INSTITUT FÜR INFORMATIK

A Parameterized Algorithm for 2-Stage Stochastic IPs with one row

Klaus Jansen, Lis Pirotton, Malte Tutas, Corinna Wambsganz

Bericht Nr. 2503
September 2025
ISSN 2192-6247

CHRISTIAN-ALBRECHTS-UNIVERSITÄT

ZU KIEL

Institut für Informatik der
Christian-Albrechts-Universität zu Kiel
Olshausenstr. 40
D – 24098 Kiel

A Parameterized Algorithm for 2-Stage Stochastic IPs with one row

Klaus Jansen, Lis Pirotton, Malte Tutas, Corinna Wambsganz

Bericht Nr. 2503
September 2025
ISSN 2192-6247

e-mail: {kj, lpi, mtu, cwa}@informatik.uni-kiel.de

Technical Report

Abstract

We consider 2-stage stochastic integer linear problems (ILPs) with one row and identical first stage. A 2-stage stochastic ILP is an integer program of the form $\max\{c^T x | \mathcal{A}x = b, x \in \mathbb{Z}^{nt+s}_{\geq 0}\}$. It is split into a first stage of vertically aligned matrices $A_i \in \mathbb{Z}^{r \times s}$ and diagonally aligned second stage matrices $B_i \in \mathbb{Z}^{r \times t}$.

We consider a special case of such 2-stage ILPs where the first stage matrices A_i are identical, and $r = 1$. For this case, we present separate algorithms for deciding the feasibility problem and for solving the optimization problem. For the feasibility problem, our algorithm achieves a running time of $O(n \cdot \Delta^2 \log(\Delta))$, where $\Delta = \|\mathcal{A}\|_\infty$, which significantly improves upon state-of-the-art algorithms for general 2-stage stochastic ILPs such as Cslovjecsek et al. (SODA '24) when applied to our context. For the optimality problem, our algorithm yields a running time of $3^{O(\Delta)} \cdot \tilde{O}(n \cdot \Delta^2)$ or $O(n \cdot 3^\Delta)$. We achieve our results by novel use of structural properties to the ILPs we consider. Additionally, we provide an upper bound on the right hand side $\|b\|_\infty \leq 3^\Delta + 2\Delta^2$. We further complement our positive results by showing a lower bound on the running times of 2-stage stochastic ILPs with one row of $2^{\delta\Delta^{1/2-\varepsilon}}$ for some $\delta, \varepsilon > 0$ which follows via a bound derived from ETH.

2012 ACM Subject Classification Theory of computation → Fixed parameter tractability

Keywords and phrases 2-stage, Integer Linear Programming, Knapsack, diophantine equation

Funding The authors and this research was supported by Deutsche Forschungsgemeinschaft (DFG) - Project number 528381760: "Structural results for integer linear programs"

Acknowledgements We thank Kim-Manuel Klein for hinting residue tables.

1 Introduction

This work considers *integer linear programming* (ILP) problems

$$\max\{c^T x : \mathcal{A}x = b, x \in \mathbb{N}_{\geq 0}\}, \tag{1}$$

which contain a constraint matrix $\mathcal{A} \in \mathbb{Z}^{m \times n}$, a linear objective function $c \in \mathbb{Z}^n$ and a right-hand side vector $b \in \mathbb{Z}^m$. When considering the problem without an objective function, i.e. only deciding whether an integer solution satisfying $\mathcal{A}x = b$ exists, we refer to the problem as the feasibility integer programming problem (ILP). Integer programming is part of the twenty-one problems originally shown by Karp to be NP-hard in his seminal work [31]. These general ILPs have been the focus of long-running lines of research, see e.g. [19, 30, 36, 39, 27], where the last two references are the current state of the art in running times regarding n and m respectively.

Two-stage Integer Linear Programming

With the general problem being NP-hard to solve, focus gradually expanded to finding structures for which integer programs are solvable more efficiently. One such structure is the *two-stage stochastic* IP, which can be illustrated as follows:

$$\mathcal{A} := \begin{pmatrix} A_1 & B_1 & \mathbf{0} & \dots & \mathbf{0} \\ A_2 & \mathbf{0} & B_2 & & \vdots \\ \vdots & \vdots & & \ddots & \mathbf{0} \\ A_n & \mathbf{0} & \dots & \mathbf{0} & B_n \end{pmatrix} \tag{2}$$

Here, we have n blocks $A_i \in \mathbb{Z}^{r \times s}$ and $B_i \in \mathbb{Z}^{r \times t}$. Consequentially, we have $\mathcal{A} \in \mathbb{Z}^{nr \times s+nt}$, $c \in \mathbb{Z}^{s+nt}$ and $b \in \mathbb{Z}^{nr}$. Note how two-stage stochastic IPs can be viewed as n independent problems when removing the first s columns. In this work, we refer to submatrices A_i, B_i as *blocks* of the original matrix. Similarly, sections of the solution x or the cost-vector c can be separated into $n+1$ *bricks*, one of length s and n of length t. Each of these bricks corresponds to the A_i blocks or one B_i block respectively. We denote the i-th brick of x as $x^{(i)}$, such that we decompose $x^T = (x^{(0)}, x^{(1)}, \dots, x^{(n)})$. When referring to a single index of a brick, we use an index, i.e. $x_3^{(2)}$ refers to the third entry of brick 2.

Due to their specific structure, such two-stage stochastic IPs are known to be solvable in FPT-time depending on the dimensions of their blocks s, t and $\Delta := \|\mathcal{A}\|_\infty$, the largest entry of the constraint matrix, with a running time of $2^{((s+t)\Delta)^{O(s^2t+st^2)}} \cdot n \log^{O(st)} n$ [10].

Two-stage stochastic ILPs are used to model decision making processes under uncertainty, for example in the context of manufacturing [13]. In these contexts, the A blocks represent a decision to be made and each of the B_i blocks represents consequences in each of n different possible circumstances (e.g. sickness of employees, customer demands, etc.). The objective is to make a decision in the first stage that maximizes profits across all possible uncertain circumstances, with bricks in the objective function weighted according to likelihoods of the respective events occurring. However, the general two-stage formulation cannot be solved in time of $2^{2^{\delta(t+r)}}|I|^{O(1)}$, where $|I|$ denotes the input length, unless the ETH fails, as shown by Jansen, Klein and Lassota [24]. As such, we focus our work on a more restricted sub-case to better understand its complexity and since many applications can be modelled with it.

In this work, we consider a restricted variant of two-stage ILPs. We consider matrices with $A_i = A, i \in \{1, \dots, n\}$ and $r = 1$, i.e. matrices where the first stage is identical for all

blocks and each block contains only one row:

$$\mathcal{A} := \begin{pmatrix} A & B_1 & \mathbf{0} & \dots & \mathbf{0} \\ A & \mathbf{0} & B_2 & & \vdots \\ \vdots & \vdots & & \ddots & \mathbf{0} \\ A & \mathbf{0} & \dots & \mathbf{0} & B_n \end{pmatrix} \cdot \begin{pmatrix} x^{(0)} \\ x^{(1)} \\ \vdots \\ x^{(n)} \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \tag{♠}$$

with $A \in \mathbb{Z}_{\geq 0}^{1\times s}, B_i \in \mathbb{Z}_{\geq 0}^{1\times t}$ for all $i \in \{1, \dots, n\}$ and $b \in \mathbb{Z}_{\geq 0}^{n\times 1}$. Because all A matrices are the same and $r = 1$, we can preprocess the problem in two ways. First, we can eliminate all 0 entries in the A or B_i blocks, as they do not affect feasibility. Further, duplicate columns can be eliminated because we do not consider bounds on variables. Thus, we have $s, t \leq \Delta$. Two-stage ILPs of the form (♠) find application in, for example, the planning of electricity distribution [21]. Further applications are found in the mixing set problem [17], which plays an important role in production planning [37], and the response time computation problem [15]. Such two-stage ILPs can also be used to solve the bi-level knapsack problem [9, 12], wherein a leader (the first stage) defines the capacity of the knapsack and a follower (the second stage) aims to maximize the objective function for himself. Modelling these problems in the form of (♠) allows us to solve these problems in the running times discussed below.

Our contribution

In this paper, we provide an approach to solve restricted two-stage ILPs of the form (♠). We use combinatorial results, like the Frobenius problem, to reduce the dimensions of the matrix further. For the feasibility problem, this yields the following result:

▶ **Theorem 1.** *A feasible solution of the two-stage problem in form* (♠) *can be computed in time* $O(n \cdot \Delta^2 \log(\Delta))$.

When applied to ILPs of the form (♠), this result improves upon the current best known result by Cslovjecsek, Koutecký, Lassota, Pilipczuk and Polak [11] who provide a running time of $f(\max\{s, t\}, \max_i \|B_i\|_\infty) \cdot \|\mathcal{A}\|^{O(1)}$, where f is a computable function and $\|\mathcal{A}\|^{O(1)}$ denotes a polynomial in the input length of the problem. The function f is doubly exponential. We note here that this result, and others mentioned, solve a more general case of two-stage ILPs and are therefore not directly comparable as they cannot use the specific structure of (♠). All comparisons mentioned treat algorithms as being applied to ILPs of form (♠).

We further extend our results to find an optimal solution. Here, we show that optimal solutions for (♠) have a nice structure. This yields the following theorem:

▶ **Theorem 2.** *An optimal solution to a two-stage problem in form* (♠) *can be computed in time* $3^{O(\Delta)} \cdot \tilde{O}(n^2 \cdot \Delta^2)$ *or alternatively* $O(n \cdot 3^\Delta)$.

Here, the second running time comes as a result of applying Theorem 3, while the first is due to a known Graver bound stated by Klein [33]. This improves upon the best known running time of $2^{((s+t)\Delta)^{O(s^2t+st^2)}} \cdot n \log^{O(st)} n$ given in [10] for $r = 1$. We note that Eisenbrand and Rothvoss [18] recently extended the results of [11] to the optimality problem, yielding a running time dependent on a computable, but doubly exponential, function $f(\max\{s, t\}, \max_i \|B_i\|_\infty)$. Thus, their algorithm also runs in time $f(\max\{s, t\}, \max_i \|B_i\|_\infty) \cdot \|\mathcal{A}\|^{O(1)}$.

For the optimality case, a core piece of our analysis is an upper bound of the right hand side $\|b\|_\infty$, presented in the appendix.

Feasibility	Optimality
$g(s,t,\Delta)\cdot\|\mathcal{A}\|^{O(1)}$ [23](†)	$g(s,t,\Delta)\cdot\|\mathcal{A}\|^{O(1)}$ [23]
$(s\Delta)^{(s\Delta)^{O(s^2)}}\cdot\|\mathcal{A}\|^{O(1)}$ [33](†)	$(s\Delta)^{(s\Delta)^{O(s^2)}}\cdot\|\mathcal{A}\|^{O(1)}$ [33]
$3^{O((r+s)s^s(2r\Delta+1)rs)}\cdot\|\mathcal{A}\|^{O(1)}$ [26] (†)	$3^{O((r+s)s^s(2r\Delta+1)rs)}\cdot\|\mathcal{A}\|^{O(1)}$ [26]
$2^{((s+t)\Delta)^{O(s^2t+st^2)}}\cdot n\log^{O(st)} n$ [10] (†)	$2^{((s+t)\Delta)^{O(s^2t+st^2)}}\cdot n\log^{O(st)} n$ [10]
$f(\max\{s,t\},\max_i\|B_i\|_\infty)\cdot\|\mathcal{A}\|^{O(1)}$ [11] (‡)	$f(\max\{s,t\},\max_i\|B_i\|_\infty)\cdot\|\mathcal{A}\|^{O(1)}$ [18] (‡)
$O(n\cdot\Delta^2\log(\Delta))$ ($\diamond$)	$3^{O(\Delta)}\cdot\tilde{O}(n\cdot\Delta^2)$ ($\diamond$)

Table 1 A Table comparing running times of 2-stage stochastic ILP algorithms for $r=1$. Note that algorithms denoted with † are not explicitly given for feasibility but rather extended from the optimality algorithm. The function g is lower bounded by Ackermann's function. The algorithms denoted by ‡ allow for arbitrarily large entries in the first stage, i.e. the A_i bricks. Our result, $\diamond$, only covers the case of $r=1$ with $A_i=A$.

▶ **Theorem 3** (✂). *For problems of the form* (♠) *with* $s=1$*, we have* $\|b\|_\infty \leq lcm\{a, B_e^{(1)},\dots,B_e^{(n)}\}+2\Delta^2 \leq 3^\Delta+2\Delta^2$*, where* $B_e^{(i)} := B_j^{(i)} : \max\{c_j^{(i)}/B_j^{(i)} | j\in[t]\}$ *for all* $i\in[n]$.

Furthermore, we give the following lower bound for solving an optimality 2-stage problem with one row in form (♠), analogously to [24].

▶ **Corollary 4** (✂). *The 2-stage problem with one row in form* (♠) *cannot be solved in time less than* $2^{\delta\Delta^{1/2-\varepsilon}}$ *for some* $\delta,\varepsilon>0$ *assuming ETH where* Δ *is the largest entry of the 2-stage matrix* $\mathcal{A}$*.*

1.1 Related Work

Integer linear programming is shown to be NP-hard in the seminal work by Karp [31]. To solve general ILPs, two separate lines of research have formed. The first of these lines focuses on the case where m, i.e. the number of rows in the matrix A, is small. The first result was provided by Papadimitriou [36], who presented an algorithm running in time $O(n^{2m+2}\cdot(m\cdot\max\{\Delta,\|b\|_\infty\})^{(m+1)(2m+1)})$. Research with the focus on fixed values of m has continued, with the most recent result being given by Jansen and Rohwedder [27], who give an algorithm running in time $O(\sqrt{m}\Delta)^{(1+o(1))m}+O(nm)$. This running time is essentially tight under the SETH [27]. The algorithm of Jansen and Rohwedder improves upon an earlier result by Eisenbrand and Weismantel, who utilized the Steinitz Lemma to achieve an algorithm running in time $n\cdot O(m\Delta)^m\cdot\|b\|_1$ [19]. These algorithms all focus on a small/fixed number of rows m. Perhaps unsurprisingly, there is a parallel line of research, which focuses on a small amount of columns/variables, n. The first result in this line of research was provided by Lenstra in 1983 [30] who gave an algorithm running in time $2^{O(n^3)}\cdot\|\mathcal{A}\|^{O(1)}$, where $\|\mathcal{A}\|^{O(1)}$ denotes a polynomial in the binary encoding length of the ILP. The current state of the art is given by Reis and Rothvoss [39], who present an algorithm running in time $(\log(2n))^{O(n)}\cdot\|\mathcal{A}\|^{O(1)}$.

While the results given above tackle the general ILP formulation, there has also been significant research invested into structures of ILPs with specific structures. Of particular relevance to this work are two-stage stochastic ILPs. Research into these has been pioneered by Hemmecke and Schultz [23], who present an augmentation algorithm running in time $g(r,s,t,\Delta)\cdot|\mathcal{A}|^{O(1)}$, where g is a function lower bounded by Ackermann's function. This was subsequently improved upon by Klein [33], who gave an algorithm running in time $(rs\Delta)^{(rs\Delta)^{O(s^2r)}}\cdot\|\mathcal{A}\|^{O(1)}$. This was further improved by Jansen, Klein and Reuter [25], who

give an algorithm running in time $3^{O((r+s)s^s(2r\Delta+1)rs)} \cdot \|\mathcal{A}\|^{O(1)}$. Eisenbrand, Hunkeschröder, Klein, Koutecký, Levin and Onn provide two improved algorithms in [16] running in time $2^{(2\Delta)^{O(s^2t+st^2)}} \cdot \|\mathcal{A}\|^{O(1)}$, where the difference lies in the polynomial dependent on the input length. When considering the general parameterization of the largest entry, the most recent result is given by Cslovjecsek, Eisenbrand, Pilipczuk, Venzin and Weismantel [10], who provide an algorithm running in time $2^{(2\Delta)^{O(s(s+t))}} \cdot nr \log^{O(st)}(nr)$. Then, these results are complemented by a lower bound of $2^{2^{\delta(t+r)}}|I|^{O(1)}$ given by Jansen, Klein and Lassota [24]. They show that an algorithm solving two-stage ILPs in this running time contradicts the SETH. Recently, there have been advances made when considering matrices with large entries. First, Cslovjecsek, Koutecký, Lassota, Pilipczuk and Polak [11] showed that the feasibility problem for two-stage ILPs can be solved in time $f(\max\{s,t\}, \max_i \|B_i\|_\infty) \cdot \|\mathcal{A}\|$, where f is a doubly exponential function and $\|\mathcal{A}\|^{O(1)}$ denotes a polynomial in the input length of the problem. These results were very recently extended to the optimality problem by Eisenbrand and Rothvoss [18] who obtain a similar running time to to find an optimal solution.

Another relevant classical NP-hard problem to our work is that of knapsack, specifically the unbounded Knapsack problem. Like ILPs, it is one of Karp's original twenty one NP-hard problems [31]. There is a classical dynamic programming formulation to solve the knapsack problem given by Bellman [3]. With the problem being NP-hard, recent work is directed to improving the dependency on parameters like the maximum weight of a single item Δ. In this regime, the best known results for 0-1 knapsack are by Jin [28] and Bringmann [6], who each, independently of each other, give algorithms running in time $\tilde{O}(n+\Delta^2)$, where $\tilde{O}$ omits polylogarithmic factors. Bringmanns algorithm extends to the bounded knapsack case as well. In this work, we utilize results in the unbounded knapsack. The ILP algorithm given by Jansen and Rohwedder [27] directly yields an algorithm running in time $O(n+\Delta^2)$. Klein gives an algorithm solving the closely related subset sum problem in time $O(\Delta \log \Delta \log \frac{F}{\Delta})$, where F denotes the Frobenius number of the item weights. Jansen and Rohwedders algorithm solves subset sum in time $O(n+\Delta \log^2(\Delta))$. Solving a more general All-capacities unbounded knapsack problem, where we are not only interested in the solution for a given capacity C but rather all solutions in $[0, C]$, can be achieved with the algorithm by Chan and He [8]. They give an algorithm running in time $O(\Delta^2 \log \Delta + C)$ for this problem. We note that there have been other parameterizations studied when considering the knapsack problem, such as by Axiotis and Tzamos [1], who consider the amount of different weights in the knapsack instance, or Bringmann, Dürr and Polak [7], who consider the maximum profit of an item $p_{\max}$. Furthermore, Bateni, Hajiaghayi, Seddighin and Stein study the knapsack problem under the framework of predictions [2].

1.2 Overview

We first give some preliminary results and definitions required for the algorithms we present in Section 2. Then, in Section 3, we tackle the (simpler) feasibility case. Here, we present how to decide whether a solution to a problem of the form (♠) exists by careful case analysis, yielding Theorem 1. Finally, in Section 4, we show how to adapt and extend those techniques to find an optimal solution for problems of the form (♠). Here, we expand upon the ideas presented in Section 3 and solve more involved problems to yield the optimal solution.

Due to space constraints, some proofs were moved into the appendix. The corresponding statements are marked with (✂).

2 Preliminaries

We denote by $[n]$ the set $\{1,\dots,n\}$ for $n \in \mathbb{N}$ and $[n]_0 := [n] \cup \{0\}$. For solving the 2-stage IP with one row, we first need to define some problems that occur.

The *unbounded Equivalence Knapsack Problem (uEKP)* is the following knapsack problem: Given a capacity $C \in \mathbb{N}_0$ and a set of items I with weights w_i and profits p_i for all $i \in I$, determine a solution $x \in \mathbb{N}_0^{|I|}$ with maximum total profit $\max \sum_{i\in I} p_i x_i$ such that $\sum_{i\in I} w_i x_i = C$ if there exists such a solution. Here, we denote by $i_e \in I$ the most efficient item, i.e., $p_i/w_i \leq p_{i_e}/w_{i_e}, i \in I$.

A main observation regarding the structure of a solution of an unbounded Equivalence Knapsack Problem P gives the following theorem. It claims for every solution with capacity greater than $w_{\max}^2$ that there exists an optimal solution such that the most efficient item is part of it.

▶ **Theorem 5** (✂). *Let $k := \max\{j \mid \exists n \in \mathbb{N}_0 :\ C = j + n \cdot w_{i_e}, j \in [w_{\max}^2]_0\}$, let $n := (C-k)/w_{i_e}$ and let x^* be an optimal solution for the unbounded Equivalence Knapsack Problem with items I and capacity k. Let $x^T = (x_1,\dots,x_{|I|})$ with $x_i := \begin{cases} x_i^* + n & i = i_e \\ x_i^* & \text{otherwise} \end{cases}, i \in I$. If $C > w_{\max}^2$, then x is an optimal solution for P.*

For solving feasibility, we get the *unbounded Subset Sum Problem*: Given a target value $t \in \mathbb{N}_0$ and a set of items I with values $a_i, i \in I$, determine a solution $x \in \mathbb{N}_0^{|I|}$ such that $\sum_{i\in I} a_i x_i = t$. Thus, an unbounded Subset Sum Problem instance is an unbounded Equivalence Knapsack Problem instance with $p_i = 0, i \in I$.

Let f be a polynomial function with integer coefficients. A *diophantine equation* is of the form $f(x_1,\dots,x_n) = 0$. We search for a solution $x \in \mathbb{Z}^n$ such that $f(x_1,\dots,x_n) = 0$ holds. We say that a value $x_1 \in \mathbb{Z}$ *solves* a diophantine equation f if there are appropriate values $x_2,\dots,x_n \in \mathbb{Z}$ such that $f(x_1,\dots,x_n) = 0$ holds. Let Δ be the biggest coefficient of a diophantine equation. The following theorem is a well-known theorem about the solutions of a linear diophantine equation:

▶ **Theorem 6** ([38], ✂). *Let $ax + by = c$ be a diophantine equation with $a,b,c \in \mathbb{Z}, a,b \neq 0$. This equation has an integral solution $(x,y); x,y \in \mathbb{Z}$ if and only if c is a multiple of the greatest common divisor of a and b. If (x,y) is a solution of the diophantine equation, then all other solutions are of the form $(x + kv, y - ku)$ with arbitrary $k \in \mathbb{Z}$, $u := a/\gcd(a,b)$ and $v := b/\gcd(a,b)$.*

A system of linear diophantine equations can be solved in polynomial time by computing the Hermite Normal Form [40]. Malaschonok uses different approaches to solve a system of linear diophantine equations of size $n \times n$ in time $O(n^3)$ [35]. Since we assume $\Delta \ll n$ and our system of linear diophantine equations has 2-stage form, we give in the appendix an algorithm which implies the following theorem:

▶ **Theorem 7** (✂). *A system of linear diophantine equations can be solved in time $O(n \cdot \Delta)$.*

If $\Delta \ll n$ doesn't hold, it could be better to use a known algorithm e.g. [35] for solving a system of linear diophantine equations. However, the overall running time of our algorithms wouldn't significantly change.

Another theorem we need for the analysis of our algorithm for the optimality case is the following theorem. It proves an upper bound of the right hand side $\|b\|_\infty$.

▶ **Theorem 3** (✂). *For problems of the form (♠) with $s = 1$, we have $\|b\|_\infty \leq lcm\{a, B_e^{(1)},\dots,B_e^{(n)}\} + 2\Delta^2 \leq 3^\Delta + 2\Delta^2$, where $B_e^{(i)} := B_j^{(i)} : \max\{c_j^{(i)}/B_j^{(i)} | j \in [t]\}$ for all $i \in [n]$.*

As a result of this, our algorithm only needs to compute solutions with a right-hand side at most $\operatorname{lcm}\{a, B_e^{(1)}, \ldots, B_e^{(n)}\} + 2\Delta^2$.

3 Feasibility

In this section, we present the proof of Theorem 1. We split the analysis into three cases, dependent on the size of the first stage ($Ax^{(0)}$) or the smallest second stage ($B_i x^{(i)}$) in a feasible solution. Somewhat surprisingly, the running time of Theorem 1 is dominated by the small sized first or second stages. We can utilize insights from the Frobenius problem to solve the large case more efficiently.

Here, we consider the feasibility 2-stage problem with one row. Consider a target value t such that $Ax^{(0)} = t$. Then, a solution for the feasibility problem (♠) is $x^T = (x^{(0)}, \ldots, x^{(n)})$ where $x^{(0)}$ is a solution for the unbounded Subset Sum Problem $Ax^{(0)} = t$ and $x^{(i)}$ is a solution for the unbounded Subset Sum Problem $B_i x^{(i)} = b_i - t$. In order to solve the feasibility 2-stage IP with one row, we considered different approaches such as residue tables. For more details see the appendix. Here, we use a case distinction to get in every case only few possible target values t. For the solution x we get the following three cases:

1. $Ax^{(0)} \leq \Delta^2$
2. $B_i x^{(i)} \leq \Delta^2$ holds for at least one $i \in [n]$
3. $Ax^{(0)} > \Delta^2$ and $B_i x^{(i)} > \Delta^2$, $i \in [n]$

Case 1: $Ax^{(0)} \leq \Delta^2$: Here, the contribution of the first stage is small. The core observation we use is that we can enumerate all possible ways to reach these values and check, for each reachable value, whether a solution for that value exists. We compute, for all target values up to Δ^2, of the unbounded Subset Sum Problem and decide whether there exists a solution of size A and save the result in a list ℓ_A. For all $j \in [\Delta^2]_0$, we have $\ell_A[j] = 1$ if there is a solution for the unbounded Subset Sum Problem $Ax^{(0)} = j$ and $\ell_A[j] = 0$ otherwise. Analogously, for all $i \in [n]$, we compute for all target values in $[b_i - \Delta^2, b_i]$ for the unbounded Subset Sum Problem with values B_i if there exists a solution and save the result in lists ℓ_{B_i}. For all $i \in [n]$ and $j \in [\Delta^2]_0$, we have $\ell_{B_i}[j] = 1$ if there is a solution for the unbounded Subset Sum Problem $B_i x^{(i)} = b_i - j$ and $\ell_{B_i}[j] = 0$ otherwise. We can compute all these lists in time $O(n \cdot \Delta^2 \log(\Delta))$ by Lemma 15.

Since $Ax^{(0)} \leq \Delta^2$, there exists a feasible solution if and only if there exists a $j \in [\Delta^2]_0$ such that all entries $\ell_{B_i}[j] = 1$ for all $i \in [n]$ and the entry $\ell_A[j] = 1$. Thus, we check all these entries for every $j \in [\Delta^2]_0$. If all these entries are equal to 1, then compute concrete solutions for the unbounded Subset Sum Problem $Ax^{(0)} = j$ and for the unbounded Subset Sum Problems $B_i x^{(i)} = b_i - j$ for all $i \in [n]$. Otherwise, in the end, return that there exists no solution. This step takes $O(\Delta^2 \cdot n)$ time to check if there exists a suitable j and $O(n \cdot \Delta \log^2(\Delta))$ time to compute concrete solutions for all unbounded Subset Sum Problems by Jansen and Rohwedder [27].

All together, we can solve the feasibility problem (♠) in time $O(n \cdot \Delta^2 \log(\Delta))$.

Case 2: $B_i x^{(i)} \leq \Delta^2$ holds for at least one $i \in [n]$: In this case, the contribution of the second stage is small for at least one block B_i. We observe that we can determine this block and use this to bound the contribution of the A blocks. Then, we proceed similar to the first case. For $b_m := \min_{i \in [n]} b_i$, we get $B_m x^{(m)} \leq \Delta^2$. Otherwise, we would have $b_i \geq b_m \Leftrightarrow Ax^{(0)} + B_i x^{(i)} \geq Ax^{(0)} + B_m x^{(m)} \Leftrightarrow B_i x^{(i)} \geq B_m x^{(m)}$ and with this, $B_i x^{(i)} \geq B_m x^{(m)} > \Delta^2$ for all $i \in [n]$, a contradiction. With $Ax^{(0)} + B_m x^{(m)} = b_m \Leftrightarrow Ax^{(0)} = b_m - B_m x^{(m)} \geq b_m - \Delta^2$, we get $Ax^{(0)} \in [b_m - \Delta^2, b_m]$. So, we compute, for all

target values in $[b_m - \Delta^2, b_m]$, of the unbounded Subset Sum Problem and decide whether there exists a solution of size A and save the result in a list ℓ_A. For all $j \in [\Delta^2]_0$, we have $\ell_A[j] = 1$ if there is a solution for the unbounded Subset Sum Problem $Ax^{(0)} = b_m - j$ and $\ell_A[j] = 0$ otherwise. Analogously, for all $i \in [n]$, we compute for all target values in $[b_i - b_m, b_i - b_m + \Delta^2]$ for the unbounded Subset Sum Problem with values B_i if there exists a solution and save the result in lists ℓ_{B_i}. For all $i \in [n]$ and $j \in [\Delta^2]_0$, we have $\ell_{B_i}[j] = 1$ if there is a solution for the unbounded Subset Sum Problem $B_i x^{(i)} = b_i - b_m + j$ and $\ell_{B_i}[j] = 0$ otherwise. We can compute all these lists in time $O(n \cdot \Delta^2 \log(\Delta))$ by Lemma 15.

Since $B_m x^{(m)} \leq \Delta^2$, there exists a feasible solution if and only if there exists a $j \in [\Delta^2]_0$ such that all entries $\ell_{B_i}[j] = 1$ for all $i \in [n]$ and the entry $\ell_A[j] = 1$. Thus, we check for every $j \in [\Delta^2]_0$ all these entries. If all these entries are equal to 1, then compute concrete solutions for the unbounded Subset Sum Problem $Ax^{(0)} = b_m - j$ and for the unbounded Subset Sum Problems $B_i x^{(i)} = b_i - b_m + j$ for all $i \in [n]$. Otherwise, in the end, return that there exists no solution. This step takes $O(\Delta^2 \cdot n)$ time to check if there exists a suitable j and $O(n \cdot \Delta \log^2(\Delta))$ time to compute concrete solutions for all unbounded Subset Sum Problems by Jansen and Rohwedder [27].

All together, we can solve the feasibility problem (♠) in time $O(n \cdot \Delta^2 \log(\Delta))$.

Case 3: $Ax^{(0)} > \Delta^2$ and $B_i x^{(i)} > \Delta^2$, $i \in [n]$: Consider every block individually: Here, the contribution of all B_i and A blocks is large. We can use the following combinatorial results to reduce the problem to a diophantine equation which we can solve more quickly. For $A = (a_0, \ldots, a_{s-1})$, we get the equation $a_0 x_0^{(0)} + \ldots + a_{s-1} x_{s-1}^{(0)} = p$ and for $B_i = (d_{i_0}, \ldots, d_{i_{t-1}})$, we get the equation $d_{i_0} x_0^{(i)} + \ldots + d_{i_{t-1}} x_{t-1}^{(i)} = b_i - p$ for all $i \in [n]$ for a specific $p > \Delta^2$. Since $p > \Delta^2$ as well as $b_i - p > \Delta^2$, we get that these equations have a solution if and only if p is a multiple of the greatest common divisor $\gcd(A) = \gcd(a_0, \ldots, a_{s-1})$ and $b_i - p$ is a multiple of the greatest common divisor $\gcd(B_i) = \gcd(d_{i_0}, \ldots, d_{i_{t-1}})$ for all $i \in [n]$ by Frobenius [32], analogously to Theorem 6. Applying the euclidean algorithm Δ times takes $O(\Delta \log(\Delta))$ time and computes the greatest common divisor of Δ natural numbers smaller than $\Delta + 1$. Thus, we can transform the matrix into the following matrix in $O(n \cdot \Delta \log(\Delta))$ time:

$$\begin{pmatrix} \gcd(A) & \gcd(B_1) & 0 & \ldots & 0 \\ \gcd(A) & 0 & \gcd(B_2) & & \vdots \\ \vdots & \vdots & & \ddots & 0 \\ \gcd(A) & 0 & \ldots & 0 & \gcd(B_n) \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \tag{3}$$

Now, this problem is a diophantine equation system with equations $\gcd(A) \cdot x_0 + \gcd(B_i) \cdot x_i = b_i$ for all $i \in [n]$. We can solve this system with Algorithm 1 in time $O(n \cdot \Delta)$ by Theorem 7. If the algorithm does not find a solution, then there exists no solution by Lemma 10. Otherwise, let $(x_0, \ldots, x_n)$ be the solution of Algorithm 1 and thus, by Lemma 10 a solution for the diophantine equation system. It does not have to hold that $x_i \geq 0$ for all $i \in [n]_0$. In order to get a solution which satisfies this condition, we can go on as follows:

For every equation $i \in [n]$ the value (x_0, x_i) is a solution and by Theorem 6 all other solutions are of the form $(x_0 + k_i v_i, x_i - k_i u_i)$ with arbitrary $k_i \in \mathbb{Z}$, $u_i := {}^{\gcd(A)}/_{\gcd(\gcd(A),\gcd(B_i))}$ and $v_i := {}^{\gcd(B_i)}/_{\gcd(\gcd(A),\gcd(B_i))}$. Thus, all solutions for problem (3) are of the form

$z := (x_0 + k \cdot \operatorname{lcm}(v_1, \ldots, v_n), x_1 - {}^{k \cdot \operatorname{lcm}(v_1,\ldots,v_n)}/_{v_1} \cdot u_1, \ldots, x_n - {}^{k \cdot \operatorname{lcm}(v_1,\ldots,v_n)}/_{v_n} \cdot u_n)$

for arbitrary $k \in \mathbb{Z}$. So, we know how all feasible solutions of problem (♠) look like. Now, we consider the solution with the smallest first component.

Since $Ax^{(0)} > \Delta^2$ and $B_i x^{(i)} > \Delta^2$ for all $i \in [n]$, a feasible solution z for problem (3) always satisfies $z_0 > {}^{\Delta^2}/_{\gcd(A)}$ and $z_i > {}^{\Delta^2}/_{\gcd(B_i)}$ for all $i \in [n]$. Thus,

$$z_\ell := (x_0 - k \cdot \operatorname{lcm}(v_1, \dots, v_n), x_1 + \frac{k \cdot \operatorname{lcm}(v_1, \dots, v_n)}{v_1} \cdot u_1, \dots, x_n + \frac{k \cdot \operatorname{lcm}(v_1, \dots, v_n)}{v_n} \cdot u_n)$$

$$\text{with } k := \left\lceil \frac{x_0 - \frac{\Delta^2}{\gcd(A)}}{\operatorname{lcm}(v_1, \dots, v_n)} \right\rceil - 1 \in \mathbb{Z}$$

is the solution with the smallest first component z_{ℓ_0}. For a proof of this statement, see the appendix. If z_ℓ is no valid solution, then return that there exists no feasible solution. Otherwise, compute concrete solutions for the unbounded Subset Sum Problem $Ax^{(0)} = \gcd(A) z_{\ell_0}$ and for the unbounded Subset Sum Problems $B_i x^{(i)} = \gcd(B_i) z_{\ell_i}$. This takes $O(n \cdot \Delta \log^2(\Delta))$ time by Jansen and Rohwedder [27].

All together, we can solve the feasibility problem (♠) in time $O(n \cdot \Delta \log^2(\Delta))$.

▶ **Theorem 1.** *A feasible solution of the two-stage problem in form (♠) can be computed in time $O(n \cdot \Delta^2 \log(\Delta))$.*

Proof. We solve the feasibility problem of the form (♠) as described in the three cases. First, we assume that case 1 holds and search for a solution as written in case 1. If there exists a feasible solution, we return it. Otherwise, we assume case 2 holds and search for a solution as written in case 2. If there exists a feasible solution, we return it. Otherwise, we assume case 3 holds and search for a solution as written in case 3. If there exists a feasible solution, we return it. Otherwise, we return that there exists no solution. Both case 1 and case 2 take $O(n \cdot \Delta^2 \log(\Delta))$ time. Case 3 takes $O(n \cdot \Delta \log^2(\Delta))$ time. Thus, we obtain a computational complexity of $O(n \cdot \Delta^2 \log(\Delta))$ for solving feasibility 2-stage IP with one row. ◀

4 Optimality

In this section, we show how to adapt the techniques presented in Section 3 to find an optimal solution to the two-stage ILPs of form (♠). Once again, we distinguish cases based upon the sizes of the first and second stages in an optimal solution. Contrary to the, arguably simpler, feasibility case, the running time of Theorem 2 is dominated by the case where both the first and second stage have large values in the optimal solution. The techniques utilizing the Frobenius problem do not translate to the optimality problem. To remedy this, we limit the search space for optimal solutions based on the coefficients of bricks in the cost function c.

4.1 Case Distinction

Let P be a 2-stage IP with one row of the form (♠) and let $c^T = (c^{(0)}, \dots, c^{(n)})$ be the cost vector of P. Consider a capacity C such that $Ax^{(0)} = C$. Then, a solution for P is $x^T = (x^{(0)}, \dots, x^{(n)})$ where $x^{(0)}$ is an optimal solution for the unbounded Equivalence Knapsack Problem $Ax^{(0)} = C$ with profits $c^{(0)}$ and $x^{(i)}$ is an optimal solution for the unbounded Equivalence Knapsack Problem $B_i x^{(i)} = b_i - C$ with profits $c^{(i)}$. In order to solve P, we use a case distinction to get in every case only few possible capacity values C. For the solution $x \in {\mathbb{N}_0}^{s+n \cdot t}$ we get the following three cases:

1. $Ax^{(0)} \leq \Delta^2$ (✂)
2. $B_i x^{(i)} \leq \Delta^2$ holds for at least one $i \in [n]$ (✂)

3. $Ax^{(0)} > \Delta^2$ and $B_i x^{(i)} > \Delta^2$, $i \in [n]$

Due to space constraints, the cases 1 and 2 were moved into the appendix. In both cases the contribution for at least one block is small. Thus, we can limit the search space for optimal solutions and can check all of these efficiently, analogously to the solution of the feasibility problem.

Case 3: $Ax^{(0)} > \Delta^2$ and $B_i x^{(i)} > \Delta^2$, $i \in [n]$: This case is the most involved one. Once again, we can transform the problem similar to the feasibility case. However, the restrictions placed upon us by looking for an optimal solution further complicate this. We distinguish based upon the efficiency of the most efficient columns to limit the search spaces for optimal solutions. If there is no significant difference in efficiency, we use a Graver bound on optimal solutions to upper bound the search space.

Here, we get the same transformation by Frobenius [32] as in case 3 of the feasibility problem. Thus, consider we are at the point where we computed z_ℓ. Analogous to z_ℓ,

$$z_r := (x_0 + k_r \cdot \operatorname{lcm}(v_1, \dots, v_n), x_1 - \frac{k_r \cdot \operatorname{lcm}(v_1, \dots, v_n)}{v_1} \cdot u_1, \dots, x_n - \frac{k_r \cdot \operatorname{lcm}(v_1, \dots, v_n)}{v_n} \cdot u_n)$$

$$\text{with } k_r := \min_{1 \le i \le n} \left\{ \left\lceil \frac{x_i - \frac{\Delta^2}{\gcd(B_i)}}{\frac{\operatorname{lcm}(v_1, \dots, v_n)}{v_i} \cdot u_i} \right\rceil - 1 \right\}$$

is the solution with the biggest first component z_{r_0}.

Problem (♠) represents two EKPs in every row. One EKP P_A has capacity $C \in \mathbb{N}$, the entries of the matrix A as weights of the items and the corresponding entries $c_0, \dots, c_{s-1}$ of the cost vector as profits of the items. The second EKP P_{B_i} in each row $i \in [n]$ has capacity $b_i - C \in \mathbb{N}$, the entries of the matrix B_i as weights of the items and the corresponding entries $c_{s+(i-1)\cdot t}, \dots, c_{s+i\cdot t-1}$ of the cost vector as profits of the items. Let I_{e_A} be the most efficient item of A and respectively, let $I_{e_{B_i}}$ be the most efficient item of B_i for all $i \in [n]$.

By the feasible solution, we get the capacity $C = z_0 \cdot \gcd(A)$. Since $Ax^{(0)} > \Delta^2$ and $B_i x^{(i)} > \Delta^2$ for all $i \in [n]$, an optimal solution for an EKP consists of an optimal solution for the EKP with capacity $k \in \mathbb{N}_0$ for $k \in [\Delta^2 - \Delta, \Delta^2]$ and of $q \in \mathbb{N}_0$ copies of the most efficient item by Theorem 5.

Let $C \in \mathbb{N}$ be the capacity of EKP P_A such that problem (♠) is optimally solved. Let `optx` be an optimal solution for problem (♠) and let `opta` be an optimal solution for this problem P_A. Then, `opta` consists of an optimal solution of the EKP with capacity $k_a \in [\Delta^2 - \Delta, \Delta^2]$ and of $q \in \mathbb{N}_0$ copies of item I_{e_A}. We guess the capacity k_a with expense of $O(\Delta)$ factor and take it for granted. We calculate a solution of the uEKP with capacity k_a in $O(s + \Delta^2)$ time by Jansen and Rohwedder [27]. Now, we only need to find the right amount q of copies of item I_{e_A} and with this the correct capacity C. According to the ratio between the most efficient items of the B blocks and the most efficient item of the A block, we start from z_ℓ or z_r and search in a specific range for the correct number q and thus, for an optimal solution. The following Figure 1 sketches the introduced variables and the case distinction.

Case 3.1: $2 \cdot \sum_{i \in [n]} {}^{p_{I_{e_{B_i}}}}/_{w_{I_{e_{B_i}}}} < {}^{p_{I_{e_A}}}/_{w_{I_{e_A}}}$: So, the most efficient item of A is much more efficient than the most efficient items of the matrices $B_i, i \in [n]$ together. In order to obtain a best possible solution, we should take as much as possible from A. Thus, the optimal solution has a high value $\texttt{optx}^{(0)}$. So, we can go on like this: For every $Ax^{(0)} \in [\gcd(A)(z_{r_0} - \Delta^2 \cdot \operatorname{lcm}(v_1, \dots, v_n)), \gcd(A) z_{r_0}]$ with $Ax^{(0)} = \gcd(A)(x_0 + k \cdot \operatorname{lcm}(v_1, \dots, v_n))$ for $k \in \mathbb{Z}$ solve the IPs as described in case 1. Since we have at most Δ^2 many capacities

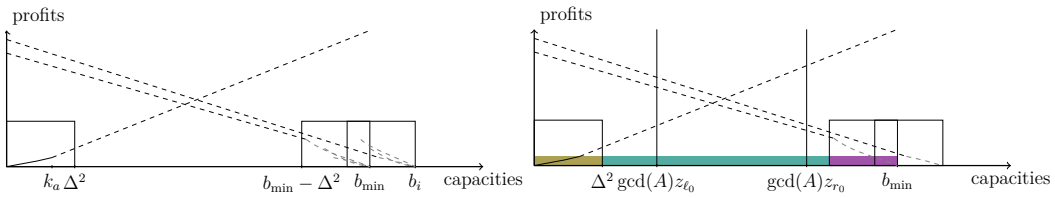

Figure 1 The **left** figure shows the 2-stage problem with one row of the form (♠). For the uEKP $Ax^{(0)} = C$ we already know the capacity k_a. For the uEKPs $B_i x^{(i)} = b_i - C$ are more values k_{B_i} possible. Starting from k_a, we add the most efficient item I_{e_A} to reach the capacity C. Analogously, we add the most efficient item $I_{e_{B_i}}$ starting from some point k_{B_i} for every $i \in [n]$. Consider a line parallel to the profit-axis. If the dashed straight line which represents I_{e_A} hits this line and if for every $i \in [n]$ exists a start point k_{B_i} such that the dashed straight line which represents $I_{e_{B_i}}$ hits this line, then this line corresponds to a feasible solution. The intersection point of this line with the capacities-axis is the capacity C. The sum of all profit values of the intersections of this line with the dashed straight lines is the maximum total profit. The **right** figure shows the three cases. The brown part shows the search range for case 1, the purple part shows the search range for case 2 and the green part shows the search range for case 3. The lines in the green part represent the solution with the smallest first component z_ℓ and the solution with the biggest first component z_r.

$Ax^{(0)}$, this takes $O(n \cdot \Delta^2 \log(\Delta))$ time, analogous to case 1. The solution we get from this is already an optimal solution for problem (♠):

▷ Claim 8 (✂). We have $A \cdot \texttt{optx}^{(0)} \in [\gcd(A)(z_{r_0} - \Delta^2 \cdot \operatorname{lcm}(v_1, \ldots, v_n)), \gcd(A)z_{r_0}]$.

We prove this claim by contradiction.

Case 3.2: $\gcd(A) \cdot \operatorname{lcm}(v_1, \ldots, v_n) \neq 1 \wedge$
$\sum_{i\in[n]} {}^{p_{I_{e_{B_i}}}}\!/_{w_{I_{e_{B_i}}}} \geq {}^{\gcd(A)\cdot\operatorname{lcm}(v_1,\ldots,v_n)}\!/_{(\gcd(A)\cdot\operatorname{lcm}(v_1,\ldots,v_n)-1)} \cdot {}^{p_{I_{e_A}}}\!/_{w_{I_{e_A}}}$: So, the most efficient items of the matrices $B_i, i \in [n]$ together are much more efficient than the most efficient item of A. In order to obtain a best possible solution, we should take as much as possible from the $B_i, i \in [n]$ blocks. Thus, the optimal solution has a small value $\texttt{optx}^{(0)}$. So, we can go on like this: For every $Ax^{(0)} \in [\gcd(A)z_{\ell_0}, \gcd(A)(z_{\ell_0} + \Delta^2 \cdot \operatorname{lcm}(v_1, \ldots, v_n))]$ with $Ax^{(0)} = \gcd(A)(x_0 + k \cdot \operatorname{lcm}(v_1, \ldots, v_n))$ for $k \in \mathbb{Z}$ solve the IPs as described in case 1. Since we have at most Δ^2 many capacities $Ax^{(0)}$, this takes $O(n \cdot \Delta^2 \log(\Delta))$ time, analogous to case 1. The solution we get from this is already an optimal solution for problem (♠):

▷ Claim 9 (✂). We have $A \cdot \texttt{optx}^{(0)} \in [\gcd(A)z_{\ell_0}, \gcd(A)(z_{\ell_0} + \Delta^2 \cdot \operatorname{lcm}(v_1, \ldots, v_n))]$.

We prove this claim by contradiction, analogously to Claim 8.

Case 3.3: $2 \cdot \sum_{i\in[n]} {}^{p_{I_{e_{B_i}}}}\!/_{w_{I_{e_{B_i}}}} \geq {}^{p_{I_{e_A}}}\!/_{w_{I_{e_A}}} \wedge$
$\gcd(A) \cdot \operatorname{lcm}(v_1, \ldots, v_n) \cdot {}^{p_{I_{e_A}}}\!/_{w_{I_{e_A}}} > (\gcd(A) \cdot \operatorname{lcm}(v_1, \ldots, v_n) - 1) \cdot \sum_{i\in[n]} {}^{p_{I_{e_{B_i}}}}\!/_{w_{I_{e_{B_i}}}}$:
Because the A-part as well as the B_i-parts for all $i \in [n]$ are uEKPs, we know that the solution consists of some optimal solution for the EKP with capacity $k_a, k_{B_i} \leq \Delta^2$ and some copies of the most efficient item. Since we guessed k_a and already computed an optimal solution of the uEKP with capacity k_a, we know the values $x_i^{(0)}$ for $i \neq I_{e_A}$. Thus, we can transform problem (♠) into the following 2-stage problem with one row:

$$\begin{pmatrix} w_{I_{e_A}} & B_1 & \mathbf{0} & \ldots & \mathbf{0} \\ w_{I_{e_A}} & \mathbf{0} & B_2 & & \vdots \\ \vdots & \vdots & & \ddots & \mathbf{0} \\ w_{I_{e_A}} & \mathbf{0} & \ldots & \mathbf{0} & B_n \end{pmatrix} \cdot \begin{pmatrix} x_{I_{e_A}}^{(0)} \\ x^{(1)} \\ \vdots \\ x^{(n)} \end{pmatrix} = \begin{pmatrix} b_1 - k_a \\ \vdots \\ b_n - k_a \end{pmatrix} \tag{4}$$

We get $c'^T = (c^{(0)}_{I_{e_A}}, c^{(1)}, \dots, c^{(n)})$ as corresponding cost vector. Moreover, we have an initial solution z_l or z_r for problem (3) and can compute a feasible solution to problem (♠) and with this also to problem (4) in time $O(n \cdot \Delta^2 \log(\Delta))$. So we need to solve a 2-stage IP with one row and already have a feasible solution. Thus, we can use for example the augmentation algorithm by Klein [33]. According to the ratio between the most efficient item in the A-part and the sum of the most efficient items in the B-parts we have two more cases:

Case 3.3.1: $\sum_{i\in[n]} {}^{p_{I_{e_{B_i}}}}/_{w_{I_{e_{B_i}}}} \geq {}^{p_{I_{e_A}}}/_{w_{I_{e_A}}}$: Since the most efficient items of the matrices $B_i, i \in [n]$ together are more efficient than the most efficient item of A, the optimal solution has a small value $\texttt{optx}^{(0)}$. We already know k_a and that $k_{B_i} \in [\Delta^2 - \Delta, \Delta^2]$. By pigeonhole, we get the same combination of k_{B_i} values after checking Δ^n successive solutions. Together, we get that the value $A \cdot \texttt{optx}^{(0)} \in [\gcd(A)z_{l_0}, \gcd(A)(z_{l_0} + \Delta^n \cdot \operatorname{lcm}(v_1, \dots, v_n))]$. Thus, we can derive the following bounds for problem (4) (see Figure 1): The lower bound of our problem is $l := (l^{(0)}_{I_{e_A}}, l^{(1)}, \dots, l^{(n)})$ with

- $l^{(0)}_{I_{e_A}} = \frac{\gcd(A)z_{l_0} - k_a}{w_{I_{e_A}}}$
- $l^{(i)}_{I_{e_{B_i}}} = \left\lceil \frac{b_i - \gcd(A)(z_{l_0} + \Delta^n \cdot \operatorname{lcm}(v_1,\dots,v_n)) - \Delta^2}{w_{I_{e_{B_i}}}} \right\rceil$ and $l^{(i)}_j = 0$ for all $j \neq I_{e_{B_i}} \in [t-1]_0, i \in [n]$

since $x \in \mathbb{N}_0$. The upper bound of our problem is $u := (u^{(0)}_{I_{e_A}}, u^{(1)}, \dots, u^{(n)})$ with

- $u^{(0)}_{I_{e_A}} = \left\lfloor \frac{\gcd(A)(z_{l_0} + \Delta^n \cdot \operatorname{lcm}(v_1,\dots,v_n))}{w_{I_{e_A}}} \right\rfloor$
- $u^{(i)}_{I_{e_{B_i}}} = \left\lfloor \frac{b_i - \gcd(A)z_{l_0}}{w_{I_{e_{B_i}}}} \right\rfloor$ and $u^{(i)}_j = w_{I_{e_{B_i}}}$ for all $j \neq I_{e_{B_i}} \in [t-1]_0, i \in [n]$

We have $u^{(i)}_j = w_{I_{e_{B_i}}}$ for all $j \neq I_{e_{B_i}} \in [t-1]_0, i \in [n]$. If $x^{(i)}_j > w_{I_{e_{B_i}}}$ would hold, we could replace $w_{I_{e_{B_i}}}$ many items $I_{j_{B_i}}$ by $w_{I_{j_{B_i}}}$ many items $I_{e_{B_i}}$ and would get a higher profit, a contradiction. So,

$$\max_{i\in[1+n\cdot t]} \{u_i - l_i\} \leq \frac{\gcd(A) \cdot \Delta^n \cdot \operatorname{lcm}(v_1, \dots, v_n) + \Delta^2}{\min\{w_{I_{e_A}}, w_{I_{e_{B_i}}}\}} \leq \Delta^{n+1} \cdot \operatorname{lcm}(v_1, \dots, v_n) + \Delta^2.$$

In the augmenting algorithm of Klein [33] we compute one augmenting step for every power of two λ with $\lambda \leq \lfloor \log(\max_{i\in[1+n\cdot t]} \{u_i - l_i\}) \rfloor \leq \lfloor \log(\Delta^{n+1} \cdot \operatorname{lcm}(v_1, \dots, v_n) + \Delta^2) \rfloor$. In one augmenting step we compute the optimal solution of our 2-stage IP with one row (4) for every $\|y^0\|_1 \leq L$ and return the one with the best objective value. Here, $L := \|\mathcal{G}\|_1$ where $\mathcal{G}$ is the size of the Graver basis. Jansen, Klein and Reuter showed that $L \leq 3^{O(s^s(2r\Delta+1)^{rs})}$ [26]. In the appendix we show how to adapt a bound by Klein [33] yielding a similar result for our case. Computing an optimal solution takes $O(n + t \cdot \Delta)$ time and finding the one with the best objective value takes $O(s + n \cdot t)$ time. Since $\operatorname{lcm}(v_1, \dots, v_n) + 1 \leq \Delta^n$, we get:

$$\begin{aligned}
&\lfloor \log(\Delta^{n+1} \cdot \operatorname{lcm}(v_1, \dots, v_n) + \Delta^2) \rfloor \\
\leq\ & \log(\Delta^{n+1} \cdot \operatorname{lcm}(v_1, \dots, v_n) + \Delta^{n+1}) \\
=\ & \log(\Delta^{n+1} \cdot (\operatorname{lcm}(v_1, \dots, v_n) + 1)) \\
\leq\ & \log(\Delta^{n+1} \cdot \Delta^n) \\
=\ & (2n+1) \cdot \log(\Delta)
\end{aligned}$$

In total we get the following running time, because $s = r = 1$ and $\Delta \ll n$ for problem (4):

$$\begin{aligned}
&\underbrace{\lfloor \log(\Delta^{n+1} \cdot \operatorname{lcm}(v_1, \dots, v_n) + \Delta^2) \rfloor}_{\text{for every } \lambda} \cdot (\underbrace{3^{O(s^s(2r\Delta+1)^{rs})}}_{\text{for every } \|y^0\|_1 \leq L} \cdot \underbrace{(O(n + t \cdot \Delta) + O(s + n \cdot t))}_{\text{optimal solution}}) \\
\leq\ & ((2n+1) \cdot \log(\Delta)) \cdot 3^{O(\Delta)} \cdot (O(n + t \cdot \Delta) + O(n \cdot t)) \\
\leq\ & 3^{O(\Delta)} \cdot \tilde{O}(n^2 \cdot \Delta)
\end{aligned}$$

If $\Delta \ll n$ doesn't hold, we get a running time of $3^{O(\Delta)} \cdot \tilde{O}(\Delta^2)$. Alternatively, we use Theorem 3 in order to limit the search space for the value $A \cdot \texttt{optx}^{(0)}$. With this, we get $A \cdot \texttt{optx}^{(0)} \in [\gcd(A)z_{\ell_0}, \gcd(A) \cdot (z_{\ell_0} + \operatorname{lcm}\{w_{I_{e_A}}, w_{I_{e_{B_1}}}, \ldots, w_{I_{e_{B_n}}}\})]$. So, we can go on like this: For every $Ax^{(0)} \in [\gcd(A)z_{\ell_0}, \gcd(A) \cdot (z_{\ell_0} + \operatorname{lcm}\{w_{I_{e_A}}, w_{I_{e_{B_1}}}, \ldots, w_{I_{e_{B_n}}}\})]$ with $Ax^{(0)} = \gcd(A)(x_0 + k \cdot \operatorname{lcm}(v_1, \ldots, v_n))$ for $k \in \mathbb{Z}$ solve the IPs as described in case 1. Since we have at most 3^Δ many capacities $Ax^{(0)}$, this takes $O(n \cdot (\Delta^2 \log(\Delta) + 3^\Delta)) = O(n \cdot 3^\Delta)$ time, analogous to case 1.

Case 3.3.2(✂): $\sum_{i \in [n]} {}^{p_{I_{e_{B_i}}}}/_{w_{I_{e_{B_i}}}} < {}^{p_{I_{e_A}}}/_{w_{I_{e_A}}}$: This case is analogous to case 3.3.1.

4.2 Algorithm for 2-stage IP with one row

The algorithm to solve the problem (♠) processes the three cases as written in Section 4.1. For the pseudocode version see the appendix. The following theorem summarizes the proofs.

▶ **Theorem 2.** *An optimal solution to a two-stage problem in form* (♠) *can be computed in time* $3^{O(\Delta)} \cdot \tilde{O}(n^2 \cdot \Delta^2)$ *or alternatively* $O(n \cdot 3^\Delta)$.

Proof. The Algorithm 2 works exactly like the case distinction in Section 4.1. So, Algorithm 2 computes an optimal solution for a 2-stage problem with one row. We get the running time of Algorithm 2 by the running time of individual cases from Section 4.1. The lines 1 and 3 take $O(n \cdot \Delta^2 \log(\Delta))$ time each. The lines 7 and 9 take $O(n \cdot \Delta^2 \log(\Delta))$ time each. Line 11 takes $3^{O(\Delta)} \cdot \tilde{O}(n^2 \cdot \Delta)$ time, line 2 takes $O(n)$ time, line 14 takes $O(\Delta^2 \cdot (s + n \cdot t))$ time and line 4 takes $O(n \cdot \Delta \log(\Delta))$ time. Since the lines $6 - 12$ get executed $\Delta + 1$ times, we get a computational complexity of $3^{O(\Delta)} \cdot \tilde{O}(n \cdot \Delta^2)$. ◀

Indeed, we need to check all three cases. This is shown by some examples in the appendix.

5 Conclusion

We presented an algorithm which solves the feasibility 2-stage problem with one row of the form (♠) in time $O(n \cdot \Delta^2 \log(\Delta))$. Analogously to this algorithm, we presented an algorithm which solves the 2-stage problem with one row of the form (♠) optimally in time $3^{O(\Delta)} \cdot \tilde{O}(n \cdot \Delta^2)$. Furthermore, we gave an lower bound of $2^{\delta \Delta^{1/2-\varepsilon}}$ for the optimality 2-stage problem with one row of the form (♠). Both approaches use the main technique of the transformation by Frobenius [32]. If there were upper bounds, we could not use this transformation. With upper bounds, we cannot guarantee that we always get multiples of the greatest common divisor. Moreover, we would not be allowed to take the most efficient item as often as we like.

If we consider the 2-stage IP with different matrices $A_i, i \in [n]$ or with more than one row, so $A_i \in \mathbb{Z}^{r \times s}, i \in [n]$ with $r \geq 2$, then we also cannot use this transformation to get a solution. In this case one vector $x^{(0)}$ would need to solve two different equations. Given a solution x'_0 for the 2-stage IP with transformed matrix and entries $a_{11}, \ldots, a_{1s}$ and $a_{21}, \ldots, a_{2s}$, a solution x for the original 2-stage IP needs at least to satisfy $a_{11} \cdot x_0^{(0)} + \ldots + a_{1s} \cdot x_{s-1}^{(0)} = x'_0 \cdot \gcd(a_{11}, \ldots, a_{1s})$ and $a_{21} \cdot x_0^{(0)} + \ldots + a_{2s} \cdot x_{s-1}^{(0)} = x'_0 \cdot \gcd(a_{21}, \ldots, a_{2s})$. In general, this is not possible. So we cannot use our algorithms for the 2-stage IP with more than one row or with different matrices $A_i, i \in [n]$. Thus, we need another transformation. Using the Deep-in-the-Cone Lemma of Cslovjecsek, Koutecký, Lassota, Pilipczuk and Polak [11] to transform the matrices A_i and B_i for all $i \in [n]$ into their respective Hermite Normal Form is a promising approach and will be our future work.

References

1 Kyriakos Axiotis and Christos Tzamos. Capacitated dynamic programming: Faster knapsack and graph algorithms. In *ICALP*, volume 132 of *LIPIcs*, pages 19:1–19:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

2 MohammadHossein Bateni, MohammadTaghi Hajiaghayi, Saeed Seddighin, and Cliff Stein. Fast algorithms for knapsack via convolution and prediction. In *STOC*, pages 1269–1282. ACM, 2018.

3 Richard Bellman. Notes on the theory of dynamic programming iv - maximization over discrete sets. *Naval Research Logistics Quarterly*, 3(1-2):67–70, 1956. doi:10.1002/nav.3800030107.

4 Sebastian Böcker and Zsuzsanna Lipták. A fast and simple algorithm for the money changing problem. *Algorithmica*, 48(4):413–432, 2007.

5 Karl Bringmann. A near-linear pseudopolynomial time algorithm for subset sum. In *SODA*, pages 1073–1084. SIAM, 2017.

6 Karl Bringmann. Knapsack with small items in near-quadratic time. In *STOC*, pages 259–270. ACM, 2024.

7 Karl Bringmann, Anita Dürr, and Adam Polak. Even faster knapsack via rectangular monotone min-plus convolution and balancing. In *ESA*, volume 308 of *LIPIcs*, pages 33:1–33:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

8 Timothy M. Chan and Qizheng He. More on change-making and related problems. *J. Comput. Syst. Sci.*, 124:159–169, 2022.

9 Lin Chen and Guochuan Zhang. Approximation algorithms for a bi-level knapsack problem. *Theor. Comput. Sci.*, 497:1–12, 2013.

10 Jana Cslovjecsek, Friedrich Eisenbrand, Michal Pilipczuk, Moritz Venzin, and Robert Weismantel. Efficient sequential and parallel algorithms for multistage stochastic integer programming using proximity. In *ESA*, volume 204 of *LIPIcs*, pages 33:1–33:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

11 Jana Cslovjecsek, Martin Koutecký, Alexandra Lassota, Michal Pilipczuk, and Adam Polak. Parameterized algorithms for block-structured integer programs with large entries. In *SODA*, pages 740–751. SIAM, 2024.

12 Stephan Dempe and Klaus Richter. *Bilevel programming with knapsack constraints*. Citeseer, 2000.

13 Michael A. H. Dempster, Marshall L. Fisher, L. Jansen, B. J. Lageweg, Jan Karel Lenstra, and A. H. G. Rinnooy Kan. Analytical evaluation of hierarchical planning systems. *Oper. Res.*, 29(4):707–716, 1981.

14 Mingyang Deng, Xiao Mao, and Ziqian Zhong. On problems related to unbounded subsetsum: A unified combinatorial approach. In *SODA*, pages 2980–2990. SIAM, 2023.

15 Max A. Deppert and Klaus Jansen. The power of duality: Response time analysis meets integer programming. In *RTNS*, pages 37–47. ACM, 2024.

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

17 Friedrich Eisenbrand and Thomas Rothvoß. New hardness results for diophantine approximation. In *APPROX-RANDOM*, volume 5687 of *Lecture Notes in Computer Science*, pages 98–110. Springer, 2009.

18 Friedrich Eisenbrand and Thomas Rothvoss. A parameterized linear formulation of the integer hull, 2025. URL: https://arxiv.org/abs/2501.02347, arXiv:2501.02347.

19 Friedrich Eisenbrand and Robert Weismantel. Proximity results and faster algorithms for integer programming using the steinitz lemma. *ACM Trans. Algorithms*, 16(1):5:1–5:14, 2020.

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

21 Willem K. Klein Haneveld and Maarten H. van der Vlerk. *Optimizing electricity distribution using two-stage integer recourse models*, pages 137–154. Springer US, 2001.

22 Denis Hanson. On the product of the primes. *Canadian Mathematical Bulletin*, 15(1):33–37, 1972.

23 Raymond Hemmecke and Rüdiger Schultz. Decomposition of test sets in stochastic integer programming. *Math. Program.*, 94(2-3):323–341, 2003.

24 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.

25 Klaus Jansen, Kim-Manuel Klein, and Janina Reuter. Complexity bounds for block-ips, 1992. URL: `https://macau.uni-kiel.de/receive/macau_mods_00002035`.

26 Klaus Jansen, Kim-Manuel Klein, and Janina Reuter. Complexity bounds for block-ips. *Bericht des Instituts für Informatik*, 2021.

27 Klaus Jansen and Lars Rohwedder. On integer programming, discrepancy, and convolution. *Math. Oper. Res.*, 48(3):1481–1495, 2023.

28 Ce Jin. 0-1 knapsack in nearly quadratic time. In *STOC*, pages 271–282. ACM, 2024.

29 Gareth A. Jones and Josephine M. Jones. *Elementary Number Theory*. 1998.

30 Hendrik W. Lenstra Jr. Integer programming with a fixed number of variables. *Math. Oper. Res.*, 8(4):538–548, 1983.

31 Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972.

32 Kim-Manuel Klein. On the fine-grained complexity of the unbounded subsetsum and the frobenius problem. *CoRR*, abs/2108.05581, 2021.

33 Kim-Manuel Klein. About the complexity of two-stage stochastic ips. *Math. Program.*, 192(1):319–337, 2022.

34 Konstantinos Koiliaris and Chao Xu. A faster pseudopolynomial time algorithm for subset sum. In *SODA*, pages 1062–1072. SIAM, 2017.

35 Gennadi I. Malaschonok. Solution of systems of linear diophantine equations. In *CASC*, pages 401–415. Springer Berlin Heidelberg, 2001.

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

37 Yves Pochet and Laurence A Wolsey. *Production planning by mixed integer programming*, volume 149. Springer, 2006.

38 S. A. Rankin. The euclidean algorithm and the linear diophantine equation $ax + by = \gcd(a, b)$. *Am. Math. Mon.*, 120(6):562–564, 2013.

39 Victor Reis and Thomas Rothvoss. The subspace flatness conjecture and faster integer programming. In *FOCS*, pages 974–988. IEEE, 2023.

40 Alexander Schrijver. *Theory of linear and integer programming*. Wiley-Interscience series in discrete mathematics and optimization. Wiley, 1999.

41 Ernst Steinitz. Bedingt konvergente reihen und konvexe systeme. *Journal für die reine und angewandte Mathematik*, 1913.

A Diophantine Equation

In this section we prove Theorem 6 and Theorem 7. Theorem 6 describes how solutions of a linear diophantine equation look like. We use this theorem to give an algorithm which solves a system of diophantine equations in time $O(n \cdot \Delta)$. This proves Theorem 7.

Proof of Theorem 6. Let $ax + by = c$ be a diophantine equation with $a, b, c \in \mathbb{Z}, a, b \neq 0$.

$\Leftarrow$ Let c be a multiple of the greatest common divisor of a and b. By Bézouts lemma [29], there exist integral numbers $e, f \in \mathbb{Z}$ such that $ae + bf = d$, with $d := \gcd(a, b)$ holds. Since c is a multiple of the greatest common divisor of a and b, we have $c = dh$ for a $h \in \mathbb{Z}$. So, we get $a(eh) + b(fh) = aeh + bfh = (ae + bf)h = dh = c$ and with this, (eh, fh) is an integral solution for this diophantine equation.

$\Rightarrow$ Let $(x, y); x, y \in \mathbb{Z}$ be an integral solution for this diophantine equation and let $d := \gcd(a, b)$. Since d divides the value $ax + by$, the value c is a multiple of the greatest common divisor of a and b.

In total, we get: This equation has an integral solution $(x, y); x, y \in \mathbb{Z}$ if and only if c is a multiple of the greatest common divisor of a and b.

Let $(x, y); x, y \in \mathbb{Z}$ be an integral solution of this diophantine equation and let $d := \gcd(a, b)$. Then, we get $a = ud$ and $b = vd$ for $u, v \in \mathbb{Z}$. So, for $k \in \mathbb{Z}$ we get:

$$\begin{aligned} a(x + kv) + b(y - ku) &= ax + by + k(av - bu) \\ &= ax + by + k(udv - vdu) \\ &= ax + by = c \end{aligned}$$

Thus, also $(x + kv, y - ku)$ for $k \in \mathbb{Z}$ with $u = {}^{a}\!/_{d}$ and $v = {}^{b}\!/_{d}$ is a solution of this diophantine equation.

Now, let $(x_1, y_1); x_1, y_1 \in \mathbb{Z}$ and $(x_2, y_2); x_2, y_2 \in \mathbb{Z}$ be integral solutions of this diophantine equation. Let $d := \gcd(a, b)$. Then, we get $a = ud$ and $b = vd$ for $u, v \in \mathbb{Z}$. We get:

$$\begin{aligned} & ax_1 + by_1 = ax_2 + by_2 \\ \Leftrightarrow\ & a(x_1 - x_2) + b(y_1 - y_2) = 0 \\ \Leftrightarrow\ & ud(x_1 - x_2) + vd(y_1 - y_2) = 0 \\ \Leftrightarrow\ & d(u(x_1 - x_2) + v(y_1 - y_2)) = 0 \\ \Leftrightarrow\ & u(x_1 - x_2) + v(y_1 - y_2) = 0 \end{aligned}$$

Since u and v are relatively prime by definition, it holds that v divides $x_1 - x_2$ by the lemma of Euclid. So, there is a $k \in \mathbb{Z}$ such that $x_1 - x_2 = kv$. With this, we get:

$$\begin{aligned} & u(x_1 - x_2) + v(y_1 - y_2) = 0 \\ \Leftrightarrow\ & ukv + v(y_1 - y_2) = 0 \\ \Leftrightarrow\ & v(uk + (y_1 - y_2)) = 0 \\ \Leftrightarrow\ & y_1 - y_2 = -ku \end{aligned}$$

The last equivalence holds, since $b \neq 0$. So, we get $x_1 = x_2 + kv$ and $y_1 = y_2 - ku$.

In total, we get that if (x, y) is a solution of the diophantine equation, then all other solutions are of the form $(x + kv, y - ku)$ with arbitrary $k \in \mathbb{Z}$, $u := {}^{a}\!/_{\gcd(a,b)}$ and $v := {}^{b}\!/_{\gcd(a,b)}$. ◀

We can use Theorem 6 about the solutions of one diophantine equation to get solutions for a system of diophantine equations.

Let $a_i x + b_i y_i = c_i$ with $a_i, b_i, c_i \in \mathbb{N}$ be diophantine equations for all $i \in \{1, \ldots, n\}$. Algorithm 1 solves this system of diophantine equations in $O(n \cdot \Delta)$ time. It goes through all equations iteratively and searches for a solution by testing every possible value. While proceeding to the next equation, it only checks possible values that also solve all equations before the current equation.

Algorithm 1 DiophantineEquations

Require: $a_i, b_i, c_i \in \mathbb{N}, i \in [n]$
Ensure: $(x, y_1, \ldots, y_n)$ solves the given system of diophantine equations

1: $x \leftarrow 0$
2: $\texttt{stepSize} \leftarrow 1$
3: **for** $i = 1 \ldots n$ **do** ▷ Go through every equation.
4: $\quad \texttt{found} \leftarrow \texttt{False}$
5: $\quad$ **for** $k = 0 \ldots b_i - 1$ **do**
6: $\quad\quad x' \leftarrow x + k \cdot \texttt{stepSize}$
7: $\quad\quad y_i \leftarrow {}^{c_i - a_i x'}/_{b_i}$
8: $\quad\quad$ **if** $y_i \in \mathbb{Z}$ **then**
9: $\quad\quad\quad \texttt{found} \leftarrow \texttt{True}$
10: $\quad\quad\quad$ **break**
11: $\quad\quad$ **end if**
12: $\quad$ **end for**
13: $\quad$ **if** $\texttt{found}$ **then**
14: $\quad\quad x \leftarrow x'$
15: $\quad\quad \texttt{stepSize} \leftarrow \text{lcm}(\texttt{stepSize}, {}^{b_i}/_{\gcd(a_i, b_i)})$
16: $\quad$ **else**
17: $\quad\quad$ **return** There exists no solution.
18: $\quad$ **end if**
19: **end for**
20: **for** $i = 1 \ldots n$ **do** ▷ Update the old y_i values with the new x.
21: $\quad y_i \leftarrow {}^{c_i - a_i x}/_{b_i}$
22: **end for**
23: **return** $(x, y_1, \ldots, y_n)$

▶ **Lemma 10.** *Algorithm 1 returns a solution for the system of diophantine equations if and only if the system of diophantine equations has at least one solution.*

Proof. Let $a_i x + b_i y_i = c_i$ with $a_i, b_i, c_i \in \mathbb{N}$ be diophantine equations for all $i \in \{1, \ldots, n\}$. The algorithm iterates through all equations. For every equation, it checks possible b_i values in a distance of $\texttt{stepSize}$, beginning at a starting point x. Let $\texttt{stepSize}_i$ be the value of the variable $\texttt{stepSize}$ at the beginning of the i-th iteration of the first for-loop for all $i \in [n]$. Furthermore, let x_i be the solution for the equation i that the algorithm finds in this iteration. So, $x_0 = 0$ holds. In order to prove that the algorithm works correctly, we show the following three claims.

▷ Claim 11. The values x' that are at a distance other than $k \cdot \texttt{stepSize}_i$ for a $k \in \mathbb{Z}$ from the starting point x_{i-1} do not solve the system of diophantine equations.

Proof. We prove by induction on i. For $i = 1$, we have $\texttt{stepSize}_1 = 1$ and $x_0 = 0$. All values x' that are in another distance than $k \cdot \texttt{stepSize}_1 = k \cdot 1 = k$ for arbitrary $k \in \mathbb{Z}$ from the starting point $x_0 = 0$ are not integers and do not solve the system of diophantine equations. Suppose the claim holds for i. We show that it holds for $i + 1$. We have $\texttt{stepSize}_{i+1} = \text{lcm}(\texttt{stepSize}_i, {}^{b_i}/_{\gcd(a_i, b_i)})$ and $x_{i+1-1} = x_i$. So, $a_i x_i + b_i y_i = c_i$ holds for one $y_i \in \mathbb{Z}$. Note that $x_{i+1-1} = x_i$ only gets updated to x' in line 14 and the value of x' gets updated to $x_{i+1-2} + k' \cdot \texttt{stepSize}_i = x_{i-1} + k' \cdot \texttt{stepSize}_i$ for a $k' \in \mathbb{Z}$ in line 6. Therefore, the point x_i has a distance of $k' \cdot \texttt{stepSize}_i$ to the starting point x_{i-1}.

So for $k' \in \mathbb{Z}$ holds

$$\begin{aligned}
&\{x' \mid \exists k \in \mathbb{Z} : x' = x_{i-1} + k \cdot \texttt{stepSize}_i\} \\
&= \{x' \mid \exists k \in \mathbb{Z} : x' = x_{i-1} + k' \cdot \texttt{stepSize}_i + k \cdot \texttt{stepSize}_i\} \\
&= \{x' \mid \exists k \in \mathbb{Z} : x' = x_i + k \cdot \texttt{stepSize}_i\}.
\end{aligned}$$

The induction hypothesis transitively implies that all values x' that are at a distance other than $k' \cdot \texttt{stepSize}_i$ for a $k' \in \mathbb{Z}$ from the starting point x_i do not solve the system of diophantine equations. By Theorem 6 we get that only values of the form $x_i + k \cdot {}^{b_i}/_{\gcd(a_i,b_i)}$ for arbitrary $k \in \mathbb{Z}$ solve the equation i. So, all values that are at a distance other than $k \cdot {}^{b_i}/_{\gcd(a_i,b_i)}$ for arbitrary $k \in \mathbb{Z}$ from the starting point x_i do not solve the diophantine equation i and with this, they do not solve the system of diophantine equations. Together, we get that all values x' that are at a distance other than $k \cdot \operatorname{lcm}(\texttt{stepSize}_i, {}^{b_i}/_{\gcd(a_i,b_i)}) = k \cdot \texttt{stepSize}_{i+1}$ for a $k \in \mathbb{Z}$ from the starting point $x_i = x_{i+1-1}$ do not solve the system of diophantine equations.

Thus, Claim 11 holds. ◀

▷ Claim 12. If there is a solution (u_i, v_i) with $u_i = x_{i-1} + k \cdot \texttt{stepSize}_i$ for a $k \in \mathbb{Z}$ and with $u_i < x_{i-1}$ or with $u_i > x_{i-1} + (b_i - 1) \cdot \texttt{stepSize}_i$ for the diophantine equation i, then there is also a solution (x', y_i) for the diophantine equation i with $x' = x_{i-1} + k' \cdot \texttt{stepSize}_i$ for a $k' \in \mathbb{Z}$ such that $x_{i-1} \leq x' \leq x_{i-1} + (b_i - 1) \cdot \texttt{stepSize}_i$.

Proof. Let $i \in [n]$ and let (u_i, v_i) be a solution for the diophantine equation i with $u_i = x_{i-1} + k \cdot \texttt{stepSize}_i$ for a $k \in \mathbb{Z}$ and with $u_i < x_{i-1}$ or with $u_i > x_{i-1} + (b_i - 1) \cdot \texttt{stepSize}_i$. Then, we get:

$$\begin{aligned}
&a_i u_i + b_i v_i = c_i \\
\Leftrightarrow\ &a_i(x_{i-1} + k \cdot \texttt{stepSize}_i) + b_i v_i = c_i \\
\Leftrightarrow\ &a_i(x_{i-1} + (k - \left\lfloor \frac{k}{b_i} \right\rfloor \cdot b_i + \left\lfloor \frac{k}{b_i} \right\rfloor \cdot b_i) \cdot \texttt{stepSize}_i) + b_i v_i = c_i \\
\Leftrightarrow\ &a_i(x_{i-1} + (k - \left\lfloor \frac{k}{b_i} \right\rfloor \cdot b_i) \cdot \texttt{stepSize}_i + \left\lfloor \frac{k}{b_i} \right\rfloor \cdot b_i \cdot \texttt{stepSize}_i) + b_i v_i = c_i \\
\Leftrightarrow\ &a_i(x_{i-1} + (k - \left\lfloor \frac{k}{b_i} \right\rfloor \cdot b_i) \cdot \texttt{stepSize}_i) + a_i \cdot \left\lfloor \frac{k}{b_i} \right\rfloor \cdot b_i \cdot \texttt{stepSize}_i + b_i v_i = c_i \\
\Leftrightarrow\ &a_i(x_{i-1} + (k - \left\lfloor \frac{k}{b_i} \right\rfloor \cdot b_i) \cdot \texttt{stepSize}_i) + b_i(a_i \cdot \left\lfloor \frac{k}{b_i} \right\rfloor \cdot \texttt{stepSize}_i + v_i) = c_i
\end{aligned}$$

We get $x_{i-1} \leq x_{i-1} + (k - \lfloor {}^{k}/_{b_i} \rfloor \cdot b_i) \cdot \texttt{stepSize}_i \leq x_{i-1} + (b_i - 1) \cdot \texttt{stepSize}_i$, since $\texttt{stepSize}_i > 0$ holds for all $i \in [n]$ by line 15 of the algorithm. So, with $k' := k - \lfloor {}^{k}/_{b_i} \rfloor \cdot b_i \in \mathbb{Z}$, the value $(x' := x_{i-1} + k' \cdot \texttt{stepSize}_i, a_i \cdot \lfloor {}^{k}/_{b_i} \rfloor \cdot \texttt{stepSize}_i + v_i)$ is a solution for the equation i with $x_{i-1} \leq x' \leq x_{i-1} + (b_i - 1) \cdot \texttt{stepSize}_i$. Thus, Claim 12 holds. ◀

▷ Claim 13. The solution x_i solves also all other equations j for $j \in [i-1]$.

Proof. We prove by induction on i. For $i = 1$, the set $\{1, \ldots, i-1\}$ is empty and the claim

is true. Suppose the claim holds for i. We show that it holds for $i+1$. We get

$$\begin{aligned}\texttt{stepSize}_{i+1} &= \operatorname{lcm}\left(\texttt{stepSize}_i, \frac{b_i}{\gcd(a_i,b_i)}\right)\\ &= \operatorname{lcm}\left(1, \frac{b_1}{\gcd(a_1,b_1)}, \ldots, \frac{b_i}{\gcd(a_i,b_i)}\right)\\ &= \operatorname{lcm}\left(\frac{b_1}{\gcd(a_1,b_1)}, \ldots, \frac{b_i}{\gcd(a_i,b_i)}\right).\end{aligned}$$

Since x_{i+1} gets updated to x' in line 14 and x' itself gets updated to $x_i + k \cdot \texttt{stepSize}_{i+1} = x_i + k \cdot \operatorname{lcm}({}^{b_1}/_{\gcd(a_1,b_1)}, \ldots, {}^{b_i}/_{\gcd(a_i,b_i)})$ for a $k \in \mathbb{Z}$ in line 6, the point x_{i+1} has a distance of $k \cdot \operatorname{lcm}({}^{b_1}/_{\gcd(a_1,b_1)}, \ldots, {}^{b_i}/_{\gcd(a_i,b_i)})$ for a $k \in \mathbb{Z}$ to the starting point x_i. By the induction hypothesis, the value x_i solves all equations j for $j \in [i-1]$ and by definition, it solves equation i. By Theorem 6, every $x'_j := x_i + k_j \cdot {}^{b_j}/_{\gcd(a_j,b_j)}$ for arbitrary $k_j \in \mathbb{Z}$ solves equation j for all $j \in [i]$. Because $x_i + k \cdot \operatorname{lcm}({}^{b_1}/_{\gcd(a_1,b_1)}, \ldots, {}^{b_i}/_{\gcd(a_i,b_i)}) = x_i + k_j \cdot {}^{b_j}/_{\gcd(a_j,b_j)}$ for a $k_j \in \mathbb{Z}$ holds for every $j \in [i]$, the value x_{i+1} solves all equations j for $j \in [i]$.

So, Claim 13 holds. ◀

The algorithm checks for every equation $i \in [n]$ the values $x_{i-1} + k \cdot \texttt{stepSize}_i$ for all $k \in [b_i - 1]_0$. If it finds a solution, it saves this solution as x_i. If it does not find a solution, it returns that no solution exists. Otherwise, it returns at the end the last solution with adjusted values y_i for all $i \in [n]$. By Claim 11, it is correct that for every $i \in [n]$ our algorithm does not check the values x' that are in another distance than $k \cdot \texttt{stepSize}_i$ for a $k \in \mathbb{Z}$ from the starting point x_{i-1} since these values do not solve the system of diophantine equations. In combination with Claim 12, it is correct that for every $i \in [n]$ our algorithm only checks the values x' with $x' = x_{i-1} + k' \cdot \texttt{stepSize}_i$ for all $k' \in [b_i - 1]_0$. Since the algorithm checks for all $i \in [n]$ by Claim 11 all potential solutions x' in the interval $[x_{i-1}, x_{i-1} + (b_i - 1) \cdot \texttt{stepSize}_i]$ and by Claim 12, in combination with Claim 11, a solution in this interval exists, if a solution exists, our algorithm finds a solution if and only if the system of diophantine equations has a solution. By Claim 13, the value x_n, if x_n exists, solves all equations j for $j \in [n-1]$ and by definition it solves equation n. Because for all $i \in [n]$ the values y_i are correctly adjusted to the value x_n, our algorithm returns a correct solution if it returns a solution. So, Lemma 10 holds. ◀

▶ **Lemma 14.** *Algorithm 1 has computational complexity of $O(n \cdot \Delta)$.*

Proof. The lines $1-2$, 4, $6-14$, $16-19$ and $21-23$ take $O(1)$ time. Since $\operatorname{lcm}(a,b) = \gcd(a,b) \cdot {}^{a}/_{\gcd(a,b)} \cdot {}^{b}/_{\gcd(a,b)}$ holds for all $a, b \in \mathbb{N}$, the running time of line 15 is determined by the computation of the greatest common divisor of two natural numbers. The greatest common divisor of two natural numbers $a > b$ can be computed in time $O(\log(a))$ by the euclidean algorithm. Line 15 takes $O(\log(\Delta))$ time. The first for loop runs n times, so the lines $4-18$ get executed n times. The second for loop runs b_i times, so the lines $6-11$ get executed at most $n \cdot \Delta$ times. The third for loop runs n times. So, line 21 gets executed n times. In total, we get a computational complexity of $O(n \cdot \Delta)$ for Algorithm 1. ◀

All together, we get the following theorem.

▶ **Theorem 7 (✂).** *A system of linear diophantine equations can be solved in time $O(n \cdot \Delta)$.*

Proof. We get this right away by Lemma 10 and Lemma 14. ◀

B All-Target Subset Sum

In this section, we provide an auxiliary algorithm required for the solution of the 2-stage formulation. In particular, we require an algorithm that returns all reachable target values for an unbounded Subset Sum Problem. We note that there are several approaches that solve the classical subset sum problem in near-linear time, see [5, 34].

▶ **Lemma 15** ([5]). *Let $b \in \mathbb{N}$ with $b \geq \Delta^2$. It takes $O(\Delta^2 \log(\Delta))$ time to compute for every target value $t \in [b - \Delta^2, b]$ if there exists a solution for unbounded Subset Sum Problem with at most Δ values.*

Proof. Let S be a unbounded Subset Sum Problem with at most Δ values. For all target values $t \in [\Delta^2]_0$ we compute if there is a solution for S with target value t in $O(\Delta^2 \log(\Delta))$ time by Bringmann [5]. Since S is a unbounded Equivalence Knapsack Problem with cost vector $c = \mathbf{0}$, we can use Theorem 5. Thus, there is a solution for S with target value $t > \Delta^2$ if and only if there is a solution for S with target value $k := \max\{j \mid \exists n \in \mathbb{N}_0 : \ t = j + n \cdot w_{i_e}, j \in [\Delta^2]_0\}$. The item w_{i_e} is an arbitrary item of S since all profits are 0. W.l.o.g. let w_{i_e} be the biggest item of S. In order to check if there exists a solution for S with target value $t > \Delta^2$, we compute $n := \left\lceil {}^{t-\Delta^2}/_{w_{i_e}} \right\rceil$ and $k := t - n \cdot w_{i_e}$ and then check if there is a solution for S with target value k. Since $k \leq \Delta^2$, we can look this up. Together, it takes $O(\Delta^2 \log(\Delta))$ time to compute for every target value $t \in [b - \Delta^2, b]$ if there exists a solution for S. ◀

C All-Capacities Unbounded Knapsack

Here, we describe an algorithm to solve the uEKP. The algorithm computes an optimal solution for a value $k \leq w_{\max}^2$ and adds the most efficient item i_e, i.e. the item with the best profit to weight ratio, as many times until the capacity is reached. The following theorem shows that this procedure gives us an optimal solution. Let P be an uEKP with items I, weights $w_i, i \in I$, profits $p_i, i \in I$ and capacity $C \in \mathbb{N}$.

▶ **Theorem 5** (✂). *Let $k := \max\{j \mid \exists n \in \mathbb{N}_0 : \ C = j + n \cdot w_{i_e}, j \in [w_{\max}^2]_0\}$, let $n := {}^{C-k}/_{w_{i_e}}$ and let x^* be an optimal solution for the unbounded Equivalence Knapsack Problem with items I and capacity k. Let $x^T = (x_1, \ldots, x_{|I|})$ with $x_i := \begin{cases} x_i^* + n & i = i_e \\ x_i^* & \text{otherwise} \end{cases}, i \in I$. If $C > w_{\max}^2$, then x is an optimal solution for P.*

We prove this theorem using the following lemma. It claims that in every solution with capacity greater than $w_{\max}^2$ the most efficient item can appear. Thus, for every solution with capacity greater than $w_{\max}^2$ exists an optimal solution such that the most efficient item is part of it.

▶ **Lemma 16.** *Let $\Delta \in \mathbb{N}$. For a non-empty subset $M \subseteq [\Delta]$ with $|M| = n$ and multiplicities x_i for all $k_i \in M$ with $\sum_{i \in M} x_i k_i = K > \Delta^2$ we have for all $k_i \in M$ a non-empty subtotal $t := \sum_{i \in M} x_{i'} k_i \leq K$ with $k_i | t$.*

Proof. Select an arbitrary $k \in M$. As the sum of all values and their multiplicities is larger than Δ^2 and all values are at most Δ, the sum of multiplicities $\sum_{i \in M} x_i \geq \Delta \geq k$ must hold. Write the sum by removing multiplicities and adding x_i copies of k_i, i.e. $k_1 + k_1 + \ldots + k_1 + k_2 + \ldots + k_n$. Take the all partial sums of this sum by stopping summation after the j-th summand for all $j \in [n]$ and denote their value by K_j. There are at least

$k \leq j$ of these sums. Either, for one sum up to some j it holds $K_j \mod k = 0$, or there are two partial sums K_j, $K_{j'}$ with $K_j \mod k \equiv K_{j'} \mod k$ by pigeonhole. Then, $K_j - K_{j'} \mod k = 0$ must hold and thus the claim holds. ◀

A consequence of this for unbounded Equivalence Knapsack Problem, we know that every solution with capacity greater than $w_{\max}^2$ contains a block of items which can be replaced by copies of the most efficient item. Since this replacement never reduces the total profit, there exists an optimal solution with capacity greater than $w_{\max}^2$ with the most efficient item. Repeating this, we can prove Theorem 5:

Proof. Let $C > w_{\max}^2$. Let $k := \max\{j \mid \exists n \in \mathbb{N}_0 : \ C = j + n \cdot w_{i_e}, j \in [w_{\max}^2]_0\}$, let $n := {}^{C-k}/_{w_{i_e}}$ and let x^* be an optimal solution for the unbounded Equivalence Knapsack Problem with items I and capacity k. Let $x^T = (x_1, \ldots, x_{|I|})$ with $x_i := \begin{cases} x_i^* + n & i = i_e \\ x_i^* & \text{otherwise} \end{cases}, i \in I$.

Suppose there is an optimal solution x' with $\sum_{i \in I} w_i x_i' = C$ and $\sum_{i \in I} p_i x_i' > \sum_{i \in I} p_i x_i$.

Since $\sum_{i \in I} w_i x_i' = C > w_{\max}^2$, we get that there exists a non-empty subtotal $t := \sum_{i \in T} w_i x_i'', T \subseteq I, x_i'' \leq x_i'$ with $w_{i_e} | t$ by Lemma 16. Consider the optimal solution x' without the items of this subtotal: Let $\hat{x}^T = (\hat{x}_1, \ldots, \hat{x}_{|I|})$ with $\hat{x}_i := x_i' - x_i''$. If we get for the total weight of these items $k' := \sum_{i \in I} w_i \hat{x}_i > w_{\max}^2$, then we can apply Lemma 16 and repeat this procedure. Otherwise, we get $k' \leq w_{\max}^2$. So, in the end we remain $k' \leq w_{\max}^2$ and get the following three cases:

Case 1: $k' > k$: Since $k := \max\{j \mid \exists n \in \mathbb{N}_0 : \ C = j + n \cdot w_{i_e}, j \in [w_{\max}^2]_0\}$ holds, $k' > k$ is a contradiction.

Case 2: $k' = k$: There exists a subtotal $t' := \sum_{i \in T'} w_i x_i'', T' \subseteq I, x_i'' \leq x_i'$ with $t' = k$. Now, we can divide the optimal solution x' in two parts. One part $\bar{x}'$ is defined by the subtotal t', so $\bar{x}' := \sum_{i \in T'} x_i''$, and the other part $\hat{x}'$ is the remaining part, so $\hat{x}_i' := x_i' - x_i''$. Analogously, we can divide the solution x in two parts $\bar{x}$ and $\hat{x}$. Since $\sum_{i \in I} p_i x_i' > \sum_{i \in I} p_i x_i$ holds, we get that $\sum_{i \in I} p_i \bar{x}_i' > \sum_{i \in I} p_i \bar{x}_i$ or $\sum_{i \in I} p_i \hat{x}_i' > \sum_{i \in I} p_i \hat{x}_i$ hold. By definition of x, $\bar{x}$ is the optimal solution of the unbounded Equivalence Knapsack Problem with items I and capacity k. So, $\sum_{i \in I} p_i \bar{x}_i' > \sum_{i \in I} p_i \bar{x}_i$ doesn't hold. Thus, $\sum_{i \in I} p_i \hat{x}_i' > \sum_{i \in I} p_i \hat{x}_i$ holds. Because $\hat{x} := \begin{cases} n & i = i_e \\ 0 & \text{otherwise} \end{cases}, i \in I$ holds, this is a contradiction of the definition of the most efficient item.

Case 3: $k' < k$: There exists a subtotal $t' := \sum_{i \in T'} w_i x_i'', T' \subseteq I, x_i'' \leq x_i'$ with $t' = C - k' = m \cdot w_{i_e}$ for one $m \in \mathbb{N}_0$. Since $k' < k$ holds, we get $n < m$. We can replace t' by m copies of the most efficient item i_e and the profit remains the same. Otherwise, x' would not be an optimal solution by the definition of the most efficient item. After this replacing, there exists a subtotal $t_1' := \sum_{i \in T_1'} w_i x_i''', T_1' \subseteq I$ with $t_1' = C - k = n \cdot w_{i_e}$. Thus, we can get to a contradiction analogous to case 2. ◀

With Theorem 5 we can describe an algorithm which solves the unbounded Equivalence Knapsack Problem.

▶ **Lemma 17** ([8]). *Let $b \in \mathbb{N}$ with $b \geq \Delta^2$ and let P be a unbounded Equivalence Knapsack Problem with highest weight $\Delta \in \mathbb{N}$. It takes $O(\Delta^2 \log(\Delta))$ time to compute for every capacity $C \in [b - \Delta^2, b]$ the maximum profit of an optimal solution for P if there exists a solution.*

Proof. Let P be a unbounded Equivalence Knapsack Problem with highest weight $\Delta \in \mathbb{N}$. Chan and He [8] provide an algorithm for All-Capacities Unbounded Knapsack. This problem is equivalent to the problem of finding the maximum total profit for all capacities up to a given value C for a unbounded Equivalence Knapsack Problem with the only difference that

the capacity does not have to be reached exactly. They use a dynamic program to compute entries

$D[j] :=$ maximum total profit that can be reached only using items i with weight $w_i \leq j$

If we slightly modify their algorithm by setting $D[0] = 0$ and $D[j] = \max_{1 \leq i \leq \lceil 3\Delta^2/j \rceil : w_i \leq j}(D[j - w_i] + p_i)$, we get an algorithm which solves the problem of finding the maximum total profit for all capacities up to a given value C for a unbounded Equivalence Knapsack Problem. This is analog to their all-targets change-making problem [8]. The correctness and the running time follow analogously to these both algorithms of Chan and He [8]. Thus, we get for every capacity $C \in [\Delta^2]_0$ the maximum profit of an optimal solution for P if there exists a solution in time $O(\Delta^2 \log(\Delta))$ [8]. For every capacity $C > \Delta^2$, let $k := \max\{j \mid \exists n \in \mathbb{N}_0 : C = j + n \cdot w_{i_e}, j \in [\Delta^2]_0\}$, let $n := {}^{C-k}/_{w_{i_e}}$ and let x^* be an optimal solution for the unbounded Equivalence Knapsack Problem with items I and capacity k. Let $x^T = (x_1, \ldots, x_{|I|})$ with $x_i := \begin{cases} x_i^* + n & i = i_e \\ x_i^* & \text{otherwise} \end{cases}, i \in I$. By Theorem 5, we get that x is an optimal solution for P. Since we know the maximum profit of the solution x^* by looking it up ($k \leq \Delta^2$) and we know p_{i_e}, we can compute the maximum profit of x in time $O(1)$.

Together, it takes $O(\Delta^2 \log(\Delta))$ time to compute for every capacity $C \in [b - \Delta^2, b]$ the maximum total profit of an optimal solution for P if there exists a solution. ◀

D Reducing the search space of optimal solutions

In this section, we show an alternative result to limit the size of an optimal solution to problems of the form (♠). We show that, when $\mathcal{A}x$ is large enough, there is a number K that is a multiple of the most efficient column of all blocks in $\mathcal{A}$. Then, we can replace this solution with one taking exclusively the more efficient columns of A or all B_i respectively. This yields that the right-hand side b can be limited by $\|b\|_\infty \leq \max\{s, t\}3^\Delta$.

▶ **Lemma 18.** *Given multisets $T_1, \ldots, T_n \subset \mathbb{Z}_{\geq 0}$, where all elements $t \in T_i$ have bounded size $\|t\|_\infty \leq \Delta$ and each multiset contains at most k distinct values. Assuming that the total sum of all elements in each set is equal, i.e.*

$$\sum_{t \in T_1} t = \ldots = \sum_{t \in T_n} t$$

then there exist nonempty submultisets $S_1 \subseteq T_1, \ldots, S_n \subseteq T_n$ of bounded size $|S_i| \leq lcm\{T\} \cdot k \leq lcm\{1, \ldots, \Delta\} \cdot k \leq 3^\Delta \cdot k$, where $T = \bigcup T_i$, i.e. all numbers of the sets T_i, such that

$$\sum_{s \in S_1} s = \ldots = \sum_{s \in S_n} s.$$

Proof. As a warm-up, consider the case where each T_i consists only of a single natural number $x_i \in \{1, \ldots, \Delta\}$. Then, it is easy to find common multiples, as we have $\frac{\text{lcm}\{T\}}{1} \cdot 1 = \frac{\text{lcm}\{T\}}{2} \cdot 2 = \ldots = \frac{\text{lcm}\{T\}}{t_{\max}} \cdot t_{\max}$, where $t_{\max}$ is the largest number in T. Then, for each x_i we can choose the subset consisting of $\frac{\text{lcm}\{T\}}{x_i} \cdot x_i$. The sum of all these subsets is then equal to $\text{lcm}\{T\}$ and we are done. Now, we consider the case of arbitrary T_i. Let the number of distinct numbers in each subset T_i be at most k. If, for any $i \in [n]$, $|T_i| \leq k \cdot \text{lcm}\{T\}$, the sum of this T_i is bounded by $b = \sum_{t \in T_i} t \leq \Delta k \text{lcm}\{T\}$. As we have $\sum_{t \in T_1} t = \cdots = \sum_{t \in T_i} t = \cdots = \sum_{t \in T_n} t$, all other T_j must have at most $|T_j| \leq k \cdot \text{lcm}\{T\}$ elements, as all elements are larger than zero. If, on the other hand, we have $|T_i| > k \cdot \text{lcm}\{T\}$ for all $i \in [n]$, then, by pigeonhole, at least one number $x_i \in \{1, \ldots, \Delta\}$ must occur at least $\text{lcm}\{T\}$ many times. Then, from our warm-up, we know that we need at most $\text{lcm}\{T\}$ many copies of a number $x_i \in \{1, \ldots, \Delta\}$ to sum

up to exactly $\operatorname{lcm}\{T\}$. Repeating this for each multiset yields a bound of $|S_i| \leq k \cdot \operatorname{lcm}\{T\}$. Plugging in the bound of $\operatorname{lcm}\{T\} \leq \operatorname{lcm}\{1, \ldots, \Delta\} \leq 3^{\Delta}$ by Hanson [22] completes the proof. ◀

Next, we adapt Klein's bound on the Graver basis to two-stage ILPs of the for (♠). In this proof, we utilize the Steinitz Lemma, originally given by Steinitz [41] and later improved in the bound by Grinberg and Sevast'yanov [20]. It shows that, for a set of vectors $v_i \in \mathbb{R}^m$ that sum up to 0 and are small with respect to some norm, we can find an ordering such that any partial sum $v_0 + \cdots + v_j$ is not far from 0 with respect to the same norm. This Lemma has found several applications in the context of integer programming, see e.g. Eisenbrand's and Weismantel's result [19] for general ILPs for details. The following theorem is a direct application of [33, Theorem 2] to our context.

▶ **Theorem 19** ([33]). *Let y be a Graver element of the constraint matrix $\mathcal{A}$ of (♠) with $s = 1$. Then, $\|y\|_\infty$ is bounded by $(2\Delta+1) \cdot 3^{2\Delta+1}$. More precisely, $\|y^{(i)}\|_1 \leq (2\Delta+1) \cdot 3^{2\Delta+1}$ for every $0 \leq i \leq n$.*

Proof. Let $y \in \mathbb{Z}^{1+nt}_{\geq 0}$ be a cycle of ILP (♠), i.e. $\mathcal{A}y = 0$. Consider a submatrix of $\mathcal{A}$ denoted by $(AB^{(i)}) \in \mathbb{Z}^{1 \times (1+t)}$ consisting of the block matrix A of the first stage and the i-th block $B^{(i)}$ of the second stage. Further consider the corresponding variables $v^{(i)} = \begin{pmatrix} y^{(0)} \\ y^{(i)} \end{pmatrix} \in \mathbb{Z}^{1+t}$ of the respective matrix A and $B^{(i)}$. As we have $\mathcal{A}y = 0$, we must also have $(AB^{(i)})v^{(i)} = 0$. We can decompose $v^{(i)}$ into a multiset C_i of indecomposable elements, i.e. $v^{(i)} = \sum_{c \in C_i} c$. Thus, we can use the Steinitz Lemma to bound $\|c\|_1 \leq (2\Delta + 1)$ for each $c \in C_i$.

Since all matrices $(AB^{(i)})$ share the same set of variables in the overlapping block matrices A, we cannot choose indecomposable elements independently in each block to obtain a cycle of smaller size for the entire matrix $\mathcal{A}$. Let $p : \mathbb{Z}^{1+t} \to \mathbb{Z}$ be the projection that maps a cycle c of a block matrix $(AB^{(i)})$ to the variables in the overlapping part, i.e. $p(c) = p(\begin{pmatrix} c^{(0)} \\ c^{(i)} \end{pmatrix} = c^{(0)}$. In the case that $\|y\|_\infty$ is large we show that we can find a cycle $\bar{y}$ of smaller length and $\bar{y} \leq y$. In order to obtain this cycle $\bar{y}$ for the entire matrix $\mathcal{A}$, we have to find a multiset of cycles $\bar{C}_i \subset C_i$ in each block matrix $(AB^{(i)})$ such that the sum of projected parts is identical, i.e. $\sum_{c \in \bar{C}_1} p(c) = \cdots = \sum_{c \in \bar{C}_n} p(c)$. We apply Lemma 18 to the multiset $p(C_1), \ldots, p(C_n)$, where $p(C_i) = \{p(c) | c \in C_i\}$ is the multiset of projected elements in C_i with $\|p(c)\|_1 \leq (2\Delta + 1)$. Note that $\sum_{x \in p(C_1)} x = \cdots = \sum_{x \in p(C_n)} x = y^{(0)}$ and hence we can apply Lemma 18. Note that $y^{(0)} \in \mathbb{Z}_{\geq 0}$. Since every $v^{(i)}$ is decomposed in a sign compatible way, every entry of the vector in $p(C_i)$ has the same sign. Thus, we can flip the negative signs in order to apply Lemma 18.

By the Lemma, there exist submultisets $S_1 \subseteq p(C_1), \ldots, S_n \subseteq p(C_n)$ such that $\sum_{x \in S_1} x = \cdots = \sum_{x \in S_n} x$ and $|S_i| \leq s3^{\|c\|_1} = 3^{2\Delta+1}$. As there exist submultisets $\bar{C}_1 \subseteq C_1, \ldots, \bar{C}_n \subseteq C_n$ with $p(\bar{C}_1) = S_1, \ldots, p(\bar{C}_n) = S_n$, we can use those submultisets $\bar{C}_i$ to define a solution $\bar{y} \leq y$. For $i > 0$ let $\bar{y} = \sum_{c \in \bar{C}_i} \bar{p}(c)$, where $\bar{p}(c)$ is the projection that maps a cycle $c \in \bar{C}_i$ to the part that belongs to matrix $B^{(i)}$, i.e. $\bar{p}(\begin{pmatrix} c^{(0)} \\ c^{(i)} \end{pmatrix} = c^{(i)}$. Let $\bar{y}^{(0)} = \sum_{c \in \bar{C}_i} p(c)$ for an arbitrary $i > 0$, which is well defined as the sum is identical for all multisets $\bar{C}_i$. As the cardinality of multisets $\bar{C}_i$ is bounded, we know by construction of $\bar{y}$ that the one norm of every $y^{(i)}$ is bounded by

$$\|y^{(i)}\|_1 \leq (2\Delta + 1) \cdot 3^{2\Delta+1}.$$

This directly implies the infinity norm bound as well.

◀

▶ **Theorem 3** (✂). *For problems of the form* (♠) *with* $s = 1$*, we have* $\|b\|_\infty \leq lcm\{a, B_e^{(1)}, \ldots, B_e^{(n)}\} + 2\Delta^2 \leq 3^\Delta + 2\Delta^2$*, where* $B_e^{(i)} := B_j^{(i)} : \max\{c_j^{(i)}/B_j^{(i)} | j \in [t]\}$ *for all* $i \in [n]$.

Proof. We prove this claim by arguing over the structure of optimal solutions. When $s = 1$, either by reduction or given, let $K := k \cdot A$, where $k \in \mathbb{Z}_{\geq 0}$ is a factor, be a multiple of the column $a = A$. The only change in a feasible solution can be found by adding (or subtracting) K to a and subtracting (or adding) K in each of the blocks. We know, by Theorem 5, that the optimal solution of (♠) with $s = 1$ contains, for every block $B^{(i)}$, many copies of the most efficient column and at most Δ copies of each other column, i.e. $x_j^{(i)} \leq \begin{cases} b_i/B_j^{(i)} & \text{if } j \text{ is the most efficient column} \\ \Delta & \text{otherwise.} \end{cases}$

As a reminder, the most efficient column for each block i is the column j which maximizes $c_j^{(i)}/B_j^{(i)}$. Denote the most efficient column of block i as $B_e^{(i)}$ for all $i \in [n]$. Following this observation, we can now argue the structure of optimal solutions. For every K which is a multiple of $\text{lcm}\{a, B_e^{(1)}, \ldots, B_e^{(n)}\}$, there is a simple exchange possible. Either we have $K/a \cdot c_0 \geq \sum_{i=1}^n K/B_e^{(i)} \cdot c_e^{(i)}$, i.e. the column a adds more profit than removing all other efficient columns for each block i, or $K/a \cdot c_0 < \sum_{i=1}^n K/B_e^{(i)} \cdot c_e^{(i)}$. In the first case, we can add K/a many copies of a to the solution and know that there is no more profitable way to fill this size. In the second case, we add $K/B_e^{(i)}$ copies of the other items and again know that this is the most profitable way to fill this size.

This observation, in turn, allows us to limit the right-hand side b. Assume that $b_i > \text{lcm}\{a, B_e^{(1)}, \ldots, B_e^{(n)}\} + \Delta^2$ for all $i \in [n]$. Then, we can find a size K which is a multiple of $\text{lcm}\{a, B_e^{(1)}, \ldots, B_e^{(n)}\}$ and fill this space like above. This is feasible until we have at least one $b_i < \text{lcm}\{a, B_e^{(1)}, \ldots, B_e^{(n)}\} + \Delta^2$. However, once we have one $b_i < \text{lcm}\{a, B_e^{(1)}, \ldots, B_e^{(n)}\} + \Delta^2$, we know that we can also reduce the remaining $b_{i'}$ to $\text{lcm}\{a, B_e^{(1)}, \ldots, B_e^{(n)}\} + 2\Delta^2$. This is because we have $x_0 a \leq \text{lcm}\{a, B_e^{(1)}, \ldots, B_e^{(n)}\} + \Delta^2$ due to b_i. Thus, the remaining $b_{i'}$ can be reduced to a value at most Δ^2 larger by filling the remaining $b_{i'} - (b_i + \Delta^2)$ with copies of the most efficient column for all $b_{i'}$. As a result, we have $\|b\|_\infty \leq \text{lcm}\{a, B_e^{(1)}, \ldots, B_e^{(n)}\} + 2\Delta^2$. ◀

As a result of this, our algorithm only needs to compute solutions with a right-hand side at most $\text{lcm}\{a, B_e^{(1)}, \ldots, B_e^{(n)}\} + 2\Delta^2$. This limits the number of values for y^0 we have to check, speeding up the algorithm significantly.

E 2-stage IP with one row

E.1 Proofs

Case 3: (missing parts, feasibility) We have

$$\begin{aligned}
z_{\ell_0} &= x_0 - k \cdot \operatorname{lcm}(v_1, \dots, v_n) \\
&= x_0 - \left(\left\lceil \frac{x_0 - \frac{\Delta^2}{\gcd(A)}}{\operatorname{lcm}(v_1, \dots, v_n)} \right\rceil - 1 \right) \cdot \operatorname{lcm}(v_1, \dots, v_n) \\
&> x_0 - \left(\frac{x_0 - \frac{\Delta^2}{\gcd(A)}}{\operatorname{lcm}(v_1, \dots, v_n)} + 1 - 1 \right) \cdot \operatorname{lcm}(v_1, \dots, v_n) \\
&= x_0 - \frac{x_0 - \frac{\Delta^2}{\gcd(A)}}{\operatorname{lcm}(v_1, \dots, v_n)} \cdot \operatorname{lcm}(v_1, \dots, v_n) \\
&= x_0 - x_0 + \frac{\Delta^2}{\gcd(A)} \\
&= \frac{\Delta^2}{\gcd(A)},
\end{aligned}$$

since $\lceil a \rceil < a + 1$ holds for all $a \in \mathbb{Q}_0$. If $z_{\ell_i} > {\Delta^2}/{\gcd(B_i)}$ for all $i \in [n]$, then z_ℓ is a feasible solution. Otherwise, there exists no feasible solution for problem (3), since z_{ℓ_0} cannot be reduced further and with this, no z_{ℓ_i} for $i \in [n]$ can be increased. We get this, because

$$\begin{aligned}
&x_0 - \left(\left\lceil \frac{x_0 - \frac{\Delta^2}{\gcd(A)}}{\operatorname{lcm}(v_1, \dots, v_n)} \right\rceil - 1 + 1 \right) \cdot \operatorname{lcm}(v_1, \dots, v_n) \\
&= x_0 - \left\lceil \frac{x_0 - \frac{\Delta^2}{\gcd(A)}}{\operatorname{lcm}(v_1, \dots, v_n)} \right\rceil \cdot \operatorname{lcm}(v_1, \dots, v_n) \\
&\leq x_0 - \frac{x_0 - \frac{\Delta^2}{\gcd(A)}}{\operatorname{lcm}(v_1, \dots, v_n)} \cdot \operatorname{lcm}(v_1, \dots, v_n) \\
&= x_0 - x_0 + \frac{\Delta^2}{\gcd(A)} \\
&= \frac{\Delta^2}{\gcd(A)}
\end{aligned}$$

Since $z_0 = x_0 + h \cdot \operatorname{lcm}(v_1, \dots, v_n)$ with $h \in \mathbb{Z}$, the value $k' := k + 1$ is the next integer which would reduce z_{ℓ_0} further. But with this inequalities we get $x_0 - k' \cdot \operatorname{lcm}(v_1, \dots, v_n) \leq {\Delta^2}/{\gcd(A)}$ and thus, z_{ℓ_0} cannot be reduced further. Otherwise, we would get a contradiction to the assumption of case 3.

Case 1: $Ax^{(0)} \leq \Delta^2$: Again, the contribution of the first block is small. Thus, we can, similar to the feasibility case, generate all optimal solutions for these values and use these to check all optimal solutions for the blocks B_i. We compute, for all capacities $C \in [\Delta^2]_0$, of the unbounded Equivalence Knapsack Problem $Ax^{(0)} = C$ the maximum total profit and save them in a list ℓ_A. For all $j \in [\Delta^2]_0$, we have at position ℓ_{A_j} the maximum total profit for the unbounded Equivalence Knapsack Problem $Ax^{(0)} = j$ if there exists a solution and -1 otherwise. Analogously, for all $i \in [n]$, we compute for all capacities in $[b_i - \Delta^2, b_i]$ for the unbounded Equivalence Knapsack Problem with weights B_i and profits $c^{(i)}$ the

maximum total profits and save them in lists ℓ_{B_i}. For all $i \in [n]$ and $j \in [\Delta^2]$, we have at position $\ell_{B_{i_j}}$ the maximum total profit for the unbounded Equivalence Knapsack Problem $B_i x^{(i)} = b_i - j$ if there exists a solution and -1 otherwise. We can compute all these lists in time $O(n \cdot \Delta^2 \log(\Delta))$ by Lemma 17.

Since $Ax^{(0)} \leq \Delta^2$, the profit of the optimal solution is the highest sum $\max\{\ell_{A_j} + \sum_{i\in[n]} \ell_{B_{i_j}} \mid \ell_{A_j} \neq -1, \ell_{B_{i_j}} \neq -1 \text{ for all } i \in [n], j \in [\Delta^2]_0\}$. Thus, for every $j \in [\Delta^2]_0$, we check all the entries ℓ_{A_j} and $\ell_{B_{i_j}}$. If all these entries are not equal to -1, we sum up the values and update the current maximum profit value. In the end, we get a maximum profit and a capacity j for which this profit is reached, or the result that there exists no solution. If there exists a solution, we solve the unbounded Equivalence Knapsack Problem $Ax^{(0)} = j$ with profits $c^{(0)}$ and the unbounded Equivalence Knapsack Problems $B_i x^{(i)} = b_i - j$ with profits $c^{(i)}$ for all $i \in [n]$. It takes $O(\Delta^2 \cdot n)$ time to determine the maximum profit and the corresponding j value. Solving all unbounded Equivalence Knapsack Problems takes $O(n \cdot \Delta^2)$ time by Jansen and Rohwedder [27].

All together, we can solve problem (♠) in time $O(n \cdot \Delta^2 \log(\Delta))$.

Case 2: $B_i x^{(i)} \leq \Delta^2$ holds for at least one $i \in [n]$: We have a small contribution of the second stage blocks B_i. We once again use a similar observation as in the feasibility problem to limit the search space for optimal solutions and can check all of these quickly. For $b_m := \min_{i\in[n]} b_i$, we get $B_m x^{(m)} \leq \Delta^2$. Otherwise, we would have $b_i \geq b_m \Leftrightarrow Ax^{(0)} + B_i x^{(i)} \geq Ax^{(0)} + B_m x^{(m)} \Leftrightarrow B_i x^{(i)} \geq B_m x^{(m)}$ and with this, $B_i x^{(i)} \geq B_m x^{(m)} > \Delta^2$ for all $i \in [n]$, a contradiction. With $Ax^{(0)} + B_m x^{(m)} = b_m \Leftrightarrow Ax^{(0)} = b_m - B_m x^{(m)} \geq b_m - \Delta^2$, we get $Ax^{(0)} \in [b_m - \Delta^2, b_m]$. So, for all capacities $C \in [b_m - \Delta^2, b_m]$ of the unbounded Equivalence Knapsack Problem $Ax^{(0)} = C$ we compute the maximum total profit and save them in a list ℓ_A. For all $j \in [\Delta^2]_0$, we have at position ℓ_{A_j} the maximum total profit for the unbounded Equivalence Knapsack Problem $Ax^{(0)} = b_m - j$ if there exists a solution and -1 otherwise. Analogously, for all $i \in [n]$, we compute for all capacities in $[b_i - b_m, b_i - b_m + \Delta^2]$ for the unbounded Equivalence Knapsack Problem with weights B_i and profits $c^{(i)}$ the maximum total profits and save them in lists ℓ_{B_i}. For all $i \in [n]$ and $j \in [\Delta^2]$, we have at position $\ell_{B_{i_j}}$ the maximum total profit for the unbounded Equivalence Knapsack Problem $B_i x^{(i)} = b_i - b_m + j$ if there exists a solution and -1 otherwise. We can compute all these lists in time $O(n \cdot \Delta^2 \log(\Delta))$ by Lemma 17.

Since $B_m x^{(m)} \leq \Delta^2$, the profit of the optimal solution is the highest sum $\max\{\ell_{A_j} + \sum_{i\in[n]} \ell_{B_{i_j}} \mid \ell_{A_j} \neq -1, \ell_{B_{i_j}} \neq -1 \text{ for all } i \in [n], j \in [\Delta^2]_0\}$. Thus, for every $j \in [\Delta^2]_0$ we check all the entries ℓ_{A_j} and $\ell_{B_{i_j}}$. If all these entries are not equal to -1, we sum up the values and update the current maximum profit value. In the end, we get a maximum profit and a capacity j for which this profit is reached or the result that there exists no solution. If there is a solution, we solve the unbounded Equivalence Knapsack Problem $Ax^{(0)} = b_m - j$ with profits $c^{(0)}$ and the unbounded Equivalence Knapsack Problems $B_i x^{(i)} = b_i - b_m + j$ with profits $c^{(i)}$ for all $i \in [n]$. It takes $O(\Delta^2 \cdot n)$ time to determine the maximum profit and the corresponding j value. Solving all unbounded Equivalence Knapsack Problems takes $O(n \cdot \Delta^2)$ time by Jansen and Rohwedder [27].

All together, we can solve problem (♠) in time $O(n \cdot \Delta^2 \log(\Delta))$.

Proof of Claim 8. As already stated, all feasible and with this all optimal solutions are of the form $z_0 = x_0 + k \cdot \operatorname{lcm}(v_1, \ldots, v_n)$ with $k \in \mathbb{Z}$. Moreover, z_r is the solution with the biggest first component z_{r_0} by definition. So, for a proof by contradiction the following supposition is sufficient:

Suppose, `optx`' is an optimal solution for problem (♠) with $A \cdot \texttt{optx}'^{(0)} < \gcd(A)(z_{r_0} -$

$\Delta^2 \cdot \mathrm{lcm}(v_1, \ldots, v_n)$).

Let $\mathtt{optb}'_i$ be an optimal solution for the EKP that is represented by B_i with capacity $b_i - A \cdot \mathtt{optx}'^{(0)}$ as written above. Then, $\mathtt{optb}'_i$ consists of an optimal solution for the EKP with capacity $k'_{B_i} \leq \Delta^2$ and of copies of item $I_{e_{B_i}}$. Let $\mathtt{optr}$ be the solution for problem (♠) with $A \cdot \mathtt{optr}^{(0)} = A \cdot \mathtt{optx}'^{(0)} + \gcd(A) \cdot \Delta^2 \cdot \mathrm{lcm}(v_1, \ldots v_n)$. Since $A \cdot \mathtt{optx}'^{(0)} < \gcd(A)(z_{r_0} - \Delta^2 \cdot \mathrm{lcm}(v_1, \ldots, v_n))$ holds by the supposition and Theorem 6, $\mathtt{optr}$ is a feasible solution. Analogously, let $\mathtt{optb}_i$ be an optimal solution for the EKP that is represented by B_i with capacity $b_i - A \cdot \mathtt{optr}^{(0)}$ as written above and let $k_{B_i} \leq \Delta^2$ be the capacity of the EKP of which solution $\mathtt{optb}_i$ partly consists. Let $j = \gcd(A) \cdot \Delta^2 \cdot \mathrm{lcm}(v_1, \ldots, v_n)$. We have $A \cdot \mathtt{optr}^{(0)} = A \cdot \mathtt{optx}'^{(0)} + \gcd(A) \cdot \Delta^2 \cdot \mathrm{lcm}(v_1, \ldots, v_n) \Leftrightarrow A \cdot \mathtt{optx}'^{(0)} = A \cdot \mathtt{optr}^{(0)} - j$. Moreover, since $\gcd(A) \geq 1$ and $\mathrm{lcm}(v_1, \ldots, v_n) \geq 1$ and with this, $j \geq \Delta^2$, we get:

$$
\begin{aligned}
& 2 \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} < \frac{p_{I_{e_A}}}{w_{I_{e_A}}} \\
\Leftrightarrow\ & j \cdot 2 \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} < j \cdot \frac{p_{I_{e_A}}}{w_{I_{e_A}}} \\
\Leftrightarrow\ & j \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} + j \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} < j \cdot \frac{p_{I_{e_A}}}{w_{I_{e_A}}} \\
\Leftrightarrow\ & j \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} + \Delta^2 \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} \leq j \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} + j \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} \\
& < j \cdot \frac{p_{I_{e_A}}}{w_{I_{e_A}}} \qquad (*)
\end{aligned}
$$

Now, we can prove with (*) that $p_{\mathtt{optr}} > p_{\mathtt{optx}'}$. For a better overview, the sequence chain is divided into several sections.

We have $\sum_{i \in [n]} p_{\mathtt{optb}'_i} - \sum_{i \in [n]} p_{\mathtt{optb}_i} \leq \sum_{i \in [n]} {}^{p_{I_{e_{B_i}}}}/_{w_{I_{e_{B_i}}}} \cdot k'_{B_i}$ and thus:

$$
\begin{aligned}
& \sum_{i \in [n]} p_{\mathtt{optb}'_i} - \sum_{i \in [n]} p_{\mathtt{optb}_i} \leq \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} \cdot k'_{B_i} \\
\Leftrightarrow\ & \sum_{i \in [n]} p_{\mathtt{optb}'_i} - \sum_{i \in [n]} p_{\mathtt{optb}_i} - \sum_{i \in [n]} \frac{k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} \\
& \leq \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} \cdot k'_{B_i} - \sum_{i \in [n]} \frac{k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}}
\end{aligned}
$$

Since $k_{B_i} \leq \Delta^2$, we get:

$$
\begin{aligned}
& \sum_{i \in [n]} p_{\mathtt{optb}'_i} - \sum_{i \in [n]} p_{\mathtt{optb}_i} - \sum_{i \in [n]} \frac{k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} + \sum_{i \in [n]} \frac{k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} \\
& \leq \Delta^2 \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}}
\end{aligned}
$$

Now, we can apply **(*)** and the last line is equivalent to:

$$
\begin{aligned}
& j \cdot \sum_{i\in[n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} + \sum_{i\in[n]} p_{\texttt{optb}'_i} - \sum_{i\in[n]} p_{\texttt{optb}_i} - \sum_{i\in[n]} \frac{k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} + \\
& \sum_{i\in[n]} \frac{k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} \leq j \cdot \sum_{i\in[n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} + \Delta^2 \cdot \sum_{i\in[n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} < j \cdot \frac{p_{I_{e_A}}}{w_{I_{e_A}}} \\
\Rightarrow\ & j \cdot \sum_{i\in[n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} + \sum_{i\in[n]} p_{\texttt{optb}'_i} - \sum_{i\in[n]} \frac{k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} - \\
& (\sum_{i\in[n]} p_{\texttt{optb}_i} - \sum_{i\in[n]} \frac{k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}}) < j \cdot \frac{p_{I_{e_A}}}{w_{I_{e_A}}} \\
\Leftrightarrow\ & \sum_{i\in[n]} p_{\texttt{optb}'_i} + \sum_{i\in[n]} \frac{j}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} - \sum_{i\in[n]} \frac{k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} \\
& < \frac{j}{w_{I_{e_A}}} \cdot p_{I_{e_A}} + \sum_{i\in[n]} p_{\texttt{optb}_i} - \sum_{i\in[n]} \frac{k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} \\
\Leftrightarrow\ & - \frac{j}{w_{I_{e_A}}} \cdot p_{I_{e_A}} + \sum_{i\in[n]} p_{\texttt{optb}'_i} + \sum_{i\in[n]} \frac{j - k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} \\
& < \sum_{i\in[n]} p_{\texttt{optb}_i} - \sum_{i\in[n]} \frac{k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}}
\end{aligned}
$$

Here, we add terms at both sides of the inequality:

$$
\begin{aligned}
& \frac{A \cdot \texttt{optr}^{(0)} - k_a}{w_{I_{e_A}}} \cdot p_{I_{e_A}} - \frac{j}{w_{I_{e_A}}} \cdot p_{I_{e_A}} + \sum_{i\in[n]} p_{\texttt{optb}'_i} + \sum_{i\in[n]} \frac{b_i - A \cdot \texttt{optr}^{(0)}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} + \sum_{i\in[n]} \frac{j - k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} \\
& < \frac{A \cdot \texttt{optr}^{(0)} - k_a}{w_{I_{e_A}}} \cdot p_{I_{e_A}} + \sum_{i\in[n]} p_{\texttt{optb}_i} + \sum_{i\in[n]} \frac{b_i - A \cdot \texttt{optr}^{(0)}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} - \sum_{i\in[n]} \frac{k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} \\
& \Leftrightarrow \\
& \frac{(A \cdot \texttt{optr}^{(0)} - j) - k_a}{w_{I_{e_A}}} \cdot p_{I_{e_A}} + \sum_{i\in[n]} (p_{\texttt{optb}'_i} + \frac{b_i - (A \cdot \texttt{optr}^{(0)} - j) - k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}}) \\
& < \frac{A \cdot \texttt{optr}^{(0)} - k_a}{w_{I_{e_A}}} \cdot p_{I_{e_A}} + \sum_{i\in[n]} (p_{\texttt{optb}_i} + \frac{b_i - A \cdot \texttt{optr}^{(0)} - k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}})
\end{aligned}
$$

Since $A \cdot \texttt{optx}'^{(0)} = A \cdot \texttt{optr}^{(0)} - j$, we can go on like this:

$$\frac{A \cdot \texttt{optx}'^{(0)} - k_a}{w_{I_{e_A}}} \cdot p_{I_{e_A}} + \sum_{i \in [n]} (p_{\texttt{optb}'_i} + \frac{b_i - A \cdot \texttt{optx}'^{(0)} - k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}})$$
$$< \frac{A \cdot \texttt{optr}^{(0)} - k_a}{w_{I_{e_A}}} \cdot p_{I_{e_A}} + \sum_{i \in [n]} (p_{\texttt{optb}_i} + \frac{b_i - A \cdot \texttt{optr}^{(0)} - k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}})$$
$$\Leftrightarrow$$
$$p_{\texttt{optx}'} = p_{\texttt{opta}} + \frac{A \cdot \texttt{optx}'^{(0)} - k_a}{w_{I_{e_A}}} \cdot p_{I_{e_A}} + \sum_{i \in [n]} (p_{\texttt{optb}'_i} + \frac{b_i - A \cdot \texttt{optx}'^{(0)} - k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}})$$
$$< p_{\texttt{opta}} + \frac{A \cdot \texttt{optr}^{(0)} - k_a}{w_{I_{e_A}}} \cdot p_{I_{e_A}} + \sum_{i \in [n]} (p_{\texttt{optb}_i} + \frac{b_i - A \cdot \texttt{optr}^{(0)} - k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}}) = p_{\texttt{optr}}$$

Thus, we have $p_{\texttt{optr}} > p_{\texttt{optx}'}$ and with this, $\texttt{optx}'$ cannot be an optimal solution for problem (♠). That is a contradiction. ◀

Proof of Claim 9. As already stated, all feasible and with this all optimal solutions are of the form $z_0 = x_0 + k \cdot \text{lcm}(v_1, \ldots, v_n)$ with $k \in \mathbb{Z}$. Moreover, z_ℓ is the solution with the smallest first component z_{ℓ_0} by definition. So, for a proof by contradiction the following supposition is sufficient:

Suppose, optx' is an optimal solution for problem (♠) with $A \cdot \texttt{optx}'^{(0)} > \text{gcd}(A)(z_{\ell_0} + \Delta^2 \cdot \text{lcm}(v_1, \ldots, v_n))$.

Let $\texttt{optb}'_i$ be an optimal solution for the EKP that is represented by B_i with capacity $b_i - A \cdot \texttt{optx}'^{(0)}$ as written above. Then, $\texttt{optb}'_i$ consists of an optimal solution for the EKP with capacity $k'_{B_i} \leq \Delta^2$ and of copies of item $I_{e_{B_i}}$. Let optl be the solution for problem (♠) with $A \cdot \texttt{optl}^{(0)} = A \cdot \texttt{optx}'^{(0)} - \text{gcd}(A) \cdot \Delta^2 \cdot \text{lcm}(v_1, \ldots, v_n)$. Since $A \cdot \texttt{optx}'^{(0)} > \text{gcd}(A)(z_{\ell_0} + \Delta^2 \cdot \text{lcm}(v_1, \ldots, v_n))$ holds by the supposition and Theorem 6, optl is a feasible solution. Analogously, let $\texttt{optb}_i$ be an optimal solution for the EKP that is represented by B_i with capacity $b_i - A \cdot \texttt{optl}^{(0)}$ as written above and let $k_{B_i} \leq \Delta^2$ be the capacity of the EKP of which solution $\texttt{optb}_i$ partly consists. Let $j = \text{gcd}(A) \cdot \Delta^2 \cdot \text{lcm}(v_1, \ldots, v_n)$. We have $A \cdot \texttt{optl}^{(0)} = A \cdot \texttt{optx}'^{(0)} - \text{gcd}(A) \cdot \Delta^2 \cdot \text{lcm}(v_1, \ldots, v_n) \Leftrightarrow A \cdot \texttt{optx}'^{(0)} = A \cdot \texttt{optl}^{(0)} + j$. Moreover, since $\text{gcd}(A) \cdot \text{lcm}(v_1, \ldots, v_n) \neq 1$ and with this, $\text{gcd}(A) \cdot \text{lcm}(v_1, \ldots, v_n) > 1$, we get:

$$\frac{\text{gcd}(A) \cdot \text{lcm}(v_1, \ldots, v_n)}{(\text{gcd}(A) \cdot \text{lcm}(v_1, \ldots, v_n) - 1)} \cdot \frac{p_{I_{e_A}}}{w_{I_{e_A}}} \leq \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}}$$
$$\Leftrightarrow \text{gcd}(A) \cdot \text{lcm}(v_1, \ldots, v_n) \cdot \frac{p_{I_{e_A}}}{w_{I_{e_A}}} \leq (\text{gcd}(A) \cdot \text{lcm}(v_1, \ldots, v_n) - 1) \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}}$$
$$\Leftrightarrow \Delta^2 \cdot \text{gcd}(A) \cdot \text{lcm}(v_1, \ldots, v_n) \cdot \frac{p_{I_{e_A}}}{w_{I_{e_A}}} \leq \Delta^2 \cdot (\text{gcd}(A) \cdot \text{lcm}(v_1, \ldots, v_n) - 1) \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}}$$
$$\Leftrightarrow j \cdot \frac{p_{I_{e_A}}}{w_{I_{e_A}}} \leq \Delta^2 \cdot \text{gcd}(A) \cdot \text{lcm}(v_1, \ldots, v_n) \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} - \Delta^2 \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}}$$
$$\Leftrightarrow j \cdot \frac{p_{I_{e_A}}}{w_{I_{e_A}}} \leq j \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} - \Delta^2 \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}}$$
$$\Leftrightarrow \frac{j}{w_{I_{e_A}}} \cdot p_{I_{e_A}} \leq \sum_{i \in [n]} \frac{j}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} - \Delta^2 \cdot \sum_{i \in [n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} \qquad (+)$$

Now, we can prove with $(+)$ that $p_{\texttt{optl}} \geq p_{\texttt{optx}'}$. For a better overview, the sequence chain is divided into several sections.

We have $-\sum_{i\in[n]} {}^{p_{I_{e_{B_i}}}}/_{w_{I_{e_{B_i}}}} \cdot k'_{B_i} \leq \sum_{i\in[n]} p_{\texttt{optb}_i} - \sum_{i\in[n]} p_{\texttt{optb}'_i}$ and thus:

$$
\begin{aligned}
& -\sum_{i\in[n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} \cdot k'_{B_i} \leq \sum_{i\in[n]} p_{\texttt{optb}_i} - \sum_{i\in[n]} p_{\texttt{optb}'_i} \\
\Leftrightarrow\ & -\sum_{i\in[n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} \cdot k'_{B_i} + \sum_{i\in[n]} \frac{k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} \\
& \leq \sum_{i\in[n]} p_{\texttt{optb}_i} - \sum_{i\in[n]} p_{\texttt{optb}'_i} + \sum_{i\in[n]} \frac{k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}}
\end{aligned}
$$

Since $k_{B_i} \leq \Delta^2$, we get:

$$
\begin{aligned}
& -\Delta^2 \cdot \sum_{i\in[n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} \\
& \leq \sum_{i\in[n]} p_{\texttt{optb}_i} - \sum_{i\in[n]} p_{\texttt{optb}'_i} + \sum_{i\in[n]} \frac{k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} - \sum_{i\in[n]} \frac{k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}}
\end{aligned}
$$

Now, we can apply $(+)$ and the last line is equivalent to:

$$
\begin{aligned}
& \frac{j}{w_{I_{e_A}}} \cdot p_{I_{e_A}} \leq \sum_{i\in[n]} \frac{j}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} - \Delta^2 \cdot \sum_{i\in[n]} \frac{p_{I_{e_{B_i}}}}{w_{I_{e_{B_i}}}} \\
& \leq \sum_{i\in[n]} p_{\texttt{optb}_i} - \sum_{i\in[n]} p_{\texttt{optb}'_i} + \sum_{i\in[n]} \frac{k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} - \sum_{i\in[n]} \frac{k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} + \\
& \sum_{i\in[n]} \frac{j}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} \\
\Rightarrow\ & \frac{j}{w_{I_{e_A}}} \cdot p_{I_{e_A}} \leq \sum_{i\in[n]} p_{\texttt{optb}_i} - \sum_{i\in[n]} p_{\texttt{optb}'_i} - \sum_{i\in[n]} \frac{k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} + \\
& \sum_{i\in[n]} \frac{k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} + \sum_{i\in[n]} \frac{j}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} \\
\Leftrightarrow\ & \frac{j}{w_{I_{e_A}}} \cdot p_{I_{e_A}} \leq \sum_{i\in[n]} p_{\texttt{optb}_i} - \sum_{i\in[n]} \frac{k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} - \\
& (\sum_{i\in[n]} p_{\texttt{optb}'_i} - \sum_{i\in[n]} \frac{k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}}) + \sum_{i\in[n]} \frac{j}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} \\
\Leftrightarrow\ & \frac{j}{w_{I_{e_A}}} \cdot p_{I_{e_A}} \leq \sum_{i\in[n]} p_{\texttt{optb}_i} - \sum_{i\in[n]} \frac{k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} - \\
& (\sum_{i\in[n]} p_{\texttt{optb}'_i} - \sum_{i\in[n]} \frac{j + k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}}) \\
\Leftrightarrow\ & \frac{j}{w_{I_{e_A}}} \cdot p_{I_{e_A}} + \sum_{i\in[n]} p_{\texttt{optb}'_i} - \sum_{i\in[n]} \frac{j + k'_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}} \\
& \leq \sum_{i\in[n]} p_{\texttt{optb}_i} - \sum_{i\in[n]} \frac{k_{B_i}}{w_{I_{e_{B_i}}}} \cdot p_{I_{e_{B_i}}}
\end{aligned}
$$

Here, we add terms at both sides of the inequality:

$$\frac{A\cdot \texttt{optl}^{(0)}-k_a}{w_{I_{e_A}}}\cdot p_{I_{e_A}}+\frac{j}{w_{I_{e_A}}}\cdot p_{I_{e_A}}+\sum_{i\in[n]} p_{\texttt{optb}'_i}+\sum_{i\in[n]}\frac{b_i-A\cdot \texttt{optl}^{(0)}}{w_{I_{e_{B_i}}}}\cdot p_{I_{e_{B_i}}}-$$

$$\sum_{i\in[n]}\frac{j+k'_{B_i}}{w_{I_{e_{B_i}}}}\cdot p_{I_{e_{B_i}}}$$

$$\leq \frac{A\cdot \texttt{optl}^{(0)}-k_a}{w_{I_{e_A}}}\cdot p_{I_{e_A}}+\sum_{i\in[n]} p_{\texttt{optb}_i}+\sum_{i\in[n]}\frac{b_i-A\cdot \texttt{optl}^{(0)}}{w_{I_{e_{B_i}}}}\cdot p_{I_{e_{B_i}}}-\sum_{i\in[n]}\frac{k_{B_i}}{w_{I_{e_{B_i}}}}\cdot p_{I_{e_{B_i}}}$$

$$\Leftrightarrow \frac{(A\cdot \texttt{optl}^{(0)}+j)-k_a}{w_{I_{e_A}}}\cdot p_{I_{e_A}}+\sum_{i\in[n]}(p_{\texttt{optb}'_i}+\frac{b_i-(A\cdot \texttt{optl}^{(0)}+j)-k'_{B_i}}{w_{I_{e_{B_i}}}}\cdot p_{I_{e_{B_i}}})$$

$$\leq \frac{A\cdot \texttt{optl}^{(0)}-k_a}{w_{I_{e_A}}}\cdot p_{I_{e_A}}+\sum_{i\in[n]}(p_{\texttt{optb}_i}+\frac{b_i-A\cdot \texttt{optl}^{(0)}-k_{B_i}}{w_{I_{e_{B_i}}}}\cdot p_{I_{e_{B_i}}})$$

Since $A\cdot \texttt{optx}'^{(0)}=A\cdot \texttt{optl}^{(0)}+j$, we can go on like this:

$$\frac{A\cdot \texttt{optx}'^{(0)}-k_a}{w_{I_{e_A}}}\cdot p_{I_{e_A}}+\sum_{i\in[n]}(p_{\texttt{optb}'_i}+\frac{b_i-A\cdot \texttt{optx}'^{(0)}-k'_{B_i}}{w_{I_{e_{B_i}}}}\cdot p_{I_{e_{B_i}}})$$

$$\leq \frac{A\cdot \texttt{optl}^{(0)}-k_a}{w_{I_{e_A}}}\cdot p_{I_{e_A}}+\sum_{i\in[n]}(p_{\texttt{optb}_i}+\frac{b_i-A\cdot \texttt{optl}^{(0)}-k_{B_i}}{w_{I_{e_{B_i}}}}\cdot p_{I_{e_{B_i}}})$$

$$\Leftrightarrow p_{\texttt{optx}'}=p_{\texttt{opta}}+\frac{A\cdot \texttt{optx}'^{(0)}-k_a}{w_{I_{e_A}}}\cdot p_{I_{e_A}}+\sum_{i\in[n]}(p_{\texttt{optb}'_i}+\frac{b_i-A\cdot \texttt{optx}'^{(0)}-k'_{B_i}}{w_{I_{e_{B_i}}}}\cdot p_{I_{e_{B_i}}})$$

$$\leq p_{\texttt{opta}}+\frac{A\cdot \texttt{optl}^{(0)}-k_a}{w_{I_{e_A}}}\cdot p_{I_{e_A}}+\sum_{i\in[n]}(p_{\texttt{optb}_i}+\frac{b_i-A\cdot \texttt{optl}^{(0)}-k_{B_i}}{w_{I_{e_{B_i}}}}\cdot p_{I_{e_{B_i}}})=p_{\texttt{optl}}$$

Thus, we have $p_{\texttt{optl}}\geq p_{\texttt{optx}'}$ and with this, $\texttt{optx}'$ cannot be an optimal solution for problem (♠). That is a contradiction.

◀

Case 3.3.2: $\sum_{i\in[n]} p_{I_{e_{B_i}}}/w_{I_{e_{B_i}}} < p_{I_{e_A}}/w_{I_{e_A}}$: Since the most efficient items of the matrices $B_i, i\in[n]$ together are less efficient than the most efficient item of A, the optimal solution has a high value $\texttt{optx}^{(0)}$. We already know k_a and that $k_{B_i}\in[\Delta^2-\Delta,\Delta^2]$. By pigeonhole, we get the same combination of k_{B_i} values after checking Δ^n successive solutions. Together, we get that the value $A\cdot \texttt{optx}^{(0)}\in[\gcd(A)(z_{r_0}-\Delta^n\cdot \operatorname{lcm}(v_1,\ldots,v_n)),\gcd(A)z_{r_0}]$. Thus, we can derive the following bounds for problem (4) (see Figure 1): The lower bound of our problem is $l^T:=(l^{(0)}_{I_{e_A}},l^{(1)},\ldots,l^{(n)})$ with

- $l^{(0)}_{I_{e_A}}=\frac{\gcd(A)(z_{r_0}-\Delta^n\cdot \operatorname{lcm}(v_1,\ldots,v_n))-k_a}{w_{I_{e_A}}}$
- $l^{(i)}_{I_{e_{B_i}}}=\left\lceil\frac{b_i-\gcd(A)\cdot z_{r_0}-\Delta^2}{w_{I_{e_{B_i}}}}\right\rceil$ and $l^{(i)}_j=0$ for all $j\neq I_{e_{B_i}}\in[t-1]_0, i\in[n]$

since $x\in\mathbb{N}_0$. The upper bound of our problem is $u^T:=(u^{(0)}_{I_{e_A}},u^{(1)},\ldots,u^{(n)})$ with

- $u^{(0)}_{I_{e_A}}=\left\lfloor\frac{\gcd(A)\cdot z_{r_0}}{w_{I_{e_A}}}\right\rfloor$
- $u^{(i)}_{I_{e_{B_i}}}=\left\lfloor\frac{b_i-\gcd(A)(z_{r_0}-\Delta^n\cdot \operatorname{lcm}(v_1,\ldots,v_n))}{w_{I_{e_{B_i}}}}\right\rfloor$ and $u^{(i)}_j=w_{I_{e_{B_i}}}$ for all $j\neq I_{e_{B_i}}\in[t-1]_0, i\in[n]$

We have $u_j^{(i)} = w_{I_{e_{B_i}}}$ for all $j \neq I_{e_{B_i}} \in [t-1]_0, i \in [n]$. If $x_j^{(i)} > w_{I_{e_{B_i}}}$ would hold, we could replace $w_{I_{e_{B_i}}}$ many items $I_{j_{B_i}}$ by $w_{I_{j_{B_i}}}$ many items $I_{e_{B_i}}$ and would get a higher profit, a contradiction.

So $\max_{i\in[n]}\{u_i - l_i\} \leq {}^{\gcd(A)\cdot\Delta^n\cdot\operatorname{lcm}(v_1,\ldots,v_n)+\Delta^2}/_{\min\{w_{I_{e_A}}, w_{I_{e_{B_i}}}\}} \leq \Delta^{n+1}\cdot\operatorname{lcm}(v_1,\ldots,v_n)+\Delta^2$ and we can go on like in case 3.3.1.

E.2 Pseudocode

Algorithm 2 2-stage1rowAlgorithm

Require: $\mathcal{A}, b, c$
Ensure: `optx` solves the given 2-stage problem with one row

1: Find of all $x^{(0)}$ with $Ax^{(0)} \leq \Delta^2$ the optimal solution z_0 as described in the first case.
2: $b_j \leftarrow \min_{i\in[n]} b_i$
3: Find of all $x^{(j)}$ with $B_j x^{(j)} \leq \Delta^2$ the optimal solution z_1 as described in the second case.
4: Run a transformation with Frobenius as described in the third case.
5: **for** $k_a \in [\Delta^2 - \Delta, \Delta^2]$ **do**
6: **if** $2 \cdot \sum_{i\in[n]} {}^{p_{I_{e_{B_i}}}}/_{w_{I_{e_{B_i}}}} < {}^{p_{I_{e_A}}}/_{w_{I_{e_A}}}$ **then**
7: Find of all $Ax^{(0)} \in [\gcd(A)(z_{r_0} - \Delta^2 \cdot \operatorname{lcm}(v_1,\ldots,v_n)), \gcd(A)z_{r_0}]$ with $Ax^{(0)} = \gcd(A)(x_0 + k \cdot \operatorname{lcm}(v_1,\ldots,v_n))$ for $k \in \mathbb{Z}$ the optimal solution z_2 as described in the case 3.1 and add z_2 to the list `list`.
8: **else if** $\gcd(A) \cdot \operatorname{lcm}(v_1,\ldots,v_n) \neq 1 \wedge \sum_{i\in[n]} {}^{p_{I_{e_{B_i}}}}/_{w_{I_{e_{B_i}}}} \geq {}^{\gcd(A)\cdot\operatorname{lcm}(v_1,\ldots,v_n)}/_{(\gcd(A)\cdot\operatorname{lcm}(v_1,\ldots,v_n)-1)\,\cdot\, {}^{p_{I_{e_A}}}/_{w_{I_{e_A}}}}$ **then**
9: Find of all $Ax^{(0)} \in [\gcd(A)z_{\ell_0}, \gcd(A)(z_{\ell_0} + \Delta^2 \cdot \operatorname{lcm}(v_1,\ldots,v_n))]$ with $Ax^{(0)} = \gcd(A)(x_0 + k \cdot \operatorname{lcm}(v_1,\ldots,v_n))$ for $k \in \mathbb{Z}$ the optimal solution z_2 as described in the case 3.2 and add z_2 to the list `list`.
10: **else**
11: Find the optimal solution z_2 with the augmentation algorithm by Klein [33] as described in the case 3.3 and add z_2 to the list `list`.
12: **end if**
13: **end for**
14: **return** `optx` as optimal solution of $z_0, z_1,$ `list`

E.3 Examples

Indeed, all three cases of Algorithm 2 are necessary. This show the following examples.

Case 1: $Ax^{(0)} \leq \Delta^2$: Consider $A := (11) \in \mathbb{N}^{1\times1}$, $B_1 := (7) \in \mathbb{N}^{1\times1}$ and $B_2 := (5) \in \mathbb{N}^{1\times1}$ as matrices, $c^T := (0,1,2)$ as cost vector and $b^T := (259, 250)$ as target vector for a problem of the form (♠).

$$\begin{pmatrix} 11 & 7 & 0 \\ 11 & 0 & 5 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 259 \\ 250 \end{pmatrix}$$

Case 2: $B_i x^{(i)} \leq \Delta^2$ holds for at least one $i \in [n]$: Consider $A := (11) \in \mathbb{N}^{1\times1}$, $B_1 := (7) \in \mathbb{N}^{1\times1}$ and $B_2 := (5) \in \mathbb{N}^{1\times1}$ as matrices, $c^T := (1,0,0)$ as cost vector and

$b^T := (264, 264)$ as target vector for a problem of the form (♠).

$$\begin{pmatrix} 11 & 7 & 0 \\ 11 & 0 & 5 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 264 \\ 264 \end{pmatrix}$$

Case 3: $Ax^{(0)} > \Delta^2$ and $B_i x^{(i)} > \Delta^2$ hold for all $i \in [n]$: Consider $A := (11) \in \mathbb{N}^{1\times 1}$, $B_1 := (7) \in \mathbb{N}^{1\times 1}$ and $B_2 := (5) \in \mathbb{N}^{1\times 1}$ as matrices, $c^T := (0, 10, 20)$ as cost vector and $b^T := (258, 257)$ as target vector for a problem of the form (♠).

$$\begin{pmatrix} 11 & 7 & 0 \\ 11 & 0 & 5 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 258 \\ 257 \end{pmatrix}$$

E.4 Residue Tables

Considering unbounded Subset Sum Problems in the feasibility case, one can have the idea to solve the feasibility case with residue tables. These tables can be computed in $\tilde{O}(\Delta)$ time [14]. Let $\{a_1, \ldots, a_k\}$ be the items of an unbounded Subset Sum instance with greatest value $\Delta := \max_{i\in[k]}\{a_i\}$. For $t = 0, \ldots, \Delta - 1$, let n_r be the smallest integer such that $n_r \mod \Delta = r$ and n_r is decomposable; set $n_r = \infty$ if no such integer exists. The *residue table* RT is a one-dimensional table of size Δ where $\mathrm{RT}(r) = n_r$. These definitions are from [4].

In the feasibility case, we need to find a solution $x^T = (x^{(0)}, \ldots, x^{(n)})$ where $x^{(0)}$ is a solution for the unbounded Subset Sum Problem $Ax^{(0)} = t$ and $x^{(i)}$ is a solution for the unbounded Subset Sum Problem $B_i x^{(i)} = b_i - t$, given a target value t. If there exists a target value t such that all these unbounded Subset Sum problems have a solution, then there exists a feasible solution for the feasibility problem Equation (♠). We compute the residue table for A, i.e. $\mathrm{RT}_A(\Delta_A)$ where $\Delta_A := \max_{i\in[s]} a_i$ is the greatest entry of the block A. Moreover, we compute the residue tables for all blocks $B_i, i \in [n]$, i.e. $\mathrm{RT}_{B_i}(\Delta_{B_i})$ where $\Delta_{B_i} := \max_{j\in[t]} B_j^{(i)}$ is the greatest entry of the block B_i. Then, we check, for each entry $j \in \{0, \ldots, \Delta_A - 1\}$, if $\mathrm{RT}_A[j] + \mathrm{RT}_{B_i}[(b_i - \mathrm{RT}_A[j]) \mod \Delta_{B_i}] \leq b_i$ for all $i \in [n]$ holds. If there is an entry $j \in \{0, \ldots, \Delta_A - 1\}$ such that this holds for all $i \in [n]$, then there is a feasible solution for the problem Equation (♠). But this holds only in one direction. If there is a feasible solution for the problem Equation (♠), then it can hold for all entries $j \in \{0, \ldots, \Delta_A - 1\}$ that there exists at least one $i \in [n]$ such that the equation does not hold. Consider for example the block $A = (5)$ and the block $B_1 = (7)$. Then we get the following two residue tables, see Table 2.

Consider $b_1 = 5$. Then $\mathrm{RT}_A[0] + \mathrm{RT}_{B_i}[(5 - \mathrm{RT}_A[0]) \mod \Delta_{B_1}] = 0 + \mathrm{RT}_{B_i}[(5 - 0) \mod \Delta_{B_1}] = 0 + \infty > 5$. This shows that only a lookup operation is not enough to check feasibility for the problem Equation (♠). A solution is to check all these entries like written above but with the addition to subtract all possible multiples of Δ_A: For every entry $j \in \{0, \ldots, \Delta_A - 1\}$ check if there exists a $k \in \mathbb{Z}_{\geq 0}$ such that $\mathrm{RT}_A[j] + \mathrm{RT}_{B_i}[(b_i - \mathrm{RT}_A[j] - k \cdot \Delta_A) \mod \Delta_{B_i}] + k \cdot \Delta_A \leq b_i$ for all $i \in [n]$ holds. Here, we get that there exists a feasible solution for the problem Equation (♠) if and only if this holds. That is because we just try all possibilities. As one can observe, it suffices to check all possible values $k \in [\mathrm{lcm}(\Delta_A, \Delta_{B_1}, \ldots, \Delta_{B_n})]$. If we check higher values k, we would end up in the same rows r_i of the residue tables B_i for $i \in [n]$ which we considered before but with an even lower value to compare to n_{r_i}. However, this procedure is not faster than our algorithm.

$\mathrm{RT}_A(\Delta_A)$		$\mathrm{RT}_{B_1}(\Delta_{B_1})$	
r	n_r	r	n_r
0	0	0	0
1	∞	1	∞
2	∞	2	∞
3	∞	3	∞
4	∞	4	∞
		5	∞
		6	∞

Table 2 Residue Table $\mathrm{RT}_A(\Delta_A)$ for $A = (5)$ and Residue Table $\mathrm{RT}_{B_1}(\Delta_{B_1})$ for $B_1 = (7)$

F Lower Bounds

In this section we prove lower bounds of the 2-stage problem with one row (♠), some for the feasibility version and some for the optimality version. Following the approach from Jansen, Klein and Lassota [24], we use a reduction from the *Non-Unique Remainder problem*: Given numbers $x_1, \ldots, x_{n_{\mathrm{NR}}}, y_1, \ldots, y_{n_{\mathrm{NR}}}, \zeta \in \mathbb{N}$ and pairwise coprime numbers $q_1, \ldots, q_{n_{\mathrm{NR}}} \in \mathbb{N}$, decide whether there exists a number $z \in \mathbb{Z}_{>0}$ with $z \leq \zeta$ satisfying $z \mod q_i \in \{x_i, y_i\}$ for all $i \in [n_{\mathrm{NR}}]$.

▶ **Theorem 20.** *The Non-Unique Remainder problem is reducible to the optimality 2-stage problem with one row of the form (♠) in polynomial time with the properties $n \in O(n_{NR})$, $r = 1$, $s, t, \|c\|_\infty \in O(1)$, $\Delta \in O(\max_{i\in[n]}\{q_i\})$ and $\|b\|_\infty \in O(\zeta)$.*

Proof. We prove this analogously to Theorem 3 of Jansen, Klein and Lassota [24]. Let R be an instance of the Non-Unique Remainder problem and let $n = n_{\mathrm{NR}}$. Consider the following 2-stage problem with one row of the form (♠):

$$\mathcal{A} \cdot x = \begin{pmatrix} 2 & 2q_1 & 2x_1+1 & 2y_1+1 & 0 & \ldots & 0 \\ \vdots & & \ddots & & \vdots & & \vdots \\ 2 & 0 & \ldots & 0 & 2q_n & 2x_n+1 & 2y_n+1 \end{pmatrix} \cdot x = b = \begin{pmatrix} 2\zeta+1 \\ \vdots \\ 2\zeta+1 \end{pmatrix}$$

We set the cost vector $c^T = (0, c^{(1)}, \ldots, c^{(n)})$ with $(c^{(i)})^T = (0, -1, -1)$ for all $i \in [n]$ and $\max\{c^T x\} \geq -n$. This transformation only takes polynomial time regarding the largest occurring number and n. We have $s = r = 1$, $t = 3$ and $n = n_{\mathrm{NR}}$. The largest entry Δ can be bounded by $2\max_{i\in[n]}\{q_i\}$. The cost vector c has constant size and the largest value in the right-hand side is $\|b\|_\infty = 2\zeta + 1$. Now, we prove that this reduction is correct.

$\Rightarrow$ Let R be a yes-instance. Thus, there exists a solution $z^* \leq \zeta$ satisfying all equations. Let $X \in \mathbb{Z}_{\geq 0}$ such that $z^* + X = \zeta$. Let v_i correspond to the remainder that was satisfied in each equation i, i.e. $v_i = x_i$ or $v_i = y_i$, and let $\alpha_i \in \mathbb{Z}_{\geq 0}$ such that $z^* = \alpha_i \cdot q_i + v_i$. Then, $x^T = (x^{(0)}, x^{(1)}, \ldots, x^{(n)})$ with $x^{(0)} = X$ and $(x^{(i)})^T = (\alpha_i, 1, 0)$ if $v_i = x_i$ or $(x^{(i)})^T = (\alpha_i, 0, 1)$ otherwise for all $i \in [n]$ is an optimal solution of our 2-stage IP with one row. We get for each block $i \in [n]$:

$$2 \cdot X + 2q_i \cdot \alpha_i + (2v_i + 1) \cdot 1 = 2(X + q_i \cdot \alpha_i + v_i) + 1 = 2(X + z^*) + 1 = 2\zeta + 1$$

It is optimal because for every feasible solution y we get: Since $2y^{(0)} + 2q_i \cdot y_1^{(i)} \mod 2 = 0$ but $2\zeta + 1 \mod 2 = 1$ and $y \in \mathbb{Z}_{\geq 0}$, we get $y_2^{(i)} > 0$ or/and $y_3^{(i)} > 0$ for all $i \in [n]$.

$\Leftarrow$ Let the 2-stage problem with one row of the form (♠) be a yes-instance. Thus, there exists a solution $(x^*)^T = (x^{*(0)}, x^{*(1)}, \ldots, x^{*(n)})$ with $x^{*(i)} = (x^{*(i)}_1, x^{*(i)}_2, x^{*(i)}_3)$ satisfying $\max\{c^T x^*\} \geq -n$ and $2x^{*(0)} + 2q_i \cdot x^{*(i)}_1 + (2x_i + 1) \cdot x^{*(i)}_2 + (2y_i + 1) \cdot x^{*(i)}_3 = 2\zeta + 1$ for all $i \in [n]$. Since $2x^{*(0)} + 2q_i \cdot x^{*(i)}_1 \mod 2 = 0$ but $2\zeta + 1 \mod 2 = 1$ and $x^* \in \mathbb{Z}_{\geq 0}$, we get $x^{*(i)}_2 > 0$ or/and $x^{*(i)}_3 > 0$ for all $i \in [n]$. Together with $\max\{c^T x^*\} \geq -n$ and $c^T = (0, c^{(1)}, \ldots, c^{(n)})$ with $(c^{(i)})^T = (0, -1, -1)$ for all $i \in [n]$, we get either $x^{*(i)}_2 = 1$ and $x^{*(i)}_3 = 0$ or $x^{*(i)}_2 = 0$ and $x^{*(i)}_3 = 1$ for all $i \in [n]$. Let v_i correspond to the remainder in each equation i, i.e. $v_i = x_i$ if $x^{*(i)}_2 = 1$ and $x^{*(i)}_3 = 0$ and $v_i = y_i$ if $x^{*(i)}_2 = 0$ and $x^{*(i)}_3 = 1$. Thus, we have

$$
\begin{aligned}
&2x^{*(0)} + 2q_i \cdot x^{*(i)}_1 + (2v_i + 1) \cdot 1 = 2\zeta + 1 \\
&\Leftrightarrow 2(x^{*(0)} + q_i \cdot x^{*(i)}_1 + v_i) + 1 = 2\zeta + 1 \\
&\Leftrightarrow x^{*(0)} + q_i \cdot x^{*(i)}_1 + v_i = \zeta
\end{aligned}
$$

for all $i \in [n]$. Define $z = q_i \cdot x^{*(i)}_1 + v_i$. Then, $z \leq \zeta$ and $z \mod q_i \in \{x_i, y_i\}$ for all $i \in [n]$. Thus, z is a solution for R. ◀

With this theorem, we get lower bounds for solving the 2-stage problem with one row of the form (♠) assuming the Exponential Time Hypothesis (ETH), analogously to Jansen, Klein and Lassota [24].

▶ **Corollary 21.** *The 2-stage problem with one row of the form (♠) cannot be solved in time less than $2^{\delta\sqrt{n}}$ for some $\delta > 0$ assuming ETH where n is the number of the blocks.*

Proof. By Theorem 20, we have a reduction from the Non-Unique Remainder problem to the 2-stage problem with one row of the form (♠) in polynomial time with the properties $n \in O(n_{\text{NR}})$, $r = 1$, $s, t, \|c\|_\infty \in O(1)$, $\Delta \in O(\max_{i \in [n]}\{q_i\})$ and $\|b\|_\infty \in O(\zeta)$. Since $n \in O(n_{\text{NR}})$, we get Corollary 26 analogously to the proof of Corollary 1 of Jansen, Klein and Lassota [24]. ◀

▶ **Corollary 4 (✂).** *The 2-stage problem with one row in form (♠) cannot be solved in time less than $2^{\delta\Delta^{1/2-\varepsilon}}$ for some $\delta, \varepsilon > 0$ assuming ETH where Δ is the largest entry of the 2-stage matrix $\mathcal{A}$.*

Proof. By Theorem 20, we have a reduction from the Non-Unique Remainder problem to the 2-stage problem with one row of the form (♠) in polynomial time with the properties $n \in O(n_{\text{NR}})$, $r = 1$, $s, t, \|c\|_\infty \in O(1)$, $\Delta \in O(\max_{i \in [n]}\{q_i\})$ and $\|b\|_\infty \in O(\zeta)$. Consider the proof of Corollary 1 of Jansen, Klein and Lassota [24]: Let n_3 be the number of variables of an instance of the 3-SAT problem. By Theorem 20, we get $\Delta \in O(n_3^2 \log(n_3))$, analogously to the proof of [24]. Hence, if there is an algorithm solving the 2-stage problem with one row of the form (♠) in time less than $2^{\delta\Delta^{1/2-\varepsilon}}$ this would result in the 3-SAT problem to be solved in time less than $2^{\delta(C_1 n_3^2 \log(n_3))^{1/2-\varepsilon}} \leq 2^{\delta(C_2 n_3)^{1-2\varepsilon} \log(n_3)} \leq 2^{\delta C_3 n_3}$ for some constants C_1, C_2, C_3. Setting $\delta_3 \leq \delta/C_3$, this would violate the ETH. ◀

▶ **Corollary 22.** *The feasibility 2-stage problem with one row of the form (♠) with upper bounds cannot be solved in time less than $2^{\delta\Delta^{1/2-\varepsilon}}$ for some $\delta > 0$ assuming ETH where Δ is the largest entry of the 2-stage matrix $\mathcal{A}$. Moreover, it cannot be solved in time less than $2^{\delta\sqrt{n}}$ for some $\delta > 0$ assuming ETH where n is the number of the blocks.*

Proof. Consider the reduction of Theorem 20. We use the cost vector c to upper bound the variables $x_2^{(i)}$ and $x_3^{(i)}$ for all $i \in [n]$. If we use the following upper bound $u^T = (u^{(0)}, u^{(1)}, \ldots, u^{(n)})$ with $u^{(0)} = \zeta$ and $(u^{(i)})^T = (\zeta, 1, 1)$ for all $i \in [n]$, the proof of Theorem 20 works exactly the same without the cost vector. Thus, we get Corollary 22 right away by Corollary 4 and Corollary 26. ◀

For the feasibility 2-stage problem with one row of the form (♠) without upper bounds, we use the following theorem to get lower bounds.

▶ **Theorem 23.** *The Non-Unique Remainder problem is reducible to the feasibility 2-stage problem with one row of the form (♠) in polynomial time with the properties $n \in O(n_{NR})$, $r = s = 1$, t, $\|c\|_\infty \in O(1)$ and $\Delta, \|b\|_\infty \in O(\zeta)$.*

Proof. We prove this analogously to Theorem 3 of Jansen, Klein and Lassota [24]. Let R be an instance of the Non-Unique Remainder problem and let $n = n_{\mathrm{NR}}$. Consider the following feasibility 2-stage problem with one row of the form (♠):

$$\begin{pmatrix} 2 & 2q_1 & 2x_1 + 2\zeta + 1 & 2y_1 + 2\zeta + 1 & 0 & \ldots & 0 \\ \vdots & & \ddots & & \vdots & & \vdots \\ 2 & 0 & \ldots & 0 & 2q_n & 2x_n + 2\zeta + 1 & 2y_n + 2\zeta + 1 \end{pmatrix} \cdot x = \begin{pmatrix} 4\zeta + 1 \\ \vdots \\ 4\zeta + 1 \end{pmatrix}$$

This transformation only takes polynomial time regarding the largest occurring number and n. We have $s = r = 1$, $t = 3$ and $n = n_{\mathrm{NR}}$. The largest entry Δ can be bounded by $O(\zeta)$. The largest value in the right-hand side is $\|b\|_\infty = 4\zeta + 1$. Now, we prove that this reduction is correct.

$\Rightarrow$ Let R be a yes-instance. Thus, there exists a solution $z^* \leq \zeta$ satisfying all equations. Let $X \in \mathbb{Z}_{\geq 0}$ such that $z^* + X = \zeta$. Let v_i correspond to the remainder that was satisfied in each equation i, i.e. $v_i = x_i$ or $v_i = y_i$, and let $\alpha_i \in \mathbb{Z}_{\geq 0}$ such that $z^* = \alpha_i \cdot q_i + v_i$. Then, $x^T = (x^{(0)}, x^{(1)}, \ldots, x^{(n)})$ with $x^{(0)} = X$ and $(x^{(i)})^T = (\alpha_i, 1, 0)$ if $v_i = x_i$ or $(x^{(i)})^T = (\alpha_i, 0, 1)$ otherwise for all $i \in [n]$ is a feasible solution of our 2-stage IP with one row. We get for each block $i \in [n]$:

$$\begin{aligned} &2 \cdot X + 2q_i \cdot \alpha_i + (2v_i + 2\zeta + 1) \cdot 1 = 2(X + q_i \cdot \alpha_i + v_i) + 2\zeta + 1 = 2(X + z^*) + 2\zeta + 1 \\ &= 2\zeta + 2\zeta + 1 = 4\zeta + 1 \end{aligned}$$

$\Leftarrow$ Let the feasibility 2-stage problem with one row of the form (♠) be a yes-instance. Thus, there exists a feasible solution $(x^*)^T = (x^{*(0)}, x^{*(1)}, \ldots, x^{*(n)})$ with $x^{*(i)} = (x^{*(i)}_1, x^{*(i)}_2, x^{*(i)}_3)$ satisfying $2x^{*(0)} + 2q_i \cdot x^{*(i)}_1 + (2x_i + 2\zeta + 1) \cdot x^{*(i)}_2 + (2y_i + 2\zeta + 1) \cdot x^{*(i)}_3 = 4\zeta + 1$ for all $i \in [n]$. Since $2x^{*(0)} + 2q_i \cdot x^{*(i)}_1 \mod 2 = 0$ but $4\zeta + 1 \mod 2 = 1$ and $x^* \in \mathbb{Z}_{\geq 0}$, we get $x^{*(i)}_2 > 0$ or/and $x^{*(i)}_3 > 0$ for all $i \in [n]$. Together with $(2\zeta + 1) \cdot k > 4\zeta + 1$ for $k > 1, k \in \mathbb{N}$, we get either $x^{*(i)}_2 = 1$ and $x^{*(i)}_3 = 0$ or $x^{*(i)}_2 = 0$ and $x^{*(i)}_3 = 1$ for all $i \in [n]$. Let v_i correspond to the remainder in each equation i, i.e. $v_i = x_i$ if $x^{*(i)}_2 = 1$ and $x^{*(i)}_3 = 0$ and $v_i = y_i$ if $x^{*(i)}_2 = 0$ and $x^{*(i)}_3 = 1$. Thus, we have

$$\begin{aligned} &2x^{*(0)} + 2q_i \cdot x^{*(i)}_1 + (2v_i + 2\zeta + 1) \cdot 1 = 4\zeta + 1 \\ &\Leftrightarrow 2(x^{*(0)} + q_i \cdot x^{*(i)}_1 + v_i) + 2\zeta + 1 = 4\zeta + 1 \\ &\Leftrightarrow 2(x^{*(0)} + q_i \cdot x^{*(i)}_1 + v_i) = 2\zeta \\ &\Leftrightarrow x^{*(0)} + q_i \cdot x^{*(i)}_1 + v_i = \zeta \end{aligned}$$

for all $i \in [n]$. Define $z = q_i \cdot x^{*(i)}_1 + v_i$. Then, $z \leq \zeta$ and $z \mod q_i \in \{x_i, y_i\}$ for all $i \in [n]$. Thus, z is a solution for R. ◀

With this theorem, we get lower bounds for solving the feasibility 2-stage problem with one row of the form (♠) assuming the Exponential Time Hypothesis (ETH), analogously to Jansen, Klein and Lassota [24].

▶ **Corollary 24.** *The feasibility 2-stage problem with one row of the form (♠) cannot be solved in time less than $2^{\delta\sqrt{n}}$ for some $\delta > 0$ assuming ETH where n is the number of the blocks.*

Proof. By Theorem 23, we have a reduction from the Non-Unique Remainder problem to the feasibility 2-stage problem with one row of the form (♠) in polynomial time with the properties $n \in O(n_{\text{NR}})$, $r = s = 1$, $t, \|c\|_\infty \in O(1)$ and $\Delta, \|b\|_\infty \in O(\zeta)$. Since $n \in O(n_{\text{NR}})$, we get Corollary 26 analogously to the proof of Corollary 1 of Jansen, Klein and Lassota [24]. ◀

G Lower Bound for 2 rows

In this section we prove lower bounds of the 2-stage problem with two rows, again following the approach from Jansen, Klein and Lassota [24].

▶ **Theorem 25.** *The Non-Unique Remainder problem is reducible to the optimality 2-stage problem with two rows in polynomial time with the properties $n \in O(n_{NR})$, $r = 2$, $s, t, \|c\|_\infty \in O(1)$, $\Delta \in O(\max_{i\in[n]}\{q_i\})$ and $\|b\|_\infty \in O(\zeta)$.*

Proof. We prove this analogously to Theorem 3 of Jansen, Klein and Lassota [24]. Let R be an instance of the Non-Unique Remainder problem and let $n = n_{\text{NR}}$. Consider the following 2-stage problem with two rows:

$$\mathcal{A} \cdot x = \begin{pmatrix} 1 & q_1 & x_1 & y_1 & 0 & \dots & 0 \\ 0 & 0 & 1 & 1 & 0 & \dots & 0 \\ \vdots & & \ddots & & \vdots & & \vdots \\ 1 & 0 & \dots & 0 & q_n & x_n & y_n \\ 0 & 0 & \dots & 0 & 0 & 1 & 1 \end{pmatrix} \cdot x = b = \begin{pmatrix} \zeta \\ 1 \\ \vdots \\ \zeta \\ 1 \end{pmatrix}$$

We set the cost vector $c^T = (0, \dots, 0)$. This transformation only takes polynomial time regarding the largest occurring number and n. We have $r = 2, s = 1, t = 3$ and $n = n_{\text{NR}}$. The largest entry Δ can be bounded by $\max_{i\in[n]}\{q_i\}$. The cost vector c has constant size and the largest value in the right-hand side is $\|b\|_\infty = \zeta$. Now, we prove that this reduction is correct.

⇒ Let R be a yes-instance. Thus, there exists a solution $z^* \leq \zeta$ satisfying all equations. Let $X \in \mathbb{Z}_{\geq 0}$ such that $z^* + X = \zeta$. Let v_i correspond to the remainder that was satisfied in each equation i, i.e. $v_i = x_i$ or $v_i = y_i$, and let $\alpha_i \in \mathbb{Z}_{\geq 0}$ such that $z^* = \alpha_i \cdot q_i + v_i$. Then, $x^T = (x^{(0)}, x^{(1)}, \dots, x^{(n)})$ with $x^{(0)} = X$ and $(x^{(i)})^T = (\alpha_i, 1, 0)$ if $v_i = x_i$ or $(x^{(i)})^T = (\alpha_i, 0, 1)$ otherwise for all $i \in [n]$ is an optimal solution of our 2-stage IP with one row. We get for each block $i \in [n]$:

$$1 \cdot X + q_i \cdot \alpha_i + v_i \cdot 1 = X + z^* = \zeta$$
$$1 \cdot 1 = 1$$

It is optimal because $c^T = (0, \dots, 0)$.

⇐ Let the 2-stage problem with two rows be a yes-instance. Thus, there exists a solution $(x^*)^T = (x^{*(0)}, x^{*(1)}, \dots, x^{*(n)})$ with $x^{*(i)} = (x^{*(i)}_1, x^{*(i)}_2, x^{*(i)}_3)$ satisfying $x^{*(0)} + q_i \cdot x^{*(i)}_1 + x_i \cdot x^{*(i)}_2 + y_i \cdot x^{*(i)}_3 = \zeta$ and $1 \cdot x^{*(i)}_2 + 1 \cdot x^{*(i)}_3 = 1$ for all $i \in [n]$. Since $1 \cdot x^{*(i)}_2 + 1 \cdot x^{*(i)}_3 = 1$

for all $i \in [n]$, we get either ${x^*}_2^{(i)} = 1$ and ${x^*}_3^{(i)} = 0$ or ${x^*}_2^{(i)} = 0$ and ${x^*}_3^{(i)} = 1$ for all $i \in [n]$. Let v_i correspond to the remainder in each equation i, i.e. $v_i = x_i$ if ${x^*}_2^{(i)} = 1$ and ${x^*}_3^{(i)} = 0$ and $v_i = y_i$ if ${x^*}_2^{(i)} = 0$ and ${x^*}_3^{(i)} = 1$. Thus, we have

$$x^{*(0)} + q_i \cdot {x^*}_1^{(i)} + v_i \cdot 1 = \zeta$$

for all $i \in [n]$. Define $z = q_i \cdot {x^*}_1^{(i)} + v_i$. Then, $z \leq \zeta$ and $z \mod q_i \in \{x_i, y_i\}$ for all $i \in [n]$. Thus, z is a solution for R. ◀

With this theorem, we get lower bounds for solving the 2-stage problem with two rows assuming the Exponential Time Hypothesis (ETH), analogously to Jansen, Klein and Lassota [24].

▶ **Corollary 26.** *The 2-stage problem with two rows cannot be solved in time less than $2^{\delta\sqrt{n}}$ for some $\delta > 0$ assuming ETH where n is the number of the blocks.*

Proof. By Theorem 25, we have a reduction from the Non-Unique Remainder problem to the 2-stage problem with two rows in polynomial time with the properties $n \in O(n_{\mathrm{NR}})$, $r = 2$, $s, t, \|c\|_\infty \in O(1)$, $\Delta \in O(\max_{i\in[n]}\{q_i\})$ and $\|b\|_\infty \in O(\zeta)$. Since $n \in O(n_{\mathrm{NR}})$, we get Corollary 26 analogously to the proof of Corollary 1 of Jansen, Klein and Lassota [24]. ◀

▶ **Corollary 27.** *The 2-stage problem with two rows cannot be solved in time less than $2^{\delta\Delta^{1/2-\varepsilon}}$ for some $\delta, \varepsilon > 0$ assuming ETH where Δ is the largest entry of the 2-stage matrix $\mathcal{A}$.*

Proof. By Theorem 25, we have a reduction from the Non-Unique Remainder problem to the 2-stage problem with two rows in polynomial time with the properties $n \in O(n_{\mathrm{NR}})$, $r = 2$, $s, t, \|c\|_\infty \in O(1)$, $\Delta \in O(\max_{i\in[n]}\{q_i\})$ and $\|b\|_\infty \in O(\zeta)$. Consider the proof of Corollary 1 of Jansen, Klein and Lassota [24]: Let n_3 be the number of variables of an instance of the 3-SAT problem. By Theorem 25, we get $\Delta \in O(n_3^2 \log(n_3))$, analogously to the proof of [24]. Hence, if there is an algorithm solving the 2-stage problem with two rows in time less than $2^{\delta\Delta^{1/2-\varepsilon}}$ this would result in the 3-SAT problem to be solved in time less than $2^{\delta(C_1 n_3^2 \log(n_3))^{1/2-\varepsilon}} \leq 2^{\delta(C_2 n_3)^{1-2\varepsilon}\log(n_3)} \leq 2^{\delta C_3 n_3}$ for some constants C_1, C_2, C_3. Setting $\delta_3 \leq \delta/C_3$, this would violate the ETH. ◀