Quantum Approximate Optimization Algorithm and Quantum Annealing for Combinatorial Optimization

David E. Bernal[1], Baljit Singh[2], Prateek Singh[2], and Mahmoud Al Ismail[2]

[1] Department of Chemical Engineering, Carnegie Mellon University `bernalde@cmu.edu`
[2] Language Technologies Institute, Carnegie Mellon University `(baljits,prateeks,mahmoudi)@andrew.cmu.edu`

1 Introduction

Quantum Computing has been receiving a great amount of attention lately by both academia and industry. This attention has been attributed to the recent advances in quantum hardware and algorithms that have the potential to provide an advantage over their classical counterparts, especially ones that demonstrated quantum supremacy [1]. Two such algorithms that sparked this interest are: Quantum Annealing (QA)[2] and the Quantum Approximate Optimization Algorithm (QAOA) Farhi et al. [3]. These algorithms had given quantum computers potentially useful applications.

One of the seemingly potential applications of quantum computing is combinatorial optimization. By taking advantage of the particular properties of these devices; such as superposition, entanglement, and interference; algorithms, *quantum algorithms*, can provide efficient alternatives to the current classical algorithms.

Quantum Annealing is a metaheuristic designed for solving discrete combinatorial optimization problems. This algorithm has been particularly famous since it has been analogously implemented in devices called Quantum Annealers, of which DWave [4] is the main developer and its similarities to the classical metaheuristic Simulated annealing (SA).

QAOA is a quantum algorithm that provides an approximated solution to a combinatorial optimization problem. The algorithm has become popular due to its low depth which makes it a feasible algorithm to be implemented on Noisy Intermediate-Scale Quantum (NISQ) computers, a near term quantum computer which is defined to have a small number of qubits.

Farhi et al. [3] showed that QAOA's complexity depends on p (number of unitary operators) than n (nodes in the graph). In [3] Farhi applied the QAOA to compute the MaxCut of 3-regular graph for $p=1$ and reported an approximation ratio of 0.6942, which is better than the approximation ratio, of $\frac{2}{3}$, yielded by the naive algorithm.

This success of QAOA had attracted numerous development to it. Farhi and Harrow [5] took the first steps to show how QAOA can demonstrate a form of "Quantum Supremacy". Wecker et al. [6] solved MAX-2-SAT by using a variant of QAOA.

Hadfield et al. [7] formulated five problems to be solved using QAOA using hard and soft constraints. The latter ideas were used in [8] to address lattice protein folding.

In the original proposal of QAOA, Farhi et al. [9] highlighted the relationship of the algorithm with the Quantum Adiabatic Algorithm. The Quantum Adiabatic Algorithm (QAA) ensures that if the evolution from an initial Hamiltonian towards a final Hamiltonian is made slow enough, and the system starts at a ground state, the system will never leave its ground state, practically finding the least energy configuration determined by the final Hamiltonian.

As a fun fact, the theory around QAA was also developed by Farhi et al. [10]. The difference between QAA and QAOA is that the later sacrifices the guarantees of optimality (but withholds an approximation property) by making the computation more efficient. The efficiency is achieved through the discretization, a.k.a. *Trotterization*, of the adiabatic path by considering a series of rotations.

QA was initially proposed to be run on classical hardware, by considering quantum mechanics phenomena of SA [11].But, later experimental QA devices, quantum annealers, appeared and turned the algorithm in an analogous algorithm. QA is the practical implementation of QAA, where the "slowly enough" evolution is implemented practically though a continuous evolution of the Hamiltonian using an actual physical device.

In this report, we present an introduction to the Quantum Annealing and the Quantum Approximation Optimization Algorithm. These two algorithms are focused on the solution of discrete combinatorial optimization problems, of which we will focus mainly on MaxCut. We present the formulation of the MaxCut problem for both QAOA and QA, though the representation of the problem as an Ising Hamiltonian. The formulations are used to generate gate-based circuits for gate quantum computers which are simulated and ran in actual quantum computing devices. Besides the Ising formulation of the problem is solved via Simulated annealing and Quantum Annealing. Three different graphs are considered for MaxCut with the approaches mentioned above. Finally, we mention our remarks and discuss the applicability of quantum algorithms for optimization.

2 Algorithms

In this section, we will present the formulation for two algorithms: 1) QAOA and 2) Quantum Annealing for solving combinatorial optimization problems. These algorithms are better suited to solve optimization problems with local constraints.

A combinatorial optimization problem is specified by n bits and m clauses. The optimization task is to find bit-string that maximizes or approximately maximizes the given clauses. Thus, given a bit-string $z \in 0{,}1^n$, we can define our objective function as follows:

$$C(z) = \Sigma_{i=1}^{m} C_i(z)$$

The objective function, $C(z)$, counts the number of clauses that are satisfied using z. The goal of combinatorial optimization is to find a z' such that $C(z')$ is maximal.

2.1 QAOA

The general idea of QAOA is to find z' that is yields an approximation ratio α with high probability. The aim in QAOA is to a find a quantum state $|\gamma,\beta\rangle$ where $\gamma,\beta \in \mathbb{R}^p$ which allows the expected value with respect to the problem Hamiltonian, H, to be maximized. We can define $F_n = \langle\gamma,\beta|H|\gamma,\beta\rangle$ to be the expected value w.r.t H. The goal is to find the parameters, γ,β that maximizes F_n, which would in turn maximizes H.

QAOA is based on the adiabatic quantum idea that the system starts with a driver Hamiltonian and transitions to the problem Hamiltonian in discrete time steps. Thus, given a problem Hamiltonian H along with single bit Pauli X rotation, we can define two unitary rotations

$$U(H,\gamma) = e^{-i\gamma H}$$
$$U(B,\beta) = e^{-i\beta B}$$

where $B = \Sigma_{i=1}^{n} X_i$. We can prepare the state $|\gamma,\beta\rangle$ as follows:

$$|\gamma,\beta\rangle = U(B,\beta_p)U(C,\gamma_p)...U(B,\beta_1)U(C,\gamma_1)|s\rangle$$

The two unitary rotations are applied p times to eventually reach our problem Hamiltonian. Note that $|s\rangle = H^{\otimes n}|0\rangle^{\otimes n}$ which is simply a uniform superposition.

Therefore, the goal of QAOA is just to find γ,β such that F_p is maximized. This algorithm can be easily applied to a problem if Hamiltonian for this problem is well defined. The algorithm can be broken down in the following steps: 1) pick $|\gamma*,\beta*\rangle$, 2) prepare the state $|\gamma*,\beta*\rangle$ and 3) compute the expectation F_n. We choose the solution that yields the largest F_n for a given $|\gamma*,\beta*\rangle$. There are numerous ways to find the best parameters, γ and β. One such way is by grind search or by using specialized solvers.

2.2 Quantum Annealing

In Quantum Annealing, we specify a Quadratic Unconstrained Binary Optimization (QUBO) model that is implemented as the final Hamiltonian in a plausible adiabatic evolution. A QUBO can be expressed as

$$\min_x \sum_{i,j}^{n} Q_{ij} x_i x_j + \sum_{i}^{n} Q_{ii} x_i, x \in \{0,1\}^n \tag{1}$$

where we minimize an quadratic expression with respect to an n-dimensional vector of binary variables $\mathbf{x}$, the matrix Q determines the quadratic and linear part of the QUBO. Notice that in this case we separate the elements of the diagonal of Q given the fact that $x_i^2 = x_i, x_i \in \{0,1\}$. We can define a graph $G = (V,E)$ such that we have a vertex for every binary variable and include an edge for each non-zero term in Q (technically speaking the Boolean matrix of Q is the adjacency matrix of G). Besides the many applications of this formalism, solving a general QUBO is NP-hard.

This QUBO can be easily transformed into a well-studied problem in Physics, the Ising model. In the Ising model, we represent the energy of para-magnetic materials by assuming that each particle of the system i has a corresponding spin (measured along the z axis) $\sigma_i^z \in \{+1,-1\}$. These particles interact with their immediate neighbors in a graph $G=(V,E)$ and are subject to an external magnetic field that interacts with each particle by accounting for an energy $\pm h_i$. The interaction with other particles can lead to an increase of the total energy if their spins are oriented in the same direction ($J_{ij}>0$, ferromagnetic), or in the opposite direction ($J_{ij}<0$, anti-ferromagnetic). The function that accounts for the total energy of the system is known as the Hamiltonian H_f.

$$H_f=\sum_{(ij)\in E} J_{ij}\sigma_i^z\sigma_j^z+\sum_{i\in V} h_i\sigma_i^z \tag{2}$$

Using the transformation $\sigma_i^z=1-2x_i$ and the corresponding transformations for the matrices Q and J and vector h we obtain the Hamiltonian from the original QUBO.

The DWave devices start from an initial Hamiltonian H_i and drive a time evolution to the final Hamiltonian H_f slowly enough such that at every time step, represented by the dimensionless time $s=t/T$, where T is the total annealing time, the system remains at its ground state, minimizing its energy. The energy of the system is described then as

$$H(s)=A(s)H_i+B(s)H_f \tag{3}$$

where $A(s)$ and $B(s)$ are time functions such that $A(0)=B(1)=1$ and $A(1)=B(0)=0$ (in practice we just need $A(0)/B(0)>>1$ and $B(1)/A(1)>>1$).

Usually the initial Hamiltonian is defined such that its ground state is a total superposition, and the usual choice is having

$$H_i=\sum_{i=1}^{n}\sigma_i^x \tag{4}$$

This procedure allows that at the end of the annealing procedure $s=1$, the ground state of H_f is obtained. If that is the case and H_f represents our $(QUBO)$ we would solve the optimization problem. Quantum Annealing offers an analog implementation of this algorithm and uses a niobium superconducting quantum interference device (SQUID) to implement the spins as qubits. Practically speaking, the Quantum Annealing can be considered as a metaheuristic for solving combinatorial optimization problems by translating them into constrained Ising problems. Whether this method actually takes advantage of quantum mechanics is still debated[12].

3 MaxCut

The problem of MaxCut is that of separating the edges of a graph into two sets such that the number of edges between the two complementary sets are maximum.

Suppose we have a connected graph $G=(V,E)$ which can be denoted by the vertices V and corresponding edges E. We can assign each vertex to a set using a function $f(v_i)$ where $v_i \in V$ which assigns each vertex a set/class $+1$ or -1.

We can formulate the objective function for MaxCut as shown in 5. Given an edge between two vertices j and k the cost function assigns a cost of 1 if j and k lie in different sets otherwise it assigns a cost of zero. Our goal is to find a configuration of the nodes that will maximize the objective function.

$$C=\sum_{\langle jk\rangle} C_{\langle jk\rangle} \tag{5}$$

where $C_{\langle jk\rangle}$ is the cost assigned to the edge between the vertices j and k and can be calculated as,

$$C_{\langle jk\rangle}=\frac{1}{2}(-f(j)f(k)+1) \tag{6}$$

where

$$C_{\langle jk\rangle}=\begin{cases}1, & \text{if edge } \langle jk\rangle \text{ is a cut, that is } f(j)f(k)=-1\\ 0, & \text{if edge } \langle jk\rangle \text{ is not a cut, that is } f(j)f(k)=1\end{cases} \tag{7}$$

3.1 QAOA Formulation

From QAOA perspective, we can assign the n vertices of our graph to n qubits. Since each qubit is in a superposition of $|0\rangle$ and $|1\rangle$, all the 2^n different states of the n qubits correspond to the 2^n classification of the n vertices in a graph.

Let $|z\rangle$ be our basis state which is a tensor product of $|0\rangle$ and $|1\rangle$. To apply the objective function function from 6 we can treat our objective function C as a diagonal operator as shown below.

$$C|z\rangle=\sum_{\langle jk\rangle} C_{\langle jk\rangle}(z)|z\rangle \tag{8}$$

where

$$C_{\langle jk\rangle}|z\rangle=\frac{1}{2}(-f_j\otimes f_k+I)|z\rangle \tag{9}$$

such that

$$C_{\langle jk\rangle}|z\rangle=\begin{cases}|z\rangle, & \text{if edge } \langle jk\rangle \text{ is a cut, that is } f_j\otimes f_k|z\rangle = \text{-I}\\ 0, & \text{if edge } \langle jk\rangle \text{ is not a cut, that is } f_j\otimes f_k|z\rangle = \text{I}\end{cases} \tag{10}$$

Since $(f_j \otimes f_k)|z\rangle = (f_j|z_j\rangle) \otimes (f_k|z_k\rangle)$ and we want the product to be $-I$ if both the qubits are $|0\rangle$ or $|1\rangle$ and I if they are different. In order to exhibit this behaviour in a quantum computer, we can set our function $f(i)=\sigma_i^z$ (the Pauli-z matrix). This is due to the property of Pauli-Z matrices such that $\sigma^z|0\rangle=1$ and $\sigma^z|1\rangle=-1$. Therefore our objective function from 9 now becomes

$$C_{\langle jk\rangle}|z\rangle=\frac{1}{2}\left(-\sigma_j^z\otimes\sigma_k^z+I\right) \tag{11}$$

Now, in order to optimize the above equation 11, we need to define two unitary rotations as below:

$$\begin{aligned} U(C,\gamma) &= e^{-i\gamma C} = e^{-i\gamma \sum_{\langle jk \rangle} C_{\langle jk \rangle}} = \prod_{jk} e^{-i\gamma C_{\langle jk \rangle}} = \prod_{\langle jk \rangle} U\left(C_{\langle jk \rangle},\gamma\right) \\ &= e^{-i\gamma \frac{1}{2}\left(-\sigma_j^z \otimes \sigma_k^z + I\right)} = e^{-i\frac{-\gamma}{2}\left(-\sigma_j^z \otimes \sigma_k^z\right)} e^{-i\gamma \frac{1}{2} I} \end{aligned} \tag{12}$$

The phase change $U(C,\gamma)$ alone does not change the probability of obtaining a certain basis. So we have to introduce a rotation operator $U(B,\beta)$ where B is defined as:

$$B = \sum_{j=1}^{n} \sigma_j^x = \sigma_1^x \otimes I^{\otimes(n-1)} + I_1 \otimes \sigma_2^x \otimes I^{\otimes(n-2)} + \dots + I^{\otimes(n-1)} \otimes \sigma_2^x \tag{13}$$

and

$$U(B,\beta) = e^{-i\beta B} = e^{-i\beta \sum_{j=1}^{n} \sigma_j^x} = \prod_{j=1}^{n} e^{-i\beta \sigma_j^x} = \prod_{j=1}^{n} U(B_j,\beta) \tag{14}$$

where $U(B_j,\beta) = e^{-i\beta \sigma_j^x}$.

3.2 Quantum Annealing Formulation

A simple formulation of this problem can be proposed using binary variables $f(i) \in \{+1,-1\}, \forall i \in V$ that represent whether each vertex belong to one side (V^+) or the other (V^-) of the cut. In that case, to maximize the cut's value we maximize the number of edges that connect the two set of vertices $\delta(V^+)$.

$$C = \sum_{(j,k) \in \delta(V^+)} (-1) w_{jk} f(j) f(k) \tag{15}$$

where we include the term $f(j)f(k)$ that is -1 if the cut crosses the edge between V^+ and V^- times a sign correction. Since there is no way that we can know the set $\delta(V^+)$ a-priori, we add the weights times the spins for all edges in the graph.

$$\begin{aligned} \sum_{(j,k) \in E} w_{jk} f(j) f(k) &= \sum_{(j,k) \in E(V^+)} w_{jk} + \sum_{(j,k) \in E(V^+)} w_{jk} - \sum_{(j,k) \in \delta(V^+)} w_{jk} f(j) f(k) \\ &= \sum_{(j,k) \in E} w_{jk} - 2 \sum_{(j,k) \in \delta(V^+)} w_{jk} f(j) f(k) \end{aligned} \tag{16}$$

There is and offset given by the weights of the edges belonging to the same set. Writing the full optimization problem we have:

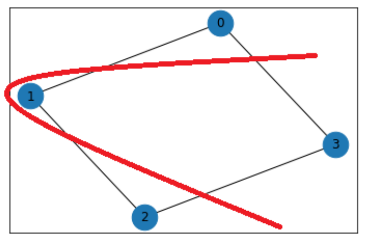
$$\begin{aligned} C_{max} &= \max_{\mathbf{f} \in \{+1,-1\}^n} \frac{1}{2} \sum_{(j,k) \in E} w_{jk} (-f(j) f(k) + 1) = \\ &= \frac{1}{2} \left(\sum_{(j,k) \in E} w_{jk} + \min_{\mathbf{f} \in \{+1,-1\}^n} \sum_{(j,k) \in E} w_{jk} f(j) f(k) \right) \end{aligned} \tag{17}$$

4 Results

The results presented in this paper correspond to our experiments. The code that generates all these results is available in our project repository `https://github.com/bernalde/11860_project_qaoa`. The `Qiskit` simulations and the simulated annealing were run on a Windows laptop with an Intel Core i7 CPU and 16 Gb of RAM. The Quantum Annealing experiments were run on the `DWave-2000Q` annealer, while the quantum circuits were run on the `IBM-Essex` quantum computer. The `Cirq` simulations were run using the `Google Collab` platform. Finally, some extra optimization subproblems were solved using the General Algebraic Modeling System `GAMS`[3] using the mixed-integer linear solver `CPLEX`[4] and the global mixed-integer nonlinear solver `COUENNE`[5]. Every simulation of run in quantum devices performs 2048 anneals or reads unless otherwise stated.

4.1 Small degree-2 graph - Square

Our first approach to implement QAOA was using the simulator `Cirq`[6], developed by Google. This simulator is entirely hosted in Github and provided a very concise way of implementing the circuit for QAOA. The first instance that we implemented was MaxCut of an unweighted regular degree-2 graph with 4 vertices, a square. This graph was selected because we can easily describe its MaxCut (as with any cycle, it is the cut that makes the values of the binary variables be interspersed). This instance is small enough that we can easily enumerate all the solutions. The graph, its MaxCut, and the cut values enumeration can be seen in Fig. 1.

(a) 

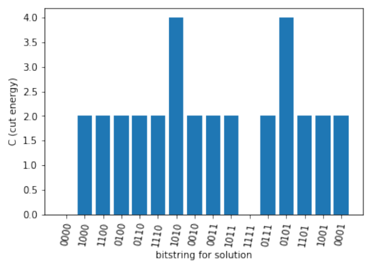
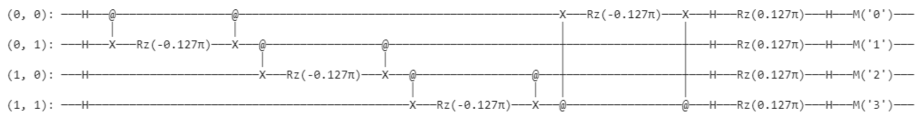
(b) 

Fig. 1: (a) Square graph with numbered vertices and MaxCut in red, (b) Cut energies for every combination of the binary variables. $C_{max} = 4$ is given by bitstrings 0101 and 1010

In order to implement QAOA, we consider the case with depth $p = 1$. As a first consideration, we consider that the hardware that is used has the same

[3] https://www.gams.com/
[4] https://www.ibm.com/analytics/cplex-optimizer
[5] https://projects.coin-or.org/Couenne
[6] `https://github.com/quantumlib/Cirq`

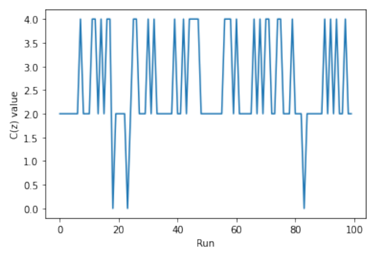
layout as the graph to be analyzed. We associate a qubit for each binary variable determining whether a vertex is on one side or the other of the cut. The algorithm starts with a simple mixer Hamiltonian in complete superposition, achieved though applying Hadamard gates to all the qubits. After having all the qubits in superposition, we apply the unitary matrix that considers the objective and the angle γ, $U(C,\gamma) = \prod_{i,j} e^{-i\gamma C} = \prod_{i,j} e^{-i\gamma w_{ij}/2}$. The implementation of this first rotation is done through the addition of three gates, a rotation around Z of $2\gamma/2$ surrounded by two $CNOT$ gates. The $CNOT$ gates are applied such that the control qubit is one of the vertices of each edge, and the target qubit is the other vertex in the edge, which is also subject to the rotation. After doing the γ rotation, then a second rotation now with respect to β is implemented. This rotation is determined by our mixer Hamiltonian, the single-flip operator σ^x as follows $U(B,\beta) = \prod_{i,j} e^{-i\beta B} = \prod_{i,j} e^{-i\beta\sigma^x}$. This rotation is implemented with three gates, a rotation around Z of 2β, and two Hadamard gates around it. Keep in mind that the rotation angles γ,β should be determined in order to maximize the expectation of the objective function. The value of the angles is restricted. For example, since the eigenvalues of C have to be integer, then the value of γ is in $[0,2\pi]$, while the β is constrained in $[0,\pi]$. For this simple example we just chose arbitrary values for the angles, as $\gamma = 0.4, \beta = 0.2$. The resulting circuit is shown in Figure 2.



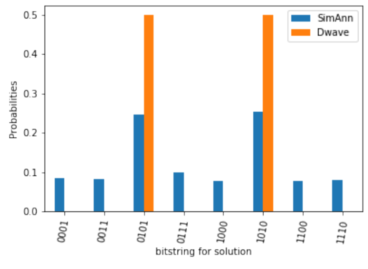
```
(0, 0): ───H───@─────────────────@───────────────────────────────────────────────X───Rz(-0.127π)───X───H───Rz(0.127π)───H───M('0')───
               │                 │                                               │                 │
(0, 1): ───H───X───Rz(-0.127π)───X───@─────────────────@─────────────────────────┼─────────────────┼───H───Rz(0.127π)───H───M('1')───
                                     │                 │                         │                 │
(1, 0): ───H─────────────────────────X───Rz(-0.127π)───X───@─────────────────@───┼─────────────────┼───H───Rz(0.127π)───H───M('2')───
                                                           │                 │   │                 │
(1, 1): ───H───────────────────────────────────────────────X───Rz(-0.127π)───X───@─────────────────@───H───Rz(0.127π)───H───M('3')───
```

Fig. 2: Circuit for QAOA for square graph

We run the given circuit 100 times and report the objective value cost in Figure 3. We can observe that the MaxCut has been found in more than 50% of the runs, even without optimizing the angles.



(a)



(b)

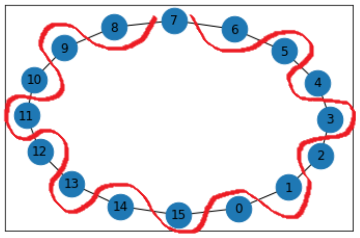
Fig. 3: (a) Cut cost of 100 runs of the circuit shown in Fig. 2, (b) probabilities of the different bitstrings obtained via simulated and quantum annealing

We also implement the given MaxCut instance via the Ising model, using the python package `DWave-ocean-sdk`[7] provided by DWave and solve it using simulated and Quantum Annealing. The simulated annealing algorithm used was the one provided in `DWave-ocean-sdk` and the default parameters have been used for this and the other experiments here. Although the certain parameters in the DWave Quantum Annealing can be tuned to improve the efficiency of this algorithm, such as the ferromagnetic coupling J_{ferro}, the embedding technique, or the annealing time T_{ann} [13], we use the default values for all the cases presented in this document.

The results of this experiment are presented in Fig. 3(b), where we see that both results concentrate in the C_{max} solutions, particularly Quantum Annealing was able to converge in all the runs to the MaxCut. This problem is small enough that the total enumeration takes in a normal laptop 0.002 s, while all the runs of simulated annealing take a total of 0.123 s. Accounting for the whole communication, preprocessing, computation, and postprocessing, the Quantum Annealing procedure takes 6 s, of which the total time spent doing operations in the quantum chip is 0.65 s.

4.2 Larger degree-2 graph - cycle of length 16

In order to try a more challenging problem, we increase the size of the graph to which we want to calculate the MaxCut. The second graph considered here is a cycle of 16 vertices, whose MaxCut has a cost of 16. A motivation for using `Cirq` is that the architecture of Google's superconductor based quantum computers, such as `Sycamore`[1] has a grid architecture. We will simulate this particular example to be run on a 4×4 lattice, a subset of the qubits on `Sycamore` as seen in Fig. 4.

(a)


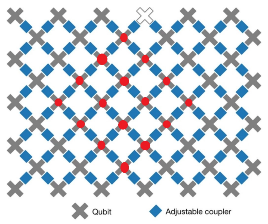
(b)


Fig. 4: (a) Cycle graph with numbered vertices and MaxCut in red, (b) Google's Sycamore chip with red qubits highlighted as hardware to be simulated (image modified from [1]

The circuit that we use for this experiment is similar to the one used in the previous example, just considering the larger graph (more vertices and edges, therefore more qubits and gates, respectively). Here we also consider the circuit depth to be $p=1$. The circuit is presented in Figure 5.

[7] https://github.com/DWavesystems/DWave-ocean-sdk

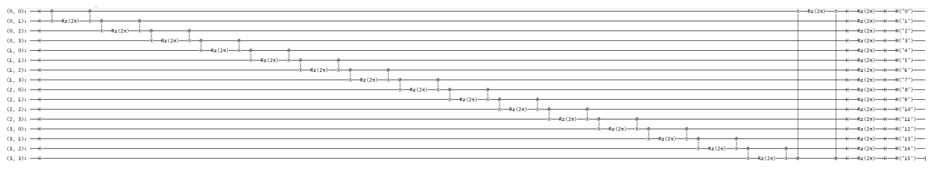


Fig. 5: Circuit for QAOA for cycle graph

Instead of randomly guessing the values of the angles γ,β, we took inspiration in [9], which comments that given the fact that p does not grow with n we can just have a fine grid over the set $[0,2\pi]^p \times [0,\pi]^p$ and evaluate the different values for the angles to obtain good performance. By having 20 grid partitions in each axis, and running 100 simulations for each circuit, we can plot the best-found state for each combination of angles and select the best among them. In words of Farhi et al. [9]

> This works because the function F_p (expected value of the objective function) does not have peaks that are so narrow that they are not seen by the grid.

The obtained surface can be seen in Fig. 6(a) , and the best parameters were $\gamma^* = 0.992, \beta^* = 0.331$.

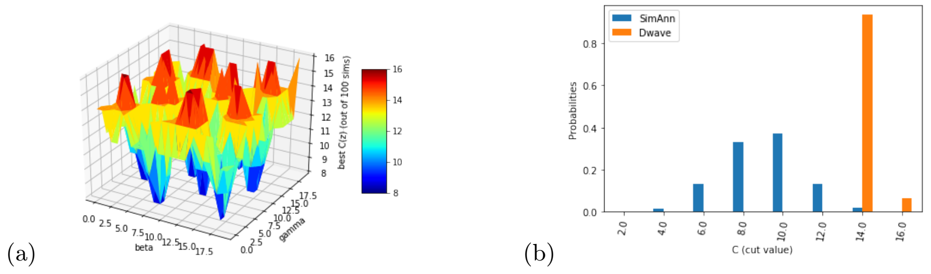
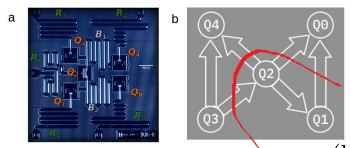
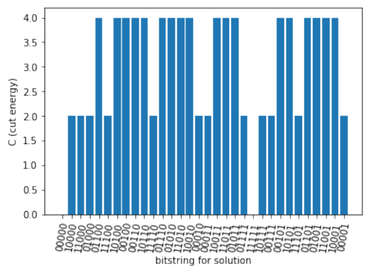


Fig. 6: (a) Best objective cut cost found over 100 simulations of circuit in Fig. 5 for varying angles γ and β, (b) probabilities of the different cut energies obtained via simulated and quantum annealing

This instance was solved using simulated and quantum annealing as the previous example, and the solutions show that given the default parameters, the quantum annealing was able to yield the MaxCut only in 10% of the runs, but it is still 10 times more likely than the simulated annealing to obtain the MaxCut. As a heuristic, DWave also appears to beat the simulated annealing code, since it returns mostly a cut with high value (14), whereas the simulated annealing yields in average cuts of cost 8 (as good as random guessing). These results are presented in Fig. 6(b).



(a)



(b)

Fig. 7: (a) Butterfly graph corresponding to IBM-Essex quantum chip and MaxCut in red (image adapted from [14]), (b) Cut energies for every combination of the binary variables $C_{max} = 4$.

4.3 Butterfly graph

As shown in the previous two examples, a key aspect of QAOA is that it simplifies considerably if the hardware architecture resembles the cut that we want to find the MaxCut to. To further show this we decide to find the MaxCut of the graph in Fig. 7.

The advantage while considering the Butterfly graph is that the analysis on the QAOA circuit is easy enough to have a closed-form derivation of the expected value of the objective function and the hardware natively implements the graph. The derivation of the rotation operators for this particular case was made based on the QAOA `Qiskit` tutorial[8]. By decomposing the cost of the cut in the different kinds of edges in the graph (those with two neighboring sides and those with four) the function for the expectation value of the MaxCut objective for this particular graph is

$$F_1(\gamma,\beta) = 3 - \left(sin^2(2\beta) sin^2(2\gamma) - \frac{1}{2} sin(4\beta) sin(4\gamma) \right) \left(1 + cos^2(4\gamma)\right) \tag{18}$$

We plot the function $F_1(\gamma,\beta)$ in Fig. 8 and use a simple grid search to find the parameters (γ^*,β^*) that maximize the expectation value.

From our grid optimization, by discretizing each axis with a stepsize of 0.01, we obtain that $\gamma^* = 1.230$ and $\beta^* = 1.370$. We also use the global nonlinear optimization solver `COUENNE` to find the global optimal solution, and obtain that the optimal parameters are $\gamma^* = 0.201$ and $\beta^* = 0.340$. Although both points are different, both yield the same expected value $F_1(\gamma^*,\beta^*) = 3.432$, that as we can observe in Fig. 8, happens given the oscillatory nature of F_p. In a practical implementation, the optimization can be done over a reduced space (even smaller than $[0,2\pi]^p \times [0,\pi]^p$) to find the optimal angles.

Although in this case we could derive an analytical description of F_1, for more complicated cases this might become intractable. Therefore, to optimize the parameters given we could rely on a grid search as shown in the previous example, or in "black-box" optimization methods. These methods access the function to be

[8] https://qiskit.org/textbook/ch-applications/qaoa.html

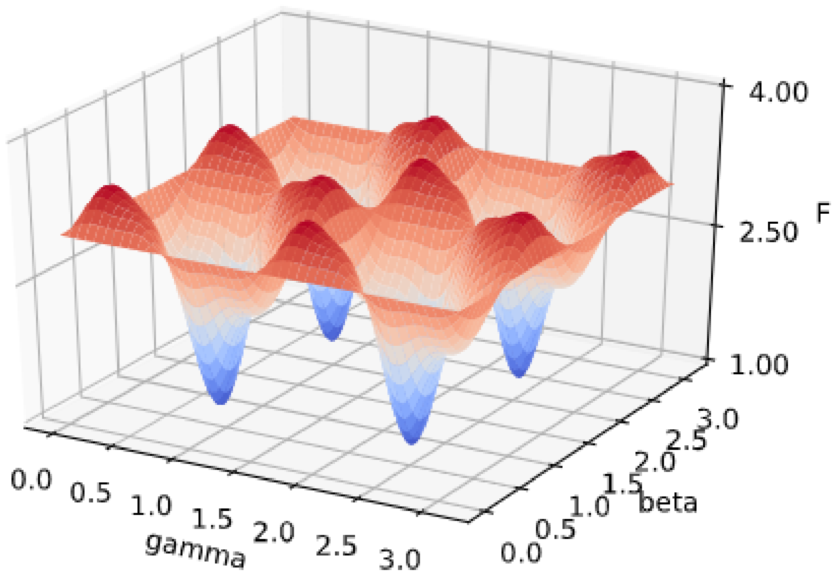


Fig. 8: Expected value of the objective function F with respect to the rotation angles γ,β for the butterfly graph

optimized as an oracle and from there try to optimize the smooth (but unknown function). Different optimization methods that work well for these cases include BOBYQA (Bound Optimization BY Quadratic Approximation), Constrained optimization by linear approximation (COBYLA), the Nelder–Mead method, and Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm [15].

Since we implement this example to be run on the IBM devices, we had to switch our gate-based software from `Cirq` to `Qiskit`[9]. Although the algorithm is the same, the implementation of the circuit differs slightly to the previous example. To implement the circuit we start with a fully superposed state, achieved by using Hadamard gates on all the qubits (5 for this example). Then for each one of the edges (k,l) in the graph (six in this case) we implement the rotation matrix $U_{k,l}(\gamma)$. The first rotation matrix can be natively expressed in `Qiskit` through single qubit rotations along the Z axis $(u_1(\lambda))$ and controlled rotations rotations along the Z axis $(C_{u_1}(\lambda))$ as follows $U_{k,l}(C,\gamma) = C_{u1}(-2\gamma)_{k,l} u1(\gamma)_k u1(\gamma)_l$. The second rotation matrix is applied to all the qubits k, and it is implemented as a $U_k(B,\beta) = R_x(2\beta)_k$, where $R_x(\lambda)$ is a single qubit rotation along the X axis. The outcome of all qubits is finally measured and leads to the answer bitstring. The circuit implementation of this example is presented in Fig. 9.

[9] https://github.com/Qiskit

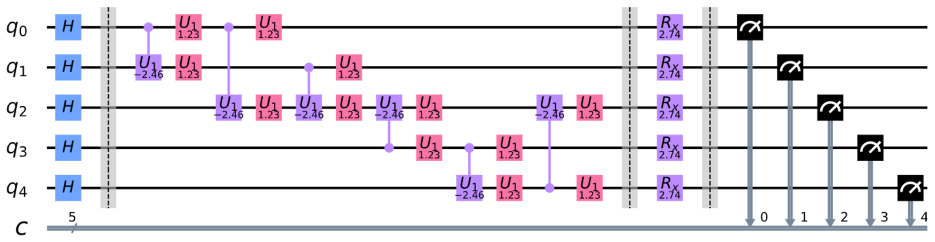


Fig. 9: Circuit for QAOA for butterfly graph

With the QAOA circuit, we can simulate the algorithm and execute it in the `IBM Essex` chip. For each one of these approaches we record the counts of obtaining each bistring and cut value. Besides the runs using the gate-base circuit, we also implement this problem via Ising model and solve it through simulated and quantum annealing. Since we have the same number of runs/anneals in all the simulations and runs on quantum devices, we can compare the results at least from the point of view of probability of finding (either a given bitstring or a cut quantity). The MaxCut for this instance is $C_{max} = 4$ and it can be seen that the Ising model based approaches (simualted and quantum annealing) are finding it for all anneals. The gate-base circuit sometimes finds solutions with the cut number being equal to 2, and finds in very few instances the "worst" solution of cut value being zero.

We have to mention the fact that we are running QAOA with depth $p = 1$, and it is expected that the solution quality improves with the depth of the circuit. Another issue that needs to be mentioned is the fact that, contrary to QAOA, quantum annealing does not provide approximation guarantees. For this particular instance Quantum annealing shows to behave better than QAOA, but this might change with the particular instance.

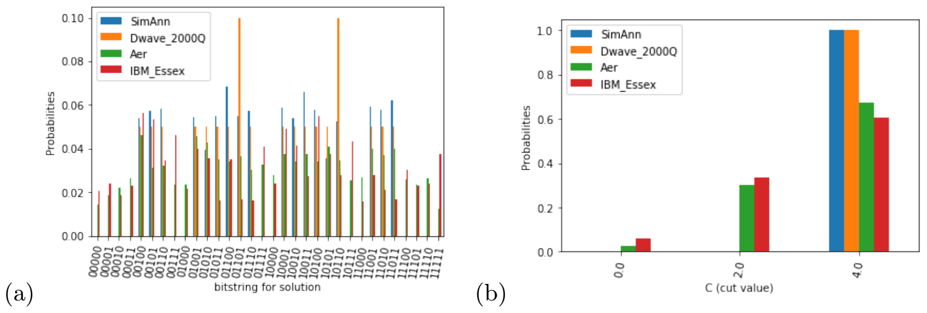


Fig. 10: Probabilities of the different (a) bitstrings and (b) cut energies obtained via QAOA simulation and run og the `IMB Essex` chip and simulated and quantum annealing for the butterfly graph

5 Perspectives on the Quantum Algorithms

Considering the current scale of the Quantum computing chips, the DWave chips have 2 order of magnitudes more qubits available (2048 vs. 53), at the expense that these chips are designed to only run one algorithm (quantum annealing) compared tlo the gate-model computers where using the different gates, one can implement virtually any quantum algorithm. This applies even to the point of being able to implementing a (discretized) version of the Quantum Adiabatic Algorithm: QAOA.

With the above performed experiments, can we conclude that QAOA is better than classical algorithms? From our literature review and the limited experiments we have carried out we cannot guarantee this. The above description of the Max-Cut problem has been the root of competition in recent years. When QAOA was proposed it could for example provide an approximation ratio of 0.6924 to the MaxCut of a 3-regular graph. This approximation ratio is better than a random guess, whose approximation ratio is 0.5, but not as good as the best classical approximation algorithm, the Goemans-Williamson algorithm [16]. On other combinatorial optimization example, MAX E3LIN2, QAOA proved to provide better approximation bounds than any classical algorithm known to the proposal date [17]. This advantage lasted little since soon after a classical algorithm was proposed with a better approximation ratio [18].

We believe that the QAOA algorithm implementation is largely impacted by the gate infidelities. Gate Fidelity, in simple terms, defines how likely a gate will introduce error when operating on qubits [19]. As mentioned in [20], the highest fidelity is achieved with a single gate and this fidelity reduces as the number of gates are increased. We think that as the number of gate layer increases in order to approximate problems with large inputs or to run the algorithm for more steps p, the accuracy will be even worse since the effect of increasing the p value would be cancelled out the **higher** error value.

One issue that we didn't cover in our experimental results but that will be vital for the future implementation of the algorithm has to do with the available hardware for quantum computation. In the examples we covered, we were able to solve small instances of MaxCut that fit natively on the quantum chips. Unfortunately, when going to larger and more complicated graphs, the chances that those graphs are natively *embeddable* in the chips is less. An example is the regular-3 graph shown in Fig. 11(a). Although there are very well based theoretical guarantees of QAOA to find the MaxCut of this graph, the practical implementation of this problem is not trivial and becomes harder with increasing size. Fig. 11(b) shows how the limited connectivity in the graph that defines the DWave chip requires a user to use a minor-embedding technique[21], which is non-trivial and increases the qubit requirement of the problem. For QAOA, the problem is not any simpler. The example in Fig. 12 shows how, since the quantum chips are not *fully connected*, meaning that not every pair of qubits is connected, then a gate that involves two-qubits can only be implemented among neighboring qubits. To solve more complicated problems, it is required to perform $SWAP$ gates such that the qubits are moved to positions where they are required to be to implement the gates in the circuits[22]. These issues are not usually considered in the algorithms analysis, but pose a challenge to practically implement this methods for instances of considerable complexity/size.

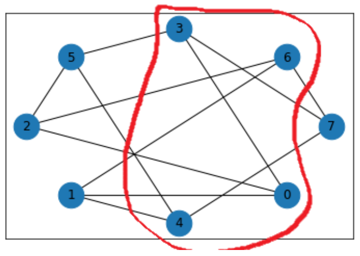
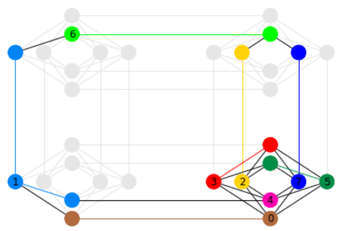
(a) 
(b) 

Fig. 11: (a) Regular-3 graph with 8 vertices with numbered vertices and MaxCut in red, (b) embedding of the graph to solve its MaxCut on DWave Chimera chip

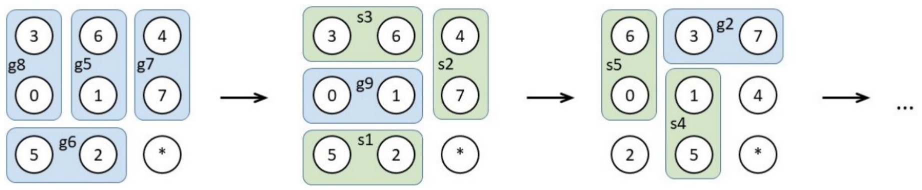


Fig. 12: Schedule of MaxCut QAOA executed for Regular-3 graph with 8 vertices shown in Fig. 11 on a 3×3 lattice chip (taken from [23])

These perspectives should not discourage the research in quantum algorithms for combinatorial optimization. In fact, in our opinion, which will be further discussed in the next and final section of this report, understanding the strengths and current limitations of these algorithms should result in a better realization of both quantum hardware and algorithms.

6 Conclusion

As discrete combinatorial optimization one of the most promising applications of quantum computing, quantum algorithms that aim to solve these problems are the target of a lot of attention. Given the current limitations in terms of hardware, algorithms have been proposed to work with the so-called near-term quantum devices. The algorithms discussed in this paper are Quantum Annealing (QA) and Quantum Approximate Optimization Algorithm (QAOA), designed to run on experimental quantum annealers and noisy intermediate-scale quantum technology, respectively.

Both algorithms use ideas from the Quantum Adiabatic Algorithm (QAA), an algorithm that is guaranteed to obtain the optimal solution represented by the ground-state of a system by carrying a very slow time evolution from an initial (mixer) to a final (driver) Hamiltonian. In QA, a practical version of the QAA is implemented by analogously running a Hamiltonian evolution in a low (but non-zero) temperature devices. QAOA can be seen as a discretized version of QAA, where the discretization steps are interspersed rotations guided by the mixer and driver Hamiltonians.

The close relationship between these two algorithms allowed us to compare them in this paper. One issue we realized is that, although related to QAA, QAOA cannot be fully explained using QAA theory[24, 25]. This fact should motivate the community to produce the theoretical framework on QAOA, as states in [26]. Similarly, further study on QA is required to fully characterize this algorithm[27]. A main weakness of QA is the fact that it cannot provide an approximation ratio, making it a metaheuristic in terms of performance guarantees. It would benefit the algorithm to obtain some sense of theoretical guarantees on its behavior.

Considering a practical point of view of the algorithms, the parameter tuning of the algorithms might allow the users to obtain the best performance out of them. Although we didn't include them in our results, for QA the annealing schedule parameters, e.g. pauses, annealing times, embedding techniques, play a major role in the success rate of the algorithm [13]. Similarly, for QAOA there exist trade-offs when varying the depth of the circuit. Many of these issues have not been or cannot be answered analytically, leading to large-scale experiments to determine their effects [28]. An important and yet unanswered question for QAOA is how to efficiently compute the optimal $2p$ angles $(\beta_1, \gamma_1, \cdots, \beta_p, \gamma_p)$, which we hope to be an active field of research in the near future.

Finally, we expect that these algorithms will not be overall better than the classical alternatives. On the other hand, we do not think that they are of no use, even as heuristic methods there is a potential use of quantum algorithms. There are for sure instances where it might be advantageous to use either classical or quantum algorithms, depending on those instances characteristics. Efforts have been made to answer this issue [23, 29, 30]. Predicting which alternative would be better is a promising direction that we acknowledge regarding classical and quantum algorithms for combinatorial optimization problems.

Bibliography

[1] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando GSL Brandao, David A Buell, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.

[2] M. W. Johnson, M. H. S. Amin, S. Gildert, T. Lanting, F. Hamze, N. Dickson, R. Harris, A. J. Berkley, J. Johansson, P. Bunyk, E. M. Chapple, C. Enderud, J. P. Hilton, K. Karimi, E. Ladizinsky, N. Ladizinsky, T. Oh, I. Perminov, C. Rich, M. C. Thom, E. Tolkacheva, C. J. S. Truncik, S. Uchaikin, J. Wang, B. Wilson, and G. Rose. Quantum annealing with manufactured spins. *Nature*, 473(7346):194–198, 05 2011.

[3] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv:1411.4028*, ArXiv:1411.4028. 2014.

[4] Mark W Johnson, Mohammad HS Amin, Suzanne Gildert, Trevor Lanting, Firas Hamze, Neil Dickson, Richard Harris, Andrew J Berkley, Jan Johansson, Paul Bunyk, et al. Quantum annealing with manufactured spins. *Nature*, 473(7346):194–198, 2011.

[5] Edward Farhi and Aram W Harrow. Quantum supremacy through the quantum approximate optimization algorithm. *arXiv preprint arXiv:1602.07674*, 2016.

[6] Dave Wecker, Matthew B Hastings, and Matthias Troyer. Training a quantum optimizer. *Physical Review A*, 94(2):022309, 2016.

[7] Stuart Hadfield, Zhihui Wang, Eleanor G Rieffel, Bryan O'Gorman, Davide Venturelli, and Rupak Biswas. Quantum approximate optimization with hard and soft constraints. In *Proceedings of the Second International Workshop on Post Moores Era Supercomputing*, pages 15–21, 2017.

[8] Mark Fingerhuth, Tomáš Babej, et al. A quantum alternating operator ansatz with hard and soft constraints for lattice protein folding. *arXiv preprint arXiv:1810.13411*, 2018.

[9] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

[10] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, Joshua Lapan, Andrew Lundgren, and Daniel Preda. A quantum adiabatic evolution algorithm applied to random instances of an np-complete problem. *Science*, 292(5516): 472–475, 2001.

[11] Bruno Apolloni, C Carvalho, and Diego De Falco. Quantum stochastic optimization. *Stochastic Processes and their Applications*, 33(2):233–244, 1989.

[12] Salvatore Mandra, Helmut G Katzgraber, and Creighton Thomas. The pitfalls of planar spin-glass benchmarks: raising the bar for quantum annealers (again). *Quantum Science and Technology*, 2(3):038501, 2017.

[13] Davide Venturelli, Salvatore Mandra, Sergey Knysh, Bryan O'Gorman, Rupak Biswas, and Vadim Smelyanskiy. Quantum optimization of fully connected spin glasses. *Physical Review X*, 5(3):031040, 2015.

[14] Bikash K Behera, Swarnadeep Seth, Antariksha Das, and Prasanta K Panigrahi. Demonstration of entanglement purification and swapping protocol to design quantum repeater in ibm quantum computer. *Quantum Information Processing*, 18(4):108, 2019.

[15] Leo Zhou, Sheng-Tao Wang, Soonwon Choi, Hannes Pichler, and Mikhail D Lukin. Quantum approximate optimization algorithm: performance, mechanism, and implementation on near-term devices. *arXiv preprint arXiv:1812.01041*, 2018.

[16] Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

[17] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm applied to a bounded occurrence constraint problem. *arXiv preprint arXiv:1412.6062*, 2014.

[18] Boaz Barak, Ankur Moitra, Ryan O'Donnell, Prasad Raghavendra, Oded Regev, David Steurer, Luca Trevisan, Aravindan Vijayaraghavan, David Witmer, and John Wright. Beating the random assignment on constraint satisfaction problems of bounded degree. *arXiv preprint arXiv:1505.03424*, 2015.

[19] Salonik Resch and Ulya R. Karpuzcu. Quantum computing: An overview across the system stack, 2019.

[20] Easwar Magesan, Robin Blume-Kohout, and Joseph Emerson. Gate fidelity fluctuations and quantum process invariants. *Physical Review A*, 84(1), Jul 2011. ISSN 1094-1622. https://doi.org/10.1103/physreva.84.012309. URL `http://dx.doi.org/10.1103/PhysRevA.84.012309`.

[21] David E Bernal, Kyle EC Booth, Raouf Dridi, Hedayat Alghassi, Sridhar Tayur, and Davide Venturelli. Integer programming techniques for minor-embedding in quantum annealers. *arXiv preprint arXiv:1912.08314*, 2019.

[22] Davide Venturelli, Minh Do, Eleanor Rieffel, and Jeremy Frank. Compiling quantum circuits to realistic hardware architectures using temporal planners. *Quantum Science and Technology*, 3(2):025004, 2018.

[23] Gian Giacomo Guerreschi and Anne Y Matsuura. Qaoa for max-cut requires hundreds of qubits for quantum speed-up. *Scientific reports*, 9, 2019.

[24] Zhang Jiang, Eleanor G Rieffel, and Zhihui Wang. Near-optimal quantum circuit for grover's unstructured search using a transverse field. *Physical Review A*, 95(6):062317, 2017.

[25] Michael Streif and Martin Leib. Comparison of qaoa with quantum and simulated annealing. *arXiv preprint arXiv:1901.01903*, 2019.

[26] Stuart Hadfield, Zhihui Wang, Bryan O'Gorman, Eleanor G Rieffel, Davide Venturelli, and Rupak Biswas. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms*, 12(2):34, 2019.

[27] Tameem Albash and Daniel A Lidar. Adiabatic quantum computation. *Reviews of Modern Physics*, 90(1):015002, 2018.

[28] Pontus Vikstål, Mattias Grönkvist, Marika Svensson, Martin Andersson, Göran Johansson, and Giulia Ferrini. Applying the quantum approximate optimization algorithm to the tail assignment problem. *arXiv preprint arXiv:1912.10499*, 2019.

[29] Iain Dunning, Swati Gupta, and John Silberholz. What works best when? a systematic evaluation of heuristics for max-cut and qubo. *INFORMS Journal on Computing*, 30(3):608–624, 2018.
[30] Charles Moussa, Henri Calandra, and Vedran Dunjko. To quantum or not to quantum: towards algorithm selection in near-term quantum optimization, 2020.
[31] Andrea Montanari. Optimization of the sherrington-kirkpatrick hamiltonian. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1417–1433. IEEE, 2019.
[32] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Leo Zhou. The quantum approximate optimization algorithm and the sherrington-kirkpatrick model at infinite size. *arXiv preprint arXiv:1910.08187*, 2019.
[33] Ryan Hamerly, Takahiro Inagaki, Peter L McMahon, Davide Venturelli, Alireza Marandi, Tatsuhiro Onodera, Edwin Ng, Carsten Langrock, Kensuke Inaba, Toshimori Honjo, et al. Experimental investigation of performance differences between coherent ising machines and a quantum annealer. *Science advances*, 5(5):eaau0823, 2019.
[34] Tomas Boothby, Andrew D King, and Aidan Roy. Fast clique minor generation in chimera qubit connectivity graphs. *Quantum Information Processing*, 15(1):495–508, 2016.
[35] Giacomo Nannicini. Performance of hybrid quantum-classical variational heuristics for combinatorial optimization. *Physical Review E*, 99(1):013304, 2019.
[36] Panagiotis Kl Barkoutsos, Giacomo Nannicini, Anton Robert, Ivano Tavernelli, and Stefan Woerner. Improving variational quantum optimization using cvar. *Quantum*, 4:256, 2020.

7 Extra interesting references

During the development of this project we encountered a lot of references which unfortunately couldn't be incorporated in the final project. We include a list below, in no particular order, which should serve the interested reader to get more information on QAOA, QA, and combinatorial optimization.

– [29] Great effort to gather classical QUBO and MaxCut heuristics under the same roof. Besides that, the authors provide a trained ML model to choose the best heuristic. Worth a read. The provide a Github repository with all their methods. `https://github.com/MQLib/MQLib` which includes a free-access version of the paper.
– [31] Montanari shows a very interesting result by showing a classical algorithm which might be able to solve the Sherrington-Kirkpatrick model (a generalization of the Ising model) to an arbitrary tolerance ϵ with a time complexity of $C(\epsilon)n^2$ where n is the number of spins. This result is revelant since this would be a classical algorithm that can solve an special (but widely applicable) subcase of combinatorial problems.

- [32] Fahri et al. show that QAOA for Sherrington-Kirkpatrick model (a generalization of the Ising model) and proved that complexity is $O(p^{16})$ to make a competition against previous reference.
- [15] QAOA summary. Quite complete and with newer methods among them two algorithms to determine optimal (heuristically) γ_p,β_p. Interesting comparison with Quantum Annealing, give it a read!
- [33] MIT, NASA and Japanese quantum laboratories paper comparing QA and Coherent Ising Machines for MaxCut and Sherrington-Kirkpatrick. Important reference here. Interesting datapoints for J and T_ann for MaxCut and give formulas of Time to solution (TTS) and other good metrics to quantify the performance of metaheuritic methods.
- [34] Clique embedding algorithm for quantum annealing followed in previous reference and implemented here `https://github.com/DWavesystems/chimera-embedding`
- [26] NASA paper on QAOA Ansatz. Good read. Includes Hamiltonians for several applications in QAOA and the whole Mixers discussion.
- [10] Farhi's proposal of Adiabatic Quantum computing.
- [27] 70 pages on Adiabatic Quantum Computing. A very comprehensive review that discusses the main differences between Adiabatic Quantum Computing and Quantum Annealing.
- [35] Nannicini's paper on variational methods (VQE mainly, but QAOA by simplification) for combinatorial optimization problems. Interesting formulations and comparison using classical integer programming solver CPLEX to benchmark Quantum algorithms in currently inexisting quantum hardware.
- [36] IBM paper improving VQE by not using the mean of the measurement as an obsejctive function but the conditional value at risk (cVar).
- [9] Fahri's, Goldstone's original proposal of QAOA.
- [5] Farhi proposes QAOA as a Quantum Supremacy test problem.
- [28] The Authors went ahead and run almost 1000 million QAOA MaxCut simulations for graphs no larger than 10 nodes and found optimal parameters for these graphs. Found that increasing p makes little impact.
- [25] Volkswagen comparison of QAOA with quantum and simulated annealing. They agree with literature saying that QAOA is Trotterized version of QA but find instances that QAOA can solve and QA cannot.
- [4] DWave's main reference for in Nature presenting their devices.
- [24] NASA paper with interesting phrase: "Our algorithm is a QAOA circuit that demonstrates a quantum advantage with a large number of iterations that is not derived from Trotterization of an adiabatic quantum optimization (AQO) algorithm. It also suggests that the analysis required to understand QAOA circuits involves a very different process from estimating the energy gap of a Hamiltonian in AQO."
- [13] NASA paper on Quantum Optimization of fully connected Ising models.
- [26] NASA paper Quantum Approximate Optimization with Hard and Soft Constraints. We cited their work in several parts of this report