University of Warsaw
Interdisciplinary Centre for Mathematical and Computational Modelling

Marek Wieczorek
Student no. 426777

Title in English

Master's thesis
in COMPUTATIONAL ENGINEERING

Supervisor:
dr Marek Michalewicz

Warsaw, May 2021

Abstract

Here an abstract will show up in some months to follow.

Keywords

quantum annealing, D-Wave

Thesis domain (Socrates-Erasmus subject area codes)

11.3 Informatyka

Subject classification

D. Software
D.127. Blabalgorithms
D.127.6. Numerical blabalysis

Tytuł pracy w języku polskim

Tytuł po polsku

Contents

1. Introduction 5
- 1.1. Gate model 6
  - 1.1.1. Error correction 6
- 1.2. Quantum Annealing 6
  - 1.2.1. Thermal Annealing 6
  - 1.2.2. Simulated Annealing 6
  - 1.2.3. Quantum Annealing 8

2. How to Use Quantum Annealing 9
- 2.1. Problem Description 9
  - 2.1.1. Lenz-Ising Model 9
  - 2.1.2. QUBO 10
  - 2.1.3. Usefull Tools for formulating QUBOS 14
- 2.2. Solving 15
  - 2.2.1. Hardware 15
  - 2.2.2. Embedding 15
- 2.3. Performance 16

3. Problem 17

4. Conclusions and remarks 19

5. Scratchbook 21

Chapter 1

Introduction

There is probably no single point in space-time where the concept of *quantum computing* emerged for the very first time. In the 70' Paul Benioff was investigating the topic and in year 1980 published a foundational article *The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines*[3]. But it took celebrity-physicist and Nobel Prize winner Richard Feynman to mark the beginning of the discipline (for sure anegdoticaly). He discussed the topic in 1981 talk *Simulating physics with computers* during a small conference on the topic of *Physics of Computation* held at MIT. Present were also Paul Benioff and other scientists whose work contributed to the birth of the discipline: Rolf Landauer and Tom Toffoli [11]. In his foundational talk, Feynman indicated that for an efficient simulation of quantum phenomena, a different kind of computer was needed. The reasoning behind the idea is that quantum systems description grows exponentially with the number of particles involved. A system large enough will quickly overwhelm any classical computer. The conclusion of the talk:

> Nature isn't classical, dammit, and if you want to make a simulation of Nature, you'd better make it quantum mechanical, and by golly it's a wonderful problem because it doesn't look so easy [9].

became famous and is often treated as a spark that ignited the discipline [16][23][14]:

> With those words, in 1981, Richard Feynman, an American physicist, introduced the idea that, by harnessing quantum mechanics, it might be possible to build a new kind of computer, capable of tackling problems that would cause a run-of-the-mill machine to choke [8].

Following years have seen a gradual development of the ideas, with formalisation of the notion of a quantum computer by David Deutsch[7] and other important results, all theoretical. Major breakthrough came in 1994 when Peter Shor demonstrated a quantum algorithm for efficient factorisation of big numbers. As modern cryptography (RSA public key encryption) relies on prohibitive cost of factorizing large enough numbers, the implications of the proposed algorithm were clear and caused quantum computing to draw major attention. Some like Rolf Landauer shared their scepticism whether an effective quantum computer could be ever built. They pointed out that discussed approaches required perfect devices, neglecting effects such as *quantum decoherence*. Peter Shore was again the one to make a major contribution by proposing quantum error-correction. He proved, that a quantum code could be executed effectively even if errors occur, given that they are sufficiently rare. This of course comes at a cost — a big computational overhead.

At this stage, the theory was clearly ahead of the hardware, but the attempts to build quantum computer have been made. **What was the first practical attempt?** Digital vs. analog? How to introduce D-Wave into the topic? Is it the first quantum computer?

1.1. Gate model

Some text about the gate model computers

1.1.1. Error correction

Some for subsection of error correction

1.2. Quantum Annealing

Quantum annealing has some interesting properties. It is analog in nature and because the computation happens in a ground state it is unaffected by decoherence [19]. Quantum annealing is a computational process, an optimization metaheuristic proposed theoretically in 1988 in [2], as an analogy to simulated annealing, formulated five years earlier [17]. It was first simulated classically, to finally become realised in an analog hardware by D-Wave Inc. First working quantum annealer was presented in 2007 and contained 16 qubits [12]. To fully understand the term and to develop a bottom-up intuition of the process it is best to track the gradual development of the concept.

1.2.1. Thermal Annealing

Traditionally annealing refers to thermal annealing, a metallurgical process known to humanity for centuries. Annealing reduces hardness and increases ductility of a (most often) metal peace, making it more workable. The peace is heated until glowing (for steel) and kept in high temperature. The temperature increases the rate at which atoms diffuse in the metal, eradicating dislocations of the crystalline structure, hence moving towards equilibrium state - lower internal stresses. Lower number of dislocations results in higher ductility. The process of stress-relief is spontaneous, so it also progresses at room temperature, although at a very slow pace. It is the heat that increases the energy of thermal fluctuations, making transitions more probable. This concept is the basis on which, by analogy, optimisational algorithms were build. Annealing may also be applied to glass, with similar purposes and process. As glass has no crystalline structure, the mechanics differ [5].

A process with opposite effects is quenching, where hot material is cooled rapidly. This results in sub-optimal configuration of the material. Remaining stresses make the material harder but more brittle [6].

1.2.2. Simulated Annealing

As an optimizational algorithm, simulated annealing emerged in the early '80 in a paper by Kirkpatrick, Gelatt, and Vecchi [17]. As of 2021, it was cited almost 50 000 times, indicating how influential the idea is. The work of Kirkpatrick et al. is based on much older concept, the Metropolis algorithm by Metropolis et al. [20]. The authors devised a computational method for studying aggregate properties of the large number of particles, a subject of the statistical mechanics. As the number of particles is typically big (an order of $10^23/cm^3$) only the most probable behaviour of the system is observed in the experiments. To calculate parameters of this observable state, a Gibbs ensemble is used. It is an average of a parameter taken over all possible configurations, each defined by particle positions and weighted by a corresponding Boltzmann probability factor:

$$p \propto e^{-E/kT}$$

where k is the Boltzmann constant, T is temperature, E the potential energy of the configuration, defined as:

$$E = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} V(d_{ij})$$

where V is the potential between particles, d is the distance between them.

The averaging involves integration over a high-dimensional configuration space. As the number of particles is high, so is the dimensionality of the integral. Metropolis et al. proposed a computational method for tackling the problem of integration, based on Monte Carlo method. They studied a group of interacting particles (56 or 224) packed in a two-dimensional, closed space, at a given temperature. Standard Monte Carlo method involves random, uniform sampling of points and assigning them a weight which represents the probability of finding a particular configuration. Any randomly selected state will most probably be far from "optimum", hence with high energy and finally with low weight (probability) assigned. Standard Monte-Carlo method which involves sampling from a uniform distribution would require a huge number of configurations to be calculated, most of which with a very low contribution. Instead the authors choose configurations with probability $e^{-E/kT}$ and weight them evenly.

The algorithm starts from a random particle configuration of the particles, next following steps are executed in a loop :

1. Move a chosen particle to new positions:

$$X \to X + \alpha\gamma_1$$

$$Y \to Y + \alpha\gamma_2$$

   where X, Y are the coordinates of the particle, α is the maximum allowed step, γ_1 and γ_2 are random numbers between -1 and 1.

2. Calculate the change in energy ΔE. If $\Delta E < 0$ the move is always allowed, so we always accept "better" solution. If $\Delta E > 0$ the move is allowed with probability $e^{-E/kT}$. To realise the probability, γ_3 between 0 and 1 is randomly selected. If $\gamma_3 < e^{-E/kT}$ the move is accepted, else rejected.

3. Whether the configuration has changed or not, each step is taken into account for counting the ensembles.

The described algorithm allows for studying the properties of a set of particles at a given temperature. There is a connection with optimisation as we have discussed states of lower energy, the optimal ones. Still there is a little chance of finding them randomly. So how to transform a computational method of statistical mechanics into a optimisation algorithm? Kirkpatrick et al. realised that by lowering the temperature of the algorithm, one would simulate cooling process. At the end of the process, the Boltzmann distribution collapses into the lowest energy states. Unfortunately, low temperature is not a sufficient condition for finding the optimal state. Please refer to section **??** for detailed description of thermal processes. Rapid cooling is a process called quenching where sub-optimal state, a local minimum is produced. Process of slow cooling aiming at releasing residual stresses in the material is annealing. When done slow enough, it results in an optimal material configuration. Hence the name of the algorithm *simulated annealing*.

The process of annealing clearly is a method of optimizing the material configuration. To universally apply it to optimisation problems it is important to clarify the idea of temperature, energy and the Boltzmann constant. All these values have a clear meaning in thermodynamics but in an algorithmic sense they relate to progress parameter, cost function and a coefficient.

1.2.3. Quantum Annealing

Chapter 2

How to Use Quantum Annealing

2.1. Problem Description

Quantum annealing allows for solving a wide range of optimisation problems. A common description of the problems is needed to allows for easy comparison, tooling compatibility but also to determine what is solvable with the available equipment. This section discusses two most common descriptions which are interchangeable.

2.1.1. Lenz-Ising Model

The model is most often called *Ising model* after Ernst Ising. In his brief paper from 1925 he clearly states that he developed a concept first proposed by his research director — Wilhelm Lenz in 1920[15]. Hence I follow naming used by Brush and Niss[4][21]. Model proposed by Lenz assumed a magnetic material to be constituted of elementary micromagnets (as proposed in 1907 by P. Weiss) but with an additional constraint that each micromagnet is only able to take two positions "up" and "down". Ising developed the model by considering nearest-neighbour interactions in a linear chain of micromagnets. Exact solution obtained by Ising showed that one-dimensional model was not capable of explaining the ferromagnetism. Erroneously, Ising concluded that his result was valid in three dimensions:

> Es ist gezeigt, das für lineare Ketten und gewisse Modifikationen kein Ferromagnetismus unter den obigen Voraussetzungen entsteht. Es besteht die Vermutung, daß auch beim dreidimensionalen analogen Modell ein ferromagnetisches Verhalten nicht erzielt werden. *(It is shown that for linear chains and certain modifications no ferromagnetic effect was achieved under the above conditions. It is assumed that also in case of a three-dimensional model, ferromagnetic behaviour will not be achieved)*[15]

The ideas proposed by Lenz and Ising survived despite the lack of apparent success. In 1936 Rudolf Peierls showed that for a two-dimensional case the model indeed is capable of explaining the ferromagnetism. Progress in other areas of study such as binary alloys and transition points resulted in formulation of descriptions mathematically equivalent to Lenz-Ising model. To describe any of these phenomena in a system, one has to consider joint action of units the system is composed of, hence the term *cooperative phenomena.* Finally in 1944 Rals Onsager published a paper titled *A Two-Dimensional Model with an Order-Disorder Transition.* He provided a mathematical proof that two-dimensional Lenz-Ising model is capable of explaining ferromagnetism. The history of Lenz-Ising model is very interesting and often surprising. Please refer to Brush [4] and Niss[21] for a detailed description.

The model, however basic and idealised, proved to offer a great trade-off between realism and mathematical usability. This resulted in its great success both in terms of citations (thousands for papers of Ising and Onsager) but also a growing number of areas it is used in. One of proofs of this success is the important role the model currently plays in the area of combinatorial optimisation.

the model below is a hamiltonian. How did the original concept end up being hamiltonian? modern version of the model description

$$H(\sigma) = -\sum_{\langle i\ j\rangle} J_{ij}\sigma_i\sigma_j - \mu\sum_j h_j\sigma_j,$$

Many combinatorial optimization problems have well documented Lenz-Ising formulations. Please refer to [18] for a comprehensive collection of such formulations for many NP problems.

2.1.2. QUBO

Quadratic Unconstrained Binary Optimisation (QUBO) is an NP-hard combinatorial optimisation problem. It serves as a general mathematical description for many widely studied and relevant problems such as: Travelling Salesman Problem, Max-cut problem, Map coloring. In general form it is defined as:

$$f_Q(x) = \sum_{i=1}^{n}\sum_{j=1}^{i} q_{ij}x_ix_j$$

with $x_i \in \{0,1\}$ for $i \in [n]$ and coefficients $q_{ij} \in \mathbb{R}$ for $1 \leq j \leq i \leq n$.

An equivalent matrix notation of QUBO is:

$$min\ y = x^\top Qx$$

where $Q \in \mathbb{R}^{n\times n}$ is the symmetric $n \times n$ matrix containing the coefficients q_{ij}.

In general, QUBO is a minimisation problem, but if $f(x)$ is a function to be maximised, it is sufficient to minimise $-f(x)$. By being a mathematical description of wide use, it is similar to Lenz-Ising model. Interestingly both descriptions are interchangeable.

Natural QUBO formulations

Some of combinatorial problems fit QUBO "naturally" without any tricks. This is true for problems that do not have any constraints. One example of such problem is Number Partitioning. In common definition, its goal is to divide a set of numbers so the respective sums of the two resulting subsets are close to each other as possible. Given a set of numbers $S = \{s_1, s_2, ..., s_m\}$, let $x_i = 1$ if s_i is assigned to set 1, $x_i = 0$ otherwise. The sum for subset 1 is given as $sum_1 = \sum_{i=1}^m s_ix_i$ and the sum for subset 2 is $sum_2 = \sum_{i=1}^m s_i - \sum_{i=1}^m s_ix_i$. The difference:

$$diff = sum_1 - sum_2 = \sum_{i=1}^m s_i - 2\sum_{i=1}^m s_ix_i = c - 2\sum_{i=1}^m s_ix_i$$

Let us consider the square of the difference, hence minimum will be the optimum (0):

$$diff^2 = \left(c - 2\sum_{i=1}^m s_ix_i\right)^2 = c^2 + 4\sum_{i=1}^m s_ix_i\left(\sum_{i=1}^m s_ix_i - c\right)$$

Now we can drop constant values c^2 and 4 and convert the sums into vectors and matrix:

$$min\ y = \sum_{i=1}^{m} s_i x_i \left(\sum_{i=1}^{m} s_i x_i - c \right) = \sum_{i=1}^{m} s_i x_i (s_i x_i - c) + 2 \sum_{i=1}^{m} \sum_{\substack{j=1 \\ j \neq i}}^{m} s_i x_i s_j x_j = x^\top Q x$$

, where $q_{ii} = s_i(s_i - c)$ and $q_{ij} = q_{ji} = s_i s_j$. Elements of the Q matrix diagonal correspond to x_i^2 terms. It is important to note that because for boolean variables $x^2 = x$:

$$s_i x_i (s_i x_i - c) = s_i^2 x_i^2 - c s_i x_i = x_i^2 s_i (s_i - c)$$

Other examples of problems that naturally map to QUBO are Max-Cut Problem what else?

QUBO by reformulation

Of course the majority of interesting problems do not map so easily to QUBO. Often, additional constraints are required which do not match the "unconstrained" description. The main idea of mapping such problems is not restricting the solution but increasing the cost of infeasible solutions by using penalties. Corresponding terms are added to the basic function. If the penalty terms reach zero, whole function becomes the original, unconstrained function. Great feature of QUBO is that it is additive in nature so ideally, the final solution will be in the intersection of minimums for all constituting terms of QUBO. Sometimes such solution does not exist and some trade-offs have to be made. An increased cost is acceptable but non-physical result is clearly not. This gradation of penalty terms is done by multiplying them with weights. More important constraint will result in a penalty term with a high weight. Table 2.1 presents common constraints and their corresponding penalties.

Classical constraint	Equivalent penalty
$x_1 + x_2 \leq 1$	$P(x_1 x_2)$
$x_1 + x_2 \geq 1$	$P(1 - x_1 - x_2 + x_1 x_2)$
$x_1 + x_2 = 1$	$P(1 - x_1 - x_2 + 2x_1 x_2)$
$x_1 \leq x_2$	$P(x_1 - x_1 x_2)$
$x_1 + x_2 + x_3 \leq 1$	$P(x_1 x_2 + x_2 x_3 + x_3 x_1)$
$x_1 = x_2$	$P(x_1 + x_2 - 2x_1 x_2)$

Table 2.1: Common constraints and their corresponding penalties.

where $x_i \in \{0, 1\}$ and P is penalty weight. Also logical constraints are possible. Table 2.2 presents them. When the equality is met, the penalty term is 0. x_3 can be substituted with a constant value if needed.

Logical constraint	Equivalent penalty
$NOT(x_1) = x_2$	$P(2x_1 x_2 - x_1 - x_2 + 1)$
$AND(x_1, x_2) = x_3$	$P(x_1 x_2 - 2(x_1 + x_2)x_3 + 3x_3))$
$OR(x_1, x_2) = x_3$	$P(x_2 x_1 + (x_1 + x_2)(1 - 2x_3) + x_3)$

Table 2.2: Logical constraints and their corresponding penalties[22][24].

For constructing more elaborate penalties, logical gates are also of use. Table Table 2.3 presents such gates and their corresponding penalties. Please note that AND gate appeared in table Table 2.1.

Logical gate	Equivalent penalty
$NOT(x_1)$	$P(1 - x_1)$
$AND(x_1, x_2)$	$P(x_1 x_2)$
$OR(x_1, x_2)$	$P(x_1 + x_2 - x_1 x_2)$

Table 2.3: Logical gates and their corresponding penalties [22][24].

Now let us consider a practical example of Maximal Independent Set (equivalent to Set Packing). The task is best described as graph coloring problem: mark as many graph vertices so that no two colored vertices are connected by an edge. Let us define $x_i = 1$ if vertex i is colored, $x_i = 0$ otherwise. The objective is:

$$max \sum_{i=1}^{m} x_i$$

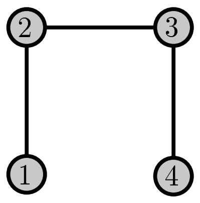
with additional constraints that no two vertices sharing an edge are allowed, so penalty has to be applied if $AND(x_i, x_j)$. We can use Table 2.3 or Table 2.1 and formulate a set of constraints:

$$P \sum_{ij \in E}^{m} x_i x_j$$

finally, one has to remember that minimisation task should be considered, so finally:

$$min \ - \sum_{i=1}^{m} x_i + P \sum_{ij \in E}^{m} x_i x_j$$

Figure 2.1: Simple graph with three edges.



For a particular graph presented in Table 2.1.2 the task becomes:

$$min \ \left[- \sum_{i=1}^{4} x_i + P \sum_{i=1}^{3} x_i x_{i+1} \right]$$

Table 2.4 contains QUBO values for all possible solutions and 3 values of penalty factor $P \in 1, 2, 3$. Valid solutions (rows 1-5, 10 and 11) are unaffected by the value of P. This is easily explained by "switching-off" of the penalties for valid solutions. Row 12 is an interesting case: for $P = 1$ is seems to be among the best solutions but that is not the case for any other value of P. Clearly, P that is to low might not exclude invalid solutions. So should it then be much higher? Rightmost column contains solutions for $P = 10$. Clearly the problem

row	$x1$	$x2$	$x3$	$x4$	$P = 1$	$P = 2$	$P = 10$
1	0	0	0	0	0	0	0
2	0	0	0	1	-1	-1	-1
3	0	0	1	0	-1	-1	-1
4	0	1	0	0	-1	-1	-1
5	1	0	0	0	-1	-1	-1
6	0	0	1	1	-1	0	8
7	0	1	0	1	-2	-2	-2
8	0	1	1	0	-1	0	8
9	0	1	1	1	-1	1	17
10	1	0	0	1	-2	-2	-2
11	1	0	1	0	-2	-2	-2
12	1	0	1	1	-2	-1	7
13	1	1	0	0	-1	0	8
14	1	1	0	1	-2	-1	7
15	1	1	1	0	-1	1	17
16	1	1	1	1	-1	2	26

Table 2.4: All solution of Maximal Independent Set QUBO

discussed earlier is solved, unfortunately other issue arises: the gap between best and second best solutions gets much smaller compared to overall range of possible values ($\frac{1}{28} = 3,6\%$). For $P = 2$ the gap is 25%. when is it a problem? DWave 4bit precision? Is it for couplings or readout as well?

QUBO - graph equivalence

Let us consider one more example: 3 binary variables a,b, c. Exactly one of them should be equal 1. We can think of a constraint:

$$a + b + c = 1$$

As we seek a minimization task, it is best to reformulate the constraint:

$$min(a + b + c - 1)^2$$

Without this step, $a = 0$, $b = 0$, $c = 0$ would make up the minimum, despite the fact that the original constraint is not satisfied. After expansion:

$$min(a^2 + b^2 + c^2 + 2ab + 2ac + 2bc - 2a - 2b - 2c + 1)$$

Note that for binary variables $x = x^2$, also a constant value increases or decreases all solution values by same amount, hence it can be omitted. Finally:

$$min(2ab + 2ac + 2bc - a - b - c)$$

Now let us change the notation:

$$min(-\sum_{i=1}^{3} x_i + 2\sum_{i=1}^{3}\sum_{\substack{j=1 \\ j\neq i}}^{3} x_i x_j)$$

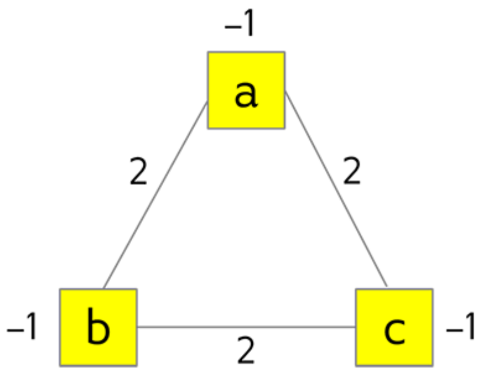


Figure 2.2: A graph equivalent to QUBO describing three binary variables, where only one should be equal 1 (true) [13].

After this step it becomes clear that this is an example of QUBO. In previous example it was shown how to describe a graph problem with a QUBO, but the reverse is also possible. In fact, each QUBO describes a graph. This key property that is used by D-Wave quantum annealers - a graph is constructed in the hardware and its annealing takes place. A graph of current example is shown in Figure 2.2. q_i coefficients describe node biases, q_{ij} describe edge weights. Considering $i, j \in \{1, \ldots N\}$, it becomes clear that a universal hardware that allows to construct any QUBO with N variables, should be a hardware chip with topology of a complete graph with N nodes. With sufficiently high N, this becomes impossible due to enormous number of required edges. This disparity and methods for dealing with it are discussed in subsection 2.2.2.

There of course are many cases that do not fit natural QUBO formulation discussed in **??** nor are required penalties formulated as discussed in this section. For such cases a more general method is available. Please refer to [10] for detailed explanation. It is also possible to use dedicated software libraries as discussed in subsection 2.1.3.

2.1.3. Usefull Tools for formulating QUBOS

As presented before, it is possible to formulate a QUBO by hand. This approach is limited by the type of problems - not all required penalties may be available, but also the size of problems. It is very helpful to use dedicated tools, many of which are open source. Below, a selection of tools is presented.

D-Wave dimod

D-Wave dimod is a shared API for various D-Wave samplers. It contains Binary Quadratic Model (BQM) class for quadratic models: Lenz-Ising and QUBO. Also possible are models with integer and discrete variables. The library also enables reduction of higher order models to quadratic ones as well as generators for specific classes of problems eg. knapsack or bin packing problems. The main strength of the library is the ease of translation between various equivalent formulations. One can formulate a constrained problem and solve it as a QUBO. The library is available under Apache License 2.0 and has documentation[1].

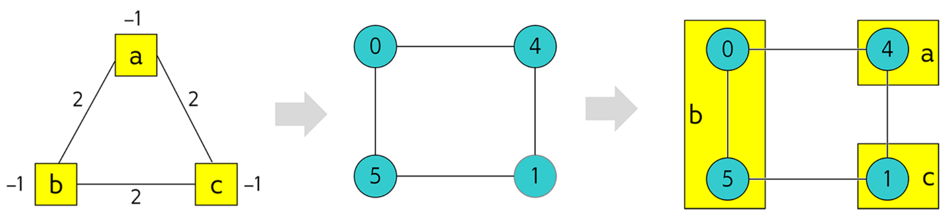


Figure 2.3: Embedding of a triangle graph into Chimera topology [13].

pyqubo

pyqubo is a library created outside of D-Wave Systems Inc. but fully integrated with Ocean SDK. It allows for formulation of QUBOs and Lenz-Ising models using mathematical expressions, their translation and export to BQM class of D-Wave dimod. The library is available under Apache License 2.0[22][24]. so what is the difference between dimod and pyqubo?

find qubo

BSD 3-Clause License

2.2. Solving

2.2.1. Hardware

D-Wave

D-Wave

Other companies

2.2.2. Embedding

As described in section 2.1.2, each QUBO has a corresponding graph. This graph is constructed using D-Wave hardware: superconducting qubits represent graph nodes, couplers represent graph edges. At best, the topology of the hardware should be a complete graph. This would enable a hardware realisation of any QUBO with number of variables up to the total number of qubits in a chip - n. This however, would require $\frac{n(n-1)}{2}$ of couplers. For a sufficiently large n this becomes a prohibitively large number considering the complexity and technical limitations of chip manufacturing. Considering the actual topologies of D-Wave chips (as described in section 2.2.1) and an arbitrary QUBO one would like to solve, there arises a disparity, to some extent solved by embedding. It is a process of "fitting" a graph into the hardware topology, described with a different graph. A key concept of embedding is the representation of a single logical node with many hardware nodes (qubits). This is done by strong coupling of two or more nodes, so they behave in a same way, mimicking being the same node. For analogy, one can think of two points in an electrical circuit. If they behave the same, so when the electrical potential between them is 0, it is equivalent to a circuit where two or more points become one.

Let us recall a graph from Figure 2.2. It is a complete graph K_3, a triangle. Now let us go back to the Chimera topology as shown in section 2.2.1. It is clear that there is no single triangle in the hardware topology of chips using Chimera structure. Even in such a simple case, embedding is necessary. Figure 2.3 shows the process conceptually.

NP of embedding itself de-embedding D-Wave tools for embedding and de-embedding

2.3. Performance

Chapter 3

Problem

Chapter 4

Conclusions and remarks

Chapter 5

Scratchbook

Ocean tool-kit is the software suite currently used by DWave to solve computational problems. It consists of various sub-modules, each aimed at different stage of the process. The tool-kit is freely available on Github [**?**].

Bibliography

[1] D-Wave Systems Inc . dimod — Ocean Documentation 4.3.0 documentation, 2021.

[2] B. Apolloni, N. Cesa-Bianchi, and D. (Milan Univ (Italy) Dipt di Scienze dell'Informazione) De Falco. A numerical implementation of 'quantum annealing', 1988.

[3] Paul Benioff. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. *Journal of Statistical Physics*, 22(5):563–591, May 1980.

[4] Stephen G. Brush. History of the Lenz-Ising Model. *Reviews of Modern Physics*, 39(4):883–893, October 1967.

[5] Wikipedia contributors. Annealing (metallurgy), March 2021. Page Version ID: 1015346526.

[6] Wikipedia contributors. Quenching, April 2021. Page Version ID: 1020513336.

[7] David Deutsch. Quantum theory, the Church–Turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 400(1818):97–117, July 1985.

[8] The Economist. Google claims to have demonstrated "quantum supremacy" | The Economist, 2019.

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

[10] Fred Glover, Gary Kochenberger, and Yu Du. Quantum Bridge Analytics I: a tutorial on formulating and using QUBO models. *4OR*, 17(4):335–371, December 2019.

[11] MIT Endicott House. The Physics of Computation Conference, March 2018.

[12] D-Wave Inc. D-Wave Demonstrates First Quantum Computer - ExtremeTech, 2007.

[13] D-Wave Inc. D-Wave System Documentation, 2022.

[14] The Quantum Insider. Global Chip Crisis – Quantum Computing Could Be The Answer, November 2021. Section: Business.

[15] Ernst Ising. Beitrag zur Theorie des Ferromagnetismus. *Zeitschrift für Physik*, 31(1):253–258, February 1925.

[16] John Preskill. Feynman verbatim: "Nature isn't classical, dammit, and if you want to make a simulation of Nature, you'd better make it quantum mechanical", March 2017.

[17] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220(4598):671–680, May 1983. Publisher: American Association for the Advancement of Science.

[18] Andrew Lucas. Ising formulations of many NP problems. *Frontiers in physics*, 2:5, 2014. Publisher: Frontiers.

[19] Catherine C. McGeoch. Adiabatic Quantum Computation and Quantum Annealing: Theory and Practice. *Synthesis Lectures on Quantum Computing*, 5(2):1–93, July 2014. Publisher: Morgan & Claypool Publishers.

[20] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092, June 1953. Publisher: American Institute of Physics.

[21] Martin Niss. History of the Lenz-Ising Model 1920?1950: From Ferromagnetic to Cooperative Phenomena. *Archive for History of Exact Sciences*, 59(3):267–318, March 2005.

[22] Kotaro Tanahashi, Shinichi Takayanagi, Tomomitsu Motohashi, and Shu Tanaka. Application of Ising Machines and a Software Development for Ising Machines. *Journal of the Physical Society of Japan*, 88(6):061010, June 2019. Publisher: The Physical Society of Japan.

[23] Andreas Trabesinger. Quantum simulation. *Nature Physics*, 8(4):263–263, April 2012. Bandiera_abtest: a Cg_type: Nature Research Journals Number: 4 Primary_atype: Editorial Publisher: Nature Publishing Group Subject_term: Quantum information Subject_term_id: quantum-information.

[24] Mashiyat Zaman, Kotaro Tanahashi, and Shu Tanaka. PyQUBO: Python Library for QUBO Creation. *IEEE Transactions on Computers*, pages 1–1, 2021. Conference Name: IEEE Transactions on Computers.