MaxCut quantum approximate optimization algorithm performance guarantees for $p > 1$

Jonathan Wurtz[*] and Peter Love

Department of Physics and Astronomy, Tufts University, Medford, Massachusetts 02155, USA

We obtain worst-case performance guarantees for $p = 2$ and 3 QAOA for MaxCut on uniform 3-regular graphs. Previous work by Farhi *et al.* obtained a lower bound on the approximation ratio of 0.692 for $p = 1$. We find a lower bound of 0.7559 for $p = 2$, where worst-case graphs are those with no cycles $\leqslant 5$. This bound holds for any 3-regular graph evaluated at particular fixed parameters. We conjecture a hierarchy for all p, where worst-case graphs have with no cycles $\leqslant 2p + 1$. Under this conjecture, the approximation ratio is at least 0.7924 for all 3-regular graphs and $p = 3$. In addition, using an indistinguishable argument we find an upper bound on the worst-case approximation ratio for all p, which indicates classes of graphs for which there can be no quantum advantage for at least $p < 6$.

I. INTRODUCTION

In the rapidly developing field of quantum technology, near-term quantum devices [1] are the focus of much interest. Such noisy intermediate-scale quantum (NISQ) devices lack error correction and have imperfect gate implementations and environmental isolation, which restrict them to implementing only low-depth algorithms. Even with these constraints, can such a device display quantum advantage?

One algorithm suitable for NISQ devices is the quantum approximate optimization algorithm (QAOA), a hybrid quantum classical combinatorial optimization algorithm [2]. In QAOA, a classical computer optimizes $2p$ angles parametrizing an ansatz wave function by querying a near-term quantum device. This wave function encodes an approximate solution to some combinatorial optimization problem. For $p \to \infty$, it is known that the ansatz wave function encodes the exact solution, which follows from the adiabatic theorem [3]. For finite p, the picture is less clear. What p is needed to outperform the best classical algorithm? Asking such questions leads to competition between quantum and classical algorithms [4]. For example, a QAOA algorithm for E3LIN2 [5] with a quantum advantage was answered by an improved classical algorithm [6], which prompted an improved QAOA version without an advantage [7].

One can find worst-case performance guarantees for particular classes of problem instances in QAOA. Approximate solutions to a problem achieve some fraction C of the exact solution, called the approximation ratio. A worst-case performance guarantee bounds this approximation ratio from below. If the minimum approximation ratio $C_{\min}$ obtained from the quantum algorithm is larger than the value for the best classical algorithm, then the quantum algorithm has a quantum advantage, as it will produce better approximate answers for all instances. It is important to ask what this worst-case performance guarantee $C_{\min}$ is for QAOA as a function of p.

In this paper, we apply QAOA to the NP-hard graph partitioning problem of MaxCut [8], which partitions some graph into two sets by cutting a maximum number of edges. We will find that the worst-case performance guarantee for 3-regular graphs and $p = 2$ is $C_2 \geqslant 0.7559$, confirming the observation of Ref. [2] and improving on the original result for $p = 1$ of $C_1 \geqslant 0.692$, as expected. Under a conjecture that graphs with no "visible" cycles are worst case, we find $C_3 \geqslant 0.7924$ for $p = 3$. Additionally, we use an argument where fixed p QAOA cannot distinguish between large cycles of even and odd length to find an upper bound on expectation values, which upper bounds the minimum approximation ratio.

The paper is structured as follows. Section II reviews QAOA applied to the MaxCut problem. Sections III and IV detail how expectation values and the approximation ratio can be computed efficiently for any bounded degree graph for fixed values of p. Section V computes the worst-case performance guarantee for the $p = 1$ and 2 cases. Sections VI–IX discuss some of the implications of the worst-case performance, and Sec. X concludes with a discussion and interesting future directions.

II. THE MAXCUT PROBLEM AND QAOA

The MaxCut problem is defined as follows. Given a graph $\mathcal{G}$ with vertices V and edges E, the vertices are partitioned into two sets labeled by, say, $+$ or $-$. The goal is to find the partition of vertices such that a maximal number of edges have one vertex in each set. Restated, a solution to the MaxCut problem separates a graph $\mathcal{G}$ into two subgraphs by cutting the maximum number of edges.

This problem is encoded in qubits as follows. For each vertex, assign a qubit. Given vertices $\langle i \rangle$ and edges $\langle ij \rangle$ for a graph $\mathcal{G}$, the maximum cut is given by the maximal eigenstate of the objective function

$$\hat{C} = \sum_{\langle ij \rangle} \frac{1}{2} \left(1 - \hat{\sigma}_z^i \hat{\sigma}_z^j \right). \tag{1}$$

———————
[*]Corresponding author: jonathan.wurtz@tufts.edu

Each term is a clause representing an edge of the graph $\mathcal{G}$, with an eigenvalue of 1 if the edge is cut, and 0 if the edge is not. Because $\hat{C}$ is made of a sum of commuting Pauli $\hat{\sigma}_z$ terms, any eigenstate is a product state, and the maximal state can be simply read out in the Z basis. The partitioning of vertices is obtained by assigning each vertex according to a Z measurement outcome ± 1.

One method of computing an approximate maximal state of $\hat{C}$ is the quantum approximate optimization algorithm (QAOA) [2,9–14]. QAOA optimizes a variational wave function by maximizing the expectation value of the objective function with respect to a set of parameters $\{\gamma\}$, $\{\beta\}$,

$$F(\gamma, \beta) = \langle \gamma, \beta | \hat{C} | \gamma, \beta \rangle,$$
$$F_{\max} = \underset{\gamma, \beta}{\text{MAX}}: \quad F(\gamma, \beta). \tag{2}$$

The state preparation and evaluation of the expectation value $\langle \hat{C} \rangle$ can be done on a small quantum device, while the optimization of variational parameters $\{\gamma\}$, $\{\beta\}$ can be performed on a classical computer. The QAOA ansatz wave function $|\gamma, \beta\rangle$ is defined as [2]

$$|\gamma, \beta\rangle = e^{-i\beta_p \hat{B}} e^{-i\gamma_p \hat{C}} (\cdots) e^{-i\beta_1 \hat{B}} e^{-i\gamma_1 \hat{C}} |+\rangle, \tag{3}$$

where $\hat{B}$, the "mixing Hamiltonian," is defined as $\hat{B} = \sum_i \hat{\sigma}_x^i$ and $|+\rangle$ is the equally weighted superposition state or analogously the largest eigenstate of $\hat{B}$. Ellipses represent p iterations of unitarily evolving the wave function alternatively with generators $\hat{C}$ and $\hat{B}$. In the limit $p \to \infty$ the optimal state $|\gamma, \beta\rangle$ approaches the exact maximal state [2].

Given an approximate wave function with expectation value $\langle \hat{C} \rangle$, the state can be repeatedly measured m times in the Z basis to find a bit string whose expectation value evaluates to at least $\langle \hat{C} \rangle (1 - 1/m)$ as an approximate MaxCut solution. This is due to the phenomena of concentration, wherein the variance of expectation values is much smaller than the expectation value itself [12].

The approximation ratio for MaxCut is

$$C(\gamma, \beta) = \frac{F(\gamma, \beta)}{C_{\max}}, \tag{4}$$

where $C_{\max}$ is the maximum number of edges cut for an ideal partition of a particular graph $\mathcal{G}$. A number in between 0 and 1 measures how close the variational state is to the exact maximal state. A larger number indicates better performance, as bit strings from the measurement procedure will have a better MaxCut value. If $F_{\max} = C_{\max}$, then the variational state is the exact maximal state, and the approximation ratio is 1. Note that in practice, $C_{\max}$ may not be known, so bounding the approximation ratio from below requires $C_{\max}$ to be bounded from above.

III. FIXED-p ALGORITHM

It was found by Farhi *et al.* in 2014 [2] that for a fixed graph degree ν and particular value of p, the numerical difficulty of simulating QAOA evolution grows at most doubly exponentially in p, and linearly with the number of vertices N. In the interests of fixing notation and making the present paper self-contained let us begin by repeating the derivation of Ref. [2] here.

The expectation value $F(\gamma, \beta)$ is

$$F(\gamma, \beta) = \sum_{\langle ij \rangle} f_{\langle ij \rangle}(\gamma, \beta),$$

with

$$f_{\langle ij \rangle}(\gamma, \beta) = \tfrac{1}{2} \langle \gamma, \beta | 1 - \hat{\sigma}_z^i \hat{\sigma}_z^j | \gamma, \beta \rangle, \tag{5}$$

where the expectation value $F(\gamma, \beta)$ has been broken into a sum of terms $f_{\langle ij \rangle}(\gamma, \beta)$ corresponding to individual edges $\langle ij \rangle$. For a particular value of p, each value of $f_{\langle ij \rangle}(\gamma, \beta)$ may be computed as

$$\tfrac{1}{2} - \tfrac{1}{2} \langle +|(\cdots) e^{i\gamma_p \hat{C}} e^{i\beta_p \hat{B}} \hat{\sigma}_z^i \hat{\sigma}_z^j e^{-i\beta_p \hat{B}} e^{-i\gamma_p \hat{C}} (\cdots)|+\rangle, \tag{6}$$

where ellipses denote the action of the other $2p - 2$ generators.

In the Heisenberg picture, this expectation value can be computed for any value of N. The first generator $\hat{B}$ rotates each objective function clause as

$$\hat{\sigma}_z^i \hat{\sigma}_z^j \to \left[\cos(2\beta_p)\hat{\sigma}_z^i + \sin(2\beta_p)\hat{\sigma}_y^i \right]$$
$$\times \left[\cos(2\beta_p)\hat{\sigma}_z^j + \sin(2\beta_p)\hat{\sigma}_y^j \right], \tag{7}$$

keeping the Heisenberg rotated operator local to the span of the two sites i, j. Terms $\hat{\sigma}_z^k \hat{\sigma}_z^l$ in the second generator $\hat{C}$ commute and cancel unless the edges j, k overlap with i or j. In that case, the $\hat{\sigma}_y$ are rotated into $\hat{\sigma}_x$ and $\hat{\sigma}_z$ by terms $\hat{\sigma}_z^k \hat{\sigma}_z^i$, growing to a span supporting the three sites i, j, k for terms such as $\hat{\sigma}_z^k \hat{\sigma}_y^i \hat{\sigma}_y^j$. Repeating this one layer deeper can rotate Pauli operators on site k, and so forth. From this argument, it can be seen that after p steps, the operator will have a support over a subgraph with vertices at most p edges away from the initial vertices i, j.

Given a graph $\mathcal{G}$ and edge $\langle ij \rangle$, in order to compute a value $f_{\langle ij \rangle}(\gamma, \beta)$ one may truncate the graph to an induced subgraph only including vertices which are at most p edges away from either i or j, and the presence of the other vertices does not contribute to the expectation value. We denote such a subgraph of edge $\langle ij \rangle$ within graph $\mathcal{G}$ to depth p as $\mathcal{G}_{\langle ij \rangle}^p$. For a fixed p and graph degree ν, there are a finite number of unique subgraphs. For 3-regular graphs, there are exactly three edges incident on the vertex, and for $p = 1$ there are three subgraphs, with at most six vertices; for $p = 2$, there are 123 subgraphs with at most 14 vertices; for $p = 3$, there are 913 088 subgraphs with at most 30 vertices. See Appendix B for more details, and the table in the Supplemental Material [15] for enumerated subgraphs for $p = 1$ and 2. We will focus on 3-regular graphs, but these results generalize to other graphs with a small bounded degree.

As a technical note, because only vertices within p steps of the edge $\langle ij \rangle$ need be considered to compute $f_{\langle ij \rangle}(\gamma, \beta)$, such expectation values may be efficiently calculated in the Schrödinger picture. Because a $\hat{\sigma}_z \hat{\sigma}_z$ operator only spreads to a span over the subgraph, one need only apply unitary operators over the subgraph. If the wave function is evolved in the Schrödinger picture under these unitaries, the state on all other sites remains an unentangled $|+\rangle$ product state, and thus one can consider the wave function only acting on the reduced Hilbert space of the n vertices of the subgraph. This allows computation with the order 2^n values of the wave function on the subgraph, instead of the order 4^n values from a general
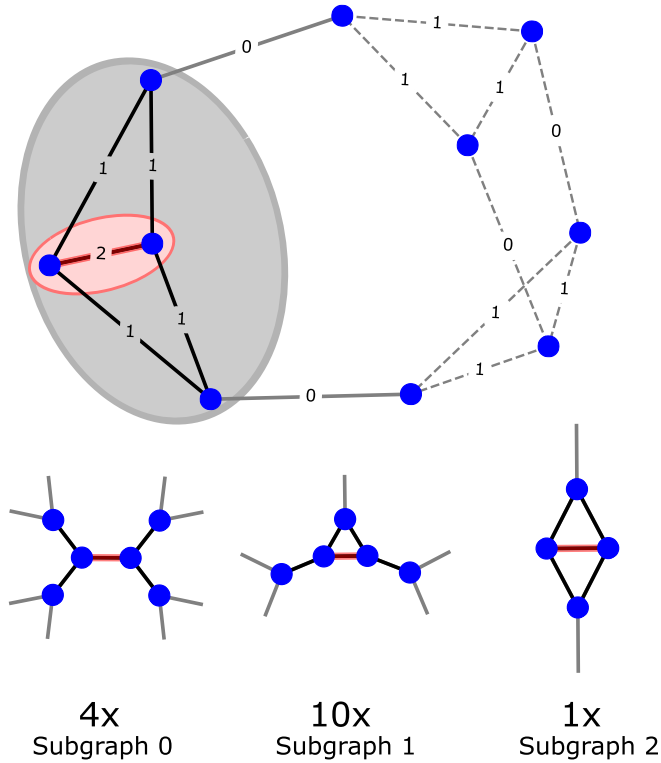
FIG. 1. An example graph with subgraph assignments for $p = 1$. Each edge (red) exists in a subgraph of edges and vertices a distance $p \leqslant 1$ away (gray circle). Edges within one step of the red edge (black) uniquely define the subgraph assignment of the edge. For this graph, there are four instances of subgraph 0 ("the tree"), ten instances of subgraph 1 ("single triangle"), and one instance of subgraph 2 ("two triangles"). The subgraph of each edge is labeled on the edge. A more detailed visual of this decomposition is shown in Fig. 2.

operator acting on the subgraph in the Heisenberg picture, or order 2^N values of the wave function on the entire graph.

The procedure for computing the expectation value $F(\gamma, \beta)$ for a particular graph $\mathcal{G}$ of bounded degree ν and fixed p is as follows. For each edge $\langle ij \rangle$, identify the subgraph $\mathcal{G}^p_{\langle ij \rangle}$ of all edges and vertices within p steps of i and j (see Fig. 1). This defines a collection of subgraphs $\{\mathcal{G}^p_{\langle ij \rangle} | \langle ij \rangle \in \mathcal{G}\}$, one for each edge, for which each $f_{\langle ij \rangle}(\gamma, \beta)$ can be computed in parallel. This collection of subgraphs can be further decomposed by counting the number $N_\lambda(\mathcal{G})$ of each kind of subgraph λ, $\mathcal{S}_\lambda \in \{\mathcal{S}\}$ in the collection of all subgraphs of depth p, with each edge of the graph given a particular subgraph assignment $\langle ij \rangle \to \lambda$. The expectation value is then

$$F(\gamma, \beta) = \sum_{\text{subgraphs } \lambda} N_\lambda(\mathcal{G}) f_\lambda(\gamma, \beta), \qquad (8)$$

where $f_\lambda(\gamma, \beta)$ is the expectation value of the center edge of the λth subgraph.

IV. LOWER BOUNDS ON THE APPROXIMATION RATIO

A performance guarantee for QAOA can be obtained by computing a lower bound of the approximation ratio $C_p(\mathcal{G}) = F_M/C_M$ of any graph, then finding the graph(s) with the lowest

lower bound. Such a lower bound is given by the ratio of a lower bound on the maximum expectation value F_M, and an upper bound on the best MaxCut value C_M.

A. Lower bound on the maximum expectation value F_M

The value F_M is bounded from below by

$$F_M = \underset{\gamma, \beta}{\text{MAX}}: \quad F(\gamma, \beta) \geqslant \sum_\lambda N_\lambda(\mathcal{G}) f_\lambda, \qquad (9)$$

where $f_\lambda \equiv f_\lambda(\gamma, \beta)$ is the expectation value of the center edge of the particular subgraph λ, chosen for a particular set of values (γ, β).

The sum on the right-hand side of Eq. (9) is guaranteed to be less than or equal to the global maximum F_{max}, which simultaneously optimizes (γ, β) for all clauses. We may choose the set of values for which to compute f_λ; we use the following set of angles,

$$p = 1: \quad \{\gamma_1, \beta_1\} = \{35.3°, 22.5°\},$$
$$p = 2: \quad \{\gamma_1, \beta_1, \gamma_2, \beta_2\} = \{28.0°, 31.8°, 51.4°, 16.8°\}.$$
$$(10)$$

These values are one of the optima for the tree subgraph, which does not have any cycles (see Fig. 1, bottom left). For more details on this choice of angles, see Sec. VI.

B. Upper bound on the MaxCut value C_M

It is hard to find the exact MaxCut value C_M, which is after all one of the objectives of the QAOA algorithm. Fortunately, it suffices to find an upper bound of C_M to yield a lower bound on the approximation ratio. Equivalently, we may find a *lower* bound on the number of *uncut* edges in a partition of a graph, $R_M(\mathcal{G})$, to find the upper bound on the number of cut edges $C_M = N(\mathcal{G}) - R_M(\mathcal{G})$. While these two views are equivalent, we find it more convenient to count uncut edges.

The lower bound on the number of uncut edges can be found by considering only the local structure of a graph. Locally, there is some amount of "visible" frustration, which forces a minimum number of edges to remain uncut. For example, a triangle of three vertices requires at least one edge uncut. Additional edges may remain uncut due to a global structure which is not "visible" locally. We will use the local structure of a graph to get the tightest possible lower bound on the number of uncut edges. While other simpler methods such as semidefinite program (SDP) relaxations [16] find upper bounds on the MaxCut value, they do not use a local structure, as will be necessary for the proof. This locally computed MaxCut bound generalizes that of Ref. [2] beyond $p = 1$. The bound will be done in three steps, each of which tighten the bound.

As a first step, a trivial underestimate of the number of uncut edges is that no edges remain uncut in the graph, and so the MaxCut value is bounded from above by the number of edges in the graph. This bound is not tight, and does not take into account any of the structure of the graph. This trivial bound can be tightened by considering the local structure.

As a next step, consider for a graph $\mathcal{G}$, a graph $\mathcal{H}_p$ which is a collection of disconnected subgraphs $\mathcal{G}^p_{\langle ij \rangle}$, one for each edge $\langle ij \rangle$ in $\mathcal{G}$. An example of this decomposition is shown in
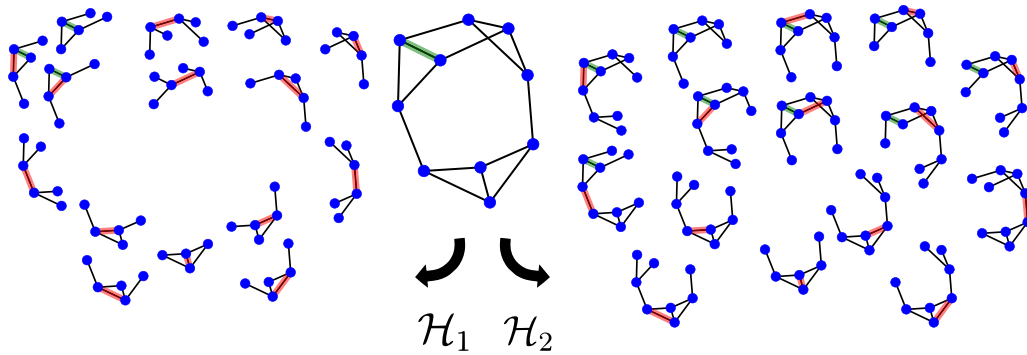
FIG. 2. A lower bound on the approximation ratio can be found by a decomposition into subgraphs. Any graph $\mathcal{G}$ (center) can be decomposed into the subgraph graph $\mathcal{H}_p$. A lower bound on the expectation value is given by a sum over all subgraphs [Eq. (8)] where the center edge is in red. An upper bound on the MaxCut value is found by a careful counting of uncut edges. A particular edge (green) appears five times in $\mathcal{H}_1$ (left) and nine times in $\mathcal{H}_2$ (right). A subgraph contributes to the sum (15) if the green edge participates in an odd length cycle of a subgraph. For this particular graph environment, three of five subgraphs contribute for $p = 1$ and all nine subgraphs contribute for $p = 2$; Eq. (15) is 3/5 for $p = 1$ and 452/495 for $p = 2$.

Fig. 2. Heuristically, the graph $\mathcal{H}_p$ "sees" the local structure of $\mathcal{G}$ out to a distance p.

Each edge $\langle ij \rangle$ in $\mathcal{G}$ appears in multiple subgraphs of $\mathcal{G}$ and therefore of $\mathcal{H}_p$. It appears in the subgraph $\mathcal{G}_{\langle ij \rangle}^p$ as the center edge, but it also appears in all subgraphs $\mathcal{G}_{\langle kl \rangle}^p$ whose center edge $\langle kl \rangle$ is $\leqslant p$ steps from edge $\langle ij \rangle$. Because distance is symmetric, the number of edges in subgraph $\mathcal{G}_{\langle ij \rangle}^p$ is equal to the number of subgraphs in which $\langle ij \rangle$ appears.

The subgraphs in which edge $\langle ij \rangle$ appears are thus identified by all edges within a distance $2p$, because distance is symmetric. The first p steps identify the subgraph assignment of the center edge, while the second p steps identify the subgraph assignment of the edges $\langle kl \rangle$ within a radius p of $\langle ij \rangle$ containing edge $\langle ij \rangle$.
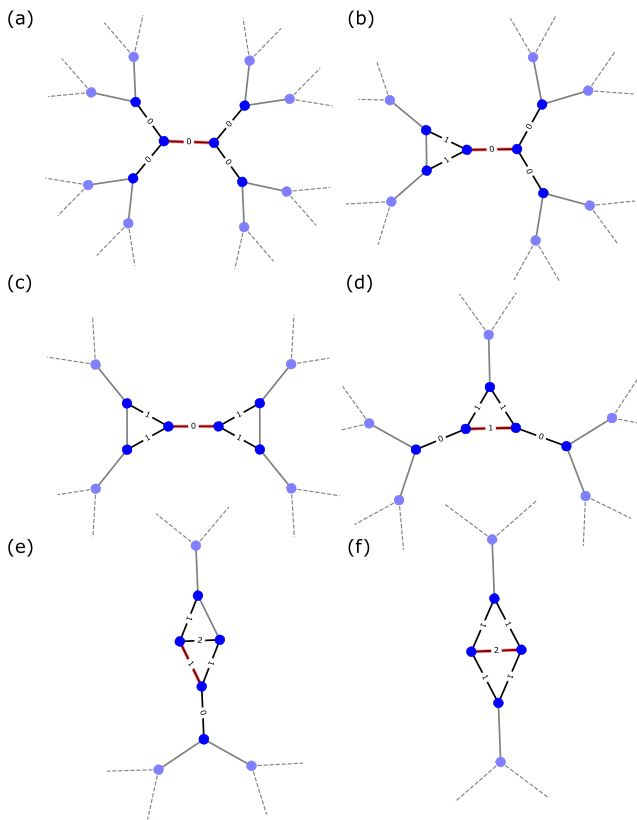

FIG. 3. The six $p = 1$ graph environments, which fix edges to particular subgraphs, as shown by edge labels. Black edges are of the center subgraph, while gray edges are the choice of subgraph environment, and red is the special center edge. These graph environments mean that certain subgraphs cannot appear in isolation, and there are adjacency restrictions for certain subgraphs.
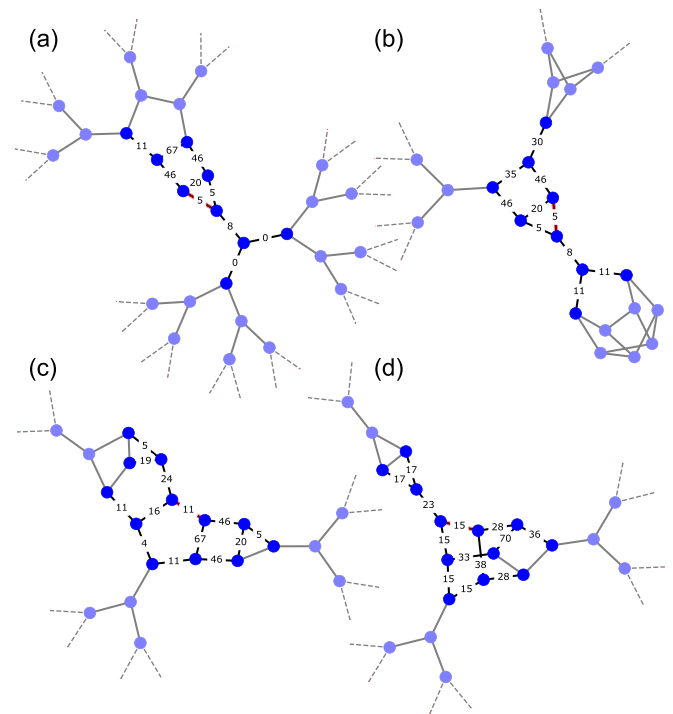

FIG. 4. Some example graph environments for $p = 2$ graphs. Subgraph edges fixed by the environment are shown by edge labels. (a) and (b) show two example environments for subgraph 5, while (c) and (d) show some example environments for subgraphs 11 and 15.

We call the surroundings $\mathcal{G}_{\langle ij \rangle}^{2p}$ which fix the subgraph assignment of adjacent edges to depth p the *graph environment* of edge $\langle ij \rangle$. The set of all possible graph environments of depth p is the set of all unique combinations of subgraph assignments on the edges of all subgraphs of depth p. It is also all possible combinations of subgraphs that an edge $\langle ij \rangle$ can appear in. This set of graph environments allows for a search through all possible local graph structures, without needing to be concerned with the global structure of the arbitrarily larger graph.

The six subgraph environments for $p = 1$ are shown in Fig. 3. There are three combinations of subgraph assignments for subgraph 0, "the tree" [Figs. 3(a)–3(c)], two combinations of subgraph assignments for subgraph 1, "single triangle" [Figs. 3(d) and 3(e)], and one combination for subgraph 2, "two triangles" [Fig. 3(f)].

These graph environments restrict which subgraph assignments are allowed to be adjacent. For instance, consider subgraph 2, "two triangles." There is only one graph environment in which it is the center edge, and the adjacent edges must be assigned to subgraph 1, "one triangle." Thus, every instance of subgraph 2 in a graph must have at least four instances of subgraph 1. Equivalently, the set of graph environments are the set of all possible combinations of subgraphs which an edge can appear in. Some example graph environments for $p = 2$ are shown in Fig. 4.

The set of graph environments is a subset of all subgraphs of depth $2p$. It is a subset because subgraphs of depth $2p$ may have cycles of length more than $2p + 1$, which the subgraph assignments of depth p cannot distinguish. Because of this fact, the set of all graph environments is equivalent to the set of subgraphs of depth $2p$, subject to the constraint of no cycles greater than $2p + 1$ in the minimum cycle basis [17].

Now, consider any partition P of vertices of the graph $\mathcal{G}$, including the MaxCut partition M. The partition leaves some edges $\langle ij \rangle$ uncut, denoted by the characteristic function r_P,

$$r_{P\langle ij \rangle} = \begin{cases} 1, & \text{if edge } \langle ij \rangle \text{ is uncut in the partition } P \text{ of } \mathcal{G}, \\ 0, & \text{otherwise.} \end{cases}$$

The number of uncut edges in $\mathcal{G}$ given partition P is the sum over edges $R_P(\mathcal{G}) = \sum_{\langle ij \rangle} r_{P\langle ij \rangle}$.

Similarly, consider any partition $\mathcal{P}$ of vertices of the graph $\mathcal{H}_p$, including its MaxCut partition $\mathcal{M}$. An edge of $\mathcal{H}_p$ is labeled by the index of its parent edge $\langle ij \rangle$ in G, as well as the center edge $\langle kl \rangle$ of the subgraph it participates in. The partition $\mathcal{P}$ leaves some edges $\langle ij \rangle$ of the subgraph $\mathcal{G}_{\langle kl \rangle}^p$ uncut. These cuts are labeled by the characteristic function $r_{\mathcal{P}}$,

$$r_{\mathcal{P}\langle ij \rangle}^{\langle kl \rangle} = \begin{cases} 1, & \text{if edge } \langle ij \rangle \text{ is uncut in the partition } \\ & \mathcal{P} \text{ of subgraph } \mathcal{G}_{\langle kl \rangle}^p, \\ 0, & \text{if edge } \langle ij \rangle \text{ is cut in the partition } \\ & \mathcal{P} \text{ of subgraph } \mathcal{G}_{\langle kl \rangle}^p \text{ or if } \langle ij \rangle \notin \mathcal{G}_{\langle kl \rangle}^p. \end{cases}$$

The number of uncut edges in $\mathcal{H}_p$ given partition $\mathcal{P}$ is the sum over edges,

$$R_{\mathcal{P}}(\mathcal{H}_p) = \sum_{\langle ij \rangle} \sum_{\langle kl \rangle} r_{\mathcal{P}\langle ij \rangle}^{\langle kl \rangle}.$$

The MaxCut M of $\mathcal{G}$ induces a partition of $\mathcal{H}_p$, $M \mapsto \mathcal{I}$, in which each vertex of $\mathcal{H}_p$ is assigned to the same set as its parent vertex in $\mathcal{G}$. This implies that all copies of cut edges in M of $\mathcal{G}$ are cut in $\mathcal{H}_p$, e.g., $r_{\mathcal{I}\langle ij \rangle}^{\langle kl \rangle} = r_{\mathcal{M}\langle ij \rangle} \; \forall \langle kl \rangle \in \mathcal{G}$.

The set of cuts of $\mathcal{H}_p$ is a superset of the cuts of $\mathcal{G}$ because the vertices of $\mathcal{G}$ appear multiple times in $\mathcal{H}_p$. The number of uncut edges in the MaxCut of $\mathcal{H}_p$ is therefore bounded above by the number of uncut edges in the partition $\mathcal{I}$ induced by the MaxCut of $\mathcal{G}$: $R_{\mathcal{M}}(\mathcal{H}_p) \leqslant R_{\mathcal{I}}(\mathcal{H}_p)$.

Additionally, each edge $\langle ij \rangle$ of $\mathcal{G}$ appears $N(\mathcal{G}_{\langle ij \rangle}^p)$ times in $\mathcal{H}_p$. This number is bounded from above by the largest possible subgraph $N(\mathcal{S}_{\lambda'}^p)$. For $p = 1$, every subgraph has exactly five edges. For $p = 2$, the largest subgraphs have 13 edges.

Using the fact that the number of times each edge $\langle ij \rangle$ appears in $\mathcal{H}_p$ is bounded, and solutions on $\mathcal{H}_p$ can be induced from $\mathcal{G}$, we may bound the number of uncut edges in the MaxCut of $\mathcal{G}$ from below. For $p = 1$ we may write the following inequalities,

$$
\begin{aligned}
R_{\mathcal{M}}(\mathcal{H}_1) \leqslant R_{\mathcal{I}}(\mathcal{H}_1) &= \sum_{\langle ij \rangle, \langle kl \rangle} r_{\mathcal{I}\langle ij \rangle}^{\langle kl \rangle} \\
&= \sum_{\langle ij \rangle, \langle kl \rangle} r_{M\langle ij \rangle} = 5 \sum_{\langle ij \rangle} r_{M\langle ij \rangle} \\
&\Rightarrow \frac{1}{5} R_{\mathcal{M}}(\mathcal{H}_1) \leqslant R_M(\mathcal{G}).
\end{aligned}
\tag{11}
$$

Similarly, for $p = 2$,

$$
\begin{aligned}
R_{\mathcal{M}}(\mathcal{H}_2) \leqslant R_{\mathcal{I}}(\mathcal{H}_2) &= \sum_{\langle ij \rangle, \langle kl \rangle} r_{\mathcal{I}\langle ij \rangle}^{\langle kl \rangle} \\
&= \sum_{\langle ij \rangle, \langle kl \rangle} r_{M\langle ij \rangle} \leqslant 13 \sum_{\langle ij \rangle} r_{M\langle ij \rangle} \\
&\Rightarrow \frac{1}{13} R_{\mathcal{M}}(\mathcal{H}_2) \leqslant R_M(\mathcal{G}).
\end{aligned}
\tag{12}
$$

Because $\mathcal{H}_p$ is separated into disconnected subgraphs, it is simple to find the MaxCut partition of $\mathcal{H}_p$ for any size graph. This method of computing upper bounds on the MaxCut value reproduces that of Ref. [2]. The bound on the MaxCut used in Ref. [2] counted the number T of isolated triangles ("single triangle" subgraphs) and S of crossed squares ("two triangle" subgraphs). Each isolated triangle and crossed square will have one uncut edge, so a 3-regular graph with n vertices and $3n/2$ edges has at least $S + T$ uncut edges and thus at most $3n/2 - S - T$ cut edges. Similarly, the number of $p = 1$ subgraphs present in $\mathcal{H}_1$ are then functions of S and T. Specifically, there are S subgraphs of type 2 ("two triangle"), and $4S + 3T$ subgraphs of type 1 ("single triangle"). Each of these subgraphs has one uncut edge, and so Eq. (11) upper bounds the MaxCut of $\mathcal{G}$ to $3n/2 - (S + 4S + 3T)/5$. This bound is looser than that obtained from directly counting S and T in $\mathcal{G}$ because H_1 mistakes some of the crossed squares for isolated triangles. For $p = 2$ the number of edges per subgraph in H_2 varies and the use of Eq. (12) will give looser bounds.

The third and final step for finding a tighter bound for $p > 1$ can be found by considering the local structure of subgraphs $G_{\langle kl \rangle}^p$. This requires the following fact: If an edge of some graph $\mathcal{G}$ participates in no odd length cycles, it must be cut in the MaxCut solution. This is a consequence of balance in signed graphs [18]. This condition is labeled by
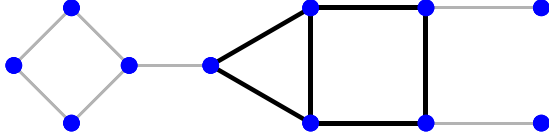
FIG. 5. An example of the function $\delta^{\langle kl \rangle}_{\langle ij \rangle}$ on an example graph. Here, $\delta = 1$ for dark edges, which participate in odd cycles of length 3 or 5. Similarly, $\delta = 0$ for light edges which do not participate in odd length cycles. This can occur even if the graph has odd length cycles, because parts of the graph are connected by only a single edge.

the characteristic function,

$$
\delta^{\langle kl \rangle}_{\langle ij \rangle} = \begin{cases} 1, & \text{if edge } \langle ij \rangle \text{ in subgraph } \mathcal{G}^p_{\langle kl \rangle} \\ & \text{participates in at least one odd length cycle,} \\ 0, & \text{otherwise.} \end{cases}
$$

An example of this function on a graph is shown in Fig. 5.

For large connected graphs, $\delta = 1$ for almost every edge, suggesting this function is not very interesting for graphs with few "loose edges," where a loose edge is defined as an edge connecting a vertex of degree one to the graph. However, the subgraphs $G^p_{\langle kl \rangle}$ have many loose edges (see the table in the Supplemental Material [15]). Given some subgraph $G^p_{\langle kl \rangle}$ with MaxCut solution $\mathcal{M}$, this implies $r^{\langle kl \rangle}_{\mathcal{M}\langle ij \rangle} = r^{\langle kl \rangle}_{\mathcal{M}\langle ij \rangle} \delta^{\langle kl \rangle}_{\langle ij \rangle}$.

With this fact, let us follow the same procedure as above, inducing a solution $\mathcal{I}$ on subgraph $G^p_{\langle kl \rangle}$ from the MaxCut solution M on graph $\mathcal{G}$, with $r^{\langle kl \rangle}_{\mathcal{I}\langle ij \rangle} = r_{M\langle ij \rangle}$. It is simple to see that there exists a different partition $\mathcal{I}'$ for which

$$
r_{\mathcal{I}'}{}^{\langle kl \rangle}_{\langle ij \rangle} = r_{\mathcal{I}}{}^{\langle kl \rangle}_{\langle ij \rangle} \quad \text{when} \quad \delta^{\langle kl \rangle}_{\langle ij \rangle} = 1,
$$

$$
r_{\mathcal{I}'}{}^{\langle kl \rangle}_{\langle ij \rangle} = 0 \quad \text{when} \quad \delta^{\langle kl \rangle}_{\langle ij \rangle} = 0.
$$

The partition $\mathcal{I}'$ exists based on the fact that $\delta = 0$ only on "loose edges" of a subgraph that are only connected by a single edge, and thus the loose edges can be solved independently from the rest of the subgraph.

These facts lead to a chain of inequalities, where the sum is partitioned into edges which participate in odd length cycles, and those that do not,

$$
\begin{aligned}
R_{\mathcal{I}}\left(\mathcal{G}^p_{\langle kl \rangle}\right) &= \sum_{\langle ij \rangle} \left[r_{I}{}^{\langle kl \rangle}_{\langle ij \rangle} \delta^{\langle kl \rangle}_{\langle ij \rangle} + r_{I}{}^{\langle kl \rangle}_{\langle ij \rangle} \left(1 - \delta^{\langle kl \rangle}_{\langle ij \rangle}\right) \right] \\
&\geqslant R_{\mathcal{I}'}\left(\mathcal{G}^p_{\langle kl \rangle}\right) = \sum_{\langle ij \rangle} \left[r_{I'}{}^{\langle kl \rangle}_{\langle ij \rangle} \delta^{\langle kl \rangle}_{\langle ij \rangle} + r_{I'}{}^{\langle kl \rangle}_{\langle ij \rangle} \left(1 - \delta^{\langle kl \rangle}_{\langle ij \rangle}\right) \right] \\
&= \sum_{\langle ij \rangle} r_{M\langle ij \rangle} \delta^{\langle kl \rangle}_{\langle ij \rangle} \\
&\geqslant \sum_{\langle ij \rangle} r_{\mathcal{M}}{}^{\langle kl \rangle}_{\langle ij \rangle} = R_{\mathcal{M}}\left(\mathcal{G}^p_{\langle kl \rangle}\right).
\end{aligned} \tag{13}
$$

The first step is from the partition $\mathcal{M}'$ having a larger MaxCut than $\mathcal{M}$; the second step is from properties and definitions of the partition $\mathcal{M}'$ as induced from $\mathcal{M}$ as induced from M. The third step is from the MaxCut partition $\mathcal{M}$ being larger than the partition $\mathcal{M}'$.
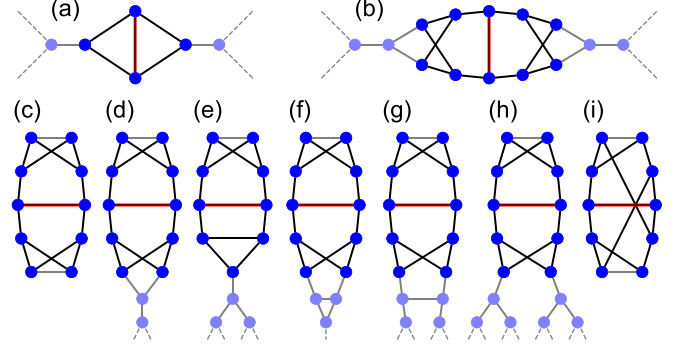

FIG. 6. The worst-case graph environments which bound the value of μ_p for (a) $p = 1$ and (b) $p = 2$, found by enumeration of all possible graph environments. These particular graph environments count the contribution of the central (red) uncut edge the maximal amount. There are seven graph environments which have a larger sum than (b), shown as (c)–(i), which are excluded due to additional uncut edges.

Next consider the following inequality,

$$
\sum_{\langle kl \rangle} \frac{\sum_{\langle ij \rangle} r_{M}{}^{\langle kl \rangle}_{\langle ij \rangle}}{N\left(\mathcal{G}^p_{\langle kl \rangle}\right)} \leqslant \sum_{\langle ij \rangle} r_{M\langle ij \rangle} \left(\sum_{\langle kl \rangle} \frac{\delta^{\langle kl \rangle}_{\langle ij \rangle}}{N\left(\mathcal{G}^p_{\langle kl \rangle}\right)} \right), \tag{14}
$$

which follows from Eq. (13). It gives a lower bound on a weighted sum over edges present in the MaxCut of G. The maximum coefficient in the parentheses is

$$
\text{MAX} : \sum_{\langle kl \rangle} \frac{\delta^{\langle kl \rangle}_{\langle ij \rangle}}{N\left(\mathcal{G}^p_{\langle kl \rangle}\right)} = \frac{1}{\mu_p} \tag{15}
$$

for some yet undetermined factor μ_p. This term gives the contribution from the worst-case graph environment and implies

$$
\sum_{\langle kl \rangle} \frac{\sum_{\langle ij \rangle} r_{M}{}^{\langle kl \rangle}_{\langle ij \rangle}}{N\left(\mathcal{G}^p_{\langle kl \rangle}\right)} \leqslant \sum_{\langle ij \rangle} \sum_{\langle kl \rangle} \frac{r_{M\langle ij \rangle} \delta^{\langle kl \rangle}_{\langle ij \rangle}}{N\left(\mathcal{G}^p_{\langle kl \rangle}\right)} \leqslant \frac{1}{\mu_p} \sum_{\langle ij \rangle} r_{M\langle ij \rangle}. \tag{16}
$$

It thus suffices to search through every possible graph environment and thus every possible combination of subgraphs an edge can participate in to find the worst-case graph environment which gives $1/\mu_p$. Equation (15) ultimately bounds how much each edge is counted in the sum over subgraphs.

For $p = 1$ there are six possible graph environments (shown in Fig. 3). The worst-case graph environment is Fig. 6(a), which is the central edge of two triangles. In this graph environment, the edge $\langle ij \rangle$ participates in an odd length (triangle) cycle in five subgraphs where each have five edges, and so Eq. (15) sums to 1. Thus, $\mu_1 = 1$.

For $p = 2$ there are a large number of graph environments. As the $p = 1$ case, the value μ_2 can be found by searching through every possible combination of subgraphs an edge can participate in, e.g., all graph environments, and finding the largest value of the sum. This search can be simplified by avoiding trivial instances of graph environments. If the edge $\langle ij \rangle$ does not participate in an odd length cycle in the subgraph where it is the center edge, it will not participate in an odd length cycle in any other subgraph of the graph environment. This excludes all subgraphs for which the center

edge is the only connection between two sides, such as the tree graph, and so reduces the number of graph environments to a manageable amount. Under this exclusion, there are 1 002 191 graph environments to search for $p = 2$.

Evaluating the sum from Eq. (15) for each graph environment finds 117 for which $\mu_2 < 1$. The worst-case graph environment is shown in Fig. 6(c). The central edge participates in an odd length cycle for all 13 of the subgraphs it appears in; one subgraph has 13 edges, four have 12 edges, and eight have nine edges, and so Eq. (15) gives μ_2 as

$$\frac{1}{13} + \frac{4}{12} + \frac{8}{9} = \frac{152}{117} = \frac{1}{\mu_2}. \qquad (17)$$

However, for this graph environment and the six following as ordered by Eq. (15) [Figs. 6(c)–6(i)] we can evaluate the weighted sum on the right-hand side of (14) directly. We find that these graphs have at least one additional uncut edge in the MaxCut partition. For example, for the subgraph environment shown in Fig. 6(c), the center edge plus the two additional uncut edges give the weighted sum on the right-hand side of (14) as $152/117 + 95/126 + 95/126 < 3$, and similar for Figs. 6(d)–6(i). Thus, these graph environments may be excluded to get a tighter bound on $R(\mathcal{G})$.

The eighth graph environment [Fig. 6(b)] has only one uncut edge in a MaxCut solution and so sets μ_2. The central edge participates in an odd length cycle for all 13 of the subgraphs it appears in; five of these subgraphs have 13 edges, and eight have 11 edges. Using Eq. (15), this worst-case graph environment bounds μ_2 to be $\mu_2 = 143/159$. Equation (16) thus simplifies to a lower bound on the number of uncut edges in a graph $\mathcal{G}$, using the fact that $R(\mathcal{G}_{\langle kl\rangle}^p) = \sum_{\langle ij\rangle} r_{\langle ij\rangle}^{\langle kl\rangle}$ and similar for $r_{\langle ij\rangle}$,

$$\frac{143}{159} \sum_{\langle kl\rangle} \frac{R(\mathcal{G}_{\langle kl\rangle}^2)}{N(\mathcal{G}_{\langle kl\rangle}^2)} \leqslant R_M(\mathcal{G}). \qquad (18)$$

The constant μ_p serves as a guarantee that uncut edges in $\mathcal{G}$ are not overcounted in the sum over subgraphs $\mathcal{G}_{\langle kl\rangle}^p$. Equivalently, the number of cut edges is a sum over subgraphs,

$$C_M \leqslant \sum_\lambda N_\lambda(\mathcal{G})c_\lambda, \quad c_\lambda = 1 - \frac{143}{159}\frac{R(S_\lambda)}{N(S_\lambda)}, \qquad (19)$$

where c_λ is the local MaxCut fraction for subgraph λ. Values for c_λ are enumerated in the table of subgraphs in the Supplemental Material [15]. This combinatoric search emphasizes a thesis of the paper: There could be some unexpected graph structure (as here) which in fact is a worst-case graph that cannot be generated from simple intuition. Thus, any proof of performance guarantees must be combinatoric in nature.

C. Bounding the approximation ratio as a fraction of sums

The numerator F_M is a lower bound and the denominator C_M is an upper bound and so a fraction of sums serves as a lower bound on the approximation ratio,

$$C_p(\mathcal{G}) \geqslant \frac{\sum_\lambda N_\lambda(\mathcal{G})f_\lambda}{\sum_\lambda N_\lambda(\mathcal{G})c_\lambda}. \qquad (20)$$

Values for f_λ and c_λ for the enumerated subgraphs of $p = 1, 2$ and fixed degree $\nu = 3$ are shown in the table in the Supple-

mental Material [15], and details of the computation of f_λ are shown in Appendix A.

As an example, consider the graph shown in Fig. 1. Each edge is labeled by the index of the unique subgraph identified from $\mathcal{G}_{\langle ij\rangle}^p$. For $p = 1$, there are four instances of subgraph 0 ("the tree"), ten instances of subgraph 1 ("single triangle"), and one instance of subgraph 2 ("two triangles"). For $p = 2$ there are six different kinds of subgraphs. Equation (20) lower bounding the approximation ratio for $p = 1$ becomes

$$C_1(\mathcal{G}) \geqslant \frac{4f_0 + 10f_1 + f_2}{4c_0 + 10c_1 + c_2}. \qquad (21)$$

Using the table of subgraphs in the Supplemental Material [15], one can look up the expectation values and local MaxCut values. The upper bound on the MaxCut value is $12.8 \geqslant 12$, the exact value, and the approximation ratio for this particular graph will be at least $C_1 \geqslant 0.759$ and $C_2 \geqslant 0.808$ for $p = 1$ and 2, respectively.

In this way, a lower bound on the approximation ratio of any graph can be found. This lower bound is rather pessimistic, as it chooses seemingly arbitrary angles (γ, β). However, the particular choice of angles given by Eq. (10) appears to still have large expectation values for all subgraphs. This fact will be discussed later.

V. WORST CASE FOR 3-REGULAR GRAPHS

What is the worst-case approximation ratio for a particular fixed value of p and given set of graphs $\{\mathcal{G}\}$? There exists some graph $\mathcal{G}_* \in \{\mathcal{G}\}$ which can be chosen maliciously such that the maximal approximation ratio $C(\mathcal{G}_*)$ is minimal in $\{\mathcal{G}\}$. This graph $\mathcal{G}_*$ represents a problem instance for which a QAOA device with fixed p has the worst performance. Any other graph will have a larger approximation ratio and thus this worst case is a performance guarantee on QAOA.

Naively, finding such a graph $\mathcal{G}_*$ is hard. The number of possible graphs is exponential in the number of vertices and we are interested in the general performance for arbitrarily large graphs, so a simple search through many graphs will not work. Because of this, a more careful approach must be taken to find lower bounds on the approximation ratio. Two methods are presented below.

The first method obtains a lower bound by finding worst-case combinations of subgraphs which may or may not form a consistent graph. By considering more and more complicated combinations of subgraphs, one can get a tighter bound from below on the minimum approximation ratio. This is the approach used for the original $p = 1$ bound by Farhi *et al.* [2].

The second method presents a graph hierarchy which finds that the class of graphs with no cycles less than 4 (for $p = 1$) or 6 (for $p = 2$) are worst case. This is done by finding that, given a graph $\mathcal{G}$, there always exists a new graph $\mathcal{G}'$ with a smaller or equal approximation ratio. This is done by replacing edges with a subgraph to reduce the number of small cycles in the graph.

A. Lower bounds for $p = 1$

Instead of finding the exact approximation ratio of the worst-case graph, one can instead *bound* the worst-case

approximation ratio from below, by only including subgraphs with a small approximation ratio. This is an extension of the original analysis of Farhi *et al.* [2]. By Eq. (20) a lower bound can be found by decomposing a graph $\mathcal{G}$ into subgraphs of a particular set $\{\mathcal{S}_\lambda\}$. Consider the inequality

$$\frac{F}{C} \leqslant \frac{f'}{c'} \Leftrightarrow \frac{F}{C} \leqslant \frac{F+f'}{C+c'} \leqslant \frac{f'}{c'} \qquad (22)$$

for all $f, c, F, C > 0$. In the context of Eq. (20), the terms are expectation values F, f and local MaxCut values C, c of two sets of subgraphs. One can order all subgraphs S_λ by their own local approximation ratio $C_\lambda = f_\lambda/c_\lambda$ and constructively add subgraphs, starting with the subgraph with the smallest local approximation ratio.

By Eq. (22), including only the worst subgraph of a graph, or excluding the best subgraph, gives a lower bound on the global approximation ratio. Including any other subgraph with a larger local approximation ratio will only increase the value, and excluding a subgraph with a larger local approximation ratio will only decrease the value.

Taking Fig. 1 as an example graph, one can order the local approximation ratios $f_0/c_0 \leqslant f_2/c_2 \leqslant f_1/c_1$. Successive lower bounds on the approximation ratio of the graph can be found by including more and more subgraphs, e.g.,

$$\frac{4f_0}{4c_0} \leqslant \frac{4f_0 + f_2}{4c_0 + c_2} \leqslant \frac{4f_0 + 10f_1 + f_2}{4c_0 + 10c_1 + c_2} \leqslant C_1(\mathcal{G}). \qquad (23)$$

Adding additional subgraphs to the count gets a larger lower bound on the approximation ratio of the graph. If the graph $\mathcal{G}$ was worst case, then this ordering results in a strict lower bound on the approximation ratio.

The worst-case graph will include some number of each subgraph in its count $N_\lambda(\mathcal{G}_*)$. It is simple to iterate through the three possible subgraphs as enumerated in the table in the Supplemental Material [15] to find that subgraph 0, the tree, is minimal, with $f_0/c_0 = 0.692/1.000$. Thus, the approximation ratio for the worst-case graph $\mathcal{G}_*$ is bounded from below by only including the minimal subgraph,

$$C_1(\mathcal{G}) \geqslant C_{\min} \geqslant 0.692. \qquad (24)$$

The analysis for the minimum approximation ratio for the $p = 1$ case ends here. This is because there are graphs which only include the tree subgraph, which have no cycles less than 4. The minimum approximation ratio $C_{\min} \leqslant C(\mathcal{G})$ for any graph $\mathcal{G}$ by definition; however, $C_{\min} \leqslant C([\text{tree}]) = 0.692$, and so the *exact* minimum approximation ratio for $p = 1$ is this value. This is the analysis of Farhi *et al.* [2]: They observe that the worst graph is made only of the tree subgraph, and then observe that such a graph exists. This analysis does not hold for the $p = 2$ case.

B. Graph hierarchy for $p = 1$

Before continuing to the more difficult $p = 2$ case, let us introduce a hierarchy of graphs for $p = 1$ where, heuristically, graphs with fewer small cycles have a smaller approximation ratio. We will show that, given a graph $\mathcal{G}$, one can always find a new graph $\mathcal{G}'$ with $C_1(\mathcal{G}) \geqslant C_1(\mathcal{G}')$ unless the graph is of a specific class of graphs with no cycles of length $\leqslant 3$. We denote such graphs as "1-tree graphs," which are constructed
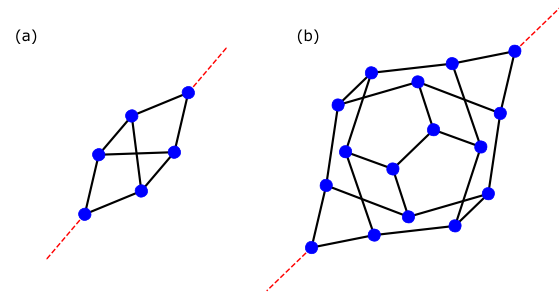

FIG. 7. Edge replacements for $p = 1$ and $p = 2$ graphs. Given an edge E, the edge is replaced by this subgraph, where red dashed indicates the original edge. The left replacement has no cycles $\leqslant 3$, while the right has no cycles $\leqslant 5$, appropriate for $p = 1$ and $p = 2$, respectively.

only out of the $p = 1$ tree subgraph (see Fig. 1 bottom left and Fig. 9). Similar graphs can be defined for p-tree graphs, which have no cycles of length $\leqslant 2p + 1$.

Given a graph with small cycles, a new graph can be found with a worse approximation ratio, which proves inductively that the graph with no small cycles is the worst-case graph via recursion. Given a worst-case candidate graph $\mathcal{G}$ which is not a 1-tree, a 1-tree graph can be shown to have a smaller approximation ratio by recursion $\mathcal{G} \to \mathcal{G}' \to \mathcal{G}'' \to \cdots \to \mathcal{G}_{[1\text{-tree}]} = \mathcal{G}_*$. Thus, 1-tree graphs have a lower approximation ratio than any other graph. Let us continue by proving the graph reduction $\mathcal{G} \to \mathcal{G}'$.

For a graph $\mathcal{G}$, choose an edge E which participates in at least one cycle of length 3. Then, replace the edge with the 6-vertex subgraph of Fig. 7(a). This creates a new graph $\mathcal{G}'$, where the cycle of length 3 that the original edge participated in is replaced with a new cycle of length 7. An example is shown in Fig. 8. Let us prove that this new graph has a smaller approximation ratio.

When this edge is replaced, the graph is modified and so the subgraphs $\mathcal{G}_{\langle ij \rangle}^p$ of the surrounding edges may also be modified. To prove the graph reduction, we must show that this replacement reduces the approximation ratio for all possible modifications of the edges. All possible combinations of subgraph assignments to the edges of $\mathcal{G}_{\langle ij \rangle}^p$ are the set of graph environments of an edge, and so one may prove the graph reduction by checking some condition for all possible graph environments.

A subset of graph environments is the relevant graph environments, which only includes edges whose subgraph assignment is modified under replacement of the center edge. For $p = 1$, there are four relevant graph environments, as replacing the central edge of Figs. 3(a)–3(c) does not change the subgraph assignment of its surroundings. Thus, the graph reduction will involve checking a condition for all possible relevant graph environments.

Now, consider replacing a particular edge of some graph $\mathcal{G}$ with the subgraph of Fig. 7(a) yielding a graph $\mathcal{G}'$. The edges of the graph can be partitioned into two sets: edges which are replaced or have their subgraph assignment modified by the replacement procedure, which are found in the relevant graph environment, and edges which are not modified. The edges which are not modified have expectation values which
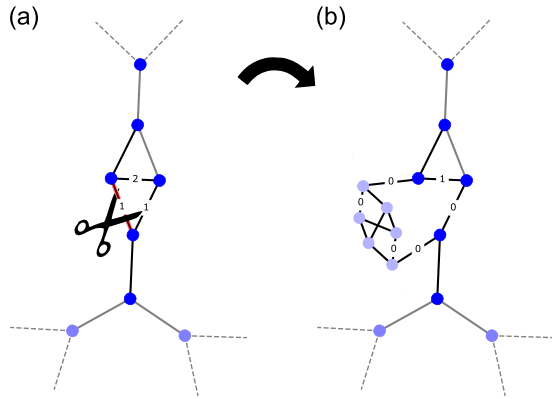
FIG. 8. An example edge replacement operation. The red center edge in some graph environment (a) is replaced with ten edges and six vertices (b), removing the original size-3 cycle. The new graph with the replaced edges will have a smaller approximation ratio. Two additional edges' subgraphs (labeled) are also modified.

$$\mathcal{G} \Rightarrow \cdots \Rightarrow \mathcal{G}' \Rightarrow \cdots \Rightarrow \mathcal{G}_{[1\text{-tree}]}$$
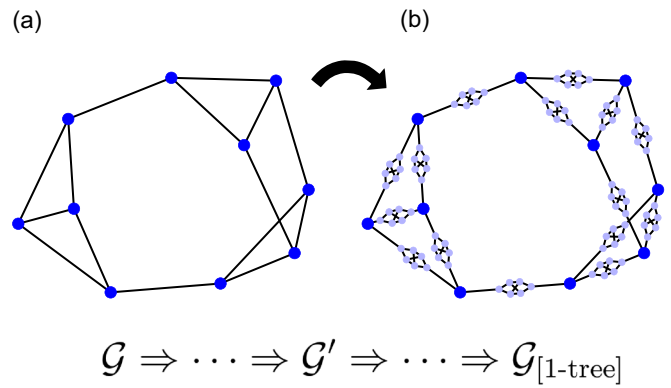
FIG. 9. An example graph reduction. Each edge of some original graph $\mathcal{G}$ (a) is iteratively replaced with a 6-vertex subgraph, until eventually every edge is replaced to find a worst-case graph $\mathcal{G}_{[1\text{-tree}]}$ (b).

sum to F and local MaxCut sum C, and the edges which are replaced or modified have an expectation value sum f and local MaxCut sum c. The replaced or modified edges have a new expectation value sum f' and local MaxCut sum c', which include the additional ten edges of the replacement operation. Now, consider the following clauses and their implication:

$$\mathcal{A}: \quad \frac{F}{C} \leqslant \frac{f}{c},$$

$$\mathcal{B}: \quad \frac{f}{c} \geqslant \frac{f' - f}{c' - c},$$

$$\mathcal{C}: \quad \frac{F + f}{C + c} \geqslant \frac{F + f'}{C + c'},$$

$$\mathcal{A} \wedge \mathcal{B} \Rightarrow \mathcal{C}. \tag{25}$$

Clause $\mathcal{A}$ is a restriction on choice of edge E. One must choose an edge such that, if the replaced or modified edges are removed from the count of subgraphs, the approximation ratio will decrease. By Eq. (22), this choice of edge will always exist by the ordering of subgraphs.

Clause $\mathcal{B}$ is a condition on modifying subgraph assignments of an edge and its surroundings under replacement. This can be checked for every graph environment. For $c' - c > 0$, which is the case when adding more edges under replacement, this clause is equivalent to $f/c \geqslant f'/c'$, that the new subgraph has a smaller local approximation ratio. Clause $\mathcal{B}$ generalizes to other graph modifications, such as reducing a 3-cycle to a single vertex.

Clause $\mathcal{C}$ compares the lower bound approximation ratios for the graphs $\mathcal{G}$ and $\mathcal{G}'$ before and after a replacement procedure. The inequality states that the approximation ratio of the new graph will be smaller. If clauses $\mathcal{A}$ and $\mathcal{B}$ are true, then the new graph has a smaller approximation ratio, which proves the graph reduction for a particular choice of edge replacement.

As an example, consider Fig. 8, performing an edge replacement within some graph with the particular graph environment of Fig. 3(e). Here, there are three edges whose subgraph assignments will be replaced or modified by

the replacement procedure, specifically the three edges of the triangle. Two edges have a subgraph assignment of 1 (single triangle), and one has a subgraph assignment of 2 (two triangles). After the procedure, there are 11 copies of subgraph 0 (the tree), and one copy of subgraph 1 (an edge of a triangle): two from the original graph environment, plus an additional ten from the cut replacement of Fig. 7. Using the table of subgraphs in the Supplemental Material [15], one can compute

$$f = 1.911, \quad f' = 8.253, \quad c = 2.4, \quad c' = 11.8.$$

It is simple to check that these values satisfy clause $\mathcal{B}$ of Eq. (25). To prove that there always exists a graph reduction $\mathcal{G} \to \mathcal{G}'$, one can check all four of the relevant graph environments. It is found that clause $\mathcal{B}$ is true for all relevant environments. Thus, it is shown that, given a graph $\mathcal{G}$, a new graph $\mathcal{G}'$ can be constructed which will have a smaller or equal approximation ratio, done by replacing edges to remove cycles of length 3 in the graph. This creates a hierarchy of graphs for which graphs with fewer small cycles have a smaller approximation ratio,

$$C(\mathcal{G}) \geqslant C(\mathcal{G}') \geqslant \cdots \geqslant C(\mathcal{G}_{[1\text{-tree}]}) = 0.6924, \tag{26}$$

here proved for the $p = 1$ case and consistent with the lower bound and original results [2]. An example graph reduction to a 1-tree is shown in Fig. 9. This hierarchy holds for the fixed angles of Eq. (10), and so this performance guarantee holds for any graph evaluated at these fixed angles.

C. Lower bounds for $p = 2$

Finding a lower bound for $p = 2$ is more complicated, because each edge lives in a larger graph environment. These larger graph environments mean that the simple lower bound method for $p = 1$ is no longer exact. This is because the subgraph assignment of an edge constrains the subgraph assignments of neighboring edges, as there are only a finite number of graph environments. As an example, consider Fig. 3(f), which is the only $p = 1$ graph environment for the two-triangle subgraph 2. Any graph which includes this subgraph must by necessity also include at least four instances
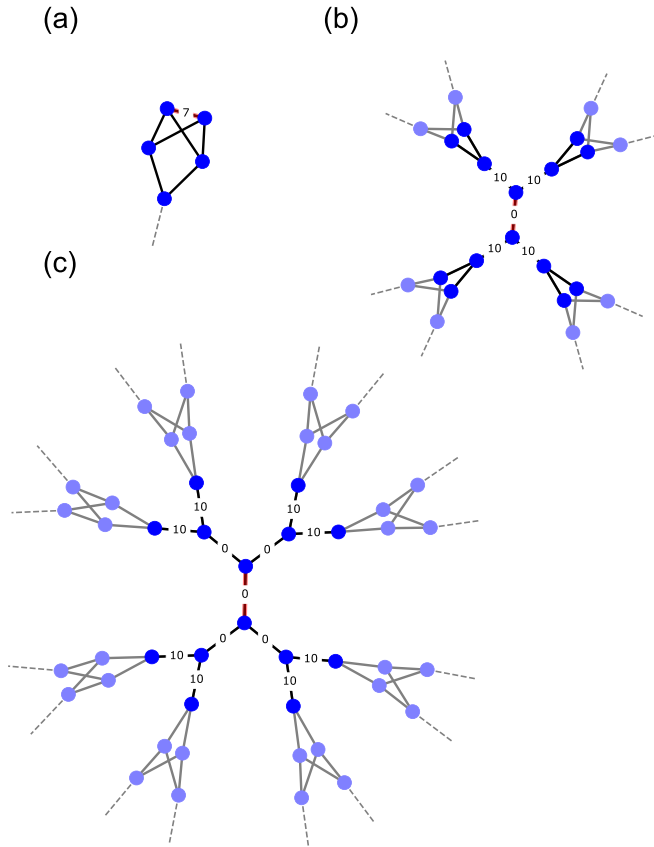
FIG. 10. The three first worst-case graph environments for $p = 2$. For $N = 1$ (a), the approximation ratio is 0.4886. For $N \leqslant 5$ (b), it is 0.7376. For $N \leqslant 14$ (c), it is 0.7424. Each of these graph environments serves to find the lower bound on the approximation ratio of the worst-case graph.

of subgraph 1 (an edge of a triangle). One cannot construct a graph out of just subgraph 2 for $p = 1$.

More generally, even though one may have some count of subgraphs N_λ, there is no guarantee that there exists a graph which has that particular count $N_\lambda \neq N_\lambda(\mathcal{G}) \ \forall \mathcal{G}$. A similar constraint holds for $p = 2$: Neighboring and next-nearest-neighboring edges may be constrained to particular configurations of subgraph assignments, due to the finite numbers of graph environments.

The worst-case graph is constructed out of some count of each kind of subgraph. A lower bound on its approximation ratio can be found by only including a certain subset of its subgraphs with a small local approximation ratio, even though the subset may not have an associated graph. As a first step, one can ignore this constraint and search through the 123 unique $p = 2$ subgraphs of the table in the Supplemental Material [15] to find the subgraph with the smallest local approximation ratio. This is shown in Fig. 10(a), with $f_7/c_7 = 0.4258/0.8571 = 0.4968$. By the argument of Sec. V A, adding any kind of any other subgraph will increase the approximation ratio, and so this number serves as a lower bound on the minimum approximation ratio for the $p = 2$ case.

However, as is clear from inspecting this subgraph, it is impossible to construct a graph out of only this subgraph. This

means that any graph which includes this subgraph will also include some combination of other subgraphs, the inclusion of which increases the approximation ratio.

Thus, this lower bound is loose, as no graph can be constructed with this approximation ratio, but any graph is guaranteed to have a larger approximation ratio. In fact, this bound is so loose that it is below the original $p = 1$ bound, which still holds for $p = 2$. This contrasts with the $p = 1$ case, where there were graphs constructed out of only the worst-case subgraph and the bound was tight.

The next step in tightening this bound is to search through larger graph environments which also identify the subgraphs of the four neighboring edges to find a larger minimum approximation ratio. This graph environment is shown in Fig. 10(b), and has an approximation ratio lower bounded by 0.7431, with two kinds of subgraphs. Again, it is impossible to construct a full graph out of just this graph environment: The outer edges are not allowed to be built of those two subgraphs, and thus any graph will have a strictly larger approximation ratio.

Including graph environments one step larger identifies edges out to a depth 2 and finds the graph environment of Fig. 10(c), with an approximation ratio of at least 0.7461. Beyond this limit, it becomes infeasible to find larger graph environments, due to the rapid growth in their number.

Note that, unlike the simpler $p = 1$ case, one cannot get an exact lower bound, and is instead recursively improvable by searching through larger and larger graph environments. In the next section we consider how to approach the exact lower bound from above using the graph hierarchy.

D. Graph hierarchy for $p = 2$

Let us proceed by repeating the graph hierarchy argument of $p = 1$ for the $p = 2$ case. This case is complicated by having many more potential graph environments, because each subgraph is sensitive to a larger portion of its surroundings. Given a graph $\mathcal{G}$, a new graph $\mathcal{G}'$ can be found with $C_2(\mathcal{G}) \geqslant C_2(\mathcal{G}')$. As in the $p = 1$ case, this is done by choosing a specific edge and replacing it with that of in Fig. 7(b), which is a subgraph of 16 vertices and 25 edges.

Similarly to the $p = 1$ case, one can show that this graph reduction $\mathcal{G} \rightarrow \mathcal{G}'$ leads to a smaller approximation ratio by doing an edge replacement for every possible relevant graph environment, and checking the clauses of Eq. (25) for each. When an edge is replaced, edges up to two steps away from the replaced edge may have their subgraph assignments changed, as such subgraphs include all vertices within two steps of their center edge. Thus, one must check all $p = 2$ graph environments. These can be found via the methods discussed in Appendix B, finding all $p = 4$ subgraphs subject to the constraint that there are no cycles >5 in the minimum cycle basis [17].

However, there are at least 3×10^{10} $p = 2$ graph environments, which is found by estimating a combinatorial lower bound on the number of graph environments for the $p = 2$ tree subgraph. Instead, we find only the relevant graph environments. These relevant graphs are found by attempting to enumerate all relevant $p = 4$ subgraphs in parallel starting with $p = 3$ seed subgraphs, only including edges whose sub-

graph assignment is modified under center edge replacement. There are found to be 7058 such relevant graph environments; some examples are shown in Fig. 4.

The proof of graph reduction for $p = 2$ is as follows. For each relevant $p = 2$ graph environment, replace the special center edge and check the clauses $\mathcal{A} \wedge \mathcal{B}$ of Eq. (25). We find that clause $\mathcal{B}$ is not satisfied for every relevant graph environment; however, there are no relevant graph environments for which $\mathcal{A}$ (the condition on the choice of edge) is false and $\mathcal{B}$ is true. This puts a condition on the choice of edge to be replaced: One must choose an edge such that $\mathcal{A}$ is true, by only choosing edges whose inclusion increases the approximation ratio, for which $\mathcal{B}$ will be true.

This confirms the graph hierarchy for $p = 2$. For each graph $\mathcal{G}$ with some cycles of length $\leqslant 5$, choose an edge and surrounding relevant graph environment $\mathcal{G}_{(ij)}^{2p}$ which, upon removing it from the calculation of the approximation ratio, decreases the approximation ratio. This is forced by clause $\mathcal{A}$ of Eq. (25). Upon replacing this edge with the 16-vertex edge replacement subgraph, the new graph $\mathcal{G}'$ will be guaranteed to have a smaller (or equal) approximation ratio.

Inductively, this constructs a graph where every edge is replaced by the subgraph of Fig. 7 and has no cycles of length $\leqslant 5$, constructed only out of the tree subgraph. The expectation value of the tree subgraph and thus minimum approximation ratio is

$$C_2 \geqslant 0.7559, \tag{27}$$

where worst-case graphs are 2-trees, which have no cycles $\leqslant 5$. This is consistent with the observation in Ref. [2].

In this section, we have found worst-case approximation ratios for $p = 1$ and 2 QAOA. Extending the original methods of Ref. [2], we find a recursively improvable lower bound of $C_2 \geqslant 0.7424$, by considering larger and larger graph environments. Unlike the $p = 1$ case, this lower bound cannot be made exact due to the adjacency restrictions implicit in the construction of graph environments. Using a recursive graph reduction and combinatoric proof, we find that 2-tree graphs, which have no cycles $\leqslant 5$, are exactly worst case. For every graph which is not a 2-tree graph, a new graph can be found with a smaller approximation ratio by finding some particular edge and replacing it with a 16-vertex subgraph which has no cycles $\leqslant 5$. Applied recursively, this eventually turns every edge of the original graph into one of these subgraphs, which is a worst-case 2-tree graph.

VI. FIXING VARIATIONAL PARAMETERS

We have shown that every graph $\mathcal{G}$ has an approximation ratio of $C_2 \geqslant 0.7559$. However, this result is more general. The choice of angles (γ, β), instead of being optimized for the particular graph, is fixed to a particular choice given by Eq. (10). This means that this performance guarantee is stronger: For *fixed* angles and any graph $\mathcal{G}$, the bound still holds,

$$C_1(\mathcal{G}, \{35°\}, \{22°\}) \geqslant 0.6924, \tag{28}$$

$$C_2(\mathcal{G}, \{28°, 31°\}, \{51°, 17°\}) \geqslant 0.7559. \tag{29}$$

TABLE I. The angular parameters for the four degenerate maxima of the $p = 1$ tree subgraph, and eight degenerate maxima of the $p = 2$ tree subgraph. The expectation value of the objective function of any 3-regular graph evaluated at any of these angles is equal, and the approximation ratio is guaranteed to be above 0.6924 and 0.7559, respectively.

γ_1	β_1	γ_2	β_2
35.3°	22.5°		
144.7°	22.5°		
215.3°	67.5°		
324.7°	67.5°		
28.0°	31.8°	51.4°	16.8°
28.0°	31.8°	231.4°	73.2°
152.0°	31.8°	128.6°	73.2°
152.0°	31.8°	308.6°	16.8°
208.0°	58.2°	51.4°	73.2°
208.0°	58.2°	231.4°	16.8°
332.0°	58.2°	128.6°	16.8°
332.0°	58.2°	308.6°	73.2°

This particular set of angles is useful for experiments: Using them with any graph guarantees a particular approximation ratio without the need for a classical optimizer back end. Additionally, we find numerically that gradient descent optimization from these angles finds the global optimum for almost every graph.

It is clear why these angles were chosen: This set of angles is optimal for the worst-case p-graphs. The minimum approximation ratio is found by minimizing over the set of all graphs while maximizing over angles. Any other choice of angles may have been valid, but may not have resulted in a tight minimum approximation ratio or even have the graph hierarchy be true. In fact, this particular choice of angles generates expectation values on subgraphs which are close to the global maximum of each subgraph, which can be seen comparing rows 3 and 5 in the table of subgraphs in the Supplemental Material [15]. There is no reason *a priori* for this to be so.

We also find that these angles are not unique. The landscape of expectation values $F(\gamma, \beta)$ is periodic modulo 2π in γ and modulo $\pi/2$ in β. This is due to SU(2) and $\mathbb{Z}_2$ symmetry (see Appendix A for details). Within these bounds, we find four degenerate maxima for $p = 1$, and eight degenerate maxima for $p = 2$. The angles for the $p = 1$ and 2 tree subgraphs are shown in Table I. Further, we find that for each subgraph, the expectation value of the center edge of that subgraph is the same evaluated at each of the eight angles of any other subgraph.

This means that for all graphs, the expectation value of the objective function will be the same for each of the four angular maxima of any subgraph; for example,

$$C_1(\mathcal{G}, 35°, 113°) = C_1(\mathcal{G}, 145°, 23°) \geqslant 0.6924 \tag{30}$$

for any graph and $p = 1$, within numerical precision, and so forth for the additional two angles of Table I. Any of these angles provides a good starting point for an experimentalist needing good variational parameters.

VII. CONJECTURE: *p*-TREE GRAPHS ARE WORST CASE

We found that replacing an edge with a subgraph with no small cycles results in a smaller approximation ratio for the $p = 1, 2$ cases. It is reasonable to expect that this behavior should extend to larger p. This naturally leads to the following conjectures

Graph hierarchy conjecture. For any fixed p, graph $\mathcal{G}$, and fixed angles $\vec{\gamma}, \vec{\beta}$ optimal to the tree subgraph, there exists an edge replacement with a subgraph generalized from Fig. 7 which results in a graph with a smaller approximation ratio.

In other words, there is a hierarchy of graphs, for which graphs with many small cycles less than $2p + 2$ will have a better quality QAOA result than graphs with few cycles. This is shown to be true for $p = 1$ and 2 in Secs. V B and V D. For larger p the edge replacement must be a larger subgraph with no cycles less than $2p + 2$. This conjecture has two corollaries.

Large loop conjecture. The worst-case graphs for fixed p are p-trees, which have no cycles less than $2p + 2$.

This conjecture is well motivated physically. When an edge is replaced, the algorithm "sees" less of the full graph, due to the fact that it only knows of relations between vertices $\leqslant p$ steps away [19]. This lesser knowledge of the full graph leads to worse answers, as the QAOA algorithm is then oblivious to improved solutions which "see" more of the graph. Similarly, having no "visible" cycles means the algorithm cannot distinguish between large cycles of even versus odd length, and thus cannot make good cut estimates which require this distinction. The worst-case graph for all p would be the Bethe lattice.

Fixed-angle conjecture. Any graph evaluated at fixed angles optimum to the tree subgraph will have an approximation ratio larger than the guarantee.

Angles optimal to the tree subgraph for larger p (e.g., an expansion of Table I) should result in MaxCut answers to any graph approximation ratios guaranteed to be above some value. These angles could be used as initial points for optimizers, or even excluding the optimization loop to compute good answers without feedback. The computation of these optimal angles for larger p is left to future work. This conjecture is consistent with the phenomena of concentration [20], wherein optimal angles appear to be mostly independent of graph instance.

Worst-case approximation ratio for $p = 3$

Under the large loop conjecture, worst-case graphs for $p = 3$ are 3-trees, with no cycles $\leqslant 7$ and constructed only out of the 30-vertex tree subgraph, which has no cycles. Using the same methods for the $p = 1$ and 2 case it is possible to compute the expectation value f_0 for this subgraph, which is thus the worst-case performance guarantee for $p = 3$ under the large loop conjecture. This subgraph has 30 vertices with a Hilbert space dimension of 2^{30}. Using the symmetries of the tree subgraph, this can be reduced to a dimension of $1\,631\,721 \approx 2^{20.6}$, with an additional factor of $1/2$ due to spin-flip symmetry (see Appendix A for details). Optimization of angles using the methods of Appendix A finds

$$C_3 \geqslant 0.7924 \qquad (31)$$

under the large loop conjecture that 3-tree graphs are worst case for $p = 3$. In principle, this bound can be made rigorous by searching through every possible $p = 3$ graph environment and checking the inequalities of Eq. (25) for each. However there are 913 088 unique $p = 3$ graphs and a much larger number of graph environments, which must extend up to six steps away from the replaced edge. While the combinatoric proof is in principle possible as the task is extremely parallelizable, we leave this challenging calculation to future work.

VIII. COMPARISON TO CLASSICAL ALGORITHMS

Here, we compare performance bounds to the best classical algorithms. The most naive classical algorithm is a random guess; it is simple to see that this cuts on average half the edges and so has an approximation ratio of at least 0.5 [21]. It is known that calculating a cut with an approximation ratio $\geqslant 16/17 \approx 0.9412$ is NP-hard [22]. The algorithm of Goemans and Williamson [16] gives an approximation ratio of at least 0.8786 using semidefinite programming, and holds for any graph. For particular subsets of graphs this approximation ratio can be higher; for example, planar graphs can be solved efficiently in polynomial time [23]. The 3-regular graphs, which are the subject of this paper, have a lower bound of at least 0.9326 [24], also using semidefinite programming.

Even comparing to the general Goemans-Williamson algorithm, $p = 2$ QAOA does not achieve quantum advantage, as $0.7559 < 0.8786$. This does not discount the possibility that QAOA has better performance on a subset of graphs than any classical algorithm on the same subset. Finding such subsets of graphs is challenging for two reasons. First, for a particular subset of graphs there may exist some classical algorithm improving on the Goemans-Williamson bound, as for planar graphs or 3-regular graphs, but finding such a specialized algorithm may be nontrivial. Second, the analysis of the subgraph structure of the subgraphs of the table in the Supplemental Material [15] dictates the subset of allowed graphs. By the graph ordering conjecture, graphs with many cycles will have better approximation ratios and thus have the most potential for finding instances with a quantum advantage. Finding such subsets is beyond the scope of this paper.

It is a curious fact that while the classical algorithms of this paper might be able to *find* MaxCut instances which have a quantum advantage, it may not be possible to *solve* those instances classically. This is because sampling bit strings from a QAOA wave function is in complexity class #P, and having such an algorithm would collapse the polynomial hierarchy at the third level [25].

IX. UPPER BOUNDS ON MINIMUM APPROXIMATION RATIO

While computing the minimum approximation ratio for $p > 3$ is challenging, it is reasonable to ask how the minimum approximation ratio behaves with p. As $p \to \infty$, the approximation ratio approaches 1 in accordance with the adiabatic theorem [2]. How does it do so? One way to compute this behavior is to bound the minimum approximation ratio from above: For a particular p, it can be *at most* some value.
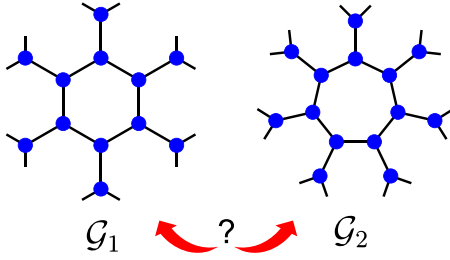
FIG. 11. $p = 2$ QAOA cannot distinguish between a tiling of hexagons ($\mathcal{G}_1$) or heptagons ($\mathcal{G}_2$), as both are constructed only from the tree subgraph. Graph $\mathcal{G}_1$ has a partition which cuts every edge, while graph $\mathcal{G}_2$ has at least one uncut edge per heptagon due to the odd length cycles. This puts upper bounds on the expectation value of the tree graph, and thus the minimum approximation ratio. This generalizes for all p and graph connectivity v.

One way of finding such a bound is to consider pairs of graphs which are indistinguishable under some fixed p QAOA. First, construct a bipartite graph $\mathcal{G}_1$ as a tiling of q-gons with cycles of length $q = 2p + 2$. For example, for $p = 1$ this is a square ladder, while for $p = 2$ this is a hexagonal honeycomb lattice (see Fig. 11). Because all cycles are of even length, it is simple to find a partition which cuts every edge, so that $C_{\max} = n_{\text{edges}}$.

Next, construct a graph $\mathcal{G}_2$, which is a tiling of q-gons with cycles of length $q = 2p + 3$. As an example, for $p = 1$ or 2 this can be seen as pentagons or heptagons (nonmetrically) tiled on some curved surface (see Fig. 11). For N q-gons, there are $Nq/2$ edges. Because the cycles are of odd length, at least one edge per q-gon must remain uncut. Each edge is shared by two q-gons and so for N q-gons there are at least $N/2$ uncut edges and $C_{\max} = N(2p + 2)/2$ cut edges.

For both graphs, there are no cycles of length $\leqslant 2p + 1$, which means only the tree subgraph contributes to the expectation value. Critically, the tree graph cannot distinguish between the two graphs, even though they have different MaxCut values. Consider

$$C(\mathcal{G}_1) = \frac{n_{\text{edges}} f_0}{C_{\max}} = f_0,$$ (32)

and the approximation ratio of the bipartite graph $\mathcal{G}_1$ is simply the expectation value of the tree graph, as every edge is cut. By definition, $C_{\min} \leqslant C(\mathcal{G}_1)$. One may also compute the approximation ratio of $\mathcal{G}_2$, which is bounded from above by 1:

$$C(\mathcal{G}_2) = \frac{n_{\text{edges}} f_0}{C_{\max}} = \frac{(2p + 3) f_0}{(2p + 2)} \leqslant 1.$$ (33)

This bound from above puts an upper bound on the expectation value f_0. In combination, Eq. (32) gives an upper bound on the minimum approximation ratio,

$$C_{\min} \leqslant C(\mathcal{G}_1) = f_0 \leqslant \frac{2p + 2}{2p + 3}.$$ (34)

For p large the bound goes as
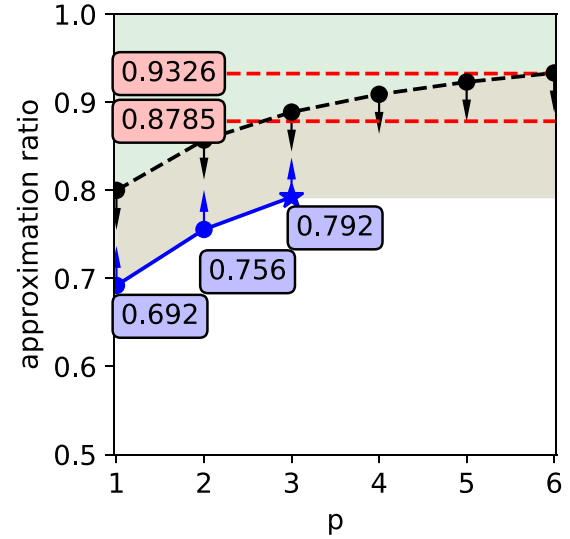
$$C_{\min} \leqslant 1 - \frac{1}{2p}.$$ (35)

FIG. 12. Results of the paper: Approximation ratios vs p. The blue line is the worst-case approximation ratio for $p = 1, 2, 3$; the $p = 3$ case (star) assumes the large loop conjecture. Red dashed are the Goemans-Williamson [16] and 3-regular [24] bounds. The minimum worst-case approximation ratio is guaranteed to be below the black dashed line, which converges as $1 - 1/2p$.

This bound is independent of the graph connectivity and is consistent with the convergence of 2-regular graphs observed in Ref. [2].

Given a particular malicious MaxCut instance with no small cycles, the convergence of QAOA will be inverse polynomial with a power of *at most* 1. Note that other graphs may converge much faster, as this bound only holds for graphs which have no cycles $\leqslant 2p + 1$, which are exponentially large in p. This means for a fixed cycle length, after some very large p convergence can be faster than this bound; the $p \rightarrow \infty$ behavior occurs only for the Bethe lattice, which has an infinite cycle length. The upper bound of Eq. (34) is plotted in Fig. 12. The computed values for 1, 2, and 3 do not come close to this limit, as this bound is loose. From Sec. VIII, the best classical algorithms for 3-regular MaxCut have an approximation ratio of at least 0.9326; thus, p must be at least greater than 5 to have quantum advantage for a general 3-regular graph. This argument is based on a particular graph feature and the $p \leqslant 3$ guarantees tend below the bound, and so one might have a more pessimistic estimate on p due to special purpose classical algorithms and performance guarantees which may not saturate the bounds.

X. CONCLUSION

Bounding the performance of near-term quantum algorithms is critical to understand where, how, and why quantum computers may gain advantage in the NISQ era. In this paper, we find a worst-case performance guarantee for $p = 2$ QAOA solving MaxCut on 3-regular graphs. This performance guarantee was found to be $C_2 \geqslant 0.7559$, which holds for any graph evaluated not just at its optimized angles, but for a fixed set of angles given by Table I. Because this set of angles is fixed,

they can act as good initial guesses for a classical optimizer with a guaranteed approximation ratio.

More important than the number itself, the methods and the particular worst-case graphs for which the bound was derived may be able to motivate particular ensembles and problem classes of graph instances for which QAOA exhibits quantum advantage.

The worst-case graphs for $p = 1$ and 2 were proved to be graphs with no cycles $\leqslant 2p + 1$. This was done via a graph reduction, replacing an edge with an expanded subgraph which distances the two original vertices of the edge and removes small cycles. The QAOA algorithm can only "see" the structure of a graph within some small number of steps, and so the effect of the graph reduction is twofold. In removing and lengthening cycles, due to its local nature the algorithm cannot distinguish between large even and odd length cycles. In expanding the edge into a larger subgraph, the two previously adjacent vertices of the edge are distanced so their previous relation is obscured to the algorithm.

These two properties stemming from the graph reduction suggest which graphs may have good QAOA solutions. Good graphs should have many small cycles, and should have a small-world structure [26] where only a logarithmic number of steps is necessary to move from one vertex to any other.

The properties presented here are heuristic and stem from the graph hierarchy, which is proved for $p \leqslant 2$ but can only be conjectured for $p > 2$. Future work remains to find more rigorous specifications and properties of graphs and problem instances on which it may be possible to demonstrate a quantum advantage in QAOA.

ACKNOWLEDGMENTS

APPENDIX A: COMPUTING OPTIMAL PARAMETERS γ, β

This Appendix details computing the optimal objective function f_λ for particular subgraphs. For each subgraph $\mathcal{S}_\lambda \in \{\mathcal{S}\}$, we wish to compute MAX $: f_\lambda(\gamma, \beta)$. First, generate the local objective function for the subgraph by including one clause $\frac{1}{2}(1 - \hat{\sigma}_z^i \hat{\sigma}_z^j)$ per edge $\langle ij \rangle$ in the subgraph, and similar for the local objective function on the special edge $\langle 0, 1 \rangle$ and mixing Hamiltonian $\hat{B}$ of $\hat{\sigma}_x^i$ for each vertex i in the subgraph. We can compute expectation values exactly, which similarly enables access to derivatives of the objective function

$$\partial_{\gamma_1} \langle \hat{C} \rangle = [\partial_{\gamma_1} \langle \gamma, \beta |] \hat{C} | \gamma, \beta \rangle + \langle \gamma, \beta | \hat{C} [\partial_{\gamma_1} | \gamma, \beta \rangle],$$

$$\partial_{\gamma_1} | \gamma, \beta \rangle = -i e^{-i\beta_p \hat{B}} e^{-i\gamma_p \hat{C}} (\cdots) e^{-i\beta_1 \hat{B}} (\hat{C} e^{i\gamma_1 \hat{C}}) | \gamma, \beta \rangle, \quad \text{(A1)}$$

and similar for the other γ, β, with the ellipsis denoting the other $2p - 4$ generators. With access to both the exact expectation values and derivatives, the parameters were optimized via a multistart gradient ascent algorithm. For each subgraph, the initial parameters were chosen uniformly in parameter space, which is compact in $\{[-\pi/4, \pi/4), [-\pi, \pi)\}^p$. Note that unlike for general QAOA, β is periodic modulo $\pi/2$, due
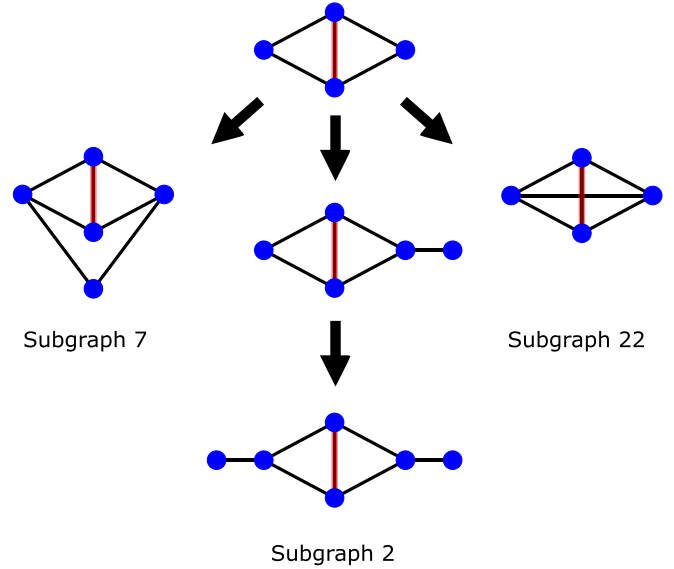

FIG. 13. Recursively constructing some $p = 2$ subgraphs. Starting from some seed $p = 1$ subgraph (top), edges are added (arrows) to vertices which do not have three edges already, connecting to new or existing edges. Recursion through all possibilities finds all subgraphs, up to isomorphisms. The other two $p = 1$ subgraphs have larger recursive trees, eventually finding all 123 $p = 2$ subgraphs.

to $\mathbb{Z}_2$ symmetry, e.g., $\hat{\sigma}_z \rightarrow -\hat{\sigma}_z$. This is because the unitary over the mixing term

$$e^{i(\beta+\pi/2)\hat{B}} = \left(\prod_i e^{i\pi/2\hat{\sigma}_x^i} \right) e^{i\beta\hat{B}} = \hat{\mathbb{Z}}_2 e^{i\beta\hat{B}}, \quad \text{(A2)}$$

and $[\hat{\mathbb{Z}}_2, \hat{B}] = [\hat{\mathbb{Z}}_2, \hat{C}] = 0$. For each optimization, 25 random initial parameters were chosen to find maxima with high probability. To find all degenerate maxima of a subgraph, the initial parameters were chosen on a mesh, and each also optimized with a gradient descent. At each step, the parameters are updated to change along the direction with the largest gradient, with size $0.075|\vec{\nabla}\langle C\rangle|$, where the constant is an implicit choice of the maximum second derivative. A particular optimization ends when the expectation value changes by less than 10^{-5}.

For the $p = 3$ tree, a free optimization was used to speed up the calculation by reducing the Hilbert space using symmetry. Every graph has a certain set of isomorphisms, e.g., relabeling of each vertex. These isomorphisms define SWAP operators which commute with the generators $\hat{B}$ and $\hat{C}$, and thus the eigenvalues are good quantum numbers defining conserved subspaces of the Hilbert space. The initial wave function $|+\rangle$ is unchanged under any relabeling of indices, and so lives in the $+1$ subspace of all isomorphisms. While this reduction is in principle possible for any subgraph, it is particularly simple for the recognizable SWAP symmetries of the tree. Swapping any two "branches" of the tree leaves the wave function invariant, and so only symmetric combinations remain. For two vertices (e.g., $p = 0$), this is only three states: both up, both down, and the triplet state. For six vertices, there are three isomorphisms: swapping the left or right two vertices, or reflecting the three left with three right. Under

these symmetries there are 21 states. This can be done recursively by knowing that given two indistinguishable Hilbert spaces of dimension $\mathcal{D}$, there are only $\mathcal{D}(\mathcal{D}+1)/2$ symmetric combinations. Using this, for $p = 2$ there are 903 states; for $p = 3$ there are $1\,631\,721 \approx 2^{20.6}$; and for $p = 4$ there are $5\,325\,028\,475\,403 \approx 2^{42.3}$. There is an additional factor of ≈ 2, as the generators have a $\mathbb{Z}_2$ spin-flip symmetry $\hat{\sigma}_z \leftrightarrow -\hat{\sigma}_z$. In principle, it may be possible to simulate the worst-case 4-tree exactly, as 41 qubits is within the range of classical simulatability, but is beyond the scope of this work. It is reasonable to expect that these tree subgraphs can be simulated with tensor methods [27] up to $p \sim 10$.

APPENDIX B: SUBGRAPH GENERATION FOR FIXED p

In order to efficiently compute the expectation values of graphs, one must go "in reverse" to find all possible subgraphs $\mathcal{G}_{\langle ij \rangle}^p$. This Appendix details the enumeration of the set of these subgraphs, denoted as $\{\mathcal{S}_\lambda\}_p$. First, find the set of all subgraphs $\{\mathcal{S}_\lambda\}_{p-1}$. For example, for $p = 1$ this is the two-vertex graph connected by an edge; for $p = 2$ these are the three $p = 1$ sub-

graphs, and so forth, generated recursively. Next, for each of these subgraphs, find all the exterior vertices which have less than three edges (see Fig. 13). Then, iterate through adding one, two, or three edges. One may add one edge connecting two vertices of the original seed subgraph (Fig. 13 right), or add one edge connecting a vertex to a new vertex (Fig. 13 middle). Additionally, one may add two edges connecting two vertices to a new vertex (Fig. 13 left) or three edges connecting three vertices to a new vertex (not shown by Fig. 13). Iterating by adding these graphs to a heap, additional edges are added until all exterior vertices of the original subgraphs have three edges, and the heap is empty. This is guaranteed to find all subgraphs, as it searches through every possible permutation of new vertices connected to every combination of exterior vertices of the seed subgraph. When constructing all unique subgraphs, if a subgraph was isomorphic to an already-discovered graph, it is excluded from the heap.

Using this recursion, there were found to be three subgraphs for $p = 1$, 123 subgraphs for $p = 2$, and 913 088 subgraphs for $p = 3$. The $p = 1$ and $p = 2$ subgraphs are enumerated in the table in the Supplemental Material [15].

[1] J. Preskill, Quantum **2**, 79 (2018).

[2] E. Farhi, J. Goldstone, and S. Gutmann, arXiv:1411.4028.

[3] G. E. Crooks, arXiv:1811.08419.

[4] M. B. Hastings, arXiv:1905.07047.

[5] E. Farhi, J. Goldstone, and S. Gutmann, arXiv:1412.6062.

[6] B. Barak, A. Moitra, R. O'Donnell, P. Raghavendra, O. Regev, D. Steurer, L. Trevisan, A. Vijayaraghavan, D. Witmer, and J. Wright, arXiv:1505.03424.

[7] E. Farhi, J. Goldstone, and S. Gutmann, arXiv:1412.6062.

[8] M. Garey, D. Johnson, and L. Stockmeyer, Theor. Comput. Sci. **1**, 237 (1976).

[9] E. Crosson, E. Farhi, C. Y.-Y. Lin, H.-H. Lin, and P. Shor, arXiv:1401.7320.

[10] Z. Wang, S. Hadfield, Z. Jiang, and E. G. Rieffel, Phys. Rev. A **97**, 022304 (2018).

[11] M. Willsch, D. Willsch, F. Jin, H. De Raedt, and K. Michielsen, Quantum Inf. Process. **19**, 197 (2020).

[12] J. Larkin, M. Jonsson, D. Justice, and G. G. Guerreschi, arXiv:2006.04831.

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

[14] M. P. Harrigan, K. J. Sung, M. Neeley, K. J. Satzinger, F. Arute, K. Arya, J. Atalaya, J. C. Bardin, R. Barends, S. Boixo *et al.*, Nat. Phys. **17**, 332 (2021).

[15] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevA.103.042612 for a list of enumerated subgraphs.

[16] M. X. Goemans and D. P. Williamson, J. ACM **42**, 1115 (1995).

[17] K. Paton, Commun. ACM **12**, 514 (1969).

[18] T. Zaslavsky, AKCE Int. J. Graphs Comb. **15**, 31 (2018).

[19] E. Farhi, D. Gamarnik, and S. Gutmann, arXiv:2005.08747.

[20] F. G. S. L. Brandao, M. Broughton, E. Farhi, S. Gutmann, and H. Neven, arXiv:1812.04170.

[21] S. Sahni and T. Gonzalez, J. ACM **23**, 555 (1976).

[22] J. Håstad, J. ACM **48**, 798 (2001).

[23] M. Deza and M. Laurent, J. Comput. Appl. Math. **55**, 217 (1994).

[24] E. Halperin, D. Livnat, and U. Zwick, J. Algorithms **53**, 169 (2004).

[25] E. Farhi and A. W. Harrow, arXiv:1602.07674.

[26] D. J. Watts and S. H. Strogatz, Nature (London) **393**, 440 (1998).

[27] M. Streif and M. Leib, Quantum Sci. Technol. **5**, 034008 (2020).