Alma Mater Studiorum · University of Bologna

SCUOLA DI SCIENZE
Dipartimento di Fisica ed Astronomia
Corso di Laurea in Fisica

EMBEDDING PROCESS IN D-WAVE QUANTUM ANNEALER

Relatore:
Prof. Elisa Ercolessi

Presentata da:
Lorenzo Calandra Buonaura

Correlatore:
Dr. Claudio Massimiliano Sanavio

Anno Accademico 2022/2023

Πάντα ρ́ϵῖ

Abstract

Questa tesi fornisce uno studio dei computer quantistici che si basano sul *quantum annealing*, realizzati dall'azienda D-Wave. Per prima cosa si introducono gli aspetti base della computazione quantistica e il concetto di qubit; in seguito si analizza a fondo la struttura e il funzionamento dei quantum annealer. Il focus è in particolare su come funziona il processo di *embedding*, ossia di mappatura, dei problemi sulla QPU e su come questo influenzi lo spettro energetico e la soluzione del problema. Per procedere con lo studio si utilizza un classico problema di ottimizzazione computazionale, detto *knapsack problem*.

Abstract

This thesis provides a study of quantum computers that are based on *quantum annealing*, produced by the company D-Wave. First, the basic aspects of quantum computation and the concept of qubit are introduced; then, the structure and operation of quantum annealers are thoroughly analyzed. The focus is particularly on how the process of *embedding*, i.e. mapping, of problems onto the QPU works and how this affects the energy spectrum and the solution of the problem. A classical computational optimization problem, called the *knapsack problem*, is used to proceed with the study.

Contents

Introduction

The modern incarnation of computer science was announced by the great mathematician Alan Turing in a remarkable 1936 paper. Turing developed in detail an abstract notion of what we would now call a programmable computer, a model for computation now known as the Turing machine, in his honor [1]. Turing's work set the starting point of classical computer science and, since then, more and more powerful and optimized computers have been developed. However, in 1982, Feynman pointed out that there seemed to be essential difficulties in simulating quantum mechanical systems on classical computers, and suggested that building computers based on the principles of quantum mechanics would allow us to avoid those difficulties. In the 1990s several teams of researchers began fleshing this idea out, showing that it is indeed possible to use quantum computers to efficiently simulate systems that have no known efficient simulation on a classical computer and quantum computing began to be a very new but promising field of research, as it still is today. But why quantum computing is such a revolutionary field? Unlike classical computers that use bits to represent information as either a 0 or a 1, quantum computers use the principles of quantum mechanics to work with quantum bits, or qubits. These qubits can exist in superposition, allowing them to represent multiple states simultaneously and perform parallel computations. Moreover, qubits can also be entangled, creating a powerful and interconnected network of information. The potential of quantum computing lies in its ability to tackle complex problems that are intractable for classical computers, opening up new possibilities in cryptography, optimization, drug discovery, and more.

Within the realm of quantum computing, one prominent approach is quantum annealing, which is a specialized technique designed to solve optimization problems. At the heart of quantum annealing lies the concept of adiabatic quantum computing: the adiabatic process involves starting from an initial quantum state, known as the ground state, and gradually transitioning to a final state that encodes the solution to the problem. The system evolves through a process where the qubits are subjected to a slowly changing Hamiltonian, representing the problem's energy landscape. Quantum annealing has shown promise in tackling a variety of optimization problems, such as portfolio optimization, protein folding, and route optimization. However, it is important to note that quantum annealing is not a panacea for all computational challenges. Its effectiveness heavily depends on the specific problem being solved and the structure of its energy landscape; moreover, quantum annealers are not universal quantum computers, as they can only perform specific operations on qubits.

In this thesis we will first of all discuss the properties of qubits, the quantum counterparts of the classical bits: will initially describe a single qubit, its representation and the unitary operations that can be performed on it. We will then move to a system of multiple qubits, where we will introduce the concept of entanglement between qubits, which is one of the core concept of quantum annealers.

In the second chapter, we will describe D-Wave technology. In addition, the quantum adiabatic theorem and the other theoretical aspects that lie behind quantum annealing will be presented. We will explain how it is possible to encode a problem onto the QPU, exploring the Ising and QUBO formulation. Finally, we will focus on the different kinds of topology that are possible for a D-Wave QPU, Chimera, Pegasus and Zephyr, thereby introducing the process of minor-embedding, which is the main focus of this study.

In the third chapter we will concentrate on the embedding process. Firstly, we will study how the solutions change when different Ising coefficients are submitted to the quantum annealer. After that, we will use a basic knapsack problem to highlight how the energy spectrum and the results are influenced by the embedding process and in particular how they are modified when the embedding involves qubit-chains formation. In addition, we will study the energy distribution of some annealing process and compare it to a Boltzmann distribution, in order to find the effective temperature of the system of qubits. At the end, we will present a comparison between the QPU access time and the CPU execution time, highlighting the computational power of quantum annealers.

Chapter 1

Quantum computation

In this chapter we give a brief introduction to quantum computation. We start from the concept of a *qubit*, the quantum unit of information, and its difference from the classical counterpart: in particular we focus on the superposition of classical states, which is crucial in quantum mechanics. After considering the different representation a qubit can have, we will describe which operations are possible with a single qubit and with multiple qubits.

1.1 Quantum bits

The *bit* is the fundamental concept of classical computation and classical information: a bit can be in 2 different states, 0 and 1; we can use multiple bits to describe logic circuits and to solve operations on a classic computer. Quantum computation and quantum information are built upon an analogous concept, the so called *quantum bit*, or *qubit* for short [1]: in this chapter, we will examine the characteristics of single and multiple qubits and discuss the operations that may be performed with them.

What exactly is a qubit? As we said, a qubit is the quantum counterpart of a classical bit, and, as a classical bit, a qubit can have different states: two possible states for a qubit are the states $|0\rangle$ and $|1\rangle$ (which correspond to the states 0 and 1 for a classical bit). But here comes the difference between a bit and a qubit: while a bit only has two possible states, a qubit can be in a state other than $|0\rangle$ and $|1\rangle$, more specifically it can be in any *superposition* of $|0\rangle$ and $|1\rangle$:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \qquad\qquad |\alpha|^2 + |\beta|^2 = 1 \tag{1.1}$$

In other words, the state of a qubit is a vector in a two-dimensional complex vector space, so it is a vector in an Hilbert space with dimension $d = 2$: the simpler Hilbert space we can think of is $\mathbb{C}^2$. The special states $|0\rangle$ and $|1\rangle$ are the two fundamental vectors we use to describe a general state $|\psi\rangle$: this is why they form a orthonormal basis of the Hilbert space called *computational basis*, $B = \{|0\rangle, |1\rangle\}$. Using the vector representation, the computational basis is given as:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \rightarrow \quad |\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

Another difference between a bit and a qubit is that the latter follows quantum mechanics laws, while the former follows the principles of classical mechanics: this means that we can examine a bit to determine whether it is in the state 0 or 1. Rather remarkably, we cannot examine a qubit to determine its quantum state, meaning we cannot determine the values of α and β; instead, quantum mechanics tells us that we can only acquire much more restricted information about the quantum state. The numbers α and β are complex numbers: when we measure a qubit, we get either the result 0, with probability $|\alpha|^2$, or the result 1, with probability $|\beta|^2$. This is why we require the condition $|\alpha|^2 + |\beta|^2 = 1$, since the probabilities must always sum to one; geometrically, we can interpret this as the condition that the qubit's state be normalized to length 1. Thus, in general, a qubit's state is a unit vector in a two-dimensional Hilbert space.

1.1.1 Geometrical representation

It is useful to introduce a geometric representation of a qubit; this will give us a better understanding of what operations with a single qubit look like. First of all, we know that α and β are complex numbers, meaning we can use a polar representation:

$$|\psi\rangle = e^{i\phi_1}|\alpha|\,|0\rangle + e^{i\phi_2}|\beta|\,|1\rangle = e^{i\phi_1}\left[|\alpha|\,|0\rangle + e^{i(\phi_2-\phi_1)}|\beta|\,|1\rangle\right]$$

We can *ignore* the factor $e^{i\phi_1}$ in front because it's only a phase of our state, which has no observable effects, so we can drop it and write:

$$|\psi\rangle = \cos\frac{\theta}{2}\,|0\rangle + e^{i\phi}\sin\frac{\theta}{2}\,|1\rangle \qquad \phi \in [0, 2\pi], \quad \theta \in [0, \pi] \tag{1.2}$$

We obtain that our state can be defined by two angles (instead of the two complex coefficients α and β): geometrically, we can represent a generic state on a sphere with

$r = 1$, called *Bloch sphere* (see Figure 1.1). This representation provides a useful mean of visualizing the state of a single qubit, and often serves as an excellent testbed for ideas about quantum computation and quantum information. Many of the operations on single qubits which we describe are neatly described within the Bloch sphere picture. However, it must be kept in mind that this intuition is limited, because there is no simple generalization of the Bloch sphere known for multiple qubits [1].

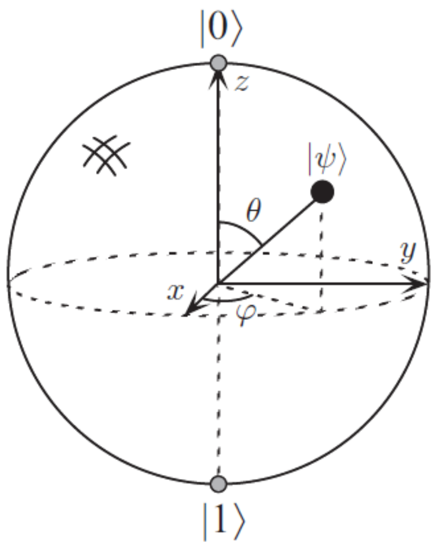


Figure 1.1: Bloch sphere representation of a qubit [1].

How much information is represented by a qubit? There are an infinite number of points on the unit sphere, so that in principle one could store an entire text or book in the infinite binary expansion of θ. However, this conclusion turns out to be misleading, because of the behavior of a qubit when observed, meaning measured: in fact, the measurement of a qubit will give only either the state $|0\rangle$ or $|1\rangle$, with different probabilities. Furthermore, measurement changes the state of a qubit, collapsing it from its superposition of $|0\rangle$ and $|1\rangle$ to the specific state consistent with the measurement result [1]. For example, if measurement of $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ gives $|0\rangle$, then the post-measurement state of the qubit will be $|0\rangle$. Why does this type of collapse occur? Nobody knows; this behavior is simply one of the *fundamental postulates* of quantum mechanics.

1.1.2 Single qubit operations

What operations can a quantum computer perform upon a single qubit? The reversible operations that a quantum computer can perform upon a single qubit are represented by the action of any linear transformation that takes unit vectors into unit vectors, due to the probability interpretation of the coefficients α and β [2]. This means that every operator **U** which operates on a single qubit is unitary:

$$UU^\dagger = U^\dagger U = I$$

Consequently, for a single qubit we can represent any operator as a 2×2 unitary matrix, so a generic operator, called *gate*, can be written as:

$$U = \begin{pmatrix} a & b \\ -e^{i\varphi}b^* & e^{i\varphi}a^* \end{pmatrix} \qquad \text{with } |a|^2 + |b|^2 = 1 \tag{1.3}$$

Some of the most used single qubit gates are:

- Pauli matrices:

$$\sigma_X \equiv X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \sigma_Y \equiv Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad \sigma_Z \equiv Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

- Hadamard gate:

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

Moreover, if we think about the geometrical description of a qubit, it is easy to understand that any operation on a qubit is no different than a rotation on the Bloch sphere. This means we can perform infinite operations on a single qubit in order to prepare it in the state we want it to be. Thus, the most generic single qubit gate can also be written as:

$$U = e^{i\alpha} e^{i\frac{\theta}{2}\hat{n}\cdot\vec{\sigma}} = e^{i\alpha}\left(\cos\frac{\theta}{2}\mathbb{I} + i\sin\frac{\theta}{2}\hat{n}\cdot\vec{\sigma}\right) \tag{1.4}$$

where $\vec{\sigma} = (\sigma_X, \sigma_Y, \sigma_Z)$, and it represents a rotation on the Bloch sphere by an angle θ about the axis $\hat{n} = (n_x, n_y, n_z)$.

1.2 Multiple qubits

Now it is in our interest to understand how to analytically cope with multiple qubits: in particular we want to understand which kind of operations are possible with multiple qubits. As previously described, each qubit can be thought as a vector in a 2-dimensional Hilbert space: this means that multiple qubits belong to different Hilbert spaces and if we want to consider multiple qubits at the same time we need to expand our Hilbert space. In order to do that, we need to carry out the so called *tensor product* of the Hilbert spaces of our qubits. Here we will further explain what this product is and how it works.

1.2.1 Tensor product

Let V, W be Hilbert spaces defined over the same field. We define *tensor product* of V and W a Hilbert space to which is associated a bilinear map $V \times W \to V \otimes W$ that maps a pair (v, w), $v \in V$ and $w \in W$, to an element of $V \otimes W$ denoted by $v \otimes w$, called the tensor product of v and w [3]. This Hilbert space is constructed in order to satisfy two properties:

- If U is an Hilbert space on the same field and $g : V \times W \in U$ is a bilinear map, then $\exists!$ a linear map $g_* : V \otimes W \in U$ such that, for each pair (v, w) where $v \in V$ and $w \in W$, we have $g(v, w) = g_*(v \otimes w)$.
- If $\{v_1, \dots, v_n\}$ and $\{w_1, \dots, w_n\}$ are basis respectively of V and W, then the elements $v_i \otimes w_j$, with $i, j = 1, \dots, n$, form a basis of $V \otimes W$.

The consequences of these two properties are:

- If V and W are vectors spaces of finite dimension, then also the space $V \otimes W$ is finite-dimensional, and its dimension is the product of the dimensions of V and W: $dim(V \otimes W) = dim(V) \times dim(W)$. This results from the fact that a basis of $V \otimes W$ is formed by taking all tensor products of a basis elements of V and a basis elements of W.
- The tensor product is associative in the sense that, given three vector spaces U, V, W, there is a canonical isomorphism that maps $(u \otimes v) \otimes w$ to $u \otimes (v \otimes w)$. This holds for the product of vectors as well, which means that $(u \otimes v) \otimes w = u \otimes (v \otimes w)$.
- The tensor product of two vector spaces V, W is commutative in the sense that there is a canonical isomorphism that maps $v \otimes w$ to $w \otimes v$. On the other hand,

even when $V = W$, the tensor product of vectors is not commutative; that is $v \otimes w \neq w \otimes v$, in general.

As previously highlighted, we are interested in linear maps that represent operations on single qubits. If we have multiple qubits, we want to perform operations on all of them, and not necessarily the same operations. So we are most interested in the tensor product of linear maps. Given a linear map $f\colon U \to V$ and a vector space W, the tensor product $f \otimes W\colon U \otimes W \to V \otimes W$ is the unique linear map such that $(f \otimes W)(u \otimes w) = f(u) \otimes w$. The tensor product $W \otimes f$ is defined similarly. Given two linear maps $f\colon U \to V$ and $g\colon W \to Z$ their tensor product $f \otimes g\colon U \otimes W \to V \otimes Z$ is the unique linear map that satisfies $(f \otimes g)(u \otimes w) = f(u) \otimes g(w)$. This is an example of the composition of two tensor product, because one has $f \otimes g = (f \otimes Z) \circ (U \otimes g) = (V \otimes g) \circ (f \otimes W)$.

By choosing bases of all vector spaces involved, the linear maps A and B can be represented by matrices. Then, depending on how the tensor $v \otimes w$ is vectorized, the matrix describing the tensor product $A \otimes B$ is the Kronecker product of the two matrices [3]. For example, if V, W, X and Y are all two-dimensional and bases have been fixed for all of them, and $A : V \to W$ and $B : X \to Y$ are given by the matrices:

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix}, \qquad B = \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix},$$

respectively, then the tensor product of these two matrices is computed as follows:

$$A \otimes B = \begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \otimes \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} = \begin{pmatrix} a_{1,1} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} & a_{1,2} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} \\ a_{2,1} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} & a_{2,2} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} \end{pmatrix}$$

$$\rightarrow \quad A \otimes B = \begin{pmatrix} a_{1,1}b_{1,1} & a_{1,1}b_{1,2} & a_{1,2}b_{1,1} & a_{1,2}b_{1,2} \\ a_{1,1}b_{2,1} & a_{1,1}b_{2,2} & a_{1,2}b_{2,1} & a_{1,2}b_{2,2} \\ a_{2,1}b_{1,1} & a_{2,1}b_{1,2} & a_{2,2}b_{1,1} & a_{2,2}b_{1,2} \\ a_{2,1}b_{2,1} & a_{2,1}b_{2,2} & a_{2,2}b_{2,1} & a_{2,2}b_{2,2} \end{pmatrix}$$

1.2.2 Multiple qubits representation

Just as the general state of a single qubit is any normalized superposition of the two possible classical states (see Eq. (1.1)), the general state that nature allows us to associate with two qubits is any normalized superposition of the four orthogonal classical states [2]:

$$|\Psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle = \begin{pmatrix} \alpha_{00} \\ \alpha_{01} \\ \alpha_{10} \\ \alpha_{11} \end{pmatrix} \tag{1.5}$$

with the complex amplitudes being constrained only by the normalization condition

$$|\alpha_{00}|^2 + |\alpha_{01}|^2 + |\alpha_{10}|^2 + |\alpha_{11}|^2 = 1$$

The measurement of $|\Psi\rangle$ can have four different results: $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ with probabilities given by $|\alpha_x|^2$. After the measurement, the system collapses in the state that is measured; for example, if we measure $|00\rangle$, the final state of our system will be $|\Psi\rangle = |00\rangle$. If we measure only one of the two qubits, for example the first one and we obtain $|0\rangle$, then the final state of the system will be:

$$|\Psi\rangle = \frac{\alpha_{00}|00\rangle + \alpha_{01}|01\rangle}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}}$$

where the coefficient is changed in order to respect the normalization condition. The four states represent an orthonormal basis of the total Hilbert space of the two qubits, called *computational basis*. The two qubits system will be later thoroughly studied, so it is easier to name these states in order to lighten the notation, as follows:

$$|00\rangle = e_1 \qquad |01\rangle = e_2 \qquad |10\rangle = e_3 \qquad |11\rangle = e_4 \tag{1.6}$$

This generalizes in the obvious way to n qubits, whose general state can be any superposition of the 2^n different classical states, with amplitudes whose squared magnitudes sum to unity:

$$|\Psi\rangle = \sum_{0\leq x\leq 2^n} \alpha_x |x\rangle_n \qquad \text{with} \sum_{0\leq x\leq 2^n} |\alpha_x|^2 = 1 \tag{1.7}$$

where $|x\rangle_n$ represents the classical state $|0\dots1\dots0\rangle$ with the 1 in the n-position. The set of 2^n classical states is formed considering all the possible tensor products of n individual qubit states $|0\rangle$ and $|1\rangle$ and it is called computational basis in this case as well.

If we have two qubits, one in the state $|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle$, $|\psi\rangle \in H_1$ and the other in the state $|\phi\rangle = \beta_0 |0\rangle + \beta_1 |1\rangle$, $|\phi\rangle \in H_2$, then the state $|\psi\phi\rangle$ of the pair is taken to be the tensor product of the individual states [2]:

$$|\psi\phi\rangle = |\psi\rangle \otimes |\phi\rangle = \alpha_0\beta_0 |00\rangle + \alpha_0\beta_1 |01\rangle + \alpha_1\beta_0 |10\rangle + \alpha_1\beta_1 |11\rangle = \begin{pmatrix} \alpha_0\beta_0 \\ \alpha_0\beta_1 \\ \alpha_1\beta_0 \\ \alpha_1\beta_1 \end{pmatrix} \tag{1.8}$$

and the space where $|\psi\phi\rangle$ lives in is $H_1 \otimes H_2$. Note that a general 2-qubit state is in this form if and only if $\alpha_{00}\alpha_{11} = \alpha_{01}\alpha_{10}$. Since the four amplitudes in Eq. (1.8) are constrained only by the normalization condition, this relation need not hold, and the general 2-qubit state is *not* a product of two single qubit states. The same is obviously true for states of n qubits: a general state of n Qbits is a superposition of the 2^n product states that form the canonical basis and cannot, in general, be expressed as a product of any set of 1-Qbit states. Such nonproduct states of two or more qubits are called *entangled* states. For clarity, here is an example of a product state and of an entagled state of two qubits:

$$|\Psi\rangle = \frac{|00\rangle + |01\rangle}{\sqrt{2}} = |0\rangle \frac{|0\rangle + |1\rangle}{\sqrt{2}} = |\psi\rangle \otimes |\phi\rangle \rightarrow \text{product state}$$

$$|\Psi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}} \neq |\psi\rangle \otimes |\phi\rangle \rightarrow \text{entangled state}$$

But why do we call them entangled states? The reason is that if we know we have two entangled qubits, the measurement of one of them immediately gives us also the state of the other. If we consider the two examples we just made, in the product state a measurement of the first qubit will always gives us the state $|0\rangle$ and the second one can be both in the states $|0\rangle$ and $|1\rangle$. Instead, in the entangled state a measurement of the first qubit can give $|0\rangle$ or $|1\rangle$ with the same probability: if we measure $|0\rangle$, then the final state is $|\Psi\rangle = |00\rangle$ and if we measure $|1\rangle$, then $|\Psi\rangle = |11\rangle$, so we automatically know the state of the second qubit, because the two are entangled, i. e. related.

1.2.3 Multiple qubits gates

Now we can generalize the operations on multiple qubits. The first requirement that has to be respected by our n-qubits operator $\mathbf{U}$, just like in the case of a single qubit gate, is that it must be unitary:

$$UU^\dagger = U^\dagger U = I$$

The most important 2-qubits gate is the controlled-NOT (CNOT) gate: this gate has two input qubits, known as the *control qubit* and the *target qubit*, respectively. The action of the gate may be described as follows: if the control qubit is set to 0, then the target qubit is left alone, while if the control qubit is set to 1, then the target qubit is flipped [1]. In equations:

$$\begin{cases} |00\rangle \to |00\rangle \\ |01\rangle \to |01\rangle \\ |10\rangle \to |11\rangle \\ |11\rangle \to |10\rangle \end{cases} \quad \to \quad U_{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Another way of describing the CNOT is as a generalization of the classical XOR gate, since the action of the gate may be summarized as $|A, B\rangle \to |A, A \oplus B\rangle$, where $\oplus$ is addition modulo two, which is exactly what the XOR gate does. That is, the control qubit and the target qubit are XOR-ed and stored in the target qubit [1]. In Fig. 1.2 we can see the circuit symbol and representation of the CNOT gate.

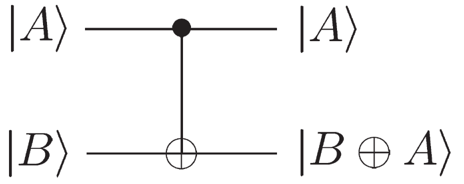


Figure 1.2: Circuit representation for the CNOT gate [1].

Of course, there are many interesting quantum gates other than the CNOT. However, the CNOT and single qubit gates are the prototypes for all other gates because of the following remarkable universality result: *any multiple qubit logic gate may be composed from CNOT and single qubit gates.* This theorem is the quantum parallel of the universality of the NAND gate; proof is given in [1].

Chapter 2

D-Wave technology

This chapter is dedicated to D-Wave quantum annealer, which is the quantum computer studied in this thesis. First of all we describe what is quantum annealing and what are the physical principles this particular quantum computer relies on, using a simple yet effective example of a two-qubits system. Secondly, we describe the model that must be used to formulate a problem that can be submitted to the quantum annealer. At the end, we focus on the topology of a D-Wave QPU and how the qubits are connected; we also introduce the concept of *minor embedding*, which will be later studied in detail in the next chapter.

2.1 Quantum annealing

Quantum computers can be realized following two main paradigms for their architectures: gate model (also known as circuit model) and quantum annealing. Gate-model quantum computing implements compute algorithms with quantum gates, analogously to the use of Boolean gates in classical computers: some example of this implementation are the quantum computer by IBM, which use superconducting chips (transmons), and by PASQAL, which use neutral atoms. The quantum computers that follow this logic are universal, meaning that can ideally perform any n-qubits operation; at the end of the computation the results are obtained performing a measurement on the states of the qubits. On the other hand, quantum annealers, which are currently produced by a single company, named D-Wave, are not universal and operate according to a totally different logic than gate ones. Quantum annealers are based on the so called *adiabatic quantum computation*, which is an application of the *quantum adiabatic theorem*: the system is initialize in a low-energy state and gradually the parameters of a problem we wish to solve are introduced. The slow change makes it likely that the system ends in the lowest-energy state of the problem, which corresponds to an optimal solution. In order to act adiabatically, the system must be isolated from the environment to have no interference from outside; due to no real world computation can run in perfect isolation, adiabatic quantum computation is only a theoretical system and its real world counterpart is quantum annealing. The main difference between them is that at the and of a quantum annealing process, the system could be in a state which is different from the ground state; this is why it is mandatory to repeat the quantum annealing process several times, in order to have a statistics of the results.

2.1.1 Quantum adiabatic theorem

The quantum adiabatic theorem, from Max Born and Vladimir Fock, states that: *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* [4].

What does this mean and how is it applied in quantum annealing? The adiabatic theorem tells us that if we take a system with a discrete spectrum - so there is always have a gap between the eigenvalues - prepare it in its ground state and then let it evolve under a slow-enough perturbation, the system will remain in the ground state: this happens if the ground state does not become degenerate during the process, otherwise there could be a level-crossing and the system could end in a different state. If the adiabatic theorem is respected, in terms of quantum annealing, the final state of the system describes the optimal solution for the problem we are trying to solve, because corresponds with the lowest-energy state. The proof of the theorem can be found in [4]. But how slow is slow-enough for the perturbation? It can be proved that the condition our perturbation must respect is given by:

$$\frac{\hbar \langle\psi_m| \dot{H} |\psi_n\rangle}{\overline{E_n - E_m}^2} \ll 1 \tag{2.1}$$

where E_n and E_m are the eigenvalues associated to the eigenfunctions ψ_n and ψ_m and $\overline{E_n - E_m}$ is the so called minimum energy gap, which is the smallest energy difference between the two levels we are considering (in the present study we will always consider the ground state and one of the excited states) [4]. It is important to notice that the condition depends on the derivative of our Hamiltonian, so what we do not want is that the eigenvalues describing our system's spectrum change too rapidly. This is really important as we will see in quantum annealing; if this condition is not respected the final solution given by the annelaer will not be the optimal solution for our problem, because the system is not in the ground state anymore but has moved during the process to an excited state.

2.1.2 D-Wave qubits

On a quantum annealer, the qubits are encoded in the lowest energy states of the superconducting loops that make up the D-Wave QPU [5]. These states have a circulating current and a corresponding magnetic field, that can be $|\uparrow\rangle$, $|\downarrow\rangle$ or any superposition of those two: we identify the computational basis as $|\downarrow\rangle = |0\rangle$ and $|\uparrow\rangle = |1\rangle$, as we can see in Fig. 2.1. At the end of the quantum annealing process, each qubit collapses from a superposition state into either $|0\rangle$ or $|1\rangle$ (a classical state).

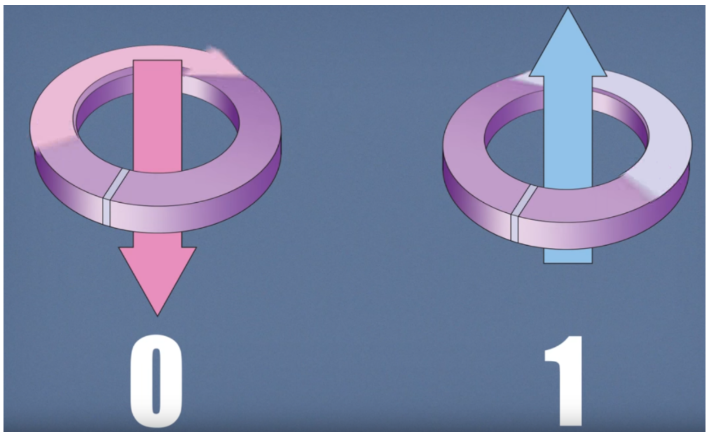


Figure 2.1: D-Wave superconducting loops and corresponding magnetic field, which encode the qubits in the quantum annealer [5].

How does quantum annealing work from a physical point of view? Is it useful to consider an energy diagram, first of all in the simplest case, with only one qubit (see Fig. 2.2). This diagram changes over time, as shown in (a), (b), and (c). To begin, there is just one valley (a), with a single minimum, because the qubit is prepared in

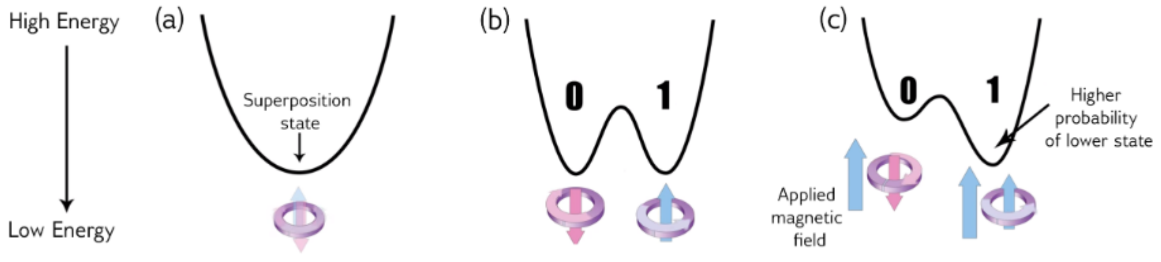


Figure 2.2: Energy diagram changes over time [5].

a superposition of $|0\rangle$ and $|1\rangle$. The quantum annealing process runs, the potential barrier is raised, and this turns the energy diagram into what is known as a double-well potential (b). Here, the low point of the left valley corresponds to the $|0\rangle$ state, and the low point of the right valley corresponds to the $|1\rangle$ state; at the end of the annealing process we will find the qubit in one of these two valleys. The moment that we can control is how the potential barrier is formed: in particular, we can control the probability of the qubit falling into the $|0\rangle$ or the $|1\rangle$ state by applying an external magnetic field to the qubit (c): this tilts the double-well potential, increasing the probability of the qubit ending up in the lower well. The programmable quantity that controls the external magnetic field is called a bias, and the qubit minimizes its energy in the presence of the bias [5].

The bias term alone is not useful, however; the real power of the qubits emerges when they are linked together so they can influence each other, in other words, when the qubits are entangled. This is done with a device called a coupler, which can make two qubits tend to end up in the same state or it can make them tend to be in

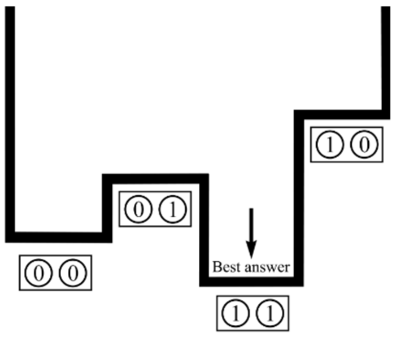


Figure 2.3: Final energy landscape for a two qubits system [5].

opposite states. As previously described, two entangled qubits can be thought of as a superposition of four different states, meaning that they will form a well with four minima during the annealing process (see Fig. 2.3); similarly to what happens with a single qubit, we can use a coupler to make one of these minima lower than the others. During the anneal, the qubit states are potentially delocalized in this landscape before finally settling into the lowest-energy state.

As stated, each qubit has a bias and qubits interact via the couplers: when formulating a problem, users choose values for the biases and couplers, so, together, the programmable biases and weights are the means by which a problem is defined in the D-Wave quantum annealer. They in fact define an energy landscape, and the D-Wave quantum computer finds the minimum energy of that landscape: this is quantum annealing.

2.1.3 Mathematical description

To describe the system mathematically, we use its Hamiltonian; for a D-Wave quantum annealer, the Hamiltonian may be represented as an Ising Hamiltonian (see Sec. 3.1) and its general form is:

$$H(s) = -\frac{A(s)}{2}\left(\sum_i \hat{\sigma}_x^{(i)}\right) + \frac{B(s)}{2}\left(\sum_i h_i \hat{\sigma}_z^{(i)} + \sum_{i>j} J_{ij}\hat{\sigma}_z^{(i)}\hat{\sigma}_z^{(j)}\right) \tag{2.2}$$

where $\hat{\sigma}_{x,z}^{(i)}$ are Pauli matrices operating on a single qubit q_i, while h_i and J_{ij} are respectively the single qubits biases and the coupling strengths between qubit q_i and qubit q_j [5]. As we can see, the Hamiltonian is the sum of two different terms:

- **Initial Hamiltonian**

  $$H_i = \sum_i \hat{\sigma}_x^{(i)} \tag{2.3}$$

  This term is the Hamiltonian of the system prepared in the ground state before the quantum annealing starts. The lowest energy state of this term is when all the qubits are in a superposition of the states $|0\rangle$ and $|1\rangle$; in particular in a D-Wave quantum computer we will always have that the starting situation is with all the qubits in the $|+\rangle$ state, which corresponds to:

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

  This term can also be addressed as the tunneling Hamiltonian.

- **Final Hamiltonian**

$$H_f = \sum_i h_i \hat{\sigma}_z^{(i)} + \sum_{i>j} J_{ij} \hat{\sigma}_z^{(i)} \hat{\sigma}_z^{(j)} \qquad (2.4)$$

  This term is the Hamiltonian that contains our problem, encoded through the values of the biases h_i and the coupling strengths J_{ij}; this is why it is called the problem Hamiltonian. The lowest energy state of this term is the answer to the problem we are trying to solve; the final state of our system is always a classical state (a superposition may occur only if there is more than one optimal solution) and depending in which classical state the system ends we have a different solution for our problem.

During a quantum annealing process, the system begins in the lowest-energy eigenstate of the initial Hamiltonian H_i (which is, as we already said, the state whit all the qubits in the $|+\rangle$ state); as it anneals, it introduces the problem Hamiltonian H_f with biases and coupling strengths and reduces the influence of the initial Hamiltonian; at the end of the anneal we find the system in an eigenstate of the problem Hamiltonian. Ideally, the system has stayed in the ground state throughout the entire process so

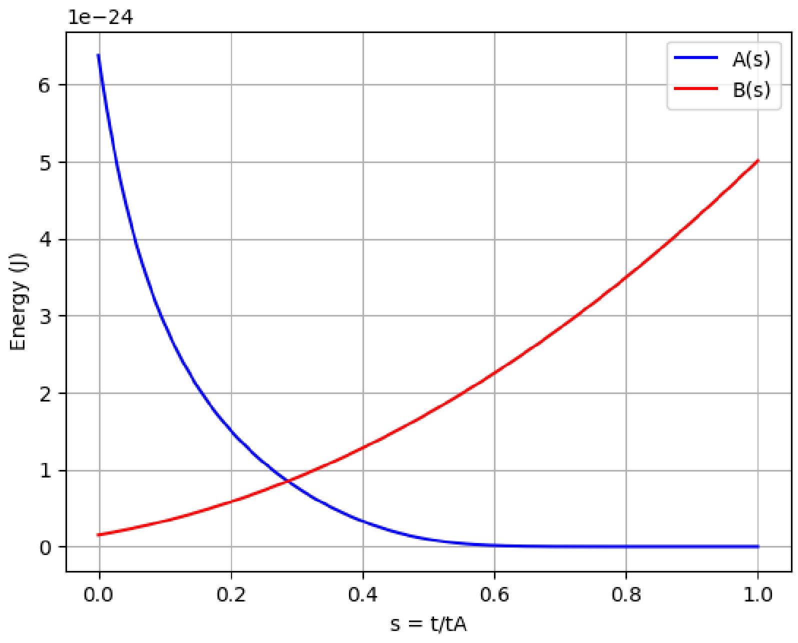


Figure 2.4: A(s) and B(s) for an Advantage_system6.2 QPU, expressed in *Joule*.

that, by the end, it is in the minimum energy state of the problem Hamiltonian and therefore has an answer to the problem we are trying to solve.

The changes in the Hamiltonian are introduced through two parameters, $A(s)$ and $B(s)$, where s is an abstract parameter ranging from 0 to 1, called *normalized anneal fraction* (see Fig. 2.4). In general s depends on the type of the process considered; a linear anneal sets $s = t/t_{an}$, where t_{an} is the total time of the anneal and is specific for each QPU [5]. When we perform an annealing process we have to set the annealing time correctly: if it is too small, the transition will be too fast, and sometimes the system will end up in a state different from the ground state, because the condition expressed in Eq. (2.1) is violated. As we can see, when $t = 0$, $A(0) \gg B(0)$, which leads to the quantum ground state of the system. As the system is annealed, $A(s)$ decreases and $B(s)$ increases until $s = 1$, when the final state of the qubits represents a low-energy solution.

2.1.4 Energy landscape

As previously explained, quantum annealing works if the condition of the quantum adiabatic theorem is respected: we can better understand this condition if we take a look at the eigenspectrum and how it varies during the anneal.

At the beginning, the system starts in the ground state, and it is well separated from any other energy level. As the problem Hamiltonian is introduced ($A(s)$ decreases while $B(s)$ increases), the higher energy levels and the ground state may get closer: as we can see in Fig. 2.5, which represents how the energy levels change during the annealing, there is a minimum energy gap that can become smaller, near 0. This term is the denominator of the condition in Eq. (2.1), so if it becomes too small the probability that the system will jump from the lowest energy state into one of the excited states increases rapidly. The system can jump from the ground state into a higher energy state due to many factors such as thermal fluctuations that exist in any physical system, or because we are running the annealing process too quickly. To prevent this from happening, two conditions must be satisfied: firstly, that the process is executed with no interference from outside energy sources and secondly that the Hamiltonian is changed slowly enough.

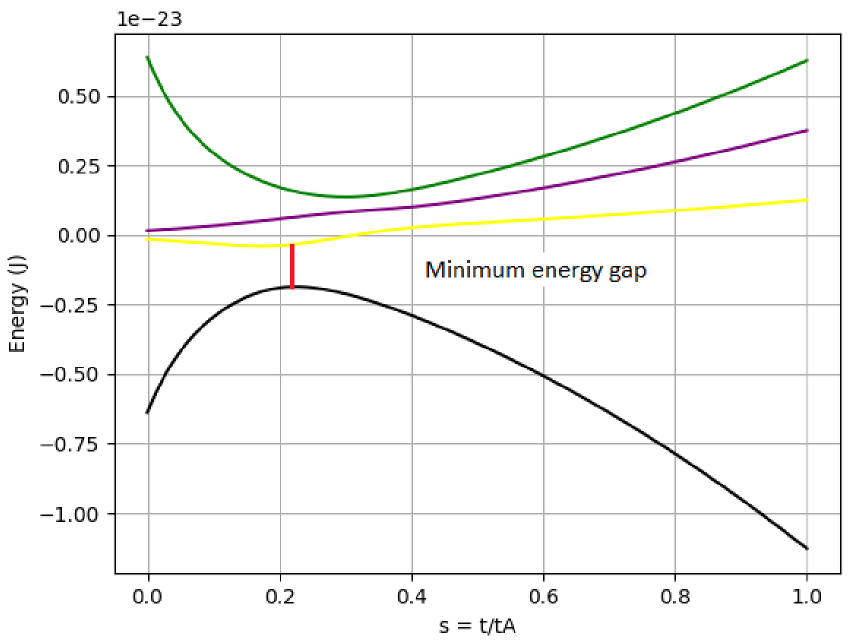


Figure 2.5: Eigenspectrum: the lowest line (black) is the ground state.

2.1.5 Two qubits example

In order to better understand how the quantum anneling works from an analytical point of view, it is useful to consider a trivial example of a system with two qubits and underline the various steps that our annealing procedure is made of. The two qubits will be prepared in the $|+\rangle$ state and after the anneling we will find our system in one of the classical eigenstates of the final Hamiltonian: we have four possibilities, e_1, e_2, e_3 and e_4 (see Eq. (1.6)) . Running the anneling many times will return that one of these final states is more frequent than the others, and that particular state will be the solution to the problem we are trying to solve. This example will describe all the steps that are mathematically required in order to obtain the eigenvectors and eigenvalues of the Ising Hamiltonian and that later will be taken for granted:

- First of all, for a limited number of qubits, we can explicitly write the Hamiltonian of our system (when the number of qubits increases it is not convenient to study the system analytically); if we have two qubits the initial and the problem Hamiltonians read out as:

$$H_i = \hat{\sigma}_x^{(1)} \otimes I^{(2)} + I^{(1)} \otimes \hat{\sigma}_x^{(2)} \tag{2.5}$$

$$H_f = h_1\,\hat{\sigma}_z^{(1)} \otimes I^{(2)} + h_2\,I^{(1)} \otimes \hat{\sigma}_z^{(2)} + J_{12}\,\hat{\sigma}_z^{(1)} \otimes \hat{\sigma}_z^{(2)} \tag{2.6}$$

- Secondly, we resolve the tensor products in Eq (2.5) e (2.6); we have 2×2 matrices, so the results will be 4×4 matrices:

$$\hat{\sigma}_x^{(1)} \otimes I^{(2)} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

$$I^{(1)} \otimes \hat{\sigma}_x^{(2)} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$\hat{\sigma}_z^{(1)} \otimes I^{(2)} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

$$I^{(1)} \otimes \hat{\sigma}_z^{(2)} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

$$\hat{\sigma}_z^{(1)} \otimes \hat{\sigma}_z^{(2)} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Then, we compute the two Hamiltonians in Eq (2.5) e (2.6), introducing the biases h_1 and h_2 and the coupling strength J_{12}:

$$H_i = \hat{\sigma}_x^{(1)} \otimes I^{(2)} + I^{(1)} \otimes \hat{\sigma}_x^{(2)} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

$$H_f = \begin{pmatrix} h_1 + h_2 + J_{12} & 0 & 0 & 0 \\ 0 & h_1 - h_2 - J_{12} & 0 & 0 \\ 0 & 0 & -h_1 + h_2 - J_{12} & 0 \\ 0 & 0 & & -h_1 - h_2 + J_{12} \end{pmatrix}$$

- We obtain a final 4×4 matrix as the Ising Hamiltonian using Eq. (2.2):

$$H_{ising}(s) = -\frac{A(s)}{2}\, H_i + \frac{B(s)}{2}\, H_f = \tag{2.7}$$

$$= \frac{1}{2} \begin{pmatrix} B(s)[h_1 + h_2 + J_{12}] & -A(s) & -A(s) & 0 \\ -A(s) & B(s)[h_1 - h_2 - J_{12}] & 0 & -A(s) \\ -A(s) & 0 & B(s)[-h_1 + h_2 - J_{12}] & -A(s) \\ 0 & -A(s) & -A(s) & B(s)[-h_1 - h_2 + J_{12}] \end{pmatrix}$$

- Now, we want to find the eigenvalues of our $H_{ising}(s)$ in order to find the ground state, which is the state we want our system to stay into. The eigenvalue calculation depends on the values of the biases and the coupling strength; what we need to do is to encode our problem in those values in order to obtain a solution for our problem. How to decide those values (in which is encoded the problem we want to solve) will be discussed later(see Sec. 2.2); here we will take as example $h_1 = 0.1$, $h_2 = 0.2$ and $J_{12} = -0.4$ and use the values of an Advantage_system6.2 QPU annealing schedule for $A(s)$ and $B(s)$. Our Hamiltonian will be:

$$H_{ising}(s) = \frac{1}{2} \begin{pmatrix} -0.1\, B(s) & -A(s) & -A(s) & 0 \\ -A(s) & 0.3\, B(s) & 0 & -A(s) \\ -A(s) & 0 & 0.5\, B(s) & -A(s) \\ 0 & -A(s) & -A(s) & -0.7\, B(s) \end{pmatrix}$$

- Then I wrote a Python program, using the `numpy` and `matplotlib.pyplot` libraries, to calculate the eigenvalues as they change over time and plot them; the result can be seen in Fig. 2.6, where the ground state is bound to 0 (black line) and the other lines represent the excited states. The minimum energy gap is $\Delta E = 3.73 \cdot 10^{-25}\ J$ and occurs when $s = 0.3933934$: we will discuss later how the band gap changes when increasing the number of qubits.

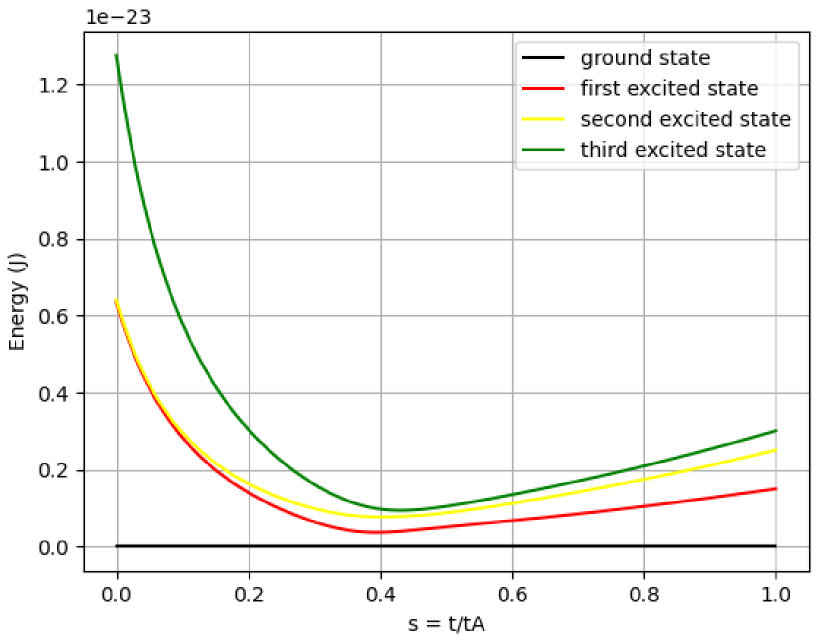


Figure 2.6: Eigenvalues evolution over time of a two qubits system with $h_1 = 0.1$, $h_2 = 0.2$ and $J_{12} = -0.4$ as they change over time, normalized to the ground state.

- Finally, we can also obtain the final eigenvectors of our system: these, in particular the eigenvector corresponding to the ground state, will tell us the real solution of the problem we are solving. In this particular case, we obtained the values reported in Tab. 2.1, where e_0 is the ground state and the others e_i are the excited states. Our system is made by two qubits so each eigenstate has an expansion in the computational basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, which as in Eq. (1.5) can be written in a four-element vector. This expansion is possible because the eigenstate of the final

	gs	$1e$	$2e$	$3e$
e_1	0.00	1.00	0.00	0.00
e_2	0.00	0.00	-1.00	0.00
e_3	0.00	0.00	0.00	1.00
e_4	1.00	0.00	0.00	0.00

Table 2.1: Final eigenvectors of the two qubits example.

Hamiltonian, which is the only one that survives the annealing process, are always the vectors of the computational basis (see Eq. (1.6)), whitout superposition. The ground state is then $gs = (0, 0, 0, 1)$, meaning the final state of our system using the computational basis is $e_4 = |11\rangle$: therefore, in the Ising spin basis (see Eq. (2.8)) we can say that the final state of our system is $|\downarrow\downarrow\rangle$.

2.2 Problem formulation

How can we programme the biases and the couplers of the quantum annealer in order to resolve our problem? To do so, it is necessary to express the problem in a form that enables solution by minimization: this is achieved by writing the objective function of our system, which is a mathematical expression of the energy of a system. There are different possibilities to express our problem as an objective function; typically one of the *quadratic models* provided by Ocean software (see Sec. 2.3) is used:

- **Binary Quadratic Models** are unconstrained and have binary variables; typically used for applications that optimize over decisions that could either be true (or yes) or false (no).
- **Constrained Quadratic Models** can be constrained and have integer and binary variables; typically used for applications that optimize problems that might include real, integer and/or binary variables and one or more constraints.
- **Discrete Quadratic Models** are unconstrained and have discrete variables; typically used for applications that optimize over several distinct options; for example, which shift should employee X work, or should the state on a map be colored red, blue, green or yellow?

For a QPU, two formulations for objective functions are the Ising Model and QUBO (that stands for Quadratic Unconstrained Binary Model). Both these formulations are binary quadratic models and conversion between them is trivial.

2.2.1 Ising and QUBO formulation

The **Ising model** is traditionally used in statistical mechanics. Variables are "spin up" ($|\uparrow\rangle$) and "spin down" ($|\downarrow\rangle$), states that correspond to $+1$ and -1 values. It is important to underline that in the spin basis we have:

$$|\uparrow\rangle = \begin{pmatrix}1\\0\end{pmatrix} \equiv |0\rangle \qquad |\downarrow\rangle = \begin{pmatrix}0\\1\end{pmatrix} \equiv |1\rangle \tag{2.8}$$

where $|0\rangle$ and $|1\rangle$ represent the vector of the computational basis for the qubits used by a quantum annealer.

The possible relationships between the spins, represented by couplings, are correlations or anti-correlations. The objective function expressed as an Ising model is as follows:

$$E_{ISING}(s_i, s_j) = \sum_{i=1}^{N} h_i s_i + \sum_{i=1}^{N}\sum_{j>i}^{N} J_{ij} s_i s_j \tag{2.9}$$

where $s_i = \pm 1$, the linear coefficients corresponding to qubit biases are h_i and the quadratic coefficients corresponding to coupling strengths are J_{ij}.

On the other hand, **QUBO problems** are traditionally used in computer science, with variables taking values 1 (TRUE) and 0 (FALSE). A QUBO problem is defined using Q, a $N \times N$ upper-triangular matrix of real weights, and $\vec{x}$, a vector of binary variables, as minimizing the function:

$$f(\vec{x}) = \sum_{i=1}^{N} Q_{ii} x_i + \sum_{i=1}^{N}\sum_{j>i}^{N} Q_{ij} x_i x_j \tag{2.10}$$

where $x_i = 0, 1$, the diagonal terms Q_{ii} are the linear coefficients and the nonzero off-diagonal terms Q_{ij} are the quadratic coefficients. In the scalar notation, the QUBO formulation becomes:

$$E_{QUBO}(q_i, q_j) = \sum_{i=1}^{N} a_i q_i + \sum_{i=1}^{N}\sum_{j>i}^{N} b_{ij} q_i q_j \tag{2.11}$$

where a_i corresponds to the diagonal terms Q_{ii}, b_{ij} corresponds to the nonzero off-diagonal terms Q_{ij} and $q_i = 0, 1$. It is important to underline that QUBO problems are not constrained, meaning that there are no constraints on the variables other than those expressed in Q.

The Ising and QUBO models are mathematically equivalent; a simple substitution allows to translate one into the other. This can be easily done by setting:

$$q_i = \frac{s_i + 1}{2} \qquad s_i = 2q_i - 1$$

Thus, it's easy to derive other relations between the parameters of the two formulations that allow us to write the Ising Hamiltonian using the QUBO parameters; this proves that the two formulation are completely equivalent and we can use either one of the two to encode a generic problem.

2.2.2 Constrained problem formulation

How can a real world problem be written in a binary quadratic model? In general a realistic problem can be described as an objective function with multiple constraints that simultaneously play a role: so we describe it as a *constrained problem*. In order to translate the problem in an Ising or QUBO formulation there are some steps that must be followed [5].

1. **Writing objective function and constraint**
   First of all, we have to describe what is an objective function and what is a constraint: in general, the objective function is a function which have minima that correspond to solutions, while a constraint is a rule that must be respected. If an answer does not respect the constraint, then it is not an acceptable answer and must be discarded. Thus, the first thing to do is to define which are the objective function and the constraints of the problem: after that, the objective function and the constraints will be mapped to the energy of our qubit system.
2. **Converting the problem into binary math expressions**
   At this point, it is necessary to define the problem using binary variables. This is the hardest part, because real world problems are generally much more complicated than binary problems. Nevertheless, it is useful to understand what kind of answers is required: in fact, the problem can often be simplified in order to model the objective function and the constrains as binary information.
3. **Transforming math expression into a BQM**
   After that, we want to write the mathematical expression we found in what we have called a binary quadratic model. Different types of expressions will require different strategies, since there is not a general rule to follow. Frequently, some operations are required to achieve this point, in particular:
   - **Maximization to Minimization**. If the solution of our problem is encoded as the maximum of the objective function, we can convert it to a minimum by multiplying the entire expression by -1.

- **Equality to Minimization**. Each equation describing the problem can be converted to a minimization expression by moving all arguments and constants to one side of the equality and squaring the side different from zero. This will result in an expression that is satisfied only at its smallest value which is zero and generally it is the most common system for complicate problems. We must notice that in this way squared terms are introduced but neither QUBO nor Ising models have squared binary variables; this problem resolves easily because QUBO variables are $q_i = 0, 1$, so $q_i^2 = q_i$, and Ising variables are $s_i = \pm 1$, so $s_i^2 = 1$.

4. **Writing the final BQM expression**
When all of the components are written as BQM expressions, we can combine them together to get the final BQM. In this step, each constraint is multiplied by a parameter λ_i: this weighting is done using Lagrange multiplier that are in general different for each component. The answer that minimizes the BQM will provide also the best value for these parameters, but what matters is to set a strong enough value for these λ_i, which assures that the constraints are respected. Thus, the final BQM is:

$$BQM = min(\text{objective function} + \sum_i (\lambda_i \ \text{constraint}_i))$$

2.3 D-Wave topology

In order to map our problem to the D-Wave QPU, we can use the Ocean SDK, which allows us to write a Python program and submit it directly to the quantum annealer; the software automates the mapping from the linear and quadratic coefficients of a quadratic model to qubit biases and coupling values set on the QPU [5]. Nevertheless, it is critical to understand the layout of the D-Wave QPU and how this mapping is done: in fact, it has implications for the problem-graph size and solution quality and is the main focus of this thesis. First of all, the D-Wave QPU is a lattice of interconnected qubits: while some qubits connect to others via couplers, the D-Wave QPU is not fully connected. This means that the layout is not a complete graph, so if we choose two qubits and want to connect them using a coupler for our problem it could be not possible because the two are not connected. The qubits of D-Wave annealing quantum computers interconnect in one of two topologies: *Pegasus* for Advantage

QPUs (the ones that are now used) and *Zephyr* for next-generation QPUs (currently under development). The first topology was simpler and named Chimera: we will use it in order to understand how the QPU works. Before analyzing these architectures, it is important to point out that some small number of qubits and couplers in a QPU may not meet the specifications to function as desired: these are therefore removed from the programmable fabric that users can access. Thus, the users can access only a smaller subset of the QPU's graph, called the *working graph*. The yield of the working graph is the percentage of working qubits that are present: calibrated commercial systems typically have more than 97% of fabricated qubits available in their working graphs [5].

2.3.1 Chimera topology

The Chimera topology is relatively simple, so it is helpful for understanding the more complex topologies of newer QPUs: in addition to shared concepts, the newer topologies are often described in terms derived from Chimera. In general qubits on D-Wave QPUs are represented by loops, "oriented" vertically or horizontally as shown in Fig. 2.7 for a Chimera graph: this graph shows three rows of 12 vertical qubits and three columns of 12 horizontal qubits for a total of 72 qubits, 36 vertical and 36 horizontal.

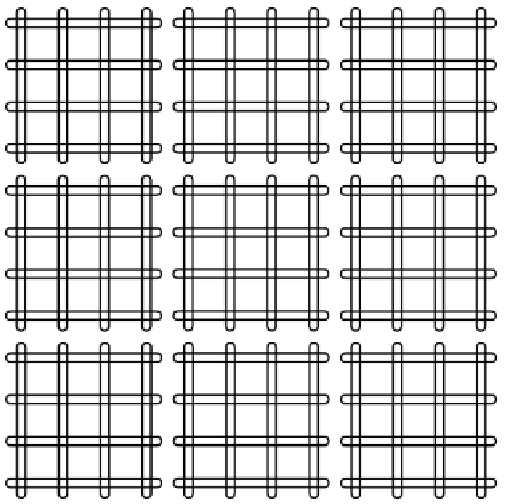


Figure 2.7: Chimera structure: qubits are represented by loops [5].

There are two different types of couplers supported by the Chimera topology:

- **Internal couplers:** these couplers connect pairs of orthogonal (with opposite orientation) qubits. In Fig. 2.8, green circles at the intersections of qubits are used to signify internal couplers; for example, the central horizontal qubit, highlighted in green, internally couples to four vertical qubits, highlighted in bold.
- **External couplers:** these couplers connect colinear pairs of qubits, so pairs of parallel qubits in the same row or column. In Fig. 2.8, external couplers are shown as connected blue circles and couple vertical qubits to adjacent vertical qubits and horizontal qubits to adjacent horizontal qubits; for example, the green horizontal qubit in the center couples to the two blue horizontal qubits in adjacent unit cells.

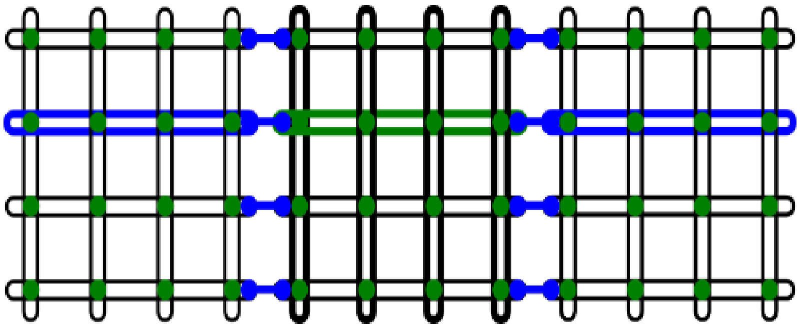


Figure 2.8: Internal and external couplers [5].

The Chimera topology has a clear recurring structure of four horizontal qubits coupled to four vertical qubits in a $\mathbf{K_{4,4}}$ bipartite graph, called a *unit cell* and which is typically rendered as either a cross or a column (see Fig. 2.9). In each of these

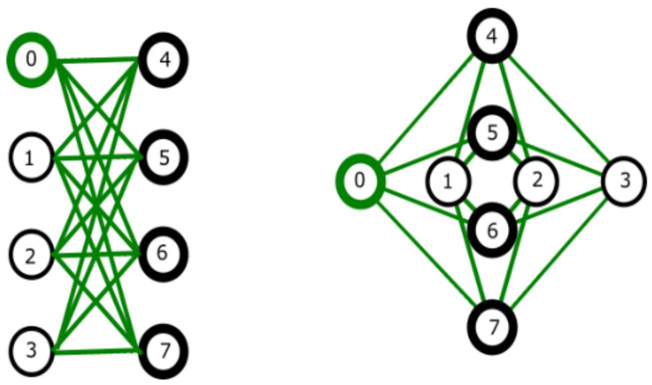


Figure 2.9: Chimera unit cell [5].

representation there are two sets of four qubits, where each qubit connects to all qubits in the other set but to none in its own: for example, the green qubit labeled 0 connects to qubits 4 to 7 in bold. Each unit cell is then connected to the others trough external couplers, as a lattice: the notation CN refers to a Chimera graph consisting of an $N \times N$ grid of unit cells (for example, the D-Wave 2000Q QPU supported a $C16$ Chimera graph: more than 2000 qubits were logically mapped into a 16×16 matrix of unit cells of 8 qubits). As we can see, this topology is not a complete-graph because qubits in the same set are not connected. Chimera qubits are characterized as having:

- Nominal length 4: this means each qubit is connected to 4 orthogonal qubits through internal couplers.
- Degree 6: this means each qubit is coupled to 6 different qubits (4 internal e 2 external couplers).

2.3.2 Pegasus topology

In Advantage QPUs, qubits are "oriented" vertically or horizontally, as in the Chimera topology, but similarly aligned qubits are also shifted, so that the clear 4×4 structure is not present anymore. Moreover, there is a new type of couplers, other than internal and external one, called odd couplers, which connect similarly aligned pairs of qubits: this type of connection wasn't possible on the Chimera topology. Fig. 2.10 shows the different type of couplers that a single qubit can have, taking the green central vertical qubit as an example:

- **Internal couplers:** connect pairs of orthogonal (with opposite orientation) qubits as shown in Figure 20. The green qubit is connected via internal coupling to 12 horizontal qubits, highlighted in bold .
- **External couplers:** connect vertical qubits to adjacent vertical qubits and horizontal qubits to adjacent horizontal qubits. The green qubit is coupled to the two adjacent vertical qubits, highlighted in blue.
- **Odd couplers:** connect similarly aligned pairs of qubits. The green vertical qubit is coupled to the red vertical qubit by an odd coupler.

Each translucent green square in Fig. 2.10 represents a Chimera unit cell structure; the Pegasus structure features qubits with native $\mathbf{K_4}$ and $\mathbf{K_{6,6}}$ subgraphs. Pegasus qubits are considered to have nominal length of 12 (each qubit is connected to 12 orthogonal qubits through internal couplers) and degree of 15 (each qubit is coupled

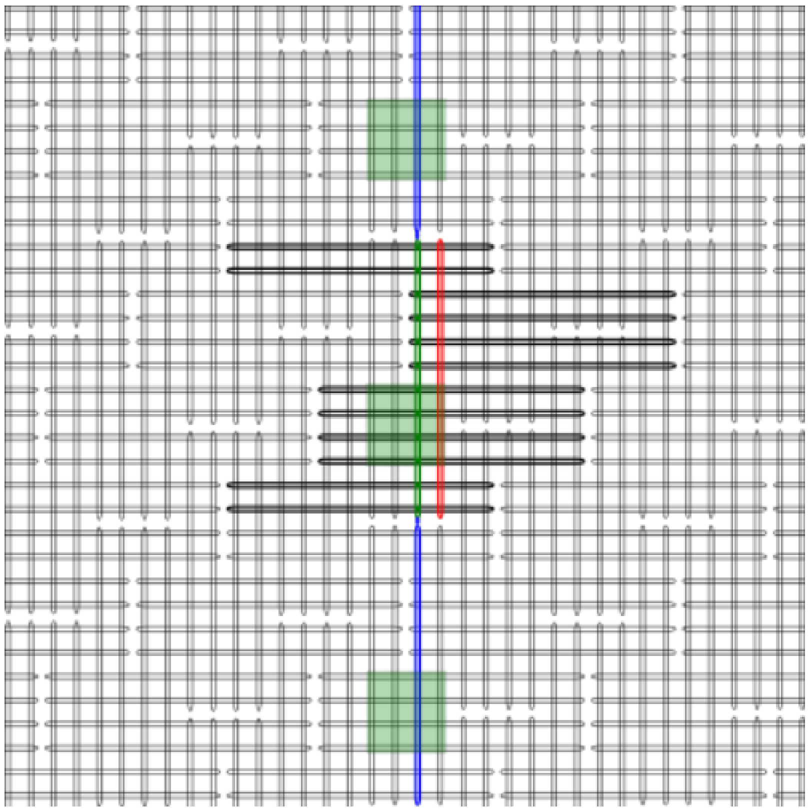


Figure 2.10: Pegasus structure and couplers [5].

to 15 different qubits). The general structure of the Pegasus topology is much more complicate than Chimera's one: as we can see in Fig. 2.11, the structure becomes a 3D lattice, where each unit cell contains twenty-four qubits. More formally, the Pegasus unit cell consists of 48 halves of qubits that are divided between adjacent unit cells.

2.3.3 Zephyr topology

D-Wave is currently developing its next-generation QPU with the Zephyr topology: qubits are "oriented" vertically or horizontally, as in Chimera and Pegasus, and are shifted and connected with three coupler types as in Pegasus, but this new graph achieves higher nominal length (16) and degree (20). A qubit in the Zephyr topology has sixteen internal couplers connecting it to orthogonal qubits and two external couplers and two odd couplers connecting it to similarly aligned qubits. Zephyr topology enables native $\mathbf{K_4}$ and $\mathbf{K_{8,8}}$ subgraphs.

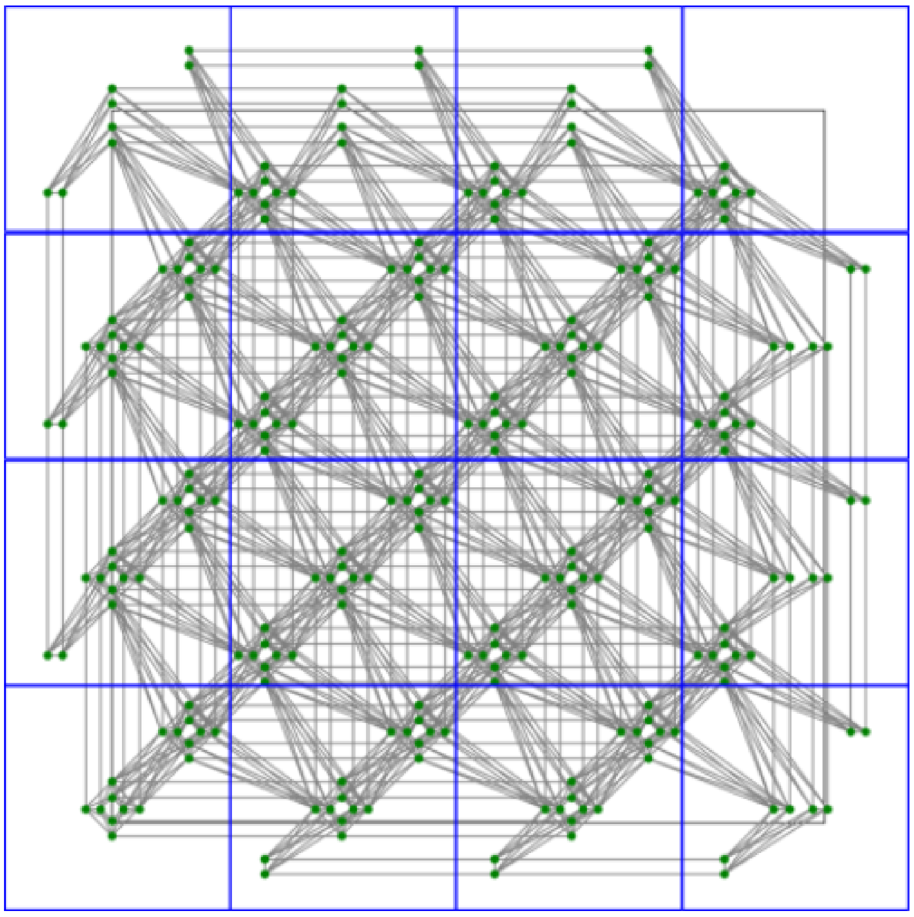


Figure 2.11: Pegasus unit cells in a $\mathbf{P_4}$ graph, with qubits represented as green dots and couplers as gray lines [5].

2.3.4 Minor embedding

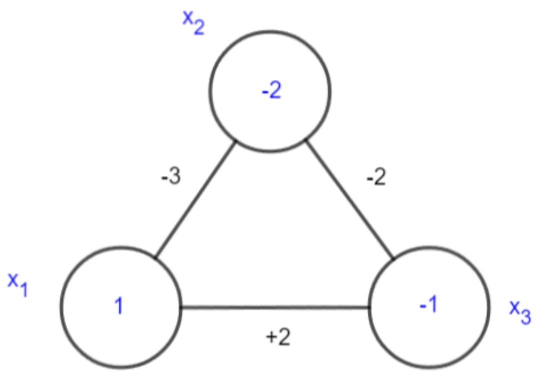
Each graph of a chosen topology is a collection of nodes and edges, which can be used to represent an objective function's variables and the connections between them, respectively. This process of mapping variables in the problem formulation to qubits on the QPU is known as *minor embedding*. In particular, one has:

- Nodes, that represent the objective function's variables (that can be either s_i ore q_i depending on the model used, Ising or QUBO) that are mapped to qubits on the QPU.
- Edges, that represent the objective function's quadratic coefficients (either $J_{i,j}$ or $b_{i,j}$) which are mapped to couplers.

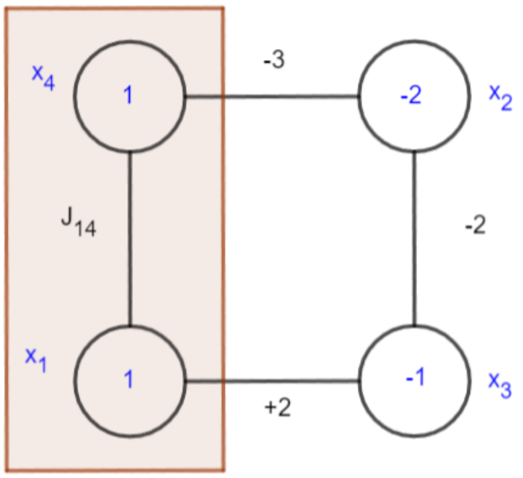
Embedding seems a straightforward process, but actually it is not that simple, because the QPU topologies used by D-Wave are not complete graphs. It is easier to comprehend this concept and why it is a complicate process with an example; suppose that we have to map the following objective function to a QPU that supports Chimera topology:

$$f(x_1, x_2, x_3) = x_1 + 2x_2 - x_3 - 3x_1x_2 - 2x_2x_3 + 2x_1x_3$$

The ideal representation that we can have for this function is by connecting 3 qubits on a complete graph, as we can see in Fig. 2.12a. But in a Chimera graph this is not possible, because the graph is not complete, as it only allows groups of four qubits (see Fig. 2.9): in order to connect 3 logical qubits we need to use 4 physical qubits and in particular we have to chain two of them, as in Fig. 2.12b. Chaining qubits is accomplished by setting the strength of their connecting coupler negative enough to strongly correlate the states of the chained qubits, so we choose the value of J_{14} in order to entangle these two qubits; if at the end of most anneals these qubits are in the same classical state, representing the same binary value in the objective function, they are in effect acting as a single variable.



(a) Ideal graph.



(b) Embedded graph.

Figure 2.12: Comparison between the ideal and the real embedding graph.

This procedure gets more complex as the number of qubits is increased, more and more physical qubits are used to represent a smaller number of logical qubits: thus, the real and ideal behaviour of the QPU are different. Using more qubits causes the energy in general to increase and the system can be in a different state from the ground state at the end of the anneal if the embedding procedure gets too complicated. In the next chapter we will focus on this process and study how it affects the energy spectrum.

Chapter 3

Minor Embedding

In this chapter we discuss the minor embedding process, which allows to map a BQM formulated problem to the quantum annealer QPU and which constitutes the main focus of this thesis. First of all, we take into consideration how much the choice of the parameters in the formulation of the problem can alter the solution. After that, we use a NP-hard problem, called knapsack problem, to study the embedding procedure and how this process modifies the energy spectrum during the anneal. In particular, we study the results obtained when the embedding employs more physical qubits than the logical qubits that form the problem, in order to enlighten whether this process changes the solution: we also study the energy distribution obtained by a repeated annealing procedure. Finally, we compare the QPU access time and the CPU execution time for an increasing number of objects of a knapsack-problem.

3.1 Ising model

In section 2.1 we focused on the D-Wave qubits and their mathematical description, which is carried on using the Ising Hamiltonian (see Eq. (2.2)). We use this specific Hamiltonian because the Ising model is a theoretical model which describes the behaviour of interacting spins in a magnetic field. If we consider spin as vectors of fixed length in space, the simplest form of interaction is a pairwise one, so that for two interacting vector spins $\vec{s_1}$ and $\vec{s_2}$, the energy of the configuration is given by [6]:

$$E = -J\vec{s_1} \cdot \vec{s_2} = -J|\vec{s_1}||\vec{s_2}|\cos\theta \tag{3.1}$$

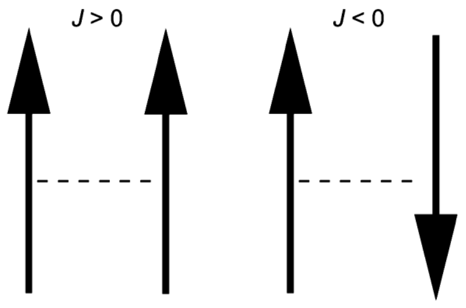


Figure 3.1: Spin configuration depending on the sign of J [6].

where θ is the angle between the spins and J is the exchange energy (we called it *coupling strength*). The configuration that the pair of spins choose will depend on the sign of J: if $J > 0$, the lowest energy is obtained for $\cos\theta = 1$, in which case the spins line up parallel to each other. If $J < 0$, the lowest-energy configuration has $\cos\theta = -1$, and so the two spins align antiparallel to each other (see Fig. 3.1).

It is possible to generalize from the two-spin example to many spins sitting on a lattice, with the spin on lattice site i denoted by s_i, which is the case of the quantum annealer QPU. Thus, the interaction energy for each couple of qubit is given by Eq. (3.1) and the total interaction energy can be cast as:

$$E = -\sum_{i>j} J_{ij} s_i s_j \tag{3.2}$$

where the spins s_i and s_j can only take on the values ± 1 and are known as *Ising spins*. The sign of J_{ij} again determines the type of ordering that results from the expression for the energy; if $J_{ij} > 0 \ \forall\ \{i, j\}$, then the lowest-energy configuration corresponds to all spins pointing in the same direction, which is what happens in a *ferromagnet* (note that if all spins point up this has the same energy as when all spins point down). Alternatively, if $J_{ij} < 0 \ \forall\ \{i, j\}$, then the lowest-energy configuration is *antiferromagnetic*, consisting of spins that are antiparallel on neighbouring sites [6].

3.1.1 Ising coefficients

The quantum annealer QPU we consider is a lattice of N sites, where at each site i there is a spin s_i that can take one of the values ± 1. The Hamiltonian for a given configuration $\{s_i\}$ is Eq. (2.2), where the parameters h_i represent the external magnetic field experienced by the spins, which we are able to set in order to describe different problems. In particular, if we turn the interaction between qubits off, a positive value of h_i will make the qubit end in the state $|1\rangle = |\uparrow\rangle$, while a negative value in the state $|0\rangle = |\downarrow\rangle$. As we can see, the initial Hamiltonian represents a lattice of non interacting spins, whether the problem Hamiltonian is a proper Ising Hamiltonian that describes how the spins interact. Moreover, it is important to notice that there is not a minus sign in front of interacting energy term; this means that when $J_{ij} < 0$ the spins s_i and s_j will be correlated, while when when $J_{ij} > 0$ they will be anti-correlated.

As presented in Sec. 2.2, problems are formulated as BQM and then the values of the coefficients h_i and J_{ij} can be passed to the quantum annealer. Those values are actually processed before they are communicated to the QPU, because each QPU has an allowed range of values for the biases and strengths of qubits and couplers. This is due to the fact that both biases and coupling strengths are communicated to the qubits through a magnetic field: if the magnetic field is too strong, i. e. h_i and J_{ij} are big numbers, it becomes really difficult to hit the single qubit and prepare it for the annealing process. Thus, the values we define in our problem are adjusted to fit the entire available range, by dividing them by a positive (non-zero) factor defined as [5]:

$$\begin{gathered}\max\{\max\left(\frac{\max(h)}{\max(h_{range})}, 0\right), \max\left(\frac{\min(h)}{\min(h_{range})}, 0\right), \\ \max\left(\frac{\max(J)}{\max(J_{range})}, 0\right), \max\left(\frac{\min(J)}{\min(J_{range})}, 0\right)\}\end{gathered} \tag{3.3}$$

which simply means that the highest absolute value between every h_i and J_{ij} is set to be equal to the limit of the accessible range, and the other values are rescaled. For an Advantage_system6.2 QPU the ranges for h and J are:

$$h \in [-1; +1] \qquad\qquad J \in [-2; +2]$$

As we can see, the ranges are really small: this means that when there is one value of the coefficients is much larger than the others, the scaling procedure reduces those values near to zero. This must not be confused with the request that the coefficients must necessarily be small, which is not true, because the scaling is automatically made by a classical computer before the problem is runned on the QPU; the important aspect is that the coefficients should have the same magnitude.

3.1.2 Solution spectrum

How does the solution change for different values of J_{ij} and h_i? For the sake of simplicity, I started from the simplest case of only two qubits, already addressed before in Sec. 2.1.5. The problem Hamiltonian of the system in this case is:

$$H = h_1\, \hat{\sigma}_z^{(1)} \otimes I^{(2)} + h_2\, I^{(1)} \otimes \hat{\sigma}_z^{(2)} + J_{12}\, \hat{\sigma}_z^{(1)} \otimes \hat{\sigma}_z^{(2)} \tag{3.4}$$

In Fig. 3.2, we see the solution of the annealing process for a fixed value of $J = 2.5$, which I simulated with Python, using the values of an Advantage system6.2 annealing schedule for $A(s)$ and $B(s)$. There are four different zones, each of them corresponding to a possible solution: the color-solution correspondence can be found in the legend next to the panel, where $\{e_i\}$ are the vectors of the computational basis (see Eq. (1.6)). As we can see, when $h_1, h_2 > J$ the solution is $e_4 = |\downarrow\downarrow\rangle$: this is the case where both the qubits have a bias stronger than the coupling strength, so J is not strong enough to anti-correlate the two qubits. The same happens when $h_1, h_2 < -J$, when we obtain

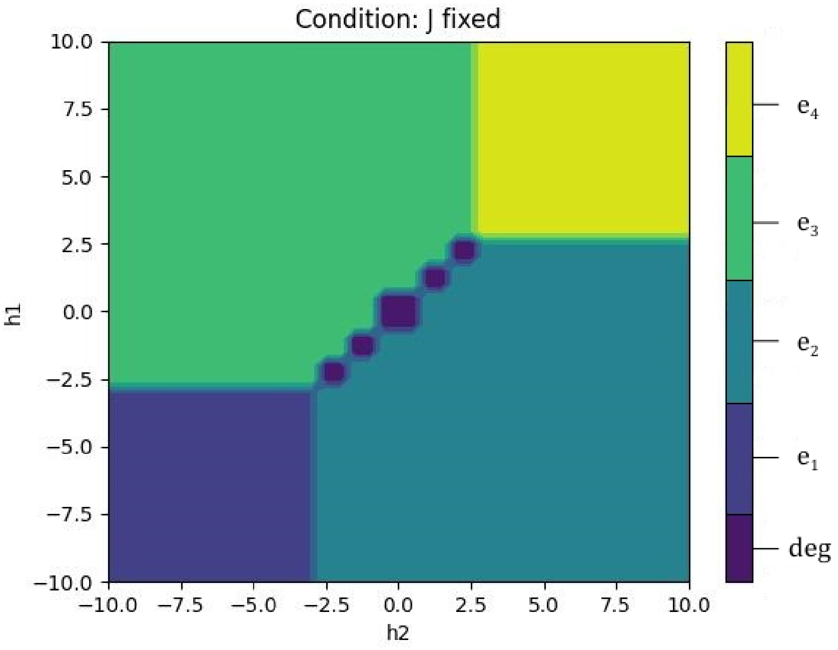


Figure 3.2: Results of the annealing process for two qubits when $J_{12} = 2.5$ is fixed. The legend explains the color-solution correspondence, where $\{e_i\}$ are the vectors of the computational basis (Eq. 1.6), while *deg* indicates a degenerate solution.

$e_1 = |\uparrow\uparrow\rangle$ as final state. After that, we have two symmetrical zones, divided by the $h_1 = h_2$ axis; when $h_1 > h_2$, within the limits imposed by the previous cases, the two qubits end in the state $e_3 = |\downarrow\uparrow\rangle$ (green area), because J is strong enough to anti-correlate the qubits: the second qubit ends in $|\uparrow\rangle$ because its bias h_2 is too strong ($h_2 > -J$) to be flipped by J. Similarly, when $h_2 > h_1$ the two qubits ends in the state $e_2 = |\uparrow\downarrow\rangle$ (light blue area). As we can see, there is also another color in Fig. 3.2, which is placed on the $h_1 = h_2$ line. The solutions for values in this area are degenerate (for example $\frac{1}{\sqrt{2}}(e_2 + e_3) = \frac{1}{\sqrt{2}}(|\downarrow\uparrow\rangle + |\uparrow\downarrow\rangle)$). We have this degeneracy because neither h_1 nor h_2 is strong enough to determine the final state, and multiple final states have the same final energy, which means both of the solutions are correct. This type of solution usually means that the problem has more than one optimal solution or it can also mean that it wasn't formulated correctly. Moreover, when all the coefficients tend to 0, the whole problem loses meaning and we obtain a near zero Hamiltonian, which is not physically acceptable.

We can also study what happens when J is not fixed: in order to do so, we set $h_1 = h_2 = h$ and plot the solutions of the annealing process (see Fig. 3.3). This time

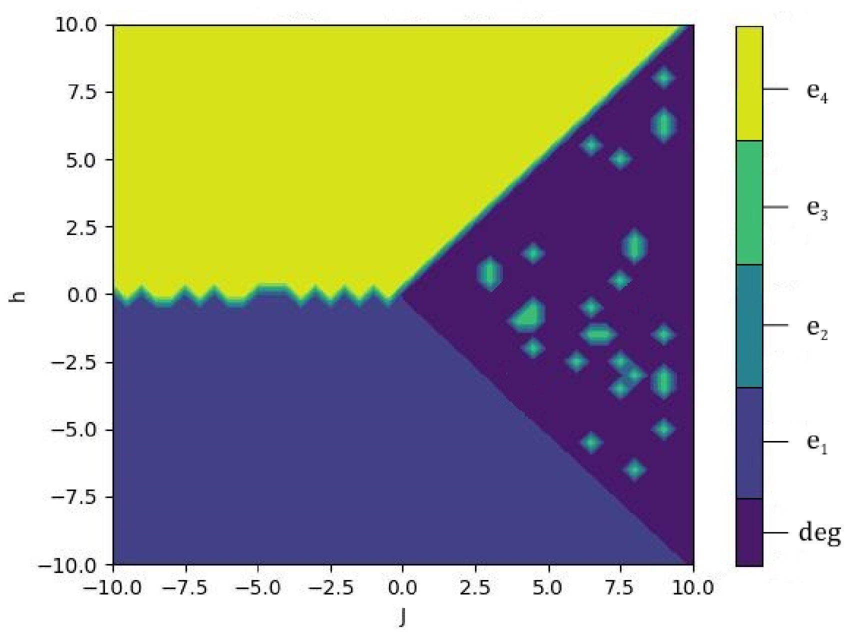
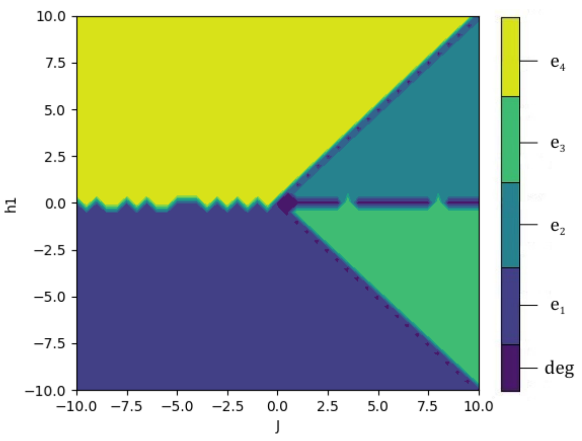


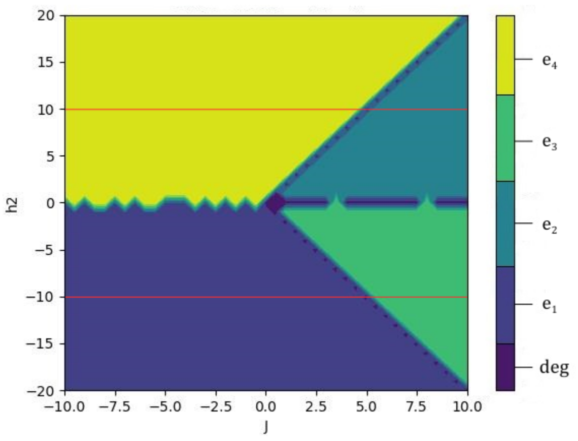
Figure 3.3: Results of the annealing process for two qubits as functions of h and J.

the graph is roughly divided in three areas: if $h > 0$ and $h > J$ the solution is $|\downarrow\downarrow\rangle$, while if $h < 0$ and $h < -J$ the solution is $|\uparrow\uparrow\rangle$. These two areas correspond to those in which the system behaves as a ferromagnetic system, where all the spins are aligned. The transition curve between these regions (which corresponds to $h \approx 0$) has a zigzag pattern: this emerges because, when the value of $h \to 0$, the computer approximates the value 0 with a number, which is very small but different from zero itself. This number can be either positive or negative, and so the system can end both in e_1 and e_2 for different values of J. When $h < |J|$, the coupling strength is strong enough to anti-correlate the two qubits; this area contains degenerate solutions, as already seen in Fig. 3.2. The reason is that we always have $h_1 = h_2$, which means that system is symmetrical: $|\uparrow\downarrow\rangle$ and $|\downarrow\uparrow\rangle$ are both optimal solutions. As we can see there are some spots in this region as well: they represent spurious solutions, which correspond to the system ending in a triply degenerate state.

We can remove this degeneration by setting for example $h_2 = 2h_1$; if we plot again the solution spectra (once for h_1 and once for h_2), we can notice that the degeneration remains only when h_1 or h_2 are zero and on the lines $J = |h_1|$ and $J = |h_2/2|$, as it stands out in Fig. 3.4. The two graphs are really similar, but we must pay attention because they are not to scale: in Fig. 3.4a we have $h_1 \in [-10; +10]$, while in Fig. 3.4b $h_2 \in [-20; +20]$. The new zones that are now separated in the solution spectra correspond to the two final states that were degenerate in the previous case: in

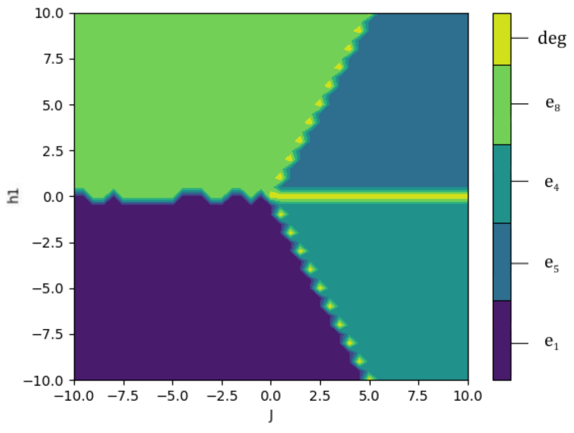


(a) Results as functions of h_1 and J.

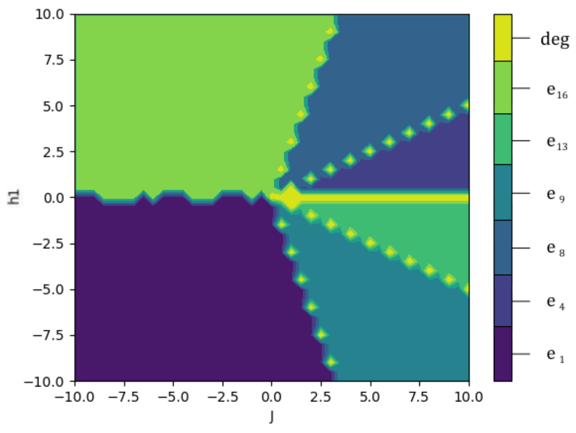


(b) Results as functions of h_2 and J.

Figure 3.4: Solution spectra for a two qubits problem when $h_2 = 2h_1$.



(a) Results with 3 qubits.



(b) Results with 4 qubits.

Figure 3.5: Solution spectra for example problems with 3 and 4 qubits. The legend explains the color-solution correspondence, where $\{e_i\}$ are the vectors of the respective computational basis, while *deg* indicates a degenerate solution.

particular, when $h_1, h_2 > 0$ and $J > h_1 = h_2/2$ the result is $|\uparrow\downarrow\rangle$, while when $h_1, h_2 < 0$ and $J > -h_1 = -h_2/2$ the result is $|\downarrow\uparrow\rangle$. These areas represent the anti-ferromagnetic behaviour of the system.

When the number of qubits increases, the solution spectrum presents only minimal changes: in Fig. 3.5 we can see some example of what happens, where I considered three and four qubits employed. As in the previous cases, we have two areas which represent the ferromagnetic behaviour of the system: we can see that in the example with three qubits (see Fig. 3.5a) these zones are a little bigger than those in the example with four qubits (see Fig. 3.5b). This is a direct consequence of the increasing number of qubits employed: there are many more possible states and therefore a smaller difference in J can make the system end in a different state. In the remaining zones we can find the anti-ferromagnetic behaviour of the system, while the degenerate solutions are only on the dividing lines.

3.2 Knapsack problem

As we already said, quantum annealing is a strong tool for the resolution of optimization problems, because a quantum annealing process returns the low-energy states of a system and therefore the optimal or near-optimal combination of elements. One of the most famous and studied combinatorial optimization problem is the so called 'knapsack problem' (from now on we will address it as KP), which is a NP-hard problem. We will use this problem to study the embedding procedure that is performed by the quantum annealer, because with some accurate conditions we can predict the correct solution of the problem and compare it with the result provided by the annealer.

3.2.1 Binary KP

Suppose a hitch-hiker has to fill up his knapsack by selecting from among various possible objects those which will give him maximum comfort [7]. The knapsack problem can be mathematically formulated by numbering the objects from 1 to n and introducing a vector of binary variables X_j with $j = 1, \ldots, n$ having the following meaning:

$$x_j = \begin{cases} 1, & \text{if object } j \text{ is selected} \\ 0, & \text{otherwise} \end{cases} \tag{3.5}$$

We can now write the objective function for the KP: we call p_j the measure of the comfort given by object j, so the function we want to maximize is given by:

$$\sum_{j=1}^{n} p_j x_j \tag{3.6}$$

We need to remember that we also have W, which is the maximum weight of the knapsack; our problem will be to select, from among all binary vectors x satisfying the constraint (where w_j are the weights of the objects):

$$\sum_{j=1}^{n} w_j x_j \leq W$$

the one that maximizes the objective function (3.6) [7]. We will use this simple optimization problem to study how the quantum annealer faces this problem and finds the correct solutions; in particular we want to understand how the embedding process modifies the system and its energy and how to choose the parameters h_i and J_{ij} of the Ising Hamiltonian (2.2) in order to obtain the correct solution.

3.2.2 Two qubits KP

The first scenario we will study is really simple: we will consider a KP with only one object x_j that we can choose to fill our knapsack with. The total weight is $W = 1$ and the weight of the object is $w_j = 1$; we will keep the value of the object as a variable v in order to study the whole process without biases and obtain a general solution. In order to solve the KP with quantum annealing, we have to follow the procedure described in Sec. 2.2; we will first use the QUBO fomulation and then transform it to an Ising model.

1. **Writing objective function and constraint**

   Objective function $\rightarrow$ We want to maximize the total value of the chosen items: we have only one item so we want to maximize its value.

   Constraint $\rightarrow$ The total weight must be less or equal to $W = 1$.

2. **Converting o.f. and constraint into binary math expressions**

   In order to write the objective function we have to associate to each binary variable the value of the qubit whether the object is chosen or not; if the object is chosen, the qubit will have the highest value, if not, the qubit will have the lowest value as we can see in Tab. 3.1.

	Object i is chosen	Object i is NOT chosen
QUBO	$q_i = 1$	$q_i = 0$
Ising	$s_i = +1$	$s_i = -1$

   Table 3.1: Value of the qubits states in the Ising and QUBO formulation.

   Now we can write the objective function and the constraint as a mathematical expression using the QUBO formulation:

   - Objective function: $min(-vx-0a) = min(-vx)$. We introduce an auxiliary binary variable (called 'slack-variable') with weight $w_a = 1$ and value $v_a = 0$.
   - Constraint: $w_x x + w_a a = W \rightarrow x + a = 1$. We use the slack-variable in order to have always the total weight equal to W.

3. **Transforming the math expression into a BQM**

   In order to have a BQM we need to change a little the mathematical expression of the constraint; we need to move each term to the same part of the = sign and square the expression, obtaining: $(x - a - 1)^2$.

4. **Writing the final BQM expression**
   Now that we have both the objective function and the constraint written as a BQM we can use the Lagrange multipliers to combine them together and obtain the final BQM for our problem:
$$Q(x,a) = min(-vx + \lambda(x+a-1)^2) = min((-v-\lambda)x - \lambda a + 2\lambda ax)$$
$$\rightarrow \quad Q(x,a) = \begin{pmatrix} x & a \end{pmatrix} \begin{pmatrix} -v-\lambda & \lambda \\ \lambda & -\lambda \end{pmatrix} \begin{pmatrix} x \\ a \end{pmatrix}$$
5. **Transforming the QUBO expression to an Ising model**
   Now we want to transform the QUBO expression to an Ising one, because we want to study the final state obtained from the Ising Hamiltonian. We operate a change of variables, as follows:
$$x = \frac{s_1+1}{2} \qquad a = \frac{s_2+1}{2}$$
   The Ising BQM becomes:
$$Q(s_1,s_2) = -\frac{v}{2}s_1 + \frac{\lambda}{2}s_1s_2$$

Once we have the Ising formulation of our BQM we can compare it to the final Ising Hamiltonian for two qubits (Eq. (2.6)): notably, we have a perfect correspondence between the two expression if we have $h_1 = -\frac{v}{2}$, $h_2 = 0$ and $J_{12} = \frac{\lambda}{2}$. In this case the final Hamiltonian becomes:
$$H_f = \frac{1}{2}\begin{pmatrix} -v+\lambda & 0 & 0 & 0 \\ 0 & -v-\lambda & 0 & 0 \\ 0 & 0 & +v-\lambda & 0 \\ 0 & 0 & 0 & +v+\lambda \end{pmatrix}$$
Obviously the eigenstates of this Hamiltonian are represented by the 4 vectors:
$$\begin{pmatrix}1\\0\\0\\0\end{pmatrix} = |00\rangle \qquad \begin{pmatrix}0\\1\\0\\0\end{pmatrix} = |01\rangle \qquad \begin{pmatrix}0\\0\\1\\0\end{pmatrix} = |10\rangle \qquad \begin{pmatrix}0\\0\\0\\1\end{pmatrix} = |11\rangle$$
with eigenvalues respectively $-v+\lambda$, $-v-\lambda$, $+v-\lambda$ and $+v+\lambda$. Now, how can we find the final state of our system? The final state of our system will be the ground state

of this Hamiltonian, which is the eigenstate with the the lowest associated eigenvalue. Moreover, we need to pay attention to the fact that we want to express our final solution using the spin base:

$$|\uparrow\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad\qquad |\downarrow\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

The eigenvalues are function of v and λ, and we can find 4 different scenarios depending on the values of these two parameters and the solution we expect to find for our KP:

- $\boldsymbol{\lambda > 0, v > 0}$
  In this case, we have both the value of the object and the parameter λ positive: it is convenient for the hiker to chose the object s_1 and leave the slack-variable s_2, so the solution we expect is the state $|\uparrow\downarrow\rangle$. The lowest eigenvalue is $-v - \lambda$, which corresponds to the eigenstate $|01\rangle$; if we write it in the spin base we have $|\uparrow\downarrow\rangle$ as final state, so the solution is the one expected.
- $\boldsymbol{\lambda > 0, v < 0}$
  In this case, the value of the object is negative and the parameter λ is positive: therefore, it is convenient for the hiker to leave the object s_1 and chose the slack-variable s_2 (because in this case the objective function is minimized and the total weight is always $W = 1$), so the solution we expect is the state $|\downarrow\uparrow\rangle$. The lowest eigenvalue is $+v - \lambda$, which corresponds to the eigenstate $|10\rangle$ and in the spin base we have $|\uparrow\downarrow\rangle$ as final state so the solution is the one expected.
- $\boldsymbol{\lambda < 0, v > 0}$
  In this case, we have the value of the object positive and the parameter λ negative: it is convenient for the hiker to chose the object s_1 and also the slack-variable s_2 (because when $\lambda < 0$ we are inverting the constraint, so $W = -1$), so the solution we expect is the state $|\uparrow\uparrow\rangle$. The lowest eigenvalue is $-v + \lambda$, which corresponds to the eigenstate $|00\rangle$ and in the spin base we have $|\uparrow\uparrow\rangle$ as final state so the solution is the one expected.
- $\boldsymbol{\lambda < 0, v < 0}$
  In this case, values of both the object and the parameter λ are negative: it is convenient for the hiker to leave both of them, so the solution we expect is the state $|\downarrow\downarrow\rangle$. In this case the lowest eigenvalue is $+v + \lambda$, which corresponds to the eigenstate $|11\rangle$ and in the spin base we have $|\downarrow\downarrow\rangle$ as final state so the solution is the one expected.

The canonical solution of a knapsack problem is the one where $\lambda > 0$ and $v > 0$; in the next examples we will always consider this condition satisfied, so that we can predict the correct solution. Moreover, as will be further explained in the next paragraphs, the value of λ must be higher than a particular value (which differs for each problem) in order for the constraint to be respected.

3.2.3 Three qubits KP

The situation described in the last paragraph can be complicated by increasing the number of qubits used; what happens if we have for example 2 objects, x_1 and x_2? Suppose that we want to resolve a knapsack problem in which the values of the objects are $v_1 = 1$ and $v_2 = 2$ and their weights are $w_1 = w_2 = 1$: the total weight is still $W = 1$, thus the number of qubits we must use to solve our problem is three (the third one is the slack variable we introduced before, which has $w = 1$ and $v = 1$. This problem, similarly to the previous one, is really simple and the solution we expect is $|{\downarrow\uparrow\downarrow}\rangle$, because the most valuable object that respect the constraint is x_2.

It can easily be proved that in general the coefficients for a n-object knapsack problem where the total weight is fixed to $W = 1$ can be simply computed as:

$$h_i = \frac{(n-1)\lambda - v_i}{2} \qquad\qquad J_{ij} = \frac{\lambda}{2} \tag{3.7}$$

The problem formulation is carried out in the same way as in Sec. 2.1.5 and using Eq. (3.7), the final Ising BQM we obtain is:

$$Q(s_1, s_2, s_3) = \left(\frac{\lambda}{2} - \frac{1}{2}\right) s_1 + \left(\frac{\lambda}{2} - 1\right) s_2 + \frac{\lambda}{2} s_3 + \frac{\lambda}{2} s_1 s_2 + \frac{\lambda}{2} s_1 s_3 + \frac{\lambda}{2} s_2 s_3$$

As we can see, the Ising BQM contains all the mixed terms $s_i s_j$: this means that the graphical representation of the problem is a complete graph. This will be the case also in the next examples, which will trigger the chain formation during the embedding process because the topology of the QPU involved is not a complete graph. Once we have the Ising BQM, we can submit our problem to a quantum annealer: in particular we will use the Advantage_system6.2, located in North America, that is configured with the Pegasus topology. Using the Ocean SDK, I submitted the problem with a value of $\lambda = 3$, which is big enough for the constraint to be respected, so the coefficient of the Ising BQM are:

$$\vec{h} = \left(1, \frac{1}{2}, \frac{3}{2}\right) \qquad J = \frac{3}{2}\begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

The results obtained by the quantum annealer are organized in Tab. 3.2: as we can see there is a total of 5000 samples, which is the maximum for an Advantage_system6.2 QPU. In the majority of the runs, the final state of the system is $|{\downarrow\uparrow\downarrow}\rangle$, which is the expected solution, in particular we have a probability of:

$$P(|{\downarrow\uparrow\downarrow}\rangle) = \frac{\text{num\_oc.}}{\text{samples}} = \frac{4471}{5000} = 89.42\%$$

The correct solution is also the one with minimum energy: this means that the annealing procedure worked well and the system stayed in the ground state. The energy obtained is expressed in GHz, meaning that the energy of the system in $Joule$ is:

$$E = \text{energy} * h * B(1) = -1.7547 \cdot 10^{-32}\ J \tag{3.8}$$

where h is the Planck constant and $B(1)$ is the value of the parameters $B(s)$, which multiplies the problem Hamiltonian in Eq. (2.2), at the end of the anneal ($s = 1$).

The other states that occurs as final state of the annealing process corresponds to excited state: sometimes the anneal fails and the system jumps from the ground state to one of these states. The second most frequent state (which has a probability of only 8.94%) corresponds to the first excited state and so on: the possible combination of 3 qubits are 8 in total, but some of them never occurs because they have much more

	s_1	s_2	s_3	energy	num_oc.
0	-1	+1	-1	-3.5	4471
1	+1	-1	-1	-2.5	447
2	-1	-1	+1	-1.5	65
3	+1	+1	-1	-1.5	16
4	-1	+1	+1	-0.5	1

Table 3.2: Advantage_system6.2 annealing results for a KP with two objects. The energy is expressed in units of GHz, which can be converted in J with Eq. (3.8).

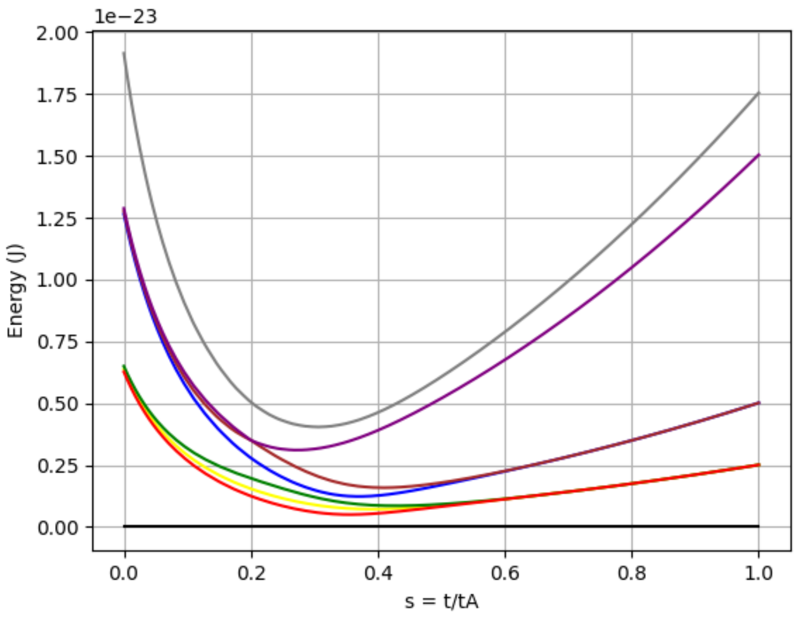


Figure 3.6: Eigenvalue evolution over time of the KP with two objects. The black line represent the ground state's eigenvalue, while the others the excited states' eigenvalues (the red one is the first excited state's eigenvalue).

energy than the ground state. The eigenvalue evolution over time is represented in Fig. 3.6: the first excited state (red line) defines the band gap $\Delta E = 4.98 \cdot 10^{-25}\ J$, which occurs when $s = 0.3553554$. Trough the `dwave.inspector` tool we can visualize the qubits involved in the annealing process directly on the QPU. For this particular

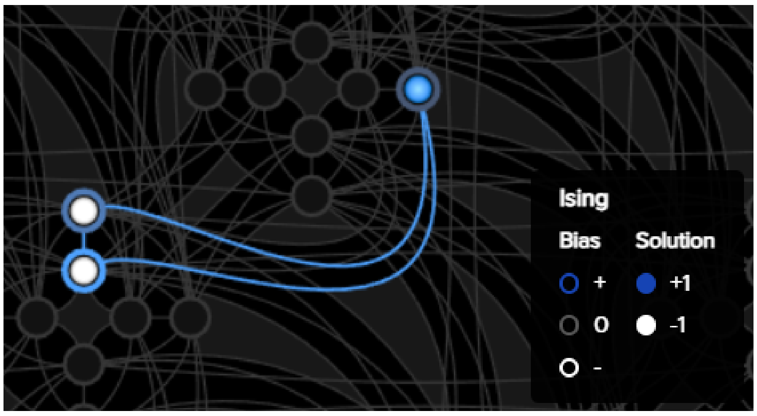


Figure 3.7: Embedding of the KP with 2 objects on the Advantage_system6.2 QPU.

problem, the qubits used for the 3 variables are: $s_2 = (1499), s_1 = (5587), s_3 = (5602)$. As we can see in Fig. 3.7, the Pegasus topology allows complete graph of 3 qubits, differently from the Chimera topology. Thus, no chains are needed, and only three physical qubits are used, corresponding to the number of the logical ones.

3.2.4 Four qubits KP

For the sake of completeness, we include also the description of a KP with 3 objects, x_1, x_2 and x_3, with values $v_1 = 1$, $v_2 = 2$, $v_3 = 3$ and equal weight $w_i = 1$. The total weight is $W = 1$, so we need to use a slack variable a, with $v = 0$ and $w = 1$. Using the formulas in Eq. (3.7) and setting $\lambda = 4$, we obtain:

$$\vec{h} = \left(\frac{7}{2}, 3, \frac{5}{2}, 4\right) \qquad J = 2\begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

This case is not really interesting because the Pegasus topology allows a complete graph of 4 qubits, so there are not many difference in the embedding process from the previous one (see Fig. 3.8).

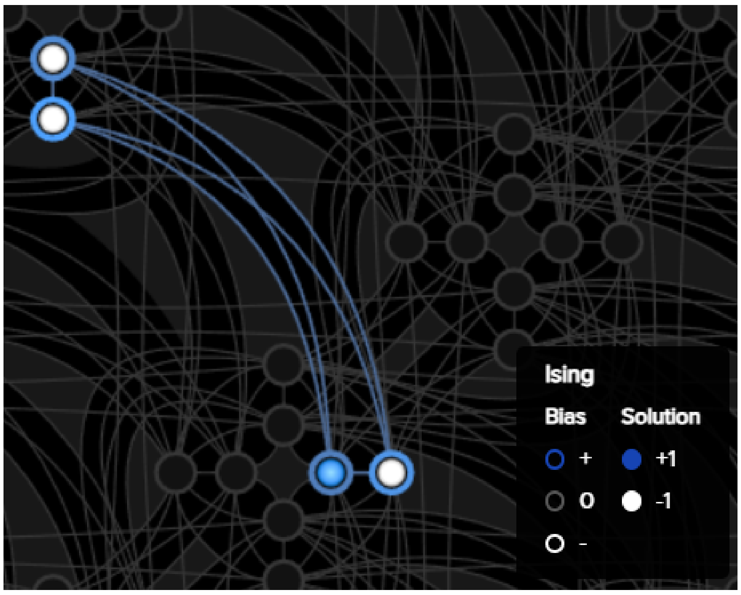


Figure 3.8: Embedding of the KP with 3 objects on the Advantage_system6.2 QPU.

	s_1	s_2	s_3	s_4	energy	num_oc.
0	-1	-1	+1	-1	-10.0	3742
1	-1	+1	-1	-1	-9.0	938
2	+1	-1	-1	-1	-8.0	214
3	-1	-1	-1	+1	-7.0	56
4	-1	+1	+1	-1	-7.0	38
5	+1	-1	+1	-1	-6.0	10
6	+1	+1	-1	-1	-5.0	1
7	-1	-1	+1	+1	-5.0	1

Table 3.3: Advantage_system6.2 annealing results for a KP with three objects. The energy is expressed in units of GHz, which can be converted in J with Eq. (3.8).

Tab. 3.3 contains the solutions from the quantum annealer; the most frequent final state is $|{\downarrow\downarrow\uparrow\downarrow}\rangle$, which is the expected solution (because x_3 is the most valuable object with a weight $w_3 \leq W$). As we can already see here, the probability of the exact solution has decreased and now it's $P = 74.48\%$; when many qubits are employed, the system struggles to stay in the ground state and the desired final state is reached in fewer runs.

3.3 Chain formation

The previous examples depicted how to formulate a knapsack problem and how to submit it to a quantum annealer. However, now we want to focus to what happens when the embedding procedure uses a different number of physical qubits from the logical qubits used in the problem formulation: this process cause the creation of *chains* of qubits that behave as a single variable.

3.3.1 Five qubits KP

The perfect example of this process is represented by a KP with four objects: we use five logical qubits (four variables and one slack-variable), but as shown later the embedding involves six physical qubits. Among those six qubits, two of them (i, j) are entangled: the entanglement is obtained by choosing a strong negative value for $J_{i,j}$ that makes the two qubits end in the same state.

Consider four object, x_1, x_2, x_3, x_4, with values $v_1 = 1$, $v_2 = 2$, $v_3 = 3, v_4 = 4$ and equal weight $w_i = 1$. The total weight is always $W = 1$, so we need to use only one slack variable a, with $v = 0$ and $w = 1$. The coefficients of the Ising BQM, setting $\lambda = 5$, become:

$$\vec{h} = \left(7, \frac{13}{2}, 6, \frac{11}{2}, \frac{15}{2}\right) \qquad J = \frac{5}{2}\begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

We can see in Fig. 3.9 the embedding on the QPU that is automatically made by the quantum annealer: the left pane shows the complete graph of 5 qubits, which is what our problem is logically equivalent to, while the right pane shows the physical embedding on the QPU. The variable s_5, which corresponds to a, is mapped to two different qubits, highlighted in the picture: $s_5 = (4793, 1630)$. These two qubits form a so called *chain* and behave as a single variable: each node has half of the initial bias (in this case $h_{4793} = h_{1630} = \frac{h_5}{2} = \frac{15}{4}$) and they are strongly connected with $J = -7.07$, which ensures they end up in the same state. This is the simplest case of what happens when a problem is submitted to a quantum annealer: the embedding consists in finding the physical qubits that can represent the logical qubits of the problem.

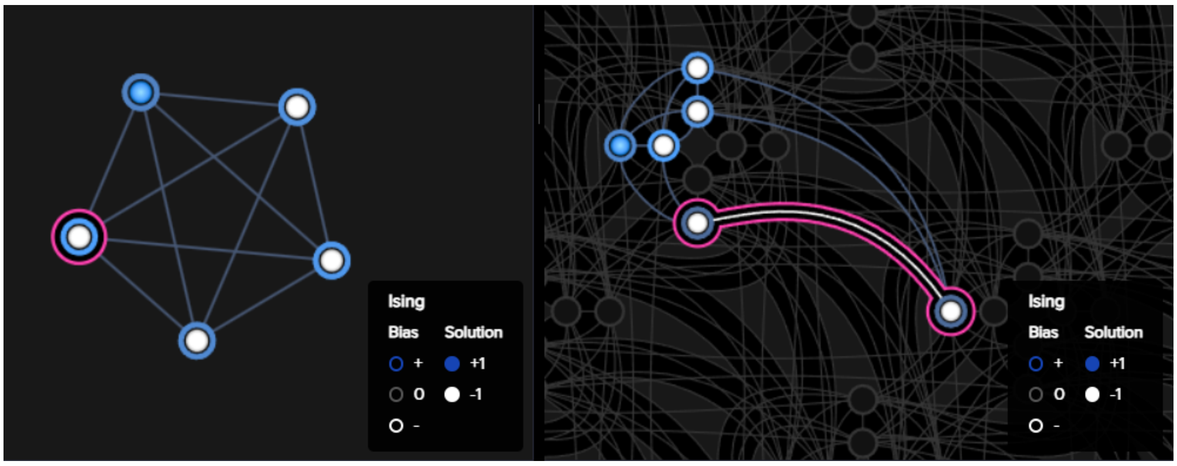


Figure 3.9: Chain formation during the embedding process.

What happens to the results? Is the optimal solution, which we already know to be $|\downarrow\downarrow\downarrow\uparrow\downarrow\rangle$, still the most frequent one? Tab. 3.4 contains part of the 5000 final states, those with the most occurrences. The first state, which occurred with $P = 29.14\%$, is the optimal solution we expect: the probability has decreased a lot and only roughly 1 anneal out of 3 gives the correct answer. The second state corresponds to the first excited state of the system: in this case $P = 25.96\%$, which is a little less than the previous one. As it clearly stands out, the embedding procedure this time modifies the system a lot, and the first excited state occurs nearly as frequently as the ground state, even though there is only one qubit difference between the physical and logical ones.

In order to study more in-depth why the chain formation changes that much the probability of the correct answer, it is useful to consider what differs between the energy spectrum of a 5-qubits complete graph and a 6-qubits graph with only the couplers used by the QPU. Therefore, I plotted the ground state and the first excited state of both graphs together, as in Fig. 3.10. The black line is the ground state, which in both cases is the same and corresponds to the optimal solution. The blue line represents the first excited state of the 5-qubits complete graph, while the red one is the first excited state of the 6-qubits embedded graph. The two lines overlap and this means that the first excited state is the same in both cases; as it stands out, the energy band gap changes a lot. In the 5-qubits complete graph, we have $\Delta E = 5.07 \cdot 10^{-25}\ J$ when $s = 0.3233233$, while in the 6-qubits embedded graph, we

	s_1	s_2	s_3	s_4	s_5	energy	num_oc.
0	-1	-1	-1	+1	-1	-16.5	1457
1	-1	-1	+1	-1	-1	-15.5	1298
2	-1	+1	-1	-1	-1	-14.5	404
3	-1	-1	+1	+1	-1	-14.5	421
4	+1	-1	-1	-1	-1	-13.5	435
5	-1	+1	-1	+1	-1	-13.5	117

Table 3.4: Advantage_system6.2 annealing results for a KP with four objects, with `num_reads` = 5000. The energy is expressed in units of GHz, which can be converted in J with Eq. (3.8).

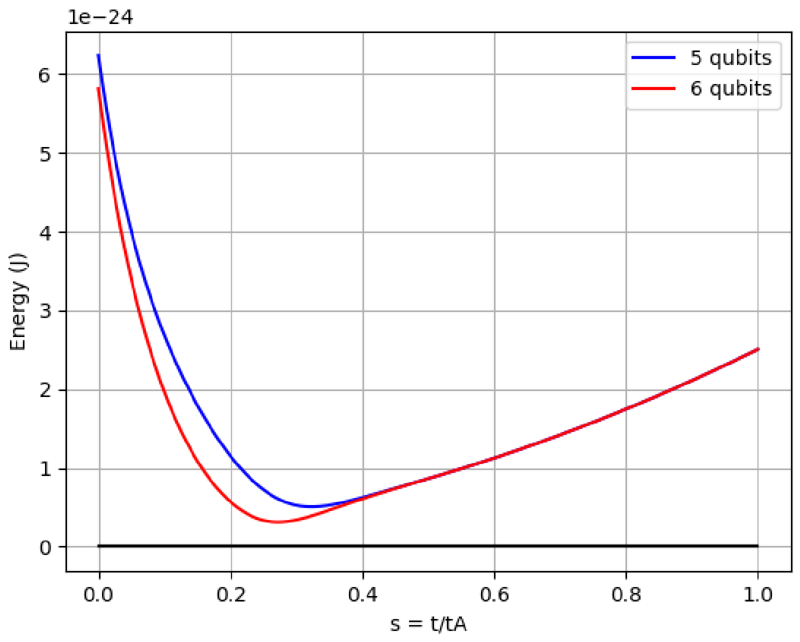


Figure 3.10: Energy spectrum comparison between the 5-qubits complete graph and the 6-qubits embedded graph.

have $\Delta E = 3.12 \cdot 10^{-25}\ J$ when $s = 0.2722723$. In the real embedding performed on the QPU the band gap is roughly $\frac{3}{5}$ of the band gap in the complete graph: this is the reason why the frequency of the correct solution decreases a lot in the 4-objects KP, because it is easier for the system to jump in the first excited state during the annealing procedure. This is also consistent with the fact that the probability of the first excited state is almost equal to that of the ground state. Moreover as we can see the minimum of the energy gap is reached a bit sooner in the real embedding and this is another aspect which contributes to a decrease in the probability. The theoretical aspect involved is obviously the quantum adiabatic theorem, and in particular the condition expressed in Eq. (2.1): a smaller energy gap and a more rapid change in the energy cause this condition to be not satisfied more often, so the annealing ends more frequently in a different state from the ground state.

If we submit our problem to the quantum annealer again we may obtain something different: the probabilities change and also the qubits involved in the embedding. Moreover, the chain can change and the two qubits employed can behave as another variable, with different bias. In any case, the ground state results to be the most frequent one, with a probability between 25% and 35%: this is important because running the problem a few times can help us identify the correct solution.

3.3.2 Chain strength

As previously described, the qubit-chains are formed when a direct coupling of all the involved qubits is not possible; the qubits that form a chain are coupled with a strong negative J, which causes them to end in the same state. Thus, the quantum annealer has to choose a value for the coupling strength strong enough to succeed in this purpose. Typically, chains are generated by minor-embedding tools such as Ocean's `minorminer`: nonetheless, the chain strength is a parameter that we are able to decide and which determines the values for the couplers used in forming these chains. Moreover, J values of chain couplers are scaled together with the given J values of the problem, following Eq. (3.3): this is the reason why the chain strength will never be set too large. As chain strengths get larger, the individual Ising coefficients, which were introduced to express the terms we want to minimize, shrink to near zero. Each

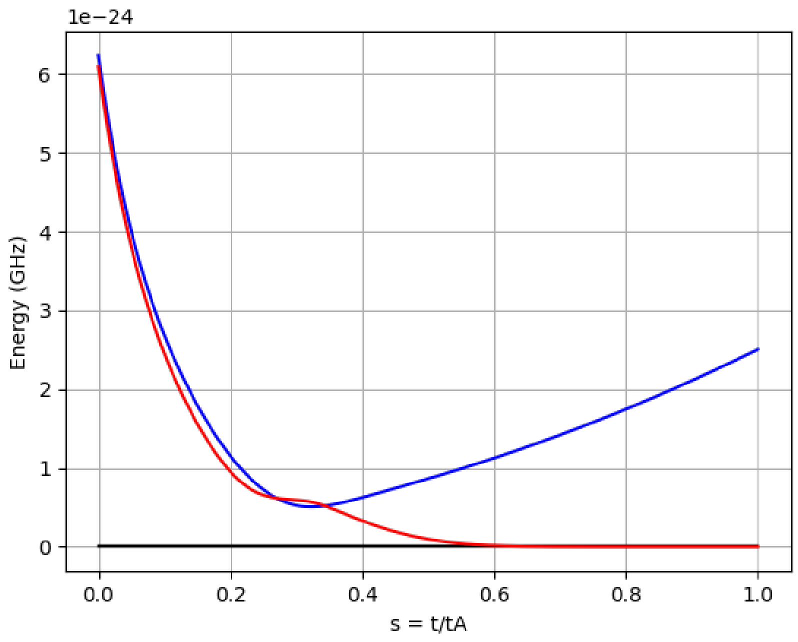


Figure 3.11: Energy spectrum for $J = -2$.

chain begins to act like a separate entity, and our problem evolves in a problem of N independent variables that do not interact with each other: such problem would have many optimal solutions, all with the same energy and it no longer represents the original problem [5]. If we consider the previous example of a 4-objects KP, the chain strength was set to `chain_strength` $= 7.07$ and consequently $J_{chain} = -7.07$: as we can see $|J_{chain}| >> J_{ij} = \frac{5}{2}$, which assures the coupling is effective.

What happens if this parameter is not strong enough? Well, if the chain strength is set not strong enough the band gap reduces to zero and the eigenvalues become degenerate (see Fig. 3.11). This violates the condition of the quantum adiabatic theorem (Eq. (2.1)), so the system jumps more easily in an excited state and the problem is not correctly solved. D-Wave's Problem Inspector helps us to visualize the problem: Fig. 3.12 shows that the the two qubits that form the chain end in two different states (one in $|\uparrow\rangle$ and one in $|\downarrow\rangle$), which means that the chain is broken and they do not represent the same variable.

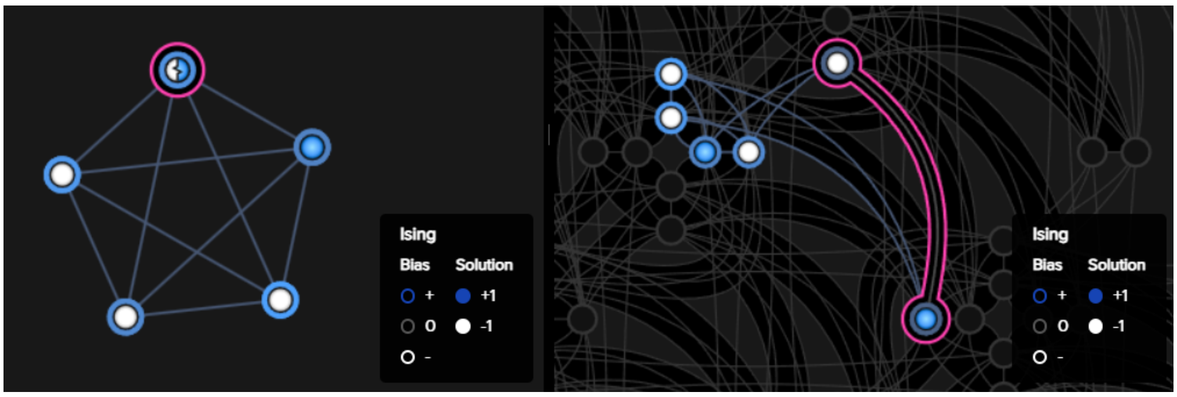


Figure 3.12: Chain break: the coupling strength is set too small.

The results of such case are reported in Tab. 3.5: the lowest energy state is still the correct solution of the problem, with $P = 46.52\%$. The last column of the table represent the chain breaking: when this value is not zero, it means that the final state of the system contains a broken chain, which implies that the annealing process didn't work. All those occurrences (2160 out of 5000) are incorrect annealing runs: the logical qubits are mapped to more physical qubits trough chain formation, but the qubits that represent a single variable (and must end in the same state) behave as different entities and end in different states. It is obvious that such results are not acceptable: almost half of the results don't respect the condition of the quantum adiabatic theorem and cannot be considered in the energy spectrum.

It clearly stands out in the previous example that the chain strength is a fundamental parameter which determines whether the problem is correctly mapped to the QPU or not. This means that we should try to keep chain strength within a reasonable range: large enough to avoid chain breaks, but small enough to maintain the importance of the Ising coefficients.

	s_1	s_2	s_3	s_4	s_5	energy	num_oc.	chain_b.
0	-1	-1	-1	+1	-1	-16.5	2326	0.0
2	-1	-1	+1	-1	-1	-15.5	370	0.0
1	-1	-1	+1	+1	-1	-14.5	1897	0.2
4	-1	-1	+1	+1	-1	-14.5	75	0.0
5	-1	+1	-1	-1	-1	-14.5	44	0.0
3	-1	+1	-1	+1	-1	-13.5	212	0.2
7	-1	+1	-1	+1	-1	-13.5	9	0.0
8	+1	-1	-1	-1	-1	-13.5	12	0.0
6	+1	-1	-1	+1	-1	-12.5	44	0.2
10	-1	-1	-1	-1	+1	-12.5	1	0.0
11	+1	-1	-1	+1	-1	-12.5	1	0.0
9	-1	-1	-1	+1	+1	-11.5	7	0.2
12	-1	-1	-1	+1	+1	-11.5	1	0.0
13	-1	-1	+1	-1	+1	-10.5	1	0.0

Table 3.5: Annealing results when `chain_strength` = 2. The energy is expressed in units of GHz, which can be converted in J with Eq. (3.8).

3.4 Energy distribution

The example of the 4-objects KP highlights how the probability of the correct solution and the energy gap of the system during the anneal change, and in particular decrease. We now want to understand what happens when the number of qubits increases further and how the embedding is performed by the quantum annealer. Moreover we can also focus on the energy distribution: in particular we can assume that our system follows a Boltzmann distribution and study its temperature, in order to check whether if it is consistent with the working temperature of the QPU.

3.4.1 Probability and energy gap

A knapsack problem with an increasing number of objects can be used to study the changes that occur as the number of qubits increases; for the sake of simplicity, I decided to consider an n-objects KP, where the x_i object has weight $w_i = 1$, value v_i and the total weight is $W = 1$. Obviously, the system gets more and more complex, but the solution is always given by the single most valuable object, so it is easy for us to check if the annealing worked. As previously illustrated, such problems need $n + 1$ qubits, which represent the n objects and one slack variable for W; the Ising coefficients are computed with the formulas in Eq. (3.7). Thus, these problems are easily fed to the quantum annealer: we keep the parameter `num_reads` $= 5000$, while the chain strength is not specified, and so it is automatically computed before the embedding takes place.

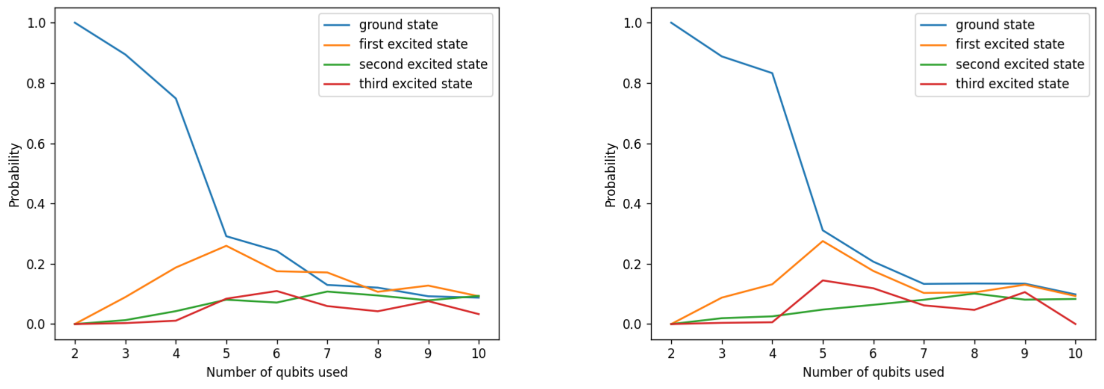


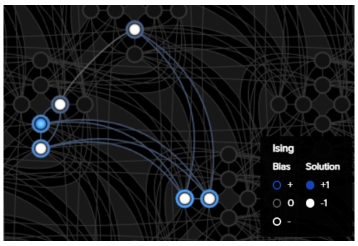
Figure 3.13: Probability trend of ground state and excited states.

Fig. 3.13 depicts two examples of the probabilities of the first four energy states that we find at the end of the anneal, when the number of qubits employed ranges between 2 and 10. We can clearly notice that, for a small number of qubits, the probability of the final state being the ground state is nearly 1 and then rapidly decreases, with an exponential-like trend. The excited states, on the other hand, start with a near zero probability, which then increases: when the number of qubits is $n > 6$, the probabilities of the four states are fairly equal, which means that, while the ground state could still be the most frequent, the system will end in most of the anneal in a generic excited state rather than in the ground state. In order to understand why such a trend shows up, we can examine the data reported in Tab. 3.6, which depicts the energies of the ground state and of the first three excited states (in GHz, conversion to $Joule$ can be done as in Eq. (3.8)) for the KPs considered. As we can see, the more qubits are used, the more the energies become higher in absolute value; moreover, degeneracy appears and many states have the same energy. We can also see that the difference between the ground state and the first excited state is always $\Delta E = 1\ GHz$: this difference is significant for a small number of qubits, when $E \approx 1 - 10\ GHz$ (and $P \approx 1$), while it becomes more and more insignificant when $E \approx 100 - 2000\ GHz$. These values of energy, which are specific for the KPs that I have considered, explain why the probability of the ground state decreases and becomes nearly equal to the one of the excited states. In fact, while the final energy gap stays still, as clearly shown in Tab. 3.6, during the anneal the band gap, i. e. the minimum energy difference between the ground state and the first excited state, becomes thinner and thinner and the system ends in an excited state more easily. This happens because we have a proliferation of states as the number of qubits increases, and the level degeneration involves the first excited state as well, which gets closer and closer to the ground state.

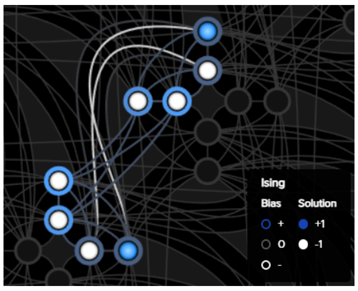
n_qubits	2	3	4	5	6	7	8	9	10
$E_{gs}(GHz)$	-1.5	-3.5	-10	-16.5	-30.5	-51.5	-81	-125	-171.5
$E_{1e}(GHz)$	-0.5	-2.5	-9	-15.5	-29.5	-50.5	-80	-124	-170.5
$E_{2e}(GHz)$	0.5	-1.5	-8	-14.5	-28.5	-49.5	-79	-123	-170.5
$E_{3e}(GHz)$	1.5	-1.5	-7	-13.5	-28.5	-49.5	-79	-123	-170.5

Table 3.6: Energy values: ground state and first three excited states. The energy is expressed in units of GHz, which can be converted in J with Eq. (3.8).

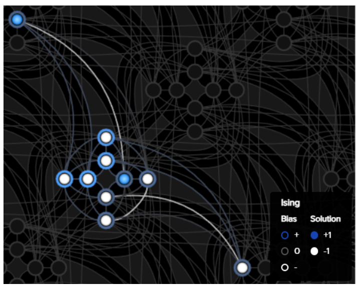
This process is also amplified by the embedding structure, which gets more and more complex (see Fig. 3.14): the number of chains and physical qubits employed rapidly increases and, as a consequence, the system loses precision on the solution of our problem. This can become an important issue when submitting large problems: sometimes we are not sure of the results and we also have difficulties in simulating the



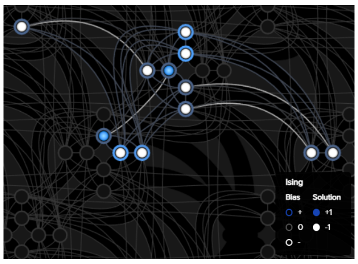
(a) 5 qubits.



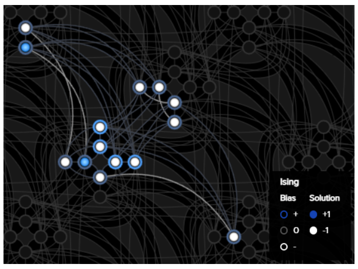
(b) 6 qubits.



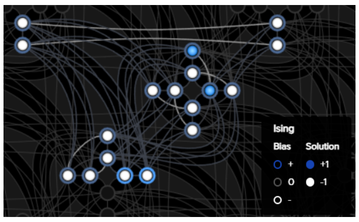
(c) 7 qubits.



(d) 8 qubits.



(e) 9 qubits.



(f) 10 qubits.

Figure 3.14: Some examples of how larger number of logical qubits are mapped to increasing number of physical qubits and chains.

system evolution on a classical computer due to the fact we would have to compute the eigenvalue and eigenvector of a $2^n \times 2^n$ matrix, where n is the number of the physical qubits used in the embedding. If we consider, for example, Fig. 3.14f, there are 18 physical qubits employed, so a $2^{18} \times 2^{18}$ matrix of which we want eigenvectors and eigenvalues; it is obvious that this cannot be computed on a classical computer.

3.4.2 Boltzmann distribution and temperature

We now want to check if the annealing process works within the correct physical working conditions, even when the number of qubits increases. In order to do so, we once again consider the Hamiltonian that describe our system, given by Eq. (2.2): it is a sum of independent terms and resembles the Hamiltonian of a set of distinguishable particles. This implies that we can consider the Boltzmann distribution as the probability distribution that reflects the probability that the system will be in a certain state as a function of that particular state's energy E_i and the temperature of the system

T, which is given by [8]:

$$P(E_i, T) = g_i \frac{e^{-\frac{E_i}{k_B T}}}{Z(T)} \tag{3.9}$$

where g_i is the degeneration for the energy E_i, while $Z(T)$ is the canonical partition function:

$$Z(T) = \sum_i^n g_i e^{-\frac{E_i}{k_B T}} \tag{3.10}$$

I therefore submitted the 9-objects KP to the Advantage_system6.2 QPU many times, for a total of 50000 samples: then I plotted the whole energy spectrum in a single graph (see Fig. 3.15) and fitted the data with the Boltzmann distribution. The ground state in the panel is the one with minimum energy and, as we can see, has less occurrences than some other states with higher energy: this is an expected trend actually, because the ground state has always $g_{gs} = 1$, while the other energies have $g_i > 1$ and so the distribution shows a peak for the first excited states. After this peak, the occurrences rapidly decrease (exponentially, as expected) and tend to zero. The fit

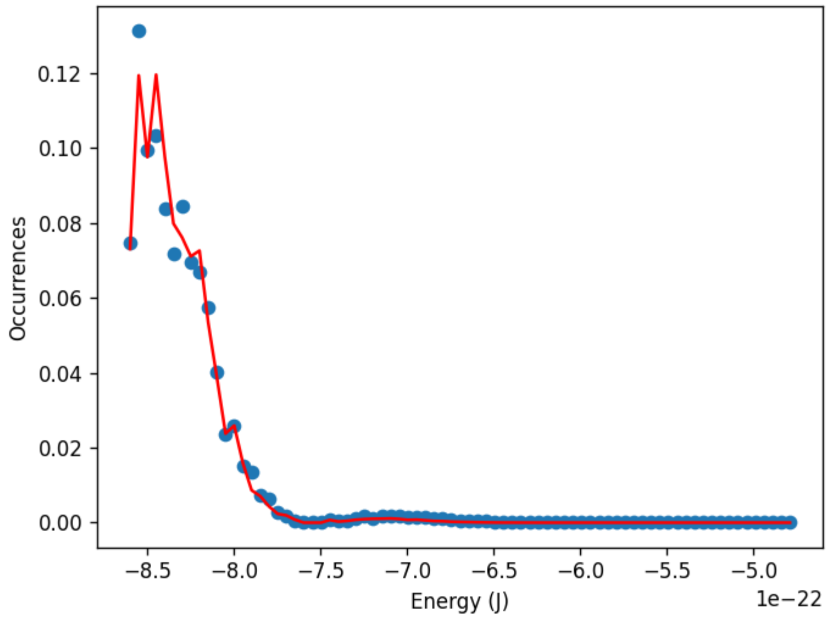
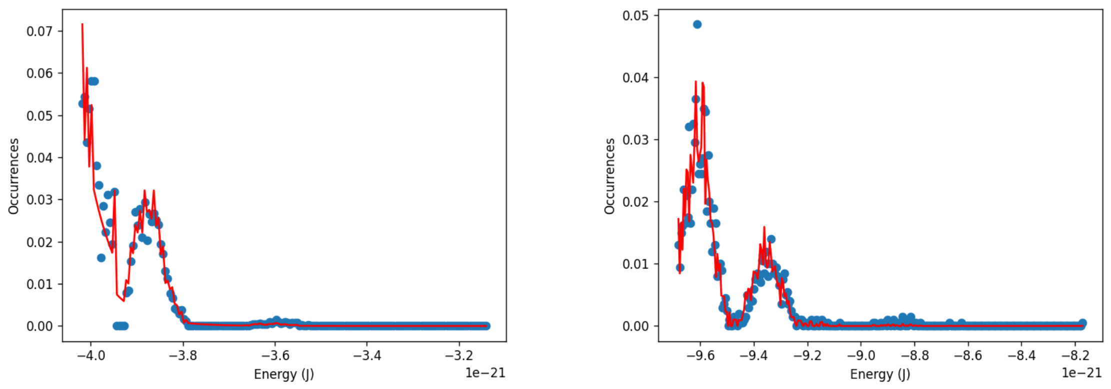


Figure 3.15: Energy distribution: the energy spectrum can be fitted using the Boltzmann distribution in Eq. (3.9).

returns the best value of the temperature for this set of data, $T = (1.798 \pm 0.001)\ K$: this value must then be compared to the working temperature of the QPU, which D-Wave sets as $T_{QPU} = 1.5\ mK$. The difference between this two temperatures explains why we have some energies which are more frequent than the energy of the ground state: the higher temperature of this distribution causes a higher thermal agitation, which makes it easier for the system to end in an excited state. However, as we can see, the temperature returned by the fit is still a really low temperature and so most of the times the annealing works well and the system stays in the ground state trough the whole process. Nevertheless, we can expect that a higher number of qubits increases the thermal agitation and worsens the energy spectrum; consequently, I similarly studied the 14-object KP and the 19-object KP, as depicted in Fig. 3.16. As it immediately stand out, I noticed the presence of a new peak, which was not present in the previous example: moreover this peak becomes more and more defined when the number of qubits increases (as in Fig. 3.16b).



(a) 15 qubits. (b) 21 qubits.

Figure 3.16: Boltzmann distributions as the number of logical qubits increases.

The presence of this peak is not expected, but can be easily explained: the energy values corresponding to the minima between the two peaks probably represent another state with $g_i = 1$, as the ground state. If this is the case, the second peak can be easily explained as the first one: the thermal agitation causes the system to easily move in other states which are degenerate and have energy a little higher than this one, which is not the ground state and is not degenerate. However, further studies are needed to better understand and characterize this phenomenon.

The two fits returned $T = 4.308 \pm 0.008$ for the 15 qubits problem and $T = 20.503 \pm 1.012$ for the 20 qubits problem, respectively. These values confirm the trend suggested before: as the number of qubits increases, so does the temperature and the thermal agitation. This causes a worse rate of success of the annealing process and the system ends in a state which is different from the ground state more times as the number of qubits increases. However, the dataset used is specific for these type of problems, which are really simple and schematic: when the problems become more difficult, this trend could be different, depending on which and how many physical qubits are employed during the embedding. This is why a general relation for the temperature as function of the number of qubits it is really difficult to derive and we can only point out this growing trend.

3.5 Execution time: QPU vs CPU

We have, so far, addressed all the issues concerning the chain formation which causes the energy landscape to change and lowers the probability of the system to end up in the ground state. Consequently, this section focuses on system timing of an annealing process and points out the enormous difference in execution time between a quantum annealing and a classical computer, where the execution time is respectively the QPU access time and the CPU time.

3.5.1 QPU timing

Fig. 3.17 shows a simplified diagram of the sequence of steps, the red set of arrows, to execute a quantum machine instruction (QMI) on a D-Wave system, starting and ending on a user's client system. Each QMI consists of a single input together with parameters: this QMI is then sent across a network to the solver API server and joins a queue, then each queued QMI is assigned to one of possibly multiple workers, which may run in parallel. A worker prepares the QMI for the quantum processing unit (QPU) and postprocessing, sends the QMI to the QPU queue, receives samples (results) and post-processes them (overlapping in time with QPU execution), and bundles the samples with additional QMI-execution information for return to the client system [5]. The total time for a QMI to pass through the D-Wave system is the service time. The

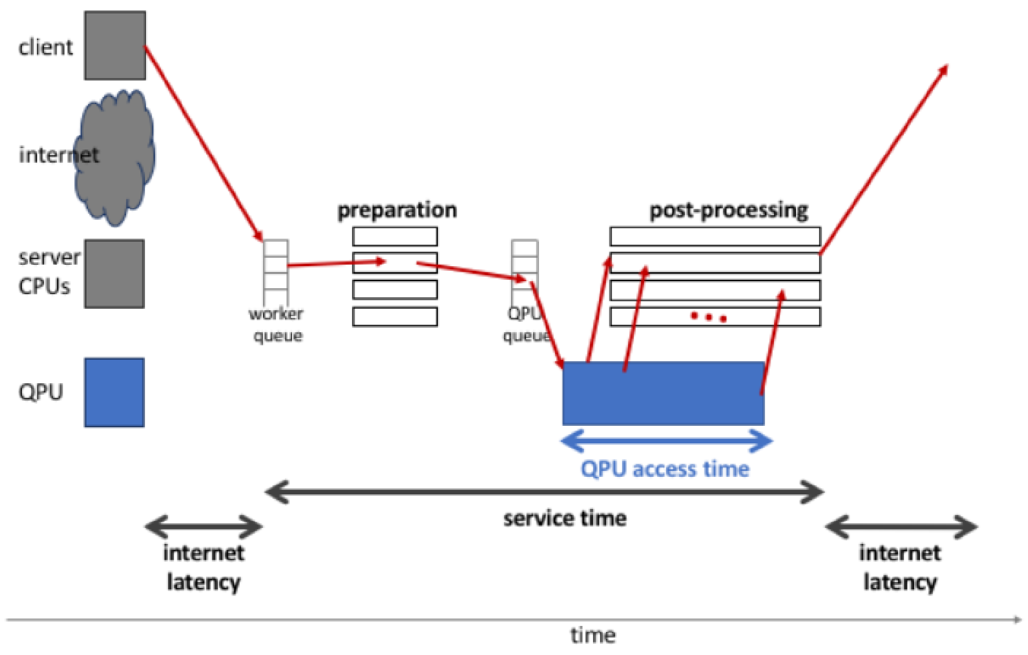


Figure 3.17: Overview of execution of a single QMI, starting from a client system, and distinguishing classical (client, CPU) and quantum (QPU) execution [5].

execution time for a QMI as observed by a client includes service time and internet latency. The QPU executes one QMI at a time, during which the QPU is unavailable to any other QMI; this execution time is known as the QMI's *QPU access time* [5]. The service time, which is defined as the difference between time-in and time-out for each QMI, can be broken into:

- Any time required by the worker before and after QPU access.
- Wait time in queues before and after QPU access
- QPU access time: this is the time in which the qubits evolve and end in one of the possible final states.
- Postprocessing time: postprocessing is a process used to improve samples, which is automatically carried out by the annealer.

We want to focus in particular on the *QPU Access Time*: this is divided in two smaller steps, a one-time initialization step to programme the QPU, where the biases and the coupler strengths are set, and typically multiple sampling times for the actual execution on the QPU, as in Fig. 3.18. The implicit horizontal axis in the diagram is time, because it shows how the QPU access time changes over time. QPU access time begins with the single largest fixed block of time: programming time. This block is the same regardless of the number of samples specified. The QPU access time also includes some overhead, so we can write that $T = T_p + \Delta + T_s$, where T_p is the programming time, T_s is the sampling time, and Δ is an initialization time spent in low-level operations, roughly 10-20 ms for Advantage systems. The time for a single sample is further broken into anneal (green), readout (read the sample from the QPU; red), and thermalization (wait for the QPU to regain its initial temperature; pink).

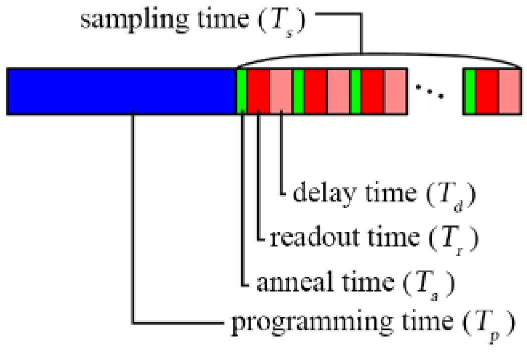


Figure 3.18: Detail of QPU access time [5].

3.5.2 Execution time dependence on the number of qubits

The QPU access time varies when the number of qubits employed changes: in particular, when we use a larger number of qubit, i. e. we submit more complex problems, the QPU access time increases. We can still use the n-objects KP problem to study this phenomenon: in Fig. 3.19 we can see how the QPU access time changes as function of the number of qubits. These data represent the mean of the QPU access time over different annealing processes when the number of qubits is fixed; the error bars are obtained as standard deviation of these set of data, where each annealing time is associated to an absolute error of $\Delta E = 0.01 \mu s$. As we can see, the distribution has a linear trend, therefore I fitted the data with the red line represented in the panel; the result I obtained is that this data distribution is compatible with a linear distribution, with a chi square $\chi^2 = 1.54$. Thus, we can assert that the QPU access time is a linear function of the number of qubits employed, at least for a small number of qubits.

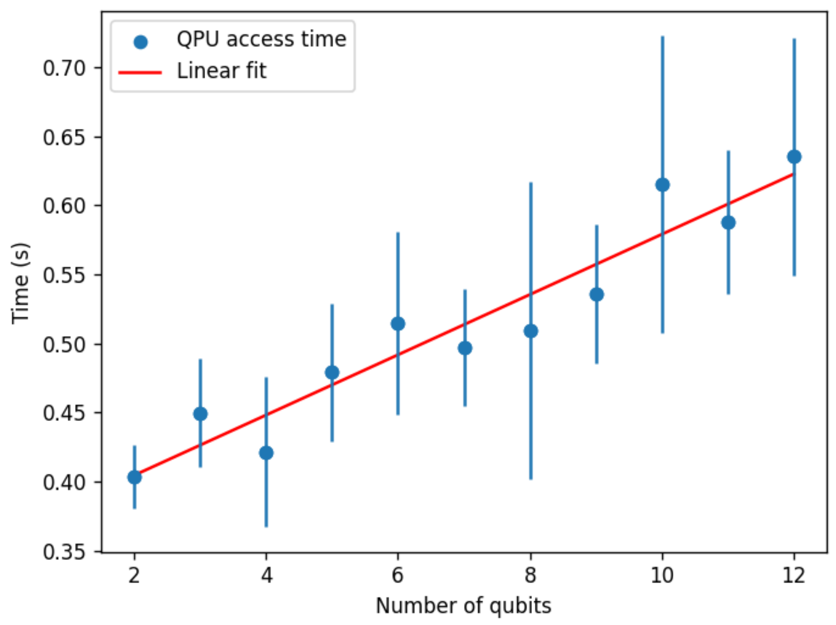


Figure 3.19: Linear trend and linear fit of the QPU access time as function of the number of qubits.

This conclusion alone is not indicative of the computational power of the quantum annealer: this is the reason why I thought it appropriate to compare our results with what happens in a classical computer when we consider a n-objects KP. The knapsack problems are classically resolved with a recursive method, where each possible combination of objects that respect the weight constraint is compared in value and at the end the best one is selected. For this comparison, I used a KP which is a little different from the previous ones: we have n objects, with value $v_i = i$ and weight $w_i = 2i - 1$ and total weight $W = 2n - 2$. This KP allows us to increase the possible combination as the number of objects increases, so it emphasizes the execution time dependence on this value. The Ising coefficients can be computed with moderate difficulty: the number of qubits used increases a lot, because we need more slack variables in order to write the weight constraint in the objective function as a binary variable, but the general procedure is the same explained in Sec. 3.2.2. The direct comparison between the quantum annealer and the classical computer (my PC) execution time is portrayed in Fig. 3.20, where the time on the y-axis is represented with a logarithmic scale. As

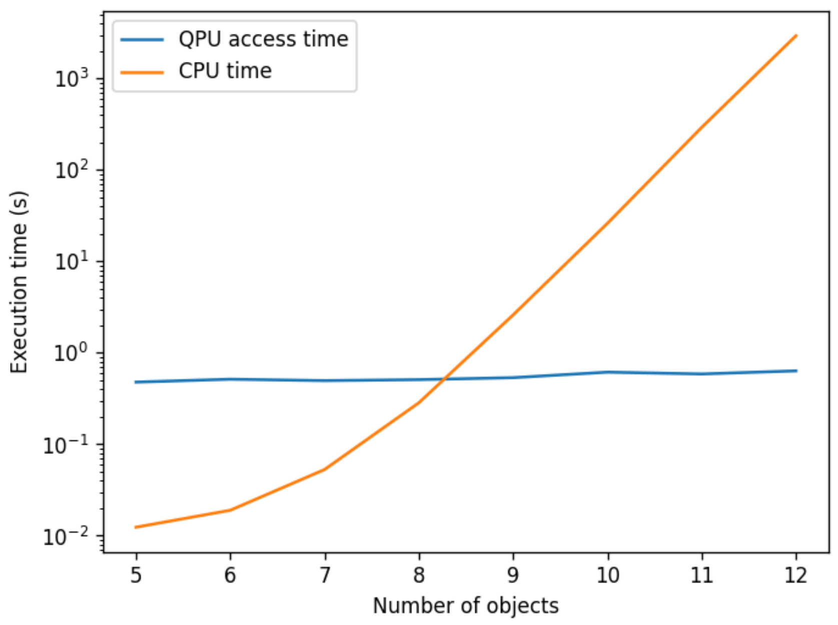


Figure 3.20: Execution time comparison: CPU (orange) and QPU (blue).

it clearly stands out and as previously mentioned, the knapsack problem on a classical computer is a NP-hard problem, which means that it cannot be resolved in polynomial time: the orange curve is in fact roughly a line in logarithmic scale, which means it is an exponential in linear scale. On the other hand, the blue line, which represents the QPU access time, is quite horizontal, which means that it has a linear or polynomial trend and increasing the number of variable of our problem does not make it uncomputable.

Conclusions

This study has explored the fascinating field of quantum annealing with a particular focus on the minor embedding process and its profound impact on the energy spectrum, final state, and attainment of correct solutions. The analysis of this process has been carried out by studying simple knapsack problems, that allowed a prediction of the correct solution and then the comparison with the solution spectrum obtained by the annealing. Through a detailed examination of the embedding process, it has become evident that the selection and mapping of problem instances onto quantum hardware plays a crucial role in achieving desirable outcomes.

The embedding process has been shown to significantly influence the final state of the system: when the number of physical qubits employed increases and the chain formation begins, the energy landscape changes and the probability of the system staying in the ground state during the whole process decreases. Thus, the more complex the problems become, the more the system will end in an excited state, rather than in the ground state, as pointed out in Sec. 3.4; this is obviously an important issue that must be tackled, in order to improve the quantum annealing efficiency in finding the correct solution. Moreover, it has been pointed out that a larger number of physical qubits involved results in an increased thermal agitation: this emphasizes the changes in the energy spectra, which consequently present more and more occurrences of excited states and a decreasing probability of the ground state.

On the other hand, the n-objects KPs have been used to study the timing of the quantum annealing process, in order to compare it with the timing performance of a normal CPU when the number of object increases: as expected, the QPU access time results to be polynomial and in particular for a small number of qubits it resembles a linear function of the number of qubits. Thus, this is a very important marker of the computational power of a quantum annealers: while the CPU time has an exponential dependence on the number of qubits, a D-Wave system can solve both complex and easier problems in a fraction of time of the same magnitude. Therefore computational optimization problems can be solved way quicker, thanks to quantum annealing, even though an increasing number of qubits worsens the energy spectra and lowers the correct solution probability.

In summary, this thesis has provided insights into the crucial role of the embedding process in quantum annealing: by optimizing the embedding, which can be achieved with a more complex topology, we can influence the energy spectrum, guide the final state of the system, and enhance the attainment of correct solutions. The small findings of this study contribute to advancing our understanding of quantum annealing: as quantum technologies continue to evolve, further research and improvements in the embedding process will undoubtedly contribute to unlocking the full potential of quantum annealing for solving complex optimization problems.

Bibliography

[1]M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*, 10th anniversary ed (Cambridge University Press, Cambridge ; New York, 2010).

[2]N. D. Mermin, *Quantum computer science: an introduction* (Cambridge University Press, Cambridge, 2007).

[3]S. Lang, *Algebra lineare* (Bollati Boringhieri, Torino, 2003).

[4]D. J. Griffiths and D. F. Schroeter, *Introduction to quantum mechanics*, Third edition (Cambridge University Press, Cambridge ; New York, NY, 2018).

[5]I. D-Wave Systems, *D-Wave System Documentation*, `https://docs.dwavesys.com/docs/latest/index.html`.

[6]M. P. Kennett, *Essential Statistical Physics*, 1st ed. (Cambridge University Press, July 16, 2020).

[7]S. Martello and P. Toth, *Knapsack problems: algorithms and computer implementations*, Wiley-Interscience Series in Discrete Mathematics and Optimization (J. Wiley & Sons, Chichester ; New York, 1990).

[8]D. A. McQuarrie, *Statistical mechanics* (University Science Books, Sausalito, Calif, 2000).

Acknowledgments

At the end of this dissertation I would like to express my heartfelt gratitude to the individuals who have contributed to the completion of this thesis. Since all involved people are native Italian speakers, you will excuse me if I continue the present section in Italian.

Innanzitutto ci tengo a ringraziare la Prof.ssa Elisa Ercolessi, che mi ha fatto scoprire e appassionare al mondo del quantum computing e mi ha permesso di realizzare questa tesi; il suo interesse verso la materia mi ha chiarito i dubbi che avevo sul futuro e mi ha convinto ad approfondire questo argomento anche nel mio percorso magistrale. Inoltre un ringraziamento dal profondo del cuore va anche al Dott. Claudio Massimiliano Sanavio, che ha avuto lo spunto per questa tesi e mi ha seguito in questi mesi di intenso lavoro: senza la sua guida e il suo supporto non sarei riuscito ad andare così a fondo nello studio del quantum annealing e del processo di embedding.

Dopodichè vorrei ringraziare la mia famiglia, che in questi tre anni di studio mi ha sempre supportato e spinto a dare il meglio di me. Ringrazio mio fratello per il suo sostanziale aiuto nella correzione e revisione di questa tesi. Ringrazio mio papà per i suoi tentativi di capire i termini che mi sente dire a tavola, anche se a volte gli scappa un 'quantistico' di troppo. Ringrazio mia mamma per avermi ascoltato ripetere prima degli esami, anche se parlo una lingua completamente diversa. Infine ringrazio i miei due cagnolini, Bianca e Lucky, per avermi tenuto compagnia tutte le mattine passate in casa da solo a studiare.

A questo punto non posso che ringraziare Andrea per questi ulteriori tre anni passati insieme: ci conosciamo dalla prima superiore e da allora è un confronto continuo, un'amicizia che spero che duri tutta la vita. Grazie per i viaggi in treno, le risate durante le lezioni e le chiamate per confrontarci sugli esercizi, anche se io inizio sempre due settimane dopo a studiare.

Ci tengo a ringraziare anche Giada e Pietro per questi tre anni, per le camminate dall'università alla stazione e per le pause pranzo passate in Pippo Re. Ringrazio inoltre tutti gli altri miei compagni di università con cui ho condiviso questi tre anni di vita, pieni di difficoltà ma soprattutto di gioie e soddisfazioni

Ringrazio anche il mio gruppo di teatro (con amici aggiunti): grazie per le serate passate a casa di Cave, per gli spettacoli e le prove più divertenti di sempre e per essere un mio punto di riferimento costante (ah e anche grazie per il Covid).

Infine l'ultimo ringraziamento va a Lucia: sei un sostegno fondamentale nella mia vita, mi hai sempre supportato quando ero in piena sessione, ascoltandomi ripetere e dandomi un modo per staccare anche dallo studio. Grazie per essere la mia pianura e la mia *λόγος*.