



OPEN

Scaling quantum approximate optimization on near-term hardware

Phillip C. Lotshaw^{1✉}, Thien Nguyen^{2,3,6}, Anthony Santana^{2,7}, Alexander McCaskey^{2,3,8}, Rebekah Herrman⁴, James Ostrowski⁴, George Siopsis⁵ & Travis S. Humble^{1,3}

The quantum approximate optimization algorithm (QAOA) is an approach for near-term quantum computers to potentially demonstrate computational advantage in solving combinatorial optimization problems. However, the viability of the QAOA depends on how its performance and resource requirements scale with problem size and complexity for realistic hardware implementations. Here, we quantify scaling of the expected resource requirements by synthesizing optimized circuits for hardware architectures with varying levels of connectivity. Assuming noisy gate operations, we estimate the number of measurements needed to sample the output of the idealized QAOA circuit with high probability. We show the number of measurements, and hence total time to solution, grows exponentially in problem size and problem graph degree as well as depth of the QAOA ansatz, gate infidelities, and inverse hardware graph degree. These problems may be alleviated by increasing hardware connectivity or by recently proposed modifications to the QAOA that achieve higher performance with fewer circuit layers.

Combinatorial optimization problems are commonly viewed as a potential application for near-term quantum computers to obtain a computational advantage over conventional methods¹. A common approach to solving these problems uses the quantum approximate optimization algorithm (QAOA)², which begins with a “cost” Hamiltonian typically defined as

$$C = \sum_i h_i Z_i + \sum_{ij} J_{ij} Z_i Z_j \quad (1)$$

with real coefficients J_{ij} and h_i that encode a quadratic unconstrained binary optimization problem in the eigenspectrum of C ³. The QAOA prepares a quantum state $|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle$ on n qubits using p layers of unitary operators, where each layer alternates between Hamiltonian evolution under C and under a “mixing” Hamiltonian $B = \sum_{i=1}^n X_i$ composed of independent Pauli-X operators,

$$|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle = \left(\prod_{l=1}^p e^{-i\beta_l B} e^{-i\gamma_l C} \right) |+\rangle^{\otimes n}. \quad (2)$$

The state is then measured to yield the n -bit binary string z as a candidate solution to the problem. The angles $\boldsymbol{\beta} = (\beta_1, \dots, \beta_p)$ and $\boldsymbol{\gamma} = (\gamma_1, \dots, \gamma_p)$ are variational parameters chosen to minimize or maximize the expectation value $\langle C \rangle = \langle \boldsymbol{\gamma}, \boldsymbol{\beta} | C | \boldsymbol{\gamma}, \boldsymbol{\beta} \rangle$, depending on whether the optimal solution in C is the minimum or maximum value, respectively.

Farhi et al. have argued that QAOA recovers the ground state of C as $p \rightarrow \infty$ ², but the primary interest in QAOA is in reaching high performance with a modest number of layers p that could realistically be implemented on a quantum computer. A significant body of theoretical^{4–8}, computational^{9–13}, and experimental^{14,15} research has focused on understanding QAOA performance at $p \approx 1$, mostly on the MaxCut problem with a small number of qubits n , but also for other types of problems^{16–18}. These studies have shown some promising results,

¹Quantum Computational Sciences Group, Oak Ridge National Laboratory, Oak Ridge, TN 37830, USA. ²Beyond Moore Computing Group, Oak Ridge National Laboratory, Oak Ridge, TN 37830, USA. ³Quantum Science Center, Oak Ridge National Laboratory, Oak Ridge, TN 37830, USA. ⁴Department of Industrial and Systems Engineering, University of Tennessee, Knoxville, TN 37996-2315, USA. ⁵Department of Physics and Astronomy, University of Tennessee, Knoxville, TN 37996-1200, USA. ⁶Present address: Quantum Brilliance, Acton, Australia. ⁷Present address: Q-CTRL, Los Angeles, USA. ⁸Present address: NVIDIA, Santa Clara, USA. ✉email: lotshawpc@ornl.gov

for example, with QAOA outperforming the conventional lower bound of the GW algorithm for MaxCut on some small instances^{19,20}. There have also been a variety of proposed modifications to the algorithm to improve performance^{21–28} and solve optimization problems with constraints^{29–31}. The results from these and other studies have encouraged research into extending the QAOA to larger and more complex problems.

In contrast to the QAOA studies focused on a small number of variables n , conventional computational methods are capable of handling problem instances with hundreds of variables or more. To assess the usefulness of QAOA it will be necessary to scale to larger and more complex instances where it can be directly compared against these methods on practically relevant problems. A recent study suggests that hundreds of qubits are needed³² to compete in time-to-solution, while the theoretical and experimental performance in this context are important open questions. Theoretical considerations indicate that the number of layers p will need to scale at least as $\log(n)$ in some instances, as the locality of the ansatz limits the ability to build global correlations that are needed for globally optimal solutions^{33,34}. Classical algorithms have also been developed that outperform QAOA at low p ^{35,36}, further suggesting large p may be necessary to compete with conventional methods. To optimize parameters at large n and p , a variety of computational^{37,38} and theoretical^{39–44} approaches have been developed and in some cases the theoretical performance has been characterized. With parameter setting strategies at hand, what remains to be seen is how the QAOA will perform in experimental implementations. The prospect of experimentally implementing the QAOA at large n and p raises questions about how quantum computing resources will scale with problem size and complexity, and how noise will influence the behavior of the algorithm.

Here we report on the scaling of resources needed by QAOA on near-term intermediate-scale quantum (NISQ) devices. We show how features of the combinatorial problem and the target hardware influence the total number of gates and measurements required to reach a specified threshold of accuracy. First we consider problem features such as the average degree d_G of the graph defining the problem instance, where d_G is related to the number of non-zero terms in the quadratic unconstrained binary optimization problem. While much of the QAOA literature has focused on problems with small d_G , larger d_G arises naturally in constrained combinatorial optimization problems^{45,46}. In addition to d_G , the problem size n and the number of QAOA layers p also contribute to the gate counts and hence the resources required to implement the algorithm. It is furthermore important to consider the constraints that arise in current NISQ hardware due to limited connectivity on the hardware device qubit register, which can require costly SWAP gates to transport logical qubits. We show that the interplay between these logical requirements and hardware constraints generate steep scaling in the resources required for high-fidelity implementation of QAOA as n , p , and d_G increase.

Our approach synthesizes optimized circuit representations of QAOA for varying problem sets targeting constrained noisy hardware. We compile circuits in terms of a generic gate set of controlled-not, Hadamard, and rotation gates; these can be translated into native gates for specific hardware, though we do not include this here. We optimize both the number of gates and the overall performance through placement of the logical qubits and injected SWAP gates. Placement and routing are difficult optimization problems and it is not clear a priori how an ideal QAOA instance expressed as Eq. (2) will map to a given hardware^{47–50}. To understand the role of hardware connectivity, we synthesized optimized QAOA circuits on scaled versions of each of the connectivity architectures shown in Fig. 1. These planar architectures correspond to contemporary and hypothesized hardware designs. Each architecture has a distinct connectivity defined as the average hardware graph degree d_H , i.e., the average number of distinct two-qubit gate connections per hardware register element (ignoring perimeter elements to give a size-independent d_H). The architectures range from $d_H = 2.5$ for the heavy hexagonal lattice in Fig. 1a to $d_H = 6$ for the triangular lattice in Fig. 1d. We quantify the SWAP gate counts with respect to d_H , d_G , n , and p , and we fit scaling relations to these results.

Resource counts also give insight into the scalability of the QAOA in the presence of noise. We define a simple noise model for a quantum state traversing a circuit with gate counts estimated from our resource analysis and use this to quantify the reliability of QAOA as it scales to larger and more complex problems. Our analysis complements previous theoretical results describing how noise influences the QAOA cost expectation value, trainability, and eigenvectors of the density operator^{51–54}. We compute an upper bound for the number of measurements M that are needed to obtain a single result from the idealized state that would be produced by a noiseless version of the circuit, based on effective gate error rates but without assuming any specific structure or correlations in the noise process. This characterizes the reliability of the algorithm and the expected time-to-solution T , assuming $T \propto M$. The results assess the scalability of the QAOA on noisy near-term hardware and the expected influence of d_H , d_G , n , and p .

Results

Mapping to hardware. We express the QAOA unitary operators of Eq. (2) in terms of a hardware gate set of Hadamards H , Z -rotations $R(\theta)$, and controlled-not CNOT, as described in “Methods”. The gate-to-unitary operator correspondences given there provide the minimal numbers of each type of gate that must be implemented in the algorithm, for example, on fully connected hardware.

It is useful to classify problem instances C in terms of their circuit structure. We define problem graphs G with vertices for each qubit i and edges (i, j) for each non-zero $J_{i,j}$ constant in Eq. (1). Each edge (i, j) requires a set of two-qubit gates $\text{CNOT}_{i,j}R_j(2J_{i,j}\gamma_l)\text{CNOT}_{i,j}$ and the total set of edges defines all two-qubit gates that are needed on fully connected hardware. The specific values of the parameters $J_{i,j} \neq 0$, h_i , γ_l , and β_l enter as rotation angles in the circuit, hence all problem instances with the same problem graph have the same circuits up to choices of these angles. When an $h_i = 0$ then a single-qubit gate can be further removed from the circuit, but this does not affect the two-qubit gate structure. We consider all non-isomorphic connected problem graphs with $n = 7$ qubits to determine how the circuits scale with the average problem graph degree d_G ; to determine scaling with

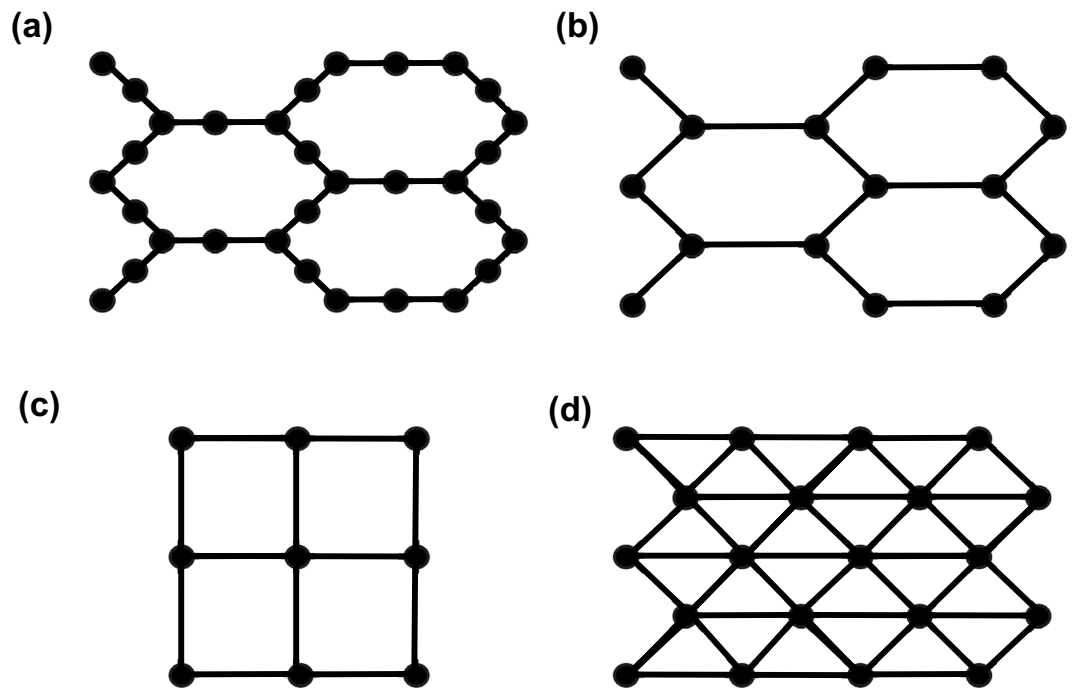


Figure 1. Hardware connectivity graphs for (a) heavy-hexagon, $d_H = 2.5$ (b) hexagon, $d_H = 3$, (c) square, $d_H = 4$, and (d) triangle, $d_H = 6$.

the number of qubits we assess 3-regular problem graphs with $d_G = 3$ at varying n . On fully connected hardware, the number of gates of each type are

$$N_H = 2np + n, \quad (3)$$

$$N_R = p \left(\eta + \frac{n(d_G + 2)}{2} \right), \quad (4)$$

$$N_{\text{CNOT}}^{\text{fc}} = pnd_G - N_0, \quad (5)$$

where η is the number of non-zero h_i in Eq. (1) and $N_0 \leq \lfloor n/2 \rfloor$ is an instance-dependent number of CNOT gates that can be removed from the first layer of the circuit as they do not affect the initial state⁵⁵, see Supplemental Information for details.

However, on hardware with limited connectivity, it is often the case that some of the two-qubit gates cannot be implemented by any initial placement of the logical qubits onto the hardware register. For example, a non-planar problem graph cannot be mapped onto any of the planar registers in Fig. 1. It is therefore necessary to use SWAP gates to shuttle logical qubits around the register during execution of the circuit, to realize connections that are not available to the initial qubit placement. There are many potential circuits that can be created and these can result in different total numbers of SWAP gates, with up to $\binom{n}{2}$ SWAP gates in n circuit layers in the worst case^{56,57}. An ideal circuit will minimize the number of gates or circuit depth to minimize the negative impacts of noise in the circuit.

We compute circuits that minimize CNOT gate counts for each register architecture in Fig. 1 using an optimization routine. We optimize single layers of the QAOA algorithm as additional layers have the same circuit structure apart from differences in the qubit locations due to SWAP gates. These differences can be accounted for by mirroring the circuit implementation of $\exp(-i\gamma C)$ in subsequent layers, so that qubits move back and forth between locations from layer to layer. For an n -qubit problem instance, we use register grids of sizes just larger than $\sqrt{n} \times \sqrt{n}$, as we found that further increasing the grid size tended to result in larger optimized circuits. Our optimization procedure uses two nested loops. The inner loop is called in the circuit mapping algorithm SABRE⁴⁷, which generates a set of random placements of the logical qubits onto the hardware register then optimizes each placement, ultimately returning the final optimized circuit with the smallest depth. For our circuits, we have found that SABRE sometimes yields sub-optimal placements, as it does not recognize the commutativity of the terms $\exp(-i\gamma J_{i,j} Z_i Z_j)$ in Eq. (2), but instead tries to implement these in the order it is given. We therefore define an outer loop that randomly shuffles these commuting terms, to optimize over varying term orderings. This outer loop decreases the number of gates in our optimized circuits compared to a more basic implementation with SABRE only. For each problem graph, we take our final result from these nested loops as

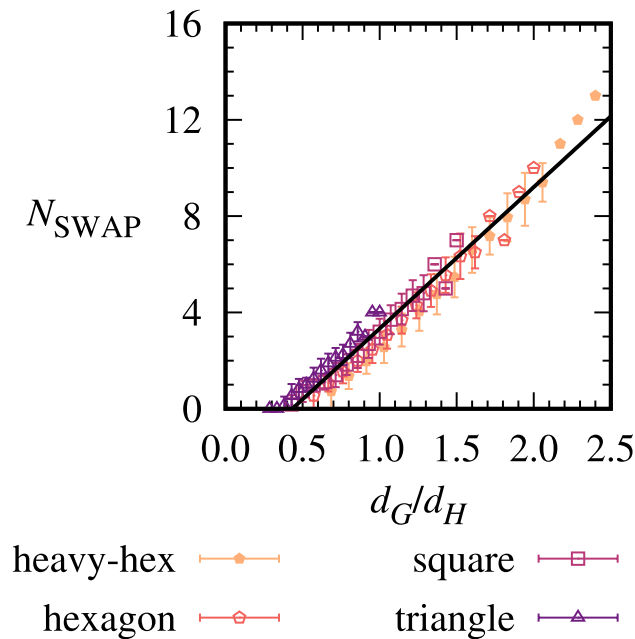


Figure 2. SWAP gate scaling with average problem degree d_G and hardware degree d_H for 7-vertex graphs. The solid line shows the non-linear least squares fit to $N_{\text{SWAP}}(d_G, d_H) = ad_G/d_H + b$, with $a = 5.9 \pm 0.1$ and $b = -2.5 \pm 0.2$, with \pm indicating the asymptotic standard error of the fit parameters.

the circuit with the fewest CNOT gates. The total number of CNOT gates on hardware with limited connectivity with N_{SWAP} SWAP gates is

$$N_{\text{CNOT}} = N_{\text{CNOT}}^{\text{fc}} + p\sigma N_{\text{SWAP}}, \quad (6)$$

where σ quantifies the average increase in CNOT gates per SWAP gate, beyond the $N_{\text{CNOT}}^{\text{fc}}$ gates that are needed on fully connected hardware. Each SWAP gate is defined as a product of three CNOT gates, so $\sigma = 3$ in the worst case. In better cases, a SWAP $_{ij}$ gate is placed adjacent to a CNOT $_{ij}$ gate in the circuit and CNOT $_{ij}$ CNOT $_{ij} = \mathbb{1}$ is used to remove a pair of gates. This gives $1 \leq \sigma \leq 3$ in our accounting. Further details of the implementation, convergence behavior, and performance can be found in Supplemental Information.

Scaling with problem size and degree. We next mapped circuits for each of the 853 non-isomorphic problem graphs at $n = 7^8$. The results in Fig. 2 show how the number of SWAP gates N_{SWAP} scales with the average problem graph degree d_G at this n across our hardware with varying d_H . As d_G increases so does the number of edges in the graph, and hence the number of two-qubit gates in each layer of the QAOA algorithm. Greater numbers of SWAP gates are needed on average to accommodate these two-qubit gates; in other words, problem graphs that are highly connected (with large d_G) are likely to have edges that cannot be realized by the initial placement of qubits onto the limited-connectivity registers, thereby requiring SWAP gates. A complementary analysis is given in Supplemental Information in terms of the problem graph diameter, which is the maximum of all minimum distances between qubits in the graph, and is also related to the connectivity of the graph. As the hardware degree d_H increases a greater number of two-qubit gates are available natively on the hardware, so fewer SWAP gates are needed. The mean numbers of SWAP gates at each d_G and d_H are fit by an empirical linear relation $N_{\text{SWAP}}(d_G, d_H) \sim d_G/d_H$ with fit parameters in the figure caption and a root-mean-square-error (RMSE) of 0.58 SWAP gates. The small error indicates the empirical relation is successful in providing a unified account of the N_{SWAP} scaling across problem graphs and hardware architectures at this n .

Next we consider how the number of SWAP gates scales with the size of the problem n . We considered sets of 3-regular graphs with 108 graph instances each at $n = 20, 40$, and 60 qubits. The 3-regular problem graphs have three non-zero $J_{i,j}$ terms for each qubit i in Eq. (1) and this standardizes $d_G = 3$ as we scale to larger sizes. Three-regular graphs have also been studied with considerable interest in the QAOA MaxCut literature^{2,4,9,10,32,39,41} and in a previous experimental demonstration of QAOA¹⁴. They are appealing targets for near-term hardware since most graphs at the same n have higher average degree d_G , hence we expect them to require more noisy two-qubit gates, due to both the increase in the minimal number of CNOT gates in Eq. (5) and also the expected increase in SWAP gates following the previous analysis of Fig. 2.

We computed optimized circuit mappings for these 3-regular instances to obtain the key result pictured in Fig. 3, which relates the number of SWAP gates to the average hardware degree d_H as the problem size n increases. We fit the data with an empirical curve that is based on counting the number of two-qubit terms that cannot be implemented by the initial qubit placement and assuming the number of SWAP gates needed to bring the qubits

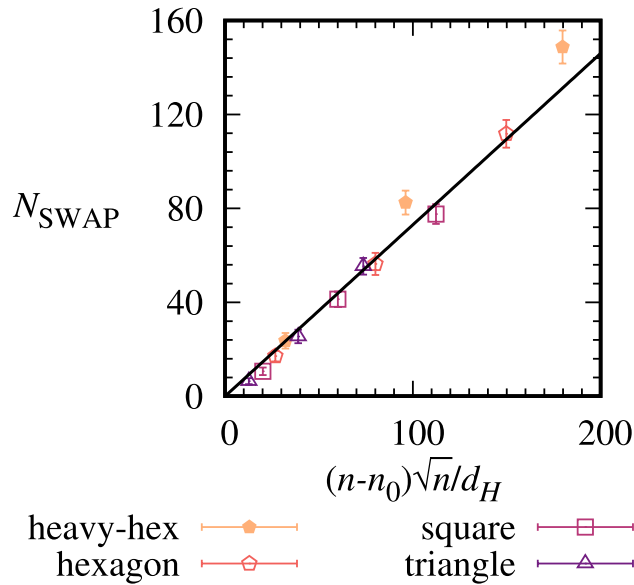


Figure 3. Average SWAP gate scaling with number of qubits n and hardware degree d_H for 3-regular graphs.

together for these edge terms increases on average in proportion to the length and width of the hardware grid, see [Methods](#) for details. This leads to the empirical relation shown by the solid line in the figure

$$N_{\text{SWAP}}(n, d_H) = \mu(n - n_0)\sqrt{n}/d_H, \tag{7}$$

where $\mu = 0.73 \pm 0.02$ is a fit parameter computed through non-linear least squares and ± 0.02 is the asymptotic standard error. Here n_0 sets the zero of N_{SWAP} and represents the maximum problem graph size at which all graphs can be mapped to hardware, for example, for fully connected hardware $n_0 = n$ and $N_{\text{SWAP}}(n, d_H) = 0$. For the triangle lattice in Fig. 1(d), all 3-vertex problem graphs can be mapped directly onto the lattice but the 4-vertex complete graph cannot be, so $n_0 = 3$. For the other hardware lattices, $n_0 = 2$.

We assess the performance of the empirical formula using the RMSE between the average N_{SWAP} and the empirical $N_{\text{SWAP}}(n, d_H)$. Across all results in Fig. 3, the RMSE=7.2 SWAP gates. The RMSE is strongly influenced by the outliers for the heavy-hexagon array at $n = 40$ and $n = 60$, where the empirical formula is up to 16% smaller than the results. These deviations may be related to the bimodal degree structure of the heavy-hexagon array in Fig. 1a, which has a mixture of register elements of degrees two and three, unlike the other constant-degree hardwares. Excluding the results for the heavy-hexagon at $n = 40$ and $n = 60$ decreases the RMSE to 2.7 SWAP gates. We conclude the empirical formula is giving a good fit to the majority of data in the figure, apart from the heavy-hexagon at large n , where the formula gives a looser bound to the observed N_{SWAP} .

Noisy architecture model and measurement count scaling. We use a simple noise model for our circuits to assess how noise influences the scalability of the QAOA, in terms of the number of measurements M that are needed from a noisy circuit to obtain a single result from the intended noiseless quantum state distribution. This quantifies the reliability of a noisy QAOA circuit in producing the intended output and also characterizes the scaling in the time-to-solution $T \propto M$.

An instance of a QAOA circuit is expressed in terms of a series of gates with ideal unitary evolution operators U_0, U_1, \dots , with $U_\alpha \in \{\text{H}, \text{R}, \text{CNOT}\}$ the unitary for the α th gate, acting on an initial state $\rho_0 = (|0\rangle\langle 0|)^{\otimes n}$. The noisy state produced by the α th gate is expressed using a quantum channel as⁵⁹

$$\rho_{\alpha+1} = (1 - \epsilon_\alpha)U_\alpha\rho_\alpha U_\alpha^\dagger + \sum_{k=1}^K \epsilon_\alpha^{(k)} E_\alpha^{(k)} U_\alpha\rho_\alpha U_\alpha^\dagger E_\alpha^{(k)\dagger}, \tag{8}$$

where the Kraus operators $(\epsilon_\alpha^{(k)})^{1/2} E_\alpha^{(k)}$ give noisy deviations from the intended evolution with probabilities $\epsilon_\alpha^{(k)}$. The final state of the circuit is⁵⁴

$$\rho = F_0\rho_{\text{ideal}} + (1 - F_0)\rho_{\text{noise}} \tag{9}$$

where $\rho_{\text{ideal}} = |\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle\langle \boldsymbol{\gamma}, \boldsymbol{\beta}|$ is the density operator for the intended pure state $|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle$, ρ_{noise} is a density operator composed of all terms with at least one Kraus operator, and $F_0 = \prod_\alpha (1 - \epsilon_\alpha)$ is a lower bound to the state preparation fidelity $F = \langle \boldsymbol{\gamma}, \boldsymbol{\beta} | \rho | \boldsymbol{\gamma}, \boldsymbol{\beta} \rangle \geq F_0$, with equality when $\text{Tr} \rho_{\text{ideal}} \rho_{\text{noise}} = 0$. If we assume constant error rates $\epsilon_{\text{CNOT}}, \epsilon_{\text{H}}$, and ϵ_{R} for each CNOT, H, and R gate respectively, then

$$F_0 = (1 - \epsilon_{\text{CNOT}})^{N_{\text{CNOT}}} (1 - \epsilon_{\text{H}})^{N_{\text{H}}} (1 - \epsilon_{\text{R}})^{N_{\text{R}}}, \tag{10}$$

where the N are the corresponding gate counts.

A noisy implementation of QAOA will be effective when it can produce measurement results from the intended state distribution ρ_{ideal} . In the absence of readout errors, a measurement projects the total state ρ onto a computational basis state $|z\rangle$ that is the result of the measurement, with probability $P(z) = \langle z|\rho|z\rangle = F_0 P_{\text{ideal}}(z) + (1 - F_0) P_{\text{noise}}(z)$. This has a lower bound $P(z) \geq F_0 P_{\text{ideal}}(z)$ independent of the specific noise process, apart from the values of the error rates ϵ_α that determine F_0 . Summed over all $|z\rangle$ in the support S of ρ_{ideal} , the total probability $P = \sum_{|z\rangle \in S} P(z)$ to obtain any result from the ideal state distribution is

$$P \geq F_0. \quad (11)$$

We use this probability inequality to bound the number of measurements $M = \log(1 - \mathcal{P}) / \log(1 - P)$ that are needed to obtain a single sample from the distribution of the intended state with probability \mathcal{P} ^{18,20},

$$M \leq \frac{\log(1 - \mathcal{P})}{\log(1 - F_0)}. \quad (12)$$

It is useful to consider a few examples. In a theoretical best case of QAOA, the intended state is a single computational basis state $|\gamma, \beta\rangle = |z_{\text{opt}}\rangle$ that gives the optimal cost value $C(z_{\text{opt}}) = C_{\text{opt}} \in \mathbb{R}$. If we assume that noise does not contribute significantly to the probability for $|z_{\text{opt}}\rangle$, then $P \approx F_0$ and M is close to the upper bound. In more generic cases of interest, the intended state has non-zero probability for a variety of approximately optimal states and the goal is to measure any one of these states. In this case M may be smaller than the upper bound, and potentially much smaller if the probability to measure approximately optimal states is significant for the ρ_{noise} component. Smaller upper bounds for M might then be obtained using information about the noise process and its expected influence in ρ_{noise} . However, without detailed information about a specific state and noise process we do not have a way to decrease M below the upper bound, which serves as a generic guide for any possible intended QAOA state and noisy evolution of the type in Eqs. (8)–(9).

We assessed the scalability of the number of measurement samples by computing the upper bound for M for 3-regular graphs at varying sizes n and at $p = 20$ QAOA layers, with a probability $\mathcal{P} = 0.99$ to sample from the intended state distribution. We evaluate the analytic bound for M in (12), following from earlier analytic expressions in (8)–(10), along with gate counts from our optimized circuits; these calculations are not based on simulations of noisy hardware. We consider 3-regular problem graph instances with gate counts N_H , N_R , and N_{CNOT} in Eqs. (3), (4), and (6) respectively, assuming all $h_i \neq 0$ in Eq. (1) so that $\eta = n$. We use N_{SWAP} computed from the empirical formula of Eq. (7) for each hardware architecture in Fig. 1, $\sigma = 3$ as the number of additional CNOT gates per SWAP gate in Eq. (6), in accord with our results at large n from Supplemental Information, and we approximate $N_0 = 0$ since $N_0 \ll N_{\text{CNOT}}$ when $p = 20$. The F_0 in M is then computed from Eq. (10) with assumed error rates of $\epsilon_{\text{CNOT}} = 5 \times 10^{-5}$ and $\epsilon_R = \epsilon_H = \epsilon_{\text{CNOT}}/10$. For comparison, recent advances in transmon qubits have achieved two-qubit gate error rates of 6.4×10^{-3} and single-qubit error rates of 3.8×10^{-460} .

Figure 4 shows how this M scales with problem size n . The number of measurements increases exponentially with n at a rate that depends on the hardware degree d_H . The variations in hardware themselves give an exponential divergence in M as the reciprocal hardware degree $1/d_H$ increases and the hardware becomes less connected (Fig. 4 inset), due to the empirical dependence of $N_{\text{SWAP}} \sim 1/d_H$ from Eq. (7). The hardware dependence is significant at the large n that are required for practical problems. For example, at $n = 500$ (vertical dotted line), the number of measurement samples is approximately 20 for fully connected hardware but increases by four orders of magnitude going to the least connected hardware (heavy-hexagon, Fig. 1a). Here $n = 500$ exemplifies a nontrivial problem size but is otherwise arbitrary—similar scaling behavior is observed for other large n . Curves similar to Fig. 4 can also be computed for fixed n as the error rates ϵ_α , number of QAOA layers p , or as the problem graph degree d_G increase, see Supplemental Information for details.

Discussion

Prospects for obtaining a quantum computational advantage with the QAOA are expected to require hundreds of qubits or more to compete against conventional methods on practically relevant problems^{18,32}. As the QAOA scales to larger and more complex problems, the number of gates to implement the algorithm on fully connected hardware increases with the problem graph degree d_G and number of qubits n . For sparsely connected hardware additional SWAP gates are needed. We computed optimized circuits to determine how the number of SWAP gates N_{SWAP} scales with n and d_G on a variety of real and hypothetical hardware architectures with varying levels of connectivity in terms of the hardware degree d_H . The reciprocal hardware degree $1/d_H$, average problem graph degree d_G , and number of qubits n were each found to be important scaling factors in the empirical behavior of N_{SWAP} . Using a simple noise model with gate counts extrapolated from our circuits we computed the number of measurement samples M from a noisy circuit that are needed to obtain a single measurement from the distribution of an idealized noiseless version of the state with probability \mathcal{P} . This is a measure of the reliability of a noisy circuit in producing the intended outcome. We argued that M increases exponentially with n , d_G , $1/d_H$, the number of QAOA layers p , and the gate error rates ϵ_α . Assuming that M is proportional to the time to solution, this corresponds to an exponential time complexity in each of these factors.

We considered $n = 500$ as an example of a nontrivial problem size to compare the number of measurements across different hardware. Our results show that the number of measurement samples is $2 \times 10^3 \leq M \leq 5 \times 10^5$ at this n and $p = 20$ for the considered error rates and hardware. These numbers of measurements should not be difficult to obtain from a quantum computer. However, our parameter choices and problem sets were optimistic in some respects. The assumed error rates were about two orders of magnitude below current state of the art

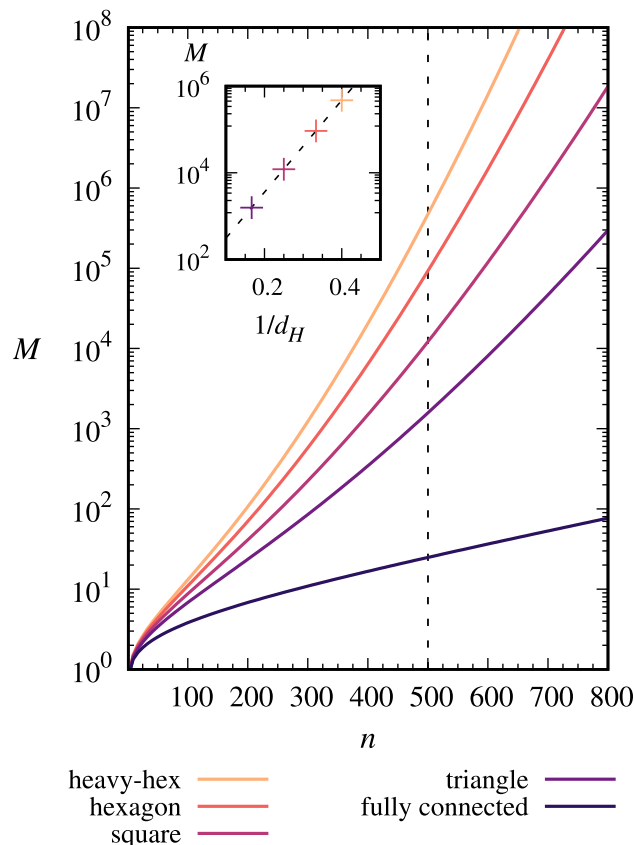


Figure 4. The number of measurement samples M to measure a result from the intended state for 3-regular graphs, see text for details. Inset: M diverges exponentially in $1/d_H$.

devices⁶⁰ and larger error rates exponentially increase the number of measurements. For example, doubling the error rates so that $\epsilon_{\text{CNOT}} = 10^{-4}$ gives $5 \times 10^5 \leq M \leq 5 \times 10^{10}$ for our hardware. We also assumed 3-regular problem graphs, which have been studied with great interest in the QAOA literature. However, many practically relevant problems use denser problem graphs, for example in constrained optimization problems^{18,45,46}. For denser graphs the average degree can scale as n and changes in degree can significantly affect M . For example, using our approach and parameter choices for a 500 qubit problem graph with average degree $d_G = 25$ we obtain $M = 3 \times 10^6$ on fully connected hardware. For the sparsely connected hardware we consider we do not have a precise scaling relation for N_{SWAP} on $d_G = 25$ graphs, but if we optimistically use the same relationship $N_{\text{SWAP}}(n, d_H)$ we found for 3-regular graphs we obtain $2 \times 10^8 \leq M \leq 5 \times 10^{10}$ at $d_G = 25$. This is ignoring any dependence of N_{SWAP} on d_G , which would be significant if our small n observation $N_{\text{SWAP}} \sim d_G$ holds also at large n . A final note is that if more than one measurement is needed from the state with high probability, then this will introduce an additional scaling beyond the M presented here. The numbers of measurements quickly become greater than what can realistically be expected from near-term quantum computers.

We expect the measurement scaling will significantly inhibit the ability to implement the QAOA at scales relevant for quantum advantage. When the QAOA parameters are optimized using measurements from a quantum computer, this optimization will also be greatly inhibited. Parameter optimization has been addressed in some instances using theoretical approaches^{9,19,20,37–44}, though for generic instances it is unclear if such approaches can be applied. However, even with a good set of parameters the circuit must still be run to obtain the final bit-string solution to the problem, and in our model this requires a number of measurements that quickly becomes prohibitive at scales relevant for quantum advantage. Straightforward attempts to scale the QAOA will face a significant barrier if these scaling problems are not addressed.

Our expectations for performance are based on a general upper bound that is saturated when the noisy and ideal components of the total circuit density operator give distinct measurement results in the computational basis. A vanishing overlap in measurement results is expected when the ideal QAOA circuit prepares a computational basis state, while intermediate superposition states may have non-negligible overlap with the noisy subspace. Further analysis will require details from hardware-specific noise models to determine more precise estimates for how such errors influence M . In addition, there are methods to overcome the measurement count limitations. One approach is to significantly increase hardware connectivity, for example, through non-planar hardware grids. These would overcome the basic inability to directly implement non-planar problem graphs, and detailed accounting of SWAP behavior for such architectures is an important topic for future research. Another possibility is to modify the gate set, for example, using ion-trap quantum computers with globally-entangling

Mølmer-Sørensen gates⁶¹ or Rydberg atoms that naturally enforce constraints in some instances of QAOA⁶². Another approach is to modify the QAOA ansatz. This includes introducing additional parameters within layers of QAOA²¹, modifying the structure of the ansatz^{22–25}, modifying the cost function²⁷, objective function²⁸, and circuit structure²⁶. Such technological and algorithmic advances are likely necessary to reduce the numbers of layers or gates, and hence the accumulated noise, as the QAOA scales to larger sizes.

Methods

We generated circuits using the XACC quantum programming framework^{63,64} to map the unitary quantum operators of Eq. (2) to a gate set of Hadamards H , Z -rotations $R(\theta) = \exp(-i(\theta/2)Z)$, and controlled-NOT CNOT gates. To map these circuits to hardware with limited connectivity, we used the Enfield software library⁴⁸ and SABRE algorithm⁴⁷ implemented within XACC. Details of the implementation, convergence behavior, and comparison with a lower bound for N_{SWAP} at small n are described in the Supplemental Information.

In terms of our gate set, the unitary operators in Eq. (2) are

$$\exp(-i\gamma I_{i,j} Z_i Z_j) = \text{CNOT}_{ij} R_j(2\gamma I_{i,j}) \text{CNOT}_{ij}, \quad (13)$$

$$\exp(-i\gamma h_i Z_i) = R_i(2\gamma h_i) \quad (14)$$

$$\exp(-i\beta I X_i) = H_i R_i(2\beta I) H_i. \quad (15)$$

Empirical formula for 3-regular graphs. We construct the empirical curve $N_{\text{SWAP}}(n, d_H)$ in Eq. (7) by considering how many two-qubit gates cannot be implemented by the initial mapping of qubits onto the register along with the average expected behavior for how many SWAP gates are needed to bring qubits together for each of these gates. We begin by separating the edge terms in a mapped problem graph instance into edges $s = \langle s_1, s_2 \rangle$ that are “satisfied” by the initial placement of qubits on the register, in the sense that the two-qubit gates between s_1 and s_2 can be implemented in the initial placement, and edges $u = \langle u_1, u_2 \rangle$ that are “unsatisfied,” in the sense that SWAP gates are needed to bring the qubits u_1 and u_2 together to implement their two-qubit gates. Our approach is to express the total number of SWAP gates as $N_{\text{SWAP}} = \sum_u N_{\text{SWAP}}^{(u)}$, where $N_{\text{SWAP}}^{(u)}$ is the number of SWAP gates that are used in the circuit to bring qubits u_1 and u_2 together to implement the two-qubit gates for u .

Some care is needed to define the $N_{\text{SWAP}}^{(u)}$ to give a consistent total N_{SWAP} . Each SWAP gate moves locations of two qubits and hence can contribute to two terms $N_{\text{SWAP}}^{(u)}$ and $N_{\text{SWAP}}^{(u')}$; one approach is to allow for fractional values in the $N_{\text{SWAP}}^{(u)}$, for example, values of 1/2 in $N_{\text{SWAP}}^{(u)}$ and $N_{\text{SWAP}}^{(u')}$ when a SWAP gate moves two qubits that help to satisfy u and u' . Another consideration is that a series of SWAP gates may be implemented before the gates for a given u , while along the way the SWAP gates that are relevant for u may also allow for implementations of two-qubit gates for a variety of other u', u'', \dots . We could then assign fractional values to each of the $N_{\text{SWAP}}^{(u)}, N_{\text{SWAP}}^{(u')}, N_{\text{SWAP}}^{(u'')}, \dots$ based on which qubits are moved by the series of SWAP gates and which unsatisfied edges they contribute to, such that $N_{\text{SWAP}} = \sum_u N_{\text{SWAP}}^{(u)}$. A final consideration is that sometimes the circuits will SWAP qubits that are in initially satisfied edges s before the two-qubit gates for those edges are implemented. Although additional SWAP gates are sometimes used in these cases for the satisfied edges s , these SWAP gates are only needed because there were initially unsatisfied edges u which began a series of SWAP gates earlier in the circuit, so it is reasonable to systematically assign the SWAP gates for these s to the $N_{\text{SWAP}}^{(u)}$. Although the calculation of the $N_{\text{SWAP}}^{(u)}$ is somewhat complicated by these considerations, by design the total must always sum to N_{SWAP} . This can be expressed as an average $N_{\text{SWAP}} = N_u \overline{N_{\text{SWAP}}^{(u)}}$, where N_u is the total number of unsatisfied edges and $\overline{N_{\text{SWAP}}^{(u)}}$ is the average number of SWAP gates per unsatisfied edge. The N_u is determined solely by the initial placement of qubits onto the register, while the average $\overline{N_{\text{SWAP}}^{(u)}} = N_{\text{SWAP}}/N_u$. We argue for the behavior of these terms in determining N_{SWAP} and the empirical fit curve of Eq. (7).

For each hardware architecture and circuit, we computed the number of two-qubit edge terms N_u that cannot be implemented directly on the hardware with the initial qubit placement. The N_u for each hardware are found to scale as $N_u \sim (n - n_0)$, where n_0 is a threshold size at which all graphs can be mapped directly to the hardware. The quantity n_0 sets the zero of N_u and hence N_{SWAP} , for example, on fully connected hardware $n_0 = n$ so $N_u = 0$ and no SWAP gates are needed. The rationale for the n dependence is that, on average, the number of unsatisfied edges increases linearly with the total number of edges, $E = 3n/2$ for the 3-regular graphs. The linear relations $N_u \sim (n - n_0)$ for each individual hardware are shown in Supplemental Information. They can be related to one another with a factor $d_H^{-1/2}$ that decreases the number of unsatisfied edges when more two-qubit connections d_H are available on the register. This gives a single unified relationship $N_u(n, d_H) \sim (n - n_0)/\sqrt{d_H}$ for all our hardware architectures as shown in Fig. 5. This motivates and accounts for a factor $(n - n_0)/\sqrt{d_H}$ in the empirical formula in Eq. (7).

The remaining factor $\sqrt{n/d_H}$ in the empirical $N_{\text{SWAP}}(n, d_H)$ of Eq. (7) relates to the average numbers of SWAP gates per unsatisfied edge $\overline{N_{\text{SWAP}}^{(u)}}$. We can rationalize the \sqrt{n} dependence by considering how many SWAP gates are needed to bring qubits together to satisfy an edge u , based on the typical distance between qubits on the approximately $\sqrt{n} \times \sqrt{n}$ hardware grids with $\sqrt{n} \in \mathbb{N}$. We begin by considering uniform random placements of logical qubits along a single dimension of length \sqrt{n} . The probability for the first qubit to be at location i is $P_i = 1/\sqrt{n}$, the probability for the second qubit to be at any other location j is $P_j = 1/(\sqrt{n} - 1)$, and the average

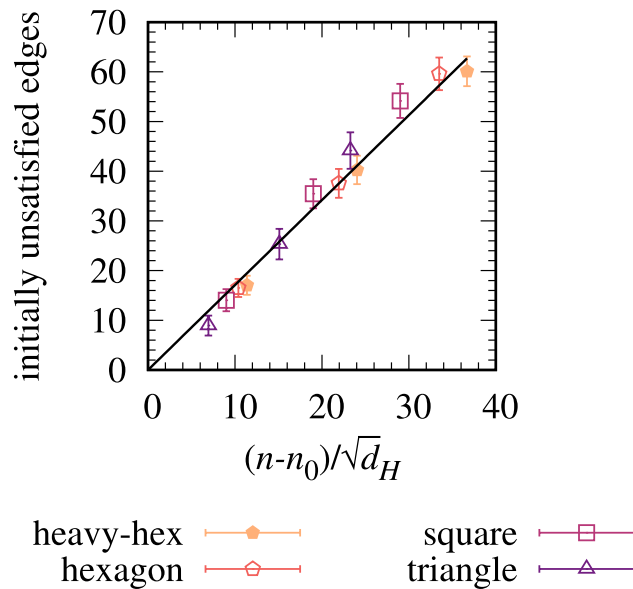


Figure 5. The number of initially unsatisfied edges N_u in the initial qubit placement at each n and d_H for 3-regular graphs.

distance between the qubits is $\sum_{i=1}^{\sqrt{n}} \sum_{j=1}^{\sqrt{n}} P_i P_j |i-j| = (n-1)/[3(\sqrt{n}-1)]$. This scales approximately as \sqrt{n} . If qubits are placed uniformly at random in two-dimensions and they move along each dimension separately, for example in the square hardware lattice of Fig. 1c, then the total distance is twice the distance in a single dimension and this again scales as \sqrt{n} . In reality the qubit placements are optimized instead of uniformly random, but still the length scales as \sqrt{n} in each dimension and this gives some justification for the appearance of \sqrt{n} in $\overline{N}_{\text{SWAP}}^{(u)}$. Finally, we need to account for a factor $1/\sqrt{d_H}$ to obtain the desired relation $\overline{N}_{\text{SWAP}}^{(u)} \sim \sqrt{n/d_H}$. We rationalize this factor by considering that fewer SWAP gates are needed to move a qubit from one location to another when there are more connections d_H on the register, for example, in the triangle lattice some diagonal movements are allowed on the planar grid and we expect this to decrease the number SWAP gates that are needed. We incorporate this through a factor $\sim 1/\sqrt{d_H}$ such that $\overline{N}_{\text{SWAP}}^{(u)} \sim \sqrt{n/d_H}$. Combined with the previous analysis of N_u , we have $N_{\text{SWAP}}(n, d_H) = N_u \overline{N}_{\text{SWAP}}^{(u)} \sim (n-n_0)\sqrt{n}/d_H$, giving the empirical formula of Eq. (7).

Data availability

Data from this study is available at <https://code.ornl.gov/5ci/dataset-scaling-qaoa-on-near-term-hardware/>.

Received: 25 February 2022; Accepted: 13 June 2022

Published online: 20 July 2022

References

1. Preskill, J. Quantum computing in the NISQ era and beyond. *Quantum* **2**, 79 (2018).
2. Farhi, E., Goldstone, J. & Gutmann, S. A quantum approximate optimization algorithm. <http://arxiv.org/abs/1411.4028> (2014).
3. Lucas, A. Ising formulations of many NP problems. *Front. Phys.* **2**, 125 (2014).
4. Wurtz, J. & Love, P. MaxCut quantum approximate optimization algorithm performance guarantees for $p > 1$. *Phys. Rev. A* **103**(4), 042612 (2021).
5. Wang, Z., Hadfield, S., Jiang, Z. & Rieffel, E. G. Quantum approximate optimization algorithm for maxcut: A fermionic view. *Phys. Rev. A* **97**(2), 022304 (2018).
6. Shaydulin, R., Hadfield, S., Hogg, T. & Saffo, I. Classical symmetries and QAOA. <http://arxiv.org/abs/2012.04713> (2020).
7. Hadfield, S. Quantum algorithms for scientific computing and approximate optimization. <http://arxiv.org/abs/1805.03265> (2018).
8. Hadfield, S., Hogg, T. & Rieffel, E. G. Analytical framework for quantum alternating operator ansatz. <http://arxiv.org/abs/2105.06996> (2021).
9. Galda, A., Liu, X., Lykov, D., Alexeev, Y. & Saffo, I. Transferability of optimal QAOA parameters between random graphs. <http://arxiv.org/abs/2106.07531> (2021).
10. Zhou, L., Wang, S.-T., Choi, S., Pichler, H. & Lukin, M. D. Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices. *Phys. Rev. X* **10**, 021067 (2020).
11. Akshay, V., Philathong, H., Morales, M. E. S. & Biamonte, J. D. Reachability deficits in quantum approximate optimization. *Phys. Rev. Lett.* **124**, 090504 (2020).
12. Shaydulin, R., Saffo, I. & Larson, J. Multistart methods for quantum approximate optimization. in *2019 IEEE High Performance Extreme Computing Conference (HPEC)*, 1–8 (IEEE, 2019).
13. Shaydulin, R. & Alexeev, Y. Evaluating quantum approximate optimization algorithm: A case study. in *2019 Tenth International Green and Sustainable Computing Conference (IGSC)*, 1–6 (2019).
14. Harrigan, M. P. et al. Quantum approximate optimization of non-planar graph problems on a planar superconducting processor. *Nat. Phys.* **17**, 332–336 (2021).

15. Pagano, G. *et al.* Quantum approximate optimization of the long-range ising model with a trapped-ion quantum simulator. *Proc. Natl. Acad. Sci.* **117**(41), 25396–25401 (2020).
16. Vikstål, P. *et al.* Applying the quantum approximate optimization algorithm to the tail-assignment problem. *Phys. Rev. Appl.* **14**, 034009 (2020).
17. Szegedy, M. What do QAOA energies reveal about graphs? <http://arxiv.org/abs/1912.12272v2> (2020).
18. Harwood, S. *et al.* Formulating and solving routing problems on quantum computers. *IEEE Trans. Quant. Eng.* **2**, 1–10 (2021).
19. Crooks, G. E. Performance of the quantum approximate optimization algorithm on the maximum cut problem. <http://arxiv.org/abs/1811.08419> (2018).
20. Lotshaw, P. C., Humble, T. S., Herrman, R., Ostrowski, J. & Siopsis, G. Empirical performance bounds for quantum approximate optimization. *Quant. Inf. Process.* **20**, 403 (2021).
21. Herrman, R., Lotshaw, P. C., Ostrowski, J., Humble, T. S. & Siopsis, G. Multi-angle quantum approximate optimization algorithm. <http://arxiv.org/abs/2109.11455> (2021).
22. Tate, R., Farhadi, M., Herold, C., Mohler, G. & Gupta, S. Bridging classical and quantum with SDP initialized warm-starts for QAOA. <http://arxiv.org/abs/2010.14021> (2020).
23. Zhu, L., Tang, H. L., Barron, G. S., Mayhall, N. J., Barnes, E. & Economou, S. E. An adaptive quantum approximate optimization algorithm for solving combinatorial problems on a quantum computer. <http://arxiv.org/abs/2005.10258> (2020).
24. Egger, D. J., Mareček, J. & Woerner, S. Warm-starting quantum optimization. *Quantum* **5**, 125 (2021).
25. Wurtz, J. & Love, P. Classically optimal variational quantum algorithms. <http://arxiv.org/abs/2103.17065> (2021).
26. Farhi, E., Goldstone, J., Gutmann, S. & Neven, H. Quantum algorithms for fixed qubit architectures. <http://arxiv.org/abs/1703.06199> (2017).
27. Patti, T. L., Kossaifi, J., Anandkumar, A. & Yelin, S. F. Nonlinear quantum optimization algorithms via efficient ising model encodings. <http://arxiv.org/abs/2106.13304> (2021).
28. Li, L., Fan, M., Coram, M., Riley, P. & Leichenauer, S. Quantum optimization with a novel Gibbs objective function and ansatz architecture search. *Phys. Rev. Res.* **2**, 1–10 (2020).
29. Hadfield, S. *et al.* From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms* **12**(2), 34 (2019).
30. Bartschi, A. & Eidenbenz, S. Grover mixers for QAOA: Shifting complexity from mixer design to state preparation. <http://arxiv.org/abs/2006.00354v2> (2020).
31. Cook, J., Eidenbenz, S. & Bartschi, A. The quantum alternating operator ansatz on maximum k -vertex cover. <http://arxiv.org/abs/1910.13483v2> (2020).
32. Guerreschi, G. G. & Matsuura, A. Y. QAOA for Max-Cut requires hundreds of qubits for quantum speed-up. *Sci. Rep.* **9**, 1–7 (2019).
33. Farhi, E., Gamarnik, D. & Gutmann, S. The quantum approximate optimization algorithm needs to see the whole graph: A typical case. <http://arxiv.org/abs/2004.09002> (2020).
34. Farhi, E., Gamarnik, D. & Gutmann, S. The quantum approximate optimization algorithm needs to see the whole graph: Worst case examples. <http://arxiv.org/abs/2005.08747> (2020).
35. Hastings, M. B. Classical and quantum bounded depth approximation algorithms. <http://arxiv.org/abs/1905.07047> (2019).
36. Marwaha, K. Local classical MAX-CUT algorithm outperforms $p = 2$ QAOA on high-girth regular graphs. *Quantum* **5**, 437 (2021).
37. Lykov, D., Schutski, R., Galda, A., Vinokur, V. & Alexeev, Y. Tensor network quantum simulator with step-dependent parallelization. <http://arxiv.org/abs/2012.02430> (2020).
38. Medvidović, M. & Carleo, G. Classical variational simulation of the quantum approximate optimization algorithm. *NPJ Quant. Inf.* **7**, 1–7 (2021).
39. Brandão, F. G. S. L., Broughton, M., Farhi, E., Gutmann, S. & Neven, H. For fixed control parameters the quantum approximate optimization algorithm's objective function value concentrates for typical instances. <http://arxiv.org/abs/1812.04170> (2018).
40. Wurtz, J. & Love, P. Counterdiabaticity and the quantum approximate optimization algorithm. <http://arxiv.org/abs/2106.15645> (2021).
41. Wurtz, J. & Lykov, D. The fixed angle conjecture for QAOA on regular MaxCut graphs. <http://arxiv.org/abs/2107.00677> (2021).
42. Akshay, V., Rabinovich, D., Campos, E. & Biamonte, J. Parameter concentrations in quantum approximate optimization. *Phys. Rev. A* **104**, L010401 (2021).
43. Rabinovich, D., Sengupta, R., Campos, E., Akshay, V. & Biamonte, J. Progress towards analytically optimal angles in quantum approximate optimisation. <http://arxiv.org/abs/2109.11566> (2021).
44. Basso, J., Farhi, E., Marwaha, K., Villalonga, B. & Zhou, L. The quantum approximate optimization algorithm at high depth for MaxCut on large-girth regular graphs and the Sherrington-Kirkpatrick model. <http://arxiv.org/abs/2110.14206> (2021).
45. Herrman, R., Ostrowski, J., Humble, T. S. & Siopsis, G. Lower bounds on circuit depth of the quantum approximate optimization algorithm. *Quant. Inf. Process.* **20**(2), 1–17 (2021).
46. Herrman, R. *et al.* Globally optimizing QAOA circuit depth for constrained optimization problems. *Algorithms* **14**, 10 (2021).
47. Li, G., Ding, Y. & Xie, Y. Tackling the qubit mapping problem for NISQ-era quantum devices. <http://arxiv.org/abs/1809.02573> (2019).
48. Marcos, Y. S., Vinícius, F. D. S., Caronline, C. & Fernando, M. Q. P. Qubit allocation as a combination of subgraph isomorphism and token swapping. *Proc. ACM Program. Lang.* **3**(OOPSLA, Article 120) (2019).
49. Zulehner, A., Paler, A. & Wille, R. An efficient methodology for mapping quantum circuits to the IBM QX architectures. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **38**, 1–10 (2019).
50. Nannicini, G., Bishop, L. S., Gunluk, O. & Jurcevic, P. Optimal qubit assignment and routing via integer programming. <http://arxiv.org/abs/2106.06446v3> (2021).
51. Wang, S. *et al.* Noise-induced barren plateaus in variational quantum algorithms. <http://arxiv.org/abs/2007.14384v4> (2021).
52. Quiroz, G. *et al.* Quantifying the impact of precision errors on quantum approximate optimization algorithms. <http://arxiv.org/abs/2109.04482> (2021).
53. Xue, C., Chen, Z.-Y., Yu-Chun, W. & Guo, G.-P. Effects of quantum noise on quantum approximate optimization algorithm. *Chin. Phys. Lett.* **38**, 030302 (2021).
54. Koczor, B. Dominant eigenvector of a noisy quantum state. <http://arxiv.org/abs/2104.00608> (2021).
55. Majumdar, R. *et al.* Optimizing ansatz design in QAOA for Max-cut. <http://arxiv.org/abs/2106.02812v3> (2021).
56. O'Gorman, B., Huggins, W. J., Rieffel, E. G. & Whaley, K. Generalized swap networks for near-term quantum computing. <http://arxiv.org/abs/1905.05118> (2019).
57. Kivlichan, I. D. *et al.* Quantum simulation of electronic structure with linear depth and connectivity. *Phys. Rev. Lett.* **120**, 110501 (2018).
58. Brendan McKay. <https://users.cecs.anu.edu.au/~bdm/data/graphs.html>.
59. Nielsen, M. A. & Chuang, I. Quantum computation and quantum information (2002).
60. Jurcevic, P. *et al.* Demonstration of quantum volume 64 on a superconducting quantum computing system. *Quant. Sci. Technol.* **6**, 025020 (2021).
61. Rajakumar, J., Moondra, J., Gupta, S. & Herold, C. D. Generating target graph couplings for QAOA from native quantum hardware couplings. <http://arxiv.org/abs/2011.08165> (2020).
62. Quantum optimization for maximum independent set using Rydberg atom arrays. <http://arxiv.org/abs/1808.10816> (2018).

63. McCaskey, A. J., Dumitrescu, E. F., Liakh, D., Feng, W. & Humble, T. S. A language and hardware independent approach to quantum-classical computing. *SoftwareX* 7, 245–254 (2018).
64. McCaskey, A. J., Lyakh, D. I., Dumitrescu, E. F., Powers, S. S. & Humble, T. S. XACC: A system-level software infrastructure for heterogeneous quantum-classical computing. <http://arxiv.org/abs/1911.02452> (2019).

Acknowledgements

This work was supported by the Defense Advanced Research Project Agency ONISQ program under award W911NF-20-2-0051. J. Ostrowski acknowledges the Air Force Office of Scientific Research award, AF-FA9550-19-1-0147. G. Siopsis acknowledges the Army Research Office award W911NF-19-1-0397. J. Ostrowski and G. Siopsis acknowledge the National Science Foundation award OMA-1937008. This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan. (<http://energy.gov/downloads/doe-public-access-plan>).

Author contributions

P.C.L. contributed to computations, design, analysis, and writing of the manuscript. T.N. contributed to computations, analysis, and writing. A.S. and A.M. contributed to computations. R.H., J.O., G.S., and T.S.H. contributed to design and analysis; R.H. and T.S.H. also contributed to writing.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary Information The online version contains supplementary material available at <https://doi.org/10.1038/s41598-022-14767-w>.

Correspondence and requests for materials should be addressed to P.C.L.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

© UT-Battelle, LLC 2022