The Quantum Approximate Optimization Algorithm at High Depth for MaxCut on Large-Girth Regular Graphs and the Sherrington-Kirkpatrick Model

Joao Basso ✉ 🏠

Google Quantum AI, Venice, CA, USA

Edward Farhi ✉

Google Quantum AI, Venice, CA, USA
Center for Theoretical Physics, Massachusetts Institute of Technology, Cambridge, MA, USA

Kunal Marwaha ✉ 🏠

Department of Computer Science, University of Chicago, IL, USA

Benjamin Villalonga ✉

Google Quantum AI, Venice, CA

Leo Zhou ✉

Walter Burke Institute for Theoretical Physics, California Institute of Technology, Pasadena, CA, USA

Abstract

The Quantum Approximate Optimization Algorithm (QAOA) finds approximate solutions to combinatorial optimization problems. Its performance monotonically improves with its depth p. We apply the QAOA to MaxCut on large-girth D-regular graphs. We give an iterative formula to evaluate performance for any D at any depth p. Looking at random D-regular graphs, at optimal parameters and as D goes to infinity, we find that the $p = 11$ QAOA beats all classical algorithms (known to the authors) that are free of unproven conjectures. While the iterative formula for these D-regular graphs is derived by looking at a single tree subgraph, we prove that it also gives the ensemble-averaged performance of the QAOA on the Sherrington-Kirkpatrick (SK) model defined on the complete graph. We also generalize our formula to Max-q-XORSAT on large-girth regular hypergraphs. Our iteration is a compact procedure, but its computational complexity grows as $O(p^2 4^p)$. This iteration is more efficient than the previous procedure for analyzing QAOA performance on the SK model, and we are able to numerically go to $p = 20$. Encouraged by our findings, we make the optimistic conjecture that the QAOA, as p goes to infinity, will achieve the Parisi value. We analyze the performance of the quantum algorithm, but one needs to run it on a quantum computer to produce a string with the guaranteed performance.

2012 ACM Subject Classification Theory of computation → Quantum computation theory; Theory of computation → Approximation algorithms analysis; Mathematics of computing → Combinatorial optimization

Keywords and phrases Quantum algorithm, Max-Cut, spin glass, approximation algorithm

Digital Object Identifier 10.4230/LIPIcs.TQC.2022.7

Related Version *Full Version*: `https://arxiv.org/abs/2110.14206` [4]

Funding *Kunal Marwaha*: National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1746045. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

17th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2022).
Editors: François Le Gall and Tomoyuki Morimae; Article No. 7; pp. 7:1–7:21
LIPICS Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Acknowledgements The authors thank Sam Gutmann for being there and Matthew P. Harrigan for a careful read of the manuscript.

1 Introduction

We are at the start of an era in which quantum devices are running algorithms. We need to understand the power of quantum computers for solving or finding approximate solutions to combinatorial optimization problems. One approach is to learn by experimenting on hardware. Although useful for probing the hardware and testing algorithms at small sizes, it does not give a convincing picture of asymptotic behavior. To this end we need mathematical studies of the behavior of quantum algorithms, running on ideal circuits, at large sizes. In this paper we take a step in that direction by analyzing the Quantum Approximate Optimization Algorithm as applied to a certain combinatorial optimization problem. The instances are large and the depth of the algorithm is high. For this task, we will see that the QAOA outperforms the best assumption-free classical algorithm.

MaxCut is a combinatorial optimization problem on bit strings whose input is a graph. Each bit is associated with a vertex, and the goal is to maximize the number of edges with bit assignments that disagree on the two ends of the edge. It is NP-hard to solve this problem exactly, and even approximating the optimal solution beyond a certain ratio is NP-hard [30]. We focus on MaxCut for large-girth D-regular graphs. On these graphs, the currently known best classical algorithms (including Goemans-Williamson and the Gaussian wave process [23, 20, 3, 29]) achieve an average-case cut fraction (the number of cut edges output by the algorithm divided by the number of edges) of $1/2 + (2/\pi)/\sqrt{D}$ as both the girth and D go to ∞, where $2/\pi \approx 0.6366$.

We apply the Quantum Approximate Optimization Algorithm (QAOA) [15] to large-girth D-regular graphs. The QAOA depends on a parameter p, the algorithm's depth. At small p, the QAOA has been realized in current quantum hardware [19]. Some analytic results are also known. At $p = 1$, the QAOA has a guaranteed approximation ratio (the number of cut edges output by the algorithm divided by the maximum number of edges that can be cut) of at least 0.6924 on all 3-regular graphs [15] and an expected cut fraction of at least $1/2 + 0.3032/\sqrt{D}$ on triangle-free graphs [31]. For $p = 2$, the QAOA has an approximation ratio of at least 0.7559 on 3-regular graphs with girth more than 5 and, for $p = 3$, that ratio becomes 0.7924 when the girth is more than 7 [32]. So far, expressions for the QAOA's performance on any fixed-D regular, large-girth graph are known only for $p = 1$ [31] and $p = 2$ [21].

In this work, we analyze the performance of the QAOA on any large-girth D-regular graph for any choice of p by looking at a single tree subgraph. Using the regularity of this tree subgraph, we derive an iteration that computes the performance of the QAOA. After optimizing over the $2p$ input parameters, we find that the $p = 11$ QAOA improves on $1/2 + (2/\pi)/\sqrt{D}$, when D is large and the girth is more than 23. This is better than all assumption-free classical algorithms known to the authors.[1]

We also show that this performance, obtained from one subgraph, is mathematically equal to the ensemble-averaged performance of the QAOA applied to the Sherrington-Kirkpatrick

[1] There is a recent classical message-passing algorithm [1] that also does better than $1/2 + (2/\pi)/\sqrt{D}$ for MaxCut on large-girth D-regular graphs. It gets asymptotically close to the optimum assuming the solution space has no "overlap gap property" (see [17] for a review).

(SK) model [16]. This implies that the iteration in this paper can also be used to give the QAOA's performance on the SK model. A recent related work can be found in Ref. [8]. Our iteration is more efficient than the one originally shown in Ref. [16], and we have been able to go numerically to higher depth.

Encouraged by our findings, we conjecture that the large p performance of the QAOA will achieve the optimal cut fraction on large random D-regular graphs, where a vanishing fraction of neighborhoods are not locally tree-like. The optimal cut fraction on these graphs is also related to the SK model. It is $1/2+\Pi_*/\sqrt{D}+o(1/\sqrt{D})$, where $\Pi_* = 0.763166\ldots$, the Parisi value, is the ground state energy density of the SK model [25, 12]. If our conjecture is right we have a simple, though computationally intensive, new iteration for calculating Π_*.

Generalizing our formalism, we also analyze the performance of the QAOA for Max-q-XORSAT (of which MaxCut is a special case at $q = 2$) on large-girth D-regular hypergraphs. The $p = 1$ QAOA was recently found to do better than an analogous classical threshold algorithm for $q > 4$ [22]. The iterative formula for general q is very similar to that for MaxCut and has the same time and memory complexities in the $D \to \infty$ limit. We run this iteration to find optimal QAOA parameters and performance for $3 \leq q \leq 6$ and $1 \leq p \leq 14$. Moreover, we discuss potential obstructions to the QAOA from not "seeing" the whole graph.

The paper is organized as follows. In Section 2, we introduce the necessary definitions to describe the QAOA and the MaxCut problem. In Section 3, we describe two iterations that compute the performance of the QAOA for MaxCut on large-girth D-regular graphs at fixed depth: one for finite D and the other for $D \to \infty$ (proof in Appendix A). We also present our results from numerical evaluation and optimization of the QAOA objective function up to $p = 20$. In Section 4, we argue that the performance of the QAOA on large-girth regular graphs and on the SK model are equivalent. We conjecture in Section 5 that the iteration in Section 3.2 for infinite D is an alternative procedure to compute the Parisi value. In Section 6, we generalize our iterations to evaluate the QAOA's performance for Max-q-XORSAT on large-girth regular hypergraphs. Finally, in Section 7 we discuss our results and suggest some future avenues of work.

2 Background on the QAOA and MaxCut

The QAOA [15] is a quantum algorithm for finding approximate solutions to combinatorial optimization problems. The cost function counts the number of clauses satisfied by an input string. Given a cost function $C(\boldsymbol{z})$ on strings $\boldsymbol{z} \in \{\pm 1\}^n$, we can define a corresponding quantum operator, diagonal in the computational basis, as $C|\boldsymbol{z}\rangle = C(\boldsymbol{z})|\boldsymbol{z}\rangle$. Moreover, let $B = \sum_{j=1}^n X_j$, where X_j is the Pauli X operator acting on qubit j. Let $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_p)$ and $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_p)$. The QAOA initializes the system of qubits in the state $|s\rangle = |+\rangle^{\otimes n}$ and applies p alternating layers of $e^{-i\gamma_j C}$ and $e^{-i\beta_j B}$ to prepare the state

$$|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle = e^{-i\beta_p B} e^{-i\gamma_p C} \cdots e^{-i\beta_1 B} e^{-i\gamma_1 C} |s\rangle. \tag{2.1}$$

For a given cost function C, the corresponding QAOA objective function is $\langle \boldsymbol{\gamma}, \boldsymbol{\beta}|C|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle$. Preparing the quantum state $|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle$ and then measuring in the computational basis enough times, one will find a bit string $\boldsymbol{z}$ such that $C(\boldsymbol{z})$ is near $\langle \boldsymbol{\gamma}, \boldsymbol{\beta}|C|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle$ or better.

We study the performance of the QAOA on MaxCut. Given a graph $G = (V, E)$ with vertices in V and edges in E, the MaxCut cost function is

$$C_{\mathrm{MC}}(\boldsymbol{z}) = \sum_{(u,v)\in E} \frac{1}{2}(1 - z_u z_v). \tag{2.2}$$

We restrict our attention to graphs that are regular and have girth greater than $2p+1$. We work with these graphs because the subgraph that the QAOA at depth p sees on them are regular trees and this enables our calculation. Here, by "seeing" we refer to the fact that the output of the QAOA on a qubit depends only on a neighborhood of qubits that are within distance p to the given qubit on the graph. In what follows, we focus on $(D+1)$-regular graphs, which implies the subgraph seen by the QAOA on each edge is a D-ary tree.

With D large, we will see that the optimal $\boldsymbol{\gamma}$ are of order $1/\sqrt{D}$. So we find it convenient to prepare the QAOA state $|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle$ using the scaled cost function operator

$$C = -\frac{1}{\sqrt{D}} \sum_{(u,v)\in E} Z_u Z_v, \tag{2.3}$$

where we have subtracted a constant that only introduces an irrelevant phase. The factor of $1/2$ has been dropped so that this form of the cost function will match the cost function used in the Sherrington-Kirkpatrick model. Note we are preparing the state $|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle$ using C as a driver instead of the C_{MC} operator. With this scaling, the optimal $\boldsymbol{\gamma}$ will be of order unity instead of $1/\sqrt{D}$.

Given any edge in a $(D+1)$-regular graph with girth greater than $2p+1$ the subgraph with vertices at most p away from the edge is a D-ary tree regardless of which edge. Since the QAOA at depth p only sees these trees, we have

$$\langle\boldsymbol{\gamma},\boldsymbol{\beta}|C_{\mathrm{MC}}|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle = \frac{1}{2}|E|\Big(1-\langle\boldsymbol{\gamma},\boldsymbol{\beta}|Z_uZ_v|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle\Big) \tag{2.4}$$

where $(u,v)\in E$ is any edge. The cut fraction output by the QAOA is then

$$\frac{\langle\boldsymbol{\gamma},\boldsymbol{\beta}|C_{\mathrm{MC}}|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle}{|E|} = \frac{1}{2} - \frac{1}{2}\langle\boldsymbol{\gamma},\boldsymbol{\beta}|Z_uZ_v|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle\,. \tag{2.5}$$

Since the QAOA cannot beat the optimal cut fraction of $1/2+\mathrm{order}(1/\sqrt{D})$ in a typical random regular graph, we write

$$\frac{\langle\boldsymbol{\gamma},\boldsymbol{\beta}|Z_uZ_v|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle}{2} = -\frac{\nu_p(D,\boldsymbol{\gamma},\boldsymbol{\beta})}{\sqrt{D}} \tag{2.6}$$

where $\nu_p(D,\boldsymbol{\gamma},\boldsymbol{\beta})$ for good parameters will be of order unity.

3 The QAOA on large-girth $(D+1)$-regular graphs

We describe two iterations to evaluate the performance of the QAOA at high depth on MaxCut on large-girth $(D+1)$-regular graphs. The cut fraction output by the QAOA at any parameters is

$$\frac{\langle\boldsymbol{\gamma},\boldsymbol{\beta}|C_{\mathrm{MC}}|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle}{|E|} = \frac{1}{2} + \frac{\nu_p(D,\boldsymbol{\gamma},\boldsymbol{\beta})}{\sqrt{D}}. \tag{3.1}$$

We give one iteration to evaluate $\nu_p(D,\boldsymbol{\gamma},\boldsymbol{\beta})$ at finite D, and one for the $D\to\infty$ limit. We have attempted to make this section self-contained for those readers only interested in the form of the iterations, and deferred the detailed proofs of these iterations to Appendix A.

In what follows, we index vectors in the following order:

$$\boldsymbol{a} = (a_1, a_2, \cdots, a_p, a_0, a_{-p}, \cdots, a_{-2}, a_{-1})\,. \tag{3.2}$$

Define, for $1 \leq r \leq p$,

$$\Gamma_r = \gamma_r, \qquad \Gamma_0 = 0, \qquad \Gamma_{-r} = -\gamma_r. \tag{3.3}$$

That is, $\mathbf{\Gamma}$ is a $(2p+1)$-component vector. Furthermore, let

$$\begin{aligned} f(\boldsymbol{a}) = \frac{1}{2} &\langle a_1|e^{i\beta_1 X}|a_2\rangle \cdots \langle a_{p-1}|e^{i\beta_{p-1} X}|a_p\rangle \langle a_p|e^{i\beta_p X}|a_0\rangle \\ &\times \langle a_0|e^{-i\beta_p X}|a_{-p}\rangle \langle a_{-p}|e^{-i\beta_{p-1} X}|a_{-(p-1)}\rangle \cdots \langle a_{-2}|e^{-i\beta_1 X}|a_{-1}\rangle \end{aligned} \tag{3.4}$$

where $a_i \in \{+1, -1\}$ enumerates the two computational basis states, and

$$\langle a_1|e^{i\beta X}|a_2\rangle = \begin{cases} \cos(\beta) & \text{if } a_1 = a_2 \\ i\sin(\beta) & \text{if } a_1 \neq a_2. \end{cases} \tag{3.5}$$

3.1 An iteration for any finite D

Here we give an iteration that allows us to evaluate $\nu_p(D, \boldsymbol{\gamma}, \boldsymbol{\beta})$ for any input parameters and D.

Let $H_D^{(m)} \colon \{-1,1\}^{2p+1} \to \mathbb{C}$ for $0 \leq m \leq p$. We start with $H_D^{(0)}(\boldsymbol{a}) = 1$ and let

$$H_D^{(m)}(\boldsymbol{a}) = \left(\sum_{\boldsymbol{b}} f(\boldsymbol{b}) H_D^{(m-1)}(\boldsymbol{b}) \cos\left[\tfrac{1}{\sqrt{D}} \mathbf{\Gamma} \cdot (\boldsymbol{ab})\right] \right)^D \quad \text{for } 1 \leq m \leq p \tag{3.6}$$

where we denote $\boldsymbol{ab}$ as the entry-wise product, i.e. $(\boldsymbol{ab})_j = a_j b_j$. By starting with $H_D^{(0)}(\boldsymbol{a}) = 1$ and iteratively evaluating Eq. (3.6) for $m = 1, 2, \ldots, p$, we arrive at $H_D^{(p)}(\boldsymbol{a})$ that can be used to compute

$$\nu_p(D, \boldsymbol{\gamma}, \boldsymbol{\beta}) = \tfrac{i\sqrt{D}}{2} \sum_{\boldsymbol{a},\boldsymbol{b}} a_0 b_0 f(\boldsymbol{a}) f(\boldsymbol{b}) H_D^{(p)}(\boldsymbol{a}) H_D^{(p)}(\boldsymbol{b}) \sin\left[\tfrac{1}{\sqrt{D}} \mathbf{\Gamma} \cdot (\boldsymbol{ab})\right]. \tag{3.7}$$

We prove this in Appendix A.1. The key idea is to use the fact that when girth $> 2p+1$, the subgraph seen by the QAOA is a pair of D-ary trees of p levels glued at their roots (see Figure 1(a) for an example). Then ν_p is given as a sum over all $O(D^p)$ nodes in this subgraph. Since every node in the tree has exactly D children that couples to their parent in exactly the same way, we can greatly simplify the process by summing from the leaves of the tree, then their parents, and their parents' parents, and so on. This yields a p-step iteration where at each step $m = 1, 2, \ldots, p$, we have a compact description of the contributions of the nodes from the bottom $m-1$ levels via $H_D^{(m-1)}$ (see Figure 1(b)).

Note that each step of the above iteration involves a sum with 2^{2p+1} terms for each of the 2^{2p+1} entries of $H_D^{(m)}(\boldsymbol{a})$. The final step has a sum with $O(16^p)$ terms. Overall, this iteration has a time complexity of $O(p\,16^p)$ and a memory complexity of $O(4^p)$. This is much faster than the original "light cone" approach that directly evaluates $\langle Z_u Z_v \rangle$ on the subgraph seen by the QAOA [15]. That procedure takes $2^{O(D^p)}$ time without utilizing the symmetric structure of the regular tree subgraph.

3.2 An iteration for $D \to \infty$

We find that in the infinite D limit we get a more compact iteration which takes fewer steps to evaluate. We state the result here and prove it in Appendix A.2.

Define matrices $G^{(m)} \in \mathbb{C}^{(2p+1)\times(2p+1)}$ for $0 \le m \le p$ as follows. For $j,k \in \{1,\dots,p,0,-p,\dots,-1\}$, let $G^{(0)}_{j,k} = \sum_{\boldsymbol{a}} f(\boldsymbol{a}) a_j a_k$, and

$$G^{(m)}_{j,k} = \sum_{\boldsymbol{a}} f(\boldsymbol{a}) a_j a_k \exp\Big(-\frac{1}{2}\sum_{j',k'=-p}^{p} G^{(m-1)}_{j',k'} \Gamma_{j'} \Gamma_{k'} a_{j'} a_{k'}\Big) \quad \text{for } 1 \le m \le p. \tag{3.8}$$

Starting at $m=0$ and going up by p steps, we arrive at $G^{(p)}$ which is used to compute

$$\nu_p(\boldsymbol{\gamma},\boldsymbol{\beta}) := \lim_{D\to\infty} \nu_p(D,\boldsymbol{\gamma},\boldsymbol{\beta}) = \frac{i}{2}\sum_{j=-p}^{p} \Gamma_j (G^{(p)}_{0,j})^2. \tag{3.9}$$

Since there are $p+1$ matrices with $O(p^2)$ entries, and each involves a sum over $O(4^p)$ terms, this iteration naïvely has a time complexity of $O(p^3 4^p)$. This is quadratically better than the time complexity of the finite-D formula. The memory complexity is only $O(p^2)$ for storing the $G^{(m)}$ matrix, which is exponentially better than $O(4^p)$ memory needed to store the entries of $H^{(m)}_D$ in the finite-D iteration.

We note some properties about this iteration. Superficially Eq. (3.8) looks like a recursive map on the matrices $G^{(m)}$ which one might think would only asymptotically converge in the number of steps. However it converges to a fixed point $G^{(p)}$ after p steps in a highly structured way. In particular, the iteration has the following three sets of properties, whose proof can be found in Ref. [4, Appendix A]. We use the convention $1 \le r < s \le p$ and $j,k \in \{1,\dots,p,0,-p,\dots,-1\}$.

(a) Values of the diagonal and anti-diagonal of $G^{(m)}$ are all 1. $G^{(m)}$ is symmetric with respect to the diagonal, reflection with respect to the anti-diagonal results in complex conjugation, and the matrix consists of 8 triangular regions which are rotations, reflections, and/or complex conjugations of each other. To be precise, $G^{(m)}$ satisfies the following properties:

(1) $G^{(m)}_{j,k} = G^{(m)}_{k,j}$

(2) $G^{(m)}_{j,j} = G^{(m)}_{j,-j} = 1$

(3) $G^{(m)}_{0,r} = G^{(m)*}_{0,-r}$

(4) $G^{(m)}_{r,s} = G^{(m)}_{r,-s} = G^{(m)*}_{-r,-s} = G^{(m)*}_{-r,s}$

These are sketched in Figure 1(c).

(b) $G^{(m)}_{r,s}$ only depends on $G^{(m-1)}_{r',s'}$ where $1 \le r' < s' < s$. Similarly, $G^{(m)}_{0,r}$ only depends on $G^{(m-1)}_{r',s'}$ for $1 \le r' < s' \le p$.

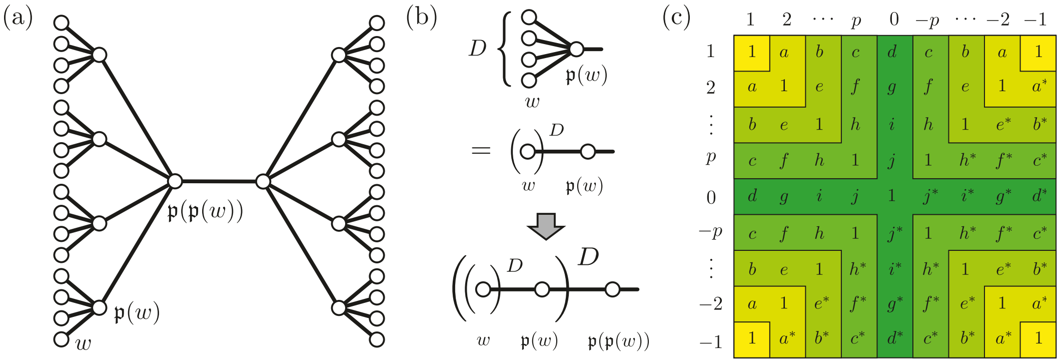
(c) As a consequence of (b), at each step m of the iteration the corner blocks of size $(m+1)\times(m+1)$ of $G^{(m)}$ converge to their final value, i.e., they reach a fixed point and do not change in later iteration steps. This implies that matrix $G^{(p)}$ is a fixed point. This is sketched in Figure 1(c), where matrix entries of the same color reach their fixed point at the same step of the iteration, starting from the corners and ending with the central "cross" at step p.

Making use of (b) and some properties of $f(\boldsymbol{a})$ allows us to lower the complexity of the iterative procedure to $O(p^2 4^p)$. For more details, see Ref. [4, Appendix A.4].

3.3 Numerical evaluation and optimization for the $D \to \infty$ limit

Let

$$\bar{\nu}_p = \max_{\boldsymbol{\gamma},\boldsymbol{\beta}} \nu_p(\boldsymbol{\gamma},\boldsymbol{\beta}). \tag{3.10}$$



	1	2	$\cdots$	p	0	$-p$	$\cdots$	-2	-1
1	1	a	b	c	d	c	b	a	1
2	a	1	e	f	g	f	e	1	a^*
$\vdots$	b	e	1	h	i	h	1	e^*	b^*
p	c	f	h	1	j	1	h^*	f^*	c^*
0	d	g	i	j	1	j^*	i^*	g^*	d^*
$-p$	c	f	h	1	j^*	1	h^*	f^*	c^*
$\vdots$	b	e	1	h^*	i^*	h^*	1	e^*	b^*
-2	a	1	e^*	f^*	g^*	f^*	e^*	1	a^*
-1	1	a^*	b^*	c^*	d^*	c^*	b^*	a^*	1

Figure 1 (a) Example tree subgraph seen by the QAOA at $p = 2$ on a large-girth regular graph. For any node w on the tree, we denote $\mathfrak{p}(w)$ as its parent. (b) A visualization of our iteration for finite D. (c) Sketch of the properties of matrices $G^{(m)}$ in our iteration for $D \to \infty$, at $p = 4$. Regions of the same color converge in the same iteration step, starting from the corners and with the central row and column converging after p steps.

Numerically implementing the iteration summarized in Section 3.2 and optimizing for γ, β we find $\bar{\nu}_p$ up to $p = 17$. The values are given in Table 1 and plotted in Figure 2 as a function of $1/p$. The optimal γ and β can be found in Ref. [4, Table 4], and some examples are plotted in Figure 3. Based on the smooth pattern of the optimal γ and β up to p of 17, we guess these parameters at $p = 18, 19, 20$ using heuristics similar to that in Ref. [33]. Then evaluation of $\nu_p(\gamma, \beta)$ gives lower bounds on $\bar{\nu}_p$ at higher p which are listed in Table 2, and their corresponding γ and β are listed in Ref. [4, Table 5].

Note that, at $p = 11$ and beyond, the QAOA achieves a cut fraction better than $\frac{1}{2} + \frac{2/\pi}{\sqrt{D}}$ in the large D limit, making it the best currently known assumption-free algorithm for MaxCut on large random regular graphs.

Table 1 Optimal values of $\bar{\nu}_p$ up to $p = 17$.

p	1	2	3	4	5	6	7	8	9
$\bar{\nu}_p$	0.3033	0.4075	0.4726	0.5157	0.5476	0.5721	0.5915	0.6073	0.6203

p	10	11	12	13	14	15	16	17
$\bar{\nu}_p$	0.6314	0.6408	0.6490	0.6561	0.6623	0.6679	0.6729	0.6773

Table 2 Lower bounds of $\bar{\nu}_p$ for $p = 18, 19, 20$.

p	18	19	20
$\bar{\nu}_p$ lower bound	0.6813	0.6848	0.6879

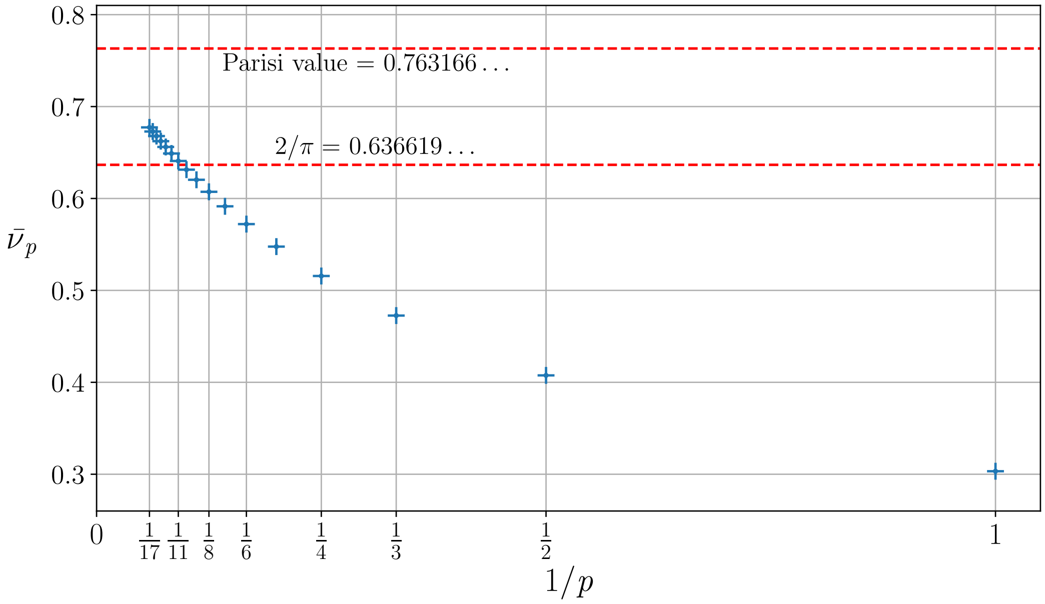


Figure 2 Optimal values $\bar{\nu}_p$ as a function of $1/p$. At $p = 11$, $\bar{\nu}_p$ exceeds $2/\pi$, related to the cut fraction of the best currently known assumption-free classical algorithms. Here we made the somewhat arbitrary choice of plotting the data against $1/p$ to see the large p region in a compact plot.

We implement the iterative procedure described in Section 3.2 in C++. Our code is available at Ref. [5]. Bit strings are encoded as `unsigned long int` variables, which allow for fast bit-wise manipulations. Matrices and vectors are implemented using the Eigen library [18]. We parallelize the sum over $\boldsymbol{a}$ in Eq. (3.8) using OpenMP [11]. We optimize $\boldsymbol{\gamma}, \boldsymbol{\beta}$ for each value of p using the LBFGS++ library, which implements the Limited-memory BFGS algorithm for unconstrained optimization problems [26]. Each evaluation of the gradient of $\nu_p(\boldsymbol{\gamma}, \boldsymbol{\beta})$ in Eq. (3.9) is a subroutine of the optimization which takes $2p + 1$ function calls. We run on a `n2d-highcpu-224` machine in Google Cloud, which has 224 vCPUs, using one thread per vCPU. A function call at $p = 16$ takes about 133 seconds, and a function call at $p = 17$ takes about 595 seconds. The run time of each function call is roughly multiplied by 4 every time p is increased by 1. At $p = 20$, a single function call takes slightly under 14 hours to evaluate. Memory usage is dominated by the need to store matrix $G^{(m)}$, which is negligible and quadratic in p. Further optimizations might be possible.

4 Agreement with the Sherrington-Kirkpatrick model

We note that Table 1 in this paper seems to be an extension of Table 1 in Ref. [16]. There, the authors study the performance of the QAOA on the Sherrington-Kirkpatrick (SK) model [24], which describes a spin-glass system with all-to-all random couplings. The cost function is

$$C_J^{\mathrm{SK}}(\boldsymbol{z}) = \frac{1}{\sqrt{n}} \sum_{1 \le i < j \le n} J_{ij} z_i z_j \tag{4.1}$$

where the J_{ij} are independently drawn from a distribution with mean 0 and variance 1. The authors arrive at an iterative formula for the ensemble-averaged performance of the QAOA on the SK model

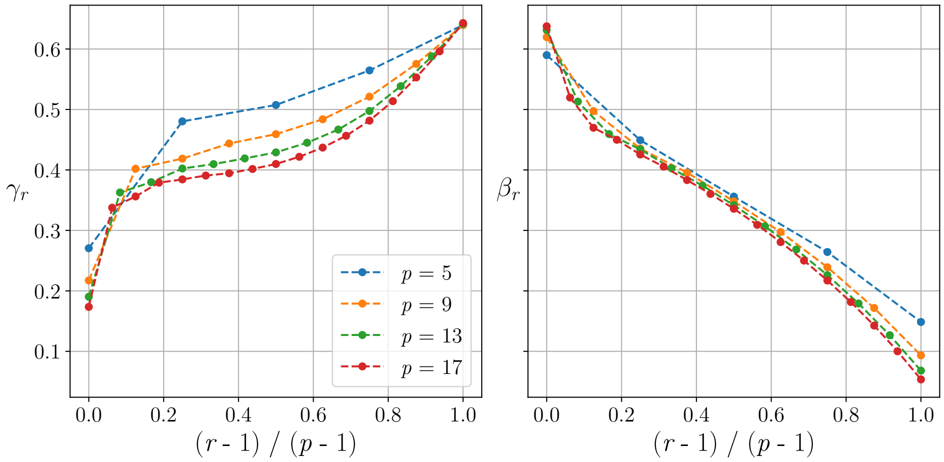


Figure 3 Optimal γ_r and β_r as a function of $(r-1)/(p-1) \in [0,1]$ for $p = 5, 9, 13, 17$. For each p, the index $r = 1, 2, \ldots, p$ enumerates the entries of $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$. Dashed lines in between data points are solely intended to guide the eye.

$$V_p(\boldsymbol{\gamma}, \boldsymbol{\beta}) := \lim_{n\to\infty} \mathbb{E}_J \Big[{}_J\langle \boldsymbol{\gamma}, \boldsymbol{\beta} | C_J^{\mathrm{SK}}/n | \boldsymbol{\gamma}, \boldsymbol{\beta}\rangle_J \Big], \tag{4.2}$$

where $|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle_J$ is the QAOA state prepared with C_J^{SK}. Since concentration is shown to hold, we know that typical instances of the SK model all behave as the ensemble average.

Observe that $\bar{\nu}_p$, the optimized values of $\nu_p(\boldsymbol{\gamma}, \boldsymbol{\beta})$, listed in Table 1 of this paper agree with the values of $\bar{V}_p = \max_{\boldsymbol{\gamma},\boldsymbol{\beta}} V_p(\boldsymbol{\gamma}, \boldsymbol{\beta})$ in Table 1 of Ref. [16]. It turns out that this is true in a general sense:

▶ **Theorem 1.** *For all p and all parameters $(\boldsymbol{\gamma}, \boldsymbol{\beta})$, we have*

$$V_p(\boldsymbol{\gamma}, \boldsymbol{\beta}) = \nu_p(\boldsymbol{\gamma}, \boldsymbol{\beta}). \tag{4.3}$$

The proof of this theorem is provided in Ref. [4, Section 6], where the iteration for ν_p in this paper is carefully mapped to the previously known formula for V_p. This theorem establishes the fact that for each p and fixed parameters, the performance of the QAOA on large-girth D-regular graphs in the $D \to \infty$ limit is *equal* to its performance on the SK model in the $n \to \infty$ limit. We remark that in the iteration in this paper there is only one tree subgraph, with of order D^p vertices, for every large-girth D-regular graph. On the other hand, in the SK case, there is an ensemble of instances given by different weights on the complete graph. It is interesting to us that the ensemble average in Eq. (4.2) can be replaced by a single subgraph.

Theorem 1 also implies that the iteration in Section 3.2 works for evaluating the performance of the QAOA applied to both large-girth regular graphs and the SK model.

5 Conjecture that our iteration achieves the Parisi value

The cut fraction output by the QAOA on MaxCut for large-girth $(D+1)$-regular graphs is

$$\frac{\langle \boldsymbol{\gamma}, \boldsymbol{\beta} | C_{\mathrm{MC}} | \boldsymbol{\gamma}, \boldsymbol{\beta}\rangle}{|E|} = \frac{1}{2} + \frac{\nu_p(D, \boldsymbol{\gamma}, \boldsymbol{\beta})}{\sqrt{D}}. \tag{5.1}$$

We have given an iteration for evaluating $\nu_p(D, \boldsymbol{\gamma}, \boldsymbol{\beta})$ for any depth p and parameters $\boldsymbol{\gamma}, \boldsymbol{\beta}$. Furthermore, in Section 3.2 we give a compact iteration for $\nu_p(\boldsymbol{\gamma}, \boldsymbol{\beta}) = \lim_{D\to\infty} \nu_p(D, \boldsymbol{\gamma}, \boldsymbol{\beta})$. Using this iteration we can optimize over parameters to get $\bar{\nu}_p = \max_{\boldsymbol{\gamma},\boldsymbol{\beta}} \nu_p(\boldsymbol{\gamma}, \boldsymbol{\beta})$. Note $\bar{\nu}_p$ cannot be bigger than the Parisi value, $\Pi_* = \lim_{n\to\infty} \mathbb{E}_J[\max_{\boldsymbol{z}} C_J^{\mathrm{SK}}(\boldsymbol{z})/n]$. From our numerics out to $p = 17$ we see that $\bar{\nu}_p$ is headed in that direction.

Now we make the bold conjecture:

▶ **Conjecture.** *Let $\Pi_* = 0.763166...$ be the Parisi value [25, 27]. Then*

$$\lim_{p\to\infty} \bar{\nu}_p = \Pi_*. \tag{5.2}$$

That is, the iteration in Section 3.2 is an alternative procedure to compute Π_*. To prove this conjecture, perhaps one can show that the iteration in this paper is equivalent to one of the known procedures for computing Π_*. (It may be interesting to note that $\Pi_* = \lim_{k\to\infty} \mathscr{P}_k$, where $\mathscr{P}_k$ is the minimum of the Parisi variational principle over a k-step replica symmetry breaking ansatz with $2k + 1$ parameters [24, 2]. This is not unlike $\bar{\nu}_p$.) Or one can find a way to analytically evaluate the $p \to \infty$ limit.

There is an order of limits issue we now address. For any combinatorial optimization problem of fixed size, the QAOA can be shown to give the optimal solution in the $p \to \infty$ limit [15]. This may require p to grow exponentially in the system size. But we calculate the performance $\bar{\nu}_p$ of the QAOA at fixed p in the $D \to \infty$ limit (which means infinite system size). Then we take $p \to \infty$. Our conjecture is about whether, under this new order of limits, the QAOA achieves the optimum as $p \to \infty$.

6 Generalized iterations for Max-q-XORSAT

It turns out we can easily generalize our iterations for the QAOA's performance on MaxCut in Section 3 to the Max-q-XORSAT problem. MaxCut is a special case of Max-2-XORSAT. Given a q-uniform hypergraph $G = (V, E)$ where $E \subseteq V^q$, and given a signed weight $J_{i_1 i_2 \ldots i_q} \in \{\pm 1\}$ for each edge $(i_1, i_2, \ldots, i_q) \in E$, Max-$q$-XORSAT is the problem of maximizing the following cost function:

$$C_J^{\mathrm{XOR}}(\boldsymbol{z}) = \sum_{(i_1,\ldots,i_q)\in E} \frac{1}{2}(1 + J_{i_1 i_2 \ldots i_q} z_{i_1} z_{i_2} \cdots z_{i_q}). \tag{6.1}$$

This cost function can be understood as counting the number of satisfied clauses, where a clause is satisfied if $z_{i_1} z_{i_2} \cdots z_{i_q} = J_{i_1 i_2 \ldots i_q}$ on the associated edge. Note the MaxCut cost function in Eq. (2.2) is a special case of this problem where $q = 2$ and all $J_{i_1 i_2} = -1$.

We consider this problem on $(D+1)$-regular hypergraphs, where each vertex has degree $D+1$, i.e., it is part of exactly $D+1$ hyperedges. (As in Section 2, working with $(D+1)$-regular hypergraphs means the subgraphs that the QAOA sees are D-ary hypertrees.) The total number of hyperedges is $|E| = n(D+1)/q$, where $n = |V|$ is the number of vertices. Due to a result by Sen [28], we know that with high probability as $n \to \infty$, the maximum fraction of satisfied clauses for a random $(D+1)$-regular hypergraph for sufficiently large D is

$$\frac{1}{|E|} \max_{\boldsymbol{z}} C_J^{\mathrm{XOR}}(\boldsymbol{z}) = \frac{1}{2} + \Pi_q \sqrt{\frac{q}{2D}} + o(1/\sqrt{D}) \tag{6.2}$$

where Π_q is the generalized Parisi value that can be determined explicitly.[2] In particular, $\Pi_2 = \Pi_* = 0.763166\ldots$.

[2] See Ref. [28] for how this value can be calculated. Take care to note that the conventions slightly differ, and our $\Pi_q = \mathsf{P}_q/\sqrt{2}$ where P_q is defined in Section 2.1 of Ref. [28].

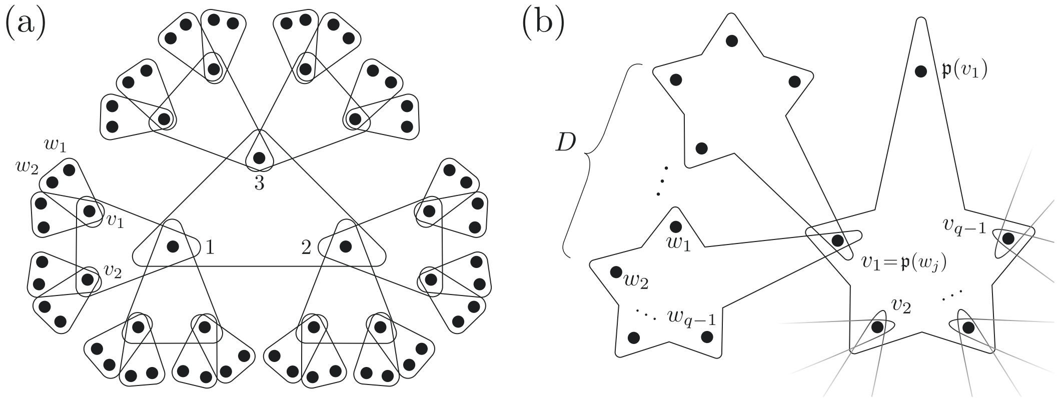


Figure 4 (a) The hypertree subgraph seen by the QAOA at $p = 2$ for the hyperedge $(1, 2, \ldots, q)$ on a $(D + 1)$-regular q-uniform hypergraph with girth $> 2p + 1$, for $q = 3$ and $D = 2$. (b) A partial view near the leaves of the hypertree subgraph for a general q and D. The starfish are hyperedges. Here $w_1, w_2, \ldots, w_{q-1}$ are leaf nodes in the same hyperedge, and we denote their common parent as $v_1 = \mathfrak{p}(w_1) = \cdots = \mathfrak{p}(w_{q-1})$.

We want to evaluate how the QAOA performs on the Max-q-XORSAT problem for large-girth $(D + 1)$-regular hypergraphs. Here, girth is defined as the minimum length of Berge cycles in the hypergraph [7]. Similar to the MaxCut problem discussed in Section 2, we will see that the QAOA has optimal parameters $\boldsymbol{\gamma}$ that are of order $1/\sqrt{D}$ for these graphs. For this reason, it will be convenient to prepare the QAOA state $|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle_J$ with the following shifted and scaled cost function operator

$$C_J = \frac{1}{\sqrt{D}} \sum_{(i_1,\ldots,i_q)\in E} J_{i_1 i_2 \ldots i_q} Z_{i_1} Z_{i_2} \cdots Z_{i_q}. \tag{6.3}$$

For any such hypergraph, we are interested in the fraction of satisfied clauses output by the QAOA at any parameters, for any choices of $J_{i_1 i_2 \ldots i_q}$ drawn from $\{+1, -1\}$. We show the following:

▶ **Theorem 2.** *Consider C_J^{XOR} on any $(D + 1)$-regular q-uniform hypergraphs with girth $> 2p + 1$. Let $|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle_J$ be the QAOA state generated generated using C_J. Then for any choice of J,*

$$\frac{1}{|E|}\, {}_J\langle \boldsymbol{\gamma}, \boldsymbol{\beta}| C_J^{\mathrm{XOR}} |\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle_J = \frac{1}{2} + \nu_p^{[q]}(D, \boldsymbol{\gamma}, \boldsymbol{\beta}) \sqrt{\frac{q}{2D}} \tag{6.4}$$

where $\nu_p^{[q]}(D, \boldsymbol{\gamma}, \boldsymbol{\beta})$ is independent of J and can be evaluated (classically) with an iteration using $O(p4^{pq})$ time and $O(4^p)$ memory. In the infinite D limit, $\lim_{D\to\infty} \nu_p^{[q]}(D, \boldsymbol{\gamma}, \boldsymbol{\beta})$ can be evaluated with an iteration using $O(p^2 4^p)$ time and $O(p^2)$ memory.

The full proof can be found in Ref. [4, Section 8], where we also describe iterations for $\nu_p^{[q]}$ in detail. It is based on the same idea as the iterations in Section 3, as we exploit the regularity of the hypertree subgraph seen by the QAOA on these hypergraphs.

In the next section, we give a part the proof that shows the J-independence $\nu_p^{[q]}$, and discuss its implication of a worst-case limitation on the QAOA's performance. In Section 6.2 that follows, we describe the infinite-D iteration and present results from its numerical evaluation.

6.1 J-independence of $\nu_p^{[q]}$ and implied worst-case limitation

We argue that the left hand side of Eq. (6.4) is independent of the choice of J's, so there is no J needed on the right hand side. When the girth of the hypergraph is larger than $2p+1$, the subgraph seen by the QAOA on any hyperedge is always a D-ary q-uniform hypertree. See Figure 4(a) for an example. In this figure each triangle is associated with a coupling J that can be either $+1$ or -1. Look at the triangle containing vertices $1, 2$ and 3. We can absorb the sign of J_{123} into the bit at vertex 1 as follows: if $J_{123} = -1$ do nothing, whereas if $J_{123} = +1$ flip the sign of the bit at vertex 1 by redefining $Z_1 \to -Z_1$. Then $J_{123}Z_1Z_2Z_3 \to -Z_1Z_2Z_3$ under this transformation. Now look at the triangle containing bits 1, v_1 and v_2. The sign of $J_{1v_1v_2}$ may have been modified by the last step. But we can now absorb the sign of $J_{1v_1v_2}$ into the bit at v_1 so that $J_{1v_1v_2}Z_1Z_{v_1}Z_{v_2} \to -Z_1Z_{v_1}Z_{v_2}$. This might affect the sign of $J_{v_1w_1w_2}$ in the triangle containing v_1, w_1 and w_2. But we can redefine the bit at w_1 appropriately so that $J_{v_1w_1w_2}Z_{v_1}Z_{w_1}Z_{w_2} \to -Z_{v_1}Z_{w_1}Z_{w_2}$. Since there are no cycles in the hypertree, we can move through the whole picture in this way resetting all the couplings J to -1.

We have reset all the couplings J to -1 in the picture, and we now argue that this makes the quantum expectation (6.4) independent of the J's. At the quantum level we flip the sign of the operator Z_u by conjugating with X_u, that is, $X_uZ_uX_u = -Z_u$. Since the driver B commutes with each X_u and the initial state is an eigenstate of each X_u, we can sprinkle X_u's into the left hand side of Eq. (6.4) and establish the J-independence of the expression coming from any particular hyperedge. Now the cost function (6.1) is a sum over the hyperedges of a given hypergraph, but the expected value of each term in the QAOA state is independent of the J's. So for every $(D+1)$-regular q-uniform hypergraph with girth $> 2p+1$ we can write

$$\frac{1}{|E|}\, {}_J\langle \boldsymbol{\gamma},\boldsymbol{\beta}|C_J^{\mathrm{XOR}}|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle_J = \frac{1}{2} - \frac{1}{2}\langle \boldsymbol{\gamma},\boldsymbol{\beta}|Z_1Z_2\ldots Z_q|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle \tag{6.5}$$

where $(1,2,3,\ldots,q)$ is any hyperedge, and the state $|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle$ without the J label has all the couplings set to -1.

A corollary to this J-independence is that the QAOA at low depth fails to find the optimal assignment in the worst case. To see this, let us go back to the $q=2$ case where we studied MaxCut on a large-girth regular graph which has all of the couplings $J=-1$. At optimal parameters, the fraction of satisfied clauses is $1/2 + \bar{\nu}_p/\sqrt{D}$ in the large D limit, where $\bar{\nu}_p \le \Pi_*$. Consider the corresponding instance where all the couplings on the same graph are set to $J=+1$, which makes the instance fully satisfiable. In that case, the best possible fraction of satisfied clauses is 1. However, the fraction output by the QAOA at optimal parameters is the same as in the $J=-1$ case, that is, at most $1/2 + \Pi_*/\sqrt{D}$, which is only a bit more than $1/2$ in the large D limit.

Here we have an example of the QAOA failing to reach the optimum in the worst case because it does not "see" the whole graph. (Unlike previous results of the similar flavor in Refs. [9, 14], we do not need the graph to be bipartite to bound the worst-case approximation ratio.) Regardless of the signs of the couplings, the low-depth QAOA sees a tree subgraph surrounding each edge. On the tree subgraph the signs of the couplings are irrelevant so the QAOA does not distinguish between instances where the cost function favors disagreement and instances where agreement is favored. Without seeing cycles the QAOA cannot do better than what it can achieve in the most frustrated case, and this yields an upper bound on the worst-case approximation ratio.

6.2 The infinite-D iteration for $\nu_p^{[q]}$

We now describe the iteration mentioned in Theorem 2 for the $D \to \infty$ limit. Similar to Section 3.2, we define matrices $G^{(m)} \in \mathbb{C}^{(2p+1)\times(2p+1)}$, for $0 \le m \le p$ as follows. For $j,k \in \{1,\dots,p,0,-p,\dots,-1\}$, let $G^{(0)}_{j,k} = \sum_{\boldsymbol{a}} f(\boldsymbol{a}) a_j a_k$, and

$$G^{(m)}_{j,k} = \sum_{\boldsymbol{a}} f(\boldsymbol{a}) a_j a_k \exp\left[-\frac{1}{2}\sum_{j',k'=-p}^{p} \left(G^{(m-1)}_{j',k'}\right)^{q-1} \Gamma_{j'}\Gamma_{k'} a_{j'} a_{k'}\right] \quad \text{for } 1 \le m \le p. \tag{6.6}$$

Starting at $m = 0$ and going up by p steps we arrive at $G^{(p)}$ which is used to compute

$$\nu_p^{[q]}(\boldsymbol{\gamma},\boldsymbol{\beta}) := \lim_{D\to\infty} \nu_p^{[q]}(D,\boldsymbol{\gamma},\boldsymbol{\beta}) = \frac{i}{\sqrt{2q}} \sum_{j=-p}^{p} \Gamma_j (G^{(p)}_{0,j})^q. \tag{6.7}$$

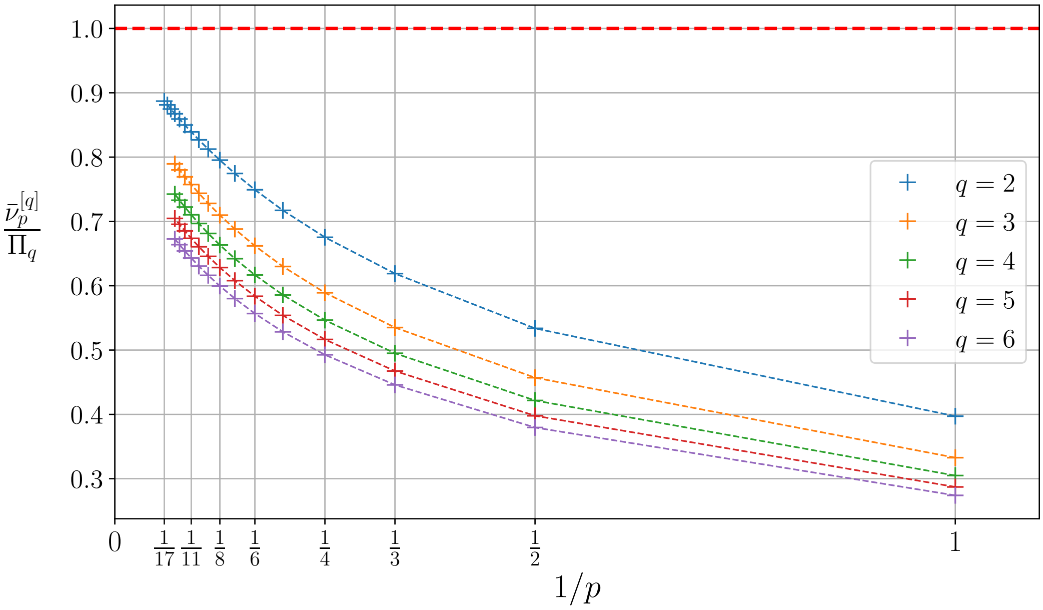


Figure 5 Optimal values $\bar{\nu}_p^{[q]}$ normalized by their corresponding Parisi values Π_q as a function of $1/p$ for $q = 2,3,4,5,6$. The Parisi values are taken from Ref. [22]. Similar to Fig. 2, we made the somewhat arbitrary choice of plotting the data against $1/p$ to see the large p region in a compact plot. Dashed lines in between data points are intended to guide the eye.

Note the only difference between Max-q-XORSAT and MaxCut, where $q = 2$, can be seen by comparing Eqs. (3.8) and (3.9) in Section 3.2 to Eqs. (6.6) and (6.7) in the current iteration, where we are raising the matrix elements of G to some q-dependent power. Hence, this iteration also takes at most $O(p^2 4^p)$ time and $O(p^2)$ memory to be evaluated, regardless of q. This is polynomially faster than the finite D case with exponentially better memory usage.

We take this iteration and numerically optimize over $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ to find

$$\bar{\nu}_p^{[q]} = \max_{\boldsymbol{\gamma},\boldsymbol{\beta}} \lim_{D\to\infty} \nu_p^{[q]}(D,\boldsymbol{\gamma},\boldsymbol{\beta}). \tag{6.8}$$

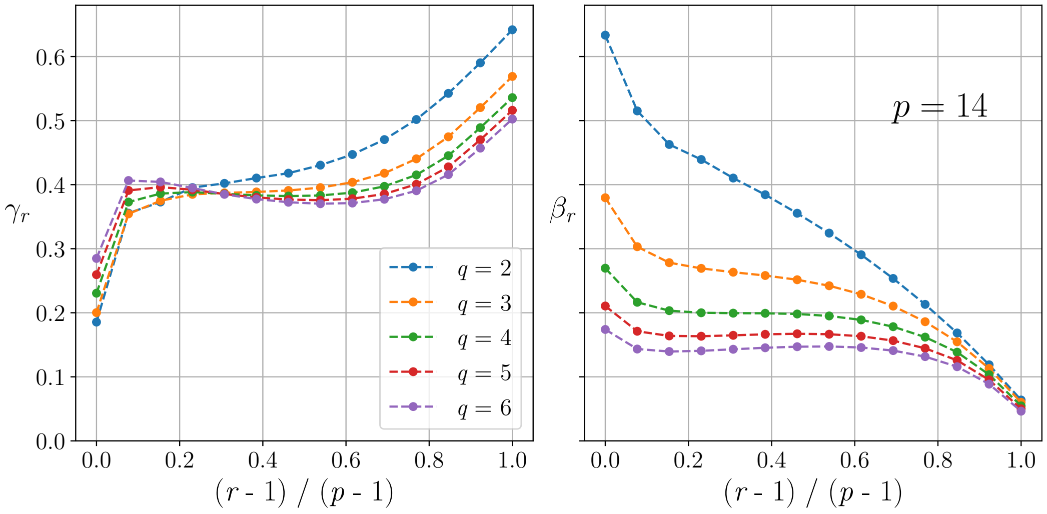


Figure 6 Optimal QAOA parameters $(\boldsymbol{\gamma}, \boldsymbol{\beta})$ at $p = 14$ for various Max-q-XORSAT on D-regular hypergraphs in the $D \to \infty$ limit. This data can be found in Ref. [5].

up to $p = 14$ for $3 \leq q \leq 6$. Combining with the data we have for $q = 2$ in Table 1, we plot the results in Figure 5. For ease of comparison across different values of q we have normalized $\bar{\nu}_p^{[q]}$ by its corresponding Parisi value Π_q. See Figure 6 for a plot of the optimal $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ we found at $p = 14$. Numerical values for $\bar{\nu}_p^{[q]}$ and optimal $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ for all $1 \leq p \leq 14$ can be found in Ref. [5].

In some cases, there are thresholds on how well the QAOA at low depths can do. It is known that for problems that exhibit the overlap gap property, the locality property of the QAOA prevents it from getting close to the optimum at low depths where it does not see the whole graph [13, 10]. Specifically, using an overlap gap property in the Max-q-XORSAT problem on random Erdős-Rényi hypergraphs with constant average degree and even $q \geq 4$, Ref. [10] showed that the QAOA (or any local algorithm) has limited performance when the depth p is less than $\epsilon \log n$, where n is the graph size and ϵ is a constant that depends on the degree and q. Assuming the overlap gap property also holds when the hypergraphs are regular, one can use similar arguments to show that the QAOA's performance as measured by $\bar{\nu}_p^{[q]}/\Pi_q$ does not converge to 1 as $p \to \infty$ when $q \geq 4$ and is even. This is because our large-girth assumption implies the graph has at least D^p vertices, so p is always less than $\epsilon \log n$ in this limit.

7 Discussion

In this paper, we have introduced new techniques for evaluating the performance of a quantum algorithm at high qubit number and at high depth. In particular we do this by finding a compact iteration for the QAOA's performance on MaxCut on instances with locally tree-like neighborhoods. On random large-girth D-regular graphs, the QAOA at $p = 11$ and higher has the highest approximation ratio of any assumption-free algorithm. We have given performance guarantees for the QAOA, but it is necessary to run a quantum computer to produce a string with the calculated performance.

We have also shown that for any depth p and for any parameters, $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$, the performance of the QAOA on large-girth D-regular graphs, as $D \to \infty$, matches the typical performance of the QAOA on the Sherrington-Kirkpatrick model at infinite size. We find it remarkable that the ensemble averaging done in the SK model can be replaced by analyzing a single tree subgraph. For both of these models the best conceivable performance is upperbounded by the Parisi constant, Π_*. There are optimal parameters at each p, and we speculate that as $p \to \infty$ these optimal parameters give QAOA performance that matches the Parisi constant for both models.

Moreover, in Section 6, we have generalized our iteration for MaxCut on large-girth regular graphs to evaluate the QAOA's performance on Max-q-XORSAT problems for large-girth regular hypergraphs. We have shown that, at fixed parameters, the QAOA gives the same value of the objective function regardless of the signs of the couplings on these hypergraphs. This implies a worst-case algorithmic threshold at low depth for fully satisfiable instances. Building on our work, Ref. [6] recently generalized the equivalence between MaxCut and the SK model to between Max-q-XORSAT and the fully connected q-spin model.

There are a number of ideas to explore coming out of this work. Can we find a more efficient iterative formula for the QAOA's performance than the one in Section 3.2? If so, we can better probe the large-p behavior of the QAOA. Can the iteration in Section 3.2 be recast in the $p \to \infty$ limit in terms of continuous functions corresponding to $\boldsymbol{\gamma}, \boldsymbol{\beta}$? This might be a way to verify, or falsify, the conjecture in Section 5.

Can one find other problems at high qubit number and high depth where the performance of the QAOA can be established using techniques similar to the ones introduced in this paper?

References

1 Ahmed El Alaoui, Andrea Montanari, and Mark Sellke. Local algorithms for Maximum Cut and Minimum Bisection on locally treelike regular graphs of large degree. *arXiv preprint*, 2021. arXiv:2111.06813.

2 Antonio Auffinger, Wei-Kuo Chen, and Qiang Zeng. The SK Model Is Infinite Step Replica Symmetry Breaking at Zero Temperature. *Communications on Pure and Applied Mathematics*, 73(5):921–943, 2020. doi:10.1002/cpa.21886.

3 Boaz Barak and Kunal Marwaha. Classical Algorithms and Quantum Limitations for Maximum Cut on High-Girth Graphs. *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, 215:14:1–14:21, 2022. doi:10.4230/LIPIcs.ITCS.2022.14.

4 Joao Basso, Edward Farhi, Kunal Marwaha, Benjamin Villalonga, and Leo Zhou. The Quantum Approximate Optimization Algorithm at High Depth for MaxCut on Large-Girth Regular Graphs and the Sherrington-Kirkpatrick Model. *arXiv preprint*, 2021. Full version on arXiv. arXiv:2110.14206.

5 Joao Basso, Edward Farhi, Kunal Marwaha, Benjamin Villalonga, and Leo Zhou. Performance of the QAOA on MaxCut over Large-Girth Regular Graphs. Online, 2022. URL: https://github.com/benjaminvillalonga/large-girth-maxcut-qaoa.

6 Joao Basso, David Gamarnik, Song Mei, and Leo Zhou. Performance and limitations of the QAOA at constant levels on large sparse hypergraphs and spin glass models. *arXiv preprint*, 2022. arXiv:2204.10306.

7 Claude Berge. *Hypergraphs, Combinatorics of Finite Sets.* North-Holland, 1989. URL: http://compalg.inf.elte.hu/t̃ony/Oktatas/Algoritmusok-hatekonysaga/Berge-hypergraphs.pdf.

8 Sami Boulebnane and Ashley Montanaro. Predicting parameters for the Quantum Approximate Optimization Algorithm for MAX-CUT from the infinite-size limit. *arXiv preprint*, 2021. arXiv:2110.10685.

9 Sergey Bravyi, Alexander Kliesch, Robert Koenig, and Eugene Tang. Obstacles to variational quantum optimization from symmetry protection. *Phys. Rev. Lett.*, 125:260505, December 2020. doi:10.1103/PhysRevLett.125.260505.

10 Chi-Ning Chou, Peter J. Love, Juspreet Singh Sandhu, and Jonathan Shi. Limitations of Local Quantum Algorithms on Random Max-k-XOR and Beyond. *arXiv preprint*, 2021. arXiv:2108.06049.

11 Leonardo Dagum and Ramesh Menon. OpenMP: an industry standard API for shared-memory programming. *Computational Science & Engineering, IEEE*, 5(1):46–55, 1998. doi:10.1109/99.660313.

12 Amir Dembo, Andrea Montanari, and Subhabrata Sen. Extremal cuts of sparse random graphs. *The Annals of Probability*, 45(2):1190–1217, 2017. doi:10.1214/15-AOP1084.

13 Edward Farhi, David Gamarnik, and Sam Gutmann. The Quantum Approximate Optimization Algorithm Needs to See the Whole Graph: A Typical Case. *arXiv preprint*, 2020. arXiv:2004.09002.

14 Edward Farhi, David Gamarnik, and Sam Gutmann. The Quantum Approximate Optimization Algorithm Needs to See the Whole Graph: Worst Case Examples. *arXiv preprint*, 2020. arXiv:2005.08747.

15 Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A Quantum Approximate Optimization Algorithm. *arXiv preprint*, 2014. arXiv:1411.4028.

16 Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Leo Zhou. The Quantum Approximate Optimization Algorithm and the Sherrington-Kirkpatrick Model at Infinite Size. *arXiv preprint*, 2019. arXiv:1910.08187.

17 David Gamarnik. The overlap gap property: A topological barrier to optimizing over random structures. *Proceedings of the National Academy of Sciences*, 118(41), 2021. doi:10.1073/pnas.2108492118.

18 Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. Online, 2010. URL: https://eigen.tuxfamily.org.

19 Matthew P Harrigan, Kevin J Sung, Matthew Neeley, Kevin J Satzinger, Frank Arute, Kunal Arya, Juan Atalaya, Joseph C Bardin, Rami Barends, Sergio Boixo, et al. Quantum approximate optimization of non-planar graph problems on a planar superconducting processor. *Nature Physics*, 17(3):332–336, 2021. doi:10.1038/s41567-020-01105-y.

20 Russell Lyons. Factors of iid on trees. *Combinatorics, Probability and Computing*, 26(2):285–300, 2017. arXiv:1401.4197.

21 Kunal Marwaha. Local classical MAX-CUT algorithm outperforms $p = 2$ QAOA on high-girth regular graphs. *Quantum*, 5:437, 2021. doi:10.22331/q-2021-04-20-437.

22 Kunal Marwaha and Stuart Hadfield. Bounds on approximating Max kXOR with quantum and classical local algorithms. *arXiv preprint*, 2021. arXiv:2109.10833.

23 Andrea Montanari and Subhabrata Sen. Semidefinite programs on sparse random graphs and their application to community detection. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '16, page 814–827, 2016. doi:10.1145/2897518.2897548.

24 Dmitry Panchenko. *The Sherrington-Kirkpatrick model.* Springer Science & Business Media, 2013. doi:10.1007/978-1-4614-6289-7.

25 Giorgio Parisi. Toward a mean field theory for spin glasses. *Physics Letters A*, 73(3):203–205, 1979. doi:10.1016/0375-9601(79)90708-4.

26 Yixuan Qiu and Dirk Toewe. LBFGS++. Online, 2020. URL: https://github.com/yixuan/LBFGSpp.

27 Manuel J Schmidt. *Replica symmetry breaking at low temperatures.* PhD thesis, Universität Würzburg, 2008. URL: https://d-nb.info/991972910/34.

28 Subhabrata Sen. Optimization on sparse random hypergraphs and spin glasses. *Random Structures & Algorithms*, 53(3):504–536, 2018. doi:10.1002/rsa.20774.

29 Jessica K Thompson, Ojas Parekh, and Kunal Marwaha. An explicit vector algorithm for high-girth MaxCut. *Symposium on Simplicity in Algorithms (SOSA)*, pages 238–246, 2022. doi:10.1137/1.9781611977066.17.

30 Luca Trevisan, Gregory B Sorkin, Madhu Sudan, and David P Williamson. Gadgets, approximation, and linear programming. *SIAM Journal on Computing*, 29(6):2074–2097, 2000. doi:10.1137/S0097539797328847.

31 Zhihui Wang, Stuart Hadfield, Zhang Jiang, and Eleanor G Rieffel. Quantum approximate optimization algorithm for MaxCut: A fermionic view. *Phys. Rev. A*, 97(2):022304, 2018. doi:10.1103/PhysRevA.97.022304.

32 Jonathan Wurtz and Peter Love. MaxCut quantum approximate optimization algorithm performance guarantees for $p > 1$. *Phys. Rev. A*, 103:042612, April 2021. doi:10.1103/PhysRevA.103.042612.

33 Leo Zhou, Sheng-Tao Wang, Soonwon Choi, Hannes Pichler, and Mikhail D. Lukin. Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices. *Phys. Rev. X*, 10:021067, 2020. doi:10.1103/PhysRevX.10.021067.

A Proof of the iterations for MaxCut

In this appendix, we prove the correctness of the two iterations in Section 3.1 and Section 3.2. We hope this proof illustrates two key technical ideas in this paper: namely, we can exploit the regularity of the tree subgraph seen by the QAOA to yield a compact formula for its performance, and we find an algebraic simplification in the $D \to \infty$ limit. This appendix also serves as the proof of a special case of Theorem 2 at $q = 2$. The remaining proofs of our results can be found in the full version of this paper at Ref. [4].

A.1 Proof of the finite D iteration

We start by proving the finite D iteration that was stated in Section 3.1. We focus on the iteration for $p = 2$ as an example, and its generalization to other p is immediate.

The goal is to evaluate the energy expectation for a single edge (L, R) on a $(D+1)$-regular graph whose girth is larger than $2p+1$. For $p = 2$, this is

$$\langle \boldsymbol{\gamma}, \boldsymbol{\beta} | Z_L Z_R | \boldsymbol{\gamma}, \boldsymbol{\beta} \rangle = \langle s | e^{i\gamma_1 C} e^{i\beta_1 B} e^{i\gamma_2 C} e^{i\beta_2 B} Z_L Z_R e^{-i\beta_2 B} e^{-i\gamma_2 C} e^{-i\beta_1 B} e^{-i\gamma_1 C} | s \rangle \tag{A.1}$$

where $C = -(1/\sqrt{D}) \sum_{(u,v)\in E} Z_u Z_v$, and E denotes the set of edges for the given graph. In the Heisenberg picture, it can be seen that the operator $e^{i\gamma_1 C} \cdots e^{i\beta_p B} Z_L Z_R e^{-i\beta_p B} \cdots e^{-i\gamma_1 C}$ only acts nontrivially on the subgraph induced by including all vertices distance p or less from either node L or R. For a $(D+1)$-regular graph with girth greater than $2p+1$, this subgraph looks like a pair of D-ary trees that are glued at their roots (see Figure 7), with a total of $n = 2(D^p + \cdots + D + 1)$ nodes. In what follows, we compute Eq. (A.1) by restricting our attention to only the qubits in this subgraph.

We start by inserting 5 complete sets in the computational Z-basis that we will label as $\boldsymbol{z}^{[1]}, \boldsymbol{z}^{[2]}, \boldsymbol{z}^{[0]}, \boldsymbol{z}^{[-2]}$, and $\boldsymbol{z}^{[-1]}$. Each of these complete sets iterates over 2^n basis states since the number of qubits in the subgraph is n. Then

$$\begin{aligned}
\langle \boldsymbol{\gamma}, \boldsymbol{\beta} | Z_L Z_R | \boldsymbol{\gamma}, \boldsymbol{\beta} \rangle = & \sum_{\{\boldsymbol{z}^{[i]}\}} \langle s | \boldsymbol{z}^{[1]} \rangle \, e^{i\gamma_1 C(\boldsymbol{z}^{[1]})} \, \langle \boldsymbol{z}^{[1]} | e^{i\beta_1 B} | \boldsymbol{z}^{[2]} \rangle \, e^{i\gamma_2 C(\boldsymbol{z}^{[2]})} \, \langle \boldsymbol{z}^{[2]} | e^{i\beta_2 B} | \boldsymbol{z}^{[0]} \rangle \, z_L^{[0]} z_R^{[0]} \\
& \times \langle \boldsymbol{z}^{[0]} | e^{-i\beta_2 B} | \boldsymbol{z}^{[-2]} \rangle \, e^{-i\gamma_2 C(\boldsymbol{z}^{[-2]})} \, \langle \boldsymbol{z}^{[-2]} | e^{-i\beta_1 B} | \boldsymbol{z}^{[-1]} \rangle \, e^{-i\gamma_1 C(\boldsymbol{z}^{[-1]})} \, \langle \boldsymbol{z}^{[-1]} | s \rangle \\
= & \frac{1}{2^n} \sum_{\{\boldsymbol{z}^{[i]}\}} \exp \left[i\gamma_1 C(\boldsymbol{z}^{[1]}) + i\gamma_2 C(\boldsymbol{z}^{[2]}) - i\gamma_2 C(\boldsymbol{z}^{[-2]}) - i\gamma_1 C(\boldsymbol{z}^{[-1]}) \right] z_L^{[0]} z_R^{[0]} \\
& \times \prod_{v=1}^{n} \langle z_v^{[1]} | e^{i\beta_1 X} | z_v^{[2]} \rangle \, \langle z_v^{[2]} | e^{i\beta_2 X} | z_v^{[0]} \rangle \, \langle z_v^{[0]} | e^{-i\beta_2 X} | z_v^{[-2]} \rangle \, \langle z_v^{[-2]} | e^{-i\beta_1 X} | z_v^{[-1]} \rangle \, .
\end{aligned} \tag{A.2}$$

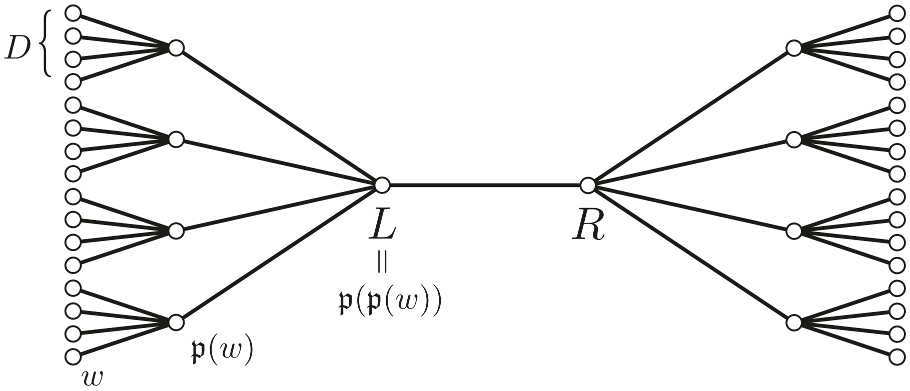


Figure 7 The tree subgraph seen by the QAOA at $p = 2$ for the edge (L, R) on a $(D+1)$-regular graph with girth $> 2p+1$. For any node v on either of the D-ary trees we denote $\mathfrak{p}(v)$ as the parent of that node. In the figure w is a leaf node, and we show its parent and its parent's parent.

Let us define the following function which is the $p = 2$ version of Eq. (3.4):

$$f(a_1, a_2, a_0, a_{-2}, a_{-1}) = \frac{1}{2} \langle a_1|e^{i\beta_1 X}|a_2\rangle \langle a_2|e^{i\beta_2 X}|a_0\rangle \langle a_0|e^{-i\beta_2 X}|a_{-2}\rangle \langle a_{-2}|e^{-i\beta_1 X}|a_{-1}\rangle \,. \tag{A.3}$$

Then, using $\mathbf{\Gamma}$ as defined in Eq. (3.3), we can rewrite Eq. (A.2) as

$$\langle \boldsymbol{\gamma}, \boldsymbol{\beta}|Z_L Z_R|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle = \sum_{\{\boldsymbol{z}^{[i]}\}} z_L^{[0]} z_R^{[0]} \exp\left[i \sum_{j=-2}^{2} \Gamma_j C(\boldsymbol{z}^{[j]})\right] \prod_{v=1}^{n} f(\boldsymbol{z}_v) \tag{A.4}$$

where $\boldsymbol{z}_v = (z_v^{[1]}, z_v^{[2]}, z_v^{[0]}, z_v^{[-2]}, z_v^{[-1]})$ are the bits from the 5 complete sets associated with node v. Using the fact that $C(\boldsymbol{z}) = -(1/\sqrt{D}) \sum_{(u,v)\in E} z_u z_v$, we can rewrite $\langle \boldsymbol{\gamma}, \boldsymbol{\beta}|Z_L Z_R|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle$ as

$$\langle \boldsymbol{\gamma}, \boldsymbol{\beta}|Z_L Z_R|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle = \sum_{\{\boldsymbol{z}_u\}} z_L^{[0]} z_R^{[0]} \exp\left[-\tfrac{i}{\sqrt{D}} \sum_{(u',v')\in E} \mathbf{\Gamma} \cdot (\boldsymbol{z}_{u'} \boldsymbol{z}_{v'})\right] \prod_{v=1}^{n} f(\boldsymbol{z}_v) \tag{A.5}$$

where we have replaced the sum over the 5 complete sets $\{\boldsymbol{z}^{[i]} : -2 \le i \le 2\}$ with an equivalent sum over the bit configurations of each node $\{\boldsymbol{z}_u : 1 \le u \le n\}$. Now to evaluate $\langle Z_L Z_R\rangle$ we need to perform a sum over the bit configurations $\boldsymbol{z}_v$ of every node v in the tree subgraph, where each node is coupled to its neighbors on the graph via the term in the exponential of Eq. (A.5).

We can start by considering a single leaf node w who is only connected to its parent node $\mathfrak{p}(w)$ on the tree, as shown in Figure 7. Then the sum over the 32 bit values of the configuration $\boldsymbol{z}_w = (z_w^{[1]}, z_w^{[2]}, z_w^{[0]}, z_w^{[-2]}, z_w^{[-1]})$ yields

$$\sum_{\boldsymbol{z}_w} f(\boldsymbol{z}_w) \exp\left[-\tfrac{i}{\sqrt{D}} \mathbf{\Gamma} \cdot (\boldsymbol{z}_w \boldsymbol{z}_{\mathfrak{p}(w)})\right] \tag{A.6}$$

which is a function of the parent node's configuration $\boldsymbol{z}_{\mathfrak{p}(w)}$. Note that doing this on every leaf node contributes the same function to its parent. Since there are exactly D leaf nodes per parent, we get the following contribution

$$H_D^{(1)}(\boldsymbol{z}_{\mathfrak{p}(w)}) := \left(\sum_{\boldsymbol{z}_w} f(\boldsymbol{z}_w) \exp\left[-\tfrac{i}{\sqrt{D}} \boldsymbol{\Gamma} \cdot (\boldsymbol{z}_w \boldsymbol{z}_{\mathfrak{p}(w)}) \right] \right)^D . \tag{A.7}$$

This is true for every parent node of any of the leaves.

After performing the sums for all the leaf nodes, we can move to the sums for their parents. Let us look at the sum on the node $\mathfrak{p}(w)$ for example, which yields

$$\sum_{\boldsymbol{z}_{\mathfrak{p}(w)}} f(\boldsymbol{z}_{\mathfrak{p}(w)}) H_D^{(1)}(\boldsymbol{z}_{\mathfrak{p}(w)}) \exp\left[-\tfrac{i}{\sqrt{D}} \boldsymbol{\Gamma} \cdot (\boldsymbol{z}_{\mathfrak{p}(w)} \boldsymbol{z}_{\mathfrak{p}(\mathfrak{p}(w))}) \right]. \tag{A.8}$$

Again, because its parent node $\mathfrak{p}(\mathfrak{p}(w))$ has D identical children like $\mathfrak{p}(w)$, this yields

$$H_D^{(2)}(\boldsymbol{z}_{\mathfrak{p}(\mathfrak{p}(w))}) := \left(\sum_{\boldsymbol{z}_{\mathfrak{p}(w)}} f(\boldsymbol{z}_{\mathfrak{p}(w)}) H_D^{(1)}(\boldsymbol{z}_{\mathfrak{p}(w)}) \exp\left[-\tfrac{i}{\sqrt{D}} \boldsymbol{\Gamma} \cdot (\boldsymbol{z}_{\mathfrak{p}(w)} \boldsymbol{z}_{\mathfrak{p}(\mathfrak{p}(w))}) \right] \right)^D . \tag{A.9}$$

Note at $p = 2$ we have reached the root of the tree $L = \mathfrak{p}(\mathfrak{p}(w))$ after these two iterations.

To evaluate $\langle \boldsymbol{\gamma}, \boldsymbol{\beta} | Z_L Z_R | \boldsymbol{\gamma}, \boldsymbol{\beta} \rangle$, it only remains to sum over the 5 bits in $\boldsymbol{z}_L$ and the 5 bits in $\boldsymbol{z}_R$:

$$\langle \boldsymbol{\gamma}, \boldsymbol{\beta} | Z_L Z_R | \boldsymbol{\gamma}, \boldsymbol{\beta} \rangle = \sum_{\boldsymbol{z}_L, \boldsymbol{z}_R} z_L^{[0]} z_R^{[0]} f(\boldsymbol{z}_L) f(\boldsymbol{z}_R) H_D^{(2)}(\boldsymbol{z}_L) H_D^{(2)}(\boldsymbol{z}_R) \exp\left[-\tfrac{i}{\sqrt{D}} \boldsymbol{\Gamma} \cdot (\boldsymbol{z}_L \boldsymbol{z}_R) \right]. \tag{A.10}$$

For higher p, we can see that the evaluation of $\langle \boldsymbol{\gamma}, \boldsymbol{\beta} | Z_L Z_R | \boldsymbol{\gamma}, \boldsymbol{\beta} \rangle$ simply involves more iterations of Eq. (A.9) corresponding to more levels in the tree subgraph. In summary, the iteration for general p can be written as starting with

$$H_D^{(0)}(\boldsymbol{a}) = 1 \tag{A.11}$$

and then evaluating for $m = 1, 2, \ldots, p$,

$$H_D^{(m)}(\boldsymbol{a}) = \left(\sum_{\boldsymbol{b}} f(\boldsymbol{b}) H_D^{(m-1)}(\boldsymbol{b}) \exp\left[-\tfrac{i}{\sqrt{D}} \boldsymbol{\Gamma} \cdot (\boldsymbol{a}\boldsymbol{b}) \right] \right)^D , \tag{A.12}$$

since there are p levels in the tree subgraph seen by the QAOA with p layers. At the end we get

$$\langle \boldsymbol{\gamma}, \boldsymbol{\beta} | Z_L Z_R | \boldsymbol{\gamma}, \boldsymbol{\beta} \rangle = \sum_{\boldsymbol{a}, \boldsymbol{b}} a_0 b_0 f(\boldsymbol{a}) f(\boldsymbol{b}) H_D^{(p)}(\boldsymbol{a}) H_D^{(p)}(\boldsymbol{b}) \exp\left[-\tfrac{i}{\sqrt{D}} \boldsymbol{\Gamma} \cdot (\boldsymbol{a}\boldsymbol{b}) \right]. \tag{A.13}$$

This is almost what we have stated for the iteration in Section 3.1.

To finish the proof, we note from Eq. (A.3) as well as its general p version in Eq. (3.4) that

$$f(-\boldsymbol{a}) = f(\boldsymbol{a}). \tag{A.14}$$

We now claim that

$$H_D^{(m)}(-\boldsymbol{a}) = H_D^{(m)}(\boldsymbol{a}) \quad \text{for } 0 \le m \le p \tag{A.15}$$

which we will show by induction on m. Note this is trivially true for the base case $m = 0$ since $H_D^{(0)}(\boldsymbol{a}) = 1$ is constant. Assuming that $H_D^{(m-1)}(-\boldsymbol{a}) = H_D^{(m-1)}(\boldsymbol{a})$, we can take $\boldsymbol{b} \to -\boldsymbol{b}$ in the summand of Eq. (A.12) and combine it with its original form to see that

$$H_D^{(m)}(\boldsymbol{a}) = \left(\sum_{\boldsymbol{b}} f(\boldsymbol{b}) H_D^{(m-1)}(\boldsymbol{b}) \cos \left[\tfrac{1}{\sqrt{D}} \boldsymbol{\Gamma} \cdot (\boldsymbol{ab}) \right] \right)^D. \tag{A.16}$$

From this form it follows that $H_D^{(m)}(-\boldsymbol{a}) = H_D^{(m)}(\boldsymbol{a})$ since $\boldsymbol{a}$ only appears in the cosine which is an even function, establishing Eq. (A.15).

Similarly, we can take $\boldsymbol{b} \to -\boldsymbol{b}$ in Eq. (A.13) and combine with its original form to get

$$\langle \boldsymbol{\gamma}, \boldsymbol{\beta} | Z_L Z_R | \boldsymbol{\gamma}, \boldsymbol{\beta} \rangle = -i \sum_{\boldsymbol{a},\boldsymbol{b}} a_0 b_0 f(\boldsymbol{a}) f(\boldsymbol{b}) H_D^{(p)}(\boldsymbol{a}) H_D^{(p)}(\boldsymbol{b}) \sin \left[\tfrac{1}{\sqrt{D}} \boldsymbol{\Gamma} \cdot (\boldsymbol{ab}) \right]. \tag{A.17}$$

Thus to get the ν_p as defined in Eq. (2.6) that tells us the cut fraction, we have

$$\nu_p(D, \boldsymbol{\gamma}, \boldsymbol{\beta}) = \frac{i\sqrt{D}}{2} \sum_{\boldsymbol{a},\boldsymbol{b}} a_0 b_0 f(\boldsymbol{a}) f(\boldsymbol{b}) H_D^{(p)}(\boldsymbol{a}) H_D^{(p)}(\boldsymbol{b}) \sin \left[\tfrac{1}{\sqrt{D}} \boldsymbol{\Gamma} \cdot (\boldsymbol{ab}) \right]. \tag{A.18}$$

This proves our iteration for any finite D in Section 3.1.

A.2 Proof of $D \to \infty$ iteration

We wish to evaluate Eq. (3.7) in the $D \to \infty$ limit:

$$\lim_{D\to\infty} \nu_p(D, \boldsymbol{\gamma}, \boldsymbol{\beta}) = \lim_{D\to\infty} \frac{i\sqrt{D}}{2} \sum_{\boldsymbol{a},\boldsymbol{b}} a_0 b_0 f(\boldsymbol{a}) f(\boldsymbol{b}) H_D^{(p)}(\boldsymbol{a}) H_D^{(p)}(\boldsymbol{b}) \sin \left[\tfrac{1}{\sqrt{D}} \boldsymbol{\Gamma} \cdot (\boldsymbol{ab}) \right]. \tag{A.19}$$

We first prove by induction that for $0 \le m \le p$,

$$H^{(m)}(\boldsymbol{a}) := \lim_{D\to\infty} H_D^{(m)}(\boldsymbol{a}) \tag{A.20}$$

exists and is finite. For $m = 0$, our claim holds because $H_D^{(0)}(\boldsymbol{a}) = 1$. Assuming the claim is true for $m - 1$, we examine $H^{(m)}(\boldsymbol{a})$ by taking the limit on Eq. (A.16)

$$H^{(m)}(\boldsymbol{a}) = \lim_{D\to\infty} \left[\sum_{\boldsymbol{b}} f(\boldsymbol{b}) H_D^{(m-1)}(\boldsymbol{b}) \cos \left(\tfrac{1}{\sqrt{D}} \boldsymbol{\Gamma} \cdot (\boldsymbol{ab}) \right) \right]^D. \tag{A.21}$$

Then performing a Taylor expansion of $\cos(\cdots)$, we get

$$H^{(m)}(\boldsymbol{a}) = \lim_{D\to\infty} \left[\sum_{\boldsymbol{b}} f(\boldsymbol{b}) H_D^{(m-1)}(\boldsymbol{b}) \left(1 - \tfrac{1}{2D} \left(\boldsymbol{\Gamma} \cdot (\boldsymbol{ab}) \right)^2 + O\left(\tfrac{1}{D^2} \right) \right) \right]^D. \tag{A.22}$$

Using the fact that for any m,

$$\sum_{\boldsymbol{a}} f(\boldsymbol{a}) H_D^{(m)}(\boldsymbol{a}) = 1 \tag{A.23}$$

which is proved in Ref. [4, Lemma 5], we get

$$H^{(m)}(\boldsymbol{a}) = \lim_{D\to\infty} \left[1 - \tfrac{1}{2D} \sum_{\boldsymbol{b}} f(\boldsymbol{b}) H_D^{(m-1)}(\boldsymbol{b}) \left(\boldsymbol{\Gamma} \cdot (\boldsymbol{ab}) \right)^2 + O\left(\tfrac{1}{D^2} \right) \right]^D. \tag{A.24}$$

Finally, taking the limit,

$$H^{(m)}(\boldsymbol{a}) = \exp\Big[-\frac{1}{2}\sum_{\boldsymbol{b}} f(\boldsymbol{b})H^{(m-1)}(\boldsymbol{b})\big(\boldsymbol{\Gamma}\cdot(\boldsymbol{ab})\big)^2\Big] \tag{A.25}$$

which yields an iteration on $H^{(m)}$.

Returning to Eq. (A.19), we apply the product rule of limits to $H_D^{(p)}(\boldsymbol{a}), H_D^{(p)}(\boldsymbol{b})$, and $\sqrt{D}\sin[\boldsymbol{\Gamma}\cdot(\boldsymbol{ab})/\sqrt{D}]$ and get

$$\lim_{D\to\infty}\nu_p(D,\boldsymbol{\gamma},\boldsymbol{\beta}) = \frac{i}{2}\sum_{\boldsymbol{a},\boldsymbol{b}} a_0 b_0 f(\boldsymbol{a})f(\boldsymbol{b})H^{(p)}(\boldsymbol{a})H^{(p)}(\boldsymbol{b})\boldsymbol{\Gamma}\cdot(\boldsymbol{ab}). \tag{A.26}$$

This iteration can be simplified by expanding the dot products in Eqs. (A.25) and (A.26) to get

$$H^{(m)}(\boldsymbol{a}) = \exp\Big[-\frac{1}{2}\sum_{j,k=-p}^{p}\Gamma_j\Gamma_k a_j a_k\Big(\sum_{\boldsymbol{b}} f(\boldsymbol{b})H^{(m-1)}(\boldsymbol{b})b_j b_k\Big)\Big], \tag{A.27}$$

$$\lim_{D\to\infty}\nu_p(D,\boldsymbol{\gamma},\boldsymbol{\beta}) = \frac{i}{2}\sum_{j=-p}^{p}\Gamma_j\Big(\sum_{\boldsymbol{a}} f(\boldsymbol{a})H^{(p)}(\boldsymbol{a})a_0 a_j\Big)\Big(\sum_{\boldsymbol{b}} f(\boldsymbol{b})H^{(p)}(\boldsymbol{b})b_0 b_j\Big) \tag{A.28}$$

and noticing that the quantity $\sum_{\boldsymbol{a}} f(\boldsymbol{a})H^{(m)}(\boldsymbol{a})a_j a_k$ appears repeatedly. For $0 \le m \le p$ and $-p \le j,k \le p$, define

$$G_{j,k}^{(m)} := \sum_{\boldsymbol{a}} f(\boldsymbol{a})H^{(m)}(\boldsymbol{a})a_j a_k. \tag{A.29}$$

For $m = 0$, this is

$$G_{j,k}^{(0)} = \sum_{\boldsymbol{a}} f(\boldsymbol{a})a_j a_k. \tag{A.30}$$

For $1 \le m \le p$, we plug Eq. (A.27) into Eq. (A.29) to get

$$G_{j,k}^{(m)} = \sum_{\boldsymbol{a}} f(\boldsymbol{a})a_j a_k \exp\Big[-\frac{1}{2}\sum_{j',k'=-p}^{p} G_{j',k'}^{(m-1)}\Gamma_{j'}\Gamma_{k'}a_{j'}a_{k'}\Big]. \tag{A.31}$$

Finally, Eq. (A.28) can be written as

$$\lim_{D\to\infty}\nu_p(D,\boldsymbol{\gamma},\boldsymbol{\beta}) = \frac{i}{2}\sum_{j=-p}^{p}\Gamma_j(G_{0,j}^{(p)})^2 \tag{A.32}$$

which establishes the iteration stated in Section 3.2.