Quantum optimization for scheduling

Author: Thomas Lagkalis, Prof. DG Angelakis @Technical University of Crete

February 9, 2024

Abstract

This project explores the potential of quantum computing in addressing complex scheduling problems. As classical optimization algorithms face challenges in handling exponentially growing problem sizes, quantum computing emerges as a promising solution due to its inherent parallelism and superposition properties. The research focuses on the application of quantum optimization algorithms, including the Quantum Approximate Optimization Algorithm (QAOA) and algorithms based on Quantum Annealers (QA), to tackle scheduling problems.

The project aims to bridge the gap between theoretical quantum optimization concepts and practical implementations, considering constraints and dynamic variables encountered in real-world scenarios. Through the exploration of quantum algorithms tailored to specific scheduling contexts, the research seeks investigate optimization speed, scalability, and solution quality. This interdisciplinary investigation not only delves into the theoretical aspects but also aims to discover practical insights into the integration of quantum computing in solving complex scheduling challenges.

1 Introduction

In the ever-evolving landscape of computational science, the quest for efficient solutions to complex optimization problems has become increasingly vital. Scheduling problems, which involve the allocation of resources, time, and tasks to optimize a desired outcome, is involved in various domains of operational research such as manufacturing, logistics, project management, and more. Classical optimization algorithms have played a significant role in addressing these challenges, but as problem sizes grow exponentially, their limitations become more apparent. Specifically, the scheduling problems are NP-hard, and only a few particular special cases are efficiently solvable.

Quantum computing technology (e.g. quantum annealing), a revolutionary paradigm that leverages the principles of quantum mechanics to process information, offers a promising avenue for tackling complex optimization problems. This project delves into the application of quantum optimization techniques to scheduling problems, exploring how (and if) quantum algorithms can transcend classical constraints.

In section 2 the quantum approximation optimization algorithm (QAOA) is described followed by a practical implementation of the algorithm in the job shop scheduling problem (JSSP) in section 3, using quantum gates technology as described in [5]. In section 4 we try to tackle the nurse scheduling problem (NSP) using Ising Hamiltonian for Quantum Annealing (QA) technology [4]. Finally, in sections 5 we implement the algorithm (described in paper [5]) using Ocean software suite and run the algorithm in D-Wave's QPU (Quantum Processing Unit).

2 Quantum Approximation Optimization Algorithm

The Quantum Approximate Optimization Algorithm (QAOA) is a quantum algorithm designed for solving combinatorial optimization problems. Developed by Edward Farhi, Jeffrey Goldstone, and Sam Gutmann in 2014 [2], QAOA is part of the broader family of variational quantum algorithms.

Cobinatorial optimization problems are described by n bits and m clauses. Clauses are constraints over a subset of the bits which are either satisfied or not. The problem can be described by the following generic formulation:

$$\textit{maximize: } C(\mathbf{x}) = \sum_{a=1}^{m} C_a(\mathbf{x}) \quad , x_i \in \{0,1\} \forall i \in \{1,2...n\} \tag{1}$$

where C is the objective function and $C_a(\mathbf{x}) = 1$ iff the clause a is satisfied by bitstring $\mathbf{x}$, or 0 otherwise. The challenge is to find a string $\mathbf{x}$ that maximizes the number of clauses that are satisfied, i.e. that are evaluated to 1. The satisfiability problem asks for all clauses to be satisfied while MaxSat problem asks for maximazing C($\mathbf{x}$), these two problems are computationally challenging (NP-Hard problems). QAOA is an algorithms which addresses the approximation optimization, i.e to find a solution $\mathbf{x}$ for which C($\mathbf{x}$) is close to max{C}.

2.1 Algorithm preparation and formulation

In this subsection we describe the basic formulation which will be used later when describing the algorithm. Firstly, the user has to translate the clause function (1) in quantum terms:

1. Replace bits in (1) $x_i \in \{0, 1\}$ with spin variables $z_i \in \{-1, 1\}$ using the linear transformation $x_i = (z_i + 1)/2$
2. In $C(z) = \sum_m C_a(z_i)$ replace constants 1s with identity operator $I^{\otimes n}$ and z_i with Pauli Z-operator $Z_i = I^1 \otimes ... \otimes I^{i-1} \otimes \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \otimes ... \otimes I^n$ acting on qubit i.

After these translations the quantum clause Hamiltonian $C(z)$ is been formulated. Next user defines the B operator as the sum of Pauli single bit X-operators

$$B = \sum_j^n X_j$$

where $X_j = I^1 \otimes ... \otimes I^{j-1} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes ... \otimes I^n$ acting on qubit j.

Lastly, user has to use C and B operators in order to define the following unitary operators:

- $U(C, \gamma)$ depends on an angle $\gamma \in [0, 2\pi)$

$$U(C, \gamma) = e^{-i\gamma C} = \prod_{a=1}^{m} e^{-i\gamma C_a} \tag{2}$$

- $U(B, \beta)$ depends on an angle $\beta \in [0, \pi)$

$$U(B, \beta) = e^{-i\beta B} = \prod_{j=1}^{n} e^{-i\gamma X_j} \tag{3}$$

2.2 Algorithm description

For a given positive integer $\mathbf{p} \geq 1$, that represents the number of rounds in the algorithm (or layers in the quantum circuit) the goal is to approximate the maximum of the objective function:

$$F_p(\gamma, \beta) = \langle \gamma, \beta | C | \gamma, \beta \rangle \tag{4}$$

where $|\gamma, \beta\rangle = U(C, \gamma_p)U(B, \beta_p)...U(C, \gamma_1)U(B, \beta_1)|s\rangle$. Note that $[\gamma_1, ...\gamma_p] \equiv \boldsymbol{\gamma}$ and $[\beta_1, ...\beta_p] \equiv \boldsymbol{\beta}$

The QAOA is illustrated in the following steps:

1. Construct the initial quantum state, i.e the n-qubit uniform superposition state by applying $H^{\otimes n}$ to $|0...0\rangle$. This will result to:

$$|s\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{n} |0\rangle$$

2. Construct the two unitary operators (2) and (3) as described in the previous section.

3. Repeat for a predefined number of p iterations:

   3a. Apply (2) and (3) using current gammas and betas.

   3b. Calculate the objective function.

   3c. Update parameters based on classical optimization (e.g., gradient descent)

4. Measure the final state and extract information about the optimal solution.

Algorithm 1 Quantum subroutine of the Quantum Approximate Optimization Algorithm [3]

Input

- Number of rounds of optimization p
- Two size p array of angles, γ and β.
- Hamiltonians C_a corresponding to the clauses of the optimization problem.

Output

- An approximation to the solution of problem in Eq.(1)

Procedure

Step 1. Construct the n-qubit uniform superposition state.
for $i = 1$ to p **do**

Step 2a. Apply $U(C, \gamma[i])$

Step 2b. Apply $U(B, \beta[i])$

end for

Step 3. We call the state so constructed $|\beta, \gamma\rangle$. **return** $|\beta, \gamma\rangle$

Note: This is a demonstration of the quantum subroutine of QAOA. After Step 3 a classical optimization algorithm is used (e.g gradient descend) on variational parameters α and β to maximize $F_p(\gamma, \beta) = \langle \gamma, \beta | C | \gamma, \beta \rangle$

The maximization at $p - 1$ can be viewed as a constrained maximization at p so as integer p increasing the approximation of the objective function gets better (or at least the same)

$$\max_{\gamma,\beta} F_p(\gamma, \beta) \geq \max_{\gamma,\beta} F_{p-1}(\gamma, \beta)$$

It is proved [2] that when p tends to infinity the approximation $F_p(\gamma, \beta)$ tends to the global optimum of the optimization problem

$$\lim_{p\to\infty} \max_{\gamma,\beta} F_p(\gamma, \beta) = \max_z C(z)$$

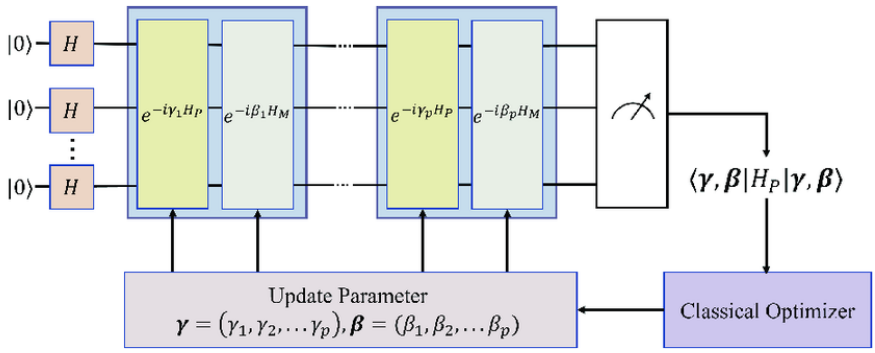


Figure 1: The quantum circuit which implements the QAOA.
Source: www.researchgate.net/figure/QAOA-circuit-diagram_fig3_365507469

2.3 Recap of QAOA

Pick a positive integer p and an initial set of angles $(\boldsymbol{\gamma}, \boldsymbol{\beta})$. Use a quantum computer (e.g Fig. 1) and obtain the state $|\gamma, \beta\rangle$. Measure in the computational basis to get a string z and evaluate C(z) try to pick a new set of variational angle that make F_p as large as possible. Enough repetitions will produce a string z with C(z) (and therefore $F_{p'}$) very near or greater than $F_p(\gamma, \beta)$

3 Implementation of QAOA to job shop scheduling problem

The Job Shop Scheduling Problem (JSSP) is a classic optimization problem in the field of operations research and manufacturing. It belongs to the broader category of scheduling problems and involves the allocation of machines to a set of jobs over a specified time period. The objective is to optimize a certain criterion, often minimizing makespan or total completion time.

The problem is considered NP-hard, meaning that finding an optimal solution becomes computationally challenging as the problem size increases. Therefore, researchers and engineers often use heuristic and metaheuristic algorithms to find near-optimal solutions within a reasonable amount of time. In this section we will apply such an algorithm, QAOA, to tackle the job shop scheduling problem (JSSP) as described in paper "Application of quantum approximate optimization algorithm to job shop scheduling problem, 2023" [5].

3.1 Problem description and formulation

The Job Shop Scheduling Problem is one of the most studied optimization problems over a few decades. In the problem a set of machines (i.e. different) is to perform tasks of jobs, i.e. each job is composed of an ordered list of tasks, each task requires a specific machine for a known processing time. There exist several constraints imposed on jobs and machines:

1. Tasks are non-preemptive. Meaning running tasks cannot be interrupted by other tasks with the intention of resuming it at a later time.
2. Tasks of different jobs are independent.
3. Each task can be performed on one machine at a time.
4. Each machine can process only one job at a time.

The objective is to minimize the makespan, i.e. the maximum completion time of all tasks. In this project the following formulation is used, proposed in parer [5].

1. There are J jobs $J = \{j_1, ..., j_J\}$.
2. Each of the jobs has O_j tasks (operations) to be processed in predefined order $O_j = \{O_{j1} \rightarrow ... \rightarrow O_{jO_j}\}$
3. Each of these single job operations must be processed on a specified and distinct machine from a set of M machines, $M = \{m_1, ..., m_M\}$. Note that $\forall j, O_j \leq M$

The binary variable used in this problem is representing specific timestamps on which the operation can start.

$$x_{k,t} = \begin{cases} 1 & \text{if operation } o_k \text{ starts at time } t \leq T. \\ 0 & \text{otherwise} \end{cases}$$

In general deadline T (which bounds time t) must be estimated by an (e.g. heuristic) algorithm. In the context of this project T is the makespan of a feasible solution. Index k is a positive integer representing the index of the task (operation) in each job, e.g. o_{1k} represents the kth task in the 1st job.

Let's now define the constraints of the problem:

1. The operation must start once and only once. This described by the following formula:

$$C_1(x) = \sum_k (\sum_t x_{k,t} - 1) = 0$$

2. There can be only one operation running at a given machine at any time.

$$C_2(x) = \sum_m (\sum_{k,t,k',t' \in R_m} x_{k,t} x_{k',t'}) = 0$$

where $R_m = A_m \cup B_m$. A_m is the set constraints operation o'_k to start on a machine m if operation o_k is still running on the machine, and B_m is a set that constraints two operations from starting at the same time unless one of their times is 0.

3. The last constraint is defined so that the original order of the operations is kept for all the operations for every job in a given instance:

$$C_3(x) = \sum_{n=1}^{J} (\sum_{\substack{k_{n-1}<k<k_n \\ t+l_k>t'}} x_{k,t} x_{k+1,t'}) = 0$$

4. The JSSP is not only limited to finding a feasible schedule, but also optimizing it's makespan. So we introduce an additional term that will put a penalty favouring any optimal schedule over any non-optimal schedule.

$$C_4(x) = \sum_{n=1}^{J} (J+1)^{t_{k_n}}, \quad t_{k_n} \leq \tau$$

where t_{k_n} is the completion time of last operation of job n and tau the completion time of an optimal schedule.

If the values of all three objectives are equal 0, then all the constraints are satisfied, i.e., there is a feasible schedule and the makespan of this schedule is less or equal to T.

So the problem's objective function is:

$$C(\mathbf{x}) = \sum_{a=1}^{4} C_a(x) \tag{5}$$

Now in order to implement QAOA we have to follow the steps mentioned in Sec. 2.1 and derive the cost Hamiltonian:

$$C(\sigma^z) = \sum_{\alpha=1}^{4} C_\alpha(\sigma^z)$$

Then construct the mixing Hamiltonian:

$$B = \sum_{r=1}^{R} \sigma_r^x$$

where R is the length of the bit string $\mathbf{x}$. Finally, implement the algorithm described in the previous section with a quantum gate based circuit in ordered to derive an approximate optimal solution. In paper [5] are some experimental results mentioned but in this project we will implement a scheduling problem which falls under the umbrella of JSSP in the final section.

4 Quantum annealing and the Nurse Scheduling Problem

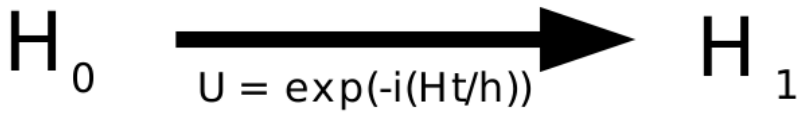
Quantum annealing (QA) is a quantum computing approach that leverages quantum fluctuations to find the global minimum of a given objective function. It's a specialized quantum optimization technique that has been applied to various combinatorial optimization problems. The Nurse Scheduling Problem (NSP) is one such problem where quantum annealing can be explored for potential advantages.

4.1 Adiabatic evolution

Quantum annealing uses adiabatic evolution to find the ground state of the problem (cost) Hamiltonian. The concept of adiabatic evolution is that we start from an initial Hamiltonian H_0, whose ground state can be prepared easily and we evolve the Hamiltonian slowly enough using the Unitary operator $U = exp(-i\frac{Ht}{h})$ to reach the ground state of the problem Hamiltonian H_1. This time depended evolution can be expressed mathematically with the total Hamiltonian:

$$H(t) = A(t)H_0 + B(t)H_1, \qquad t \in [0, T]$$

Where A(t) and B(t) are two monotonic functions which satisfy, $A(0) = 1$, $B(0) = 0$ and $A(T) = 0$, $B(T) = 1$. This process is done by adjusting the magnetic field acting on the qubits.



H_0 is selected to be the easy to prepare Hamiltonian:

$$H_0 = -\sum_{i \in V} \sigma_i^x$$

where σ_i^x is the Pauli-x operator acting on variable i and V is the set of spin variables. And H_1 has the Ising spin model form:

$$H_1 = -\sum_{i,j} J_{ij}\sigma_i^z\sigma_j^z + \sum_i h_i\sigma_i^z \tag{6}$$

Where σ_i^z is the Pauli-z operator acting of variable i and J_{ij} is the strength of interaction between magnets i and j and h_i is the strength of external field experienced by magnet i. After the evolution has happened we end up with the ground state of the final Hamiltonian H_1 which represents the problem objective function. Assuming the initial state is an eigenstate of H0, then the adiabatic theorem promises that the quantum state will remain an instantaneous eigenstate of H(t) provided the dynamics evolve sufficiently slow. The **adiabatic theorem** is crucial to quantum annealing:

> *If we begin in the ground state of system and slowly change the external conditions of our system, we will remain in the ground state of the new system.*

Pros	Cons
Guaranteed global optimum	Slow due to adiabatic evolution
Simple idea	Requires dedicated hardware (e.g. quantum annealers).

Table 1: Pros and Cons of the adiabatic evolution process to solve optimization problems

4.2 Quadratic Unconstrained Binary Optimization

Before we apply adiabatic evolution to solve the problem we have to map the problem to a Quadratic Unconstrained Binary Optimization (**QUBO**) problem. QUBO is a combinatorial optimization problem which contain a binary vector of length $n > 0$, $\mathbf{x} \in \{0, 1\}^n$ and a matrix $Q \in R^{n \times n}$ whose entries Q_{ij} define a weight for each pair of indices $i, j \in \{1, \ldots, n\}$ within the binary vector. In this context the objective function is defined:

$$f(\mathbf{x}) = \mathbf{x}^T Q \mathbf{x} = \sum_{i,j} Q_{i,j} x_i x_j \tag{7}$$

We aim to minimize (or maximize) f. Also, Eq. (7) is unconstrained meaning it doesn't contain any hard or shoft constraints besides the equation (as happened in QAOA).

After we have formulated the problem in QUBO form we have to translate the eq. (7) to the Ising model Hamiltonian. The way to do this is similar to the corresponding steps in QAOA in described in Sec. 2.1. First, we change the binary variables x_i to the bipolar spin variables $s_i = 2x_i - 1$ and then promoting them to pauli-z matrices σ_i^z.

Summarizing, someone can solve an optimization problem using adiabatic evolution (and quantum annealing) by following the steps:

1. Map the problem in QUBO form.
2. Translate the problem's objective function to the ising model form Hamiltonian (Eq. (6)).
3. Prepare the easy Hamiltonian H_0. And use dedicated hardware like quantum annealer to run the adiabatic evolution.
4. Get the final Hamiltonian ground state and post-process it: decode it to get the solution of the problem i.e. the optimal bit string which minimizes (or maximizes) the objective function.

4.3 Nurse Scheduling Problem

The nurse scheduling problem (NSP) is a problem to create a rotating roster of nurses in a hospital respecting a set of constraints. There are two types of constraints (i) hard constraints which must be followed by all valid solutions and (ii) soft constraints which is not obligatory to satisfied by a feasible solution but denotes the quality of this solution.

To begin the formulation of the problem we consider N nurses labeled as n = 1,...,N and a schedule consisting of D days labeled as d = 1,...,D. We also consider a binary variable $x_{n,d} \in \{0, 1\}$, where $x_{n,d} = 1$ specifies the assignment of nurse n to day d. Then, we introduce the constraints of the problem:

- Hard **shift** constraint: at least 1 nurse is assigned each working day. To address this constraint we introduce the required workforce W(d) needed on each day d and the amount of effort E(n) each nurse has available. In Eq.(8) these terms are used in quadratic form and obtain the minimum when the constraint is satisfied.
- Hard **nurse** constraint: no nurse works two or more consecutive days. To construct this constraint we introduce two composite indices i(n, d) and j(n, d) as functions of the nurse n and the day d. We also introduce the matrix J (symmetric) such that $J_{i(n,d),j(n,d+1)} = a$ and 0 otherwise. The positive constant a enforces the constraint by penalizing the objective function. Therefore, the resulting objective function is quadratic i.e. $J_{i,j}q_iq_j$ and takes it's minimum when the constraint is satisfied.
- Soft **nurse** constraint: all nurses should have approximately even work schedules. For this constraint a similar approach to hard shift constraint is being used. We use F(n) specify the number of work days that each nurse wishes to be scheduled and $G(n, d) = h_1(n)h_2(d)$ to denote the preference of nurse n to work at day d, such that:

$$h_1 = \begin{cases} 3 & busy \\ 2 & moderate \\ 1 & idle \end{cases}$$

and the option to work weekend/night or not:

$$h_2 = \begin{cases} 2 & weekenornight \\ 1 & weekday \end{cases}$$

All the above constraints can be summarized in the QUBO formulation Eq. (8)

$$f(\mathbf{x}) = \sum_{i,j} J_{i,j} x_i x_j + \lambda \sum_d (\sum_n E(n) x_i - W(d))^2 + \gamma \sum_n (\sum_d G(n,d) x_i - F(n))^2 \tag{8}$$

Where λ, γ are weight factors which denote the significance of each term and i,j are composite indices i=i(n,d), j=j(n,d). Finally, we translate the Eq. (8) to match the ising model Hamiltonian H_1 (Eq. (6)) and apply quantum annealing as described earlier in the section. More information about experimental results on this problem can be found in paper [4].

5 NSP in D-Wave's hybrid quantum-classical computer

We implemented the Nurse Scheduling Problem using D-Wave's built in libraries which simplify the process of formulating and accessing quantum (hybrid) computers. Full code and detailed description of the code can be found in this repository[6].

5.1 Formulation of the problem

Libraries used in the code make things easier when it comes to formulating the problem. Basically, the only thing the user has to do is formulate the objective function and the constraints in a matrix (in code denoted as **Q**) and then pass it in one of the built in functions. The official documentation[1] has step by step guides on how to formulate the problems and submit them in a quantum computer.

The crucial part of understanding the code is the expand of the hard and shoft **shift** constraints. Note that the indexes **i** and **j** are composite, i.e. i = i(n,d) and j = j(n,d) where n is the nurse and d the day. Also, note the following property of sums:

$$(\sum_k x_k)^2 = \sum_k \sum_t x_k x_t = \sum_k x_k^2 + \sum_k \sum_{t \neq k} x_k x_t \tag{9}$$

So the expansion of the hard shift constraint is as follows (for simplicity we assume that E(n), W(n), F(n) and G(n,d) defined in previous section are all constant):

$$\lambda \sum_d (\sum_n E x_i - W)^2 = \lambda \sum_d (E^2 (\sum_n x_i)^2 + W^2 - 2EW \sum_n x_i)$$

Because the constant term (W^2) in the sum don't effect the optimization (it's just an offset) we ignore it. After applying property of Eq.(9) the expression takes it's final form where we have a term containing the diagonal elements of **Q** and a term with the non diagonal elements:

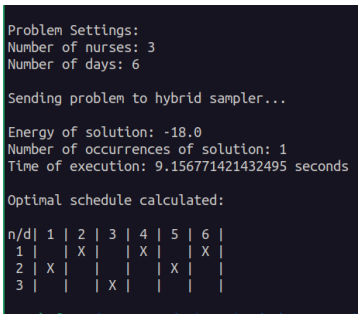
$$(E^2 - 2EW) \sum_d \sum_n x_{i(n,d)} + E^2 \sum_d \sum_{(n,n') \in A} x_{i(n,d)} x_{j(n',d)}$$

where $A = \{(n, n') \in [0, N] \times [0, N] : i(n, d) \neq j(n, d)\}$

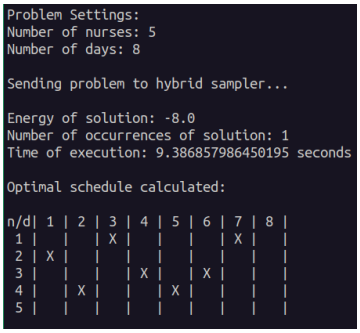
5.2 Experimental results

For this experiment all constants and parameters where set equal to 1 (in the future they could fine tuned for optimization). The results are shown in Fig.2.

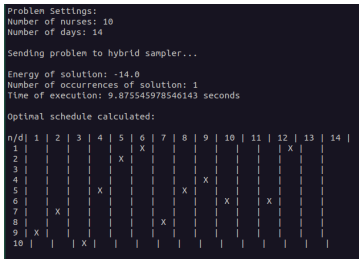
For a small number of input (Fig.2a and Fig. 2b) the solution is feasible (all hard constraints are satisfied) and the execution time is much better than the classical solver (with a classical solver the execution time for 3 nurses and 5 days was $\simeq 3.5sec$). When the input size got bigger (Fig.2c) it failed to deliver a feasible solution (hard shift constrained is not satisfied).



(a) 3 nurses 6 days



(b) 5 nurses 8 days



(c) 10 nurses 14 days

Figure 2: Three experiments on D-Wave's QPU with different input sizes.

6 Conclusion

In this project, we explored the application of quantum optimization algorithms, specifically Quantum Approximate Optimization Algorithm (QAOA) and Quantum Annealing (QA), for solving scheduling problems. For small input sizes, the results obtained from the quantum computer were promising. The algorithm successfully provided feasible solutions, showcasing the potential of quantum optimization in handling scheduling problems efficiently.

However, as the input size increased, the algorithm faced challenges, and the solutions obtained became infeasible. This limitation suggests that, while quantum optimization algorithms show promise for certain problem sizes, there are practical constraints when dealing with larger instances. Factors which depend on the quantum hardware, such as gate error rates, and qubit connectivity can contribute to the diminishing performance on larger problem instances.

References

[1] Dwave developers. Dwave official documatation, 2023. https://docs.ocean.dwavesys.com/en/stable/.

[2] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm, 2014.

[3] Abhijith J., Adetokunbo Adedoyin, John Ambrosiano, Petr Anisimov, William Casper, Gopinath Chennupati, Carleton Coffrin, Hristo Djidjev, David Gunter, Satish Karra, Nathan Lemons, Shizeng Lin, Alexander Malyzhenkov, David Mascarenas, Susan Mniszewski, Balu Nadiga, Daniel O'malley, Diane Oyen, Scott Pakin, Lakshman Prasad, Randy Roberts, Phillip Romero, Nandakishore Santhi, Nikolai Sinitsyn, Pieter J. Swart, James G. Wendelberger, Boram Yoon, Richard Zamora, Wei Zhu, Stephan Eidenbenz, Andreas Bärtschi, Patrick J. Coles, Marc Vuffray, and Andrey Y. Lokhov. Quantum algorithm implementations for beginners. *ACM Transactions on Quantum Computing*, 3(4):1–92, July 2022.

[4] Ikeda Kazuki, Nakamura Yuma, and Humble Travis S. Application of quantum annealing to nurse scheduling problem. *Scientific Reports*, 2019.

[5] Krzysztof Kurowski, Tomasz Pecyna, Mateusz Slysz, Rafał Różycki, Grzegorz Waligóra, and Jan Wglarz. Application of quantum approximate optimization algorithm to job shop scheduling problem. *European Journal of Operational Research*, 310(2):518–528, 2023.

[6] Thomas Lagkalis. Application of optimization algorithm to solve nurse scheduling problem, 2023. https://github.com/ThomasLagkalis/TUC-ARCHIVE/tree/main/Quantum_Qomputin_MATH303.