non-empty submultisets $\left\langle S'_{\mathbf{v}} \subseteq T'_{\mathbf{v}} \colon \mathbf{v} \in V \right\rangle$ with the same sum. By performing this reasoning iteratively until all multisets become empty (which occurs simultaneously due to Claim 5.6.1), we find a collection of partitions

$$\left\langle \left\langle S^i_{\mathbf{v}} \colon i \in I \right\rangle : \mathbf{v} \in V \right\rangle$$

for some index set I such that:

— $\langle S^i_{\mathbf{v}} : i \in I \rangle$ is a partition of $T_{\mathbf{v}}$ for each $\mathbf{v} \in V$;
— $S^i_{\mathbf{v}}$ is non-empty and $|S^i_{\mathbf{v}}| \leqslant \Xi$ for all $i \in I$ and $\mathbf{v} \in V$; and
— $\sum S^i_{\mathbf{v}} = \sum S^i_{\mathbf{v}'}$ for all $i \in I$ and $\mathbf{v}, \mathbf{v}' \in V$.

For $i \in I$, let $\mathbf{p}_i$ be the common sum of multisets $S^i_{\mathbf{v}}$ for $\mathbf{v} \in V$, that is,

$$\mathbf{p}_i = \sum S^i_{\mathbf{v}} \qquad \text{for all } \mathbf{v} \in V.$$

Note that $\sum_{i \in I} \mathbf{p}_i = \mathbf{b}$ and $\|\mathbf{p}_i\|_\infty \leqslant \Delta\Xi =: \Delta'$ for all $i \in I$.

We now apply Lemma 5.6 to the collection of vectors $\langle \mathbf{p}_i : i \in I \rangle$. This yields a partition $\langle I_j : j \in J \rangle$ of I with some index set J such that for each $j \in J$, we have

$$|I_j| \leqslant (2k\Delta' + 1)^k \leqslant 2^{(k\Delta)^{O(k)}} \qquad \text{and} \qquad \sum_{i \in I_j} \mathbf{p}_i \sqsubseteq \mathbf{b}.$$

By setting $g(k, \Delta) \in 2^{(k\Delta)^{O(k)}}$ large enough, we may guarantee that $|I_j| \cdot \Delta \leqslant g(k, \Delta)$ for all $j \in J$. Since we assumed that $\|b\|_\infty > g(k, d)$, we conclude that $\|b\|_\infty > \sum_{i \in I_j} \mathbf{p}_i$ for all $j \in J$, and consequently it must be the case that $|J| \geqslant 2$.

Let $\{J', J''\}$ be any partition of J with both J' and J'' being non-empty. Define

$$I' := \bigcup_{j \in J'} I_j \qquad \text{and} \qquad I'' := \bigcup_{j \in J''} I_j.$$

Further, we set

$$\mathbf{b}' := \sum_{i \in I'} \mathbf{p}_i \qquad \text{and} \qquad \mathbf{b}'' := \sum_{i \in I''} \mathbf{p}_i.$$

As $\sum_{i \in I} \mathbf{p}_i = \mathbf{b}$ and $\mathbf{p}_i \sqsubseteq \mathbf{b}$ for all $i \in I$, it follows that $\mathbf{b}' + \mathbf{b}'' = \mathbf{b}$ and $\mathbf{b}', \mathbf{b}'' \sqsubseteq \mathbf{b}$. Further, from Claim 5.6.1 we have that both $\mathbf{b}'$ and $\mathbf{b}''$ are non-zero.

Consider any $\mathbf{v} \in V$. Let

$$S'_{\mathbf{v}} := \bigcup_{i \in I'} S^i_{\mathbf{v}} \qquad \text{and} \qquad S''_{\mathbf{v}} := \bigcup_{i \in I''} S^i_{\mathbf{v}}.$$

Note that

$$\sum S'_{\mathbf{v}} = \mathbf{b}' \qquad \text{and} \qquad \sum S''_{\mathbf{v}} = \mathbf{b}''. \tag{14}$$

Further, let $\mathbf{v}' \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ be the vector of multiplicities in which the columns of D appear in $S'_{\mathbf{v}}$: for $y \in \mathbf{y}$, $\mathbf{v}'[y]$ is the number of times $D[y]$ appears in $S'_{\mathbf{v}}$. Define $\mathbf{v}''$ analogously for $S''_{\mathbf{v}}$. Since $\{S'_{\mathbf{v}}, S''_{\mathbf{v}}\}$ is a partition of $T_{\mathbf{v}}$, we have $\mathbf{v}' + \mathbf{v}'' = \mathbf{v}$ and $\mathbf{v}', \mathbf{v}'' \sqsubseteq \mathbf{v}$. Moreover, from (14) it follows that

$$D\mathbf{v}' = \mathbf{b}' \qquad \text{and} \qquad D\mathbf{v}'' = \mathbf{b}''.$$

The above reasoning already settles the second condition from the lemma statement for every minimal solution. So consider now any solution $\mathbf{v} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$. Then there exists a minimal

solution $\widehat{\mathbf{v}} \in V$ such that $\widehat{\mathbf{v}} \sqsubseteq \mathbf{v}$. As both $\mathbf{v}$ and $\widehat{\mathbf{v}}$ are solutions, we have $D(\mathbf{v} - \widehat{\mathbf{v}}) = \mathbf{0}$. It follows that we may simply take

$$\mathbf{v}' := \widehat{\mathbf{v}}' + (\mathbf{v} - \widehat{\mathbf{v}}) \qquad \text{and} \qquad \mathbf{v}'' := \widehat{\mathbf{v}}''.$$

This concludes the proof. ■

From now on, we adopt the function $g(\Delta, k)$ provided by Lemma 5.4 in the notation.

Now that Lemma 5.4 is established, it is tempting to apply it iteratively: first break $\mathbf{b}$ into $\mathbf{b}'$ and $\mathbf{b}''$, then break $\mathbf{b}'$ into two even smaller vectors, and so on. To facilitate the discussion about such decompositions, we introduce the following definition.

DEFINITION 5.7. Let $D \in \mathbb{Z}^{\mathbf{y} \times \mathbf{t}}$ be a matrix and $\mathbf{b} \in \mathbb{Z}^{\mathbf{t}}$ be a vector. A *faithful decomposition* of $\mathbf{b}$ with respect to D is a collection $\langle \mathbf{b}_i : i \in I \rangle$ of non-zero vectors in $\mathbb{Z}^{\mathbf{t}}$ satisfying the following conditions:

- $\mathbf{b}_i \sqsubseteq \mathbf{b}$ for each $i \in I$ and $\mathbf{b} = \sum_{i \in I} \mathbf{b}_i$; and
- for every $\mathbf{v} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ satisfying $D\mathbf{v} = \mathbf{b}$, there exist a collection of vectors $\langle \mathbf{v}_i : i \in I \rangle$ in $\mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ such that

$$\mathbf{v} = \sum_{i \in I} \mathbf{v}_i \qquad \text{and} \qquad D\mathbf{v}_i = \mathbf{b}_i \quad \text{for each } i \in I.$$

The *order* of a faithful decomposition $\langle \mathbf{b}_i : i \in I \rangle$ is $\max_{i \in I} \|\mathbf{b}_i\|_\infty$.

By applying Lemma 5.4 iteratively, we get the following:

LEMMA 5.8. *For every matrix $D \in \mathbb{Z}^{\mathbf{y} \times \mathbf{t}}$ and a non-zero vector $\mathbf{b} \in \mathbb{Z}^{\mathbf{t}}$, there exists a faithful decomposition of $\mathbf{b}$ with respect to D of order at most $g(\Delta, k)$, where $k = |\mathbf{t}|$ and $\Delta = \|D\|_\infty$.*

PROOF. Start with a faithful decomposition $\mathcal{B}$ consisting only of $\mathbf{b}$. Then, as long as $\mathcal{B}$ contains a vector $\mathbf{p}$ with $\|\mathbf{p}\| > g(k, \Delta)$, apply Lemma 5.4 to $\mathbf{p}$ yielding suitable vectors $\mathbf{p}'$ and $\mathbf{p}''$, and replace $\mathbf{p}$ with $\mathbf{p}'$ and $\mathbf{p}''$ in $\mathcal{B}$. It is straightforward to verify that throughout this procedure $\mathcal{B}$ remains a faithful decomposition. Moreover, the size of $\mathcal{B}$ increases in each step of the procedure, while every faithful decomposition of $\mathbf{b}$ has cardinality bounded by $\|\mathbf{b}\|_1$. Therefore, the procedure must eventually end, yielding a faithful decomposition of order at most $g(k, \Delta)$. ■

Observe that in the worst case, a faithful decomposition $\mathcal{B}$ provided by Lemma 5.8 can consist of as many as $\Omega(\|\mathbf{b}\|_1)$ vectors, which in our case is exponential in the size of the bit encoding of $\mathbf{b}$. However, as all vectors in $\mathcal{B}$ have ℓ_∞-norm bounded in terms of k and Δ, the total number of distinct vectors is bounded by a function of k and Δ. Therefore, a faithful decomposition $\mathcal{B}$ provided by Lemma 5.8 can be encoded compactly using $g(k, \Delta) \cdot \log \|\mathbf{b}\|_\infty$ bits, by storing each vector present in $\mathcal{B}$ together with its multiplicity in $\mathcal{B}$, encoded in binary.

In all further algorithmic discussions we will assume that this encoding scheme for faithful decompositions.

Our next goal is to prove that the faithful decomposition of Lemma 5.8 can be computed algorithmically, in fixed-parameter time when parameterized by $|\mathbf{y}|$, k, and Δ, assuming $\mathbf{b}$ is given on input in binary. The idea is to iteratively extract a large fraction of the elements of the decomposition, so that the whole process finishes after a number of steps that is logarithmic in $\|\mathbf{b}\|_\infty$. Each extraction step will be executed using a fixed-parameter algorithm for deciding Presburger Arithmetic. Let us give a brief introduction.

Presburger Arithmetic is the first-order theory of the structure $\langle \mathbb{Z}_{\geqslant 0}, +, 0, 1, 2, \ldots \rangle$, that is, of nonnegative integers equipped with addition (treated as a binary function) and all elements of the universe as constants. More precisely, a *term* is an arithmetic expression using the binary function +, variables (meant to be evaluated to nonnegative integers), and constants (nonnegative integers). Formulas considered in Presburger Arithmetic are constructed from atomic formulas — equalities of terms — using the standard syntax of first-order logic, including standard boolean connectives, negation, and both existential and universal quantification (over nonnegative integers). The semantics is as expected. Note that while comparison is not directly present in the signature, it can be easily emulated by existentially quantifying a slack variable: for two variables x, y, the assertion $x \leqslant y$ is equivalent to $\exists_z\, x + z = y$. As usual, $\varphi(\mathbf{x})$ denotes a formula with a tuple of free variables $\mathbf{x}$, while a *sentence* is a formula without free variables. The *length* of a formula $\varphi(\mathbf{x})$, denoted $\|\varphi\|$, is defined in a standard way by structural induction on the formula and the terms contained within. Here, all constants are deemed to have length 1.

Presburger [46] famously proved that Presburger Arithmetic is decidable: there is an algorithm that given a sentence φ of first-order logic over $\langle \mathbb{Z}_{\geqslant 0}, +, 0, 1, 2, \ldots \rangle$, decides whether φ is true. As observed by Koutecký and Talmon [41], known algorithms for deciding Presburger Arithmetic can be understood as fixed-parameter algorithms in the following sense.

THEOREM 5.9 (follows from [41, Theorem 1]). *Given a first-order sentence φ over the structure $\langle \mathbb{Z}_{\geqslant 0}, +, 0, 1, 2, \ldots \rangle$, with constants encoded in binary, one can decide whether φ is true in time $f(\|\varphi\|) \cdot (\log \Delta)^{O(1)}$, where $\Delta \geqslant 3$ is an upper bound on all the constants present in φ and f is a computable function.*

We remark that Koutecký and Talmon discuss a more general setting where formulas are written in a more concise form, including allowing a direct usage of modular arithmetics, and one considers minimization of convex functions over tuples of variables satisfying constraints expressible in Presburger Arithmetic. Nevertheless, Theorem 5.9, as stated above, follows readily from [41, Theorem 1]. For a proof of [41, Theorem 1], see [40, Theorem 2.2].

We now use Theorem 5.9 to argue that $\forall\exists$ integer programs can be solved efficiently.

LEMMA 5.10. *Suppose $A \in \mathbb{Z}^{\mathbf{x}\times\mathbf{y}}$ and $B \in \mathbb{Z}^{\mathbf{y}\times\mathbf{t}}$ are matrices and $\mathbf{d} \in \mathbb{Z}^{\mathbf{t}}$ is a vector. Then the satisfaction of the following sentence*

$$\forall_{\mathbf{v}\in\mathbb{Z}^{\mathbf{y}}} \left[(B\mathbf{v} \leqslant \mathbf{d}) \implies \exists_{\mathbf{u}\in\mathbb{Z}^{\mathbf{x}}} (A\mathbf{u} \leqslant \mathbf{v})\right]$$

can be verified in time $f(\Delta, |\mathbf{x}|, |\mathbf{y}|, |\mathbf{t}|) \cdot (\log \|\mathbf{d}\|_\infty)^{O(1)}$, where $\Delta = \max(\|A\|_\infty, \|B\|_\infty)$ and f is a computable function.

PROOF. It suffices to combine Theorem 5.9 with the observation that the considered sentence can be easily rewritten to an equivalent first-order sentence over $\langle \mathbb{Z}_{\geqslant 0}, +, 0, 1, 2, \ldots \rangle$, whose length depends only on $\Delta, |\mathbf{x}|, |\mathbf{y}|, |\mathbf{t}|$ and whose constants are bounded by $\|\mathbf{d}\|_\infty$. Indeed, every inequality present in the constraint $B\mathbf{v} \leqslant \mathbf{d}$ can be rewritten to an equality of two terms as follows:

- Every summand on the left hand side with a negative coefficient is moved to the right hand side. Also, if the constant on the right hand side is negative, it is moved to the left hand side. Thus, after this step, every side is a sum of variables multiplied by positive coefficients and positive constants.
- Every summand of the form αv, where α is a positive coefficient and $v = \mathbf{v}[y]$ for some $y \in \mathbf{y}$, is replaced by $\underbrace{v + v + \ldots + v}_{\alpha}$. Note here that $\alpha \leqslant \Delta$.
- Inequality is turned into an equality by existentially quantifying a nonnegative slack variable.

We apply a similar rewriting to the constraints present in $A\mathbf{u} \leqslant \mathbf{v}$. After these steps, the sentence is a first-order sentence over $\langle \mathbb{Z}_{\geqslant 0}, +, 0, 1, 2, \ldots \rangle$. ■

We remark that Lemma 5.10 follows also from the work of Eisenbrand and Shmonin on $\forall\exists$ integer programs; see [21, Theorem 4.2]. In [21] Eisenbrand and Shmonin only speak about polynomial-time solvability for fixed parameters, but a careful inspection of the proof shows that in fact, the algorithm works in fixed-parameter time with relevant parameters, as described in Lemma 5.10.

With Lemma 5.10 in place, we present an algorithm to compute faithful decompositions. We first prove the following statement, which shows that a large fraction of a decomposition can be extracted efficiently.

LEMMA 5.11. *Given a matrix $D \in \mathbb{Z}^{\mathbf{y}\times\mathbf{t}}$ and a non-zero vector $\mathbf{b} \in \mathbb{Z}^{\mathbf{t}}$, one can in time $f(\Delta, |\mathbf{y}|, k) \cdot (\log \|\mathbf{b}\|_\infty)^{O(1)}$, where f is a computable function, $\Delta = \|D\|_\infty$, and $k = |\mathbf{t}|$, compute a faithful decomposition $\mathcal{B}_0$ of $\mathbf{b}$ with respect to D with the following property: the ℓ_∞-norms of all the elements of $\mathcal{B}_0$ are bounded by $g(\Delta, k)$ except for at most one element $\mathbf{b}'$, for which it holds that*

$$\|\mathbf{b}'\|_1 \leqslant \left(1 - \frac{1}{2^{(k\Delta)^{O(k)}}}\right) \cdot \|\mathbf{b}\|_1.$$

PROOF. By Lemma 5.8, $\mathbf{b}$ admits a faithful decomposition $\mathcal{B}$ with respect to D such that the order of $\mathcal{B}$ is at most $\Xi := g(\Delta, k)$. Observe that there are $M := (2\Xi+1)^k$ different vectors in $\mathbb{Z}^{\mathbf{t}}$ of ℓ_∞-norm at most Ξ, hence also at most M different vectors present in $\mathcal{B}$. Therefore, there exists a vector $\mathbf{b}_0$ appearing in $\mathcal{B}$ with multiplicity $\alpha \geqslant 1$ such that

$$\alpha \cdot \|\mathbf{b}_0\|_1 \geqslant \frac{1}{M} \cdot \|\mathbf{b}\|_1.$$

Let α' be the largest power of two such that $\alpha' \leqslant \alpha$. Then we have

$$\alpha' \cdot \|\mathbf{b}_0\|_1 \geqslant \frac{1}{2M} \cdot \|\mathbf{b}\|_1. \tag{15}$$

The algorithm guesses, by trying all possibilities, vector $\mathbf{b}_0$ and power of two α'. Note that there are at most $M \leqslant 2^{(k\Delta)^{O(k)}}$ choices for $\mathbf{b}_0$ and at most $\log \|\mathbf{b}\|_1$ choices for α', as every faithful decomposition of $\mathbf{b}$ consists of at most $\|\mathbf{b}\|_1$ vectors. Having guessed, $\mathbf{b}_0$ and α', we define decomposition $\mathcal{B}_0$ to consist of

— α' copies of the vector $\mathbf{b}_0$, and
— vector $\mathbf{b}' := \mathbf{b} - \alpha'\mathbf{b}_0$.

As $\mathcal{B}$ is a faithful decomposition, it follows that $\mathcal{B}_0$ defined in this way is a faithful decomposition as well, provided $\mathbf{b}_0$ and α' were chosen correctly. Further, we have $\|\mathbf{b}_0\| \leqslant \Xi = g(\Delta, k)$ and, by (15), also

$$\|\mathbf{b}'\|_1 \geqslant \left(1 - \frac{1}{2M}\right) \cdot \|\mathbf{b}\|_1.$$

Note that $2M = 2 \cdot (2\Xi+1)^k = 2 \cdot (2g(\Delta,k)+1)^k \in 2^{(k\Delta)^{O(k)}}$, as promised. So what remains to prove is that for a given choice of $\mathbf{b}_0$ and α', one can efficiently verify whether $\mathcal{B}_0$ defined as above is indeed a faithful decomposition of $\mathbf{b}$.

Before we proceed, observe that by Lemma 5.3, every solution $\mathbf{w} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ to $D\mathbf{w} = \mathbf{b}_0$ can be decomposed as

$$\mathbf{w} = \widehat{\mathbf{w}} + \sum_{\mathbf{g} \in \mathcal{G}} \mathbf{g}, \tag{16}$$

where $\widehat{\mathbf{w}} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ is also a solution to $D\widehat{\mathbf{w}} = \mathbf{b}_0$ that additionally satisfies $\|\widehat{\mathbf{w}}\|_\infty \leqslant (2k(\Xi+\Delta)+1)^k$, while $\mathcal{G}$ is a multiset of nonnegative vectors belonging to the Graver basis of D. Vector $\widehat{\mathbf{w}}$ will be called the *base solution* for $\mathbf{w}$.

With this observation, we can formulate the task of verifying faithfulness of $\mathcal{B}_0$ as a $\forall\exists$ integer program as follows. We need to verify whether for every $\mathbf{v} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ satisfying $D\mathbf{v} = \mathbf{b}$, there is a vector $\mathbf{v}' \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ satisfying $D\mathbf{v}' = \mathbf{b}'$ and vectors $\mathbf{w}_1, \ldots, \mathbf{w}_{\alpha'} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ satisfying $D\mathbf{w}_i = \mathbf{b}_0$ for all $i \in \{1, \ldots, \alpha'\}$ such that $\mathbf{v}' + \sum_{i=1}^{\alpha'} \mathbf{w}_i = \mathbf{v}$. Note that every vector $\mathbf{w}_i$ will have a decomposition (16), with some base vector $\widehat{\mathbf{w}}_i$. Hence, to verify the existence of suitable $\mathbf{v}', \mathbf{w}_1, \ldots, \mathbf{w}_{\alpha'}$, it suffices to find the following objects:

— a vector $\mathbf{v}' \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ satisfying $D\mathbf{v}' = \mathbf{b}'$;

— for every $\widehat{\mathbf{w}} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ such that $\|\widehat{\mathbf{w}}\|_\infty \leqslant (2k(\Xi+\Delta)+1)^k$ and $D\widehat{\mathbf{w}} = \mathbf{b}_0$, a multiplicity $\gamma_{\widehat{\mathbf{w}}} \in \mathbb{Z}_{\geqslant 0}$ signifying how many times $\widehat{\mathbf{w}}$ will appear as the base solution $\widehat{\mathbf{w}}_i$; and
— for every Graver element $\mathbf{g} \in \mathrm{Graver}(D) \cap \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$, a multiplicity $\delta_{\mathbf{g}} \in \mathbb{Z}_{\geqslant 0}$ with which $\mathbf{g}$ will appear in the decompositions (16) of the solutions $\langle \mathbf{w}_i : i \in \{1, \ldots, \alpha'\}\rangle$ in total.

Note that by Lemma 5.2, $\mathrm{Graver}(D)$ can be computed in time depending only on Δ, $|\mathbf{y}|$, and k. The constraints that the above variables should obey, except for integrality and nonnegativity, are the following:

$$\begin{aligned} D\mathbf{v}' &= \mathbf{b}', \\ \mathbf{v}' + \sum_{\widehat{\mathbf{w}}} \gamma_{\widehat{\mathbf{w}}} \cdot \widehat{\mathbf{w}} + \sum_{\mathbf{g}} \delta_{\mathbf{g}} \cdot \mathbf{g} &= \mathbf{v}, \\ \sum_{\widehat{\mathbf{w}}} \gamma_{\widehat{\mathbf{w}}} &= \alpha', \end{aligned}$$

where summations over $\widehat{\mathbf{w}}$ and $\mathbf{g}$ go over all relevant $\widehat{\mathbf{w}}$ and $\mathbf{g}$ for which the corresponding variables are defined. It is easy to see that the feasibility of the integer program presented above is equivalent to the existence of suitable $\mathbf{v}', \mathbf{w}_1, \ldots, \mathbf{w}_{\alpha'}$; note here that the nonnegative elements of the Graver basis can be added to any solution of $D\mathbf{w} = \mathbf{b}_0$ without changing feasibility. As we need to verify the feasibility for every $\mathbf{v} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ satisfying $D\mathbf{v} = \mathbf{b}$, we have effectively constructed a $\forall\exists$ integer program of the form considered in Lemma 5.10. It now remains to observe that the number of variables of this program as well as the absolute values of all the entries in its matrices are bounded by computable functions of Δ, $|\mathbf{y}|$, and k. So we may use the algorithm of Lemma 5.10 to solve this $\forall\exists$ program within the promised running time bounds. ■

We now give the full algorithm, which boils down to applying Lemma 5.11 exhaustively.

LEMMA 5.12. *Given a matrix $D \in \mathbb{Z}^{\mathbf{y}\times\mathbf{t}}$ and a non-zero vector $\mathbf{b} \in \mathbb{Z}^{\mathbf{t}}$, one can in time $f(\Delta, |\mathbf{y}|, k) \cdot (\log \|\mathbf{b}\|_\infty)^{O(1)}$, where f is a computable function, $\Delta = \|D\|_\infty$, and $k = |\mathbf{t}|$, compute a faithful decomposition $\mathcal{B}$ of $\mathbf{b}$ with respect to D of order at most $g(\Delta, k)$.*

PROOF. Again, start with a faithful decomposition $\mathcal{B}$ consisting only of $\mathbf{b}$. Next, as long as $\mathcal{B}$ contains a vector $\mathbf{b}'$ with $\|\mathbf{b}'\| > g(\Delta, k)$, apply Lemma 5.11 to $\mathbf{b}'$, which yields a faithful decomposition $\mathcal{B}'$ of $\mathbf{b}'$. Then replace $\mathbf{b}$ with $\mathcal{B}'$ in $\mathcal{B}$ and continue. It is easy to see that $\mathcal{B}$ remains a faithful decomposition throughout this procedure, hence once all vectors in $\mathcal{B}$ have ℓ_∞-norm bounded by $g(\Delta, k)$, one can simply output $\mathcal{B}$.

It is easy to see that at each point of the procedure, $\mathcal{B}$ contains at most one vector with ℓ_∞-norm larger than $g(\Delta, k)$. Moreover, by Lemma 5.11, at every iteration the ℓ_1-norm of this vector decreases by being multiplied by at most $1 - \frac{1}{2^{(k\Delta)^{O(k)}}}$. It follows that the total number of iterations performed by the procedure is at most $2^{(k\Delta)^{O(k)}} \cdot \log \|\mathbf{b}\|_1 = 2^{(k\Delta)^{O(k)}} \cdot \log \|\mathbf{b}\|_\infty$. Since by Lemma 5.11 every iteration takes time $f(\Delta, |\mathbf{y}|, k) \cdot (\log \|\mathbf{b}\|_\infty)^{O(1)}$, the claim follows. ■

5.3 Algorithm for n-fold integer programming

We are now ready to prove Theorem 5.1.

PROOF OF THEOREM 5.1. Let $P = (C, \mathbf{a}, \langle D_i, \mathbf{b}_i, \mathbf{c}_i : i \in I\rangle)$ be the given program. By adding some dummy variables and constraints if necessary, we may assume that all vectors $\mathbf{b}_i$ are non-zero. The first step of the algorithm is to construct a new program P', equivalent to P, in which all the right-hand sides $\mathbf{b}_i$ will be bounded in the ℓ_∞-norm. This essentially boils down to applying Lemma 5.12 to every $\mathbf{b}_i$ as follows.

For every $i \in I$, apply Lemma 5.12 to matrix D_i and vector $\mathbf{b}_i$, yielding a faithful decomposition $\mathcal{B}_i$ of $\mathbf{b}_i$ with respect to D_i of order at most $\Xi := g(\Delta, |\mathbf{t}|)$. Obtain a new uniform n-fold program $P' = (C, \mathbf{a}, \langle D_j, \mathbf{b}_j, \mathbf{c}_j : j \in J\rangle)$ by replacing the single vector $\mathbf{b}_i$ with the collection of vectors $\mathcal{B}_i$, for each $i \in I$. Thus every index $j \in J$ originates in some index $i \in I$, and for all indices j originating in i we put $D_j = D_i$ and $\mathbf{c}_j = \mathbf{c}_i$.

Since collections $\mathcal{B}_i$ were faithful decompositions of vectors $\mathbf{b}_i$ with respect to D_i, for all $i \in I$, it follows that the program P' is equivalent to P in terms of feasibility and the minimum attainable value of the optimization goal. This is because every solution to P can be decomposed into a solution to P' with the same optimization goal value using faithfulness of the decompositions $\mathcal{B}_i$, and also every solution to P' can be naturally composed into a solution to P with the same value.

We note that the program P' is not computed by the algorithm explicitly, but in a high-multiplicity form. That is, decompositions $\mathcal{B}_i$ are output by the algorithm of Lemma 5.12 by providing every vector present in $\mathcal{B}_i$ together with its multiplicity in $\mathcal{B}_i$. Consequently, we may describe P' concisely by writing the multiplicity of every *brick type* present in P', where the brick type of an index $j \in J$ is defined by

- the diagonal matrix D_j;
- the right-hand side $\mathbf{b}_j$; and
- the optimization vector $\mathbf{c}_j$.

In other words, indices $j, j' \in J$ with $D_j = D_{j'}$, $\mathbf{b}_j = \mathbf{b}_{j'}$, and $\mathbf{c}_j = \mathbf{c}_{j'}$ are considered to have the same brick type. Note that there are at most $(2\Delta + 1)^{|\mathbf{y}|\cdot|\mathbf{t}|}$ different diagonal matrices D_j, at most $(2\Xi + 1)^{|\mathbf{t}|}$ different right-hand sides $\mathbf{b}_j$, and at most $|I| \leqslant \|P\|$ different optimization vectors $\mathbf{c}_j$, hence the total number of possible brick types is bounded $h(\Delta, |\mathbf{y}|, |\mathbf{t}|) \cdot \|P\|$, for some computable function h. For each such brick type, we store one positive integer of bitsize bounded by $\|P\|$ describing the multiplicity.

We now aim to solve the n-fold integer program P' efficiently by formulating it as another integer program M, and then relaxing M to a mixed integer program M' with a bounded number of integral variables. Intuitively, M will determine multiplicities in which solutions to individual bricks are taken, and assign them to the blocks in an optimal manner. The solutions to bricks will not be guessed explicitly, but through their decompositions provided by Lemma 5.3.

To formulate M, we need first some notation regarding P'. Let

— $\mathsf{Diag} = \{-\Delta, \dots, \Delta\}^{\mathbf{y}\times\mathbf{t}}$ be the set of all possible diagonal blocks; and
— $\mathsf{RHS} = \{-\Xi, \dots, \Xi\}^{\mathbf{t}}$ be the set of all possible right-hand sides.

Also, I is the index set of the collection $\langle \mathbf{c}_i : i \in I\rangle$ consisting of all possible optimization vectors. Thus, the set of all possible brick types that may be present in P' is $\mathsf{Types} := \mathsf{RHS} \times \mathsf{Diag} \times I$, and we assume that P' is stored by providing, for each type $(D, \mathbf{b}, i) \in \mathsf{Types}$, the multiplicity $\mathsf{count}[D, \mathbf{b}, i]$ with which the type $(D, \mathbf{b}, i)$ appears in P'.

For $D \in \mathsf{Diag}$ and $\mathbf{b} \in \mathsf{RHS}$, we let $\mathsf{Base}[D, \mathbf{b}]$ be the set of all *base solutions* for D and $\mathbf{b}$: vectors $\widehat{\mathbf{w}} \in \mathbb{Z}^{\mathbf{y}}_{\geqslant 0}$ such that $D\widehat{\mathbf{w}} = \mathbf{b}$ and $\|\widehat{\mathbf{w}}\| \leqslant (2|\mathbf{t}|(\Delta+\Xi)+1)^{|\mathbf{t}|}$. Recall that by Lemma 5.3, every nonnegative integer solution $\mathbf{w}$ to $D\mathbf{w} = \mathbf{b}$ can be decomposed as

$$\mathbf{w} = \widehat{\mathbf{w}} + \sum_{\mathbf{g}\in\mathcal{G}} \mathbf{g}, \tag{17}$$

where $\widehat{\mathbf{w}} \in \mathsf{Base}[D, \mathbf{b}]$ and $\mathcal{G}$ is a multiset consisting of nonnegative vectors from the Graver basis. For convenience, denote $\mathsf{Gra}^+(D) := \mathsf{Graver}(D) \cap \mathbb{Z}^{\mathbf{t}}_{\geqslant 0}$.

We may now define the variables of M. These will be:

— $\zeta^{D,\mathbf{b}}_{\widehat{\mathbf{w}}}$ for $D \in \mathsf{Diag}$, $\mathbf{b} \in \mathsf{RHS}$, and $\widehat{\mathbf{w}} \in \mathsf{Base}[D, \mathbf{b}]$, signifying how many times in total the base solution $\widehat{\mathbf{w}}$ will be taken for a brick with diagonal block D and right-hand side $\mathbf{b}$;
— $\delta^D_{\mathbf{g}}$ for $D \in \mathsf{Diag}$ and $\mathbf{g} \in \mathsf{Gra}^+(D)$, signifying how many times in total the Graver basis element $\mathbf{g}$ will appear in the decompositions (17) of solutions to bricks; and
— $\omega^{D,\mathbf{b},i}_{\widehat{\mathbf{w}}}$ for $D \in \mathsf{Diag}$, $\mathbf{b} \in \mathsf{RHS}$, $i \in I$, and $\widehat{\mathbf{w}} \in \mathsf{Base}[D, \mathbf{b}]$, signifying how many times the base solution $\widehat{\mathbf{w}}$ will be assigned to a brick of type $(D, \mathbf{b}, i)$.

Next, we define the constraints of M:

$$\sum_{D\in\mathsf{Diag}} \sum_{\mathbf{b}\in\mathsf{RHS}} \left(\sum_{\widehat{\mathbf{w}}\in\mathsf{Base}[D,\mathbf{b}]} \zeta^{D,\mathbf{b}}_{\widehat{\mathbf{w}}} \cdot C\widehat{\mathbf{w}} + \sum_{\mathbf{g}\in\mathsf{Gra}^+(D)} \delta^D_{\mathbf{g}} \cdot C\mathbf{g} \right) = \mathbf{a}; \tag{C1}$$

$$\sum_{i\in I} \omega^{D,\mathbf{b},i}_{\widehat{\mathbf{w}}} = \zeta^{D,\mathbf{b}}_{\widehat{\mathbf{w}}} \qquad \text{for all } D \in \mathsf{Diag}, \mathbf{b} \in \mathsf{RHS}, \text{ and } \widehat{\mathbf{w}} \in \mathsf{Base}[D, \mathbf{b}]; \tag{C2}$$

$$\sum_{\widehat{\mathbf{w}}\in\mathsf{Base}[D,\mathbf{b}]} \omega^{D,\mathbf{b},i}_{\widehat{\mathbf{w}}} = \mathsf{count}[D, \mathbf{b}, i] \qquad \text{for all } D \in \mathsf{Diag}, \mathbf{b} \in \mathsf{RHS}, \text{ and } i \in I; \tag{C3}$$

$$\zeta^{D,\mathbf{b}}_{\widehat{\mathbf{w}}} \in \mathbb{Z}_{\geqslant 0} \qquad \text{for all } D \in \mathsf{Diag}, \mathbf{b} \in \mathsf{RHS}, \text{ and } \widehat{\mathbf{w}} \in \mathsf{Base}[D, \mathbf{b}]; \tag{C4}$$

$$\delta^D_{\mathbf{g}} \in \mathbb{Z}_{\geqslant 0} \qquad \text{for all } D \in \mathsf{Diag} \text{ and } \mathbf{g} \in \mathsf{Gra}^+[D, \mathbf{b}]; \tag{C5}$$

$$\omega^{D,\mathbf{b},i}_{\widehat{\mathbf{w}}} \in \mathbb{Z}_{\geqslant 0} \qquad \text{for all } D \in \mathsf{Diag}, \mathbf{b} \in \mathsf{RHS}, \widehat{\mathbf{w}} \in \mathsf{Base}[D, \mathbf{b}], \text{ and } i \in I. \tag{C6}$$

Let us explain these constraints:

— Constraint (C1) corresponds to the linking constraints of the n-fold program P'.
— Constraint (C2) assures that the numbers $\omega^{D,\mathbf{b},i}_{\widehat{\mathbf{w}}}$ of base solutions $\widehat{\mathbf{w}} \in \mathsf{Base}[D, \mathbf{b}]$ assigned to bricks with different optimization vectors add up to the total number of such base solutions.

— In a similar vein, Constraint (C3) states that the number of bricks of a particular brick type matches the total number of suitable base solutions assigned to them.
— Finally, constraints (C4), (C5), and (C6) ensure the integrality and nonnegativity of our variables.

Last but not least, the objective function of M is to

$$\text{minimize} \quad \sum_{D\in\mathsf{Diag}} \sum_{\mathbf{b}\in\mathsf{RHS}} \sum_{\widehat{\mathbf{w}}\in\mathsf{Base}[D,\mathbf{b}]} \sum_{i\in I} \omega_{\widehat{\mathbf{w}}}^{D,\mathbf{b},i} \cdot \langle \mathbf{c}_i, \widehat{\mathbf{w}} \rangle + \sum_{D\in\mathsf{Diag}} \sum_{\mathbf{g}\in\mathsf{Gra}^+(D)} \delta_{\mathbf{g}}^{D} \cdot \mathrm{best}_{\mathbf{g}}^{D},$$

where for $D \in \mathsf{Diag}$ and $\mathbf{g} \in \mathsf{Gra}^+(D)$, we define $\mathrm{best}_{\mathbf{g}}^{D}$ to be the minimum of $\langle \mathbf{c}_i, \mathbf{g} \rangle$ over all $i \in I$ such that $D_i = D$. Here, the first summand is just the contribution of the base solutions to the optimization goal, while in the second summand we observe that every Graver basis element that appears in decompositions (17) can be assigned freely to any brick with the corresponding diagonal block D, hence we may greedily assign it to the brick where its contribution to the optimization goal is the smallest. Note that here we crucially use the assumption that the n-fold program P' is uniform, because we exploit the fact that assigning both the base solutions and the Graver basis elements to different bricks has always the same effect on the linking constraints.

From the construction it is clear that the integer programs P' and M are equivalent in terms of feasibility and the minimum attainable value of the optimization goal. However, the number of variables of M is too large to solve it directly: while the total number of variables $\zeta_{\widehat{\mathbf{w}}}^{D,\mathbf{b}}$ and $\delta_{\mathbf{g}}^{D}$ is indeed bounded in terms of the parameters, this is not the case for the assignment variables $\omega_{\widehat{\mathbf{w}}}^{D,\mathbf{b},i}$, of which there may be as many as $\Omega(|I|)$.

To solve this difficulty, we consider a mixed program M' that differs from M by replacing (C6) with the following constraint:

$$\omega_{\widehat{\mathbf{w}}}^{D,\mathbf{b},i} \in \mathbb{R}_{\geqslant 0} \quad \text{for all } D \in \mathsf{Diag}, \mathbf{b} \in \mathsf{RHS}, \widehat{\mathbf{w}} \in \mathsf{Base}[D,\mathbf{b}], \text{ and } i \in I. \tag{C7'}$$

That is, we relax variables $\omega_{\widehat{\mathbf{w}}}^{D,\mathbf{b},i}$ to be fractional. We now observe that M and M' in fact have the same optima, because after fixing the integral variables, the program M' encodes a weighted flow problem and therefore its constraint matrix is totally unimodular.

CLAIM 5.12.1. The minimum attainable optimization goal values of programs M and M' are equal.

Proof. As M' is a relaxation of M, it suffices to prove that M' has an optimum solution that is integral. Observe that after fixing the integral variables $\zeta_{\widehat{\mathbf{w}}}^{D,\mathbf{b}}$ and $\delta_{\mathbf{g}}^{D}$, the only remaining constraints are (C2) and (C3). These constraints encode that values $\omega_{\widehat{\mathbf{w}}}^{D,\mathbf{b},i}$ form a feasible flow in the following flow network:

— There is a source s, a sink t, and two set of vertices: $V_s := \{(D,\mathbf{b},\widehat{\mathbf{w}}) : D \in \mathsf{Diag}, \mathbf{b} \in \mathsf{RHS}, \widehat{\mathbf{w}} \in \mathsf{Base}[D,\mathbf{b}]\}$ and $V_t := \{(D,\mathbf{b},i) : D \in \mathsf{Diag}, \mathbf{b} \in \mathsf{RHS}, i \in I\}$.
— For every $(D,\mathbf{b},\widehat{\mathbf{w}}) \in V_s$, there is an arc from s to $(D,\mathbf{b},\widehat{\mathbf{w}})$ with capacity $\zeta_{\widehat{\mathbf{w}}}^{D,\mathbf{b}}$.

— For every $(D, \mathbf{b}, i) \in V_t$, there is an arc from $(D, \mathbf{b}, i)$ to t with capacity $\mathsf{count}[D, \mathbf{b}, i]$.
— For every $D \in \mathsf{Diag}$, $\mathbf{b} \in \mathsf{RHS}$, $\widehat{\mathbf{w}} \in \mathsf{Base}[D, \mathbf{b}]$, and $i \in I$, there is an arc from $(D, \mathbf{b}, \widehat{\mathbf{w}})$ to $(D, \mathbf{b}, i)$ with infinite capacity.

It is well known that constraint matrices of flow problems are totally unimodular, hence so is the matrix of program M' after fixing any evaluation of the integral variables. Vertices of polyhedra defined by totally unimodular matrices are integral, hence after fixing the integral variables of M', the fractional variables can be always made integral without increasing the optimization goal value. This shows that M' always has an optimum solution that is integral. ◆

Consequently, to find the optimum value for the initial n-fold program P, it suffices to solve the mixed program M'. As M' can be constructed in time $f(\Delta, |\mathbf{y}|, |\mathbf{t}|) \cdot \|P\|^{O(1)}$ and has $f(\Delta, |\mathbf{y}|, |\mathbf{t}|)$ integral variables, for some computable function f, we may apply any fixed-parameter algorithm for solving mixed programs, for instance that of Lenstra [44], to solve M' within the promised running time bounds.

Finally, let us remark that while the presented procedure outputs only the optimum value for P, one can actually find an optimum solution to P by (i) finding an optimum mixed solution to M', (ii) finding an optimum integer solution to M' by solving the fractional part integrally using e.g. the flow formulation constructed in Claim 5.12.1, and (iii) translating the obtained optimum solution to M' back to an optimum solution to P. We leave the details to the reader. ■

References

[1] **Divesh Aggarwal**, **Antoine Joux**, **Miklos Santha**, **and Karol Węgrzycki**. Polynomial time algorithms for Integer Programming and Unbounded Subset Sum in the total regime. *ArXiv Preprint*, abs/2407.05435, 2024. URL (11)

[2] **Iskander Aliev and Martin Henk**. Feasibility of integer knapsacks. *SIAM J. Optim.* 20(6):2978–2993, 2010. DOI (8, 11)

[3] **Matthias Aschenbrenner and Raymond Hemmecke**. Finiteness theorems in stochastic integer programming. *Found. Comput. Math.* 7(2):183–227, 2007. DOI (4)

[4] **Eleonore Bach**, **Friedrich Eisenbrand**, **Thomas Rothvoss**, **and Robert Weismantel**. Forall-exist statements in pseudopolynomial time. *36th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2025*, pages 2225–2233. SIAM, 2025. DOI (11)

[5] **Marcin Briański**, **Martin Koutecký**, **Daniel Král'**, **Kristýna Pekárková**, **and Felix Schröder**. Characterization of matrices with bounded Graver bases and depth parameters and applications to integer programming. *Math. Program.* 208(1):497–531, 2024. DOI (4)

[6] **Timothy F. N. Chan**, **Jacob W. Cooper**, **Martin Koutecký**, **Daniel Král**, **and Kristýna Pekárková**. Matrices of optimal tree-depth and a row-invariant parameterized algorithm for integer programming. *SIAM J. Comput.* 51(3):664–700, 2022. DOI (4)

[7] **Hua Chen**, **Lin Chen**, **and Guochuan Zhang**. Block-structured integer programming: Can we parameterize without the largest coefficient? *Discret. Optim.* 46:100743, 2022. DOI (7)

[8] **Hua Chen**, **Lin Chen**, **and Guochuan Zhang**. FPT algorithms for a special block-structured integer program with applications in scheduling. *Math. Program.* 208(1):463–496, 2024. DOI (7)

[9] **Lin Chen**, **Martin Koutecký**, **Lei Xu**, **and Weidong Shi**. New bounds on augmenting steps of block-structured integer programs. *28th Annual European Symposium on Algorithms, ESA 2020*, volume 173 of *LIPIcs*, 33:1–33:19. Schloss Dagstuhl — Leibniz-Zentrum für Informatik, 2020. DOI (7)

[10] **Lin Chen and Dániel Marx**. Covering a tree with rooted subtrees — parameterized and approximation algorithms. *29th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018*, pages 2801–2820. SIAM, 2018. DOI (4)

[11] **Lin Chen**, **Dániel Marx**, **Deshi Ye**, and **Guochuan Zhang**. Parameterized and approximation results for scheduling with a low rank processing time matrix. *34th Symposium on Theoretical Aspects of Computer Science, STACS 2017*, volume 66 of *LIPIcs*, 22:1–22:14. Schloss Dagstuhl — Leibniz-Zentrum für Informatik, 2017. DOI (3, 4)

[12] **Jana Cslovjecsek**, **Friedrich Eisenbrand**, **Christoph Hunkenschröder**, **Lars Rohwedder**, and **Robert Weismantel**. Block-structured integer and linear programming in strongly polynomial and near linear time. *32nd Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2021*, pages 1666–1681. SIAM, 2021. DOI (4–6)

[13] **Jana Cslovjecsek**, **Friedrich Eisenbrand**, **Michał Pilipczuk**, **Moritz Venzin**, and **Robert Weismantel**. Efficient sequential and parallel algorithms for multistage stochastic integer programming using proximity. *29th Annual European Symposium on Algorithms, ESA 2021*, volume 204 of *LIPIcs*, 33:1–33:14. Schloss Dagstuhl — Leibniz-Zentrum für Informatik, 2021. DOI (4–6, 13, 32)

[14] **Jana Cslovjecsek**, **Martin Koutecký**, **Alexandra Lassota**, **Michal Pilipczuk**, and **Adam Polak**. Parameterized algorithms for block-structured integer programs with large entries. *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024, Alexandria, VA, USA, January 7-10, 2024*, pages 740–751. SIAM, 2024. DOI (1)

[15] **Daniel Dadush**. Integer Programming, Lattice Algorithms, and Deterministic Volume Estimation. PhD thesis, Georgia Institute of Technology, USA, 2012. (5)

[16] **Daniel Dadush**, **Friedrich Eisenbrand**, and **Thomas Rothvoss**. From approximate to exact integer programming. *Math. Program.* 210(1):223–241, 2025. DOI (5)

[17] **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. DOI (31)

[18] **Jesús 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 (3, 4)

[19] **Friedrich Eisenbrand**, **Christoph Hunkenschröder**, and **Kim-Manuel Klein**. Faster algorithms for integer programs with block structure. *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018*, volume 107 of *LIPIcs*, 49:1–49:13. Schloss Dagstuhl — Leibniz-Zentrum für Informatik, 2018. DOI (4–6, 15, 30)

[20] **Friedrich Eisenbrand**, **Christoph Hunkenschröder**, **Kim-Manuel Klein**, **Martin Koutecký**, **Asaf Levin**, and **Shmuel Onn**. An algorithmic theory of integer programming. *ArXiv Preprint*, abs/1904.01361, 2019. URL (4)

[21] **Friedrich Eisenbrand and Gennady Shmonin**. Parametric integer programming in fixed dimension. *Math. Oper. Res.* 33(4):839–850, 2008. DOI (15, 37)

[22] **András Frank and Éva Tardos**. An application of simultaneous diophantine approximation in combinatorial optimization. *Combinatorica*, 7(1):49–65, 1987. DOI (5)

[23] **Tomáš Gavenčiak**, **Martin Koutecký**, and **Dušan Knop**. Integer programming in parameterized complexity: five miniatures. *Discret. Optim.* 44(Part):100596, 2022. DOI (3, 4)

[24] **Raymond Hemmecke**, **Matthias Köppe**, and **Robert Weismantel**. Graver basis and proximity techniques for block-structured separable convex integer minimization problems. *Math. Program.* 145(1-2):1–18, 2014. DOI (7)

[25] **Raymond Hemmecke**, **Shmuel Onn**, and **Lyubov Romanchuk**. n-fold integer programming in cubic time. *Math. Program.* 137(1-2):325–341, 2013. DOI (4)

[26] **Klaus Jansen**, **Kim-Manuel Klein**, and **Alexandra Lassota**. The double exponential runtime is tight for 2-stage stochastic ILPs. *Math. Program.* 197(2):1145–1172, 2023. DOI (4)

[27] **Klaus Jansen**, **Kim-Manuel Klein**, **Marten Maack**, and **Malin Rau**. Empowering the configuration-IP: new PTAS results for scheduling with setup times. *Math. Program.* 195(1):367–401, 2022. DOI (3, 4)

[28] **Klaus Jansen**, **Alexandra Lassota**, and **Lars Rohwedder**. Near-linear time algorithm for n-fold ILPs via color coding. *SIAM J. Discret. Math.* 34(4):2282–2299, 2020. DOI (4, 5)

[29] **Ravi Kannan**. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12(3):415–440, 1987. DOI (5, 10, 16, 28, 30)

[30] **Kim-Manuel Klein**. About the complexity of two-stage stochastic IPs. *Math. Program.* 192(1):319–337, 2022. DOI (4, 13, 32)

[31] **Kim-Manuel Klein and Janina Reuter**. Collapsing the tower — On the complexity of multistage stochastic IPs. *ACM Trans. Algorithms*, 20(3):22, 2024. DOI (4)

[32] **Dušan Knop and Martin Koutecký**. Scheduling kernels via configuration LP. *30th Annual European Symposium on Algorithms, ESA 2022*, volume 244 of *LIPIcs*, 73:1–73:15. Schloss Dagstuhl — Leibniz-Zentrum für Informatik, 2022. DOI (4)

[33] **Dušan Knop and Martin Koutecký**. Scheduling meets n-fold integer programming. *J. Sched.* 21(5):493–503, 2018. DOI (3, 4)

[34] **Dušan Knop**, **Martin Koutecký**, **Asaf Levin**, **Matthias Mnich**, and **Shmuel Onn**. High-multiplicity N-fold IP via configuration LP. *Math. Program.* 200(1):199–227, 2023. DOI (15)

[35] **Dušan Knop**, **Martin Koutecký**, **Asaf Levin**, **Matthias Mnich**, and **Shmuel Onn**. Parameterized complexity of configuration integer programs. *Oper. Res. Lett.* 49(6):908–913, 2021. DOI (3, 4)

[36] **Dušan Knop**, **Martin Koutecký**, **and Matthias Mnich**. Combinatorial n-fold integer programming and applications. *Math. Program.* 184(1):1–34, 2020. DOI (3, 4)

[37] **Dušan Knop**, **Martin Koutecký**, **and Matthias Mnich**. Voting and bribing in single-exponential time. *ACM Trans. Economics and Comput.* 8(3):12:1–12:28, 2020. DOI (3, 4)

[38] **Dušan Knop**, **Michał Pilipczuk**, **and Marcin Wrochna**. Tight complexity lower bounds for integer linear programming with few constraints. *ACM Trans. Comput. Theory*, 12(3):19:1–19:19, 2020. DOI (4)

[39] **Martin Koutecký**, **Asaf Levin**, **and Shmuel Onn**. A parameterized strongly polynomial algorithm for block structured integer programs. *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018*, volume 107 of *LIPIcs*, 85:1–85:14. Schloss Dagstuhl — Leibniz-Zentrum für Informatik, 2018. DOI (4–6)

[40] **Martin Koutecký and Nimrod Talmon**. Multi-party campaigning. *ArXiv Preprint*, abs/2005.04455, 2020. URL (36)

[41] **Martin Koutecký and Nimrod Talmon**. Multi-party campaigning. *35th AAAI Conference on Artificial Intelligence, AAAI 2021*, pages 5506–5513. AAAI Press, 2021. DOI (15, 36)

[42] **Martin Koutecký and Johannes Zink**. Complexity of scheduling few types of jobs on related and unrelated machines. *31st International Symposium on Algorithms and Computation, ISAAC 2020*, volume 181 of *LIPIcs*, 18:1–18:17. Schloss Dagstuhl — Leibniz-Zentrum für Informatik, 2020. DOI (4)

[43] **J. C. Lagarias**. The computational complexity of simultaneous diophantine approximation problems. *SIAM J. Comput.* 14(1):196–209, 1985. DOI (5)

[44] **H. W. Lenstra**. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983. (5, 16, 43)

[45] **Loic Pottier**. Minimal solutions of linear diophantine systems: bounds and algorithms. *4th International Conference on Rewriting Techniques and Applications, RTA-91*, volume 488 of *Lecture Notes in Computer Science*, pages 162–173. Springer, 1991. DOI (11, 15, 21, 30)

[46] **Mojżesz Presburger**. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. *Comptes Rendus du I congrès de Mathématiciens des Pays Slaves*, pages 92–101, 1929. (36)

[47] **Victor Reis and Thomas Rothvoss**. The subspace flatness conjecture and faster integer programming. *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023*, pages 974–988. IEEE, 2023. DOI (5)

[48] **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 (3)

[49] **Hermann Weyl**. Elementare Theorie der konvexen Polyeder. *Commentarii Mathematici Helveticii*, 7:290–306, 1935. (10, 20)

A. Proof of Observation 1.3

In this section, we prove Observation 1.3, restated below for convenience.

OBSERVATION 1.3. (Restated) *Suppose the feasibility problem for uniform 4-block programs can be solved in time $f(k,\Delta)\cdot\|P\|^{O(1)}$ for some computable function f, where k is the dimension of every block and Δ is the maximum absolute value of any entry in the constraint matrix. Then the feasibility problem for uniform 4-block programs can be also solved in time $g(k,\max_i\|D_i\|_\infty)\cdot\|P\|^{O(1)}$ for some computable function g under the assumption that all the absolute values of the entries in matrices A,B,C are bounded by n.*

PROOF. Let P be the input program:

$$\mathbf{x}\in\mathbb{Z}^k_{\geqslant 0},\ \mathbf{y}_t\in\mathbb{Z}^k_{\geqslant 0},$$

$$B\mathbf{x}+\sum_{t=1}^{n}C_t\mathbf{y}_t=\mathbf{a}, \qquad \text{and} \tag{18}$$

$$A_t\mathbf{x}+D_t\mathbf{y}_t=\mathbf{b}_t \qquad \text{for all } t=1,2,\ldots,n, \tag{19}$$

where A,B,C,D_i are integer $k\times k$ matrices and $\mathbf{a},\mathbf{b}_i$ are integer vectors of length k. Denote $A=[a_{ij}]_{i,j\in[k]}$, $B=[b_{ij}]_{i,j\in[k]}$, and $C=[c_{ij}]_{i,j\in[k]}$, and recall that we assume that all entries in A,B,C have absolute values bounded by n. Also, let $\mathbf{a}=(a_1,\ldots,a_k)^\intercal$, $\mathbf{x}=(x_1,\ldots,x_k)^\intercal$, and $\mathbf{y}_t=(y_{t,1},\ldots,y_{t,k})^\intercal$, for $t=1,\ldots,n$. Variables $\mathbf{x}$ will be called *global variables* and variables $\mathbf{y}_t$ will be called *local variables*. Similarly, constraints (18) will be called *linking*, and constraints (19) will be called *local*. Local variables and local constraints come in *t groups*, each consisting of k variables/constraints.

We will gradually modify the program P so that at the end, in the modified blocks A,B,C all the coefficients will belong to $\{-1,0,1\}$. During the modification, we do not insist on all blocks being always square matrices; this can be always fixed at the end by adding some zero rows or zero columns.

First, we reduce large entries in C. Note that the ith linking constraint takes the form

$$\sum_{j=1}^{k}b_{ij}\,x_j+\sum_{t=1}^{n}\sum_{j=1}^{k}c_{ij}\,y_{t,j}=a_i,$$

or equivalently,

$$\sum_{j=1}^{k}b_{ij}\,x_j+\sum_{j=1}^{k}c_{ij}\sum_{t=1}^{n}y_{t,j}=a_i.$$

Hence, for every pair $(i,j)\in[k]\times[k]$ we introduce a new global variable z_{ij} together with the constraint

$$-z_{ij}+\sum_{t=1}^{n}y_{t,j}=0,$$

and in the ith linking constraint, we replace the summand $c_{ij} \sum_{t=1}^{n} y_{t,j}$ with the summand $c_{ij} z_{ij}$. It is easy to see that after this modification, only $\{0, 1\}$ entries will appear in the blocks C. This comes at the price of increasing the number of global variables and the number of linking constraints by k^2, and possibly adding some large (but bounded by n in absolute value) entries to the block B.

Next, we remove large entries in A. Suppose then $a_{ij} > 1$ for some $i, j \in [k]$; recall that we assume also that $a_{ij} \leqslant n$. In program P, entry a_{ij} occurs in summands of the form $a_{ij} x_j$; every group of local constraints contains one such summand. Therefore, the idea is to (i) copy the variable x_j to a_i new local variables, (ii) use one new linking constraint to define a new global variable z'_{ij} equal to the sum of the copies, and hence equal to $a_{ij} x_j$, and (iii) replace each usage of the summand $a_{ij} x_j$ in local constraints with just the variable z'_{ij}. This idea can be executed as follows:

(i) For every $t = 1, \ldots, n$, introduce two new local variables $p_{ij,t}$, $p'_{ij,t}$ together with two new local constraints:

$$-x_j + p'_{ij,t} = 0; \qquad \text{and}$$
$$p_{ij,t} - p'_{ij,t} = 0 \text{ when } t \leqslant a_{ij} \qquad \text{and} \qquad p_{ij,t} = 0 \text{ when } t > a_{ij}.$$

(ii) Add a new global variable z'_{ij} together with a new linking constraint

$$-z'_{ij} + \sum_{t=1}^{n} p_{ij,t} = 0.$$

(iii) Replace every usage of summand $a_{ij} x_{ij}$ with variable z'_{ij} in every local constraint within P.

Entries $a_{ij} \in \{-n, \ldots, -2\}$ can be removed similarly: we perform the above construction for $|a_{ij}|$, and then replace each summand $a_{ij} x_{ij}$ with $-z'_{ij}$ instead of z'_{ij}. Entries with large absolute values in B can be removed in exactly the same way.

It is straightforward to see that by applying all the modifications presented above, we obtain a uniform 4-block program with $O(k^2)$ global variables, $O(k^2)$ linking constraints, $O(k^2)$ local variables in each group of local variables, and $O(k^2)$ local constraints in each group of local constraints. Moreover, all entries in the blocks A, B, C belong to $\{-1, 0, 1\}$, while blocks D_i got modified only by adding some $\{-1, 0, 1\}$ entries. Hence, we can just solve the modified program using the assumed fixed-parameter algorithm for feasibility of uniform 4-block programs parameterized by the dimensions of the block and the maximum absolute value of any entry appearing in the constraint matrix. ■

B. Hardness of two-stage stochastic integer programming with large diagonal entries

We conclude our discussion of two-stage stochastic integer programming by a short argument showing that it is indeed necessary to include $\Delta = \max_{i \in I} \|D_i\|_\infty$ among the parameters. This is because when Δ is unbounded the problem becomes strongly NP-hard already for blocks of size $k = 16$, as we prove below. This rules out even a running time of the form $f(k) \cdot (\|P\| + \Delta)^{O(1)}$, and shows that the dependence on Δ needs to be superpolynomial.

We give a reduction from the 3-SAT problem. Let N denote the number of *variables* and M the number of *clauses* in the input formula. The reduction creates an instance of Two-Stage Stochastic ILP Feasability with only one *global* variable $x \in \mathbb{Z}_{\geqslant 0}$, whose role is to encode a satisfying 3-SAT assignment. Let $p_1, p_2, \ldots, p_N$ be the first N primes. In our encoding, $p_i \mid x$ if and only if the i-th SAT variable is set to false. The IP instance contains M diagonal blocks, each corresponding to a single clause.

We use a certain gadget to construct the blocks. The gadget consists of five ILP variables, named $a, b, c, d, e \in \mathbb{Z}_{\geqslant 0}$, and three constraints that involve those five variables as well as the global variable x. Let p be a fixed prime (not an ILP variable). The key properties of the gadget are:

- if $p \mid x$, then every feasible assignment to the five local variables has $b = 0$;
- if $p \nmid x$, then every feasible assignment to the five local variables has $b = 1$; and
- for every $x \in \mathbb{Z}_{\geqslant 0}$ there exists a feasible assignment to the five local variables.

We leave it to the reader to verify that the following gadget satisfies the desired properties.

$$\begin{aligned} -x + p \cdot a + b + c &= 0 \\ (p-2) \cdot b - c - d &= 0 \qquad \text{(equivalent to } c \leqslant (p-2) \cdot b\text{)} \\ b + e &= 1 \qquad \text{(equivalent to } b \leqslant 1\text{)} \end{aligned}$$

Each block consists of three such independent gadgets, instantiated for the three primes corresponding to the three SAT variables involved in the corresponding clause. Moreover, each block contains one additional constraint (and one additional local variable) ensuring that the three b-variables (or their negations in case of negative literals) sum up to at least one (minus the number of negative literals in the clause), which corresponds to the clause being satisfied.

Summarizing, the resulting two-stage stochastic ILP instance is:

$$D_i = \begin{bmatrix}
p_{v_{i,1}} & 1 & 1 & & & & & & & & & & & & & \\
 & p_{v_{i,1}}-2 & -1 & -1 & & & & & & & & & & & & \\
 & 1 & & & 1 & & & & & & & & & & & \\
 & & & & & p_{v_{i,2}} & 1 & 1 & & & & & & & & \\
 & & & & & & p_{v_{i,2}}-2 & -1 & -1 & & & & & & & \\
 & & & & & & 1 & & & 1 & & & & & & \\
 & & & & & & & & & & p_{v_{i,3}} & 1 & 1 & & & \\
 & & & & & & & & & & & p_{v_{i,3}}-2 & -1 & -1 & & \\
 & & & & & & & & & & & 1 & & & 1 & \\
 & t_{i,1} & & & & & t_{i,2} & & & & & t_{i,3} & & & & -1
\end{bmatrix},$$

$$A_i = \begin{bmatrix} -1 \\ 0 \\ 0 \\ -1 \\ 0 \\ 0 \\ -1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{b}_i = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1-\ell_i \end{bmatrix}, \quad \text{for } i \in [M],$$

where, for every for $i \in [M]$ and $j \in \{1,2,3\}$, $v_{i,j}$ denotes the index of the variable in the j-th literal in the i-th clause, $t_{i,j} = 1$ if this is a positive literal and $t_{i,j} = -1$ otherwise, and ℓ_i equals the number of negative literals in the i-th clause, i.e., $\ell_i = (3 - (t_{i,1} + t_{i,2} + t_{i,3}))/2$.

From the preceding discussion it follows that the obtained integer program is feasible if and only if the input 3-SAT formula is satisfiable. Each A_i is a 10×1 matrix, each D_i is a 10×16 matrix, and we have $\max_{i \in I} \|D_i\|_\infty \leqslant p_N = O(N \log N)$ by the Prime Number Theorem. We conclude that the two-stage stochastic integer programming feasibility problem is strongly NP-hard already for $k = 16$.