

Efficient fault-tolerant circuit for preparing quantum uniform superposition states via quantum measurement

Xiang-Qun Fu(付向群), Tian-Ci Tian(田天赐), Hong-Wei Li(李宏伟), Jian-Hong Shi(史建红), Xiao-Liang Yang(杨晓亮), Tan Li(李坦), and Wan-Su Bao(鲍皖苏)

Citation: Chin. Phys. B, 2025, 34 (**12**): 120303. DOI: 10.1088/1674-1056/addcc2
http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

Effect of quantum measurement errors on witnessing network topology

Shu-Yuan Yang(杨舒媛), Kan He(贺衍), and Ming-Xing Luo(罗明星)
Chin. Phys. B, 2025, 34 (**9**): 090302. DOI: 10.1088/1674-1056/ade3af

New construction of mutually unbiased bases for odd-dimensional state space

Chenghong Wang(王成红), Kun Wang(王昆), and Zhu-Jun Zheng(郑驻军)
Chin. Phys. B, 2024, 33 (**8**): 080304. DOI: 10.1088/1674-1056/ad47ae

Design of a novel hybrid quantum deep neural network in INEQR images classification

Shuang Wang(王爽), Ke-Han Wang(王柯涵), Tao Cheng(程涛), Run-Sheng Zhao(赵润盛), Hong-Yang Ma(马鸿洋), and Shuai Guo(郭帅)
Chin. Phys. B, 2024, 33 (**6**): 060310. DOI: 10.1088/1674-1056/ad342e

Double quantum images encryption scheme based on chaotic system

She-Xiang Jiang(蒋社想), Yang Li(李杨), Jin Shi(石锦), and Ru Zhang(张茹)
Chin. Phys. B, 2024, 33 (**4**): 040306. DOI: 10.1088/1674-1056/ad1174

Sharing quantum nonlocality in the noisy scenario

Shu-Yuan Yang(杨舒媛), Jin-Chuan Hou(侯晋川), and Kan He(贺衍)
Chin. Phys. B, 2024, 33 (**1**): 010302. DOI: 10.1088/1674-1056/ad062d

---

Efficient fault-tolerant circuit for preparing quantum uniform superposition states via quantum measurement

Xiang-Qun Fu(付向群), Tian-Ci Tian(田天赐), Hong-Wei Li(李宏伟), Jian-Hong Shi(史建红), Xiao-Liang Yang(杨晓亮), Tan Li(李坦)†, and Wan-Su Bao(鲍皖苏)‡

Henan Key Laboratory of Quantum Information and Cryptography, Information Engineering University, Zhengzhou 450004, *China*

(Received 10 April 2025; revised manuscript received 22 May 2025; accepted manuscript online 23 May 2025)

Preparing quantum superposition states is a crucial step in realizing quantum algorithms, which demands substantial resources. In this paper, we propose a new method for preparing quantum uniform superposition states via quantum measurement, and design the bitwise implementation circuit, which only contains Hadamard, CNOT, and $\pi/8$ phase gates. Compared to the Shukla–Vedula method, the number of quantum gates required by both methods scales the same, while, the new method offers stronger fault tolerance, and the ancillary qubits employed during the implementation process can be reused, making it more suitable for implementation on real quantum computers. As an application, we provide the circuit for Shor's discrete logarithm quantum algorithm, based on the new method, demonstrating its technical advantage for implementation of quantum algorithms.

Keywords: quantum superposition state, quantum measurement, quantum computing algorithm, quantum circuit

PACS: 03.67.Lx, 03.65.Sq, 03.65.Ta, 03.67.–a

DOI: 10.1088/1674-1056/addcc2 **CSTR:** 32038.14.CPB.addcc2

1. Introduction

Since the concept of quantum computing was proposed, researchers have been persistently exploring methods to harness its exceptional power. In 1985, Deutsch proposed a quantum computing algorithm to assess the balance of Boolean functions,[1] which demonstrated the advantages of quantum computing for the first time. Subsequently, Shor's algorithm,[2] and Grover's algorithm[3,4] were successively proposed. Shor's algorithm is capable of solving large integer factorization and discrete logarithm problems in polynomial time, thus posing a severe threat to the security of RSA and ECC. Grover's algorithm is a quantum search algorithm, which offers a quadratic speedup compared to classical search. The emergence of these two algorithms significantly accelerated the development of quantum computing and the implementation of quantum algorithms.[5–9] To realize large-scale quantum computing algorithms, researchers strive to enhance quantum computers' qubit capacity and reduce the required qubits of algorithms. Especially, in recent years, the development of quantum computers has been particularly rapid, for example, superconducting, ion trap, linear optical, and cold atom quantum computers. In 2023, Tsinghua University achieved a two-dimensional array of about 500 ion; the University of Science and Technology of China built the Jiuzhang 3.0 quantum computer with 255 photons; the American atomic computing company "Atom Computing" released a cold atom quantum computer with 1000 qubits; Harvard University and the Massachusetts Institute of Technology jointly developed a cold atom quantum computing prototype that can construct 48 logical qubits based on 280 physical qubits. In 2024, Google released its newest quantum computer Sycamore 3 with improved stability and performance; and IBM unveiled Heron quantum computer with 156 qubits. Although the number of qubits in quantum computers has exceeded several hundred, it still falls short of the requirements for implementing large-scale quantum computing algorithms. Therefore, while developing quantum computers, researchers are also focused on reducing the number of qubits required to implement quantum computing algorithms.

In fact, quantum computing algorithms are composed of a series of quantum transformations. As long as these transformations are realized, the implementation of quantum computing algorithms becomes feasible. Preparing quantum superposition states is a crucial step in realizing quantum algorithms, which demands substantial resources. The depth of the realization circuit can reach an exponential level $O(2^n/n)$ without the condition of auxiliary qubits.[10] When the circuit depth reaches $O(n)$, the scale of auxiliary qubits required is $O(2^n)$.[11] Thus, the preparation of quantum superposition states directly determines whether the algorithm can be effectively implemented, such as the HHL algorithm and Hamiltonian evolution. While, preparing quantum superposition states with certain properties requires fewer quantum

†Corresponding author. E-mail: litanzh@sina.com

‡Corresponding author. E-mail: bws@qiclab.cn

© 2025 Chinese Physical Society and IOP Publishing Ltd. All rights, including for text and data mining, AI training, and similar technologies, are reserved.

resources.[11–22] For example, preparing a k-sparse quantum superposition state with only k non-zero probability amplitudes, requires a circuit depth of $\log(nk)$ and $O(nk\log k)$ auxiliary qubits.[11] In 2019, the difficulty of preparing quantum state is given.[23] And the low-depth quantum state preparation is presented, which needs exponential scale of auxiliary qubits.[24] However, the quantum uniform superposition state $\frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle$ can be easily realized based on the quantum Fourier transform (QFT) over Z_N.[8,12]

The QFT, a widely used quantum transformation, has been employed both to prepare quantum uniform superposition state, and to enhance the probability of the target quantum state after measurement. The implementation methods of QFT have consistently attracted significant attention from researchers. When $N=2^n$, QFT is relatively straightforward to implement. It can be employed for preparing uniform superposition states $\frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle$ using n qubits and n one-qubit gates in the full quantum manner,[12] and only 1 qubit and 1 one-qubit gate in the semi-classical manner.[25] How to implement the QFT on quantum computing physical system is the research hotspot.[26–29] But when N is not a power of 2, the implementation of QFT becomes different and difficult. Before 2014, the methods for implementing QFT were either not universal or approximated.[30–33] In 2014, the authors in Ref. [34] presented a method for precisely implementing QFT, whose circuit requires $2n+1$ qubits, $2n$ two-qubit gates, and n^2 three-qubit gates. In 2024, the Shukla–Vedula method for preparing any quantum uniform superposition state was presented.[35] This method only requires n qubits and $O(n)$ basic quantum gates, mainly including the quantum NOT gate, Hadamard gate and rotation around the Y-axis gate

$$R_Y(\theta)=\begin{bmatrix}\cos\frac{\theta}{2} & -\sin\frac{\theta}{2}\\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2}\end{bmatrix}.$$

However, the fault tolerance of current quantum computers is limited. According to the Solovay–Kitaev theorem, the circuit can be designed into an efficient fault-tolerant form using only Hadamard, controlled-NOT (CNOT), and $\pi/8$ gates.[36] Thus, to enhance the precision of preparing quantum uniform superposition states, further research on circuits with higher fault-tolerance is needed, which will also provide technical support for the implementation of quantum algorithms such as Shor's discrete logarithm algorithm and the Proos–Zalka algorithm.

2. Quantum Fourier transform

Definition 1 (Quantum Fourier transform, QFT):[12] QFT on a standard orthonormal basis $|0\rangle,|1\rangle,\ldots,|N-1\rangle$ is defined to be a linear operator with the following action on the basis states,

$$F:|j\rangle\mapsto\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}\omega_N^{jk}|k\rangle, \tag{1}$$

where "$\mapsto$" indicates that the transformation is reversible and $\omega_N=\mathrm{e}^{2\pi\mathrm{i}/N}$. And it is also called quantum Fourier transform over Z_N. And the action on an arbitrary state can be written

$$F:\sum_{j=0}^{N-1}x_j|j\rangle\mapsto\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}\sum_{j=0}^{N-1}x_j\omega_N^{jk}|k\rangle. \tag{2}$$

If $N=2^n$, the action of QFT on $|j\rangle$ can be expressed as following product representation:

$$|j\rangle=|j_1,\ldots,j_n\rangle\mapsto\frac{(|0\rangle+\mathrm{e}^{2\pi\mathrm{i}0.j_n}|1\rangle)(|0\rangle+\mathrm{e}^{2\pi\mathrm{i}0.j_{n-1}j_n}|1\rangle)\cdots(|0\rangle+\mathrm{e}^{2\pi\mathrm{i}0.j_1J_2,\ldots,j_n}|1\rangle)}{2^{n/2}}, \tag{3}$$

where $(j_1,\ldots,j_n)$ is the binary representation of j, $j=j_1 2^{n-1}+\cdots+j_n 2^0$ and $0.j_l j_{l+1}\ldots j_m=j_l/2+j_{l+1}/4+\cdots+j_m/2^{m-l+1}$.

Based on the above product representation, the QFT over Z_{2^n} can be easily implemented using the circuit shown in Fig. 1.[12] In Fig. 1, H is the Hadamard gate and $R_k=\begin{bmatrix}1 & 0\\ 0 & \mathrm{e}^{2\pi\mathrm{i}/2^k}\end{bmatrix}$. And for simplicity, quantum swap gates at the end of the circuit and the normalization factor $1/\sqrt{2}$ in the output are not shown. Therefore, this circuit for QFT over Z_{2^n} requires n qubits, n one-qubit gates, and $n(n-1)/2$ two-qubit gates, achieving a depth of $2n-1$ without ancillary qubits. The depth of QFT over Z_{2^n} is 4 when $n=2$, and $(n^2-n+4)/2$ when $n>2$.

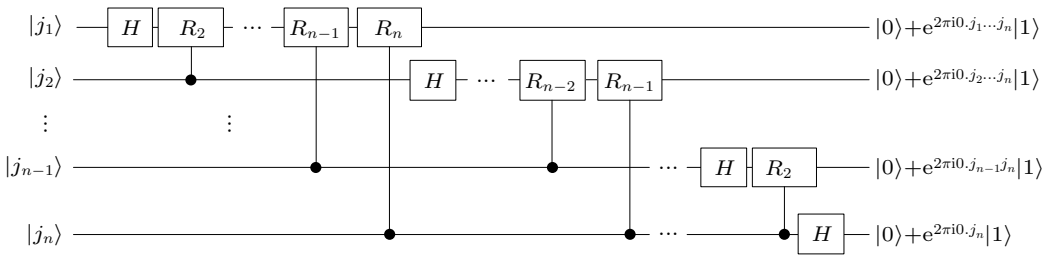


Fig. 1. The circuit for the QFT over Z_{2^n}.

3. Method for preparing quantum uniform superposition state

When $N = 2^n$, $\frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle$ can be obtained by applying $H^{\otimes n}$ on $|0\rangle^{\otimes n}$. While, for arbitrary N (not the power of 2), preparing such a state becomes challenging. The Shukla–Vedula method for preparing any quantum uniform superposition state mainly contains the Y-axis rotation gate $R_Y(\theta)$. And $R_Y(\theta)$ can be compiled into an efficient fault-tolerant circuit with arbitrary precision ε, which requires $O(\log^{3.97}(1/\varepsilon))$ elementary quantum gates.[36] And the elementary quantum gate contains Hadamard, CNOT, and $\pi/8$ phase gates. If the Shukla–Vedula method is compiled into an efficient fault-tolerant circuit, it will require more than $O(n)$ quantum gates. Thus, designing an efficient fault-tolerant circuit for preparing quantum uniform superposition states with fewer quantum resources needs further study. A new method for preparing quantum uniform superposition states is designed as follows. Assume $2^{n-1} < N < 2^n$, and let $n = \lceil \log_2 N \rceil$.

3.1. New method for preparing quantum uniform superposition states via quantum measurment

Step 1 Initialize two quantum registers $|0\rangle^{\otimes n}|0\rangle$.

Step 2 Apply the Hadamard transform $H^{\otimes n}$ on the first quantum register, yielding the initial quantum state

$$\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle|0\rangle = (\sin\theta|\alpha\rangle + \cos\theta|\beta\rangle)|0\rangle, \tag{4}$$

where $|\alpha\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle^{\otimes n}$, $|\beta\rangle = \frac{1}{\sqrt{2^n-N}}\sum_{x=N}^{2^n-1}|x\rangle^{\otimes n}$, and $\theta = \arcsin\sqrt{N/2^n}$.

Step 3 Apply the quantum Oracle function $f(x,N) = \begin{cases} 1, & 0 \le x < N, \\ 0, & N \le x < 2^n, \end{cases}$ and store the corresponding function evaluation in the second register, yielding the state to become

$$\sin\theta|\alpha\rangle|1\rangle + \cos\theta|\beta\rangle|0\rangle.$$

Step 4 Measure the second register. If the outcome is 1, then the first register collapses to the desired superposition $|\alpha\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle^{\otimes n}$, otherwise the algorithm terminates unsuccessfully.

3.2. Quantum circuit for new method

The $H^{\otimes n}$ in Step 2 is easily implemented, and the quantum measurement in Step 4 is also straightforward. However, implementing the quantum Oracle function $f(x,N)$ in Step 3 poses a significant challenge. Thus, the key to implement the entire algorithm lies in successfully implementing this Oracle function $f(x,N)$.

Theorem 1 Let $(x_1,x_2,\ldots,x_n)$ and $(k_1,k_2,\ldots,k_n)$ are the binary representation of x and k, and $e_i = E(x_i,k_i) = \overline{x_i \oplus k_i}$, $s_i = S(x_i,k_i) = \overline{x_i}k_i$ for $i = 1,2,\ldots,n$, then

$$\begin{aligned} f(x,k) &= s_1 \oplus e_1 s_2 \oplus e_1 e_2 s_3 \oplus \cdots \oplus e_1 \cdots e_{n-1} s_n \\ &= \begin{cases} 1, & 0 \le x < k, \\ 0, & k \le x < 2^n, \end{cases} \end{aligned} \tag{5}$$

where $^-$ and $\oplus$ represent NOT and modulo 2 addition operations, respectively.

Proof When $x = k$, $x_i = k_i$, then $e_i = 1$ and $s_i = 0$. Thus

$$s_1 \oplus e_1 s_2 \oplus e_1 e_2 s_3 \oplus \cdots \oplus e_1 \cdots e_{n-1} s_n = 0 = f(x,k).$$

When $x < k$, there are two scenarios:

(i) If $x_1 < k_1$, i.e., $x_1 = 0$ and $k_1 = 1$, then $s_1 = 1$ and $e_1 = 0$. Thus,

$$s_1 \oplus e_1 s_2 \oplus e_1 e_2 s_3 \oplus \cdots \oplus e_1 \cdots e_{n-1} s_n = 1 = f(x,k).$$

(ii) If $x_1 = k_1$, then there must exist j such that $x_1 = k_1,\ldots,x_j = k_j$ and $x_{j+1} < k_{j+1}$, where $1 \le j \le n-1$. Furthermore, we can derive $e_1 = \cdots = e_j = 1$, $s_1 = \cdots = s_j = 0$, $e_{j+1} = 0$, and $s_{j+1} = 1$. Thus,

$$s_1 \oplus e_1 s_2 \oplus e_1 e_2 s_3 \oplus \cdots \oplus e_1 \cdots e_{n-1} s_n = 1 = f(x,k).$$

When $x > k$, there are also two scenarios:

i) If $x_1 > k_1$, i.e., $x_1 = 1$ and $k_1 = 0$, then $s_1 = 0$ and $e_1 = 0$. Thus,

$$s_1 \oplus e_1 s_2 \oplus e_1 e_2 s_3 \oplus \cdots \oplus e_1 \cdots e_{n-1} s_n = 0 = f(x,k).$$

ii) If $x_1 = k_1$, then there must exist j such that $x_1 = k_1,\ldots,x_j = k_j$ and $x_{j+1} > k_{j+1}$, where$1 \le j \le n-1$. Furthermore, we can derive $e_1 = \cdots = e_j = 1$, $s_1 = \cdots = s_j = s_{j+1} = 0$, and$e_{j+1} = 0$. Thus,

$$s_1 \oplus e_1 s_2 \oplus e_1 e_2 s_3 \oplus \cdots \oplus e_1 \cdots e_{n-1} s_n = 0 = f(x,k).$$

This completes the proof of Theorem 1.

In fact, Theorem 1 allows the Oracle function f to be implemented bit-by-bit, which is conducive to the design of its quantum circuit. Specifically, in $s_1 \oplus e_1 s_2 \oplus e_1 e_2 s_3 \oplus \cdots \oplus e_1 \cdots e_{n-1} s_n$, we should focus on the quantum circuit for e_i and s_i, namely $E(y,l) = \overline{y \oplus l}$ and $S(y,l) = \bar{y}l$, which are shown in Figs. 2 and 3, respectively.

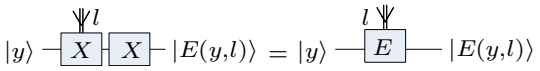


Fig. 2. The quantum circuit for $E(y,l) = \overline{y \oplus l}$.

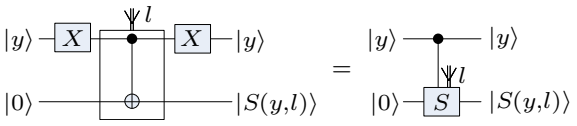
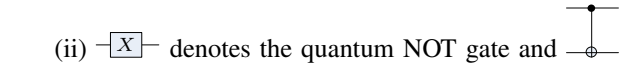
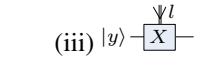


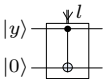
Fig. 3. The quantum circuit for $S(y,l) = \bar{y}l$.

The descriptions of Figs. 2 and 3 are as follows.

(i) Single line (−) represents the quantum signal, and double line (=) represents the classical signal.

(ii)  denotes the quantum NOT gate and denotes the quantum CNOT gate.

(iii)  is the quantum NOT gate controlled by one classical bit l. If $l=0$, the quantum NOT gate is ineffective and outputs $|y\rangle$, otherwise, the gate is effective and outputs $|\bar{y}\rangle$. Note that, this gate is one-qubit quantum gate with classical control.

(iv)  is quantum CNOT gate controlled by one classical bit l. If $l=0$, the quantum gate is ineffective and outputs $|y\rangle|h\rangle$, otherwise, is effective and outputs $|y\rangle|y\oplus h\rangle$. Note that, this gate is two-qubit quantum gate with classical control.

(v) The implementation circuit for $E(y,l)=\overline{y\oplus l}$ requires 2 one-qubit gates, and the circuit for $S(y,l)=\bar{y}l$ requires 2 one-qubit gate and 1 two-qubit gate.

Based on the circuits for $E(y,l)=\overline{y\oplus l}$ and $\mathrm{S}(y,l)=\bar{y}l$, the circuit for the quantum Oracle function f can be designed as shown in Fig. 4.

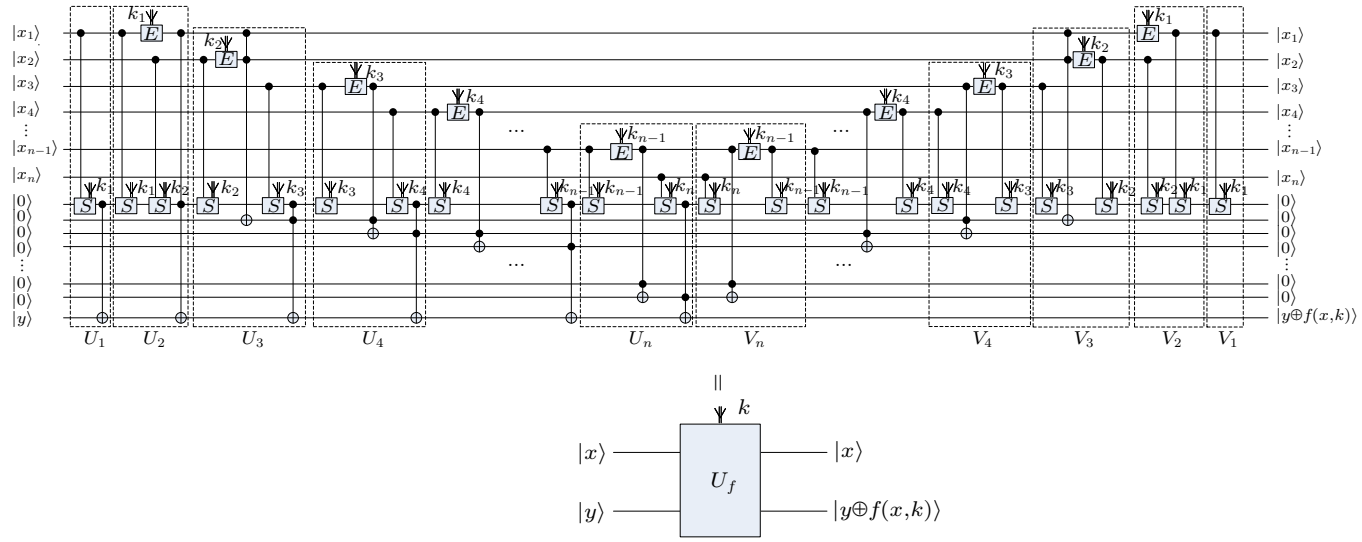


Fig. 4. Quantum circuit for the Oracle function $f(x,k)$.

The description of Fig. 4 is as follows.

(I) $(x_1,x_2,\ldots,x_n)$ and $(k_1,k_2,\ldots,k_n)$ are the binary representation of x and k, respectively.

(II) When $n=1$, the implementation circuit for $f(x,k)$ requires 3 qubits (including 1 ancillary qubit), 2 one-qubit gates, 2 two-qubit gates, and the circuit depth is 3.

(III) When $n>1$, the circuit requires $2n$ qubits (including $n-1$ ancillary qubits), $12n-8$ one-qubit gates, $4n-1$ two-qubit gates, $3n-5$ three-qubit Toffoli gates, and the circuit depth is $19n-18$.

Note: When n and k are known, the circuit for Oracle function f can be directly drawn using the Python Qiskit library. The specific program code is shown in the Appendix.

The detailed depth analysis is shown as follow.

The depth of E and S are respectively 2 and 3. Thus, the depth of U_1 and U_2 are respectively 4 and 7. The depth of $U_3,\ldots,U_n$ are all 10. And the depth of V_i is one less than U_i $(i=1,2,\ldots,n)$. Thus, the depth of Quantum circuit for f is $19n-18$.

Then, based on the circuit for the Oracle function f, the circuit for the new method presented in this Section to prepare quantum uniform superposition states can be designed as shown in Fig. 5.

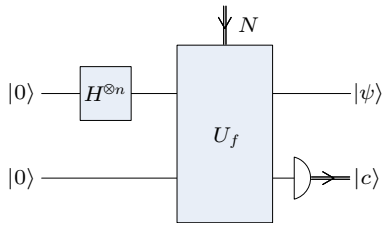


Fig. 5. The circuit for preparing quantum uniform superposition states via quantum measurement.

The description of Fig. 5 is as follows.

(i) The measurement result of c is either 0 or 1. H is Hadamard gate.

(ii) $|\psi\rangle=\frac{c}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle+\frac{1-c}{\sqrt{2^n-N}}\sum_{x=N}^{2^n-1}|x\rangle$.

3.3. Algorithm analysis and comparison

In Step 3, after the Oracle transformation, the resulting quantum superposition state is

$$\sin\theta\,|\alpha\rangle^{\otimes n}|0\oplus f(x)\rangle^{\otimes 1}+\cos\theta\,|\beta\rangle^{\otimes n}|0\oplus f(x)\rangle^{\otimes 1}$$

$$= \sin\theta\,|\alpha\rangle^{\otimes n}|1\rangle^{\otimes 1} + \cos\theta\,|\beta\rangle^{\otimes n}|0\rangle^{\otimes 1}.$$

Thus, in Step 4, the measurement result is 1 with a probability $|\sin\theta|^2$. At this time, the first quantum register collapses into the quantum superposition state $|\alpha\rangle$. Since $2^{n-1} < N < 2^n$, we have $|\sin\theta|^2 > 1/2$, therefore this new method can prepare the quantum uniform superposition state $\frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle$ with a probability greater than $1/2$.

Compared to Fig. 4, the number of additional ancillary qubits remains unchanged, only n Hadamard gates are required in Fig. 5. Thus, when $n=1$, the implementation circuit for the entire algorithm requires 3 qubits (including 1 ancillary qubit), 3 one-qubit gates, 2 two-qubit gates, and the circuit depth is 4. When $n>1$, the circuit requires $2n$ qubits (including $n-1$ ancillary qubits), $13n-8$ one-qubit gates, $4n-1$ two-qubit gates, $3n-5$ three-qubit Toffoli gates, and the circuit depth is $9n-9$. In conclusion, the total computational complexity of the new method is $O(n)$.

Figure 5 includes only quantum NOT, Hadamard, CNOT, and Toffoli gates, in which the circuit for Toffoli gate only requires 2 Hadamard gates, 9 $\pi/8$ phase gates, and 6 CNOT gates.[12] Thus, in Fig. 5, the number of one-qubit gates and two-qubit gates both scale as $O(n)$. Additionally, an NOT gate can be implemented using a CNOT gate. Thus, according to the Solovay–Kitaev theorem, figure 5 is an efficient fault-tolerant circuit.

Compared to the Shukla–Vedula method, the number of quantum gates required by both methods scales as $O(n)$. While, the new method requires n additional qubits. But it has stronger fault tolerance and is more suitable for implementation on the real quantum computers. In fact, the Y-axis rotation gate $R_Y(\theta)$ is not an efficient fault-tolerant gate. When the Shukla–Vedula method is compiled into an efficient fault-tolerant circuit with arbitrary precision ε, it requires $O(n\log^{3.97}(1/\varepsilon))$ additional elementary quantum gates. If the required ε is very low, then the Shukla–Vedula method requires more quantum gates than the new method.

Furthermore, the n ancillary qubits in Fig. 5 can be reused. For example, in Shor's discrete logarithm quantum algorithm, the ancillary qubits used to prepare the uniform superposition state can also be employed for modular exponentiation. This means that the number of qubits required for implementing Shor's algorithm does not increase, and it remains the same as the number of qubits needed when implementing Shor's algorithm based on the Shukla–Vedula method.

To make it clear, preparing $\frac{1}{\sqrt{3}}\sum_{x=0}^{2}|x\rangle$ based on the new method is shown as follows:

Step 1 Initialize two quantum registers $|0\rangle|0\rangle$.

Step 2 Apply the Hadamard transform $H^{\otimes 2}$ on the first quantum register, yielding the initial quantum state

$$\frac{1}{2}(|0\rangle+|1\rangle+|2\rangle+|3\rangle)|0\rangle = \left(\frac{\sqrt{3}}{2}|\alpha\rangle + \frac{1}{2}|\beta\rangle\right)|0\rangle,$$

where $|\alpha\rangle = \frac{1}{\sqrt{3}}\sum_{x=0}^{2}|x\rangle$, $|\beta\rangle = |3\rangle$.

Step 3 Apply the quantum Oracle function $f(x,3) = \begin{cases} 1, & 0 \le x < 3, \\ 0, & x = 3, \end{cases}$ and store the corresponding function evaluation in the second register, yielding the state to become

$$\frac{\sqrt{3}}{2}|\alpha\rangle|1\rangle + \frac{1}{2}|\beta\rangle|0\rangle.$$

Step 4 Measure the second register. If the outcome is 1, then the first register collapses to the desired superposition $|\alpha\rangle = \frac{1}{\sqrt{3}}\sum_{x=0}^{2}|x\rangle$, otherwise the algorithm terminates unsuccessfully.

4. Implementing Shor's discrete logarithm quantum algorithm

Given a prime number p, the multiplicative group $(Z/pZ)^*$ is a cyclic group, and there exists a generator g. For a given element x, the problem of finding an integer t such that $x = g^t \bmod p$ is known as the discrete logarithm problem.

In 1994, Shor introduced a quantum algorithm for solving the discrete logarithm problem with polynomial computation complexity.[2] The specific steps of the algorithm are as follows:

Step 1 Initializing the three quantum registers $|0\rangle^{\otimes n}|0\rangle^{\otimes n}|0\rangle^{\otimes n}$, where $p < 2^n < 2p$.

Step 2 Preparing the quantum uniform superposition states in the first two registers, which yields

$$\frac{1}{p-1}\sum_{a=0}^{p-2}\sum_{b=0}^{p-2}|a\rangle^{\otimes n}|b\rangle^{\otimes n}|0\rangle^{\otimes n}. \tag{6}$$

Step 3 Applying the Oracle function $g^a x^{-b} (\bmod p)$, and store the results in the third quantum register, and obtain

$$\frac{1}{p-1}\sum_{a=0}^{p-2}\sum_{b=0}^{p-2}|a\rangle^{\otimes n}|b\rangle^{\otimes n}\left|g^a x^{-b} (\bmod p)\right\rangle^{\otimes n}. \tag{7}$$

Step 4 Applying the QFT over Z_{2^n} on the first two quantum registers, respectively, and the state becomes

$$\frac{1}{2^n(p-1)}\sum_{a,b=0}^{p-2}\sum_{\chi,\delta=0}^{2^n-1} e^{2\pi i(a\chi+b\delta)/2^n}|\chi\rangle^{\otimes n}|\delta\rangle^{\otimes n} \times |g^a x^{-b} (\bmod p)\rangle^{\otimes n}. \tag{8}$$

Step 5 Measuring the third register, and the state of registers collapses to $|\chi,\delta,y\rangle$.

In fact, Step 2 of Shor's discrete logarithm quantum algorithm can be implemented using the circuit of Fig. 5, and the specific circuit is depicted in Fig. 6.

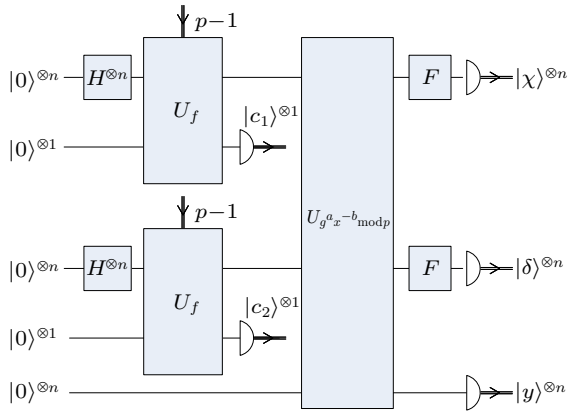


Fig. 6. The circuit for Shor's discrete logarithm quantum algorithm.

The description of Fig. 6 is as follows.

(i) $U_{g^a x^{-b} \bmod p}$ is the circuit for $g^a x^{-b} (\mathrm{mod}\, p)$, which can be vided in Refs. [37,38].

(ii) F represents the circuit for QFT over Z_{2^n}.

(iii) If $c_1 = c_2 = 1$, then the measurement results of χ, δ, y can be used to solve the discrete logarithm problem; otherwise, the circuit needs to be rerun. And the probability of $c_1 = c_2 = 1$ is greater than $1/4$.

There are mainly two methods for implementing modular exponentiation $g^a x^{-b} (\mathrm{mod}\, p)$: one is based on basic quantum gates,[37] and the other utilizes Draper's adders with the QFT.[38] The former requires $7n+1$ qubits and has a circuit depth of $400n^3 - 400n^2 + 75n$, while the latter requires $9n+2$ qubits and has a circuit depth of $2000n^2$. Each method has its own advantages and disadvantages. If sufficient qubits are available, it is preferable to use Draper's adder based on the QFT; otherwise, the method based on basic quantum gates should be used. Thus, the quantum resources required for Shor's discrete logarithm quantum algorithm are as follows:

(I) **Based on basic quantum gates**

When $n = 2$, the circuit requires $7n+1$ qubits with a circuit depth of $400n^3 - 400n^2 + 94n - 13$. When $n > 2$, the circuit requires $7n+1$ qubits with a circuit depth of $400n^3 - 399.5n^2 + 93n - 13$.

(II) **Based on the QFT Draper's adders**

When $n = 2$, the circuit requires $9n+2$ qubits with a circuit depth of $2000n^2 + 19n - 17$. When $n > 2$, the circuit requires $9n+2$ qubits with a circuit depth of $2000.5n^2 + 18n - 17$.

Note: The ancillary qubits for preparing quantum uniform superposition states can be reused for implementing the modular exponentiation $g^a x^{-b} (\mathrm{mod}\, p)$.

5. Conclusions

In this paper, a new method for preparing quantum uniform superposition states via quantum measurement is presented, and the bitwise implementation circuit is designed. Compared to the Shukla–Vedula method, the new method has stronger fault tolerance, while the number of quantum gates required by both methods scales the same as $O(n)$. Moreover, the ancillary qubits employed during the implementation process can be reused. Finally, as an application, the circuit for Shor's discrete logarithm quantum algorithm, based on the new method, is provided. Further research could be expanded to develop an efficient fault-tolerant circuit for preparing a quantum uniform superposition state on some arbitrary selected bitstrings, and modular exponentiation which is a crucial component of Shor's algorithm.

Appendix A: Program code for drawing implementation circuit of Oracle function based on Python Qiskit library

```
From qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit, execute, Aer
from qiskit.circuit.library.standard_gates.x import CCXGate, CXGate,XGate
from qiskit.visualization import circuit_drawer
def Efun(xi,k,qc):
    if k==1:
        qc.x(xi)
    qc.x(xi)
def Sfun(xi,zeroi,k,qc):
    qc.x(xi)
    if k == 1:
        qc.append(CXGate(),[xi,zeroi])
    qc.x(xi)
# define Ui i>=4,j=i-1
def Ufun(xi,zeroi,ki,kj,qc):
    Sfun(xi-1,zeroi,kj,qc)
    Efun(xi-1,kj,qc)
    qc.append(CCXGate(),[xi-1,zeroi+xi-2,zeroi+xi-1])
    Sfun(xi,zeroi,ki,qc)
    qc.append(CCXGate(),[zeroi,zeroi+xi-1,2*zeroi-1])
def Vfun(xi,zeroi,ki,kj,qc):
    Sfun(xi,zeroi,ki,qc)
    qc.append(CCXGate(),[xi-1,zeroi+xi-2,zeroi+xi-1])
    Efun(xi-1,kj,qc)
    Sfun(xi-1,zeroi,kj,qc)
# Specify the number of qubits, lenn, and generate the corresponding circuit
num = 4
k = 5
binary_k = bin(k)[2:]
lenn = max(num, len(binary_k))
# binary_x = binary_x.zfill(lenn)
binary_k = binary_k.zfill(lenn)
# x_list = [int(bit) for bit in binary_x]
k_list = [int(bit) for bit in binary_k]
```

```
# if lenn>1:
#       n = 2*lenn
#       ini_str = binary_x+"0"*lenn
# else:
#       n = 3
#       ini_str = binary_x+"0"*2
sim = Aer.get_backend('aer_simulator')
qc = QuantumCircuit(n)
# qc.initialize(ini_str[::-1])
if lenn>0:
    Sfun(0,lenn,k_list[0],qc)
    qc.append(CXGate(),[lenn,n-1])
    qc.barrier()
    f lenn>1:
        Sfun(0,lenn,k_list[0],qc)
        Efun(0,k_list[0],qc)
        Sfun(1,lenn,k_list[1],qc)
        qc.append(CCXGate(),[0,lenn,n-1])
        qc.barrier()
        if lenn>2:
            Sfun(1,lenn,k_list[1],qc)
            Efun(1,k_list[1],qc)
            qc.append(CCXGate(),[0,1,lenn+1])
            Sfun(2,lenn,k_list[2],qc)
            qc.append(CCXGate(),[lenn,lenn+1,n-1])
            qc.barrier()
            if lenn>3:
                for i in range(3,lenn):
                    Ufun(i,lenn,k_list[i],k_list[i-1],qc)
                    qc.barrier()
                for i in range(lenn-1,2,-1):
                    Vfun(i,lenn,k_list[i],k_list[i-1],qc)
                    qc.barrier()
            Sfun(2,lenn,k_list[2],qc)
            qc.append(CCXGate(),[0,1,lenn+1])
            Efun(1,k_list[1],qc)
            Sfun(1,lenn,k_list[1],qc)
            qc.barrier()
        Sfun(1,lenn,k_list[1],qc)
        Efun(0,k_list[0],qc)
        Sfun(0,lenn,k_list[0],qc)
        qc.barrier()
    Sfun(0,lenn,k_list[0],qc)
print(qc)
```

Acknowledgements

Project supported by National Key Research and Development Program of China (Grant No. 2020YFA0309702), the National Natural Science Foundation of China (Grant No. 61502526), NSAF (Grant No. U2130205), and the Natural Science Foundation of Henan Province, China (Grant Nos. 202300410532 and 252300421818).

References

[1] Deutsch D 1985 *Proc. R. Soc. Lond. A* **400** 97
[2] Shor P W 1997 *SIAM J. Comput.* **26** 1484
[3] Grover L K 1996 *Proceedings of the 28th ACM Symposium on Theory of Computation*, May 22–24, 1996, Philadelphia Pennsylvania, USA, pp. 212–219
[4] Li T, Zhang S, Fu X Q, Wang X, Wang Y, Lin J and Bao W S 2019 *Chin. Phys. B* **28** 120301
[5] Bhattacharya N, van Linden van den Heuvell H B and Spreeuw R J C 2002 *Phys. Rev. Lett.* **88** 137901
[6] Zhang H Y, Wang S X, Liu X J, Shen Y T and Wang Y K 2024 *Chin. Phys. B* **33** 020310
[7] Pan S J, Wan L C, Liu H L, Wu Y S, Qin S J, Wen Q Y and Gao F 2022 *Chin. Phys. B* **31** 060304
[8] Liu W W, Li H C and Yang R C 2009 *Chin. Phys. B* **18** 23
[9] Pokharel B and Lidar D A 2024 *NPJ Quantum Inf.* **10** 23
[10] Sun X M, Tian G J, Yang S, Yuan P and Zhang S Y 2023 *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **42** 3301
[11] Zhang X M, Li T Y and Yuan X 2022 *Phys. Rev. Lett.* **129** 230504
[12] Nielsen M A and Chuang I L 2010 *Quantum Computation and Quantum Information*, 2nd edn. (Cambridge: Cambridge University Press) pp. 217–219
[13] Zylberman J and Debbasch F 2024 *Phys. Rev. A* **109** 042401
[14] Iaconis J, Johri S and Zhu E Y 2024 *NPJ Quantum Inf.* **10** 15
[15] Lemieux J, Lostaglio M, Pallister S, Pol W, Seetharam K, Sim S and Şahinoğlu B 2024 arXiv: 2405.11436 [quant-ph]
[16] Bausch J 2022 *Quantum* **6** 773
[17] Wang S B, Wang Z M, Cui G L, Shi S S, Shang R M, Fan L X, Li W D, Wei Z Q and Gu Y J 2021 *Quantum Inf. Process.* **20** 270
[18] Sanders Y R, Low G H, Scherer A and Berry D W 2019 *Phys. Rev. Lett.* **122** 020502
[19] McArdle S, Gilyén A and Berta M 2022 arXiv: 2210.14892 [quant-ph]
[20] Rosenkranz M, Brunner E, Marin-Sanchez G, Fitzpatrick N, Dilkes S, Tang Y, Kikuchi Y and Benedetti M 2025 *Quantum* **9** 1703
[21] de Veras T M L, da Silva L D and da Silva A J 2022 *Quantum Inf. Process.* **21** 204
[22] Gleinig N and Hoefler T 2021 *Proceedings of the 58th ACM/IEEE Design Automation Conference (DAC)*, December 5–9, 2021, San Francisco, USA, p. 433–438
[23] Girolami D 2019 *Phys. Rev. Lett.* **122** 010505
[24] Zhang X M, Yung M H and Yuan X 2021 *Phys. Rev. Res.* **3** 043200
[25] Griffiths R B and Niu C S 1996 *Phys. Rev. Lett.* **76** 3228
[26] Scully M and Zubairy M 2002 *Phys. Rev. A* **65** 052324
[27] Dong L, Xiu X M, Shen H Z, Gao Y J and Yi X X 2013 *J. Opt. Soc. Am. B* **30** 2765
[28] Wang H F, Zhang S, Zhu A D and Yeon K H 2012 *J. Opt. Soc. Am. B* **29** 1078
[29] Weinstein Y S, Pravia M A, Fortunato E M, Lloyd S and Cory D G 2001 *Phys. Rev. Lett.* **86** 1889
[30] Kitaev A Y 1995 arXiv: quant-ph/9511026v1 [quant-ph]
[31] Hales L and Hallgren S 2000 *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, November 12–14, 2000, Redondo Beach, USA, pp. 515–525
[32] Proos J and Zalka C 2003 *Quantum Inf. Comput.* **3** 317
[33] Childs A M and Dam W V 2008 arXiv:0812.0380v1 [quant-ph]
[34] Fu X Q, Bao W S, Li F D and Zhang Y C 2014 *Chin. Phys. B* **23** 020306
[35] Shukla A and Vedula P 2024 *Quantum Inf. Process.* **23** 38
[36] Dawson C M and Nielsen M A 2006 *Quantum Inf. Comput.* **6** 81
[37] Meter R V and Itoh K M 2005 *Phys. Rev. A* **71** 052320
[38] Pavlidis A and Gizopoulos D 2014 *Quantum Inf. Comput.* **14** 649