

Auxiliary-qubit-free quantum approximate optimization algorithm for the minimum dominating set problem

Guanghui Li(李广辉), Xiaohui Ni(倪晓慧), Junjian Su(苏俊健), Sujuan Qin(秦素娟), Fenzhuo Guo(郭奋卓), Bingjie Xu(徐兵杰), Wei Huang(黄伟), and Fei Gao(高飞)

Citation: Chin. Phys. B, 2026, 35 (**5**): 050304. DOI: 10.1088/1674-1056/ae5052
http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

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

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)
Chin. Phys. B, 2025, 34 (**12**): 120303. DOI: 10.1088/1674-1056/addcc2

Quantum algorithm for marginal Fisher analysis

Jing Li(李静), Yanqi Song(宋燕琪), Sujuan Qin(秦素娟), Wenmin Li(李文敏), and Fei Gao(高飞)
Chin. Phys. B, 2025, 34 (**12**): 120302. DOI: 10.1088/1674-1056/addeb8

Exact quantum algorithm for unit commitment optimization based on partially connected quantum neural networks

Jian Liu(刘键), Xu Zhou(周旭), Zhuojun Zhou(周卓俊), and Le Luo(罗乐)
Chin. Phys. B, 2025, 34 (**10**): 100303. DOI: 10.1088/1674-1056/adf4aa

Progressive quantum algorithm for maximum independent set with quantum alternating operator ansatz

Xiao-Hui Ni(倪晓慧), Ling-Xiao Li(李凌霄), Yan-Qi Song(宋燕琪), Zheng-Ping Jin(金正平), Su-Juan Qin(秦素娟), and Fei Gao(高飞)
Chin. Phys. B, 2025, 34 (**7**): 070304. DOI: 10.1088/1674-1056/addd83

Distributed quantum circuit partitioning and optimization based on combined spectral clustering and search tree strategies

Zilu Chen(陈子禄), Zhijin Guan(管致锦), Shuxian Zhao(赵书娴), and Xueyun Cheng(程学云)
Chin. Phys. B, 2025, 34 (**5**): 050305. DOI: 10.1088/1674-1056/adbada

---

Auxiliary-qubit-free quantum approximate optimization algorithm for the minimum dominating set problem

Guanghui Li(李广辉)[1], Xiaohui Ni(倪晓慧)[1], Junjian Su(苏俊健)[1], Sujuan Qin(秦素娟)[1], Fenzhuo Guo(郭奋卓)[1], Bingjie Xu(徐兵杰)[2], Wei Huang(黄伟)[2,†], and Fei Gao(高飞)[1,‡]

[1] *State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing* 100876, *China*
[2] *National Key Laboratory of Security Communication, Institute of Southwestern Communication, Chengdu* 610041, *China*

(Received 23 January 2026; revised manuscript received 9 March 2026; accepted manuscript online 11 March 2026)

Quantum approximate optimization algorithm (QAOA) is a promising framework for solving combinatorial optimization problems on near-term quantum devices. One such problem is the minimum dominating set (MDS), which is known to be NP-hard. Existing QAOA algorithms for this problem typically require numerous auxiliary qubits, increasing circuit overhead and hardware requirements. In this paper, we propose an auxiliary-qubit-free QAOA algorithm based on Hamiltonian evolution (AQFH-QAOA) for the MDS problem. Unlike previous studies that require numerous auxiliary qubits, our algorithm eliminates the need for auxiliary qubits, thereby significantly reducing circuit overhead. In addition, we present an auxiliary-qubit-free optimized implementation of the previously proposed Guerrero's QAOA algorithm (AQFG-QAOA) by utilizing gate decomposition techniques. Through a detailed analysis of gate complexity, we evaluate the applicability of these two algorithms. Numerical experiments demonstrate that our proposed algorithm achieves competitive solution quality compared with existing QAOA algorithms, making it a promising candidate for implementation on near-term quantum devices.

Keywords: quantum algorithm, quantum approximate optimization algorithm, minimum dominating set, Boolean algebra identities

PACS: 03.67.Ac, 03.67.Lx

DOI: 10.1088/1674-1056/ae5052 **CSTR:** 32038.14.CPB.ae5052

1. Introduction

Quantum algorithms offer the potential for significant computational speedups over classical algorithms by exploiting fundamental quantum mechanical principles. These algorithms have demonstrated advantages in multiple fields, including cryptanalysis,[1] mathematical optimization,[2,3] and machine learning.[4–9] However, many well-known quantum algorithms, such as Shor's algorithm,[10] Grover's algorithm,[11] and the HHL algorithm,[12] rely on fault-tolerant quantum hardware with a large number of high-fidelity qubits, which is beyond the capabilities of current quantum devices.

To circumvent these limitations, variational quantum algorithms (VQAs) have emerged as a promising framework. VQAs work by combining parameterized quantum circuits (PQCs) with classical optimization routines to iteratively minimize an objective function.[13] This hybrid quantum-classical approach is well-suited for noisy intermediate-scale quantum (NISQ) devices.[14] VQAs can be categorized according to their application domains, including the variational quantum eigensolver (VQE) for quantum chemistry[15] and the quantum approximate optimization algorithm (QAOA) for combinatorial optimization.[16]

QAOA was originally proposed for unconstrained combinatorial optimization problems (UCOPs) and has become a widely studied quantum optimization method. It encodes the objective function into an objective Hamiltonian whose ground state corresponds to the optimal solution and employs a PQC to find this state. The PQC is constructed by alternating the unitary evolution operators of the objective Hamiltonian and the mixing Hamiltonian. As a promising approach, QAOA has been successfully applied to various UCOPs, including tail assignment,[17] vehicle routing,[18] and job shop scheduling.[19] For problems with hard constraints, the general strategy of QAOA is to utilize the penalty method to incorporate the constraints as penalty terms into the objective function, thereby constructing the objective Hamiltonian.[20] This strategy penalizes infeasible solutions to enforce the hard constraints. Although infeasible solutions that violate constraints may still appear, feasible solutions usually have a higher probability in the output.[21] This extends the applicability of QAOA to a wider range of constrained combinatorial optimization problems (CCOPs).[22–25]

The minimum dominating set (MDS) problem is a representative CCOP with broad applications in fields such as wireless networks, biological networks, and social networks. Its

†Corresponding author. E-mail: huangwei096505@aliyun.com
‡Corresponding author. E-mail: gaof@bupt.edu.cn

© 2026 Chinese Physical Society and IOP Publishing Ltd. All rights, including for text and data mining, AI training, and similar technologies, are reserved.

goal is to find the smallest subset of vertices such that every vertex in the graph is either included in this subset or adjacent to at least one vertex in it.[26] As an NP-hard problem, various exact and approximate algorithms have been proposed to solve it.[27,28]

In addition to classical algorithms, recent studies have also explored the potential of QAOA for solving the MDS problem. Dinneen and Hua[29] introduced surplus variables to convert the inequality constraints of the MDS problem into equalities and then applied the penalty method to transform the problem into a UCOP, providing a quadratic unconstrained binary optimization (QUBO) model for the MDS problem. Pan and Lu[30] adopted a similar approach but refined the penalty term range and reduced the number of surplus variables for vertices with small degrees. Both methods can be directly mapped into the objective Hamiltonian to construct the corresponding quantum circuit. However, they require a large number of auxiliary qubits, with the worst-case number of qubits reaching $\mathcal{O}(n+\log_2 n)$ and $n+2m$, respectively, where n and m represent the numbers of vertices and edges. Unlike these methods, Guerrero[31] proposed a QAOA that avoids introducing surplus variables by constructing two clauses to encode the objective function and hard constraints of the MDS problem. However, this algorithm requires multiple high-level multi-OR-controlled phase gates, whose worst-case decomposition requires n auxiliary qubits, posing a challenge to the limited qubit resources of current NISQ devices.

In this study, we propose an auxiliary-qubit-free QAOA algorithm based on Hamiltonian evolution (AQFH-QAOA) for the MDS problem, which is specifically designed to address the qubit limitations of NISQ devices. Different from previous algorithms, the proposed algorithm transforms the inequality constraints of the MDS problem into equalities via Boolean algebra identities, thereby eliminating the need for auxiliary qubits in the constructed quantum circuit. Furthermore, we provide an auxiliary-qubit-free optimized implementation of Guerrero's QAOA (AQFG-QAOA) by exploiting gate decomposition techniques. Subsequently, we analyze and compare the gate complexity of both algorithms and discuss their respective applicability on representative 3-regular graphs and Erdös–Rényi (ER) random graphs. The results show that for graphs with a smaller average degree, such as 3-regular graphs, the AQFH-QAOA algorithm is preferable due to its lower gate complexity. For ER random graphs with an edge probability of 0.5, the preferable algorithm depends on the problem size. Numerical experiments demonstrate that the proposed algorithm achieves competitive solution quality compared with existing QAOA algorithms. Additionally, an ablation study based on multi-angle QAOA indicates that the performance of the AQFH-QAOA algorithm can be further improved by replacing shared circuit parameters with independent ones.

The rest of this paper is organized as follows. In Section 2, we provide background information and a review of related work. In Section 3, we present our proposed AQFH-QAOA algorithm for the MDS problem. In Section 4, we introduce the AQFG-QAOA algorithm and perform an algorithm comparison. In Section 5, we conduct numerical simulations and perform an ablation study to explore performance improvements. Finally, we conclude this work and provide prospects for future research in Section 6.

2. Background

In this section, we provide background information related to this study in three parts. First, we present the definition of the MDS problem. Second, we provide a brief introduction to the QAOA. Finally, we review existing research related to this work, focusing on three representative algorithms.

2.1. Introduction to the MDS problem

Given an undirected graph $G=(V,E)$, where V is the set of vertices and E is the set of edges, a subset $D\subseteq V$ is called a dominating set if every vertex $v\in V$ is either in D or adjacent to at least one vertex in D. The goal of the MDS problem is to find a dominating set with the smallest number of vertices.

Let $x_i\in\{0,1\}$ be a binary variable associated with each vertex $v_i\in V$, where $x_i=1$ if the vertex v_i is included in the dominating set and $x_i=0$ otherwise. $N(v_i)$ represents the set of neighbors of vertex v_i. The MDS problem can be formulated as follows:

$$\min_{x_i\in\{0,1\}} \quad \sum_{v_i\in V} x_i, \tag{1}$$

$$\text{s.t.} \quad x_i+\sum_{v_j\in N(v_i)} x_j\geq 1, \quad \forall v_i\in V. \tag{2}$$

2.2. Introduction to QAOA

The QAOA is a VQA designed to tackle combinatorial optimization problems on near-term quantum devices. It was first proposed by Farhi *et al.*[16] as a promising approach for finding approximate solutions to optimization problems. QAOA encodes an optimization problem as a quantum Hamiltonian whose ground state corresponds to the optimal solution. The central idea is to construct a PQC to prepare this ground state through a set of adjustable parameters. The algorithm starts from the ground state of an initial Hamiltonian and evolves the quantum state through an alternating sequence of two parameterized unitary operators. One of the operators is derived from the objective function of the problem and is referred to as the objective Hamiltonian. The other is called the mixing Hamiltonian, which promotes exploration of the solution space.

Formally, let $f(x)$ be an objective function, where $x \in \{0,1\}^{\otimes n}$ encodes a candidate solution. QAOA starts from an initial state $|\psi_0\rangle$, which is typically set to a uniform superposition of all computational basis states, i.e., $|+\rangle^{\otimes n}$. For a given number of circuit layers p, the PQC prepares the quantum state

$$\left|\psi_{\mathrm{p}}(\gamma,\beta)\right\rangle = U_{\mathrm{M}}\left(\beta_{\mathrm{p}}\right)U_{\mathrm{P}}\left(\gamma_{\mathrm{p}}\right)\cdots U_{\mathrm{M}}\left(\beta_1\right)U_{\mathrm{P}}\left(\gamma_1\right)\left|\psi_0\right\rangle, \quad (3)$$

where $U_{\mathrm{P}}(\gamma) = \mathrm{e}^{-\mathrm{i}\gamma H_{\mathrm{P}}}$ is the phase-separation operator, i.e., the unitary evolution operator of the objective Hamiltonian H_{P}, and $U_{\mathrm{M}}(\beta) = \mathrm{e}^{-\mathrm{i}\beta H_{\mathrm{M}}}$ is the mixing operator, i.e., the unitary evolution operator of the mixing Hamiltonian H_{M}. Typically, $H_{\mathrm{M}} = \sum_{i=1}^{n}\sigma_i^x$, which is the sum of the Pauli-X operators. The parameters $\boldsymbol{\gamma} = (\gamma_1,\gamma_2,\ldots,\gamma_{\mathrm{p}})$ and $\boldsymbol{\beta} = (\beta_1,\beta_2,\ldots,\beta_{\mathrm{p}})$ are optimized using a classical optimizer (e.g., Adam, Nelder–Mead, or COBYLA) to maximize (or minimize) the expected value

$$F_{\mathrm{p}}(\gamma,\beta) = \left\langle\psi_{\mathrm{p}}(\gamma,\beta)\right| H_{\mathrm{P}}\left|\psi_{\mathrm{p}}(\gamma,\beta)\right\rangle. \quad (4)$$

This quantum–classical hybrid loop continues until the expected value converges or the parameter optimization reaches a predefined number of iterations. The expressive power of the PQC increases with the number of layers p. In theory, sufficiently large p allows the quantum circuit to approximate the optimal solution arbitrarily well. In practice, circuit depth is constrained by hardware limitations. Despite this, even QAOA with low depth has been shown to perform competitively with classical heuristics on certain problem instances.[32]

2.3. Literature review

This subsection reviews prior work on QAOA for the MDS problem. Dinneen and Hua,[29] as well as Pan and Lu,[30] encode the MDS problem as a QUBO model but do not explicitly construct the corresponding objective Hamiltonian, although the derivation is straightforward. Nonetheless, their work provides a modeling strategy that bridges classical combinatorial optimization problems and variational quantum algorithms. Different from these two methods, Guerrero designed a QAOA algorithm for the MDS problem without using the QUBO model.[31] While this algorithm also requires auxiliary qubits, it reduces circuit overhead compared with the previous two methods. Collectively, these studies provide valuable modeling strategies and confirm the potential of QAOA for addressing the MDS problem.

2.3.1. Review of Dinneen and Hua's QAOA for the MDS problem

To solve the MDS problem using standard QAOA, it must first be reformulated as an UCOP. A common approach is the penalty method, which incorporates the problem's constraints into the objective function to penalize violations. Since the constraints in the MDS problem are inequalities, Dinneen and Hua[29] introduced surplus variables to convert them into equalities and then derived a QUBO formulation for the MDS problem, making it compatible with standard QAOA.

The objective function to be minimized is

$$F(x) = \sum_{v_i\in V} x_i + P\sum_{v_i\in V} p_i, \quad (5)$$

where

$$p_i = \left[1 - \left(x_i + \sum_{v_j\in N(v_i)} x_j\right) + \sum_{k=0}^{\lfloor\log_2 d_i\rfloor} 2^k y_{i,k}\right]^2. \quad (6)$$

Here, $x_i \in \{0,1\}$ indicates whether the vertex v_i belongs to the dominating set ($x_i = 1$) or not ($x_i = 0$), P represents the penalty coefficient, p_i represents the penalty term, $N(v_i)$ denotes the set of neighbors of vertex v_i, and d_i denotes the degree of vertex v_i. Moreover, $y_{i,k}$ represents the k-th binary surplus variable introduced for vertex v_i, where $0 \le k \le \lfloor\log_2 d_i\rfloor$ and $k \in \mathbb{Z}$.

The objective function $F(x)$ is designed to balance two goals: minimizing the number of selected vertices and penalizing vertices that violate the constraints. The first term $\sum_{v_i\in V} x_i$ encourages choosing fewer vertices, and the second term $P\sum_{v_i\in V} p_i$, scaled by a positive penalty constant $P > 1$, penalizes vertices that violate the dominating set constraint. The penalty term p_i equals zero if vertex v_i is dominated, i.e., $x_i = 1$ or at least one neighbor satisfies $x_j = 1$. If vertex v_i is not dominated, the surplus variable expression $\sum_{k=0}^{\lfloor\log_2 d_i\rfloor} 2^k y_{i,k}$ adjusts the penalty to balance the gap. This expression can represent integers exceeding d_i, ensuring that even in the extreme case where all vertices in the closed neighborhood $\{v_i\}\cup N(v_i)$ are selected, the penalty term can be offset by appropriately adjusting the value of $y_{i,k}$.

Consequently, each vertex requires $\lfloor\log_2 d_i\rfloor + 1$ surplus variables to balance the gap so that the penalty term p_i approaches zero. The total number of binary variables is

$$n + \sum_{v_i\in V}\left(\lfloor\log_2 d_i\rfloor + 1\right) \le \mathcal{O}(n + n\log_2 n). \quad (7)$$

This means that the total number of binary variables used is at most $\mathcal{O}(n + n\log_2 n)$.

The QUBO formulation can be mapped to the objective Hamiltonian of QAOA by substituting the binary variables with Pauli-Z operators. This mapping allows the quantum state to encode candidate solutions and evolve toward the low-energy configuration corresponding to the optimal solution of the minimum dominating set. Using the same initial state and mixing operators as in standard QAOA, the QAOA for solving the MDS problem can be constructed.

2.3.2. Review of Pan and Lu's QAOA for the MDS problem

Inspired by Krauss *et al.*'s[33] use of quantum annealing to solve the maximum flow problem in QUBO form, Pan and Lu[30] proposed an improved QUBO formulation for the MDS problem. Compared with Dinneen and Hua's model, their approach adjusts the coefficient of the last surplus variable in the surplus variable expression in Eq. (6) to correct its upper bound and refine its range. Additionally, they reduce surplus variable usage by applying the penalty term proposed by Glover *et al.*[34] for vertices with degree $d_i = 0$ and $d_i = 1$, thereby avoiding extra qubits for low-degree vertices.

Similar to Dinneen and Hua's method, they convert the inequality constraints in Eq. (2) into equalities via surplus variables and incorporate them into the objective function using the penalty method. The objective function to be minimized is

$$F(x) = \sum_{v_i \in V} x_i + P \sum_{v_i \in V} p_i, \tag{8}$$

where the penalty term p_i depends on the degree d_i of vertex v_i. The specific forms are as follows:

(i) When $d_i = 0$, the original constraint reduces to $x_i \geq 1$, and the corresponding penalty term is given by

$$p_i = (1 - x_i)^2. \tag{9}$$

(ii) When $d_i = 1$, the original constraint becomes $x_i + x_j \geq 1$, where v_j is the sole neighbor of vertex v_i. Following the method proposed in Ref. [34], the penalty term can be expressed as

$$p_i = [1 - (x_i + x_j) + x_i x_j]^2. \tag{10}$$

(iii) When $d_i \geq 2$, surplus variables are introduced to convert the inequality constraints into equalities. Let the surplus variable expression be denoted by S. For vertices with $d_i \geq 2$, the inequality constraint is transformed into $x_i + \sum_{v_j \in N(v_i)} x_j - S = 1$. Accordingly, the penalty term becomes

$$p_i = \left[1 - \left(x_i + \sum_{v_j \in N(v_i)} x_j\right) + S\right]^2. \tag{11}$$

The expression

$$S = \sum_{k=0}^{\lfloor \log_2 d_i \rfloor - 1} 2^k y_{i,k} + \left(d_i + 1 - 2^{\lfloor \log_2 d_i \rfloor}\right) y_{i,\lfloor \log_2 d_i \rfloor},$$

where $y_{i,k} \in \{0,1\}$ are auxiliary variables used to represent integers up to d_i, thereby offsetting the gap in the original constraints.

To summarize, vertices with $d_i < 2$ do not require any surplus variables, whereas those with $d_i \geq 2$ require $\lfloor \log_2 d_i \rfloor + 1$ surplus variables. Thus, the total number of binary variables is less than $n + \sum_{v_i \in V} (\lfloor \log_2 d_i \rfloor + 1)$. Since

$$n + \sum_{v_i \in V} (\lfloor \log_2 d_i \rfloor + 1) \leq n + \sum_{v_i \in V} d_i = n + 2m, \tag{12}$$

where n and m are the numbers of vertices and edges, respectively, the total number of binary variables in the worst case of this QUBO formulation is $n + 2m$. The QUBO formulation is mapped to an objective Hamiltonian and combined with the standard mixing Hamiltonian to construct a QAOA circuit, which requires at most $n + 2m$ qubits.

2.3.3. Review of Guerrero's QAOA for the MDS problem

Given a bitstring $x = x_0 x_1 \cdots x_{n-1}$ representing n vertices of a graph, each bit $x_i \in \{0,1\}$ indicates whether the vertex v_i is included in the dominating set ($x_i = 1$) or not ($x_i = 0$). Guerrero proposed the following objective function to be maximized:[31]

$$C(x) = \sum_{k=0}^{n-1} (T_k(x) + D_k(x)) \tag{13}$$

where

$$T_k(x) = \begin{cases} 1, \text{ if the } k\text{-th vertex is connected to some} \\ \quad i\text{-th vertex where } x_i = 1, \\ 0, \text{ otherwise,} \end{cases} \tag{14}$$

and

$$D_k(x) = \begin{cases} 1, \text{ if } x_k = 0, \\ 0, \text{ if } x_k = 1. \end{cases} \tag{15}$$

Here, the $T_k(x)$ clause indicates whether a vertex v_k is dominated, so the term $\sum_{k=0}^{n-1} T_k(x)$ counts the total number of vertices dominated by the selected set. The $D_k(x)$ clause indicates whether a vertex v_k is excluded from the dominating set, so the term $\sum_{k=0}^{n-1} D_k(x)$ measures the size of the non-dominating set. Maximizing the term $\sum_{k=0}^{n-1} T_k(x)$ ensures that as many vertices as possible are dominated, satisfying the constraint of the MDS problem. Simultaneously, maximizing the term $\sum_{k=0}^{n-1} D_k(x)$ is equivalent to minimizing the size of the dominating set. Taken together, the objective function is formulated as maximizing the sum of these two terms, which balances constraint satisfaction with minimizing the size of the dominating set, aligning with the goal of the MDS problem.

Guerrero constructed a quantum circuit to implement this objective function, introducing an auxiliary qubit that remains unchanged throughout the circuit. The $D_k(x)$ clause is implemented using a single-controlled phase gate, with the qubit corresponding to vertex v_k as the control qubit and the auxiliary qubit as the target qubit. The implementation of the $T_k(x)$ clause is more complicated, as it depends on the neighbors of vertex v_k. Specifically, each $T_k(x)$ clause is implemented through a multi-OR-controlled phase gate, where the qubits corresponding to the vertex v_k and its neighbors are control qubits and the auxiliary qubit is the target qubit. For vertices with different neighbor structures, the control qubits differ.

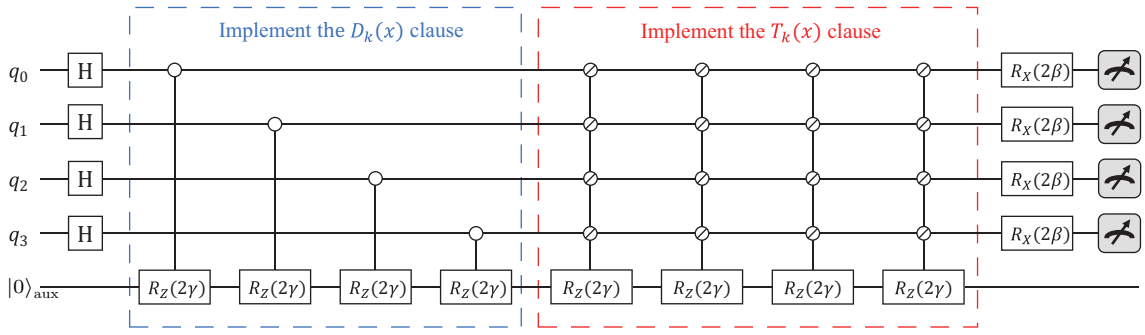


Fig. 1. A single-layer quantum circuit for solving the MDS problem on a 4-vertex 3-regular graph, constructed according to Guerrero's QAOA algorithm.[31] The qubit q_k corresponds to vertex v_k of the graph, and $|0\rangle_{\text{aux}}$ represents the auxiliary qubit. The blue dashed box represents the circuit part that implements all the $D_k(x)$ clauses, while the red dashed box indicates the circuit part that implements all the $T_k(x)$ clauses.

The mixing Hamiltonian is constructed as in standard QAOA for the MaxCut problem, i.e., $H_{\text{M}} = \sum_{i=0}^{n-1} \sigma_i^x$, where σ_i^x denotes the Pauli-X operator acting on qubit i. The corresponding unitary evolution is given by $U_{\text{M}}(\beta) = \mathrm{e}^{-\mathrm{i}\beta H_{\text{M}}} = \prod_{i=0}^{n-1} R_X^{(i)}(2\beta)$, where $R_X^{(i)}(2\beta)$ represents a rotation operator around the x-axis. The initial state $|\psi_0\rangle$ is prepared as an equal superposition of all computational basis states over the n qubits, which can be achieved by applying Hadamard gates, i.e., $|\psi_0\rangle = H^{\otimes n}|0\rangle^{\otimes n} = |+\rangle^{\otimes n}$. Both the $R_X^{(i)}(2\beta)$ gates and the Hadamard gates are only applied to the qubits corresponding to vertices, excluding the auxiliary qubit. Figure 1 illustrates the quantum circuit constructed according to this algorithm for the 4-vertex 3-regular graph. All vertices in this graph share the same set of neighbors, so the quantum gate that implements the $T_k(x)$ clause is identical for each vertex.

Appendix A presents Guerrero's decomposition method for the multi-OR-controlled phase gate and single-controlled phase gate. According to this decomposition, implementing a multi-OR-controlled phase gate generally requires additional auxiliary qubits. In the worst case, apart from the one auxiliary qubit introduced previously for the $D_k(x)$ and $T_k(x)$ clauses, implementing the $T_k(x)$ clause corresponding to a vertex of degree $n-1$ requires $n-1$ additional auxiliary qubits. In total, the number of auxiliary qubits can reach n. Consequently, the overall qubit requirement of Guerrero's QAOA is at most $2n$.

2.3.4. Summary of the number of qubits required for existing algorithms

We summarize the number of qubits required by existing algorithms for the MDS problem in Table 1, where n and m represent the number of vertices and edges, respectively. In all cases, the quantum circuits of these algorithms contain qubits corresponding to the vertices in the graph, while the number of auxiliary qubits used is related to the number of neighbors of each vertex. Therefore, based on the original literature, we summarize the qubit requirements of these algorithms in the worst-case scenario.

Table 1. The qubit number in the worst-case scenario of existing algorithms for solving the MDS problem.

Algorithm	Dinneen and Hua's QAOA	Pan and Lu's QAOA	Guerrero's QAOA
Number of auxiliary qubits	$\mathcal{O}(n\log_2 n)$	$2m$	n
Total number of qubits	$\mathcal{O}(n+n\log_2 n)$	$n+2m$	$2n$

3. Auxiliary-qubit-free QAOA algorithm based on Hamiltonian evolution

In this section, we introduce the AQFH-QAOA for solving the MDS problem. We first reformulate the integer programming model of the MDS problem and then construct the corresponding objective Hamiltonian. Using this objective Hamiltonian together with the standard mixing Hamiltonian, we design a PQC that implements the corresponding unitary evolutions. Finally, we summarize the overall algorithm procedure using pseudocode and analyze the gate complexity of the proposed algorithm.

3.1. Transformation of integer programming model

Consider an undirected graph $G = (V, E)$. Let $N(v_i)$ denote the set of neighbors of vertex $v_i \in V$, and let $x_i \in \{0,1\}$ be the binary decision variable associated with vertex v_i, where $x_i = 1$ if vertex v_i belongs to the dominating set D, and $x_i = 0$ otherwise. The integer programming model of the MDS problem is given in Subsection 2.1. However, since the constraints are inequalities, this model is not compatible with the standard QAOA framework.

To address this issue, we use Boolean algebra formulations to transform the inequality constraints and reformulate the integer programming model. Inspired by the two clauses proposed by Guerrero,[31] the model can be equivalently rewritten in the following form:

$$\max_{x_i\in\{0,1\}} \quad \sum_{v_i\in V}(1-x_i), \tag{16}$$

$$\text{s.t.} \quad x_i \vee \left(\bigvee_{v_j\in N(v_i)} x_j\right) = 1, \quad \forall\, v_i \in V. \tag{17}$$

Equation (16) reformulates the objective function expressed in Eq. (1) by converting the problem of minimizing the size of the dominating set into the equivalent task of maximizing the size of the non-dominating set. Meanwhile, Eq. (17) rewrites the inequality constraints in Eq. (2) as equality constraints using Boolean algebra.

The Boolean expression $x_i \vee \left(\bigvee_{v_j \in N(v_i)} x_j\right)$ evaluates to 1 if vertex v_i is dominated by itself or by at least one of its neighbors, and 0 otherwise. Since the maximum value of this expression is 1, the constraint in Eq. (17) can be equivalently enforced by maximizing its value. Accordingly, the integer programming model of the MDS problem is further expressed as follows:

$$\max_{x_i \in \{0,1\}} \quad \sum_{v_i \in V} (1 - x_i), \tag{18}$$

$$\max_{x_i \in \{0,1\}} \quad \sum_{v_i \in V} \left[x_i \vee \left(\bigvee_{v_j \in N(v_i)} x_j \right) \right]. \tag{19}$$

To convert the Boolean expression in Eq. (19) into an arithmetic expression suitable for Hamiltonian construction, we prove the following theorem:

Theorem 1 Let $x_i \in \{0,1\}$ for all $v_i \in V$. The following identity holds:

$$x_i \vee \left(\bigvee_{v_j \in N(v_i)} x_j \right) = 1 - (1 - x_i) \prod_{v_j \in N(v_i)} (1 - x_j). \tag{20}$$

Proof

We begin with a basic Boolean algebra identity: for any $a, b \in \{0,1\}$, the OR operation satisfies

$$a \vee b = 1 - (1 - a)(1 - b). \tag{21}$$

This identity can be verified by checking all possible values of a and b. By induction, it generalizes to any finite disjunction:

$$\bigvee_{k=1}^{m} x_k = 1 - \prod_{k=1}^{m} (1 - x_k), \tag{22}$$

where each $x_k \in \{0,1\}$.

Now, let us define $S_i = \{x_j \mid v_j \in N(v_i)\}$ to denote the set of variables corresponding to the neighbors of vertex v_i. Then the Boolean expression $x_i \vee \left(\bigvee_{x_j \in S_i} x_j\right)$ is equivalent to the disjunction over the set $\{x_i\} \cup S_i$. By applying the generalized OR identity, we obtain

$$x_i \vee \left(\bigvee_{x_j \in S_i} x_j \right) = 1 - \prod_{x \in \{x_i\} \cup S_i} (1 - x). \tag{23}$$

The right-hand side of Eq. (23) can be rewritten as $1 - (1 - x_i) \prod_{x_j \in S_i} (1 - x_j)$, which completes the proof.

Using the identity in Theorem 1, the objective function of the MDS problem can be reformulated as

$$\max_{x_i \in \{0,1\}} \quad \sum_{v_i \in V} (1 - x_i) + \lambda \sum_{v_i \in V} \left[1 - (1 - x_i) \prod_{v_j \in N(v_i)} (1 - x_j) \right], \tag{24}$$

where the first term encodes the goal of maximizing the size of the non-dominating set, and the second term introduces a penalty for non-dominated vertices. The penalty coefficient $\lambda > 1$ is typically chosen empirically to strike a balance between constraint enforcement and objective optimization.

3.2. PQC construction

In this section, we construct the PQC for the AQFH-QAOA algorithm, which consists of three fundamental components: the initial state, the phase separation operator, and the mixing operator.

3.2.1. The initial state

The initial state is set to the uniform superposition over all computational basis states

$$|\psi_0\rangle = |+\rangle^{\otimes n} = \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} \right)^{\otimes n}. \tag{25}$$

This state is prepared by applying Hadamard gates to $|0\rangle^{\otimes n}$, providing an unbiased starting point for the subsequent variational optimization.

3.2.2. The phase separation operator

The phase separation operator $U_{\mathrm{P}}(\gamma) = \mathrm{e}^{-\mathrm{i}\gamma H_{\mathrm{P}}}$ is defined as the unitary evolution generated by the objective Hamiltonian H_{P}. The objective Hamiltonian H_{P} is derived from the transformed objective function of the MDS problem, which encodes both the objective and the constraints of the MDS problem.

Consider an undirected graph $G = (V, E)$, where $V = \{v_0, v_1, \ldots, v_{n-1}\}$. Suppose that the neighbors of vertex v_i are $v_p, v_{p+1}, \ldots, v_q$. The objective function to be maximized in Eq. (24) can be equivalently reformulated as a minimization problem as follows:

$$\min_{x_i \in \{0,1\}} -\sum_{i=0}^{n-1} (1 - x_i) - \lambda \sum_{i=0}^{n-1} \left[1 - (1 - x_i) \prod_{j=p}^{q} (1 - x_j) \right]. \tag{26}$$

Using the standard mapping $x_i = (I - \sigma_i^z)/2$,[35] the binary variable x_i is replaced by $(I - \sigma_i^z)/2$, where σ_i^z is the Pauli-Z operator acting on qubit i. The objective Hamiltonian is then expressed as

$$H_{\mathrm{P}} = -\sum_{i=0}^{n-1} \left(I - \frac{I - \sigma_i^z}{2} \right)$$

$$
\begin{aligned}
&-\lambda\sum_{i=0}^{n-1}\left[I-\left(I-\frac{I-\sigma_i^z}{2}\right)\prod_{j=p}^{q}\left(I-\frac{I-\sigma_j^z}{2}\right)\right]\\
&=-\sum_{i=0}^{n-1}\frac{I+\sigma_i^z}{2}-\lambda\sum_{i=0}^{n-1}\left(I-\frac{I+\sigma_i^z}{2}\prod_{j=p}^{q}\frac{I+\sigma_j^z}{2}\right)\\
&=-\frac{n}{2}I-\frac{1}{2}\sum_{i=0}^{n-1}\sigma_i^z-\lambda nI+\frac{\lambda}{2}\sum_{i=0}^{n-1}\left[(I+\sigma_i^z)\prod_{j=p}^{q}\frac{I+\sigma_j^z}{2}\right]\\
&=-\frac{(2\lambda+1)n}{2}I-\frac{1}{2}\sum_{i=0}^{n-1}\sigma_i^z\\
&\quad+\frac{\lambda}{2}\sum_{i=0}^{n-1}\left[\frac{1}{2^{q-p+1}}(I+\sigma_i^z)\prod_{j=p}^{q}\left(I+\sigma_j^z\right)\right].
\end{aligned}
\tag{27}
$$

By expanding the product terms and combining like terms, the objective Hamiltonian can be further simplified as

$$
\begin{aligned}
H_{\mathrm{P}} &= -\frac{(2\lambda+1)n}{2}I-\frac{1}{2}\sum_{i=0}^{n-1}\sigma_i^z+\frac{\lambda}{2}\sum_{i=0}^{n-1}\left[\frac{1}{2^{q-p+1}}\left(I+\sigma_i^z+\sum_{j=p}^{q}\sigma_j^z+\sum_{j=p}^{q}\sigma_i^z\sigma_j^z\right.\right.\\
&\quad\left.\left.+\sum_{p\le j<k\le q}\sigma_j^z\sigma_k^z+\sum_{p\le j<k\le q}\sigma_i^z\sigma_j^z\sigma_k^z+\cdots+\sigma_i^z\sigma_p^z\cdots\sigma_q^z\right)\right]\\
&=-\frac{(2\lambda+1)n}{2}I+\frac{\lambda}{2}\sum_{i=0}^{n-1}\frac{1}{2^{q-p+1}}I+\frac{1}{2}\sum_{i=0}^{n-1}\left(\frac{\lambda-2^{q-p+1}}{2^{q-p+1}}\right)\sigma_i^z\\
&\quad+\frac{\lambda}{2}\sum_{i=0}^{n-1}\left[\frac{1}{2^{q-p+1}}\left(\sum_{j=p}^{q}\sigma_j^z+\sum_{j=p}^{q}\sigma_i^z\sigma_j^z+\sum_{p\le j<k\le q}\sigma_j^z\sigma_k^z+\sum_{p\le j<k\le q}\sigma_i^z\sigma_j^z\sigma_k^z+\cdots+\sigma_i^z\sigma_p^z\cdots\sigma_q^z\right)\right].
\end{aligned}
\tag{28}
$$

After neglecting the constant term in the Hamiltonian, the phase separation operator can be derived as

$$
U_{\mathrm{P}}(\gamma)=\mathrm{e}^{-\mathrm{i}\frac{\gamma}{2}\sum_{i=0}^{n-1}\left(\frac{\lambda-2^{q-p+1}}{2^{q-p+1}}\right)\sigma_i^z}\cdot\mathrm{e}^{-\mathrm{i}\frac{\gamma}{2}\lambda\sum_{i=0}^{n-1}\left[\frac{1}{2^{q-p+1}}\left(\sum_{j=p}^{q}\sigma_j^z+\sum_{j=p}^{q}\sigma_i^z\sigma_j^z+\sum_{p\le j<k\le q}\sigma_j^z\sigma_k^z+\sum_{p\le j<k\le q}\sigma_i^z\sigma_j^z\sigma_k^z+\cdots+\sigma_i^z\sigma_p^z\cdots\sigma_q^z\right)\right]}.
\tag{29}
$$

It is known that when $[A,B]=AB-BA=0$, the identity $\mathrm{e}^{A+B}=\mathrm{e}^{A}\cdot\mathrm{e}^{B}$ holds. Thus, $\mathrm{e}^{-\mathrm{i}(A+B)t}=\mathrm{e}^{-\mathrm{i}At}\cdot\mathrm{e}^{-\mathrm{i}Bt}$. Since all Pauli-$Z$ operators commute, the operator can be further factorized into the product of exponential terms as follows:

$$
\begin{aligned}
U_{\mathrm{P}}(\gamma) &= \prod_{i=0}^{n-1}\mathrm{e}^{-\mathrm{i}\frac{\gamma}{2}\left(\frac{\lambda-2^{q-p+1}}{2^{q-p+1}}\right)\sigma_i^z}\cdot\prod_{i=0}^{n-1}\left(\prod_{j=p}^{q}\mathrm{e}^{-\mathrm{i}\frac{\gamma}{2}\frac{\lambda}{2^{q-p+1}}\sigma_j^z}\cdot\prod_{j=p}^{q}\mathrm{e}^{-\mathrm{i}\frac{\gamma}{2}\frac{\lambda}{2^{q-p+1}}\sigma_i^z\sigma_j^z}\cdot\prod_{p\le j<k\le q}\mathrm{e}^{-\mathrm{i}\frac{\gamma}{2}\frac{\lambda}{2^{q-p+1}}\sigma_j^z\sigma_k^z}\right.\\
&\quad\left.\cdot\prod_{p\le j<k\le q}\mathrm{e}^{-\mathrm{i}\frac{\gamma}{2}\frac{\lambda}{2^{q-p+1}}\sigma_i^z\sigma_j^z\sigma_k^z}\cdots\mathrm{e}^{-\mathrm{i}\frac{\gamma}{2}\frac{\lambda}{2^{q-p+1}}\sigma_i^z\sigma_p^z\cdots\sigma_q^z}\right).
\end{aligned}
\tag{30}
$$

Equation (30) contains multi-qubit operators whose number grows with the number of neighbors of each vertex. By expressing these operators as quantum gates, we obtain

$$
\begin{aligned}
U_{\mathrm{P}}(\gamma) &= \prod_{i=0}^{n-1}R_Z^{(i)}\left(\frac{\lambda-2^{q-p+1}}{2^{q-p+1}}\gamma\right)\cdot\prod_{i=0}^{n-1}\left[\prod_{j=p}^{q}R_Z^{(j)}\left(\frac{\lambda}{2^{q-p+1}}\gamma\right)\cdot\prod_{j=p}^{q}R_{ZZ}^{(i,j)}\left(\frac{\lambda}{2^{q-p+1}}\gamma\right)\cdot\prod_{p\le j<k\le q}R_{ZZ}^{(j,k)}\left(\frac{\lambda}{2^{q-p+1}}\gamma\right)\right.\\
&\quad\left.\cdot\prod_{p\le j<k\le q}R_{ZZZ}^{(i,j,k)}\left(\frac{\lambda}{2^{q-p+1}}\gamma\right)\cdots R_{ZZ\cdots Z}^{(i,p,\ldots,q)}\left(\frac{\lambda}{2^{q-p+1}}\gamma\right)\right].
\end{aligned}
\tag{31}
$$

Here, $R_Z^{(i)}(\theta)=\mathrm{e}^{-\mathrm{i}\theta\sigma_i^z/2}$ denotes a single-qubit R_Z gate, $R_{ZZ}^{(j,k)}(\theta)=\mathrm{e}^{-\mathrm{i}\theta\sigma_j^z\sigma_k^z/2}$ denotes a two-qubit R_{ZZ} gate, and higher-order operators such as $R_{ZZZ}^{(j,k,l)}(\theta)$ represent multi-qubit R_{ZZZ} gates. Figure 2 illustrates the quantum gate decomposition of several operators, from which quantum gates for higher-order operators can be constructed. In fact, each term in $U_{\mathrm{P}}(\gamma)$ can be implemented using only CNOT gates and a single-qubit R_Z gate, which makes it straightforward to construct a quantum circuit for the phase separation operator $U_{\mathrm{P}}(\gamma)$. However, graphs with high connectivity produce a large number of multi-qubit operator terms, resulting in excessive circuit depth, which slows down parameter optimization and increases susceptibility to errors.

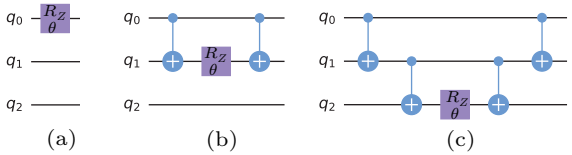


Fig. 2. Quantum circuits for (a) the single-qubit operator $\mathrm{e}^{-\mathrm{i}\theta\sigma_0^z/2}$, (b) the two-qubit operator $\mathrm{e}^{-\mathrm{i}\theta\sigma_0^z\sigma_1^z/2}$, and (c) the three-qubit operator $\mathrm{e}^{-\mathrm{i}\theta\sigma_0^z\sigma_1^z\sigma_2^z/2}$. Similarly, a k-qubit operator $\mathrm{e}^{-\mathrm{i}\theta\sigma_0^z\sigma_1^z\cdots\sigma_{k-1}^z/2}$ can be implemented using $2(k-1)$ CNOT gates together with a single-qubit R_Z gate.[35]

3.2.3. The mixing operator

The mixing operator is defined as

$$U_{\mathrm{M}}(\beta) = \mathrm{e}^{-\mathrm{i}\beta H_{\mathrm{M}}}, \tag{32}$$

where H_{M} is the mixing Hamiltonian that induces transitions between quantum states. For the MDS problem, we adopt the standard transverse-field Hamiltonian

$$H_{\mathrm{M}} = \sum_{i=0}^{n-1} \sigma_i^x, \tag{33}$$

with σ_i^x denoting the Pauli-X operator acting on qubit i.

The corresponding mixing operator can then be factorized as

$$U_{\mathrm{M}}(\beta) = \mathrm{e}^{-\mathrm{i}\beta\sum_{i=0}^{n-1}\sigma_i^x} = \prod_{i=0}^{n-1} \mathrm{e}^{-\mathrm{i}\beta\sigma_i^x} = \prod_{i=0}^{n-1} R_X^{(i)}(2\beta), \tag{34}$$

where $\beta \in [0,\pi]$ is a variational parameter. Here, each $R_X^{(i)}(2\beta)$ represents a single-qubit R_X rotation acting on qubit i, enabling bit flips between $|0\rangle$ and $|1\rangle$. This mixing operator redistributes amplitude across different computational states, which is essential for exploring the Hilbert space efficiently.

3.3. Algorithm description

This subsection presents the pseudocode and procedural details of the proposed AQFH-QAOA algorithm. The complete pseudocode is provided in Algorithm 1, which adheres to the standard QAOA framework. The algorithm proceeds as follows.

Algorithm 1 AQFH-QAOA algorithm for the MDS problem.

Input: Graph $G = (V,E)$ with $|V| = n$; Number of circuit layer p; Maximum number of iterations T.
Output: A bitstring $x^* \in \{0,1\}^n$ representing the approximate solution to the MDS problem.

1: Prepare the initial state $|\psi_0\rangle = |+\rangle^{\otimes n}$;
2: Construct the objective Hamiltonian H_{P} according to the transformed integer programming model of the MDS problem;
3: Set the mixing Hamiltonian to $H_{\mathrm{M}} = \sum_{i=0}^{n-1} \sigma_i^x$;
4: Initialize the variational parameters $\boldsymbol{\gamma} = (\gamma_1, ..., \gamma_p)$ and $\boldsymbol{\beta} = (\beta_1, ..., \beta_p)$;
5: **while** the number of iterations does not reach T **do**
6: **for** $k = 1$ to p **do**
7: Add the quantum circuit of the phase separation operator $U_{\mathrm{P}}(\gamma_k) = \mathrm{e}^{-\mathrm{i}\gamma_k H_{\mathrm{P}}}$;
8: Add the quantum circuit of the mixing operator $U_{\mathrm{M}}(\beta_k) = \mathrm{e}^{-\mathrm{i}\beta_k H_{\mathrm{M}}}$;
9: **end for**
10: Measure the final state $|\psi_p(\boldsymbol{\gamma},\boldsymbol{\beta})\rangle$ in the computational basis and evaluate the expected value $F_{\mathrm{p}}(\boldsymbol{\gamma},\boldsymbol{\beta})$;
11: Update the parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ using a classical optimizer to minimize $F_{\mathrm{p}}(\boldsymbol{\gamma},\boldsymbol{\beta})$;
12: **end while**
13: Return a bitstring $x^* = x_0^* x_1^* ... x_{n-1}^*$.

Step 1: Prepare the initial state as a uniform superposition over all computational basis states, i.e., $|\psi_0\rangle = |+\rangle^{\otimes n}$.

Step 2: Construct the objective Hamiltonian H_{P} based on the transformed integer programming model of the MDS problem given in Subsection 3.1.

Step 3: Set the mixing Hamiltonian to the standard transverse-field Hamiltonian $H_{\mathrm{M}} = \sum_{i=0}^{n-1} \sigma_i^x$;

Step 4: Initialize the variational parameters $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_p)$ and $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_p)$, with $\gamma_k \in [0, 2\pi]$ and $\beta_k \in [0,\pi]$ for $k = 1, 2, \ldots, p$.

Step 5: For each layer $k = 1, 2, \ldots, p$, alternately apply the phase separation operator $U_{\mathrm{P}}(\gamma_k) = \mathrm{e}^{-\mathrm{i}\gamma_k H_{\mathrm{P}}}$ and the mixing operator $U_{\mathrm{M}}(\beta_k) = \mathrm{e}^{-\mathrm{i}\beta_k H_{\mathrm{M}}}$ to construct the quantum state $|\psi(\boldsymbol{\gamma},\boldsymbol{\beta})\rangle = \left(\prod_{k=1}^{p} U_{\mathrm{M}}(\beta_k) U_{\mathrm{P}}(\gamma_k)\right)|\psi_0\rangle$;

Step 6: Measure the final state $|\psi_p(\boldsymbol{\gamma},\boldsymbol{\beta})\rangle$ of the quantum circuit in the computational basis to obtain a bitstring $\tilde{x} \in \{0,1\}^n$ and evaluate the expected value $F_{\mathrm{p}}(\boldsymbol{\gamma},\boldsymbol{\beta}) = \langle\psi_p(\boldsymbol{\gamma},\boldsymbol{\beta})|H_{\mathrm{P}}|\psi_p(\boldsymbol{\gamma},\boldsymbol{\beta})\rangle$.

Step 7: Update the parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ using a classical optimizer (e.g., Adam, COBYLA, or Nelder-Mead) to minimize $F_{\mathrm{p}}(\boldsymbol{\gamma},\boldsymbol{\beta})$.

Step 8: Repeat Steps 5–7 until the maximum number of iterations T is reached.

Step 9: After the optimization is completed, output the bitstring $x^* \in \{0,1\}^n$ with the lowest cost, which approximates the solution to the MDS problem.

3.4. Gate complexity

Next, we analyze the gate complexity of the AQFH-QAOA algorithm. Consider an undirected graph $G = (V,E)$ with n vertices. While vertex degrees may vary, for simplicity we assume each vertex has d neighbors, denoted by $v_p, v_{p+1}, \ldots, v_q$, where $d = q - p + 1$.

The number of quantum gates in the circuit for the phase separation operator $U_{\mathrm{P}}(\boldsymbol{\gamma})$ can be estimated from Eq. (31). Some gates can be merged; for instance, a single-qubit $R_Z^{(j)}$ can be combined with $R_Z^{(i)}$ acting on the same qubit. Multi-qubit gates, however, generally cannot be merged, since they act on different sets of qubits. The worst-case gate count for $U_{\mathrm{P}}(\boldsymbol{\gamma})$ is therefore

$$N_{\mathrm{gate}} \le n \cdot R_Z + n \cdot \Big(C_d^1 \cdot R_{ZZ} + C_d^2 \cdot R_{ZZ} + C_d^2 \cdot R_{ZZZ} + \cdots + C_d^d \cdot R_{ZZ\cdots Z} \Big), \tag{35}$$

where we omit the angle parameters and specific qubits for clarity.

As shown in Fig. 2, a k-qubit $R_{Z\cdots Z}$ gate can be decomposed into $2(k-1)$ CNOT gates and a single-qubit R_Z gate. Using this, the worst-case number of elementary gates in $U_{\mathrm{P}}(\boldsymbol{\gamma})$ is

$$N_{\mathrm{CNOT}} \le 0 \cdot n + n \cdot \Big(2 \cdot C_d^1 + 2 \cdot C_d^2 + 4 \cdot C_d^2 + 4 \cdot C_d^3 + \cdots$$

$$+2d\cdot C_d^d\Big) = n(d-1)\cdot 2^{d+1}+2n, \tag{36}$$

$$N_{\text{Single-qubit}} \le 1\cdot n + n\cdot\Big(1\cdot C_d^1+1\cdot C_d^2+1\cdot C_d^2+1\cdot C_d^3 +1\cdot C_d^3\cdots+1\cdot C_d^d\Big) = n\cdot 2^{d+1}-n(d+1), \tag{37}$$

where C_d^k is the binomial coefficient.

Including the n Hadamard gates for preparing the initial state and the n single-qubit R_X gates of the mixing operator, a single-layer AQFH-QAOA circuit requires at most $n(d-1)\cdot 2^{d+1}+2n$ CNOT gates, and $n\cdot 2^{d+1}-n(d-1)$ single-qubit gates. This shows that the total number of gates grows polynomially with n but exponentially with the average degree d.

For comparison, Appendix A presents the gate count for Guerrero's QAOA algorithm, which requires $20nd-6n$ CNOT gates and $16nd$ single-qubit gates for a graph of n vertices and average degree d. Since CNOT gates dominate circuit depth due to slower execution and higher error rates,[36] we use N_{CNOT} as a proxy for depth.

To estimate when AQFH-QAOA loses its advantage over Guerrero's QAOA, we solve

$$n(d-1)\cdot 2^{d+1}+2n \approx 20nd-6n, \tag{38}$$

yielding $d\approx 3.62$. This implies that AQFH-QAOA is advantageous for graphs with average degree $d \lesssim 3.62$. For small d, the total gate count scales approximately as $\mathcal{O}(n)$. For example, for 3-regular graphs, a single-layer circuit requires at most $34n$ CNOT gates and $14n$ single-qubit gates.

4. Auxiliary-qubit-free optimized implementation of Guerrero's QAOA

In this section, we present an implementation of Guerrero's QAOA without using auxiliary qubits, based on gate decomposition techniques described in Subsection 4.1. To distinguish it from the original algorithm, we refer to this version as the AQFG-QAOA algorithm. After decomposing its quantum circuit into elementary gates, we compare its gate complexity with that of the AQFH-QAOA algorithm.

4.1. Decomposition of multi-OR-controlled phase gates without auxiliary qubits

In Guerrero's QAOA, apart from the single auxiliary qubit used to encode the $T_k(x)$ and $D_k(x)$ clauses, additional auxiliary qubits arise from the decomposition of multi-OR-controlled phase gates (see Appendix A).

In the AQFG-QAOA algorithm, each multi-OR-controlled phase gate can first be decomposed into a sequence of single-qubit gates and multi-controlled X (MCX) gates, as illustrated in Fig. 3. Subsequently, these MCX gates can be further decomposed into elementary gates of quadratic size without requiring any additional auxiliary qubits.[37]

Throughout this section, the term "auxiliary qubit" excludes the qubit used for encoding the $T_k(x)$ and $D_k(x)$ clauses. Accordingly, we consider an $(n+1)$-qubit circuit, where the qubit $|0\rangle_{\text{aux}}$ associated with $T_k(x)$ and $D_k(x)$ is not counted as an auxiliary qubit in our analysis.

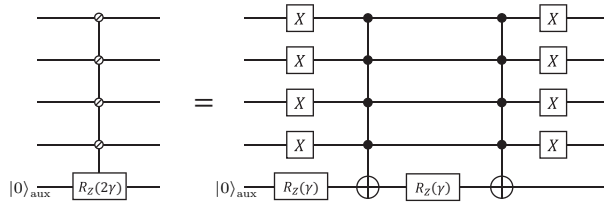


Fig. 3. Decomposition of a multi-OR-controlled phase gate with n control qubits into elementary gates without auxiliary qubits. Here, $n=4$. The decomposition uses $2n+2$ single-qubit gates and two n-controlled X (MCX) gates. The construction exploits the relations $R_Z(\theta)R_Z(\theta)=R_Z(2\theta)$, $XR_Z(\theta)X=R_Z(-\theta)$, and $R_Z(\theta)R_Z(-\theta)=I$.

Barenco *et al.*[37] pioneered a systematic method for decomposing MCX gates, as presented in Corollary 7.6. Subsequent studies[38–41] proposed alternative decompositions of MCX gates into elementary gates of quadratic size without auxiliary qubits, aiming to optimize circuit depth and size. However, most of these works do not provide precise counts of the resulting elementary gates. A notable exception is Liu *et al.* [38], which presents the exact number of CNOT and single-qubit gates obtained from MCX gate decomposition. In this work, we evaluate the number of elementary gates after multi-OR-controlled phase gate decomposition using the exact counts reported by Liu *et al.*[38]

For completeness, we briefly review the MCX decomposition from Refs. [37,38]. An MCX gate with n control qubits (denoted $C^n(X)$) can be decomposed into two $(n-1)$-controlled X gates ($C^{n-1}(X)$), one $(n-1)$-controlled $\sqrt{X}$ gate ($C^{n-1}(\sqrt{X})$), and two single-controlled gates, as illustrated in Fig. 4. Here, $\sqrt{X}$ denotes a square root of the Pauli-X operator, satisfying $\sqrt{X}^2=X$, explicitly given by $\sqrt{X}=\frac{1-i}{2}(I+iX)$. We note a difference in notation: in Refs. [37,38], $C^n(X)$ denotes a gate with $n-1$ control qubits and one target qubit, whereas in this work $C^n(X)$ denotes an $(n+1)$-qubit gate with n control qubits and one target qubit. This distinction arises because Guerrero's QAOA circuit introduces an auxiliary qubit as the target, while the qubits corresponding to a vertex and its neighbors act as controls.

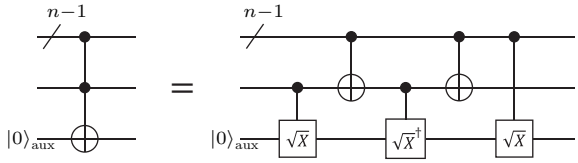


Fig. 4. Decomposition of an n-controlled X gate ($C^n(X)$) into two $(n-1)$-controlled X gates ($C^{n-1}(X)$), one $(n-1)$-controlled $\sqrt{X}$ gate ($C^{n-1}(\sqrt{X})$), and two single-controlled gates. Here, $\sqrt{X}$ denotes the square root of the Pauli-X operator, and $\sqrt{X}^\dagger$ its Hermitian conjugate. Adapted from Lemma 7.5 of Ref. [37].

The next step is to decompose the $C^{n-1}(X)$ and $C^{n-1}(\sqrt{X})$ gates generated in the previous step.

For the $C^{n-1}(X)$ gate, a naive approach is to iteratively apply the decomposition shown in Fig. 4. However, this leads to exponential growth in circuit size (Lemma 7.1 of Ref. [37]). To avoid this overhead, one can use a single borrowed auxiliary qubit to split the $C^{n-1}(X)$ gate into four half-sized MCX gates, which are then further decomposed into Toffoli gates, as illustrated in Fig. 5. Here, the auxiliary qubit is borrowed — initially in an arbitrary state and restored after the operation — so no additional qubits are needed.

Specifically, a $C^{n-1}(X)$ gate is decomposed into two m_1-controlled X gates ($C^{m_1}(X)$) and two (m_2+1)-controlled X gates ($C^{m_2+1}(X)$), where

$$m_1 = \left\lceil \frac{n}{2} \right\rceil, \quad m_2 = n - m_1 - 1.$$

Each $C^{m_1}(X)$ gate can be further decomposed into $4\lceil n/2 \rceil - 8$ Toffoli gates (Fig. 7 in Ref. [38]), while each $C^{m_2+1}(X)$ gate is decomposed into $4n - 4\lceil n/2 \rceil - 8$ Toffoli gates (Fig. 8 in Ref. [38]). Qubits not involved in the current gate serve as borrowed auxiliary qubits, and no additional auxiliary qubits are introduced. These decompositions are valid for $n \geq 6$; for $n < 6$, see Table 1 in Ref. [38].

For the $C^{n-1}(\sqrt{X})$ gate, the decomposition proceeds recursively using the identity in Fig. 4. The recursion continues $n-2$ times until the gate is expressed in terms of Toffoli gates and single-controlled gates. Toffoli gates are implemented via the congruent modulo phase shifts (CMPS) method,[37] which decomposes each Toffoli gate into 3 CNOT gates and 4 single-qubit gates, as shown in Fig. 6. Single-controlled gates are decomposed into elementary gates using the identity in Fig. 7, where $\sqrt[n]{Z}$ denotes the n-th root of the Pauli-Z operator

$$\sqrt[n]{Z} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/n} \end{bmatrix}.$$

Reference [38] presents a cascaded structure for the iterative decomposition of an arbitrary multi-controlled unitary gate, which is equivalent to decomposing a $C^n(X)$ gate when the unitary matrix is X. Using this decomposition method, when $n \geq 6$, a $C^n(X)$ gate can be decomposed into $24n^2 - 164n + 352$ CNOT gates and $32n^2 - 224n + 483$ single-qubit gates. It is important to note that the above decomposition assumes the absence of auxiliary qubits. If at least one auxiliary qubit is available, the circuit size of MCX gate decomposition can be reduced from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$.[42,43]

Combining the decomposition shown in Fig. 3 with the exact number of elementary gates mentioned above, when $n \geq 6$, a multi-OR-controlled phase gate with n control qubits can be decomposed into $48n^2 - 328n + 704$ CNOT gates and $32n^2 - 222n + 485$ single-qubit gates. For the cases $1 < n < 6$, the corresponding number of elementary gates can be calculated from Table 1 of Ref. [38]. We summarize these results in Table 2.

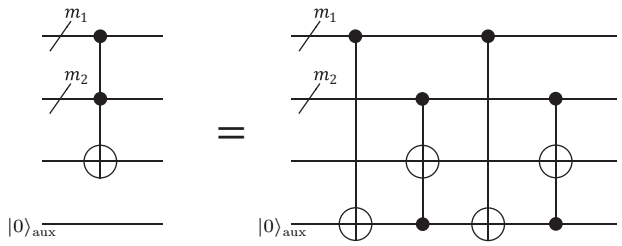


Fig. 5. Decomposition of a $C^{n-1}(X)$ gate using two m_1-controlled X gates ($C^{m_1}(X)$) and two (m_2+1)-controlled X gates ($C^{m_2+1}(X)$), where $m_1 = \lceil \frac{n}{2} \rceil$ and $m_2 = n - m_1 - 1$. Adapted from Lemma 7.3 of Ref. [37].

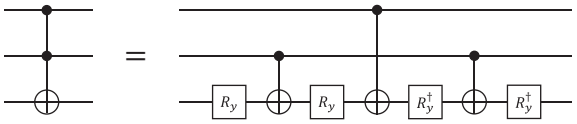


Fig. 6. Decomposition of a Toffoli gate using the CMPS method, where the angle of all R_y ($R_y^\dagger$) operators is $\pi/4$. Adapted from Ref. [37].

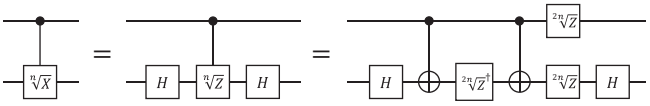


Fig. 7. Decomposition of a single-controlled gate using the circuit identity, where $\sqrt[n]{Z}$ denotes the n-th root of the Pauli-Z operator.

Table 2. Number of elementary gates after decomposing different types of multi-controlled quantum gates.

Gate type	Elementary gate	2	3	4	5	$n \geq 6$
n-controlled X gate (MCX)	CNOT	6	16	56	132	$24n^2 - 164n + 352$
	Single-qubit	8	22	71	163	$32n^2 - 224n + 483$
	Total	14	38	127	295	$56n^2 - 388n + 835$
n-OR-controlled phase gate	CNOT	12	32	112	264	$48n^2 - 328n + 704$
	Single-qubit	14	30	81	175	$32n^2 - 222n + 485$
	Total	26	62	193	439	$80n^2 - 550n + 1189$

4.2. Gate complexity

In this section, we analyze the gate complexity of the AQFG-QAOA algorithm for different graph structures. Specifically, we consider two representative graph classes: 3-regular graphs and ER random graphs with an edge probability of 0.5. In a 3-regular graph, each vertex has a degree of exactly 3, forming a sparse structure when the number of vertices n is large. In contrast, an ER graph $G(n, p_e)$ exhibits a non-

uniform degree distribution determined by the edge probability p_e.[44] For a fixed constant p_e, the expected degree of each vertex is $p_e(n-1)$, corresponding to an unstructured probabilistic connectivity model. For the MDS problem on an ER graph, the graph is extremely sparse when p_e is close to 0, making the problem relatively easy to solve. When p_e approaches 1, the graph approaches a complete graph, and the problem becomes simple again. The most challenging case occurs when p_e takes a value between 0 and 1, at which the ER graph is neither too sparse nor too dense. From an information theory perspective, the Shannon entropy quantifies the uncertainty of the edge structure in a random graph, with higher entropy indicating greater structural complexity.[45] For ER graphs, the Shannon entropy of the edge set is maximized at $p_e = 0.5$. Accordingly, in this work, we focus on ER graphs with edge probability $p_e = 0.5$.

For a 3-regular graph with n ($n \geq 4$) vertices, each vertex has 3 neighbors, plus the vertex itself, which corresponds to 4 control qubits in the multi-OR-controlled phase gate that implements the $T_k(x)$ clause. Consequently, each $T_k(x)$ clause is implemented by a 4-OR-controlled phase gate. According to Table 2, such a gate can be decomposed into 112 CNOT gates and 81 single-qubit gates. Therefore, the circuit part that implements all $T_k(x)$ clauses requires a total of $112n$ CNOT gates and $81n$ single-qubit gates. In addition, using the decomposition shown in Fig. A4, all the single-controlled phase gates that implement all $D_k(x)$ clauses can be decomposed into $2n$ CNOT gates and $4n$ single-qubit gates. Including the n Hadamard gates for preparing the initial state and the n single-qubit $R_X(2\beta)$ gates in the mixing operator, the single-layer quantum circuit of the AQFG-QAOA algorithm on a 3-regular graph with n vertices contains a total of $114n$ CNOT gates and $87n$ single-qubit gates.

Table 3. Number of elementary gates after decomposition of a single-layer quantum circuit constructed using the AQFG-QAOA algorithm.

Graph type	Graph size	Elementary gate		
		CNOT	Single-qubit	Total
n-vertex 3-regular graph	$n \geq 4$	$114n$	$87n$	$201n$
n-vertex ER graph ($p_e = 0.5$)	($n \geq 6$, odd)	$12n^3 - 140n^2 + 554n$	$8n^3 - 95n^2 + 388n$	$20n^3 - 235n^2 + 942n$
	($n \geq 6$, even)	$12n^3 - 116n^2 + 426n$	$8n^3 - 79n^2 + 301n$	$20n^3 - 195n^2 + 727n$

For an ER graph $G(n, p_e)$ with edge probability $p_e = 0.5$, the average degree of each vertex is $\frac{n-1}{2}$, meaning each vertex is connected to an average of $\lceil \frac{n-1}{2} \rceil$ other vertices. Including the vertex itself, a multi-OR-controlled phase gate that implements the $T_k(x)$ clause has $\lceil \frac{n+1}{2} \rceil$ control qubits. When $n \geq 6$ and n is odd, such a gate can be decomposed into $12n^2 - 140n + 552$ CNOT gates and $8n^2 - 95n + 382$ single-qubit gates. When $n \geq 6$ and n is even, it can be decomposed into $12n^2 - 116n + 424$ CNOT gates and $8n^2 - 79n + 295$ single-qubit gates. Consequently, for $n \geq 6$, implementing all n $T_k(x)$ clauses requires $12n^3 - 140n^2 + 552n$ CNOT gates and $8n^3 - 95n^2 + 382n$ single-qubit gates when n is odd, and $12n^3 - 116n^2 + 424n$ CNOT gates and $8n^3 - 79n^2 + 295n$ single-qubit gates when n is even. The number of elementary gates required to implement the n $D_k(x)$ clauses, as well as those used for initial state preparation and the mixing operator, is identical to the corresponding counts for 3-regular graphs. Summing all gates yields the total number of elementary gates needed on an ER graph with edge probability $p_e = 0.5$ using the AQFG-QAOA algorithm. For the cases $n < 6$, the gate counts must be calculated individually. A summary of these results is provided in Table 3.

4.3. Algorithm comparison

Here, we compare the gate complexity of the AQFG-QAOA algorithm with that of the AQFH-QAOA algorithm proposed in Section 3. The comparison is summarized in Table 4.

Table 4. Number of elementary gates after decomposition of a single-layer quantum circuit constructed using different algorithms.

Graph type	Elementary gate	Algorithm			
		AQFG-QAOA		AQFH-QAOA	
n-vertex 3-regular graph		($n \geq 4$)		($n \geq 4$)	
	CNOT	$114n$		$34n$	
	Single-qubit	$87n$		$18n$	
	Total	$201n$		$52n$	
n-vertex ER graph ($p_e = 0.5$)		($n \geq 6$, odd)	($n \geq 6$, even)	($n \geq 2$, odd)	($n \geq 2$, even)
	CNOT	$12n^3 - 140n^2 + 554n$	$12n^3 - 116n^2 + 426n$	$(n^2 - 3n) \cdot 2^{\frac{n-1}{2}} + 2n$	$(n^2 - 2n) \cdot 2^{\frac{n}{2}} + 2n$
	Single-qubit	$8n^3 - 95n^2 + 388n$	$8n^3 - 79n^2 + 301n$	$n \cdot 2^{\frac{n+1}{2}} - \frac{n^2}{2} + \frac{3n}{2}$	$2n \cdot 2^{\frac{n}{2}} - \frac{n^2}{2} + n$
	Total	$20n^3 - 235n^2 + 942n$	$20n^3 - 195n^2 + 727n$	$(n^2 - n) \cdot 2^{\frac{n-1}{2}} - \frac{n^2}{2} + \frac{7n}{2}$	$n^2 \cdot 2^{\frac{n}{2}} - \frac{n^2}{2} + 3n$

For 3-regular graphs, the AQFH-QAOA algorithm uses fewer CNOT gates and single-qubit gates than the AQFG-QAOA algorithm, suggesting that the MDS problem on 3-regular graphs is more suitable for solution using the AQFH-QAOA algorithm. For ER graphs with edge probability $p_e = 0.5$, the number of elementary gates used in the AQFG-QAOA algorithm grows polynomially with the number of vertices, whereas the gate count of the AQFH-QAOA algorithm grows exponentially. Nevertheless, the practical applicability of each algorithm can be evaluated by estimating the maximum feasible problem size based on the number of CNOT gates.

Figure 8 presents the number of CNOT gates required by the AQFG-QAOA and AQFH-QAOA algorithms for the MDS problem on ER graphs with edge probability $p_e = 0.5$ at different vertex sizes. The discrete data points correspond to the exact values computed from Table 4, while the solid curves represent fitted growth trends based on these data. As expected, the number of CNOT gates for both algorithms increases with the number of vertices n, but the growth rates differ substantially. Specifically, the number of CNOT gates in the AQFH-QAOA algorithm grows rapidly with n, exhibiting an exponential growth trend, whereas the AQFG-QAOA algorithm shows a much slower increase that is well approximated by a cubic polynomial. The fitted curves intersect at approximately $n \approx 13.41$, and the exact crossover point in terms of integer vertex size satisfies $13 < n < 14$. This indicates that for ER graphs with vertex size $n \leq 13$, the AQFH-QAOA algorithm requires fewer CNOT gates than the AQFG-QAOA algorithm, suggesting a potential advantage due to its lower circuit overhead. In contrast, for ER graphs with $n > 13$, the number of CNOT gates for the AQFG-QAOA algorithm is significantly lower than that of the AQFH-QAOA algorithm, demonstrating that the AQFG-QAOA algorithm has superior scalability.

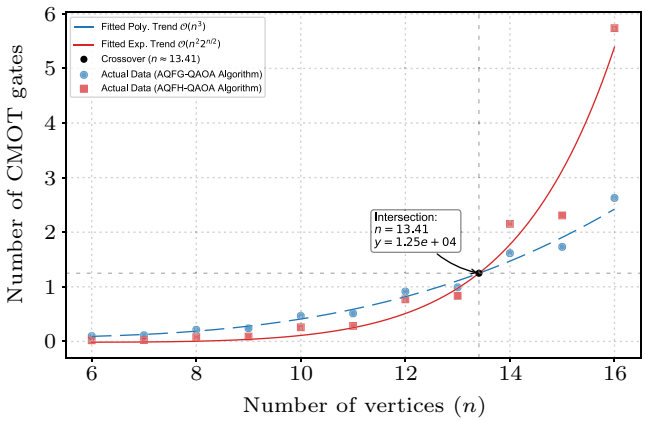


Fig. 8. Number of CNOT gates in quantum circuits constructed using different algorithms to solve the MDS problem on ER graphs with various vertex sizes. Discrete data points represent actual computed values, while the corresponding curves show fitted growth trends based on these data.

As described in Subsection 4.2, the average degree of each vertex in an n-vertex ER graph ($p_e = 0.5$) is $d = \frac{n-1}{2}$. For such graphs, the AQFH-QAOA algorithm exhibits low gate complexity when the vertex size $n \leq 13$, and the corresponding average vertex degree is $d = 6$. Therefore, for ER graphs with $d < 6$, the AQFH-QAOA algorithm maintains an advantage in gate complexity even on graphs with larger vertex sizes.

5. Numerical simulations

In this section, we perform numerical simulations using the MindSpore quantum platform.[46] We compare the performance of the AQFH-QAOA algorithm with other related algorithms reviewed in Subsection 2.3. The benchmark instances include 3-regular graphs and ER graphs with an edge probability of 0.5. Algorithmic performance is evaluated using the success probability defined in Definition 1.

Definition 1 (success probability) The success probability is defined as the probability that the final state $|\psi_p(\gamma,\beta)\rangle$ of the quantum circuit collapses to the ground state $|x^*\rangle$ of the objective Hamiltonian upon a single measurement, i.e.

$$P_{\text{suc}} = |\langle x^*|\psi_p(\gamma,\beta)\rangle|^2. \tag{39}$$

The success probability quantifies the overlap between the final state $|\psi_p(\gamma,\beta)\rangle$ and the ground state $|x^*\rangle$. A larger value of P_{suc} indicates better algorithmic performance and a higher likelihood of obtaining a high-quality solution.

5.1. Sensitivity analysis of the penalty coefficient

In the AQFH-QAOA algorithm, the choice of the penalty coefficient λ is crucial for balancing constraint enforcement and objective optimization. If λ is too small, the penalty term may fail to suppress infeasible solutions that violate the dominance constraints. Conversely, excessively large values of λ may distort the optimization landscape and make the classical optimization process more difficult. Therefore, it is important to investigate the influence of the penalty coefficient λ on algorithm performance.

To this end, we perform numerical experiments by varying the penalty coefficient over the set $\lambda \in \{1.0, 1.2, 1.5, 2.0, 3.0, 5.0\}$, with $\lambda = 1.0$ used as the baseline. Two types of graph instances are considered: a 6-vertex 3-regular graph and a set of 20 Erdös–Rényi graphs with six vertices. For the ER graphs, we report the average success probability, defined as the mean of the success probabilities over all graph instances with the same number of vertices.

Figure 9(a) shows the success probability of the AQFH-QAOA algorithm on a 6-vertex 3-regular graph for different values of λ. For all tested penalty coefficients, the success probability increases steadily with the circuit layer, indicating that deeper circuits generally improve the solution quality. However, noticeable differences appear among different values of λ. In particular, when $\lambda = 5.0$, the success probability is consistently lower than that obtained with the other choices for small and moderate circuit depths, suggesting that

an excessively large penalty coefficient may reduce the effectiveness of the optimization. In contrast, moderate values such as $\lambda = 1.5$ and $\lambda = 2.0$ achieve higher success probabilities across most circuit layers. When the circuit layer becomes sufficiently large (e.g., $p = 7$), the performance differences among most penalty coefficients diminish, and the success probabilities converge to similar values.

Figure 9(b) presents the average success probability over 20 ER graphs. A similar trend is observed: the success probability increases monotonically with the circuit layer for all tested values of λ. For smaller penalty coefficients ($\lambda = 1.2$), performance is slightly lower than that of moderate values such as $\lambda = 1.5$ and $\lambda = 2.0$. Conversely, when λ is too large (e.g., $\lambda = 5.0$), the algorithm exhibits reduced performance across different circuit layers. Overall, these results indicate that moderate penalty coefficients provide better and more stable performance across different graph instances.

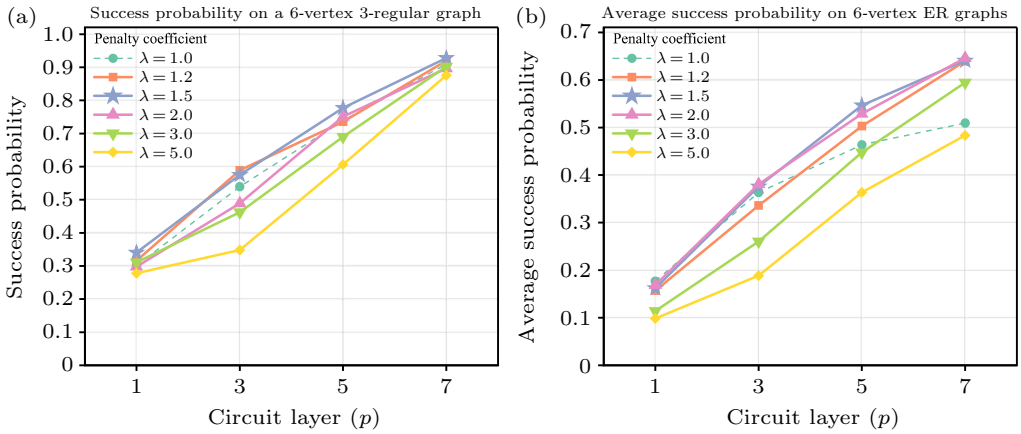


Fig. 9. Success probabilities for different penalty coefficients on two types of graphs. Subfigure (a) shows the success probability on a 6-vertex 3-regular graph, while subfigure (b) shows the average success probability over 20 ER graphs.

These observations suggest that the AQFH-QAOA algorithm is relatively robust with respect to the choice of λ within a reasonable range. In particular, moderate values of λ (around $\lambda \approx 1.5$–2.0) tend to achieve the best performance for the tested graph instances, while excessively small or large values may lead to slightly degraded results. In the subsequent simulations, we adopt a fixed penalty coefficient $\lambda = 1.5$.

5.2. Performance comparison of related algorithms

We perform numerical simulations on randomly generated 3-regular graphs and ER graphs with an edge probability of 0.5, comparing the proposed algorithm with existing approaches. The algorithms proposed by Dinneen *et al.* and Pan *et al.* require a large number of auxiliary qubits. For example, both algorithms require 12 auxiliary qubits (18 qubits in total) to solve the MDS problem on a 6-vertex 3-regular graph, with even more qubits needed for larger graphs or higher edge densities. Consequently, due to classical simulation limitations, these two algorithms can only be evaluated on graphs with 4 and 6 vertices in our experiments.

Figure 10 demonstrates that the AQFH-QAOA algorithm consistently achieves higher success probabilities than the algorithms of Dinneen and Hua,[29] and Pan and Lu[30] across all tested graph instances. Moreover, as the number of circuit layers p increases, the success probability of the AQFH-QAOA algorithm improves significantly, whereas the success probabilities of the other two algorithms either fluctuate or increase only slightly, leading to an increasingly pronounced performance gap.

Compared with the AQFG-QAOA algorithm, the AQFH-QAOA algorithm generally exhibits superior performance, with the advantage becoming more pronounced as p increases. However, for the 6-vertex 3-regular graph, it performs slightly worse than the AQFG-QAOA algorithm. This behavior can be attributed to circuit depth. Although the quantum circuits of the AQFG-QAOA algorithm contain more elementary gates on small graphs, classical numerical simulations do not decompose the multi-OR-controlled phase gates due to the limitations of the simulation platform. In contrast, the quantum circuits of the AQFH-QAOA algorithm are implemented using elementary gates through the unitary evolution of Hamiltonians. This results in a significantly smaller actual circuit depth for the AQFG-QAOA algorithm compared to the AQFH-QAOA algorithm for the same problem instances, which can lead to improved algorithmic performance in simulation.

Overall, these results indicate that the AQFH-QAOA algorithm significantly outperforms the QAOA algorithms of Dinneen and Hua,[29] and Pan and Lu.[30] However, due to the absence of multi-OR-controlled phase gate decomposition in the simulations, its performance is comparable to or slightly worse than that of the AQFG-QAOA algorithm in some cases.

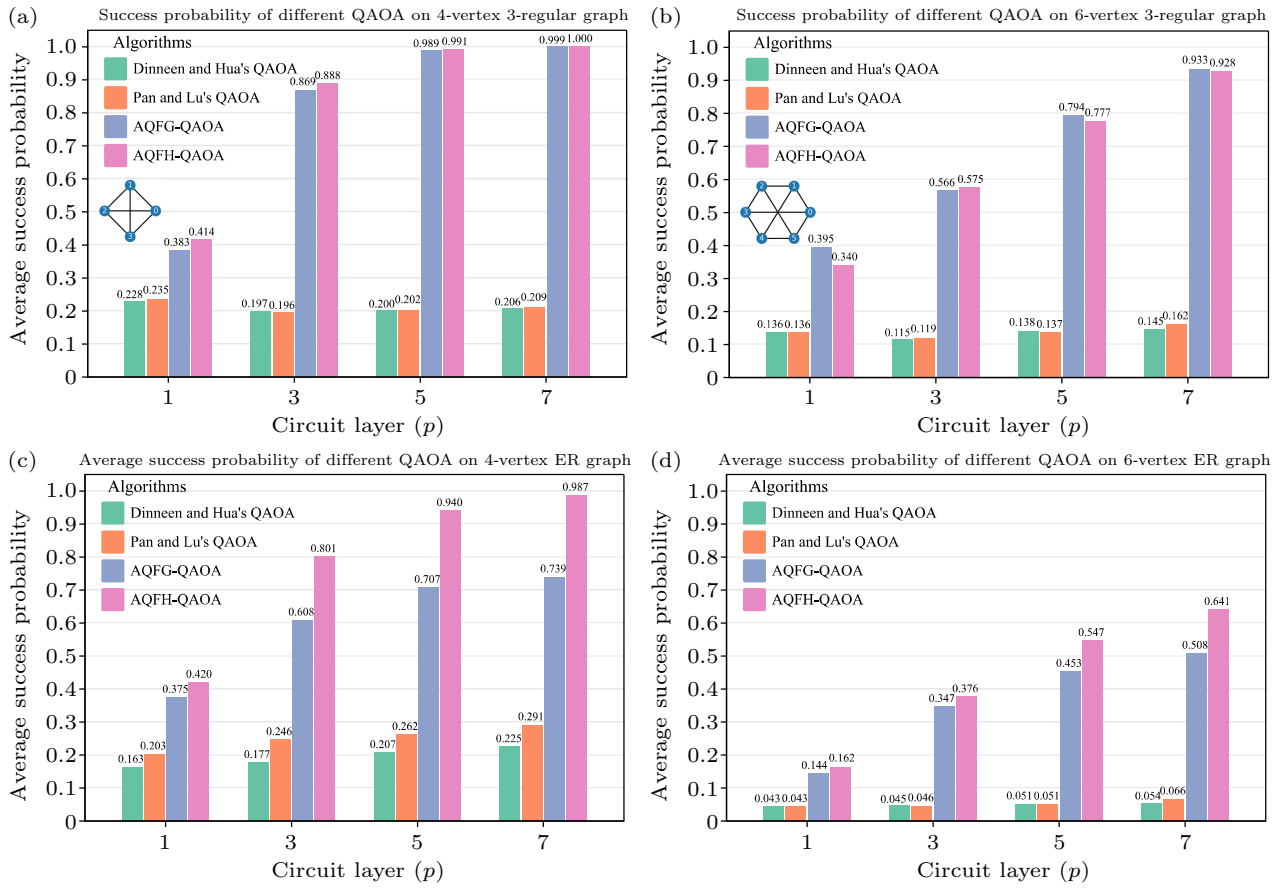


Fig. 10. Success probabilities of related algorithms for the MDS problem on 3-regular and ER graphs. Subfigures (a) and (b) present the success probabilities on 3-regular graphs with 4 and 6 vertices, respectively. Subfigures (c) and (d) show the average success probabilities on 10 randomly generated 4-vertex ER graphs and 20 randomly generated 6-vertex ER graphs, respectively.

5.3. Algorithm comparison and ablation study

We further compared the AQFH-QAOA algorithm with the AQFG-QAOA algorithm on 3-regular graphs and ER graphs with 8, 10, and 12 vertices. As discussed in Subsection 5.2, the circuit depth of the AQFG-QAOA algorithm remains much smaller for these instances because its multi-OR-controlled phase gates are not decomposed in simulation. To explore whether algorithmic performance could be improved by enhancing the ansatz, we design an ablation study. Inspired by the multi-angle QAOA (MA-QAOA),[47] we introduce multi-angle parameters into the quantum circuit of the AQFH-QAOA algorithm, creating a variant denoted as the AQFH-MA-QAOA algorithm. This modification is not readily applicable to AQFG-QAOA due to its fixed ansatz design.[31] The algorithms of Dinneen and Hua,[29] and Pan and Lu[30] are excluded from this comparison due to their previously observed poor performance and excessive auxiliary qubit requirements, which exceed our simulation capabilities.

A brief overview of MA-QAOA is provided in Appendix B. In the ablation study, we retain the quantum circuit structure of the AQFH-QAOA algorithm and only modify its parameters. Unlike the standard QAOA described in Subsection 2.2, we assign an independent angle parameter to each parameterized quantum gate corresponding to the phase separation operator $U_{\mathrm{P}}(\gamma)$ and the mixing operator $U_{\mathrm{M}}(\beta)$.

Figure 11 displays the distribution of success probabilities across 20 graph instances. Several key trends can be observed from the results. First, the AQFH-MA-QAOA algorithm consistently achieves higher success probabilities across all tested circuit layers and graph types, outperforming both the AQFH-QAOA and AQFG-QAOA algorithms. Second, the success probabilities of AQFH-QAOA and AQFG-QAOA decline significantly as the graph size increases, reflecting the growing complexity of the problem. In contrast, the AQFH-MA-QAOA algorithm exhibits stable performance on most of the graph instances tested. Third, the advantage of the AQFH-MA-QAOA algorithm is particularly pronounced on ER graphs. For circuit layers $p \geq 3$, it achieves both higher median success probabilities and tighter interquartile ranges, suggesting stronger robustness to variations in graph structure.

These results demonstrate that the AQFH-MA-QAOA algorithm enhances the expressivity and adaptability of the quantum circuit, enabling a more effective exploration of the solution space. Overall, the AQFH-MA-QAOA algorithm provides higher solution quality than existing algorithms for the MDS problem and remains competitive as the problem size grows, making it a promising candidate for implementation on near-term quantum devices.

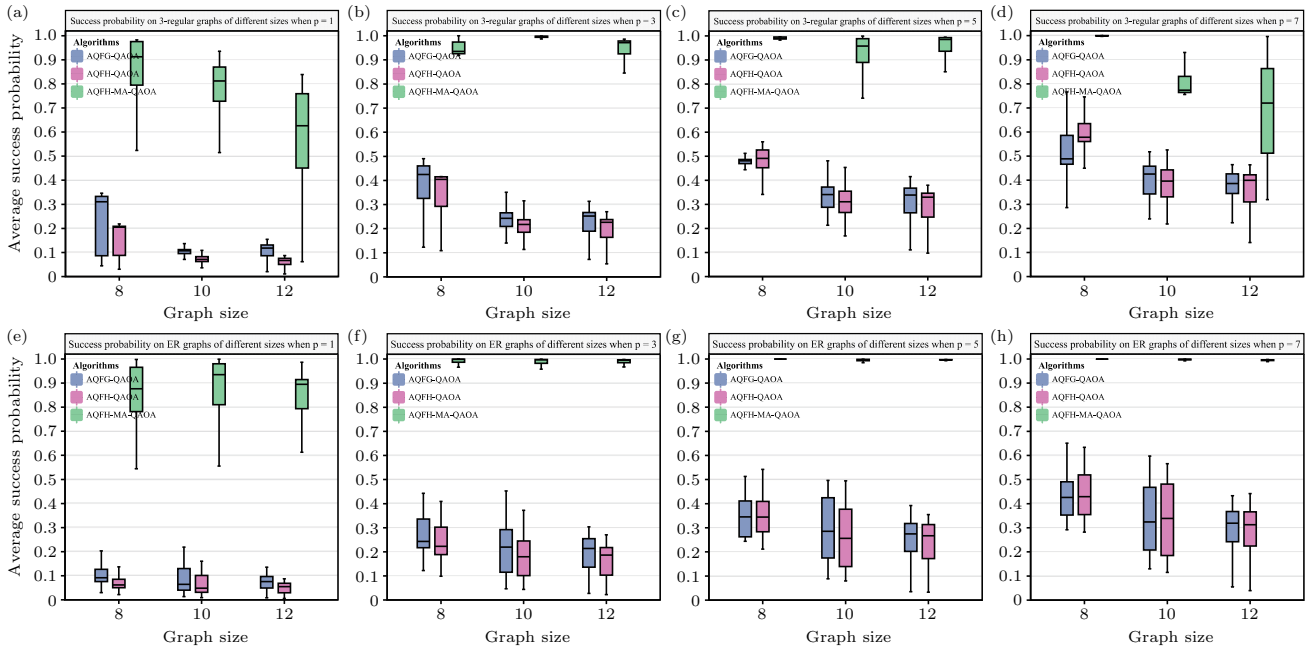


Fig. 11. Success probabilities of different algorithms for the MDS problems on 3-regular and ER graphs with varying circuit layers. Subfigures (a)–(d) correspond to the success probabilities on 3-regular graphs at circuit layers $p = 1, 3, 5, 7$, respectively. Subfigures (e)–(h) correspond to the success probabilities on ER graphs at circuit layers $p = 1, 3, 5, 7$, respectively. Each vertex size includes 20 graph instances, and outlier data are omitted for clarity.

6. Conclusion

In this study, we present two auxiliary-qubit-free QAOA algorithms for solving the MDS problem, namely AQFH-QAOA and AQFG-QAOA. The AQFH-QAOA algorithm reformulates the inequality constraints of the MDS problem into equality constraints using Boolean algebra, avoiding the need for auxiliary qubits in circuit construction. Specifically, the inequality constraints are first expressed as equivalent Boolean expressions, which are then converted into arithmetic expressions using generalized Boolean algebra identities. These arithmetic expressions are incorporated into the objective function as penalty terms to construct the objective Hamiltonian, allowing quantum circuits to be implemented via the unitary evolution of the Hamiltonian without auxiliary qubits. In contrast, existing QAOA algorithms typically rely on a large number of surplus variables to handle inequality constraints, which increases circuit overhead by introducing auxiliary qubits. Theoretical analysis demonstrates that AQFH-QAOA is particularly suitable for sparse graphs with small vertex degrees, reducing qubit overhead and exhibiting lower gate complexity.

Furthermore, by employing the decomposition technique for multi-controlled X gates, we provide an auxiliary-qubit-free optimized implementation of Guerrero's QAOA, referred to as the AQFG-QAOA algorithm. A detailed gate complexity analysis shows that AQFH-QAOA is well-suited for MDS problems on arbitrary 3-regular graphs, whereas for ER graphs with an edge probability of 0.5, the preferred algorithm depends on the graph size. Numerical simulations on randomly generated 3-regular and ER graphs demonstrate that AQFH-QAOA achieves higher solution quality than most existing QAOA algorithms. However, due to the simulation platform's inability to decompose multi-OR-controlled phase gates in AQFG-QAOA, its performance is comparable to or slightly better than AQFH-QAOA. Nevertheless, an enhanced variant based on the multi-angle QAOA framework, referred to as AQFH-MA-QAOA, effectively exploits the expressive power of independent parameters at low circuit depths, providing a competitive solution to the MDS problem on near-term quantum devices.

Although the proposed algorithms are evaluated in ideal simulations, their performance on real quantum devices may be affected by quantum noise. In particular, circuit depth increases with vertex degree, potentially causing the accumulation of gate errors, especially for two-qubit gates such as CNOT gates. Nevertheless, the auxiliary-qubit-free design significantly reduces the total number of qubits compared with previous approaches. Since qubit decoherence and readout errors generally scale with the number of physical qubits, this reduction may partially mitigate the impact of increased circuit depth. Therefore, the proposed algorithms offer a potential trade-off between circuit width and circuit depth for near-term quantum device implementation.

Looking ahead, this study can be extended in two directions. First, it would be valuable to implement and benchmark the algorithms on real quantum hardware to assess their practical performance under realistic noise conditions. Second, the Boolean algebra-based approach for handling inequality constraints can be generalized to other combinatorial optimization

problems with inequality constraints, enabling the design of more resource-efficient quantum algorithms.

Appendix A: Guerrero's gate decomposition method

Guerrero provides an explicit decomposition of multi-OR-controlled phase gates in Ref. [31]. Specifically, such a gate can be decomposed into two multi-OR-controlled X gates and a single-controlled phase gate using an auxiliary qubit, as illustrated in Fig. A1. Subsequently, each multi-OR-controlled X gate with n control qubits can be further decomposed into $2n-3$ OR-controlled Toffoli gates using $n-2$ auxiliary qubits, as shown in Fig. A2. Each OR-controlled Toffoli gate can then be implemented using one standard Toffoli gate and two CNOT gates, as depicted in Fig. A3. Furthermore, Fig. A4 illustrates a method for decomposing a single-controlled phase gate with $|0\rangle$ as the control qubit into elementary gates consisting of CNOT and single-qubit gates. As shown, such a gate can be implemented using two CNOT gates and four single-qubit gates.

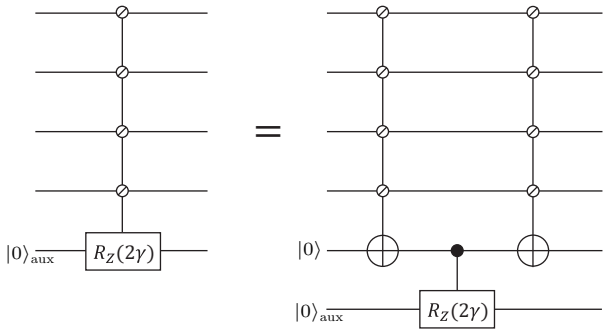


Fig. A1. Decomposition of a multi-OR-controlled phase gate with four control qubits using an auxiliary qubit and two multi-OR-controlled X gates. Adapted from Fig. 15 of Ref. [31].

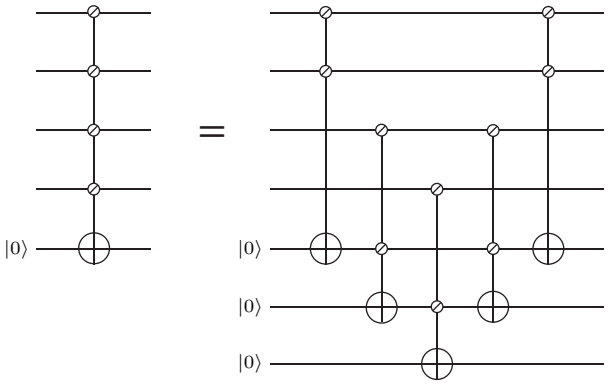


Fig. A2. Decomposition of a multi-OR-controlled X gate with n control qubits using $n-2$ auxiliary qubits and $2n-3$ OR-controlled Toffoli gates. Here, $n=4$. Adapted from Fig. 13 of Ref. [31].

According to Guerrero's decomposition method, an n-OR-controlled phase gate can be decomposed into $4n-6$ Toffoli gates, $8n-12$ CNOT gates, and a single-controlled phase gate, using a total of $n-1$ auxiliary qubits. These gates can be further decomposed into elementary gates, facilitating their implementation on quantum hardware. A complementary discussion of this decomposition scheme for multi-OR-controlled phase gates can also be found in Appendix E of Ref. [48].

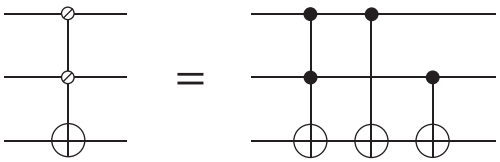


Fig. A3. Decomposition of an OR-controlled Toffoli gate using a single Toffoli gate and two CNOT gates. Adapted from Fig. 11 of Ref. [31].

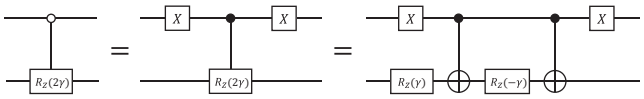


Fig. A4. Decomposition of a single-controlled phase gate using CNOT gates and single-qubit gates.

Next, we explicitly determine the number of elementary gates and auxiliary qubits required to decompose an n-vertex undirected graph using the methods described above. For simplicity, we assume that the average degree of the vertices in the undirected graph is d. For such a graph, each vertex has d neighbors, and the multi-OR-controlled phase gate that implements the corresponding $T_k(x)$ clause contains $d+1$ control qubits (including the vertex itself). As described above, such a $(d+1)$-OR-controlled phase gate can be decomposed into $4d-2$ Toffoli gates, $8d-4$ CNOT gates, and a single-controlled phase gate, using d auxiliary qubits. A Toffoli gate can be further decomposed into 3 CNOT gates and 4 single-qubit gates using the Congruent Modulo Phase Shifts method,[37] while a single-controlled phase gate can be decomposed into 2 CNOT gates and 2 single-qubit gates, as illustrated in Fig. A4. Consequently, a $(d+1)$-OR-controlled phase gate can be implemented using $20d-8$ CNOT gates and $16d-6$ single-qubit gates, together with d auxiliary qubits. Since the degree of each vertex is assumed to be d, there are n such $(d+1)$-OR-controlled phase gates in total. Therefore, implementing all $T_k(x)$ clauses requires $20nd-8n$ CNOT gates, $16nd-6n$ single-qubit gates, and d auxiliary qubits.

Using the decomposition method shown in Fig. A4, implementing all the single-controlled phase gates corresponding to the n $D_k(x)$ clauses requires $2n$ CNOT gates and $4n$ single-qubit gates. Including the n Hadamard gates used to prepare the initial state and the n single-qubit $R_X(2\beta)$ gates in the mixing operator, the single-layer quantum circuit of Guerrero's QAOA algorithm for an n-vertex undirected graph with average vertex degree d contains a total of $20nd-6n$ CNOT gates and $16nd$ single-qubit gates, using d auxiliary qubits.

Appendix B: Introduction to MA-QAOA

The MA-QAOA is a generalization of the standard QAOA that introduces a distinct angle parameter for each term in the phase separation operator and the mixing operator, rather than using a single parameter per operator per layer.[47] This enhanced parametrization increases the expressive power of the quantum circuit, enabling it to explore a larger portion of Hilbert space and potentially yielding higher-quality solutions.

In the standard QAOA with circuit depth p, the variational quantum state is generated by alternately applying the unitary evolution operators of the objective Hamiltonian $H_{\rm P}$ and the mixing Hamiltonian $H_{\rm M}$

$$|\psi_p(\gamma,\beta)\rangle = \prod_{k=1}^{p} \mathrm{e}^{-\mathrm{i}\beta_k H_{\rm M}} \mathrm{e}^{-\mathrm{i}\gamma_k H_{\rm P}} |\psi_0\rangle, \tag{B1}$$

where $\boldsymbol{\gamma} = (\gamma_1,\ldots,\gamma_p)$ and $\boldsymbol{\beta} = (\beta_1,\ldots,\beta_p)$ are the variational parameters. A p-layer standard QAOA circuit thus contains $2p$ parameters $\{\gamma_k,\beta_k\}_{k=1}^{p}$. Previous studies have shown that using a limited number of parameters may restrict the circuit's ability to explore the solution space, potentially leading to suboptimal outcomes.[47,49]

MA-QAOA addresses this limitation by assigning independent parameters to each parameterized quantum gate in the circuit. Specifically, since the objective Hamiltonian $H_{\rm P}$ and the mixing Hamiltonian $H_{\rm M}$ can be expressed as sums of operators, Herrman *et al.* write $H_{\rm P} = \sum_u C_u$ and $H_{\rm M} = \sum_v B_v$. Accordingly, the phase separation operator for the k-th layer is reformulated as

$$U_{\rm P}(\gamma_k) = \mathrm{e}^{-\mathrm{i}\gamma_k \sum_u C_u} = \mathrm{e}^{-\mathrm{i}\sum_u \gamma_{k,u} C_u} = \prod_u \mathrm{e}^{-\mathrm{i}\gamma_{k,u} C_u}, \tag{B2}$$

where $\boldsymbol{\gamma}_k = (\gamma_{k,u_1},\gamma_{k,u_2},\ldots)$ is the vector of independent parameters for the k-th layer. Similarly, the mixing operator becomes

$$U_{\rm M}(\boldsymbol{\beta}_k) = \mathrm{e}^{-\mathrm{i}\beta_k \sum_v B_v} = \mathrm{e}^{-\mathrm{i}\sum_v \beta_{k,v} B_v} = \prod_v \mathrm{e}^{-\mathrm{i}\beta_{k,v} B_v}, \tag{B3}$$

where $\boldsymbol{\beta}_k = (\beta_{k,v_1},\beta_{k,v_2},\ldots)$ is the vector of independent parameters for the mixing operator at layer k. This multi-angle parametrization allows the circuit to adjust each term individually, improving expressivity and optimization flexibility.

Hence, the ansatz state prepared by MA-QAOA with circuit depth p can be expressed as

$$|\psi_p\left(\{\boldsymbol{\gamma}_k,\boldsymbol{\beta}_k\}_{k=1}^{p}\right)\rangle = \prod_{k=1}^{p}\left(\prod_v \mathrm{e}^{-\mathrm{i}\beta_{k,v}B_v}\prod_u \mathrm{e}^{-\mathrm{i}\gamma_{k,u}C_u}\right)|\psi_0\rangle. \tag{B4}$$

By assigning an independent rotation angle to each quantum gate, MA-QAOA generally achieves higher success probabilities with fewer circuit layers compared to standard QAOA.[47] Figure B1 illustrates the difference between the two algorithms, highlighting that their circuit structures are identical except for the parameterization.

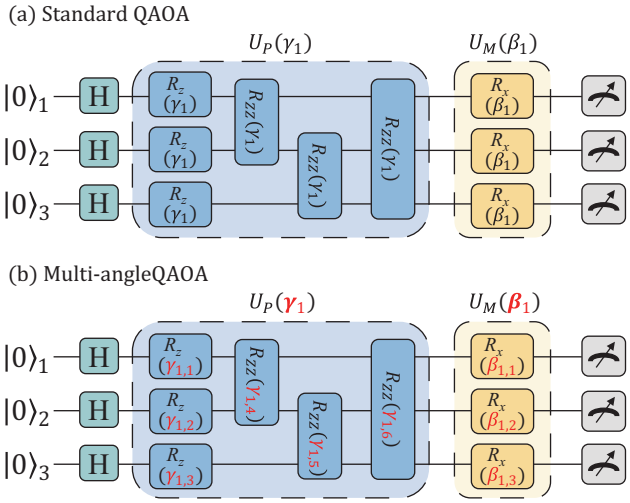


Fig. B1. Single-layer quantum circuits of (a) standard QAOA and (b) multi-angle QAOA.

Acknowledgments

We thank Jiacheng Fan, Lingxiao Li, and Jing Li for valuable discussions on this work. This project was supported by the National Natural Science Foundation of China (Grant Nos. 62372048, 62272056, 62371069, and U25B2014) and the National Key Laboratory of Secure Communication Foundation (Grant No . 2025,6142103042503).

References

[1] Wu H Y, Feng X N, Zhang K J and Sun H W 2024 *Adv. Quantum Technol.* **7** 2300384
[2] Zhang H, Wang S, Liu X, Shen Y and Wang Y 2024 *Chin. Phys. B* **33** 020310
[3] Xu H Z, Zhuang W F, Wang Z A, Huang K X, Shi Y H, Ma W G, Li T M, Chen C T, Xu K and Feng Y L 2024 *Chin. Phys. B* **33** 050302
[4] Lloyd S, Mohseni M and Rebentrost P 2014 *Nat. Phys.* **10** 631
[5] Wu Z, Song T and Zhang Y 2022 *Quantum Inf. Process.* **21** 19
[6] Zeguendry A, Jarir Z and Quafafou M 2023 *Entropy* **25** 287
[7] Yu K, Lin S and Cai B B 2025 *J. Phys. A: Math. Theor.* **58** 495302
[8] Gao S and Yang Y G 2023 *Chin. Phys. B* **32** 100309
[9] Su J, Fan J, Wu S, Li G, Qin S and Gao F 2025 *Sci. China Inf. Sci.* **68** 180507
[10] Shor P W 1994 *Proceedings 35th Annual Symposium on Foundations of Computer Science* (Santa Fe: IEEE) pp. 124
[11] Grover L K 1996 *Proceedings of the 28th Annual ACM Symposium on Theory of Computing* (Philadelphia: ACM) pp. 212
[12] Harrow A W, Hassidim A and Lloyd S 2009 *Phys. Rev. Lett.* **103** 150502
[13] Cerezo M, Arrasmith A, Babbush R, Benjamin S C, Endo S, Fujii K, McClean J R, Mitarai K, Yuan X and Cincio L 2021 *Nat. Rev. Phys.* **3** 625
[14] Preskill J 2018 *Quantum* **2** 79
[15] Peruzzo A, McClean J, Shadbolt P, Yung M H, Zhou X Q, Love P J, Aspuru-Guzik A and O'brien J L 2014 *Nat. Commun.* **5** 4213
[16] Farhi E, Goldstone J and Gutmann S 2014 arxiv:1411.4028
[17] Vikstål P, Grönkvist M, Svensson M, Andersson M, Johansson G and Ferrini G 2020 *Phys. Rev. Appl.* **14** 034009
[18] Azad U, Behera B K, Ahmed E A, Panigrahi P K and Farouk A 2022 *IEEE Trans. Intell. Transp. Syst.* **24** 7564
[19] Kurowski K, Pecyna T, Slysz M, Różycki R, Waligóra G and Wglarz J 2023 *Eur. J. Oper. Res.* **310** 518

[20] Hadfield S, Wang Z, Rieffel E G, O'Gorman B, Venturelli D and Biswas R 2017 *Proceedings of the Second International Workshop on Post Moores Era Supercomputing* (New York: ACM) pp. 15

[21] Saleem Z H, Tomesh T, Tariq B and Suchara M 2023 *SN Comput. Sci.* **4** 183

[22] Pan H and Lu C 2024 *Entropy* **26** 1057

[23] Li G H, Wang S S, Ni X H, Gao F, Guo F Z, Qin S J and Wen Q Y 2025 *Phys. Lett. A* **553** 130690

[24] Li G, Wang S, Zhao X, Gao F, Qin S, Guo F and Jin Z 2025 *Quantum Inf. Process.* **24** 166

[25] Ni X H, Li L X, Song Y Q, Jin Z P, Qin S J and Gao F 2025 *Chin. Phys. B* **34** 070304

[26] Ore O 1962 *Theory of Graphs* (*American Mathematical Society Colloquium Publications* vol. 38) (Providence: AMS)

[27] Fomin F V, Kratsch D and Woeginger G J 2005 *Graph-Theoretic Concepts in Computer Science* ed. Hromkovič J, Nagl M and Westfechtel B (Berlin, Heidelberg: Springer) pp. 245

[28] Casado A, Bermudo S, López-Sánchez A and Sánchez-Oro J 2023 *Math. Comput. Simul.* **207** 41

[29] Dinneen M J and Hua R 2017 *Proceedings of the Australasian Computer Science Week Multiconference* (New York: ACM) pp. 1

[30] Pan H and Lu C 2025 *J. Math UK* **2025** 3201094

[31] Guerrero N J 2020 *Solving Combinatorial Optimization Problems Using The Quantum Approximation Optimization Algorithm* Master's Thesis, Air Force Institute of Technology

[32] Zhou L, Wang S T, Choi S, Pichler H and Lukin M D 2020 *Phys. Rev. X* **10** 021067

[33] Krauss T, McCollum J, Pendery C, Litwin S and Michaels A J 2020 *IEEE Trans. Quantum Eng.* **1** 1

[34] Glover F, Kochenberger G, Hennig R and Du Y 2022 *Ann. Oper. Res.* **314** 141

[35] Hadfield S 2021 *ACM T. Quantum. Comput.* **2** 1

[36] Maslov D and Zindorf B 2022 *IEEE Trans. Quantum Eng.* **3** 1

[37] Barenco A, Bennett C H, Cleve R, DiVincenzo D P, Margolus N, Shor P, Sleator T, Smolin J A and Weinfurter H 1995 *Phys. Rev. A* **52** 3457

[38] Liu Y, Long G L and Sun Y 2008 *Int. J. Quantum Inf.* **6** 447

[39] Saeedi M and Pedram M 2013 *Phys. Rev. A* **87** 062318

[40] Luo M X and Li H R 2016 *Phys. Rev. A* **94** 026301

[41] da Silva A J and Park D K 2022 *Phys. Rev. A* **106** 042602

[42] Maslov D and Dueck G W 2003 *Electron. Lett.* **39** 1790

[43] He Y, Luo M X, Zhang E, Wang H K and Wang X F 2017 *Int. J. Theor. Phys.* **56** 2350

[44] Bollobás B 1998 *Random Graphs in: Modern Graph Theory Graduate Texts in Mathematics* (New York: Springer) pp. 215

[45] Anand K and Bianconi G 2009 *Phys. Rev. E* **80** 045102

[46] Xu X, Cui J, Cui Z, *et al*. 2024 arXiv:2406.17248

[47] Herrman R, Lotshaw P C, Ostrowski J, Humble T S and Siopsis G 2022 *Sci. Rep.* **12** 6781

[48] Donkers H, Mesman K, Al-Ars Z and Möller M 2022 arXiv:2205.12142

[49] Vijendran V, Das A, Koh D E, Assad S M and Lam P K 2024 *Quantum Sci. Technol.* **9** 025010