Recursive Quantum Relaxation for Combinatorial Optimization Problems

Ruho Kondo[1,2], Yuki Sato[1,2], Rudy Raymond[2,3], and Naoki Yamamoto[2,4]

[1]Toyota Central R&D Labs., Inc., 41-1, Yokomichi, Nagakute, Aichi 480-1192, Japan
[2]Quantum Computing Center, Keio University, 3-14-1 Hiyoshi, Kohoku-ku, Yokohama, Kanagawa 223-8522, Japan
[3]Department of Computer Science, The University of Tokyo, 7-3-1, Hongo, Bunkyo-ku, Tokyo 113-0033, Japan
[4]Department of Applied Physics and Physico-Informatics, Keio University, Hiyoshi 3-14-1, Kohoku-ku, Yokohama 223-8522, Japan

Quantum optimization methods use a continuous degree-of-freedom of quantum states to heuristically solve combinatorial problems, such as the MAX-CUT problem, which can be attributed to various NP-hard combinatorial problems. This paper shows that some existing quantum optimization methods can be unified into a solver to find the binary solution which is most likely measured from the optimal quantum state. Combining this finding with the concept of quantum random access codes (QRACs) for encoding bits into quantum states on fewer qubits, we propose an efficient recursive quantum relaxation method called recursive quantum random access optimization (RQRAO) for MAX-CUT. Experiments on standard benchmark graphs with several hundred nodes in the MAX-CUT problem, conducted in a fully classical manner using a tensor network technique, show that RQRAO not only outperforms the Goemans–Williamson and recursive QAOA methods, but also is comparable to state-of-the-art classical solvers. The code is available at `https://github.com/ToyotaCRDL/rqrao`.

1 Introduction

The MAX-CUT problem is one of Karp's 21 NP-complete problems [4] and is frequently used as a benchmark of combinatorial optimization algorithms because of its simplicity. Namely, given a graph that consists of nodes connected by weighted edges, the optimization variant of the MAX-CUT problem asks for the labeling of nodes into two classes such that the sum of the weighted edges connecting nodes with different labels is maximized. Obtaining good solutions for the MAX-CUT problem is in great demand, because various NP-hard combinatorial problems, e.g., partitioning, covering, and coloring problems, can be transformed into the MAX-CUT problem [5, 6]. Although the MAX-CUT problem is a discrete optimization problem, it can be solved by relaxing the discrete variables into continuous ones. For example, the semidefinite program in [7] finds an embedding of binary variables as real-valued vectors in a high-dimensional sphere so that the labels can be probabilistically determined by partitioning the vectors with a random hyperplane cutting. (see Supplementary Information H.1). However, the quantum optimization does not take such an approximation because the probabilistic characteristics are naturally incorporated into the quantum states. In most approaches to the MAX-CUT problem using quantum algorithms, all solutions are encoded in the eigenstates of the problem Hamiltonian, and then the quantum state is optimized such that the expected value of the problem Hamiltonian is maximized (see Supplementary Information H.2). From a theoretical viewpoint, it has been proven that the eigenstate with the maximum eigenvalue can be obtained by the infinite-level quantum approximate optimization algorithm (QAOA) [8, 9] with the appropriate parameters. However, in a noisy quantum device, it is intractable to carry out even a sufficiently large level QAOA, let alone an in-

Ruho Kondo: r-kondo@mosk.tytlabs.co.jp
Rudy Raymond: Former affiliation was IBM Quantum, IBM Japan.

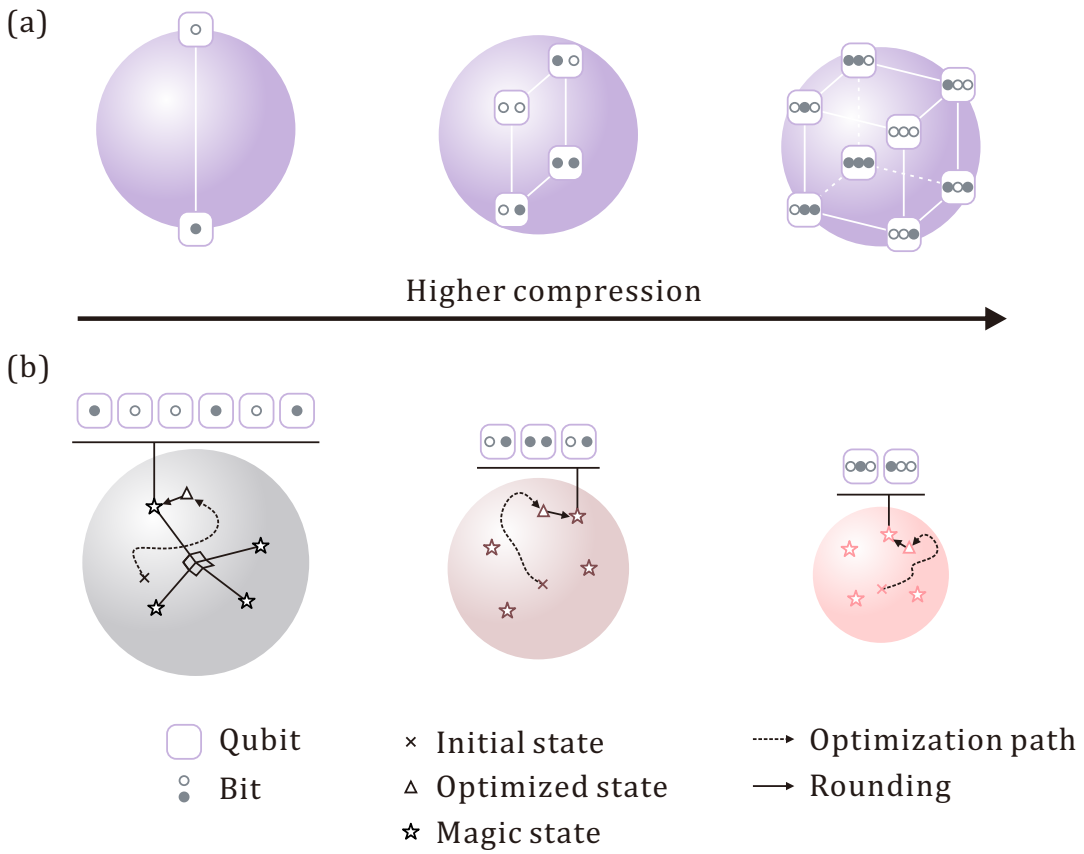


Figure 1: Concept of the proposed method, recursive quantum random access optimization (RQRAO). (a) One/two/three classical bits are embedded into one qubit using a quantum random access code (QRAC) [1, 2]. A total of two/four/eight variants of the information of the one/two/three classical bits are represented as a vector pointing to the corner of the line/square/cube inscribed in the Bloch sphere. These specific quantum states are called (one-qubit) magic states. (b) Using a high compression QRAC, the size of the Hilbert space corresponding to the given problem is reduced and each (multi-qubit) magic state is no longer necessarily orthogonal to the others. Note that only four of the 2^6 magic states are illustrated. After optimizing the objective function defined on the reduced Hilbert space, the bit string is decoded by searching for the nearest magic state as measured in the fidelity, which is called *rounding*. To accomplish a good rounding, a recursive strategy similar to the RQAOA [3] is adopted.

finite level QAOA. For this reason, the field of combinatorial problems with quantum computing is dominated by the development of heuristics, that mainly focus on the development of the ansatz variants and improving the trainability of QAOA [10].

Rather than developing the QAOA ansatz, an alternative approach is to develop a non-diagonal Hamiltonian instead of the diagonal Ising-type one used in the QAOA. In [11], the MAX-CUT Hamiltonian is constructed using quantum random access codes (QRACs) [1, 2] (see Sec. 2.3 for details, and Supplementary Information H.3 for a review). Using QRACs, the n-qubit quantum state can be used to encode m binary variables for $m > n$. The QRAC Hamiltonian is still a 2-local Hamiltonian, and therefore it is QMA-hard to find the ground state [12]. However, the QRAC formulation has the advantage that the number of qubits can be reduced, making it easier to optimize the variational state.

Recently, [3] proposed the recursive QAOA (RQAOA), showing that the recursive application of level-1 QAOA gives an empirically higher approximation ratio than the naïve application of level-1 QAOA and the Goemans–Williamson (GW) algorithm [7], where the latter is the best generic classical algorithm with a theoretical guarantee for the MAX-CUT problem under the unique games conjecture [13]. This result is interesting because in [3], it was proved that the approximation ratio of the naïve level-1 QAOA cannot outperform the GW for some specific graphs. The application of the RQAOA has been extended to the MAX-k-CUT problem in [14], which also shows that the RQAOA out-

performs the best-known classical approximation algorithms.

Here, the following questions naturally arise: Is it possible to apply the QRAC in the RQAOA framework? Furthermore, if it is possible, are there any benefits of using the QRAC in the RQAOA framework, such as improved approximation ratios? In this paper, we propose a general framework for the MAX-CUT problem that incorporates the QRAC [11] into the RQAOA [3] framework, namely *recursive quantum random access optimization (RQRAO, see Fig. 1)*. Our contributions are summarized as follows:

1. We theoretically show that incorporating QRACs into the RQAOA framework is a natural extension of [11] and [3], i.e., they are all uniformly expressed as a method that (i) maximizes the expectation of the problem Hamiltonian so that all candidate solutions are embedded using QRACs, and (ii) searches for the nearest embedding as measured in the fidelity (Theorem 1).

2. We give numerical experiments to show the benefit of incorporating QRACs into the RQAOA framework. Demonstrations on an open graph dataset, Gset [15], were carried out using the tensor network simulation conducted on a classical computer. The results showed that the approximation ratio of the proposed method outperforms not only GW but also the methods from which inspiration was taken [3, 11], and is comparable to existing state-of-the-art classical heuristics [16].

Note that the proposed framework is entirely classical except for the ground state preparation step, and therefore it may not provide the so-called quantum advantage. However, when an efficient quantum algorithm is used for the ground-state preparation of the framework, the overall runtime can be improved.

The remainder of the paper is organized as follows: Section 2 presents the mathematical definition of the MAX-CUT problem and describes the RQAOA and QRAC in brief. Section 3 describes the proposed algorithm and its theoretical background. Section 4 presents the results of numerical experiments. Section 5 concludes the paper.

2 Preliminaries

2.1 MAX-CUT Problem

As stated in Sec. 1, the goal of the MAX-CUT problem is to find binary node labels such that the sum of the weights of edges with different node labels is maximized. It is mathematically formulated as follows:

Definition 1 (MAX-CUT problem) *Given a graph $G = (V, E)$, where V is a node set and E is an edge set with edge weights $w_{jk} \in \mathbb{R}$ for $(j, k) \in E$, the MAX-CUT problem is defined as*

$$\max_{\mathbf{b}\in\{0,1\}^{|V|}} \mathit{CW}(\mathbf{b}) \tag{1}$$

where

$$\mathit{CW}(\mathbf{b}) := \sum_{(j,k)\in E} w_{jk} \frac{1-(-1)^{b_j+b_k}}{2} \tag{2}$$

is the cut weight corresponding to $\mathbf{b} = (b_0, \ldots, b_{|V|-1})$, where $b_i \in \{0, 1\}$ is an attribute of node i.

2.2 Recursive QAOA

Recursive QAOA (RQAOA) [3] is a variant of QAOA for the MAX-CUT problem, which is an iterative method that determines the parity of one edge in a graph per iteration, deletes a node connected by the edge and modifies the graph. Here, edge parity is defined as follows:

Definition 2 (Edge parity) *Given edge (j, k), positive and negative edges are defined as $b_j = b_k$ and $b_j \neq b_k$, respectively. The edge parity is said positive (negative) for a positive (negative) edge.*

In the first step of RQAOA, a candidate edge whose parity is to be determined is calculated using optimization. Let $|\psi(\boldsymbol{\theta})\rangle$ be a QAOA ansatz with learnable parameters $\boldsymbol{\theta}$ and

$$H_{\text{Ising}} := \sum_{(j,k)\in E} w_{jk} \frac{I - Z_j Z_k}{2} \tag{3}$$

be the Ising-type MAX-CUT Hamiltonian, where Z_j is a Pauli Z matrix corresponding to the jth qubit. The optimized parameters $\boldsymbol{\theta}^*$ are obtained by maximizing $\langle\psi(\boldsymbol{\theta})|H_{\text{Ising}}|\psi(\boldsymbol{\theta})\rangle$. After optimization, the energy of edge $(j, k) \in E$ is calculated as

$$\mathcal{E}_{jk} := \langle\psi(\boldsymbol{\theta}^*)|Z_j Z_k|\psi(\boldsymbol{\theta}^*)\rangle. \tag{4}$$

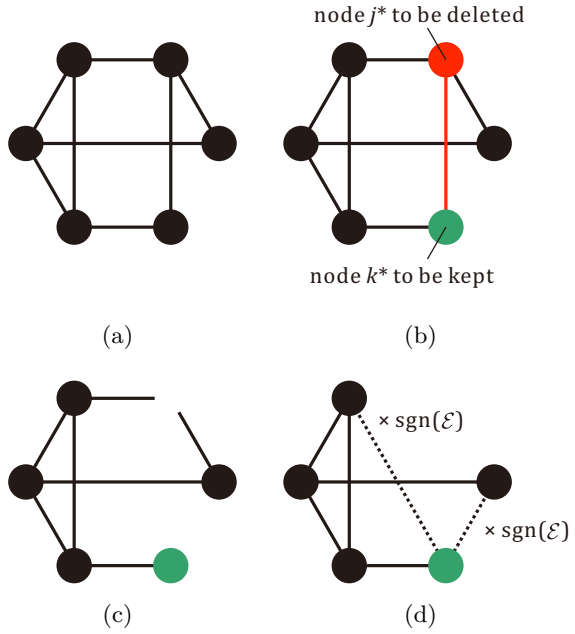


Figure 2: Graph modification procedure of RQAOA. (a) Initial graph. (b) The red edge indicates $(j^*, k^*) = \mathrm{argmax}_{(j,k)}|\mathcal{E}_{jk}|$, where $|\mathcal{E}_{jk}| := |\langle\psi|Z_jZ_k|\psi\rangle|$. One of the nodes in edge (j^*, k^*) is kept (the green node) while the other one is deleted (the red node). (c) After node removal. (d) Edges are re-connected to the retained node by multiplying $\mathrm{sgn}(\mathcal{E}_{j^*k^*})$ by its weight.

The candidate edge (j^*, k^*) is defined as a maximal energy edge as

$$(j^*, k^*) = \underset{(j,k)\in E}{\mathrm{argmax}}\, |\mathcal{E}_{jk}|. \tag{5}$$

Then, the parity of edge (j^*, k^*) is determined as

$$\begin{cases} b_{j^*} = b_{k^*} & \mathcal{E}_{j^*k^*} > 0 \\ b_{j^*} \neq b_{k^*} & \mathcal{E}_{j^*k^*} < 0 \end{cases}. \tag{6}$$

Note that the probability that all edge energy is equal to zero ($\mathcal{E}_{j^*k^*} = 0$) is negligibly small and can be ignored.

In the next step, the problem Hamiltonian is modified to satisfy Eq. (6) as a constraint. This modification can be interpreted as modifying the graph such that the number of its nodes is reduced by one, as shown in Fig. 2. Node j^* is deleted from the graph and edge (j^*, l) $(l \neq k^*)$ is re-connected as (k^*, l). At the same time, edge weight w_{k^*l} is updated to $w_{k^*l} + \mathrm{sgn}(\mathcal{E}_{j^*k^*}) \cdot w_{j^*l}$ $(l \neq k^*)$. As a result, one node is deleted from the graph by a single iteration. Subsequently, the variational state is re-initialized and optimized using the Hamiltonian defined with respect to the modified graph.

The number of nodes is gradually decreased by repeating the edge parity determination and graph size reduction. When the number of nodes becomes smaller than a threshold value M, the parities of the remaining nodes are determined by brute-force search. Once all parities are determined, a label for one node is randomly set to either 0 or 1. Subsequently, the labels for all other nodes are determined based on the assigned parities.

2.3 Quantum Random Access Optimization

In the Quantum Random Access Optimization (QRAO) [11], binary node attributes are encoded in fewer qubits than the number of nodes using quantum random access code (QRAC) [1, 2]. For each node $j \in V$, binary node attribute $b_j \in \{0, 1\}$ is randomly mapped to $(q^{(j)}, P^{(j)})$, where $q^{(j)}$ is a qubit index and $P^{(j)}$ is a Pauli matrix with the following constraints:

$$\begin{cases} P^{(j)} \neq P^{(k)} & \text{for} \quad q^{(j)} = q^{(k)} \\ q^{(j)} \neq q^{(k)} & \forall \quad (j,k) \in E \end{cases}. \tag{7}$$

In the quantum random access coding, there are variations that depend on the number of variants of the Pauli matrices. If only Z is available, $P^{(j)} = Z$, it reduces to the Ising-type formulation, which we refer to $(1, 1)$-QRAC. For $(2, 1)$-QRAC, two variants of 1-local Pauli matrices, e.g., $P^{(j)} \in \{X, Z\}$, are available. For $(3, 1)$-QRAC, $P^{(j)} \in \{X, Y, Z\}$ are available. Then, the relaxed MAX-CUT Hamiltonian, which we refer to as the $(m, 1)$-QRAC Hamiltonian $(m = 1, 2, 3)$, is defined as

$$H_m := \sum_{(j,k)\in E} w_{jk} \frac{I - mP_{\langle j\rangle}P_{\langle k\rangle}}{2}, \tag{8}$$

where $P_{\langle j\rangle} := P^{(j)}_{q^{(j)}}$ is a 1-local Pauli matrix corresponding to the $q^{(j)}$th qubit. If the given graph is sparse, the second constraint in Eq. (7) rarely restricts the assignment, and hence, the total number of qubits can be reduced by up to $1/m$-th the number of nodes.

Let $\mu_m^{[q]} \in \mathbb{C}^{2\times 2}$ be the qth single qubit pure state in the form of a density matrix defined as

$$\mu_m^{[q]}(\{b_\bullet^{[q]}\}) := \frac{1}{2}\left(I + \frac{1}{\sqrt{m}} \sum_{P\in\mathfrak{P}} (-1)^{b_P^{[q]}} P\right), \tag{9}$$

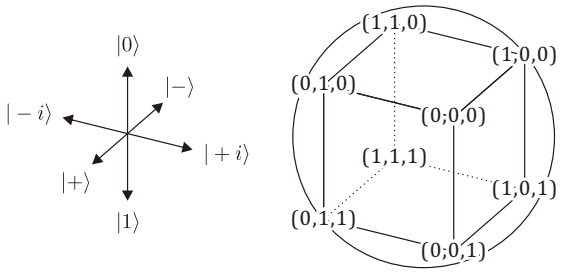


Figure 3: Representation of the $(3,1)$-magic state in the Bloch sphere. The coordinates of the vertex of the cube inscribed in the sphere correspond to the embedded classical bits $(b_X^{[q]}, b_Y^{[q]}, b_Z^{[q]})$.

where $\mathfrak{P} \subset \{X,Y,Z\}$, $|\mathfrak{P}| = m$, and $b_P^{[q]} \in \{0,1\}$ is a binary node attribute mapped to (q,P). Because $\mu_m^{[q]}(\{b_\bullet^{[q]}\})$ is a one-qubit pure state, it can be drawn in the Bloch sphere (Fig. 3). For example, in the case of $m=3$, the states $\mu_3^{[q]}(\{b_\bullet^{[q]}\})$ for $(b_X^{[q]}, b_Y^{[q]}, b_Z^{[q]}) \in \{0,1\}^3$ correspond to the vectors pointing the vertices of the cube inscribed in the Bloch sphere. The n-qubit product state of $\mu_m^{[q]}$ is defined as

$$\mu_m(\mathbf{b}) := \mu_m^{[n-1]}(\{b_\bullet^{[n-1]}\}) \otimes \cdots \otimes \mu_m^{[0]}(\{b_\bullet^{[0]}\}). \tag{10}$$

According to [11, Proposition 1], the following property holds.

$$\mathrm{tr}(H_m\, \mu_m(\mathbf{b})) = \mathtt{CW}(\mathbf{b}). \tag{11}$$

Of course, the n-qubit quantum state is not required to be a product state such as Eq. (10). If ρ is assumed to be an arbitrary n-qubit state that includes the aforementioned product state, $\mathrm{tr}(H_m\, \rho)$ could be higher than $\mathrm{tr}(H_m\, \mu(\mathbf{b}))$ when $m > 1$, where ρ may not be directly related to an optimal binary solution. This is why H_m $(m > 1)$ is called the *relaxed* Hamiltonian. Note that it has not been clarified what exactly $\mathrm{tr}(H_m\, \rho)$ represents when ρ is an arbitrary state in [11], but this will be clarified in Corollary 1 in Sec. 3.2.

After maximizing $\mathrm{tr}(H_m \rho)$ with respect to ρ, the classical bit string $\mathbf{b}$ is decoded from the optimized ρ. This process is called *rounding*. In [11], two kinds of rounding were proposed, magic state rounding and Pauli rounding. Note that magic state rounding is the projective measurement whose basis is the magic state selected uniformly at random. The detailed protocol is defined in Definition 3 of Supplementary Information A, and as can be seen to closely resemble the classical shadow with Random Pauli measurements [17]. For this reason, the magic state rounding is henceforth referred to as *random magic measurement*. Note that $(1,1)$-random magic measurement with $\mathfrak{P} = \{Z\}$ is equivalent to measurements on a computational basis. Pauli rounding is a decoding method based on the value of $\mathrm{tr}(P_{\langle j \rangle}\rho)$. The value of $\mathbf{b}$ is determined as

$$b_j = \begin{cases} 0 & (\mathrm{tr}(P_{\langle j \rangle}\rho) > 0) \\ 1 & (\mathrm{tr}(P_{\langle j \rangle}\rho) < 0) \end{cases}. \tag{12}$$

If $\mathrm{tr}(P_{\langle j \rangle}\rho) = 0$, b_j is determined as 0 or 1 uniformly at random. In Sec. 3.2, we provide theorems to show the relationship between random magic measurements and Pauli rounding and reveal what assumption is made in Pauli rounding.

3 Proposed Method

The proposed method, named RQRAO, is introduced in this section.

Section 3.1 describes the concrete algorithm of the proposed method. In brief, the proposed method incorporates the $(m,1)$-QRAC Hamiltonian (Eq. (8)) into the RQAOA framework (Sec. 2.2). The edge parity is recursively determined based on the value of the edge energy, which is defined using the $(m,1)$-QRAC Hamiltonian. Two new heuristics are also incorporated into the proposed method: the ensemble method of defining the edge energy, and determining several edge parities at once using the maximum spanning tree. The former improves the resulting cut weight whereas the latter reduces the runtime of the proposed method.

In Sec. 3.2, we show that the proposed method, RQAOA, and QRAO

1. maximize the expected cut weight using $(m,1)$-random magic measurements with respect to the quantum state, and
2. decode the bit string embedded in the nearest magic state as measured in the fidelity against the optimized quantum state.

3.1 Algorithm

The pseudocode of RQRAO is presented in Algorithm 1. The algorithm describes how the *ensemble edge energy* $\mathfrak{E}_{jk}$ of edge (j,k) is calculated

and how the edge parity is calculated from $\mathfrak{E}_{jk}$. The details are described below.

Line 1: Add Noise to Edges Add small perturbation noise to all edge weights to mitigate the *isolated nodes problem* (see Supplementary Information B). We note that the isolated nodes problem also occurs at RQAOA [3] although not reported in the previous literature. We found out the perturbation is effective to deal with the problem.

Line 4: Define the Relaxed Hamiltonian Assign 1-local Pauli $P^{(t)}$ to each node included in the current graph $G' = (V', E')$, where superscript (t) describes the tth trial. Note that t is the sample index of the ensemble, not to be confused with the recursive iteration index.

The assignment is randomly carried out so as not to violate constraints Eq. (7). Then, the tth relaxed Hamiltonian is defined as

$$H_m^{(t)} := \sum_{(j,k)\in E'} \frac{I - mP_{\langle j\rangle}^{(t)} P_{\langle k\rangle}^{(t)}}{2} w'_{jk}. \tag{13}$$

Line 5: Optimize the Quantum State Define ansatz $|\psi(\boldsymbol{\theta})\rangle$, where $\boldsymbol{\theta} \in \mathbb{R}^K$ are K learnable parameters. For example, we used the matrix product state ansatz in the numerical experiments, where all matrix elements were treated as the learnable parameters (see Implementation Details in Sec. 4). The objective function is defined as

$$\mathcal{L}^{(t)}(\boldsymbol{\theta}) := \langle\psi(\boldsymbol{\theta})|H_m^{(t)}|\psi(\boldsymbol{\theta})\rangle. \tag{14}$$

The optimized quantum state $|\psi(\boldsymbol{\theta}^*)\rangle$ is obtained by maximizing $\mathcal{L}^{(t)}(\boldsymbol{\theta})$. Any optimization method for the parameters of the ansatz can be used here.

Line 6: Compute the Edge Energy The tth edge energy is computed as

$$\mathcal{E}_{jk}^{(t)} := \langle\psi(\boldsymbol{\theta}^*)|P_{\langle j\rangle}^{(t)} P_{\langle k\rangle}^{(t)}|\psi(\boldsymbol{\theta}^*)\rangle. \tag{15}$$

The resulting edge energy strongly depends on both the assignment of 1-local Pauli $P^{(t)}$ and the reached quantum state, which is in general not a eigenstate corresponding to the maximum eigenvalue of $H_m^{(t)}$.

Lines 8–12: Compute the Ensemble Edge Energy Using N edge energies, $\{\mathcal{E}_{jk}^{(t)}\}_{t=1}^N$, the ensemble edge energy is calculated as

$$\mathfrak{E}_{jk} := \mu_{jk} - \mathrm{sign}(\mu_{jk}) \cdot \min(S\sigma_{jk}, |\mu_{jk}|) \tag{16}$$

where μ_{jk} and σ_{jk} are the mean and standard deviation of $\{\mathcal{E}_{jk}^{(t)}\}_{t=1}^N$, which are defined as $\mu_{jk} := \frac{1}{N}\sum_{t=1}^N \mathcal{E}_{jk}^{(t)}$ and $\sigma_{jk} := \sqrt{\frac{1}{N}\sum_{t=1}^N (\mathcal{E}_{jk}^{(t)} - \mu_{jk})^2}$, respectively , and S is a hyperparameter designed such that if $|\mu_{jk}| \le S\sigma_{jk}$, then $\mathfrak{E}_{jk} = 0$.

The reason for taking an ensemble is twofold. First, unlike QAOA whose edge energy is solely from the ZZ interaction as in Eq. (4), the edge energy of QRAO can be defined by 9 different pairs of 1-local Pauli as in Eq. (15) whose corresponding Hamiltonian and ground state can be significantly different: some may be easier to compute with certain ansatz than others. The ensemble is utilized to take advantage of the various Hamiltonians and ground states. Second, taking an ensemble average of the edge energy from different pairs of 1-local Pauli is akin to *bagging* [18] in machine learning. Namely, even if an individual pair of 1-local Pauli is a weak predictor of the edge parity, their averages can give higher predictive performance when they are weakly correlated with each other. In our method, when optimization is performed on each of the N different $(m, 1)$-QRAC Hamiltonians created from different Pauli assignments, N quantum states are obtained. Since these quantum states are optimized with respect to different Hamiltonians as objective functions, they can be considered analogous to the weak predictors mentioned above. Therefore, we expect an ensemble edge energy from these quantum states will yield a stronger predictor of the edge parity.

Line 14: Compute the Maximum Spanning Tree After computing the ensemble edge energy $\mathfrak{E}_{jk}$, the parities of edges with nonzero $\mathfrak{E}_{jk}$ are determined according to the sign of $\mathfrak{E}_{jk}$. However, sometimes inconsistency arises. For example, this occurs when three negative edges form a cycle $\{(b_0, b_1), (b_1, b_2), (b_2, b_0)\}$, one of which is inconsistent because $(b_0 \neq b_1) \wedge (b_1 \neq b_2) \wedge (b_2 \neq b_0)$ cannot hold. To avoid this inconsistency, we adopt the maximum spanning tree to remove any cycles from the candidates.

Edges with nonzero $\mathfrak{E}_{jk}$ are collected and $|\mathfrak{E}_{jk}|$ are treated as edge weights. For each subgraph

Algorithm 1: Algorithm for recursive quantum random access optimization (RQRAO).

input : Graph $G = (V, E)$, number of ensembles N, scale factor S, brute-force search threshold M
output : MAX-CUT solution $\mathbf{b}^*$
initialize: $\mathcal{P} \leftarrow \emptyset$, $G' = (V', E') \leftarrow G = (V, E)$
1 $w'_{jk} \leftarrow w_{jk} + \xi, \quad \xi \sim \mathrm{Uniform}([-10^{-5}, 10^{-5}])$;
2 **while** $|V'| > M$ **do**
  `/* Compute the ensemble edge energy */`
3 **for** $t = 1, \ldots, N$ **do**
4 Assign Pauli $P^{(t)}$ to each node $j \in V'$ and make the $(m, 1)$-QRAC Hamiltonian $H_m^{(t)}(G')$;
5 $\theta^{(t)*} \leftarrow$ Result of maximization of $\langle\psi(\theta)|H_m^{(t)}(G')|\psi(\theta)\rangle$;
6 $\mathcal{E}_{jk}^{(t)} \leftarrow \langle\psi(\theta^{(t)*})|P_{\langle j\rangle}^{(t)} P_{\langle k\rangle}^{(t)}\,|\psi(\theta^{(t)*})\rangle$ for all $(j, k) \in E'$;
7 **end**
8 **for** $(j, k) \in E'$ **do**
9 $\mu_{jk} \leftarrow \mathtt{Mean}(\{\mathcal{E}_{jk}^{(t)}\})$;
10 $\sigma_{jk} \leftarrow \mathtt{StandardDeviation}(\{\mathcal{E}_{jk}^{(t)}\})$;
11 $\mathfrak{E}_{jk} \leftarrow \mu_{jk} - \mathrm{sign}(\mu_{jk}) \cdot \min(S\sigma_{jk}, |\mu_{jk}|)$;
12 **end**
  `/* Compute the edge parity and modify the graph */`
13 $E_{\mathrm{S}} \leftarrow \{(j, k, |\mathfrak{E}_{jk}|) : (j, k) \in E', |\mathfrak{E}_{jk}| > 0\}$;
14 $E_{\mathrm{T}} \leftarrow \mathtt{MaximumSpanningTree}(E_{\mathrm{S}})$;
15 **for** $(j, k) \in E_{\mathrm{T}}$ **do**
16 $\mathcal{P} \leftarrow \mathcal{P} \cup \{(j, k, \mathrm{sign}(\mathfrak{E}_{jk}))\}$;
17 $G' = (V', E') \leftarrow \mathtt{GetReducedGraph}(G', (j, k))$
18 **end**
19 **end**
20 $\mathbf{b}^* \leftarrow \mathtt{BruteForceSearch}(G, \mathcal{P})$;

E_{S} made by this collection that is disconnected from the others, the maximum spanning tree E_{T} can be calculated in linear time in the number of edges by Prim's algorithm [19] or by the randomized KKT algorithm [20]. The obtained tree E_{T} can be easily converted into a rooted tree by setting an arbitrary node as the root.

Lines 15–18: Modify the Graph Now, we have candidates E_{T} that can be used to determine the parity and can be deleted from graph G' without inconsistency. The parity for each edge in E_{T} is determined from the leaf to the root one by one according to its sign. At the same time, the node on the leaf side is deleted and the edge weights are updated. The graph modification procedure is the same as the RQAOA, as depicted in Fig. 2.

Here, we note that the proposed method differs from the RQAOA in two parts: on which nodes and on how many nodes to be removed per iteration. In RQAOA, only one node at one end of the edge with the maximum absolute edge energy is removed per iteration, which is essentialy the only choice up to symmetricity of the ZZ interaction. In contrast, in the proposed method, any node that is an endpoint of an edge whose absolute edge energy exceeding the threshold is a candidate for removal per iteration (Fig. 4(a)). To perform this multiple node removal without inconsistency, some edges that form a cycle are removed by calculating a spanning tree. If an edge is not part of the spanning tree, it is ignored, even if its edge energy exceeds the threshold. In the step where spanning trees are formed from the edges exceeding the threshold, a tree is created for each disconnected set of edges (Fig. 4(b)). For each tree, a rooted tree is created by selecting a node at random as the root (Fig. 4(c)), and node removal is performed from the leaves toward the

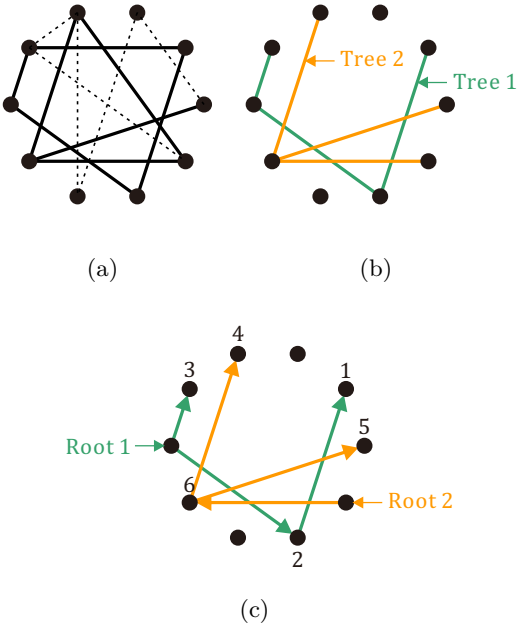


Figure 4: Schematic illustration of tree construction and node deletion ordering: (a) Solid and dashed lines indicate edges with energy above and below the threshold, respectively; (b) Trees formed from the solid lines in (a); (c) Root nodes are randomly selected for each tree, and the nodes to be deleted are ordered from the leaf to the root.

root.

Line 20: Brute-force Search This is essentialy the same as in RQAOA [3]: the recursive procedure is stopped when the number of nodes of G' is less than or equal to some constant M. In such case, the number of nodes whose parities are undetermined is at most M, and therefore the number of candidate solutions is at most 2^M. The best solution $\mathbf{b}^*$ can be quickly obtained by a brute-force search.

Example The whole RQRAO procedure is schematically shown in Fig. 5. The problem graph G is shown in Fig. 5(a), where the red and blue edges have weights $+1$ and -1, respectively. The random Pauli assignments and optimization are carried out to obtain the edge energy for each trial t, $\mathcal{E}_{jk}^{(t)}$ (Fig. 5(b)). The ensemble edge energy $\mathfrak{E}_{jk}$ is then calculated using $\{\mathcal{E}_{jk}^{(t)}\}_{t=1}^{N}$ (Fig. 5(c)). The inconsistent edges are removed by solving the maximum spanning tree problem, where $|\mathfrak{E}_{jk}|$ are regarded as the edge weights. The rooted tree is constructed by giving the direction from root to leaf, where the root node has been randomly selected (Fig. 5(d)). The edge parities in the rooted tree are determined from leaf to root according to the sign of corresponding $\mathfrak{E}_{jk}$. At the same time, the graph is modified to satisfy the determined parity. Figure 5(e) represents the modified graph after all edge parities in the rooted tree have been determined. The protocol from Fig. 5(a) to 5(e) is repeated until the number of nodes of the modified graph becomes at most M.

3.2 Theoretical Analysis

The proposed method, RQRAO, combines the existing RQAOA [3] and QRAO [11]. Although the existing methods work well experimentally, there are interesting theoretical questions: (i) Why does the recursive approach improve the cut weight performance in the RQAOA? (ii) What is the meaning of maximizing the expectation value of the QRAC Hamiltonian, and what is the theoretical background of Pauli rounding in QRAO? To answer these questions, we prove the following theorem:

Theorem 1 *The objective of the RQAOA, QRAO, and RQRAO can be unifiedly expressed as the problem of solving*

$$\rho^* = \operatorname*{argmax}_{\rho} \mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\Big[\mathit{CW}(\mathbf{b})\Big], \tag{17}$$

$$\mathbf{b}^{\mathrm{R}} = \operatorname*{argmax}_{\mathbf{b}\in\{0,1\}^{|V|}} \mathcal{P}_m(\mathbf{b};\rho^*) + \mathcal{P}_m(\overline{\mathbf{b}};\rho^*), \tag{18}$$

where $\overline{\mathbf{b}}$ *is the bit-flip of* $\mathbf{b}$*, and* $\mathcal{P}_m(\mathbf{b};\rho)$ *is the probability of obtaining* $\mathbf{b}$ *by* $(m,1)$*-random magic measurements whose value is proportional to the fidelity between* ρ *and the* $(m,1)$*-magic state embedding* $\mu(\mathbf{b})$*.*

Here, the superscript "R" stands for "rounding" and emphasizes that it represents the classical bits obtained by decoding the information embedded in the qubits. From Theorem 1, it is clarified that both the RQAOA and QRAO, let alone RQRAO, are models searching for the highest probability sample of the probability distribution that maximizes the expectation value of the cut weight through the quantum state as the hidden variable. It will be proved later that maximizing the expectation value of the QRAC Hamiltonian, let alone that of the Ising-type Hamiltonian, can be seen as carrying out

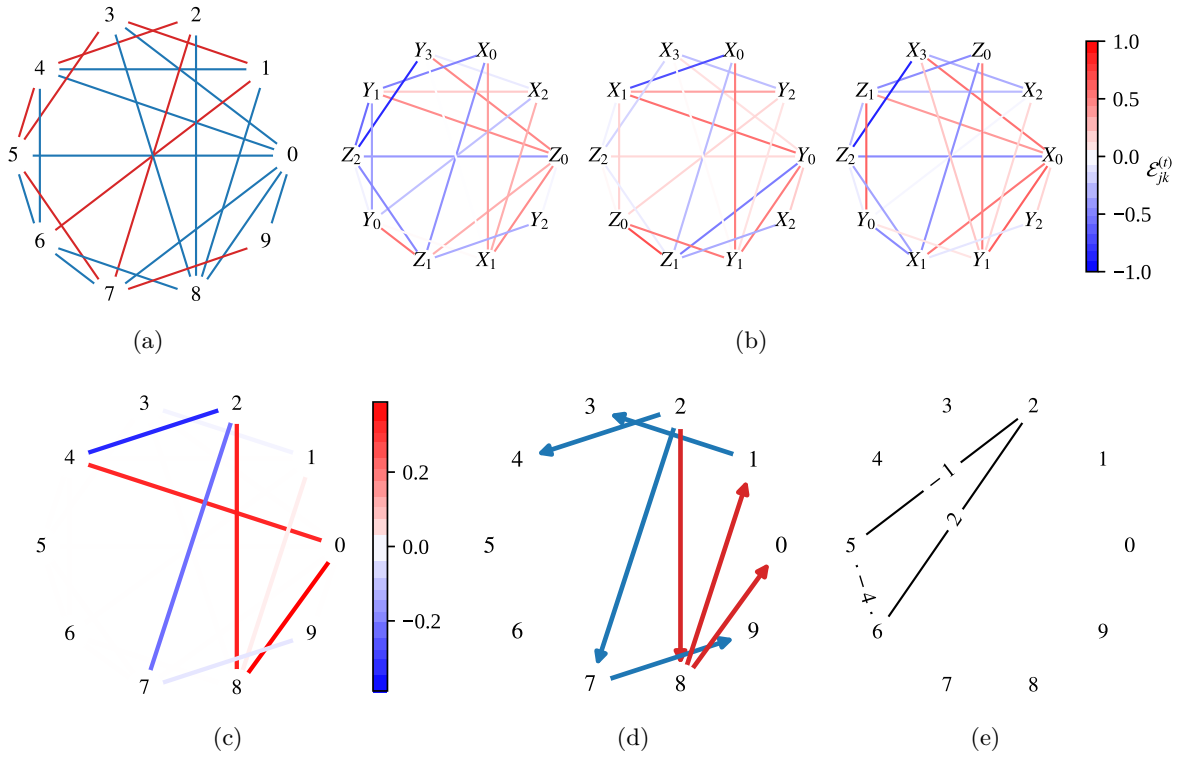


Figure 5: Schematic illustration of the RQRAO procedure. (a) Initial graph whose edge weights are $+1$ (red) or -1 (blue). (b) Three samples of edge energy, $\mathcal{E}_{jk}^{(1)}$, $\mathcal{E}_{jk}^{(2)}$, and $\mathcal{E}_{jk}^{(3)}$, are evaluated using optimized states whose corresponding Hamiltonians are $H_3^{(1)}$, $H_3^{(2)}$, and $H_3^{(3)}$, respectively. Node labels are assigned 1-local Paulis. (c) Ensemble edge energy $\mathfrak{E}_{jk}$ using the three samples illustrated in (b). Closed loop $0-4-2-8-0$ is inconsistent because $(b_0 = b_4) \wedge (b_4 \neq b_2) \wedge (b_2 = b_8) \wedge (b_8 = b_0)$ cannot be satisfied. (d) Maximum spanning tree with $|\mathfrak{E}_{jk}|$ edge weights. Red and blue indicate positive and negative edges, respectively. Edge $(0, 4)$ is ignored at this step. (e) Modified graph after all leaf nodes of (d) have been removed and the edges in the tree have been reconnected to the remaining nodes. RQRAO repeats steps (a) to (e) until the number of nodes becomes smaller than M.

Eq. (17). In addition, it will also be proved later that both the recursive approach in the RQAOA and Pauli rounding in the QRAO are just methods that approximate Eq. (18). One of the contributions of this paper is a discovery that QRAO, RQAOA, and RQRAO, which at first glance seem unrelated in terms of their decoding processes, all decode bit strings according to Eq. (18). This allows us to consider a unified framework that encompasses all methods.

Surely, Theorem 1 does not say the MAX-CUT problem is efficiently solvable because both maximizing the expected cut weight and obtaining the highest probability sample is, in general, intractable when $\mathbf{b}$ lives in an extremely high dimensional space. However, we can say that it is natural to take the $\mathbf{b}$ with the highest probability because, after solving Eq. (17), which maximizes the expected cut weight, the probability for larger $\texttt{CW}(\mathbf{b})$ is expected to be large. It is therefore reasonable to solve Eqs. (17) and (18) to obtain a better cut weight.

In Sec. 3.2.1, Eq. (17) is proved, while Eq. (18) is proved in Sec. 3.2.2. All the proofs of the theorems and corollaries in the following section are deferred to Supplementary Information C.

3.2.1 Proof of Eq. (17)

The objectives of the RQAOA, QRAO, and RQRAO are to maximize $\mathrm{tr}(H_m\rho)$, where $m = 1$ for the RQAOA whereas $m \in \{2, 3\}$ for QRAO and RQRAO. Hence, to prove Eq. (17), we need to show that maximizing $\mathrm{tr}(H_m\rho)$ is equivalent to maximizing $\mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}[\texttt{CW}(\mathbf{b})]$. For this, let us start with the following theorems and corollaries, which are given for the first time in this paper.

Theorem 2 *Let* $P^{\otimes k} := \prod_{i=1}^{k} P_{q^{(i)}} \in \mathbb{C}^{2^n\times 2^n}$ *be the* k*-local Pauli observable, where* P_q *represents*

1-local Pauli $P \in \mathfrak{P} \subset \{X, Y, Z\}$ acting on the qth qubit, and $q^{(i)}$ is the ith qubit index, where $q^{(i)} \neq q^{(j)}$ for $i \neq j$. Suppose the case $|\mathfrak{P}| = m$. Let $\rho \in \mathbb{C}^{2^n \times 2^n}$ be the arbitrary n qubit state, $\mu_m(\mathbf{b}) \in \mathbb{C}^{2^n \times 2^n}$ be the n qubit $(m,1)$-magic state defined in Eq. (10), in which $\mathbf{b} \in \{0,1\}^{mn}$ is embedded, and $f_m(\mathbf{b}, \rho) := \mathrm{tr}(\mu_m(\mathbf{b})\rho)$ be the fidelity between $\mu_m(\mathbf{b})$ and ρ. The following relationship holds:

$$\mathrm{tr}(P^{\otimes k}\rho) = m^{k/2}\, \mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\Big[(-1)^{\sum_{i=1}^{k} b_P^{[q^{(i)}]}}\Big], \tag{19}$$

where $b_P^{[q]} \in \{0,1\}$ is a binary mapped to the Pauli P of the qth qubit and

$$\mathcal{P}_m(\mathbf{b};\rho) := \frac{f_m(\mathbf{b},\rho)}{\sum_{\mathbf{b}'\in\{0,1\}^{mn}} f_m(\mathbf{b}',\rho)}. \tag{20}$$

Note that $\mathbf{b}$ following the probability $\mathcal{P}_m$ is the outcome of the $(m,1)$-random magic measurements, whose details are described in Supplementary Information A.

Corollary 1 *Let $\rho \in \mathbb{C}^{2^n \times 2^n}$ be the arbitrary n qubit state and H_m be the $(m,1)$-QRAC Hamiltonian defined as Eq. (8). The following relationship holds:*

$$\begin{aligned}&\mathrm{tr}(H_m\,\rho)\\ &= \sum_{(j,k)\in E} \frac{1 - m^2\,\mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\big[(-1)^{b_j+b_k}\big]}{2} w_{jk}\\ &= m^2\,\mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\big[\mathtt{CW}(\mathbf{b})\big] - \frac{m^2-1}{2}\sum_{(j,k)\in E} w_{jk},\end{aligned} \tag{21}$$

where $\mathcal{P}_m$ is defined as Eq. (20).

According to Eq. (21), the expectation value of the $(m,1)$-QRAC Hamiltonian is proportional to the expectation value of the cut weight, where the expectation is over the bit strings obtained from $(m,1)$-random magic measurements. Hence, maximizing the expectation value of the $(m,1)$-QRAC Hamiltonian can be regarded as maximizing the expected cut weight. Note that the lower bounds of $\mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\big[\mathtt{CW}(\mathbf{b})\big]$ for $w_{jk} = 1$ when $\mathrm{tr}(H_m\rho) \geq \mathtt{CW}(\mathbf{b}^*)$ have been given in [11, Theorem 2 and Supplementary Information VII]. They are $\mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\big[\mathtt{CW}(\mathbf{b})\big] \geq \frac{5}{9}\mathtt{CW}(\mathbf{b}^*)$ for $m = 3$ and $\mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\big[\mathtt{CW}(\mathbf{b})\big] \geq \frac{5}{8}\mathtt{CW}(\mathbf{b}^*)$ for $m = 2$. These bounds are reproduced by substituting $\mathrm{tr}(H_m\rho) \geq \mathtt{CW}(\mathbf{b}^*)$ and $\sum_{(j,k)\in E} w_{jk} \geq \mathtt{CW}(\mathbf{b}^*)$ into Eq. (21).

Note that not only the expectation value but also the variance of the cut weight can be obtained using the $(m,1)$-QRAC Hamiltonian as follows.

Corollary 2 *Let $\rho \in \mathbb{C}^{2^n \times 2^n}$ be the arbitrary n qubit state and H_m be the $(m,1)$-QRAC Hamiltonian defined as Eq. (8). The variance of the cut weight is*

$$\begin{aligned}&\mathrm{Var}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\big[\mathtt{CW}(\mathbf{b})\big]\\ &= \frac{1}{m^4}\Big(\mathrm{tr}(H_m^2\rho) - \mathrm{tr}(H_m\rho)^2\Big),\end{aligned} \tag{22}$$

where $\mathcal{P}_m$ is defined as Eq. (20).

3.2.2 Proof of Eq. (18)

Because the cut weight is invariant to the bit flip, we can fix one of the b_j to 0 or 1. In this case, there exists a one-to-one relationship between $\mathcal{P}_m(\mathbf{b};\rho^*) + \mathcal{P}_m(\overline{\mathbf{b}};\rho^*)$ and

$$\mathcal{P}_m(\mathfrak{p};\rho^*,T) = \prod_{b=1}^{|V|-1} \mathcal{P}_m(\mathfrak{p}_{e_b}|\mathfrak{p}_{e_{<b}};\rho^*,T), \tag{23}$$

where

$$\mathfrak{p}_{e=(c,p)} := \begin{cases} + & (b_c = b_p)\\ - & (b_c \neq b_p)\end{cases} \tag{24}$$

is a random variable of an edge parity, where "+" and "−" indicate positive and negative parities, respectively, $T = \{e_{j_b} = (c_{j_b}, p_{j_b})\}_{b=1}^{|V|-1}$ is an arbitrary spanning tree of given graph G, and $\mathfrak{p}_{e_{<b}} := \{\mathfrak{p}_{e_1}, \ldots, \mathfrak{p}_{e_{b-1}}\}$. Note that $\mathbf{b}^{\mathrm{R}}$ can be easily recovered using T and $\mathfrak{p}^{\mathrm{R}}(T)$. Here, the reason for considering the probability of the edge parity on the tree is to exclude inconsistent parities such as a triangle with negative parities. Excluding such an inconsistency does not lose the generality because all possible configurations of solutions can be treated by considering only the probability on the tree. Using the edge parity expression, Eq. (18) can be rewritten as

$$\begin{aligned}\mathfrak{p}^{\mathrm{R}}(T) &= \operatorname*{argmax}_{\mathfrak{p}\in\{+,-\}^{|V|-1}} \mathcal{P}_m(\mathfrak{p};\rho^*,T)\\ &= \operatorname*{argmax}_{\mathfrak{p}\in\{+,-\}^{|V|-1}} \prod_{b=1}^{|V|-1} \mathcal{P}_m(\mathfrak{p}_{e_b}|\mathfrak{p}_{e_{<b}};\rho^*,T).\end{aligned} \tag{25}$$

In the following, we show that QRAO follows Eq. (18), whereas both the RQAOA and RQRAO follow Eq. (23) to obtain the solution. All of them

assume a moderate approximation on a probability distribution. The following corollary is frequently used for the proof.

Corollary 3 *Define* $\mathcal{E}_j := \mathrm{tr}(P_{\langle j\rangle}\rho)$ *and* $\mathcal{E}_{jk} := \mathrm{tr}(P_{\langle j\rangle}P_{\langle k\rangle}\rho)$. *When the bit string* $\mathbf{b} = b_{|V|-1}\dots b_1 b_0$ *is obtained by the* $(m,1)$*-random magic measurements,* $\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)$*, the probabilities of* $b_j=0$ *and* $b_j=1$ *are*

$$\begin{aligned}\mathcal{P}(b_j=0) &= \frac{1}{2}+\frac{1}{2\sqrt{m}}\mathcal{E}_j\\ \mathcal{P}(b_j=1) &= \frac{1}{2}-\frac{1}{2\sqrt{m}}\mathcal{E}_j\end{aligned}, \tag{26}$$

respectively, whereas the probability of $b_j=b_k$ *and* $b_j\neq b_k$ *are*

$$\begin{aligned}\mathcal{P}(b_j=b_k) &= \begin{cases}\frac{1}{2}+\frac{1}{2m}\mathcal{E}_j\mathcal{E}_k & (q^{(j)}=q^{(k)})\\ \frac{1}{2}+\frac{1}{2m}\mathcal{E}_{jk} & (q^{(j)}\neq q^{(k)})\end{cases}\\ \mathcal{P}(b_j\neq b_k) &= \begin{cases}\frac{1}{2}-\frac{1}{2m}\mathcal{E}_j\mathcal{E}_k & (q^{(j)}=q^{(k)})\\ \frac{1}{2}-\frac{1}{2m}\mathcal{E}_{jk} & (q^{(j)}\neq q^{(k)})\end{cases}\end{aligned}, \tag{27}$$

respectively, where $q^{(j)}$ *is a qubit with* b_j *embedded.*

Equation (26) is consistent with the upper bound of the decoding probability, $\frac{1}{2}+\frac{1}{2\sqrt{m}}$ [1], which is recovered when $|\mathcal{E}_j|=1$.

Pauli rounding Let us consider two assumptions: bit-flip and independent ones, which we refer to as *assumptions A*, denoted by the superscript A. The first one is the bit-flip assumption

$$\begin{aligned}&\operatorname*{argmax}_{\mathbf{b}}\mathcal{P}_m^{\mathrm{A}}(\mathbf{b};\rho^*)\\ &=\operatorname*{argmax}_{\mathbf{b}}\mathcal{P}_m^{\mathrm{A}}(\mathbf{b};\rho^*)+\mathcal{P}_m^{\mathrm{A}}(\overline{\mathbf{b}};\rho^*).\end{aligned} \tag{28}$$

In many cases, this assumption is practically satisfied for MAX-CUT . This is because the ground states of the MAX-CUT Hamiltonians are degenerate; $\mathbf{b}$ and $\overline{\mathbf{b}}$ represent the same cut. Moreover, unless a specific constraint is intentionally imposed to balance $\mathcal{P}_m(\mathbf{b};\rho)$ and $\mathcal{P}_m(\overline{\mathbf{b}};\rho)$, optimization can result in one of $\mathcal{P}_m(\mathbf{b};\rho)$ or $\mathcal{P}_m(\overline{\mathbf{b}};\rho)$ becoming large while the other becomes close to zero. Alternatively, in the case of an ansatz with $\mathbb{Z}_2$ symmetry [3], such as QAOA, the bit-flipped solution appears with the same probability, so the assumption is also satisfied in this case.

The second one is the independent assumption that all b_{j_a} for $a=1,2,\dots,|V|$ are independent where $\{j_a\}_{a=1}^{|V|}=\{1,\dots,|V|\}$, which is equivalent to write the joint distribution as

$$\mathcal{P}_m^{\mathrm{A}}(\mathbf{b};\rho^*)=\prod_{a=1}^{|V|}\mathcal{P}_m^{\mathrm{A}}(b_{j_a};\rho^*). \tag{29}$$

Under these assumptions, Eq. (18) is approximated as

$$\begin{aligned}b_{j_a}^{\mathrm{A}} &= \operatorname*{argmax}_{b_{j_a}\in\{0,1\}}\mathcal{P}_m^{\mathrm{A}}(b_{j_a};\rho^*)\\ &= \operatorname*{argmax}_{b_{j_a}\in\{0,1\}}\left(\frac{1}{2}+\frac{(-1)^{b_{j_a}}}{2\sqrt{m}}\mathcal{E}_{j_a}\right)\\ &= \begin{cases}0 & (\mathcal{E}_{j_a}>0)\\ 1 & (\mathcal{E}_{j_a}<0)\end{cases},\end{aligned} \tag{30}$$

where the second equality uses Corollary 3. Interestingly, $\mathbf{b}^{\mathrm{A}}$ is equivalent to the Pauli rounding of QRAO [11] defined in Eq. (12). This comes out that the Pauli rounding seeks the variable with the highest probability of $\mathcal{P}_m(\mathbf{b};\rho)$ under the assumptions of Eqs. (28) and (29).

Tree rounding Assume all $\mathfrak{p}_{e_b}$ for $b=1,2,\dots,|V|-1$ are independent, which we refer to as *assumption B*, denoted by the superscript B. Under this assumption, Eq. (23) is approximated as

$$\mathcal{P}_m^{\mathrm{B}}(\mathfrak{p};\rho^*,T)=\prod_{b=1}^{|V|-1}\mathcal{P}_m^{\mathrm{B}}(\mathfrak{p}_{e_b};\rho^*,T), \tag{31}$$

and Eq. (25) is approximated as

$$\begin{aligned}\mathfrak{p}_{e_b=(c,p)}^{\mathrm{B}}(T) &= \operatorname*{argmax}_{\mathfrak{p}_{e_b}\in\{+,-\}}\mathcal{P}_m^{\mathrm{B}}(\mathfrak{p}_{e_b};\rho^*,T)\\ &= \operatorname*{argmax}_{(b_c,b_p)\in\{0,1\}^2}\left(\frac{1}{2}+\frac{(-1)^{b_c+b_p}}{2m}\mathcal{E}_{cp}\right)\\ &= \begin{cases}+ & (\mathcal{E}_{cp}>0)\\ - & (\mathcal{E}_{cp}<0)\end{cases},\end{aligned} \tag{32}$$

where the second equality uses Corollary 3. We refer to this rounding method as *tree rounding*.

RQRAO with $N = 1$, which has no recursive steps, is equivalent to using tree rounding. When $N = 1$, the ensemble edge energy $\mathfrak{E}_{jk}$ in Eq. (16) is equal to the edge energy estimate $\mathcal{E}_{jk}$ in Eq. (15) from a single evaluation, and almost surely it takes a non-zero value. Since the spanning tree is constructed by gathering edges with $\mathfrak{E}_{jk} \neq 0$, a spanning tree covering all nodes can be formed with high probability. By progressively removing nodes from the leaves to the root of this spanning tree as illustrated in Fig. 4, all nodes except the root are eliminated, determining all parities at once. This is the same rounding method referred to as the tree rounding defined above. Since both Pauli rounding and tree rounding assume that all random variables are independent, it is expected that QRAO and RQRAO with $N = 1$ will yield similar cut weights.

Recursive rounding Let us impose the following two assumptions, which we refer to as *assumptions C*, denoted by the superscript C:

$$\begin{aligned}&\max_{\mathfrak{p}\in\{+,-\}^{|V|-1}} \prod_{b=1}^{|V|-1} \mathcal{P}_m^{\mathrm{C}}(\mathfrak{p}_{e_b}|\mathfrak{p}_{<e_b};\rho^*,T)\\ &= \prod_{b=1}^{|V|-1} \max_{\mathfrak{p}_{e_b}\in\{+,-\}} \mathcal{P}_m^{\mathrm{C}}(\mathfrak{p}_{e_b}|\mathfrak{p}_{<e_b};\rho^*,T), \qquad (33)\end{aligned}$$

$$\mathcal{P}_m^{\mathrm{C}}(\mathfrak{p}_{e_b}|\mathfrak{p}_{<e_b};\rho^*,T) = \mathcal{P}_m^{\mathrm{C}}(\mathfrak{p}_{e_b};\rho_b^*(\mathfrak{p}_{<e_b}),T), \qquad (34)$$

where $\rho_b^*(\mathfrak{p}_{<e_b})$ is the optimized quantum state that involves the information of $\mathfrak{p}_{<e_b}$. Under these assumptions, Eq. (25) can be approximated as

$$\begin{aligned}\mathfrak{p}_{e_b}^{\mathrm{C}}(T) &= \operatorname*{argmax}_{\mathfrak{p}_{e_b}} \mathcal{P}_m^{\mathrm{C}}(\mathfrak{p}_{e_b};\rho_b^*(\mathfrak{p}_{<e_b}))\\ &\text{for} \quad b = 1,\ldots,|V|-1. \qquad (35)\end{aligned}$$

Compared to the Pauli and tree roundings, which impose the strong approximation on $\mathcal{P}_m$ that all random variables are independent, assumptions in Eqs. (34) and (35) are weak because each random variable is no longer independent. Although the assumption of independence in Pauli rounding and tree rounding was made due to the difficulty of conditional sampling, this approach addresses the problem by imposing conditions $\mathfrak{p}_{<e_b} = \mathfrak{p}^*_{<e_b}$ on the quantum state itself instead of on $\mathcal{P}_m$, where $\mathfrak{p}^*_{<e_b}$ is the determined edge parities before step b. Of course, this approach is not exactly equivalent to the original one, but imposing the condition as a constraint in the optimization should provide a good approximation to conditional sampling. In the following, we specifically demonstrate how the method of recursively decoding bit strings, as adopted in Recursive QAOA [3], follows Eq. (25) to obtain $\mathfrak{p}^{\mathrm{R}}(T)$ under the assumptions in Eqs. (34) and (35).

First, one parity of e_1 is determined using the optimization result, ρ_1^* as

$$\mathfrak{p}^*_{e_1} = \operatorname*{argmax}_{\mathfrak{p}_{e_1}\in\{+,-\}} \mathcal{P}_m(\mathfrak{p}_{e_1};\rho_1^*), \qquad (36)$$

where

$$e_1 = \operatorname*{argmax}_{e} \max_{\mathfrak{p}_e\in\{+,-\}} \mathcal{P}_m(\mathfrak{p}_e;\rho_1^*). \qquad (37)$$

According to Corollary 3, this can be rewritten as

$$\mathfrak{p}^{\mathrm{rec}}_{e_1=(c_1,p_1)} = \begin{cases} + & (\mathcal{E}_{c_1p_1} > 0)\\ - & (\mathcal{E}_{c_1p_1} < 0)\end{cases}, \qquad (38)$$

where

$$\begin{aligned}(c_1,p_1) &= \operatorname*{argmax}_{(c,p)\in E}\left(\frac{1}{2}+\frac{1}{2m}|\mathcal{E}_{cp}|\right)\\ &= \operatorname*{argmax}_{(c,p)\in E}|\mathcal{E}_{cp}|.\end{aligned} \qquad (39)$$

After that, the given graph G is modified to satisfy $\mathfrak{p}_{e_1} = \mathfrak{p}^*_{e_1}$, which refers to G_2. Next, the second optimization is carried out with the $(m,1)$-QRAC Hamiltonian based on G_2, and the optimized state ρ_2^* is obtained. Because the modified graph is forced to be $\mathfrak{p}_{e_1} = \mathfrak{p}^*_{e_1}$, it can be seen that ρ_2^* is conditioned on $\mathfrak{p}_{e_1} = \mathfrak{p}^*_{e_1}$. Hence, the determined parity of the second step can be written as

$$\mathfrak{p}^*_{e_2} = \operatorname*{argmax}_{\mathfrak{p}_{e_2}\in\{+,-\}} \mathcal{P}_m(\mathfrak{p}_{e_2};\rho_2^*(\mathfrak{p}_{e_1} = \mathfrak{p}^*_{e_1})) \qquad (40)$$

where

$$e_2 = \operatorname*{argmax}_{e} \max_{\mathfrak{p}_e\in\{+,-\}} \mathcal{P}_m(\mathfrak{p}_e;\rho_2^*(\mathfrak{p}_{e_1} = \mathfrak{p}^*_{e_1})). \qquad (41)$$

Of course, this could also be rewritten using the edge energy $\mathcal{E}_{cp}$, but we will leave that out as it is self-explanatory. Repeating this procedure gives

$$\mathfrak{p}^*_{e_b} = \operatorname*{argmax}_{\mathfrak{p}_{e_b}} \mathcal{P}_m(\mathfrak{p}_{e_b};\rho_b^*(\mathfrak{p}_{<e_b} = \mathfrak{p}^*_{<e_b}))$$

$$\text{for} \quad b = 1, \ldots, |V| - 1. \tag{42}$$

Finally, $\mathfrak{p}^* = \{\mathfrak{p}_1^*, \ldots, \mathfrak{p}_{|V|-1}^*\}$ is obtained. The corresponding bit string $\mathbf{b}^{\mathrm{C}}$ can be easily recovered from $\mathfrak{p}^*$ by fixing one of the b_j^{C} to 0 or 1. The condition $\mathfrak{p}_{<e_b} = \mathfrak{p}^*_{<e_b}$ is indirectly imposed through ρ_b^* to $\mathcal{P}_m$, and hence this rounding method imposes an approximation that is weaker than that of Pauli and tree rounding. We refer to this rounding method as *recursive rounding*.

In recursive rounding, the edge that determines the parity is selected using the absolute value of its edge energy (Eq. (41)), and its parity is determined by the sign of the corresponding edge energy (Eq. (40)). This process is the same as explained in Sec. 2.2 and hence, it can be said that the RQAOA adopts recursive rounding. The main idea behind RQAOA is to approximate $\max_{\mathbf{b}\in\{0,1\}^{|V|}} \mathrm{tr}(H_1|\mathbf{b}\rangle\langle\mathbf{b}|)$ [3], but in actuality, it approximates $\max_{\mathbf{b}\in\{0,1\}^{|V|}} \mathcal{P}_1(\mathbf{b};\rho^*) = \max_{\mathbf{b}\in\{0,1\}^{|V|}} \mathrm{tr}(\rho^*|\mathbf{b}\rangle\langle\mathbf{b}|)$.[1] By contrast, the rounding for RQRAO with $N > 1$ can be considered to be an intermediate approach between tree rounding and recursive rounding because the edge parities that are being determined simultaneously in one step are assumed to be independent.

4 Numerical Experiments

In this section, we show the results of the numerical experiments of the MAX-CUT problem using RQRAO and other existing algorithms. The main claim is that RQRAO achieves a better cut weight than existing algorithms and is comparable to the state-of-the-art classical heuristic.

Datasets We used three graph datasets.

The first dataset is used to show the validity of Theorem 1. That is, it shows that the bit string with the highest probability yields a good cut weight after maximizing the expectation of the $(m, 1)$-QRAC Hamiltonian. Only one graph instance is generated by a machine-independent graph generator `rudy` [21] using `-rnd_graph 14 50 0 -random 0 1 0 -times 2 -plus -1`, where the number of nodes is 14, the graph density is 0.5, and the edge weights are set to ± 1 uniformly at random. This instance has only one MAX-CUT solution, $\mathbf{b} = 00001101101010$, except for its bit-flip solution, where the maximum cut weight is 12. We call this graph instance `Rnd14`.

The second dataset is the benchmark graph dataset Gset [15] (`http://www.stanford.edu/~yyye/yyye/Gset/`). Gset includes random, toroidal, and almost planar graphs with $|V| = 800$ to 20000 indexed by G1 to G81, which are generated by `rudy`. Only G1, G6, G11, G14, and G18 are used for evaluation, all of which have 800 nodes. G1 and G6 are random graphs, G11 is a toroidal graph, and G14 and G18 are the combined graphs of two planar graphs. Graphs G1 and G6, and graphs G14 and G18 have the same edge structure except for the weights, where G1 and G14 have $+1$ edge weights whereas G6 and G18 have edge weights of ± 1 assigned at random. The edge weights of G11 are ± 1 assigned at random. We call this graph dataset `Gset800`.

The third dataset consists of 3-regular graphs with ± 1 edge weights generated at random. The number of nodes varies from 10 to 10^4. Ten graphs are generated for each number of nodes. These graphs are used to evaluate the scalability with respect to the cut weight and the runtime of the algorithms. We call this graph dataset `Rnd3R`.

Baselines We compared our model with GW [7], CirCut [16], QRAO [11], and RQAOA [3]. GW is a well-known classical approximation algorithm based on semidefinite programming. CirCut is the Fortran program name of the rank-two relaxation algorithm; it is a classical heuristic algorithm that empirically outperforms other classical heuristics on various graph instances [22]. The details of GW and CirCut are described in Supplementary Information H.1. The RQAOA is a QAOA-based algorithm whose details are explained in Sec. 2.2. QRAO is a quantum relaxation-based optimization method whose details are explained in Sec. 2.3.

Implementation Details The matrix product state (MPS) ansatz is used for $|\psi(\boldsymbol{\theta})\rangle$, where all the matrices in the MPS are treated as trainable parameters (see Supplementary Information D for details). The default hyperparameters for RQRAO used in the numerical experiments are listed in Table 1.

[1] Since $\mu_1(\mathbf{b}) = |\mathbf{b}\rangle\langle\mathbf{b}|$, $f_1(\mathbf{b},\rho) = \mathrm{tr}(\mu_1(\mathbf{b})\rho)$, and $\sum_{\mathbf{b}\in\{0,1\}^{|V|}} f_1(\mathbf{b},\rho) = 1$, $\mathcal{P}_1(\mathbf{b};\rho) = f_1(\mathbf{b},\rho)/\sum_{\mathbf{b}'} f_1(\mathbf{b}',\rho) = \mathrm{tr}(|\mathbf{b}\rangle\langle\mathbf{b}|\rho)$.

Table 1: Default hyperparameters for RQRAO.

Hyperparameter	Symbol	Value
# embeddings / qubits	m	3
# ensemble	N	20
Scale factor	S	2
Bond dimension	χ	2
Brute-force search threshold	M	10

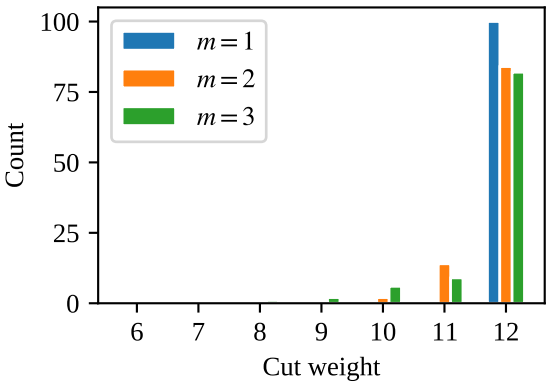
We implemented our model in Python, and the MPS optimization is implemented using PyTorch version 1.10.0 [23]. For the optimizer, we used L-BFGS [24] with line search using strong Wolfe conditions. The termination tolerance for changing the function value and parameters is set to 10^{-2}. Implementation details for GW, CirCut, the RQAOA, and QRAO are described in Supplementary Information E.

All experiments were carried out on a single Intel(R) Core(TM) i9-9900X CPU @ 3.50GHz with 32GB (8GBx4) DDR4-2666 Quad-Channel.

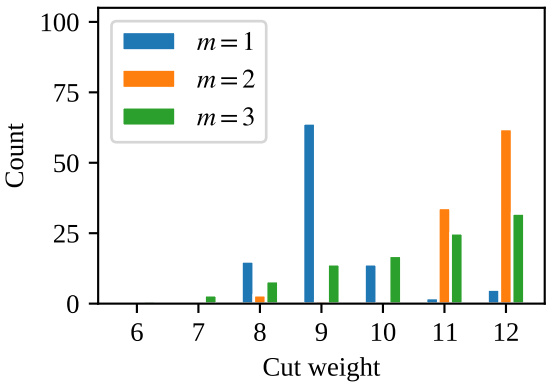
Details of the Experiments Using the `Rnd14` dataset, Theorem 1 was verified. First, $(m, 1)$-QRAC Hamiltonians were randomly generated 100 times each for $m = 1, 2$, and 3. For each Hamiltonian, the eigenstate with the maximum eigenvalue and optimized state with bond dimension $\chi = 2$ were then calculated. Using each state, the cut weight with the highest probability was calculated. We call this experiment `ExpVer`.

Using the `Gset800` dataset, the procedure was run 10 times on the same graph instance and the best cut weight of the trials was recorded. RQRAO and all baseline methods were used. We call this experiment `ExpGset`.

Using the `Rnd3R` dataset, the procedure was run once for each graph instance and the cut weight and elapsed time were recorded. The cut weight is normalized with respect to that of the GW algorithm and is referred to as the relative cut weight. Using `Rnd3R`, two experiments were carried out. The first one evaluated the scalability. RQRAO and all baseline methods were used. We call this experiment `ExpScale`. The second one analyzed the sensitivity of the RQRAO hyperparameters. Only one hyperparameter of RQRAO was varied from its default value (Table 1). We call this experiment `ExpHyp`.



(a) MaxEig



(b) $\chi = 2$

Figure 6: Solutions with the highest probability of 100 trials for the same graph instance, `Rnd14`, where the state used was (a) the eigenstate corresponding to the maximum eigenvalue and (b) the state obtained through optimization with $\chi = 2$.

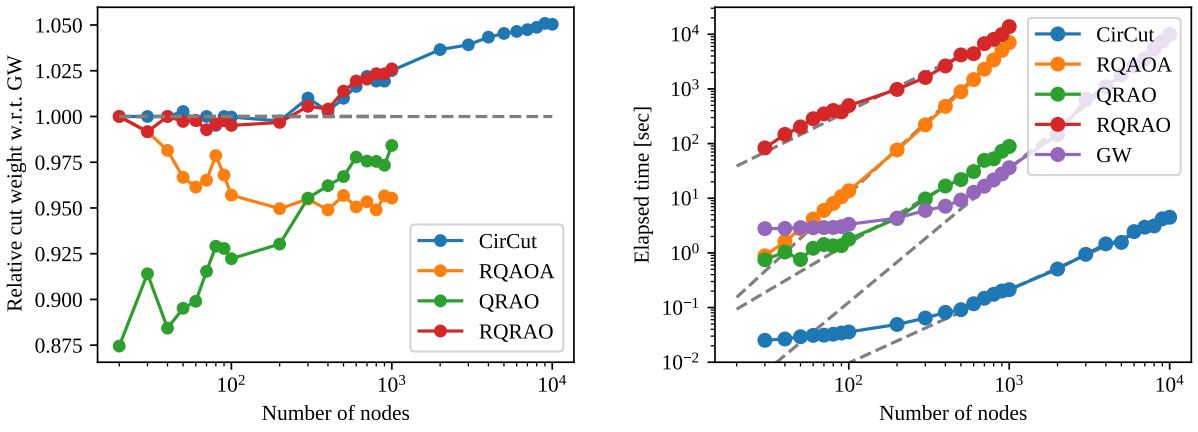
4.1 Results and Discussion

Verification of Theorem 1 (`ExpVer`) Figure 6 summarizes the results of `ExpVer`. When the eigenstate with the maximum eigenvalue is obtained, the $(1, 1)$-QRAC is consistently the best choice as it provides the optimal MAX-CUT solution with the highest probability (Fig. 6(a)). On the other hand, for $(2, 1)$ and $(3, 1)$-QRACs, particularly when higher cut weight solutions cluster in a different location within the Hilbert space from the MAX-CUT solution, the solution with the highest probability may not correspond to the optimal MAX-CUT solution. Therefore, even if the ground state is obtained, there is a disadvantage that the nearest solution may not be the optimal MAX-CUT solution.

Nevertheless, when the quantum state is obtained through optimization (Fig. 6(b)), $(2, 1)$

Table 2: Comparison with existing methods on MAX-CUT problems from the Gset. All instances have $|V| = 800$. All best-known values were reported in [25] with the breakout local search algorithm. GW: Goemans–Williamson algorithm [7], RQAOA: recursive quantum approximate optimization algorithm [3], QRAO: quantum random access optimization [11], RQRAO: recursive quantum random access optimization (proposed).

Instance	Best-known	GW	CirCut	QRAO	RQAOA	RQRAO$_{N=1}$	RQRAO$_{N=20}$
G1	11624	11467	11622	11453	11460	11466	11562
G6	2178	2013	2178	1980	1821	1980	2148
G11	564	536	556	540	560	544	564
G14	3064	2999	3052	2971	2965	2965	3043
G18	992	924	987	929	882	920	980



(a) (b)

Figure 7: Comparison of scalability with existing methods. Ten 3-regular graphs with the same number of nodes were randomly generated whose edge weights were ± 1 at random. The markers indicate the means. (a) Relative cut weight with respect to the cut weight obtained by the GW algorithm; (b) Algorithm run time. The gray dashed line is a fitted curve using $(\text{Elapsed time [sec]}) = a(\text{Number of nodes})^b$

.

and $(3, 1)$-QRACs achieve higher cut weights than when using $(1, 1)$-QRAC. The reason why higher cut weights can be obtained by setting $m \geq 2$ can be explained as follows. As shown in Fig. 1, as m increases, more solutions are packed into fewer qubits, resulting in an average over a greater number of solutions being observed. In other words, as m increases, we can only observe the average of a larger number of solutions, leading to a *blurred* expectation value. Since the optimization process attempts to reach the optimal state from such a blurred function value that only reflects the average, increasing m does not necessarily bring the optimized state closer to the nearest state embedding the optimal solution. However, this blurring could also potentially make the function value smoother, thus simplifying the optimization process. Such optimization-related advantages occur for $m = 2$ and $m = 3$, and it is likely that $m = 2$ resulted in the highest cut weight because it strikes a balance between the aforementioned advantages and disadvantages.

In this study, we have optimized a classically trackable MPS ansatz using classical optimization methods. However, recent developments, such as quantum state preparation using Lindbladians [26], suggest that *pure* quantum algorithms are promising. In our method, more than 90% of the computation time is spent on parameter optimization, so there could be benefits in terms of overall algorithm speedup by replacing the optimization process with a pure quantum algorithm. However, since the discussion on optimization is beyond the scope of this study, these considerations will be addressed in future work. In any case, the MAX-CUT problem, traditionally a problem of finding the eigenstate with the maximum eigenvalue, has effectively been shifted

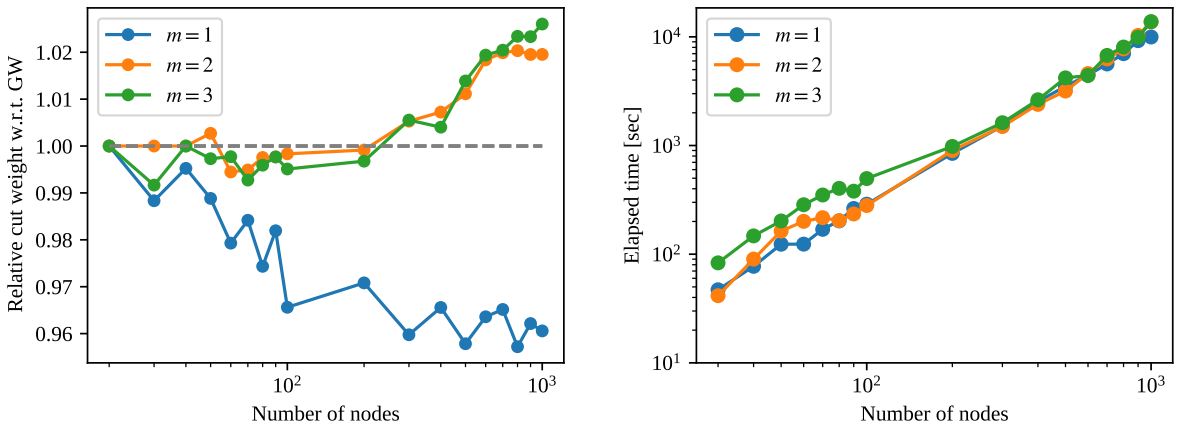


Figure 8: Results of the ablation study for the number of embeddings per qubit m. The problem graph instances are 3-regular graphs with ± 1 edge weights. For each number of nodes, 10 graph instances were randomly generated.

to finding the magic state embedding with the highest fidelity to the optimized quantum state. While challenging, this approach may yield better solutions, particularly with $(2, 1)$ and $(3, 1)$-QRACs. In the following, we will explore the advantages of $m \geq 2$ through additional experiments.

Gset (`ExpGset`) The cut weight on the Gset dataset for each algorithm is summarized in Table 2. RQRAO with $N = 20$ exceeds both the classical approximation algorithm (GW) and quantum algorithms (QRAO and the RQAOA), but is slightly worse than the classical heuristic algorithm (CirCut) on all graphs except for G11.

As expected in Sec. 3.2.2, the cut weight of RQRAO with $N = 1$ is very similar to that of QRAO because both assume all random variables are independent. The cut weight performance of RQRAO increases when the number of ensembles N is increased, which confirms the effectiveness of using an ensemble.

Scalability (`ExpScale`) Figure 7 summarizes the results on 3-regular graphs. Although the relative cut weights of QRAO and the RQAOA are lower than 1, that of RQRAO exceeds 1 when the number of nodes is larger than 200 and is larger than that of CirCut (Fig. 7(a)). In addition, although the absolute runtime of RQRAO is longer than that of CirCut, the order of empirical time complexity of these methods is close (Fig. 7(b)). To quantitatively compare the empirical time complexity of the methods, the runtime curves were fitted using

$$(\text{Wall time [sec]}) = \alpha(\text{Number of nodes})^{\beta}$$

using only data with a large number of nodes. The fitted values of β are 1.34 for CirCut, 2.71 for the RQAOA, 1.73 for QRAO, 1.44 for RQRAO, and 2.44 for GW. That is, the empirical time complexity of the proposed method is much lower than that of GW and the RQAOA, and comparable to that of CirCut. Because most of the computational time is spent on optimizing the variational states, the computational time of the proposed method can be reduced by improving the optimization algorithm. For example, the runtime cost of RQRAO can be reduced by parallel computation in the ensemble part. Further reductions in the computational costs of RQRAO are future work. In Supplementary Information G, we also compared the methods using the 2D toric graphs with one extra node connected to all other nodes, as used in [3] to demonstrate the power of RQAOA. The results reaffirm what we have observed when comparing the methods using 3-regular graphs.

Sensitivity Analysis (`ExpHyp`) Only the results for varying the number of embeddings per qubit m are described here. Other hyperparameter sensitivities such as the number of ensemble N, the scale factor S, and the bond dimension χ, are summarized in Supplemental Information F. In Fig. 8(a), we can see the clear advantage of

using $m > 1$ over $m = 1$; only the cut weight in the cases of $m > 1$ overcomes that of GW. When $m = 1$, the optimized state may be stacked in bad local optima, as shown in Fig. 6(b), and this is the reason for the lower cut weight. Almost no runtime dependency on m is seen in Fig. 8(b), which means there is no disadvantage to use $m > 1$ as long as the given graph is sufficiently sparse to allow applying $(m, 1)$-QRAC Hamiltonian.

5 Conclusion

In this study, we proposed a recursive and quantum-relaxed algorithm to solve the MAX-CUT problem. The quantum relaxation is achieved using the QRAC, which is incorporated into the existing recursive MAX-CUT algorithm. To show the validity of the proposed algorithm, new properties of the quantum-relaxed Hamiltonian were revealed. According to these findings, we showed that the MAX-CUT problem can be approximately reduced to the problem finding the variable with the highest probability, which is determined by the quantum state optimized through the quantum-relaxed Hamiltonian. We confirmed that the proposed algorithm is consistent with this objective and includes the existing methods as special cases. The numerical experiments on graphs with several hundred nodes showed that the proposed algorithm outperforms the GW algorithm, which is the best-known classical approximation algorithm, on graphs with more than 300 nodes both in the approximation ratio and the empirical time complexity.

Acknowledgement

This work is supported by MEXT Quantum Leap Flagship Program Grant Number JPMXS0118067285 and JPMXS0120319794, and JSPS KAKENHI Grant Number 20H05966. We would like to thank Yohichi Suzuki and Michihiko Sugawara of Keio Quantum Computing Center and Hiroshi C. Watanabe of Kyushu University for discussing the theoretical and experimental justification of RQRAO. RR would like to acknowledge Prof. Hiroshi Imai for information on Gset instances and MAX-CUT .

References

[1] Ambainis, A., Nayak, A., Ta-Shma, A. and Vazirani, U. Dense quantum coding and a lower bound for 1-way quantum automata. In Proceedings of the thirty-first annual ACM symposium on Theory of computing, pages 376–383, 1999. DOI: 10.1145/301250.301347.

[2] Ambainis, A., Nayak, A., Ta-Shma, A. and Vazirani, U. Dense quantum coding and quantum finite automata. Journal of the ACM (JACM), 49(4):496–511, 2002. DOI: 10.1145/581771.581773.

[3] Bravyi, S., Kliesch, A., Koenig, R. and Tang, E. Obstacles to variational quantum optimization from symmetry protection. Physical Review Letters, 125 (26):260505, 2020. DOI: 10.1103/PhysRevLett.125.260505.

[4] Karp, R.M. Reducibility among combinatorial problems. Springer, 2010. DOI: 10.1007/978-1-4684-2001-2˙9.

[5] Lucas, A. Ising formulations of many NP problems. Frontiers in physics, 2:5, 2014. DOI: 10.3389/fphy.2014.00005.

[6] Glover, F., Kochenberger, G., Hennig, R. and Du, Y. Quantum bridge analytics I: a tutorial on formulating and using QUBO models. Annals of Operations Research, 314 (1):141–183, 2022. DOI: 10.1007/s10479-022-04634-2.

[7] Goemans, M.X. and Williamson, D.P. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. Journal of the ACM (JACM), 42(6):1115–1145, 1995. DOI: 10.1145/227683.227684.

[8] Farhi, E., Goldstone, J. and Gutmann, S. A quantum approximate optimization algorithm. arXiv preprint arXiv:1411.4028, 2014. DOI: 10.48550/arXiv.1411.4028.

[9] Wang, Z., Hadfield, S., Jiang, Z. and Rieffel, E.G. Quantum approximate optimization algorithm for MaxCut: A fermionic view. Physical Review A, 97(2):022304, 2018. DOI: 10.1103/PhysRevA.97.022304.

[10] Blekos, K. et al. A review on quantum approximate optimization algorithm and its variants. Physics Reports, 1068:1–66, 2024. DOI: 10.1016/j.physrep.2024.03.002.

[11] Fuller, B. et al. Approximate solutions of combinatorial problems via quantum relaxations. IEEE Transactions on Quantum Engineering, 2024. DOI: 10.1109/TQE.2024.3421294.

[12] Kempe, J., Kitaev, A. and Regev, O. The complexity of the local Hamiltonian problem. Siam journal on computing, 35(5): 1070–1097, 2006. DOI: 10.1007/978-3-540-30538-5˙31.

[13] Khot, S. On the power of unique 2-prover 1-round games. In Proceedings of the thiry-fourth annual ACM symposium on Theory of computing, pages 767–775, 2002. DOI: 10.1145/509907.510017.

[14] Bravyi, S., Kliesch, A., Koenig, R. and Tang, E. Hybrid quantum-classical algorithms for approximate graph coloring. Quantum, 6:678, 2022. DOI: 10.22331/q-2022-03-30-678.

[15] Helmberg, C. and Rendl, F. A spectral bundle method for semidefinite programming. SIAM Journal on Optimization, 10(3):673–696, 2000. DOI: 10.1137/S1052623497328987.

[16] Burer, S., Monteiro, R.D. and Zhang, Y. Rank-two relaxation heuristics for max-cut and other binary quadratic programs. SIAM Journal on Optimization, 12(2):503–521, 2002. DOI: 10.1137/S1052623400382467.

[17] Huang, H.Y., Kueng, R. and Preskill, J. Predicting many properties of a quantum system from very few measurements. Nature Physics, 16(10):1050–1057, 2020. DOI: 10.1038/s41567-020-0932-7.

[18] Breiman, L. Bagging predictors. Machine learning, 24:123–140, 1996. DOI: 10.1007/BF00058655.

[19] Prim, R.C. Shortest Connection Networks And Some Generalizations. Bell System Technical Journal, 36(6):1389–1401, November 1957. DOI: 10.1002/j.1538-7305.1957.tb01515.x.

[20] Karger, D.R., Klein, P.N. and Tarjan, R.E. A randomized linear-time algorithm to find minimum spanning trees. Journal of the ACM (JACM), 42(2):321–328, 1995. DOI: 10.1145/201019.201022.

[21] Rinaldi, G. Rudy, 1998. https://www-user.tu-chemnitz.de/~helmberg/rudy.tar.gz.

[22] Dunning, I., Gupta, S. and Silberholz, J. What works best when? A systematic evaluation of heuristics for Max-Cut and QUBO. INFORMS Journal on Computing, 30(3):608–624, 2018. DOI: 10.1287/ijoc.2017.0798.

[23] Paszke, A. et al. PyTorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems, 32, 2019. DOI: 10.48550/arXiv.1912.01703.

[24] Liu, D.C. and Nocedal, J. On the limited memory BFGS method for large scale optimization. Mathematical programming, 45(1-3):503–528, 1989. DOI: 10.1007/BF01589116.

[25] Benlic, U. and Hao, J.K. Breakout local search for the max-cutproblem. Engineering Applications of Artificial Intelligence, 26(3):1162–1173, 2013. DOI: 10.1016/j.engappai.2012.09.001.

[26] Ding, Z., Chen, C.F. and Lin, L. Single-ancilla ground state preparation via lindbladians. Physical Review Research, 6(3): 033147, 2024. DOI: 10.1103/PhysRevResearch.6.033147.

[27] Schollwöck, U. The density-matrix renormalization group in the age of matrix product states. Annals of physics, 326(1):96–192, 2011. DOI: 10.1016/j.aop.2010.09.012.

[28] Toh, K.C., Todd, M.J. and Tütüncü, R.H. SDPT3—a MATLAB software package for semidefinite programming, version 1.3. Optimization methods and software, 11(1-4):545–581, 1999. DOI: 10.1080/10556789908805762.

[29] Tütüncü, R.H., Toh, K.C. and Todd, M.J. Solving semidefinite-quadratic-linear programs using SDPT3. Mathematical programming, 95:189–217, 2003. DOI: 10.1007/s10107-002-0347-5.

[30] Majumdar, A., Hall, G. and Ahmadi, A.A. Recent scalability improvements for semidefinite programming with applications in machine learning, control, and robotics. Annual Review of Control, Robotics, and Autonomous Systems, 3:331–360, 2020. DOI: 10.1146/annurev-control-091819-074326.

[31] Jiang, H., Kathuria, T., Lee, Y.T., Padmanabhan, S. and Song, Z. A faster in-

terior point method for semidefinite programming. In 2020 IEEE 61st annual symposium on foundations of computer science (FOCS), pages 910–918. IEEE, 2020. DOI: 10.1109/FOCS46700.2020.00089.

[32] Huang, B., Jiang, S., Song, Z., Tao, R. and Zhang, R. Solving SDP faster: A robust IPM framework and efficient implementation. In 2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS), pages 233–244. IEEE, 2022. DOI: 10.1109/FOCS54457.2022.00029.

[33] Lee, Y.T. and Padmanabhan, S. An $\widetilde{\mathcal{O}}(m/\varepsilon^{3.5})$-cost algorithm for semidefinite programs with diagonal constraints. In Abernethy, J. and Agarwal, S., editors, Proceedings of Thirty Third Conference on Learning Theory, volume 125 of Proceedings of Machine Learning Research, pages 3069–3119. PMLR, 09–12 Jul 2020. URL https://proceedings.mlr.press/v125/lee20c.html.

[34] Brandao, F.G. and Svore, K.M. Quantum speed-ups for solving semidefinite programs. In 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS), pages 415–426. IEEE, 2017. DOI: 10.1109/FOCS.2017.45.

[35] Van Apeldoorn, J., Gilyén, A., Gribling, S. and de Wolf, R. Quantum SDP-solvers: Better upper and lower bounds. Quantum, 4: 230, 2020. DOI: 10.22331/q-2020-02-14-230.

[36] Brandão, F.G.S.L., Kalev, A., Li, T., Lin, C.Y.Y., Svore, K.M. and Wu, X. Quantum SDP Solvers: Large Speed-Ups, Optimality, and Applications to Quantum Learning. In Baier, C., Chatzigiannakis, I., Flocchini, P. and Leonardi, S., editors, 46th International Colloquium on Automata, Languages, and Programming (ICALP 2019), volume 132 of Leibniz International Proceedings in Informatics (LIPIcs), pages 27:1–27:14, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. ISBN 978-3-95977-109-2. DOI: 10.4230/LIPIcs.ICALP.2019.27. URL http://drops.dagstuhl.de/opus/volltexte/2019/10603.

[37] van Apeldoorn, J. and Gilyén, A. Improvements in Quantum SDP-Solving with Applications. In Baier, C., Chatzigiannakis, I., Flocchini, P. and Leonardi, S., editors, 46th International Colloquium on Automata, Languages, and Programming (ICALP 2019), volume 132 of Leibniz International Proceedings in Informatics (LIPIcs), pages 99:1–99:15, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. ISBN 978-3-95977-109-2. DOI: 10.4230/LIPIcs.ICALP.2019.99. URL http://drops.dagstuhl.de/opus/volltexte/2019/10675.

[38] Kerenidis, I. and Prakash, A. A quantum interior point method for LPs and SDPs. ACM Transactions on Quantum Computing, 1(1): 1–32, 2020. DOI: 10.1145/3406306.

[39] Brandao, F.G.L., Kueng, R. and França, D.S. Faster quantum and classical SDP approximations for quadratic binary optimization. Quantum, 6:625, 2022. DOI: 10.22331/q-2022-01-20-625.

[40] Bharti, K., Haug, T., Vedral, V. and Kwek, L.C. Noisy intermediate-scale quantum algorithm for semidefinite programming. Physical Review A, 105(5):052445, 2022. DOI: 10.1103/PhysRevA.105.052445.

[41] Patel, D., Coles, P.J. and Wilde, M.M. Variational quantum algorithms for semidefinite programming. Quantum, 8:1374, 2024. DOI: 10.22331/q-2024-06-17-1374.

[42] Patti, T.L., Kossaifi, J., Anandkumar, A. and Yelin, S.F. Quantum Goemans-Williamson Algorithm with the Hadamard Test and Approximate Amplitude Constraints. Quantum, 7:1057, 2023. DOI: 10.22331/q-2023-07-12-1057.

[43] Giovannetti, V., Lloyd, S. and Maccone, L. Quantum random access memory. Physical Review Letters, 100(16):160501, 2008. DOI: 10.1103/PhysRevLett.100.160501.

[44] Arora, S. and Kale, S. A combinatorial, primal-dual approach to semidefinite programs. In Proceedings of the thirty-ninth annual ACM symposium on Theory of computing, pages 227–236, 2007. DOI: 10.1145/2837020.

[45] Tsuda, K., Rätsch, G. and Warmuth, M.K. Matrix exponentiated gradient updates for on-line learning and Bregman projection. Journal of Machine Learning Research, 6(Jun):995–1018, 2005. Retrieved

from https://dl.acm.org/doi/10.5555/1046920.1088706.

[46] Peruzzo, A. et al. A variational eigenvalue solver on a photonic quantum processor. Nature communications, 5(1):1–7, 2014. DOI: 10.1038/ncomms5213.

[47] Cerezo, M. et al. Variational quantum algorithms. Nature Reviews Physics, 3(9):625–644, 2021. DOI: 10.1038/s42254-021-00348-9.

[48] Huang, B., Jiang, S., Song, Z., Tao, R. and Zhang, R. A faster quantum algorithm for semidefinite programming via robust IPM framework. arXiv preprint arXiv:2207.11154, 2022. DOI: 10.48550/arXiv.2207.11154.

[49] Muñoz-Arias, M.H., Kourtis, S. and Blais, A. Low-depth clifford circuits approximately solve maxcut. Physical Review Research, 6(2):023294, 2024. DOI: 10.1103/PhysRevResearch.6.023294.

[50] Zhu, L. et al. Adaptive quantum approximate optimization algorithm for solving combinatorial problems on a quantum computer. Physical Review Research, 4(3):033029, 2022. DOI: 10.1103/PhysRevResearch.4.033029.

[51] Wiesner, S. Conjugate coding. ACM Sigact News, 15(1):78–88, 1983. DOI: 10.1145/1008908.1008920.

[52] Nayak, A. Optimal lower bounds for quantum automata and random access codes. In 40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039), pages 369–376. IEEE, 1999. DOI: 10.1109/SFFCS.1999.814608.

[53] Iwama, K., Nishimura, H., Raymond, R. and Yamashita, S. Unbounded-error one-way classical and quantum communication complexity. In Automata, Languages and Programming: 34th International Colloquium, ICALP 2007, Wrocław, Poland, July 9-13, 2007. Proceedings 34, pages 110–121. Springer, 2007. DOI: 10.1007/978-3-540-73420-8˙12.

[54] Hayashi, M., Iwama, K., Nishimura, H., Raymond, R. and Yamashita, S. (4, 1)-quantum random access coding does not exist—one qubit is not enough to recover one of four bits. New Journal of Physics, 8(8):129, 2006. DOI: 10.1088/1367-2630/8/8/129.

[55] Liabøtrø, O. Improved classical and quantum random access codes. Physical Review A, 95(5):052315, 2017. DOI: 10.1103/PhysRevA.95.052315.

[56] Imamichi, T. and Raymond, R. Constructions of quantum random access codes. In Asian Quantum Information Symposium (AQIS), volume 66, 2018. Retrieved from https://research.ibm.com/publications/constructions-of-quantum-random-access-codes.

[57] Mančinska, L. and Storgaard, S.A. The geometry of Bloch space in the context of quantum random access codes. Quantum Information Processing, 21(4):143, 2022. DOI: 10.1007/s11128-022-03470-4.

[58] Teramoto, K., Raymond, R., Wakakuwa, E. and Imai, H. Quantum-Relaxation Based Optimization Algorithms: Theoretical Extensions. arXiv preprint arXiv:2302.09481, 2023. DOI: 10.48550/arXiv.2302.09481.

[59] Ambainis, A., Leung, D., Mancinska, L. and Ozols, M. Quantum random access codes with shared randomness. arXiv preprint arXiv:0810.2937, 2008. DOI: 10.48550/arXiv.0810.2937.

[60] Pawłowski, M. and Żukowski, M. Entanglement-assisted random access codes. Physical Review A, 81(4):042326, 2010. DOI: 10.1103/PhysRevA.81.042326.

[61] Tavakoli, A., Hameedi, A., Marques, B. and Bourennane, M. Quantum random access codes using single d-level systems. Physical Review Letters, 114(17):170502, 2015. DOI: 10.1103/PhysRevLett.114.170502.

[62] Ben-Aroya, A., Regev, O. and De Wolf, R. A hypercontractive inequality for matrix-valued functions with applications to quantum computing and ldcs. In 2008 49th Annual IEEE Symposium on Foundations of Computer Science, pages 477–486. IEEE, 2008. DOI: 10.1109/FOCS.2008.45.

[63] Doriguello, J.F. and Montanaro, A. Quantum random access codes for boolean functions. Quantum, 5:402, 2021. DOI: 10.22331/q-2021-03-07-402.

[64] Yano, H., Suzuki, Y., Itoh, K.M., Raymond, R. and Yamamoto, N. Efficient discrete feature encoding for variational

quantum classifier. IEEE Transactions on Quantum Engineering, 2:1–14, 2021. DOI: 10.1109/QCE49297.2020.00012.

Table 3: Measurement outcome of the random magic measurement and corresponding classical bits. Here, i_q is the selected $i \in \{1,2,3,4\}$ for qubit q.

i_q	Outcome	$(b_X^{[q]}, b_Y^{[q]}, b_Z^{[q]})$
1	0	$(0,0,0)$
1	1	$(1,1,1)$
2	0	$(0,1,1)$
2	1	$(1,0,0)$
3	0	$(1,0,1)$
3	1	$(0,1,0)$
4	0	$(1,1,0)$
4	1	$(0,0,1)$

A Random Magic Measurements and the Relationship with Classical Shadows

Let us define $(m,1)$-random magic measurements, which is referred to as *magic state rounding* in [11]. In this section, only $(3,1)$-random magic measurements are introduced, as $(2,1)$- and $(1,1)$-random magic measurements are easily derived similarly.

Definition 3 (Random magic measurements [11]) *Let $\mu_i^{[q]\pm}$ be the single qubit $(3,1)$-magic state corresponding to the qth qubit defined as follows:*

$$\mu_1^{[q]+} := \mu_3^{[q]}(0,0,0), \quad \mu_1^{[q]-} := \mu_3^{[q]}(1,1,1),$$

$$\mu_2^{[q]+} := \mu_3^{[q]}(0,1,1), \quad \mu_2^{[q]-} := \mu_3^{[q]}(1,0,0),$$

$$\mu_3^{[q]+} := \mu_3^{[q]}(1,0,1), \quad \mu_3^{[q]-} := \mu_3^{[q]}(0,1,0),$$

$$\mu_4^{[q]+} := \mu_3^{[q]}(1,1,0), \quad \mu_4^{[q]-} := \mu_3^{[q]}(0,0,1),$$

where $\mu_3^{[q]}(b_X^{[q]}, b_Y^{[q]}, b_Z^{[q]})$ is defined as Eq. (9). Random magic measurement is the protocol that obtains $\mathbf{b} \in \{0,1\}^{3n}$ as follows:

1. *Select $i \in \{1,2,3,4\}$ uniformly at random for each q.*
2. *Measure the state ρ in the basis $\{\mu_i^{[q]\pm}\}$ for all qubits $q \in [n]$ with the selected i.*
3. *Convert the measurement outcomes to the corresponding classical bits as listed in Table 3.*

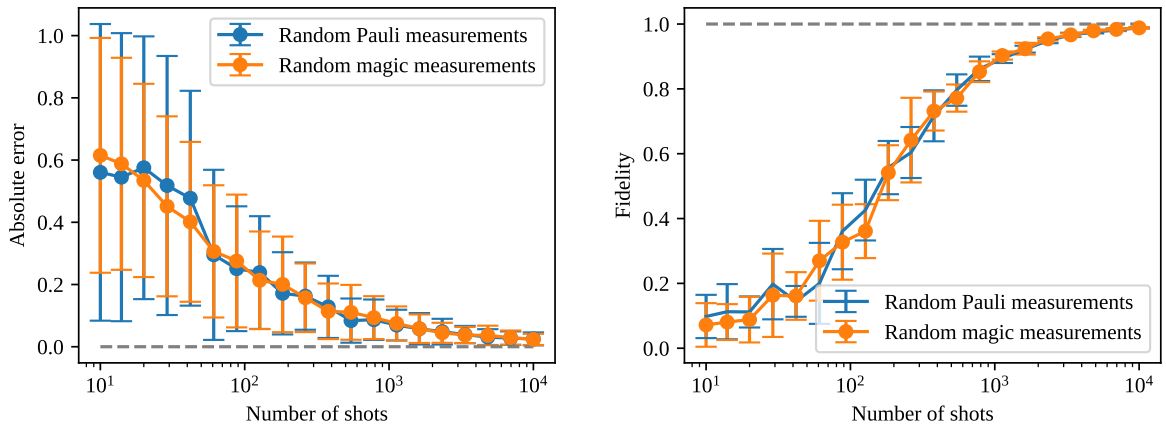
The bit-strings $\mathbf{b}$ obtained by a random magic measurement follows the probability distribution $\mathcal{P}_{\mathrm{f}}(\mathbf{b})$ (see the proof of Theorem 2).

Because the expectation of the k-local Pauli observable in Eq. (19) is written as the expectation of the random variable

$$\hat{o} := 3^{k/2}(-1)^{\sum_{i=1}^{k} b_P^{[q^{(i)}]}}, \tag{S.1}$$

the classical shadow [17] based on random magic measurements can be considered. That is, the expectation value of the k-local Pauli observables can be estimated by averaging $\hat{o}$ computed from the corrected bits $\{b_P^{[q]}\}$ by the random magic measurements. The upper bound of the variance of random variable $\hat{o}$ is readily derived as

$$\mathrm{Var}[\hat{o}] = \mathbb{E}[\hat{o}^2] - (\mathbb{E}[\hat{o}])^2 \leq \mathbb{E}[\hat{o}^2] = 3^k, \tag{S.2}$$



(a) (b)

Figure S.1: Accuracy comparison of a classical shadow obtained using random Pauli measurements and that obtained using $(3, 1)$-random magic measurements. (a) Absolute error of the expectation value of a 2-local Pauli observable in a ten qubit system; (b) Fidelity between the true state and the one obtained using state tomography in a five qubit system.

which is equivalent to that of random Pauli measurements [17]. The classical snapshot based on random magic measurement is

$$\hat{\rho} = \bigotimes_{q=0}^{n-1} \left(3U^{[q]\dagger} |\hat{b}^{[q]}\rangle\langle\hat{b}^{[q]}| U^{[q]} - I \right), \quad U^{[q]} \sim \mathcal{U}_{\mathrm{ms}}, \tag{S.3}$$

where $\hat{b}^{[q]}$ is the measurement outcome of the qth qubit. Equation (S.3) is the same as that of random Pauli measurements except for the unitary ensemble $\mathcal{U}_{\mathrm{ms}}$. In the case of random magic measurements, the unitary ensemble is $\mathcal{U}_{\mathrm{ms}} = \{U_i\}_{i=1}^4$, where $\mu_i^+ := U_i|0\rangle\langle 0|U_i^\dagger$ and $\mu_i^- := U_i|1\rangle\langle 1|U_i^\dagger$. Note that U_1 can be expressed as $U_1^\dagger = e^{-itX}e^{-isZ}$ where $s = \pi/8$ and $t = \arccos(\sqrt{(1+1/\sqrt{3})/2})$, and $U_2^\dagger = U_1^\dagger X$, $U_3^\dagger = U_1^\dagger Y$, $U_4^\dagger = U_1^\dagger Z$ [11].

To demonstrate the performance of random magic measurements, two numerical experiments were carried out.

Expectation Estimation A state ρ is sampled from ten-qubit Haar's random states and is used throughout the experiments. A hundred 2-local Pauli observables are randomly generated and their expectations are evaluated using random Pauli measurements and random magic measurements, respectively. Although the expectation of the k-local Pauli observable is within $[-1, 1]$, the range of the expectation estimated by random magic measurements is $[-3^{k/2}, 3^{k/2}]$. To mitigate the anomaly estimation, the expectation value estimated by the random magic measurements is truncated so that the value is within $[-1, 1]$. In Fig. S.1(a), the accuracy of the expectation evaluation of 2-local Pauli observable is shown. The dependence of the accuracy of the expectation evaluation on the number of shots is almost the same for both the random Pauli measurements and random magic measurements, which is as expected because the variance upper bounds of these methods are the same.

State Tomography A state ρ is sampled from five-qubit Haar's random states and used throughout the experiments. N snapshots, $\{\hat{\rho}_i\}_{i=1}^N$, are collected using random Pauli measurements and the random magic measurements. The candidate matrix is then defined as $\hat{\rho} = \sum_{i=1}^N \hat{\rho}_i/N$. Here, in general, $\hat{\rho}$ does not satisfy the property of the density matrix, i.e., $\mathrm{tr}(\hat{\rho}) = 1$ and $\hat{\rho} \succeq 0$. To avoid improper estimation of the fidelity, the nearest state to $\hat{\rho}$ that satisfied the property of the density matrix is obtained by the optimization, and is defined as the result of the state tomography. The optimization

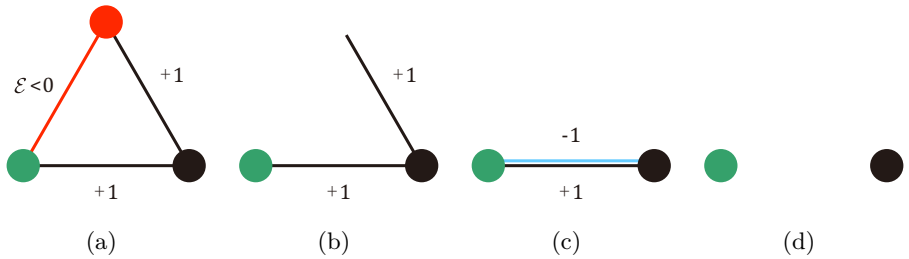


Figure S.2: Schematic illustration of the node isolation process. (a) The red and green nodes represent the node to be deleted and the node to be kept, respectively. The red edge indicates the candidate edge. The ensemble edge energy of the candidate edge $\mathcal{E}$ is negative. (b) After removing the candidate edge. (c) Multiplying the sign of the ensemble edge energy of the candidate edge and reconnecting it to the green node. (d) After removing the zero-weight edge.

process is as follows. The n-qubit nearest state is parametrized as $|\hat{\psi}(\boldsymbol{\theta})\rangle_k = f_k/\sqrt{\sum_{k'=1}^{2^n} f_{k'}^* f_{k'}}$ where $f_k = \theta_k^{\mathrm{Re}} + \mathrm{i}\theta_k^{\mathrm{Im}}$ and $\boldsymbol{\theta} = \{\theta_k^{\mathrm{Re}}, \theta_k^{\mathrm{Im}}\}_{k=1}^{2^n}$. Here, $|\hat{\psi}(\boldsymbol{\theta})\rangle_k$ denotes the kth component of $|\hat{\psi}(\boldsymbol{\theta})\rangle$ in the computational basis. The objective function is defined as $\mathcal{L}(\boldsymbol{\theta}) := \|\hat{\rho} - |\hat{\psi}(\boldsymbol{\theta})\rangle\langle\hat{\psi}(\boldsymbol{\theta})|\|_{\mathrm{F}}$ where $\|\bullet\|_{\mathrm{F}}$ is the Frobenius norm. Parameter $\boldsymbol{\theta}$ is optimized by L-BFGS to minimize $\mathcal{L}(\boldsymbol{\theta})$. Then, the tomography result is defined as $\overline{\rho} := |\hat{\psi}(\boldsymbol{\theta}^*)\rangle\langle\hat{\psi}(\boldsymbol{\theta}^*)|$, where $\boldsymbol{\theta}^*$ is the optimized parameter. According to Fig. S.1(b), the dependence of the state tomography accuracy on the number of shots is almost the same for both the random Pauli measurements and random magic measurements.

B Node Isolation Problem

When the edge weights of problem graph G are integers, the number of isolated nodes, which are defined as nodes with undetermined parity with respect to other nodes, will increase as the iteration progresses. An example is illustrated in Fig. S.2. In Fig. S.2(a), the node to be deleted and the node to be kept are shown in red and green, respectively. The red edge indicates the candidate whose ensemble edge energy $\mathcal{E}$ is negative. After removing the candidate edge (Fig. S.2(b)), the edge that was connected to the red node at one end is reconnected to the green node (Fig. S.2(c)). At this time, the edge weight is multiplied by the sign of $\mathcal{E}$. As a result, the weight of the edge the bottom of the illustration is updated to zero (Fig. S.2(d)), which means that the black node becomes isolated. If there are N_{iso} isolated nodes after the parity determination procedure, a brute-force search must be carried out on the $M + N_{\mathrm{iso}}$ nodes, which is intractable when there are several tens of isolated nodes. One easy way to mitigate the node isolation problem is to apply a small perturbation to all edge weights at random. When perturbed edge weights are used, the probability that a reconnected edge has zero weight becomes negligibly small.

C Proofs

Proof of Theorem 2. Consider a quantum channel $\mathcal{E}$ that measures an n qubit state ρ with the measurement basis $\mu_m(\mathbf{b})$, which is an n qubit product state of magic states selected uniformly at random:

$$\mathcal{E}(\rho) := \frac{1}{2^{(m-1)n}} \sum_{\mathbf{b}\in\{0,1\}^{mn}} \mathrm{tr}(\mu_m(\mathbf{b})\rho)\mu_m(\mathbf{b}), \tag{S.4}$$

where $2^{(m-1)n}$ is the total number of $(m,1)$-magic state bases living in n-qubit quantum state space. According to [11], the following holds:

$$\mathrm{tr}(P^{\otimes k}\mathcal{E}(\rho)) = \frac{1}{m^k}\mathrm{tr}(P^{\otimes k}\rho) \tag{S.5}$$

In addition, it is easily confirmed that

$$\mathrm{tr}(P^{\otimes k}\mu_m(\mathbf{b})) = \frac{(-1)^{\sum_{i=1}^{k} b_P^{[q^{(i)}]}}}{m^{k/2}}. \tag{S.6}$$

Substituting Eqs. (S.4) and (S.6) into Eq. (S.5) yields

$$\begin{aligned}
\mathrm{tr}(P^{\otimes k}\rho) &= m^k \mathrm{tr}(P^{\otimes k}\mathcal{E}(\rho)) \\
&= m^k \mathrm{tr}\left(P^{\otimes k}\frac{1}{2^{(m-1)n}}\sum_{\mathbf{b}\in\{0,1\}^{mn}} \mathrm{tr}(\mu_m(\mathbf{b})\rho)\mu_m(\mathbf{b})\right) \\
&= \frac{m^k}{2^{(m-1)n}}\sum_{\mathbf{b}\in\{0,1\}^{mn}} \mathrm{tr}(P^{\otimes k}\mu_m(\mathbf{b}))f_m(\mathbf{b},\rho) \\
&= \frac{m^{k/2}}{2^{(m-1)n}}\sum_{\mathbf{b}\in\{0,1\}^{mn}} (-1)^{\sum_{i=1}^{k} b_P^{[q^{(i)}]}} f_m(\mathbf{b},\rho),
\end{aligned} \tag{S.7}$$

where

$$f_m(\mathbf{b},\rho) := \mathrm{tr}(\mu_m(\mathbf{b})\rho) \tag{S.8}$$

is a fidelity of $\mu_m(\mathbf{b})$ and ρ. Meanwhile, it is easily confirmed that

$$1 = \mathrm{tr}(\mathcal{E}(\rho)) = \frac{1}{2^{(m-1)n}}\sum_{\mathbf{b}'\in\{0,1\}^{mn}} f(\mathbf{b}',\rho). \tag{S.9}$$

Then, the following holds:

$$\begin{aligned}
\mathrm{tr}(P^{\otimes k}\rho) &= m^{k/2}\,\frac{\sum_{\mathbf{b}\in\{0,1\}^{mn}}(-1)^{\sum_{i=1}^{k} b_P^{[q^{(i)}]}} f_m(\mathbf{b},\rho)}{\sum_{\mathbf{b}'\in\{0,1\}^{mn}} f_m(\mathbf{b}',\rho)} \\
&= m^{k/2}\,\mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\left[(-1)^{\sum_{i=1}^{k} b_P^{[q^{(i)}]}}\right],
\end{aligned} \tag{S.10}$$

where

$$\mathcal{P}_m(\mathbf{b};\rho) := \frac{f_m(\mathbf{b},\rho)}{\sum_{\mathbf{b}'\in\{0,1\}^{mn}} f_m(\mathbf{b}',\rho)} \tag{S.11}$$

□

Proof of Corollary 1. Consider that a binary variable of node j, b_j, is mapped to $(q^{(j)}, P^{(j)})$, where $q^{(j)}$ is a qubit index, $P^{(j)} \in \mathfrak{P} \subset \{X, Y, Z\}$ and $|\mathfrak{P}| = m \in \{1, 2, 3\}$ (see Sec. 2.3). Equation (S.7) in the case of $k = 2$ reduces

$$\mathrm{tr}(P^{(j)}_{q^{(j)}} P^{(k)}_{q^{(k)}}\rho) = m\,\mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\left[(-1)^{b_j+b_k}\right]. \tag{S.12}$$

The left-hand side of Eq. (S.12) is written as $\mathrm{tr}(P_{\langle j\rangle}P_{\langle k\rangle}\rho)$ in the main body. Substituting the $(m, 1)$-QRAC Hamiltonian (Eq. (8)) into Eq. (S.12) leads to

$$\mathrm{tr}(H_m\,\rho) = \sum_{(j,k)\in E}\frac{\mathrm{tr}(\rho) - m\mathrm{tr}(P_{\langle j\rangle}P_{\langle k\rangle}\,\rho)}{2}w_{jk} = \sum_{(j,k)\in E}\frac{1 - m^2\mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\left[(-1)^{b_j+b_k}\right]}{2}w_{jk}. \tag{S.13}$$

According to the definition of $\mathtt{CW}(\mathbf{b})$ (Eq. (2)),

$$\sum_{(j,k)\in E}\frac{(-1)^{b_j+b_k}}{2}w_{jk} = \sum_{(j,k)\in E}\frac{1}{2}w_{jk} - \mathtt{CW}(\mathbf{b}). \tag{S.14}$$

Substituting Eq. (S.14) into (S.13) gives

$$\mathrm{tr}(H_{\mathrm{QRAC}}\,\rho) = m^2 \mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\big[\mathtt{CW}(\mathbf{b})\big] - \frac{m^2-1}{2}\sum_{(j,k)\in E} w_{jk}. \tag{S.15}$$

□

Proof of Corollary 2. According to Eq. (8),

$$\begin{aligned} H_m^2 &= \Big(\sum_{(j,k)\in E}\frac{I - mP_{\langle j\rangle}P_{\langle k\rangle}}{2}w_{jk}\Big)\cdot\Big(\sum_{(j',k')\in E}\frac{I - mP_{\langle j'\rangle}P_{\langle k'\rangle}}{2}w_{j'k'}\Big) \\ &= \frac{m^2}{4}\sum_{(j,k)\in E}\sum_{(j',k')\in E} P_{\langle j\rangle}P_{\langle k\rangle}P_{\langle j'\rangle}P_{\langle k'\rangle}w_{jk}w_{j'k'} - \frac{mW}{2}\sum_{(j,k)\in E}P_{\langle j\rangle}P_{\langle k\rangle}w_{jk} + \frac{1}{4}W^2 \end{aligned},$$

where $W := \sum_{(j,k)\in E} w_{jk}$. The expectation value of H_m^2 for an arbitrary state ρ is calculated as

$$\begin{aligned} \mathrm{tr}(H_m^2\rho) &= \frac{m^2}{4}\sum_{(j,k)\in E}\sum_{(j',k')\in E}\mathrm{tr}(P_{\langle j\rangle}P_{\langle k\rangle}P_{\langle j'\rangle}P_{\langle k'\rangle}w_{jk}w_{j'k'}\rho) - \frac{mW}{2}\sum_{(j,k)\in E}\mathrm{tr}(P_{\langle j\rangle}P_{\langle k\rangle}w_{jk}\rho) + \frac{1}{4}W^2 \\ &= \frac{m^2}{4}\sum_{(j,k)\in E}\sum_{(j',k')\in E}\mathrm{tr}(P_{\langle j\rangle}P_{\langle k\rangle}P_{\langle j'\rangle}P_{\langle k'\rangle}w_{jk}w_{j'k'}\rho) + W\mathrm{tr}(H_m\rho) - \frac{1}{4}W^2. \end{aligned}$$

Meanwhile, from Eq. (21),

$$\begin{aligned} (\mathtt{CW}(\mathbf{b}))^2 &= \Big(\sum_{(j,k)\in E}\frac{1-m^2(-1)^{b_j+b_k}}{2m^2}w_{jk} + \frac{m^2-1}{2m^2}W\Big)\Big(\sum_{(j',k')\in E}\frac{1-m^2(-1)^{b_{j'}+b_{k'}}}{2m^2}w_{j'k'} + \frac{m^2-1}{2m^2}W\Big) \\ &= \frac{1}{4}\sum_{(j,k)\in E}\sum_{(j',k')\in E}(-1)^{b_j+b_k+b_{j'}+b_{k'}}w_{jk}w_{j'k'} - \frac{1}{2}W\sum_{(j,k)\in E}(-1)^{b_j+b_k}w_{jk} + \frac{1}{4}W^2. \end{aligned}$$

Using Theorem 2 and substituting the expression of $\mathrm{tr}(H_m^2\rho)$ derived above yields

$$\mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\big[(\mathtt{CW}(\mathbf{b}))^2\big] = \frac{1}{m^4}\mathrm{tr}(H_m^2\rho) + \frac{m^2-1}{m^4}W\mathrm{tr}(W\rho) + \frac{(m^2-1)^2}{4m^2}W^2.$$

Combining with Eq. (21), the variance of the cut weight can be written as

$$\begin{aligned} \mathrm{Var}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\big[\mathtt{CW}(\mathbf{b})\big] &= \mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\big[\mathtt{CW}(\mathbf{b})^2\big] - \mathbb{E}_{\mathbf{b}\sim\mathcal{P}_m(\mathbf{b};\rho)}\big[\mathtt{CW}(\mathbf{b})\big]^2 \\ &= \frac{1}{m^4}\mathrm{tr}(H_m^2\rho) + \frac{m^2-1}{m^4}W\mathrm{tr}(H_m\rho) + \frac{(m^2-1)^2}{4m^4}W^2 \\ &\quad - \Big(\frac{1}{m^2}\mathrm{tr}(H_m\rho) + \frac{m^2-1}{2m^2}W\Big)^2 \\ &= \frac{1}{m^4}\Big(\mathrm{tr}(H_m^2\rho) - \mathrm{tr}(H_m\rho)^2\Big). \end{aligned}$$

□

Proof of Corollary 3. According to Eq. (19),

$$\begin{aligned} \mathcal{E}_j &:= \mathrm{tr}(P_{\langle j\rangle}\rho) \\ &= \sqrt{m}\,\mathbb{E}_{\mathbf{b}\sim\mathcal{P}_{\mathrm{f}}(\mathbf{b};\rho)}[(-1)^{b_j}] \\ &= \sqrt{m}\,[\mathcal{P}(b_j=0)\cdot(+1) + \mathcal{P}(b_j=1)\cdot(-1)] \\ &= \sqrt{m}\,[\mathcal{P}(b_j=0) - (1-\mathcal{P}(b_j=0))] \\ &= 2\sqrt{m}\mathcal{P}(b_j=0) - \sqrt{m}. \end{aligned}$$

Then,

$$\mathcal{P}(b_j = 0) = \frac{1}{2} + \frac{1}{2\sqrt{m}}\mathcal{E}_j$$

$$\mathcal{P}(b_j = 1) = 1 - \mathcal{P}(b_j = 0) = \frac{1}{2} - \frac{1}{2\sqrt{m}}\mathcal{E}_j.$$

Similarly, when $q^{(j)} \neq q^{(k)}$,

$$\begin{aligned}
\mathcal{E}_{jk} &:= \mathrm{tr}(P_{\langle j\rangle}P_{\langle k\rangle}\rho) \\
&= m\,\mathbb{E}_{\mathbf{b}\sim\mathcal{P}_{\mathrm{f}}(\mathbf{b};\rho)}[(-1)^{b_j+b_k}] \\
&= m\left[\mathcal{P}(b_j = b_k)\cdot(+1) + \mathcal{P}(b_j \neq b_k)\cdot(-1)\right] \\
&= m\left[\mathcal{P}(b_j = b_k) - (1 - \mathcal{P}(b_j = b_k))\right] \\
&= 2m\mathcal{P}(b_j = b_k) - m.
\end{aligned}$$

Then,

$$\mathcal{P}(b_j = b_k) = \frac{1}{2} + \frac{1}{2m}\mathcal{E}_{jk}$$

$$\mathcal{P}(b_j \neq b_k) = 1 - \mathcal{P}(b_j = b_k) = \frac{1}{2} - \frac{1}{2m}\mathcal{E}_{jk}.$$

However, when $q^{(j)} = q^{(k)}$

$$\begin{aligned}
\mathcal{P}(b_j = b_k) &= \mathcal{P}(b_j = 0)\cdot\mathcal{P}(b_k = 0) + \mathcal{P}(b_j = 1)\cdot\mathcal{P}(b_k = 1) \\
&= \frac{1}{2} + \frac{1}{2m}\mathcal{E}_j\mathcal{E}_k
\end{aligned}$$

and

$$\begin{aligned}
\mathcal{P}(b_j \neq b_k) &= 1 - \mathcal{P}(b_j = b_k) \\
&= \frac{1}{2} - \frac{1}{2m}\mathcal{E}_j\mathcal{E}_k.
\end{aligned}$$

□

D Matrix Product State

Let $\Psi \in \mathbb{C}^{2^n}$ be n-qubit quantum state. Generally, Ψ can be represented as the matrix multiplication of n matrices as

$$\Psi = A^{[n-1]}A^{[n-2]}\cdots A^{[1]}A^{[0]}, \tag{S.16}$$

where

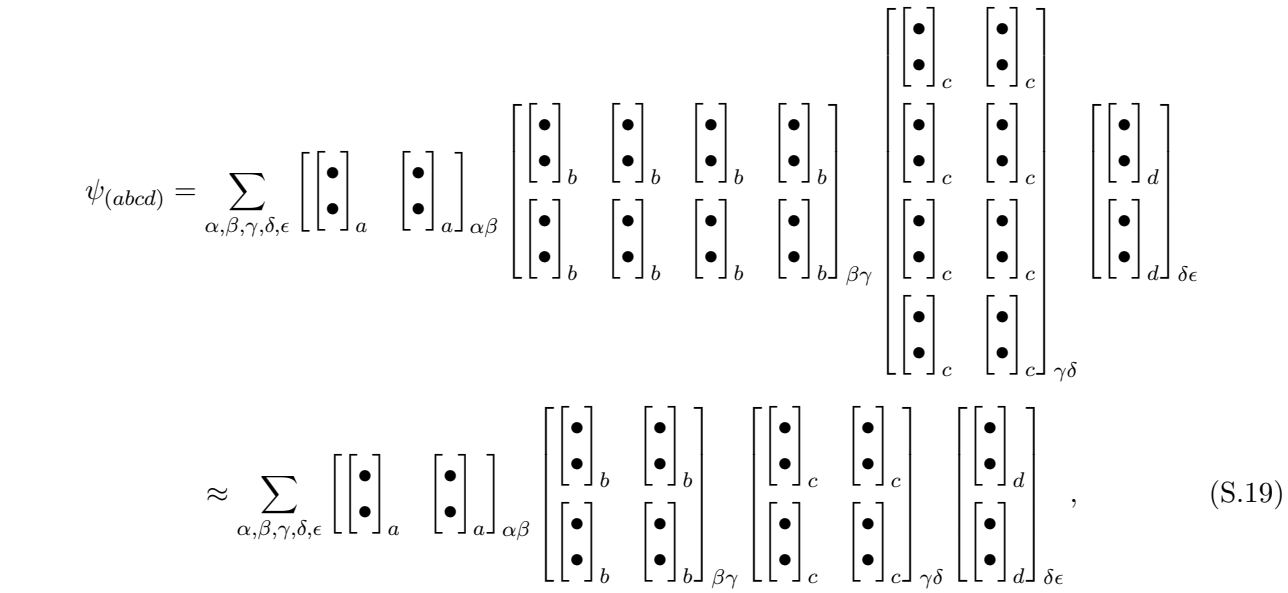
$$A^{[i]} \in \mathbb{C}^{\min(2^{i+1},2^{n-i-1},\chi)\times 2\times\min(2^i,2^{n-i-1},\chi)} \tag{S.17}$$

and

$$(A^{[i]}A^{[j]})_{k(ab)m} = \sum_{l=1}^{\min(2^i,2^{n-i-1},\chi)} (A^{[i]})_{kal}(A^{[j]})_{lbm}. \tag{S.18}$$

The subscript in the parentheses, (ab), summarizes two indices in one index, i.e., $(a,b) = (1,1),(1,2),\ldots,(2,1),(2,2),\ldots$. The expression of Eq. (S.16) is called the MPS [27]. The dimension between matrices is called the bond dimension and can be restricted to χ at the expense of representation power. The number of components of an n-qubit state is 2^n, whereas the total number

of matrix components of MPS is $\sim 2n\chi^2$. For example, a 4-qubit state $|\psi\rangle \in \mathbb{C}^{2^4}$ with bond dimension $\chi = 2$ can be represented as



where $\bullet$ indicates the matrix element and the subscripts represent the index symbols of the inner submatrix elements.

Let P be the identity or Pauli matrix and

$$H = \sum_{k=1}^{K} C_k \bigotimes_{l=0}^{n-1} P_{kl} \tag{S.20}$$

be a n-qubit Hamiltonian that consists of a weighted sum of K Pauli products. Similar to the MPS, Eq. (S.20) can be represented as matrix multiplication of n matrices as

$$H = B^{[n-1]} B^{[n-2]} \cdots B^{[1]} B^{[0]}, \tag{S.21}$$

where

$$B^{[i]} \in \begin{cases} \mathbb{C}^{K\times 2\times 2\times 1} & i = 0 \\ \mathbb{C}^{1\times 2\times 2\times K} & i = n-1 \\ \mathbb{C}^{K\times 2\times 2\times K} & \text{else} \end{cases}, \tag{S.22}$$

$$\begin{cases} (B^{[i]})_{kaa'1} = C_k (P_{ki})_{aa'} & i = 0 \\ (B^{[i]})_{1aa'k} = (P_{ki})_{aa'} & i = n-1 \\ (B^{[i]})_{kaa'k} = (P_{ki})_{aa'} & \text{else} \end{cases} \tag{S.23}$$

and

$$(B^{[i]} B^{[j]})_{l(ab)(a'b')m} = \sum_{k=1}^{K} (B^{[i]})_{laa'k} (B^{[j]})_{kbb'm}. \tag{S.24}$$

For example,

$$\begin{aligned} H_{(abcd)(a'b'c'd')} &= (C_1 IXYI + C_2 IZII + C_3 IYIX)_{(abcd)(a'b'c'd')} \\ &= \sum_{\alpha,\beta,\gamma,\delta,\epsilon} \begin{bmatrix} [I]_{aa'} & [I]_{aa'} & [I]_{aa'} \end{bmatrix}_{\alpha\beta} \begin{bmatrix} [X]_{bb'} & & \\ & [Z]_{bb'} & \\ & & [Y]_{bb'} \end{bmatrix}_{\beta\gamma} \begin{bmatrix} [Y]_{cc'} & & \\ & [I]_{cc'} & \\ & & [I]_{cc'} \end{bmatrix}_{\gamma\delta} \begin{bmatrix} [C_1 I]_{dd'} \\ [C_2 I]_{dd'} \\ [C_3 X]_{dd'} \end{bmatrix}_{\delta\epsilon} \end{aligned} \tag{S.25}$$

The expression of Eq. (S.21) is called the matrix product operator (MPO) [27].

The MPO is a set of sparse matrices and thus can be efficiently stored in classical memory in COOrdinate (COO), Compressed Sparse Row (CSR), or Compressed Sparse Column (CSC) formats.

Using the MPS and MPO, the expectation value $\langle\Psi|H|\Psi\rangle$ can be efficiently calculated by the matrix multiplication of $\{A^{[i]}\}$ and $\{B^{[i]}\}$.

All edge energy can be simultaneously evaluated by modifying the MPO of the problem Hamiltonian.

$$\begin{cases} (B^{[i]})_{kaa'1} = (P_{ki})_{aa'} & i = 0 \\ \\ (B^{[i]})_{kaa'k} = (P_{ki})_{aa'} & \text{else} \end{cases} \tag{S.26}$$

For example,

$$\begin{bmatrix} C_1 IXYI_{(abcd)(a'b'c'd')} \\ C_2 IZII_{(abcd)(a'b'c'd')} \\ C_3 IYIX_{(abcd)(a'b'c'd')} \end{bmatrix}$$

$$= \sum_{\alpha,\beta,\gamma,\delta,\epsilon} \begin{bmatrix} [I]_{aa'} & & \\ & [I]_{aa'} & \\ & & [I]_{aa'} \end{bmatrix}_{\alpha\beta} \begin{bmatrix} [X]_{bb'} & & \\ & [Z]_{bb'} & \\ & & [Y]_{bb'} \end{bmatrix}_{\beta\gamma} \begin{bmatrix} [Y]_{cc} & & \\ & [I]_{cc'} & \\ & & [I]_{cc'} \end{bmatrix}_{\gamma\delta} \begin{bmatrix} [C_1 I]_{dd'} \\ [C_2 I]_{dd'} \\ [C_3 X]_{dd'}. \end{bmatrix}_{\delta\epsilon} \tag{S.27}$$

E Implementation Details for GW, CirCut, the RQAOA, and QRAO

For GW, an official MATLAB implementation of SDPT3 [28, 29] is used to solve the SDP. The gap tolerance is set to 10^{-9} for high-precision calculation. For hyperplane cutting, which is a post-processing step in GW performed after computing the SDP solution, in-house Python code is used. The number of hyperplane cuttings is set to $10,000$.

For CirCut [16], the official Fortran implementation (`https://www.cmor-faculty.rice.edu/~zhang/circut/index.html`) is used. The hyperparameters for CirCut are not changed from the default ones.

For RQAOA, in-house Fortran code is used. The analytic form of $\langle\psi_{\mathrm{QAOA}}(\boldsymbol{\theta})|Z_jZ_k|\psi_{\mathrm{QAOA}}(\boldsymbol{\theta})\rangle$ [3] is used to evaluate the expectation value of the problem Hamiltonian and edge energy, where $|\psi_{\mathrm{QAOA}}(\boldsymbol{\theta})\rangle = e^{i\beta B}e^{i\gamma C}|+\rangle^{\otimes n}$ is a level-1 QAOA ansatz, $B = \sum_{i=0}^{n-1} X_i$, $C = \sum_{(j,k)\in E} w_{jk}Z_jZ_k$, and $\boldsymbol{\theta} = \{\beta, \gamma\}$. Only the optimal γ is determined using a grid search; the optimal β is calculated for each γ using the same method used in [14]. We derived the concrete representation of the optimal β as

$$\beta^* = -\frac{1}{4}\arctan\left(\frac{\frac{1}{2}(R_{11} - R_{22} - R_{33} + R_{44}) + R_{14} - R_{23}}{-I_{12} - I_{13} + I_{24} + I_{34}}\right) + \frac{\pi}{8}, \tag{S.28}$$

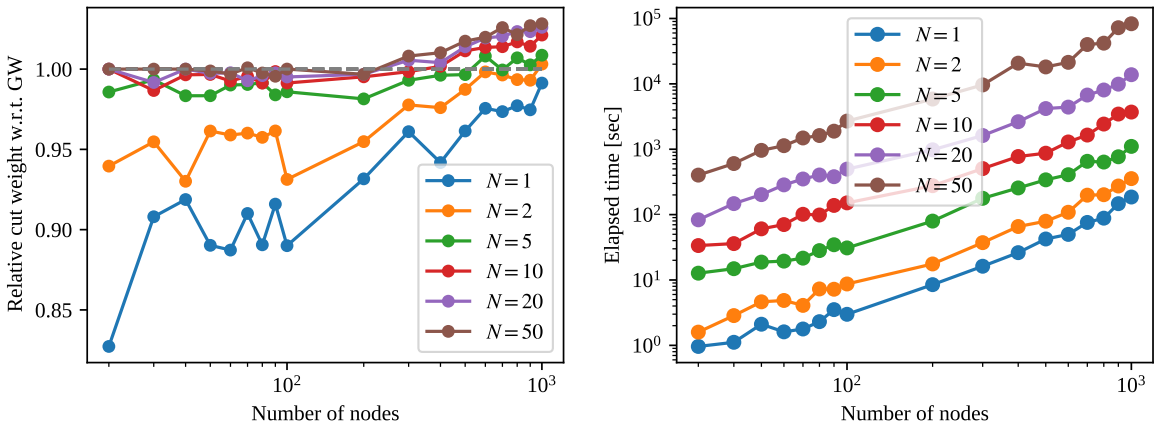
where

$$\begin{bmatrix} R_{11} & R_{12} + iI_{12} & R_{13} + iI_{13} & R_{14} + iI_{14} \\ R_{12} - iI_{12} & R_{22} & R_{23} + iI_{23} & R_{24} + iI_{24} \\ R_{13} - iI_{13} & R_{23} - iI_{23} & R_{33} & R_{34} + iI_{34} \\ R_{14} - iI_{14} & R_{24} - iI_{24} & R_{34} - iI_{34} & R_{44} \end{bmatrix} := \sum_{(u,v)\in E} w_{uv}\rho_{uv} \tag{S.29}$$

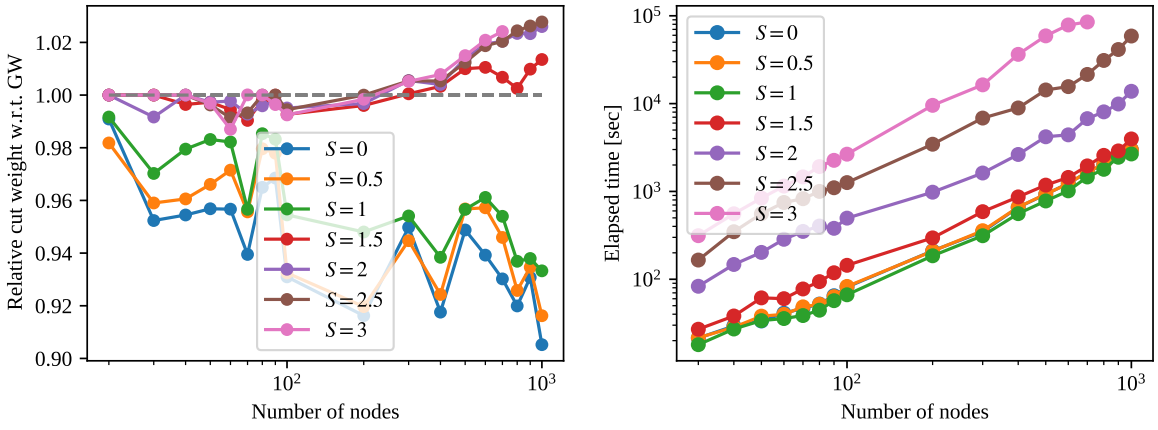
and

$$\rho_{uv} := \mathrm{tr}_{uv}\left(e^{i\gamma C}|+\rangle^{\otimes n}\langle+|^{\otimes n}e^{-i\gamma C}\right). \tag{S.30}$$

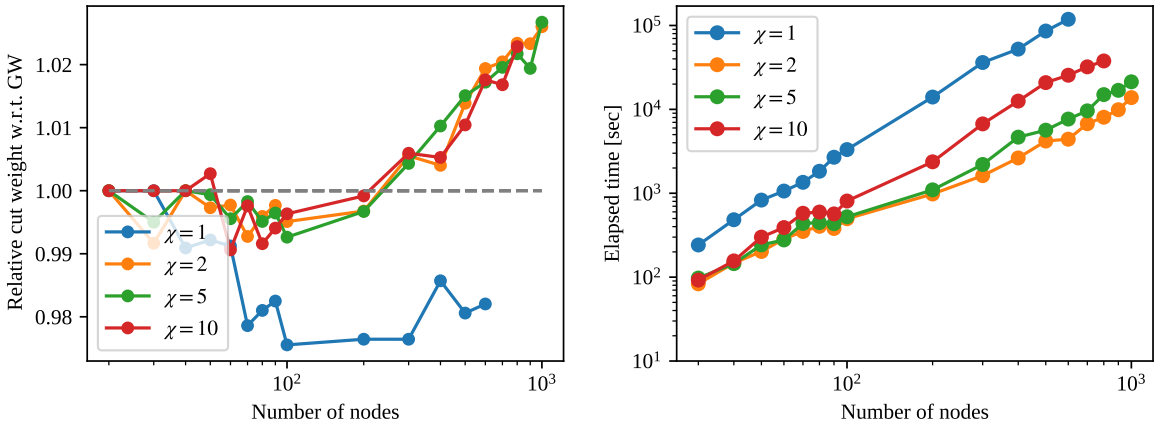
Here, $\mathrm{tr}_{uv}(\sigma)$ denotes the reduced state obtained by tracing out, excepting the uth and vth qubits from σ. For an algorithm to calculate ρ_{uv}, see [14]. The parameter spaces are set to $\beta \in [0, \frac{\pi}{2}]$ and $\gamma \in [0, \pi]$. Once the optimal γ is found in 50 equidistant grids of $[0, \pi]$, the more precise solution is searched for in the fine-grained grids around the course-grained solution with a mesh size is $1/50$. Fine-graining is repeated twice, resulting in a mesh size of $1/250$ of the parameter space.



(a)



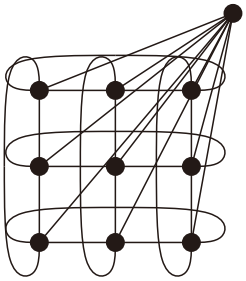
(b)



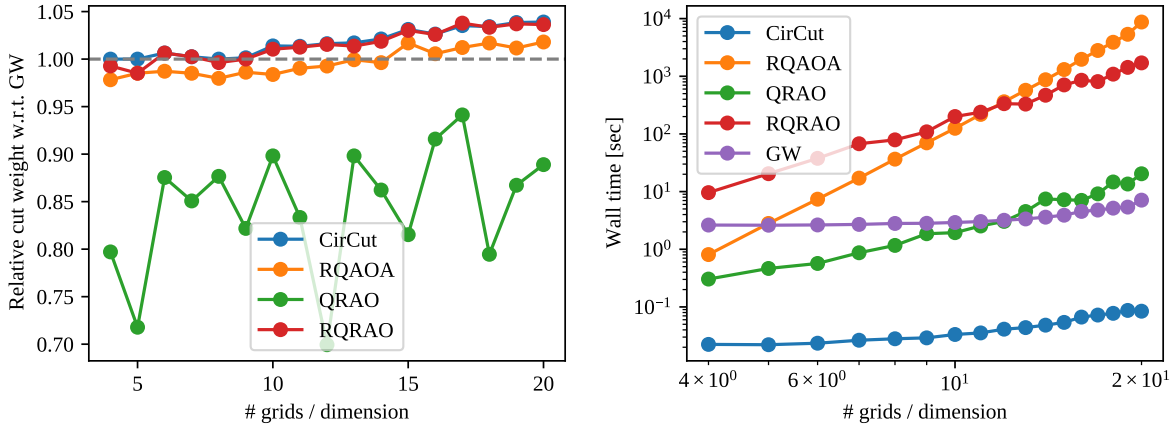
(c)

Figure S.3: Results of the ablation study. Only one hyperparameter is varied for each trial. The problem graph instances are 3-regular graphs with ± 1 edge weights. For each number of nodes, 10 graph instances are randomly generated. The varying hyperparameters are the (a) number of ensembles N; (b) scale factor S; and (c) bond dimension χ.

For QRAO, in-house Python code is used. The MPS ansatz with bond dimension $\chi = 2$ is used for $|\psi(\boldsymbol{\theta})\rangle$ and Pauli rounding is used for decoding. The same optimizer as used for RQRAO, L-BFGS with the strong Wolfe condition, is used to optimize $|\psi(\boldsymbol{\theta})\rangle$.



(a)



(b)

Figure S.4: (a) Example of 2D toric graph with one external nodes of size 3×3 (# girds / dimension $= 3$); (b) Summary of results for 2D-toric graphs.

F Additional Sensitivity Analysis Results (ExpHyp)

Here, the sensitivities of all RQRAO hyperparameters except for the number of embeddings per qubit m on the performance of the cut weight and runtime are reported. The results are summarized in Fig. S.3. In brief, a higher cut weight is obtained through a longer runtime.

Except for the number of embeddings per qubit m, the sensitive hyperparameters with respect to cut weight performance are the number of ensembles N and scale factor S. In Fig. S.3(a), the cut weight increases as the number of ensembles N increases. In addition, as the number of ensembles increases, the number of edges whose parities are simultaneously determined in one optimization step tends to decrease. This is because the edges, which are a candidate for determining the parity when the number of ensembles is small, drop out as the accuracy of the parity confidence improves, which is achieved by increasing the number of ensembles. This trend (i.e., the number of candidates decreases with as the accuracy of the parity confidence increases), can be seen in the larger scale factor S in Fig. S.3(b). A larger S tends to decrease the number of candidates because the number of edges with nonzero $\mathcal{E}_{jk}$ decreases.

The effect of the bond dimension χ on the cut weight performance needs to be carefully considered. Recall that a large entanglement can be treated using a large bond dimension. The $\chi = 1$ result is worse both in terms of relative cut weight and runtime than the $\chi > 1$ results. In the case of $\chi = 1$, $|\psi(\boldsymbol{\theta})\rangle$ is restricted to a direct product state that tends to be trapped in a lower energy state than in the $\chi > 1$ case because the search space is much smaller. As a result, each edge energy in an ensemble, $\mathcal{E}_{jk}^{(t)}$, is no longer biasedly distributed in either positive or negative sides because the lower energy state includes smaller information of a large cut weight. By contrast, when $\chi > 1$, no significant difference in the relative cut weight can be seen. This does not mean, however, that a large entanglement will not affect the cut weight performance because even the value of $\chi = 10$, which was the maximum value

considered in the experiment, is small compared with the maximum bond dimension that the system can take, which is $\chi_{\max} = 2^{\lfloor n/2 \rfloor}$. For example, consider the case of $|V| = 300$. The number of qubits is assumed to be $300/3 = 100$, which can be achieved by a $(3,1)$-QRAC formulation if the graph is sufficiently sparse. In this case, $\chi_{\max} = 2^{\lfloor 99/2 \rfloor} \approx 10^{14}$, which is much larger than the experiments considered in this study. To take into account a large bond dimension within a reasonable runtime, computation using a real quantum device would be required.

G Results for 2D Toric Graphs

While the specific characteristics of instances where RQAOA performs particularly well are not fully understood, in [3] RQAOA was compared against GW using 2D toric graphs with one extra node connected with all other nodes (Fig. S.4(a)), referred to as G_{toric}. To verify that our proposed model also surpasses RQAOA on G_{toric}, we conducted numerical simulations for G_{toric} under the same settings as `ExpScale`. In Fig. S.4(b), it is evident that RQAOA outperforms GW in terms of cut weight when the number of grids per dimension exceeds 15, consistent with the findings of [3]. However, both CirCut and our proposed RQRAO surpass RQAOA. Note that the trend in time complexity is nearly identical to the results observed for 3-regular graphs, as shown in Fig. 7(b).

H Related Work

H.1 Classical Algorithm for the MAX-CUT problem

Before reviewing the classical algorithms for the MAX-CUT problem, we reformulate Definition 1 to make the existing classical algorithms easier to understand. Because $(-1)^{b_j+b_k} = v_j v_k$ where $v_j := (-1)^{b_j}$, the MAX-CUT problem can be reformulated as

$$\max_{\mathbf{X}} \mathtt{CW}_{\mathrm{X}}(\mathbf{X}), \tag{S.31}$$

where

$$\mathtt{CW}_{\mathrm{X}}(\mathbf{X}) := \mathrm{tr}(\mathbf{w}\mathbf{F}(\mathbf{X})), \tag{S.32}$$

$\mathbf{F}(\mathbf{X}) := (\mathbf{1} - \mathbf{X})/2$, and $\mathbf{X} := \mathbf{v}\mathbf{v}^{\mathsf{T}}$. By definition, $\mathbf{X}$ is restricted as $\mathrm{diag}(\mathbf{X}) = \mathbf{1}$, $\mathrm{rank}(\mathbf{X}) = 1$, and $\mathbf{X} \succeq 0$.

The best known theoretically guaranteed generic algorithm for the MAX-CUT problem is the GW [7] as stated in Sec. 1. In the GW algorithm, the constraint $\mathrm{rank}(\mathbf{X}) = 1$ is removed. This transforms the MAX-CUT problem to a semidefinite program (SDP), which can be solved in polynomial time for the number of nodes. After solving the SDP, the solution $\mathbf{v} \in \{\pm 1\}^{|V|}$ needs to be extracted from the obtained $\mathbf{X}$. When $\mathbf{X}$ is written in the form $\mathbf{X} = \mathbf{v}\mathbf{v}^{\mathsf{T}}$, the solution is $\mathbf{v}$. If not, as is generally the case here, the obtained $\mathbf{X}$ is decomposed as $\mathbf{X} = \mathbf{L}\mathbf{L}^{\mathsf{T}}$ using, for example, Cholesky decomposition, where $\mathbf{L} = \begin{bmatrix} \mathbf{l}_1 & \dots & \mathbf{l}_{|V|} \end{bmatrix}^{\mathsf{T}} \in \mathbb{R}^{|V| \times |V|}$ is a lower triangular matrix and $\mathbf{l}_i \in \mathbb{R}^{|V|}$ is a unit vector. Then, the solution $\{v_i\}$ is determined by the random projection of $\mathbf{l}_i$ as $v_i = 1$ if $\mathbf{l}_i^{\mathsf{T}}\mathbf{r} \geq 0$ and $v_i = -1$ otherwise. Here, $\mathbf{r} \in \mathbb{R}^{|V|}$ is a random vector uniformly distributed on the $|V|$-dimensional unit sphere. This can be viewed as dividing the set $\{\mathbf{l}_i\}_{i=1}^{|V|}$ into two sets by a hyperplane with $\mathbf{r}$ as the normal vector. When the hyperplane cutting is performed uniformly at random, the lower bound of the expected cut weight is $0.878\,\mathtt{CW}(\mathbf{b}^*)$ [7]. This bound is explained as follows. According to [7, Theorem 3.1], the MAX-CUT problem can be further reformulated as

$$\{\mathbf{l}_i^{\mathrm{opt}}\} := \operatorname*{argmax}_{\{\mathbf{l}_i\}} \mathbb{E}_{\mathbf{b} \sim \mathcal{P}^{\mathrm{MC}}(\mathbf{b}; \{\mathbf{l}_i\})}[\mathtt{CW}(\mathbf{b})] \tag{S.33}$$

where

$$\mathcal{P}^{\mathrm{MC}}(b_j \neq b_k; \{\mathbf{l}_i\}) := \frac{\arccos(\mathbf{l}_j^{\mathsf{T}}\mathbf{l}_k)}{\pi}. \tag{S.34}$$

In the GW algorithm, the objective is rewritten as

$$\{\mathbf{l}_i^*\} := \underset{\{\mathbf{l}_i\}}{\operatorname{argmax}}\, \mathbb{E}_{\mathbf{b}\sim\mathcal{P}^{\mathrm{GW}}(\mathbf{b};\{\mathbf{l}_i\})}[\mathtt{CW}(\mathbf{b})] \tag{S.35}$$

where

$$\begin{aligned}\mathcal{P}^{\mathrm{GW}}(b_j \neq b_k; \{\mathbf{l}_i\}) &:= \frac{1-\mathbf{l}_j^{\mathsf{T}}\mathbf{l}_k}{2} \\ &\leq \frac{1}{0.878}\mathcal{P}^{\mathrm{MC}}(b_j \neq b_k; \{\mathbf{l}_i\}).\end{aligned} \tag{S.36}$$

The last inequality of Eq. (S.36) comes from [7, Lemma 3.4, Lemma 3.5]. [2] The right-hand side of Eq. (S.35) is an SDP, and hence it can be solved in polynomial time. The resulting $\mathbb{E}_{\mathbf{b}\sim\mathcal{P}^{\mathrm{MC}}(\mathbf{b};\{\mathbf{l}_i^*\})}[\mathtt{CW}(\mathbf{b})]$, which is the expected cut weight of random hyperplane cutting, is guaranteed to be greater than $0.878\,\mathbb{E}_{\mathbf{b}\sim\mathcal{P}^{\mathrm{MC}}(\mathbf{b};\{\mathbf{l}_i^{\mathrm{opt}}\})}[\mathtt{CW}(\mathbf{b})] = 0.878\max_{\mathbf{b}}\mathtt{CW}(\mathbf{b})$ because

$$\begin{aligned}\mathcal{P}^{\mathrm{MC}}(b_j \neq b_k; \{\mathbf{l}_i^*\}) &\geq 0.878\,\mathcal{P}^{\mathrm{GW}}(b_j \neq b_k; \{\mathbf{l}_i^*\}) \\ &\geq 0.878\,\mathcal{P}^{\mathrm{GW}}(b_j \neq b_k; \{\mathbf{l}_i^{\mathrm{opt}}\}) \\ &= 0.878\,\mathcal{P}^{\mathrm{MC}}(b_j \neq b_k; \{\mathbf{l}_i^{\mathrm{opt}}\}).\end{aligned} \tag{S.37}$$

Equation (S.35) is very similar to Eq. (17), but the rounding process of the GW is not the same as that of Eq. (18). Instead of carrying out $\mathbf{b}^{\mathrm{R}} = \operatorname{argmax}_{\mathbf{b}}\mathcal{P}^{\mathrm{GW}}(\mathbf{b};\{\mathbf{l}_i^*\})$ or $\mathbf{b}^{\mathrm{R}} = \operatorname{argmax}_{\mathbf{b}}\mathcal{P}^{\mathrm{MC}}(\mathbf{b};\{\mathbf{l}_i^*\})$, a number of $\mathbf{b}$ obeying $\mathcal{P}^{\mathrm{MC}}(\mathbf{b};\{\mathbf{l}_i^*\})$ are sampled by random hyperplane cutting and then, the best $\mathtt{CW}(\mathbf{b})$ is adopted as the candidate solution.

To solve the GW algorithm faster with less memory, various SDP solvers have been developed [30]. The runtime of the state-of-the-art SDP solvers are $\widetilde{O}(\sqrt{n}(mn^2 + m^\omega + n^\omega)\log(1/\epsilon)))$ [31] and $\widetilde{O}((\sqrt{n}(m^2+n^4) + m^\omega + n^{2\omega})\log(1/\epsilon)$ [32], where n is the problem size, m the number of constraints, ω the matrix multiplication constant ≤ 2.372927, and ϵ the relative accuracy. In the case of the MAX-CUT problem, the runtime is $\widetilde{O}(|V|^{3.5}\log(1/\epsilon))$ because $n = m = |V|$. The overall time complexity for the GW algorithm depends on the runtime of SDP because the naïve Cholesky decomposition takes $O(n^3) = O(|V|^3)$ to run, which is faster than the state-of-the-art SDP solver, as long as the number of trials of random hyperplane cutting is less than the time complexity of the SDP. Note that the computational complexity of SDP with respect to the number of nodes can be further reduced to $\widetilde{O}(|V|/\epsilon^{3.5})$ using a specific algorithm for the MAX-CUT , by taking a disadvantage with respect to the accuracy [33].

Although there is no approximation ratio guarantee nor computational complexity bound, there are several classical heuristics that are empirically faster and have higher approximation ratios than the GW algorithm. The rank-two relaxation algorithm [16] relaxed $\operatorname{rank}(\mathbf{X}) = 1$ to $\mathbf{X} = \mathbf{A}\mathbf{A}^{\mathsf{T}}$, where $\mathbf{A} = \begin{bmatrix}\mathbf{a}_1 & \dots & \mathbf{a}_{|V|}\end{bmatrix}^{\mathsf{T}}$, $\mathbf{a}_j = \begin{bmatrix}\cos\theta_j & \sin\theta_j\end{bmatrix}^{\mathsf{T}}$, and $\theta_j \in [0, 2\pi)$. Similar to the GW, the objective of the rank-two relaxation can be reformulated as

$$\{\theta_j^*\} := \underset{\{\theta_j\}}{\operatorname{argmax}}\, \mathbb{E}_{\mathcal{P}^{\mathrm{R2}}(\mathbf{b};\{\theta_j\})}[\mathtt{CW}(\mathbf{b})], \tag{S.38}$$

where

$$\mathcal{P}^{\mathrm{R2}}(b_j \neq b_k; \{\theta_j\}) = \sin^2\Big(\frac{\theta_j - \theta_k}{2}\Big). \tag{S.39}$$

As with the GW algorithm, the rank-two relaxation does not take the sample with the highest probability as a candidate. Moreover, because the hyperplane is only two-dimensional, random hyperplane cutting is not used. Instead, exhaustive search is used, expressed as

$$\mathbf{b}^{\mathrm{R2}} = \mathtt{CW}(\mathbf{b}^*(\alpha^*)), \tag{S.40}$$

[2] $\min_{x\in[-1,1]}\frac{\arccos(x)}{\pi}\left(\frac{1-x}{2}\right)^{-1} \approx 0.878$.

where

$$b_j^*(\alpha) = \begin{cases} 0 & \text{for} \quad \cos(\theta_j^* + \alpha) \geq 0 \\ 1 & \text{for} \quad \cos(\theta_j^* + \alpha) < 0 \end{cases} \tag{S.41}$$

and

$$\alpha^* = \underset{\alpha}{\operatorname{argmax}}\, \texttt{CW}(\mathbf{b}^*(\alpha)). \tag{S.42}$$

To optimize θ_j, a gradient-based optimizer is used. In addition, $\{1, 2\}$-local search and restarting optimization are incorporated. Here, k-local search is an exhaustive method that searches the entire solution within Hamming distance k of the current solution. Breakout local search [25] repeats the 1-local search and perturbation of the current solution using a tabu list. Once a local search falls into a local optimum, perturbation is added to the current solution, and then the local search is restarted. In the perturbation phase, one of three kinds of perturbation is applied with a specific probability. The tabu list is used to prohibit the perturbation of the listed nodes so that the same local minimum will not be visited again.

H.2 Quantum Algorithms for the MAX-CUT problem

The quantum algorithms for the MAX-CUT problem are categorized into two research streams. One is quantum approximate optimization algorithms (QAOAs) [8, 9] and the other is quantum algorithms for SDP [34–42]. Here, quantum annealing is excluded.

QAOA variants are well summarized in a recent review [10] and are not re-summarized here.

The quantum SDP algorithms are categorized into two types. One encodes the solution to a density matrix and the other uses QRAM [43] to store the solution.

For the density matrix encoding approach, there are two types of encoding methods. One uses a Gibbs state to represent a solution [34–37, 39], which can be regarded as a solution that is encoded in the corresponding Hamiltonian. The Hamiltonian used in the Gibbs state is updated by the matrix multiplicative weight update method [44] or the matrix exponentiated gradient update method [45]. The other density matrix encoding approach uses a parametrized quantum circuit to represent a solution [40–42]. The parameter is optimized by frameworks using the variational quantum algorithm [46, 47].

The state-of-the-art quantum SDP algorithm [48] is based on the primal-dual central path method, which uses quantum linear algebra and stores the solution in QRAM. The runtime of this quantum SDP algorithm is $O((mn^{1.5} + n^3)\mathrm{poly}(\kappa, \log(mn/\epsilon))$ where κ is the condition number of the matrices appearing in the algorithm, and ϵ is the accuracy parameter. In the case of the MAX-CUT problem, we have $\widetilde{O}(|V|^3\mathrm{poly}(\kappa, \log(|V|^2/\epsilon))$, which is a slight improvement on its classical counterpart, which is $\widetilde{O}(|V|^{3.5}\log(1/\epsilon))$.

Although these algorithms have the potential to solve an SDP faster than classical ones, the approximation ratio is the same as that of the GW algorithm, which is empirically worse than that of classical heuristics.

Very recently, [49] proposed the quantum-inspired algorithm based on the observation that ADAPT-QAOA [50] can be approximately expressed as the Clifford circuit, which can be efficiently simulatable using a classical computer. Their model showed the comparable cut weight performance to the GW with several thousand hyperplane cuttings.

H.3 QRACs

The concept that encodes m bits into n (qu)bits was originally proposed by Wiesner as *conjugate coding* [51] and later rediscovered by Ambainis et al. [1] as *(quantum) random access codes ((Q)RACs)*. (m, n, p) random access encoding is defined as the function that maps m classical bits into n (qu)bits with a decoding probability at least p [1, Definition 1.1]. The minimum number of n with $p > 1/2$ is bounded as $(1 - \eta(p))m$ for both classical [1, Theorem 2.1] and quantum [52, Theorem 2.3] settings, where $\eta(p)$ is the binary entropy function. In the case of $(2, 1, p)$- and $(3, 1, p)$-QRACs, there exists

$p = \frac{1}{2} + \frac{1}{2\sqrt{2}}$ [1, Lemma 3.1] and $p = \frac{1}{2} + \frac{1}{2\sqrt{3}}$ [1, attributed to Chuang], respectively, which are consistent with our results (Eq. (26)) in the case of $\mathcal{E}_j \in \{\pm 1\}$. It was proved that $(2^n, n, > 1/2)$ classical random access encoding [53, Theorem 5.5] and $(2^{2n}, n, > 1/2)$ quantum random access encoding [54, Theorem 2] do not exist. From these observations, $(3, 1)$-QRAC shows the maximum number of classical bits that can be embedded in one qubit with a decoding probability that is larger than $1/2$. This is why we only consider $m \in \{1, 2, 3\}$ in this study. In addition to $(m, 1, p)$-QRAC, $(m, 2, p)$-QRAC is particularly of interest and well studied in terms of the theoretical bound [54–57] and concrete construction scheme [58]. For arbitrary (m, n, p)-QRAC, the upper bound of decoding probability p is improved by considering shared randomness [59], shared entanglement [60], and d-level system [55, 61]. The information that can be decoded is not limited to the m bits that are encoded, but also extends to functions that use k-out-of-m bits [62, 63]. Using this extension, QRACs have also been applied in machine learning to efficiently encode discrete features [64].