SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

August 2022, Vol. 65 182501:1–182501:14
https://doi.org/10.1007/s11432-021-3334-1

Quantum algorithm and experimental demonstration for the subset sum problem

Qilin ZHENG, Pingyu ZHU, Shichuan XUE, Yang WANG, Chao WU, Xinyao YU, Miaomiao YU, Yingwen LIU, Mingtang DENG, Junjie WU & Ping XU*

Institute for Quantum Information and State Key Laboratory of High Performance Computing, College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, China

Received 24 July 2021/Revised 10 August 2021/Accepted 14 September 2021/Published online 13 December 2021

Abstract To solve the subset sum problem, a well-known nondeterministic polynomial-time complete problem that is widely used in encryption and resource scheduling, we propose a feasible quantum algorithm that utilizes fewer qubits to encode and achieves quadratic speedup. Specifically, this algorithm combines an amplitude amplification algorithm with quantum phase estimation, and requires $n+t+1$ qubits and $O(2^{(0.5+o(1))n})$ operations to obtain the solution, where n is the number of elements, and t is the number of qubits used to store the eigenvalues. To verify the performance of the algorithm, we simulate the algorithm with the online quantum simulator of IBM named ibmq_simulator using Qiskit and then run it on two IBM quantum computers called ibmq_santiago and ibmq_bogota. The experimental results indicate that compared with the brute force algorithm, the proposed algorithm results in quadratic acceleration for the problem of a set S with four elements and two subsets whose sum equals target w. Using the iterator twice, we obtain success probabilities of 0.940 ± 0.004, 0.751 ± 0.040, and 0.665 ± 0.060 on the simulator, ibmq_santiago, and ibmq_bogota, respectively, and the fidelity between the theoretical and experimental quantum states is calculated to be 0.944 ± 0.002, 0.753 ± 0.017, and 0.657 ± 0.028, respectively. If the error rates of the experimental quantum logic gates can be reduced, the success probabilities of the proposed algorithm on real quantum devices can be further improved.

Keywords quantum algorithm, subset sum, quadratic speedup, encryption, algorithm complexity

Citation Zheng Q L, Zhu P Y, Xue S C, et al. Quantum algorithm and experimental demonstration for the subset sum problem. Sci China Inf Sci, 2022, 65(8): 182501, https://doi.org/10.1007/s11432-021-3334-1

1 Introduction

The subset sum problem (SSP) is a typical problem in the nondeterministic polynomial-time complete (NPC) class [1] and is closely related to the knapsack problem [2]. It currently has a wide range of applications in encryption [3–5], resource scheduling and programming [6–8], and graph theory problems [9,10]. Taking a set as an example, the problem can be formally defined as follows: given a random integer set $S = \{s_1, s_2, s_3, \ldots, s_{n-1}, s_n\}$ with n elements and a target integer $w \in (0, \mu]$, where $\mu = \sum_{j=1}^{n} s_j$, the problem is to determine whether there is a sequence x consisting of 0 and 1 that satisfies $\sum_{j=1}^{n} s_j x_j = w$. If one or more sequences that satisfy the condition can be found, then w is the subset sum of set S; otherwise, w is not.

For the SSP, commonly used calculation methods on classical computers include brute force, dynamic programming [1], a greedy algorithm [11], a divide-and-conquer algorithm [12], and optimization [13–15]. In addition to using classical computers to solve this problem, many novel calculation methods can also be used, such as molecular calculations [16–18], DNA or protein calculations [19, 20], soap film calculations [21, 22] (please note that the soap film calculations were proposed by Isenberg [21] and discussed by Aaronson [22]), and photonic calculations [23–26]. Although these algorithms have pseudo-polynomial-time complexity in the best case, when the size of the set elements grows exponentially, the algorithms degenerate to exponential time complexity [12].

* Corresponding author (email: pingxu520@nju.edu.cn)

In terms of the classical complexity of the SSP problem, the brute force algorithm enumerates all the subsets to verify whether there is a solution in time $O(2^n)$. Schroeppel and Shamir [27] improved the time complexity to $O(2^{0.5n})$ with space complexity $O(2^{n/4})$ through the left-right split approach. Thereafter, Becker et al. [28] reduced the runtime to $O(2^{0.291n})$ by improving the representation of the algorithm proposed by Howgrave-Graham and Joux [29], which is currently the best classical time complexity of the SSP. In terms of the quantum complexity of the SSP problem, the authors [30–32] theoretically analyzed the complexity with the quantum walk, and obtained a heuristic asymptotic time complexity of $O(2^{2n/3})$, $O(2^{0.241n})$, and $O(2^{0.226n})$, respectively. However, these algorithms have exponential space complexity and use a quantum memory model with quantum random access to maintain a merging tree, which is currently difficult to implement on noisy intermediate-scale quantum systems. Therefore, none of these three algorithms has been experimentally demonstrated and verified.

In [33], the authors used a classical and quantum hybrid algorithm to analyze the SSP, and obtained a runtime of $O(2^{0.218n})$. Their primary strategy was a merge-and-filter operation that required the help of classical memory. Chang et al. [34] designed a quantum adder network and used the Grover algorithm to solve the SSP with a runtime of $O(2^{0.5n})$. Their algorithm required a total of $2nt_1 + n + 3t_1 + 5$ qubits because of the classic encoding strategy, where t_1 is the number of classical binary bits used to encode all elements in S. In their experiment, they verified the example of $S = \{1\}$ and $w = 1$ on a nuclear magnetic resonance machine; however, they did not use the Grover algorithm to accelerate the search for solutions. Daskin [35] converted the SSP into an optimization problem and applied quantum amplitude amplification (AA) and quantum counting algorithms to select the maximum element less than w. However, their algorithm could not determine whether there was one subset sum equal to w and needed to select the upper limit of the number of measurements based on experience; however, their strategy of coding with the quantum phase estimation (QPE) algorithm was valuable.

Table 1 compares the classical and quantum SSP algorithms with our proposed algorithm in terms of runtime and memory consumption. The classical algorithm proposed in [28] has a runtime of $O(2^{0.291n})$, which is faster than that of our algorithm. However, it requires more than $O(2^{0.256n})$ classical memory resources. When n is large, the memory consumption of the algorithm becomes non-negligible, which reduces the performance of the algorithm. However, we use quantum parallelism to encode and thus require only $n + t + 1$ qubits. Therefore, our algorithm has advantages in space consumption.

In this work, we aimed to construct a feasible and straightforward quantum algorithm to solve the SSP faster than the brute force algorithm and with less qubit consumption, which was further verified by simulation and real quantum devices. The remainder of this paper is organized as follows. In Section 2, we define the notations used in the paper and introduce the QPE and AA algorithms, which are used as building blocks in our algorithm. In Section 3, we present our algorithm, QSSP, and describe how to combine the QPE and AA algorithms to handle the SSP. In Section 4, we analyze the qubit consumption, time complexity, and success probabilities of our algorithm. In Section 5, we use Qiskit [36] to validate our algorithm, and deploy it in real quantum devices through the IBM Q Experience [37]. Finally, we present our conclusion and discuss future work in Section 6.

2 Preliminaries

2.1 Notations

$|\cdots\rangle$ denotes a quantum state, and $\langle\cdots|$ denotes the conjugate transpose of a quantum state. A state such as $|1\rangle$ represents all the qubits in this state are ones and its bitwise negation is defined as $|\bar{1}\rangle = |0\rangle$. Besides, the bold letters or words such as $\boldsymbol{A}$ or $\boldsymbol{U}_{\mathrm{QPE}}$ represent operators and their inverse operations are recorded as $\boldsymbol{A}^{-1}$ or ${\boldsymbol{U}_{\mathrm{QPE}}}^{-1}$. The Hardmard, Pauli-$\boldsymbol{X}$, Pauli-$\boldsymbol{Z}$ and controlled-$\boldsymbol{X}$ gates are noted as $\boldsymbol{H}$, $\boldsymbol{X}$, $\boldsymbol{Z}$ and CNOT, respectively. CZ$(\boldsymbol{j}, \boldsymbol{k})$ and CNOT$(\boldsymbol{j}, \boldsymbol{k})$ mean the controlled-$\boldsymbol{Z}$ gate and controlled-$\boldsymbol{X}$ gate, in which j acting as the control qubit to perform $\boldsymbol{Z}$ gate or $\boldsymbol{X}$ gate to the qubit k. The rotate gate has a matrix form of $\boldsymbol{R}(\theta) = \left(\begin{smallmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{smallmatrix}\right)$, it is also called $\boldsymbol{U}_1$ gate in the Qiskit [36]. The rotation gate along the y-axis is called $\boldsymbol{Ry}(\theta)$, it can be represented as $\boldsymbol{Ry}(\theta) = \left(\begin{smallmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{smallmatrix}\right)$. The function that finds the maximum (minimum) number is called max (min), and we use the notations $\lfloor\cdots\rfloor$ and $\lceil\cdots\rceil$ to represent the floor and ceiling functions, respectively. The notation $\pm$ and error bar represent the standard error (SE) of the samples. Specifically, $\mathrm{SE} = \Delta_s/\sqrt{N_s}$, where Δ_s is the sample standard deviation, and N_s is the number of samples.

Table 1 Heuristic asymptotic performance of various subset-sum algorithms[a)]

Quantum	Experiment	Time$^{(e)}$	Space	Algorithm
No	Yes	1	$O(1)$	Brute force
No	No	0.5	$O(2^{n/4})$	Left-right split [27]
No	Yes	0.291	$> O(2^{0.256n})$	Moduli + representation + overlap [28]
Yes	No	0.667	$> O(n2^{2n/3})$	Quantum walk [30]
Yes	Yes	0.5	$2nt_1 + n + 3t_1 + 5$ qubits	Quantum search [34]
Yes	Yes	0.5	$n + t + 1$ qubits	Our work*
Yes	No	0.241	$> O(8 \times 2^{0.271n})$	Quantum walk + representation [31]
Yes	No	0.226	$O(2^{0.226n})$	Quantum walk + representation [32]
Yes	No	0.218	$> O(2^{0.2356n})$	Merge and filter + representation [33]

a) Time$^{(e)}$ represents the algorithm using $O(2^{(e+o(1))n})$ operations. t_1 is the number of classical binary bits used to encode all elements in S, and t is the number of qubits used to encode all eigenvalues. The representation algorithm was introduced by Howgrave-Graham and Joux in [29].

2.2 Quantum phase estimation algorithm

The QPE algorithm is an eigenvalue solver proposed by Kitaev [38] that estimates the phase of the eigenvalue of a unitary operator $\boldsymbol{U}$. Suppose that the eigenvalue of $\boldsymbol{U}$ has the form $\mathrm{e}^{2\pi \mathrm{i}\phi}$, and that the corresponding eigenvector is $|b\rangle$. The QPE algorithm can return the approximate value of ϕ with an accuracy determined by the qubits used to store the eigenvalue when inputting the eigenvector $|b\rangle$. Usually, the QPE requires two registers to perform operations: the first register Reg0 contains t qubits initialized to $|0\rangle$ and is used to store phases, and the second register contains n qubits initialized to $|b\rangle$ and is used to store eigenvectors. The $\boldsymbol{H}$ gate is applied on all the qubits of Reg0, followed by the controlled operator controlled-$\boldsymbol{U}^{2^j}$ applied on the second register with the j-th qubit acting as the control qubit; then, the inverse quantum Fourier transform is applied to Reg0. We can read out the state of Reg0 on the computational basis to obtain the estimation of ϕ. In total, the QPE algorithm requires t applications of controlled-$\boldsymbol{U}^{2^j}$ and $O(t^2)$ other operations.

2.3 Amplitude amplification algorithm

The AA algorithm [39] is a generalized version of the Grover algorithm [40] and is used to amplify the probabilities of the partial quantum states that we require. Suppose that we have the superposition state of $|\psi\rangle = \boldsymbol{A}|0\rangle = \sqrt{M/N}|\psi_{\mathrm{good}}\rangle + \sqrt{(N-M)/N}|\psi_{\mathrm{bad}}\rangle$ generated by algorithm $\boldsymbol{A}$ from state $|0\rangle$, where $|\psi_{\mathrm{good}}\rangle$ is the state that we require, and $|\psi_{\mathrm{bad}}\rangle$ is the state that we want to eliminate. The AA algorithm repeats the iterator $\boldsymbol{G}$ to amplify the amplitude of $|\psi_{\mathrm{good}}\rangle$. The iterator $\boldsymbol{G} = \boldsymbol{U}_s\boldsymbol{U}_f$ contains two steps.

Step 1. Apply the oracle operator $\boldsymbol{U}_f$ to mark the desired state and keep the other states unchanged by the following equation:

$$\boldsymbol{U}_f(|\psi_{\mathrm{good}}\rangle + |\psi_{\mathrm{bad}}\rangle) = -|\psi_{\mathrm{good}}\rangle + |\psi_{\mathrm{bad}}\rangle. \tag{1}$$

Step 2. Perform the second reflect operator $\boldsymbol{U}_s$ to amplify the probability of $|\psi_{\mathrm{good}}\rangle$, where $\boldsymbol{U}_s$ is defined as

$$\boldsymbol{U}_s = \boldsymbol{I} - 2|\psi\rangle\langle\psi| = -\boldsymbol{A}\boldsymbol{U}_t\boldsymbol{A}^{-1}, \tag{2}$$

where $\boldsymbol{U}_t = \boldsymbol{I} - 2|0\rangle\langle 0|$. Let $\sin(\theta) = \sqrt{M/N}$ and $\theta \in (0, \pi/2]$. After j applications of operator $\boldsymbol{G}$, the state becomes

$$\boldsymbol{G}^j|\psi\rangle = \sin((2j+1)\theta)|\psi_{\mathrm{good}}\rangle + \cos((2j+1)\theta)|\psi_{\mathrm{bad}}\rangle. \tag{3}$$

By measuring the state, we can obtain $|\psi_{\mathrm{good}}\rangle$ with a probability equal to $\sin^2((2j+1)\theta)$. Supposing that the value of $\frac{M}{N} > 0$ is known, we can set the iteration time to $j = \lfloor \frac{\pi}{4}\sqrt{N/M} \rfloor$. Then, when we compute $\boldsymbol{G}^j|\psi\rangle$ and measure the system, the outcome is $|\psi_{\mathrm{good}}\rangle$, with a probability of at least $\max(1 - \frac{M}{N}, \frac{M}{N})$. This conclusion is regarded as a quadratic speedup for the reason that if algorithm $\boldsymbol{A}$ has a success probability of $\frac{M}{N}$ greater than zero, then after an expected number of $\frac{N}{M}$ applications of $\boldsymbol{A}$, we can obtain a good solution. Applying the above AA algorithm reduces the expected number to at most $O(\sqrt{N/M})$ applications of $\boldsymbol{A}$ and $\boldsymbol{A}^{-1}$.

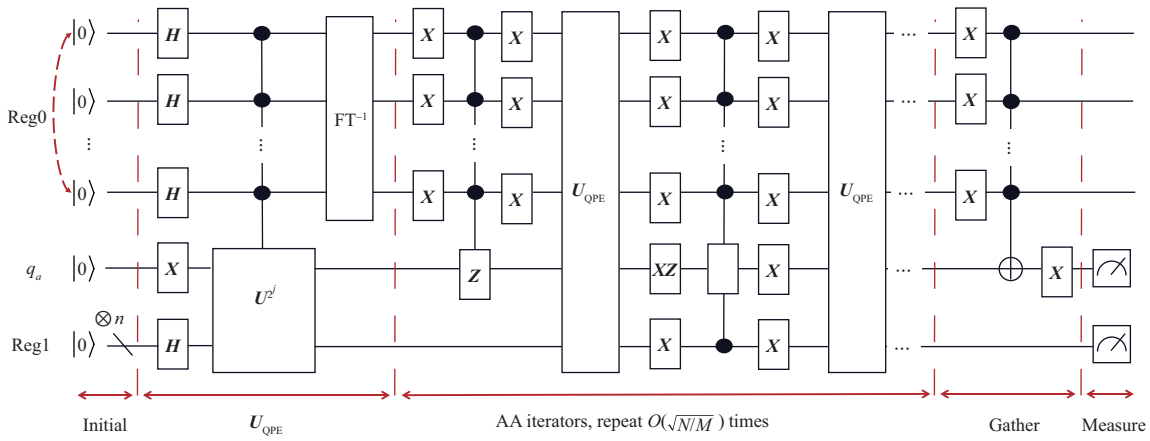

Figure 1 (Color online) Schematic diagram of the proposed algorithm. All the qubits are initialized to $|0\rangle$. $\boldsymbol{U}_{\mathrm{QPE}}$ represents the quantum phase estimation operator, FT^{-1} represents the inverse quantum Fourier transform, and q_a represents the ancilla qubit.

3 Proposed algorithm

The main strategy of our proposed algorithm is to convert the problem of whether there is a subset sum equal to w to the problem of whether there is a phase equal to zero. We encode each element of S and w to a phase gate acting on one qubit, thereby generating a unitary matrix $\boldsymbol{U}$ with all the subset sums of S and $-w$ acting as its diagonal elements. Then, we use $\boldsymbol{U}$ to form the controlled operator and estimate all the phases of the diagonal elements with equal weight through QPE. Thereafter, the AA algorithm can be used to amplify the amplitude of $|0\rangle$, and we can obtain all the solution subsets with the indices stored in the eigenvector register. In total, there are five steps.

(1) Encode S and w into rotate operators.

(2) Construct the controlled operator controlled-$\boldsymbol{U}^{2^j}$ used by QPE.

(3) Apply QPE to the equal superposition state of all the subset sums of S and $-w$.

(4) Utilize the AA algorithm to amplify the amplitude of $|0\rangle$.

(5) Use the controlled operator to obtain the final results and measure the state on a computational basis.

The entire quantum circuit of the above five steps is presented in Figure 1, and the pseudocode of our algorithm QSSP is outlined in Algorithm 1. Each step is explained in detail in Subsections 3.1–3.5.

Algorithm 1 QSSP

Inputs: $|\psi\rangle = |\mathrm{Reg1}\rangle|q_a\rangle|\mathrm{Reg0}\rangle = |0\rangle^{\otimes n}|0\rangle|0\rangle^{\otimes t}$. // q_a is the ancilla qubit.
Outputs: $|\mathrm{Reg1}\rangle|q_a\rangle$. // When $|q_a\rangle = |1\rangle$, the index of answer subset stores in Reg1.
Runtime: $O(2^{(0.5+o(1))n})$ operations with success probabilities of $P = \sin^2((2j+1)\arcsin\sqrt{M/2^n})$. When $M \ll 2^n$, we will get $P \approx 1$ after $O(\lfloor \frac{\pi}{4}\sqrt{2^n/M} \rfloor)$ iterations.
Procedure:
1: Build rotate operators $\boldsymbol{R}_i$ and $\boldsymbol{R}_w$.
2: Construct oracle operator used by QPE through $\boldsymbol{U} = \boldsymbol{R}_w \otimes \boldsymbol{R}_n \otimes \boldsymbol{R}_{n-1} \otimes \cdots \otimes \boldsymbol{R}_1$.
3: Apply $\boldsymbol{U}_{\mathrm{QPE}}$ to $|\psi\rangle$.
4: Utilize AA iterator $\boldsymbol{G}' = \boldsymbol{U}_s\boldsymbol{U}_f$ for $j \leqslant \lfloor \frac{\pi}{4}\sqrt{2^n/M} \rfloor$ times.
5: Apply $\boldsymbol{U}_{\mathrm{gather}}$ to gather the results and measure $|\mathrm{Reg1}\rangle|q_a\rangle$.

3.1 Encoding

The phase gate is a generalize method to encode the value to phase. Specifically, for the integer set $S = \{s_1, s_2, s_3, \ldots, s_{n-1}, s_n\}$ and target w, we scaler them so that both $\tilde{w}$ and $\tilde{s_j}$ are in (0,1) by

$$\tilde{s_j} = \frac{s_j}{\sum_{j=1}^{n} s_j + w}, \quad \tilde{w} = \frac{w}{\sum_{j=1}^{n} s_j + w}. \tag{4}$$

Then, coding $\tilde{s_j}$ in the rotation gate applies to the j-th qubit, and coding w in $\boldsymbol{R}_w$ applies to the $(n+1)$-th qubit by the following equations:

$$\boldsymbol{R}_j = \begin{pmatrix} 1 & 0 \\ 0 & \mathrm{e}^{2\pi \mathrm{i}\tilde{s_j}} \end{pmatrix}, \quad \boldsymbol{R}_w = \begin{pmatrix} 1 & 0 \\ 0 & \mathrm{e}^{2\pi \mathrm{i}(-\tilde{w})} \end{pmatrix}. \tag{5}$$

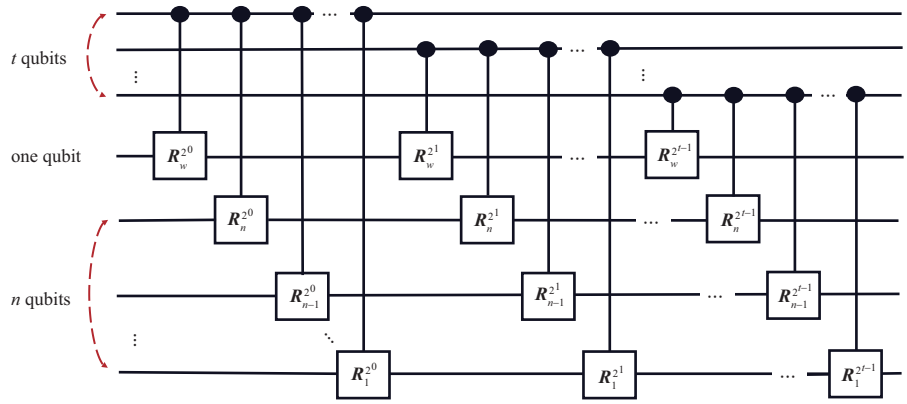

Figure 2 (Color online) Schematic diagram of controlled-$\boldsymbol{U}^{2^j}$.

3.2 Controlled operator

The key of QPE is the oracle of controlled-operator and we first generate the oracle operator by

$$\begin{aligned}\boldsymbol{U} &= \boldsymbol{R}_w \otimes \boldsymbol{R}_n \otimes \boldsymbol{R}_{n-1} \otimes \cdots \otimes \boldsymbol{R}_1 \\ &= \begin{pmatrix} \mathrm{e}^{2\pi \mathrm{i}\tilde{w}}\boldsymbol{D} & 0 \\ 0 & \boldsymbol{D} \end{pmatrix},\end{aligned} \tag{6}$$

where $\boldsymbol{D}$ is shown as

$$\begin{pmatrix} \mathrm{e}^{\theta(-\tilde{w})} & 0 & 0 & \cdots & 0 \\ 0 & \mathrm{e}^{\theta(\tilde{s_1}-\tilde{w})} & 0 & \cdots & 0 \\ 0 & 0 & \mathrm{e}^{\theta(\tilde{s_2}-\tilde{w})} & \cdots & 0 \\ 0 & 0 & 0 & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \mathrm{e}^{\theta(\sum_j \tilde{s_j}-\tilde{w})} \end{pmatrix}, \tag{7}$$

where $\theta = 2\pi\mathrm{i}$, and the diagonal elements of $\boldsymbol{D}$ are formed by $\lambda_k = \mathrm{e}^{\theta(\delta_k - \tilde{w})}$, where δ_k is the k-th element of all the subset sums of $\tilde{S}$ with the order $\{0, \tilde{s_1}, \tilde{s_2}, \tilde{s_1}+\tilde{s_2}, \ldots, \tilde{s_1}+\tilde{s_2}+\cdots+\tilde{s_n}\}$, which corresponds to the eigenvector of the k-th standard basis. $\delta_k - \tilde{w} = 0$ if there is at least one subset sum equal to w; therefore, we can judge whether w is the subset sum of the set by verifying whether the phase is zero. Then, we add control qubits of each rotation gate in ascending order and construct the controlled operator controlled-$\boldsymbol{U}^{2^j}$, as illustrated in Figure 2.

3.3 Quantum phase estimation

At this stage, we add an ancilla qubit q_a to encode w. Reg1, q_a, and Reg0 are initialized with $|\psi_0\rangle = |\mathrm{Reg1}\rangle|q_a\rangle|\mathrm{Reg0}\rangle = |0\rangle^{\otimes n}|0\rangle|0\rangle^{\otimes t}$. To generate all the possible sums of all subsets, we apply $\boldsymbol{H}^{\otimes n} \otimes \boldsymbol{X} \otimes \boldsymbol{H}^{\otimes t}$ to $|\psi_0\rangle$, and the state becomes

$$|\psi_1\rangle = \left(\frac{1}{\sqrt{2^n}}\sum_{k=0}^{2^n-1}|k\rangle\right)|b\rangle = \left(\frac{1}{\sqrt{2^n}}\sum_{k=0}^{2^n-1}|k\rangle\right)|1\rangle\left(\frac{1}{\sqrt{2^t}}\sum_{j=0}^{2^t-1}|j\rangle\right). \tag{8}$$

After that, the controlled-$\boldsymbol{U}^{2^j}$ and inverse Fourier transform are applied to the system. We finally gain the state:

$$|\psi_2\rangle = \frac{1}{\sqrt{2^n}}\sum_{k=0}^{2^n-1}|k\rangle|1\rangle|\phi_k\rangle. \tag{9}$$

When $|\mathrm{Reg0}\rangle = |\phi_k\rangle = |0\rangle$, we have one subset sum equal to w, and the exact index of the subset is represented by the binary form of k. Suppose that $k = k_n k_{n-1} \cdots k_1$, where k_1 is the least significant bit. When $k_j = 1$, s_j is an element of the solution subset; otherwise, s_j is not. For convenience, we denote the entire phase estimation operation above as operator $\boldsymbol{U}_{\mathrm{QPE}}$.

3.4 Quantum amplitude amplification

By the QPE algorithm, all subsets with a sum equal to w are associated with $|\text{Reg0}\rangle = |0\rangle$. We can use the AA algorithm to amplify their probabilities. Prior to this, we divide the quantum state of $|\psi_2\rangle$ into the combination of a good state and bad state; the good state refers to the state of the subset that sums to w (meaning that the eigenvector leads to $|\text{Reg0}\rangle = |0\rangle$), while the other states are bad states. Specifically,

$$\begin{aligned}|\psi_2\rangle &= \frac{1}{\sqrt{2^n}} \sum_{k\in\text{good}} |k\rangle|1\rangle|\phi_k\rangle + \frac{1}{\sqrt{2^n}} \sum_{k\in\text{bad}} |k\rangle|1\rangle|\phi_k\rangle \\ &= \frac{1}{\sqrt{2^n}} \sum_{k\in\text{good}} |k\rangle|1\rangle|0\rangle + \frac{1}{\sqrt{2^n}} \sum_{k\in\text{bad}} |k\rangle|1\rangle|\phi_k\rangle. \end{aligned} \tag{10}$$

Then, we need to construct the iterator $\boldsymbol{G} = \boldsymbol{U}_s\boldsymbol{U}_f = -\boldsymbol{A}\boldsymbol{U}_t\boldsymbol{A}^{-1}\boldsymbol{U}_f$, where operator $\boldsymbol{U}_f$ marks the state when $|\text{Reg0}\rangle = |0\rangle$, and $\boldsymbol{U}_s$ amplifies its amplitude. Suppose that we have $j + k$ qubits, that $\boldsymbol{U}$ is a k qubit unitary operator, and that j qubits act as the control qubits. We define the multiple controlled operator $\boldsymbol{C}^{j,k}(\boldsymbol{U})$ by the following equation:

$$\boldsymbol{C}^{j,k}(\boldsymbol{U})|y_1y_2\cdots y_j\rangle|\psi\rangle = |y_1y_2\cdots y_j\rangle\boldsymbol{U}^{y_1y_2\cdots y_j}|\psi\rangle, \tag{11}$$

where $y_1y_2\cdots y_j$ in the exponent of $\boldsymbol{U}$ means the product of bits $y_1, y_2, \ldots, y_j$. The operator $\boldsymbol{U}$ is applied to the last k qubits if the first j qubits are all equal to one. Otherwise, nothing is done. Then $\boldsymbol{U}_f$ and $\boldsymbol{U}_t$ can be defined as

$$\boldsymbol{U}_f = \boldsymbol{X}^{\otimes t}\boldsymbol{C}^{t,1}(\boldsymbol{Z})\boldsymbol{X}^{\otimes t},\ \boldsymbol{U}_t = \boldsymbol{X}^{\otimes t+n+1}\boldsymbol{C}^{t+n,1}(\boldsymbol{Z})\boldsymbol{X}^{\otimes t+n+1}. \tag{12}$$

The $\boldsymbol{C}^{t,1}(\boldsymbol{Z})$ means all the qubits of Reg0 acting as control qubits to perform controlled-$\boldsymbol{Z}$ gate to the ancilla qubit, and the $\boldsymbol{C}^{t+n,1}(\boldsymbol{Z})$ means all the qubits of Reg0 and Reg1 acting as control qubits to perform controlled-$\boldsymbol{Z}$ gate to the ancilla qubit. Hence the revised iterator $\boldsymbol{G}'$ can be expressed as

$$\boldsymbol{G}' = -\boldsymbol{U}_{\text{QPE}} \times \boldsymbol{U}_t \times {\boldsymbol{U}_{\text{QPE}}}^{-1} \times \boldsymbol{U}_f. \tag{13}$$

Assume that the number of elements in good is $|\text{good}| = M$. Since the total number is $N = 2^n$, the number of elements in bad is $|\text{bad}| = N - M$. When we apply operator $\boldsymbol{G}'$ j times to $|\psi_2\rangle$, we obtain

$$|\psi_3\rangle = \boldsymbol{G}'^j|\psi_2\rangle = \frac{\sin((2j+1)\theta)}{\sqrt{M}} \sum_{k\in\text{good}} |k\rangle|1\rangle|0\rangle + \frac{\cos((2j+1)\theta)}{\sqrt{N-M}} \sum_{k\in\text{bad}} |k\rangle|1\rangle|\phi_k\rangle. \tag{14}$$

3.5 Obtaining the results

In state $|\psi_3\rangle$, all the subsets that sum to w lead to $|\text{Reg0}\rangle = |0\rangle$. Otherwise, the qubits of Reg0 have at least one $|1\rangle$. Now, we use the ancilla qubit q_a to obtain the results. We first flip all the qubits of Reg0 through the $\boldsymbol{X}$ gate. In detail,

$$|\psi_4\rangle = (\boldsymbol{I}^{\otimes n} \otimes \boldsymbol{I} \otimes \boldsymbol{X}^{\otimes t})|\psi_3\rangle = \frac{\sin((2j+1)\theta)}{\sqrt{M}} \sum_{k\in\text{good}} |k\rangle|1\rangle|1\rangle + \frac{\cos((2j+1)\theta)}{\sqrt{N-M}} \sum_{k\in\text{bad}} |k\rangle|1\rangle|\bar{\phi_k}\rangle. \tag{15}$$

The $\bar{\phi_k}$ means all the qubits of Reg0 are flipped. Then apply $\boldsymbol{C}^{t,1}(\boldsymbol{X})$ to Reg0 and q_a, we will get

$$|\psi_5\rangle = \boldsymbol{C}^{t,1}(\boldsymbol{X})|\psi_4\rangle = \frac{\sin((2j+1)\theta)}{\sqrt{M}} \sum_{k\in\text{good}} |k\rangle|0\rangle|1\rangle + \frac{\cos((2j+1)\theta)}{\sqrt{N-M}} \sum_{k\in\text{bad}} |k\rangle|1\rangle|\bar{\phi_k}\rangle. \tag{16}$$

$\boldsymbol{C}^{t,1}(\boldsymbol{X})$ means all the qubits of Reg0 acting as control qubits to perform controlled-$\boldsymbol{X}$ gate on q_a. Finally, we can get the results from q_a. These eigenvectors stored in Reg1 lead to $|q_a\rangle = |0\rangle$ that are the indexes of answer subsets. In order to make the consistency with logic, we then use $\boldsymbol{X}$ gate to flip q_a and get

$$|\psi_6\rangle = (\boldsymbol{I}^{\otimes n} \otimes \boldsymbol{X} \otimes \boldsymbol{I}^{\otimes t})|\psi_5\rangle = \frac{\sin((2j+1)\theta)}{\sqrt{M}} \sum_{k\in\text{good}} |k\rangle|1\rangle|1\rangle + \frac{\cos((2j+1)\theta)}{\sqrt{N-M}} \sum_{k\in\text{bad}} |k\rangle|0\rangle|\bar{\phi_k}\rangle. \tag{17}$$

When measuring, we only need to measure the state of the ancilla qubit q_a. If it is $|1\rangle$, we have a solution. We measure the eigenvector register Reg1, which stores the detailed index of the solution subset. For convenience, we define all operations used in obtaining the results as $\boldsymbol{U}_{\text{gather}}$.

4 Performance analysis

For set $S = \{s_1, s_2, s_3, \ldots, s_{n-1}, s_n\}$, we suppose that there are M subsets whose sum is equal to target w. Now, we must determine the resource consumption and total runtime.

4.1 Qubit consumption

From our encoding strategy, to construct one controlled-$\boldsymbol{U}^{2^j}$, we require n qubits to encode each element of S, and one qubit to encode w. To distinguish all phases generated by the phase estimation algorithm, we require $\min_{j,k=0}^{2^n} |\phi_j - \phi_k|$, which can be represented by t qubits when $\phi_j \neq \phi_k$. That is

$$\begin{aligned} t &= \left\lceil \log_2 \left(\frac{1}{\min_{j,k=0, j\neq k}^{2^n} |\phi_j - \phi_k|} \right) \right\rceil \\ &\leqslant \left\lceil \log_2 \left(\sum_{j=1}^{n} s_j + w \right) \right\rceil = \lceil \log_2(\mu + w) \rceil. \end{aligned} \tag{18}$$

Then the total qubits we need is $n+t+1$. Assume that all elements in S can be represented by classic bits of $t_1 \geqslant 1$, that is, $s_j \leqslant 2^{t_1}$ holds for $j \in [1, n]$. Since $w \in (0, \mu]$ will cause $\mu + w = \sum_{j=1}^{n} s_j + w \leqslant 2n \times 2^{t_1}$, our qubit consumption becomes

$$\begin{aligned} n + t + 1 &\leqslant n + (\log_2(\mu + w) + 1) + 1 \\ &\leqslant n + \log_2(2n \times 2^{t_1}) + 2 \\ &= n + 2 + t_1 + \log_2(2n) \\ &\leqslant n + 2 + t_1 + 2n, \text{ when } n > 1 \\ &\leqslant n + 2 + t_1 + 2nt_1 \\ &< n + 2 + t_1 + 2nt1 + (3 + 2t_1) \\ &= 2nt_1 + n + 3t_1 + 5. \end{aligned} \tag{19}$$

Therefore, our algorithm uses fewer qubits than the method in [34] when $n > 1$.

4.2 Time complexity

When we regard $\boldsymbol{C}^{j,1}(\boldsymbol{U})$ and the basic single-qubit gate as a one-atom operation with the time complexity of $O(1)$, then the total complexity of our algorithm can be divided into three parts. The first part is the QPE stage, which has one $\boldsymbol{U}_{\mathrm{QPE}}$ with operations of

$$\mathrm{Count}_{\mathrm{QPE}} = t^2/2 + 2nt + 2t + 1. \tag{20}$$

The second part is the AA procedure with $O(\sqrt{2^n/M})$ AA iterations, and the third part is the stage of obtaining results with $\mathrm{Count}_{\mathrm{gather}} = t + 2$ operations. For the second part, we know that each iterator $\boldsymbol{G}'$ contains one QPE, one inverse QPE, one operator $\boldsymbol{U}_f$, and one operator $\boldsymbol{U}_t$. The total number of operations of one $\boldsymbol{G}'$ is as follows:

$$\mathrm{Count}_{G'} = t^2 + 4nt + 8t + 6. \tag{21}$$

From the AA algorithm, we need about $\lfloor \frac{\pi}{4}\sqrt{\frac{2^n}{M}} \rfloor$ iterations to gain the maximum probability. So the operations used in the AA stage are

$$\mathrm{Count}_{\mathrm{AA}} = \left\lfloor \frac{\pi}{4}\sqrt{\frac{2^n}{M}} \right\rfloor \times \mathrm{Count}_{G'} = \left\lfloor \frac{\pi}{4}\sqrt{\frac{2^n}{M}} \right\rfloor (t^2 + 4nt + 8t + 6). \tag{22}$$

Combining these three stages, the AA procedure dominates the total complexity of $O(\frac{\pi}{4}\sqrt{2^n/M}(t^2+4nt))$. Therefore, we can approximately use the number of iterations j of the AA operator $\boldsymbol{G}'$ to express the performance of our algorithm. When $M \ll 2^n$, we obtain a success probability $P \approx 1$ after using $j = \lfloor \frac{\pi}{4}\sqrt{2^n/M} \rfloor$ AA iterations (see Subsection 4.3 for a detailed discussion); therefore, we use $O(\lfloor \frac{\pi}{4}\sqrt{2^n/M} \rfloor)$

applications of the AA iterator to represent our runtime. Suppose that $(t^2+4nt)/\sqrt{M}$ is not exponentially large; that is, t is not in the form of $O(2^n)$. Specifically, if the maximum element of S is not equal to $O(2^{(2^{n-1}/n)})$, then $(t^2+4nt)/\sqrt{M}$ is not exponentially large. For the convenience of analysis, let $M=1$, and let S be a random integer set of n (sufficiently large) elements. This is a difficult problem for a classical computer because the probability of success is $1/2^n$, which is small. Considering that the general case on classical computers is $t=O(\text{poly}(n))$, the complexity of our algorithm can be rewritten as $O(2^{(0.5+o(1))n})$, where $o(1)=(\log_2(t^2+4nt))/n$. For example, supposing that we randomly select $n=2^{21}\approx 2\times 10^6$ integers from $(0,2^{42}]$ to form S, we can use up to $t=64$ qubits to store the eigenvalues. Then, $o(1)=0.0000138$ can be ignored compared with 0.5, and our complexity becomes approximately $O(2^{0.5n})$. Compared with the brute force algorithm with a time complexity of $O(2^n)$, we obtain a quadratic speedup.

4.3 Success probability

Since we just focus on the situations where $|\phi\rangle=|0\rangle$ that will lead to $|q_a\rangle=|1\rangle$. When we set t so that all phases are distinguishable. If there is no error, the total success probability after the j-th AA iteration is then $P=\sin^2((2j+1)\arcsin\sqrt{M/2^n})$ according to (17). Let $\theta=\arcsin(\sqrt{M/2^n})$ and we will gain the maximum success probability $P=1$ when $(2j+1)\theta=\pi/2$, that means $j=(\pi-2\theta)/(4\theta)$. But j must be an integer number, let $j=\lfloor\pi/(4\theta)\rfloor$ and $\tilde{j}=(\pi-2\theta)/(4\theta)$. Note that $|j-\tilde{j}|\leqslant 1/2$, therefore $|(2j+1)\theta-(2\tilde{j}+1)\theta|\leqslant\theta$. Since $(2\tilde{j}+1)\theta=\pi/2$ that leads to $|\cos((2j+1)\theta)|\leqslant|\sin(\theta)|$, we will get a success probability of $P=1-\cos^2((2j+1)\theta)\geqslant 1-\sin^2(\theta)=1-M/2^n$ when iterations $j=\lfloor\pi/(4\theta)\rfloor$. If $M\ll 2^n$, then $P\approx 1$. Since $\theta>\sin(\theta)=\sqrt{M/2^n}$ and $j=\lfloor\frac{\pi}{4\theta}\rfloor\leqslant\lfloor\frac{\pi}{4}\sqrt{2^n/M}\rfloor$, we need $O(\lfloor\frac{\pi}{4}\sqrt{2^n/M}\rfloor)$ iterations to gain the maximum success probability $P\approx 1$.

5 Validation

5.1 Simulation

We used the set $S=\{2,3,5,7\}$ and $w=12$ to validate our algorithm. We first simulated the algorithm with the ibmq_simulator through the open-source tool Qiskit [36]. The experimental circuit followed Figure 1. To make all the phases distinguishable, we selected $t=\lceil\log_2(\sum_{j=1}^{n}s_j+w)\rceil=5$. In total, there needed to be $n+t+1=4+5+1=10$ qubits to perform operations. Of these, five qubits Reg0 were used to distinguish all phases generated by QPE, four qubits Reg1 were used to encode the set, and one ancilla qubit q_a was used to code the target w and store the results. To illustrate the speedup process, we obtained the results and measured the eigenvector register Reg1 and ancilla qubits q_a after each AA iteration, where q_a is the least significant qubit. For each quantum circuit, we repeated the measurement 8192 times. Each experiment was repeated five times. Figure 3 presents the test results. The good states are 23 ($|\text{Reg1}\rangle|q_a\rangle=|1011\rangle|1\rangle$) and 25 ($|\text{Reg1}\rangle|q_a\rangle=|1100\rangle|1\rangle$). Therefore, the subset indices are 1011 and 1100, which correspond to the solution subsets of $\{2,3,7\}$ and $\{5,7\}$, respectively. For iterations $j=[0,1,2,3]$, we obtained the success probabilities of these good states as $[0.124\pm0.003, 0.773\pm0.003, 0.914\pm0.007, 0.293\pm0.007]$. From Section 4, the theoretical success probabilities are $[0.125, 0.781, 0.945, 0.330]$, which indicates that the simulation is consistent with the theory. These results demonstrate that for a set with four elements and two subsets whose sum is equal to w, we require two AA iterations to obtain the maximum success probability of 0.914 ± 0.007. In contrast, using the brute force algorithm requires an average of eight iterations. Since $2\leqslant\lfloor\frac{\pi}{4}\sqrt{N/M}\rfloor=\lfloor\frac{\pi}{4}\sqrt{16/2}\rfloor=2$, this demonstrates the quadratic acceleration of our algorithm.

In fact, when the above simulation is performed, Qiskit decomposes the quantum circuit of our experiment into a complete set of gates composed of single-qubit gates and double-qubit gates (CNOT). Since each quantum gate has an error probability, the performance of our algorithm is related to the error rates of the gate. In Figure 4, we illustrate the relationship between the success probabilities of our algorithm in solving $S=\{2,3,5,7\}$ and $w=12$ and the single-qubit gate error rates and CNOT gate error rates. We explored the influence of the single-qubit gate error rates and CNOT gate error rates on the probability of good states ($|1011\rangle|1\rangle$ and $|1100\rangle|1\rangle$) when the number of iterations was 1 and 2, respectively. The results indicate that when the single-qubit gate error rates and CNOT gate error rates were approximately 10^{-6}, we obtained a probability of success close to the theoretical value. When the error rates of the quantum gate were greater than 10^{-5}, the success probabilities of the algorithm

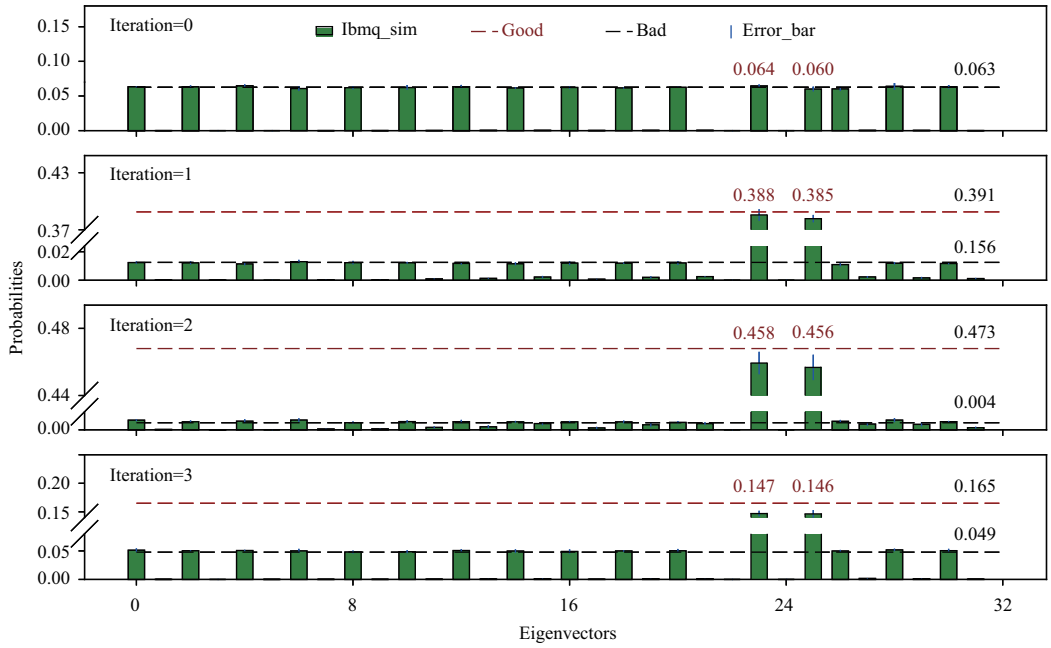

Figure 3 (Color online) Simulation of $S = \{2, 3, 5, 7\}$ and $w = 12$. We measured a total of five qubits, including Reg1 and q_a (where q_a is the least significant qubit), when the number of iterations was equal to $[0, 1, 2, 3]$. The green bars represent the measurement results of the decimal form of $|\text{Reg1}\rangle|q_a\rangle$. The black dashed lines represent the theoretical probabilities of the bad items, while the red dashed lines represent the theoretical probabilities of the good states we require. Note that the error bars represent the standard errors of the samples.

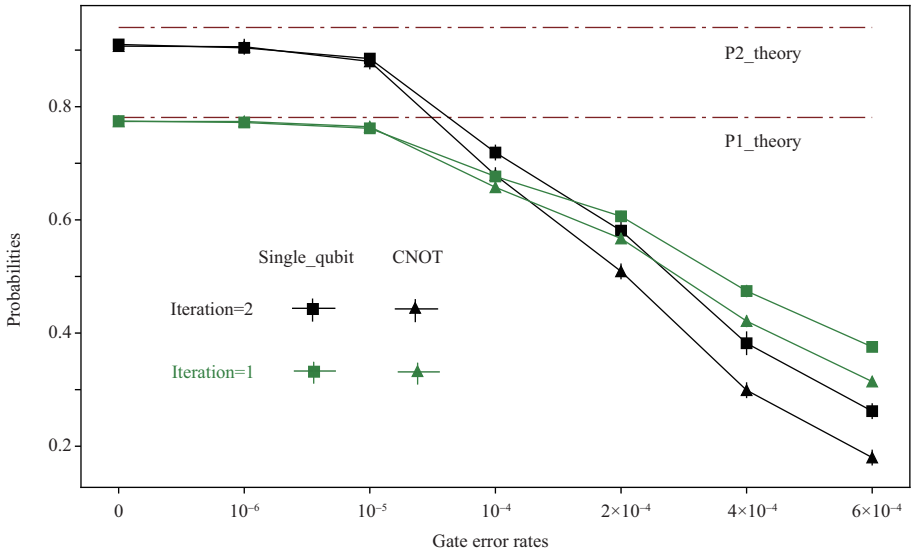

Figure 4 (Color online) Relationship between the success probabilities of our algorithm in solving $S = \{2, 3, 5, 7\}$ and $w = 12$ and the single-qubit gate error rates and CNOT gate error rates. P1_theory and P2_theory denote the theoretical success probabilities for iteration=1, 2 respectively.

decreased rapidly. In addition, in our example, the CNOT gate error rates had a slightly greater impact on the success probabilities than the single-qubit gate error rates. Therefore, reducing the error rates of the quantum gate to an appropriate range has significant meaning for improving the probabilities of success of our algorithm.

5.2 Experiments

The simulation results indicate that we need 10 qubits to perform operations and two applications of the AA iterator to obtain the maximum probability. We then considered how to implement this algorithm on

Table 2 Calibration parameters of ibmq_bogata and ibmq_santiago archived on August 6, 2021 from the IBM quantum website[a)]

Device	Qubit	Readout error (10^{-2})	Single-qubit gate error (10^{-4})	CNOT error (10^{-3})
bogota	q_0	2.78	1.97	0_1:9.118
	q_1	2.29	2.81	1_2:8.288; 1_0:9.118
	q_2	3.23	1.41	2_3:8.477; 2_1:8.288
	q_3	1.48	6.28	3_4:9.054; 3_2:8.477
	q_4	1.41	1.55	4_3:9.054
santiago	q_0	4.94	5.67	0_1:9.770
	q_1	1.80	2.77	1_2:12.54; 1_0:9.770
	q_2	4.33	9.54	2_3:14.94; 2_1:12.54
	q_3	0.76	2.31	3_4:5.568; 3_2:14.94
	q_4	1.31	1.78	4_3:5.568

a) It should be noted that these parameters are updated on a daily basis. In the column "CNOT error", "j_k" means that j is the control qubit.

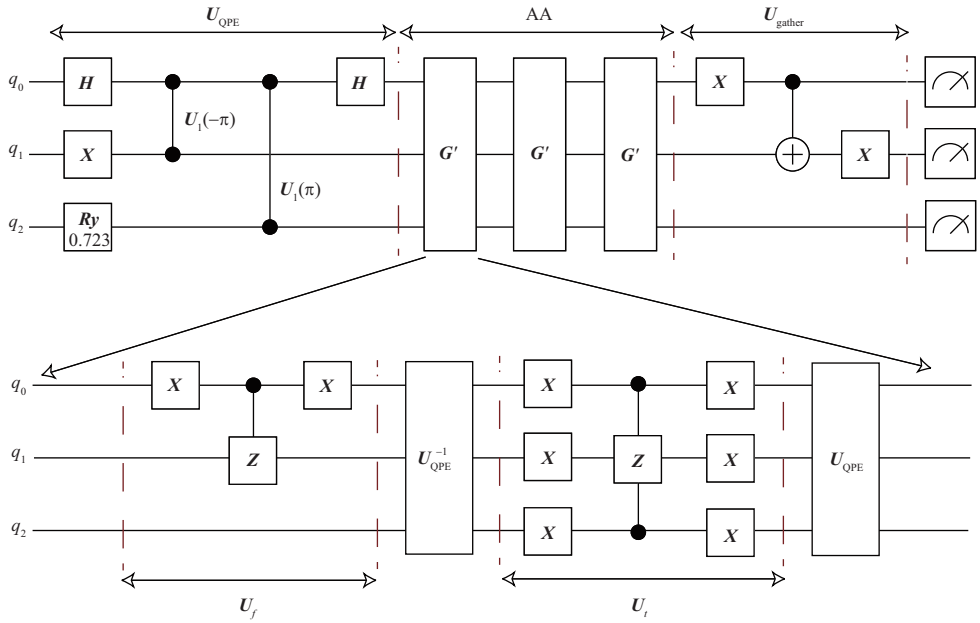

Figure 5 (Color online) Simplified experimental circuits for SSP of $S = \{2, 3, 5, 7\}$ and $w = 12$. We simplify the problem to $S = \{1\}$, $w = 1$, and a single element of S has a probability of 1/8 to be selected. The AA stage contains iterations of $[0, 1, 2, 3]$, and the above circuit pertains to three iterations.

truly quantum devices. The IBM Q Experience [37] has allowed access to its cloud quantum computers. There are more than 20 quantum qubits as of now, and the best-performing computer has reached a quantum volume (QV) [41] of 128. However, licenses are only available for computers with a QV of up to 32. Therefore, we implemented our algorithm on machines called ibmq_santiago and ibmq_bogota, which had five qubits and a QV of 32. The topological structures of these two machines are chains, and the error rates of each are presented in Table 2,

Due to the limited qubits, our problem must be simplified. For the SSP of set $S = \{2, 3, 5, 7\}$ and $w = 12$, we know that two subsets $\{2, 3, 5\}$ and $\{5, 7\}$ satisfy this condition from the simulation results, and there are a total of 16 subsets of S; therefore, the success probability is 1/8. We first divide $\{2, 3, 5\}$ and $\{5, 7\}$ into the good state $|1\rangle$ and the other subsets into the bad state $|0\rangle$; therefore, we have the state $|\psi\rangle = \sqrt{1/8}|1\rangle + \sqrt{7/8}|0\rangle$. Then, we convert the problem into $S = \{1\}$, $w = 1$, and a single element of S has a probability of 1/8 to be selected. Now, the eigenvector register Reg1 is one qubit, we use Reg1 to store the eigenvector. Then, we can apply $\boldsymbol{Ry}(2 \times \arcsin(\sqrt{1/8}))$ gate to $|0\rangle$ to generate this state. To distinguish all the phases of 0 and 0.5, we need one qubit to store the eigenvalue. In addition, we need another qubit to encode w. After simplification, we need three qubits: Reg0 is formed by q_0, Reg1 is formed by q_2, and q_a is q_1. The experimental circuits are presented in Figure 5.

During the QPE, all the qubits are initialed to $|0\rangle$. Let $|b\rangle = (\boldsymbol{Ry}(2 \times \arcsin(\sqrt{1/8})) \otimes \boldsymbol{X})|0\rangle|0\rangle$ and

Table 3 The success probabilities of the simulator, ibmq_bogata (device) and ibmq_santiago (device) under different AA iterations

	Iteration = 0	Iteration = 1	Iteration = 2	Iteration = 3
P_{theory}	0.125	0.781	0.945	0.330
$P_{\text{simulator}}$	0.133 ± 0.010	0.782 ± 0.015	0.940 ± 0.004	0.332 ± 0.013
P_{bogota}	0.153 ± 0.018	0.694 ± 0.034	0.665 ± 0.060	0.448 ± 0.045
P_{santiago}	0.190 ± 0.024	0.726 ± 0.057	0.751 ± 0.040	0.425 ± 0.053

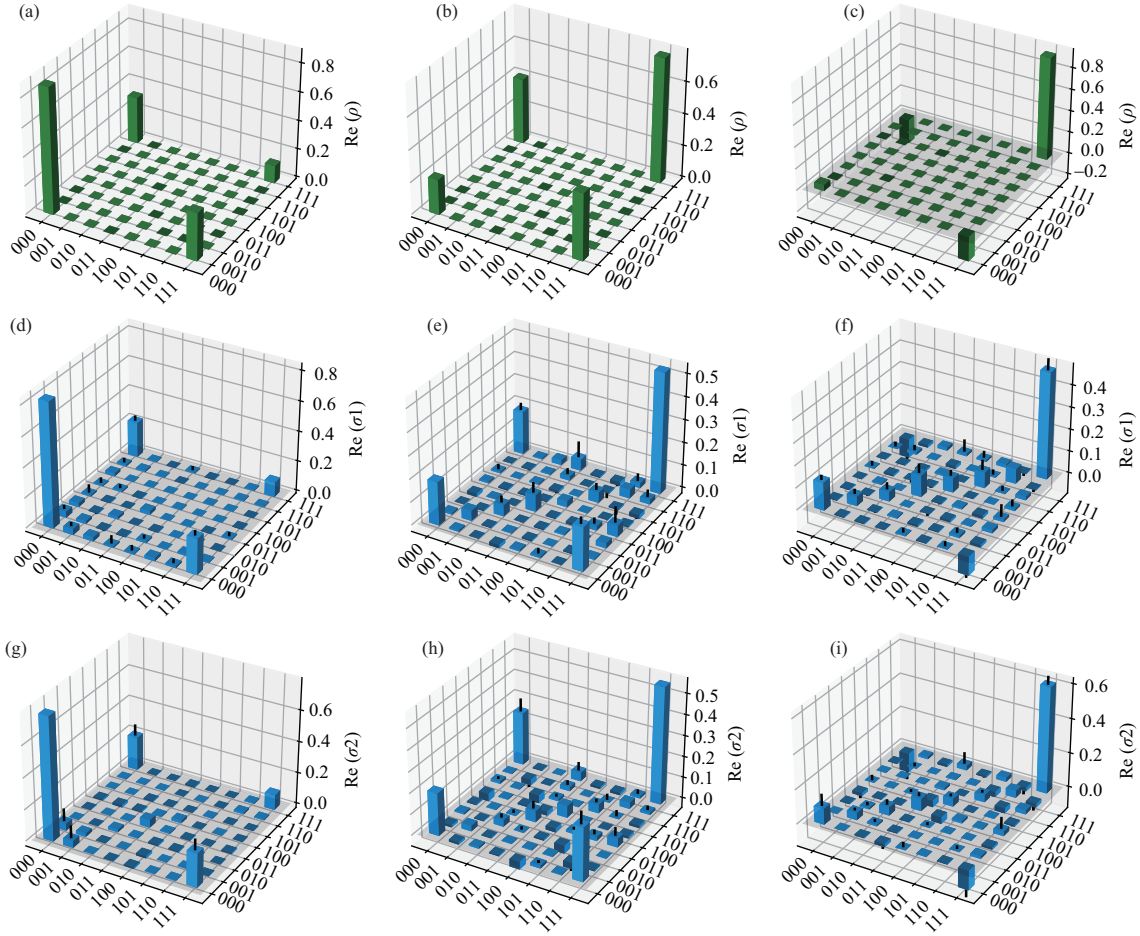

Figure 6 (Color online) ρ is dense matrix of the theory, $\sigma 1$ is the reconstructed dense matrix of ibmq_bogota, and $\sigma 2$ is the reconstructed dense matrix of ibmq_santiago. $|111\rangle$ is the desired state. The error bars represent the standard errors (one side) of the samples. (a)–(c) Real parts of the theory dense matrix when the number of AA iterations is 0, 1, and 2. (d)–(f) Test results on ibmq_bogota when the number of iterations is equal to 2 and the fidelity is 0.657 ± 0.028. (g)–(i) Results on ibmq_santiago when the number of iterations is equal to 2 and the fidelity is 0.753 ± 0.017.

construct the oracle operator of QPE by

$$\begin{aligned} \boldsymbol{U} &= \boldsymbol{U}_1(-\pi) \otimes \boldsymbol{U}_1(\pi) = \begin{pmatrix} \mathrm{e}^{0} & 0 \\ 0 & \mathrm{e}^{-\mathrm{i}\pi} \end{pmatrix} \otimes \begin{pmatrix} \mathrm{e}^{0} & 0 \\ 0 & \mathrm{e}^{\mathrm{i}\pi} \end{pmatrix} \\ &= \begin{pmatrix} \mathrm{e}^{0} & 0 & 0 & 0 \\ 0 & \mathrm{e}^{\mathrm{i}\pi} & 0 & 0 \\ 0 & 0 & \mathrm{e}^{-\mathrm{i}\pi} & 0 \\ 0 & 0 & 0 & \mathrm{e}^{0} \end{pmatrix} = \begin{pmatrix} \mathrm{e}^{2\pi \mathrm{i}\tilde{w}}\boldsymbol{D} & 0 \\ 0 & \boldsymbol{D} \end{pmatrix}. \end{aligned} \tag{23}$$

Since $q1$ is $|1\rangle$ during the entire QPE, we focus only on the bottom right block $\boldsymbol{D}$, which generates phases 0 and 0.5. After the QPE, we have state $|\psi_2\rangle = |\mathrm{Reg1}\rangle|q_a\rangle|\mathrm{Reg0}\rangle = \sqrt{1/8}|110\rangle + \sqrt{7/8}|011\rangle$. When $|\mathrm{Reg1}\rangle = |1\rangle$, we have a phase equal to zero, and the success probability is 1/8. Next, we can apply the AA iterator to amplify the amplitude of $|1\rangle$. To demonstrate the speedup process, we obtained the

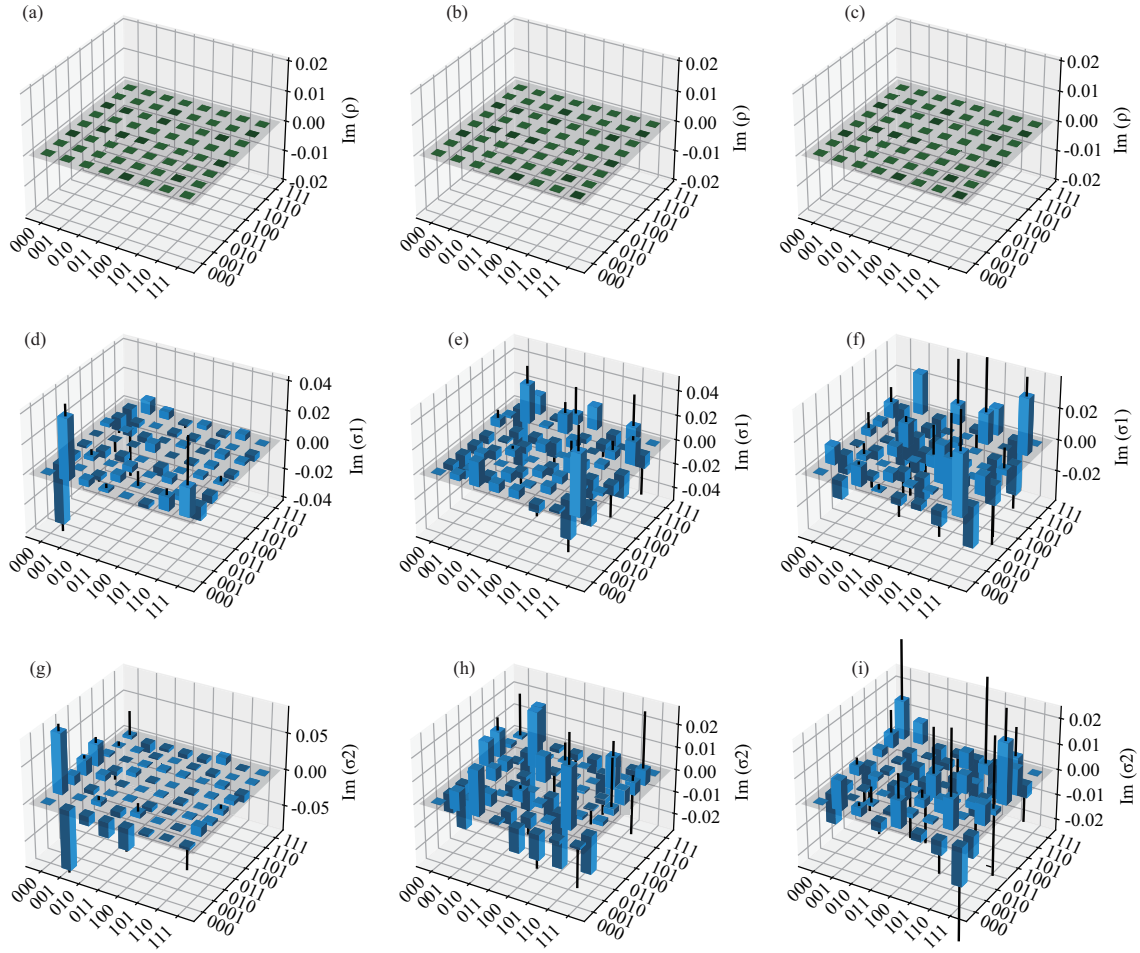

Figure 7 (Color online) ρ is dense matrix of the theory, $\sigma 1$ is the reconstructed dense matrix of ibmq_bogota, and $\sigma 2$ is the reconstructed dense matrix of ibmq_santiago. The error bars represent the standard errors (one side) of the samples. (a)–(c) Imaginary parts of the theory dense matrix when the number of AA iterations is 0, 1, and 2. (d)–(f) and (g)–(i) Results on ibmq_bogota and ibmq_santiago, respectively.

Table 4 The fidelities of the simulator, ibmq_bogata (device) and ibmq_santiago (device) under different AA iterations

	Iteration = 0	Iteration = 1	Iteration = 2	Iteration = 3
$F_{\text{simulator}}$	0.996 ± 0.004	0.997 ± 0.003	0.944 ± 0.002	0.746 ± 0.003
F_{bogota}	0.939 ± 0.009	0.761 ± 0.027	0.657 ± 0.028	0.536 ± 0.037
F_{santiago}	0.901 ± 0.027	0.856 ± 0.036	0.753 ± 0.017	0.570 ± 0.051

results and measured one qubit q_a to obtain the success probabilities when $|q_a\rangle = |1\rangle$ after each iteration. For each circuit, we repeated the measurement 1024 times. Each experiment was repeated five times. Table 3 presents the success probabilities of the theory, simulator, ibmq_bogata, and ibmq_santiago after different iterations. We obtained maximum success probabilities of 0.940 ± 0.004, 0.665 ± 0.060, and 0.751 ± 0.040 on the simulator, ibmq_bogata, and ibmq_santiago when applying the AA iterator twice. In contrast, $\frac{N}{M} = 8$ applications of the iteration are required on a classical computer when using the brute force algorithm. These results experimentally demonstrate the quadratic acceleration of our algorithm.

To evaluate the generated state and demonstrate the speedup process comprehensively, we obtained the results and measured all three qubits through state tomography [42] after each iteration, and calculated the fidelity between the theoretical and experimental states. After we gather the results from the state of $|\psi_2\rangle = |\text{Reg1}\rangle|q_a\rangle|\text{Reg0}\rangle = \sqrt{1/8}|110\rangle + \sqrt{7/8}|011\rangle$ generated by QPE through

$$\begin{aligned} \boldsymbol{U}_{\text{gather}}|\psi_2\rangle &= \boldsymbol{U}_{\text{gather}}(\sqrt{1/8}|110\rangle + \sqrt{7/8}|011\rangle) \\ &= \sqrt{1/8}|111\rangle + \sqrt{7/8}|000\rangle. \end{aligned} \tag{24}$$

The good state we desired was $|111\rangle$. Figures 6 and 7 illustrate the real and imaginary parts for the reconstructed matrix of the states generated by the theory, ibmq_bogata, and ibmq_santiago after each iteration (0, 1, and 2). The value of (111, 111) in the real parts that correspond to the good state that we want to keep increasing when the number of iterations increased. This demonstrates that the amplitude algorithm was effective. Table 4 presents the fidelity calculated from the experimental results by $F = \mathrm{Tr}\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}}$, where ρ is dense matrix of the theory, and σ is the matrix reconstructed through state tomography. From the experimental results, we obtained a fidelity of 0.944 ± 0.002, 0.657 ± 0.028, and 0.753 ± 0.017 on the simulator, ibmq_bogata, and ibmq_santiago, respectively, when the number of iterations reached 2. These results indicate that the simulator performed better than the two quantum devices when the number of iterations was less than or equal to 3. If the error rates of the experimental quantum logic gates can be reduced, the fidelity of executing our algorithm on real quantum devices is expected to be further improved.

6 Conclusion and future work

For most computer scientists and mathematicians, finding an efficient way to solve NPC problems is of great value. However, it is difficult to achieve this goal due to the high computation time and space complexity required. Quantum computing has unique advantages for these problems due to quantum parallelism and superposition. On this basis, we constructed a relationship between the SSP and quantum circuit model, and then proposed a feasible quantum algorithm to solve it and obtain quadratic speedup through AA iteration. At the same time, our coding method is practical and easy to implement. To handle the SSP problem, five steps and $n + t + 1$ qubits are required for operation. To only determine whether there is a solution, we need to measure only the ancilla qubit q_a to obtain the results. To obtain the actual solutions, we need to measure Reg1 to obtain the exact index of the solution subset when the ancilla qubit is one. We observed that the quantum simulator performed better than real quantum devices. Therefore, determining how to further reduce the error rates of real quantum devices and improve the available qubits remains a vital problem for the implementation of our circuit. Our algorithm can make full use of quantum parallelism to encode, and achieves a quadratic speedup with less qubit consumption. Therefore, it can be further used in encryption [4] and resource scheduling [8].

Acknowledgements This work was supported by National Key Research and Development Program of China (Grant Nos. 2019YFA0308700, 2017YFA0303700) and National Natural Science Foundation of China (Grant No. 11690031).

References

1 Garey M R. Computers and intractability: a guide to the theory of np-completeness. Rev Esc Enferm USP, 1979, 44: 340
2 Merkle R, Hellman M. Hiding information and signatures in trapdoor knapsacks. IEEE Trans Inform Theor, 1978, 24: 525–530
3 Kate A, Goldberg I. Generalizing cryptosystems based on the subset sum problem. Int J Inf Secur, 2011, 10: 189–199
4 Murakami Y, Sakai R. Security of knapsack cryptosystem using subset-sum decision problem against alternative-solution attack. In: Proceedings of International Symposium on Information Theory and Its Applications (ISITA), 2018. 614–617
5 Murakami Y, Hamasho S, Kasahara M. A public-key cryptosystem based on decision version of subset sum problem. In: Proceedings of International Symposium on Information Theory and its Applications, 2012. 735–739
6 Glass C A, Kellerer H. Parallel machine scheduling with job assignment restrictions. Naval Res Logist, 2007, 54: 250–257
7 Guéret C, Prins C. A new lower bound for the open-shop problem. Ann Oper Res, 1999, 92: 165–183
8 Qi X T. Coordinated logistics scheduling for in-house production and outsourcing. IEEE Trans Automat Sci Eng, 2008, 5: 188–192
9 Cai L, Chan S M, Chan S O. Random separation: a new method for solving fixed-cardinality optimization problems. In: Proceedings of International Workshop on Parameterized and Exact Computation. Berlin: Springer, 2006. 239–250
10 Caro Y, Yuster R. The characterization of zero-sum (mod 2) bipartite Ramsey numbers. J Graph Theor, 1998, 29: 151–166
11 Borgwardt K H, Tremel B. The average quality of greedy-algorithms for the Subset-Sum-Maximization Problem. ZOR - Methods Model Oper Res, 1991, 35: 113–149
12 Koiliaris K, Xu C. Faster pseudopolynomial time algorithms for subset sum. ACM Trans Algorithms, 2019, 15: 1–20
13 Horowitz E, Sahni S. Computing partitions with applications to the knapsack problem. J ACM, 1974, 21: 277–292
14 Pisinger D. Linear time algorithms for knapsack problems with bounded weights. J Algorithms, 1999, 33: 1–14
15 Kellerer H, Pferschy U, Pisinger D. Multidimensional Knapsack Problems. In: Knapsack Problems. Berlin: Springer, 2004. 235–283
16 Tsai S. Fast parallel molecular solution for DNA-based computing: the 0-1 knapsack problem. In: Proceedings of International Conference on Algorithms and Architectures for Parallel Processing, 2009. 416–427
17 Nicolau J D V, Lard M, Korten T, et al. Parallel computation with molecular-motor-propelled agents in nanofabricated networks. Proc Natl Acad Sci USA, 2016, 113: 2591–2596
18 van Delft F C M J M, Ipolitti G, Nicolau J D V, et al. Something has to give: scaling combinatorial computing by biological agents exploring physical networks encoding NP-complete problems. Interface Focus, 2018, 8: 20180034
19 Henkel C V, Bäck T, Kok J N, et al. DNA computing of solutions to knapsack problems. Biosystems, 2007, 88: 156–162
20 Henkel C V, Bladergroen R S, Balog C I A, et al. Protein output for DNA computing. Nat Comput, 2005, 4: 1–10

21 Isenberg C. The soap film: an analogue computer: soap films provide a simple method of obtaining analogue solutions to some mathematical problems. Am Scientist, 1976, 64: 514–518
22 Aaronson S. Guest column: NP-complete problems and physical reality. SIGACT News, 2005, 36: 30–52
23 Wu K, García de Abajo J, Soci C, et al. An optical fiber network oracle for NP-complete problems. Light Sci Appl, 2014, 3: e147
24 Shaked N T, Messika S, Dolev S, et al. Optical solution for bounded NP-complete problems. Appl Opt, 2007, 46: 711–724
25 Caulfield H J, Dolev S. Why future supercomputing requires optics. Nat Photon, 2010, 4: 261–263
26 Xu X Y, Huang X L, Li Z M, et al. A scalable photonic computer solving the subset sum problem. Sci Adv, 2020, 6: eaay5853
27 Schroeppel R, Shamir A. A $T = O(2^{n/2})$, $S = O(2^{n/4})$ algorithm for certain NP-complete problems. SIAM J Comput, 1981, 10: 456–464
28 Becker A, Coron J S, Joux A. Improved generic algorithms for hard knapsacks. In: Proceedings of Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 2011. 364–385
29 Howgrave-Graham N, Joux A. New generic algorithms for hard knapsacks. In: Proceedings of Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 2010. 235–256
30 Childs A M, Eisenberg J M. Quantum algorithms for subset finding. 2003. ArXiv:quant-ph/0311038
31 Bernstein D J, Jeffery S, Lange T, et al. Quantum algorithms for the subset-sum problem. In: Proceedings of International Workshop on Post-Quantum Cryptography. Berlin: Springer, 2013. 16–33
32 Helm A, May A. Subset sum quantumly in 1.17^n. In: Proceedings of the 13th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2018), 2018
33 Bonnetain X, Bricout R, Schrottenloher A, et al. Improved classical and quantum algorithms for subset-sum. In: Proceedings of International Conference on the Theory and Application of Cryptology and Information Security. Cham: Springer, 2020. 633–666
34 Chang W L, Ren T T, Feng M, et al. Quantum algorithms of the subset-sum problem on a quantum computer. In: Proceedings of WASE International Conference on Information Engineering, 2009. 54–57
35 Daskin A. A quantum approach to subset-sum and similar problems. 2017. ArXiv:1707.08730
36 Aleksandrowicz G, Alexander T, Barkoutsos P, et al. Qiskit: an open-source framework for quantum computing. 2019. https://www.qiskit.org/
37 García-Pérez G, Rossi M A C, Maniscalco S. IBM Q Experience as a versatile experimental testbed for simulating open quantum systems. npj Quantum Inf, 2020, 6: 10
38 Kitaev A Y. Quantum measurements and the Abelian stabilizer problem. 1995. ArXiv:quant-ph/9511026
39 Brassard G, Hoyer P, Mosca M, et al. Quantum amplitude amplification and estimation. Contemporary Math, 2002, 305: 53–74
40 Grover L K. A fast quantum mechanical algorithm for database search. In: Proceedings of the 28th Annual ACM Symposium on Theory of Computing, Philadelphia, 1996. 212–219
41 Cross A W, Bishop L S, Sheldon S, et al. Validating quantum computers using randomized model circuits. Phys Rev A, 2019, 100: 032328
42 Altepeter J B, Jeffrey E R, Kwiat P G. Photonic state tomography. Adv Atom Mol Opt Phys, 2005, 52: 105–159