Alma Mater Studiorum – Università di Bologna

DOTTORATO DI RICERCA IN

DATA SCIENCE AND COMPUTATION

35° ciclo

Settore Concorsuale: 09/H1 - SISTEMI DI ELABORAZIONE DELLE INFORMAZIONI

Settore Scientifico Disciplinare: ING-INF/05 - SISTEMI DI ELABORAZIONE DELLE INFORMAZIONI

ASSESSING THE CURRENT STATE OF ADIABATIC QUANTUM COMPUTERS IN SOLVING CHALLENGING COMPUTATIONAL PROBLEMS

Presentata da : Lorenzo Rocutto

Coordinatore Dottorato:
Prof. Daniele Bonacorsi

Supervisore:
Prof. Andrea Cavalli

Co-Supervisore:
Dott. Sergio Decherchi

Esame finale anno 2024

Acknowledgements

I would like to thank all my colleagues and co-authors for their work, with a special thanks to Marco Maronese, that thought me a lot and supported me daily during my PhD studies. I want to thank my supervisor, Andrea Cavalli, for the amazing opportunities to learn and grow he provided me, and my co-supervisor, Sergio Decherchi, for the support he showed me and for all the time he spent teaching me how to do research and write articles. I am also really grateful to the Quantum Team at the CNR-IFN in Milan and to all the researchers and students working there. A special thanks goes to Enrico Prati, who supported my research work and helped me to grow as a scientist, and to Lorenzo Moro.

I am also really grateful to Fabio Traversa, Memcomputing CTO, for hosting me at Memcomputing offices in San Diego for three months and for all he taught me.

I also want to acknowledge Leonardo company for giving me the opportunity to collaborate with Daniele Dragoni on an interesting research project that is now part of this Thesis.

Then, I would like to thank CINECA for providing access to computational time on D-Wave Systems' quantum computers through the ISCRA-C project FENSIA. Such time resources were essential to complete the results soon to be published as [1]. We are also grateful for the additional time we won via a second ISCRA-C project that we put to good use in [2].

I gratefully acknowledge D-Wave for partially granting access to the D-Wave 2000Q System processor, used to obtain the experimental results presented in [3] and [4]. For that project, I would like to also thank Daniele Ottaviani and Carlo Cavazzoni of CINECA for useful discussions.

A last thanks goes to Paolo Gastaldo and Daniele Ottaviani, for reviewing the present Thesis and suggesting useful improvements.

Introduction to the Thesis (Abstract)

Since Charles Babbage devised and designed his *Analytical Engine*[5] humanity has put great efforts in devising methods to harness natural phenomena to perform computation. During the XX century, the Von Neumann–Zuse Architecture [6], and hence the Turing paradigm [7] affirmed as the most proficient and useful approach to computing. Since then and up to now, most of the computing processes worldwide have been brought on using processors based on that same architecture, despite almost any natural process is in principle as complex as a Turing machine [8]. This all-in strategy allowed the computing power density of the processors to increase exponentially during the second half of the XX century, following the so-called Moore's law [9], [10]. During the last two decades, several positive trends into integrated processors have slowed down from the forecast exponential speed [11], mainly due to the energy dissipation during the switch events and the physical inferior limit on the size of the transistor [12]. Today, computing machines based on the Von Neumann–Zuse architecture continue improving, mainly thanks to the parallelization of computational processes via GPUs [13], but the scientific community has realized the inherent limits of this approach to computation. The "classical" computing paradigm will never be able to efficiently solve some relevant modern industrial challenges, mainly those that comprise NP-hard problems[1].

The computational limits of the "classical" approach can be overcome by developing new computational paradigms based on different natural phenomena. In 1980, in the Soviet Union, Yuri Manin published a book suggesting the idea of a computer based on quantum mechanics, capable of simulating complex quantum mechanical systems [14]. In the same year, in the USA, Paul Benioff published an article describing a quantum mechanical model of the Turing machine [15]. The proposed approach exploited *quantum gates* to perform calculations, a reversible analogue of the logical gates used in classical computation. Being reversible, there is no need for a thermal dissipation following

[1]NP-hard problems are problems at least as hard as the hardest decision problems whose solution can be verified in polynomial time.

a logical gate, as it is the case for classical computing. Despite this interesting feature, the idea of a QC (quantum computer) didn't gain popularity yet. Things changed after the seminal works of Deutsch and Jozsa [16], Grover [17] and Shor [18]. They showed that QCs could have an exponential speed up over classical computers in decision problems on Boolean functions, in inverting Boolean functions, and in factoring semiprimes. Since then, other works have found speedups for mathematical procedures implemented on QCs, such as the fast quantum fourier transform [19]. The promise of a new powerful approach to computing sparked a great interest that pushed companies to take the first steps into the realization of the blueprints for prototypical quantum computing hardware. In a seminal work, Deutsch [20] introduced the concept of the universal quantum gate, forecasting that QCs would have been realized in the future as networks of instances of such gate (*quantum computational networks*). Although the prophecy realized in the form of gate-based QCs, nowadays there are many different physical implementations of the quantum computing concept. These devices can vary widely, both in terms of the hardware technology employed and the computational logic they utilize.

One of the most mature hardware realizations of the QC concept is the Adiabatic Quantum Computer (AQC). The hardware of an AQC comprises thousand of small superconducting loops where quantum bits of information (qubits) can be encoded by carefully tuning the superconducting currents. This ensamble of qubits is then exposed to a time-dependent Hamiltonian that drives the system towards the minimum of a binary quadratic cost function by exploiting the adiabatic theorem. Thus, AQCs constitute a new quantum approach to solve hard optimization problems. AQCs cannot currently implement all quantum algorithms, which implies they are not considered universal QCs. Despite this shortcoming, AQCs are able to tackle large problems comprising thousand of variables, since they are made up of up to five thousand qubits [21], as opposed to the universal gate-based competitors that can reach up to few hundreds qubits [22].

During the last four PhD years, we analyzed the capabilities of modern AQCs in solving hard optimization problems stemming from industrially relevant challenges. In particular, we devised some useful practical recipes to tune the internal parameters of the AQC and enhance its performances. By using such techniques we were able to reduce the computational time required to find the global optimum of a problem by up to 78 times. We also tested the possible use of AQCs to produce thermal samples faster than classical approaches. Despite this latter application is less explored in literature, we were able to demonstrate that current AQCs can already outperform classical algorithms implemented on GPUs in specific sampling tasks that have an application in Machine Learning.

In this Thesis I aim to achieve three main objectives. The first is to present and comment the experimental results obtained during my PhD [1]–[4], [23], [24], which show that performances of modern AQCs are steadily improving thanks to both hardware and software advancements, and are getting close to practical utility.

The second objective is to provide the reader with an extensive literature review regarding the design, the use, the capabilities, and the possible applications of AQCs. Indeed, there is currently no document or book on AQCs that contains the comprehensive amount of commented and ordered references presented in this Thesis. Thus, I hope this work can serve as an introductory compendium to a good portion of the modern knowledge regarding this topic.

The third objective is to provide the reader with a collection of the most popular and effective ways to manipulate an AQC at the software (middelware) level in order to enhance its performances.

We tried to achieve these three objectives in the four main Chapters of this Thesis, which are named, respectively, *Experimental results* (first aim), *Theory* and *Practical applications* (second aim), and *Performance optimization techniques* (third aim).

In the *Theory* Chapter (1), we introduce the concept of Quantum Computing and present the current alternative approaches to realize a physical QC. We then introduce the adiabatic paradigm to quantum computation from both a mathematical and a hardware perspective. Next, in the *Practical Applications of Adiabatic Quantum Computers and other unconventional computing paradigms* Chapter 2 we discuss the practical applications of AQCs, chiefly solving complex optimization problems and quickly generating configurations sampled from a thermal distribution. Then we describe other unconventional computing platforms that can be considered competitors of the AQC, with particular attention given to the Memcomputing machine, which we experimentally compared to an AQC in [2]. The last Section of this Chapter present a brief literature review of experimental results that benchmark AQCs' performance to other classical or exotic computational paradigms.

In the *Performance optimization techniques* Chapter (3) we introduce several software-level techniques that aim to enhance the performances of hardware AQCs. Hybrid quantum-classical approaches and potential future hardware improvements are also discussed.

In the *Results* Chapter (4), we present the experimental results obtained in the four main research works I conducted during my PhD. The first two experiments focus on solving industrially-relevant optimization problems, while the other two focus on a sampling application, namely the use of AQCs to train an unsupervised learning model known as the Boltzmann Machine. Each work is described in its design and experimental outcome.

I want to summarize here few highlights originated from my PhD work:

- Despite being publicized as optimization problem solvers, AQCs can already show a computational advantage with respect to classical algorithms in sampling applications. The parallel quantum annealing approach is fundamental to achieve such advantage.

- A proper tuning of the internal parameters of an AQC can result in a considerable performance boost, reducing the computational time required to reach the global optimum of an optimization problem by up to 78 times.

- Literature presents several proposals to improve the hardware architecture of AQCs. Theoretical arguments show that AQCs could evolve in the future to host fully-connected problems of any size and to achieve the state of universal quantum computers.

Contents

Chapter 1

Theory

1.1 Introduction to Quantum Computing

What all existing QCs have in common is the concept of the quantum bit, which was famously named the "qubit" by B. Schumacher in 1995 [25]. The qubit serves as the fundamental unit of logic in these computers, and its state exists as a quantum superposition of two classical states. Unlike a classical bit, which can only take on a value of 0 or 1, a qubit typically exists in a quantum state represented by the equation:

$$|\phi\rangle = \alpha |0\rangle + \beta |1\rangle \,, \tag{1.1}$$

where α and β are complex coefficients that satisfy the condition $|\alpha|^2 + |\beta|^2 = 1$. Different quantum computing devices differentiate in both the hardware architecture[1] and the way they perform calculations[2].

In 2018, the most powerful classical simulations could emulate a universal a 50 qubits QC. John Preskill then coined the term NISQ (Noisy Intermediate-Scale Quantum) era to identify the future moment in which QC comprising hundreds of qubits will be available, achieving the first computational processes beyond classical capabilities [26]. Depite the one-hundred qubits threshold has been surpassed few years ago in universal QCs [27], the efficiency of simulations of QCs on classical hardware has improved as well. It still debated when QCs will achieve super-classical capabilities on a specific task (*quantum supremacy*). In [28] the Google team claimed to have reached Quantum Supremacy with their Sycamore QPU, while in [29] the IBM team contended such claim. Later, [30] and [31] showed how a modern supercomputer can match the aforementioned "quantum supremacy" performances.

[1]See Section 1.1.2.
[2]See Section 1.1.1.

The NISQ era is yet to come mainly because modern implementations violate all of the necessary conditions to build a functioning quantum computer introduced in 2000 by the theoretical physicist David P. DiVincenzo [32]:

- A scalable physical system with well-characterized qubit
- The ability to initialize the state of the qubits to a simple fiducial state
- Long relevant decoherence times
- A universal set of quantum gates
- A qubit-specific measurement capability

It should be noted that some QC companies, such as D-Wave, have purposely avoided to respect the rule regarding the universal set of quantum gates. This choice was made to realize a non-universal quantum-based device capable of surpassing classical performances on specific optimization tasks [33].

In the next sections, the current efforts in building a quantum computer are described, focusing on both the computational paradigms and hardware architectures.

1.1.1 Computational paradigms

The first distinction we can make among modern quantum computers is based on the computational paradigm, namely the particular manner in which quantum phenomena within the device are harnessed for computational purposes.

Gate-based quantum computers

– Numerous existing physical devices aim to implement the QC concept introduced by Deutsch [20], which is envisioned as a machine capable of performing calculations by manipulating the quantum state of individual qubits through the application of *quantum gates* [34]. These gates function as unitary operators, acting on single-qubit or multi-qubit states [35]. Certain quantum gates generate entangled states among input qubits, enabling the execution of algorithms that exhibit exponential speedup compared to their classical counterparts. For instance, Shor's algorithm [18] utilizes quantum gates to efficiently factorize integers in polynomial time (with respect to the input length). Quantum computers employing quantum gates are commonly referred to as *circuital quantum computers* or alternatively, *gate-model quantum computers*.

One of the most notable advantages of this computational paradigm is its ability to realize any conceivable unitary operation on qubits. This implies that virtually every quantum algorithm can be executed on such devices, making most gate-based quantum computers universal in nature. To qualify as a universal quantum computer, a quantum device must incorporate a sufficient repertoire of quantum gates to approximate any desired unitary operation.

Prominent companies such as IBM [36], [37], Google [28], Rigetti [38], and Intel [39] are presently engaged in quantum hardware development, exploring the capabilities of gate-model quantum computers.

Measurement-based Quantum Computing

Measurement-based quantum computation [40] is achieved by performing a sequence of measurements on the qubits of a properly initialized entangled state. The measurement results are then used to determine subsequent measurement operations, effectively guiding the computation through a feedback loop of measurements (adaptive measurements). The final measurements produce the result, which encodes the output of the computation.

The two most prominent examples of measurement-based quantum computation are the teleportation-based approach [41] and the so called "one-way quantum computer" [42].

The main difference between this two approaches comes from the requirement of teleportation-based approach to use joint measurements, measurement operations that simultaneously evaluate multiple observables exploiting the entanglement between qubits. On the other hand, in a one-way quantum computer universal computation can be achieved with single-qubit measurements alone.

A one-way computer performs calculations starting from a highly entangled state, known as a cluster state [43]. The cluster state is created by entangling multiple qubits in a specific pattern, usually on a two-dimensional lattice. The entire resource for the computation is provided by the entangled cluster state in which the system is initialized. Consequently, the challenge of realizing a quantum computer is reduced to preparing a specific multi-particle state and implement single-qubit measurements, which may offer advantages in suitable physical set-ups.

A few years ago, Xanadu, the current world-leading company in photonic quantum computing hardware, presented a blueprint for the realization of a scalable photonic fault-tolerant quantum computer [44].

Adiabatic Quantum Computing

By exploiting the adiabatic theorem, Adiabatic Quantum Computing (AQC) employs a time-dependent Hamiltonian to drive a system of entangled qubits towards a configuration that represents a superposition of the global minima of a given classical optimization problem.

Contrary to the other computational paradigms, AQC does not enable universal quantum computation. In fact, AQC does not even prescribe to execute operations at specific time steps.

This computational paradigm constitutes the central focus of this thesis. Section 1.2 provides a historical introduction to the AQC concept and a detailed exploration from a mathematical perspective.

1.1.2 Hardware architectures

It took approximately 20 years to go from the original concept of a quantum computer by Benioff in 1980 to the first implementation of a quantum algorithm. In 1998, Chuang et al. [45] were among the first to build an empirical system with physical qubits to compute an instance of the Deutsch–Jozsa (D-J) quantum algorithm [16]. The algorithm determines whether an unknown Boolean function is constant (same image for every input) or balanced (half of the possible input have image 1, the others have image 0). The D-J algorithm was applied in its simplest possible form, using only one input bit (two balanced functions and two constant functions can be defined on a single input bit). The prototypal QC was composed by two atoms, whose nuclear spins were used as qubits. The atoms were immersed in a strong, static external magnetic field, and the quantum gates on the qubits were realized via pulsed radiofrequency electromagnetic fields, effectively exploiting nuclear magnetic resonance (NMR).

Another relevant milestone was the 1999 article by Nakamura et al. [46], who suggested that a superconducting circuit could serve as a stable quantum bit, and that the state of such qubit can be controlled by applying a short voltage pulse. Few months later, Friedman et al. [47] demonstrated in a superconducting circuit the presence of a quantum superposition of macroscopically distinct states, namely the flow of a current clockwise or counter-clockwise. This was one of the first examples of a macroscopic object exhibiting quantum properties.

Since then, many different technologies have been developed to realize a quantum computer. The following Sections present a high-level overview on various approaches to physically encode and manipulate quantum information.

Superconducting qubits

The first qubits realized in history constituted a groundbreaking discovery that ignited the advent of quantum computing technologies. They were initially implemented using loops of superconducting materials called SQUIDs (Superconducting Quantum Interference Devices) [48]. Many gate-based quantum computers and the majority of Adiabatic Quantum Computers (AQCs) perform operations on SQUID-based qubits. For a more detailed exploration of superconducting qubits from a hardware perspective, please refer to Section 1.3.2.

Integrated quantum photonics

In photonic quantum computers, information is encoded in photonic quantum states and manipulated using optical elements. In this context, qubits states are typically defined based on the polarization of photons. Photonic quantum computers exhibit good coherence times, thanks to the tendency of photons to interact weakly among themselves. For this same reason, performing entangling operations can be challenging. The concept of a quantum computer based on photons originated from the influential work of Knill, Laflamme, and Milburn in 2001 [49]. They demonstrated that efficient quantum computation can be achieved using beam splitters, phase shifters, single photon sources, and photodetectors. The feasibility of this groundbreaking idea was subsequently validated through laboratory experiments [50], [51]. Currently, several prominent companies, including Xanadu [44], PsiQuantum [52], Quandela [53], Quix Quantum [54], and ORCA Computing [55], are actively involved in the development of integrated quantum photonics for quantum computing.

Neutral atoms

As anticipated, nuclear degrees of freedom were among the first quantum variables to be used as qubits. Despite the current popularity of superconducting circuits and integrated photonics as hardware realizations of QCs, atoms are still considered a viable alternative by many researchers. Atomic set ups for QC can be roughly divided into using *neutral atoms* or *ions*. This section introduces neutral atoms devices, while the next one will explain how to build QCs based on trapped ions.

A possible approach to realize QCs based on ultracold atoms is to use Rydberg atoms [56], which are named after the Rydberg states, high-energy orbitals much farther away from the nucleus than the ground state. In fact, the highest-energy electrons in atoms such as Rubidium and Caesium can be excited to Rydberg states, making them sensitive to electric and magnetic fields conveyed via lasers. The computational basis ofthe qubits can then be encoded in hyperfine excited states close to each other. The huge polarizability and dipole moment of Rydberg atoms allow them to strongly interact with each other over a distance of a few micrometres. If the atoms are too close to each other, the *Rydberg blockade* effect prevents the two atoms to both occupy Rydberg states [57]. This behaviour is an ideal mechanism for conditional logic, as the state of a first atom dictates the excitation of a second atom.

Among the advantages of using neutral atoms we can find long decoherence times and accurate state manipulation, as well as a promising scalability. Indeed, neutral atoms can be arranged in a large one-, two- or three-dimensional array, and be addressed individually by laser beams for qubit operations with little crosstalk to quantum states of nearby atoms. This also means that the connectivity between the physical qubits can be programmed before every computation, reducing the impact of the limited connectivity. Besides, the neutral atom devices can offer the possibiltiy to implement the Lechner-Hauke-Zoller (LHZ) encoding procedure to reduce any fully connected problem to many plaquettes of 4-local interactions [58].

Rydberg atoms can be used to realize both gate-based [59], [60] and adiabatic [61], [62] quantum computers.

The most prominent companies that are currently exploring the neutral atoms QCs are ParityQC, whose revenues mainly come from licensing the parity LHZ encoding developed by the CEOs of the company [58]; QuEra [63], [64]; and Pasqal [62]. For a review on the problems where neutral atoms devices have been applied, see [65].

Trapped Ions

In this approach, individual ions are trapped using electromagnetic fields, and their internal energy levels serve as the quantum states or qubits. Quantum computers based on trapped ions typically adopt a gate-based computation paradigm, where gates are implemented using laser beams to induce coherent transitions between the energy levels of the ions. In most trapped ion devices, users have the capability to physically arrange ions on a 2D wafer, creating a suitable topology that facilitate specific calculations. A proposal by Cirac and Zoller in 1995 introduced the first implementation scheme for a controlled-NOT quantum gate on trapped ions [66]. Trapped ions systems have since demonstrated remarkable coherence times, extending up to several minutes [67], along with high-fidelity quantum operations. These achievements position trapped ions as a promising candidate for scalable quantum computing. Notable companies actively involved in the development of trapped ion quantum computers include Quantinuum [68], IonQ [69], [70], and Alpine Quantum Technologies [71], which was founded by Zoller himself.

Topological Quantum Computing

Topological quantum computing is a methodology that focuses on the physical manipulation of qubits to enable fault-tolerant computation. It is not necessarily bounded to a specific hardware technology. Topological quantum computing harnesses the unique characteristics of anyons, exotic particles that exhibit fractional quantum statistics, deviating from the familiar fermionic or bosonic statistics. Within the realm of topological quantum computing, quantum information is encoded by leveraging the non-local properties intrinsic to anyons, including their topological charge and braiding behavior. These properties allow for the manipulation and storage of quantum information. Notably, topological quantum computing holds the potential for achieving fault tolerance, as the topological nature of anyons renders them less susceptible to certain types of errors commonly encountered in quantum systems, such as local fluctuations and noise. This inherent resilience against errors positions topological quantum computing as a promising avenue for the development of scalable and fault-tolerant quantum computers. The concept of using non-Abelian anyons as potential qubits was first proposed by Alexei Kitaev in 2003 [72]. Despite the efforts of the quantum team at Microsoft, a world-leading research team in the field [73], the development of topological qubits is still in its infancy. However, a recent breakthrough came from the Quantinuum team, who have achieved the creation of a quantum system based on trapped ions capable of simulating a topological quantum state [74].

1.2 Fundamentals of Adiabatic Quantum Computing

This Section provides an introduction to the fundamental concepts necessary for comprehending the mathematical principles underlying Adiabatic Quantum

Computing (AQC). Following a historical overview, the Ising model for ferromagnetic spin systems is presented in both its classical and quantum forms. Subsequently, the Simulated Annealing process is introduced, along with its quantum counterpart known as Quantum Annealing, which serves as the foundation for the functionality of AQC. The conditions under which Simulated and Quantum Annealing converge to the optimal solution of an optimization problem are discussed. Lastly, a suggestive mapping between classical and quantum mechanical systems is presented, establishing a connection between the two approaches.

If you are looking for an additional resource to introduce you to the world of Adiabatic Quantum Computation, you can refer to [75].

1.2.1 History and motivations for Adiabatic Quantum Computing

In 1989, Apolloni, Carvalho, and de Falco introduced a combinatorial optimization procedure based on the physical idea of using the quantum tunnel effect to allow the search of global minima of a function of many Boolean variables [76]. The optimization problem was mapped to the Schröedinger Hamiltonian of a quantum spin 1/2 system, while the exploration of the local minima was enforced by a general kinetic term of the form $\frac{1}{2}v^2\frac{d^2}{d^2x}$. They showed from a mathematical point of view that following the state of the system with a low kinetic energy was an efficient method to look for the solution of the optimization problem. In a subsequent companion paper [77], they provided a more efficient numerical implementation and coined the term "Quantum Annealing" to refer to the new approach. In 1994, Finnila et al. introduce the term Quantum Annealing (QA) for the first time [78]. In their paper, which somewhat lacks a convincing mathematical structure, they simulate the same approach that Apolloni, Varvalho and de Falco had previously introduced, namely a quantum system where the kinetic term is slowly decreased to find the global minimum. Few years later, it was the turn of Kadowaki and Nishimori to study a modification to classical simulated annealing (SA) where quantum fluctuations are introduced [79]. In this seminal paper, they tested the approach on an Ising spin system [80], where a transverse field was controlled in an analogous way to the temperature schedule in SA. They also called this procedure Quantum Annealing (QA), and they showed that QA had a higher probability to find the ground state when compared to SA on spin-glass systems. The advantage was supported by previous results showing that spin-glass models in a transverse field can tunnel through high energy barriers (if they are narrow enough) [81], contrary to classical techniques. Shortly after, some research groups began to empirically observe the predicted QA behaviour in real $LiHo_{0.44}Y_{0.56}F_4$ Ising ferromagnets [82], [83].

In 1999, the company D-Wave Systems was founded in Canada. The first white paper published by the company was written by one of its founders, Zagoskin [84], and made no reference to QA. In fact, it aimed at representing qubits with a particular type of superconducting circuits based on Josephson junctions[3], called *d-wave superconductors.* Meanwhile, Farhi et al. published the seminal paper [85], where the authors first discussed the possibility of solving combinatorial problems by "adiabatic quantum evolution", exploiting the adiabatic algorithm[4]. The technique presented in the aforementioned paper will later be named Adiabatic Quantum Computing (AQC). It prescribes to start in the ground state of a Hamiltonian H_B to then gradually mix it with the *problem Hamiltonian* H_P, that encodes the solution of a combinatorial problem in its ground state. While slowly reducing the value of H_B and increasing H_P, the system will keep its state close to the ground state of the istantaneous mixed Hamiltonian, thanks to the adiabatic theorem. At the end of the process, the system is expected to be in the ground state of H_P, and a measure operation can thus output the solution of the combinatorial problem. In 2001, Childs, Fahri and Preskill explained why an AQC protocol can be more robust than other QCs to thermal noises [86]. They showed with numerical simulations that AQC is resilient to thermal noises when the gap between the ground state and the first excited state is wide. Whenever this is not the case, AQCs can be run faster to avoid the asymptotic thermalization of the quantum system. In this case the hypothesis of slow evolution at the basis of the adiabatic theorem could be violated, but the thermal decoherence can counterintuitively improve the performance of the procedure, since the primary effect of decoherence at low temperature is to drive transitions towards the ground state.

Then, in 2004, Kaminsky et al. presented a scalable architecture for AQC [87]. Similarly to what Kadowaki and Nishimori had done in their prototypal QA approach, Kaminsky et al. applied the AQC concept to an Ising model. This was a clever choice, because calculating the ground state of an antiferromagnetically coupled Ising model in a uniform magnetic field is isomorphic to solving the graph theory problem Maximum Independent Set, which is the problem of finding for a graph $G = (V, E)$ the largest subset S of the vertices

[3]See Section 1.3.1.
[4]See Section 1.2.6

V such that no two members of S are joined by an edge from E [88]. This popular graph problem is known to be NP-complete, which means that the architecture proposed by Kaminsky et al. could tackle any NP problem. The Ising model was cleverly mapped by the authors on a lattice of qubits realized via Superconducting Interference Devices[5] (SQUIDs), where the user is allowed to tune both the external magnetic field and the coupling between the qubits. The paper also introduces the fundamental concept of *embedding*, namely the idea to couple two or more qubits with a strong ferromagnetic coupling so that they behave as a single logical qubit, enhancing the connectivity of the device[6].

Inspired by such revolutionary results, D-Wave Systems financed a research project that in 2005 resulted in the publication of the paper *Possible implementation of adiabatic quantum algorithm with superconducting flux qubits* [89]. In this work the authors discuss a possible implementation of adiabatic quantum annealing on superconducting flux qubits, and they show how to solve small instances of MAXCUT using three of such qubits. In that same paper, the authors state that "The idea of quantum computation by adiabatic evolution is very simple but, surprisingly, was discovered only recently", referencing to the two previously mentioned seminal papers by Farhi et al. [85] and Kaminsky et al. [87].

In 2011, D-Wave Systems secured the first historical agreement to sell a quantum computer [90] to global security firm Lockheed Martin. The QC was an AQC device that allowed users to solve Quadratic Unconstrained Binary Optimization[7] (QUBO) problems comprising up to 128 variables. Few months earlier than this announcement D-Wave Systems had published a long-awaited paper that for the first time demonstrated the actual presence of quantum effects inside their devices [91]. The paper partially mitigated the controversy arised after a demonstration held in February 2007 at Silicon Valley's Computer History Museum [92], where D-Wave utilized a 16-qubit AQC to solve a simple sudoku problem. At the time, the scientific community concluded that the solution was obtained mainly thanks to classical thermal effects, such as in SA. Several articles from this period posed the question whether AQCs were actually able to close the gap from the global optimum in hard optimization problems [93]–[95]. Anyway, the 2011 paper proving the D-Wave machines "quantumness" was later backed up by several experiments [96]–[98]. In addition to this, other experiments observed multi-qubit tunnelling events [99], [100] and phase transitions [101] in D-Wave devices.

[5]See Section 1.3.2.
[6]See Section 1.2.10.
[7]See Section 2.1.1.

The D-Wave machine sold to Lockheed Martin was fundamentally different from the (few) competing quantum computing devices. First, it had a much higher number of qubits. IBM had published the first results regarding a prototype of gate-based QC comprising 7 qubits as early as 2001 [102], but this first version was implemented using nuclear magnetic resonance (NMR), while the company later shifted to superconducting qubits. In 2017, IBM unveiled on of its first functioning gate-based QCs, which contained 17 qubits. D-Wave Systems managed to stay far ahead in the qubit count race thanks to its AQC approach. Another important difference is that AQC is not universal, because many quantum algorithms have no mapping on AQC devices. In 2004, Aharonov et al. provided a computational procedure to simulate a gate-based QC using an Adiabatic Quantum Computation [103]. This demonstrated a fundamental equivalence between AQC and universal QC, despite the theorems proved in the paper do not map to the D-Wave Systems architecture[8]. D-Wave researchers have stated that their strategic decision is "to focus on increasing qubit counts and developing a strong user base, rather than implementing the fully general computational model" [33]. In fact, the D-Wave choice partially avoids some of the main shortcomings of gate-based quantum computing, namely the precision required in preparing the initial state and in manipulating the quantum states [104].

Today, D-Wave Systems leads the industry as a prominent company in the production of AQC technology. They provide remote access to their devices, enabling programmers and researchers to develop early-stage quantum applications. Modern AQC prototypes have demonstrated the ability to tackle optimization problems involving thousands of variables, which often pose significant challenges for classical solvers. Besides solving industrial problems, AQC devices have been exploited to study interesting physical phenomena, since they constitute a laboratory-grade implementation of a quantum spin glass model. As an example, in [105] an example of the Kosterlitz-Thouless phase transition was simulated in a D-Wave AQC whose qubits where arranged in a frustrated lattice. On the other hand, despite encouraging results suggesting that recent D-Wave devices exhibit quantum behaviour [106], [107], a solid methodology to prove it is still subject to debate [108].

The following Sections introduce the modern concept of AQC from a mathematical point of view. To understand how AQC can be applied to solve industrially and mathematically relevant problems, jump to Section 2.1.

[8]See Section 1.2.8 to learn under which circumstances AQC can be considered equivalent to gate-based QC

1.2.2 Spin-glass Hamiltonian for AQC

This Section introduces the classical and quantum formulation of an Ising spin-glass model, which stands at the basis of AQC's functioning.

Consider a d-dimensional lattice Λ, where each site i is occupied by a spin variable s_i. The term spin variable refers to a two-state system. The Ising model is a statistical system whose behavior depends on the following Hamiltonian:

$$H = J \sum_{\langle ij \rangle} s_i s_j + h \sum_i s_i \ , \tag{1.2}$$

where s_i is a random spin variable that assumes values ± 1 on the N sites of the lattice, and the first summation runs on sites i and j such that the two sites are nearest neighbours. The Ising model was named after Ernst Ising, which was the first to study and characterize it in 1925 [80].

The Ising model can represent several different physical systems. For instance, consider a system of magnetic dipoles aligned along the same axis, with only two allowed orientations. In this case, the first term in Eq. 1.2 is a two-sites interaction which can produce an ordered ferromagnetic state (if $J < 0$). The second term represents the paramagnetic effect of a uniform external field. We may also consider a binary alloy of type AB. In this case, the spin variables indicate whether a certain site on the crystalline lattice is occupied by an atom of either type A or type B. Neighbors of the same type will contribute to the total energy with a term $+J$ due to reciprocal repulsion, while neighbors of different types will contribute with $-J$. Lastly, one could use the spin values to represent the presence $(+1)$ or absence (-1) of a molecule in a certain cell of a *lattice gas* (a useful model for modelling the critical behavior of a fluid system).

The Ising model can be generalized by allowing each spin-pair appearing in the first summation of Eq. 1.2 to interact with different values of J_{ij}. In the magnetic analogy, it means abandoning the hypothesis that spins are equally distributed across the lattice. Thus, certain couples will be more closely packed, interacting with higher values of the coupling J_{ij}. The model obtained is known as *spin-glass.*

We may also remove the finite range of the interaction, allowing distant spins to interact with each other. It means that the summation over the nearest neighbors appearing in Eq. 1.2 is now performed over any spin pair. If the quadratic couplings are distributed according to a Gaussian probability density, the system we have obtained is then called the *Sherrington and Kirkpatrick model of spin glasses* [109].

We can push the generalization further by considering a scenario where the external magnetic field is not uniform and can vary from site to site. This leads to the following Hamiltonian:

$$H = \sum_{i,j,i\neq j} J_{ij} s_i s_j + \sum_i h_i s_i \ , \tag{1.3}$$

where J and h now depend on the lattice site considered, and the summation is performed over any pair (i,j) such that $i \neq j$.

We can now abandon the idea that the spin-glass model is defined over a lattice. Since interactions have infinite range, there is no need to think of spin variables as bounded in a certain spatial location. It is far more useful to think of the model as a graph, where each spin variable possesses connections with many (potentially all) other spin variables. A coupling $J_{ij} = 0$ corresponds to a missing link in the graph. Nonetheless, the expression *site* will still be used in the present work, meaning a particular position in the graph.

We will call *spin configuration* any function

$$S : \Lambda \to \{\pm 1\} \tag{1.4}$$

which assigns spin up (+1) or down (-1) to each site in Λ. Let Ω be the set of all possible spin configurations. The cardinality of the set is $|\Omega| = 2^N$.

1.2.3 Spin-glass model in the canonical ensemble

We will now study how the spin-glass model described by the Hamiltonian in Eq. 1.3 behave in the canonical ensemble. This means that the system is immersed in a heat bath at a fixed temperature T. Suppose each spin variable is randomly fixed to an up or down configuration, and we refer to this collection of fixed spin variables as S^*. Only the spin variable in site j is let free to change orientation. It follows from statistical mechanics that

$$P_\beta(s_j = 1) = \frac{e^{-\beta H(\{S^*, s_j=1\})}}{e^{-\beta H(\{S^*, s_j=1\})} + e^{-\beta H(\{S^*, s_j=-1\})}} \ , \tag{1.5}$$

where $\beta = (k_B T)^{-1}$ is called *inverse temperature*, defined as the reciprocal of the product of the *Boltzmann constant* k_B and the temperature T. It follows that

$$P_\beta(s_j = 1) = \sigma(-\beta \Delta_j H) \ , \tag{1.6}$$

where $\Delta_j H = [H(\{S^*, s_j = 1\}) - H(\{S^*, s_j = -1\})]$, and $\sigma(x) = (1 + e^{-x})^{-1}$ is the sigmoid function.

If we let every spin variable free to change, we obtain an ensemble of spin configurations S described by the following Boltzmann distribution:

$$P_\beta(S) = \frac{e^{-\beta H(S)}}{Z(\beta)} \ , \tag{1.7}$$

where $Z(\beta)$ is the *partition function* of the system at inverse temperature β:

$$Z(\beta) = \sum_{S \in \Omega} e^{-\beta H(S)} \tag{1.8}$$

The partition function Z ensures that the expression in Eq. 1.7 is a properly normalized probability distribution.

As can be seen in Eq. 1.7, the lower the energy of a spin configuration, the higher its probability. Let S^1, S^2 be two distinct spin configurations with different energies. The ratio of their probabilities is:

$$\frac{P_\beta(S^1)}{P_\beta(S^2)} = e^{-\beta(H(S^1)-H(S^2))} \ . \tag{1.9}$$

The above ratio depends on the temperature and the energy difference between configurations. At high temperatures, this dependence on the energy difference becomes less relevant. For $T \to \infty$ ($\beta \to 0$), the Boltzmann distribution in Eq. 1.7 becomes a uniform distribution over Ω, and the ratio in Eq. 1.9 equals 1 for any pair of configurations. This is not surprising in the magnetic analogy, since thermal fluctuations tend to disrupt the effects of the magnetic interaction between spins. On the other hand, low temperatures make lower energy states exponentially more likely. Let us call E_0 the energy of the ground state. For $T \to 0$ ($\beta \to \infty$), the Boltzmann distribution becomes:

$$\begin{aligned} P_\infty(S) &= 1 \quad \text{if} \quad H(S) = E_0 \\ P_\infty(S) &= 0 \quad \text{if} \quad H(S) > E_0 \ , \end{aligned} \tag{1.10}$$

which means only the ground state is accessible for the system.

1.2.4 Quantum spin-glass model

A quantum analog of the spin-glass model can be defined by mapping each spin variable s_i to a two-state quantum system q_i:

$$\begin{aligned} s_i = +1 &\to |\psi\rangle_i = |{+1}\rangle_i \\ s_i = -1 &\to |\psi\rangle_i = |{-1}\rangle_i \end{aligned} \tag{1.11}$$

The system Hamiltonian can be redefined as follows:

$$H = \sum_{i \neq j} J_{ij} \sigma_i^z \sigma_j^z + \sum_i h_i \sigma_i^z \ , \tag{1.12}$$

where σ_i^z are Pauli-z matrices such that.

$$\begin{aligned} \sigma_i^z \left|\psi\right\rangle_{j \neq i} &= 0 \\ \sigma_i^z \left|+1\right\rangle_i &= \left|+1\right\rangle_i \\ \sigma_i^z \left|-1\right\rangle_i &= -\left|-1\right\rangle_i \ . \end{aligned} \tag{1.13}$$

In the Hamiltonian in Eq. 1.12, all the interaction terms act along the z-axis. For this reason, the eigenstates of the Hamiltonian correspond to the set Ω of the classical states of the classical spin-glass model. This means that if the system is initialized with a configuration in Ω, it does not evolve as time passes, i.e. there is no chance for the spins to flip. If we want to build a true analog of the classical spin-glass model, we should allow the configurations in Ω to relax and become a superposition of multiple classical states. In the classical spin-glass model spin-flip events are possible thanks to thermal effects. In the quantum system, one may introduce a new interaction term that acts along a different direction, like a transverse magnetic field that acts along the x-axis:

$$H = \sum_{i,j,i \neq j} J_{ij} \sigma_i^z \sigma_j^z + \sum_i h_i \sigma_i^z + g \sum_i \sigma_i^x \ , \tag{1.14}$$

with $g \in \mathbb{R}$. The last term in the above expression allows the mixing of the eigenstates of the Hamiltonian in Eq. 1.12. In a suggestive classical analogy, the quantum superposition of different configurations may corresponds to the statistical ensemble, while the transverse magnetic field emulates the presence of a non-zero temperature.

The effect of the transverse field depends on the magnitude of g. If $g \to 0$, the effect vanishes and the eigenstates become increasingly similar to the classical states. If $g \to \infty$, the effect amplifies and the total Hamiltonian H in Eq. 1.14 becomes $H \sim g \sum_i \sigma_i^x$. In this case, the eigenstates of the Hamiltonian correspond to the eigenstates of σ^x, which means each spin has an equal probability to be in the $+1$ or -1 state.

1.2.5 Simulated annealing

Simulated Annealing (SA) is an algorithmic technique that can be used to find the global minima of a cost function. The cost function to be minimized is identified with the energy of a statistical-mechanical system. The system is then given a control parameter T, called *temperature*, which is initially set to a high value, then it is slowly decreased. As $T \to 0$, the system is driven to the state corresponding to the lowest energy, which is the solution of the optimization problem. SA is named after the annealing technique in metallurgy, which involves slowly cooling a metal to increase its hardness.

This Section introduces the concept of Simulated Annealing, stating the hypotheses that are at the basis of its mechanism. Then, arguments concerning the speed of convergence of the method are presented. The main reference for this Section is Ref. [110].

The basic elements of SA are the following:

1. A finite set of possible configurations S.
2. A real-valued cost function F defined on S. Let S^* be the set of global minima of the function F.
3. For each $i \in S$, a set $S(i) \subset S - i$, called the set of neighbors of i.
4. For every $i \in S$, a collection of positive coefficients q_{ij}, $j \in S$, such that $\sum_{j \in S(i)} = 1$. It is assumed that $j \in S(i)$ if and only if $i \in S(j)$.
5. A nonincreasing function $T : N \to (0, \inf)$, called te *cooling schedule*. Here N is the set of positive integers, and $T(t)$ is called the *temperature* at time t
6. An initial state $x(0) \in S$

Given the above elements, the SA algorithm consists of a discrete-time Markov chain $x(t)$ which evolves as follows:

- Choose a neighbour j of the state i at random, where i is the current state $x(t)$. The probability that any particular j is selected is equal to q_{ij}.
- The state $x(t+1)$ is chosen equal to j with a probability $P = \min\left[1, e^{-(F(j)-F(i))/T(t)}\right]$. Otherwise, $x(t+1) = i$.
- Repeat the procedure for any following time step t'.

From the above procedure it follows that

$$P[x(t+1) = j \mid x(t) = i] = q_{ij} e^{-\frac{1}{T(t)} \max\{0, F(j)-F(i)\}}\,, \tag{1.15}$$

if $j \neq i$, $j \in S(i)$.

SA algorithm is best understood by considering a homogeneous Markov chain (the transition probability P in Eq. 1.15 does not depend on time) in which the temperature $T(t)$ is held at constant value T. Let us assume that such Markov chain $x_T(t)$ is irreducible (each state is accessible from any other state[9]), aperiodic (it is false that the probability for the chain to return in a state i is non-zero only for steps that are multiple of an integer k) and that $q_{ij} = q_{ji}$ for all i, j. Then $x_T(t)$ is a *reversible* Markov chain, and its invariant probability is given by

$$\pi_T(i) = \frac{1}{Z_T} e^{-\frac{F(i)}{T}} \ , \ \ i \in S \ , \tag{1.16}$$

where Z_T is a normalizing constant.

From Eq. 1.16 it follows that, as $T \to 0$ the probability is concentrated on the set S^* of global minima of F.

The probability distribution in Eq. 1.16 is the canonical (Boltzmann) distribution. Its important role in statistical mechanics makes it relevant to devise methods to generate sample elements of S drawn according to π_T. A strategy to produce such samples consists in simulating the Markov chain $x_T(t)$ until it reaches equilibrium, and this method is known as the Metropolis algorithm [111].

In the optimization contest, low energy elements of S can be produced with high probability if we produce random samples according to $\pi_T(t)$ with T small. Unfortunately, when T is very small the Markov process requires a great time to reach equilibrium. For this reason, the SA algorithm exploits an inhomogeneous Markov process which shortens the time needed to reach optimal samples by slowly decreasing temperature.

Convergence conditions for Simulated Annealing

It is interesting to know under which conditions SA converges to the global minima. We say SA achieves convergence when:

$$\lim_{t \to \inf} P[x(t) \in S^*] = 1 \ . \tag{1.17}$$

Note that the method converges in probability, which means that for any time t_f there is a non-zero probability for the chain to produce a state which is not a global minimum. In 1988, Hajek described a sufficient and necessary condition

[9]A state j is said to be accessible from a state i if there is a number of steps $n > 0$ such that a chain started in state i has a non-zero probability to reach j after n steps.

on the cooling schedule for SA to converge in probability to the set of global minima of the cost function [112].

Theorem 1.2.1 (Hajek, 1988)**.** *We say that state i communicates with S^* at height h if there exist a path in S (where each element is a neighbour of the previous one) that starts at i and ends at some element of S^*, and such that the target value of F along the path is $F(i) + h$. Let d^* be the smallest number such that every $i \in S$ communicates with S^* at height d^*. Then, the SA algorithm converges if and only if* $\lim_{t \to \inf} T(t) = 0$ *and*

$$\sum_{t=1}^{\inf} e^{-d^*/T(t)} = \inf \ . \tag{1.18}$$

The constant d^* is a measure of the difficulty for the Markov chain to escape from a local minimum and go from a nonoptimal state to S^*. We can suppose that $d^* > 0$ always, since $d^* = 0$ holds only if the problem does not have sub-optimal local minima (in which case the solution could be found by applying a naive gradient descent method). If the Markov chain falls into a local minimum, the SA algorithm makes an infinite number of trials to escape from it, and the probability of success at each trial is of the order of $e^{-d^*/T(t)}$. Then condition in Eq. 1.18 amounts to saying that an infinite number of these trials will be successful.

Following the result of Theorem 1.2.1, many popular cooling schedules are of the form

$$T(t) = \frac{d}{\log(\alpha t + 1)} \ , \tag{1.19}$$

where d and α are positive constants. It is easy to see that such a cooling schedule respects the conditions of the theorem if and only if $d \geq d^*$. In 1984 Geman and Geman [113] proved for the first time that SA converges with this schedule.

Respecting the conditions for the convergence of SA it is not sufficient to make SA useful. We also need to know the speed of convergence. As stated in [110], it can be shown that for any schedule $T(t) = d/\log(\alpha t + 1)$, and for all t,

$$\max_{x(0)} P[x(t) \notin S^* \mid x(0)] \leq A/t^a \ , \tag{1.20}$$

where A and a are positive constants depending on the function F and the neighborhood structure. If we wish $x(t)$ to be outside S^* with probability less than ε, we need $t \geq (A/\varepsilon)^{1/a}$.

In many real scenarios the selected schedule for SA is too fast to ensure statistical guarantee of finding an optimal solution, due to the need to find a solution while respecting wall time limitations. Sometimes the term Simulated Quenching is used to define an abrupt acceleration of SA towards the end of the annealing schedule [114].

1.2.6 Quantum Annealing

This Section presents and describes *Quantum Annealing* (QA), a different approach to solve combinatorial optimization problems that extend simulated annealing to quantum systems. After an introduction to the concept, some mathematical foundations of QA are presented.

If you are interested in further information regarding Quantum Annealing, beyond what is presented in this Thesis, you can consult [115].

A notation disclaimer: QA vs AQC

As anticipated in Section 1.2.1, the terms Quantum Annealing (QA) and Adiabatic Quantum Computing (AQC) have a different historical origin but their meanings can overlap. QA was introduced by Kadowaki and Nishomori 1998 as an approach to solve optimization problems via quantum tunnelling, by tuning the time-dependent transverse field of an Ising model in the same way as the temperature is tuned in SA [79]. AQC was introduced by Fahri et al. in 2000 [85] as an optimization technique that explicitly exploited the adiabatic theorem. Based on these seminal papers, AQC can be seen as a generalization of the QA concept, since Fahri et al. did not limit the discussion on the sole Ising model.

The present Section is thus named after QA since the main result is a convergence condition that applies to the Ising model Hamiltonian (the mathematical derivation is inspired by the 2008 paper by Morita and Nishimori [116]).

Introduction to Quantum Annealing

Optimization problems consist in finding the global minima of a certain cost function, which can also been seen as an energy functional. In SA, we make use of thermal fluctuations to let the system overcome energy barriers standing between the actual state and the ground state for the energy functional. Temperature is then slowly lowered so that the system eventually converges to the set of global minima.

It is interesting to wonder if such a paradigm can be extended to a quantum mechanical system. Quantum fluctuations can be considered as the tendency of the quantum system to explore the phase space of the classical configurations. Setting a high amplitude for such quantum fluctuations makes it easier for the system to explore the phase space. Then this tendency could be suppressed to force the system to the ground state. In this way, we are mimicking the effects of temperature in SA.

An algorithm that controls a quantum system by implementing a schedule where quantum fluctuations amplitude is gradually reduced is called a Quantum Annealing (QA) algorithm. The physical idea underlying such a procedure is to keep the system close to the instantaneous ground state of the quantum system, analogously to the quasi-equilibrium state to be kept during the time evolution of SA. In QA, quantum tunneling between different states replaces thermal hopping in SA.

Similarly to SA, QA is a generic algorithm applicable, in principle, to any combinatorial optimization problem and is used as a method to reach an approximate solution within a given finite amount of time. Although SA is usually considered a useful and effective method for solving such problems, some experimental evidence suggests that QA can outperform SA in certain cases. The performances of SA vs QA are discussed thoroughly in Section 2.1.

One major drawback of QA is that a full practical implementation should rely on a quantum computer since time-dependent Schrödinger equations with a very large scale have to be solved. Quantum computers are still at an early stage of development, and only small-size problems can be effectively solved. Nonetheless, QA theory suggests that future quantum devices could tackle problems considered difficult for SA methods.

In recent years, adiabatic quantum computers have undergone a fast development. Such devices, presented in detail in Section 1.3, are a physical realization of the QA concept and constitute one prominent paradigm of quantum computation.

Adiabatic theorem and convergence conditions of Quantum Annealing

In this Section, the principles of QA are introduced in mathematical terms. The main references for this Section are [116] and [117]. First, the proof of the adiabatic theorem is reviewed. Then, the spin-glass model with a transverse field is introduced as an example of QA implementation. Finally, the convergence condition for QA is presented.

Adiabatic theorem – Let us consider a general Hamiltonian which depends on time t only through the dimensionless time $s = t/\tau$, where τ is a characteristic time scale of the system:.

$$H(t) = \widetilde{H}\left(\frac{t}{\tau}\right) \equiv \widetilde{H}(s) \ . \tag{1.21}$$

The parameter τ is introduced to control the rate of change in the Hamiltonian. By varying the dimensionless time s, we can analyze how the system evolves and responds to the changes in the Hamiltonian in a more manageable way. In quantum systems the state vector $|\phi(t)\rangle$ follows the real-time Schrödinger equation,

$$i\frac{d}{dt}|\phi(t)\rangle = H(t)|\phi(t)\rangle \,, \tag{1.22}$$

where we set $\hbar = 1$. In terms of the dimensionless time we get:

$$i\frac{d}{ds}\left|\widetilde{\phi}(s)\right\rangle = \tau\widetilde{H}(s)\left|\widetilde{\phi}(s)\right\rangle \ . \tag{1.23}$$

We assume that the initial state of the system at $s = 0$ is chosen to be the ground state of the initial Hamiltonian $\widetilde{H}(0) = H(0)$ and that the ground state of $\widetilde{H}(s)$ is not degenerate for $s \geq 0$.

We are interested in obtaining a sufficient condition that allows to evolve the $\widetilde{H}(s)$ in time in such a way that, at any s, the state of the system corresponds to the istantaneous eigenstate $|0(s)\rangle$ corresponding to the lowest eigenvalue at that time. This is usually called *adiabatic evolution*. We will now present the adiabatic theorem (allegedly formulated for the first time by Born and Fock in 1928 [118]) which provides a sufficient condition to adiabatically evolve the system. To keep track of the distance between the state vector and the ground state, it is natural to expand the state vector by the instantaneous eigenstates of $\widetilde{H}(s)$.

First, we derive useful formulas for the eigenstates. The kth instantaneous eigenstate of $\widetilde{H}(s)$ is denoted as $|k(s)\rangle$:

$$\widetilde{H}(s)|k(s)\rangle = \varepsilon_k(s)|k(s)\rangle \ , \tag{1.24}$$

where $\varepsilon_k(s)$ is the eigenvalue of the state $|k(s)\rangle$ with respect to the Hamiltonian $\widetilde{H}(s)$. We assume that $|0(s)\rangle$ is the ground state of $\widetilde{H}(s)$ and that the eigenstates are orthonormal, which means that $\langle j(s)|k(s)\rangle = \delta_{jk}$. Let us differentiate Eq. 1.24 with respect to s, and then project the obtained expression onto the state

$|j(s)\rangle$. We obtain

$$\left\langle j(s)\left|\frac{d}{ds}\right|k(s)\right\rangle = \frac{-1}{\varepsilon_j(s)-\varepsilon_k(s)}\left\langle j(s)\left|\frac{d\widetilde{H}(s)}{ds}\right|k(s)\right\rangle \, , \tag{1.25}$$

if $j \neq k$. In the case $j = k$ we can impose the following condition:

$$\left\langle k(s)\left|\frac{d}{ds}\right|k(s)\right\rangle = 0 \, . \tag{1.26}$$

Note that, if $j = k$, the calculation that brings from Eq. 1.24 to Eq. 1.25 does not produce any condition on the term appearing in Eq. 1.26. Nonetheless, we can demonstrate that the condition in Eq. 1.26 is always achievable by a time-dependent phase shift. Indeed, if we define $\left|\widetilde{k}(s)\right\rangle = e^{i\theta(s)}\,|k(s)\rangle$, we find

$$\left\langle \widetilde{k}(s)\left|\frac{d}{ds}\right|\widetilde{k}(s)\right\rangle = i\frac{d\theta}{ds} + \left\langle k(s)\left|\frac{d}{ds}\right|k(s)\right\rangle \, . \tag{1.27}$$

The second term on the right hand side is purely imaginary, because

$$\left\langle k(s)\left|\frac{d}{ds}\right|k(s)\right\rangle^* + \left\langle k(s)\left|\frac{d}{ds}\right|k(s)\right\rangle = \frac{d}{ds}\,\langle k(s)|k(s)\rangle = 0 \, . \tag{1.28}$$

Then, a proper tuning of the phase factor $\theta(s)$ suffices at making the sum of the right-handed terms of Eq. 1.27 equal to zero. Thus, condition Eq. 1.26 holds for the phase-tuned eigenstate $\left|\widetilde{k}(s)\right\rangle$ even if the original eigenstate $|k(s)\rangle$ does not satisfy it.

The following theorem holds [116]:

Theorem 1.2.2. *If the instantaneous ground state of the Hamiltonian $\widetilde{H}(s)$ is not degenerate for $s \geq 0$ and the initial state is the ground state at $s = 0$, i.e. $\left|\widetilde{\psi}(0)\right\rangle = |0(0)\rangle$, the state vector $\left|\widetilde{\psi}(s)\right\rangle$ has the asymptotic form in the limit of large τ as*

$$\left|\widetilde{\psi}(s)\right\rangle = \sum_j c_j(s)e^{-i\tau\phi_j(s)}\,|j(s)\rangle \ , \tag{1.29}$$

where

$$c_0(s) \approx 1 + \mathcal{O}(\tau^{-2}) \, , \tag{1.30}$$

and

$$c_{j\neq 0}(s) \approx \frac{i}{\tau}\left[A_j(0) - e^{i\tau\left[\phi_j(s)-\phi_0(s)\right]}A_j(s)\right] + \mathcal{O}(\tau^{-2}) \, , \tag{1.31}$$

where $\phi_j(s) \equiv \int_0^s ds' \varepsilon_j(s')$, $\Delta_j(s) \equiv \varepsilon_j(s) - \varepsilon_0(s)$, *and*

$$A_j(s) \equiv \frac{1}{\Delta_j(s)^2} \left\langle j(s) \middle| \frac{d\widetilde{H}(s)}{ds} \middle| 0(s) \right\rangle \tag{1.32}$$

We can conclude that the system evolves adiabatically if the right-hand side of 1.31 is much smaller than unity. In such case, at any time s, the system has a low probability to occupy states different from the istantaneous ground state $|0(s)\rangle$. This condition can be rewritten as

$$\tau \gg |A_j(s)| \; . \tag{1.33}$$

Expression 1.33 implies that

Adiabatic evolution is possible when τ is large, which means $\widetilde{H}(s)$ changes slowly in time.

Using the original time variable t, the adiabaticity condition is further rewritten as

$$\frac{1}{\Delta_j(t)^2} \left| \left\langle j(t) \middle| \frac{dH(t)}{dt} \middle| 0(t) \right\rangle \right| = \delta \ll 1 \; , \tag{1.34}$$

which must hold for all times. This is the usual expression of the sufficient condition for adiabatic evolution.

Convergence conditions of quantum annealing – We now derive a condition which guarantees the convergence of QA. The problem consists of finding an anneal schedule (i.e. the time dependence of the control parameters) such that the adiabaticity condition (Eq. 1.34) is satisfied. We consider the transverse-field spin-glass model introduced in Section 1.2.4 since modern QA devices implement this Hamiltonian.

Suppose we want to solve an optimization problem that can be represented as the ground-state search of a spin-glass model of the general form

$$H_{\text{glass}} \equiv -\sum_{i=1}^{N} J_i \sigma_i^z - \sum_{ij} J_{ij} \sigma_i^z \sigma_j^z - \sum_{ijk} J_{ijk} \sigma_i^z \sigma_j^z \sigma_k^z - \ldots \; , \tag{1.35}$$

where the σ_i^z are the Pauli matrices that act along the z-direction. Many combinatorial optimization problems can be written in this form, by mapping binary variables to spin variables.

An important assumption is that Hamiltonian 1.35 is extensive, i.e. proportional to the number of spins N for large N. To realize QA, a fictitious kinetic energy is typically introduced by the time-dependent transverse field

$$H_T \equiv -\sum_{i=1}^{N} \sigma_i^x \ . \tag{1.36}$$

As anticipated in Section 1.2.4, each term σ_i^x enables spin flips, quantum fluctuations, or quantum tunnelling between the states that possess eigenvalues $+1$ and -1 with respect to σ_i^z. Such effects allow a quantum search of the phase space. The total Hamiltonian takes the expression

$$\begin{aligned} H(t) &= -F(t)\left(\sum_{i,j} J_{ij}\sigma_i^z\sigma_j^z + \sum_i h_i\sigma_i^z\right) - G(t)\sum_i \sigma_i^x \\ &\equiv F(t)H_P + G(t)H_T \ , \end{aligned} \tag{1.37}$$

where t is the physical time, $F(t)$ and $G(t)$ are positive real numbers, H_P is the Hamiltonian whose ground state correspond to the solution of the optimization problem, and $H(t)$ is the transverse field Hamiltonian. The *problem Hamiltonian* H_P in Eq. 1.37 is a simplified version of the more general H_{glass}. The reason for this restriction comes from the fact that modern quantum annealers can only implement Hamiltonians with interaction terms that are at most quadratic. Nonetheless, the following deductions about convergence for QA algorithms hold for a generic H_{glass}.

Each eigenstate of $H(\tau)$ is a set of N binary values $S = \{s_1, s_2 ... s_N\}$, where N is the total number of spin degrees of freedom of the system. Each spin variable s_i can assume two different states, $+1$ and -1. There are 2^N different eigenstates of H_P, one for each possible combination of the N spin variables. A generic state of the system can be expressed as:

$$\begin{aligned} |\phi\rangle &= \sum_{\{s_1,...,s_N\}\in\Omega} \alpha\big(\{s_1,...,s_N\}\big)\,|s_1,...,s_N\rangle \ , \\ \text{with} & \sum_{\{s_1,...,s_N\}\in\Omega} |\alpha\big(\{s_1,...,s_N\}\big)|^2 = 1 \end{aligned} \tag{1.38}$$

where the sums are over each possible configuration of the N spins.

Suppose we are interested in finding which of the eigenstates corresponds to the minimum energy of H_P. By choosing a correct annealing schedule for $F(t)$ and $G(t)$ (i.e. choosing their dependence on time), we can encourage the system to converge to the global minimum. At $t = 0$ we set $G(0) \gg F(0)$. This way, the initial ground state will be an eigenstate of Hamiltonian H_T, which means an equally probable superposition of all the classical states in the phase space Ω. As t grows, the system must be gradually forced into states that are a mixture of low-energy configurations with respect to Hamiltonian H_P, so we must raise $F(t)$ and lower $G(t)$. If the annealing schedule is sufficiently slow, the adiabatic theorem assures that the system will remain close to the lowest energy eigenstate of the instantaneous Hamiltonian $H(t)$. For $t \to \infty$, we impose $G(t) \ll F(t)$, so that the system finds itself in a superposition dominated by the state corresponding to the spin configuration that minimizes H_P.

An important issue is how slowly we should modify $F(t)$ and $G(t)$ to keep the state vector arbitrarily close to the instantaneous ground state of total Hamiltonian 1.37. For simplicity, we suppose to fix $F(t)$ to a positive constant k for every time t. Expression 1.37 can then be rewritten as

$$H'(t) = H_P + \Gamma(t)H_T \tag{1.39}$$

where $H' = H/k$ and $\Gamma(t) = G(t)/k$. Note that modern quantum computers allows to control the time dependency of both F and G terms. Nonetheless, the evolution of the ratio $G(t)/F(t)$ is the driving force of quantum annealing. For this reason, what can be learned for $F(t)$ fixed is instructive and can then be extended to more general contexts.

The following theorem provides a sufficient condition for convergence.

Theorem 1.2.3. *Imposing the adiabaticity condition in Eq. 1.34 on the transverse field spin-glass model in Eq. 1.39 yields the following sufficient condition of convergence for QA:*

$$\Gamma(t) = a(t\delta + c)^{-1/(2N-1)} \tag{1.40}$$

for $t > 0$. Here a and c are constants of order $\mathcal{O}(N^0)$ and δ is a parameter sufficiently small such that the adiabaticity condition in Eq. 1.34 holds.

A proof for this theorem can be found in reference [116].

Computational complexity – The power-law dependence on t in Eq. 1.40, sufficient to ensure convergence for QA, is much faster than the log-inverse law typical of SA, $T(t) = pN/\log(\alpha t + 1)$ (from Eq. 1.19). However, we can not conclude that QA provides an algorithm to solve NP problems in polynomial time. Indeed, suppose we want $\Gamma(t)$ to reach a certain small value ε so that the system is close to the ground state of H_P. The time required to satisfy such condition is estimated from Eq. 1.40 as

$$t_f \approx \frac{1}{\delta}\left(\frac{1}{\varepsilon}\right)^{2N-1} . \tag{1.41}$$

This relation shows that the QA algorithm requires a time exponential in N to converge. In general, ensuring convergence is difficult for those cases where the gap from the global minimum is vanishingly small [93].

Quantum phase transitions in quantum annealing

The adiabaticity condition expressed in Eq. 1.34 can be also written as [119]:

$$\frac{\max\left[\langle 1(t)|\,\frac{dH(t)}{dt}\,|g(t)\rangle\right]}{\min\left[\Delta(t)\right]^2} \ll 1 \, , \tag{1.42}$$

where $\langle g(t)|$ and $\langle 1(t)|$ are the instantaneous ground and first-excited states at time t, respectively, and $\Delta(t)$ denotes the instantaneous minimum energy gap from the ground state. In the previous Section, we interpreted this results as a prescription for the annealing process to be performed slowly. That being said, the condition in Eq. 1.42 also implies that QA works better when $\Delta(t)$ is large [120]. Unfortunately, $\min\left[\Delta(t)\right]$ tends to decrease when the size of the system increases. In particular, $\min\left[\Delta(t)\right]$ vanishes in correspondence of a quantum phase transition (QPT). At the beginning of the quantum annealing process of an Ising system with longitudinal and transverse fields (look at the Hamiltonian defined in Eq. 1.37) the system is in the ground state of the transverse field, which implies an expected magnetization $\langle \sigma^z | \sigma^z \rangle = 0$ for each spin. This means the system is in a disordered state. On the other hand, at the end of the annealing, each spin has a fixed magnetization $+1$ or -1 in the ground state, therefore the state exhibits order. This implies the system encounters a QPT during QA, where the gap from the ground state is expected to be null in the thermodinamic limit, which implies a degenerate ground state. At finite sizes, the scaling of the minimum energy gap at the critical point follows a polynomial law $\Delta_c \sim L^{-z}$, where L denotes the linear size of the system and the critical exponent z characterizes the associated quantum critical point [121]. Therefore, in order to respect the adiabaticity condition, the time required for the QA process scales polynomially with the system size. Unfortunately, what we just discussed holds only in the case of a continuous transition. In general, we must expect the spin-glass Ising system to undergo a discontinuous QPT which corresponds to the change of the ground state from one state to another, without their energies ever become degenerate. In an approximate picture, we can estimate that the minimum gap is attained at two times the tunnelling energy between the two lowest-lying states, which in general decays exponentially as the system size increases [122]. This means that a discontinuous QPT is expected to hinder QA. Consequently, several ways to avoid the discontinuous QPT have been proposed. To learn more on the topic, read the great introduction to Quantum Annealing written by Rajak et al. [122].

1.2.7 Convergence condition of SA and quantum adiabaticity

We now study the convergence condition of SA to be compared with QA. An approach to this problem has been presented in Section 1.2.5 which made use of the inhomogeneous Markov chains as in Ref. [113]. Following such an approach we already presented a cooling schedule that ensures convergence in the limit $t \to \infty$:

$$T(t) = \frac{pN}{\log(\alpha t + 1)} \tag{1.43}$$

It may be surprising to note that it exists a classical-quantum mapping that allows obtaining the same cooling schedule as a consequence of the quantum adiabaticity condition. There is indeed a strong relationship between the quasi-equilibrium condition for SA in a classical system and the adiabaticity condition in the corresponding quantum system.

The description of the classical-quantum mapping and the following conclusions follow Refs. [123] and [116] respectively.

Classical-quantum mapping

Mappings between quantum and classical systems often recur to the introduction of an extra imaginary temporal dimension (sometimes called Trotter dimension) to the quantum system. The mapping discussed in the present Section makes use of a different approach, which allows expressing the thermal expectation value of a classical system in terms of the ground-state expectation value of a corresponding quantum system without extra dimensions.

Suppose that we want to minimize the value of a classical Hamiltonian H that can be written as a spin system as in Eq. 1.35. Consider a classical physical quantity Q that depends on the set of values $\{s_i\} = \{s_1, s_2, ..., s_N\}$ for the N spin degrees of freedom of the system. The thermal expectation value of such quantity $Q(\{s_i\})$ is

$$\langle Q \rangle_T = \frac{1}{Z(T)} \sum_{\{s_i\}} e^{-\beta H} Q(\{s_i\}) \ , \tag{1.44}$$

where the sum runs over all configurations of spins, i.e., over the combinations of eigenvalues $s_i = \pm 1$ of the Pauli matrices σ_i^z.

To proceed further the following theorem is an important element.

Theorem 1.2.4. *Thermal expectation value 1.44 is equal to the expectation value of Q with respect to the quantum wave function*

$$|\phi(T)\rangle = e^{-\beta H/2} \sum_{\{s\}} |\{\sigma_i\}\rangle \ , \tag{1.45}$$

where $|\{s_i\}\rangle$ is the basis of the system composed of states that are simultaneous eigenstates of all σ_i^z. The sum runs over all such possible assignments.

Assume $T > 0$. Wave function 1.45 is the ground state of the quantum Hamiltonian

$$H_q(T) = -\chi \sum_j H_q^j(T) \equiv -\chi \sum_j (\sigma_x^j - e^{\beta H_j}) \ , \tag{1.46}$$

where H_j is the sum of the terms of the longitudinal field Ising Hamiltonian in Eq. 1.35 involving site j:

$$H_j = -J_j \sigma_j^z - \sum_k J_{jk} \sigma_j^z \sigma_k^z - \sum_{kl} J_{jkl} \sigma_j^z \sigma_k^z \sigma_l^z - \dots \tag{1.47}$$

and the coefficient χ in 1.46 is defined by $\chi = e^{-\beta p}$ with $p = \max_j |H_j|$.

The first half of the theorem is trivial, since

$$\frac{\langle \psi(T)|\, Q \,|\psi(T)\rangle}{\langle \psi(T)|\psi(T)\rangle} = \frac{1}{Z(T)} \sum_{\{s_i\}} e^{-\beta H} \langle \{s_i\}|\, Q \,|\{s_i\}\rangle = \langle Q \rangle_T \ . \tag{1.48}$$

To show the second half, first note that

$$\sigma_x^j \sum_{\{s_i\}} |s_i\rangle = \sum_{\{s_i\}} |s_i\rangle \ , \tag{1.49}$$

since in the states summation above each possible spin orientation is considered, and $(|s_j = +1\rangle + |s_j = -1\rangle)$ is an (un-normalized) eigenstate for operator σ_x^j

with unitary eigenvalue. It is also easy to see that

$$\sigma_x^j e^{-\beta H/2} e^{\beta H_j} e^{-\beta H/2} \sigma_x^j \,, \tag{1.50}$$

because

$$\sigma_x^j e^{-\beta H/2} \sigma_x^j = e^{-\beta(H-H_j)/2} \sigma_x^j e^{-\beta H_j/2} \sigma_x^j = e^{-\beta(H-H_j)/2} e^{\beta H_j/2} = e^{\beta H_j} e^{-\beta H/2} \,, \tag{1.51}$$

as both H and H_j are diagonal in the present representation and $H - H_j$ does not include σ_j^z, so $[H - H_j, \sigma_j^x] = 0$. Instead, $\{H_j, \sigma_x^z\} = 0$, since each term in H_j contains σ_j^z one time. We therefore have

$$H_q^j(T) \, |\psi(T)\rangle = (\sigma_x^j - e^{\beta H_j}) e^{-\beta H/2} \sum_{\{s_i\}} |\{s_i\}\rangle = 0 \,. \tag{1.52}$$

Thus $|\psi(T)\rangle$ is an eigenstate of $H_q(T)$ with eigenvalue 0. In the present representation, the nonvanishing off-diagonal elements of $H_q(T)$ are all negative and the coefficients of $|\psi(T)\rangle$ are all positive. Then $|\psi(T)\rangle$ is the unique ground state of $H_q(T)$, according to the Perron-Frobenius theorem [124]. This proves the second and last part of theorem 1.2.4.

An important observation is that, in the high-temperature limit, the quantum Hamiltonian that appears in the theorem is composed just of the transverse term, since the longitudinal term appears multiplied by the inverse temperature β (Eq. 1.46):

$$\lim_{T\to\infty} H_q(T) = -\sum_j (\sigma_x^j - 1) \,. \tag{1.53}$$

It follows that in this limit the ground-state wave function $|\psi(T)\rangle$ corresponds to a summation where each state of the basis appears with equal weight.

The low-temperature limit has, in contrast, the purely classical Hamiltonian

$$\lim_{T\to\infty} H_q(T) = \chi \sum_j e^{\beta H_j} \,. \tag{1.54}$$

In this limit the ground state of H_q corresponds to the ground state of the Hamiltonian of the classical system H.

From the two limits, we see that thermal fluctuations in the original classical systems are mapped to the quantum fluctuations. A decrease in thermal fluctuations in SA is mapped to the decrease in quantum fluctuations for the quantum system.

Convergence condition of SA from adiabaticity

The correspondence found is used to analyze the condition for quasi-equilibrium in classical SA using the adiabaticity condition for the quantum system.

Theorem 1.2.5. *The adiabaticity conditon for the quantum system of $H_q(T)$ yileds the time dependence of $T(t)$ as*

$$T(t) = \frac{pN}{\log(\alpha t + 1)} \tag{1.55}$$

in the limit of large N. The coefficient α is exponentially small in N.

The above theorem can be proved by making use of the following three lemmas. The first one is not demonstrated here, but the proof can be found in Ref. [116].

Lemma 1.2.6. *The energy gap $\Delta(T)$ of $H_q(T)$ between the ground state and the first excited state is bounded below as*

$$\Delta(T) \geq a\sqrt{N} e^{-(\beta p + c)N} , \tag{1.56}$$

where a and c are N-independent positive constants in the asymptotic limit of large N.

Lemma 1.2.7. *The matrix element of the derivative of $H_q(T)$ satisfies*

$$\langle \psi_1(T) | \, \partial_T H_q(T) \, | \psi(T) \rangle = -\frac{\Delta(T) \langle \psi_1(T) | \, H \, | \psi(T) \rangle}{2 k_B T^2} \tag{1.57}$$

where $\psi_1(T)$ is the normalized first excited state of $H_q(T)$.

Proof. By differentiating the identity

$$H_q(T) \, | \psi(T) \rangle = 0 , \tag{1.58}$$

we find

$$\left(\frac{\partial}{\partial T} H_q(T) \right) | \psi(T) \rangle = -H_q(T) \frac{\partial}{\partial T} | \psi(T) \rangle = H_q(T) \left(-\frac{1}{2 k_B T^2} H \right) | \psi(T) \rangle \tag{1.59}$$

The lemma is easily proved if we note that, since the ground state of $H_q(T)$ is zero, we can write $H_q(T) \, | \psi_1(T) \rangle = \Delta(T) \, | \psi_1(T) \rangle$. □

Lemma 1.2.8. *The matrix element of H satisfies*

$$| \langle \psi_1(T) | \, H \, | \psi(T) \rangle \leq pN \sqrt{Z(T)} \tag{1.60}$$

Proof. There are N terms in $H = \sum_j H_j$, each of which is of norm of at most p. The factor $\sqrt{Z(T)}$ appears from normalization of $|\psi(T)\rangle$. $\square$

We can now prove Theorem 1.2.5. The condition of adiabaticity (Eq. 1.34) for the quantum system $H_q(T)$ reads

$$\frac{1}{\Delta(T)^2\sqrt{(Z)}}\left|\langle\psi_1(T)|\,\partial_T H_q(T)\,|\psi(T)\rangle\,\frac{dT}{dt}\right| = \delta\ , \tag{1.61}$$

with sufficiently small δ. If we rewrite the matrix element by Lemma 1.2.7, the left-hand side is

$$\frac{|\langle\psi_1(T)|\,H\,|\psi(T)\rangle|}{2k_BT^2\Delta(T)\sqrt{Z(T)}}\left|\frac{dT}{dt}\right| . \tag{1.62}$$

By replacing the numerator by its bound (Lemma 1.2.8), we have

$$\frac{pN}{2k_BT^2\Delta(T)}\left|\frac{dT}{dt}\right| = \widetilde{\delta} \ll 1 \tag{1.63}$$

as a sufficient condition fo adiabaticity. The statement of Theorem 1.2.5 can now be proved using the bound of Lemma 1.2.6 and integrating the differential equation 1.63 for $T(t)$ (noticing that $dT/dt < 0$).

1.2.8 Computational equivalence between Adiabatic Quantum Computing and Gate-Based Quantum Computing

To narrate how the discussion about how the universality of AQCs evolved in time, we need to introduce the probabilistic analogue of the NP class in the quantum setting, which is the QMA (*Quantum Arthur-Merlin games*) class. The QMA class was introduced by Watrous in [125], and the name was inspired by the classical class MA, which is the randomized analogue of NP [126]. QMA it is the class of all languages that can be probabilistically verified by a quantum verifier in polynomial time. Proving that a computational padaradigm can solve efficiently a QMA complete problem is equivalent to prove it can solve efficiently any problem in QMA.

Kitaev defined the quantum analogue of the classical SAT problem, which he called the k-LOCAL HAMILTONIAN problem [127]. A k-local Hamiltonian contains a set of constraints involving at most k qubits. In the k-LOCAL HAMILTONIAN problem we are asked to determine whether the ground state energy of a given k-local Hamiltonian is below one given threshold or above another. AQC is precisely the framework in which we can tackle and solve the k-LOCAL HAMILTONIAN problem in a native way, by QA. Therefore, if a particular instance of the k-LOCAL HAMILTONIAN is QMA-complete, it means that a system capable of implementing QA of a k-local Hamiltonian can solve any problem in QMA. Kitaev proved that the 5-LOCAL HAMILTONIAN problem is QMA complete [127]. Later, Kempe and Regev showed that even 3-LOCAL HAMILTONIAN is QMA complete [128].

As anticipated in Section 1.2.1, the first (rather complex) proof of equivalence between AQC and universal QC was presented by Aharonov et al. in a 2004 paper that can now be found published in its most recent 2008 version [103]. The showed how to simulate a gate-based quantum algorithm by implementing Hamiltonian terms that served as "clocks" for counting time during the adiabatic process. The total Hamiltonian required nearest neighbour two-body interactions between quantum particles possessing six distinct accessible states. Proving the universality of a paradigm is a stronger result than proving it can solve QMA-complete problems. It is therefore correct to say that an AQC system based on Aharonov's Hamiltonian can also solve QMA-complete problems.

Further results followed regarding QMA completeness. It can indeed be proved that MAX-k-SAT[10] is a special case of the k-LOCAL HAMILTONIAN problem. One can represent the n binary variables involved in the MAX-k-SAT as n qubits, and represent each clause with a diagonal Hamiltonian containing a k-qubits interaction that assign a higher energy to the combination of values forbidden by the MAX-SAT clause. Therefore, the lowest eigenvalue of the sum of the Hamiltonians corresponds to the maximum number of clauses that can be satisfied simultaneously. Since MAX-2-SAT is known to be NP-complete, we can conclude that the 2-LOCAL HAMILTONIAN problem is at least as hard as any problem in NP, which means it is NP-hard. This does not imply any conclusion regarding QMA completeness, but in 2004 Kempe et al. proved that the 2-LOCAL HAMILTONIAN problem is indeed also QMA-complete [129]. In the same paper, they showed that adiabatic computation with 2-local Hamiltonians is equivalent to quantum computation in the circuit model. This result is interesting because modern AQC devices are limited to 2-local Hamiltonians. Anyway, the result by Kempe et al. requires in general all-to-all connectivity. Nonetheless, a year later, Oliveira and Terhal generalized the result about QMA completeness, showing that the 2-LOCAL HAMILTONIAN problem remains QMA-complete even if the Hamiltonians are restricted

[10]MAX-k-SAT is a reduction of the well-known MAX-SAT problem where each clause has exactly k variables. For a brief introduction to SAT and MAX-SAT problems, see Section 2.2.1.

to nearest neighbor interactions between qubits on a two-dimensional grid [130]. Leaving universality aside for a moment, this result imply that AQCs can solve any problem in QMA using a spin glass system equipped with local, sparse, two-terms interactions. Unfortunately, there is still an operative difference from real-world AQCs. Indeed, D-Wave Systems devices implement only a particular type of 2-local interaction, which is $\sigma^z\sigma^z$. The results by Kempe and then Oliveira and Terhal are not specialized for this single type of interaction. In 2007, Biamonte and Love [131] showed that AQCs based on spin glasses could be made universal by inserting a new two-local interaction of the form $\sigma^x\sigma^x$ or $\sigma^x\sigma^z$, besides the $\sigma^z\sigma^z$ usual term. D-Wave tested this possibility in 2019 [132] but no commercially available solver with such 2-local interactions have been unveiled up to now (2024).

1.2.9 Simulated Quantum Annealing

Earlier than the first physical implementations, researchers implemented simulations of the QA algorithm. Simulations have been useful and remain useful today to understand QA at a theoretical and numerical level. In addition to that, we will see how Simulated Quantum Annealing (SQA) has proved more efficient than SA in several cases, thus constituting a viable classical approach to solve optimization problems (see also Section 2.6.2 for empirical results).

The usual approach to simulate QA is a Quantum Monte Carlo approach [133]. There are two main options, mainly using a (low) finite temperature approach or a zero-temperature method. The zero-temperature transfer-matrix Monte Carlo [134] and the Green's function Monte Carlo [135] belong to the zero-temperature methods, but they suffer severely from different drawbacks, which renders them much slower than finite-temperature algorithms in practice. The finite temperature approach is usually implemented via the Path Integral Monte Carlo (PIMC). This method maps the partition function of a d-dimensional quantum Hamiltonian H onto that of an effective classical Hamiltonian in $d+1$ dimensions. Then, the classical Hamiltonian is simulated at a fixed low temperature so that thermal fluctuations are present but limited. Quantum fluctuations are gradually reduced during the "annealing" process in $d+1$ dimensions by reducing the kinetic term of the classical Hamiltonian [135]. PIMC is thus used to simulate the equilibrium behaviour of a system at finite temperature T. This is a limitation, since in the quantum annealing context we would like to follow the low-lying states (ideally, the ground state) of the time-dependent Hamiltonian of the system [134]. It is also clear that this evolution is not the correct integration of the Schrödinger equation, most obviously because the system is not able to explore the whole configuration space as a quantum wave function could do. Nonetheless, this is a really useful approximation, since an integration of the time-dependent Schrödinger equation, that would give the exact time evolution of the system, is almost always infeasible, except for extremely small systems. As an example, in 1998, Kadowaki and Nishimori [79] solved the time-dependent Schrödinger equation for a small spin glass system of $N = 8$ spins, showing that QA converged to the global minimum with much higher probability than SA in all cases where the same schedule was used for the temperature in SA and the transverse field in QA. In particular, they found that QA approached the ground state with high accuracy for any schedule slower than $\propto 1/\sqrt{t}$.

The PIMC method was used by Martonak et al. in [136] to simulate QA on a two-dimensional random Ising. They provided data supporting previous findings from the same author [137], stating that for both classical and quantum annealing the residual energy after annealing is inversely proportional to a power of the logarithm of the annealing time, but the quantum case has a larger power that makes it faster. They also forecast a *freezing effect*, namely the tendency of the system to converge to a low-temperature excited state at the end of the annealing if the schedule was too fast[11]. Such findings are supported by [138], where SQA achieved better performances and scaling with respect to SA on an instance of the travelling salesman problem implemented as an Ising model. On the other hand, [139] proved that SQA performs worse than SA on k-satisfiability problems. The result shows that, despite the shortcomings of SA, SQA is not straightforwardly better than SA in every application.

The aforementioned research works made use of the spin glass model as a benchmark to test SQA. Anyway, SQA is not limited by a physical hardware, which means it can be used to explore adiabatic quantum evolution in systems different from the spin glass model [140]–[142]. In this case, for the reason presented in Section 1.2.6, the term QA should be substituted by AQC. PIMC has been applied to perform simulations of boson systems in [140], obtaining good agreement between simulations and experimental measurements on superfluid ^{4}He for various physical quantities. In [141], a similar approach is used to the minimization (folding) of a simplified protein model that is know to exhibit frustrated behaviour. In this case the authors chose to perform a simulation with a non-negligible temperature to better exploit both quantum and thermal effects. In [142], simulated AQC is used to obtain the ground-state structure of classical Lennard–Jones clusters, namely the geometrical structure in which

[11]See Section 1.3.6 for more information on the freezing point.

atoms dispose themselves when bonded by Lennard-Jones interactions. In this case the simulated quantum system unexpectedly exhibited ergodicity breaking, not being able to efficiently explore the configurations, which is the main shortcoming expected from SA.

SQA can also be compared to the exact QA algorithm to gain insights of what can we actually expect from a system that is able to exploit quantum tunneling. In [143] the authors tested SQA on a simple optimization problem with a single tall thin energy barrier. They showed that SQA takes polynomial time to solve the problem, while SA takes an exponential time. This is considered by the authors evidence against the prospect of exponential quantum speedup using tunneling, at least in this case, since in [144] it was demonstrated for the same problem that QA was exponentially faster than SA. The results in [143] thus suggest that the benefits of tunneling through energy barriers with adiabatic evolution should not be thought of as an exclusively quantum advantage, since it can also be achieved by a general-purpose classical optimization algorithm. Nonetheless, we can also find situations where SQA takes exponentially longer than the quantum evolution being simulated [145], [146]. In particular, results from [147] suggests that QA has an advantage over PIMC when there are multiple homotopy-inequivalent paths for tunneling. The authors demonstrated that frustration can generate an exponential number of tunneling paths, which under certain conditions can lead to an exponential advantage for incoherent tunneling over classical PIMC escape.

SQA performances can also serve to inspect what physical AQC could achieve. Indeed, one could expect that hardware AQC devices will be much faster in implementing a time-evolution that is so difficult to simulate. An important question arises whether physical QA can beat its simulated analogue. In [107], the authors study the correlation between the probability of success of SQA, SA, and a D-Wave device in finding the ground state of various spin glass instances. Their findings show that D-Wave correlates much better with SQA, which suggests QA is a more accurate model to forecast success probabilities for AQC than SA. Anyway, in [148] the authors show a significantly better scaling for SQA compared to both D-Wave and SA when applied to an optimization problem whose energy landscape had been designed to favor solvers exploiting tunneling effects. This suggest that physical implementations of QA could be limited by non-ideal effects[12].

[12]See Section 1.3.6 to learn more regarding all the effects that make hardware AQC devices differ from the theoretical QA process.

1.2.10 Embedding techniques

In this section we introduce the concept of embedding, a fundamental procedure that allows to solve on AQCs problems that do not possess a direct map to the AQC topology. After a brief introduction, we define the embedding problem and describe the most suitable topologies to realize hardware devices that can embed most of the problem graphs. We then present the detrimental consequences of poor embeddings and the most popular computational methods to perform the embedding. We will also discuss the relevant literature on the topic, while we relegate to Section 3.1.4 the outline of the various techniques that have the scope of enhancing or speed up the computation of a suitable embedding.

Among the various bottlenecks that impede an efficient physical implementation of a QA protocol the main ones are the noise sources inside and outside the device (see Sec. 1.3.6), and the limited number of connections among qubits. At the time of writing (2024) the most highly connected AQC device publicly available is the Advantage2_prototype1.1, which comprises 563 working qubits, each connected at most with 20 other qubits. According to the D-Wave Leap website [149] the chip implements 4790 connections, which implies that, on average, every qubit is connected to $2 \times 4790/563 \sim 17$ other qubits. A fully connected network of 563 qubits would require $563 \times (563 - 1)/2 = 158203$ connections, which means that only roughly 1/33 of the required connections are actually implemented. A poor connectivity implies that most QUBO[13] problems cannot be mapped directly on the QPU. To make an example, in [150] Dumoulin, Goodfellow, Courville, and Bengio tested the Chimera[14] topology (the graph of qubits connections in the D-Wave 2000Q models) to train a Restricted Boltzmann Machine (RBM), an unsupervised learning model which constitutes the most important machine learning application for AQC[15]. They state that the RBM can cope with a reasonable amount of missing connections bewtween its visible and hidden layer, but they also estimate that an RBM with 784 visible units and 784 hidden units would require to set to zero 99% of its weights to be cast on a Chimera graph without resorting to embedding, resulting in dreadful performances. The authors conclude the analysis suggesting that designers of new physical AQC devices should focus their efforts on overcoming the limitations imposed by the topology restrictions.

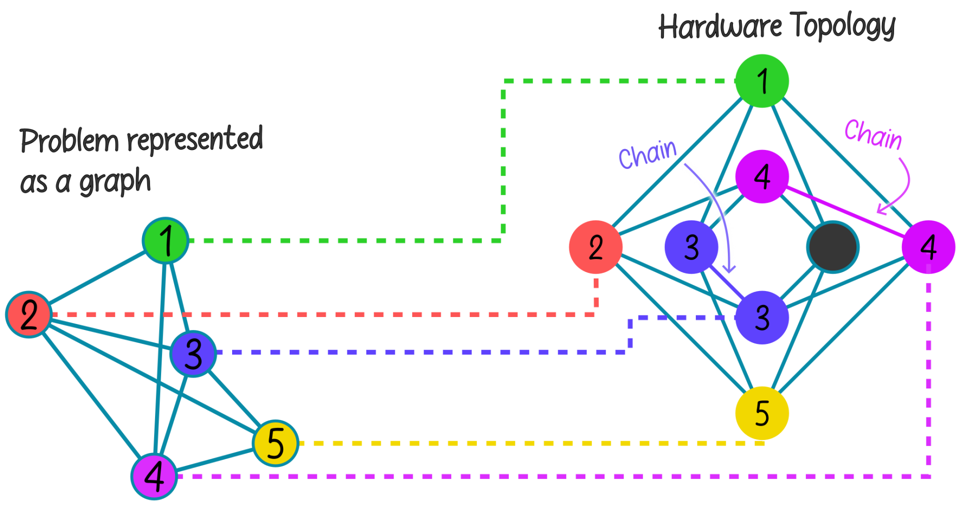

FIGURE 1.1: Depiction of the embedding process of a problem on a $K_{4,4}$ (Chimera) bipartite graph. On the left, the optimization problem we want to solve is represented as a graph where each units stands for a binary variable. Each pair of units is connected by an edge only if the cost function contains a quadratic term where the two units appear multiplied. There is no way to directly map the variables of this sample problem on the qubits in the lattice on the right, since some edge is bound to be missing (you can try and see by yourself). The mapping can instead be performed by recurring to embedding techniques: it is sufficient to represent some of the variables as a chain of qubits, instead of considering single qubits. Qubits in a chain are forced to stay parallel via a strong ferromagnetic coupling, so they can be seen as a single, two-state quantum system (also called a virtual qubit). This way we have artificially enhanced the lattice topology, allowing for all couplings among variables to be faithfully represented.

The minor-embedding

Luckly, there are purely software techniques that allow to enhance the connectivity of AQC. It is sufficient to add redundancy in the QUBO problem formulation by inserting ancilla qubits that have the sole objective of extending the reach of the connections of physical qubits. Suppose we have at our disposal an AQC device where qubits are connected one to the other according to a square lattice topology. We want to use this device to minimize the following simple QUBO formula:

$$C(q_1, q_2, q_3) = q_1 q_2 - q_2 q_3 + q_3 q_1 \tag{1.64}$$

This problem can be represented by a triangular graph where each node represents a Boolean variable. Unfortunately, there is no place in the device topology where three qubits are connected all-to-all. We can modify the formula in Eq. 1.64 by substituting one of the variables with two variables:

$$C(q_1, q_2, q_3^1, q_3^2) = q_1 q_2 - q_2 q_3^1 + q_3^2 q_1 + J_{\text{chain}} q_3^1 q_3^2 \tag{1.65}$$

where $J_{\text{chain}} < 0$ is a real parameter. The formula expressed in Eq. 1.65 can be mapped directly on the QPU topology, since variables are now connected in a square fashion. Additionally, every configuration of the variables in Eq. 1.65 where q_3^1 and q_3^2 are equal correspond to a configuration of Eq. 1.64. If we are able to ensure that $q_3^1 = q_3^2$ always, then we can think of them as two clones of a single Boolean variable. This can be done thanks to the term $-J_{\text{chain}} q_3^1 q_3^2$. If $J_{\text{chain}} \gg 1$, the two qubits corresponding to q_3^1 and q_3^2 will tend to stay parallel (same value). In particular, if $J_{\text{chain}} \gg$ than other coefficients in the QUBO formula, then configurations respecting the constraint $q_3^1 = q_3^2$ will have a wide energy gap from their analogue where the constraint is not realized. The strategy we just adopted to enhance the connectivity of the device is an *embedding technique*. From a physical point of view it consists in coupling qubits with strong ferromagnetic coupling to make them behave as a single two-states quantum system. Whenever two or more qubits are linked together with this purpose, it is customary to refer to them as a *qubits chain*. Figure 1.1 can help understand how this process work on a $K_{4,4}$ (Chimera) bipartite graph. The notation $K_{c,c}$ will be used throughout the text to identify bipartite $c \times c$ graphs (namely, a graph where each one of c nodes is connected to each one of another

[13]See Section 2.1.1.

[14]The Chimera topology is introduced in Section 1.2.10.

[15]See Section 2.4.

group of c nodes). K_c will instead represent a fully-connected graph comprising c nodes.

Our examples involved coupling two qubits to make them behave as a single one. Nonetheless, this same strategy can be in principle applied to an arbitrary number of qubits. We now formally introduce the general concept of *minor-embedding* as presented in [151].

Definition 1. *Let U be a fixed hardware graph. Given a graph G, the minor-embedding of G is defined by*

$$\phi \,:\, G \to U \tag{1.66}$$

such that

- *each vertex in $V(G)$ (the set of vertices in G) is mapped to a connected subtree T_i of U;*
- *there exists a map $\tau : V(G) \times V(G) \to V(U)$ such that, for each $i, j \in E(G)$ (the set of edges in G), there are corresponding $i_{\tau(i,j)} \in V(T_i)$ and $j_{\tau(j,i)} \in V(T_j)$ with $i_{\tau(i,j)} j_{\tau(j,i)} \in E(U)$. In other words, for each edge in G connecting vertices i and j, there are two vertices, one in the subtree corresponding to i and one in the subtree corresponding to j, which are connected by an edge in U. This edge in the hardware graph will represent the edge in G between vertices i and j.*

Given G, if ϕ exists, we say G is embeddable *in U.*

The mapping is named minor embedding since, in graph theory, G is a minor of U. In few words, a G is a minor of a graph U if G can be formed by U by deleting vertices, edges, and contracting edges (deleting an edge by identifying its two vertices as a single vertex). There are two special cases of minor-embedding

- *Subgraph-embedding*: Each T_i consists of a single vertex in U. In other words, G is isomorphic to a subgraph of U, and the minor-embedding is trivial.
- *Topological-minor-embedding*: Each T_i is a chain of vertices in U. In other words, for any T_i, vertices in $V(T_i)$ can be ordered and each of such vertices is connected by an edge only to the previous one and the next one.

The minor-embedding function implemented in the Ocean Software Development Kit [152] (the D-Wave coding package) looks precisely for topological-minor-embedding[16]. Any set $V(T_i)$ where qubits are coupled together according to a topological-minor-embedding is called a *qubit chain*.

We just learned how the embedding process works from a topological perspective. But how do the weights and biases transform after the embeddings? The more adopted and sensible approach is called *uniform spreading* [153], and it prescribes to divide the total value of the bias of any variable equally among the qubits composing the corresponding chain. In the same fashion, a coupling between two logical variables can be divided by the number of couplings existing between the two qubits chains representing such variables. This ensures there is a one-to-one mapping between the states where all chains are aligned and target problem states of the same energy. In fact, this is only true up to a constant offset coming from the ferromagnetic chain couplings, which can always be ignored. The uniform spreading approach has the immediate advantage of reducing the absolute values of both biases and weights. While this is not relevant for optimization purposes, since we can always rescale the values of such parameters, it can have a huge advantage for sampling tasks[17]. When sampling, we usually want the sample distribution to be characterized by a specific effective temperature, so we need to rescale the values of weights and biases in order to achieve the correct temperature. Reducing such parameters raises the temperature arbitrarily, while increasing them reduces the temperature, but only to a finite value. Dealing with smaller weights and biases means we can scale them up by a larger factor, being thus able to simulate lower temperatures.

It is important to underline that the problem of connecting multiple computing units together is a well-known problem in all types of hardware computing technology. In particular, most quantum computing machinery available today has a much sparser connectivity with respect to D-Wave devices. In recent years, competing quantum technologies have focused mainly on increasing the number of qubits, with remarkable results such as the 433-qubits Osprey gate-based quantum computer [22]. This number of qubits is now comparable with D-Wave devices, and it is debatable when, but not if, the topology problem will become the biggest obstacle also for gate-based quantum computation. We are probably getting very close to that turning point.

The next Section will introduce the topologies realized inside D-Wave AQC devices, together with the shortcomings of a poor embedding and the modern

[16]See Section 1.2.10.

[17]See Section 2.3 to learn about this relevant application of AQCs.

approach to find a good embedding. Section 3.1.4 in the next Chapter will instead present the most recent ideas to circumvent and overcome the connectivity limitations of modern AQC devices.

Suitable topologies for Adiabatic Quantum Computing

When building a physical AQC device, the problem of devising an efficient topology to connect the qubits is not trivial. First, the hardware graph must respect some physical constraints [154]. In particular, there is a degree-constraint in that each qubit can have at most a constant number of couplers (edges). This restriction come from the (trivial) observation that any physical connection must take up some space and thus there is a finite limit on the number of realizable connections among qubits. Additionally, the coupler length must be finite (that is, all neighbor qubits are within a bounded distance). In other words, an adiabatic quantum hardware graph is a bounded-degree, edge-length bounded geometric graph. Notice that crossing is allowed (it can be a non-planar graph).

While respecting such constraints, researchers aim to realize an efficient adiabatic quantum hardware topology. A validation technique consists in asking if a proposed topology can implement complete K_n graphs. The notation K_n here represents a graph composed by n nodes and characterized by an all-to-all connectivity, which means each node is connected by an edge to each other node. If the topology of an AQC device can minor-embed any K_n graph, then the device can tackle any problem comprising n (or less) Boolean variables, since any of those problems will be a subgraph of K_n. For example, a square grid satisfies the physical constraints, but it does not admit K_n minor-embedding for $n \geq 5$ because of the Kuratowski's theorem (see for example [155]). Stated more simply, the square grid poses strict limitations to the ability of subtrees T_i to overlap and connect to each other.

Instead of defining topologies and then asking if they can embed K_n graphs, it is more effective to build ab initio a topology suitable to embed those graphs. This is what D-Wave systems did in 2008 when they devised the optimal hardware graph presented in patent [156], which was popularized as TRIAD by Choi [154]. TRIAD is constructed in such a way that one can easily decompose it into a bounded degree graph that satisfies the physical constraints.

Construction of TRIAD – TRIAD maps each vertex of K_n to $n-1$ nodes in the hardware graph, arranged in a triangular shape. The procedure to dispose the chains is somewhat complex to describe by words and it's probably easier to look at Figure 1.2**b**. The chain corresponding to the first node is diagonal going from lower-left to upper-right. Then, each one of the $n-1$ qubits composing the

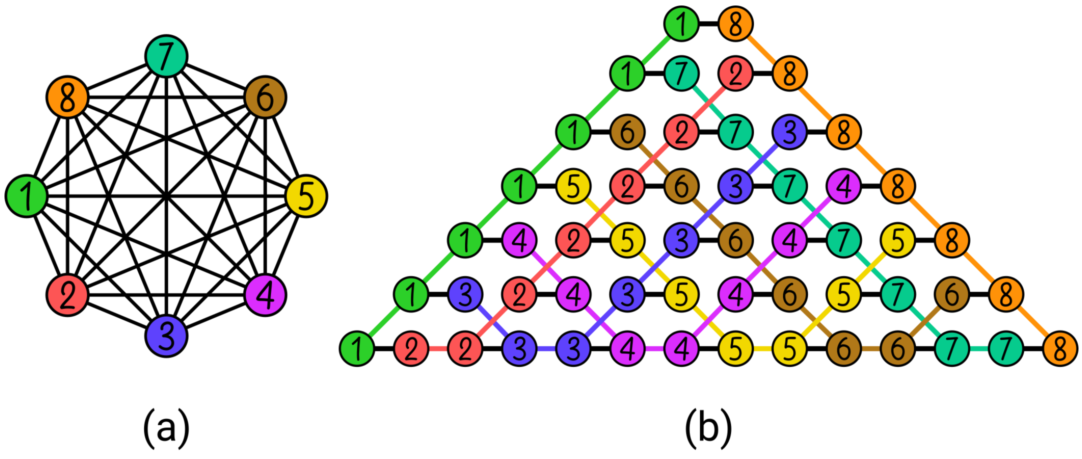

FIGURE 1.2: (a): Graph K_n for $n = 8$. (b): TRIAD construction of an embedding for K_n with $n = 8$. Each chain is composed by $n - 1$ qubits and is connected with a single connections with every other chain. Black chains represent connections between different embedded variables. Image adapted from [154].

chain of the first node is connected on the right to a qubit, each one being the first qubit of the chain of a different one of the $n-1$ remaining nodes to embed. For each "starting qubit", the chain is built by moving towards lower-right until the height of the lower node in the first chain is reached, then a step to the right, and then steps towards the upper-right until $n - 1$ qubits compose the chain. The resulting shape is triangular, every node is represented by a chain of $n - 1$ qubits, and any qubit is connected at most to 3 other qubits. The embedding in Figure 1.2**b** effectively represent the K_8 graph in Figure 1.2**a**.

Decomposition of TRIAD – We now aim to reduce the number of qubits used in the embedding to simplify it and reduce the chain length[18]. Suppose the available degree (the number of allowed couplers) of a physical qubit is $d > 3$. Then, we can reduce the length of chains in the TRIAD embedding by exploiting the higher connectivity. The new embedding will require at least N_{min} qubits for each chain, where:

$$\begin{aligned} N_{\text{min}} &= \text{minimum} x \text{ s.t.} x \cdot (d-2) + 2 > n - 1 \,, \rightarrow \\ \rightarrow N_{\text{min}} &= \lceil \frac{n-3}{d-2} \rceil \end{aligned} \tag{1.67}$$

because for each qubit we have $d - 2$ couplers available, since two are used to connect it to the chain. Terminal qubits have an extra connection available, so a $+2$ is added. The sum of all the available connections with other qubits must exceed $n - 1$ to ensure we can implement all the needed connections.

[18]See Section 1.2.10 to understand why shorter chains work better.

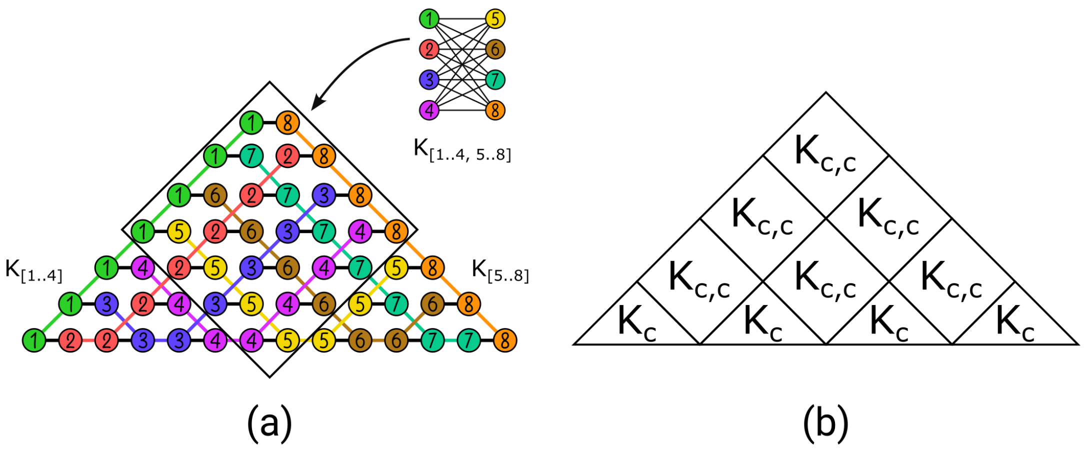

FIGURE 1.3: (a): the black square selects a part of the TRIAD construction that corresponds to the embedding of a $K_{4,4}$ graph, which is also represented without embeddings on the upper right. The two little triangles on the sides of the TRIAD embedding are K_4 fully-connected graphs. You can verify these statements by checking the black lines connecting the units. (b): The decomposition can be extended recursively by composing $K_{c,c}$ bipartite graphs in a square lattice. This means that, by arranging qubits in a square grid of bipartite graphs as the one showed in the figure, we ensure that fully connected graphs can be embedded efficiently in the hardware. Image adapted from [154].

An example of the described reduction can be appreciated in Figure 1.3**a**. Any graph K_n (suppose n divisible by 2) can be seen as a complete bipartite graph $K_{\frac{n}{2},\frac{n}{2}}$ where the two sets of $\frac{n}{2}$ nodes are then equipped with an all-to-all connections. We can make good use of this apparently trivial observation by noting that if the number of available couplers for each qubit is d, we can then embed K_{2d} using a TRIAD decomposition where each variable is represented by two physical qubits, each with $d-1$ connections to other qubits. Indeed, imagine $2d$ qubits disposed in a complete bipartite graph $K_{d,d}$. Every qubit in the graph represent a different variable in the K_{2d} graph to be embedded. Then, for each qubit, connect it to a second qubit to form a two-qubits chain. Now the chains are complete, but the logical qubits are still connected in a bipartite fashion. We can now connect the newly added qubits among themselves into two K_d graphs, saturating the missing connections.

If n is greater or d is smaller, we have to apply this "bipartite" approach in a more extended fashion, as shown in Figure 1.3**b**. As an example, we can decompose a complete graph K_n (assume $n = 2^{ck}$ with k positive integer) using only $K_{c,c}$ ad K_c graphs. Since K_c graphs can be realized using $K_{c,c}$ bipartite graphs[19], we can conclude that an adiabatic quantum hardware implementing

[19]A trivial way to embed K_c on $K_{c,c}$ is to form c two-qubits chains, each comprising a qubit from the first subset and a qubit from the second subset.

a topology where $K_{c,c}$ graphs are connected as in a square grid can embed a K_n complete graph where $n = 2^{ck}$, and k is bounded by the size of the hardware. Being able to embed K_n, it follows that any K_c with $c < n$ can be embedded on the same hardware.

Dwave topologies – Chimera[20] is the name given by D-Wave to the topology implemented in its first commercial AQC devices up to D-Wave 2000Q[157]. In the Chimera topology qubits are arranged in $K_{4,4}$ cells disposed in a square lattice. Each cell thus comprises a bipartite graph of 8 physical qubits, which are also connected to the four nearby cells, resulting in 8 couplers per qubit (4 inside the cell and 4 to connect to other cells) [158]. In 2020, D-Wave Systems released the Advantage series of ACQ devices, the first to implement the Pegasus topology [159]. Pegasus features qubits of degree (number of couplers per qubit) 15 and native $K_{6,6}$ subgraphs. While in Chimera K_3 graphs required an embedding, in Pegasus complete graphs up to K_4 are implemented without resorting to embedding. Thus, contrary to the Chimera case, for the Pegasus topology there are specific embedding techniques that allow to find better embeddings by using a non-topological embedding, where each logical qubit is represented by a K_4 graph of physical qubits [160]. Pegasus is the topology currently used in most of the available AQC devices. A new topology, named Zephyr, is recently being tested on a prototype composed by approximately 500 qubits[161]. The Zephyr topology has degree 20.

D-Wave systems have been proposing new topologies with the aim to increase the degree of the available couplings and thus shorten the embedding or to avoid them at all. This can be useful to improve the quality of the solutions but also to reduce the overhead time required for the embedding procedure, which can be demanding[162]. Despite this often improves the QPUs performances, there are cases in which new topologies lose to older ones, namely in those cases where the problem graph to be embedded is so sparse that the new connections in the improved topologies are not useful[163].

Detrimental consequences of a poor embedding

Whenever an AQC program is implemented using an embedding, there are three main threats that can impede a successful computation:

- Entropy – The phase space grows after the embedding, which means the solution must now be found among a larger number of available configurations, leading to a lower probability of success. Failures due to the

[20]See the upper little graph in Figure 1.3 for a depiction of the Chimera topology, and the right side of Figure 4.16 for the same graph presented with a different layout.

embedding manifest themselves as *chain breaking* events, when one or more qubits in a chain are measured in a different state from the rest of the chain. In the Ocean SDK this scenario is usually managed by *majority voting*, which means the value of the logical qubit represented by the chain is considered equal to the value assumed by most of the qubits in the chain. Longer chains are expected to perform worse [164] and experience chain breaking with a higher probability. See Section 3.1.3 to learn how to mitigate chain breaking.

- Slower exploration of the phase space – To avoid chain breaking, one can increase J_{chain}. If the parameter becomes much higher than the problem weights the exploration of the phase space can slow down due to the reduced probability of spin flip events. Indeed, switching the value of a single qubit in a linear chain requires a shift in the overall energy by $2J_{\text{chain}}$. The introduction of many energy barriers to perform a single spin flip can have detrimental effects on the performance of QA.
- Change in the distribution temperature – We will learn in Section 1.3.6 that the sample outputs obtained by querying a physical AQC are distributed according to a Boltzmann distribution at a given temperature. This is a consequence of the thermal noise inside the device and impedes optimization tasks, but such effect becomes useful in machine learning and sampling applications[21]. The final temperature of the samples is relevant both when solving optimization problems or performing sampling tasks. Indeed, when sampling we need a fixes nonzero temperature, while in optimization we want to reduce the final temperature as much as possible. In both cases, the embedding procedure can cause problems, since longer chains cause the production of hotter samples[22].

During the years, several authors have underlined that these threats coming from the sparse connectivity of AQC devices constitutes some of the most important obstacles to achieving efficient Adiabatic Quantum Computation. In [165], the authors compared AQC performances on a Max-Cut problem (see Sec. 2.2.2) to those of a coherent Ising machine (CIM), which possesses all-to-all connectivity since it is implemented as a software. D-Wave devices outperformed the tested CIM on smaller problems, while highly connected problems caused the AQC device to lose to the CIM for several orders of magnitude. Turning

[21]See Section 2.3.
[22]See Section 1.3.6.

to sampling application, an interesting paper from Marshall, Giochino, and Rieffel [166] shows how, as the embedding size grows, the probability to sample from the desired portion of the phase space can be exponentially suppressed. Using combinatorial arguments they prove that the probability P_0 of obtaining a sample with zero broken chains can be written as

$$P_0 = (1 + e^{2\beta J_{\text{chain}}})^{-(K-1)N}, \tag{1.68}$$

where β is the inverse of the effective temperature at which samples are extracted by the AQC, K is the length of each chain (chains are supposed to be equally-long here), and N is the number of chains in the embedding. We can immediately appreciate that the probability P_0 decays exponentially in problem size and chain size. Employing shorter chains can thus bring huge benefits. At the same time, the subspace is sampled with higher probability if $\beta J_{\text{chain}} < 0$ increases in absolute value, which means we whould aim to reducing the temperature or increasing the ferromagnetic coupling J_{chain}. Increasing J_{chain} is indeed the first strategy one can apply to increase the probability to sample from the correct space. This conclusions apply also to optimization problems, since it is as important in that case to avoid broken chains.

D-Wave Systems itself has published few guidelines suggesting which embeddings to prefer when facing a choice between multiple possible embeddings output by an heuristic [167]. The manual state that embeddings comprising short, uniform-length chains are to be preferred, and that the chain strength has to be properly balanced with respect to the problem range. In [168] D-Wave Systems' researchers present compelling evidence proving that chain length plays an important role in performance. They compare the performances of two AQC devices based on Chimera and Pegasus topologies on the problem of minimizing the energy of three-dimensional spin glasses. They observed improved scaling of solution time and improved consistency over multiple graph embeddings, due to the shorter chain length required to embed the same problem instances on the Pegasus topology.

Heuristic methods to find an embedding

In Section 1.2.10 we defined what is a minor-embedding. In Section 1.2.10 we described how to devise efficient AQC topologies that allow simple embeddings. Now we hypothesize to have at our disposal a good quantum adiabatic hardware topology U, and we need a way to find the best embedding possibile of a given problem graph G on U.

Unfortunately, we have to start this discussion by noting that finding a minor-embedding can be NP-hard [169]–[171]. In [170], Eppstein proved that, given a positive integer h and a graph G, determining if G contains K_h as a minor is NP-complete. If G is fixed, which is usually the case for adiabatic quantum hardware, we can find all minors of G, but then we should still check if a given graph P we want to embed is a subgraph of any of them. This procedure is still NP-complete for arbitrary inputs. With respect to the required qubit resources, in [172] the authors remind that a quadratic scaling of the embedding resources for the spin glass model is expected for any hardware graph with fixed degree. As a particular case, they show that for a Chimera graph a clique of N variables is embedded using $N^2/4+N$ physical qubits. For problems where three-body and higher-order interaction terms have to be included in the Hamiltonian (which is the case in many quantum chemistry problems) the scaling is much worse, typically exponential in the number of variables [173], [174].

The problem of deciding whether a graph H is a minor of G has been know for decades as the H-MINOR CONTAIMENT problem. It was studied extensively and successfully by Robertson and Seymour [175], which were the first to provide exact algorithmic procedures for finding graph minors. Elaborating their findings, Hicks [176] devised an algorithm that decides if a graph G with m edges and branchwidth[23] k contains a fixed graph H on h vertices as a minor in time $\mathcal{O}\big(3^{k^2} \cdot (h+k-1)! \cdot m\big)$. Then Adler et al. [178] found a way to improve the dependency on k obtaining a time $\mathcal{O}\big(3^{(2k+1)\cdot \log k} \cdot h^{2k} \cdot 2^{2h^2} \cdot m\big)$ with their algorithm.

As a consequence of the hardness of H-MINOR CONTAIMENT, in 2014 D-Wave Systems researches changed approach and focused on heuristic approaches to find a feasible embedding in a reasonable amount of time [171]. They defined an embedding algorithm that is efficient if both the hardware topology and the problem graph are sparse. Such algorithm is currently implemented in the primary utility function `find_embedding()` in the `minorminer` package of the D-Wave Ocean SDK[179]. The `minorminer` package is currently the most popular tool to look for minor embeddings of a problem of interest on D-Wave devices.

The "heuristic approach" gained popularity as it allowed to effectively embed problems on the D-Wave topology, that was already commercially available at that time. In [158], the authors introduced an efficient approach to generate large clique minors in subgraphs of a hardware graph (even if they did not study

[23]Branchwidth quantifies how closely a graph can be represented as a collection of disjoint trees. It was introduced for the first time in [177].

the computational complexity of the method). They exploited an iterative tree decomposition technique to realize one of the first embedding heuristics resilient to missing qubits in the hardware. Then, Boothby et al. [180] exploited the TRIAD origin of the Chimera topology to define *triangle embeddings*, native TRIAD reductions on Chimera that have the characteristics of being uniform (each chain has the same number of qubits) and near optimal (chains contain at most on qubit more than the minimum achievable). Using triangle embedding the authors introduced a dynamic programming technique that, given a subgraph of the Chimera topology with missing qubits, finds a maximum-sized native clique embedding in polynomial time. Their approach results in embeddings of uniform chain length, which induce a more predictable dynamics throughout the anneal [172] (see also Section 1.3.6). Zaribafiyan et al. [181] suggest that the path to better or faster embedding algorithms lies in restricting the graph minor embedding problem to specific cases. They indeed support the TRIAD-reductions approach of Boothby et al., but they also underline its limitations, since sparse problems will usually require more qubits if embedded using a strategy based on clique minors. Thus, Zaribafiyan et al. tried to identify a common structure across many problems that could be exploited advantageously. They realized that the graph formulations of many NP-hard optimization problem correspond to the Cartesian product of two graphs (as an example see [172], [182], [183]). In [181] they propose a deterministic and scalable embedding algorithm for embedding the Cartesian product of two complete graphs into D-Wave Systems' Chimera hardware graph. Such algorithm proved faster than existing alternatives at the time and tends to produce equal-length chains. Later, Lobe et al. [184] introduced a mapping from the embedding problem to an integer linear problem to better investigate the embedding of complete graphs on Chimera topologies with missing qubits. Their approach is particularly interesting since they devised an exact embedding method for complete graphs that can beat some of the available heuristic methods, but they also describe an heuristic extension of the algorithm that can tackle larger instances that the exact one.

Pinilla et al. [185] introduced the concept of *layout awareness*: exploiting structural information of the problem graph to enhance the embedding search. They developed the Layout-Aware MinorMiner (LAMM) method, which diffuses the nodes of the QUBO problem to be embedded on the hardware topology layout to achieve even spreading of the candidate chains across the cells of the topology. It then runs `minorminer` from this supposedly advantageous position. Zbinden et al. [186] tested LAMM and `minorminer` against their new algorithms Spring-Based MinorMiner (SPMM) and Clique-Based MinorMiner (CLMM). They showed that CLMM outperform `minorminer` for dense graphs to be embedded on Pegasus, while SPMM outperforms `minorminer` for Chimera embeddings and for sparse embeddings on Pegasus. Additionally, they found out that, while `minorminer` achieves overall good performance on Chimera graphs, on Pegasus it fails to embed even medium-density graphs on 175–180 nodes which are known to have clique embeddings. They thus suggest D-Wave to extend their `minorminer` implementation with SPMM and CLMM, which are both actually based on the original `minorminer`.

See Section 3.1.4 to find further information on how to improve embeddings.

1.3 Hardware implementation of Adiabatic Quantum Computers

The D-Wave QPU is built with a network of radio-frequency superconducting quantum–interference device (rf-SQUID) qubits. A SQUID is a device based on one or more Josephson junctions embedded in a superconducting closed loop. The seminal idea for this approach arised in early 2000s, mainly thanks to [87] and [89]. The first empirical evidence of quantum annealing in such a setup was obtained in 2010 on a system of eight superconducting qubits [157]. The result was followed in 2011 by a seminal paper by Johnson et al. [91] that studied the same eight-qubit system, this time coupling qubits together and providing empirical hints of quantum effects. The same setup has since then evolved up to the modern AQC devices comprising thousands of qubits.

Each qubit and coupler in an AQC device has several controls that are manipulated by individual on-QPU digital-to-analog converters (DACs). Along with the DACs, a small number of analog control lines provide the time-dependent control required by the quantum Hamiltonian:

$$H_{\text{glass}} = F(s)\left(\sum_i h_i\sigma_i^z + \sum_{ij\in\rho} J_{ij}\sigma_i^z\sigma_j^z\right) - G(s)\left(\sum_i \sigma_i^x\right) \tag{1.69}$$

where ρ is the set of available couplers in the QPU graph. To implement quantum annealing, $F(s)$ and $G(s)$ values are slowly modified, following the principles introduced in Section 1.2.6. Typical values for such quantities during anneal are represented in Fig. 1.4.

This Section describes the physical realization of a specific QPU that performs quantum annealing according to Eq. 1.37. First, the Josephson junction

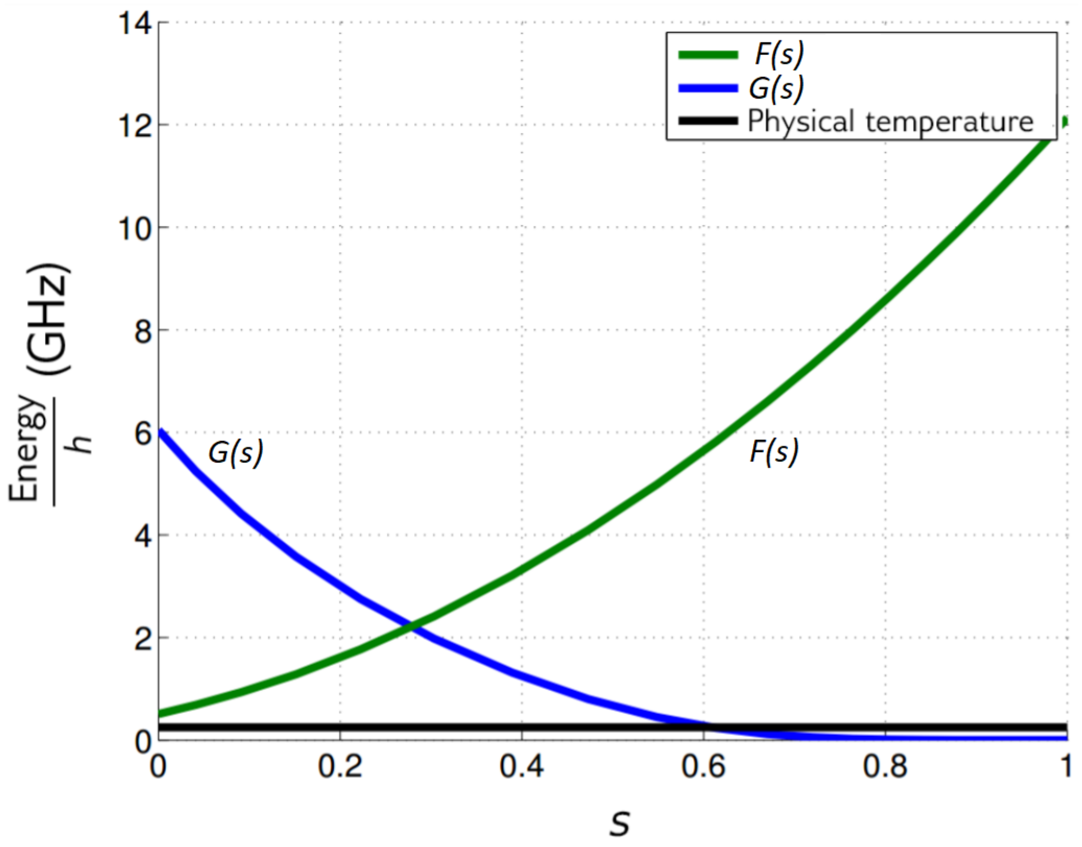

FIGURE 1.4: Annealing functions $F(s)$, $G(s)$. Data shown are representative of D-Wave 2X Systems. Image from Ref. [187].

and the rf-SQUID are presented as the fundamental building blocks of the superconducting qubtis for quantum annealing. Then, we explain how to implement the full annealing Hamiltonian exploiting the presented technology. Next, we describe the undesirable behaviours that arise in hardware implementations of AQC devices, such as integrated control errors, thermal noise, and decoherence. The same Section also details how the output samples distribute after the annealing cycle. The last Section presents alternative hardware approaches to implement AQC which are not based on superconducting qubits.

If you desire to acquire more information than that presented in this Thesis regarding the hardware realization of Adiabatic Quantum Computers, you can start by consulting [188] and [189].

1.3.1 The Josephson junction

When two superconducting metals are put in contact with a thin insulating layer separating them, a zero-voltage superconducting current may arise. This can happen if there is a phase difference between the superconducting wave functions on the two sides of the insulator. The resulting *supercurrent* flows continuously without the need to apply any voltage. Such interesting effect is an example of a macroscopic quantum phenomenon that can happen at macroscopic scales. It is named *Josephson effect* after Brian D. Josephson, who predicted this effect in 1962 [190], and was later awarded with the Nobel prize for this same discovery in 1973 (read his Nobel lecture at [191]). The *Josephson junction* (J junction) is a piece of hardware designed to physically realize the Josephson effect. It is built from two pieces of superconducting metal, separated by a *weak link*, a thin layer of normal metal or some type of insulator. We will now briefly outline the main results of Josephson's research.

When a superconductor is cooled under the critical temperature $T = T_C$, the electrons in the superconductor form pairs, called Cooper pairs. Each Cooper pair is to be considered a boson, and thus all electrons can condense in the ground energy level. It follows that all the electrons can be described by a collective wave-function having a single quantum phase:

$$\Psi(\vec{r}, t) = |\Psi(\vec{r}, t)| e^{i\varphi(\vec{r}, t)} \, . \tag{1.70}$$

Since such macroscopic wave function must be single-valued in going once around a superconducting loop, flux quantization arises. It means that the flux contained in a closed superconducting loop takes values that are multiples of the quantum flux $\Phi_0 = h/2e \approx 2.07 \times 10^{-15}$ Wb. Here h is the Planck's constant and e is the electronic charge.

Josephson predicted the tunneling of Cooper pairs through an insulating barrier separating two superconductors (*weakly coupled superconductors*). He showed that the current I flowing through such a junction is given by

$$I = I_0 \sin \delta \tag{1.71}$$

where $\delta = \phi_1 - \phi_2$ is the difference between the phases ϕ_1 and ϕ_2 of the condensates in the two superconducting electrodes and I_0 is the critical current of the superconductor. The presence of such current is usually referred to as *dc Josephson effect.*

Josephson also described a dynamic property of the junction, known as the *ac Josephson effect.* In the presence of a voltage V between the electrodes, δ evolves with time t according to

$$\frac{\mathrm{d}\delta}{\mathrm{d}t} = \frac{2eV}{\hbar} = \frac{2\pi V}{\Phi_0} \ . \tag{1.72}$$

As the current through a junction is increased from zero, the flow of Cooper pairs constitutes a supercurrent and the voltage across the junction remains zero until the current exceeds the critical current. At higher currents, the phase difference evolves according to Eq. 1.72, and there is a voltage across the junction.

1.3.2 The rf-SQUID

In 1963, only one year after the Josephson effect was theorized, John Rowell and Philip Anderson at Bell Labs realized the first experimental setup that was able to sense the zero-voltage supercurrent in a Josephson junction (J-junction) [192]. On more year later, in 1964, Robert Jaklevic, John J. Lambe, James Mercereau, and Arnold Silver of Ford Research Labs invented the dc-SQUID (direct current Superconducting Quantum Interference Device) [193]. The dc-SQUID implements two J-junctions in parallel, and two terminals to connect the loop to an external circuit. The critical current of a circuit with two J-junctions is a periodic function of the magnetic flux threading the loop, with a period equal to the flux quantum Φ_0. The SQUID can be biased to a critical regime by applying to the loop a proper current. The voltage across the SQUID then varies with increasing external flux with a sensibility of the order of Φ_0 [194]. If coupled with a device capable of amplifying the differences in the critical current of the loop, the dc-SQUID becomes an extremely sensible instrument to measure many different physical quantities, such as temperature, currents, and magnetic fields, reaching a precision of up to 3×10^{-34} J/Hz $\sim 0.5h$ [195].

In 1965, at the same Ford Research Labs where the DC-SQUID was born, Robert Jaklevic, John J. Lambe, Arnold Silver, and James Edward Zimmerman tested a simpler SQUID architecture, the RF-SQUID (radio-frequency SQUID) [48], shown in Figure 1.5. It is constituted by a single J-junction interrupting a superconducting loop, and as the same suggest it is biased by a radio frequency signal. It possesses similar capabilities as the DC-SQUID, since it can be used as a measurement device in simlar ways, but its sensitivity is worse by few orders of magnitude [194]. Nonetheless, the RF-SQUID arised great interest in 2000 after the seminal paper by Friedman et al. [47], in which they empirically demonstrated a quantum superposition of macroscopic quantum states in a compound Josephson junction[24], which is based on both RF and DC SQUIDs. The experiment constituted one of the first examples of a quantum superposition of macroscopic states. Indeed, thesuperconducting circuit used in their experiment had a dimension of few micrometers, several orders of magnitude larger than the atomic scales where this effect are customary. In the following discussion, we will explain how this interesting experimental result regarding RF-SQUIDs can be exploited to manufacture superconducting qubits.

In Figure 1.5 and from now on, the resistive-capacitive shunted junction (RCSJ) model of the J junction is assumed. In this model, the behavior of the J junction is modeled by substituting the junction with a capacity C, a resistance R and a Josephson element having a current according to Eq. 1.71.

The superconducting loop has inductance L, which is coupled to the inductor L_T of a tank circuit via a mutual inductance $M = k(LL_T)^{1/2}$. The tank circuit has a resonant frequency $f_0 = \omega/2\pi$ and has a quality factor[25] $Q_0 = \omega_0 L_T/R_T$ in absence of the SQUID. A radio-frequency (rf) current generator supplies the tank circuit with a current $I_{rf}\cos(\omega_{rf}t)$ so that the resulting rf magnetic flux applied to the SQUID loop is

$$\Phi_{a,rf} = \frac{\Phi_0}{2\pi} \upsilon_{\mathrm{T}} \cos(\omega_{\mathrm{rf}} t + \rho) \ , \tag{1.73}$$

where $(\Phi_0/2\pi)\upsilon_T$ is the amplitude of the external rf flux, ρ is its phase and t is the time.

[24]See Section 1.3.4.

[25]The quality factor is a dimensionless parameter that compares the frequency at which a system oscillates to the rate at which it dissipates its energy [196].

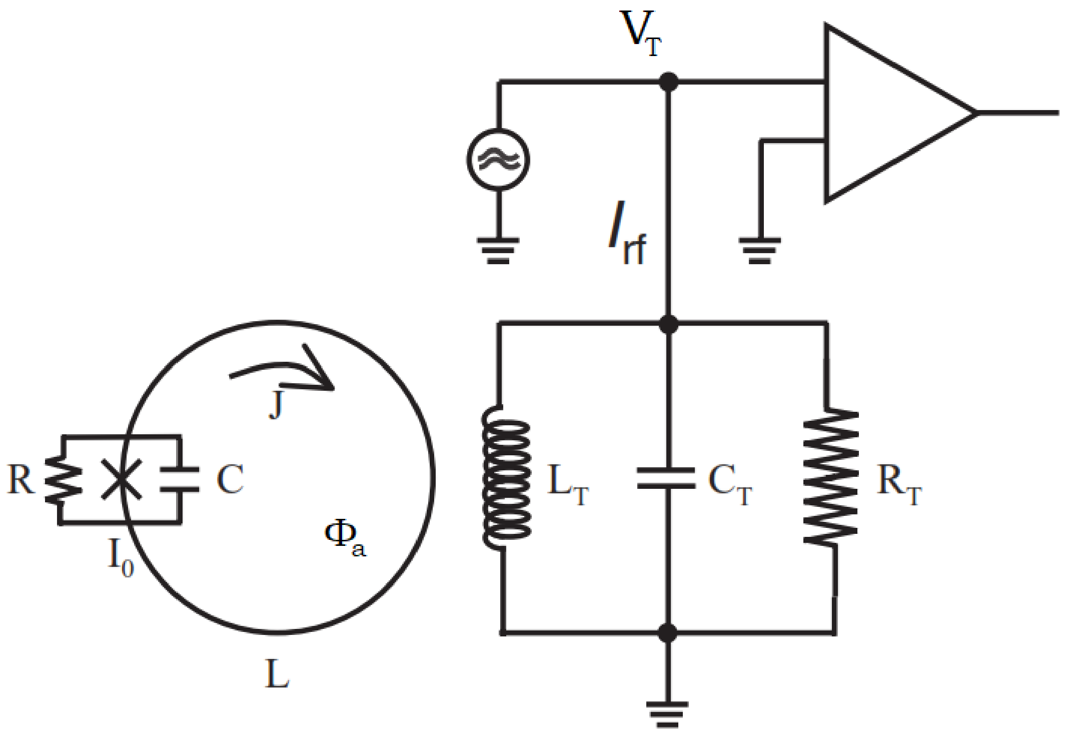

FIGURE 1.5: Schematic representation of the rf SQUID, with tank circuit and preamplifier. The cross symbol corresponds to the J junction. Image adapted from Ref. [48].

The voltage that provides the output signal of the rf-SQUID is picked up from the amplifier A whose input is connected to the tank circuit. The input signal magnetic flux to be measured Φ_a is applied to the SQUID loop. This flux can be thought of as quasistatic since the pumping frequency ω_{rf} is usually far larger than the highest frequency of Φ_a.

We call Φ_T the total flux in the loop. Flux quantization imposes the constraint:

$$\delta + 2\pi\Phi_T/\Phi_0 = 2\pi n \ , \tag{1.74}$$

where n is an integer. The supercurrent J flowing in the loop is determined by the phase difference δ across the junction (Eq. 1.71):

$$J = -I_0 \sin\left(\frac{2\pi\Phi_T}{\Phi_0}\right) \tag{1.75}$$

We then obtain the following relation:

$$\delta = \frac{2\pi}{\Phi_0}\Big(\Phi_{\text{a,rf}} + \Phi_{\text{a}} + LJ\Big) \ , \tag{1.76}$$

since the term in brackets is the sum of any flux in the loop, i.e. Φ_T. We call $\Phi_{\text{a,T}} = \Phi_{\text{a,rf}} + \Phi_{\text{a}}$ the total applied flux, which differs from Φ_T by the supercurrent-flux term.

On the basis of Eq. 1.71 and Eq. 1.76 we can write the SQUID potential [48]. It has two components, the Josephson coupling energy and the inductive energy that is due to the screening current flowing into the SQUID loop:

$$\frac{U_{\text{SQUID}}(\delta)}{U_q} = \frac{\left(\delta - \varphi_{\text{a,T}}\right)^2}{2\beta} - \cos\delta \tag{1.77}$$

where $\varphi_{\text{a,T}} = 2\pi\Phi_{\text{a,T}}/\Phi_0$, and $\beta = 2\pi L I_0/\Phi_0$ is the so called *SQUID hysteresis parameter* (or screening parameter). Here, two different scenarios arise for different values of β. If $\beta < 1$, the potential has one minimum, while for $\beta > 1$ there can be several metastable states. Such behaviour is shown in Fig. 1.6.

1.3.3 The single-qubit potential

If we define $V(\delta) = \beta U_{\text{SQUID}}$, we can write

$$V(\delta) = U_q\left\{\frac{(\delta - \varphi_{a,T})^2}{2} - \beta\cos(\delta)\right\} . \tag{1.78}$$

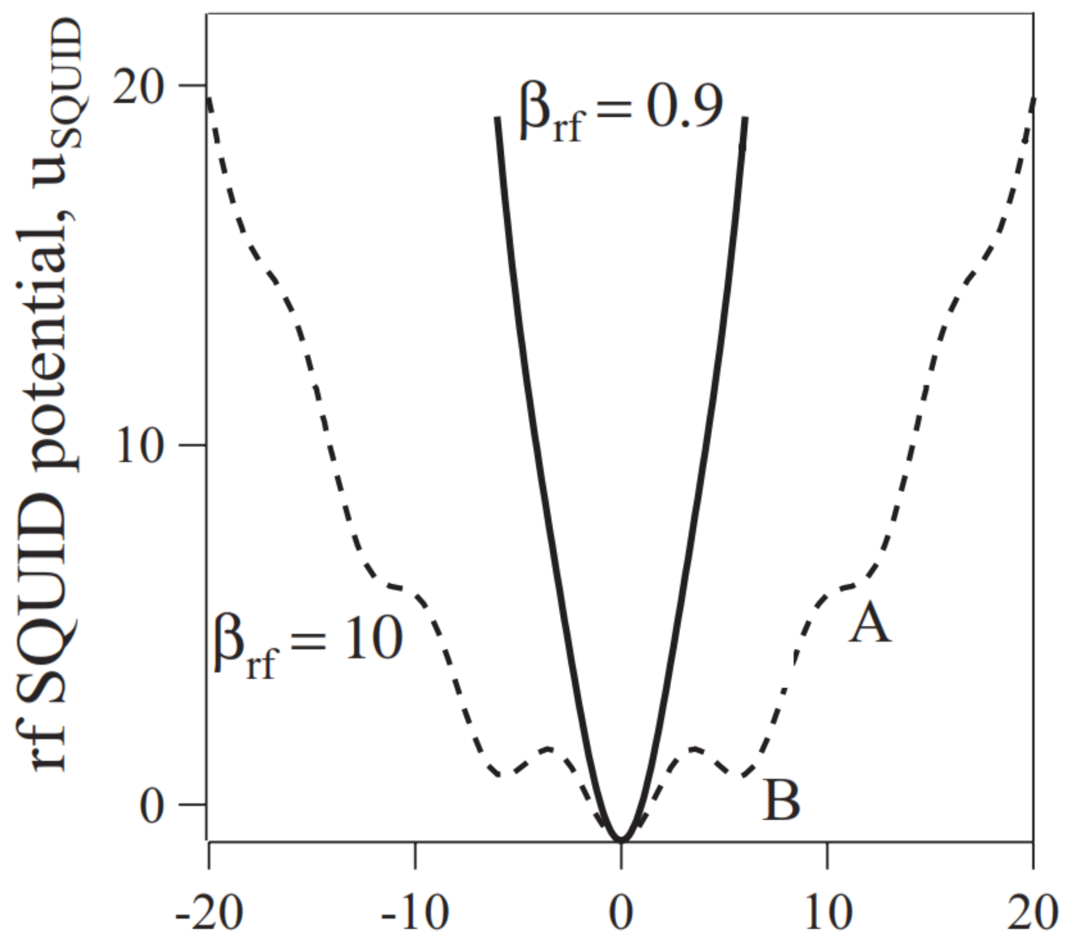

FIGURE 1.6: rf SQUID normalized potential $u_{\text{SQUID}} = U_{\text{SQUID}}/U_q$ as a function of the Josephson phase difference δ (from Eq. 1.77). Image from Ref. [48]

Consider a device designed such that $\beta > 1$. If the flux is biased such that $\varphi_{a,T} \approx \pi$, then the potential energy $V(\delta)$ will be bistable (two distinct potential wells at the same energy). If β is increased, the potential energy barrier between such two local minima of $V(\delta)$ becomes higher. The two lowest-lying states of the rf SQUID may couple via quantum tunneling through the barrier. These two states are separated from all other rf-SQUID states by an energy of the order of the rf-SQUID plasma energy $\hbar\omega_p \equiv \hbar/\sqrt{LC}$.

Figure 1.7 shows the potential energy and the two lowest-energy states of an rf SQUID at degeneracy ($\varphi_{a,T} \approx \pi$, i.e. $\Phi_{\mathrm{a,T}} = \Phi_0/2$). In the Figure, $|g\rangle$ and $|e\rangle$ are the ground and first excited states respectively.

We can consider a low energy approximation of the potential $V(\delta)$ in Eq. 1.78, where we suppose that only the two lowest energy states $|g\rangle$ and $|e\rangle$ are accessible.

In this approximation, we can introduce another possible basis of the system:

$$\begin{aligned} |{\downarrow}\rangle &= (|g\rangle_d + |e\rangle_d)/\sqrt{2} \\ |{\uparrow}\rangle &= (|g\rangle_d - |e\rangle_d)/\sqrt{2}\,, \end{aligned} \tag{1.79}$$

where $|g\rangle_d$ and $|e\rangle_d$ are respectively the ground and first excited state at degeneracy. When $\varphi_{a,T}$ changes, the relation than links the basis $\{|{\uparrow}\rangle, |{\downarrow}\rangle\}$ to the new energy eigenstates changes as well. The states $|{\uparrow}\rangle$ and $|{\downarrow}\rangle$ are roughly Gaussian-shaped wave functions that are centered about each of the wells shown in Fig. 1.7.

In the low energy approximation, the two-state system is described by the following Hamiltonian (Ref. [197]):

$$H_q = \frac{1}{2}\varepsilon\sigma^z - \frac{1}{2}\hbar\Delta_0\sigma^x\,, \tag{1.80}$$

where σ^x and σ^z are Pauli matrices. The states $|g\rangle$ and $|e\rangle$ are now the only two eigenstates of Hamiltonian H_q. If $\varepsilon_0 = 0$, $|g\rangle$ and $|e\rangle$ are also eigenstates of σ^x, and Δ_0 is the difference between their energy eigenvalues. We make this state correspond to the degenerate case depicted in 1.6 (i.e. $\varphi_{\mathrm{a,T}} = \pi$), so that $|g\rangle_d$ and $|e\rangle_d$ are eigenstates of σ^x. It follows that $\{|{\uparrow}\rangle, |{\downarrow}\rangle\}$ is the basis of the eigenstates of σ^z. If $\Delta_0 = 0$, $|{\uparrow}\rangle$ and $|{\downarrow}\rangle$ are degenerate, and the difference between their energy is ε. For any nonzero value of Δ_0, the system can experience quantum tunnelling between the two states.

The properties just described, together with the expression of H_q 1.80, suggest that such a system can be used to implement quantum annealing on a single qubit.

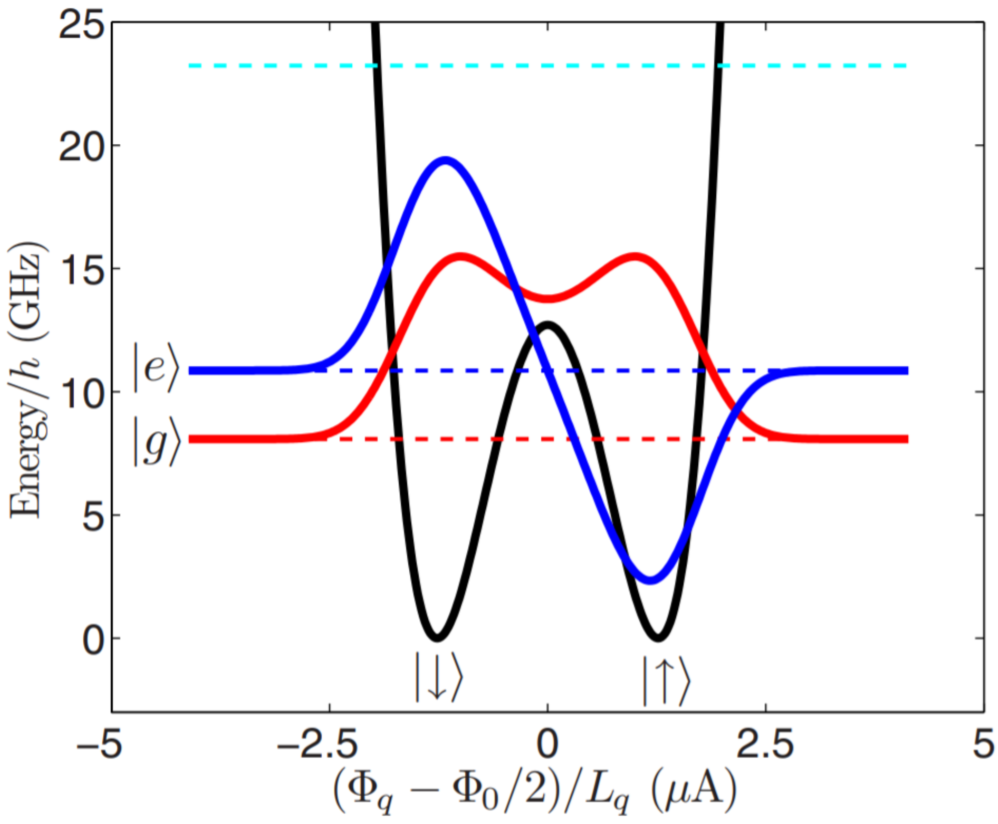

FIGURE 1.7: The two lowest-lying states of an rf SQUID at degeneracy ($\varepsilon = 0$). The expression on the x-axis corresponds to the intensity of the supercurrent flowing in the SQUID, which is linearly related to the phase δ. Image from Ref. [198].

We have two possible pairs of vectors that can be used as a basis for single-qubit states. The energy eigenbasis $\{|e\rangle, |g\rangle\}$ can be useful as it provides a low-frequency flux noise resistance [199]. Nonetheless, the most common choice is to map the logical basis of the qubit onto the localized-states-basis: $|0\rangle \to |\downarrow\rangle$ and $|1\rangle \to |\uparrow\rangle$. In this way, it is easier to implement interactions between flux-qubits via inductive coupling. A very important property that arises from such choice is that, if $\varepsilon = 0$, the ground state corresponds to $|g\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$, an equally probable superposition of the two logical states. For non-zero values of ε, the relative amplitudes of $|\uparrow\rangle$ and $|\downarrow\rangle$ in the ground state wavefunction $|g\rangle$ change. Their value is the result of a competition between ε and Δ_0. Being able to control such parameters allows to perform annealing, starting from $\{\varepsilon = 0, \Delta_0 > 0\}$ and ending at $\{\varepsilon > 0, \Delta_0 = 0\}$. During the whole process, the system should always be in the instantaneous lowest energy eigenstate of H_q. The computation ends operating a measurement with respect to the logical basis, which is also known as *computational basis*. The measurement extracts from a binary distribution whose probabilities depend on the amplitudes that the logical states $|0\rangle, |1\rangle$ have with respect to the final ground state $|g\rangle$.

Note that, for a single qubit, the computational basis at the end of the annealing process ($\Delta_0 = 0$) corresponds to the eigenstates of σ^z. For a system composed of N qubits, the computational basis is the tensor product of the N single-qubit computational basis. It means that the computational basis for the N-qubits system is the collection of the eigenstates of the Hamiltonian

$$H_{\text{N}} = \sum_{i=1}^{N} \sigma_i^z \, , \tag{1.81}$$

where each σ_i^z acts on the i-th qubit.

In [200], D-Wave Systems' researchers evaluated a scenario where qudits (qubits with more than two states) could be exploited to perform AQC, but this idea hasn't been followed by other relevant experiments up to now.

1.3.4 Implementation of the full quantum spin-glass Hamiltonian

A big step towards the employment of rf-SQUIDs as qubits was the realization of the so-called *Compound Josephson-junction* (CJJ) (image (b) in Fig. 1.8). This device became one of the first examples of a macroscopic object that behave quantum-mechanically. Indeed, some measurements managed to reveal the presence of quantum superposition of macroscopic states [47]. In this SQUID

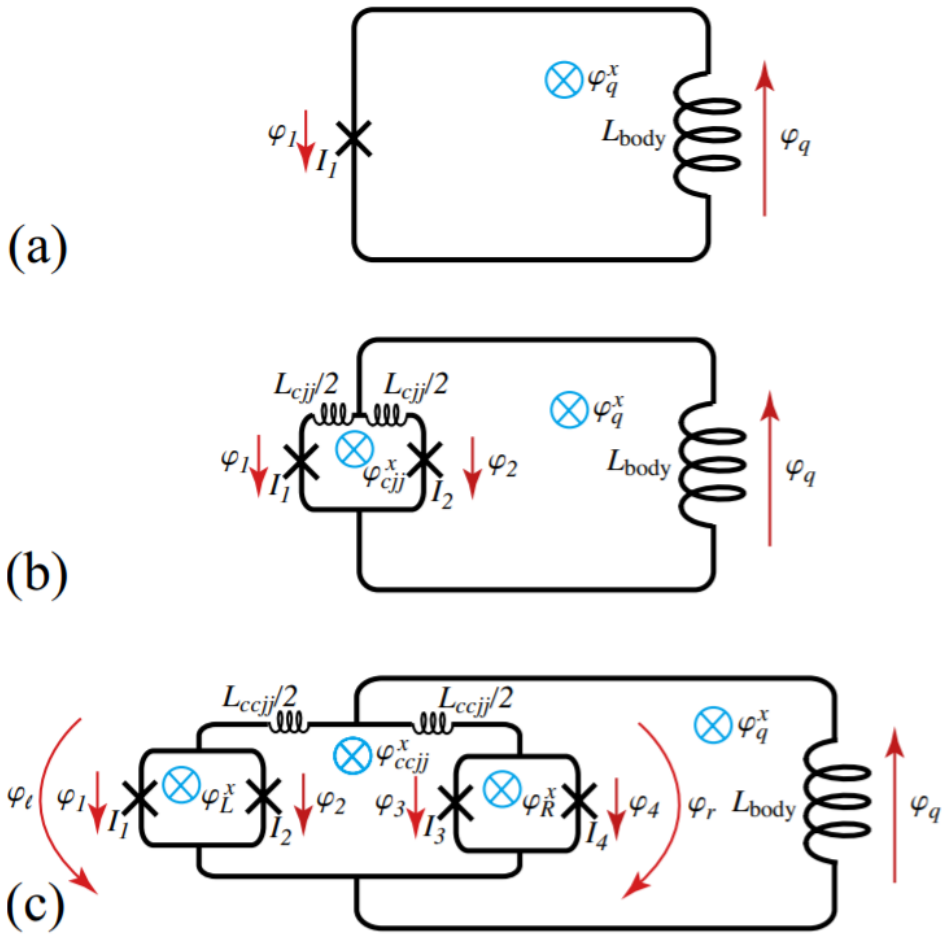

FIGURE 1.8: (a) A single-junction rf-SQUID qubit. (b) CJJ rf-SQUID qubit. (c) CCJJ rf-SQUID qubit. Image from Ref. [198].

architecture the single junction of Fig. 1.8 (a) has been replaced by a flux biased DC SQUID. The so obtained CJJ RF SQUID facilitates the *in situ* tuning of the tunneling energy. This possibility makes the CJJ RF SQUID far more useful than the simple RF SQUID. Nonetheless, such SQUID poses few relevant limitations that reduce the possibility for its application in scalable AQC [198]. As an example, it is difficult to compensate for fabrication variability among distinct qubits. Thus, one can not obtain the required homogeneity of the flux qubit parameters between the many SQUIDs that would compose an AQC.

A better topology has later been developed, which goes under the name of *Compound-compound Josephson junction SQUID* (CCJJ rf SQUID), represented in Fig. 1.8 (c). Such architecture consists of an rf-SQUID interrupted by a loop of two DC SQUIDs. The CCJJ RF SQUID overcomes the main limitation of the CJJ RF SQUID, by being more robust to variations in the fabrication of the J-junctions. Such robustness comes from the possibility to correct the shape of the potential energy separately for each qubit, by tuning the external fluxes applied to the dc-SQUIDs. The CCJJ rf SQUID can thus be used in the realization of large-scale AQCs.

The methods outlined in the previous sections suggest that CCJJ rf-SQUIDs can be used to build superconducting flux qubits. The implementation of an artificial spin-glass system that makes use of SQUIDs is also possible, but a necessary additional step is the realization of couplers that provide tunable spin-spin coupling energy. A reference outlining the principles of couplers implementation is Ref. [201]. One can then achieve a system whose behavior can be described with a spin-glass model Hamiltonian:

$$\begin{aligned} H(t) &= G(t)\sum_{i=1}^{N}\Delta_i\sigma_i^x + F(t)\left(\sum_{i=1}^{N}h_i\sigma_i^z + \sum_{i,j=1}^{N}J_{ij}\sigma_i^z\sigma_j^z\right) \\ &\equiv G(t)H_T + F(t)H_P \end{aligned} \tag{1.82}$$

where the eigenvectors of the Pauli matrices σ^z are $|\uparrow\rangle$ and $|\downarrow\rangle$. The implementation based on CCJJ rf SQUIDs allows each J_{ij} and h_i to be programmed independently.

1.3.5 Operation and timing of AQCs

Understanding how a problem is submitted to the QPU is relevant during any estimation of the computational time requirements of a problem on AQCs. Which time intervals should we consider when evaluating the computational cost, and which time interval is billed to the final user? In this Section we detail the distinct practical, operative phases that realize the annealing process in hardware, with a focus on their timings. Reference [202] points to the web-accessible D-Wave Systems manual that served as the source for the information presented in this Section.

The full pipeline from the job submission on the local Ocean SDK to getting back the collection of samples is divided into several steps. There is an initial latency time due to the communication travelling through the internet. Then, the job is queued with other jobs submitted by other users in the *worker queue*. After the worker queue is cleared, the job is prepared to be submitted to the QPU, and it is queued again with other ready-to-be-executed jobs in the *QPU queue*. After this second queue is cleared, the actual *QPU access time* starts, where the AQC is programmed to host the annealing process corresponding to the job, the annealing cycles are performed, and the obtained samples are postprocessed. Finally, a last internet-dependent latency elapses before the user can get back the collection of samples.

The QPU access time is the time interval where the AQC is exclusively reserved to the user, and thus is the amount of time that the D-Wave Systems company bills to the user. It is divided into parts: `qpu_programming_time` (T_P) and *qpu_ sampling_ time* (T_S). The programming time T_P can vary slightly depending on the problem size, but it is invariant on the number of requested samples. On the other hand, the sampling time T_S increases linearly with the number of samples, since each sample is obtained sequentially. For each annealing cycle, T_S is further divided into anneal time, readout time, and delay time, which is the time that the QPU waits before executing a new annealing cycle in order to reduce correlations due to magnetic fields induced by the previous annealing cycle.

To make a concrete example, in [1] we embed on the QPU a fully-connected problem comprising 50 binary variables, which requires 297 qubits on the Advantage 4.1 AQC. We submit several jobs associated to this QUBO problem requesting 2000 samples with a total duration of the annealing process $t_{\text{ann}} = 25\mu$s. While the total annealing time for all samples is $25.0\mu\text{s} \times 2000 = 50$ms, the total QPU access time is $\sim$301ms, which is approximately six times the annealing time. The total sum is impacted primarily by readout time, which accounts for 95μs per sample, for a total of $97.1\mu\text{s} \times 2000 = 194.2$ms, which is more than half the total billed time of 301ms. The delay time per sample is 20.54μs, for a total of $20.54\mu\text{s} \times 2000 = 41.1$ms. The initial programming time required 15.8ms. Summing the initial programming time, the annealing time per sample, the readout time per sample, and the delay time per sample we correctly obtain the total QPU access time. From this example we learn that the annealing time is relatively short with respect to the total access time. This means increasing the annealing time has a relatively small impact on the cost of the computation. Additionally, we can conclude that producing few samples (such as 100 or less) is usually not recommended, since the initial programming time would go from the $\sim 5\%$ in our example to more than 50%.

1.3.6 Undesirable behaviours in physical AQC devices

Ideal AQC

In the previous Sections we introduced the mathematical concept of adiabatic quantum evolution, a quantum process that is expected to converge to the classical configuration corresponding to the solution of a target optimization problem. This expectation derives from five main hypotheses:

1. The initial state of the system corresponds exactly to the ground state of the initial Hamiltonian H_T (see Eq. 1.37).
2. The annealing is performed slowly enough to respect the adiabaticity condition expressed by Eq. 1.34.
3. The system is completely decoupled from the environment, but for the interaction terms in the annealing Hamiltonian H_P and H_T.
4. The device operating the annealing schedule can implement exactly the desired problem Hamiltonian H_P, without errors on the coupling and bias values.
5. The final quantum configuration produced by the annealing cycle can be measured without readout errors.

The first, second and third hypotheses enforce conditions necessary for the adiabatic theorem to hold. Thus, the system will evolve keeping itself in the ground state of the instantaneous Hamiltonian. The fourth hypothesis enforces that, at the end of the annealing, the system interacts with a Hamiltonian which is exactly H_P. Since the solution state corresponds to the ground state of H_P, it follows that a measurement of the quantum system will produce with high probability the desired configuration. The fifth hypothesis must be true for the measurement to return the actual state of the qubits without errors.

An ideal AQC device should respect all the above conditions to work as expected. Unfortunately, when realizing physical AQC devices, several technological limitations may arise in manufacturing and calibration, causing defects in some qubits and couplers. In such a case, the worst scenario is the inability to program and utilize those qubits and couplers. The percentage of qubits and couplers that remain functional once the AQC device is fully exposed to users is referred to as the hardware yield. For D-Wave QPUs, the qubit yield is typically around 97% for current processors [203]. Even when a qubit or a coupler is programmable, *Integrated Control Errors* can affect the value of biases and weights, preventing the device from implementing the correct instantaneous Hamiltonian (negation of hypothesis 4). Section 1.3.6 explains how this errors can arise and what is their impact.

Nonetheless, the biggest obstacle is to ensure the validity of the hypothesis 3, because physical devices experience interaction with the outer world, making the process diabatic. The thermal noise affecting the device is described in Section 1.3.6, while Section 1.3.6 explains an interesting behaviour of the adiabatic QPU due to thermal noise, and 1.3.6 details the probability distribution of the samples obtained via quantum annealing in AQC devices.

A consequence of violating hypothesis 3 is that longer annealing process exposes the spin-glass system to noise sources for a longer time. This choice directly affects the validity of the second hypothesis, since to obtain fairly good results we are forced to operate shorter annealing cycles[26]. While AQC devices are believed to be fundamentally more resistant to noise than competitors such as gate-based QC [33], [86], decoherence and thermal noise still play a paramount detrimental role in the AQC dynamic.

Understanding the sources of errors inside the QPU allow us to design better experiments and perform successful calculations exploiting AQC. See Chapter 3 to learn how the use of proper optimization techniques can enhance the performances of AQC devices despite the presence of the undesirable behaviours presented in the following Sections.

Integrated control errors (ICEs)

Although the control parameters h and J in the annealing Hamiltonian are specified as double-precision floats, some loss of fidelity occurs in implementing these values in the D-Wave QPU. This fidelity loss may affect performance for some types of problems. Specifically, instead of finding low-energy states to an optimization problem defined by h and J as in Eq. 1.37, the QPU solves a slightly altered problem that can be modeled as:

[26]Decoherence effects are discussed in Section 1.3.6.

$$E^{\delta}_{\text{glass}}(s) = \sum_{i=1}^{N}(h_i + \delta h_i)s_i + \sum_{i=1}^{N}\sum_{j=i+1}^{N}(J_{ij} + \delta J_{ij})s_i s_j \ , \qquad (1.83)$$

where δh_i and δJ_{ij} characterize the errors in the parameters h_i and J_{ij}, respectively.

Error δh_i depends on h_i, on all incident couplings J_{ij} and on neighbour biases h_j. At higher orders, δh_i also depends on second neighbors and their incident couplings, and so on. In the same way, δJ_{ij} depends mainly on spin and coupling in the local neighborhood of J_{ij}, and to a lower extent on further elements. For this reason, forecasting the entity of the errors can be very difficult, since a modification on the value of a single bias or coupling propagates and influences also neighboring control parameters. D-Wave Systems declares that the probability distribution of δh_i and δJ_{ij} is approximately Gaussian, with mean μ^h_i, μ^J_{ij}, respectively, and standard deviation σ^h_i, σ^J_{ij}, respectively. Such values depend on the annealing fraction s. The Gaussian distribution is thus interpreted as the sum of two errors, a systematic contribution μ and a random component with standard deviation σ. For the biases, typical values for the systematic and random errors are $|\mu^h| \sim 0.02$, $|\sigma^h| \sim 0.015$, while for the weights $|\mu^J| \sim 0.01$, and $|\sigma^J| \sim 0.012$ (values representative of D-Wave 2X systems [204]). The term *Integrated Control Errors* (ICEs) refers collectively to effects that limit the dynamic ranges of bias and couplings, such as the one just described. ICEs are sources of infidelity in problem representation. Results in [205] shows that, even under the assumption that a given AQC device can be considered ideal in any other aspect besides ICEs, the success probability still decays exponentially with system size for any fixed nonzero noise level. The wild influence that control errors have on the probability of obtaining the optimal solution has been dubbed $J-$chaos [206]. Subject to $J-$chaos, an otherwise perfectly functioning AQC device will typically find the correct answer to the wrong problem. —.

Even if ICEs are present also in QCs based on quantum logic gates, in such case one can apply fault-tolerant error correction to compensate the ICEs effect [207]. This observation could be used as an argument against the adoption of AQC, but in fact a multitude of software techniques to compensate for AQC ICEs exist (see Chapter 3). As an example, Pearson et al. [208] demonstrated that an AQC device subject to $J-$chaos that achieved sub-classical scaling on an optimization problem managed to reach scaling performances superior to a classical solver after quantum annealing correction (QAC) was applied[27].

[27] See Section 3.2 to learn about quantum annealing correction.

Besides reducing the probability to find the global minimum in optimization problems, control errors can also distort the distribution of the configurations produced by AQC [209], whose shape is relevant for sampling applications[28].

An interesting empirical result regarding ICEs arised in [210], when Chancellor et al. applied the quantum annealing protocol to a frustrated chain of qubits, in order to estimate the effective "quantumness" of a D-Wave device. The experiment was designed so that the quantum device was expected to produce samples whose bits changed value approximately at the middle point of the chain. To compensate for ICEs, they evaluated the sistematic errors on the longitudinal fields when couplings and biases were set to zero. They then used biases and coupling biased to compensate such errors. The results of the experiment suggested that a quantum behaviour was indeed present, but the switching point for the chain was affected by a random noise that the researchers attributed to random longitudinal fields inside the device. This result support the above picture of a Gaussian distributed random noise on the biases and couplings. The experiment designed in [210] can be used to determine the strength of the effective random fields (noise) characterizing the annealer.

ICEs are usually divided into five classes [204]

- **Background Susceptibility (ICE1)** – The supercurrents flowing in the on-QPU junctions that allow to implement both biases and couplings can interfere with nearby coupling and biases. This leads to induced next-nearest-neighbor J interactions and a leakage of applied h biases from a qubit to its neighbors. This error source is likely the reason why in [163] the success probability on a Chimera graph was higher than the Pegasus case when addressing sparsely connected problems. The unused couplers in the Pegasus topology are indeed expected to introduce noise through *ghost couplings* [203].

- **Flux Noise of the Qubits (ICE2)** – Each bias h_i is subject to an independent (but time-dependent) error term that manifest itself as a periodic low-frequency fluctuation of the bias current. There are fluctuations in the flux noise that have lower frequency than the typical inverse annealing time, so problems solved in quick succession have correlated contributions from flux noise. By default, flux drift is automatically corrected every hour by the D-Wave system so that it is bounded and approximately Gaussian when averaged across all times.

[28]See Section 2.3.

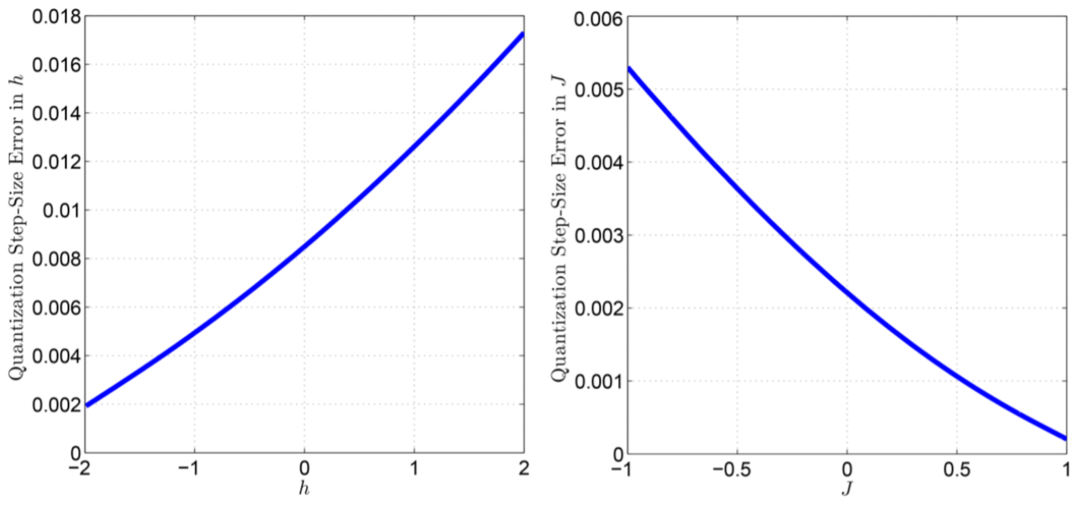

FIGURE 1.9: Typical quantization step for the DAC controlling the h parameter (left) and J parameter (right). Image from: D-Wave user manual [204].

- **DAC Quantization (ICE3)** – The on-QPU DACs that provide the specified h and J values have a finite quantization step size. This result in a random error approximately described by a uniform distribution centered at 0. Typical errors for h and J are shown in Fig. 1.9.

- **I/O System Effects (ICE4)** – Several time-dependent analog signals are applied to the QPU during the annealing process. Because the input/output (I/O) system that delivers these signals has finite bandwidth, the waveforms must be tuned for each anneal to minimize any potential distortion of the signals throughout the annealing process. As a result, the ratio h/J may vary slightly with t_f and with scaled anneal fraction s.

- **Distribution of h Scale Across Qubits (ICE5)** – It is impossible for manufactured qubits to be exactly identical one to the other, and ICEs can arise from small variations in the physical size of each qubit. When users define an h_{user} value, the actual h_{real} value realized in the hardware is different not only for a shift δh, but also for the value of the slope $(h_{\text{real}} - \delta h)/h_{\text{user}}$, that should ideally be 1. The measured slopes, however, are different for each spin, and are Gaussian-distributed with a standard deviation of approximately 1%.

Decoherence

Suppose that the quantum system of a single qubit is well defined by a certain Hamiltonian H, which has two eigenstates. If we keep H fixed in time, we can surely conclude that both eigenstates of H are stationary states of the system, i.e. do not change in time. If the system is in a quantum superposition, there will be a certain phase and amplitude difference between the occupation of the two states. Our system can be said to be coherent if such phase and amplitude differences evolve in time only according to the system dynamic. In other words, coherence is present whenever it is true that our system is correctly described by H.

Physical implementations of the qubit concept, (e.g. with a CCJJ rf-SQUID technology) are inevitably coupled to the outer environment. The presence of interactions that are not included in H means that our system will behave such that eigenstates amplitudes can decay, and phase differences can vary. In such cases, the system is said to experience decoherence. The phase of the ground state has no effect on the efficacy of QA, and therefore *dephasing* in the energy eigenstate basis is presumably harmless. On the other hand, the interactions with the environment that induce transitions between eigenstates of the Hamiltonian might cause trouble. Therefore, any physical realization of a quantum computer is built posing great attention to prolong as much as possible *decoherence time*. This term refers to the time it takes for the qubit to experience sufficient decoherence to make its quantum state useless for computing.

For flux qubits (based on the SQUID) decoherence times are usually measured to be around dozens [211] or hundreds of nanoseconds [198]. It is important to note that such times are much lower than the fastest annealing time available on a state-of-the-art AQC (0.5 μs on D-Wave Systems Advantage). Indeed, in general decoherence in AQC devices limits the fraction of Hilbert space that may be accessed, and so the extent to which the exponential scaling of Hilbert space can be taken advantage of in computation [212]. Nonetheless, some authors reported that AQCs appear to be surprisingly resistant to noise and imperfections and that they can show evidence of quantum behavior for annealing times of the order of microseconds [91], [106], [213]. In 2001, Childs, Fahri, and Preskill were among the first to study the resilience of AQC against decoherence [86]. They showed with numerical simulations that QA is robust against thermal noise and certain kinds of random unitary perturbations. In [214], authors experimentally demonstrated that, for those instances where the gap between the ground and first excited state becomes small during the annealing process, the probabilities of performing a successful computation are similar to those expected for a fully coherent system even with annealing times eight orders of magnitude longer than the predicted single-qubit decoherence time. In [215] performances of QA were enhanced by decoherence in a scenario where the gap from the ground state was chosen to be much lower than the thermal noise. Nonetheless, in this case the enhanced capabilities didn't came from thermal noise but was a consequence of the number of thermally accessible states from the instantaneous ground state at the quantum critical point[216].

The next Section provides further details regarding temperature, the number one cause of decoherence.

Temperature

Inside the physical device, the qubits system may interact with outer subsystems, namely the other components inside the chamber of the refrigerator. We will call *environment* the collection of such subsystems. When the external control signals have been carefully filtered and the processor has been cooled down, the environment seen by the quantum processor may be a thermal bath at a fixed temperature.

The magnitude of temperature indicates the quantity of energy that the refrigerator can not remove from the QPU. The presence of thermal energy results in a diabatic evolution, which breaks the third condition for the ideal AQC. This condition is fundamental both for the validity of the adiabatic theorem and for the functioning of every other QC. Nonetheless, AQC as an hardware approach to QC appears to be much more resilient to thermal noise than competitors. In some cases, QA is even expected to be enhanced by a proper amount of thermal noise [141], [214].

In reference [214], Dickson et al. notice that, in the limit of slow evolution, the ground state will always have the dominant probability. This is true only if we suppose that the excited states are occupied approximately with equilibrium Boltzmann probabilities[29]. It should be noted that a slow evolution exposes the system to additional error sources. Indeed, the computation should be executed in a time short enough to consider it *robust against the environment*. Robustness against environmental noise can be defined as the ability of an open quantum annealing system to yield the correct solution with acceptable probability within a time comparable to the closed system adiabatic timescale.

Reference [214] presents a truly interesting scenario in which thermal noise can help the annealing process, raising the probability to reach the ground state. During the annealing process, the energy gap $g(s)$ between the ground state and the first excited state varies as the parameter s goes from 0 to 1. We call $g_{\min} = g(s*)$ the minimum value for $g(s)$, which is reached at some

[29]See Section 1.3.6.

point $s = s*$. Consider a system where $g(s*)$ is small: $s*$ is then known as an avoided crossing point, or *anticrossing* point [217] (illustrated in Fig. 1.10). The effect is a manifestation of the Wigner-von Neumann non-crossing rule, which states that the curves that describe the dependence of two eigenenergies do not cross on the energy-time plane [93]. Indeed, imagine a 2×2 Hamiltonian that describes the two (close to each other) lowest energy states of the system, and neglects the rest of the spectrum. Let E_1 and E_2 be the diagonal matrix elements of the Hamiltonian, and $V_{12} = V_{21}^*$ be its off-diagonal matrix elements. We then find the energy gap to be:

$$\Delta = E_{\mathrm{ES}} - E_{\mathrm{GS}} = \sqrt{(E_1 - E_2)^2 + |V_{12}|^2}. \tag{1.84}$$

Where E_{ES} is the energy of the first excited state and E_{GS} that of the ground state. Now suppose $E_1(s) = E_2(s)$ at a specific $s = s^*$. Even in such case, $\Delta > 0$ due to the off-diagonal elements. An actual crossing takes place only if the two lowest instantaneous eigenstates have exactly the same eigenvalue, which can only happen in the presence of particular simmetries (in the example, a diagonal Hamiltonian). Despite the avoided crossing in the energies, the ground state is effectively changed from $s < s^*$ to $s > s^*$. Passing through the anticrossing very quickly can in fact swap the probabilities of the ground and first excited states [214]. If we call $|a\rangle$ the initial ground state, and $|b\rangle$ the initial first excited state, then, for a system starting in state $|a\rangle$, the probability P_b of ending in the ground state would be small. However, with an environment at $T > 0$, thermal transitions can excite the system beforehand and relax it afterward. The direct effect is to increase P_b. From quantum annealing schedule, we know that spin-flip amplitudes are larger in the early phases of the annealing, and tend to vanish as the σ_x component of the Hamiltonian is gradually reduced. For this reason, excitation before the anticrossing point will be more frequent than relaxation afterward. This effect raises the probability for the system to be driven into the first excited state before the anticrossing point, which means P_b is also raised. A temperature $T > \delta E_k/k_B$ would lead to the occupation of higher energy excited states, which would reduce P_b. A peak in P_b is expected at $T \sim T_{\mathrm{peak}} = \delta E_k/k_B$.

This thermal annealing enhancement holds only if the gap $g_{\min}$ is small and the two lowest-energy states are separated from other energy level by a gap $\delta E \gg g_{\min}$.

In general though, thermal excitation can reduce the instantaneous probability of the ground state by populating the excited states and that the final

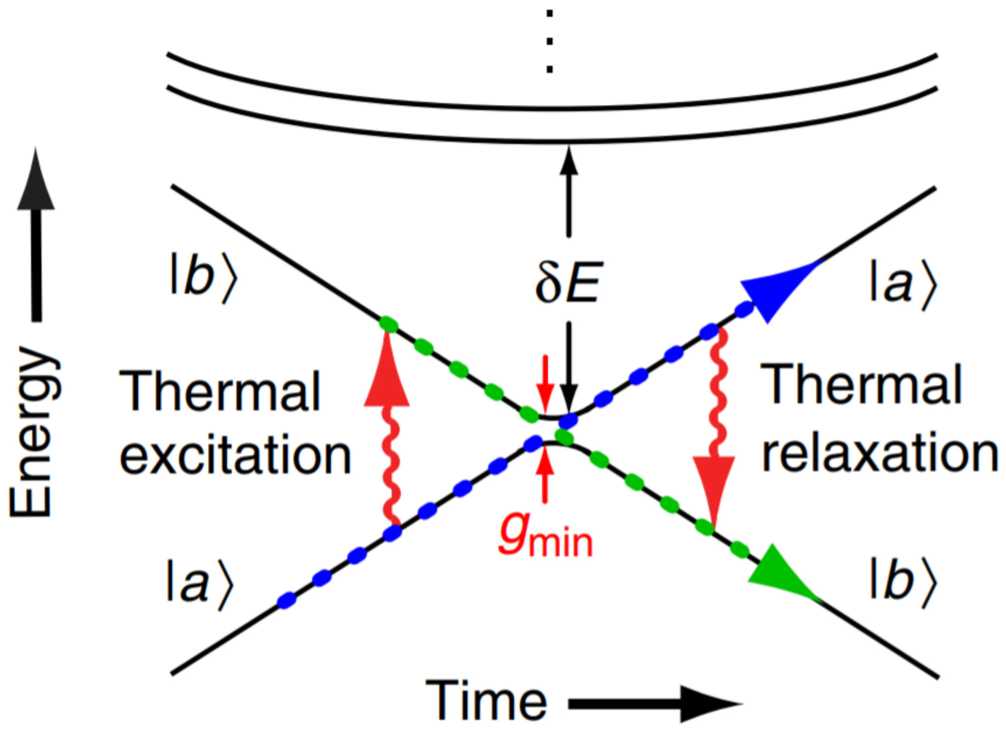

FIGURE 1.10: An avoided crossing (anticrossing) of two lowest-energy eigenstates,$|a\rangle$ and $|b\rangle$, with a small minimum energy gap, $g$min, separated from other eigenstates by energy δE. "Passing through the anticrossing very quickly swaps the probabilities of the ground and first excited states (blue and green dotted arrows), leaving the probabilities of $|a\rangle$ and $|b\rangle$ unchanged. Thus, for a closed-system starting in $|a\rangle$, the final ground-state probability, P_b, would be vanishingly small. However, with an environment at $T > 0$, thermal transitions can excite the system beforehand and relax it afterward (red arrows), the net effect of which is to increase P_b (green arrow). Single-qubit tunnelling amplitudes are significantly larger before the anticrossing, making thermal excitations earlier in the annealing process much more likely than relaxation later. If $T \gtrsim \delta E/k_B$, higher excited states would also be occupied, reducing P_b; so a peak in P_b is expected at $T = T_{\text{peak}} \sim \delta E/k_B$.

measurement may produce a configuration that has a non-zero Hamming distance from the ground state. Such distance is related to the amount of energy deposited on the QPU. For such reason, the QPU must be kept to the lowest temperature achievable. The most recent D-Wave Systems processor, D-Wave 2000Q Systems, has an operating temperature of $\sim$ 14 mK.

Freezing point

We previously stated that QA makes use of quantum fluctuations to explore the phase space, as opposed to SA that exploits thermal fluctuations. Nonetheless, thermal fluctuations are unavoidably present in the physical realization of an AQC. During the annealing process, the energy barrier δU between the two minima represented in Fig.1.7 is gradually raised. If thermal fluctuations dominate the qubit dynamics, the qubit state can move from one well to the other with a rate proportional to $e^{-\delta U/k_B T}$, where T is the temperature of the system. The qubit motion is then expected to approximately stop when $\delta U \approx k_B T$. The freezing happens at $t \approx t^T_{\text{freeze}}$, so that $\delta U(t^T_{\text{freeze}}) \approx k_B T$.

Otherwise, if quantum mechanical fluctuations dominate the dynamics of the system, the qubit may tunnel across the barrier. When δU is increased, quantum tunneling becomes less and less likely to happen. Then we expect the quantum effects to vanish at a certain time t^Q_{freeze}, which is expected to be nearly independent of T. This freezing effect was forecast by a numerical simulation based on PIMC performed by Martovnak et al. in 2002 [136].

A study of the T-dependence of the freezing time t_{freeze} can show which is the dominant effect driving the system dynamics. Experiments conducted on chains of superconducting flux qubits show that t_{freeze} saturates at low T. This observation suggests that, for low temperatures, the dynamics of the system is dominated by quantum mechanical effects. Fig. 1.11 shows the results of an experiment conducted on an eight-qubit chain [91]. At higher temperatures, the freezing time t_{freeze} follows an approximately linear dependence on T. When $T <\sim$ 50mK, such dependence disappears, and t_{freeze} assumes a fixed value.

When thermal fluctuations dominate, the linear dependence of t_{freeze} with respect to T is to be expected, since δU is increased linearly with time, and $\delta U(t^T_{\text{freeze}}) \approx k_B T$.

In general, the freezing time t_{freeze} depends on the device temperature and the potential energy landscape (which also means it depends on its time evolution, i.e. the annealing schedule). Its value comes from the combined effect of quantum and thermal fluctuations, thus it can be difficult to predict. Furthermore, the dependence of the effect on δU implies that the freezing point is

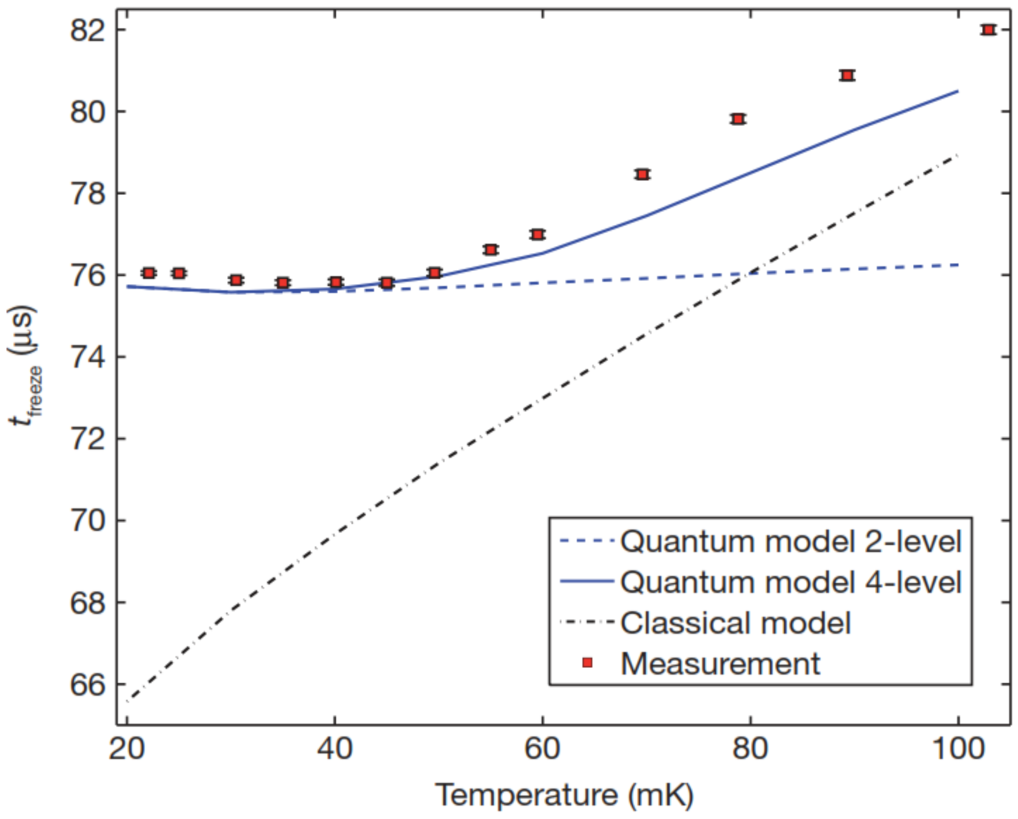

FIGURE 1.11: Measured t_{freeze} versus T (red dots). Simulated plots of t_{freeze} have been estimated from two-level (dashed blu) and four level (solid blue) quantum machanical models. The dashed black line shows the value for t_{freeze} simulated with a classical model of the qubits. Error bars 1σ. Image from Ref. [91]

expected earlier when employing long chains in the embedding, since the energy gap between the upward and downward configuration for the chain qubits increases. A consequence of this is that embeddings using chains of different lengths could experience the freeze-out phenomenon at different times for each chain, potentially warping the expected Boltzmann distribution in sampling applications[30].

The freeze-out effect is not limited to AQC device, but extends to other quantum systems such as a crystals of the Ising magnet $\mathrm{LiHo}_x\mathrm{Y}_1-x\mathrm{F}_4$. In [218], a cubic-centimeter–sized crystal of such insulating magnetic salt was coupled to a heat bath. The authors showed that, when the material is more strongly connected to the heat bath, the local magnetic clusters behave more classically and freeze simultaneously into a glassy state. Tuning the coupling with the heat bath allows to set the tendency of the system to exploit quantum mechanical modes of relaxation. Such results support previous findings obtained in a similar experimental setup [83].

The next Section extends the discussion above by describing the expression of the distribution of states that results at the end of the annealing schedule. Such expression is strongly correlated with the position of the freeze-out point during annealing.

States distribution at the freezing point

QA theory suggests choosing long annealing times so that the adiabatic theorem requirements are satisfied. More specifically, the time dependence of the system should be very slow compared to its relaxation time, so that the system can reach equilibrium at all times. In thermodynamics, such a system is called *quasistatic*. We already pointed out that in physical implementations of QA the system follows the equilibrium distribution only up to a certain point $s_{\text{freeze}} = t_{\text{freeze}}/t_{\text{f}}$. After this point, system dynamics appears to stop, so that the final distribution after the anneal closely resembles the equilibrium distribution at s_{freeze}.

We are now interested in studying the distribution of the states that are obtained after an annealing cycle. Reference [219] provides a numerical approach to the problem. A 16-qubit problem is considered, with h_i selected from $\pm 1/3$ and J_{ij} from $\pm 1/3$ or -1 uniform randomly. Fig. 1.13(a) shows the 12 lowest energy eigenvalues of the problem considered. Their time evolution is obtained by the shapes of $F(s)$ and $G(s)$ plotted in Fig. 1.12, which are realistic shapes obtainable in a D-Wave device. The dashed black lines represent the corresponding classical energies, i.e., eigenvalues of $F(s)H_P$. Occupation

[30]See Section 2.3 to learn how AQCs can accelerate sampling tasks.

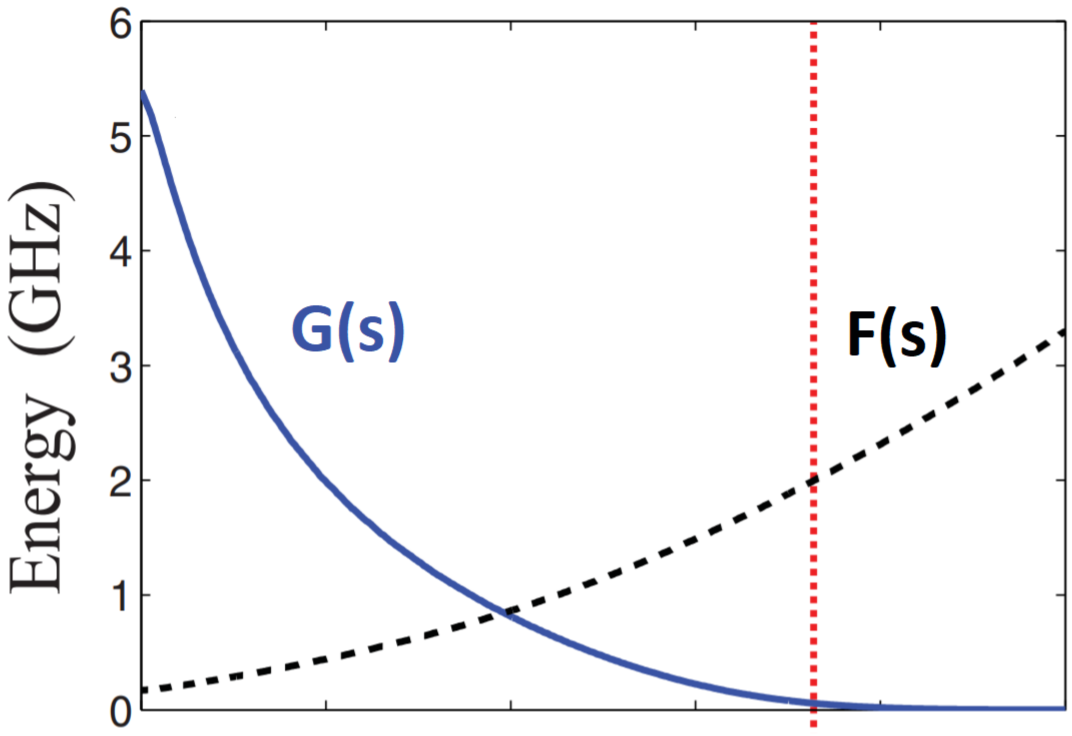

FIGURE 1.12: Typical shapes for $F(s)$ and $G(s)$ obtainable in a D-Wave processor. Image from Ref. [219]

probabilities are then calculated by using the open quantum (Redfield) master equation, assuming an Ohmic environment at equilibrium temperature $T = 40$ mK (See ref. [100]). Circles in Fig. 1.13(b) represent the occupation probabilities of the lowest 12 eigenstates during the evolution. Fig. 1.13 (b) shows by solid lines the equilibrium probability calculated by using the Boltzmann distribution:

$$P_n(s) = \frac{e^{-E_n(s)/k_B T}}{Z}, \tag{1.85}$$

where E_n is an instantaneous eigenvalue of the full anneal Hamiltonian $H(s)$, and Z is the partition function. In Fig. 1.13(a) the probabilities closely follow $P_n(s)$ up to almost $2/3$ of the evolution (green region). As $s \to 1$, $G(s)$ becomes smaller, making thermal relaxation slower. When relaxation becomes too slow, probabilities deviates from Boltzmann distribution. The system enters a freezing region (yellow), where the probabilities saturate and stop changing. If this area is narrow, final probabilities will follow the Boltzmann distribution at a single freeze-out point $s = s*$, marked by the red dotted line. This behaviour has been demonstrated experimentally [220] (see also Section 2.3).

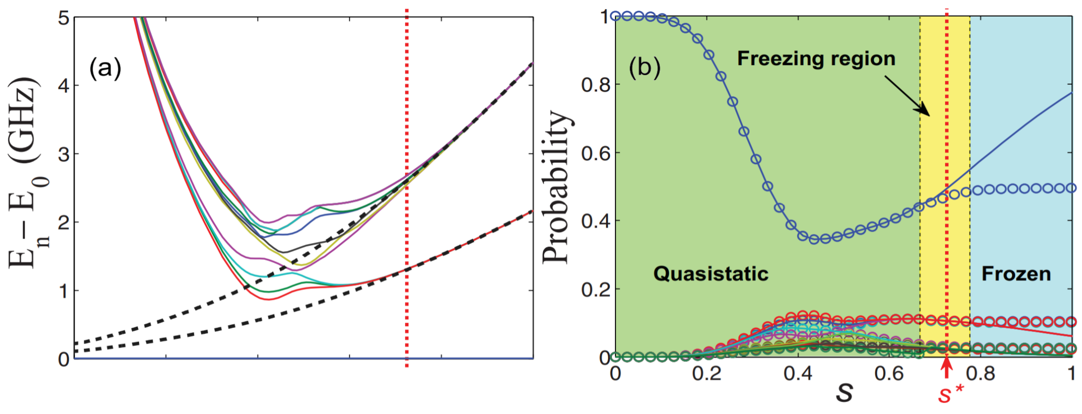

FIGURE 1.13: (a) The lowest 12 energy levels of a randomly generated 16-qubit problem. Dashed black lines are classical energies. Notice that the quantum energy eigenvalues are close to the classical ones at s^*. (b) Occupation probabilities during the annealing calculated by using the Redfield formalism (circles) and the Boltzmann distribution (solid lines). All probabilities follow the Boltzmann distribution in the quasistatic region (green) until they start freezing in the freezing region (yellow) and stay constant in the frozen region (blue). All final probabilities are close to the Boltzmann probabilities at the freeze-out point s^*, marked by the vertical (red) dashed line. Images from Ref. [219]

We can then approximate the final quantum state $|\phi\rangle$ of the system as:

$$|\phi(\tau > \tau_{\text{freeze}})\rangle = \frac{1}{Z^{1/2}} \sum_{j=1}^{2^N} e^{i\theta_j} e^{-\frac{E_j}{2T_{\text{eff}}}} |S_j\rangle \ , \tag{1.86}$$

where the sum over j is the sum over each state S_j of the eigenstates basis (computational basis), Z is such that the quantum superposition is correctly normalized and E_j is the energy eigenvalue of state $|S_j\rangle$ with respect to H_P. Equation 1.86 is such that each configuration S_j is measured with a probability equal to $e^{-E_j/T_{\text{eff}}}/Z$. Measuring this system after the freeze-out point can be seen as sampling an istantaneous configuration from a classical spin-glass model at the effective temperature $T = T_{\text{eff}}$.

It is fundamental to observe that the hypothesis expressed by Eq. 1.86 allows us to think of a quantum annealing device as a generator of samples that follows the Boltzmann distribution of the classical cost function encoded in H_P. Even if this behavior is undesirable when we are solving an optimization problem, it is exactly what we need to generate the correct samples needed for RBM training[31].

The parameter T_{eff} can vary between different annealing cycles and its evolution is considered extremely difficult to predict. It can be viewed as an empirical

[31]See Section 2.4.

parameter that depends on the operating temperature of the hardware and other sources of error that affect the device.

The intuition behind this freeze-out phenomenon is that the dominant coupling of the qubits to the environment or bath degrees of freedom works via the σ^x operator [221]. Since at the freezing point we have $F(\tau_{\text{freeze}}) \gg G(\tau_{\text{freeze}})$, and the interaction with the bath lacks a strong σ^x component capable of causing relaxation between the states of the computational basis (i.e., eigenstates of σ^z), the system cannot relax its population anymore; in other words, its population dynamics freezes.

A technique to estimate the inverse temperature β_{eff} that defines the distribution at the end of the annealing was devised in [221] based on the observations outlined in this Section. The probability for QA to produce a sample with energy E is $P_\beta)(E) = g(E)\exp(-\beta E)/Z(\beta)$, where we renamed $\beta_{\text{eff}} = \beta$ for simplicity, $g(E)$ is the degeneracy of the energy level E, and $Z(\beta)$ is the partition function needed to normalize the probability. Then, for two distinct energies E_1, E_2 the log-ratio of the probabilities becomes:

$$\ell(\beta) \equiv \ln \frac{P_\beta(E_1)}{P_\beta(E_2)} = \ln \frac{g(E_1)}{g(E_2)} - \beta \Delta E \ , \tag{1.87}$$

where $\Delta E \equiv E_1 - E_2$. The ingenious procedure prescribes to rescale all coefficients of the Hamiltonian by a factor $x \in (0, 1)$. Letting $\beta' = x\beta_{\text{eff}}$, we can evaluate $\Delta\ell \equiv \ell(\beta) - \ell(\beta')$ obtaining:

$$\Delta\ell = \ln \frac{P_\beta(E_1)P'_\beta(E_2)}{P_\beta(E_2)P'_\beta(E_1)} = \Delta\beta\Delta E \ , \tag{1.88}$$

where $\Delta\beta = \beta' - \beta = (x - 1)\beta_{\text{eff}}$. In this way, by generating a second set of samples at a suitable value of x and then taking the differences of all pairs of populated levels, we can plot $\Delta\ell$ against ΔE. According to Eq. 1.88 this is expected to be a straight line with slope given by $(x - 1)\beta_{\text{eff}}$. This is then a viable strategy to estimate T_{eff}. See Ref. [221] and [222] for rules-of-thumb to chose x.

As stated in Section 1.3.6, if the dynamic is dominated by thermal fluctuations, the motion of the system is expected to freeze-out when $\delta U \approx k_B T$, where δU is the approximate gap between the occupied states and the closer excited states. We then expect the dynamic to stop at s_{freeze} s.t. $B(s^*) \sim 1/\beta_{\text{eff}}$. Knowing β_{eff} thus gives an estimate of the freezing time.

In a recent article, Pelofske et al. [222] introduce a *slicing method* to infer the istantaneous state of the quantum annealer at various points during the annealing. They are able to take a snapshot of the adiabatic evolution by *quenching*[32] the system, which consists in quickly ending the annealing process speeding up the anneal schedule as much as allowed by the hardware. Since the quenching phase will take at least 0.5μs on modern AQC devices, and a usual full annealing takes around 20μs, the quenching is not instantaneous, and thus the results obtained via this method can be afflicted by errors. They observe that the rule-of-thumb that estimates s^* based on the thermal fluctuations argument seems to underestimate s^*, since the slicing method, which is not expected to overestimate s^*, consistently forecast a later freezing point.

It is important to underline that empirical evidence and additional theoretical arguments suggest that an output Boltzmann distribution is to be expected only when the dynamic is in a negligible quantum fluctuations regime [223]. It should be noted that in the majority of cases the negligible quantum fluctuations hypothesis does not hold, which means more often than not there is no reason to believe that the output should follow a classical Boltzmann distribution. The resultant distribution will generally not correspond to an equilibration at any given point, but it may instead result from different parts of the system equilibrating at different temperatures and times [219]. Nonetheless, in many scenarios the Boltzmann distribution hypothesis has been exploited with success. As an example, see Section 2.3, where multiple empirical applications of AQC on sampling problems are presented.

1.3.7 An alternative hardware approach to AQC: atomic lattices

The concept of adiabatic quantum evolution presented in Section 1.2.1 is not bounded to a specific physical system or hardware. While QA is usually associated with the adiabatic evolution of an Ising system, the concept of AQC is more general, as it hypothesizes simply a system composed by two tunable Hamiltonians (one implementing the problem as its ground state and the other allowing the quantum degrees of freedom to switch between different states of the computational basis). The most successful approach up to date is the D-Wave Systems SQUID-based AQC technology, which is the main topic of this Section 1.3. Nonetheless, several AQC schemes based on atomic lattices have been explored in the past and continue to arise interest as the most promising alternatives to superconducting AQC devices [224].

In [225], Hauke et al. proposed a system of cold trapped ions[33] that uses two

[32]See Section 3.1.2.
[33]See Section 1.1.2.

electronic hyperfine sublevels of each ion as qubits. The Hamiltonians required for AQC can be realized by coupling the ions via lasers or microwave radiation and qubits are then measured by appropriate single-qubit rotations followed by stimulated fluorescence measurements. The approach has been further studied and detailed in [226]. In [227], Graß et al. empirically tested this approach using a system of six $^{40}Ca^{+}$ ions to solve via AQC the number partitioning problem [228].

In 2017, Glaetzle et al. proposed an AQC scheme based on neutral Rydberg atoms [61]. Qubits are encoded in two long-lived hyperfine ground states of ^{87}Rb and ^{133}Cs. Qubits are then coupled using Rydberg dressing [229]. A potential advantage of such approach is that the atoms can be arranged in almost arbitrary 2D geometries using optical tweezers (as done in [230] as an example).

Chapter 2

Practical Applications of Adiabatic Quantum Computers and other unconventional computing paradigms

2.1 Introduction to the practical applications of Adiabatic Quantum Computing

This Section presents the computational fields where AQCs are expected to have a beneficial impact in the near future. First, the class of problems natively accepted by AQC devices (QUBO) is introduced, along with basic techniques to approximate non-QUBO problems as QUBO problems. Then, the most relevant optimization and sampling problems that can be tackled with AQC are explained from a mathematical perspective, explaining in particular how to map them to QUBO form.

2.1.1 QUBO problems

As anticipated in Section 1.2, quantum annealing has a preferred application as technique to solve combinatorial optimization problems called QUBO problems. The acronym stands for Quadratic Unconstrained Binary Optimization problems. These keywords are reordered and explained in the following list:

- Optimization: the problem consists in finding the combination of values for the problem variables that minimize a given cost function;
- Binary: the variables of the cost function are binary, and in general they must assume values $\in \{0; 1\}$;

- Quadratic: the cost function is a polynomial of the variables comprising only linear or quadratic terms;
- Unconstrained: the optimization problem has no constraints, meaning the variables can in principle assume any combination of values.

In other words, every QUBO problem consists in finding a combination of values for the binary variables $q_1, ..., q_N \in \{0; 1\}^N$ such that the cost function

$$F(q_1, ..., q_N) \equiv \sum_i h_i q_i + \sum_{i,j} J_{ij} q_i q_j \tag{2.1}$$

is minimized, where h_i and J_{ij} are real parameters defined by the problem statement. When $q_1, ..., q_N \in \{-1; +1\}^N$, the problem is sometimes called *Ising problem*. This convention will be used throughout the text.

It follows that a user of AQC is required to translate the problem she wants to solve into a minimization problem, and then submit it to the QPU. Then, contrary to gate based QC, there is no algorithmic procedure to be devised in order to obtain a solution. This is why the programming paradigm for AQCs is sometimes defined *declarative* rather than *imperative* [33]. Nonetheless, efficiently mapping some optimization problems to QUBO form can still pose some difficulties. The reader can learn more about the topic reading [231], where relevant NP-hard problems, including Karp's 21 NP-complete problems, are translated into QUBO form, and [232], where strategies are presented to make the mapping hardware-aware. Additionally, several specialized middleware software tools are available to lighten the burden of the generation of medium-sized QUBO problems [233], [234].

2.1.2 Introduction of constraints as penalty terms

By definition, QUBO problem are Unconstrained, which means no constraint can be implemented exactly. Nonetheless, optimization problems of industrial relevance often require the presence of constraints, usually represented by equalities. The most widely adopted technique to impose equality constraint in QUBO form is to write the equality such that the right-hand side is zero, and then square the left hand side, obtaining a positive expression that attains the minimum value at zero, when the equality constraint is respected [235]. A typical equality constraint can take the form

$$\sum_{i=1}^{n} x_i = k \ , \tag{2.2}$$

which means that exactly k of the QUBO variables x_i must be set equal to 1. We can implement this constraint by adding to the problem cost function the term:

$$P(x) = M \left(\sum_{i=1}^{n} x_i - k \right)^2 \ , \tag{2.3}$$

where M is a real positive value parametrizing the strength of the constraint. The constraint in Eq. 2.3 attains zero value only for those x_i that respect the equality condition in Eq. 2.2. In all other cases, the constraint is unsatisfied and $P(x) > 0$ [236].

This penalty based method presents some relevant shortcomings. First, the squared expression can end up containing higher-than-quadratic terms, resulting in a non-QUBO expression[1]. Additionally, contrary to what happens in classical methods, the phase space is actually not restricted by the imposed constraints, since the penalty term is merely suggesting the system to avoid the phase space where the constraint is unsatisfied. Since the phase space portion that corresponds to feasible solutions can be exponentially small, the presence of noise can strongly reduce the probability to find a solution [237]. Furthermore, the squaring of the constraints normally results in a pattern of interactions that pairwise couple all the input variables. This is known to potentially introduce additional complexities[2].

Implementing an *inequality* constraint is harder, since the squaring strategy does not apply. A general inequality could take the form:

$$\sum_{i=1}^{n} x_i \leq k \ . \tag{2.4}$$

The easiest way to implement this inequality is to introduce a slack variable z such that $(\sum_{i=1}^{n} x_i) + z = k$ [238]. Since an arbitrary k can assume non-binary values, z will require potentially many Boolean variables to be correctly represented[3]. More elaborated techniques to efficiently implement constraints have been recently developed [239], [240] using the Hubbard-Stratonovich transformation in an approach that resembles the Lagrangian relaxation in traditional combinatorial optimization.

[1]See Section 2.1.4 to learn how to manage this situation.
[2]See Section 1.2.10.
[3]See Section 2.1.3 to learn how to represent integer or continuous variables in QUBO form.

2.1.3 Non-binary discrete variables encoding

One of the first questions that could arise when facing the QUBO formulation is: what about problems whose variables are not binary? Do we have to give up the idea of solving such problems with AQC? Luckily, there are various techniques to map quadratic optimization problems with integer or continuous variables to QUBO form. Nonetheless, sometimes the overhead of the conversion can result in poor performances of AQC devices.

We can face problems where variables are continuous or non-binary integers. Since there is no way to represent continuous variables with binary variables, in the case of continuous variables we will simply require the continuous numbers to be cut-off at a specific decimal (as it is customary in classical computation). We can then rescale the whole problem by a sufficiently large power of ten to obtain a discrete approximation of the continuous problem. We are then interested only in mapping integers to binary variables.

The most naive way to map a quadratic integer optimization problem to QUBO form is to perform the following substitution for each variable:

$$V_i = c\sum_{j=0}^{n} b^j a_j \tag{2.5}$$

where $V_i < b^{n+1}$ is the $i-$th integer variable of the original formulation, while a_j is the $j-$th binary variable in the QUBO formulation, b is the logarithmic resolution of the mapping, and $c \in \mathcal{R}$ is a scale factor defining the range of the original variable. Note that to perform the mapping the integer variables have to be both upper and lower bounded. If the variable can be negative, it can be properly shifted to be always positive and respect Eq. 2.5. Usually, $b = 2$ to be able to represent every possible integer between the upper and lower bound. Basically each integer variable is decomposed in basis 2 and expressed by $\lceil \log_b V_i \rceil$ binary variables.

Practically, this mapping can result problematic for two main reasons. First, introducing many binary variables can lead to increased complexity of the energy landscape reducing the probability to find the global minimum. This is to be put in perspective considering that classical solvers require no additional overhead in managing Integer Linear or Quadratic Programming [241], which means the expectancy of quantum advantage diminishes in such cases. Secondly, representing a large integer variable could require the use of many binary variables with corresponding weights 2^j ranging from 1 to 2^n. If $n \gg 1$, the smallest weights could be heavily affected by ICEs[4], causing the implementation of a warped problem Hamiltonian.

Very few cases exist in literature where researchers tackled quadratic optimization problems with continuous variables using AQC, and they all use the binary encoding just presented [242]–[244]. In such cases, the precision requirements are the most detrimental factor in AQC performance.

Besides binary encoding, two other approaches exist to map e higher-than-binary discrete variables to Boolean variables: *one hot encoding* and *domain wall encoding* [245]. They are unary encoding where the number of Boolean variables needed to encode the integer variable grows linearly with the integer variable range.

In one hot encoding, each Boolean variable simply represents a possible value of an integer variable. The mapped problem is then defined on two index variables, $x_{i,\alpha}$ such that:

$$x_{i,\alpha} = \delta(i, \alpha) \tag{2.6}$$

where $\delta(i, \alpha) = 1$ if variable i equals α, and $\delta(i, \alpha) = 0$ otherwise. We also have to introduce a constraint to impose that $x_{i,\alpha} = 0$ for every value of α but one. This constraint can be realized as:

$$H_{\text{one-hot}} = k\left(\sum_{\alpha} x_{i,\alpha} - 1\right)^2 , \tag{2.7}$$

where the sum is performed over any value for α, and k parametrizes the strength of the constraint. A general DQM[5] (discrete quadratic model) can then be expressed as [246]:

$$H_{\text{DQM}} = \sum_{i,j}\sum_{\alpha,\beta} D_{(i,j,\alpha,\beta)} x_{i,\alpha} x_{j,\beta} , \tag{2.8}$$

where $D_{(i,j,\alpha,\beta)}$ are the pairwise interactions which determine the overall energy of a configuration.

In domain wall encoding, each integer variable $i \in [0, m]$ is encoded in a frustrated chain of m Boolean variables σ_j with $j \in [0, m-1]$. Boundary conditions are enforced setting $\sigma_-1 = -1$ and $\sigma_m = 1$. The Hamiltonian is

[4]See Section 1.3.6 to learn about Integrated Control Errors.

[5]Discrete quadratic models are optimization problems expressed on discrete variables with arbitrary pairwise interactions. They can be seen as a generalization of QUBO problems where variables are allowed to assume (non-binary) integer values.

defined as

$$H_{\text{domain-wall}} = -k \sum_{\alpha=-1}^{m-1} \sigma_\alpha \sigma_{\alpha+1} \tag{2.9}$$

Problems can then be defined as in one hot encoding by noting that

$$x_{i,\alpha} = \frac{1}{2}(\sigma_{i,\alpha} - \sigma_{i,\alpha-1}) \ . \tag{2.10}$$

Basically, the integer variable is encoded as the point where the spins in the frustrated chain change value. Since $x_{i,\alpha}$ in Eq. 2.10 is expressed by a linear term, the domain-wall encoding of a DQM is always quadratic in the underlying binary variables.

In terms of number of binary variables used to encode a discrete variable (ignoring embedding overhead), the most efficient method to encode a DQM is to use binary encoding. If a unary encoding is preferred, domain wall encoding was shown to lead to preferable dynamics during the annealing process if compared to simple one-hot encoding, which in turn leads to an increased probability of finding feasible solutions [245]–[247]. From empirical results, domain wall seems to outperform one-hot encoding on any reasonable metric, even when the experiment setup compares on the same problem one-hot on Pegasus and domain wall on Chimera topology [248].

While other encodings from DQM to QUBO exist in literature [249], those presented above are used in the vast majority of the existing literature.

2.1.4 HUBO problems

The previous Section showed how to extend the boundaries of QUBO problem to comprise also *constrained* instances. In this Section we will show how to implement cost functions with a degree higher than two. Binary optimization problems with terms of a degree higher than two are called HUBO (or HOBO) problems (Higher-order QUBO problems) [35].

We saw in Section 1.2.10 that a problem that does not respect the topology of an AQC device can be nonetheless submitted using proper embedding techniques. In a similar way, HUBO problems can be tackled by AQC if we map them to QUBO problems having equivalent solutions. Whenever a HUBO problem contains a cubic term, e.g. $5q_1q_2q_3$, we can introduce a new binary variable $q_{1,2} = q_1q_2$, $q_{1,2} \in \{0, 1\}$. The new expression $5q_{1,2}q_3$ is of degree two, so it is a QUBO problem. To guarantee the problem is equivalent to its previous formulation, we have to enforce the condition $q_{1,2} = q_1q_2$. This is done by adding a *penalty term* to the cost function. Consider the following expression:

$$C(q_{1,2}, q_1, q_2) = 3q_{1,2} + q_1q_2 - 2q_1q_{1,2} - 2q_2q_{1,2} \tag{2.11}$$

The above expression for $C(q_{1,2}, q_1, q_2)$ has the interesting property of being equal to zero in each and only those cases where $q_{1,2} = q_1q_2$. In the remaining cases, $C(q_{1,2}, q_1, q_2) > 0$ (in particular, it is equal to 1 or 3). This means that, by adding this penalty term, any configuration that does not respect $q_{1,2} = q_1q_2$ cannot be a global minima, which also means the equality is respected by any global minima. From a different point of view, Eq. 2.11 can be seen as enforcing an AND gate: $q_{1,2} = q_1 \wedge q_2$. In general, the penalty term $C(q_{1,2}, q_1, q_2)$ will be added to the QUBO cost function with a multiplicative factor λ, that can be reduced or increased. A value too high for a penalty term can interfere with the quantum solver's ability to respect the cost function. On the other hand, a λ value too small lowers the probability for the constraint to be respected.

We underline that expression in Eq. 2.11 is a particular example for the coefficients of the constraint cost $C(q_{1,2}, q_1, q_2)$. Other valid expressions addressing the same objective exist. The form presented in 2.11 is the same originally appearing in the quadratization technique of [250], which was first exploited in the context of quantum annealing by [251] and since then has been used in several applications [252]–[254]. A more complete discussion regarding quadratization techniques (the general approach to reducing the degree of an optimization problem) can be found in [255], who wrote the book having in mind the application to quantum annealing.

As a last remark, we underline that the computational time required to reduce HUBO problems to QUBO problems can be negligible with respect to the time required by the solver to find the solution. Indeed, the reduction from k-local to 2-local interactions is known to scale in polynomial time [256]. Such scaling can be negligible for large problem sizes that require an exponentially scaling cost to be solved. On the other hand, one should be aware that the reduction to QUBO can create problems that are much harder to solve than their original HUBO version [174].

2.2 Optimization problems

In research and industry, hard combinatorial optimization problems are ubiquitous. Any advancement in the performances of modern solvers for such problems has an impact on practically any field where heavy computation is involved.

Many hard optimization problems can be cast in QUBO form [257], so AQC is a potential candidate for revolutionizing many industrially relevant fields, such as pharmaceutics and logistics. In this Section, we make few examples of hard optimization problems that can be tackled by AQCs.

2.2.1 Boolean satisfiability problems

Boolean satisfiability problem (often abbreviated as SAT) is the problem of determining if there exists a combination of values for the input variables of a given Boolean formula that makes the formula true. If at least one of those combinations exists, the formula is said to be satisfiable. SAT was the first problem that was proven to be NP-complete [258]. Due to its simple expression, numerous proofs of NP-completeness for other problems makes use of a reduction from SAT to the problem of interest. Therefore, it would not be crazy to state that SAT is the most important hard problem in the NP class (from a computational complexity point of view).

A particular reduction of SAT is MAX-SAT. MAX stands for maximum, which means this time we are not asking if the statement is satisfiable, but we are asking which is the maximum number of clauses in the formula that can be satisfied. For this purpose, the Boolean formula is required to be written in conjunctive normal form[6].

We will restrict the MAX-SAT problem to those Boolean formulas containing clauses composed by an OR of only two variables. This restriction of the MAX-SAT domain is named MAX 2-SAT problem. The decision version of MAX 2-SAT has been proved NP-complete by a reduction from 3-SAT [259], in contrast with the 2-SAT problem, which can be solved in polynomial time by a deterministic Turing machine.

We will now show that MAX 2-SAT can be mapped in a natural way to QUBO form using a number of binary variables equal to the number of variables appearing in the Boolean formula.

For MAX 2-SAT, each clause is satisfied if either or both variables in the clause are true. Thus, there are three possible types of clauses for this problem: the case in which no variable is negated, the case with one negation, and the case where both variables are negated. These clauses can be represented by the

[6]A Boolean formula is written in conjunctive normal form if it is expressed as an AND of clauses containing only ORs. For example, $(A \lor B) \land (\neg A \lor C) \land (A \lor B \lor \neg D)$ is correctly written in this form.

following QUBO expressions:

$$
\begin{aligned}
(A \lor B) &\rightarrow 1 - q_A - q_B + q_A q_B \\
(A \lor \neg B) &\rightarrow q_B - q_A q_B \\
(\neg A \lor \neg B) &\rightarrow q_A q_B
\end{aligned}
\tag{2.12}
$$

where q_A, q_B are the QUBO variables corresponding to the Boolean variables A, B.

Note that whenever one of the clauses in Eq. 2.12 is true, the value of the corresponding expression on the right is 0, while unsatisfied clauses correspond to expressions of value 1.

2.2.2 Max Cut

Given an undirected graph $G(V, E)$ with a vertex set V and an edge set E, the Max Cut problem seeks to partition V into two sets such that the number of edges between the two sets, is a large as possible. In other words, we are aiming to draw a line that divides the vertices in two sets while crossing the highest possible number of edges. Max Cut is an NP-hard problem. In particular, the decision problem related to Max Cut, which is asking if there exist a partition where at least C edges are cut, is NP-complete. This can be shown, for example, by a reduction from maximum 2-satisfiability [259].

Max-cut is a relevant problem from the mathematical point of view. It is one of the 21 fundamental NP-complete problems listed by Karp in [260]. It also possess real-world applications in very-large-scale-integrated (VLSI) circuit design and in printed circuit board design [261], as well as in studying balance in social networks [262].

To map Max-Cut to a QUBO problem we use a Boolean variable for each node, to represent the set the node belongs to. Then we introduce the following cost function:

$$C(q_1, .., q_N) = \sum_{(i,j) \in E} -q_i - q_j + 2 q_i q_j \tag{2.13}$$

where the summation runs on E, the set of all edges present in the problem graph. The expression for a single edge equals 0 if the two nodes belong to the same set, while it equals -1 if the two nodes belong to different sets. Thus, minimizing $C(q_1, .., q_N)$ is equivalent to looking for the arrangement of the nodes in the two sets that maximizes the number of edges connecting nodes that belong to different sets, which is exactly the solution of the Max-Cut problem.

2.2.3 Semiprime Factorization

The problem of factoring an integer number into its prime factors is of paramount interest in mathematics, in particular in cryptography, since the notorious hardness of such problem [263] is at the basis of many cryptographic protocols on which the modern exchange of information over internet heavily relies, in particular the RSA algorithm [264], [265]. Given N, the number to be factored, one must find the prime factors p and q. The best-known classical algorithm for solving FP is the General Number Field Sieve whose complexity is not polynomial [266]:

$$\mathcal{O}\left(e^{\sqrt[3]{\frac{64}{9}}(\log N)^{1/3}(\log\log N)^{2/3}}\right) \tag{2.14}$$

A considerable amount of the attention received by the quantum computing field in the early twenties is probably due to the seminal paper by Shor, where he introduced an efficient algorithm to factorize semiprimes (numbers that are obtained as the product of two primes) on universal quantum computers [18].

Due to the prototypical state of current universal quantum computers, the RSA protocol seems to have a long way to go before retiring. Researchers have thus tried to tackle factorization using AQC [267]. A seminal work on the topic has been published in 2008 by Peng et al. [268]. They hypothesized a general Hamiltonian to implement adiabatic quantum evolution of a custom system whose final state encodes the two factors. Nonetheless, their Hamiltonian is equivalent to that of an Ising system with local transverse fields and up to 4-local longitudinal interactions, which means their protocol is in principle equivalent to a HUBO problem of degree four. They provided empirical evidence of the proposed approach by factoring the number 21 on a three-qubits NMR quantum processor. Then, Xu et al. [269] used the same approach to factor 143 on a similar quantum processor. Then, Schaller et al. [270] developed a new approach to factorize using AQC based on tables for binary multiplication. Dridi et al. [271] elaborated the same idea, achieving a reduction in the number of ancillary variables needed, and were able to factorize the number 200099. Approximately 990 qubits (n.d.r. visually estimated by the author from a figure in the article) were required to embed the 75 binary variables of the corresponding QUBO problem on the Chimera topology of D-Wave 2X. At that time, 200099 constituted the biggest semiprime factorized on any quantum processor. Shortly after, Li et al. [272] applied both theoretical reductions and Hamiltonian transformations to successfully factor 291311.

Recently, Saida et al. [273] proposed an hardware setup composed by superconducting flux qubits capable of performing AQC on a Hamiltonian natively implementing binary multiplication. They used the novel quantum annealer to perform an experimental 2-bit factorization. In perspective, ad-hoc quantum annealing hardware could constitute a game changer to achieve quantum advantage on specific applications.

In [2], we tested the capabilities of an AQC and a Virtual Memcomputing Machine on a semiprime factorization problem (FP). We will now explain how to formulate FP in ILP and QUBO form, to then be able to solve it via AQC and Memcomputing machines. This introduction is functional for the discussion in Section 4.1.2 of the results we obtained in [2].

FP can be expressed as follows: let M be the number to be factored, one wants to find the prime factors p and q such that

$$M = p \times q \tag{2.15}$$

This problem can be solved as a binary optimization problem. For this, one needs the bit representation of the numbers:

$$M := \sum_{i=0}^{L_M-1} 2^i m_i, \quad p := \sum_{i=0}^{L_p-1} 2^i p_i, \quad q := \sum_{i=0}^{L_q-1} 2^i q_i \tag{2.16}$$

in which $m_i, p_i, q_i \in \{0, 1\}$.
The bit sizes L_p and L_q are unknowns, but one can arbitrarily consider $p \geq q$, from which it follows that $L_q \leq \lceil \frac{L_M}{2} \rceil$. To reduce the problem's variables, it is useful to consider that, since p and q are prime numbers, then $p_0 = q_0 = m_0 = 1$. Moreover, since the problem's complexity is not changed by a priori knowledge of the lengths L_p and L_q, then the most significant bit of p and q can be fixed to 1 if one intends to evaluate the problem's scaling, as in this case. Since VMM solves optimization problems in the ILP formulation, the SAT problem (derived from the equations 2.15 and the equation 2.16) must be converted to a linear version. An ILP formulation of a semiprime factorization problem is derived from the Column-Based Procedure [271]. The Column-Based Procedure, or Multiplication Table Method [274], involves splitting the equation 2.15, written in the binary form, into at most $L_p + L_q + 1$ equations. The factors of equations 2.15, rewritten according to the definitions in equation 2.16, can be grouped by collecting those terms with the same powers of 2 as coefficients. Therefore the final system comprises up to $L_M + 1$ equations:

$$\sum_{j=0}^{L_q-1} q_j p_{i-j} + \sum_{j=1}^{i} c_{i,j} - m_i - \sum_{j=1}^{L_i} 2^{j-i} c_{i,i+j} = 0\ , \quad 0 \leq i \leq L_M \tag{2.17}$$

where i is column index of the corresponding multiplication table $p \times q$ or, in other words, the power of 2 that unites these terms of the i-th equation. The number $L_i = \lceil \log_2 (L_q + i - m_i) \rceil$ is the number of carry variables introduced in the i-th equation.
The carry variables $c_{i,j}$ were added and subtracted from equation 2.15 to set to 0 the sum of the terms with the same power of 2 and thus obtain the system in equation 2.17. The system of equations is not linear due to the presence of terms like $p_i q_j$. These terms correspond to the logical `AND` operation whose equation $z_{ij} \equiv p_i q_j$ is equivalent to the following two linear inequalities:

$$\begin{cases} p_i + q_j \leq z_{i,j} + 1 \\ p_i + q_j \geq 2z_{i,j} \end{cases} \tag{2.18}$$

From equation 2.17 and 2.18, an ILP problem $\mathcal{P}_F$ can be defined as follows:

$$\begin{cases} \sum_{j=0}^{L_q-1} z_{i-j,j} + \sum_{j=1}^{i} c_{i,j} - m_i - \sum_{j=1}^{L_i} 2^{j-i} c_{i,i+j} = 0 \,, \quad 0 \leq i \leq L_M \\ p_i + q_j \leq z_{i,j} + 1 \,, \quad 1 \leq i \leq L_p - 2 \,, \quad 1 \leq j \leq L_q - 2 \\ p_i + q_j \geq 2z_{i,j} \,, \quad 1 \leq i \leq L_p - 2 \,, \quad 1 \leq j \leq L_q - 2 \end{cases} \tag{2.19}$$

The optimization problem $\mathcal{P}_F$ is an ILP problem that does not bear a cost function, therefore a feasible solution is also the problem's solution.
To reduce the number of carry variables and the number of equality constraints, the problem's equations $\mathcal{P}_F$ can be rearranged in order to consider blocks of columns of the multiplication table. This method, called Blocks Multiplication Table Method [275], follows the same procedure of the previously cited Multiplication Table Method. But instead of the system in equation 2.17 (where each equation corresponds to one power of 2), the new equations correspond to a group of power of 2. For the implementations on the VMM, Gurobi, and D-Wave, we used the Blocks Multiplication Table Method with a block size of 2 columns. Therefore the total number of equations in the ILP problem is up to $\lceil \frac{L_M}{2} \rceil$.
The ILP formulation was used for VMM and Gurobi, but cannot be implemented on D-Wave devices. For D-Wave, the ILP problem $\mathcal{P}_F$ was converted into the corresponding QUBO (Quadratic Unconstrained Binary Optimization) formulation, which includes a penalty component that takes into account the constraints. The QUBO cost function for D-Wave is then:

$$\mathcal{P}_F^{\text{QUBO}}(p, q, z, c) = \sum_i H_i^2(p, q, z, c) + \lambda \sum_{i,j} R(p_i, q_j, z_{i,j}) \tag{2.20}$$

where H_i^2 are the square of the equalities in equations 2.19 but grouped by blocks of 2 to support the Blocks Multiplication Table Method. The penalty terms $R_(p_i, q_j, z_{i,j})$ are QUBO terms corresponding to the inequalities:

$$R(p_i, q_j, z_{i,j}) = p_i q_j - 2p_i z_{i,j} - 2q_j z_{i,j} + 3z_{i,j} \; . \tag{2.21}$$

2.2.4 Logistic

Logistic problems deal with the efficient flow of goods or information (routing or travel problems), as well as scheduling the ideal time-sequence for a collection of tasks (scheduling problems). Most of the difficult problems to solve in logistic are combinatorial problems that belong to the NP-hard class. Any improvement in the computational time required to solve this problems has a wide impact on numerous industrial and research fields. Many quantum computing experts have identified in D-Wave processors the ideal QC candidate to solve logistic problems in the near future, since such problems usually involve a large number of variables. One main limitation is the necessity to map a logistic problem to QUBO form. This is usually doable, since combinatorial problems are often expressible as an optimization problem, but the mapping to QUBO can introduce an overhead that potentially hinders the utility of the QC solution.

One of the first examples we can find in literature dates back to 2004, before the actual realization of hardware AQCs. Martoňák et al. [138] proposed a path-integral Monte Carlo quantum annealing scheme implemented on an ising model to solve the symmetric Travelling Salesman Problem (TSP) (Warren [276] will later show how to map TSP specifically to a QUBO model). TSP is an ubiquitous hard optimization problem defined on a graph that requires to find the shortest path across multiple nodes so that all nodes in the graph are visited at least one. In [138] the simulated quantum annealing algorithm achieved performances superior to the classical SA.

More recently, after the advent of prototypical AQCs, literature became abundant of actual experiments on the QPU. As an example, in 2017 Neukart et al. [277] explained how to map a traffic flow optimization problem on D-Wave processors. The objective of this problem is to minimize the time for a given set of cars to travel between their sources and their destinations. The authors were forced to utilize a hybrid framework, due to the limited topology of the D-Wave AQCs at the time. Later, the same group (Yarkoni et al. [278]) extended [277] and managed to navigate a small bus fleet in real time for the duration of the Web Summit 2019 conference in Lisbon. This was the first historical real-world, real-time application of a QC. Anyway, the largest QUBO that was solved was quite limited in size (12 variables, with five buses being navigated simultaneously). Neukart et al. [277] inspired also Clark et al. [279], who applied the methdology to a real time routing problem where multiple robots are moving on a grid. The authors introduced an additional constraint to avoid collisions and tested the algorithm on D-Wave 2000Q, reaching 200 robots moving simultaneously on the grid.

Scheduling and planning problems have received special attention from quantum computing researchers. Rieffel et al. [182] were among the first to apply AQCs to a practical, industrially relevant problem. They focused on planning problems (both travelling and scheduling). Their work is one of the first to underline the importance of an efficient mapping to QUBO and an efficient embedding on the device. If such steps are performed poorly, the AQC will have a hard time in solving the original problem. Recently (2021), Volkswagen partnered with two European universities to optimize the shipment of goods on road via trucks [243]. The goal of the optimization was to minimize the total distance travelled for all trucks transporting shipments. They tested both simple minimization objectives (truck kilometers), and hard constraints (limitations on the truck load). The researchers underlined that finding a valid QUBO representations for the considered problem required considerable efforts.

Another popular benchmark scheduling problem for AQCs is the Job Shop Scheduling Problem (JSSP). JSSP is NP-hard, and it is considered one of the most difficult combinatorial optimization problems. It is described by a set of jobs that must be scheduled on a set of machines. Each job is composed by a sequence of operations to be executed in a predefined order. Each operation takes a particular time and necessitates of a particular machine. To map JSSP on AQCs, the researchers have resorted to decompose the problem in smaller subproblems [280], since JSSP can involve a large number of variables. Denkena at al. [281] extended previous works implementing the more general Flexible JSSP, where each operation is allowed to be executed on various machines. They obtained encouraging results when compared to the classic literature. Recently (2022) Carugno et al. [282] extended previous works by describing each step required from the problem formulation to the fine-tuning of the quantum annealer. This evolved approach is a consequence of the scientific community realizing that tuning the internal parameters of the AQC allows to sensibly enhance the performances, as we discuss in Chapter 3. Another interesting approach to a scheduling problem can be found in Ikeda et al. [283], who applied AQC to the nurse scheduling problem. They were able to increase the quality of the solutions by using a modification of the annealing schedule known as *reverse annealing*[7].

2.3 Introduction to Sampling problems

In Section 1.3.6 we underlined that the annealing dynamics usually "freezes" at a certain time $t_{\text{freeze}} < t_{\text{ann}}$, after which the quantum effects stop, and the system is mainly affected by thermal jumps. Section 1.3.6 explained how such effect is expected to cause readout samples to be distributed according to the Boltzmann distribution (Eq. 1.86). As discussed in Section 1.2.1, AQCs have been conceived and designed to solve optimization problems cast in QUBO form, but the freeze-out phenomenon is one of the main obstacles that hinder AQCS' performance. Nonetheless, at least since 2011 [284] researchers have tried to exploit this otherwise annoying behaviour for sampling purposes.Indeed, drawing samples distributed according to the Boltzmann distribution is a hard computational task [285], and existing approaches are mainly approximate heuristic samplers based on Markov chain Monte Carlo (MCMC) procedures [286]–[288]. The Boltzmann distribution is relevant in many areas of science, and sampling from such distribution can become a significant bottleneck in many applications. Examples of such hard tasks are simulating dynamical systems [289], training probabilistic unsupervised models such as the Boltzmann Machine [290], and, more generally, approximate counting and inference of marginal probabilities, which are often NP-hard tasks [291], [292].

From an historical point of view, Denil et al. [284] and then Vinci et al. [293] were among the firsts to perform an experiment on an AQC while being interested in the samples distribution, rather than focusing solely on the probability of achieving the ground state. Later, literature began populating with several new works that tested the actual capabilites of AQCs when used as Boltzmann samplers [220], [294], [295]. In particular, [220] performed one of the first experimental demonstrations that AQCs sample from the Boltzmann distribution. During time, certain limitations have also been underlined, such as the detrimental consequences of the embedding process on the shape of the sampled distribution [166], or the difficulty in estimating and tuning the sampling temperature (see next Section 2.3 and Section 3.1.6).

In the next Section we explain how to tune the temperature at which the AQC samples from the Boltzmann distribution. Such technique is useful in both the main sampling applications for AQCs, which are statistical mechanics

[7]See Section 3.1.2.

systems and the Boltzmann Machine, respectively. You can find a brief literature review on how to estimate the sampling temperature of AQCs in Section 3.1.6 in the next Chapter.

Tuning the sampling temperature

Starting from the expected distribution of quantum states at the freezing point in Equation 1.86 we can conclude that the expected probability to sample a state s after the measurement operation is

$$P(s, T_{\text{eff}}) = \frac{e^{\frac{-E(s)}{T_{\text{eff}}}}}{Z(T_{\text{eff}})} , \tag{2.22}$$

where s is a specific classical configuration of the qubits, the adimensional functional $E(s)$ is the energy of the state divided by the typical energy scale of the magnetic couplers, T_{eff} is an adimensional effective temperature obtained by dividing $k_B T$ for the same typical energy scale, and $Z(T_{\text{eff}}) = \sum_{s \in S} e^{-E(s)/T_{\text{eff}}}$ normalizes the probability distribution, and it can be interpreted as the partition function of the final distribution of the states over the set of possible configurations S. By using a dimensionless scale for both the energy $E(s)$ and the effective temperature T_{eff} we are following a popular convention that stems from the simplicity of identifying $E(s)$ as the energy of the Ising or QUBO problem[8] we want to solve using the AQC. As an example, in [221] the authors estimated that, for the DW2X AQC, an adimensional coupling $J = 1.0$ in an Ising problem corresponded to 7.9GHz when realized in the hardware. In this example, the typical energy scale of the annealing process was 7.9GHz. This constant can vary depending on the considered hardware. They estimated that the physical temperature of the DW2X AQC $T_{\text{DW2X}} = 12.5\text{mK}$ corresponds to $T_{\text{DW2X}} = 0.033$ in the same adimensional scale. Actually, the effective sampling temperature T_{eff} is challenging to estimate[9], but it is far more difficult to forecast it before the sampling, since it can simultaneously depend on the specific problem instance, on the noise on the programmable parameters, on the physical temperature of the device, and on the specific annealing schedule employed. For this reasons, estimating T_{eff} a priori has been dubbed "a daunting task" [221]. Typical values found experimentally tend to be around an order of magnitude

[8]QUBO problems (whose variables assume values in $\{0, 1\}$) can be cast into equivalent Ising problems (with variables in $\{-1, +1\}$) by means of a simple mapping that preserves the energy of every configuration, apart from a constant shift. This shift is irrelevant in sampling applications, since it get simply reabsorbed from all probabilities and normalized by the partition function.

[9]See Section 3.1.6 for a literature review about the available methodologies.

higher than the estimate $T_{\text{DW2X}} = 0.033$ for the DW2X QPU. In [221] authors registered effective temperatures around 0.1, which was later confirmed by [4] in a similar implementation on the more recent D-Wave 2000Q AQC. When using embedding comprising longer chains, the effective temperature is observed to raise, probably as a consequence of the increased height of the energy barrier, consequently causing an earlier freeze out event [219], [296]. As an example, in [4] we observed an effective temperature increase from 0.11 to 0.32 when going from 1-qubit to 4-qubits chains.

When passing a problem to the QPU we can thus expect to measure samples distributed according to the Boltzmann distribution at an effective temperature T_{eff} that is difficult to estimate. Nonetheless, we have a certain degree of control over such temperature. Indeed, suppose we submit to the AQC a slightly modified version of our Ising problem where each weight and bias is rescaled by a constant α, such that we have a new problem Hamiltonian

$$\begin{aligned} H'_P = \sum_{i,j} J'_{ij}\sigma_i^z\sigma_j^z + \sum_i h'_i\sigma_i^z \ , \\ \text{where} \quad J'_{ij} = \alpha J_{ij} \quad \text{and} \quad h'_i = \alpha h_i \ . \end{aligned} \tag{2.23}$$

It is obvious that the parameter α can be factored out so that $H_P = \alpha H'_P$. As a consequence, at the end of the annealing we will have a new, different probability of sampling each state s:

$$\begin{aligned} P'(s, T_{\text{eff}}) = \frac{e^{\frac{-\alpha E(s)}{T_{\text{eff}}}}}{Z'(T_{\text{eff}})} = \\ = \frac{e^{\frac{-E(s)}{T'_{\text{eff}}}}}{Z(T'_{\text{eff}})} \ , \end{aligned} \tag{2.24}$$

where $T'_{\text{eff}} = T_{\text{eff}}/\alpha$. By rescaling the weights and biases we are thus able to simulate the extraction of the samples at a different effective temperature, which will be reduced as we increase α. For classical solvers, the possibility of tuning the sampling temperature by rescaling the problem weights implies that the temperature can be always set to 1 without loss of generality. For AQCs this is not the case. Indeed, all AQCs have both lower and upper limitations on the possible absolute values for the couplings and the biases. From above, such parameters are bounded by the maximum values realizable in the hardware. From below, the limitation comes from the quantization step of on-QPU digital-analog converters (DACs), that randomly bias the actual values of such parameters [297]. As an example, the most recent D-Wave prototype Advantage2_1.1 can implement couplings in $[-1, 1]$ and biases in $[-4, 4]$, while the typical DAC errors are of the order of $\sim$0.001 on both the couplings and the biases [297]. While sampling from an AQC we are usually concerned with the upper limit, because it limits the maximum value for α we can choose, which in turns limits the lowest T_{eff} we can sample at. An additional limitation can come from the use of strong ferromagnetic couplings in the embedding, which are supposed to be higher than the typical problem coupling, but still need to be lower than the maximum allowed coupling value, further restricting the accessible values for the problem couplings. For instance, in [4] we set $\alpha = 0.32$ to sample at $T_{\text{eff}} = 1$ from a bipartite graph of 16 by 16 units that was embedded using four qubits for each unit. We were thus able to sample at $T_{\text{eff}} = 1$ by simply reducing our model weights and biases roughly by a third, keeping well above the DAC quantization errors.

2.4 Boltzmann Machines

Boltzmann Machines (BMs) are a prominent example of unsupervised learning algorithm [298], [299], introduced in 1985 by Ackley, Hinton and Sejnowski [300]. It has been proved that a trained BM can be used as a universal approximator of probability distributions on binary variables [301], [302]. Such property makes BMs particularly suitable if used as a generative model [303], [304] to reconstruct partially missing data. Despite the theoretical representative power, one of the steps of the training algorithm makes it computationally expensive to train large BMs [305]. More specifically, such a step requires to extract samples from the instantaneous distribution approximated by the BM. The cost to produce each sample grows rapidly as the problem size increases [285]. Thus, BMs are usually not applicable in useful–sized problems. Being able to train large BMs on an AQC would enable to tackle and solve relevant problems in many fields ranging from recommendation systems [306], to anomaly detection [307],[308], to quantum tomography [309].

During the years, BMs have found application mostly in a simplified form, called Restricted Boltzmann Machine (RBM) [310], [305], where the model network is characterized by a bipartite graph, as opposed to the fully-connected graph of general BMs. RBMs have lower representative power than BMs, but they are easier to train and use, despite remaining an expensive algorithm [285]. Due to their popularity in the classical version, RBMs are usually chosen as a benchmark to evaluate the potential advantages of the quantum learning algorithms [221].

Since 2011, researchers explored the possibility to use an AQC to approximate the probability distribution needed during the training of a RBM [284], [150]. Experimental results suggest that an AQC can be operated to extract samples from the distribution associated with a RBM, at the computational cost of a single quantum operation [219], [100], [296]. Thus, sampling from AQCs seems to have a cost that does not depend on the size or the complexity of the RBM. In 2015, Adachi and Henderson [311] performed computations on an actual AQC to train a Deep Belief Network, a directed graph version of RBM (and tested it on MNIST [312]). In 2016, Benedetti et al. [221] introduced an efficient technique for estimating the effective temperature of the distribution produced by the AQC. Several papers have also approached the problem of training a fully connected BM [296], [313]–[315], usually limiting the model to having only visible units, limiting the potential representative power of the model. Recently, a general, fully-connected BM comprising hidden variables has been successfully trained by our team [23], [24] and another team [316].

Quantum-trained BMs have been tested on a variety of applications, ranging from cybersecurity [317], [318], to cancer detection [319], classification of neutrino detection data [320], and training Generative Adversarial Networks [321], [322][10].

A quantum computational advantage is believed to be close for RBMs trained on AQCs [324], and experimental data obtained by our team suggest a limited quantum advantage is already achievable for both RBMs and general BMs [23], [24], [318]. Recently, the legacy hypothesis that AQCs produce samples distributed as those obtainable by classical techniques has been challenged. Quantum annealing seems to explore the local minima in the energy landscape of the RBM energy functional in a better way, if compared to MCMC approaches [325], lowering the classification error when used in the exploitation phase of a trained RBM. Finally, we consider worth noticing that other non-quantum approaches such as the implemention of RBMs on FPGA has also shown an asymptotic advantage similar to that promised by quantum computers [326].

In this Section we will first introduce the Boltzmann Machine model and explain how it can be trained using classical computers. Then, we will explain how to train both restricted and fully-connected Boltzmann Machines using an AQC.

[10]We recommend reading [323] for further examples of possible applications.

2.4.1 Theory of the classical Boltzmann Machine

Before diving deep in the following detailed sections, we start with a brief introductory overview of the BM functioning. Every concept presented in this introduction will be explained in detail later.

The Boltzmann machine (BM) is a kind of unsupervised learning algorithm that is particularly powerful if used as a generative method. It is a network composed of two layers of neurons: one receives inputs and produces outputs (*visible units*), and the other enhances the representational power of the model (*hidden units*) (Fig. 2.1). Each visible unit corresponds to a specific bit of information in the input data, such as a pixel in a black and white image. A BM learns to mimic the conditional probability distributions that connect the various bits of information in input data, which means it can infer the values of missing bits in a new, partially corrupted input. The training phase requires loading a sample from the dataset into the visible units, and then update all units (both visible and hidden) iteratively, to achieve thermalization of the system. Indeed, an energy functional associates each possible configuration of the units to an energy value, and the thermalization is conducted so to produce final configurations of the units distributed according to the Boltzmann distribution. This makes the BM a probabilistic model, since the final "answer" of the machine corresponds to the value appearing on the visible units after several thermalization steps. During training, this answer is confronted with the input, and then weights and biases are updated with the aim to reduce the energy of those configurations whose visible units correspond to the training data. This is done to make such configurations more likely to be produced by the BM, since a lower energy corresponds to a higher probability of being extracted from the associated Boltzmann distribution. Thus, a trained BM is defined by specific values for weights and biases that correspond to an internal abstract representation of the training data. After being trained, a BM is likely to produce configurations where the value of the visible units is sampled according to the abstract conditional probability distribution that the BM has inferred from the training data. When exploiting a trained BM, known input information is loaded into the corresponding visible units, which will be fixed during the thermalization phase, while visible units corresponding to unknown information will be randomly initialized and left free to thermalize. The BM will thus reconstruct the most probable values for the free units, based on the value of the fixed ones.

In this Section, we present the structure, functioning, and training of the BM.

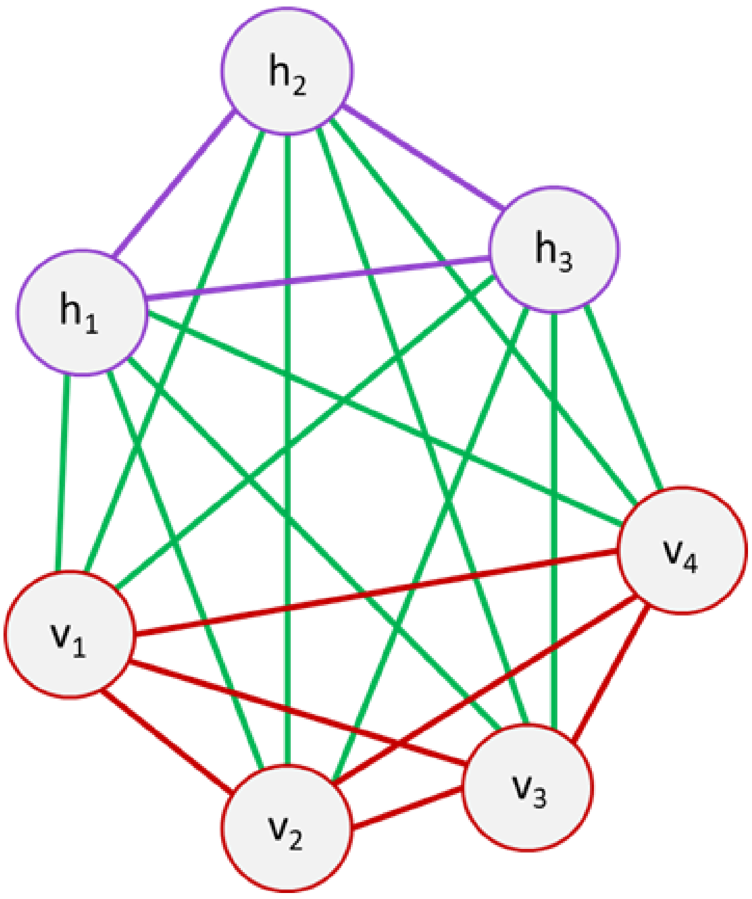

FIGURE 2.1: An example of a fully-connected BM. Each unit belongs to the set h (internal or hidden units) or v (external or visible units). Lines represent the edges, which can be inter-layer (red, purple) or connecting the two different layers (green).

Structure of the Network

The BM is composed of primitive computing elements called units that are connected by undirected links [300]. Each link has an associated a real-valued weight, and each unit had an associated real-valued bias. A unit is always in one of two states, 1 or 0. The value adopted is a probabilistic function of the value of the neighboring units, which also depends on the weights of the corresponding links and on the bias of the considered unit. The weight on a link represents the tendency of two connected units to behave in a similar or opposite fashion. The bias on a unit represents the tendency of the unit to prefer 0 or 1 values. The units are arranged into two layers, named *visible* and *hidden* layers. Units in the same layer can be connected, as it happens in a general BM, or not connected. In this latter case, the network is effectively represented by a bipartite graph with only inter-layer connections, and the model is called a Restricted Boltzmann Machine (RBM). AS we will, see, an increased connectivity corresponds to an increase in the computational cost required to train and exploit the model.

Energy-based model

Each global state of the units in the network can be assigned a single number called the *energy* of that state. Under the appropriate assumptions, the individual units can be made to act to minimize the global energy. If some of the units are externally forced or *clamped* into particular states to represent a particular input, the system will then find a low energy configuration that is compatible with that input. The lower the energy of a configuration, the higher the probability that the BM will produce that configuration as an output. Therefore, reducing the energy of a configuration corresponds to the BM memorizing that configuration.

We define the energy of the system E as:

$$\begin{aligned} E\big(\{v\},\{h\}\big) = & -\sum_{i=1}^{n_v}\sum_{j=1}^{n_h} w_{ij} v_i h_j - \sum_{i=1}^{n_v}\sum_{j=1}^{n_v} w^v_{ij} v_i v_j - \sum_{i=1}^{n_h}\sum_{j=1}^{n_h} w^h_{ij} h_i h_j \\ & -\sum_{i=1}^{n_v} a_i v_i - \sum_{j=1}^{n_h} b_j h_j \ , \end{aligned} \tag{2.25}$$

where w_{ij} is the weight of the connection between the visible unit i and the hidden unit j, w^v_{ij} and w^h_{ij} are the weights of the visible-visible and hidden-hidden connections, respectively, v_i and h_j represent the value of the i-th visible unit and the j-th hidden unit, respectively, n_v is the total number of visible units, and n_h is the total number of hidden units. Finally, a_i and b_j are the units biases, which appear in the energy expression multiplying visible and hidden units, respectively.

In the special case of a RBM, any weight w^v_{ij} and w^h_{ij} must be set to zero. A representation of such a graph is depicted in Fig. 2.2.

The visible units will be the ones fed with the dataset we want to train the network with, and the hidden units will represent the latent factors that the network uses to interpret the input. Visible units have a double role since they also constitute the output of the model. Indeed, the BM is applied for instance when we have an element that is partially corrupted. The available information about the element (input) is loaded on some of the visible units, while the missing information (output) is produced by the values assumed by the remaining visible units.

For a RBM the energy expression becomes:

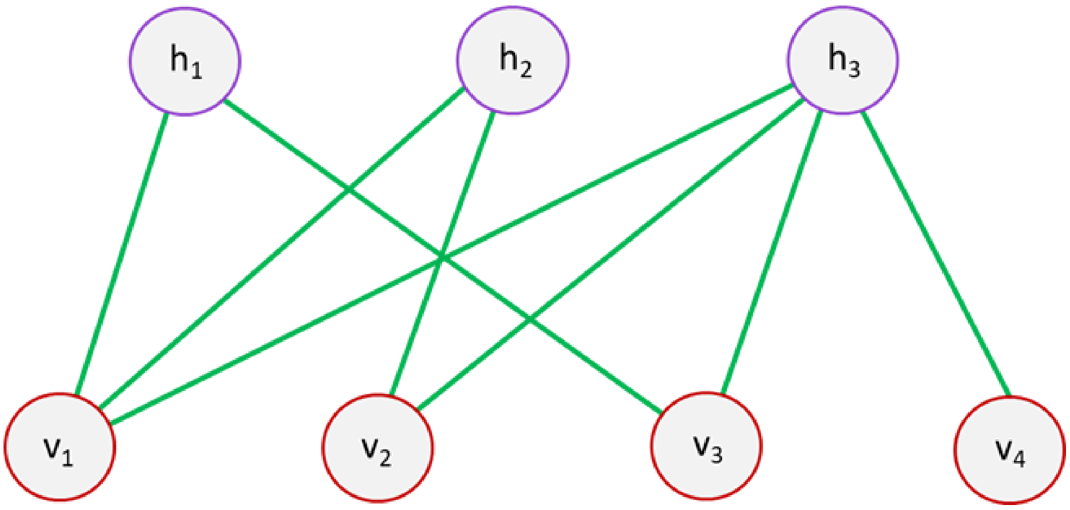

FIGURE 2.2: An example of RBM. Green lines represent the edges. Each unit belongs to the set h or v, and there are no edges connecting two units from the same set.

$$E\big(\{v\},\{h\}\big) = -\sum_{i=1}^{n_v}\sum_{j=1}^{n_h} w_{ij}v_i h_j - \sum_{i=1}^{n_v} a_i v_i - \sum_{j=1}^{n_h} b_j h_j \ . \tag{2.26}$$

Minimizing Energy

A simple algorithm for finding a combination of unit values that is a local minimum consists of switching each unit into whichever of its two states yields the lower total energy, given the current states of the other units. If hardware units make their decisions asynchronously, and if transmission times are negligible, then the system always settles into a local energy minimum. If a Boltzmann machine network is equipped with such an update rule, it should be called a Hopfield network [327]. Hopfield networks are simple models where the energy is minimized via local updates which always move toward lower energies. Thus, getting stuck in local minima is the intended scope of Hopfield networks, because the local energy minima of this network are used to store training data[11]. Being able to reconstruct an image seen in the past can be useful, but it is far more interesting to create an algorithm that can reconstruct images it has never seen before.

Finding a way to implement generalization is not obvious at all. Ackley, Hinton, and Sejnowski [300] managed to do so by allowing the algorithm to

[11] A Hopfield network can be used to recover from a distorted input to the trained state that is most similar to that input. The network is then said to have *associative memory*, because it recovers memories on the basis of similarity.

occasionally make updates that do not lower the energy of the model. They chose an update rule that sets a (visible or hidden) unit s_k equal to 1 with probability:

$$\begin{aligned} P(s_k = 1) &= \frac{e^{-E(\{s_i\}_{i\neq k}, s_k=1)/T}}{e^{-E(\{s_i\}_{i\neq k}, s_k=1)/T} + e^{-E(\{s_i\}_{i\neq k}, s_k=0)/T}} \\ &= \frac{1}{1 + e^{\Delta E_k / T}} \ , \end{aligned} \tag{2.27}$$

where

$$\Delta E_k = E(\{s_i\}_{i\neq k}, s_k = 1) - E(\{s_i\}_{i\neq k}, s_k = 0) \ , \tag{2.28}$$

while $T > 0$ is a parameter that we will call *temperature* (Fig. 2.3 shows its effect). This rule implies that sometimes the jump will not happen even if $\Delta E_k < 0$ (which means the jump would have lowered the energy), and sometimes the jump will happen even if this means raising the energy of the system. We have thus added *thermal noise* to the system, allowing it to eventually reach *thermal equilibrium*. Indeed, the update probability expressed by Eq. 2.27 is exactly the probability that $s_k = 1$ if the system is thought of as a statistical system with energy E from Eq. 2.25 and with temperature T. The probability for a configuration $(\{v\}, \{h\})$ to be generated by such an update rule follows the Boltzmann distribution:

$$P(\{v\}, \{h\}) = \frac{1}{Z(T)} e^{-E(\{v\},\{h\})/T} \ , \tag{2.29}$$

where the partition function $Z(T)$ is given by summing over all possible pairs of visible and hidden vectors:

$$Z(T) = \sum_{\{v\},\{h\}} e^{-E(\{v\},\{h\})/T} \tag{2.30}$$

BMs are trained and exploited at a finite temperature different from zero. It can be seen from Eq. 2.27 that a rescaling in T can be reabsorbed by rescaling weights and biases in the expression of the energy E (Eq. 2.26). Since weights and biases values can be freely modified during training, the absolute value of the temperature chosen has no meaning, and it is usually set $T = 1$. From now on, we will follow this convention.

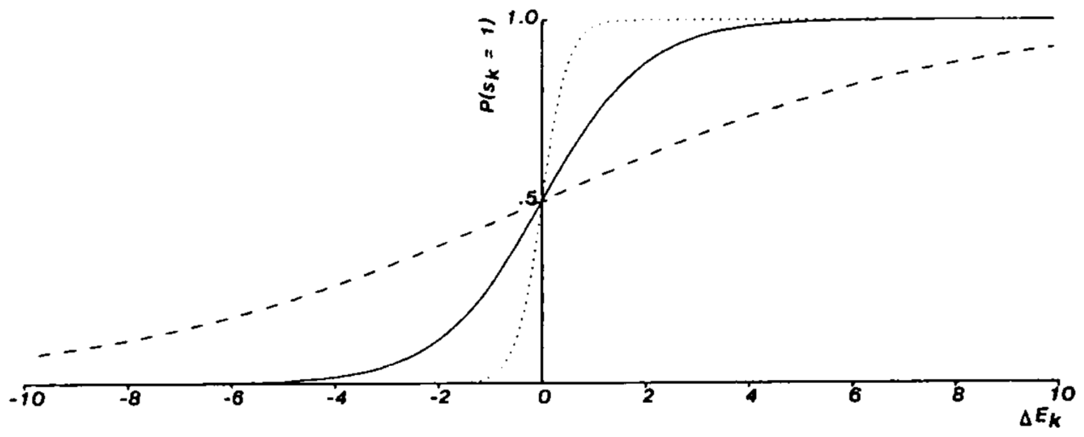

FIGURE 2.3: $P(s_k = 1)$ (Eq. 2.27) at $T = 1.0$ (solid), $T = 4.0$ (dashed), and $T = 0.25$ (dotted). Image from Ref. [300]

A learning algorithm

Consider a training set composed of binary vectors of length l. We build an RBM with $n_v = l$ visible units. We can make the visible units correspond one-to-one to the elements of any given training vector. In this context, learning means modifying the weights of the network to maximize the likelihood of the Boltzmann machine reproducing the given training vectors on the visible units, where the likelihood is defined as

$$L_{\text{av}} = \frac{1}{N_{\mathcal{D}}} \sum_{\mathbf{v} \in \mathcal{D}} \log \left(\sum_{\{\mathbf{h}\}} P(\mathbf{v}, \mathbf{h}) \right) , \tag{2.31}$$

where $P(\mathbf{v}, \mathbf{h})$ can be evaluated numerically as it appears in Equation 2.29, with the hypothesis that $T = 1$.

According to the Boltzmann distribution (Eq. 2.29), the network assigns to a visible vector $\mathbf{v}$ a probability given by summing over all possible hidden vectors.

$$p(\mathbf{v}) = \frac{1}{Z} \sum_{\{h\}} e^{-E(\mathbf{v},\{h\})} , \tag{2.32}$$

where the sum is to be performed over any combination of the values for the hidden units. The probability that the network assigns to a training vector $\mathbf{r}$ can be raised by adjusting the weights and biases to lower the energy of that vector and to raise the energy of other vectors, especially those that have low energies and therefore make a big contribution to the partition function. The derivative of the log probability of a training vector with respect to a weight is fairly simple

$$\begin{aligned}\frac{\partial \log p(\mathbf{r})}{\partial w_{ij}} &= \frac{\sum_{\{h\}} r_i h_j e^{-E(\{h\},\mathbf{r})}}{\sum_{\{h\}} e^{-E(\{h\},\mathbf{r})}} - \frac{\sum_{\{v\},\{h\}} v_i h_j e^{-E(\{v\},\{h\})}}{\sum_{\{v\},\{h\}} e^{-E(\{v\},\{h\})}} \\ &\equiv \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model}\end{aligned} \tag{2.33}$$

where the angle brackets are used to denote expectations under the distribution specified by the subscript that follows. The term $\langle v_i h_j \rangle_{data}$ is sometimes called *positive statistics*. It represents the expectation value of the product $v_i h_j$ when the visible units are set equal to the elements of vector $\mathbf{r}$. The summation is performed over any possible combination of the hidden units (i.e. 2^{n_h} elements, where n_h is the number of hidden units), while each v_i is set equal to the i-th elements of the vector of the dataset $\mathbf{r}$.

On the other hand, $\langle v_i h_j \rangle_{model}$ is called *negative statistics*, and represents the expectation value of $v_i h_j$ when both visible and hidden units are sampled by the model. It means that the summation is performed over any combination of visible and hidden units. It sums up to $2^{(n_v+n_h)} = 2^N$ elements, where N is the total number of units.

In order to train the BM, we need to be able to increase $p(\mathbf{v})$ for those vectors corresponding to training data. The learning rule for performing stochastic steepest ascent in the log probability of the training data is:

$$\Delta w_{ij} \equiv \eta(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model}) \ , \tag{2.34}$$

where $\eta > 0$ is a parameter called *learning rate*. Such a parameter regulates how much the weights are modified in each update. η is usually big at the beginning of the learning process (i.e. weights change more during such first phase), while it is lowered in the final stages (i.e. when the algorithm must finely refine its ability to interpret input).

For what concern biases, they are updated as follows:

$$\begin{aligned}\Delta a_i \equiv& \frac{\partial \log p(\mathbf{r})}{\partial a_i} = \frac{\sum_{\{h\}} r_i e^{E(\{h\},\mathbf{r})}}{\sum_{\{h\}} e^{E(\{h\},\mathbf{r})}} - \frac{\sum_{\{v\},\{h\}} v_i e^{-E(\{v\},\{h\})}}{\sum_{\{v\},\{h\}} e^{-E(\{v\},\{h\})}} \\ \equiv& r_i - \langle v_i \rangle_{\text{model}}\end{aligned} \tag{2.35}$$

and

$$\begin{aligned}\Delta b_j \equiv& \frac{\partial \log p(\mathbf{r})}{\partial b_j} = \frac{\sum_{\{h\}} h_j e^{E(\{h\},\mathbf{r})}}{\sum_{\{h\}} e^{E(\{h\},\mathbf{r})}} - \frac{\sum_{\{v\},\{h\}} h_j e^{-E(\{v\},\{h\})}}{\sum_{\{v\},\{h\}} e^{-E(\{v\},\{h\})}} \\ \equiv& \langle h_j \rangle_{\text{data}} - \langle h_j \rangle_{\text{model}} \ .\end{aligned} \tag{2.36}$$

Estimating the expectation values in the previous Equations requires averaging samples obtained after the system has been conducted to thermal equilibrium. This can be done fairly easily in the case of a RBM. Indeed, as explained in Appendix D, estimating the positive statistics for a RBM is straightforward, while estimating the negative statistics is harder and requires several iterations of *Gibbs sampling*, a method inspired by the Metropolis algorithm [111] used to generate samples according to the Boltzmann distribution. The iterative Gibbs sampling approach is made easier thanks to the bipartite nature of the RBM network, but remains a hard step [285]. See Appendix D for the full estimation procedure for the RBM. In the general, fully-connected BM case, things get worse, and almost no trick can be used to simplify the computation. Thermalizing the fully-connected network requires a synchronous or asynchronous update of each individual unit according to Eq. 2.27, namely the conditional probability distribution inherited from the target Boltzmann distribution. This had to be done for all units at least hundreds of times also for small cases of few dozens units [24].

Exploiting a BM

A trained BM is exploited by asking it to reconstruct an incomplete element of the dataset. The values of n_v^k visible units are known, while the machine should output a value for each of the remaining n_v^u unknown visible units, where $n_v^k + n_v^u = n_v$, the total number of visible units. The input is an array $\mathbf{z}$ with n_v^k elements. The answer of the RBM is an array $\mathbf{r}$ with n_v^u elements, extracted from a conditional Boltzmann distribution such that

$$P(\mathbf{r}) = \frac{\sum_{\{h\}} e^{-E(\mathbf{r},\mathbf{z},\{h\})}}{\sum_{\{h\},\{v\in U\}} e^{-E(\{v\in U\},\mathbf{z},\{h\})}} \tag{2.37}$$

where U is the set of unknown visible units.

Calculating the exact probability of each possible answer is still exponentially hard in the number of hidden and unknown units. We can approximately sample from the distribution in Eq. 2.37 after an iterative thermalization. More steps result in a more precise sampling. Fig. 2.4 shows the sketch of how a

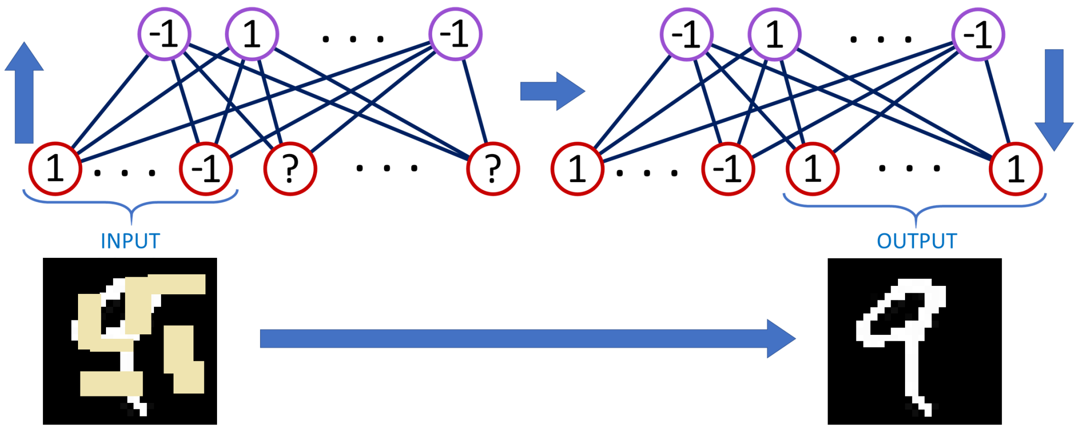

FIGURE 2.4: Sketch of the exploitation of a trained RBM. At the first step, a partially corrupted image is shown to the RBM. Hidden units are updated according to Eq. D.1. At the second step, the visible units corresponding to missing pixels are updated, according to Eq. D.2.

(trained) RBM is exploited. In the first step, an incomplete image is shown to the RBM. Each visible unit corresponds to a pixel in the image. Each known pixel sets the value of the corresponding visible unit (1 for a white pixel, 0 for black one). Next, the hidden units are updated according to Eq. D.1 from Appendix D.1. In the second step, the visible units corresponding to missing pixels are updated, according to Eq. D.2. Usually, many steps of such procedure are performed, before considering the visible units adequately approximating the final output. Note that visible units corresponding to known pixels are held fixed, and are not involved during the update.

Limits of the classical method

The training method for BMs outlined in this Section is only roughly approximating the gradient of the log probability of the training data. The learning rule is much more closely approximating the gradient of another objective function called the *Contrastive Divergence* [305] which is the difference between two Kullback-Leibler divergences[12], but it ignores one term in this objective function so it is not following that gradient. Indeed, Sutskever and Tieleman have shown that it is not following the gradient of any function [328]. Nevertheless, it works well enough to achieve success in many significant applications. We

[12]Kullback-Leibler divergence is an asymmetric measurement of how two probability distributions, P and Q, are different. The Kullback-Leibler divergence of Q from P is defined as $D_{KL}(P \parallel Q) = \sum_i P(i) \log_2 \left(\frac{P(i)}{Q(i)}\right)$. It can be seen as the amount of information that is lost when we approximate P as Q.

will refer to this learning method for estimating $\partial \log p(v)/\partial w_{ij}$ as *contrastive divergence*.

Apart from the approximation described above, the training procedure of BMs requires a computational cost that is still expensive also in the restricted case. When scaling a RBM to a high number of units, we have to perform a high number of Gibbs sampling iterations, slowing down the learning process. The increase in the number of steps comes from the fact that a larger network requires more time to reach a thermalized state. This is especially true if one wants to train a fully connected BM. In such a case, the cost to produce correctly distributed samples grows exponentially with the number of units. As a consequence, BMs have not found any useful application in real life-sized problems.

As opposed to this, RBMs have been applied with success, e.g. in the case of the Netflix Prize [329]. Nonetheless, the approximations made during learning and the high computational cost makes it difficult for RBMs to become a useful model in large problems.

The next Section explains how to use AQCs to relieve the computational burden of estimating the required statistics. This promising approach could realize a breakthrough in training methods for BMs in general, allowing us to realize quantum-trained BMs that overcome the classical limitations.

2.4.2 Boltzmann Machine training on an AQC

Section 2.3 explained that AQCs tend to produce samples distributed according to the Boltzmann distribution. Section 2.4.1 presented the Boltzmann Machine (BM) model and underlined that the main computational bottleneck of the training procedure consist of producing an ensamble of configurations that are distributed according to the Boltzmann distribution. It is intuitive how these two results can be combined to gain a computational advantage. Producing a correctly distributed sample on the AQC requires a single annealing cycle with a fixed duration that does not depend on the model size. On the other hand, classical methods to thermalize the model require an exponential time in the number of units.

The procedure to train a BM using an AQC is simple, relative to the classical approach. Both positive and negative statistics can be estimated by sampling from the model, after embedding it on the QPU. In the case of the negative statistics, where all units are free to evolve, the whole fully-connected graph with N units must be embedded. In the positive case only the hidden units are free to evolve. For RBMs, this means we can simply calculate their expectation values as in the classical case (see Appendix D). For general BMs, it means we have to embed a fully-connected graph with n_h units. After collecting the samples, we can use them to estimate the expectation values in the two statistics, as we would do in the classical case. Contrary to implementations on universal gate-based quantum computers [330], BM training on AQCs appears intuitive and natural, promising exciting advantages [331]. One important caveat regards the sampling temperature. As discussed in Section 2.3, the effective temperature can be tuned by properly rescaling the problem weights and biases. In general it is possible to rescale them such as the adimensional effective temperature becomes equal to 1, as has been done, for instance, in [221] and in our works [3], [4], [23], [24].

Looking at the computational cost of producing a single sample, the AQC achieves a $O(1)$ cost, since the time required is the chosen annealing time plus constant factors such as the readout and delay times[13]. In the classical case, the computational cost is similar to $O(e^N)$, where N is the total number of units. Then, we could expect to achieve an exponential advantage when using AQCs to train BMs. Unfortunately, an often overlooked difference between the two methods can reduce this advantage. Indeed, the measure operation performed in AQCs at the end of the adiabatic evolution is bounded to produce digital values for the units. On the other hand, classical computers can iteratively update units while retaining information regarding their current state in a continuous variable. As explained by Hinton [332], while a binary sampling could be considered closer to the mathematical model of an RBM, using the continuous probability value for hidden variables is usually preferable at a classical level because it reduces sampling noise, thus allowing faster learning. A huge difference comes from the final sample, which can also be expressed with an array of continuous values representing the expectation values for each unit. This means that each classical sample retain more information than a quantum sample. As an example, in [24] we managed to obtain a successful learning for a fully connected classically-trained BM using 256 samples per epoch. To successfully train the same BM using an AQC we were forced to use 4018 samples per epoch.

Besides this difference, we can still expect to get an advantage from using AQCs to train BMs. An advantage has indeed been achieved at least two times recently, in two articles from our team, namely [24], [318] (See Chapter 4 for further details).

[13]See Section 1.3.5.

2.5 Other unconventional computing paradigms

As discussed in Section 1.1, the 21-th century is characterized by a global effort of the scientific community to overcome the emerging limitations of the classical Von Neumann-Zuse paradigm of computation by finding alternative methods to process information and solve hard computational problems. This Section introduces the most promising competitors of Adiabatic Quantum Computers that are based on (or inspired by) non-classical computational paradigms. The first Section 2.5.1 introduces the concept of Ising machines and presents the main examples, namely the NTT Coherent Ising Machine, the Fujitsu Digital Annealer, and the Toshiba Simulated Bifurcation Machine. Then, the Memcomputing Machine is introduced in Section 2.5.2, together with its simulated version. The simulated Memcomputing Machine has been used to produce part of the experimental results presented in Chapter 4.

2.5.1 Ising machines

One of the most thoroughly studied physical systems in history is the Ising model, introduced in Sections 1.2.2 and 1.2.4 in its classical and quantum versions, respectively. We learned in Section 1.2.10 how to map QUBO problems on a quantum Ising system subject to a transverse field, to then obtain the solution via an adiabatic evolution of the system. Lastly, 2.1 presented several examples of industrially or mathematically relevant problems that can be mapped to QUBO form. This pipeline has been presented to justify the adoption of AQCs as solvers for optimization problems, but the same rationale encourages experimenting on different ways to minimize the energy functional of an Ising model, which is known as the *Ising problem.* Optimization problems solvers designed to solve Ising problems are collectively called *Ising machines* [333]. Such solvers are of great interest to the computational community, since, as extensively discussed in Section 1.2.1, Ising problems fall in the NP-hard category in computational complexity [88].

This Section presents the most mature Ising machines available at the present moment, namely the Coherent Ising Machine developed by NTT, the Digital Annealer developed by Fujitsu, and the Simulated Bifurcation Machine developed by Toshiba.

The Coherent Ising Machine

While searching for suitable systems to represent and solve Ising problems with, a team of researchers at the National Institute of Informatics in Tokyo, Japan, became particularly interested in optical interference circuits realized with lasers. In 2011, S. Utsunomiya, K. Takata, and Y. Yamamoto [334] proposed a mapping protocol to implement Ising models on such a system whith one master laser and several mutually injection-locked slave lasers, using right or left circular polarization of the slave lasers to represent a spin degree of freedom at each site. Indeed, each laser in the network is polarization degenerate and attains spin $+1$ in the case where right circularly polarized photons outnumber left circularly polarized photons, and spin -1 in the opposite case. In 2013, the work was extended by a team of Stanford researchers (California) still led by Y. Yamamoto, who improved the approach studying the collective behaviors of a degenerate optical parametric oscillator network [335]. The Ising problem is represented by coherently injecting the output fields of the degenerate oscillator to the other oscillators (with the amplitudes and phases governed by the coupling coefficients in the given problem). For such reason, this particular Ising machine has since then been known as the Coherent Ising Machine (CIM). One main difference with respect to the previous blueprint is that, contrary to lasers, degenerate optical parametric oscillators are phase sensitive, which eventually resulted in improved performances of the Ising machine (see [335]).

Another relevant step was accomplished in 2016, when an extended team comprising researchers from the Japanese National Institute of Informatics, Stanford, and NTT Basic Research Laboratories presented a working experimental realization of a fully programmable 100-spin CIM [336]. The system was realized on an optical processor with electronic feedback, which was presented as easily scalable and requiring only room-temperature technology. In 2021 a joint effort of NTT and the Japanese National Institute of Informatics resulted in the realization of a 100,000-spin CIM [337], which also came with the claim that such system was in fact 1000 times faster than a digital computer in solving MAX-CUT instances. Thanks to the optical approach, there is no a priori routing limitation to the number of realizable connections, which means the CIM can represent problems characterized by all-to-all connectivity. See [338] for a practical and complete introduction to the CIM concept.

The Digital Annealer

In 2017, a team of researchers working for Fujitsu, Japan, introduced an a fully-connected 1024-variables Ising machine implemented in an FPGA [339] ([340] is an additional closely related paper by the same team). The implementations made use of multiple engines, performing each a Markov-Chain Monte

Carlo stochastic search to minimize the Ising energy. Later, in 2019, the Fujitsu team collaborated with 1QB and Microsoft, published a paper [341] that helped understanding the underlying structure of this new approach. The paper clarifies that the new optimization algorithm implemented in [339] is an improved version of the legacy simulated annealing, specialized for solving Ising problems. On the software level, the new algorithm uses smarter choices for the initialization and for the acceptance procedure for new samples, while on the hardware level the use of application-specific CMOS hardware in FPGAs allows for massive parallelization and speed. This Ising machine has been named Digital Annealer by its creators due to its similarity to simulated annealing.

Lately, the Digital Annealer has been defined by Fujitsu as a physics-inspired [341] and quantum-inspired [342] solver.

The Simulated Bifurcation Machine

In 2016, H. Goto (Toshiba corporation) [343] detailed a network of Kerr-nonlinear parametric oscillators capable of implementing an Ising machine. Such system is related to that presented in [335], one of the seminal papers for the Coherent Ising Machine , which also comprises parametric oscillators. Nonetheless, while the NTT machine is an inherently dissipative system, the novel system introduced by Goto implements coupling between two oscillators via photon exchange, where the energy of the network is conserved, ultimately thanks to the Kerr effect. This novel type of Ising machine can thus be operated without dissipation. Additionally, the authors showed that the solutions of an Ising problem can be achieved in their system exploiting bifurcation-based adiabatic quantum computation. The term bifurcation comes from the fact that, during the prescribed adiabatic evolution, nonlinear terms are slowly raised, causing the position variable of the oscillators to bifurcate between two (potentially degenerate) stable states. Thanks to quantum phenomena, the system can simultaneously follow both paths, as it happens in AQCs. This Ising machine introduced by Goto will be later named Bifurcation Machine. Later, in 2018, Goto et al. [344] performed a numerical study proving that the final distribution of samples to be expected from the Bifurcation Machine in presence of dissipation is the Boltzmann distribution, similarly to what happens in AQCs.

In 2019, H. Goto et al [345] introduced a classical Hamiltonian exhibiting bifurcation phenomena inspired by the quantum Hamiltonian of the original Bifurcation Machine. The authors simulated the new system using FPGAs to tackle 2000-nodes instances of MAX-CUT. The simulated system is named Simulated Bifurcation Machine, and it's currently commercially available.

2.5.2 The Memcomputing machine

Memcomputing is the name given to an emerging computational paradigm that exploits the evolution of a circuit based on memristors to perform computations. It should not be confused with the in-memory or near-memory computing paradigm which was conceived to avoid most of the costs of moving data by processing directly within the memory subsystem [346]. In fact, Memcomputing is a non-Turing paradigm that does not exploit the Von Neumann architecture. It does not have a dedicated memory component but rather exploits the evolution of a physical system. The dynamical evolution of this system, therefore, plays the role of a fictitious memory. Memcomputing devices perform computations harnessing the nonlinear dynamics of a physical system. Such concept was pioneered by Chua [347]. The Chua's approach allows to solve nonlinear optimization problems through possibly such nonlinear dynamics. One of the key differences between the Chua's approach and Memcomputing is that, while the former is a fully analogical method, the latter exploits logical gates, hence the dynamical evolution is used to support a fully bit-based solver. The following Sections will introduce the concept of Self-Organizing Logical Gates, which are the basic building blocks of Memcomputing Machines, and explain how they can be realized in hardware.

Self-Organizing Logical Gates

Consider an AND gate. This gate has only four logically consistent states, according to the truth values assigned to the in-terminals:

$$0 \wedge 0 = 0; \;\; 0 \wedge 1 = 0; \;\; 1 \wedge 0 = 0 \;\; ; 1 \wedge 1 = 1 \tag{2.38}$$

Now suppose we can build a physical system having only these four states as its equilibrium points, and no other attractor (namely, no periodic orbits or chaos). Such a physical system will tend dynamically self-organize (SO) into any one of these four states, according to the initial conditions. Such a gate is called self-Organizing Logic Gate (SOLG), in which only the final states are important, not how such states are reached during dynamics [348]. The variables of the original Boolean gate will be mapped into voltages of an actual electronic circuit that implements the dynamic system underlying the SOLGs. However, the three terminals of this new SO-AND gate are not necessarily digital, because the voltages follow a trajectory $\vec{x}(t)$ in the phase space $X \subset \mathbb{R}^3$. It is necessary to relax the digital condition at the terminals of the gate and allow them to adapt to any bounded value they may support during dynamics.

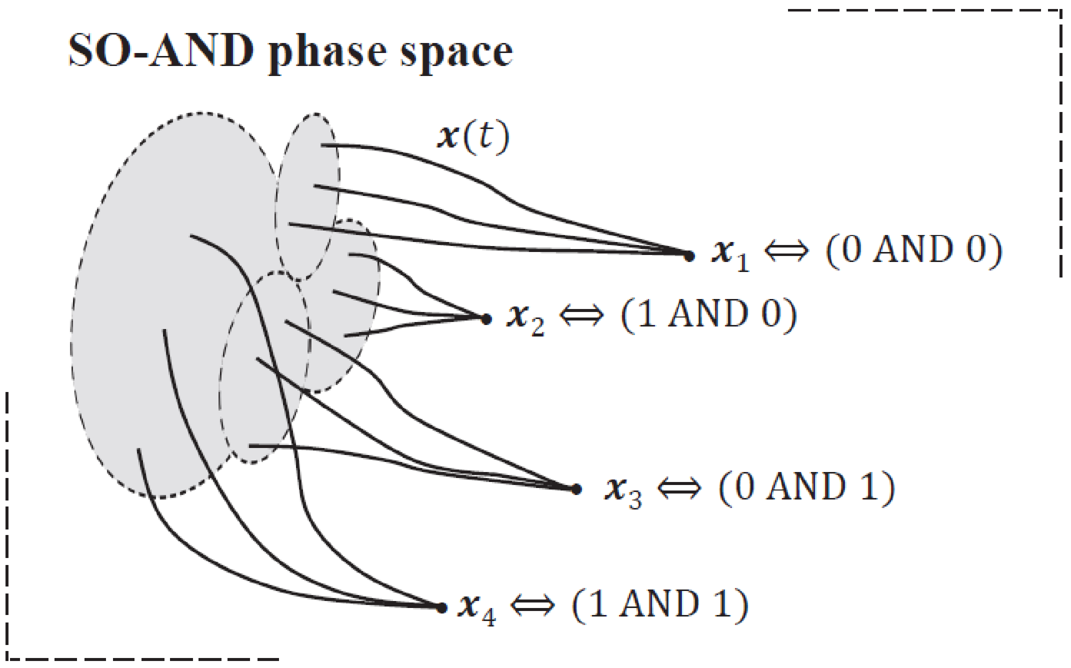
SO-AND phase space
$x(t)$
$x_1 \Leftrightarrow$ (0 AND 0)
$x_2 \Leftrightarrow$ (1 AND 0)
$x_3 \Leftrightarrow$ (0 AND 1)
$x_4 \Leftrightarrow$ (1 AND 1)

FIGURE 2.5: The phase space of a self-organizing AND (SO-AND) gate with only four equilibria, each one corresponding to a logically consistent state of an AND gate. In the absence of any other attractor, the phase space of this gate clusters into four basins of attraction (grey areas). Image from [349].

Therefore, when the system is at equilibrium (in one of the four logically consistent states), no dynamics should occur. On the other hand, away from the logically consistent states, the system will change its state, always attempting to converge to one of the equilibrium points (Figure 2.5). In fact, at any given time during dynamics (after the initial condition has been set, but before the output is reached), the state of the system could be in any non-linear combination of 'input' and 'output' states. When the system is in a non-linear combination of states then the system is in an unstable configuration as shown in Figure 2.6.

This behavior can be realized if we introduce extra degrees of freedom, namely, the dimension of our (phase) space of dynamical variables is not three (or whatever is the number of gate terminals), but larger. The extra (memory) degrees of freedom are represented by $\vec{\tilde{x}}$. They could be due to any physical mechanism that induces time non-locality (memory) in the system [350]. To summarize the concept of the SOLG two steps have to be specified: first, provide dynamics to the voltages $v_i = v_i(t)$ at the terminals, where i is the index associated with the gate terminal; second, add as many extra dynamical (memory) degrees of freedom $\tilde{x}_k(t)$ as necessary to let the system evolve to the only logically consistent equilibrium states of the gate. Let's consider first the step: by providing dynamics to the literals of the terminals of the original gate, one

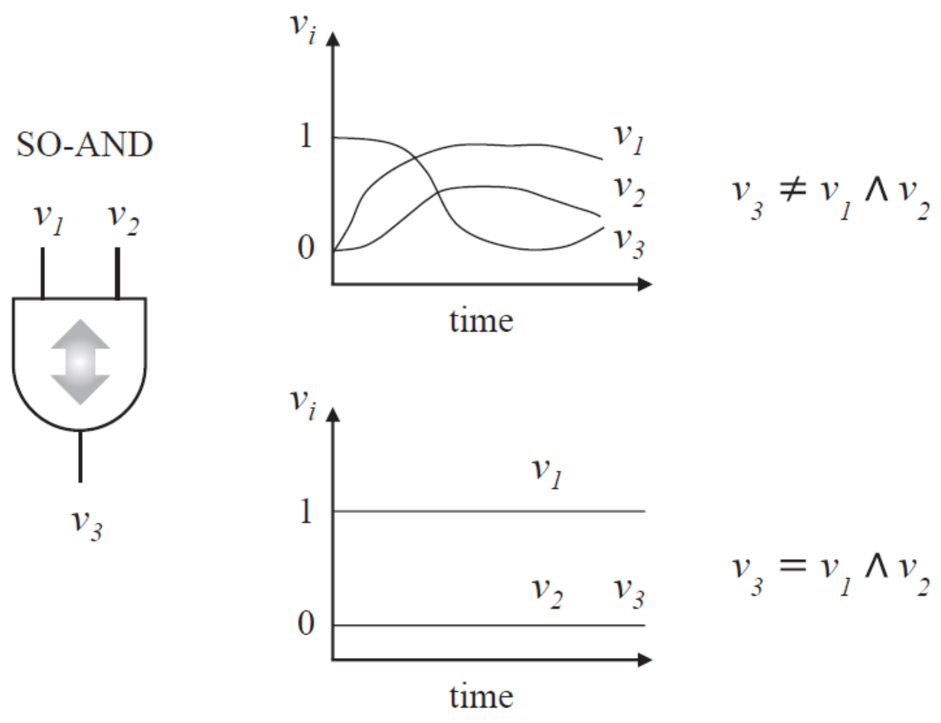

FIGURE 2.6: The SO-AND gate is in an unstable configuration if its logical relations are unsatisfied. It is in a stable configuration if one of its logical relations is satisfied. Image from [349].

goes from a set of discrete states (the logically consistent solutions of the gate) to a dynamical system of the voltage variables only. The resulting system may have several types of critical points, in addition to the equilibrium points which correspond to the correct logical proposition of the gate. In particular, this (reduced) phase space of the voltages may contain local minima that could trap the system dynamics. This is where the second step of the procedure comes in handy. If memory variables are appropriately introduced, they expand the reduced phase space of the voltages, transforming any possible local minimum into saddle points. This leaves the equilibrium points representing the logically consistent solutions as the only equilibrium points. Active elements are needed to obtain appropriate extra degrees of freedom. The reason is that the dynamical system has to be able to always end up in the correct equilibrium points, which represent the logical states of the gate. In other words, the terminal voltages have to be guided toward the solution. To accomplish this, the system needs feedback so that if it strays away from the correct path in the phase space, it will immediately correct it.

Physical realization of SOLGs

Although the specific gate realization may be different, the general idea is the same for all types of Boolean gates. The first step to describe the physical realization is to choose reference voltage v_c to associate the logical 0 to v_c (for example $v_c = 1V$) and 1 to $-v_c$. Consider, again, an AND gate with initial configuration 01 as input and 1 as output. It means that the SOLG is in an initially unstable configuration, and will attempt to dynamically change its state to reach one of the four possible equilibrium states compatible with an AND gate. A set of active devices called dynamic correction modules (DCMs) [351], attached to each terminal reads the voltages of the other terminals as well and they provide the necessary feedback to the system. DCMs are made of resistive memories, with a minimum R_{on} and a maximum R_{off}, and voltage-controlled voltage generators (VCVGs), which are active devices. In Figure 2.7 an illustration of the circuit that implements a DCM is shown. The VCVGs are linear voltage generators piloted by the voltages v_1, v_2, and v_3 at the terminals of the SOLG. The output voltage of the VCVG is given by

$$v_{VCVG} = a_1 v_1 + a_2 v_2 + a_3 v_3 + dc \tag{2.39}$$

The parameters a_1, a_2, a_3 and dc are determined to satisfy a set of constraints characteristic of the gate, which will induce the physical system to always satisfy the gate logic. Therefore, these parameters are different for an AND, OR, XOR, or any other gate. Since 5 voltages are involved in DCMs, 20 parameters have to be defined. If the gate is connected to a network and the gate configuration is correct, no current flows from any terminal: the gate is in stable equilibrium (steady state). Note that one could choose a different set of parameters (or even a different design altogether for SOLGs) that satisfies the same conditions, or simply the general principles of operation of SOLGs. However, finding a suitable set of parameters to guide the system toward the equilibrium point is the main problem concerning the memcomputing paradigm. In Section 4.1.2 we discuss the parameter tuning approach we adopted in [2] to boost the performances of a Virtual Memcomputing Machine.

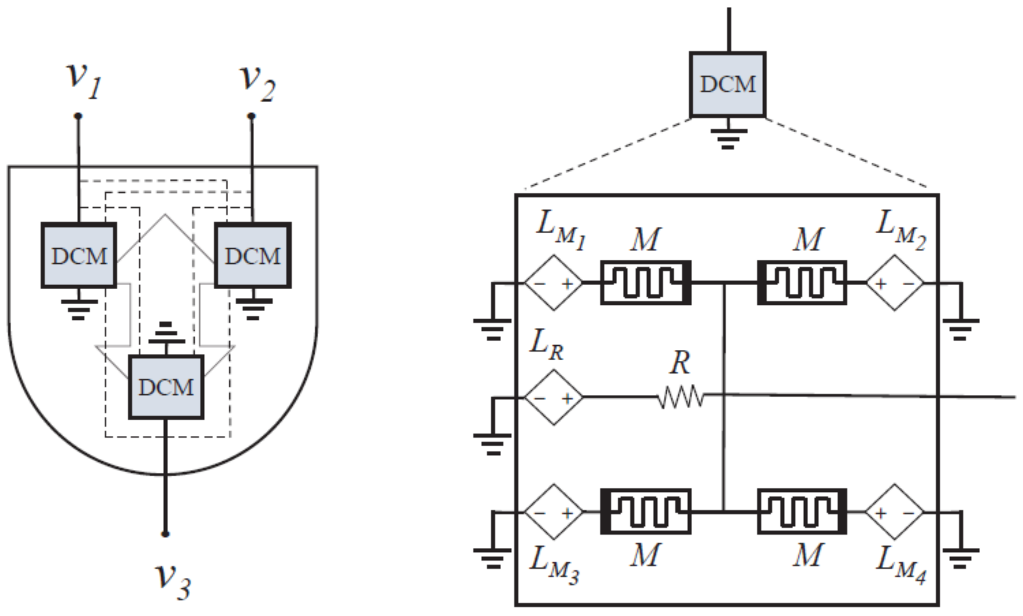

FIGURE 2.7: Self-organizing (SO) AND gate, left panel, formed by dynamic correction modules (DCMs), right panel. M indicates the resistive memories, while The linear functions L drive the voltage-controlled voltage generators (VCVG). Image from [349].

Integer linear programming with memcomputing

Integer Linear Programming (ILP) is a class of combinatorial optimization problem that can be formalized as follows:

$$\begin{aligned} &\min_{\boldsymbol{x}} \sum_i f_i x_i \\ &A_{eq}\boldsymbol{x} = \boldsymbol{b}_{eq} \\ &A_{ineq}\boldsymbol{x} \leq \boldsymbol{b}_{ineq} \end{aligned} \tag{2.40}$$

where $\boldsymbol{x} \in \mathbb{N}^n$, $f_i \in \mathbb{R}\, \forall i \in \{1, 2, \cdots, n\}$, $A_{eq} \in \mathbb{N}^{m_{eq}\times n}$, $A_{ineq} \in \mathbb{N}^{m_{ineq}\times n}$, $b_{eq} \in \mathbb{N}^{m_{eq}}$, $b_{ineq} \in \mathbb{N}^{m_{ineq}}$ and m_{eq} and m_{ineq} are the number of equalities and inequalities constraints respectively.
The memcomputing approach to ILP problems is based on the concept of Self-Organizing Algebraic Gates (SOAGs) [352]. These gates are similar to the SOLGs in their mode of operation. However, instead of satisfying a Boolean relation, they satisfy an algebraic relation, e.g., an inequality relation between variables. They can also use any terminal simultaneously as 'input' or 'output' to satisfy an algebraic relation. Using SOAGs, one can assemble a Self-Organizing Algebraic Circuit (SOAC). The SOAC collectively self-organizes in

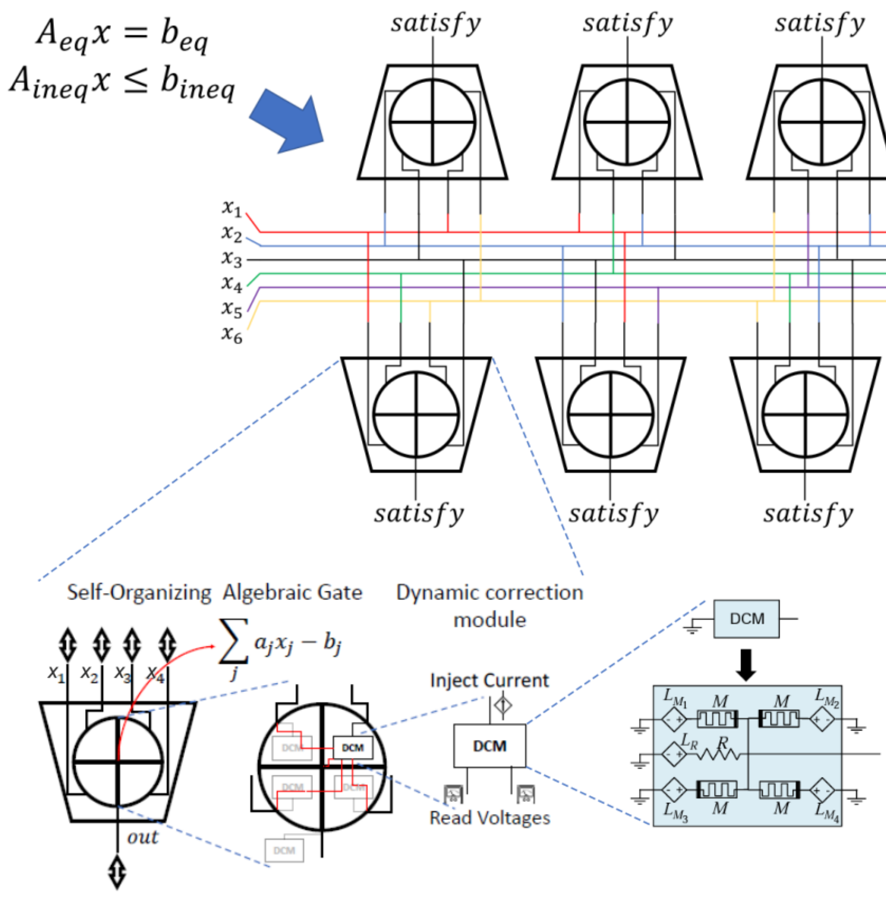

FIGURE 2.8: A Self-Organizing Algebraic Circuit (SOAC) represents an ILP problem. Each Self-Organizing Algebraic Gate (SOAG) is a linear condition that has to be satisfied when solving the ILP. The output of the SOAGs is imposed in order to obtain feasible solutions. The cost function is mapped into an additional SOAG whose inequality value is progressively reduced. The SOAGs at the circuital level are composed by dynamic correction modules (DCMs); the circuit components of a DCM are illustrated in the figure below on the right. Image from [349].

order to satisfy the constraints of an ILP problem 2.8. Indeed the linear equalities and inequalities of a given ILP problem can be directly mapped on a SOAC (see Figure 2.8). Instead, the cost function can be easily reformulated as an extra linear inequality with an extra bounding parameter. Iteratively, this bound is reduced forcing the SOAC to self-organize and find a new feasible solution, each time closer to the global optimum as described for MAX-SAT problem mapped on SOLC. The interested reader can consult [352] for the application of SOAGs to ILP.

Currently, no physical implementation of such concept exists, but Memcomputing Inc. has realized software able to simulate the circuital dynamic of Memcomputing machines on classical computers. The simulation software exploits GPUs to increase the overall performance, and has already been applied to solve some problems faster and with a better cost scaling than classical state-of-the-art rivals [2]. Nonetheless, the realization of a hardware Memcomputing Machine remains the long-term objective of the company, since a physical system could implement the same dynamic in a fraction of the simulation time. On the other hand, a simulated Memcomputing Machine is a classical solver, and we should remember that according to the extended Church-Turing thesis, all classical computers are equivalent up to polynomial factors [353].

2.6 Comparison of AQCs' performance to other computational paradigms

The search for a quantum advantage is still open across all currently available quantum devices, and probably several more years are needed before a quantum computer will surpass the performance of a classical supercomputer in solving most instances of a specific problem. It is indeed still debated when the current *quantum gold rush* [354] will generate practical utility [28]–[31]. Nonetheless, the great economic interest of an quantum computational advantage has induced a non-negligible amount of hype both in the scientific community and in those who support the research through funding, resulting in a trigger-happy approach towards supremacy claims. A notorious example is a 2019 paper where the Google team claimed to have reached *Quantum Supremacy* with their Sycamore QPU, surpassing what could be done with a classical supercomputer [28]. The IBM team, though, contended such claim [29], and [30] and [31] later showed how a modern supercomputer can match the aforementioned "quantum supremacy" performances.

In this context, it is fundamental to possess a toolbox of interpretable and clear metrics to correctly benchmark the current capabilities of new computational paradigms.

Section 2.6.1 introduces useful metrics to compare optimization solvers. Such metrics will be useful to compare AQCs' performance to those of state of the art classical solvers. Section 2.6.2 reports a literature review of the most significant works on AQCs' performance benchmarking.

2.6.1 Metrics to benchmark solvers on optimization problems

This Section introduces the most popular metrics to evaluate and compare performances of optimization problems solvers (Time to Solution, Gap to optimal solution) and a novel proposal that better estimates the average expected expense to find a solution (MFST). Lastly, problems with planted solutions are presented as a way to compare solvers performances also on the hardest instances.

Time To Solution (TTS)

The Time To Solution (TTS) is defined as:

$$T_{TTS} = \frac{\mathbb{E}\{t\}}{p} \tag{2.41}$$

where $\mathbb{E}\{t\}$ is the expected solution time and p is the solution probability. $\mathbb{E}\{t\}$ can be easily estimated via the usual sample mean estimator:

$$\bar{t}_s = \frac{1}{|I_s|}\sum_{j\in I_s} t_j \tag{2.42}$$

where I_s is the set of solved instances, $|I_s|$ its cardinality, and t_j is the time required to solve instance j. Thus, the solution probability p can be estimated as $\bar{p} = |I_s|/|I|$, hence the estimator:

$$\overline{T}_{TTS} = \frac{\bar{t}_s}{\bar{p}} \tag{2.43}$$

Mean First Solution Time (MFST)

It is customary, when running algorithms, to define a maximal execution time, after which the computation is stopped and a new parameter (e.g. seed) is used to run the computation. This requires proper metrics to evaluate the overall expected execution time. In particular, the estimation should include the time spent in failures and not just the average time of successes. The estimated execution time then correlates with the allocated computing time and hence with the monetary budget for the computation. In [2], we introduced a new metric which takes fully into account the time spent in failed runs. We called this metric the Mean First Solution Time (MFST) (as it is inspired by the Mean First Passage Time concept in physics [355]).

Given a problem instance, the MFST is defined as:

$$T_{MFST} = \mathbb{E}\{k\}T_{max} + \mathbb{E}\{t\} \tag{2.44}$$

where $\mathbb{E}\{k\}$ is the expected number of failures before the first solution is found, T_{max} is the maximal allowed execution time (e.g. for one seed) and is not a random variable, and $\mathbb{E}\{t\}$ is the expected solution time (see Eq. 2.42).

The variable k is a random variable which follows the negative hypergeometric distribution $\text{NHG}(|I|, |I| - |I_s|, 1)$

$$k \sim \text{NHG}(|I|, |I| - |I_s|, 1) = \frac{\binom{|I|-k-1}{|I|-|I_s|-k}}{\binom{|I|}{|I|-|I_s|}} \tag{2.45}$$

where $|I|$ is the cardinality of the run set. Therefore the expected value of the number of failures k before a success is:

$$\frac{|I| - |I_s|}{|I_s| + 1} \tag{2.46}$$

Hence, we estimate the MFST via the formula:

$$\overline{T}_{MFST} = \frac{|I| - |I_s|}{|I_s| + 1} T_{max} + \frac{1}{|I_s|} \sum_{j \in I_s} t_j \tag{2.47}$$

Clearly, if all instances are solved, the mean solution time is obtained.

It is easy to show that there is a simple relation between $\overline{T}_{TTS}$ and $\overline{T}_{MFST}$:

$$\overline{T}_{MFST} = \frac{|I| - |I_s|}{|I_s| + 1} T_{max} + \frac{|I_s|}{|I|} \overline{T}_{TTS} \tag{2.48}$$

This relation shows that, if all problems are solved ($|I_s|$=$|I|$), the two metrics are the same. However when not all instances are solved, the presence of T_{max} creates a discrepancy. If T_{max} is much higher than the TTS, the TTS can significantly underestimate the real execution time.

MFST and TTS variance

Together with the estimators of the MFST and TTS, we can obtain the variances. The variance of $\overline{T}_{MFST}$ is computed following the error propagation formula:

$$\begin{aligned}\sigma^2_{\overline{T}_{MFST}} &= \left(\frac{\partial \overline{T}_{MFST}}{\partial |I_s|}\right)^2 \sigma^2_{|I_s|} + \left(\frac{\partial \overline{T}_{MFST}}{\partial \bar{t}_s}\right)^2 \sigma^2_{\bar{t}_s} \\ &= T^2_{max} \frac{(|I|+1)^2 |I_s|^2}{(|I_s|+1)^4} \frac{1}{|I|} \frac{1-\bar{p}}{\bar{p}} + \sigma^2_{\bar{t}_s}\end{aligned} \tag{2.49}$$

where:

$$\sigma^2_{\bar{t}_s} = \frac{\sigma^2_{t_s}}{|I_s|} \tag{2.50}$$

and $\sigma^2_{t_s}$ is the variance of the solution time.

For the time to solution, we can estimate the variance by observing that $|I_s|$ is distributed as a binomial distribution. That leads to:

$$\sigma^2_{\bar{p}} = \frac{\bar{p}(1-\bar{p})}{|I|} \tag{2.51}$$

Using the error propagation formula, it follows that:

$$\sigma^2_{\overline{T}_{TTS}} = \overline{T}^2_{TTS} \left(\frac{1}{|I|} \frac{1-\bar{p}}{\bar{p}} + \frac{\sigma^2_{\bar{t}_s}}{\bar{t}^2_s}\right) \tag{2.52}$$

All the presented formulas are meaningful if at least one instance has been solved, that is $|I_s| > 0$ or equivalently $p \in (0, 1]$.

Gap to solution

Evaluating the gap from the solution is useful to estimate the quality of a specific answer of a solver, or to estimate the ability of a solver to reach values for the cost function that are close to the global minimum. The gap is defined as:

$$g = \frac{C_{\text{solver}} - C_{\text{obj}}}{C_{\text{solver}}} , \tag{2.53}$$

where C_{solver} is the best value of the cost function achieved by the solver, while C_{obj} is the global minimum of the cost function. It follows from Eq. 2.53 that $g \in [0\%, 100\%]$. A low gap means the solvers has been able to approximate the correct solution in the allowed wall time, while a high gap could mean the tested solver is not suitable for the task at hand.

The gap to solution, or its average, can be used to evaluate the performance of an optimization problem solver or to tune it by comparing performances attained at different parameters sets. See Section 4.1.2 and 4.1.1 to learn how we used such approach to perform parameter tuning in [2] and [1], respectively.

In those cases where the global optimum cannot be obtained, and the gap cannot thus be calculated, one can simply resort to the absolute value of the cost function. Indeed, we can argue that readouts with lower energies suggest a better performance of the AQC. A slight improvement for this metric is described in [356], where authors introduce the *elite mean* of a collection of n_{reads} readouts. The elite mean is simply defined as the average cost of a percentage ϵ of the readouts, namely those $\epsilon \cdot n_{\text{reads}}$ readouts that attained the lowest values for the cost function. If $\epsilon = 100\%$ the elite mean becomes the simple mean, while if $\epsilon/100 = 1/n_{\text{reads}}$ only the best readout is considered. Usually the experimenter will want to chose $\epsilon \sim 1$ to cut out most of the readouts while still averaging over the best readouts.

Optimization problems with planted solutions

What TTS, MFST and also the gap to solution have in common is that computing their values requires knowing the solution or C_{obj}, which means these metrics cannot be used to evaluate performances for hard instances that cannot be solved by available solvers. To compare solvers on such hard instances, it is useful to test them on problems with a planted solution. It means starting from a given combination of the variables and defining the instance so that the chosen configuration is the solution of the instance. As an example, T. Albash and D. A. Lidar [148] compared QA and SA on Ising problems with planted solutions, demonstrating a scaling advantage for QA[14]. They built the planted solutions problems by defining a set of frustrated[15] loop Hamiltonians H_l, then composing them in a problem Hamiltonian $H_P = \sum_l H_L$ such that the ground state of H_P is the simultaneous ground state of each H_l. See the original paper [148] to also learn how to customize the loops H_l for a specific topology.

The concept of planted solutions has been extended by Hen, who formally introduced the concept of *equation planting* in the context of Ising problems [357]. Using equation planting means considering a simple mathematical problem that, when cast to an optimization problem, becomes a hard problem for heuristic

[14]See Section 2.6.2 for an explanation of the findings.

[15]In the optimization context, a frustrated (signed) graph is a graph whose constraints cannot be all satisfied at the same time. In the specific context of Ising problems, it means that at the global optimum some linear or quadratic coupling is actually increasing the energy of the configuration.

solvers. Such approach enables the researcher to comprehend in depth the structure of the problem, comprising global and suboptimal solutions, while also challenging the solver on a "hard" problem.

2.6.2 Literature Review: Comparative Analysis of Adiabatic Quantum Computers and Other Solvers

This section provides an overview on the most relevant papers that estimate AQCs capabilities by comparing its performances to those of other unconventional or classical solvers. In literature, QA performances have often been compared to those of SA both on the theoretical and experimental level, due to their analogies and similarities[16]. While benchmarking QA performances with SA, one has to keep in mind that SA is not a competitive algorithm for solving any optimization problem. Nonetheless, for this reason, SA performances have sometimes served as a pretext to assert the achievement of a *quantum advantage* (you will soon learn about this in Section 2.6.2). Unfortunately, when we compare AQCs to state-of-the-art classical solvers, no advantage have ever been found in the context of optimization problems. We will instead show in Section 4.2.2 that an advantage can already be achieved for sampling problems. Since this is a novel results without close analogues in the existing literature, we will describe it only in the Results Chapter.

Attention must be given also to the chosen benchmark problems. Several of the papers presented in this Section tested QA on a spin-glass system, which is conveniently the same system physically implemented by AQC devices. The use of the spin-glass system as a benchmark problem for QA has been debated, since it is not representative of several applicative problems. Indeed, authors in [358] show that industrially relevant problems are much harder in general than solving spin-glass models. On the other hand, in [359] the authors suggest than spin-glass problems could lead to understating the performances achievable by AQCs, and the study in [360] discusses how spin-glass benchmarks could advantage Simulated Annealing approaches with respect to AQCs.

In the following Section we will present articles comparing QA to SA. Then, we will show how QA performances compare to those of other unconventional solvers and state-of-the-art classical solvers.

[16]See Section 1.2.6.

Quantum Annealing vs. Simulated Annealing

To find literature references comparing Quantum Annealing to Simulated Quantum Annealing, see Section 1.2.9. For an introduction to Simulated Annealing, visit Section 1.2.5.

Simulated annealing (SA) has consistently served as the primary benchmark for assessing the capabilities of quantum annealing (QA) since the first experiments on the Ising ferromagnet $\mathrm{LiHo_{0.44}Y_{0.56}F_4}$ revealed that hardware implementations of QA protocols were possible [82], [83]. Before commercial AQCs became available in 2011 [90], early works by Santoro et al. (2002) [137] and Farhi et al. (also 2002) [144] provided support to a future adoption of QA by numerical simulating real world QA processes and comparing their speed and accuracy to those of SA implemented on CPUs. In particular, Farhi et al. [144] show that there exist problems for which QA running time is polynomial in n and SA running time grows more than polynomially in n. These works underscore the significance of this comparison by noting that QA is almost equivalent to SA, albeit with the unique capability of surmounting energy barriers in the landscape via quantum tunneling effects without an associated increase in energy. Comparing the performance of QA and SA enables us to evaluate whether quantum effects does facilitate or impede the search for global optima.

The theoretical advantage of QA over SA has since been proven in several works [138], [361]–[363]. In [138], the authors implement a simple Monte Carlo approach to simulate QA and compare it to SA on a Travelling Salesman Problem instance comprising 1002 cities. Their results show that the annealing process converges faster in QA as opposed to SA. In [361], the authors describe the similarities between the theoretical frameworks of QA an SA (similarly to what has been done in Section 1.2.7). They show that, when mapping SA to QA, the Markovian dynamics of a short-range classical Ising model is mapped to a short-range quantum system, but when going from QA to SA a short-range quantum system is mapped to a classical one comprising long-range interactions. They thus conclude that SA can be efficiently simulated by QA, but the converse is not necessarily true, which suggests that QA is more flexible than SA on a mathematical basis. In [362] authors make use of theoretical arguments and calculations to show that QA is expected to exhibit a faster convergence to the global minimum if compared to SA. The deduction is valid only on the benchmark problem they considered, which is a random spin-glass model shaped as a single chain. It is interesting to note that the authors suggested a "quantum inspired" simulated system where the QA evolves in an imaginary time. Their calculations suggest that such system could converge even faster than QA on such problems, which means that in this context "quantum inspired could be better than quantum". In [363], the authors study the behaviour of QA and SA on a glued-trees problem, which requires finding nodes with specific properties in a given graph. The authors perform theoretical calculations to obtain an annealing schedule that grants QA an exponential speed up over SA. One recent study utilizing numerical methods is [364], where authors present a theoretical advantage of SQA over SA in an optimization problem characterized by a large amount of local minima that slow down SA.

Such findings on QA shed light on the possibility for AQCs to overcome their close classical counterpart, but noise sources and hardware constraints can invalidate theoretical arguments in real-world experiments, as we extensively discussed in Section 1.3.6. Nonetheless, several experiments have empirically demonstrated an advantage of AQCs over classical SA [148], [172], [294], [365]–[368]. In [172], authors showed that a (now outdated) D-Wave Two AQC device was able to achieve performances similar, and in some case superior, to those of optimized SA. The methods were tested on a random spin-glass problem. The good performance was achieved thanks to the tuning of the strength of the ferromagnetic coupling of the qubits chains in the embedding[17]. In [294], Denchev et al. designed a set of optimization problems characterized by tall and narrow energy barriers that separate local minima, with the intent of favoring QA. The hardware AQC resulted 10^8 times faster than both SA and a Monte Carlo method on instances comprising 945 variables. Both the classical approaches were implemented on single-thread CPU. These encouraging results constituted one of the first experimental proofs of a *limited quantum advantage* [369] (see also [370] for additional remarks on [294]). Results from [294] were confirmed by King et al. [371], who modified the problem to make it more similar to real world scenarios while also implementing SA on GPUs. The D-WAve AQC mantained a great advantage, resulting 2600 times faster than SA on GPU. A different approach was used in [148], where authors compared QA to SA on Ising problems with planted solutions. No advantage was found for QA in terms of wall time required to find a solution at any problem size. Nonetheless, QA exhibited a better scaling than SA. Such result is really interesting in perspective. Indeed, it suggests a potential advantage of QA over SA on large-sized problems, which is exactly where an advantage could be useful. Since the Ising problems were created to respect the topology of the QPU, it is important

[17]See Section 1.2.10 for an overview of this parameter, and Section 3.1.3 to learn how to tune it for optimal performances.

to underline that the findings are limited to those cases where no embedding is required. In [366], authors operated an AQC to optimize the traffic lights in a large-scale traffic management problem. Using a fixed wall time, the AQC output solutions with a lower cost with respect to those obtained via SA. In [367], a parallel quantum computing approach allowed authors to outperform SA using an outdated D-Wave 2000Q AQC on a matrix factorization task. The AQC required considerably less computation time in comparison to the classical SA, while returning comparable error rates. Recently, in [368], the D-Wave team itself showed experimental evidence that QA achieves an advantage in terms of wall time with respect to SA on a spin-glass optimization benchmark.

When examining the extensive literature comparing QA vs. SA, one can realize that there is indeed an elephant in the room that few researchers seem to consider, namely the way SA is implemented and executed in classical hardware. SA performances can be boosted modifying the annealing schedule [372], and there are specific prescription to optimize SA for Ising problems [369], [373]. On the other hand, recent papers often makes use of the classical SA routine implemented in Ocean SDK, developed by D-Wave itself, that should not be expected to be optimal.

A few papers have underlined the shortcomings of QA, while shedding light on potential improvements. Of particular interest is the work by Heim et al. [374], where authors explain that the advantage found for QA over SA on 2D Ising spin glass problems in reference [294] is a byproduct of choosing large imaginary time steps in the path integral and choosing the lowest energy over all time slices. Indeed, such advantage disappears in the continuous limit. These findings are consistent with [359], where authors suggest that no speed up is expected for QA on 2D spin glasses. In [375], SQA underperform SA on complex instances, but results also suggest that fast, nonadiabatic, annealing schedules can improve the performance of simulated quantum annealing for very hard instances by many orders of magnitude. More recently, Zardini et al. [376] showed that QA underperforms SA on Bayesian network structure learning problems. They underline that a proper parameter tuning procedure could have improved the performance of QA.

Adiabatic Quantum Computers vs. other quantum computing devices

Comparing the performance of various quantum hardware platforms can pose challenges because of the diverse computational methodologies they implement. AQCs are inherently designed for solving optimization problems expressed in the

QUBO form, whereas gate-based QCs execute quantum algorithms through a series of quantum logic gates. It's worth noting that, while universal gate-based QCs have the capability to implement Quantum Annealing (QA), this process is generally not straightforward, and the reverse mapping is not always possible. Consequently, meaningful comparisons between AQCs and gate-based QCs are only feasible for problems that can be formulated both as QUBO instances and as quantum circuits. One notable example is the Quantum Approximate Optimization Algorithm (QAOA), a variational method designed for solving combinatorial optimization problems on a gate-based quantum computer [377]. Notably, QAOA can also be seen as a form of quantum annealing whith discretized time steps. In the limit where such time steps are infinitesimal, the arguments deriving from the adiabatic theorem that we introduced in Section 1.2.6 can be extended to QAOA, which can then be interpreted as an algorithm for performing adiabatic quantum computation on a gate-based QC. QAOA is recently gaining popularity, partly because it is expected to perform well also on current noisy hardware [26], and someone forecast it will be one of the first algorithms to show quantum supremacy on near-term devices [378].

QAOA is an interesting problem to benchmark AQCs vs. gate-based QCs, since it is the most efficient way currently known to use the latter to simulate the former. AQCs should be expected to have a better performance, since they are specialized hardware to perform precisely QA. On the other hand, if gate-based QCs can efficiently implement QAOA and outperform AQCs, it could be useless to put efforts in developing the AQC technology. Willsch et al. [379] were among the first to benchmark QAOA on these two competing paradigms. Their findings confirm the superiority of AQCs over gate-based QCs in solving combinatorial problems, but they also show that a simulated gate-based QC outperforms quantum annealing implemented on a real device. The latter result underlines once more the significance of noise effects in present-day quantum devices.

Adiabatic Quantum Computers vs. other solvers

AQCs empirical achievements with respect to SA are promising, but SA is no longer considered a competitive method to solve optimization problems. AQCs should also be compared to high-performances classical solvers and other novel computational approaches to solve optimization problems. This Section presents a literature review of significant papers that compare the performances of AQCs to the hardware and software solvers presented in Section 2.5 and to state-of-the-art classical solvers [380], [381].

Proving an absolute advantage of AQCs over competitive classical solvers on a specific application remain an elusive goal, and, as far as we know, there are no results in literature where AQCs outperform commercial solvers (e.g. Gurobi). An encouraging result come from Santra et al. [380], who compared a (now outdated) AQC device to the exact classical solver `akmaxsat` on MAX 2-SAT. They stated that within the ensemble of hard random MAX 2-SAT problems there are likely to be found problems for which QA has an advantage over exact classical solvers, and vice versa. At the same time, they found hints that simulations of QA on classical devices can achieve a scaling which is even better than QA for this particular problem. Despite the current shortcomings of AQCs on optimization tasks, few researchers have recently produced results that suggest a limited quantum advantage can be achieved employing AQCs in sampling tasks [23], [318]. In [318], authors provide evidence that AQCs can compute the negative phase for a Restricted Boltzmann Machine faster than Gibbs sampling on classical processors. Such result is confirmed by [23], where a fully-connected Boltzmann Machine with 32 units is trained on an AQC 8.6 times faster than using a classical approach parallelized on GPU. The result has been obtained exploiting parallel quantum annealing to accelerate the sampling process[18]. Authors show that a hypothetical AQC capable of extracting all samples in a single annealing cycle could complete the sampling task 822 times faster than the classical algorithm. When employing AQCs to perform sampling tasks onw exploits noise sources inside the quantum processing unit to achieve the desired computational results, which implies a milder detrimental effect of noise on the performances, if compared to optimization tasks.

AQCs performance has also been compared to that of new unconventional solvers. We start our literature review by commenting a great recent paper by Mohseni et al. [333], who tested various Ising machines on a set of benchmark problems. Authors underline how most of the Ising machines tested tend to have similar scalings in terms of the error probability and the time-to-solution metrics as a function of the number of spins, despite the extremely different approaches and technologies used to realize them. AQCs obtain the worst scaling and absolute results in almost all benchmark problems, while classical digital methods still appear to be the best approaches. Authors also comment on the inability of AQCs to sample uniformly all low-lying states, as opposed to other solvers (such as those based on SA) which tends to provide a set of more diverse solutions if initialized in different ways. In another recent work [381], a D-Wave Advantage AQC (DWA), a Virtual Memcomputing Machine (VMM),

[18]See Section 3.1.1.

the Fujitsu Digital Annealer (DA), and the Toshiba Simulated Bifurcation Machine (SBM) are compared on a satisfiability problem with planted solutions. The classical benchmark is constituted by the SATonGPU algorithm [382], that makes use of highly-parallelized computation. DWA exhibits the worst scaling of all tested solvers, being also the only analog device tested, and the only solver implemented on a custom hardware. Among the testes solvers, the DA was the unconventional method that obtained the best performances with respect to the scaling and the absolute time to solution. In a previous work [165] an AQC is compared to a Coherent Ising Machine (CIM). The AQC outperforms the CIMs on MAX-CUT on cubic graphs. On denser problems, however, the AQC exhibits a worse scaling than CIM, which in the end enables CIM to solve hard instances with more than 50 vertices an order of magnitude faster than the competitor.

Our team also performed benchmarking of AQC, VMM, and Gurobi on three problems of mathematical and industrial interest. The outcomes are thoroughly discussed in Section 4.1.2.

Chapter 3

Performance optimization techniques

Many papers regarding new computational approaches (in particular quantum computing ones) refer in their final remarks to the expectation that the specific tested hardware will undergo great improvements in the near future. Indeed, engineering limitations at the hardware level are often the main bottleneck to the performance achievable by the computing device. Nonetheless, a new buzzword has recently emerged in the quantum computing community: *middleware*. In the field of classical computing the middleware "is software that enables one or more kinds of communication or connectivity between applications or application components in a distributed network"[1]. In the quantum computing community, the term middleware often refers to software solutions that run on classical hardware aimed at optimizing in some way the quantum computing resources or enhancing the quality of the results produced by the quantum device. Companies such as Multiverse [384], Qbrain [385], and Q-ctrl [386] can be considered quantum middleware companies, even if they work to solve different problems in that layer. Solutions working at the middelware layer are gaining increased scientific and commercial interest, because they can deliver empirically verifiable computational advantages that can sometimes reach unexpected efficacy. As an example, novel results presented in this Thesis confirm that performances of AQCs can be significantly enhanced by simply tuning internal parameters governing the strength of the embedding chains and the duration of the adiabatic evolution. Thank to parameter tuning, we empirically demonstrated a 78x reduction in the computational time required for an AQC to reach the global minimum of an optimization problem [2].

The amount of middleware techniques currently undergoing scrutiny from the scientific community is surprisingly vast, especially in the field of Adiabatic

[1]As defined by IBM at [383]

Quantum Computation. The next (main) Section of this Chapter will summarize what can be found in literature regarding programming techniques that enhance the performance of AQCs, mainly at the middleware level. First, the concept of parallel computing is introduced. Then, we explain how to modify the annealing schedule and how to tune the chain strength to increase the probability to reach a solution faster. Afterwards, we describe how to make the embedding process more efficient and successful. We then explain how to modify the mathematical structure of the problem to increase the probability of finding a solution via adiabatic quantum computing. At last, we present postprocessing techniques to enhance the results and we briefly introduce the concept of hybrid classical-quantum computation. The second and last Section of this Chapter presents a literature review of proposals for future hardware improvements that could boost the computational capabilities of future AQCs.

This is a mixed theory-and-results chapter, meaning it introduces theoretical concepts supported by a literature review and also by experimental results obtained by the authors.

3.1 Software techniques to improve the performances of AQCs

3.1.1 Parallel quantum annealing

In recent years, the computing power of HPC centers has grown mainly thanks to specialized and highly parallel accelerators, chiefly graphics processing units (GPUs). A similar approach can be extended to quantum computation, which is usually prone to parallelization. Indeed, most of the quantum computing paradigms force the user to collect a lot of samples to overcome the detrimental effect of noise, and in particular gate-based QCs requires numerous runs since the resulting probability density function is often the actual desired output of the quantum circuit. Such sampling process can usually be parallelized on simultaneously running clones of the QPU, since each run is independent from the other. To parallelize computation on an AQC it is sufficient to embed the given QUBO problem in multiple locations of the QPU. Each anneling cycle will result in a number of samples equal to the number of embedded clones. This approach has been dubbed *parallel quantum annealing* by Pelofske et al. in 2022 [387]. Anyway, parallel quantum computing had already been explicitly used in previous experiments, as in Rocutto et al. [4] (2019), where we cloned in eight locations on the QPU the necessary functional to sample from the negative statistics of a RBM. Notably, parallel quantum annealing was the sole software optimization technique used in our work where we achieved an experimental limited quantum advantage in training a Boltzmann Machine [23]. In that case both the negative and positive statistics samplings were mapped on the AQC. The former was cloned 26 times, while the latter 112 times.

3.1.2 Modifications to the annealing schedule

In Section 1.2.6 we discussed the adiabatic condition for QA that ensures the global minimum is attained with high probability at the end of the evolution. The evolution must be slow, but it also has to be decoupled from the environment. Then, in Section 1.3.6, we learned that the available AQCs let the user set a time for the adiabatic evolution (t_{ann}) that is necessarily much longer than the expected coherence times. We also underlined how this force the system to be subject to noise sources that are nonetheless suspected to sometimes raise the probability of success of the computation. But t_{ann} is not the only way in which we as user can control the annealing schedule. Ocean SDK allows user to set up to 12 annealing schedule points (on D-Wave Advantage 6.2 [388]), which means the user can define the intensity of the transverse an longitudinal fields at 12 distinct point in time. In the following Sections we present the most popular ways to modify the annealing schedule to improve performance.

A different approach consists of starting the anneal with $A = 1$. It means that at the beginning of the anneal ($S = 0$) $F(s)$ is maximum and $G(s)$ is zero (the same scenario that we usually find at the end of the annealing). Then, annealing is performed in reverse, lowering the σ^z component in the annealing Hamiltonian and raising the σ^x component. Qubits are then allowed to slightly relax. When A reaches a user-set value, e.g. $A = 0.5$, the anneal proceeds as customary, raising A again. Such a procedure is known as *reverse annealing*, and its application is explored more deeply in Section 3.1.2.

At the beginning of the process, the Hamiltonian does not possess a σ^x term, so the system is locked to a classical spin state. The user can specify such a state, and the annealing will involve neighboring states of the specified state. Indeed note that since $F(s)$ never reaches zero, the collective quantum state of the qubits is never in a superposition of all the classical spin states. For this reason, reverse annealing is considered a local search algorithm. It can enhance the suboptimal solutions of an optimization problem by looking for better solutions among the neighboring configurations. A sketch of this concept is drawn in Fig. 3.1

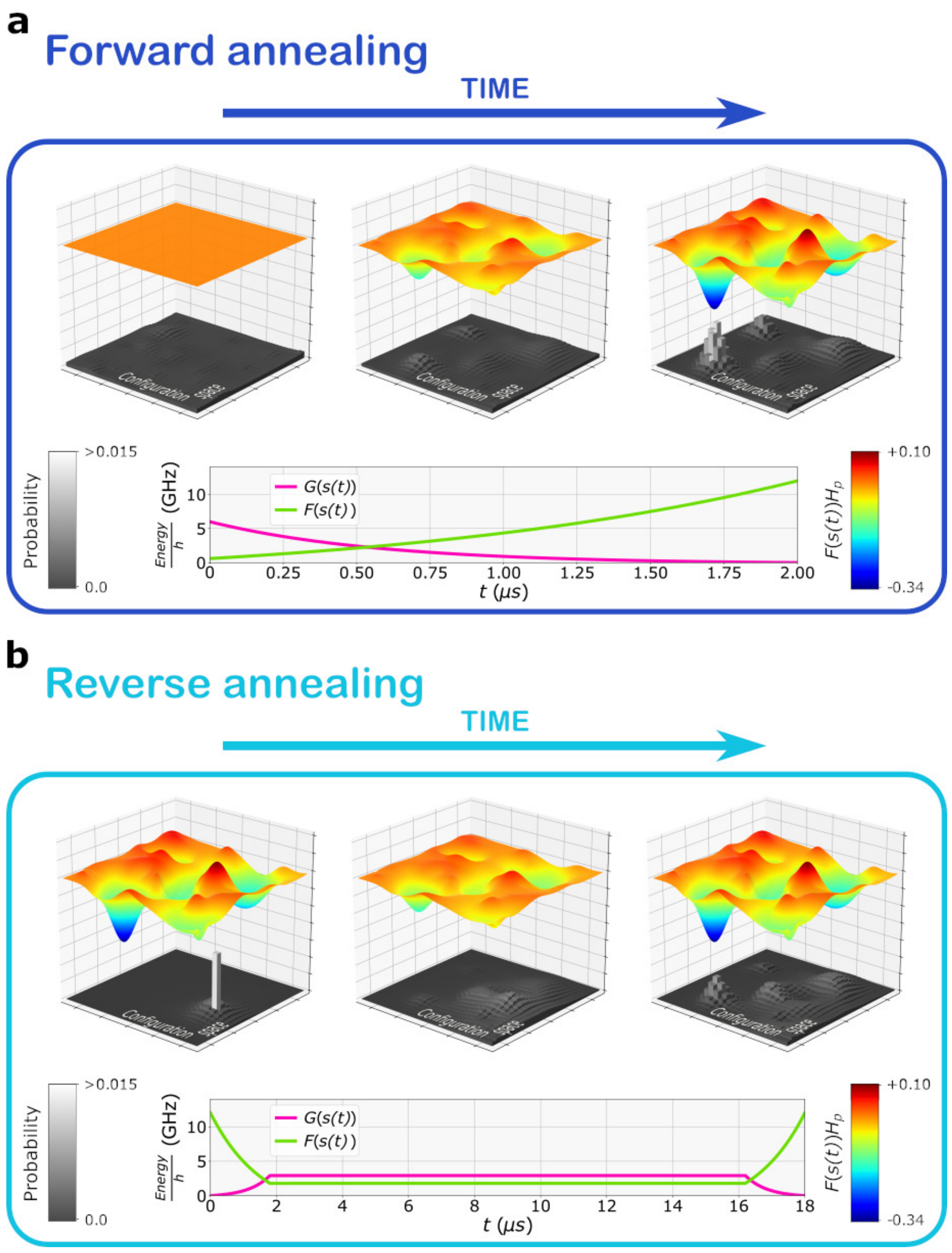

FIGURE 3.1: Visual representation of the process according to both quantum forward and reverse annealing schedule. Each bin of the gray 2–D histograms represents a possible configuration produced by the AQC. The height of a bin represents the probability that the corresponding configuration is the output of a single annealing process. The colored surface is a representation of the problem Hamiltonian H_P. Each configuration corresponds to a different value of H_P. For clarity, the problem is reduced to only two dimensions, and neighboring configurations differ by a single bit. A further graphical simplification consists of smoothing the surface making it continuous. **a**: During the *forward annealing*, the σ^z component of the annealing Hamiltonian increases from 0 to its maximum value. The behavior is represented by the increase in the weight of H_P. The configurations histogram, initially almost flat, becomes more and more peaked around the configurations that minimize H_P. **b**: During the *reverse annealing*, the σ^z component of the annealing Hamiltonian $H(t)$ (Equation 1.37) is initially set at its maximum value and the σ^x component is null. It is then possible to initialize the system such that a single configuration has probability almost equal to 1 (apart from noise effects). The annealing is then performed in reverse, lowering σ^z components and raising σ^x components in the annealing Hamiltonian $H(t)$. The system partially relaxes, and configurations close to the initial one become populated according to the corresponding value for H_P, thanks to quantum tunneling and thermal effects. The final step consists of standard forward annealing.

Tuning the annealing time

A basic tool that is present in any ACQ is the possibility to control the annealing time t_{ann}, slowing or speeding up the annealing schedule. As an example, D-Wave Advantage 4.1 System allows the user to chose a total annealing time from 0.5 to 2000 μs [388]. Choosing the proper value for t_{ann} can have a huge impact on the probability to find a solution, while in other cases we can find a milder dependence. In [2], we performed a simple tuning of the annealing time that consisted evaluating the average time required to find the global minimum (Time to Solution, TTS) at different t_{ann}. In our case, longer annealing times resulted in a lower TTS[2]. It means that we expect to find solutions to new instances in a shorter time when using the tuned t_{ann}, at least for the problem size at which the tuning was performed. If the tuned parameter can be used at other problem sizes with success means they exhibit good *transferability.*

Evaluating the expected TTS can be impossible when facing a difficult problem where the global minimum is hard to obtain. In such a case, it is still useful to evaluate the average gap from the global solution[3]. In [1], we performed a simultaneous tuning of t_{ann} and the chain strength[4]. When comparing parameters with respect to the expected gap, it is fundamental that we set a fixed runtime that must be respected by every tested parameter set. Indeed, while TTS estimates the time required to reach the solution, and therefore tends to the "true" value at infinite computational time, the gap provides a meaningful comparison only at a finite, fixed time. So, in this case, we need to appropriately reduce the number of annealing cycles (and, thus, readouts) while we increase t_{ann}. We evaluated the gap by averaging over the best gap obtained on a set of distinct runs, each one with a fixed runtime. When estimating the TTS, we only consider the number of times the readout corresponds to the global optimum, while for the gap we are considering a set of suboptimal samples that may or may not contain enough information to learn about the TTS. Both hypothesis require further scrutiny, but the gap mantains a useful function whenever the global minumum is not reached by the tested solver. It has been suggested to improve the gap by averaging over a fixed percentage of the best gap values obtained in the whole available wall time, an approach that has been dubbed *elite mean*[5] [356]. In Chapter 4 you can find several experimental results regarding the tuning of the annealing time, in particular in Sections 4.1.1 and 4.1.2.

[2]See Section 4.1.2.
[3]See Section 2.6.1.
[4]See Section 3.1.3.
[5]See Section 2.6.1.

Quenching and Pausing

D-Wave Advantage System allows the user to modify the annealing schedule by setting 12 *anneal schedule points* [388], which consist in couples of values (s, A), $s \in [0, 1]$, $A \in [0, 1]$. The variable s indicates the fraction of time elapsed from the start of the annealing process and A indicates the fraction of the annealing schedule that has to be performed up to that point. Such a tool can be exploited, as an example, by performing *quenching* or *pausing*. Quenching consists in an extremely rapid acceleration in the annealing process (steep dependence of s on t) located at a particular s_q. On the other hand, pausing consists of setting the anneal points such that the annealing Hamiltonian does not change for a certain time, keeping $F(s)$ and $G(s)$ fixed (s is constant for a certain time interval).

Researchers have been interested for a long time in learning how modifications in the Hamiltonian of a spin-glass system can affect the order of the system (see as an example the pioneering work by Barouch et al. [389] (1971) on the Ising-XY model). Additionally, as we saw in Section 1.2.6, during QA the Ising system is expected to undergo a Quantum Phase Transition (QPT) where an exponential long annealing time can be required to respect the adiabaticity condition. So an interesting question emerges: how does quenching and pausing influence the behaviour of the system near the critical point?

Sengupta et al. [390] studied the change in state of the Ising system upon a sudden change in the transverse field, using a numerical approach. They understood that the sudden quenching is expected to generate long range correlations if the final value of the transverse field after the quenching corresponds to a critical value for the Ising system at hand. It is important to note that during the experiment the authors held the Hamiltonian fixed after the quenching. So we can interpret their result as suggesting that quenching the Hamiltonian up to a critical point and then pausing could result in long-range order and, consequently, a higher success probability for the QA process. In [391], Calabrese et al. performed several numeric tests to evaluate the behaviour of a linear Ising model after a fast modification of the external magnetic field from an arbitrary value to the critical value for the system. Also in this case the authors did not perform a QA process, but simply observed how the system would evolve in time after a quenching point in which the magnetic field was increased to a specific value. They showed that a quench of the system from a noncritical to a critical point causes a linear increase in time of the entropy, until its value saturates. The authors conclude that an arbitrary large entanglement entropy is to be expected in the asymptotic state. This behaviour differs from the ground-state case where the entropy diverges only at critical point. This picture applies in the more general context of dynamical correlation functions after a quench as discussed by [392] (see also [393] for a comprehensive overview of interesting effects on the entanglement in many-body systems). The aformentioned results were confirmed and extended by Das et al. [394], who calculated the exact dynamics of an infinite-range Ising model in a transverse field after a sudden quench. Their theoretical arguments support the thesis that, after quenching the transverse field, the system finds itself in a nonstationary state that has a large overlap with the new ground state. In fact, this is only true if the quenching did not cross a critical point in the transverse field. If that happens, there is a very limited overlap between the istantaneous ground states before and after the quenching. Finally, it the value of the transverse field after the quenching is close to the critical value, the old ground state is expected to have a significant overlap with many of the new eigenstates, and the magnetization of the system will undergo great fluctuations[6].

Few papers are instead focused exclusively on pausing. In a seminal paper [399] (2019), Marshall et al. showed that pausing midway through the anneal can cause a dramatic change in the output distribution. In particular, the authors show evidence that a strong peak in the success probability is achieved when a pause is inserted into the regular annealing schedule within a narrow band of values for s. Thus, it appears there is a specific time at which one should pause to actually boost performances. In the instances tested by Marshall et al., the pause has a negligible effect if it is shorter than 10μs. After that threshold, the success probability has a linear dependence on the pause duration, with the longest lasting pause corresponding to the highest success probability. The next year, Pelofske et al. [222] confirmed that the quantum state keeps evolving even during a pause in the annealing process, even if, in fact, their results show only a mild variation of the system energy during the pause.

Reverse annealing

In the previous Section we learned how the annealing process can be accelerated or paused to improve the probability of finding the solution to an Ising problem, and we briefly introduced the concept of *reverse annealing*. In this Section we explore more deeply this concept and its possible applications.

[6]When the final state of the system does not conform to the ground state of its final Hamiltonian, defects are produced [395]. The presence of such defects can be seen as a manifestation of the broader Kibble-Zurek theory of topological defects [396], [397] (see also [398] for an application of the Kibble Zurek theory on a simple two-states quantum system).

Reverse annealing prescribes to start the annealing process by initializing the system in a specific classical configuration and setting the annealing Hamiltonian at its final value $H(s = 1)$. Thus, the system will initially be in an eigenstate of the initial Hamiltonian, which only contains longitudinal terms σ^z. Anyway, the eigenstate is usually not the global minimum of the energy, or otherwise we wouldn't need the annealing process at all. During the first annealing phase s is gradually reduced, so the Hamiltonian evolves as if we were performing the usual annealing process backward in time. This initial phase usually stops around $s \sim 0.5$, then, after a pause, the usual annealing schedule takes place, and we raise s from ~ 0.5 to 1 again, completing the process with a measurement. See Figure 3.1 for a visual representation of the annealing schedule. To refer to the regular annealing process and distinguish it from reverse annealing, usually the term *forward annealing* is used in literature.

The idea behind reverse annealing is to initialize the annealing process starting from a known configuration, thus influencing the exploration of the quantum phase space so that the final readout is somewhat close to the initial configuration. One of the first cases in which reverse annealing was supposed to have a beneficial impact on minimizing the energy functional was in Battaglia et al. [139] (2005). The authors were working on a random satisfiability problem were QA was clearly worse than classical SA. They decided to test an annealing schedule where the time evolution of the annealing Hamiltonian was repeatedly advanced and reversed, which improved the performance of QA so much that it surpassed SA. Six years later (in 2011), Perdomo et al. [400] basically introduced the concept of reverse annealing as we know it today, but without naming it. They declare that this new strategy allows "introducing educated guesses as initial states" and "allows for the possibility of restarting a failed adiabatic process from the measured excited state". As cleverly pointed out in the text [400], the reverse annealing approach provides also a new perspective on hybrid techniques, where a classical solver could be used to find a good intermediate solution to be then optimized via QA[7].

Reverse annealing gained experimental prominence after D-Wave Systems published its whitepaper on reverse annealing [401] in 2017[8]. Since then, several teams tried to gain a better insight in the actual effectiveness of the technique. We will comment about a restricted selection [4], [242], [282], [283], [365], [403]–

[7]Nonetheless, it is more likely that QA will be used to find intermediate solutions to then be optimized via classical solvers, as suggested in [1] (see also the relative Section 4.1.1).

[8]A patent regarding various approaches to solve optimization problems using QA (comprising reverse annealing) was also registered in 2017 by members of the D-Wave team [402]

[405]. In [242] Ottaviani et al. demonstrate the effectiveness of reverse annealing on a matrix factorization problem. The use of reverse annealing enabled the authors to find the global optimum for most of the instances, while forward annealing failed on each instance. They also show that reverse annealing can be improved by controlling the *reversal distance*, namely the average difference in Hamming distance among a set of readouts. The reversal distance can be influenced by changing the time duration for the annealing process and the minimum s value reached during the reverse annealing. The result by Ottaviani et al. were supported three years later by Golden et al. [405], who applied the same technique in a similar setting, obtaining compatible results. In [365], King et al. officially introduced the concept of reverse annealing with the first peer-reviewed paper written by D-Wave Systems' researchers in collaboration with the Google team. The paper introduces a quantum-assisted genetic algorithm where reverse annealing is used as a mutation operator. The results of reverse annealing were promising, since it surpassed the competing classical algorithms. In [404], the authors present a numerical study of the closed-system quantum dynamics of both reverse annealing and iterated reverse annealing, which simply prescribes to feed the final configuration found during the previous annealing to the next reverse annealing cycle, in the hope of finding a better solution. The authors forecast that reverse annealing could provide an exponential speed up over QA if its use can help avoid the first order quantum phase transition that QA usually crosses (see also Section 1.2.6). On the other hand, they obtained poor results for the iterated reverse annealing. Venturelli et al. [403], Ikeda et al.[283] and Carugno et al. [282] obtained promising results when applying reverse annealing to a portoflio optimization problem, a nurse scheduling problem, and a job shop scheduling problem, respectively. In particular, reverse annealing was 100 times faster than forward annealing on the portfolio optimization problem. Rocutto et al. [4] tested reverse annealing to train a Restricted Boltzmann Machine, but obtained results which were statistically compatible with the forward annealing ones. Authors also show how the different technique impacts the temperature of the samples distribution and the log likelihood of single elements from the dataset.

Annealing offset

In addition to control the annealing schedule, D-Wave users can also set the *annealing offset* for each qubit, namely the time after which ithe qubit will undergo the annealing process [406]. D-Wave Systems advises to use the annealing offset to compensate for the different times at which qubit chains freeze[9]. In fact, longer chains might freeze out sooner than shorter ones, so that during the intermediate phase of the annealing process certain variables can become fixed while others remain uncertain. If, instead, the user advances the annealing of qubits in the shorter chains, they will freeze out earlier than they would under normal circumstances. A proper tuning of the annealing offset is mandatory to obtain a performance improvement. Indeed, findings from Yarkoni et al. [407] suggest that, by tuning the annealing offsets, QA can improve by an order of magnitude in ground state probability on instances of the maximum independent set problem. To achieve this result, authors used the covariance matrix adaptation evolutionary strategy [408] to heuristically optimize the annealing offset of the qubit chains. Similarly, Adame and McMahon [409] investigated classes of problems that remain challenging for D-Wave QPUs, and they showed that by using annealing offset tailored on the local connectivity of each qubit, the Time to Solution can be reduced by two orders of magnitude. Pushing even further the claimed advantage, D-Wave Systems declared that on an integer factorization problem the use of the annealing offset can result in a 1000-fold reduction in the computational time [410]. A possible explanation for the improved performance could come from [411], where authors underline how the annealing offset can be used to enlarge the minimal spectral gap for those problems (such as the 2-SAT instances they considered) where the first excited state is highly degenerate. In fact, annealing offsets have been shown to mitigate first-order phase transitions, exponentially increasing performance compared to SA [412].

3.1.3 Tuning the chain strength

As introduced in Section 1.2.10, the chain strength J_{chain} is the intensity of the ferromagnetic coupling set by the user to force the chains of qubits to behave effectively as two-state systems (virtual qubits). The coupling must be sufficiently strong to keep the chain together. Anyway, setting a value too big for J_{chain} will have two distinct detrimental effects. First, the energy gap between the up and down state for virtual qubits represented by a long chain could become too wide, impeding the possibility of a spin flip. Second, the actual cost function to be minimized could become negligible with respect to J_{chain}, thus experiencing increased influence of Integrated Control Errors[10], which would result in a misrepresentation of the problem couplings and biases. Choosing a

[9]See Section 1.3.6.
[10]See Section 1.3.6.

proper value for J_{chain} is therefore mandatory to perform quantum annealing successfully.

In Rocutto et al. [2], we managed to reduce the Time to Solution by 78 times by optimizing J_{chain}, the annealing time and the the weight parameter multiplying the contraints in the cost function. In that situation, we optimized $c = J_{\text{chain}}/Q_{\max}$, namely the ratio between the intensity of the chains coupling and $Q_{\max} = \max_{i,j}\{a_i, b_{i,j}\}$, where a_i and $b_{i,j}$ are the biases and the couplings, respectively, of the QUBO problem Q submitted to the D-Wave device. The tuning procedure was successful. The chain strength was the parameter with the most impact, resulting in a $\sim 10\times$ net increase in the probability of finding the global minimum. The tuned value exhibited good transferability, sensibly improving the performances at any tested problem size. Anyway, tuning the ratio c does only take into account the highest coupling or bias, which could potentially be expected to lead to suboptimal values. For this reason, Ocean SDK sets J_{chain} based on the values of the problem couplings as follows:

$$J_{\text{chain}} = I_{\text{chain}} \frac{2N_J}{N_{\text{vars}}} \sqrt{\frac{\sum_{\{i,j\}, i \neq j} J_{ij}^2}{N_J}}, \tag{3.1}$$

where I_{chain} is a tunable real parameter, N_J is the total number of quadratic connections in the original (not embedded) QUBO problem, and N_{vars} is the number of logical variables in the original QUBO problem (which implies $2\frac{N_J}{N_{\text{vars}}}$ is the average number of connections per logical variable). Note that in Ocean SDK $I_{\text{chain}} = 1.414$ by default, as can be learned by the function `uniform_torque_compensation` [413]. Choosing J_{chain} based on the above formula with $I_{\text{chain}} = 1.414$ can be a good educated guess, but one can in principle optimize either the formula or I_{chain} for the problem at hand to maximize performances.

One of the first contribution on the topic of optimizing J_{chain} came from Choi (D-Wave Systems) in 2008 [151]. There, the author derived an upper bound for the ferromagnetic coupler strengths in the presence of a constraint implemented as a penalty term in the optimization problem[11], later extended by [414]. Next, Venturelli et al. [172] investigated the optimal value for J_{chain} on fully-connected problems with random $+1$ and -1 couplings. They underline that, after the longitudinal field reaches the intensity of the transverse field, the dynamic of the qubits slows down (we described this watershed moment as the freezing point in Section 1.3.6). Thus, authors suggest that the chain strength must have a proper value that allows correlation of the chains when the original unembedded problem enters the spin glass phase, prior to the freezing point.

[11]See Section 2.1.2 to learn how to introduce constraints as penalty terms.

Thanks to numerical tests, they discovered the optimal value for J_{chain} scales as $\sqrt{N}$ as the number N of logical variables in the unembedded problem increases. Their argument is supported by the scaling of the critical transverse field in the Sherrington-Kirkpatrick model, which is indeed proportional to $\sqrt{N}$. Note that embedding the problem required $\sim N^2$ physical qubits, thus (in this case) the optimal chain strength in the experiment scales linearly with the number of physical qubits involved. Yarkoni et al. [415] experimentally tested a different scaling of J_{chain}. While tackling a scheduling problem, they decided to optimize the constant $s = J_{\text{chain}}/L_{\text{chain}}$, where L_{chain} was the length of the longest chain required in the embedding. The chain length scales linearly with the problem size for a fully connected problem[12], which means Yarkoni et al. adopted a method that would result in linear scaling for J_{chain} for fully-connected instances, contrary to the prescription from [172]. Further testing is required to compare and evaluate these different (and inevitably problem-dependent) approaches.

The chain strength ought to be optimized also for sampling purposes. Marshall et al. [166] introduced a methodology to map erroneous samples back to the desired logical space, and they equipped such approach with a proper scaling of J_{chain} to minimize the number of broken chains that has a logarithmic dependence on the problem size. Nonetheless, the resulting scaling would require an upgrade of current AQC hardware to implement the necessary chain strength.

3.1.4 Techniques to enhance embedding

As we learned in Section 1.2.10, the embedding process is mandatory to map a QUBO problem defined on logical variables to its version defined on physical qubits. The proper embedding is usually found via heuristic techniques that can be computationally demanding, but whose cost is hardly ever considered when benchmarking AQCs with classical solutions. Nonetheless, the time spent in finding an embedding can end up requiring a relevant amount of the total wall time in specific applications [279]. Consequently, researchers have tried optimizing the embedding heuristics presented in Section 1.2.10. While in Section 1.2.10 we introduced few heuristics that are currently used to look for a good embedding, we will now focus on new emerging techniques or strategies that can be implemented by researchers in their own experiments to optimize performances.

[12]See Section 1.2.10 and Table 4.1, where the chain length for different fully-connected problem sizes appears to be linearly dependent on the problem size.

Due to the specific design of Chimera, Pegasus, and Zephyr topology, it is particularly easy to embed complete clique graphs, namely those graphs presenting all-to-all connection[13]. Thus, some authors [158] have suggested to always embed the problem using a clique embedding with the same number of variables, since each problem with the same or less variables is a subgraph of the clique embedding. An obvious drawback is that the resulting embedding will often be suboptimal, comprising longer chains than required. This way, we lose the inherent advantage of sparse problems, that can be usually embedded efficiently and achieve better performances [416]. On the other hand, a great advantage is that the biggest cliques embeddable on current hardware are still smaller than a hundred variables, which means optimal embeddings could be computed for a specific hardware for each possible clique size, building a library of embeddings that would not need to be updated until a new device hit the market. After such effort, it would be possible to immediately embed any QUBO problem with a number of variables equal or inferior to the greatest clique embeddable, without ever computing an embedding again. Such approach is pushed further in [417], where Hamilton and Humble, inspired by Klymko et al. [158], describe the concept of minor set cover (MSC) of a graph G. The MSC is a subset of graph minors which contain any remaining minor of the graph G as a subgraph. As a consequence, any graph that can be embedded into G will be embeddable into a member of the MSC. In other words, the MSC is the set of minors for a given graph G such that any subgraph or minor of G will either be a member of the MSC or is a subgraph contained in one of the members. Thus, the MSC of a particular hardware is a finite set of embeddable graph minors which can be precomputed without reference to the input problem, and can then act as a lookup table. The discussion has been extended by Boothby et al. [180], who introduced a polynomial-time algorithm that finds a maximum native clique minor in a given subgraph of a Chimera graph.

Another approach exploiting the specific topology of AQC devices was introduced by Pelofske [160]. Indeed, the Pegasus graph natively contains a lot of 4-cliques (four fully-connected qubits). Pelofske devised a method to embed a general QUBO problem by composing connected paths of 4-cliques (*4-clique chains*). Despite requiring more qubits, this method allows more ferromagnetic couplers per chain, which can lead to more stable chains and a reduced occurrence of chain breaking events.

In [232] several novel techniques are devised for efficiently embed NP-hard

[13]See Section 1.2.10 to learn why.

optimization problems including knapsack and partitioning problems. One technique is based on embedding fractal-like graphs on the QPU, while another embeds the problem variables in a two-dimensional square graph before minor embedding it on the QPU. The obtained embeddings are efficient and can be found quickly.

As opposed to the above mentioned heuristic approaches, Zaribafiyan et al. [181] propose a systematic and deterministic approach to specifically embed cartesian products of graphs. As the authors point out, a a systematic embedding approach must rely on the regularity of the target AQC topology. Their methodology proves more efficient and reliable than the standard minor embedding strategy implemented in the Ocean SDK function `find_embedding()`.

Logical J-Compensation

In Section 1.2.10, we enunciated the concept of *uniform spreading*, which prescribes to divide the value of both biases and couplings during embedding, in order to distribute the reduced values to all qubits in each chain and to all available edges connecting chains. Despite this seems like the most reasonable way to map weights and biases on the embedded graph, in [153] Raymond et al. propose a different approach called *Logical J-Compensation*. Indeed, while the energy of any state without broken chains is correctly represented by uniform spreading, the actual ground state wave function during the annealing dynamic can be dominated by configurations comprising broken chains (which usually outnumber the correct ones by orders of magnitude). If this happens close to the freeze-out point [219], the final distribution can be irremediably affected, preventing optima from being found. The technique proposed by Raymond et al. aims at compensating inter-chain couplings so that effective couplings are balanced earlier in the anneal at the expense of inbalance later in the anneal (hopefully after the dynamics freezes out). The idea is thus to prevent defects to appear early during the anneal, where the dynamics is more interesting and spatially global, while possible defects after the freeze out point are expected to exert local influences that can be more easily post-processed. This is accomplished by changing the way a weight is divided among physical couplers in the embedding. The weight is no longer divided in equal parts, but each embedded coupling corresponding to that same weight will have a value that is inversely proportional to the *pairwise-logical susceptibility*, a measure of how much weaker the coupling is between chains relative to what would be expected for a pair of directly coupled physical qubits [153]. The couplings are then normalized so that their sum still gives the original weight.

Reuse of static embeddings

In [1] we tackled a problem[14] where various instances had the same (fully-connected) graph structure. In similar cases, it can makes sense to put efforts in obtaining a good embedding, since it can then be re-used for any instance of the same size. In [418], Bass et al. point out that in particular quantum applications the embedding procedure is instead called several times, which can become the major bottleneck for the overall process. This is particularly true when the user is applying partitioning heuristics (see next Section 3.3) that produce graphs that are not guaranteed to be embeddable, since the embedding heuristic will consume relevant time resources before halting. One potential strategy to overcome this issue is to pre-calculate the embedding for the largest fully connected problem graph that can be embedded onto the chosen quantum device (a *static embedding*). Then, such embedding can be used for any problem comprising a smaller or equal number of variables, whichever the structure of the couplings could be. One shortcoming of this strategy is that potentially embeddable problems with larger number of variables than the maximum clique on the AQC topology will be unnecessarily partitioned in smaller subgraphs.

3.1.5 Problem transformations

While optimizing all the different internal parameters of the device, it is equally important to focus on how to effectively express the mathematical problem we want to solve so that the probability of finding its solution is increased. In the following subsections we will talk about expressing and tuning constraints, how to improve performances via the introduction of useful additional penalty terms, gauge transformations, and parity compiling.

Tuning constraints

As we discussed in Section 2.1.2, implementing constraints in QUBO formulation requires the introduction of an additional cost function that makes unacceptable configuration energetically unfavorable. This approach, despite being the only one possible, has many shortcomings, chiefly the possibility for the system to explore and converge to those unacceptable configurations. In other words, there is no guarantee that solutions produced by solving the QUBO via QA satisfy all the constraints of the original optimization problem. This problem can be mitigated by raising the multiplicative constant that weights the

[14]See Section 4.1.1 for the results.

constraint in the global cost function, which can in turn hinder performances due to the high energy barriers that appear as a result in the energy landscape.

It is therefore usually desirable to perform an optimization procedure for the constraints weights, where various values for the weights are evaluated with respect to a performance metric[15]. In [2] we performed such tuning by evaluating the probability of reaching the global optimum in a fixed number of samples. Similarly, in [419], Wang et al. try to make the quantum annealing process more resilient to errors by tuning weights and biases of the QUBO problem in order to expand the *final energy gap*, namely the energy gap between the eigenvalues of the Ising form at the end of the adiabatic evolution. In other words, they tuned the weight of the constraints to make unfeasible configuration energetically unlikely.

One additional useful approach to respect constraints is *sample persistence*[420]. Originally introduced for simulated annealing [421], it identifies variables whose value is persistent throughout repeated independent cycles of annealing. Such variables are candidate to be fixed to their persistent value. The remaining ones are said to form a core of difficult variables, which then becomes the subject of search intensification. A recent paper by Ceselli and Premoli [416] makes use of sample persistence, testing it on four different QUBO formulations of the cardinality constrained quadratic knapsack problem (CQKP).

Ceselli and Premoli identify as the best formulation for CQKP the one in which the inequality constraint regulating the maximum weight of the chosen items is relaxed linearly. Authors consider this to be in contrast with the best practice, which advises to transform it in a quadratic fashion. The main advantage of the linear relaxation is that it avoids the explicit addition of the slack variable of the constraint, which in turns would require to be encoded as a set of binary variables. The QUBO resulting from linear penalties is thus more sparse, and its embedding requires less qubits. The curious reader can found in literature recent works providing theoretical background on how to transform fully connected interactions of quadratic terms into linear terms [239], [422].

Gauges

Gauges are modifications of the expression of an Ising problem that do not affect the energy of any configuration. To make an example, we can always choose a gauge factor $a_i = \pm 1$ for each qubit, and then modify weights and biases as $h_i \to h_i a_i$, $J_{i,j} \to J_{i,j} a_i a_j$. The new Ising problem obtained is equivalent to the original expression if we flip each qubit k in the solution that has a gauge factor

[15]See Section 2.6.1.

$a_k = -1$. In principle, the performance of an AQC should not be influenced by gauge transformations. In fact, several works have proved that modern AQCs' performance is sensible to this mapping [107], [356], [423]. In particular, Boixo et al. [107] show that different gauges can result in wildly different performances, due to calibration errors in the device. The authors mitigate the calibration errors by averaging over different gauges. Such approach is recommended in almost any application, to avoid situations where the problem results hard to solve due to an unfortunate calibration error that, as an example, promotes orientation of one or more qubits in an opposite direction than that required for achieving the global optimum.

Spin reversal tranforms

A spin reversal transform is a simple transformation that exploits a symmetry in the annealing Hamiltonian [203]. It consists in choosing a random array $\eta \in \{\pm 1\}^N$, where N is the number of qubits. Then, the parameters $J_{i,j}$ and h_i in the annealing Hamiltonian are transformed as $h'_i \leftarrow h_i\eta_i$, $J'_{i,j} \leftarrow J_{i,j}\eta_i\eta_j$. The new Hamiltonian has exactly the same energy spectrum as the original one, while the states have been changed by a local inversion of some variables, according to η. The internal dynamic of AQCs should be invariant to this transformation, but the symmetry is in fact weakly violated due to noise sources [106]. Averaging the results obtained by annealing an Hamiltonian under various spin reversal transforms effectively reduces the impact of systematic errors inside the QPU [424].

Spin reversal transforms should be used carefully, since they require submitting a new problem to the AQC, incurring in non-negligible additional overheads in the computational time that should be taken into account. Indeed, for each transform, the problem must be submitted from scratch to the QPU, multiplying the impact of the initial programming time on the total wall time of the computation[16]. Furthermore, it has been shown that increasing the number of transforms has diminishing returns for a fixed number of samples [424].

Ocean SDK allows users to set the desired number of spin reversal transforms via the `num_spin_reversal_transforms` parameter, declaring that "applying a spin-reversal transform can improve results by reducing the impact of analog errors that may exist on the QPU" [425]. Due to the expected increase in computational time, the default value for `num_spin_reversal_transforms` is zero.

[16]See Section 1.3.5.

3.1.6 Effective temperature estimation for sampling applications

Sampling from the Boltzmann distribution is a promising application for AQCs[17]. The temperature of the sampling distribution can be partially controlled by properly rescaling weights and biases in the QUBO formulation[18].

In order to evaluate the sampling temperature T_{eff} one must resort to temperature estimation techniques. We can divide possible approaches to this problem into two classes: dependent estimations and independent estimations. In the former case, we need to be able to approximate the Boltzmann distribution at the desired temperature with another heuristic, and then compare it with the quantum sampler. In the latter case, the temperature is estimated based exclusively on the results produced by the quantum annealer.

Dependent estimators

The dependent estimators are more reliable, but could fail for larger instances, since producing the samples with a different heuristic could become expensive [285]. Among this class we find the Likelihood approach and the Mean Square Error (MSE) approach [424]. In the former case, we want to maximize the likelihood that samples produced by the AQC are distributed as in the target Boltzmann distribution at a fixed temperature. In order to do that, we can minimize the Kullback-Leibler divergence D_{KL} between the sampled distribution P_A and the corresponding Boltzmann distribution B_β at the inverse temperature $\beta = 1/T_{\text{eff}}$:

$$D_{KL}[P_A, B_\beta] = \sum_x P_A(x) \log\left(\frac{P_A(x)}{B_\beta(x)}\right) . \tag{3.2}$$

Since P_A does not depend on β, the minimum of this functional with respect to β is attained when the energy matching criterion $EM(\beta) = 0$ is met in the following expression:

$$EM(\beta) = \sum_x P_A(x)H(x) - \sum_x B_\beta(x)H(x) \tag{3.3}$$

Minimizing $EM(\beta)$ works as a maximum likelihood estimator for β, evaluating the likelihood that the annealed samples were drawn from a Boltzmann distribution. Since the Boltzmann distribution is an exponential model, it is natural to

[17]See Section 2.3.
[18]See Section 2.3.

define the estimator in terms of expected energy, which is the sufficient statistic associated to the parameter β [426]. This proposed method is expensive, since estimating $B_\beta(x)$ for various values of β can be demanding.

An alternative objective is the Mean Square Error (MSE) on correlations:

$$\mathrm{MSE}[P_A, P_\beta] = \frac{1}{M} \sum_{i,j:J_{ij}\neq 0} \left(\sum_x [P_A(x) - P_\beta(x)] x_i x_j \right)^2 , \tag{3.4}$$

where M is the number of non-zero couplings. The expression inside the round brackets estimate the difference between the expectation values of the two distributions for each possible product of variables. This is then summed (squared) for each possible combination of variables. The above expression can be derived by β to obtain a complex expression that yields a criterion for local optimality [424].

Independent estimators

Independent estimators are promising approaches because, since they do not require estimating the Boltzmann distribution with other heuristics, they could be use in perspective when AQCs will become the fastest samplers available. On the other hand, current independent estimators are not reliable and often output wrong estimates. A prominent example for this class of estimators was introduced by Benedetti et al. [221]. Their procedure begins with noticing that at a generic inverse temperature β, the probability of observing a sample of energy E is given by $P_\beta(E) = g(E)e^{-\beta E}/Z(\beta)$, where $g(E)$ is the degeneracy of states at the energy level E, and $Z(\beta)$ is the partition function. Then, consider the logarithmic ratio of the probabilities associated with two different energy levels, E_1 and E_2:

$$L(\beta) \equiv \log \frac{P_\beta(E_1)}{P_\beta(E_2)} = \log \frac{g(E_1)}{g(E_2)} - \beta \Delta E , \tag{3.5}$$

where $\Delta E = E_1 - E_2$. We can perform sampling at the unknown current temperature β, but we can also rescale weights and biases in the energy functional by a parameter x, in order to sample from a second effective inverse temperature $\beta' = x\beta$ and esimate $L(\beta')$. Taking the difference $\Delta L = L(\beta) - L(\beta')$ we obtain

$$\Delta L = \log \frac{P_\beta(E_1)P_{\beta'}(E_2)}{P_\beta(E_2)P_{\beta'}(E_1)} = \Delta\beta\Delta E , \tag{3.6}$$

where $\Delta\beta = x\beta - \beta = (x-1)\beta$. By sampling at the two distinct effective temperatures we can evaluate ΔL for many different pairs of energy levels E_1

and E_2. We can thus plot ΔL versus ΔE, and a linear interpolation will yield the slope $(x-1)\beta$.

3.1.7 Postprocessing Techniques

Whether we are using AQCs to solve an optimization problem or to sample from a Boltzmann distribution, postprocessing techniques can come result useful to enhance the output samples.

The most necessary and well-known postprocessing techniques in the AQC field are those used to unembed a sampleset. Indeed, after the annealing process, ebedded samples can contain broken chains that still need to be mapped into digital-valued problem variables. The default approach used by Ocean SDK in the `dwave.embedding.chain_breaks` function is called *majority voting* [427]. It simply consists in counting the number of qubits in the chain that attained $+1$ or -1 values, and then setting the output value of the corresponding variable to the most represented case (ties are decided randomly). Other approaches listed in the Ocean SDK manual comprise choosing the variable value at random (with a weighted extraction) or choosing the value that minimizes the local term in the global energy functional.

After unembedding a sampleset, other postprocessing techniques can be applied to enhance the quality of the samples. When solving an optimization problem, we can apply a *greedy postprocessing*. It manipulates output samples flipping those bits that cause a constraint to be unsatisfied. Such approach can be useful to force each sample to become a feasible solution, as was done, for instance, in [418], but it works only for those samples having a short Hamming distance from the closest feasible solution.

Each cited technique to repair chain breaks has the collateral effect of warping the energy distribution of the samples. This jeopardizes the possibility of using AQCs as Boltzmann samplers. In our works on the Boltzmann Machine [4], [23] we applied majority voting to enable the use of all the broken samples, and we were able to obtain good learning scores. Nonetheless, postprocessing techniques to fix the broken samples are recommended [424]. Marshall et al. [166] proposed a strategy to fix broken chains while respecting the original distribution, named *restricted resampling* (RRS). It prescribes to perform a Monte Carlo simulation over the variable subset corresponding to broken chains. The simulation is performed over the logical subspace, namely the subspace of configurations of the logical variables (as opposed to the physical qubits), and it is performed at an inverse temperature β that is chosen based on the expected effective sampling temperature of the AQC. The results presented in

[166] demonstrate that RRS outperforms majority voting in sampling applications. Despite performing better, RSS still has open problems to solve, chiefly the fact that it requires to know the effective temperature of the AQC, which, as we saw in the previous Section 3.1.6, can be challenging to estimate.

3.2 Quantum annealing correction

In the gate-based quantum computing community, error correction techniques are strategies that, by adding redundant ancillary qubits in the circuit to be executed, enable the user to correct errors in the output samples [428]. Error correction is not usually associated to quantum annealing, since AQCs cannot implement the protocols designed for gate-based devices. Nonetheless, in 2014 Pudenz et al. [429] introduced the concept of Quantum Annealing Correction (QAC). Their protocol simply prescribes to encode the problem Hamiltonian operating the following substitutions:

$$\begin{aligned}\overline{\sigma_i^z} &\leftarrow \sum_{l=1}^{n} \sigma_{i_l}^z \\ \overline{\sigma_i^z \sigma_j^z} &\leftarrow \sum_{l=1}^{n} \sigma_{i_l}^z \sigma_{j_l}^z \ ,\end{aligned} \tag{3.7}$$

effectively substituting each qubit with a collection of n qubits. Supposing the initial number of logical qubits was N, we are now considering a Hamiltonian comprising $\overline{N} = N \cdot n$ qubits. Since we use the same weights and biases in the original and in the error-corrected Hamiltonian, the energy scale of the problem is thus increased by a factor of n, which, according to the authors, is expected to suppress thermal excitations. In addition to this, the added redundancy enables to correct some errors during postprocessing via majority voting, similarly to what it is usually done with embedded problems[19]. Authors also add an additional penalty term with the aim of penalizing incomplete flips of the chains:

$$H_{\text{penalty}} = -\sum_{i=1}^{N} \left(\sigma_{i_1}^z + ... + \sigma_{i_n}^z\right) \sigma_{i_p}^z \tag{3.8}$$

where $\sigma_{i_p}^z$ are N additional qubits. Thanks to this additional term, all the qubits corresponding to the same encoded logical qubit will tend to stay parallel to $\sigma_{i_p}^z$, and, as a consequence, parallel one to the other.

[19]See Section 3.1.7.

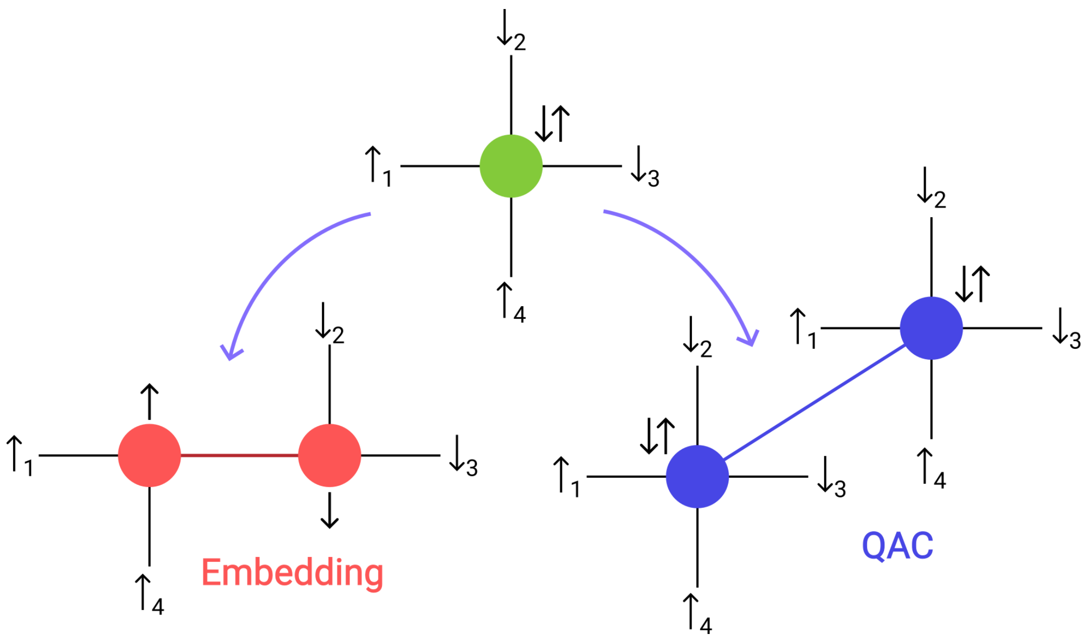

FIGURE 3.2: Depiction of the difference between embedding and QAC in a frustrated scenario. A virtual qubit (above) is characterized in this example by four ferromagnetic interactions (black) with neighbouring qubits. The upward and downward states are degenerate due to the orientation of neighbouring spins. When embedding the virtual qubit on two physical qubits (lower left) using a ferromagnetic coupling (red), the frustration induces the two qubits to become antiparallel, favoring chain breaking. In QAC (lower right) this does not apply, since each qubit in the chain is subject to the same interactions experienced by the original qubit that the chain represents. Image adapted from [430].

The proposed approach could seem very similar to a minor embedding procedure. Anyway, the dynamics of the system is expected to be fairly different in the two cases, as depicted in Figure 3.2 [430]. Consider the case (depicted in the image) of a frustrated qubit whose possible states have became degenerate due to the orientation of neighbour spins. If the qubit has been encoded via minor embedding with two physical qubits, then the frustration will elicit a chain breaking event. Conversely, the QAC protocol clones the qubit in two location, which means both qubits perceive the same energy landscape (except for a potential difference in the number of ferromagnetic couplings linking them to other clones). We can thus conclude that frustration does not induce chain breaking in QAC. As a consequence, while minor embedding typically reduces performance relative to a direct implementation of a given optimization problem, QAC is known to be effective in improving the performance when compared to a direct embedding [429], [431]. It is then natural to consider a concatenation of this two maps. Applying QAC after minor embedding can generate a graph that is not directly embeddable on the QPU, rendering the procedure ill-defined. Vinci et al. [430] present a potential (not always applicable) solution, called *square code*, which essentially exploits the particular shape of the Chimera topology to clone the qubits in a minor embedded problem in a procedural and safe way. According to the results in [430], the combined use of minor embedding and QAC achieved higher probability success than using only minor embedding. Testing with a directly embeddable problem they managed to estimate that QAC is able to retrieve the performace loss due to the minor embedding of the problem. In a particular case, forcefully expanding the directly emeddable problem with a composite minor embedding and QAC mapping achieved better results than the directly embedded instance [430].

3.3 Hybrid classical-quantum approach

In a future perspective, when AQCs or other quantum computing approaches will consistently outperform classical solvers on specific computational tasks, we can expect such new technologies to be used in a hybrid architecture. According to Callison and Chancellor [432], hybrid quantum-classical algorithm is *"an algorithm that requires non-trivial amounts of both quantum and classical computational resources to run, and which cannot be sensibly described, even abstractly, without reference to the classical computation"*. Indeed, classical computers are expected to remain for a long time the best approach to solve many simple tasks that do not require quantum computational power, which means we can forecast that a lot of applications harnessing quantum resources in the future could actually use a hybrid quantum-classical methodology. A famous example of a hybrid algorithm in gate-based quantum computing is Shor's algorithm [18], where most of the steps are entirely classical.

In AQCs the concept of hybrid classical-quantum applications can be confusing. AQCs are designed to be quantum accelerators for optimization problems, and are not expected to manage by themselves a whole computational pipeline. This means that, in a sense, AQCs are inherently created to be operated in a hybrid framework. Nonetheless, the "hybrid" keyword is recently seeing a broad use in the AQC community, probably influenced by the ubiquitous hybrid approaches in the gate-based field. Hybrid techniques for AQCs are basically divided into two main classes. Some approaches aim to overcome the resource limitation arising from the fact that practical problems typically require more qubits than are available on existing devices (often as a result of the expansion required to cast the problem in QUBO form). Such methods operate a decomposition of large optimization problems into smaller, more manageable instances that can be tackled by AQCs. The second class of hybrid methods aim to combine quantum annealing with classical annealing and optimisation techniques, in particular by using quantum annealing to perform local optimisations and classical techniques to guide the global search direction. In such cases, the objective is to exploit the best of both worlds, namely the ability of AQCs to perform quantum tunnelling and the ability of classical computers to read and copy intermediate states. Few other "hybrid" techniques have been proposed outside these two classes. As an example, a technique that manages to map multiple instances of a problem on the AQC with the same embedding, reducing the time required to embed each instance [433].

We will now present a brief literature review regarding these two classes of quantum-classical hybrid algorithms for AQCs.

Hybrid methods exploiting classical optimization

One of the first methodologies legitimately belonging to this class has been devised by Chancellor [434], who tried to answer the question of whether QA can be used to gain advantages over modern classical algorithms by exploiting both classical and quantum search techniques sequentially. He basically suggested to use AQCs only to perform local search, enhancing a global optimization that is simultaneously brought forth by a classical routine. A promising way to follow such advice is to exploit QA to obtain good feasible intermediate solutions, and then enhance them via classical postprocessing techniques[20], how was done for instance in [435] and [436]. Conversely, classical devices can be used to preprocess or re-elaborate the original problem, to then feed it into the AQC. This has been done for instance in [279], where classical computation was used to generate candidate paths to move a robots fleet on a grid in real time. The optimal combination of paths was later found among the candidate paths via QA.

Another interesting proposal came from Pastorello et al. [437], who introduced the Adiabatic Quantum Computing Learning Search (AQCLS). AQCLS exploits classical computation to find the solution of a given optimization problem that outputs the problem Hamiltonian and the annealing schedule for a computation with an adiabatic quantum machine. In other words, classical computation is used to formulate the best possible Hamiltonian to solve the original problem via QA. Contrary to previously cited methods, the classical part is not working only prior to the quantum annealing process. In fact, AQCLS is based on a classical iterative structure with repeated runs of an adiabatic quantum machine and a mechanism to induce modifications of the problem Hamiltonian towards a better encoding. The authors demonstrated that the hybrid quantum-classical algorithm converges to the solution of the optimization problem and provides a corresponding problem Hamiltonian. AQCLS is an extension of the previously introduced Quantum Annealing Learning Search (QALS) [438]. The main difference between the two algorithms comes from the different nature of the considered quantum architectures. Indeed, QALS is specialized for AQCs, while AQCLS is formulated so to be executable on a general device implementing adiabatic evolution, which can potentially also be a universal machine [103], [439]. In fact, the optimization problems tackled by AQCLS are not limited to Quadratic Unconstrained Binary Optimization (QUBO) problems. Analogously, Liu et al. [440] used a parallel algorithm that harnesses results coming simultaneously from QA and DESS (Double Elite Spiral Search) to adress a mixed-integer optimal control problem. The combined QA-DESS algorithm exploits QA to solve integer optimization with high efficiency due to the unique quantum-tunnelling-based annealing mechanism, while the classical DESS algorithm is used to optimize continuous decisions.

Hybrid techniques to decompose large problems

The combined limitations coming from the topology and the number of qubits in the AQC hardware pose a threshold on the maximum allowed complexity

[20]See Section 3.1.7 to learn about postprocessing techniques.

for an optimization problem embeddable in a given adiabatic quantum device. Researchers have been studying several ways to overcome this limitations, one being decomposing a hard problem in subproblems that can then be separately solved via quantum annealing. Among the functions belonging to the hybrid codes in Ocean SDK the most used is `qbsolv` [441], [442], which decomposes larger problems into subproblems to then solve them separately using D-Wave devices or a Tabu search[443]. Okada et al. [444] have pointed out that `qbsolv` embeds the subproblems by using an embedding of a complete graph even for sparse problem graphs. The authors have thus proposed an evolved approach that circumvents this problem by reducing the number of qubits used while embedding the subproblems. Other authors have also extended the qbsolv approach to only use the AQC on subproblems possessing a direct embedding on the QPU (in such cases the embedding process does not make use of chains) [278].

A different viable approach to tackle this decomposition task is called Core Halo (CH) partitioning. The CH partitioning consists in a min-max problem where we look for a suitable partitioning of the original graph that minimizes the maximum sum of nodes and first neighbors in each subpartition. Originally introduced in [445], it was used in [446] to decompose the QUBO formulation of a Maximum Clique problem (finding the largest fully connected set of nodes in a graph). The approach is extended by Bass et al. in [418], where they introduce an iterative version of CH partitioning. The iterative process fixes the value of some QUBO variables according to the solution obtained via quantum annealing for a chosen subproblem. At each iteration, the fixed variables are removed from the original QUBO problem modifying the biases of nearby qubits, which effectively reduces the topological complexity of the problem. This partitioning technique can in principle be applied to reduce the size of any QUBO problem. Iterative CH is designed such that an embedding is guaranteed to exist for any sub-problem to be solved, which means no time is wasted waiting for an embedding heuristic to fail, as it is the case for unembeddable graphs.

The same paper by Bass et al. [418] also describes the *freeze and anneal* technique, previously introduced by the same authors in [447]. The approach is similar to CH, but this time a genetic algorithm [448] is applied to the "population" of possible solutions to a given QUBO problem. Each solution is represented by a bit string. After several generations, bits that often assumes the same value are fixed, based on the hypothesis that their optimal value is the predicted one. Repeating this process effectively eliminates the "most obvious" variables, reducing the dimension and complexity of the QUBO problem, until it is possible to map it on the quantum device.

Bass et al. [418] also explain how the Principal component decomposition (PCD) can be applied to decompose QUBO problems. The PCD algorithm makes a series of cuts along the problem graph, where each cut removes edges that intersect the cut. This process is repeated until each of the resulting disconnected graph components, representing subproblems of the original QUBO, is directly embeddable on an available quantum annealing device. To determine where to cut the problem graph, first the nodes are laid out in a 2D plane according to the spring layout algorithm of NetworkX [449], where nodes connected by an edge are attracted to each other with a force proportional to the edge weight. Then, Principla Component Analysis is applied to the 2D coordinates of all the nodes in the graph, finding the primary axis of the graph, which can then be sliced. If a subgraph cannot be embedded due to size or connectivity issues, it is continually sliced along the primary axis until an embedding of each of its components is found. The resulting embeddable QUBO problems can be solved via QA, and their solutions must then be recomposed. This can be accomplished using a greedy recomposition that simply concatenates all subproblems solutions and then flips one random qubit in each pair of qubits that breaks a problem constraint [418].

Other notable approaches to decomposition of QUBO problems can be found in [450] and [376]. In [450], Okada et al. introduce a particular decomposition technique called *binary encoding*, which is specifically designed for problems using one-hot encoding. In [376], Zardini et al. propose a divide et impera approach where the original problem gets decomposed in smaller instances with partial overlaps. The solution of the original problem is obtained starting from the overlapping sub-solutions, which are recomposed by selecting the option in the overlapping regions that reduces the cost.

3.4 Potential hardware improvements

The previous Sections have described many techniques we can adopt to enhance the performance of AQCs. Such techniques can be defined "software-level" because they act by manipulating the problem formulation, the way the problem is embedded in the hardware, the parameters determining the duration and the schedule of the annealing process, or the obtained results. In this Section, we present those improvements that need to act at the hardware level to be implemented. In the future, manufacturers could realize devices that host or enable such approaches, but the end-user cannot implement them today. The two most promising way to evolve AQC hardware in the future are probably the concept of parity compiling, which is introduced in the next Section, and the implementation of $\sigma_i^x \sigma_j^x$ couplings that would enable universal quantum computation[21], which is presented next.

3.4.1 Parity compiling

As we discussed in Section 1.2.8 and 2.1.4, AQCs have a hard time in tackling optimization problems with higher-than-quadratic interactions (HUBO problems). The usual way to represent HUBO problems on an AQC is to introduce ancillary qubits that are then imposed equal to the multiplication of two other qubits via the insertion of additional penalty terms in the global cost function[22]. Additionally, all-to-all connected problems require large embedding comprising long, prone-to-breaking chains. Both these problems could be overcame by the *parity transformation* mapping [451], recently presented (and patented) by Parity Quantum Computing [452]. The parity transformation protocol allows to represent problems with all-to-all interactions of any order k on an AQC, requiring only a square lattice as hardware topology. The parity transformation is the generalization of the LHZ mapping, named after Lechner, Hauke, and Zoller, who introduced it in 2015 [58].

This Section presents LHZ and parity mappings as a future possible way to implement more complex optimization problems on quantum annealing hardware. Despite their great advantages, both mappings require four-local interactions in their general case, which means current AQCs cannot implement this protocols without adding several ancilla qubits, hindering their potential effectiveness. Future adiabatic quantum computers could be natively realized to host such mappings, as it is the case for NEC corporation (Japan), which recently announced their use of parametron qubits to build a unit cell for a quantum annealer using LHZ encoding [453].

LHZ mapping

To represent all-to-all interactions between N qubits, the LHZ scheme requires $N(N-1)/2$ qubits arranged in a 2D square lattice with 4-local interactions. The mapping $\sigma_i^z \sigma_j^z \to \widetilde{\sigma}_{ij}^z$ goes from the variables in the fully-connected problem to new binary variables that represent the product of the two logical qubits i and j. This means that a new qubit is introduced for each one of the $N(N-1)/2$

[21]See Section 1.2.8 to learn about the requirements for universal computation in AQCs.
[22]See Section 2.1.4.

possible quadratic terms. Staying in the ising picture (so variables assuming -1 and $+1$ values) we can conclude that the new qubit $\widetilde{\sigma}^z_{ij}$ will be equal to $+1$ if σ^z_i and σ^z_j are parallel, and -1 otherwise. Thus, quadratic coefficients are translated into local fields that act on physical qubits, and the cost function does not contain any quadratic coefficient. Nonetheless, the last fundamental requirement to perform a faithful mapping is that the new qubits $\widetilde{\sigma}^z_{ij}$ assume the desired value $\sigma^z_i \sigma^z_j$. Let's consider the four-cycle of the product variables $\widetilde{\sigma}^z_{1,2}, \widetilde{\sigma}^z_{2,3}, \widetilde{\sigma}^z_{3,4}, \widetilde{\sigma}^z_{4,1}$. Any one of the 16 possible orientations of the four qubits $\sigma^z_1, \sigma^z_2, \sigma^z_3, \sigma^z_4$ imply

$$\widetilde{\sigma}^z_{1,2}\widetilde{\sigma}^z_{2,3}\widetilde{\sigma}^z_{3,4}\widetilde{\sigma}^z_{4,1} = 1 \ , \tag{3.9}$$

since each variable appears exactly two times. Conversely, each configuration respecting Eq. 3.9 correspond to a different pair of (opposite) configurations of the four original qubits. To ensure that $\widetilde{\sigma}^z_{ij} = \sigma^z_i \sigma^z_j$ the LHZ scheme prescribes to introduce $N(N-1)/2 - N$ constraints of the form in Eq. 3.9 [58], which can be imposed by adding

$$-\lambda\widetilde{\sigma}^z_{1,2}\widetilde{\sigma}^z_{2,3}\widetilde{\sigma}^z_{3,4}\widetilde{\sigma}^z_{4,1} \tag{3.10}$$

to the cost function. This requires a four-local interaction in the device, which cannot be realized on current AQC hardware without resorting to the addition of several ancilla qubits. In the presented scheme, linear terms can be added by inserting a single ancillary qubit k set to $+1$. Indeed, the original Ising problem can be rewritten as follows:

$$\sum_{i,j,i\neq j} J_{ij} s_i s_j + \sum_i h_i s_i \to \sum_{i,j,i\neq j} J_{ij} s_i s_j + \sum_i h_i s_i s_k, \quad \text{with} \quad s_k \equiv 1, \tag{3.11}$$

effectively expressing the whole optimization problem without explicitly making use of linear terms. The added qubit must be connected to all other qubits, but this is not a shortcoming, since the LHZ scheme specifically enables all-to-all couplings. Indeed, this additional spin can be included in the programmable system by the addition of another row of physical qubits [58].

The LHZ mapping can be extended to problems described by k-local all-to-all connections, which are mapped by the scheme to a k-dimensional hyper-cubic lattice [58] (although this is impractical for realistic devices).

Parity transformation

The *parity transformation* has been recently introduced in [451] as a generalization of the LHZ mapping. The parity transformation enable the representation of an arbitrary binary optimization problems (with mixed k-body terms of various k) on a 2D square lattice. This extends the LHZ mapping since here the mapping happens without resorting to higher-dimensional structures. The parity transformation composes two mappings, named *embedding map* and *layout map*. The embedding mapping maps every k-fold product of logical qubits to a single *parity qubit*. The layout map then places the parity qubits on the physical device. Let's suppose our optimization problem comprises N qubits and M interaction terms. This means the embedding mapping will produce a problem with $M < 2^N$ parity qubits, each one representing a k-fold product of logical qubits. As in the LHZ mapping, we are then required to impose some constraints to ensure the parity qubits faithfully represent the possible combinations of the original logical qubits. Also in this case this is achieved by imposing conditions similar to 3.9 for cyclic products of the logical qubits. We remand reader to reference [451] to learn more regarding the parity transformation. The parity transformation also promises to simplify the implementation of constraints. This is usually done in AQCs by adding a penalty term in the cost functional, raising the energy of those configuration that do not satisfy constraints. Instead, in the parity approach constraints on sums, products of arbitrary k-body terms and sums over such products can be implemented without penalty terms [454].

3.4.2 Nonstoquastic Hamiltonian

Almost all existing hardware realizations of the adiabatic quantum computing concept makes use of *stoquastic Hamiltonians* [75]. This means the implemented Hamiltonians have non-negative off-diagonal matrix elements. In other words, the interactions between different quantum states do not introduce sign changes in the matrix elements. Stoquastic Hamiltonians are relatively well-behaved and can be simulated more efficiently on classical computers. The term "stoquastic" was introduced due to the similarity to stochastic matrices, such as arise in the theory of classical Markov chains. As opposed to this, non-stoquastic Hamiltonians introduce changes in sign between quantum states in the matrix elements, making it more challenging to simulate classically [455]. Adiabatic quantum computation with non-stoquastic Hamiltonians is as powerful as the circuit model of quantum computation [103]. In other words, non-stoquastic AQC and all other models for universal quantum computation can simulate one another with at most polynomial resource overhead.

In 2019, a team of researchers from D-Wave Systems has been able to implement and measure two superconducting flux qubits coupled via two canonically conjugate degrees of freedom (charge and flux) to achieve a nonstoquastic Hamiltonian [132]. The resulting coupling manifested itself as a $\sigma^y \sigma^y$ interaction in the computational basis.

Until now (2024) there has been no announcement from D-Wave Systems or other companies regarding the future commercialization or availability of an adiabatic quantum computer implementing non-stoquastic Hamiltonians.

Chapter 4

Experimental results

In this Chapter I present the main results obtained during my doctoral studies. The Chapter is divided into two main sections.

The first section presents results obtained in two main experiments where the AQC was tested in optimization tasks. In the first experiment we benchmarked different D-Wave System AQCs on a feature extraction task with applications in Machine Learning (the respective article is currently submitted for publication [1]). In the second experiment we benchmarked a D-Wave Systems AQC, a Virtual Memcomputing Machine, and the Gurobi solver on three relevant hard problems (the respective article has been published in [2]). Both experiments were conducted in collaboration with the Italian Institute of Technology (IIT). In particular, the first experiment was conducted within the IIT-Leonardo joint-lab.

The second section presents results obtained in two main experiments where the AQC was tested in sampling tasks. In the first experiment, which extended the work I conducted during my Master's Thesis, we trained a Restricted Boltzmann Machine (RBM) on a D-Wave Systems AQC, and we tested reverse annealing to enhance performance (the respective article has been published as a full paper [4] and as a conference proceeding [3]). In the second experiment we trained a fully-connected Boltzmann Machine on a D-Wave Systems AQC, showing an experimental limited quantum advantage with respect to our classical implementation on GPUs (the respective article is currently submitted for publication [24] and a further proceeding will be published later [23]). Both articles were conducted in collaboration with the Italian National Research Council (CNR).

4.1 AQCs as optimizers

As discussed in the introduction of Section 2.6.2, most articles comparing AQCs to other solvers on optimization problems follow two popular tendencies. The first is comparing AQCs performances to simulated annealing algorithms, which cannot be considered a competitive classical approach in most cases. The second is choosing a spin glass problem as a benchmark, which can make it difficult to objectively evaluate AQCs capabilities in real-world scenarios. In both the research works presented in this Section, we decided to test AQCs on real, industrially-relevant problems, and we compared their performance to a state-of-the-art classical solver, Gurobi, which is often considered the best commercially available optimization problem solver.

4.1.1 Benchmarking different generations of AQCs on a feature extraction task

Introduction

In this Section we discuss results obtained while testing various generations of AQCs on the problem of Feature Extraction [456] (FE), an important branch of computer vision that provides methods to automatically capture meaningful patterns or features associated with images, or image datasets. If properly selected, such features can be used for dimensionality reduction, acting as a basis set for a more compact latent space representation of the original data, which typically turns out beneficial to generate Machine Learning (ML) models with improved classification, recognition, and detection capabilities [457]. Among the most used classical techniques to tackle FE, one can find Principal Component Analysis (PCA) [458]–[461] and Independent Component Analysis (ICA) [462], [463]. Deep learning-based methods like Convolutional Neural Networks (CNNs) [464], [465] and Recurrent Neural Networks (RNNs) [466] also make use of FE techniques. Such approaches can be computationally demanding and may require substantial processing power depending on the size of the dataset involved or the level of sophistication of the selected algorithm.

AQCs offer practical means to explore quantum approaches to FE because of their ability to handle a high number of variables [467], [468], which is enabled by the availability of thousands of physical qubits and connectivity that characterize such devices [21]. O'Malley et al. [405], [469] were among the first to test AQCs to perform FE tasks. In [469], they introduced a workflow to perform Non-negative Binary Matrix Factorization (NBMF), which allows to exploit AQCs to factorize a collection of images as the product of two matrices, one of which must be binary-valued. The continuous matrix is interpreted as the collection of basis "feature" images, while the binary one is the "weights" matrix that lists which feature images must be summed up to recompose the original images (see the Methods for an extensive explanation). Such an approach is widely based on the method to factorize matrix-vector multiplication on AQCs previously introduced by Li et al. [470]. In [405], they managed to boost the performances of the quantum FE algorithm by modifying the shape of the annealing schedule, similar to what was done for a similar problem in Ref. [242].

In our experiment, we explored opportunities, challenges, and current limitations of different generations of D-Wave quantum annealers when utilized to address FE tasks using NBMF methods. We compare the legacy lower noise D-Wave 2000Q (now dismissed) with 2038 qubits arranged in a Chimera topology [471] (2kQ), the D-Wave Advantage 4.1 [21] with 5627 qubits arranged in a Pegasus topology [159] (Adv1), and the most recent D-Wave Advantage2 prototype 1.1 [472] with 563 qubits arranged in Zephyr topology (Adv2). Such devices represent three important steps in the history of AQCs, namely an increase in topology complexity, going from Chimera (8 couplers per qubit) to Pegasus (15 couplers per qubit) to Zephyr (20 couplers per qubit).

The comparison between different quantum annealing hardware allows us to assess the evolution in the computational capabilities of AQCs and provides us indications about the most relevant working conditions for which this technology is expected to challenge classical digital approaches. Our study was inspired in particular by O'Malley et al. [469], where the NBMF methodology was applied to extract salient features from gray-scale images of human faces. We considered instead an extended, more complex, dataset comprising low-resolution RGB satellite images of airplanes, using the Adv1 processor to also double the maximum number of features extracted with the NBMF methodology in [469] (70 vs. 35). Additionally, we applied a tuning procedure to choose the optimal values for both the problem parameters and the internal parameters of the AQCs, which allowed us to get better results in the computational time at our disposal.

Appendix C presents the theory of NBMF, explains how to solve it via an iterative procedure that makes use of an optimization step, and explains how the optimization step can be cast in QUBO form. The next Sections will instead focus on presenting the dataset, the necessary embedding, and the obtained results.

Description of the dataset

We consider a dataset of satellite images that is publicly available under the CC-BY-SA license at https://www.kaggle.com/rhammell/planesnet. The

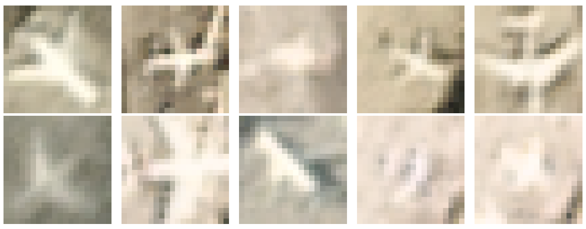

FIGURE 4.1: A selection of images of the *aircraft* class.

FIGURE 4.2: A selection of images of the *not-aircraft* class.

dataset is actually composed of two sub-datasets containing respectively 8000 and 24000 image files in *.png* format. These two sub-datasets contain images that were previously classified as either *aircraft* or *not-aircraft*, examples of which are reported in Figs. 4.1, 4.2. The images display a squared aspect ratio and are composed of 20×20 RGB pixels which can be considered representative of the low-resolution images typically provided by low-cost constellation satellites. For the purpose of this work, we only consider *aircraft* images, which are transformed into a grey scale. Each of such images depicts a single near-centered aircraft at various zoom levels, in-plane orientations, and atmospheric and light conditions. Wings, tails, and tips of the aircraft are fully contained in the perimeter of the images in most cases. The number of details embodied in this dataset yields an overall complexity that we expect to capture only by using a relatively large number of features.

Embedding the problem

Any QUBO problem can in principle be submitted to an AQC [35]. The only limitations regard the total number of variables and the topology of the problem, namely the amount and structure of non-zero quadratic coefficients in Eq. C.5 in the Appendix. Given a problem with an acceptable number of variables, a proper *embedding* procedure is required whenever the mathematical structure of the problem cannot be mapped directly on the AQC topology [171]. Such a procedure involves connecting multiple physical qubits together via a strong ferromagnetic coupling J_{chain}, which makes them behave like a single two-level quantum system, i.e. a logical qubit. The embedding approach augments the effective connectivity for each logical qubit and eventually allows for the mapping onto the AQC of problems with up to all-to-all connectivity. Obtaining a suitable embedding is generally non-trivial, and it is typically addressed through a procedure known as *minor embedding.* [151].

The mathematical structure of the NBMF problems considered in this work implies the presence of k binary variables connected in an all-to-all fashion by the quadratic coefficients b_{ij} appearing in Eq. C.6 in the Appendix. In principle, any of these coefficients could become equal to zero, but this condition can change at each iteration, and in general, most of the coefficients assume nonzero values. For this reason, it is practical to assume the problem is represented by a fully connected topology of k binary variables. A favorable consequence of this hypothesis is that the embedding procedure has to be performed only once for each tested k value. For the sake of completeness, we report the details of the minor embedding at various problem sizes in Table 4.1. The embeddings have been obtained using the `minorminer.find_embedding` function from Ocean SDK [179].

	2kQ		Adv1		Adv2	
k	q	l_{chain}	q	l_{chain}	q	l_{chain}
10	34	3.40	16	1.60	16	1.60
20	121	6.05	52	2.60	47	2.35
30	273	9.10	115	3.83	93	3.10
40	505	12.62	199	4.97	161	4.02
50	791	15.82	297	5.94	252	5.04
60	-	-	422	7.03	-	-
70	-	-	555	7.93	-	-

TABLE 4.1: Number of qubits (q) and average chain length (l_{chain}) associated with the embeddings at different problem sizes k. Results are obtained running the `minorminer` software multiple times until there was no improvement in the required number of qubits for ten consecutive trials. Missing values in the table correspond to those cases where minorminer did not return any embedding after ten consecutive trials.

Optimization of the NBMF hyperparameters

The NBMF-based FE algorithm relies on some hyperparameters, namely the number of epochs of the iterative process, the initialization of the H matrix, and the regularization parameter α. The associated values are chosen within a preselected range in order to minimize the reconstruction error $||V - WH||_F$ at the end of the iterative NBMF process. In practice, this is achieved by conducting preliminary runs on a dataset of 625 images with $k = 50$, which are typical values for problem sizes relevant to this work. For this analysis, we

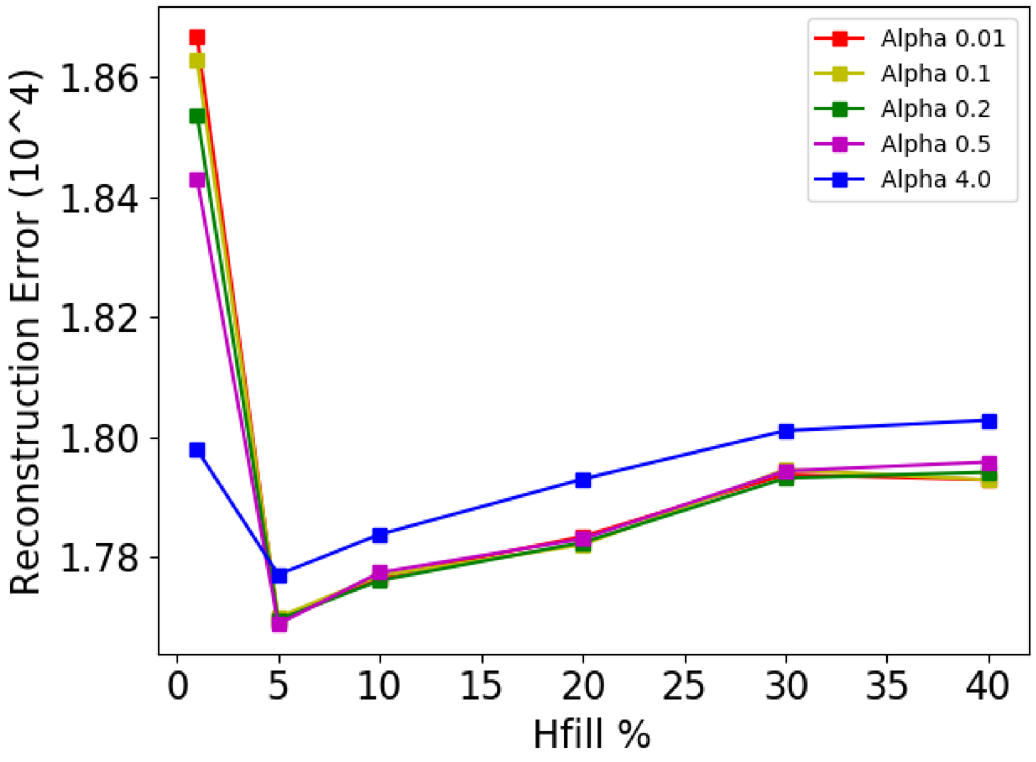

FIGURE 4.3: Hyperparameters selection: Reconstruction error as a function of H_{fill} at different α values. Results are obtained using the classical Gurobi solver for a dataset of 625 images with $k = 50$. Each datapoint is the mean of three independent runs using three different random seeds.

exclusively use the Gurobi solver. In this work, Gurobi version 9.5.0 was used, and calculations were performed on a laptop CPU Intel i5-11400H.

Preliminary experimental results showed that the reconstruction error between successive NBMF iterations decreases rapidly within the first 5 steps. For this reason, from now on, we set $n_{\text{epochs}} = 5$ as a meaningful number of iterations for the process. As for the initialization of H, we opt for a random uniform filling with a defined density H_{fill}, which represents the percentage of ones in it. Figure 4.3 shows the reconstruction error $||V - WH||_F$ as a function of H_{fill} at different α values. The error varies slightly for different values of such hyperparameters, accounting to few percentage points across all the tested combinations for α and H_{fill}. The optimal combination resulting in the lowest reconstruction error was $\alpha = 0.1$ and $H_{\text{fill}} = 5\%$.

Figure 4.4 shows the dependence of the sparsity of the final H after 5 iterations as a function of α and H_{fill}.

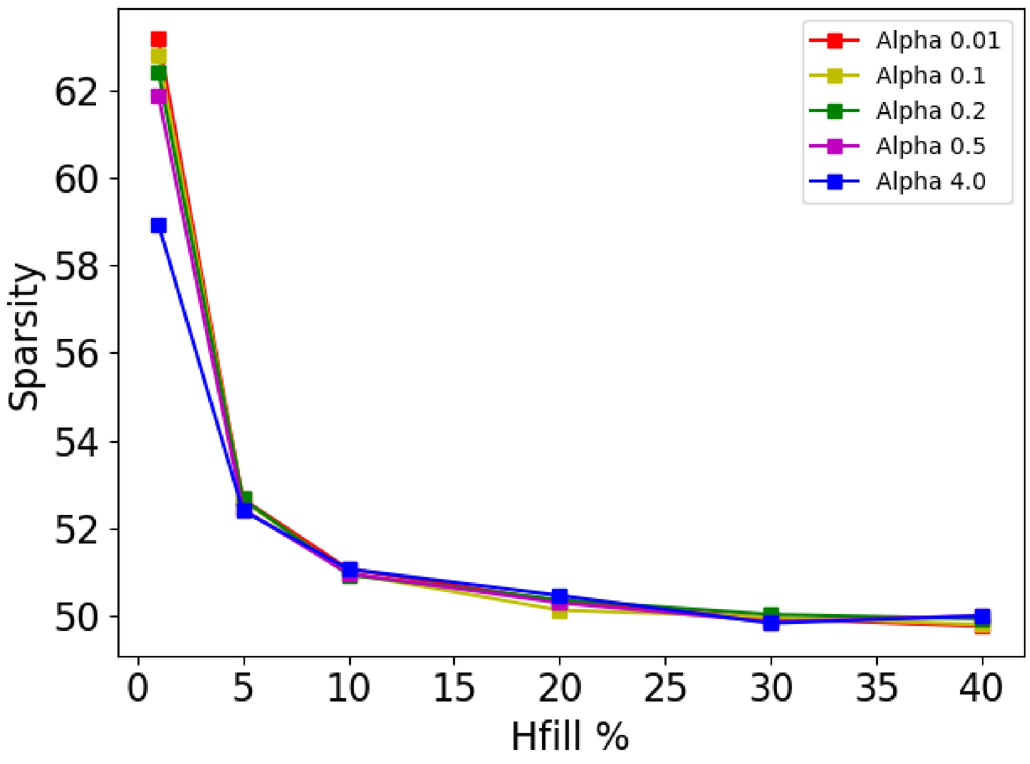

FIGURE 4.4: Final (percent) sparsity of H matrix as a function of H_{fill} at different α values. Results are obtained using the classical Gurobi solver for a dataset of 625 images with $k = 50$. Each data point is the mean of three independent runs using three different random seeds.

Tuning of the AQC parameters

Ocean SDK enables users to adjust multiple parameters that influence the dynamics of the physical device. It has been demonstrated that the performance of AQCs can be significantly boosted upon tuning these parameters, resulting in improvements of up to two orders of magnitude in the average time to solution [2], as showed in the previous experiment (Section 4.1.2). The parameters with the most substantial impact on performance are two:

- t_{ann}: the annealing time, namely the time duration of a single quantum annealing cycle.
- I_{chain}: parameter controlling the intensity of the chain coupling J_{chain} inside the AQC according to the following equation:

$$J_{\text{chain}} = I_{\text{chain}} \frac{2N_J}{N_{\text{vars}}} \sqrt{\frac{\sum_{\{i,j\},i\neq j} J_{ij}^2}{N_J}}, \tag{4.1}$$

  where N_J is the total number of quadratic connections in the original (not embedded) QUBO problem, and N_{vars} is the number of logical variables in the original QUBO problem (which implies $2\frac{N_J}{N_{\text{vars}}}$ is the average number of connections per logical variable). Note that in Ocean SDK $I_{\text{chain}} = 1.414$ by default, as prescribed in the function `uniform_torque_compensation` [413].

Determining the optimal values for these parameters is a non-trivial task in practical terms. With regard to the annealing time, one could rely on the adiabatic theorem that prescribes maximizing t_{ann} to obtain high-quality solutions [116], [117]. However, as described in Section 4.1.2, it is known that due to the coupling of the quantum state to the environment, longer annealing times can trigger quantum decoherence that deteriorates the overall device performance. As a result, only a direct tuning of this parameter can provide reliable indications of the optimal values to be used for tackling the problem of interest.

We therefore tested various combinations of t_{ann} and I_{chain}, and we selected the one that resulted in the lowest average gap from the global optimum for `find_H` instances. The gap for each problem instance is defined as follows:

$$\text{gap} = \frac{(C_{\text{best}} - C_{\text{optimal}})}{C_{\text{best}}} \cdot 100 \ , \tag{4.2}$$

with C_{best} being the cost of the best solution found with the quantum device, and C_{optimal} being the global optimum of each problem instance found by Gurobi. Note that by definition this gap is bounded $0\% \leq gap \leq 100\%$.

We performed the test on 100 *single-column problems*, where a single column of the H matrix is optimized (see Eq. C.4 in the Appendix). Such test set was created by initializing V with 500 random images from the dataset, and H with random binary digits with $\mathrm{H_{fill}} = 5\%$. The initialization was repeated for 5 different seeds, then executing for each case `find_W` using Gurobi, and then selecting the first 20 column problems of `find_H` for each seed, for a total of 100 problems. Since we aim to use the optimal parameters to enhance the computation at different problem sizes, we decided to perform the tuning procedure at $k = 50$ for Adv1, while we chose $k = 30$ for Adv2 and 2kQ. This way the optimal parameters are obtained at a problem size that is not trivial and at the same time is not close to the maximum achievable size on the hardware (see Table 4.1).

To conduct a fair comparison, we fixed the overall QPU time t_{QPU} available for each combination of parameters. This is achieved by varying the number of annealing samples to compensate for the variable annealing time used in the analysis. Specifically, we decided to set $t_{\mathrm{QPU}} = 0.200s$ for each single-column problem, which we found sufficient to allow for relatively good solutions for the problems at hand while staying within the overall time budget at our disposal. In detail, this value allows for a number of samples per problem ranging from 919 for $t_{\mathrm{ann}} = 100\mu s$ to 1644 for $t_{\mathrm{ann}} = 4\mu s$ (the longest and shortest annealing times tested, respectively). The reason why the number of samples changes so little with respect to t_{ann} is due to the impact of the delay and readout times per sample, which contribute to t_{QPU} especially when t_{ann} is low[1]. Note that t_{QPU} does not include the time required to communicate with the QPU over the Internet, which has a fixed duration.

The left image in figure 4.5 shows the average gap obtained at $k = 50$ for Adv1, while the right image in the same figure shows the percentage of *broken samples*. A sample is considered broken if at least one of the chains in the embedding contains antiparallel qubits.

Figure 4.5 shows that the average gap appears to be independent of the annealing time. This is in contrast with the heatmap on the right, which shows that, on average, samples obtained using a shorter annealing time contain more broken samples. Apparently, the higher number of samples collected for shorter annealing times compensates for the lower average quality of the collected samples. Additionally, the fluctuations induced by the outer noise could be boosting the ability of the system to reach the global minimum, as suggested in [473]. On the other hand, the chain strength heavily influences the quality of the samples.

[1]See Section 1.3.5.

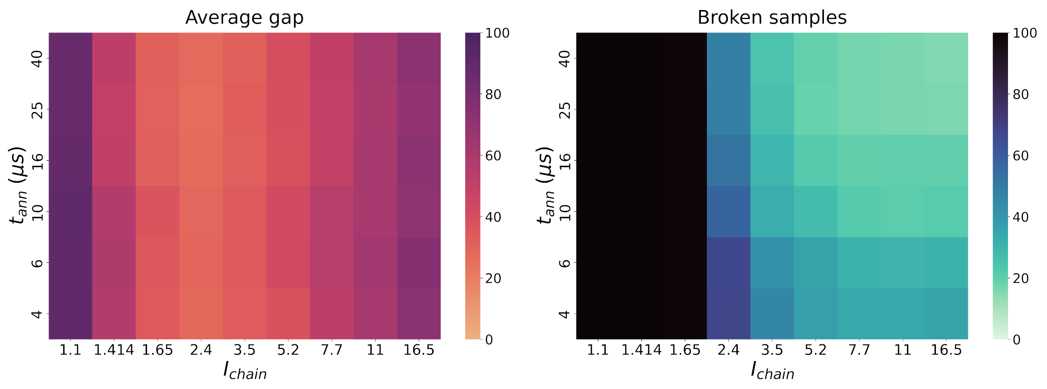

FIGURE 4.5: Average gap and average number of broken samples for every combination of annealing time t_{ann} and chain strength I_{chain} in the case $k = 50$ for Advantage 4.1. Optimal setting is $t_{ann} = 25$ and $I_{\text{chain}} = 2.4$, corresponding to gap 26.8%.

If a sufficiently strong I_{chain} is imposed, the annealing process should produce samples whose chains are composed of parallel qubits. Such expectation is confirmed by the right heatmap in figure 4.5, which shows how higher I_{chain} values correspond to a lower average number of broken samples. Nonetheless, reducing I_{chain} lowers the energy gap between different configurations, raising the probability of both quantum tunneling and thermal fluctuations. The optimal value for I_{chain} must balance these effects. The combination $t_{\text{ann}} = 25$ and $I_{\text{chain}} = 2.4$ resulted in the optimal gap of 26.8% and 48.1% broken samples, with only 1.43% of the chains broken. The low average number of single broken chains could explain the limited impact of a high number of broken samples. The optimal setting is to be compared with the D-Wave base setting $I_{\text{chain}} = 1.414$, which resulted in gap values ranging from 49.4% (at $t_{\text{ann}} = 25$) to 58.19% (at $t_{\text{ann}} = 6$), a percentage of broken samples $> 99.97\%$, and a percentage of broken chains $\in [19.25\%, 22.11\%]$ (varying the annealing time). The parameter tuning procedure effectively halved the number of samples containing broken chains, and reduced the number of broken chains by more than an order of magnitude, drastically improving the expected average gap from 49.4% to 26.8%.

We conducted an identical analysis in the case $k = 30$ for D-Wave 2kQ, Adv1, and Adv2. Figure 4.6 shows the obtained results. We can observe that Adv1 seems to be the most resilient to chain breaking, with a resulting lowest gap score. As evident also in Figure 4.5, for each solver we have a "critical" value for I_{chain} below which the number of samples containing broken chains rapidly increases from almost zero to almost 100%. In all cases, the I_{chain} value closest one to this transition point also corresponded to the lowest gap. Figure 4.6 also confirms the weak dependency of the lowest gap on the annealing time, as was already noted for Figure 4.5.

Gap dependency on problem dimension and wall time

After tuning the NBMF hyperparameters and the quantum devices parameters, we can now compare the performance of quantum devices provided by D-Wave (hybrid workflow) and the classical solver Gurobi (purely classical workflow) on single-column problems. Our goal is to understand if there are indications of quantum advantage or conditions in which such quantum advantage could be reached earlier, for the set of problems considered in this work. The analysis is conducted at varying problem sizes, varying the value of k from 10 to 70, using the average gap defined in Eq.4.2 as an evaluation metric. For each problem size k, we establish a maximum allowed wall time for the AQCs, which depends on the average time required by Gurobi to solve problems of the same size. The computational time needed by the classical solver serves as an indicator of problem complexity at each size. This specific methodology enables us to employ the average gap obtained by AQCs as a metric to evaluate the quantum solver's ability to match classical capabilities.

We compared the solvers on a total of 250 single-column problems coming from 5 different initializations of matrix V and H (50 problems per initialization). We used the optimal hyperparameters and per-solver optimal I_{chain} and t_{ann} found previously. On average, Gurobi required 0.09 ± 0.02 to solve problems at $k = 10$, and 0.23 ± 0.12 to solve those at $k = 70$.

Figures 4.7**a**-**c** show the results obtained for the different solvers and different maximum wall times. For each size, when *mult*$= n$ it means that the quantum solver had a runtime n times longer the average time required by the classical solver to find the optimal solution at that problem size. Figures 4.7**d**-**f** present the same data but grouped with respect to the *mult* value so that it is easier to compare the performances of the solvers. Violin plots in Figs.4.7**g**-**h** show the distribution of the gaps obtained by sampling at different k values using the three D-Wave devices.

Figures 4.7**a**-**f** highlight a monotonic increase. Given that we already corrected the runtime according to the problem complexity, this means that the quantum solvers performance is degrading as k increases, if compared to the classical solver.

A performance boost is to be expected at the specific k value where the tuning procedure has taken place. We performed the optimal parameter search at $k = 50$ for Adv1, while 2kQ and Adv2 have been tuned at $k = 30$. 2kQ and Adv1 at mult$= 0.1$ do indeed show a swift decrease in the average gap at $k = 30$

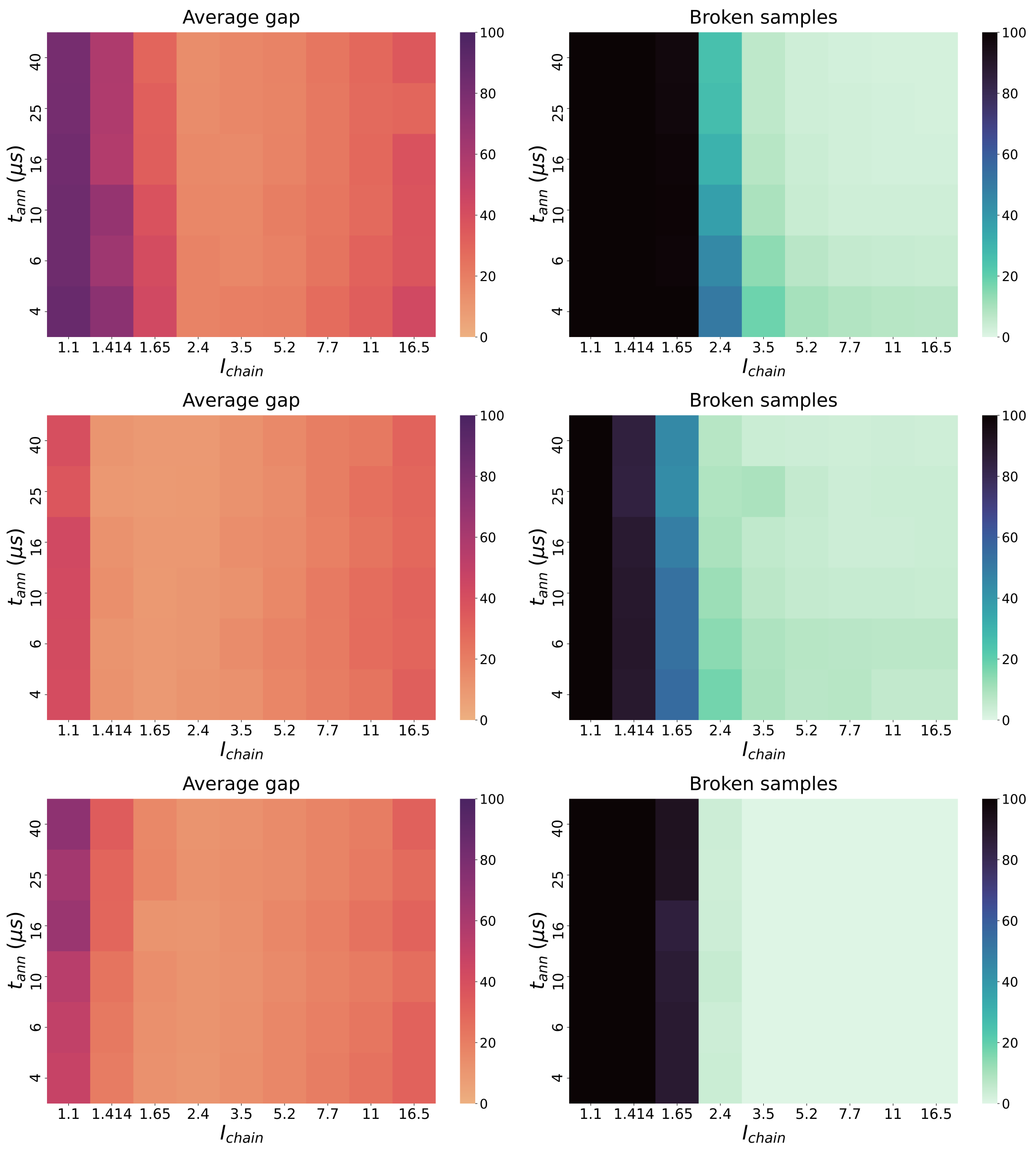

FIGURE 4.6: Heatmaps showing the average gap and average number of broken samples for every combination of annealing time t_{ann} and chain strength I_{chain}. From top to bottom: 2kQ, Adv1, Adv2. All devices were tested at $k = 30$. The number of broken samples exhibit a transition-like behaviour around the optimal value of I_{chain} for the lowest gap.

and $k = 50$, respectively, but the change is not evident at any other multiplier, nor in the Adv2 case. This behavior suggests that the parameter tuning procedure is boosting performances also at different problem sizes, suggesting that the optimal parameters found possess good transferability.

In addition to the previous evaluations based on the average optimal gap obtained on multiple problems, the violin plots in Figs.4.7**g**-**h** allow us to compare the different solvers with respect to the distribution of all samples. In Figure 4.7**g** we can appreciate how, for $k \geq 40$, 2kQ tends to produce samples with a much higher gap if compared to Adv1. The impact of this behavior is confirmed by Figs.4.7**d**-**f**, where the average gap attained by 2kQ rapidly grows for $k \geq 45$, while Adv1 and Adv2 are less affected. This detrimental effect is probably due to the different chain lengths required to embed problems on the three solvers (see Table 4.1). At $k = 40$ the average length of the chains of the 2kQ embedding rises above 12 qubits, to reach almost 16 qubits at $k = 50$. On the other hand, Adv1 and Adv2 are much closer to each other, reaching 5.94 and 5.04 average length, respectively, at $k = 50$, which results in similar distributions in figure 4.7**h**.

Reconstruction error after the iterative process

After having analyzed the quantum hardware performance in solving single-column QUBO problems, we focused on analyzing the performances in solving the full interactive process that constitutes the FE NBMF-based algorithm. To this end, we compared the fully classical and hybrid quantum-classical FE workflows to extract $k = 50$ features from a dataset with $m = 500$ images. For the hybrid workflow, we used Adv1 with 2000 samples per each single-column `find_H` problem, setting the optimal parameters $I_{\text{chain}} = 2.4$, $t_{\text{ann}} = 25$.

We performed runs with $n_{\text{epochs}} = 5$ iterations for both D-Wave and Gurobi, which led to a decrease in the reconstruction error $||V - WH||_F$, as shown in Figure 4.8 **d**. The Figure also shows that the `findH` step in the hybrid workflow increases the reconstruction error achieved in the same epoch during the `findW` step. This means that, given the W matrix found at a certain iteration, D-Wave is finding an updated H matrix that is worse than the previous one. This is compensated at each new epoch by the `findW` step (Eq.C.1 in the Appendix), yielding a reduction of the reconstruction error during the overall process. In absolute values the purely classical workflow is observed to outperform the hybrid one.

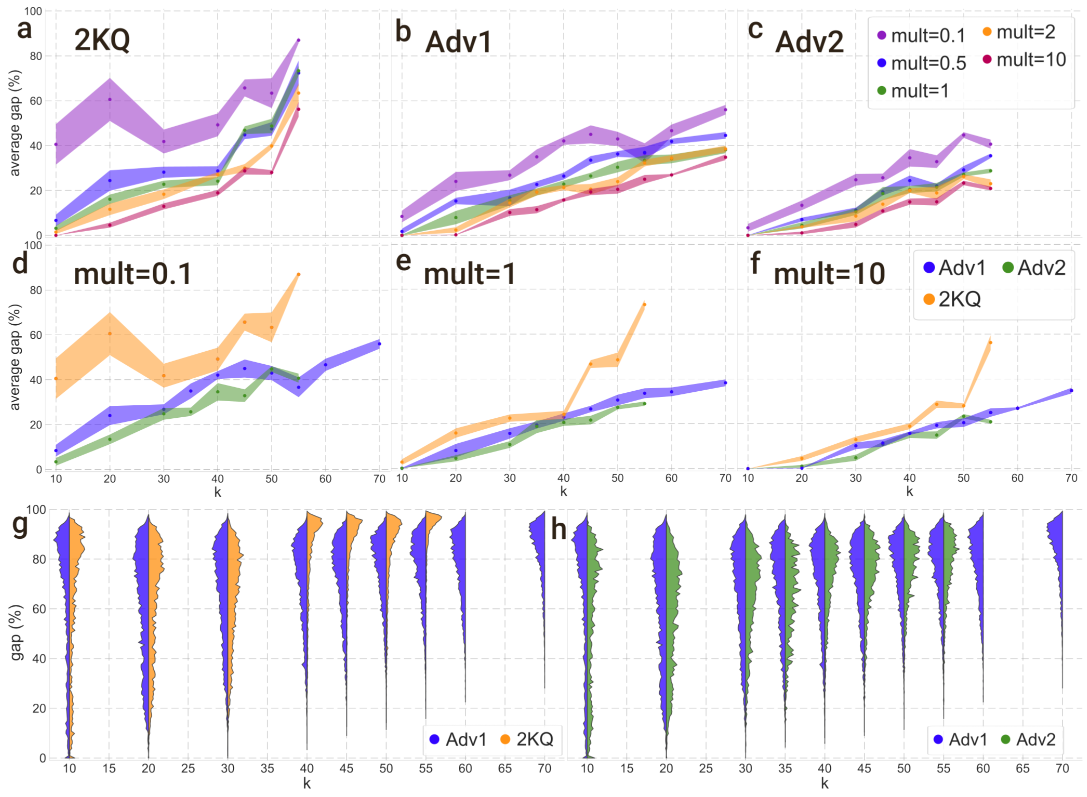

FIGURE 4.7: Performance analysis of the Advantage 4.1 (Adv1), the Advantage2_1.1 prototype (Adv2), and the 2000Q (2kQ) quantum computers. Plots **a** → **f** display the average gap from the exact solution as a function of the problem size k. Each data point is estimated by averaging the best gap obtained by the selected D-Wave QPU on 250 distinct single-column problems. The shaded area represents the standard deviation. The runs were executed using the tuned optimal parameters for each solver. The maximum wall time allowed for each D-Wave run is a multiple of the average time that Gurobi required to solve the same instance of the problem. Plots **g** and **h** compare the distribution of samples produced by the three D-Wave solvers. For every k, 10,000 samples were uniformly extracted from the whole collection of samples obtained from the previous runs on the 250 single-column problems. The samples produced by 2kQ display a distribution peaked towards higher gaps for $k \geq 40$, while samples produced by Adv1 and Adv2 display broader distributions.

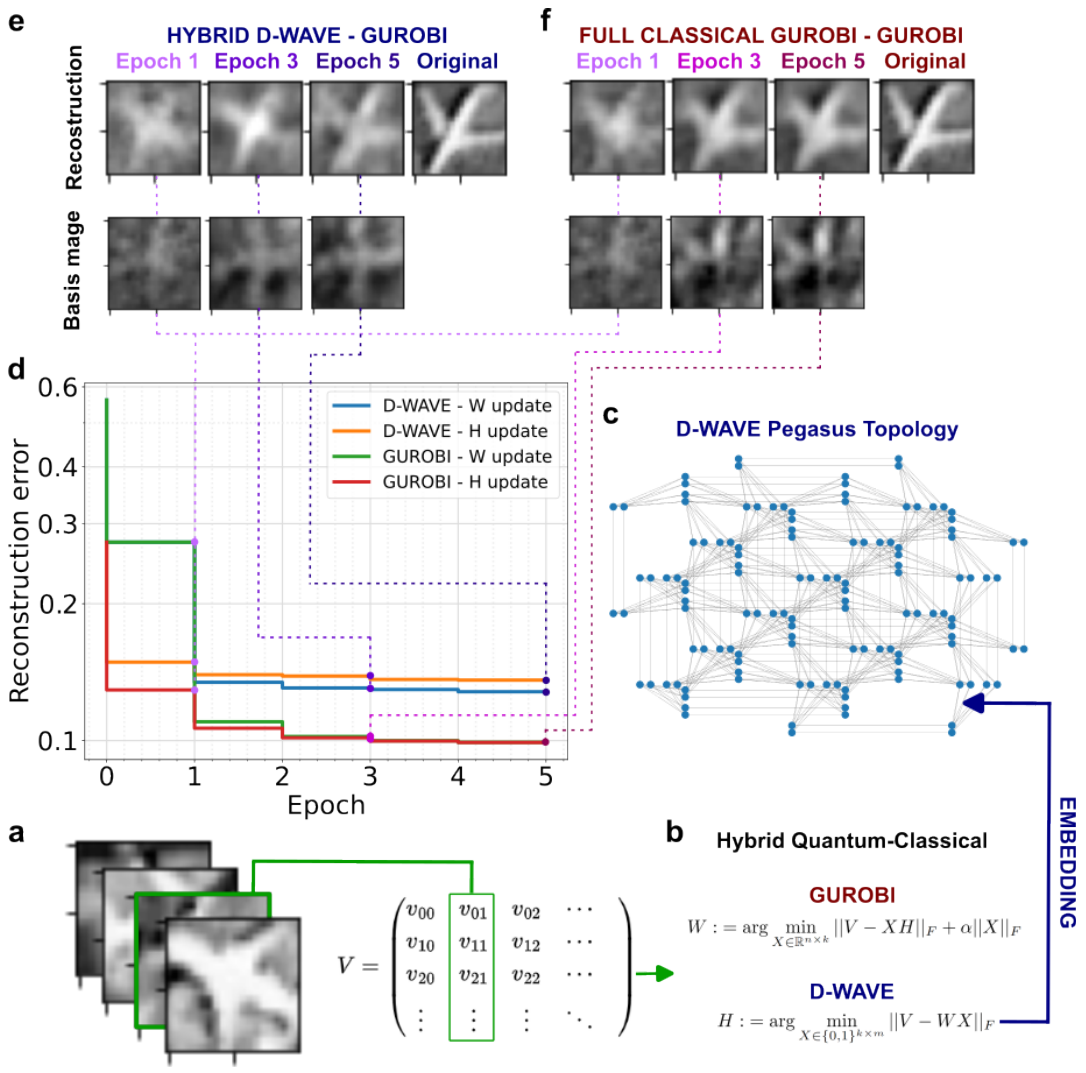
e
HYBRID D-WAVE - GUROBI
Epoch 1
Epoch 3
Epoch 5
Original
Recostruction
Basis mage
f
FULL CLASSICAL GUROBI - GUROBI
Epoch 1
Epoch 3
Epoch 5
Original
d
Reconstruction error
Epoch
D-WAVE - W update
D-WAVE - H update
GUROBI - W update
GUROBI - H update
c
D-WAVE Pegasus Topology
EMBEDDING
a
$V = \begin{pmatrix} v_{00} & v_{01} & v_{02} & \cdots \\ v_{10} & v_{11} & v_{12} & \cdots \\ v_{20} & v_{21} & v_{22} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}$
b
Hybrid Quantum-Classical
GUROBI
$W := \arg\min_{X\in\mathbb{R}^{n\times k}} ||V - XH||_F + \alpha||X||_F$
D-WAVE
$H := \arg\min_{X\in\{0,1\}^{k\times m}} ||V - WX||_F$

FIGURE 4.8: NBMF algorithm Workflow on AQC (D-Wave) and Gurobi. **a** In the lower-left panel are reported some samples of the $\sqrt{n} \times \sqrt{n}$ satellite images of aircrafts. The m images are flattened and stacked to form the $n \times m$ matrix V. **b** The optimization strategy is based on an iterative updating of the continuous values matrix W (always on Gurobi) and the binary matrix H (on D-Wave or Gurobi). **c** The optimization step to update H is embedded on the Pegasus, Chimera, and Zephyr graphs of the Adv1, 2kQ, Adv2 devices, respectively. **d** The reconstruction error $||V - WH||_F$, in logarithmic scale, after `find_W` and after `find_H` is shown at each epoch for the quantum-classical workflow and the full-classical one. **e-f** Reconstruction of a sampled image at the epoch 1, 3, and 5 respect the original image (series of images above), the images on the left side are the reconstruction with the quantum-classical workflow and in the right side the reconstruction obtained by the fully-classical workflow. The series of images below, instead, contains a sampled basis image, which is a column of W, at the epochs 1, 3, and 5.

Figures 4.8**e-f** show an example of dominant feature images (basis images from matrix W) obtained by running the quantum and classical NBMF workflows. The corresponding basis images, shown at different epochs, resemble some sort of spherical harmonics. As expected from the reconstruction score, we can visually verify that from epoch 3 to 5 the feature images are modified only slightly. The same figures also display examples of image reconstructions across the iteration process. As expected from the reconstruction score, at visual inspection the fully classical workflow provides a slightly sharper reconstructed image.

The hybrid workflow tends to combine more feature images to reconstruct images in V. Indeed, 34.6% of the elements in the final matrix H are equal to 1, as opposed to only 14.6% in the classical workflow. Thus, the hybrid workflow produced a potentially more informative decomposition of the images from the dataset into base features.

Summary of the obtained results

We analyzed the capabilities of AQCs on a task with applications in machine learning, namely performing feature extraction (FE) from a dataset of low-resolution satellite images of airplanes, exploiting an approach based on Non-negative Binary Matrix Factorization (NBMF). The NBMF is implemented through a hybrid iterative algorithm where the portion of the workflow corresponding to the optimization of the binary latent H matrix is offloaded to AQCs. This methodology can be used to perform FE on large datasets thanks to the possibility of decomposing the original latent H matrix optimization problem into a set of independent problems corresponding to single columns of the H matrix.

We considered three generations of AQCs provided by the D-Wave company, namely the legacy device 2000Q, Advantage 4.1, and the most recent prototype Advantage2_1.1. First, we devoted consistent efforts to fine-tuning the algorithm hyperparameters and the AQCs parameters. We noted an interesting behaviour, namely an abrupt increase in the number of samples containing broken chains below a specific value of I_{chain}. This transition-like effect happens in the correspondence of the optimal value for I_{chain}, which means the value that correspond to the lowest gap.

Using the optimal AQC parameters, we were able to reduce the average number of broken chains by one order of magnitude while halving the expected average gap from the optimal solution (results obtained on Advantage 4.1). Hence, we compared the performance of three hybrid quantum-classical workflows based on the three AQCs against that of a purely classical workflow based on the state-of-the-art Gurobi classical solver running on classical digital hardware. As an evaluation metric, we used the solution gap from the global optimum obtained at fixed runtimes and averaged over a selection of problem instances.

We observed all quantum solvers to provide a similar qualitative behavior in terms of gap performances as a function of the problem size, i.e. k. In particular, we observed the average gap to the exact solution to increase along with k. We can nonetheless appreciate a slight but statistically significant quantitative difference in the performance of the three AQCs, with Adv2 achieving the lowest gap values, Adv1 following close, and 2kQ falling short, particularly on larger problem sizes. This effect is more evident at shorter runtimes. The fast degrading of the performance of 2kQ for $k > 40$ is probably correlated with the excessive chain length required to embed such problems on the sparse Chimera topology, which also notably affects the distribution of samples with respect to the attained gap. These observations suggest an incremental overall performance with the more recent solver generations, which seems to correlate mainly to the average chain length found by the embedding procedure. Such conclusion supports the idea, common in literature, that the topology of the available hardware currently represents the main bottleneck for adiabatic quantum computation [150], [279], [376].

During the iterative process, the optimization of H at fixed W made with D-Wave at a relatively large k increases the reconstruction error rather than reducing it as in the case of utilization of a classical solver. For small k, however, the average gap on single instances of the optimization problem remains close to zero for all quantum solvers. In particular, we observed that both Adv1 and Adv2 are capable of solving all the submitted instances at $k = 10$ with 0% gap (i.e. exactly) within the average time required by the Gurobi solver (mult$=$ 1). This result suggests that recent generations of AQCs have finally reached competitive performances for industrially-relevant, small-sized problems.

At least for the set of problems considered here, the classical solvers provide in general still the reference tools. There are conditions, however, where the AQCs start to provide alternative solutions with comparable performances. In this perspective, the first applications where AQCs are expected to become competitive with classical counterparts are those where limited and short running times are required (e.g. real-time or near-real-time applications).

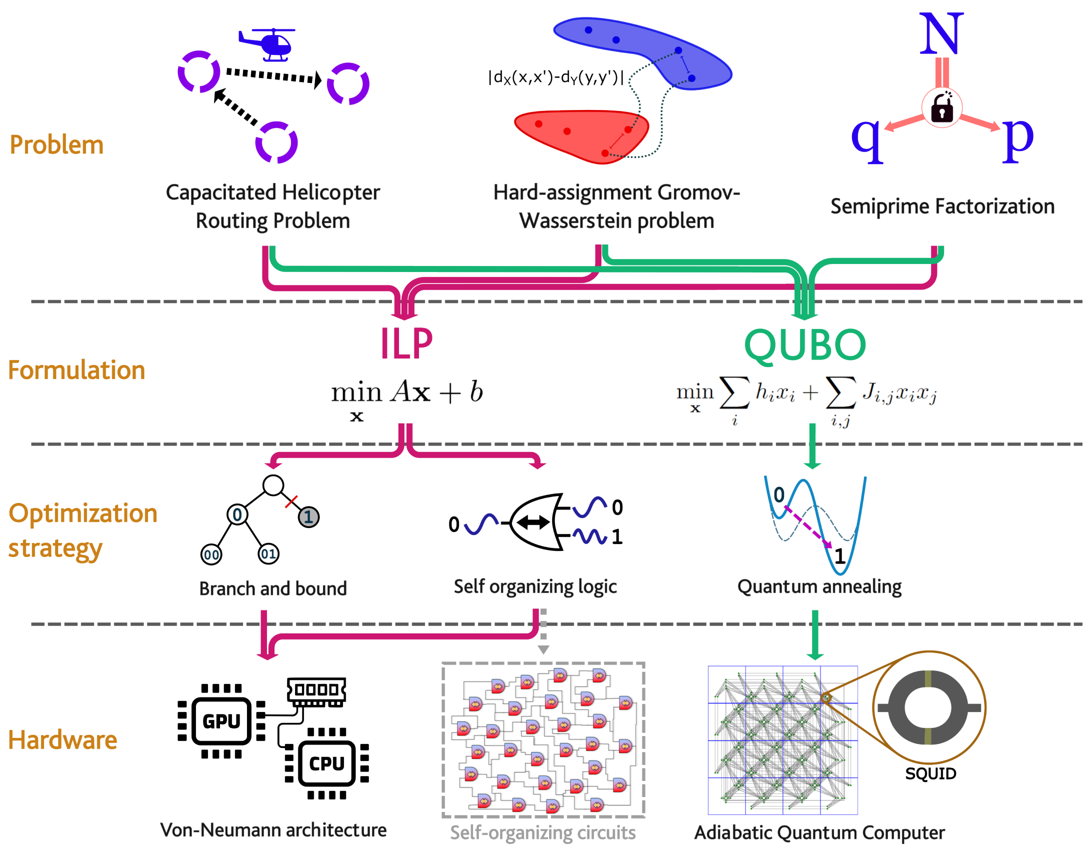

FIGURE 4.9: From the problem to the computing hardware. Three problems are formulated in ILP and QUBO forms and solved with three different solvers: i) the Gurobi optimization software based on branch and bound and other heuristics; ii) a Virtual Memcomputing Machine exploiting self-organizing logic; and iii) a Quantum Annealer. These solvers are physically implemented on hardware based on the Von-Neumann architecture or on an adiabatic quantum computer based on superconducting qubits. Memcomputing machines could be implemented on self-organizing memristor-based circuits.

4.1.2 Benchmarking AQC, Memcomputing Machines, and Gurobi solver on hard optimization problems

In this experiment we assess a D-Wave System Advantage 4.1 AQC (DWA from now on) and a Virtual Memcomputing Machine (VMM) and compare them to a classical solver (see Figure 4.9). DWA and VMM can solve equivalent problems and they represent a quantum (not gate-based) and classical (not quantum) approach, respectively. As we know, DWA solves Quadratic Unconstrained Binary Optimization problems (QUBO). Instead, VMM solves Integer Linear Programming problems (ILP) by mapping them to a physical circuit, with the physical circuit evolution emulated via a software[2]. The two approaches can

[2]See Section 2.5.2.

be directly compared with an established baseline, namely the Gurobi suite of optimization methods [474].

We benchmark these approaches on three difficult and well-known combinatorial problems of broad interest that can be expressed in ILP and QUBO format: the Semiprime Factorization problem (FP), the Hard-Assignment Gromov-Wasserstein problem (GWP), and the Capacitated Helicopter Routing Problem (CHRP). The security of large part of the public key cryptography (RSA[264]) is based upon the assumed intractability of FP. GWP is a particularly hard example of the optimal transport theory [475], [476] and is an instance of the well-known Quadratic Assignment Problem [477], a fundamental combinatorial problem. CHRP is an industrial optimization problem concerning the route scheduling of helicopters.

We also compare the baseline performance of VMM and DWA to their performance with enhanced parameters. The baseline imply using the two solvers with default setting and parameters, as proposed by the respective vendors. The enhanced parameters are instead obtained after a parameter tuning procedure conducted at a specific size of each tested problem, with the aim of reducing MFST as much as possible. The enhanced parameters for each problem are then used at each problem size for that solver. We show how the scalability of the problem changes after such procedure on both solvers.

For an accurate comparison, we introduce the concept of Mean First Solution Time (MFST)[3] which is the expected waiting time to obtain a first solution of the problem; we also give a finite sample estimator of this quantity. It is thus directly proportional to the expected total amount of monetary budget required to solve the problem. We also show that the performances of the solvers strongly depend on the selected set of internal parameters. This is the first time AQCs and Memcomputing machines are compared to each other on such a comprehensive set of industrially and mathematically relevant problems, and it's one of the first times that the dependence of their performances on internal parameters is analyzed in detail.

In the following sections, results regarding scalability and parameter dependency of both unconventional solvers are presented for each one of the three problems.

[3]See Section 2.6.1.

Scalability assessment for the baseline solvers

We now discuss the scalability of the analyzed platforms while using their default parameters (baseline) in terms of Mean First Solution Time (MFST)[4] for further details). This metric allows one to precisely capture the expected waiting time to obtain the first solution, or in other words the expected required computing time in an operative scenario. This is achieved by taking into account explicitly failed runs (no solutions found in the maximum allowed wall time). The timeout for D-Wave and VMM were set based on the available computing budget for each machine. D-Wave was run for dozens of seconds for each problem instance coherently with the granted computational time. The VMM was executed for a few hours while running the GPU backend, and for several days when utilizing the CPU backend (see each problem section for precise timeout values). Gurobi was granted a timeout of 72 hours (maximum wall time of the IIT HPC infrastructure). For every problem, Gurobi was run on a cluster node with 32 cores (2 physical sockets).

To evaluate the scalability and remove possible biases, we define n instances of a problem given a prescribed problem size N. We estimate the MFST of each instance and report the average MFSTs between all the instances at the same problem size, for each tested problem size.

Semiprime factorization

We defined $n = 5$ different problem instances (five different p, q pairs). To run the benchmark on Gurobi, we randomized 5 times the seed for each problem instance, for a total of 25 runs per problem size. In Figure 4.10a, a plot with the average MFSTs of the performed runs is shown. Each point is the average MFST of the problem instances belonging to the same bit size N.

The average MFST for VMM was estimated based on the same 5 instances and 120 different seeds for each instance. This setting was required to better estimate the solution probability. We used the standard CPU backend of the Memcomputing Software As a Service platform (Saas). For each run, VMM simulated 2 replicas of the circuit corresponding to FP, with a maximum timeout of 10 hours. To solve the problem, VMM usually runs a Monte Carlo algorithm to explore the space of the circuit parameters before simulating it. For this specific problem, no parameter space exploration was performed and the number of Markov chains was set to 120 to perform the calculation of the 120 different seeds in parallel on 60 cores. Hence, all the runs correspond to the same single circuit topology and parameter set.

[4]See Section 2.6.1.

We analyzed the scaling with respect to the increasing problem size, namely the bit size in the interval [14, 64]. Gurobi was very effective in relative terms (see MFST in Figure 4.10), since it performed a quick presolve of about half of the instances under consideration. Gurobi's ability to reduce the number of variables and constraints of the optimization problem depends on the instance and thus on the semiprime to be factorized. For $N = 64$, Gurobi performed a presolve calculation effectively enough to reduce solution time by four orders of magnitude compared to the problems where this simplification was not possible. At $N = 68$, the problems that Gurobi could not simplify exceeded the wall time of 72 hours of computation. For this reason, problems with a bit size greater than 64 were not considered. In the range $N = 37$ to $N = 64$, the MFST for Gurobi scales with a slope of 16.91 ± 0.92 (see Figure 4.10b).

All instances between $N = 15$ and $N = 42$ were solved at least once by VMM. The linear fit of the MFSTs in the range between 37 and 42 bits resulted in a scaling with a slope of 14.64 ± 0.62. Therefore, VMM obtained a slightly better slope with respect to Gurobi, but the overall execution time was still superior, and VMM could not solve all instances for larger bit sizes.

We also used D-Wave devices to solve FP in the interval between 14 and 17 bits. The number of device queries (which can be considered as the number of different seeds in a quantum device) was 10^4 up to $N = 14$ and then was gradually increased up to 8×10^4 at $N = 17$ (~ 12 seconds of computational time). The number of runs was doubled each time that N increased by a unit. We found no significant differences in the slope between D-Wave Advantage and the D-Wave 2000Q, but D-Wave Advantage proved slightly faster, probably because its greater connectivity allows for shorter physical qubit chains [35].

Overall, we found that D-Wave devices could only tackle small instances of the problems, whereas VMM and Gurobi could deal with significantly higher bit sizes. Gurobi was the fastest approach overall.

Hard-assignment Gromov-Wasserstein problem

GWP's size and complexity depends on the variable N i.e. the number of points to match between the two sets[5]. To run our benchmark, we defined 5 problem instances for each problem size. For each instance, multiple runs were performed with different random seeds for the solvers. Results are reported in Figure 4.10b. For Gurobi, each instance was solved 5 times with different random seeds, for a total of 25 runs for each problem size. For VMM, each instance was solved 5 times with different random seeds for the solver, for a total of 25 runs for each problem size, as for Gurobi. Contrary to FP and CHRP,

[5]See Section A.

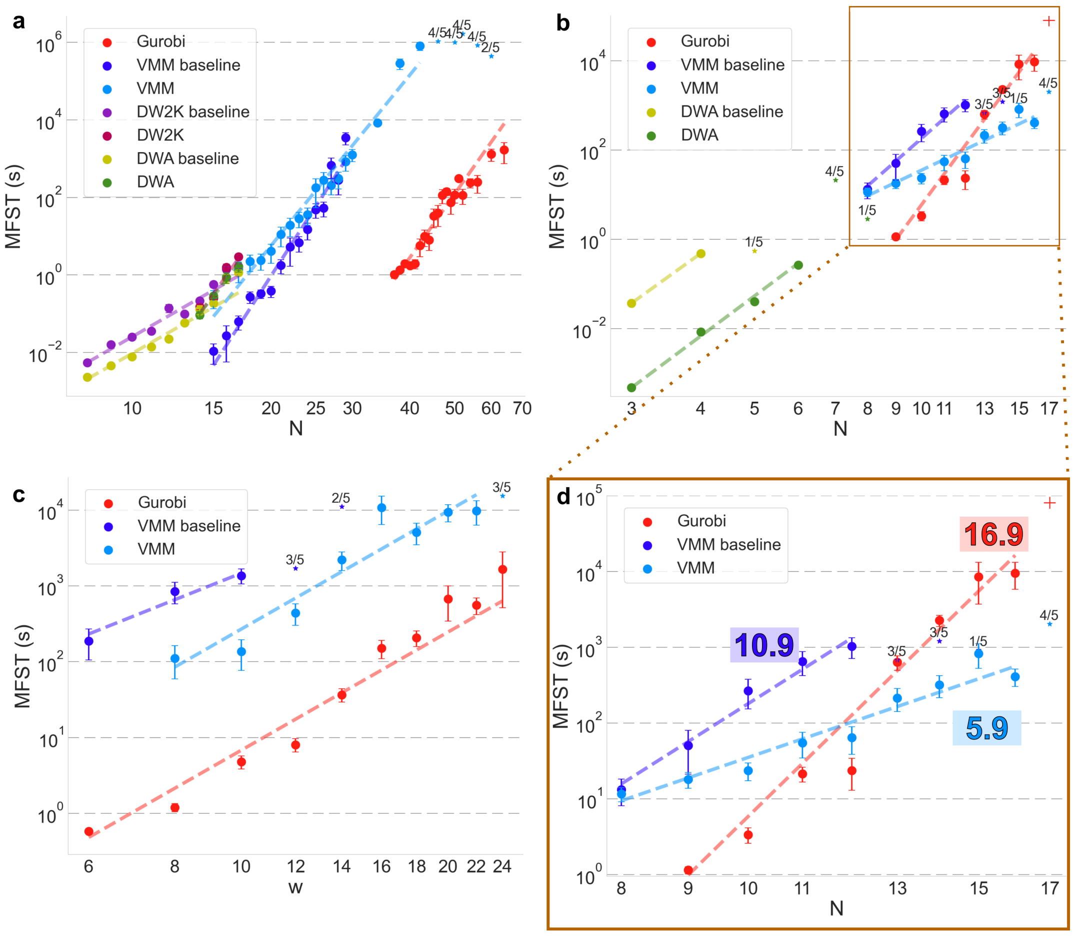

FIGURE 4.10: MFST plots for all the tested computing platforms, in log-log scale. Every problem size corresponds to 5 different problem instances using different seeds. Whenever a problem size on a given machine includes unsolved instances, a smaller dot is used and the number of solved instances is shown. The error bars represent the standard deviation (see Methods). We report both baseline and parameter-optimized scaling results. In the scaling section, we discuss these results. **a**: FP MFST with respect to the number of bits of the semiprime. **b**: GWP MFST with respect to the number of points. The red plus mark is the point $N = 17$ for Gurobi, which was obtained by solving each instance only once, due to time constraints. **c**: CHRP MFST with respect to the number of workers. **d**: Zoom of the GWP plot showing the slopes for Gurobi and VMM solvers for the biggest problems. The VMM with enhanced settings achieved the best performances. The highlighted rounded slope values have the following values and standard deviations: Gurobi 16.89 ± 0.83; VMM 5.89 ± 0.50; VMM baseline 10.93 ± 0.98.

we found an advantage in using the GPU backend of the Memcomputing Saas solver. We thus used GPUs to solve GWP on VMM, setting the timeout for each instance to $T_{max} = 1900$ seconds. For D-Wave Advantage, every instance was sampled with thousands of annealing cycles, going from $50,000$ samples for $N = 3, 4, 5$ up to $390,000$ samples when $N = 7, 8$. We report that $390,000$ samples resulted in ~ 60 seconds of access time for D-Wave Advantage.

Gurobi solved all the runs for each instance of each problem size up to $N = 16$ points. At $N = 17$, the most computationally intensive instances required a wall time exceeding 72 hours, so we solved every instance only once (a single seed). The corresponding point is indicated with a $'+'$ marker on the plot. The slope for Gurobi was 16.89 ± 0.83.

Figure 4.10b shows that the VMM achieves a better overall scaling than Gurobi. The slope of the fit in log-log scale is 5.89 ± 0.50, which is significantly better than the competitors. At $N = 12$ and $N = 13$, Gurobi and VMM require a similar time to solve the problem, but the wall times quickly diverge for bigger problem sizes. At $N = 16$, the VMM requires an average MFST of $409 \pm 107s$ (~ 7 minutes) compared to the $9560 \pm 3730s$ required by Gurobi, i.e. VMM is ~ 23 times faster than Gurobi. VMM was able to tackle problems almost up to the same size as Gurobi, but did not solve every instance of the hardest problem size, $N = 17$.

D-Wave could only tackle small instances ($N \leq 8$), thus it is hard to make a sound comparison with other technologies. At $N = 8$, D-Wave Advantage solved only one of the five instances, finding the correct solution for this instance only once over $390,000$ trials. The resulting point (the rightmost in the D-Wave plot) is thus of limited statistical significance. Deeper insights will be achieved when the D-Wave hardware is advanced enough to tackle bigger instances of this problem.

Overall, VMM was slightly less reliable than Gurobi in the biggest problem sizes, but showed the best scaling behaviour.

Capacitated Helicopter Routing Problem

We fixed the number of rigs to 25 and varied the number of workers (w) and the pick-up and drop-off locations generated at random (see Appendix B). When converted into QUBO and embedded into chimera topology, the problem becomes too large to fit in D-Wave, even for a few passengers, thus the device was not included in the comparison plot.

Summary results are presented in Figure 4.10c. We considered five instances for each problem size, and every instance was submitted to Gurobi and VMM with 5 different seeds, for a total of 25 submitted problems for every worker size.

We used the GPU backend of the Memcomputing Saas, setting the timeout for each instance to 5 hours. The slopes of the two solvers in log-log scale were very similar for the problem sizes considered (5.2 for Gurobi versus 5.1 for VMM). In contrast to GWP, the CHRP did not show an advantage for VMM in scaling terms. Additionally, while Gurobi solved all instances, VMM was slightly less reliable beacause it could not solve all instances of the biggest size ($w = 24$).

Overall, the results of the three benchmark problems show that D-Wave's current hardware can solve only very small instances. VMM managed the biggest problems well in most cases. VMM's scaling was similar to Gurobi, but was superior for one problem.

We note that, for GWP, VMM used GPU hardware to achieve significant speed-ups. In contrast, Gurobi, and the branch-and-bound [478] method in general, is not particularly amenable to a GPU implementation. This is important because the ability to exploit GPU architectures may be critical to improving the overall computing time.

Parameter tuning procedure

Here, we provide details of the parameter tuning procedures for each platform.

Parameter tuning procedure for VMM – The VMM User Interface (UI) provides access to two main classes of tunable parameters. The first class sets the physical features of the circuit. The second class sets the simulation settings. The parameters that set the physical features, and hence the internal dynamics of the VMM, are 19. They represent electronic element characteristics like resistances, capacitances, and transistor model parameters. On the other hand, the simulation is performed according to several *simulation parameters* used to set the control unit of the VMM [479]. Examples are the total timeout for a job, the amount of virtual time to simulate, and limits for the wall time. The VMM UI includes a parallel tempering (replica exchange Markov Chain Monte Carlo) to test and optimize multiple circuit physical parameters. This is called the Dynamic Parameter Search (DPS) mode for the VMM. It runs multiple circuit realizations using different physical parameters. Each of these realizations are a Markov chain for the parallel tempering. Each Markov chain changes the physical parameters, simulates the circuit, returns a score based on the VMM performance for that parameter set, and accepts or rejects the change following the Metropolis–Hastings algorithms with adaptive temperatures. The user can set the number of Markov chains, the number of iterations, the standard deviation, and the number of parallel processes. This latter hyper-parameter allows one to run multiple Markov chains in parallel. The hardware used were virtual machines running on Google Cloud (Intel Xeon 60 virtual core processor for the CPU VMs and NVIDIA 8 V100 GPUs for the GPU VMs). The number of parallel processes were therefore always set to equal or double the virtual machine cores or number of physical GPUs, depending on the type of virtual machine used. During the tests, it was clear that the physical parameter tuning would heavily impact the VMM performance.

In our tests, six distinct rounds of DPS with 1000 iterations each were performed for the benchmark problems. Based on the problem-solving performance, the best physical parameter sets from each DPS round were selected and used to initialize the chains for the next round. In each round, we checked if the boundaries for the physical parameters needed to be enlarged or shrank based on the score distribution. The final result was a collection of sets of physical parameters. Multiple sets can be useful because the user can then run multiple realizations of VMM, increasing the convergence and so obtaining better results.

Once good samples of physical parameters were found for one problem instance, we used these parameters to run the other problems of the same type but different sizes and instances. We did this using the Dynamic Solution Search (DSS) mode. However, once we had found the physical parameters, a few extra parameters for the VMM control unit were tuned. To describe these parameters, it is useful to briefly detail how the VMM control unit works. The control unit sets up the VMM, assembling and connecting the self-organizing gates to embed the problem. It sets the virtual time, checks the limit of wall time, and initializes the circuit components (e.g. capacitors, transitors). This initialization is set at random by default, but the user can feed a custom initialization (warm start) or use the rounding of the solution of the ILP relaxation. For our tests, we used random initialization. Moreover, the control unit can also restart the circuit, using a perturbation of the best solution found in the previous simulation. Therefore the user can set up the number of iterations where a restart will be operated, how to perturb the best assignment to use as an initial condition for the restart (the "switch fraction" parameter), the virtual time of each iteration, the limit to the wall time, and the total time out. Other advanced parameters for the control unit can be set, e.g. a target for objective function and the number or replicas of the circuit. Each circuit actually comprises several coupled interconnected replicas of itself to enhance the convergence. In our tests, we used 2 replicas.

Parameter tuning procedure for DWA We tuned the following parameters for the D-Wave devices:

- t_{ann}: the annealing time, namely the time duration of a single annealing cycle.
- λ: the regularization term that accounts for the penalty on the constraints. For the factorization problem, this parameter was obtained from previous literature [275].
- $c = J_{\text{chain}}/Q_{\text{max}}$: the ratio between the intensity of the chains coupling, J_{chain}, and $Q_{\text{max}} = \max_{i,j}\{a_i, b_{i,j}\}$, where a_i and $b_{i,j}$ are the biases and the couplings, respectively, of the QUBO problem Q submitted to the D-Wave device.

Note that in the experiment described in Section 4.1.1 we optimized the chain strength by tuning a different parameter that multiplies a quadratic average of the problem weights (Eq. 4.1), which is also the default approach implemented in Ocean SDK. Here, we optimize the simple ratio between the chain strength and the maximum among weights and biases of the problem. This particular approach can be expected to be less reliable, since the considered ratio retains less information regarding the problem structure, if compared to an average of the problem weights. Nonetheless, as we will show, in this experiment we obtained great improvements in the MFST for DWA, which are not matched by the improvements seen in the experiment presented in Section 4.1.1.

The parameter tuning procedure involves running 10,000 annealing cycles for several combinations of the parameters and choosing the best one in terms of TTS (Time To Solution). For both the problems where D-Wave devices were used (semiprime factorization and hard–assignment Gromov–Wasserstein), we ran the parameter tuning on the biggest problem size.

For FP, the tuning procedure involved c and t_{ann}, at the point $N = 14$. The penalty factor λ was fixed to 2, as in [274], [275]. We tested a total of 32 different combinations for c and t_{ann}, which can be found in table (4.2).

Parameter	**Values tested**
c	0.330, 0.402, 0.490, 0.598, 0.729, 0.888, 1.083, 1.320
t_{ann}	1,2,4,8

TABLE 4.2: Set of tested values for D-Wave and semiprime factorization

This strategy was used for both the available devices D-Wave 2000Q and D-Wave Advantage. The best value found was $c = 0.598$ for D-Wave 2000Q and

$c = 0.490$ for D-Wave Advantage. Finally, the best annealing times found were $t_{ann} = 8$ for D-Wave Advantage and $t_{ann} = 1$ for D-Wave 2000Q (additional annealing times, namely $t_{ann} = 16, 32, 64$, were tested, keeping the c value fixed, but without any visible improvement).

For GWP, we used the $N = 6$ point to tune parameters. We tested a total of 64 different combinations for c and λ, whose values are reported in table (4.3). Here, we additionally tuned λ to limit the total needed computing time to stratify the parameter selection. That is, first we selected c and λ and then we optimized the annealing time. Results are shown in figure 4.11, where the

Parameter	**Values tested**
c	0.330, 0.402, 0.490, 0.598, 0.729, 0.888, 1.083, 1.320
λ	4,6.5,10,16,25.5,40,64,101

Table 4.3: Set of tested values for D-Wave and Gromov-Wasserstein

color scale encodes the percentage probability that the solver found the global optimum. Figure 4.12 correspondingly shows the percentage of samples in which the constraints are satisfied (i.e. the solution corresponds to a correct permutation matrix). We observe that a high probability of satisfying the constraints does not always correspond to a high probability of finding the global optimum, although the two quantities are partially correlated. Indeed, according to figure 4.12, one should choose $\lambda = 25.5$ or $\lambda = 40$, while in figure 4.11 $\lambda = 16$ seems a much more robust choice. Since the aim is to fully solve the problem, we chose $\lambda = 16$. The value of c was easier to pick since $c = 0.598$ produces consistently better results in both figures 4.11 and 4.12. Keeping $\lambda = 16$ and $c = 0.598$ fixed, we performed $10,000$ annealing cycles for five different problems with $N = 6$, using seven different annealing times. Figure 4.13 represents the TTS versus t_{ann}. Increasing t_{ann} steadily reduces the TTS, in contrast to the semiprime factorization case. This dependence of TTS vs t_{ann} is in accordance with the adiabatic theorem, which predicts that increasing the annealing time will produce a better solution [116], [117]. Nonetheless, lower annealing times are useful for decoupling the quantum state from the outer environment. Indeed, for flux qubits (the technology used in D-Wave), decoherence times are usually measured in dozens [211] or hundreds of nanoseconds [198]. Such times are much lower than the fastest annealing time available on state-of-the-art Adiabatic Quantum Computers (AQC) (0.5 μs on D-Wave Advantage). Nonetheless,

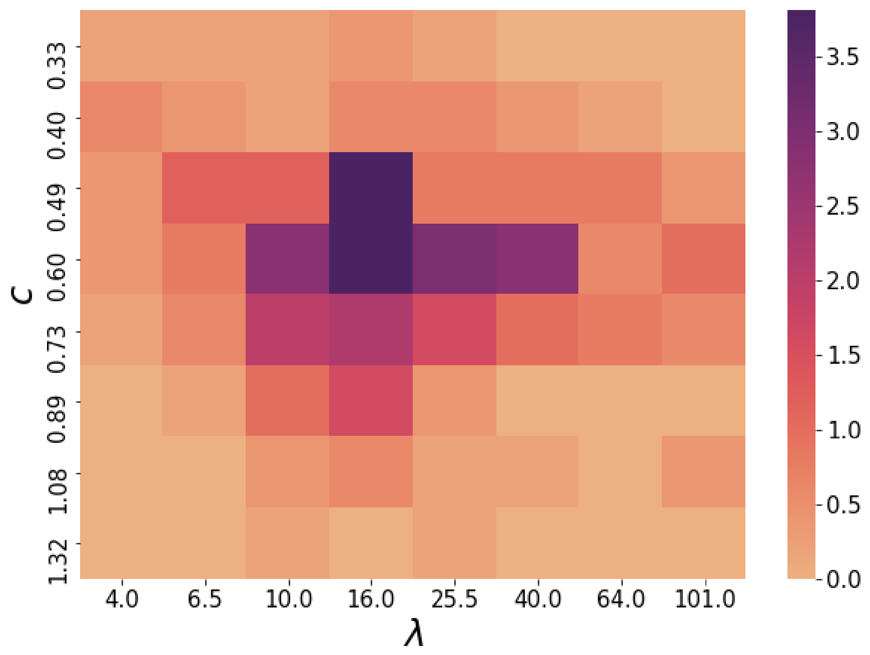

FIGURE 4.11: Test performed on the D-Wave machine for $N = 6$ for GWP. The color of each box encodes the percentage probability that the global optimum is found.

some authors have reported that AQCs appear to be surprisingly resilient to noise and imperfections and that they can show evidence of quantum behavior for annealing times in the order of microseconds [91], [106], [213]. Here, the annealing process seemed to benefit from longer annealing times, which could mean that the noise had a limited impact, or that the fluctuations induced by the outer noise boosted the system's ability to reach the global minimum [473]. On the other hand, in the semiprime factorization case, longer annealing times did not improve performances.

Parameters dependency

D-Wave and VMM allow users to tune several parameters that directly affect the dynamic of the physical and simulated device, respectively. As discussed in Chapter 3, tuning such parameters can result in a sharp performance increase. For each machine, we identified a single problem size on which to execute a tuning protocol. Those optimal parameters were then used for all the other problem sizes, improving performances. Below, we systematically compare the scaling results using the default (baseline) parameters against their tuned version.

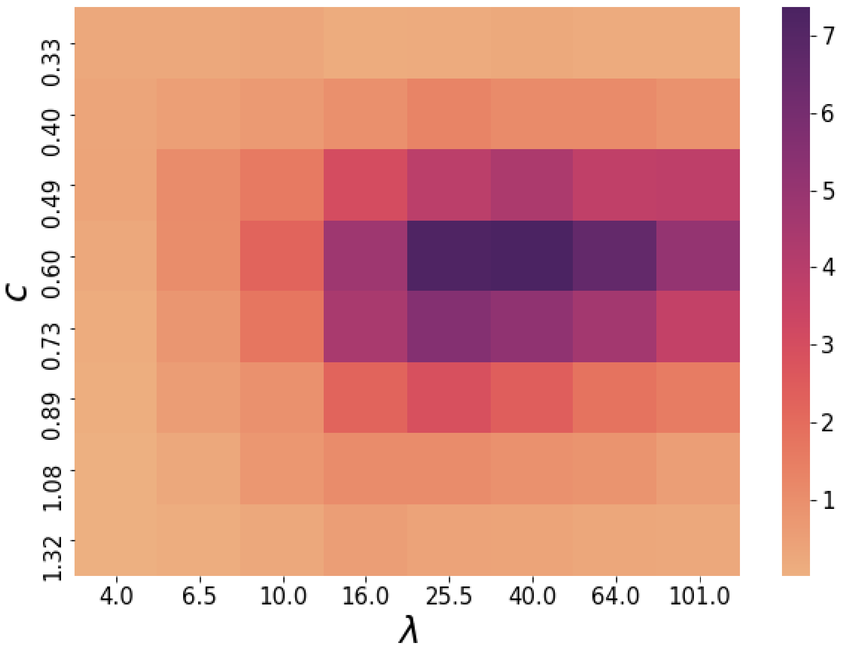

FIGURE 4.12: Test performed on the D-Wave machine for $N = 6$ for GWP. The color of each box encodes the percentage probability that the matrix used in GWP is a valid permutation matrix.

Semiprime factorization – For VMM, we considered $N = 29$ as the tuning point. In Figure 4.10a, a comparison between the results before and after the parameter tuning is shown. Instances $N \in [30, 42]$ were not solved using default parameters. After the parameter tuning, all the instances up to $N = 42$ were solved in less than $\sim 4 \times 10^3$s. For D-Wave, the size $N = 14$ was used as the tuning point. The tuned parameters didn't result in an improved solution probability.

Hard-assignment Gromov-Wasserstein problem – In the GWP, VMM without any parameter tuning solved only one of the five instances for $N = 15$, with no instances solved for $N = 16$ and $N = 17$ (see Figure 4.10b). The parameter tuning was performed on one instance at $N = 16$, enabling VMM to solve all other instances for $N = 16$ and all but one instance for $N = 17$. The tuning process also sensibly reduced the computing time for all $N < 16$ cases. This means that for GWP the parameter tuning of VMM shows good transferability: spending computational time to get the right parameters is an effort that systematically boosts the solver's ability to tackle new GWP instances, even at different problem sizes. The most remarkable effect was at $N = 12$, where the MFST of VMM was reduced by a factor of 16. The effect for

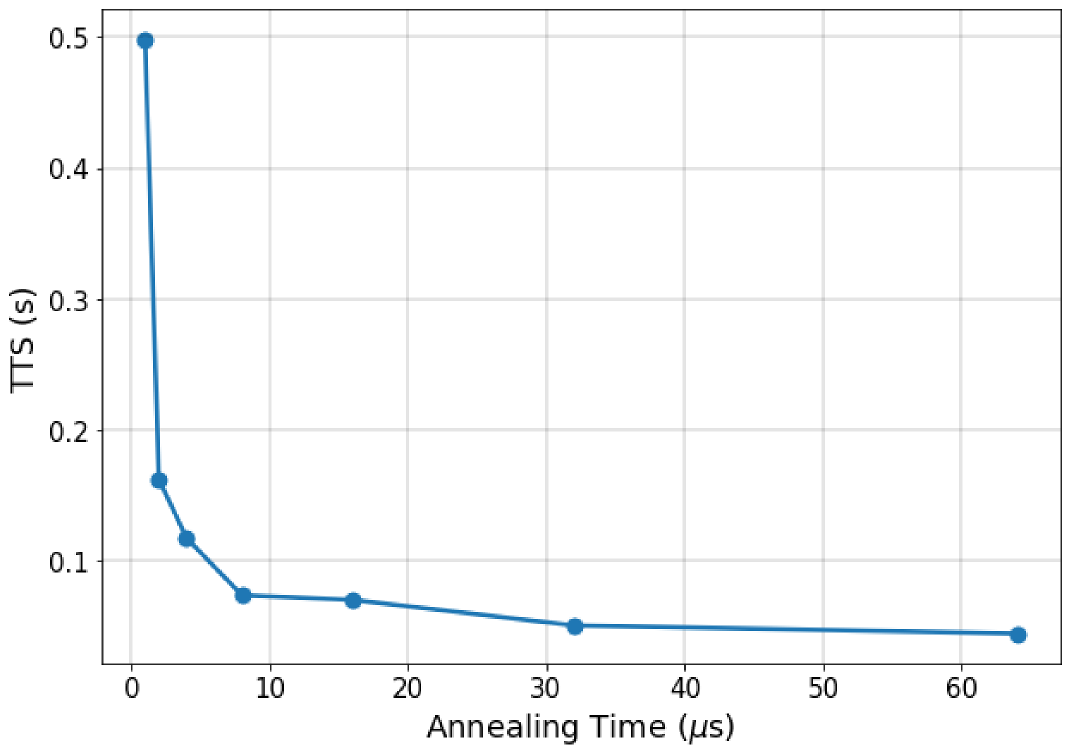

FIGURE 4.13: Test performed on the D-Wave machine for $N = 6$, $c = 0.598$, $\lambda = 16$ for GWP. For each annealing time (x-axis), we considered 5 different problems of that size and we performed 10.000 annealing cycles for each. The values on the y-axis represent the average TTS.

D-Wave Advantage was even more remarkable: the tuning procedure at $N = 6$ reduced the MFST by more than one order of magnitude at all problem sizes. The highest speed-up (~ 78 times faster) was found at $N = 3$.

Capacitated Helicopter Routing Problem – Parameter tuning allowed VMM to reduce all its MFSTs, peaking to a tenfold reduction for $p = 10$ (see Figure 4.10c). While the baseline VMM solved all instances up to $p = 10$, VMM with tuned parameters solved all instances up to $p = 22$. The tuned VMM could tackle problems with ~ 3.5 times the number of nonzero elements in the ILP problem matrix, compared to the baseline. The efficacy of the solver was therefore greatly increased, showing good transferability of the optimal parameters.

Overall, we conclude that the tested parameter tuning procedures are fundamental for VMM and D-Wave solvers. For some instances of GWP and CHRP, using optimal parameters reduced by more than one order of magnitude the required time to find the global minimum. The ability to obtain an advantage using new computational approaches such as VMM and D-Wave heavily depends on the development of better and automated parameter tuning procedures.

4.2 AQCs as samplers

In this Section we present two research works where AQCs are exploited for their ability to efficiently sample from a Boltzmann distribution defined over binary variables[6]. In particular, both works focus on using the samples obtained via Quantum Annealing to train a Boltzmann Machine[7] (BM). The first work introduces a reverse annealing approach to train a Restricted BM, and has resulted in two publications, one as a full article [4] and one as a proceeding [3]. The second work implements a fully-connected, AQC-trained BM, and shows that a limited quantum advantage can already be achieved even against a classical rival implemented on GPU. The relative article is currently under review [24], and a second contribution is going to be published as a proceeding of the IEEE QCE2023 conference [23].

4.2.1 Comparing forward and reverse annealing in RBM training

This Section presents the results and design of the experiment we published in [4] and [3]. We studied in detail how a RBM is implemented using AQCs, comparing performance obtained using the Forward Quantum Annealing (FQA) schedule, which is the usual annealing schedule, to those obtained using our novel approach based on Reverse Quantum Annealing (RQA). We also analyzed the differences between the distributions produced by FQA, RQA, and by the classical approach, detailing the shape of the final probability distributions produced by each method.

Introduction

To assess the performance of the training process we trained a RBM with 16 visible units and 16 hidden units to reconstruct the Bars and Stripes dataset. We applied embedding techniques to prepare a graph of virtual qubits arranged to implement all the weights present in a classical RBM. The performance is evaluated in terms of the percentage of reconstructed pixels and the average Log–Likelihood of the dataset. All the quantum computations have been performed on a D–Wave 2000Q System processor.

The choice of 16 visible units is somewhat limited if compared to recent implementations that made use of up to 64 visible units [325], [480]. Nonetheless the chosen structure allows to test embedding techniques with chains of

[6]See Section 2.3.
[7]See Section 2.4.

moderate length (four bond qubits), and it allows direct comparison with the previous relevant work from Benedetti et al. [221], where a network characterized by sparse connections between the visible and the hidden layers was implemented.

The results we are presenting in this Section show that embedded topologies do not sensibly interfere with the capacity of the AQC to produce correctly distributed samples. The use of embedding techniques thus allows faster learning if compared to the sparse case.

According to our results, the samples produced by the AQC while using a reverse annealing schedule are closer to the configurations belonging to the training dataset than that produced by the more conventional forward annealing. As a consequence, the learning process exhibits a novel semantic behavior that brings pros and cons. In our case, the RBM trained with the use of reverse annealing achieves an overall reconstruction score that is comparable to the usual quantum algorithm for AQCs, which means we don't detect a statistically significant improvement in performances when using the reverse schedule. Nonetheless, the reverse-annealing-trained RBM reproduces elements from the training set with a probability that is double that of the forward annealing approach, and also higher than that of the classical method.

Dataset and parameters of the learning process

The training involves a dataset of images composed of $M \times M$ pixels, known as Bars and Stripes dataset (BAS), defined in Ref.[481]. The images consist of either black or white stripes or columns. Figure 4.18 **c** shows some sample images taken from the 4×4 version of such dataset. Each training image can be mapped to a M^2–dimensional binary vector $\mathbf{r}$, where each term in the vector corresponds to a visible unit of the RBM.

Figure 4.14 **b** shows the structure of the chosen RBM.

We trained five times the RBM for 1100 epochs using quantum forward annealing. Each instance was initialized with random weights and biases extracted from the same probability distribution. The weight initialization spread σ is the value of the standard deviation of the Gaussian used for randomly generate the weights and the biases. The weight initialization range I_W defines the symmetric range centered in zero truncating the Gaussian from which the weights and the biases are sampled. The amplitude of the interval I_W is bound below by the signal to noise ratio, due to both the uncertainty in the values of the currents and the random fluctuations of the chip. It is also bound above by the maximum values of the current. Among the amplitudes tested, $I_W = [-3, 3]$

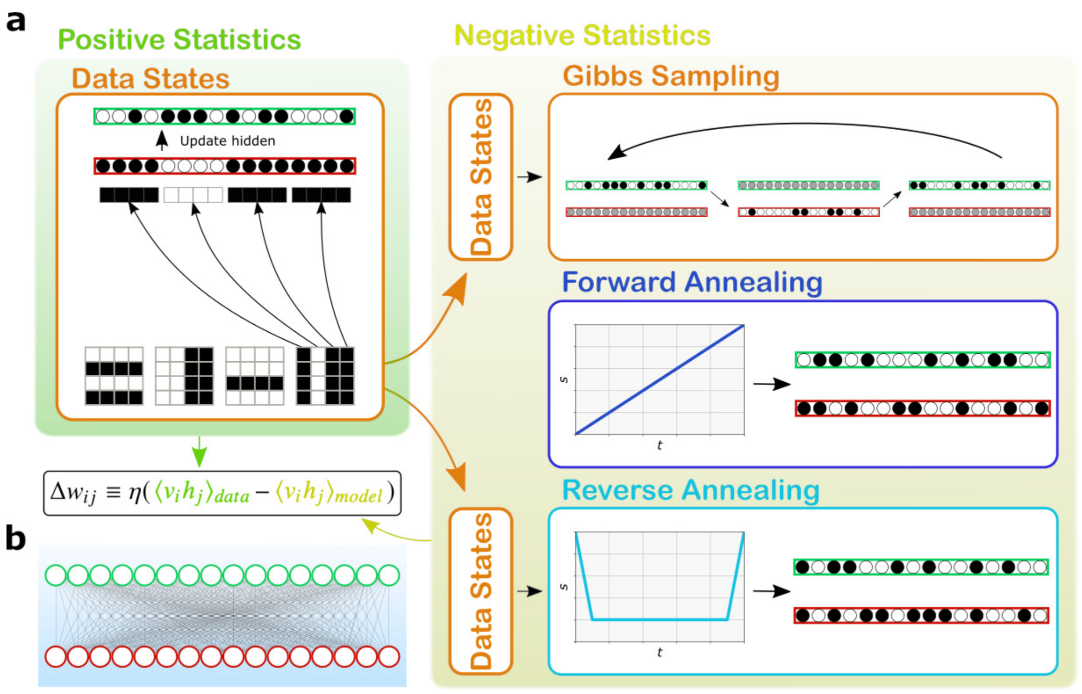

FIGURE 4.14: **a**: Schematic overview of the methods to train a Restricted Boltzmann Machine. The green and yellow boxes show which part of the gradient is estimated by each step of the algorithm. Steps that require only classical computations are circled in orange. During the data states phase, the elements of the dataset are loaded in the visible units (red), and the hidden units (green) are updated accordingly. The result allows the estimation of the positive statistics. Next, one step among Gibbs, forward annealing, and reverse annealing sampling is chosen to estimate the negative statistics. Forward annealing is the only one that does not depend on states loaded from the dataset. **b**: Structure of the Restricted Boltzmann Machine used in this work, with 16 visible units fully connected to 16 hidden units. See Section 2.4.1 to learn more regarding RBM training.

was sufficiently large to overcome weights and sufficiently small to not reach the maximum values allowed for the currents. Figure 4.19 presents the learning curve for a RBM initialized with smaller weights. The result shows that, if the aforementioned boundaries are respected, the initial distribution of the weights could have a reduced impact on the learning speed.

For each run we set $\alpha = 0.32$. Figure 4.18 **b** shows the results obtained by averaging over the five runs, in terms of the average Log–Likelihood of the dataset L_{av}, defined as in Section 2.4.1. The most expensive step in computing L_{av} is obtaining the explicit expression of the partition function $Z(T_{\mathrm{eff}})$. The number of units in the RBM (32 total) is sufficiently low so that we could calculate L_{av} without resorting to approximations. We therefore used a classic algorithm to scan through each possible configuration of the units, performing the summation required in the exact form of L_{av}.

We also trained five times the RBM for 1100 epochs using the classical algorithm. The instances were initialized with the weights already used in the forward annealing case. It means that for each quantum instance, there is a classical instance that used the same weight initialization.

Weights and biases of the RBM were initialized by extracting random values from a Gaussian distribution with $\mu = 0$, $\sigma = 2$, truncated in $[-3, 3]$. The learning rate was set to $\eta = 0.15$. In the forward annealing case, an annealing time of 2μs was chosen. For the reverse annealing case, we set a reverse step of 1μs, followed by 18μs of pause and finally a last 1μs of forward annealing. During the pause $s(t) = 0.2$.

For the classical case, training was performed with $n_g = 200$. For the quantum case, we performed 150 annealing cycles in each training epoch, both for the forward and reverse annealing. Considering that each annealing cycle outputs 8 configurations, it sums up to 1200 configurations.

The chain strength was set to $J_{\mathrm{chain}} = 1$, the maximum value allowed by the device. Figure 4.15 shows a comparison between two values for J_{chain} we tested. Varying the chain strength slightly affects the sampling temperature, but it does not seem to dramatically impact the learning curve.

Experimental results for the forward annealing training

As one might expect, thanks to the full connectivity based on the embedding on virtual qubits, our forward annealing approach obtains Log–Likelihood scores sensibly better than sparse implementations [150], [221]. Contrary from Ref. [221], where a 16×16 RBM is implemented with 80 connections between visible and hidden units, our implementation realizes all 256 connections. Figure 4.16

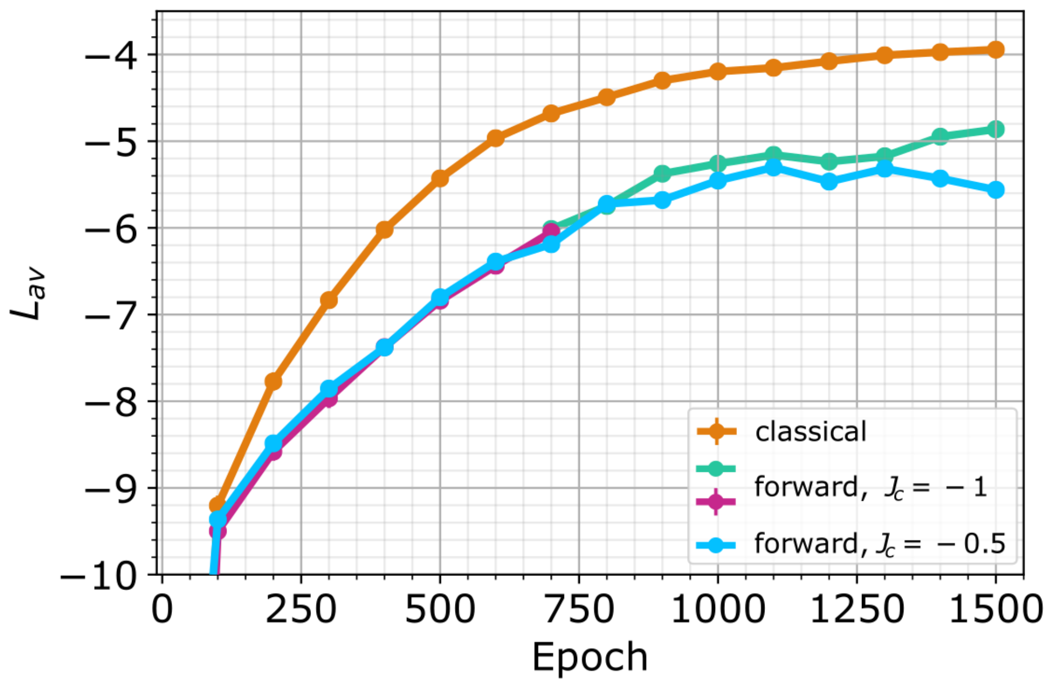

FIGURE 4.15: Comparison between two forward annealing learning processes at different coupling J_C. The RBM considered is the complete one. Here $I_W = [-2, 2]$. The two-colored line corresponds to $J_C = -1$ and $\alpha = 0.33$. Such value for α has been chosen among five tested so to maximize L_{av} at epoch epoch 700. Five runs have been trained and averaged (violet) up to epoch 700, then the second-best run (green) has been trained up to epoch 1500. The light blue line is obtained with $J_C = -0.5$. The different choice for J_C slightly affects the effective temperature of the distribution, so the best value for the rescaling parameter was found to be $\alpha = 0.30$. The learning of a classical RBM (orange) is reported as reference.

shows the embedding we used to implement all connections, which required four physical qubits for each unit. At epoch 1000, our forward annealing method obtains a score of -5.00 ± 0.08 which is better than the best forward-annealing-based result [221], optimized step-by-step by a temperature estimation tool, which reaches a score of -5.3 at iteration 5000. Even without using any T_{eff} estimation technique, the higher number of connections was sufficient to gain a great advantage over the sparse implementation. For the sake of completeness, the same sparse embedding of Ref. [221] has been tested at a fixed temperature. The results are presented in Figure 4.17, where we can appreciate the evident difference in performances between the sparse and complete network. As pointed out in Ref. [150], the complexity of the topology of the AQC seems to be a major bottleneck for the implementation of RBMs.

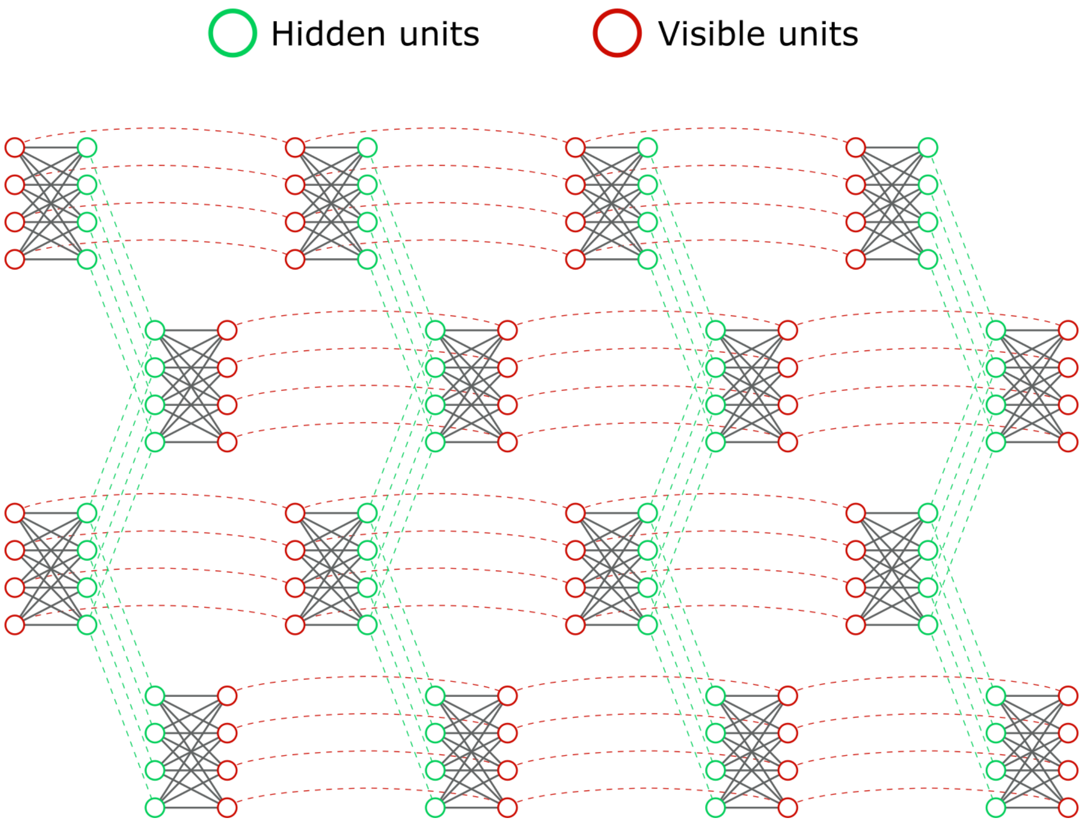

FIGURE 4.16: Embedding of a complete RBM with 16 visible units and 16 hidden units on the graph of a D-Wave 2000Q System processor. Each circle corresponds to a physical qubit. The lines represent active couplers between qubits. Each gray coupler implements a weight of the RBM. Green and red couplers are set to a strong negative value for the coupling, and they force physical qubits to stay parallel to each other. Such couplings make groups of four physical qubits act as a single, virtual qubit. Red qubits compose the visible units of the RBM, while green qubits compose the hidden units.

Motivations for the reverse annealing approach

In [3], to introduce a semantic search emulating the initialization of the Gibbs sampling of the classical case on a quantum adiabatic computer, we adopted a novel approach based on a reverse annealing schedule (introduced in Section 3.1.2). The customary quantum annealing method allows a unique system initialization, corresponding to the equally–probable quantum superposition of all the classical states. On the contrary, reverse annealing allows us to perform a quantum search in the neighborhood of the configuration set by the user as the initial state. It then constitutes a possible way to emulate the initialization of the Gibbs sampling in the classical case.

We thus modified the usual quantum algorithm for the estimation of $\langle v_i h_j \rangle_{model}$, which exploited forward annealing, to exploit a sampling procedure based on

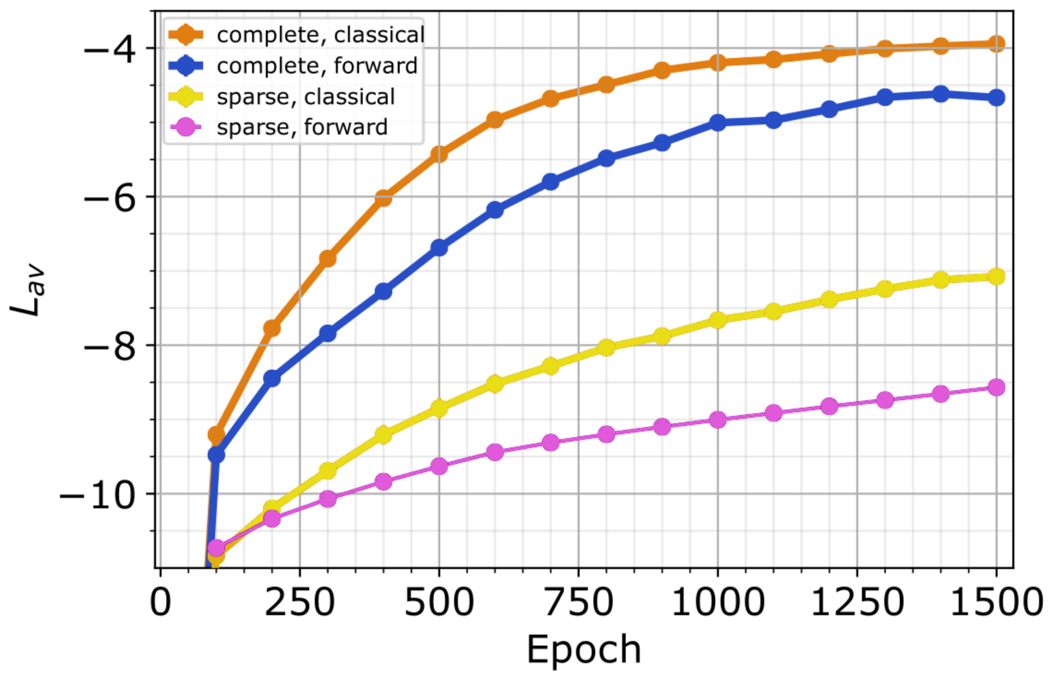

FIGURE 4.17: Comparison between sparse (violet) and complete (blue) topology of the RBM for a forward annealing-based training. For the sparse case, we adopted the same embedding of Ref. [221]. Initial weights and biases are extracted from a normal distribution with $\sigma = 2$, $I_W = [-3, 3]$. In the sparse case, the best L_{av} was found for $\alpha = 0.13$. For the complete case we instead selected $\alpha = 0.32$ (the embedding is reported in Figure 4.16). The classical training of the complete (orange) and of the sparse (light orange) models are shown for reference.

reverse annealing. The complete procedure is detailed in Appendix E.

Experimental results for the reverse annealing training

We trained twice the RBM for 1000 epochs using reverse annealing. The instances were initialized with two weights set randomly chosen from the five already used in the classical and forward annealing case. We chose $\alpha = 0.32$ as in the forward annealing case.

Figure 4.18 **b** shows the results in terms of the average Log–Likelihood of the dataset, compared with the results obtained in the forward annealing case.

The value $\alpha = 0.32$ was chosen for the forward annealing case because it achieves the better Log–Likelihood score at epoch 1000, among a range of tested values. We kept the poor man's choice of $\alpha = 0.32$ also in the reverse annealing case, and we set the reverse annealing schedule without further exploration. Despite such lack of fine optimization, the reverse annealing performs even better than the forward annealing up to epoch 900, therefore proving capable to speed up the learning in the initial phase, and comparable at the epoch 1000.

We now turn our attention to the differences among the energy distributions produced by the two approaches during learning. Both the forward and reverse annealing approaches initially produce a distribution corresponding to $T < 1$ (Figure 4.20 **b** and 4.20 **c**). We know this because the quantum–sampled distributions are visibly cooler than the theoretical distribution calculated at $T = 1$. Such behavior is a consequence of the choice to keep α fixed. The value of α has been chosen to maximize the score at iteration 1000 for the forward annealing case, so the same value is not necessarily the best during the first iterations.

At epoch 1000, the lowest energy configurations for the forward case are at energy $\sim (-38)$, while in the reverse case the lowest achieved energy at the same epoch is $\sim (-50)$, where the values are dimensionless, as obtained by the definition of $E(s)$ (Equation 2.26). Thus, despite the two cases share the same learning rate η, the training based on reverse annealing modifies the RBM weights so that some configurations correspond to considerably lower energies. We will now explore the consequences of such behavior.

Let's consider only those configurations whose visible units match exactly one among the elements in the dataset. We define Δ the set composed of such configurations. Thus, a well-trained RBM will have a high probability of sampling elements from Δ. Figure 4.20 **a** shows the probability that, if we sample at T=1 from the Boltzmann distribution defined by the weights of the RBM, we get a configuration in Δ. We can appreciate how such probability evolves as a function of the training epochs, for the forward, reverse, and classical cases. The Figure also shows the evolution of the probability associated with each element of the dataset, dividing the plot in 30 distinct bands.

In the classical learning case, the probabilities appear more homogeneous than in the quantum cases. The training with forward–annealing runs at a slower pace but all the probabilities are growing steadily. Regarding the training with reverse–annealing, it is manifest that it resulted in an increased overall probability for configurations belonging to the set Δ. In particular, the probability to extract a configuration in Δ from the RBM trained with reverse–annealing is approximately double with respect to the probability in the forward-trained case. Nonetheless, the average log-likelihood is similar in the two cases. The reason resides in the low probability associated with some configurations by the RBM trained with reverse-annealing. The black line in Figure 4.20 **a** represents the summation of the probabilities of the 15 less probable images. Since the

4×4 BAS dataset is composed of 30 images, the black line divides the dataset into two equal parts, which are not guaranteed to be equally represented. The RBM trained with reverse–annealing is the best of the three cases if we look at the total probability of the dataset, at the cost of a lower probability of the lesser represented images. Such behavior could be connected to the fact that reverse annealing has a higher probability to produce configurations that are similar to those appearing in the dataset.

The different sampling distribution brings two competing consequences. At the beginning of the learning, the gradient estimation is influenced more by the configurations belonging to the dataset, which results in lowering their energy faster. This effect is exactly what the reverse annealing was intended for (Figure 4.20 **a**). Later during learning, some configurations belonging to Δ are driven at low energies by the weights update and their probability is heavily underestimated. The training algorithm does not capture the relevance that such configurations have in the partition function, and thus allows them to lower energy at each step.

The sampling described above may be affected by the extraction being too bounded to the dataset elements (which could be overcome by optimizing the annealing schedule) or to a high sampling temperature, which in turn could be overcome by optimizing α.

As a last consideration, we suggest evaluating an alternative and complementary figure of merit together with the average Log–Likelihood to score a model. A potential issue connected to Log–Likelihood manifests if some images have low probability. Indeed, they carry a lower Log–Likelihood score, but it could not be informative about the capability to reconstruct data. Indeed, during reconstruction part of the visible units are clamped, thus many of the other configurations will have zero probability to be produced by the RBM. A better scoring method should only consider the ability of the final RBM to reconstruct each element of the training set. In our case, the RBMs have been trained on the whole dataset, so the training set and the test set coincide.

Figure 4.18 **a** presents the reconstruction score as the probability for a RBM to reconstruct one of the four central pixels of a BAS image, averaged over the four pixels and each image in the set. The reconstruction is performed by keeping the outer pixels set to the correct values. The number of steps for Gibbs sampling is $n_g = 500$. Note that the choice for the clamped units reduces dramatically the probability for configurations corresponding to different images in the dataset to take part in the reconstruction process since for each choice of the outer pixels there is a single dataset image compatible. Each colored band represents the probability that the corresponding image of the dataset is sampled from the Boltzmann distribution associated with the RBM. The reconstruction score confirms the quality of the learning already evaluated by the average Log–Likelihood. As expected, the low–probability images in the training with reverse–annealing have a reduced impact on the score. At epoch 1000, the reconstruction score of the reverse and forward method are statistically close, but reverse annealing exhibits better performances in previous epochs (Figure 4.18 **a**).

In general, the reverse annealing schedule introduces a meaningful search method during the learning process of a RBM on a quantum computer. Despite its slower learning rate if compared to a classical machine in terms of epochs, one should remember that the total computational time is accounted for by the product of the number of epochs with the computational time per epoch. Therefore, the advantage of the adiabatic quantum computer to manage RMBs and more generally BMs resides on its ability to be employed once the conventional hardware fails as soon as the number of qubits of the quantum processor can handle the size of the problem.

Summary of the obtained results

Boltzmann Machines can be trained with an algorithm that exploits AQCs with the spirit of achieving a computational advantage over the classical method. We showed that the use of embedding techniques does preserve the quality of produced configurations, and thus it results in significantly better scores thanks to the increased connectivity. As opposed to the usual algorithm based on forward annealing schedule, we implemented a semantic quantum search based on reverse annealing schedule. We showed that such an algorithm quickly raises the sampling probability of a subset of the configurations set corresponding to elements of the dataset and can achieve good reconstruction scores in slightly less training epochs than forward annealing. Our results suggest that reverse annealing captures the benefit of starting the algorithm by exploiting the full information provided by the elements of the dataset. It leads to a sampling probability of elements of the dataset which is double that of the forward annealing and higher than that of the classical method.

Our results, combined with hyperparameter optimization of the annealing schedule and temperature estimation techniques, pave the way towards the exploitation of both restricted and unrestricted Boltzmann machines as soon as new generations of hardware with increased connectivity will be available.

4.2.2 Fast training of a fully-connected BM

In [23] and [24], we described for the first time the procedure for training a general BM model (without any simplifications) using an AQC. We consider a fully connected network comprising both visible and hidden nodes, extending previous works that focused on quantum-trained fully connected BMs with only visible units [313]–[315] and recent works that focused on using AQCs to train Restricted BMs [4], [482], [483].

The primary advantage of the AQC-based approach over the classical method is that it eliminates the iterative nature of the Metropolis algorithm, thereby eliminating the need for a thermalization phase[8]. Exploiting this theoretical advantage, we have been able to empirically demonstrate that our training algorithm based on the complete BM is 8.6 times faster than the maximally parallelized classical algorithm. Here, by *maximally parallelized* we mean that the problem can not be parallelized further, and all parallel processes have been executed simultaneously using GPUs. We also provide a detailed analysis of the sampling times for both the quantum and classical approaches.

Experimental procedure

We conducted our experiments using D-Wave's Advantage 4.1 AQC, which is composed by 5627 working qubits connected by 40279 couplers. This particular AQC implements the Pegasus topology [159], which prescribes 15 connections per qubit (at most). We thus limited ourselves to study the training algorithms on a 16×16 Boltzmann machine, because larger graphs force the employment of long chains that can quickly become unstable and degrade the learning process. Additionally, this allow us to compare the obtained results to our previous work [4] that has been presented in the previous Section 4.2.1. Furthermore, using a relatively small BM model allows to investigate parallel quantum annealing for improved performances[9]. This last observation is of paramount importance, since the parallel approach single-handedly allowed us to outperform the maximally-parallelized approach on GPU by almost an order of magnitude, with respect to the wall time.

The dataset used in this work is the 4x4 Bars and stripes (BAS) dataset defined in Ref.[481]. Each element is a picture composed by 4 rows and 4 columns, for a total of 16 pixels. Pictures can either be composed by black and white rows, or by black and white columns. The totally black and totally white pictures have not be considered part of the dataset, so the dataset contains 28

[8]See Section 2.4.2 to learn how AQCs are used to train BMs.

[9]See Section 3.1.1.

elements. Figure 4.18 in the previous Section shows some sample elements taken from such dataset. The dataset has been chosen as it is simple yet non-trivial for this type of task [484].

During the estimation of the negative statistics, every unit in the BM contributes to the calculation of the expectation value. As a result, when mapping the BM onto the AQC, a fully connected graph is required, with a number of logical units corresponding to the total number of nodes in the BM. In contrast, the computation of positive statistics necessitates the clamping of the visible nodes, which involves fixing specific logical units within the AQC to match the clamped data. Such task can be challenging to accomplish. However, it is possible to mimic the clamping by mapping a problem on the AQC that only comprises hidden units. In fact, the mapping is faithful only after applying the following shift to the biases b_i of the hidden units:

$$b_i^{t'} = \sum_{j \in V} w_{ij} r_j^t + b_i \ , \tag{4.3}$$

where the summation is performed over the set V of the visible nodes, and r_j^t are the binary values composing the given array $\mathbf{r}^t$ belonging to the training set. In Eq.4.3 the bias of the i^{th} hidden node is modified by its interactions with the clamped visible nodes, and the clamped nodes are thus eliminated by the energy functional. The result is a graph comprising only hidden nodes. Different configurations of the visible nodes give rise to different graphs. It is thus necessary to submit a different QUBO problem for each element in the training set.

Multiple instances of the BM model can be concurrently embedded on the Quantum Processing Unit (QPU), offering a means to mitigate the computational cost associated with the training process. The computation of the negative statistics can be readily parallelized by embedding multiple copies of the same graph on the QPU. In our study, we successfully embedded 26 copies of a a 16x16 BM at the same time on the QPU (See Figure 4.21). Consequently, each QPU call generated 26 distinct samples, effectively reducing the sampling time by the same factor.

However, the computation of positive statistics necessitates a more careful approach. Due to the presence of multiple smaller graphs, each corresponding to a different element of the training set (as shown in Eq. 4.3), special attention is required. In our specific scenario, we managed to simultaneously embed four copies of each of the 28 graphs derived from Eq. 4.3 on the QPU (See Figure 4.21).

Esperimental results

To assess the effectiveness of the training, we employed the loglikelihood test and the root mean square of the reconstruction error as estimators. To compute this latter quantity, we challenged the trained BM to infer the value of the four central pixel of an image taken from the dataset, based on the value of the remaining pixels. The four central pixels have been blurred with random noise , while the outer pixels are unmodified. The random noise is created by extracting from a uniform distribution in the interval [0,0.5). At fixed intervals the Boltzmann machine is consulted to reconstruct the blurred pixels. If we call s_i the pixel proposed by the BM and d_i the correct pixel missing, the RMS reconstruction error σ_{RMS} is obtained as

$$\sigma_{RMS} = \frac{1}{R}\sqrt{\sum_{j}^{R}\sum_{i}^{N}(d_i - s_i)_j^2} \ . \tag{4.4}$$

$R = 10$ different reconstructions are performed each time the BM is called.

For the negative statistics, we set the parameter α to 0.56, while for the positive statistics we used a value of 0.4. These optimal values of α were determined by selecting those resulted in the closest overlap between the sampled distribution and the exact Boltzmann distribution at unit temperature.

The chain strength has been set to the maximum value allowed by the device [21] and the annealing time has been set to 2μs. The number of Metropolis iterations for the classical algorithm has been set to 2800, as lower numbers led to failed training, probably due to incomplete thermalization. The classical algorithm was executed on a NVidia A5000 GPU. The quantum training involved multiple calls to the AQC in order to extract the required number of samples, which was set to 2000 for both the positive and negative statistics after preliminary testing. The positive part of the computation has been embedded 112 times on the QPU, requiring 18 calls to complete the sampling process (for a total of 2016 samples). The negative part has been embedded 26 times, requiring 77 QPU calls (for a total of 2002 samples). In total, the training required the QPU to perform 95 quantum annealing processes to produce 4018 samples. Details on the multiple embeddings utilized for the 16×16 case and achievable at different sizes are reported in Table 4.4. See also Figure 4.21 for a visual representation of the embedding of the problems corresponding to the positive and negative statistics.

It is noteworthy that the samples obtained from the quantum computer are binary vectors, whereas the classical samples consist of continuous numbers

Total nodes	Positive multiembedding	Negative multiembedding
18	108	83
32	112	26
50	43	10
72	21	5

TABLE 4.4: Number of times the model has been embedded on the QPU. The *total nodes* number comprises both visible and hidden nodes, supposing the number of visible nodes equals the hidden nodes. The two columns *positive multiembedding* and *negative multiembedding* report the number of samples that can be obtained during the same annealing process for the positive and negative statistics, respectively.

ranging from 0 to 1. As a result, the classical samples contain significantly more information, which means a larger number of samples are required for the quantum training. Contrary to the 2000 samples required by the quantum algorithm for both statistics, only 256 samples were sufficient in the classical case to reach a nearly optimal learning curve. The optimal number of samples was obtained through a study on the impact of the training batch size on the quality of the training. Such preliminary testing revealed that, as the batch size grows above 256, the asymptotic results for the reconstruction error degrade. For completeness, we report in Table 4.5 the wall time required for a single epoch of training at each different batch size tested.

tbs	time(s)	σ(s)
32	2.713276	0.055844
64	2.742950	0.017200
128	2.691335	0.065300
256	2.757422	0.031071
512	2.750695	0.024460

TABLE 4.5: The Table presents the time required for a single epoch of training for different batch sizes.

The article presenting the results discussed in this Section is currently undwer review and it will soon be published [24]. For this reason, we decided to omit from this Thesis the main plot of the work, which depicts the comparison between the average log-likelihood and the reconstruction score for both the quantum and classical training. We can nonetheless describe the obtained results numerically. The BM trained with the classical procedure achieves better results in terms of both likelihood and reconstruction error if we compare the methodologies based on the number of epochs. However, in many real-world scenarios it is more relevant to compare the performances of the algorithms at equal wall time. In the 16×16 case, the average wall time required to train the BM for a single epoch in the parallelized classical case is 2.75s (512 samples) while the full quantum algorithm required 0.0572s (4000 samples). With respect to the wall time, the quantum algorithm shows far superior performances, reaching a likelihood of -3.655 in 28.6s, while it takes 247.5s for the classical algorithm to reach the same results, which is 8.6 times longer. In addition, we pictured the hypothetical scenario of an AQC that supports fully parallelized computation of both positive and negative statistics. Here, the assumption is that such hypothetical AQC would be affected by the same errors of the Advantage 4.1, but would contain a number of qubits sufficiently large to generate all the required samples in a single annealing cycle. We estimate that such fully parallelized quantum algorithm would be 95 times faster than the current quantum results (the number of individual calls to the QPU required for training was indeed 95). Thus, the training would become 817 times faster than its classic counterpart running on a Nvidia A5000 GPU. The superior performances achieved in terms of wall time imply that the quantum algorithm could soon become a competitive approach to train BMs. It is worth noting that for this specific application, the challenges of sampling from an embedded model described in Ref. [166] did not pose a significant obstacle to learning. The model appears to be stable even when embedding the negative statistics using 131.8 ± 3.76 qubits arranged in 32 configurations.

A last remark regarding the computational time: the average sampling time required by the D-Wave device to output an individual sample is approximately 600μs, as measured in our experiments. In contrast, the annealing time, which represents the physical duration of the annealing process, was set to 2μs. It means that the time of the sampling process on the AQC is dominated by setup, reading and postprocessing times, rather than the actual duration of the adiabatic evolution of the system. Potential future improvements in this pipeline could lead to a reduction in the sampling time, bringing it closer to the annealing time and further enhancing the advantages of the quantum algorithm. Moreover, the capabilities of AQCs would also benefit from the implementation of more connections among qubits, enabling the training of larger and more complex models.

Future works could focus on training the model using different, more complex datasets, to better test its reconstruction and generation capabilities. Moreover, the quality of the training would benefit from a detailed quantitative analysis of the effective temperature of the AQC. Investigating the relationship between the effective temperature, chain strength J_{chain}, and annealing time τ would provide valuable insights into optimizing the training procedure and achieving better performance.

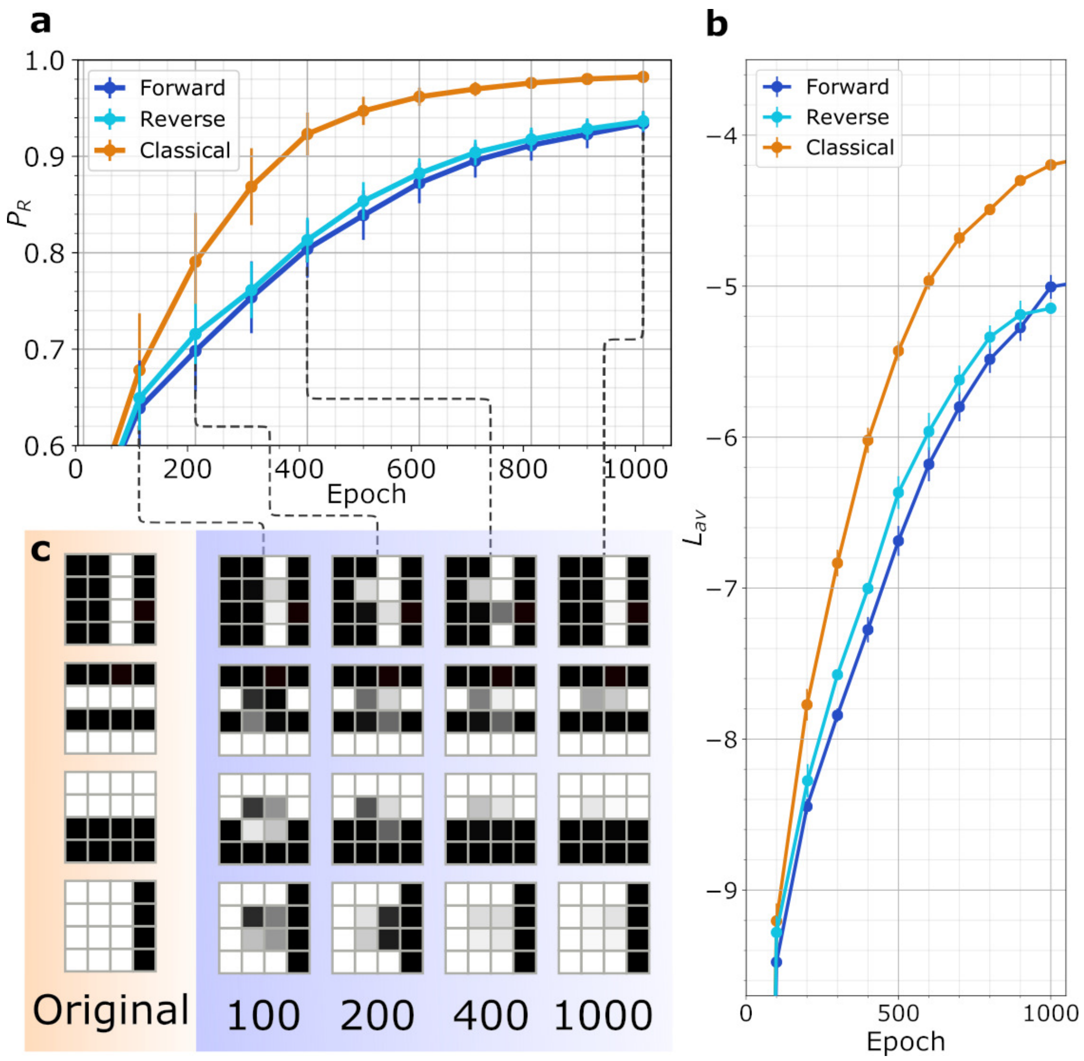

FIGURE 4.18: **a**: Reconstruction score vs training epochs for the classical, forward annealing, and reverse annealing case. The learning curves are obtained with a rescaling parameter $\alpha = 0.32$ and the weights initialization parameters $\mu = 0$, $\sigma = 2$ and $I_W = [-3, 3]$. **b**: Average Log–Likelihood vs training epochs for the classical, forward annealing, and reverse annealing case. **c**: Sample images used during training, taken from the $4x4$ Bars and Stripes dataset, together with the reconstruction obtained with $n_g = 500$ steps of Gibbs sampling at different epochs. The gray-scaled color of the pixels represents the expected value for that pixel according to the chosen RBM.

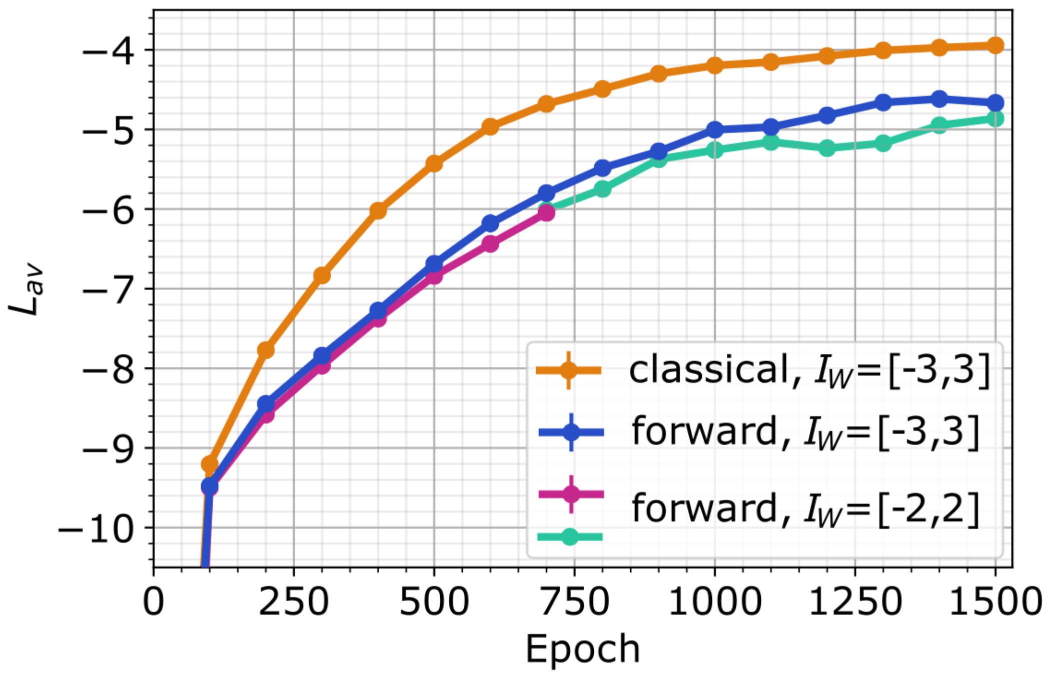

FIGURE 4.19: Comparison between the learning curves obtained using two different I_W for the forward annealing schedule. The embedding of the RBM is complete. The blue line corresponds to $I_W = [-3, 3]$, $\mu = 0$ and $\sigma = 1.5$, at $\alpha = 0.32$. The double-coloured line has initial weights and biases extracted from a gaussian with $\mu = 0$, $\sigma = 1.5$, distributed in $I_W = [-2, 2]$, rescaled with $\alpha = 0.33$. Such value for α has been chosen among five tested so to maximize L_{av} at epoch 700. Five runs at $\alpha = 0.33$ have been trained and averaged up to epoch 700 (violet line), then the best run (green line) has been trained up to epoch 1500. Error bars are hidden by the points in the plot. The performance of the classical algorithm (orange line) is reported for reference.

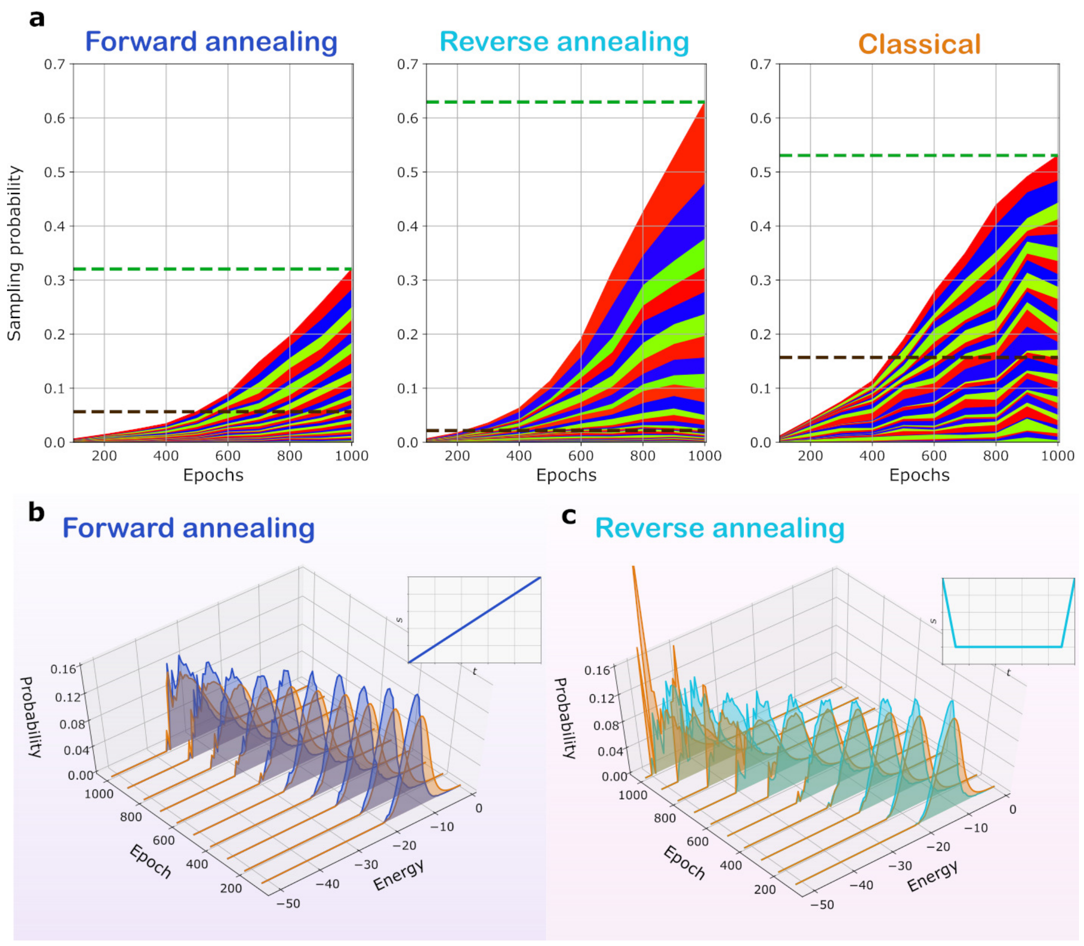

FIGURE 4.20: The distributions presented are reconstructed from the weights obtained during the training processes presented in Figure 4.18. **a**, Each colored band represents the probability that the corresponding image of the dataset is sampled from the Boltzmann distribution defined by the RBM at that epoch. Thus, each plot contains 30 bands. The green line represents the probability that any of the images is sampled at epoch 1000. The black line represents the probability that any of the 15 less probable images is sampled at epoch 1000. **b** and **c** represent the distributions produced by the D–Wave machine using forward annealing (**b**, in blue) and reverse annealing (**c**, in light blue) each 100 epochs, from epoch 100 to epoch 1000, starting from the same weight initialization used in **a**. In both **b** and **c**, the Boltzmann distribution at $T = 1$ corresponding to the instantaneous value of the RBM weights is represented in orange. The qualitative shape of the annealing schedule $s(t)$ is also shown as a reference.

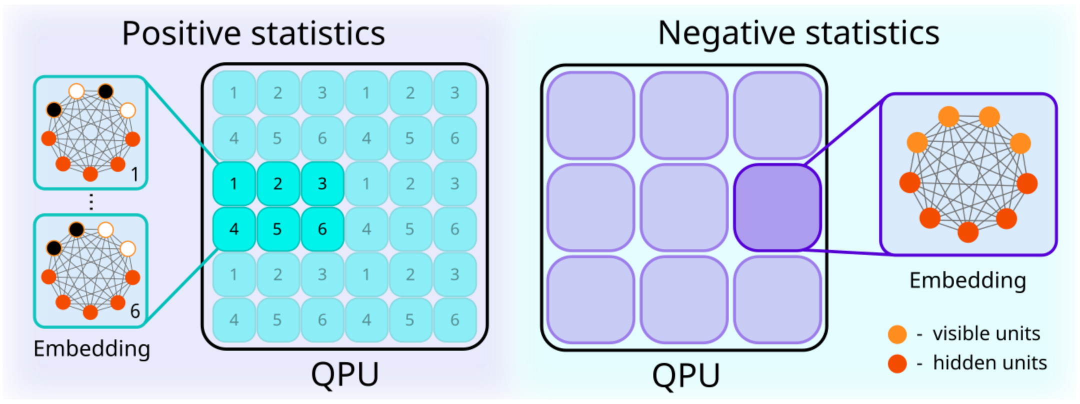

FIGURE 4.21: Comparing positive and negative statistics and their embeddings on the QPU. The left side of the Figure illustrates the multi-embedding of positive statistics. Each numbered square represents a different configuration of hidden nodes, adjusted according to Eq 4.3. The white and black visible nodes have been fixed to 0 and 1, respectively. The numbers highlight the differences between the graphs. The multi-embedding of negative statistics is illustrated on the right side of the Figure. In this scenario, each embedding maps the same graph in a different location on the QPU.

Appendix A

Hard-assignment Gromov-Wasserstein problem

This Section presents the Hard-assignment Gromov-Wasserstein problem (GWP), and explains how to formulate GWP in ILP and QUBO form, to then be able to solve it via AQC and Memcomputing machines. This introduction is functional for the discussion in Section 4.1.2 of the results we obtained in [2], where we challenged Virtual Memcomputing machines and AQCs on GWP.

Optimal transport theory deals with the problem of moving mass from one place to another with minimal effort. This effort is accounted by a cost function; the integral of these mass moves has to be minimized[485]. The main constraint here is represented by mass conservation; this leads naturally to approach optimal transport as a mapping problem between probability distributions. The original problem has two main formulations: the hard formulation from Monge and the relaxed formulation from Kantorovich [485]. In both, one assumes that the two metric spaces involved are the same. The Gromov–Wasserstein distance was introduced by Mémoli [475] and it is an instance of optimal transport between metric spaces having different dimensions (nonregistered) [486]. The GWP finds application in generative machine learning [487], [488] and computer graphics [475], [476], among others. Finding this distance is equivalent to finding a permutation matrix that allows this mapping between distributions.

In literature one can find several regularized, approximate or simplified forms of the Gromov-Wasserstein problem (GWP): Authors in [489] introduce the entropic regularization approach and uses it in combination with the Sinkhorn's matrix scaling algorithm for solving GWP; another paper [490] considers a variant of the optimal transport problem that restricts the set of admissible couplings to those having a low-rank factorization, achieving a linear time approximation for GWP. We will instead consider the problem's *hard–assignment* version, where the desired mapping between points is bijective and the probability distributions in the two spaces assign equal weight to all points. In this form, the cost function is an instance of a Quadratic Assignment Problem (QAP) [477], which makes it an NP-hard problem in general [491].

To define the hard–assignment Gromov–Wasserstein problem, one begins by considering two sets of N points, $S_1 = \{x_1, ..., x_N\}$ and $S_2 = \{y_1, ..., y_N\}$, belonging to two distinct vector spaces each endowed of a distance, $a_{ij} = d_1(x_i, x_j)$ and $b_{hk} = d_2(y_h, y_k)$, respectively. One can define a distance between two pairs of points. In [2], we used the squared euclidean metric $d(a_{ij}, b_{hk}) = (a_{ij} - b_{hk})^2$. To solve the hard–assignment Gromov–Wasserstein problem, one must find a permutation matrix γ ,such that the following expression is minimized:

$$\mathcal{P}_{\mathrm{GW}}(\gamma) = \sum_{ij} \sum_{hk} d(a_{ij}, b_{hk}) \gamma_{ih} \gamma_{jk} \ . \tag{A.1}$$

Starting from the initial formulation $\mathcal{P}_{\mathrm{GW}}$, one can convert the problem into a Integer Linear Programming problem to support VMM and Gurobi. One can indeed create a linear form by introducing n^4 binary variables such that:

$$z_{ihjk} = \gamma_{ih} \gamma_{jk} \tag{A.2}$$

This strategy is typically used to linearize a Quadratic Assignment Problem [492]. The expression to minimize is now:

$$\mathcal{P}^{\mathrm{ILP}}_{\mathrm{GW}}(\{z_{ihjk}\}) = \sum_{ij} \sum_{hk} d(a_{ij}, b_{hk}) z_{ihjk} \tag{A.3}$$

subject to

$$\gamma_{ih} + \gamma_{jk} \leq z_{ihjk} + 1 \quad \forall i, h, j, k \tag{A.4}$$

to enforce expression A.2.

Inequalities in A.4 ensure that the condition $z_{ihjk} = 0$ implies $\gamma_{ij}\gamma_{jk} = 0$, while $\gamma_{ij}\gamma_{jk} = 1$ can be realized only if $z_{ihjk} = 1$. In addition, since $d(a_{ij}, b_{hk}) \geq 0$ for every combination of indices, minimizing A.3 automatically ensures that, if $\gamma_{ij}\gamma_{jk} = 0$, then $z_{ihjk} = 0$. Thus the addition of the constraint in expression A.4 is sufficient to impose $\gamma_{ij}\gamma_{jk} = z_{ihjk}$. One can reduce the number of variables in the problem by noting that, if $i = j$ or $h = k$, then the product on the right hand of expression A.2 equals zero. Indeed, in this case, the two elements of the gamma matrix would belong to the same row or column, which means that at least one of them must equal zero. Furthermore, $z_{ihjk} = z_{jkih}$ holds true. Thus, in the cost function, one should consider only the terms z_{ihjk}, such that $i < j$:

$$\begin{aligned}
\mathcal{P}_{GW}^{\mathrm{ILP}}(\gamma) &= \sum_{i>j}\sum_{hk} d(a_{ij}, b_{hk}) z_{ihjk} + \sum_{i<j}\sum_{hk} d(a_{ij}, b_{hk}) z_{ihjk} = \\
&= \sum_{i<j}\sum_{hk} \left(d(a_{ij}, b_{hk}) + d(a_{ji}, b_{kh})\right) z_{ihjk} = \\
&= 2\sum_{i<j}\sum_{hk} d(a_{ij}, b_{hk}) z_{ihjk} \ ,
\end{aligned} \tag{A.5}$$

since the distance matrices are symmetric. Note that the case $i = j$ has already been ruled out by the previous observation. The above approach reduces the number of z–variables from n^4 to $\frac{1}{2}n^2 \cdot (n-1)^2$, that is, the number of z variables is halved asymptotically.

As we already observed, the matrix γ must be a permutation matrix. The sum of each row must equal one, so one gets n equations with n coefficients, resulting in n^2 nonzero coefficients in the constraint equations. Since the same is true for columns, the total number of nonzero coefficients is $2n^2$. Since VMM is able to implement inequalities only, equations of the form $a = b$ are automatically and internally mapped by VMM into two inequalities $a \leq b$ and $a \geq b$. This this leads to $4n^2$ nonzero coefficients in the constraint expressions. This is an internal remapping of VMM. Both Gurobi and VMM receive the same specification with the equalities as input file.

To implement inequalities, one should consider the condition on γ_{ij} and z_{ihjk} appearing in eq. A.4. The constraint matrix thus contains $3n^2 \cdot (n^2 - 1)/2$ nonzero elements. The total number of nonzero coefficients in the constraint expressions is $\frac{1}{2}n^2 \cdot (3n^2 + 1)$, which become $\frac{1}{2}n^2 \cdot (3n^2 + 5)$ when implemented on VMM. Asymptotically, the number of constraints goes as $\frac{3}{2}n^4$, while the number of binary variables goes as $\frac{1}{2}n^4$, so that the number of constraints is 3 times the number of variables asymptotically. Similarly to the factorization case for D-Wave, we reformulate the constraints as penalty terms. This leads to following cost:

$$\mathcal{P}_{\mathrm{GW}}^{\mathrm{QUBO}}(\{\gamma_{ij}\}) = \mathcal{P}_{\mathrm{GW}}(\{\gamma_{ij}\}) + \lambda R(\{\gamma_{ij}\}) \tag{A.6}$$

where the first term is the original Gromov-Wasserstein cost and the penalty regularizer $R(\{\gamma_{ij}\})$ is ruled by the $\lambda > 0$ coefficient, which enforces the correctness of the γ matrix:

$$R(\{\gamma_{ij}\}) = -4\sum_{i,j} \gamma_{i,j} + \sum_{h,k,j} \gamma_{h,j}\gamma_{k,j} + \sum_{i,h,k} \gamma_{i,h}\gamma_{i,k} \tag{A.7}$$

$R(\{\gamma_{ij}\})$ correctly attains a minimum if and only if all the following constraints are satisfied:

$$\begin{aligned} \sum_i \gamma_{i,j} - 1 = 0 \quad \forall j \\ \sum_j \gamma_{i,j} - 1 = 0 \quad \forall i \end{aligned} \tag{A.8}$$

Indeed, starting from A.8, we can square the left–hand size of both constraints, obtaining two expressions that attain the sole global minimum only when the constraints in A.8 are satisfied (i.e. when γ is a permutation matrix). Summing those two expressions gives back exactly A.7.

The proper value for λ in A.7 must be found by careful testing. This is because a higher or lower value can lead to only the satisfaction of the constraints or only to the minimization of the cost, respectively, while both are desired. The cost function in equation A.6 contains every possible quadratic term of the variables. The graph representing the problem is therefore fully connected, while D-Wave Advantage is topologically characterized by a very sparse graph of qubits. The solution to this hardware limitation is to use embedding techniques to link together nearby qubits with a strong ferromagnetic coupling, so that they behave as a single two-state system. With this approach, one can implement highly connected graphs.

Appendix B

Capacitated Helicopter Routing Problem

The Capacitated Helicopter Routing Problem (CHRP) is a helicopter flight scheduling problem, in which one aims to minimize the overall flight time. This Section explains how to formulate CHRP in ILP and QUBO form, to then be able to solve it via AQC and Memcomputing machines. This introduction is functional for the discussion in Section 4.1.2 of the results we obtained in [2], where we challenged Virtual Memcomputing machines and AQCs on CHRP.

CHRP was introduced in [493], based on the formulation of the Dial-a-ride problem (DARP) [494]. Each flight can have multiple legs to connect offshore oil rigs. The flights are scheduled to transport workers from heliport to rigs, from rig to rig, and from rig to heliport. The problem has limiting constraints, such as the maximum range for each helicopter type and maximum capacities for the weight of the workers and luggage. As a hard constraint, CHRP requires all workers to be transported. CHRP is then a multi-agent routing problem where agents (helicopters) interact through the temporal worker assignment constraints. These characteristics make CHRP very hard, even for small instances, and therefore intractable for real world scenarios. Indeed, CHRP and similar problems such as the DARP are NP-hard in the strong sense [495] since they generalize the Travelling Salesman problem with time windows, which is proven to be NP-complete [496].

Because of its hardness and commercial relevance, multiple heuristics have been developed to provide approximate solutions, using clustering search [497], genetic algorithms [498], and a League Championship Algorithm [499]. CHRP can be cast to an integer linear programming problem with all variables being binary. The problem is then automatically in QUBO form with a null quadratic term and thus usable on D-Wave. The problem size can scale with the number of rigs (locations) and the number of workers, while the maximum number of flights is usually set according to the number of workers. In the Integer Linear

Programming format, all binary variables are decision variables. The formulation has two blocks of constraints. One block defines the helicopter routes as cyclic paths in a dynamic graph. The other block defines the assignment of workers to flights.

Let us now detail better the mathematical formulation. Consider a directed graph $G = (V, E)$, where V is the set of vertices and E the set of edges. The vertex with label 0 maps the heliport, and the remaining vertices map the oil rigs. The graph is fully connected since we have legs connecting all rigs and heliport. We define binary variables $v_{r,t,f}$ being 1 if the flight f stops at the location (rig or heliport) r at time t and binary variable $e_{l,t,f}$ being 1 if the flight f includes the leg l departing at time t. The time t is an integer index that only defines the order of the events and not the actual time. Hence, t has a range that goes from 0 to T, with T being the maximum number of legs for a given flight (T can either be set by the user or evaluated as the maximum number of legs a flight can include given the helicopter range). The index $r \in E$ is 0 for the heliport and ranges from 1 to R for the rigs. The index $l \in E$ spans all possible $2R(R + 1)$ directed edges. Finally, the f is the flight index, which ranges form 1 to F, with F either set by the user or estimated from the number of passengers. To define the equations, it is also useful to introduce maps $\delta : E \to V$ and $\alpha : E \to V$, which return the departing and arrival vertices of an edge, respectively. We can also define the formal inverse maps $\delta^{-1} : V \to E$ and $\alpha^{-1} : V \to E$, which return all outgoing edges from and all incoming edges to a vertex, respectively. Using these definitions, we have

$$\sum_{l\in\delta^{-1}(r)} e_{l,t,f} \leq v_{r,t,f} \qquad t \in \{0,..,T-1\}, \forall r, f \tag{B.1}$$

$$\sum_{l\in\alpha^{-1}(r)} e_{l,t,f} = v_{r,t+1,f} \qquad t \in \{0,..,T-1\}, \forall r, f \tag{B.2}$$

$$\sum_{r} v_{r,t,f} \leq 1 \qquad \forall t, f \tag{B.3}$$

$$\sum_{r} v_{r,t+1,f} \leq \sum_{r} v_{r,t,f} \qquad t = \{0,...,T-1\}, \forall f \tag{B.4}$$

$$v_{r=0,t,f} \leq 1 \qquad t \in \{0,T\}, \forall f \tag{B.5}$$

$$v_{r,t,f} = 0 \qquad t \in \{0,T\}, r \in \{1,...,R\}, \forall f \tag{B.6}$$

$$\sum_{r=1}^{R} v_{r,t+1,f} + v_{r=0,t,f} \leq 1 \qquad t = \{0,...,T-1\}, \forall f \tag{B.7}$$

$$e_{l,t,f} = 0 \qquad l \in \delta^{-1}(r=0), t \in \{1,...,T\}, \forall f \tag{B.8}$$

$$\sum_{t=1}^{T} v_{r=0,t,f} \geq v_{r=0,t=0,f} \qquad \forall f \tag{B.9}$$

$$e_{l,t,f} = 0 \qquad l \in \delta^{-1}(r) \cap \alpha^{-1}(r), t \in \{0,...,T-1\}, \forall f \tag{B.10}$$

$$v_{r,t,f} + v_{r,t+1,f} \leq 1 \qquad t \in \{0,...,T-1\}, \forall r, f \tag{B.11}$$

$$\sum_{l,t} d_l e_{l,t,f} \leq range(f) \qquad \forall f. \tag{B.12}$$

This set of linear relations fully defines each flight. Equation (B.1) requires that, if a flight includes a location at a given time, then that flight can have one leg departing from that location at that time. Equation (B.2) requires that, if a flight includes a location at a give time, then that flight must have a leg arriving to that location at that time. Equation (B.3) requires that the flight includes at most one location at time. Equation (B.4) requires that, if at a given time the flight does not include any location, then it does not include locations at the subsequent times either. Equation (B.5) and (B.6) require that, at time $t = 0$ and $t = T$, if the flight includes locations, they must be the heliport. Equation (B.7) and (B.8) require that, at any time except $t = 0$, a flight cannot have departing legs from the heliport. This means that, if the heliport is visited at a time other than 0, then it must be the last stop. These two constraints are redundant. One would be enough. However, including both help the convergence of solvers like Gurobi since it includes more cutting planes. Equation (B.9) requires that, if a flight has legs, it must stop at the heliport twice. Equations (B.10) and (B.11) require that a helicopter cannot remain two

consecutive times at the same location. Again, these constraints are redundant but they help the convergence of solvers. Finally, equation (B.12) requires that a flight cannot exceed the range associated with the corresponding helicopter. We note that these constraints do not require that a flight has any legs, so we can have empty flights. Moreover, this formulation allows flights to include the same rig visited multiple nonconsecutive times.
This first set of equations defines closed routes for helicopters in terms of closed paths in a graph. However, they are actually independent routes since we have not yet included the worker assignments. To this end, we define the binary variables $pk_{p,t,f}$ being 1 if the helicopter operating the flight f picks up the passenger p at time t and $df_{p,t,f}$ being 1 if the helicopter operating the flight f drops off the passenger p at time t; the passenger index p ranges from 1 to P. We also define maps $\rho : \{1, ..., P\} \to V$ and $\gamma : \{1, ..., P\} \to V$, which return the pick up and drop off locations for each worker, respectively. Conversely, we define the inverse maps $\rho^{-1} : V \to \{1, ..., P\}$ and $V \to \gamma : \{1, ..., P\}$, which return all the workers that need to be picked up and dropped off at a given location, respectively. Using these variables with the previous variables, the assignment problem can be formalized as:

$$\sum_{t,f} pk_{p,t,f} = 1 \qquad \forall p \qquad \text{(B.13)}$$

$$\sum_{t} pk_{p,t,f} = \sum_{t} df_{p,t,f} \qquad \forall p, f \qquad \text{(B.14)}$$

$$\sum_{t} pk_{p,t,f}t \leq \sum_{t} df_{p,t,f}t \qquad \forall p, f \qquad \text{(B.15)}$$

$$\sum_{p\in\rho^{-1}(r)} pk_{p,t,f} + \sum_{p\in\gamma^{-1}(r)} df_{p,t,f} \geq v_{r,t,f} \qquad \forall r, t, f \qquad \text{(B.16)}$$

$$\sum_{p\in\rho^{-1}(r)} pk_{p,t,f} + \sum_{p\in\gamma^{-1}(r)} df_{p,t,f} \leq (|\rho^{-1}(r)| + |\gamma^{-1}(r)|)v_{r,t,f} \qquad \forall r, t, f \qquad \text{(B.17)}$$

$$\sum_{t'=0}^{t}\sum_{p} pk_{p,t',f} - \sum_{t'=0}^{t}\sum_{p} df_{p,t',f} \leq capacity(f) \qquad \forall t, f \qquad \text{(B.18)}$$

$$\sum_{t'=0}^{t}\sum_{p} w_p pk_{p,t',f} - \sum_{t'=0}^{t}\sum_{p} w_p df_{p,t',f} \leq maxweight(f) \qquad \forall t, f \qquad \text{(B.19)}$$

$$\sum_{t'=0}^{t}\sum_{p} wl_p pk_{p,t',f} - \sum_{t'=0}^{t}\sum_{p} wl_p df_{p,t',f} \leq maxluggage(f) \qquad \forall t, f \qquad \text{(B.20)}$$

This second set of equations assigns passengers to flights. Equation (B.13) enforces that all workers are picked up and none are excluded. Equation (B.14) requires that, if the helicopter operating the flight f picks up the worker p at any time, then it must drop off the same passenger at some time. Equation (B.15) enforces that the drop off happens after the pick up. Equations (B.16) and (B.17) enforce that a flight has a leg to the location r if and only if (i.e. the vice versa is enforced too) the helicopter operating that flight either picks up or drops off a worker at that location. In equation (B.17), $|\rho^{-1}(r)|$ and $|\gamma^{-1}(r)|$ are the number of elements returned by the inverse maps. Equation (B.18) requires that the number of passengers on the helicopter operating the flight f does not exceed its maximum capacity at any time. Equation (B.19) requires that the total passenger weight on the helicopter operating the flight f does not exceed its maximum weight capacity at each instant of time. Equation (B.20) requires that the total luggage weight on the helicopter operating the flight f does not exceed its maximum luggage weight capacity at each instant of time. The model is finally completed by the cost function:

$$\min_{e} \sum_{l,t,f} d_l e_{l,t,f}. \tag{B.21}$$

Appendix C

Non-Negative Binary Matrix Factorization

This Section presents the Non-Negative Binary Matric Factorization problem (NBMF), and explains how to formulate it in QUBO form, to then be able to solve it via AQC. This introduction is solely functional to the discussion in Section 4.1.1 of the results we obtained in [1], where we challenged various generations of AQCs and the classical solver Gurobi on NBMF. Further insights regarding the relevance of NBMF and previous literature on the topic can be found in the introduction of Section 4.1.1.

C.1 Problem definition

Given a dataset of m images composed by $\sqrt{n} \times \sqrt{n}$ pixels, we seek to find the best possible representation of each image through a linear combination of $k << m$ basis images weighted with binary values, i.e. 0 or 1. In other words, we look for a set of k basis images to optimally span via linear combination the data-space of reference when only binary coefficients are made available. Such a problem can be formally cast in a matrix factorization problem. Given a matrix V of size $n \times m$, with its columns encoding the m flattened gray-scale images of the original dataset, the goal is to determine the non-negative matrices W (real) and H (binary) of size $n \times k$ and $k \times m$ respectively, for which $||V - WH||_F$ is minimum, where the norm is the Frobenius norm (square root of the summation of the square of every element in the matrix). The complexity of the task comes from the fact that both matrices W and H must be optimized to achieve the best possible approximation for V. We call W the basis-image matrix, and H the reconstruction, or latent matrix. The columns of the matrix W encode the flattened basis images used to reconstruct the original images of the database. The columns of the H matrix, instead, encode the binary weights associated with the basis images. These are to be interpreted as the values of the features of the original images in a binary latent space over which to perform classification. A lower number of basis images will generally imply a poorer capability to reconstruct the original images in the dataset. On the other hand, setting a higher number of features will diminish the advantages of FE, and each value in the binary latent space decomposition will carry less information.

C.2 Computational procedure

As discussed in Refs. [469], [470], the original problem of factorizing the matrix V into the matrix product WH via NBMF can be recast into an iterative optimization problem where either W and H are optimized one at a time, alternatively. The full problem can hence be decomposed into two consecutive optimization steps as follows:

$$\textbf{find\_W} := \arg \min_{X \in \mathbb{R}^{n \times k}} ||V - XH||_F + \alpha ||X||_F, \tag{C.1}$$

$$\textbf{find\_H} := \arg \min_{X \in \{0,1\}^{k \times m}} ||V - WX||_F, \tag{C.2}$$

where $|| \cdot ||_F$ represents the Frobenius norm, a measure of the distance between two matrices, and $\alpha \in \mathbb{R}^+$ a free parameter to be tweaked beforehand. The second term in eq.C.1 is used as a regularization component to penalize solutions with large $||X||_F$. Such a term forces the candidate H matrix to be sparse, so that images in V will be reconstructed by combining only some of the basis images in W. The two optimization problems are then solved iteratively until the condition $||V - WH||_F < \epsilon$ is matched, with ϵ being the desired threshold error in the reconstruction of V.

Finding W– the first problem is solved by finding the X matrix that minimizes the quadratic cost function

$$C = ||V - XH||_F^2 + \alpha ||X||_F^2, \tag{C.3}$$

with H initialized as discussed in Sec. 4.1.1.

This quadratic form is minimized via the Gurobi [474] mathematical programming solver. Gurobi implements a wide array of heuristics that make it a reference tool for the minimization of quadratic cost functions. It is a licensed software with a free license for academic utilization.

Finding H– as in Ref.[469], the original problem of finding the H matrix that best reproduces V given W can be reduced to a set of independent optimization sub-problems to be solved for each image in the dataset. We can indeed solve:

$$H_z = \arg \min_{\mathbf{q}\in\{0,1\}^k} ||V_z - W\mathbf{q}||_2, \tag{C.4}$$

where $z = 1, 2, ..., m$, and $\mathbf{q}$ is an array of k binary variables. The L^2 norm is used here in place of the Frobenius norm to account for dealing with vectors rather than matrices.

Such a problem can be solved by minimizing the correspondent Quadratic Unconstrained Binary Optimization (QUBO) cost function, obtained by squaring the norm in Eq. C.4:

$$Q(\mathbf{q}) = \sum_i a_i q_i + \sum_{i<j} b_{ij} q_i q_j, \tag{C.5}$$

with

$$\begin{aligned} a_i &= \sum_l W_{li}(W_{li} - 2V_{lz}), \\ b_{ij} &= 2\sum_l W_{li} W_{lj}. \end{aligned} \tag{C.6}$$

Such a cost function is then minimized with respect to the binary variables $q_i \in \{0, 1\}$.

Appendix D

Estimation of the positive and negative statistics for a RBM

D.1 Estimation of RBM statistics using classical methods

Because there are no direct connections between hidden units in an RBM, it is very easy to get an unbiased sample of $\langle v_i h_j \rangle_{data}$. Given a randomly selected training vector $\mathbf{r}$, the binary state h_j of each hidden unit j is set to 1 with probability

$$\begin{aligned}
&p(h_j = 1 \mid \mathbf{r}) \\
&\quad = \sigma(-E(\{s_i\}_{i \neq j}, s_j = 1) + E(\{s_i\}_{i \neq j}, s_j = -1)) \\
&\quad = \sigma(-2b_j - 2\sum_i r_i w_{ij}) \\
&\quad = (1 + \langle h_j \rangle_{\text{data}})/2
\end{aligned} \tag{D.1}$$

where $\sigma(x) = 1/(1 + \exp(-x))$ is the logistic sigmoid function, and the last step has been added since it is useful for the hidden bias gradient estimation (Eq. 2.36).

Because there are no direct connections between visible units in an RBM, it is also very easy to get an unbiased sample of the state of a visible unit, given a hidden vector.

$$\begin{aligned}
p(v_i = 1 \mid \mathbf{h}) &= \sigma(-2a_i - 2\sum_j h_j w_{ij}) \\
&= (1 + \langle v_i \rangle_{\text{data}})/2\ .
\end{aligned} \tag{D.2}$$

The last step is needed for the gradient estimation of the visible biases (Eq. 2.35).

Getting an unbiased sample of $\langle v_i h_j \rangle_{model}$ is much more difficult. An exact computation would involve a summation over all possible combinations of values for hidden and visible units, which means summing over 2^N elements. Each unit would then double the computational cost of this procedure, leading to a computation time that is exponential in the number of units. In many practical situations, an exact computation of the negative gradient is unfeasible. Note that the same effort is needed to compute $\langle h_j \rangle$. Nonetheless, an estimation technique that allows computing $\langle v_i h_j \rangle_{model}$ collaterally produces an estimation for $\langle h_j \rangle$, so we will focus on estimating the negative statistics.

The negative statistics is usually approximated in the following way. First, the values of the visible units are set equal to those of a training vector. Then the binary states of the hidden units are all computed in parallel using Eq. D.1. Once binary states have been chosen for the hidden units, a reconstruction is produced by setting each v_i to 1 with a probability given by Eq. D.2 [332]. This process is called *Gibbs sampling* and generates real values for each unit of the Boltzmann machine, which can be used to estimate $\langle v_i h_j \rangle_{model}$.

CD$_n$ will be used to denote learning using n full steps of alternating Gibbs sampling. Indeed, RBMs typically learn better if more steps of alternating Gibbs sampling are used before collecting vectors for the negative statistics. It happens because a greater number of steps grants a better convergence to the distribution encoded in the weights of the RBM. It is important to note that Gibbs sampling is not required to start by initializing visible units with a vector taken from the training set. Nonetheless, it has been observed that this choice enhances the training quality.

Gibbs sampling method corresponds to putting the system in contact with a *thermal reservoir*. Indeed, $\langle v_i h_j \rangle$ should be estimated by averaging over a collection of configurations extracted from the Boltzmann distribution

$$P(\{v_i\}, \{h_j\}) = \frac{e^{-E(\{v_i\},\{h_j\})}}{Z} \ , \tag{D.3}$$

where Z is the partition function

$$Z = \sum_{\{v_i\},\{h_j\}} e^{-E(\{v_i\},\{h_j\})} \tag{D.4}$$

where the summation is to be performed over each combination of hidden and visible units.

Gibbs sampling consists of updating the hidden units and then updating the visible units. While updating the hidden units, it is very important to make the hidden states binary rather than using the probabilities themselves. If the probabilities are used, each hidden unit can communicate a real-value to the visible units during the reconstruction. This seriously violates the information bottleneck created by the fact that a hidden unit can convey at most one bit of information (on average). This information bottleneck acts as a strong regularizer [332].

For the last update of the hidden units, it is silly to use stochastic binary states because nothing depends on which state is chosen. So it is customary to use the probability itself to avoid unnecessary sampling noise. When using CD$_n$, only the final update of the hidden units should use the probability.

For the visible units, it is common to use the probability itself instead of sampling a binary value. This is not nearly as problematic as using probabilities for the data-driven hidden states and it reduces sampling noise thus allowing faster learning.

It is possible to update the weights after estimating the gradient on a single training case, but it is often more efficient to divide the training set into small *mini-batches* of 10 to 100 cases.

It is a serious mistake to make the mini-batches too large when using stochastic gradient descent. Increasing the mini-batch size by a factor of N leads to a more reliable gradient estimate but it does not increase the maximum stable learning rate by a factor of N, so the net effect is that the weight updates are smaller per gradient evaluation.

D.2 Estimation of RBM statistics using an AQC

The annealing Hamiltonian H_P appearing in Eq. 1.37 can be rewritten as

$$H_P^R = \sum_{ij\in\rho} J_{ij}\sigma_i^z\sigma_j^z + \sum_{i\in\nu} A_i\sigma_i^z + \sum_{j\in\lambda} B_j\sigma_j^z \tag{D.5}$$

were the superscript R refers to restricted. A_i are the biases corresponding to the visible units, while B_j are those of hidden units. Bias summations are to be performed over the two distinct sets ν and λ, which collect each qubit corresponding to either a visible or hidden unit, respectively. W_{ij} are the weights of the RBM, and the summation of such terms is to be performed on the set ρ of non-zero couplers. The introduction of set ρ is useful also to remember that the physical realizations of QA do not allow us to implement all RBMs. Indeed, the low connectivity of physical devices often forces the user to use a simplified version of an RBM.

D.2.1 Model states generation (negative statistics)

This Section and the following one explain how an AQC can be used to estimate the positive and negative part of the weights update rule for RBMs presented in Eq. 2.34.

First, consider the negative statistics defined in Eq. 2.33:

$$\langle v_i h_j \rangle_{model} = \frac{\sum_{\{v\},\{h\}} v_i h_j e^{-E(\{v\},\{h\})}}{\sum_{\{v\},\{h\}} e^{-E(\{v\},\{h\})}} \, , \tag{D.6}$$

where the sum over $\{v\}, \{h\})$ must be taken over each possible configuration of the visible and hidden units.

For the moment we assume that the effective temperature T_{eff} can vary but it is known for each quantum annealing instance. Then the expectation value in D.6 can be estimated thanks to the following algorithm.

Algorithm 1: Quantum algorithm for the negative statistics estimation with known effective temperature

```
Input: Number of visible units $n_v$, number of hidden units $n_h$, desired
       number of iterations $N_{\text{iter}}$, initial weights $w_{ij}$, initial biases $a_i$
       and $b_i$, waiting time $t$. We suppose $T_{\text{eff}}^{(j)}$ is known for each step
       $j$.
Output: Matrix $ans_{pq}$ of dimensions $(n_v, n_h)$ that will contain the
        estimate for $\langle v_p h_q \rangle_{model}$.
Functions: $eval(T_{\text{eff}}, w_{ij}, a_i, b_i)$ returns values for $J_{ij}$, $A_i$ and $B_j$
           correctly rescaled to account for the effective temperature;
           $anneal(J_{ij}, A_i, B_j)$ uses quantum annealing with
           $H_P^R(J_{ij}, A_i, B_j)$ as final Hamiltonian (Eq. D.5);
           $measure(x_i)$ operates a measure operation over unit $x_i$.
1 $ans_{pq} \leftarrow 0 \;\; \forall p < n_v \;\; \forall q < n_h$
2 for $j = 0$ to $N_{iter}$ do
3     $[J_{ij}, A_i, B_j] \leftarrow \text{eval}(T_{\text{eff}}^{(j)}, w_{ij}, a_i, b_i)$
4     $\text{anneal}(J_{ij}, A_i, B_j)$
5     $v_p \leftarrow \text{measure}(v_p) \;\; \forall p < n_v$
6     $h_q \leftarrow \text{measure}(h_q) \;\; \forall q < n_h$
7     $ans_{pq} \leftarrow ans_{pq} + v_p h_q \;\; \forall p < n_v \;\; \forall q < n_h$
8 $ans_{pq} \leftarrow \frac{ans_{pq}}{N_{iter}} \;\; \forall p < n_v \;\; \forall q < n_h$
```

This algorithm outputs an estimate for $\langle v_i h_j \rangle_{model}$ by averaging over N_{iter} samples. The *anneal*() step is the only quantum computation on the input data required to obtain the samples. If we do not consider the computational cost of initializing the system, the quantum algorithm needs $O(1)$ (quantum) operations to obtain a single sample. As opposed to this, contrastive divergence algorithm needs $O(n_g \cdot (n_h + n_v))$ operations, where n_g is the number of rounds for the Gibbs sampling.

In addition to this, the classical algorithm is an *inexact* approach, since the contrastive divergence method does not approximate the correct model distribution $P(v, h)$. On the other hand, the proposed quantum algorithm is supposed to sample from a Hamiltonian that is similar to the desired Hamiltonian, thus obtaining unit values distributed according to the desired Gibbs distribution. Note that this proposition is true only if T_{eff} is known exactly.

A fundamental remark to make is that a sample produced using Gibbs sampling possesses information content that is higher than that of a sample produced through quantum annealing. In general contrastive divergence samples continuum values for hidden units, while Algorithm 1 samples binary values, as obtained from a measurement operation. As explained by Hinton [332], while a binary sampling could be considered closer to the mathematical model of an RBM, using the continuous probability value for hidden variables is usually preferable at a classical level because it reduces sampling noise, thus allowing faster learning. Algorithm 1 in this form could then be affected by a slower learning speed. It follows that to yield the same information content, the quantum algorithm has to produce more samples.

Suppose that Algorithm 1 needs N_{qsamp} samples for each epoch to generate a "good" average sample. To sum up, the computational cost to generate the estimate positive statistics (for a single epoch) is:

- **Classic** : $\mathcal{O}\big(N_{\text{batch}} \cdot n_g \cdot (n_h + n_v)\big)$
- **Quantum** : $\mathcal{O}\big(N_{\text{qsamp}}\big)$

where $N_{\text{qsamp}} > N_{\text{batch}}$ since the number of quantum samples must be greater than the number of classical samples, as mentioned. In Section 4.2.1, a 16x16 RBM is trained using this algorithm for the estimation of the negative statistics. In such case, a good learning process required $N_{\text{qsamp}} \approx 800$, while $N_{\text{batch}} \approx 30$ in the same case.

It is interesting to observe how the computational cost can be reduced by parallelizing the computation on both hardware. The parallelization of classical computations is nowadays extremely developed, with an increasing interest in exploiting Graphics Processing Units (GPUs) to execute thousand of different operations at the same time.

In AQCs classical parallelization capabilities appear as a natural property of the quantum hardware. Indeed, any problem that uses a subset of the processor graph can be cloned multiple times on different locations of the processor. The number of copies depends on the ratio of the processor size to the number of qubits required by the problem. Since each copy has no active connections with the other copies, multiple results are collected during a single annealing cycle, without causing correlations among them.

We now suppose to push to the extreme the parallelization capabilities of the two hardware, supposing we have at our disposal an infinite number of classical processors and an AQC with an infinite number of qubits. For the classical algorithm, it means that we can parallelize the computation for each element of the batch and the update of each unit. For the quantum algorithm described in this Section, having an infinite number of qubits means that we can clone the problem in N_{qsamp} locations. As a consequence, the computational cost of estimating the negative statistics changes as follows:

- **Classic** (**Parallelized**) : $\mathcal{O}(n_g)$
- **Quantum** (**Parallelized**) : $\mathcal{O}(1)$

In the classical case, Gibbs sampling proceeds by processing information created in the previous step, so it is not parallelizable. On the other hand, Algorithm 1 does not possess any iterative step, then it is parallelizable.

A remark must be made regarding computational cost. Nowadays, quantum computing can not offer an advantage for algorithms that requires a number of elementary operations equal to the classical case. Indeed, a single operation on a modern quantum annealer requires several microseconds, while a classical processor executes elementary operations in nanoseconds. Nonetheless, when talking about computational cost, it makes sense to not take into consideration the time required for a single operation. The annealing schedule usually does not vary much as the system size grows, so that the annealing time can be considered a constant. Besides that, larger problem sizes can require more cycles (in the present case, N_{qsamp} incorporates such tendency).

D.2.2 Data states generation (positive statistics)

Consider the positive statistics introduced in Eq. 2.33:

$$\langle v_i h_j \rangle_{data} = \frac{1}{N_{\text{batch}}} \sum_{\{v \in \text{batch}\}} \frac{\sum_{\{h\}} v_i h_j e^{-E(\{h\},v)}}{\sum_{\{h\}} e^{-E(\{h\},v)}} , \tag{D.7}$$

where we have supposed to make use of batches of size N_{batch}.

This expectation value can be estimated in a way similar to **Algorithm 1**, but in this case, we have to execute annealing with visible units clamped to the dataset values. This can be done easily by considering the visible units as biases that influence the hidden units state. The Hamiltonian H_P^R (Eq. D.5) can then be rewritten as:

$$\begin{aligned} H_P^{RC} &= \sum_{j \in \lambda_R} B_j^C \sigma_j^z , \\ B_j^C &= B_j \quad + \sum_{i \ s.t.(i,j) \in \rho_R} J_{ij} v_i , \end{aligned} \tag{D.8}$$

where v_i are the values assumed by the visible units. Such values are different for each element of the dataset. The superscript C stays for 'clamped'.

Algorithm 2: Quantum algorithm for the positive statistics estimation with known effective temperature

```
Input: Number of visible units $n_v$, number of hidden units $n_h$, dataset
       containing $N_{data}$ vectors $x^j$ of length $n_v$, initial weights $w_{ij}$,
       initial biases $b_i$, desired number of iterations $N_{\text{iter}}$ for each
       element of the dataset. We suppose $T_{\text{eff}}^{(k)}$ is known for each step
       $k$.
Output: Matrix $ans_{pq}$ of dimensions $(n_v, n_h)$ that will contain the
        estimate for $\langle v_p h_q \rangle_{data}$
Functions: $eval(T_{\text{eff}}, w_{ij}, b_j)$ updates values for $J_{ij}$ and $B_j^C$ correctly
           rescaled to account for the effective temperature and
           modified according to D.8; $anneal(J_{ij}, B_j^C)$ uses quantum
           annealing with $H_P^{RC}(J_{ij}, B_j^C)$ as the final Hamiltonian (Eq.
           D.8); $measure(x_i)$ operates a measure operation over unit
           $x_i$.
1 $ans_{pq} \leftarrow 0 \;\; \forall p < n_v \;\; \forall q < n_h$
2 for $k = 0$ to $N_{data}$ do
3     for $m = 0$ to $N_{iter}$ do
4         $v_p \leftarrow x_p^k \;\; \forall p < n_v$
5         $[J_{ij}, B_j^C] \leftarrow \text{eval}(T_{\text{eff}}^{(j)}, w_{ij}, b_i)$
6         $\text{anneal}(J_{ij}, B_j^C)$
7         $h_q \leftarrow \text{measure}(h_q) \;\; \forall q < n_h$
8         $ans \leftarrow ans + v_p h_q \;\; \forall p < n_v \;\; \forall q < n_h$
9 $ans_{pq} \leftarrow \frac{ans_{pq}}{N_{\text{data}} N_{\text{iter}}} \;\; \forall p < n_v \;\; \forall q < n_h$
```

Algorithm 2 requires $O(N_{\text{data}} \cdot N_{\text{iter}})$ operations to compute the estimate. N_{iter} can depend on the problem and the quality of the hardware. It is the number of iterations the user thinks is sufficient to produce a representative sample. A reasonable value for the product $O(N_{\text{data}} \cdot N_{\text{iter}})$ is of the order of N_{qsample} used in the previous Section. Nonetheless, references in literature about empirical values for N_{iter} are almost inexistent, since the estimation of the positive statistics is hardly ever performed using quantum hardware. The reason for that is the following. The classical algorithm requires $O(N_{\text{data}} \cdot n_h)$ operations, which comes from updating the state of each hidden unit once for each element of the dataset. By using multiple processors, such computational cost can be reduced to $O(1)$, since the update of hidden units does not require information other than the dataset and the actual weights of the RBM. On an AQC, the update of a single hidden unit can be executed on a single qubit. If the hardware possesses more than $N_{\text{data}} \cdot N_{\text{iter}}$ qubits, the cost of Algorithm 2 is $O(1)$, since all the samples can be produced in a single annealing cycle.

It can be concluded that for the positive statistics case the quantum algorithm does not speed up the process in terms of the number of elementary operations. Indeed, the physical time for a single quantum operation is orders of magnitude greater than a classical one. This means that an optimized quantum algorithm for training RBMs on AQCs estimates the negative statistics using Algorithm 1, and the positive statistics using the classical update of hidden units (Eq. D.1).

Appendix E

Training a RBM using reverse annealing

Algorithm 3: Quantum algorithm for the negative statistics estimation using reverse annealing

Input: Number of visible units n_v, number of hidden units n_h, desired number of iterations N_{iter}, initial weights w_{ij}, initial biases a_i and b_i, annealing schedule $s(t)$ such that $s(0) = s(1) = 1,\ 0 < s(t) < 1\ \forall t \neq 0, 1$, training dataset $\mathcal{D}$ composed by $N_{\mathcal{D}}$ elements. We suppose $T_{\text{eff}}^{(j)}$ is known for each step j.

Output: Matrix ans_{pq} of dimensions (n_v, n_h) that contains the estimates for $\langle v_p h_q \rangle_{model}\ \forall p,\ \forall q$.

Functions: $eval(T_{\text{eff}}, w_{ij}, a_i, b_j)$ returns values for J_{ij}, A_i and B_j to be loaded to the QPU, rescaled by the parameter α that estimates the effective temperature[a]; $initialize\_qubits(\mathcal{D}, k)$ initialize each visible units of the RBM with the corresponding pixel in the k-th element of $\mathcal{D}$, while hidden units are updated at $T = 1$ with a single step of Gibbs sampling, then qubits of the AQC are initialized with the corresponding values mapped in $\{-1, +1\}$; $anneal(J_{ij}, A_i, B_j, s(t))$ uses quantum annealing with $H_P(J_{ij}, A_i, B_j)$ as final Hamiltonian and $s(t)$ as annealing schedule; $measure(x_i)$ retrieves the binary value of x_i after the annealing process ends.

```
1 $ans_{pq} \leftarrow 0\ \ \forall p,\ \ \forall q$
2 for $m = 0$ to $N_{\mathcal{D}}$ do
3     for $k = 0$ to $N_{iter}/N_{\mathcal{D}}$ do
4         initialize_qubits$(\mathcal{D}, m)$
5         $[J_{ij}, A_i, B_j] \leftarrow$ eval$(T_{\text{eff}}^{(j)}, w_{ij}, a_i, b_i)$
6         anneal$(J_{ij}, A_i, B_j)$
7         $v_p \leftarrow$ measure$(v_p)\ \ \forall p$
8         $h_q \leftarrow$ measure$(h_q)\ \ \forall q$
```

In [3], we were the first to modify the usual quantum algorithm for the estimation of $\langle v_i h_j \rangle_{model}$, which exploited forward annealing, to exploit a sampling procedure based on reverse annealing.

If compared to the usual procedure, the N_{iter} annealing cycles are performed separated in $N_{\mathcal{D}}$ groups. For each group, qubits corresponding to visible units are initialized with an element from the dataset. Qubits corresponding to hidden units are initialized with binary values classically computed with a single step of Gibbs sampling. After all elements of the dataset have been used to initialize the annealing process, the output configurations are used to estimate $\langle v_i h_j \rangle_{model}$.

Bibliography

[1] L. Rocutto, M. Maronese, D. Dragoni, A. Cavalli, and C. Cavazzoni, "Comparing adiabatic quantum computers for satellite images feature extraction," to be published.

[2] L. Rocutto, M. Maronese, F. Traversa, S. Decherchi, and A. Cavalli, "Assessing the effectiveness of non-turing computing paradigms," *submitted*, 2023.

[3] L. Rocutto and E. Prati, "A complete restricted boltzmann machine on an adiabatic quantum computer," *International Journal of Quantum Information*, vol. 19, no. 04, p. 2 141 003, 2021.

[4] L. Rocutto, C. Destri, and E. Prati, "Quantum semantic learning by reverse annealing of an adiabatic quantum computer," *Advanced Quantum Technologies*, vol. 4, no. 2, p. 2 000 133, 2021.

[5] L. F. Menabrea, "Sketch of the Analytical Engine (1843) with notes by the translator, Ada Agusta, Countess of Lovelace," in https://doi.org/10.7551/mitpress/12274.003.0005, 2021, ch. 3, pp. 9–26.

[6] J. Schmidhuber, "Colossus was the first electronic digital computer," *Nature*, vol. 441, no. 7089, pp. 25–25, 2006.

[7] R. Herken, *The Universal Turing Machine A Half-Century Survey.* Springer-Verlag, 1995.

[8] S. Wolfram and M. Gad-el-Hak, "A new kind of science," *Appl. Mech. Rev.*, vol. 56, no. 2, B18–B19, 2003.

[9] G. E. Moore *et al.*, *Cramming more components onto integrated circuits*, 1965.

[10] G. E. Moore *et al.*, "Progress in digital integrated electronics," in *Electron devices meeting*, Washington, DC, vol. 21, 1975, pp. 11–13.

[11] L. B. Kish, "End of moore's law: Thermal (noise) death of integration in micro and nano electronics," *Physics Letters A*, vol. 305, no. 3-4, pp. 144–149, 2002.

[12] T. N. Theis and H.-S. P. Wong, “The end of moore’s law: A new beginning for information technology,” *Computing in Science & Engineering*, vol. 19, no. 2, pp. 41–50, 2017.

[13] E. Peláez, “Parallelism and the crisis of von neumann computing,” *Technology in Society*, vol. 12, no. 1, pp. 65–77, 1990.

[14] Y. Manin, “Computable and uncomputable,” *Sovetskoye Radio, Moscow*, vol. 128, 1980.

[15] P. Benioff, “The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines,” *Journal of statistical physics*, vol. 22, no. 5, pp. 563–591, 1980.

[16] D. Deutsch and R. Jozsa, “Rapid solution of problems by quantum computation,” *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, vol. 439, no. 1907, pp. 553–558, 1992.

[17] L. K. Grover, “A fast quantum mechanical algorithm for database search,” in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996, pp. 212–219.

[18] P. W. Shor, “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer,” *SIAM review*, vol. 41, no. 2, pp. 303–332, 1999.

[19] D. Coppersmith, “An approximate fourier transform useful in quantum factoring,” *arXiv preprint quant-ph/0201067*, 2002.

[20] D. E. Deutsch, “Quantum computational networks,” *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, vol. 425, no. 1868, pp. 73–90, 1989.

[21] D-Wave Systems Inc., *Qpu-specific physical properties: Advantage_ system4.1*, 2022.

[22] IBM, *IBM Unveils 400 Qubit-Plus Quantum Processor and Next-Generation IBM Quantum System Two*, https://newsroom.ibm.com/2022-11-09-IBM-Unveils-400-Qubit-Plus-Quantum-Processor-and-Next-Generation-IBM-Quantum-System-Two, Accessed: 30-Oct-2023, 2022.

[23] L. Rocutto, D. Noè, L. Moro, and E. Prati, “Fast training of fully-connected boltzmann machines on an adiabatic quantum computer,” to be published.

[24] D. Noè, L. Rocutto, L. Moro, and E. Prati, "Adiabatic quantum computers are faster than gpus for training fully-connected boltzmann machines," to be published.

[25] B. Schumacher, "Quantum coding," *Physical Review A*, vol. 51, no. 4, p. 2738, 1995.

[26] J. Preskill, "Quantum computing in the nisq era and beyond," *Quantum*, vol. 2, p. 79, 2018.

[27] J. Chow, O. Dial, and J. Gambetta, "Ibm quantum breaks the 100-qubit processor barrier," *IBM Research Blog*, 2021.

[28] F. Arute, K. Arya, R. Babbush, *et al.*, "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 505–510, 2019.

[29] E. Pednault, J. Gunnels, D. Maslov, and J. Gambetta, *\NoCaseChange{https://www.ibm.com/blogs/research/2019/10/on-quantum-supremacy/}*, 2019.

[30] Y. Liu, X. Liu, F. Li, *et al.*, "Closing the" quantum supremacy" gap: Achieving real-time simulation of a random quantum circuit using a new sunway supercomputer," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2021, pp. 1–12.

[31] F. Pan, K. Chen, and P. Zhang, "Solving the sampling problem of the sycamore quantum circuits," *Phys. Rev. Lett.*, vol. 129, p. 090 502, 9 Aug. 2022. DOI: 10.1103/PhysRevLett.129.090502. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.129.090502.

[32] D. P. DiVincenzo, "The physical implementation of quantum computation," *Fortschritte der Physik: Progress of Physics*, vol. 48, no. 9-11, pp. 771–783, 2000.

[33] C. C. McGeoch, R. Harris, S. P. Reinhardt, and P. I. Bunyk, "Practical annealing-based quantum computing," *Computer*, vol. 52, no. 6, pp. 38–46, 2019.

[34] E. G. Rieffel and W. H. Polak, *Quantum computing: A gentle introduction*. MIT Press, 2011.

[35] M. Maronese, L. Moro, L. Rocutto, and E. Prati, "Quantum compiling," in *Quantum Computing Environments*, Springer, 2022, pp. 39–74.

[36] D. Castelvecchi, "Ibm's quantum cloud computer goes commercial," *Nature*, vol. 543, no. 7644, 2017.

[37] G. J. Mooney, G. A. White, C. D. Hill, and L. C. Hollenberg, "Whole-device entanglement in a 65-qubit superconducting quantum computer," *Advanced Quantum Technologies*, vol. 4, no. 10, p. 2 100 061, 2021.

[38] M. Reagor, C. B. Osborn, N. Tezak, *et al.*, "Demonstration of universal parametric entangling gates on a multi-qubit lattice," *Science advances*, vol. 4, no. 2, eaao3603, 2018.

[39] Intel, `\NoCaseChange{https://www.intel.com/content/www/us/en/research/quantum-computing.html}`, 2021.

[40] H. J. Briegel, D. E. Browne, W. Dür, R. Raussendorf, and M. Van den Nest, "Measurement-based quantum computation," *Nature Physics*, vol. 5, no. 1, pp. 19–26, 2009.

[41] D. Gottesman and I. L. Chuang, "Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations," *Nature*, vol. 402, no. 6760, pp. 390–393, 1999.

[42] R. Raussendorf and H. J. Briegel, "A one-way quantum computer," *Physical review letters*, vol. 86, no. 22, p. 5188, 2001.

[43] H. J. Briegel and R. Raussendorf, "Persistent entanglement in arrays of interacting particles," *Physical Review Letters*, vol. 86, no. 5, p. 910, 2001.

[44] J. E. Bourassa, R. N. Alexander, M. Vasmer, *et al.*, "Blueprint for a scalable photonic fault-tolerant quantum computer," *Quantum*, vol. 5, p. 392, 2021.

[45] I. L. Chuang, L. M. Vandersypen, X. Zhou, D. W. Leung, and S. Lloyd, "Experimental realization of a quantum algorithm," *Nature*, vol. 393, no. 6681, pp. 143–146, 1998.

[46] Y. Nakamura, Y. A. Pashkin, and J. Tsai, "Coherent control of macroscopic quantum states in a single-cooper-pair box," *nature*, vol. 398, no. 6730, pp. 786–788, 1999.

[47] J. R. Friedman, V. Patel, W. Chen, S. Tolpygo, and J. E. Lukens, "Quantum superposition of distinct macroscopic states," *nature*, vol. 406, no. 6791, pp. 43–46, 2000.

[48] J. Clarke and A. I. Braginski, *The SQUID handbook*. Wiley Online Library, 2004, vol. 1.

[49] E. Knill, R. Laflamme, and G. J. Milburn, “A scheme for efficient quantum computation with linear optics,” *nature*, vol. 409, no. 6816, pp. 46–52, 2001.

[50] J. L. O’Brien, G. J. Pryde, A. G. White, T. C. Ralph, and D. Branning, “Demonstration of an all-optical quantum controlled-not gate,” *Nature*, vol. 426, no. 6964, pp. 264–267, 2003.

[51] T. Pittman, M. Fitch, B. Jacobs, and J. Franson, “Experimental controlled-not logic gate for single photons in the coincidence basis,” *Physical Review A*, vol. 68, no. 3, p. 032 316, 2003.

[52] S. Bartolucci, P. M. Birchall, M. Gimeno-Segovia, *et al.*, “Creation of entangled photonic states using linear optics,” *arXiv preprint arXiv:2106.13825*, 2021.

[53] A. Fyrillas, B. Bourdoncle, A. Maïnos, *et al.*, “Certified randomness in tight space,” *arXiv preprint arXiv:2301.03536*, 2023.

[54] C. Taballione, R. van der Meer, H. J. Snijders, *et al.*, “A universal fully reconfigurable 12-mode quantum photonic processor,” *Materials for Quantum Technology*, vol. 1, no. 3, p. 035 002, 2021.

[55] J. Nunn, “Orca computing: The route from nisq processors to fault-tolerance,” in *Quantum Computing, Communication, and Simulation III*, SPIE, 2023, PC124460D.

[56] W. Li, “A boost to rydberg quantum computing,” *Nature Physics*, vol. 16, no. 8, pp. 820–821, 2020.

[57] M. D. Lukin, M. Fleischhauer, R. Cote, *et al.*, “Dipole blockade and quantum information processing in mesoscopic atomic ensembles,” *Physical review letters*, vol. 87, no. 3, p. 037 901, 2001.

[58] W. Lechner, P. Hauke, and P. Zoller, “A quantum annealing architecture with all-to-all connectivity from local interactions,” *Science advances*, vol. 1, no. 9, e1500838, 2015.

[59] D. Jaksch, J. I. Cirac, P. Zoller, S. L. Rolston, R. Côté, and M. D. Lukin, “Fast quantum gates for neutral atoms,” *Physical Review Letters*, vol. 85, no. 10, p. 2208, 2000.

[60] V. Kasper, D. González-Cuadra, A. Hegde, *et al.*, “Universal quantum computation and quantum error correction with ultracold atomic mixtures,” *Quantum Science and Technology*, vol. 7, no. 1, p. 015 008, 2021.

[61] A. W. Glaetzle, R. M. van Bijnen, P. Zoller, and W. Lechner, "A coherent quantum annealer with rydberg atoms," *Nature communications*, vol. 8, no. 1, p. 15 813, 2017.

[62] P. Scholl, M. Schuler, H. J. Williams, *et al.*, "Quantum simulation of 2d antiferromagnets with hundreds of rydberg atoms," *Nature*, vol. 595, no. 7866, pp. 233–238, 2021.

[63] M. Endres, H. Bernien, A. Keesling, *et al.*, "Atom-by-atom assembly of defect-free one-dimensional cold atom arrays," *Science*, vol. 354, no. 6315, pp. 1024–1027, 2016.

[64] S. Ebadi, T. T. Wang, H. Levine, *et al.*, "Quantum phases of matter on a 256-atom programmable quantum simulator," *Nature*, vol. 595, no. 7866, pp. 227–232, 2021.

[65] J. Wurtz, P. Lopes, N. Gemelke, A. Keesling, and S. Wang, "Industry applications of neutral-atom quantum computing solving independent set problems," *arXiv preprint arXiv:2205.08500*, 2022.

[66] J. I. Cirac and P. Zoller, "Quantum computations with cold trapped ions," *Physical review letters*, vol. 74, no. 20, p. 4091, 1995.

[67] P. Wang, C.-Y. Luan, M. Qiao, *et al.*, "Single ion qubit with estimated coherence time exceeding one hour," *Nature communications*, vol. 12, no. 1, p. 233, 2021.

[68] S. Moses, C. Baldwin, M. Allman, *et al.*, "A race track trapped-ion quantum processor," *arXiv preprint arXiv:2305.03828*, 2023.

[69] S. Debnath, N. M. Linke, C. Figgatt, K. A. Landsman, K. Wright, and C. Monroe, "Demonstration of a small programmable quantum computer with atomic qubits," *Nature*, vol. 536, no. 7614, pp. 63–66, 2016.

[70] K. Wright, K. M. Beck, S. Debnath, *et al.*, "Benchmarking an 11-qubit quantum computer," *Nature communications*, vol. 10, no. 1, p. 5464, 2019.

[71] I. Pogorelov, T. Feldker, C. D. Marciniak, *et al.*, "Compact ion-trap quantum computing demonstrator," *PRX Quantum*, vol. 2, no. 2, p. 020 343, 2021.

[72] A. Y. Kitaev, "Fault-tolerant quantum computation by anyons," *Annals of physics*, vol. 303, no. 1, pp. 2–30, 2003.

[73] E. Gibney, "Inside microsoft's quest for a topological quantum computer," *Nature*, 2016.

[74] M. Iqbal, N. Tantivasadakarn, R. Verresen, *et al.*, "Creation of non-abelian topological order and anyons on a trapped-ion processor," *arXiv preprint arXiv:2305.03766*, 2023.

[75] T. Albash and D. A. Lidar, "Adiabatic quantum computation," *Reviews of Modern Physics*, vol. 90, no. 1, p. 015 002, 2018.

[76] B. Apolloni, C. Carvalho, and D. De Falco, "Quantum stochastic optimization," *Stochastic Processes and their Applications*, vol. 33, no. 2, pp. 233–244, 1989.

[77] B. Apolloni, N. Cesa-Bianchi, and D. De Falco, "A numerical implementation of "quantum annealing"," in *Stochastic Processes, Physics and Geometry: Proceedings of the Ascona-Locarno Conference*, 1990, pp. 97–111.

[78] A. Finnila, M. Gomez, C. Sebenik, C. Stenson, and J. Doll, "Quantum annealing: A new method for minimizing multidimensional functions," *Chemical physics letters*, vol. 219, no. 5-6, pp. 343–348, 1994.

[79] T. Kadowaki and H. Nishimori, "Quantum annealing in the transverse ising model," *Physical Review E*, vol. 58, no. 5, p. 5355, 1998.

[80] E. Ising, "Beitrag zur theorie des ferromagnetismus," *Zeitschrift für Physik A Hadrons and Nuclei*, vol. 31, no. 1, pp. 253–258, 1925.

[81] P. Ray, B. K. Chakrabarti, and A. Chakrabarti, "Sherrington-kirkpatrick model in a transverse field: Absence of replica symmetry breaking due to quantum fluctuations," *Physical Review B*, vol. 39, no. 16, p. 11 828, 1989.

[82] J. Brooke, D. Bitko, Rosenbaum, and G. Aeppli, "Quantum annealing of a disordered magnet," *Science*, vol. 284, no. 5415, pp. 779–781, 1999.

[83] J. Brooke, T. Rosenbaum, and G. Aeppli, "Tunable quantum tunnelling of magnetic domain walls," *Nature*, vol. 413, no. 6856, pp. 610–613, 2001.

[84] A. M. Zagoskin, "A scalable, tunable qubit, based on a clean dnd or grain boundary dd junction," *arXiv preprint cond-mat/9903170*, 1999.

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

[86] A. M. Childs, E. Farhi, and J. Preskill, "Robustness of adiabatic quantum computation," *Physical Review A*, vol. 65, no. 1, p. 012 322, 2001.

[87] W. M. Kaminsky, S. Lloyd, and T. P. Orlando, "Scalable superconducting architecture for adiabatic quantum computation," *arXiv preprint quant-ph/0403090*, 2004.

[88] F. Barahona, "On the computational complexity of ising spin glass models," *Journal of Physics A: Mathematical and General*, vol. 15, no. 10, p. 3241, 1982.

[89] M. Grajcar, A. Izmalkov, and E. Il'ichev, "Possible implementation of adiabatic quantum algorithm with superconducting flux qubits," *Physical Review B*, vol. 71, no. 14, p. 144 501, 2005.

[90] Z. Merali *et al.*, "First sale for quantum computing," *Nature*, vol. 474, no. 7349, p. 18, 2011.

[91] M. W. Johnson, M. H. Amin, S. Gildert, *et al.*, "Quantum annealing with manufactured spins," *Nature*, vol. 473, no. 7346, p. 194, 2011.

[92] S. Lloyd, *Riding D-Wave. a pioneer of quantum computing asks: Has a Canadian startup really demonstrated a prototype for a working, commercially viable quantum computer?* https://www.technologyreview.com/2008/04/22/97331/riding-d-wave/, Accessed: 20-Jun-2023, 2008.

[93] B. Altshuler, H. Krovi, and J. Roland, "Anderson localization makes adiabatic quantum optimization fail," *Proceedings of the National Academy of Sciences*, vol. 107, no. 28, pp. 12 446–12 450, 2010.

[94] E. Farhi, D. Gosset, I. Hen, *et al.*, "Performance of the quantum adiabatic algorithm on random instances of two optimization problems on regular hypergraphs," *Physical Review A*, vol. 86, no. 5, p. 052 334, 2012.

[95] S. Knysh, "Computational bottlenecks of quantum annealing," *arXiv preprint arXiv:1506.08608*, 2015.

[96] T. Lanting, A. J. Przybysz, A. Y. Smirnov, *et al.*, "Entanglement in a quantum annealing processor," *Physical Review X*, vol. 4, no. 2, p. 021 041, 2014.

[97] T. Albash, I. Hen, F. M. Spedalieri, and D. A. Lidar, "Reexamination of the evidence for entanglement in the d-wave processor," *arXiv preprint arXiv:1506.03539*, 2015.

[98] T. Albash, T. F. Rønnow, M. Troyer, and D. A. Lidar, "Reexamining classical and quantum models for the d-wave one processor: The role of excited states and ground state degeneracy," *The European Physical Journal Special Topics*, vol. 224, no. 1, pp. 111–129, 2015.

[99] S. Boixo, V. N. Smelyanskiy, A. Shabani, *et al.*, "Computational role of collective tunneling in a quantum annealer," *arXiv preprint arXiv:1411.4036*, 2014.

[100] S. Boixo, V. N. Smelyanskiy, A. Shabani, *et al.*, "Computational multi-qubit tunnelling in programmable quantum annealers," *Nature communications*, vol. 7, p. 10 327, 2016.

[101] R. Harris, Y. Sato, A. Berkley, *et al.*, "Phase transitions in a programmable quantum spin glass simulator," *Science*, vol. 361, no. 6398, pp. 162–165, 2018.

[102] L. M. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang, "Experimental realization of shor's quantum factoring algorithm using nuclear magnetic resonance," *Nature*, vol. 414, no. 6866, pp. 883–887, 2001.

[103] D. Aharonov, W. Van Dam, J. Kempe, Z. Landau, S. Lloyd, and O. Regev, "Adiabatic quantum computation is equivalent to standard quantum computation," *SIAM review*, vol. 50, no. 4, pp. 755–787, 2008.

[104] M. I. Dyakonov, "When will we have a quantum computer?" *arXiv preprint arXiv:1903.10760*, 2019.

[105] A. D. King, J. Carrasquilla, J. Raymond, *et al.*, "Observation of topological phenomena in a programmable lattice of 1,800 qubits," *Nature*, vol. 560, no. 7719, pp. 456–460, 2018.

[106] S. Boixo, T. Albash, F. M. Spedalieri, N. Chancellor, and D. A. Lidar, "Experimental signature of programmable quantum annealing," *Nature communications*, vol. 4, p. 2067, 2013.

[107] S. Boixo, T. F. Rønnow, S. V. Isakov, *et al.*, "Evidence for quantum annealing with more than one hundred qubits," *Nature physics*, vol. 10, no. 3, pp. 218–224, 2014.

[108] A. M. Zagoskin, E. Il'ichev, M. Grajcar, J. J. Betouras, and F. Nori, "How to test the "quantumness" of a quantum computer?" *Frontiers in Physics*, vol. 2, p. 33, 2014.

[109] D. Sherrington and S. Kirkpatrick, "Solvable model of a spin-glass," *Physical review letters*, vol. 35, no. 26, p. 1792, 1975.

[110] D. Bertsimas, J. Tsitsiklis, *et al.*, "Simulated annealing," *Statistical science*, vol. 8, no. 1, pp. 10–15, 1993.

[111] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, “Equation of state calculations by fast computing machines,” *The journal of chemical physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[112] B. Hajek, “Cooling schedules for optimal annealing,” *Mathematics of operations research*, vol. 13, no. 2, pp. 311–329, 1988.

[113] S. Geman and D. Geman, “Stochastic relaxation, gibbs distributions, and the bayesian restoration of images,” *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 721–741, 1984.

[114] L. Ingber, “Simulated annealing: Practice versus theory,” *Mathematical and computer modelling*, vol. 18, no. 11, pp. 29–57, 1993.

[115] S. Tanaka, R. Tamura, and B. K. Chakrabarti, *Quantum spin glasses, annealing and computation.* Cambridge University Press, 2017.

[116] S. Morita and H. Nishimori, “Mathematical foundation of quantum annealing,” *Journal of Mathematical Physics*, vol. 49, no. 12, p. 125 210, 2008.

[117] S. Morita and H. Nishimori, “Convergence of quantum annealing with real-time schrödinger dynamics,” *Journal of the Physical Society of Japan*, vol. 76, no. 6, p. 064 002, 2007.

[118] M. Born and V. Fock, “Beweis des adiabatensatzes,” *Zeitschrift für Physik*, vol. 51, no. 3-4, pp. 165–180, 1928.

[119] A. Messiah, *Quantum mechanics.* Courier Corporation, 2014.

[120] S. Suzuki and M. Okada, “Residual energies after slow quantum annealing,” *Journal of the Physical Society of Japan*, vol. 74, no. 6, pp. 1649–1652, 2005.

[121] S. L. Sondhi, S. Girvin, J. Carini, and D. Shahar, “Continuous quantum phase transitions,” *Reviews of modern physics*, vol. 69, no. 1, p. 315, 1997.

[122] A. Rajak, S. Suzuki, A. Dutta, and B. K. Chakrabarti, “Quantum annealing: An overview,” *Philosophical Transactions of the Royal Society A*, vol. 381, no. 2241, p. 20 210 417, 2023.

[123] R. D. Somma, C. D. Batista, and G. Ortiz, “Quantum approach to classical statistical mechanics,” *Physical review letters*, vol. 99, no. 3, p. 030 603, 2007.

[124] E. Seneta, *Non-negative matrices and Markov chains.* Springer Science & Business Media, 2006.

[125] J. Watrous, "Succinct quantum proofs for properties of finite groups," in *Proceedings 41st Annual Symposium on Foundations of Computer Science*, IEEE, 2000, pp. 537–546.

[126] L. Babai and S. Moran, "Arthur-merlin games: A randomized proof system, and a hierarchy of complexity classes," *Journal of Computer and System Sciences*, vol. 36, no. 2, pp. 254–276, 1988.

[127] A. Y. Kitaev, A. Shen, and M. N. Vyalyi, *Classical and quantum computation.* American Mathematical Soc., 2002.

[128] J. Kempe and O. Regev, "3-local hamiltonian is qma-complete," *arXiv preprint quant-ph/0302079*, 2003.

[129] J. Kempe, A. Kitaev, and O. Regev, "The complexity of the local hamiltonian problem," *Siam journal on computing*, vol. 35, no. 5, pp. 1070–1097, 2006.

[130] R. Oliveira and B. M. Terhal, "The complexity of quantum spin systems on a two-dimensional square lattice," *arXiv preprint quant-ph/0504050*, 2005.

[131] J. D. Biamonte and P. J. Love, "Realizable hamiltonians for universal adiabatic quantum computers," *Physical Review A*, vol. 78, no. 1, p. 012 352, 2008.

[132] I. Ozfidan, C. Deng, A. Smirnov, *et al.*, "Demonstration of a nonstoquastic hamiltonian in coupled superconducting flux qubits," *Physical Review Applied*, vol. 13, no. 3, p. 034 037, 2020.

[133] A. Das and B. K. Chakrabarti, "Colloquium: Quantum annealing and analog quantum computation," *Reviews of Modern Physics*, vol. 80, no. 3, p. 1061, 2008.

[134] A. Das and B. K. Chakrabarti, *Quantum annealing and related optimization methods.* Springer Science & Business Media, 2005, vol. 679.

[135] G. E. Santoro and E. Tosatti, "Optimization using quantum mechanics: Quantum annealing through adiabatic evolution," *Journal of Physics A: Mathematical and General*, vol. 39, no. 36, R393, 2006.

[136] R. Martoňák, G. E. Santoro, and E. Tosatti, "Quantum annealing by the path-integral monte carlo method: The two-dimensional random ising model," *Physical Review B*, vol. 66, no. 9, p. 094 203, 2002.

[137] G. E. Santoro, R. Martonák, E. Tosatti, and R. Car, "Theory of quantum annealing of an ising spin glass," *Science*, vol. 295, no. 5564, pp. 2427–2430, 2002.

[138] R. Martoňák, G. E. Santoro, and E. Tosatti, "Quantum annealing of the traveling-salesman problem," *Physical Review E*, vol. 70, no. 5, p. 057 701, 2004.

[139] D. A. Battaglia, G. E. Santoro, and E. Tosatti, "Optimization by quantum annealing: Lessons from hard satisfiability problems," *Physical Review E*, vol. 71, no. 6, p. 066 707, 2005.

[140] D. M. Ceperley, "Path integrals in the theory of condensed helium," *Reviews of Modern Physics*, vol. 67, no. 2, p. 279, 1995.

[141] Y.-H. Lee and B. Berne, "Global optimization: Quantum thermal annealing with path integral monte carlo," *The Journal of Physical Chemistry A*, vol. 104, no. 1, pp. 86–95, 2000.

[142] T. Gregor and R. Car, "Minimization of the potential energy surface of lennard–jones clusters by quantum optimization," *Chemical physics letters*, vol. 412, no. 1-3, pp. 125–130, 2005.

[143] E. Crosson and A. W. Harrow, "Simulated quantum annealing can be exponentially faster than classical simulated annealing," in *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, IEEE, 2016, pp. 714–723.

[144] E. Farhi, J. Goldstone, and S. Gutmann, "Quantum adiabatic evolution algorithms versus simulated annealing," *arXiv preprint quant-ph/0201031*, 2002.

[145] M. B. Hastings, "Obstructions to classically simulating the quantum adiabatic algorithm," *Quantum Information & Computation*, vol. 13, no. 11-12, pp. 1038–1076, 2013.

[146] M. Jarret, S. P. Jordan, and B. Lackey, "Adiabatic optimization versus diffusion monte carlo methods," *Physical Review A*, vol. 94, no. 4, p. 042 318, 2016.

[147] E. Andriyash and M. H. Amin, "Can quantum monte carlo simulate quantum annealing?" *arXiv preprint arXiv:1703.09277*, 2017.

[148] T. Albash and D. A. Lidar, "Demonstration of a scaling advantage for a quantum annealer over simulated annealing," *Physical Review X*, vol. 8, no. 3, p. 031 016, 2018.

[149] *D-Wave Leap*, https://cloud.dwavesys.com/leap/, Accessed: 23-Jun-2023.

[150] V. Dumoulin, I. J. Goodfellow, A. Courville, and Y. Bengio, "On the challenges of physical implementations of rbms," in *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[151] V. Choi, "Minor-embedding in adiabatic quantum computation: I. the parameter setting problem," *Quantum Information Processing*, vol. 7, pp. 193–209, 2008.

[152] D-Wave Systems, *Ocean SDK public Github repository*, https://github.com/dwavesystems/dwave-ocean-sdk, Accessed: 27-Sep-2023, 2023.

[153] J. Raymond, N. Ndiaye, G. Rayaprolu, and A. D. King, "Improving performance of logical qubits by parameter tuning and topology compensation," in *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*, IEEE, 2020, pp. 295–305.

[154] V. Choi, "Minor-embedding in adiabatic quantum computation: Ii. minor-universal graph design," *Quantum Information Processing*, vol. 10, no. 3, pp. 343–353, 2011.

[155] R. Diestel, *Graph theory, springer-verlag*, 2005.

[156] G. Rose, P. Bunyk, M. D. Coury, W. Macready, and V. Choi, *Systems, devices, and methods for interconnected processor topology*, US Patent 8,195,596, Jun. 2012.

[157] R. Harris, M. W. Johnson, T. Lanting, *et al.*, "Experimental investigation of an eight-qubit unit cell in a superconducting optimization processor," *Physical Review B*, vol. 82, no. 2, p. 024 511, 2010.

[158] C. Klymko, B. D. Sullivan, and T. S. Humble, "Adiabatic quantum programming: Minor embedding with hard faults," *Quantum information processing*, vol. 13, pp. 709–729, 2014.

[159] K. Boothby, P. Bunyk, J. Raymond, and A. Roy, "Next-generation topology of d-wave quantum processors," *arXiv preprint arXiv:2003.00133*, 2020.

[160] E. Pelofske, "4-clique network minor embedding for quantum annealers," *arXiv preprint arXiv:2301.08807*, 2023.

[161] K. Boothby, A. King, and J. Raymond, "Zephyr topology of d-wave quantum processors," *D-Wave Tech. Rep. Ser*, pp. 1–18, 2021.

[162] M. S. Könz, W. Lechner, H. G. Katzgraber, and M. Troyer, “Embedding overhead scaling of optimization problems in quantum annealing,” *PRX Quantum*, vol. 2, no. 4, p. 040 322, 2021.

[163] D. Willsch, M. Willsch, C. D. Gonzalez Calaza, *et al.*, “Benchmarking advantage and d-wave 2000q quantum annealers with exact cover problems,” *Quantum Information Processing*, vol. 21, no. 4, p. 141, 2022.

[164] Z. Bian, F. Chudak, R. Israel, B. Lackey, W. G. Macready, and A. Roy, “Discrete optimization using quantum annealing on sparse ising models,” *Frontiers in Physics*, vol. 2, p. 56, 2014.

[165] R. Hamerly, T. Inagaki, P. L. McMahon, *et al.*, “Experimental investigation of performance differences between coherent ising machines and a quantum annealer,” *Science advances*, vol. 5, no. 5, eaau0823, 2019.

[166] J. Marshall, A. Di Gioacchino, and E. G. Rieffel, “Perils of embedding for sampling problems,” *Physical Review Research*, vol. 2, no. 2, p. 023 020, 2020.

[167] D-Wave Systems, *D-Wave Systems 2023 Problem-solving handbook, QPU solvers: minor-embedding*, https://docs.dwavesys.com/docs/latest/handbook_embedding.html, Accessed: 19-Oct-2023, 2023.

[168] A. D. King and W. Bernoudy, “Performance benefits of increased qubit connectivity in quantum annealing 3-dimensional spin glasses,” *arXiv preprint arXiv:2009.12479*, 2020.

[169] J. Matoušek and R. Thomas, “On the complexity of finding iso-and other morphisms for partial k-trees,” *Discrete Mathematics*, vol. 108, no. 1-3, pp. 343–364, 1992.

[170] D. Eppstein, “Finding large clique minors is hard,” *arXiv:0807.0007*, 2008.

[171] J. Cai, W. G. Macready, and A. Roy, “A practical heuristic for finding graph minors,” *arXiv preprint arXiv:1406.2741*, 2014.

[172] D. Venturelli, S. Mandrà, S. Knysh, B. O’Gorman, R. Biswas, and V. Smelyanskiy, “Quantum optimization of fully connected spin glasses,” *Physical Review X*, vol. 5, no. 3, p. 031 040, 2015.

[173] M. Streif, F. Neukart, and M. Leib, “Solving quantum chemistry problems with a d-wave quantum annealer,” in *Quantum Technology and Optimization Problems: First International Workshop, QTOP 2019, Munich, Germany, March 18, 2019, Proceedings 1*, Springer, 2019, pp. 111–122.

[174] E. Valiante, M. Hernandez, A. Barzegar, and H. G. Katzgraber, “Computational overhead of locality reduction in binary optimization problems,” *Computer Physics Communications*, vol. 269, p. 108 102, 2021.

[175] N. Robertson and P. D. Seymour, “Graph minors. xiii. the disjoint paths problem,” *Journal of combinatorial theory, Series B*, vol. 63, no. 1, pp. 65–110, 1995.

[176] I. V. Hicks, “Branch decompositions and minor containment,” *Networks: An International Journal*, vol. 43, no. 1, pp. 1–9, 2004.

[177] N. Robertson and P. D. Seymour, “Graph minors. x. obstructions to tree-decomposition,” *Journal of Combinatorial Theory, Series B*, vol. 52, no. 2, pp. 153–190, 1991.

[178] I. Adler, F. Dorn, F. V. Fomin, I. Sau, and D. M. Thilikos, “Faster parameterized algorithms for minor containment,” *Theoretical Computer Science*, vol. 412, no. 50, pp. 7018–7028, 2011.

[179] D-Wave Systems, *dwavesystems/minorminer Github public repository*, https://github.com/dwavesystems/minorminer.

[180] T. Boothby, A. D. King, and A. Roy, “Fast clique minor generation in chimera qubit connectivity graphs,” *Quantum Information Processing*, vol. 15, pp. 495–508, 2016.

[181] A. Zaribafiyan, D. J. Marchand, and S. S. Changiz Rezaei, “Systematic and deterministic graph minor embedding for cartesian products of graphs,” *Quantum Information Processing*, vol. 16, no. 5, p. 136, 2017.

[182] E. G. Rieffel, D. Venturelli, B. O’Gorman, M. B. Do, E. M. Prystay, and V. N. Smelyanskiy, “A case study in programming a quantum annealer for hard operational planning problems,” *Quantum Information Processing*, vol. 14, pp. 1–36, 2015.

[183] R. Dridi and H. Alghassi, “Homology computation of large point clouds using quantum annealing,” *arXiv preprint arXiv:1512.09328*, 2015.

[184] E. Lobe, L. Schürmann, and T. Stollenwerk, “Embedding of complete graphs in broken chimera graphs,” *Quantum Information Processing*, vol. 20, no. 7, p. 234, 2021.

[185] J. P. Pinilla and S. J. Wilton, “Layout-aware embedding for quantum annealing processors,” in *High Performance Computing: 34th International Conference, ISC High Performance 2019, Frankfurt/Main, Germany, June 16–20, 2019, Proceedings 34*, Springer, 2019, pp. 121–139.

[186] S. Zbinden, A. Bärtschi, H. Djidjev, and S. Eidenbenz, "Embedding algorithms for quantum annealers with chimera and pegasus connection topologies," in *High Performance Computing: 35th International Conference, ISC High Performance 2020, Frankfurt/Main, Germany, June 22–25, 2020, Proceedings*, Springer, 2020, pp. 187–206.

[187] D.-W. Systems, "Technical description of the d-wave quantum processing unit," 2019.

[188] P. Hauke, H. G. Katzgraber, W. Lechner, H. Nishimori, and W. D. Oliver, "Perspectives of quantum annealing: Methods and implementations," *Reports on Progress in Physics*, vol. 83, no. 5, p. 054 401, 2020.

[189] P. I. Bunyk, E. M. Hoskinson, M. W. Johnson, *et al.*, "Architectural considerations in the design of a superconducting quantum annealing processor," *IEEE Transactions on Applied Superconductivity*, vol. 24, no. 4, pp. 1–10, 2014.

[190] B. D. Josephson, "Possible new effects in superconductive tunnelling," *Physics letters*, vol. 1, no. 7, pp. 251–253, 1962.

[191] B. D. Josephson, "The discovery of tunnelling supercurrents," *Reviews of Modern Physics*, vol. 46, no. 2, p. 251, 1974.

[192] P. W. Anderson and J. M. Rowell, "Probable observation of the josephson superconducting tunneling effect," *Physical Review Letters*, vol. 10, no. 6, p. 230, 1963.

[193] R. Jaklevic, J. Lambe, A. Silver, and J. Mercereau, "Quantum interference effects in josephson tunneling," *Physical Review Letters*, vol. 12, no. 7, p. 159, 1964.

[194] W. Nawrocki, "Introduction to quantum metrology," *Quantum standards and instrumentation. Springer, Heidelberg*, 2015.

[195] M. Ketchen, "Design and fabrication considerations for extending integrated dc-squids to the deep sub-micron regime," in *Superconducting Devices and Their Applications: Proceedings of the 4th International Conference SQUID'91 (Sessions on Superconducting Devices), Berlin, Fed. Rep. of Germany, June 18–21, 1991*, Springer, 1992, pp. 256–264.

[196] I. Hickman, *Analog Electronics: Analog Circuitry Explained.* Newnes, 2013.

[197] A. J. Leggett, S. Chakravarty, A. T. Dorsey, M. P. Fisher, A. Garg, and W. Zwerger, "Dynamics of the dissipative two-state system," *Reviews of Modern Physics*, vol. 59, no. 1, p. 1, 1987.

[198] R. Harris, J. Johansson, A. Berkley, *et al.*, "Experimental demonstration of a robust and scalable flux qubit," *Physical Review B*, vol. 81, no. 13, p. 134 510, 2010.

[199] D. Vion, A. Aassime, A. Cottet, *et al.*, "Manipulating the quantum state of an electrical circuit," *Science*, vol. 296, no. 5569, pp. 886–889, 2002.

[200] M. H. Amin, N. G. Dickson, and P. Smith, "Adiabatic quantum optimization with qudits," *Quantum information processing*, vol. 12, pp. 1819–1829, 2013.

[201] R. Harris, T. Lanting, A. Berkley, *et al.*, "Compound josephson-junction coupler for flux qubits with minimal crosstalk," *Physical Review B*, vol. 80, no. 5, p. 052 506, 2009.

[202] D.-W. Systems, *Ocean SDK: Operation and Timing*, https://docs.dwavesys.com/docs/latest/c_qpu_timing.html, Accessed:17-Oct-2023, 2023.

[203] S. Yarkoni, E. Raponi, T. Bäck, and S. Schmitt, "Quantum annealing for industry applications: Introduction and review," *Reports on Progress in Physics*, 2022.

[204] D-Wave Systems, *Error sources for problem representation*, https://docs.dwavesys.com/docs/latest/c_qpu_ice.html, Accessed: 06-Jul-2023, 2023.

[205] T. Albash, V. Martin-Mayor, and I. Hen, "Analog errors in ising machines," *Quantum Science and Technology*, vol. 4, no. 2, 02LT03, 2019.

[206] V. Martin-Mayor and I. Hen, "Unraveling quantum annealers using classical hardness," *Scientific reports*, vol. 5, no. 1, p. 15 324, 2015.

[207] E. Knill and R. Laflamme, "Theory of quantum error-correcting codes," *Physical Review A*, vol. 55, no. 2, p. 900, 1997.

[208] A. Pearson, A. Mishra, I. Hen, and D. A. Lidar, "Analog errors in quantum annealing: Doom and hope," *npj Quantum Information*, vol. 5, no. 1, p. 107, 2019.

[209] P. Hauke, F. M. Cucchietti, L. Tagliacozzo, I. Deutsch, and M. Lewenstein, "Can one trust quantum simulators?" *Reports on Progress in Physics*, vol. 75, no. 8, p. 082 401, 2012.

[210] N. Chancellor, P. J. Crowley, T. Durić, *et al.*, "Error measurements for a quantum annealer using the one-dimensional ising model with twisted boundaries," *npj Quantum Information*, vol. 8, no. 1, p. 73, 2022.

[211] C. Kaiser, J. Meckbach, K. Ilin, *et al.*, "Aluminum hard mask technique for the fabrication of high quality submicron nb/al–alox/nb josephson junctions," *Superconductor science and technology*, vol. 24, no. 3, p. 035 005, 2010.

[212] P. Crowley, T. Ðurić, W. Vinci, P. Warburton, and A. Green, "Quantum and classical dynamics in adiabatic computation," *Physical Review A*, vol. 90, no. 4, p. 042 317, 2014.

[213] T. Albash and D. A. Lidar, "Decoherence in adiabatic quantum computation," *Physical Review A*, vol. 91, no. 6, p. 062 320, 2015.

[214] N. G. Dickson, M. Johnson, M. Amin, *et al.*, "Thermally assisted quantum annealing of a 16-qubit problem," *Nature communications*, vol. 4, p. 1903, 2013.

[215] A. Mishra, T. Albash, and D. A. Lidar, "Finite temperature quantum annealing solving exponentially small gap problem with non-monotonic success probability," *Nature communications*, vol. 9, no. 1, p. 2917, 2018.

[216] D. S. Fisher, "Critical behavior of random transverse-field ising spin chains," *Physical review b*, vol. 51, no. 10, p. 6411, 1995.

[217] V. Choi, "The effects of the problem hamiltonian parameters on the minimum spectral gap in adiabatic quantum optimization," *Quantum Information Processing*, vol. 19, no. 3, p. 90, 2020.

[218] M. Schmidt, D. Silevitch, G. Aeppli, and T. Rosenbaum, "Using thermal boundary conditions to engineer the quantum state of a bulk magnet," *Proceedings of the National Academy of Sciences*, vol. 111, no. 10, pp. 3689–3694, 2014.

[219] M. H. Amin, "Searching for quantum speedup in quasistatic quantum annealers," *Physical Review A*, vol. 92, no. 5, p. 052 323, 2015.

[220] N. Chancellor, S. Szoke, W. Vinci, G. Aeppli, and P. A. Warburton, "Maximum-entropy inference with a programmable annealer," *Scientific reports*, vol. 6, no. 1, pp. 1–14, 2016.

[221] M. Benedetti, J. Realpe-Gómez, R. Biswas, and A. Perdomo-Ortiz, "Estimation of effective temperatures in quantum annealers for sampling applications: A case study with possible applications in deep learning," *Physical Review A*, vol. 94, no. 2, p. 022 308, 2016.

[222] E. Pelofske, G. Hahn, and H. Djidjev, "Inferring the dynamics of the state evolution during quantum annealing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 2, pp. 310–321, 2020.

[223] J. Marshall, E. G. Rieffel, and I. Hen, “Thermalization, freeze-out, and noise: Deciphering experimental quantum annealers,” *Physical Review Applied*, vol. 8, no. 6, p. 064 025, 2017.

[224] X. Qiu, P. Zoller, and X. Li, “Programmable quantum annealing architectures with ising quantum wires,” *PRX Quantum*, vol. 1, no. 2, p. 020 311, 2020.

[225] P. Hauke, L. Bonnes, M. Heyl, and W. Lechner, “Probing entanglement in adiabatic quantum optimization with trapped ions,” *Frontiers in Physics*, vol. 3, p. 21, 2015.

[226] V. Torggler, S. Krämer, and H. Ritsch, “Quantum annealing with ultracold atoms in a multimode optical resonator,” *Physical Review A*, vol. 95, no. 3, p. 032 310, 2017.

[227] T. Graß, D. Raventós, B. Juliá-Díaz, C. Gogolin, and M. Lewenstein, “Quantum annealing for the number-partitioning problem using a tunable spin glass of ions,” *Nature communications*, vol. 7, no. 1, p. 11 524, 2016.

[228] S. Mertens, “Phase transition in the number partitioning problem,” *Physical Review Letters*, vol. 81, no. 20, p. 4281, 1998.

[229] G. Pupillo, A. Micheli, M. Boninsegni, I. Lesanovsky, and P. Zoller, “Strongly correlated gases of rydberg-dressed atoms: Quantum and classical dynamics,” *Physical review letters*, vol. 104, no. 22, p. 223 002, 2010.

[230] D. Barredo, S. De Léséleuc, V. Lienhard, T. Lahaye, and A. Browaeys, “An atom-by-atom assembler of defect-free arbitrary two-dimensional atomic arrays,” *Science*, vol. 354, no. 6315, pp. 1021–1023, 2016.

[231] A. Lucas, “Ising formulations of many np problems,” *Frontiers in physics*, vol. 2, p. 5, 2014.

[232] A. Lucas, “Hard combinatorial problems and minor embeddings on lattice graphs,” *Quantum Information Processing*, vol. 18, no. 7, p. 203, 2019.

[233] M. Zaman, K. Tanahashi, and S. Tanaka, “Pyqubo: Python library for mapping combinatorial optimization problems to qubo form,” *IEEE Transactions on Computers*, vol. 71, no. 4, pp. 838–850, 2021.

[234] K. Tanahashi, S. Takayanagi, T. Motohashi, and S. Tanaka, “Application of ising machines and a software development for ising machines,” *Journal of the Physical Society of Japan*, vol. 88, no. 6, p. 061 010, 2019.

[235] I. Hen and F. M. Spedalieri, “Quantum annealing for constrained optimization,” *Physical Review Applied*, vol. 5, no. 3, p. 034 007, 2016.

[236] T. Vyskocil and H. Djidjev, “Embedding equality constraints of optimization problems into a quantum annealer,” *Algorithms*, vol. 12, no. 4, p. 77, 2019.

[237] M. Streif, M. Leib, F. Wudarski, E. Rieffel, and Z. Wang, “Quantum algorithms with local particle-number conservation: Noise effects and error correction,” *Physical Review A*, vol. 103, no. 4, p. 042 412, 2021.

[238] T. Vyskočil, S. Pakin, and H. N. Djidjev, “Embedding inequality constraints for quantum annealing optimization,” in *Quantum Technology and Optimization Problems: First International Workshop, QTOP 2019, Munich, Germany, March 18, 2019, Proceedings 1*, Springer, 2019, pp. 11–22.

[239] M. Ohzeki, “Breaking limitation of quantum annealer in solving optimization problems under constraints,” *Scientific reports*, vol. 10, no. 1, pp. 1–12, 2020.

[240] S. Yu and T. Nabil, “Applying the hubbard-stratonovich transformation to solve scheduling problems under inequality constraints with quantum annealing,” *Frontiers in Physics*, vol. 9, p. 730 685, 2021.

[241] Gurobi Optimization, LLC, *Gurobi Optimizer Reference Manual*, `https://www.gurobi.com/documentation/current/refman/index.html`, Accessed: 10-Jul-2023, 2023.

[242] D. Ottaviani and A. Amendola, “Low rank non-negative matrix factorization with d-wave 2000q,” *arXiv preprint arXiv:1808.08721*, 2018.

[243] S. Yarkoni, A. Huck, H. Schülldorf, *et al.*, “Solving the shipment rerouting problem with quantum optimization techniques,” in *Computational Logistics: 12th International Conference, ICCL 2021, Enschede, The Netherlands, September 27–29, 2021, Proceedings 12*, Springer, 2021, pp. 502–517.

[244] V. S. Denchev, N. Ding, S. Vishwanathan, and H. Neven, “Robust classification with adiabatic quantum optimization,” *arXiv preprint arXiv:1205.1148*, 2012.

[245] N. Chancellor, “Domain wall encoding of discrete variables for quantum annealing and qaoa,” *Quantum Science and Technology*, vol. 4, no. 4, p. 045 004, 2019.

[246] J. Berwald, N. Chancellor, and R. Dridi, “Understanding domain-wall encoding theoretically and experimentally,” *Philosophical Transactions of the Royal Society A*, vol. 381, no. 2241, p. 20 210 410, 2023.

[247] K. Tamura, T. Shirai, H. Katsura, S. Tanaka, and N. Togawa, “Performance comparison of typical binary-integer encodings in an ising machine,” *IEEE Access*, vol. 9, pp. 81 032–81 039, 2021.

[248] J. Chen, T. Stollenwerk, and N. Chancellor, “Performance of domain-wall encoding for quantum annealing,” *IEEE Transactions on Quantum Engineering*, vol. 2, pp. 1–14, 2021.

[249] S. Karimi and P. Ronagh, “Practical integer-to-binary mapping for quantum annealers,” *Quantum Information Processing*, vol. 18, no. 4, p. 94, 2019.

[250] I. G. Rosenberg, “Reduction of bivalent maximization to the quadratic case,” 1975.

[251] J. Biamonte, “Nonperturbative k-body to two-body commuting conversion hamiltonians and embedding problem instances into ising spins,” *Physical Review A*, vol. 77, no. 5, p. 052 331, 2008.

[252] A. Perdomo, C. Truncik, I. Tubert-Brohman, G. Rose, and A. Aspuru-Guzik, “Construction of model hamiltonians for adiabatic quantum computation and its application to finding low-energy conformations of lattice protein models,” *Physical Review A*, vol. 78, no. 1, p. 012 320, 2008.

[253] A. B. Suksmono and Y. Minato, “Finding hadamard matrices by a quantum annealing machine,” *Scientific reports*, vol. 9, no. 1, p. 14 380, 2019.

[254] T. H. Chang, T. C. Lux, and S. S. Tipirneni, “Least-squares solutions to polynomial systems of equations with quantum annealing,” *Quantum Information Processing*, vol. 18, pp. 1–17, 2019.

[255] N. Dattani, “Quadratization in discrete optimization and quantum mechanics,” *arXiv preprint arXiv:1901.04405*, 2019.

[256] E. Boros and P. L. Hammer, “Pseudo-boolean optimization,” *Discrete applied mathematics*, vol. 123, no. 1-3, pp. 155–225, 2002.

[257] F. Glover, G. Kochenberger, and Y. Du, “A tutorial on formulating and using qubo models,” *arXiv preprint arXiv:1811.11538*, 2018.

[258] S. A. Cook, “The complexity of theorem-proving procedures,” in *Proceedings of the third annual ACM symposium on Theory of computing*, 1971, pp. 151–158.

[259] M. R. Garey, D. S. Johnson, and L. Stockmeyer, “Some simplified np-complete problems,” in *Proceedings of the sixth annual ACM symposium on Theory of computing*, 1974, pp. 47–63.

[260] R. M. Karp, *Reducibility among combinatorial problems.* Springer, 2010.

[261] F. Barahona, M. Grötschel, M. Jünger, and G. Reinelt, “An application of combinatorial optimization to statistical physics and circuit layout design,” *Operations Research*, vol. 36, no. 3, pp. 493–513, 1988.

[262] G. Facchetti, G. Iacono, and C. Altafini, “Computing global structural balance in large-scale signed social networks,” *Proceedings of the National Academy of Sciences*, vol. 108, no. 52, pp. 20 953–20 958, 2011.

[263] S. G. Krantz, *The proof is in the pudding: The changing nature of mathematical proof.* Springer, 2011.

[264] R. L. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signatures and public-key cryptosystems,” *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[265] D. Boneh *et al.*, “Twenty years of attacks on the rsa cryptosystem,” *Notices of the AMS*, vol. 46, no. 2, pp. 203–213, 1999.

[266] A. K. Lenstra, H. W. Lenstra, M. S. Manasse, and J. M. Pollard, “The number field sieve,” in *The development of the number field sieve*, Springer, 1993, pp. 11–42.

[267] J. Suo, L. Wang, S. Yang, W. Zheng, and J. Zhang, “Quantum algorithms for typical hard problems: A perspective of cryptanalysis,” *Quantum Information Processing*, vol. 19, pp. 1–26, 2020.

[268] X. Peng, Z. Liao, N. Xu, *et al.*, “Quantum adiabatic algorithm for factorization and its experimental implementation,” *Physical review letters*, vol. 101, no. 22, p. 220 405, 2008.

[269] N. Xu, J. Zhu, D. Lu, X. Zhou, X. Peng, and J. Du, “Quantum factorization of 143 on a dipolar-coupling nuclear magnetic resonance system,” *Physical review letters*, vol. 108, no. 13, p. 130 501, 2012.

[270] G. Schaller and R. Schützhold, “The role of symmetries in adiabatic quantum algorithms,” *arXiv preprint arXiv:0708.1882*, 2007.

[271] R. Dridi and H. Alghassi, “Prime factorization using quantum annealing and computational algebraic geometry,” *Scientific reports*, vol. 7, no. 1, pp. 1–10, 2017.

[272] Z. Li, N. S. Dattani, X. Chen, *et al.*, "High-fidelity adiabatic quantum computation using the intrinsic hamiltonian of a spin system: Application to the experimental factorization of 291311," *arXiv preprint arXiv:1706.08061*, 2017.

[273] D. Saida, M. Hidaka, K. Imafuku, and Y. Yamanashi, "Factorization by quantum annealing using superconducting flux qubits implementing a multiplier hamiltonian," *Scientific reports*, vol. 12, no. 1, p. 13 669, 2022.

[274] S. Jiang, K. A. Britt, A. J. McCaskey, T. S. Humble, and S. Kais, "Quantum annealing for prime factorization," *Scientific reports*, vol. 8, no. 1, p. 17 667, 2018.

[275] R. Mengoni, D. Ottaviani, and P. Iorio, "Breaking rsa security with a low noise d-wave 2000q quantum annealer: Computational times, limitations and prospects," *arXiv preprint arXiv:2005.02268*, 2020.

[276] R. H. Warren, "Adapting the traveling salesman problem to an adiabatic quantum computer," *Quantum information processing*, vol. 12, pp. 1781–1785, 2013.

[277] F. Neukart, G. Compostella, C. Seidel, D. Von Dollen, S. Yarkoni, and B. Parney, "Traffic flow optimization using a quantum annealer," *Frontiers in ICT*, vol. 4, p. 29, 2017.

[278] S. Yarkoni, F. Neukart, E. M. G. Tagle, *et al.*, "Quantum shuttle: Traffic navigation with quantum computing," in *Proceedings of the 1st ACM SIGSOFT International Workshop on Architectures and Paradigms for Engineering Quantum Software*, 2020, pp. 22–30.

[279] J. Clark, T. West, J. Zammit, X. Guo, L. Mason, and D. Russell, "Towards real time multi-robot routing using quantum computing technologies," in *Proceedings of the International Conference on High Performance Computing in Asia-Pacific Region*, 2019, pp. 111–119.

[280] K. Kurowski, J. Weglarz, M. Subocz, R. Różycki, and G. Waligóra, "Hybrid quantum annealing heuristic method for solving job shop scheduling problem," in *Computational Science–ICCS 2020: 20th International Conference, Amsterdam, The Netherlands, June 3–5, 2020, Proceedings, Part VI 20*, Springer, 2020, pp. 502–515.

[281] B. Denkena, F. Schinkel, J. Pirnay, and S. Wilmsmeier, "Quantum algorithms for process parallel flexible job shop scheduling," *CIRP Journal of Manufacturing Science and Technology*, vol. 33, pp. 100–114, 2021.

[282] C. Carugno, M. Ferrari Dacrema, and P. Cremonesi, “Evaluating the job shop scheduling problem on a d-wave quantum annealer,” *Scientific Reports*, vol. 12, no. 1, p. 6539, 2022.

[283] K. Ikeda, Y. Nakamura, and T. S. Humble, “Application of quantum annealing to nurse scheduling problem,” *Scientific reports*, vol. 9, no. 1, pp. 1–10, 2019.

[284] M. Denil and N. De Freitas, “Toward the implementation of a quantum rbm,” 2011.

[285] P. M. Long and R. A. Servedio, “Restricted boltzmann machines are hard to approximately evaluate or simulate,” 2010.

[286] R. Salakhutdinov, “Learning deep boltzmann machines using adaptive mcmc,” in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 943–950.

[287] G. Desjardins, A. Courville, Y. Bengio, P. Vincent, and O. Delalleau, “Tempered markov chain monte carlo for training of restricted boltzmann machines,” in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, 2010, pp. 145–152.

[288] J. T. Rolfe, “Discrete variational autoencoders,” *arXiv:1609.02200*, 2016.

[289] P. Kairys, A. D. King, I. Ozfidan, *et al.*, “Simulating the shastry-sutherland ising model using quantum annealing,” *Prx Quantum*, vol. 1, no. 2, p. 020 320, 2020.

[290] G. E. Hinton, T. J. Sejnowski, *et al.*, “Learning and relearning in boltzmann machines,” *Parallel distributed processing: Explorations in the microstructure of cognition*, vol. 1, no. 282-317, p. 2, 1986.

[291] A. Sinclair and M. Jerrum, “Approximate counting, uniform generation and rapidly mixing markov chains,” *Information and Computation*, vol. 82, no. 1, pp. 93–133, 1989.

[292] G. F. Cooper, “The computational complexity of probabilistic inference using bayesian belief networks,” *Artificial intelligence*, vol. 42, no. 2-3, pp. 393–405, 1990.

[293] W. Vinci, K. Markström, S. Boixo, *et al.*, “Hearing the shape of the ising model with a programmable superconducting-flux annealer,” *Scientific reports*, vol. 4, no. 1, p. 5703, 2014.

[294] V. S. Denchev, S. Boixo, S. V. Isakov, *et al.*, “What is the computational value of finite-range tunneling?” *Physical Review X*, vol. 6, no. 3, p. 031 015, 2016.

[295] C. Baldwin, C. Laumann, A. Pal, and A. Scardicchio, “The many-body localized phase of the quantum random energy model,” *Physical Review B*, vol. 93, no. 2, p. 024 202, 2016.

[296] D. Korenkevych, Y. Xue, Z. Bian, *et al.*, “Benchmarking quantum hardware for training of fully visible boltzmann machines,” *arXiv preprint arXiv:1611.04528*, 2016.

[297] D-Wave Systems, *QPU-Specific Physical Properties: Advantage2_prototype1.1*, https://docs.dwavesys.com/docs/latest/_downloads/08c75269a89583c35e421c45c354 09-1275A-B_QPU_Properties_Advantage2_prototype1_1.pdf, Accessed: 19-Oct-2023, 2022.

[298] H. B. Barlow, “Unsupervised learning,” *Neural computation*, vol. 1, no. 3, pp. 295–311, 1989.

[299] Q. V. Le, “Building high-level features using large scale unsupervised learning,” in *2013 IEEE international conference on acoustics, speech and signal processing*, IEEE, 2013, pp. 8595–8598.

[300] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, “A learning algorithm for boltzmann machines,” *Cognitive science*, vol. 9, no. 1, pp. 147–169, 1985.

[301] H. J. Sussmann, “Learning algorithms for boltzmann machines,” in *Proceedings of the 27th IEEE Conference on Decision and Control*, IEEE, 1988, pp. 786–791.

[302] L. Younes, “Synchronous boltzmann machines can be universal approximators,” *Applied Mathematics Letters*, vol. 9, no. 3, pp. 109–113, 1996.

[303] L. Theis, A. v. d. Oord, and M. Bethge, “A note on the evaluation of generative models,” *arXiv preprint arXiv:1511.01844*, 2015.

[304] N. Srivastava and R. R. Salakhutdinov, “Multimodal learning with deep boltzmann machines,” in *Advances in neural information processing systems*, 2012, pp. 2222–2230.

[305] G. E. Hinton, “Training products of experts by minimizing contrastive divergence,” *Neural computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[306] R. M. Bell and Y. Koren, “Lessons from the netflix prize challenge.,” *SiGKDD Explorations*, vol. 9, no. 2, pp. 75–79, 2007.

[307] J. Yang, J. Deng, S. Li, and Y. Hao, “Improved traffic detection with support vector machine based on restricted boltzmann machine,” *Soft Computing*, vol. 21, no. 11, pp. 3101–3112, 2017.

[308] S. Rastatter, T. Moe, A. Gangopadhyay, and A. Weaver, “Abnormal traffic pattern detection in real-time financial transactions,” EasyChair, Tech. Rep., 2019.

[309] J. Carrasquilla, G. Torlai, R. G. Melko, and L. Aolita, “Reconstructing quantum states with generative models,” *Nature Machine Intelligence*, vol. 1, no. 3, p. 155, 2019.

[310] P. Smolensky, “Information processing in dynamical systems: Foundations of harmony theory,” Colorado Univ at Boulder Dept of Computer Science, Tech. Rep., 1986.

[311] S. H. Adachi and M. P. Henderson, “Application of quantum annealing to training of deep neural networks,” *arXiv preprint arXiv:1510.06356*, 2015.

[312] Y. LeCun, “The mnist database of handwritten digits,” *http://yann. lecun. com/exdb/mnist/*, 1998.

[313] M. Benedetti, J. Realpe-Gómez, R. Biswas, and A. Perdomo-Ortiz, “Quantum-assisted learning of hardware-embedded probabilistic graphical models,” *Physical Review X*, vol. 7, no. 4, p. 041 052, 2017.

[314] R. Y. Li, T. Albash, and D. A. Lidar, “Improved boltzmann machines with error corrected quantum annealing,” *arXiv preprint arXiv:1910.01283*, 2019.

[315] M. Henderson, J. Novak, and T. Cook, “Leveraging quantum annealing for election forecasting,” *Journal of the Physical Society of Japan*, vol. 88, no. 6, p. 061 009, 2019.

[316] S. Srivastava and V. Sundararaghavan, “Generative and discriminative training of boltzmann machine through quantum annealing,” *Scientific Reports*, vol. 13, no. 1, p. 7889, 2023.

[317] V. Dixit, R. Selvarajan, T. Aldwairi, *et al.*, “Training a quantum annealing based restricted boltzmann machine on cybersecurity data,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 6, no. 3, pp. 417–428, 2021.

[318] L. Moro and E. Prati, “Anomaly detection speed-up by quantum restricted boltzmann machines,” *Communications Physics*, vol. 6, no. 1, p. 269, 2023.

[319] A. Ward and N. Bambos, *Quantum annealing assisted deep learning for lung cancer detection.*

[320] J. Liu, F. M. Spedalieri, K.-T. Yao, *et al.*, "Adiabatic quantum computation applied to deep learning networks," *Entropy*, vol. 20, no. 5, p. 380, 2018.

[321] E. R. Anschuetz and C. Zanoci, "Near-term quantum-classical associative adversarial networks," *arXiv preprint arXiv:1905.13205*, 2019.

[322] M. Wilson, T. Vandal, T. Hogg, and E. G. Rieffel, "Quantum-assisted associative adversarial network: Applying quantum annealing in deep learning," *Quantum Machine Intelligence*, vol. 3, pp. 1–14, 2021.

[323] R. K. Nath, H. Thapliyal, and T. S. Humble, "A review of machine learning classification using quantum annealing for real-world applications," *SN Computer science*, vol. 2, pp. 1–11, 2021.

[324] J. Sleeman, J. Dorband, and M. Halem, "A hybrid quantum enabled rbm advantage: Convolutional autoencoders for quantum image compression and generative learning," in *Quantum Information Science, Sensing, and Computation XII*, International Society for Optics and Photonics, vol. 11391, 2020, 113910B.

[325] Y. Koshka and M. A. Novotny, "Toward sampling from undirected probabilistic graphical models using a d-wave quantum annealer," *Quantum Information Processing*, vol. 19, no. 10, pp. 1–23, 2020.

[326] S. Patel, L. Chen, P. Canoza, and S. Salahuddin, "Ising model optimization problems on a fpga accelerated restricted boltzmann machine," *arXiv preprint arXiv:2008.04436*, 2020.

[327] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the national academy of sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.

[328] I. Sutskever and T. Tieleman, "On the convergence properties of contrastive divergence," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 789–795.

[329] X. Amatriain, "Big & personal: Data and models behind netflix recommendations," in *Proceedings of the 2nd international workshop on big data, streams and heterogeneous source Mining: Algorithms, systems, programming models and applications*, ACM, 2013, pp. 1–6.

[330] M. H. Amin, E. Andriyash, J. Rolfe, B. Kulchytskyy, and R. Melko, "Quantum boltzmann machine," *Physical Review X*, vol. 8, no. 2, p. 021 050, 2018.

[331] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," *Nature*, vol. 549, no. 7671, pp. 195–202, 2017.

[332] G. E. Hinton, "A practical guide to training restricted boltzmann machines," in *Neural networks: Tricks of the trade*, Springer, 2012, pp. 599–619.

[333] N. Mohseni, P. L. McMahon, and T. Byrnes, "Ising machines as hardware solvers of combinatorial optimization problems," *Nature Reviews Physics*, vol. 4, no. 6, pp. 363–379, 2022.

[334] S. Utsunomiya, K. Takata, and Y. Yamamoto, "Mapping of ising models onto injection-locked laser systems," *Optics express*, vol. 19, no. 19, pp. 18 091–18 108, 2011.

[335] Z. Wang, A. Marandi, K. Wen, R. L. Byer, and Y. Yamamoto, "Coherent ising machine based on degenerate optical parametric oscillators," *Physical Review A*, vol. 88, no. 6, p. 063 853, 2013.

[336] P. L. McMahon, A. Marandi, Y. Haribara, *et al.*, "A fully programmable 100-spin coherent ising machine with all-to-all connections," *Science*, vol. 354, no. 6312, pp. 614–617, 2016.

[337] T. Honjo, T. Sonobe, K. Inaba, *et al.*, "100,000-spin coherent ising machine," *Science advances*, vol. 7, no. 40, eabh0952, 2021.

[338] Y. Yamamoto, K. Aihara, T. Leleu, *et al.*, "Coherent ising machines - optical neural networks operating at the quantum limit," *npj Quantum Information*, vol. 3, no. 1, p. 49, 2017.

[339] S. Tsukamoto, M. Takatsu, S. Matsubara, and H. Tamura, "An accelerator architecture for combinatorial optimization problems," *Fujitsu Sci. Tech. J*, vol. 53, no. 5, pp. 8–13, 2017.

[340] S. Matsubara, H. Tamura, M. Takatsu, *et al.*, "Ising-model optimizer with parallel-trial bit-sieve engine," in *Complex, Intelligent, and Software Intensive Systems: Proceedings of the 11th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS-2017)*, Springer, 2018, pp. 432–438.

[341] M. Aramon, G. Rosenberg, E. Valiante, T. Miyazawa, H. Tamura, and H. G. Katzgraber, “Physics-inspired optimization for quadratic unconstrained problems using a digital annealer,” *Frontiers in Physics*, vol. 7, p. 48, 2019.

[342] K. Hatakeyama-Sato, T. Kashikawa, K. Kimura, and K. Oyaizu, “Tackling the challenge of a huge materials science search space with quantum-inspired annealing,” *Advanced Intelligent Systems*, vol. 3, no. 4, p. 2 000 209, 2021.

[343] H. Goto, “Bifurcation-based adiabatic quantum computation with a nonlinear oscillator network,” *Scientific reports*, vol. 6, no. 1, p. 21 686, 2016.

[344] H. Goto, Z. Lin, and Y. Nakamura, “Boltzmann sampling from the ising model using quantum heating of coupled nonlinear oscillators,” *Scientific reports*, vol. 8, no. 1, p. 7154, 2018.

[345] H. Goto, K. Tatsumura, and A. R. Dixon, “Combinatorial optimization by simulating adiabatic bifurcations in nonlinear hamiltonian systems,” *Science advances*, vol. 5, no. 4, eaav2372, 2019.

[346] S. Ghose, A. Boroumand, J. S. Kim, J. Gómez-Luna, and O. Mutlu, “Processing-in-memory: A workload-driven perspective,” *IBM Journal of Research and Development*, vol. 63, no. 6, pp. 3–1, 2019.

[347] L. O. Chua and G. Lin, “Non-linear optimization with constraints: A cook-book approach,” *International Journal of Circuit Theory and Applications*, vol. 11, no. 2, pp. 141–159, 1983.

[348] F. L. Traversa and M. Di Ventra, “Polynomial-time solution of prime factorization and np-complete problems with digital memcomputing machines,” *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 27, no. 2, 2017.

[349] M. Di Ventra, *MemComputing: fundamentals and applications.* Oxford University Press, 2022.

[350] Y. V. Pershin and M. Di Ventra, “Memory effects in complex materials and nanoscale systems,” *Advances in Physics*, vol. 60, no. 2, pp. 145–227, 2011.

[351] M. Di Ventra and F. L. Traversa, *Self-organizing logic gates and circuits and complex problem solving with self-organizing circuits*, US Patent 9,911,080, Mar. 2018.

[352] F. L. Traversa and M. Di Ventra, “Memcomputing integer linear programming,” *arXiv preprint arXiv:1808.09999*, 2018.

[353] I. Parberry, "Parallel speedup of sequential machines: A defense of parallel computation thesis," *ACM SIGACT News*, vol. 18, no. 1, pp. 54–67, 1986.

[354] E. Gibney, "Quantum gold rush: The private funding pouring into quantum start-ups," *Nature*, vol. 574, no. 7776, pp. 22–24, Oct. 2019. DOI: 10.1038/d41586-019-02935-4. [Online]. Available: https://doi.org/10.1038/d41586-019-02935-4.

[355] M. Gitterman, "Mean first passage time for anomalous diffusion," *Phys. Rev. E*, vol. 62, no. 5, p. 6065, 2000.

[356] A. Perdomo-Ortiz, J. Fluegemann, R. Biswas, and V. N. Smelyanskiy, "A performance estimator for quantum annealers: Gauge selection and parameter setting," *arXiv preprint arXiv:1503.01083*, 2015.

[357] I. Hen, "Equation planting: A tool for benchmarking ising machines," *Physical Review Applied*, vol. 12, no. 1, p. 011 003, 2019.

[358] A. Perdomo-Ortiz, A. Feldman, A. Ozaeta, *et al.*, "Readiness of quantum optimization machines for industrial applications," *Phys. Rev. Appl.*, vol. 12, no. 1, p. 014 004, 2019.

[359] H. G. Katzgraber, F. Hamze, and R. S. Andrist, "Glassy chimeras could be blind to quantum speedup: Designing better benchmarks for quantum annealing machines," *Phys. Rev. X*, vol. 4, no. 2, p. 021 008, 2014.

[360] H. G. Katzgraber, F. Hamze, Z. Zhu, A. J. Ochoa, and H. Munoz-Bauza, "Seeking quantum speedup through spin glasses: The good, the bad, and the ugly," *Phys. Rev. X*, vol. 5, no. 3, p. 031 026, 2015.

[361] H. Nishimori, J. Tsuda, and S. Knysh, "Comparative study of the performance of quantum annealing and simulated annealing," *Physical Review E*, vol. 91, no. 1, p. 012 104, 2015.

[362] T. Zanca and G. E. Santoro, "Quantum annealing speedup over simulated annealing on random ising chains," *Physical Review B*, vol. 93, no. 22, p. 224 431, 2016.

[363] R. D. Somma, D. Nagaj, and M. Kieferová, "Quantum speedup by quantum annealing," *Physical review letters*, vol. 109, no. 5, p. 050 501, 2012.

[364] C. Baldassi and R. Zecchina, "Efficiency of quantum vs. classical annealing in nonconvex learning problems," *Proceedings of the National Academy of Sciences*, vol. 115, no. 7, pp. 1457–1462, 2018.

[365] J. King, M. Mohseni, W. Bernoudy, *et al.*, “Quantum-assisted genetic algorithm,” *arXiv preprint arXiv:1907.00707*, 2019.

[366] D. Inoue, A. Okada, T. Matsumori, K. Aihara, and H. Yoshida, “Traffic signal optimization on a square lattice with quantum annealing,” *Scientific reports*, vol. 11, no. 1, pp. 1–12, 2021.

[367] E. Pelofske, G. Hahn, D. O’Malley, H. N. Djidjev, and B. S. Alexandrov, “Quantum annealing algorithms for boolean tensor networks,” *Scientific Reports*, vol. 12, no. 1, p. 8539, 2022.

[368] A. D. King, J. Raymond, T. Lanting, *et al.*, “Quantum critical dynamics in a 5,000-qubit programmable spin glass,” *Nature*, pp. 1–6, 2023.

[369] T. F. Rønnow, Z. Wang, J. Job, *et al.*, “Defining and detecting quantum speedup,” *science*, vol. 345, no. 6195, pp. 420–424, 2014.

[370] S. Mandra, Z. Zhu, W. Wang, A. Perdomo-Ortiz, and H. G. Katzgraber, “Strengths and weaknesses of weak-strong cluster problems: A detailed overview of state-of-the-art classical heuristics versus quantum approaches,” *Physical Review A*, vol. 94, no. 2, p. 022 337, 2016.

[371] J. King, S. Yarkoni, J. Raymond, *et al.*, “Quantum annealing amid local ruggedness and global frustration,” *Journal of the Physical Society of Japan*, vol. 88, no. 6, p. 061 007, 2019.

[372] B. Suman and P. Kumar, “A survey of simulated annealing as a tool for single and multiobjective optimization,” *Journal of the operational research society*, vol. 57, pp. 1143–1160, 2006.

[373] S. V. Isakov, I. N. Zintchenko, T. F. Rønnow, and M. Troyer, “Optimised simulated annealing for ising spin glasses,” *Computer Physics Communications*, vol. 192, pp. 265–271, 2015.

[374] B. Heim, T. F. Rønnow, S. V. Isakov, and M. Troyer, “Quantum versus classical annealing of ising spin glasses,” *Science*, vol. 348, no. 6231, pp. 215–217, 2015.

[375] D. S. Steiger, T. F. Rønnow, and M. Troyer, “Heavy tails in the distribution of time to solution for classical and quantum annealing,” *Physical review letters*, vol. 115, no. 23, p. 230 501, 2015.

[376] E. Zardini, M. Rizzoli, S. Dissegna, E. Blanzieri, and D. Pastorello, “Reconstructing bayesian networks on a quantum annealer,” *arXiv preprint arXiv:2204.03526*, 2022.

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

[378] E. Farhi and A. W. Harrow, "Quantum supremacy through the quantum approximate optimization algorithm," *arXiv preprint arXiv:1602.07674*, 2016.

[379] M. Willsch, D. Willsch, F. Jin, H. De Raedt, and K. Michielsen, "Benchmarking the quantum approximate optimization algorithm," *Quantum Information Processing*, vol. 19, pp. 1–24, 2020.

[380] S. Santra, G. Quiroz, G. Ver Steeg, and D. A. Lidar, "Max 2-sat with up to 108 qubits," *New Journal of Physics*, vol. 16, no. 4, p. 045 006, 2014.

[381] M. Kowalsky, T. Albash, I. Hen, and D. A. Lidar, "3-regular three-xorsat planted solutions benchmark of classical and quantum heuristic optimizers," *Quantum Science and Technology*, vol. 7, no. 2, p. 025 008, 2022.

[382] M. Bernaschi, M. Bisson, M. Fatica, *et al.*, "How we are leading a 3-xorsat challenge: From the energy landscape to the algorithm and its efficient implementation on gpus (a)," *Europhysics Letters*, vol. 133, no. 6, p. 60 005, 2021.

[383] IBM, *What is middleware?* https://www.ibm.com/topics/middleware, Accessed: 05-Oct-2023, 2023.

[384] Multiverse, *Multiverse website*, https://multiversecomputing.com/, Accessed: 05-Oct-2023, 2023.

[385] Qbrain, *Qbrain website*, https://q-brain.io/, Accessed: 05-Oct-2023, 2023.

[386] Q-ctrl, *Q-ctrl website*, https://q-ctrl.com/, Accessed: 05-Oct-2023, 2023.

[387] E. Pelofske, G. Hahn, and H. N. Djidjev, "Parallel quantum annealing," *Scientific Reports*, vol. 12, no. 1, p. 4499, 2022.

[388] D-Wave Systems, *Qpu-specific characteristics*, https://docs.dwavesys.com/docs/latest/doc_physical_properties.html, Accessed: 06-Oct-2023, 2023.

[389] E. Barouch and B. M. McCoy, "Statistical mechanics of the xy model. iii," *Physical Review A*, vol. 3, no. 6, p. 2137, 1971.

[390] K. Sengupta, S. Powell, and S. Sachdev, "Quench dynamics across quantum critical points," *Physical Review A*, vol. 69, no. 5, p. 053 616, 2004.

[391] P. Calabrese and J. Cardy, “Evolution of entanglement entropy in one-dimensional systems,” *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2005, no. 04, P04010, 2005.

[392] P. Calabrese and J. Cardy, “Time dependence of correlation functions following a quantum quench,” *Physical review letters*, vol. 96, no. 13, p. 136 801, 2006.

[393] L. Amico, R. Fazio, A. Osterloh, and V. Vedral, “Entanglement in many-body systems,” *Reviews of modern physics*, vol. 80, no. 2, p. 517, 2008.

[394] A. Das, K. Sengupta, D. Sen, and B. K. Chakrabarti, “Infinite-range ising ferromagnet in a time-dependent transverse magnetic field: Quench and ac dynamics near the quantum critical point,” *Physical Review B*, vol. 74, no. 14, p. 144 423, 2006.

[395] K. Sengupta, D. Sen, and S. Mondal, “Exact results for quench dynamics and defect production in a two-dimensional model,” *Physical review letters*, vol. 100, no. 7, p. 077 204, 2008.

[396] T. W. Kibble, “Some implications of a cosmological phase transition,” *Physics Reports*, vol. 67, no. 1, pp. 183–199, 1980.

[397] W. H. Zurek, “Cosmological experiments in superfluid helium?” *Nature*, vol. 317, no. 6037, pp. 505–508, 1985.

[398] B. Damski, “The simplest quantum model supporting the kibble-zurek mechanism of topological defect production: Landau-zener transitions from a new perspective,” *Physical review letters*, vol. 95, no. 3, p. 035 701, 2005.

[399] J. Marshall, D. Venturelli, I. Hen, and E. G. Rieffel, “Power of pausing: Advancing understanding of thermalization in experimental quantum annealers,” *Physical Review Applied*, vol. 11, no. 4, p. 044 083, 2019.

[400] A. Perdomo-Ortiz, S. E. Venegas-Andraca, and A. Aspuru-Guzik, “A study of heuristic guesses for adiabatic quantum computation,” *Quantum Information Processing*, vol. 10, pp. 33–52, 2011.

[401] D.-W. Systems, “Reverse quantum annealing for local refinement of solutions,” 2017.

[402] V. S. Denchev, M. Mohseni, and H. Neven, *Quantum assisted optimization*, US Patent 11,449,760, Sep. 2022.

[403] D. Venturelli and A. Kondratyev, “Reverse quantum annealing approach to portfolio optimization problems,” *Quantum Machine Intelligence*, vol. 1, no. 1-2, pp. 17–30, 2019.

[404] Y. Yamashiro, M. Ohkuwa, H. Nishimori, and D. A. Lidar, “Dynamics of reverse annealing for the fully connected p-spin model,” *Physical Review A*, vol. 100, no. 5, p. 052 321, 2019.

[405] J. Golden and D. O’Malley, “Reverse annealing for nonnegative/binary matrix factorization,” *Plos one*, vol. 16, no. 1, e0244026, 2021.

[406] *QPU Solvers: Configuration*, https://docs.dwavesys.com/docs/latest/handbook_qpu.html, Accessed: 23-Jun-2023.

[407] S. Yarkoni, H. Wang, A. Plaat, and T. Bäck, “Boosting quantum annealing performance using evolution strategies for annealing offsets tuning,” in *Quantum Technology and Optimization Problems: First International Workshop, QTOP 2019, Munich, Germany, March 18, 2019, Proceedings 1*, Springer, 2019, pp. 157–168.

[408] N. Hansen, “The cma evolution strategy: A comparing review,” *Towards a new evolutionary computation: Advances in the estimation of distribution algorithms*, pp. 75–102, 2006.

[409] J. I. Adame and P. L. McMahon, “Inhomogeneous driving in quantum annealers can result in orders-of-magnitude improvements in performance,” *Quantum Science and Technology*, vol. 5, no. 3, p. 035 011, 2020.

[410] E. Andriyash, Z. Bian, F. Chudak, *et al.*, “Boosting integer factoring performance via quantum annealing offsets,” *D-Wave Technical Report Series*, vol. 14, no. 2016, 2016.

[411] T.-J. Hsu, F. Jin, C. Seidel, F. Neukart, H. De Raedt, and K. Michielsen, “Quantum annealing with anneal path control: Application to 2-sat problems with known energy landscapes,” *arXiv preprint arXiv:1810.00194*, 2018.

[412] Y. Susa, Y. Yamashiro, M. Yamamoto, and H. Nishimori, “Exponential speedup of quantum annealing by inhomogeneous driving of the transverse field,” *Journal of the Physical Society of Japan*, vol. 87, no. 2, p. 023 002, 2018.

[413] *D-Wave github repository - uniform_torque_compensation https://github.com/dwavesys system/blob/1.18.0/ dwave/embedding/chain_strength.py#L38*, 2021.

[414] Y.-L. Fang and P. Warburton, “Minimizing minor embedding energy: An application in quantum annealing,” *Quantum Information Processing*, vol. 19, no. 7, p. 191, 2020.

[415] S. Yarkoni, A. Alekseyenko, M. Streif, D. Von Dollen, F. Neukart, and T. Bäck, “Multi-car paint shop optimization with quantum annealing,” in *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)*, IEEE, 2021, pp. 35–41.

[416] A. Ceselli and M. Premoli, “On good encodings for quantum annealer and digital optimization solvers,” *Scientific Reports*, vol. 13, no. 1, p. 5628, 2023.

[417] K. E. Hamilton and T. S. Humble, “Identifying the minor set cover of dense connected bipartite graphs via random matching edge sets,” *Quantum Information Processing*, vol. 16, pp. 1–17, 2017.

[418] G. Bass, M. Henderson, J. Heath, and J. Dulny, “Optimizing the optimizer: Decomposition techniques for quantum annealing,” *Quantum Machine Intelligence*, vol. 3, pp. 1–14, 2021.

[419] C. Wang, H. Chen, and E. Jonckheere, “Quantum versus simulated annealing in wireless interference network optimization,” *Scientific reports*, vol. 6, no. 1, p. 25 797, 2016.

[420] H. Karimi and G. Rosenberg, “Boosting quantum annealer performance via sample persistence,” *Quantum Information Processing*, vol. 16, no. 7, p. 166, 2017.

[421] P. Chardaire, J. L. Lutton, and A. Sutter, “Thermostatistical persistency: A powerful improving concept for simulated annealing algorithms,” *European Journal of Operational Research*, vol. 86, no. 3, pp. 565–579, 1995.

[422] K. Yonaga, M. J. Miyama, and M. Ohzeki, “Solving inequality-constrained binary optimization problems on quantum annealer,” *arXiv:2012.06119*, 2020.

[423] A. Perdomo-Ortiz, J. Fluegemann, S. Narasimhan, R. Biswas, and V. N. Smelyanskiy, “A quantum annealing approach for fault detection and diagnosis of graph-based systems,” *The European Physical Journal Special Topics*, vol. 224, pp. 131–148, 2015.

[424] J. Raymond, S. Yarkoni, and E. Andriyash, “Global warming: Temperature estimation in annealers,” *Frontiers in ICT*, vol. 3, p. 23, 2016.

[425] D.-W. Systems, *Ocean SDK: num_spin_reversal_transforms*, https://docs.dwavesys.com/docs/latest/c_solver_parameters.html#param-num-srt, Accessed:17-Oct-2023, 2023.

[426] M. J. Wainwright, M. I. Jordan, *et al.*, "Graphical models, exponential families, and variational inference," *Foundations and Trends® in Machine Learning*, vol. 1, no. 1–2, pp. 1–305, 2008.

[427] D.-W. Systems, *Ocean SDK: Source code for dwave.embedding.chain_breaks*, https://docs.ocean.dwavesys.com/projects/system/en/stable/_modules/dwave/embedding/chain_breaks.html, Accessed:17-Oct-2023, 2023.

[428] J. Roffe, "Quantum error correction: An introductory guide," *Contemporary Physics*, vol. 60, no. 3, pp. 226–245, 2019.

[429] K. L. Pudenz, T. Albash, and D. A. Lidar, "Error-corrected quantum annealing with hundreds of qubits," *Nature communications*, vol. 5, no. 1, p. 3243, 2014.

[430] W. Vinci, T. Albash, G. Paz-Silva, I. Hen, and D. A. Lidar, "Quantum annealing correction with minor embedding," *Physical Review A*, vol. 92, no. 4, p. 042 310, 2015.

[431] K. L. Pudenz, T. Albash, and D. A. Lidar, "Quantum annealing correction for random ising problems," *Physical Review A*, vol. 91, no. 4, p. 042 302, 2015.

[432] A. Callison and N. Chancellor, "Hybrid quantum-classical algorithms in the noisy intermediate-scale quantum era and beyond," *Physical Review A*, vol. 106, no. 1, p. 010 101, 2022.

[433] A. A. Abbott, C. S. Calude, M. J. Dinneen, and R. Hua, "A hybrid quantum-classical paradigm to mitigate embedding costs in quantum annealing," *International Journal of Quantum Information*, vol. 17, no. 05, p. 1 950 042, 2019.

[434] N. Chancellor, "Modernizing quantum annealing using local searches," *New Journal of Physics*, vol. 19, no. 2, p. 023 024, 2017.

[435] H. Alghassi, R. Dridi, and S. Tayur, "Graver bases via quantum annealing with application to non-linear integer programs," *arXiv preprint arXiv:1902.04215*, 2019.

[436] T. Graß and M. Lewenstein, "Hybrid annealing: Coupling a quantum simulator to a classical computer," *Physical Review A*, vol. 95, no. 5, p. 052 309, 2017.

[437] D. Pastorello, E. Blanzieri, and V. Cavecchia, "Learning adiabatic quantum algorithms over optimization problems," *Quantum Machine Intelligence*, vol. 3, pp. 1–19, 2021.

[438] D. Pastorello and E. Blanzieri, "Quantum annealing learning search for solving qubo problems," *Quantum Information Processing*, vol. 18, pp. 1–17, 2019.

[439] A. Mizel, D. A. Lidar, and M. Mitchell, "Simple proof of equivalence between adiabatic quantum computation and the circuit model," *Physical review letters*, vol. 99, no. 7, p. 070 502, 2007.

[440] Z. Liu, S. Li, and Y. Ge, "A parallel algorithm based on quantum annealing and double-elite spiral search for mixed-integer optimal control problems in engineering," *Applied Soft Computing*, vol. 124, p. 109 018, 2022.

[441] M. Booth, S. P. Reinhardt, and A. Roy, "Partitioning optimization problems for hybrid classical," *quantum execution. Technical Report*, pp. 01–09, 2017.

[442] D-Wave Systems, *Ocean SDK: Qbsolv documentation*, https://docs.ocean.dwavesys.com/projects/qbsolv/en/latest/, Accessed: 12-Oct-2023, 2021.

[443] F. Glover, "Laguna. m., 1997, tabu search," *Modern Heuristic Techniques for Combinatorial Problems. Blackwell Scientific Publishing, Oxford*, pp. 70–150, 1993.

[444] S. Okada, M. Ohzeki, M. Terabe, and S. Taguchi, "Improving solutions by embedding larger subproblems in a d-wave quantum annealer," *Scientific reports*, vol. 9, no. 1, p. 2098, 2019.

[445] H. N. Djidjev, G. Hahn, S. M. Mniszewski, C. F. Negre, A. M. Niklasson, and V. B. Sardeshmukh, "Graph partitioning methods for fast parallel quantum molecular dynamics," in *2016 Proceedings of the Seventh SIAM Workshop on Combinatorial Scientific Computing*, SIAM, 2016, pp. 42–51.

[446] G. Chapuis, H. Djidjev, G. Hahn, and G. Rizk, "Finding maximum cliques on the d-wave quantum annealer," *Journal of Signal Processing Systems*, vol. 91, pp. 363–377, 2019.

[447] G. Bass, C. Tomlin, V. Kumar, P. Rihaczek, and J. Dulny, "Heterogeneous quantum computing for satellite constellation optimization: Solving the weighted k-clique problem," *Quantum Science and Technology*, vol. 3, no. 2, p. 024 010, 2018.

[448] K. De Jong, "Learning with genetic algorithms: An overview," *Machine learning*, vol. 3, pp. 121–138, 1988.

[449] A. Hagberg, P. Swart, and D. S Chult, "Exploring network structure, dynamics, and function using networkx," Los Alamos National Lab.(LANL), Los Alamos, NM (United States), Tech. Rep., 2008.

[450] S. Okada, M. Ohzeki, and S. Taguchi, "Efficient partition of integer optimization problems with one-hot encoding," *Scientific reports*, vol. 9, no. 1, p. 13 036, 2019.

[451] K. Ender, R. ter Hoeven, B. E. Niehoff, M. Drieb-Schön, and W. Lechner, "Parity quantum optimization: Compiler," *Quantum*, vol. 7, p. 950, 2023.

[452] E. Cartlidge, "Rewriting the quantum-computer blueprint.," *Nature*, 2023.

[453] NEC Corporation, *NEC develops the world's first unit cell facilitating scaling up to a fully-connected quantum annealing architecture*, https://www.nec.com/en/press/202203/global_20220317_01.html, Accessed: 16-Oct-2023, 2022.

[454] M. Drieb-Schön, K. Ender, Y. Javanmard, and W. Lechner, "Parity quantum optimization: Encoding constraints," *Quantum*, vol. 7, p. 951, 2023.

[455] M. Marvian, D. A. Lidar, and I. Hen, "On the computational complexity of curing non-stoquastic hamiltonians," *Nature communications*, vol. 10, no. 1, p. 1571, 2019.

[456] W. K. Mutlag, S. K. Ali, Z. M. Aydam, and B. H. Taher, "Feature extraction methods: A review," in *Journal of Physics: Conference Series*, IOP Publishing, vol. 1591, 2020, p. 012 028.

[457] R. Zebari, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed, "A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction," *Journal of Applied Science and Technology Trends*, vol. 1, no. 2, pp. 56–70, 2020.

[458] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proceedings. 1991 IEEE computer society conference on computer vision and pattern recognition*, IEEE Computer Society, 1991, pp. 586–587.

[459] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[460] R. Bro and A. K. Smilde, "Principal component analysis," *Analytical methods*, vol. 6, no. 9, pp. 2812–2831, 2014.

[461] C. O. S. Sorzano, J. Vargas, and A. P. Montano, "A survey of dimensionality reduction techniques," *arXiv preprint arXiv:1403.2877*, 2014.

[462] A. Hyvärinen and E. Oja, "Independent component analysis: Algorithms and applications," *Neural networks*, vol. 13, no. 4-5, pp. 411–430, 2000.

[463] O. Déniz, M. Castrillon, and M. Hernández, "Face recognition using independent component analysis and support vector machines," *Pattern recognition letters*, vol. 24, no. 13, pp. 2153–2157, 2003.

[464] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[465] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, Springer, 2014, pp. 818–833.

[466] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[467] K. Mato, R. Mengoni, D. Ottaviani, and G. Palermo, "Quantum molecular unfolding," *Quantum Science and Technology*, vol. 7, no. 3, p. 035 020, 2022.

[468] C. F. Negre, H. Ushijima-Mwesigwa, and S. M. Mniszewski, "Detecting multiple communities using quantum annealing on the d-wave system," *Plos one*, vol. 15, no. 2, e0227538, 2020.

[469] D. O'Malley, V. V. Vesselinov, B. S. Alexandrov, and L. B. Alexandrov, "Nonnegative/binary matrix factorization with a d-wave quantum annealer," *PloS one*, vol. 13, no. 12, e0206653, 2018.

[470] R. Y. Li, R. Di Felice, R. Rohs, and D. A. Lidar, "Quantum annealing versus classical machine learning applied to a simplified computational biology problem," *NPJ quantum information*, vol. 4, no. 1, p. 14, 2018.

[471] D. Vert, R. Sirdey, and S. Louise, “On the limitations of the chimera graph topology in using analog quantum computers,” in *Proceedings of the 16th ACM international conference on computing frontiers*, 2019, pp. 226–229.

[472] D-Wave Systems Inc., *Qpu-specific physical properties: Advantage2_prototype1.1*, 2022.

[473] L. Buffoni and M. Campisi, “Thermodynamics of a quantum annealer,” *Quantum Sci. Technol.*, vol. 5, no. 3, p. 035 013, 2020.

[474] Gurobi Optimization, LLC, *Gurobi Optimizer Reference Manual*, https://www.gurobi.com, Accessed: 29-Oct-2023, 2021.

[475] F. Mémoli, “Gromov–wasserstein distances and the metric approach to object matching,” *Found. Comput. Math.*, vol. 11, no. 4, pp. 417–487, 2011.

[476] G. Peyré, M. Cuturi, and J. Solomon, “Gromov-wasserstein averaging of kernel and distance matrices,” in *International Conference on Machine Learning*, PMLR, 2016, pp. 2664–2672.

[477] E. L. Lawler, “The quadratic assignment problem,” *Manage. Sci.*, vol. 9, no. 4, pp. 586–599, 1963.

[478] L. Mitten, “Branch-and-bound methods: General formulation and properties,” *Oper. Res.*, vol. 18, no. 1, pp. 24–34, 1970.

[479] F. L. Traversa and M. Di Ventra, “Universal memcomputing machines,” *IEEE Trans. Neural. Netw. Learn. Syst.*, vol. 26, no. 11, pp. 2702–2715, 2015.

[480] V. Dixit, R. Selvarajan, M. A. Alam, T. S. Humble, and S. Kais, “Training and classification using a restricted boltzmann machine on the d-wave 2000q,” *arXiv preprint arXiv:2005.03247*, 2020.

[481] D. J. MacKay and D. J. Mac Kay, *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[482] V. Dixit, R. Selvarajan, M. A. Alam, T. S. Humble, and S. Kais, “Training restricted boltzmann machines with a d-wave quantum annealer,” *Frontiers in Physics*, vol. 9, p. 589 626, 2021.

[483] M. Kālis, A. Locāns, R. Šikovs, H. Naseri, and A. Ambainis, “A hybrid quantum-classical approach for inference on restricted boltzmann machines,” *arXiv preprint arXiv:2304.12418*, 2023.

[484] J. Ariño Bernad and A. Teixidó Bonfill, *Exploration of boltzmann machines via bars and stripes*, 2019. [Online]. Available: http://hdl.handle.net/2117/187271.

[485] L. Ambrosio, L. A. Caffarelli, Y. Brenier, G. Buttazzo, C. Villani, and S. Salsa, *Optimal Transportation and Applications*. Springer Berlin Heidelberg, 2003. DOI: 10.1007/b12016. [Online]. Available: https://doi.org/10.1007/b12016.

[486] F. Mémoli, "Spectral gromov-wasserstein distances for shape matching," in *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, IEEE, 2009, pp. 256–263.

[487] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*, PMLR, 2017, pp. 214–223.

[488] C. Bunne, D. Alvarez-Melis, A. Krause, and S. Jegelka, "Learning generative models across incomparable spaces," in *International Conference on Machine Learning*, PMLR, 2019, pp. 851–861.

[489] J. Solomon, G. Peyré, V. G. Kim, and S. Sra, "Entropic metric alignment for correspondence problems," *ACM Transactions on Graphics (ToG)*, vol. 35, no. 4, pp. 1–13, 2016.

[490] M. Scetbon, G. Peyré, and M. Cuturi, "Linear-time gromov wasserstein distances using low rank couplings and costs," in *International Conference on Machine Learning*, PMLR, 2022, pp. 19 347–19 365.

[491] T. Vayer, R. Flamary, R. Tavenard, L. Chapel, and N. Courty, "Sliced gromov-wasserstein," in *Neural Information Processing Systems*, 2019.

[492] R. E. Burkard, E. Cela, P. M. Pardalos, and L. S. Pitsoulis, "The quadratic assignment problem," in *Handbook of combinatorial optimization*, Springer, 1998, pp. 1713–1809.

[493] M. T. Fiala Timlin and W. R. Pulleyblank, "Precedence constrained routing and helicopter scheduling: Heuristic design," *Interfaces*, vol. 22, no. 3, pp. 100–111, 1992.

[494] J.-F. Cordeau and G. Laporte, "The dial-a-ride problem (darp): Variants, modeling issues and algorithms," *Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, vol. 1, pp. 89–101, 2003.

[495] R. W. Calvo and A. Colorni, "An effective and fast heuristic for the dial-a-ride problem," *4or*, vol. 5, no. 1, pp. 61–73, 2007.

[496] M. W. Savelsbergh, "Local search in routing problems with time windows," *Annals of Operations research*, vol. 4, no. 1, pp. 285–305, 1985.

[497] R. de Alvarenga Rosa, A. Manhães Machado, G. Mattos Ribeiro, and G. Regis Mauri, "A mathematical model and a clustering search metaheuristic for planning the helicopter transportation of employees to the production platforms of oil and gas," *Comput. Ind. Eng.*, vol. 101, pp. 303–312, 2016, ISSN: 0360-8352. DOI: https://doi.org/10.1016/j.cie.2016.09.006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360835216303382.

[498] A. Motta, R. Vieira, and J. Soletti, "Optimal routing offshore helicopter using genetic algorithm," in *2011 6th IEEE Joint International Information Technology and Artificial Intelligence Conference*, vol. 2, 2011, pp. 6–9.

[499] A. Husseinzadeh Kashan, A. Abbasi-Pooya, and S. Karimiyan, "A rig-based formulation and a league championship algorithm for helicopter routing in offshore transportation," in *Proceedings of the 2nd International Conference on Data Engineering and Communication Technology*, A. J. Kulkarni, S. C. Satapathy, T. Kang, and A. H. Kashan, Eds., Springer Singapore, 2019, pp. 23–38.