New Quantum-Classical Algorithm for the Discrete Logarithm Problem over $\mathbb{Z}_p^*$ *

Hidenori Kuwakado[1] and Shoichi Hirose[2]

[1] Kansai University, 2–1–1 Ryozenji-cho, Takatsuki-shi, Osaka, 569-1095 Japan

[2] University of Fukui, 3–9–1 Bunkyo, Fukui-shi, Fukui 910-8507, Japan

Abstract. Shor demonstrated that the discrete logarithm problem in the multiplicative group $\mathbb{Z}_p^*$, where p is an odd prime, can be solved efficiently using a period-finding algorithm based on the quantum Fourier transform. In this paper, we propose a quantum-classical algorithm based on algebraic properties that do not rely on periodicity. Specifically, we show that the hardcore predicate for the discrete logarithm problem, which was introduced by Blum and Micali, can be reduced, using the swap test, to the problem of distinguishing between two Bernoulli distributions. In our algorithm, although the swap test is used as a quantum subroutine, most of the computation is performed classically. Moreover, the algorithm does not require the quantum Fourier transform.

Keywords: Quantum algorithm · Discrete logarithm problem · Hardcore predicate · Swap test

1 Introduction

Shor's algorithms for factoring integers and computing discrete logarithms are widely regarded as among the most groundbreaking and influential achievements in quantum computing [7]. They can be understood as quantum algorithms for period finding using the quantum Fourier transform. Shor's algorithms are quantum-classical algorithms, in which the core computation is carried out by quantum circuits supported by classical processing.

We propose a new quantum-classical algorithm to solve the discrete logarithm problem. Unlike Shor's algorithm, our approach is not based on period finding, but rather on the hardcore predicate for the discrete logarithm problem, which was introduced by Blum and Micali [3]. Specifically, our algorithm recovers the bits of the discrete logarithm sequentially from the least significant bit to the most significant bit, using a quantum subroutine that computes the hardcore predicate. The swap test [2, 4] plays a central role in the quantum subroutine. The quantum subroutine also differs from Shor's algorithm in that it employs swap gates rather than the quantum Fourier transform. While the quantum

* A preliminary version of this paper was presented at the 54th Quantum Information Technology Symposium [1].

subroutine is the most critical component, most of the computation is performed classically.

This paper is organized as follows: Section 2 describes the hardcore predicate introduced by Blum and Micali and a polynomial-time classical algorithm for solving the discrete logarithm problem using the predicate as an oracle. After defining a principal and a non-principal square roots associated with the predicate, we describe how to compute square roots. Next, we describe the swap test in detail. Section 3 describes a quantum procedure for evaluating the hardcore predicate using the swap test. An example execution of the quantum procedure is given in Appendix B. Then, we show that evaluating the hardcore predicate is reducible to distinguishing between two Bernoulli distributions. We derive the misclassification probability, which is the probability of incorrectly evaluating the hardcore predicate. We also estimate the number of executions of the swap test required to ensure that the probability of correctly recovering the discrete logarithm is greater than 0.5. Section 4 gives concluding remarks.

2 Preliminaries

2.1 Discrete Logarithm Problem and Its Hardcore Predicate

We define the discrete logarithm problem addressed in this paper. Let p be an odd prime greater than 3. Let g be an element of order $p-1$, and let $\mathbb{Z}_p^*$ denote the cyclic group generated by g, defined by

$$\mathbb{Z}_p^* = \{g^i \bmod p \mid 0 \le i < p-1\}.$$

The *discrete logarithm problem* is defined as follows: Given an odd prime p, a generator g of $\mathbb{Z}_p^*$, and an element $y \in \mathbb{Z}_p^*$, find the integer x such that $g^x \equiv y \pmod p$ and $0 \le x < p-1$.

Blum and Micali [3] defined a hardcore predicate for the discrete logarithm problem, and presented an algorithm for computing discrete logarithms that runs in $O(\log p)$ modular arithmetic operations and makes $O(\log p)$ queries to an oracle for this predicate. We describe the hardcore predicate introduced by Blum and Micali. Every quadratic residue in $\mathbb{Z}_p^*$ has exactly two square roots in $\mathbb{Z}_p^*$. A quadratic residue in $\mathbb{Z}_p^*$ can be written as g^{2w} with $0 \le w < (p-1)/2$. Its square roots are g^w and $g^{w+(p-1)/2}$, which can be computed using methods described in Sect. 2.2. Of these two roots, g^w is called the *principal* square root, and $g^{w+(p-1)/2}$ is called the *non-principal* square root [3]. The *Blum-Micali hardcore predicate* takes a generator and a square root as inputs and determines whether the square root is principal.

For the discussion in later sections, Fig. 1 shows a slightly modified version of the Blum-Micali hardcore predicate. The hardcore predicate in Fig. 1 differs from the original Blum-Micali hardcore predicate only in the range of w; in the original definition, w ranges over $0 \le w < (p-1)/2$.

Figure 2 shows an algorithm obtained by modifying the Blum-Micali algorithm so that it uses the oracle for the predicate in Fig. 1. This algorithm runs in

Input: p, g, $y \in \mathbb{Z}_p^*$ s.t. $y \equiv g^{2w} \bmod p$ with $0 \leq w < (p-1)/4$, and $u \in \mathbb{Z}_p^*$ s.t. $u^2 \equiv y \bmod p$
Output: True or False
1: **if** u is principal **then** ▷ $u \equiv g^w \pmod p$
2: **return** True
3: **else** ▷ non-principal, i.e., $u \equiv g^{w+(p-1)/2} \pmod p$
4: **return** False
5: **end if**

Fig. 1. Modified hardcore predicate.

$O(\log p)$ modular arithmetic operations and makes $O(\log p)$ queries to an oracle for the predicate in Fig. 1. The algorithm recovers the discrete logarithm of y bit by bit, starting from the least significant bit. Note that the least significant bit of the discrete logarithm of y can be determined from the Legendre symbol $\mathtt{L}(y, p)$. Moreover, $\mathtt{L}(y, p)$ can be computed using $O(\log p)$ modular arithmetic operations [5, 6]. If the oracle always returns correct answers, then the algorithm computes the discrete logarithm of y using $O(\log p)$ modular arithmetic operations and oracle queries. In this sense, the hardness of the discrete logarithm problem in a classical setting is caused by the fact that no efficient algorithm for evaluating the predicate is known.

2.2 Computing Square Roots Modulo p

Lemma 1 ([5, 6]). *Let $y \in \mathbb{Z}_p^*$ be a quadratic residue. There exists an algorithm for computing a square root $u \in \mathbb{Z}_p^*$ of y using $O(\log p)$ modular arithmetic operations. The other square root of y is $p - u$.*

We describe well-known methods for computing a square root of y modulo p [5, 6]. Note that these methods compute a square root of y without determining the discrete logarithm of y with respect to a fixed generator. We also show that it is impossible to determine from the output alone whether it is the principal square root.

The Tonelli-Shanks algorithm probabilistically computes a square root of y for any odd prime p and any quadratic residue y modulo p [5, 6]. The core idea of the Tonelli-Shanks algorithm is as follows. Let $p = 2^e \cdot p' + 1$ where p' is odd, and let $\mathbb{Z}_p^{(2)}$ denote the cyclic subgroup of $\mathbb{Z}_p^*$ of order 2^e defined by

$$\mathbb{Z}_p^{(2)} = \{g^{p' \cdot i} \bmod p \mid 0 \leq i < 2^e\},$$

where g is a generator of $\mathbb{Z}_p^*$. Let $\xi = g^{p'} \bmod p$, which is a generator of $\mathbb{Z}_p^{(2)}$. Then there exists an even integer κ with $0 \leq \kappa < 2^e$ satisfying

$$y^{p'} \cdot \xi^{\kappa} \equiv 1 \pmod p,$$

or equivalently

$$2w + \kappa \equiv 0 \pmod{2^e}. \tag{1}$$

Input: p, g, and $y \in \mathbb{Z}_p^*$.
Output: $\perp$ or an integer x s.t. $y \equiv g^x \bmod p$ and $0 \le x \le p-2$.

$y_{\rm org} \leftarrow y$.
$p_{\rm qut} \leftarrow \lfloor (p-1)/4 \rfloor$.
$n \leftarrow \lfloor \log_2 p_{\rm qut} \rfloor + 1$. ▷ $w = \sum_{i=0}^{n-1} b_i 2^i$, $b_i \in \{0,1\}$ in Fig. 1.
for $\iota \leftarrow 0$ to 3 **do**
    $y \leftarrow y_{\rm org} \cdot g^{-\iota \cdot p_{\rm qut}}$.
    for $i \leftarrow 0$ to $n-1$ **do**
        Compute the Legendre symbol $\mathtt{L}(y, p)$.
        if $\mathtt{L}(y,p) = 1$ **then**
            $b_i \leftarrow 0$.
        else ▷ $\mathtt{L}(y,p) = -1$
            $b_i \leftarrow 1$.
            $y \leftarrow y \cdot g^{-1} \bmod p$.
        end if ▷ Now y is a quadratic residue.
        Compute a square root u of y using the method described in Sect. 2.2.
        Query the oracle for the predicate in Fig. 1 to determine whether u is principal.
        if u is principal **then**
            $y \leftarrow u$.
        else
            $y \leftarrow p - u$.
        end if
    end for
    $x \leftarrow \iota \cdot p_{\rm qut} + \sum_{i=0}^{n-1} b_i 2^i$.
    if $g^x \equiv y_{\rm org} \pmod p$ **then**
        return x.
    end if
end for
return $\perp$.

Fig. 2. Modified algorithm for solving the discrete logarithm problem.

A square root u of y is given by

$$u \equiv y^{\frac{p'+1}{2}} \cdot \xi^{\frac{\kappa}{2}} \pmod p. \tag{2}$$

Using Eq. (1), we can verify that u is a square root of y, i.e., $u^2 \equiv y \pmod p$. Note that the Tonelli-Shanks algorithm can output u without explicitly determining κ. In cryptographic applications, since e is small, κ could also be found by exhaustive search over the range $0 \le \kappa < 2^e$, without using the refined technique of the Tonelli-Shanks algorithm. In either case, the value of $w \bmod 2^{e-1}$, which gives partial information about the discrete logarithm of y, can be determined.

We examine Eq. (2) in detail. Let $y \equiv g^{2w} \pmod p$ where $0 \le w < (p-1)/2$. Since this paper focuses on the discrete logarithm problem, we regard w itself as unknown. Substituting this congruence into Eq. (2) yields

$$u \equiv g^w \cdot g^{p' \cdot (w + \frac{\kappa}{2})} \pmod p. \tag{3}$$

From Eq. (1), we have

$$w + \frac{\kappa}{2} \equiv 0 \pmod{2^{e-1}},$$

or equivalently, there is an integer χ such that $\chi \geq 1$ and

$$w + \frac{\kappa}{2} = \chi \cdot 2^{e-1}. \tag{4}$$

Substituting this equation into Eq. (3) gives

$$u \equiv g^w \cdot g^{2^{e-1}p' \cdot \chi} \pmod{p}.$$

This congruence implies that whether u is the principal square root depends on the parity of χ. Equation (4) implies that determining the parity of χ reduces to determining the value of $w \bmod 2^e$. Hence, we cannot determine from u alone whether u is principal.

If $p \equiv 3 \pmod 4$ or $p \equiv 5 \pmod 8$, then a square root of y can be computed deterministically and more efficiently than using the Tonelli-Shanks algorithm [5, 6]. When $p \equiv 3 \pmod 4$, a square root of y is given by

$$u = y^{\frac{p+1}{4}} \bmod p.$$

Let w be an integer satisfying $0 \leq w < (p-1)/2$ and $g^{2w} \equiv y \pmod p$. Using the above formula, we have

$$\begin{aligned} u &\equiv g^{2w \cdot \frac{p+1}{4}} \pmod p \\ &\equiv g^w \cdot g^{\frac{w(p-1)}{2}} \pmod p. \end{aligned}$$

Hence, whether u is the principal square root is determined by the parity of w. Since the parity of w is unknown, we cannot determine from u alone whether u is the principal square root.

When $p \equiv 5 \pmod 8$, a square root of y can be computed as follows:

$$u = \begin{cases} y^{\frac{p+3}{8}} \bmod p & \text{if } y^{\frac{p-1}{4}} \equiv 1 \pmod p, \\ 2y \cdot (4y)^{\frac{p-5}{8}} \bmod p & \text{if } y^{\frac{p-1}{4}} \equiv -1 \pmod p. \end{cases} \tag{5}$$

Note that $y^{(p-1)/4} \bmod p$ is either 1 or -1 since y is a quadratic residue. Let w be an integer satisfying $0 \leq w < (p-1)/2$ and $g^{2w} \equiv y \pmod p$. The first condition $y^{(p-1)/4} \equiv 1 \pmod p$ implies that w is even, that is, $w \equiv 0 \pmod 4$ or $w \equiv 2 \pmod 4$. Then, the square root u can be written as

$$\begin{aligned} u &\equiv g^{2w \cdot \frac{p+3}{8}} \pmod p \\ &\equiv g^w \cdot g^{w \cdot \frac{p-1}{4}} \pmod p. \end{aligned}$$

Since the value of $w \bmod 4$ is unknown, we cannot determine whether u is the principal square root. The second condition $y^{(p-1)/4} \equiv -1 \pmod{p}$ implies that w is odd. Then, the square root u can be written as

$$\begin{aligned} u &\equiv 2g^{2w} \cdot (4g^{2w})^{\frac{p-5}{8}} \pmod{p} \\ &\equiv g^w \cdot (2g^w)^{\frac{p-1}{4}} \pmod{p}. \end{aligned}$$

Since 2 is a quadratic non-residue modulo p when $p \equiv 5 \pmod 8$, there exists an integer t satisfying $0 \le t < (p-1)/2$ and $g^{2t+1} \equiv 2 \pmod{p}$. Using this fact, we can rewrite the above congruence as

$$u \equiv g^w \cdot \left(g^{2t+1+w}\right)^{\frac{p-1}{4}} \pmod{p}.$$

Since $2t+1+w$ is even, we have

$$\left(g^{2t+1+w}\right)^{\frac{p-1}{4}} \equiv \begin{cases} 1 & \text{if } 2t+1+w \equiv 0 \pmod 4, \\ g^{\frac{p-1}{2}} & \text{if } 2t+1+w \equiv 2 \pmod 4. \end{cases}$$

Since the value of $2t+1+w \bmod 4$ is unknown, we cannot determine whether u is principal.

The above discussion suggests that no known efficient classical algorithm for computing square roots modulo p is guaranteed to return the principal square root. Conversely, the existence of such a classical algorithm would imply an efficient classical algorithm for solving the discrete logarithm problem.

2.3 Swap Test

The swap test is a quantum procedure for quantifying the similarity of two quantum states [2, 4]. Let $|\phi\rangle$ and $|\psi\rangle$ be two multi-qubit states. For notational clarity, we denote single-qubit states by $|\cdot\rangle_1$ and $\langle\cdot|_1$ to distinguish them from multi-qubit states.

The initial state is prepared as $|0\rangle_1 |\phi\rangle |\psi\rangle$. After applying the Hadamard gate H to the first qubit, the state becomes

$$|0\rangle_1 |\phi\rangle |\psi\rangle \xrightarrow{H\otimes I} \frac{1}{\sqrt{2}} \left(|0\rangle_1 |\phi\rangle |\psi\rangle + |1\rangle_1 |\phi\rangle |\psi\rangle\right),$$

where I denotes the identity operator. Applying a controlled-SWAP gate with the first qubit as the control qubit yields

$$\frac{1}{\sqrt{2}} \left(|0\rangle_1 |\phi\rangle |\psi\rangle + |1\rangle_1 |\phi\rangle |\psi\rangle\right) \xrightarrow{\text{C-SWAP}} \frac{1}{\sqrt{2}} \left(|0\rangle_1 |\phi\rangle |\psi\rangle + |1\rangle_1 |\psi\rangle |\phi\rangle\right).$$

Applying the Hadamard gate to the first qubit again yields

$$\begin{aligned} &\frac{1}{\sqrt{2}} \left(|0\rangle_1 |\phi\rangle |\psi\rangle + |1\rangle_1 |\psi\rangle |\phi\rangle\right) \\ &\quad \xrightarrow{H\otimes I} \frac{1}{2} |0\rangle_1 \left(|\phi\rangle |\psi\rangle + |\psi\rangle |\phi\rangle\right) + \frac{1}{2} |1\rangle_1 \left(|\phi\rangle |\psi\rangle - |\psi\rangle |\phi\rangle\right). \end{aligned}$$

Let $|\Psi\rangle$ denote the above state. We measure the first qubit using the measurement operators $\{|0\rangle_1 \langle 0|_1 \otimes I, |1\rangle_1 \langle 1|_1 \otimes I\}$. The probability of obtaining outcome $|0\rangle_1$ is given by

$$\begin{aligned}\Pr(|0\rangle_1) &= \langle\Psi| (|0\rangle_1 \langle 0|_1 \otimes I) |\Psi\rangle \\ &= \frac{1}{2} (\langle\psi| \langle\phi| + \langle\phi| \langle\psi|) \cdot \frac{1}{2} (|\phi\rangle |\psi\rangle + |\psi\rangle |\phi\rangle) \\ &= \frac{1}{2} \left(1 + |\langle\phi \mid \psi\rangle|^2\right). \end{aligned} \tag{6}$$

Conditioned on observing $|0\rangle_1$, the post-measurement state is given by

$$\frac{(|0\rangle_1 \langle 0|_1 \otimes I) |\Psi\rangle}{\sqrt{\Pr(|0\rangle_1)}} = \frac{|0\rangle_1 (|\phi\rangle |\psi\rangle + |\psi\rangle |\phi\rangle)}{\sqrt{2(1 + |\langle\phi \mid \psi\rangle|^2)}}. \tag{7}$$

The probability of observing $|1\rangle_1$ is computed in a similar manner:

$$\begin{aligned}\Pr(|1\rangle_1) &= \langle\Psi| (|1\rangle_1 \langle 1|_1 \otimes I) |\Psi\rangle \\ &= \frac{1}{2} (\langle\psi| \langle\phi| - \langle\phi| \langle\psi|) \cdot \frac{1}{2} (|\phi\rangle |\psi\rangle - |\psi\rangle |\phi\rangle) \\ &= \frac{1}{2} \left(1 - |\langle\phi \mid \psi\rangle|^2\right). \end{aligned}$$

The post-measurement state is given by

$$\frac{(|1\rangle_1 \langle 1|_1 \otimes I) |\Psi\rangle}{\sqrt{\Pr(|1\rangle_1)}} = \frac{|1\rangle_1 (|\phi\rangle |\psi\rangle - |\psi\rangle |\phi\rangle)}{\sqrt{2(1 - |\langle\phi \mid \psi\rangle|^2)}}. \tag{8}$$

Equations (7) and (8) show that the information about $|\phi\rangle$ and $|\psi\rangle$ is preserved in the post-measurement state. In the setting of Sect. 3.1, since $|\phi\rangle$ and $|\psi\rangle$ are known, the state $|\phi\rangle |\psi\rangle$ could theoretically be recovered if appropriate operations were applied to the post-measurement state.

Let $|\phi\rangle$ and $|\psi\rangle$ be defined by

$$|\phi\rangle = \sum_{i=0}^{t-1} a_i |i\rangle, \text{ and } |\psi\rangle = \sum_{i=0}^{t-1} b_i |i\rangle,$$

where $|i\rangle$ denotes a $\lfloor \log_2 t \rfloor + 1$-qubit basis state and the coefficients satisfy

$$\sum_{i=0}^{t-1} |a_i|^2 = 1, \text{ and } \sum_{i=0}^{t-1} |b_i|^2 = 1.$$

Then the squared inner product $|\langle\phi \mid \psi\rangle|^2$ is given by

$$|\langle\phi \mid \psi\rangle|^2 = \left| \sum_{i=0}^{t-1} a_i^* b_i \right|^2,$$

where a_i^* denotes the complex conjugate of a_i. Consequently, the swap test quantifies the similarity of two states via the squared inner product of their amplitude vectors.

3 Quantum Algorithm for the Hardcore Predicate

3.1 Quantum Algorithm Using the Swap Test

We propose a quantum algorithm for determining whether a given $u \in \mathbb{Z}_p^*$ is principal, thereby evaluating the hardcore predicate in Fig. 1. We denote $\lfloor (p-1)/4 \rfloor$ by $p_{\rm qut}$, i.e.,

$$p_{\rm qut} = \lfloor (p-1)/4 \rfloor .$$

The initial state $|\Phi\rangle$ is prepared as follows:

$$|\Phi\rangle = |0\rangle_1 |\phi\rangle |\psi\rangle$$

where $|\phi\rangle$ and $|\psi\rangle$ are defined by

$$|\phi\rangle = \frac{1}{\sqrt{\frac{p-1}{2}}} \sum_{k=0}^{(p-3)/2} |g^k \bmod p\rangle , \tag{9}$$

$$|\psi\rangle = \frac{1}{\sqrt{p_{\rm qut}}} \sum_{k=0}^{p_{\rm qut}-1} |u \cdot g^k \bmod p\rangle . \tag{10}$$

Note that $|\phi\rangle$ is independent of u, while $|\psi\rangle$ depends on u. To prepare these states, we assume oracle access to a unitary operator $U_{g,p}$ defined by

$$U_{g,p} |k\rangle |d\rangle = |k\rangle |d \cdot g^k \bmod p\rangle .$$

This operator is analogous to the one used in Shor's algorithm. In Appendix A, we give the explicit procedures for preparing $|\phi\rangle$ and $|\psi\rangle$ using this operator.

Suppose that u is principal, that is, there exists an integer w satisfying $0 \le w < (p-1)/4$ and $g^w \equiv u \pmod p$. Note that this is the range of w in the predicate in Fig. 1. Then, Eq. (10) can be written as

$$|\psi\rangle = \frac{1}{\sqrt{p_{\rm qut}}} \sum_{k=0}^{p_{\rm qut}-1} |g^{w+k} \bmod p\rangle .$$

Since $w + p_{\rm qut} - 1 \le (p-3)/2$ holds for this range of w, all basis states in $|\psi\rangle$ are included in $|\phi\rangle$. Therefore, we have

$$\begin{aligned} \langle \phi \mid \psi \rangle &= p_{\rm qut} \cdot \frac{1}{\sqrt{p_{\rm qut}}} \cdot \frac{1}{\sqrt{\frac{p-1}{2}}} \\ &= \begin{cases} \sqrt{\frac{1}{2}} & \text{if } p \equiv 1 \pmod 4 \\ \sqrt{\frac{1}{2} - \frac{1}{p-1}} & \text{if } p \equiv 3 \pmod 4. \end{cases} \end{aligned}$$

Substituting these expressions into Eq. (6) yields

$$\Pr\left(|0\rangle_1\right) = \begin{cases} \frac{3}{4} & \text{if } p \equiv 1 \pmod 4 \\ \frac{3}{4} - \frac{1}{2(p-1)} & \text{if } p \equiv 3 \pmod 4. \end{cases}$$

The term $1/(2(p-1))$ is negligible for $p \equiv 3 \pmod 4$ since p is large in cryptographic applications. In the rest of this paper, we neglect this term.

Suppose that u is non-principal, that is, there exists an integer w satisfying $0 \leq w < (p-1)/4$ and $g^{w+(p-1)/2} \equiv u \pmod p$. Then, Eq. (10) can be written as

$$|\psi\rangle = \frac{1}{\sqrt{p_{\mathrm{qut}}}} \sum_{k=0}^{p_{\mathrm{qut}}-1} |g^{w+k+(p-1)/2} \bmod p\rangle .$$

Since $(p-1)/2 < w + k + (p-1)/2 \leq p-2$ for all $0 \leq k \leq p_{\mathrm{qut}} - 1$, the basis states in $|\psi\rangle$ are completely disjoint from those in $|\phi\rangle$. Therefore, we have

$$\langle \phi \mid \psi \rangle = 0.$$

Substituting this value into Eq. (6) yields

$$\Pr\left(|0\rangle_1\right) = \frac{1}{2},$$

which is also independent of p. A simple example illustrating the above procedure is given in Appendix B.

In summary, if u is principal, then the probabilities of observing $|0\rangle_1$ and $|1\rangle_1$ are approximately $3/4$ and $1/4$, respectively. If u is non-principal, then both probabilities are $1/2$. Therefore, the problem of determining whether u is principal reduces to distinguishing between the two Bernoulli distributions $B(3/4)$ and $B(1/2)$, where $B(q)$ denotes the discrete probability distribution of a random variable that takes the value 1 with probability q and the value 0 with probability $1-q$. Note that the value 1 in the Bernoulli distribution corresponds to observing $|0\rangle_1$.

Theorem 1. *Using the swap test, we can reduce the problem of evaluating the hardcore predicate for the discrete logarithm problem to distinguishing between the two Bernoulli distributions $B(3/4)$ and $B(1/2)$.*

3.2 Identification of the Bernoulli Distribution

We describe a method for distinguishing between the two Bernoulli distributions $B(3/4)$ and $B(1/2)$. To minimize the probability of misclassification, we use the maximum a posteriori (MAP) rule. Let ℓ be the number of trials, and let s denote the number of outcomes equal to 1, which corresponds to the number of $|0\rangle_1$ outcomes in our algorithm. When the distribution is $B(3/4)$, the probability that the number of outcomes equal to 1 is s is given by

$$\Pr\left(s \mid B(3/4)\right) = \binom{\ell}{s} \cdot \left(\frac{3}{4}\right)^s \left(\frac{1}{4}\right)^{\ell-s}.$$

Similarly, we have

$$\Pr\left(s \mid B(1/2)\right) = \binom{\ell}{s} \cdot \left(\frac{1}{2}\right)^{\ell}.$$

No prior knowledge is available about the discrete logarithm of the given y, except for the value of its least significant bit. In other words, it is unknown whether the square root u of y is principal. It is therefore reasonable to assign equal prior probabilities to the two distributions. Then the decision rule for selecting $B(3/4)$ reduces to

$$\left(\frac{3}{4}\right)^s \left(\frac{1}{4}\right)^{\ell-s} \geq \left(\frac{1}{2}\right)^{\ell}. \tag{11}$$

Hence, if $s \geq \ell \cdot \log_3 2$, then we estimate the distribution to be $B(3/4)$; otherwise, we estimate it to be $B(1/2)$.

Next, we evaluate the probability of misclassification when the above decision rule is used. Let η denote the threshold defined by

$$\eta = \lceil \ell \cdot \log_3 2 \rceil . \tag{12}$$

Since the prior probabilities assigned to $B(3/4)$ and $B(1/2)$ are equal, the misclassification probability $\Pr^{\mathrm{err}}(\ell)$ is given by

$$\Pr^{\mathrm{err}}(\ell) = \frac{1}{2}\sum_{s=0}^{\eta-1} \binom{\ell}{s} \left(\frac{3}{4}\right)^s \left(\frac{1}{4}\right)^{\ell-s} + \frac{1}{2}\sum_{s=\eta}^{\ell} \binom{\ell}{s} \left(\frac{1}{2}\right)^{\ell} \tag{13}$$

$$= \frac{1}{2}\sum_{s=0}^{\ell} \binom{\ell}{s} \cdot \min\left(\left(\frac{3}{4}\right)^s \left(\frac{1}{4}\right)^{\ell-s}, \left(\frac{1}{2}\right)^{\ell}\right), \tag{14}$$

where the last equality follows from Eq. (11) and Eq. (12). In general, the following inequality holds for any positive real numbers a, b and any real number θ with $0 < \theta < 1$:

$$\min(a, b) \leq a^{\theta} b^{1-\theta}.$$

Using this inequality, Eq. (14) can be bounded as follows:

$$\begin{aligned}
\Pr^{\mathrm{err}}(\ell) &\leq \frac{1}{2}\sum_{s=0}^{\ell} \binom{\ell}{s} \cdot \left(\left(\frac{3}{4}\right)^s \left(\frac{1}{4}\right)^{\ell-s}\right)^{\theta} \cdot \left(\frac{1}{2}\right)^{\ell\cdot(1-\theta)} \\
&\leq \frac{1}{2^{\ell(1+\theta)+1}} \sum_{s=0}^{\ell} \binom{\ell}{s} \cdot 3^{\theta\cdot s} \\
&\leq \frac{1}{2} \cdot \left(\frac{1+3^{\theta}}{2^{1+\theta}}\right)^{\ell}.
\end{aligned} \tag{15}$$

We now minimize the right-hand side of the above inequality. For $0 < \theta < 1$, define

$$f(\theta) = \frac{1+3^{\theta}}{2^{1+\theta}}.$$

The derivative of f with respect to θ is given by

$$f'(\theta) = \frac{3^{\theta}\ln(3/2) - \ln 2}{2^{1+\theta}},$$

where ln denotes the natural logarithm. Solving $f'(\theta) = 0$ yields

$$\begin{aligned}\theta_{\min} &= \log_3 \left(\frac{\ln 2}{\ln(3/2)} \right) \\ &\approx 0.488,\end{aligned}$$

which satisfies the condition $0 < \theta < 1$. The function f attains its minimum at $\theta = \theta_{\min}$. Substituting $\theta_{\min}$ into Ineq. (15) gives

$$\begin{aligned}\Pr^{\mathrm{err}}(\ell) &\leq \frac{1}{2} \cdot \left(\frac{1 + 3^{\theta_{\min}}}{2^{1+\theta_{\min}}} \right)^{\ell} \\ &\leq \frac{1}{2} \cdot 0.966^{\ell}. \end{aligned} \tag{16}$$

Hence, the misclassification probability converges to 0 as the number of trials ℓ increases. Moreover, this probability is independent of p.

3.3 Number of Executions of the Swap Test

The purpose of the procedure in Sect. 3.2 is to identify the principal square root, that is, to evaluate the predicate shown in Fig. 1. As can be seen from Fig. 2, the identification procedure must succeed in n consecutive predicate evaluations to recover the correct x. Hence, the probability of recovering the correct x is

$$\Pr^{\mathrm{suc}}(\ell, n) = \left(1 - \Pr^{\mathrm{err}}(\ell)\right)^n, \tag{17}$$

where $n = \lfloor \log_2 p_{\mathrm{qut}} \rfloor + 1$ and $p_{\mathrm{qut}} = \lfloor (p-1)/4 \rfloor$, as defined in Fig. 2. Let $\ell_{\min}$ be the smallest value of ℓ satisfying $\Pr^{\mathrm{suc}}(\ell, n) > 0.5$. Table 1 shows $\ell_{\min}$ corresponding to typical bit lengths of p, where $\Pr^{\mathrm{err}}(\ell)$ is computed using Eq. (13).

We consider an asymptotic bound on ℓ. Substituting Eq. (16) into Eq. (17) yields

$$\Pr^{\mathrm{suc}}(\ell, n) \geq \left(1 - \frac{1}{2} \cdot 0.966^{\ell}\right)^n.$$

Since we now consider the case where $\Pr^{\mathrm{suc}}(\ell, n) > 0.5$, it suffices to require

$$\left(1 - \frac{1}{2} \cdot 0.966^{\ell}\right)^n > 0.5.$$

The minimum value of ℓ satisfying the above inequality, denoted by ℓ_{ubn}, is

$$\ell_{\mathrm{ubn}} = \left\lceil \log_{0.966} \left(2 - \frac{1}{2^{\frac{1}{n}-1}} \right) \right\rceil. \tag{18}$$

The values of ℓ_{ubn} are shown in the third row of Table 1. Furthermore, ℓ_{ubn} satisfies the following asymptotic bound:

$$\ell_{\mathrm{ubn}} = O(\log n).$$

Table 1. Smallest values of ℓ satisfying $\Pr^{\mathrm{suc}}(\ell, n) > 0.5$.

Bit length of p	1024	2048	4096
$\ell_{\min}$	148	167	185
ℓ_{ubn}	195	215	236

Table 2. Maximum numbers of swap tests for $\Pr^{\mathrm{suc}}(\ell, n) > 0.5$.

Bit length of p	1024	2048	4096
Swap test	605,616	1,367,396	3,030,300

The derivation of this bound is given in Appendix C. Since $n = \lfloor \log_2 p_{\mathrm{qut}} \rfloor + 1$, this implies that

$$\ell_{\mathrm{ubn}} = O(\log(\log p)).$$

Therefore, the number of swap tests required to determine whether the square root u is the principal square root increases slowly as p increases.

Since Step 15 in Fig. 2 is executed at most $4n$ times, the maximum number of swap tests required to achieve $\Pr^{\mathrm{suc}}(\ell, n) > 0.5$ is obtained by multiplying $4n$ by the corresponding $\ell_{\min}$ in Table 1. The resulting values are shown in Table 2. Although the numbers in Table 2 may seem large, they are asymptotically $O((\log p) \cdot (\log(\log p)))$.

4 Concluding Remarks

We have proposed the quantum-classical algorithm to solve the discrete logarithm problem. Our algorithm employs the swap test to determine each bit of the discrete logarithm. The hardcore predicate for the discrete logarithm problem, introduced by Blum and Micali, plays a central role in our approach. Unlike Shor's algorithm, our algorithm relies neither on the quantum Fourier transform nor on period finding.

As in the case of Shor's algorithm, the complexity of the quantum circuit is dominated by the exponentiation unitary rather than by the quantum Fourier transform or the swap test. Therefore, implementing the exponentiation unitary is crucial. A significant feature of the swap test is that the information about $|\phi\rangle$ and $|\psi\rangle$ is retained in the post-measurement state. Since $|\phi\rangle$ and $|\psi\rangle$ are known, the state $|\phi\rangle|\psi\rangle$ could theoretically be recovered if appropriate operations were applied to the post-measurement state. That is, it could suffice to apply the exponentiation unitary only once to prepare both $|\phi\rangle$ and $|\psi\rangle$ as shown in Eq. (9) and Eq. (10).

Future work includes estimating the number of quantum gates required for the swap test in our algorithm and comparing this number with the number required for the quantum Fourier transform.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number 26K14832.

References

1. The 54th Quantum Information Technology Symposium (QIT54) (May 27–29, 2026), https://www.ieice.org/es/qit/qit54/index_e.html
2. Barenco, A., Berthiaume, A., Deutsch, D., Ekert, A., Jozsa, R., Macchiavello, C.: Stabilization of quantum computations by symmetrization. SIAM Journal on Computing **26**(5), 1541–1557 (1997). https://doi.org/10.1137/S0097539796302452
3. Blum, M., Micali, S.: How to generate cryptographically strong sequences of pseudo-random bits. SIAM Journal on Computing **13**(4), 850–864 (1984). https://doi.org/10.1137/0213053
4. Buhrman, H., Cleve, R., Watrous, J., Wolf, R.D.: Quantum fingerprinting. Physical Review Letters **87**(16), 167902 (2001)
5. Cohen, H.: A Course in Computational Algebraic Number Theory, Graduate Texts in Mathematics, vol. 138. Springer (1993)
6. Menezes, A., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press (1996). https://doi.org/10.1201/9781439821916
7. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Journal on Computing **26**(5), 1484–1509 (1997). https://doi.org/10.1137/S0097539795293172

A How to Produce $|\phi\rangle$ and $|\psi\rangle$

Let n_{hf} and n_p be defined as follows:

$$n_{\mathrm{hf}} = \left\lfloor \log_2 \left(\frac{p-1}{2} \right) \right\rfloor + 1,$$
$$n_p = \lfloor \log_2 p \rfloor + 1.$$

Let H be the Hadamard transform on the first n_{hf}-qubit register, and let I denote the identity operator on the second n_p-qubit register. The target state $|\phi\rangle$ is obtained by the following procedure:

$$\begin{aligned}
|0\rangle |1\rangle &\xrightarrow{H\otimes I} \frac{1}{\sqrt{\frac{p-1}{2}}} \sum_{k=0}^{(p-3)/2} |k\rangle |1\rangle \\
&\xrightarrow{U_{g,p}} \frac{1}{\sqrt{\frac{p-1}{2}}} \sum_{k=0}^{(p-3)/2} |k\rangle |g^k \bmod p\rangle \\
&\xrightarrow{H^{-1}\otimes I} |0\rangle \frac{1}{\sqrt{\frac{p-1}{2}}} \sum_{k=0}^{(p-3)/2} |g^k \bmod p\rangle = |0\rangle |\phi\rangle ,
\end{aligned}$$

where $H^{-1} = H$ because the Hadamard transform is self-inverse. Since the state $|0\rangle$ in the first register does not affect any subsequent computation, the first register can be safely omitted.

The state $|\psi\rangle$ is obtained in a similar manner:

$$|0\rangle\,|u\rangle \longrightarrow |0\rangle\,\frac{1}{\sqrt{p_{\rm qut}}}\sum_{k=0}^{p_{\rm qut}-1}|u\cdot g^k \bmod p\rangle = |0\rangle\,|\psi\rangle\,.$$

As with $|\phi\rangle$, the first register can be safely omitted.

B Example of the Swap Test

Let $p = 29$ and $g = 3$. It follows that $p_{\rm qut} = 7$. Suppose that $y = 23$ at Step 14 of Fig. 2. Note that 23 is a quadratic residue. Since $29 \equiv 5 \pmod 8$ and $23^{(29-1)/4} \equiv 1 \pmod{29}$, we compute a square root u of 23 using Eq. (5), i.e.,

$$\begin{aligned} u &\equiv 23^{\frac{29+3}{8}} \pmod{29} \\ &\equiv 20. \end{aligned}$$

For $u = 20$, the states $|\phi\rangle$ and $|\psi\rangle$ are defined by Eq. (9) and Eq. (10), respectively. They are given by

$$\begin{aligned} |\phi\rangle &= \frac{1}{\sqrt{14}}\sum_{k=0}^{13}|3^k \bmod 29\rangle \\ &= \frac{1}{\sqrt{14}}\big(\,|1\rangle + |3\rangle + |9\rangle + |27\rangle + |23\rangle + |11\rangle + |4\rangle \\ &\quad + |12\rangle + |7\rangle + |21\rangle + |5\rangle + |15\rangle + |16\rangle + |19\rangle\,\big), \end{aligned}$$

and

$$\begin{aligned} |\psi\rangle &= \frac{1}{\sqrt{7}}\sum_{k=0}^{6}|20\cdot 3^k \bmod 29\rangle \\ &= \frac{1}{\sqrt{7}}\big(\,|20\rangle + |2\rangle + |6\rangle + |18\rangle + |25\rangle + |17\rangle + |22\rangle\,\big). \end{aligned}$$

Since $\langle\phi \mid \psi\rangle$ is equal to 0, the probability of observing $|0\rangle_1$ is 1/2. In fact, since $20 \equiv 3^{16} \pmod{29}$ and $16 > (29-1)/2$, 20 is the non-principal square root of 23.

Next, we consider the other square root, $u = 9$. Then, $|\psi\rangle$ is given by

$$\begin{aligned} |\psi\rangle &= \frac{1}{\sqrt{7}}\sum_{k=0}^{6}|9\cdot 3^k \bmod 29\rangle \\ &= \frac{1}{\sqrt{7}}\big(\,|9\rangle + |27\rangle + |23\rangle + |11\rangle + |4\rangle + |12\rangle + |7\rangle\,\big). \end{aligned}$$

Since $\langle \phi \mid \psi \rangle$ is computed as

$$\begin{aligned}\langle \phi \mid \psi \rangle &= \frac{1}{\sqrt{14}} \cdot \frac{1}{\sqrt{7}} \cdot \big(\langle 9 \mid 9 \rangle + \langle 27 \mid 27 \rangle + \langle 23 \mid 23 \rangle \\ &\quad + \langle 11 \mid 11 \rangle + \langle 4 \mid 4 \rangle + \langle 12 \mid 12 \rangle + \langle 7 \mid 7 \rangle \big) \\ &= \frac{1}{\sqrt{2}},\end{aligned}$$

the probability of observing $|0\rangle_1$ is $3/4$. In fact, 9 is the principal square root of 23 because $9 \equiv 3^2 \pmod{29}$ and $2 < (29-1)/2$.

C Asymptotic Bound on ℓ

We derive an asymptotic estimate for the expression in parentheses in Eq. (18).

$$\begin{aligned}2 - \frac{1}{2^{\frac{1}{n}-1}} &= 2\left(1 - 2^{-\frac{1}{n}}\right) \\ &= 2\left(\frac{\ln 2}{n} + O\left(\frac{1}{n^2}\right)\right). \end{aligned} \tag{19}$$

The last equality follows from the following Maclaurin expansion of $f(x) = 2^x$ around $x = 0$:

$$2^x = \sum_{k=0}^{\infty} \frac{(\ln 2)^k}{k!} x^k.$$

Substituting $x = -1/n$ into this equation yields

$$\begin{aligned}2^{-\frac{1}{n}} &= \sum_{k=0}^{\infty} \frac{(\ln 2)^k}{k!} \left(-\frac{1}{n}\right)^k \\ &= 1 - \frac{\ln 2}{n} + O\left(\frac{1}{n^2}\right).\end{aligned}$$

Applying $\log_{0.966}$ to both sides of Eq. (19) gives

$$\begin{aligned}\log_{0.966}\left(2 - \frac{1}{2^{\frac{1}{n}-1}}\right) &= \frac{\ln\left(2\left(\frac{\ln 2}{n} + O\left(\frac{1}{n^2}\right)\right)\right)}{\ln(0.966)} \\ &= \frac{\ln 2 + \ln\left(\frac{\ln 2}{n} + O\left(\frac{1}{n^2}\right)\right)}{\ln(0.966)} \\ &= \frac{\ln 2 + \ln(\ln 2) - \ln n + O\left(\frac{1}{n}\right)}{\ln(0.966)} \\ &= O(\log n).\end{aligned}$$