Improved Complexity Estimates for Underdetermined MQ Systems via Generalized Variable Partitioning

Hideki Asanuma[1*], Yilong Chen[1*], Hiroki Furue[2], Kosuke Sakata[1], and Tsuyoshi Takagi[1]

[1] Department of Mathematical Informatics, The University of Tokyo, Tokyo, Japan
{sakata-kosuke-rb,takagi}@g.ecc.u-tokyo.ac.jp
[2] NTT Social Informatics Laboratories, Tokyo, Japan
hiroki.furue@ntt.com

Abstract. Multivariate quadratic (MQ) signature schemes are an important class of post-quantum digital signatures. These schemes rely on the hardness of solving underdetermined MQ systems, where the number of variables n exceeds the number of equations m. Therefore, analyzing the efficiency of algorithms for underdetermined MQ systems is essential for evaluating the security of MQ-based signature schemes. Several algorithms have been proposed to solve underdetermined MQ systems efficiently. Among them, Hashimoto's method is one of the best known partition-based direct attacks; it partitions the variables into three groups and reduces the original problem to two smaller MQ subproblems. In this paper, we propose a generalized partition-based algorithm for solving underdetermined MQ systems. The proposed algorithm extends Hashimoto's method by partitioning the variables into more groups and reducing the original problem to smaller MQ subproblems. We derive its applicability conditions and time complexity, and develop an efficient parameter search strategy to find the best partition parameters without a naive exhaustive search. Finally, we evaluate the proposed algorithm on parameter sets of MAYO, QR-UOV, and SNOVA, and compare the resulting estimates with those of existing algorithms for underdetermined MQ systems. In the classical case, the proposed method improves on Hashimoto's method for some parameter sets; in particular, it reduces the estimate for MAYO1 from 2^{156} to 2^{145}. In the quantum case, the proposed method gives estimates smaller than both Just Guess and Hashimoto's method for many parameter sets.

Keywords: post-quantum cryptography, multivariate-based cryptography, MQ problem, underdetermined MQ problem

1 Introduction

Widely used public-key cryptosystems such as RSA and elliptic-curve cryptography (ECC) can be broken in polynomial time on a quantum computer using

* This work was carried out while these authors were at the University of Tokyo.

Shor's algorithm [24]. This threat motivated the National Institute of Standards and Technology (NIST) to launch the Post-Quantum Cryptography (PQC) standardization project, which aims to develop cryptographic schemes based on mathematical problems believed to resist both classical and quantum attacks. Major directions in PQC include lattice-based cryptography, code-based cryptography, hash-based signatures, isogeny-based cryptography, and multivariate quadratic (MQ) cryptography.

In July 2022, NIST selected CRYSTALS-Kyber [6], CRYSTALS-Dilithium [10], SPHINCS+ [2], and FALCON [12] for post-quantum cryptographic standardization. Standards based on the first three schemes, namely ML-KEM, ML-DSA, and SLH-DSA, were finalized in August 2024, while a standard based on FALCON is also under development. In addition, NIST selected HQC [1] for standardization as an additional key-encapsulation mechanism. In parallel, NIST recently advanced nine candidates to the third round of its additional call for post-quantum digital signatures [22]. These candidates include the multivariate signature schemes UOV [20], MAYO [4], QR-UOV [14], and SNOVA [26].

In general, solving a system of multivariate quadratic equations over a finite field—the multivariate quadratic (MQ) problem—is NP-hard. MQ-based cryptography is constructed based on this hardness assumption and uses multivariate quadratic polynomials as public keys. One natural attack on MQ-based cryptography is to solve the MQ system defined by the public key directly. Most practical schemes currently use underdetermined MQ systems, in which the number of variables n exceeds the number of equations m. Thus, analyzing the efficiency of algorithms for such systems is essential for evaluating the security of cryptosystems.

Various algorithms have been proposed to solve underdetermined MQ systems more efficiently than the generic hybrid approach, which fixes some variables in advance and then solves the resulting reduced system using MQ solvers. In 2012, Thomae and Wolf [25] proposed a method for even q that reduces the original problem to a smaller MQ instance with fewer variables and equations. In 2014, Cheng et al. [7] extended this approach to arbitrary q under milder applicability conditions. In 2021, Furue et al. [15] incorporated the hybrid approach into Thomae–Wolf's method.

In 2023, Hashimoto [18] further refined the algebraic transformations used in Furue et al.'s method. Hashimoto's method partitions the variables into three groups and applies linear transformations so that the original underdetermined MQ problem is reduced to smaller subproblems. After solving a system of linear equations for one group of variables, the method first solves an MQ problem coming from one of the remaining variable groups, and then solves another reduced MQ problem for the remaining variables. In this way, the original problem is eventually handled by solving two smaller MQ problems, together with the cost of constructing the transformations. As a result, Hashimoto's algorithm has served as one of the best known direct attacks for several proposed multivariate signature schemes.

Recently, May, Ostuzzi, and Ressler proposed the Just Guess algorithm for underdetermined MQ systems [21]. Their approach follows a different direction from Hashimoto's method. Instead of reducing the problem to several smaller MQ subproblems to be solved using MQ solvers, the Just Guess algorithm transforms the system so that the nonlinear part becomes a sequence of one-variable quadratic equations. Then, the remaining difficulty is shifted to exhaustive search. As a result, the attack becomes more efficient in the quantum setting, because the exhaustive-search part can benefit from Grover's algorithm.

1.1 Contributions of This Research

In this study, we propose a generalized partition-based algorithm that extends Hashimoto's method by partitioning the variables into more groups and reducing the original problem to smaller MQ subproblems. We derive its applicability conditions and complexity estimates, and evaluate the proposed algorithm on parameter sets of multivariate signature schemes.

The proposed algorithm partitions the variables into $\ell + 1$ groups for $\ell \geq 2$, whereas Hashimoto's method partitions them into three groups. Indeed, the proposed algorithm includes Hashimoto's method as a special case. The algorithm consists of three steps: **Step 1** applies a sequence of linear transformations to obtain a structured system, **Step 2** solves a linear system for the last variable group, and **Step 3** solves the resulting smaller MQ subproblems. As in the hybrid approach, **Step 2** and **Step 3** are repeated until a solution is found. Hashimoto's method produces two reduced MQ subproblems, while our framework allows the original problem to be decomposed into a larger number of smaller MQ subproblems. This additional flexibility can reduce the total estimated complexity for some parameter sets. We derive explicit applicability conditions and a complexity formula for the proposed algorithm. Since the algorithm involves many integer parameters, we also develop an efficient parameter search strategy to find the best parameters without a naive exhaustive search.

We then evaluate the proposed algorithm for the parameter sets of MAYO, QR-UOV, and SNOVA. We do not include UOV in our evaluation, since its underdeterminedness is relatively small and the proposed algorithm is not expected to be effective for such parameter sets. In the classical case, the proposed method gives estimates comparable to those of Hashimoto's method and improves them for some parameter sets. In particular, for MAYO1, the proposed method reduces the estimate obtained by Hashimoto's method from 2^{156} to 2^{145}. In the quantum case, the proposed method gives estimates smaller than both Just Guess and Hashimoto's method for many parameter sets. In particular, for some parameter sets, it improves on the smaller of the two quantum estimates by more than 10 bits. These results suggest that the generalized variable partitioning used in the proposed method can improve the efficiency of solving underdetermined MQ instances in both the classical and quantum settings. Moreover, for some parameter sets, the partition parameters selected by our parameter search coincide with those corresponding to Hashimoto's method, thereby providing evidence

for the optimality of Hashimoto's method in these cases within our generalized framework.

1.2 Related Work

Independently and concurrently with our work, Ostuzzi [23] proposed an extension of Hashimoto's algorithm based on pseudo-oil subspaces. As in our approach, this method generalizes Hashimoto's algorithm by decomposing the original MQ problem into multiple smaller MQ subproblems that are solved sequentially. Although the two approaches were developed independently, their resulting structures are closely related. In particular, under the corresponding parameter setting, the matrix structure induced by the pseudo-oil decomposition in [23] can be matched, up to permutations of variables and equations, with the zero-block structure obtained by the linear transformations in **Step 1** of our algorithm.

A notable difference between the two approaches lies in how linear equations are used in the reduced MQ subproblems. In [23], linear equations are used to eliminate variables in the final overdetermined MQ problem, whereas our framework also allows variables to be eliminated by b_i linear equations for $2 \leq i \leq \ell - 1$ in the intermediate reduced MQ problems. This means that, in our parametrization, the construction in [23] can be represented by setting $b_i = 0$ for $2 \leq i \leq \ell - 1$, while b_1 corresponds to the number of equations linearized in the last MQ subproblem to be solved. In addition, while [23] focuses on classical attack complexity under several memory-access models, we also analyze the quantum complexity of the generalized partition framework. For the resulting parameter space, we further develop an efficient parameter-search strategy to determine optimized parameters for our framework.

Organization. The rest of this paper is organized as follows. Section 2 recalls the MQ problem and the XL algorithm for the problem. Section 3 reviews existing algorithms for underdetermined MQ systems, with particular focus on Hashimoto's method. Section 4 presents the proposed algorithm and derives its applicability conditions and complexity estimates. Section 5 describes the parameter search strategy used to find optimized partition parameters efficiently. Section 6 reports the complexity evaluation for MAYO, QR-UOV, and SNOVA. Finally, Section 7 concludes the paper.

2 Preliminaries on MQ Problems

This section defines the multivariate quadratic (MQ) problem and reviews the basic solving techniques used later in the paper.

2.1 MQ Problem

Let q be a power of a prime, and let $\mathbb{F}_q$ denote the finite field of size q. An element $\mathbf{a}$ of the n-dimensional vector space $\mathbb{F}_q^n$ is represented as a column vector:

$$\mathbf{a} = (a_1, \ldots, a_n)^\top \in \mathbb{F}_q^n.$$

For positive integers n and m, let F be a quadratic polynomial map from $\mathbb{F}_q^n$ to $\mathbb{F}_q^m$. That is, for m quadratic polynomials $f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})$ in n variables $\mathbf{x}$, we define

$$F(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))^\top.$$

The Multivariate Quadratic (MQ) problem is defined as follows.

Definition 1. *Given a quadratic polynomial map*

$$F(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))^\top : \mathbb{F}_q^n \to \mathbb{F}_q^m,$$

the MQ problem over $\mathbb{F}_q$ is to find a solution to the system of m quadratic equations in n variables

$$F(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))^\top = \mathbf{0}.$$

In multivariate cryptography, both even- and odd-characteristic fields are used, and small field sizes such as $q = 31$ or $q = 2^8$ are common. We denote the MQ problem in Definition 1 by $\mathrm{MQ}(q, n, m)$. The most naive approach, namely exhaustive search over all assignments to the n variables, requires q^n operations, which is exponential in n.

2.2 XL-Based Solving and Hybrid Approach

Algebraic methods such as Gröbner-basis algorithms [11] and XL-type algorithms [9] are standard approaches to solving the MQ problem. In this paper, we use an XL-based estimate for the cost of solving $\mathrm{MQ}(q, n, m)$ with $n \leq m$. This estimate is combined with the hybrid approach [3, 28, 29], which fixes some variables in advance and then solves the resulting reduced system.

The hybrid approach. Let k be a nonnegative integer satisfying $k \geq n-m$. Assign values $v_1, \ldots, v_k \in \mathbb{F}_q$ to the variables $x_{n-k+1}, \ldots, x_n$. This yields a reduced MQ problem consisting of m quadratic polynomials in the remaining $n-k$ variables $x_1, \ldots, x_{n-k}$. The reduced system is then solved by the XL algorithm. If it has no solution, the procedure is repeated with a different choice of $v_1, \ldots, v_k$ until a solution is found.

We denote the complexity of solving $\mathrm{MQ}(q, n, m)$ with $n \leq m$ by the hybrid approach combined with the XL algorithm as $\mathrm{XL}(q, n, m)$. The complexity $\mathrm{XL}(q, n, m)$ is estimated as follows.

Theorem 1 (Complexity of the hybrid approach with XL). *Let d_{reg} be the smallest degree such that the coefficient of the corresponding power of z in*

$$\frac{(1-z^2)^m}{(1-z)^{n-k+1}}$$

is non-positive [8]. Then the complexity of the hybrid approach combined with the Wiedemann XL algorithm [29], computed as the number of operations in $\mathbb{F}_q$ required to solve non-homogeneous multivariate systems, can be estimated by

$$\mathrm{XL}(q,n,m) = \min_{0\le k\le n}\left\{O\left(q^k\cdot 3\cdot\binom{n-k+2}{2}\cdot\binom{d_{\mathrm{reg}}+n-k}{d_{\mathrm{reg}}}^2\right)\right\}.$$

The XL-based estimate in Theorem 1 is used as a basic formula for evaluating the computational hardness of MQ instances. In particular, this type of estimate is used in the derivation of secure parameters for UOV-type and related multivariate signature schemes, including UOV [20], MAYO [4, 5], QR-UOV [14, 13], and SNOVA [26, 27], where the parameters are chosen so that known direct attacks based on solving the corresponding MQ systems remain above the target security level.

3 Existing Algorithms

This section reviews efficient algorithms for solving the underdetermined MQ problem. In particular, we focus on the method of Hashimoto [18], which forms the basis of our proposed approach.

3.1 Previous Research on Underdetermined MQ Problems

For the underdetermined MQ problem $\mathrm{MQ}(q,n,m)$ with $n > m$, several algorithms have been proposed that first apply algebraic transformations and then reduce the original problem to smaller MQ subproblems.

In 2012, Thomae and Wolf [25] proposed an algorithm for even q that reduces the problem to $\mathrm{MQ}(q,m-a,m-a)$ for an integer a with $1\le a\le m$ satisfying $n\ge(a+1)m$.

In 2014, Cheng et al. [7] proposed an algorithm for arbitrary q that reduces the problem to $\mathrm{MQ}(q,m-a,m-a)$ under the condition $n\ge(a+1)(m-a/2)$, which relaxes the condition for a from Thomae–Wolf's method.

In 2021, Furue et al. [15] incorporated the hybrid approach into Thomae–Wolf's method. For even q, an integer a with $1\le a\le m$, and the hybrid parameter k, they obtained the time complexity of

$$q^k\mathrm{XL}(q,m-a-k,m-a)$$

when $n\ge(a+1)(m-k)+k$.

Furthermore, in 2023, Hashimoto [18] improved the algebraic transformations of Furue et al.'s method. The details of Hashimoto's method are described in the following section.

These analyses rely on an assumption about the probability that an MQ system has a solution, and we use the same assumption for the proposed algorithm.

Assumption 1 (Assumption on the Solvability of the MQ Problem). We use the standard heuristic that a random system of m equations in n variables over $\mathbb{F}_q$ has expected number of solutions q^{n-m}. In particular, when $n = m$, we regard the probability that a solution exists as being of constant order.

The same heuristic is also used in the complexity analysis of Hashimoto's method reviewed below. In particular, when estimating the success probability of the reduced MQ subproblems and the resulting number of repetitions, Hashimoto's algorithm treats square random MQ systems as having a solution with constant-order probability. Our analysis adopts the same convention, so that the proposed method and Hashimoto's method are compared under a common solvability heuristic.

3.2 Hashimoto's Method

This subsection reviews the details of Hashimoto's method [18], which is one of the algorithms for solving underdetermined MQ problems.

Let $f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})$ be quadratic polynomials in n variables $\mathbf{x} = (x_1, \ldots, x_n)^\top$. Each f_i with $1 \le i \le m$ can be written in the form

$$f_i(\mathbf{x}) = \mathbf{x}^\top F_i \mathbf{x} + (\text{affine terms}), \tag{1}$$

where $F_i \in \mathbb{F}_q^{n \times n}$. When $\mathrm{char}(\mathbb{F}_q) \neq 2$, one may take F_i to be symmetric.

Let a and k be non-negative integers such that $a + k \le m$. The variables $\mathbf{x}$ in (1) are partitioned into three groups:

$$\begin{aligned} X &: x_1, \ldots, x_{m-a-k+1}, \\ Y &: x_{m-a-k+2}, \ldots, x_{m-k}, \\ Z &: x_{m-k+1}, \ldots, x_n, \end{aligned}$$

where $|X| = m - a - k + 1$, $|Y| = a - 1$, and $|Z| = n - m + k$.

Then, Hashimoto's method can be described as follows:

Step 1. First, apply a linear transformation to $\mathbf{x}$ in (1). More precisely, find a matrix

$$M_1 \in \mathbb{F}_q^{(n-m+a+k-1) \times (m-a-k+1)}$$

such that the transformed polynomials

$$\tilde{f}_i(\mathbf{x}) := f_i\left(\begin{pmatrix} I_{m-a-k+1} \\ M_1 \quad I_{n-m+a+k-1} \end{pmatrix} \mathbf{x}\right) \quad (1 \le i \le m) \tag{2}$$

satisfy, for $1 \le j \le a$,

$$\tilde{f}_j(\mathbf{x}) = \mathbf{x}^\top \begin{pmatrix} 0_{m-a-k+1} & * & * \\ * & *_{a-1} & * \\ * & * & *_{n-m+k} \end{pmatrix} \mathbf{x} + (\text{affine terms}). \tag{3}$$

Next, apply another linear transformation to $\mathbf{x}$ in (3). Find a matrix

$$M_2 \in \mathbb{F}_q^{(n-m+k)\times(a-1)}$$

such that the transformed polynomials

$$\bar{f}_i(\mathbf{x}) := \tilde{f}_i\left(\begin{pmatrix} I_{m-a-k+1} & & \\ & I_{a-1} & \\ & M_2 & I_{n-m+k} \end{pmatrix}\mathbf{x}\right) \quad (1 \le i \le m) \tag{4}$$

satisfy, for $1 \le j \le a-1$,

$$\bar{f}_j(\mathbf{x}) = \mathbf{x}^\top \begin{pmatrix} 0_{m-a-k+1} & 0 & * \\ 0 & *_{a-1} & * \\ * & * & *_{n-m+k} \end{pmatrix}\mathbf{x} + (\text{affine terms}). \tag{5}$$

Under these transformations, $\bar{f}_j(\mathbf{x})$ for $1 \le j \le a$ have the form

$$\begin{aligned} \bar{f}_1 &= \sum_{i=1}^{m-a-k+1} x_i L_{1,i}(Z) + Q_1(Y,Z), \\ &\vdots \\ \bar{f}_{a-1} &= \sum_{i=1}^{m-a-k+1} x_i L_{a-1,i}(Z) + Q_{a-1}(Y,Z), \\ \bar{f}_a &= \sum_{i=1}^{m-a-k+1} x_i L_{a,i}(Y,Z) + Q_a(Y,Z), \end{aligned}$$

where $L_{j,i}$ are linear forms and $Q_1, \ldots, Q_a$ are quadratic polynomials, including linear and constant terms.

Step 2. Solve the system

$$L_{1,i}(Z) = \cdots = L_{a-1,i}(Z) = 0$$

for all $1 \le i \le m-a-k+1$, and substitute a solution for Z. Then the polynomials have the form

$$\begin{aligned} \bar{f}_1 &= Q'_1(Y), \\ &\vdots \\ \bar{f}_{a-1} &= Q'_{a-1}(Y), \\ \bar{f}_a &= \sum_{i=1}^{m-a-k+1} x_i L'_{a,i}(Y) + Q'_a(Y), \end{aligned}$$

where $L'_{a,i}(Y)$ are linear forms and $Q'_1(Y), \ldots, Q'_a(Y)$ are quadratic polynomials in Y.

Step 3. Solve

$$Q'_1(Y) = \cdots = Q'_{a-1}(Y) = 0$$

for Y, and substitute the resulting values. Then solve the remaining equations

$$\bar{f}_j = 0 \qquad (a \leq j \leq m)$$

for X. If a solution exists, recover the corresponding solution for the original variables $\mathbf{x}$ by applying the inverse transformations from **Step 1**. Otherwise, return to **Step 2** and continue the search with a different solution for Z.

The applicability conditions and time complexity of the above algorithm are given by the following theorem:

Theorem 2 (Applicability conditions and complexity of Hashimoto's method [18]). *The applicability condition for Hashimoto's method is:*

$$n \geq \max\{(a+1)(m-k-a+1), a(m-k) - (a-1)^2 + k\}.$$

The time complexity is expressed as:

$$(m-a-k+1)\mathrm{XL}(q,a,a) + q^k(\mathrm{XL}(q,a-1,a-1) + \mathrm{XL}(q,m-a-k,m-a)).$$

Theorem 2, together with the XL-based estimate in Theorem 1, provides concrete direct-attack complexity estimates for underdetermined MQ instances. Such estimates are an important basis for selecting and justifying secure parameters in multivariate signature schemes whose security relies on the hardness of underdetermined MQ problems, including MAYO, QR-UOV, and SNOVA. For example, using the complexity formula of Hashimoto's method with the XL costs from Theorem 1, the parameters $(q,n,m) = (16,860,78)$ corresponding to MAYO1 are estimated to have complexity 2^{156}, as shown in Table 3 below. Similarly, the estimates are 2^{160} for $(q,n,m) = (127,210,54)$, corresponding to QR-UOV Level I, and 2^{188} for $(q,n,m) = (16,464,80)$, corresponding to SNOVA Level I; see Tables 4 and 5 below. These values illustrate the role of Hashimoto's algorithm as one of the benchmark direct attacks used in evaluating the concrete security of such parameter sets.

4 Generalized Partition-Based Algorithm

This section presents a new algorithm for solving underdetermined MQ problems constructed by generalizing the variable partitioning used in Hashimoto's method. By increasing the number of partitions, we aim to further reduce the overall complexity of solving underdetermined MQ problems.

4.1 Overview of the Algorithm

We first set parameters $\ell, a_1, \ldots, a_\ell$ and $b_1, \ldots, b_{\ell-1}$ related to our partitioning method. Here, ℓ and $a_1, \ldots, a_\ell$ are positive integers satisfying $a_1 + \cdots + a_\ell < m$.

For each $1 \leq i \leq \ell - 1$, b_i is an integer satisfying $0 \leq b_i < a_i$. Furthermore, we define $a_{\ell+1} := n - (a_1 + \cdots + a_\ell)$ and $s := m - (a_1 + \cdots + a_\ell)$. The variables $x_1, \ldots, x_n$ are partitioned into $(\ell + 1)$ groups as follows:

$$\begin{aligned} X_1 &: x_1, \ldots, x_{a_1} \ (a_1 \text{ variables}), \\ X_2 &: x_{a_1+1}, \ldots, x_{a_1+a_2} \ (a_2 \text{ variables}), \\ &\vdots \\ X_{\ell+1} &: x_{a_1+\cdots+a_\ell+1}, \ldots, x_n \ (a_{\ell+1} \text{ variables}). \end{aligned}$$

Then, the proposed algorithm is described as follows:

Step 1. As in Hashimoto's method [18], we construct ℓ matrices to transform the original system into specific forms. The transformed equations contain many zero entries across a total of $(a_2 + \cdots + a_\ell + b_1)$ equations. The details of this transformation are given in Subsection 4.2.

We partition the first $a_2 + \cdots + a_\ell + b_1$ equations into $A_1, \ldots, A_{2\ell-2}$ as follows. First, set

$$A_1 = \{f_1, \ldots, f_{a_\ell}\}.$$

For $2 \leq i \leq \ell - 1$, define

$$A_{2\ell-2i} = \{f_{a_{i+1}+\cdots+a_\ell+1}, \ldots, f_{a_{i+1}+\cdots+a_\ell+b_i}\},$$

and

$$A_{2\ell-2i+1} = \{f_{a_{i+1}+\cdots+a_\ell+b_i+1}, \ldots, f_{a_i+\cdots+a_\ell}\}.$$

Finally, define

$$A_{2\ell-2} = \{f_{a_2+\cdots+a_\ell+1}, \ldots, f_{a_2+\cdots+a_\ell+b_1}\}.$$

The remaining equations

$$f_{a_2+\cdots+a_\ell+b_1+1}, \ldots, f_m$$

have no specific structure after the transformation and are used only in the final step for determining X_1. When $b_i = 0$, the corresponding set $A_{2\ell-2i}$ is understood to be empty.

Using the procedure described in Section 4.2, we transform each polynomial f_i belonging to the groups $A_1, \ldots, A_{2\ell-2}$ into a polynomial $\bar{f}_i$ so that the result-

ing polynomials in each group take the following form:

$$
\begin{aligned}
A_1 \ni \bar{f}_j(\mathbf{x}) &= \mathbf{x}^\top \left(\begin{array}{ccc|c} 0_{a_1+\cdots+a_{\ell-1}} & 0 & & * \\ 0 & *_{a_\ell} & & * \\ \hline * & * & & *_{a_{\ell+1}} \end{array}\right) \mathbf{x} + (\text{affine terms}), \\
&\vdots \\
A_{2\ell-2i} \ni \bar{f}_j(\mathbf{x}) &= \mathbf{x}^\top \left(\begin{array}{cccc|c} 0_{a_1+\cdots+a_{i-1}} & 0 & 0 & & * \\ 0 & 0_{a_i} & * & & * \\ 0 & * & *_{a_{i+1}+\cdots+a_\ell} & & * \\ \hline * & * & * & & *_{a_{\ell+1}} \end{array}\right) \mathbf{x} + (\text{affine terms}), \\
A_{2\ell-2i+1} \ni \bar{f}_j(\mathbf{x}) &= \mathbf{x}^\top \left(\begin{array}{ccc|c} 0_{a_1+\cdots+a_{i-1}} & 0 & & * \\ 0 & *_{a_i+\cdots+a_\ell} & & * \\ \hline * & * & & *_{a_{\ell+1}} \end{array}\right) \mathbf{x} + (\text{affine terms}), \\
&\vdots \\
A_{2\ell-2} \ni \bar{f}_j(\mathbf{x}) &= \mathbf{x}^\top \left(\begin{array}{ccc|c} 0_{a_1} & * & & * \\ * & *_{a_2+\cdots+a_\ell} & & * \\ \hline * & * & & *_{a_{\ell+1}} \end{array}\right) \mathbf{x} + (\text{affine terms}).
\end{aligned}
\tag{6}
$$

Then, the resulting polynomials have the form

$$
\begin{aligned}
A_1 \ni \bar{f}_j &= \sum_{x_t \in X_1,\ldots,X_{\ell-1}} x_t L_{j,t}(X_{\ell+1}) + Q_j(X_\ell, X_{\ell+1}), \\
&\vdots \\
A_{2\ell-2i} \ni \bar{f}_j &= \sum_{x_t \in X_1,\ldots,X_{i-1}} x_t L_{j,t}(X_{\ell+1}) + \sum_{x_t \in X_i} x_t L_{j,t}(X_{i+1},\ldots,X_{\ell+1}) \\
&\quad + Q_j(X_{i+1},\ldots,X_{\ell+1}), \\
A_{2\ell-2i+1} \ni \bar{f}_j &= \sum_{x_t \in X_1,\ldots,X_{i-1}} x_t L_{j,t}(X_{\ell+1}) + Q_j(X_i,\ldots,X_{\ell+1}), \\
&\vdots \\
A_{2\ell-2} \ni \bar{f}_j &= \sum_{x_t \in X_1} x_t L_{j,t}(X_2,\ldots,X_{\ell+1}) + Q_j(X_2,\ldots,X_{\ell+1}).
\end{aligned}
\tag{7}
$$

Here, each L_* denotes a linear form, and each Q_* denotes a quadratic polynomial, including linear and constant terms.

Step 2. Solve, for $X_{\ell+1}$, the system of linear equations obtained by setting all linear forms in (7) that depend only on $X_{\ell+1}$ equal to zero, and substitute a solution. More precisely, we solve the equations

$$
L_{j,t}(X_{\ell+1}) = 0 \quad \left(\bar{f}_j \in A_k \ (1 \le k \le 2\ell-3), x_t \in X_1,\ldots,X_{\left\lfloor \frac{2\ell-1-k}{2} \right\rfloor}\right).
$$

After the substitution, the transformed polynomials $\bar{f}_i$ take the form

$$\begin{aligned}
A_1 \ni \bar{f}_j &= Q'_j(X_\ell),\\
&\vdots\\
A_{2\ell-2i} \ni \bar{f}_j &= \sum_{x_t \in X_i} x_t L'_{j,t}(X_{i+1}, \ldots, X_\ell) + Q'_j(X_{i+1}, \ldots, X_\ell),\\
A_{2\ell-2i+1} \ni \bar{f}_j &= Q'_j(X_i, \ldots, X_\ell),\\
&\vdots\\
A_{2\ell-2} \ni \bar{f}_j &= \sum_{x_i \in X_1} x_i L'_{j,i}(X_2, \ldots, X_\ell) + Q'_j(X_2, \ldots, X_\ell).
\end{aligned} \tag{8}$$

Step 3. First solve $\bar{f}_j = 0$ with $\bar{f}_j \in A_1$ for X_ℓ and substitute the result into the remaining equations. Next solve $\bar{f}_j = 0$ with $\bar{f}_j \in A_2, A_3$ for $X_{\ell-1}$ and substitute the result. Continue this process iteratively. In general, for $2 \leq i \leq \ell - 1$, determine X_i from $\bar{f}_j = 0$ with $\bar{f}_j \in A_{2\ell-2i}, A_{2\ell-2i+1}$. In practice, when a reduced MQ system for determining X_i has multiple solutions, all of them can be enumerated, and the subsequent reduced MQ systems can be solved for each solution, as discussed in Subsection 4.4. Finally, determine X_1, if it exists, from $\bar{f}_j = 0$ with $\bar{f}_j \in A_{2\ell-2}$, together with the remaining equations

$$\bar{f}_{a_2+\cdots+a_\ell+b_1+1} = \cdots = \bar{f}_m = 0,$$

which have no specific structure. If such an X_1 exists, recover a solution to the original system by applying the inverse linear transformations from **Step 1**. Otherwise, return to **Step 2** and continue the search with a different choice of solutions for $X_{\ell+1}$.

Through these operations, the original problem can be solved by reducing it to smaller MQ subproblems. Note that Hashimoto's method corresponds to the case where $\ell = 2$ and $b_1 = 1$.

4.2 Details of Step 1

We now describe how to transform the given equations so that the coefficient matrices of the resulting equations $A_1, \ldots, A_{2\ell-2}$ have the specific forms of (6).

To perform this transformation, we construct ℓ linear transformations as follows:

$$S_1 = \begin{pmatrix} I_{a_1} & 0 \\ M_1 & I_{n-a_1} \end{pmatrix}, \cdots, S_i = \begin{pmatrix} I_{a_1+\cdots+a_{i-1}} & 0 & 0 \\ 0 & I_{a_i} & 0 \\ 0 & M_i & I_{n-(a_1+\cdots+a_i)} \end{pmatrix},$$
$$\cdots, S_\ell = \begin{pmatrix} I_{a_1+\cdots+a_{\ell-1}} & 0 & 0 \\ 0 & I_{a_\ell} & 0 \\ 0 & M_\ell & I_{a_{\ell+1}} \end{pmatrix} \in \mathbb{F}_q^{n\times n}.$$

That is, the transformed polynomials $\bar{f}_j$ with $1 \leq j \leq m$ are obtained by

$$\bar{f}_j(\mathbf{x}) = f_j(S_\ell \cdots S_1 \mathbf{x}). \tag{9}$$

In the construction of S_1, we seek a linear transformation such that for all $f_j \in A_1, \ldots, A_{2\ell-2}$, all the coefficients of $x_k x_o$ with $1 \leq k \leq o \leq a_1$ become zero, i.e.,

$$f_j(S_1\mathbf{x}) = \mathbf{x}^\top \begin{pmatrix} 0_{a_1} & * \\ * & *_{a_2+\cdots+a_{\ell+1}} \end{pmatrix} \mathbf{x} + (\text{affine terms}).$$

In the construction of S_i with $2 \leq i \leq \ell$, we seek a linear transformation satisfying the following condition:

- For all $f_j \in A_1, \ldots, A_{2\ell-2i}$, all the coefficients of $x_k x_o$ with $1 \leq k \leq a_1 + \cdots + a_i$ and $a_1 + \cdots + a_{i-1} < o \leq a_1 + \cdots + a_i$ become zero.
- For all $f_j \in A_{2\ell-2i+1}$, all the coefficients of $x_k x_o$ with $1 \leq k \leq a_1 + \cdots + a_{i-1}$ and $a_1 + \cdots + a_{i-1} < o \leq a_1 + \cdots + a_i$ become zero.
- For all $f_j \in A_{2\ell-2i+2p}, A_{2\ell-2i+2p+1}$ with $1 \leq p \leq i-2$, all the coefficients of $x_k x_o$ with $1 \leq k \leq a_1 + \cdots + a_{i-p-1}$ and $a_1 + \cdots + a_{i-1} < o \leq a_1 + \cdots + a_i$ become zero.

That is, for $f_j \in A_1, \ldots, A_{2\ell-2i}$,

$$f_j(S_i \cdots S_1\mathbf{x}) = \mathbf{x}^\top \begin{pmatrix} 0_{a_1+\cdots+a_{i-1}} & 0 & * \\ 0 & 0_{a_i} & * \\ * & * & *_{a_{i+1}+\cdots+a_{\ell+1}} \end{pmatrix} \mathbf{x} + (\text{affine terms}).$$

For $f_j \in A_{2\ell-2i+1}$,

$$f_j(S_i \cdots S_1\mathbf{x}) = \mathbf{x}^\top \begin{pmatrix} 0_{a_1+\cdots+a_{i-1}} & 0 & * \\ 0 & *_{a_i} & * \\ * & * & *_{a_{i+1}+\cdots+a_{\ell+1}} \end{pmatrix} \mathbf{x} + (\text{affine terms}).$$

For $f_j \in A_{2\ell-2i+2p}, A_{2\ell-2i+2p+1}$ with $1 \leq p \leq i-2$,

$$f_j(S_i \cdots S_1\mathbf{x}) = \mathbf{x}^\top \begin{pmatrix} 0_{a_1+\cdots+a_{i-1-p}} & 0 & 0 & * \\ 0 & H_{a_{i-p}} & * & * \\ 0 & * & *_{a_{i-p+1}+\cdots+a_i} & * \\ * & * & * & *_{a_{i+1}+\cdots+a_{\ell+1}} \end{pmatrix} \mathbf{x} + (\text{affine terms}),$$

where $H_{a_{i-p}}$ is the zero matrix $0_{a_{i-p}}$ for $A_{2\ell-2i+2p}$, and the arbitrary matrix $*_{a_{i-p}}$ for $A_{2\ell-2i+2p+1}$.

By repeating the process described above, the transformation of the equations in **Step 1** is completed.

4.3 Applicability Conditions and Complexity

We now derive the applicability conditions and time complexity of the proposed method.

Theorem 3 (Applicability conditions and time complexity of the proposed method). *Let $\ell, a_1, \ldots, a_\ell$ and $b_1, \ldots, b_{\ell-1}$ be parameters for the proposed method. Assume that $\ell, a_1, \ldots, a_\ell$ are positive integers satisfying $a_1 + \cdots + a_\ell < m$, and for each $1 \leq i \leq \ell - 1$, b_i is an integer satisfying $0 \leq b_i < a_i$. Furthermore, define $a_{\ell+1} := n - (a_1 + \cdots + a_\ell)$ and $s := m - (a_1 + \cdots + a_\ell)$. Under these settings, the applicability conditions for the proposed method are given by the following:*

$$\begin{cases} n - a_1 \geq a_1(a_2 + \cdots + a_\ell + b_1), \\ \quad \vdots \\ n - (a_1 + \cdots + a_i) \geq a_1(a_2 + \cdots + a_\ell) + \cdots + a_{i-1}(a_i + \cdots + a_\ell) \\ \qquad\qquad + a_i(a_{i+1} + \cdots + a_\ell + b_i), \\ \quad \vdots \\ n - (a_1 + \cdots + a_{\ell-1}) \geq a_1(a_2 + \cdots + a_\ell) + \cdots + a_{\ell-2}(a_{\ell-1} + a_\ell) \\ \qquad\qquad + a_{\ell-1}(a_\ell + b_{\ell-1}), \\ n - (a_1 + \cdots + a_\ell) \geq a_1(a_2 + \cdots + a_\ell) + \cdots + a_{\ell-1}a_\ell + s. \end{cases} \tag{10}$$

Under Assumption 1, the expected time complexity of the proposed method is estimated by

$$\sum_{i=1}^{\ell-1} a_i \mathrm{XL}\left(q, b_i + \sum_{w=i+1}^{\ell} a_w, b_i + \sum_{w=i+1}^{\ell} a_w\right) + q^s \Bigg(\mathrm{XL}(q, a_\ell, a_\ell) + \sum_{i=2}^{\ell-1} (a_i b_i^2 + \mathrm{XL}(q, a_i - b_i, a_i - b_i)) + a_1 b_1^2 + \mathrm{XL}(q, a_1 - b_1, s + a_1 - b_1) \Bigg). \tag{11}$$

Proof. In **Step 1** of the proposed method, to determine $M_1 \in \mathbb{F}_q^{(n-a_1)\times a_1}$, we first compute the first column of M_1 by setting the coefficients of x_1^2 of $f_j \in A_1, \ldots, A_{2\ell-2}$ to zero. Next, we compute the second column by setting the coefficients of x_1x_2 and x_2^2 to zero. By iteratively repeating this process, the a_1-th column is computed by setting the coefficients of $x_1x_{a_1}, x_2x_{a_1}, \ldots, x_{a_1}^2$ to zero. When computing the j-th column of M_1 with $1 \leq j \leq a_1$, it is necessary to solve a system of quadratic equations consisting of $(n - a_1)$ variables, $(a_2 + \cdots + a_\ell + b_1)$ quadratic forms, and $(j-1)(a_2 + \cdots + a_\ell + b_1)$ linear forms (at most $(a_1 - 1)(a_2 + \cdots + a_\ell + b_1)$ linear forms). Therefore, under Assumption 1, we use the following applicability condition:

$$n - a_1 \geq a_1(a_2 + \cdots + a_\ell + b_1).$$

The time complexity for deriving the entire matrix M_1 is given by:

$$a_1\mathrm{XL}(q, a_2 + \cdots + a_\ell + b_1, a_2 + \cdots + a_\ell + b_1).$$

Similarly, for each $2 \leq i \leq \ell - 1$, determining $M_i \in \mathbb{F}_q^{(n-(a_1+\cdots+a_i))\times a_i}$ requires solving a system of quadratic equations a_i times. Each system consists of $(n - (a_1 + \cdots + a_i))$ variables, $(a_{i+1} + \cdots + a_\ell + b_i)$ quadratic forms, and at most $a_1(a_2 + \cdots + a_\ell) + a_2(a_3 + \cdots + a_\ell) + \cdots + a_{i-1}(a_i + \cdots + a_\ell) + (a_i - 1)(a_{i+1} + \cdots + a_\ell + b_i)$ linear forms. Therefore, according to Assumption 1, the applicability condition is:

$$\begin{aligned} n - (a_1 + \cdots + a_i) \geq a_1(a_2 + \cdots + a_\ell) + \cdots \\ + a_{i-1}(a_i + \cdots + a_\ell) + a_i(a_{i+1} + \cdots + a_\ell + b_i). \end{aligned}$$

The time complexity is given by:

$$a_i\mathrm{XL}(q, a_{i+1} + \cdots + a_\ell + b_i, a_{i+1} + \cdots + a_\ell + b_i).$$

To determine $M_\ell \in \mathbb{F}_q^{a_{\ell+1}\times a_\ell}$, it is necessary to solve a system of linear equations a_ℓ times. Each system consists of $(n - (a_1 + \cdots + a_\ell))$ variables, no quadratic forms, and $a_1(a_2 + \cdots + a_\ell) + a_2(a_3 + \cdots + a_\ell) + \cdots + a_{\ell-1}a_\ell$ linear forms. Thus, the applicability condition is expressed as:

$$n - (a_1 + \cdots + a_\ell) \geq a_1(a_2 + \cdots + a_\ell) + \cdots + a_{\ell-1}a_\ell.$$

Since this step involves solving only linear equations, the time complexity remains within polynomial time and is not dominant in the total complexity.

In **Step 2**, to determine $X_{\ell+1}$, it is necessary to solve a system of linear equations consisting of $n - (a_1 + \cdots + a_\ell)$ variables and $a_1(a_2 + \cdots + a_\ell) + a_2(a_3 + \cdots + a_\ell) + \cdots + a_{\ell-1}a_\ell$ linear forms. To provide at least q^s distinct candidates for the repeated **Step 3** search, we require the affine solution space obtained in **Step 2** to have dimension at least s. Consequently, the practical applicability condition is expressed as:

$$n - (a_1 + \cdots + a_\ell) \geq a_1(a_2 + \cdots + a_\ell) + \cdots + a_{\ell-1}a_\ell + s.$$

Since this step solves only linear equations, the time complexity remains within polynomial time and is not dominant in the total complexity.

In **Step 3**, to determine X_ℓ, it is necessary to solve a system of quadratic equations consisting of a_ℓ variables and a_ℓ quadratic forms. Therefore, the time complexity is given by $\mathrm{XL}(q, a_\ell, a_\ell)$. Furthermore, to determine X_i $(2 \leq i \leq \ell-1)$ in **Step 3**, one must solve a system of quadratic equations consisting of a_i variables, $(a_i - b_i)$ quadratic forms, and b_i linear forms. Thus, the cost of this step is given by

$$O(a_i b_i^2) + \mathrm{XL}(q, a_i - b_i, a_i - b_i),$$

where the first term accounts for solving the linear equations and the second for solving the resulting quadratic system. The linear-equation cost $O(a_i b_i^2)$ is

included because it can dominate this step when $a_i - b_i$ is small and the XL cost $\mathrm{XL}(q, a_i - b_i, a_i - b_i)$ becomes relatively small. Finally, to determine X_1 in **Step 3**, it is necessary to solve a system of quadratic equations consisting of a_1 variables, $(s + a_1 - b_1)$ quadratic forms, and b_1 linear forms. Then, the complexity is given by

$$O(a_1 b_1^2) + \mathrm{XL}(q, a_1 - b_1, s + a_1 - b_1).$$

By the heuristic in Assumption 1, the final system for X_1 has expected number of solutions q^{-s}. From this estimate, we use q^{-s} as the heuristic scale of the one-trial success probability, and therefore estimate that **Step 2** and **Step 3** require $O(q^s)$ trials. The validity of the resulting q^s repetition estimate is examined experimentally in Section 4.4. □

We finally describe the quantum version of the above evaluation. The main difference from the classical setting is that exhaustive-search factors are replaced by their square roots using Grover search [17]. In general, we do not suppose a quantum speedup for the XL algorithm itself. We only apply Grover search to the guessing part of the hybrid method. Thus, whenever a classical MQ estimate contains a factor q^k, we replace it by $q^{k/2}$. We thus define the resulting quantum complexity of solving $\mathrm{MQ}(q, n, m)$ with $n \leq m$ by

$$\mathrm{XL}_{\mathrm{Q}}(q, n, m) = \min_{0 \leq k \leq n} \left\{ O\left(q^{k/2} \cdot 3 \cdot \binom{n-k+2}{2} \cdot \binom{d_{\mathrm{reg}} + n - k}{d_{\mathrm{reg}}}^2 \right) \right\}.$$

Similarly, in the proposed method, the repeated search over the q^s candidates produced after **Step 2** is replaced by a quantum search over the same candidate space. Hence, in the quantum evaluation, the classical factor q^s is replaced by $q^{s/2}$. As a result, the time complexity of the proposed method on quantum computers is given by the following:

$$\sum_{i=1}^{\ell-1} a_i \mathrm{XL}_{\mathrm{Q}}\left(q, b_i + \sum_{w=i+1}^{\ell} a_w, b_i + \sum_{w=i+1}^{\ell} a_w\right) + q^{s/2}\Bigg(\mathrm{XL}(q, a_\ell, a_\ell) + \sum_{i=2}^{\ell-1}(a_i b_i^2 + \mathrm{XL}(q, a_i - b_i, a_i - b_i)) + a_1 b_1^2 + \mathrm{XL}(q, a_1 - b_1, s + a_1 - b_1)\Bigg). \quad (12)$$

In the quantum estimate, Grover search is applied at only one search level at a time. The MQ systems solved in **Step 1** are preprocessing computations performed before the outer search over the **Step 2** candidate space, and are therefore evaluated using XL_{Q}. By contrast, the q^s candidates produced after **Step 2** are searched by an outer Grover procedure. Thus, once one candidate for $X_{\ell+1}$ is fixed inside this outer search, all reduced MQ systems solved in **Step 3** are evaluated using the classical XL complexity, rather than XL_{Q}. This convention avoids assuming a nested application of Grover search and yields the $q^{s/2}$ factor in (12).

4.4 Validity of Complexity Analysis under Assumption 1

The complexity analysis in Theorem 3 relies on Assumption 1 on the solvability of random MQ systems. We therefore discuss how the existence and number of solutions affect the practical complexity of **Step 1** and **Step 3**, and finally support the analysis with experimental results.

We first discuss the practical complexity of **Step 1**. For each M_i with $1 \leq i \leq \ell - 1$, the MQ systems for determining its columns are solved sequentially, and only the MQ system arising when determining the last column can become square when the corresponding applicability condition is satisfied with equality. If this square MQ system has no solution, one can backtrack within the construction of M_i and choose an alternative solution obtained when determining a preceding column. By Assumption 1, the probability that such a square MQ system has a solution is of constant order. Hence, the expected number of trials is of constant order. Moreover, when the corresponding applicability condition is satisfied with strict inequality, the MQ system for the last column remains underdetermined, and the expected overhead becomes even smaller. We therefore omit this constant-order factor from the complexity estimate of **Step 1**.

We next discuss the practical complexity of **Step 3**. In this step, square MQ systems are solved sequentially to determine $X_\ell, \ldots, X_2$, followed by the final overdetermined MQ system for determining X_1. In practice, a square MQ system may have multiple solutions or no solution. However, the expected number of solutions to a random square MQ system is one [16, 19]. Therefore, when a square MQ system for determining X_i has multiple solutions, we consider solving the subsequent MQ system for X_{i-1} for each of these solutions. With this procedure, for each of the $O(q^s)$ repetitions of **Step 3**, the expected number of MQ systems solved at each stage for determining X_i is one. Consequently, the expected complexity of **Step 3** is given by the sum of the solving costs for the individual MQ systems, as in Theorem 3. This shows that the complexity estimate in Theorem 3 remains valid even when the actual number of solutions to each square MQ system is taken into account.

Finally, we experimentally examine whether a solution can be found with the q^s repetitions of **Step 3** when all solutions of the reduced square MQ systems encountered in **Step 3** are enumerated, as described above. Specifically, we considered the 40 parameter settings listed in Tables 1 and 2. For $q = 16$, we set $n = 64$ and $m \in \{8, 9\}$, while for $q = 7$, we set $n = 64$ and $m \in \{8, 9, 10\}$. In both cases, we considered $\ell \in \{2, \ldots, 7\}$ and $\sum_{i=1}^{\ell} a_i = 7$. Since $s = m - \sum_{i=1}^{\ell} a_i$, these settings correspond to $s \in \{1, 2\}$ for $q = 16$ and $s \in \{1, 2, 3\}$ for $q = 7$. Since the experimental cost grows with q^s, we restrict the experiments to relatively small values of s. For each of the 40 parameter settings, we generated 128 valid **Step 2** transcripts, selected an s-dimensional affine subspace of the solution space, and exhaustively solved all q^s corresponding **Step 3** candidates. Every recovered solution was verified against the original MQ system.

In Tables 1 and 2, the success probability denotes the fraction of the 128 **Step 2** transcripts for which at least one solution was found among the q^s **Step 3** candidates. Across the parameter settings, the probability that at least

Table 1. Experimental results for the q^s repetitions in **Step 3** over $\mathbb{F}_{16}$.

q	n	m	s	ℓ	$(a_1, \ldots, a_\ell)$	$(b_1, \ldots, b_{\ell-1})$	Success probability
16	64	8	1	2	$(4, 3)$	(1)	0.625
16	64	8	1	2	$(4, 3)$	(3)	0.680
16	64	8	1	3	$(3, 2, 2)$	$(1, 0)$	0.617
16	64	8	1	3	$(2, 3, 2)$	$(1, 2)$	0.586
16	64	8	1	4	$(2, 2, 2, 1)$	$(0, 0, 0)$	0.609
16	64	8	1	4	$(3, 1, 2, 1)$	$(2, 0, 1)$	0.578
16	64	8	1	5	$(2, 2, 1, 1, 1)$	$(0, 0, 0, 0)$	0.602
16	64	8	1	5	$(3, 1, 1, 1, 1)$	$(2, 0, 0, 0)$	0.578
16	64	8	1	6	$(2, 1, 1, 1, 1, 1)$	$(0, 0, 0, 0, 0)$	0.563
16	64	8	1	6	$(1, 1, 1, 1, 2, 1)$	$(0, 0, 0, 0, 1)$	0.570
16	64	8	1	7	$(1, 1, 1, 1, 1, 1, 1)$	$(0, 0, 0, 0, 0, 0)$	0.578
16	64	9	2	2	$(4, 3)$	(1)	0.664
16	64	9	2	2	$(4, 3)$	(3)	0.648
16	64	9	2	3	$(3, 2, 2)$	$(1, 0)$	0.570
16	64	9	2	3	$(2, 3, 2)$	$(1, 2)$	0.609
16	64	9	2	4	$(2, 2, 2, 1)$	$(0, 0, 0)$	0.641
16	64	9	2	4	$(3, 1, 2, 1)$	$(2, 0, 1)$	0.594
16	64	9	2	5	$(2, 2, 1, 1, 1)$	$(0, 0, 0, 0)$	0.656
16	64	9	2	5	$(3, 1, 1, 1, 1)$	$(2, 0, 0, 0)$	0.664
16	64	9	2	6	$(2, 1, 1, 1, 1, 1)$	$(0, 0, 0, 0, 0)$	0.594
16	64	9	2	6	$(1, 1, 1, 1, 2, 1)$	$(0, 0, 0, 0, 1)$	0.633
16	64	9	2	7	$(1, 1, 1, 1, 1, 1, 1)$	$(0, 0, 0, 0, 0, 0)$	0.609

Table 2. Experimental results for the q^s repetitions in **Step 3** over $\mathbb{F}_7$.

q	n	m	s	ℓ	$(a_1, \ldots, a_\ell)$	$(b_1, \ldots, b_{\ell-1})$	Success probability
7	64	8	1	2	$(4, 3)$	(3)	0.648
7	64	8	1	3	$(2, 3, 2)$	$(1, 2)$	0.547
7	64	8	1	4	$(2, 2, 2, 1)$	$(1, 1, 1)$	0.578
7	64	8	1	5	$(1, 1, 2, 2, 1)$	$(0, 0, 1, 1)$	0.484
7	64	8	1	6	$(1, 1, 1, 1, 2, 1)$	$(0, 0, 0, 0, 1)$	0.484
7	64	8	1	7	$(1, 1, 1, 1, 1, 1, 1)$	$(0, 0, 0, 0, 0, 0)$	0.555
7	64	9	2	2	$(4, 3)$	(3)	0.641
7	64	9	2	3	$(2, 3, 2)$	$(1, 2)$	0.609
7	64	9	2	4	$(2, 2, 2, 1)$	$(1, 1, 1)$	0.617
7	64	9	2	5	$(1, 1, 2, 2, 1)$	$(0, 0, 1, 1)$	0.625
7	64	9	2	6	$(1, 1, 1, 1, 2, 1)$	$(0, 0, 0, 0, 1)$	0.547
7	64	9	2	7	$(1, 1, 1, 1, 1, 1, 1)$	$(0, 0, 0, 0, 0, 0)$	0.547
7	64	10	3	2	$(4, 3)$	(3)	0.641
7	64	10	3	3	$(2, 3, 2)$	$(1, 2)$	0.563
7	64	10	3	4	$(2, 2, 2, 1)$	$(1, 1, 1)$	0.594
7	64	10	3	5	$(1, 1, 2, 2, 1)$	$(0, 0, 1, 1)$	0.617
7	64	10	3	6	$(1, 1, 1, 1, 2, 1)$	$(0, 0, 0, 0, 1)$	0.602
7	64	10	3	7	$(1, 1, 1, 1, 1, 1, 1)$	$(0, 0, 0, 0, 0, 0)$	0.641

one solution was found among the q^s candidates ranged from 0.563 to 0.680 over $\mathbb{F}_{16}$ and from 0.484 to 0.648 over $\mathbb{F}_7$. For reference, if the total number of solutions obtained from the q^s candidates follows a Poisson distribution with mean one, the probability that at least one solution exists is

$$1 - e^{-1} \approx 0.6321.$$

Many of the experimental results are broadly consistent with this value. Although lower success probabilities are observed for some parameter settings with $q = 7$ and $s = 1$, this tendency becomes less pronounced as s increases, suggesting that the decrease for small s does not appear to affect the constant-probability behavior underlying our complexity estimate. Overall, these results suggest that examining the q^s candidates in **Step 3** is sufficient to find a solution with constant probability, supporting the $O(q^s)$ repetition factor used in Theorem 3.

5 Parameter Search Using Reformulated Conditions

The proposed algorithm in Section 4 uses the parameters $(\ell, a_1, \ldots, a_\ell, b_1, \ldots, b_{\ell-1})$. For the complexity estimation, we need to find the parameters that minimize the complexity of the proposed algorithm. However, since there are many parameters, a naive exhaustive search over all candidates is inefficient. To solve this problem, we present an efficient way to find parameters $(\ell,\ a_1, \ldots, a_\ell,\ b_1, \ldots, b_{\ell-1})$ that minimize the estimated complexity.

This section first introduces the auxiliary quantities and rewrites the original conditions for the parameters in a simpler form. Then, by using this reformulation, we derive the admissible range for parameters $(\ell,\ a_1, \ldots, a_\ell,\ b_1, \ldots, b_{\ell-1})$. Finally, we describe the strategy for finding the parameters that minimize the complexity using the obtained range for the parameters. Note that, although this section focuses on the classical complexity estimate, the same strategy can be applied to the quantum case.

5.1 Reformulated Conditions

We use the following notation U_i and V_i with $1 \le i \le \ell$ and W_i with $1 \le i \le \ell - 1$ defined as follows:

$$U_i := \sum_{u=1}^{i} a_u, \quad V_i := \sum_{1 \le u < v \le i} a_u a_v, \quad W_i := \sum_{u=1}^{i} \left(a_u \sum_{v=u+1}^{\ell} a_v \right).$$

Then, from the definition, we have $s = m - U_\ell$ and $W_{\ell-1} = V_\ell$.

For these notations, from the condition $a_1 + \cdots + a_\ell < m$, we obtain

$$U_\ell \le m - 1. \tag{13}$$

Furthermore, by directly applying the above notations, the original condition (10) for the proposed algorithm can be reformulated as

$$n - U_i \geq W_i + a_i b_i \quad (1 \leq i \leq \ell - 1), \tag{14}$$

$$n - U_\ell \geq W_{\ell-1} + s. \tag{15}$$

For the second condition (15), by substituting $s = m - U_\ell$, we have

$$W_{\ell-1} \leq n - m. \tag{16}$$

This observation is the basis of the parameter search algorithm described in the following.

5.2 Range of Parameters

We derive explicit ranges for the parameters $(\ell, a_1, \ldots, a_\ell, b_1, \ldots, b_{\ell-1})$ based on the reformulated applicability conditions obtained in Subsection 5.1. These ranges are used in the parameter search algorithm described in Subsection 5.3 below.

Range of ℓ. We first derive an upper bound for the number of partitions ℓ. We construct this range only using the given parameters n and m, since we first fix ℓ in our parameter search in Subsection 5.3 below.

Since $a_i \geq 1$ with $1 \leq i \leq \ell$, we have

$$\ell \leq \sum_{i=1}^{\ell} a_i = U_\ell.$$

Together with (13), this gives

$$\ell \leq m - 1. \tag{17}$$

From (16), since each a_i is positive, we have

$$n - m \geq W_{\ell-1} = \sum_{1 \leq u < v \leq \ell} a_u a_v \geq \binom{\ell}{2}.$$

Solving this inequality for ℓ, we obtain

$$\ell \leq \left\lfloor \frac{1 + \sqrt{1 + 8(n-m)}}{2} \right\rfloor. \tag{18}$$

Combining the two upper bounds (17), (18), the admissible range of ℓ is given by

$$2 \leq \ell \leq \min \left\{ m - 1, \left\lfloor \frac{1 + \sqrt{1 + 8(n-m)}}{2} \right\rfloor \right\}. \tag{19}$$

We denote the upper bound on the right-hand side by u_ℓ. Here, the lower bound $\ell \geq 2$ is imposed because the case $\ell = 1$ does not correspond to the multi-partition setting considered in this paper.

Range of a_i. We next derive an upper bound for each a_i with $1 \le i \le \ell$. We assume here that $\ell, a_1, \ldots, a_{i-1}$ have already been fixed, i.e., the range is constructed using these parameters.

First, since $a_{i+1}, \ldots, a_\ell \ge 1$, we have

$$U_\ell \ge U_{i-1} + a_i + (\ell - i).$$

Together with the condition (13), this implies

$$a_i \le m - 1 - U_{i-1} - \ell + i. \tag{20}$$

We next derive another upper bound from the condition (16). We decompose $W_{\ell-1}$ according to the position of a_i as follows:

$$\begin{aligned} W_{\ell-1} = V_\ell = V_{i-1} &+ \sum_{1 \le u \le i-1} a_u a_i + \sum_{\substack{1 \le u \le i-1 \\ i+1 \le v \le \ell}} a_u a_v \\ &+ \sum_{i+1 \le v \le \ell} a_i a_v + \sum_{i+1 \le u < v \le \ell} a_u a_v, \\ &\ge V_{i-1} + U_{i-1} a_i + U_{i-1}(\ell - i) + a_i(\ell - i) + \binom{\ell - i}{2}, \end{aligned}$$

since $a_{i+1}, \ldots, a_\ell \ge 1$. Then, using (16), we obtain

$$V_{i-1} + U_{i-1} a_i + U_{i-1}(\ell - i) + a_i(\ell - i) + \binom{\ell - i}{2} \le n - m.$$

Rearranging this inequality with respect to a_i, we get

$$a_i \le \left\lfloor \frac{n - m - V_{i-1} - U_{i-1}(\ell - i) - \binom{\ell - i}{2}}{U_{i-1} + \ell - i} \right\rfloor. \tag{21}$$

For $1 \le i \le \ell - 1$, we also derive an upper bound from the condition (14). Since $b_i \ge 0$, we can derive the necessary condition $n - U_i - W_i \ge 0$. From the definition of W_i, we have

$$\begin{aligned} W_i &= V_{i-1} + \sum_{u=1}^{i-1} a_u \left(\sum_{v=i}^{\ell} a_v \right) + a_i \left(\sum_{v=i+1}^{\ell} a_v \right) \\ &\ge V_{i-1} + U_{i-1}(a_i + \ell - i) + a_i(\ell - i), \end{aligned}$$

since $a_{i+1}, \ldots, a_\ell \ge 1$. Substituting this bound into $n - U_i - W_i \ge 0$ and using $U_i = U_{i-1} + a_i$, we obtain

$$n - (U_{i-1} + a_i) - (V_{i-1} + U_{i-1}(a_i + \ell - i) + a_i(\ell - i)) \ge 0.$$

Rearranging the inequality with respect to a_i, we get

$$a_i \le \left\lfloor \frac{n - V_{i-1} - U_{i-1}(\ell - i + 1)}{U_{i-1} + \ell - i + 1} \right\rfloor. \tag{22}$$

This bound should be understood as an additional bound only for $1 \le i \le \ell - 1$, because b_ℓ is not defined and (14) is not a condition for $i = \ell$. When $i = \ell$, however, the numerical expression in (22) does not impose any additional restriction beyond the other conditions. Indeed, the bounds (20) and (21) imply

$$U_{\ell-1} + a_\ell \le m - 1 \quad \text{and} \quad U_{\ell-1}a_\ell + V_{\ell-1} \le n - m.$$

Hence

$$(U_{\ell-1} + 1)a_\ell \le n - m - V_{\ell-1} + m - U_{\ell-1} - 1 = n - V_{\ell-1} - U_{\ell-1} - 1,$$

which implies (22) for $i = \ell$. Thus, including the third bound in the uniform range below does not exclude any feasible value of a_ℓ.

For $1 \le i \le \ell - 1$, combining the three upper bounds (20), (21), (22) gives the admissible range of a_i. For $i = \ell$, the range is obtained from (20) and (21); as explained above, the third bound is redundant in this case. Therefore, the following uniform range can be used for all $1 \le i \le \ell$:

$$1 \le a_i \le \min\Bigg\{ m - U_{i-1} - \ell + i - 1, \left\lfloor \frac{n - m - V_{i-1} - U_{i-1}(\ell - i) - \binom{\ell-i}{2}}{U_{i-1} + \ell - i} \right\rfloor, \left\lfloor \frac{n - V_{i-1} - U_{i-1}(\ell - i + 1)}{U_{i-1} + \ell - i + 1} \right\rfloor \Bigg\}. \tag{23}$$

We denote the upper bound on the right-hand side by u_{a_i}. If the upper bound is smaller than 1, then there is no feasible choice of a_i for the current prefix $a_1, \ldots, a_{i-1}$, and the corresponding branch of the search can be discarded.

Range of b_i. We derive the admissible range of b_i with $1 \le i \le \ell - 1$, assuming that $a_1, \ldots, a_\ell$ have been fixed.

From the reformulated applicability condition (14), for each $1 \le i \le \ell - 1$, we have

$$b_i \le \frac{n - U_i - W_i}{a_i},$$

since $a_i > 0$. Thus, using the original condition $0 \le b_i \le a_i - 1$, the admissible range of b_i is given by

$$0 \le b_i \le \min\left\{ a_i - 1, \left\lfloor \frac{n - U_i - W_i}{a_i} \right\rfloor \right\}. \tag{24}$$

We denote the upper bound on the right-hand side by u_{b_i}.

For each i, the upper bound u_{b_i} does not depend on the other parameters $b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_{\ell-1}$. Hence, the range of each b_i can be computed independently. This property will be used in the parameter search algorithm.

5.3 Parameter Search Strategy

In this section, we present an algorithm for searching for parameters $(\ell, a_1, \ldots, a_\ell, b_1, \ldots, b_{\ell-1})$ that minimize the estimated complexity of the proposed method. The algorithm is based on the range constraints derived in Subsection 5.2.

In our parameter search algorithm, we first fix ℓ in the range (19) which depends only on n and m. Then, for a fixed ℓ, the parameters $a_1, \ldots, a_\ell$ can be fixed sequentially using the range (23), since the range (23) depends only on $\ell, a_1, \ldots, a_{i-1}$. Once $a_1, \ldots, a_\ell$ are fixed, the admissible range (24) of each b_i is determined. We can then choose each b_i so as to minimize the part of the total complexity that depends on b_i. This is because, in the time complexity estimation (11), each b_i is only involved in the part of

$$a_i \mathrm{XL}\left(q, b_i + \sum_{w=i+1}^{\ell} a_w, b_i + \sum_{w=i+1}^{\ell} a_w\right) + q^s \left(a_i b_i^2 + \mathrm{XL}(q, a_i - b_i, a_i - b_i)\right),$$

if $i \geq 2$, or

$$a_1 \mathrm{XL}\left(q, b_1 + \sum_{w=2}^{\ell} a_w, b_1 + \sum_{w=2}^{\ell} a_w\right) + q^s \left(a_1 b_1^2 + \mathrm{XL}(q, a_1 - b_1, a_1 - b_1 + s)\right),$$

if $i = 1$. This enables us to optimize each b_i separately with respect to its contribution to the total complexity and avoid the exhaustive search over all tuples $(b_1, \ldots, b_{\ell-1})$ satisfying the condition (24).

Through the proposed strategy, we can avoid generating many infeasible or unnecessary candidates for parameters. The details of the parameter search algorithm are given in Appendix A.

6 Computational Complexity Evaluation

In this section, we evaluate the complexity of the proposed method in Section 4 for the parameter sets of multivariate signature schemes with highly underdetermined public key maps, such as MAYO [5], QR-UOV [13], and SNOVA [27]. We compare the resulting estimates with those of the Just Guess algorithm [21], Hashimoto's method [18] and the best attack in the corresponding specifications [5, 13, 27].

In the following tables, the complexity is expressed as the base-two logarithm of the estimated number of gates. Following the cost model used in the specifications, one operation over $\mathbb{F}_q$ is converted into $2(\log_2 q)^2 + \log_2 q$ gates. Thus, the operation counts estimated by (11) and (12) are multiplied by this factor for the classical and quantum cases, respectively, before taking the base-two logarithm. All displayed values are rounded to the nearest integer. The column "ours" reports the complexity obtained with the parameters selected by our parameter search in Section 5. The entries for ℓ, $a = (a_1, \ldots, a_\ell)$, $b = (b_1, \ldots, b_{\ell-1})$, and s give the partition parameters selected by this search. These parameters determine how the original underdetermined MQ instance is decomposed into smaller

subproblems. Since the optimal parameters for the decomposition depend on the complexity formula, the selected values may differ between the classical and quantum cases.

The columns headed by [21] and [18] show the corresponding estimates for the Just Guess algorithm and Hashimoto's method, respectively. The final columns also show the complexity of the best attacks reported in the corresponding specifications [5, 13, 27]. In the columns for the best attacks, a dash indicates that the corresponding quantum attack complexity is not reported in the specifications. The specification columns may include attacks outside the partition-based direct attacks considered in this paper.

Table 3 shows the complexity estimates for the MAYO Round 2 parameter sets. For a MAYO parameter set (n, m, o, k, q), we evaluate the complexity of solving the system of m equations in kn variables over $\mathbb{F}_q$. In the classical case, except for MAYO2, for which the specification reports a smaller claw-finding attack estimate, Hashimoto's method gives the best direct-attack estimate among those considered in the specification. The proposed method gives estimates that are equal to or smaller than those of Hashimoto's method. In particular, for MAYO1, Hashimoto's method gives an estimate of 2^{156} in our cost model, whereas the proposed method reduces this estimate to 2^{145}. On the other hand, for MAYO2 and MAYO5, in the classical case, the optimized parameters are $\ell = 2$ and $b_1 = 1$, which correspond to Hashimoto's method. These classical results suggest the optimality of Hashimoto's method for these parameter sets within our generalized framework and cost model. In the quantum case, the proposed method gives estimates smaller than both Just Guess and Hashimoto's method. In particular, for MAYO5, our estimate improves on the Just Guess and Hashimoto estimates by 11 bits and 26 bits, respectively.

Table 4 shows the complexity estimates for the main parameter sets of QR-UOV with $q = 127$ in the Round 2 specification [13]. For a QR-UOV parameter set (q, v, m, l), we evaluate the complexity of solving the system of m equations in $v+m$ variables over $\mathbb{F}_q$. In both the classical and quantum cases, the proposed method gives estimates that are almost the same as those of Hashimoto's method. In particular, for Level III, the parameters corresponding to Hashimoto's method are selected as optimal. For all QR-UOV parameter sets, the proposed method gives smaller estimates than the Just Guess algorithm.

Table 5 shows the complexity estimates for the SNOVA Round 2 parameter sets. For a SNOVA parameter set (v, o, q, l), we evaluate the complexity of solving the system of ol^2 equations in $(v + o)l^2$ variables over $\mathbb{F}_q$. Following the evaluation in [21], we consider the three parameter sets, $(v, o, q, l) = (24, 5, 16, 4)$ for Level I, $(v, o, q, l) = (24, 5, 16, 5)$ for Level III, and $(v, o, q, l) = (60, 10, 16, 4)$ for Level V. In the classical case, the proposed method gives the same estimates as Hashimoto's method for all the parameter sets. In the quantum case, the proposed method gives estimates that are smaller than those of both Just Guess and Hashimoto's method.

Table 3. Complexity comparison for the MAYO Round 2 parameter sets. The selected partition is displayed in two consecutive rows for each model, labelled a and b. All complexity values are base-two logarithms of estimated gate counts. The columns ours, JG, Hash., and Spec. denote the proposed method, the Just Guess algorithm, Hashimoto's method, and the best attack reported in the MAYO specification, respectively. For each model, the smallest numerical estimate is highlighted in bold; ties are bolded together. Here, (q, n, m) denotes the resulting MQ instance. For MAYO2, the specification value corresponds to the claw-finding attack, rather than a direct MQ-solving attack.

MAYO1 $(q = 16, n = 860, m = 78)$								
Model	ℓ	Part.	Selected partition	s	ours	JG [21]	Hash. [18]	Spec. [5]
classical	5	a	$(20, 12, 8, 6, 1)$	31	**145**	158	156	156
		b	$(15, 9, 5, 4)$					
quantum	6	a	$(6, 16, 10, 6, 6, 1)$	33	**88**	90	104	–
		b	$(0, 14, 9, 5, 5)$					
MAYO2[*] $(q = 16, n = 324, m = 64)$								
Model	ℓ	Part.	Selected partition	s	ours	JG [21]	Hash. [18]	Spec. [5]
classical	2	a	$(45, 5)$	14	155	172	155	**141**
		b	(1)					
quantum	5	a	$(10, 8, 7, 1, 1)$	37	**90**	96	102	–
		b	$(9, 7, 6, 0)$					
MAYO3 $(q = 16, n = 1298, m = 108)$								
Model	ℓ	Part.	Selected partition	s	ours	JG [21]	Hash. [18]	Spec. [5]
classical	6	a	$(23, 14, 9, 8, 2, 1)$	51	**223**	240	224	224
		b	$(21, 13, 8, 7, 1)$					
quantum	6	a	$(23, 14, 9, 8, 2, 1)$	51	**121**	132	146	–
		b	$(21, 13, 8, 7, 1)$					
MAYO5 $(q = 16, n = 1848, m = 142)$								
Model	ℓ	Part.	Selected partition	s	ours	JG [21]	Hash. [18]	Spec. [5]
classical	2	a	$(92, 18)$	32	**296**	333	**296**	**296**
		b	(1)					
quantum	6	a	$(26, 17, 12, 10, 2, 1)$	74	**168**	179	194	–
		b	$(24, 16, 10, 8, 1)$					

[*] The Just Guess estimates for MAYO2 were computed using the GitHub code accompanying the Just Guess paper [21].

Table 4. Complexity comparison for the QR-UOV Round 2 parameter sets.* The selected partition is displayed in two consecutive rows for each model, labelled a and b. All complexity values are base-two logarithms of estimated gate counts. The columns ours, JG, Hash., and Spec. denote the proposed method, the Just Guess algorithm, Hashimoto's method, and the best attack reported in the QR-UOV specification, respectively. For each model, the smallest numerical estimate is highlighted in bold; ties are bolded together. Here, (q, n, m) denotes the resulting MQ instance.

QR-UOV Level I ($q = 127, n = 210, m = 54$)								
Model	ℓ	Part.	Selected partition	s	ours	JG [21]	Hash. [18]	Spec. [13]
classical	2	a	$(49, 3)$	2	160	258	160	**150**
		b	(0)					
quantum	2	a	$(35, 3)$	16	131	139	131	**128**
		b	(2)					
QR-UOV Level III ($q = 127, n = 306, m = 78$)								
Model	ℓ	Part.	Selected partition	s	ours	JG [21]	Hash. [18]	Spec. [13]
classical	2	a	$(73, 2)$	3	222	399	222	**211**
		b	(1)					
quantum	2	a	$(51, 4)$	23	181	210	181	**180**
		b	(1)					
QR-UOV Level V ($q = 127, n = 411, m = 105$)								
Model	ℓ	Part.	Selected partition	s	ours	JG [21]	Hash. [18]	Spec. [13]
classical	2	a	$(99, 3)$	3	288	561	288	**277**
		b	(0)					
quantum	2	a	$(79, 3)$	23	**237**	292	**237**	**237**
		b	(1)					

* The Just Guess estimates for the QR-UOV parameter sets with $q = 127$ were computed using the GitHub code accompanying the Just Guess paper [21].

7 Conclusion

In this paper, we studied algorithms for solving underdetermined MQ systems and their application to the security evaluation of multivariate signature schemes. We proposed a generalized partition-based algorithm that extends Hashimoto's method by partitioning the variables into more groups. Whereas Hashimoto's method reduces the original problem to two smaller MQ subproblems, our framework allows the problem to be decomposed into a larger number of smaller MQ subproblems. We derived the applicability conditions and complexity estimates of the proposed algorithm, and developed an efficient parameter search strategy to find optimized partition parameters without a naive exhaustive search.

The complexity evaluation of the proposed method demonstrates the benefit of the generalized partitioning for several multivariate signature schemes. In the classical setting, the proposed method improves on Hashimoto's method for some parameter sets; in particular, for MAYO1, it reduces the estimate from 2^{156} to 2^{145}. In the quantum setting, the proposed algorithm gives smaller esti-

Table 5. Complexity comparison for the SNOVA Round 2 parameter sets. The selected partition is displayed in two consecutive rows for each model, labelled a and b. All complexity values are base-two logarithms of estimated gate counts. The columns ours, JG, Hash., and Spec. denote the proposed method, the Just Guess algorithm, Hashimoto's method, and the best attack reported in the SNOVA specification, respectively. For each model, the smallest numerical estimate is highlighted in bold; ties are bolded together. Here, (q, n, m) denotes the resulting MQ instance.

SNOVA Level I ($q = 16, n = 464, m = 80$)								
Model	ℓ	Part.	Selected partition	s	ours	JG [21]	Hash. [18]	Spec. [27]
classical	2	a	$(56, 6)$	18	188	218	188	**180**
		b	(1)					
quantum	5	a	$(14, 9, 6, 3, 1)$	47	**111**	120	125	–
		b	$(13, 8, 5, 2)$					
SNOVA Level III ($q = 16, n = 725, m = 125$)								
Model	ℓ	Part.	Selected partition	s	ours	JG [21]	Hash. [18]	Spec. [27]
classical	2	a	$(95, 6)$	24	285	368	285	**257**
		b	(0)					
quantum	5	a	$(18, 11, 11, 1, 1)$	83	**186**	196	197	–
		b	$(15, 10, 8, 0)$					
SNOVA Level V ($q = 16, n = 1120, m = 160$)								
Model	ℓ	Part.	Selected partition	s	ours	JG [21]	Hash. [18]	Spec. [27]
classical	2	a	$(124, 7)$	29	355	469	355	**343**
		b	(1)					
quantum	5	a	$(21, 14, 11, 5, 1)$	108	**235**	247	248	–
		b	$(19, 13, 10, 4)$					

mates than both Just Guess and Hashimoto's method for many parameter sets. These results indicate that allowing more flexible variable partitions can lead to improved complexity estimates for underdetermined MQ instances in both classical and quantum settings.

The parameter search also shows that, for several parameter sets, the selected partition parameters coincide with those corresponding to Hashimoto's method. This provides evidence for the optimality of Hashimoto's partitioning strategy for these cases within our generalized framework and cost model. Thus, the proposed framework not only improves the estimates for some parameters, but also provides a way to assess the optimality of existing algorithms for underdetermined MQ systems.

As future work, it would be important to develop an efficient implementation of the proposed algorithm and evaluate its practical performance. It would also be useful to develop a complexity evaluation that incorporates more detailed polynomial factors.

Acknowledgements

This work was supported by JST K Program Grant Number JPMJKP24U2, Japan.

References

1. C. Aguilar-Melchor, O. Blazy, J.-C. Deneuville, P. Gaborit, and G. Zémor, "Efficient encryption from random quasi-cyclic codes", IEEE Transactions on Information Theory, Vol.64, No.5, pp.3927–3943, 2018.
2. D. J. Bernstein, A. Hülsing, S. Kölbl, R. Niederhagen, J. Rijneveld, and P. Schwabe, "The SPHINCS+ Signature Framework", Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, pp.2129–2146, 2019.
3. L. Bettale, J.-C. Faugère, and L. Perret, "Hybrid Approach for Solving Multivariate Systems over Finite Fields", Journal of Mathematical Cryptology, Vol.2, pp.1–22, 2008.
4. W. Beullens, "MAYO: Practical Post-quantum Signatures from Oil-and-Vinegar Maps", Selected Areas in Cryptography, SAC 2021, LNCS 13203, pp.355–376, 2021.
5. W. Beullens, F. Campos, S. Celi, B. Hess, and M. J. Kannwischer, "MAYO Round 2 Version", 2025. `https://pqmayo.org/assets/specs/mayo-round2.pdf`
6. J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehlé, "CRYSTALS-Kyber: A CCA-Secure Module-Lattice-Based KEM", European Symposium on Security and Privacy, IEEE 2018, pp.353–367, 2018.
7. C. Cheng, Y. Hashimoto, H. Miura, and T. Takagi, "A Polynomial-Time Algorithm for Solving a Class of Underdetermined Multivariate Quadratic Equations over Fields of Odd Characteristics", International Workshop on Post-Quantum Cryptography, PQCrypto 2014, LNCS 8772, pp.40–58, 2014.
8. C. M. Cheng, T. Chou, R. Niederhagen, and B.-Y. Yang, "Solving quadratic equations with XL on parallel architectures", Conference on Cryptographic Hardware and Embedded Systems, CHES 2012, LNCS 7428, pp.356–373, 2012.
9. N. Courtois, A. Klimov, J. Patarin, and A. Shamir, "Efficient algorithms for solving overdefined systems of multivariate polynomial equations", Advances in Cryptology – EUROCRYPT 2000, pp.392–407, 2000.
10. L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé, "CRYSTALS-Dilithium: A Lattice-Based Digital Signature Scheme", IACR Transactions on Cryptographic Hardware and Embedded Systems, Vol.2018, No.1, pp.238–268, 2018.
11. J.-C. Faugère, "A New Efficient Algorithm for Computing Gröbner Bases (F5)", International Symposium on Symbolic and Algebraic Computation, ISSAC 2002, pp.75–83, 2002.
12. P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, Z. Zhang, et al., "Falcon: Fast-Fourier Lattice-based Compact Signatures over NTRU", Specification v1.2, NIST Post-Quantum Cryptography Standardization, 2020. `https://falcon-sign.info/falcon.pdf`
13. H. Furue, Y. Ikematsu, F. Hoshino, T. Takagi, H. Kosuge, K. Yamakoshi, R. Akiyama, S. Nakamura, S. Orihara, and K. Kinjo, "QR-UOV: Specification Document", Version 2.0, NIST PQC Additional Digital Signature Schemes, Round 2, 2025. `https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/qruov-spec-round2-web.pdf`

14. H. Furue, Y. Ikematsu, Y. Kiyomura, and T. Takagi, "A New Variant of Unbalanced Oil and Vinegar Using Quotient Ring: QR-UOV", Advances in Cryptology – ASIACRYPT 2021, LNCS 13093, pp.187–217, 2021.
15. H. Furue, S. Nakamura, and T. Takagi, "Improving Thomae-Wolf Algorithm for Solving Underdetermined Multivariate Quadratic Polynomial Problem", International Conference on Post-Quantum Cryptography, PQCrypto 2021, LNCS 12841, pp.65–78, 2021.
16. G. Fusco and E. Bach, "Phase transition of multivariate polynomial systems", Theory and Applications of Models of Computation, TAMC 2007, LNCS 4484, pp.632–645, 2007.
17. L. K. Grover, "A Fast Quantum Mechanical Algorithm for Database Search", Annual ACM Symposium on Theory of Computing, STOC 1996, pp.212–219, 1996.
18. Y. Hashimoto, "An Improvement of Algorithms to Solve Under-defined Systems of Multivariate Quadratic Equations", JSIAM Letters Vol.15, pp.53–56, 2023.
19. M. Iwata, R. Sugai, K. Sakata, and T. Takagi, "Explicit Bounds on the Existence Probability of Random Multivariate Quadratic Systems over Finite Fields", Arithmetic of Finite Fields, WAIFI 2026, LNCS 16611, pp.366–383, 2026.
20. A. Kipnis, J. Patarin, and L. Goubin, "Unbalanced Oil and Vinegar Signature Schemes", Advances in Cryptology – EUROCRYPT 1999, LNCS 1592, pp.206–222, 1999.
21. A. May, M. Ostuzzi, and H. Ressler, "Just Guess: Improved (Quantum) Algorithm for the Underdetermined MQ Problem", Advances in Cryptology – EUROCRYPT 2026, LNCS 16544, pp.334–362, 2026. See also Cryptology ePrint Archive, Report 2025/1788. `https://eprint.iacr.org/2025/1788`
22. National Institute of Standards and Technology, "Status Report on the Second Round of the Additional Digital Signature Schemes for the NIST Post-Quantum Cryptography Standardization Process", NIST Internal Report 8610, 2026. `https://nvlpubs.nist.gov/nistpubs/ir/2026/NIST.IR.8610.pdf`
23. M. Ostuzzi, "Pseudo-Oil Subspaces and the Geometry of Underdetermined MQ Problems", Cryptology ePrint Archive, Report 2026/1122, 2026. `https://eprint.iacr.org/2026/1122`
24. P. W. Shor, "Algorithms for Quantum Computation: Discrete Logarithms and Factoring", Proceedings of the 35th Annual Symposium on Foundations of Computer Science, pp.124–134, 1994.
25. E. Thomae and C. Wolf, "Solving Underdetermined Systems of Multivariate Quadratic Equations Revisited", International Workshop on Public Key Cryptography, PKC 2012, LNCS 7293, pp.156–171, 2012.
26. L.-C. Wang, P.-E. Tseng, Y.-L. Kuan, and C.-Y. Chou, "A Simple Noncommutative UOV Scheme", Cryptology ePrint Archive, Report 2022/1742, 2022. `https://eprint.iacr.org/2022/1742`
27. L.-C. Wang, C.-Y. Chou, J. Ding, Y.-L. Kuan, J. A. Leegwater, M.-S. Li, B.-S. Tseng, P.-E. Tseng, and C.-C. Wang, "SNOVA: Proposal for NIST PQC: Additional Digital Signature Schemes", Version 2.0, NIST PQC Additional Digital Signature Schemes, Round 2, 2025. `https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-2/spec-files/snova-spec-round2-web.pdf`
28. B.-Y. Yang, J.-M. Chen, and N. Courtois, "On Asymptotic Security Estimates in XL and Gröbner Bases-Related Algebraic Cryptanalysis", Information and Communications Security, ICICS 2004, LNCS 3269, pp.401–413, 2004.
29. B.-Y. Yang, O. C.-H. Chen, D. J. Bernstein, and J.-M. Chen, "Analysis of QUAD", Fast Software Encryption Workshop, FSE 2007, pp.290–308, 2007.

A Parameter Search Algorithm

In this section, we present the details of the parameter search algorithm described in Section 5, which finds parameters $(\ell, a_1, \ldots, a_\ell, b_1, \ldots, b_{\ell-1})$ that minimize the estimated complexity of the proposed method.

A.1 Implementation Details

We describe several implementation details used in the parameter search.

First, during the depth-first search over $a_1, \ldots, a_\ell$, the quantities U_i and V_i are updated incrementally. When a_i is newly chosen, we compute $U_i = U_{i-1} + a_i$ and $V_i = V_{i-1} + a_i U_{i-1}$. After the tuple $a_1, \ldots, a_\ell$ has been fixed, the quantities W_i are computed by

$$W_i = V_i + U_i(U_\ell - U_i) \qquad (1 \le i \le \ell - 1).$$

This formula follows from

$$W_i = \sum_{u=1}^{i} a_u \left(\sum_{v=u+1}^{\ell} a_v \right) = V_i + U_i(U_\ell - U_i).$$

Since W_i depends on U_ℓ, it is computed after all $a_1, \ldots, a_\ell$ have been determined.

Second, all cost estimates are stored in logarithmic form, namely as $\log_2 C$. That is, instead of storing a cost C, we store $\log_2 C$. Products of costs are then evaluated by addition in the logarithmic domain. Sums are evaluated using

$$\log_2(C_1 + C_2) = \max\{T_1, T_2\} + \log_2\left(1 + 2^{-|T_1 - T_2|}\right),$$

where $T_1 = \log_2 C_1$ and $T_2 = \log_2 C_2$. This representation avoids overflow and allows stable comparison of very large costs. For readability, the pseudocode below is written in the ordinary cost domain. In the actual implementation, all cost products and sums are evaluated in the logarithmic domain as described above.

Third, the costs of smaller MQ instances are precomputed. More precisely, we store

$$\mathrm{XL}[i][j] = \log_2 \mathrm{XL}(q, i, j)$$

for $0 \le i \le j \le m$. Since every reduced MQ instance appearing in the parameter search has at most m variables and at most m equations, this table is sufficient. After this precomputation, each XL-cost query can be answered in constant time by table lookup.

Finally, the search is parallelized by task decomposition. A naive static parallelization over ℓ or over the first few choices of a_i may cause load imbalance, because the size of the remaining search tree depends strongly on the early choices of the parameters. To reduce this imbalance, we fix ℓ and the first few a_i and regard the remaining search below this partial assignment as one task. These tasks are executed using OpenMP with dynamic scheduling.

Algorithm 1 Algorithm for complexity of the proposed method via parameter search

Require: $q, n, m \in \mathbb{Z}$

Ensure: Minimum estimated complexity for solving an MQ system over $\mathbb{F}_q$ with n variables and m equations using the proposed method

1: $best_cost \leftarrow +\infty$
2: $best_param \leftarrow \perp$
3: Compute the upper bound u_ℓ as (19)
4: **for** $\ell = 2, \ldots, u_\ell$ **do**
5: $\quad U_0 \leftarrow 0, V_0 \leftarrow 0$
6: $\quad$ DFS-A$(1, \emptyset, (U_0), (V_0))$
7: **end for**
8: **return** $best_cost, best_param$

Algorithm 2 DFS-A

Require: $i \in \mathbb{Z}$, integer sequences $(a_1, \ldots, a_{i-1}), (U_0, \ldots, U_{i-1}), (V_0, \ldots, V_{i-1})$

1: **if** $i = \ell + 1$ **then**
2: $\quad$ **for** $j = 1, \ldots, \ell - 1$ **do**
3: $\quad\quad W_j \leftarrow V_j + U_j(U_\ell - U_j)$
4: $\quad$ **end for**
5: $\quad$ Optimize-B$((a_1, \ldots, a_\ell), (U_1, \ldots, U_\ell), (V_1, \ldots, V_\ell), (W_1, \ldots, W_{\ell-1}))$
6: $\quad$ **return**
7: **end if**
8: Compute the upper bound u_{a_i} as (23)
9: **for** $a_i = 1, \ldots, u_{a_i}$ **do**
10: $\quad U_i \leftarrow U_{i-1} + a_i$
11: $\quad V_i \leftarrow V_{i-1} + a_i U_{i-1}$
12: $\quad$ DFS-A$(i + 1, (a_1, \ldots, a_i), (U_0, \ldots, U_i), (V_0, \ldots, V_i))$
13: **end for**
14: **return**

A.2 Description of the Algorithms

Algorithms 1–3 summarize the parameter search described above. The purpose of these algorithms is to find parameters $(\ell, a_1, \ldots, a_\ell, b_1, \ldots, b_{\ell-1})$ that minimize the estimated complexity of the proposed method under the applicability conditions (19), (23), (24). The variables *best_cost* and *best_param* are global variables shared by Algorithms 1–3.

Algorithm 1 controls the whole search. It first computes the upper bound u_ℓ for the number of partitions ℓ, and then enumerates all admissible values $2 \leq \ell \leq u_\ell$. For each fixed ℓ, the algorithm initializes $U_0 = 0$ and $V_0 = 0$, and calls Algorithm 2 to construct the sequence $a_1, \ldots, a_\ell$. The current best value of the estimated complexity is stored as best_cost.

Algorithm 2 performs a depth-first search over $a_1, \ldots, a_\ell$. Suppose that $a_1, \ldots, a_{i-1}$ have already been fixed. Then the admissible range of a_i is determined by the upper bound in (23). The algorithm tries all integers in this range. Whenever a

Algorithm 3 OPTIMIZE-B

Require: integer sequences $(a_1, \ldots, a_\ell), (U_1, \ldots, U_\ell), (V_1, \ldots, V_\ell), (W_1, \ldots, W_{\ell-1})$

$s \leftarrow m - U_\ell$
$cost \leftarrow q^s \cdot \mathrm{XL}(q, a_\ell, a_\ell)$
$b \leftarrow \emptyset$
for $i = 1, \ldots, \ell - 1$ **do**
    $best_local_cost \leftarrow +\infty$
    $best_b_i \leftarrow 0$
    Compute the upper bound u_{b_i} as (24)
    for $b_i = 0, \ldots, u_{b_i}$ **do**
        $local_cost \leftarrow a_i \cdot \mathrm{XL}(q, b_i + U_\ell - U_i, b_i + U_\ell - U_i)$
        if $i = 1$ **then**
            $local_cost \leftarrow local_cost + q^s \cdot \left(a_i b_i^2 + \mathrm{XL}(q, a_i - b_i, s + a_i - b_i)\right)$
        else
            $local_cost \leftarrow local_cost + q^s \cdot \left(a_i b_i^2 + \mathrm{XL}(q, a_i - b_i, a_i - b_i)\right)$
        end if
        if $local_cost < best_local_cost$ **then**
            $best_local_cost \leftarrow local_cost$
            $best_b_i \leftarrow b_i$
        end if
    end for
    $cost \leftarrow cost + best_local_cost$
    $b \leftarrow (b, best_b_i)$
    if $cost > best_cost$ **then**
        return
    end if
end for
$best_cost \leftarrow cost$
$best_param \leftarrow (\ell, a, b)$
return

new value of a_i is chosen, the quantities

$$U_i = U_{i-1} + a_i, \qquad V_i = V_{i-1} + a_i U_{i-1}$$

are updated. If the upper bound for a_i is smaller than 1, then no feasible continuation exists for the current prefix $a_1, \ldots, a_{i-1}$, and the branch is discarded.

When $i = \ell + 1$, the tuple $a_1, \ldots, a_\ell$ has been completely fixed. At this point, the algorithm computes

$$W_j = V_j + U_j(U_\ell - U_j) \qquad (1 \le j \le \ell - 1),$$

and calls Algorithm 3 to determine the parameters $b_1, \ldots, b_{\ell-1}$.

Algorithm 3 optimizes the b-parameters for a fixed tuple $a_1, \ldots, a_\ell$. First, it sets $s = m - U_\ell$ and initializes the cost with the term $q^s \mathrm{XL}(q, a_\ell, a_\ell)$, which corresponds to the subproblem for the last block. Then, for each $1 \le i \le \ell - 1$, the algorithm computes the admissible range of b_i from (24) and searches all values in this range.

The key point is that, once $a_1, \ldots, a_\ell$ are fixed, the terms depending on b_i are separated in the complexity formula. Therefore, it is unnecessary to enumerate all tuples $(b_1, \ldots, b_{\ell-1})$. Instead, each b_i can be chosen independently so as to minimize its local contribution to the total cost. More precisely, the local cost for b_i is

$$a_i \mathrm{XL}(q, b_i + U_\ell - U_i, b_i + U_\ell - U_i)$$

together with one repeated-trial term multiplied by q^s. For $i = 1$, this repeated-trial term is

$$q^s \left(a_i b_i^2 + \mathrm{XL}(q, a_i - b_i, s + a_i - b_i) \right),$$

whereas for $2 \leq i \leq \ell - 1$, it is

$$q^s \left(a_i b_i^2 + \mathrm{XL}(q, a_i - b_i, a_i - b_i) \right).$$

The algorithm selects the value of b_i that minimizes this local cost and adds the resulting value to the cumulative cost.

The algorithm also uses a simple pruning rule. If the cumulative cost already exceeds the current best value best_cost, then the current candidate cannot improve the best cost, because all remaining cost terms are nonnegative. Hence, the evaluation of the current tuple $a_1, \ldots, a_\ell$ is stopped immediately.

In summary, the search avoids a direct exhaustive search over all parameters in two ways. First, the bounds on a_i allow infeasible prefixes to be discarded during the depth-first search. Second, the independence of the b_i's reduces the search over all b-tuples to separate one-dimensional optimizations. These two properties make the parameter search practical for the ranges considered in our experiments.