Quantum Annealing Explained

Table of Contents

1. Introduction

Note

Readers must have sufficient knowledge of mathematics, (quantum) physics, electronics, classical computing (hardware and software) and numerical optimisation methods.

1.1. Adiabatic theorem

The adiabatic theorem is a concept in quantum mechanics. Its original form, due to the German-British physicist Max Born and the Russian physicist Vladimir Aleksandrovich Fock, was stated as follows: A physical system remains in its instantaneous eigenstate (Box 1.1) if a given perturbation is acting on it slowly enough and if there is a gap between the eigenvalue and the rest of the Hamiltonian's spectrum (Box 1.2).

An eigenstate is the measured state of some object possessing quantifiable characteristics such as position, momentum, etc. (the word "eigenstate" is derived from the German word "eigen", meaning "inherent" or “characteristic"). The state being measured and described must be observable (i.e. something such as position or momentum that can be experimentally measured either directly or indirectly), and must have a definite value, called an eigenvalue. In the everyday world, it is natural and intuitive to think of every object being in its own eigenstate; this is just another way of saying that every object appears to have a definite position, a definite momentum, a definite measured value and a definite time of occurrence. However, in quantum mechanics, Heisenberg’s uncertainty principle, named after the German theoretical physicist Werner Karl Heisenberg, implies that it is impossible to measure the exact value for the momentum of a particle, given that its position has been determined at a given instant and likewise, it is impossible to determine the exact location of that particle once its momentum has been determined at a particular instant. Therefore, it becomes necessary to formulate clearly the difference between the state of something that is uncertain and the state of something having a definite value. When an object can definitely be "pinned down" in some respect, it is said to possess an eigenstate.

Box 1.1: Eigenstate

The Hamiltonian of a quantum system, named after the Irish mathematician and physicist William Rowan Hamilton, is an operator corresponding to the total energy of that system, including both kinetic energy and potential energy. Its spectrum, the system's energy spectrum or its set of energy eigenvalues, is the set of possible outcomes obtainable from a measurement of the system's total energy.

Box 1.2: Hamiltonian

1.2. Adiabatic theorem-based problem solving

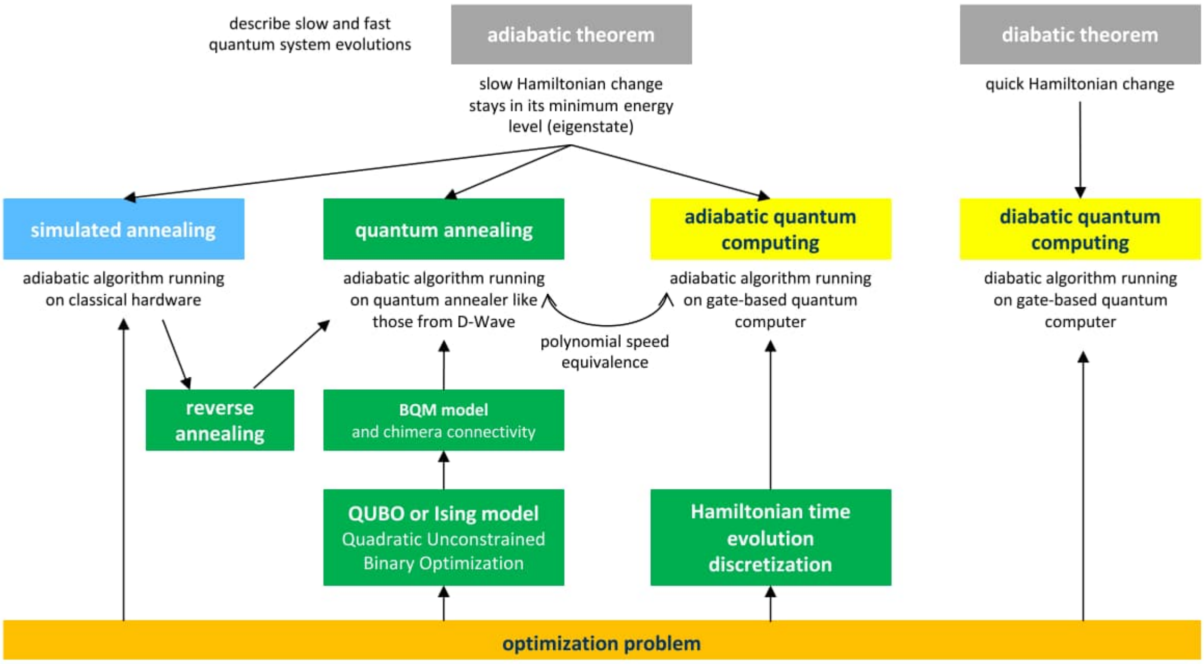


Figure 1.1: Adiabatic theorem-based problem solving (source: Olivier Ezratty 2024)

An adiabatic process consists of gradually changing conditions, allowing the quantum system to adapt its configuration, hence the spatial probability density is modified by the process. If the quantum system starts in an eigenstate of the initial Hamiltonian, it will end in the corresponding eigenstate of the final Hamiltonian.

Box 1.3: Adiabatic process

A diabatic process consists of rapidly changing conditions, preventing the quantum system from adapting its configuration during the process, hence the spatial probability density remains unchanged. Typically there is no eigenstate of the final Hamiltonian with the same functional form as the initial state. The quantum system ends in a linear combination of states that sum to reproduce the initial probability density.

Box 1.4: Diabatic process

Reverse annealing uses classical simulated annealing to find a trivial solution which is then transferred to quantum annealing to find better solutions.

Box 1.5: Reverse annealing

Note

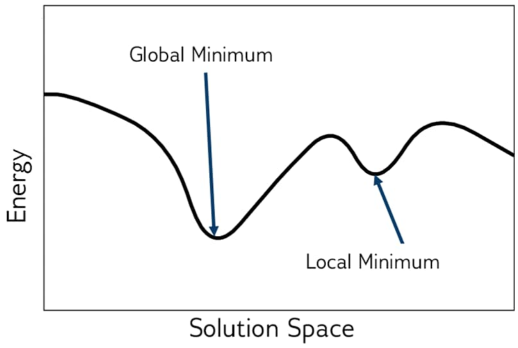
Annealing is a heat treatment that alters the physical and sometimes chemical properties of a material to increase its ductility and reduce its hardness, making it more workable. In the case of ferrous metals such as steel, annealing is performed by heating the material (generally until glowing) for a while and then slowly letting it cool to room temperature in still air. In this fashion, the metal is softened and prepared for further work such as shaping, stamping or forming.

Optimisation problems can be solved in three different ways based on the adiabatic theorem:

1. **Quantum annealing with a Quantum Annealer (QA)**

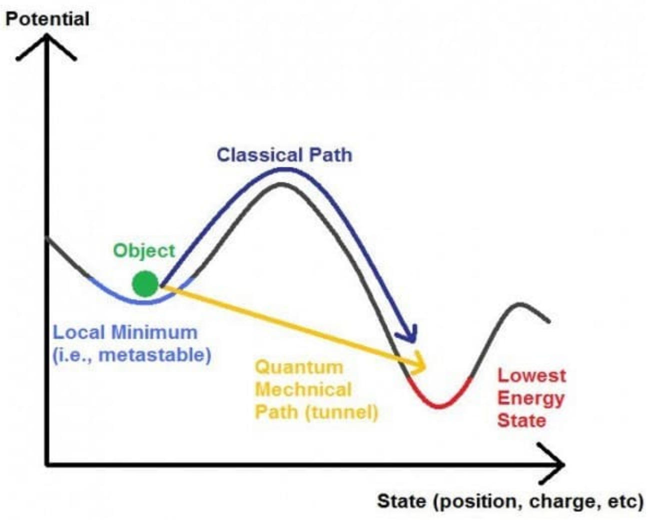
   Quantum annealing is based on an optimisation process for finding the global minimum of an objective function (Box 1.6) using a slow Hamiltonian making use of quantum tunnelling (Box 1.7); see § 2.2 for details.

An objective function is either a cost function (aka loss function) or a profit function (aka reward function), which an optimisation problem seeks to minimise (cost function) or maximise (profit function).



Box 1.6: Objective function

Quantum tunnelling is a quantum mechanical phenomenon in which a particle passes through a potential energy barrier that, according to classical mechanics, the particle does not have sufficient energy to enter or surmount. Quantum tunnelling is a consequence of the wave nature of matter, where wave equations such as the Schrödinger wave equation (named after the Austrian and naturalised Irish physicist Erwin Rudolf Josef Alexander Schrödinger) describe the behaviour of a particle (see § 2.1). The probability of transmission of a particle wave packet through a barrier decreases exponentially with the barrier height, the barrier width and the particle's mass. Therefore, tunnelling is seen most prominently in low-mass particles such as electrons or protons tunnelling through microscopically narrow barriers.

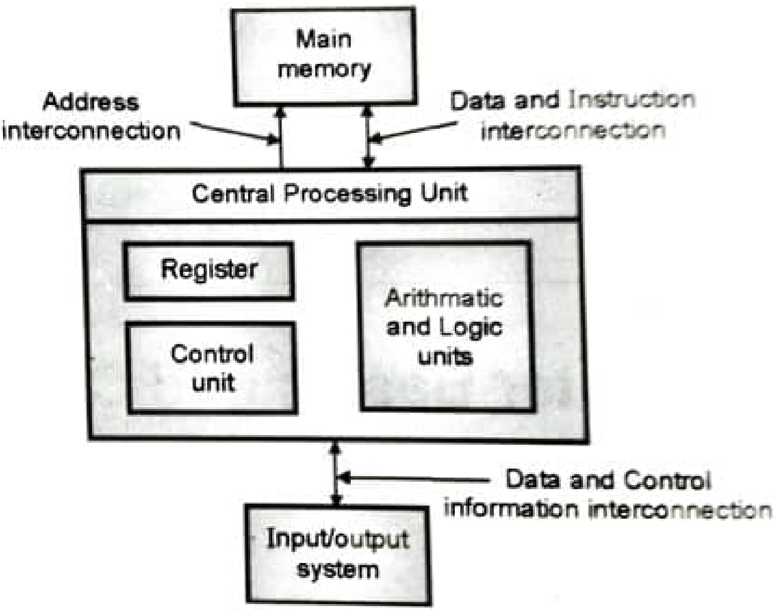


Box 1.7: Quantum tunnelling

2. **Simulated annealing with a classical Digital Annealer (DA)**

A DA is a dedicated digital computing system that use a non-Von Neumann architecture (Box 1.8) to minimise data movement in solving combinatorial optimisation problems. Such a system is composed of thousands of bit-updating blocks with on-chip memory that stores weights and biases, logic blocks to perform bit flips, and interfacing and control circuitry. Rather than programming the DA, a problem is uploaded in the form of weight matrices and bias vectors so as to convert the problem into an "energy landscape". Problem solving with a DA is very similar to problem solving with a QA.

The Von Neumann architecture (named after the American mathematician, physicist, computer scientist and engineer John von Neumann) is a computer architecture based on a 1945 description by Von Neumann and others. It describes a design architecture for a digital computer system with the following components: a Central Processing Unit (CPU) that contains an Arithmetic Logic Unit (ALU) and processor registers, a Control Unit (CU) that contains an Instruction Register (IR) and Program Counter (PC), memory that stores data and instructions, external mass storage and input and output mechanisms.



Box 1.8: Von Neumann computer architecture

3. **Adiabatic quantum algorithm on a universal gate-based quantum computer**

A universal gate-based quantum computer is a device that takes input data and transforms this input data according to a quantum circuit specification. A quantum circuit specifies a set of qubits and the sequence of operations to be performed on these qubits, i.e. preparation of qubits, quantum gate operations on the qubits and qubit measurements.

In classical computing the information is encoded in bits, where each bit can have the value zero or one. In quantum computing the information is encoded in qubits. A qubit is a two-level quantum system where the two basis qubit states are usually written as $|0\rangle$ and $|1\rangle$. A qubit can be in state $|0\rangle$, state $|1\rangle$ or (unlike a classical bit) in a linear combination (superposition) of both states $\alpha|0\rangle + \beta|1\rangle$. The amplitudes α and β are complex numbers that correspond with the probabilities that their measured value is either “0” or “1”. The trick in devising an algorithm for a quantum computer is to choreograph a pattern of constructive and destructive interference for its qubits, so that for each wrong answer the contributions to these qubit amplitudes cancel each other out, whereas for the right answer the contributions reinforce each other.

1.3. History and current status of quantum annealing and related technologies

The idea to implement quantum annealing using the quantum tunnelling effect originated in 1988 and 1989 in Italy and Germany. It was then perfected in Japan in 1998 with the introduction of quantum fluctuations into the simulated annealing process of optimisation problems, aiming at faster convergence to the optimal state.

A year later, D-Wave Quantum Systems Inc. was established in British Columbia (Canada), as an offshoot from the University of British Columbia (UBC). It funded academic research in quantum computing, thus building a collaborative network of research scientists from several universities and institutions, including UBC (Canada), IPHT Jena (Germany), Université de Sherbrooke (Canada), University of Toronto (Canada), University of Twente (Netherlands), Chalmers University of Technology (Sweden), University of Erlangen (Germany) and NASA’s Jet Propulsion Laboratory (JPL).

On February 13, 2007, D-Wave Quantum Systems demonstrated a prototype 16-qubit quantum annealing processor called Orion.

On May 11, 2011, D-Wave Quantum Systems announced D-Wave One, described as "the world's first commercially available quantum computer", operating on a 128-qubit chipset. The oldest D-Wave Quantum Systems publicised case study came from Google and NASA, using a D-Wave One QA to solve an optimisation and combinatorial problem in a graph.

In 2013, Google and NASA set up a joint quantum computing laboratory, named Quantum Artificial Intelligence Lab (QuAIL), and they experimented with D-Wave QAs.

In December 2015, Google and NASA announced that the D-Wave 2X QA outperformed both simulated annealing and QMC (Box 1.9) by a factor up to 100,000,000 on a set of hard optimisation problems (Figure 1.2). This was without any doubt the first oversold quantum advantage (Box 1.10) claim.

Quantum Monte Carlo (QMC) encompasses a large family of computational methods whose common aim is the study of complex quantum systems. One of the major goals of these approaches is to provide a reliable solution (or an accurate approximation) of the quantum many-body problem. The diverse flavours of QMC approaches all share the common use of the Monte Carlo method to handle the multi-dimensional integrals that arise in the different formulations of the many-body problem. The Monte Carlo method refers to a broad class of computational algorithms that rely on repeated random sampling to obtain

numerical results. The underlying concept is to use randomness to solve problems that might be deterministic in principle. The name comes from the Monte Carlo Casino in Monaco, where the primary developer of the method, the Polish-American mathematician, nuclear physicist and computer scientist Stanisław Ulam (who participated in the Manhattan Project), was inspired by his uncle's gambling habits.

Box 1.9: Quantum Monte Carlo (QMC)

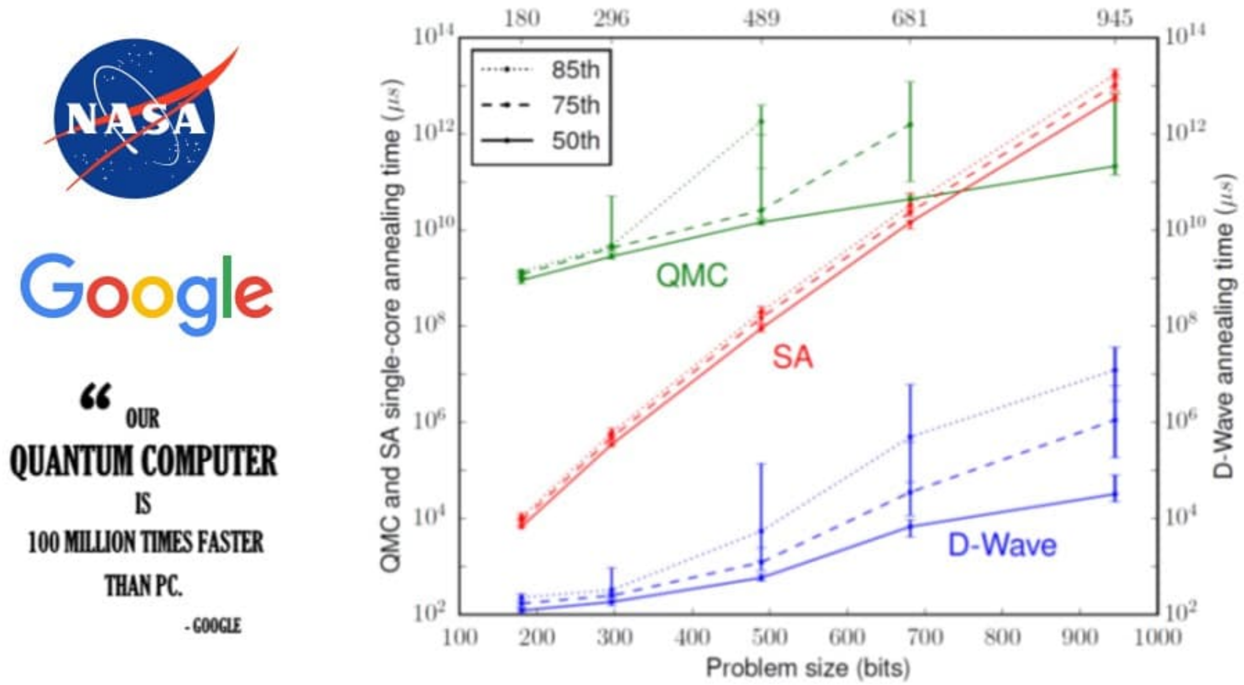


Figure 1.2: Google and NASA 2015 announcement (source: V.S. Denchev et al.)

Quantum advantage is the goal of demonstrating that a quantum computer can solve a practical problem that no classical computer can solve in any feasible amount of time. Conceptually, quantum advantage involves both the engineering task of building a powerful quantum computer and the computational complexity-theoretic task of finding a problem that can be solved by that quantum computer and has a more than polynomial speedup over the best known or possible classical algorithm for that task.

Box 1.10: Quantum advantage

Quantum annealing was explored in 2016 by the IARPA agency in its Quantum-Enhanced Optimization (QEO) project, which aimed to create an adiabatic computer void of some of the limitations from D-Wave QAs, particularly in terms of connectivity and quality of qubits. This project was folded into DARPA's Quantum Annealing Feasibility Study (QAFS) project in February 2020, which produced an experimental 25-qubit QA system.

Stanford University has been working on quantum annealing for many years. In 2016, they created a prototype photonic based QA with 100 qubits having an all-to-all connectivity (10,000 connections). This research is still going on and involves NTT in Japan.

The European Annealing-based Variational Quantum processors (AVaQus) project, launched in October 2020, brings together five research laboratories: Institut de Física d'Altes Energies of Barcelona in Spain, Karlsruhe Institut für Technologie (KIT) in Germany, CNRS Institut Néel in France, the University of Glasgow in the UK and the Consejo Superior de Investigaciones Científicas in Madrid (Spain) and it is associated with three start-ups: Delft Circuits (Netherlands), Qilimanjaro (Spain) and HQS (Germany). The project obtained funding independently of the EU Quantum Flagship program.

The Taiwanese National Science and Technology Council (NSTC) provides funding for the development of Compal Electronics' GPU Annealer system which is driven by NVIDIA CUDA-Q Solvers.

D-Wave Quantum Systems is currently the only commercial manufacturer of QA systems (see § 3.1). An undisclosed number of their QA systems is deployed on-premises (mostly by very large organisations), but the vast majority of its customers use cloud-based quantum computing services provided by D-Wave Quantum Systems, either directly or through Amazon Marketplace. Most customers are currently either experimenting with quantum annealing or else conducting proof-of-concept projects, only a few of them use QAs in their production environments.

Qilimanjaro benefits from European funding through AVaQus and is developing a QA based on superconducting flux qubits (see § 3.2).

NEC is developing a QA based on superconducting parametrons (see § 3.3).

NTT developed a Coherent Ising Machine (CIM) system based on optical technology (see § 3.4).

NEC (see 3.3) , Fujitsu (see § 3.5), Hitachi (see § 3.6) and Toshiba (see § 3.7) developed quantum-inspired annealing systems that are based on classical digital technology.

Sharp is developing a Simulated Quantum Annealer (SQA) for controlling Automatic Guided Vehicles (AGVs); see § 3.8.

<u>Notes</u>

1. Theoretically, quantum algorithms for universal gate-based quantum computers can be converted to algorithms for QAs (and vice versa), with polynomial time overhead (which can be quite substantial). However, many more qubits will be required for the QA quantum algorithm than for the universal gate-based quantum computer algorithm (dependent on the quantum algorithm type).

2. The American theoretical physicist John Phillip Preskill believes that the QA architecture is theoretically not as scalable as the universal gate-based quantum computer architecture hence there is no convincing theoretical basis for the advantage of quantum annealing. As of today, there has been no convincing proof that quantum annealing is capable of outperforming the best classical solutions to optimisation problems in terms of speed, but there may be other valid reasons to implement quantum annealing

solutions, such as for example lower Total Cost of Ownership (TCO), environmentally friendlier solution[1], etc.

3. While quantum advantage obtained with quantum annealing experiments is routinely reported, such claims lack definite proof. Researchers infer that they have achieved quantum advantage, but they cannot prove that this superiority is over any competing classical solution.

4. In March 2025 D-Wave Quantum Systems claimed having achieved quantum advantage in solving a useful real-world complex optimisation problem, outperforming state-of-the-art classical methods. According to D-Wave Quantum Systems, a D-Wave QA performed magnetic materials simulation in just a few minutes while it would have taken more than one million years on a classical supercomputer built with GPU clusters (and would have consumed more than than the world's annual electricity production).

[1] For example, D-Wave Quantum Systems developed and demonstrated a prototype Proof-of-Quantum Work (PoQW) blockchain mechanism that replaces classical Proof-of-Work (PoW) mechanisms. PoQW enables generation and validation of blockchain hashes by means of QAs, thus eliminating reliance on power-hungry classical mining based on ASICs and GPUs (a reduction of three orders of magnitude has been observed during the demonstration).

2. Quantum annealing basics

2.1. Quantum-mechanical evolution of quantum objects

According to the Copenhagen interpretation of quantum mechanics, the Schrödinger wave equation is the best possible description of a quantum system.

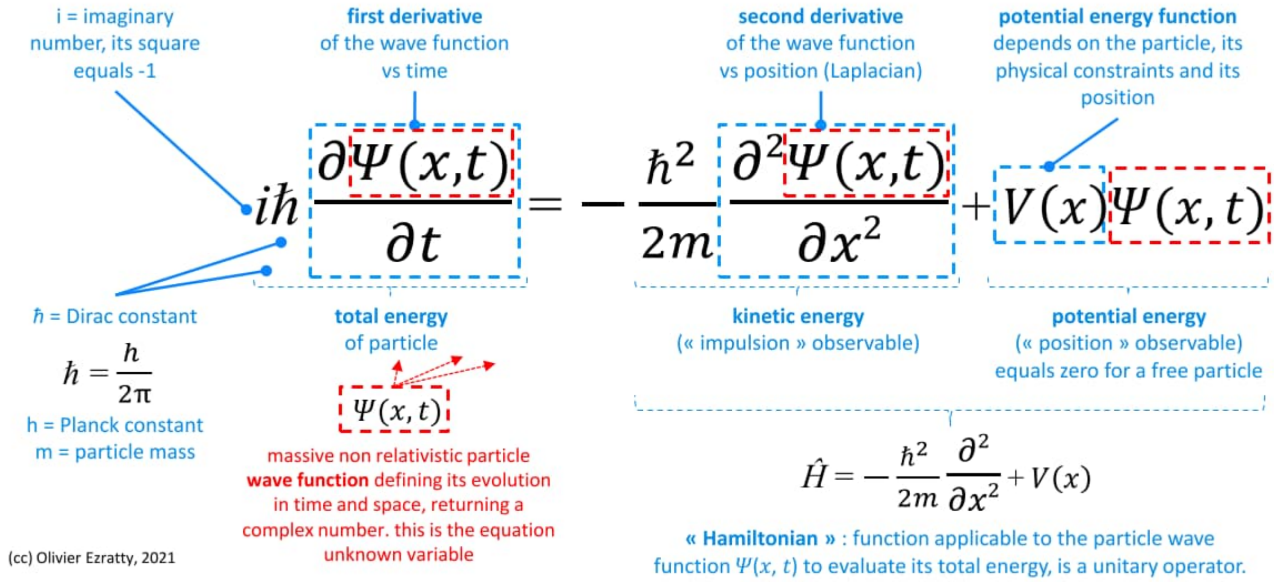


Figure 2.1: Schrödinger wave equation (source: Olivier Ezratty 2021)

This equation describes the quantum-mechanical evolution of a massive non-relativistic quantum object as a wave function, giving the probability of finding the quantum object at a particular position in space at a given time. “Massive” refers to quantum objects that have mass (unlike for example photon particles which are massless). “Non-relativistic” refers to quantum objects whose kinetic energy is smaller than twice their rest mass energy as defined by the famous equation $E=mc^2$ of the Swiss-American theoretical physicist Albert Einstein. This implies that the speed of these quantum objects is not close to the speed of light which is 299,792,458 metres per second in a vacuum (denoted by c from the Latin celeritas).

Schrödinger’s equation is a partial differential equation, i.e. it connects its components via derivative functions, in this case of first degree (a slope on a curve) and of second degree (an acceleration). The quantum object’s wave function appears three times in the equation: to the left of the equation with a first derivative on the time of the wave function, to the right with a second derivative on its position and with a simple multiplication with the function V(x).

The unknown in Schrödinger’s equation is the wave function of the quantum object ψ(x,t) which describes its probabilistic behaviour in space and time (x indicates the position of the quantum object in space, with one, two or three dimensions depending on its constraints, and t is the time). This function returns a complex number that encodes the wave’s amplitude and phase.

The wave function's square is equal to the probability of finding the quantum object at location x at time t. The sum of the probabilities of finding the quantum object somewhere is of course equal to 1; this is called the normalisation constraint.

The ψ(x,t) function must be a continuous function and "filled" everywhere in space. Its value is bounded by 0 and 1. It is a single value, even in the case of superposition. In that case, the ψ(x,t) is a linear superposition of two ψ functions and is itself a ψ function (because a quantum superposition is just another wave function).

The operator that acts on the right side of the Schrödinger equation and accumulates kinetic and potential energy function is the Hamiltonian, which describes the total energy of the system. Its spectrum, the quantum system's energy spectrum or its set of energy eigenvalues, is the set of possible outcomes obtainable from a measurement of the quantum system's total energy. This Hamiltonian plays an important role in the quantum annealing process (see § 2.2).

The potential energy of the quantum object is defined by the function V(x) which depends only on the quantum object's position in space and its physical constraints. When a quantum object is free and moves without constraints, this function returns zero.

The Schrödinger equation is linear over time, which means that any combination of solutions of the equation becomes a new solution of the equation. This makes it possible to decompose a wave function into several elementary wave functions that are called the "eigenstates" of the quantum object. They correspond to the different energy levels of the quantum object that are discrete when it is constrained in space. The equation's linearity has a lot of consequences, like for example superposition and entanglement.

Any nanoscopic, microscopic or macroscopic massive non-relativistic object (all the way to the entire universe) has a Schrödinger wave function but the equation only makes practical sense for nanoscopic objects as it can only be analytically solved in a limited number of simple cases (e.g. for the electron of an hydrogen atom, for a free particle, for a particle in a potential well or box, or for a quantum harmonic oscillator). In more complex cases, the resolution of the equation requires non-analytical methods, raw calculation and simulation (Box 2.1). It is one of the fields of application of quantum simulation to solve the Schrödinger equation in cases where analytical methods are not available.

A quantum simulator is an analogue quantum computer that is capable of simulating quantum objects and solving related problems, particularly in materials physics. These are the quantum computers that the American theoretical physicist Richard Phillips Feynman had in mind when he introduced the term "quantum computer" in 1981. Quantum simulators should not be confused with quantum emulators which are classical computer systems capable of emulating quantum circuits.

Box 2.1: Quantum simulator

The Schrödinger equation is <u>not</u> applicable to massive relativistic quantum objects, i.e. particles that have mass and velocities near light speed and very high energies, or particles which are massless, e.g. photons (Box 2.2).

The photon is an elementary subatomic particle. It is the quantum of the electromagnetic field, including electromagnetic radiation such as light and radio waves, and it is the force carrier for the electromagnetic force. Photons do not have electrical charge, they have zero mass and zero rest energy, and they only exist as moving particles. The speed of photons in a medium depends upon the medium and is always slower than c.

Box 2.2: Photon

The time evolution of massive relativistic quantum objects is described by the Dirac equation, named after the British theoretical physicist Paul Adrien Maurice Dirac, and Klein-Gordon equation, named after the Swedish theoretical physicist Oskar Benjamin Klein and the German theoretical physicist Walter Gordon.

The time evolution of photons is described by Maxwell's equations, named after the Scottish physicist and mathematician James Clerk Maxwell, and their various derivations.

2.2. Quantum annealing process

The problem to be solved with quantum annealing is converted into either an Ising problem (Box 2.3) or a Quadratic Unconstrained Binary Optimization (QUBO) problem (Box 2.4). The Ising or QUBO problem formulation is then translated into a Binary Quadratic Model (BQM) formulation which defines an objective function with binary variables, a quadratic component and linear constraints. BQM problems are NP hard problems (Box 2.5).

The Ising model aka Lenz-Ising model, named after the German physicists Ernst Ising and Wilhelm Lenz, is a mathematical model of ferromagnetism in statistical mechanics. The model consists of discrete variables that represent magnetic dipole moments of atomic spins that can be in one of two states (+1 or −1). The spins are arranged in a graph, usually a lattice (where the local structure repeats periodically in all directions), allowing each spin to interact with its neighbours. Neighbouring spins that agree have a lower energy than those that disagree. The system tends to the lowest energy but heat disturbs this tendency, thus creating the possibility of different structural phases. The model allows the identification of phase transitions as a simplified model of reality.

Box 2.3: Ising model

Quadratic Unconstrained Binary Optimization (QUBO) is a combinatorial optimisation problem with a wide range of applications. Moreover, due to its close connection to the Ising model, QUBO constitutes a central problem class for Adiabatic Quantum Computing (AQC), where it is solved through quantum annealing.

Box 2.4: Quadratic Unconstrained Binary Optimization (QUBO)

The Polynomial (P) versus Nondeterministic-Polynomial (NP) problem asks whether every problem whose solution can be quickly verified can also be solved quickly. The informal term “quickly” means the existence of an algorithm solving the task that runs in polynomial time, such that the time to complete the task varies as a polynomial function on the size of the input to the algorithm which solves the problem instance. The class of questions for which some algorithm can provide an answer in polynomial time is P. For some questions, there is no known way to find an answer quickly, but if one is provided with information showing what the answer is, it is possible to verify the answer quickly. The class of questions for which an answer can be verified in polynomial time is NP.

Box 2.5: P versus NP

Figure 2.2 and the paragraphs that follow describe quantum annealing based on the Ising problem formulation, where the binary variables σ_i of the objective function are represented by physical Ising spins, $J_{ij}\,\sigma_i\,\sigma_j$ are elements of the quadratic components and $h_i\,\sigma_i$ are linear constraints.

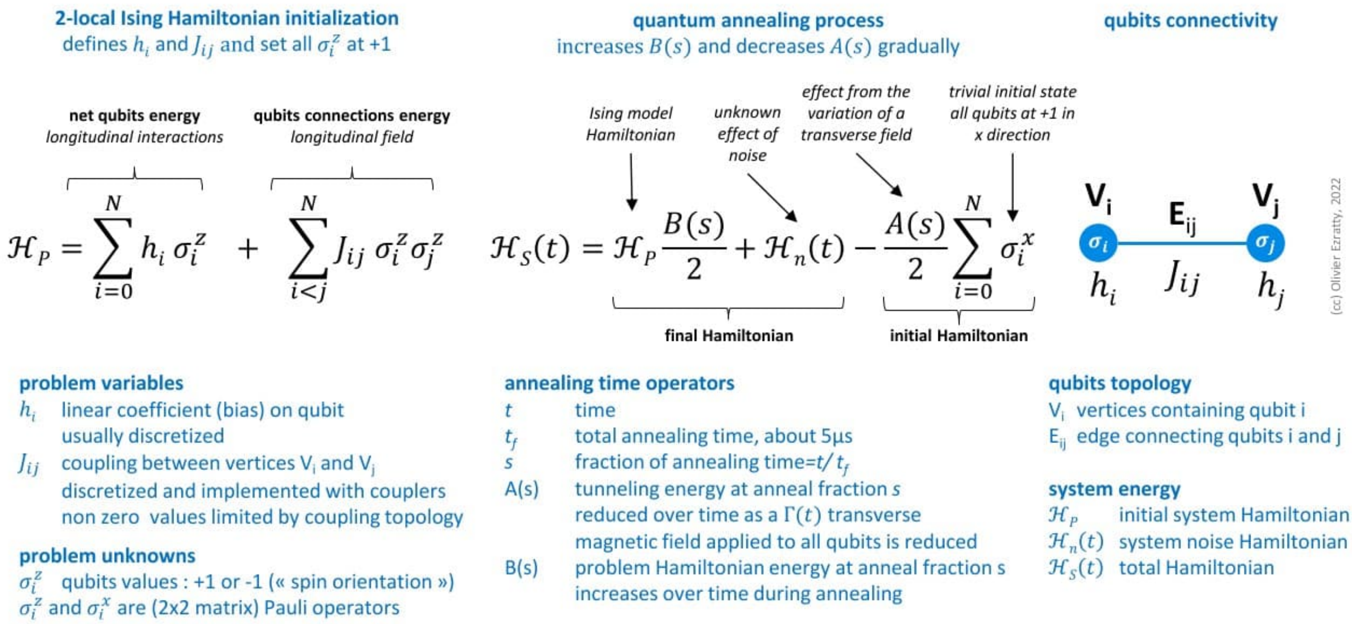


Figure 2.2: Quantum annealing - Ising problem formulation (source: Olivier Ezratty 2022)

Note

In the QUBO problem formulation, the N binary variables of the objective function are represented by an upper-diagonal matrix, where diagonal terms are the linear coefficients, and the nonzero off-diagonal terms are the quadratic coefficients.

The quantum annealing process starts with assigning linear coefficients aka biases (h_i in Figure 2.2) to a set of interconnected qubits (with values σ_i) on the QA system. This corresponds to setting the absolute qubit energy on each qubit as a linear coefficient (bias).

The links between the qubits are assigned weights (J_{ij} in Figure 2.2) that are defined by the qubit couplers. This corresponds to setting the relative qubit connection energy in the longitudinal field (z) for each pair of qubits.

Note

With current QA technologies, the values of h_i and J_{ij} are discretised by Digital-to-Analogue Converters (DACs); see for example § 4.4. These DACs introduce significant sampling noise due to their low sampling rate with typically only a couple of hundred different steps. Consequently, the precision of the data of the problem to be solved is rather low (and certainly far from high-precision floating-point used in scientific computation).

The QA system is then initialised with setting the qubits at $|+\rangle$, which is a perfect superposition between $|0\rangle$ and $|1\rangle$, corresponding to the lowest-energy state of the system, i.e. the tunnelling Hamiltonian. A perfect superposition between the qubit's $|0\rangle$ and $|1\rangle$ basis states means that there is an equal probability of measuring 0 and 1.

After initialisation of the QA system, a transverse magnetic field is applied to the set of qubits. It is then progressively reduced down to zero, which, as a result of quantum tunnelling through peaks, will drive the QA system to an equilibrium state that corresponds to a minimum energy level.

The strength of the transverse magnetic field determines the quantum-mechanical probability to change the amplitudes of all quantum states in parallel. The rate of change of the transverse magnetic field is slow enough so that the QA system stays close to the ground state of the instantaneous Hamiltonian.

The quantum annealing process takes place with controlled evolutions of the tunnelling Hamiltonian energy A(s), also called transverse energy, and the problem Hamiltonian energy B(s), with tuning of the transverse magnetic field affecting the QA's qubit chipset. In the equations in Figure 2.2, it means reducing the value of A(s) and increasing the value of B(s) accordingly (usually not linearly), see Figure 2.3. This leads to automatically modifying the quantum states of the qubits (spin up or down in the z direction) towards a result that corresponds to the solution of the submitted problem (when the QA system is expected to have reached the ground state of the Ising model).

The qubits are then read (aka measured) and this generates a value of +1 or a value of -1 for each of them depending on their quantum state. The result that is obtained is inherently probabilistic, and not just because noise gets involved with an unknown time-evolving Hamiltonian $\mathcal{H}_n(t)$. Hence the whole process is iterative with several quantum annealing passes and their results being averaged. The accuracy of the result obtained result improves with the number of passes executed.

Note
There are variations in the implementation of this process with regards to the qubit coupling mechanism. It can have only one degree of freedom (z) as for the D-Wave QAs (see § 4.3), or two and three degrees of freedom (x, y and z) as for the QAs that Qilimanjaro is planning to develop (see § 3.2).

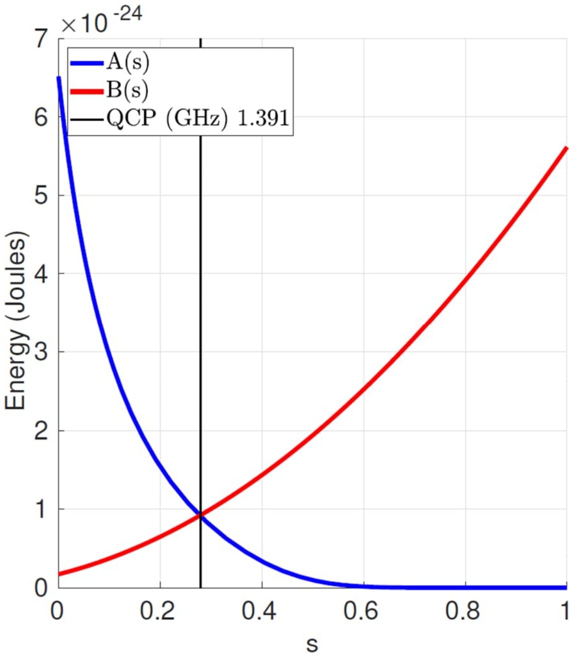


Figure 2.3: Energy states evolution example (source: D-Wave Quantum Systems 2024)

3. Quantum annealing, related technologies and manufacturers

3.1. D-Wave Quantum Systems Inc.

D-Wave Quantum Systems (Canada) has advanced quantum annealing technology steadily with the first three generations of QA prototypes created between 2007 and 2009 and four generations of commercial QAs, starting with the D-Wave One in 2012 (128 pairwise-coupled qubits), followed by the D Wave 2000Q in 2017 (2,048 qubits, each qubit being coupled to 6 other qubits), the D-Wave Advantage with the Pegasus chipset launched in 2020 (5,640 qubits, each qubit being coupled to 15 other qubits), and the D-Wave Advantage 2 (with 4,400 qubits and 20-way qubit connectivity) in 2025.

D-Wave Quantum Systems' Leap quantum computing cloud service (Figure 3.1) provides real-time access to D-Wave 2000Q and Advantage QA platforms and to the Hybrid Solver Service (HSS). D-Wave QAs and HSS can also be accessed via the AWS Marketplace.

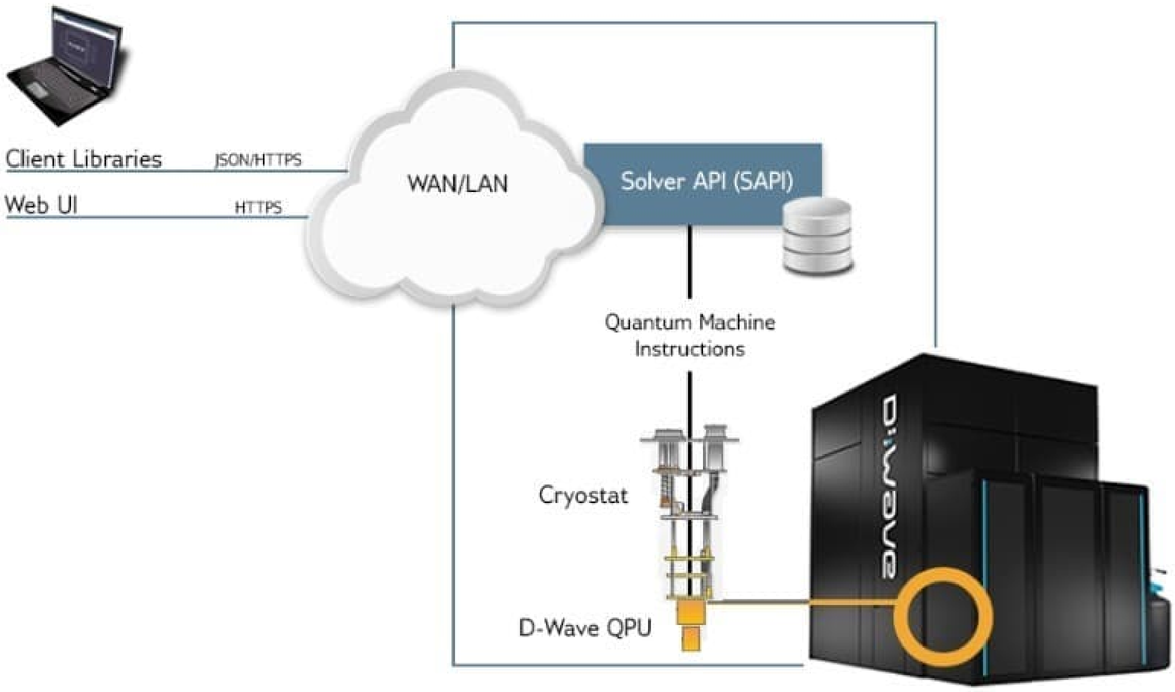


Figure 3.1: QA quantum computing environment (source: D-Wave Quantum Systems 2024)

Figure 3.2 shows a simplified diagram of the sequence of steps, the dark red set of arrows, to execute a quantum job on a D-Wave QA, starting and ending on a user's client system. Each quantum job consists of a single input together with parameters. Quantum jobs are sent across a network to the Solver API (SAPI) server and join a queue. Each queued quantum job is assigned to one of possibly multiple workers, which can run in parallel. A worker prepares the quantum job for the Quantum Processing Unit (QPU) and postprocessing, sends the quantum job to the QPU queue, receives samples (results) and post-processes them (overlapping in time with QPU execution), and bundles the samples with additional quantum job execution information for return to the client system.

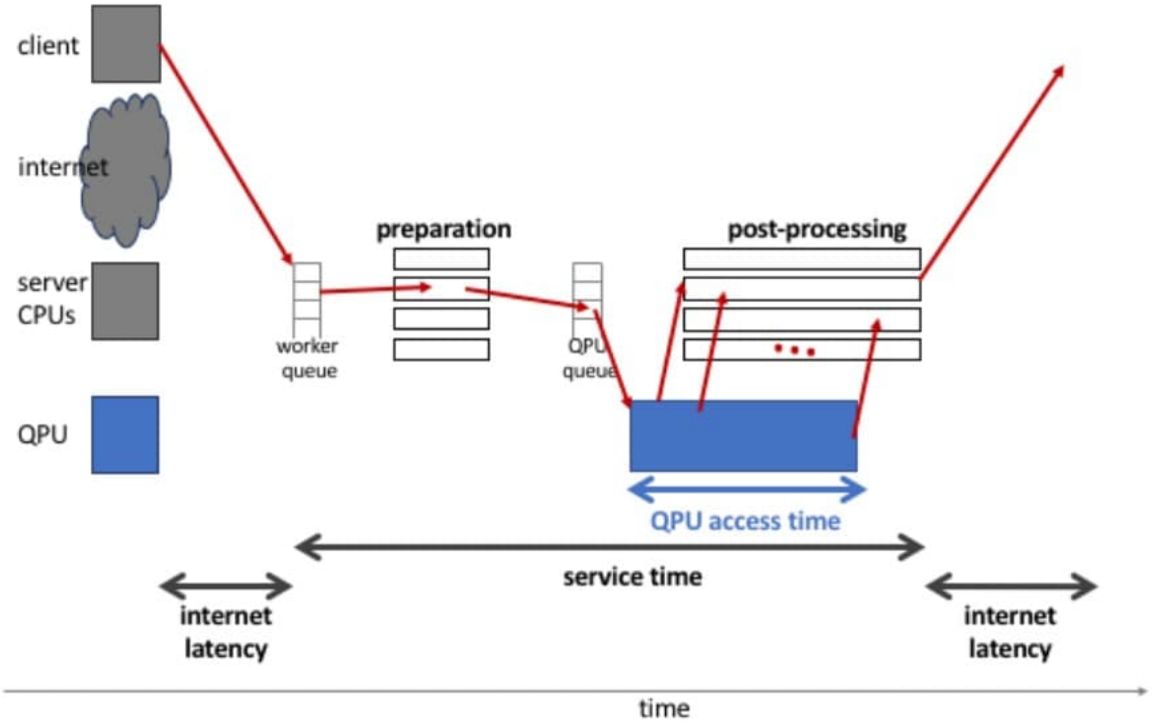


Figure 3.2: Quantum job execution (source: D-Wave Quantum Systems 2024)

Notes

1. The QPU executes one quantum job at a time (this execution time is known as the quantum job's QPU access time), during which the QPU is unavailable to any other quantum job.

2. The total time for a quantum job to pass through the D-Wave QA system is the service time. However, the execution time for a quantum job as observed by a client includes service time and internet latency.

3. Server-side postprocessing is limited to computing the energies of returned samples. D-Wave Quantum Systems' Ocean software (see § 5.1) provides additional client-side postprocessing tools (more complex postprocessing can provide performance benefits at low timing cost).

According to D-Wave Quantum Systems, their QAs could provide quantum speedup for many problem types for which universal gate-based quantum computers provide quantum speedup, but this has not been proven yet on large scale problems.

D-Wave quantum annealing hardware is described in Chapter 4 and D-Wave quantum annealing software is described in Chapter 5.

3.2. Qilimanjaro Quantum Tech SL

Qilimanjaro Quantum (Spain) is a start-up from the Barcelona Supercomputing Center (IFAE) and the University of Barcelona. Qilimanjaro Quantum develops a QA based on superconducting flux qubits. Their differentiation compared to QAs from D-Wave Quantum Systems is better qubit coherence and better qubit connectivity.

Qilimanjaro Quantum benefits from European funding through the Annealing-based Variational Quantum processors (AVaQus) project. It involves the superconducting team at Institut Néel in Grenoble (France) who designs the microwave amplifiers used in flux qubits readouts.

Qilimanjaro Quantum is also developing Qibo, an open-source quantum middleware platform. Qibo (Figure 3.3) features an open-source full-stack API for quantum emulation and quantum hardware control. Qibo is the cloud operating service to run software batches on the future Qilimanjaro Quantum QA, classical quantum emulators and gate-based quantum computers, with a design pattern to create classical/quantum hybrid algorithms. Qilimanjaro Quantum also plans to sell their future QA systems to customers willing to use them on-premises.

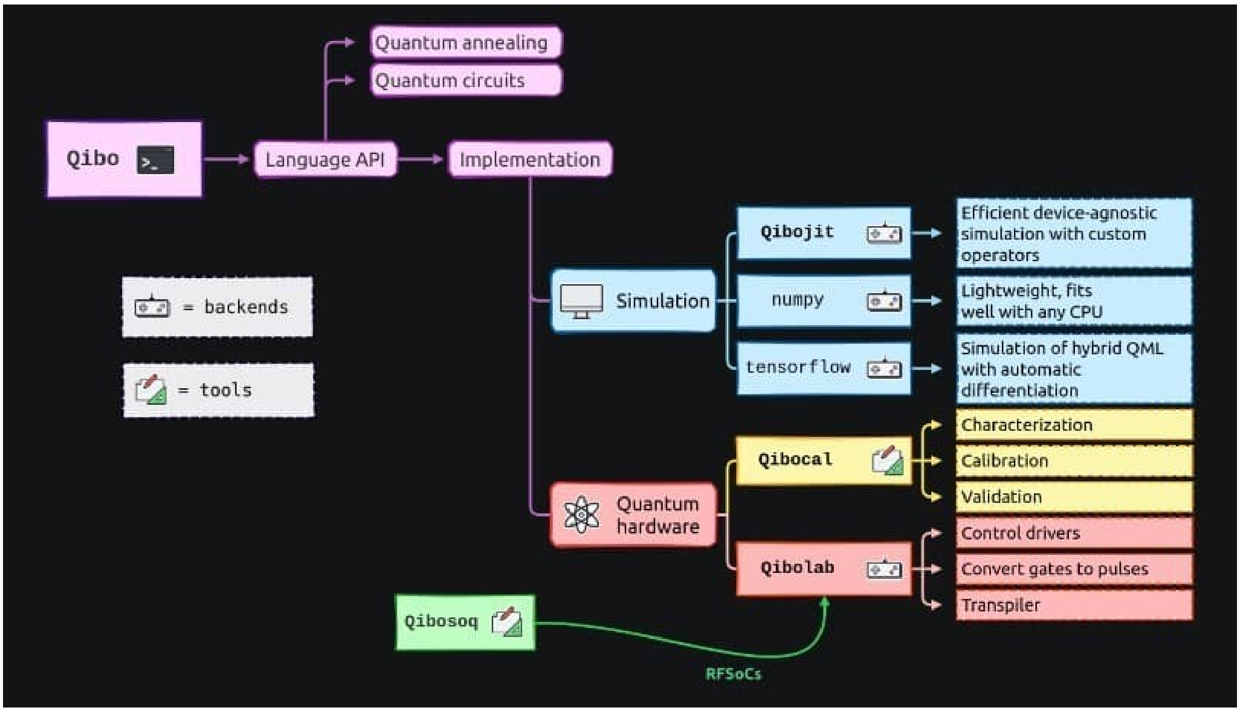


Figure 3.3: Qibo components (source: Qibo 2023)

3.3. NEC Corporation

NEC (Japan) is developing a QA using superconducting parametrons (Box 3.1) based on the Josephson effect (Box 3.3) for implementing superconducting qubits.

A parametron, aka Parametric Phase-Locked Oscillator (PPLO) is a logic circuit element invented by the Japanese computer scientist Eiichi Goto in 1954. The parametron is essentially a resonant circuit with a nonlinear reactive element which oscillates at half the driving frequency. The oscillation can be made to represent a binary digit by the choice between two stationary phases π radians (180 degrees) apart. Parametrons were used in early Japanese computers due to being reliable and inexpensive but were ultimately surpassed by transistors due to differences in speed. See see Box 3.2 and Figure 3.4.

Box 3.1: Parametron

A Phase-Locked Oscillator (PLO) is an electronic device designed to generate a stable and precise output frequency by locking the phase of its internal oscillator to a reference signal. The PLO uses a feedback control system called a Phase-Locked Loop (PLL) to synchronise the output signal of a Voltage-Controlled Oscillator (VCO) with the phase and frequency of a reference oscillator. This ensures that the generated signal maintains a fixed frequency with high stability, even in the presence of environmental changes such as temperature variations.

Box 3.2: Phase-Locked Oscillator (PLO)

The Josephson effect, named after the British physicist Brian David Josephson, is a phenomenon that occurs when two superconductors are placed in proximity, with some barrier or restriction between them. It is an example of a macroscopic quantum phenomenon, where the effects of quantum mechanics are observable at ordinary, rather than atomic, scale. The Josephson effect produces a current, known as a supercurrent, that flows continuously without any voltage applied, across a device known as a Josephson junction, which consists of two or more superconductors coupled by a weak link creating a quantum tunnel junction. The weak link can be a thin insulating barrier (known as a superconductor-insulator-superconductor junction), a short section of non-superconducting metal or a physical constriction that weakens the superconductivity at the point of contact.

Box 3.3: Josephson effect

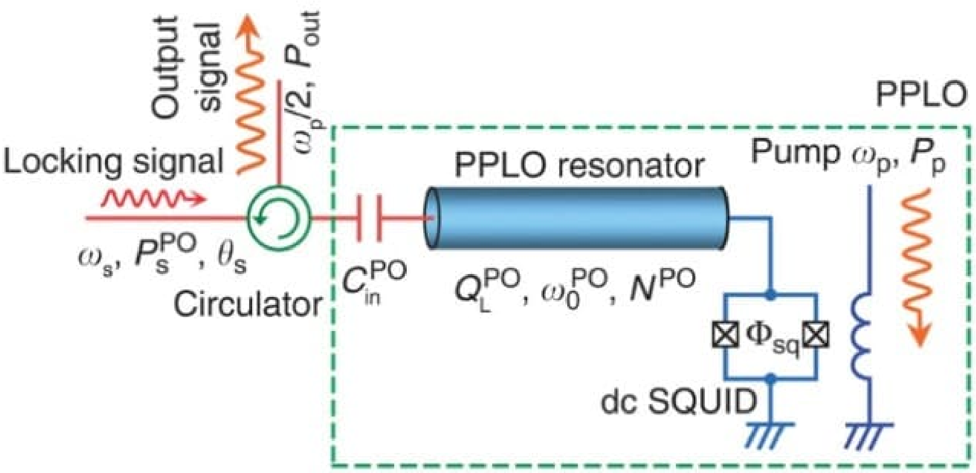


Figure 3.4: Parametric Phase-Locked Oscillator (source: NEC 2023)

NEC's ambition is to produce a coherent QA system "with all-to-all qubit connectivity" but which actually seems to be only nearest-neighbour connectivity (Figures 3.5 and 3.6).

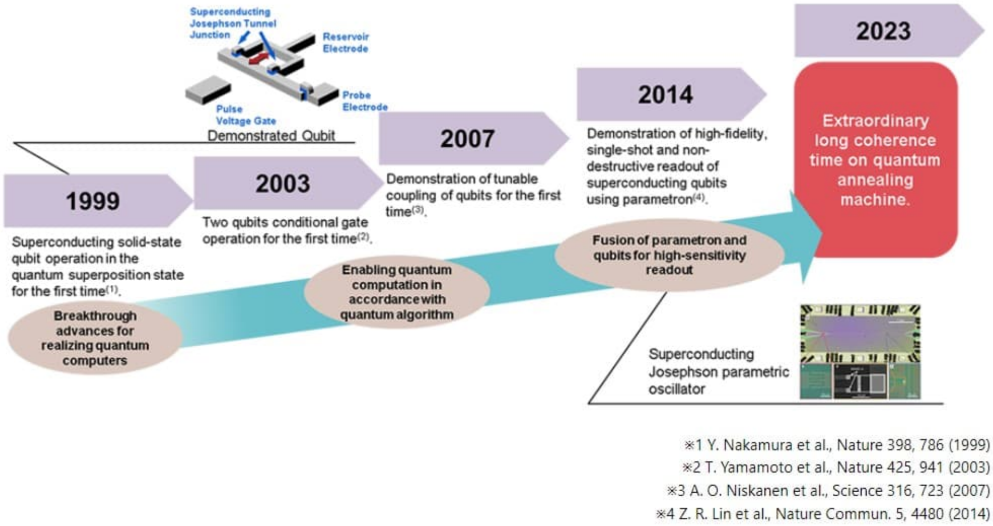


Figure 3.5: QA development roadmap (source: NEC 2023)

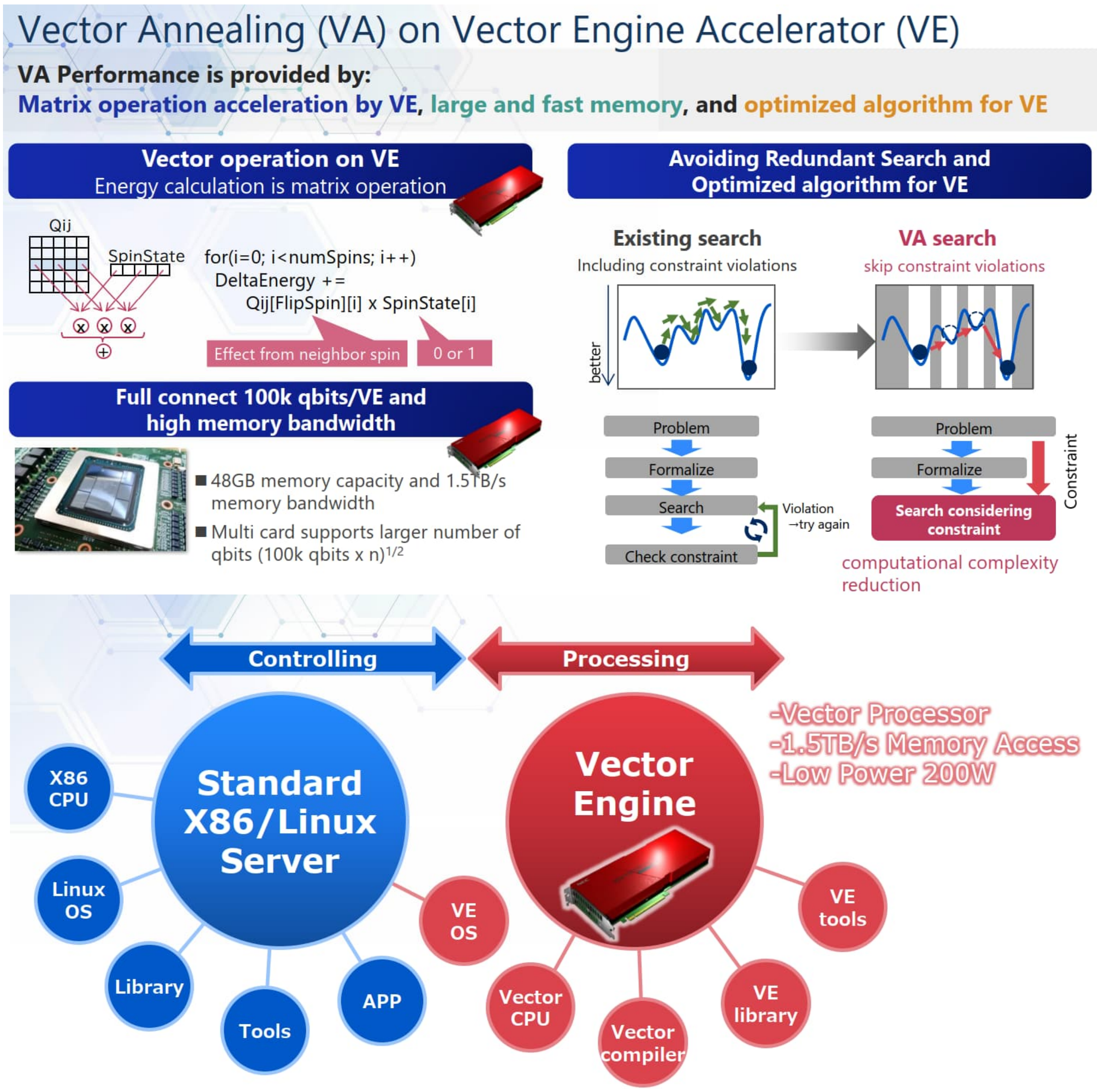


Figure 3.6: Vector Annealing (VA) with Vector Engine (VE) accelerator (source: NEC 2024)

3.4. Nippon Telegraph and Telephone (NTT) Corporation

NTT (Japan) developed a Coherent Ising Machine (CIM). CIM is is a computing technique based on optical neural networks which can solve combinatorial optimisation problems by mapping them onto “hard” Ising optimisation problems.

CIM systems (Figure 3.7) use single-mode photon squeezing (Box 3.4), oscillation at degenerate frequency, Optical Parametric Oscillators (OPOs, Box 3.5) and a measurement feedback technique.

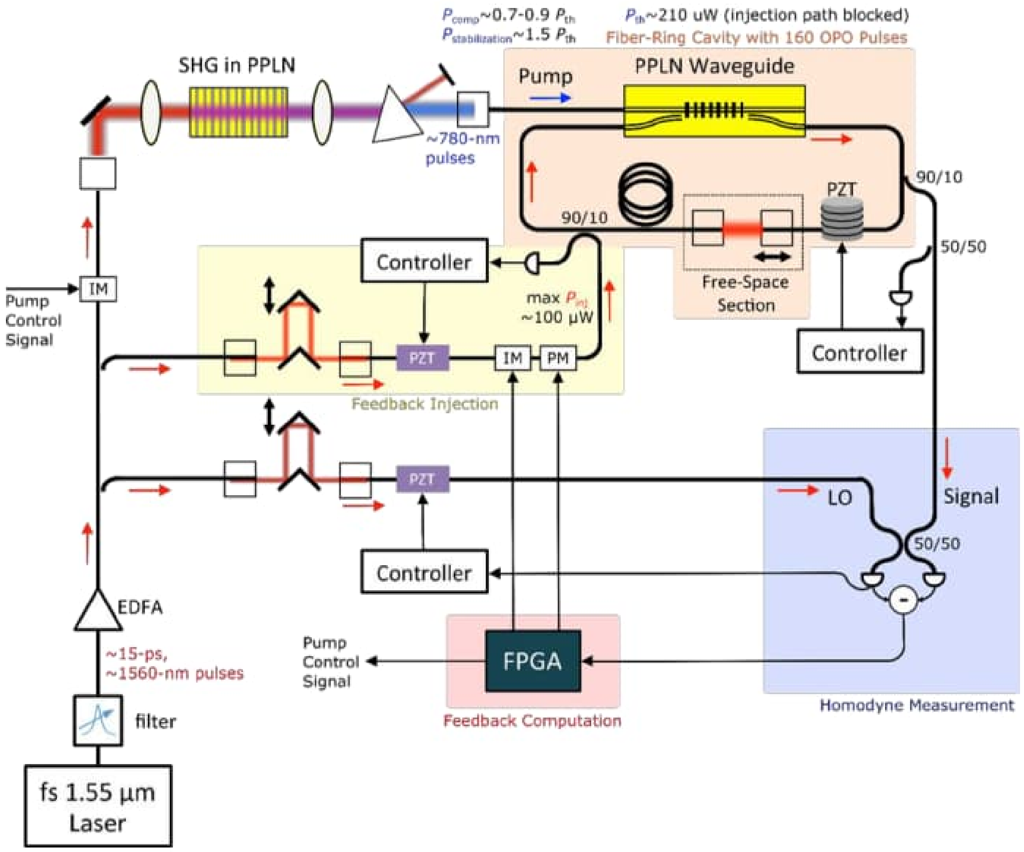


Figure 3.7: Coherent Ising Machine (source: Peter McMahon 2023)

A beam of squeezed light has a lower quantum uncertainty than a beam of coherent photons, at least for some phases of the electromagnetic oscillation hence squeezed light has quantum correlations that enable more precise measurement.

Box 3.4: Photon squeezing

An Optical Parametric Oscillator (OPO) is a coherent light source based on parametric amplification within an optical resonator. It converts an input laser wave (called "pump") into two output waves of lower frequencies.

Box 3.5: Optical Parametric Oscillator (OPO)

CIM optimisation is a bifurcation process guided by both the OPO nonlinearity and the optical coupling between OPOs (implementing many-to-many connectivity).

CIM has distinct advantages over other types of optimisation technologies:

- it is faster and more energy-efficient than classical computing and CMOS digital annealing;
- it is not constrained to the relatively short annealing times and corresponding problem sizes of current QA and gate-based quantum computer technology.

CIM technology may also prove to be useful for Artificial Intelligence (AI) applications such as machine learning. Research is currently underway at MIT on applying CIM-like hardware to accelerate deep neural networks.

3.5. Fujitsu Ltd.

Fujitsu (Japan) developed a quantum-inspired DA that is capable of solving large-scale complex combinatorial optimisation problems in near real-time. Using a digital circuit CMOS design inspired by quantum phenomena, the DA focuses on rapidly solving complex combinatorial optimisation problems without the added complications and costs typically associated with quantum computing methods.

The Fujitsu DA supports 8,192-bit full connectivity, with flexible partitioning for parallel operation and scaling to match problem size and precision requirements. It can be deployed rapidly and easily accessed remotely as a cloud service via Web APIs (Figure 3.8). Fujitsu's DA can also be installed at a customer site for a monthly subscription (Figure 3.9).

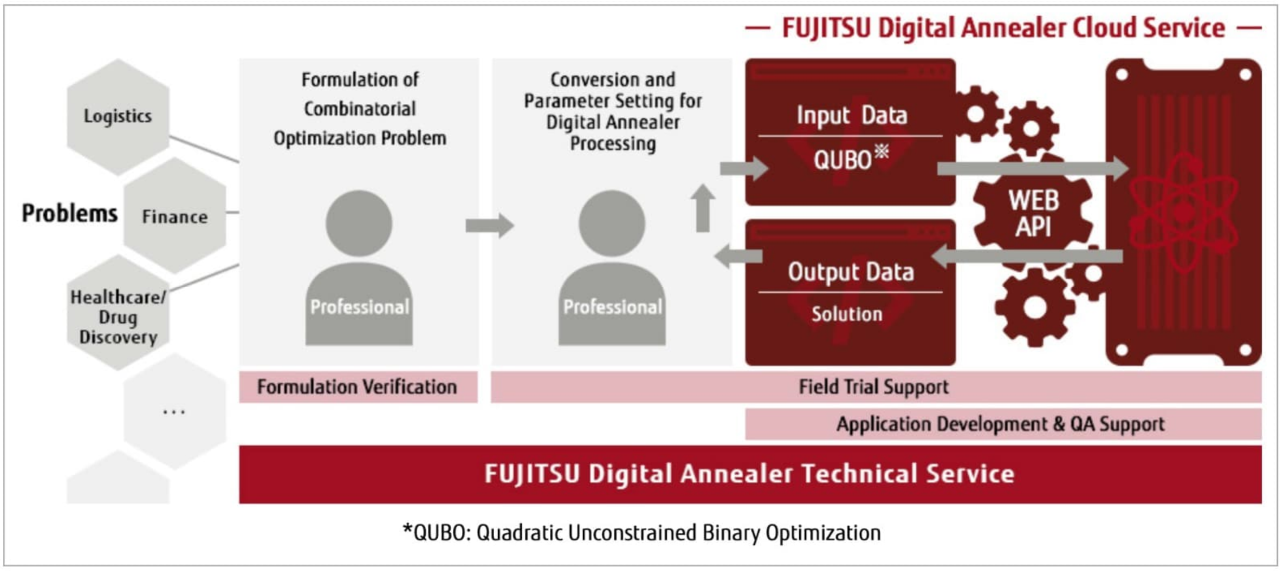


Figure 3.8: Fujitsu digital annealer cloud service (source: Fujitsu 2024)

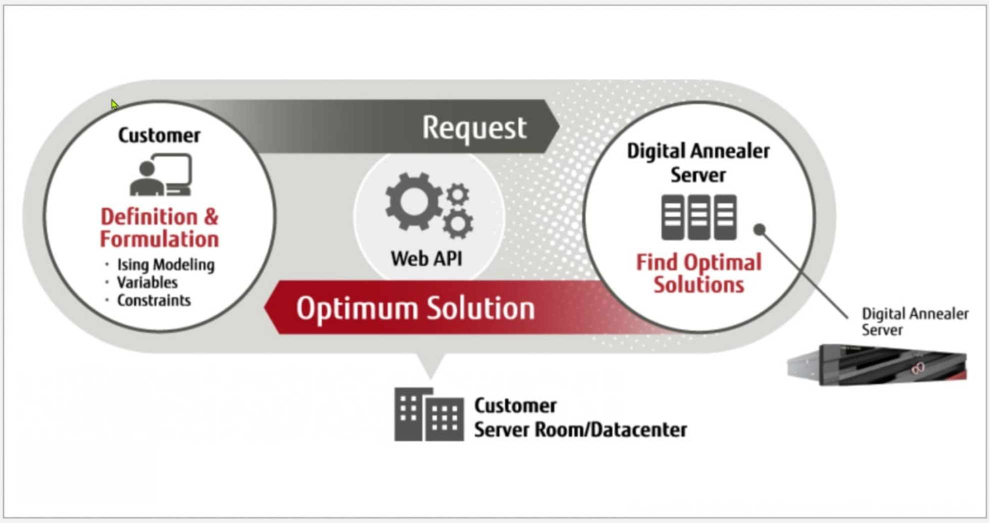


Figure 3.9: On-premises Fujitsu digital annealer (source: Fujitsu 2024)

3.6. Hitachi Ltd.

Hitachi (Japan) developed a DA which implements the behaviour of the Ising model by means of a CMOS circuit. Hitachi's CMOS Annealing Machine (Figure 3.10) can efficiently find practical solutions to combinatorial optimisation problems at room temperature.

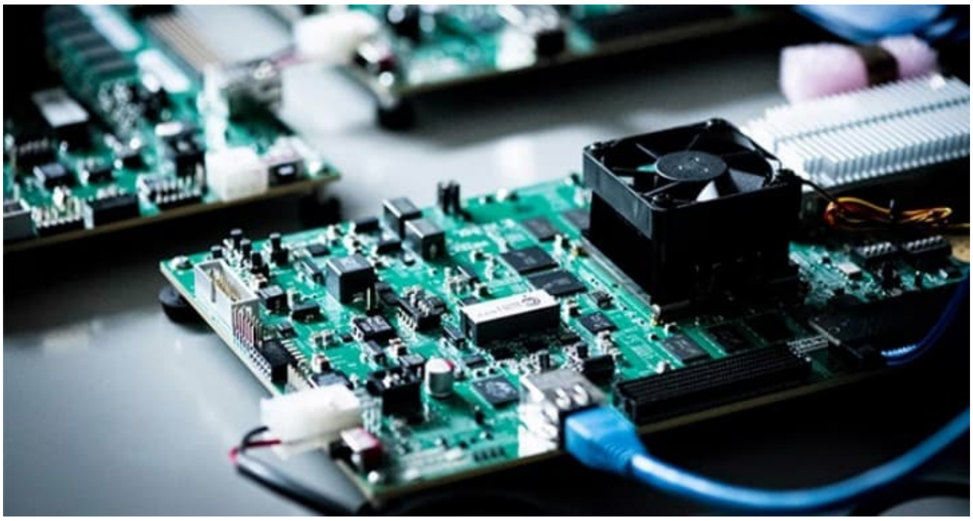


Figure 3.10: CMOS Annealing Machine circuit board (source: Hitachi 2024)

The annealing comprises two processes (Figure 3.11). The first process reproduces the interactions between spins in the Ising model, thereby decreasing the amount of energy in the annealing system. The second process injects noise into the circuit that is reproducing the spins, thereby intentionally disrupting the spins' states. With only the first process, there is a risk that the process becomes fixed on a local solution, i.e. a section where energy expenditure is not at a minimum. The second process avoids fixation on local solutions by seeking a global solution, i.e. the section with minimal total energy expenditure.

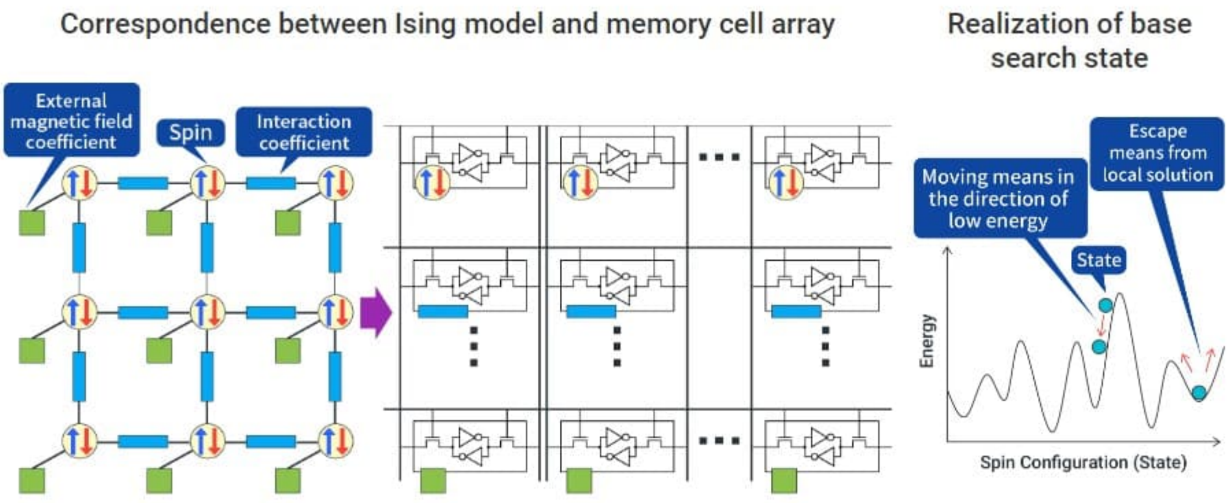


Figure 3.11: CMOS Annealing Machine - Ising model implementation (source: Hitachi 2024)

3.7. Toshiba Corporation

Toshiba (Japan) developed a Simulated Quantum Bifurcation Machine (SQBM), a quantum-inspired optimisation solution based on the Simulated Bifurcation Machine (SBM). SBM is a combinatorial optimisation solver utilising Toshiba's Simulated Bifurcation (SB) algorithm.

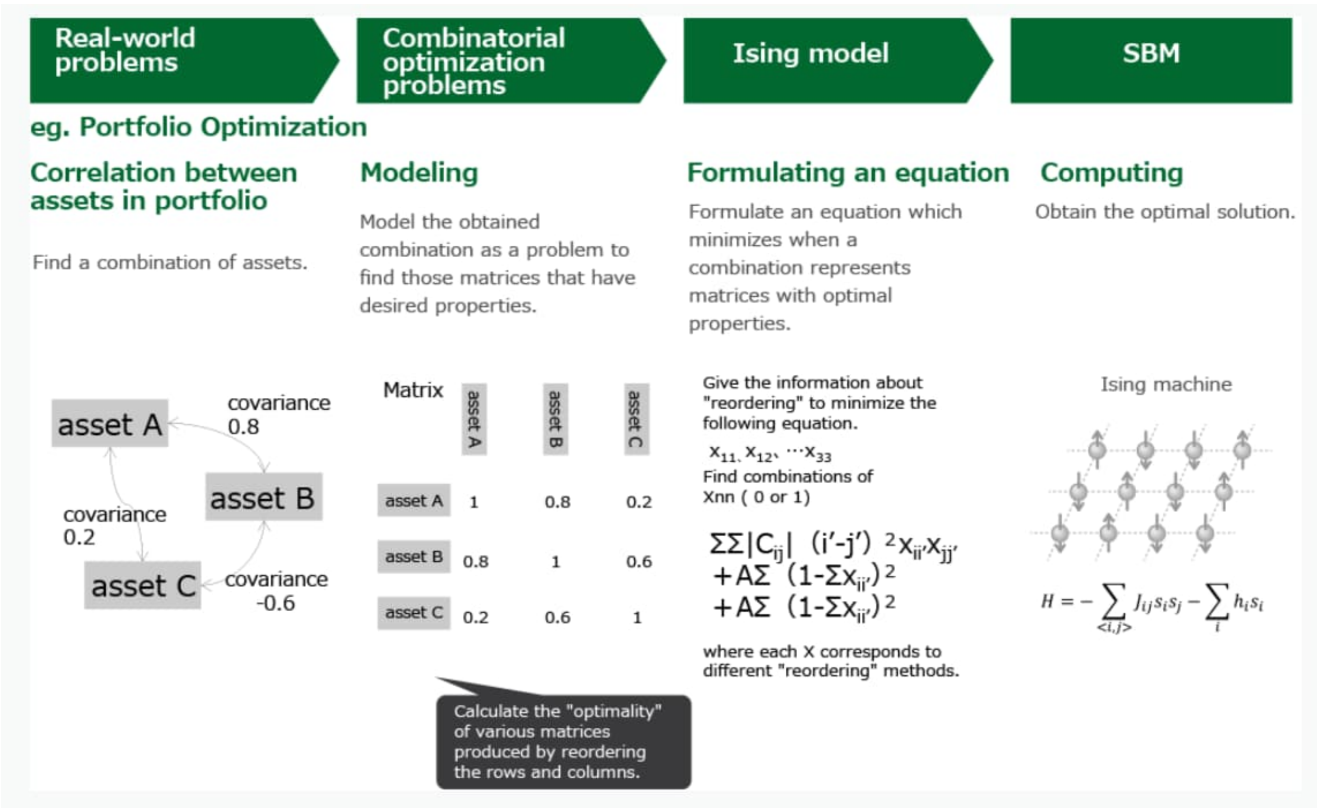


Figure 3.12: Mapping real-world problems to the SBM solver (source: Toshiba 2024)

SQBM implements a practical and ready-to-use Ising machine that solves large-scale combinatorial optimisation problems (with up to 10 million variables). It is implemented on standard classical computers complemented with FPGAs or GPUs. SQBM is also available as a SaaS solution on the AWS cloud platform.

In addition to the general-purpose QUBO solver, Toshiba also provides additional solvers, including:

- Quadratic Assignment Problem (QAP) solver

  A solver that directly solves QAP problems without expressing them in QUBO. Use case example: optimal placement of facilities to minimise the cost of transporting goods between them.

- Quadratic Programming (QP) solver

  A solver that directly solves quadratic binary optimisation problems with linear constraints. Compared to solving similar problems with the QUBO solver, there is no need to incorporate linear constraints into and adjust penalty parameters, thus making it easier to obtain highly-accurate solutions.

- Polynomial Unconstrained Binary Optimization (PUBO) solver

  Supports cubic and quartic problems for solving real-life combinatorial optimisation problems that contain cubic or higher terms.

- Traveling Salesperson Problem (TSP) solver

  A solver that directly solves TSP type problems without expressing the solution in QUBO.

- SHIFT solver

  A solver that directly solves shift scheduling problems. Use case example: assigning jobs to employees while considering various constraints, without using QUBO.

Real-life combinatorial optimisation problems may contain continuous variables. Solving such problems with the QUBO solver requires conversion to quadratic expressions and binary variables, which would result in severe performance degradation. SQBM therefore utilises the features of the SB algorithm to support continuous variables, in order to achieve higher solution performance for these types of combinatorial optimisation problems.

SQBM's parameter automatic adjustment function automates the tuning of unique SBM parameters and quickly finds better solutions without the hassle of manual adjustment.

3.8. Sharp Corporation

Sharp Corporation (Japan) develops an SQA and an AGV Operating System (AOS), which is capable of controlling more than 1,000 AGVs (also called transport robots) that carry goods and parts in logistics warehouses[2] (Figure 3.13).

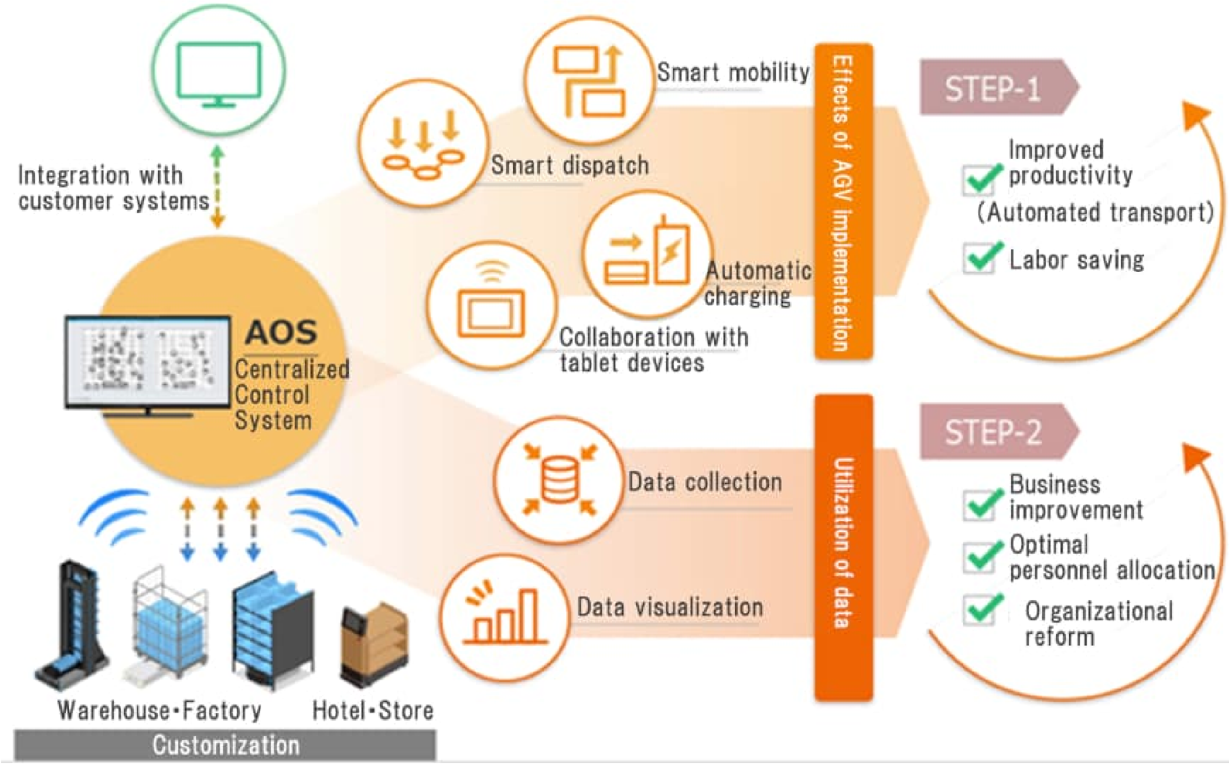


Figure 3.13: Sharp AGV SQA system (source: Sharp 2024)

Sharp's SQA machine is developed jointly with with Tohoku University. It uses FPGA circuits capable of performing parallel calculations at high speed and implements quantum algorithms in a special circuit design, enabling the simulation of quantum annealing on classical computers.

AGV route calculation is a combinatorial optimisation problem that can be solved with an Ising model and an objective function. Sharp's SQA technology is not limited to AGV route calculations in logistics warehouses and could be applied to a wider range of use cases.

[2] According to Sharp, the most advanced QAs currently only support no more than 5,000 AGVs.

4. D-Wave quantum annealing hardware

4.1. Cryogenic subsystem

D-Wave QA qubits operate at 10 to 15 mK and thus require a cryostat (using a dry dilution system).

The cryogenic part includes an enclosure with five layers of magnetic isolation (Figure 4.1) and consumes about 16 kW out of a total energy consumption of about 25 kW. The remaining 9 kW is consumed by the classical part of the qubit control system outside the cryostat.

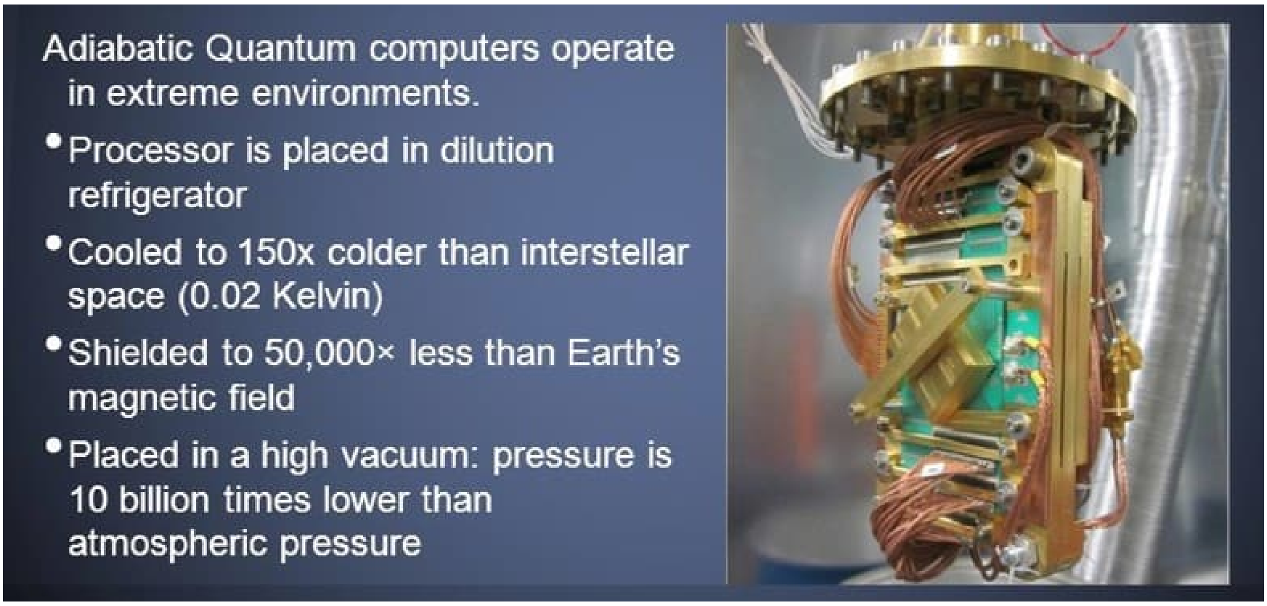


Figure 4.1: Cooling, magnetic shielding and vacuum (source: D-Wave Quantum Systems 2023)

4.2. QPU technology

The D-Wave QPU is a network of superconducting flux qubits that are tunably coupled.

The qubits deployed in D-Wave QAs are niobium-based radio-frequency Superconducting Quantum Interference Devices (rf-SQUIDs). These rf-SQUID devices exploit superconducting current loops interrupted by two Josephson effect barriers that are controlled by variable magnetic fluxes (Figure 4.2).

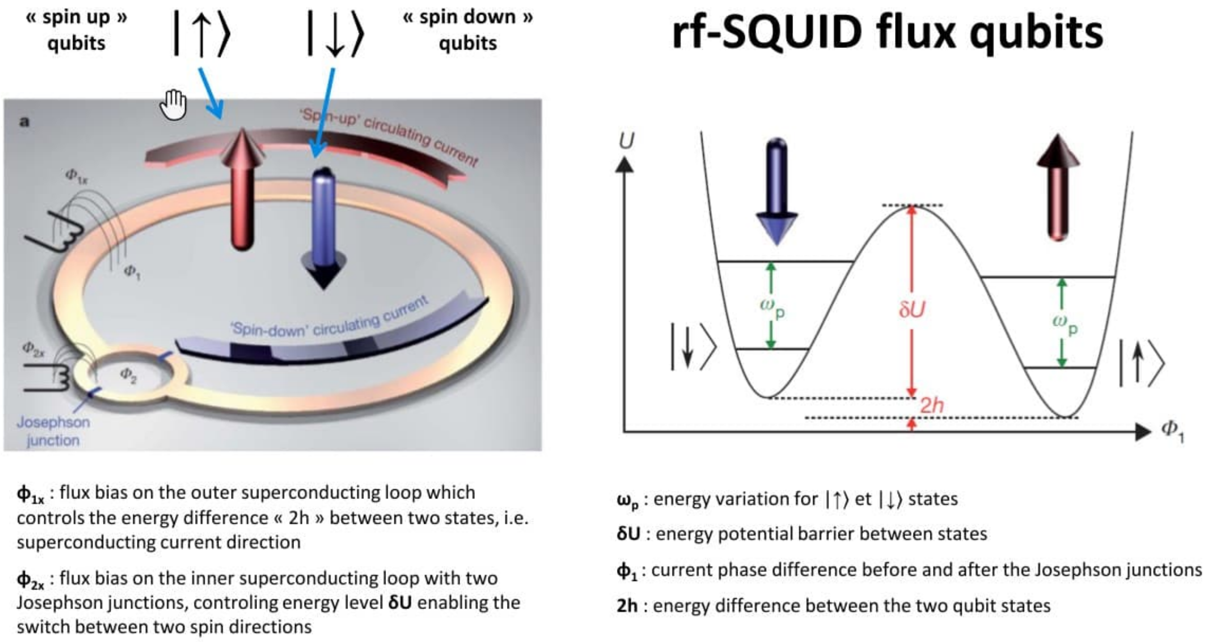


Figure 4.2: rf-SQUID flux qubit (source: D-Wave Quantum Systems 2023)

4.3. Qubit connection technology

In a D-Wave QA system, the hardware graph topology describes the pattern of physical connections between qubits and their couplers. The most important difference between the 2000Q and Advantage QPUs is the upgrade from the Chimera to the Pegasus topology.

Figure 4.3 compares a Chimera example graph on the left (a 6-by-6 grid of unit cells) with a Pegasus example graph on the right (which contains 27 unit cells on a diagonal grid, plus partial cells around the perimeter). Both example graph topologies contain about the same number of qubits: 288 in the Chimera graph and 264 in the Pegasus graph.

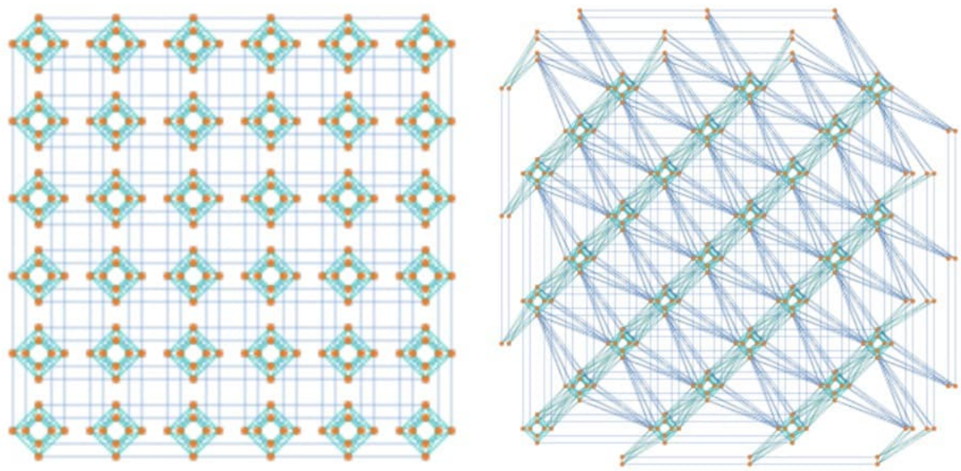


Figure 4.3: Chimera vs. Pegasus qubit connectivity (source: D-Wave Quantum Systems 2023)

In Chimera qubits are oriented vertically or horizontally.

Chimera has two types of coupler: internal couplers connecting pairs of orthogonal qubits (i.e. pairs of qubits with opposite orientation) and external couplers connecting colinear pairs of qubits (i.e. pairs of qubits that are parallel, in the same row or column).

In the Chimera topology, qubits have a nominal length of 4 (each qubit is connected to 4 orthogonal qubits through internal couplers) and degree of 6 (each qubit is connected to 6 different qubits through couplers).

In Pegasus, as in Chimera, qubits are oriented vertically or horizontally.

Pegasus has, in addition to Chimera's internal and external couplers, a third type of coupler: odd couplers. Odd couplers connect parallel qubit pairs in adjacent rows or columns.

In the Pegasus topology, qubits have a nominal length of 12 (each qubit is connected to 12 orthogonal qubits through internal couplers) and degree of 15 (each qubit is connected to 15 different qubits through couplers).

In the Advantage2 Zephyr topology, as in Pegasus and Chimera, qubits are oriented vertically or horizontally. The Zephyr topology features the same three coupler types as Pegasus, with a total of 16 internal couplers, 2 external couplers, and 2 odd couplers. In the Zephyr topology, qubits have a nominal length of 16 (each qubit is connected to 16 orthogonal qubits through internal couplers) and degree of 20 (each qubit is connected to 20 different qubits through couplers).

Figure 4.4 shows an example of the 20 couplers of a qubit in a Zephyr graph with different colours for each of the different coupler types: the internal couplers are green, the external couplers are blue and the odd couplers are red.

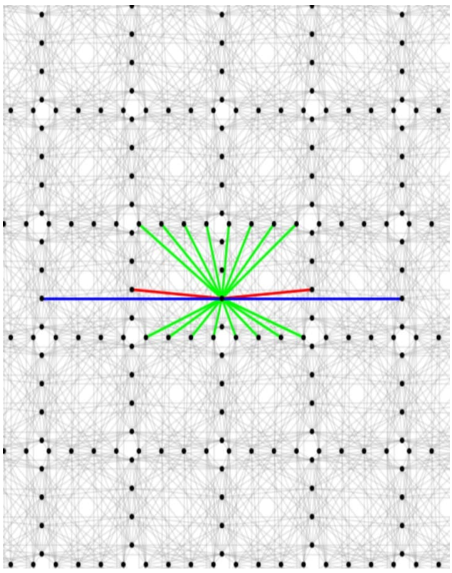


Figure 4.4: Zephyr qubit connectivity (source: D-Wave Quantum Systems 2023)

Qubit state superposition in D-Wave's QAs comes from the connectivity of qubits combined with the tunnelling effect. D-Wave QAs also use entanglement of qubit states but this is limited to the nearest qubits.

4.4. Qubit control technology

Room-temperature electronics generate the qubit control signals, which are multiplexed and sent in digital format from the outside via coaxial cables into the cryostat to program the Digital-to-Analogue Converters (DACs) embedded in the QPU (Figure 4.5). The DACs apply static magnetic control signals locally to the qubits and couplers. There are 5 DACs per qubit for handling qubit control signals and 6 (D-Wave 2000Q), 15 (D-Wave Advantage) or 20 (D-Wave Advantage2) coupler DACs per QA.

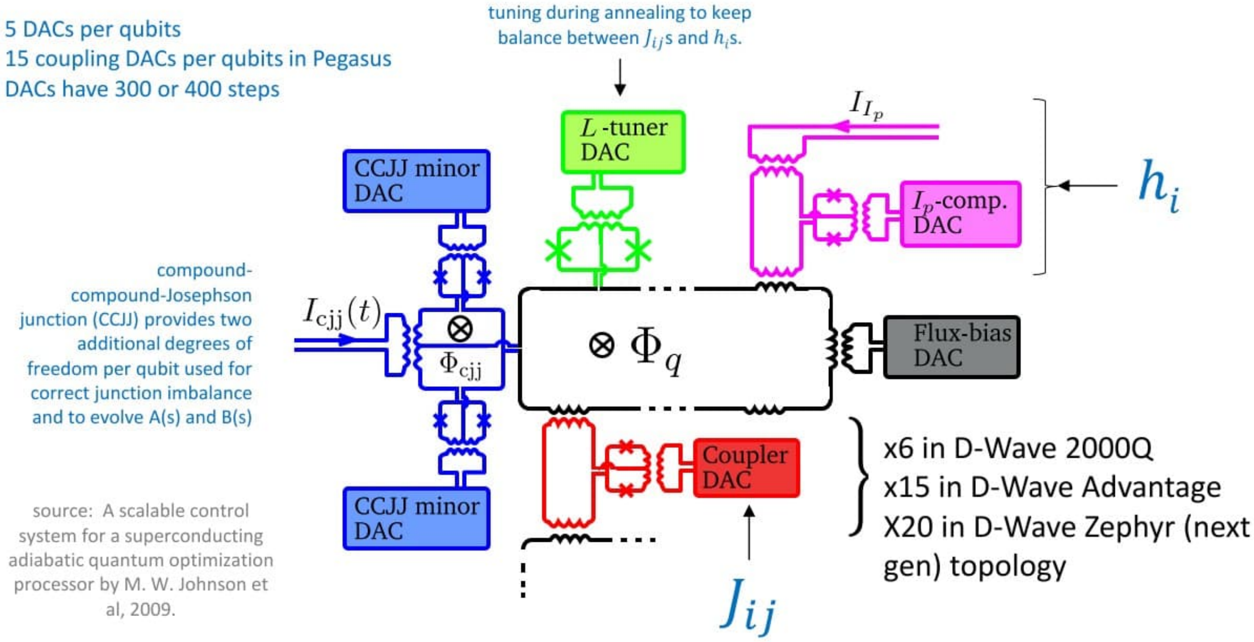


Figure 4.5: D-Wave QA qubit control signals (source: M.W. Johnson et al. 2009)

Integrated DC ramp pulse generation circuits embedded in the QPU quantum chip use RSFQ devices (Box 4.1 and Figure 4.6) for implementing the DACs.

Rapid Single Flux Quantum (RSFQ) is a digital electronic device that uses superconducting Josephson junctions to process digital signals. Josephson junctions are the active elements for RSFQ electronics, just as transistors are the active elements for semiconductor electronics. In RSFQ logic, information is stored in the form of magnetic flux quanta and transferred in the form of Single Flux Quantum (SFQ) voltage pulses.

Box 4.1: Rapid Single Flux Quantum (RSFQ)

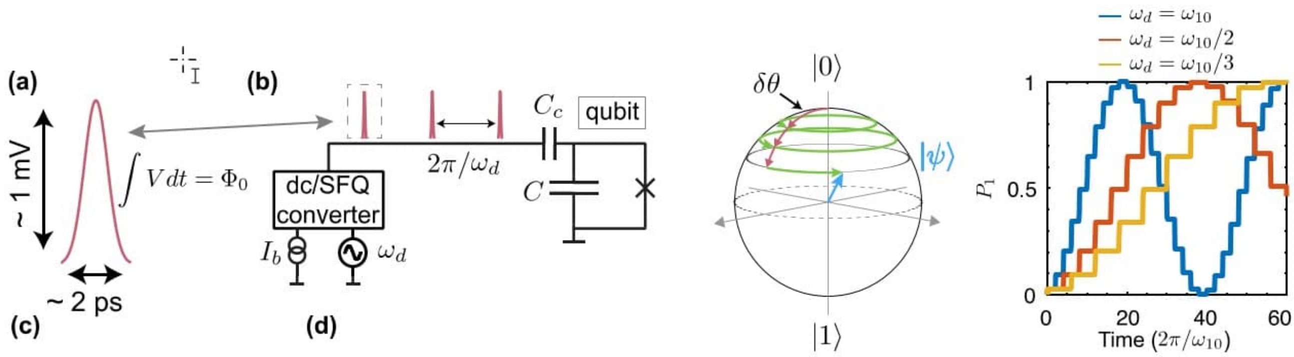


Figure 4.6: RSFQ-based qubit control signal generation (source: E. Leonard et al. 2017)

D-Wave Quantum Systems was the first superconducting qubit manufacturer to use RSFQ electronics in its systems (since its inception). RFSQ's advantages for implementing qubit control are:

- very low power consumption (up to 500 times less than CMOS);
- operating at the same temperature as the superconducting qubits.

The latter advantage means that all the pulse generation electronics can be included in the QPU chipset, which allows to greatly simplify the coaxial cabling that leads from the classical control hardware to the QPU. Furthermore, QAs do not have to send microwave pulses to qubits for controlling quantum gate operations and can thus avoid the related coaxial cables. All in all, the cabling in a D-Wave QA cryostat is far less complex than in typical superconducting gate-based quantum computers.

There is however also a downside as RSFQ devices are quite noisy and thus contribute to the noise affecting the QA's qubits. D-Wave QAs therefore require frequent (re)calibration (calibration is a technique to reduce systematic errors in quantum hardware components).

Note
The error rates in D-Wave QAs are not determined in the same way as the error rates in gate-based universal quantum computers. In a D-Wave QA, the error rate is measured as the precision of the implementation of the Ising model parameters, which is about 2% given the precision of the D-Wave DACs. This high error rate can be mitigated to some extent with quantum error correction techniques, post-processing error correction and machine learning aided error correction.

4.5. Qubit readout technology

Qubit readout is performed as follows. A Quantum Flux Parametron (QFP)-based shift register moves data from the qubits along linear horizontal and vertical tracks to the perimeter of the QPU. At each end of every linear horizontal and vertical track there is a Frequency And Sensitivity Tunable Resonator (FASTR) micro resonator (Figure 4.7). The resonant frequency of each micro resonator is set by an LC tank circuit (Box 4.2). Part of the resonator inductance is provided by a

Direct Current SQUID (DC-SQUID) loop that is coupled to the end stage of the QFP shift register track. Data in the last stage QFP body (circulating or counter-circulating persistent current) modulates the inductance of the DC-SQUID loop which modulates the micro resonator resonance frequency.

Each of the micro resonators is connected to one of two microwave transmission lines that follow the perimeter of the quantum processor. A frequency tone is generated that addresses a particular micro resonator and the transmission of this tone is monitored. The data state of the shift register modulates the micro resonator frequency and thus the transmission of this tone. This enables to quickly read out the state of the shift register via a transmission measurement.

The micro resonators are separated in frequency (frequency multiplexing); this allows to read out all the micro resonators in parallel.

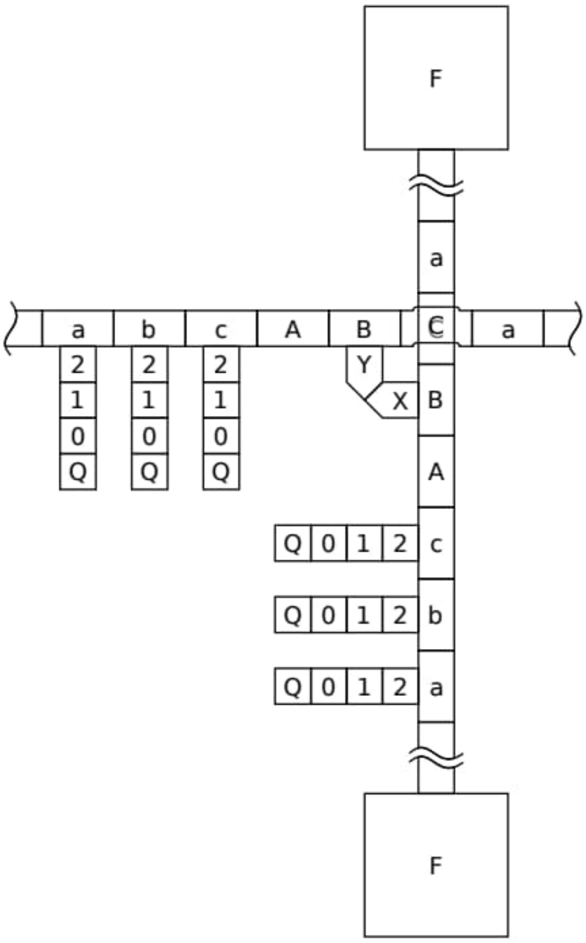


Figure 4.7: Qubits (Q) and FASTRs (F) (source: D-Wave Quantum Systems 2023)

An LC tank circuit (aka resonant circuit) is an electric circuit consisting of an inductor (represented by the letter L) and a capacitor (represented by the letter C) connected together. The circuit can act as an electrical resonator, storing energy oscillating at the circuit's resonant frequency.

Box 4.2: LC tank circuit

5. D-Wave quantum annealing software

To program a D-Wave QA for solving a given optimisation problem, a user maps the problem into a search for the "lowest point in a vast landscape" corresponding to the best possible outcome. The QA considers all the possibilities simultaneously to determine the lowest energy required to form those relationships. The solutions are values that correspond to the optimal configurations of qubits found, e.g. the lowest points in the energy landscape. These values are returned to the user program over the network.

5.1. Ocean Quantum Software Development Kit (QSDK)

D-Wave Quantum Systems' Ocean Quantum Software Development Kit (QSDK) contains software development tools, hybrid solvers and a large set of libraries to solve various optimisation and constraint satisfaction problems (Figure 5.1).

The complexity of quantum programming is abstracted away to enable developers to focus on the business problem at hand. The Ocean QSDK enables users to formulate problems in the QUBO and Ising models. Results can be obtained by submitting a quantum job to an online D-Wave QA or to an hybrid quantum-classical solver in Leap, D-Wave Quantum Systems' real-time quantum computing cloud service (which is also accessible on Amazon Marketplace).

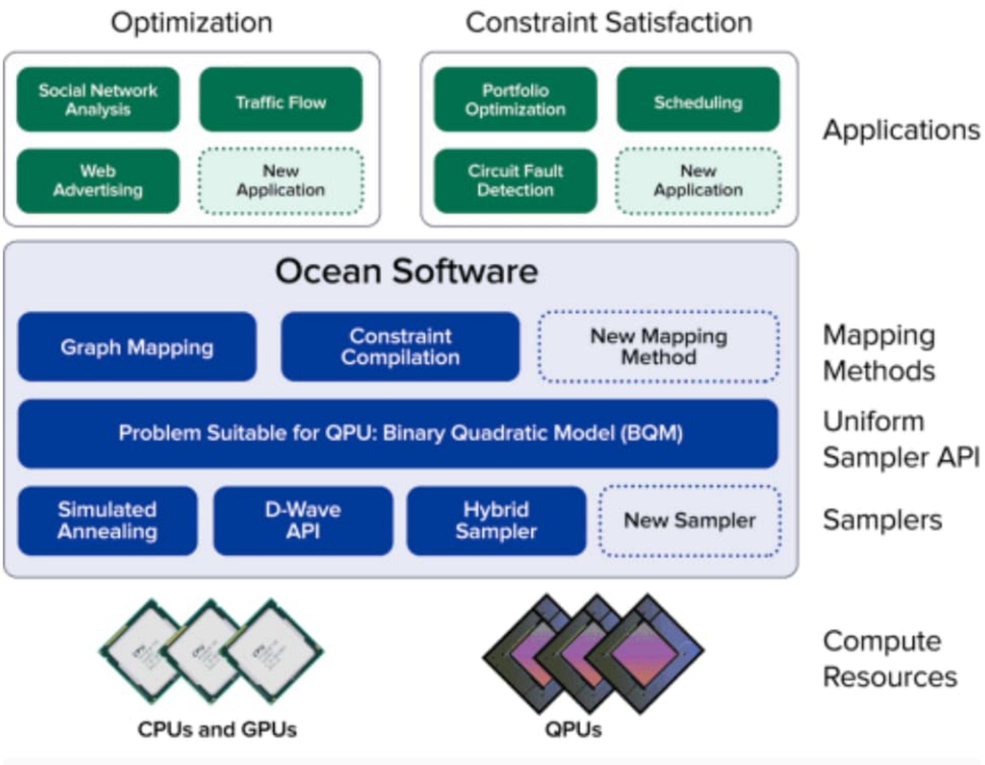


Figure 5.1: Ocean development framework (source: Quantum Zeitgeist 2025)

Ocean QSDK is a suite of open source Python tools and libraries accessible on both the D-Wave Quantum Systems GitHub repository[3] and Leap.

5.2. Leap quantum computing access service

D-Wave Quantum Systems' Leap quantum computing cloud service provides real-time access to D-Wave 2000Q and Advantage QA platforms and to the HSS hybrid quantum-classical solver, both of which are shared resources that continually process user-submitted problems. The problems are solved in a few milliseconds and the solutions are typically returned within seconds. D-Wave QAs and HSS can also be accessed via the AWS Marketplace. The Leap quantum cloud service supports third-party IDEs, both local and cloud-based, that implement the Development Containers specification (aka "devcontainers"), which allows the use of a container as a full-featured development environment..

5.3. Hybrid Solver Service (HSS)

D-Wave Quantum Systems launched the Hybrid Solver Service (HSS) in 2020. HSS contains a portfolio of heuristic solvers that leverage both quantum and classical solution approaches to solve several generic optimisation problems with various categories of inputs and use cases. Furthermore, the HSS solvers provide interface support for applications well outside the native problem formulation, which is quadratic, unconstrained and binary. This interface reduces, and sometimes completely eliminates, the need for developers to translate their application problems into a formulation that matches the QA hardware.

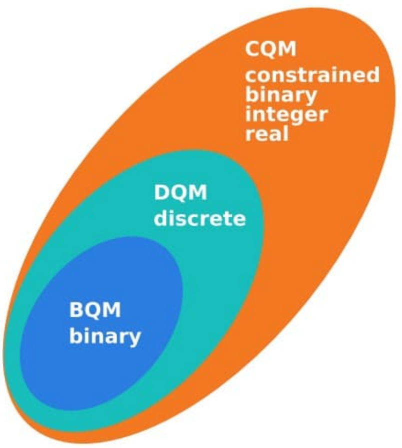


Figure 5.2: Hybrid Solver Service (source: D-Wave Quantum Systems 2022)

[3] GitHub (a subsidiary of Microsoft) provides internet hosting for source code version control using Git (open-source software for tracking changes in a set of files). GitHub offers features for source code development projects, e.g. collaboration among programmers, task management, bug tracking, continuous integration and wikis. It is the largest source code host for open-source projects.

The Binary Quadratic Model (BQM) solver is for unconstrained quadratic problems defined on binary variables (taking two values).

The Discrete Quadratic Model (DQM) solver is for unconstrained quadratic problems defined on discrete variables (taking multiple values).

The Constrained Quadratic Model (CQM) solver expands the optimisation problems that D Wave QAs can solve. It can solve constrained problems defined on binary, integer and real variables with up to 500,000 variables and up to 100,000 constraints.

The DQM and CQM solvers are part of efforts by D-Wave Quantum Systems to expand the scope of models that can be solved directly in HSS, without needing additional translation to BQMs. These solvers can be more convenient to use and, in some cases, can deliver better hybrid performance than the BQM solver.

A solver in the HSS portfolio incorporates a hybrid quantum-classical workflow. Each solver has a classical front end that reads an input Q and (optionally) a time limit T. It then invokes one or more hybrid heuristic solvers (computation threads) to search for good-quality solutions to Q (Figure 5.3).

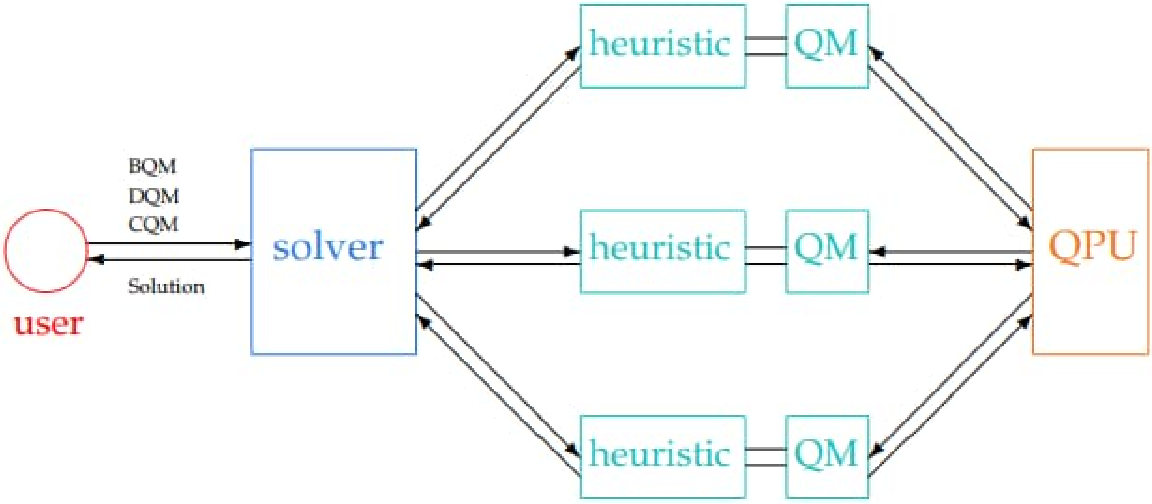


Figure 5.3: HSS hybrid quantum-classical workflow (source: D-Wave Quantum Systems 2022)

The HSS solvers are designed in such a way that the QPU always has a chance to speed up convergence. However, this does not necessarily mean that quantum speed-up always occurs, because some inputs are easily solved heuristically without needing a quantum boost, and some inputs may have complex structures that resist quantum speed-up.

The heuristic solvers run in parallel on CPU and/or GPU platforms. Each of them contains a classic module, which explores the solution space, and a quantum module, which formulates quantum queries that are sent to a back-end D-Wave QPU.

Replies from the QPU are used to guide the heuristic module toward more promising areas of the search space, or to find improvements to existing solutions. Each heuristic sends its best solutions to the front end before the time limit is reached, and the front end forwards best results to the user.

5.4. Comparison of BQM, DQM and CQM solvers

A comparison of the currently available features for BQM, DQM and CQM solvers is provided in Table 5.1.

	BQM	DQM	CQM
Objective Function	linear & quadratic	linear & quadratic	linear & quadratic
Variable Type **Max Values per Variable**	binary 2	discrete 65,000	binary, integer, real 2, $\pm 2^{53}$ [1]
Constraint Representation	via penalties	case restriction [2] via penalties	variable bounds integer linear & quadratic equality integer linear & quadratic inequality real linear equality & inequality via penalties
Max Variables [3]	1 million	5,000	500,000
Max Constraints	–	–	100,000
Max Biases [4]	200 million	5 billion	2 billion

Table 5.1: Feature comparison of HSS solvers (source: D-Wave Quantum Systems 2022)

Notes

[1] Variables are represented as BINARY, INTEGER and REAL types.

[2] The BQM solver uses case restriction for constraints involving forbidden combinations of values assigned to variables or pairs of variables.

[3] For BQM and CQM solvers, the maximum number of variables is also limited by the maximum number of biases.

[4] For BQM and CQM solvers, the number of biases is the number of nonzero weights on all nodes and edges of the input graph; for DQM this is the number of all nonzero weights on all cases assigned to nodes and edges.

D-Wave Quantum Systems compared the relative performance of its BQM, DQM and CQM solvers. Such a performance comparison requires testing of problems that can be translated to run on all three solvers. Reformulating problems from BQM to DQM to CQM is straightforward. However, reformulating problems from CQM to DQM to BQM can sometimes be prohibitively complicated. For this reason, D-Wave Quantum Systems elected three problem test sets that are simple enough to allow easy translation in both directions:

1. The BQM problem test set comprises 15 inputs from the MQLib problem repository of MaxCut and QUBO inputs. These unconstrained binary inputs represent a variety of application domains and contain $N \in [1200 \ldots 2500]$ variables.

For a given graph, the Maximum Cut (MaxCut) is a cut whose size is at least the size of any other cut. That is, it is a partition of the graph's vertices into two complementary sets, such that the number of edges between them is as large as possible. Finding such a cut is known as the Maximum Cut Problem aka MaxCut Problem (MCP).

Box 5.1: MaxCut problem

2. The DQM problem test set consists of 15 inputs from the DIMACS graph colouring problem repository. The graph colouring problem is to assign colours to nodes of a graph, so that no two edge endpoints have the same colour, in a way that minimises the total number of different colours used. These inputs come from a variety of applications and have sizes $N \in [74 \ldots 561]$.

The graph colouring problem involves assigning colours to certain elements of a graph subject to certain restrictions and constraints. The process of assigning colours to the vertices such that no two adjacent vertexes have the same colour is called graph colouring aka vertex colouring.

Box 5.2: Graph colouring problem

3. The CQM problem test set consist of 15 randomly generated inputs for the TSP problem. The TSP problem is to assign a “visit index” (first, second, etc.) to nodes in a graph, to minimise the total weight of edges between successively visited nodes, under the constraints that each node is visited exactly once and that each index is assigned exactly once. These inputs have sizes $N \in [35 \ldots 63]$, and uniform edge weights in $[1, 2N]$.

The Travelling Salesperson Problem (TSP) is an optimisation problem where a salesperson must visit a given set of cities exactly once, starting and ending at the same city. The goal is to find the shortest possible route that covers all the cities and returns to the starting point.

Box 5.3: TSP problem

The outcome of the solver performance comparison is shown in the Figure 5.4. The area between box endpoints corresponds to the middle 50% of the distribution, horizontal lines within the boxes are medians, and lines and individual points outside the boxes show the distribution tails and outliers.

<u>Note</u>
As currently deployed, the BQM solver always returns a single solution, while the DQM and CQM solvers may return multiple solutions, depending on input properties and internal configurations. The best-quality solution returned with a 5 minute time limit was recorded for the purpose of performance comparison.

The BQM solver (blue) performs best on the MQLib test set and shows the worst performance on the DIMACS Graph Coloring test set. The CQM solver (teal) performs best on the TSP test set and shows the worst performance on the MQLib test set. These outcomes show the importance of choosing the right solver for the task at hand.

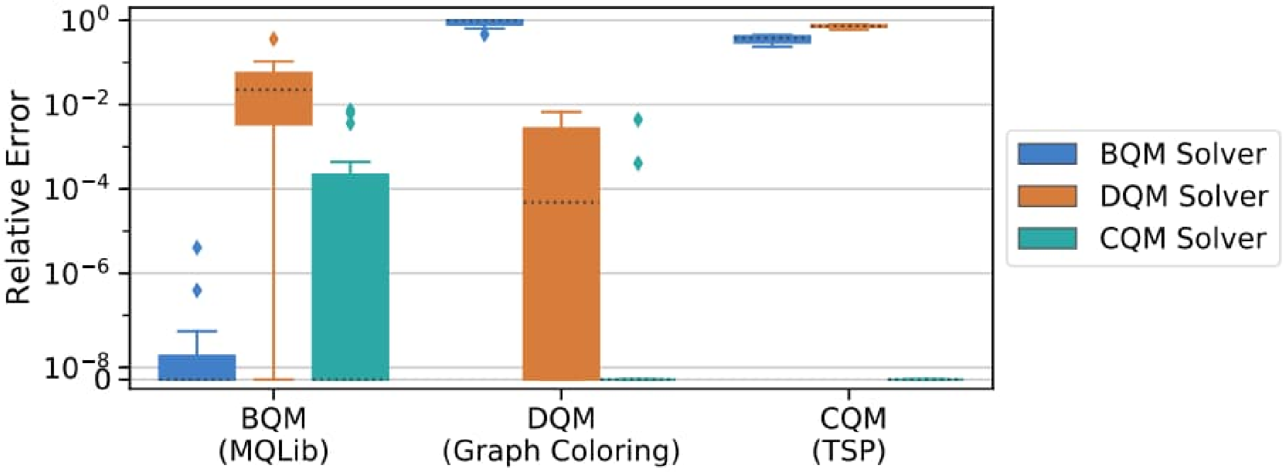


Figure 5.4: Performance comparison of HSS solvers (source: D-Wave Quantum Systems 2022)

The overall conclusion is that, because CQM is able to represent constraints explicitly, it tends to be more efficient than BQM and DQM at finding good-quality feasible solutions to constrained problems. However, unconstrained binary problems can be more efficiently solved by BQM.

D-Wave Quantum Systems has performed performance testing for different CQM solver releases, including algorithmic improvements to existing test problems. It was shown that the CQM solver's performance significantly improved with each new CQM release. It is expected that the CQM solver will eventually replace the DQM solver in HSS.

5.5. Examples of problem solving with BQM, DQM and CQM

The definition of a problem to be solved includes the following four steps:

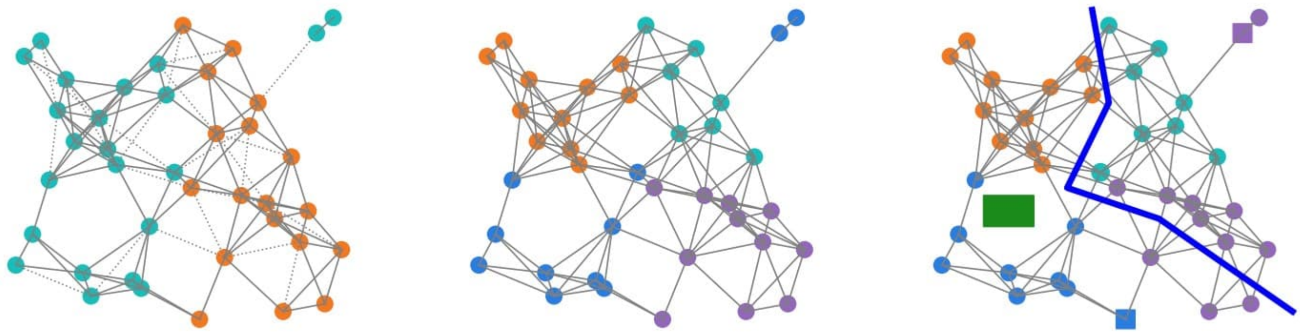
1. Defining the decision variables.
2. Entering the information necessary to describe the problem as constants (if needed).
3. Defining the problem constraints (if any).
4. Formulating the objective function.

Figure 5.5 provides examples of MaxCut problem solutions for the BQM, DQM and CQM solvers:

(a) BQM problem solution with 2 values (teal and orange);

(b) DQM problem solution with 4 values (orange, teal, purple and blue);

(c) CQM problem solution with constraints (defined on integer variables assigned to graduation of colours).



(a) BQM (b) DQM (c) CQM

Figure 5.5: MaxCut problem solution examples (source: D-Wave Quantum Systems 2022)

Problem solving with BQM

The BQM graph (a) in the Figure 5.5 shows a simple BQM MaxCut problem. The nodes of the graph are variables and the edges represent interactions between pairs of variables. A solution to the problem corresponds to an assignment of values (in this case colours) to the nodes of the graph. This is a binary problem because only two values 0 (teal) or 1 (orange) can be assigned to the nodes.

The edges of the graph are assigned numbers, called biases, that express preferences for certain value combinations on endpoint nodes. In this example, a solid edge has negative bias, expressing preference for same values (0,0) or (1,1), and a dotted edge has positive bias, expressing preference for different values (1,0) or (0,1). Assume that the length of an edge indicates the magnitude of the bias and the strength of the preference.

Each possible solution has a quality score *S*, computed according to how well the assignment satisfies the preferences expressed by biases. This is a quadratic problem because calculation of *S* incorporates edge and node biases, whereas a linear problem only considers node biases. The MaxCut problem is, given a graph and its biases, to find an assignment of binary values to nodes that maximises *S*.

Variations on this abstract problem arise in many real-world application areas, for example:

- **VLSI circuit design**

  Nodes represent circuit components and edge biases represent preferences that components be located on "same" or "different" design layers. An optimal MaxCut solution assigns components to two layers (0 or 1) in a way that minimises the cost of wires needed to connect components within and between the layers.

- portfolio allocation

  Nodes represent financial assets available for purchase. Node biases represent expected returns, and edge biases represent price correlations (positive or negative) between asset pairs. A robust portfolio minimises risk by using diversification to maximise negative correlations within a group of selected assets. An optimal solution to this problem divides the assets into two groups (“select” and “omit’), to maximise return and minimise risk in the selected set.

- social network analysis

  Nodes represent people, and edge biases represent friendly and hostile encounters between them. An optimal solution to the “community detection” problem assigns people to two groups to maximise a score measuring friendly relationships within groups, and hostile relationships between members of different groups.

  Imagine that the edge biases represent friendly (solid) and hostile (dotted) encounters among citizens of twelfth-century Verona in Italy. The computational problem is to assign citizens to the groups Montague (0, teal) and Capulet (1, orange), to maximise the MaxCut score *S*. The BQM graph (a) in Figure 5.5 shows one possible solution.

  Researchers in social network analysis study the number of “frustrated” edges in an optimal solution, i.e. hostile encounters within a group or friendly encounters between members of different groups. For example, Juliet is friendly with both Romeo and her father, but Romeo and Lord Capulet have a hostile relationship: any assignment must frustrate at least one edge of this triangle. In another example, Mercutio has a hostile relationship with both groups, so some edges must be frustrated no matter which group he is in. A high number of frustrated edges in an optimal solution is a sign of structural imbalance, which is associated with increased potential for clashes, violence, and perhaps even tragedy.

Problem solving with DQM and CQM

Suppose now that the BQM graph (a) and the DQM graph (b) in the Figure 5.5 represent solutions to a social network analysis problem which involves four different social groups in Verona and Florence: Capulet=orange=1, Montague=teal=2, Medici=purple=3, and Albizzi=blue=4. DQMs and CQMs may be defined directly on discrete and integer variables which allows the user to circumvent both problems. In this context, discrete refers to categorical values such as colours or surnames, whereas integer refers to numerical values.

Although it is technically possible to formulate this problem as a BQM, techniques for doing so would involve replacing each node in the original graph with four binary nodes (one for each possible value assignment), creating a four-fold increase in problem size. It would also require rewriting the objective function to ensure that at exactly one of each binary combination (node, value) is selected. Using the DQM or CQM solvers allows the user to circumvent the problem of expanding input size and the inconvenience of problem reformulation.

Problem solving with constraints

In the previous example problem the notational differences between DQM and CQM formulations are small: in DQM the four values are called cases, and the input would contain lists of valid cases that can be assigned to each node; in CQM the input would specify valid ranges of integers [0 … 3] for each node.

The primary difference between the CQM solver and BQM and DQM is that CQM offers a rich language for expressing constraints, i.e. rules about what constitutes a feasible (valid) solution to the problem. In contrast, all solutions to (unconstrained) BQMs and DQMs are considered feasible. While it is technically possible to incorporate constraints in objective functions for BQM and DQM formulations, the constraint language of CQM is much more convenient to use. In addition, the direct approach gives the CQM solver a performance edge by allowing it to recognise and avoid infeasible regions of the solution space. Furthermore, representation of more realistic models can greatly improve the practical value of solutions found by the CQM solver.

To illustrate this point, suppose that the CQM graph (c) in the Figure 5.5 describes city features including a river (blue line), a duomo (large green square), and two palazzi (square nodes). With integer variable it becomes possible to express constraints involving (linear) sums of node and edge weights, (quadratic) sums of products of nodes and edge weights, and sums of node values. Rules such as the following can be expressed using this interface:

- the two palazzi must be assigned to two different families;
- the five nodes surrounding the duomo cannot all be from the same family;
- every family must be assigned to at least 8 and no more than 12 nodes;
- no more than half the edges across the river can have endpoints assigned to different families.

Problem solving with integer and real variables

The CQM solver supports representation of continuous models defined on real-valued variables as well as integers and binaries. Models containing real variables are typically found when the values to be assigned to nodes represent locations in space or time.

Note
As a general rule, problems that are naturally defined in terms of real values are best solved using continuous models. However, the CQM solver currently supports a broader set of integer constraints than real constraints, and therefore, formulation as an integer model may be the only available option in some cases.

However, development of the CQM solver progresses rapidly and the variety of supported constraint types is expected to increase in future versions.

An input to a Job Shop Scheduling (JSS) problem consists of a list of jobs to be performed. There are five jobs (gold, orange, green, blue, teal).

Each job is divided into a sequence of tasks (coloured blocks numbered 0, 1, 2, 3 and 4) of varying durations (indicated by block widths). Each task is performed using a specific machine (A, B, C, D and E) in the shop.

There is one variable per task, and the values assigned to tasks are their start times. The optimisation problem is to assign a start time to each task so as to minimise the total makespan (Box 5.4), i.e. the time between the start of the first task and the finish of the last task, while obeying two constraints:

- within a job, each task must finish before its successor task can start; in Figure 5.6, all tasks of the same colour obey this constraint;
- a machine can perform only one task at a time; in Figure 5.6, no machine is assigned tasks that overlap in time.

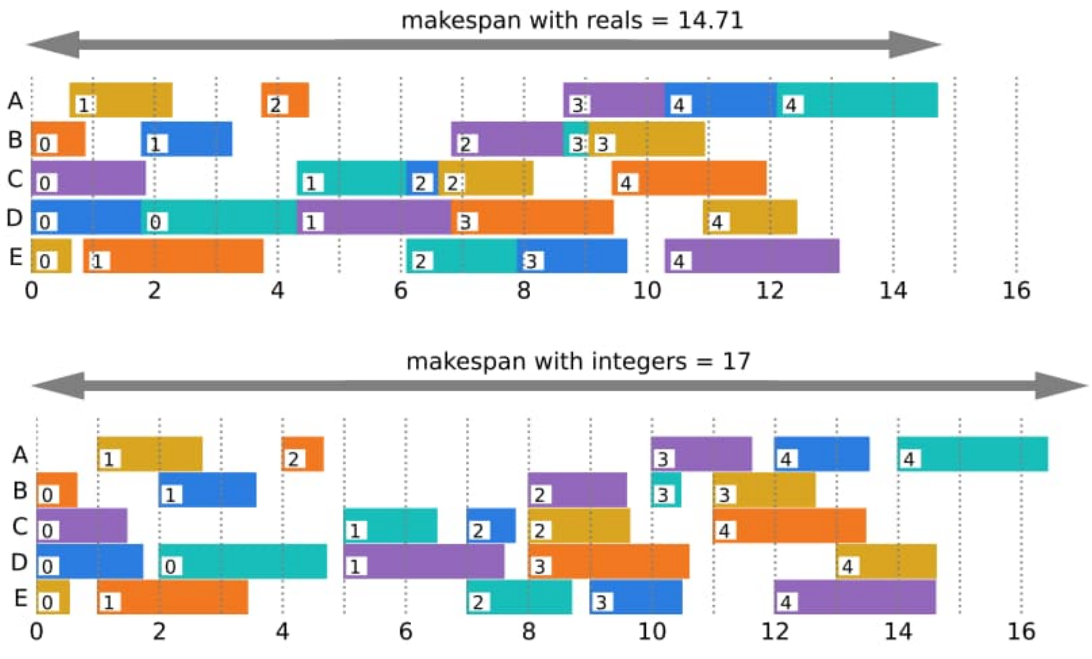


Figure 5.6: Real vs. integer JSS variables (source: D-Wave Quantum Systems 2022)

In Operations Research (OR), the makespan of a project is the length of time that elapses from the start of work to the end. This type of multi-mode resource constrained project scheduling problem seeks to create the shortest logical project schedule, by efficiently using project resources, adding the lowest number of additional resources as possible to achieve the minimum makespan.

Box 5.4 Makespan

In the top version of Figure 5.6 where real values are assigned to the task start time variable, a task can start immediately after its predecessor ends. In the bottom version, where integers are assigned to the task interval variable, time is divided into discrete intervals, e.g. one hour each, and tasks are assigned to the start of each interval.

A comparison of the top and bottom solutions shows that requiring each task to start at the top of the hour creates wasted time whenever a task finishes early. The use of integers instead of reals increases the total makespan by about 16 percent (from 14.71 to 17 hours).

Variations on the JSS problem may be found in many real-world applications, for example:

- work crew scheduling

  The "jobs" are construction sites, each consisting of certain tasks (HVAC installation, plumbing, flooring, painting, etc.) to be performed, in a specified order. The "machines" are specialised work crews that travel from site to site (one site per day) to perform the tasks. The optimal schedule assigns days to work crews, to minimise the time to complete construction at all sites.

- airport scheduling

  Arrival of an aircraft at an airport consists of a sequence of steps requiring exclusive use of certain airport resources: approach on path A, land on runway B, taxi across runway intersections C and D, and so forth. The "jobs" are the aircraft, the tasks are the arrival steps, and the "machines" are the airport resources and/or ground crew members necessary to each step. An optimal schedule assigns times to each step to minimise the total time required for all incoming flights to arrive at their gates.

5.6. NL-Hybrid Solver

In 2024, D-Wave Quantum Systems added the Nonlinear-Program Hybrid Solver (NL-Hybrid) to its HSS solver portfolio. NL-Hybrid allows for the definition of variables in more advanced formats. In addition to allowing variables defined as binary and integer values, NL-Hybrid permits the definition of the following types of decision variables:

- list (number_variables)

  The solver can use a list as the decision variable to optimise, this list being an ordered permutation of size number_variables describing a possible itinerary.

- set (number_variables)

  The decision variable can be a set, being a subset of an array of size number_variables, representing possible items included in a knapsack.

- **disjoint_list (n_variables, n_lists)**

  The solver can employ a disjoint_list as the decision variable, which divides a set of n_variables into n_lists disjoint ordered partitions, each representing a permutation of variables. This encoding is appropriate for complex logistic problems such as the Vehicle Routing Problem (VRP).

  There is a variant of this variable, called disjoint_bit_sets, where the order of the produced partitions is not semantically meaningful.

Like CQM, NL-Hybrid permits linear, quadratic, inequality and equality constraints, expressed even arithmetically. This aspect represents a significant contribution compared to the BQM and DQM hybrid solvers.

Benefits of using NL-Hybrid over the other HSS solvers (BQM, DQM and CQM) are:

- In the field of optimisation, whether by means of classical or quantum systems, the performance of a solver is closely tied to the size of the solution space of the problem at hand. Generally, algorithms perform better in smaller solution spaces. Therefore, employing decision variables that act as implicit constraints is an effective way to reduce the solution space and the complexity of the problem. For example, using list (number_variables) to represent a TSP problem implicitly ensures that no nodes are visited more than once along the route. For this reason, the use of the aforementioned decision variables is a significant advantage for NL-Hybrid.

- NL-Hybrid is built to manage low-level operational specifics, eliminating the need for users to have any expertise in properly parameterising the QPU.

- NL-Hybrid accepts inputs that are much larger than those of other solvers focused on solving problems in QUBO format and even larger than those of the rest of the solvers within HSS. NL-Hybrid is intended to take advantage of the QPU's capability to quickly find promising solutions, expanding this property to a wider range of input types and sizes than would otherwise be feasible.

- NL-Hybrid provides a user-friendly use of quantum resources, allowing the user to model a problem in an intuitive way. This is an advantage in comparison to QUBO, which is the native formulation for QPUs, mainly because translating a problem to this binary formulation is often a challenging task . In fact, inefficient translation can critically affect the performance of the solver.

The existence of NL-Hybrid does however not imply the complete deprecation of the other HSS solvers. Depending on the characteristics of the problem to be solved and the decision variables used, NL-Hybrid may not always be the most efficient solution (for example, BQM or CQM solvers might be more suitable for problems primarily composed of binary variables). The objective for the development of NL-Hybrid was not to replace the existing HSS solvers, but to complement them.

5.7. dwave-hybrid

For quantum annealing developers who prefer to implement their own hybrid approaches to combining quantum and classical computation, D-Wave Quantum Systems offers dwave-hybrid, an open-source Python framework with support for building hybrid workflows that interface with D-Wave QAs..

5.8. Quantum Macro Assembler (QMASM)

Quantum Macro Assembler (QMASM) is a low-level language specific to D-Wave QAs[4].

QMASM fills a gap in the software ecosystem of D-Wave QAs by shielding the programmer from having to know system-specific hardware details while still enabling programs to be expressed at a fairly low level of abstraction. It is therefore analogous to a conventional macro assembler and can be used in much the same way: as a target either for programmers who want a great deal of control over the hardware or for compilers that implement higher-level languages.

Some relevant QMASM language features:

- allows programs to refer to variables symbolically;
- accepts arbitrary values for the function coefficients and automatically maps those onto what is accepted by the underlying hardware;
- provides shortcut syntax for biasing two variables to have the same value (or, respectively, the opposite value);
- supports macros to facilitate code reuse;
- allows sets of macros to appear in a separate file that can be included into a main program routine.

[4] This tool used to be called "QASM" but was renamed to avoid confusion with MIT's QASM, which is used to describe quantum circuits (a different model of quantum computation from what the D-Wave QAs use), and the IBM Q QASM language (now OpenQASM) language, which is also used for describing quantum circuits.

Appendix A - References

[Ezratty 2024] Understanding Quantum Technologies Seventh Edition

[GQI 2025] Quantum Computing Report

[IEEE 2024] D-Wave's Nonlinear-Program Hybrid Solver: Description and Performance Analysis

[NOREA 2025] Quantum Computing Explained

[Shakespeare 1591-1595] Romeo and Juliet

[Wikipedia 2025]

D-Wave Quantum Systems Inc. website

Fujitsu Ltd. website

Hitachi Ltd. website

NEC Corporation website

NTT Corporation website

Qilimanjaro Quantum Tech SL website

Toshiba Corporation website

Appendix B - Acronyms and abbreviations

µm	micrometre
µs	microsecond
µW	microWatt
ADC	Analogue-to-Digital Converter
AI	Artificial Intelligence
aka	also known as
ALU	Arithmetic Logic Unit
AMD	Advanced Micro Devices
API	Application Programming Interface
APP	APPlication
arg	argument
AQC	Adiabatic Quantum Computing
AVaQus	Annealing-based Variational Quantum processors
AWS	Amazon Web Services
bit	binary digit
BQM	Binary Quadratic Model
c	celeritas
C	Capacitance Capacitor
CC	Creative Commons
CCJJ	Compound-Compound Josephson Junction
cjj	compound *J*osephson *J*unction
CIM	Coherent Ising Machine
CMOS	Complementary-Metal Oxide Semiconductor
CNRS	Centre *n*ational de la *r*echerche *s*cientifique

Commun.	Communications
comp.	compound
CPU	Central Processing Unit
CQM	Constrained Quadratic Model
cryostat	*from* cryo *meaning cold and* stat *meaning stable*
CU	Control Unit
DA	Digital Annealer
DAC	Digital-to-Analogue Converter
DARPA	Defense Advanced Research Projects Agency
dc	direct current
DC	Direct Current
DC-SQUID	Direct Current SQUID
DIMACS	Center for Discrete Mathematics and Theoretical Computer Science
DQM	Discrete Quadratic Model
E	Edge Energy
e.g.	exempli gratia
EDFA	Erbium-Doped Fiber Amplifier
EDP	Electronic Data Processing
et al.	et alia
etc.	et cetera
EU	European Union
F	FASTR
FASTR	Frequency And Sensitivity Tunable Resonator
FPGA	Field-Programmable Gate Array
fs	femtosecond
GB	GigaByte

GHz	GigaHerz
GPU	Graphics Processing Unit
GQI	Global Quantum Intelligence
h	*P*lanck constant
ħ	reduced *P*lanck constant (aka *D*irac constant)
$\mathcal{H}$	Hamiltonian
HQS	Honeywell Quantum Systems
HSS	Hybrid Solver Service
HTTP	HyperText Transfer Protocol
HTTPS	HTTP Secure
HVAC	Heating, Ventilation, and Air Conditioning
i	imaginary number
I	*c*urrent
i.e.	id est
IARPA	Intelligence Advanced Research Projects Activity
IBM	International Business Machines
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IFAE	Institut de Física d'Altes Energies
IM	Intensity Modulation
Inc.	Incorporated
Intel	*Integrated Electronics*
IPHT	*L*eibniz Institute of P*h*otonic Technology
IR	Instruction Register
JPL	Jet Propulsion Laboratory
JSON	JavaScript Object Notation
JSS	Job Shop Scheduling

k	kilo
KIT	Karlsruhe Institut für Technologie
kW	kiloWatt
L	inductance
lab	laboratory
Lab	Laboratory
LAN	Local Area Network
LC	*I*nductance-Capacitance
LO	Local Oscillator
LS	Locking Signal
Ltd.	Limited
m	mass
M	Million
max	maximum
Max	Maximum
MaxCut	Maximum Cut
MCP	Maximum Cut Problem
min	minimum
MIT	Massachusetts Institute of Technology
mK	milliKelvin
mV	milliVolt
N	photon *n*umber
NASA	National Aeronautics and Space Administration
NEC	Nippon Electric Company
next gen	next generation
NL-Hybrid	Non*l*inear-*P*rogram Hybrid *S*olver
nm	nanometre
NOREA	Nederlandse Orde van Register EDP-Auditors

NP	Nondeterministic Polynomial
NTT	Nippon Telegraph and Telephone
NumPy	Numerical Python library
OPO	Optical Parametric Oscillator
OpenQASM	Open Quantum Assembly language
OR	Operations Research
OS	Operating System
P	Polynomial Power Pump
PC	Personal Computer Program Counter
PLL	Phase-Locked Loop
PLO	Phase-Locked Oscillator
PM	Phase Modulation
PO	Parametric Oscillator
PPLN	Periodically Poled Lithium Niobate
PPLO	Parametric Phase-Locked Oscillator
ps	picosecond
PUBO	Polynomial Unconstrained Binary Optimization
PZT	Lead Zirconate Titanate ("Lead" is "Plumbum" in Latin)
Q	Qubit
QA	Quantum Annealer Quantum Annealing
QAFS	Quantum Annealing Feasibility Study
QAP	Quadratic Assignment Problem
QASM	Quantum Assembly Language
QCP	Quantum Critical Point
QEO	Quantum-Enhanced Optimization

QFP	Quantum Flux Parametron
Qibocal	Qibo calibration
Qibojit	Qibo just-in-time
Qibolab	Qibo laboratory
Qibosoq	Qibo server on *QICK*
QICK	Quantum Instrumentation Control Kit
QM	Quantum Module
QMASM	Quantum Macro A*ss*e*m*bler
QMC	Quantum Monte Carlo
QML	Quantum Machine Learning
QP	Quadratic Programming
QPU	Quantum Processing Unit
QSDK	Quantum Software Development Kit
QuAIL	Quantum Artificial Intelligence Lab
QUBO	Quadratic Unconstrained Binary Optimization
qubit	quantum bit
rf-SQUID	radio-frequency Superconducting Quantum Interference Device
RFSoC	Radio Frequency System-on-Chip
RSFQ	Rapid Single Flux Quantum
s	second
SA	Simulated Annealing
SaaS	Software-as-a-Service
SAPI	Solver API
SB	Simulated Bifurcation
SBM	Simulated Bifurcation Machine
SFQ	Single Flux Quantum
SHG	Second-Harmonic Generation
SL	Sociedad Limitada
sq	*SQUID*

SQBM	Simulated Quantum Bifurcation Machine
SQUID	Superconducting Quantum Interference Device
SRAM	Static Random-Access Memory
t	time
T	Time
TB	TeraByte
TCO	Total Cost of Ownership
TSP	Traveling Salesperson Problem
U	energy potential
UBC	University of British Columbia
UI	User Interface
UK	United Kingdom
V	Vertex Voltage
VA	Vector Annealing
VCO	Voltage-Controlled Oscillator
VE	Vector Engine
VLSI	Very Large-Scale Integration
VRP	Vehicle Routing Problem
vs	versus
vs.	versus
W	Watt
WAN	Wide Area Network
x	location