

Solving the Perfect Domination Problem by Quantum Approximate Optimization Algorithm with Small Layers

Haoqian Pan^a, Changhong Lu^a, Yuqing Zheng^b

^a*School of Mathematical Sciences, Key Laboratory of MEA(Ministry of Education) & Shanghai Key Laboratory of PMMP, East China Normal University, Shanghai, 200241, China*

^b*School of Astronomy and Space Science, University of Science and Technology of China, Hefei, 230026, China*

Abstract

The Perfect Domination Problem (PDP), a classical challenge in combinatorial optimization, has significant applications in real-world scenarios, such as wireless and social networks. Over several decades of research, the problem has been demonstrated to be NP-complete across numerous graph classes. With the recent advancements in quantum computing, there has been a surge in the development of quantum algorithms aimed at addressing NP-complete problems, including the Quantum Approximate Optimization Algorithm (QAOA). However, the applicability of quantum algorithms to the PDP, as well as their efficacy in solving it, remains largely unexplored. This study represents a pioneering effort to apply QAOA, with a limited number of layers, to address the PDP. Comprehensive testing and analysis were conducted across 420 distinct parameter combinations. Experimental results indicate that QAOA successfully identified the correct Perfect Dominating Set (PDS) in 82 cases, including 17 instances where the optimal PDS was computed. Furthermore, it was observed that with appropriate parameter settings, the approximation ratio could achieve a value close to 0.9. These findings underscore the potential of QAOA as a viable approach for solving the PDP, signifying an important milestone that introduces this problem into the realm of quantum computing.

Keywords:

Quantum approximate optimization algorithm, Perfect domination problem, Qiskit

1. Introduction

For a graph $G = (V, E)$, a dominating set (DS) is a subset $D \subseteq V$ such that every vertex $v \in V \setminus D$ has at least one neighbor in D . A more stringent requirement defines the perfect dominating set (PDS), where every vertex $v \in V \setminus D$ must have exactly one neighbor in D . The Perfect Domination Problem (PDP) seeks to identify the smallest such set D . The PDP is closely related to various practical applications, including parallel computer networks (Livingston and Stout, 1990), wireless communication networks (Fei, 2020), and social networks (Hamja, 2023). As a specialized variant of the domination problem (DP), the PDP has been proven NP-complete for many graph classes (Fellows and Hoover, 1991; Yen and Lee, 1990). Over the years, numerous algorithms have been developed to address the PDP, often tailored to specific graph types or specialized problem variations, such as the weighted version, to manage its NP-complete nature. Significant studies have explored the PDP in various graph categories, including rectangular grid graphs (Dejter and Delgado, 2007), distance-hereditary graphs (Hsieh, 2007), circular-arc graphs (Lin et al., 2015), and knight's graphs (Fenstermacher et al., 2018). Additionally, the weighted PDP has been investigated in tree graphs (Yen and Lee, 1990), chordal graphs, and split graphs (Chang and Liu, 1993), while weighted independent PDP solutions have been studied for cocomparability graphs (Chang et al., 1995). Despite some algorithms achieving polynomial or even linear time complexity for restricted graph types, solving the PDP remains computationally challenging in its general form due to its NP-complete nature.

In recent years, the rapid advancement of quantum computing (Raussendorf and Briegel, 2001; O’brien, 2007; Preskill, 2023; Caleffi et al., 2024) has spurred the development of numerous quantum algorithms aimed at addressing NP-complete combinatorial optimization problems. Prominent among these algorithms are Quantum Annealing (Finnila et al., 1994), the Quantum Approximate Optimization Algorithm (QAOA) (Farhi et al., 2014), and the Variational Quantum Eigensolver (Tilly et al., 2022), among others. As one of the most influential quantum algorithms of the past decade, QAOA has been applied to a wide array of combinatorial optimization problems, including max-cut (Farhi et al., 2014), the traveling salesman problem (Radzhovsky et al., 2019), the domination problem (DP) (Guerrero, 2020), max-flow (Krauss et al., 2020), the minimum vertex cover problem (Zhang et al., 2022), the boolean satisfiability problem (SAT) (Yu et al., 2023), the independent domination

problem (Pan and Lu, 2024b), and the total domination problem (Pan et al., 2024), among others.

In this study, we explore the application of QAOA to solve the PDP. Using a graph with 6 vertices, we conducted an extensive evaluation of low-layer QAOA across 420 distinct parameter combinations to assess its effectiveness in solving the PDP. The experimental results demonstrate that QAOA is capable of solving the PDP and exhibits significant potential for further exploration. By analyzing parameter distributions, we identified distinct trends in the approximation ratio and in the algorithm’s ability to compute both the optimal and correct PDS. The primary contributions of this work are as follows: (1) This is the first study to apply a quantum algorithm to the PDP; (2) The results under various parameter settings reveal QAOA’s tendencies when addressing the PDP, offering valuable insights and guidance for future research.

The structure of the paper is as follows. In Section 2, we present a 0-1 integer programming model for the PDP based on its definition and outline the steps to transform it into a Hamiltonian. In Section 3, we introduce the fundamental concept of QAOA. In Section 4, we employ a quantum simulator to solve the PDP using QAOA under various parameter combinations and provide a comprehensive analysis of the experimental results. Finally, in Section 5, we conclude the paper with a summary of the key findings.

2. Problem modeling of PDP

The premise of using QAOA is to first map the objective function of a combinatorial optimization problem to the Hamiltonian of a quantum system. In this process, the **Quadratic Unconstrained Binary Optimization (QUBO)** model often acts as a bridge. The standard approach involves transforming the original problem into, or directly modeling it as, a QUBO model, which is then converted into a Hamiltonian through variable substitution. Following this methodology, our approach sequentially transforms the PDP into a 0-1 integer programming model, a QUBO model, and finally a Hamiltonian. We

begin by presenting the 0-1 integer programming model for the PDP.

$$\min_{\{X_i\}} \sum_{i=1}^{|V|} X_i \quad (1)$$

$$\text{s.t. } \sum_{j \in N[i]} X_j \geq 1 \quad \forall i \in V \quad (2)$$

$$X_i \in \{0, 1\} \quad \forall i \in V \quad (3)$$

$$\sum_{ij \in E} [X_i * (1 - X_j) + X_j * (1 - X_i)] = |V| - \sum_i^{|V|} X_i \quad (4)$$

The decision variable X_i is defined such that $X_i = 1$ if $i \in D$, and $X_i = 0$ otherwise. Accordingly, the objective function in Eq. 1 represents the total number of vertices in D . Constraint 2 ensures that for each vertex i , either i is in D or it has at least one neighbor in D . This constraint guarantees that D forms a DS. For constraint 4, the left-hand side calculates the number of edges in E where one endpoint belongs to $V \setminus D$ and the other belongs to D . The right-hand side represents the number of vertices in $V \setminus D$. Since D is a DS, every vertex in $V \setminus D$ must have at least one neighbor in D . Thus, the number of edges connecting $V \setminus D$ to D must be at least $|V| - \sum_i^{|V|} X_i$. When this inequality holds as an equality, it signifies that each vertex in $V \setminus D$ has exactly one neighbor in D , satisfying the perfect domination condition. Therefore, constraint 4 is the defining constraint of the PDP.

After formulating the 0-1 integer programming model for the PDP, the next step is to transform it into a QUBO model. The standard representation of the QUBO model is provided in Eq. 5, where the matrix Q , commonly referred to as the QUBO matrix.

$$\text{minimize/maximize } y = x^t Q x \quad (5)$$

Given that $X_* \in \{0, 1\}$, it follows that $X_* = X_*^2$. As a result, the objective function of the 0-1 integer programming model for the PDP inherently satisfies the requirements of the QUBO model. The next step involves transforming the constraints into quadratic penalty terms. The general form of constraint 2 is expressed as follows:

$$X_1 + X_2 + \cdots + X_n \geq 1, \quad n = |N[i]|, \quad \forall i \in V \quad (6)$$

According to Glover et al. (2022), for $n = 1$ or $n = 2$, the constraint can be transformed into $P \cdot (X_1 - 1)^2$ and $P \cdot (1 - X_1 - X_2 + X_1 \cdot X_2)$, respectively, where P is the penalty coefficient. The value of P typically requires adjustment based on the specific characteristics of the problem. As suggested in Glover et al. (2022), setting P to 0.75 to 1.5 times the value of the original objective function serves as a reasonable starting point. For $n \geq 3$, slack variables must be introduced to convert the inequality in Eq. 6 into an equality constraint, as shown in Eq. 7.

$$X_1 + X_2 + \dots + X_n - S - 1 = 0 \quad (7)$$

It is evident that the range of S encompasses all integers within the interval $[0, n - 1]$. To represent S , we introduce additional 0-1 variables, following the formulation provided in Eq. 8 (Krauss et al., 2020; Pan and Lu, 2024a). This approach ensures that S can take any integer value within the specified range $[0, n - 1]$.

$$S = \sum_{i=1}^{bl_{n-1}-1} X'_i * 2^{i-1} + (n - 1 - \sum_{i=1}^{bl_{n-1}-1} 2^{i-1}) * X'_{bl_{n-1}} \quad (8)$$

In Eq. 8, $X'_* \in \{0, 1\}$ represents the newly introduced binary variables, while bl_{n-1} denotes the length of the binary representation of $n - 1$. This approach, as utilized by Krauss et al. (2020), has been applied to represent flow values in the maximum flow problem, with its correctness formally established in Pan and Lu (2024a). Building on these works, we transform Eq. 7 into quadratic penalties, as expressed in Eq. 9.

$$P \cdot (X_1 + X_2 + \dots + X_n - [\sum_{i=1}^{bl_{n-1}-1} X'_i \cdot 2^{i-1} + (n - 1 - \sum_{i=1}^{bl_{n-1}-1} 2^{i-1}) \cdot X'_{bl_{n-1}}] - 1)^2 \quad (9)$$

It is easy to see that for Eq. 4, when D is a DS,

$$\sum_{ij \in E} [X_i * (1 - X_j) + X_j * (1 - X_i)] \geq |V| - \sum_i X_i \quad (10)$$

Therefore, we can convert it without using the square form into

$$P \cdot (\sum_{ij \in E} [X_i \cdot (1 - X_j) + X_j \cdot (1 - X_i)] - |V| + \sum_i X_i) \quad (11)$$

Since constraint 4 builds upon the premise of constraint 2, we assign distinct symbols to represent their respective penalty coefficients, P_2 and P_1 , where $P_2 \leq P_1$. By transforming the two types of constraints in the PDP into quadratic penalty terms, we ultimately derive the QUBO model for the PDP.

$$\begin{aligned}
& \min_{\{X, X'\}} \sum_{i=1}^{|V|} X_i \\
& + \sum_{i \in V, |N[i]| \geq 3} P_1 \cdot \left[\sum_{j \in N[i]} X_j - \left(\sum_{i=1}^{bl_{|N[i|-1}-1}} X'_i \cdot 2^{i-1} + (|N[i]| - 1 - \sum_{i=1}^{bl_{|N[i|-1}-1}} 2^{i-1}) \cdot X'_{bl_{|N[i|-1}}} \right) - 1 \right]^2 \\
& + \sum_{i \in V, |N[i]|=2, N[i]=\{j,k\}} P_1 \cdot (1 - X_j - X_k + X_j \cdot X_k) \\
& + \sum_{i \in V, |N[i]|=1, N[i]=\{j\}} P_1 \cdot (X_j - 1)^2 \\
& + P_2 \cdot \left(\sum_{ij \in E} [X_i \cdot (1 - X_j) + X_j \cdot (1 - X_i)] - |V| + \sum_i^{|V|} X_i \right)
\end{aligned} \tag{12}$$

Next, to proceed with converting the QUBO model into a Hamiltonian, we replace X and X' with binary variables s , which take values in $\{-1, 1\}$. The conversion process is outlined as follows:

$$X_i = \frac{s_i + 1}{2} \tag{13}$$

After the substitution, Eq. 12 becomes:

$$\begin{aligned}
& \min_{\{s, s'\}} \sum_{i=1}^{|V|} \frac{s_i + 1}{2} \\
& + \sum_{i \in V, |N[i]| \geq 3} P_1 * \left[\sum_{j \in N[i]} \frac{s_j + 1}{2} \right. \\
& \quad \left. - \left(\sum_{i=1}^{bl_{|N[i]|-1}-1} \frac{s'_i + 1}{2} * 2^{i-1} + (|N[i]| - 1 - \sum_{i=1}^{bl_{|N[i]|-1}-1} 2^{i-1}) * \frac{s'_{bl_{|N[i]|-1}} + 1}{2} \right) - 1 \right]^2 \\
& + \sum_{i \in V, |N[i]|=2, N[i]=\{j, k\}} P_1 \cdot \left(1 - \frac{s_j + 1}{2} - \frac{s_k + 1}{2} + \frac{s_j + 1}{2} \cdot \frac{s_k + 1}{2} \right) \\
& + \sum_{i \in V, |N[i]|=1, N[i]=\{j\}} P_1 \cdot \left(\frac{s_j + 1}{2} - 1 \right)^2 \\
& + P_2 \cdot \left(\sum_{ij \in E} \left[\frac{s_i + 1}{2} \cdot \left(1 - \frac{s_j + 1}{2} \right) + \frac{s_j + 1}{2} \cdot \left(1 - \frac{s_i + 1}{2} \right) \right] - |V| + \sum_i \frac{s_i + 1}{2} \right)
\end{aligned} \tag{14}$$

Finally, by replacing s and s' with the Pauli-Z operator σ^z , we can ultimately

obtain the Hamiltonian.

$$\begin{aligned}
H_c = & \sum_{i=1}^{|V|} \frac{\sigma_i^z + 1}{2} \\
& + \sum_{i \in V, |N[i]| \geq 3} P_1 \cdot \left[\sum_{j \in N[i]} \frac{\sigma_j^z + 1}{2} \right. \\
& \quad \left. - \left(\sum_{i=1}^{bl_{|N[i]|-1}-1} \frac{(\sigma_i^z)' + 1}{2} \cdot 2^{i-1} + (|N[i]| - 1 - \sum_{i=1}^{bl_{|N[i]|-1}-1} 2^{i-1}) \cdot \frac{(\sigma_{bl_{|N[i]|-1}}^z)' + 1}{2} \right) - 1 \right]^2 \\
& + \sum_{i \in V, |N[i]|=2, N[i]=\{j,k\}} P_1 \cdot \left(1 - \frac{\sigma_j^z + 1}{2} - \frac{\sigma_k^z + 1}{2} + \frac{\sigma_j^z + 1}{2} \cdot \frac{\sigma_k^z + 1}{2} \right) \\
& + \sum_{i \in V, |N[i]|=1, N[i]=\{j\}} P_1 \cdot \left(\frac{\sigma_j^z + 1}{2} - 1 \right)^2 \\
& + P_2 \cdot \left(\sum_{ij \in E} \left[\frac{\sigma_i^z + 1}{2} \cdot \left(1 - \frac{\sigma_j^z + 1}{2} \right) + \frac{\sigma_j^z + 1}{2} \cdot \left(1 - \frac{\sigma_i^z + 1}{2} \right) \right] - |V| + \sum_i^{|V|} \frac{\sigma_i^z + 1}{2} \right)
\end{aligned} \tag{15}$$

At this stage, we have completed the process of transforming the PDP from a 0-1 integer programming model into a Hamiltonian. In the subsequent sections, we will outline the basic workflow of QAOA.

3. QAOA

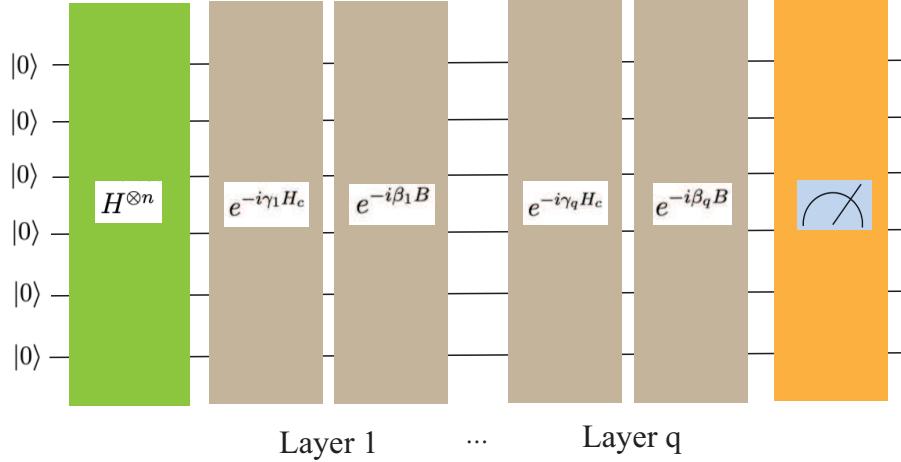


Figure 1: Working flow of QAOA.

The basic concept of QAOA is illustrated in Fig. 1. To begin, the spin operator and the Pauli operator are related as follows:

$$\hat{s} = \frac{\hbar}{2} \hat{\sigma} \quad (16)$$

For the z -component of \hat{s} , it is naturally related to $\hat{\sigma}^z$. In the Pauli representation,

$$\hat{\sigma}^z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \quad (17)$$

It has two eigenstates, $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$, which correspond to two distinct spin directions. Initially, we assume that each qubit in a quantum system with n qubits is in the $|0\rangle$ state, such that the system's state is represented as $|\underbrace{00\dots 0}_n\rangle$. Subsequently, QAOA applies the Hadamard gate to prepare this state into an equal superposition of all basis states, as described in Eq. 18. Here, the bit string $z = z_1 z_2 z_3 \dots z_n$, where $z_i \in \{0, 1\}$.

$$|s\rangle = \underbrace{\hat{H} \otimes \hat{H} \cdots \otimes \hat{H}}_n |\underbrace{00\dots 0}_n\rangle = \frac{1}{\sqrt{2^n}} \cdot \sum_z |z\rangle \quad (18)$$

Next, QAOA applies two types of rotation operators, $U(C, \gamma)$ and $U(B, \beta)$ (Eq. 19, Eq. 20), to the initial state $|s\rangle$, repeating the process q times. Here, $C = H_c$, $B = \sum_{j=1}^n \sigma_j^x$, with $\gamma \in [0, 2\pi]$ and $\beta \in [0, \pi]$. In Fig. 1, γ_q and β_q denote the angles used in the q -th layer. After q iterations, the final state $|\gamma, \beta\rangle$ is obtained, as shown in Eq. 21.

$$U(C, \gamma) = e^{-i\gamma H_c} \quad (19)$$

$$U(B, \beta) = e^{-i\beta B} \quad (20)$$

$$|\gamma, \beta\rangle = U(B, \beta_q)U(C, \gamma_q)\cdots U(B, \beta_1)U(C, \gamma_1)|s\rangle \quad (21)$$

With fixed values of γ and β , by repeatedly executing the quantum circuit depicted in Fig. 1 and measuring the final state, the expectation value of H_c , denoted as $F_q(\gamma, \beta)$, can be obtained.

$$F_q(\gamma, \beta) = \langle \gamma, \beta | H_c | \gamma, \beta \rangle \quad (22)$$

Since the PDP is a minimization problem, the values of γ_* and β_* at each layer must be adjusted to minimize $F_q(\gamma, \beta)$. The essence of QAOA lies in applying two types of operators in a manner that increases the probability of the system collapsing into basis states corresponding to the smallest values of H_c . As a hybrid algorithm, QAOA leverages classical optimization methods, such as COBYLA or Nelder-Mead, to tune the angles at each layer. The optimization process terminates when either the maximum number of iterations is reached or the function tolerance falls below a predefined threshold. Once the optimal values of γ_* and β_* are determined, the quantum circuit is updated accordingly, and multiple final samples are generated. The bit string z_* with the highest probability from these samples is then output. In z_* , the spin direction of each qubit corresponds to the 0-1 values of the decision variables in the original optimization problem, allowing for the direct extraction of the final PDS. According to theoretical results (Farhi et al., 2014), increasing the number of layers q brings $F_q(\gamma, \beta)$ closer to the optimal value. However, deeper layers lead to more complex quantum circuits, which present significant challenges when implementing QAOA on both real quantum computers and local quantum simulators. Therefore, this study focuses on analyzing the performance of QAOA in solving the PDP with a limited number of layers.

In this chapter, we introduced the basic workflow of QAOA. In the subsequent chapters, we will conduct experiments to evaluate the performance of low-layer QAOA in solving the PDP.

4. Experiment

The CPU used in this experiment was an AMD R9 7950X3D, paired with 48GB of memory. Qiskit was employed to construct the quantum circuit, simulate the backend, and perform sampling. We adopted the angle initialization method proposed in Sack and Serbyn (2021) and used COBYLA as the optimization algorithm, with a tolerance of 10^{-6} . All code for this work was implemented in Python. The primary parameters for this experiment included the layer number q , penalty coefficients P_1 and P_2 , and the maximum number of iterations. A total of 420 parameter combinations were tested, with $q \in \{1, 2, 5\}$, $P_1 \in \{0.8, 1, 1.2, 1.4, 1.6, 1.8, 2\} \times |V|$, $\text{rate} = \frac{P_2}{P_1} \in \{0.3, 0.5, 0.7, 1\}$, and the maximum iterations set to $\{100, 200, 500, 1000, 10000\}$. For the range of P_1 , the upper bound of the PDS was considered to be $|V|$. Based on recommendations from Glover et al. (2022), penalty coefficients were initially set between 0.7 and 1.5 times the value of the original objective function, with an extended range of $[1.6, 2] \times |V|$. For P_2 , since the validity of constraint 4 depends on constraint 2, we tested cases where $P_1 = P_2$ as well as $P_2 < P_1$. The graph used in this experiment is depicted in Fig. 2, consisting of 6 vertices and 6 edges. The minimal DS of this graph is $\{1, 4\}$, while the minimal PDS is either $\{0, 4\}$ or $\{1, 5\}$. Since this study focuses on solving the PDP rather than the DP, it was crucial to select a graph where the minimal DS and minimal PDS differ.

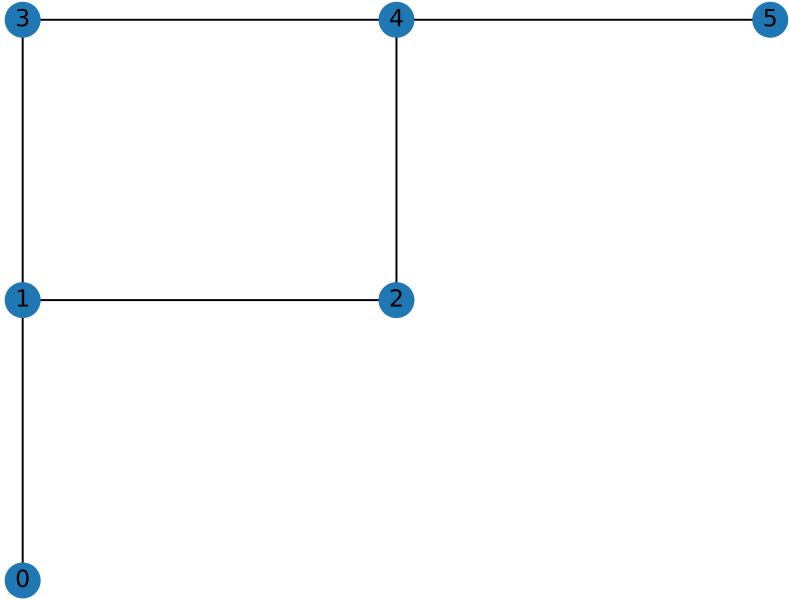


Figure 2: A graph with 6 nodes and 6 edges.

According to the conversion method outlined in Section 2, the QUBO model for the PDP of this graph is expressed in Eq. 23. Modeling this graph requires a total of 14 qubits. The corresponding Hamiltonian is derived by first substituting x_* with s_* as per Eq. 13, and then replacing s_* with σ^z .

For simplicity, the detailed expansion of the Hamiltonian is omitted here.

$$\begin{aligned}
\text{minimize} \quad & x_0 + x_1 + x_2 + x_3 + x_4 + x_5 \\
& + P_1 \cdot (1 - x_0 - x_1 + x_0 \cdot x_1) \\
& + P_1 \cdot (x_1 + x_0 + x_2 + x_3 - (x_6 + 2 \cdot x_7) - 1)^2 \\
& + P_1 \cdot (x_2 + x_1 + x_4 - (x_8 + x_9) - 1)^2 \\
& + P_1 \cdot (x_3 + x_1 + x_4 - (x_{10} + x_{11}) - 1)^2 \\
& + P_1 \cdot (x_4 + x_2 + x_3 + x_5 - (x_{12} + 2 \cdot x_{13}) - 1)^2 \\
& + P_1 \cdot (1 - x_5 - x_4 + x_5 \cdot x_4) \\
& + P_2 \cdot (x_0 \cdot (1 - x_1) + x_1 \cdot (1 - x_0) + x_1 \cdot (1 - x_2) \\
& + x_2 \cdot (1 - x_1) + x_1 \cdot (1 - x_3) + x_3 \cdot (1 - x_1) \\
& + x_2 \cdot (1 - x_4) + x_4 \cdot (1 - x_2) + x_3 \cdot (1 - x_4) \\
& + x_4 \cdot (1 - x_3) + x_4 \cdot (1 - x_5) + x_5 \cdot (1 - x_4) - 6 \\
& + x_0 + x_1 + x_2 + x_3 + x_4 + x_5)
\end{aligned} \tag{23}$$

Based on the results, out of 420 parameter combinations, 82 produced z_* that satisfied the PDS condition, and 17 produced z_* corresponding to the optimal PDS. Figures 3 and 4 illustrate the probability distributions of bit strings under the conditions $q = 1$, $P_1 = 7.2$, $P_2 = 7.2$, and maximal iterations of 100 and 200, respectively. In these figures, the probabilities of z_* are highlighted in purple. Notably, in both figures, $z = 100010$ and $z = 010001$ emerge as the two most probable bit strings, which correspond exactly to the two optimal PDS for the graph depicted in Fig. 2 (Figs. 5, 6). This alignment demonstrates that the final sampling results capture the symmetry inherent to the graph. Furthermore, increasing the number of iterations tends to reduce the probabilities of non-optimal bit strings, such as $z = 100001$ and $z = 011110$ in Fig. 3. Although the performance of QAOA at low layers is limited, resulting in probabilities for $z = 100010$ and $z = 010001$ not being significantly higher than other bit strings, the experimental results suggest that with carefully chosen parameters, even low-layer QAOA can produce the expected results. These findings indicate that QAOA holds significant potential for solving the PDP.

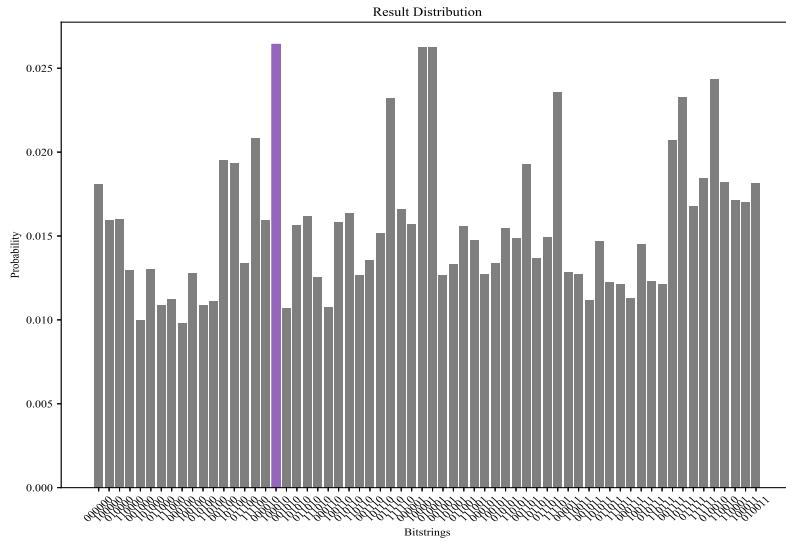


Figure 3: The probability distribution of bit strings when $q = 1$, $P_1 = 7.2$, $P_2 = 7.2$ and maximal iterations = 100.

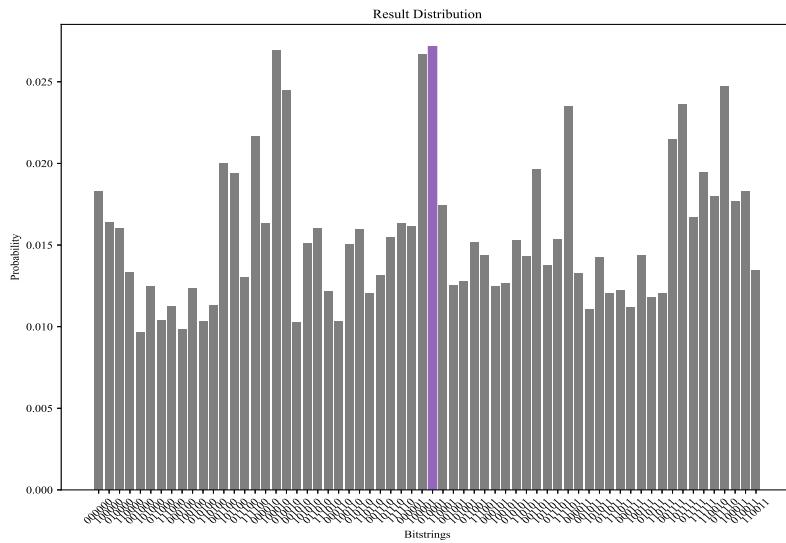


Figure 4: The probability distribution of bit strings when $q = 1$, $P_1 = 7.2$, $P_2 = 7.2$ and maximal iterations = 200.

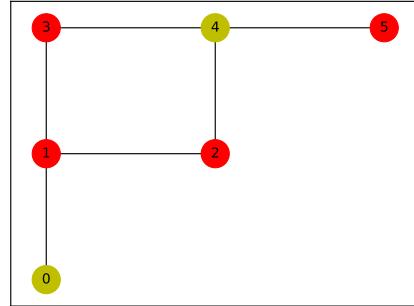


Figure 5: Visualization of the $z = 100010$ with PDS $\{0,4\}$

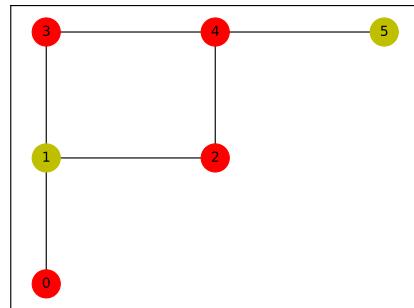


Figure 6: Visualization of the $z = 010001$ with PDS $\{1,5\}$.

To evaluate the convergence capability, we present the cost variation curves under different parameters in Figs. 7, 8, and 9. These figures reveal that the cost converges within approximately 20 iterations, highlighting a key advantage of employing low-layer QAOA to solve the PDP.

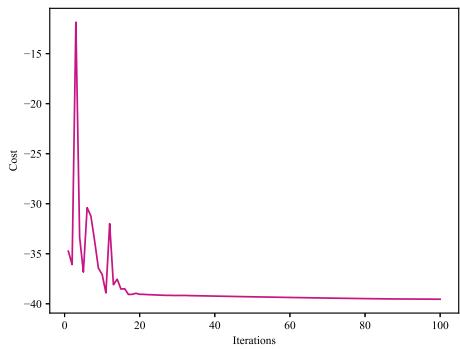


Figure 7: Cost of $q = 1$, $P_1 = 7.2$, $P_2 = 7.2$ and maximal iterations = 100.

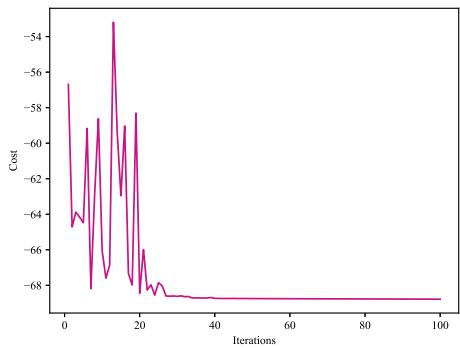


Figure 8: Cost of $q = 2$, $P_1 = 12$, $P_2 = 6$ and maximal iterations = 100.

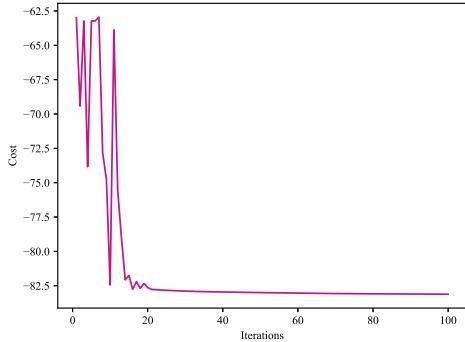


Figure 9: Cost of $q = 1$, $P_1 = 12$, $P_2 = 6$ and maximal iterations = 100.

Next, we computed and analyzed the approximation ratios for the 420 experiments. Given that the PDP is a constrained minimization combinatorial optimization problem, bit strings that do not satisfy the constraints were excluded when calculating the approximation ratio (Saleem et al., 2023). The approximation ratio formula used is presented in Eq. 24, where $|PDS_{opt}|$ denotes the size of the optimal PDS, i is the index of the bit string satisfying the PDS condition, c_i represents the number of samples for the i -th bit string, $|PDS_i|$ is the size of the PDS represented by the i -th bit string, and N_{total} is the total number of samples. In Fig. 10, we present the minimal, maximal, and average approximation ratios for different layer numbers. It is evident that as the number of layers increases, the approximation ratio exhibits an overall upward trend. When the number of layers reaches 5, the highest approximation ratio achieves approximately 0.9, clearly demonstrating the effectiveness of QAOA in approximating the solution to the PDS. Additionally, we observed that the upward trends for the minimal and average approximation ratios are less pronounced compared to the maximal approximation ratio. This disparity suggests that the maximal approximation ratio is more sensitive to the choice of parameters. Consequently, further analysis of the parameter dependency of the approximation ratio is crucial, as it would provide valuable insights for selecting optimal parameters to enhance the performance of QAOA in solving the PDP.

$$R = \frac{|PDS_{opt}|}{\left(\frac{\sum_i c_i \cdot |PDS|_i}{N_{total}}\right)} \quad (24)$$

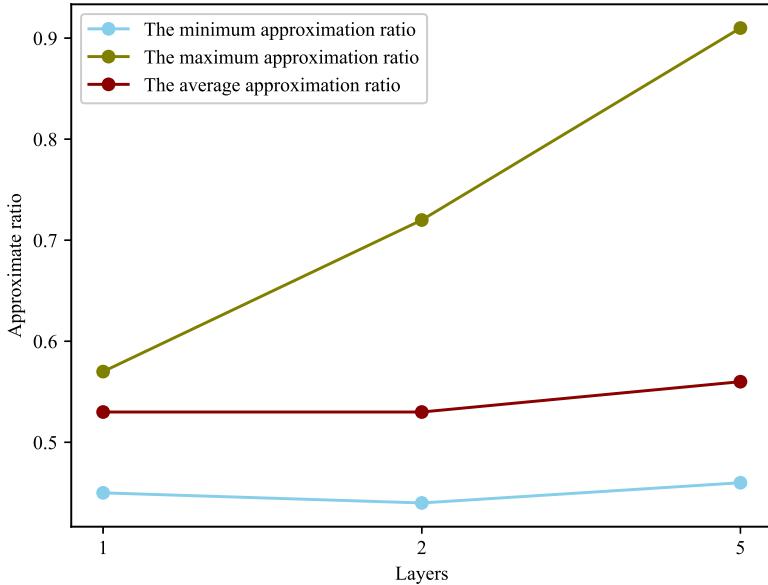


Figure 10: Minimal, maximal and average approximate ratios of different Layers.

We sorted the approximation ratios for the 420 experiments in descending order and selected the top 20% (84 experiments) with the highest approximation ratios for further analysis. For each parameter set, we counted the frequency of its occurrence among these 84 experiments. The statistical results are presented in Fig. 11. It is evident that experiments achieving higher approximation ratios are strongly correlated with higher layer numbers, while no clear trend is observed concerning the maximum number of iterations. Regarding the weight parameters, setting P_1 to 1.8 times $|V|$ and P_2 equal to P_1 appears to be more favorable for attaining higher approximation ratios. The observation that higher approximation ratios require higher values of P_1 and P_2 is reasonable. In the model, emphasizing the satisfaction of the DS condition (P_1) and the perfect condition (P_2) aligns with the pursuit of higher approximation ratios, as a PDS is, by definition, a DS that satisfies the perfect condition. However, when calculating the approximation ratio, erroneous bit strings must be excluded, so we cannot assume that experiments with the highest approximation ratios necessarily yield a higher probability for the optimal PDS compared to all other bit strings (including

non-PDS bit strings). Instead, we can infer that the probability of the optimal PDS has a relative advantage locally, compared to other non-optimal PDS probabilities. This concern stems from two main factors: (1) excessively high penalty weights can prevent the original objective function from converging to a desirable solution (Glover et al., 2022), and (2) constraint 4 involves overlapping qubits with all vertices in constraint 2, leading to mutual influence, particularly when the two constraints are assigned similar weights. Subsequent experimental results will further validate this analysis.

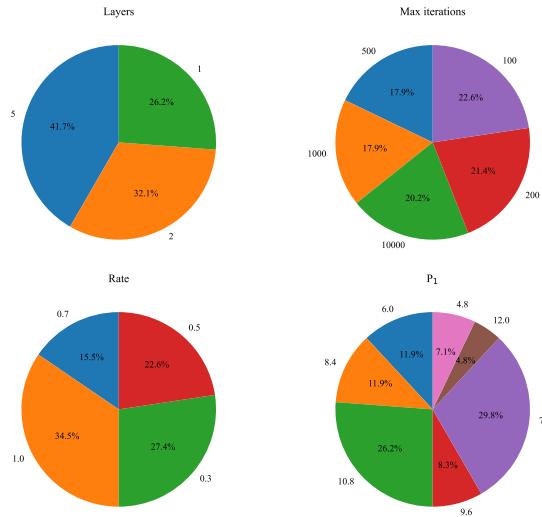


Figure 11: Analysis of the parameter distribution in the top 20% based on approximate ratio.

In Figs. 12 and 13, we present the statistical distribution of parameters for all experiments where z_* corresponds to either an optimal PDS or a correct PDS. The statistical data indicate that smaller layer numbers are more conducive to allowing the optimal PDS and correct PDS to dominate the overall probability distribution. Similar to the approximation ratio, these results appear to be independent of the maximum number of iterations. Regarding the weights, setting P_1 to 2 times the number of vertices and $\frac{P_2}{P_1} = 0.5$ provides a notable numerical advantage. In comparison to the approximation ratio, these results differ in terms of layer numbers and the ratio of P_2 to P_1 . Larger layer numbers may pose a challenge for the optimization algorithm, as the number of angles it must adjust increases proportionally, being twice

the number of layers. Consequently, higher layer numbers and larger penalty values are more likely to lead the algorithm into a local optimum. To improve the overall probability of obtaining the optimal PDS, selecting discriminative and well-tuned values for P_1 and P_2 might be a more effective approach.

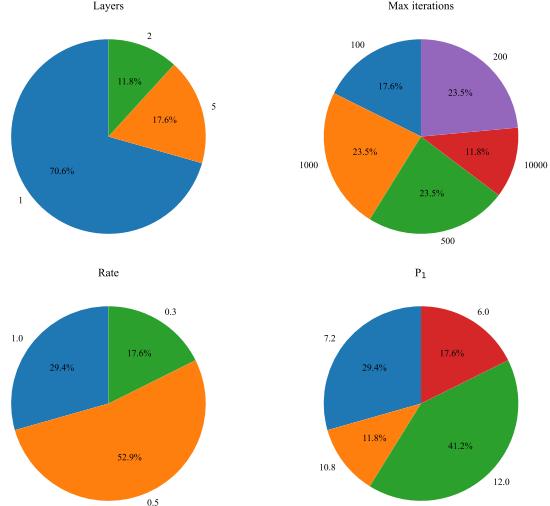


Figure 12: Analysis of the parameter distribution of the experiments of which z_* is optimal.

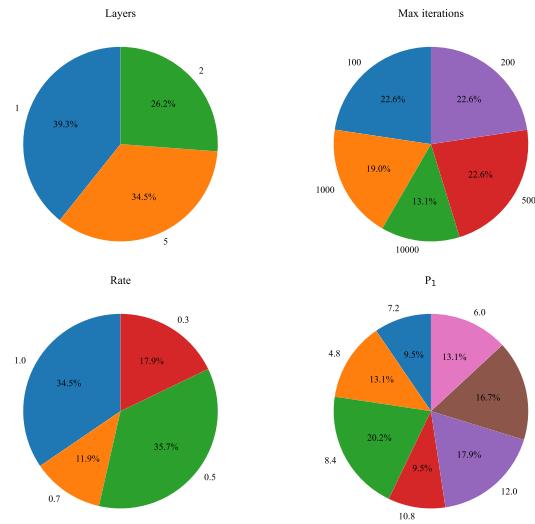


Figure 13: Analysis of the parameter distribution of the experiments of which z_* is correct.

Based on the analysis of Figs. 11, 12, and 13, we conclude that when using QAOA to solve the PDP, pursuing a higher approximation ratio and increasing the probability of sampling the optimal PDS can be two distinct objectives. To enable QAOA to directly output z_* as a PDS, particularly the optimal PDS, it is advisable to use a smaller number of layers and assign distinct weights to the two types of penalties. Conversely, if an additional filtering step can be applied to the final sampling results, a larger number of layers can be used, with both penalty weights set to equal and higher values. This approach would be more effective in achieving a higher approximation ratio.

5. Conclusion

In this paper, we explored the use of QAOA to solve the PDP. We began by modeling the PDP as a 0-1 integer programming problem based on its definition and transformed its two types of constraints into quadratic penalties to derive the QUBO model. Through variable substitution, we ultimately obtained the Hamiltonian for the PDP. Using IBM’s Qiskit quantum simulator, we conducted extensive tests and analyses on the performance of QAOA in solving the PDP, testing 420 parameter combinations involving layer numbers, penalty coefficients, and maximum iterations. On a macro scale, QAOA successfully computed the correct PDS in 82 parameter combinations and identified the optimal PDS in 17 combinations, achieving a highest approximation ratio of approximately 0.9. These results confidently demonstrate that even with low-layer QAOA, excellent outcomes can be achieved with appropriately chosen parameters. This underscores the significant potential of quantum algorithms in solving the PDP. In the parameter analysis, we examined the parameter distributions for three categories of experiments: (1) those in the top 20% for approximation ratio, (2) those where z_* corresponds to the optimal PDS, and (3) those where z_* corresponds to the correct PDS. The findings revealed distinct parameter tendencies between the first category and the latter two. Higher approximation ratios were associated with larger layer numbers and higher penalty coefficients for both constraints. In contrast, the latter two categories favored smaller layer structures and distinct penalty coefficient settings. These insights provide valuable guidance for optimizing parameter settings when applying QAOA to solve the PDP, paving the way for further advancements in quantum algorithm applications.

The limitations of this paper include: (1) the algorithm was not tested on

a real quantum computer, and (2) the quantum circuit was not optimized. All quantum circuits used in this study were generated using IBM’s Qiskit.

Based on the research content, experimental results, and limitations discussed in this paper, we propose the following directions for future work: (1) Evaluate the effectiveness of QAOA in solving the PDP on a real quantum computer. (2) Incorporate the noise environment into the QAOA algorithm and optimize the quantum circuit accordingly. (3) Extend the application of QAOA to address other variants of the DP, such as the k -domination problem and related challenges.

Acknowledgement

This paper was inspired by the GitHub project named qopt-best-practices [<https://github.com/qiskit-community/qopt-best-practices.git>] when using QAOA. At the same time, the quantum programming tools we used are all from IBM’s Qiskit package.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of interest

The authors declare that they have no conflicts of interest that could have appeared to influence the work reported in this paper.

Funding statement

This work is supported by National Natural Science Foundation of China (No. 12331014).

Declaration of Generative AI

During the preparation of this work the authors used chatgpt in order to improve readability and language. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

References

- Caleffi, M., Amoretti, M., Ferrari, D., Illiano, J., Manzalini, A., Cacciapuoti, A.S., 2024. Distributed quantum computing: a survey. *Computer Networks* 254, 110672.
- Chang, G.J., Rangan, C.P., Coorg, S.R., 1995. Weighted independent perfect domination on cocomparability graphs. *Discrete Applied Mathematics* 63, 215–222.
- Chang, M.S., Liu, Y.C., 1993. Polynomial algorithms for the weighted perfect domination problems on chordal graphs and split graphs. *Information Processing Letters* 48, 205–210.
- Dejter, I.J., Delgado, A.A., 2007. Perfect domination in rectangular grid graphs. arXiv preprint arXiv:0711.4345 .
- Farhi, E., Goldstone, J., Gutmann, S., 2014. A quantum approximate optimization algorithm. URL: 10.48550/arXiv.1411.4028.
- Fei, Y., 2020. Study on neutrosophic graph with application in wireless network. *CAAI Transactions on Intelligence Technology* 5, 301–307.
- Fellows, M.R., Hoover, M.N., 1991. Perfect domination. *Australas. J Comb.* 3, 141–150.
- Fenstermacher, T., Ganguly, S., Laskar, R., 2018. Perfect domination in knights graphs. arXiv preprint arXiv:1805.03335 .
- Finnila, A.B., Gomez, M.A., Sebenik, C., Stenson, C., Doll, J.D., 1994. Quantum annealing: A new method for minimizing multidimensional functions. *Chemical physics letters* 219, 343–348.
- Glover, F., Kochenberger, G., Hennig, R., Du, Y., 2022. Quantum bridge analytics i: a tutorial on formulating and using qubo models. *Annals of Operations Research* 314, 141–183.
- Guerrero, N.J., 2020. Solving combinatorial optimization problems using the quantum approximation optimization algorithm .
- Hamja, J., 2023. Certified perfect domination in graphs. *European Journal of Pure and Applied Mathematics* 16, 2763–2774.

- Hsieh, S.Y., 2007. An efficient parallel strategy for the perfect domination problem on distance-hereditary graphs. *The Journal of Supercomputing* 39, 39–57.
- Krauss, T., McCollum, J., Pendery, C., Litwin, S., Michaels, A.J., 2020. Solving the max-flow problem on a quantum annealing computer. *IEEE Transactions on Quantum Engineering* 1, 1–10.
- Lin, M.C., Mizrahi, M.J., Szwarcfiter, J.L., 2015. Efficient and perfect domination on circular-arc graphs. *Electronic Notes in Discrete Mathematics* 50, 307–312.
- Livingston, M., Stout, Q.F., 1990. Perfect dominating sets. Citeseer.
- O'brien, J.L., 2007. Optical quantum computing. *Science* 318, 1567–1570.
- Pan, H., Lu, C., 2024a. Qubo formulations for variation of domination problem. arXiv preprint arXiv:2410.21277 .
- Pan, H., Lu, C., 2024b. Solving the independent domination problem by quantum approximate optimization algorithm. arXiv preprint arXiv:2410.17227 .
- Pan, H., Wang, S., Lu, C., 2024. Application of quantum approximate optimization algorithm in solving the total domination problem. arXiv preprint arXiv:2411.00364 .
- Preskill, J., 2023. Quantum computing 40 years later. CRC Press. pp. 193–244.
- Radzhovskiy, M., Murphy, J., Mason, S., 2019. A qaoa solution to the traveling salesman problem using pyquil.
- Raussendorf, R., Briegel, H.J., 2001. A one-way quantum computer. *Physical review letters* 86, 5188. URL: <https://journals.aps.org/prl/abstract/10.1103/PhysRevLett.86.5188>.
- Sack, S.H., Serbyn, M., 2021. Quantum annealing initialization of the quantum approximate optimization algorithm. *quantum* 5, 491.

- Saleem, Z.H., Tomesh, T., Tariq, B., Suchara, M., 2023. Approaches to constrained quantum approximate optimization. SN Computer Science 4, 183.
- Tilly, J., Chen, H., Cao, S., Picozzi, D., Setia, K., Li, Y., Grant, E., Wossnig, L., Rungger, I., Booth, G.H., 2022. The variational quantum eigensolver: a review of methods and best practices. Physics Reports 986, 1–128.
- Yen, C.C., Lee, R.C.T., 1990. The weighted perfect domination problem. Information Processing Letters 35, 295–299.
- Yu, Y., Cao, C., Wang, X.B., Shannon, N., Joynt, R., 2023. Solution of sat problems with the adaptive-bias quantum approximate optimization algorithm. Physical Review Research 5, 023147.
- Zhang, Y.J., Mu, X.D., Liu, X.W., Wang, X.Y., Zhang, X., Li, K., Wu, T.Y., Zhao, D., Dong, C., 2022. Applying the quantum approximate optimization algorithm to the minimum vertex cover problem. Applied Soft Computing 118. doi:10.1016/j.asoc.2022.108554.