Science Advances
AAAS

Supplementary Materials for

Quantum enhanced greedy combinatorial optimization solver

Maxime Dupont *et al.*

Corresponding author: Maxime Dupont, mdupont@rigetti.com; Matthew J. Reagor, matt@rigetti.com

Sci. Adv. **9**, eadi0487 (2023)
DOI: 10.1126/sciadv.adi0487

This PDF file includes:

S1 Decomposition of Two-qubit Gates with $\sqrt{i\text{SWAP}}$

In Fig. 5 of the main text, we showed decompositions of the parametric two-qubit gates Rzz(φ) and Rzz$(\varphi) \times$ SWAP in terms of the two-qubit gate $\sqrt{i\text{SWAP}}$ and one-qubit gates $U_{1\ldots 14}$. Here, we give explicit forms of the U_i in terms of hardware-native one-qubit gates Rz$(\theta \in \mathbb{R}) = \exp(-iZ\theta/2)$ and Rx$(\phi \in \mathbb{Z}) = \exp(-iX\phi\pi/4)$ where X and Z are Pauli operators. The one-qubit gates in the decomposition of Rzz(φ) from Fig. 5a are,

$$\begin{aligned}
U_1 &= \text{Rz}(\pi)\text{Rx}(1)\text{Rz}(\alpha)\text{Rx}(1)\text{Rz}(\pi/2), \\
U_2 &= \text{Rz}(\pi/2)\text{Rx}(1)\text{Rz}(\pi/2), \\
U_3 &= \text{Rx}(1)\text{Rz}(\beta)\text{Rx}(1), \\
U_4 &= \text{I}, \\
U_5 &= \text{Rz}(\gamma)\text{Rx}(1)\text{Rz}(\alpha)\text{Rx}(1)\text{Rz}(\pi), \\
U_6 &= \text{Rz}(\gamma)\text{Rx}(1)\text{Rz}(\pi/2),
\end{aligned} \tag{S1}$$

where $\alpha = \sin^{-1}\left(\tan\frac{\tilde{\varphi}}{2}\right)$, $\beta = 2\arccos\left(\sqrt{2}\sin\frac{\tilde{\varphi}}{2}\right)$, $\gamma = \frac{\pi}{2}\text{sgn}\left(\frac{\pi}{2} - |\varphi|\right)$, $\tilde{\varphi} = \text{mod}\left(\varphi + \frac{\pi}{2}, \pi\right) - \frac{\pi}{2}$, and I is the identity. Note that gates in each line in Eq. (S1) are applied right to left. The one-qubit gates in the decomposition of Rzz(φ) $\times$ SWAP from Fig. 5B are,

$$\begin{aligned}
U_7 &= \mathrm{Rz}(\kappa_1)\mathrm{Rx}(1)\mathrm{Rz}(\kappa_1),\\
U_8 &= \mathrm{Rz}(\pi/2)\mathrm{Rx}(1)\mathrm{Rz}(\pi/2),\\
U_9 &= \mathrm{Rz}(\kappa_1)\mathrm{Rx}(1)\mathrm{Rz}(\zeta_1),\\
U_{10} &= \mathrm{Rz}(\pi/2)\mathrm{Rx}(1)\mathrm{Rz}(\zeta_2),\\
U_{11} &= \mathrm{Rz}(\kappa_1)\mathrm{Rx}(1)\mathrm{Rz}(\alpha)\mathrm{Rx}(1)\mathrm{Rz}(\kappa_1),\\
U_{12} &= \mathrm{Rz}(\pi/2)\mathrm{Rx}(1)\mathrm{Rz}(\beta)\mathrm{Rx}(1)\mathrm{Rz}(\pi/2),\\
U_{13} &= \mathrm{Rz}(\kappa_2)\mathrm{Rx}(1)\mathrm{Rz}(\zeta_3)\mathrm{Rx}(1)\mathrm{Rz}(\kappa_1),\\
U_{14} &= \mathrm{Rz}(\kappa_3)\mathrm{Rx}(1)\mathrm{Rz}(\zeta_4)\mathrm{Rx}(1)\mathrm{Rz}(\pi/2).
\end{aligned} \tag{S2}$$

For simplicity, below we give the values for the gate angles only for $0 \le \varphi \le \frac{\pi}{2}$:

$$\begin{aligned}
\alpha &= \arccos\left[\frac{\cos\varphi + \sin\varphi - 1 + \sqrt{\sin 2\varphi}}{\sqrt{2}}\mathrm{sign}\left(\frac{\pi}{4} - \varphi\right)\right] - \pi,\\
\beta &= \arccos\left(\frac{\cos\varphi + \sin\varphi - 1 - \sqrt{\sin 2\varphi}}{\sqrt{2}}\right) - \pi,\\
\lambda &= -\arccos\sqrt{\frac{1 + \tan\frac{\varphi}{2}}{2}},\\
\psi &= \arccos\sqrt{\frac{1}{1 + \tan\frac{\varphi}{2}}},\\
\zeta_1 &= (\lambda + \psi) + \frac{\pi}{2},\\
\zeta_2 &= (\lambda - \psi) + \frac{\pi}{2},\\
\zeta_3 &= (\lambda + \psi) - \pi,\\
\zeta_4 &= (\lambda - \psi) - \pi,\\
\kappa_1 &= \kappa_2 = \kappa_3 = \frac{\pi}{2}.
\end{aligned} \tag{S3}$$

S2 Number of Native Gates Executed on Hardware

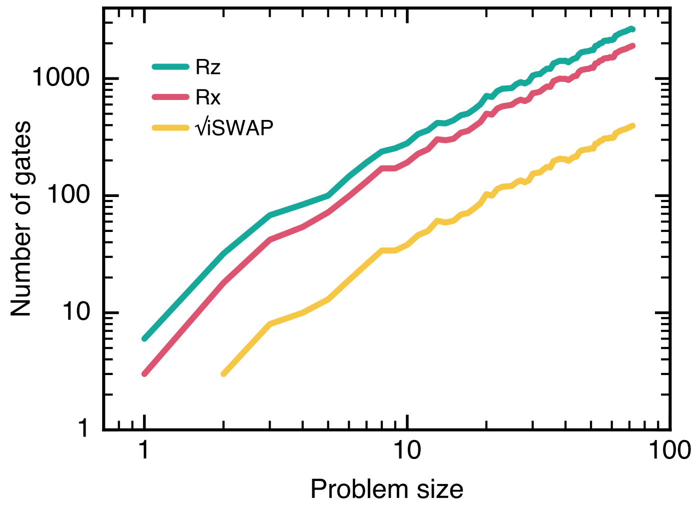

Figure S1: Number of native gates executed on hardware. Number of native one-qubit Rz and Rx gates and two-qubit $\sqrt{i\text{SWAP}}$ gates executed on the hardware as a function of the problem size (related to the iteration step).

In Fig. S1, we show the number of native one-qubit Rz and Rx gates and two-qubit $\sqrt{i\text{SWAP}}$ gates executed on the hardware as a function of the problem size up to $N = 72$. The number of gates scales linearly with the problem size.

S3 Random Sampling and Extreme Value Distribution

In the main text we compared the approximation ratio of the quantum-enhanced greedy algorithm with the approximation ratio of bit strings drawn randomly from a uniform distribution. The cost for a bit string $\mathbf{B} = (B_1, \cdots, B_N)$ is $C = \sum_{i=1,j<i}^{N} w_{ij} Z_i Z_j$ where $Z_i = (-1)^{B_i}$. The average value of C is 0 for random B_i, therefore the average approximation ratio of random samples is $r = 1/2$ (see Methods). Here, we elaborate on the distribution of C for random B_i, and the expected *best* random guess from *many* random samples.

For random bits B_i and weights w_{ij}, each term in the cost is a Bernoulli random variable with mean 0 and variance 1. The cost C, which is the sum of these random variables, is a normally distributed random variable with mean 0 and variance $N(N-1)/2$ as $N \to +\infty$ [see Fig. S4].

Suppose we draw M random bit strings and keep the one which has the minimum cost C. The probability density for the best random guess converges for sufficient samples to the type-I generalized extreme value distribution, also called the Gumbel distribution, $C \sim \text{Gumbel}(\mu_M, \sigma_M)$, where $\mu_M = \sqrt{N(N-1)/2} \cdot \Phi^{-1}(1 - 1/M)$ and $\sigma_M = \sqrt{N(N-1)/2} \cdot \Phi^{-1}(1 - 1/eM) - \mu_M$. Φ^{-1} is the inverse of the cumulative distribution function of the normal distribution. The mean value of the Gumbel distribution is,

$$C_{\text{best-rnd}}(N, M) \simeq -\left(1 + \frac{\gamma}{\ln M}\right) \sqrt{\frac{N(N-1)}{2} \ln\left(\frac{M^2}{2\pi \ln \frac{M^2}{2\pi}}\right)}, \tag{S4}$$

where $\gamma \simeq 0.577215...$ is the Euler-Mascheroni constant. Eq. (S4) is expected to hold for $1 \ll M \lesssim 2^N$. The upper limit for M is set by the fact that when M saturates this limit, the best random guesses sample from the tail of the distribution of C, where the assumption that C is normally distributed is no longer valid.

The quantum-enhanced algorithm has a total of N iterative steps. For each of these steps, the embedded QAOA scanned a grid of angles of size 16×16 and sampled 256 bit strings for each pair of angles. This leads to a total of $N \times 16 \times 16 \times 256 = 2^{16} N$ bit strings generated in practice for a given problem instance. This is about $M \simeq 10^6$ bit strings generated for problem sizes $N = 40$, 56, and 72. Generating at random this many bit strings and keeping the best one would lead to an average approximation ratio $r \simeq 0.8$-0.9 according to Eq. (S4) and as shown in Fig. S2. For comparison, achieving the average approximation of $r \simeq 0.92$ of the quantum-enhanced algorithm with random sampling would require generating an average $M \simeq 10^7$, 10^{10}, and $M = 10^{12}$ random bit strings for $N = 40$, 56, and 72, respectively [Fig. S2].

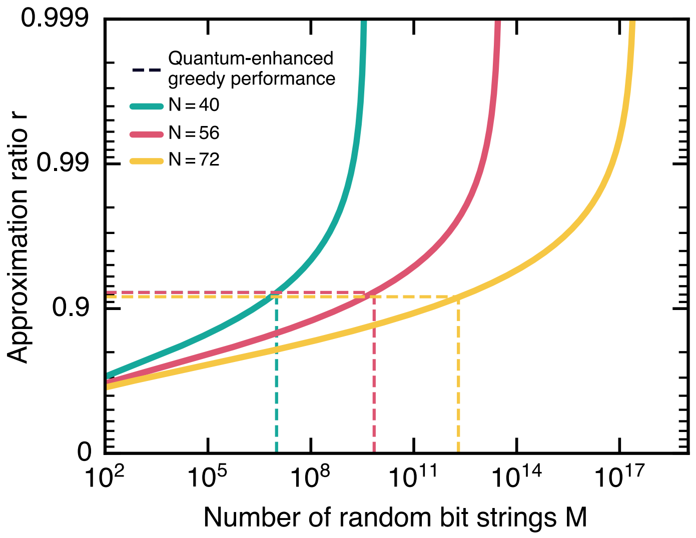

Figure S2: Performance of random sampling through the lens of extreme value statistics. Approximation ratio r based on Eq. (S4) as a function of the number M of uniformly drawn random bit strings for different problem sizes $N = 40$, 56, and 72. Dashed lines report the performance for the quantum-enhanced greedy algorithm and the corresponding number of bitstrings needed to be drawn at random on average to match that performance.

S4 Finite-Size Classical Greedy Baseline for SK Problem Instances

When provided with random bit strings, the quantum-enhanced greedy algorithm maps to a classical greedy algorithm for solving the optimization problem of interest. We proved in Methods that for an infinite-size Sherrington-Kirkpatrick (SK) instance, the classical greedy algorithm has an approximation ratio of $r \simeq 0.848497...$. Here, we run the algorithm on finite-size SK problems to get a more realistic number with respect to the finite sizes studied in this work.

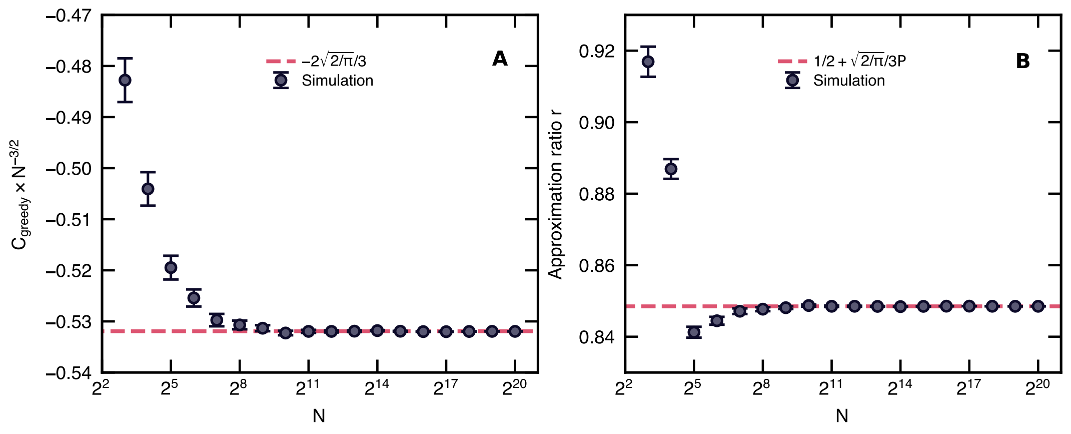

Figure S3: Performance of the classical greedy baseline. **A.** Average ground state energy density obtained by the classical greedy algorithm on $1,000$ SK problem instances. As $N \to +\infty$, it converges to $-2\sqrt{2/\pi}/3$. **B.** Average approximation ratio r versus the problem size N for the classical greedy algorithm. Each data point is averaged over $1,000$ random SK problem instances. For $N \to +\infty$, we expect that $r = 1/2 + \sqrt{2/\pi}/3\mathrm{P} \simeq 0.848497...$ where P is the Parisi constant (see Methods). For $N \leq 16$, the exact ground state is computed by brute force. For larger sizes, the ground state is approximated as described in Methods. Error bars indicate one standard deviation.

S5 Average Spectrum of Sherrington-Kirkpatrick Problem Instances

We consider SK problem instances with random weights ± 1. For each problem instance of size N, we iterate over all possible 2^N bit strings and store the corresponding cost. The nature of the weights ± 1 strongly restricts the possible values of the cost, making it possible to store the costs $C \in \mathbb{Z}$ as a map $'\mathrm{map}[C] = \ \mathbb{N}'$ with $\mathbb{N}$ counting the number of bit strings with the given cost, and which is updated during the iteration over the bit strings. The set of all costs is called the spectrum.

For various problem sizes, we generate $1,000$ random problem instances. We compute their individual spectra, average them, and plot the result as a probability distribution $P(C)$ in Fig. S4. We find that it is symmetric around $C = \ 0$ and that the bulk are fitted well by a

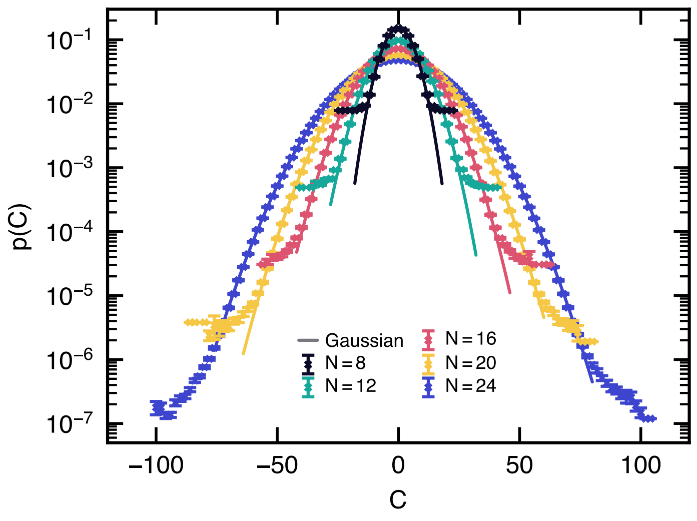

Figure S4: Spectrum of Sherrington-Kirkpatrick models. For each problem size $N = 8$, 12, 16, 20, and 24, we compute the spectra for $1,000$ randomly generated problems. The spectra are then averaged and plotted as a probability distribution. Error bars indicate one standard deviation.

Gaussian. This shows that, as expected, $C_{\text{min}} \simeq -C_{\text{max}}$ and that a randomly generated bit string would be symmetrically distributed between the two extrema of the spectrum, leading to an average approximation ratio $r \simeq 1/2$ for solving SK models by random sampling.

For various problem sizes, we generate $1,000$ random problem instances. For each of them, we count the number of bit strings having the optimal cost C_{min}. As shown in Fig. S5, the average (which is ≈ 3) is roughly independent of N. Therefore, the ground states of SK ± 1 problems are not massively degenerate and appear to occupy a fraction of the total space that is exponentially suppressed in N. Note that it is not possible to infer anything about the exact scaling with N for large N due to the small sizes accessible.

S6 Additional Details Regarding Matrix Product State Simulations

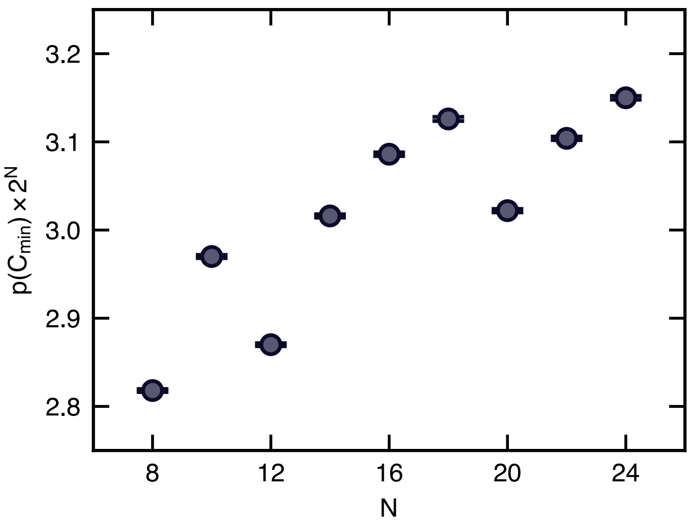

Figure S5: Degeneracy of the optimal solutions. Probability of finding an optimal solution multiplied by the total number of valid solutions as a function of the problem size N. In other words, this is the average degeneracy of the optimal solution versus N. Each data point is averaged over $1,000$ randomly generated problem instances. Error bars indicate one standard deviation.

The quantum-inspired classical algorithm of the main text simulates shallow circuits up to $N = 72$ qubits using a tensor network approach based on matrix product states (*57*). Such circuits are truncated one-layer QAOA circuit with two swap cycles embedded into a one-dimensional lattice with open boundary conditions, as shown in Fig. S6. The circuits are shallow enough to be executed exactly with a relatively low bond dimension (the bond dimension is a control parameter of a matrix product state simulator), independent of the number of qubits involved. We embed the graph problem randomly onto the one-dimensional topology of the matrix product state. The truncated one-layer QAOA circuit of Fig. S6 loads at random a total of $2(N-1)$ edges of the graph problem (to be compared with the total number of edges $N(N-1)/2$). It has been shown that simulating, even approximately, generic QAOA circuits with matrix product states is hard (*58, 59*).

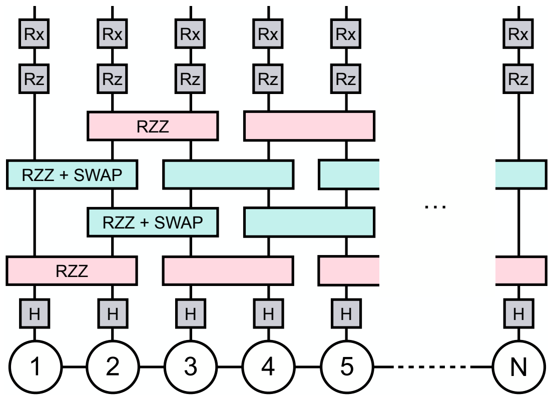

Figure S6: Quantum circuit simulated with matrix product states. A graph problem of N nodes is embedded at random onto the linear topology of the matrix product state simulating N qubits (*57*). A total of four layers of two-qubit gates are used in a brick wall pattern on even/odd edges.

S7 Data: Approximation Ratio Versus Problem Size

Problem size	**Quantum**	**Q-inspired**	**Random**	**QAOA$_1$** (*29*)	**Greedy**	**SDP** (*41–43*)
$N = 8$	$0.97(2)$	$0.989(9)$				
$N = 24$	$0.97(2)$	$0.959(5)$				
$N = 40$	$0.93(2)$	$0.964(4)$				
$N = 56$	$0.93(2)$	$0.963(4)$				
$N = 72$	$0.92(1)$	$0.954(3)$				
$N = \infty$			$1/2$	$0.698688...$	$0.848497...$	$0.917090...$

Table S1: Approximation ratio. Approximation ratio r for different methods and different problem sizes N. The data is the same as in the figure of the main text showing the approximation ratio versus the problem size.

We provide in Table S1 the data from the figure of the main text showing the approximation ratio versus the problem size.

S8 Exact Small-Scale Simulations with Nontruncated QAOA Circuits

We worked with truncated QAOA circuits in the main text due to hardware limitations. Here, we provide additional data for the quantum-enhanced greedy algorithm based on exact small-scale classical simulations of the QAOA with a complete (nontruncated) circuit.

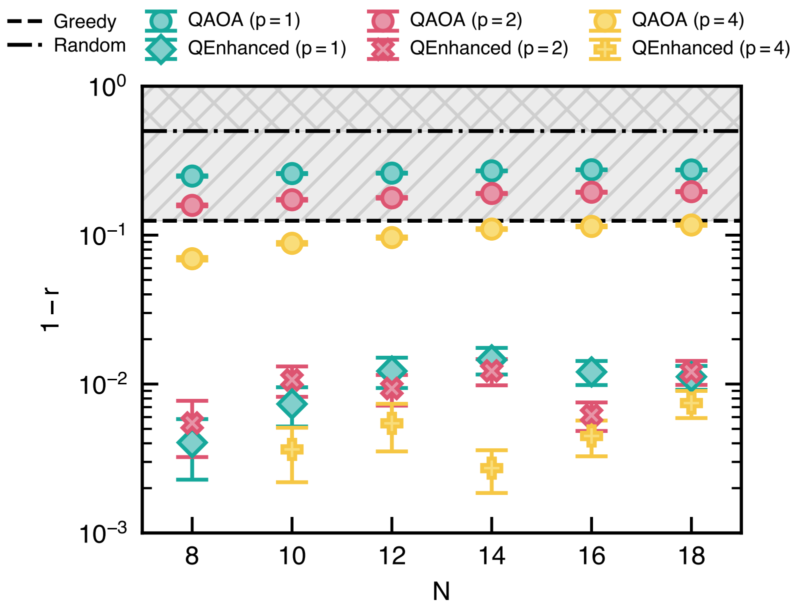

Figure S7: Performance of the quantum-enhanced greedy algorithm based on exact small-scale QAOA simulations. $1 - r$, with r the average approximation ratio over 100 random SK instances, as a function of the problem size N. For each of these instances, the complete (non-truncated) QAOA is executed with various number of layers $p = 1$, 2, and 4. Results are shown for the QAOA and the quantum-enhanced greedy algorithm. Two baselines are represented by the hatched regions: The classical greedy baseline with $1 - r \simeq 0.12$ for the largest sizes considered (see Fig. S3) and the random sampling baseline with $1 - r = 1/2$. Error bars indicate one standard deviation.

Data are plotted in Fig. S7. The performance of the QAOA increases with the number of layers p. It is systematically above the random sampling baseline $1 - r = 1/2$, but barely passes the classical greedy baseline $1 - r \simeq 0.12$ at $p = 4$, and achieves systematically worse performance for $p < 4$. On the other hand, the quantum-enhanced greedy algorithm performs better than the classical greedy baseline. In addition, we observe that improving the performance of the underlying QAOA by increasing the number of layers p leads to an improvement in the performance of the quantum-enhanced greedy algorithm. The improvement over the QAOA or the classical greedy baseline is between $\times 10$ and $\times 50$.

S9 Quantumness in the Second Versus the Third Step of the Iterative Algorithm

In the main text, we studied the case of the quantum-enhanced algorithm with quantum-generated bit strings, and the randomized classical greedy algorithm with randomly generated bit strings. Here, we consider a mix-and-match scenario for step 2 (selection) and step 3 (decision) of the algorithm, where in each of these steps, the bit strings may be generated randomly or from a quantum computer. Specifically, we used quantumly generated bit strings in both steps 2 and 3 in the quantum algorithm in the main text, and using randomly generated bit strings in both steps 2 and 3 maps to the randomized classical greedy algorithm in the main text. Here, we additionally consider two mores cases: one case where we use quantumly generated bit strings for the selection step and randomly generated bit strings for the freezing step, and another case with vice versa.

We consider exact small-scale simulations using QAOA with one layer ($p = 1$). Data are plotted in Fig. S8 and show that the use of quantum-generated bit strings is comparatively very beneficial to inform the selection step. Using random or quantum-generated bit strings in the freezing step give nearly equal answers when the selection step is done with quantum-generated bit strings. We explain this as follows. If a variable i is selected, and we use random bit strings in the freezing step, expectation values of all observables that do not involve Z_i are 0. Therefore, the only relevant nonzero term in the cost is $v_i Z_i$, thus Z_i is frozen to $Z_i = -\text{sign}(v_i)$. If

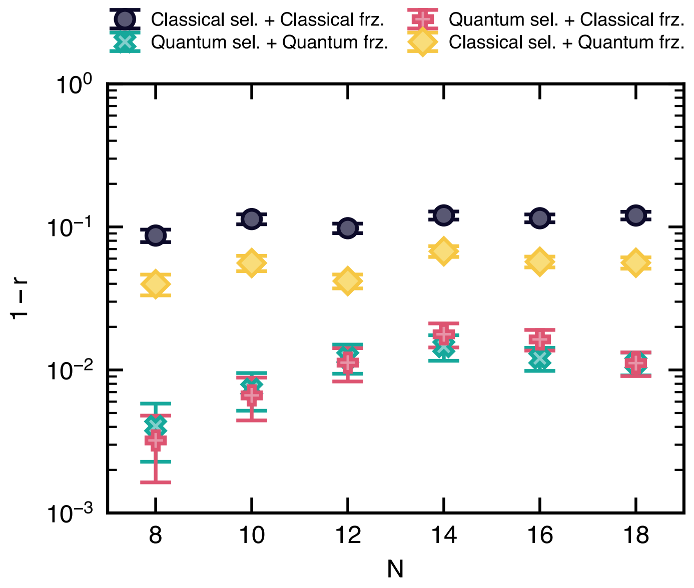

Figure S8: Contribution of bit string selection and freezing steps to the performance of the quantum-enhanced greedy algorithm. The algorithm includes steps which select and freeze bit strings which can randomly generated or be the output of a quantum computer. The greedy baseline corresponds to the bit strings being random in both steps while the algorithm works with the output of a quantum computer in both steps. Here, the intermediate scenarios are considered. The data are $1 - r$, with r the average approximation ratio over a 100 random SK instances, as a function of the problem size N. For each of these instances, the complete (nontruncated) QAOA is emulated classically a single layer ($p = 1$). Error bars indicate one standard deviation.

instead, quantum-generated bit strings are used in the freezing step, the relevant terms in the cost are both one-body term $v_i Z_i$ and two-body terms $|w_{ij} Z_i \langle Z_j \rangle|$. The two-body terms may be nonzero; However, they tend to be small relative to $|v_i|$ in the intermediate iterative steps. Therefore, predominantly, the freezing again leads to $Z_i = -\text{sign}(v_i)$. This need not be true at arbitrary p, but appears to be true at $p = 1$.

S10 Performance Versus Runtime

We investigate the performance versus runtime of different algorithms for a single typical $N = 72$ variables Sherrington-Kirkpatrick instance with random ± 1 weights.

We implement classical heuristics such as tabu search (*60*) and simulated annealing (*61, 62*), a classical semidefinite programming solver (*41–43*), the classical randomized greedy algorithm developed in this work as well as its quantum-enhanced version executed on quantum hardware and executed through classical matrix product state simulations (see the main text and the Methods). We also report data for the truncated one-layer QAOA executed at the first iterative step of the quantum-enhanced greedy algorithm.

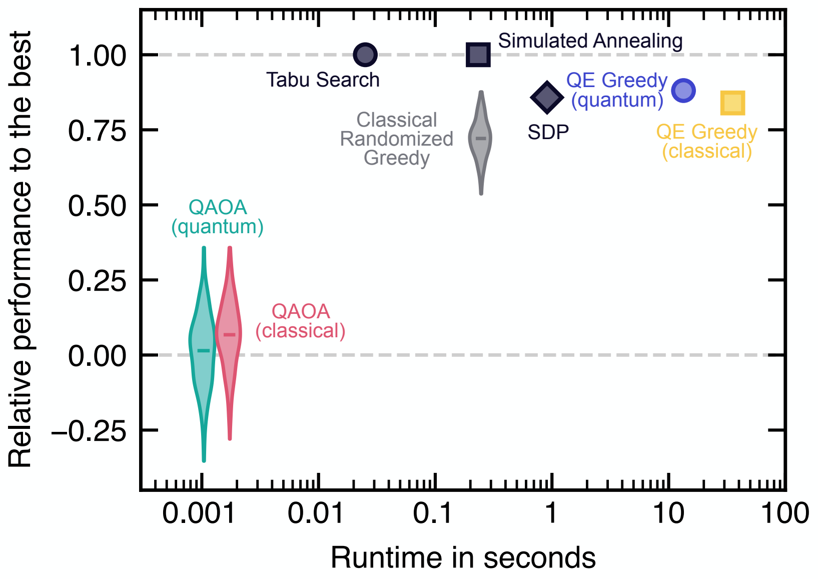

Figure S9: Relative performance versus runtime for different approaches. A typical $N = 72$ variables Sherrington-Kirkpatrick instance with random ± 1 weights is considered. Classical execution is performed on a single core of an Intel Skylake E5-2686 v5 3.1GHz processor.

Data are plotted in Fig. S9. Tabu search and simulated annealing find the best answer in the least amount of time, with standard parameterization. The classical randomized greedy algorithm is a randomized algorithm that will return a different solution at every run. The distribution (over 256 independent runs) shows the distribution of solutions and the runtime is that of obtaining a single solution. For the truncated one-layer QAOA, we also display the distribution of solutions (over 256 shots), assuming optimal angles. The runtime is that of obtaining a single bit string by running a single QAOA circuit at optimal angles. Quantum refers to running on Rigetti Aspen-M-3 superconducting processor and classical to quantum-inspired classical simulations based on matrix product states (see the Methods). Finally, the quantum-enhanced greedy algorithm is displayed: The iterative process with the freezing of a single variable at a time requires running the QAOA $N = 72$ times to get to the final solution. At each step, we assume we know the optimal angles. The runtime is that of collecting 256 shots for each of the 72 iterative steps at optimal angles. We note that for this specific problem instance, the quantum run returned a better solution than the classical (quantum-inspired) simulation.

A gap of a few orders of magnitude need to be closed for the quantum-enhanced greedy algorithm to be competitive with state-of-the-art classical methods. As noted in the main text, the complexity of the quantum-enhanced greedy algorithm is $O[(N/K)N_{\text{edges}}]$ with N_{edges} the number of two-body terms in the graph problem and K the number of variables frozen at each iteration step. Here, we emphasize that minimizing runtime was not a goal of this work. Reducing the runtime of the underlying QAOA is one direction: One could imagine an adaptive scheme for setting the number of shots at each iterative step. Another direction would seek to reduce the length of the iterative process by, e.g., implementing freezing strategies considering multiple variables at a time (K of them at once) and brute-forcing the problem once it becomes small enough. We believe this would bring for $N = 72$ the quantum-enhanced greedy algorithm in the 0.1 to 1 seconds range of runtime. Moreover, SK problem instances have the largest possible number of edges $N_{\text{edges}} \sim O(N^2)$ because of the all-to-all connectivity. Sparser graphs with, e.g., $N_{\text{edges}} \sim O(N)$, might lead to a better absolute and relative runtime for the quantum-enhanced greedy algorithm.

S11 Dealing with Hard Constraints by Example: Portfolio Optimization

S11.1 Problem Definition

We take as an example the portfolio optimization problems of Ref. (*49*). The goal is to build a portfolio maximizing the potential return while minimizing the volatility from a given basket of N assets. Whether an asset is selected for the portfolio is encoded as a binary variable. Without entering into the details (refer to Ref. (*49*)), the problem takes the form of minimizing an objective function with binary variables and nonzero scalar parameters v_i and w_{ij}, as per the definition of the cost function (Eq. (1)) in the main text. In addition to the minimization, there are two hard constraints on what makes a valid portfolio. They are of the form,

$$\sum\nolimits_{i=1}^{N} Z_i \leq A, \quad \text{and} \quad \sum\nolimits_{i=1}^{N} \mu_i Z_i \geq B, \quad \text{with } A, B, \mu_i \in \mathbb{R}. \tag{S5}$$

The first inequality constrains the maximum size of the portfolio and the second one the minimum expected return from the portfolio. The parameters A, B, and μ_i are provided as part of the problem.

S11.2 Hard Constraints in the Quantum-Enhanced Greedy Algorithm

The hard constraints of Eq. (S5) can be enforced classically as part of the quantum-enhanced greedy algorithm, thus ensuring that the final bit string is necessarily a valid solution. Moreover, it is not required to introduce auxiliary slack variables to map the inequalities into equalities, making the algorithm even more friendly for near term quantum devices.

The algorithm presented in the main text is modified as follows to account for the constraints. In the freezing step, a variable otherwise identified as being a candidate for freezing, will not be frozen if it would take the potential final solution out of the space of valid bit strings. However, this may only be possible for certain types of constraints, including those of Eq. (S5).

Different strategies can be envisioned to adapt the freezing decision process. For instance, if freezing to a classical value is not possible, then the other value can be selected despite its higher cost. One could also go back to the election step and select another variable for freezing.

Whereas the hard-constraints will be fulfilled independently of the underlying quantum algorithm (e.g., QAOA or adiabatic quantum evolution), it might be advantageous for these algorithms to favor in-constraint bit strings for their output. A possible approach is to design quantum circuits working within the in-constraint space, as discussed in general in Ref. (*12*) and more specifically for Hamming constrained problems in Ref. (*63*), but it is practically infeasible on current quantum hardware due to noise. Another typical approach uses penalty terms that will disfavor the appearance of out-of-constraint bit strings, but implementing them on near-term devices and tuning their strength can be challenging. Another strategy might be to classically filter the out-of-constraint bit strings returned by the quantum algorithm, if any. However, this requires that the density of valid bit strings is sufficiently high.

S11.3 Results

Ref. (*49*) introduced Zeno dynamics embedded into QAOA to deal with hard-constraints in the form of Eq. (S5). The authors generated a problem of each size $N = 4, \dots, 10$ and solved it using a one-layer QAOA circuit with Zeno dynamics. The corresponding data points are reported in Fig. S10, with an approximation ratio of about $r \simeq 0.74$, independent of N. Here, the approximation ratio is defined with respect to the best and worst solutions of the in-constraint space.

On the exact same set of problems, we run the quantum-enhanced greedy algorithm. We use a one-layer QAOA circuit. We use two-body expectation values to inform the selection process. The freezing decision is such that if setting a variable to a classical value makes the solution invalid, the other value is selected despite its higher cost. The constraints on all

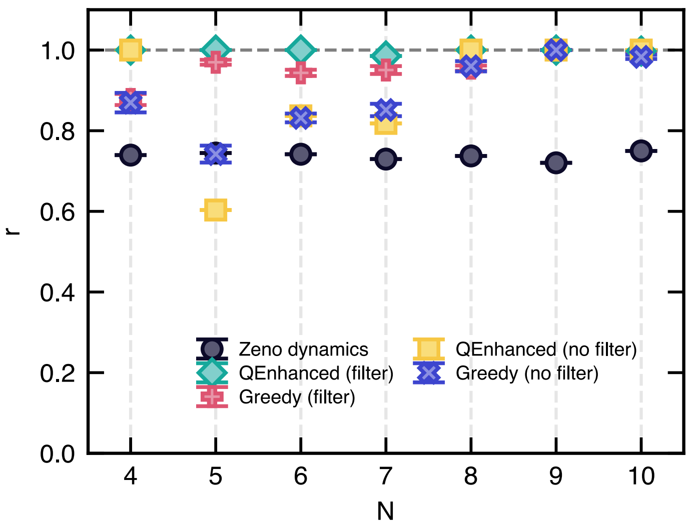

Figure S10: Performance of different approaches for constrained portfolio optimization problems from Ref. (*49*). Average approximation ratio for different problem sizes N computed from classical emulations. Each size N corresponds to a given problem provided by Ref. (*49*). The Zeno dynamics data points were extracted from Ref. (*49*) and serve as a reference. The performance of the quantum-enhanced greedy algorithm (based on a one-layer QAOA circuit) and its classical greedy counterpart are also displayed, with and without filtering. The classical greedy baseline data points are averaged over 100 trials. Error bars indicate one standard deviation.

problem instances are relatively loose since bit strings drawn at random have more than a 50% chance of being valid. This makes it possible to classically filter bit strings outputted from the QAOA. Here, filtering means discarding bit strings not fulfilling the hard constraints of Eq. (S5). For example, an $N-$bit string $\{1, 1, 1, 1, \dots 1\}$ will not fulfill the first constraint of Eq. (S5) $\sum_{i=1}^{N} Z_i \leq A$ for $A = N/2$ as $\sum_{i=1}^{N} Z_i = N$. Thu, such a bit string would be discarded.

We try the filtered and unfiltered versions as well as the classical greedy baseline where bit strings are generated at random. Results are plotted in Fig. S10. We find that filtering bit strings leads to better performance. We also find that the classical greedy baseline systematically outperforms the one-layer QAOA circuit with Zeno dynamics data and that it has a very high approximation ratio $r \gtrsim 0.95$ for the sizes considered.

S11.4 Outlook

It would be precipitate to draw conclusions about the absolute performance of the quantum-enhanced algorithm in the presence of constraints since: (i) the classical greedy baseline performs very well, making it difficult to discern any significant improvements; and (ii) the problem set is very limited, with just one instance per problem size and all problems being relatively small ($N \leq 10$). Further work is required to extend the analysis to a statistically significant problem set. Moreover, it would be interesting to explore the different parameters entering the quantum-enhanced algorithm. Tighter constraints would not enable filtering out bit strings and it would be worthwhile to explore problems (not necessarily related to portfolio optimization) for which sampling the valid space of solutions is difficult due to the problem size. Finally, we note that a QAOA circuit with Zeno dynamics could be embedded in the quantum-enhanced algorithm presented here.

S12 Classical Greedy Baseline for Other Graphs

The randomized greedy algorithm is iterative from step $\ell = 1$ to $\ell = N$ for a graph with N nodes. At step ℓ, we select an active variable at random. This means that there are $N-1$ other variables in the problem, including $\ell - 1$ frozen ones and $N - \ell$ active ones. The goal is to find the classical value to which freeze this variable to minimize the objective function [Eq. (1)]. In the following, we show how one can derive the classical greedy baseline analytically for other problems than the SK instances considered in the main text. The list is nonexhaustive.

S12.1 Ring Graph with Random ± 1 weights

We consider a ring graph with random weights $w_{i,i+1} = \pm 1$ on the edges [Eq. (1)]. We now compute the average approximation of the classical greedy baseline for this problem following the same ideas developed for SK problem instances in Methods. For the freezing decision of a randomly selected node, we need to know the probabilities that (a) none of its nearest-neighbors are frozen, that (b) one of its nearest-neighbor is frozen, and that (c) both of its nearest-neighbors are frozen. At step ℓ, these probabilities read,

$$p_a\big(N,\ell\big) = \left(\frac{N-\ell}{N-1}\right)\left(\frac{N-\ell-1}{N-2}\right), \quad (\ell \geq 1) \tag{S6}$$

$$p_b\big(N,\ell\big) = 2\left(\frac{\ell-1}{N-1}\right)\left(\frac{N-\ell}{N-2}\right), \quad (\ell \geq 2) \tag{S7}$$

$$p_c\big(N,\ell\big) = \left(\frac{\ell-1}{N-1}\right)\left(\frac{\ell-2}{N-2}\right) \quad (\ell \geq 3), \tag{S8}$$

where $p_a + p_b + p_c = 1$. The final value of the objective function takes the general form,

$$C_{\text{greedy}} = \sum\nolimits_{\ell=1}^{N} C_\ell = C_1 + C_2 + \sum\nolimits_{\ell=3}^{N} C_\ell, \tag{S9}$$

where, on average, $C_1 = 0$, $C_2 = -p_b(N,\ell=2)$, and $C_{\ell\geq 3} = -p_b(N,\ell) - p_c\big(N,\ell\big)$. Hence, on average, the expectation value obtained for the objective function with the greedy algorithm on the ring graph is,

$$C_{\text{greedy}} = -p_b(N,\ell=2) - \sum\nolimits_{\ell=3}^{N}\left[p_b(N,\ell) + p_c\big(N,\ell\big)\right] = -\frac{2N}{3}. \tag{S10}$$

Since the problem is trivially solved with $C_{\text{min}} = -C_{\text{max}} = -N$, this leads to an average approximation ratio of,

$$r = \frac{1}{2}\left(1 + \frac{C_{\text{greedy}}}{C_{\text{min}}}\right) = \frac{5}{6} \simeq 0.833333... \tag{S11}$$

S12.2 Random 3-Regular Graphs with Random ± 1 weights

We employ the same strategy than for the ring graph to solve random 3-regular graphs with uniform random weights $w_{ij} = \pm 1$ on the edges $\{(i,j)\}$ [Eq. (1)]. In a 3-regular graph, each node has three neighbors, and we assume $N \geq 3$. Therefore, for the freezing decision of a randomly selected node, we need to know the probabilities that (a) none of its nearest-neighbors are frozen, that (b) one of its nearest-neighbor is frozen, that (c) two of its nearest-neighbors are frozen, and that (d) all of its nearest-neighbors are frozen. At step ℓ, these probabilities read,

$$p_a(N,\ell) = \left(\frac{N-\ell}{N-1}\right)\left(\frac{N-\ell-1}{N-2}\right)\left(\frac{N-\ell-2}{N-3}\right), \quad (\ell \geq 1) \tag{S12}$$

$$p_b(N,\ell) = 3\left(\frac{\ell-1}{N-1}\right)\left(\frac{N-\ell}{N-2}\right)\left(\frac{N-\ell-1}{N-3}\right), \quad (\ell \geq 2) \tag{S13}$$

$$p_c(N,\ell) = 3\left(\frac{\ell-1}{N-1}\right)\left(\frac{\ell-2}{N-2}\right)\left(\frac{N-\ell}{N-3}\right), \quad (\ell \geq 3) \tag{S14}$$

$$p_d(N,\ell) = \left(\frac{\ell-1}{N-1}\right)\left(\frac{\ell-2}{N-2}\right)\left(\frac{\ell-3}{N-3}\right) \quad (\ell \geq 4), \tag{S15}$$

where $p_a + p_b + p_c + p_d = 1$. The final value of the objective function takes the general form,

$$C_{\text{greedy}} = \sum_{\ell=1}^{N} C_\ell = C_1 + C_2 + C_3 + \sum_{\ell=4}^{N} C_\ell, \tag{S16}$$

where, on average, $C_1 = 0$, $C_2 = -p_b(N,\ell=2)$, $C_3 = -p_b(N,\ell=3) - p_c(N,\ell=3)$, and $C_{\ell\geq 4} = -p_b(N,\ell) - p_c(N,\ell) - 3p_d(N,\ell)/2$. Hence, on average, the expectation value obtained for the objective function with the greedy algorithm on random 3-regular graphs is,

$$\begin{aligned} C_{\text{greedy}} = &- p_b(N,\ell=2) - p_b(N,\ell=3) - p_c(N,\ell=3) \\ &- \sum_{\ell=4}^{N}\Big[p_b(N,\ell) + p_c(N,\ell) + 3p_d(N,\ell)/2\Big] = -\frac{7N}{8} \end{aligned} \tag{S17}$$

S12.3 Numerical simulations

We perform small-scale noiseless simulations of the quantum-enhanced greedy algorithm for the ring and random 3-regular graphs with random ±1 weights. The simulations are based on a nontruncated one-layer ($p = 1$) QAOA circuit and averaged over 100 randomly generated problem instances. The classical randomized greedy baseline is also displayed. Results are reported in Fig. S11 with the quantum-enhanced greedy algorithm systematically outperforming the classical greedy algorithm.

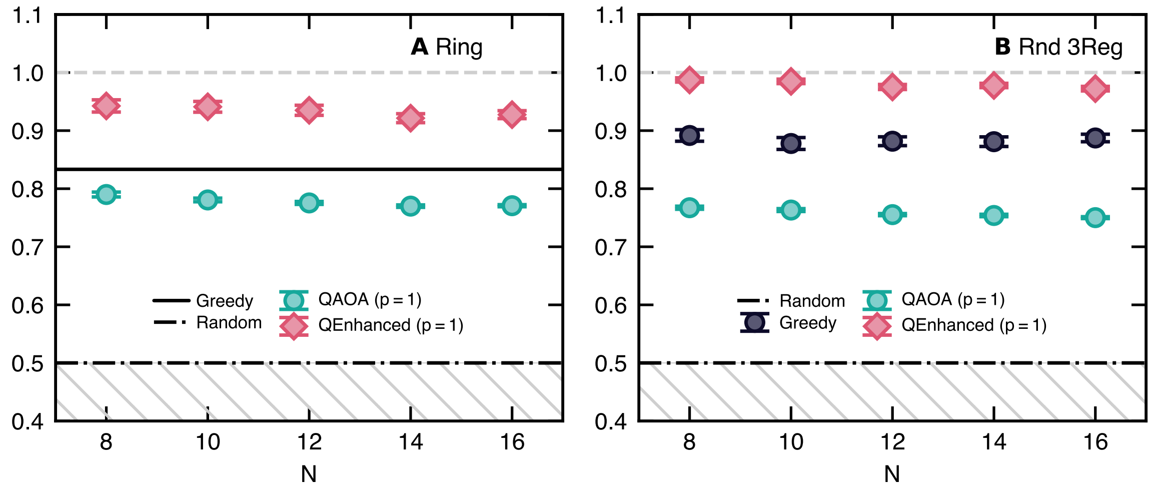

Figure S11: Exact small-scale simulations for ring and random 3-regular graphs. Approximation ratio r, averaged over 100 random problem instances, as a function of the problem size N. For each of these instances, a nontruncated one-layer $(p = 1)$ QAOA is executed. Error bars indicate one standard deviation. **A.** Ring graphs with random ±1 weights. **B** Random 3-regular graphs with random ±1 weights.

REFERENCES AND NOTES

1. E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, D. Preda, A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem. *Science* **292**, 472–475 (2001)
2. E. Farhi, J. Goldstone, S. Gutmann. A quantum approximate optimization algorithm. arXiv :1411.4028 [quant-ph] (2014).
3. E. Farhi, J. Goldstone, S. Gutmann. A quantum approximate optimization algorithm applied to a bounded occurrence constraint problem. arXiv :1412.6062 [quant-ph] (2014).
4. M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, P. J. Coles, Variational quantum algorithms. *Nat. Rev. Phys.* **3**, 625–644 (2021).
5. A. Montanaro, Quantum speedup of branch-and-bound algorithms. *Phys. Rev. Res.* **2**, 013056 (2020).
6. C.-M. Alexandru, E. Bridgett-Tomkinson, N. Linden, J. MacManus, A. Montanaro, H. Morris, Quantum speedups of some general-purpose numerical optimisation algorithms. *Quantum Sci. Technol.* **5**, 045014 (2020).
7. F Barahona, On the computational complexity of Ising spin glass models. *J. Phys. A Math. Theor.* **15**, 3241 (1982).
8. S. Boixo, T. F. Rønnow, S. V. Isakov, Z. Wang, D. Wecker, D. A. Lidar, J. M. Martinis, M. Troyer, Evidence for quantum annealing with more than one hundred qubits. *Nat. Phys.* **10**, 218–224 (2014)
9. TF Rønnow, Z. Wang, J. Job, S. Boixo, S. V. Isakov, D. Wecker, J. M. Martinis, D. A. Lidar, M. Troyer, Defining and detecting quantum speedup. *Science* **345**, 420–424 (2014).
10. D. Venturelli, S. Mandrà, S. Knysh, B. O'Gorman, R. Biswas, V. Smelyanskiy, Quantum optimization of fully connected spin glasses. *Phys. Rev. X* **5**, 031040 (2015).
11. E. Farhi A. W. Harrow. Quantum supremacy through the quantum approximate optimization algorithm . arXiv:1602.07674 [quant-ph] (2016).
12. S. Hadfield, Z. Wang, B. O'Gorman, E. G. Rieffel, D. Venturelli, R. Biswas, From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms* **12** 34 (2019).
13. Otterbach, J. S. et al. Unsupervised machine learning on a hybrid quantum computer . arXiv:1712.05771 [quant-ph] (2017).
14. G. Pagano, A. Bapat, P. Becker, K. S. Collins, A. De, P. W. Hess, H. B. Kaplan, A. Kyprianidis, W. L. Tan, C. Baldwin, L. T. Brady, A. Deshpande, F. Liu, S. Jordan, A. V. Gorshkov, C. Monroe,

Quantum approximate optimization of the long-range Ising model with a trapped-ion quantum simulator. *Proc. Natl. Acad. Sci.* **117**, 25396–25401 (2020).

15. M. P. Harrigan, K. J. Sung, M. Neeley, K. J. Satzinger, F. Arute, K. Arya, J. Atalaya, J. C. Bardin, R. Barends, S. Boixo, M. Broughton, B. B. Buckley, D. A. Buell, B. Burkett, N. Bushnell, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, S. Demura, A. Dunsworth, D. Eppens, A. Fowler, B. Foxen, C. Gidney, M. Giustina, R. Graff, S. Habegger, A. Ho, S. Hong, T. Huang, L. B. Ioffe, S. V. Isakov, E. Jeffrey, Z. Jiang, C. Jones, D. Kafri, K. Kechedzhi, J. Kelly, S. Kim, P. V. Klimov, A. N. Korotkov, F. Kostritsa, D. Landhuis, P. Laptev, M. Lindmark, M. Leib, O. Martin, J. M. Martinis, Jarrod R. Mc Clean, M. M. Ewen, A. Megrant, X. Mi, M. Mohseni, W. Mruczkiewicz, J. Mutus, O. Naaman, C. Neill, F. Neukart, M. Y. Niu, Thomas E. O'Brien, B. O'Gorman, E. Ostby, A. Petukhov, H. Putterman, C. Quintana, P. Roushan, N. C. Rubin, D. Sank, A. Skolik, V. Smelyanskiy, D. Strain, M. Streif, M. Szalay, A. Vainsencher, T. White, Z. Jamie Yao, P. Yeh, A. Zalcman, L. Zhou, H. Neven, D. Bacon, E. Lucero, E. Farhi, R. Babbush, Quantum approximate optimization of non-planar graph problems on a planar superconducting processor. *Nat. Phys.* **17**, 332–336 (2021
16. S. Ebadi, A. Keesling, M. Cain, T. T. Wang, H. Levine, D. Bluvstein, G. Semeghini, A. Omran, J. Liu, R. Samajdar, X.-Z. Luo, B. Nash, X. Gao, B. Barak, E. Farhi, S. Sachdev, N. Gemelke, L. Zhou, S. Choi, H. Pichler, S. Wang, M. Greiner, V. Vuletic, M. D. Lukin, Quantum optimization of maximum independent set using rydberg atom arrays. *Science* **376**, 1209–1215 (2022).
17. T. M. Graham, Y. Song, J. Scott, C. Poole, L. Phuttitarn, K. Jooya, P. Eichler, X. Jiang, A. Marra, B. Grinkemeyer, M. Kwon, M. Ebert, J. Cherek, M. T. Lichtman, M. Gillette, J. Gilbert, D. Bowman, T. Ballance, C. Campbell, E. D. Dahl, O. Crawford, N. S. Blunt, B. Rogers, T. Noel, M. Saffman, Multi-qubit entanglement and algorithms on a neutral-atom quantum computer. *Nature* **604**, 457–462 (2022).
18. E. Pelofske, A. B¨artschi, S. Eidenbenz. Quantum Annealing vs. QAOA: 127 Qubit Higher-Order Ising Problems on NISQ Computers . *arXiv:2301.00520* [quant-ph] (2023).
19. Moses, S. A. *et al.* A Race Track Trapped-Ion Quantum Processor . *arXiv:2305.03828* [quant-ph] (2023).
20. Shaydulin, R. *et al.* Evidence of Scaling Advantage for the Quantum Approximate Optimization Algorithm on a Classically Intractable Problem . *arXiv:2308.02342* [quant-ph] (2023).
21. S. H. Sack D. J. Egger Large-scale quantum approximate optimization on non-planar graphs with machine learning noise mitigation . *arXiv:2307.14427* [quant-ph] (2023).

22. Maciejewski, F. B. *et al.* Design and execution of quantum circuits using tens of superconducting qubits and thousands of gates for dense ising optimization problems . *arXiv:2308.12423* [quant-ph] (2023).
23. Y. Zhu, Z. Zhang, B. Sundar, A. M. Green, C. H. Alderete, N. H. Nguyen, K. R. A. Hazzard,N. M. Linke, Multi-round QAOA and advanced mixers on a trapped-ion quantum computer. *Quantum Sci. Technol.* **8**, 015007 (2023).
24. E. Farhi, J. Goldstone, S. Gutmann, H. Neven. Quantum Algorithms for Fixed Qubit Architectures . *arXiv:1703.06199* [quant-ph] (2017)
25. Z. Wang, S. Hadfield, Z. Jiang, E. G. Rieffel, Quantum approximate optimization algorithm for Maxcut: A fermionic view. *Phys. Rev. A* **97**, 022304 (2018).
26. K. Marwaha, Local classical MAX-CUT algorithm outperforms $p = 2$ QAOA on high-girth regular graphs. *Quantum* **5**, 437 (2021).
27. J. Wurtz P. Love, Maxcut quantum approximate optimization algorithm performance guarantees for $p > 1$. *Phys. Rev. A* **103**, 042612 (2021).
28. 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 . In Le Gall, F. & Morimae, T. (eds.) *17th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2022)* , vol. 232 of *Leibniz International Proceedings in Informatics (LIPIcs)* , 7:1–7:21 (Schloss Dagstuhl – Leibniz-Zentrum für Informatik , Dagstuhl, Germany , 2022). URL https://drops.dagstuhl.de/opus/volltexte/2022/16514 .
29. E. Farhi, J. Goldstone, S. Gutmann, L. Zhou, The quantum approximate optimization algorithm and the Sherrington-Kirkpatrick model at infinite size. *Quantum* **6**, 759 (2022).
30. K. Marwaha S. Hadfield, Bounds on approximating Max k XOR with quantum and classical local algorithms. *Quantum* **6**, 757 (2022).
31. R. Ayanzadeh, N. Alavisamani, P. Das, M. Qureshi. Frozenqubits: Boosting Fidelity of QAOA by Skipping Hotspot Nodes . *arXiv:2210.17037* [quant-ph] (2022).
32. S. Bravyi, A. Kliesch, R. Koenig, E. Tang, Obstacles to variational quantum optimization from symmetry protection . *Phys. Rev. Lett.* **125** , 260505 (2020).
33. F Wagner, J Nüßlein, F Liers. Enhancing Quantum Algorithms for Maximum Cut via Integer Programming . *arXiv:2302.05493* [quant-ph] (2023).

34. R. Ayanzadeh, J. Dorband, M. Halem, T. Finin. Quantum-assisted greedy algorithms . In *IGARSS 2022–2022 IEEE International Geoscience and Remote Sensing Symposium* , 4911–4914 (2022).

35 D. Sherrington S. Kirkpatrick, Solvable model of a spin-glass. *Phys. Rev. Lett.* **35**, 1792–1796 (1975).

36. A Montanari. Optimization of the Sherrington-Kirkpatrick Hamiltonian . *arXiv:1812.10897* [quant-ph] (2018).

37. J. Weidenfeller, L. C. Valor, J. Gacon, C. Tornow, L. Bello, S. Woerner, D. J. Egger, Scaling of the quantum approximate optimization algorithm on superconducting qubit based hardware. *Quantum* **6**, 870 (2022).

38. S Boettcher, Extremal optimization for Sherrington-Kirkpatrick spin glasses. *Eur. Phys. J. B.* **46**, 501–505 (2005).

39. G Parisi, Infinite number of order parameters for spin-glasses. *Phys. Rev. Lett.* **43**, 1754–1756 (1979).

40. MJ Schmidt. *Replica Symmetry Breaking at Low Temperatures* . Ph.D. thesis, Julius Maximilians-Universität Würzburg. (2008).

41. M. Aizenman, J. L. Lebowitz, D. Ruelle, Some rigorous results on the Sherrington-Kirkpatrick spin glass model. *Commun. Math. Phys.* **112**, 3–20 (1987).

42. A. Montanari S. Sen. Semidefinite Programs on Sparse Random Graphs and their Application to Community Detection . *arXiv:1504.05910* [quant-ph] (2015).

43. A. S. W. Afonso S. Bandeira, Dmitriy Kunisky. Computational Hardness of Certifying Bounds on Constrained PCA Problems . *arXiv:1902.07324* [quant-ph] (2019).

44. Z. Cai, R. Babbush, S. C. Benjamin, S. Endo, W. J. Huggins, Y. Li, J. R. McClean, T. E. O'Brien. Quantum Error Mitigation . *arXiv:2210.00921* [quant-ph] (2022).

45. M. Dupont B. Sundar. Quantum Relax-and-Round Algorithm for Combinatorial Optimization . *arXiv:2307.05821* [quant-ph] (2023).

46. N. Mohseni, P. L. McMahon, T. Byrnes, Ising machines as hardware solvers of combinatorial optimization problems. *Nat. Rev. Phys.* **4**, 363–379 (2022).

47. S Marsh, J Wang, A quantum walk-assisted approximate algorithm for bounded NP optimisation problems. *Quantum Inf. Process.* **18**, 61 (2019).

48. A Bärtschi, S Eidenbenz. Grover Mixers for QAOA: Shifting Complexity from Mixer Design to State Preparation . In *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)* , 72–82 (2020).

49. D. Herman, R. Shaydulin, Y. Sun, S. Chakrabarti, S. Hu, P. Minssen, A. Rattew, R. Yalovetzky, M. Pistoia. Portfolio Optimization via Quantum Zeno Dynamics on a Quantum Processor . *arXiv:2209.15024* [quant-ph] (2022).

50. J. Tilly, H. Chen, S. Cao, D. Picozzi, K. Setia, Y. Li, E. Grant, L. Wossnig, I. Rungger, G. H. Booth, J. Tennyson, The variational quantum eigensolver: A review of methods and best practices. *Phys. Rep.* **986**, 1–128 (2022).

51. C. J. Morningstar M.Weinstein, Contractor renormalization group technology and exact Hamiltonian real-space renormalization group transformations. *Phys. Rev. D* **54**, 4131–4151 (1996).

52. D. M. Abrams, N. Didier, B. R. Johnson, M. P. da Silva, C. A. Ryan, Implementation of XY entangling gates with a single calibrated pulse. *Nat. Electron.* **3**, 744–750 (2020).

53. E. Knill, D. Leibfried, R. Reichle, J. Britton, R. B. Blakestad, J. D. Jost, C. Langer, R. Ozeri, S. Seidelin, D. J.Wineland. Randomized benchmarking of quantum gates. *Phys. Rev. A* **77**, 012307 (2008).

54. A. Erhard, J. J. Wallman, L. Postler, M. Meth, R. Stricker, E. A. Martinez, P. Schindler, T. Monz, J. Emerson, R. Blatt, Characterizing large-scale quantum computers via cycle benchmarking. *Nat. Commun.* **10**, 5347 (2019).

55. A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, H. Weinfurter. Elementary gates for quantum computation. *Phys. Rev. A* **52**, 3457–3467 (1995).

56. C. Huang, T. Wang, F. Wu, D. Ding, Q. Ye, L. Kong, F. Zhang, X. Ni, Z. Song, Y. Shi, *et al.* Quantum instruction set design for performance . *arXiv:2105.06074* [quant-ph] (2021).

57. G Vidal, Efficient simulation of one-dimensional quantum many-body systems. *Phys. Rev. Lett.* **93**, 040502 (2004).

58. M. Dupont, N. Didier, M. J. Hodson, J. E. Moore, M. J. Reagor, Calibrating the classical hardness of the quantum approximate optimization algorithm. *PRX Quantum* **3**, 040339 (2022).

59. M. Dupont, N. Didier, M. J. Hodson, J. E. Moore, M. J. Reagor, Entanglement perspective on the quantum approximate optimization algorithm. *Phys. Rev. A* **106**, 022423 (2022).

60. F. Glover M. Laguna. *Tabu Search* , 2093–2229 (Springer US , Boston, MA , 1998).

61. S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by simulated annealing. *Science* **220**, 671–680 (1983).

62. S Kirkpatrick, Optimization by simulated annealing: Quantitative studies. *J. Stat. Phys.* **34**, 975–986 (1984).