Quantum Annealing: Research and Applications

by

Mayar Mohamed

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Science
in
Physics (Quantum Information)

Waterloo, Ontario, Canada, 2021

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

This thesis is based on the forthcoming publications:

- Chapter 2 is based on

  - M. Mohamed, S. Ghose, and A. Mashatan, *Quantum Annealing for Solving QKP: Computational Analysis, In preparation, 2021.*

- Chapter 3 is based on

  - M. Mohamed, S. Ghose, and A. Mashatan, *Quantum Annealing approach to QUBO-Based Proof-of-Work in Blockchain, In preparation, 2021.*

Abstract

This thesis studies several aspects of the quantum annealing (QA) computing approach. Quantum annealers' primary objective is to solve hard computational optimization problems. Because these optimization problems are in the NP-Hard complexity class, they are of great interest in several fields. One of the leading open questions concerning quantum annealers asks whether they will outperform other classical methods for solving these problems; Some aspects of this question are addressed in this thesis. The first part of the thesis investigates whether quantum annealing provides improved performance for solving a particular family of NP problems, called the Quadratic Knapsack Problem (QKP), using the D-Wave Quantum Annealer. The performance metrics used to assess QKP solving are the solution quality and the total runtime, and are benchmarked against other classical solvers.

Furthermore, we extend our research on quantum annealers to propose two use cases for such systems. One is for Blockchain technology, and the second is in the area of quantum chaos. For the first use case of QA, an application for Blockchain's Proof of Work (PoW) is proposed, based on having hard optimization problems as an alternative to PoW hashing challenge, and using quantum annealers as solvers. For the second use case of QA, we propose simulating quantum chaos on the D-Wave Quantum Annealer to study the transition between the deep quantum realm and the classical limit in a chaotic system, and obtain insights into the "quantumness" of quantum annealers.

Acknowledgements

Undoubtedly, spending half of the graduate school experience under the pandemic lockdown has taken some toll on students everywhere, and I was no exception to the stress and anxiety of the uncertain. However, reaching this point where I am proudly able to present my dissertation to you wouldn't have been possible without the continuous and immense support of my supervisor Dr. Shohini Ghose; she is truly an inspiration to me and my role model on both professional and personal levels. I also would like to thank my co-supervisors, Dr. Robert Mann and Dr. Atefeh Mashatan, for their support and for giving me the opportunity to work on my research interests with them. Finally, I am also grateful to Hossein Sadeghi, the Team Lead at the Applications and Algorithms department at DWave Systems, for his continuous guidance through the testing phase and for providing me with free time access to the D-Wave's system.

During these challenging times, I am blessed with having people in my life who made it a little bit easier. I am forever grateful for having my closest friend Hiba in my life and my close friends Sara, Ali, and Manar. I am grateful for my mother, Nahed, and my three sisters, Noran, Nayerah and Mai, my cat Sushi, my bike, my paints and brushes, and my favorite music band, Bangtan Sonyondan!

Dedication

Dedicated to All Women.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

This thesis is motivated by the possible advantage of using the Quantum Annealing algorithm over classical methods for solving NP-Hard optimization problems. NP-Hard problems are of importance in a wide range of industrial and research fields. However, previous studies show that solving these problems using exact classical algorithms is highly inefficient, requiring an exponential amount of time. Hence, the interest in finding better algorithms for solving these problems is growing. One of the recently studied algorithms for this purpose is the quantum annealing algorithm. Quantum annealing was first proposed as a method to solve combinatorial optimization problems in [ACd89]. The proposed procedure is based on using the quantum tunneling effect to search for the global minima of the optimization problem while escaping from the local minima.

The interest in solving NP-Hard optimization problems using Quantum Annealing is due to the fact that it seems to provide a speed-up compared to the existing classical algorithms in some cases [AL18]. However, the performance of Quantum Annealing is still an open debate. Quantum Annealing is not always guaranteed to provide a quantum speed-up; in some cases, it can perform similarly or even worse than the classical methods [AKR10], implying that the Quantum Annealing algorithm is problem-dependent. Hence, empirical testing is needed for characterizing Quantum Annealing performance and whether

it provides an advantage over other methods for solving a particular family of a problem with a consistent structure. This research aims to study if quantum advantage is realized for solving a particular class of the NP-Hard problems, called the Quadratic Knapsack Problem (QKP), using the Quantum Annealing algorithm. The Canadian company D-Wave Systems, founded in 2000, is aiming to physically implement the Quantum Annealing algorithm for scalable quantum computers via the Adiabatic Quantum Computing (AQC) model. Their Quantum Annealer solvers are used to test Quantum Annealing performance for solving QKP in the Chapter (2). The second aim is to study Quantum Annealers for potential use cases, specifically, for Blockchain technology (chapter 3), and in the field of Quantum Chaos (chapter 4).

In the following sections, we provide the necessary background regarding Adiabatic Quantum Computing, Quantum Annealing, D-Wave's Quantum Annealer architecture, and the complexity class of the problems they solve. At the end of this chapter, we provide an overview of the structure of this thesis.

1.2 Adiabatic Quantum Computing

The Adiabatic Quantum Computing (AQC) model exploits the adiabatic theorem to solve an optimization problem. The adiabatic theorem states that given an initial system in the n_{th} eigenstate (typically the ground state for an optimization problem), the final state of the system remains in the same eigenstate if the system is evolved "adiabatically" [McG14]. For a system to evolve adiabatically, the evolution process should happen very slowly, such that the global changes in the system can take place without any changes happening locally in the system [JP94]. Hence, the AQC method can solve an optimization problem by evolving an initial Hamiltonian that is easily prepared in its ground state to a final Hamiltonian that remains in the system's ground state, which encodes the solution to the optimization problem. In its ideal form, AQC allows universal quantum computation, which means it can simulate other universal quantum computing models like the circuit model, with at most polynomial resource overhead [AvDK+08].

Typically, the universal AQC model is referred to as a "non-stochastic" model; it can simulate non-stochastic Hamiltonians that have both positive and negative off-diagonal

elements in their matrix representation. This allows the AQC model to solve any Turing-computable problem, which makes the model universal [McG14]. Hence if a problem can be designed on the circuit model, it can be mapped to the AQC model. Also, it is important to note that the computation on the AQC model does not have to occur in the ground energy state, which means that the problem could start with any initial Hamiltonian n_{th} eigenstate and still end up in the same eigenstate of the final Hamiltonian according to the adiabatic theorem. However, because the AQC is commonly used for solving optimization problems, the initial Hamiltonian is then prepared in its ground state, causing the final Hamiltonian to end up in the ground energy state as well, encoding the solution to the optimization problem [McG14].

Quantum Annealers Hardware tries to incorporate the AQC algorithm to solve computational problems by encoding the optimization problem via evolving the system to the ground states of final Hamiltonian. However, the physical realization for a Quantum Annealing system does not insist on adiabaticity condition due to the physical limitations of the hardware [VL17]. Hence, the Quantum Annealing model is a limited realization of the universal AQC model. Moreover, Quantum Annealing is a non-universal model as it can only simulate "stochastic" Hamiltonians, which can only have non-positive off-diagonal elements in their matrix representation [HBCT17].

Three main studies greatly influenced the development of Adiabatic Quantum Computing (AQC) in the literature:

1. The Hamiltonian-to-Circuit construction that Richard Feynman introduced in 1985 [Fey82], then later developed by Kitaev in 2002.

2. Farhi et al. in their study: Quantum computation by adiabatic evolution, as they were the first to develop the quantum adiabatic algorithms.

3. The 2004 study by Aharonov et al. [AvDK+08], which showed that adiabatic quantum computing is polynomially equivalent to the standard quantum circuit model.

Before describing the AQC model in more detail, we give a very brief description of the quantum circuit-based model for comparison. However, the main focus of the chapter

is the AQC model. First, because both models are universal descriptions of the quantum computing model, it is only appropriate to present the fundamental similarities between them before discussing their differences. In the general description, the building block of any quantum computing model is the qubit, a two-level quantum state. A convenient way to present the pure quantum state of the qubit is in the computational basis $\{|0\rangle, |1\rangle\}$, which is typically represented by the vector state $|\psi\rangle$ in a Hilbert space [SC95];

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \rightarrow \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \tag{1.1}$$

In Eq. (1.1), α and β are complex values of the probability amplitudes of the qubit state. These probability amplitudes determine the qubit's state being measured as 0 or 1, and they need to be normalized for the qubit state to be a valid quantum state [DW19].

$$|\alpha|^2 + |\beta|^2 = 1 \tag{1.2}$$

Furthermore, this description can generalized to describe a system state $|\Psi\rangle$ with multiple qubits as;

$$|\Psi\rangle = \sum_{i=0}^{N} \alpha_i\,|\psi_i\rangle \tag{1.3}$$

$$\sum_{i=0}^{N} |\alpha_i|^2 = 1$$

An alternative representation to the vector state in Eq. (1.3), is the density matrix representation, which will become relevant when we discuss quantum chaos in Chapter (4). The quantum state is written as the density matrix ρ;

$$\rho = |\psi\rangle\,\langle\psi| \tag{1.4}$$

The representation in Eq. (1.3) is an ideal quantum state described by pure states; however, it is not always possible to construct states that are purely deterministic, meaning

that the system initially might be in a probabilistic superposition of these pure states. Hence, a more general state vector that can describe this probabilistic superposition of pure states is called the "mixed state" and it can be represented by the density matrix [Mer07];

$$\hat{\rho} = \sum_i p_i \left|\psi_i\right\rangle \left\langle\psi_i\right| \tag{1.5}$$

where,

$$\sum_i p_i = 1 =, 0 \leq p_i \leq 1$$

Performing operations on a quantum state to evolve the system is done by applying unitary operations on the quantum state; the unitary operator $\hat{U}$ can be represented as:

$$\left|\psi(t_2)\right\rangle = \hat{U}(t_2, t_1) \left|(t_1)\right\rangle \tag{1.6}$$

For the operator U to be unitary, $\hat{U}\hat{U}^\dagger = 1$ needs to hold.

Finally, the last piece to complete the description of the quantum computing model is the system measurement, which is a crucial part of doing quantum computing. Because of the probabilistic nature of the measurement outcomes that corresponds to a physical observable A in the system, the measured value of any observable in the system must be calculated as an expectation value. To every measurable physical quantity in the system, there exists a Hermitian operator $\hat{A}$ for it. The probability p of measuring the system in the specific state m is presented as [Gri60];

$$p(m) =< \psi|\hat{A_m}^\dagger \hat{A_m}|\psi > \tag{1.7}$$

In the quantum circuit model, as shown in Fig. (1.1), the horizontal lines follow the qubits through time, and the boxes represent quantum gates that can operate on single or multiple qubits. The state of n qubits evolves by these unitary operations in discrete time steps. These quantum gates operating on the qubits are often represented by matrices, as they act on qubits represented by state vectors [DW19]. The state $|\Psi_0\rangle$ in Fig. (1.1) is the

input state in the tensor product of the subsystems $|\psi_1\rangle \otimes |\psi_2\rangle$, as shown, the qubits states are changed after each operation in discrete time steps, and finally, $|\Psi_3\rangle$ is the output state [Mer07].

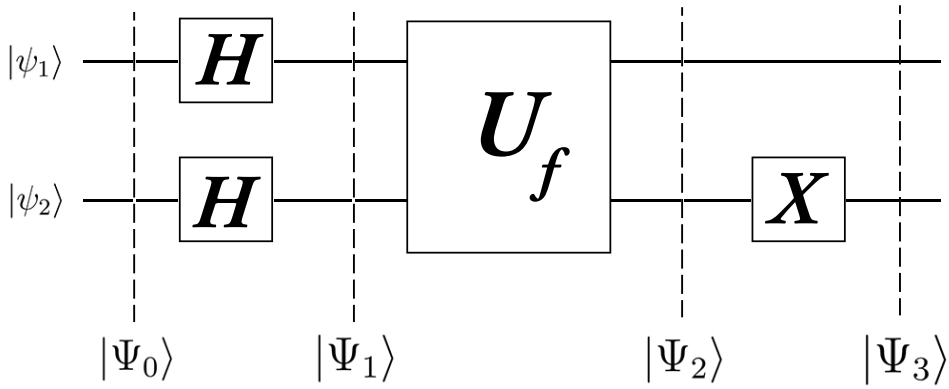

Figure 1.1: Quantum ciruit gate model

Unlike the circuit model, there are no gates or discrete time steps in AQC. Instead, as shown in Fig. (1.2), the qubit states gradually evolve according to certain forces represented by the Hamiltonian $H(t)$ which makes the system evolve from the initial Hamiltonian H_0 to the final Hamiltonian H_f. The system evolution in AQC is continuous and is governed by Schrodinger's equation in Eq. (1.8), where the state of the n qubits at time t is defined by $\Psi(t)$ [McG14].

$$i\hbar \frac{\partial}{\partial t}\Psi(t) = H\Psi(t) \tag{1.8}$$

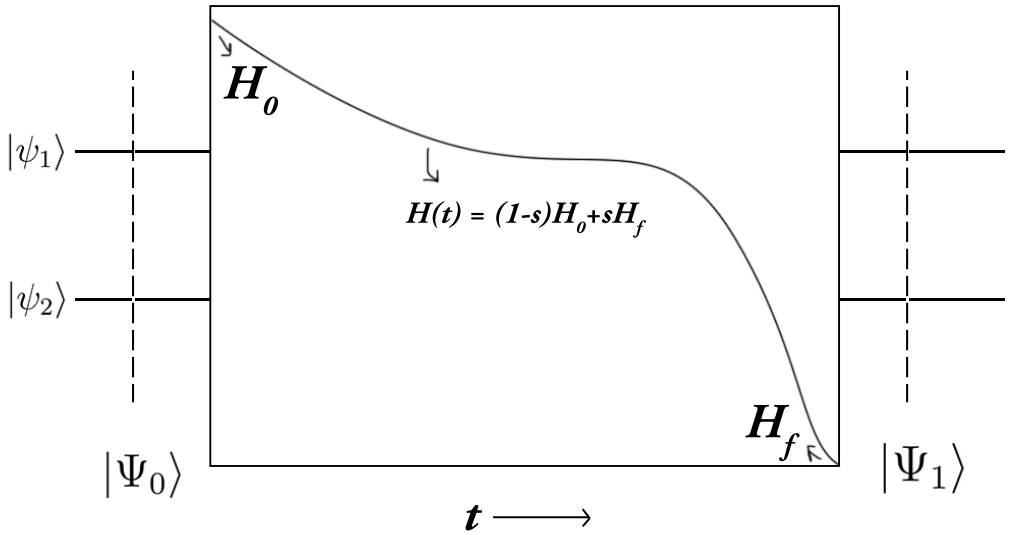

Figure 1.2: Adiabatic Quantum Computing model

1.2.1 The Adiabatic Quantum Algorithm

In Farhi et al. [FGG+01], the quantum adiabatic algorithm is proposed for solving various instances of satisfiability problems (SAT) (known to be NP-Hard), using adiabatic evolution. Moreover, they suggested that the adiabatic quantum algorithm can solve hard optimization problems as a formulation of an energy minimization problem.

The algorithm for solving an optimization problem P is described as follows:

(i) The system is initialized into the ground energy state of an easily prepared Hamiltonian H_0.

(ii) The system is then adiabatically evolved for time T, to a more complex final Hamiltonian H_f, such that its ground energy state encodes the solution to the problem.

The final Hamiltonian evolves according to:

$$H(t) = (1 - \frac{t}{T})H_0 + \frac{t}{T}H_f = (1 - s)H_0 + sH_f \tag{1.9}$$

This adiabatic quantum algorithm exploits the quantum adiabatic theorem. The theorem in its original form, formulated by Max Born and Vladimir Fock (1928), is stated as follows:

> "A physical system remains in its instantaneous eigenstate if a given perturbation is acting on it slowly enough and if there is a gap between the eigenvalue and the rest of the Hamiltonian's spectrum."[BF28], [Hen13]

This means that if a system is in the ground eigenstate of H_0, it will remain in the ground eigenstate of H_f at the end of the process if the change is adiabatic. For the evolution to be adiabatic, the time over which the Hamiltonian changes needs to be much larger than the characteristic time scale for changes in the system.

To achieve an adiabatic change, the time scale of the change must be proportional to the inverse of the minimum energy gap between the ground state $E_0(s)$ and the lowest excited state $E_1(s)$ [FGG+01].

$$\begin{gathered} T >> \frac{1}{g_{min}^2} \\ g_{min} = min_{0 \leq s \leq 1}(E_1(s) - E_0(s)). \end{gathered} \tag{1.10}$$

Consequently, the total run time for the adiabatic change depends on the inverse square of the energy gap. Hence, finding g_{min} is important for assessing the capabilities of adiabatic quantum computing. From this relationship, the closer together the two lowest energies get, the slower we have to run the algorithm. In case the levels cross, sending $g_{min} \to 0$, the running time will $T \to \infty$. However, finding g_{min} to assess the performance of AQC is as difficult as solving the original problem. Hence, bounds on g_{min} are typically considered when assessing the performance of AQC optimization.

This implies that the adiabatic quantum algorithms for optimization sometimes fail for some NP-hard problems, as the ground energy and the first excited energy often do get

exponentially close to each other. Hence, to avoid the level crossing, the algorithm needs to run for an exponentially long time to remain in the ground state. Several studies in the literature [FGGN08], [VDMV01], [JRS07], [AC09], [FGG+09] explored the conditions under which the AQC algorithm fails. [VDMV01] showed that it is problem-dependent, such that the energy gap can become exponentially small for hard-designed instances.

1.3 Quantum Annealing

Quantum Annealing (QA) is designed to solve hard computational optimization problems, where the solution is encoded into the final ground state of a time-dependent quantum Hamiltonian. QA performs a slow adiabatic process to let the initial ground state of the Hamiltonian evolve to the final ground state of the Hamiltonian that gives the desired solution of the problem [DC08]. However, unlike the universal AQC model, quantum annealing discounts various physical aspects of the system environment, and it does not necessarily insist on following the adiabatic process, making it a non-universal model [HBCT17].

Quantum Annealing can be compared to Simulated Thermal Annealing, which is used to find the optimal solution via thermal fluctuations. In Simulated Annealing, the algorithm implements the metal-cooling analogy to simulate the annealing procedure. Random points are generated in the neighborhood of the current best optimal point, and the problem functions are evaluated at those points. If the evaluated value obtained at one of these points is better (or lower) than its current best value, then the algorithm accepts this point, and the better value is updated accordingly. However, the point is sometimes accepted or rejected if its value is worse (or higher) than its best-known value so far. The decision of accepting or rejecting a point is probabilistic and is based on the probability value of the density function in the Bolzman-Gibbs distribution [Aro04].

Quantum Annealing is similar to Simulated Annealing in the sense that the temperature parameter is analogous to the tunneling field strength in Quantum Annealing, since it exploits quantum tunneling on the quantum scale. The idea behind producing the quantum tunneling effect by a transverse field to help evolve the system into its ground state was first proposed in [RCC89]. In simulated annealing in fig (1.3), the temperature determines the likelihood of moving to a state of upper "energy" from one current state. The weak

thermal fluctuations allow the system to navigate its energy landscape and reach a low-energy configuration. However, the strength of the transverse field in quantum annealing determines the quantum-mechanical likelihood to manipulate the amplitudes of all states in superposition. The qubits need not get trapped in energy minima that are not actually the solution (local minimum) to solve the optimization problem. Classically, if the qubit gets trapped in a local minimum, it would need extra energy to climb out of the local minimum to reach the global minimum. However, with quantum tunneling, qubits can tunnel through these barriers as they look for the lowest energy minimum [TC15]. Hence, the advantage of Quantum Annealing over the classical Simulated Annealing lies in the exploitation of quantum tunneling. However, this does not guarantee that Quantum Annealing will always perform better than the classical methods because of the existence of exponentially small minimum energy gaps g_{min} for some hard instances.

In Chapter (2), we will compare the performance of simulated annealing to quantum annealing using the D-Wave Quantum Annealer.

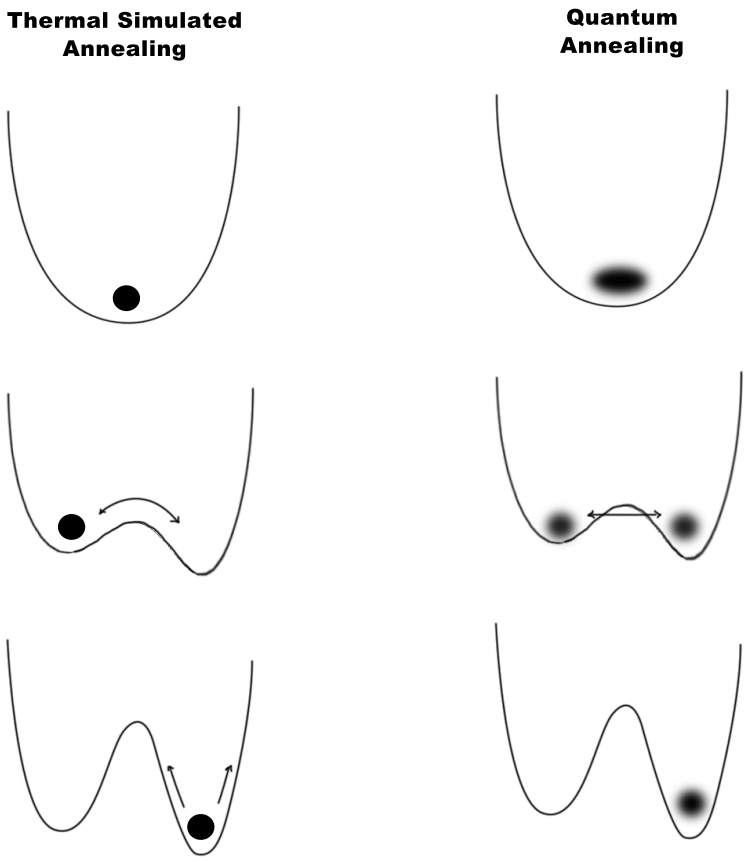

Figure 1.3: Simulated annealing versus quantum annealing

To perform Quantum Annealing, quantum fluctuations are induced by adding a transverse magnetic field in the x-direction, represented by the Pauli matrix $\hat{\sigma}_i^{(x)}$ acting on qubits i, obtaining the time-dependent N-qubit Hamiltonian [McG14]

$$H(t) = \Gamma(t) \sum_{i=1} \Delta_i \sigma_i^x + A(t) H_f, \tag{1.11}$$

such that $\sum_{i=1} \Delta_i \sigma_i^x$ is a suitably chosen non-commuting quantum tunneling Hamiltonian, and its ground state is a superposition of the states 0 and 1. The Δ_i parameter is introduced

to account for the non-zero probability of the qubit tunneling between the two potential wells associated with the states. The system is easily initialized in this state, and during quantum annealing, the transverse term Γ in Eq. (1.11) is gradually decreased from one to zero, and A of the final Hamiltonian is increased from zero to one, thus ending up in the ground state of H_f according to the adiabatic theorem.

1.3.1 Combinatorial Optimization Problems

The class of problems we are interested in are the **Combinatorial Optimization** problems, which are concerned with finding an optimal configuration of objects from a finite set of objects that optimize the cost of the problem [PS98]. This set of problems includes the infamous Traveling Salesman problem, satisfiability problems, and many others. For example, in the Traveling Salesman Problem, given a set of cities and the distance between each possible pair of them, an optimization algorithm seeks to find the best possible configuration of paths between these cities under the constraint that a city must be visited only once, and the traveling path must return to the starting point at the end [GG78]. Combinatorial optimization problems are in the NP-Hard complexity class, making them of huge interest in several fields, including artificial intelligence, machine learning, lattice protein models in biology, and many other areas in industry and businesses.

To understand the complexity class of the combinatorial optimization problems, thr P, the NP, NP-complete, and NP-hard classes are defined as follows [Häm06]:

1. **P** (Polynomial) complexity class contains all decision problems that are easy to solve and have deterministic polynomial-time algorithms.

2. A problem in the **NP** (Non-deterministic Polynomial) complexity class is a decision problem that is hard to solve in polynomial time, but easy to verify a given answer in polynomial time.

3. **NP-complete** problems, shown in Fig. (1.4), are also decision problems that belong to both NP and NP-hard classes. As a result, NP-Complete problems are verifiable in polynomial time and any NP problem can be reduced to it in polynomial time. NP-complete problems are the hardest problems in the NP complexity class. [MRKA18].

4. The **NP-hard** problems are not necessarily decision problems, so they don't need to be necessarily in the NP class as shown in Fig. (1.4), and we cannot prove that we can verify their solutions in polynomial time. In order to define a problem A as NP-hard, there should exist an NP-complete problem B, such that B can be reduced to A in polynomial time.

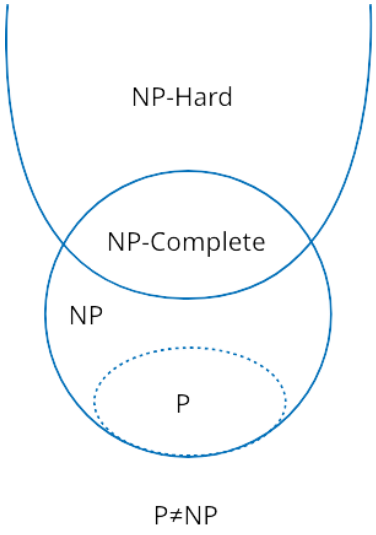

Figure 1.4: complexity classes provided that $P \neq NP$

The optimization version of a combinatorial optimization problem is NP-hard if its decision version is proved to be an NP-complete problem.

For example, the question that asks 'Given a graph G that is defined by $G = (V, E)$, with a finite set of vertices V and a set of edges E that connects each pair of vertices; is there a clique of size K, defined as a complete subgraph of G, that exists in the graph?' is an NP-Complete problem with the answer 'yes' or 'no' [BBPP99]. However, the NP-hard problem form of this problem seeks to find the largest clique in the graph G.

A NP-Hard problem is formulated as follows [Häm06]:

Given an objective function $f : D^n -> R$ defined on n variables $x = x_1...x_n$

from some discrete domain D, find an assignment of value to x that minimizes $f(x)$.

It is not difficult to cast a given NP-Hard problem into an optimization framework expressed in binary (Boolean) variables. Let there be an optimization problem of the form [McG14]:

$$E(\vec{s}) = -\sum_{i} h_i s_i + \sum_{i,j>i} J_{ij} s_i s_j \tag{1.12}$$

where $s_i = \pm 1$, for any given set of h_i and J_{ij} ranges from -1 to 1, and there exists at least one optimal solution $\vec{s}$ that minimizes the objective E. This problem is a NP-hard problem.

To solve these kinds of problems with Quantum Annealers, a physical problem has to be identified, where some Hamiltonian represents the objective function. The following subsection discusses the Ising Hamiltonian on D-Wave, used to express the objective function in (1.12).

1.3.2 Quantum Annealing on D-Wave

The D-Wave 2000Q implements the quantum annealing algorithm using the following Hamiltonian:

$$\mathcal{H}_{ising} = A(s)\mathcal{H}_0 + B(s)\mathcal{H}_f \tag{1.13}$$

In Eq. (1.13), $A(s)$ and $B(s)$ are the driving functions and are changed over time. The $\mathcal{H}_0$ is the starting Hamiltonian, and the $\mathcal{H}_f$ is the final Hamiltonian, which is the objective function of the problem that the algorithm seeks to find the ground state of it. Typically, the annealing process begins at $s = 0$ with $A(s) >> B(s)$ and ends at $s = 1$ with $A(s) << B(s)$. The system is slowly annealed by decreasing A and increasing B. Therefore, as annealing happens, the Hamiltonian's problem is introduced, and the influence of the initial Hamiltonian is reduced. At the end of the anneal, the eigenstate of the problem Hamiltonian is reached, making the final Hamiltonian's lowest energy state the answer to the problem we are trying to solve.

For the D-Wave system, the Hamiltonian is the sum of two terms, the initial Hamiltonian and the final Hamiltonian, and it is represented as [McG14]:

$$\mathcal{H}_{ising} = -A(s)\left(\sum_i \hat{\sigma}_x^{(i)}\right) + B(s)\left(\sum_i h_i \hat{\sigma}_z^{(i)} + \sum_{i>j} J_{i,j}\hat{\sigma}_z^{(i)}\hat{\sigma}_z^{(j)}\right) \tag{1.14}$$

In the first part of the Hamiltonian, the initial system is prepared in the ground energy state by applying a strong transverse magnetic field represented by the Pauli matrix $\hat{\sigma}_x^{(i)}$ in Eq. (1.17), and all qubits i are in a superposition state of 0 and 1. The initial Hamiltonian is diagonal in the Hadamard basis, which is an alternative basis to the computational basis $|0\rangle$ and $|1\rangle$ in the z basis, represented by $|+\rangle$ and $|-\rangle$. The Hadamard basis states can be prepared by applying the Hadamard operation in Eq. (1.20) to the computational basis states to create a superposition;

$$|+\rangle = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \tag{1.15}$$

$$|-\rangle = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{pmatrix} \tag{1.16}$$

The Pauli matrices operating on qubits

- Pauli-X is equivalent to the classical NOT gate. It maps $|0\rangle$ to $|1\rangle$, and $|1\rangle$ to $|0\rangle$.

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{1.17}$$

- Pauli-Y maps $|0\rangle$ to $i\,|1\rangle$, and $|1\rangle$ to $-i\,|0\rangle$

$$\sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \tag{1.18}$$

- Pauli-Z maps $|1\rangle$ to $-\,|1\rangle$, and does not change $|0\rangle$

$$\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{1.19}$$

Hadamard Operation

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{1.20}$$

$$H|0\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle = |+\rangle$$

$$H|1\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle = |-\rangle$$

Therefore, to start the annealing process at $t = 0$, the system starts from the lowest energy eigenstate of the initial Hamiltonian $\mathcal{H}_0$ in the hadamard basis, given by;

$$|+(t=0)\rangle = \frac{1}{\sqrt{2}^N} \otimes_{i=1}^{N} (|0\rangle_i + |1\rangle_i) \tag{1.21}$$

In the second term of the Hamiltonian in Eq. (1.13), i and j represent two neighboring qubits, and interactions between them are quantified by coupling strengths J_{ij}. The linear coefficients corresponding to qubit biases are h_i, and $\sigma_z^{(i),(j)}$ in Eq. (1.19) are Pauli matrices that represent the local longitudinal magnetic fields acting upon qubits i and j, respectively, according to the qubit biases h_i. This Ising model is shown to be in the NP-Hard complexity class, and it could be mapped to any other NP-Hard problem in polynomial time. Hence, the objective function in Eq. (1.12) can be expressed by the Ising model.

D-Wave supports two equivalent formulations for the objective functions:

1. The Ising model
2. The Quadratic Unconstrained Binary Optimization model (QUBO).

Ising model:

$$E_{ising}(s) = \sum_{i=1} h_i s_i + \sum_{i<j} J_{i,j} s_i s_j \tag{1.22}$$

The variables in Eq. (1.22) are the qubit states that correspond to +1 and -1 values, which could be the state “spin up” or “spin down”.

However, because we deal with computational optimization problems, it is more convenient to represent those problems using logical values $0, 1$ rather than $-1, 1$. Moreover, as mentioned, D-Wave also supports the QUBO model, which can be trivially converted to and from the Ising model.

QUBO model:

$$f(x) = \sum_i Q_{i,i} x_i + \sum_{i<j} Q_{i,j} x_i x_j \tag{1.23}$$

Q is an upper-diagonal matrix, which is an N x N upper-triangular matrix of real weights.

QUBO can also be expressed more concisely:

$$\text{Min } x^T Q x \ , x \in \{0, 1\}$$

Several hard optimization problems can be properly encoded into the QUBO model. Some of these problems are Quadratic Assignment Problems, Constraint Satisfaction Problems (CSPs), Task Allocation Problems, P-Median Problems, Quadratic Knapsack Problems, Multiple Knapsack Problems, Set Partitioning Problems, and more [GKD19]. In this thesis, we will focus on the Quadratic Knapsack Problem.

1.3.3 D-Wave architecture

When a QUBO problem is submitted to D-Wave's Quantum Processing Unit (QPU), it gets translated and embedded on a physical system. D-Wave's architecture is simply a lattice of interconnected qubits that are connected via "couplers". However, the qubits in the current topology are not fully connected. This is something that should be considered while mapping the problem onto the physical topology, although it is usually handled automatically by the D-Wave software. The first topology D-Wave developed was the "Chimera" topology. At the end of year in 2020, they released the new topology "Pegasus" [Doc].

The D-Wave 2000Q QPU consists of a 16×16 grid of unit cells on the Chimera topology. The unit cell on the Chimera graph is a $K_{4,4}$ bipartite graph, as shown in Fig. (1.5). Each unit cell on the Chimera graph has eight qubits, and each qubit is connected to its neighboring qubit (represented by the purple dots in the figure) by internal couplers (the green lines in the figure), resulting in each qubit being connected to 4 other orthogonal qubits via these internal couplers. Different unit cells are connected by external couplers (the blue lines in the figure), and they connect the parallel qubits in the same horizontal or vertical line. The external couplers give more freedom to the qubit, allowing it to be connected to a maximum of 6 qubits.

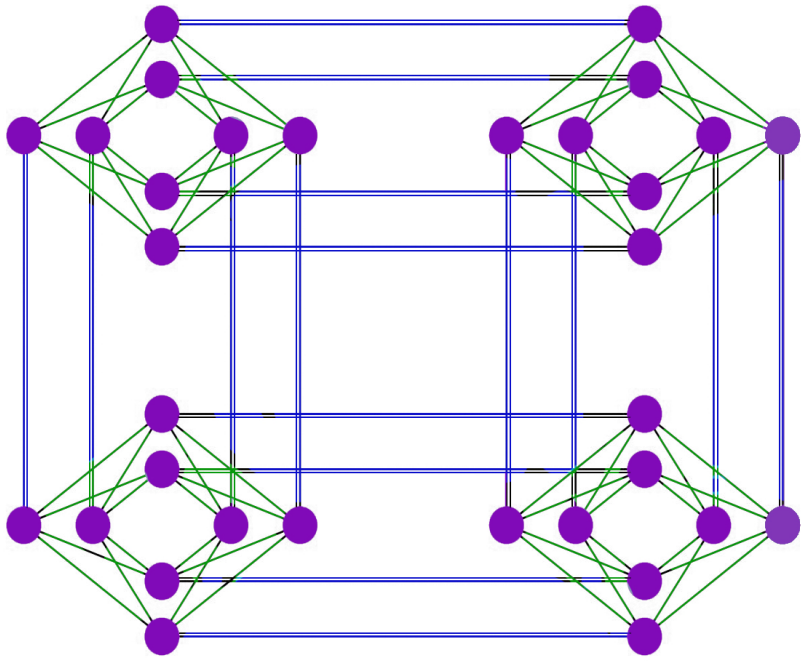

Figure 1.5: Chimera topology graph, showing four unit cells.

D-wave's new QPU is called the Advantage System, and it implements the "Pegasus" topology [Doc]. This new topology is also a lattice of a 16×16 grid of unit cells but is more complicated than the Chimera topology. In addition to the internal and external couplers,

Pegasus adds "Odd couplers", which allows each qubit to be connected to a maximum of 15 qubits.

1.4 Structure of Dissertation

The general theme of this thesis is the study of Quantum Annealers and their potential use cases.

Chapter (2) investigates whether Quantum Annealing provides improved performances for solving a particular family of NP problems, called the Quadratic Knapsack Problem (QKP), by evaluating the solution quality and the total runtime. Furthermore, transforming the native QKP problem to D-Wave's QUBO form and the impact of this transformation on the problem's complexity are also studied.

In Chapter (3), we propose a use case for Blockchain's Proof-of-Work (PoW) using Quantum Annealing. Our proposal involves formulating a PoW challenge based on the NP-hard optimization problem QUBO, with the goal of reducing blockchain's energy waste and possibly improving its decentralization and its scalability. The main challenge of this study is to formulate a good PoW that satisfies the hardness adjustability property, such that the PoW challenge achieves the desired difficulty. Expanding on the results found in Chapter (2), the relationships found between the problem's parameter and runtime is used as tuning parameter for the PoW difficulty.

Chapter (4) discusses the second use case of QA. We propose simulating quantum chaos on a quantum annealing system like D-Wave. Quantum Annealers with large numbers of qubits may be better suited than small-scale circuit-based quantum computers for simulating chaotic quantum models to study the chaotic transition between the deep quantum realm to the classical limit, This proposal studies the well-known kicked Ising model, which displays many-body quantum chaos, and is naturally mapped onto the Quantum Annealing hardware.

Finally, Chapter (5) summarizes the main results found in this thesis and discusses potential future work.

Chapter 2

Quantum Annealing for Solving QKP: Computational Analysis

2.1 Overview

Solving NP-hard combinatorial optimization problems is of enormous interest in several fields, including artificial intelligence, machine learning, and many other areas in industry and businesses. However, the question of whether Quantum Annealing (QA) computing will exceed the classical methods for solving such problems is still an open one. Our research investigates whether Quantum Annealing provides increased performance for solving a particular family of NP problems, called the Quadratic Knapsack Problem (QKP), regarding the solution quality and the total runtime. Furthermore, transforming the native QKP problem to the Quantum Annealer D-Wave's QUBO form is also studied, and the impact of this transformation on the problem's structure and complexity.

2.2 Background

Unlike exact algorithms, metaheuristics algorithms come with no theoretical proofs, which makes computational evaluation important. Metaheuristic solvers, like Quantum Annealers, are evaluated for their solution quality, and computation time [ZE81]. In this project,

we obtain empirical results from solving the NP-Hard Quadratic Knapsack Problem (QKP) on the D-Wave QA solver and comparing it against classical solvers.

2.2.1 Mapping QKP onto Quantum Annealers

An example of the Quadratic Knapsack Problem arises in telecommunication, which was first discussed in [Wit75]. In Fig. (2.1), the locations of a set of satellite stations must be selected in a way that maximizes the global traffic between them while respecting a budget constraint. This problem can be extended to other applications, for example, it could be used to optimize the distribution of freight handling terminals, railway stations and many others.

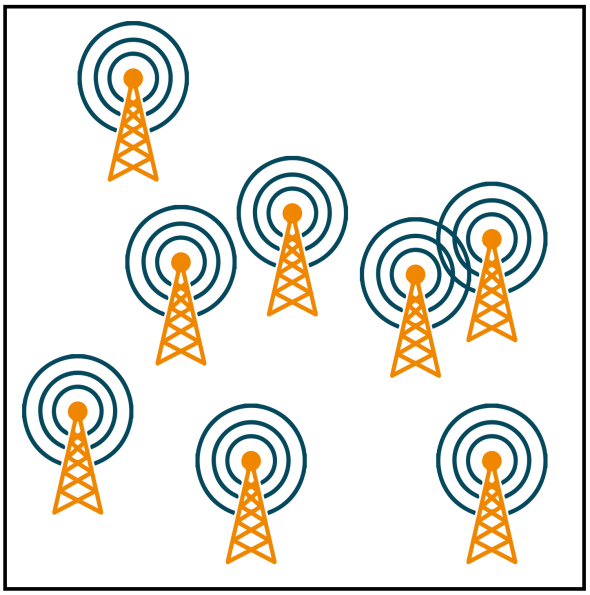

Figure 2.1: QKP example for satellite stations locations selection

QKP In scalar form [Pis07];

$$\text{Max f} \left\{ \sum_{i=1}^{n} \sum_{j=1}^{n} q_{ij} x_i x_j : x \in X, x \text{ binary} \right\}$$
$$\text{s.t } X \equiv \left\{ x \in \{0,1\}^n : \sum_{i=1}^{n} w_i x_i \leq c; x_i \in \{0,1\} \right\} \quad \text{for } i = 1, \ldots, n \tag{2.1}$$

Applying this formulation on the mentioned satellite example in Fig. (2.1), q_{ij} would be the link benefit, and it's realized whenever both sites i and j are selected for the communication facilities. This link benefit could be a measure of revenue, service improvement or both combined.

The objective is to determine which sites are the best out of a mass of potential candidates, based on some measure of goodness, while respecting some constraints, which could be financial, geographical or both.

Equivalently, QKP In the matrix form [Pis07];

$$\text{Max } x^T Q x$$
$$\text{s.t } w^T x \leq c, x \text{ binary} \tag{2.2}$$

We see that QUBO and Ising formulations are unconstrained functions; however, QKP is a constrained function that involves inequalities. To transform the constrained NP optimization problem to unconstrained Ising, a common approach in the literature is to add the constraint as a penalty term to the objective function, such that this term adds a positive energy cost to any proposed solutions that exceed the constraint it replaces. However, this approach has a drawback in that the penalty term is a squared expression, resulting in increased complexity with the growing size of the problem. Fortunately, there are few suggested approaches for more efficient mapping [VD19].

As the focus of this project is assessing the quantum advantage and studying the hardness and the structure of the QUBO reformulation, we will be only exploring the common approach for mapping the QKP into the Ising formulation in the first stage of

the project, which is explored in this chapter. In [Luc14], the QUBO mapping of 21 NP-complete problems into an Ising formulation is represented using the common penalty approach. One of the problems that is represented in [Luc14] is the Knapsack Problem (KP) with Integer Weights. However, following Lucas's approach, the KP formulation is expanded to include quadratic terms for QKP. To formulate QKP as an Ising problem, we solve this an energy function with goal of minimizing the energy cost, the function consists of two components:

$$H = H_A + H_B \tag{2.3}$$

In Eq. (2.3), H_A is the energy part that penalizes the function if the weight of the objects in QKP exceeds the value of the constraint, and H_B is energy that maximizes over the number w of subjects included. The goal is to have Eq. (2.3) be 0 at the end of the anneal.

To formulate the first part of Eq. (2.3), we refer back to Eq. (2.1). This knapsack can only carry weight W. Let the total weight in the knapsack be:

$$W = \sum_{i=1}^{N} w_i x_i$$

The total cost is

$$\chi = \sum_{i=1}^{N} \sum_{j=1}^{N} q_{ij} x_i x_j$$

The goal is to maximize χ subjects to the constraint that $W \leq c$.

Next, let the final weight of the knapsack be n after the optimization process; hence, we want to construct a Hamiltonian such that the ground state of this Hamiltonian is 0 iff the final weight $n \leq W$ exists.

We introduce a binary variable y_n, which is 1 if the final weight of the knapsack is n for $1 \leq n \leq c$, and 0 otherwise. The Hamiltonian that can give us this result is

$$H_A = A(1 - \sum_{n=1}^{c} y_n)^2 + A(\sum_{n=1}^{c} n y_n - \sum_{i}^{N} w_i x_i)^2 \tag{2.4}$$

In this Hamiltonian, we ensure that weight takes only on one value, and also the weight of the objects in a knapsack will equal the value we said it would.

The second part of the Hamiltonian (2.3) maximizes over the number of χ subjects included, and it penalizes solutions that weakly violate H_A at the expense of a more negative H_B, meaning that it's not allowed to add one item with heavy weight versus maximizing over multiple items, therefore:

$$H_B = -B \sum_{i=1}^{N} \sum_{j=1}^{N} q_{ij} x_i x_j \tag{2.5}$$

Consequently, N^3 spins are required to encode this formulation. However, we can reduce the number of extra y_n spins that must be added (slack variables) in Eq. (2.4) from $O(c)$ to $O(log(c))$ by using the logarithmic reduction trick in [Luc14]:

$$2^M \leq c < 2^{M+1}$$
$$\text{where, } M = log_2(c)$$

In this case, we only need $M + 1$ binary variables, and the spins required are reduced to $N^2 + [1 + log(c)]$ After updating the variables, the formulation becomes,

$$H_A = A \left(\sum_{n=0}^{M-1} 2^n y_n + (c + 1 - 2^M) y_M - \sum_{i=1}^{N} w_i x_i \right)^2 \tag{2.6}$$

In eqs. (2.5) and (2.6), A and B are some constants for separation of energy scales, and they are typically scaled to 1 in the literature. However in the QKP formulation case, to minimize the energy, we must ensure that B is smaller than A in order to avoid violating the constraint of H_A. For simplicity, we let $B = 1$, and define A as the penalty scalar $\mathbf{P}$, which could be any positive value. Hence, the complete formulation becomes:

$$H = P \left(\sum_{n=0}^{M-1} 2^n y_n + (c + 1 - 2^M) y_M - \sum_{i=1}^{N} w_i x_i \right)^2 - \sum_{i=1}^{N} \sum_{j=1}^{N} q_{ij} x_i x_j$$

The qubogen package from github/tamuhey is QUBO matrix generator on different major combinatorial optimization problems written in Python, and it follows the same procedure we outline in this section. The following QKP-to-QUBO code is used to convert the QKP benchmarking instances to QUBO.

```
1  Quadratic Knapsack Problem (QKP) → QUBO generator
2  Function qubo_qkp(obj, w, c, p):
       Input:  obj ⟵ array values of the objective fn.
       w ⟵ array values of the constraint fn.
       c ⟵ integer value of the constraint capacity
       p ⟵ integer penalty value
       Output: QUBO Matrix Q
3      n = len(obj)
                                          // size of the QKP objective fn.
4      nslack = np.ceil(np.log2(c))
              // the ceil of the number of the slack variables is log2(c)
5      slack = 2^(np.arange(nslack))
6      w = np.concatenate([w, slack])
           // the updated constraint fn.  with the added slack variables
7      Q = penalty * (np.outer(w, w) - 2 * c * np.diag(w))
8      Q[:n, :n] -= obj
9      return Q
```

2.3 Approach

This section presents our approach to empirically test the QUBO reformulation of the QKP native problem and compare QA performance against a variety of classical algorithms, including exact algorithms and other classical metaheuristics. Furthermore, we want to place other literature results on QKP benchmarking in context with our findings.

To select appropriate benchmarking instances for the computational experiments, randomly generated QUBO problems are unlikely to provide useful benchmarking results for a metaheuristic solver, due to the limitations of generalizing the optimization benchmark results to other problems with different structure [SM20]. Also, random instances are unlikely to be representative of real-world problems. Therefore, we need families of related instances with controllable parameters to test the quantum annealing algorithm [PWS+16]. The group of instances that are tested in this study includes common instances used in other existing computational experiments for QKP.

The group of instances used for testing was generated by Billionnet and Soutif in [BS04] and is made up of 100 small and medium-sized instances, and their optimal solutions are known in [CH17]. These instances are defined by their object numbers $n \in \{100, 200, 300\}$, density $d \in \{25\%, 50\%, 75\%, 100\%\}$, where density here is the probability of non-zero coefficients of the objective function. In the objective function in Eq. (2.1), the integer coefficients q_{ij} are uniformly distributed in the interval $[0, 100]$, and each weight w_j in the constraint function in Eq. (2.1) is uniformly distributed in $[1, 50]$, and also the capacity c is randomly selected from $[50, max(\sum w_j)]$. The structure of these instances assures assures a feasible solution. Finally, all runs are carried out on MacBook Pro with an 2.7 GHz Quad-Core Intel Core i7 and 16 GB RAM. The code scripts written for the tests are included in Appendix B.

The test contains the following components:

1. Problem types:
    - Native QKP problem
    - Transformed QKP-QUBO
2. Problem parameters:
    - n, native QKP problem size
    - d, native QKP density
    - **P**, scalar penalty added to the QUBO

- p, QUBO matrix diagonal dominance

3. Algorithms & solvers:

   - CPLEX - Classical Solver and uses Branch and Bound algorithm (Exact algorithm)
   - D-Wave, Simulated Annealing - Classical Solver and uses Metaheuristic algorithm
   - D-Wave, Advantage System - Quantum Solver and uses Metaheuristic algorithm
   - D-Wave, Hybrid Solver - Classical & Quantum solvers and uses Metaheuristic algorithm

2.3.1 Working Mechanisms of the Used Algorithms

CPLEX - Exact Algorithm

CPLEX Algorithm is based on using Branch and Cut algorithm within Branch and Bound [Man87]. In the Branch & Cut algorithm, when a problem is submitted to CPLEX for MIP (mixed integer program) optimization, CPLEX builds a tree at the root with the continuous relaxation of the MIP problem. The algorithm solves a series of continuous subproblems in which each subproblem is a node in this tree.

In the case of a relaxation with fractional variables, CPLEX will look for cuts, defining "cuts" as constraints that cut away areas where fractional solutions (non-integer solutions) which would be the solutions in case the problem is in its continuous relaxation form. Cuts are added as a way to reduce the branches number that are needed to solve the MIP problem.

After CPLEX tries to add cuts, the subproblems might still result in a fractional solution again, in an infeasible solution, or in an all-integer solution. If the relaxation solutions still contain one or more fractional values, CPLEX then executes Branch and Bound algorithm, such that it branches on the fractional variable to generate two new subproblems. For the newly generated subproblems, more restrictive bounds will be placed on each of them on

the branching variable [Man87]. For example, in a problem with binary variables, one node might set the variable to 1, and the other sets it to 0.

D-Wave - Simulated Annealing

D-wave's Simulated Annealing Sampler implements the simulated annealing algorithm that simulates the cooling technique of metals from a high temperature to a cooler temperature with the purpose of improving its structure [PB21]. The idea behind translating this analogy to the simulated annealing process is by doing random walks in the problem's landscape and generating random points in the neighborhood or the vicinity of the current best point, and then calculate the problem's cost at this point, if its cost is better than the older point, then the older point is updated with the new point. The algorithm is described in more detail in Chapter (1).

D-Wave - Advantage Quantum Solver

D-Wave's quantum solver, in both the Chimera and Advantage solvers, implements the Quantum Annealing algorithm physically. Quantum annealing can be compared to simulated thermal annealing, such that the temperature parameter is analogous to the tunneling field strength in Quantum Annealing, as QA process exploits the quantum tunneling phenomenon that happens on the quantum scale. The Quantum Annealing algorithm is described in more detail in Chapter (1).

D-Wave - Hybrid Solver

The main idea behind solving a problem using D-Wave's Hybrid solver is that the solver loops through parallel solvers to find an optimal solution [Sys]. In the default setting, the algorithm implements four parallel solvers, each branch generally consists of: decomposer, sampler, and composer. Tabu Search runs in the top branch on the entire problem, until another branch interrupts it by completing first. D-Wave's quantum sampler runs in the second-highest branch, such that, the hybrid algorithm allocates the pure Quantum solver to the parts of the problem where it benefits most. In addition, the algorithm allows for a

user-defined criterion as well as shown in the last branch. Due to the different parts of the algorithm, the Hybrid solver can accommodate even large problems.

2.3.2 Testing Approach

First, to evaluate the suggested mapping, the native QKP and the reformulated QUBO are both solved classically via the standard branch-and-cut optimizers available via CPLEX. However, we note that solving QUBO on CPLEX is highly inefficient as the runtime can take up to hours and even days for large problems. Experimental evidence in [MW13] shows that solving three instances in the NP-hard problem domain on the D-Wave quantum computer system outperformed the CPLEX of IBM by 3600x times faster. Therefore, we put a time limit when we solve large QUBO problems.

Second, the QKP-QUBO reformulated instances are solved by the classical simulated annealing solver, D-Wave's pure quantum solver, and the Hybrid solver. Then a computational comparison is made between the native QKP instances optimal solutions and the reformulated instances obtained solutions solved by both the classical and the quantum solvers, to further investigate the suggested mapping, evaluating the quantum and the hybrid solvers performance and to investigate relationships between the problem structure and its complexity.

For solving QKP original instances on CPLEX, we use the original formulation in Eq. (2.1). Furthermore, For solving QUBO instances on CPLEX, we use the following classical mixed-integer linear programming (MILP) formulation of QUBO problem:

$$
\begin{aligned}
\text{Min } & \sum_{i<j} Q_{i,j} y_{ij} + \sum_{i}^{n} Q_{i,i} x_i \\
& \text{subject to} \\
& y_{ij} \leq x_i \\
& y_{ij} \leq x_j \\
& x_i + x_j y_{ij} \leq 1 \\
& y_{ij} \geq 0 \\
& \forall j < j, x \in \{0,1\}^n
\end{aligned}
\tag{2.7}
$$

For any fixed i, j, the constraints in Eq. (2.7) (called Fortet inequalities [For60] or McCormick inequalities [McC76]), which force y_{ij} to equal $x_i x_j$.

Finally, the same QUBO instance is solved on D-Wave's simulated annealing, Advantage, and Hybrid solver.

2.4 Results and Discussion

All the numerical results represented in this section are included in Appendix A.

2.4.1 Results (1) - Classical comparison between the native QKP and the mapped QKP-QUBO on CPLEX

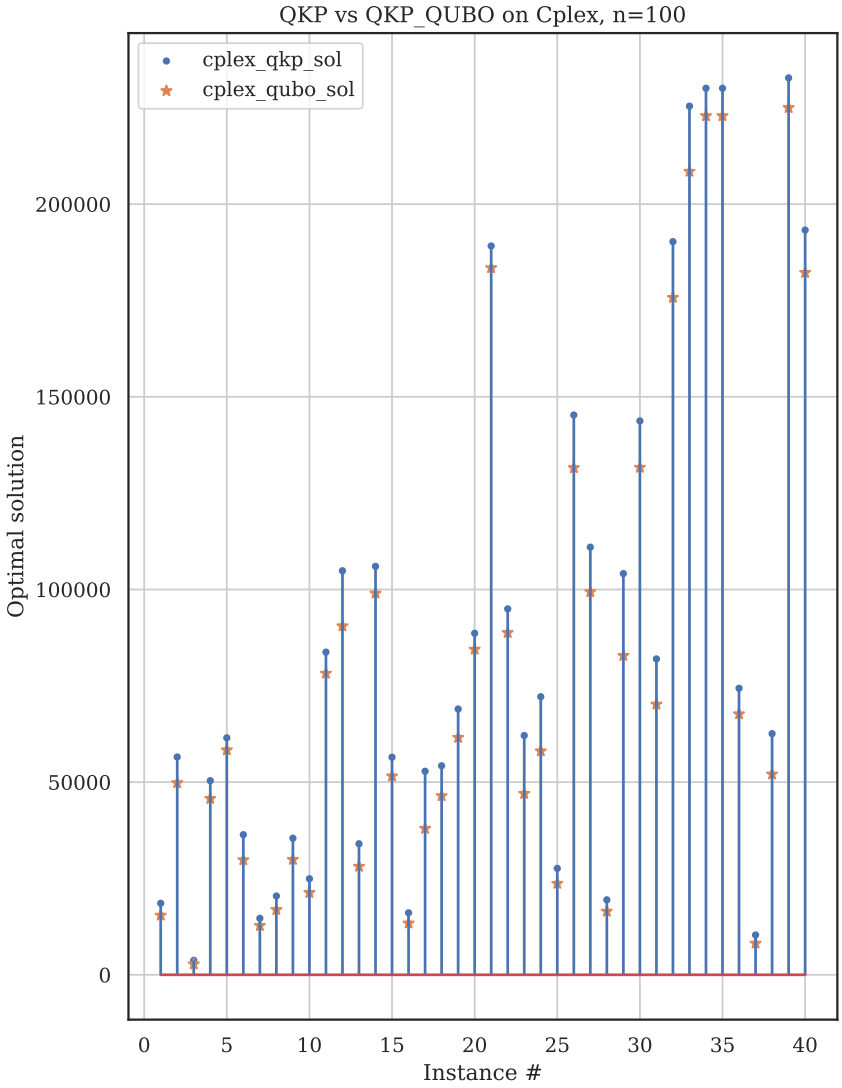

Figure 2.2: Classical comparison between the native QKP and the mapped QKP-QUBO on CPLEX

In Fig. (2.2), the native QKP and the reformulated QUBO are both evaluated on the classical solver CPLEX. The x-axis represents the index of the benchmark instance in [BS04], and the values on y-axis represents the optimal cost solution.

To judge the solutions quality, we are comparing the estimated optimal solutions against the actual optimal solutions obtained by the CPLEX QKP solver, we calculate the mean absolute percentage error, which we call the Average error gap.

$$\text{Avg. Err. gap} = \frac{100\%}{2} \sum \frac{|actual - estimated|}{actual} \tag{2.8}$$

The error gap is approximately 12% in Fig. (2.2) between the native QKP and the reformulated QUBO on CPLEX.

According to other computational studies in the literature, [Das13], solving QUBO is very difficult on CPLEX; it can take up to hours and even days to get an exact solution that is guaranteed to be optimal. In the testing code, we had a time limit to solve QUBO on CPLEX, so the optimal solution wasn't guaranteed; hence there was an error gap between the native QKP and the QUBO on CPLEX. However, to test if the only reason for getting an error gap is the time limit, we generated small-sized instances with the same benchmarking instances structure and removed the time limit; both the native QKP and QUBO produced the same optimal solutions except for some instances. We traced the error and found that the exact algorithm on CPLEX produces this error when the knapsack capacity is very close to the weight of selected objects at the boundary condition, and the objects are nearly equally heavy. We selected one of the failed instances to demonstrate this failure:

The objective matrix

$$\begin{bmatrix} 0 & 68 & 29 & 81 \\ 68 & 3 & 0 & 13 \\ 29 & 0 & 0 & 0 \\ 81 & 13 & 0 & 0 \end{bmatrix}$$

subject to the knapsack constraint

$$20x_0 + 35x_1 + 47x_2 + 48x_3 \leq 54 \tag{2.9}$$

When this instance is solved in its native QKP form on CPLEX, the optimal value found is **3**, such that only x_1 is selected. However, when the instance is transformed to QUBO form and solved on CPLEX, the optimal value found is **129**, where the objects x_0 and x_1 are selected $(20 + 35 = 55)$, violating the constraint value (54) by a value of 1.

2.4.2 Results (2) - Penalty's choice impact comparison between the mapped QKP-QUBO on CPLEX and D-Wave

The QUBO form of the same failed instance in Eq. (2.9) is also solved on D-Wave. Surprisingly as shown in fig (2.4), the instance has chances of success when it is solved using the simulated annealing algorithm. We find that the algorithm has an average success probability of finding the correct cost solution seven times out of 100 run times. However, it still displayed a failed solution, as shown in Fig. (2.3).

```
Lowest energy:  -29289.0
Cost solution:  129.0
   0  1  2  3  4  5  6  7  8  9   energy num_oc.
0  1  1  0  0  0  0  0  0  0  0 -29289.0      11
1  0  1  0  0  1  1  0  0  1  0 -29163.0       9
2  0  0  0  1  0  1  1  0  0  0 -29160.0      12
3  0  0  1  0  1  1  1  0  0  0 -29160.0      10
4  1  0  0  0  0  1  0  0  0  1 -29160.0      12
5  1  0  0  0  1  0  0  0  0  1 -29150.0       2
['BINARY', 6 rows, 56 samples, 10 variables]
```

Figure 2.3: Energies sample using simulated annealing, with the lowest energy = -29289 and the cost solution = 129

```
Lowest energy:  -29163.0
Cost solution:  3.0
   0  1  2  3  4  5  6  7  8  9   energy num_oc.
0  0  1  0  0  1  1  0  0  1  0 -29163.0      10
1  0  0  0  1  0  1  1  0  0  0 -29160.0      11
2  1  0  0  0  0  1  0  0  0  1 -29160.0      11
3  0  0  1  0  1  1  1  0  0  0 -29160.0      13
4  0  0  0  0  0  1  1  0  1  1 -29160.0       9
5  1  0  0  0  1  0  0  0  0  1 -29150.0       2
['BINARY', 6 rows, 56 samples, 10 variables]
```

Figure 2.4: Energies sample using simulated annealing, with the lowest energy = -29163 and the cost solution = 3

The same instance was tested on the chimera pure quantum solver on D-wave; the success probability for the chimera solver was found to be 3%, and does not improve with increasing the penalty value. The hybrid solver is not included as it's not efficient for small sized problems [Sys].

These discrepancies can be traced back to the effect of the penalty P scalar value has on the computational efficiency. Suppose the scalar penalty P value is too large. In that case, it can disrupt the computation process since the penalty terms can overwhelm the original objective function, making it hard to determine the quality of one solution over another. Alternatively, a penalty value that is too small compromises the search for feasible solutions [GKD19]. However, deciding on what penalty values are appropriate may be problem-specific, and they are not unique. For the example shown in Fig. (2.3), the penalty has the value of **10**, which has a success probability of **7%** when solved using the simulated annealing algorithm. However, this success probability keeps increasing as we increase the penalty value. For example, when we increased the penalty to the value of **50**, it had a success probability of **99%**, suggesting that the penalty value indeed impacts the quality of the computational efficiency.

However, the CPLEX optimizer never gets the correct optimal solution except for only one penalty value which is found to be **136**.

2.4.3 Results (3) – Evaluating Density vs Runtime

Second, we want to test whether the argument made in the literature that there is a correlation between the matrix density and difficulty.

In Fig. (2.5), (2.6), and (2.7), the relationship between the native QKP density and the runtime for different solvers is shown. There was no relationship found using the pure quantum Advnatge solver on D-wave; also, the runtime was very long with an average of 330s.

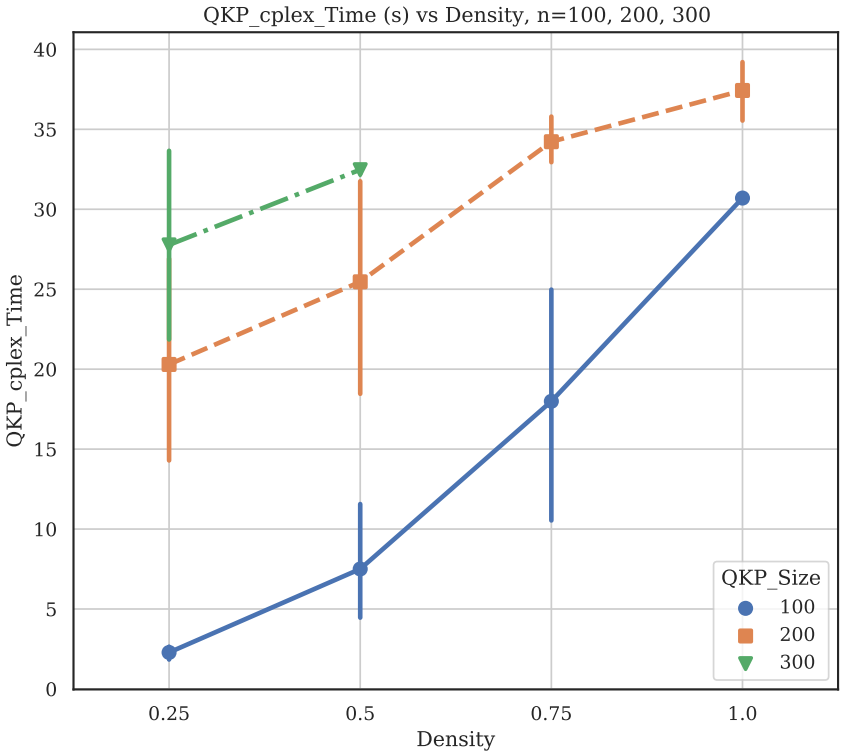

Figure 2.5: QKP native problem run time on CPLEX vs density, n=100, 200, 300

Fig (2.5) shows a positive correlation between the QKP native problem density and it's difficulty.

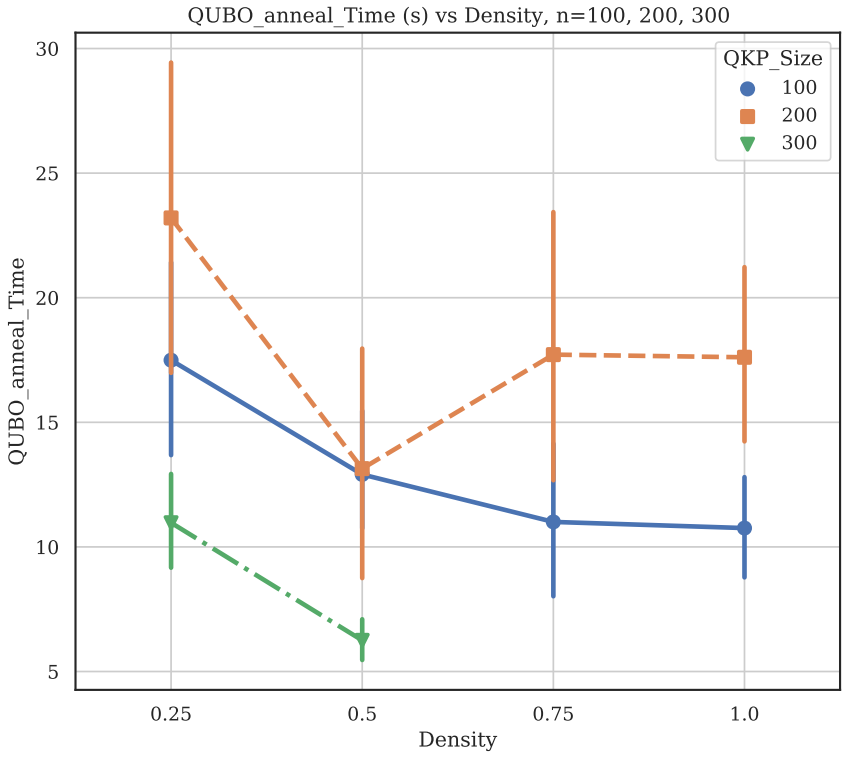

Figure 2.6: QUBO anneal time for simulated annealing vs original QKP Density, n=100, 200, 300

We observe something interesting here in fig (2.6), the QUBO anneal time for simulated annealing decreases with increasing the native problem's density for the problem size $n = 100$ and 300. However, in $n = 200$, we observe that the runtime decreases for $d = 0.5$, but goes up again for $d = 0.75$. We also note that, even though there is a general trend of decreasing the runtime, the spread in some the error bars in Fig. (2.6) overlaps with other data points error bars. Hence, we should be careful while drawing the conclusion that the runtime "always" decrease with increasing the density of the native problem. We also found a similar trend for $n = 200$ and $n = 300$. It seems that the original problem's density somehow influences the difficulty of the reformulated QUBO, even though, whether the original problem is sparse or dense, the reformulated QUBO is always fully dense. This result implies a negative relationship between the hardness of the native QKP problem

and the reformulated QUBO problem. This not intuitive result suggests that further investigation is needed.

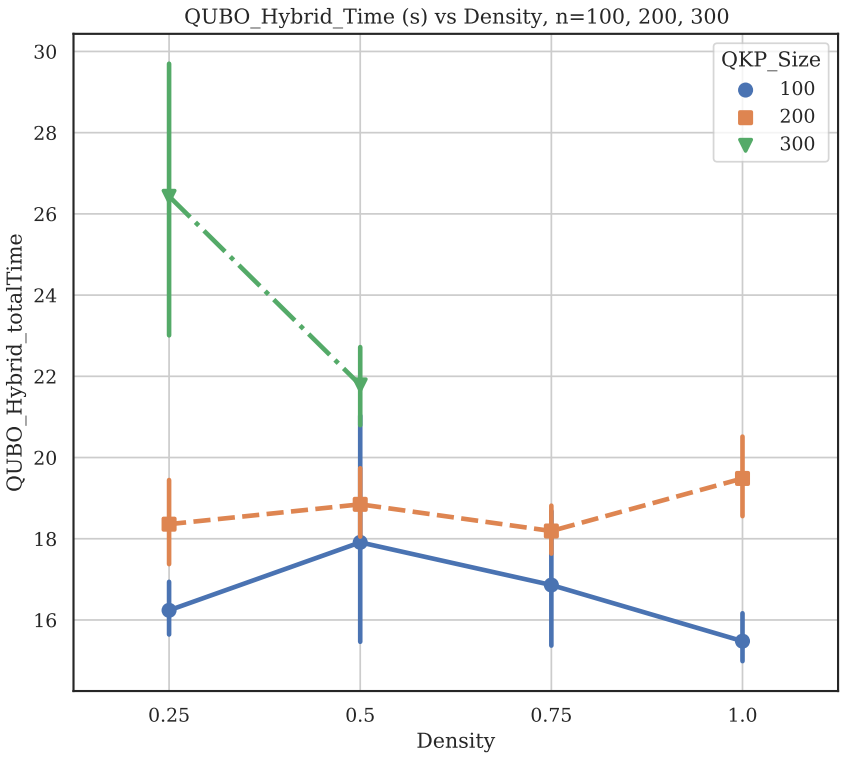

Figure 2.7: QUBO anneal time for Hybrid solver vs original QKP Density, n=100, 200, 300

2.4.4 Results (4) – Solution quality comparison between all solvers

In the first run of the instances on all solvers, in Fig. (2.8), we used penalty value = 10. The x-axis represents the index of the benchmark instance in [BS04], and the values on y-axis represents the optimal cost solution according to the different solvers shown. As shown in Tables (2.1)-(2.3), we compare between all solvers in terms of the average error gap and the average runtimes.

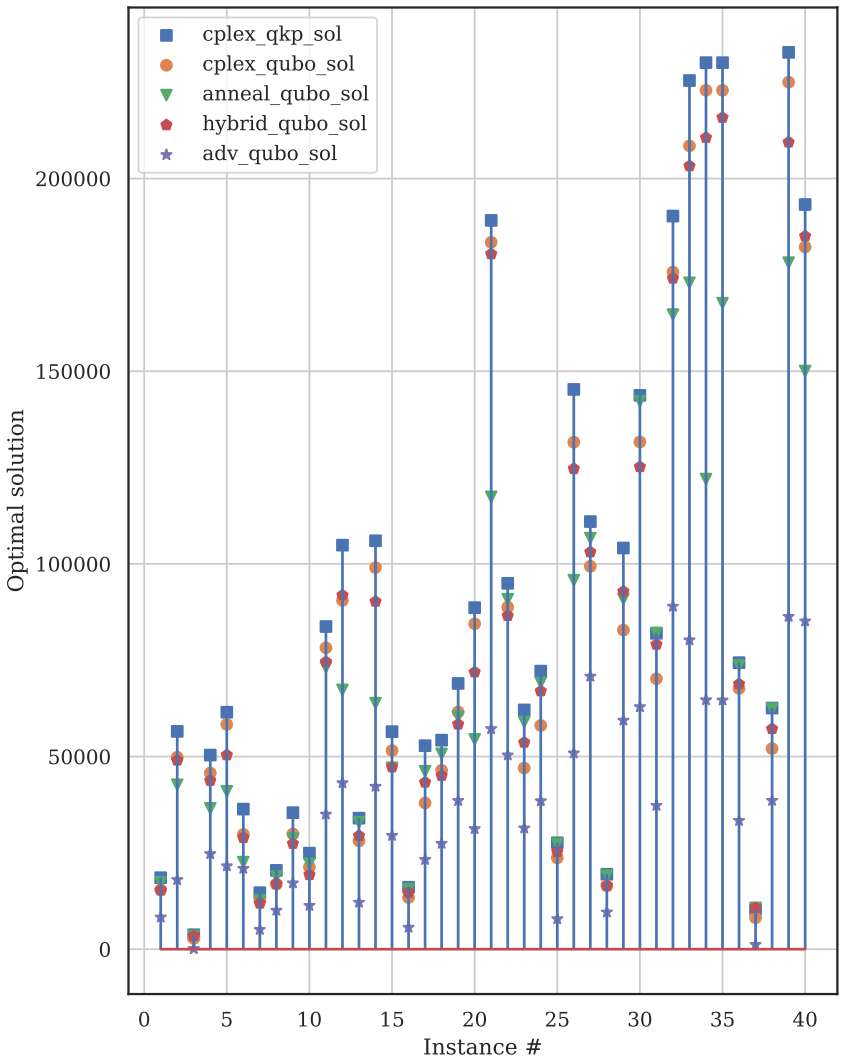

Figure 2.8: Solution quality comparison between all solvers, P=10, n=100

Table 2.1: Comparison between solvers in terms of the Average Error gap (Avg. Err. gap) and Avg. runtime, n=100, P =10

	Avg. Err. gap (%)	Avg. runtime (s)
QKP_CPLEX	0	14.62 (time-limit is applied for for d=100%)
QUBO_CPLEX	12.01	34.20 (time-limit)
QUBO_SA	15.74	13.04
QUBO_Hybrid	12.10	16.62
QUBO_Adv	66.72	329.10

From our initial testing, we found that the pure quantum solver (Advantage system) was doing bad in all testing cases, so its results have been eliminated in the following tables.

Table 2.2: Comparison between solvers in terms of the Avg. Err. gap and Avg. runtime, n=200, P =10

	Avg. Err. gap (%)	Avg. runtime (s)
QKP_CPLEX	0	29.35 (time-limit)
QUBO_CPLEX	12.01	47.30 (time-limit)
QUBO_SA	20.46	17.91
QUBO_Hybrid	9.18	18.72

Table 2.3: Comparison between solvers in terms of the Avg. Err. gap and Avg. runtime, n=300, P =10

	Avg. Err. gap (%)	Avg. runtime (s)
QKP_CPLEX	0	30.12 (time-limit)
QUBO_CPLEX	12.01	47.14 (time-limit)
QUBO_SA	20.47	8.61
QUBO_Hybrid	10.00	24.11

After we found that the quality of the solution can improve by increasing the penalty value in result (2), we did another run on the instance with a larger penalty value; however,

we found the opposite, that in fact, the solution becomes worse with increasing the penalty on these instances, and instead, the solution quality improved with decreasing the penalty value.

For the third run of the instances on all solvers Fig. (2.9), we used penalty value = 4, in Tables (2.4)-(2.6) we compare between SA and Hybrid solvers in terms of the average error gap and the average runtimes.

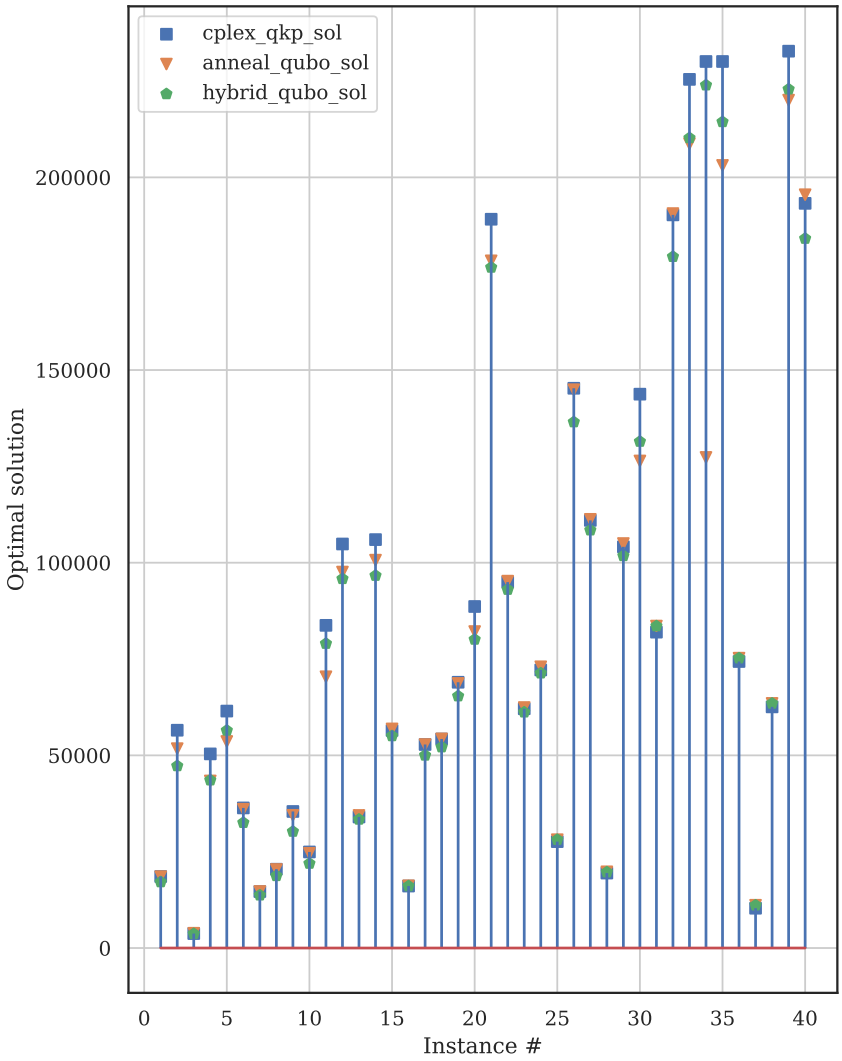

Figure 2.9: Solution quality comparison between all solvers, P=4, n=100

Table 2.4: Comparison between classical SA and Hybrid solver in terms of the Avg. Err. gap and Avg. runtime, n=100, P =4

	Avg. Err. gap (%)	Avg. runtime (s)
QUBO_SA	3.48	63.99
QUBO_Hybrid	4.93	12.94

Table 2.5: Comparison between classical SA and Hybrid solver in terms of the Avg. Err. gap and Avg. runtime, n=200, P =4

	Avg. Err. gap (%)	Avg. runtime (s)
QUBO_SA	2.56	20.28
QUBO_Hybrid	2.45	17.29

Table 2.6: Comparison between classical SA and Hybrid solver in terms of the Avg. Err. gap and Avg. runtime, n=300, P =4

	Avg. Err. gap (%)	Avg. runtime (s)
QUBO_SA	9.97	53.47
QUBO_Hybrid	3.16	17.03

2.5 Conclusion

In this project, the performance in terms of the quality and runtime of the solution was compared for solving QKP reformulated QUBO of the hybrid solver on four different solvers: CPLEX exact solver, D-Wave's simulated annealing algorithm, pure quantum solver, and Hybrid Solver. Also, these solvers' response to the change in the parameters of QK-QUBO reformulation is investigated.

There are few important insights we can draw from the results found in Sec. (2.4). First, in result (1), when the native QKP problem is solved classically on CPLEX and compared to the formulated QKP-QUBO problem on CPLEX as well, we found that solving the reformulated problem using the CPLEX exact algorithm sometimes fails for instances that

have knapsack capacity very close to the weight of selected objects, suggesting that the solution is sensitive to the boundary conditions of the problem; also, the objects are nearly equally heavy in the failed instances. Second, in result (2), the discrepancies found due to the choice of the penalty values suggest that the penalty value does indeed impact the quality of the computational efficiency.

In result (3), we can see that there is a positive correlation between the QKP native problem density and its difficulty. According to the literature, research on developing a methodology to generate QKP instances with predictable and consistent difficulty levels is ongoing. However, the studies in [HM09], and [For97] include a discussion of the QKP problem difficulty, and both studies conclude that there is a positive correlation between the density of objective function coefficients and CPU time, which supports our initial results.

Next, we observe something interesting in fig (2.6): the QUBO anneal time for simulated annealing decreases with increasing native problem density. We also found a similar trend for n=200 and n=300. It seems that the original problem's density somehow influences the difficulty of the reformulated QUBO, even though, whether the original problem is sparse or dense, the reformulated QUBO is always fully dense. This result implies a negative relationship between the hardness of the native QKP problem and the reformulated QUBO problem. This not intuitive result suggests that further investigation is needed. We also found a similar trend with the hybrid approach in fig (2.7), but the relationship is weakly correlated; however, this result is not found when we use the Advantage D-Wave solver. This result shows strongly in the simulated annealing approach but shows a weak relationship in the hybrid approach and an almost non-existent relationship using the Advantage Quantum solver. Hence, this suggests that this result is associated with the Simulated Annealing algorithm, as simulated annealing is partially used in the Hybrid solver algorithm, so the result is still showing in the Hybrid approach but not as strong as in only using the Simulated Annealing approach. From this result, we can conclude that QUBO reformulation might influence the structure and the landscape of the original QKP problem when transformed and solved using the SA and the Hybrid solver approaches. The next step would be investigating how does the density of the native problem influences the transformed QUBO.

Result (4) answers the main question of our project. The results in Tables (2.1)-(2.6) suggest that the Simulated Annealing and hybrid solver outperform both the pure quantum solver and the classical implementation on CPLEX. However, it's found that D-Wave's Hybrid solver does slightly better, in terms of the solution quality, by $2\times$ times, than the simulated annealing with the growing size of the QKP problem when $n > 100$, and the pure quantum Advantage solver does worse in all cases. Hence, it's concluded that Quantum Annealing can display quantum speed-up in the Hybrid solver case. This means that if QA-Hybrid scaling advantage is due to quantum effects in the current generation of the highly noisy quantum annealers, then we speculate the future generation of quantum annealers with less implementation errors will at least achieve the same (or even more) performance of SA for this particular class of problems.

To explain why the Advantage solver is worse in all cases, we think that with the limited development of the current hardware, it is able to solve small instances on the chimera topology with better quality; however, to solve larger instances on the Pegasus topology, which is more complicated than chimera, the software takes up a long time to find a suitable embedding for these larger instances (> 90). To improve the embedding times, good decomposition algorithms and efficient embedding algorithms are needed. Also, the thermal fluctuation in the current hardware might be causing it to be incoherent affecting its solutions quality, hence, making it challenging to observe any noticeable advantage over other algorithms.

Finally, we seek to understand some of the results further for future work. First, regarding the non-intuitive density-runtime relationship we get in result (3), the next step would be to investigate how the density of the native problem influences the complexity of the transformed QUBO. Second, as the Hybrid approach showed potential for better quality solutions and runtime for solving QKP-QUBO, the next step would be developing our defined criterion in its algorithm to improve its performance. A future study that might potentially provide increased performance, is relating the formation of the minimum energy gap g_{min} (described in chapter 1) to the structure of the QKP problem, and using this insight in designing a better algorithm on the Hybrid solver.

Chapter 3

Quantum Annealing approach to Blockchain Consensus Mechanism

3.1 Overview

Proof-of-Work (PoW) is a consensus mechanism commonly used in blockchain technology due to its robustness. However, although PoW is highly robust against misbehavior on the blockchain, the enormous amount of energy spent by PoW is becoming an overwhelming problem. We propose an alternative approach for the blockchain PoW by considering a wide range of real-world NP-hard optimization problems, mapped into quadratic unconstrained binary optimization (QUBO) problems. Our approach joins "Proofs of Useful Work" schemes, where we solve PoW challenge into a useful resource. Finally, we study the potential of Quantum Annealers as solvers for these NP-hard optimization instances. Numerical experiments in this study test the difficulty adjustability of the QUBO problem to investigate if the suggested problem satisfies the criteria for a practical PoW challenge that could be used in the blockchain network.

3.2 Introduction

Since the introduction of the cryptocurrency Bitcoin in 2008, blockchain has become a tempting new technology; because of its promising security, transparency, and decentralization that can be beneficial for various businesses to integrate blockchain within their applications [YMRS18].

The most widely used consensus protocol for the public (or permissionless) blockchain is the Proof-of-Work (PoW) [YMRS18]. While PoW is very robust against misbehavior on the blockchain network, the enormous amount of energy spent by PoW is becoming an overwhelming problem. At the time of writing this chapter, Bitcoin alone consumes around 132 TWh of energy over the course of a year [bit21], which is more than the country of Norway's energy consumption. Several alternatives to the PoW with better energy efficiency have been proposed; however, some of these alternatives exhibit at least a single flaw [WHH+19], and that is why most of the current cryptocurrencies based on PoW are still dominant.

Our proposed approach involves formulating a PoW challenge based on hard optimization problems, aiming to have this wasted energy used for useful purposes. Consequently, this approach requires an efficient optimization algorithms that can serve as solvers for these problems. Heuristic algorithms are the most attractive class of algorithms to solve optimization problems [ZE81]. Unlike exact algorithms, heuristic algorithms do not spend an exponential amount of time searching for the optimal solution that can take up to hours or even days when large real-world problems are considered. Heuristic algorithms can run in polynomial time according to the size of the input and settle on solutions near the optimal in an acceptable amount of time. In this chapter, the metaheuristic algorithm, Quantum annealing, is considered to be used for the proposed approach.

The proposed protocol is not restricted to one family of optimization problems; we take advantage of the ability to map a vast number of optimization problems into QUBO problems. We formulate a PoW challenge based on hard optimization problems and map it into the QUBO problem. D-Wave's Quantum Annealing sampler is then used to 'mine' for the best configuration of this problem to solve the PoW challenge. The solution is then verified simply using classical resources to test whether it satisfies the given constraint in

the challenge; the block is then added to the blockchain. There are many hard optimization problems that can be appropriately embodied into the QUBO model. This study only considers the Quadratic Knapsack problem (QKP) in the NP-Hard complexity class. However, this approach is not limited to the QKP and can be extended to any optimization problem that can be mapped into QUBO and satisfies the criteria for a practical PoW challenge. A practical PoW would require to satisfy the hardness property mainly, such that, the PoW challenge should be quickly generated, and the challenge should achieve the desired difficulty. The criteria for a practical PoW are explained formally in Sec. (3.3). Finally, the solver is then expected to spend a certain amount of (non-reusable) work in producing a solution that is easily verified in polynomial time [WHH+19].

This chapter is organized as follows: Sec.(3.3) gives a background on blockchain and consensus protocols. In Sec. (3.4), we discuss the related work found in the literature, and our contributions. In Sec. (3.5), the proposed protocol scheme and the used algorithms are discussed. In sec (3.6), QUBO as PoW challenge is evaluated according to the baseline properties for a good PoW challenge. Finally, we conclude in Sec. (3.7).

3.3 Background

Blockchain is an immutable, distributed ledger that facilitates recording transactions in a peer-to-peer (P2P) network. As the name suggests, blockchain is a series of blocks containing the information associated with the recorded transaction. Usually, each block in the blockchain includes the transaction's ID, timestamp, quantity, and the status of the transaction, and the blocks are securely linked to each other using cryptographic functions like SHA-256 hashing, and elliptic curve cryptography (ECC) [YMRS18]. Furthermore, each block consists primarily of the previous block's hash and a list of all transactions to prevent any block from being changed or inserting a block between two already existing blocks. Thus, each new block added to the blockchain reinforces the validation of the preceding block, thus making the blockchain safe from tampering. This prevents the possibility of malicious agents and supports the creation of a ledger of trusted transactions from participants of the entire network [YMRS18]. Although, this brings the core strength of the concept of trust in the blockchain, it cannot be considered entirely immutable as

often misunderstood, as there exist attacks to which the blockchain is prone and could be modified[YMRS18].

Four steps must happen to add a new block to the blockchain:

1. A transaction must occur, like a purchase or money transfer.

2. This transaction must be verified. There is no single entity specializing in verifying transactions, as is the case with banks. This task is handled by a network of thousands (or even millions) of computers in the blockchain case. When a participant makes a purchase, this network will verify that the transaction data is valid. There are different ways to achieve this transaction verification process; For a block to be verified, the blockchain network participants must achieve a consensus among themselves to agree on the transaction's validity. The most currently used consensus algorithms in blockchains are the Proof-of-Work (PoW) and the Proof of Stake (PoS). Many other consensus algorithms are proposed and used; however, each comes with its advantages and disadvantages. This section explains the two consensus protocols and focuses on the most common one, PoW.

3. Store the transaction in a block. After verifying that the transaction data is correct, the transactions are stored in a single block.

4. Mark that block with a unique hash value. After verifying all the block transactions, the block is given a hash token identifying it as the latest block added to the blockchain (that is, this block identifies the previous block, and the previous block also knows the previous block, so all blocks are connected as a blockchain). After the block is given a special token, it can be added to the blockchain.

Fig. (3.1) shows the basic design structure of the blockchain network. Each block consists mainly of the block's transactions and the block's header, containing information like the block's id, timestamp, and block's size. The Merkle root field in each block represents the hash value of all hashing values of the current block's data. The next block's hash is combined with the hash value of the previous block for tracking and verifying each block. This method is called Merkle Tree hashing, and it is commonly used to secure distributed systems and P2P networks.

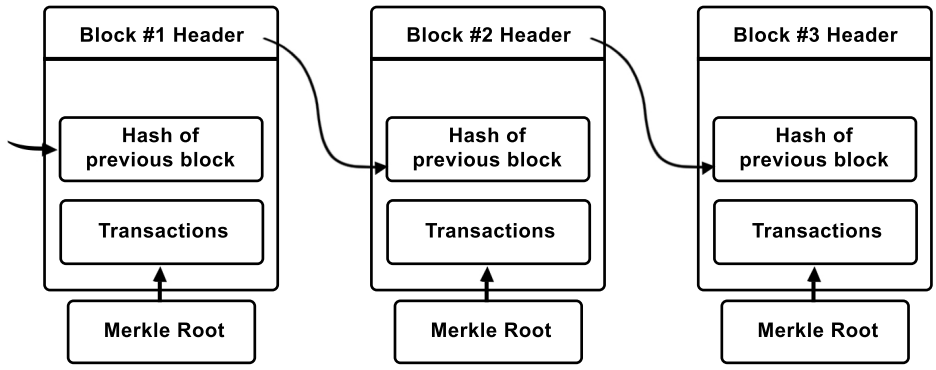

Figure 3.1: Basic design structure for blockchain.

3.3.1 Consensus protocols in Blockchain

The goal of the consensus algorithm in a blockchain network is to allow multiple users to agree on the state of the blockchain even though they do not trust each other or any central authority. The two main types of consensus algorithms that are widely used are PoW and PoS; however, both have some fundamental differences:

(a) PoW is the consensus algorithm that Bitcoin uses to secure the network through a process known as "mining". Mining requires specially designed hardware to solve a difficult mathematical problem. When the problem is solved, a block is generated and the "miner" is awarded for solving the mathematical problem for that block. In the case of the Bitcoin blockchain, the reward is the creation of a new Bitcoin. This reward is compensation for the resources required to solve the problem. The reward in the PoW algorithm is usually a fixed amount per block, but it can also vary over time. In the case of Bitcoin, the block reward is reduced every four years. The reward is determined in advance by the system.

(b) PoS is a consensus algorithm for validating existing transactions and achieving distributed consensus. Unlike the PoW algorithm, which requires the user to perform a certain amount of computational work (mining) to validate transactions, the Proof-of-Stake algorithm requires the user to prove their ownership of a certain amount of

currency or "stakes". The PoS algorithm depends on the strength of the wallet that the participant owns, and it also depends mainly on the time duration of the coins remaining in the participant's wallet. Hence, the "mining" difficulty in the PoS algorithm is determined by the amount and time of the tokens occupied by each node. Accordingly, the mining difficulty is reduced or increased in proportion to each node's ownership strength. This means the system naturally disadvantages small nodes, and the wealthiest nodes have control of the network.

In the usual PoW algorithm, the miner is asked to solve a hard cryptographic problem based on a Hash function $H(x)$, which maps to the input x with any size to the hash value y with a fixed-sized output. This function is designed so that y can be easily computed from the input x if $H(x) = y$, but finding x from y is computationally difficult.

In the Bitcoin case, the miner spends computational resources to find a is a 32-bit number that is called "nonce" n, such that, when this nonce is hashed with the block's header h using the SHA-256 hashing function, the output should be less than or equal to the "target" τ value specified by the network. According to the network requirement, the difficulty of finding this nonce can be adjusted by tuning the target value τ.

$$Hash(n) = \text{SHA-256}(h|n) \leq \tau \tag{3.1}$$

The main advantage of PoW is that it's incredibly robust; however, one of its weak points is that it wastes electricity and computational resources. The use of hash puzzles in blockchain systems allows to solve two main problems: to cryptographically bind transactions to a block and to provide transactions with computational work to reach consensus among users of the blockchain. The main drawback of hash puzzles is that they are useless outside of blockchain systems. Various alternatives to PoW have been proposed to address this problem. However, although many of these alternatives offer better energy efficiency than PoW, some of them still introduce some degree of failure. For this reason, Bitcoin and other PoW-based cryptocurrencies are still dominant, as they are robust and offer unbeatable security. This chapter proposes an alternative to the blockchain PoW, where a hard NP optimization problem replaces the hash puzzle.

In the next section, the alternative scheme that uses useful problems is proposed, but first, we need to discuss the properties of a good PoW challenge in the blockchain as we try to meet its criteria while we design our protocol. We find the criteria used in the studies [Din19] and [Sho17] to be encompassing for the essential properties a good PoW challenge must-have.

1. Intrinsic Hardness

   This property requires that a certain amount of computational work be spent to find the problem's solution, and the problem difficulty should reflect on solving the challenge with no "easy" way of solving it.

2. Adjustable Hardness

   This property requires the problem difficulty to be tunable, and it could be decreased or increased according to the network requirements.

3. Easy Verifiability

   This property requires the problem's solution to be easy to verify; hence, reaching a consensus between the blockchain's participants.

4. Block Sensitivity

   This property requires the problem to be unique and only tied to one block; using the same problem for several blocks is prohibited as it tampers with the blockchain's security.

5. Non-reusability

   This property requires that the computational work done on one problem cannot be reused on another as this gives an advantage to one node over the other.

In section (3.6), we compare our alternative protocol against these properties.

3.4 Related Work

Few studies in the literature suggested PoW based on optimization problems; a study done in [Shi19] proposed a consensus protocol for the blockchain, named proof-of-search (PoS). Their protocol allows any node (called client) to submit optimization problems via the blockchain. Furthermore, any client can submit a job to find a solution to an optimization problem. The clients in this protocol can submit an instance of the problem and receive the found solution, rewarding the node that finds the best approximate solution. However, if clients submit no job, the protocol falls back to the basic PoW. Motivated by Shibata's protocol, in our study, we are proposing a more efficient protocol. Our protocol would not have empty jobs; instead, randomized optimization problems, modeling real-life problems, are generated, and their solutions are stored. Also, Shibata's protocol demands the client to meet specific requirements, like submitting an 'Evaluator' program and a 'Searcher' program for the particular submitted optimization problem. However, in our protocol, we propose a more automated network by mapping a large set of various optimization problems into one model. Thus, it was immediately natural to think of QUBO.

The second study in [SDK19] used the Traveling Salesperson Problem as a model and looked at using optimization algorithms as a method for providing the PoW required to add a new block to the blockchain. Their basic idea is to add one extra city to the best cities path found for block n to satisfy blockchain requirements for adding a new block $n + 1$. Their protocol also allows for designing limited blockchains while solving the underlying combinatorial optimization problem.

Our contribution would be generalizing the protocol to work with various types of optimization problems mapped to the general QUBO problem. Experimental results on the D-Wave machine support the study. In particular, we are testing one of the hard optimization problems (QKP) as one of the potential candidates for a good PoW problem. The QKP problem is mapped to the QUBO model to be implemented on the D-Wave sampler, and evaluating it QUBO according to the criteria of a good PoW protocol.

A study done by Kalinin and Berlo in [KB18] is worth mentioning, as we were not the first to think about using quantum resources for this approach. Their study provided a general framework of the blockchain PoW based on analog Hamiltonian optimisers (Quantum

Annealers, OPO-based simulators, Gain-Dissipative Simulators..etc) with no implementation on an actual Hamiltonian optimiser machine and with no consideration of how QUBO problem would satisfy the blockchain consensus protocol requirements for a good PoW problem.

3.5 Proposed Protocol

In this section, we describe our enhanced protocol, which builds on the PoS protocol in [Shi19], and enhances some of its parts. However, before we move to describe our protocol, we discuss briefly the PoS protocol in [Shi19], its working mechanism, and some of its security considerations and challenges.

3.5.1 Proof-of-Search (PoS) With Optimization Problems

The working principle of PoS protocol in [Shi19], is close to that of PoW, however, instead of wasting power on finding hash puzzles in proof-of-work, PoS utilized this computational power in solving instances of optimization problems. PoS protocol allows any node (called *client*) to submit optimization problems to the blockchain. The client needs to submit a program called an *evaluator* that is included in a submitted *job*, which is used by the nodes (or miners) to evaluate the solution candidates of the problem to be solved. The nonce in the PoS protocol is created by concatenating a candidate solution and its evaluation value, rather than an integer as in PoW. In order for a node to generate a valid nonce, the evaluator must be executed several times to evaluate different solution candidates. In PoS protocol, the client provides a computer program *searcher*, which implements a randomized search algorithm, in order to facilitate the search for solution candidates. The nodes execute this searcher program, which is included in the jobs, and it's designed in a way that it calls the evaluator many times every time it finds a solution candidate.

Because the consensus process requires the generation of a large number of nonces, consequently, many solutions candidates must be evaluated. The protocol uses two methods of rewarding nodes separately to prevent collusion between nodes. There are two possible ways for a node to be rewarded: succeeding in adding a new block by finding a valid nonce

given a target value, or succeeding in finding the best solution to the optimization problem. In the first case, the miner is rewarded in the same way as in PoW. In the second case, the miner is rewarded the charge paid by the client. However, in case there was no job submitted by a client, then the blockchain automatically adds an empty job, which makes PoS fall back to the ordinary PoW protocol.

There is no need to protect against the reuse of past computation results in PoW. This is because computing the hash function is the only way to get a hash value, and the hash value is linked to the miner's ID and the last block. However, in PoS, it is necessary to ensure that an evaluation is performed each time a the hash value of a block is generated. The miners may try to reuse and share the results of computations by among multiple nodes, as the amount of computation required to calculate them can be significantly greater than this of hash values in PoW. To avoid this, the evaluation result is linked to the miner's ID and other data of the block. This is accomplished by giving the evaluator two arguments. The first argument contains all the hash values of all the items in a block, and the nonce is used as its second argument. Then the evaluator introduces a small amount of error in its output to force it to depend on the second argument. The PoS protocol requires that the algorithm for introducing this error has to be designed and implemented separately by each client.

Making each node's probability of winning proportional to its computational power spent on the job, it is possible to adjust the block-time as needed; this technique, however, presents the risk of increased fork occurrence since its probability is dependent on the block-time. To reduce the probability of fork occurrence without changing the block-time, the PoS protocol introduced the idea of 'miniblocks' between blocks. A *miniblock* contains a nonce, the mining node's ID, and the last block's hash value. As shown in Fig. (3.2), each node attempts to find a valid nonce for each miniblock whose hash value starts with a number of zero bits that corresponds to the job associated with the miniblock. When a node finds such a nonce, the node broadcasts the miniblock along with the found nonce, and it's rewarded accordingly if it succeeds in adding this miniblock. Upon completion of all the jobs in the miniblocks specified by the blockchain, the block is then added.

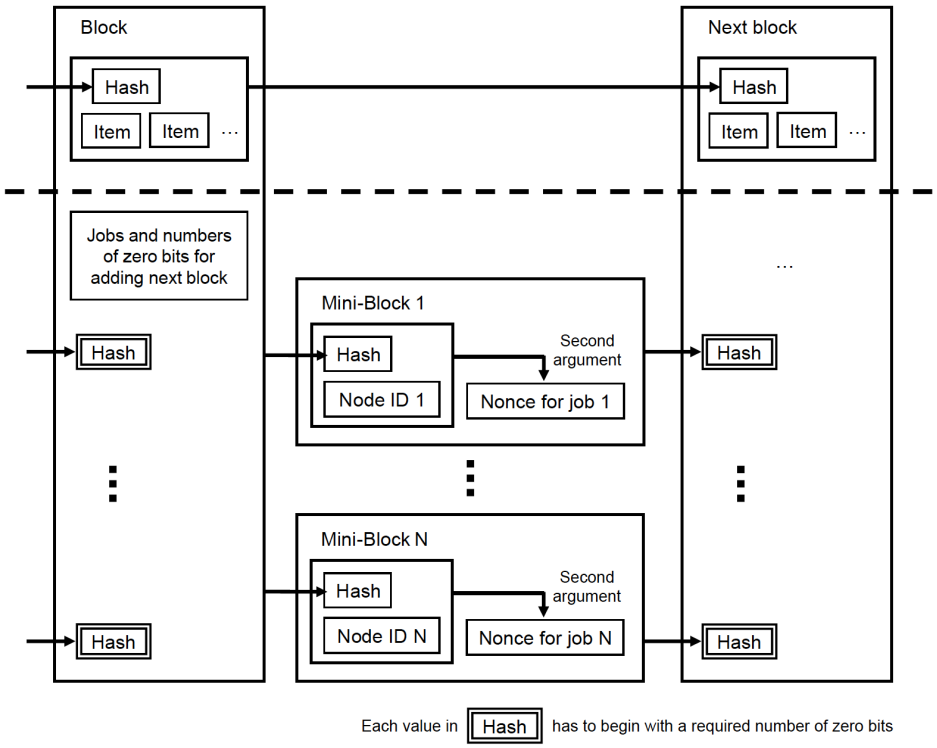

Figure 3.2: Distributed timestamp server in the PoS protocol. Reproduced from [Shi19]

Potential Attacks and their Resolutions

In the following paragraphs, we summarize some of the potential attacks addressed in [Shi19] against the PoS protocol.

1. The client submits a problem where they already know a good solution, and colludes with other miners to obtain the reward unfairly.

   Resolution:

   PoS rewards the miner for finding the best solution, but it also rewards the miner that finds a valid nonce that makes the hash value less than or equal to the target value specified by the network. This means, the miner essentially picks a random solution candidate and calculates the hash value. Hence, the problem and its true

best solution are irrelevant, so the colluding between the client and the miner has no advantage.

However, in case a client solves the problem before submitting it and tells the solution to a colluding miner, there will be no advantage doing so as well, as the client pays the colluding miner the reward. This is basically considered a regular transaction in a cryptocurrency, as the colluding client sends their coins to the colluding miner with no financial gain.

2. A client submits an infeasible problem instance that doesn't have a solution to force miners to spend work on these instances while colluding with other miners to work on tasks submitted by other clients.

   Resolution:

   If the submitted problem doesn't have a solution, the 'evaluator' will always return the same worst evaluation value for a problem instance that does not have a solution. In this case, this worst evaluation value will be considered the best solution if it's found by multiple miners with no improvement. Consequently, the reward will be divided equally between those miners, hence, discouraging the colluding client as they will have to pay other miners.

The rest of the potential cheats and attacks are discussed in [Shi19] in more detail, however, we specifically address the above scenarios as we think some certain aspects to them are not considered in [Shi19]. We discuss this further at the end of this section.

3.5.2 QUBO-based PoW Proposed Protocol

Our QUBO-based PoW protocol has two functionalities.

a. It allows any participant in the blockchain to submit any combinatorial optimization problem that can be converted into QUBO problem. The participant receives the found solution, and the system rewards the miner that finds the best approximate solution. The potential problem of a participant submitting a problem they solved

beforehand is discussed in Sec. (3.5.1), we also discuss this further in the end of this section.

b. If there is no problem submitted to the network, then randomized optimization problems are generated, modeling real-life problems, and their solutions are stored.

Although solving hard optimization problems that arise in real-life practical situations, is more advantageous than solving randomized problems; in case of a shortage of guaranteed useful tasks, the network allows the generation of hard optimization problems that are "close" to the structure of real-life problems, which, however, will lead to questions about what kind of problems structures that can model useful practical problems.

The implementation of task submission can be organized in different ways. We see three fundamental options:

1. Tasks are published by the blockchain itself (as smart contracts or transactions of a special type associated with one of the blocks),
2. Or a set of tasks of practical importance is created outside the blockchain by a separate service that creates and maintaining a database of tasks,
3. Or finally, the tasks are created and submitted by independent clients.

In all cases, a database of tasks should be formed that allows the selection of tasks based on an estimate of the average block-time to find a solution.

In the first case, assuming the blockchain is public, we allow the disclosure of tasks, after which they can be solved and the solutions can subsequently be used to form the blocks of the blockchain. In the second and third cases, such disclosure can be prevented, however, a potential problem with the second case would be the question of who will be the owner of the tasks database, and can lead to centralization of control. Hence, in our protocol, we only consider the first and the third case, which is similar to the protocol in [Shi19]. After adding the subsequent blocks, the solved problems and their solutions can be deleted from the database or marked as archived, which can be decided by the client or the service who sets the problem.

Our protocol consists of five algorithms:

1. The challenge generator, in Algorithm (1)
2. The challenge submitter, in Algorithm (2)
3. QKP (Or any combinatorial optimization problem family)-to-QUBO converter, in Algorithm (3)
4. The solver algorithm
5. The evaluator algorithm

Algorithms (1), (2), (3) are using classical resources and are expected to work in polynomial time, where Algorithm (4) is using a metaheuristic algorithm, mainly we are considering Quantum Annealing algorithm, and it should be difficult for exact algorithms on classical computers to solve. The evaluator Algorithm (5), follows the same ideas as in [Shi19], and is expected to work in polynomial time, however, this algorithm should be provided by the blockchain network and not by the client.

Algorithm 1: Challenge Generator

Input: NP problem family type, block-time T

Output: Block data + QUBO Matrix Q

1 Generate problem instance p_i with pre-defined parameters to satisfy the block-time T ;
2 Execute Algorithm (3) ;
3 Attach the problem to the requested block;
4 **return** *Block & Q*

The block-time T is the time required to add a block to the blockchain. For example, the average block-time for Bitcoin is 10 minutes. The time-block T depends on the problem difficulty.

Algorithm 2: Challenge Submitter

Input: problem instance p_x, block-time T

Output: Block data + QUBO Matrix Q

if *if instance* $T - \delta T \leq t_x \leq T + \delta T$ **then**
    go to step (7);
else
    call Algorithm (1) to generate a problem p_i such that
    $T - \delta T \leq t_x + t_i \leq T + \delta T$;
end
Execute Algorithm (3) ;
Attach the problem to the requested block;
return *Block & Q*

Algorithm 3: NP-to-QUBO

Input: NP problem family, Problem instance **P**

Output: QUBO Matrix **Q**

Select proper mapping algorithm based on NP problem family;
Map P to Q;
return Q;

In case that the family type of the submitted problem is QKP for Algorithm (3), as we discussed in chapter (2), to transform QKP to QUBO, the problem is first formulated as an Ising problem, which can be properly embedded on a Quantum Annealer. The Ising problem is solved as an energy function with goal of minimizing the energy cost, to briefly summarize the transformation in simple steps:

1. Introduce slack variables in the form of the binary expansion to get an equality constraint.

2. Change the maximization objective function to minimization function.

3. Add the penalty term with equality constraint to the minimized objective function.

Putting everything together, Fig. (3.3) shows the data structure scheme of a blockchain with a QUBO-based PoW. The procedure of the proposed scheme is presented in the next steps:

- The Genesis Block
- The blockchain of N blocks requests to add the next block $N+1$, which contains the the hash of former block.
- A task is submitted by a client or by the blockchain itself.
- In alg. (2), if the task p_x is submitted by a client, then task is checked if it satisfies the required block-time T. If it doesn't satisfy the block-time, then the task is submitted along with other task(s) p_i that are selected from the tasks database in alg. (1), such that, their total time should be within the block-time requirement $T - \delta T \leq t_x + t_i \leq T + \delta T$.
- Charges are deducted from the client's account.
- The submitted problem is mapped to QUBO matrix Q. Alg. (3) selects a proper mapping algorithm based on the submitted NP-hard problem family.
- Each node runs the solver algorithm, and broadcasts the found solution's hash value.
- Because the solution should be tightly tied to just one block, the solution hash is concatenated with the block's data, to serve as block N hash and carried on to the next block $N+1$.
- Block $N+1$ is added and the charge is paid to the client that found the best solution.

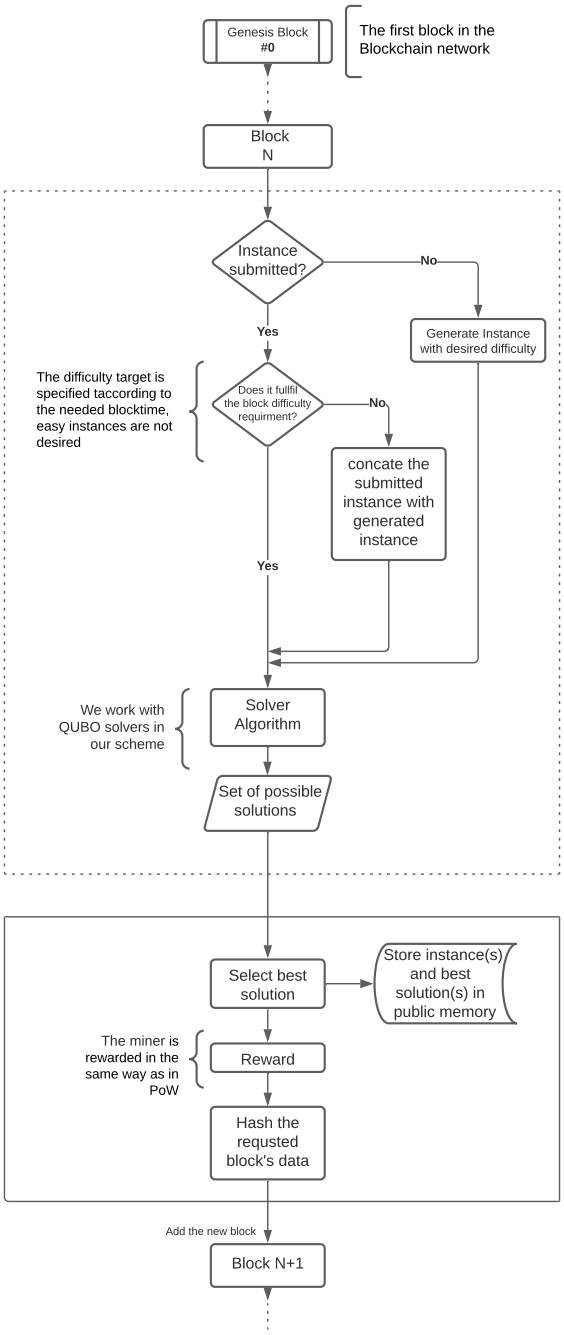

Figure 3.3: QUBO-based PoW scheme

Our QUBO-based PoW protocol borrows the same concepts from the PoS protocol in [Shi19] as explained in Sec. (3.4), like the concept of the client, searcher (we call it solver in our protocol), evaluator and miniblocks. However, because we introduced some new elements or alternatives to some existing elements in PoS, a comparison needs to be made to study how these changes might impact certain aspects of the protocol or how it might introduce new vulnerabilities to it. In Table 3.1, we compare the main properties of the PoS with our proposed protocol.

In Table 3.1, (✓) means that this property exists in our protocol. (✗) means that this property doesn't exists or has an alternative property in our protocol. (∼) means that it's not clear if this property exists in our protocol and it needs further discussion.

Table 3.1: Comparison between the properties of PoS and QUBO-based PoW

#	Proof-of-Search (PoS) properties in [Shi19]	QUBO-based PoW
1	All the properties of PoW are preserved	∼
2	Any node can submit a job and become a client.	✓
3	A client can specify any instance of an optimization problem in a job.	✓
4	A client can implement any search algorithm for any optimization problem for a job.	✗
5	A client pays a charge for their job.	✓
6	If clients submit no job, the protocol falls back to the basic PoW.	✗
7	The clients submit the task with an evaluator, which is used by the nodes to evaluate the solution candidates of the problem to be solved	✗
8	Miners have a financial incentive to find a good solution for the instance in a job.	✓
9	Miners have a financial incentive to provide the best-found solution to the client.	✓

10 The charge is automatically paid to the node that provides the best solution to the client.	✓
11 A probabilistic proof that the miners have evaluated a large number of solution candidates is provided.	∼
12 Multiple jobs can be executed at a time.	✓
13 The winning probability for each miner to find the best solution is proportional to the computational power spent by the miner for the job.	∼
14 The expected amount of computation for a job is proportional to the charge paid for the job.	✓
15 The variance in block time is lower than that with PoW.	∼
16 The probability of a fork is lower than that with PoW.	∼
17 The storage capacity required to manage a blockchain based on PoS is not too large.	∼
18 A blockchain based on PoS accepts jobs that run on computers with different architectures.	∼

In property #3 in Table 3.1, our protocol agrees with PoS, such that, it allows any participant in the blockchain to submit any optimization problem. However, in our protocol we require that the optimization problem can be converted into QUBO problem. There are few advantages to using the QUBO problems format. When we consider privacy concerns, many instances of useful NP-hard problems can arise in the framework of commercial developments. The problems themselves and their solutions may be trade secrets. For private (permissioned) blockchain networks this is not a critical problem, but the inability to use the wider possibilities of public blockchain can be restricting and limiting. One of our protocol advantages, is that it's possible to "mask" the original problem. Masking the original problem is possible due to the polynomial reduction of one problem to another. In our protocol, the client can submit a wide range of optimization problems, which are mapped to the QUBO formulation. For example, if a client submits a task to the blockchain network to find the best route for TSP problem, the converter algorithm in alg. (3) will

map this TSP instance to a QUBO problem and it will be solved accordingly, without publishing the original problem. Another property that our protocol doesn't agree with is the property #6, as we mentioned in the beginning of this section, our protocol doesn't have empty jobs, instead, in case of a shortage of guaranteed useful tasks, the network allows the generation of hard optimization problems that can model the structure of real-life problems.

One of the main concerns we find in the PoS protocol is the power given to the client by letting them implement the 'searcher' and the 'evaluator' themselves in property #4 and #7. Instead, our protocol only allows the client to submit an optimization problem, the solver and the 'evaluator' algorithms are implemented by the blockchain network. For example, in one of the potential attacks, [Shi19] discusses a cheat in which an 'evaluator' implemented by a malicious client returns an unfairly bad evaluation value when a special candidate solution is presented. In such case, the searcher set up by the malicious client will disregard this special solution candidate. In most cases, a client who submits a job with such an 'evaluator', can find the best solution to the problem and collect back the charge. Although these potential cheats in [Shi19] are argued to be resolved through the financial discouragement, as we explained in Sec. (3.5.1), we argue that malicious attacks are not necessarily performed to obtain financial advantage. In a situation where a client and a miner are colluding, doesn't necessarily mean that they are trying to obtain an unfair amount of coins where they already know a good solution, one of the possible attacks that can happen if they already know a good solution, is that they get to decide on what goes in the next block, and essentially can control certain transactions that occur within it. Our protocol resolves such cheats that are related to the client's power of implementing the 'evaluator' and the 'searcher' programs, as they are provided by the blockchain itself.

For the properties that are marked with $\sim$ in Table (3.1), we argue that our protocol agrees with them, as we kept the same components of the original PoS protocol with the same mechanism flow, instead, we limited the control of the client, which in return, enhanced parts of the protocol as discussed above.

Finally, as the proposed protocol needs a suitable optimization algorithm that can be an alternative to the 'searcher' algorithm in PoS, we proposed using the Quantum annealing Algorithm as the solver algorithm for our protocol. As we explained in Sec. (3.2), heuristic

algorithms are the most suitable algorithms to solve large optimization problems, as they do not spend an exponential amount of time searching for the optimal solution that can take up to hours or even days when large real-world problems are considered. Expanding on our study of the quantum annealing algorithm in the previous chapter, Quantum annealing is then considered as the solver algorithm for our protocol. However, there is a trade that needs to be made for using quantum computers that run such algorithm. In property #18, PoS accepts jobs that can run on computers with different architectures. While, our protocol could be used with any classical heuristic algorithm that can run on any computers with different architectures, in this project, however, we study the use of Quantum Annealers solvers, such as D-Wave.

Using quantum solvers, however, poses the question, if this would make the blockchain centralized with one central power. As in [KB18], the authors argue that quantum optimizers can be considered as computational blackboxes with approximately the same computational power, which is still higher than the power of any classical computer, and assuming that such Quantum Annealers will be available with more numbers and with more public access in the future, these solvers could be distributed between 30+ independent nodes and would still be relatively more decentralized compared with any of the current blockchain networks.

For example, when we consider Bitcoin that runs on PoW protocol, which is a decentralized protocol in principle, we find that in reality, Bitcoin's network is controlled by few pools as shown in Fig. (3.4), which are groups of miners that share computational power and split the reward if any of them successfully mine a block. This leaves any individual with an average computer with no possibility to ever successfully mine a block. However, with fair access to these Quantum Annealers, which are assumed to have approximately the same computational power, the average miner would be able to participate in the mining process.

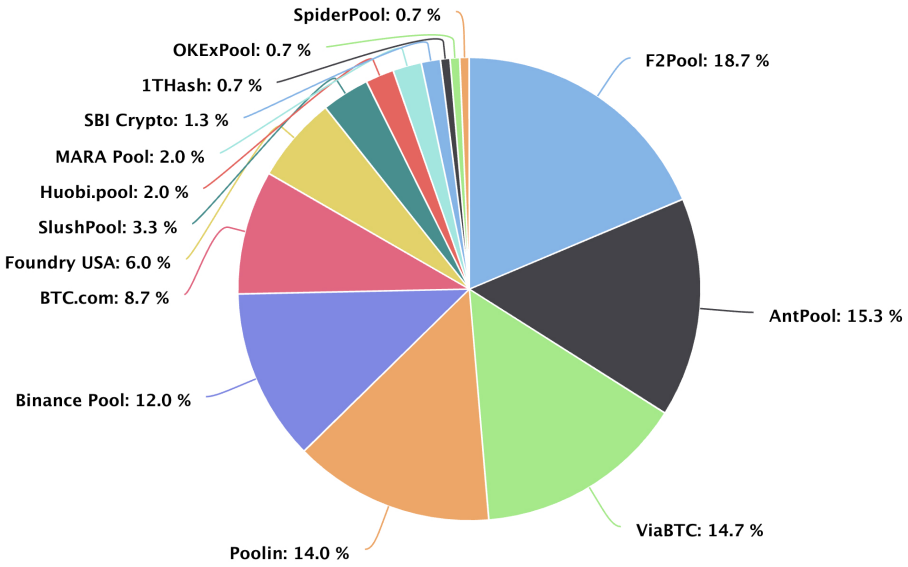

Figure 3.4: Hashrate Pools Distribution in Bitcoin as of September 12, 2021. Reproduced from [poo21]

Additionally, PoW-based Blockchain, such as Bitcoin, is prone to 51% attack, in which one or more hashing pools can obtain control of the the majority (51% or more) of the hashing power of the system to use it for malicious purposes. Hence, another advantage that can be obtained in exchange for using centralized Qunatun Annealers, is the possibility of using such solvers as a bottleneck, in which, they can be aware when nodes collude to obtain more than 51% of the solvers power.

3.6 QKP-QUBO as PoW challenge for Blockchain

In this section, we consider the task (or the challenge) component of our proposed protocol in the previous section, and study if it can be used as a practical alternative challenge to the hashing puzzle in the ordinary PoW.

Essentially, a good PoW would require satisfying the hardness property. According to the literature, research is still ongoing to develop an approach for generating 0-1 QKP

instances that are predictable and consistent in their difficulty. Studies that include a discussion of problem difficulty are Hansen and Meyer in [HM09], and Forrester in [For97]. It is concluded in both studies that the density of the objective function coefficients is positively correlated with CPU time, which is confirmed by our results in the chapter (2).

In this section, we do more analysis on the benchmarking instances we used in the chapter (2) to evaluate the suitability of using the QKP-QUBO problem as a PoW challenge. The baseline properties of the standard PoW in the Subsection (3.3.1) are used to compare them against our approach.

3.6.1 Intrinsic hardness

The intrinsic hardness is guaranteed by the NP-Hardness of the QUBO problem [KHG+14]. Because the general quadratic 0-1 contains problems like the minimum cut and maximum clique problems, which are NP-hard, hence, complexity-wise, the general quadratic 0-1 is NP-hard. Also, [PJ92] showed that the general quadratic 0-1 problem can be reduced to a minimization problem of a quadratic function with a unique solution in polynomial-time, and since the general quadratic 0-1 problem is NP-hard; hence, the minimization problem of a quadratic function with a unique solution is also considered NP-hard.

3.6.2 Adjustable Hardness property

For the QKP-QUBO problem to qualify as a good challenge for PoW, we want to adjust the hardness of the problem based on tuning some parameters of the challenge. In this section, we introduce a parameter called the diagonal dominance; this parameter has been used in [HBT08] to characterize the QUBO's difficulty in finding its optimal solution.

Defining diagonal dominance, a diagonally dominant matrix has the diagonal entry magnitude of a row to be larger than or equal to the sum of the magnitudes of all the other entries that are non-diagonal in that row. In a QUBO matrix, where

$$P = C^+ + Q^+ \text{ where } C^+ = \sum_{c_j>0} c_j \text{ and } Q^+ = \sum_{c_{ij}>0} c_{ij} \tag{3.2}$$

$$N = C^- + Q^- \text{ where } C^- = \sum_{c_j<0} c_j \text{ and } Q^- = \sum_{c_{ij}>0} c_{ij} \tag{3.3}$$

such that, P (and respectively, N) is the sum of the positive (and respectively, negative) coefficients of the QUBO. In the above equations, C represents the linear coefficients, and Q represents the quadratic coefficients.

The instance matrix is said to have p diagonal dominance when [HBT08],

$$p = \frac{C^+ - C^-}{Q^+ - Q^-} \ . \tag{3.4}$$

The second parameter we are defining is the WC-Ratio, which is the objects' weight to knapsack capacity ratio of the native QKP [WKG12],

$$\text{WC-ratio} = \frac{\sum w_i}{c} \ . \tag{3.5}$$

Most heuristics for QKP use this parameter to determine an initial solution by comparing the total object weights w_i with the knapsack capacity c and selecting those objects whose weight sum does not exceed c. Such that, fewer objects will be chosen in an initial solution if the ratio is large. However, more objects will be chosen in the initial solution if this ratio is small, implying that the problem may be harder to solve [WKG12].

In Fig. (3.5) and Fig. (3.6), we found a strong relationship between the diagonal dominance and WC-ratio. Although we did not see a clear effect of this relationship on the runtime scaling, it opens the door for further testing in determining the relationship between the QKP native problem and the transformed QUBO. However, we found a promising result in Fig. (3.6), there is a strong relationship between these parameters and the error gap produced when the transformed QUBO is solved using the simulated annealing algorithm. This could be potentially used to determine the number of runs required to reduce the error gap, hence, increasing the runtime of the problem.

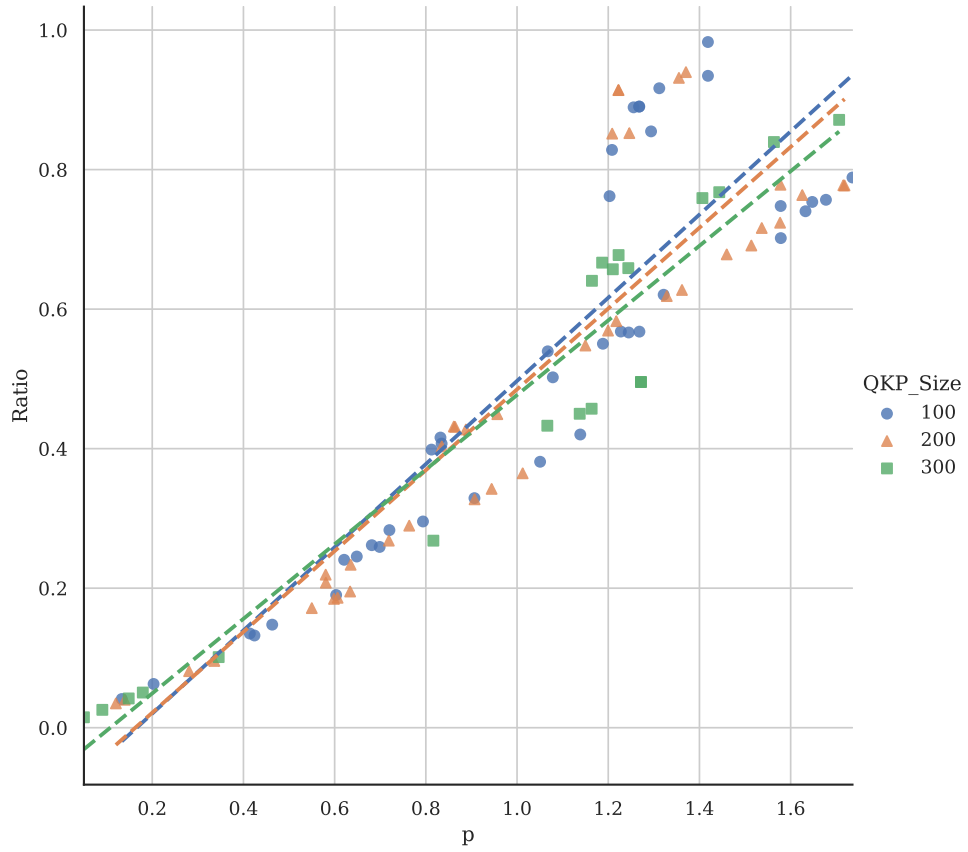

Figure 3.5: Regression plot for Diagonal dominance vs WC-Ratio plot, n =100, 200, 300

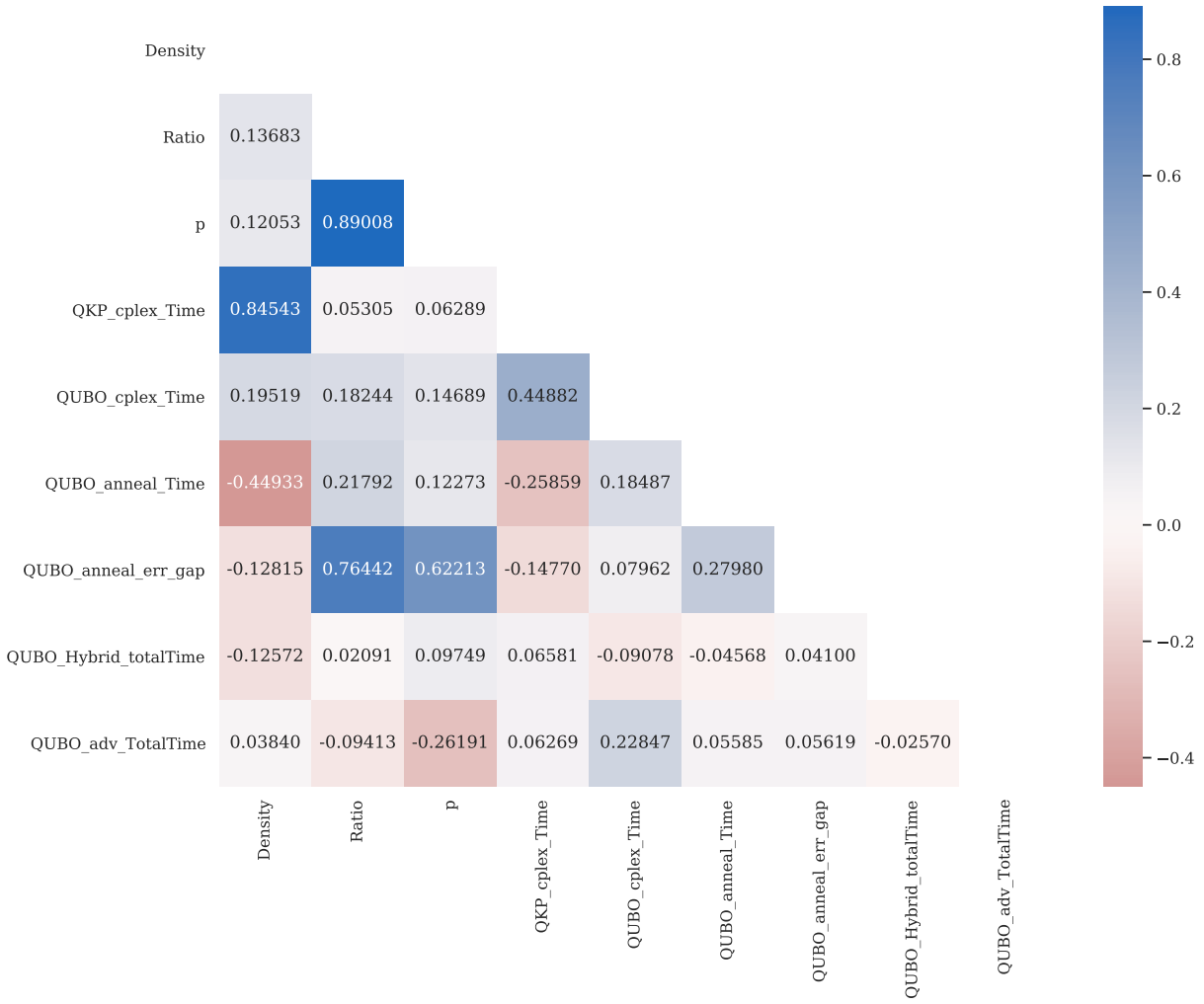

Figure 3.6: Correlation heatmap for n = 100

Another approach we could use to achieve the desired hardness is using the runtime scaling against the problem size. For example, if a QUBO problem has a small size, several challenges could be combined to be one challenge for just one block to achieve the desired complexity (block-time).

3.6.3 Easy verifiability property

The decision problem form of the Quadratic knapsack problem is an NP-complete problem, which asks, "Can a cost of at least C be achieved that does not exceed the weight W of the Knapsack?" [Pis07]. Even though the decision problem is NP-complete, the optimization problem form of the QKP is not, it's in the NP-Hard class, but it's still at least as difficult as the decision problem. Although verifying that the QKP obtained solution is the optimal answer is as difficult as solving the quadratic knapsack problem, given a set of solutions, we can verify the best of those solutions in polynomial time. Thus, even though one miner's solution is not guaranteed to be optimal, we can take advantage of this fact by requiring the network to be continuously updated by a better solution among the miner's different solutions, and the network rewards the best solution.

3.6.4 Block Sensitivity property

The Block Sensitivity property is fulfilled, because the solution is hashed together with the block's data hash, as discussed in the protocol, which is only tied to this block.

3.6.5 Non-reusability property

The QKP-QUBO problem associated with the given block is uniquely associated with the given block and is independent from other blocks' associated problems, thus, work done on one problem cannot be reused on another.

3.7 Conclusion

This chapter proposed solving hard combinatorial optimization problems as an alternative to the Hash puzzle used in the current blockchain's PoW protocol, improving the wasted energy problem in the usual PoW protocol. In particular, we studied the QKP family of problems in the NP-Hard complexity class. While the future of quantum computers imposes a considerable threat to the application of the blockchain, quantum technology can

be harnessed to improve parts of the blockchain better. We proposed Quantum Annealing as the proper metaheuristic algorithm to use as the solver for this family of problems.

The suitability of using the QKP-QUBO problem as PoW challenge is evaluated according to the baseline criteria of what constitutes a good PoW challenge. One of these properties is that a good PoW would require to satisfy the adjustable hardness property mainly. Our numerical experiment in the chapter (2) showed the problem complexity does indeed grow in a time when being solved by the metaheuristics: classical simulated annealing and Hybrid Quantum annealing; which is the basic requirement for the PoW concept to be practical. Also, from the difficulty analysis done in Sec. (3.6), we find a relationship between the diagonal dominance parameter of the QUBO problem and the WC-Ratio parameter of the native QKP. In the literature, both of these parameters are used to characterize the problem's difficulty; hence, finding this relationship between the QUBO and the QKP parameters could potentially help in tuning the desired difficulty for the PoW challenge.

Chapter 4

Simulating Chaos of The Quantum Kicked Ising Model on Quantum Annealer

4.1 Overview

In this chapter, we propose an approach for simulating the Quantum Kicked Ising Model (QKI) on the Quantum Annealer D-Wave. Simulating quantum chaotic systems on a real quantum machine could provide interesting insights into studying quantum chaos, and because the circuit model lacks simulating large qubit system, Quantum Annealing is more suited for this simulation, specifically, to study the chaotic transition between the deep quantum realm to the classical limit. To achieve this, we propose simulating the QKI model on D-Wave, which a well studied model that exhibits chaotic behavior, and it's naturally mapped onto the Quantum Annealing hardware. Also, this approach could provide insights into the connection between entanglement and quantum chaos, which could be utilized to study the "Quantumness" of D-Wave. The implementation of this approach is outside of the thesis's scope, and is left for future work.

4.2 Introduction

The classical chaos theory describes the unpredictability in the system's evolution dynamics due to its sensitivity to the system's initial conditions. The concept of chaos was first developed by Poincaré in [Poi05] in the 19th century. In classical dynamic systems, particles move along trajectories in the phase-space, where chaos is typically characterized by positive Lyapunov exponent [Gre10] in nonlinear systems, leading to trajectories with similar initial conditions to diverge exponentially fast from each other.

It follows, however, that since quantum physics is a limit of classical physics according to the correspondence principle, the search for quantum signatures of classical chaos is sought after. Hence, the subject of quantum chaos studies the relation between quantum mechanics and classical chaos. While classical chaos principles are well established in classical physics, it has not been rigorously established in the area of quantum physics. One of the reasons is the fundamental difference between both descriptions; according to quantum mechanics, conjugate observables like position and momentum cannot simultaneously take on determined values, unlike classical systems, where the states can be completely described using a set of dynamical variables [SM19]. The dynamics of the quantum systems are instead described by the linearity of the Schrödinger equation, which allows the superpositions of quantum states, and described by the uncertainty principle. As a result, the formulation of quantum mechanics in its current form is fundamentally incompatible with classical chaos, because the classical chaotic dynamics require nonlinearity and the exponential divergence of neighboring trajectories [GFS12].

One area of studying quantum chaos seeks to find signatures of classical chaos in quantum systems, to help to identify quantum realm's properties to distinguish the underlying regular dynamics of the system from the chaotic dynamics of classical dynamics. Even though many attempts in the literature have been made to characterize the signature of quantum chaos, this problem is still an open one [Kum19]. In this chapter, we focus on a particular area of research, which tries to identify quantum chaos signatures in the quantum entanglement's dynamics. In particular, we consider the Kicked Ising model, which has been shown in [Pro02], [LS05], [VKRPS18], to display both regular (integrable) and chaotic (non-integrable) behavior according to a certain choice of parameters in the model

simulation. We hope this proposed work can provide insights into studying entanglement and its relationship with classical dynamics and nonlocality in the Quantum Kicked Ising (QKI) model.

In the following section (4.3), we describe the QKI model and give an overview of the possible quantum chaos measures and entanglement measures. In Sec. (4.4), we outline the basic approach for mapping the QKI on D-Wave's Quantum Annealer. Finally in Sec. (4.5), we propose a possible future study under the general theme of our thesis, which is Quantum Annealing. We discuss the potential of using the connection between quantum chaos and entanglement to further investigate the problem of the "Quantumness" of D-Wave.

4.3 Background

4.3.1 Model Description

The model presented in this study is the quantum Kicked Ising (QKI) model, which is a 1D lattice chain of N qubits, coupled and locally interact with each other via the nearest neighbour Ising interaction. The system is periodically kicked with spatially homogeneous transverse magnetic field [Pro07].

The time dependent Hamiltonian can be written as:

$$\mathcal{H}(t)_{KI} = \mathcal{H}_I + \mathcal{H}_0 \sum_n \delta(t - n\tau) \tag{4.1}$$

where the kicks are provided by the $\delta(t - n\tau)$ which is a unit-periodic Dirac delta.

The 1-dim Ising Hamiltonian $\mathcal{H}_I$:

$$\mathcal{H}_I = \sum_i h_i \hat{\sigma}_z^{(i)} + \sum_{i>j} J_{i,j} \hat{\sigma}_z^{(i)} \hat{\sigma}_z^{(j)} \tag{4.2}$$

The Hamiltonian $\mathcal{H}_0$ for a spatially homogeneous magnetic field b applied along the x axis :

$$\mathcal{H}_0 = b \sum_i \hat{\sigma}_x^{(i)} \tag{4.3}$$

such that:

The indices i and j represent two neighboring qubits, and interactions between them are quantified by coupling strengths J_{ij}.

J_{ij}: homogeneous Ising spin-coupling interaction between spins i and j along the z-axis.

b: transverse magnetic field applied along the x-axis to the whole chain

h_i: longitudinal magnetic field applied along the z-axis on spin i, represents the linear coefficients corresponding to qubit biases

and $\sigma^i_{x,y,z}$ are the usual Pauli spin variables on a finite lattice, satisfying the commutation algebra $[\sigma^i_\alpha, \sigma^j_\beta] = \sum_\gamma 2\mathbf{i}\epsilon_{\alpha\beta\gamma}\delta^{ij}\sigma^i_\gamma$

Note that J_{ij}, b and h_i, are independent dimensionless parameters that specify the model [PPV14], and their sign does not impact the dynamics, as the system with positive parameters can be mapped into the system with negative parameters by performing some rotations.

4.3.2 Quantum Chaos Measures

Understanding quantum chaos requires a close look at classic chaos in the deep quantum regime. Several measures of quantum chaos have been developed in the literature, however, the two measures of chaos that are considered in this proposal are the Out-of-time-order correlators(OTOCs) and Level Spacing Statistics, as both of these measures and their connection to entanglement are explored in the literature [GHR18], [KSL07], [VLRK03].

Out-of-time-order correlators (OTOCs)

Chaos in non-integrable many-body systems is predicted to cause both thermalization and scrambling of information, leading to nonlocal information. Under the chaotic dynamics of the Heisenberg picture, when the local operator A grows with time, this growth is then

reflected in the noncommutativity of $A(t)$ and B, which is another local operator that exists at a different site, hence, resulting in the OTOC decaying with time [NZZ+19].

Assuming that A and B are Hermitian and unitary local operators, then [NZZ+19],

$$F_{AB}(t) =< A(t)B(0)A(t)B(0) > \tag{4.4}$$

In chaotic many-body systems, the OTOC F(t) is expected to drift away from 1 and approaches zero at a relatively late time, reflecting the chaotic nature of the dynamics, and this applied for almost any choice of operators A and B.

Level spacing distribution

For the QKI model, the system can be driven from the integrable regimes to the chaotic ones. This transition can be observed by studying the spectrum of eigenvalues and the distribution of eigenstates. One of the chaos measures we are considering is the level spacing statistics.

Level spacing distribution is used in literature as an indicator of chaos. To specify when the system becomes chaotic, the level spacing distribution is studied, such that, the distribution $P(s)$ of neighboring energy levels spacings, s, differs depending on whether the system is integrable or non-integrable system [Pra15]. In integrable systems, the energy levels are not correlated, and are not denied from crossing, so the distribution is Poissonian (P) [GFS12].

$$P(s)_p = e^{-s} \tag{4.5}$$

However, the eigenvalues in chaotic system become correlated resulting in "avoided crossings", meaning there is level repulsion [GFS12], the level statistics in this case P(s) is given by the Wigner-Dyson (WD) distribution given by:

$$P_{WD}(s) = \frac{\pi s}{2} e^{\frac{\pi s^2}{4}} \tag{4.6}$$

To demonstrate this, a study done in [KSL07] shows the connection between nonintegrability and avoided crossings, As shown in Fig. (4.1), the presence of the Wigner-Dyson distribution is an indicator of a certain measure of quantum chaos.

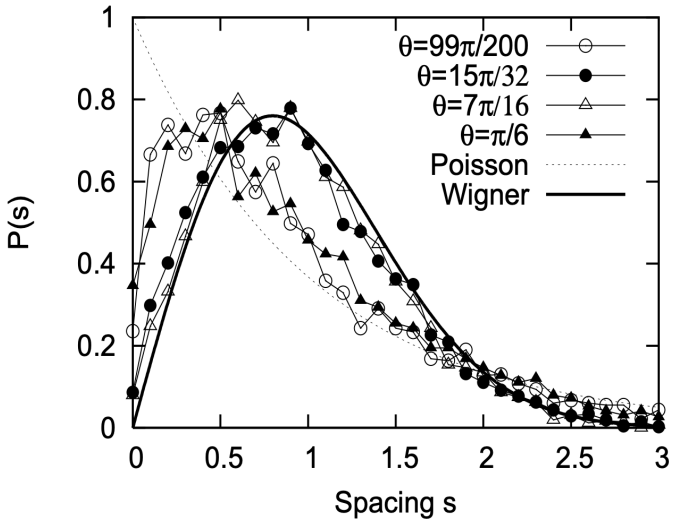

Figure 4.1: The nearest neighbor spacing distribution of the even states of a chain with N = 13 spins at various angles θ of tilt of the transverse magnetic field in the non-integrable region. The parameters are $J_{ij} = h_i = 1$ in QKI model from Eq. (4.2). Reproduced from [KSL07]

4.3.3 Entanglement Measures and Chaos

Entanglement is a quantum phenomenon that is used often as a powerful tool in quantum information theory. In principle, entanglement of two distant systems can produce instantaneous correlations between them that local hidden variable theories cannot explain [Wit14]; this phenomenon is called nonlocality, which cannot be realized by classical systems.

To quantify entanglement in any given quantum state, the use of entanglement measures is needed. Entanglement can exist in both two-body (or bipartite) and many-body (or

multipartite) systems. However, there has been more focus on bipartite entanglement measures in pure states and their relationship to chaos, mostly due to the fact that the von Neumann entropy of the reduced density matrices is a clear measure of entanglement.

A bipartite system consists of two different subsystems A and B, represented by the density matrix [Kum19];

$$\rho_{AB} = \sum_{i=0}^{N} p_i \rho_A \otimes \rho_B \tag{4.7}$$

However, if a bipartite system can be written in this form, it means that the bipartite is separable and the subsystems are not entanglement. For an entangled bipartite state, the von Neumann Entropy is used to measure the entanglement entropy E, and it's defined as [Kum19];

$$E(\rho_{AB}) = S(\rho_A) = S(\rho_B) \tag{4.8}$$

where $S(\rho)$ is the von Neumann Entropy and is defined as

$$S(\rho) = -tr\rho \log \rho \tag{4.9}$$

Another measure of entanglement content in a bipartite system is called Concurrence [HW97]. We explain it later in the section as it's a commonly used measure in studying entanglement and chaos in QKI model.

Despite the fact that the entanglement content of an Ising spin-chain could be greater than the amount of entanglement present in two-body correlations, the lack of studying the multipartite entanglement compared to the bipartite entanglement could be due to the difficulty of defining proper measures of global entanglement in these spin chains [LS05]. However, one of the few promising multipartite entanglement measures in the literature is the Meyer and Wallach Q measure, proposed in [MW02].

Concurrence

The concurrence measure associated with two qubits i and j, can be obtained from [AMMV20]

$$C(\rho_{ij}) = max\{0, \sqrt{\lambda_1} - \sqrt{\lambda_2} - \sqrt{\lambda_3} - \sqrt{\lambda_4}\} \tag{4.10}$$

where ρ_{ij} is the reduced density matrix.

λ_n is the eigenvalues of the matrix $R = \rho\widetilde{\rho}$ in decreasing order,

and $\widetilde{\rho_{ij}} = (\sigma_i^y \otimes \sigma_j^y)\rho_{ij}^*(\sigma_i^y \otimes \sigma_j^y)$.

The concurrence measure is always between 0 and 1, with the concurrence vanishing for unentangled states and reaching 1 for maximally entangled bipartite system.

The Meyer and Wallach Q measure

The Meyer and Wallach Q measure is based on the fact that the purity or the presence of mixed states of the reduced density matrices ρ_i can be viewed as an indicator of entanglement of the i^{th} qubit, which calculates the entanglement content between the given ith qubit and the rest spins in the system of N spins. The measure is defined as [MW02];

$$Q(\psi) = 2(1 - \frac{1}{N}\sum_{k=1}^{N} Tr(\rho_k^2)) \tag{4.11}$$

A study done in [LS05] on QKI model, showed that for spin chain system with spins N = 2 in the integrable regime (with zero-tilt of the transverse magnetic field), the concurrence coincides with Q, but for a higher number of spins, as shown in Fig. (4.2), they find that at times which the nearest neighbor concurrence vanish, the multipartite entanglement content as measure by Q is at its maximum during these periods, indicating that two-body correlations are being globally distributed throughout the spin-chain system.

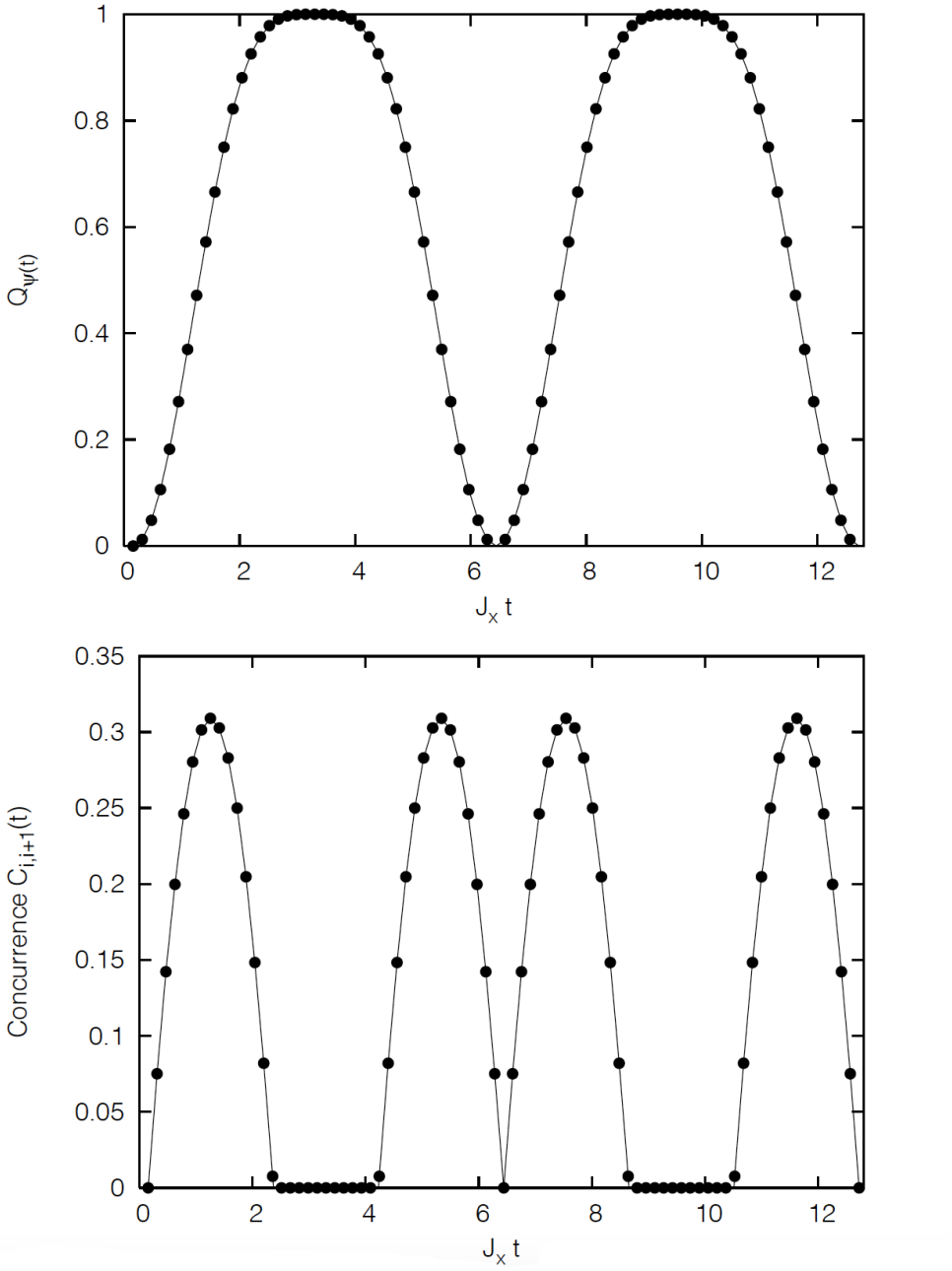

Figure 4.2: The Meyer and Wallach measure of entanglement Q and the nearest neighbour concurrence for spin chain N=4, as functions of (scaled) time. Reproduced from [LS05]

However, in the non-integrable regime, With a non-zero tilt angle of the transverse field applied on the QKI model, they find that while the maximum of the Q measure drops from 1, as in Fig. (4.3), the average of the measure is still larger then the entanglement content for the integrable case.

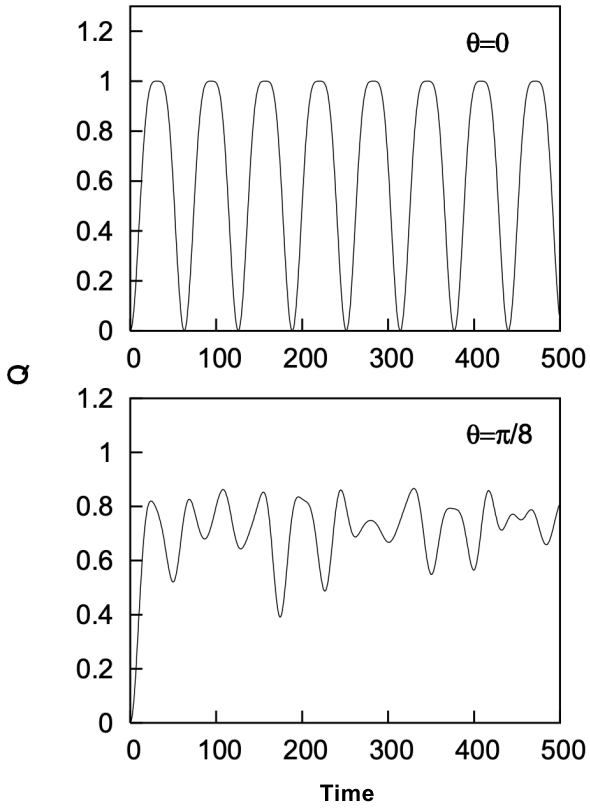

Figure 4.3: The entanglement measure Q as a function of time, for different tilt angles of the transverse field. The parameters are $J_{ij} = 0.1$, $h_i = 0.1$, $N = 10$ in QKI model from Eq. (4.2). Reproduced from [LS05]

4.4 Approach

4.4.1 Mapping QKI model on D-Wave Quantum Annealer

The D-Wave 2000Q implements the following schedule of quantum annealing:

$$\mathcal{H}_{ising} = A(s)\mathcal{H}_0 + B(s)\mathcal{H}_1 \tag{4.12}$$

where the driving functions $A(s)$ and $B(s)$ are changed over time as shown in fig (4.4). $A(s)$ represents the transverse field strength, and $B(s)$ is the energy applied to the problem Hamiltonian. The $\mathcal{H}_1$ is the target Hamiltonian, and $\mathcal{H}_0$ is the starting Hamiltonian, this is explained in more details in Chapter (1).

In a simplistic approach, we can think of the annealing functions as $A(s) = 1 - s(t)$, and $B(s) = s(t)$, such that $s(t)$ is a normalized anneal fraction, parameter ranging from 0 to 1. The typical choice is $s = \frac{t}{\tau}$, where τ is the annealing time.

For the D-Wave system, the Hamiltonian is represented as [McG14]:

$$\mathcal{H}_{ising} = -A(s)\left(\sum_i \hat{\sigma}_x^{(i)}\right) + B(s)\left(\sum_i h_i\hat{\sigma}_z^{(i)} + \sum_{i>j} J_{i,j}\hat{\sigma}_z^{(i)}\hat{\sigma}_z^{(j)}\right) \tag{4.13}$$

where $\hat{\sigma}_{x,z}^{(i)}$ are Pauli matrices operating on a qubit i. the coupling strength J_{ij} and longitudinal field h_i can be set to an arbitrary value chosen within the limits of the hardware connectivity respectively [CV+14].

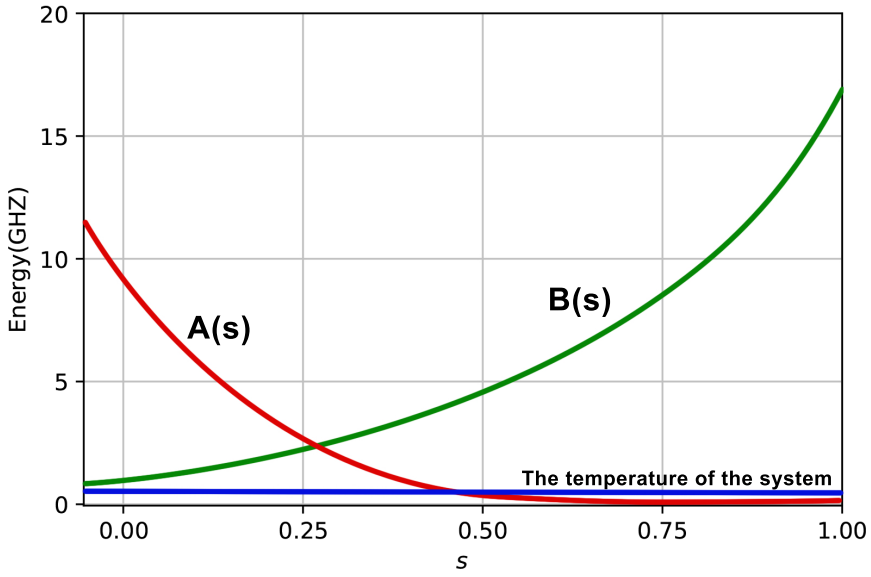

Figure 4.4: Functions $A(s)$ and $B(s)$ for the Annealing schedule

To map QKI model to the D-Wave's Ising model, simulating the periodic "kicks" would be the main challenge. One way to simulate these period kicks, is by manipulating the D-Wave "annealing Schedule". The time-dependent functions A(s) and B(s) give the annealing schedule. On D-Wave, the annealing schedule can be traversed at any rate the user chooses. Rather than having $s(t) = \frac{t}{\tau}$, which has the dimensionless time parameter s, be related to the time τ in a linear relationship, where τ is the total annealing time, D-Wave allows us to use more flexible relationships such as ds/dt for different annealing rates in different portions of the anneal. D-Wave implements two algorithms for the annealing schedule, the 'forward' annealing and the 'reverse' annealing [MVHR19].

Working with the forward annealing schedule, we can use the annealing protocol to simulate one kick, by annealing the system from $s = 0$ to some other point $s = s_p$, We then pause at $s = s_p$ for some time tp and again, we can continue the anneal to another $s = s_{p2}$ and then we can pause the anneal for some time to simulate the second kick and so on. Finally, we finish by annealing to s = 1. Hence, by having s, and consequently the Hamiltonian, to stay fixed for some period of time [MVHR19], we can achieve the kicks we aim for to implement the QKI model.

4.5 Proposed/Future Work and Remarks

In this section we discuss a possible advantage of this study under the general theme of our thesis, which is Quantum Annealing. We propose simulating the QKI model on the Quantum Annealer D-Wave for future work, and use the connection between entanglement and chaos in spin systems [Kum19] to study the "Quantumness" claim of D-Wave. In the following paragraphs, we discuss the problem of the "Quantumness" of D-Wave and the potential of using the connection between chaos and entanglement to further investigate this problem.

The "Quantumness" of D-Wave claims have provoked much controversy, as to whether or not their machine truly performs quantum annealing. It is very sensible to be skeptical and question whether this macroscopic artificial spin system truly behaves quantum mechanically. There are two fundamental quantum phenomena claimed to be present during the Quantum Annealing process on D-Wave. The first is quantum tunneling, and the second is entanglement. Because both phenomena are a critical indication of the quantumness of a system, several attempts have been considered in the literature to prove the "Quantumness" of D-Wave. For the first phenomenon, an experimental study [JAG+11] was carried out to determine whether the qubit dynamics during the annealing process is dominated by thermal activation or quantum tunneling. Their study used a superconducting flux quantum qubit array with programmable spin-spin couplings to implement an artificial Ising spin system that underwent quantum annealing to find the ground state of this system. To demonstrate a signature of quantum annealing in this artificial Ising spin system in their experiments, they measured the temperature dependence of the time when the dynamics of the system freezes, which is the point at which the system stops responding to changes in the landscape of its energy. The authors were able to show a clear indication of quantum annealing that is different from the classical thermal annealing, supporting the "Quantumness" argument of D-Wave.

For the second claim that entanglement is present during the Quantum Annealing process, another study in support of "Quantumness" of D-Wave is presented in [LPS+14]. Their study showed that during a critical part of the Quantum Annealing process, experimental evidence showed that the qubits become entangled, and this entanglement remains

even after reaching equilibrium with the surrounding thermal environment, proving the "Quantumness" of QA. Moreover, the presence of entanglement during Quantum Annealing has also been observed in [LKEH17] and [ARTL15]. In this proposal, we focus on investigating the claim of entanglement presence in D-Wave. The presented idea is to use the connection between entanglement and quantum chaos by simulating the chaotic model QKI on D-Wave, and use the available measures of quantum chaos as a possible witness of the presence of entanglement if it exists at all.

Two measures of chaos were considered in this proposal: Out-of-time-order correlators (OTOCs) and Level Spacing Statistics. Both of these measures and their connection to entanglement are explored in the literature. First, for the OTOCs measure, the late-time vanishing of OTOCs is related to the scrambling of information in quantum systems that takes place in chaotic systems. Also, witnessing entanglement in quantum systems is used as a probe of the scrambling of quantum information. In a study done in [GHR18], the authors showed that a specific family of OTOCs could serve as a witness to the entanglement of multiparticle systems. Second, for the level spacing measure [SLB20], when the spacing between energy levels move close to each other, sometimes reaching 0, resulting in an energy-level crossing. Studies in [VLRK03], [KSL07] and [OHPK08] showed that entanglement is generically witnessed at its maximum at this avoided crossing region.

For our final remarks, we discuss the advantage of simulating QKI on a quantum annealer like D-Wave. The current available qubit size using the circuit model for quantum chaos studies is limited. For example, IBM's current largest quantum computer contains 65 qubits [Cho20]. However, most quantum chaos experiments done on the available quantum computers or in the labs, are done with smaller qubit systems, that range from 2 qubits system to less than 10 qubits [LTF21]. With the launch of D-Wave's Advantage Quantum processor, which has over $5,000$ qubits, and with the promising scaling of the quantum annealing technology, simulating quantum chaotic models with such large qubit systems can provide an advantage to study the chaotic transition between the deep quantum realm and the classical limit. Also, it could provide insights into entanglement and its relationship with classical dynamics and nonlocality in the Quantum Kicked Ising model, which in return, could be useful in studying the "Quantumness" of quantum annealers, such as D-Wave.

Chapter 5

Conclusion

There are two main goals that are addressed in this thesis to assess the potential of Quantum Annealers. The first goal of this research was to study if quantum advantage is realized for solving NP-Hard problems with Quantum Annealing. The second goal was to study Quantum Annealers for potential use cases; specifically, we proposed a use case for Blockchain technology and a use-case in the field of Quantum Chaos.

For the first goal, we asked the specific question of whether Quantum Annealing can provide better performance for solving NP-hard Quadratic Knapsack problem (QKP) in terms of the solution quality and the total runtime. In addition to whether the transformation from QKP to the D-Wave's QUBO would impact the problem's structure and complexity. To answer this question for a metaheuristic solver like D-Wave's Quantum Annealer, benchmarking and computational analysis were necessary as such algorithms do not have a theoretical guarantee like exact algorithms.

In Chapter (2), we presented our approach to empirically test the QUBO reformulation of the QKP native problem, to compare the pure and the hybrid QA performance on D-Wave, against the CPLEX exact algorithm and the classical metaheuristic algorithm Simulated Annealing. To achieve this, we selected the appropriate benchmarking instances for the computational experiments as randomly generated QUBO problems are unlikely to provide useful benchmarking results for a metaheuristic solver. The results from the computational analyses suggest that the Simulated Annealing and D-Wave's Hybrid solver

outperform both the pure quantum solver and the classical implementation on CPLEX. However, as the problem grows in size, we found that that D-Wave's Hybrid solver does slightly better, in terms of the solution quality, by 2× times compared to simulated annealing. Hence, we concluded that Quantum Annealing can display quantum speed-up in the Hybrid solver case. Another result found a negative relationship between the density parameter of the native QKP problem and the complexity of the reformulated QUBO problem when solved using the Hybrid solver and Simulated Annealing, which is a non-intuitive result. From this result, we concluded that QUBO reformulation might influence the structure and the landscape of the original QKP problem when transformed and solved using the Simulated Annealing and the Hybrid solver approaches. Another interesting result found that solving the reformulated problem using the CPLEX exact algorithm sometimes fails for instances that have knapsack capacity very close to the weight of selected objects suggesting that the solution is sensitive to the boundary condition of the problem. However, the instances sometimes have some success probability when solved using metaheuristic algorithms. For future work, we seek to understand some of these non-intuitive results further and to possibly improve the solution quality and runtime on the Hybrid solver by developing a customized algorithm inside one of the Hybrid solver branches.

For the second goal of this thesis, in Chapter (3) we proposed a protocol for solving QKP-QUBO as an alternative to the Hash puzzle used in the current Blockchain PoW protocol, improving the wasted energy problem in the usual PoW protocol, and use Quantum Annealer as their solvers in the protocol. While we were not the first to propose using QUBO and Quantum Annealing for PoW in Blockchain, as it was first proposed in [KB18], our contribution was generalizing the protocol to work with various types of optimization problems mapped to the general QUBO problem. In particular, we studied QKP as one of the other possible potential candidates for a practical PoW problem. Also, we evaluated the QKP-QUBO problem with consideration of how it would satisfy the Blockchain consensus protocol requirements for a practical PoW problem. We showed that the suggested problem does satisfy the baseline criteria to work in a practical PoW protocol.

Another use-case for Quantum Annealers was proposed in the Chapter (4). We proposed an approach to simulate the Quantum Kicked Ising Model on the Quantum Annealer D-wave, motivated by two aims. First, gaining more interesting insights into studying quan-

tum chaos by simulating a chaotic quantum system on a real quantum machine. Second, to study the "Quantumness" of D-Wave via the connection between entanglement and quantum chaos. The testing and development of this study in the area of quantum chaos are left for future work.

References

[AC09] Mohammad HS Amin and Vicki Choi. First-order quantum phase transition in adiabatic quantum computation. *Physical Review A*, 80(6):062326, 2009.

[ACd89] B. Apolloni, C. Carvalho, and D. de Falco. Quantum stochastic optimization. *Stochastic Processes and their Applications*, 33(2):233–244, 1989.

[AKR10] Boris Altshuler, Hari Krovi, and Jérémie Roland. Anderson localization makes adiabatic quantum optimization fail. *Proceedings of the National Academy of Sciences*, 107(28):12446–12450, 2010.

[AL18] Tameem Albash and Daniel A Lidar. Demonstration of a scaling advantage for a quantum annealer over simulated annealing. *Physical Review X*, 8(3):031016, 2018.

[AMMV20] Atefeh Ashouri, Saeed Mahdavifar, Grégoire Misguich, and Javad Vahedi. Concurrence and quantum discord in the eigenstates of chaotic and integrable spin chains. *Annalen der Physik*, 532(8):1900515, 2020.

[Aro04] Jasbir S. Arora. 15 - discrete variable optimum design concepts and methods. In Jasbir S. Arora, editor, *Introduction to Optimum Design (Second Edition)*, pages 513–530. Academic Press, San Diego, second edition edition, 2004.

[ARTL15] Tameem Albash, Troels F Rønnow, Matthias Troyer, and Daniel A Lidar. Reexamining classical and quantum models for the d-wave one processor. *The European Physical Journal Special Topics*, 224(1):111–129, 2015.

[AvDK+08] Dorit Aharonov, Wim van Dam, Julia Kempe, Zeph Landau, Seth Lloyd, and Oded Regev. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM Review*, 50(4):755–787, 2008.

[BBPP99] Immanuel M Bomze, Marco Budinich, Panos M Pardalos, and Marcello Pelillo. The maximum clique problem. In *Handbook of combinatorial optimization*, pages 1–74. Springer, 1999.

[BF28] M. Born and V. A. Fock. "beweis des adiabatensatzes.". *Zeitschrift f¨ur Physik A, Hadrons and Nuclei, 51(3-4): 165 - 180*, 1928.

[bit21] Bitcoin energy consumption index, June 2021. https://digiconomist.net/bitcoin-energy-consumption/.

[BRSV18] Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. Proofs of work from worst-case assumptions. Cryptology ePrint Archive, Report 2018/559, 2018. https://eprint.iacr.org/2018/559.

[BS04] Alain Billionnet and Eric Soutif. An exact method based on lagrangian decomposition for the 0-1 quadratic knapsack problem. *European Journal of Operational Research*, 157(3):565–575, 2004.

[CH17] Yuning Chen and Jin-Kao Hao. An iterated "hyperplane exploration" approach for the quadratic knapsack problem. *Computers and Operations Research*, 77:226–239, 2017.

[Cho20] Adrian Cho. Ibm promises 1000-qubit quantum computer—a milestone—by 2023. *Science, September*, 10, 2020.

[CV+14] PJD Crowley, T urić, W Vinci, PA Warburton, and AG Green. Quantum and classical dynamics in adiabatic computation. *Physical Review A*, 90(4):042317, 2014.

[Das13] Sanjeeb Dash. A note on QUBO instances defined on Chimera graphs. *arXiv e-prints*, page arXiv:1306.1202, June 2013.

[DC08] Arnab Das and Bikas K Chakrabarti. Colloquium: Quantum annealing and analog quantum computation. *Reviews of Modern Physics*, 80(3):1061, 2008.

[Din19] Jintai Ding. A new proof of work for blockchain based on random multivariate quadratic equations. In Jianying Zhou, Robert Deng, Zhou Li, Suryadipta Majumdar, Weizhi Meng, Lingyu Wang, and Kehuan Zhang, editors, *Applied Cryptography and Network Security Workshops*, pages 97–107, Cham, 2019. Springer International Publishing.

[Doc] D-Wave System Documentation. D-wave qpu architecture: Topologies. Technical report. https://docs.dwavesys.com/docs/latest/c_gs_4.html.

[DW19] Ronald De Wolf. Quantum computing: Lecture notes. *arXiv preprint arXiv:1907.09415*, 2019.

[EMG20] Marcus Edwards, Atefeh Mashatan, and Shohini Ghose. A review of quantum and hybrid quantum/classical blockchain protocols. *Quantum Information Processing*, 19:1–22, 2020.

[Fey82] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6):467–488, 1982. ID: Feynman1982.

[FGG+01] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, Joshua Lapan, Andrew Lundgren, and Daniel Preda. A quantum adiabatic evolution algorithm applied to random instances of an np-complete problem. *Science*, 292(5516):472–475, 2001.

[FGG+09] Edward Farhi, Jeffrey Goldstone, David Gosset, Sam Gutmann, Harvey B Meyer, and Peter Shor. Quantum adiabatic algorithms, small gaps, and different paths. *arXiv preprint arXiv:0909.4766*, 2009.

[FGGN08] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Daniel Nagaj. How to make the quantum adiabatic algorithm fail. *International Journal of Quantum Information*, 6(03):503–516, 2008.

[For60] Robert Fortet. Applications de l'algebre de boole en recherche opérationelle. *Revue Française de Recherche Opérationelle*, 4(14):17–26, 1960.

[For97] R Forrester. Effective, compact linear formulations of 0–1 quadratic programs. *Master's Project. Department of Mathematical Sciences, Clemson University, Clemson, SC*, 1997.

[GAH20] Zoe Gonzalez Izquierdo, Tameem Albash, and Itay Hen. Testing a quantum annealer as a quantum thermal sampler. *arXiv e-prints*, page arXiv:2003.00361, February 2020.

[GFS12] Aviva Gubin and Lea F. Santos. Quantum chaos: An introduction via chains of interacting spins 1/2. *American Journal of Physics*, 80(3):246–251, 2012.

[GG78] Bezalel Gavish and Stephen C Graves. The travelling salesman problem and related problems. 1978.

[GHR18] Martin Gärttner, Philipp Hauke, and Ana Maria Rey. Relating out-of-time-order correlations to entanglement via multiple-quantum coherences. *Physical review letters*, 120(4):040402, 2018.

[GKD19] Fred Glover, Gary Kochenberger, and Yu Du. Quantum bridge analytics i: a tutorial on formulating and using qubo models. *4OR*, 17(4):335–371, 2019.

[Gre10] Walter Greiner. Lyapunov exponents and chaos. In *Classical Mechanics*, pages 503–516. Springer, 2010.

[Gri60] David J Griffiths. *Introduction to quantum mechanics*. Pearson International Edition (Pearson Prentice Hall, Upper Saddle River, 2005), 1960.

[Häm06] W Hämäläinen. Class np, np-complete, and np-hard problems, 2006.

[HBCT17] Layla Hormozi, Ethan W Brown, Giuseppe Carleo, and Matthias Troyer. Non-stoquastic hamiltonians and quantum annealing of an ising spin glass. *Physical review B*, 95(18):184416, 2017.

[HBT08] P. Hammer, E. Boros, and Gabriel Tavares. New algorithms for quadratic unconstrained binary optimization (qubo) with applications in engineering and social sciences. 2008.

[Hen13] Itay Hen. Complexity of the quantum adiabatic algorithm. In *APS March Meeting Abstracts*, volume 2013, pages B27–004, 2013.

[HM09] Pierre Hansen and Christophe Meyer. Improved compact linearizations for the unconstrained quadratic 0–1 minimization problem. *Discrete Applied Mathematics*, 157(6):1267–1290, 2009. Reformulation Techniques and Mathematical Programming.

[HW97] Scott Hill and William K Wootters. Entanglement of a pair of quantum bits. *Physical review letters*, 78(26):5022, 1997.

[JAG+11] M. W. Johnson, M. H. S. Amin, S. Gildert, T. Lanting, F. Hamze, N. Dickson, R. Harris, A. J. Berkley, J. Johansson, P. Bunyk, E. M. Chapple, C. Enderud, J. P. Hilton, K. Karimi, E. Ladizinsky, N. Ladizinsky, T. Oh, I. Perminov, C. Rich, M. C. Thom, E. Tolkacheva, C. J. S. Truncik, S. Uchaikin, J. Wang, B. Wilson, and G. Rose. Quantum annealing with manufactured spins. *Nature*, 473(7346):194–198, May 12, 2011.

[JP94] Alain Joye and Charles-Edouard Pfister. Quantum adiabatic evolution. In *On Three Levels*, pages 139–148. Springer, 1994.

[JRS07] Sabine Jansen, Mary-Beth Ruskai, and Ruedi Seiler. Bounds for the adiabatic approximation with applications to quantum computation. *Journal of Mathematical Physics*, 48(10):102111, 2007.

[KB18] Kirill P. Kalinin and Natalia G. Berloff. Blockchain platform with proof-of-work based on analog hamiltonian optimisers, 2018.

[KHG+14] Gary Kochenberger, Jin-Kao Hao, Fred Glover, Mark Lewis, Zhipeng Lü, Haibo Wang, and Yang Wang. The unconstrained binary quadratic programming problem: a survey. *Journal of Combinatorial Optimization*, 28(1):58–81, 2014.

[KSL07] J Karthik, Auditya Sharma, and Arul Lakshminarayan. Entanglement, avoided crossings, and quantum chaos in an ising model with a tilted magnetic field. *Physical Review A*, 75(2):022304, 2007.

[Kum19] Meenu Kumari. Quantum-classical correspondence and entanglement in periodically driven spin systems. 2019.

[LG17] Mark Lewis and Fred Glover. Quadratic unconstrained binary optimization problem preprocessing: Theory and empirical analysis. *Netw.*, 70(2):79–97, September 2017.

[LKEH17] Trevor Lanting, Andrew D King, Bram Evert, and Emile Hoskinson. Experimental demonstration of perturbative anticrossing mitigation using nonuniform driver hamiltonians. *Physical Review A*, 96(4):042322, 2017.

[LPS+14] Trevor Lanting, Anthony J Przybysz, A Yu Smirnov, Federico M Spedalieri, Mohammad H Amin, Andrew J Berkley, Richard Harris, Fabio Altomare, Sergio Boixo, Paul Bunyk, et al. Entanglement in a quantum annealing processor. *Physical Review X*, 4(2):021041, 2014.

[LS05] Arul Lakshminarayan and V Subrahmanyam. Multipartite entanglement in a one-dimensional time-dependent ising model. *Physical Review A*, 71(6):062334, 2005.

[LTF21] S Leontica, F Tennie, and T Farrow. Simulating molecules on a cloud-based 5-qubit ibm-q universal quantum computer. *Communications Physics*, 4(1):1–7, 2021.

[Luc14] Andrew Lucas. Ising formulations of many np problems. *Frontiers in Physics*, 2:5, 2014.

[Man87] CPLEX User's Manual. Ibm ilog cplex optimization studio. *Version*, 12:1987–2018, 1987.

[McC76] G. P. McCormick. Computability of global solutions to factorable nonconvex programs: Part i — convex underestimating problems. *Mathematical Programming*, 10(1):147–175, 1976.

[McG14] Catherine C McGeoch. Adiabatic quantum computation and quantum annealing: Theory and practice. *Synthesis Lectures on Quantum Computing*, 5(2):1–93, 2014.

[Mer07] N David Mermin. *Quantum computer science: an introduction*. Cambridge University Press, 2007.

[Moa18] Mohsen Moarefdoost. Optimization modeling in python: Pulp, gurobi, and cplex, October 2018. [Online; posted 10-October-2018].

[MRKA18] Aarne Mämmelä, Jukka Riekki, Adrian Kotelba, and Antti Anttonen. Multidisciplinary and historical perspectives for developing intelligent and resource-efficient systems. *IEEE Access*, 6:17464–17499, 2018.

[MVHR19] Jeffrey Marshall, Davide Venturelli, Itay Hen, and Eleanor G Rieffel. Power of pausing: Advancing understanding of thermalization in experimental quantum annealers. *Physical Review Applied*, 11(4):044083, 2019.

[MW02] David A Meyer and Nolan R Wallach. Global entanglement in multiparticle systems. *Journal of Mathematical Physics*, 43(9):4273–4278, 2002.

[MW13] Catherine C. McGeoch and Cong Wang. Experimental evaluation of an adiabiatic quantum system for combinatorial optimization. In *Proceedings of the ACM International Conference on Computing Frontiers*, CF '13, New York, NY, USA, 2013. Association for Computing Machinery.

[NP09] Apurva Narayan and C. Patvardhan. A novel quantum evolutionary algorithm for quadratic knapsack problem. In *2009 IEEE International Conference on Systems, Man and Cybernetics*, pages 1388–1392, 2009.

[NZZ+19] Xinfang Nie, Ze Zhang, Xiuzhu Zhao, Tao Xin, Dawei Lu, and Jun Li. Detecting scrambling via statistical correlations between randomized measurements on an nmr quantum simulator. *arXiv preprint arXiv:1903.12237*, 2019.

[OHPK08] Sangchul Oh, Zhen Huang, Uri Peskin, and Sabre Kais. Entanglement, berry phases, and level crossings for the atomic breit-rabi hamiltonian. *Physical Review A*, 78(6):062106, 2008.

[PB19] Davide Pastorello and Enrico Blanzieri. Quantum annealing learning search for solving qubo problems. *Quantum Information Processing*, 18(10):1–17, 2019.

[PB21] Frank Phillipson and Harshil Singh Bhatia. Portfolio optimisation using the d-wave quantum annealer. In *International Conference on Computational Science*, pages 45–59. Springer, 2021.

[Pis07] David Pisinger. The quadratic knapsack problem—a survey. *Discrete Applied Mathematics*, 155(5):623–648, 2007.

[PJ92] Panos M Pardalos and Somesh Jha. Complexity of uniqueness and local search in quadratic 0–1 programming. *Operations research letters*, 11(2):119–123, 1992.

[Poi05] Henri Poincaré. Science and method, 2005.

[poo21] Bitcoin pool distribution, September 2021. https://btc.com/stats/pool.

[PPV14] Carlos Pineda, Tomaž Prosen, and Eduardo Villaseñor. Two dimensional kicked quantum ising model: dynamical phase transitions. *New Journal of Physics*, 16(12):123044, 2014.

[Pra15] Enrico Prati. Towards room temperature solid state quantum devices at the edge of quantum chaos for long-living quantum states. In *Journal of Physics: Conference Series*, volume 626, page 012011. IOP Publishing, 2015.

[Pro02] Tomaž Prosen. General relation between quantum ergodicity and fidelity of quantum dynamics. *Physical Review E*, 65(3):036208, 2002.

[Pro07] Tomaž Prosen. Chaos and complexity of quantum motion. *Journal of Physics A: Mathematical and Theoretical*, 40(28):7881–7918, jun 2007.

[PS98] Christos H Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1998.

[PWS+16] Ojas Parekh, Jeremy Wendt, Luke Shulenburger, Andrew Landahl, Jonathan Moussa, and John Aidun. Benchmarking adiabatic quantum optimization for complex network analysis. *arXiv preprint arXiv:1604.00319*, 2016.

[RCC89] P. Ray, B. K. Chakrabarti, and Arunava Chakrabarti. Sherrington-kirkpatrick model in a transverse field: Absence of replica symmetry breaking due to quantum fluctuations. *Phys. Rev. B*, 39:11828–11832, Jun 1989.

[RKY+19] Scott Ruoti, Ben Kaiser, Arkady Yerukhimovich, Jeremy Clark, and Robert Cunningham. Sok: Blockchain technology and its potential use cases. *arXiv preprint arXiv:1909.12454*, 2019.

[SC95] Jun John Sakurai and Eugene D Commins. Modern quantum mechanics, revised edition, 1995.

[SDK19] Willa Ariela Syafruddin, Sajjad Dadkhah, and Mario Köppen. Blockchain scheme based on evolutionary proof of work. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pages 771–776, 2019.

[Shi19] N. Shibata. Proof-of-search: Combining blockchain consensus formation with solving optimization problems. *IEEE Access*, 7:172994–173006, 2019.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, October 1997.

[Sho17] Ali Shoker. Sustainable blockchain through proof of exercise. In *2017 IEEE 16th International Symposium on Network Computing and Applications (NCA)*, pages 1–9, 2017.

[SLB20] B Sharmila, S Lakshmibala, and V Balakrishnan. Signatures of avoided energy-level crossings in entanglement indicators obtained from quantum tomograms. *Journal of Physics B: Atomic, Molecular and Optical Physics*, 53(24):245502, 2020.

[SM19] Kevin M Short and Matthew A Morena. Signatures of quantum mechanics in chaotic systems. *Entropy*, 21(6):618, 2019.

[SM20] Ramses Sala and Ralf Müller. Benchmarking for metaheuristic black-box optimization: perspectives and open challenges. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2020.

[Sys] D-Wave Systems. D-wave hybrid. Technical report. https://docs.ocean.dwavesys.com/_/downloads/hybrid/en/latest/pdf/.

[TC15] Boaz Tamir and Eliahu Cohen. Notes on adiabatic quantum computers. /12/23 2015.

[VD19] Tomas Vyskocil and Hristo Djidjev. Embedding equality constraints of optimization problems into a quantum annealer. *Algorithms*, 12(4), 2019.

[VDMV01] Wim Van Dam, Michele Mosca, and Umesh Vazirani. How powerful is adiabatic quantum computation? In *Proceedings 42nd IEEE symposium on foundations of computer science*, pages 279–287. IEEE, 2001.

[VKRPS18] CW Von Keyserlingk, Tibor Rakovszky, Frank Pollmann, and Shivaji Lal Sondhi. Operator hydrodynamics, otocs, and entanglement growth in systems without conservation laws. *Physical Review X*, 8(2):021013, 2018.

[VL17] Walter Vinci and Daniel A Lidar. Non-stoquastic hamiltonians in quantum annealing via geometric phases. *npj Quantum Information*, 3(1):1–6, 2017.

[VLRK03] Guifre Vidal, José Ignacio Latorre, Enrique Rico, and Alexei Kitaev. Entanglement in quantum critical phenomena. *Physical review letters*, 90(22):227902, 2003.

[WHH+19] Wenbo Wang, Dinh Thai Hoang, Peizhao Hu, Zehui Xiong, Dusit Niyato, Ping Wang, Yonggang Wen, and Dong In Kim. A survey on consensus mechanisms and mining strategy management in blockchain networks. *IEEE Access*, 7:22328–22370, 2019.

[Wit75] Christoph Witzgall. Mathematical methods of site selection for electronic message systems (ems). *NASA STI/Recon Technical Report N*, 76:18321, 1975.

[Wit14] Peter Wittek. 3 - quantum mechanics. In Peter Wittek, editor, *Quantum Machine Learning*, pages 25–39. Academic Press, Boston, 2014.

[WKG12] Haibo Wang, Gary Kochenberger, and Fred Glover. A computational study on the quadratic knapsack problem with multiple constraints. *Comput. Oper. Res.*, 39(1):3–11, January 2012.

[YMRS18] Dylan Yaga, Peter Mell, Nik Roby, and Karen Scarfone. Nistir 8202 blockchain technology overview. *National Institute of Standards and Technology, US Department of Commerce, Washington, USA*, 2018.

[YWC13] Zhen Yang, Guoqing Wang, and Feng Chu. An effective GRASP and tabu search for the 0-1 quadratic knapsack problem. *Computers and Operations Research*, 40(5):1176–1185, May 2013.

[ZE81] Stelios H. Zanakis and James R. Evans. Heuristic "optimization": Why, when, and how to use it. *Interfaces*, 11(5):84–91, 1981.

APPENDICES

Appendix A

Computational Experiment Results

- Instance name $\longrightarrow$ I
- QKP size $\longrightarrow$ n
- QKP density $\longrightarrow$ d
- QKP WC-Ratio $\longrightarrow$ r
- QUBO diagonal dominance $\longrightarrow$ p
- QKP optimal on CPLEX $\longrightarrow$ S_{qkp}
- QKP runtime on CPLEX in seconds $\longrightarrow$ t_{qkp}
- QUBO optimal on CPLEX $\longrightarrow$ S_{qubo}
- QUBO runtime on CPLEX in seconds $\longrightarrow$ t_{qubo}
- QUBO optimal on SA $\longrightarrow$ S_{anl}
- QUBO runtime on SA in seconds $\longrightarrow$ t_{anl}
- QUBO optimal on Hybrid Solver $\longrightarrow$ S_{hy}
- QUBO runtime on Hybrid Solver in seconds $\longrightarrow$ t_{hy}
- QUBO optimal on Advantage Solver $\longrightarrow$ S_{adv}
- QUBO runtime in seconds on Advantage Solver $\longrightarrow$ t_{adv}

I	n	d	r	S_{qkp}	t_{qkp}	p	S_{qubo}	t_{qubo}	S_{anl}	t_{anl}	S_{hy}	t_{hy}	S_{adv}	t_{adv}
r_100_25_1	100	0.25	0.259	18558	2.688	0.699	15438	33.245	17373	13.4839582	15298	18.405	8290	308.509
r_100_25_2	100	0.25	0.855	56525	1.132	1.294	49806	34.858	42669	24.7356815	49023	16.213	18018	492.285
r_100_25_3	100	0.25	0.063	3752	2.870	0.203	2775	34.950	3455	14.2860336	3417	15.805	100	493.237
r_100_25_4	100	0.25	0.762	50382	2.812	1.203	45716	34.956	36555	22.6666112	43760	15.254	24748	197.125
r_100_25_5	100	0.25	0.934	61494	2.323	1.419	58334	37.100	40986	29.3656941	50389	15.377	21593	349.684
r_100_25_6	100	0.25	0.540	36360	2.444	1.067	29796	34.104	22593	21.3408511	28922	17.353	20935	468.322
r_100_25_7	100	0.25	0.190	14657	2.852	0.603	12740	33.764	12730	11.0596263	11890	17.605	5087	372.677
r_100_25_8	100	0.25	0.262	20452	1.120	0.681	16872	33.983	18845	16.4170437	17015	15.535	10066	355.670
r_100_25_9	100	0.25	0.568	35438	1.818	1.268	29899	36.133	28726	9.05917239	27399	15.501	17153	248.868
r_100_25_10	100	0.25	0.399	24930	2.758	0.812	21305	33.516	22213	12.5192857	19298	15.325	11310	356.729
r_100_50_1	100	0.5	0.621	83742	4.175	1.321	78216	32.022	73174	22.3957322	74576	15.911	34991	207.690
r_100_50_2	100	0.5	0.789	104856	8.542	1.736	90512	33.407	67296	14.4557595	91872	14.344	43150	266.549
r_100_50_3	100	0.5	0.241	34006	2.980	0.621	28111	34.032	32950	8.34003878	29453	17.204	12126	512.010
r_100_50_4	100	0.5	0.828	105996	5.846	1.208	99023	33.383	63856	10.2393811	90247	15.988	42228	381.986
r_100_50_5	100	0.5	0.420	56464	10.757	1.139	51552	33.976	47048	10.8758123	47279	16.076	29469	126.800
r_100_50_6	100	0.5	0.135	16083	5.212	0.414	13396	33.045	15595	11.2068839	14698	15.547	5644	400.015
r_100_50_7	100	0.5	0.407	52819	3.708	0.835	37948	32.263	46161	12.999752	43308	14.606	23234	182.865
r_100_50_8	100	0.5	0.416	54246	2.912	0.832	46431	32.837	50681	8.80625868	45038	24.187	27410	279.023
r_100_50_9	100	0.5	0.550	68974	6.620	1.188	61569	32.231	60375	15.4210713	58374	15.663	38536	212.043
r_100_50_10	100	0.5	0.702	88634	24.278	1.578	84456	34.333	54450	14.3333266	71913	29.586	31234	279.599
r_100_75_1	100	0.75	0.983	189137	1.182	1.419	183462	32.003	117379	5.59568429	180435	15.436	57186	335.881
r_100_75_2	100	0.75	0.502	94957	30.816	1.078	88771	33.985	90831	6.19878244	86553	23.330	50368	305.562
r_100_75_3	100	0.75	0.329	62098	11.001	0.907	47021	33.091	58965	6.28894353	53653	15.258	31412	280.601
r_100_75_4	100	0.75	0.381	72167	30.618	1.051	58062	35.057	69140	19.882086	66992	17.645	38465	421.623
r_100_75_5	100	0.75	0.148	27616	9.849	0.463	23698	33.011	27327	16.1517782	25264	15.446	7860	396.619
r_100_75_6	100	0.75	0.740	145273	3.508	1.633	131575	33.101	95765	10.6766684	124690	14.873	50885	245.345
r_100_75_7	100	0.75	0.568	110979	18.497	1.228	99363	34.674	106677	10.3770986	103074	20.635	70805	305.484
r_100_75_8	100	0.75	0.132	19453	31.157	0.424	16454	35.255	19120	18.5227401	16499	15.626	9607	229.401
r_100_75_9	100	0.75	0.566	104120	31.104	1.245	82818	34.373	90899	9.05056834	92847	14.373	59374	235.803
r_100_75_10	100	0.75	0.754	143740	12.173	1.647	131652	33.679	142213	7.28371167	125184	15.979	62882	177.158
r_100_100_1	100	1	0.283	81978	30.670	0.720	70201	34.705	82064	10.8005917	79073	15.442	37273	186.862
r_100_100_2	100	1	0.757	190267	30.700	1.677	175736	36.369	164700	12.7350283	174102	14.808	88953	296.050
r_100_100_3	100	1	0.889	225434	30.679	1.256	208476	33.822	172957	17.8171499	203276	18.296	80228	359.589
r_100_100_5	100	1	0.890	230076	30.778	1.268	222907	34.982	122000	12.5483854	210622	15.429	64663	505.067
r_100_100_5	100	1	0.890	230076	30.554	1.268	222907	34.859	167658	10.0338805	215830	15.222	64617	321.373
r_100_100_6	100	1	0.296	74358	30.596	0.794	67677	34.036	73628	8.12566853	68825	15.206	33365	617.844
r_100_100_7	100	1	0.041	10330	30.537	0.133	8165	33.908	10713	6.98741198	10713	14.526	1158	261.131
r_100_100_8	100	1	0.245	62582	30.641	0.649	52048	36.349	62366	6.13812065	57191	14.747	38569	458.270
r_100_100_9	100	1	0.917	232754	30.560	1.312	225005	36.363	178168	13.6318729	209427	15.467	86312	393.194
r_100_100_10	100	1	0.748	193262	31.288	1.578	182226	36.309	150039	8.74122953	185054	15.638	85131	339.456

Table A.1: Computational results for n=100, P=10

I	n	d	r	S_{qkp}	t_{qkp}	p	S_{qubo}	t_{qubo}	S_{anl}	t_{anl}	S_{hy}	t_{hy}
r_200_25_1	200	0.25	0.777	204401	31.276	1.718	158521	42.34	152873	18.840	168143	17.04
r_200_25_2	200	0.25	0.914	239573	5.419	1.222	224572	40.40	128793	37.759	225211	17.53
r_200_25_2	200	0.25	0.914	239573	8.044	1.222	224572	47.22	133463	23.160	212067	16.11
r_200_25_4	200	0.25	0.852	222361	19.601	1.246	183877	40.48	135270	23.032	194444	18.01
r_200_25_5	200	0.25	0.724	187324	26.108	1.577	142781	39.99	123495	6.460	152849	17.93
r_200_25_6	200	0.25	0.327	80086	32.005	0.907	34713	39.74	63924	17.109	63120	20.31
r_200_25_7	200	0.25	0.234	58921	32.755	0.635	23240	47.44	43285	21.846	45913	18.27
r_200_25_8	200	0.25	0.569	149433	12.581	1.200	83335	45.30	97924	33.291	121075	21.22
r_200_25_9	200	0.25	0.195	49366	26.437	0.634	11536	47.57	33523	37.992	37995	16.88
r_200_25_10	200	0.25	0.186	48459	8.719	0.607	11591	44.83	37104	12.530	36933	20.30
r_200_50_1	200	0.5	0.716	372097	6.368	1.537	281200	38.54	316053	15.124	312736	21.40
r_200_50_2	200	0.5	0.427	209061	32.299	0.888	116028	46.67	158726	13.889	190448	17.89
r_200_50_3	200	0.5	0.432	227185	32.300	0.861	122138	39.20	166775	12.491	207659	18.81
r_200_50_4	200	0.5	0.431	228572	4.264	0.864	127100	36.17	160530	10.767	204131	20.69
r_200_50_5	200	0.5	0.940	479651	31.340	1.370	432170	38.21	370958	4.112	405512	18.02
r_200_50_6	200	0.5	0.852	425849	31.314	1.209	360587	37.78	326335	5.796	394907	18.25
r_200_50_7	200	0.5	0.449	219947	31.948	0.957	125572	41.43	191317	30.243	196003	18.10
r_200_50_8	200	0.5	0.619	317942	21.108	1.328	212715	36.83	157914	4.690	269723	18.03
r_200_50_9	200	0.5	0.219	102256	31.839	0.580	51017	40.32	95740	17.076	93788	20.00
r_200_50_10	200	0.5	0.548	284751	31.845	1.150	212083	40.20	232188	17.226	258281	17.29
r_200_75_1	200	0.75	0.583	442894	32.280	1.218	307956	41.90	263343	8.591	410279	18.23
r_200_75_2	200	0.75	0.403	286538	31.961	0.835	169732	43.86	222546	12.646	269740	17.80
r_200_75_3	200	0.75	0.081	61924	33.742	0.281	19406	41.69	62954	6.187	62907	19.70
r_200_75_4	200	0.75	0.208	128351	32.421	0.581	36265	43.17	119614	9.615	109553	18.71
r_200_75_5	200	0.75	0.171	137701	32.476	0.550	52619	40.81	128950	14.684	136249	17.93
r_200_75_6	200	0.75	0.290	229552	35.599	0.764	116751	47.87	167799	17.900	223501	17.21
r_200_75_7	200	0.75	0.342	269887	32.913	0.944	128246	42.31	217130	32.652	255158	18.13
r_200_75_8	200	0.75	0.778	600717	35.491	1.578	510865	50.51	413083	30.516	546796	17.60
r_200_75_9	200	0.75	0.678	516647	40.303	1.460	380561	60.96	433988	25.617	464902	16.83
r_200_75_10	200	0.75	0.185	142694	35.060	0.599	45458	48.66	135087	18.754	137522	19.77
r_200_100_1	200	1	0.931	937149	41.731	1.355	879215	102.87	741746	19.715	890433	18.05
r_200_100_2	200	1	0.268	303007	38.314	0.719	167577	53.92	263130	18.434	301855	22.02
r_200_100_3	200	1	0.035	29296	38.329	0.120	2200	74.68	30353	29.394	30353	17.74
r_200_100_4	200	1	0.096	100837	41.211	0.336	22906	55.43	102089	15.759	102089	20.04
r_200_100_5	200	1	0.778	786402	37.968	1.716	624903	49.82	742723	23.697	717809	20.30
r_200_100_6	200	1	0.040	39981	38.223	0.140	9918	45.54	42346	9.445	42346	18.09
r_200_100_7	200	1	0.691	700966	33.456	1.514	577465	47.52	611518	11.355	657031	20.13
r_200_100_8	200	1	0.763	782121	33.765	1.625	602567	56.62	612027	18.990	717740	18.46
r_200_100_9	200	1	0.627	628948	38.414	1.362	517615	59.74	547596	13.140	605930	21.99
r_200_100_10	200	1	0.365	378237	32.887	1.012	235776	53.54	327497	16.173	370129	18.04

Table A.2: Computational results for n=200, P=10

I	n	d	r	S_{qkp}	t_{qkp}	p	S_{qubo}	t_{qubo}	S_{anl}	t_{anl}	S_{hy}	t_{hy}
r_300_25_1	300	0.25	0.050	29140	15.37	0.179	3922	50.00	26150	14.024	27601	23.18
r_300_25_2	300	0.25	0.496	280747	33.88	1.272	167425	52.83	192416	8.265	243781	24.84
r_300_25_2	300	0.25	0.496	280537	33.48	1.272	167425	52.99	200324	7.824	241254	31.47
r_300_25_4	300	0.25	0.759	444498	33.41	1.406	336499	52.62	345976	10.191	381116	33.96
r_300_25_5	300	0.25	0.026	14988	16.00	0.091	1776	51.26	15144	8.513	15099	31.53
r_300_25_6	300	0.25	0.450	269782	33.59	1.137	129327	47.92	204985	13.233	231912	20.75
r_300_25_7	300	0.25	0.840	484578	33.57	1.563	404520	52.69	415311	16.343	415012	20.93
r_300_25_8	300	0.25	0.015	9343	10.75	0.050	943	52.02	9570	8.904	9570	33.44
r_300_25_9	300	0.25	0.433	250257	33.70	1.066	124038	52.52	146596	7.744	211748	21.79
r_300_25_10	300	0.25	0.659	382626	33.98	1.244	281304	53.36	273131	14.661	296480	22.43
r_300_50_1	300	0.5	0.457	513379	32.66	1.164	300640	43.16	412183	7.868	489193	23.33
r_300_50_2	300	0.5	0.101	105543	32.82	0.346	22650	43.26	79398	5.401	102457	22.41
r_300_50_3	300	0.5	0.768	875788	33.14	1.443	739189	45.19	672528	5.952	775795	23.73
r_300_50_4	300	0.5	0.268	307117	32.44	0.817	164792	44.80	228351	9.060	292752	19.81
r_300_50_5	300	0.5	0.641	727820	33.03	1.164	486055	41.90	615348	5.823	612407	23.87
r_300_50_6	300	0.5	0.657	734053	32.62	1.210	554372	43.32	553163	6.501	604206	20.01
r_300_50_7	300	0.5	0.042	43595	32.06	0.148	3066	40.46	44020	5.958	43760	20.38
r_300_50_8	300	0.5	0.677	767977	32.04	1.223	558280	40.16	614623	4.456	664985	22.70
r_300_50_9	300	0.5	0.667	761351	31.97	1.187	553928	41.10	447389	4.742	665653	20.88
r_300_50_10	300	0.5	0.871	996070	31.97	1.706	889418	41.42	811339	6.745	922592	20.78

Table A.3: Computational results for n=300, P=10

I	S_{anl}	t_{anl}	S_{hy}	t_{hy}
r_100_25_1	18515	59.60	17134	11.16
r_100_25_2	51741	90.85	47254	15.44
r_100_25_3	3837	48.68	3689	15.23
r_100_25_4	43361	107.90	43567	16.18
r_100_25_5	53601	90.19	56466	9.40
r_100_25_6	36022	76.48	32531	8.74
r_100_25_7	14669	56.68	13706	15.63
r_100_25_8	20377	51.39	18734	15.42
r_100_25_9	34442	102.83	30238	14.01
r_100_25_10	24593	84.39	21936	15.74
r_100_50_1	70398	51.22	79010	10.77
r_100_50_2	97575	121.41	95821	10.01
r_100_50_3	34404	58.13	33378	9.88
r_100_50_4	100662	97.46	96659	10.29
r_100_50_5	56865	54.60	55036	9.99
r_100_50_6	16152	46.33	16080	10.04
r_100_50_7	52802	61.50	50001	11.75
r_100_50_8	54272	63.36	52149	10.03
r_100_50_9	68668	68.49	65382	11.62
r_100_50_10	82131	90.74	80081	15.48
r_100_75_1	178340	50.65	176629	15.31
r_100_75_2	95215	55.62	93025	13.43
r_100_75_3	62410	56.68	61195	15.45
r_100_75_4	72988	45.99	71324	9.50
r_100_75_5	28110	43.16	28110	15.54
r_100_75_6	144894	58.13	136504	8.68
r_100_75_7	111256	92.97	108443	10.16
r_100_75_8	19776	46.79	19691	10.45
r_100_75_9	104974	89.92	101775	16.26
r_100_75_10	126405	74.64	131447	16.02
r_100_100_1	83557	31.82	83557	10.23
r_100_100_2	190606	55.55	179420	15.40
r_100_100_3	208758	43.93	210230	15.69
r_100_100_5	127317	39.29	223932	15.53
r_100_100_5	203096	43.03	214406	15.66
r_100_100_6	75225	40.38	75225	15.42
r_100_100_7	11101	34.14	11101	10.18
r_100_100_8	63483	33.81	63483	10.77

r_100_100_9	220049	59.12	222887	15.34
r_100_100_10	195495	81.94	184123	15.77

Table A.4: Computational results for n=100, P=4 on SA and Hybrid Solver

I	S_{anl}	t_{anl}	S_{hy}	t_{hy}
r_200_25_1	180334	15.57	180334	15.57
r_200_25_2	219138	15.89	219138	15.89
r_200_25_2	219138	15.89	230162	16.19
r_200_25_4	202220	16.56	202220	16.56
r_200_25_5	165635	16.08	165635	16.08
r_200_25_6	73072	16.75	73072	16.75
r_200_25_7	54139	16.43	54139	16.43
r_200_25_8	140052	29.32	140052	29.32
r_200_25_9	46935	16.52	46935	16.52
r_200_25_10	44882	16.44	44882	16.44
r_200_50_1	342182	16.24	342182	16.24
r_200_50_2	207344	15.37	207344	15.37
r_200_50_3	225575	16.20	225575	16.20
r_200_50_4	227438	17.41	227438	17.41
r_200_50_5	453574	17.12	453574	17.12
r_200_50_6	387425	17.19	387425	17.19
r_200_50_7	217265	37.51	217265	37.51
r_200_50_8	306264	19.07	306264	19.07
r_200_50_9	105251	16.96	105251	16.96
r_200_50_10	281190	17.02	281190	17.02
r_200_75_1	437519	17.02	437519	17.02
r_200_75_2	287669	17.13	287669	17.13
r_200_75_3	63976	17.51	63976	17.51
r_200_75_4	129129	10.48	129129	10.48
r_200_75_5	139986	17.26	139986	17.26
r_200_75_6	231032	18.19	231032	18.19
r_200_75_7	270714	17.53	270714	17.53
r_200_75_8	583161	16.89	583161	16.89
r_200_75_9	495786	10.31	495786	10.31
r_200_75_10	144302	16.22	144302	16.22
r_200_100_1	936373	15.98	936373	15.98
r_200_100_2	306669	15.25	306669	15.25
r_200_100_3	31607	15.91	31607	15.91
r_200_100_4	103303	15.74	103303	15.74
r_200_100_5	758311	17.74	758311	17.74
r_200_100_6	43874	17.51	43874	17.51
r_200_100_7	673124	16.30	673124	16.30
r_200_100_8	759514	17.92	759514	17.92

r_200_100_9	619577	17.40	619577	17.40
r_200_100_10	381630	17.73	381630	17.73

Table A.5: Computational results for n=200, P=4 on SA and Hybrid Solver

I	S_{anl}	t_{anl}	S_{hy}	t_{hy}
r_300_25_1	29634	49.35	29634	18.90
r_300_25_2	251356	35.70	266879	18.21
r_300_25_2	251356	35.70	273279	17.42
r_300_25_4	398414	52.05	411860	16.84
r_300_25_5	15459	44.45	15459	11.39
r_300_25_6	230898	54.41	259059	17.93
r_300_25_7	241797	32.92	460537	16.78
r_300_25_8	9851	39.10	9851	17.36
r_300_25_9	201342	87.91	244308	16.94
r_300_25_10	339417	38.90	324251	16.88
r_300_50_1	471555	107.78	511998	18.65
r_300_50_2	106810	50.69	106810	17.45
r_300_50_3	773710	84.52	830755	17.52
r_300_50_4	308906	45.91	308824	19.13
r_300_50_5	668010	49.88	694503	11.84
r_300_50_6	678858	77.55	672273	18.97
r_300_50_7	44701	39.01	44701	18.31
r_300_50_8	670625	57.57	702453	13.29
r_300_50_9	757115	45.95	731071	18.75
r_300_50_10	611039	40.11	942103	18.14

Table A.6: Computational results for n=300, P=4 on SA and Hybrid Solver

Appendix B

Python Codes

B.1 Function to generate Instances with the benchmark structure

```
    #function that takes QKP parameters as arguments and generates an objective
    and constraint function for QKP problem

def gen(n: int, r:str, d:float):
    """
    n     number of objects
    r     range of linear coefficients
    d     density
    }"""

    #objective function coefficients:

    value = np.zeros((n, n))
    w = np.zeros(n)
    for i in range(n):
        #Quad coeff:
```

```
            value[i][i] = random.randint(1,r)
            for j in range(i):
                #linear coef:
                value[j][i] = random.randint(1,r)

            #constraint function coefficients:
            w[i] = random.randint(1,r/2)
        #print(value)
        #applying the density parameter..
        sparse_matrix = np.array(sparse.random(n, n, density=d, data_rvs=np.ones).
        toarray())
        obj_mat = value * sparse_matrix

        inds = np.triu_indices(len(obj_mat))
        #print(inds)
        obj_mat[(inds[1], inds[0])] = obj_mat[inds]

        #obj_mat = value
        #capacity:
        c = random.randint(r/2,max(r/2,np.sum(w)))
        return obj_mat, w, c
```

B.2 CPLEX models

The code in this section is written with help from [Moa18].

B.2.1 QKP model

```
def cplex_QKP(n, obj_mat, a, b ):
    #CplexMethodforQKP Solve a problem using the solver's Cplex default approach
    to quadratics
```

```
set_I = range(0, n)
set_J = range(0, n)
c = {(i,j): obj_mat[i,j] for i in set_I for j in set_J}


opt_model = cpx.Model('docplex model')
opt_model.set_time_limit(30)
# if x is Binary
xi_vars  = [opt_model.binary_var(name="x_{0}".format(i))
for i in set_I ]

# <= constraints
constraints = opt_model.add_constraint(
ct=opt_model.sum(a[i] * xi_vars[i] for i in set_I) <= b)

objective = opt_model.sum(xi_vars[i] * xi_vars[j] * c[i,j]
                        for i in set_I
                        for j in set_J)

# for maximization

opt_model.maximize(objective)

# solving with local cplex
solution = opt_model.solve()

print(opt_model.statistics)
#print(opt_model.solution.quality_metrics(True))

solution_cost = opt_model.solution.get_objective_value()
print("Status is ", opt_model.solve_details.status)
```

```
    variables_values = []
    for index, dvar in enumerate(opt_model.solution.iter_variables()):
        variables_values.append([dvar.to_string(),solution[dvar]])


    return solution_cost, variables_values
```

B.2.2 QUBO model

```
def cplex_QUBO(QUBO, a, b):

    n_qb = QUBO.shape[0]

    set_I = range(0, n_qb)
    set_J = range(0, n_qb)
    c = {(i,j): QUBO[i,j] for i in set_I for j in set_J}


    opt_model2 = cpx.Model('docplex QUBO model')
    #time_limit
    #opt_model2.set_time_limit(30)

    xi_vars = opt_model2.binary_var_list(n_qb, name="binary_var")
    y_vars = opt_model2.continuous_var_matrix(keys1=n_qb, keys2=n_qb,
    lb=-opt_model2.infinity)

    # constraints


    for i in set_I:
        for j in set_J:
            if i != j:
```

```
            if c[i,j] < 0:
                opt_model2.add_constraint( y_vars[i,j] <= xi_vars[i] )
                opt_model2.add_constraint( y_vars[i,j] <= xi_vars[j] )
            if c[i,j] > 0:
                opt_model2.add_constraint( y_vars[i,j] >= (xi_vars[i]
                + xi_vars[j] - 1) )
                opt_model2.add_constraint( y_vars[i,j] >= 0 )
            else:
                opt_model2.add_constraint( y_vars[i,j] <= xi_vars[i] )
                opt_model2.add_constraint( y_vars[i,j] <= xi_vars[j] )
                opt_model2.add_constraint( y_vars[i,j] >= (xi_vars[i]
                + xi_vars[j] - 1) )
                opt_model2.add_constraint( y_vars[i,j] >= 0 )



objective = opt_model2.sum(( (y_vars[i,j] * c[i,j]))
                        for i in set_I
                        for j in set_J if i !=j) + opt_model2.sum((  (xi_vars[i]
                        * c[i,i] )) for i in set_I)

# for minimization
opt_model2.minimize(objective)

# solving with local cplex
solution = opt_model2.solve()
solution_cost = opt_model2.solution.get_objective_value()
print("Status is ", opt_model2.solve_details.status)


return solution_cost
```