Algorithms for the Minimal Rational Fraction Representation of Sequences Revisited

Jun Che, Chengliang Tian, Yupeng Jiang, and Guangwu Xu, *Senior Member, IEEE*

***Abstract*— Given a binary sequence with length n, determining its minimal rational fraction representation (MRFR) has important applications in the design and cryptanalysis of stream ciphers. There are many studies of this problem since Klapper and Goresky first introduced an adaptive rational approximation algorithm with a time complexity of $O(n^2 \log n \log\log n)$. In this paper, we revisit this problem by considering both adaptive and non-adaptive efficient algorithms. Compared with the state-of-art methods, we make several contributions to the problem of finding MRFR. Firstly, we find a general and precise recursive relationship between the minimal bases for two adjacent lattices generated by successive truncation sequences. This enables us to improve the currently fastest adaptive algorithm proposed by Li *et al.*. Secondly, by optimizing a time-consuming step of the well-known Lagrange reduction algorithm for 2-dimensional lattices, we obtain a non-adaptive, and yet practically faster MRFR-solving algorithm named *global* Euclidean algorithm. Thirdly, we identify theoretical flaws on some non-adaptive methods in the literature by counter-examples and correct the problems by designing modified Euclidean algorithm named *partial* Euclidean algorithm. Meanwhile, we further reduce the time complexity of existing algorithm from $O(n^2)$ to $O(n \log^2 n \log\log n)$ by invoking the half-gcd algorithm. We also conduct a comprehensive experimental comparative analysis on the above algorithms to validate our theoretical analysis.**

***Index Terms*— Sequence, rational fraction representation, extended euclidean algorithm, lattice, minimal basis.**

I. Introduction

GIVEN a binary sequence $\underline{a}(n) = (a_0\ a_1\ \cdots\ a_{n-1})$ with length n, define $S_n = \sum_{i=0}^{n-1} a_i 2^i$. A rational $\frac{p}{q}$ is called a rational fraction representation (RFR) of $\underline{a}(n)$ if and only if, (1) $q > 0$ is odd, (2) p and q are relatively prime, *i.e.*, $\gcd(p, q) = 1$, and (3) $p \equiv q \cdot S_n \mod 2^n$. Particularly, let

$$\mathcal{S} = \left\{ \frac{p}{q} : \frac{p}{q} \text{ is a RFR of } \underline{a}(n) \right\},$$

and $\Phi(p, q) = \max\{|p|, |q|\}$. If $\frac{p_0}{q_0} \in \mathcal{S}$ and $\Phi(p_0, q_0) = \min_{p/q \in \mathcal{S}} \Phi(p, q)$, then $\frac{p_0}{q_0}$ is defined as the minimal rational fraction representation (MRFR) of $\underline{a}(n)$.

Determining the MRFR of a given sequence has important applications in the design and analysis of stream ciphers. The study of this problem was initialized by Klapper and Goresky [4]–[6]. In their papers, they designed a new device called feedback with carry shift register (FCSR) to generate pseudo-random sequences and investigated the properties of FCSRs. Concretely, an FCSR is uniquely determined by its connection integer. A sequence $\underline{a}(n)$ can be generated by an FCSR with connection integer q if and only if, there exists an integer p such that $\frac{p}{q}$ is a RFR of $\underline{a}(n)$. Conversely, if $\frac{p}{q}$ is a RFR of a sequence $\underline{a}(n)$, then $\log_2 \Phi(p, q)$ captures the number of the registers of the FCSR with connection integer q that can generate this sequence. Particularly, if $\frac{p_0}{q_0}$ is the MRFR of $\underline{a}(n)$, then $\log_2 \Phi(p_0, q_0)$ is called the 2-adic complexity of the sequence $\underline{a}(n)$, which determines the minimal number of registers of the FCSR generating this sequence. Therefore, finding the MRFR of a given sequence is of great importance in the study of the security of pseudo-random sequences for cryptographic applications.

A. Related Work

In Crypto'95, Klapper and Goresky [5] proposed the first algorithm (called as rational approximation (RA) algorithm) for this problem. Their algorithm is based on the rational approximation theory of De Weger [3] and Mahler [8] and can be regarded as analogous to Berlekamp and Massey algorithm [9] for linear feedback shift registers (LFSRs). One strength of their algorithm is that it is adaptive, which means that the MRFR of the $(k+1)$-truncation sequence $\underline{a}(k+1)$ is derived from the MRFR of the k-truncation sequence $\underline{a}(k)$. However, it is with a time complexity of $O(n^2 \log n \log\log n)$ and thus is time-consuming for longer and widely used cryptographic sequences. Recently, Li *et al.* [7] improved Klapper and Goresky's RA algorithm by transforming the MRFR problem into the minimal basis problem of a 2-dimensional lattice. Their algorithm is based on an efficient recursive design between the minimal bases of two adjacent lattices generated by the successive truncation sequences, which reduces the

Manuscript received November 1, 2020; revised July 30, 2021; accepted October 12, 2021. Date of publication November 8, 2021; date of current version January 20, 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 61702294, in part by the National Development Foundation of Cryptography under Grant MMJJ20170126, in part by the National Key Research and Development Program of China under Grant 2018YFA0704702, in part by the Department of Science and Technology of Shandong Province of China under Grant 2019JZZY010133, and in part by the Open Research Project of State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, under Grant 2016-MS-23. *(Corresponding author: Chengliang Tian.)*

Jun Che is with the College of Computer Science and Technology, Qingdao University, Qingdao 266071, China, and also with the State Key Laboratory of Cryptology, Beijing 100878, China (e-mail: chejun@mail.sdu.edu.cn).

Chengliang Tian is with the College of Computer Science and Technology, Qingdao University, Qingdao 266071, China (e-mail: tianchengliang@qdu.edu.cn).

Yupeng Jiang is with the School of Cyber Science and Technology, Beihang University, Beijing 100191, China (e-mail: jiangyupeng@amss.ac.cn).

Guangwu Xu is with the School of Cyber Science and Technology, Shandong University (Qingdao), Qingdao 266273, China (e-mail: gxu4sdq@sdu.edu.cn).

Communicated by G. Gong, Associate Editor for Sequences.

Digital Object Identifier 10.1109/TIT.2021.3125988

"actual" time complexity of the algorithm to $O(n^2)$. Here, "actual" means their analysis is mainly based on experimental results without a rigorous theoretical analysis. For non-adaptive yet faster algorithms, as early as the year 2004, Arnault *et al.* [2] developed Euclidean algorithm to compute the MRFR problem and the minimal basis of the 2-dimensional lattice generated by the sequence. Their algorithms are with a provable time complexity $O(n^2)$ and are faster than that of adaptive algorithms in practice. However, there seem to be some issues with their presentation and theoretical analysis. The authors had revised their original algorithm in [1] to remedy some error, but the new algorithm still fails to produce correct output for some input instances. The theoretical analysis is not thorough either, especially the arguments on the minimal basis is not completely correct. This means that their algorithm cannot always find a minimal basis. On the implementation side, their presented algorithms are also not optimal and can be further accelerated.

B. Our Contributions

In this paper, we revisit MRFR algorithms with more elaborate analysis and make substantial advances in this problem. Concretely, our contributions can be summarized as the following three aspects.

1) We find a general and concise recursive relationship between the minimal bases for two adjacent lattices by utilizing the properties of Lagrange-reduced basis of 2-dimensional lattices. This enables us to obtain an improved RA algorithm. Our new algorithm is with a provable time complexity $O(n^2)$ that fills the gap between the theory and the practice in the complexity analysis of Li *et al.* algorithm [7]. The practical running time of the algorithm is also improved over that of Li *et al.* algorithm, by 27%. Finally, our new algorithm is more general and can be adapted to compute the minimal basis under any computable norm.
2) Through optimizing a time-consuming step of the well-known Lagrange reduction algorithm for 2-dimensional lattices in ℓ_∞ norm, we design a non-adaptive, and yet practically faster MRFR-solving algorithm, named *global* Euclidean algorithm.
3) We analyze Arnault *et al.* Euclidean algorithm for MRFR problem [1], [2] and address some issues there, with some counter-examples. A revised Euclidean algorithm named *partial* Euclidean algorithm is designed. Moreover, by invoking the half-gcd algorithm, we further accelerate the *partial* Euclidean algorithm and propose an **Hgcd-Par-Euc** algorithm with a provable time complexity of $O(n \log^2 n \log\log n)$.

C. Road Map

The rest of our paper is arranged as follows. In Section II, some useful notation and terminologies are introduced, followed by a review of related mathematical concepts and properties frequently used in our paper. In Section III, we transform the MRFR problem into the minimal basis problem of a lattice, based on which, our improved adaptive RA algorithm is presented in Section IV. Later, two improved non-adaptive algorithms, the *global* Euclidean algorithm and the *partial* Euclidean algorithm, are proposed in Section V and Section VI, respectively. In Section VII, we compare the practical performance of our improved algorithms to that of the previous work with a comprehensive experimental analysis. Finally, we conclude our paper in Section VIII.

II. Preliminaries

A. Notation and Terminology

Throughout the paper, $\mathbb{Z}$ denotes the ring of integers, and $\mathbb{Q}$, $\mathbb{R}$ denote the fields of rational numbers and real numbers respectively. For $x, y \in \mathbb{R}$, $\lfloor x \rfloor$ is the biggest integer that is less than or equal to x, $\lceil x \rceil$ is the smallest integer that larger than or equal to x, $\#x = \lceil \log_2(|x|+1) \rceil$ denotes the bit size of x, $\#(x,y) = \max\{\#x, \#y\}$, $\underline{\#}(x,y) = \min\{\#x, \#y\}$, and

$$sgn(x) = \begin{cases} 1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases}$$

stands for the sign function of x. Moreover, we use bold upper (resp. lower) case letters to denote matrices (resp. column vectors). For a matrix $\mathbf{B}$ (resp. row vector $\mathbf{v}$), $\mathbf{B}^T$ (resp. $\mathbf{v}^T$) denotes the transpose of $\mathbf{B}$ (resp. $\mathbf{v}$). For some set A of vectors, $\min A$ (resp. $\max A$) denotes the vector with the minimal (resp. maximal) norm in A, and $\min_2 A = [\min A, \min(A \backslash \{\min A\})]$ denotes the nonzero shortest vector and the second shortest vector in A.

B. Norm

A norm is a function $\|\cdot\| : \mathbb{R}^m \to \mathbb{R}$ such that, $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^m, \forall c \in \mathbb{R}$,

- (Positive definite) $\|\mathbf{x}\| \geq 0$, and $\|\mathbf{x}\| = 0$ if and only if $\mathbf{x} = \mathbf{0}$.
- (Homogeneous) $\|c \cdot \mathbf{x}\| = |c| \cdot \|\mathbf{x}\|$.
- (Triangle inequality) $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$.

A common and important family of norm functions are given by the ℓ_p norms. For any $p \geq 1$, the ℓ_p norm of a vector $\mathbf{x} \in \mathbb{R}^m$ is $\|\mathbf{x}\|_p = \left(\sum_{i=1}^m |x_i|^p\right)^{1/p}$. Specially, $\|\mathbf{x}\|_\infty = \max_{1 \leq i \leq m} |x_i|$.

Lemma 1 ([10]): For three vectors $\mathbf{x}, \mathbf{x}+\mathbf{y}$, and $\mathbf{x}+\sigma\mathbf{y}$ with $\sigma \in (1, +\infty)$. For any computable norm $\|\cdot\|$, if $\|\mathbf{x}\| \leq \|\mathbf{x}+\mathbf{y}\|$, then $\|\mathbf{x}+\mathbf{y}\| \leq \|\mathbf{x}+\sigma\mathbf{y}\|$. Moreover, if $\|\mathbf{x}\| < \|\mathbf{x}+\mathbf{y}\|$, then $\|\mathbf{x}+\mathbf{y}\| < \|\mathbf{x}+\sigma\mathbf{y}\|$.

C. Lattices

A lattice is a set of all the integer linear combinations of certain linearly independent vectors. Concretely, let $\mathbf{b_1}, \mathbf{b_2}, \cdots, \mathbf{b_n}$ be n $(n \leq m)$ linearly independent vectors in $\mathbb{R}^m$, and the lattice $\mathcal{L}$ generated by them is

$$\mathcal{L} = L(\mathbf{B}) = \left\{ \sum_{i=1}^n z_i \mathbf{b}_i : z_i \in \mathbb{Z} \right\},$$

where the $m \times n$ matrix $\mathbf{B} = [\mathbf{b_1} \ \cdots \ \mathbf{b_n}]$ is called a basis of $\mathcal{L}$, m, n are called the dimension and the rank of $\mathcal{L}$ respectively.

Specially, if $m = n$, the lattice is called full-rank. Clearly, when $n \geq 2$, every lattice has infinitely many bases. In fact,

Lemma 2 ([10]): $\mathcal{L} = L(\mathbf{B}_1) = L(\mathbf{B}_2)$ if and only if $\mathbf{B}_1 = \mathbf{B}_2\mathbf{U}$ for some unimodular matrix $\mathbf{U} \in \mathbb{Z}^{n\times n}$.

The following three lattice concepts are closely related to our topic.

Definition 1: The determinant of a lattice $\mathcal{L}$ is defined as $\det(\mathcal{L}) = \sqrt{|\det(\mathbf{B}^T\mathbf{B})|}$.

It is easy to see that the determinant of a lattice is independent of the choice of the basis. For a full-rank lattice $\mathcal{L}$, $\det(\mathcal{L}) = |\det(\mathbf{B})|$.

Definition 2 (Successive minima): Let $\mathcal{L} = L(\mathbf{B}) \subset \mathbb{R}^m$ be a lattice of rank n $(n \leq m)$, for any computable norm $\|\cdot\|$, the ith $(1 \leq i \leq n)$ successive minimum of $\mathcal{L}$ is defined as the radius of the smallest ball centered at origin containing i linearly independent lattice vectors, i.e.,

$$\lambda_i(\mathcal{L}) = \inf\{r > 0 : \dim(\mathrm{span}(\mathcal{L} \cap \mathcal{B}(\mathbf{0}, r))) \geq i\},$$

where $\mathcal{B}(\mathbf{0}, r) = \{\mathbf{x} \in \mathbb{R}^m : \|\mathbf{x}\| < r\}$ denotes the open ball of radius r around $\mathbf{0}$. In particular, $\lambda_1(\mathcal{L}) = \min_{\mathbf{v}\in\mathcal{L}\setminus\{\mathbf{0}\}} \|\mathbf{v}\|$ is the length of the shortest non-zero lattice vector.

Definition 3 (Lagrange-reduced basis): For any computable norm $\|\cdot\|$, a basis $[\mathbf{a}, \mathbf{b}]$ of a 2-dimensional lattice $\mathcal{L}$ is called Lagrange-reduced if $\|\mathbf{a}\| \leq \|\mathbf{b}\| \leq \|\mathbf{a} - \mathbf{b}\|, \|\mathbf{a} + \mathbf{b}\|$.

Generally, a Lagrange-reduced basis is also called a minimal basis due to the following property.

Lemma 3 ([10]): Let $\mathcal{L} = L([\mathbf{a}, \mathbf{b}]) \subset \mathbb{R}^2$ be a 2-dimensional lattice. Then, the basis $[\mathbf{a}, \mathbf{b}]$ is Lagrange-reduced if and only if $\|\mathbf{a}\| = \lambda_1(\mathcal{L}), \|\mathbf{b}\| = \lambda_2(\mathcal{L})$.

It is worth noting that the successive minima and the Lagrange-reduced basis can be defined with respect to any computable norm. Here and in the rest of our paper, unless particularly stated, we refer to $\|\mathbf{x}\|$ as the ℓ_∞ norm of $\mathbf{x}$.

III. Relationship Between Rational Fraction Representation and Lattices

In this section, we make an important observation about the relationship between lattices and rational fraction representation. Firstly, we give some necessary lemmas.

Lemma 4: Given a binary sequence $\underline{a}(n)$, if $\frac{p}{q}$ is a RFR of $\underline{a}(n)$, then the column vector $(p\ q)^T \in \mathcal{L}_n = L([\mathbf{a}\ \mathbf{b}])$, where $\mathbf{a} = (S_n\ 1)^T, \mathbf{b} = (2^n\ 0)^T$.

Proof: Since $\frac{p}{q}$ is a rational fraction representation of $\underline{a}(n)$, $p \equiv q \cdot S_n \bmod 2^n$. Namely, $\exists k \in \mathbb{Z}$, such that $p = q \cdot S_n + k \cdot 2^n$. Therefore,

$$\begin{aligned}(p\ q)^T &= (q{\cdot}S_n + k{\cdot}2^n\ q)^T = (q{\cdot}S_n\ q)^T + (k{\cdot}2^n\ 0)^T \\ &= q{\cdot}(S_n\ 1)^T + k{\cdot}(2^n\ 0)^T. \end{aligned} \tag{1}$$

Let $\mathbf{a} = (S_n\ 1)^T, \mathbf{b} = (2^n\ 0)^T$, then equation (1) indicates $(p\ q)^T \in \mathcal{L}_n$. □

Based on Lemma 4, it is easy to transform the problem of finding the MRFR of some given sequence into that of finding a minimal basis of the corresponding 2-dimensional lattice. We summarize the main result in the following theorem which, in essence, has been proved in [7] (Lemma 7).

Theorem 1 ([7]): Given a binary sequence $\underline{a}(n)$ with two vectors $\mathbf{a} = (S_n\ 1)^T, \mathbf{b} = (2^n\ 0)^T$, and let $\boldsymbol{\alpha} = (p\ q)^T$, $\boldsymbol{\beta} = (p'\ q')^T$ be a new basis for $\mathcal{L}_n = L([\mathbf{a}\ \mathbf{b}])$ with $\|\boldsymbol{\alpha}\| = \lambda_1(\mathcal{L}_n), \|\boldsymbol{\beta}\| = \lambda_2(\mathcal{L}_n)$. Without loss of generality, we assume $q > 0, q' > 0$. Then

1) If q is odd, then $\frac{p}{q}$ is an MRFR of $\underline{a}(n)$,
2) If q is even, then $\frac{p'}{q'}$ is an MRFR of $\underline{a}(n)$.

IV. Improved Rational Approximation Algorithm

A. Klapper and Goresky's RA and Li et al. RA Algorithms Reviewed

Given a sequence $\underline{a}(n) = (a_0\ a_1 \cdots a_{n-1})$, we consider the k-truncation sequence $\underline{a}(k) = (a_0\ \cdots a_{k-1})$. The main idea underlying the adaptive RA algorithm is seeking an efficient recursive formula between the MRFR of $\underline{a}(k)$ and that of $\underline{a}(k+1)$. In 1995, Klapper and Goresky [5] designed an adaptive RA algorithm based on the following recursive relationship.

Theorem 2 ([5]): Let $\mathcal{L}_k = L([\mathbf{a}_k\ \mathbf{b}_k])$ with $\mathbf{a}_k = (S_k\ 1)^T$, $\mathbf{b}_k = (2^k\ 0)^T$ be the lattice corresponding to the k-truncation sequence $\underline{a}(k) = (a_0\ \cdots\ a_{k-1})$. Assuming $\mathbf{f}^{(k)}, \mathbf{g}^{(k)} \in \mathcal{L}_k$ satisfy

$$\begin{aligned}\|\mathbf{f}^{(k)}\| &= \min\{\|\mathbf{h}\| : \mathbf{h} = (h_1\ h_2) \in \mathcal{L}_k \text{ with } h_1, h_2 \text{ even}\}, \\ \|\mathbf{g}^{(k)}\| &= \min\{\|\mathbf{h}\| : \mathbf{h} = (h_1\ h_2) \in \mathcal{L}_k \text{ with } h_2 \text{ odd}\}.\end{aligned}$$

Then, $S_{k+1} = S_k + a_k 2^k$ and

1) if $S_{k+1} \cdot g_2^{(k)} - g_1^{(k)} \equiv 0 \bmod 2^{k+1}$, $[\mathbf{f}^{(k+1)}, \mathbf{g}^{(k+1)}] = [2\mathbf{f}^{(k)}, \mathbf{g}^{(k)}]$.
2) if $S_{k+1} \cdot g_2^{(k)} - g_1^{(k)} \not\equiv 0 \bmod 2^{k+1}$ and $\|\mathbf{g}^{(k)}\| \leq \|\mathbf{f}^{(k)}\|$, $[\mathbf{g}^{(k+1)}, \mathbf{f}^{(k+1)}] = [\mathbf{f}^{(k)} + d\mathbf{g}^{(k)}, 2\mathbf{g}^{(k)}]$, where d is odd and minimize $\|\mathbf{f}^{(k)} + z\mathbf{g}^{(k)}\|$ for any odd $z \in \mathbf{Z}$.
3) if $S_{k+1} \cdot g_2^{(k)} - g_1^{(k)} \not\equiv 0 \bmod 2^{k+1}$ and $\|\mathbf{g}^{(k)}\| > \|\mathbf{f}^{(k)}\|$, $[\mathbf{g}^{(k+1)}, \mathbf{f}^{(k+1)}] = [\mathbf{g}^{(k)} + d\mathbf{f}^{(k)}, 2\mathbf{f}^{(k)}]$, where d is odd and minimize $\|\mathbf{g}^{(k)} + z\mathbf{f}^{(k)}\|$ for any odd $z \in \mathbf{Z}$.

The procedure corresponding the theorem is given in Appendix A. Since the time-consuming operations of large integer multiplication in step 6, step 9 and step 12, the asymptotic complexity of their algorithm is $O\left(n^2 \log n \log\log n\right)$.

Li *et al.* [7] made good use of the transformation relationship between the MRFR and the Lagrange-reduced basis shown in Theorem 1, and found a new recursive relationship between the Lagrange-reduced basis of $\mathcal{L}_k$ and that of $\mathcal{L}_{k+1}$. More precisely, they proved

Theorem 3 ([7]): Let $\underline{a}(k) = (a_0\ a_1\ \cdots\ a_{k-1})$ be the k-truncation sequence, and $\mathcal{L}_k = L([\mathbf{a}_k\ \mathbf{b}_k])$ is the lattice with $\mathbf{a}_k = (S_k\ 1)^T$, $\mathbf{b}_k = (2^k\ 0)^T$. Assuming $[\mathbf{f}^{(k)}\ \mathbf{g}^{(k)}]$ is a Lagrange-reduced basis of $\mathcal{L}_k$ (i.e. $\|\mathbf{f}^{(k)}\| = \lambda_1(\mathcal{L}_k)$, $\|\mathbf{g}^{(k)}\| = \lambda_2(\mathcal{L}_k)$) with $\mathbf{f}^{(k)} = z_{11}^{(k)}\mathbf{a}_k + z_{12}^{(k)}\mathbf{b}_k, \mathbf{g}^{(k)} = z_{21}^{(k)}\mathbf{a}_k + z_{22}^{(k)}\mathbf{b}_k$ Then,

1) if $-a_k z_{11}^{(k)} + z_{12}^{(k)} \equiv 0 \mod 2$, $\mathbf{f}^{(k+1)} = \mathbf{f}^{(k)}$, $\mathbf{g}^{(k+1)} = 2\mathbf{g}^{(k)} - c\mathbf{f}^{(k+1)}$ for some efficiently computable integer c.
2) if $-a_k z_{11}^{(k)} + z_{12}^{(k)} \equiv 1 \mod 2$ and $-a_k z_{21}^{(k)} + z_{22}^{(k)} \equiv 0 \mod 2$, $\mathbf{f}^{(k+1)} = \min\{2\mathbf{f}^{(k)}, \mathbf{g}^{(k)}\}$, and $\mathbf{g}^{(k+1)} = \max\{2\mathbf{f}^{(k)}, \mathbf{g}^{(k)}\} - c\mathbf{f}^{(k+1)}$ for some efficiently computable integer c.
3) if $-a_k z_{11}^{(k)} + z_{12}^{(k)} \equiv 1 \mod 2$ and $-a_k z_{21}^{(k)} + z_{22}^{(k)} \equiv 1 \mod 2$, $\mathbf{f}^{(k+1)} = \min\{2\mathbf{f}^{(k)}, \mathbf{f}^{(k)} + \mathbf{g}^{(k)}, \mathbf{f}^{(k)} -$

$\mathbf{g}^{(k)}\}$ and $\mathbf{g}^{(k+1)} = \min\{\{2\mathbf{f}^{(k)}, \mathbf{f}^{(k)} + \mathbf{g}^{(k)}, \mathbf{f}^{(k)} - \mathbf{g}^{(k)}\}\backslash\{\mathbf{f}^{(k+1)}\}\} - c\mathbf{f}^{(k+1)}$ for some efficiently computable integer c.

It is not difficult to see that, for each iteration, there are only large integer additions and subtractions involved except for the computation of c. Through a complicate analysis based on the linear programming (LP) under the ℓ_∞ norm, the authors deduced a specific computation formula for c, which involved in large integer divisions. However, as stated by the authors in [7], "Through a lot of experiments, we find that both the absolute values of a_1 and a_2 are less than or equal to 2 indeed. As a result, for the complexity of LP algorithm, we choose to use a few additions and subtractions of at most k bit integers to avoid the multiplications and divisions of two large numbers, $\cdots\cdots$. The actual time complexity of our new algorithm is thus $O(n^2)$ in terms of n". In their description, a_1, a_2 refer to the estimation values of c. Thus, the "actual" time complexity here is heuristic.

B. Our Improved Rational Approximation Algorithm

In this subsection, we establish a new recursive relationship between the Lagrange-reduced basis of $\mathcal{L}_k$ and that of $\mathcal{L}_{k+1}$ by taking ideas similar to Li *et al.* Compared with prior work, our improved formula is more precise and concise. With the same notations as in Theorem 3, we give a precise computation of $\mathbf{g}^{(k+1)}$ with a simple theoretical analysis, which fills the gap between the theory and the practice in the complexity analysis of Li *et al.*algorithm. Moreover, our proof is more general in that it is independent of the choice of the norm. This means that our improved algorithm can be adapted to find a Lagrange-reduce basis of $\mathcal{L}_k$ with respect to any computable norm. More specifically, we have

Theorem 4: Let $\underline{a}(k) = (a_0\ a_1\ \cdots\ a_{k-1})$ be the k-truncation sequence of $\underline{a}(n)$, and let $\mathcal{L}_k = L([\mathbf{a}_k\ \mathbf{b}_k])$ with $\mathbf{a}_k = (S_k\ 1)^T$, $\mathbf{b}_k = (2^k\ 0)^T$ be the lattice generated by the sequence $\underline{a}(k)$. Then $2\mathcal{L}_k \subset \mathcal{L}_{k+1} \subset \mathcal{L}_k$. Further, let $[\boldsymbol{\alpha}_k\ \boldsymbol{\beta}_k]$ with $\boldsymbol{\alpha}_k = z_{11}^{(k)}\mathbf{a}_k + z_{12}^{(k)}\mathbf{b}_k, \boldsymbol{\beta}_k = z_{21}^{(k)}\mathbf{a}_k + z_{22}^{(k)}\mathbf{b}_k$ be a Lagrange-reduced basis of $\mathcal{L}_k$. Then

1) if $-a_k z_{11}^{(k)} + z_{12}^{(k)} \equiv 0 \mod 2$, we have $\boldsymbol{\alpha}_{k+1} = \boldsymbol{\alpha}_k, \boldsymbol{\beta}_{k+1} = \min\{2\boldsymbol{\beta}_k, \boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k, \boldsymbol{\alpha}_k - 2\boldsymbol{\beta}_k\}$.
2) if $-a_k z_{11}^{(k)} + z_{12}^{(k)} \equiv 1 \mod 2$ and $-a_k z_{21}^{(k)} + z_{22}^{(k)} \equiv 0 \mod 2$, we have $[\boldsymbol{\alpha}_{k+1}, \boldsymbol{\beta}_{k+1}] = \min_2\{2\boldsymbol{\alpha}_k, \boldsymbol{\beta}_k, 2\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k, 2\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\}$.
3) if $-a_k z_{11}^{(k)} + z_{12}^{(k)} \equiv 1 \mod 2$ and $-a_k z_{21}^{(k)} + z_{22}^{(k)} \equiv 1 \mod 2$, we have $[\boldsymbol{\alpha}_{k+1}, \boldsymbol{\beta}_{k+1}] = \min_2\{2\boldsymbol{\alpha}_k, \boldsymbol{\alpha}_k - \boldsymbol{\beta}_k, \boldsymbol{\alpha}_k + \boldsymbol{\beta}_k, 3\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k, 3\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\}$.

Proof: Since $\mathcal{L}_{k+1} = L([\mathbf{a}_{k+1}\ \mathbf{b}_{k+1}])$ with $\mathbf{a}_{k+1} = (S_{k+1}\ 1)^T$, $\mathbf{b}_{k+1} = (2^{k+1}\ 0)^T$ and $S_{k+1} = S_k + a_k 2^k$, we have $\mathbf{a}_{k+1} = \mathbf{a}_k + a_k\mathbf{b}_k \in \mathcal{L}_k$, $\mathbf{b}_{k+1} = (2^{k+1}\ 0) = 2\mathbf{b}_k \in \mathcal{L}_k$. That is, $\mathcal{L}_{k+1} \subset \mathcal{L}_k$. Meanwhile, $2\mathbf{a}_k = 2\mathbf{a}_{k+1} - 2a_k\mathbf{b}_k = 2\mathbf{a}_{k+1} - a_k\mathbf{b}_{k+1} \in \mathcal{L}_k$, $2\mathbf{b}_k = \mathbf{b}_{k+1} \in \mathcal{L}_k$. Thus, $2\mathcal{L}_k \subset \mathcal{L}_{k+1}$.

Since $\boldsymbol{\alpha}_k = z_{11}^{(k)}\mathbf{a}_k + z_{12}^{(k)}\mathbf{b}_k = z_{11}^{(k)}\mathbf{a}_{k+1} + (-a_k z_{11}^{(k)} + z_{12}^{(k)})\mathbf{b}_k = z_{11}^{(k)}\mathbf{a}_{k+1} + \frac{-a_k z_{11}^{(k)} + z_{12}^{(k)}}{2}\mathbf{b}_{k+1}$, and $\boldsymbol{\beta}_k = z_{21}^{(k)}\mathbf{a}_k + z_{22}^{(k)}\mathbf{b}_k = z_{21}^{(k)}\mathbf{a}_{k+1} + (-a_k z_{21}^{(k)} + z_{22}^{(k)})\mathbf{b}_k = z_{21}^{(k)}\mathbf{a}_{k+1} + \frac{-a_k z_{21}^{(k)} + z_{22}^{(k)}}{2}\mathbf{b}_{k+1}$, we have

$$\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k = (z_{11}^{(k)} + z_{21}^{(k)})\mathbf{a}_{k+1} + \left(\frac{-a_k z_{11}^{(k)} + z_{12}^{(k)}}{2} + \frac{-a_k z_{21}^{(k)} + z_{22}^{(k)}}{2}\right)\mathbf{b}_{k+1}, \quad (2)$$

$$\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k = (z_{11}^{(k)} - z_{21}^{(k)})\mathbf{a}_{k+1} + \left(\frac{-a_k z_{11}^{(k)} + z_{12}^{(k)}}{2} - \frac{-a_k z_{21}^{(k)} + z_{22}^{(k)}}{2}\right)\mathbf{b}_{k+1}. \quad (3)$$

Now, we analyze the results as follows.

(1) If $-a_k z_{11}^{(k)} + z_{12}^{(k)} \equiv 0 \mod 2$, then $\boldsymbol{\alpha}_k \in \mathcal{L}_{k+1}$. In this case, $\boldsymbol{\alpha}_{k+1} = \boldsymbol{\alpha}_k$ and $\boldsymbol{\beta}_{k+1}$ is the vector achieving the minimal norm in the set $\{2\boldsymbol{\beta}_k, \boldsymbol{\alpha}_k - 2\boldsymbol{\beta}_k, \boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k\}$. We discuss in the following three cases.

(1.1) $\|2\boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k - 2\boldsymbol{\beta}_k\|, \|\boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k\|$. In this case, $\det([\boldsymbol{\alpha}_k\ 2\boldsymbol{\beta}_k]) = 2\det(\mathcal{L}_k) = \det(\mathcal{L}_{k+1})$. Hence $[\boldsymbol{\alpha}_k\ 2\boldsymbol{\beta}_k]$ is a basis of $\mathcal{L}_{k+1}$. Meanwhile, $\|\boldsymbol{\alpha}_k\| \le \|2\boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k - 2\boldsymbol{\beta}_k\|, \|\boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k\|$, so, by Defintion 3, $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [\boldsymbol{\alpha}_k\ 2\boldsymbol{\beta}_k]$ is a Lagrage-reduced basis of $\mathcal{L}_{k+1}$.

(1.2) $\|\boldsymbol{\alpha}_k - 2\boldsymbol{\beta}_k\| \le \|2\boldsymbol{\beta}_k\|, \|\boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k\|$. In this case, $\det([\boldsymbol{\alpha}_k\ \boldsymbol{\alpha}_k - 2\boldsymbol{\beta}_k]) = 2\det(\mathcal{L}_k) = \det(\mathcal{L}_{k+1})$. Hence $[\boldsymbol{\alpha}_k\ \boldsymbol{\alpha}_k - 2\boldsymbol{\beta}_k]$ is a basis of $\mathcal{L}_{k+1}$. Meanwhile, since $[\boldsymbol{\alpha_k}\ \boldsymbol{\beta}_k]$ is Lagrange-reduced, by Defintion 3, we have $\|\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|, \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$. Thus, by Lemma 1, $\|\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\alpha}_k - 2\boldsymbol{\beta}_k\| \le \|2\boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k - 2\boldsymbol{\beta}_k\|$, and, by Defintion 3, $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [\boldsymbol{\alpha}_k\ \boldsymbol{\alpha}_k - 2\boldsymbol{\beta}_k]$ is a Lagrage-reduced basis of $\mathcal{L}_{k+1}$.

(1.3) $\|\boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k\| \le \|2\boldsymbol{\beta}_k\|, \|\boldsymbol{\alpha}_k - 2\boldsymbol{\beta}_k\|$. In this case, $\det([\boldsymbol{\alpha}_k\ \boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k]) = 2\det(\mathcal{L}_k) = \det(\mathcal{L}_{k+1})$. Hence $[\boldsymbol{\alpha}_k\ \boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k]$ is a basis of $\mathcal{L}_{k+1}$. Meanwhile, since $[\boldsymbol{\alpha}_k\ \boldsymbol{\beta}_k]$ is Lagrange-reduced, by Definition 3, we have $\|\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|, \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$. Thus, by Lemma 1, $\|\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k\| \le \|2\boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k\|$, and, by Defintion 3, $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [\boldsymbol{\alpha}_k\ \boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k]$ is a Lagrage-reduced basis of $\mathcal{L}_{k+1}$.

(2) If $-a_k z_{21}^{(k)} + z_{22}^{(k)} \equiv 0 \mod 2$, then $\beta_k \in \mathcal{L}_{k+1}$. In this case, $\boldsymbol{\alpha}_{k+1}$ and $\boldsymbol{\beta}_{k+1}$ are the shortest two vectors in the set $\{2\boldsymbol{\alpha}_k, \boldsymbol{\beta}_k, 2\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k, 2\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\}$. Since $\|\boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|, \|2\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$, we discuss in the following four cases.

(2.1) $\|2\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|, \|2\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$. In this case, $\det([2\boldsymbol{\alpha}_k\ \boldsymbol{\beta}_k]) = 2\det(\mathcal{L}_k) = \det(\mathcal{L}_{k+1})$. Hence $[2\boldsymbol{\alpha}_k\ \boldsymbol{\beta}_k]$ is a basis of $\mathcal{L}_{k+1}$. Meanwhile, $\|2\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|, \|2\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$, so, by Defintion 3, $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [2\boldsymbol{\alpha}_k\ \boldsymbol{\beta}_k]$ is a Lagrage-reduced basis of $\mathcal{L}_{k+1}$.

(2.2) $\|\boldsymbol{\beta}_k\| < \|2\boldsymbol{\alpha}_k\| \le \|2\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|, \|2\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$. This case is essentially the same as case (2.1). $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [\boldsymbol{\beta}_k\ 2\boldsymbol{\alpha}_k]$.

(2.3) $\|\boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k\|, \|2\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$. In this case, $\det([\boldsymbol{\beta}_k\ 2\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k]) = 2\det(\mathcal{L}_k) = \det(\mathcal{L}_{k+1})$. Hence $[\boldsymbol{\beta}_k\ 2\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k]$ is a basis of $\mathcal{L}_{k+1}$. Meanwhile, since $[\boldsymbol{\alpha}_k\ \boldsymbol{\beta}_k]$ is Lagrange-reduced, by Definition 3, we have $\|\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|, \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$. Thus, by Lemma 1, $\|\boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k\| \le$

$\|2\boldsymbol{\alpha}_k - 2\boldsymbol{\beta}_k\|$, and, by Defintion 3, $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [\boldsymbol{\beta}_k\ 2\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k]$ is a Lagrage-reduced basis of $\mathcal{L}_{k+1}$.

(2.4) $\|\boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k\|, \|2\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|$. In this case, $\det([\boldsymbol{\alpha}_k\ \boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k]) = 2\det(\mathcal{L}_k) = \det(\mathcal{L}_{k+1})$. Hence $[\boldsymbol{\alpha}_k\ \boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k]$ is a basis of $\mathcal{L}_{k+1}$. Meanwhile, since $[\boldsymbol{\alpha}_k\ \boldsymbol{\beta}_k]$ is Lagrange-reduced, by Lemma 3, we have $\|\boldsymbol{\alpha}_k\| = \lambda_1(\mathcal{L}_k), \|\boldsymbol{\beta}_k\| = \lambda_2(\mathcal{L}_k)$ and $\|\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|, \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$. Thus, $\|\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k\|, \|2\boldsymbol{\beta}_k\|$, and, by Defintion 3, $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [\boldsymbol{\beta}_k\ 2\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k]$ is a Lagrage-reduced basis of $\mathcal{L}_{k+1}$.

(3) If $-a_k z_{11}^{(k)} + z_{12}^{(k)} \equiv 1 \mod 2$ and $-a_k z_{21}^{(k)} + z_{22}^{(k)} \equiv 1 \mod 2$, then, by equations (2) and (3), $\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k, \boldsymbol{\alpha}_k - \boldsymbol{\beta}_k \in \mathcal{L}_{k+1}$. Since $\|\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k \pm \boldsymbol{\beta}_k\|$, by Lemma 1, we have $\|\boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k \pm \boldsymbol{\beta}_k\| \le \|3\boldsymbol{\alpha}_k \pm \boldsymbol{\beta}_k\|$. Therefore, $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}]$ can be constructed as the following cases.

(3.1) $\|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k\|$. In this case, $\det([\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\ \boldsymbol{\alpha}_k + \boldsymbol{\beta}_k]) = 2\det(\mathcal{L}_k) = \det(\mathcal{L}_{k+1})$. Hence $[\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\ \boldsymbol{\alpha}_k + \boldsymbol{\beta}_k]$ is a basis of $\mathcal{L}_{k+1}$. Meanwhile, $\|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k\| \le \|2\boldsymbol{\beta}_k\|$, so, by Defintion 3, $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\ \boldsymbol{\alpha}_k + \boldsymbol{\beta}_k]$ is a Lagrage-reduced basis of $\mathcal{L}_{k+1}$.

(3.2) $\|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\| < \|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k\|$. This case is essentially the same as case (3.1), $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\ \boldsymbol{\alpha}_k - \boldsymbol{\beta}_k]$.

(3.3) $\|2\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$. In this case, $\det([2\boldsymbol{\alpha}_k\ \boldsymbol{\alpha}_k - \boldsymbol{\beta}_k]) = 2\det(\mathcal{L}_k) = \det(\mathcal{L}_{k+1})$. Hence $[\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\ 2\boldsymbol{\alpha}_k]$ is a basis of $\mathcal{L}_{k+1}$. Since $\|\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|, \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$, by Lemma 1, we have $\|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\| \le \|3\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|$. Thus, $\|2\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|, \|3\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|$, and, by Defintion 3, $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [2\boldsymbol{\alpha}_k\ \boldsymbol{\alpha}_k - \boldsymbol{\beta}_k]$ is a Lagrage-reduced basis of $\mathcal{L}_{k+1}$.

(3.4) $\|2\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\| < \|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|$. In this case, $\det([2\boldsymbol{\alpha}_k\ \boldsymbol{\alpha}_k + \boldsymbol{\beta}_k]) = 2\det(\mathcal{L}_k) = \det(\mathcal{L}_{k+1})$. Hence $[2\boldsymbol{\alpha}_k\ \boldsymbol{\alpha}_k + \boldsymbol{\beta}_k]$ is a basis of $\mathcal{L}_{k+1}$. Since $\|\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|, \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$, by Lemma 1, we have $\|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\| \le \|3\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$. Thus, $\|2\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|, \|3\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$, and, by Defintion 3, $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [2\boldsymbol{\alpha}_k\ \boldsymbol{\alpha}_k + \boldsymbol{\beta}_k]$ is a Lagrage-reduced basis of $\mathcal{L}_{k+1}$.

(3.5) $\|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|, \|3\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|$. In this case, $\det([\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\ 2\boldsymbol{\alpha}_k]) = 2\det(\mathcal{L}_k) = \det(\mathcal{L}_{k+1})$. Hence $[\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\ 2\boldsymbol{\alpha}_k]$ is a basis of $\mathcal{L}_{k+1}$. Thus, by Defintion 3, $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\ 2\boldsymbol{\alpha}_k]$ is a Lagrage-reduced basis of $\mathcal{L}_{k+1}$.

(3.6) $\|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\| \le \|3\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k\|, \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$. In this case, $\det([\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\ 3\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k]) = 2\det(\mathcal{L}_k) = \det(\mathcal{L}_{k+1})$. Hence $[\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\ 3\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k]$ is a basis of $\mathcal{L}_{k+1}$. Meanwhile, since $[\boldsymbol{\alpha}_k\ \boldsymbol{\beta}_k]$ is Lagrange-reduced, by Lemma 3, we have $\|\boldsymbol{\alpha}_k\| = \lambda_1(\mathcal{L}_k), \|\boldsymbol{\beta}_k\| = \lambda_2(\mathcal{L}_k)$ and $\|\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\beta}_k\| \le \|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|, \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$. Thus, $\|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\| \le \|3\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k\| \le \|2(2\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k)\| = \|4\boldsymbol{\alpha}_k - 2\boldsymbol{\beta}_k\|$, and, by Defintion 3, $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\ 3\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k]$ is a Lagrage-reduced basis of $\mathcal{L}_{k+1}$.

(3.7) $\|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k\| \le \|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|, \|3\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|$. In this case, $\det([\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\ 2\boldsymbol{\alpha}_k]) = 2\det(\mathcal{L}_k) = \det(\mathcal{L}_{k+1})$. Hence $[\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\ 2\boldsymbol{\alpha}_k]$ is a basis of $\mathcal{L}_{k+1}$. Thus, by Defintion 3, $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\ 2\boldsymbol{\alpha}_k]$ is a Lagrage-reduced basis of $\mathcal{L}_{k+1}$.

(3.8) $\|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\| \le \|3\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k\|, \|\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k\|$. In this case, $\det([\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\ 3\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k]) = 2\det(\mathcal{L}_k) = \det(\mathcal{L}_{k+1})$. Hence $[\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\ 3\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k]$ is a basis of $\mathcal{L}_{k+1}$. Meanwhile, since $[\boldsymbol{\alpha}_k\ \boldsymbol{\beta}_k]$ is Lagrange-reduced, by Definition 3, we have $\|\alpha_k\| \le \|\beta_k\| \le \|\alpha_k - \beta_k\|, \|\alpha_k + \beta_k\|$. Thus, by Lemma 1, $\|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\| \le \|3\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\| \le \|2\boldsymbol{\alpha}_k\| \le \|2(2\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k)\| = \|4\boldsymbol{\alpha}_k + 2\boldsymbol{\beta}_k\|$, and, by Defintion 3, $[\boldsymbol{\alpha}_{k+1}\ \boldsymbol{\beta}_{k+1}] = [\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\ 3\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k]$ is a Lagrage-reduced basis of $\mathcal{L}_{k+1}$.

□

Theorem 4 and its proof can be readily turned to the pseudocode as shown in Algorithm 1. Since there are only large integer additions and subtractions involved in the operation of each iteration of our improved algorithm, it is immediate to get

Theorem 5: On inputting a binary sequence $\underline{a}(n) = (a_0\ a_1\ \cdots\ a_{n-1})$, the Algorithm 1 outputs an MRFR of $\underline{a}(n)$ in time $O(n^2)$.

V. Global Euclidean Algorithm

The analysis in Section III indicates that how to find a minimal basis of the 2-dimensional lattice associated with the input binary sequence $\underline{a}(n)$ is of great concern. In this section, we present our first faster algorithm for the MRFR problem through optimizing the existing algorithm for minimal basis problem.

A. Lagrange Reduction Algorithm Revisited

As early as in 1773, Lagrange presented an efficient algorithm to output a shortest basis for any 2-dimensional lattice. Micciancio and Goldwasser [10] reviewed Lagrange's reduction algorithm as shown in Appendix B, and proved the following results.

Theorem 6 ([10]): For any lattice basis vectors $\mathbf{a} = (a_1\ a_2)^T$, $\mathbf{b} = (b_1\ b_2)^T \in \mathbb{Z}^2$ of a lattice $\mathcal{L}$, and any computable norm $\|\cdot\|$, the Algorithm 6 outputs a shortest basis $[\mathbf{a}^{new}\ \mathbf{b}^{new}]$ with $\|\mathbf{a}^{new}\| = \lambda_1(\mathcal{L})$ and $\|\mathbf{b}^{new}\| = \lambda_2(\mathcal{L})$ in time $(2 + \log\max\{\|\mathbf{a}\|, \|\mathbf{b}\|\}) \cdot T$, where T denotes the time overhead of once loop.

Clearly, the main idea underlying the algorithm is the alternately mutual reduction between two lattice vectors, which can be treated as the generalization of the well-known Euclidean algorithm over integers. However, different from the Euclidean division over integers, the computation of the "quotient" μ (*i.e.*, step 7) is the most expensive step in each loop of Algorithm 6. By an important observation presented in Lemma 5, Micciancio and Goldwasser [10] suggested a binary search strategy. Despite it can find the correct μ in $O(\log(\|\mathbf{b}\|/\|\mathbf{a}\|))$ steps, when $[\mathbf{a}\ \mathbf{b}]$ is a matrix with large integer entries, the time overhead is still unsatisfactory.

Lemma 5 ([10]): At the beginning of each loop in Algorithm 6, the vectors $\mathbf{a}, \mathbf{b}$ satisfy $\|\mathbf{b}\| > \|\mathbf{b} - \mathbf{a}\|$, and the integer μ that minimizes $\|\mathbf{b} - \mu\mathbf{a}\|$ belongs to $[1, 2\|\mathbf{b}\|/\|\mathbf{a}\|]$, where $\|\cdot\|$ denotes any computable norm.

Algorithm 1 Improved RA

Input: A sequence $\underline{a}(n) = (a_0\ a_1\ \cdots\ a_{n-1})$.
Output: A vector $(p\ q)$ with p/q being an MRFR of the sequence $\underline{a}(n)$
1: input $a_i's$ until the first nonzero a_{k-1} is found
2: if $k = 1$, then
3: $\boldsymbol{\alpha} = (1\ 1)$ $\boldsymbol{\beta} = (-1\ 1)$, $z_{11} = 1$, $z_{12} = 0$, $z_{21} = 1$, $z_{22} = -1$
4: if $k \geq 2$, then
5: $\boldsymbol{\alpha} = (0\ 2)$ $\boldsymbol{\beta} = (2^{k-1}\ 1)$, $z_{11} = 2$, $z_{12} = -1$, $z_{21} = 1$, $z_{22} = 0$
6: **while** there are more bits, **do**
7: input a new bit a_k
8: **if** $-a_k z_{11} + z_{12} \equiv 0 \mod 2$, **then**
9: $z_{12} := \frac{-a_k z_{11} + z_{12}}{2}$
10: **if** $\|2\boldsymbol{\beta}\| \leq \|\boldsymbol{\alpha} - 2\boldsymbol{\beta}\|, \|\boldsymbol{\alpha} + 2\boldsymbol{\beta}\|$, **then**
11: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [\boldsymbol{\alpha}\ 2\boldsymbol{\beta}]$, $z_{22} := -a_k z_{21} + z_{22}$, $z_{21} = 2z_{21}$.
12: **else if** $\|\boldsymbol{\alpha} - 2\boldsymbol{\beta}\| \leq \|2\boldsymbol{\beta}\|, \|\boldsymbol{\alpha} + 2\boldsymbol{\beta}\|$, **then**
13: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [\boldsymbol{\alpha}\ \boldsymbol{\alpha} - 2\boldsymbol{\beta}]$, $z_{22} := z_{12} + a_k z_{21} - z_{22}$, $z_{21} := z_{11} - 2z_{21}$.
14: **else**
15: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [\boldsymbol{\alpha}\ \boldsymbol{\alpha} + 2\boldsymbol{\beta}]$, $z_{22} := z_{12} - a_k z_{21} - z_{22}$, $z_{21} := z_{11} + 2z_{21}$.
16: **else if** $-a_k z_{21} + z_{22} \equiv 0 \mod 2$, **then**
17: **if** $\|2\boldsymbol{\alpha}\| \leq \|\boldsymbol{\beta}\| \leq \|2\boldsymbol{\alpha} - \boldsymbol{\beta}\|, \|2\boldsymbol{\alpha} + \boldsymbol{\beta}\|$, **then**
18: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [2\boldsymbol{\alpha}\ \boldsymbol{\beta}]$, $z_{12} := -a_k z_{11} + z_{12}$, $z_{11} := 2z_{11}$, $z_{22} := \frac{-a_k z_{21} + z_{22}}{2}$
19: **else if** $\|\boldsymbol{\beta}\| < \|2\boldsymbol{\alpha}\| \leq \|2\boldsymbol{\alpha} - \boldsymbol{\beta}\|, \|2\boldsymbol{\alpha} + \boldsymbol{\beta}\|$, **then**
20: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [\boldsymbol{\beta}\ 2\boldsymbol{\alpha}]$, $swap(z_{12}, z_{22})$, $z_{22} := -a_k z_{11} + z_{22}, z_{12} := \frac{-a_k z_{21} + z_{12}}{2}, swap(z_{11}, z_{21}), z_{21} := 2z_{21}$
21: **else if** $\|\boldsymbol{\beta}\| \leq \|2\boldsymbol{\alpha} - \boldsymbol{\beta}\| \leq \|2\boldsymbol{\alpha}\|, \|2\boldsymbol{\alpha} + \boldsymbol{\beta}\|$, **then**
22: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [\boldsymbol{\beta}\ 2\boldsymbol{\alpha} - \boldsymbol{\beta}]$, $swap(z_{12}, z_{22}), z_{12} := \frac{-a_k z_{21} + z_{12}}{2}, z_{22} := -a_k z_{11} + z_{22} - z_{12}, swap(z_{11}, z_{21}), z_{21} := 2z_{21} - z_{11}$
23: **else**
24: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [\boldsymbol{\beta}\ 2\boldsymbol{\alpha} + \boldsymbol{\beta}]$, $swap(z_{12}, z_{22}), z_{12} := \frac{-a_k z_{21} + z_{12}}{2}, z_{22} := -a_k z_{11} + z_{22} + z_{12}, swap(z_{11}, z_{21}), z_{21} := 2z_{21} + z_{11}$
25: **else**
26: **if** $\|\boldsymbol{\alpha} - \boldsymbol{\beta}\| \leq \|\boldsymbol{\alpha} + \boldsymbol{\beta}\| \leq \|2\boldsymbol{\alpha}\|$, **then**
27: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [\boldsymbol{\alpha} - \boldsymbol{\beta}\ \boldsymbol{\alpha} + \boldsymbol{\beta}]$, $z_{11} := z_{11} - z_{21}, z_{21} := z_{11} + 2z_{21}, z_{12} := z_{12} - z_{22}, z_{22} := z_{12} + 2z_{22}, z_{12} := \frac{-a_k z_{11} + z_{12}}{2}, z_{22} := \frac{-a_k z_{21} + z_{22}}{2}$
28: **else if** $\|\boldsymbol{\alpha} + \boldsymbol{\beta}\| < \|\boldsymbol{\alpha} - \boldsymbol{\beta}\| \leq \|2\boldsymbol{\alpha}\|$, **then**
29: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [\boldsymbol{\alpha} + \boldsymbol{\beta}\ \boldsymbol{\alpha} - \boldsymbol{\beta}]$, $z_{11} := z_{11} + z_{21}, z_{21} := z_{11} - 2z_{21}, z_{12} := z_{12} + z_{22}, z_{22} := z_{12} - 2z_{22}, z_{12} := \frac{-a_k z_{11} + z_{12}}{2}, z_{22} := \frac{-a_k z_{21} + z_{22}}{2}$
30: **else if** $\|2\boldsymbol{\alpha}\| \leq \|\boldsymbol{\alpha} - \boldsymbol{\beta}\| \leq \|\boldsymbol{\alpha} + \boldsymbol{\beta}\|$, **then**
31: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [2\boldsymbol{\alpha}\ \boldsymbol{\alpha} - \boldsymbol{\beta}]$, $z_{22} := z_{12} - z_{22}, z_{12} := -a_k z_{11} + z_{12}, z_{21} := z_{11} - z_{21}, z_{11} := 2z_{11}, z_{22} := \frac{-a_k z_{21} + z_{22}}{2}$
32: **else if** $\|2\boldsymbol{\alpha}\| \leq \|\boldsymbol{\alpha} + \boldsymbol{\beta}\| < \|\boldsymbol{\alpha} - \boldsymbol{\beta}\|$, **then**
33: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [2\boldsymbol{\alpha}\ \boldsymbol{\alpha} + \boldsymbol{\beta}]$, $z_{22} := z_{12} + z_{22}, z_{12} := -a_k z_{11} + z_{12}, z_{21} := z_{11} + z_{21}, z_{11} := 2z_{11}, z_{22} := \frac{-a_k z_{21} + z_{22}}{2}$
34: **else if** $\|\boldsymbol{\alpha} - \boldsymbol{\beta}\| \leq \|2\boldsymbol{\alpha}\| \leq \|\boldsymbol{\alpha} + \boldsymbol{\beta}\|, \|3\boldsymbol{\alpha} - \boldsymbol{\beta}\|$, **then**
35: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [\boldsymbol{\alpha} - \boldsymbol{\beta}\ 2\boldsymbol{\alpha}]$, $swap(z_{12}, z_{22}), z_{12} := z_{22} - z_{12}, z_{22} := -a_k z_{11} + z_{22}, swap(z_{11}, z_{21}), z_{11} := z_{21} - z_{11}, z_{21} := 2z_{21}, z_{12} := \frac{-a_k z_{11} + z_{12}}{2}$
36: **else if** $\|\boldsymbol{\alpha} - \boldsymbol{\beta}\| \leq \|3\boldsymbol{\alpha} - \boldsymbol{\beta}\| \leq \|2\boldsymbol{\alpha}_k\|, \|\boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\|_,$ **then**
37: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [\boldsymbol{\alpha} - \boldsymbol{\beta}\ 3\boldsymbol{\alpha} - \boldsymbol{\beta}]$, $swap(z_{11}, z_{21}), z_{11} := z_{21} - z_{11}, z_{21} := 2z_{21} + z_{11}, swap(z_{12}, z_{22}), z_{12} := z_{22} - z_{12}, z_{22} := 2z_{22} + z_{12}, z_{12} := \frac{-a_k z_{11} + z_{12}}{2}, z_{22} := \frac{-a_k z_{21} + z_{22}}{2}$
38: **else if** $\|\boldsymbol{\alpha} + \boldsymbol{\beta}\| \leq \|2\boldsymbol{\alpha}\| \leq \|\boldsymbol{\alpha} - \boldsymbol{\beta}\|, \|3\boldsymbol{\alpha} + \boldsymbol{\beta}\|$, **then**
39: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [\boldsymbol{\alpha} + \boldsymbol{\beta}\ 2\boldsymbol{\alpha}]$, $swap(z_{12}, z_{22}), z_{12} := z_{22} + z_{12}, z_{22} := -a_k z_{11} + z_{22}, swap(z_{11}, z_{21}), z_{11} := z_{21} + z_{11}, z_{21} := 2z_{21}, z_{12} := \frac{-a_k z_{11} + z_{12}}{2}$
40: **else**
41: $[\boldsymbol{\alpha}\ \boldsymbol{\beta}] := [\boldsymbol{\alpha} + \boldsymbol{\beta}\ 3\boldsymbol{\alpha} + \boldsymbol{\beta}]$, $swap(z_{11}, z_{21}), z_{11} := z_{21} + z_{11}, z_{21} := 2z_{21} + z_{11}, swap(z_{12}, z_{22}), z_{12} := z_{22} + z_{12}, z_{22} := 2z_{22} + z_{12}, z_{12} := \frac{-a_k z_{11} + z_{12}}{2}, z_{22} := \frac{-a_k z_{21} + z_{22}}{2}$
42: $k := k + 1$
43: **end while**
44: **if** $\boldsymbol{\alpha} = [\alpha'\ \alpha'']$ with $\alpha'' \equiv 1 \mod 2$, return $\boldsymbol{\alpha}$.
45: **else** return $\boldsymbol{\beta}$.

B. Global Euclidean Algorithm for MRFR Problem

In this section, we provide an optimization of the step 7 in Algorithm 6 and use it to design a faster algorithm for the MRFR problem. First, we note that for the ℓ_∞ norm in Algorithm 6, the μ in step 7 has only four possible values. That is,

Lemma 6: Let $\mathbf{a} = (a_1\ a_2)^T, \mathbf{b} = (b_1\ b_2)^T \in \mathbb{Z}^2$ be two lattice vectors with $\|\mathbf{b}\| > \|\mathbf{b} - \mathbf{a}\|$. For any $x \in \mathbb{R}$, define $f(x) = \|\mathbf{b} - \mathbf{a}x\| = \max(|b_1 - a_1 x|, |b_2 - a_2 x|)$. If $f(\mu) = \min_{x \in \mathbb{Z}} f(x)$, then

$$\mu \in \begin{cases} \left\{ \left\lceil \frac{|b_1|+|b_2|}{|a_1|+|a_2|} \right\rceil, \left\lfloor \frac{|b_1|+|b_2|}{|a_1|+|a_2|} \right\rfloor \right\}, & \text{if } sgn(\frac{b_1}{a_1}) = sgn(\frac{b_2}{a_2}), \\ \left\{ \left\lceil \frac{\big||b_2|-|b_1|\big|}{|a_2|+|a_1|} \right\rceil, \left\lfloor \frac{\big||b_2|-|b_1|\big|}{|a_2|+|a_1|} \right\rfloor \right\}, & \text{otherwise.} \end{cases}$$

Proof: Let $f_1(x) = |b_1 - a_1 x|$, $f_2(x) = |b_2 - a_2 x|$. Since $\|\mathbf{b}\| > \|\mathbf{b} - \mathbf{a}\|$, a_1 and a_2 are not both zeros. The analysis of the case that $a_1 = 0$ or $a_2 = 0$ is trivial. Without loss of generality, we assume $a_1 \neq 0$ and $a_2 \neq 0$. Let $x_1 = \frac{b_1}{a_1}$ and $x_2 = \frac{b_2}{a_2}$ be the zero points of $f_1(x)$ and $f_2(x)$, respectively. Now, we discuss in the following cases.

1) $|a_1| \geq |a_2|$,
   a) $|b_1| \geq |b_2|$. Since, by Lemma 5, $\mu \geq 1$, the image of the function $f(x)$ must be one of the following three cases:
      i) $x_2 > x_1 > 0$. As shown in Figure 1(a), the minimum achieves in $x = \frac{|b_1|+|b_2|}{|a_1|+|a_2|}$, which is derived from the equation $-|b_1|+|a_1|x = |b_2|-|a_2|x$.
      ii) $x_2 = x_1 > 0$. As shown in Figure 1(b), the minimum achieves in $x = \frac{|b_1|+|b_2|}{|a_1|+|a_2|}$.
      iii) $x_2 \leq 0 < x_1$. As shown in Figure 1(c), the minimum achieves in $x = \frac{|b_1|-|b_2|}{|a_1|+|a_2|} = \frac{\big||b_1|-|b_2|\big|}{|a_1|+|a_2|}$, which is derived from the equation $|b_1| - |a_1|x = |b_2| + |a_2|x$.
   b) $|b_1| < |b_2|$. Also, by Lemma 5, $\mu \geq 1$. Thus, the image of the function $f(x)$ must be one of the following two cases:
      i) $x_2 > x_1 > 0$. As shown in Figure 1(d), the minimum achieves in $x = \frac{|b_1|+|b_2|}{|a_1|+|a_2|}$, which is derived from the equation $-|b_1|+|a_1|x = |b_2|-|a_2|x$.
      ii) $x_2 > 0 \geq x_1$. As shown in Figure 1, the minimum achieves in $x = \frac{|b_2|-|b_1|}{|a_1|+|a_2|} = \frac{\big||b_1|-|b_2|\big|}{|a_1|+|a_2|}$, which is derived from the equation $|b_1| + |a_1|x = |b_2| - |a_2|x$.
2) $|a_1| < |a_2|$. By symmetry, the analysis is essentially the same as case 1. We omit here.

Since $\mu \in \mathbb{Z}$, to sum up, we have that, if $x_1 x_2 > 0$ (*i.e.*, $sgn\,(b_1/a_1) = sgn\,(b_2/a_2)$), $\mu \in \left\{ \left\lceil \frac{|b_1|+|b_2|}{|a_1|+|a_2|} \right\rceil, \left\lfloor \frac{|b_1|+|b_2|}{|a_1|+|a_2|} \right\rfloor \right\}$, and, if $x_1 x_2 \leq 0$, $\mu \in \left\{ \left\lceil \frac{\big||b_2|-|b_1|\big|}{|a_2|+|a_1|} \right\rceil, \left\lfloor \frac{\big||b_2|-|b_1|\big|}{|a_2|+|a_1|} \right\rfloor \right\}$. □

Based on Theorem 1, Theorem 6 and Lemma 6, our optimized algorithm for MRFR problem, which we called *global* Euclidean (Glo-Euc) algorithm, is shown in Algorithm 2. In our algorithm $\mathbf{a} = (S_n\ 1)^T$, $\mathbf{b} = (2^n\ 0)^T$. Therefore, by Theorem 6, the theoretical complexity of Algorithm 2 is $O(n^2 \log n \log\log n)$, which is larger than that of our improved RA algorithm. However, for a given sequence, the algorithm's practical performance outperforms the improved RA algorithm. The details are shown in Section VII.

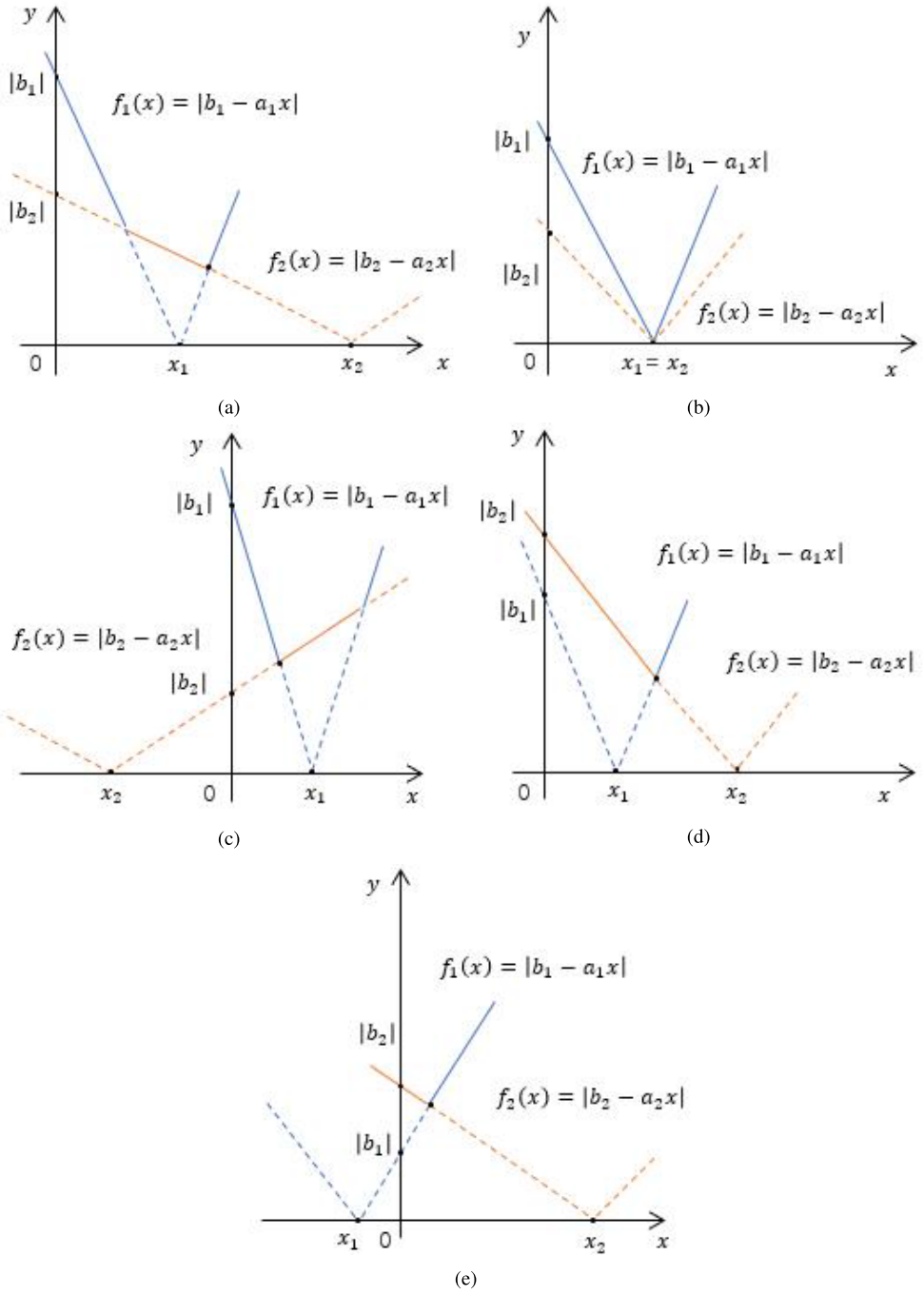


Fig. 1. Images of $f(x)$ in different cases.

VI. Partial Euclidean Algorithm

The *golbal* Euclidean algorithm reduces the length of the vectors by iteratively performing the Euclidean division on two adjacent lattice vectors, while, in this section, we will introduce *partial* Euclidean algorithm which reduces the length of the vectors by iteratively performing the Euclidean division on the first entries of two adjacent lattice vectors. In fact, as early as in 2004, Arnault *et al.* [2] had developed the same idea to compute the MRFR problem (Appendix C) and the minimal basis problem (Appendix D). However, it seems that they did not provide a thorough and correct analysis. Later, in 2008, the authors [1] revised the algorithm (Appendix E), but as indicated by the counterexample below, the new algorithm still fails to give correct output for some input instances. Also, their theoretical analysis on the minimal basis is not completely correct, a minimal basis may not be found by their algorithm in Appendix D (see Example 2).

Example 1: A counterexample for Algorithm 9. Consider the input sequence $\underline{a}(65) = (010000\ 0000100111111110001000001111111111011100000000111\ 1100001011)$, $k = 31$ and $n = 2k + 3 = 65$. In this case, the initialize values $r_0 = 3689348814741\ 9103232$, $r_1 = 30115587199846048770$, and the Algorithm 9

Algorithm 2 Glo-Euc

Input: A sequence $\underline{a}(n)$.
Output: A vector $(p\ q)$ with p/q is an MRFR for the sequence $\underline{a}(n)$
1: Compute $\mathbf{a} = (S_n\ 1)^T,\ \mathbf{b} = (2^n\ 0)^T$
2: **If** $\|\mathbf{a}\| \le \|\mathbf{a}-\mathbf{b}\|$, **then** goto *loop*
3: $\mathbf{a} := \mathbf{b} - \mathbf{a}$
4: *loop*
5: **If** $\frac{b_1}{a_1}, \frac{b_2}{a_2} \neq 0$ and $sgn\left(\frac{b_1}{a_1}\right) = sgn\left(\frac{b_2}{a_2}\right)$ **then**
6: Find $\mu \in \left\{ \left\lceil \frac{|b_1|+|b_2|}{|a_1|+|a_2|} \right\rceil, \left\lfloor \frac{|b_1|+|b_2|}{|a_1|+|a_2|} \right\rfloor \right\}$, s.t. $\|\mathbf{b}-\mu\mathbf{a}\|$ is minimal
7: **Else**
8: Find $\mu \in \left\{ \left\lceil \frac{\left||b_2|-|b_1|\right|}{|a_2|+|a_1|} \right\rceil, \left\lfloor \frac{\left||b_2|-|b_1|\right|}{|a_2|+|a_1|} \right\rfloor \right\}$, s.t. $\|\mathbf{b}-\mu\mathbf{a}\|$ is minimal
9: $\mathbf{b} := \mathbf{b} - \mu\mathbf{a}$
10: **If** $\|\mathbf{a}-\mathbf{b}\| > \|\mathbf{a}+\mathbf{b}\|$ **then** $\mathbf{b} := -\mathbf{b}$
11: Swap($\mathbf{a}, \mathbf{b}$)
12: **If** $[\mathbf{a}, \mathbf{b}]$ is Lagrange-reduced
13: **If** a_2 is odd, **then** return $\mathbf{a}$
14: **Else** return $\mathbf{b}$
15: **Else** goto *loop*

outputs $(p,q) = (r_1, v_1) = (6066854802, 597912521)$ with $\#p = 33$ and $\#q = 30$, which contradicts with the claim that the size of p and q should be no larger than $k = 31$. In fact, a MRFR of $\underline{a}(n)$ is $(p,q) = (-5986034578, 5491208247)$ with $\#p = \#q = 33$.

Example 2: A counterexample for Algorithm 8. Consider the input sequence $\underline{a}(21) = (010000000010011111111)$. In this case, the initialized values $r_0 = 2097152$ and $r_1 = 2089986$. The output of the Algorithm 8 is $[\mathbf{a}\ \mathbf{b}] = [(2194\ \ 585)^T\ \ (1902, 1463)^T]$. This is clearly incorrect. In fact, a minimal basis for this sequence is $[\mathbf{a}\ \mathbf{b}] = [(-292\ 878)^T\ (1902, 1463)^T]$.

We address these problems by first noting that the termination conditions in step 3 of Algorithm 8, step 3 and step 9 of Algorithm 9 are not precisely correct. We present a thorough and more transparent analysis which is built up on the following key observation.

Lemma 7: Let $\mathcal{L} = L([\mathbf{a}\ \mathbf{b}])$ be a 2-dimensional lattice with $\mathbf{a} = (a_1\ a_2)^T, \mathbf{b} = (b_1\ b_2)^T \in \mathbb{Z}^2$ satisfying

$$a_1 \ge b_1 > 0, a_2 b_2 \le 0, \tag{4}$$

and let $\mathcal{C} = \left\{\mathbf{v} = (v_1\ v_2)^T \in \mathcal{L}\backslash\{\mathbf{0}\} : 0 \le v_1 < b_1\right\}$. If $\mathbf{c} = (c_1\ c_2)^T = \mathbf{a} - q\mathbf{b} = \left(a_1 - \lfloor\frac{a_1}{b_1}\rfloor b_1\ \ a_2 - \lfloor\frac{a_1}{b_1}\rfloor b_2\right)^T$, then $L([\mathbf{a}\ \mathbf{b}]) = L([\mathbf{b}\ \mathbf{c}])$, and $\mathbf{c} \in \mathcal{C}$ satisfying $|c_2| = \min_{\mathbf{v}=(v_1\ v_2)^T \in \mathcal{C}} |v_2|$ and $|c_2| \ge \max\{|a_2|, |b_2|\}$.

Proof: Clearly, $\mathbf{c} \in \mathcal{C}$. Also, $[\mathbf{b}\ \mathbf{c}] = [\mathbf{a}\ \mathbf{b}]\begin{pmatrix} 0 & 1 \\ 1 & -q \end{pmatrix}$ and $\begin{pmatrix} 0 & 1 \\ 1 & -q \end{pmatrix}$ is unimodular, which immediately implies $L([\mathbf{a}\ \mathbf{b}]) = L([\mathbf{b}\ \mathbf{c}])$. Simultaneously, $a_1 \ge b_1 > 0$ and $a_2 b_2 \le 0$ clearly imply $|c_2| = |a_2 - \lfloor\frac{a_1}{b_1}\rfloor b_2| \ge \max\{|a_2|, |b_2|\}$. Hence, we just need to prove $\forall \mathbf{v} = (v_1\ v_2)^T \in \mathcal{C}, |v_2| \ge |c_2|$. Since $\exists z_1, z_2 \in \mathbb{Z}$, such that $\mathbf{v} = z_1\mathbf{a} + z_2\mathbf{b} = (z_1 a_1 + z_2 b_1\ \ z_1 a_2 + z_2 b_2)^T \neq (0\ 0)^T$, we have $v_1 = z_1 a_1 + z_2 b_1$, $v_2 = z_1 a_2 + z_2 b_2$ and $z_1 \neq 0$. Now we argue the result as follows.

Case 1: if $z_1 < 0 \wedge z_2 \le 0$, then $v_1 = z_1 a_1 + z_2 b_1 < 0$, and thus, $\mathbf{v} \notin \mathcal{C}$.

Case 2: if $z_1 < 0 \wedge z_2 > 0$, then

$$0 \le v_1 = z_1 a_1 + z_2 b_1 < b_1 \iff 0 \le z_1\frac{a_1}{b_1} + z_2 < 1$$
$$\implies -z_2 > z_1\frac{a_1}{b_1} - 1 \implies -z_2 \ge z_1\lfloor\frac{a_1}{b_1}\rfloor$$
$$\implies |v_2| = |z_1 a_2 + z_2 b_2| = |z_1||a_2| + |z_2||b_2|$$
$$\ge (-z_1)\left(|a_2| + \lfloor\frac{a_1}{b_1}\rfloor|b_2|\right) \ge \left|a_2 - \lfloor\frac{a_1}{b_1}\rfloor b_2\right| = |c_2|.$$

Case 3: if $z_1 > 0 \wedge z_2 \le 0$, then

$$0 \le v_1 = z_1 a_1 + z_2 b_1 < b_1 \iff 0 \le z_1\frac{a_1}{b_1} + z_2 < 1$$
$$\implies -z_2 > z_1\frac{a_1}{b_1} - 1 \implies -z_2 \ge z_1\lfloor\frac{a_1}{b_1}\rfloor$$
$$\implies |v_2| = |z_1 a_2 + z_2 b_2| = |z_1||a_2| + |z_2||b_2|$$
$$\ge z_1\left(|a_2| + \lfloor\frac{a_1}{b_1}\rfloor|b_2|\right) \ge \left|a_2 - \lfloor\frac{a_1}{b_1}\rfloor b_2\right| = |c_2|.$$

Case 4: $z_1 > 0 \wedge z_2 > 0$. $v_1 = z_1 a_1 + z_2 b_1 > b_1$, and thus, $\mathbf{v} \notin \mathcal{C}$. □

With the assistance of Lemma 7, we deduce the precise decisional condition for a Lagrange reduced basis.

Theorem 7: Let $\mathcal{L} = L([\mathbf{a}\ \mathbf{b}])$ be a lattice generated by the vectors $\mathbf{a} = (a_1\ a_2)^T, \mathbf{b} = (b_1\ b_2)^T \in \mathbb{Z}^2$ satisfying $a_1 \ge b_1 > 0, a_2 b_2 \le 0$ and $|b_2| < b_1$. If we iteratively reduce the length of lattice vectors as follows

$$\mathbf{v}^{(0)} = (v_1^{(0)}\ v_2^{(0)}) = \mathbf{a}, \mathbf{v}^{(1)} = (v_1^{(1)}\ v_2^{(1)}) = \mathbf{b} \rightarrow$$
$$\mathbf{v}^{(2)} = (v_1^{(2)}\ v_2^{(2)})^T = \mathbf{v}^{(0)} - q_0\mathbf{v}^{(1)} \rightarrow \cdots \rightarrow$$
$$\mathbf{v}^{(k)} = (v_1^{(k)}\ v_2^{(k)})^T = \mathbf{v}^{(k-2)} - q_{k-2}\mathbf{v}^{(k-1)} \rightarrow$$
$$\mathbf{v}^{(k+1)} = (v_1^{(k+1)}\ v_2^{(k+1)})^T = \mathbf{v}^{(k-1)} - q_{k-1}\mathbf{v}^{(k)} \rightarrow \cdots \rightarrow$$
$$\mathbf{v}^{(n)} = (v_1^{(n)}\ v_2^{(n)})^T = \mathbf{v}^{(n-2)} - q_{n-2}\mathbf{v}^{(n-1)} = (0\ v_2^{(n)}),$$

where $q_i = \left\lfloor\frac{v_1^{(i)}}{v_1^{(i+1)}}\right\rfloor$, then there exists a unique integer $k \in [1, n]$ such that $v_1^{(k)} > |v_2^{(k)}|$, $v_1^{(k+1)} \le |v_2^{(k+1)}|$, and $\mathcal{L} = L([\mathbf{a}\ \mathbf{b}]) = L([\mathbf{v}^{(k)}\ \mathbf{v}^{(k+1)}])$. Further,

(1) if $v_1^{(k)} \le |v_2^{(k+1)}|$, we have $\lambda_1(\mathcal{L}) = \|\mathbf{v}^{(k)}\|$, $\lambda_2(\mathcal{L}) = \min\left\{\left\|\mathbf{v}^{(k+1)} - \lfloor x\rfloor\mathbf{v}^{(k)}\right\|, \left\|\mathbf{v}^{(k+1)} - \lceil x\rceil\mathbf{v}^{(k)}\right\|\right\}$ for $x = -\frac{|v_2^{(k+1)}| - v_1^{(k+1)}}{|v_2^{(k)}| + v_1^{(k)}}$.

(2) if $v_1^{(k)} > |v_2^{(k+1)}|$, we have $\lambda_1(\mathcal{L}) = \|\mathbf{v}^{(k+1)}\|$, $\lambda_2(\mathcal{L}) = \min\left\{\left\|\mathbf{v}^{(k)} - \lfloor y\rfloor\mathbf{v}^{(k+1)}\right\|, \left\|\mathbf{v}^{(k)} - \lceil y\rceil\mathbf{v}^{(k+1)}\right\|\right\}$ for $y = -\frac{|v_1^{(k)}| - v_2^{(k)}}{|v_1^{(k+1)}| + v_2^{(k+1)}}$.

Proof: By the property of Euclidean division and Lemma 7, we have

$$b_1 = v_1^{(1)} > v_1^{(2)} > \cdots > v_1^{(k)} > v_1^{(k+1)} > \cdots > v_1^{(n)} = 0,$$
$$0 \le |b_2| \le |v_2^{(2)}| \le \cdots \le |v_2^{(k)}| \le |v_2^{(k+1)}| \le \cdots \le |v_2^{(n)}|.$$

Since $|b_2| < b_1$, there exists a unique integer $k \in [1, n]$ such that

$$v_1^{(k)} > |v_2^{(k)}|, v_1^{(k+1)} \le |v_2^{(k+1)}|. \tag{5}$$

Clearly, by Lemma 7, $\mathcal{L} = L([\mathbf{a}\ \mathbf{b}]) = L([\mathbf{v}^{(0)}\ \mathbf{v}^{(1)}]) = L([\mathbf{v}^{(1)}\ \mathbf{v}^{(2)}]) = \cdots = L([\mathbf{v}^{(k)}\ \mathbf{v}^{(k+1)}])$.

Now, we estimate the length of $\lambda_1(\mathcal{L})$. $\forall \mathbf{u} = (u_1\ u_2) \in \mathcal{L}\backslash\{\mathbf{0}\}$, without loss of generality, we assume $u_1 \ge 0$. If $u_1 < v_1^{(k)}$, then, by Lemma 7, we have $|v_2^{(k+1)}| \le |u_2|$, and thus $\|\mathbf{u}\| = \max\{|u_1|, |u_2|\} \ge |u_2| \ge |v_2^{(k+1)}| = \max\left\{\left|v_1^{(k+1)}\right|, \left|v_2^{(k+1)}\right|\right\} = \|\mathbf{v}^{(k+1)}\|$. If $u_1 \ge v_1^{(k)}$, then

$\|\mathbf{u}\| = \max\{|u_1|, |u_2|\} \geq u_1 \geq v_1^{(k)} = \max\{|v_1^{(k)}|, |v_2^{(k)}|\} = \|\mathbf{v}^{(k)}\|$. That is, $\|\mathbf{u}\| \geq \min\left\{\|\mathbf{v}^{(k)}\|, \|\mathbf{v}^{(k+1)}\|\right\}$. Therefore, $\lambda_1(\mathcal{L}) = \min\{\|\mathbf{v}^{(k)}\|, \|\mathbf{v}^{(k+1)}\|\}$.

Next, we estimate the length of $\lambda_2(\mathcal{L})$.

- If $\|\mathbf{v}^{(k)}\| \leq \|\mathbf{v}^{(k+1)}\|$, then, by equation (5), $|v_2^{(k)}| < v_1^{(k)} \leq |v_2^{(k+1)}|$. Define

$$\begin{aligned} f(x) &= \left\|\mathbf{v}^{(k+1)} - x\mathbf{v}^{(k)}\right\| \\ &= \max\left\{\left|v_1^{(k+1)} - xv_1^{(k)}\right|, \left|v_2^{(k+1)} - xv_2^{(k)}\right|\right\}, \\ f_1(x) &= \left|v_1^{(k+1)} - xv_1^{(k)}\right|, f_2(x) = \left|v_2^{(k+1)} - xv_2^{(k)}\right|. \end{aligned}$$

First, we prove $\lambda_2(\mathcal{L}) = f(z) = \min_{x\in\mathbb{Z}} f(x)$. In fact, by the minimality of $f(z)$ and $\lambda_1(\mathcal{L}) = \|\mathbf{v}^{(k)}\|$, we have $\|\mathbf{v}^{(k)}\| \leq \|\mathbf{v}^{(k+1)} - z\mathbf{v}^{(k)}\| \leq \|\mathbf{v}^{(k+1)} - (z+1)\mathbf{v}^{(k)}\|, \|\mathbf{v}^{(k+1)} - (z-1)\mathbf{v}^{(k)}\|$. That is, by Definition 3 and Lemma 3, $[\mathbf{v}^k \ \mathbf{v}^{(k+1)} - z\mathbf{v}^{(k)}]$ is Largerage-reduced and $\lambda_2(\mathcal{L}) = f(z)$. Next, we estimate the value of z. By Lemma 7, $v_2^{(k+1)}v_2^{(k)} \leq 0$, $|v_2^{(k)}| \leq |v_2^{(k+1)}|$ and $0 \leq v_1^{(k+1)} < v_1^{(k)}$. Thus, for $x \geq 0$, we have $f(x) \geq f_2(x) = |v_2^{(k+1)}| + x|v_2^{(k)}| \geq |v_2^{(k+1)}| = \|\mathbf{v}^{(k+1)}\| = f(0)$. For $x < 0$, we have $f_1(x) = -v_1^{(k)}x + v_1^{(k+1)}$, and

$$f_2(x) = \begin{cases} -|v_2^{(k)}|x - |v_2^{(k+1)}|, & \text{if } x \leq -\frac{|v_2^{(k+1)}|}{|v_2^{(k)}|}, \\ |v_2^{(k)}|x + |v_2^{(k+1)}|, & \text{if } x > -\frac{|v_2^{(k+1)}|}{|v_2^{(k)}|}. \end{cases}$$

Let $-v_1^{(k)}x + v_1^{(k+1)} = |v_2^{(k)}|x + |v_2^{(k+1)}|$, we have $x = -\frac{|v_2^{(k+1)}| - v_1^{(k+1)}}{|v_2^{(k)}| + v_1^{(k)}}$, which is greater than $-\frac{|v_2^{(k+1)}|}{|v_2^{(k)}|}$. Thus, when $x < 0$, the image of function $f(x)$ is shown in Figure 2(a). As we can observe from the figure, $f(x)$ achieves the minimum in $x = -\frac{|v_2^{(k+1)}| - v_1^{(k+1)}}{|v_2^{(k)}| + v_1^{(k)}}$, and the minimum is less than $f(0)$. Therefore,

$$z = \left\lfloor -\frac{|v_2^{(k+1)}| - v_1^{(k+1)}}{|v_2^{(k)}| + v_1^{(k)}} \right\rfloor \text{ or } z = \left\lceil -\frac{|v_2^{(k+1)}| - v_1^{(k+1)}}{|v_2^{(k)}| + v_1^{(k)}} \right\rceil.$$

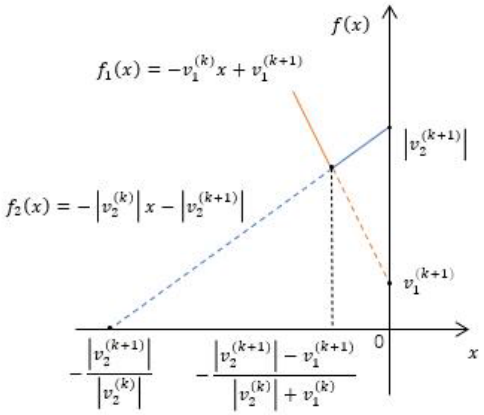
- If $\|\mathbf{v}^{(k)}\| > \|\mathbf{v}^{(k+1)}\|$, then, by equation (5), $v_1^{(k)} > |v_2^{(k+1)}| \geq v_1^{(k+1)}$. Define

$$\begin{aligned} g(y) &= \left\|\mathbf{v}^{(k)} - y\mathbf{v}^{(k+1)}\right\| \\ &= \max\left\{\left|v_1^{(k)} - yv_1^{(k+1)}\right|, \left|v_2^{(k)} - yv_2^{(k+1)}\right|\right\}, \\ g_1(y) &= \left|v_1^{(k)} - yv_1^{(k+1)}\right|, g_2(y) = \left|v_2^{(k)} - yv_2^{(k+1)}\right|. \end{aligned}$$

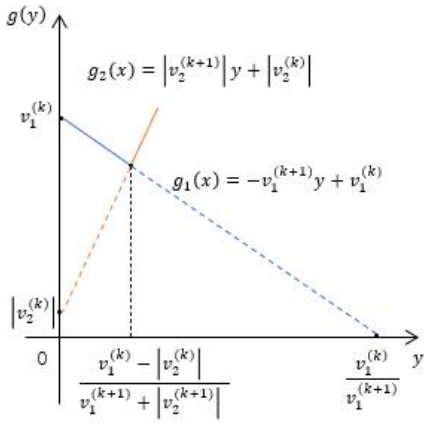
With essentially the same analysis as that for $f(x)$, we can prove that $\lambda_2(\mathcal{L}) = g(z) = \min_{y\in\mathbb{Z}} g(y)$ and the image of function $g(y)$ is shown in Figure 2. As we can observe from the figure, $g(y)$ achieves the minimum in $y = \frac{v_1^{(k)} - |v_2^{(k)}|}{v_1^{(k+1)} + |v_2^{(k+1)}|}$, and the minimum is less than $g(0)$. Therefore,

$$z = \left\lfloor \frac{v_1^{(k)} - |v_2^{(k)}|}{v_1^{(k+1)} + |v_2^{(k+1)}|} \right\rfloor \text{ or } z = \left\lceil \frac{v_1^{(k)} - |v_2^{(k)}|}{v_1^{(k+1)} + |v_2^{(k+1)}|} \right\rceil.$$

□



(a)



(b)

Fig. 2. Images of $f(x)$ and $g(y)$.

Clearly, Theorem 7 can be directly turned into the Algorithm 3. However, our theoretical analysis indicates this algorithm can be further sped up by invoking Möller's HGCD algorithm [11]. Here we include the main statement for the HGCD algorithm.

Lemma 8 ([11]): On inputting two positive integers a, b with $\#(a, b) = \ell$, the HGCD algorithm (Appendix F) outputs two relatively small positive integers c, d with $\underline{\#}(c, d) > \lfloor\frac{\ell}{2}\rfloor + 1$, $\#(c - d) \leq \lfloor\frac{\ell}{2}\rfloor + 1$, and an unimodular matrix $\mathbf{M}$ with non-negative entries such that $(a \ b)^T = \mathbf{M}(c \ d)^T$ in time $O(\ell \log^2 \ell \log\log \ell)$.

In Lemma 8, corresponding to our settings, $a = 2^n$, $b = S_n$, $\ell = n+1$, and $c, d > 2^{\lfloor\frac{n+1}{2}\rfloor}$. Now, the problem is whether the output of Möller's HGCD algorithm satisfies the conditions of Theorem 7. If so, we can perform the *partial* Euclidean algorithm after the HGCD operations. The following lemma answers the above question in the affirmative.

Lemma 9: Let $\mathcal{L} = L([\mathbf{a} \ \mathbf{b}])$ be a lattice generated by $\mathbf{a} = (2^n \ 0)^T, \mathbf{b} = (S_n \ 1)^T \in \mathbb{Z}^2$. If $[\mathbf{c} \ \mathbf{d}]$ is a new lattice basis of $\mathcal{L}$ with $\mathbf{c} = (c_1 \ c_2)^T$, $\mathbf{d} = (d_1 \ d_2)^T$, $c_1 \geq d_1 > 2^{\lfloor\frac{n+1}{2}\rfloor}$ and $c_2 d_2 \leq 0$. Then,

1) $|d_2| < d_1$;
2) reduce the length of lattice vectors beginning with $[\mathbf{c} \ \mathbf{d}]$ by the iterative procedure as in Theorem 7, namely,

$$\mathbf{v}^{(0)} = (v_1^{(0)} \ v_2^{(0)}) = \mathbf{c}, \mathbf{v}^{(1)} = (v_1^{(1)} \ v_2^{(1)}) = \mathbf{d} \rightarrow$$

Algorithm 3 Par-Euc

Input: A sequence $\underline{a}(n) = (a_0 \ \cdots a_{n-1})$
Output: Two integers p, q such that $\frac{p}{q}$ is a MRFR of $\underline{a}(n)$
1: $a := 2^n, b := \sum_{i=0}^{n-1} a_i 2^i$
2: $(v_1^{(0)} \ v_2^{(0)}) := (a \ 0)$
3: $(v_1^{(1)} \ v_2^{(1)}) := (b \ 1)$
4: $i := 1$
5: **while** $v_1^{(i)} > |v_2^{(i)}|$ **do**
6: $q = \left\lfloor \frac{v_1^{(i-1)}}{v_1^{(i)}} \right\rfloor$, $r = v_1^{(i-1)} - q v_1^{(i)}$
7: $v_1^{(i+1)} = r$, $v_2^{(i+1)} = v_2^{(i-1)} - q v_2^{(i)}$
8: $i = i + 1$
9: **end while**
10: **if** $v_1^{(i-1)} \le |v_2^{(i)}|$
11: **if** $v_2^{(i-1)}$ is odd, **then** return $\mathbf{v}^{(i-1)} = (v_1^{(i-1)} \ v_2^{(i-1)})$
12: **else**
13: **if**

$$\left\| \mathbf{v}^{(i)} - \left\lfloor -\frac{|v_2^{(i)}| - v_1^{(i)}}{|v_2^{(i-1)}| + v_1^{(i-1)}} \right\rfloor \mathbf{v}^{(i-1)} \right\| \le \left\| \mathbf{v}^{(i)} - \left\lceil -\frac{|v_2^{(i)}| - v_1^{(i)}}{|v_2^{(i-1)}| + v_1^{(i-1)}} \right\rceil \mathbf{v}^{(i-1)} \right\|,$$

return $\mathbf{v}^{(i)} - \left\lfloor -\frac{|v_2^{(i)}| - v_1^{(i)}}{|v_2^{(i-1)}| + v_1^{(i-1)}} \right\rfloor \mathbf{v}^{(i-1)}$.
14: **else** return $\mathbf{v}^{(i)} - \left\lceil -\frac{|v_2^{(i)}| - v_1^{(i)}}{|v_2^{(i-1)}| + v_1^{(i-1)}} \right\rceil \mathbf{v}^{(i-1)}$.
15: **else**
16: **if** $v_2^{(i)}$ is odd, return $\mathbf{v}^{(i)} = (v_1^{(i)} \ v_2^{(i)})$
17: **else**
18: **if**

$$\left\| \mathbf{v}^{(i-1)} - \left\lfloor \frac{v_1^{(i-1)} - |v_2^{(i-1)}|}{v_1^{(i)} + |v_2^{(i)}|} \right\rfloor \mathbf{v}^{(i)} \right\| \le \left\| \mathbf{v}^{(i-1)} - \left\lceil \frac{v_1^{(i-1)} - |v_2^{(i-1)}|}{v_1^{(i)} + |v_2^{(i)}|} \right\rceil \mathbf{v}^{(i)} \right\|,$$

return $\mathbf{v}^{(i-1)} - \lfloor \frac{v_1^{(i-1)} - |v_2^{(i-1)}|}{v_1^{(i)} + |v_2^{(i)}|} \rfloor \mathbf{v}^{(i)}$.
19: **else** return $\mathbf{v}^{(i-1)} - \left\lceil \frac{v_1^{(i-1)} - |v_2^{(i-1)}|}{v_1^{(i)} + |v_2^{(i)}|} \right\rceil \mathbf{v}^{(i)}$.

$$\mathbf{v}^{(2)} = (v_1^{(2)} \ v_2^{(2)})^T = \mathbf{v}^{(0)} - q_0 \mathbf{v}^{(1)} \to \cdots \to$$
$$\mathbf{v}^{(k)} = (v_1^{(k)} \ v_2^{(k)})^T = \mathbf{v}^{(k-2)} - q_{k-2} \mathbf{v}^{(k-1)} \to$$
$$\mathbf{v}^{(k+1)} = (v_1^{(k+1)} \ v_2^{(k+1)})^T = \mathbf{v}^{(k-1)} - q_{k-1} \mathbf{v}^{(k)},$$

where $q_i = \left\lfloor \frac{v_1^{(i)}}{v_1^{(i+1)}} \right\rfloor$, $v_1^{(k)} > |v_2^{(k)}|, v_1^{(k+1)} \le |v_2^{(k+1)}|$,

the following hold

$$v_1^{(k-1)} > 2^{\frac{n-1}{2}}, v_1^{(k+1)} < 2^{\frac{n}{2}} < v_1^{(k-2)}. \tag{6}$$

Proof: Since $L([\mathbf{a} \ \mathbf{b}]) = L([\mathbf{c} \ \mathbf{d}])$, we have $|\det([\mathbf{c} \ \mathbf{d}])| = |c_1 d_2 - c_2 d_1| = |\det([\mathbf{a} \ \mathbf{b}])| = 2^n$. While, $c_1 \ge d_1 > 2^{\lfloor \frac{n+1}{2} \rfloor}$ and $c_2 d_2 \le 0$. Thus, $|c_1 d_2 - c_2 d_1| = |c_1||d_2| + |c_2||d_1| = 2^n$ implies $|c_2|, |d_2| \le 2^{\frac{n}{2}} < d_1$, which is consistent with the conditions in Theorem 7. Now, we prove the equation (6). Since $q_{k-1} = \left\lfloor \frac{v_1^{(k-1)}}{v_1^{(k)}} \right\rfloor$, $v_2^{(k+1)} = -q_{k-1} v_2^{(k)} + v_2^{(k-1)}$ and, by Theorem 7, $v_1^{(k-1)} > v_1^{(k)} > |v_2^{(k)}| \ge |v_2^{(k-1)}|$, we have $v_1^{(k+1)} \le \left|v_2^{(k+1)}\right| = |q_{k-1}| \left|v_2^{(k)}\right| + \left|v_2^{(k-1)}\right| < v_1^{(k-1)} + v_1^{(k)}$. Hence

$$\begin{aligned}(v_1^{(k+1)})^2 &< v_1^{(k)} \left|v_2^{(k+1)}\right| + \left|v_2^{(k)}\right| v_1^{(k+1)} \\ &< v_1^{(k)}(v_1^{(k)} + v_1^{(k-1)}) + v_1^{(k)} v_1^{(k+1)} \\ &= v_1^{(k)}(v_1^{(k)} + v_1^{(k-1)} + v_1^{(k+1)}) \le v_1^{(k)} \cdot 2 v_1^{(k-1)} \\ &< \min\{2(v_1^{(k-1)})^2, (v_1^{(k-2)})^2\}. \end{aligned} \tag{7}$$

Also, by Theorem 7,

$$v_1^{(k)} \cdot \left|v_2^{(k+1)}\right| + \left|v_2^{(k)}\right| \cdot v_1^{(k+1)} = 2^n. \tag{8}$$

Combining equation (7) with equation (8), we have $(v_1^{(k+1)})^2 < 2^n < \min\{2(v_1^{(k-1)})^2, (v_1^{(k-2)})^2\}$. That is, $v_1^{(k-1)} > 2^{\frac{n-1}{2}}$ and $v_1^{(k+1)} < 2^{\frac{n}{2}} < v_1^{(k-2)}$. □

Based on the above analysis, we summarize the main results in this section as follows.

Theorem 8: On inputting an n-length sequence $\underline{a}(n)$, the *partial* Euclidean algorithm (Algorithm 3) and the **Hgcd-Par-Euc** algorithm (Algorithm 4) output a MRFR of $\underline{a}(n)$ in time $O(n^2)$ and $O(n \log^2 n \log\log n)$, respectively.

Proof: The correctness of the *partial* Euclidean algorithm **Par-Euc** is immediately obtained from Theorem 1 and Theorem 7, and the correctness of the algorithm **HGCD-Par-Euc** can also be obtained by Lemma 8 and Lemma 9. Now, we estimate their time complexity.

Firstly, the complexity of Algorithm 3 is mainly from step 5 to step 9. For the ith iteration, let n_i and l_i denote the bit length of $v_1^{(i)}$ and the bit length of q, respectively. Then, $n_{i-1} = n_i + l_i$, the complexity of the division is bounded by $O(n_{i-1} l_i)$, the complexity of the first multiplication is bounded by $O(n_i l_i)$, and the complexity of the second multiplication is bounded by $O(n_i l_i)$. Thus, the complexity of step 5-step 9 is $O(\sum_{i=1}^{k}(n_i l_i + n_{i-1} l_i)) = O(\sum_{i=1}^{k}(n_i + n_{i-1})(n_{i-1} - n_i)) = O(\sum_{i=1}^{k}(n_{i-1}^2 - n_i^2)) = O(n^2)$. Here, k stands for the number of iterations. Therefore the total complexity of Algorithm 3 is $O(n^2)$.

Now, we analyze the asymptotic complexity of Algorithm 4. By Lemma 8, the time complexity of step 2 is $O(n \log^2 \log\log n)$. Hence, we only need to estimate the complexity of step 6-step 10. Assuming the number of iterations in step 6-step 10 is k. Now, we prove $k \le 5$. In fact, by Lemma 8, at the beginning of the first iteration of step 6, $v_1^{(0)} \ge v_1^{(1)} > 2^{\lfloor \frac{n+1}{2} \rfloor} \ge 2^{\frac{n}{2}}$ and $v_1^{(0)} - v_1^{(1)} \le 2 \cdot 2^{\frac{n+1}{2}}$, and, at the end of the last iteration, $i = k+1$, $v_1^{(k)} > |v_2^{(k)}|$ and $v_1^{(k+1)} \le |v_2^{(k+1)}|$. By Lemma 9, $|v_1^{(k-1)}| > 2^{\frac{n-1}{2}}$ and $v_1^{(k+1)} < 2^{\frac{n}{2}}$. Meanwhile, by the property of Euclidean division, $v_1^{(6)} < \frac{1}{2} v_1^{(4)} < \frac{1}{2^2} v_1^{(2)} \le \frac{1}{2^2}(v_1^{(0)} - v_1^{(1)}) \le 2^{\frac{n-1}{2}}$. Therefore, $k \le 5$. This indicates that, after the HGCD operation in step 2, the algorithm will terminate by performing at most 5 iterations. Consequently, the total time complexity is essentially the same as that of HGCD algorithm, which is $O(n \log^2 n \log\log n)$. □

VII. Practical Performance Evaluation

In this section, we compare the practical performance of our improved algorithms with the existing algorithms. We perform our experiments on a Ubuntu 16.04 machine with Intel Xeon Skylake 6146, 3.2 GHz CPU and 4GB RAM, and implement the algorithms with C language. In our experiments, the length n of the sequences is in the range from 10000 to 1300000. For a fixed n, we test 10 different random sequences and

Algorithm 4 Hgcd-Par-Euc

Input: A sequence $\underline{a}(n) = (a_0 \ \cdots a_{n-1})$
Output: Two integers p, q such that $\frac{p}{q}$ is a MRFR of $\underline{a}(n)$
$a := 2^n, b := \sum_{i=0}^{n-1} a_i 2^i$
$(c, d, \mathbf{M}) \leftarrow \mathrm{HGCD}(a, b)$
if $c > d$, **then** $(v_1^{(0)} \ v_1^{(1)}) := (c, d)$, $(v_2^{(0)} \ v_2^{(1)}) := (0 \ 1)\mathbf{M}^{-T}$
else $(v_1^{(0)} \ v_1^{(1)}) := (d, c)$, $(v_2^{(1)} \ v_2^{(0)}) := (0 \ 1)\mathbf{M}^{-T}$
$i := 1$
while $v_1^{(i)} > |v_2^{(i)}|$, **do**
  $q = \lfloor \frac{v_1^{(i-1)}}{v_1^{(i)}} \rfloor$, $r = v_1^{(i-1)} - q v_1^{(i)}$
  $v_1^{(i+1)} = r$, $v_2^{(i+1)} = -q v_2^{(i)} + v_2^{(i-1)}$
  $i = i + 1$
end while
if $v_1^{(i-1)} \le |v_2^{(i)}|$
  if $v_2^{(i-1)}$ is odd, **then** return $\mathbf{v}^{(i-1)} = (v_1^{(i-1)} \ v_2^{(i-1)})$
  else
    if

$$\left\| \mathbf{v}^{(i)} - \left\lfloor \frac{v_2^{(i)} - \left| v_1^{(i)} \right|}{v_2^{(i-1)} + |v_1^{(i-1)}|} \right\rfloor \mathbf{v}^{(i-1)} \right\| \le \left\| \mathbf{v}^{(i)} - \left\lceil \frac{v_2^{(i)} - \left| v_1^{(i)} \right|}{v_2^{(i-1)} + \left| v_1^{(i-1)} \right|} \right\rceil \mathbf{v}^{(i-1)} \right\|,$$

return $\mathbf{v}^{(i)} - \left\lfloor \frac{v_2^{(i)} - \left| v_1^{(i)} \right|}{v_2^{(i-1)} + \left| v_1^{(i-1)} \right|} \right\rfloor \mathbf{v}^{(i-1)}$.

    else return $\mathbf{v}^{(i)} - \left\lceil \frac{v_2^{(i)} - |v_1^{(i)}|}{v_2^{(i-1)} + |v_1^{(i-1)}|} \right\rceil \mathbf{v}^{(i-1)}$.
  else
    if $v_2^{(i)}$ is odd, return $\mathbf{v}^{(i)} = (v_1^{(i)} \ v_2^{(i)})$
    else
      if

$$\left\| \mathbf{v}^{(i-1)} - \left\lfloor -\frac{\left| v_1^{(i-1)} \right| - v_2^{(i-1)}}{\left| v_1^{(i)} \right| + v_2^{(i)}} \right\rfloor \mathbf{v}^{(i-1)} \right\| \le \left\| \mathbf{v}^{(i-1)} - \left\lceil -\frac{\left| v_1^{(i-1)} \right| - v_2^{(i-1)}}{\left| v_1^{(i)} \right| + v_2^{(i)}} \right\rceil \mathbf{v}^{(i)} \right\|,$$

return $\mathbf{v}^{(i-1)} - \left\lfloor -\frac{|v_1^{(i-1)}| - v_2^{(i-1)}}{|v_1^{(i)}| + v_2^{(i)}} \right\rfloor \mathbf{v}^{(i)}$.

      else return $\mathbf{v}^{(i-1)} - \left\lceil -\frac{\left| v_1^{(i-1)} \right| - v_2^{(i-1)}}{\left| v_1^{(i)} \right| + v_2^{(i)}} \right\rceil \mathbf{v}^{(i)}$.

take the average runtime as the final time overhead. Table I lists the time overheads of Klapper and Goresky's rational approximation algorithm **K-G RA**, Li *et al.*algorithm **LYLQ RA**, and our improved algorithms **Hgcd-Par-Euc**,**Par-Euc**, **Gol-Euc** and **Improved-RA**.

As we can observe from the table, the practical performance of the non-adaptive algorithms is superior to that of adaptive algorithms. In terms of adaptive algorithms, our improved algorithm **Improved-RA** is the fastest and the time cost is less than 73% of that of Li *et al.* algorithm **LYLQ RA**, and, as the bitsize of the input sequence increases, we can achieve more computational savings. For non-adaptive algorithms, the *global* Euclidean algorithm **Gol-Euc** is the slowest, and, only when the bitsize of the input sequence is larger than about 90000, the hgcd-based partial Euclidean algorithm **Hgcd-Par-Euc** can beat the *partial* Euclidean algorithm **Par-Euc**.

VIII. Conclusion

In this paper, we discussed algorithms for the MRFR problem of sequences and obtained results that improve the currently best adaptive and non-adaptive algorithms both in theory and practice. The decision of whether to use adaptive algorithm **Improved-RA** or non-adaptive algorithm **Hgcd-Par-Euc** can be easily determined by inspecting different application scenarios. Furthermore, we would like to point out that an $O(n^2)$-time algorithm for finding the minimal basis of any 2-dimensional lattice generated by two vectors $\mathbf{a} = (a_1 \ a_2)^T, \mathbf{b} = (b_1 \ b_2)^T \in \mathbb{Z}^2$ satisfying $a_1 \ge b_1 > 0, a_2 b_2 \le 0$ and $|b_2| < b_1$ is readily produced by Theorem 7. Lemma 8 and Lemma 9 imply an $O(n \log^2 n \log\log n)$-time algorithm for finding the minimal basis of any 2-dimensional lattice $\mathcal{L} = L([\mathbf{a} \ \mathbf{b}])$ with $\mathbf{a} = (a_1 \ 0)^T, \mathbf{b} = (b_1 \ 1)^T \in \mathbb{Z}^2$. These minimal basis algorithms for 2-dimensional lattices might be of independent interest in lattice-based cryptography.

TABLE I
Time Overhead of Different Algorithms (Seconds)

length n	Hgcd-Par-Euc	Par-Euc	Gol-Euc	Improved-RA	LYLQ RA	K-G RA
10000	0.003455	0.000888	0.002043	0.005391	0.007159	0.022849
30000	0.010032	0.004797	0.013479	0.02485	0.036265	0.307885
50000	0.016576	0.01168	0.034934	0.060148	0.090597	1.06886
70000	0.024162	0.023998	0.070184	0.113492	0.173212	2.413481
90000	0.030820	0.038072	0.123214	0.185118	0.28458	4.433946
100000	0.034017	0.045361	0.137741	0.234118	0.309931	5.752553
300000	0.108437	0.426801	1.264407	2.116471	2.870553	80.397822
500000	0.182584	1.163474	3.560321	5.871737	7.992801	270.357691
700000	0.262421	2.269855	6.922843	11.525406	15.739535	558.298801
900000	0.339741	3.779008	11.468116	19.143314	26.085824	971.005226
1100000	0.421889	5.548559	17.418869	28.80822	39.160171	1498.387269
1300000	0.510131	7.756622	24.567148	40.604187	55.252542	2149.024857

Appendix A
Klapper and Goresky's Rational Approximation Algorithm **K-G RA**

Algorithm 5 K-G RA [5]

Input: A sequence $\underline{a}(n) = (a_0 \ a_1 \ \cdots \ a_{n-1})$.
Output: An MRFR of $\underline{a}(n)$.
input $a_i's$ until the first nonzero a_{k-1} is found
$S = a_{k-1} \cdot 2^{k-1}$, $\mathbf{f} = (0 \ 2)$, $\mathbf{g} = (2^{k-1} \ 1)$
while there are more bits, **do**
  input a new bit a_k
  $S := S + a_k 2^k$
  if $S \cdot g_2 - g_1 \equiv 0 \bmod 2^{k+1}$, **then**
    $\mathbf{f} := 2\mathbf{f}$
  else if $\|\mathbf{g}\| \le \|\mathbf{f}\|$, **then**
    Compute the odd integer d such that $\|\mathbf{f} + d\mathbf{g}\|$ is minimal
    $[\mathbf{g} \ \mathbf{f}] := [\mathbf{f} + d\mathbf{g} \ 2\mathbf{g}]$
    else
    Compute the odd integer d such that $\|\mathbf{g} + d\mathbf{f}\|$ is minimal
    $[\mathbf{g} \ \mathbf{f}] := [\mathbf{g} + d\mathbf{f} \ 2\mathbf{f}]$
    end if
  end if
  $k := k + 1$
end while
return $\mathbf{g} = (g_1 \ g_2)$

Appendix B
Lagrange's Reduction Algorithm **Lag-red**

See Algorithm 6.

Appendix C
Arnault et al.'s MRFR Algorithm **EEAapprox**

See Algorithm 7.

Algorithm 6 Lag-Red [10]

Input: A basis $[\mathbf{a}\ \mathbf{b}]$.
Output: A shortest basis for $\mathcal{L}([\mathbf{a}\ \mathbf{b}])$
If $\|\mathbf{a}\| > \|\mathbf{b}\|$, **then** swap $(\mathbf{a}, \mathbf{b})$
If $\|\mathbf{a} - \mathbf{b}\| > \|\mathbf{a} + \mathbf{b}\|$, **then** return $[\mathbf{a}\ \mathbf{b}]$
If $\|\mathbf{a}\| \leq \|\mathbf{a} - \mathbf{b}\|$, **then** goto *loop*
If $\|\mathbf{a}\| = \|\mathbf{b}\|$, **then** return $[\mathbf{a} - \mathbf{b}, \mathbf{a}]$
$\mathbf{a} := \mathbf{b} - \mathbf{a}$
loop
  Find $\mu \in \mathbb{Z}$ s.t. $\|\mathbf{b} - \mu\mathbf{a}\|$ is minimal
  $\mathbf{b} := \mathbf{b} - \mu\mathbf{a}$
  If $\|\mathbf{a} - \mathbf{b}\| > \|\mathbf{a} + \mathbf{b}\|$ **then** $\mathbf{b} := -\mathbf{b}$
  Swap$(\mathbf{a}, \mathbf{b})$
  If $[\mathbf{a}, \mathbf{b}]$ is Lagrange-reduced, **then** return $[\mathbf{a}, \mathbf{b}]$
  Else goto *loop*

Algorithm 7 EEAapprox [2]

Input: A sequence $\underline{a}(n) = (a_0\ \cdots, a_{n-1})$ with $n = 2k + 1$ for some integer $k > 0$.
Output: Two integers p, q of size (i.e., bit length) less or equal to k such that that p/q is a RFR of $\underline{a}(n)$ if they exist, or "FALSE" if no such integers exist.
$(r_0, u_0, v_0) := (2^n, 1, 0)$
$(r_1, u_1, v_1) := (\sum_{i=0}^{n-1} a_i 2^i, 0, 1)$
while $r_1 > 2^{n/2}$ **do**
  $(s, t) :=$ Euclidean quotient s and remainder t of r_0 divided by r_1,
  $(u_2, v_2) := (u_0 - su_1, v_0 - sv_1)$
  $(r_0, u_0, v_0) := (r_1, u_1, v_1)$
  $(r_1, u_1, v_1) := (t, u_2, v_2)$
end while
If $\max(|r_1|, |v_1|) \leq 2^{(n-1)/2}$ and v_1 is odd, then
  output (r_1, v_1) (as (p, q)).
else
  output FALSE

APPENDIX D
ARNAULT ET AL.'S MINIMAL BASIS ALGORITHM

Algorithm 8 Minimal Basis Algorithm [2]

Input: A sequence $\underline{a}(n) = (a_0\ \cdots, a_{n-1})$
Output: A minimal basis $[\mathbf{a}, \mathbf{b}]$ of $\mathcal{L}_n$ with $\|\mathbf{a}\| = \lambda_1(\mathcal{L}_n)$, $\|\mathbf{b}\| = \lambda_2(\mathcal{L}_n)$.
$(r_0, u_0, v_0) := (2^n, 1, 0)$
$(r_1, u_1, v_1) := (\sum_{i=0}^{n-1} a_i 2^i, 0, 1)$
while $r_1 > 2^{n/2}$ **do**
  $(s, t) :=$ Euclidean quotient s and remainder t of r_0 divided by r_1,
  $(u_2, v_2) := (u_0 - su_1, v_0 - sv_1)$
  $(r_0, u_0, v_0) := (r_1, u_1, v_1)$
  $(r_1, u_1, v_1) := (t, u_2, v_2)$
end while
$\mathbf{a} := (r_0\ v_0)$
compute the four integers a_1, a_2, a_3, a_4 adjacent to the quotients $(r_0 - v_0)/(r_1 - v_1)$ and $-(r_0 + v_0)/(r_1 + v_1)$.
compute the four vectors $(r'_j\ v'_j) := (r_0 - a_j r_1\ v_0 - a_j v_1)$ for $j = 1, 2, 3, 4$.
set $(r'\ v') := (r'_j\ v'_j)$ with the j minimizing $\max(|r'|, |v'|)$.
$\mathbf{b} := (r'\ v')$.
return $[\mathbf{a}\ \mathbf{b}]$

APPENDIX E
ARNAULT ET AL.'S CORRECTED MRFR ALGORITHM **COR-EEAAPPROX**

See Algorithm 9.

APPENDIX F
MÖLLER'S HGCD ALGORITHM **HGCD**

See Algorithm 10.

Algorithm 9 Cor-EEAapprox [1]

Input: A sequence $\underline{a}(n) = (a_0\ \cdots, a_{n-1})$ with $n = 2k + 3$ for some integer $k > 0$.
Output: Two integers p, q of size (i.e., bit length) less or equal to k such that that p/q is a RFR of $\underline{a}(n)$ if they exist, or "FALSE" if no such integers exist.
$(r_0, u_0, v_0) := (2^n, 1, 0)$
$(r_1, u_1, v_1) := (\sum_{i=0}^{n-1} a_i 2^i, 0, 1)$
while $r_1 > 2^{n/2}$ **do**
  $(s, t) :=$ Euclidean quotient s and remainder t of r_0 divided by r_1,
  $(u_2, v_2) := (u_0 - su_1, v_0 - sv_1)$
  $(r_0, u_0, v_0) := (r_1, u_1, v_1)$
  $(r_1, u_1, v_1) := (t, u_2, v_2)$
end while
If $|v_1| < 2^{n/2}$, then
  output (r_1, v_1) (as (p, q)).
else
  output FALSE

Algorithm 10 HGCD [11]

Input: Two integers a, b with $a, b > 0$ and $\#(a, b) = \ell$.
Output: Two integers c, d and an unimodular matrix $\mathbf{M}$ with $c, d > 0$, $\underline{\#}(c, d) > \lfloor \frac{\ell}{2} \rfloor + 1$, $\#(c - d) \leq \lfloor \frac{\ell}{2} \rfloor + 1$, and $(a\ b)^T = \mathbf{M}(c\ d)^T$
if $|a - b| \leq 1$, **then**
  $\mathbf{M} := \mathbf{I}$, return $a, b, \mathbf{M}$
end if
$s := \lfloor \frac{\ell}{2} \rfloor + 1$
if $\underline{\#}(a, b) \geq \lfloor \frac{3}{4}\ell \rfloor + 2$, **then**
  $p_1 := \lfloor \ell/2 \rfloor$, $a_1 := \lfloor a/2^{p_1} \rfloor$, $a_2 := a - 2^{p_1} a_1$, $b_1 := \lfloor b/2^{p_1} \rfloor$, $b_2 := b - 2^{p_1} b_1$
  $(\alpha, \beta, \mathbf{M}) \leftarrow \mathrm{HGCD}(a_1, b_1)$
  $(a\ b)^T := 2^{p_1}(\alpha\ \beta)^T + \mathbf{M}^{-1}(a_2\ b_2)^T$
else $\mathbf{M} := \mathbf{I}$
end if
while $\#(a, b) > \lfloor 3\ell/4 \rfloor + 1$ and $\#(a - b) > s$ **do**
  if $a \geq b$, then
    $q := \lfloor a/b \rfloor$
    if $\#(a - qb) \geq s$, **then** $(a\ b) := (b\ a - qb)$ and $\mathbf{M} := \mathbf{M} \begin{pmatrix} q & 1 \\ 1 & 0 \end{pmatrix}$
    else $q := q - 1$, $(a\ b) := (b\ a - qb)$ and $\mathbf{M} := \mathbf{M} \begin{pmatrix} q & 1 \\ 1 & 0 \end{pmatrix}$
  else
    $q := \lfloor b/a \rfloor$
    if $\#(b - qa) > s$, **then** $(a\ b) := (b - qa\ a)$ and $\mathbf{M} := \mathbf{M} \begin{pmatrix} 0 & 1 \\ 1 & q \end{pmatrix}$
    else $q := q - 1$, $(a\ b) := (b - qa\ a)$ and $\mathbf{M} := \mathbf{M} \begin{pmatrix} 0 & 1 \\ 1 & q \end{pmatrix}$
end while
if $\underline{\#}(a, b) > s + 2$, **then**
  $n' := \#(a, b)$, $p_2 := 2s - n' + 1$
  $a_1 := \lfloor a/2^{p_2} \rfloor$, $a_2 := a - 2^{p_2} a_1$, $b_1 := \lfloor b/2^{p_2} \rfloor$, $b_2 := b - 2^{p_2} b_1$
  $(\alpha, \beta, \mathbf{M}') \leftarrow \mathrm{HGCD}(a_1, b_1)$
  $(a\ b)^T := 2^{p_2}(\alpha\ \beta)^T + (\mathbf{M}')^{-1}(a_2\ b_2)^T$
  $\mathbf{M} := \mathbf{M}\mathbf{M}'$
end if
while $\#(a - b) > s$ **do**
  if $a \geq b$, then
    $q := \lfloor a/b \rfloor$
    if $\#(a - qb) > s$, **then** $(a\ b) := (b\ a - qb)$ and $\mathbf{M} := \mathbf{M} \begin{pmatrix} q & 1 \\ 1 & 0 \end{pmatrix}$
    else $q := q - 1$, $(a\ b) := (b\ a - qb)$ and $\mathbf{M} := \mathbf{M} \begin{pmatrix} q & 1 \\ 1 & 0 \end{pmatrix}$
  else
    $q := \lfloor b/a \rfloor$
    if $\#(b - qa) > s$, **then** $(a\ b) := (b - qa\ a)$ and $\mathbf{M} := \mathbf{M} \begin{pmatrix} 0 & 1 \\ 1 & q \end{pmatrix}$
    else $q := q - 1$, $(a\ b) := (b - qa\ a)$ and $\mathbf{M} := \mathbf{M} \begin{pmatrix} 0 & 1 \\ 1 & q \end{pmatrix}$
  end if
end while
return $a, b, \mathbf{M}$

REFERENCES

[1] F. Arnault and T. P. Berger, "Correction to, feedback with carry shift registers synthesis with the Euclidean algorithm," *IEEE Trans. Inf. Theory*, vol. 54, no. 1, p. 502, Apr. 2008.
[2] F. Arnault, T. P. Berger, and A. Necer, "Feedback with carry shift registers synthesis with the Euclidean algorithm," *IEEE Trans. Inf. Theory*, vol. 50, no. 5, pp. 910–917, May 2004.

[3] B. M. M. de Weger, "Approximation lattices of p-adic numbers," *J. Number Theory*, vol. 24, no. 1, pp. 70–88, Sep. 1986.
[4] A. Klapper and M. Goresky, "2-adic shift registers," in *Proc. Int. Workshop Fast Softw. Encryption*. Berlin, Germany: Springer, 1993, pp. 1–5.
[5] A. Klapper and M. Goresky, "Cryptanalysis based on 2-adic rational approximation," in *Proc. Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 1995, pp. 262–273.
[6] A. Klapper and M. Goresky, "Feedback shift registers, 2-adic span, and combiners with memory," *J. Cryptol.*, vol. 10, no. 2, pp. 111–147, Mar. 1997.
[7] Y. Li, Z. Yang, K. Li, and L. Qu, "A new algorithm on the minimal rational fraction representation of feedback with carry shift registers," *Des., Codes Cryptogr.*, vol. 88, no. 3, pp. 533–552, Mar. 2020.
[8] K. Mahler, "On a geometrical representation of p-adic numbers," *Ann. Math.*, vol. 41, no. 1, pp. 8–56, 1940.
[9] J. Massey, "Shift-register synthesis and BCH decoding," *IEEE Trans. Inf. Theory*, vol. 15, no. 1, pp. 122–127, Jan. 1969.
[10] D. Micciancio and S. Goldwasser, *Complexity of Lattice Problems: A Cryptographic Perspective* (International Series in Engineering and Computer Science), vol. 671. Boston, MA, USA: Kluwer, Mar. 2002.
[11] N. Möller, "On Schönhage's algorithm and subquadratic integer GCD computation," *Math. Comput.*, vol. 77, no. 261, pp. 589–607, 2008.

Jun Che received the B.E. degree in information security from Qingdao University in 2020. He is currently pursuing the M.S. degree with the School of Cyber Science and Technology, Shandong University. His research interests include authentication encryption and cryptography.

Chengliang Tian received the B.S. and M.S. degrees in mathematics from Northwest University, Xi'an, China, in 2006 and 2009, respectively, and the Ph.D. degree in information security from Shandong University, Jinan, China, in 2013. He was a Post-Doctoral Researcher with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing. He is currently an Associate Professor with the College of Computer Science and Technology, Qingdao University. His research interests include lattice-based cryptography and cloud computing security.

Yupeng Jiang received the B.S. degree in mathematics from Tsinghua University in 2008 and the Ph.D. degree in applied mathematics from the Academy of Mathematics and Systems Science, Chinese Academy of Sciences, in 2013. He was with the Institute of Information Engineering, Chinese Academy of Sciences, from July 2013 to March 2021. He is currently an Associate Professor with the School of Cyber Science and Technology, Beihang University. His research interests include number theory, computational number theory, and sequences.

Guangwu Xu (Senior Member, IEEE) received the Ph.D. degree in mathematics from SUNY Buffalo. He is currently with the School of Cyber Science and Technology, Shandong University, China. His research interests include cryptography, arithmetic number theory, compressed sensing, algorithms, and functional analysis.