Lagrangian Duality and Adiabatic Quantum Computation For Constrained Optimization Problems

by

Einar Gabbassov

B.Sc., Simon Fraser University, 2020

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
Department of Mathematics
Faculty of Science

Declaration of Committee

Name:	**Einar Gabbassov**
Degree:	**Master of Science**
Thesis title:	**Lagrangian Duality and Adiabatic Quantum Computation For Constrained Optimization Problems**
Committee:	**Chair:** Paul Tupper Professor, Mathematics

Abstract

The Quantum Approximate Optimization Algorithm (QAOA) is a heuristic method for solving unconstrained binary optimization problems with a gate-based quantum computer. The QAOA consists of a particular quantum circuit architecture, together with a prescription for choosing the parameterization of the circuit. The first part of the thesis studies both the architecture and optimal parameterization of the QAOA circuit. After reviewing the necessary mathematical and physical background, we derive QAOA from scratch and discuss some of its properties. The second part of the thesis focuses on solving constrained combinatorial optimization problems in the setting of fault-tolerant quantum computation and presents a novel Lagrangian duality approach to Discretized Adiabatic Quantum Computation (DAQC). The proposed method allows for building highly resource-efficient and parallelizable quantum circuits. The thesis presents numerical evidence that demonstrates that the proposed approach gives the quadratic improvement in circuit complexity and evolution time over circuits derived from the traditional Quadratic Unconstrained Binary Optimization (QUBO) formalism. We illustrate our findings in the benchmark of the QUBO- and Lagrangian-based DAQC on the NP-complete 1D 0-1 knapsack problem.

Keywords: quantum computation; combinatorial optimization

Dedication

“For Frodo”

Acknowledgements

A warm thank you to my fantastic academic supervisor, Ben Adcock. Special thank you to my industrial supervisors Gili Rosenberg, Artur Scherer, and Pooya Ronagh. I am infinitely grateful to my wife Mira and my mom Ilmira for their support and kindness.

Table of Contents

List of Figures

Chapter 1

Introduction

Digital computers are pillars of our modern civilization. Humanity entered the digital era when John Bardeen, Walter Brattain and Will Shockley developed the transistor in 1947. Since then the computing power of the hardware has grown at a phenomenal pace, so much so that the growth was codified as a law by Gordon Moore in 1965 [42]. Moore's law states that, for a constant economic cost, computing power will double every two years. Since then the law has held approximately true for almost 60 years. However, the current generation of hardware is beginning to face the limits of what is physically possible to manufacture for further power gains. The coming end of Moore's law suggests that traditional computer devices are struggling to support growing computational requirements. Due to this, research in alternative computational methods has steadily increased.

1.1 Quantum computing

One possible alternative to classical computation is the computational paradigm provided by the theory of quantum computation. The main idea is to use quantum mechanics to perform computations. Specifically, the paradigm aims to harness quantum phenomena such as *quantum superposition* and *entanglement* to realize certain computations efficiently, namely, those that are very difficult to do with classical physics-based computers.

The theory of quantum mechanics emerged at the beginning of the 20th century when physicists observed that nanoscopic entities exhibit both particle and wave-like behaviour. Driven by the intuition that simulating quantum systems was very hard for classical devices, Richard Feynman and others suggested that quantum mechanical properties could be exploited to perform calculations faster than classical computers [15]. The first theorized applications of quantum computing were in the field of simulating quantum systems themselves. Throughout the decades many different quantum algorithms were discovered that achieved various types of speedups for certain tasks [41].

As of now, a limited number of practical quantum algorithms are known to provide a theoretical speedup over classical counterparts. Just as the discovery of new practical quantum algorithms, the search for problems that can be solved with quantum computation is an ongoing challenge. So far, some of the most promising application domains are cryptography, quantum chemistry, machine learning and optimization.

1.2 Quantum computation for combinatorial optimization problems

In 1989, Appoloni, Carvalho and de Falco proposed to use the theory of quantum mechanics for solving Combinatorial Optimization (CO) problems [3]. The main idea behind the proposed approach is mapping a CO problem to a quantum system and then allowing the system to evolve under the laws of quantum mechanics. By the end of the evolution, the system is expected to be in its lowest energy state. The resulting state is also a solution to the original problem. This publication inspired the development of a variety of quantum optimization algorithms that work around the idea of mapping a classical problem into some quantum system.

Unfortunately, most of the proposed algorithms require quantum hardware that can perform calculations without introducing errors over time and fit the entire quantum algorithm into its physical memory. The current and *near-term* generation of quantum hardware cannot meet these requirements. Hence, many quantum optimization algorithms remain impractical or produce erroneous results. Therefore, the most recent research is focused on near-term quantum algorithms – algorithms that can work well in the presence of computational errors and do not require a significant amount of quantum resources.

1.3 Contributions

Almost two decades after the proposal by Appoloni et al., Farhi, Goldstone and Gutmann [13] introduced a promising quantum heuristic called the QAOA that can run on the near-term quantum hardware. QAOA aims to solve unconstrained binary optimization problems and it builds on top of the ideas proposed by Appoloni et al., that is, it approximates the *time evolution* from the lowest energy state of a simple physical system to the lowest energy state of the physical system representing a CO problem. The final lowest energy state corresponds to an optimal solution to the CO problem and the approximated time evolution is implemented as a *quantum circuit* on a gate quantum computer.

In the original paper Farhi et al. present the definition of the algorithm, but they do not explain the underlying motivation and reasoning. Moreover, the mathematical derivation of QAOA algorithm is also omitted. Since the introduction of the algorithm, many researchers studied and further developed QAOA [39, 38, 55, 34, 11]. Despite the fact that the original paper only gave the definition of the structure of the algorithm, subsequent studies assumed that the mathematical derivation and motivation of QAOA are well-known.

Many real-world optimization problems usually include constraints. Because QAOA only applies to unconstrained problems, its practical use is limited. Nevertheless, it is possible to reformulate a linear-constrained problem as an unconstrained one to which QAOA could then be applied. The most common approach is reformulating a linear-constrained problem into a QUBO problem [33]. Essentially, QUBO incorporates linear constraints as quadratic penalties into an objective function. This gives rise to many issues. For example, quadratic penalties for inequality constraints require many additional slack variables. The slack variables significantly increase the search space of a problem and make the optimization landscape more rugged. Also, converting constraints into squared penalty terms often results in all-to-all interaction between a problem's variables, making a problem more complex. The additional slack variables and complex interactions between variables also significantly increase the complexity of QAOA and the amount of quantum resources needed.

With these issues in mind, the contributions of this thesis are as follows:

- A complete mathematical derivation of QAOA accompanied with explanations of fundamental concepts of quantum mechanics.
- Quantum circuit architecture for QAOA based on Lagrangian duality theory.
- Quadratic improvement in a circuit complexity and evolution time over a QUBO-based approach for problems with linear inequality constraints.
- Highly parallelizable circuit execution with problem size-independent runtime.
- Analysis of QUBO- and Lagrangian-based formulations and their corresponding quantum circuits for the example of the NP-hard binary Knapsack Problem (KP).
- Numerical study of QUBO-based and Lagrangian-based QAOA with the latter outperforming the former on a large dataset of different size KPs.

1.4 Previous work

A number of meta-heuristics based on *adiabatic quantum computation* [1] were proposed for solving CO problems. One of such techniques is Quantum Annealing (QA) [43]. The heuristic utilizes decreasing quantum fluctuations to search for a low energy state of a Hamiltonian that encodes the CO problem. The QA heuristic uses a time-dependent Hamiltonian consisting of two non-commuting sub-Hamiltonians. The *transverse field* sub-Hamiltonian is gradually decayed, while the *Ising* sub-Hamiltonian is strengthened during the annealing process. The subsequent work [51, 16] focused on determining suitable scheduling strategies of mixing the sub-Hamiltonians to find the ground state with high probability.

Lagrangian duality for constrained binary quadratic problems in the context of quantum adiabatic evolution is presented in [48]. The study presents a method for solving the Lagrangian dual of a binary quadratic programming problem with inequality constraints. The proposed method successfully integrates the Lagrangian duality with branch-and-bound and quantum annealing heuristics. The work in [27] considers a quantum subgradient method for finding an optimal primal-dual pair for the Lagrangian dual of a constrained binary problem. The subgradients are computed using a quantum annealer and then used in a classical descent algorithm.

Variational approaches for determining adiabatic quantum computation scheduling are discussed in [37, 36]. Generally, variational methods involve a classical feedback loop where continuous optimization techniques are used to find a suitable parametrization of a schedule. Although such approaches allow for greater flexibility, they require many runs of a quantum device. Moreover, most variational approaches optimize for the expected energy of a system rather than the probability of sampling an optimal solution [39]. If the energy function is not convex with respect to the variational parameters, then the task of finding optimal parameters is in itself an NP-hard problem. Moreover, even if a suitable schedule is determined for some problem instance, each new problem instance requires finding a new parametrization.

Both QA and QAOA are well suited for optimizing unconstrained binary problems of MaxCut type which belongs to the class of QUBO problems. Constrained problems are canonically reformulated as QUBO problems. In QUBO, the constraints are integrated into an objective function as quadratic penalties. The issue of quadratic penalties present in QUBO is addressed in [24]. It was suggested to use a special initial Hamiltonian and specifically prepared initial state which is simultaneously the ground state of the initial Hamiltonian and satisfies the linear equality constraint. While this approach allows one to omit quadratic penalties for equality constraints, it becomes increasingly hard to prepare an initial state in the presence of multiple equality constraints. Overall, such a state can not be prepared in constant time. The proposed approach can be generalized to inequality constraints by introducing slack variables. However, as we will see in Section 4.2 the number of slack variables has an explicit dependence on the constraint bound and implicit dependence on problem size. This renders all approaches that involve slack variables very costly as it not only requires more quantum resources but also increases the search space of a problem.

1.5 Thesis outline

In this section, we briefly summarize the content of the following chapters. Chapter 2 introduces fundamental concepts of quantum mechanics. Chapter 3 introduces combinatorial optimization and presents the construction of QAOA with different parametrization principles. Next, Chapter 4 discusses integer programs and their conventional reformulation into QUBO problems. The chapter highlights the major issues with the QUBO reformulation and presents a novel method based on the theory of Lagrangian duality. Chapter 5 presents the results of large-scale numerical experiments that demonstrate the superiority of the proposed method over traditional QAOA. Finally, Chapter 6 concludes the thesis and discusses future work.

Chapter 2

Mathematical and physical background

To understand the inner working of QAOA, we introduce multiple fundamental concepts used in quantum mechanics. We commence with the postulates of quantum mechanics, and then we discuss evolution operators, systems with multiple qubits, entanglement and parametrized quantum gates. Finally, we introduce more advanced concepts such as the Schrödinger equation, adiabatic theorem [28] and approximation of unitary operators by Trotter [44].

2.1 Qubits and Dirac notation

In a classical system, the space of states of a system is a mathematical set containing all possible states. For example, a state-space of an ordinary coin could be a two-element set $\{\text{head}, \text{tail}\}$ or it could be $\mathbb{R}^2$ if we were interested in the coin's position on a table. However, the space of states of a quantum system is not a set; it is a vector space. We formalize this claim by stating the first postulate of quantum mechanics [29].

Postulate 1 The state of a system is described by a unit vector in a Hilbert space $\mathcal{H}$.

Like Classical Computing (CC), Quantum Computing (QC) uses bits to manipulate data. However, QC bits can exist in more than one state simultaneously. For this reason, a state of n quantum bits is described by a complex unit vector in $\mathbb{C}^{2^n}$. Hence, the entire QC paradigm can be described by linear algebra on the Hilbert space $\mathbb{C}^{2^n}$. While a complex unit vector is a familiar concept, quantum mechanics uses different notations for representing vectors. We start with a state of a single quantum bit also known as a *qubit*. If the qubit is in the state 0 we write

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}. \tag{2.1}$$

If the qubit is in the state 1 we write

$$|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{2.2}$$

The definition above gives a basis of $\mathbb{C}^2$. Hence, we can represent an arbitrary state as a vector $|\psi\rangle$ which is a linear combination of $|0\rangle$ and $|1\rangle$. That is, we write $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ for $\alpha, \beta \in \mathbb{C}$ such that

$$|\alpha|^2 + |\beta|^2 = 1. \tag{2.3}$$

The discussion above is formalized in the following definition.

Definition 2.1.1 (Quantum superposition principle)**.** If a quantum system (a single qubit) can be in the state $|0\rangle$, and can also be in $|1\rangle$, then quantum mechanics allows the system to be in any arbitrary state

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}. \tag{2.4}$$

We say $|\psi\rangle$ is in a *superposition* of $|0\rangle$ and $|1\rangle$ with *probability amplitudes* α and β.

2.2 Measurement

We have seen how a single-qubit system is represented in quantum mechanics as a unit vector in Hilbert space $\mathbb{C}^2$. Living in the classical physics world, we cannot directly access the quantum information encoded in the vector, but we can obtain partial information by measuring a system. In this section, we will present the measurement postulate and discuss how probability amplitudes are related to measurement outcomes. Let us commence with the measurement postulate of quantum mechanics.

Postulate 2 The probability of measuring a system in a given state is given by the modulus squared of the inner product of the output state and the current state of the system [32, Section 3.4].

For example, suppose our output state is $x \in \{0, 1\}$. When we measure a qubit in the state $|\psi\rangle$ we get probabilistic outcomes 0 or 1 depending on the values of α and β. Specifically, the probability of observing the outcomes 0 or 1 is given by

$$\Pr(\text{outcome is } |x\rangle) = |\langle x|\psi\rangle|^2 = \begin{cases} |\alpha|^2, & x = 0 \\ |\beta|^2, & x = 1. \end{cases} \tag{2.5}$$

Therefore the probability amplitudes α and β determine the probabilities of observing outcomes 0 or 1. We now formalize the concept of measurement of a qubit by giving the following definition.

Definition 2.2.1 (Quantum measurement)**.** Suppose we have a quantum state $|\Psi\rangle$ and an orthonormal basis $\{|\phi_1\rangle, \dots, |\phi_m\rangle\}$. We can explicitly write the state using the orthonormal basis, i.e.,

$$|\Psi\rangle = c_1 |\phi_1\rangle + \cdots + c_n |\phi_m\rangle . \tag{2.6}$$

The probability of measuring each state $|\phi_i\rangle$ is given by

$$\Pr(|\phi_i\rangle) = |c_i|^2 = |\langle\phi_i|\Psi\rangle|^2. \tag{2.7}$$

After measurement, the system is in the state of the measured outcome $|\phi_i\rangle$. This effect is called the *wave function collapse.*

For example, suppose we have the basis $\{|0\rangle, |1\rangle\}$ and a qubit's state is given by

$$|\psi\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}. \tag{2.8}$$

If we measure the qubit in the given basis, we will observe one of the outcomes $|0\rangle$ or $|1\rangle$ with probability 1/2. After the measurement, the qubit is in the state of the measured outcome. This means the state $|\psi\rangle$ was "destroyed" as we lost all information about it.

Since measurement outcomes may be randomly distributed, a single measurement is not enough in many practical applications. To obtain a sample of measurements, we need to prepare an ensemble of identical quantum states we are interested in and then perform a measurement on each member of this ensemble. We will refer to this procedure as *sampling.*

2.3 Evolution operators

In order to change the state of a qubit, we use unitary operators that act on $\mathbb{C}^2$. A *unitary operator* is a bounded linear operator $U : \mathcal{H} \to \mathcal{H}$ on a Hilbert space $\mathcal{H}$ that satisfies $U^*U = UU^* = I$ where U^* is an adjoint of U. For example, suppose we want to switch the state of a qubit by negation. This is done with the *Pauli X* operator defined in the following equation.

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \tag{2.9}$$

Note that X is unitary. Indeed, one easily verifies that $X^*X = XX^* = I$ and $||X||_2 = 1$. We can apply the operator X on the state $|0\rangle$ to get the new state $|1\rangle$.

$$X|0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle . \tag{2.10}$$

Similarly, we can use the operator X to negate the state $|1\rangle$ i.e., $X|1\rangle = |0\rangle$. We can immediately see that the operator X is analogous to the logic NOT gate used in CC. This brings us to the third postulate of quantum mechanics given below.

Postulate 3 The time-evolution of the state of a closed quantum system is described by a unitary operator. That is, for any evolution of the closed system, there exists a unitary operator U such that if the initial state of the system is $|\psi_1\rangle$, then after the evolution, the state of the system will be $|\psi_2\rangle = U|\psi_1\rangle$ [29, Section 3.2].

By a *closed quantum system* we mean an isolated system that does not interact with any other physical system. There are infinitely many operators that can be used to evolve the state of a qubit. The only requirement is unitarity. The most popular operators are Pauli X, which we have already seen and additional Pauli operators I, Y, and Z, which we define as follows:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \; Y = \begin{pmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{pmatrix}, \; Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{2.11}$$

Another important operator is the Hadamard operator:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{2.12}$$

The Hadamard operator is interesting because it allows one to model *constructive* and *destructive interference*. For example, it creates a uniform superposition of $|1\rangle$ and $|0\rangle$ when applied to either of them:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) =: |+\rangle, \; H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) =: |-\rangle. \tag{2.13}$$

Or it destroys the uniform superposition:

$$H|+\rangle = |0\rangle, \; H|-\rangle = |1\rangle. \tag{2.14}$$

The above behaviour follows from the properties of H. Note that H is symmetric and unitary. Hence $H^*H = HH = I$.

2.4 Multiple qubits

Having only a single qubit is not enough to perform a meaningful computation. In this section, we discuss how to represent multi-qubit systems.

Before we give a general definition, we start with a simpler case. Suppose we have a system of two qubits such that one qubit is in the state $|0\rangle$ and the other qubit is in the state $|1\rangle$. We describe their joint state using the tensor product:

$$|0\rangle \otimes |1\rangle = \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}1\begin{pmatrix}0\\1\end{pmatrix}\\0\begin{pmatrix}0\\1\end{pmatrix}\end{pmatrix} = \begin{pmatrix}0\\1\\0\\0\end{pmatrix}. \tag{2.15}$$

For brevity we will use the following notations,

$$|x_1x_2\rangle := |x_1\rangle \otimes |x_2\rangle \tag{2.16}$$

or alternatively

$$|x_1x_2\rangle := |x_1\rangle\,|x_2\rangle \tag{2.17}$$

where $x_1, x_2 \in \{0,1\}$. This example implies that any $|x_1x_2\rangle$ has an equivalent representation in the form of a four-dimensional standard basis vector. The tensor product of two arbitrary states $|\psi\rangle$ and $|\phi\rangle$ each representing a single qubit is given by the following equation:

$$|\psi\rangle \otimes |\phi\rangle \equiv |\psi\rangle\,|\phi\rangle = \begin{pmatrix}\psi_1\\\psi_2\end{pmatrix} \otimes \begin{pmatrix}\phi_1\\\phi_2\end{pmatrix} = \begin{pmatrix}\psi_1\begin{pmatrix}\phi_1\\\phi_2\end{pmatrix}\\\psi_2\begin{pmatrix}\phi_1\\\phi_2\end{pmatrix}\end{pmatrix}. \tag{2.18}$$

The tensor product is a way of combining spaces, vectors or operators together. Suppose that $\mathcal{H}_1$ and $\mathcal{H}_2$ are Hilbert spaces of dimension n and m respectively. Then the tensor product space $\mathcal{H}_1 \otimes \mathcal{H}_2$ is a new Hilbert space of dimension $n \times m$. For example, given a system with n qubits the tensor product space is $\mathbb{C}^2 \otimes \cdots \otimes \mathbb{C}^2 = \left(\mathbb{C}^2\right)^{\otimes n}$ which is isometrically isomorphic to $\mathbb{C}^{2^n}$. We formalize the above by the following postulate of quantum mechanics.

Postulate 4 The Hilbert space of a composite system is given by the tensor product of the separate Hilbert spaces [32, Section 3.6].

When working with multiple qubits, interesting quantum phenomena may arise. In addition to quantum superposition which we covered in the previous section, a multiple-qubit system may have quantum entanglement.

Definition 2.4.1 (Quantum entanglement)**.** If a two-qubit state $|\Psi\rangle$ cannot be written as $|\psi\rangle \otimes |\phi\rangle$ for any choice of $|\psi\rangle$ and $|\phi\rangle$, then $|\Psi\rangle$ is said to be entangled.

Entanglement is a peculiar property that is exclusive to the quantum world. The above definition implies that entangled qubits have a well-defined joint state $|\Psi\rangle$, but they do not have well-defined individual states $|\psi\rangle$ and $|\phi\rangle$. This means any single qubit cannot be described independently of its pair when entangled. The phenomenon of quantum entanglement is what makes quantum physics intrinsically different to classical physics. For clarity, we provide an example of an entangled quantum state known as the *Bell state*:

$$|\Psi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}. \tag{2.19}$$

The Bell state $|\Psi\rangle$ cannot be written as a product of any two states $|\psi\rangle$ and $|\phi\rangle$. Interestingly, the measurement outcomes of the pair of qubits are maximally correlated. If we measure one of the qubits and get the outcome $x \in \{0, 1\}$, then the measurement outcome of the second qubit is also x.

2.5 Operations on multiple qubits

We have covered how to represent states of a multiple qubit system. In this section, we cover how to perform quantum operations on such systems. Suppose we have an $m_1 \times m_1$ unitary matrix U_1 and an $m_2 \times m_2$ unitary matrix U_2. Then

$$U = U_1 \otimes U_2 \tag{2.20}$$

is an $m_1 m_2 \times m_1 m_2$ unitary matrix which satisfies the following rule:

$$U(|\psi\rangle \otimes |\phi\rangle) = (U_1 \otimes U_2)(|\psi\rangle \otimes |\phi\rangle) = U_1 |\psi\rangle \otimes U_2 |\phi\rangle . \tag{2.21}$$

For example, let $U = X \otimes Z$ with X given in (2.9) and Z given in (2.11). Then

$$\begin{aligned} U |0\rangle |+\rangle &= (X \otimes Z)(|0\rangle \otimes |+\rangle) \\ &= X |0\rangle \otimes Z |+\rangle \\ &= |1\rangle |-\rangle . \end{aligned} \tag{2.22}$$

We now introduce additional useful notations that are often used in quantum algorithms.

Definition 2.5.1 (Indexed unitary matrices)**.** Suppose we have a system of n qubits. Let U be a unitary matrix acting on a vector state $|\psi\rangle \in \mathbb{C}^2$ of the ith qubit. Then the unitary matrix U_i on $\mathbb{C}^{2^n}$ is given by

$$U_i = \bigotimes_{k=1}^{n} A_k \tag{2.23}$$

where A_k acts on the kth qubit and satisfies $A_k = I$ for all $k \neq i$ and $A_i = U$.

The above is a convenient notation for denoting an operation that modifies only the state of the ith qubit while keeping the rest of the qubits' states unchanged. For example, suppose we have $n = 3$ and $U = Z$. Then Z_2 is given by

$$Z_2 \equiv I \otimes Z \otimes I, \tag{2.24}$$

which we readily see acts only on the second qubit, while keeping the others fixed.

We generalize this notation to a pair of indices. The matrix $U_i U_j$ acts on the states of the qubits i and j if

$$U_i U_j = \bigotimes_{k=1}^{n} A_k \tag{2.25}$$

where $A_k = I$ for all $k \notin \{i, j\}$ and $A_k = U$ for $k \in \{i, j\}$. For example, for $U = Z$ and $n = 3$ we have

$$Z_1 Z_2 = Z \otimes Z \otimes I, \tag{2.26}$$

$$Z_2 Z_3 = I \otimes Z \otimes Z, \tag{2.27}$$

$$Z_1 Z_3 = Z \otimes I \otimes Z. \tag{2.28}$$

2.6 Parameterized operations on multiple qubits

We now introduce an important class of parameterized unitary matrices often called *rotation gates*. First, let us define the exponential of a square matrix T. The exponential of T is given by the Taylor series of the function $f(x) = e^x$,

$$e^T = \sum_{k=0}^{\infty} \frac{1}{k!} T^k. \tag{2.29}$$

Then for $\sigma \in \{X, Y, Z\}$ and $t \in \mathbb{R}$ the exponentiation of $\mathrm{i}\sigma t$ is given by the following Taylor series:

$$e^{\mathrm{i}\sigma t} = \sum_{m=0}^{\infty} \frac{(\mathrm{i}t)^m}{m!} \sigma^m. \tag{2.30}$$

Note that $\sigma^m = I$ for m even. Using this property and rearranging odd/even terms in the series it is possible to write

$$e^{i\sigma t} = \cos(t) I + \mathrm{i} \sin(t) \sigma \text{ for } t \in \mathbb{R}. \tag{2.31}$$

It is straightforward to verify that $e^{\mathrm{i}\sigma t}$ is a unitary matrix. Given that $\sigma^* = \sigma$ and $\sigma^2 = I$ we have

$$\begin{aligned} \left(e^{\mathrm{i}\sigma t}\right)^* e^{\mathrm{i}\sigma t} &= (\cos(t)I + \mathrm{i}\sin(t)\sigma)^* (\cos(t)I + \mathrm{i}\sin(t)\sigma) \\ &= \cos^2(t)I + \mathrm{i}\cos(t)\sin(t)\sigma - \mathrm{i}\cos(t)\sin(t)\sigma^* + \sin^2(t)\sigma^2 \\ &= I. \end{aligned}$$

It is worth noting that, in general, for a Hermitian matrix A, the exponential of $\mathrm{i}At$ is unitary.

We now can define the following parameterized unitary matrices.

$$\begin{aligned} RX(t) &\equiv e^{-\mathrm{i}Xt/2} = \cos(t/2)I - \mathrm{i}\sin(t/2)X, \\ RY(t) &\equiv e^{-\mathrm{i}Yt/2} = \cos(t/2)I - \mathrm{i}\sin(t/2)Y, \\ RZ(t) &\equiv e^{-\mathrm{i}Zt/2} = \cos(t/2)I - \mathrm{i}\sin(t/2)Z. \end{aligned} \tag{2.32}$$

The above matrices are called *1-qubit rotation gates*. Just as in the case of a unitary matrix U_i acting on the qubit i, we can also define rotation gates $RX_i(t), RY_i(t), RZ_i(t)$ which only rotate the state vector of the ith qubit. For example, for $n = 3$ we have

$$RZ_1(t) = RZ(t) \otimes I \otimes I, \tag{2.33}$$

$$RZ_2(t) = I \otimes RZ(t) \otimes I, \tag{2.34}$$

$$RZ_3(t) = I \otimes I \otimes RZ(t). \tag{2.35}$$

We can also compute their product,

$$RZ_1(t)RZ_2(t)RZ_3(t) = RZ(t) \otimes RZ(t) \otimes RZ(t). \tag{2.36}$$

Finally, we define the *2-qubit rotation gates* which arise when exponentiating $-\mathrm{i}(\sigma\otimes\sigma)t/2$ for $\sigma \in \{X, Y, Z\}$. Depending on the parameter t, the 2-qubit gates can entangle pairs of qubits. These gates are defined as follows

$$RXX(t) \equiv \exp\{-\mathrm{i}(X \otimes X)t/2\} = \cos(t/2)I - \mathrm{i}\sin(t/2)X \otimes X, \tag{2.37}$$

$$RYY(t) \equiv \exp\{-\mathrm{i}(Y \otimes Y)t/2\} = \cos(t/2)I - \mathrm{i}\sin(t/2)Y \otimes Y, \tag{2.38}$$

$$RZZ(t) \equiv \exp\{-\mathrm{i}(Z \otimes Z)t/2\} = \cos(t/2)I - \mathrm{i}\sin(t/2)Z \otimes Z. \tag{2.39}$$

In the context of 2-qubit gates, the matrix I denotes a $2^2 \times 2^2$ identity matrix. The gates above can be indexed as well. For an n-qubit system and $i, j = 1, 2, ..., n$ we have

$$RXX_{i,j}(t) \equiv \exp\{-\mathrm{i}(X_iX_j)t/2\} = \cos(t/2)I - \mathrm{i}\sin(t/2)X_iX_j, \tag{2.40}$$

$$RYY_{i,j}(t) \equiv \exp\{-\mathrm{i}(Y_iY_j)t/2\} = \cos(t/2)I - \mathrm{i}\sin(t/2)Y_iY_j, \tag{2.41}$$

$$RZZ_{i,j}(t) \equiv \exp\{-\mathrm{i}(Z_iZ_j)t/2\} = \cos(t/2)I - \mathrm{i}\sin(t/2)Z_iZ_j. \tag{2.42}$$

In the above equations, we have n qubits and the indexed 2-qubit gates only act on qubits i and j. In this case, I is a $2^n \times 2^n$ identity matrix.

For clarity, we present an example of applying the 2-qubit gate RXX on the state $|00\rangle$. This example will demonstrate that 2-qubit gates can create entangled states. We have

$$\begin{aligned} RXX(t)\,|00\rangle &= \cos(t/2)\,|00\rangle - \mathrm{i}\sin(t/2)(X \otimes X)\,|00\rangle && (2.43)\\ &= \cos(t/2)\,|00\rangle - \mathrm{i}\sin(t/2)\,|11\rangle\,. && (2.44)\end{aligned}$$

Let $t = \pi/2$, then

$$RXX(\pi/2)\,|00\rangle = \cos(\pi/4)\,|00\rangle - \mathrm{i}\sin(\pi/4)\,|11\rangle = \frac{|00\rangle - \mathrm{i}\,|11\rangle}{\sqrt{2}}. \qquad (2.45)$$

The resulting state is entangled, similar to the Bell state in (2.19), except that the second term has the complex coefficient $-\mathrm{i}$. It is possible to get rid of $-\mathrm{i}$ by multiplying the result with the unitary $e^{\mathrm{i}t/2}RZ_1(t)$ with $t = \pi/2$. For a moment, let us ignore the normalizing factor $\frac{1}{\sqrt{2}}$ and perform the following computation:

$$\begin{aligned} &e^{\mathrm{i}t/2}RZ_1(t)(|00\rangle - \mathrm{i}\,|11\rangle) && (2.46)\\ &= \left(e^{\mathrm{i}t/2}RZ(t) \otimes I\right)(|0\rangle \otimes |0\rangle) - \mathrm{i}\left(e^{\mathrm{i}t/2}RZ(t) \otimes I\right)(|1\rangle \otimes |1\rangle)\\ &= e^{\mathrm{i}t/2}\begin{pmatrix} e^{-\mathrm{i}t/2} & 0\\ 0 & e^{\mathrm{i}t/2}\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} - \mathrm{i}e^{\mathrm{i}t/2}\begin{pmatrix} e^{-\mathrm{i}t/2} & 0\\ 0 & e^{\mathrm{i}t/2}\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix}\\ &= \begin{pmatrix}1 & 0\\ 0 & e^{\mathrm{i}t}\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} - \mathrm{i}\begin{pmatrix}1 & 0\\ 0 & e^{\mathrm{i}t}\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix}\\ &= |00\rangle - \mathrm{i}e^{\mathrm{i}t}\,|11\rangle\,.\end{aligned}$$

For $t = \pi/2$ we get

$$|00\rangle - \mathrm{i}e^{\mathrm{i}\pi/2}\,|11\rangle = |00\rangle + |11\rangle \qquad (2.47)$$

Putting back the normalizing factor $\frac{1}{\sqrt{2}}$ we obtain the Bell state $|\Psi\rangle$ in (2.19).

This example demonstrates the usage of rotation gates. We have shown that using $RXX(t)$ and $e^{\mathrm{i}t/2}RZ_1(t)$ gates it is possible to construct the Bell state $|\Psi\rangle$. We have

$$|\Psi\rangle = e^{\mathrm{i}\pi/4}RZ_1(\pi/2)RXX(\pi/2)\,|00\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}. \qquad (2.48)$$

This example also shows how cumbersome some computations might be. In the next section, we visualize the computation above as a quantum circuit diagram. The circuit diagrams give a simple and clear visualization of quantum algorithms.

2.7 The quantum circuit model

From classical computation, we know that an *algorithm* is a well-defined procedure, with finite description, for implementing an information-processing task. Similarly, a *quantum algorithm* is an algorithm that makes use of quantum-mechanical phenomena, such as superposition and entanglement. Quantum algorithms can be described using the quantum circuit model of computation. Quantum circuits are the generalization of classical circuits used in computing science and engineering. Circuits are networks composed of *wires* that carry information to gates that operate on the incoming information and pass it further along the wires. The circuits are *acyclic*, meaning that there are no feedback loops in them. The wires are horizontal lines, and the *information* (qubit's state) propagates along each wire from left to right in time. Quantum gates are shown as rectangular blocks. A quantum circuit can be described by a circuit diagram similar to that shown in Fig 2.1.

When we consider classical computation, the complexity of an algorithm could be specified in terms of the amount of time (or computational steps) a machine needs to execute the algorithm. For the circuit model of computation, one of the natural measures of a circuit's complexity is the depth of the circuit. We formalize the notion of depth in the following definitions.

Definition 2.7.1 (Time step)**.** A *time step* is a single application of the maximum number of gates on a state vector such that all applied gates can be executed in parallel and simultaneously.

Definition 2.7.2 (Circuit depth)**.** The *depth* of a circuit is the maximum number of consecutive time steps necessary to execute a circuit.

If the circuit has high depth, it is more complex and takes longer to execute. We note that the depth of a circuit does not depend on the number of qubits but rather on the number of consecutive gates on wires.

The circuit's connectivity is another important notion that captures a different complexity aspect. Connectivity is a quantum resource related to the circuit's architecture and quantum hardware architecture. To understand this concept, one must distinguish between logical and physical qubits. A *physical qubit* is a physical component present on a quantum processor. A *logical qubit* is used in a quantum circuit representing a quantum algorithm and it is associated with a unitary vector state in $\mathbb{C}^2$. One logical qubit may be comprised of multiple physical qubits.

Any pair of physical qubits is *coupled* if some device (a kind of a resonator) physically connects them. The coupling allows qubits to "talk" to each other and become entangled if necessary. In the current and near-future quantum hardware, physical qubits are *sparsely* connected. Therefore, to make a non-coupled pair of physical qubits "talk" one needs to "relay" information through coupled neighbouring qubits. This operation is costly in terms

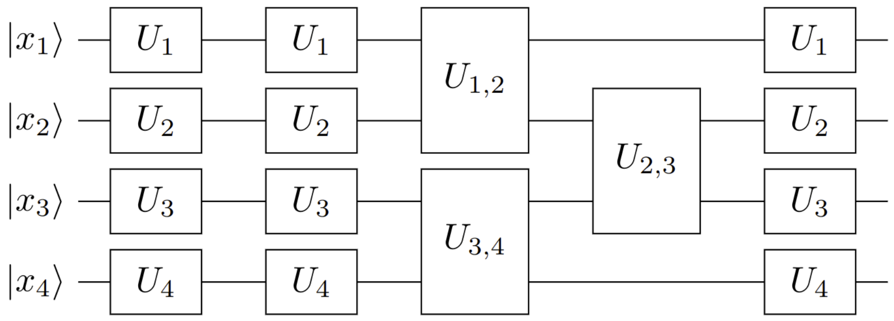


Figure 2.1: A diagram of a quantum circuit on four wires with an input $|x_1x_2x_3x_4\rangle$. The circuit contains 1-qubit gates U_i and 2-qubit gates $U_{i,j}$ for $i, j \in \{1, ..., 4\}$. The gates are applied from left to right in time. Each column of gates is applied in a single time step. In total, there are five sequential steps. Hence the circuit depth is 5. The connectivity is 1, as it suffices to remove the gate $U_{2,3}$ to split the circuit into two independent groups of wires.

of time and errors occurring during the relay. For example, to apply a 2-qubit gate on a non-coupled pair of qubits, it is necessary to insert additional $SWAP$ gates to permute the qubits so that logically they appear coupled. These additional gates increase the depth of a circuit and contribute to the accumulation of errors. To quantify how qubits interact with each other in a circuit, we introduce the following definition.

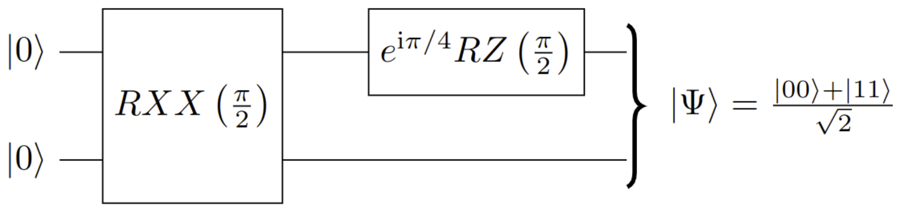
Definition 2.7.3 (Circuit connectivity)**.** The *connectivity* of a circuit is the minimum number of 2-qubit gates that must be removed to separate wires in two or more independent groups of wires.

Given the limitations of quantum hardware, quantum circuits with high connectivity are more costly and complex than sparsely connected circuits. Therefore, algorithm creators should consider connectivity when creating quantum algorithms. Circuits whose every qubit interacts with every other qubit through 2-qubit gates are called *all-to-all* connected. Alternatively, we can define an all-to-all connected circuit as follows.

Definition 2.7.4 (All-to-all connected circuit)**.** An n–qubit circuit is *all-to-all connected* or has *all-to-all connectivity* if its connectivity is $n - 1$.

As an example, we analyze the complexity of a circuit that represents the computation in (2.45)-(2.47). The circuit is illustrated in Fig 2.2. We note that it has two wires and two gates: a 1-qubit gate $e^{\mathrm{i}t/2} RZ_1(t)$ and a 2-qubit gate $RXX(t)$. Since the gates cannot be applied in parallel, we have two time steps. Therefore, the depth of the circuit is two. The circuit has all-to-all connectivity because every wire is connected to all other wires by RXX gate.

By now, it is clear that a quantum circuit is just a logical representation of a quantum state. This means we can perform a measurement on a circuit. In Section 2.2 we defined the



$|0\rangle$
$|0\rangle$
$RXX\left(\frac{\pi}{2}\right)$
$e^{\mathrm{i}\pi/4}RZ\left(\frac{\pi}{2}\right)$
$|\Psi\rangle = \frac{|00\rangle+|11\rangle}{\sqrt{2}}$

Figure 2.2: A circuit diagram for computing the Bell state $|\Psi\rangle$ by performing $e^{\mathrm{i}\pi/4}RZ_1(\pi/2)RXX(\pi/2)\,|00\rangle$ in (2.45)-(2.47).

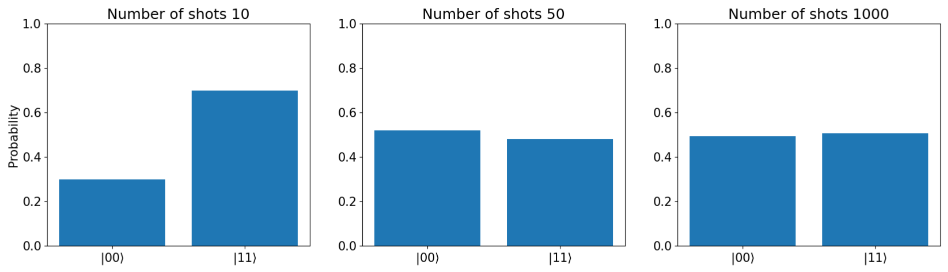


Figure 2.3: Estimates of the probabilities of outcomes of the circuit in Fig 2.2 for a different number of measurements (also called *shots*). As the number of shots increases, the probability estimates get closer to 1/2.

notion of sampling. Similarly, we refer to the procedure of obtaining a sample of measurements from an ensemble of identical circuits as *circuit sampling*. For example, sampling the circuit in Fig 2.2 results in distributions of outcomes illustrated in Fig 2.3.

2.8 Entanglement as a computational resource

Quantum speedup is the term used to describe quantum algorithms' advantage over their classical counterparts when processing a certain task. Multiple quantum algorithms demonstrate a significant speedup over classical algorithms. For example, Shor's polynomial-time algorithm for integer factorization has an exponential speedup over its classical counterparts. One might wonder what quantum effect must be present for the possibility of a speedup over classical computation. An answer to this question is entanglement [26]. Entanglement can be viewed as a particular type of superposition which cannot be expressed as a single product state. While entanglement is necessary for a quantum speedup, it is not sufficient. To see why it is necessary, we recall that if a quantum system's state is not entangled, we can express it as a product of states. This means each qubit's state can be independently represented with a vector in $\mathbb{C}^2$. Therefore, we can operate on each vector state separately and independently by applying unitary transformations computable in polynomial time. Moreover, due to the independence of state vectors, operations on them can be done in parallel.

To show that entanglement is not sufficient for quantum speedup, one must refer to the Gottesman-Knill theorem [45, Section 10.5.4], which is outside of the scope of this study. In essence, the theorem states that certain quantum computations utilizing entanglement can be efficiently simulated on a classical computer.

This discussion highlights how subtle quantum computation is. The mere presence of quantum phenomena such as superposition and entanglement does not guarantee an advantage over classical computation. However, a significant quantum speedup is not possible without entanglement.

2.9 The Schrödinger equation

The states of all quantum systems satisfy certain properties that are encapsulated by a linear differential equation called the *Schrödinger equation*. Solutions to the Schrödinger equation are called *wave functions*. Interestingly, the Schrödinger equation is partly a definition and partly a principle of quantum mechanics. As a definition, it defines a Hamiltonian – a mathematical object representing the energy of a physical system. As a principle, it states that quantum states change continuously with time such that their "unitarity" is preserved [53, Section 4.12]. Conceptually, we can view the Schrödinger equation as a quantum mechanical analog of Newton's second law of motion, which determines a state of a classical object in time. The equation is named after its discoverer Erwin Schrödinger. In 1933, Schrödinger was awarded the Nobel Prize in Physics for his numerous contributions to quantum theory. The equation turned out to be so groundbreaking that it became part of one of the postulates of quantum mechanics.

Postulate 5 The time evolution of the state $|\psi(t)\rangle$ of a closed quantum system with an initial state $|\psi_0\rangle$ at time $t = t_0$ is given by the Schrödinger equation,

$$\mathrm{i}\hbar \frac{d\,|\psi(t)\rangle}{dt} = H(t)\,|\psi(t)\rangle\,, \quad |\psi(t_0)\rangle = |\psi_0\rangle\,. \tag{2.49}$$

In this equation, d/dt is the usual time derivative of a state vector, $H(t)$ is a Hermitian operator on the space $(\mathbb{C}^2)^{\otimes n}$ known as the *Hamiltonian* of the closed system and $\hbar$ is a constant known as *Planck's constant.* If the Hamiltonian is known, then we completely understand the closed system's dynamics. Moreover, eigenvalues of a Hamiltonian represent the energy of a quantum system, and its eigenvectors represent *eigenstates*. For a time-independent Hamiltonian H the relation between H and its energies are given by the *time-independent Schrödinger equation*

$$H\,|\psi_k\rangle = E_k\,|\psi_k\rangle\,, \tag{2.50}$$

where E_k is energy of a system associated with the kth eigenstate $|\psi_k\rangle$. It is easy to see that (2.50) is an eigenvalue equation.

Given the initial state $|\psi(t_0)\rangle$ at the evolution start time t_0, we can express $|\psi(t)\rangle$ in terms of the initial state and the evolution operator $U(t, t_0)$,

$$|\psi(t)\rangle = U(t, t_0) |\psi(t_0)\rangle . \tag{2.51}$$

Therefore, (2.49) is an initial value problem where $|\psi(t_0)\rangle = |\psi_0\rangle$ is the initial state. Since the Schrödinger equation in (2.49) preserves the normalization of a state [22, Section 1.4], solving (2.49) is equivalent to finding a unitary operator $U(t, t_0)$ which evolves the initial state to the solution state $|\psi(t)\rangle$.

We will see later in Section 3.3 that a CO problem can be converted into a Hamiltonian eigenproblem. For reasons discussed therein, we see that the optimal solution of the CO problem corresponds to the eigenstate associated with the highest energy (eigenvalue) of the Hamiltonian. To find the desired eigenstate we solve the Schrödinger equation in (2.49). The solution to the Schrödinger equation $|\psi(t)\rangle$ is defined in terms of a time evolution operator $U(t, t_0)$. The main question is: How do we find the eigenstate associated with the highest energy of the Hamiltonian, i.e. how do we find the optimal solution? For this, we will need the adiabatic theorem presented in the next section.

2.10 The adiabatic theorem

Before we discuss the theorem, it is useful to understand the concept of the adiabatic process (or evolution). Suppose that we have some physical system in a particular state. Since everything in nature tends to its lowest energy state, it is reasonable and convenient to assume that our system is at its lowest energy state (e.g., the system is at rest). An adiabatic process is a process of slowly (gradually) changing external conditions so that the system preserves its state [22, Section 11.5.1]. While the concept might appear somewhat obscure, we use the adiabatic processes regularly in our everyday lives. For example, let the system be a cup of tea. Suppose that we prepared the tea and now we would like to take the cup to a work desk without spilling the hot liquid. The cup's content will remain still (at rest) if we walk slowly enough from a kitchen to a work desk. In this example, we gradually change the external conditions by slowly carrying the cup from the kitchen to the work desk. By the end of the process, we have a cup of tea with the same amount of tea and no spills. Importantly, the tea remained still (at rest) by the end of the transition process.

Before we proceed to the adiabatic process of quantum mechanics, it is helpful to give a proper definition to a state with the lowest energy.

Definition 2.10.1 (Ground state)**.** The *ground state* state of a physical system is the lowest energy state that the system can be in.

Remark: The lowest eigenstate of a Hamiltonian H is also called the ground state. If E is the lowest energy of H with the associated eigenstate $|x\rangle$, then $-E$ is the highest energy of $-H$ with the same associated eigenstate $|x\rangle$. In other words, the ground state of a Hamiltonian H is the highest energy eigenstate of the negated Hamiltonian H. Due to this relation, we will use the term "ground state" to refer to a state that produces either the lowest or the highest energy.

We now discuss the adiabatic process in the context of quantum mechanics. If we take a quantum system whose Hamiltonian slowly changes from H_1 to H_2, then, under certain conditions, the ground state of H_1 is transformed into the ground state of H_2 [2]. Or in other words, the system remains at its lowest energy state if its Hamiltonian changes slowly enough. We present the above claim more formally in the following paragraph.

Let $|\phi(t)\rangle$ be an eigenstate of a time-dependent Hamiltonian $H(t)$ with an eigenvalue $E(t)$. When we say that we apply the adiabatic evolution given by H and $|\phi\rangle$ we mean that we initialize a system in the state $|\psi(0)\rangle = |\phi(0)\rangle$ and then apply the continuously varying Hamiltonian $H(t)$ with t gradually varying from 0 to T. During the evolution process the state $|\psi(0)\rangle$ evolves according to the Schrödinger equation in (2.49) and by the end of the evolution at time $t = T$ we expect the state $|\psi(T)\rangle$ to be the eigenstate $|\phi(T)\rangle$ of the final Hamiltonian $H(T)$. We formalize the above discussion in the adiabatic theorem presented in [2].

Theorem 1 (The adiabatic theorem). *Let $H(t), 0 \leq t \leq T$, be a time dependent-Hamiltonian, let $|\psi(t)\rangle$ be one of its eigenstates, and let $E(t)$ be the corresponding eigenvalue. Assume that for any $t \in [0, T]$, all other eigevalues of $H(t)$ are either smaller than $E(t) - \lambda$ or larger then $E(t) + \lambda$ (i.e. there is a spectral gap of $\lambda > 0$ around $E(t)$). Consider adiabatic evolution given by H and $|\psi\rangle$ applied for time T. Then the following condition is enough to guarantee that the final state is at distance at most $\epsilon > 0$ from $|\psi(T)\rangle$:*

$$T \geq \frac{10^5}{\epsilon^2} \cdot \max \left\{ \frac{||H'||^3}{\lambda^4}, \frac{||H'|| \cdot ||H''||}{\lambda^3} \right\}, \tag{2.52}$$

In the equation above we use $||H||$ to denote $\max_{t\in[0,T]} ||H(t)||$ where $||\cdot||$ is the usual operator norm and H', H'' are the first and second order time derivatives of $H(t)$.

Simply speaking, the theorem states that if we change the Hamiltonian from $H(0)$ to $H(T)$ for long enough time, then the state $|\psi(T)\rangle$ will be the ground state of $H(T)$ given that the initial state $|\psi(0)\rangle$ is the ground state of $H(0)$. We note that the evolution time T depends on the spectral gap λ. The smaller the gap is, the longer the evolution time must be.

2.11 Trotter formula

To implement $U(T,0)$ that evolves the initial state $|\psi(0)\rangle$ to the ground state $|\phi(T)\rangle$ of the Hamiltonian $H(T)$ as a quantum circuit, we will need the following asymptotic approximation theorem:

Theorem 2 (Trotter formula)**.** *Let A and B be Hermitian operators on $(\mathbb{C}^2)^{\otimes n}$. Then for any real t,*

$$\lim_{n\to\infty}\left(e^{\mathrm{i}At/n}e^{\mathrm{i}Bt/n}\right)^n = e^{\mathrm{i}(A+B)t}. \tag{2.53}$$

The theorem holds even if A and B do not commute. By setting n to be finite we obtain different order approximations of $\exp\left(\mathrm{i}(A+B)t\right)$. For example, setting $n=1$ gives the first order Trotterization [45, Section 4.7.2]

$$e^{\mathrm{i}(A+B)t} = e^{\mathrm{i}At}e^{\mathrm{i}Bt} + O(t^2) \text{ as } t\to 0. \tag{2.54}$$

See the derivation in Appendix A. If A and B commute, then the error term is zero. However, for our purposes, we will work with Hermitian A and B that do not commute. The idea behind the formula is to approximate the total evolution as a sequence of simpler evolutions up to some error. If $t \ll 1$, then the error in this approximation becomes negligible.

Chapter 3

Quantum algorithms for solving combinatorial problems

We commence this chapter with an introduction to the class of binary CO problems suitable for QAOA. Next, we construct a QAOA circuit by solving the Schrödinger equation in (3.6) and then approximate the resulting unitary matrix $U(t, 0)$ with the Trotter product formula. Finally, we summarize the chapter with a discussion about the parametrization of a circuit.

3.1 Combinatorial optimisation problems

This section briefly introduces the mathematical formulation of a binary combinatorial problem whose solution can be approximated by QAOA.

Binary CO problems are specified by n *bits* and m *clauses*. Each clause may represent a constraint or some quantity we would like to optimize on a subset of the bits. Certain combinations of bits decrease the value of a clause, whereas other combinations of bits increase the value of a clause. The goal is to find an optimal combination of n bits that maximizes (or minimizes) the sum of all m clauses. Mathematically, this can be formulated as the maximization of an objective function

$$C(x) = \sum_{k=1}^{m} C_k(x), \tag{3.1}$$

where $C_k(x) \in \mathbb{R}$ is a clause and $x \in \{0,1\}^n$. If the string x does not satisfy the clause k then $C_k(x) < 0$ (or $C_k(x) > 0$ if we minimized). A negative value of C_k can be viewed as a penalty which decreases the overall value of $C(x)$.

Because we have n bits, there are 2^n possible different strings, and our goal is to find a string that maximizes $C(x)$. If we were to use a linear search, it would take 2^n different string evaluations to determine the optimal x^* such that $C(x) \leq C(x^*)$ for all $x \in \{0,1\}^n$. While $C(x)$ has a somewhat abstract definition, it can readily represent many combinatorial problems such as MaxCut, packing, covering and partitioning problems [35]. The goal of the QAOA is to find an approximate solution to (3.1) using far fewer than 2^n string evaluations.

3.2 MaxCut problem example

This section presents an example of a CO problem called MaxCut [17]. MaxCut is a well-known NP-complete problem in combinatorial optimization. Let $G(V, E)$ denote an undirected graph G with a vertex set V and an edge set E. Given a connected graph $G(V, E)$, the goal is to partition V into two subsets V_0 and V_1 such that the number of edges between the two subsets is as large as possible. We refer to such edges as a *cut-set*. We assume that the size of the vertex set V is n. Hence, the problem size is also n. For $i = 1, \ldots, n$, let $x_i \in \{0, 1\}$ denote the membership of a vertex $i \in V$. If $x_i = 0$ then the vertex i belongs to the subset V_0, otherwise i belongs to V_1. For an edge $(i, j) \in E$ to be in the cut-set, i and j must have different memberships i.e., $x_i \neq x_j$. Therefore, we can formulate the MaxCut problem as follows

$$\max_{x \in \{0,1\}^n} \sum_{(i,j) \in E} (x_i + x_j - 2x_i x_j). \tag{3.2}$$

Note that if both ends of some edge (i, j) are assigned to the same subset V_1 then $x_i, x_j = 1$ and $(x_i + x_j - 2x_i x_j) = 0$. Hence, such an assignment does not contribute to the objective function. On the other hand, if $x_i = 0$ and $x_i = 1$, then $(x_i + x_j - 2x_i x_j) = 1$ and the objective function value is increased.

Note that (3.2) is a quadratic, unconstrained problem with binary variables. Therefore it is a QUBO problem. Often, the optimization community uses the terms QUBO and MaxCut interchangeably. Also note that (3.2) is a particular case of the problem (3.1) where for each edge $(i, j) \in E$ we assign a positive index k and define $C_k(x) := x_i + x_j - 2x_i x_j - 1$. Then maximizing (3.2) is equivalent to maximizing $C(x) = \sum_k C_k(x)$. Therefore, we can apply QAOA to MaxCut problems.

3.3 Representing a CO problem as a Hamiltonian

We now put the Schrödinger equation in the context of CO. It is possible to represent the objective function in (3.1) as a time-independent Hamiltonian, see the derivation example for MaxCut in Section 3.8. Then solving the time-independent Schrödinger equation in (2.50) would yield one of the eigenstates of the Hamiltonian. Importantly, the resulting eigenstate corresponds to one of the solutions to the optimization problem in (3.1). The

Hamiltonian representing (3.1) can be written as a $2^n \times 2^n$ diagonal matrix

$$H_C = \begin{pmatrix} C(0\cdots 00) & & & \\ & C(0\cdots 01) & & \\ & & \ddots & \\ & & & C(1\cdots 11) \end{pmatrix}. \tag{3.3}$$

Let $E_0 \geq E_2 \geq ... \geq E_{2^n-1}$ denote the eigenvalues of H_C. We immediately see that eigenvalues of H_C in (3.3) correspond to ordered (from largest to smallest) objective function values $C(x)$ for $x \in \{0,1\}^n$. Now consider the state $|x^*\rangle$ where x^* is the optimal solution to the original problem in (3.1). Then

$$H_C |x^*\rangle = C(x^*) |x^*\rangle = E_0 |x^*\rangle . \tag{3.4}$$

Therefore, the optimal solution to the optimization problem in (3.1) is the eigenstate associated with the highest energy (eigenvalue) of H_C.

Remark 1. It might appear to the reader that the highest energy eigenstate could be easily computed classically because the H_C is a diagonal matrix. However, we need to perform 2^n evaluations of the objective function C to build the matrix. This is the same as performing an exhaustive search over all possible solutions. Also, we note that the dimensions of H_C are exponentially large in problem size n. Therefore, H_C requires an exponential amount of time and storage resources. As a result, it is never formulated explicitly. As we discuss next, it is the evolution operator $U(t, t_0)$ of the resulting Schrödinger equation (recall Section 2.9) that we would like to implement as a quantum circuit with at least n qubits.

3.4 Adiabatic theorem in the optimization context

In the previous chapter, we have seen that solutions to the time-independent Schrödinger equation are eigenstates of H_C. However, we are interested in finding a particular eigenstate $|x^*\rangle$ that is associated with the largest eigenvalue E_0 of H_C. Both $|x^*\rangle$ and E_0 are unknown. If we were to find the eigenstate $|x^*\rangle$ classically, we would have to find 2^n eigenvectors of H_C and perform 2^n evaluations of $C(x)$. However, it is possible to find the desired eigenstate without performing an exponential amount of computation. This is possible due to the adiabatic theorem discussed earlier in Section 2.10

We now put Theorem 1 in the combinatorial optimization context. First, let us define a Hamiltonian $H(t)$. We want $H(t)$ such that $H(0) = H_{init}$ is an initial Hamiltonian whose ground eigenstate is known and can be *prepared* in constant time and $H(T) = H_C$. We do this by defining $H(t)$ as follows:

$$H(t) = (1 - s(t))H_{init} + s(t)H_C, \tag{3.5}$$

where $s(t) \in [0,1]$ is a monotonically increasing continuous function such that $s(0) = 0$ and $s(T) = 1$. The function $s(t)$ is often called the *adiabatic schedule*. Let $|\psi(0)\rangle$ be the ground state of $H(0)$ with the associated largest eigenvalue $E(0)$. Suppose that all assumptions in Theorem 1 hold. Then the adiabatic theorem guarantees that the ground state $|\psi(0)\rangle$ of H_{init} will evolve to the ground state $|\psi(T)\rangle$ of H_C given that the transition from H_{init} to H_C is slow enough.

Following the discussion in Section 2.9, in order to find an optimal solution to (3.1) we would like to find the unitary operator $U(t,0)$ that satisfies the following Schrödinger equation

$$\mathrm{i}\hbar \frac{d}{dt} U(t,0) |\psi(0)\rangle = H(t) U(t,0) |\psi(0)\rangle , \tag{3.6}$$

where $|\psi(0)\rangle$ is the highest energy eigenstate of $H_{init} = H(0)$ and $t \in [0,T]$. Given that the conditions of Theorem 1 are satisfied, the unitary matrix $U(T,0)$ will evolve the initial state $|\psi(0)\rangle$ into the highest energy eigenstate $|\psi(T)\rangle$ of $H_C = H(T)$. Therefore, final state $|\psi(T)\rangle$ is an optimal solution to a CO problem.

3.5 Building the QAOA circuit

In this section, we construct a QAOA circuit by finding the evolution operator $U(t,0)$ and then approximating it by discretizing the evolution time T and applying the first order Trotter approximation. We assume that H_C represents $C(x)$ in (3.1) and H_{init} represents a Hamiltonian whose ground state is known and easily preparable. We also assume that H_C and H_{init} do not commute. We will discuss the structure of H_C and H_{init} and their non-commutativity in Section 3.8.1 and Section 3.8.2 respectively.

To this end, we would like to solve (3.6) for $U(t,0)$. Performing the calculation steps outlined in Appendix B we find that

$$\begin{aligned} U(t,0) &= \lim_{p\to\infty} \prod_{k=0}^{p} \exp\left\{-\frac{\mathrm{i}\Delta t}{\hbar} H(k\Delta t)\right\} \qquad (3.7) \\ &\equiv \lim_{p\to\infty} \exp\left\{-\frac{\mathrm{i}\Delta t}{\hbar} H(p\Delta t)\right\} \exp\left\{-\frac{\mathrm{i}\Delta t}{\hbar} H((p-1)\Delta t)\right\} \cdots \exp\left\{-\frac{\mathrm{i}\Delta t}{\hbar} H(0)\right\}, \end{aligned}$$

where $\Delta t = t/p$. In order to approximate (3.7) we choose p to be finite and apply the first order Trotterization presented in (2.54). For $k = 0, ..., p$ the Trotterization yields

$$\begin{aligned} \exp\left\{-\frac{\mathrm{i}\Delta t}{\hbar} H(k\Delta t)\right\} &= \exp\left\{-\frac{\mathrm{i}\Delta t}{\hbar}\Big[(1-s(k\Delta t))H_{init} + s(k\Delta t)H_C\Big]\right\} \\ &= \exp\left\{-\frac{\mathrm{i}\Delta t}{\hbar}(1-s(k\Delta t))H_{init}\right\} \exp\left\{-\frac{\mathrm{i}\Delta t}{\hbar} s(k\Delta t) H_C\right\} + \mathcal{O}(\Delta t^2). \end{aligned} \tag{3.8}$$

Next, we set $t = T$. Since $1/\hbar$ in (3.7) just scales the Hamiltonian $H(t)$ we can drop it or absorb into $H(t)$. The Trotterization with finite p and $k = 1$ yields a *p-layerd* circuit

$$\begin{aligned}\hat{U}(T,0) = \prod_{k=1}^{p} \exp\{-\mathrm{i}c_k H_{init}\} \exp\{-\mathrm{i}b_k H_C\} \qquad (3.9)\\ \equiv \exp\{-\mathrm{i}c_p H_{init}\} \exp\{-ib_p H_C\} \cdots \exp\{-\mathrm{i}c_1 H_{init}\} \exp\{-\mathrm{i}b_1 H_C\}\end{aligned}$$

with c_k, b_k defined as

$$\begin{aligned}c_k &= (1 - s(k\Delta t))\,\Delta t, \qquad (3.10)\\ b_k &= s(k\Delta t)\Delta t.\end{aligned}$$

In the equations above, we have $\Delta t = T/p$. Note that we dropped the $k = 0$ layer in (3.9). This is possible, since $\exp\{-\mathrm{i}b_0 H_C\}$ is the identity (as $b_0 = 0$) and $\exp\{-\mathrm{i}c_0 H_{init}\}$ does nothing but change the global phase of the initial state. The unitary matrix $\hat{U}(T,0)$ is the QAOA circuit with p layers. The kth layer of a circuit is given by the unitary matrix $\exp\{-\mathrm{i}c_k H_{init}\} \exp\{-\mathrm{i}b_k H_C\}$ – see Fig. 3.1. If the conditions in Theorem 1 hold and the evolution time is slow enough (i.e. T is sufficiently large), the theorem guarantees that by the end of the evolution described by $\hat{U}(T,0)$ the initial ground state $|\psi(0)\rangle$ of H_{init} will approximately evolve into the ground state of H_C i.e.

$$|x^*\rangle \approx \hat{U}(T,0)\,|\psi(0)\rangle\,. \qquad (3.11)$$

We note that the number of circuit layers does not depend on a combinatorial problem size but only on the approximation parameter p. As p and T approach infinity (3.11) becomes exact. The coefficients c_k and b_k in (3.10) are defined in terms of the adiabatic schedule function $s(t)$ and should be viewed as parameters of a circuit. This means the QAOA circuit parametrization is completely described by the values of p, T and the function $s(t)$. Alternatively, the coefficients can be viewed as free variables and determined by means of classical variational techniques in classical continuous optimization [19, 36, 50] discussed in Section 3.6.

We complete this section by briefly outlining the main ideas used in the derivation QAOA. The QAOA circuit can be viewed as a Trotterized version of the solution to the Schrödinger equation. Given that the evolution time T is long enough the adiabatic theorem guarantees that by the end of the evolution the system will be in the ground state of the problem Hamiltonian. The circuit complexity depends on the number of sub-intervals p and the evolution time T. It is straightforward to see that a p-layered circuit has a total of $2p$ parameters, see Figure 3.1.

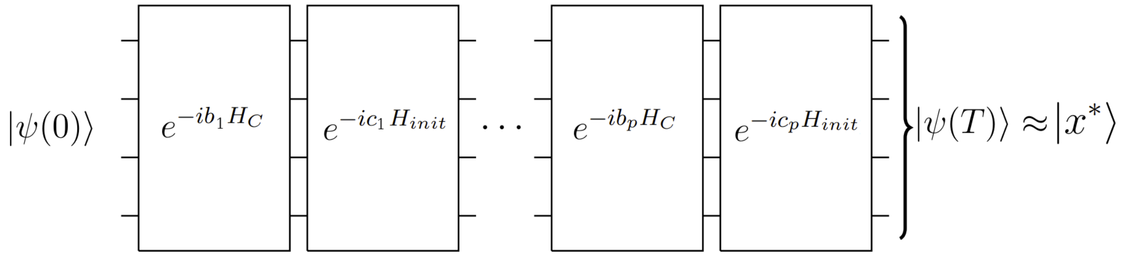


Figure 3.1: The general structure of the p-layered QAOA circuit. We can see the alternating application of unitary matrices associated with H_C and H_{init}. Each layer consists of the unitary matrices $\exp\{-b_k H_C\}$ and $\exp\{-c_k H_{init}\}$ where (b_k, c_k) is a pair of parameters for $k = 1, ..., p$.

3.6 Determining parameters of a circuit with a variational approach

In the previous section, we derived a general structure of the QAOA circuit. We showed that the circuit has p layers that are parameterized by $2p$ parameters given by the sequence $\theta = \{b_k, c_k\}_{k=1}^{p}$ defined in (3.10). In this section we discuss how to estimate θ given p. In other words, we discuss how to find the best parameterization of a circuit given the fixed number of layers p. Since p is given, it is considered to be a hyperparameter. Usually, the value of p is limited by quantum resources. For example, larger values of p require almost error-free quantum computation, whereas smaller values of p are suitable for current and near-term quantum devices which do not have error correction. Recall that $p \gg 1$ yields deep circuits. If a quantum device does not have error correction, the errors accumulate through layers. This results in a corrupted computation. Therefore, the value of p controls the trade-off between the accuracy of the approximation and the amount of errors we are ready to tolerate.

Given that the current and near-term devices do not have error correction, it is safe to assume that practical values of p are almost always insufficient for a good approximation of the evolution process. Hence, there is a need to compensate for errors and small values of p. This is achieved with the variational approach introduced in the 2014 landmark paper [46]. The proposed approach and its improvements are currently used in almost all practical applications. The idea behind the variational approach is to determine a circuit's parameterization through a mix of classical and quantum computation and applying the variational principles [5] widely used in physics and chemistry. There are two important reasons to use the proposed approach. First, it finds the best parameterization of a circuit for a given value of p. Second, the variational nature of the method allows one to find a parameterization that adapts to certain errors that occur in a given quantum hardware [39].

3.6.1 The variational approach as optimization of expected energy

Suppose we want to minimize $C(x)$ given in (3.1). As stated before, this is equivalent to finding the ground eigenstate of the Hamiltonian H_C. With the variational approach, instead of finding the ground state, we attempt to find a state that minimizes the average energy (also called *the expected energy*) of H_C. In what follows, we will define the notion of the expected energy and discuss why it makes sense to minimize it.

Let $|\psi(\theta)\rangle$ denote the state modeled by a QAOA circuit parameterized by θ. We will write $|\psi(\theta)\rangle$ in the basis given by the eigenstates of H_C

$$|\psi(\theta)\rangle = \sum_{k=0}^{2^n-1} a_k(\theta)\,|k\rangle$$

such that $H_C\,|k\rangle = E_k\,|k\rangle$ and k is an integer representation of a binary string of length n that also corresponds to the kth canonical basis vector. Then the *expected energy* of the Hamiltonian H_C with respect to $|\psi(\theta)\rangle$ is given by

$$\langle H_C\rangle_\theta := \langle\psi(\theta)|\,H_C\,|\psi(\theta)\rangle = \sum_{k=0}^{2^n-1} |a_k(\theta)|^2 E_k = \sum_{k=0}^{2^n-1} \Pr(|k\rangle \mid \theta) E_k.$$

The variational theorem of quantum mechanics states that

$$E_{min} \leq \langle H_C\rangle_\theta \tag{3.12}$$

where $E_{min} = E_{2^n-1}$ is the lowest energy (eigenvalue) of the Hamiltonian H_C. As a result the best choice of θ to approximate a ground eigenstate is the choice which minimizes $\langle H_C\rangle_\theta$. If $E_{min} = \langle H_C\rangle_\theta$ then we measure an optimal solution with certainty.

3.6.2 The variational algorithm step by step

The mechanics of the variational approach can be described as follows:

1. Randomly initialize $\theta = \{b_k, c_k\}_{k=1}^{p}$ (classical computation).

2. Estimate the expected energy $\langle H_C\rangle_\theta$ of the problem Hamiltonian H_C by repetitively sampling the circuit with parameters θ (quantum computation).

3. Using the estimated expected energy $\langle\hat{H}_C\rangle_\theta$ update θ (classical computation).

4. Repeat Steps 2, 3 until some stopping criterion is met.

5. Sample the parametrized circuit to obtain a set of candidate solutions (quantum computation).

The process above is called the *feedback loop*. The name stems from the fact that classical computation provides the "feedback" for a quantum computer based on its estimate of the expected energy. A simplified diagram of the process is illustrated in Fig 3.2.

Remark 2. Note that obtaining a single sample of the energy of H_C takes one execution of a circuit with a subsequent measurement. The process of measurement destroys the quantum state $|\psi(\theta)\rangle$ and returns a single probabilistic outcome of energy $|E_k\rangle$. Therefore, repetitive circuit executions are needed to estimate the expected energy of H_C with respect to $|\psi(\theta)\rangle$. The procedure of repetitive sampling can be very costly.

3.6.3 Methods for variational parameter update

The parameter update can be accomplished with gradient-based or gradient-free optimization methods. For example, the most naive update rule may rely on the finite difference method. Let θ_i denote the ith component of $\theta \in \mathbb{R}^{2p}$. Then the update has the following form

$$\theta_k \leftarrow \theta_k - \beta \cdot \frac{\langle \hat{H}_C \rangle_{\theta+h} - \langle \hat{H}_C \rangle_{\theta-h}}{2h_k}, \tag{3.13}$$

where $\langle \hat{H}_C \rangle_\theta$ denotes the estimated energy at θ, $h \in \mathbb{R}^{2p}$ is proportional to the kth standard basis vector and $\beta > 0$ is the step size.

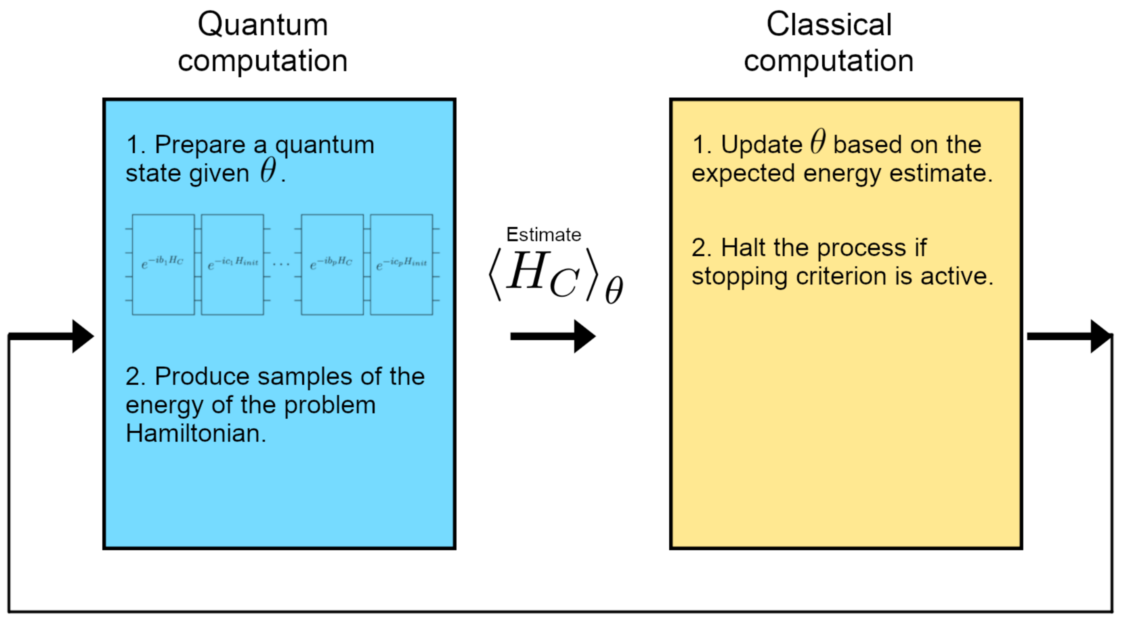

Quantum computation
1. Prepare a quantum state given θ.
$e^{-ib_1 H_C}$
$e^{-ic_1 H_{init}}$
$e^{-ib_p H_C}$
$e^{-ic_p H_{init}}$
2. Produce samples of the energy of the problem Hamiltonian.
Estimate
$\langle H_C \rangle_\theta$
Classical computation
1. Update θ based on the expected energy estimate.
2. Halt the process if stopping criterion is active.

Figure 3.2: The quantum-classical feedback loop for finding the best parameters θ.

While the variational approach allows for shallower circuits and can adapt the parameters to compensate for some errors, it suffers from several issues. First, the parameter update rule, which relies on gradient-like methods, requires many iterations. Let a single run (also called a shot) of a quantum computer refer to a quantum computation necessary to produce one sample of energy (eigenvalue E_k) of H_C with the QAOA circuit. Suppose that N_1 runs are used to estimate $\langle H_C \rangle_\theta$. If N_1 is small, then the *estimator of the expected energy* with respect to $|\psi(\theta)\rangle$, which we denote as $\langle \hat{H}_C \rangle_\theta$ will have a high variance. As a result, the gradient update will have high variance as well. The variance of the estimator $\langle \hat{H}_C \rangle_\theta$ is $O(1/N_1)$ [12, Section 5.4]. Therefore, to bound the variance by a positive ϵ_{var}, we need $N_1 = O(1/\epsilon_{var})$ runs. Since $\theta \in \mathbb{R}^{2p}$, each exact or approximate gradient requires at least $2p \cdot 2N_1$ runs of a quantum computer. If there are N_2 iterations (parameter updates), it takes at least $2p \cdot 2N_1 \cdot N_2$ total runs. For example, assume we have access to $\langle H_C \rangle_\theta$ and the exact gradients with respect to θ. Then to converge to a local minimum with an error $\epsilon_{grad} > 0$ we have $N_2 = O(1/\epsilon^2_{grad})$ [6, Section 4.7.3]. The above discussion suggests that even in the idealized scenario where we have access to the true expected energy and exact gradients, the variational approach may be quite costly and only provide locally optimal parameter vector θ.

3.7 Determining parameters of a circuit with a discretized adiabatic process

It is common to use the aforementioned variational approach to find suitable parametrization of a circuit. We have seen that such an approach is fairly expensive because finding locally optimal parameters requires many iterations and circuit reruns. This is further exacerbated by barren plateaus – a phenomenon of exponentially vanishing gradients [54]. Another serious disadvantage is that the variationally determined parameters do not necessarily generalize to similar problem instances. Hence, every problem instance requires determining a new parametrization. Finally, variational approaches do not explicitly optimize for the probability of sampling the optimal solution but rather optimize for the highest expected energy. Therefore, maximizing the expected energy does not guarantee improvement in the probability of the optimal solution. Extensive numerical studies highlighting these issues are presented in [50].

This section presents an alternative approach for finding suitable parametrization that does not require the iterative feedback loop. An alternative approach is called the Discretized Adiabatic Quantum Computation (DAQC). DAQC determines the parameters $\theta = \{b_k, c_k\}_{k=1}^p$ according to the adiabatic schedule. Specifically, we compute θ using (3.10) which requires a monotonically increasing function $s(t)$ such that $s(0) = 0$ and $s(T) = 1$ and discretized time T (hence the name DAQC). The DAQC approach is defined by the following steps.

1. Given T and p, discretize T to get a sequence $\{k\Delta t\}_{k=1}^{p}$ where $\Delta t = T/p$ (classical computation).
2. Choose a suitable $s(t)$. The most trivial choice is $s(t) = t/T$ for $t \in [0, T]$ (classical computation).
3. For $k = 1, ..., p$ compute $c_k = (1 - s(k\Delta t))\,\Delta t$ and $b_k = s(k\Delta t)\Delta t$ (classical computation).
4. Sample the parametrized circuit to obtain a set of candidate solutions (quantum computation).

Note that unlike the variational approach, the DAQC does not have a feedback loop, nor does it require any iterations. The use of a quantum computer happens only to sample the circuit. We will discuss how many times the circuit must be sampled to observe the optimal solution with high probability in Chapter 5.

3.7.1 Minimum eigengap and circuit's depth

Recall that the adiabatic theorem requires the transition time from the initial Hamiltonian to the problem Hamiltonian to be sufficiently slow. Specifically, (2.52) suggests that the transition time T depends on the spectral gap λ. In what follows, we discuss how λ contributes to the circuit complexity if the DAQC approach is used.

Since we are interested in the ground state, we will consider the spectral gap between the ground state and the closest higher energy state. This spectral gap is known as the *minimum eigengap* g_{min} – the smallest difference between the two lowest eigenvalues of $H(t)$ [14]. Let $E_1(t) < E_2(t)$ denote the lowest eigenvalues of $H(t)$. Then the minimum eigengap is given by

$$g_{min} := \min_{0 \le t \le T} \{E_2(t) - E_1(t)\}.$$

According to the adiabatic theorem, if the eigengap g_{min} between the two lowest eigenvalues strictly greater than zero then the final eigenstate can be made arbitrarily close to the ground state of the final Hamiltonian.

We note that if the eigengap is arbitrarily small, it will take arbitrarily long to complete the evolution process. Recalling that the error of the Trotter approximation in (3.8) depends on $\Delta t = T/p$, we deduce that smaller values of p produce larger Trotterization errors. Therefore, a long evolution time T requires larger values of p to keep the approximation errors small. This results in larger circuits. Interestingly, it has been argued [40] that the eigengap decreases exponentially with the problem size n. Therefore, some large combinatorial problems, in theory, may require exponentially many layers in the QAOA circuit.

In practice, computing a lower bound on T is extremely hard. Usually, the values of T and p are chosen to be hyperparameters of an algorithm and determined through hyperparameter-tuning.

3.7.2 DAQC scheduling

In this section, we give an explicit description of the adiabatic schedule $s(t)$ used in this thesis. We would like to have a single adiabatic schedule that maximizes the probability of observing optimal solutions to multiple *problem instances* of the same *problem class*. A problem class can be, for example, an n–variable MaxCut problem or an n–variable travelling salesman problem.

We adapt the QA scheduling presented in [47]. For better control of the curvature of the schedule function, we use a cubic polynomial approximation proposed in [50]. The general form is given as follows

$$s(t; a, T) = \frac{t}{T} + a \cdot \frac{t}{T}\left(\frac{t}{T} - \frac{1}{2}\right)\left(\frac{t}{T} - 1\right). \tag{3.14}$$

The parameters a and T define the slope and the evolution time respectively. The goal is to determine optimal parameters a^* and T^* that maximize the probability of observing an optimal solution for similar problem instance of the same size. This is usually achieved by using a "training" set of problem instances of size n and performing a Random Search optimization [7] on a and T to directly minimize the median of Time To Solution (TTS) for all problems in the "training" set. Once a^* and T^* are determined, we can reuse $s(t; a^*, T^*)$ across all similar problem instances of size n and achieve high probability of success. The graphs of several functions $s(t; a, T)$ for different values of a are illustrated in Fig 3.3 (a). Fig 3.3(b) illustrates how the coefficients b_k and c_k defined in (3.10) relate to $s(t; a, T)$. Essentially, each coefficient approximates the area under or above $s(t; a, T)$ on a segment of length Δt.

3.8 The QAOA for MaxCut problems

In the previous sections, we derived the quantum circuit for QAOA in the general case and discussed how to select parameters and the schedule function. In this section, we discuss QAOA for the MaxCut problem presented in Section 3.2. In particular, we will convert the objective function of the MaxCut problem presented in Section 3.2 into the Hamiltonian H_C and present a problem-independent initial Hamiltonian H_{init}. We have seen that the Hamiltonian H_C represents an objective function of a combinatorial problem. This means the structure of H_C is problem-dependent. This further implies that the unitary matrices $\exp\{-\mathrm{i} b_k H_C\}$ for $k = 1, ..., p$ are also problem-dependent.

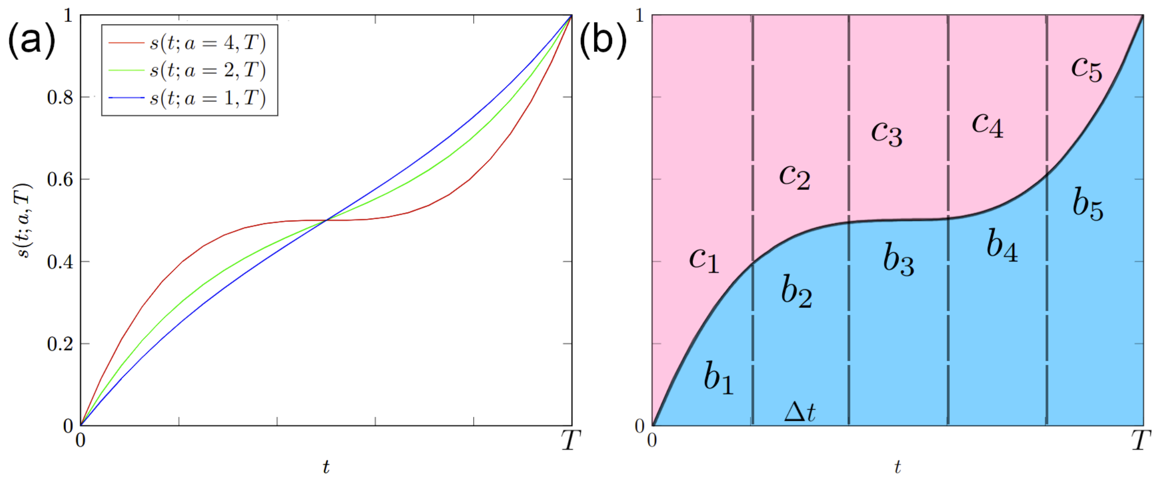


Figure 3.3: (a) The adiabatic schedule function $s(t; a, T)$ for $a = 1, 2, 4$ The parameter a determines the rate of transition. For a linear transition we set $a = 0$. For a transition whose rate of change is zero at $t = T/2$ we set $a = 4$. The best parameters T and a are found through hyperparamter tuning. (b) The curve $s(t; a = 4, T)$ together with a 5-layer circuit's coefficients c_k, b_k for $k = 1, \dots, 5$.

3.8.1 The problem Hamiltonian H_C

We now cast the combinatorial problem in (3.2) into the problem of finding the highest energy eigenstate of the problem Hamiltonian H_C. In order to construct the problem Hamiltonian H_C, we first obtain an equivalent representation of (3.2) by substituting binary variables x_i with the *spin variables* $s_i \in \{-1, 1\}$ using the following relation:

$$x_i = \frac{1 - s_i}{2} \quad \text{for } i = 1, \dots, n. \tag{3.15}$$

This substitution relates abstract values 0 and 1 with *subatomic particle spins* which are observable (or measurable) quantum mechanical values. The substitution yields:

$$\begin{aligned} \sum_{(i,j)\in E} (x_i + x_j - 2x_i x_j) &= \sum_{(i,j)\in E} \left(\frac{1}{2}(1 - s_i) + \frac{1}{2}(1 - s_j) - \frac{2}{4}(1 - s_i)(1 - s_j) \right) \\ &= \sum_{(i,j)\in E} \frac{1}{2} \left(2 - s_i - s_j - (1 - s_j - s_i + s_i s_j)\right) \\ &= \sum_{(i,j)\in E} \frac{1}{2}(1 - s_i s_j). \end{aligned} \tag{3.16}$$

Therefore an equivalent MaxCut formulation in terms of the spin variables is

$$\max_{s\in\{-1,1\}^n} \sum_{(i,j)\in E} \frac{1}{2}(1 - s_i s_j). \tag{3.17}$$

It is easy to verify that if s_i and s_j get distinct spin assignments for some vertex pair $(i, j) \in E$ then the contribution to the objective function is 1. Otherwise, the contribution is 0. Therefore, the new formulation is equivalent to the previous formulation.

The next step is to represent the objective function in (3.17) as a problem Hamiltonian. Again, this is done to represent the objective function in (3.17) as an observable quantum mechanical quantity. Specifically, we associate the values of the objective functions with observable energy levels of a Hamiltonian. To this end we need to establish the equivalence relation between the spin variable s_i and Pauli Z operator Z_i defined in (2.11) and (2.23).

Note that $|x\rangle = |x_1 \dots x_n\rangle$ is an eigenvector of Pauli Z operators Z_i and Z_iZ_j defined in (2.11) and (2.25) with possible eigenvalues $-1, +1$. These eigenvalues are also observable quantum mechanical quantities and they represent the spin variable s_i. The relation between Z_i, $|x\rangle$ and s_i is given by the following eigenvalue equation:

$$\begin{aligned}
Z_i |x\rangle &= (I \otimes \cdots \otimes Z \otimes \cdots I) |x_1 \dots x_n\rangle \\
&= I |x_1\rangle \otimes \cdots \otimes Z |x_i\rangle \otimes \cdots \otimes I |x_n\rangle \\
&= |x_1\rangle \otimes \cdots \otimes (-1)^{x_i} |x_i\rangle \otimes \cdots \otimes |x_n\rangle \\
&= (-1)^{x_i} |x\rangle \\
&= s_i |x\rangle .
\end{aligned}$$

Similarly, we have

$$Z_iZ_j |x\rangle = (-1)^{x_i}(-1)^{x_j} |x\rangle = s_is_j |x\rangle .$$

Therefore, the relation is:

$$Z_i |x\rangle = s_i |x\rangle \text{ and } Z_iZ_j |x\rangle = s_is_j |x\rangle . \tag{3.18}$$

This relation allows us to obtain the problem Hamiltonian H_C by simply multiplying the objective value in (3.17) by $|x\rangle$.

$$\sum_{(i,j)\in E} \frac{1}{2}(1 - s_is_j) |x\rangle = \sum_{(i,j)\in E} \frac{1}{2}(|x\rangle - s_is_j |x\rangle) = \sum_{(i,j)\in E} \frac{1}{2}(I |x\rangle - Z_iZ_j |x\rangle). \tag{3.19}$$

Hence, the problem Hamiltonian H_C is as follows:

$$H_C = \sum_{(i,j)\in E} \frac{1}{2}(I - Z_iZ_j). \tag{3.20}$$

Since Z_i for $i = 1, .., n$ is a diagonal matrix of size $2^n \times 2^n$, the product Z_iZ_j is also a diagonal matrix of size $2^n \times 2^n$. It follows that H_C is a diagonal Hamiltonian. The eigenvalues of H_C

are the objective function values. To see this, we write:

$$\begin{aligned} H_C \left|x_1 \dots x_n\right\rangle &= \sum_{(i,j)\in E} \frac{1}{2}(I - Z_i Z_j) \left|x_1 \dots x_n\right\rangle \\ &= \sum_{(i,j)\in E} \frac{1}{2}\left(1 - s_i s_j\right) \left|x_1 \dots x_n\right\rangle . \end{aligned}$$

Given the relation between x_i and s_i in (3.15) we can further write:

$$\begin{aligned} H_C \left|x_1 \dots x_n\right\rangle &= \sum_{(i,j)\in E} \frac{1}{2}\left(1 - s_i s_j\right) \left|x_1 \dots x_n\right\rangle \\ &= \sum_{(i,j)\in E} \left(x_i + x_j - 2x_i x_j\right) \left|x_1 \dots x_n\right\rangle \\ &= C(x) \left|x_1 \dots x_n\right\rangle . \end{aligned}$$

where $C(x) = \sum_{(i,j)\in E}(x_i + x_j - 2x_i x_j)$ is the objective function of the MaxCut problem (3.2).

3.8.2 An initial Hamiltonian H_{init}

We now discuss the initial Hamiltonian H_{init}. Recall that H_{init} must be chosen such that its highest energy state is easily preparable (can be prepared in a constant time). Another core requirement is that H_{init} does not commute with H_C (see Appendix C). This is a necessary condition for the adiabatic theorem to hold. Recall that Theorem 1 requires the eigenvalues of the Hamiltonian $H(t)$ to be separated by a positive spectral gap λ for all $t \in [0, T]$ and from Section 3.7 we know that the evolution time T is inversely proportional to a cubic or quadratic polynomial in g_{min} – see Theorem 1. As shown in Appendix C, if H_{init} commutes with H_C then $g_{min} = 0$. In this case, the entire process fails. A customary choice of H_{init} is the *transverse field Hamiltonian* [14] given below

$$H_{init} = \sum_{i=1}^{n} X_i \tag{3.21}$$

where X_i previously defined in (2.11) and (2.23). H_{init} does not commute with H_C. To see this, let us define the commutator of two squared matrices A, B to be $[A, B] = AB - BA$. Clearly, A and B commute if and only if $[A, B] = 0$. We now compute the commutator of

H_{init} and H_C and show that it is non-zero.

$$[H_{init}, H_C] = \left[\sum_{k=1}^{n} X_k, \sum_{(i,j)\in E} \frac{1}{2}(I - Z_i Z_j)\right] = \frac{1}{2}\sum_{k=1}^{n}\sum_{(i,j)\in E}[X_k, (I - Z_i Z_j)]$$
$$= \frac{1}{2}\sum_{k=1}^{n}\sum_{(i,j)\in E}([X_k, I] - [X_k, Z_i Z_j]) = -\frac{1}{2}\sum_{k=1}^{n}\sum_{(i,j)\in E}[X_k, Z_i Z_j].$$

If $k \neq i, j$ then $[X_k, Z_i Z_j] = 0$. For $k = i$ (or $k = j$) we have:

$$[X_k, Z_i Z_j] = [X_i, Z_i Z_j] = (XZ)_i Z_j - (ZX)_i Z_j = (XZ - ZX)_i Z_j = [X, Z]_i Z_j = -2\mathrm{i} Y_i Z_j.$$

Therefore, we can write:

$$[H_{init}, H_C] = -\frac{1}{2}\sum_{k=1}^{n}\sum_{(i,j)\in E}[X_k, Z_i Z_j] = \sum_{k=1}^{n}\sum_{(i,j)\in E} \mathrm{i} Y_i Z_j\,(\delta_{k,i} + \delta_{k,j}) \neq 0.$$

where $\delta_{k,i} = 1$ if $i = k$, and $\delta_{k,i} = 0$ otherwise.

Since $|+\rangle$ is the eigenvector of X with the eigenvalue $+1$, it is straightforward to see that $|+\rangle^{\otimes n}$ is the highest energy eigenstate of H_{init},

$$H_{init}\,|+\rangle^{\otimes n} = \sum_{i=1}^{n} X_i\,|+\rangle^{\otimes n} = n\,|+\rangle^{\otimes n}. \tag{3.22}$$

Since H_{init} in (3.21) does not commute with H_C and it has an easily preparable highest energy eigenstate $|+\rangle^{\otimes n}$ it is a suitable candidate for the initial Hamiltonian. Therefore, the total time-dependent Hamiltonian $H(t)$ for the MaxCut problem is

$$H(t) = (1 - s(t))\sum_{i=1}^{n} X_i + s(t)\sum_{(i,j)\in E}\frac{1}{2}(I - Z_i Z_j). \tag{3.23}$$

3.8.3 QAOA circuit for a MaxCut problem

In Section 3.5 we have constructed a general QAOA circuit which consists of p layers of alternating unitary matrices $\exp\{-ic_k H_{init}\}$ and $\exp\{-ib_k H_C\}$ for $k = 1, ..., p$. We now give an explicit description of a QAOA circuit for a MaxCut problem.

First, consider the unitary matrix $\exp\{-ic_k H_{init}\}$. Recall that $H_{init} = \sum_{i=1}^{n} X_i$. Then

$$\exp\{-\mathrm{i}c_k H_{init}\} = \exp\left\{-\mathrm{i}c_k \sum_{i=1}^{n} X_i\right\}.$$

Since X_i and X_j commute for $i, j = 1, ..., n$, we can rewrite the above as

$$\exp\left\{-\mathrm{i}c_k \sum_{i=1}^{n} X_i\right\} = \prod_{i=1}^{n} \exp\left\{-\mathrm{i}c_k X_i\right\}. \tag{3.24}$$

Recalling the definition of the $RX(t)$ gate in (2.32) we can write $\exp\{-\mathrm{i}c_k X_i\} = RX_i(2c_k)$. The subindex i indicates that $RX_i(2c_k)$ acts on the ith qubit. Therefore, the unitary matrix $\exp\{-\mathrm{i}c_k H_{init}\}$ can be easily implemented with a product of $RX_i(2c_k)$.

We now consider the unitary matrix $\exp\{-\mathrm{i}b_k H_C\}$. Since all terms in H_C commute we can write

$$\begin{aligned}\exp\{-\mathrm{i}b_k H_C\} &= \exp\left\{-\mathrm{i}b_k \sum_{(i,j)\in E} \frac{1}{2}(I - Z_i Z_j)\right\} \\ &= \prod_{(i,j)\in E} \exp\{-\mathrm{i}b_k \frac{1}{2} I\} \exp\left\{\mathrm{i}b_k \frac{1}{2} Z_i Z_j\right\} \\ &\mathrel{\hat{=}} \prod_{(i,j)\in E} \exp\left\{\mathrm{i}b_k \frac{1}{2} Z_i Z_j\right\}\end{aligned} \tag{3.25}$$

where $\mathrel{\hat{=}}$ denotes equality up to a global phase. Recalling the definition of the $RZZ_{i,j}(\theta)$ gate in (2.40) we can write $\exp\left\{\mathrm{i}\frac{b_k}{2} Z_i Z_j\right\} = RZZ_{i,j}(-b_k)$.

Therefore, combining the results in (3.24)-(3.25) the MaxCut QAOA circuit can be implemented using only 1-qubit RX and 2-qubit RZZ gates. The final p-layered circuit is

$$\hat{U}(T,0) = \prod_{i=1}^{n} RX_i(2c_p) \prod_{(i,j)\in E} RZZ_{i,j}(-b_p) \cdots \prod_{i=1}^{n} RX_i(2c_1) \prod_{(i,j)\in E} RZZ_{i,j}(-b_1), \tag{3.26}$$

and the initial state is given by $|\psi_0\rangle = |+\rangle^{\otimes n}$. Note that the circuit in (3.26) is just a particular case of the general circuit given in (3.9).

In view of the circuit construction process given by (3.16), (3.20) and (3.25) we finalize this section by stating a trivial but important observation:

Observation 1 If an objective function contains a term of the form $x_i x_j$ then a problem Hamiltonian will contain a term of the form $Z_i Z_j$, and the kth layer of a QAOA circuit will contain a corresponding 2-qubit $RZZ_{i,j}$ gate.

3.9 MaxCut QAOA circuit's structure

We now discuss the structure of the obtained circuit and how it is related to quantum hardware. First, note that the structure of the circuit in (3.26) is consistent with the structure illustrated in Fig 3.1. The graph G in a MaxCut problem has n vertices. This results in a circuit with n qubits and each layer having a product of n RX gates. Since the graph G

has $|E|$ edges, each layer has $|E|$ 2-qubit RZZ gates. We immediately deduce that the size of a MaxCut problem (the number of vertices in G) affects the *circuit width* – a number of qubits in a circuit, and the density of the graph (the number of edges in G) affects the connectivity of a circuit. Therefore, a quantum computer must have at least n qubits, and these qubits must be interconnected so that RZZ gates can be realized. For example, a complete graph on n vertices will require all-to-all connectivity in the circuit. That is, each layer will have $n(n-1)/2$ RZZ gates that act on each possible pair of qubits. Due to the current and near-term quantum hardware limitations, all-to-all connectivity introduces overheads in the form of additional gates that may significantly increase the size of a circuit. This means that large dense graphs require much more resources.

The number of layers of a circuit is given by the parameter p. A larger p means a deeper circuit. Recall that p is an approximation parameter. Larger values of p yield a circuit that better approximates the true unitary operator $U(T, 0)$ where the evolution time T depends on the minimum eigengap g_{min}. Therefore, unlike the width of a circuit which entirely depends on the structure of the graph G, the depth of a circuit implicitly depends on the eigenspectrum of a Hamiltonian $H(t)$. Due to the short coherence time of the current and near-term quantum hardware, the practical realization of deep circuits ($p > 1$) is also a challenge.

3.10 Conclusion

In this chapter, we covered QAOA - a quantum algorithm for solving binary combinatorial problems. The algorithm relies on the Schrödinger equation and the adiabatic theorem. The adiabatic theorems states that if the evolution of a quantum system is governed by a Hamiltonian $H(t)$ that varies slowly enough, then the system will stay near its instantaneous ground state. This means the adiabatic evolution can be used to switch gradually from an initial Hamiltonian, whose ground state is known, to a final Hamiltonian, whose ground state encodes the solution to a problem we would like to solve. Due to the theorem, the final eigenstate at time T is guaranteed to be sufficiently close to the solution. To obtain the QAOA circuit, the evolution time is first discretized into p sub-intervals, and then the first-order Trotterization is applied to the solution of the Schrödinger equation. This yields a p-layered QAOA circuit. The time T necessary for an adiabatic evolution is inversely proportional to a polynomial of the minimum eigengap of $H(t)$. This means the smaller the gap is, the longer the evolution time must be. In order to keep the Trotterization error small, larger values of T require larger values of p. It is widely believed that the eigengap decreases exponentially with the problem size. Therefore, the QAOA circuit may need exponentially many layers for a good approximation of the optimal solution.

Chapter 4

Combinatorial problems with constraints

So far, we introduced QAOA for solving combinatorial optimization problems. Starting from the postulates of quantum mechanics and principles of adiabatic evolution we gave a complete construction of QAOA and applied it to the MaxCut problem. Recall that QAOA attempts to find a solution to a binary problem presented in (3.1) where a problem's constraints are integrated into an objective function $C(x)$. This suggests that any constraints present in an optimization problem must be included in the objective function. This requirement creates a number of issues which we will discuss in this chapter.

4.1 Quadratic Unconstrained Binary Optimization

In the previous chapter, we saw how to build a QAOA circuit for a MaxCut problem. Our choice of the MaxCut problem was deliberate for one crucial reason – a wide range of integer CO problems with linear constraints can be reformulated as a MaxCut type problem. It turns out that MaxCut is a particular case of a more general optimization model called QUBO. The QUBO model is expressed by the optimization problem:

$$\min_{x\in\{0,1\}^n} x^T Q x = \min_{x\in\{0,1\}^n} \sum_{i=1}^{n} \sum_{j=i+1}^{n} x_i Q_{i,j} x_j + \sum_{i=1}^{n} Q_{i,i} x_i \tag{4.1}$$

where x is a vector of binary decision variables and Q is $n \times n$ upper triangular matrix of problem coefficients. It is straightforward to see that (4.1) is a generalization of the MaxCut formulation in (3.2). If $C(x)$ in (3.1) is a quadratic or linear function then it can be described by the QUBO formulation in (4.1). Also note that, since the variables are binary, we have $x_i^2 = x_i$. Therefore, we can also write the QUBO problem as follows:

$$\min_{x\in\{0,1\}^n} x^T Q x = \min_{x\in\{0,1\}^n} \sum_{i\leq j} x_i Q_{i,j} x_j .$$

4.2 QUBO issues with constrained problems

Many real-world optimization problems include constraints. The most common approach to solving constrained CO problems with QAOA is by reformulating them into QUBO, followed by recasting the QUBO formulation as the problem of finding the ground state of the corresponding problem Hamiltonian.

QUBO is incredibly successful and popular in quantum and quantum-inspired optimization. QUBO is the go-to model for many classical and quantum optimization hardware producers. Almost all cutting-edge optimization hardware is designed to solve QUBO problems. For example, optimization devices such as D-Wave's quantum annealer [9], NTT's Coherent Ising Machine [25], Fujitsu's Digital Annealer [4] and Toshiba's Simulated Quantum Bifurcation Machine [21] are designed to solve QUBO problems. This popularity can be explained by QUBO's equivalence to the Ising model [10] often used in adiabatic computation. Platform-agnostic quantum algorithms like QAOA also require a problem to be reformulated as QUBO or as a more general PUBO – Polynomial Unconstrained Binary Optimization [20].

While QUBO is a prevalent model used in quantum or quantum-inspired optimization, it has several shortcomings. Reformulation of constrained problems into QUBO often results in a QAOA circuit with increased circuit depth, width and connectivity. For example, all inequality constraints must be incorporated into an objective function as quadratic penalties. Such quadratic penalties may require many additional auxiliary binary variables that significantly increase the search space of a problem and make the optimization landscape more rugged. Also, converting constraints into squared penalty terms often results in all-to-all interaction between a problem's variables and hence qubits representing the variables.

We demonstrate these issues on widely used and studied integer programs called binary linear problems with inequality constraints. The canonical formulation of such problems is:

$$\begin{aligned} &\min_x q_0^T x \\ &q_i^T x \geq c_i \quad \text{for} \quad i = 1, ..., m \\ &x \in \{0,1\}^n, \end{aligned} \tag{4.2}$$

where $q_0 \in \mathbb{R}^n$, $q_i \in \mathbb{Z}^n$ and $c_i \in \mathbb{Z}$ for $i = 1, ..., m$. To use QAOA, we convert (4.2) into a QUBO problem by incorporating all constraints as quadratic penalties with slack variables as follows:

$$\min q_0^T x + \sum_{i=1}^{m} \gamma_i \left(q_i^T x - c_i - W_i\right)^2 \tag{4.3}$$

such that

$$W_i = \sum_{k=0}^{\lfloor \log_2(c_i^{max})\rfloor - 1} 2^k y_k^{(i)} + (c_i^{max} + 1 - 2^{\lfloor \log_2(c_i^{max})\rfloor}) y^{(i)}_{\lfloor \log_2(c_i^{max})\rfloor}, \text{ for } i = 1, \dots, m$$

$$y^{(i)} \in \{0,1\}^{\log_2(c_i^{max})+1}, \text{ for } i = 1, \dots, m$$

$$x \in \{0,1\}^n.$$

In this problem, we minimize over binary vectors x and $y^{(i)}$ for $i = 1, \dots, m$. The scalar $\gamma_i > 0$ is a penalty coefficient and $0 \leq W_i \leq c_i^{max} := \max_x\{q_i^T x - c_i\}$ is an integer slack variable given by a binary expansion of $M_i = \lfloor \log_2(c_i^{max})\rfloor + 1$ additional auxiliary binary variables $y_k^{(i)} \in \{0,1\}$. Note that (4.3) is a quadratic polynomial over binary variables x_j for $j = 1, \dots n$ and $y_k^{(i)}$ for $k = 0, \dots, \lfloor \log_2(c_i^{max})\rfloor$ and $i = 1, \dots, m$. If a constraint is satisfied, $q_i^T x \geq c_i$, then the quadratic penalty is zero, $\gamma_i \left(q_i^T x - c_i - W_i\right)^2 = 0$. However, if a constraint is violated, $q_i^T x < c_i$, then we have a non-zero quadratic penalty, $\gamma_i \left(q_i^T x - c_i - W_i\right)^2 > 0$, because $0 \leq W_i$. In the following subsections we will highlight the issues of the QUBO formulation.

4.2.1 Issue with additional auxiliary qubits

From (4.3) it is clear that the QUBO formulation has significantly more additional binary variables. Specifically, the new problem has $n + \sum_{i=1}^m M_i$ variables instead of n variables in the canonical formulation (4.2). The search space of the new problem is $2^{n+\sum_{i=1}^m M_i}$. Also, the resulting QAOA circuit will have $\sum_{i=1}^m M_i$ additional auxiliary qubits. Recall that $M_i = \lfloor \log(c_i^{max})\rfloor + 1$. If c_i^{max} is large, the number of auxiliary qubits may overshadow the number of problem qubits n. It follows that even relatively small problems in n may become much more challenging when converted to QUBO.

4.2.2 Issues with circuit connectivity

We now discuss how the QUBO formulation in (4.3) may affect a circuit's connectivity. Suppose that for some $i_0 \in \{1, \dots, m\}$ we have a vector q_{i_0} with no zero components, i.e. $q_{i_0,j} \neq 0$ for $j = 1, \dots, n$. Then the quadratic penalty $\gamma_{i_0}(q_{i_0}^T x - c_i - W_{i_0})^2$ has $\binom{n+M_{i_0}}{2}$ quadratic terms of the form $x_j x_v$, $x_j y_k^{(i_0)}$ and $y_k^{(i_0)} y_u^{(i_0)}$ for $j, v = 1, \dots, n$ and $k, u = 0, \dots, \lfloor \log_2(c_{i_0}^{max})\rfloor$. That is, every variable "interacts" with every other variable. From the QAOA circuit derivation in Section (3.8.3) and Observation 1 we deduce that every layer of a circuit will have $\binom{n+M_{i_0}}{2}$ 2-qubit RZZ gates that act on each possible pair of qubits. Therefore, each layer of a circuit is all-to-all connected. This makes the QAOA circuit an extremely computationally demanding circuit.

Moreover, if each layer is all-to-all connected, the circuit depth depends on the parameter p and the QUBO problem size $N = n + \sum_{i=1}^{m} M_i$. In this case, the circuit depth is $O(Np)$ – for a more detailed discussion on this point, see Appendix E. The QUBO circuit structure is illustrated in Fig 4.1.

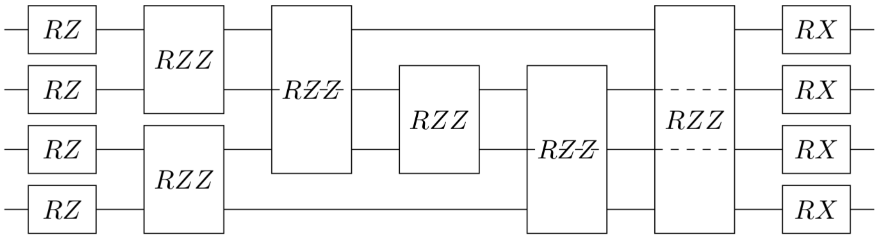


Figure 4.1: The k-th layer of the 4-qubit QUBO circuit. Rotation angles are not displayed for clarity. The circuit is all-to-all connected since each qubit interacts with every other qubit through RZZ gates. The dashed wires plotted over some RZZ gates denote wires that are not affected by the gate.

4.2.3 Issues from classical optimization perspective

From the optimization perspective, each additional variable $y_k^{(i)}$ doubles the problem search space. Hence, the search space increases exponentially with respect to the number of additional variables. Moreover, the energy landscape becomes fairly rugged as flipping the value of x_j or $y_k^{(i)}$ may lead to high peaks of energy. For example, consider a feasible solution vector $(x^*, y^*) \in \{0,1\}^{n+\sum_{i=1}^{m} M_i}$ such that all penalty terms are zero. If we now flip a single bit $y_k^{*(i)}$ for $k \geq 1$, then penalty term becomes $\gamma_i \left(q_i^T x^* - c_i - W_i\right)^2 = \gamma_i \left(2^k\right)^2$. This implies that solutions that are one-bit flip away yield exponentially high peaks of energy making the energy landscape rugged.

4.2.4 Concluding remarks about QUBO

In this section, we have seen that some CO problems, when converted to QUBO, may require significantly more quantum resources. Problems with inequality constraints require additional auxiliary binary variables, and the number of such variables grows logarithmically with the maximum constraints' value. Quadratic penalties create additional interactions between variables, giving rise to additional 2-qubit RZZ gates. From the optimization perspective, the QUBO formulation has a larger search space, and this makes the problem difficult for any type of algorithm. Moreover, solutions that are one-bit flip away from the current feasible solution are often infeasible and yield exponentially large penalties.

4.3 The novel approach and its advantages

This study departs from the conventional QUBO and investigates the implementation of constrained problems using Lagrangian duality theory. We demonstrate the superiority of the proposed approach by comparing it with the QUBO-based approach in the setting of DAQC (discussed in Section 3.7). The contributions of this study were already listed in Chapter 1. We now repeat them for convenience:

- Quadratic improvement in a circuit complexity and evolution time over the QUBO-based approach for problems with linear inequality constraints.
- Highly parallelizable circuit with problem's size-independent circuit depth.
- Analysis of QUBO- and Lagrangian-based formulations and corresponding circuits on the example of the NP-hard binary Knapsack Problem (KP).
- Numerical benchmark study of QUBO-based and Lagrangian-based DAQC.

4.4 Lagrangian duality

We propose a Lagrangian DAQC protocol for solving binary combinatorial problems with inequality constraints of the form given in (4.2).

We also note that the proposed approach easily generalizes to binary quadratic problems with quadratic inequality constraints of the following form:

$$\begin{aligned} &\min_x x^T Q_0 x \\ &x^T Q_i x \geq c_i \quad \text{for} \quad i = 1, \dots, m \\ &x \in \{0,1\}^n, \end{aligned} \tag{4.4}$$

where $Q_i \in \mathbb{R}^{n\times n}$ is a symmetric matrix for $i = 0, \dots, m$. The applicability of Lagrangian duality for such problems is investigated in [27]. If Q_i is diagonal for $i = 0, \dots, m$ then (4.4) is equivalent to (4.2) due the relation $x_i^2 = x_i$. Therefore, the formulation in (4.4) covers a wide range of linear and quadratically constrained and unconstrained combinatorial problems.

We address the QUBO issues discussed in previous sections by using Lagrangian relaxation which is often used in classical optimization. The Lagrangian dual problem corresponding to the primal constrained problem (4.2) is

$$\max_{\lambda\in\mathbf{R}_+^m} \min_{x\in\{0,1\}^n} q_0^t x + \sum_{i=1}^m \lambda_i \left(c_i - q_i^T x\right), \tag{4.5}$$

where $\lambda_i \geq 0$ for $i = 1, ..., m$ are non-negative Lagrange multipliers. Generally, weak duality holds. That is, if (x^*, λ^*) is an optimal dual pair and D^* is the corresponding objective function value of (4.5), then $D^* \leq P^*$ where P^* is the optimal value of the primal problem (4.2). For x^* to be an optimal solution to the primal problem in (4.2) it must be feasible and satisfy the complementary slackness condition $\sum_{i=1}^{m} \lambda_i^*(c_i - q_i^T x^*) = 0$ [18]. Whenever this condition holds, the duality gap is zero, that is $D^* = P^*$ and x^* is an optimal solution to the primal problem. In the case when x^* is feasible but the complementary slackness is not satisfied, then $\sum_{i=1}^{m} \lambda_i^*(q_i^T x^* - c_i) > 0$ and we call x^* an ϵ*–optimal* solution to the primal problem with $\epsilon = \sum_{i=1}^{m} \lambda_i(q_i^T x^* - c_i)$ [18]. The value of ϵ tell us how far away x^* is from being the optimal solution to the primal problem. Therefore, the Lagrangian relaxation can yield optimal or near-optimal solutions to the primal problem. Unlike QUBO the formulation (4.5) does not require auxiliary variables and squared penalties. As a result, it allows for significantly more efficient quantum circuits (see Section 4.6).

In the context of quantum optimization, the Lagrangian relaxation allows finding the optimal or ϵ-optimal solution to the primal problem through the process of repeated measurements of the specially prepared Lagrangian circuit (Section 4.6). The numerical experiments in Section 5.2.4 suggest that the optimal solution is often contained in the measured sample. Indeed, the optimal solution is measured frequently enough to outperform the traditional QUBO-based approaches.

Since our approach is based on adiabatic evolution in time, we may generalize the Lagrange multipliers λ_i to time-dependent functions $\lambda_i(t) \geq 0$ for for $i = 1, ..., m$ and $t \in [0, T]$. This allows enhanced control over the constraint terms during the adiabatic evolution. Hence, we now define the generalized time-dependent Lagrangian dual by

$$\min_{x \in \{0,1\}^n} x^T Q_0 x + \sum_{i=1}^{m} \lambda_i(t) \left(c_i - x^T Q_i x \right). \tag{4.6}$$

4.5 The algorithm

In this section, we use the Lagrangian relaxation in (4.5) or the time-dependent Lagrangian dual function (4.6) to construct an efficient Lagrangian DAQC circuit. As in Section 3.8.1, we recast (4.6) as a problem of finding the ground state of a Hamiltonian. The formulation in (4.6) can be readily represented as a Hamiltonian by substituting each variable x_j with a spin variable s_j using the following identity $x_j = (1 - s_j)/2$. Finally, each variables s_j is substituted with the Pauli-Z operator Z_j which has eigenvalues $+1$ and -1. This turns the function in (4.6) into a Hamiltonian which we denote as $H_C(t)$. Here, the problem Hamiltonian depends on time because $\lambda_i(t)$ depends on time.

We use DAQC to find the ground state of the $H_C(t)$. For this, we need an initial Hamiltonian H_{init} whose ground state is known and can be prepared in constant time. A typical example of H_{init} is the transverse-field operator introduced in (3.21). However, this time, the operator is multiplied by a negative one because we minimize, i.e. $H_{init} = -\sum_{i=1}^{n} X_i$. The adiabatic evolution is achieved by gradually mixing the initial and problem

Hamiltonians according to the relation

$$H(t) = (1 - s(t))H_{init} + s(t)H_C(t). \tag{4.7}$$

In this equation, $s(t) \in [0, 1]$ for $t \in [0, T]$ is an adiabatic schedule with the requirements $s(0) = 0$ and $s(T) = 1$. According to Theorem 1 the gradual transition from H_{init} to $H_C(t)$ guarantees that the system evolves arbitrarily close to the ground state of H_C at $t = T$ if the transition is long enough.

It is important to note that whenever (4.6) is linear, the objective function does not have quadratic terms of the form $x_i x_j$. From Observation 1 we deduce that the resulting circuit will not have 2-qubit gates RZZ and, consequently, it will not be able to create entanglement. From the discussion in Section 2.8 we know that entanglement is a necessary resource for quantum speedup. To introduce entanglement, we need to add some quadratic terms without modifying the formulation of the optimization problem. Clearly, any addition of variables $x_i x_j$ to (4.6) will change the optimization problem to another optimization problem that we are not interested in. Similarly, any addition of operators $Z_i Z_j$ to $H_C(t)$ will change the problem Hamiltonian that is not equivalent to the problem in (4.6). Our solution is to use a different initial Hamiltonian with quadratic terms that do not commute with the Z_i or $Z_i Z_j$ terms in $H_C(t)$. Specifically, we consider a *2-local* Hamiltonian – a Hamiltonian that can be written as a finite sum of terms acting upon at most two qubits each. In this study, we choose the 2-local $ZZXX$ universal Hamiltonian [8] which has the form:

$$\begin{aligned} H_{ZZXX} = &\sum_i h_i X_i + \sum_i \Delta_i Z_i \\ &+ \sum_{i,j} K_{i,j} X_i X_j + \sum_{i,j} J_{i,j} Z_i Z_j. \end{aligned} \tag{4.8}$$

We identify the terms $\sum_i h_i X_i$ and $\sum_{i,j} J_{i,j} X_i X_j$ with the new initial Hamiltonian whereas the rest of the terms are used to represent the problem Hamiltonian. Therefore, 2-qubit interactions are introduced through the use of the terms in $\sum_{i,j} K_{i,j} X_i X_j$ of the 2-local $ZZXX$ Hamiltonian. We hypothesize that a particular choice of the coefficients $K_{i,j}$ can introduce necessary correlations between qubits which could potentially yield better results. We choose a coupling strength coefficient $K_{i,j}$ such that the terms form a chain with a periodic boundary condition. Specifically, we let

$$H_{init} = -\sum_{i=1}^{n} X_i - \sum_{i=1}^{n} X_i X_{i+1} \tag{4.9}$$

where we define $n + 1 := 1$. As we will see later (Section 4.6) such a choice gives a highly parallelizable circuit with circuit depth independent of problem size n.

Note that if the problem in (4.6) is linear, the Lagrangian dual is also linear. It follows that H_C is a 1-local Hamiltonian and H_{init} in (4.9) is a 2-local Hamiltonian with a user provided qubit coupling. It is also possible to set $K_{i,j} = 0$ for all i, j. Then the Hamiltonian $H(t)$ becomes a 1-local Hamiltonian because both H_{init} and H_C are 1-local Hamiltonians. In this case, optimization of $H(t)$ is in the complexity class P. Therefore, our DAQC approach allows for efficient approximation of NP-complete linear programs and a user-controlled entanglement. Importantly, this allows one to build DAQC circuits which are extremely parallelizable. That is, the number of time steps is independent of the problem size n. We further investigate the structure of the DAQC circuit and its parallelizability in the following sections.

4.6 Lagrangian DAQC for linear problems

In this section, we construct an efficient quantum circuit which is unlike QUBO-derived circuits is substantially more parallelizable, does not require auxiliary qubits and only requires nearest neighbour qubit connectivity. Throughout this study, we will refer to the circuit as *Lagrangian circuit*.

Suppose we want to find the optimal solution to (4.2). We start the development of our circuit by approximating the problem in (4.2) using the time-dependent Lagrangian relaxation.

$$\min_{x \in \{0,1\}^n} q_0^t x + \sum_{i=1}^{m} \lambda_i(t) \left(c_i - q_i^T x \right) \tag{4.10}$$

If we let $\lambda_i(t) \equiv \lambda_i$ to be independent of time, then the function (4.10) is convex in λ_i. The optimal λ_i^* for a Lagrangian dual problem in (4.5) can be estimated to an arbitrary error $\epsilon > 0$ in $\mathcal{O}(\epsilon^{-2})$ iterations using the subgradient method [52].

Next, we recast (4.10) as a problem of finding the ground state of a Hamiltonian. By substituting binary variables x_j with spin variables s_j and introducing Pauli-Z operators Z_j we obtain the problem Hamiltonian $H_C(t)$ which is linear in Z_j. Dropping all constant terms in $H_C(t)$ yields

$$H_C(t) = \sum_{j=1}^{n} \left(-q_{0,j} + \sum_{i=1}^{m} \lambda_i(t) q_{i,j} \right) Z_j. \tag{4.11}$$

Combining $H_C(t)$ with the H_{init} in (4.9) yields the total Hamiltonian

$$H(t) = (1 - s(t)) \left(-\sum_{j=1}^{n} X_i - \sum_{j=1}^{n} X_j X_{j+1} \right) \\ + s(t) \sum_{j=1}^{n} \left(-q_{0,j} + \sum_{i=1}^{m} \lambda_i(t) q_{i,j} \right) Z_j, \ t \in [0, T], \tag{4.12}$$

where $s(t)$ is an adiabatic schedule. Let $|\psi(t)\rangle = U(t, 0) |\psi(0)\rangle$ denote a wavefunction at time t, such that $|\psi(0)\rangle = |+\rangle^{\otimes n}$ is the ground state of H_{init}. From Section 3.5, we know that the wavefunction $|\psi(t)\rangle$ is evolved according to the Schrödinger equation

$$i \frac{dU(t, 0)}{dt} = H(t) U(t, 0). \tag{4.13}$$

In order to construct the Lagrangian circuit, we follow the process outlined in Section 3.5. We subdivide the time interval $[0, T]$ into p subintervals of length $\Delta t = T/p$. Then the evolution operator satisfying (4.13) is given by

$$U(T, 0) = \lim_{p \to \infty} \prod_{k=0}^{p} \exp \{ -i \Delta t H(\Delta t k) \}. \tag{4.14}$$

To approximate (4.14), we choose p to be finite and apply the first order Trotterization. This yields a p-layered Lagrangian circuit

$$\hat{U}(T, 0) = \prod_{k=1}^{p} \exp \{ -i c_k H_{init} \} \exp \{ -i b_k H_C(\Delta t k) \}. \tag{4.15}$$

with H_{init} and $H_C(t)$ given in (4.9) and (4.11) respectively and c_k, b_k defined as in (3.10).

It was suggested in [50] to normalize the Hamiltonians H_{init} and H_C by their corresponding Frobenius norms. Therefore, we substitute c_k and b_k defined in (3.10) with

$$\gamma_k = \frac{c_k}{||H_{init}||}, \tag{4.16}$$

$$\beta_k = \frac{b_k}{||H_C||}. \tag{4.17}$$

Empirically, we verified that Frobenius normalization yields a well-performing schedule. However, the theoretical basis for this is not understood.

We now examine the structure of the resulting circuit $\hat{U}(T, 0)$ given in (4.15). For any layer $k \in \{1, ..., p\}$ consider the unitary matrix given by the initial Hamiltonian. Due to

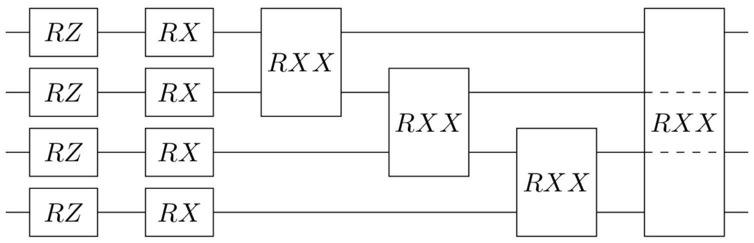


Figure 4.2: The k-th layer of the 4-qubit Lagrangian circuit. Rotation angles are not displayed for clarity.

commutativity of X_i and X_iX_j the matrix can be written as

$$\exp\{-i\gamma_k H_{init}\} = \prod_{j=1}^{n} \exp\{i\gamma_k X_j X_{j+1}\} \prod_{j=1}^{n} \exp\{i\gamma_k X_j\}. \tag{4.18}$$

The unitary matrix given by the problem Hamiltonian can be written as

$$\exp\{-i\beta_k H_C(\Delta tk)\} = \prod_{j=1}^{n} \exp\left\{-i\beta_k \left(-q_{0,j} + \sum_{i=1}^{m} \lambda_i(\Delta tk) q_{i,j}\right) Z_j\right\}. \tag{4.19}$$

We note that the Lagrange multiplier $\lambda_i(\Delta tk)$ in (4.19) contributes to the angle of the rotation and depends on the time step Δtk. We now recall the definitions of the 1-qubit and 2-qubit rotation gates in (2.32) and (2.40) respectively. Then, for any k, the unitary matrices in (4.19) can be represented by 1-qubit gates RZ. Conversly, the unitary matrices given in (4.18) can be represented by 1-qubit and 2-qubit gates RX and RXX respectively. The final Lagrangian circuit is as follows:

$$\hat{U}(T,0) = \prod_{k=1}^{p} \prod_{j=1}^{n} RXX_{j,j+1}(-2\gamma_k) \prod_{j=1}^{n} RX_j(-2\gamma_k) \prod_{j=1}^{n} RZ_j \left(2\beta_k \left(-q_{0,j} + \sum_{i=1}^{m} \lambda_i(\Delta tk) q_{i,j}\right)\right). \tag{4.20}$$

The structure of the kth layer of the $\hat{U}(T,0)$ is illustrated in Fig. 4.2. The resulting circuit is extremely parallelizable and hence has a short circuit depth. All RZ gates can be applied in a single time step. Similarly, all RX gates can be applied in a single time step. All RXX gates can be applied in 2 time steps if n is even and 3 time steps if n is odd. Hence, the circuit depth complexity is $\mathcal{O}(p)$ – for further discussion see Appendix E. That is, the circuit depth depends on the parameter p and does not depend on the problem size. Importantly, due to the choice of the coefficients $K_{i,j}$ in H_{init} (4.9), the resulting circuit only requires the nearest neighbour qubit connectivity. Finally, we note that both the formulation and the circuit construction did not involve any extra variables and auxiliary qubits.

4.6.1 Lagrangian multiplier scheduling

In this section we discuss the generalized Lagrange multipliers $\lambda_i(t)$ for $i = 1, ..., m$ and $t \in [0, T]$. First, we note that constant λ_i is a particular case of $\lambda_i(t)$. Most combinatorial problems have no notion of time. That is, the problem formulation and its solution are time-independent. Therefore, it makes sense to use Lagrangian multipliers that are also constant. However, when using the DAQC approach, the solution to a problem is gradually obtained during the adiabatic evolution. The inherent time-dependence of the adiabatic process can be used to control the strength of the Lagrange multipliers. For example, it is possible to delay the introduction of some difficult constraints allowing the process to start with an easier problem. Alternatively, the constraints whose violation is not critical can also be scheduled to appear towards the end of the evolution. In this study, we define the time-dependent Lagrangian multipliers in terms of the modified schedule in (3.14),

$$\lambda_i(t; o_i, \gamma_i) := \gamma_i \cdot s(t - o_i; a, T) \cdot \mathbb{1}_{o_i<t}(t) \quad \text{for} \quad i = 1, ..., m. \tag{4.21}$$

In the equation above, $\gamma_i > 0$ is the weight of the schedule, $o_i \in [-T, T]$ is the time offset, and $\mathbb{1}_{o_i<t}(t)$ is an indicator function such that $\mathbb{1}_{o_i<t}(t) = 1$ for $o_i < t$ and $\mathbb{1}_{o_i<t}(t) = 0$ otherwise. Setting $o_i = T/2$ introduces the constraint i in the middle of the adiabatic process. Whereas $o_i = T$ is equivalent to ignoring the constraint completely as $\lambda_i(t; o_i, \gamma_i) = 0$ for all $t \in [0, T]$.

4.7 Conclusion

In this chapter, we argue that the traditional approach of casting a constrained problem into a QUBO form is highly inefficient. We address this issue by introducing Lagrangian duality from classical optimization. The Lagrangian dual may not yield optimal solutions if solved classically. However, in the quantum optimization setting, the proximity of attained solution to the optimal solution allows sampling of the optimal solution with a high probability. Through the derivation process of the Lagrangian DAQC we demonstrate that the resulting circuit has problem-independent circuit depth, does not require auxiliary qubits and has favourable sparse connectivity that an end-user can control.

Chapter 5

Numerical experiments

This chapter presents a well-known linear problem similar to the general formulation in (4.2) and compares numerical results given by QUBO-based and Lagrangian circuits for that problem. We will see that the proposed Lagrangian circuit significantly outperforms the canonical QUBO-based circuit.

5.1 Knapsack problem

We will use a well-known 1D 0-1 Knapsack problem (KP) [30]. KP is an NP-hard linear problem with similar formulation as in (4.2) with $m = 1$. In combinatorial optimization, KP is a problem in which one must pack the most valuable items into a knapsack so that the total weight is less than or equal to the knapsack's capacity. The KP can be applied to a number of different industrial situations, such as resource allocation, scheduling, and inventory management.

We derive a KP from the general canonical formulation given in (4.2). Suppose we have a KP with n items. Let $v_j, w_j \in \mathbb{N}$ denote jth item value and weight respectively and $c \in \mathbb{N}$ is a knapsack weight bound. By setting the entries $q_{0,j} = -v_j$, $q_{1,j} = -w_j$ for all $j = 1, \ldots, n$ and $c_1 = -c$ we obtain the canonical formulation of the n-variable KP

$$\begin{aligned} &\max \sum_{j=1}^{n} v_j x_j \qquad (5.1) \\ &\sum_{j=1}^{n} w_j x_j \leq c \\ &x \in \{0,1\}^n. \end{aligned}$$

Our choice of the problem is based on several considerations. First, the KP is a typical representative of constrained binary linear programs that belong to the intersection NP-hard and NP-complete classes. Second, as we will show below, trying to solve the KP in the quantum setting by the means of adiabatic computation with the canonical QUBO reformulation (4.3) results in a prohibitively costly circuit which requires all-to-all connectivity and additional auxiliary qubits whose number grows logarithmically with the constraint bound c. Therefore, the KP is a good representative of CO problems that challenge currently available quantum optimization approaches. Developing a quantum protocol that successfully solves the KP means that the majority of other binary linear programs with inequality and equality constraints can also be successfully solved by the same protocol.

5.1.1 Lagrangian circuit

We start the development of our Lagrangian circuit by approximating the problem in (5.1) using the Lagrangian dual given below:

$$\min_{x\in\{0,1\}^n} -\sum_{j=1}^{n} v_j x_j + \lambda(t)\left(\sum_{j=1}^{n} w_j x_j - c\right). \tag{5.2}$$

As before, if we let $\lambda(t) \equiv \lambda$ to be independent of time, then the problem (4.10) is convex in λ. Note that (5.2) is a particular case of the general formulation in (4.10).

From the general Lagrangian circuit given in (4.20) it is straightforward to deduce the structure of the Lagrangian circuit for the KP; we set $m = 1$, $q_{0,j} = -v_j$ and $q_{1,j} = w_j$. This gives the Lagrangian circuit for the KP.

5.1.2 QUBO circuit

To derive a QUBO circuit for KP, we first reformulate the KP in (5.1) as a QUBO problem. For this we use the general QUBO formulation given in (4.3). In this case, we get:

$$\min_{x,y} -\sum_{j=1}^{n} v_j x_j + \gamma\left(\sum_{j=1}^{n} w_j x_j - W\right)^2 \tag{5.3}$$

$$\text{such that} \tag{5.4}$$

$$W = \sum_{k=0}^{\lfloor\log_2(c)\rfloor-1} 2^k y_k + (c+1-2^{\lfloor\log_2(c)\rfloor})y_{\lfloor\log_2(c)\rfloor}$$

$$y \in \{0,1\}^{\lfloor\log_2(c)\rfloor+1}$$

$$x \in \{0,1\}^n,$$

where γ is a penalty multiplier and c is the constraint inequality bound. We note that the number of additional auxiliary binary variables y_k grows as $\mathcal{O}(\log_2(c))$. Since $w_j > 0$ for $j = 1, \ldots, n$, from the discussion in Section 4.2.2, it immediately becomes clear that the squared penalty term introduces $\binom{n+\log_2(c)+1}{2}$ pairwise interactions between every qubit in each layer of a circuit. That is, each layer of the circuit will have $\binom{n+\log_2(c)+1}{2}$ 2-qubit RZZ gates. Therefore, each layer of a circuit is all-to-all connected. Due to this, the circuit is poorly parallelizable and has circuit depth $O\left(p(n+\log_2(c))\right)$ – for further information see Appendix E. Therefore, the circuit depth depends on the problem size n, the value of the constraint bound c and the number of layers p. For example, if the constraint bound $c \geq e^{2n}$, then the number of auxiliary qubits is at least double of the problem size n. This means even relatively small knapsack problems may be very challenging for quantum optimization.

One might attempt to decrease the number of auxiliary qubits by scaling down the coefficients w_j and the constraint bound c by dividing both sides of the inequality constraint in (5.1) by some constant $z \in \mathbb{N}$. However, this often leads to fractional values, i.e. $\sum_{j=1}^{n} w_j x_j / z \in \mathbb{Q} \setminus \mathbb{N}$ for some $x \in \{0,1\}^n$. This implies that the slack variable W must be able to approximate the rational $\sum_{j=1}^{n} w_j x_j / z$ for all feasible x. Hence, the binary expansion of W must include additional auxiliary binary variables that represent a fractional part. Therefore, in the presence of inequality constraints, it is practically impossible not to have auxiliary qubits.

5.2 Experimental setup

In this section, we discuss the computational setup used in the experiment, present procedures used to generate multiple KP datasets and various metrics used for analysis. We compare the performance of QUBO and Lagrangian circuits and demonstrate the superiority of the latter.

5.2.1 Hardware setup

The experiment features Lagrangian- and QUBO-based circuits, some of which have 100 layers, 20 qubits and all-to-all connectivity. Since there does not exist a quantum computer that can successfully execute such circuits, all quantum computation was simulated on classical hardware. This is a large-scale experiment which pushes quantum simulation to its limits. To compute the experimental results presented in this chapter, we used over 240 hours of 64-core, 32-core and 16-core CPUs and over 200 GB of RAM on the Google Compute Platform (GCP). Particularly demanding QUBO-based quantum circuits whose single execution required at least two weeks of CPU time on the GCP were executed on the Nvidia's A100 High-Performance Computing server. Interestingly, Nvidia A100, reduced the computation time from weeks to several hours.

In this experiment, more than 2000 optimization problems were specifically generated and solved using quantum simulation. The correctness of simulation results was verified with Google's classical optimization solvers: *OR-Tools Linear Optimization Solver* and *OR-Tools Brute Force Solver.*

5.2.2 Generation of random instances

We create two different types of *test* supersets of KPs:

1. The purpose of the Superset 1 is to examine performance scaling with respect to the problem size n. Thus, for each $n = 5, 6, \ldots, 15$ we generate 100 KP instances with integer coefficients v_i, w_i distributed according to the uniform probability distribution with bounds 1 and 10. That is $v_i, w_i \sim U(1, 10)$. The capacity bound c is a function of random variables w_i and it is defined as $c = \lfloor 1/2 \sum_i w_i \rfloor$.

2. The purpose of the Superset 2 is to examine performance scaling with respect to the scale of the coefficients v_i and w_i while n is fixed. For a fixed $n = 11$ and each $C = 10, 20, 30, \ldots, 100$ we generate 100 problem instances such that $v_i, w_i \sim U(1, C)$. Hence, there are 10 datasets each containing instances of size $n = 11$ but with coefficients v_i, w_i whose range is incremented by 10 units in each dataset.

Two additional *training* supersets of data were also created with the same setup as above. The *training* supersets were only used for finding the best hyperparameters p, T and a in (3.14). Training supersets were not used for producing any of the experimental results in following sections.

5.2.3 Performance Metrics

In order to gauge the performance of QUBO and Lagrangian circuits on the test datasets we will use the R_{99} and Time-To-Solution (TTS) metrics. Let a parameter sequence $\theta = \{(\gamma_k, \beta_k)\}_{k=1}^{p}$ defined in (4.16) be given. Then $R_{99}(\theta)$ is the number of shots (circuit measurements) that must be performed to ensure a 99% probability of observing the ground state of H_C under the parametrization θ - for further details see Appendix D. It is defined as

$$R_{99}(\theta) := \frac{\log(0.01)}{\log(1 - P(\theta))}, \tag{5.5}$$

where $P(\theta)$ is the probability of measuring the optimal solution under parametrization θ. The TTS denotes the expected computation time required to find an optimal solution for a particular problem instance with 99% confidence. It is defined via

$$\mathrm{TTS}(\theta) = R_{99}(\theta) \cdot t_{ss} \tag{5.6}$$

with t_{ss} denoting the necessary runtime for a single shot. In order to approximate t_{ss}, we assume the following holds:

1. A quantum processor performs any single-qubit and two-qubit gate operations in 10 and 20 nanoseconds, respectively.

2. Gate operations may be performed simultaneously if they do not act on the same qubit.

3. All components of the circuit are noise-free and there is no overhead for quantum error correction or fault-tolerant quantum computation.

Then the p-layered Lagrangian circuit has the following runtime dependence on p (Appendix E):

$$t_{ss} = \begin{cases} 50p & n \text{ is even,} \\ 70p & n \text{ is odd.} \end{cases} \tag{5.7}$$

The p-layered QUBO circuit has the runtime dependence on p, problem size n and the constraint bound c (Appendix E)

$$t_{ss} = \begin{cases} 20p\,(n + \log_2(c)) & n + \log_2(c) \text{ is even,} \\ 20p\,(n + \log_2(c) + 1) & n + \log_2(c) \text{ is odd.} \end{cases} \tag{5.8}$$

5.2.4 Experiment results

This section presents TTS scaling results for the Lagrangian and QUBO circuits. For the test Superset 1 the performance advantage of the Lagrangian circuit increases with the size of the problem (number of items). For each problem size varying from 5 to 15 items, Fig 5.1 shows the plot of median TTS, suggesting that the Lagrangian circuit has significantly better scaling.

Additionally, we examine the amount of optimal resources required for both circuit types to have comparable R_{99} across all problem sizes. The numerical experiments demonstrate that the Lagrangian circuit requires quadratically fewer resources to have a comparable R_{99} with the QUBO circuit. Fig 5.2 demonstrates the scaling of the optimal mean number of layers p and evolution time T with respect to the problem size. From the figure, it is clear that the Lagrangian circuit has a linear scaling, whereas the QUBO circuit has quadratic scaling in both p and T. This difference becomes even more critical if we take into account the overhead costs associated with *Quantum Error Correction* (QEC) and the realization of *Fault-Tolerant Quantum Computation* (FTQC) schemes that become necessary for noisy circuits. Deeper circuits are much more vulnerable to noise necessitating greater QEC and FTQC overhead costs, which have been neglected in our analysis since our aim is to compare optimistic lower bounds for the two approaches.

Recall that whenever a combinatorial problem has inequality coefficients, additional qubits are necessary in order to account for slack variables. We stated that the number of additional qubits grows logarithmically with respect to a constraint bound. Using Superset 2, we demonstrate the performance scaling on fixed size KP datasets ($n = 11$) but with an increasing range of coefficients v_i, w_i. Fig 5.3 shows R_{99} scaling for both types of circuits with each having a fixed parametrization and number of layers across all datasets. It is clear that the Lagrangian circuit's R_{99} is not affected by the coefficient values and remains

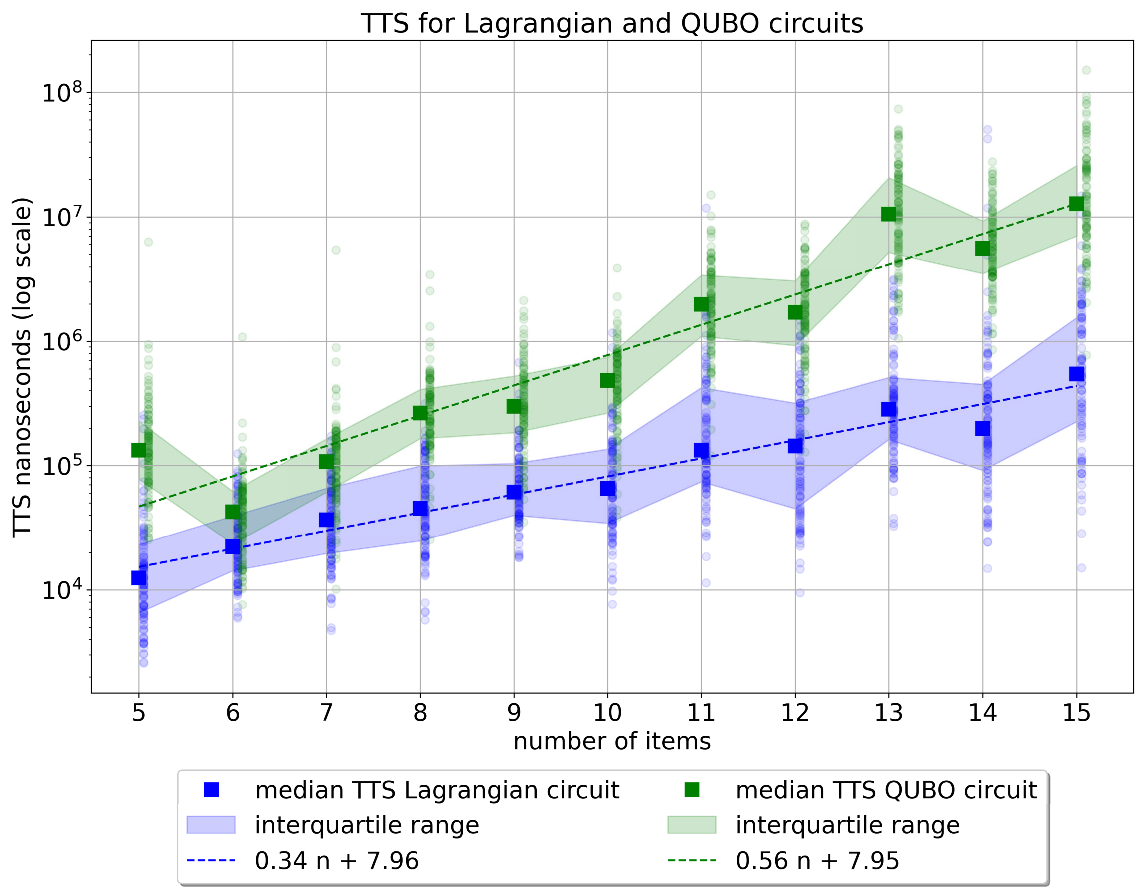


Figure 5.1: Scaling of the Lagrangian and QUBO circuit with respect to the size of a KP problem. For each dataset of size n a single pre-tuned scheduling function is used.

constant across all datasets of the Superset 2. The QUBO circuit's R_{99} has quadratically logarithmic growth. The scaling behaviour can be explained by the logarithmic increase in qubits for the QUBO circuit.

This brings us to an important point about the TTS scaling of QUBO and Lagrangian circuits. One might note that the TTS difference in Fig 5.1 is only of one order of magnitude at $n = 15$. While the difference is expected to get larger as n increases, because of the quadratically logarithmic scaling of R_{99} with the size of w_i presented in Fig 5.3 we can conclude that the TTS difference can be made arbitrarily large by simply choosing KP problems with larger weights w_i. For example, if we were to compute the TTS difference at $n = 15$ and a slightly more practical item's weight distribution $w_i \sim U(1, 1000)$, then the median TTS difference would be at least two orders of magnitude. However, even with $n = 15$ and $w_i \sim U(1, 10)$ we already encounter computational limits of simulating the results for the QUBO-based circuits.

5.3 Conclusion

In this chapter, we performed a numerical benchmark study that suggests that the proposed Lagrangian circuits have a quadratic improvement over QUBO-based circuits in terms of evolution time and circuit depth. In addition to the economic structure, the Lagrangian approach has an order of magnitude better TTS for larger-size problems. The difference in TTS can be drastically amplified if the problem size and constraint bounds become larger.

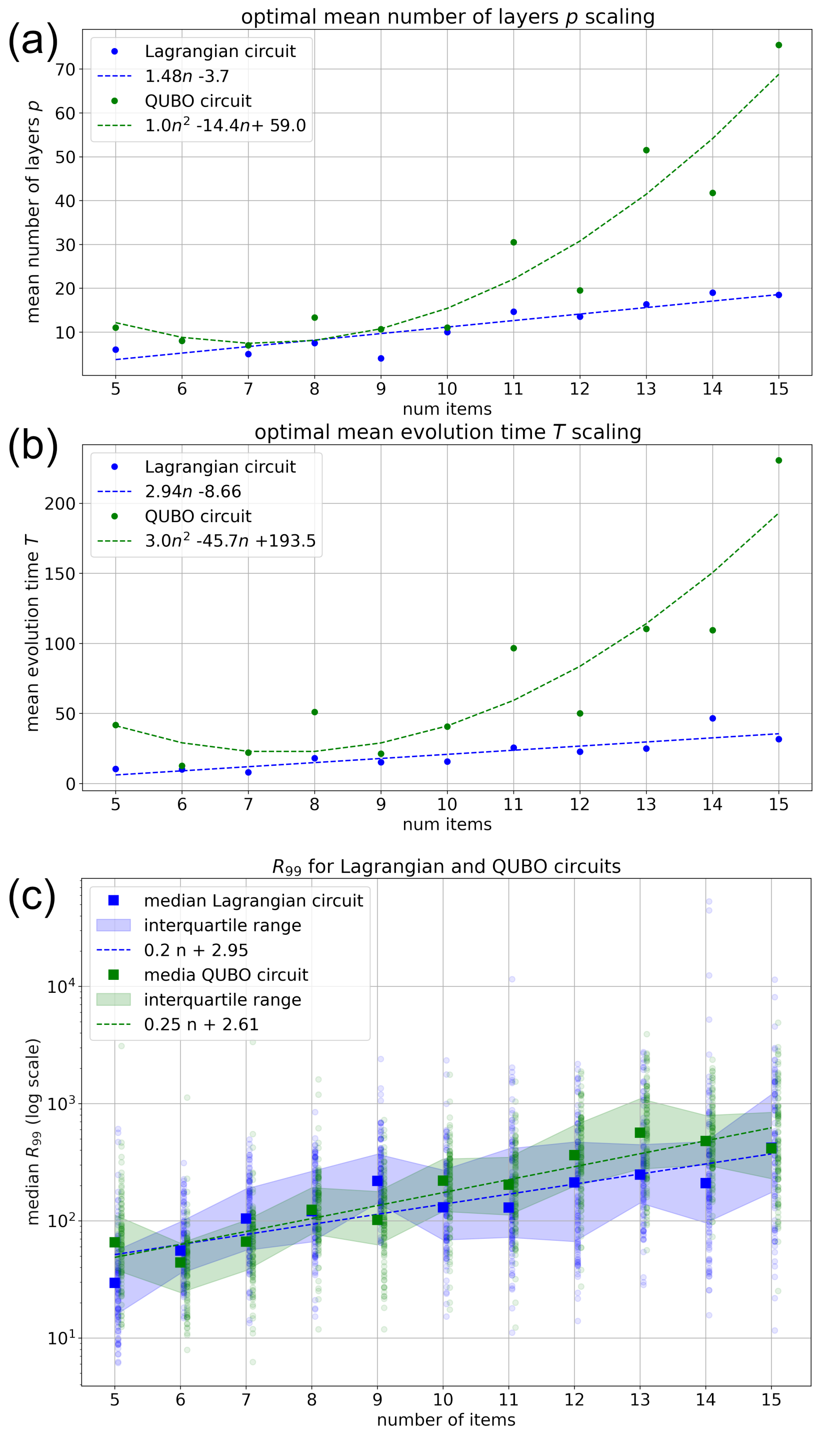
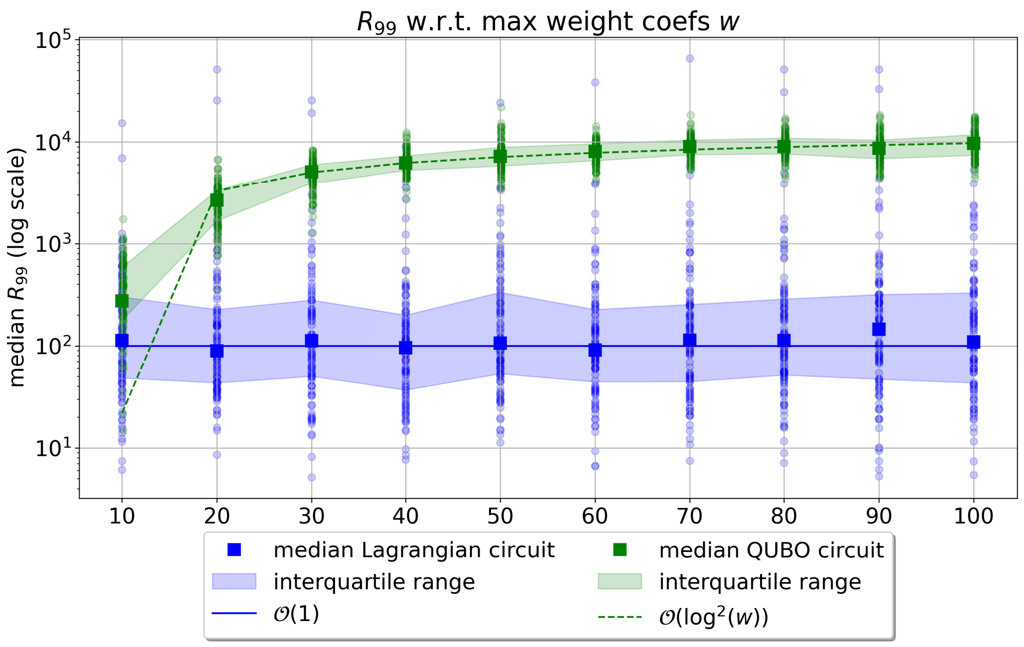


Figure 5.2: Scaling of the resources of the Lagrangian and QUBO circuits with respect to a problem size. (a) The optimal mean depth p scales linearly for the Lagrangian circuit, whereas the QUBO circuit has a quadratic growth. (b) Optimal mean evolution time T also scales linearly for the Lagrangian circuit. The QUBO circuit has quadratic growth. (c) R_{99} of both circuits under their respective optimal parameters p and T. Given optimal parameters, both circuits have a comparable R_{99} with exponential scaling.



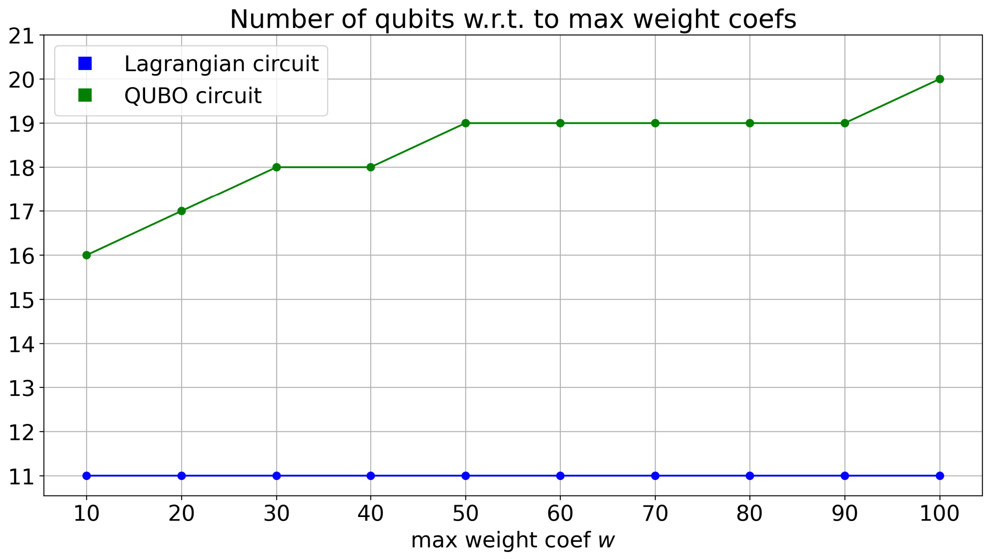


Figure 5.3: Scaling of the Lagrangian and QUBO circuits with respect to the coefficients of a KP problem. For all datasets, each circuit has a single fixed parametrization, that is p, T and $s(t)$ are fixed across all datasets.

Chapter 6

Conclusions and future work

In this thesis, we introduced fundamental concepts of quantum mechanics and presented the complete mathematical construction of QAOA accompanied by a detailed example of a QAOA circuit for the MaxCut problem. Additionally, we discussed possible parametrization techniques such as the variational principle and the principle of discretized adiabatic quantum computation. Later on, we introduced optimization concepts such as linear integer programs and their reformulation into QUBO problems. We argued that the traditional method of converting linear integer programs into QUBO caused many issues from optimization and quantum resource efficiency perspectives. Finally, we proposed a novel approach based on the theory of Lagrangian duality. We analytically demonstrated that the resulting quantum algorithm was significantly more efficient than the traditional QUBO-based QAOA. Finally, we performed a large-scale quantum simulation experiment where we numerically verified that the proposed approach was notably superior in terms of time to solution and quantum resource requirements.

In future work, we plan to investigate the behaviour of the proposed initial Hamiltonian with quadratic terms X_iX_j under different parametrization of the coefficients $K_{i,j}$ in (4.8). It is also interesting to understand and quantify the amount of entanglement introduced by the exponential of the new initial Hamiltonian and how it affects the time to solution. Another aspect we aim to investigate is the performance of the proposed method in the presence of *quantum decoherence* (which is also called *noise*) caused by an external environment. Additionally, in this thesis, we examined the applicability of the DAQC with Lagrangian duality theory to linear integer programs. However, the approach generalizes to integer programs with a quadratic objective function and linear inequality constraints. Therefore, it is interesting to see how the proposed algorithm will fair with the aforementioned problems.

Bibliography

[1] T. Albash and D Lidar. Adiabatic quantum computation. *RMP*, 90(1):015002, 2018.

[2] A. Ambainis and O. Regev. An elementary proof of the quantum adiabatic theorem. *arXiv preprint quant-ph/0411152*, 2004.

[3] B. Apolloni, C. Carvalho, and D. De Falco. Quantum stochastic optimization. *Stoch. Process. their Appl.*, 33(2):233–244, 1989.

[4] M. Aramon, G. Rosenberg, E. Valiante, T. Miyazawa, H. Tamura, and H. Katzgraber. Physics-inspired optimization for quadratic unconstrained problems using a digital annealer. *Front. Phys.*, 7:48, 2019.

[5] J. Basdevant. *Variational Principles in Physics.* Springer Science & Business Media, 2006.

[6] A. Beck. *Introduction to Nonlinear Optimization: Theory, Algorithms, and Applications with MATLAB.* SIAM, 2014.

[7] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *JMLR*, 13(2), 2012.

[8] J. Biamonte and P. Love. Realizable Hamiltonians for universal adiabatic quantum computers. *Phys. Rev. A*, 78(1):012352, 2008.

[9] S. Boixo, T. Rønnow, S. Isakov, Z. Wang, D. Wecker, D. Lidar, J. Martinis, and M. Troyer. Evidence for quantum annealing with more than one hundred qubits. *Nat. Phys.*, 10(3):218–224, 2014.

[10] S. Brush. History of the Lenz-Ising model. *RMP*, 39(4):883, 1967.

[11] J. Cui, Y. Xiong, S. Ng, and L. Hanzo. Quantum approximate optimization algorithm based maximum likelihood detection. *IEEE Trans Commun*, 2022.

[12] J. Devore. *Probability and Statistics for Engineering and the Sciences.* Cengage learning, 2011.

[13] E. Farhi, J. Goldstone, and S. Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

[14] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser. Quantum computation by adiabatic evolution. *arXiv preprint quant-ph/0001106*, 2000.

[15] R. Feynman et al. Simulating physics with computers. *Int. J. Theor. Phys.*, 21(6/7), 1982.

[16] O. Galindo and V. Kreinovich. What is the optimal annealing schedule in quantum annealing. In *2020 IEEE Symposium Series On Computational Intelligence (SSCI)*, pages 963–967. IEEE, 2020.

[17] M. Garey, D. Johnson, and L. Stockmeyer. Some simplified NP-complete problems. In *Proceedings of the sixth annual ACM symposium on Theory of computing*, pages 47–63, 1974.

[18] A. Geoffrion. Lagrangean relaxation for integer programming. In *Approaches to integer programming*, pages 82–114. Springer, 1974.

[19] A. Gilyén, S. Arunachalam, and N. Wiebe. Optimizing quantum optimization algorithms via faster quantum gradient computation. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1425–1444. SIAM, 2019.

[20] F. Glover, J.K. Hao, and G. Kochenberger. Polynomial unconstrained binary optimisation-part 2. *Int. J. Metaheuristics*, 1(4):317–354, 2011.

[21] H. Goto, K. Tatsumura, and A. Dixon. Combinatorial optimization by simulating adiabatic bifurcations in nonlinear Hamiltonian systems. *Sci. Adv.*, 5(4):eaav2372, 2019.

[22] D. Griffiths. *Introduction to Quantum Mechanics.* Pearson International Edition (Pearson Prentice Hall, Upper Saddle River, 2005), 1962.

[23] H. Haber. The time evolution operator as a time-ordered exponential. 2018.

[24] I. Hen and F. Spedalieri. Quantum annealing for constrained optimization. *Phys. Rev. Applied*, 5(3):034007, 2016.

[25] T. Inagaki, Y. Haribara, K. Igarashi, T. Sonobe, S. Tamate, T. Honjo, A. Marandi, P. McMahon, T. Umeki, K. Enbutsu, et al. A coherent Ising machine for 2000-node optimization problems. *Science*, 354(6312):603–606, 2016.

[26] R. Jozsa. Entanglement and quantum computation. *arXiv preprint quant-ph/9707034*, 1997.

[27] S. Karimi and P. Ronagh. A subgradient approach for constrained binary optimization via quantum adiabatic evolution. *Quantum Inf. Process.*, 16(8):1–21, 2017.

[28] T. Kato. On the adiabatic theorem of quantum mechanics. *JPSJ*, 5(6):435–439, 1950.

[29] P. Kaye, R. Laflamme, and M. Mosca. *An Introduction to Quantum Computing.* OUP Oxford, 2006.

[30] H. Kellerer, U. Pferschy, and D. Pisinger. Introduction to NP-Completeness of knapsack problems. In *Knapsack problems*, pages 483–493. Springer, 2004.

[31] G. Kluber. Trotterization in QM Theory. `https://web.ma.utexas.edu`, 2020.

[32] M. Laforest. *The Mathematics of Quantum Mechanics.* 2015.

[33] M. Lewis and F. Glover. Quadratic unconstrained binary optimization problem preprocessing: Theory and empirical analysis. *Networks*, 70(2):79–97, 2017.

[34] L. Li, M. Fan, M. Coram, P. Riley, S. Leichenauer, et al. Quantum optimization with a novel Gibbs objective function and ansatz architecture search. *Phys. Rev. Res.*, 2(2):023074, 2020.

[35] A. Lucas. Ising formulations of many NP problems. *Front. Phys.*, page 5, 2014.

[36] S. Matsuura, S. Buck, V. Senicourt, and A. Zaribafiyan. Variationally scheduled quantum simulation. *Phys. Rev. A*, 103(5):052435, 2021.

[37] S. Matsuura, T. Yamazaki, V. Senicourt, L. Huntington, and A. Zaribafiyan. Vanqver: the variational and adiabatically navigated quantum eigensolver. *New J. Phys.*, 22(5):053023, 2020.

[38] G. B. Mbeng, R. Fazio, and G. Santoro. Quantum annealing: A journey through digitalization, control, and hybrid quantum variational schemes. *arXiv preprint arXiv:1906.08948*, 2019.

[39] J. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik. The theory of variational hybrid quantum-classical algorithms. *New J. Phys.*, 18(2):023023, 2016.

[40] A. Mishra, T. Albash, and D. Lidar. Finite temperature quantum annealing solving exponentially small gap problem with non-monotonic success probability. *Nat. Commun.*, 9(1):1–8, 2018.

[41] A. Montanaro. Quantum algorithms: an overview. *Npj Quantum Inf.*, 2(1):1–8, 2016.

[42] G. Moore et al. Cramming more components onto integrated circuits, 1965.

[43] S. Morita and H. Nishimori. Mathematical foundation of quantum annealing. *J. Math. Phys.*, 49(12):125210, 2008.

[44] H. Neidhardt, A. Stephan, and V. Zagrebnov. Operator-norm convergence of the Trotter product formula on Hilbert and Banach spaces: A short survey. In *Current Research in Nonlinear Analysis*, pages 229–247. Springer, 2018.

[45] M. Nielsen and I. Chuang. *Quantum Computation and Quantum Information.* American Association of Physics Teachers, 2002.

[46] A. Peruzzo, J. McClean, P. Shadbolt, M. Yung, X. Zhou, P. Love, and Jeremy L. Aspuru-Guzik, A. A variational eigenvalue solver on a photonic quantum processor. *Nat. Commun.*, 5(1):1–7, 2014.

[47] J. Roland and N. Cerf. Quantum search by local adiabatic evolution. *Phys. Rev. A*, 65(4):042308, 2002.

[48] P. Ronagh, B. Woods, and E. Iranmanesh. Solving constrained quadratic binary problems via quantum adiabatic evolution. *Quantum Inf. Comput.*, 16(11-12):1029–1047, 2016.

[49] W. Rudin et al. *Principles of Mathematical Analysis*, volume 3. McGraw-hill New York, 1976.

[50] K. Sankar, A. Scherer, S. Kako, S. Reifenstein, N. Ghadermarzy, W. Krayenhoff, Y. Inui, E. Ng, T. Onodera, P. Ronagh, et al. Benchmark study of quantum algorithms for combinatorial optimization: unitary versus dissipative. *arXiv preprint arXiv:2105.03528*, 2021.

[51] Y. Seki and H. Nishimori. Quantum annealing with antiferromagnetic fluctuations. *Phys. Rev. E*, 85(5):051112, 2012.

[52] N. Z. Shor. *Minimization methods for non-differentiable functions*, volume 3. Springer Science & Business Media, 2012.

[53] L. Susskind and A. Friedman. *Quantum Mechanics: The Theoretical Minimum.* Basic Books, 2014.

[54] A. Uvarov and J. Biamonte. On barren plateaus and cost function locality in variational quantum algorithms. *J. Phys. A Math.*, 54(24):245301, 2021.

[55] R. Wiersema, C. Zhou, Y. de Sereville, J. F. Carrasquilla, Y. B. Kim, and H. Yuen. Exploring entanglement and optimization within the Hamiltonian variational ansatz. *PRX Quantum*, 1(2):020319, 2020.

Appendix A

Trotter approximation

Whenever Hermitian matrices A and B do not commute, the first order Trotter approximation of $\exp\{i(A+B)t\}$ is obtained by setting $n = 1$ in the Trotter product formula (2.53). We will show that when $n = 1$, the error grows as $O(t^2)$ for small t. We will also show that the error term is zero when A and B commute.

Let A and B non-commutative Hermitian matrices in $\mathbb{C}^{n\times n}$. We first approximate all three exponential matrices.

$$e^{\mathrm{i}(A+B)t} = I + \mathrm{i}(A+B)t + O(t^2), \tag{A.1}$$

$$e^{\mathrm{i}At} = I + \mathrm{i}At + O(t^2), \tag{A.2}$$

$$e^{\mathrm{i}Bt} = I + \mathrm{i}Bt + O(t^2). \tag{A.3}$$

Then multiplying (A.2) and (A.3) yields

$$\begin{aligned} e^{\mathrm{i}At}e^{iBt} &= \Big(I + \mathrm{i}At + O(t^2)\Big)\Big(I + \mathrm{i}Bt + O(t^2)\Big) \\ &= I + \mathrm{i}(A+B)t + O(t^2). \end{aligned} \tag{A.4}$$

Subtracting (A.4) from (A.1) gives the result

$$e^{\mathrm{i}(A+B)t} = e^{\mathrm{i}At}e^{\mathrm{i}Bt} + O(t^2). \tag{A.5}$$

We now show that the error term is zero when A and B commute. We give a derivation similar to the derivation in [31]. Using the binomial theorem we can write

$$e^{\mathrm{i}(A+B)t} = \sum_{n=0}^{\infty} \frac{(\mathrm{i}(A+B)t)^n}{n!} \tag{A.6}$$

$$= \sum_{n=0}^{\infty} \frac{1}{n!} \sum_{k=0}^{n} \binom{n}{k} (iAt)^{n-k}(iBt)^k \tag{A.7}$$

$$= \sum_{n=0}^{\infty} \sum_{k=0}^{n} \frac{1}{n!} \frac{n!}{k!(n-k)!} (iAt)^{n-k}(iBt)^k. \tag{A.8}$$

Since $||(A+B)^n|| \leq ||A+B||^n$ the series (A.6)-(A.8) are absolutely convergent, and every rearrangement of (A.8) converges to the same limit [49, Chapter 3]. Therefore, we can rearrange the terms containing $(iAt)^{n-k}(iBt)^k$ such that they appear in the sum according to their exponent value. Thus we can write

$$\sum_{n=0}^{\infty}\sum_{k=0}^{n}\frac{1}{k!(n-k)!}(iAt)^{n-k}(iBt)^k = \sum_{p=0}^{\infty}\sum_{m=0}^{\infty}\frac{1}{p!m!}(iAt)^m(iBt)^p \tag{A.9}$$

$$= \left(\sum_{m=0}^{\infty}\frac{(iAt)^m}{m!}\right)\left(\sum_{p=0}^{\infty}\frac{(iBt)^p}{p!}\right) \tag{A.10}$$

$$= e^{iAt}e^{iBt}. \tag{A.11}$$

Therefore,

$$e^{\mathrm{i}(A+B)t} = e^{iAt}e^{iBt}. \tag{A.12}$$

This shows that the error term is zero whenever A and B commute.

Appendix B

Derivation of the solution to the time dependent Schrödinger equation

The time evolution of a state vector in the $(\mathbb{C}^2)^{\otimes n}$ is governed by the Schrodinger equation,

$$\mathrm{i}\hbar \frac{d}{dt} |\psi(t)\rangle = H(t) |\psi(t)\rangle , \tag{B.1}$$

where $H(t)$ is the Hamiltonian operator on $(\mathbb{C}^2)^{\otimes n}$ and $\hbar$ is a real constant. The solution to the Schrödinger equation can be written in terms of evolution matrix $U(t, t_0)$

$$|\psi(t)\rangle = U(t, t_0) |\psi(t_0)\rangle , \tag{B.2}$$

where $|\psi(t_0)\rangle$ is the initial condition for the time evolution of $|\psi(t)\rangle$. Given $H(t)$ our goal is to find $U(t, t_0)$ that evolves $|\psi(t_0)\rangle$ into $|\psi(t)\rangle$. In order to derive $U(t, t_0)$ we use the lecture notes from Santa Cruz Institute of Particle Physics [23].

Since $|\psi(t_0)\rangle$ is arbitrary, it follows that $U(t, t_0)$ must satisfy,

$$\mathrm{i}\hbar \frac{d}{dt} U(t, t_0) = H(t) U(t, t_0), \tag{B.3}$$

subject to the initial condition, $U(t_0, t_0) = I$. The goal is to solve (B.3) for $U(t, t_0)$. Using Taylor expansion we can write

$$U(t + \Delta t, t_0) = U(t, t_0) + \Delta t \frac{dU}{dt}(t, t_0) + O(\Delta t^2). \tag{B.4}$$

Using (B.3), it follows

$$U(t + \Delta t, t_0) = U(t, t_0) - \frac{\mathrm{i}}{\hbar} H(t) U(t, t_0) \Delta t + O(\Delta t^2). \tag{B.5}$$

Set $t_0 = t$. Since $U(t,t) = I$ we obtain

$$U(t+\Delta t, t) = I - \frac{\mathrm{i}}{\hbar} H(t)\Delta t + O(\Delta t^2). \tag{B.6}$$

The above is the Taylor expansion of $\exp\{\mathrm{i}H(t)\Delta t/\hbar\}$. Hence,

$$U(t+\Delta t, t) = e^{-\mathrm{i}H(t)\Delta t/\hbar} + O(\Delta t^2). \tag{B.7}$$

To obtain the explicit formula for $U(t,t_0)$ we divide the time interval $[t_0, t]$ into p equal subintervals of length

$$\Delta t = \frac{t - t_0}{p}. \tag{B.8}$$

Noting that for $t_0 < t_1 < t_2$ we have $U(t_2, t_0) = U(t_2, t_1)U(t_1, t_0)$, we can express $U(t, t_0)$ as follows.

$$U(t, t_0) = \prod_{k=1}^{p} U(t_0 + k\Delta t, t_0 + (k-1)\Delta t). \tag{B.9}$$

For each $k = 1, 2, ..., p$ we use (B.7)

$$U\left(t_0 + k\Delta t, t_0 + (k-1)\Delta t\right) = \exp\left\{-\mathrm{i}H(t_0 + (k-1)\Delta t)\Delta t/\hbar\right\} + O(\Delta t^2). \tag{B.10}$$

Therefore, if we take the limit of $p \to \infty$ and set $t_0 = 0$ we get

$$U(t, 0) = \lim_{p\to\infty} \prod_{k=1}^{p} \exp\left\{-\mathrm{i}H((k-1)\Delta t)\Delta t/\hbar\right\} \tag{B.11}$$

$$= \lim_{p\to\infty} \prod_{k=0}^{p} \exp\left\{-\mathrm{i}H(k\Delta t)\Delta t/\hbar\right\}. \tag{B.12}$$

Appendix C

On commutativity of an initial and problem Hamiltonians

We claimed that non-commutativity of the initial Hamiltonian H_{init} and the problem Hamiltonian H_C is a necessary condition for the algorithm to not fail. We now give a mathematical treatment of this claim.

Recall that H_{init} and H_C are $2^n \times 2^n$ Hermitian matrices and hence diagonalizable. Since H_{init} and H_C commute, they are simultaneously diagonalizable. That is, there exists an invertable matrix Q s.t.

$$H_{init} = QD_{init}Q^* \text{ and } \; H_C = QD_CQ^*, \tag{C.1}$$

where D_{init} and D_C are diagonal matrices. Since H_C is a diagonal matrix we conclude that Q is an identity matrix $Q = I$. Hence, $D_{init} = H_{init}$.

Without loss of generality let $e^{init} = (1, ..., 0)^T \in \mathbb{R}^{2^n}$ be the ground state of H_{init} with an associated lowest eigenvalue a_0 and let $e^* = (0, 1, ..., 0)^T \in \mathbb{R}^{2^n}$ be the ground state of H_C with an associated lowest eigenvalue b_0. We assume that the initial state e^{init} is not the ground state of H_C, otherwise the optimization problem is solved. Hence $H_Ce^{init} = b_1e^{init}$ such that $b_1 > b_0$.

From (4.14) we know that the unitary matrix that evolves the initial ground state of H_{init} to the ground state of H_C has the following form

$$\begin{aligned} U(t,0) &= \lim_{p\to\infty} \prod_{k=1}^{p} \exp\left\{-\frac{\mathrm{i}\Delta t}{\hbar} H(k\Delta t)\right\} && \text{(C.2)}\\ &= \lim_{p\to\infty} \prod_{k=1}^{p} \exp\left\{-\frac{\mathrm{i}\Delta t}{\hbar} \left[(1 - s(k\Delta t))H_{init} + s(k\Delta t)H_C\right]\right\}. && \text{(C.3)}\end{aligned}$$

Since H_{init} and H_C commute we can write

$$U(t,0) = \lim_{p\to\infty} \prod_{k=0}^{p} \exp\left\{-\frac{\mathrm{i}\Delta t}{\hbar}(1 - s(k\Delta t))H_{init}\right\} \exp\left\{-\frac{\mathrm{i}\Delta t}{\hbar} s(k\Delta t)H_C\right\}. \tag{C.4}$$

We recall that the exponent of a diagonal matrix is a diagonal matrix and the product of diagonal matrices is diagonal. Therefore $U(t,0)$ is a diagonal matrix. It follows that

$$U(t,0)e^{init} = ze^{init} \circeq e^{init}, \tag{C.5}$$

where $z \in \mathbb{C}$ is a global phase which we can ignore. We could stop here as it is clear that $U(t,0)$ fails to evolve e^{init} to e^* for all $t \in \mathbb{R}_+$. However, we would like to put the above in the context of the minimum eigengap g_{min} which we claimed to be zero. For clarity, let us denote the adiabatic schedule $s(t)$ as s. Recalling that H_{init} and H_C are diagonal we can write

$$H(t)U(t,0)e^{init} \circeq H(t)e^{init} = ((1-s)H_{init} + sH_C)\, e^{init} \tag{C.6}$$

$$= \begin{pmatrix} (1-s)a_0 + sb_1 & & \\ & (1-s)a_1 + sb_0 & \\ & & \ddots \end{pmatrix} e^{init} \tag{C.7}$$

$$= ((1-s)a_0 + sb_1)e^{init}. \tag{C.8}$$

Note that if $s = 0$ we get the ground energy a_0 of H_{init}. When $s = 1$ we get b_1 which is greater than the ground energy b_0 of H_C. Since $a_0 < a_1$ and $b_0 < b_1$ the following equation has a solution for some $s \in (0,1)$

$$(1-s)a_0 + sb_1 = (1-s)a_1 + sb_0. \tag{C.9}$$

This means that the spectral gap λ between eigenvalues $(1-s)a_0 + sb_1$ and $(1-s)a_1 + sb_0$ of $H(s)$ is zero for some $s \in (0,1)$. Therefore, we conclude that g_{min} is zero. The evolution of energy levels of the Hamiltonian $H(s)$ is illustrated in Fig C.1.

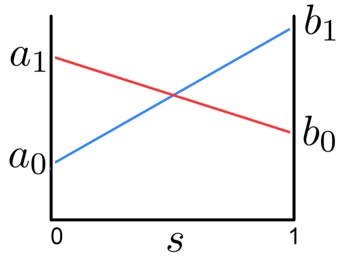


Figure C.1: The values of two eigenvalues (red and green lines) of $H(s)$ for all $s \in [0,1]$. Since H_{init} commutes with H_C we see that the spectral gap closes during the transition from H_{init} to H_C.

Appendix D

Time to solution

The objective of numerical benchmarking is to quantify how the computational effort of finding a solution to a problem scales with the size of the problem. Since DAQC is stochastic, a common approach to analyze the scaling is to measure the total time required for the algorithm to find the optimal solution at least once with a probability of 0.99. Measuring the total time for different size problems gives understanding of performance scaling. To compute TTS we first discuss R_{99}. As it was stated before, R_{99} is the number of runs that must be performed to ensure a 0.99 probability of observing the ground state of a problem Hamiltonian. To derive the R_{99} formula (5.5), we consider each run as a trail. The success of measuring the optimal state after each run is given by the probability p. Since trials are identical and independent with only two possible outcomes we conclude that it is a binomial experiment. Hence, the number of successful trials is a random variable X that has a binomial distribution. We would like to know how many trials (R_{99}) it takes so that

$$\Pr(X \geq 1) = 0.99. \tag{D.1}$$

Equivalently we can write

$$\begin{aligned} \Pr(X \geq 1) &= 1 - \Pr(X = 0) && \text{(D.2)} \\ &= 1 - \binom{R_{99}}{0} p^0 (1-p)^{R_{99}} && \text{(D.3)} \\ &= 1 - R_{99}(1-p)^{R_{99}}. && \text{(D.4)} \end{aligned}$$

Solving for R_{99} yields

$$R_{99} = \frac{\log(0.01)}{\log(1-p)}. \tag{D.5}$$

In the case when $p = 1$, we set $R_{99} = 1$. Consequently TTS is given by

$$\text{TTS} = t_{ss} \cdot R_{99}. \tag{D.6}$$

where t_{ss} is the runtime of a single trial.

Appendix E

DAQC circuit runtime complexities

Recall that one-qubit gates acting on different qubits can be executed in parallel, whereas any pair of two-qubit gates sharing at least one qubit can only be executed sequentially. As a result, highly parallelizable circuits have shorter runtime and are less prone to accumulation of errors. This section demonstrates the runtime complexity of the Lagrangian- and QUBO-based circuits for a KP. In Section 5.2.3 we stated that the Lagrangian circuit's runtime t_{ss} is $O(p)$ where p is the number of layers. In contrast, QUBO circuit has runtime complexity $O\left((n + \log_2(c))p\right)$ where n is the number of variables in a KP and c is the constraint bound.

Let us first look at the Lagrangian circuit given in (4.15)-(4.19). To show that the runtime t_{ss} is $O(p)$ it is sufficient to examine any single layer of the circuit illustrated in Fig 4.2. The kth layer has the following form.

$$\prod_{j=1}^{n} \exp\left\{\mathrm{i}\gamma_k X_j X_{j+1}\right\} \prod_{j=1}^{n} \exp\left\{\mathrm{i}\gamma_k X_j\right\} \prod_{j=1}^{n} \exp\left\{-\mathrm{i}\beta_k \left(v_j - \lambda(\Delta tk) w_j\right) Z_j\right\}. \tag{E.1}$$

We claim that the runtime of a single layer k is constant. Intuitively, we can rearrange the gates of the kth layer into at most five sublayers such that gates belonging to a sublayer can be applied in parallel. Each sublayer has the runtime $O(1)$. Fig E.1 illustrates a 4-qubit kth layer of a Lagrangian circuit and its equivalent rearrangement into four sublayers that can be applied consecutively. For all $j = 1, \ldots, n$ the unitary matrix $\exp\left\{\mathrm{i}\gamma_k X_j\right\}$ represents a one-qubit RX_j gate acting on the qubit j. Therefore, their product can be applied in parallel in one step. Similarly, the matrix $\exp\left\{-\mathrm{i}\beta_k \left(v_j - \lambda(\Delta tk) w_j\right) Z_j\right\}$ represents a one-qubit RZ_j gate acting on the qubit j. Hence, the product of all RZ gates can be applied in parallel in one step. We conclude the number of steps required to apply all RX and RZ gates is independent of n i.e. it is constant per layer k.

We now examine two-qubit gates $RXX_{j,j+1}$ given by $\exp\left\{\mathrm{i}\gamma_k X_j X_{j+1}\right\}$. We note that their product forms a closed chain where the qubit j is connected to the qubit $j+1$ with $n+1 := 1$. Since for any i and j the matrices $\exp\left\{\mathrm{i}\gamma_k X_j X_{j+1}\right\}$ and $\exp\left\{\mathrm{i}\gamma_k X_i X_{i+1}\right\}$ commute we can rearrange their order into several sublayers so that each layer could be applied in parallel. Suppose we have an even number of qubits, i.e. $n = 2m$ for some $m \in \mathbb{N}$. Then the first sublayer will contain m gates that do not share any qubits and the second sublayer will contain the other half of m gates that do not share any qubits. Hence, the two sublayers of

RXX gates can be executed in 2 steps. If n is odd, then we get a third sublayer containing a single RXX gate. Hence, the three sublayers of RXX gates can be executed in 3 steps. It follows that regardless of the value of n it takes at most 3 steps to apply all RXX gates. Therefore, in total it takes at most 5 steps to apply RX, RZ and RXX gates. Hence each DAQC layer k in (E.1) can be applied in constant time. Since there are p layers the runtime t_{ss} is $O(p)$.

For the QUBO circuit, on the other hand, the runtime depends on the bound c, the number of KP variables n and number of layers p. Due to the quadratic penalty in the objective function (5.3) the kth DAQC layer has $\binom{n+\lfloor \log_2(c)\rfloor}{2}$ two-qubit gates RZZ. Since RZZ gates commute the best arrangement of the RZZ gates yields at least $n+\lfloor \log_2(c)\rfloor - 1$ sublayers that can be executed consecutively. To see this, we note that each sublayer can accommodate at most $(n+\lfloor \log_2(c)\rfloor)/2$ two-qubit gates such that no qubit is shared between all the gates in the sublayer. Therefore it takes at least $n+\lfloor \log_2(c)\rfloor - 1$ sublayers to place all the RZZ gates. This gives $(n+\lfloor \log_2(c)\rfloor - 1)(n+\lfloor \log_2(c)\rfloor)/2 = \binom{n+\lfloor \log_2(c)\rfloor}{2}$. Since the sublayers containing RZ and RX can be applied in constant time, the runtime of a kth layer is of order $O(n+\lfloor \log_2(c)\rfloor)$. Since the QUBO circuit has p layers, its runtime $t_{ss} = O(p(n+\lfloor \log_2(c)\rfloor))$.

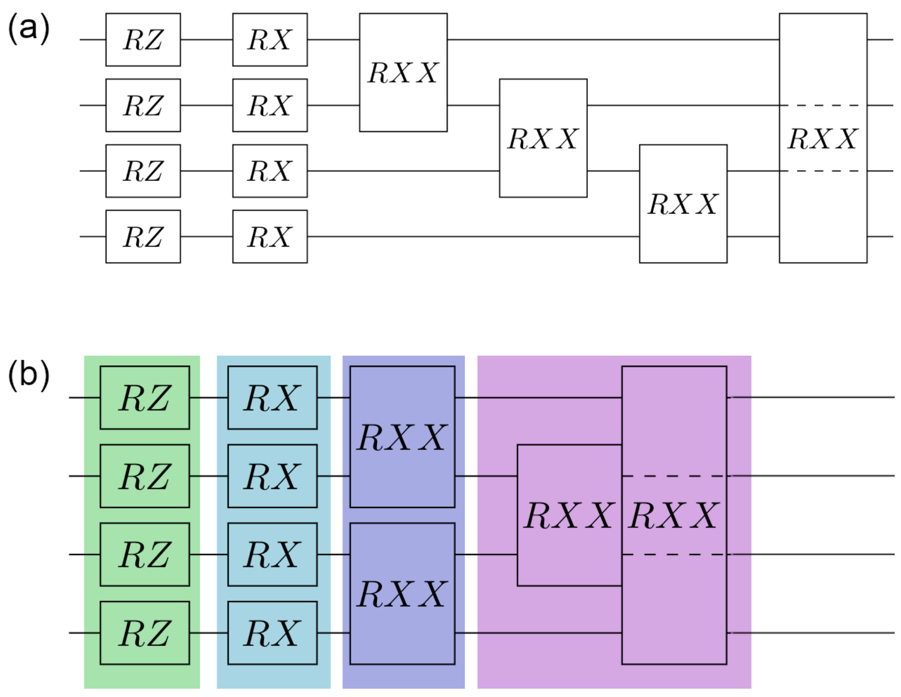


Figure E.1: (a) The kth layer of the Lagrangian circuit. By the choice of the coefficients $K_{i,j}$ in (4.8) we get the initial Hamiltonian (4.9), which in turn yields RXX gates with a closed chain-like connectivity. (b) Equivalent rearrangement of the kth layer. Gates are partitioned into four sublayers (coloured rectangles). Gates belonging to the same sublayer do not share qubits and can be applied in parallel in a single step. In total it takes four consecutive executions to apply the kth layer for any even n.