Tackling A Class of Hard Subset-Sum Problems: Integration of Lattice Attacks with Disaggregation Techniques

Bojun Lu *  Duan Li †  Rujun Jiang ‡

August 2019

Abstract

Subset-sum problems belong to the NP class and play an important role in both complexity theory and knapsack-based cryptosystems, which have been proved in the literature to become hardest when the so-called density approaches one. Lattice attacks, which are acknowledged in the literature as the most effective methods, fail occasionally even when the number of unknown variables is of medium size. In this paper we propose a modular disaggregation technique and a simplified lattice formulation based on which two lattice attack algorithms are further designed. We introduce the new concept “jump points” in our disaggregation technique, and derive inequality conditions to identify superior jump points which can more easily cut-off non-desirable short integer solutions. Empirical tests have been conducted to show that integrating the disaggregation technique with lattice attacks can effectively raise success ratios to 100% for randomly generated problems with density one and of dimensions up to 100. Finally, statistical regressions are conducted to test significant features, thus revealing reasonable factors behind the empirical success of our algorithms and techniques proposed in this paper.

Keywords: subset-sum problems, linear Diophantine equations, knapsack-based cryptosystem, lattice attack, density, LLL algorithm, lattice basis reduction, modular disaggregation technique

1 Introduction

1.1 Background

Subset-sum problems defined as follows,

\[ ax := a_1 x_1 + a_2 x_2 + \cdots + a_n x_n = b \] (1)

*School of Data Science, The Chinese University of Hong Kong (Shenzhen), P.R.C. Email: bojunlu@cuhk.edu.cn.
†School of Data Science, City University of Hong Kong, Hong Kong. Email: dli226@cityu.edu.hk.
‡School of Data Science, Fudan University, Shanghai, P.R.C. Email: rjjiang@fudan.edu.cn.
with $a = (a_1, a_2, \ldots, a_n) \in \mathbb{Z}_+^n$, $b \in \mathbb{Z}_+$ and $x \in \mathcal{X} = \{0, 1\}^n$ are important problems in complexity theory and knapsack-based cryptosystems design (see [27], [12], [54], [21], and [20]). Meanwhile subset-sum problems are also a special class of knapsack problems which are important in combinatorial optimization field and always of great interest to researchers (see [25], [44], [45], and [14]). Without loss of generality, we assume that

$$\max\{a_1, a_2, \ldots, a_n\} < b \leq \frac{\sum_{i=1}^{n} a_i}{2}. \tag{2}$$

Otherwise, the complementary problem of (1) defined as follows,

$$a y := a_1 y_1 + a_2 y_2 + \cdots + a_n y_n = \tilde{b} := \sum_{i=1}^{n} a_i - b, \tag{3}$$

with $y_i = 1 - x_i \in \{0, 1\}$, $i = 1, 2, \ldots, n$, satisfies assumption (2).

Identifying the feasibility of any subset-sum problem is NP-complete in general, as the partition problem with $b = \frac{\sum_{i=1}^{n} a_i}{2}$ is NP-complete in its feasibility form (see [17]). Meanwhile, to identify a solution of a feasible subset-sum problem is NP-hard.

A class of hard subset-sum problems can be utilized to design public-key cryptosystems (see [37], [10], [43], and [38]) for transmitting 0-1 information. Lattice attacks, which are the most critical cryptanalysis against knapsack cryptosystems, are proposed (see [7], [27], [12], and [49]) to break knapsack-based cryptosystems with relatively low density, where density is defined as follows,

$$\text{density} = \frac{n}{\max_{1 \leq i \leq n}(\log_2 a_i)}. \tag{4}$$

The literature has revealed that subset-sum problems with their density close to one constitute the hardest subclass of subset-sum problems (see [27], [11], and [50]). A subset-sum problem with density lower than one or higher than one is vulnerable to lattice attacks. Beside of the density feature defined in (4), some other structure features of subset-sum problems have also been proposed in the literature to describe the difficulty level of the problem. Nguyen and Stern (see [40]) defined pseudo-density to generalize the definition of density defined in (4), they claimed that if the value of the pseudo-density is less than a critical value, then the subset-sum problem is vulnerable to lattice attacks, even if the value of density is within the critical range proposed in the literature. Kunihiro (see [26]) introduced the problem structure feature density $D$ which unifies the notion of density and pseudo-density and he also derived conditions under which subset-sum problems are vulnerable to lattice attacks. Jen et al. (see [22] and [23]) also conducted their research study on the reliance of the density feature defined for knapsack cryptosystems.

The disaggregation problem was first proposed by Glover and Woolsey (see [18]) in 1972, and can be described in general as follows: How to decompose the following single Diophantine equation,

$$a x := a_1 x_1 + a_2 x_2 + \cdots + a_n x_n = b, \quad x \in \mathcal{X} \subseteq \mathbb{Z}^n \tag{5}$$

into an equivalent system of two Diophantine equations,

$$\begin{cases} \alpha x := \alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_n x_n = b_1 \\ \beta x := \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n = b_2 \end{cases}, \quad x \in \mathcal{X} \subseteq \mathbb{Z}^n \tag{6}$$
with \(a \in \mathbb{Z}^n_+, \alpha, \beta \in \mathbb{Z}^n, b \in \mathbb{Z}_+, b_1, b_2 \in \mathbb{Z},\) and \(\mathcal{X}\) is a bounded set, under the constriction that the feasible solution sets of (5) and (6) are identical to each other. However, after decades, research studies on the techniques dealing with the disaggregation problem are limited. Mardanov and Mamedov studied the disaggregation problem with unknowns being binary in their paper [33] and [34], and later extended their results with unknowns taking values over a more general but still bounded set in [32].

An extension of subset-sum problems (1) is the so-called linear Diophantine equations (LDEs), which can be presented as follows,

\[
Ax = b,
\]

where \(A \in \mathbb{Z}^{m \times n}\) is a matrix of full row rank, \(b \in \mathbb{Z}^m\), and \(x \in \mathcal{X} \cap \mathbb{Z}^n\) with \(\mathcal{X} = \{x \mid 0 \leq x_i \leq u_i, \ i = 1, 2, \ldots, n\}\). In 2010, Aardal and Wolsey (see [3]) studied and extended formulations for system of LDEs in (7) based on lattice theory, and also obtained a solution scheme for disaggregation problem as a by-product. Beside of the fact of limited studies on disaggregation, we are also inspired by the cell enumeration method proposed by Li et al. (see [29]). Their method solves the linear Diophantine equations with the complexity \(O((n \max\{u_1, \ldots, u_n\})^{n-m})\), which depends on the magnitude of \(n - m\). Hence, increasing the magnitude of \(m\) and thus reducing the magnitude of \(n - m\) directly improve the complexity bound.

As hard subset-sum problems are used in cryptosystem protocol designs (see [51], [42], [24], and [8]), attacking algorithms are proposed to tackle these hard problems, among which lattice attack algorithms are important (see [27], [12], [1], and [50]). Lagarias and Odlyzko (see [27]) proposed the lattice transformation of a subset-sum problem and then used LLL basis reduction (see [28], [30], and [13]) to derive short vectors in the lattice, which can help identify feasible solutions. Coster et al. (see [12]) improved the work proposed in [27] by shifting the lattice, and then can solve even sparser and harder subset-sum problems. Aardal et al. (see [1]) proposed a lattice transformation for the system of linear Diophantine equations with upper and lower bounds on the unknowns, thus to first identify an integer solution, and then to use branch-and-bound methods to further search feasible solutions.

### 1.2 Our motivation and contributions

The following considerations motivate our study in this paper. On one hand, lattice attack algorithms proposed in the literature on subset-sum problems with density approaching one fail occasionally, when the number of unknown variables is only of medium size. On the other hand, disaggregation problem has been proposed for decades but with very limited research studies on it, meanwhile increasing the number of equations can uncover more information and thus benefits the computation. Most importantly, so far there are limited research work on combing lattice attacks with cutting methods, especially with disaggregation techniques.

In this paper, we propose an improved and simplified lattice formation of subset-sum problems and also propose modular disaggregation technique, then novelty integrate lattice formulation and
disaggregation technique. The modular disaggregation technique aims to reveal more information of the given system, and to cut-off the non-binary integer solutions with small Euclidean length initially returned by lattice attack algorithms. Numerical tests support that this mechanism work efficiently, and can increase the probability of returning valid binary solutions.

In this paper, we also introduce and define the concept of “jump points of subset-sum problems”, which specially play an important role in the modular disaggregation technique. Conditions are derived to identify superior jump points which can more easily help cut-off non-desirable non-binary integer solutions with small Euclidean length.

The algorithm \texttt{Reduce}(x_b, D) proposed in this paper, can return the same integer solution as that returned by algorithm \texttt{AHL-Alg}, but with a simpler lattice formulation and with just one big auxiliary integer number \(N\). Moreover, after adopting a one-half modification to \texttt{Reduce}(x_b, D), we further propose algorithm \texttt{Reduce}_{1/2}(x_b, D) in this paper, which can return valid binary solutions with higher probability compared with algorithm \texttt{Reduce}(x_b, D).

Worth mentioning that, Theorem 2 proved in this paper is a more general result compared with the study in Havas et al.’s paper (see [19]) for a single equation.

1.3 Organization

In Section 2, we first have a quick review on the classic LLL algorithm and three important lattice attack algorithms in the literature. In Section 3, we introduce our simplification and improvement on one such lattice attack algorithm, and also propose the modular disaggregation technique. Results of numerical experiments are reported in Section 4. More analysis on the possible mechanisms behind the efficiency of integrating modular disaggregation technique with lattice attack, are summarized in Section 5. Finally, Section 6 contains conclusion and further research.

2 Technical Preliminaries

2.1 Review of the Lattice Basis Reduction Algorithm

In this paper we abbreviate the LLL basis reduction algorithm (see [28], [30], [39], [36], [46], [8], and [13]) to the \textit{LLL algorithm} and name the basis obtained by applying the LLL basis reduction algorithm as the \textit{LLL-reduced basis}. The LLL algorithm is a polynomial time and complexity algorithm which is a milestone algorithm for integer programming problems with fixed dimension. Algebraically speaking, to obtain the LLL-reduced basis, a series of unimodular column operations need to be conducted on an ordered pre-given basis. Geometrically speaking, vectors consisting an LLL-reduced basis are relatively short and nearly orthogonal to one another. In this paper, we use the Euclidean norm of a vector (see [15], [35]) to measure the length of a solution vector. For instance, a well known result has been reviewed in Lemma 1, which is about the upper bound of the length of a vector in the LLL-reduced basis. Figure 1 is an illustration of an arbitrary pre-given
basis and the LLL-reduced basis for the same lattice.

**Definition 1** (lattice, see [6]). Let \( \mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n \in \mathbb{R}^\tilde{n} \) be linearly independent column vectors with \( n \leq \tilde{n} \), the set \( \mathcal{L} \) defined as follows,

\[
\mathcal{L} := \mathbb{Z}\mathbf{b}_1 + \mathbb{Z}\mathbf{b}_2 + \cdots + \mathbb{Z}\mathbf{b}_n := \left\{ \sum_{i=1}^{n} z_i \mathbf{b}_i \mid z_i \in \mathbb{Z}, \ i = 1, 2, \ldots, n \right\},
\]

is called a lattice of dimension \( n \). Moreover, \( \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n\} \) is called a basis for the lattice \( \mathcal{L} \).

**Theorem 1** (see [6]). Given a lattice \( \mathcal{L} \), column vectors of matrix \( \mathbf{B} \) and column vectors of matrix \( \tilde{\mathbf{B}} \) are two equivalent bases for \( \mathcal{L} \), if and only if there exists a unimodular matrix \( \mathbf{U} \), such that \( \mathbf{B} = \tilde{\mathbf{B}} \mathbf{U} \).

![Figure 1: Illustration of an LLL-reduced basis.](image)

In the LLL algorithm, the Gram-Schmidt orthogonalization (GSO) process is a crucial component, which is reviewed in Algorithm 1. A review of the LLL algorithm is presented in Algorithm 2. A more detailed description of the theory, techniques, and applications of the LLL algorithm, can be reached in Bremner’s book [6], and the book [41] edited by Nguyen and Vallée.

The major steps of the LLL algorithm can be described as follows.

- Firstly, the GSO process is conducted on the input ordered basis \( \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n\} \) and an orthogonal basis \( \{\mathbf{b}_1^*, \mathbf{b}_2^*, \ldots, \mathbf{b}_n^*\} \) is obtained as follows,

\[
\begin{align*}
\mathbf{b}_1^* &= \mathbf{b}_1, \\
\mathbf{b}_2^* &= \mathbf{b}_2 - \mu_{1,2} \mathbf{b}_1^*, \quad \mu_{1,2} = \frac{\mathbf{b}_2 \cdot \mathbf{b}_1^*}{\mathbf{b}_1^* \cdot \mathbf{b}_1^*}, \\
&\ldots \\
\mathbf{b}_i^* &= \mathbf{b}_i - \mu_{i-1,i} \mathbf{b}_{i-1}^* - \mu_{i-2,i} \mathbf{b}_{i-2}^* - \cdots - \mu_{1,i} \mathbf{b}_1^*, \quad \mu_{j,i} = \frac{\mathbf{b}_j \cdot \mathbf{b}_i^*}{\mathbf{b}_j \cdot \mathbf{b}_j}, \quad 1 \leq j < i, \\
&\ldots \\
\mathbf{b}_n^* &= \mathbf{b}_n - \mu_{n-1,n} \mathbf{b}_{n-1}^* - \mu_{n-2,n} \mathbf{b}_{n-2}^* - \cdots - \mu_{1,n} \mathbf{b}_1^*.
\end{align*}
\]

- Secondly, two crucial operations which are so-called as ‘Reduce’ and ‘Exchange’ will be applied to the input basis vectors \( \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n\} \) with \( \frac{1}{4} < \alpha < 1 \) being a parameter with pre-given value. Empirically, the closer of \( \alpha \) to 1, the higher quality of the LLL-reduced basis.

  - (Reduce) If \( |\mu_{j,i}| > \frac{1}{2\alpha} \), then \( \mathbf{b}_i \leftarrow \mathbf{b}_i - [\lfloor \alpha \mu_{j,i} \rfloor] \mathbf{b}_j \).
Lemma 2

with another. reducing one basis vector by another, and Lemma 3 tells the property of exchanging one basis vector simplifying computational efforts in the LLL algorithm. Specifically, Lemma 2 tells the property of

Note: $\mu$ which satisfies the following conditions,

\[ |\mu_{j,i}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n, \]
\[ ||b_i^* + \mu_{i-1,i}b_{i-1}^*||^2 \geq \alpha ||b_{i-1}^*||^2, \quad 1 < i \leq n. \]

Lemma 1 (see [28]). If \{ $b_1, b_2, \ldots, b_n$ \} is the LLL-reduced basis with parameter $\alpha$ for the lattice $L \in \mathbb{R}^n$ with $n \leq \hat{n}$. Let $y_1, y_2, \ldots, y_t \in \mathcal{L}$ be any $t$ linearly independent vectors. Then for any $j$ with $1 \leq j \leq t$, the following inequality holds,

\[ ||b_j||^2 \leq \beta^{n-1} \max\{|||y_1||^2, ||y_2||^2, \ldots, ||y_t||^2\} \]

with $\beta = \frac{4}{4\alpha - 1}$.

Algorithm 1 ($M, B^*$) = GSO($B$)

input Matrix $B$, where $b_i$, $i = 1, 2, \ldots, n$, denotes the $i$th column of $B$.
output Matrix $M$ and matrix $B^*$, such that $B = B^*M$ with columns of $B^*$ being orthogonal with one another.

1: $M \leftarrow 0^{n \times n}$
2: for $i = 1 : n$ do
3: \quad $\mu_{i,i} \leftarrow 1$
4: end for
5: $b_1^* \leftarrow b_1$
6: for $i = 2 : n$ do
7: \quad $b_i^* \leftarrow b_i$
8: \quad for $j = 1 : i - 1$ do
9: \quad \quad $\mu_{j,i} \leftarrow \frac{b_ib_j^*}{\gamma b_j^*}$
10: \quad \quad $b_i^* \leftarrow b_i^* - \mu_{j,i}b_j^*$
11: \quad end for
12: end for
13: return $M$ and $B^*$

Note: $\mu_{i,j}$ denotes the $(i,j)$th entry of matrix $M$. $b_i^*$ denotes the $i$th column of matrix $B^*$.

We next review two important lemmas for the GSO process which play important roles in simplifying computational efforts in the LLL algorithm. Specifically, Lemma 3 tells the property of reducing one basis vector by another, and Lemma 4 tells the property of exchanging one basis vector with another.

Lemma 2 (see [6], or Lemma 2.2.1 of [31] with proof). Let $b_1, b_2, \ldots, b_n$ be a basis of the lattice $L \in \mathbb{R}^n$ with $n \leq \hat{n}$. Let $\hat{b}_1, \hat{b}_2, \ldots, \hat{b}_n$ be another basis of the lattice $\hat{L}$, and

\[ \hat{b}_k = b_k - \gamma b_l, \quad \hat{b}_i = b_i \quad (i \neq k, \ 1 \leq i \leq n), \]

where $\gamma \in \mathbb{Z}$ and $1 \leq l < k \leq n$ with $\gamma$, $k$ and $l$ being fixed. Let $b_i^*, \mu_{j,i}, 1 \leq j < i \leq n$ and $\hat{b}_i^*, \hat{\mu}_{j,i}, 1 \leq j < i \leq n$ be the GSO output of these two bases respectively. Then the following properties hold,
Then the following properties hold,

Lemma 3 (see [6], or Lemma 2.2.2 of [31] with proof)

Let \( L \in \mathbb{R} \)

Note: \( \mu_{i,j} \) denotes the \((i,j)\)th entry of \( M \). \( \hat{b}_i \) denotes the \(i\)th column of matrix \( \hat{B} \), and \( \hat{b}_i^* \) denotes the \(i\)th column of matrix \( \hat{B}^* \). In order to save efforts in updating entries of \( M \) in step 5 and step 7, conclusions in Lemma 2 and Lemma 3 can help.

1. \( \hat{b}_i^* = b_i^* \), for all \( i \) with \( 1 \leq i \leq n \).
2. \( \hat{\mu}_{i,j} = \mu_{i,j} \), for all \( i, j \) with \( i \neq k, 1 \leq j < i \leq n \). When \( i = k \), the following is true,

\[
\hat{\mu}_{j,k} = \begin{cases} 
\mu_{j,k} - \gamma \mu_{j,l}, & 1 \leq j < l, \\
\mu_{j,k} - \gamma, & j = l, \\
\mu_{j,k}, & l < j < k.
\end{cases}
\]

Lemma 3 (see [6], or Lemma 2.2.2 of [31] with proof). Let \( b_1, b_2, \ldots, b_n \) be a basis of the lattice \( \mathcal{L} \in \mathbb{R}^n \) with \( n \leq \tilde{n} \). Let \( \tilde{b}_1, \tilde{b}_2, \ldots, \tilde{b}_n \) be another basis of the lattice \( \mathcal{L} \), and

\[
\tilde{b}_{k-1} = b_k, \quad \tilde{b}_k = b_{k-1}, \quad \tilde{b}_i = b_i \quad (1 \leq i \neq k - 1, k \leq n).
\]

Let \( \hat{b}_i^* \), \( \hat{\mu}_{i,j}, 1 \leq j < i \leq n \) and \( \hat{b}_i^*, \hat{\mu}_{i,j}, 1 \leq j < i \leq n \) be the GSO output of the two bases respectively. Then the following properties hold,

1. \( \hat{b}_i^* = b_i^* \) for all \( i \) with \( 1 \leq i \leq n, i \neq k - 1, k \).
2. \( \hat{b}_{k-1}^* = b_k^* + \mu_{k-1,k} b_{k-1}^* \), and \( \hat{b}_k^* = \frac{||b_k^*||^2}{||b_{k-1}^*||^2} b_{k-1}^* - \mu_{k,k-1} ||b_{k-1}^*||^2 b_k^* \).
3. \( \hat{\mu}_{i,j} = \mu_{i,j} \), for all \( i, j \) with \( 1 \leq i \leq n \) and \( i \neq k - 1, i \neq k \), and \( 1 \leq j < i \) and \( j \neq k - 1, j \neq k \).
(d) For all $i$ with $k + 1 \leq i \leq n$, the followings hold,
\[
\hat{\mu}_{i,k} = \mu_{i,k} \frac{||b_k^*||^2 + \mu_{i,k-1} \mu_{k,k-1} ||b_{k-1}^*||^2}{||b_{k-1}^*||^2},
\]
\[
\hat{\mu}_{i,k} = \mu_{i,k} - \mu_{i,k} \mu_{k,k-1}.
\]

(e) $\hat{\mu}_{k-1,j} = \mu_{k,j}$, for all $j$ with $1 \leq j \leq k - 2$.

(f) $\hat{\mu}_{k,j} = \mu_{k-1,j}$, for all $j$ with $1 \leq j \leq k - 2$, and $\hat{\mu}_{k,k-1} = \mu_{k,k-1} \frac{||b_{k-1}^*||^2}{||b_{k-1}^*||^2}$.

### 2.2 Lattice Formulations for Linear Equation Problems

In this section we first briefly illustrate how subset-sum problems and its extension systems of LDEs can be transformed into lattice formulations in the literature. Specifically, we would briefly summarize here how the three important lattice attack algorithms (see [27] [12], and [1]) transform the equation problems into lattice formulations. Lattice formulations proposed in this paper are detailed described and explained in Section 3.1. For convenience, we abbreviate the algorithms proposed in [27] [12], and [1]) as LO-Alg, CJLOSS-Alg, and AHL-Alg, respectively.

In LO-Alg (see [27]), the following matrix of dimension $(n + 1) \times (n + 1)$,
\[
B_{LO} = \begin{pmatrix}
I_{n \times n} & 0^{n \times 1} \\
-a_{1 \times n} & b
\end{pmatrix},
\]
\[\text{(8)}\]
is proposed for problem (1). In CJLOSS-Alg (see [12]), the following matrix of dimension $(n + 1) \times (n + 1)$,
\[
B_{CJLOSS} = \begin{pmatrix}
I_{n \times n} & \frac{1}{2} \times 1^{n \times 1} \\
-a_{1 \times n} & bN \end{pmatrix}
\]
\[\text{(9)}\]
is proposed for problem (1). In AHL-Alg (see [1]), the following matrix of dimension $(n + m + 1) \times (n + 1)$,
\[
B_{AHL} = \begin{pmatrix}
I_{n \times n} & 0^{n \times 1} \\
0^{1 \times n} & N_1 \\
A_{m \times n} & -b^{m \times 1}N_2
\end{pmatrix}
\]
\[\text{(10)}\]
is proposed for problem (7).

In order to use lattice basis reduction theory, columns of matrix $B_{LO}$, columns of matrix $B_{CJLOSS}$, and columns of matrix $B_{AHL}$ are regarded as bases of three different lattices, respectively. Then the LLL reduced bases would be derived, respectively. Specifically, after transforming the subset-sum problem or its extension system of LDEs into lattice problems, the three algorithms LO-Alg, CJLOSS-Alg and AHL-Alg tried to identify the feasible integer solution in the following ways, respectively.
• Let $\tilde{B}_{LO}$ denote the matrix whose columns consist the LLL-reduced basis of the lattice generated by columns of $B_{LO}$. LO-Alg checked whether any $j$th column of $\tilde{B}_{LO}$ with $j \in \{1,2,\ldots,n+1\}$ is of the form that $\tilde{b}_{i,j} \in \{0,\lambda\}$ for all $i \in \{1,2,\ldots,n\}$ for some fixed scalar $\lambda$, and $\tilde{b}_{n+1,j} = 0$. The desired and identified column vector $(b_{1,j},b_{2,j},\ldots,b_{n,j})^T$ is divided by the scalar $\lambda$, which becomes a binary column vector, then we check whether the binary column vector is a feasible binary solution to Problem (11). If no such targeted column vector appears, the procedure can be applied to the Complementary Problem (3) as well, with $b$ replaced by $\tilde{b} := \sum_{i=1}^{n} a_i - b$. An analysis derived for LO-Alg is presented in [16].

• Let $N > \frac{1}{2}\sqrt{n}$, and let $\tilde{B}_{CJLOSS}$ denote the matrix whose columns consist the LLL-reduced basis of the lattice generated by columns of $B_{CJLOSS}$. CJLOSS-Alg checked whether any $j$th column of $\tilde{B}_{CJLOSS}$ with $j \in \{1,2,\ldots,n+1\}$ is of the form that $\tilde{b}_{i,j} \in \{-\frac{1}{2},\frac{1}{2}\}$ for all $i \in \{1,2,\ldots,n\}$, and $\tilde{b}_{n+1,j} = 0$. If no such column vector appears, The desired and identified column vector $(b_{1,j},b_{2,j},\ldots,b_{n,j})^T$ is added by $\frac{1}{2}$, which becomes a binary column vector, then we check whether the binary column vector is a feasible binary solution to Problem (11). If no such targeted column vector appears, the procedure can be applied to the Complementary Problem (3) as well, with $b$ replaced by $\tilde{b} := \sum_{i=1}^{n} a_i - b$.

• Let $N_1 > N_{01}$ and $N_2 > 2^{n+m}N_1^2 + N_{02}$, where $N_{01}$ and $N_{02}$ are big enough finite positive integers and the existence of them can be guaranteed theoretically. Let $\tilde{B}_{AHL}$ denote the matrix whose columns consist the LLL-reduced basis of the lattice generated by columns of $B_{AHL}$. AHL-Alg checked whether the $(n - m + 1)$th column of $\tilde{B}_{AHL}$ is of the form that $|\tilde{b}_{n+1,n-m+1}| = N_1$ and $\tilde{b}_{i,n-m+1} = 0$ for all $i \in \{n+2,n+3,\ldots,n+m+1\}$. If so, then $(\tilde{b}_{1,n-m+1},\tilde{b}_{2,n-m+1},\ldots,\tilde{b}_{n,n-m+1})^T$ must be an integer solution to the problem defined in Equation (7), but may out of the bounded range $[0,u_i]$, $i = 1,2,\ldots,n$ of unknowns. Then branch-and-bound methods are used to enumerate the feasible integer solution.

3 Building Blocks of Our Solution Framework

3.1 The lattice formulation

As we know, subset-sum problems are special cases of systems of linear Diophantine equations (LDEs), thus any lattice formulation for systems of LDEs can be readily applied to subset-sum problems. In this section, we propose our lattice formulation for systems of LDEs, $Ax = b$, in which the lattice is generated by column vectors of matrix $B$ defined as follows,

$$
B = \begin{pmatrix}
I_{n \times n} \\
A_{m \times n} N
\end{pmatrix},
$$

(11)

where $I_{n \times n}$ denotes identity matrix of dimension $n \times n$, $A_{m \times n}$ denotes the coefficient matrix with dimension $m \times n$ in the LDEs problem $Ax = b$ (defined in Equation (7)), and $N$ denotes a large enough and polynomially finite positive integer. In the proof of Theorem 2 the usage of $N$ would be clear.
Theorem 2. There exists polynomially finite $N_0 \in \mathbb{Z}_+$ to guarantee that if $N > N_0$ then the column-wise LLL-reduced matrix $\tilde{B}$ of $B$ defined in (11) has the following form,

\[
\tilde{B} = \begin{pmatrix}
D_{n \times (n-m)} & C_{m \times m} \\
0_{m \times (n-m)} & E_{m \times m} N
\end{pmatrix},
\]

i.e., $\tilde{b}_{i,j} = 0, \forall i, j$ with $n+1 \leq i \leq n + m$ and $1 \leq j \leq n - m$.

Moreover, when $\tilde{b}_{i,j} = 0, \forall i, j$ with $n+1 \leq i \leq n + m$ and $1 \leq j \leq n - m$, then the following properties regarding matrices $D, C,$ and $E$ can be derived.

(a) Matrix $D$ in (12) satisfies that,

\[\ker_{\mathbb{Z}}(A) = \mathcal{L}(D),\]

where $\mathcal{L}(D) := \{Dz \mid z \in \mathbb{Z}^{n-m}\}$ denotes the lattice generated by columns of $D$, and $\ker_{\mathbb{Z}}(A) := \{x \in \mathbb{Z}^n \mid Ax = 0\}$ denotes the kernel lattice of $Ax = b$.

(b) $E^{-1}$ exists, if and only if $A$ is of full row rank.

(c) There exists an integer solution to $Ax = b$ defined in (7) if and only if $E^{-1}b \in \mathbb{Z}^m$ that is $E^{-1}b$ is an integer column vector.

(d) If $E^{-1}b \in \mathbb{Z}^m$, then $x_b := CE^{-1}b$ is a special integer solution to $Ax = b$. Note that, the notation $x_b$ means that the special solution depends on $b$.

Proof. We first prove that $\tilde{b}_{i,j} = 0, \forall i, j$ with $n+1 \leq i \leq n + m$ and $1 \leq j \leq n - m$. Suppose that $AU = (H \mid 0)$, where $U$ is a unimodular matrix and $H$ is the Hermite normal form of $A$, (see [6] for unimodular transformation and the concept of Hermite normal form). Then the last $n - m$ columns of $U$ form a basis of $\ker_{\mathbb{Z}}(A)$. Let us denote the last $n - m$ columns of $U$ as $x_0^1, x_0^2, \ldots, x_0^{n-m}$. Then,

\[v_j := \begin{pmatrix}
x_j^0 \\
0_{m \times 1}
\end{pmatrix} = Bx_j^0 \in \mathcal{L}(B), \quad \forall j \text{ with } 1 \leq j \leq n - m,
\]

and $v_1, v_2, \ldots, v_{n-m}$ are linearly independent. According to Lemma [11] we have that

\[||\tilde{B}_j||^2 \leq 2^{(n-1)} \max\{||v_1||^2, ||v_2||^2, \ldots, ||v_{n-m}||^2\}, \quad \forall j \text{ with } 1 \leq j \leq n - m,
\]

where $\tilde{B}_j$ denotes the $j$th column of $\tilde{B}$.
We choose $N_0$ which satisfies that $N_0^2 > 2^{(n-1)} \max\{|\|v_1\||^2, |\|v_2||^2, \ldots, |\|v_{n-m}\||^2\}$, then when $N > N_0$, we must have that $\tilde{b}_{i,j} = 0 \forall i, j$ with $n + 1 \leq i \leq n + m$ and $1 \leq j \leq n - m$. Otherwise, there exist indices $i$ and $j$ with $n + 1 \leq i \leq n + m$ and $1 \leq j \leq n - m$ such that $\tilde{b}_{i,j} \neq 0$ and it must be a non-zero multiple of $N$, then $|\|B_j||^2 \geq |\|\tilde{b}_{i,j}\||^2 \geq N^2 > N_0^2$, which is a contradiction.

Next, given that $\tilde{b}_{i,j} = 0 \forall i, j$ with $n + 1 \leq i \leq n + m$ and $1 \leq j \leq n - m$, we prove items (a), (b), (c), and (d).

(a) For unimodular matrix $U = [D \mid C] \in \mathbb{Z}^{n \times n}$, we have that

$$\tilde{B} = BU,$$

since the LLL algorithm consists of a sequence of unimodular vector operations. Next we prove item (a) in two directions.

(i) We prove that $\mathcal{L}(D) \subseteq \ker(Z(A))$. Since $AD = 0$, the conclusion is readily obtained.

(ii) We prove that $\ker(Z(A)) \subseteq \mathcal{L}(D)$. For any $x \in \ker(Z(A))$, let $y = U^{-1}x$. Then

$$\mathcal{O}^{n \times 1} = Ax = AUU^{-1}x = AUy = (\mathcal{O}^{n \times (n-m)} \mid E^{m \times m}) y.$$

Hence we have that $E(y_{n-m+1}, y_{n-m+2}, \ldots, y_n)^T = \mathcal{O}^{n \times 1}$. Since $A$ is of full row rank, and $\tilde{b}_{i,j} = 0$ for $n + 1 \leq i \leq n + m$ and $1 \leq j \leq n - m$, $E$ must be a nonsingular matrix. Hence the following holds true,

$$(y_{n-m+1}, y_{n-m+2}, \ldots, y_n)^T = \mathcal{O}^{m \times 1}.$$

Therefore $x = Uy = D(y_1, \ldots, y_m)^T$, which implies that $x \in \mathcal{L}(D)$.

(b) $A$ is of full row rank if and only if the sub-matrix consisting of the last $m$ rows of $B$ is of full row rank. The sub-matrix consisting of the last $m$ rows of $B$ is of full row rank, is equivalent to that the sub-matrix consisting of the last $m$ rows of $\tilde{B}$ is of full row rank. While the sub-matrix consisting of the last $m$ rows of $\tilde{B}$ is of full row rank, is equivalent to that $E$ is of full row rank. As $E$ is a square matrix of dimension $n \times n$. This is equivalent to that $E^{-1}$ exists.

(c) We prove this item in two directions.

i) If $E^{-1}b \in \mathbb{Z}^m$ then $C Ey^{-1}b \in \mathbb{Z}^m$ is an integral solution to problem (7), since $ACE^{-1}b = E E^{-1}b = b$.

ii) We prove that $E^{-1}b$ must be an integer vector, if there exists $x^* \in \mathbb{Z}^n$ such that $Ax^* = b$. Since $A(D^{n \times (n-m)} \mid C^{n \times m}) = (\mathcal{O}^{n \times (n-m)} \mid E^{m \times m})$, and $(D^{n \times (n-m)} \mid C^{n \times m})$ is a unimodular matrix, then based on Theorem 1, $\mathcal{L}(A) = \mathcal{L}(E)$ is deduced. Meanwhile, since $b \in \mathcal{L}(A)$, there must exist $y \in \mathbb{Z}^m$ such that $Ey = b$. Therefore $E^{-1}b = y \in \mathbb{Z}^m$.

(d) This has already been proved in item (c).

Theorem 2 is selected from Theorem 3.1.1 of the Ph.D. thesis [31]. As a note, in Theorem 2 when $m = 1, A$ becomes a row vector with $A = (a_1, a_2, \ldots, a_n)$, and matrix $E$ becomes a scaler. Moreover, $E$ must be the extended greatest common divisor (GCD) of $a_1, a_2, \ldots, a_n$. The proof can be found in [19].
In the next subsection, we will propose two algorithms, \texttt{Reduce}(x_b, D) and \texttt{Reduce}_{1/2}(x_b, D), which are based on our lattice formulation in this part and the results proved in Theorem 2. As a fact, our algorithm \texttt{Reduce}(x_b, D) can achieve the same result compared with the lattice formulation and column reduction procedure in AHL-Alg. But in \texttt{Reduce}(x_b, D) only one big integer number is involved; while two big integers N_1 and N_2, with N_1 > N_{01} and N_2 > 2^{n+m}N_1^2 + N_{02}, must be involved in AHL-Alg. Thus \texttt{Reduce}(x_b, D) is more concise compared with AHL-Alg.

The algorithm \texttt{Reduce}_{1/2}(x_b, D) is an improved version based on \texttt{Reduce}(x_b, D), which will be explained in detail in the next subsection. \texttt{Reduce}_{1/2}(x_b, D) can return binary solutions with significantly higher success ratio.

3.1.1 Algorithms \texttt{Reduce}(x_b, D) and \texttt{Reduce}_{1/2}(x_b, D)

In this section we pursue beyond Theorem 2 and propose two algorithms which are denoted as \texttt{Reduce}(x_b, D) and \texttt{Reduce}_{1/2}(x_b, D), respectively. Both of these two algorithms enable us to get short, size-reduced integer solutions to systems of LDEs, \( Ax = b \), defined in (7). Recall that in this paper we use Euclidean norm of vector to measure the length of a solution vector, which is consistent with the norm used in Lemma 1.

The most ideal output of these two algorithms would be a feasible binary solution to the hard subset-sum problems proposed in Equation (11), when \( m = 1 \).

\begin{algorithm}
\caption{\texttt{sol}(x_b, D) = \texttt{Reduce}(x_b, D)}
\textbf{input} A special solution \( x_b \in \mathbb{Z}^n \) to \( Ax = b \), and an ordered basis \( D \in \mathbb{Z}^{n \times (n-m)} \) of ker\(_2(A)\).
\textbf{output} A reduced short integer solution, denoted as \( \texttt{sol}(x_b, D) \), to \( Ax = b \).
\begin{algorithmic}[1]
\STATE 1: \( G \leftarrow (D \mid x_b)^T \), conduct GSO process on rows of \( G \) to decompose \( G \) as \( G = MG^* \), where \( M = (\mu_{i,j}) \in \mathbb{Q}^{(n-m+1) \times (n-m+1)} \) is a lower triangular matrix and the rows of \( G^* \in \mathbb{Q}^{(n-m+1) \times n} \) are orthogonal to each other.
\STATE 2: \textbf{for} \( j = n - m \) to 1 \textbf{do}
\STATE 3: \hspace{1em} \( \lambda_j \leftarrow [\mu_{n-m+1,j}] \)
\STATE 4: \hspace{1em} \( G_{n-m+1} \leftarrow G_{n-m+1} - \lambda_j G_j \)
\STATE 5: \hspace{1em} \( M_{n-m+1} \leftarrow M_{n-m+1} - \lambda_j M_j \)
\STATE 6: \textbf{end for}
\STATE 7: \( \texttt{sol}(x_b, D) \leftarrow G_{n-m+1}^T \)
\STATE 8: \textbf{return} \( \texttt{sol}(x_b, D) \)
\end{algorithmic}
\end{algorithm}

Note: \( M_j \) denotes the \( j \)th row of \( M \), and \( G_j \) denote the \( j \)th row of \( G \). \( \mu_{i,j} \) denotes the \((i,j)\)th entry of \( M \).

In Algorithm 3 \texttt{Reduce}(x_b, D), the notation \( \texttt{sol}(x_b, D) \) means that the reduced integer solution depends on the input \( x_b \) and \( D \). For example, based on Theorem 2 if \( CE^{-1}b \in \mathbb{Z}^n \), we can let \( x_b = CE^{-1}b \). In our latter numerical implementation, we choose the LLL-reduced basis \( D \) obtained in Equation (12) as part of the input for \texttt{Reduce}(x_b, D), since the LLL-reduced basis has desirable properties as we claimed and reviewed in Section 2.

In fact, the key purpose of \texttt{Reduce}(x_b, D) is to reduce any special integer solution \( x_b \) to \( Ax = b \)
by a basis $D$ of its integer kernel space $\ker_{\mathbb{Z}}(A)$. Mathematically, there exist integer scalars $\lambda_1$, $\lambda_2$, $\ldots$, $\lambda_{n-m}$, such that,

$$\text{sol}(x_b, D) = x_b - D(\lambda_1, \lambda_2, \ldots, \lambda_{n-m})^T.$$ 

As a note, we could also input other improved reduced basis of a lattice compared with the LLL reduced basis, for instances the BKZ reduction algorithm (see [4], [18], [9], and [17]), as the input of $\text{Reduce}(x_b, D)$. The numerical performances of $\text{Reduce}(x_b, D)$ should can be improved accordingly, in the sense that returning binary solution to hard subset-sum problems with higher success ratio. Next we propose an improved variation, denoted as $\text{Reduce}_{1/2}(x_b, D)$, based on $\text{Reduce}(x_b, D)$. The variation algorithm is presented in the following Algorithm 4.

**Algorithm 4** $\text{sol}_{1/2}(x_b, D) = \text{Reduce}_{1/2}(x_b, D)$

- **input** A special solution $x_b \in \mathbb{Z}^n$ to $Ax = b$, and an ordered basis $D \in \mathbb{Z}^{n \times (n-m)}$ of $\ker_{\mathbb{Z}}(A)$.
- **output** A reduced short integer solution, denoted as $\text{sol}_{1/2}(x_b, D)$, to $Ax = b$.

1: $G \leftarrow (2D \mid 2x_b - 1^{n \times 1})^T$, conduct GSO process on rows of $G$ to decompose $G$ as $G = MG^*$, where $M = (\mu_{i,j}) \in \mathbb{Q}^{(n-m+1) \times (n-m+1)}$ is a lower triangular matrix and the rows of $G^* \in \mathbb{Q}^{(n-m+1) \times n}$ are orthogonal to each other.
2: for $j = n - m$ to 1 do
3: $\lambda_j \leftarrow [\mu_{n-m+1,j}]$
4: $G_{n-m+1} \leftarrow G_{n-m+1} - \lambda_j G_j$
5: $M_{n-m+1} \leftarrow M_{n-m+1} - \lambda_j M_j$
6: end for
7: $\text{sol}_{1/2}(x_b, D) \leftarrow \frac{G_{n-m+1}^T 1^{n \times 1}}{2}$
8: **return** $\text{sol}_{1/2}(x_b, D)$

Note: $M_j$ and $G_j$ are the $j$’th row of $M$ and $G$, respectively. $\mu_{i,j}$ is the $(i,j)$th entry of $M$.

In fact, the key purpose of $\text{Reduce}_{1/2}(x_b, D)$ is also to reduce any special integer solution $x_b$ to $Ax = b$ by a basis $D$ of its integer kernel space $\ker_{\mathbb{Z}}(A)$. Another series of integer scalars $\lambda_1$, $\lambda_2$, $\ldots$, $\lambda_{n-m}$ are generated by $\text{Reduce}_{1/2}(x_b, D)$ to yield that,

$$\text{sol}_{1/2}(x_b, D) = x_b - D(\lambda_1, \lambda_2, \ldots, \lambda_{n-m})^T.$$ 

Next we propose Theorem 3 and Theorem 4 to explore how sol($x_b, D$) and sol$_{1/2}$($x_b, D$) respond to the changes in the input vector $x_b$ and matrix $D$. Specifically, Theorem 3 shows that sol($x_b, D$) and sol$_{1/2}$($x_b, D$) will not change accordingly when another special integer solution $x_b$ is used as the input vector. Meanwhile, Theorem 4 shows that sol($x_b, D$) and sol$_{1/2}$($x_b, D$) will also not change accordingly when only directions of the basis vectors in $D$ change.

**Theorem 3** (Theorem 3.1.2 of [31]). Given a basis $D \in \mathbb{Z}^{n \times (n-m)}$ of $\ker_{\mathbb{Z}}(A)$, then

$$\text{sol}(\tilde{y}, D) = \text{sol}(y, D), \text{ and } \text{sol}_{1/2}(\tilde{y}, D) = \text{sol}_{1/2}(y, D),$$

for any $\tilde{y}$, $y \in \{x \in \mathbb{Z}^n \mid Ax = b\}$. 

---

13
Proof. Based on Algorithm 3, \( \text{sol}(\mathbf{y}, D) \) can be expressed as,

\[
\text{sol}(\mathbf{y}, D) = \mathbf{y} - \lambda_{n-m} D_{n-m} - \cdots - \lambda_1 D_1,
\]

and \( \text{sol}(\tilde{\mathbf{y}}, D) \) can be expressed as,

\[
\text{sol}(\tilde{\mathbf{y}}, D) = \tilde{\mathbf{y}} - \tilde{\lambda}_{n-m} D_{n-m} - \cdots - \tilde{\lambda}_1 D_1,
\]

where \( D_i \) denotes the \( i \)-th column of \( D \) and \( \lambda_i, \tilde{\lambda}_i \in \mathbb{Z}, 1 \leq i \leq n-m \). Meanwhile there must exist integer numbers \( z_i, i = 1, 2, \ldots, n-m \), such that \( \tilde{\mathbf{y}} = \mathbf{y} + \sum_{i=1}^{n-m} z_i D_i \), since \( \mathbf{y} \) and \( \tilde{\mathbf{y}} \) are in the same solution space. Later, by showing that \( \tilde{\lambda}_i = \lambda_i + z_i, i = 1, 2, \ldots, n-m \), we prove \( \text{sol}(\tilde{\mathbf{y}}, D) = \text{sol}(\mathbf{y}, D) \).

In the process of computing \( \text{sol}(\mathbf{y}, D) \) and \( \text{sol}(\tilde{\mathbf{y}}, D) \), we use notation \( \mu_{i,j} \) and \( \tilde{\mu}_{i,j} \) respectively in the GSO process in Step 1 of Algorithm 3. It is easy to find that \( \tilde{\mu}_{i,j} = \mu_{i,j}, 1 \leq j < i \leq n-m \). Next we analyze the relation between \( \mu_{n-m+1,j} \) and \( \mu_{n-m+1,j}, j = 1, 2, \ldots, n-m \). Initially, we have,

\[
\tilde{\mu}_{n-m+1,j} = \frac{\tilde{\mathbf{y}} \cdot D_j^*}{D_j^* \cdot D_j^*} = \frac{(\mathbf{y} + \sum_{i=1}^{n-m} z_i D_i) \cdot D_j^*}{D_j^* \cdot D_j^*} = \frac{\mathbf{y} \cdot D_j^*}{D_j^* \cdot D_j^*} + \sum_{i=j}^{n-m} z_i \frac{D_i \cdot D_j^*}{D_j^* \cdot D_j^*},
\]

\[
= \mu_{n-m+1,j} + z_j + \sum_{i=j+1}^{n-m} z_i \mu_{i,j}, \quad j = 1, 2, \ldots, n-m.
\]

Thus, we have,

\[
\tilde{\lambda}_{n-m} = [\tilde{\mu}_{n-m+1,n-m}] = [\mu_{n-m+1,n-m} + z_{n-m}] = [\mu_{n-m+1,n-m}] + z_{n-m} = \lambda_{n-m} + z_{n-m}.
\]

After subtracting \( \lambda_{n-m} D_{n-m} \) from \( \mathbf{y} \) and subtracting \( \tilde{\lambda}_{n-m} D_{n-m} \) from \( \tilde{\mathbf{y}} \), we could update the values of \( \mu_{n-m+1,j} \) and \( \tilde{\mu}_{n-m+1,j}, j = n-m, n-m-1, \ldots, 1 \) as follows,

\[
\mu_{n-m+1,n-m} \leftarrow \mu_{n-m+1,n-m} - \lambda_{n-m},
\]

\[
\mu_{n-m+1,j} \leftarrow \mu_{n-m+1,j} - \lambda_{n-m} \mu_{n-m,j}, \quad j = n-m-1, \ldots, 1,
\]

and

\[
\tilde{\mu}_{n-m+1,n-m} \leftarrow \tilde{\mu}_{n-m+1,n-m} - \tilde{\lambda}_{n-m} = \mu_{n-m+1,n-m} - \lambda_{n-m},
\]

\[
\tilde{\mu}_{n-m+1,j} \leftarrow \tilde{\mu}_{n-m+1,j} - \tilde{\lambda}_{n-m} \tilde{\mu}_{n-m,j}
\]

\[
= \mu_{n-m+1,j} + z_j + \sum_{i=j+1}^{n-m-1} z_i \mu_{i,j} - \lambda_{n-m} \mu_{n-m,j}, \quad j = n-m-1, \ldots, 1.
\]

Note that, \( \mu_{n-m+1,n-m+1} \) and \( \tilde{\mu}_{n-m+1,n-m+1} \) always equal 1.

Therefore after subtracting \( \lambda_{n-m} D_{n-m} \) from \( \mathbf{y} \) and subtracting \( \tilde{\lambda}_{n-m} D_{n-m} \) from \( \tilde{\mathbf{y}} \), we have,

\[
\tilde{\mu}_{n-m+1,n-m} = \mu_{n-m+1,n-m},
\]

\[
\tilde{\mu}_{n-m+1,j} = \mu_{n-m+1,j} + z_j + \sum_{i=j+1}^{n-m-1} z_i \mu_{i,j}, \quad j = n-m-1, \ldots, 1.
\]
Thus, we have,
\[
\tilde{\lambda}_{n-m-1} = [\tilde{\mu}_{n-m+1,n-m-1}] = [\mu_{n-m+1,n-m-1} + z_{n-m-1}]
\]
\[
= [\mu_{n-m+1,n-m-1} + z_{n-m-1}] = \lambda_{n-m-1} + z_{n-m-1}.
\]

Recursively, we subtract \(\lambda_{n-m-1} D_{n-m-1}\) from \(y\) and subtract \(\tilde{\lambda}_{n-m-1} D_{n-m-1}\) from \(\tilde{y}\) and so on, until the last step within which we subtract \(\lambda_i D_i\) from \(y\) and subtract \(\tilde{\lambda}_i D_i\) from \(\tilde{y}\). Conduct recursively similar analyses as that above for deriving \(\tilde{\lambda}_{n-m-1} = \lambda_{n-m-1} + z_{n-m-1}\), we get \(\tilde{\lambda}_i = \lambda_i + z_i\), \(i = n - m - 2, \ldots, 1\). Thus, it is proved that \(\text{sol}(\tilde{y}, D) = \text{sol}(y, D)\). □

**Theorem 4** (Theorem 3.1.3 of [31]). Let \(D\) and \(\tilde{D}\) be two bases of \(\ker_\mathbb{Z}(A)\), with \(\tilde{D}_i = D_i\) or \(-D_i\), and \(\text{sign}(i) = 1\) if \(\tilde{D}_i = D_i\), \(\text{sign}(i) = -1\) if \(\tilde{D}_i = -D_i\), \(i = 1, 2, \ldots, n - m\). Assume that in the reduction process of Algorithm 3, we get parameters \(\tilde{\mu}_{i,j}, \tilde{D}^*_i\) and \(\tilde{x}^*\) for input \(\tilde{D}\) and \(x\), and parameters \(\mu_{i,j}, D^*_i\) and \(x^*\) for input \(D\) and \(x\). Then we have the followings,

(a) In every recursive step, \(\tilde{\mu}_{i,j} = \text{sign}(i) \times \text{sign}(j) \times \mu_{i,j}, 1 \leq j < i \leq n - m + 1\).

(b) \(\tilde{D}^*_i = \text{sign}(i) \times D^*_i, i = 1, 2, \ldots, n - m,\) and \(\tilde{x}^* = x^*\).

(c) \(\text{sol}(x, \tilde{D}) = \text{sol}(x, D)\) and \(\text{sol}_{1/2}(x, \tilde{D}) = \text{sol}_{1/2}(x, D)\).

**Proof.** a), b) Without loss of generality and for simplicity, we can assume that there is only one vector in \(\tilde{D}\) with different sign from vectors in \(D\). For example, \(\tilde{D}_h = -D_h\). If this case can be proved, then the general case can be readily proved as well.

For \(1 \leq j < i \leq h - 1\), we have \(\tilde{\mu}_{i,j} = \mu_{i,j}\) and \(\tilde{D}^*_i = D^*_i\), since \(\tilde{D}_i = D_i, i = 1, 2, \ldots, h - 1\).

For \(i = h\), we have,
\[
\tilde{\mu}_{h,j} = \frac{\tilde{D}_h \cdot \tilde{D}^*_j}{D^*_j \cdot D_j} = \frac{-D_h \cdot D^*_j}{D^*_j \cdot D_j} = -\mu_{h,j}, \quad j = 1, 2, \ldots, h - 1,
\]
and
\[
\tilde{D}^*_h = \tilde{D}_h - \tilde{\mu}_{h,1} \tilde{D}^*_1 - \tilde{\mu}_{h,2} \tilde{D}^*_2 - \cdots - \tilde{\mu}_{h,h-1} \tilde{D}^*_{h-1}
\]
\[
= -D_h - \mu_{h,1} D^*_1 + \mu_{h,2} D^*_2 - \cdots + \mu_{h,h-1} D^*_{h-1}
\]
\[
= -D^*_h.
\]

For \(i = h + 1\), we have,
\[
\tilde{\mu}_{h+1,j} = \frac{\tilde{D}_{h+1} \cdot \tilde{D}^*_j}{D^*_j \cdot D_j} = \frac{D_{h+1} \cdot D^*_j}{D^*_j \cdot D_j} = \mu_{h+1,j}, \quad j = 1, 2, \ldots, h - 1,
\]
\[
\tilde{\mu}_{h+1,h} = \frac{\tilde{D}_{h+1} \cdot \tilde{D}^*_h}{D^*_h \cdot D_h} = \frac{D_{h+1} \cdot (-D^*_h)}{(-D^*_h) \cdot (-D_h)} = -\mu_{h+1,h},
\]
and
\[
\tilde{D}_{h+1} = \tilde{D}_{h+1} - \tilde{\mu}_{h+1,1} \tilde{D}^*_1 - \tilde{\mu}_{h+1,2} \tilde{D}^*_2 - \cdots - \tilde{\mu}_{h+1,h-1} \tilde{D}^*_{h-1} - \tilde{\mu}_{h+1,h} \tilde{D}^*_h \\
= D_{h+1} - \mu_{h+1,1} D^*_1 - \mu_{h+1,2} D^*_2 - \cdots - \mu_{h+1,h-1} D^*_{h-1} - (-\mu_{h+1,h})(-D^*_h) \\
= D^*_{h+1}.
\]

For \(i = h + 2, h + 3, \ldots, n - m\), we have,
\[
\tilde{\mu}_{i,j} = \frac{\tilde{D}_i \cdot \tilde{D}^*_j}{D^*_j \cdot D^*_j} = \frac{D_i \cdot D^*_j}{D^*_j \cdot D^*_j} = \mu_{i,j}, \quad j = 1, 2, \ldots, h - 1, \\
\tilde{\mu}_{i,h} = \frac{\tilde{D}_i \cdot \tilde{D}^*_h}{D^*_h \cdot D^*_h} = \frac{D_i \cdot (-D^*_h)}{(-D^*_h) \cdot (-D^*_h)} = -\mu_{i,h}, \\
\tilde{\mu}_{i,j} = \frac{\tilde{D}_i \cdot \tilde{D}^*_j}{D^*_j \cdot D^*_j} = \frac{D_i \cdot D^*_j}{D^*_j \cdot D^*_j} = \mu_{i,j}, \quad j = h + 1, \ldots, i - 1,
\]

and
\[
\tilde{D}^*_i = \tilde{D}_i - \tilde{\mu}_{i,1} \tilde{D}^*_1 - \tilde{\mu}_{i,2} \tilde{D}^*_2 - \cdots - \tilde{\mu}_{i,h-1} \tilde{D}^*_{h-1} - \tilde{\mu}_{i,h} \tilde{D}^*_h - \cdots - \tilde{\mu}_{i,i-1} \tilde{D}^*_{i-1} \\
= D_i - \mu_{i,1} D^*_1 - \mu_{i,2} D^*_2 - \cdots - \mu_{i,h-1} D^*_{h-1} - (-\mu_{i,h})(-D^*_h) - \cdots - \mu_{i,i-1} D^*_{i-1} \\
= D^*_i.
\]

For \(i = n - m + 1\), we have,
\[
\tilde{\mu}_{n-m+1,j} = \frac{x \cdot \tilde{D}^*_j}{D^*_j \cdot D^*_j} = \frac{x \cdot D^*_j}{D^*_j \cdot D^*_j} = \mu_{n-m+1,j}, \quad j = 1, 2, \ldots, h - 1, \\
\tilde{\mu}_{n-m+1,h} = \frac{x \cdot \tilde{D}^*_h}{D^*_h \cdot D^*_h} = \frac{x \cdot (-D^*_h)}{(-D^*_h) \cdot (-D^*_h)} = -\mu_{n-m+1,h}, \\
\tilde{\mu}_{n-m+1,j} = \frac{x \cdot \tilde{D}^*_j}{D^*_j \cdot D^*_j} = \frac{x \cdot D^*_j}{D^*_j \cdot D^*_j} = \mu_{n-m+1,j}, \quad j = h + 1, \ldots, n - m.
\]

Thus the special case is proved. Each time, we just change the sign of one vector, then the general case in items 1) and 2) can be derived readily.

\(c)\ \text{sol}(x, D)\) can be expressed as
\[
\text{sol}(x, D) = x - \lambda_{n-m} D_{n-m} - \cdots - \lambda_1 D_1,
\]
and \(\text{sol}(x, \tilde{D})\) can be expressed as
\[
\text{sol}(x, \tilde{D}) = x - \tilde{\lambda}_{n-m} \tilde{D}_{n-m} - \cdots - \tilde{\lambda}_1 \tilde{D}_1,
\]
where \(\lambda_i\) and \(\tilde{\lambda}_i\) for \(i = 1, 2, \ldots, n - m\) are integer numbers. We will prove that \(\tilde{\lambda}_i = \text{sign}(i) \times \lambda_i, \ i = n - m, \ldots, 1\), thus to achieve \(\text{sol}(x, \tilde{D}) = \text{sol}(x, D)\).
First, \(\lambda_{n-m} = [\mu_{n-m+1,n-m}]\), and \(\tilde{\lambda}_{n-m} = [\tilde{\mu}_{n-m+1,n-m}] = \text{sign}(n-m) \times [\mu_{n-m+1,n-m}] = \text{sign}(n-m) \times \lambda_{n-m}\).

Next we will show that if \(\tilde{\lambda}_i = \text{sign}(i) \times \lambda_i\), for some \(i \leq n-m\), then it holds that \(\tilde{\lambda}_{i-1} = \text{sign}(i-1) \times \lambda_{i-1}\). If this is the case, then based on mathematical induction, it follows that \(\tilde{\lambda}_i = \text{sign}(i) \times \lambda_i\), \(i = n-m, \ldots, 1\). Thus \(\text{sol}(\mathbf{x}, \tilde{D}) = \text{sol}(\mathbf{x}, D)\) can be proved.

Given that \(\tilde{\lambda}_i = \text{sign}(i) \times \lambda_i\) for some \(i \leq n-m\), in Step 4 and Step 5 of Algorithm 3 subtracting \(\lambda_i\) times \(D_i\) from \(\mathbf{x}\) and \(\tilde{\lambda}_i\) times \(\tilde{D}_i\) from \(\mathbf{x}\), respectively, gives us the updated values for \(\mu_{n-m+1,j}\) and \(\tilde{\mu}_{n-m+1,j},\ j = i, i-1, \ldots, 1\),

\[
\begin{align*}
\mu_{n-m+1,i} &\leftarrow \mu_{n-m+1,i} - \lambda_i, \\
\mu_{n-m+1,j} &\leftarrow \mu_{n-m+1,j} - \lambda_i \mu_{i,j}, \quad j = i - 1, \ldots, 1,
\end{align*}
\]

and

\[
\begin{align*}
\tilde{\mu}_{n-m+1,i} &\leftarrow \tilde{\mu}_{n-m+1,i} - \tilde{\lambda}_i = \text{sign}(i) \times (\mu_{n-m+1,i} - \lambda_i), \\
\tilde{\mu}_{n-m+1,j} &\leftarrow \tilde{\mu}_{n-m+1,j} - \tilde{\lambda}_i \tilde{\mu}_{i,j} = \text{sign}(j) \times (\mu_{n-m+1,j} - \lambda_i \mu_{i,j}), \quad j = i - 1, \ldots, 1.
\end{align*}
\]

Thus, in the next round of the calculation within the loop from Step 2 to Step 6 of Algorithm 3 we get that \(\tilde{\lambda}_{i-1} = [\tilde{\mu}_{n-m+1,i-1}] = [\text{sign}(i-1) \times \mu_{n-m+1,i-1}] = \text{sign}(i-1) \times [\mu_{n-m+1,i-1}] = \text{sign}(i-1) \times \lambda_{i-1}\).

\[\square\]

As a note, we mention here that the tie-breaking issue may arise in the calculation of nearest integer, \([\bullet]\). For example, should we round \([4.5]\) to 4 or to 5; and should we round \([-4.5]\) to -4 or to -5? There are different tie-breaking rules in the literature (see [53] and Appendix A of [31]). Later in our numerical simulation part, we will adopt the same tie-breaking rule as that in AHL-Alg (see [1]). That is, \([\mu_{jk}] = [\mu_{jk} - \frac{1}{2}]\), then \([4.5]\) would be rounded to 4, and \([-4.5]\) would be rounded to -5.

Also as a note, in the proof for item (a) and item (b) in Theorem 4, tie-breaking rule is adopted as that, if \([4.5]\) is rounded to 4 then \([-4.5]\) is rounded to -4 but not -5; or if \([4.5]\) is rounded to 5 then \([-4.5]\) should be rounded to -5 but not -4. More detailed discussions about the tie-breaking issues can be found in Appendix A of [31].

### 3.2 Modular disaggregation technique

In this section we propose modular disaggregation techniques (abbreviated as “DAG” in this paper) for subset-sum problems defined in Equation (1). In this text we always assume that \(b \leq \frac{1}{2} \sum_{i=1}^{n} a_i\). Otherwise, setting \(y_i = 1 - x_i,\ i = 1, 2, \ldots, n\), yields the complementary subset-sum problem,

\[
ay := a_1y_1 + a_2y_2 + \cdots + a_ny_n = \sum_{i=1}^{n} a_i - b = \bar{b},
\]
with \( \bar{b} < \frac{1}{2} \sum_{i=1}^{n} a_i \) and \( y = (y_1, y_2, \ldots, y_n) \in \{0,1\}^n \). As a note, since \( \bar{b} < b \), after transformation some \( a_i \) may be larger than \( \bar{b} \) and thus the corresponding \( y_i \) can be fixed at zero value.

In our modular disaggregation techniques, firstly two positive integer parameters, \( t \) and \( M \), will be introduced. Then after detailed analysis and deduction, we would see that only one rational parameter \( r := \frac{t}{M} \) is sufficient. Equipped with disaggregation techniques, more equations and more information can be revealed for a given equation system.

Let \( t \) and \( M \) be two positive integers with \( t < M \), for a given subset-sum problem (1), modular transformations are conducted as follows,

\[
\begin{align*}
  c_i &:= ta_i \pmod{M}, \quad i = 1, 2, \ldots, n \\
  d &:= tb \pmod{M} \\
  v_i &:= \left\lfloor \frac{t}{M} a_i \right\rfloor, \quad i = 1, 2, \ldots, n \\
  w &:= \left\lfloor \frac{t}{M} b \right\rfloor
\end{align*}
\]  

(13)

with the notation \( \left\lfloor \bullet \right\rfloor \) denoting the floor truncate integer of a rational number, for instance, \( \left\lfloor \frac{2}{3} \right\rfloor = 2 \) and \( \left\lfloor -\frac{2}{3} \right\rfloor = -3 \). It is not hard to observe that \( 0 \leq d < M \), \( 0 \leq w < b \), and \( 0 \leq c_i < M \), \( 0 \leq v_i < a_i \) for \( i = 1, 2, \ldots, n \). The relationship between \( c \) and \( v \), and between \( d \) and \( w \), can be derived as follows,

\[
c_i = ta_i - Mv_i, \quad i = 1, 2, \ldots, n, \quad \text{(14)}
\]

and

\[
d = tb - Mw. \quad \text{(15)}
\]

For a given subset-sum problem (1), any \( x \in \{0,1\}^n \) satisfying \( ax = b \) must also satisfy the following modular equation,

\[
cx := \sum_{i=1}^{n} c_i x_i \equiv d \pmod{M}, \quad \text{(16)}
\]

and the following algebraic equation,

\[
cx := \sum_{i=1}^{n} c_i x_i = d + Mk, \quad \text{with } k \in \mathbb{Z}_+, \quad \text{(17)}
\]

with \( c = (c_1, c_2, \ldots, c_n) \). Substituting (14) and (15) into (17) yields that,

\[
vx := \sum_{i=1}^{n} v_i x_i = w - k, \quad \text{(18)}
\]

with \( v = (v_1, v_2, \ldots, v_n) \).

In the following theorem, we derive an upper bound for the non-negative integer \( k \) introduced in the algebraic equations (17) and (18).
**Theorem 5.** If \( x \in X = \{0,1\}^n \), then for the integer \( k \) introduced in (17) and (18), an upper bound \( u_k \) which depends on \( \frac{t}{M} \), can be derived as follows,

\[
u_k \left( \frac{t}{M} \right) = \left\lfloor \tilde{b} \frac{t}{M} \right\rfloor + \left\lfloor b \frac{t}{M} \right\rfloor - \sum_{i=1}^{n} \left\lfloor a_i \frac{t}{M} \right\rfloor ,
\]

(19)

where \( \tilde{b} := \sum_{i=1}^{n} a_i - b \) is the right-hand-side of the complementary problem (3).

**Proof.** Firstly, \( k \) must be greater than or equal to zero which is implied from (16) and (17). Secondly, based on (17), the following inequality can be derived,

\[
k = \frac{(c_1x - d)}{M} \leq \sum_{i=1}^{n} \frac{c_i}{M} - \frac{d}{M},
\]

as \( x \in \{0,1\}^n \). The fact that \( k \) is an integer number implies that,

\[
k \leq \left\lfloor \sum_{i=1}^{n} \frac{c_i}{M} - \frac{d}{M} \right\rfloor .
\]

(20)

We denote the term within the floor function in (20) as \( g(\frac{t}{M}) \), that is,

\[
g(\frac{t}{M}) := \sum_{i=1}^{n} \frac{c_i}{M} - \frac{d}{M}.
\]

Substituting (14) and (15) into the above expression gives rise to the expression of \( g(\frac{t}{M}) \) as follows,

\[
g(\frac{t}{M}) = \tilde{b} \frac{t}{M} + \left\lfloor b \frac{t}{M} \right\rfloor - \sum_{i=1}^{n} \left\lfloor a_i \frac{t}{M} \right\rfloor ,
\]

(21)

and thus an upper bound of \( k \) as follows,

\[
u_k \left( \frac{t}{M} \right) = \left\lfloor g(\frac{t}{M}) \right\rfloor = \left\lfloor \tilde{b} \frac{t}{M} \right\rfloor + \left\lfloor b \frac{t}{M} \right\rfloor - \sum_{i=1}^{n} \left\lfloor a_i \frac{t}{M} \right\rfloor .
\]

\( \square \)

Based on the expression of \( u_k \left( \frac{t}{M} \right) \) derived in Theorem 5 and the fact that \( k \) is non-negative, Corollary 1 can be derived. The inequality derived in Corollary 1 is neat and elegant, which exhibits an unadorned, plain, and important relation among coefficients, \( a_i \)'s, \( b \) and \( \tilde{b} \), of the given subset-sum problem and its complementary problem.

**Corollary 1.** Given a subset-sum problem (11) and its complementary problem (3), the following inequality can be derived,

\[
\left\lfloor r\tilde{b} \right\rfloor + \left\lfloor rb \right\rfloor - \sum_{i=1}^{n} \left\lfloor ra_i \right\rfloor \geq 0,
\]

(22)

with \( r \in [0,1] \cap \mathbb{R} \).
Next in Theorem 6, three equivalent conditions are derived, under which \( k \equiv 0 \) holds. In fact, \( k \) depends on \( r := \frac{t}{M} \) as well. When we choose proper values of \( t \) and \( M \), thus proper value of \( r := \frac{t}{M} \), one more equation is revealed for the given subset-sum problem, but meanwhile no more unknown variables have been introduced. This is the most ideal scenario.

**Theorem 6.** The following three inequalities are equivalent to each other,

(a) \( \sum_{i=1}^{n} c_i < M + d \),

(b) \( g(\frac{t}{M}) < 1 \),

(c) \( u_k(\frac{t}{M}) = 0 \).

Proof. (a) \( \iff \) (b). Because \( g(\frac{t}{M}) = (\sum_{i=1}^{n} c_i - d)/M \) and \( M > 0 \).

(b) \( \iff \) (c). Because \( u_k(\frac{t}{M}) = \lfloor g(\frac{t}{M}) \rfloor \) and \( u_k(\frac{t}{M}) \geq 0 \).

\( \square \)

The following example illustrates the most ideal situation that modular disaggregation techniques can achieve, that is when \( k \equiv 0 \) under some specific chosen values of \( t/M \).

**Example 1.** We consider the example in Part II of Merkle and Hellman’s work in 1978 (see [37]) with

\[ a = (171, 196, 457, 1191, 2410) \quad \text{and} \quad b = 3797. \]

(i) Let \( M_1 = 4426 \) and \( t_1 = 79 \). Set then \( c^{(1)} = t_1 a \pmod{M_1} = (231,2206,695,1143,72) \) and \( d_1 = t_1 b \pmod{M_1} = 3421 \).

(ii) Let \( M_2 = 4348 \) and \( t_2 = 69 \). Set then \( c^{(2)} = t_2 c^{(1)} \pmod{M_2} = (2895,34,127,603,620) \) and \( d_2 = t_2 d_1 \pmod{M_2} = 1257 \).

(iii) Let \( M_3 = 4280 \) and \( t_3 = 3 \). Set then \( c^{(3)} = t_3 c^{(2)} \pmod{M_3} = (125,102,381,1809,1860) \) and \( d_3 = t_3 d_2 \pmod{M_3} = 3771 \).

(iv) Let \( M_4 = 4278 \) and \( t_4 = 5 \). Set then \( c^{(4)} = t_4 c^{(3)} \pmod{M_4} = (625,510,1905,489,744) \) and \( d_4 = t_4 d_3 \pmod{M_4} = 1743 \).

Here, \( c^{(j)} = (c_1^{(j)}, c_2^{(j)}, \ldots, c_5^{(j)}) \) with \( j = 1, 2, 3, 4 \). It is easy to verify that \( \sum_{i=1}^{5} c_i^{(j)} < M_j + d_j \), for all \( j = 1, 2, 3, 4 \). Therefore, based on Theorem 6, for any binary \( x \in \{0, 1\}^5 \), the following subset-sum problem,

\[ a x = b \]

is equivalent to the following system of linear equations,

\[ E x = F, \]

20
where matrix $E$ and column vector $F$ can be represented as,

$$
E = \begin{pmatrix}
a \\
c^{(1)} \\
c^{(2)} \\
c^{(3)} \\
c^{(4)}
\end{pmatrix}, \quad F = \begin{pmatrix}
b \\
d_1 \\
d_2 \\
d_3 \\
d_4
\end{pmatrix}.
$$

Since the rank of $E$ equals 5, we can solve and obtain the binary solution as

$$
x = E^{-1}F = (0, 1, 0, 1)^T.
$$

In the general scenario, for arbitrary value of $r = t/M$, based on the upper bound $u_k$ derived in Theorem 5, we further decompose the corresponding $k$ into its binary representation form,

$$
k = k_1 + 2k_2 + 4k_3 + \cdots + 2^{n(k)} - 1 k_n^{(k)},
$$

where $k_1, \ldots, k_n^{(k)} \in \{0, 1\}$ and

$$
n^{(k)} = \lceil \log_2(u_k + 1) \rceil
$$

with $u_k$ being a compact notation for $u_k(t/M)$. Substituting (23) into (18) yields the following disaggregated system with two Diophantine equations,

$$
\begin{cases}
ax = b \\
vx + k_1 + 2k_2 + 4k_3 + \cdots + 2^{n(k)} - 1 k_n^{(k)} = w
\end{cases}
$$

with $x_1, \ldots, x_n, k_1, \ldots, k_n^{(k)} \in \{0, 1\}$. As a note, later in this paper, the bold symbol $k$ will be used to denote the decomposed vector $(k_1, k_2, \ldots, k_n^{(k)})^T$.

Next in Lemma 4 we establish the relation between the original subset-sum problem (1) and the new system (25) obtained via our disaggregation techniques.

**Lemma 4.** $x^*$ is a binary solution to (1) if and only if there exists $k^*$ such that $(x^*, k^*)$ is a binary solution to (25).

Proof. The logic is clear and the proof is readily obtained. □

### 3.2.1 Jump points

We re-write down the expressions of functions $g(t/M)$ and $u_k(t/M)$ as follows,

$$
g(t/M) = b t/M + \left\lfloor b t/M \right\rfloor - \sum_{i=1}^{n} \left\lfloor a_i t/M \right\rfloor,
$$
\[ u_k \left( \frac{t}{M} \right) = \left\lfloor \frac{b}{M} \right\rfloor + \left\lfloor \frac{b}{M} \right\rfloor - \sum_{i=1}^{n} \left\lfloor \frac{a_i t}{M} \right\rfloor , \]

and also the disaggregation equation \((18)\), that is \(vx + k = w\), as follows,

\[ \sum_{i=1}^{n} \left\lfloor \frac{a_i}{M} \right\rfloor x_i + k = \left\lfloor \frac{b}{M} \right\rfloor . \]

We could see that, \(g, u_k, v,\) and \(w\) are all functions which depend on the parameter \(\frac{t}{M}\). As a note, \(v = (v_1, v_2, \ldots, v_n)\) is a vector function of \(\frac{t}{M}\) with \(v_i = \lfloor a_i \frac{t}{M} \rfloor\). Basic observations are presented in the following.

**Observation 1.**

(a) \(v\) is discontinuous if and only if the parameter \(r = \frac{t}{M}\) takes value at any of the points, \(\frac{j}{a_i}\), with \(j = 1, 2, \ldots, a_i - 1\) and \(i = 1, 2, \ldots, n\).

(b) \(w\) is discontinuous if and only if the parameter \(r = \frac{t}{M}\) takes value at any of the points, \(\frac{j}{b}\), with \(j = 1, 2, \ldots, b - 1\).

(c) \(u_k\) is discontinuous if and only if the parameter \(r = \frac{t}{M}\) takes value at any of the points, \(\frac{j}{a_i}\), with \(j = 1, 2, \ldots, a_i - 1\) and \(i = 1, 2, \ldots, n\), or \(\frac{j}{b}\), with \(j = 1, 2, \ldots, b - 1\), or \(\frac{j}{\tilde{b}}\), with \(j = 1, 2, \ldots, \tilde{b} - 1\), with \(\tilde{b} := \sum_{i=1}^{n} a_i - b\).

Based on Observation 1, the concept **jump points of subset-sum problems** is introduced and defined in Definition 2. Later some basic properties and benefits of these **jump points** are derived in Section 3.2.2 and Section 3.2.3.

**Definition 2.** [Jump points of subset-sum problems] Consider coefficients in Problem \((1)\), the following points,

(a) rational numbers \(\frac{j}{a_i}\), \(j = 1, 2, \ldots, a_i - 1\), \(i = 1, 2, \ldots, n\); and

(b) rational numbers \(\frac{j}{b}\), \(j = 1, 2, \ldots, b - 1\); and

(c) rational numbers \(\frac{j}{\tilde{b}}\), \(j = 1, 2, \ldots, \tilde{b} - 1\), with \(\tilde{b} := \sum_{i=1}^{n} a_i - b\),

are called **jump points** of subset-sum problems.

In fact, \(v, w,\) upper bound \(u_k\) of \(k,\) and number \(n^{(k)}\) of the newly introduced unknowns in the disaggregated equation, are all piecewise linear functions. Only at jump points of subset-sum problems, coefficients of the disaggregated equation,

\[ vx + uk = w \]

jump and change, with \(u = (1, 2, \ldots, 2^{n^{(k)}-1})\) and \(k = (k_1, k_2, \ldots, k_{n^{(k)}})^T \in \{0, 1\}^{n^{(k)}}\).

### 3.2.2 Cutting-off short integer solutions

In this section, we introduce our idea of trying to cut-off some non-binary integer solutions to Problem \((1)\) with small Euclidean lengths, thus to achieve the ultimate goal that to increase the probability of returning binary solutions to the given subset-sum problems.
The disaggregation techniques introduced previously can actually divide the feasible solution set of Problem (1) into several subsets, where $k$ introduced in the disaggregated equation plays an important role. Next we introduce and define five sets, denoted as $i$, $i = 1, 2, \ldots, 5$, which are generated during the disaggregation process, as follows,

1. $\{x \in \mathbb{Z}^n \mid ax = b\}$,
2. $\{x \in \mathbb{Z}^n \mid ax = b, \ vx = w - k, \ k \in \mathbb{Z}\}$,
3. $\{x \in \mathbb{Z}^n \mid ax = b, \ vx = w - k, \ k \in \mathbb{Z} \text{ and } 0 \leq k \leq u_k\}$,
4. $\{x \in \{0, 1\}^n \mid ax = b\}$,
5. $\{x \in \{0, 1\}^n \mid ax = b, \ vx = w - k, \ k \in \mathbb{Z} \text{ and } 0 \leq k \leq u_k\}$.

As a note,

(a) Set 3 depends on the value of $r = \frac{t}{M}$ and thus is also denoted as $3| r$ for clarity purpose;

(b) There may exist $k^* \in \{0, 1, \ldots, u_k\}$, such that, in terms of $x$, the integer solution set of the following system of two equations,

$$\begin{cases}
ax = b \\
vx = w - k^*
\end{cases}$$

is empty.

The set inclusion relation among these 5 sets can be described in Figure 2, which neatly illustrate that,

$$5 \subset 4 \subset 3 \subset 2 = 1.$$ 

![Figure 2: Inclusion relation.](image)

If lattice attack algorithms are applied to Problem (1), sometimes the returned solution is not binary although the Euclidean length of it is small. Actually the returned solution generally belongs to set 1. However, the desired binary solution should always belong to set 4. Our target now is to generate the desired set $3| r$ that could increase the probability of returning binary solutions.

Next we use a concrete example to illustrate how disaggregation techniques can help generate desired set $3| r$, thus can cut-off the initially returned non-binary integer solutions with small Euclidean lengths. Thereafter, binary solutions are successfully found.
Example 2. We use the following toy problem
\[ a = (3, 15, 6), \quad b = 9 \]
to illustrate how some values of \( \frac{t}{M} \) can help generate desired set \( \{3\}|r \), thus to cut-off the initially returned non-binary solutions with small Euclidean lengths. Thereafter successfully search and return the binary solution.

Solution.

(a) A general representation of all integer solutions to \( ax = b \) is presented as below,
\[ x = (1, 0, 1)^T + \lambda_1(-2, 0, 1)^T + \lambda_2(-1, 1, -2)^T \quad \text{with} \quad \lambda_1, \lambda_2 \in \mathbb{Z}, \]
and thus \( 1 = 2 = \{x \mid x = (1, 0, 1)^T + \lambda_1(-1, 1, -2)^T, \lambda_1, \lambda_2 \in \mathbb{Z}\} \).

(b) \( \tilde{x} = (0, 1, -1)^T \) is a non-binary solution to the given problem, with small Euclidean length, which is initially returned via the lattice attack algorithm, for instance, Algorithm 3.

(c) The given problem has only one binary solution \((1, 0, 1)^T\), and thus \( 4 = 5 = \{(1, 0, 1)^T\}\).

(d) In our modular disaggregation process, different values of \( r := \frac{t}{M} \) result in different sets \( \{3\}|r \), for example,
(i) Let \((t, M) = (3, 6), \ r = 3/6 = 1/2, \) then \( u_k = 0, \ v = (1, 7, 3), \) and \( w = 4. \) Thus \( \{3\}_{1/2} = \{x \mid x = (1, 0, 1)^T + \lambda_1(-1, 1, -2)^T, \lambda_1 \in \mathbb{Z}\} \), and,
\[ \tilde{x} = (0, 1, -1)^T = (1, 0, 1)^T + (-1, 1, -2)^T \in \{3\}_{1/2}. \]
(ii) Let \((t, M) = (6, 15), \ r = 6/15 = 2/5, \) then \( u_k = 0, \ v = (1, 6, 2), \) and \( w = 3. \) Thus \( \{3\}_{2/5} = \{x \mid x = (1, 0, 1)^T + \lambda_1(-2, 0, 1)^T, \lambda_1 \in \mathbb{Z}\} \), and,
\[ \tilde{x} = (0, 1, -1)^T \notin \{3\}_{2/5}. \]

(e) After disaggregation, under \( r = \frac{2}{5} \), now Algorithm 3 can successfully return the binary solution \( x = (1, 0, 1)^T. \)

Proposition 1. Given a non-binary solution \( \tilde{x} \) of subset-sum problem \( \text{(1)} \), if value of the parameter \( r = \frac{t}{M} \) is chosen such that,
\[ w - \nu \tilde{x} > u_k \quad \text{or} \quad w - \nu \tilde{x} < 0, \quad (26) \]
then there must have,
\[ \tilde{x} \notin \{3\}|r. \]

Proposition 1 summarizes the conditions, under which after disaggregation process, a given non-binary solution with small Euclidean length must can be excluded from the new feasible solution set \( \{3\}|r. \)
3.2.3 Neighbouring jump points

In the last section, the idea of cutting-off a given non-binary integer solution $\tilde{x}$ with small Euclidean length has been introduced. In this section, we investigate further the properties of jump points of subset-sum problems, and study the relations between two neighbouring jump points (NJPs).

Consider all the jump points of subset-sum problems defined in Definition 2, which are ordered in sequence based on their magnitudes. Then NJPs are the two jump points that just next to one another. Basic theorems will be derived, and some notations are introduced as follows to facilitate our analysis,

$$\Delta v := v^{(2)} - v^{(1)}, \quad \Delta w := w^{(2)} - w^{(1)}, \quad \text{and} \quad \Delta \tilde{w} := \tilde{w}^{(2)} - \tilde{w}^{(1)},$$

(27)

where

$$v^{(i)} := [ar_i] = ([a_1r_i], [a_2r_i], \ldots, [a_nr_i]) \in \mathbb{Z}^n, \quad i = 1, 2,$$

$$\Delta v = (\Delta v_1, \Delta v_2, \ldots, \Delta v_n),$$

and

$$w^{(i)} := [br_i], \quad \tilde{w}^{(i)} := [\tilde{b}r_i], \quad i = 1, 2,$$

with $\tilde{b} = \sum_{i=1}^{n} a_i - b$, and $r_2 > r_1 \in (0, 1)$ being two NJPs.

Next we conduct analysis on the difference between upper bounds $u_k^{(1)}$ and $u_k^{(2)}$ of two neighbouring jump points $r_1$ and $r_2$. The difference between $u_k^{(2)}$ and $u_k^{(1)}$ is defined as follows,

$$\Delta u_k := u_k^{(2)} - u_k^{(1)} = \Delta \tilde{w} + \Delta w - \sum_{i=1}^{n} \Delta v_i,$$

(28)

For a given non-binary integer vector $\tilde{x}$ and jump point $r_i$, the corresponding value of $\tilde{k}^{(i)}$ can be calculated as follows,

$$\tilde{k}^{(i)} := w^{(i)} - v^{(i)} \tilde{x}, \quad i = 1, 2,$$

from which the difference between $\tilde{k}^{(2)}$ and $\tilde{k}^{(1)}$ can be derived as follows,

$$\Delta \tilde{k} := \tilde{k}^{(2)} - \tilde{k}^{(1)} = \Delta w - \Delta v \tilde{x}.$$

(29)

Now it is ready to present our observations and basic theorems regarding NJPs. Consider coefficients of the disaggregated equations $v^{(1)} x + k^{(1)} = w^{(1)}$ and $v^{(2)} x + k^{(2)} = w^{(2)}$, our observations are summarized in Observation 2 and Observation 3.

Observation 2. If $r_2 = \hat{a}_k$ and $r_1$ is the left-hand-side (LHS) NJP of $r_2$, then we have the following observations.

(a) $\Delta v \in \{0, 1\}^n$.

(b) $\Delta v_h = 1$. 

25
(c) For $1 \leq i \neq h \leq n$, $\Delta v_i = 1$ if and only if $\frac{j}{a_h} = \frac{j}{a_i}$ for some $j \in \{1, 2, \ldots, a_i - 1\}$.

(d) $\Delta w \in \{0, 1\}$.

(e) $\Delta w = 1$ if and only if $\frac{j}{b} = \frac{j}{a}$ for some $j \in \{1, 2, \ldots, b - 1\}$.

Observation 3. If $r_2 = \frac{j}{b}$ and $r_1$ is the LHS NJP of $r_2$, then we have the following observations.

(a) $\Delta v \in \{0, 1\}^n$.

(b) For $1 \leq i \leq n$, $\Delta v_i = 1$ if and only if $\frac{j}{b} = \frac{j}{a}$ for some $j \in \{1, 2, \ldots, a_i - 1\}$.

(c) $\Delta w = 1$.

Next, we derive Theorem 7 and Theorem 8 which are about the properties of NJPs, in terms of cutting-off $\tilde{x}$, where $\tilde{x}$ denotes a given non-binary integer solution to Problem (1) with small Euclidean length.

Theorem 7. Given two NJPs $r_1, r_2 \in (0, 1)$ with $r_1 < r_2$, and given a non-binary integer solution $\tilde{x}$ to subset-sum problem (1), then we have the following three equivalent statements.

(a) $\Delta w \leq \Delta v \tilde{x} \leq \sum_{i=1}^{n} \Delta v_i - \Delta \tilde{w}$.

(b) $\tilde{x} \in \{0, 1\} | r_2$ $\Rightarrow$ $\tilde{x} \in \{0, 1\} | r_1$.

(c) $\tilde{x} \notin \{0, 1\} | r_1$ $\Rightarrow$ $\tilde{x} \notin \{0, 1\} | r_2$.

Proof. (a) $\Leftrightarrow$ (b). We have that,

$$\tilde{x} \in \{0, 1\} | r_2 \Leftrightarrow \tilde{k}^{(2)} \in \{0, 1, \ldots, u^{(2)}_k\}$$

$$\Leftrightarrow \tilde{k}^{(2)} - \Delta w + \Delta v \tilde{x} \in \{-\Delta w + \Delta v \tilde{x}, 1 - \Delta w + \Delta v \tilde{x}, \ldots, u^{(2)}_k - \Delta w + \Delta v \tilde{x}\}.$$ From (29) and (28), we have that,

$$\tilde{k}^{(1)} = \tilde{k}^{(2)} - \Delta w + \Delta v \tilde{x}, \text{ and } u^{(1)}_k = u^{(2)}_k - \Delta \tilde{w} + \sum_{i=1}^{n} \Delta v_i - \Delta w,$$

which implies that,

$$\tilde{x} \in \{0, 1\} | r_2 \Leftrightarrow \tilde{k}^{(1)} \in \{-\Delta w + \Delta v \tilde{x}, 1 - \Delta w + \Delta v \tilde{x}, \ldots, u^{(1)}_k + \Delta \tilde{w} - \sum_{i=1}^{n} \Delta v_i + \Delta v \tilde{x}\}.$$ Meanwhile, we have that,

$$\tilde{x} \in \{0, 1\} | r_1 \Leftrightarrow \tilde{k}^{(1)} \in \{0, 1, \ldots, u^{(1)}_k\}.$$
Based on the derivations shown above, it is now ready to obtain that,
\[
\tilde{x} \in \{0\}|r_2 \implies \tilde{x} \in \{0\}|r_1
\]
\[
\Leftrightarrow \tilde{k}^{(1)} \subseteq \{-\Delta w + \Delta v\tilde{x}, 1 - \Delta w + \Delta v\tilde{x}, \ldots, u_k^{(1)} + \Delta \tilde{w} - \sum_{i=1}^{n} \Delta v_i + \Delta v\tilde{x}\} \implies \tilde{k}^{(1)} \subseteq \{0, 1, \ldots, u_k^{(1)}\}
\]
\[
\Leftrightarrow \{-\Delta w + \Delta v\tilde{x}, 1 - \Delta w + \Delta v\tilde{x}, \ldots, u_k^{(1)} + \Delta \tilde{w} - \sum_{i=1}^{n} \Delta v_i + \Delta v\tilde{x}\} \subseteq \{0, 1, \ldots, u_k^{(1)}\}
\]
\[
\Leftrightarrow -\Delta w + \Delta v\tilde{x} \geq 0 \text{ and } \Delta \tilde{w} - \sum_{i=1}^{n} \Delta v_i \leq 0
\]
\[
\Leftrightarrow \Delta w \leq \Delta v\tilde{x} \leq \sum_{i=1}^{n} \Delta v_i - \Delta \tilde{w}.
\]
(b) \Leftrightarrow (c). By contradiction, it is straightforward.

\[\square\]

**Theorem 8.** Given two NJPs \(r_1, r_2 \in (0, 1)\) with \(r_1 < r_2\), and given a non-binary integer solution \(\tilde{x}\) to subset-sum problem (1), then we have the following three equivalent statements.

(a) \(\sum_{i=1}^{n} \Delta v_i - \Delta \tilde{w} \leq \Delta v\tilde{x} \leq \Delta w\).

(b) \(\tilde{x} \in \{0\}|r_1 \Rightarrow \tilde{x} \in \{0\}|r_2\).

(c) \(\tilde{x} \notin \{0\}|r_2 \Rightarrow \tilde{x} \notin \{0\}|r_1\).

Proof. (a) \Leftrightarrow (b). We have that,
\[
\tilde{x} \in \{0\}|r_1 \Leftrightarrow \tilde{k}^{(1)} \subseteq \{0, 1, \ldots, u_k^{(1)}\}
\]
\[
\Leftrightarrow \tilde{k}^{(1)} + \Delta w - \Delta v\tilde{x} \subseteq \{\Delta w - \Delta v\tilde{x}, 1 + \Delta w - \Delta v\tilde{x}, \ldots, u_k^{(1)} + \Delta w - \Delta v\tilde{x}\}
\]
\[
\Leftrightarrow \tilde{k}^{(2)} \subseteq \{\Delta w - \Delta v\tilde{x}, 1 + \Delta w - \Delta v\tilde{x}, \ldots, u_k^{(2)} - \Delta \tilde{w} + \sum_{i=1}^{n} \Delta v_i - \Delta v\tilde{x}\}
\]

Meanwhile, we have that,
\[
\tilde{x} \in \{0\}|r_2 \Leftrightarrow \tilde{k}^{(2)} \subseteq \{0, 1, \ldots, u_k^{(2)}\}
\]

Based on the derivations shown above, it is now ready to obtain that,
\[
\tilde{x} \in \{0\}|r_1 \Rightarrow \tilde{x} \in \{0\}|r_2
\]
\[
\Leftrightarrow \tilde{k}^{(2)} \subseteq \{\Delta w - \Delta v\tilde{x}, 1 + \Delta w - \Delta v\tilde{x}, \ldots, u_k^{(2)} - \Delta \tilde{w} + \sum_{i=1}^{n} \Delta v_i - \Delta v\tilde{x}\} \Rightarrow \tilde{k}^{(2)} \subseteq \{0, 1, \ldots, u_k^{(2)}\}
\]
\[
\Leftrightarrow \{\Delta w - \Delta v\tilde{x}, 1 + \Delta w - \Delta v\tilde{x}, \ldots, u_k^{(2)} - \Delta \tilde{w} + \sum_{i=1}^{n} \Delta v_i - \Delta v\tilde{x}\} \subseteq \{0, 1, \ldots, u_k^{(2)}\}
\]
\[
\Leftrightarrow \Delta w - \Delta v\tilde{x} \geq 0, \text{ and } -\Delta \tilde{w} + \sum_{i=1}^{n} \Delta v_i - \Delta v\tilde{x} \leq 0
\]
\[
\Leftrightarrow \sum_{i=1}^{n} \Delta v_i - \Delta \tilde{w} \leq \Delta v\tilde{x} \leq \Delta w.
\]
(b) ⇔ (c). By contradiction, it is straightforward.

Theorem 7 proofs the equivalent conditions, under which if the jump point $r_1$ can help cut-off $\tilde{x}$, then its RHS NJP $r_2$ must can as well. Therefore, the condition derived in Theorem 7 actually can identify that when the incumbent jump point is stronger than its RHS NJP. Meanwhile, Theorem 8 proofs the equivalent conditions, under which if the jump point $r_2$ can help cut-off $\tilde{x}$, then its LHS NJP $r_1$ must can as well. Therefore, the condition derived in Theorem 8 actually can identify that when the incumbent jump point is stronger than its LHS NJP. Here, $\tilde{x}$ normally represents a vector with short Euclidean length, which easily leads the possible failure of lattice attack algorithms in searching a valid binary solution to subset-sum problems.

4 Numerical Tests: Integration of Modular Disaggregation Technique with Lattice Attacks

4.1 Systems with Single Subset-Sum Equation

In this section, plenty of subset-sum problems with density one are randomly generated to test our algorithms proposed in this paper. Our progress is achieved by invoking the integration of modular disaggregation technique (DAG) with lattice attack algorithms CJLOSS-Alg and Reduce$_{1/2}$, respectively. Figure 3 is used to illustrate the procedure. Later, numerical results are summarized in Table 1, Table 2, and Table 3. Our numerical experiment confirms that the success ratio of finding valid binary solutions to the hard subset-sum problems with density one can increase dramatically, by applying this integration procedure.

Figure 3: Illustration for the process of “DAG + lattice attack”.
The designation method of how to randomly generate systems with single subset-sum equation, which are in the form of Problem (1), is described as follows,

1. 100 systems are randomly generated with dimension \( n = 16, 20, 26, 30, 36, 40, 50, 60, \) and 70, respectively;

2. \( a_i, i = 1, 2, \ldots, n, \) follows discrete uniform distribution on the interval \([1, 2^n]\), where \( a_i \) is the \( i \)th entry of \( a; \)

3. \( x \) is randomly generated with cardinality \( n/2 \), which is fixed for the systems with the same dimension \( n; \)

4. \( density := \frac{\sum_{i=1}^{n} \log_2 a_i}{n} \in (0.99, 1.00); \)

5. \( b := ax, \) which satisfies that \( b > \max(a) \) and \( b \leq \sum(a)/2. \)

The reason for generating \( x \) with cardinality \( n/2 \) is that these systems are even more difficult than the other systems with the same dimension \( n. \) Otherwise the information contained in the binary solution is sparse, either for the original problem (1) or for the complementary problem (3).

| \( n \) | \( Reduce \) | \( \text{Reduce}_{1/2} \) | CJLOSS-Alg | LO-Alg | AHL-Alg |
|---|---|---|---|---|---|
| 16 | 26% | 67% | 100% | 56% | 26% |
| 20 | 15% | 44% | 99% | 37% | 15% |
| 26 | 6% | 15% | 84% | 11% | 6% |
| 30 | 2% | 10% | 58% | 7% | 2% |
| 36 | 0% | 4% | 20% | 1% | 0% |
| 40 | 0% | 1% | 8% | 0% | 0% |
| 50 | 0% | 0% | 0% | 0% | 0% |
| 60 | 0% | 0% | 0% | 0% | 0% |
| 70 | 0% | 0% | 0% | 0% | 0% |

| \( n \) | \( DAG + \text{Reduce}_{1/2} \) | Average value of valid \( t \) searched | \( M \) |
|---|---|---|---|
| 16 | 100% | 4.939 | \( 10^3 \) |
| 20 | 100% | 4.875 | \( 10^4 \) |
| 26 | 100% | 6.682 | \( 10^4 \) |
| 30 | 100% | 22.222 | \( 10^4 \) |
| 36 | 100% | 76.354 | \( 10^5 \) |
| 40 | 100% | 216.899 | \( 10^5 \) |
Table 3: Computational performance of “DAG + CJLOSS-Alg”.

| n  | DAG + CJLOSS-Alg | Average value of valid t searched | M   |
|----|------------------|----------------------------------|-----|
| 20 | 100%             | 1.000                            | $10^4$ |
| 26 | 100%             | 2.125                            | $10^4$ |
| 30 | 100%             | 4.310                            | $10^4$ |
| 36 | 100%             | 22.200                           | $10^5$ |
| 40 | 100%             | 114.141                          | $10^5$ |

Table 3 reports the computational performance of variant lattice attacks including the revisited methods proposed in the literature and the two algorithms proposed in this paper, where column 2 to column 6 record the success ratio when calling different lattice attack algorithms.

Table 2 and Table 3 report the computational performance of integrating modular disaggregation technique with algorithms \texttt{Reduce}_{1/2} and \texttt{CJLOSS-Alg}, respectively. The reason of choosing these two lattice attack algorithms to integrate with modular disaggregation technique is that based on data reported in Table 1, they are with the better performance among variant lattice attack algorithms. We would like to further enhance their performance via integrating with module disaggregation technique.

Specifically, in both Table 2 and Table 3: 1). Column 1 records the number of unknown variables; 2). Column 2 with algorithm’s name records the success ratio which is the number of successful problems divided by the total number of tested problems; 3). Column 3 records the average value of valid $t$ that have been searched, only concerning initially failed problems; 4). Column 4 records the fixed value of $M$. Note that, the value of $\frac{t}{M}$ is just the value of parameter $r$ introduced in Section 3.2.

The codes used in this section are implemented in C++ computer language, utilizing packages in the C++ library named NTL (see [52]) which is the most cutting-edge library for doing number theory and for dealing with arbitrarily large integer numbers. For parameter setting, we set $N = 10^8$ in the matrices $B_{CJLOSS}$ and $B$ defined in Equation (9) and Equation (11), respectively, and $\alpha = \frac{99}{100}$ in the LLL algorithm (refer to Algorithm 2 in this paper).

4.2 Systems with Multiple Subset-Sum Equations

In this section, systems with multiple hard subset-sum equations are further tested, which are formulated in Problem (7). These systems are with $m$ equations and $n$ unknown variables. We invoke again the procedure of integrating our modular disaggregation technique (DAG) with lattice attack algorithms \texttt{CJLOSS-Alg} and \texttt{Reduce}_{1/2}, respectively. Figure 4 is used to illustrate the procedure. Later, numerical results are summarized in Table 4, Table 5, and Table 6. Our numerical experiment confirms that the success ratio of finding valid binary solutions to the systems with multiple hard subset-sum equations can increase dramatically, by applying this integration procedure.
The designation method of how to randomly generate systems with multiple subset-sum equations, which are in the form of Problem (7), is described as follows,

1). 100 systems are randomly generated for each fixed dimension \((m, n)\);

2). \(A_{i,j}, i=1,2,\ldots,m, j=1,2,\ldots,n\), follows discrete uniform distribution on the interval \([1,2^n]\), where \(A_{i,j}\) is the \(i\)th row and \(j\)th column entry of \(A\);

3). \(x\) is randomly generated with cardinality \(n/2\), which is fixed for the systems with the same dimension \(n\);

4). \(density_i := \frac{n}{\max_{1 \leq j \leq n}(\log_2 A_{i,j})} \in (0.99, 1.00), i=1,2,\ldots,m\);

5). \(b := Ax\), which satisfies that \(b_i > \max(A_i)\) and \(b_i \leq \text{sum}(A_i)/2, i=1,2,\ldots,m\), where \(b_i\) is the \(i\)th entry of \(b\) and \(A_i\) is the \(i\)th row of \(A\).
Table 4: Systems with multiple subset-sum equations.

| $m$ | $n$ | $\text{Reduce}_{1/2}$ | CJLOSS-Alg |
|-----|-----|------------------------|------------|
| 2   | 30  | 100%                   | 100%       |
| 2   | 40  | 94%                    | 100%       |
| 2   | 50  | 59%                    | 100%       |
| 2   | 60  | 24%                    | 84%        |
| 2   | 70  | 5%                     | 30%        |
| 2   | 80  | 0%                     | 3%         |
| 2   | 90  | 0%                     | 0%         |
| 2   | 100 | 0%                     | 0%         |
| 3   | 40  | 100%                   | 100%       |
| 3   | 50  | 100%                   | 100%       |
| 3   | 60  | 99%                    | 100%       |
| 3   | 70  | 97%                    | 100%       |
| 3   | 80  | 82%                    | 100%       |
| 3   | 90  | 47%                    | 94%        |
| 3   | 100 | 10%                    | 55%        |
| 4   | 60  | 100%                   | 100%       |
| 4   | 70  | 100%                   | 100%       |
| 4   | 80  | 100%                   | 100%       |
| 4   | 90  | 100%                   | 100%       |
| 4   | 100 | 100%                   | 100%       |
| 5   | 100 | 100%                   | 100%       |
| 6   | 100 | 100%                   | 100%       |
| 7   | 100 | 100%                   | 100%       |
| 8   | 100 | 100%                   | 100%*      |
| 9   | 100 | 100%*                  | 100%*      |
| 10  | 100 | 100%*                  | 100%*      |

Note: Numbers with superscript * mean that, these success ratios are obtained by logical inference, since systems with the same $n$, but with less equations, i.e., smaller $m$, can achieve success ratio 100%.
Table 5: Computational performance of “DAG + Reduce$_{1/2}$”.

| $m$ | $n$ | DAG + Reduce$_{1/2}$ | Average value of valid $t$ searched | $M$ |
|-----|-----|----------------------|-----------------------------------|-----|
| 2   | 40  | 100%*                | 1.000                             | $10^5$ |
| 2   | 50  | 100%*                | 1.756                             | $10^5$ |
| 2   | 60  | 100%*                | 7.184                             | $10^5$ |
| 2   | 70  | 100%*                | 66.189                            | $10^5$ |
| 2   | 80  | –%**                 | –**                               | $10^5$ |
| 2   | 90  | –%**                 | –**                               | $10^5$ |
| 2   | 100 | –%**                 | –**                               | $10^5$ |
| 3   | 60  | 100%*                | 1.000                             | $10^5$ |
| 3   | 70  | 100%*                | 1.000                             | $10^5$ |
| 3   | 80  | 100%*                | 1.111                             | $10^5$ |
| 3   | 90  | 100%*                | 3.623                             | $10^5$ |

Note: Numbers with superscript * tell that, one new equation is generated via disaggregation, and then is added to the original system. Normally, the new equation is generated based on the first equation of the original system.

Note: Numbers with superscript ** tell that, since lattice attack to all the original systems fail initially, this implies that these systems are more difficult to be solved.

Table 6: Computational performance of “DAG + CJLOSS-Alg”.

| $m$ | $n$ | DAG + CJLOSS-Alg | Average value of valid $t$ searched | $M$ |
|-----|-----|------------------|-----------------------------------|-----|
| 2   | 60  | 100%*            | 2.813                             | $10^5$ |
| 2   | 70  | 100%*            | 45.714                            | $10^5$ |
| 2   | 80  | –%**             | –**                               | $10^5$ |
| 2   | 90  | –%**             | –**                               | $10^5$ |
| 2   | 100 | –%**             | –**                               | $10^5$ |
| 3   | 90  | 100%*            | 1.500                             | $10^5$ |
| 3   | 100 | 100%*            | 16.968                            | $10^5$ |

Note: Numbers with superscript * tell that, one new equation is generated via disaggregation, and then is added to the original system. Normally, the new equation is generated based on the first equation of the original system.

Note: Numbers with superscript ** tell that, since lattice attack to all the original systems fail initially, this implies that these systems are more difficult to be solved.
Table 4 records the success ratios of Algorithm \texttt{Reduce}_{1/2} and Algorithm \texttt{CJLOSS-Alg}, respectively, i.e., the number of systems which successfully return a binary solution divided by the total number of randomly generated systems with fixed dimension. We use these two algorithms to test systems with multiple subset-sum equations, as they are with better performance among variant lattice attack algorithms reported in Table 1. Therefore, these two lattice attack algorithms are used as benchmarks. The number of unknown variables, $n$, has been tested up to 100, where the coefficients of the tested systems are already as large as $2^n = 2^{100}$.

Table 5 and Table 6 report the computational performance of integrating modular disaggregation technique with algorithms \texttt{Reduce}_{1/2} and \texttt{CJLOSS-Alg}, respectively. Specifically, in both Table 5 and Table 6: 1). Column 1 and Column 2 record the dimension of systems; 2). Column 3 with algorithm’s name records the success ratio which is the number of successful systems divided by the total number of tested systems; 3). Column 4 records the average value of valid $t$ that have been searched, only concerning initially failed systems; 4). Column 5 records the fixed value of $M$. Note that, the value of $\frac{1}{t}$ is just the value of parameter $r$ introduced in Section 3.2.

Similarly to that of Section 4.1, the codes used in this section are implemented in C++ computer language, utilizing packages in the C++ library named NTL (see [52]) which is the most cutting-edge library for doing number theory and for dealing with arbitrarily large integer numbers. For parameter setting, we set $N = 10^8$ in the matrices $B_{\text{CJLOSS}}$ and $B$ defined in Equation (9) and Equation (11), respectively, and $\alpha = 99/100$ in the LLL algorithm (refer to Algorithm 2 in this paper).

5 Statistical Analysis of Numerical Tests

Numerical tests in the previous section exhibit the efficiency of the algorithm that integrates modular disaggregation techniques with lattice attack algorithm. In this section conjectures are proposed to explain this efficiency, and we also try to partially confirm our conjectures via simulation. Recall that, the vector $\mathbf{v}$ introduced in Section 3.2 depends on the parameter $r$.

Conjecture 1. (a) If after disaggregation, set \((3)\mid r\), which are determined by parameter $r$ introduced in Section 3.2, can cut-off the first returned non-binary integer solution $\tilde{x}$ with small Euclidean length, then the probability of returning a valid binary solution $x^*$ increases.

(b) After disaggregation, whether set \((3)\mid r\) can cut-off the non-binary integer solution $\tilde{x}$ with small Euclidean length, depends on the structure of the corresponding new kernel lattice $\ker_{\mathbb{Z}}((\mathbf{a}^T, \mathbf{v}^T)^T)$. Note that, the concept of kernel lattice has been defined in item (a) of Theorem 2 which has also been studied in [2].

(c) Regarding item (b), to be even more specific, we conjecture that if the kernel lattice $\ker_{\mathbb{Z}}((\mathbf{a}^T, \mathbf{v}^T)^T)$ is sparser and more rectangular, then the corresponding set \((3)\mid r\) can cut-off the non-binary solution $\tilde{x}$ with small Euclidean length more easily.
5.1 Volume and Minimum Volume Ellipsoid of a Lattice

In order to further study and confirm the conjectures, we first explain the concept of the volume of a lattice, and the concept of the minimum volume ellipsoid of a lattice. These concepts are explained geometrically in Figure 5 and Figure 6 as well.

**Definition 3** (see Definition 1.9 in [6]). Given that columns of matrix $D$ consist a basis of lattice $L$, then the volume of lattice $L$ is defined as follows,

$$\text{vol}(L) = \sqrt{\det(D^T D)}.$$ 

Figure 5: Minimum volume ellipsoid of a basis of kernel lattice.

Next we study the minimum volume ellipsoid of a given lattice with basis $b_1, b_2, \ldots, b_m \in \mathbb{R}^n$ with $n > m$. Our idea is that first transforming the lattice to another linear space with dimension $m$, meanwhile keeping the angles of each pair of the basis vector. Let $D = (b_1 \ b_2 \ \cdots \ b_m) \in \mathbb{R}^{n \times m}$, then find matrix $U = (u_1 \ u_2 \ \cdots \ u_m) \in \mathbb{R}^{n \times m}$ and matrix $S = (s_1 \ s_2 \ \cdots \ s_m) \in \mathbb{R}^{m \times m}$ with $U^T U = I$, i.e., the columns of $D$ form the basis of the new linear space that we want to transform the lattice to be within, moreover, we require the following,

$$D = U S.$$ 

We claim that for any such basis matrix $D$, such $U$ and $S$ can always be found to satisfy that $U^T U = I$. For example, the SVD decomposition of $D$ can achieve this goal. Next we check the angles between each pair of $b_i$ and $b_j$, and each pair of $s_i$ and $s_j$. Based on Eq. (31), we have that,

$$b_i = U s_i, \ \forall \ i \in \{1, 2, \ldots, m\},$$ 

(31)
which yields that,
\[ b_i^T b_j = (Us_i)^T (Us_j) = s_i^T (U^T U) s_j = s_i^T s_j, \tag{32} \]
and
\[ ||b_i||^2 = b_i^T b_i = s_i^T s_i = ||s_i||^2, \quad \forall \ i, j \in \{1, 2, \ldots, m\}. \tag{33} \]

This directly shows that the angle between \( b_i \) and \( b_j \) is the same as the angle between \( s_i \) and \( s_j \). We only transform the lattice generated by \( D = (b_1 \ b_2 \ \cdots \ b_m) \) to another linear space, but keep the shape structure of the lattice all the same.

Now instead of studying lattice \( \mathcal{L}_D \) generated by \( D \) in a dimension \( n \) space, we study lattice \( \mathcal{L}_S \) generated by \( S \) in a dimension \( m \) space. Note that, \( \mathcal{L}_D \) and \( \mathcal{L}_S \) have the same lattice structure. Meanwhile, the minimum volume ellipsoid of lattice \( \mathcal{L}_D \) should have the same structure as that of lattice \( \mathcal{L}_S \).

We adopt the algorithm proposed in Chapter 8.4.1 of \cite{5} to calculate the minimum volume ellipsoid of lattice \( \mathcal{L}_S \), obtained by transforming \( \mathcal{L}_D \) to a lower dimension space. Here \( \mathcal{L}_D \) would be the kernel lattice of the new systems after disaggregation.

In fact, under fixed dimension, the ratio between the volume of lattice and the volume of minimum volume ellipsoid is always a constant, i.e.,
\[ \gamma := \frac{\text{volume of minimum volume ellipsoid}}{\text{volume of covered lattice}} = \frac{m^{m/2}}{2^{m-1}} \frac{1}{m} \frac{\pi^{m/2}}{\Gamma(m/2)} \tag{34} \]
is always a constant, under the same dimension \( m \) of the lattice. The ratio values are listed in Table 7 with dimensions from 2 up to 7.

**Table 7: Relation between volume of minimum volume ellipsoid and its relative covered lattice.**

| Dimension of Lattice \((m)\) | Volume of minimum volume ellipsoid \((\gamma)\) |
|------------------------------|----------------------------------|
| 2                            | 1.5708                           |
| 3                            | 2.7207                           |
| 4                            | 4.9348                           |
| 5                            | 9.1955                           |
| 6                            | 17.4410                          |
| 7                            | 33.4976                          |
5.2 Analysis on Conjecture (a)

5.2.1 Systems with single subset-sum equation

In order to test conjecture item (a), logistic regressions are conducted on the preliminary small examples recorded in Table 8, which initially fail to return binary solution under Reduce. All the possible values of ratio $\frac{t}{M}$ are enumerated, and the statistical results are recorded in Table 9. The reason of choosing Reduce in this section is that we need to enumerate all the jump points of subset-sum problems, thus we want to control the magnitude of dimension $n$, while for small $n$, it is very difficult to find initially failed problems under Reduce$_{1/2}$ and CJLOSS-Alg.

Table 8: Problems with given $a$ and $b$, dimension $n = 6$.

| Problem No. | $a$ | $b$ |
|-------------|-----|-----|
| 1 | (7, 26, 18, 43, 32, 10) | 57 |
| 2 | (24, 31, 3, 29, 17, 18) | 44 |
| 3 | (36, 21, 8, 63, 53, 52) | 97 |
| 4 | (58, 56, 5, 50, 30, 62) | 93 |
| 5 | (15, 55, 37, 11, 13, 43) | 65 |
| 6 | (8, 51, 26, 32, 21, 25) | 55 |
| 7 | (17, 52, 43, 45, 63, 40) | 123 |
| 8 | (8, 39, 47, 35, 48, 63) | 103 |
| 9 | (30, 2, 4, 47, 33, 36) | 67 |
| 10 | (14, 57, 29, 38, 60, 11) | 103 |

Table 9: Success or Failure vs. Cut or Non-cut.

| Problem No. | Coefficients (t-Value, p-Value) | Cut or Non-cut |
|-------------|---------------------------------|----------------|
|              | Constant                        |                |
| 1            | 0.4249 (1.3622, 0.1731)         | -2.4200 (-5.9737, 2.3187×10$^{-9}$) |
| 2            | 0.2364 (0.6844, 0.4937)         | -0.4959 (-1.2715, 0.2035) |
| 3            | -0.0000 (-0.0000, 1.0000)       | -2.2773 (-6.9154, 4.6646×10$^{-12}$) |
| 4            | 0.1542 (0.2771, 0.7817)         | -0.1602 (-0.2824, 0.7776) |
| 5            | 0.7577 (2.4214, 0.0155)         | -2.0070 (-5.5813, 2.3866×10$^{-8}$) |
| 6            | -1.7918 (-4.3889, 1.1393×10$^{-5}$) | 0.0000 (0.0000, 1.0000) |
| 7            | 2.3979 (3.2468, 0.0012)         | -1.7430 (-2.3331, 0.0196) |
| 8            | -0.4169 (-1.7430, 0.0813)       | -2.3844 (-6.6667, 2.6167×10$^{-11}$) |
| 9            | 0.5108 (1.7134, 0.0866)         | -2.2192 (-6.0147, 1.8025×10$^{-9}$) |
| 10           | 0.1082 (0.4648, 0.6421)         | -2.2580 (-7.1145, 1.1233×10$^{-12}$) |

Note: Table 9 records the logistic regression test result with constant, which can be compared with Table 10. We can observe that without constant, the relation between ‘Success’, ‘Failure’ (1 or 0) and ‘Non-cut’, ‘Cut’ (1 or 0) becomes even more significant.

Note that for every possible scenario of $(t, M)$ pair, ‘Success’ := 1, ‘Failure’ := 0, ‘Non-cut’ := 1,
Table 10: Success or Failure vs. Cut or Non-cut.

| Problem No. | Coefficients (t-Value, p-Value) | Coefficients (t-Value, p-Value) |
|-------------|---------------------------------|---------------------------------|
|             | Constant                        | Cut or Non-cut                  |
| 1           | -1.9951 (-7.7179, 1.1825×10^{-14}) |                                 |
| 2           | -0.2595 (-1.4328, 0.1519)       |                                 |
| 3           | -2.2773 (-10.6247, 2.2868×10^{-26}) |                                 |
| 4           | -0.0060 (-0.0548, 0.9563)       |                                 |
| 5           | -1.2493 (-7.0519, 1.7645×10^{-12}) |                                 |
| 6           | -1.7918 (-7.9556, 1.7834×10^{-15}) |                                 |
| 7           | 0.6549 (5.8202, 5.8790×10^{-9})  |                                 |
| 8           | -2.8013 (-10.5344, 5.9977×10^{-26}) |                                 |
| 9           | -1.7084 (-7.8596, 3.8540×10^{-15}) |                                 |
| 10          | -2.1498 (-9.9673, 2.1189×10^{-23}) |                                 |

Note: Table 10 records the logistic regression test result without constant, which can be compared with Table 9. We can observe that without constant, the relation between ‘Success’, ‘Failure’ (1 or 0) and ‘Non-cut’, ‘Cut’ (1 or 0) becomes even more significant, i.e., cutting-off $\tilde{x}$ strongly implies a success.

and ‘Cut’ := 0, where ‘Cut’ means cutting off the initially returned non-binary solution $\tilde{x}$, and ‘Noncut’ means not cutting off $\tilde{x}$. Based on data recorded in Table 9 and Table 10 our conjecture (a) proposed at the beginning of this section can be verified, i.e., cutting-off the initially returned non-binary short solution $\tilde{x}$ strongly correlates with a success in returning a valid binary solution.

5.2.2 Systems with multiple subset-sum equations

In this part, we further test and verify conjecture (a) for systems with dimension $(m, n)$, to check whether conjecture (a) also holds for systems with multiple subset-sum equations. The small systems used are recorded in Table 11 which initially fail to return valid binary solutions under Reduce. Similarly to the reasons in Section 5.2.1, Reduce is chosen other than Reduce_{1/2} and CJLOSS-Alg. All the jump points of subset-sum problems are enumerated, and the statistical results are recorded in Table 12 and Table 13.
Table 11: Systems with $A$ and $b$, dimension $(m, n) = (2, 6)$.

| Problem No. | $A$                  | $b$ |
|-------------|----------------------|-----|
| 1           | (63, 9, 34, 46, 2, 55) | 99  |
|             | (51, 19, 12, 44, 3, 25) | 66  |
| 2           | (52, 43, 1, 10, 9, 11) | 62  |
|             | (9, 21, 1, 37, 41, 43) | 51  |
| 3           | (8, 37, 62, 62, 17, 32) | 87  |
|             | (12, 29, 38, 51, 10, 20) | 60  |
| 4           | (13, 52, 53, 38, 2, 23) | 68  |
|             | (29, 25, 8, 42, 8, 5) | 45  |
| 5           | (37, 18, 1, 39, 37, 22) | 75  |
|             | (54, 59, 8, 43, 27, 9) | 89  |
| 6           | (12, 13, 8, 32, 50, 49) | 70  |
|             | (4, 51, 29, 37, 51, 27) | 84  |
| 7           | (4, 2, 49, 52, 7, 12) | 60  |
|             | (13, 25, 4, 34, 53, 49) | 70  |
| 8           | (27, 1, 5, 2, 35, 64) | 67  |
|             | (1, 21, 13, 8, 35, 44) | 49  |
| 9           | (31, 18, 3, 63, 61, 52) | 95  |
|             | (19, 8, 36, 58, 50, 63) | 105 |
| 10          | (30, 59, 50, 7, 2, 34) | 82  |
|             | (3, 58, 49, 42, 16, 35) | 68  |

Table 12: Success or Failure vs. Cut or Non-cut.

| Problem No. | Coefficients ($t$-Value, $p$-Value) |
|-------------|-----------------------------------|
|             | Constant                         | Cut or Non-cut                      |
| 1           | -1.9995 (-82.9485, 0)            | 1.3583 (45.1497, 0)                |
| 2           | -3.0770 (-61.0656, 0)            | 3.0673 (55.7344, 0)                |
| 3           | -2.6314 (-60.3586, 0)            | 3.2163 (70.5206, 0)                |
| 4           | -2.4893 (-64.8510, 0)            | 3.1334 (71.4363, 0)                |
| 5           | -3.6363 (-59.0880, 0)            | 4.7840 (74.8191, 0)                |
| 6           | -2.4159 (-77.1892, 0)            | 2.7534 (78.7439, 0)                |
| 7           | -2.1992 (-69.1417, 0)            | 1.4579 (37.8164, 6.1188x10^-313)   |
| 8           | -2.7930 (-55.0320, 0)            | 3.2765 (58.6059, 0)                |
| 9           | -2.3780 (-103.6493, 0)           | 2.9493 (112.1045, 0)               |
| 10          | -3.9944 (-69.1250, 0)            | 4.1811 (70.0201, 0)                |

Note: Table 12 records the logistic regression test result with constant, which can be used together with Table 13. We can observe that the relation between ‘Success’, ‘Failure’ (1 or 0) and ‘Non-cut’, ‘Cut’ (1 or 0) is significant, i.e., cutting-off $\tilde{x}$ strongly correlates with a success.

Based on the data recorded in Table 12 and Table 13 our conjecture (a) proposed at the
Table 13: Success or Failure vs. Cut or Non-cut.

| Problem No. | Coefficients (t-Value, p-Value) |
|-------------|---------------------------------|
| Constant    | Cut or Non-cut                  |
| 1           | -0.6412 (-35.6229, 6.1922×10^{-278}) |
| 2           | -0.0098 (-0.4425, 0.6581)       |
| 3           | 0.5849 (43.6602, 0)             |
| 4           | 0.6441 (30.3447, 2.9511×10^{-202}) |
| 5           | 1.1477 (66.1315, 0)             |
| 6           | 0.3375 (21.6483, 6.3047×10^{-104}) |
| 7           | -0.7413 (-34.0278, 8.6339×10^{-254}) |
| 8           | 0.4835 (20.6197, 1.8257×10^{-94}) |
| 9           | 0.5713 (44.3725, 0)             |
| 10          | 0.1867 (12.4064, 2.4124×10^{-35}) |

Note: Table 13 records the logistic regression test result without constant, which can be used together with Table 12. We can observe that the relation between ‘Success’, ‘Failure’ (1 or 0) and ‘Non-cut’, ‘Cut’ (1 or 0) is significant, i.e., cutting-off \( \tilde{x} \) strongly correlates with a success.

The beginning of this section can be verified, i.e., cutting-off the initially returned non-binary short solution \( \tilde{x} \) strongly correlates with a success in returning a valid binary solution. Note that, for every possible scenario of \((t, M)\) pair, ‘Success’ := 1, ‘Failure’ := 0, ‘Non-cut’ := 1, and ‘Cut’ := 0, where ‘Cut’ means cutting off the initially returned non-binary solution \( \tilde{x} \), and ‘Non-cut’ means not cutting off \( \tilde{x} \). Interesting phenomenon is that, when we compare Table 9 and Table 12, the signs of coefficients for variable ‘Cut’ or ‘Non-cut’ opposite to each other.

Below, a concrete example is used to illustrate.

Example 3. Consider the following problem,

\[
A = \begin{pmatrix} 63, 9, 34, 46, 2, 55 \\ 51, 19, 12, 44, 3, 25 \end{pmatrix}, \quad and \quad b = \begin{pmatrix} 99 \\ 66 \end{pmatrix},
\]

which helps to illustrate how some \( r = \frac{t}{M} \) and the corresponding set \( \{3| r \} \) enable the new system to return a valid binary solution.

Solution. Initially, the system returns a short non-binary solution \((0,0,0,1,-1,1)\) under Algorithm Reduce.

Let \( r_1 = \frac{t_1}{M_1} = \frac{1}{63} \), and apply \( r_1 \) to the first equation, thus to obtain a new equation as follows,

\[
v_{11}x_1 + v_{12}x_2 + v_{13}x_3 + v_{14}x_4 + v_{15}x_5 + v_{16}x_6 + \tilde{k}_1 = w_1,
\]

with

\[
v_{1i} := \left[ \frac{t_1}{M_1} A_{ii} \right], \quad \text{for} \quad i = 1, 2, \ldots, n, \quad \text{and} \quad w_1 := \left[ \frac{t_1}{M_1} b_1 \right],
\]

40
and the upper bound of $\tilde{k}_1$ is $u(\tilde{k}_1) = 1$.

Let $r_2 = \frac{t_2}{M_2} = \frac{22}{51}$, and apply $r_2$ to the second equation, thus to obtain a new equation as follows,

$$v_{21}x_1 + v_{22}x_2 + v_{23}x_3 + v_{24}x_4 + v_{25}x_5 + v_{26}x_6 + \tilde{k}_2 = w_2,$$

with

$$v_{2i} := \left\lfloor \frac{t_2}{M_2} A_{2i} \right\rfloor,$$

for $i = 1, 2, \ldots, n$, and $w_2 := \left\lfloor \frac{t_2}{M_2} b_2 \right\rfloor$.

and the upper bound of $\tilde{k}_2$ is $u(\tilde{k}_2) = 1$.

The new system becomes,

$$\begin{align*}
63x_1 + 9x_2 + 34x_3 + 46x_4 + 2x_5 + 55x_6 &= 99 \\
51x_1 + 19x_2 + 12x_3 + 44x_4 + 3x_5 + 25x_6 &= 66 \\
x_1 + 0x_2 + 0x_3 + 0x_4 + 0x_5 + 0x_6 + \tilde{k}_1 &= 1 \\
22x_1 + 8x_2 + 5x_3 + 18x_4 + 1x_5 + 10x_6 + \tilde{k}_2 &= 28
\end{align*}$$

with $(x, \tilde{k}_1, \tilde{k}_2) = (x_1, x_2, x_3, x_4, x_5, x_6, \tilde{k}_1, \tilde{k}_2)^T \in \{0, 1\}^8$, which returns the following solution,

$$(x, \tilde{k}_1, \tilde{k}_2) = (1, 0, 1, 0, 1, 0, 0, 0)^T,$$

under Reduce (see Algorithm 3 in this paper). Truncating the first $n = 6$ elements yields,

$$x = (1, 0, 1, 0, 1, 0)^T,$$

which is a valid binary solution to the original system $Ax = b$.

Specifically, next we enumerate all the jump points of subset-sum problems to generate the new system, thus to test and verify the reasons of why some $r$ can succeed in helping return a binary solution. For Example 3, there are total $(a_{11} + a_{12} + \cdots + a_{1n} - n) \times (a_{21} + a_{22} + \cdots + a_{2n} - n) = 30, 044$ possible scenarios for $(\frac{t_1}{M_1}, \frac{t_2}{M_2})$ in Example 3.

5.3 Analysis on Conjecture (b) and Conjecture (c)

In this part, we further verify our conjectures (b) and (c) proposed at the beginning of this section. For each system in Section 5.2.2, we first generate the data of related variables, then conduct statistical regression to test the significance of these variables. In the following, we use Example 3 again to illustrate.

**Example 4.** Consider the following problem,

$$A = \begin{pmatrix} 63, 9, 34, 46, 2, 55 \\ 51, 19, 12, 44, 3, 25 \end{pmatrix}, \quad \text{and} \quad b = \begin{pmatrix} 99 \\ 66 \end{pmatrix},$$

which helps to illustrate how some $r = \frac{t}{M}$ can enable the new systems to return binary solution.
Solution. a) Let \( r_1 = \frac{t_1}{M_1} = \frac{1}{63} \) and \( r_2 = \frac{t_2}{M_2} = \frac{22}{51} \), the new system in Eq. (39) is as follows,

\[
\begin{align*}
63x_1 + 9x_2 + 34x_3 + 46x_4 + 2x_5 + 55x_6 &= 99 \\
51x_1 + 19x_2 + 12x_3 + 44x_4 + 3x_5 + 25x_6 &= 66 \\
x_1 + 0x_2 + 0x_3 + 0x_4 + 0x_5 + 0x_6 + k_1 &= 1 \\
22x_1 + 8x_2 + 5x_3 + 18x_4 + 1x_5 + 10x_6 + k_2 &= 28
\end{align*}
\]

which returns binary solution to the original system. The reduced kernel basis of the new system consists of columns of \( D \),

\[
D = \begin{pmatrix}
-1 & 1 & 0 & -5 \\
0 & -1 & -9 & 5 \\
-1 & 4 & -3 & 5 \\
1 & 1 & 5 & 2 \\
-2 & -8 & 4 & 4 \\
1 & -4 & -1 & 0 \\
1 & -1 & 0 & 5 \\
1 & -4 & 3 & 5
\end{pmatrix}
\]

with volume of the kernel lattice \( \sqrt{\det(D^T D)} = 4112 \). The minimum volume ellipsoid of this kernel lattice is calculated with semi-axes,

\[
\left(13.4214, 12.6793, 7.8505, 3.0782\right),
\]

and with volume = 20,294.

b) Here we consider another scenario to compare. Let \( r_1 = \frac{t_1}{M_1} = \frac{3}{63} \) and \( r_2 = \frac{t_2}{M_2} = \frac{36}{51} \), the new system becomes,

\[
\begin{align*}
63x_1 + 9x_2 + 34x_3 + 46x_4 + 2x_5 + 55x_6 &= 99 \\
51x_1 + 19x_2 + 12x_3 + 44x_4 + 3x_5 + 25x_6 &= 66 \\
x_1 + 0x_2 + x_3 + 2x_4 + 0x_5 + 2x_6 + k_1 &= 4 \\
36x_1 + 13x_2 + 8x_3 + 31x_4 + 2x_5 + 17x_6 + k_2 &= 46
\end{align*}
\]

which returns a non-binary solution \((0, 0, 0, 1, -1, 1, 0, 0)\). The truncated solution \((0, 0, 0, 1, -1, 1)\) to the original system is non-binary as well. The reduced kernel basis of the new system consists of columns of \( D \),

\[
D = \begin{pmatrix}
-1 & -2 & -6 & -6 \\
0 & 1 & -4 & 5 \\
-1 & -5 & 1 & 4 \\
1 & 0 & 8 & 3 \\
-2 & 6 & 6 & 2 \\
1 & 5 & 0 & 1 \\
0 & 1 & 1 & 6 \\
0 & 2 & 0 & 5
\end{pmatrix}
\]
with volume of the kernel lattice $\sqrt{\det(D^T D)} = 3621$. The minimum volume ellipsoid of this kernel lattice is calculated with semi-axes,

$$(15.2642, 9.6705, 8.6903, 2.8230),$$

and with volume $= 17,870$.

c) Here we choose the other scenario to compare further. Let $r_1 = \frac{4}{63}$ and $r_2 = \frac{49}{43}$, the new system becomes,

$$\begin{align*}
63x_1 + 9x_2 + 34x_3 + 46x_4 + 2x_5 + 55x_6 &= 99 \\
51x_1 + 19x_2 + 12x_3 + 44x_4 + 3x_5 + 25x_6 &= 66 \\
3x_1 + 0x_2 + x_3 + 2x_4 + 0x_5 + 2x_6 + \tilde{k}_1 &= 4 \\
49x_1 + 18x_2 + 11x_3 + 42x_4 + 2x_5 + 24x_6 + \tilde{k}_2 &= 63
\end{align*}$$

which returns a binary solution $(1, 0, 1, 0, 1, 0, 1, 0)$. The reduced kernel basis of the new system consists of columns of $D$,

$$D = \begin{pmatrix}
1 & -3 & -5 & -7 \\
0 & 1 & -5 & 5 \\
1 & -6 & 5 & 3 \\
-1 & 1 & 9 & 4 \\
2 & 4 & -2 & 0 \\
-1 & 6 & -4 & 2 \\
0 & 1 & 0 & 6 \\
2 & 1 & 2 & 4
\end{pmatrix}$$

with volume of the kernel lattice $\sqrt{\det(D^T D)} = 4493$. The minimum volume ellipsoid of this kernel lattice is calculated with semi-axes,

$$(15.1941, 12.4433, 7.1633, 3.3181),$$

and with volume $= 22,176$.

Based on simulation data of the above example, we observe that normally, for scenarios with the same dimension of kernel lattices of new systems: 1) the larger the volume of the kernel lattice, the easier the algorithm to succeed; 2) the smaller the ratio between the maximum axis and minimum axis which implies that the ellipsoid is more like a spheroid, the easier the algorithm to succeed.

To conduct further analysis, for a single system, we enumerate all its scenarios of disaggregation and list the following data variables thus to test the relations among them,

- Dimension of kernel lattice of new system;
- Volume of kernel lattice of new system;
- New system cuts or non-cuts the original non-binary solution;
• New system succeeds or fails in returning binary solution;
• Minimum volume ellipsoid of kernel lattice of new system.

Logistic regressions are performed in MATLAB, and simulation data are recorded in Tables in this Section.

Table 14: Success/Failure vs. Volume of Kernel Lattice.

| Problem No. v.s. Dimension of Kernel | Coefficients (t-Value, p-Value) | Volume of Kernel×10^{-3} |
|-------------------------------------|---------------------------------|--------------------------|
| Prob. 1 with kernel dimension 2     | -1.5748 (-2.1524, 0.0314)       | 12.4943 (2.7779, 0.0055) |
| Prob. 1 with kernel dimension 2     | -6.0372 (4.0360, 5.4368×10^{-5}) |                           |
| Prob. 1 with kernel dimension 3     | -1.5698 (-8.6879, 3.6904×10^{-18}) | 1.5518 (12.7746, 2.2727×10^{-37}) |
| Prob. 1 with kernel dimension 3     | -0.5935 (14.3314, 1.3921×10^{-46}) |                           |
| Prob. 1 with kernel dimension 4     | -1.9937 (-15.1330, 9.8117×10^{-52}) | 0.3988 (12.1653, 4.7583×10^{-46}) |
| Prob. 1 with kernel dimension 4     | -5.3278 (-22.2111, 2.6799×10^{-109}) | 0.4945 (17.0606, 2.9148×10^{-65}) |
| Prob. 1 with kernel dimension 5     | -9.0442 (-7.2874, 3.1605×10^{-13}) | 0.3724 (4.7755, 1.7926×10^{-6}) |
| Prob. 1 with kernel dimension 6     | -0.1998 (-58.9966, 0)           |                           |

Data in Table 14 confirms our conjecture that, volume of kernel lattice after disaggregation has significant relation with whether successfully returning a binary solution after disaggregation. Statistically, data in Table 14 also confirms our conjecture that the larger the volume of kernel lattice after disaggregation, the easier the algorithm to succeed, since the coefficient in front of volume feature is positive.

Next, we want to test our conjecture on the rectangularity feature of kernel lattices. Multiple ways are designed to capture the rectangularity feature of lattice. We define the ratio between maximum semi-axis and minimum semi-axis of the minimum volume ellipsoid of a given lattice,

\[
\tilde{\lambda} := \frac{\text{Max semi-axis after normalization}}{\text{Min semi-axis after normalization}}
\]  

where the minimum volume ellipsoid is of the lattice after normalization, that is, the lengths of basis vectors of a lattice are all first normalized to be 1, only the directions of basis vectors are preserved, respectively. Regression data are recorded in Table 15.

Based on data in Table 15, we could see that the rectangularity feature is significantly related with whether succeed or fail. Since a rectangle is with \(\tilde{\lambda}\) equal 1, and a rhomboid is with \(\tilde{\lambda}\) greater than 1. The positive coefficient in front of \(\tilde{\lambda}\) tells that, the less rectangular the kernel lattice, the easier for the algorithm to succeed and return a binary solution.

Next we define a distance quantity to capture the rectangularity structure of kernel lattices,

\[
\text{minimize} \quad d := ||D^TD - \text{diag}(\lambda)||_2 \quad (41)
\]

subject to \(\lambda \geq 0\),
Table 15: Success/Failure vs. $\tilde{\lambda}$ in Equation (40).

| Problem No. v.s. Dimension of Kernel | Coefficients ($t$-Value, $p$-Value) | $\lambda = \frac{\text{Max semi-axis after normalization}}{\text{Min semi-axis after normalization}}$ |
|--------------------------------------|-------------------------------------|---------------------------------|
| Prob. 1 with kernel dimension 2      | -39.9368 (-2.6665, 0.0077)          | 0.18203 (4.8150, 1.4710 × 10^{-6}) |
| Prob. 1 with kernel dimension 2      |                                    | 7.6354 (15.6488, 3.3857 × 10^{-55}) |
| Prob. 1 with kernel dimension 3      | -1.2803 (4.8150, 1.4710 × 10^{-65}) | 6.3121 (14.2342, 5.0242 × 10^{-66}) |
| Prob. 1 with kernel dimension 3      |                                    | 1.9946 (17.5799, 3.3857 × 10^{-69}) |
| Prob. 1 with kernel dimension 4      |                                    | -0.2153 (-14.7667, 2.4031 × 10^{-49}) |
| Prob. 1 with kernel dimension 5      |                                    | -5.5635 (-27.8055, 3.7251 × 10^{-170}) |
| Prob. 1 with kernel dimension 5      |                                    | 2.4023 (21.7052, 1.8309 × 10^{-104}) |
| Prob. 1 with kernel dimension 6      |                                    | -0.6998 (-59.2329, 0) |
| Prob. 1 with kernel dimension 6      |                                    | -5.6999 (-13.3403, 1.3496 × 10^{-40}) |
| Prob. 1 with kernel dimension 6      |                                    | 1.4387 (6.1019, 1.0484 × 10^{-9}) |

where columns of matrix $D$ form the basis of kernel lattice, $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_s)^T \in \mathbb{R}_+^s$ with $s \in \mathbb{Z}_+$ be the dimension of kernel lattice. In fact, if columns of $D$ are orthogonal to each other, then the optimized objective value in Eq. (41) is 0, otherwise, the optimized objective value in Eq. (41) is greater than 0. The results of statistical test are recorded in Table 16.

Table 16: Success/Failure vs. Rectangularity of Kernel Lattice.

| Problem No. v.s. Dimension of Kernel | Coefficients ($t$-Value, $p$-Value) | $d$: Rectangularity Distance |
|--------------------------------------|-------------------------------------|------------------------------|
| Prob. 1 with kernel dimension 2      | -1.6194 (-2.2612, 0.0237)          | 0.3748 (1.8171, 1.8171) |
| Prob. 1 with kernel dimension 2      |                                    | 0.1434 (2.3768, 0.0175) |
| Prob. 1 with kernel dimension 3      | -1.0665 (-7.4854, 7.1341 × 10^{-14}) | 0.0305 (12.6218, 1.6008 × 10^{-36}) |
| Prob. 1 with kernel dimension 4      | -2.0257 (-23.2839, 6.4508 × 10^{-120}) | 0.0238 (19.4280, 4.4719 × 10^{-84}) |
| Prob. 1 with kernel dimension 5      | -2.5764 (-33.7724, 5.0216 × 10^{-250}) | 0.0205 (18.0961, 3.4221 × 10^{-73}) |
| Prob. 1 with kernel dimension 5      |                                    | -0.0182 (-54.7069, 0) |
| Prob. 1 with kernel dimension 6      | -4.7381 (-25.7345, 4.8013 × 10^{-146}) | 0.0280 (9.5833, 9.4002 × 10^{-22}) |
| Prob. 1 with kernel dimension 6      |                                    | -0.0581 (-58.2613, 0) |

A variation of the rectangularity distance $d$ defined in Eq. (41) is defined as follows,

$$
\begin{align*}
\text{minimize} & \quad \tilde{d} := \|\tilde{D}^T\tilde{D} - \text{diag}(\lambda)\|_2 \\
\text{subject to} & \quad \lambda \geq 0,
\end{align*}
$$

where columns of $\tilde{D}$ are normalized columns of $D$, all with length 1. The results of statistical test are recorded in Table 17.

Data in Table 16 and Table 17 reveal that $p$ values are small enough to guarantee that the rectangularity feature is significantly related with success (failure) output. As the coefficients for
Table 17: Success/Failure vs. Rectangularity of Kernel Lattice.

| Problem No. vs. Dimension of Kernel | Coefficients (t-Value, p-Value) | Constant | $d$: Normalized Rectangularity Distance |
|-------------------------------------|---------------------------------|----------|---------------------------------------|
| Prob. 1 with kernel dimension 2     | -0.7923 (-1.4310, 0.1524)       |          | 40.8038 (2.7501, 0.0060)              |
| Prob. 1 with kernel dimension 2     | -2.0401 (-13.4564, 2.8252×10^{-41}) |          | 10.7147 (16.6736, 2.0399×10^{-62})   |
| Prob. 1 with kernel dimension 3     | -2.2619 (-21.0078, 5.5669×10^{-98}) |          | 3.8588 (17.7086, 3.5996×10^{-70})    |
| Prob. 1 with kernel dimension 4     | -3.6053 (-31.3312, 1.7553×10^{-215}) |          | 4.6962 (20.9265, 3.0710×10^{-97})    |
| Prob. 1 with kernel dimension 5     | -5.5366 (-19.0951, 2.7707×10^{-81}) |          | 4.9645 (8.6713, 4.2727×10^{-18})     |

rectangularity feature are positive, and success is of a higher value than failure (1 denotes success, 0 denotes failure), thus statistically, the less rectangularity of kernel lattice, the easier the algorithm to succeed.

In summary, we use Problem 1 recorded in Table 11 to further test the significance of some features that we conjecture can predict success or failure of returning binary solution statistically. We filter and test kernel lattices with same dimension after disaggregation, respectively. After disaggregation, there are total 72 kernel lattices with dimension 2, total 1206 kernel lattices with dimension 3, total 6574 kernel lattices with dimension 4, total 13288 kernel lattices with dimension 5, and total 8904 kernel lattices with dimension 6. Generally, the larger the volume of kernel lattice after disaggregation, the easier the algorithm to succeed. Meanwhile, the less rectangularity of kernel lattice after disaggregation, the easier the algorithm to succeed and return a binary solution.

6 Conclusion and Further Study

Generally speaking, the dimensions of solution space of subset-sum problems or systems of equations depend on the magnitudes of $m$ and $n$. When restrict the general solution space to set of binary solutions, from the literature, we know that subset-sum problems with density close to 1, and systems of equations with half-half split are the most difficult. When increase the value of $m$ via disaggregation, more information will be revealed and the dimension of solution space will be lowered. Moreover, by utilizing disaggregation techniques to add more equations, it is possible to cut off some invalid short solutions, thus to increase the probability of returning valid binary solutions. However, there do exist a balance between introducing new equation via disaggregation techniques, and including new variables after disaggregation. Here, the quantities of “jump points” play a crucial role.

Based on our numerical simulations, if the success ratio of original systems with fixed dimension is non-zero, then we always can utilize disaggregation technique to increase the success ratio to 100%.
Otherwise, it is difficult to utilize disaggregation technique to increase the success ratio, for example, for problems with dimension $m = 1$ and $n = 50$ and above, in Table 1 and for problems with dimension $m = 2$ and $n = 80$ and above, in Table 3.

For further study, we observe that for some fixed dimensions, original systems all fail to return valid binary solutions. For such problem sets, effective algorithms are in urgent need to be further designed thus to return valid binary solutions.

We also would like to examine the effects of adding more than one equation in disaggregation procedure. Intuitively, adding more equations (if disaggregation is successful) will even further lower the dimension of solution space and reveal even more information of the problem, and thus to increase the chance of returning binary solution after lattice transformation.

References

[1] Aardal, K., Hurkens, C. A. J., and Lenstra, A. K. Solving a System of Linear Diophantine Equations with Lower and Upper Bounds On the Variables. *Mathematics of Operations Research* 25, 3 (2000), 427–442.

[2] Aardal, K., and von Heymann, F. On the structure of reduced kernel lattice bases. *Mathematics of Operations Research* 39, 3 (2013), 823–840.

[3] Aardal, K., and Wolsey, L. A. Lattice Based Extended Formulations for Integer Linear Equality Systems. *Mathematical Programming* 121, 2 (2010), 337–352.

[4] Aono, Y., Wang, Y., Hayashi, T., and Takagi, T. Improved progressive bkz algorithms and their precise cost estimation by sharp simulator. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques* (2016), Springer, pp. 789–819.

[5] Boyd, S., and Vandenberghe, L. *Convex Optimization*. Cambridge University Press, March 2004.

[6] Bremner, M. R. *Lattice Basis Reduction: An Introduction to the LLL Algorithm and Its Applications*. Boca Raton, FL: CRC Press, 2011.

[7] Brickell, E. F. Solving Low Density Knapsacks. In *CRYPTO* (1983), pp. 25–37.

[8] Buhler, J., and Koblitz, N. Lattice basis reduction, jacobi sums and hyperelliptic cryptosystems. *Bulletin of the Australian Mathematical Society* 58, 1 (1998), 147–154.

[9] Chen, Y., and Nguyen, P. Q. Bkz 2.0: Better lattice security estimates. In *International Conference on the Theory and Application of Cryptology and Information Security* (2011), Springer, pp. 1–20.

[10] Chor, B., and Rivest, R. L. A Knapsack Type Public Key Cryptosystem Based on Arithmetic in Finite Fields. *Proc. CRYPTO’ 84* (1984).
[11] Coster, M. J., Joux, A., LaMacchia, B. A., Odlyzko, A. M., Schnorr, C.-P., and Stern, J. An Improved Low-density Subset Sum Algorithm. In Advances in Cryptology: Proceedings of Eurocrypt '91 (1991).

[12] Coster, M. J., Joux, A., LaMacchia, B. A., Odlyzko, A. M., Schnorr, C.-P., and Stern, J. Improved Low-density Subset Sum Algorithms. Computational Complexity 2, 2 (1992), 111–128.

[13] Divasón, J., Joosten, S., Thiemann, R., and Yamada, A. A Formalization of the LLL Basis Reduction Algorithm. In International Conference on Interactive Theorem Proving (2018), Springer, pp. 160–177.

[14] Feng, Y., Wang, G.-G., Deb, S., Lu, M., and Zhao, X.-J. Solving 0-1 Knapsack Problem by a Novel Binary Monarch Butterfly Optimization. Neuro Computing and Applications 28 (2017), 1619–1634.

[15] Filaseta, M. On the factorization of polynomials with small Euclidean norm. Number theory in progress 1 (1999), 143–163.

[16] Frieze, A. M. On the Lagarias-Odlyzko Algorithm for the Subset Sum Problem. SIAM Journal on Computing 15 (1986), 536–539.

[17] Garey, M. R., and Johnson, D. S. Computers and Intractability: A Guide to the Theory of NP-completeness. 1979.

[18] Glover, F., and Woolsey, R. E. Aggregating Diophantine Equations. Zeitschrift für Operations Research 16 (1972), 1–10.

[19] Havas, G., Majewski, B. S., and Matthews, K. R. Extended GCD and Hermite Normal Form Algorithms via Lattice Basis Reduction. Experimental Mathematics 7 (1998), 125–136.

[20] Hayes, B. Computing Science: The Easiest Hard Problem. American Scientist 90, 2 (2002), 113–117.

[21] Impagliazzo, R., and Naor, M. Efficient Cryptographic Schemes Provably as Secure as Subset Sum. Journal of Cryptology 9, 4 (1996), 199–216.

[22] Jen, S.-M., Lai, T.-L., Lu, C.-Y., and Yang, J.-F. Knapsack Cryptosystems and Unreliable Reliance on Density. In AINA (2012), L. Barolli, T. Enokido, F. Xhafa, and M. Takizawa, Eds., IEEE, pp. 748–754.

[23] Jen, S.-M., Lu, C.-Y., Lai, T.-L., and Yang, J.-F. Empirical Exploration of Lattice Attacks for Building Secure Knapsack Cryptosystems. 2012 International Conference on Anti-Counterfeiting, Security and Identification, IEEE, pp. 1–5.

[24] Kate, A., and Goldberg, I. Generalizing Cryptosystems Based on the Subset Sum Problem. International Journal of Information Security 10, 3 (2011), 189–199.
[25] Kellerer, H., Pferschy, U., and Pisinger, D., Eds. *Knapsack Problems*. Springer, 2004.

[26] Kunihiro, N. New Definition of Density on Knapsack Cryptosystems. In *International Conference on Cryptology in Africa, 2008* (2008), pp. 156–173.

[27] Lagarias, J. C., and Odlyzko, A. M. Solving Low-density Subset Sum Problems. *Journal of the ACM (JACM)* 32, 1 (1985), 229–246.

[28] Lenstra, A. K., Lenstra, H. W., and Jr., L. L. Factoring Polynomials with Rational Coefficients. *Mathematische Annalen* 261 (1982), 515–534.

[29] Li, D., Sun, X., Gao, J., Gu, S., and Zheng, X. Reachability Determination in Acyclic Petri Nets by Cell Enumeration Approach. *Automatica* 47 (2011), 2094–2098.

[30] Lovász, L., and Scarf, H. E. The generalized basis reduction algorithm. *Mathematics of Operations Research* 17 (1992), 751–764.

[31] Lu, B. *Linear Diophantine Equations: Integration of Disaggregation with LLL Algorithm*. PhD thesis, The Chinese University of Hong Kong, 2014.

[32] Mamedov, K. S., and Mardanov, S. S. Reducing an Integer Linear Equation to an Equivalent System. *Cybernetics and Systems Analysis* 42 (2006), 159–162.

[33] Mardanov, S. S., and Mamedov, K. S. Disaggregation of Diophantine Equation with Boolean Variables. *Izv. AN Azerbaidzhan. Ser. Fiz.-Tech. i Mat. Nauk* 2-3 (2000), 50–54.

[34] Mardanov, S. S., and Mamedov, K. S. Disaggregation of Diophantine Equation with Boolean Variables. *Computational Optimization and Application* 27 (2004), 31–36.

[35] Martinet, J. *Perfect lattices in Euclidean spaces*, vol. 327. Springer Science & Business Media, 2013.

[36] May, A. Using lll-reduction for solving rsa and factorization problems. In *The LLL algorithm*. Springer, 2009, pp. 315–348.

[37] Merkle, R. C., and Hellman, M. E. Hiding Information and Signatures in Trapdoor Knapsacks. *IEEE Transactions on Information Theory* IT-24 (1978), 525–530.

[38] Micciancio, D. Lattice-based cryptography. *Encyclopedia of Cryptography and Security* (2011), 713–715.

[39] Nguyen, P. Q., and Stehlé, D. An lll algorithm with quadratic complexity. *SIAM Journal on Computing* 39, 3 (2009), 874–903.

[40] Nguyen, P. Q., and Stern, J. Adapting Density Attacks to Low-weight Knapsacks. *Advances in Cryptology – ASIACRYPT 2005* (2005), 41–58.

[41] Nguyen, P. Q., and Vallée, B., Eds. *The LLL Algorithm: Survey and Applications*. Information Security and Cryptography. Springer-Verlag Berlin Heidelberg, 2010.
[42] Odlyzko, A. M. The Rise and Fall of Knapsack Cryptosystems. *Cryptology and Computational Number Theory* 42 (1990), 75–88.

[43] Okamoto, T., Tanaka, K., and Uchiyama, S. Quantum Public-Key Cryptosystems. *Advances in Cryptology – CRYPTO 2000* (2000), 147–165.

[44] Pisinger, D. Where are the Hard Knapsack Problems? *Computer and Operations Research* 32 (2005), 2271–2284.

[45] Pisinger, D., and Saidi, A. Tolerance Analysis for 0-1 Knapsack Problems. *European Journal of Operational Research* 258 (2017), 866–876.

[46] Schneider, M. Probabilistic analysis of lll reduced bases. In *Dagstuhl Seminar Proceedings* (2009), Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

[47] Schneider, M., and Buchmann, J. Extended lattice reduction experiments using the bkz algorithm. *Sicherheit 2010. Sicherheit, Schutz und Zuverlässigkeit* (2010).

[48] Schnorr, C. P. Lattice reduction by random sampling and birthday methods. In *Annual Symposium on Theoretical Aspects of Computer Science* (2003), Springer, pp. 145–156.

[49] Schnorr, C.-P., and Euchner, M. Lattice Basis Reduction: Improved Practical Algorithms and Solving Subset Sum Problems. *Mathematical Programming* 66 (1994), 181–191.

[50] Schnorr, C.-P., and Shevchenko, T. Solving Subset Sum Problems of Densioty Close to 1 by “randomized” BKZ-reduction. *IACR Cryptology ePrint Archive* 2012 (2012), 620.

[51] Sharma, S., Sharma, P., and Dhakar, R. S. RSA Algorithm Using Modified Subset Sum Cryptosystem. In *2011 2nd International Conference on Computer and Communication Technology (ICCCT-2011)* (2011), IEEE, pp. 457–461.

[52] Shoup, V. NTL: A Library for doing Number Theory, http://www.shoup.net/ntl/. *New York University*.

[53] Wikipedia. Rounding, https://en.wikipedia.org/wiki/rounding.

[54] Woeginger, G. J. Exact Algorithms for NP-hard Problems: A Survey. In *Combinatorial Optimization*. Springer, 2003, pp. 185–207.