Quantum Algorithm for Optimization and Polynomial System Solving over Finite Field and Application to Cryptanalysis*

CHEN Yu-Ao · GAO Xiao-Shan · YUAN Chun-Ming

DOI:
Received: x x 20xx / Revised: x x 20xx

Abstract In this paper, we give quantum algorithms for two fundamental computation problems: solving polynomial systems over finite fields and optimization where the arguments of the objective function and constraints take values from a finite field or a bounded interval of integers. The quantum algorithms can solve these problems with any given success probability and have polynomial runtime complexities in the size of the input, the degree of the inequality constraints, and the condition number of the associated matrices of the problem. So, we achieved exponential speedup for these problems when their condition numbers are small. As applications, quantum algorithms are given to three basic computational problems in cryptography: the short integer solution problem, the shortest vector problem, the polynomial system with noise problem, and cryptanalysis for the lattice-based NTRU cryptosystem. It is shown that these problems and NTRU can against quantum computer attacks only if their condition numbers are large, so the condition number could be used as a new criterion for lattice-based post-quantum cryptosystems.

Keywords quantum algorithm, polynomial system solving, polynomial system with noise, finite field, integer programming, cryptanalysis of NTRU.

1 Introduction

Solving polynomial systems and optimization over finite fields are fundamental computation problems in mathematics and computer science, which are also typical NP hard problems. In this paper, we give quantum algorithms to these problems, which could be exponentially faster than the traditional methods under certain conditions.

CHEN Yu-Ao
KLMM, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China;
Thrust of Artificial Intelligence, Information Hub, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511453, China. Email: chenyuao@amss.ac.cn.
GAO Xiao-Shan · YUAN Chun-Ming
KLMM, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China;
University of Chinese Academy of Sciences, Beijing 100049, China.
Email: xgao@mmrc.iss.ac.cn; cmyuan@mmrc.iss.ac.cn.
*This research was supported by NKRDP grant No.2018YFA0704705, NSFC grant Nos.12288201, 12271516, the Guangdong Provincial Quantum Science Strategic Initiative grant Nos.GDZX2403008, GDZX2403001, the Guangdong Provincial Key Lab of ICSCUIT grant No.2023B1212010007, the Quantum Science Center of Guangdong-Hong Kong-Macao Greater Bay Area, and the Education Bureau of Guangzhou Municipality.
◇ *This paper was recommended for publication by Editor.*

1.1 Main results

Let $\mathbb{F}_q$ be a finite field, where $q = p^m$ for a prime number p and $m \in \mathbb{N}_{\geq 1}$. Let $\mathcal{F} = \{f_1, f_2, \ldots, f_r\} \subset \mathbb{F}_q[\mathbb{X}]$ be a set of polynomials in variables $\mathbb{X} = \{x_1, x_2, \ldots, x_n\}$ and with *total sparseness* $T_{\mathcal{F}} = \sum_{i=1}^{r} \#f_i$, where $\#f$ denotes the number of terms in f. For $\varepsilon \in (0, 1)$, we prove that

Theorem 1.1 *There exists a quantum algorithm which decides whether $\mathcal{F} = 0$ has a solution in $\mathbb{F}_q^n$ and computes one if $\mathcal{F} = 0$ has solutions in $\mathbb{F}_q^n$, with success probability at least $1 - \varepsilon$ and complexity $\widetilde{O}(T_{\mathcal{F}}^{3.5} D^{3.5} m^{5.5} \log^{4.5} p\kappa^2 \log 1/\varepsilon)$, where $D = n + \sum_{i=1}^{n} \lfloor \log_2 \max_j \deg_{x_i} f_j \rfloor$, $T_{\mathcal{F}}$ is the total sparseness of $\mathcal{F}$, and κ is the condition number of the associated matrix of $\mathcal{F}$ (see Theorem 3.13 for definition).*

The complexity of a quantum algorithm is the number of quantum gates needed to solve the problem. Since $T_{\mathcal{F}}, D, \log p^m$ are smaller than the input size, the complexity of the algorithm is polynomial in the input size and the condition number of the associated matrix of the problem (abbr. condition number of the problem), which means that we can solve polynomial systems over finite fields using quantum computers with any given success probability and in polynomial-time if the condition number κ of $\mathcal{F}$ is small, say when κ is poly(n, D).

We also give a quantum algorithm to solve the following optimization problem.

$$\begin{aligned} \min_{\mathbb{X} \in \mathbb{F}_p^n, \mathbb{Y} \in \mathbb{Z}^m} o(\mathbb{X}, \mathbb{Y}) \qquad & \text{subject to} \\ & f_j(\mathbb{X}) = 0 \mod p,\ j = 1, 2, \ldots, r; \\ & 0 \leq g_i(\mathbb{X}, \mathbb{Y}) \leq b_i,\ i = 1, 2, \ldots, s;\ 0 \leq y_k \leq u_k, k = 1, 2, \ldots, m, \end{aligned} \tag{1}$$

where $\mathcal{F} = \{f_1, f_2, \ldots, f_r\} \subset \mathbb{F}_p[\mathbb{X}]$, $\mathbb{Y} = \{y_1, y_2, \ldots, y_m\}$, $\mathcal{G}_o = \{o, g_1, g_2, \ldots, g_s\} \subset \mathbb{Z}[\mathbb{X}, \mathbb{Y}]$, and $b_1, b_2, \ldots, b_s$, $u_1, u_2, \ldots, u_m \in \mathbb{N}$. The complexity of the algorithm is polynomial in the size of the input, $\deg(g_i)$, $\deg(o)$, and the condition number of the problem (see Theorem 5.4 for definition). Since Problem (1) is NP-hard, the algorithm gives an exponential speedup over traditional methods if the condition number is small, say poly(n, m).

Note that for $q = p$, Problem (1) includes polynomial system solving over $\mathbb{F}_q$ as a special case. Problem (1) is meaningless for $\mathbb{F}_q$ with $q = p^m$ and $m > 1$, since $\mathbb{F}_q$ cannot be embedded into $\mathbb{Z}$.

We apply our quantum algorithms to three computational problems widely used in cryptography: the *short integer solution problem* (SIS) [1], the *shortest vector problem* (SVP) [4, 6, 25], and the *polynomial systems with noise problem* (PSWN) [2, 19, 22]. We also show how to recover the private keys for the latticed based cryptosystem NTRU [21, 29] with our algorithm. The complexity for solving all of these problems is polynomial in the input size and their condition numbers.

Ajtai started lattice-based cryptography in a seminal work [1], where a family of one-way functions is given based on SIS. The SIS problem is to find a nonzero solution of a homogeneous linear system $A\mathbb{X} = 0 \mod p$ for $A \in \mathbb{F}_p^{r \times n}$, such that $||\widehat{\mathbb{X}}||_2$ is smaller than a given bound. Our quantum algorithm for SIS has complexity $\widetilde{O}((n \log p + r)^{2.5}(T_A \log p + n \log^2 p)\kappa^2)$, where T_A is the number of nonzero elements in A and κ is the condition number of the problem.

The SVP and CVP are two basic NP-hard problems widely used in lattice-based cryptography. The SVP is to find a nonzero vector with the smallest Euclidean norm in a lattice in $\mathbb{R}^m$. The CVP is to find a vector in a lattice which is closest to a given vector. Our quantum algorithm for SVP has complexity $\widetilde{O}(m(n^{7.5} + m^{2.5})(n^3 + \log h) \log^{4.5} h\kappa^2)$, where n is the rank of the lattice, h is the maximal value in the lattice generators, and κ is the condition number of the problem. Our quantum algorithm for CVP has a similar complexity.

NTRU is a lattice-based public key cryptosystem proposed by Hoffstein, Pipher and Silverman [21], which is one of the most promising candidates for post-quantum cryptosystems. Our quantum algorithm can be used to recover the private key from the public key in time $\widetilde{O}(N^{4.5}\log^{3.5} q\kappa^2)$ for an NTRU with parameters (N,p,q) with $q>p$. In particular, we show that the three versions of NTRU recommended in [21] have the desired security against quantum computers only if their condition numbers are large.

The lattice-based computational problems SVP and LWE are the bases for 23 of the 69 submissions to NIST's effort to standardize the post-quantum public-key encryption systems [4]. LWE is another important computational problem in cryptography, which is the randomized versions of CVP and can be reduced to the SIS problem [32]. Lattice-based computational problems have many applications [30]. In theory, our results imply that the 23 proposed cryptosystems can withstand the attack of quantum computers only if their condition numbers are large. So, the condition number could be used as a new criterion for lattice-based post-quantum cryptosystems.

Let p be a prime and $\mathcal{F}=\{f_1,f_2,\ldots,f_r\}\subset\mathbb{F}_p[\mathbb{X}]$ with $r\gg n$. The PSWN is to find an $\mathbb{X}\in\mathbb{F}_p^n$ which satisfies the maximal number of equations in $\mathcal{F}$. The problem is also called MAX-POSSO [2, 22]. Our quantum algorithm for PSWN has complexity $\widetilde{O}(r^{3.5}T_{\mathcal{F}}^{3.5}\log^8 p\kappa^2)$, where κ is the condition number of the problem. The PSWN is very hard in the sense that, even for the *linear system with noise* (LSWN) over $\mathbb{F}_p$, finding an $\mathbb{X}$ satisfying more than $1/p$ of the equations is NP hard [19, 41].

1.2 Main ingredients of the algorithm

Let $\mathcal{F}\subset\mathbb{C}[\mathbb{X}]$ be a set of polynomials over $\mathbb{C}$. A solution of $\mathcal{F}$ is called Boolean if its components are 0 or 1. Similarly, a variable x is called *a Boolean variable* if its values are either 0 or 1. In [13], we give a quantum algorithm* to find Boolean solutions of a polynomial system over $\mathbb{C}$, which is called B-POSSO in the rest of this paper. The main idea of the quantum algorithms proposed in this paper is to reduce the problem to be solved to B-POSSO, under the condition that the number of variables and the total sparseness of the new polynomial system is polynomial in the size of the original polynomial system.

Our algorithm for problem (1) consists of three main steps: (1) The equational constraints $f_j(\mathbb{X})=0 \mod p, j=1,2,\ldots,r$ are reduced into B-POSSO. (2) The inequality constraints $0\le g_i(\mathbb{X},\mathbb{Y})\le b_i, i=1,2,\ldots,s$ are reduced into B-POSSO. (3) The problem of finding the minimal value of the objective function is reduced several B-POSSOs. We will give a brief introduction to each of these three steps below.

A key method used in our algorithm is to construct a polynomial in Boolean variables to represent the integers $0,1,\ldots,b$ for $b\in\mathbb{Z}_{>1}$. Let $\theta_b(\mathbb{G}_{\rm bit})=\sum_{k=0}^{\lfloor\log_2 b\rfloor-1}G_k2^k+(b+1-2^{\lfloor\log_2 b\rfloor})G_{\lfloor\log_2 b\rfloor}$, where $\mathbb{G}_{\rm bit}=\{G_0,G_1,\ldots,G_{\lfloor\log_2 b\rfloor}\}$ is a set of Boolean variables. Then, the values of $\theta_b(\mathbb{G}_{\rm bit})$ are exactly $0,1,\ldots,b$.

For $\mathcal{F}\subset\mathbb{F}_p[\mathbb{X}]$ and $\mathbb{F}_p=\{0,1,\ldots,p-1\}$, we use three steps to reduce the problem of finding a solution of $\mathcal{F}$ in $\mathbb{F}_p$ to a B-POSSO. (1) $\mathcal{F}$ is reduced to a quadratic polynomial system (MQ) $\mathcal{F}_1$ by introducing new variables. (2) Each variable in $\mathcal{F}_1$ is expanded as $x_i=\theta_{p-1}(\mathbb{X}_i)$ and $\mathcal{F}_1$ is reduced to another MQ $\mathcal{F}_2$ in Boolean variables $\mathbb{X}_i=\{X_{ij},j=0,1,\ldots,\lfloor\log_2(p-1)\rfloor\}$. Since $\mathcal{F}_1$ is quadratic, the total sparseness of $\mathcal{F}_2$ is well controlled. (3) We obtain a polynomial over $\mathbb{C}$ from $\mathcal{F}_2$ as follows $\mathcal{F}_3=\{g-\theta_{\#g}(\mathbb{U}_g)p\,|\,g\in\mathcal{F}_2\}$, where $\#g$ is the number of terms in g and $\mathbb{U}_g$ is a set of $\#g$ Boolean variables. It is shown that solutions of $\mathcal{F}$ in $\mathbb{F}_p$ can be recovered from Boolean solutions of $\mathcal{F}_3$, which can be found with the quantum algorithm from [13].

*No detailed knowledge of quantum algorithms is needed to read this paper. What we do in this paper is to use traditional methods to reduce the problems to be solved to this quantum algorithm.

We also reduce the inequality constraint $0 \le g_k(\mathbb{X}, \mathbb{Y}) \le b_k$ of Problem (1) into B-POSSO. There exist $\mathbb{X}$ and $\mathbb{Y}$ such that $0 \le g_k(\mathbb{X}, \mathbb{Y}) \le b_k$ if and only if $g_k(\mathbb{X}, \mathbb{Y}) - \theta_{b_k}(\mathbb{G}_{b_k}) = 0$ has a solution for $\mathbb{X}$, $\mathbb{Y}$, and $\mathbb{G}_{b_k}$, where $\mathbb{G}_{b_k}$ is a set of Boolean variables. Since $0 \le x_i \le p-1$ and $0 \le y_i \le u_i$, we can reduce $g_k(\mathbb{X}, \mathbb{Y})$ into a polynomial in Boolean variables by first reducing $g(\mathbb{X}, \mathbb{Y})$ into an MQ and then expanding the variables x_i, y_j into Boolean variables by using the θ function. As a consequence, the inequality constraint of Problem (1) is reduced into B-POSSO. Let d be the maximal degree of all g_k. Then the values of g_k is exponential in d and hence the number of Boolean variables needed is polynomial in d. This is why the complexity of the algorithm depends on d.

Since all variables are bounded, the objective function o is also bounded, and we can assume that the values of o are in a feasible interval $[\alpha, \mu)$ for some $\alpha, \mu \in \mathbb{N}$. We design a novel search scheme to reduce the minimization of $o(\mathbb{X}, \mathbb{Y}) \in [\alpha, \mu)$ into several B-POSSOs. We approximately bisect the feasible interval $[\alpha, \mu)$ into subintervals $[\alpha, 2^\beta)$ and $[2^\beta, \mu)$ and decide whether $o \in [\alpha, 2^\beta)$ has a solution, which is equivalent to solving equation $o - (\alpha + \sum_{j=0}^{\beta-1} H_j 2^j) = 0$ for Boolean variables H_j. If $o \in [\alpha, 2^\beta)$ has a solution $\widehat{\mathbb{X}}$ and $\widehat{\mathbb{Y}}$, we repeat the procedure for $[\alpha, o(\widehat{\mathbb{X}}, \widehat{\mathbb{Y}}))$; otherwise, we repeat the procedure for $[2^\beta, \mu)$. As a consequence, we can find the minimal value of o by solving several B-POSSOs.

1.3 Relation with existing work

Problem (1) includes many important problems as special cases, such as solving polynomial systems in finite fields [9, 15–17], SIS [1], SVP/CVP [4, 6, 25], PSWN [2, 19, 22, 41], the $(0,1)$-programming [18], the *quadratic unconstrained binary optimization problem* which is the mathematical problem that can be solved by the D-Wave System [23], which are all important computation problems and were widely studied. The main motivation of this study is that polynomial system solving is a key tool in cryptanalysis [27, 28, 38, 39].

Comparing to the existing work such as the symbolic computation approach [26, 36, 37, 40], our approach is new and has two major advantages. First, we give a universal approach to a very general problem. Second, the complexity of our algorithm is polynomial in the inputs size, the degree of the inequalities, and the condition number of the problem. Since the problems under consideration are NP hard, the existing algorithms are exponential in some of the parameters such as the number of variables. In this aspect, we give a new way of looking at these NP hard problems by reducing the computational difficulty to the size of the condition number.

Our algorithm is based on the quantum algorithm to solve B-POSSOs proposed in [13], which in turn is based on the HHL quantum algorithm and its variants to solve linear systems [5, 14, 20, 33]. Comparing to the HHL algorithm, we can give the exact solution, while the HHL algorithm can only give the quantum state. The speedup of our algorithms comes from the HHL algorithm. The limitation on the condition number is inherited from the HHL algorithm, and it is proved in [20] that the dependence on the condition number cannot be substantially improved. Also note that, the best classic numerical method for solving an order N linear equation $Ax = b$ has complexity $\widetilde{O}(N\sqrt{\kappa})$ [34], which also depends on the condition number κ of A.

The method of treating the inequality constraints with the function $\theta_b(\mathbb{G}_{\rm bit})$ simplifies the computational significantly. The binary representation $\eta_b = \sum_{i=0}^{\lfloor \log_2(b) \rfloor} B_i 2^i$ for b is often used in the literature to represent the integers $0, 1, \ldots, b$. The values of η_b are $0, 1, \ldots, 2^{\lfloor \log_2(b) \rfloor + 1} - 1$, which may contain integers strictly larger than b and cannot be used to represent inequality constraints of Problem (1). In [3, 7], the integer inequality $0 \le g \le b$ is reduced to $\prod_{i=0}^{b}(g - i) = 0$. Our reduction $g - \theta_b(\mathbb{G}_g)$ is better, which does not increase the degree of the equation and the size of the equation is increased in the logarithm scale, while the method used in [3, 7] increases the degree by a factor b and increases the size of the equation exponentially.

The remainder of this paper is organized as follows. In Section 2, we define the $\theta_b(\mathbb{G}_{\text{bit}})$ function and give an explicit formula to reduce a polynomial system into an MQ. In Section 3, we present the algorithm for solving polynomial systems over finite fields. In Section 4, we show how to reduce the inequality constraints in problem (1) to a B-POSSO. In Section 5, we present the algorithm for solving problem (1). In Section 6, we present a quantum algorithm for PSWN. In Section 7, we present a quantum algorithm for SIS. In Section 8, we present a quantum algorithm for SVP/CVP. In Section 9, we present a quantum algorithm to recover the private key for NTRU. In Section 10, conclusions are given.

2 Two basic reductions

In this section, we give two basic reductions frequently used in the paper: to represent an integer interval with a Boolean polynomial and to reduce a polynomial system to an MQ.

2.1 Represent an integer interval with a Boolean polynomial

A variable X is called a *Boolean variable* if it satisfies $X^2 - X = 0$. In this paper, we use uppercase symbols to represent Boolean variables. A polynomial is called a *Boolean polynomial* if it is in a set of Boolean variables. In this section, we will construct a Boolean polynomial whose values are exactly $0, 1, \ldots, b$ for a given positive integer $b > 0$.

Set $s = \lfloor \log_2(b) \rfloor$ and introduce $s+1$ Boolean variables $\mathbb{B}_{\text{bit}} = \{B_0, B_1, \ldots, B_s\}$. Inspired by $b = (2^s - 1) + (b + 1 - 2^s)$, we introduce the function $\theta_b(\mathbb{B}_{\text{bit}})$: $\theta_1(\mathbb{B}_{\text{bit}}) = B_0$ and for $b > 1$

$$\theta_b(\mathbb{B}_{\text{bit}}) = \sum_{i=0}^{s-1} 2^i B_i + (b + 1 - 2^s) B_s. \tag{2}$$

Lemma 2.1 *When evaluated in $\mathbb{C}$ or $\mathbb{F}_p = \{0, 1, \ldots, p-1\}$ with $p > b$, $\theta_b(\mathbb{B}_{\text{bit}})$ is a surjective map from $\{0,1\}^{s+1}$ to $\{0, 1, \ldots, b\}$. Furthermore, $|\mathbb{B}_{\text{bit}}| = \#\theta_b(\mathbb{B}_{\text{bit}}) = \lfloor \log_2(b) \rfloor + 1 = s + 1$, where $\#\theta_b(\mathbb{B}_{\text{bit}})$ is the number of terms in the polynomial $\theta_b(\mathbb{B}_{\text{bit}})$.*

Proof We first assume that $\theta_b(\mathbb{B}_{\text{bit}})$ is evaluated over $\mathbb{C}$. It is easy to check this lemma when $b = 1$. When $b > 1$, from the definition of s, we have $b/2 < 2^s \le b$ and hence $2^s - 1 < b$. Since the values of $\sum_{i=0}^{s-1} 2^i B_i$ are $0, 1, \ldots, 2^s - 1$, for any integer $n \in [0, 2^s - 1]$, n has a preimage of map $\theta_b(\mathbb{B}_{\text{bit}})$, where $B_s = 0$. Now consider an integer $n \in [2^s, b]$. Set set $B_s = 1$ and it suffices to show that $0 \le n - (b + 1 - 2^s) \le 2^s - 1$. Since $n \ge 2^s$, we have $n - (b + 1 - 2^s) \ge 2 \cdot 2^s - b - 1 > 2 \cdot b/2 + 1 - b - 1 = 0$. Since $n \le b$, we have $n - (b + 1 - 2^s) \le 2^s - 1$. Thus, $0 \le n - (b + 1 - 2^s) \le 2^s - 1$, and then n has a preimage of map $\theta_b(\mathbb{B}_{\text{bit}})$, where $B_s = 1$. It is clear $\#\mathbb{B}_{\text{bit}} = \lfloor \log_2(b) \rfloor + 1$. Since $b + 1 - 2^s > 0$, we have $\#\theta_b(\mathbb{B}_{\text{bit}}) = \lfloor \log_2(b) \rfloor + 1$. The lemma is also valid when $\theta_b(\mathbb{B}_{\text{bit}})$ is evaluated over $\mathbb{F}_p$, since all values in the computation are $\le p - 1$. ∎

For instance, $\theta_6(\mathbb{B}_{\text{bit}}) = B_0 + 2B_1 + 3B_2$, $\theta_7(\mathbb{B}_{\text{bit}}) = B_0 + 2B_1 + 4B_2$, $\theta_8(\mathbb{B}_{\text{bit}}) = B_0 + 2B_1 + 4B_2 + B_3$.

Remark 2.2 It is easy to check that θ_b is injective if and only if $b = 2^k - 1$ for some positive integer k. For instance, θ_6 is not injective: 3 has two preimages $B_0 = 1, B_1 = 1, B_2 = 0$ and $B_0 = 0, B_1 = 0, B_2 = 1$.

2.2 Reduce polynomial system to MQ

It is well known that a polynomial system can be reduced to an MQ by introducing some new indeterminates. In this section, we give an explicit reduction that is needed in the complexity analysis in this paper.

For any field F, let $F[\mathbb{X}]$ be the polynomial ring over F in the indeterminates $\mathbb{X} = \{x_1, x_2, \ldots, x_n\}$. Denote the sparseness (number of terms) of $f \in F[\mathbb{X}]$ as $\#f$. For $\mathcal{F} = \{f_1, f_2, \ldots, f_r\} \subset F[\mathbb{X}]$, denote $T_{\mathcal{F}} = \sum_{i=1}^{r} \#f_i$ to be the *total sparseness* of $\mathcal{F}$, $N_{\mathcal{F}} = \#\mathbb{X} = n$ to be the *number of indeterminates* in $\mathcal{F}$, $d_i = \max_j \deg_{x_i}(f_j)$ to be the degree of $\mathcal{F}$ in x_i, $M(\mathcal{F})$ to be the set of all monomials in $\mathcal{F}$, and $C(\mathcal{F})$ to be the size of the coefficients of the polynomials in $\mathcal{F}$, $(\mathcal{F})_{F[\mathbb{X}]}$ to be the ideal generated by $\mathcal{F}$ in $F[\mathbb{X}]$.

We want to introduce some new indeterminates to rewrite $\mathcal{F}$ as an MQ.

Lemma 2.3 *Let $\mathcal{F} = \{f_1, f_2, \ldots, f_r\} \subset F[\mathbb{X}]$. We can introduce a set of new indeterminates $\mathbb{V}$ and an* MQ *$Q(\mathcal{F}) \subset F[\mathbb{X}, \mathbb{V}]$ such that $(\mathcal{F})_{F[\mathbb{X}]} = (Q(\mathcal{F}))_{F[\mathbb{X},\mathbb{V}]} \cap F[\mathbb{X}]$. Furthermore, we have $\#\mathbb{V} = (T_{\mathcal{F}} + 1)\sum_{i=1}^{n} \lfloor \log_2 d_i \rfloor + nT_{\mathcal{F}} = O(T_{\mathcal{F}} D)$, $N_{Q(\mathcal{F})} = n + \#\mathbb{V} = O(T_{\mathcal{F}} D)$, $\#Q(\mathcal{F}) = r + \#\mathbb{V} = O(T_{\mathcal{F}} D)$, $T_{Q(\mathcal{F})} = T_{\mathcal{F}} + 2\#\mathbb{V} = O(T_{\mathcal{F}} D)$, and $C(Q(\mathcal{F})) = C(\mathcal{F})$, where $D = n + \sum_{i=1}^{n} \lfloor \log_2 d_i \rfloor$ and $d_i = \max_j \deg_{x_i}(f_j)$.*

Proof If $\mathcal{F}$ is already an MQ, set $Q(\mathcal{F}) = \mathcal{F}$ and $\mathbb{V} = \emptyset$. Otherwise, first, we introduce new indeterminates u_{ij} for $j = 1, 2, \ldots, \lfloor \log_2 d_i \rfloor$ and new polynomials $u_{i1} - x_i^2$ and $u_{i(j+1)} - u_{ij}^2$ for $j = 2, 3, \ldots, \lfloor \log_2 d_i \rfloor - 1$. It is clear that $x_i^{2^j} = u_{ij}$. Without loss of generality, we assume $d_i \geq 2$ and if $d_i \leq 1$, then we do not need these u_{ij}. Let $\mathbb{X}^\alpha = \prod_{i=1}^{n} x_i^{\alpha_i}$ be a monomial of $\mathcal{F}$, and $\alpha_i = \sum_{k=1}^{l_i} 2^{\nu_{ik}}$ be the binary representation of $\alpha_i \leq d_i$, where $l_i \leq \lfloor \log_2 \alpha_i \rfloor + 1 \leq \lfloor \log_2 d_i \rfloor + 1$ and $\nu_{i1} < \nu_{i2} < \cdots < \nu_{il_i}$. Thus

$$\mathbb{X}^\alpha = \prod_{i=1}^{n} x_i^{\sum_{k=1}^{l_i} 2^{\nu_{ik}}} = \prod_{i=1}^{n} \prod_{k=1}^{l_i} x_i^{2^{\nu_{ik}}} \equiv \prod_{i=1}^{n} \prod_{k=1}^{l_i} u_{i\nu_{ik}}.$$

Flatten the subscripts i and j in $\{u_{ij}\}$, we could rearrange the set $\{u_{ij}\}$ as $\{u_i \mid i = 1, 2, \ldots, L_\alpha\}$, and we have $\mathbb{X}^\alpha = \prod_{i=1}^{L_\alpha} u_i$, where $L_\alpha = \sum_k l_k \leq \sum_{i=1}^{n} (\lfloor \log_2 d_i \rfloor + 1) \leq \sum_{i=1}^{n} \lfloor \log_2 d_i \rfloor + n$. To rewrite this product as an MQ, we introduce new indeterminates $\{v_1, v_2, \ldots, v_{L_\alpha - 2}\}$ and quadratic polynomials $v_1 - u_1 u_2$, $v_i - v_{i-1} u_{i+1}$ for $i = 2, 3, \ldots, L_\alpha - 2$. Then the monomial $\mathbb{X}^\alpha$ is represented as $\mathbb{X}^\alpha = V(\mathbb{X}^\alpha) = v_{L_\alpha - 2} u_{L_\alpha}$. Denote $Q(\mathbb{X}^\alpha) = \{v_1 - u_1 u_2\} \cup \{v_i - v_{i-1} u_{i+1} \mid i = 2, 3, \ldots, L_\alpha - 2\}$. Finally, we obtain an MQ

$$\begin{aligned} Q(\mathcal{F}) \quad = \quad & \{u_{i1} - x_i^2, u_{i(k_i+1)} - u_{ik_i}^2 \mid i = 1, 2, \ldots, n, k_i = 2, 3, \ldots, \lfloor \log_2 d_i \rfloor - 1\} \\ & \cup \{\widehat{f}_j \mid j = 1, 2, \ldots, r\} \cup \cup_{\mathbb{X}^\alpha \in M(\mathcal{F})} Q(\mathbb{X}^\alpha) \subset F[\mathbb{X}, \mathbb{V}], \end{aligned} \tag{3}$$

where $\mathbb{V} = \{u_i, v_k\}$ and $\widehat{f}_i$ is obtained from f_i by replacing $\mathbb{X}^\alpha \in M(f_i)$ by $V(\mathbb{X}^\alpha) = v_{L_\alpha - 2} u_{L_\alpha}$ according to the above procedure. For convenience, we denote

$$\widehat{Q}(f_j) \quad = \quad \widehat{f}_j, j = 1, 2, \ldots, r. \tag{4}$$

Let $\mathbb{V} = \{u_i, v_k\}$ be the set of new indetermiantes. It is clear that the number of these u_{ij} is $\sum_{i=1}^{n} \lfloor \log_2 d_i \rfloor$. To represent $\mathbb{X}^\alpha$, we need $\sum_{i=1}^{n} l_i - 2 \leq \sum_{i=1}^{n} \lfloor \log_2 d_i \rfloor + n$ new indeterminates v_i. In total, we have $\#V \leq \sum_{i=1}^{n} \lfloor \log_2 d_i \rfloor + T_{\mathcal{F}} (\sum_{i=1}^{n} l_i - 2) \leq (T_{\mathcal{F}} + 1) \sum_{i=1}^{n} \lfloor \log_2 d_i \rfloor + nT_{\mathcal{F}} = O(T_{\mathcal{F}} D)$. Then, $N_{Q(\mathcal{F})} = \#\mathbb{X} + \#\mathbb{V} = O(T_{\mathcal{F}} D)$, since $n \leq D$. $\#Q(\mathcal{F}) = r + \#\mathbb{V} = O(T_{\mathcal{F}} D)$, since $r \leq T_{\mathcal{F}}$. $Q(\mathcal{F})$ contains r polynomials $\widehat{Q}(f_j), j = 1, 2, \ldots, r$ and $\#V$ binomials. Then $T_{Q(\mathcal{F})} = T_{\mathcal{F}} + 2\#\mathbb{V} = O(T_{\mathcal{F}} D)$. Since we only introduce new coefficients ± 1, we have $C(Q(\mathcal{F})) = C(\mathcal{F})$. ∎

Example 2.4 Let $\mathcal{F} = \{f_1 = x_1^3 x_2^5 + 2x_1^7 x_2^5 + 3\}$. We have $d_1 = 7$, $d_2 = 5$, and $Q_1 = \{u_{11} - x_1^2, u_{12} - u_{11}^2, u_{21} - x_2^2, u_{22} - u_{21}^2\}$. Then $x_1^3 x_2^5 = x_1 u_{11} x_2 u_{22} = v_2 u_{22}$, $x_1^7 x_2^5 = x_1 u_{11} u_{12} x_2 u_{22} = v_5 u_{22}$, where $Q_2 = \{v_1 - x_1 u_{11}, v_2 - x_2 v_1, v_3 - x_1 u_{11}, v_4 - v_3 u_{12}, v_5 - v_4 x_2\}$. Finally, $Q(\mathcal{F}) = Q_1 \cup Q_2 \cup \{v_2 u_{22} + 2v_5 u_{22} + 3\}$. Note that the above representation is not optimal and we can use less new variables to represent $f_1 = x_1 v_2 + 2x_1 v_2 x_1^4 + 3 = x_1 v_2 + 2x_1 v_3' + 3$, where $v_3' = v_2 u_{12}$.

Remark 2.5 As mentioned in Example 2.4, the representation for $Q(\mathcal{F})$ is not optimal. The binary decision diagram (BDD) [12] can be used to give a better representation for $Q(\mathcal{F})$ by using less variables v_i.

3 Polynomial system solving over finite fields

Let $\mathcal{F} = \{f_1, f_2, \ldots, f_r\} \subset \mathbb{F}_q[\mathbb{X}]$ be a finite set of polynomials over the finite field $\mathbb{F}_q$, $t_i = \# f_i$, and $T_{\mathcal{F}} = \sum_{i=1}^{r} t_i$. In this section, we give a quantum algorithm to find a solution of $\mathcal{F}$ in $\mathbb{F}_q^n$. Denote the solutions of $\mathcal{F}$ in $\mathbb{F}_q^n$ by $\mathbb{V}_{\mathbb{F}_q}(\mathcal{F})$. For a prime number p, we use the standard representation $\mathbb{F}_p = \{0, 1, \ldots, p-1\}$.

3.1 Reduce MQ over $\mathbb{F}_p$ to MQ in Boolean variables over $\mathbb{C}$

Let $\mathcal{F} = \{f_1, f_2, \ldots, f_r\} \subset \mathbb{F}_p[\mathbb{X}]$ be an MQ, $t_i = \# f_i$, and $T_{\mathcal{F}} = \sum_{i=1}^{r} t_i$. In this section, we will construct a set of Boolean polynomials over $\mathbb{C}$, from which we can obtain $\mathbb{V}_{\mathbb{F}_p}(\mathcal{F})$. The reduction procedure consists of the following two steps.

Step 1. We reduce $\mathcal{F}$ to a set of polynomials in Boolean variables over $\mathbb{F}_p$. If $p = 2$, then the x_i are already Boolean and we can skip this step. We thus assume $p > 2$ and set

$$\begin{aligned} x_i &= \theta_{p-1}(\mathbb{X}_i) = \sum_{j=0}^{\lfloor \log_2(p-1) \rfloor - 1} X_{i,j} 2^j + (p - 2^{\lfloor \log_2(p-1) \rfloor}) X_{i, \lfloor \log_2(p-1) \rfloor}, \qquad (5) \\ \mathbb{X}_i &= \{X_{i,j} \mid j = 0, 1, \ldots, \lfloor \log_2(p-1) \rfloor - 1\}, \\ \mathbb{X}_{\mathrm{bit}} &= \cup_{i=1}^{n} \mathbb{X}_i = \{X_{i,j} \mid i = 1, 2, \ldots, n,\ j = 0, 1, \ldots, \lfloor \log_2(p-1) \rfloor\}. \end{aligned}$$

where θ_{p-1} is defined in (2) and $X_{i,j}$ are Boolean variables. Let $f_i = \sum_{j=1}^{t_i} c_{i,j} \mathbb{X}^{\alpha_{ij}}$, where $\alpha_{ij} = (\alpha_{ij}(1), \alpha_{ij}(2), \ldots, \alpha_{ij}(n)) \in \mathbb{N}^n$. Substituting (5) into $\mathcal{F}$, we have

$$\begin{aligned} f_{i\mathrm{bit}} &= \sum_{j=1}^{t_i} c_{i,j} \prod_{k=1}^{n} (\theta_{p-1}(\mathbb{X}_i))^{\alpha_{ij}(k)} \in \mathbb{F}_p[\mathbb{X}_i], \qquad (6) \\ B(\mathcal{F}) &= \{f_{1\mathrm{bit}}, f_{2\mathrm{bit}}, \ldots, f_{r\mathrm{bit}}\} \subset \mathbb{F}_p[\mathbb{X}_{\mathrm{bit}}]. \end{aligned}$$

For any indeterminates set S, let

$$H_S = \{x^2 - x \mid x \in S\}.$$

We have

Lemma 3.1 *There is a surjective morphism* $\mathit{\Pi}_1 : \mathbb{V}_{\mathbb{F}_p}(B(\mathcal{F}), H_{\mathbb{X}_{\mathrm{bit}}}) \rightrightarrows \mathbb{V}_{\mathbb{F}_p}(\mathcal{F})$, *where* $\mathit{\Pi}_1(\mathbb{X}_{\mathrm{bit}}) = (\theta_{p-1}(\mathbb{X}_1), \theta_{p-1}(\mathbb{X}_2), \ldots, \theta_{p-1}(\mathbb{X}_n))$. *Furthermore,* $\#\mathbb{X}_{\mathrm{bit}} = O(n \log p)$ *and the total sparseness of* $B(\mathcal{F})$ *is* $O(T_{\mathcal{F}} \log^2 p)$.

Proof By Lemma 2.1, it is easy to check that $\mathit{\Pi}_1$ is surjective. Also by Lemma 2.1, $\#\theta_{p-1}(\mathbb{X}_i) = \lfloor \log_2(p-1) \rfloor + 1$ and hence $\#\mathbb{X}_{\mathrm{bit}} = O(n \log p)$. Since $\mathcal{F}$ is an MQ, for any monomial $\mathbb{X}^{\alpha_{ij}}$ of f_i, we have $|\alpha_{ij}| \le 2$ and $\prod_{k=1}^{n} (\theta_{p-1}(\mathbb{X}_i))^{\alpha_{ij}(k)}$ in (6) has at most $O(\log^2 p)$ terms. Therefore, the total sparseness of $f_{i\mathrm{bit}}$ is $O(\# f_i \log^2 p)$ and the total sparseness of $B(\mathcal{F})$ is $O(T_{\mathcal{F}} \log^2 p)$. ∎

Step 2. We introduce new Boolean indeterminates $U_{i,j}$ and reduce each $f_{i\mathrm{bit}}$ into a Boolean polynomial over $\mathbb{Z}$. Let $t_i' = \# f_{i\mathrm{bit}}$,

$$\begin{aligned} \mathbb{U}_i &= \{U_{i,j}, j = 0, 1, \ldots, \lfloor \log_2 t_i' \rfloor\}, \\ \mathbb{U}_{\mathrm{bit}} &= \cup_{i=1}^{r} \mathbb{U}_i = \{U_{i,j} \mid i = 1, 2, \ldots, r,\ j = 0, 1, \ldots, \lfloor \log_2 t_i' \rfloor\}, \qquad (7) \\ \theta_{t_i'}(\mathbb{U}_i) &= \sum_{j=0}^{\lfloor \log_2 t_i' \rfloor - 1} U_{i,j} 2^j + (t_i' + 1 - 2^{\lfloor \log_2 t_i' \rfloor}) U_{i, \lfloor \log_2 t_i' \rfloor} \in \mathbb{F}_p[\mathbb{U}_i], \qquad (8) \\ P(f_{i\mathrm{bit}}) &= f_{i\mathrm{bit}} - p\theta_{t_i'}(\mathbb{U}_i) \in \mathbb{Z}[\mathbb{X}_{\mathrm{bit}}, \mathbb{U}_i], \end{aligned}$$

$$P(\mathcal{F}) \quad = \quad \{P(f_{i\mathrm{bit}})\,|\,i=1,2,\ldots,r\} \subset \mathbb{Z}[\mathbb{X}_{\mathrm{bit}}, \mathbb{U}_{\mathrm{bit}}], \tag{9}$$

and we have

Lemma 3.2 *There is a surjective morphism $\Pi_2 : \mathbb{V}_{\mathbb{C}}(P(\mathcal{F}), H_{\mathbb{X}_{\mathrm{bit}}}, H_{\mathbb{U}_{\mathrm{bit}}}) \rightrightarrows \mathbb{V}_{\mathbb{F}_p}(\mathcal{F})$, where $\Pi_2(\mathbb{X}_{\mathrm{bit}}, \mathbb{U}_{\mathrm{bit}}) = \Pi_1(\mathbb{X}_{\mathrm{bit}}) = (\theta_{p-1}(\mathbb{X}_1), \theta_{p-1}(\mathbb{X}_2), \ldots, \theta_{p-1}(\mathbb{X}_n))$.*

Proof Let $(\check{\mathbb{X}}_{\mathrm{bit}}, \check{\mathbb{U}}_{\mathrm{bit}}) \in \mathbb{V}_{\mathbb{C}}(P(\mathcal{F}), H_{\mathbb{X}_{\mathrm{bit}}}, H_{\mathbb{U}_{\mathrm{bit}}})$. Then $(\check{\mathbb{X}}_{\mathrm{bit}}, \check{\mathbb{U}}_{\mathrm{bit}})$ is a Boolean solution of $P(\mathcal{F}) \subset \mathbb{Z}[\mathbb{X}_{\mathrm{bit}}, \mathbb{U}_{\mathrm{bit}}]$ and

$$0 = P(f_i)(\check{\mathbb{X}}_{\mathrm{bit}}, \check{\mathbb{U}}_{\mathrm{bit}}) = f_{i\mathrm{bit}}(\check{\mathbb{X}}_{\mathrm{bit}}) - p\theta_{\lfloor C_i/p \rfloor}(\check{\mathbb{U}}_{\mathrm{bit}}) \equiv f_{i\mathrm{bit}}(\check{\mathbb{X}}_{\mathrm{bit}}) \equiv f_i(\Pi_1(\check{\mathbb{X}}_{\mathrm{bit}})) \pmod p,$$

where the last equivalence comes from (6), and $\Pi_1(\check{\mathbb{X}}_{\mathrm{bit}}) = (\theta_{p-1}(\check{\mathbb{X}}_1), \theta_{p-1}(\check{\mathbb{X}}_2), \ldots, \theta_{p-1}(\check{\mathbb{X}}_n))$ is defined in Lemma 3.1. As a conclusion, $f_i(\Pi(\check{\mathbb{X}}_{\mathrm{bit}})) \equiv 0 \pmod p$, or $(\theta_{p-1}(\check{\mathbb{X}}_1), \theta_{p-1}(\check{\mathbb{X}}_2), \ldots, \theta_{p-1}(\check{\mathbb{X}}_n)) \in \mathbb{V}_{\mathbb{F}_p}(\mathcal{F})$.

We now prove that Π_2 is surjective. By Lemma 3.1, $\mathbb{V}_{\mathbb{F}_p}(B(\mathcal{F}), H_{\mathbb{X}_{\mathrm{bit}}}) \rightrightarrows \mathbb{V}_{\mathbb{F}_p}(\mathcal{F})$, so it is enough to prove $\mathbb{V}_{\mathbb{C}}(P(\mathcal{F}), H_{\mathbb{X}_{\mathrm{bit}}}, H_{\mathbb{U}_{\mathrm{bit}}}) \rightrightarrows \mathbb{V}_{\mathbb{F}_p}(B(\mathcal{F}), H_{\mathbb{X}_{\mathrm{bit}}})$. Let $\check{\mathbb{X}}_{\mathrm{bit}} \in \mathbb{V}_{\mathbb{F}_p}(B(\mathcal{F}), H_{\mathbb{X}_{\mathrm{bit}}})$ and $f_{i\mathrm{bit}} = \sum_{j=1}^{t'_i} c'_{i,j} \mathbb{X}_{\mathrm{bit}}^{\beta_{ij}} \in \mathbb{F}_p[\mathbb{X}_{\mathrm{bit}}]$, where $c'_{i,j} \in \{0, 1, \ldots, p-1\} \subset \mathbb{Z}$. Denote $C_i = \sum_{j=1}^{t'_i} c'_{i,j} \le (p-1)t'_i$. Then, $f_{i\mathrm{bit}}(\check{\mathbb{X}}_{\mathrm{bit}}) \equiv 0 \pmod p$ if and only if $f_{i\mathrm{bit}}(\check{\mathbb{X}}_{\mathrm{bit}}) = 0, p, 2p, \ldots,$ or $t'_i p$, since $\lfloor C_i/p \rfloor p \le t'_i$. By Lemma 2.1, there exist Boolean variables $\check{\mathbb{U}}_i = \{\check{U}_{i,j}, j = 0, 1, \ldots, \lfloor \log_2 t'_i \rfloor\}$ such that $f_{i\mathrm{bit}}(\check{\mathbb{X}}_{\mathrm{bit}}) = p\theta_{t'_i}(\check{\mathbb{U}}_i)$. Hence $(\check{\mathbb{X}}_{\mathrm{bit}}, \check{\mathbb{U}}_i)$ is a preimage of $\check{\mathbb{X}}_{\mathrm{bit}}$ for the map $\mathbb{V}_{\mathbb{C}}(P(\mathcal{F}), H_{\mathbb{X}_{\mathrm{bit}}}, H_{\mathbb{U}_{\mathrm{bit}}}) \rightrightarrows \mathbb{V}_{\mathbb{F}_p}(B(\mathcal{F}), H_{\mathbb{X}_{\mathrm{bit}}})$. Then, the map Π_2 is surjective. ∎

Since the map Π_1 in (5) is not injective, this map Π_2 is also not injective.

Lemma 3.3 *The polynomial system $P(\mathcal{F})$ defined in (9) is of total sparseness $T_{P(\mathcal{F})} = O(T_{\mathcal{F}} \log^2 p)$ and has $N_{P(\mathcal{F})} = O(n \log p + \sum_{i=1}^r \log t_i + r \log\log p)$ indeterminates. Furthermore, we can compute $P(\mathcal{F})$ from $\mathcal{F}$ in $\widetilde{O}(T_{\mathcal{F}} \log^2 p)$ binary operations.*

Proof By Lemma 3.1, $B(\mathcal{F})$ is of total sparseness $O(T_{\mathcal{F}} \log^2 p)$ and has $O(n \log p)$ indeterminates. Since $\mathcal{F}$ is an MQ, by the proof of Lemma 3.1, we have $t'_i = \#f_{i,\mathrm{bit}} \le t_i \log^2 p$. Then, the number of $U_{i,j}$ introduces in (7) is $\#\mathbb{U}_{\mathrm{bit}} = \sum_{i=1}^r \lfloor \log_2 t'_i \rfloor = O(\sum_{i=1}^r \log t'_i) = O(\sum_{i=1}^r (\log t_i + \log(\log p)^2)) = O(\sum_{i=1}^r \log t_i + r \log\log p)$. Therefore, the total number of indeterminates is $\#\mathbb{X}_{\mathrm{bit}} + \#\mathbb{U}_{\mathrm{bit}} = O(n \log p + \sum_{i=1}^r \log t_i + r \log\log p)$.

From (9), the total sparseness of $P(\mathcal{F})$ is $T_{P(\mathcal{F})} = T_{B(\mathcal{F})} + \sum_{i=1}^r \#\theta_{t'_i}(\mathbb{U}_i) = T_{B(\mathcal{F})} + \#\mathbb{U}_{\mathrm{bit}} = O(T_{\mathcal{F}} \log^2 p + \sum_{i=1}^r \log t_i + r \log\log p) = O(T_{\mathcal{F}} \log^2 p)$, since $r \le T_{\mathcal{F}}$ and $\sum_{i=1}^r \log t_i \le \sum_{i=1}^r t_i = T_{\mathcal{F}}$.

To compute each $2^j \mod p$ costs $O(\log p)$ binary operations. Using the fast polynomial arithmetics [36], to expand all the polynomials in $B(\mathcal{F})$ costs $\widetilde{O}(T_{\mathcal{F}} \log^2 p)$ binary operations. The cost of other steps to obtain $P(\mathcal{F})$ is negligible. ∎

Corollary 3.4 *If $\mathcal{F}$ is a linear system, then $T_{P(\mathcal{F})} = O(T_{\mathcal{F}} \log p)$ and $N_{P(\mathcal{F})} = \widetilde{O}(n \log p + \sum_{i=1}^r \log t_i + r \log\log p)$.*

Proof Since each f_i is linear, we have $T_{B(\mathcal{F})} = O(T_{\mathcal{F}} \log p)$, and $T_{P(\mathcal{F})} = O(T_{\mathcal{F}} \log p + \#\mathbb{U}_{\mathrm{bit}}) = \widetilde{O}(T_{\mathcal{F}} \log p)$. ∎

Remark 3.5 In (8), we can use $\theta_{\lfloor C_i/p \rfloor}$ instead of $\theta_{t'_i}$ to introduce less indeterminates. To compute each $C_i = \sum_{j=1}^{t'_i} c'_{ij}$ costs $t'_i \log p = O(t_i \log^3 p)$, and to compute all C_i costs $O(T_{\mathcal{F}} \log^3 p)$, which is more than $T_{P(\mathcal{F})} = O(T_{\mathcal{F}} \log^2 p)$. But, this is negligible comparing to the final complexity of the algorithm in Corollary 3.9.

3.2 Solving polynomial systems over $\mathbb{F}_p$

Let $\mathcal{F} = \{f_1, f_2, \ldots, f_r\} \subset \mathbb{F}_p[\mathbb{X}]$. By Lemma 2.3, we can convert $\mathcal{F}$ into an MQ $Q(\mathcal{F}) \subset \mathbb{F}_p[\mathbb{X}, \mathbb{V}]$. By Lemma 3.2, we can convert $Q(\mathcal{F})$ to an MQ in Boolean variables over $\mathbb{C}$: $P(Q(\mathcal{F})) \subset \mathbb{C}[\mathbb{X}_{\rm bit}, \mathbb{V}_{\rm bit}, \mathbb{U}_{\rm bit}]$. To solve $P(Q(\mathcal{F}))$, we need the following result, where a quantum algorithm for B-POSSO is given. A solution $\mathbf{a}$ of $\mathcal{B} \subset \mathbb{C}[\mathbb{X}]$ is called a *Boolean solution* if each coordinate of $\mathbf{a}$ is either 0 or 1.

Theorem 3.6 (see [13]) *For a finite set $\mathcal{B} \subset \mathbb{C}[\mathbb{X}]$ and $\varepsilon \in (0,1)$, there exists a quantum algorithm* **QBoolSol** *which decides whether $\mathcal{B} = 0$ has a Boolean solution and computes one if $\mathcal{B} = 0$ does have Boolean solutions, with probability at least $1-\varepsilon$ and complexity $\widetilde{O}(n^{2.5}(n+T_{\mathcal{B}})\kappa^2 \log 1/\varepsilon)$, where $T_{\mathcal{B}}$ the total sparseness of $\mathcal{B}$ and κ is the condition number of $\mathcal{B}$.*

Here is the main result of this section.

Theorem 3.7 *For $\mathcal{F} = \{f_1, f_2, \ldots, f_r\} \subset \mathbb{F}_p[\mathbb{X}]$ and $\varepsilon \in (0,1)$, there exists a quantum algorithm to find a solution of $\mathcal{F}$ in $\mathbb{F}_p$ with probability at least $1-\varepsilon$ and the complexity of the algorithm is $\widetilde{O}(T_{\mathcal{F}}^{3.5} D^{3.5} \log^{4.5} p\kappa^2$ $\log 1/\varepsilon)$, where $T_{\mathcal{F}} = \sum_{i=1}^r \#f_i$ is the total sparseness of $\mathcal{F}$, $D = n + \sum_{i=1}^n \max_j \lfloor \log_2(\deg_{x_i}(f_j)) \rfloor$, and κ is the condition number of $P(Q(\mathcal{F}))$, also called the condition number of $\mathcal{F}$.*

We first estimate the total sparseness of $P(Q(\mathcal{F}))$.

Lemma 3.8 *$P(Q(\mathcal{F}))$ is of total sparseness $O(T_{\mathcal{F}} D \log^2 p) = O(nT_{\mathcal{F}} \log d \log^2 p)$ and has $O(T_{\mathcal{F}}\ D$ $\log p) = O(nT_{\mathcal{F}} \log d \log p)$ indeterminates, where $D = n + \sum_{i=1}^n \max_j \lfloor \log_2(\deg_{x_i}(f_j)) \rfloor$ and $d = \max\{2, \log_2($ $\deg_{x_i}(f_j)), i = 1, 2, \ldots, n, j = 1, 2, \ldots, r\}$.*

Proof By Lemma 2.3, $N_{Q(\mathcal{F})} = O(T_{\mathcal{F}} D)$, $T_{Q(\mathcal{F})} = O(T_{\mathcal{F}} D)$, and $\#Q(\mathcal{F}) = O(T_{\mathcal{F}} D)$. By Lemma 3.3, $P(Q(\mathcal{F}))$ is of total sparseness $O(T_{\mathcal{F}} D \log^2 p)$, and $P(Q(\mathcal{F}))$ has $N_{P(Q(\mathcal{F}))} = O(N_{Q(\mathcal{F})} \log p + \sum_{f \in Q(\mathcal{F})} \log($ $\#f) + \#Q(\mathcal{F}) \log\log p)$ indeterminates. From the proof of Lemma 2.3, $Q(\mathcal{F})$ contains $\widehat{f}_j, j = 1, 2, \ldots, r$ and $\#V$ binomials. Then, $\sum_{f \in Q(\mathcal{F})} \log(\#f) = \sum_{j=1}^r \log(\#\widehat{f}_j) + \#V \log 2 = O(\sum_{j=1}^r \log t_j + T_{\mathcal{F}} D) = O(T_{\mathcal{F}} D)$. Then $N_{P(Q(\mathcal{F}))} = O(T_{\mathcal{F}} D \log p + T_{\mathcal{F}} D + T_{\mathcal{F}} D \log\log p) = O(T_{\mathcal{F}} D \log p)$. Since $D = n + \sum_{i=1}^n \max_j \lfloor \log_2(\deg_{x_i}(f_j)) \rfloor = O(n \log d)$, we obtain the bounds involving n and d. ∎

Proof of Theorem 3.7. We can find a solution of $\mathcal{F}$ as follows. Construct $P(Q(\mathcal{F})) \subset \mathbb{C}[\mathbb{X}_{\rm bit}, \mathbb{V}_{\rm bit}, \mathbb{U}_{\rm bit}]$ according to Lemma 2.3 and Lemma 3.2. Let $\mathbf{b} = \textbf{QBoolSol}(P(Q(\mathcal{F})), \varepsilon)$. If $\mathbf{b} = \emptyset$ then the algorithm fails to find a solutioon. Let $\mathbf{b} = (\check{\mathbb{X}}_{\rm bit}, \check{\mathbb{V}}_{\rm bit}, \check{\mathbb{U}}_{\rm bit})$ and $\check{\mathbb{X}}_{\rm bit} = (\check{X}_{1,0}, \check{X}_{1,1}, \ldots, \check{X}_{1,\lfloor \log_2(p-1) \rfloor}, \check{X}_{2,0}, \check{X}_{2,1}, \ldots, \check{X}_{n,\lfloor \log_2(p-1) \rfloor})$. Let $\widehat{X} = (\theta_{p-1}(\check{\mathbb{X}}_1), \theta_{p-1}(\check{\mathbb{X}}_2), \ldots, \theta_{p-1}(\check{\mathbb{X}}_n))$, where $\check{\mathbb{X}}_i = (\check{X}_{i,0}, \check{X}_{i,1}, \ldots, \check{X}_{i,\lfloor \log_2(p-1) \rfloor})$. By Lemma 2.3, Lemma 3.2, and Theorem 3.6, $\widehat{X}$ is a solution of $\mathcal{F}$ in $\mathbb{F}_p$ with probability at least $1 - \varepsilon$.

We now give the complexity. By Lemma 3.8, $P(Q(\mathcal{F}))$ is of sparseness $O(T_{\mathcal{F}} D \log^2 p)$ and has $O(T_{\mathcal{F}} D$ $\log p)$ indeterminates. By Theorem 3.6, we can find a Boolean solution of $P(Q(\mathcal{F}))$ in time $\widetilde{O}((T_{\mathcal{F}} D \log p)^{2.5}$ $(T_{\mathcal{F}} D\ \log p + T_{\mathcal{F}} D \log^2 p)\kappa^2 \log 1/\varepsilon) = \widetilde{O}(T_{\mathcal{F}}^{3.5} D^{3.5} \log^{4.5} p\kappa^2 \log 1/\varepsilon)$. The complexity for other steps can be neglected. ∎

Let $d = \max\{2, \max_{i=1}^n \max_j \deg_{x_i}(f_j)\}$. Then $D = O(n \log d)$. Since the solutions are in $\mathbb{F}_p$, we can assume $d < p$. By Theorem 3.7, we have

Corollary 3.9 *The complexity to find a solution for $\mathcal{F} = 0 \mod p$ is $\widetilde{O}(n^{3.5} T_{\mathcal{F}}^{3.5} \log^{3.5} d \log^{4.5} p\ \kappa^2$ $\log 1/\varepsilon) = \widetilde{O}(n^{3.5} T_{\mathcal{F}}^{3.5} \log^8 p\kappa^2\ \log 1/\varepsilon)$.*

Corollary 3.10 *If $\mathcal{F}$ is an* MQ*, then the complexity is $\widetilde{O}((n \log p + r)^{2.5}(n \log p + T_{\mathcal{F}} \log^2 p)\kappa^2 \log 1/\varepsilon)$.*

Proof If $\mathcal{F}$ is an MQ, we have $P(Q(\mathcal{F})) = P(\mathcal{F})$. By Lemma 3.3, $N_{P(\mathcal{F})} = O(n \log p + \sum_{i=1}^r \log t_i +$

$r\log\log p)$ and $T_{P(\mathcal{F})}=O(T_{\mathcal{F}}\log^2 p)$. Considering $\sum_{i=1}^r \log t_i=\log(\prod_{i=1}^r t_i)\le \log((\sum_{i=1}^r t_i/r)^r)=r\log(T_{\mathcal{F}}/r)$, $N_{P(\mathcal{F})}=O(n\log p+r\log T_{\mathcal{F}})$. By Theorem 3.6, the complexity is $\widetilde{O}(N_{P(\mathcal{F})}^{2.5}(N_{P(\mathcal{F})}+T_{P(\mathcal{F})})\kappa^2\log 1/\varepsilon)=\widetilde{O}((n\log p+r\log T_{\mathcal{F}})^{2.5}((n\log p+r\log T_{\mathcal{F}})+T_{\mathcal{F}}\log^2 p)\kappa^2\log 1/\varepsilon)=\widetilde{O}((n\log p+r)^{2.5}(n\log p+T_{\mathcal{F}}\log^2 p)\kappa^2\log 1/\varepsilon)$. In the last step, we here use the reduction $(a+b\log c)(c+d)\le(a+b)\log(c+d)(c+d)=\widetilde{O}((a+b)(c+d))$. Considering $n<T_{\mathcal{F}}$, the complexity is proved. ∎

Corollary 3.11 *If $p=2$, then the complexity to find a solution of $\mathcal{F}=0 \mod 2$ is $\widetilde{O}((n+r)^{2.5}(n+T_{\mathcal{F}})\kappa^2\log 1/\varepsilon)$.*

Proof If $p=2$, then we do not need to convert $\mathcal{G}$ to MQ and $T_{P(\mathcal{F})}=O(T_{\mathcal{F}}+\sum_{i=1}^r\log t_i)=O(T_{\mathcal{F}})$, $N_{P(\mathcal{F})}=O(n+\sum_{i=1}^r\log t_i)$. Similar to the proof of Corollary 3.10, the complexity is $\widetilde{O}((n+\sum_{i=1}^r\log t_i)^{2.5}(n+T_{\mathcal{F}})\kappa^2\log 1/\varepsilon)=\widetilde{O}((n+r)^{2.5}(n+T_{\mathcal{F}})\kappa^2\log 1/\varepsilon)$. ∎

3.3 Polynomial equation solving over $\mathbb{F}_q$

In this section, we consider polynomial equation solving in a general finite field $\mathbb{F}_q$ by reducing the problem to equation solving over $\mathbb{F}_p$.

If $q=p^m$ with p a prime number and $m\in\mathbb{Z}_{>1}$, then $\mathbb{F}_q=\mathbb{F}_p(\theta)$, where $\varphi(\theta)=0$ for a monic irreducible polynomial φ with $\deg(\varphi)=m$. Let $g\in\mathbb{F}_q[\mathbb{X}]=\mathbb{F}_p[\theta,\mathbb{X}]$. By setting $x_i=\sum_{j=0}^{m-1}x_{ij}\theta^j$ and write each coefficient $c=\sum_{j=0}^{m-1}c_j\theta^j$ in g, g can be written as $g=\sum_{j=0}^{m-1}g_j\theta^j$, where $g_j\in\mathbb{F}_p[\mathbb{X}_\theta]$ and $\mathbb{X}_\theta=\{x_{ij}\mid i=1,2,\ldots,n, j=0,1,\ldots,m-1\}$ are variables over $\mathbb{F}_p$. We denote $G(g)=\{g_0,g_1,\ldots,g_{m-1}\}\subset\mathbb{F}_p[\mathbb{X}_\theta]$. For a polynomial set $\mathcal{F}\subset\mathbb{F}_q[\mathbb{X}]$, we denote

$$G(\mathcal{F})=\bigcup_{f\in\mathcal{F}}G(f)\subset\mathbb{F}_p[\mathbb{X}_\theta]. \tag{10}$$

Lemma 3.12 *There is an isomorphism $\Pi_q:\mathbb{V}_{\mathbb{F}_p}(G(\mathcal{F}))\to\mathbb{V}_{\mathbb{F}_q}(\mathcal{F})$, where $\Pi_q(x_{ij})=(\sum_{j=0}^{m-1}x_{1j}\theta^j,\sum_{j=0}^{m-1}x_{2j}\theta^j,\ldots,\sum_{j=0}^{m-1}x_{nj}\theta^j)$. Furthermore, for an MQ $\mathcal{F}=\{f_1,f_2,\ldots,f_r\}\subset\mathbb{F}_q[\mathbb{X}]$ with total sparseness $T_{\mathcal{F}}$, $G(\mathcal{F})\subset\mathbb{F}_p[\mathbb{X}_\theta]$ is an MQ with total sparseness $\le m^3T_{\mathcal{F}}$, $\#G(\mathcal{F})=mr$, and $\#\mathbb{X}_\theta=mn$.*

Proof It is easy to show that $\#G(\mathcal{F})=m\#\mathcal{F}$, $\#\mathbb{X}_\theta=m\#\mathbb{X}$ and $G(\mathcal{F})$ is also an MQ. Then the total sparseness of $G(\mathcal{F})$ will be concerned. $\mathcal{F}$ has $T_{\mathcal{F}}$ terms, where each term is of degree ≤ 2. For $x=\sum_{i=0}^{m-1}x_i\theta^i$ and $y=\sum_{i=0}^{m-1}y_i\theta^i$, let $c\theta^k=\sum_{j=0}^{m-1}c_{jk}\theta^j \mod \varphi(\theta)$ for any $k\in\mathbb{N}$, then we have $cxy=\sum_{i=0}^{m-1}\sum_{j=0}^{m-1}x_iy_j\sum_{k=0}^{m-1}c_{k\,(i+j)}\theta^k=\sum_{i=0}^{m-1}g_i\theta^i$, where $g_i\in\mathbb{F}_p[x_0,x_1,\ldots,x_{m-1},y_0,y_1,\ldots,y_{m-1}]$ is a quadratic polynomial with $T_{G(cxy)}\le\sum_{i=0}^{m-1}\sum_{j=0}^{m-1}\sum_{k=0}^{m-1}1=m^3$. Thus an MQ $\mathcal{F}$ over $\mathbb{F}_q$ can be represented as another MQ $G(\mathcal{F})$ over $\mathbb{F}_p$ with $T_{G(\mathcal{F})}\le m^3T_{\mathcal{F}}$. ∎

We have

Theorem 3.13 *There is a quantum algorithm to find a solution of $\mathcal{F}\subset\mathbb{F}_q[\mathbb{X}]$ with probability at least $1-\varepsilon$ and in time $\widetilde{O}(m^{5.5}T_{\mathcal{F}}^{3.5}D^{3.5}\log^{4.5}p\kappa^2\log 1/\varepsilon)$, where $T_{\mathcal{F}}$ is the total sparseness of $\mathcal{F}$, $D=n+\sum_{i=1}^n\lfloor\log_2\max_j\deg_{x_i}f_j\rfloor$, and κ is the condition number of $\mathcal{F}$, defined as the condition number of the Macaulay matrix [13] of $P(G(Q(\mathcal{F})))$.*

Proof Using Lemma 3.12, we can solve $\mathcal{F}$ over $\mathbb{F}_q$ similar to the method given in the proof of Theorem 3.7. Rather than solving $\mathcal{F}_1=P(Q(\mathcal{F}))\subset\mathbb{C}[\mathbb{X}_{\rm bit},\mathbb{V}_{\theta{\rm bit}},\mathbb{U}_{\theta{\rm bit}}]$ with Algorithm **QBoolSol**, we now solve $\mathcal{F}_1=P(G(Q(\mathcal{F})))\subset\mathbb{C}[\mathbb{X}_{\theta{\rm bit}},\mathbb{V}_{\theta{\rm bit}},\mathbb{U}_{\theta{\rm bit}}]$ with algorithm **QBoolSol**, where $Q(\mathcal{F})$ is defined in (10) and $\mathbb{X}_{\theta{\rm bit}}$ is the bit representation for $\mathbb{X}_\theta$.

We now prove the complexity. By Lemma 2.3, $Q(\mathcal{F})$ has $O(T_{\mathcal{F}}D)$ indeterminates and total sparseness $O(T_{\mathcal{F}}D)$. $G(Q(\mathcal{F}))$ has $O(mT_{\mathcal{F}}D)$ indeterminates and total sparseness $O(m^3T_{\mathcal{F}}D)$. Since $G(Q(\mathcal{F}))$ is an MQ, by Corollary 3.10, the complexity is $\widetilde{O}(((mT_{\mathcal{F}}D\log p + m(T_{\mathcal{F}}D))^{2.5}(m^3T_{\mathcal{F}}D)\ \log^2 p\kappa^2 \log 1/\varepsilon) = \widetilde{O}(m^{5.5}T_{\mathcal{F}}^{3.5}D^{3.5}\log^{4.5} p\kappa^2\ \log 1/\varepsilon)$. ∎

Corollary 3.14 *If $\mathcal{F}$ is an* MQ*, the complexity is* $\widetilde{O}(m^{3.5}(n\log p+r)^{2.5}(n\log p+m^2T_{\mathcal{F}}\log^2 p)\kappa^2\log 1/\varepsilon)$.

Corollary 3.15 *If $q = 2^m$, then the complexity is $\widetilde{O}(m^{5.5}T_{\mathcal{F}}^{3.5}D^{3.5}\kappa^2\log 1/\varepsilon)$. Moreover, if $\mathcal{F}\subset\mathbb{F}_{2^m}[\mathbb{X}]$ is an* MQ*, then the complexity is $\widetilde{O}(m^{3.5}(n+r)^{2.5}(n+m^2T_{\mathcal{F}})\kappa^2\log 1/\varepsilon)$.*

4 Reduce inequalities to MQ in Boolean variables

In this section, we show how to reduce the inequality constraints $\mathcal{I} = \{0\le g_i(\mathbb{X},\mathbb{Y})\le b_i, i=1,2,\ldots,s; 0\le y_k\le u_k, k=1,2,\ldots,m; \mathbb{X}\in\mathbb{F}_p^n; \mathbb{Y}\in\mathbb{Z}^m\}$ of problem (1) into a B-POSSO, where $g_1,g_2,\ldots,g_s\in\mathbb{Z}[\mathbb{X}]$, $b_1,b_2,\ldots,b_s,u_1,u_2,\ldots,u_m\in\mathbb{N}$. We emphasize that for $g\in\mathbb{C}[\mathbb{X},\mathbb{Y}]$, $\check{\mathbb{X}}\in\mathbb{F}_p^n$, and $\check{\mathbb{Y}}\in\mathbb{Z}^m$, $g(\check{\mathbb{X}},\check{\mathbb{Y}})$ is evaluated in $\mathbb{C}$.

4.1 Reduce polynomial system over $\mathbb{C}$ to MQ in Boolean variables over $\mathbb{C}$

Let $\mathcal{G} = \{g_1,g_2,\ldots,g_s\}\subset\mathbb{Z}[\mathbb{X},\mathbb{Y}]$. We will reduce $\mathcal{G}$ into an equivalent MQ in Boolean variables over $\mathbb{C}$ under the condition $x_i\in\mathbb{F}_p=\{0,1,\ldots,p-1\}$ and $0\le y_j\le u_j$. Let $d_g=\max_{l=1}^s\deg(g_l)$.

Following Lemma 2.3, let $Q(\mathcal{G})\subset\mathbb{Z}[\mathbb{X},\mathbb{Y},\mathbb{V}]$ be the MQ defined in (3), where $\mathbb{V}$ is the set of new indeterminates introduced in Lemma 2.3. We will reduce $\mathbb{X}$, $\mathbb{Y}$, and $\mathbb{V}=\{v_1,v_2,\ldots,v_l\}$ to Boolean variables. For $\mathbb{X}$, we use (5) to rewrite them as Boolean variables $\mathbb{X}_{\text{bit}}$. For $\mathbb{Y}$, using Lemma 2.1, the integers y_i satisfying $0\le y_i\le u_i$ can be represented exactly as follows.

$$\begin{aligned} y_i &= \theta_{u_i}(\mathbb{Y}_i) = \sum_{j=0}^{\lfloor\log_2 u_i\rfloor-1} Y_{i,j}2^j + (u_i - 2^{\lfloor\log_2 u_i)\rfloor}+1)Y_{i,\lfloor\log_2 u_i\rfloor},\\ \mathbb{Y}_i &= \{Y_{i,j}\,|\,j=0,1,\ldots,\lfloor\log_2 u_i\rfloor\},\\ \mathbb{Y}_{\text{bit}} &= \cup_{i=1}^m\mathbb{Y}_i \end{aligned} \tag{11}$$

where $Y_{i,j}$ are Boolean variables.

From Lemma 2.3, each $v_i\in\mathbb{V}_k$ represents a monomial in $\mathbb{X}$ and $\mathbb{Y}$ of degree $\le d_g$. So, $0\le v_i\le h^{d_g}$ for $h=\max\{p-1,u_1,u_2,\ldots,u_m\}$. By Lemma 2.1, we can write v_i as

$$v_i = \theta_{h^{d_g}}(\mathbb{V}_{i,\text{bit}}) = \sum_{j=0}^{\lfloor d_g\log_2 h\rfloor-1} V_{i,j}2^j + (h^{d_g} - 2^{\lfloor d_g\log_2 h\rfloor}+1)V_{i,\lfloor d_g\log_2 h\rfloor}, \tag{12}$$

where $\mathbb{V}_{i,\text{bit}}=\{V_{i,j}\,|\,j=0,1,\ldots,\lfloor d_g\log_2 u_i\rfloor\}$, $\mathbb{V}_{\text{bit}}=\cup_{i=1}^t\mathbb{V}_{i,\text{bit}}$, and each $V_{i,j}$ is a Boolean variable.

Let $\widehat{g}_k=\widehat{Q}(g_k)\in\mathbb{Z}[\mathbb{X},\mathbb{Y},\mathbb{V}]$ be defined in (4), $\widehat{Q}(\mathcal{G})=Q(\mathcal{G})\setminus\{\widehat{g}_1,\widehat{g}_2,\ldots,\widehat{g}_s\}$. Substituting x_i in (5), and y_i in (11), and v_i in (12) into $Q(\mathcal{G})$, $\widehat{g}_k$, and $\widehat{Q}(\mathcal{G})$, we obtain

$$B(\mathcal{G}), \overline{g}_k, \overline{B}(\mathcal{G}) \text{ in } \mathbb{Z}[\mathbb{X}_{\text{bit}},\mathbb{Y}_{\text{bit}},\mathbb{V}_{\text{bit}}]. \tag{13}$$

The following result shows that $\mathcal{G}$ and $B(\mathcal{G})$ are equivalent.

Lemma 4.1 *For $\check{\mathbb{X}}\in\mathbb{F}_p^n$ and $\check{\mathbb{Y}}\in\mathbb{Z}^n$ such that $0\le\check{y}_j\le u_j$ for each j, there exists a $\check{\mathbb{V}}_{\text{bit}}$ such that $g_k(\check{\mathbb{X}},\check{\mathbb{Y}})=\overline{g}_k(\check{\mathbb{X}}_{\text{bit}},\check{\mathbb{Y}}_{\text{bit}},\check{\mathbb{V}}_{\text{bit}})$ for $k=1,2,\ldots,s$ and $\overline{B}(\mathcal{G})(\check{\mathbb{X}}_{\text{bit}},\check{\mathbb{Y}}_{\text{bit}},\check{\mathbb{V}}_{\text{bit}})=0$.*

Proof From (3), it is easy to see that starting from $\check{\mathbb{X}}\in\mathbb{F}_p^n$ and $\check{\mathbb{Y}}\in\mathbb{Z}^m$, one may obtain a unique $\check{\mathbb{V}}$ such that $\widetilde{B}(g_k)(\check{\mathbb{X}},\check{\mathbb{Y}},\check{\mathbb{V}})=0$ and $g_k(\check{\mathbb{X}},\check{\mathbb{Y}})=\widehat{g}_k(\check{\mathbb{X}},\check{\mathbb{Y}},\check{\mathbb{V}})$ for each k. It suffices to show that $\check{\mathbb{X}},\check{\mathbb{Y}},\check{\mathbb{V}}$ can be written in their Boolean forms, which is valid for $\check{\mathbb{X}}, \check{\mathbb{Y}}$ due to (5) and (11) and Lemma 2.1. From Lemma 2.3, each $v_i \in \mathbb{V}$ is a monomial in $\mathbb{X}$ and $\mathbb{Y}$ of degree $\le d_g$. So, $0 \le v_i \le h^{d_g}$ for $h = \max\{p-1, u_1, u_2, \ldots, u_m\}$. By Lemma 2.1, there exists a $\check{V}_{\text{bit}}$ such that $g_k(\check{\mathbb{X}}, \check{\mathbb{Y}}) = \widehat{g}_k(\check{\mathbb{X}}, \check{\mathbb{Y}}, \check{\mathbb{V}}) = \overline{g}_k(\check{\mathbb{X}}_{\text{bit}}, \check{\mathbb{Y}}_{\text{bit}}, \check{\mathbb{V}}_{\text{bit}})$ and $\overline{B}(g)(\check{\mathbb{X}}_{\text{bit}}, \check{\mathbb{Y}}_{\text{bit}}, \check{\mathbb{V}}_{\text{bit}}) = 0$. ▮

Lemma 4.2 *$B(\mathcal{G}) = \{\overline{g}_k\} \cup \overline{B}(\mathcal{G})$ defined in (13) has $O((m+n)T_{\mathcal{G}} d_g \log d_g \log h)$ number of variables and total sparseness $O((m+n)T_{\mathcal{G}} d_g^2 \log d_g \log^2 h)$, and $C(B(\mathcal{G}))$ is $O(C(\mathcal{G}) + d_g \log h)$, where $h = \max\{p-1, u_1, u_2, \ldots, u_m\}$ and $d_g = \max_{l=1}^s \deg(g_l)$.*

Proof By Lemma 2.3, $N_{Q(\mathcal{G})} = O((m+n)T_{\mathcal{G}} \log d_g)$, $T_{Q(\mathcal{G})} = O((m+n)T_{\mathcal{G}} \log d_g)$, and $C(Q(\mathcal{G})) = C(\mathcal{G})$. Note that $|\mathbb{X}| = n$, $|\mathbb{Y}| = m$, and $|\mathbb{V}|$ is bounded by $N_{Q(\mathcal{G})} = O((m+n)T_{\mathcal{G}} \log d_g)$. By (5), $|\mathbb{X}_{\text{bit}}| = O(n \log p) = O(n \log h)$. By (11), $|\mathbb{Y}_{\text{bit}}| = O(m \log h)$. By (12) and Lemma 2.1, $|\overline{\mathbb{V}}_{\text{bit}}| = O((m+n)T_{\mathcal{G}} \log d_g \log h^{d_g})) = O((m+n)d_g T_{\mathcal{G}} \log d_g \log h))$. By (14) and Lemma 2.1, we have $\#\mathbb{G}_{\text{bit}} = O(s \log b)$. Since $s \le T_{\mathcal{G}}$, $p-1, b \le h$, the number of Boolean variables are $N_{B(\mathcal{G})} = O((m+n)d_g T_{\mathcal{G}} \log d_g \log h)$.

Note that monomials of $Q(\mathcal{G})$ are of the form $x_i x_j, x_i y_j, x_i v_j, y_i y_j, y_i v_j$, or $v_i v_j$ when we rewrite them as Boolean variables, the sparseness of the new expressions are bounded by $\log^2 p$, $\log p \log h$, $d_g \log p \log h$, $\log^2 h$, $d_g \log^2 h$ and $d_g^2 \log^2 h$, respectively. The total sparseness of $B(\mathcal{G})$ is $O((m+n)T_{\mathcal{G}} d_g^2 \log d_g \log^2 h)$.

From (12) and the fact that $B(\mathcal{G})$ is MQ, the bit size of the coefficients of $B(\mathcal{G})$ is $O(C(\mathcal{G}) + d_g \log h)$. ▮

Remark 4.3 For inequalities involving variables over finite fields, the solution of the inequalities depends on the representation of $\mathbb{F}_p$. For the general optimization problem 1, we just use standard representation for $\mathbb{F}_p$. For specific problems, such as the SIS problem in Section 7, we use different representations for $\mathbb{F}_p$ to find the "correct" solution.

4.2 Reduce inequalities into MQ in Boolean variables

We now consider the inequality constraints of problem (1): $\mathcal{I} = \{0 \le g_i(\mathbb{X}, \mathbb{Y}) \le b_i, i = 1, 2, \ldots, s; 0 \le y_k \le u_k, k = 1, 2, \ldots, m; \mathbb{X} \in \mathbb{F}_p^n\}$, where $g_1, g_2, \ldots, g_s \in \mathbb{Z}[\mathbb{X}, \mathbb{Y}]$, $b_1, b_2, \ldots, b_s, u_1, u_2, \ldots, u_m \in \mathbb{N}$. We will reduce $\mathcal{I}$ into an MQ in Boolean variables. Let

$$\begin{aligned}
\mathbb{G}_i &= \{G_{i,k} \mid k = 0, 1, \ldots, \lfloor \log_2 b_i \rfloor\}, \mathbb{G}_{\text{bit}} = \cup_{i=1}^s \mathbb{G}_i, \\
\delta(g_i) &= \theta_{b_i}(\mathbb{G}_i) - \overline{g}_i = \sum_{k=0}^{\lfloor \log_2 b_i \rfloor - 1} G_{i,k} 2^k + (b_i - 2^{\lfloor \log_2 b_i \rfloor} + 1) G_{i, \lfloor \log_2 b_i \rfloor} - \overline{g}_i \qquad (14) \\
I(\mathcal{I}) &= \{\delta(g_1), \delta(g_2), \ldots, \delta(g_s)\} \cup \overline{B}(\mathcal{G}) \subset \mathbb{Z}[\mathbb{X}_{\text{bit}}, \mathbb{Y}_{\text{bit}}, \mathbb{V}_{\text{bit}}, \mathbb{G}_{\text{bit}}]
\end{aligned}$$

where $G_{i,k}$ are Boolean variables, $\overline{g}_i$ and $\overline{B}(\mathcal{G})$ are defined in (13). We summarize the result of this section as the following result.

Lemma 4.4 *$\check{\mathbb{X}} \in \mathbb{F}_p^n$ and $\check{\mathbb{Y}} \in \mathbb{Z}^m$ satisfy the constraint $\mathcal{I}$ if and only if there exist Boolean values $\check{\mathbb{V}}_{\text{bit}}, \check{\mathbb{G}}_{\text{bit}}$ such that $(\check{\mathbb{X}}_{\text{bit}}, \check{\mathbb{Y}}_{\text{bit}}, \check{\mathbb{V}}_{\text{bit}}, \check{\mathbb{G}}_{\text{bit}})$ is a solution of $I(\mathcal{I})$.*

Proof By Lemma 2.1, $0 \le \overline{g}_i(\mathbb{X}_{\text{bit}}, \mathbb{Y}_{\text{bit}}, \mathbb{V}_{\text{bit}}) \le b_i$ if and only if $\exists \mathbb{G}_{\text{bit}}$ such that $\delta(g_i)(\mathbb{X}_{\text{bit}}, \mathbb{Y}_{\text{bit}}, \mathbb{V}_{\text{bit}}, \mathbb{G}_{\text{bit}}) = 0$. Then, the lemma is a consequence of Lemma 4.1. ▮

We now estimate the parameters of $I(\mathcal{I})$. Let $b = \max_{i=1}^s b_i$, $d_g = \max_{i=1}^s \deg(g_i)$, $h = \max\{p-1, b, u_1, u_2, \ldots, u_m\}$, $\mathcal{G} = \{g_1, g_2, \ldots, g_s\}$ and $T_{\mathcal{G}} \ge s$ the total sparseness of $\mathcal{G}$. Then, we have

Lemma 4.5 *$I(\mathcal{I})$ has $O((m+n)T_{\mathcal{G}} d_g \log d_g \log h)$ variables and total sparseness $O((m+n)T_{\mathcal{G}} d_g^2 \log d_g \log^2 h)$. $C(I(\mathcal{I}))$ is $O(C(\mathcal{G}) + d_g \log h)$.*

Proof Since $B(\mathcal{G}) = \{\overline{g}_k\} \cup \overline{B}(\mathcal{G})$, from (14), $N_{I(\mathcal{G})} = N_{B(\mathcal{G})} + \#\mathbb{G}_{\text{bit}}$, $T_{I(\mathcal{G})} = T_{B(\mathcal{G})} + \sum_i \#\theta_{b_i}(\mathbb{G}_i) = T_{B(\mathcal{G})} + \#\mathbb{G}_{\text{bit}}$, and $C(I(\mathcal{G})) = C(B(\mathcal{G}))$. Note that $\#\mathbb{G}_{\text{bit}} = O(s \log b)$. Since $T_{\mathcal{G}} \geq s$, $\#\mathbb{G}_{\text{bit}}$ is negligible comparing to the complexity of $B(\mathcal{G})$ and the lemma follows directly from Lemma 4.2. ∎

From Lemma 4.5, the total sparseness and the coefficients of $I(\mathcal{G})$ are well controlled.

Corollary 4.6 *If g_i are linear, then $I(\mathcal{G}) \subset \mathbb{Z}[\mathbb{X}_{\text{bit}}, \mathbb{Y}_{\text{bit}}, \mathbb{G}_{\text{bit}}]$, $T_{B(\mathcal{G})} = O(T_{\mathcal{G}} \log h)$, and $N_{B(\mathcal{G})} = (n+m)\log h$. Furthermore, $T_{I(\mathcal{I})} = O(T_{\mathcal{G}} \log h + s \log b)$ and $N_{I(\mathcal{I})} = O((n+m)\log h + s \log b)$.*

Proof Since each g_i is linear, we have $\widehat{Q}(f_i) = f_i$. Then the variable $V_{k,i,j}$ are not needed and $\overline{B}(\mathcal{G})$ has $(n+m)\log h$ indeterminates. Also, $T_{\overline{B}(\mathcal{G})} = O(T_{\mathcal{G}} \log h)$. The results for $I(\mathcal{G})$ can be proved similarly. ∎

4.3 Bounded integer solutions of polynomial inequalities and equations

As a direct application of the reduction method given in this section, we can give a quantum algorithm to find a feasible solution to the inequality constraint $\mathcal{I} = \{0 \leq g_i(\mathbb{X}, \mathbb{Y}) \leq b_i, i = 1, 2, \ldots, s; 0 \leq y_k \leq u_k, k = 1, 2, \ldots, m; \mathbb{X} \in \mathbb{F}_p^n\}$, where $g_1, g_2, \ldots, g_s \in \mathbb{Z}[\mathbb{X}, \mathbb{Y}]$, $b_1, b_2, \ldots, b_s, u_1, u_2, \ldots, u_m \in \mathbb{N}$. Using the notation in Lemma 4.5, we have

Proposition 4.7 *For $\varepsilon \in (0, 1)$, there is a quantum algorithm to compute a feasible solution to $\mathcal{I}$ with probability $> 1 - \varepsilon$ and in time $\widetilde{O}((m+n)^{3.5} T_{\mathcal{G}}^{3.5} d_g^{4.5} \log^{4.5} h \kappa^2 \log 1/\varepsilon)$, where κ is the condition number of $I(\mathcal{I})$ defined in (14).*

Proof By Lemma 4.4, to find a feasible solution to $\mathcal{I}$, we only need to find a Boolean solution of $I(\mathcal{I})$. By Lemma 4.5, $N_{I(\mathcal{I})} = O((m+n)T_{\mathcal{G}} d_g \log d_g \log h)$, $T_{I(\mathcal{I})} = O((m+n)T_{\mathcal{G}} d_g^2 \log d_g \log^2 h)$. Since $N_{I(\mathcal{I})} < T_{I(\mathcal{I})}$, by Theorem 3.6, the complexity to find a Boolean solution of $I(\mathcal{I})$ is $\widetilde{O}(N_{I(\mathcal{I})}^{2.5} T_{I(\mathcal{I})} \kappa^2 \log 1/\varepsilon) = \widetilde{O}((m+n)^{3.5} T_{\mathcal{G}}^{3.5} d_g^{4.5} \log^{4.5} h \kappa^2 \log 1/\varepsilon)$. ∎

A closely related problem is to find bounded integer solutions of a polynomial system over $\mathbb{Z}$.

Proposition 4.8 *Let $\mathcal{G} = \{g_1, g_2, \ldots, g_s\} \subset \mathbb{Z}[\mathbb{Y}]$ and $\varepsilon \in (0, 1)$. There is a quantum algorithm to compute an integer solution $\mathbf{b} = (b_1, b_2, \ldots, b_m)$ of $\mathcal{G} = 0$ satisfying $0 \leq b_i \leq u_i$ for each i with probability $> 1 - \varepsilon$ and in time $\widetilde{O}(m^{3.5} T_{\mathcal{G}}^{3.5} d_g^{4.5} \log^{4.5} h \kappa^2 \log 1/\varepsilon)$, where κ is the condition number of $B(\mathcal{G})$ to be defined in the proof and $h = \max_i u_i$.*

Proof By Lemma 4.1, to find an integer solution to $\mathcal{G} = 0$, we need just to find a Boolean solution of $B(\mathcal{G})$ defined in (13). By Lemma 4.2, we have $N_{B(\mathcal{G})} = O(mT_{\mathcal{G}} d_g \log d_g \log h)$ and $T_{B(\mathcal{G})} = O(mT_{\mathcal{G}} d_g^2 \log d_g \log^2 h)$. Since $N_{B(\mathcal{G})} < T_{B(\mathcal{G})}$, by Theorem 3.6, the complexity to find a Boolean solution of $B(\mathcal{G})$ is $\widetilde{O}(N_{B(\mathcal{G})}^{2.5} T_{B(\mathcal{G})} \kappa^2 \log 1/\varepsilon) = \widetilde{O}(m^{3.5} T_{\mathcal{G}}^{3.5} d_g^{4.5} \log^{4.5} h \kappa^2 \log 1/\varepsilon)$. ∎

For a general polynomial system in $\mathbb{C}[\mathbb{X}]$, the bound for the coordinates of the solutions could be double-exponential, as shown by the following example.

Example 4.9 For $\mathcal{F} = \{x_1 - 2, x_2 - x_1^2, x_3 - x_2^2, \ldots, x_n - x_{n-1}^2\} \subset \mathbb{C}[\mathbb{X}]$, $\mathbb{V}_{\mathbb{C}}(\mathcal{F}) = \{(2, 2^2, 2^4, \ldots, 2^{2^{n-1}})\}$.

On the other hand, the isolated solutions of a polynomial system is at most double-exponential [40, p. 341]. In a similar way, it is also possible to find bounded rational solutions of a polynomial system.

5 Optimization over finite fields

5.1 A quantum algorithm for the optimization problem

In this section, we give a quantum algorithm to solve the optimization problem (1). The idea is to search for the minimal value of the objective function by solving several B-POSSOs, which will be done in four steps.

Step 1. By Lemmas 2.3 and 3.2, we reduce the equational constraints $f_j(\mathbb{X}) = 0 \mod p, j = 1, 2, \ldots, r$ to an MQ in Boolean variables over $\mathbb{C}$: $\mathcal{F}_1 = P(Q(\mathcal{F})) \subset \mathbb{C}[\mathbb{X}_{\rm bit}, \mathbb{V}_{\rm 1bit}, \mathbb{U}_{\rm bit}]$.

Step 2. By Lemma 4.4, we reduce the inequality constraints $\mathcal{I} = \{0 \le g_i(\mathbb{X}, \mathbb{Y}) \le b_i, i = 1, 2, \ldots, s\}$ to an MQ in Boolean variables over $\mathbb{C}$: $\mathcal{G}_1 = I(\mathcal{I}) \subset \mathbb{C}[\mathbb{X}_{\rm bit}, \mathbb{Y}_{\rm bit}, \mathbb{V}_{\rm 2bit}, \mathbb{G}_{\rm bit}]$.

Step 3. Applying Lemma 4.1 to the objective function $o(\mathbb{X}, \mathbb{Y})$, we can reduce o to a quadratic polynomial in Boolean variables $\overline{o} \in \mathbb{C}[\mathbb{X}_{\rm bit}, \mathbb{Y}_{\rm bit}, \mathbb{V}_{\rm 3bit}]$ and an MQ $\mathcal{G}_2 = \overline{B}(\{o\}) \subset \mathbb{Z}[\mathbb{X}_{\rm bit}, \mathbb{Y}_{\rm bit}, \mathbb{V}_{\rm 3bit}]$ defined in (13). For simplicity of presentation, we denote $\mathbb{V}_{\rm bit} = \mathbb{V}_{\rm 1bit} \cup \mathbb{V}_{\rm 2bit} \cup \mathbb{V}_{\rm 3bit}$. Let

$$\mathcal{C} \quad = \quad \mathcal{F}_1 \cup \mathcal{G}_1 \cup \mathcal{G}_2 \subset \mathbb{C}[\mathbb{X}_{\rm bit}, \mathbb{Y}_{\rm bit}, \mathbb{V}_{\rm bit}, \mathbb{U}_{\rm bit}, \mathbb{G}_{\rm bit}]. \tag{15}$$

A $(0, 1)$-programming is an optimization problem where all the arguments take values of 0 or 1. By Lemmas 2.3, 3.2, 4.1, and 4.4, we have

Lemma 5.1 *Problem (1) is equivalent to the following nonlinear $(0, 1)$-programming problem*

$$\min_{\mathbb{W}_{\rm bit}} \overline{o}(\mathbb{W}_{\rm bit}) \qquad \textit{subject to } \mathcal{C}(\mathbb{W}_{\rm bit}) = 0 \tag{16}$$

where $\mathbb{W}_{\rm bit} = (\mathbb{X}_{\rm bit}, \mathbb{Y}_{\rm bit}, \mathbb{V}_{\rm bit}, \mathbb{U}_{\rm bit}, \mathbb{G}_{\rm bit})$ and $\mathcal{C}$ is defined in (15).

Step 4. The basic idea to search for a minimal value of the objective function is as follows. Since all variables are bounded, the objective function is also bounded, so we may assume $\alpha \le \overline{o}(\check{\mathbb{Z}}_{\rm bit}) < \mu$ for some $\alpha, \mu \in \mathbb{N}$. We divide $[\alpha, \mu)$ into two roughly equal parts: $[\alpha, \alpha + 2^\beta)$ and $[\alpha + 2^\beta, \mu)$ and solve the following decision problem

$$\exists \mathbb{W}_{\rm bit}(\overline{o}(\mathbb{W}_{\rm bit}) \in [\alpha, \alpha + 2^\beta) \text{ and } (\mathcal{C}(\mathbb{W}_{\rm bit}) = 0)). \tag{17}$$

Let

$$\delta_{\alpha\beta}(\overline{o}) \quad = \quad \alpha + \sum_{j=0}^{\beta-1} F_j 2^j - \overline{o}(\mathbb{W}_{\rm bit}) \in \mathbb{Z}[\mathbb{Z}_{\rm bit}], \tag{18}$$

$$L_{\alpha\beta} \quad = \quad \mathcal{C} \cup \{\delta_{\alpha\beta}(\overline{o})\} \subset \mathbb{Z}[\mathbb{Z}_{\rm bit}], \tag{19}$$

where $\mathbb{Z}_{\rm bit} = \mathbb{W}_{\rm bit} \cup \mathbb{F}_{\rm bit} = \{\mathbb{X}_{\rm bit}, \mathbb{Y}_{\rm bit}, \mathbb{V}_{\rm bit}, \mathbb{U}_{\rm bit}, \mathbb{G}_{\rm bit}, \mathbb{F}_{\rm bit}\}$ and $\mathbb{F}_{\rm bit} = \{f_0, f_1, \ldots, F_{\beta-1}\}$ are Boolean variables. By Lemma 2.1, we have

Lemma 5.2 *Problem (17) has a solution $\check{\mathbb{W}}_{\rm bit}$ if and only if $L_{\alpha\beta} = 0$ has a solution $\check{\mathbb{Z}}_{\rm bit} = (\check{\mathbb{W}}_{\rm bit}, \check{\mathbb{F}}_{\rm bit})$.*

If the answer to Problem (17) is yes, we repeat the procedure for the new feasible interval $[\alpha, \overline{o}(\check{\mathbb{Z}}_{\rm bit}))$. If the answer is no, we repeat the procedure for the new feasible interval $[\alpha + 2^\beta, \mu)$. The procedure ends when $\mu = \alpha + 1$.

We now give Algorithm 5.3 to solve problem (1). For convenience in later usage, we add a new constraint $0 \le o < u$ for a given $u \in \mathbb{N}{>}\ 0$.

Algorithm 5.3 (QFpOpt)

Input: Problem (1), $\varepsilon \in (0,1)$, and a $u \in \mathbb{Z}_{>0}$ such that $0 \le o < u$.

Output: $\widehat{o}, \check{\mathbb{X}} \in \mathbb{F}_p^n$, and $\check{\mathbb{Y}} \in \mathbb{Z}^m$ such that $\widehat{o} = o(\check{\mathbb{X}}, \check{\mathbb{Y}})$ is the minimal value of o, or "fail".

Step 1: Set $\alpha = 0, \mu = u$.

Step 2: Compute $\mathcal{C}$ in (15).

Step 3: Let $\beta = \lfloor \log_2(\mu - \alpha) \rfloor - 1$ and compute $L_{\alpha\beta} \subset \mathbb{C}[\mathbb{Z}_{\text{bit}}]$ defined in (19).

Step 4: Let $\check{\mathbb{Z}}_{\text{bit}} = \textbf{QBoolSol}(L_{\alpha\beta}, \varepsilon/\log_{4/3} u)$, where **QBoolSol** is from Theorem 3.6.

Step 5: If Algorithm **QBoolSol** returns a solution: $\check{\mathbb{Z}}_{\text{bit}} = \{\check{\mathbb{X}}_{\text{bit}}, \check{\mathbb{Y}}_{\text{bit}}, \check{\mathbb{V}}_{\text{bit}}, \check{\mathbb{U}}_{\text{bit}}, \check{\mathbb{G}}_{\text{bit}}, \check{\mathbb{F}}_{\text{bit}}\}$, then

- **Step 5.1:** Compute $\check{\mathbb{X}}$ and $\check{\mathbb{Y}}$ from $\check{\mathbb{X}}_{\text{bit}}$ and $\check{\mathbb{Y}}_{\text{bit}}$ according to (5) and (11), respectively.
- **Step 5.2:** If $\check{\mathbb{F}}_{\text{bit}} = \mathbf{0}$, return α, $\check{\mathbb{X}}$ and $\check{\mathbb{Y}}$.
- **Step 5.3:** If $\check{\mathbb{F}}_{\text{bit}} \neq \mathbf{0}$, let $\mu = \overline{o}(\check{\mathbb{Z}}_{\text{bit}})$ and goto Step 3.

Step 6: If $\check{\mathbb{Z}}_{\text{bit}} = \emptyset$, then

- **Step 6.1:** If $\mu - \alpha > 1$, let $\alpha = \alpha + 2^\beta$, and goto Step 3.
- **Step 6.2:** If $\mu - \alpha = 1$ and $\mu \neq u$, return μ, $\check{\mathbb{X}}$ and $\check{\mathbb{Y}}$.
- **Step 6.3:** If $\mu - \alpha = 1$ and $\mu = u$, return "fail".

Let $b = \max_{i=1}^s b_i$, $d_f = \max_{i,j}\{2, \deg(f_i, x_j)\}$, $d_g = \max_{i,j}\{2, \deg(g_i, x_j)\}$, $h = \max\{p-1, u_1, u_2, \ldots, u_m\}$, and $\mathcal{G}_o = \{o, g_1, g_2, \ldots, g_s\}$. Then, we have

Theorem 5.4 *Algorithm 5.3 gives a solution to problem (1) with constraint $0 \le o < u$ with success probability $\ge 1 - \varepsilon$ and in complexity $\widetilde{O}(N_{L_{\alpha\beta}}^{2.5} T_{L_{\alpha\beta}} \kappa^2 \log(1/\varepsilon) \log u)$, where*

$$\begin{aligned} N_{L_{\alpha\beta}} &= \widetilde{O}(nT_{\mathcal{F}} \log d_f \log p + (m+n)T_{\mathcal{G}_o} d_g \log h + \log u), \\ T_{L_{\alpha\beta}} &= \widetilde{O}(nT_{\mathcal{F}} \log d_f \log^2 p + (m+n)T_{\mathcal{G}_o} d_g^2 \log^2 h + \log u), \end{aligned}$$

and κ is the maximal condition number of all $L_{\alpha\beta}$ in the algorithm, called the condition number *of the problem.*

Proof We first prove the termination of the algorithm by showing that the feasible interval $[\alpha, \mu)$ will decrease strictly after each loop starting from Step 3. In Step 3, we split $[\alpha, \mu) = [\alpha, \alpha + 2^\beta) \cup [\alpha + 2^\beta, \mu)$ with $(\mu - \alpha)/4 < 2^\beta \le (\mu - \alpha)/2$. In Step 5.3, we start a new loop for $[\alpha, \mu_1)$, where $\mu_1 = \overline{o}(\check{\mathbb{Z}}_{\text{bit}}) < 2^\beta$. Then after this step, the feasible interval will decrease by at least $\frac{1}{2}(\mu - \alpha)$ due to $2^\beta \le (\mu - \alpha)/2$. In Step 6.1, we start a new loop for $[\alpha + 2^\beta, \mu)$. After this step, the feasible interval will decrease by more than $\frac{1}{4}(\mu - \alpha)$ due to $(\mu - \alpha)/4 < 2^\beta$. In summary, after each loop, the algorithm either terminates or has a smaller feasible interval which is of at most 3/4 of the size of the feasible interval of the previous loop. So, the algorithm will terminate after at most $\log_{4/3} u$ loops.

We now prove the correctness of the algorithm, which follows from the following claim:

$$\text{The minimal value of } o \text{ is in } [\alpha, \mu] \text{ during the algorithm} \tag{20}$$

if the minimal value exists and Algorithm **QBoolSol** in Step 4 always returns a solution of $L_{\alpha\beta}$ if such a solution exists. The above claim is obviously true for the initial values given in Step 1.

In Step 5, by Lemma 5.2, we find a solution $\check{\mathbb{Z}}_{\rm bit}$ such that $\overline{o}(\check{\mathbb{Z}}_{\rm bit}) \in [\alpha, \alpha+2^\beta)$. In Step 5.2, the condition $\check{\mathbb{F}}_{\rm bit} = \mathbf{0}$ means that $\overline{o}(\check{\mathbb{Z}}_{\rm bit}) = \alpha$ and the minimal solution $\overline{o}$ is found by claim (20). In Step 5.3, the condition $\check{\mathbb{F}}_{\rm bit} \neq \mathbf{0}$ means that $\overline{o}(\check{\mathbb{Z}}_{\rm bit}) \neq \alpha$ and we have a new $\mu_1 = \overline{o}(\check{\mathbb{Z}}_{\rm bit})$. Since $o \in [\alpha, \mu_1]$ has a solution $\check{\mathbb{Z}}_{\rm bit}$, by claim (20), the minimal value of o is in $[\alpha, \mu_1]$, and the claim is proved in this case.

In Step 6, **QBoolSol** returns $\emptyset$, meaning that $\overline{o}(\check{\mathbb{Z}}_{\rm bit}) \in [\alpha, \alpha+2^\beta)$ has no solution and the minimal value of o must be in $[\alpha+2^\beta, \mu)$ if it exists. So, in Step 6.1, we will find the minimal value of o in $[\alpha+2^\beta, \mu)$ in the next loop, and claim (20) is proved in this case. In Step 6.2, we have $\mu - \alpha = 1$ and $\mu \neq u$. Since $o \in [\alpha, \alpha+2^\beta)$ has no solution, by claim (20), $\mu = \alpha + 1$ must be the minimal value of o. Note that in Step 6, we only update the lower bound α. In Step 5, we only update the upper bound μ, and when μ is updated we have $\mu = \overline{o}(\check{\mathbb{Z}}_{\rm bit}) = o(\check{\mathbb{X}}, \check{\mathbb{Y}})$. Therefore, $\mu = o(\check{\mathbb{X}}, \check{\mathbb{Y}})$ is always valid, once Step 5 is executed. The condition $\mu \neq u$ implies that Step 5 has been executed at least one time and hence $\mu = o(\check{\mathbb{X}}, \check{\mathbb{Y}})$. In Step 6.3, the conditions $\mu - \alpha = 1$ and $\mu = u$ means that Step 5 is never executed and the problem has no solution.

Finally, the solution obtained by Algorithm 5.3 is correct if and only if each Step 4 is correct, that is, if $L_{\alpha\beta}$ does have solutions, then **QBoolSol** will return a solution. Since Step 4 will execute at most $\log_{4/3} u$ times, by Theorem 3.6, the probability for the algorithm to be correct is at least $(1-\varepsilon/\log_{4/3} u)^{\log_{4/3} u} > 1-\varepsilon$.

We now analyse the complexity. Note that 2 is added to d_f to make sure $\log d_f \neq 0$. By Lemma 3.8, $\mathcal{F}_1$ is of total sparseness $O(nT_{\mathcal{F}} \log d_f \log^2 p)$ and has $O(nT_{\mathcal{F}} \log d_f \log p)$ indeterminates.

By Lemma 4.5, $\mathcal{G}_1 = I(\mathcal{I})$ is of total sparseness $O((m+n)T_{\mathcal{G}} d_g^2 \log d_g \log^2 h)$ and has $O((m+n)d_g T_{\mathcal{G}} \log d_g \log h)$ indeterminates. Also, $\mathcal{G}_2 = \overline{B}(o)$ is of total sparseness $O((m+n)T_o d_g^2 \log d_g \log^2 h)$ and has $O((m+n)d_g T_o \log d_g \log h)$ indeterminates.

$\delta_{\alpha\beta}(\overline{o})$ is of total sparseness $O(\log u + T_{\overline{o}}) = O(\log u + T_o d_g^2 \log^2 h)$ and has $O(\log u + (m+n) d_g T_o \log d_g \log h)$ indeterminates, since $2^\beta - \alpha < u$.

Then, $L_{\alpha\beta} = \mathcal{C} \cup \{\delta_{\alpha\beta}(\overline{o})\} = \mathcal{F}_1 \cup \mathcal{G}_1 \cup \mathcal{G}_2 \cup \{\delta_{\alpha\beta}(\overline{o})\}$ is of total sparseness $T_{L_{\alpha\beta}} = T_{\mathcal{F}_1} + T_{\mathcal{G}_1} + T_{\mathcal{G}_2} + T_{\delta_{\alpha\beta}(\overline{o})} = \widetilde{O}(nT_{\mathcal{F}} \log d_f \log^2 p + (m+n) T_{\mathcal{G}_o} d_g^2 \log^2 h + \log u)$ and has $N_{L_{\alpha\beta}} = \widetilde{O}(nT_{\mathcal{F}} \log d_f \log p + (m+n) d_g T_{\mathcal{G}_o} \log h + \log u)$ indeterminates.

In Step 4, by Theorem 3.6, since $N_{L_{\alpha\beta}} < T_{L_{\alpha\beta}}$, we can find a Boolean solution of $L_{\alpha\beta}$ in time $\widetilde{O}(N_{L_{\alpha\beta}}^{2.5} (N_{L_{\alpha\beta}} + T_{L_{\alpha\beta}}) \kappa^2 \log(\log u/\varepsilon)) = \widetilde{O}(N_{L_{\alpha\beta}}^{2.5} T_{L_{\alpha\beta}} \kappa^2 \log(\log u/\varepsilon))$. It is clear that in each loop, the complexity of the algorithm is dominated by Step 4. Since we have at most $\log_{4/3} u$ loops, the complexity for Algorithm 5.3 is $\widetilde{O}(N_{L_{\alpha\beta}}^{2.5} T_{L_{\alpha\beta}} \kappa^2 \log((\log_{4/3} u)/\varepsilon) \log_{4/3} u) = \widetilde{O}(N_{L_{\alpha\beta}}^{2.5} T_{L_{\alpha\beta}} \kappa^2 \log(1/\varepsilon) \log u)$. ▮

We now show how to solve the original problem (1).

Corollary 5.5 *Algorithm 5.3 gives a solution to problem (1) with the same probability and complexity for* $u = 2\#(o) h_o h^{d_o} + 1$*, where* h_o *is the height of the coefficients of* o*,* $h = \max\{p-1, u_1, u_2, \ldots, u_m\}$*, and* $d_o = \deg(o)$.

Proof It is easy to see that $|o| \leq \#(o) h_o h^{d_o}$, so $0 \leq \widetilde{o} < u$ for the new objective function $\widetilde{o} = o + \#(o) h_o h^{d_o}$. Then Theorem 5.4 can be used to the new optimization problem. ▮

Remark 5.6 Note that the upper bound $u = 2\#(o) h_o h^{d_o} + 1$ for the objective function is quite large. An alternative way is to use **Algorithm QBoolSol** in Theorem 3.6 to find a solution $\check{\mathbb{X}}_{\rm bit}, \check{\mathbb{Y}}_{\rm bit}$ for $\mathcal{F}_1 \cup \mathcal{G}_1 \subset \mathbb{C}[\mathbb{X}_{\rm bit}, \mathbb{Y}_{\rm bit}, \overline{\mathbb{V}}_{1\rm bit}, \overline{\mathbb{V}}_{2\rm bit}, \mathbb{U}_{\rm bit}, \mathbb{G}_{\rm bit}]$ and set $u = 2o(\check{\mathbb{X}}, \check{\mathbb{Y}}) + 1$. Then for the new objective function $\widetilde{o} = o + o(\check{\mathbb{X}}, \check{\mathbb{Y}})$, we can use the constraint $0 \leq \widetilde{o} < u$ to find a solution to problem (1). This does not change the complexity of the algorithm.

5.2 Applications to linear $(0,1)$-programming and QUBO

In this section, we use two $(0,1)$-programming problems to illustrate Algorithm 5.3. QUBO means *quadratic unconstrained binary optimization*, which is the mathematical problem that can be solved by the D-Wave [23].

The linear $(0,1)$-programming is one of Karp's 21 NP-complete problems [24] which covers lots of fundamental computational problems, such as the subset sum problem, the assignment problem, the traveling salesperson problem, the knapsack problem, etc. For more information about this problem, please refer to [18]. The linear $(0,1)$-programming can be stated as follows [8]

$$\min_{\mathbb{Y}_{\rm bit}\in\{0,1\}^m} o(\mathbb{Y}_{\rm bit}) = \sum_{j=1}^m c_j y_j \text{ subject to } \quad \sum_{j=1}^m a_{ij}y_j \le h_i, i=1,2,\ldots,s \tag{21}$$

where $\mathbb{Y}_{\rm bit}=(y_1,y_2,\ldots,y_m)$ and $a_{ij},c_j,h_i\in\mathbb{Z}$ for any i,j. We reduce problem (21) to the standard form (1). Let $e_i=\sum_{j=1}^m|a_{ij}|\in\mathbb{Z}_{\ge 0}$, $1\le i\le s$ and $g_i=\sum_{j=1}^m a_{ij}y_j+e_i$ and $b_i=h_i+e_i, 1\le i\le s$. Let $u=2\sum_{i=1}^m|c_j|+1\in\mathbb{N}$. Then, problem (21) is equivalent to

$$\begin{aligned}\min_{\mathbb{Y}_{\rm bit}\in\{0,1\}_2^m} o_B(\mathbb{Y}_{\rm bit}) = \sum_{j=1}^m c_j y_j + (u-1)/2 \text{ subject to}\\ 0\le o_B(\mathbb{Y}_{\rm bit})<u; 0\le g_i\le b_i, i=1,2,\ldots,s.\end{aligned} \tag{22}$$

So we can use Algorithm 5.3 to solve problem (22). Let $\mathcal{G}=\{g_1,g_2,\ldots,g_s\}$. Since g_i are linear, we do not need to compute $Q(g_i)$ and $\overline{\mathbb{V}}_{\rm bit}=\emptyset$ (see (12) for definition) and $\overline{B}(\mathcal{G})=\emptyset$ (see (13) for definition). Since y_j are Boolean variables, we do not need to use (11) to expand them and hence $\overline{g}_i=g_i$. So, (14) becomes

$$\begin{aligned}\delta(g_i) &= \theta_{b_i}(\mathbb{G}_i) = \sum_{k=0}^{\lfloor\log_2 b_i\rfloor-1} G_{i,k}2^k + (b_i - 2^{\lfloor\log_2 b_i\rfloor}+1)G_{i,\lfloor\log_2 b_i\rfloor} - g_i\\ I(\mathcal{I}) &= \{\delta(g_1),\delta(g_2),\ldots,\delta(g_s)\}\subset\mathbb{Z}[\mathbb{Y}_{\rm bit},\mathbb{G}_{\rm bit}],\end{aligned}$$

where $\mathbb{G}_{\rm bit}=\{G_{ikl}\}$ are Boolean variables and $\theta_{b_i}(\mathbb{G}_i)$ is defined in (2). Equation (18) becomes

$$\begin{aligned}\delta_{\alpha\beta}(o) &= \alpha+\sum_{j=0}^{\beta-1}F_j2^j - o_B \in\mathbb{Z}[\mathbb{Y}_{\rm bit},\mathbb{F}_{\rm bit}],\\ L_{\alpha\beta} &= I(\mathcal{I})\cup\{\delta_{\alpha\beta}(o)\}\subset\mathbb{Z}[\mathbb{Y}_{\rm bit},\mathbb{G}_{\rm bit},\mathbb{F}_{\rm bit}],\end{aligned} \tag{23}$$

where $\mathbb{F}_{\rm bit}=\{F_1,F_2,\ldots,F_{\beta-1}\}$ are Boolean variables.

Proposition 5.7 *We can use Algorithm 5.3 to solve problem (22) with probability* $\ge 1-\varepsilon$ *and in time* $\widetilde{O}(s(m^{2.5}+s^{2.5}\log^{2.5}h)(m+\log h)\kappa^2\log(1/\varepsilon)\log u)$ *where* $u=2\sum_{i=1}^m|c_j|+1$, $b=\max_{i=1}^s b_i$, $h=\max\{u,b\}$, *and* κ *is the maximal condition number of* $L_{\alpha\beta}$.

Proof Since $\#\mathbb{Y}_{\rm bit}=m$, $\#\mathbb{G}_{\rm bit}=s\log b$, and $\#\mathbb{F}_{\rm bit}=\log u$, $L_{\alpha\beta}$ has $m+s\log b+\log u$ Boolean variables and total sparseness $s(m+\log b)+m+1+\log u$. Since $u,b\le h$, $L_{\alpha\beta}$ has $N_{L_{\alpha\beta}}=O(m+s\log h)$ Boolean variables and total sparseness $T_{L_{\alpha\beta}}=O(s(m+\log h))$. By Theorem 3.6, the complexity is $\widetilde{O}((m+s\log h)^{2.5}(s(m+\log h))\kappa^2\log(1/\varepsilon)\log u)=\widetilde{O}(s(m^{2.5}+s^{2.5}\log^{2.5}h)(m+\log h)\kappa^2\log(1/\varepsilon)\log u)$. ∎

In the remainder of this section, we consider the QUBO problem. The QUBO problem is to find an $\mathbb{Y}_{\rm bit}=(y_1,y_2,\ldots,y_m)^T\in\{0,1\}^m$ that minimizes $\mathbb{Y}_{\rm bit}^TQ\mathbb{Y}_{\rm bit}$ for an upper-triangular matrix $Q=(Q_{i,j})$ with

$Q_{i,j} \in \mathbb{Z}$, which can be written as the following $(0,1)$-programming problem:

$$\min_{\mathbb{Y}\in\{0,1\}^m} o_Q(\mathbb{Y}_{\rm bit}) = \mathbb{Y}_{\rm bit}^T Q \mathbb{Y}_{\rm bit} \tag{24}$$

In order to solve this problem, we need to give the lower and upper bounds for the objective function. Let $Q_{\max} = \max_{i,j} |Q_{i,j}|$. Since $y_i \in \{0,1\}, 1 \le i \le m$, we have $|o(\mathbb{Y})| \le m^2 Q_{\max}$.

Problem (24) can be converted into standard form with the new objective function $\widetilde{o}_Q = o_Q + m^2 Q_{\max}$ and $u = 2m^2 Q_{\max} + 1$. Then, we can use Algorithm 5.3 to solve problem (24). Let

$$\delta_{\alpha\beta}(o) = \alpha + \sum_{j=0}^{\beta-1} F_j 2^j - \widetilde{o}_Q \in \mathbb{C}[\mathbb{Y}_{\rm bit}, \mathbb{F}_{\rm bit}],$$

where $\mathbb{F}_{\rm bit} = (F_0, F_1, \ldots, F_{\beta-1}) \in \mathbb{F}_2^\beta$ and $L_{\alpha\beta} = \{\delta_{\alpha\beta}(o)\}$. We have

Proposition 5.8 *We can use Algorithm 5.3 to solve problem (24) with probability* $\ge 1-\varepsilon$ *and in time* $\widetilde{O}(m^{2.5} + \log^{2.5} Q_{\max})(m^2 + \log Q_{\max}) \log Q_{\max} \kappa^2 \log(1/\varepsilon))$ *where* κ *is the maximal condition number of* $L_{\alpha\beta}$ *and* $Q_{\max} = \max_{i,j} |Q_{i,j}|$.

Proof Since o_Q is quadratic and the variables are Boolean, we can solve $L_{\alpha\beta} = \{\delta_{\alpha\beta}(o)\}$ directly with Theorem 3.6. Using the notations in Theorem 5.4, we have $N_{L_{\alpha\beta}} = m + \log(m^2 Q_{\max}) = \widetilde{O}(m + \log Q_{\max})$, $T_{L_{\alpha\beta}} = \widetilde{O}(m^2 + \log Q_{\max})$, $u = 2m^2 Q_{\max} + 1$. By Theorem 5.4, the complexity is $\widetilde{O}((m + \log Q_{\max})^{2.5}(m^2 + \log Q_{\max})\kappa^2 \log(1/\varepsilon)(\log m + \log Q_{\max})) = \widetilde{O}(m^{2.5} + \log^{2.5} Q_{\max})(m^2 + \log Q_{\max}) \log Q_{\max} \kappa^2 \log(1/\varepsilon))$. ∎

6 Polynomial system with noise

In this section, we consider the *polynomial systems with noise problem (PSWN)*, which is an optimization problem over finite fields and has important applications in cryptography [2, 22].

6.1 Polynomial system with noise

Definition 6.1 Let p be a prime. Given a polynomial system $\mathcal{F} = \{f_1, f_2, \ldots, f_r\} \subset \mathbb{F}_p[\mathbb{X}]$, the PSWN is to find an $\mathbb{X} = (x_1, x_2, \ldots, x_n)^{\rm T} \in \mathbb{F}_p^n$ such that $\mathcal{F} = \mathbf{e}$ for the "smallest" error-vector $\mathbf{e} = (e_1, e_2, \ldots, e_r)^{\rm T} \in \mathbb{F}_p^r$.

In most cases, the Hamming weight $\|\mathbf{e}\|_H$ of $\mathbf{e}$ is used to measure the "smallness" and it is assumed that $r \gg n$, that is, we minimize the number of non-zero components of $\mathbf{e}$ or satisfy the maximal number of equations of $\mathcal{F} = 0$. Therefore, PSWN is also called MAX-POSSO. We first give the following representation for $\|\mathbf{e}\|_H$.

Lemma 6.2 *Let* $\mathbf{e} = (e_1, e_2, \ldots, e_r)^{\rm T} \in \mathbb{F}_p^n$ *and* $H_j = e_j^{p-1}$ *in* $\mathbb{F}_p$. *Then* H_j *is Boolean and* $\|\mathbf{e}\|_H = \sum_{j=1}^m H_j$ *when the summation is over* $\mathbb{C}$.

Proof $e_j \in \mathbb{F}_p$ implies $H_j = e_j^{p-1}$ is either 0 or 1 in $\mathbb{F}_p$, and $H_j = 1$ if and only if $e_j \neq 0$. Then, H_j is a Boolean variable. Thus, we have $\sum_{j=1}^m H_j = \|\mathbf{e}\|_H$ when the summation is over $\mathbb{C}$. ∎

Let

$$E(\mathcal{F}) = (\mathcal{F} - \mathbf{e}) \cup \{H_j - e_j^{p-1} \mid j = 1, 2, \ldots, r\} \subset \mathbb{F}_p[\mathbb{X}, \mathbb{E}, \mathbb{H}_{\rm bit}] \tag{25}$$

where $\mathbb{H}_{\rm bit} = \{H_1, H_2, \ldots, H_r\}$ are Boolean variables and $\mathbb{E} = \{e_1, e_2, \ldots, e_r\}$ are variables over $\mathbb{F}_p$. By

Lemma 6.2, PSWN can be formulated as the following optimization problem over finite fields:

$$\min_{\mathbb{X}\in\mathbb{F}_p^n} o(\mathbb{X}) = \sum_{j=1}^{r} H_j \qquad \text{subject to } 0 \le o(\mathbb{X}) \le r; E(\mathcal{F}) = 0 \mod p \tag{26}$$

which can be solved by Algorithm 5.3.

Due to the special structure of $E(\mathcal{F})$, we can achieve better complexities than that given in Theorem 5.4. Following (19), the equation set $L_{\alpha\beta}$ for PSWN is

$$\begin{aligned} \delta_{\alpha\beta}(o) &= \alpha + \sum_{j=0}^{\beta-1} F_j 2^j - \sum_{j=1}^{r} H_j \in \mathbb{C}[\mathbb{H}_{\rm bit}, \mathbb{F}_{\rm bit}], \\ L_{\alpha\beta}(\mathcal{F}) &= P(Q(E(\mathcal{F}))) \cup \{\delta_{\alpha\beta}(o)\} \subset \mathbb{C}[\mathbb{X}_{\rm bit}, \mathbb{E}_{\rm bit}, \mathbb{H}_{\rm bit}, \mathbb{F}_{\rm bit}, \mathbb{V}_{\rm bit}, \mathbb{U}_{\rm bit}], \end{aligned} \tag{27}$$

where $\mathbb{F}_{\rm bit} = \{F_1, F_2, \ldots, F_{\beta-1}\}$ are Boolean variables. We have

Proposition 6.3 *There is a quantum algorithm to solve PSWN in time $\widetilde{O}(r^{3.5}T_{\mathcal{F}}^{3.5}\log^8 p\kappa^2\log 1/\varepsilon)$ and with probability $\ge 1-\varepsilon$, where $T_{\mathcal{F}}$ is the total sparseness of $\mathcal{F}$, and κ is the condition number of $\mathcal{F}$.*

Proof We first give the complexity of Step 4 of Algorithm 5.3, that is, the complexity to solve $L_{\alpha\beta}(\mathcal{F})$. Let $\mathcal{F}_1 = \{f_1 - e_1, f_2 - e_2, \ldots, f_r - e_r\}$, $\mathcal{F}_2 = \{H_1 - e_1^{p-1}, H_2 - e_2^{p-1}, \ldots, H_r - e_r^{p-1}\}$, and $\mathcal{F}_3 = \mathcal{F}_1 \cup \mathcal{F}_2$. Then $P(Q(\mathcal{F}_3)) = P(Q(\mathcal{F}_1)) \cup P(Q(\mathcal{F}_2))$. By Lemma 3.8, $T_{P(Q(\mathcal{F}_1))} = O(T_{\mathcal{F}}D\log^2 p)$ and $N_{P(Q(\mathcal{F}_1))} = O(T_{\mathcal{F}}D\log p)$. Since each monomial in $\mathcal{F}_2$ depends on one indeterminate, we have $Q(\mathcal{F}_2) \subset \mathbb{F}_p[\mathbb{H}_{\rm bit}, \mathbf{e}, \mathbb{V}]$, where $\#\mathbb{V} = O(r\log p), \#Q(\mathcal{F}_2) = O(r\log p)$, and $T_{Q(\mathcal{F}_2)} = O(r\log p)$ by the proof for Lemma 2.3. By Lemma 3.3, $T_{P(Q(\mathcal{F}_2))} = O(r\log^3 p)$ and $N_{P(Q(\mathcal{F}_2))} = O(r\log^2 p)$. Since $x \in \mathbb{F}_p$ implies $x^p = x$, we can assume $\deg_{x_i} f_j < p$. Thus $D = n + \sum_{i=1}^r \lfloor \log_2 \max_j \deg_{x_i} f_j \rfloor \le n + r\lfloor \log_2(p-1)\rfloor = O(n + r\log p)$, and we have $T_{L_{\alpha\beta}} = O(T_{\mathcal{F}}D\log^2 p + r\log^3 p + r + \log p) = O(T_{\mathcal{F}}r\log^3 p)$ and $N_{L_{\alpha\beta}} = O(T_{\mathcal{F}}D\log p + r\log^2 p + \log r) = O(T_{\mathcal{F}}r\log^2 p)$ considering $r \gg n$. By Theorem 3.6, the complexity to solve $L_{\alpha\beta}$ is $\widetilde{O}((T_{\mathcal{F}}r\log^2 p)^{2.5}(T_{\mathcal{F}}r\log^2 p + T_{\mathcal{F}}r\log^3 p)\kappa^2\log 1/\varepsilon) = \widetilde{O}(r^{3.5}T_{\mathcal{F}}^{3.5}\log^8 p\kappa^2\log 1/\varepsilon)$. The number of loops is at most $\log r$, which is negligible since $r \le T_{\mathcal{F}}$, and the complexity of the algorithm is that of Step 4. The theorem is proved. ∎

Similar to Corollary 3.10, if $\mathcal{F}$ is an MQ then the complexity is lower.

Corollary 6.4 *There is a quantum algorithm to solve the* MQ with noise *in time $\widetilde{O}(r^{2.5}(T_{\mathcal{F}} + r\log p)$ $\log^7 p\kappa^2\log 1/\varepsilon)$ with probability $1-\varepsilon$.*

6.2 Linear system with noise

When $\mathcal{F}$ becomes a linear system, we obtain the *linear system with noise (LSWN)* [19]. Given a matrix $A = (A_{ij}) \in \mathbb{F}_p^{r\times n}$ and a vector $\mathbf{b} = (b_1, b_2, \ldots, b_r)^{\rm T} \in \mathbb{F}_p^r$. The LSWN problem is to find an $\mathbb{X}$ such that $A\mathbb{X} - \mathbf{b} = \mathbf{e}$ and the error-vector $\mathbf{e} \in \mathbb{F}_p^r$ has minimal Hamming weight $\|\mathbf{e}\|_H$.

The algorithm given in Section 6.1 can be used to solve the LSWN and Proposition 6.3 becomes the following form.

Proposition 6.5 *There exists a quantum algorithm to solve LSWN with probability $\ge 1-\varepsilon$ and in time $\widetilde{O}((n + r\log p)^{2.5}(T_A + r\log^2 p)\log^{3.5} p\kappa^2\log 1/\varepsilon)$, where $T_A \ge \max\{r, n\}$ is the number of nonzero entries in A, and κ is the condition number of $A\mathbb{X}$.*

Proof Similar to the proof of Proposition 6.3, we need only consider the complexity of solving $L_{\alpha\beta}(A\mathbb{X} - \mathbf{b})$. Let $f_i = \sum_{j=1}^n A_{ij}x_j - b_i - e_i \in \mathbb{F}_p[\mathbb{X}, \mathbf{e}]$, $\mathcal{F}_1 = \{f_1, f_2, \ldots, f_r\}$, $\mathcal{F}_2 = \{H_1 - e_1^{p-1}, H_2 - e_2^{p-1}, \ldots, H_r - e_r^{p-1}\}$, and we have $E(A\mathbb{X}-\mathbf{b}) = \mathcal{F}_1 \cup \mathcal{F}_2$. Since $\mathcal{F}_1$ is a linear system, we have $P(Q(E(A\mathbb{X}-\mathbf{b}))) = P(\mathcal{F}_1) \cup P(Q(\mathcal{F}_2))$. By Corollary 3.4, $T_{P(\mathcal{F}_1)} = O(T_A \log p)$ and $N_{P(\mathcal{F}_1)} = n \log p + \sum_{i=1}^r \log t_i + r \log\log p$, where t_i is the sparseness for the i-th row of matrix A. By Lemma 3.3, $T_{P(Q(\mathcal{F}_2))} = O(r \log^3 p)$ and $N_{P(Q(\mathcal{F}_2))} = O(r \log^2 p)$. Thus, $T_{L_{\alpha\beta}(A\mathbb{X}-\mathbf{b})} = O(T_A \log p + r \log^3 p)$ and $N_{L_{\alpha\beta}(A\mathbb{X}-\mathbf{b})} = O(n \log p + \sum_{i=1}^r \log t_i + r \log^2 p)$. By Theorem 3.6, the complexity to solve $L_{\alpha\beta}(A\mathbb{X}-\mathbf{b})$ is

$$\begin{aligned} & \widetilde{O}((n\log p + \sum_{i=1}^r \log t_i + r\log^2 p)^{2.5}((n\log p + \sum_{i=1}^r \log t_i + r\log^2 p) + (T_A \log p + r\log^3 p))\kappa^2 \log 1/\varepsilon) \\ = \quad & \widetilde{O}((n\log p + r\log^2 p)^{2.5}(n\log p + T_A \log p + r\log^3 p)\kappa^2 \log 1/\varepsilon). \end{aligned}$$

Since $T_A \geq r$ and we can assume $T_A \geq n$ without loss of generality, the complexity is $\widetilde{O}((n\log p + r\log^2 p)^{2.5}(T_A \log p + r\log^3 p)\kappa^2 \log 1/\varepsilon) = \widetilde{O}((n + r\log p)^{2.5}(T_A + r\log^2 p)\log^{3.5} p\kappa^2 \log 1/\varepsilon)$. ∎

7 Short integer solution problem

In this section, we consider the *short integer solution problem (SIS)*, which is a basic problem in the latticed based cryptosystems [1].

7.1 Short integer solution problem

Consider the SIS problem introduced in [1]:

Definition 7.1 Let $A = (A_{ij}) \in \mathbb{F}_p^{r\times n}$. The *SIS* is to find an $\mathbb{X} \in \mathbb{F}_p^n$ such that $A\mathbb{X} = 0 \pmod p$ and the Euclidean norm of $\mathbb{X}$ satisfies $0 < \|\mathbb{X}\|_2 \leq b$, where b is a given integer.

We first consider the more general *SIS* for $\mathcal{F} = \{f_1, f_2, \ldots, f_r\} \subset \mathbb{F}_p[\mathbb{X}]$: find an $\mathbb{X}$ such that $\mathcal{F}(\mathbb{X}) = 0 \pmod p$ and $0 < \|\mathbb{X}\|_2 \leq b$. Note that SIS is a special case of the optimization problem (1), where the objective function is a constant and the problem is to find a feasible solution for the constraints. Precisely, the SIS can be formualted as the following standard form

$$\min_{\mathbb{X}\in\mathbb{F}_p^n} o = 1 \qquad \text{subject to} \quad \mathcal{F} = 0 \mod p,\ 0 \leq \|\mathbb{X}\|_2^2 - 1 \leq b^2 - 1. \tag{28}$$

From Remark 4.3, the representation for $\mathbb{F}_p$ affects inequality constraints. For the inequality $0 < \|\mathbb{X}\|_2^2 \leq b^2$, a better representation for $\mathbb{F}_p$ is $\{-\frac{p-1}{2}, -\frac{p-1}{2}+1, \ldots, \frac{p-1}{2}\}$, instead of $\{0, 1, \ldots, p-1\}$. In this section, we still use $\{0, 1, \ldots, p-1\}$ to represent elements in $\mathbb{F}_p$, but use the following variable expansion instead of (5):

$$x_i = \overline{\theta}_{p-1}(\mathbb{X}_i) - \frac{p-1}{2} = \sum_{j=0}^{\lfloor\log_2(p-1)-1\rfloor} X_{i,j}2^j + (p - 2^{\lfloor\log_2(p-1)\rfloor})X_{i,\lfloor\log_2(p-1)\rfloor} - \frac{p-1}{2} \tag{29}$$

where $\mathbb{X}_i$ are defined in (5). Then, x_i takes values $-\frac{p-1}{2}, -\frac{p-1}{2}+1, \ldots, \frac{p-1}{2}$ when evaluated over $\mathbb{C}$. The following easy result shows that this representation gives the "global" solution to problems involving the Euclidean norm.

Lemma 7.2 *For* $\check{\mathbb{X}} \in [-\frac{p-1}{2}, \frac{p-1}{2}]^n$ *and any vector* $\mathbf{v} \in (p\mathbb{Z})^n \backslash \mathbf{0}$, $\|\check{\mathbb{X}}\|_2 < \|\check{\mathbb{X}} + \mathbf{v}\|_2$.

Due to (14) and by Lemma 7.2, the constraint $0 \leq \|\mathbb{X}\|_2^2 - 1 \leq b^2 - 1$ can be written as the following MQ in Boolean variables

$$\overline{\delta}_b \quad = \quad \overline{\theta}_{b^2-1}(\mathbb{G}_{\text{bit}}) - \sum_{i=1}^n (\overline{\theta}_{p-1}(\mathbb{X}_i) - \frac{p-1}{2})^2 + 1 \in \mathbb{C}[\mathbb{G}_{\text{bit}}, \mathbb{X}_{\text{bit}}] \tag{30}$$

where $\mathbb{G}_{\text{bit}} = \{G_k \mid k = 0, 1, \ldots, \lfloor\log_2(b^2-1)\rfloor\}$. From the above discussion, we have

Lemma 7.3 *To solve the SIS, we need only to find a solution of* $\overline{P}(Q(\mathcal{F}))\cup\{\overline{\delta}_b\}\subset\mathbb{C}[\mathbb{X}_{\rm bit},\mathbb{V}_{\rm bit},\mathbb{U}_{\rm bit},\mathbb{G}_{\rm bit}]$*, where* $\overline{P}(Q(\mathcal{F}))$ *is obtained similar to* $P(Q(\mathcal{F}))$*, but using (29) to expand* $\mathbb{X}$*,* $\mathbb{V}$*, and* $\mathbb{U}$*.*

Proposition 7.4 *There is a quantum algorithm to solve the SIS problem (28) with probability at least* $1-\varepsilon$ *and complexity* $\widetilde{O}(n^{3.5}T_{\mathcal{F}}^{3.5}\log^{3.5}d\log^{4.5}p\kappa^2\log 1/\varepsilon)$ *where* $T_{\mathcal{F}}$ *is the total sparseness of* $\mathcal{F}$*,* $d=\max\{2,\log_2(\deg_{x_i}(f_j)),i=1,2,\ldots,n,j=1,2,\ldots,r\}$*, and* κ *is the condition number of* $P(Q(\mathcal{F}))\cup\{\overline{\delta}_b\}$*.*

Proof Since $\|\mathbb{X}\|_2^2\le np^2$, by Lemma 2.1, $\#\mathbb{G}_{\rm bit}=\widetilde{O}(\log(b^2-1))\le\widetilde{O}(\log(np^2))=\widetilde{O}(\log(n)+\log p)$ and $\#\overline{\delta}_b=\widetilde{O}(\log(b^2-1)+n\log^2 p)=\widetilde{O}(\log(b)+n\log^2 p)=\widetilde{O}(n\log^2 p)$, since $b\le np^2$. By Lemma 3.8, $\overline{P}(Q(\mathcal{F}))$ is of sparseness $O(nT_{\mathcal{F}}\log d\log^2 p)$ and with $O(nT_{\mathcal{F}}\log d\log p)$ indeterminates. By Lemma 7.3, we need to solve $\overline{P}(Q(\mathcal{F}))\cup\{\overline{\delta}_b\}$ with Theorem 3.6. Comparing to the total sparseness and number of variables of $\overline{P}(Q(\mathcal{F}))$, $\#\overline{\delta}_b$ and $\#\mathbb{G}_{\rm bit}$ are negligible. Then, the complexity of solving the SIS is the same as that of solving $\overline{P}(Q(\mathcal{F}))$. Then, the theorem follows from Corollary 3.9. ∎

For the original SIS, we have

Proposition 7.5 *There is a quantum algorithm to find an non-trivial* $\mathbb{X}\in\mathbb{Z}^n$ *for* $A\mathbb{X}=0\pmod p$ *with* $\|\mathbb{X}\|_2\le b$ *with probability* $1-\varepsilon$ *and in time* $\widetilde{O}((n\log p+r)^{2.5}(T_A\log p+n\log^2 p)\kappa^2\log 1/\varepsilon)$*, where* T_A *is the number of nonzero elements of* A*, assuming* $T_A\ge n$*.*

Proof By Corollary 3.4, $P(Q(A\mathbb{X}))=P(A\mathbb{X})$ is of total sparseness $O(T_A\log p)$ and has $O(n\log p+\sum_{i=1}^r\log t_i+r\log\log p)$ indeterminates, where t_i is the sparseness for the i-th line of matrix A.. From the proof of Proposition 7.4, $\#\mathbb{G}_{\rm bit}=\widetilde{O}(\log n+\log p)$ and $\#\overline{\delta}_b=\widetilde{O}(n\log^2 p)$. Since $T_A\ge n$, we have $T_{L_{\alpha\beta}}=O(T_A\log p+n\log^2 p)$ and $N_{L_{\alpha\beta}}=O(n\log p+\sum_{i=1}^r\log t_i+r\log\log p)$. Comparing to the total sparseness and number of variables of $\overline{P}(Q(\mathcal{F}))$, $\#\overline{\delta}_b$ is negligible. By Theorem 3.6, the complexity to solve $\overline{P}(Q(\mathcal{F}))\cup\{\overline{\delta}_b\}$ is $\widetilde{O}((n\log p+\sum_{i=1}^r\log t_i+r\log\log p)^{2.5}((n\log p+\sum_{i=1}^r\log t_i+r\log\log p)+(T_A\log p+n\log^2 p))\kappa^2\log 1/\varepsilon)=\widetilde{O}((n\log p+r)^{2.5}(T_A\log p+n\log^2 p)\kappa^2\log 1/\varepsilon)$. ∎

7.2 Smallest integer solution problem

We consider the smallest integer solution problem, which is to find a solution of $\mathcal{F}=0\pmod p$, which has the minimal Euclidean norm. The problem can be formulated as the following standard form

$$\min_{\mathbb{X}\in\mathbb{F}_p^n} o=\|X\|_2^2-1\qquad\text{subject to}\quad\mathcal{F}=0\mod p\ \text{ and } 0\le o<u\tag{31}$$

where $u=n(p-1)^2$. We can use Algorithm 5.3 to solve problem (31). The parameterized objective function and $L_{\alpha\beta}(\mathcal{F})$ are

$$\begin{aligned}\delta_{\alpha\beta}(o)&=\alpha+\sum_{j=0}^{\beta-1}F_j2^j-\sum_{i=1}^n(\theta(\mathbb{X}_i)-\frac{p-1}{2})^2+1\in\mathbb{C}[\mathbb{F}_{\rm bit},\mathbb{X}_{\rm bit}].\\ L_{\alpha\beta}&=\overline{P}(Q(\mathcal{F}))\cup\{\delta_{\alpha\beta}\}\subset\mathbb{C}[\mathbb{X}_{\rm bit},\mathbb{V}_{\rm bit},\mathbb{U}_{\rm bit},\mathbb{F}_{\rm bit}]\end{aligned}\tag{32}$$

where $\overline{P}(Q(\mathcal{F}))$ is defined in Lemma 7.3. We have

Proposition 7.6 *There is a quantum algorithm to solve problem (31) with probability* $\ge 1-\varepsilon$ *and in time* $\widetilde{O}(n^{3.5}T_{\mathcal{F}}^{3.5}\log^{3.5}d\log^{4.5}p\kappa^2\log 1/\varepsilon)$*.*

Proof From the proof of Proposition 7.4, $N_{L_{\alpha\beta}}=O(nT_{\mathcal{F}}\log d\log p)$ and $T_{L_{\alpha\beta}}=O(nT_{\mathcal{F}}\log d\log^2 p)$. Also, $\log u=O(np^2)=O(\log n+\log p)$. By Theorem 5.4, the complexity is $\widetilde{O}(N_{L_{\alpha\beta}}^{2.5}T_{L_{\alpha\beta}}\kappa^2\log 1/\varepsilon\ \log u)$ $=\widetilde{O}(n^{3.5}T_{\mathcal{F}}^{3.5}\log^{3.5}d\log^{4.5}p\kappa^2\log 1/\varepsilon)$. ∎

If $\mathcal{F}$ is a linear system $A\mathbb{X} = 0$ with $T_A \geq n$, then we can prove the following result similar to Propositions 7.6 and 7.5.

Proposition 7.7 *There is a quantum algorithm to find a non-trivial $\mathbb{X} \in \mathbb{Z}^n$ for $A\mathbb{X} = 0 \pmod p$ with minimal $\|\mathbb{X}\|_2$ with probability $\geq 1 - \varepsilon$ and in time $\widetilde{O}((n \log p + r)^{2.5}(T_A \log p + n \log^2 p)\kappa^2 \log 1/\varepsilon)$.*

8 Quantum algorithm for SVP and CVP

In this section, Algorithm 5.3 is used to solve the SVP and CVP problems [6, 25].

A *lattice* generated by $\mathcal{B} = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n\} \subset \mathbb{R}^m$ is the set of $\mathbb{Z}$-linear combinations of $\mathbf{b}_i$. $\mathcal{B}$ is called a basis of the lattice, if $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n$ are linear independent over $\mathbb{R}$. The *SVP problem* can be described as follows: given a lattice L generated by a basis $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n$ in $\mathbb{R}^m$, find a nonzero $\mathbf{v} \in L$ such that $\mathbf{v}$ has the minimal Euclidean norm. The *CVP problem* can be described as follows: given a vector $\mathbf{b_0} \in \mathbb{Z}^m$ and a lattice L generated by a basis $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n$ in $\mathbb{R}^m$, find a $\mathbf{v} \in L$ such that $\mathbf{v} - \mathbf{b_0}$ has the minimal Euclidean norm. In this paper, we assume that $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n$ are in $\mathbb{Z}^m$. The SVP problem can be written as the following optimization problem.

$$\min_{\mathbf{v} \in \mathbb{Z}^m, \mathbf{a} \in \mathbb{Z}^n,} o = \|\mathbf{v}\|_2^2 \qquad \text{subject to} \qquad \mathbf{v} \neq \mathbf{0} \text{ and } \mathbf{v} = \sum_{i=1}^{n} a_i \mathbf{b}_i, \tag{33}$$

where $\mathbf{v} = (v_1, v_2, \ldots, v_m)$ and $\mathbf{a} = (a_1, a_2, \ldots, a_n)$. Note that the SVP problem is similar to the SIS problem considered in Proposition 7.7, but the solutions are over the integers instead of finite fields.

Let

$$B = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n] \in \mathbb{Z}^{m \times n}$$

be the matrix with columns $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n$. In order to reduce problem (33) into the standard form (1), we need to find upper bounds for a_i, v_i, and $\|\mathbf{v}\|_2$. For a matrix or a vector A, let $\|A\|_\infty$ to be the maximum absolute value of the elements in A. It is easy to find bounds for v_i and $\|\mathbf{v}\|_2$.

Lemma 8.1 *Let $\overline{\mathbf{v}} = (\overline{v}_1, \overline{v}_2, \ldots, \overline{v}_m)$ be the shortest vector in L. Then we have $\|\overline{\mathbf{v}}\|_2 \leq \sqrt{m}||B||_\infty$ and $|\overline{v}_i| \leq \sqrt{m}||B||_\infty$.*

Proof It is clear that $\|\overline{\mathbf{v}}\|_2 \leq \sqrt{m}||B||_\infty$ and $|\overline{v}_i| \leq \|\overline{\mathbf{v}}\|_\infty \leq \|\overline{\mathbf{v}}\|_2 \leq \sqrt{m}||B||_\infty$. ∎

In order to bound a_i in (33), we need the concept of *Hermite normal form* (HNF). A matrix $H = (h_{i,j}) \in \mathbb{Z}^{m \times n}$ of rank n is called an (column) HNF if there exists a strictly increasing map f from $[1, n]$ to $[1, m]$ satisfying: for $j \in [1, n]$, $h_{f(j),j} \geq 1$, $h_{i,j} = 0$ if $i > f(j)$ and $h_{f(j),j} > h_{f(j),k} \geq 0$ if $k > j$. It is known that any lattice generated by a basis $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n$ is also generated by $\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n$ if $H = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n]$ is an HNF of $B = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n]$. We need the following obvious property of HNF.

Lemma 8.2 *Let $H = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n] = [h_{ij}]$ be an HNF, $\mathbf{v} = (v_1, v_2, \ldots, v_m)^{\mathrm{T}}$ an element in the lattice generated by $\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n$, and $\mathbf{v} = c_1\mathbf{h}_1 + c_2\mathbf{h}_2 + \cdots + c_n\mathbf{h}_n$ for $c_i \in \mathbb{Z}$. Then $v_{f(n)} = c_n h_{f(n),n}$ and hence $|c_n| \leq ||\mathbf{v}||_\infty$.*

We also need the following result about HNF.

Theorem 8.3 (see [35]) *Let $B \in \mathbb{Z}^{m \times n}$ with rank n and H be the HNF of B. Then, there exists an $E \in \mathbb{Z}^{m \times m}$ such that $EB = H$, $||H||_\infty \leq (\sqrt{n}||B||_\infty)^n$ and $||E||_\infty \leq (\sqrt{n}||B||_\infty)^n$. Furthermore, the bit complexity to compute H from B is $\widetilde{O}(mn^\theta ||B||_\infty)$, where θ is the matrix multiplication constant.*

We now give a bound for a_i in (33).

Lemma 8.4 *Let* $\overline{\mathbf{v}}$ *be the shortest vector in* L*. Then there exist* $a_1, a_2, \ldots, a_n \in \mathbb{Z}$ *such that* $\overline{\mathbf{v}} = \sum_{i=1}^n a_i \mathbf{b}_i$ *and* $|a_i| \leq b_B$*, where* $b_B = n\sqrt{m}||B||_\infty((\sqrt{n}||B||_\infty)^n + 1)^{n+1}$.

Proof Let $H = EB$ be the HNF of B and $\mathbf{h}_i$ the i-th column of H, $1 \leq i \leq n$. Then there exist $c_i, 1 \leq i \leq n$, such that $\overline{\mathbf{v}} = \sum_{i=1}^n c_i \mathbf{h}_i$. Denote $e_1 = \sqrt{m}||B||_\infty$ and $e_2 = (\sqrt{n}||B||_\infty)^n + 1$. We claim that $|c_i| \leq e_1(e_2+1)^n$. Let $\overline{\mathbf{v}}_i = c_1\mathbf{h}_1 + c_2\mathbf{h}_2 + \cdots + c_i\mathbf{h}_i$ for $i = 1, 2, \ldots, n$. We prove the claim by proving $||\overline{\mathbf{v}}_i||_\infty \leq e_1(e_2+1)^{n-i}$ and $|c_i| \leq e_1(e_2+1)^{n-i}$ by induction for $i = n, n-1, \ldots, i$. By Lemma 8.2, the second inequality comes from the first one: $|c_i| \leq ||\overline{\mathbf{v}}_i||_\infty \leq e_1(e_2+1)^{n-i}$, since $[\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_i]$ is also an HNF. By Lemma 8.2, $|c_n| \leq e_1$ and the case of $i = n$ is true. Suppose the claim is true for $i = n, n-1, \ldots, j+1$. By Lemma 8.1, we have $||\mathbf{h}_{j+1}||_\infty \leq e_2$. Since $\overline{\mathbf{v}}_j = \overline{\mathbf{v}}_{j+1} - c_{j+1}\mathbf{h}_{j+1}$, we have $\|\overline{\mathbf{v}}_j\|_\infty \leq ||\overline{\mathbf{v}}_{j+1}||_\infty + |c_{j+1}|||\mathbf{h}_{j+1}||_\infty \leq e_1(e_2+1)^{n-j-1} + e_1(e_2+1)^{n-j-1}e_2 = e_1(e_2+1)^{n-j}$. The claim is proved.

We have $\overline{\mathbf{v}} = \sum_{i=1}^n c_i\mathbf{h}_i = (c_1, c_2, \ldots, c_n)H = (c_1, c_2, \ldots, c_n)EB = a_1\mathbf{b}_1 + a_2\mathbf{b}_2 + \cdots + a_n\mathbf{b}_n$. Then $(a_1, a_2, \ldots, a_n) = (c_1, c_2, \ldots, c_n)E$, and hence $|a_i| \leq n\max_i |c_i| \cdot ||E||_\infty \leq ne_1(e_2+1)^n e_2 \leq ne_1(e_2+1)^{n+1}$ by Theorem 8.3. ∎

By the above lemma, we can rewrite the SVP as the standard form (1):

$$\begin{aligned}
\min_{\mathbf{v}\in\mathbb{Z}^m, \mathbf{a}\in\mathbb{Z}^n} \quad & o = \|\mathbf{v}\|_2^2 - 1 \quad \text{subject to } 0 \leq o < m||B||_\infty^2 \\
& \mathbf{v} = a_1\mathbf{b}_1 + a_2\mathbf{b}_2 + \cdots + a_n\mathbf{b}_n, \\
& 0 \leq a_i + b_B \leq 2b_B, \quad 1 \leq i \leq n, \\
& 0 \leq v_i + \sqrt{m}||B||_\infty \leq 2\sqrt{m}||B||_\infty, \quad 1 \leq i \leq m
\end{aligned}$$

where b_B is given in Lemma 8.4, and the arguments are $\mathbf{v} = (v_1, v_2, \ldots, v_m)$ and $\mathbf{a} = (a_1, a_2, \ldots, a_n)$.

Note that, the above problem is already an MQ, so we just need to change the variables to Boolean variables as follows by using Lemma 2.1.

$$\begin{aligned}
a_i &= \theta_{2b_B}(A_{i,0}, A_{i,1}, \ldots, A_{i,\lfloor\log_2(2b_B)\rfloor}) - b_B, \quad 1 \leq i \leq n, \\
v_i &= \theta_{2\sqrt{m}||B||_\infty}(V_{i,0}, V_{i,1}, \ldots, V_{i,\lfloor\log_2(2\sqrt{m}||B||_\infty)\rfloor}) - \sqrt{m}||B||_\infty, \quad 1 \leq i \leq m
\end{aligned} \tag{34}$$

where and $A_{i,j}, V_{i,j}$ are Boolean variables.

Then we can use Algorithm 5.3 to solve the problem for $u = m||B||_\infty^2$ in the input. For the objective function o, we denote by $\overline{o}$ the Boolean function obtain from o by replacing the v_i by the above equation (34). Let

$$\begin{aligned}
\delta_{\alpha\beta}(\overline{o}) &= \alpha + \sum_{j=0}^{\beta-1} C_j 2^j - \overline{o} + 1, \\
L_{\alpha\beta} &= \mathcal{C} \cup \{\delta_{\alpha\beta}(\overline{o})\},
\end{aligned} \tag{35}$$

where $\mathcal{C}$ is obtained from $\mathbf{v} = \sum_{i=1}^n a_i\mathbf{b}_i$ by replacing the a_i, v_i by the equation (34).

Proposition 8.5 *There exists a quantum algorithm to solve the SVP with probability* $\geq 1 - \varepsilon$ *and in time* $\widetilde{O}(m(n^{7.5} + m^{2.5})(n^3 + \log h)\log^{4.5} h\kappa^2 \log\frac{1}{\varepsilon})$*, where* $h = ||B||_\infty$ *and* κ *is the maximal condition number of* $L_{\alpha\beta}$.

Proof The numbers of $\{A_{i,j}\}$, $\{V_{i,j}\}$, and $\{C_j\}$ in (34) and (35) are $n\log_2(2b_B)$, $m\log_2(\sqrt{m}||B||_\infty)$ and $\log_2(m||B||_\infty^2)$, respectively. So, $N_{L_{\alpha\beta}} = n\log_2(2b_B) + m\log_2(\sqrt{m}||B||_\infty) + \log_2(m||B||_\infty^2) = \widetilde{O}(n^3\log h + m\log h)$. The total sparseness of $\mathcal{C}$ is $O(m(\log(\sqrt{m}\|B\|_\infty) + n\log(2b_B)))$ and the total sparseness of $\delta_{\alpha\beta}(\overline{o})$ is

$\log_2(m||B||_\infty^2)+m\log^2(\sqrt{m}||B||_\infty)$. So the total sparseness of $L_{\alpha\beta}$ is $T_{L_{\alpha\beta}} = m(n\log(2b_B)+\log(\sqrt{m}\|B\|_\infty))+m\log^2(\sqrt{m}||B||_\infty)+\log_2(m||B||_\infty^2) = \widetilde{O}(mn^3\log h + m\log^2 h)$. By Theorem 3.6, the SVP can be solved in time $\widetilde{O}(N_{L_{\alpha\beta}}^{2.5} T_{L_{\alpha\beta}}\kappa^2 \log\frac{1}{\varepsilon}\log(m||B||_\infty^2)) = \widetilde{O}(m(n^{7.5}+m^{2.5})(n^3+\log h)\log^{4.5} h\kappa^2\log\frac{1}{\varepsilon})$. ∎

The CVP can be solved similar to SVP, where the only difference is that the objective function is $o = \|\mathbf{v}-\mathbf{b_0}\|_2^2 - 1 < m(||B||_\infty + ||\mathbf{b_0}||_\infty)^2$. Similar to Proposition 8.5, we have

Proposition 8.6 *There exists a quantum algorithm to solve the CVP with probability* $\geq 1-\varepsilon$ *and in time* $\widetilde{O}(m(n^3\log h + m\log h + \log h_0)^{2.5}(n^3\log h + \log^2(h+h_0))\kappa^2\log\frac{1}{\varepsilon})$, *where* $h = ||B||_\infty$, $h_0 = ||\mathbf{b}_0||_\infty$, *and* κ *is the condition number of the problem.*

9 Quantum algorithm to recover the private key for NTRU

In this section, we will give a quantum algorithm to recover the private key of NTRU from its known public key.

The NTRU cryptosystem depends on three integer parameters (N,p,q) and two sets $\mathcal{L}_f, \mathcal{L}_g$ of polynomials in $\mathbb{Z}[X]$ with degree $N-1$. Note that p and q need not to be prime, but we will assume that $\gcd(p,q)=1$, and q is always considerably larger than p. Denote $\mathbb{Z}_k$ to be the ring $\mathbb{Z}/(k) = \{0,1,\ldots,k-1\}$ for any $k\in\mathbb{Z}_{>0}$. We work in the ring $R = \mathbb{Z}[X]/(X^N-1)$. An element $F\in R$ will be written as a polynomial or a vector,

$$F = \sum_{i=0}^{N-1} F_i x^i = (F_0, F_1, \ldots, F_{N-1})^{\mathrm{T}}. \tag{36}$$

Given two positive integers d_f and d_g, we set

$$\mathcal{L}_f = \{f\in R \,|\, f \text{ has } d_f \text{ coefficients } 1,\ d_f-1 \text{ coefficients } -1, \text{ and the rest } 0\}, \tag{37}$$

$$\mathcal{L}_g = \{g\in R \,|\, g \text{ has } d_g \text{ coefficients } 1,\ d_g \text{ coefficients } -1, \text{ and the rest } 0\}. \tag{38}$$

Let $f\in L_f$ be invertible both $\pmod p$ and $\pmod q$. The private key for NTRU is f and the public key is $h = gf^{-1} \pmod q$ for some $g\in\mathcal{L}_g$. A set of parameters could be $(N,p,q) = (107,3,64)$, $d_f = 15$, and $d_g = 12$ [21].

We need to find f from h. We will reduce this problem to an equation solving problem over the finite rings $\mathbb{Z}_p$ and $\mathbb{Z}_q$. Set $f = (f_0, f_1, \ldots, f_{N-1})^{\mathrm{T}}$, $g = (G_0, G_1, \ldots, g_{N-1})^{\mathrm{T}}$, $f^{-1} \pmod p = \mathbf{p} = (p_0, p_1, \ldots, p_{N-1})^{\mathrm{T}}$, $f^{-1} \pmod q = \mathbf{q} = (q_0, q_1, \ldots, q_{N-1})^{\mathrm{T}}$, and $h = (h_0, h_1, \ldots, h_{N-1})^{\mathrm{T}}$. Thus, we have the following equations:

$$f\in\mathcal{L}_f \iff 2d_f = \sum_{i=0}^{N-1} f_i^2 + 1, \sum_{i=0}^{N-1} f_i = 1 \text{ and each } f_i^3 - f_i = 0; \tag{39}$$

$$g\in\mathcal{L}_g \iff 2d_g = \sum_{i=0}^{N-1} g_i^2, \sum_{i=0}^{N-1} g_i = 0 \text{ and each } g_i^3 - g_i = 0; \tag{40}$$

$$h = gf^{-1} \pmod q \iff \sum_{j+k=i,i+N} h_j f_k \equiv g_i \pmod q \text{ for } i = 0,1,\ldots,N-1; \tag{41}$$

$$f^{-1} \pmod q \text{ exists} \iff \sum_{j+k=i,i+N} q_j f_k \equiv \delta_{0i} \pmod q \text{ for } i = 0,1,\ldots,N-1; \tag{42}$$

$$f^{-1} \pmod p \text{ exists} \iff \sum_{j+k=i,i+N} p_j f_k \equiv \delta_{0i} \pmod p \text{ for } i = 0,1,\ldots,N-1, \tag{43}$$

where $\delta_{0i} = 1$ for $i = 0$ and $\delta_{0i} = 0$ for $i \neq 0$. Let $\mathbb{X} = \{f_i, g_i, h_i, p_i, q_i \,|\, i = 0, 1, \ldots, N-1\}$, and

$$\begin{aligned}
\mathcal{F}_1 &= \{2d_f = \sum_{i=0}^{N-1} f_i^2 + 1, 2d_g = \sum_{i=0}^{N-1} g_i^2, \\
&\quad \sum_{i=0}^{N-1} f_i - 1, \sum_{i=0}^{N-1} g_i, f_i^3 - f_i, g_i^3 - g_i, i = 0, 1, \ldots, N-1\} \subset \mathbb{C}[f, g], \quad (44)\\
\mathcal{F}_2 &= \{ \sum_{j+k=i,i+N} h_j f_k - g_i, \sum_{j+k=i,i+N} q_j f_k - \delta_{0i} \,|\, i = 0, 1, \ldots, N-1\} \subset \mathbb{Z}_q[f, g, h, \mathbf{q}], \quad (45)\\
\mathcal{F}_3 &= \{ \sum_{j+k=i,i+N} p_j f_k - \delta_{0i} \,|\, i = 0, 1, \ldots, N-1\} \subset \mathbb{Z}_p[f, g, h, \mathbf{p}]. \quad (46)
\end{aligned}$$

Note that $\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3$ are over $\mathbb{C}$, $\mathbb{Z}_q$, $\mathbb{Z}_p$, respectively. We can modify the method given in Section 3.1 to solve the equation system $\mathcal{F}_1 = \mathcal{F}_2 = \mathcal{F}_3 = 0$.

We first give a simpler treatment for $\mathcal{F}_1$. Let $\mathbb{Z}_{\rm bit} = \{F_{i1}, F_{i2}, G_{i1}, G_{i2}, i = 0, 1, \ldots, N-1\}$ be Boolean variables, $f_i = F_{i1}+F_{i2}-1$ and $g_i = G_{i1}+G_{i2}-1$. Then, the constraints $f_i^3 = f_i$ and $g_i^3 = g_i$ are automatically satisfied. When $F_{i1} = 0, F_{i2} = 1$ and $F_{i1} = 1, F_{i2} = 0$, we both have $f_i = 0$. To avoid this redundancy, we add an additional equation $F_{i1}F_{i2} - F_{i2}$. We have $2d_f = \sum_{i=0}^{N-1} f_i^2 + 1 = \sum_{i=0}^{N-1}(F_{i1} + F_{i2} - 1)^2 + 1 = \sum_{i=0}^{N-1}(F_{i1} - F_{i2}) + N + 1 \pmod{(F_{i1}^2 - F_{i1}, F_{i2}^2 - f_{i2}, F_{i1}F_{i2} - F_{i2})}$. Similarly, $d_g = \sum_{i=0}^{N-1}(G_{i1} - G_{i2}) + N \pmod{(G_{i1}^2 - G_{i1}, G_{i2}^2 - G_{i2}, G_{i1}G_{i2} - G_{i2})}$. Then, $\mathcal{F}_1$ is equivalent to

$$\begin{aligned}
\mathcal{F}_{11} &= \{\sum_{i=0}^{N-1}(F_{i1} - F_{i2}) + N + 1 - 2d_f, \sum_{i=0}^{N-1}(G_{i1} - G_{i2}) + N - 2d_g, \\
&\quad \sum_{i=0}^{N-1}(F_{i1} + F_{i2} - 1) - 1, \sum_{i=0}^{N-1}(G_{i1} + G_{i2} - 1), \\
&\quad F_{i1}F_{i2} - F_{i2}, G_{i1}G_{i2} - G_{i2}, i = 0, 1, \ldots, N-1, \subset \mathbb{C}[\mathbb{F}_{\rm bit}]\}, \quad (47)
\end{aligned}$$

where $\mathbb{F}_{\rm bit} = \{F_{ij}, G_{ij} \,|\, i = 0, 1, \ldots, N-1;\ j = 1, 2\}$.

We can compute $B(\mathcal{F}_2) \subset \mathbb{Z}_p[\mathbb{X}_{\rm bit}]$ and $B(\mathcal{F}_3) \subset \mathbb{Z}_p[\mathbb{X}_{\rm bit}]$ defined in (6) by setting $q_i = \theta_{q-1}(Q_{i0}, Q_{i1}, \ldots, Q_{i\lfloor \log_2(q-1)\rfloor})$ and $p_i = \theta_{p-1}(P_{i0}, P_{i1}, \ldots, P_{i\lfloor \log_2(p-1)\rfloor})$, where

$$\begin{aligned}
\mathbb{X}_{\rm bit} &= \{F_{i1}, F_{i2}, G_{i1}, G_{i2} \,|\, i = 0, 1, \ldots, N-1\} \cup \\
&\quad \{P_{ij} \,|\, i = 0, 1, \ldots, N-1, j = 0, 1, \ldots, \lfloor \log_2(p-1)\rfloor\} \cup \\
&\quad \{Q_{ij} \,|\, i = 0, 1, \ldots, N-1, j = 0, 1, \ldots, \lfloor \log_2(q-1)\rfloor\}
\end{aligned}$$

Note that $\mathcal{F}_2$ and $\mathcal{F}_3$ are already MQ, we can compute $P(\mathcal{F}_2)$ and $P(\mathcal{F}_3)$ as in (9). Therefore, we can use Algorithm **QBoolSol** to find a Boolean solution $\check{\mathbb{X}}$ for

$$\mathcal{F}_{\rm NTRU} = \mathcal{F}_{11} \cup P(\mathcal{F}_2) \cup P(\mathcal{F}_3) \subset \mathbb{C}[\mathbb{X}_{\rm bit}].$$

Finally set $\check{f}_i = \check{F}_{i1} + \check{F}_{i2} - 1$, and we have a possible private key $\check{f} = (\check{f}_0, \check{f}_1, \ldots, \check{f}_{N-1})$.

Proposition 9.1 *There is a quantum algorithm to obtain a private key f from the public key h in time $\widetilde{O}(N^{4.5} \log^{3.5} q\kappa^2 \log 1/\varepsilon)$ with probability $\geq 1 - \varepsilon$, where κ is the condition number for $\mathcal{F}_{\rm NTRU}$.*

Proof Only the complexity need to be considered. $T_{\mathcal{F}_2} = 2N^2+N+1 = O(N^2)$, $T_{\mathcal{F}_3} = N^2+1 = O(N^2)$, $T_{\mathcal{F}_{11}} = O(N)$. Note that $\mathcal{F}_{11}$ are already Boolean polynomials over $\mathbb{C}$, we need do nothing to it. For $\mathcal{F}_2$, since $q_i = \theta_{q-1}(Q_{i0}, Q_{i1}, \ldots, Q_{i\lfloor \log_2(q-1)\rfloor})$ has $O(\log q)$ terms, $f_k = F_{1k} + F_{2k} - 1$, $g_j = G_{1j} + G_{2j} - 1$, and

$F_{1k}, F_{2k}, G_{1j}, G_{2j}$ are Boolean variables, we have $T_{P(\mathcal{F}_2)} = T_{B(\mathcal{F}_2)} + O(N \log q) = O(N^2 \log q)$. Similarly, $T_{P(\mathcal{F}_3)} = O(N^2 \log p)$. Therefore, we have $T_{\mathcal{F}_{\mathrm{NTRU}}} = O(N^2(\log q + \log p)) = O(N^2 \log q)$ and $N_{\mathcal{F}_{\mathrm{NTRU}}} = O(N \log q + N \log N + N \log\log q) = \widetilde{O}(N \log q)$ by Lemma 3.3, where we can ignore p considering $p \ll q$. By Theorem 3.6, we can obtain a possible private key f in time $\widetilde{O}(N^{4.5} \log^{3.5} q\kappa^2 \log 1/\varepsilon)$. ▮

In the design of NTRU, it is assumed that the size of f and g are small. We can use the methods given in Section 6.1 to find f and g that have the smallest $d_f + d_g$.

Proposition 9.2 *There is a quantum algorithm to obtain a private key f from the public key h such that $d_f + d_g$ is minimal in time $\widetilde{O}(N^{4.5} \log^{3.5} q\kappa^2 \log 1/\varepsilon)$ with probability $\geq 1 - \varepsilon$, where κ is the condition number for $\mathcal{F}_{\mathrm{NTRU}}$.*

Proof Remove $\sum_{i=0}^{N-1}(F_{i1} - F_{i2}) + N + 1 - 2d_f$ and $\sum_{i=0}^{N-1}(G_{i1} - G_{i2}) + N - 2d_g$ from $\mathcal{F}_1$ and still denote $\mathcal{F}_{\mathrm{NTRU}} = \mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{F}_3$. We can use the objective function $o = (2d_f - 1 - N) + 2d_g - N - 1 = \sum_{i=0}^{N-1}(F_{i1} - F_{i2} + G_{i1} - F_{i2}) - 1$ which satisfies $0 \leq o < 4N$. Following (18), we have $\delta_{\alpha\beta} = \alpha + \sum_{j=0}^{\beta-1} E_j 2^j - o$ and $L_{\alpha\beta} = \mathcal{F}_{\mathrm{NTRU}} \cup \{\delta_{\alpha\beta}\} \subset \mathbb{C}[\mathbb{X}_{\mathrm{bit}}, \mathbb{E}_{\mathrm{bit}}]$. Then we can use Algorithm 5.3 to find a private key f which minimizes o. By the proof of Proposition 9.1, we have $T_{\mathcal{F}_{\mathrm{NTRU}}} = O(N^2 \log q)$ and $N_{\mathcal{F}_{\mathrm{NTRU}}} = O(N \log q)$. Then, $T_{L_{\alpha\beta}} = \widetilde{O}(N^2 \log q)$ and $N_{L_{\alpha\beta}} = O(N \log q)$. By Theorem 5.4, the complexity is $\widetilde{O}(N^{4.5} \log^{3.5} q\kappa^2 \log 1/\varepsilon \log N) = \widetilde{O}(N^{4.5} \log^{3.5} q\kappa^2 \log 1/\varepsilon)$. ▮

For the parameters recommended in [21], $(N, p, q) = (107, 3, 64)$, $(N, p, q) = (167, 3, 128)$, $(N, p, q) = (503, 3, 256)$, and $\varepsilon = 1\%$, the complexities of Proposition 9.2 are given in the following table.

Table 1 Complexities of the quantum algebraic attack on NTRU

N	q	p	Complexity	Desired Complexity
107	64	3	$2^{42}\kappa^2$	3^N
167	128	3	$2^{46}\kappa^2$	3^N
503	256	3	$2^{54}\kappa^2$	3^N

In Table 1, κ is the condition number of the corresponding equation systems. From the table, this main part of the complexity is relatively low comparing to its desired security 3^N if κ is small, which implies that the NTRU is safe only if its condition number is large.

10 Conclusion

In this paper, we give quantum algorithms for two basic computational problems: polynomial system solving over a finite field and the optimization problem where the arguments either take values from a finite field or are bounded integers. The complexities of these quantum algorithms are polynomial in the input size, the maximal degree of the inequality constraints, and κ which is the condition number of the associated matrix of the problem. So, we achieve exponential speedup for these problems when the condition number is small.

The optimization problem considered in this paper covers many NP-hard problems as special cases. In particular, the proposed algorithms are used to give quantum algorithms for several fundamental computational problems in cryptography, including the polynomial system with noise, the short solution problem, the shortest vector problem, and the NTRU cryptosystem. The complexity for all of these problems is polynomial in the input size and their condition numbers, which means that these problems are difficult to solve by a quantum computer if and only if their condition numbers are large. As a consequence, the NTRU cryptosystem as well as the candidates recently proposed for post-quantum standard of public key cryptosystems [4] are safe against quantum computer attacks only if the condition number of its equation system is large.

The main idea of the algorithm is to convert the equality and inequality constraints of the optimization problem into polynomial equations in Boolean variables and then convert the finding of the minimal value of the objective function into several problems of finding the Boolean solutions for polynomial systems over $\mathbb{C}$, that is B-POSSO. Then the quantum algorithm from [13] is used to find Boolean solutions for these polynomial systems.

As we just mentioned that the optimization problem is reduced into the B-POSSO problem. It is interesting to give a description for all the problems that can be efficiently reduced to B-POSSO. It is also interesting to see whether it is possible to combine the reduction methods introduced in this paper with traditional algorithms for polynomial system solving such as the Gröbenr basis method [3] and the characteristic set method [17] to obtain better traditional algorithms for polynomial system solving and optimization over finite fields. Finally, in order to know the exact complexity of the algorithm proposed in this paper, we need to know the condition number, which is a main future problem for study.

Conflict of Interest

The authors declare no conflict of interest.

References

[1] Ajtai, M., Generating hard instances of lattice problems, *STOC'96*, 99-108, ACM Press, 1996.

[2] Albrecht, M.R. and Cid, C., Cold boot key recovery by solving polynomial systems with noise, *ACNS*, 57-72, 2011.

[3] Albrecht, M.R., Cid, C., Faugére, J.C., Fitzpatrick, R., Perret, L., Algebraic algorithms for LWE problems. Inria hal-01072721, 2014.

[4] Albrecht M.R., Curtis B.R., Deo A., Davidson, A., Player R., Postlethwaite E.W., Virdia F. and Wunderer T., Estimate all the {LWE, NTRU} schemes!. In: Catalano D., De Prisco R. (eds) Security and Cryptography for Networks. SCN 2018. LNCS, vol 11035. Springer, Cham.

[5] Ambainis, A., Variable time amplitude amplification and a faster quantum algorithm for solving systems of linear equations, *Proc. STACS*, 636-647, 2012.

[6] Aono Y., Nguyen P.Q., Shen Y., Quantum lattice enumeration and tweaking discrete pruning, https://eprint.iacr.org/2018/546, 2018.

[7] Arora, S. and Ge, R., New algorithms for learning in presence of errors, *ICALP, LNCS 6755*, L. Aceto, M. Henzinger, and J. Sgall, editors, 403-415, Springer Verlag, 2011.

[8] Balas E., An additive algorithm for solving linear programs with zero-one cariables. INFORMS, 1965.

[9] Bardet, M., Faugère, J.C., Salvy, B., Spaenlehauer, P.J., On the complexity of solving quadratic Boolean systems, *Journal of Complexity*, 29(1), 53-75, 2013.

[10] Berlekamp, E.R., McEliece, R.J., van Tilborg, H.C.A., On the inherent intractability of certain coding problems, *IEEE Trans. on Information Theory*, 24(3), 384-386, 1978.

[11] Blum, A., Kalai, A., Wasserman, H., Noise-tolerant learning, the parity problem, and the statistical query model, *Journal of the ACM*, 50(4), 506-519, 2003.

[12] Bryant, R., Graph-based algorithm for Boolean function manipulation, *IEEE Transactions on Computing*, C-35, 677-691, 1986.

[13] Y.A. Chen and X.S. Gao. Quantum Algorithms for Boolean Equation Solving and Quantum Algebraic Attack on Cryptosystems. *Journal of Systems Science and Complexity*, 35(1), 373-412, 2022.

[14] Childs, A.M., Kothari, R., Somma, R.D., Quantum algorithm for systems of linear equations with exponentially improved dependence on precision, *SIAM J. Comput.*, 46(6), 1920-1950, 2017.

[15] Courtois, N., Klimov, A., Patarin, J., Shamir, A., Efficient algorithms for solving overdefined systems of multivariate polynomial equations, *Eurocrypt'00*, LNCS, vol. 1807, 392-407, Springer, 2000.

[16] Ding, J., Gower, J.E., Schmidt, D.S., *Multivariate Public Key Cryptosystems*, Springer, 2006.

[17] Gao, X.S. and Huang, Z., Characteristic set algorithms for equation solving in finite fields, *Journal of Symbolic Computation*, 47, 655-679, 2012.

[18] Genova K. and Guliashki V., Linear integer programming methods and approaches - a survey, *Cybernetics and Information Technologies*, 11(1) , 2011

[19] Håstad, J., Some optimal inapproximability results, *Journal of the ACM*, 48, 798-859, 2009.

[20] Harrow, A.W., Hassidim, A., Lloyd, S., Quantum algorithm for linear systems of equations. *Physical Review Letters*, 103(15), 150502, 2009.

[21] Hoffstein, J., Pipher, J., Silverman, J.H., NTRU: A ring-based public key cryptosystem, *Algorithmic Number Theory, ANTS'98*, LNCS 1423, 267-288, Springer, Berlin, Heidelberg, 1998.

[22] Huang, Z. and Lin, D., Solving polynomial systems with noise over F2: revisited, *Theoretical Computer Science*, 676, 52-68, 2017.

[23] Kadowaki, T. and Nishimori, H., Quantum annealing in the transverse Ising model, *Phys. Rev. E*, 58, 5355-5363, 1998.

[24] Karp, R.M., Reducibility among combinatorial problems. *Complexity of Computer Computations*, 85-103, 1972.

[25] Laarhoven, T., Sieving for shortest vectors in lattices using angular locality-sensitive hashing, *Proc. CRYPTO 2015 -Part I*, Vol. 9215, LNCS, Springer, 2015.

[26] Li, H., Xia, B., Zhao, T., Square-Free Pure Triangular Decomposition of Zero-Dimensional Polynomial Systems *Journal of Systems Science and Complexity*, 36(6): 2661-2680, 2023.

[27] Li, Z., Wu, B., Lin, D., Algebraic-Differential Attacks on a Family of Arithmetization-Oriented Symmetric Ciphers, *Journal of Systems Science and Complexity*, 36(6): 2681-2702, 2023.

[28] Liu, C., Tian, T., Qi, W., A New Method for Searching Cubes and Its Application to 815-Round Trivium, *Journal of Systems Science and Complexity*, 25 April 2023 Pages: 2234 - 2254, 2024.

[29] Liu, Z., Pan, Y., Zheng, J., Polynomial-Time Key-Recovery Attacks Against NTRUReEncrypt from ASIACCS'15, *Journal of Systems Science and Complexity*, 37(3): 1308-1325, 2024.

[30] Ma, G. and Li, H., On the Security of Homomorphic Encryption Schemes with Restricted Decryption Oracles, *Journal of Systems Science and Complexity*, 37(5), 2240-2261, 2024.

[31] Peikert, C., Public-key cryptosystems from the worst-case shortest vector problem, *STOC2009)*, 333-342, 2009.

[32] Regev, O., On lattices, learning with errors, random linear codes, and cryptography. In *ACM STOC*, 84-93, 2005.

[33] Shao, C., Li, Y., Li, H., Quantum Algorithm Design: Techniques and Applications *Journal of Systems Science and Complexity*, 32(1): 375-452, 2019.

[34] Shewchuk, J.R., An introduction to the conjugate gradient method without the agonizingpain, Tech. Rep. CMU-CS-94-125, Carnegie Mellon University, Pittsburgh, Pennsylvania, 1994.

[35] Storjohann, A., *Algorithms for matrix canonical forms*, PhD Thesis, Swiss Federal Institute of Technology, 2000.

[36] von zur Gathen, J. and Gerhard, J., "Modern Computer Algebra," Cambridge University Press, 1999.

[37] Xiao, S. and Zeng, G., Decomposing the Radicals of Polynomial Ideals by Rational Univariate Representations, *Journal of Systems Science and Complexity*, 36(6): 2703-2724, 2023.

[38] Xie, X. and Tian, T., A Low-Complexity Key-Recovery Attack on 6-Round Midori64, *Journal of Systems Science and Complexity*, 36(4): 1738-1756, 2023.

[39] Yan, X, L. Tan, Qi, W., Non-Existence of One-Byte Active Impossible Differentials for 5-Round AES in the Master-Key Setting, *Journal of Systems Science and Complexity*, 36(3): 1336-1350, 2023.

[40] Yap, C.K., *Fundamental Problems of Algorithmic Algebra*, Oxford Press, 2000.

[41] Zhao, S.W. and Gao, X.S., Minimal achievable approximation ratio for MAX-MQ in finite fields, *Theoretical Computer Science*, 410, 2285-2290, 2009.