ORKUN ŞENSEBAT

QUANTUM ANNEALERS AND PARTIAL DIFFERENTIAL EQUATIONS

THE GATE-BASED ENCODING APPROACH

A thesis submitted for the degree of
Master of Science
to the Faculty of Mathematics, Computer Science and Natural Sciences at the
RWTH Aachen University

Authored at the
Jülich Supercomputing Centre
Forschungszentrum Jülich

February 2024

Authored and published by
Orkun Şensebat, B.Sc.
masterthesis (at) sensebat.com

Examined by
Prof. Dr. Kristel Michielsen
Jülich Supercomputing Centre, Institute for Advanced Simulation
Forschungszentrum Jülich, 52425 Jülich, Germany
k.michielsen (at) fz-juelich.de

Prof. Dr. Stefan Wessel
Institute for Theoretical Solid State Physics
RWTH Aachen University, 52062 Aachen, Germany
wessel (at) physik.rwth-aachen.de

Advisory provided by
Dr. Dennis Willsch
Jülich Supercomputing Centre, Institute for Advanced Simulation
Forschungszentrum Jülich, 52425 Jülich, Germany
d.willsch (at) fz-juelich.de

I hereby declare that I have completed this master thesis independently and without unauthorised assistance from third parties. I have not used any sources or aids other than those indicated. This work has not been previously submitted to any examination body in the same or a similar form.

Additional resources for this thesis at `www.sensebat.com/masterthesis`.

Third-party website links are provided in good faith. The author cannot be held responsible for the persistence of URLs linking to external internet websites referred to in this thesis and does not guarantee the accuracy or appropriateness of any content on those websites.

Aachen, 29 February 2024

Contents

1

Introduction

Solving linear systems of equations represent an insightful opportunity to benchmark problems on quantum annealers. While such equations can be efficiently solved using the Gauss algorithm on classical computers, quantum annealers necessitate the formulation of the problem as a Quadratic Unconstrained Binary Optimization (QUBO) problem, generally NP-hard. We embrace this opportunity by investigating strategies for solving discrete partial differential equations. The QUBO coefficients, derived from encoding the problem variables via fixed-point binary representations, exhibit exponential scaling, which presents a significant challenge for state-of-the-art quantum annealers. We propose a method termed 'gate-based encoding', which conceptualizes the problem in a logical circuit format. This approach effectively addresses the issue of exponential scaling by leveraging ancilla qubits, and leads to enhanced success probabilities as well as the capability to solve problems of marginally greater complexity compared to conventional QUBO formulations.

Quantum annealers and the path towards achieving quantum advantage

Quantum computing has the potential to solve problems that classical computing cannot solve efficiently. These problems are found in important applications, such as process and supply chain optimization, simulation of microscopic systems, and quantum machine learning. The term 'quantum advantage' refers to the capability of performing a useful task more efficiently than a classical computer. As of today, quantum computers are not yet advanced enough to surpass classical computers in solving real-world problems.[1,2]

Researchers are exploring various platforms to achieve quantum advantage, each with its own advantages and challenges. These platforms differ in their operational principles, scalability,

[1] Arute et al., "Quantum Supremacy Using a Programmable Superconducting Processor" (2019)

[2] Kim et al., "Evidence for the Utility of Quantum Computing before Fault Tolerance" (2023).

error rates, and physical requirements.[3,4,5] It is currently uncertain which platform, if any, will ultimately emerge as the leading contender in the race to quantum advantage. The variety of benefits and drawbacks raises the question of whether there may not be a one-size-fits-all solution, but rather a need for customised platforms designed for specific purposes. In certain situations, it may even be advantageous to combine characteristics from different platforms. Currently, quantum computing is still in its early stages. Some experts consider scalability to be an engineering issue rather than a theoretical one.[6,7] Nevertheless, as the field progresses, fundamental obstacles may emerge that hinder advancement and necessitate ongoing innovation and adaptation.

Quantum annealers have the advantage of having a much larger number of qubits available than gate-based systems. Advantage by D-Wave Systems Inc. is currently the largest quantum annealer available. However, quantum annealers are non-universal and are designed specifically for solving Ising or quadratic unconstrained binary optimization (QUBO) problems, which are generally NP-hard.[8,9] They are already capable of solving real-world problems today and can keep up with classical alternatives.[10]

The challenge with linear systems of equations

This thesis analyses the discrete Poisson equation (2.24) as a reference partial differential equation (PDE). The discrete Poisson equation is used to numerically solve electrostatics problems such as those shown in Fig. 1.1. It is expressed as a linear system of equations (LSE), which is not inherently an optimization problem.[11] The value of formulating LSEs as QUBO problems may be questioned, since they are generally NP-hard problems, whereas LSEs can already be solved in polynomial time using the Gauss algorithm on classical computers.[12,13,14] However, we argue that this research does contribute to the path towards achieving quantum advantage. Firstly, creating applications for benchmarking purposes is crucial in order to measure and compare the strengths and weaknesses of each platform across a wide range of problems.

Additionally, scientific progress can be unpredictable. Scientists explore phenomena and may uncover unforeseeable insights. For example, Albert Einstein's theory of general relativity was initially considered a beautiful but abstract concept with little practical use. However, if time dilation is not taken into account, GPS systems would be inaccurate by several kilometers.[15] Nanotechnology allows for the manipulation of matter at the atomic scale, leading

[3] Nielsen and Chuang, *Quantum Computation and Quantum Information* (2010).

[4] Resch and Karpuzcu, "Quantum Computing" (2019).

[5] E.g. superconducting qubits are the largest gate-based platforms as of today, but they require very low temperatures. Trapped ion platforms require intricate laser systems and are difficult to scale, but they offer long coherence times. Photonic platforms operate at room temperature and are naturally robust towards decoherence, but interacting with photons is technically challenging. Topological platforms offer a greater inherent error resistance.

[6] DiVincenzo, "The Physical Implementation of Quantum Computation" (2000).

[7] Gonzalez-Zalba et al., "Scaling Silicon-Based Quantum Computing Using CMOS Technology" (2021).

[8] Wang and Kleinberg, "Analyzing Quadratic Unconstrained Binary Optimization Problems via Multicommodity Flows" (2009).

[9] Pardalos and Jha, "Complexity of Uniqueness and Local Search in Quadratic 0–1 Programming" (1992).

[10] Fox et al., *mRNA Codon Optimization on Quantum Computers* (2021).

[11] See section 2.3 Discrete Poisson equation. The solution of two Poisson problems, a dipole and a plate capacitor in two dimensions, is visualized in Fig. 1.1.

[12] Wang and Kleinberg, "Analyzing Quadratic Unconstrained Binary Optimization Problems via Multicommodity Flows" (2009).

[13] Borle and Lomonaco, "How Viable Is Quantum Annealing for Solving Linear Algebra Problems?" (2022).

[14] Grcar, "How Ordinary Elimination Became Gaussian Elimination" (2011).

[15] Hafele and Keating, "Around-the-World Atomic Clocks" (1972).

to progress in medicine, electronics, and energy development.[16] Lastly, quantum mechanics was originally a theoretical framework, but it now serves as the foundation for quantum computing and is therefore the basis of this thesis.[17,18,19,20]

Conventionally solving linear systems of equations (LSEs) $Ax = \boldsymbol{b}$ on a quantum annealer may not be feasible if the matrix A is not sparse enough or if high precision is required.[21] The QUBO formulation of LSE problems minimises the norm-squared $\|Ax - \boldsymbol{b}\|^2$ after inserting a binary representation.[22] [Borle and Lomonaco, 2022] provides "supporting evidence, based on hardware-agnostic simulations, for specific cases where it may be viable to employ quantum annealing" for solving LSEs. The research shows that if a large number of precision bits are used to encode solutions, the minimum energy gap, which is the energy difference between the ground state and the first excited state, will be too small.[23]

This thesis explains how the QUBO coefficients exhibit exponential scaling and how this poses a problem for current quantum annealers due to auto-scaling.[24] The challenges with LSEs do not stem from having insufficient qubits. Gate-based quantum computing can use the Harrow-Hassidim-Lloyd (HHL) algorithm to solve LSEs. However, the algorithm's effectiveness can also be impeded by the requirement for a large number of precision bits to encode classical data. To ensure the algorithm's success, coherence must be maintained.[25]

Gate-based encoding can boost performance

The study of LSEs on quantum annealers is not as extensive as that of more popular problems. This is because LSEs can already be efficiently solved with classical algorithms. However, this presents an opportunity for new findings. Although it may seem logical to assign a higher weight to leading digits as is discussed in section 3.2.2, doing so exponentially can be problematic. The physical control mechanisms, such as magnetic field biases, may not be able to fully represent the problem. In such cases, the coupling strengths are mainly determined by the largest coefficients, with the exponentially smaller coefficients being insignificant in comparison to the precision supported by the quantum hardware.[26]

We introduce a method named *gate-based encoding* and evaluate its performance by solving the discrete Poisson equation.[27] This method enables the solving of problems with 50% more variables on D-Wave Advantage by eliminating the exponential weights of the QUBO coefficients.[28] To accomplish this, each digit is treated

[16] Eigler and Schweizer, "Positioning Single Atoms with a Scanning Tunnelling Microscope" (1990).

[17] The history of quantum physics can fill an entire novel. An interesting anecdote is how Einstein's explanation of the photoelectric effect was met with skepticism and resistance. Millikan himself called Einstein's hypothesis "bold, not to say reckless" and claimed it "has been pretty generally abandoned". He said these things in 1916, a year after he was the one to verify the hypothesis experimentally.

[18] Klassen, "The Photoelectric Effect" (2011).

[19] Millikan, "A Direct Determination of " h ."" (1914).

[20] Stuewer, "Historical Surprises" (2006).

[21] Borle and Lomonaco, "How Viable Is Quantum Annealing for Solving Linear Algebra Problems?" (2022).

[22] See section 3.1 QUBO formulation in chapter 3 Binary encoding.

[23] See equation 2.2 in section Quantum annealing

[24] See section 2.2.2 Auto-scaling of coefficients.

[25] Duan et al., "A Survey on HHL Algorithm" (2020).

[26] See section 2.2.2 Auto-scaling of coefficients.

[27] See chapter 5 Gate-based encoding.

[28] Figure 6.2 shows an increase on tuned QPU from $n_x = 8$ to 11. This is an increase from 6 variables to 9 variables. See Tab. 6.2.

as a separate set of constraints, corresponding to individual sub-QUBOs.[29] This approach effectively breaks down the entire problem into smaller, more manageable parts, similar to performing written addition and subtraction digit by digit.[30] However, it requires ancilla qubits in the form of intermediate sum bits, as well as carry and borrow bits.[31]

The name 'gate-based encoding' stems from the representation of constraints as logical gates, with the entire LSE being represented by a logical circuit.[32] This approach, or the ansatz idea, may be applicable to other arithmetic problem scenarios and assist in overcoming obstacles.

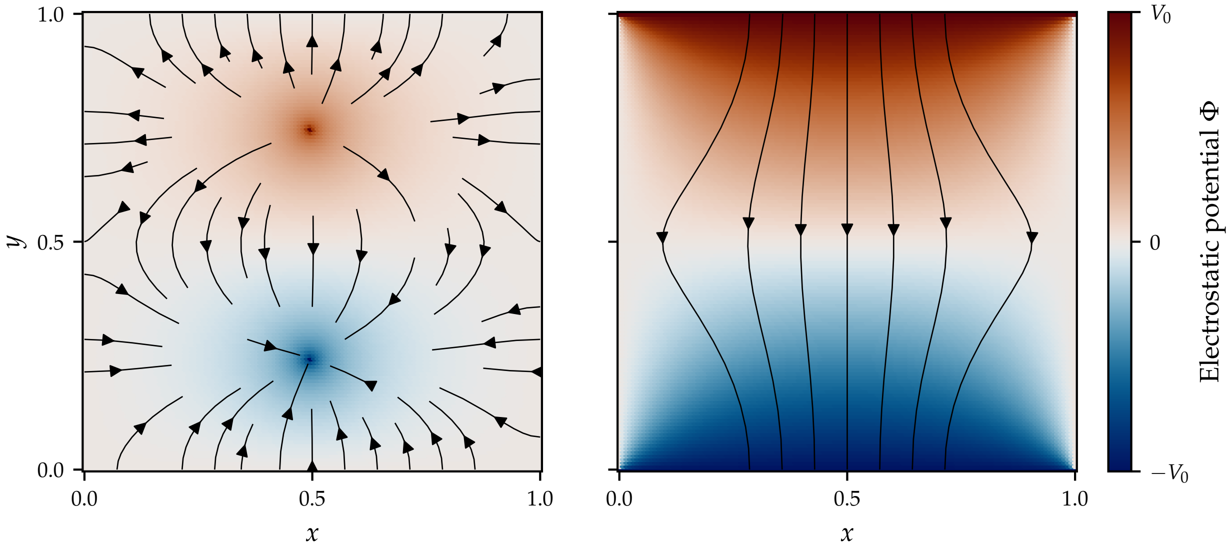


Figure 1.1: Electrostatic potential $\Phi(x)$ of dipole point charges (left) and capacitor (right). Dirichlet boundary conditions are employed with vanishing potential except for the two plates with $\Phi(y = \pm 1) = \pm V_0$. Solutions were obtained using the discrete Poisson equation as derived in section 2.3.3 with a resolution of 128×128 pixels. The electric field lines are the negative discrete gradient of Φ (See section 2.3.1). We solve for Φ using the LU-decomposition solver from scipy.

[29] See section 5.1 Constraint Satisfaction Problem.

[30] See section 5.2 Written arithmetic.

[31] See section 5.1.1 Sum and carry ancillas and section 5.1.2 Borrow ancilla.

[32] See section 5.3 Logical gates.

2

Theory and method

This chapter explains everything needed to understand the results presented in chapters 3, 4 and 5. Section 2.1 below covers the physical foundation of quantum annealing, namely adiabatic computing. It proceeds to explain the concept of quadratic unconstrained binary optimization (QUBO), which is the necessary way to formulate problems to be solved on a quantum annealer, and how QUBO problems are implemented physically using a generalized Ising model.

The subsequent section 2.2 proceeds with intricacies about the hardware implementation on commercially available quantum annealers. We explain how QUBO models are mapped onto the physical topology, how physical qubits are chained together to form logical qubits, how D-Wave internally scales down the given input to its available range of coupling strengths, and the statistical nature of solution candidates returned by quantum annealers.

2.1 Quantum annealing

Every quantum system is described by an operator H named the *Hamiltonian* of the system. H corresponds to the total energy of that system. When we measure a system's total energy, the possible outcomes correspond to the eigenvalue spectrum of H. The *ground state* of a system is the lowest energy possible.[1]

Quantum annealing is a special case of *adiabatic quantum computing*, which relies on the *adiabatic theorem*. This section names the adiabatic theorem and explains how the result of a procedure called *annealing* yields the problem solution denoted as

$$\boldsymbol{x} = (x_0, x_1, \cdots, x_{n-1}) \in \{0,1\}^n, \tag{2.1}$$

where n counts the number of binary variables. Each binary variable x_i corresponds to a qubit i and can be read out through a

[1] Sakurai and Napolitano, *Modern Quantum Mechanics* (2020).

quantum measurement after the annealing procedure concludes. Measuring each qubit corresponds to measuring the total energy of the system, namely to measuring the problem Hamiltonian H_{problem}.

To understand, why the problem solution is encoded into H_{problem}, we need to explain the adiabatic approximation as the fundamental phenomenon enabling quantum annealing.[2] In the adiabatic approximation, the timescale $t_{\max}$ for changes in the Hamiltonian is very large compared to the inverse natural frequency of the state phase factor:[3]

$$\frac{1}{t_{\max}} \equiv \frac{|\langle n;t|\dot{H}|0;t\rangle|}{E_n(t) - E_0(t)} \ll \langle n;t|\frac{\partial}{\partial t}|n;t\rangle \sim \frac{E_n}{\hbar}. \tag{2.2}$$

The time-dependent Hamiltonian describing the dynamics of adiabatic computing is the sum of two terms, the initial Hamiltonian and the problem Hamiltonian:

$$H\left(t\right) = -\frac{A\left(\frac{t}{t_{\max}}\right)}{2} H_0 + \frac{B\left(\frac{t}{t_{\max}}\right)}{2} H_{\text{problem}}. \tag{2.3}$$

The time t develops from 0 to the annealing time $t_{\max}$, and the so-called annealing functions A and B are functions of the normalized anneal fraction $t/t_{\max}$.

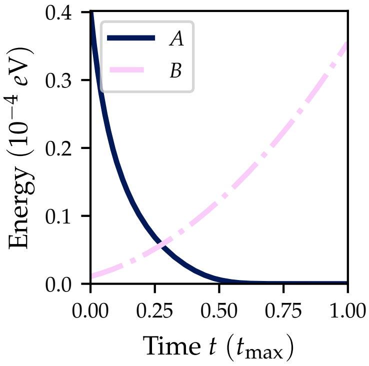


Figure 2.1: Representative annealing functions A and B with respect to time. Anneal schedule taken from D-Wave Advantage performance update *Advantage_system5.4*.

Figure 2.1 shows the annealing schedule. At $t = 0$, the annealing functions must have the property $A(0) \gg B(0)$ and the system can thus easily be prepared in the ground state $\bigotimes_i(|0\rangle + |1\rangle)/\sqrt{2}$ of the initial Hamiltonian

$$H(0) \approx -\frac{A(0)}{2} H_0 = -A(0) \sum_i \left(\frac{1}{2}\right) \sigma_i^{(x)}, \tag{2.4}$$

where $A(0) = 0.40171552 \times 10^{-4}$ eV. Due to the adiabatic theorem, the system remains in the ground state when it is slowly annealed by decreasing $A \to 0$ and increasing $B \to B(1) \gg A(1)$.[4] The anneal results in the Hamiltonian approaching

$$H(t_{\max}) = \frac{B(1)}{2} H_{\text{problem}}. \tag{2.5}$$

If we are able to encode our problem such that the ground state of H_{problem} corresponds to the solution, we can develop an adiabatic machine that can return the solution when we input the corresponding problem Hamiltonian H_{problem}.

$$x = \arg\min_x H_{\text{problem}}(x) \tag{2.6}$$

Quantum annealing and adiabatic quantum computation are

[2] To understand, how the problem solution is encoded into H_{problem}, the astute reader must be referred to the references.

[3] Sakurai and Napolitano, *Modern Quantum Mechanics* (2020).

[4] Albash and Lidar, "Adiabatic Quantum Computation" (2018).

non-universal.[5] On the other hand, *gate-based* quantum computers can run arbitrary quantum algorithms and classical algorithms that can be solved with logical circuits.

2.1.1 *Quadratic unconstrained binary optimization*

In contrast to gate-based quantum computers, quantum annealers are limited to just one problem type in the realm of optimization. They optimize an energy function H by searching for the global minimum[6,7]. Thus, we have to encode our problem model into an energy function as introduced in the last section. We achieve this by stating the problem as a quadratic unconstrained binary optimization (QUBO) problem.[8] Many real-world applications can be stated as QUBO problems,[9] and they are generally NP hard.[10,11] [Lucas, 2014] contains a wide-range of well-known problems in their QUBO formulations.

The energy function $H(\boldsymbol{x})$ in the QUBO formulation is a quadratic form determined by the QUBO matrix Q

$$H(\boldsymbol{x}) = \boldsymbol{x}^T Q \boldsymbol{x}. \tag{2.7}$$

The variables $x_i \in \{0,1\}$ are binary. Since $x_i^2 = x_i$, H can be divided into two sums consisting of linear variables and quadratic variables

$$H(\boldsymbol{x}) = \sum_i Q_i x_i + \sum_{i,j>i} x_i Q_{ij} x_j. \tag{2.8}$$

$Q_i = Q_{ii}$ are the diagonal elements of the QUBO matrix Q.

The constraint $j > i$ is simply a convention and not strictly necessary. Since $x_i Q_{ij} x_j = x_j Q_{ij} x_i$, we can set Q_{ji} to zero for $j > i$ and move the value to $Q_{ij} \rightarrow Q_{ij} + Q_{ji}$. The result is an upper triangle matrix Q that is consistent across individual works, rather than deciding how to distribute $Q_{ij} + Q_{ji}$ over the two transposed matrix elements.

Figure 2.2: Photograph of Ernst Ising in Hamburg, 1925. From [T. Ising et al., 2017]

2.2 *Hardware implementation*

Q is the data structure used to program D-Wave[12] but the underlying system is a generalized[13,14,15] Ising system with Hamiltonian

$$H_{\text{Ising}} = \sum_i h_i \sigma_i^{(z)} + \sum_{i,j>i} J_{ij} \sigma_i^{(z)} \sigma_j^{(z)}, \tag{2.9}$$

where i and j are indices for the variables in the system, h_i are the qubit biases and J_{ij} are the coupling strengths. Both $\boldsymbol{x}^T Q \boldsymbol{x}$ and H_{Ising} can encode the same problem. (A.18) in the appendix

[5] Sometimes they are called analog.

[6] Finnila et al., "Quantum Annealing" (1994).

[7] Kadowaki and Nishimori, "Quantum Annealing in the Transverse Ising Model" (1998).

[8] Kochenberger et al., "The Unconstrained Binary Quadratic Programming Problem" (2014).

[9] Lucas, "Ising Formulations of Many NP Problems" (2014).

[10] Wang and Kleinberg, "Analyzing Quadratic Unconstrained Binary Optimization Problems via Multicommodity Flows" (2009).

[11] Pardalos and Jha, "Complexity of Uniqueness and Local Search in Quadratic 0–1 Programming" (1992).

[12] D-Wave Systems Inc., *D-Wave Documentation*

[13] Originally, Ising models were non-generalized, meaning they only accounted for interactions between nearest neighbors.

[14] E. Ising, "Beitrag zur Theorie des Ferromagnetismus" (1925).

[15] Onsager, "Crystal Statistics. I. A Two-Dimensional Model with an Order-Disorder Transition" (1944).

demonstrates how to convert $\{Q_{in}\}$ to $\{h_i, J_{ij}\}$ by transforming $x_i \to s_i = 2x_i - 1 \in \{1, -1\}$.

D-Wave's qubits are superconducting quantum-interference device (SQUID) flux qubits and are described by the initial Hamiltonian proportional to H_0 with no external applied flux. Once external fluxes $\Phi_i^{(x)}$ are applied to the qubits and an external flux Φ_{CCJJ} is applied to all compound-compound Josephson-junction (CCJJ) structures, the Hamiltonian develops according to $H(t)$ in (2.3). The h_i are proportional to $\Phi_{\mathrm{CCJJ}}(t)\Phi_i^{(x)}(t)$ and the J_{ij} are proportional to the maximum mutual inductance M_{AFM} generated by the couplers between qubits i and j and proportional to the flux $\Phi_{\mathrm{CCJJ}}(t)$.[16,17,18]

2.2.1 QPU topology

Connectivity between qubits is constrained on physical quantum annealers. When mapping a qubit bias h_i to a physical qubit i on a problem with n_h variables, we can only assign a finite number of interaction strengths J_{ij} to qubit i - one for each of this qubit's couplers. In addition, the connections are limited to adjacent physical qubits located next to qubit i rather than allowing connections to arbitrary qubits j. Figure 2.3 shows a unit cell that consists of eight qubits and visualizes the constrained connectivity. The chip arranges the unit cells in a 8 × 8 grid of unit cells as shown in Fig. 2.4. Therefore, numerous problems cannot be directly mapped onto a graph of physical qubits. Such a mapping is referred to as an *embedding*. Moreover, even if such a mapping were possible, discovering an optimal mapping constitutes a challenge in itself and is typically an NP-hard problem.[19,20]

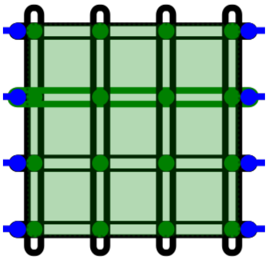


Figure 2.3: Unit cell from Chimera architecture. Shown are eight qubits. Within the unit cell, vertical qubits only couple with horizontal qubits and vice versa. Picture from [D-Wave Systems Inc., *D-Wave Documentation*].

To address this challenge, we combine several qubits $i, j, k, \cdots$ by coupling them to a single logical qubit called a *chain*. We achieve this by assigning couplings $J_{ij}, J_{jk}, \cdots = -J_{\mathrm{chain}}$ between those qubits. These chain couplings are then included in the Hamiltonian

$$H = H_{\mathrm{problem}} - \sum_{i,j \in \mathrm{chain}} J_{\mathrm{chain}} \sigma_i^{(z)} \sigma_j^{(z)}, \tag{2.10}$$

and attempt to correlate the chained qubits so strongly, that they stay synchronized and thus form a logical qubit.

One can estimate a chain strength by assuming a uniform distribution of 'torque' and compensating accordingly. The coupling

[16] D-Wave Systems Inc., *Annealing Implementation and Controls*.
[17] Harris et al., "Experimental Investigation of an Eight-Qubit Unit Cell in a Superconducting Optimization Processor" (2010).
[18] Johnson et al., "A Scalable Control System for a Superconducting Adiabatic Quantum Optimization Processor" (2010).
[19] D-Wave Systems Inc., *D-Wave QPU Architecture: Topologies*.
[20] Bunyk et al., "Architectural Considerations in the Design of a Superconducting Quantum Annealing Processor" (2014).

strength in this uniform torque compensation (UTC) model is

$$J_{\mathrm{utc}} = k\sqrt{\langle J_{ij}^2\rangle}\sqrt{\langle n_i\rangle} \tag{2.11}$$

$$= k\sqrt{\frac{\sum_{i,j>i} J_{ij}^2}{\frac{1}{2}\sum_i n_i}}\sqrt{\frac{\frac{1}{2}\sum_i n_i}{n_h}} \tag{2.12}$$

$$= k\sqrt{\frac{\sum_{i,j>i} J_{ij}^2}{n_h}}, \tag{2.13}$$

where n_i are the number of non-zero interactions of qubit i and n_h is the number of variables. k is a constant and D-Wave quantum annealers set it to $\sqrt{2}$ by default

$$J_{\mathrm{def}} = J_{\mathrm{utc}}|_{k=\sqrt{2}}. \tag{2.14}$$

This compensation is not intended to provide an accurate estimate and D-Wave recommends trying out k-values between $0.5 < k < 2$.[21,22]

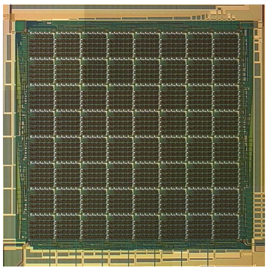


Figure 2.4: Microphotograph of a D-Wave processor chip. Picture shows 8×8 unit cells with edges that are 335 μm long. Picture and information from [Bunyk et al., Aug. 2014].

An optimal embedding and chain strength must be estimated through experimentation, as discussed in chapter 4.[23] The chain strength should be sufficiently high to ensure that the physical qubits that make up the logical qubit reliably maintain the same value during the annealing process. The event of those constituent values changing is called a chain break. In principle, if J_{chain} is sufficiently large relative to $\max_{ij}|J_{ij}|$, then it is guaranteed that the chains do not break.

Although that is true, we should not select an excessively large chain strength due to the scaling down of coupling strengths J_{ij} by auto-scaling, as described in section 2.2.2 below. Chapter 4 confirms that the optimal chain strength, J_{chain}, can be significantly smaller than relative chain strengths J_{chain} greater than one. Therefore, it is necessary to search for the optimal balance between reducing chain breakage probability and not affecting the Ising dynamics of H_{problem}.[24]

2.2.2 *Auto-scaling of coefficients*

The D-Wave QPUs are limited to assigning coupling strengths within a specific range, which corresponds to the maximum achievable physical interaction strengths. For the Advantage QPU, the h-range is between -4 and 4, and the J-range is between -1 and 1. If any values of h_i, J_{ij}, or J_{chain} fall outside of these ranges, D-Wave will proportionally scale down all Ising weights, coupling strengths, and chain strengths until they fit within the range.

Using relative chain strength, obtained by dividing the chain

[21] Djidjev, "Logical Qubit Implementation for Quantum Annealing" (2023).

[22] `uniform_torque_compensation` method in [D-Wave Systems Inc., *D-Wave Documentation*]

[23] Willsch et al., "Benchmarking Advantage and D-Wave 2000Q Quantum Annealers with Exact Cover Problems" (2022).

[24] D-Wave Systems Inc., *Programming the D-Wave QPU: Setting the Chain Strength* (2020).

strength by the maximum QUBO weight, is preferable to using absolute chain strength due to auto-scaling.[25] It is important to be aware of auto-scaling because weights that are too small get scaled down to negligible values smaller than the chip's realizable precision. The QUBO coefficients will always be analysed to ensure that the coupling strengths do not become too weak. Apart from that, auto-scaling occurs in the background and will not affect the presentation of results.

2.2.3 *Statistical nature*

The probabilistic approach is a fundamental aspect of quantum computing. Unlike traditional computing methods that aim to solve problems directly, quantum annealers sample multiple solution candidates. Therefore, they do not guarantee a unique solution but instead offer a range of potential solutions, of which typically only a portion of them are optimal or nearly optimal for a given task.

The potential solutions for landscapes are vast and varied, and it is not always easy to distinguish between optimal and near-optimal solutions. Furthermore, noise and error rates highlight the need for a statistical interpretation of results. To compare performance, we must rely on statistical metrics such as success probabilities. In optimization problems or similar, the statistical nature may even offer advantages, as traditional algorithms may have difficulty escaping local optima, while a quantum annealer could perform a more comprehensive exploration of the solution space.

2.3 *Discrete Poisson equation*

This section explains the partial differential equation (PDE) the work is based on. The choice of this particular problem is explained in chapter 1. In summary, we have adopted a reductionist approach to solving linear systems of equations (LSEs) and open up the result to an audience not too versed with higher level physics problems. The discrete Poisson equation is an ideal subject for this matter, as it is a highly sparse LSE. The formulation of gate-based encoding is a direct result of this reductionist approach and the simple structure of the discrete laplace operator.

Figure 2.5: Drawing of Siméon Denis Poisson. *Public domain*

2.3.1 *The Poisson equation*

In electrostatics, the charge and fields do not vary with time. This explains why the electric field $\boldsymbol{E}$ is the gradient of of the electro-

[25] Governing the physics of the device, one could argue that an absolute chain strength is meaningless without knowledge about the maximum absolute QUBO coefficient.

static potential Φ

$$\boldsymbol{E} = -\nabla\Phi. \tag{2.15}$$

The electrostatic potential Φ is determined through a partial differential equation (PDE)

$$\nabla^2\Phi = -\frac{\rho}{\epsilon_0} \tag{2.16}$$

called Poisson equation. The electrostatic charge density is denoted by $\rho = \rho(\boldsymbol{x})$, and the constant $\epsilon_0 \approx 8.854 \times 10^{-12}$ is called the permittivity of free space.[26]

2.3.2 *Boundary conditions*

We consider x-domains that span the interval $[0, 1]$ throughout the thesis.[27] Dirichlet boundary conditions are employed, and hence (2.16) is solved for $x \in (0, 1)$. $\Phi(0)$ and $\Phi(1)$ are fixed as part of the problem scenario.

The thesis is built upon the Laplace equation with $\rho(x) = 0$. This implies that, in one dimension, the problems correspond to an ideal plate capacitor, wherein the plates at $x = 0$ and $x = 1$ have infinite size and there is no charge in-between. The Laplace equation yields solutions $\Phi(x)$, that are linear, since the second derivative vanishes.

Figure 2.6: Drawings of Peter Gustav Lejeune Dirichlet and Pierre-Simon Laplace. *Public domain*

The scalar potential Φ is not uniquely defined, since a constant can be added to Φ on the right-hand-side of (2.15) without having an effect on physical quantities such as the electric field $\boldsymbol{E}$ or the potential difference $\Phi(x_0) - \Phi(x_1)$. As a result, problems with arbitrary values $\Phi(0) = \Phi_0$ and $\Phi(1) = \Phi_1$ can be reduced to a case where $\Phi(0) = 0$ and $\Phi(1) = V_0 \equiv \Phi_1 - \Phi_0$.

Therefore, the solution for any Laplace problem $\nabla^2\Phi = 0$ is

$$\Phi(x) = V_0 x. \tag{2.17}$$

2.3.3 *Finite difference method*

Computed solutions are necessarily numerical, hence we discretize the space domain onto a finite grid. This is achieved by dividing it into a set of $n_x > 2$ equidistant points with a spacing of $\Delta x \equiv 1/(n_x - 1)$. We refer to n_x as the **x-resolution**, and it is our integer variable that predominantly declares the problem scale.[28]

$$\boxed{n_x = \begin{pmatrix}\text{Resolution}\\ \text{in } x\end{pmatrix}} \tag{2.18}$$

Consequently,

$$x \in [0, 1] \cap (\Delta x\, \mathbb{Z}) = \{0, \Delta x, 2\Delta x, \cdots, 1\}. \tag{2.19}$$

[26] Jackson, *Classical Electrodynamics* (2021).

[27] The results can easily be transformed to arbitrary length scales by defining a unit of length.

[28] Sometimes it can be confusing whether the word *resolution* refers to Δx or n_x. Here, the word is used exclusively as it is used with displays and images. A high resolution means a high number of pixels n_x.

As such, we can obtain a numerical solution in the form of a vector $\mathbf{\Phi}$, namely

$$\Phi(x) \to \mathbf{\Phi} \equiv \begin{pmatrix} \Phi(0) \\ \Phi(\Delta x) \\ \Phi(2\Delta x) \\ \vdots \\ \Phi(1) \end{pmatrix}. \tag{2.20}$$

Our main focus is not on the continuous Poisson equation, nor shall we allocate considerable time for comparing discrete and continuous PDE solutions. Nevertheless, in several of our problem scenarios, we discover that the solution for the discrete Poisson equation corresponds to that of the continuous Poisson equation, because we primarily analyse problems with linear solutions such as (2.17).

We proceed to derive the discrete Poisson equation by using the second-order central difference

$$\begin{aligned} -\frac{\rho}{\epsilon_0} &= \frac{\mathrm{d}^2\Phi}{\mathrm{d}x^2} && (2.21) \\ &= \lim_{\Delta x \to 0} \frac{\mathrm{d}}{\mathrm{d}x} \frac{\Phi(x + \Delta x/2) - \Phi(x - \Delta x/2)}{\Delta x} && (2.22) \\ &= \lim_{\Delta x \to 0} \frac{\Phi(x + \Delta x) - 2\Phi(x) + \Phi(x - \Delta x)}{\Delta x^2}. && (2.23) \end{aligned}$$

By omitting the limit in (2.23) and inserting the discretization (2.19), we can derive the **discrete Poisson equation**

$$\boxed{-\Phi_{k-1} + 2\Phi_k - \Phi_{k+1} = \rho_k/\varepsilon}. \tag{2.24}$$

In this equation, $\rho_k = \rho(k\Delta x)$ and $\Phi_k = \Phi(k\Delta x)$ is the k-th component in (2.20). The ρ_k are expressed in terms of the constant ε, which serves as a unit and can be calculated by the equation

$$\varepsilon \equiv \frac{\epsilon_0}{\Delta x^2} = \epsilon_0(n_x - 1). \tag{2.25}$$

The final step is to convert the linear system of equations (LSE) from (2.24) into matrix form

$$D\mathbf{\Phi} = \boldsymbol{\rho}, \tag{2.26}$$

with D and $\boldsymbol{\rho}$ given by

$$\begin{pmatrix} 1 & 0 & 0 & & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ 0 & -1 & 2 & & 0 & 0 \\ & \vdots & & \ddots & & \\ 0 & 0 & 0 & & 2 & -1 \\ 0 & 0 & 0 & & 0 & 1 \end{pmatrix} \begin{pmatrix} \Phi_0 \\ \Phi_1 \\ \Phi_2 \\ \vdots \\ \Phi_{n_x-2} \\ \Phi_{n_x-1} \end{pmatrix} = \begin{pmatrix} 0 \\ \rho_1/\varepsilon \\ \rho_2/\varepsilon \\ \vdots \\ \rho_{n_x-2}/\varepsilon \\ V_0 \end{pmatrix}. \qquad (2.27)$$

We omit ε from calculations by setting it to one. Our algorithms deal with the ρ_k in units of ε.

2.3.4 *Discrete problem scenario*

After obtaining the discrete Poisson equation (2.27), we initially examine simple problem scenarios before determining the impact of increased complexity. This approach enables us to optimize the encoding to identify ideal scenarios and calibrate the analysis.[29]

The easiest problem we can solve is when $\rho = 0$ and $\Phi(0) = \Phi(1)$ is a constant, which we denote by V_0. Then the differential equation and its solutions take a simple form in (2.28).

$$D\boldsymbol{\Phi} = \begin{pmatrix} V_0 \\ 0 \\ \vdots \\ 0 \\ V_0 \end{pmatrix} \quad \Leftrightarrow \quad \boldsymbol{\Phi} = \begin{pmatrix} V_0 \\ V_0 \\ \vdots \\ V_0 \\ V_0 \end{pmatrix}. \qquad (2.28)$$

This solution is referred to as the *constant solution*.

Due to the gauge invariance of Φ that we discussed in subsection 2.3.2 we can set V_0 to zero, and thus, obtain the *trivial solution*. Obtaining the trivial solution is easier for a human in comparison to the constant solution, but this may not necessarily be the case with a quantum annealer.[30] Nonetheless, deductive verification is necessary during the analysis. Physically speaking, the constant solution models the application of a two-sided bias to an ideal plate capacitor. Therefore, the solution is a constant electrostatic potential Φ with a zero electric field strength $\boldsymbol{E}$ between the plates.

To increase the complexity, we will examine a situation wherein there is a one-sided bias, specifically $\rho = \Phi(0) = 0$ and $\Phi(1) = V_0$.

[29] This becomes clearer at the end of the section. Figure 2.7 enables an analogy with colors. Calibrating the analysis is like testing out a color palette. This approach makes use of the full encoding space the same way as testing as many colors as possible while maximizing the distinctiveness.

[30] Our reductionist approach could make this desirable. After all, we made the claim that problems easy for humans are not necessarily easy for quantum annealers. But this might just be a bit too easy. Or in physicist's terms: A problem with a solution where every qubit is *off* could lead to systematic deviations, or short, a bias. Instead, it seems prudent to try and balance out the solution variables in an attempt to avoid a bias.

As a consequence, the differential equation and its solution become

$$D\mathbf{\Phi} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ V_0 \end{pmatrix} \quad \Leftrightarrow \quad \mathbf{\Phi} = \begin{pmatrix} 0 \\ \frac{1}{n_x-1}V_0 \\ \vdots \\ \frac{n_x-2}{n_x-1}V_0 \\ V_0 \end{pmatrix}. \tag{2.29}$$

W.l.o.g. we select the voltage unit to be

$$[\Phi] = \Delta x V_0 = 1 \tag{2.30}$$

and streamline the problem to

$$\boxed{\mathbf{\Phi} = \begin{pmatrix} 0 \\ 1 \\ 2 \\ \vdots \\ n_x - 2 \\ n_x - 1 \end{pmatrix}}, \tag{2.31}$$

which we call the **balanced solution**.[31] Fig. 2.7 shows the discrete solutions for $3 \leq n_x \leq 9$ with the continuous solution in the background for comparison.

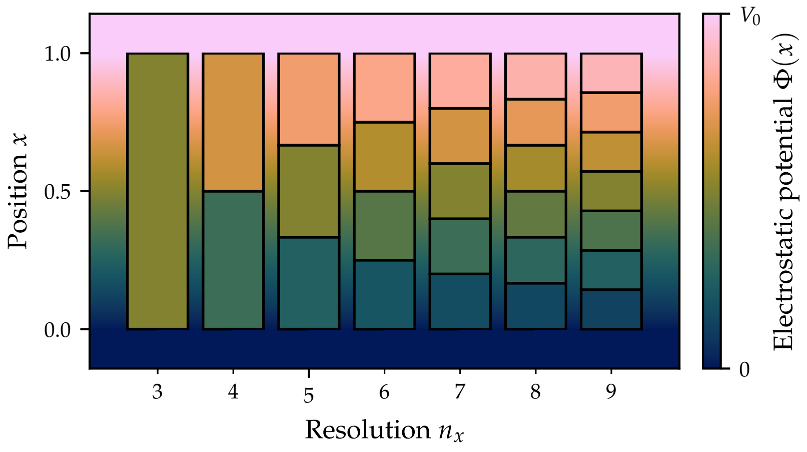


Figure 2.7: Electrostatic potential Φ^i for discrete Laplace problem with resolutions n_x of 3 to 9. The background is colored using the continuous Laplace equation. The top and bottom bands for $x \geq 1$ and $x \leq 0$ are fixed using Dirichlet boundary conditions. The problem corresponds to an infinite plate capacitor with a one-sided bias of V_0 applied to the plate at $x = 0$.

There is no loss of generality by converting the solution of the problem to $\Phi_k \in \{0, 1, \cdots\}$, as we can remove information about the Φ-resolution by choosing a simple unit factor and we can adjust the offset $\{\Phi_k\}$ by adding a constant to Φ. On the other hand, there are some benefits. In these relatively straightforward problem scenarios, we avoid unnecessary complexities such as having to approximate non-integer solutions to finite precision, or signed

[31] This will be the solution we search for throughout the analysis.

encoding which extends the binary encoding in section chapter 3 to support negative values. This enables us to encode the problem and its solution with optimal efficiency - meaning with the fewest number of bits possible.

3
Binary encoding

3.1 QUBO formulation

As explained in section 2.1.1, in order to obtain initial results, it is necessary to translate the problem scenario from section 2.3.4 into a QUBO formulation. Quantum annealers output binary solutions in the form of a bit string. Therefore, the values of Φ_k must be represented by a sequence of binary variables. The resulting bit string can be decoded into an integer solution.

3.1.1 Binary encoding

A binary representation is denoted by a bit string[1,2]

$$(x_\alpha)_2 = (x_{N-1}x_{N-2}\cdots x_1 x_0)_2, \tag{3.1}$$

where N is the number of bits or binary variables $x_\alpha \in \{0,1\}$ that make up the bit string (x_α).

$$\boxed{N = \begin{pmatrix}\text{Number}\\ \text{of bits}\end{pmatrix}} \tag{3.2}$$

N can also be referred to as the *encoding length*. (3.1) defines a way to represent integers $x \in \{0,1,\cdots 2^N - 1\}$ using N binary variables. The binary representation of x is given by

$$x = \sum_{\alpha=0}^{N-1} 2^\alpha x_\alpha, \tag{3.3}$$

where x_α represents the α-th bit of x.

Integer x	Bit string $(x_\alpha)_2$
0	000_2
1	001_2
2	010_2
3	011_2
4	100_2
5	101_2
6	110_2
7	111_2

Table 3.1: List of all integers that can be encoded with a 3-bit string.

3.1.2 Energy function

Latin superscript indices are used to denote the components of a vector or an array, while Greek subscript indices are used to denote binary digits.

[1] The binary number system was studied in Europe centuries ago by Gottfried Leibniz et al. However, related works date back to the times of ancient egypt.

[2] Leibniz, "Explication de l'arithmétique Binaire" (1703).

The discrete Poisson equation $D\boldsymbol{\Phi} = \boldsymbol{\rho}$ can be transformed into an energy function, since it is equivalent to

$$0 = H(\boldsymbol{\Phi}) \equiv \|D\boldsymbol{\Phi} - \boldsymbol{\rho}\|^2. \tag{3.4}$$

We expand the squared norm

$$\begin{aligned}\|D\boldsymbol{\Phi} - \boldsymbol{\rho}\|^2 &= (D\boldsymbol{\Phi} - \boldsymbol{\rho})^T(D\boldsymbol{\Phi} - \boldsymbol{\rho}) &(3.5)\\ &= \boldsymbol{\Phi}^T D^T D\boldsymbol{\Phi} - 2\boldsymbol{\rho}^T D\boldsymbol{\Phi} + \boldsymbol{\rho}^2 &(3.6)\\ &= \sum_{ijk} \Phi^i D^{ki} D^{kj} \Phi^j - 2\sum_{ik} \rho^k D^{ki} \Phi^i + \boldsymbol{\rho}^2 &(3.7)\end{aligned}$$

into a quadratic form. Since a QUBO problem must use binary variables, we insert the binary representation from (3.3),

$$\Phi^i = \sum_{\alpha=0}^{N-1} 2^\alpha \Phi^i_\alpha, \tag{3.8}$$

into (3.7). The result is

$$H(\boldsymbol{\Phi}) = \sum_{ijk}\sum_{\alpha\beta} \Phi^i_\alpha \widetilde{Q}^{ij}_{\alpha\beta} \Phi^j_\beta + \text{const.} \tag{3.9}$$

For multi-indices $i\alpha \equiv (i, \alpha)$ inequalities are defined hierarchically, meaning $j\beta > i\alpha$ if $j > i$ or if $j = i$ and $\beta > \alpha$.

Although the quadratic form is almost in the required form, the $\widetilde{Q}^{ij}_{\alpha\beta}$ contain non-zero coefficients on the lower triangle, which is conventionally avoided. In addition, quadratic and linear terms are mixed. To correct this, we divide the sum into terms for which $i\alpha = j\beta$ and terms for which $i\alpha \neq j\beta$. By using $x^2 = x$ for $x \in \{0, 1\}$ we are able to separate the linear and quadratic terms. This results in

$$\begin{aligned}\boldsymbol{\Phi}^T D^T D\boldsymbol{\Phi} = &\sum_{i\alpha, j\beta > i\alpha} \left(\sum_k 2^\alpha D^{ki} 2^\beta D^{kj}\right) \Phi^i_\alpha \Phi^j_\beta\\ &+ \sum_{i\alpha}\left(\sum_k (2^\alpha D^{ki})^2\right)\Phi^i_\alpha \end{aligned} \tag{3.10}$$

$$2\boldsymbol{\rho}^T D\boldsymbol{\Phi} = \sum_{i\alpha}\left(2\sum_k 2^\alpha D^{ki}\rho^k\right)\Phi^i_\alpha. \tag{3.11}$$

The QUBO Hamiltonian (2.8) can be expressed as

$$H(\boldsymbol{x}) = \sum_{i\alpha} Q^i_\alpha x^i_\alpha + \sum_{i\alpha, j\beta > i\alpha} x^i_\alpha Q^{ij}_{\alpha\beta} x^j_\beta, \tag{3.12}$$

where multi-indices $i\alpha, j\beta$ replace single indices i, j according to $i \to (i, \alpha)$ and $j \to (j, \beta)$. By inserting (3.10) and (3.11) into (3.7) we can identify the QUBO coefficients from (3.12), and get

$$Q^i_\alpha = 2^\alpha \sum_{k=0}^{n_x-1} D^{ki} \left(2^\alpha D^{ki} - 2\rho^k\right) \tag{3.13}$$

$$Q^{ij}_{\alpha\beta} = 2^{\alpha+\beta+1} \sum_{k=0}^{n_x-1} D^{ki} D^{kj}. \tag{3.14}$$

3.2 *Results*

To solve the problem, we assign the coefficients in (3.14) to the QUBO matrix Q, which is used as input to program the quantum annealer.[3]

3.2.1 *D-Wave performance*

We try 10 embeddings and sample 1000 solution candidates each, using default parameters. The embedding with the highest success probability is selected. Figure 3.1 shows the success probabilities with respect to the number of bits N for increasing problem size n_x. The error bars assume an independent binomial distribution and are not included for the sake of mathematical exactness, but rather as a rough measure of the uncertainty. To calculate them, we approximate the observations with a normal distribution based on the central limit theorem and estimate the success probabilities

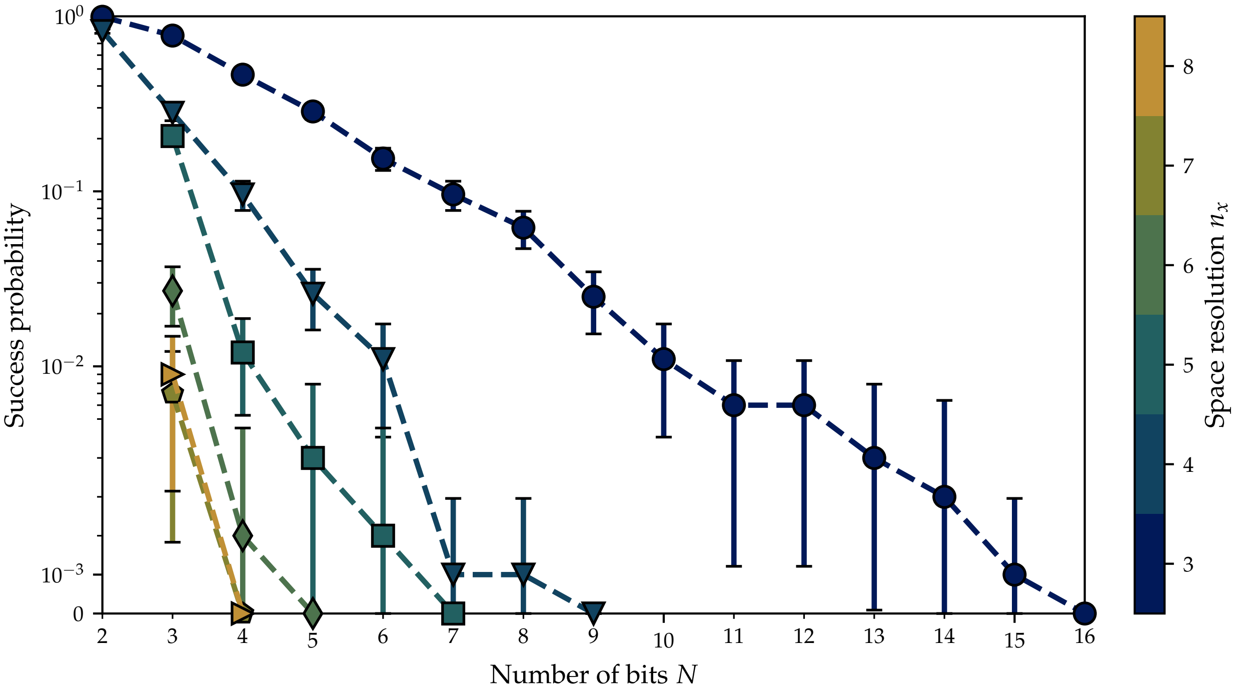


Figure 3.1: Success probabilities for binary encoded Laplace problem with respect to encoding length N. The shown graphs correspond to increasing number of grid points n_x in the space discretization. We sampled for 10 embeddings and selected the largest success probability. We solved for the balanced solution $\Phi_k = k$. Lines are guides to the eye.

[3] D-Wave maintains a python interface that takes the QUBO coefficients as an input.

using

$$p \pm z\sqrt{\frac{p(1-p)}{n_{\text{samples}}}}, \qquad (3.15)$$

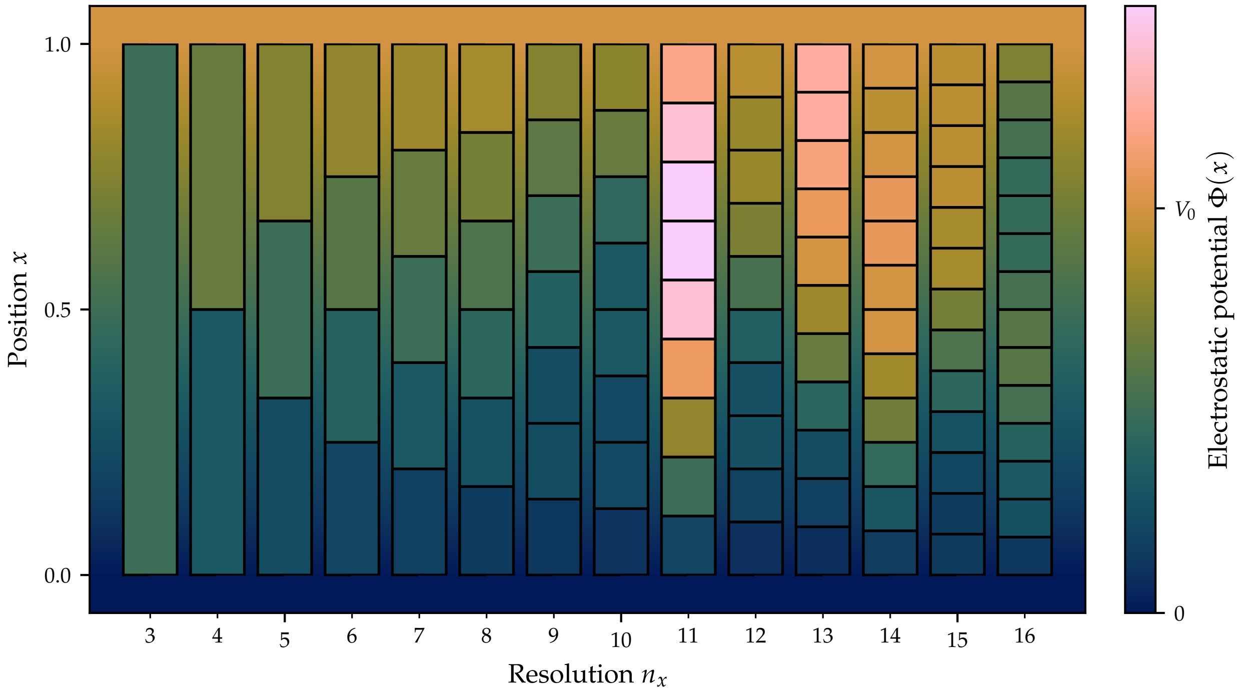
where $p = n_{\text{success}}/n_{\text{samples}}$ represents the estimator for the success probability and n_{samples} is the number of samples. z is the quantile for a 95% confidence interval, approximately equal to 1.960.

The success probabilities in Fig. 3.1 appear to decrease exponentially as the number of bits N increases for each resolution n_x. The QUBO formulation of the discrete Poisson equation is more challenging than classical formulations, which is expected given that the QUBO formulation is NP-hard. The aim is to identify enhancements to this challenging problem type, regardless of existing efficient classical methods. In section 3.2.3, the data is plotted against the number of binary variables, which essentially refers to the number of logical qubits required to solve the problem. As logical qubits are composed of chains of physical qubits, sampling $\Phi(x)$ at only 5 to 10 points in space can be misleading. Instead, the results cover problems with up to 24 logical qubits, which consist of 2 to 3 physical qubits on average.

Problems that do not yield valid solutions with the given sample size can still be analysed. Although energies are non-zero, those with low energies are closer to success on average. Custom metrics, such as the Hamming distance or the sum of squared differences, can also be defined. Samples that do not solve the problem are

Figure 3.2: Best sample of electrostatic potential Φ^i for binary encoded QUBO problem with default parameters as evaluated by energy. Plotted are the solutions for increasing resolutions n_x up to 16 which is as high as 4 bits go. Samples up to a 3-bit resolution of 8 pixels contain correct solutions, whereas 4-bit resolutions beyond that do not. Potentials that correspond to the maximum value that can be encoded are returned by the quantum annealer. These are greater than V_0 and therefore the color bar spans all 16 channels that can be encoded with 4 bits.

The background is colored using the continuous Laplace equation. The top and bottom bands for $x \geq 1$ and $x \leq 0$ are fixed using Dirichlet boundary conditions. The problem corresponds to an infinite plate capacitor with a one-sided bias of V_0 applied to the plate at $x = 0$.



evaluated qualitatively by plotting them in comparison with the reference solution. Figure 3.2 shows the electrostatic potential Φ plotted as as a function of x against increasing x-resolutions n_x. The solutions are correct for resolutions up to $n_x = 8$. However, for higher resolutions such as when $n_x = 11$, the lowest energy samples may have potentials significantly higher than V_0. It is worth noting that the searched solution values lie in the range of 1 to 9, while the annealer can only output non-negative integer results up to 15. Therefore, upward deviation is more likely.

Although the best high-resolution solution candidates are invalid, there is a clear correlation among the solution variables at various points of space x. Many of the Φ^k values appear in batches, as can be seen with n_x values of $11, 15$ and 16. Furthermore, the remaining solutions exhibit a smooth gradient. This is expected, since the discrete Laplace operator aims to make each Φ^k equal with the average of its nearest neighbours. It is worth noting that resolutions such as $n_x = 12$ and $n_x = 15$ may produce samples that appear to be reasonably close approximations of the correct solution.

3.2.2 *QUBO coefficients*

In (3.13) and (3.14) the QUBO coefficients are expressed in a compact form. However, due to the sparse nature of the discrete Laplace matrix D, it is recommended to examine the different cases to gain insight into the structure of the QUBO matrix Q. For i, j not adjacent to the boundaries,

$$Q^{ij}_{\alpha\beta} = \begin{cases} 0 & j < i \\ 0 & j = i,\ \beta < \alpha \\ 6 \cdot 2^{\alpha+\beta} - 2 \cdot 2^{\alpha}(-\rho^{i-1} + 2\rho^{i} - \rho^{i+1}) & j = i,\ \beta = \alpha \\ 2 \cdot 6 \cdot 2^{\alpha+\beta} & j = i,\ \beta > \alpha \\ -2 \cdot 4 \cdot 2^{\alpha+\beta} & j = i+1 \\ 2 \cdot 2^{\alpha+\beta} & j = i+2 \\ 0 & j > i+2 \end{cases} \tag{3.16}$$

is obtained.

Figure 3.3 illustrates the QUBO matrix by visualizing the QUBO coefficients. The equation (3.16) can be related to the visual representation of Q in Fig. 3.3. The lower block-triangle is empty by convention for columns $j < i$. Figure 3.3 illustrates an empty lower triangle for $\beta < \alpha$, and positive upper triangles for $\beta \geq \alpha$ on the block-diagonal $i = j$. The upper band of blocks where $j = i + 1$ is negative. The next band of blocks where $j = i + 2$ is positive

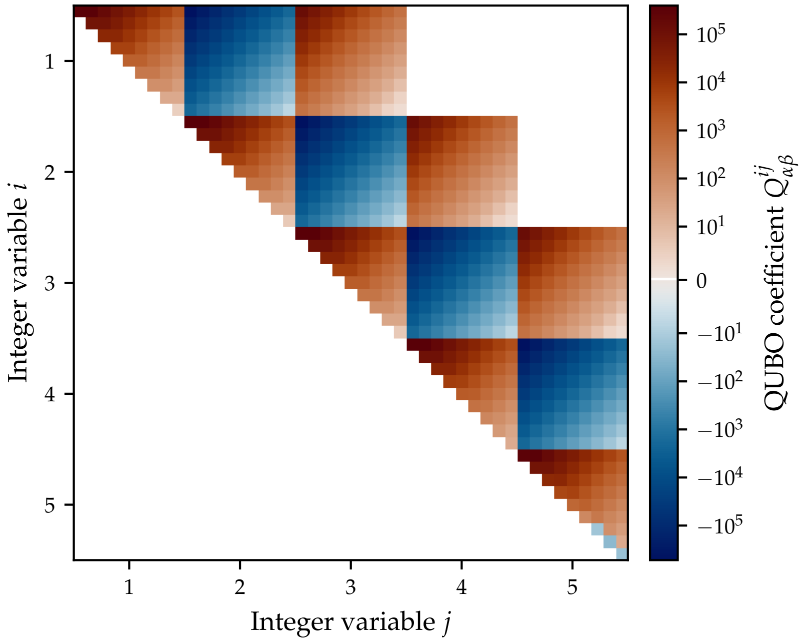


Figure 3.3: QUBO matrix for $n_x = 7$ and $N = 9$. The values of i, j correspond to the x-components of $Q^{ij}_{\alpha\beta}$. QUBO coefficients for $i, j \in \{0, 6\}$ are not present since they are fixed by Dirichlet boundary conditions. The sub-structure corresponds to the binary digit enumeration α, β. Exact zero values are coloured in plain white.

Number of bits N	$\frac{\max\left\|Q^{ij}_{\alpha\beta}\right\|}{\min\left\|Q^{ij}_{\alpha\beta}\right\|}$
1 bit	4
2 bits	16
3 bits	64
8 bits	6.6×10^4
16 bits	4.3×10^9

Table 3.2: Scaling of the largest absolute QUBO coefficients $\left|Q^{i,i+1}_{N-1,N-1}\right|$ divided by the smallest absolute QUBO coefficients $\left|Q^{i,i+2}_{00}\right|$ with respect to a selection of encoding lengths N. Values are rounded to two digits.

again. For columns where $j > i + 2$, the QUBO matrix is zero. The blocks are 9×9 checkerboards coupling the binary digits α, β in substructures.

The quadratic coefficients scale exponentially with respect to $\alpha + \beta$.

$$Q^{ij}_{\alpha\beta} \in O\left(2^{\alpha+\beta}\right) \qquad \max\left|Q^{ij}_{\alpha\beta}\right| \in O\left(4^N\right). \tag{3.17}$$

In Fig. 3.1, only $n_x = 3$ can handle more than 8 bits. This is expected, since for $N = 9$ bits the absolute coefficients $\left|Q^{ij}_{\alpha\beta}\right|$ can be off by a factor of $4 \cdot 2^{16} = 262\,144$. The largest QUBO coefficients contribute so strongly to the energy function H that the smallest QUBO coefficients have a negligible effect. This limits the problem and encoding scope that one can deal with using a binary encoded QUBO formulation of the Laplace problem.

The use of large numbers presents another limitation due to the possibility of overflow when exponentiation is performed. This effect only occurs for encoding lengths N that are beyond currently solvable limits. Nevertheless, caution must be exercised when creating the model. Figure 3.4 illustrates the largest numbers calculated and compares them to the limits of relevant data types.

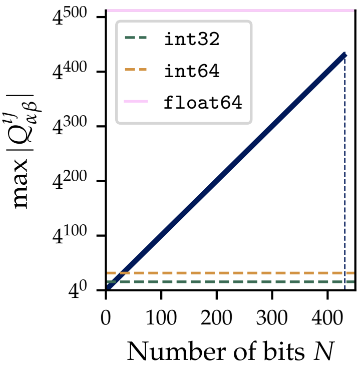


Figure 3.4: Maximum QUBO coefficient for binary model vs. encoding length N in bits. At $N > 430$ we experience overflows with $\max\left|Q^{i,i+1}_{N-1,N-1}\right| = 2 \times 4^N$. 64-bit floats can go up to approximately 1.80×10^{308}, which is indicated in the figure.

3.2.3 *Number of variables and interactions*

By counting the number of QUBO coefficients in (3.16), we can predict the number of non-zero variables and interactions for a given problem size n_x and number of bits N. With $x \in \{0, 1\}$ fixed, we solve for $n_x - 2$ integer variables Φ^i and encode those in N bits each for a total of

$$n_h = (n_x - 2)N. \tag{3.18}$$

The QUBO matrix contains $n_h(n_h - 1)/2$ interaction coefficients in its upper triangle, with many of them being zero. Each of the $n_x - 2$ integer variables Φ^i (excluding the boundary) has N^2 interactions for $j = i + 1$, and the same applies for $j = i + 2$, resulting in another batch of N^2 interactions.[4] However, when $i = j$, only coefficients with $\alpha < \beta$ are non-zero, resulting in a third batch of $N(N - 1)/2 < N^2$ interactions. The number of interactions n_J is the sum of these three batches, resulting in a total of

$$n_J = (n_x - 2)N\frac{5N - 1}{2} = \frac{5N - 1}{2}\, n_h \tag{3.19}$$

interactions. Therefore, the number of interactions per qubit is $(5N - 1)/2$.

This information can be used to test whether the number of variables predominantly determines the success probabilities in Fig. 3.1. To do this, we flatten the (n_x, N) information into a single variable $n_h = (n_x - 2)N$ and plot the same data points in Fig. 3.5. The results indicate that problem scenarios with $n_x > 3$ have higher success probabilities than the problem scenario $n_x = 3$ for the same number of variables n_h. This is due to the fact that the number of bits N is maximal for $n_x = 3$, since it is equal to the number of variables n_h. Therefore, the number of interactions n_J is larger than

i, j	Number of interactions n_J
$j < i$	0
$j = i$	$(N-1)N/2$
$j = i+1$	N^2
$j = i+2$	N^2
$j > i$	0
Total	$(5N-1)N/2$

Table 3.3: Number of interactions n_J per band of blocks j and in total.

[4] There are N combinations for α and N combinations for β with each being arbitrary.

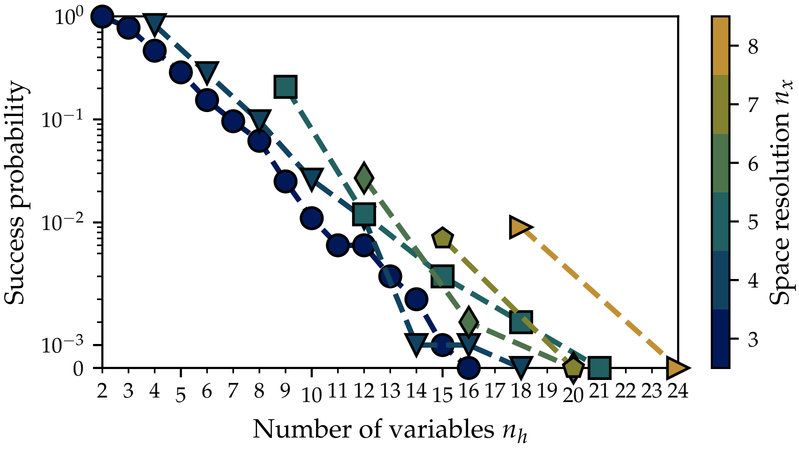


Figure 3.5: Success probabilities for binary encoded Laplace problem with respect to the number of QUBO variables n_h in (3.18). The data points used are the same as those in Fig. 3.1, where we sampled for 10 embeddings and selected the largest success probability. The solution to the problem scenario is $\Phi_k = k$. The lines serve as visual guides.

in other cases with the same number of variables n_h but fewer bits N.

4 Parameter tuning

In section 3 default values were used for the annealing parameters. Although we can anticipate that the exponential scaling of the QUBO coefficients $Q^{ij}_{\alpha\beta}$ will be the limiting factor, irrespective of the annealing parameters, we cannot ascertain the significance of comparisons to other QUBO formulations if we use default parameters. By adjusting the most relevant annealing parameters, we ensure that heuristic estimates do not limit the potential of one formulation while another formulation benefits from a fortunate inference.

We conduct grid scans of two pertinent parameters, namely

1. Chain strength J_{chain},
2. and annealing time $t_{\text{annealing}}$.

It is important to note that tuning these parameters does not eliminate the need for comparing multiple embeddings. Thus, for a given problem, we prepare a set of embeddings and adjust the relevant parameters specific to those embeddings based on the initial conditions, scale, and model used.

4.1 Chain strength

In section 2.2.1, we introduced the concept of chain strengths. We begin by searching for ten minor-embeddings for the given problem at the smallest problem scale and sample solutions with ten linearly spaced relative chain strengths. We then increase the problem scale and repeat the procedure until a solution can no longer be found.

4.1.1 Tuning results

Figure 4.1 illustrates the success probabilities with respect to the relative chain strength J_{chain} for different problem scenarios. The probabilities remain consistent across all chain strengths. Figure 4.1 indicates a decrease in probabilities towards very low chain

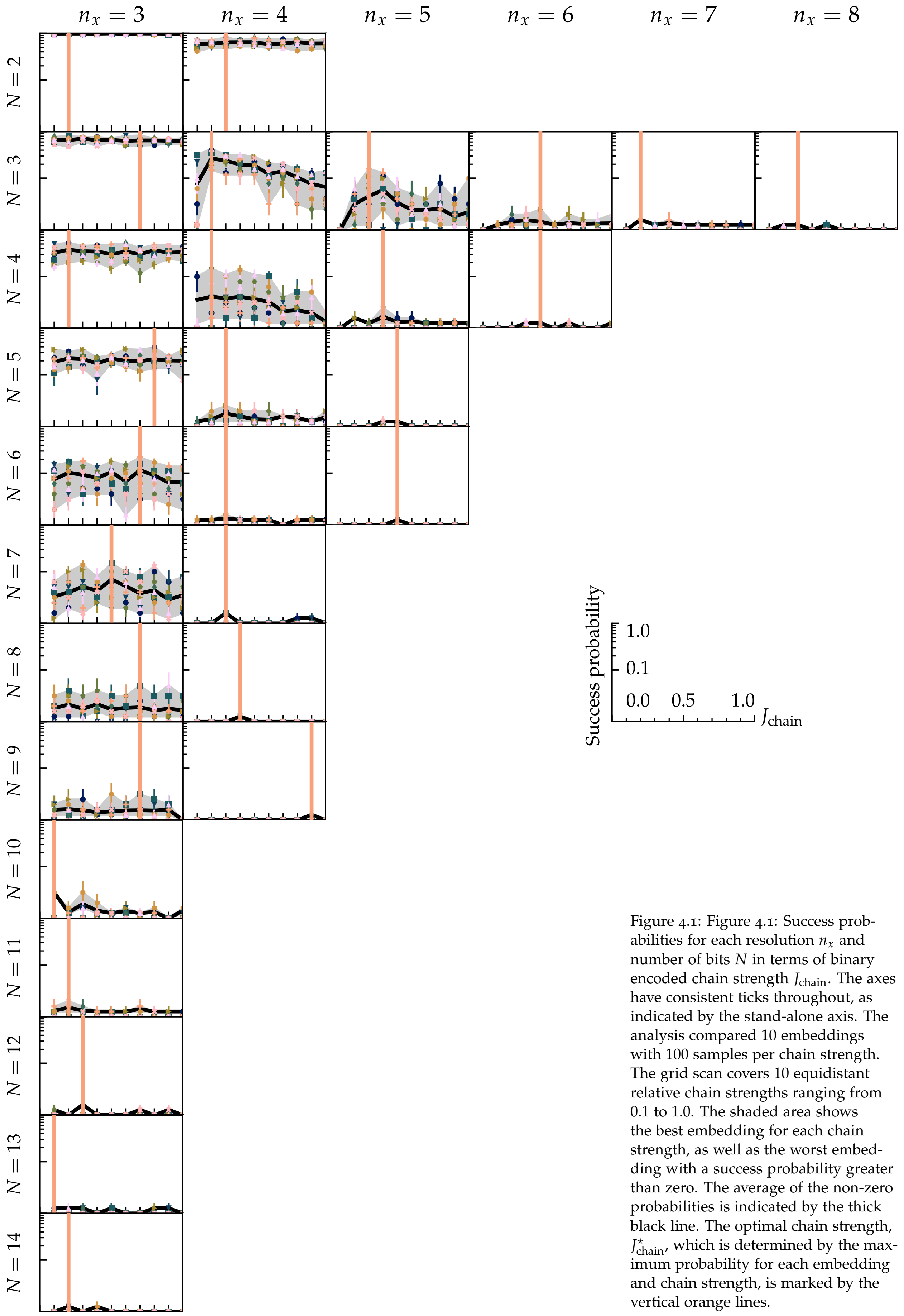


Figure 4.1: Figure 4.1: Success probabilities for each resolution n_x and number of bits N in terms of binary encoded chain strength J_{chain}. The axes have consistent ticks throughout, as indicated by the stand-alone axis. The analysis compared 10 embeddings with 100 samples per chain strength. The grid scan covers 10 equidistant relative chain strengths ranging from 0.1 to 1.0. The shaded area shows the best embedding for each chain strength, as well as the worst embedding with a success probability greater than zero. The average of the non-zero probabilities is indicated by the thick black line. The optimal chain strength, $J^{\star}_{\text{chain}}$, which is determined by the maximum probability for each embedding and chain strength, is marked by the vertical orange lines.

Number of embeddings n_{e}	10
Number of chain strengths n_{cs}	10
Number of samples n_{samples}	100
$\min J_{\mathrm{chain}} / \max \left\| Q^{ij}_{\alpha\beta} \right\|$	0.1
$\max J_{\mathrm{chain}} / \max \left\| Q^{ij}_{\alpha\beta} \right\|$	1.0

Table 4.1: List of parameters for grid scan tuning of the chain strength J_{chain}.

strengths. Further confirmation of this hypothesis requires a more detailed grid scan. However, we will not pursue the analysis of this question as it is known that excessively small chain strengths must result in chain breaks. Our objective is to determine the optimal chain strength rather than the worst.

For each combination of (n_x, N) that results in a non-zero success probability, we estimate the optimal chain strength $J^{\star}_{\mathrm{chain}}$. We assume that the optimal chain strength is independent of both n_x and N. This assumption, although incorrect, is justified as an approximation, which is discussed in detail in section 4.1.2. We have excluded problems with a resolution of $n_x = 3$ as they are near-trivial and inflate the histogram with sparse-bit-solution grid scans, especially for $N \geq 10$. In other words, we need to optimize the apparatus to try and solve the problem for higher resolutions n_x, instead of focusing on the number of bits with near-trivial resolutions.

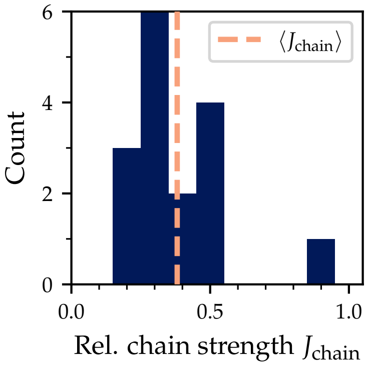


Figure 4.2: Histogram of all optimized chain strengths $J^{\star}_{\mathrm{chain}}$ from tuning procedure. No binning was performed as the tuned chain strengths correspond to the 10 bins. For every problem scenario (n_x, N) with a resolution of $n_x > 3$ pixels, one chain strength was selected. 10 embeddings were tried and 100 samples were produced per chain strength and per embedding. The dashed orange line highlights the arithmetic mean given in equation 4.1.

The optimal chain strength for the Laplace model using binary encoding is

$$J^{\star}_{\mathrm{chain}} = 0.38 \pm 0.18. \tag{4.1}$$

The uncertainty is estimated by calculating the standard deviation of the histogram shown in Fig. 4.2. Although optimizing J_{chain} seems to have little influence on the resulting probabilities, there appears to be a tendency of J_{chain} to be optimal around $[0.2, 0.5]$. The following section demonstrates that D-Wave's estimations for J_{chain} far exceed the optimal chain strengths identified in our analysis.

4.1.2 *Comparison with default chain strength*

Figure 4.3 displays the optimal chain strength for a given problem and compares it to the default chain strength (2.13). It can be seen that the D-Wave estimates are far from optimal. Three points can be emphasized: (1) Analysis shows that more than $N \geq 10$ bits inflate the tuning results. The problem transitions to a superposition with the sufficient bit-length $(n_x, N) = (3, 2)$, as well as a second prob-

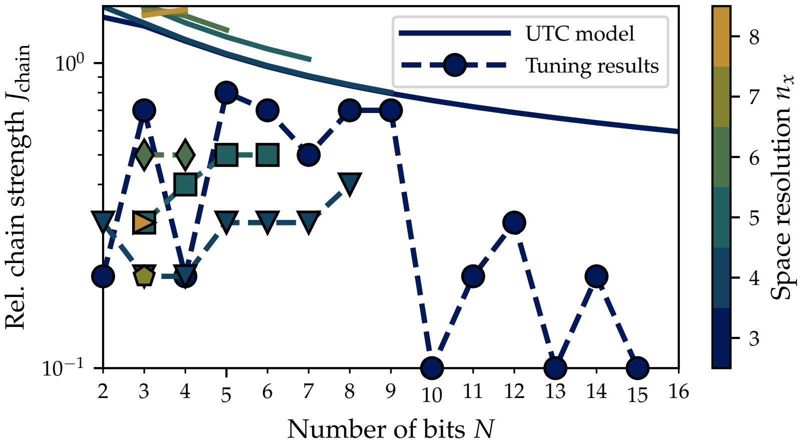


Figure 4.3: Tuned chain strengths J_chain with respect to number of bits N. The lines indicate the UTC model described in section 2.2.1. For every problem scenario (n_x, N), one chain strength was selected. 10 embeddings were tried and 100 samples were produced per chain strength and per embedding.

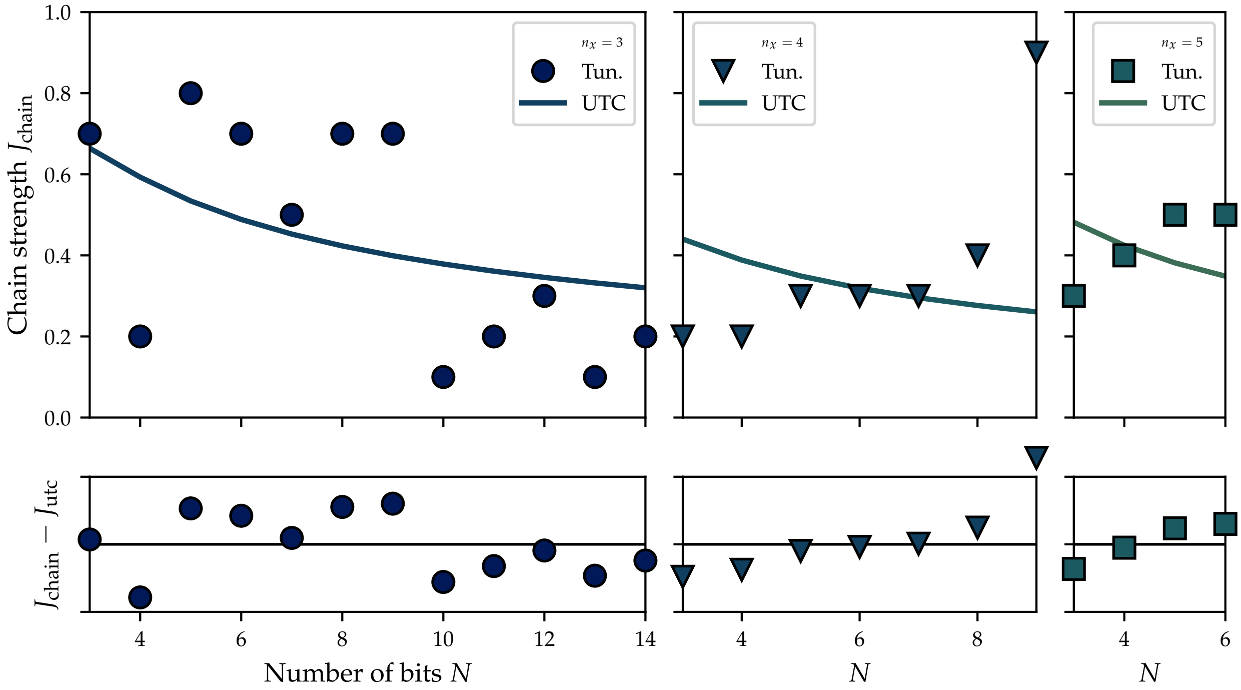
Figure 4.2 shows a histogram of the distribution for $n_x > 3$. The scaled UTC model is fitted to the tuned results for $n_x \leq 5$ in Fig. 4.4

Figure 4.4: Tuned chain strengths J_chain with respect to number of bits N for resolutions up to 5 pixels. The lines indicate the UTC model described in section 2.2.1 fitted to the tuned data. The plots on the bottom show the residuals $J^\star_\mathrm{chain} - J_\mathrm{UTC}$.

For every problem scenario (n_x, N), one chain strength was selected. 10 embeddings were tried and 100 samples were produced per chain strength and per embedding.

lem where all leading digits are set to zero. It is possible that the second part, which requires all leading digits to be zero, may necessitate very small relative chain strengths. The main analysis should not be affected by testing solutions with increasing numbers of bits N. (2) On the other hand, chain strengths estimated with a small N do seem to aggregate towards our optimized chain strength $J^\star_\mathrm{chain}$. This justifies the cut-off selected in the previous section. (3) Most importantly, it appears that the default chain-strengths J_def are consistently too strong.

A regression was performed to estimate the scaling factor and compute a UTC. The results are presented in Fig. 4.4. For $n_x = 3$,



the UTC model appears to be a suitable choice due to the clear negative correlation between $J^{\star}_{\text{chain}}$ and N. However, this does not seem to hold true for $n_x > 3$. The residuals in Fig. 4.4 indicate a significant bias. Our aim is to identify areas that merit further investigation rather than computing a quantitative result.

4.2 *Annealing time*

The time for one anneal, denoted as t_{max}, is indicated by the annealing time in Fig. 2.1. The results of the annealing time scan are included in the appendix in Fig. A.2.[1] We will use default annealing times of 20 μs for the remainder of the thesis.

We perform scans equivalent to the scans for chain strengths.

Table 4.2: List of parameters for grid scan tuning of the annealing time t_{max}.

Number of embeddings n_{e}	10
Number of annealing times n_{at}	10
Number of samples n_{samples}	10
$\min t_{\text{max}} / \mu\text{s}$	20
$\max t_{\text{max}} / \mu\text{s}$	2000

However, there are two differences.

1. Firstly, we do not scan for equidistant annealing times as we did for chain strengths of $0.1, 0.2, \cdots, 1.0$. Instead, we scan for logarithmically spaced annealing times.
2. Secondly, we use only 10 samples per embedding and scanned annealing time.

The annealing procedure remains unchanged within the scope of our analysis, regardless of the chain strength used. While a different chain strength may cause the quantum annealer to operate differently on a physical level, our analysis is not concerned with these low-level hardware intricacies. In fact, the analysis with one chain strength cannot be distinguished from an analysis with a different chain, except by examining resulting metrics such as the frequency of chain breaks or the resulting distribution of successes.

Annealing time, unlike chain strength, is not simply a setting. A higher annealing time means that every read will take longer. Therefore, we must decrease the number of samples n_{samples} from 100 to only 10. This effect is multiplied, because we perform $n_{\text{e}} \times n_{\text{at}} \times n_{\text{samples}}$ reads, one per embedding per scanned annealing time per individual read. Using a larger annealing time may lead to higher success probabilities *per read*, but not necessarily per amount of QPU time.

[1] This is because including them in the main text does not offer significant benefits and may distract from more important points.

This slowdown is important to consider as it can impact the depletion of valuable QPU time and lead to longer analysis times, potentially even requiring the splitting of jobs into multiple repetitions. However, it is important to note that the total slowdown is not proportional to the annealing time t_{max}. Increasing the annealing time by a factor of ten does not result in ten times the amount of QPU access time or run-time due to overhead. Every job has a programming time T_{p} with overhead Δ added on top. Additionally, there is a required sampling time T_{s} that consists of three procedures per anneal:

1. Annealing time t_{max},
2. Readout time T_{r},
3. And delay time T_{d}.

Each of these procedures contributes n_{samples} times to the overall sampling time T_{s}. The resulting QPU access time T is given by[2]

$$T = T_{\text{p}} + \Delta + T_{\text{s}}(n_{\text{samples}}, t_{\text{max}}, T_{\text{r}}, T_{\text{d}}). \tag{4.2}$$

Although longer annealing times may improve performance per read, the effect is minimal at best. Therefore, we did not investigate the performance with higher sample sizes. It is advisable not to invest significant time in research that pre-emptive analysis suggests may be a dead-end. Further comparisons with different annealing times would also increase the complexity of the analysis.

There is another caveat with longer annealing times. Quantum annealing uses quantum effects and tunneling to traverse the energy landscape and remain in the ground state, thereby discovering the global minimum of the encoded problem. The question arises, whether 2 ms anneals even deserve the prefix 'quantum', since anneals that are very slow may result in the determination of the outcome by decoherence and thermal effects rather than quantum effects. Although it is worth mentioning, this caveat is ultimately irrelevant to the questions we are attempting to answer. The device upon which this research is based is partially a black box. The physicist's main interest is to examine the behaviour of the device and make predictions, rather than determining the cause of its behaviour. Explaining the effects that lead to the observed results may be too complex, especially since we cannot verify or influence the inner workings of D-Wave quantum annealers due to their proprietary nature.

Given the above limitations and drawbacks, and the fact that our initial analysis with a small sample size did not show any sig-

[2] D-Wave Systems Inc., *Operation and Timing*.

nificant improvement, we have decided not to pursue this further. However, the results are included in section A2 in the appendix.

5
Gate-based encoding

We will now reformulate the QUBO in a way that avoids exponential scaling of the QUBO coefficients. We do this by giving a set of independent relations for each variable i and digit α. The relations can be represented by logical gates, each associated with a QUBO, thus breaking the problem into many small QUBO problems.

To accomplish this, we will review equation (3.4). First, we squared the norm and then inserted the binary representations. The resulting squared sum includes the powers of 2^α from the binary representation (3.3) within the QUBO coefficients $Q^{ij}_{\alpha\beta}$. The QUBO Hamiltonian has the structure

$$H = \Big[(2^0[\dots])\Phi^0_0 + \dots + (2^{2N-2}[\dots])\Phi^{n_x-1}_{N-1} \tag{5.1}$$

$$+ (2^0[\dots])\Phi^0_0\Phi^1_0 + \dots + (2^{2N-2}[\dots])\Phi^{n_x-2}_{N-1}\Phi^{n_x-1}_{N-1}\Big]\,. \tag{5.2}$$

The exponential weights within the QUBO coefficients ensure that leading digits Φ^i_{N-1}, Φ^i_{N-2}, etc., which contribute exponentially to Φ^i, are also weighed exponentially in the objective $\|D\mathbf{\Phi} - \boldsymbol{\rho}\|^2$. However, in attempting to solve the LSE exactly, weighing leading digits exponentially encounters physical limitations on today's hardware. Therefore, it is important to consider alternative methods for solving the LSE.

The approach is based on the factorization problem, which requires precise calculations.[1,2] To factorise a number using the QUBO formulation, the equation

$$a \times b = p \tag{5.3}$$

is used. This equation can be considered a linear system of equations with a only one row and one variable, which can be solved using binary written multiplication. [Jiang et al., Dec. 5, 2018] also compared both approaches, one by directly minimizing $(p - a \times b)^2$ and another by using multiplication tables.

[1] Andriyash et al., *Boosting Integer Factoring Performance via Quantum Annealing Offsets* (2016).

[2] Jiang et al., "Quantum Annealing for Prime Factorization" (2018).

Every column of written multiplication forms a set of constraints. Thus, we transform the problem into a Constraint Satisfaction Problem (CSP). The constraints are not exponentially weighed, but rather dependent and connected via ancilla qubits representing intermediate `sum` and `carry` bits used in long multiplication. Although this approach requires additional ancilla qubits, it is important to note that these ancilla qubits do not need to be sampled correctly to obtain a proper solution to the logical problem. Solutions with false ancillas are investigated in section 5.8.

5.1 *Constraint Satisfaction Problem*

The gate-based encoding approach was developed by attempting to extract the powers of 2^α in front of the QUBO coefficients. The idea behind the ansatz is represented by the equation

$$H(\boldsymbol{x}) = \sum_{i\alpha, j\beta \geq i\alpha} x^i_\alpha \left(\cancel{2^{\alpha+\beta+1}} \sum_{k=0}^{n_x - 1} D^{ki} D^{kj} \right) x^j_\beta \tag{5.4}$$

when there is no charge. To achieve the desired outcome, the objective is no longer stated as a single energy function to minimize. Instead, the LSE must be examined row by row, and the objectives associated with each row are minimized individually. Therefore, we begin again from

$$D\boldsymbol{\Phi} = \boldsymbol{\rho}, \tag{5.5}$$

without calculating the norm and squaring it. The system consists of $n_x - 2$ linear conditions, with the boundary conditions determining Φ^0 and $\Phi^{n_x - 1}$.

The aim is to convert the problem into a CSP, consisting of independent constraints that must be satisfied, rather than a single loss function. We concentrate on the i-th components corresponding to the i-th row of D.

$$-\Phi^{i-1} + 2\Phi^i - \Phi^{i+1} = \rho^i \tag{5.6}$$

The binary representation (3.3) is inserted into the equation

$$-\sum_{\alpha=0}^{N-1} 2^\alpha \Phi^{i-1}_\alpha + 2\sum_{\alpha=0}^{N-1} 2^\alpha \Phi^i_\alpha - \sum_{\alpha=0}^{N-1} 2^\alpha \Phi^{i+1}_\alpha = \sum_{\alpha=0}^{N-1} 2^\alpha \rho^i_\alpha \tag{5.7}$$

and the resulting values are sorted by powers of 2^α to yield

$$\begin{array}{rcl} \Big[2^0 \times (-\Phi_0^{i-1} + 2\Phi_0^i - \Phi_0^{i+1}) & & \Big[2^0 \times (\rho_0^i) \\ +\,2^1 \times (-\Phi_1^{i-1} + 2\Phi_1^i - \Phi_1^{i+1}) & = & +\,2^1 \times (\rho_1^i) \\ \vdots & & \vdots \\ +\,2^{N-1} \times (-\Phi_{N-1}^{i-1} + 2\Phi_{N-1}^i - \Phi_{N-1}^{i+1})\Big] & & +\,2^{N-1} \times (\rho_{N-1}^i)\Big]. \end{array} \tag{5.8}$$

It is important to note that the brackets do not define QUBO coefficients. Instead, if the brackets are equal on both sides of equation (5.8), then the LSE is satisfied.[3] The powers of 2 are outside the brackets, which allows the gate-based method to bypass the exponential scaling when formulating the QUBO problem.

There is one caveat: The brackets on the RHS are binary, while the brackets on the LHS,

$$-\Phi_\alpha^{i-1} + 2\Phi_\alpha^i - \Phi_\alpha^{i+1} \in \{-2, -1, 0, 1, 2\}, \tag{5.9}$$

may not necessarily be binary since they are a sum of binary variables. Therefore, the equation (5.8) requires the use of ancilla qubits. Any non-binary brackets equal to 2 will be carried over to the next bracket and negative brackets will need to borrow from the next bracket, exactly as in written addition and subtraction.

However, before we identify the constraints, we first use the identity $2 \times (0\,\Phi_{N-1} \cdots \Phi_2\Phi_1\Phi_0)_2 = (\Phi_{N-1} \cdots \Phi_1\Phi_0\,0)_2$ to simplify equation (5.8) by shifting the $2\Phi_\alpha^i$ terms to the next bracket.

$$\begin{array}{rcl} \Big[2^0 \times (-\Phi_0^{i-1} - \Phi_0^{i+1}) & & \Big[2^0 \times (\rho_0^i) \\ +\,2^1 \times (-\Phi_1^{i-1} + \Phi_0^i - \Phi_1^{i+1}) & & +\,2^1 \times (\rho_1^i) \\ \vdots & = & \vdots \\ +\,2^{N-1} \times (-\Phi_{N-1}^{i-1} + \Phi_{N-2}^i - \Phi_{N-1}^{i+1}) & & +\,2^{N-1} \times (\rho_{N-1}^i) \\ +\,2^N \times (\Phi_{N-1}^i)\Big] & & +\,2^N \times (\rho_N^i)\Big]. \end{array} \tag{5.10}$$

5.1.1 Sum and carry ancillas

In order for the LSE to hold true, the brackets associated with 2^0 in the first row of equation (5.10) must be equal. This results in the equation

$$\Phi_0^{i-1} \oplus \Phi_0^{i+1} \overset{!}{=} \rho_0^i, \tag{5.11}$$

which represents the first constraint for the CSP in each row i. We use the operator $\oplus$ to denote addition modulo 2 as explained in Tab.5.1.

This procedure can be repeated for each digit α. However, if

a	b	carry	sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Table 5.1: Truth table for addition modulo 2 $\oplus$. sum $= a \oplus b$ is the sum-bit and carry $= a \wedge b$ is the carry-bit.

[3] This is not a vector but instead a one-dimensional equation. The brackets are linearly dependent.

$\Phi_0^{i-1} + \Phi_0^{i+1} = 2 = (10)_2$, then the sum is carried over to the next digit $2^1 \times (\cdots)$. Since $\rho_0^i = 0$, it could be argued that both variables are always equal to zero. However, for rows $i = 1$ and $i = n_x - 2$, one of the two variables could be fixed by a boundary condition. Therefore, it is necessary to consider the possibility of the sum carrying over.

Fortunately, it is possible to express this algebraically by introducing `sum` and `carry` ancillary variables that follow

$$\Phi_0^{i-1} + \Phi_0^{i+1} = 2c_1^i + s_0^i \in \{0,1,2\}. \tag{5.12}$$

Using (5.12), we can express the constraint that the next digit ρ_1^i must satisfy as

$$-\Phi_1^{i-1} + \Phi_0^i - \Phi_1^{i+1} - c_1^i \overset{!}{=} \rho_1^i. \tag{5.13}$$

The minus sign in front of c_1^i is a result of the minus sign in $\rho_0^i \overset{!}{=} -(\Phi^{i-1} \oplus \Phi^{i+1})$. The subtrahends can be combined into a `sum` ancilla

$$s_1^i \equiv \left(\Phi_1^{i-1} \oplus \Phi^{i+1}\right) \oplus c_1^i. \tag{5.14}$$

Remember that the sums can carry over to the next bracket 2^2 if non-binary. The equation

$$\boxed{\Phi_\alpha^{i-1} + \Phi_\alpha^{i+1} + c_\alpha^i \overset{!}{=} 2c_{\alpha+1}^i + s_\alpha^i} \tag{5.15}$$

defines the ancillas for `sum` and `carry` for any given value of α. It is worth noting how the sum in (5.15) fits neatly into the two binary variables $c_{\alpha+1}^i$ and s_α^i since it is at most equal to $3 = (11)_2$.

5.1.2 *Borrow ancilla*

Equation (5.13) is now represented as

$$\Phi_0^i - s_1^i \overset{!}{=} \rho_1^i. \tag{5.16}$$

If this constraint is met, the i-th linear equation is satisfied for digits ρ_1^i and ρ_0^i. It is possible to continue this approach digit by digit until all digits have a set of constraints. However, if $\rho_1^i = 1$ and $s_1^i = 1$, then equation (5.16) can only be satisfied if $\Phi_0^i = 2$, which is not a binary value. This is invalid, so the subtraction (5.16) has to borrow from the $2^2 \times (\cdots)$ bracket.

The constraint resulting from this can be expressed algebraically as

$$\Phi_0^i - s_1^i \overset{!}{=} \rho_1^i - 2b_2^i. \tag{5.17}$$

The `borrow` bit, represented by b^i_2, can be used to borrow 2 from the next bracket if $s^i_1 = 1$ and $\rho^i_1 = 1$, thereby changing the RHS from 1 to -1. The equation

$$\boxed{\Phi^i_{\alpha-1} - s^i_\alpha - b^i_\alpha \overset{!}{=} \rho^i_\alpha - 2b^i_{\alpha+1}} \tag{5.18}$$

defines the `borrow` ancilla for any given positive value of α.

5.2 *Written arithmetic*

The constraints are summarised in equations (5.19) to (5.29).

$$\Phi^0_\alpha = \Phi(x=0)_\alpha \tag{5.19}$$

$$\Phi^{n_x-1}_\alpha = \Phi(x=1)_\alpha \tag{5.20}$$

$$\tag{5.21}$$

$$\boxed{\alpha = 0} \qquad \rho^i_0 \overset{!}{=} \Phi^{i-1}_0 \oplus \Phi^{i+1}_0 \tag{5.22}$$

$$\rho^i_0 + 2c^i_1 \overset{!}{=} \Phi^{i-1}_0 + \Phi^{i+1}_0 \tag{5.23}$$

$$\tag{5.24}$$

$$\boxed{\alpha > 0} \qquad s^i_\alpha \overset{!}{=} \left(\Phi^{i-1}_\alpha \oplus \Phi^{i+1}_\alpha\right) \oplus c^i_\alpha \tag{5.25}$$

$$s^i_\alpha + 2c^i_{\alpha+1} \overset{!}{=} \Phi^{i-1}_\alpha + \Phi^{i+1}_\alpha + c^i_\alpha \tag{5.26}$$

$$b^i_1 = 0 \tag{5.27}$$

$$\rho^i_\alpha \overset{!}{=} \left(\Phi^i_{\alpha-1} \ominus s^i_\alpha\right) \ominus b^i_\alpha \tag{5.28}$$

$$\rho^i_\alpha - 2b^i_{\alpha+1} \overset{!}{=} \Phi^i_{\alpha-1} - s^i_\alpha - b^i_\alpha \tag{5.29}$$

The variables can be arranged in a way similar to written addition and multiplication to illustrate the concept. Table 5.2 displays the addition table. The table includes columns for higher digits $\alpha = N$ and even $\alpha = N+1$, since ρ could require more digits than Φ.

	0	Φ_{N-1}	Φ_{N-2}	Φ_1	Φ_0	0
		0	Φ^-_{N-1}	Φ^-_2	Φ^-_1	Φ^-_0
$+$		0	Φ^+_{N-1}	Φ^+_2	Φ^+_1	Φ^+_0
$+$	0	c_N	c_{N-1}	c_2	c_1	
$-$		s_N	s_{N-1}	s_2	s_1	s_0
$-$	b_{N+1}	b_N	b_{N-1}	b_2	b_1	
	ρ_{N+1}	ρ_N	ρ_{N-1}	ρ_2	ρ_1	ρ_0

Table 5.2: Addition table for row *i* of LSE.

For the column with ρ_0, no `sum` ancilla is necessary as $0 \ominus s_0 = 0 \oplus s_0 = \rho_0$.

$$s_0 = \rho_0 \qquad \text{fixed.} \tag{5.30}$$

Additionally, the `borrow` ancilla is unnecessary since $b_1 = 1$ if and only if $\rho_0 = s_0 = 1$.

$$b_1 = \rho_0 \qquad \text{fixed.} \tag{5.31}$$

However, when α equals N, there is no need to perform a sum or account for a new part to carry over to column $\alpha = N+1$, since $s_N = 0 + 0 + c_N$.

$$s_N = c_N \qquad \text{fixed.} \tag{5.32}$$

The last column corresponds to $0 \ominus (0 + b_{N+1}) = \rho_{N+1}$. This is equivalent to

$$b_{N+1} = \rho_{N+1} \qquad \text{fixed.} \tag{5.33}$$

The fixed variables are struck through in Tab. 5.3.

Table 5.3: Addition table for row i of LSE. Fixed variables are struck through.

	0	Φ_{N-1}	Φ_{N-2}	Φ_1	Φ_0	0
		0	Φ^-_{N-1}	Φ^-_2	Φ^-_1	Φ^-_0
$+$		0	Φ^+_{N-1}	Φ^+_2	Φ^+_1	Φ^+_0
$+$	0	$\cancel{c_N}$	c_{N-1}	c_2	c_1	
$-$		s_N	s_{N-1}	s_2	s_1	$\cancel{s_0}$
$-$	$\cancel{b_{N+1}}$	b_N	b_{N-1}	b_2	$\cancel{b_1}$	
	ρ_{N+1}	ρ_N	ρ_{N-1}	ρ_2	ρ_1	ρ_0

When α equals 0, a single binary addition links the variables Φ_0^{i-1}, Φ_0^{i+1}, and the output `carry` c_1^i. The constraint consists of two algebraic equations, but the equation $\Phi_0^{i-1} \oplus \Phi_0^{i+1} = \rho_0^i$ is dependent because ρ_0^i is fixed and the Φ_0-variables are constrained via subtraction at the next digit α. For α values ranging from 1 to $N-1$, there is a binary addition and subtraction that connect the aforementioned Φ variables with `sum`, `carry`, and `borrow` bits. When α equals N, only a subtraction is performed by connecting variables Φ^i_{N-1}, s^i_N and b^i_N. In this case, the algebraic equation connecting the input with the borrow b_{N+1} can be disregarded, as will be explained in section 5.2.1. For each i, there are a total of $4N-2$ constraints and variables, resulting in

$$n_h = (n_x - 2)(4N - 2) \tag{5.34}$$

variables in the QUBO problem.[4] However, carries and borrows may not always be unique for valid, zero-energy solutions, and may swap roles. This suggests the possibility of ancillary constraints that are not independent.

5.2.1 *Signed encoding*

Equation (5.33) demonstrates that b_{N+1} is fixed to the charge density, which is equal to zero in the present problem scenario. This section could be concluded here. However, a question arises: What

[4] Compare with $n_h = (n_x - 2)N$ in section 3.2.3.

if $b_{N+1} = 1$? This would imply borrowing from the 2^{N+1}s. The outcome would be

$$\begin{aligned} \rho = (\rho_\alpha) &= 2^{N+1} \times (0-1) + (\rho_N \rho_{N-1} \cdots \rho_2 \rho_1 \rho_0) \\ &= -2^{N+1} + \sum_{\alpha=0}^{N} 2^\alpha \rho_\alpha. \end{aligned} \tag{5.35}$$

In equation (5.35), the expression -2^{N+1} is obtained by subtracting 1 from the 2^{N+1}-minuend, which is equal to 0, as is common practice in written subtraction. The result requires signed integer binary encoding to express the charge density ρ and ρ_{N+1}. Equation (5.35) corresponds to the two's complement representation of ρ.[5] The most significant bit determines the sign of the number and is sometimes called the sign bit. Although the implementation supports signed integers, for the purposes of this thesis, we will use unsigned integer binary encoding. The problems introduced in section subsection 2.3.4 do not involve charges. Therefore, since $\rho_\alpha = 0$ for all α, $b_{N+1} = 0$ must be fixed. This ensures that the resulting finite difference is never negative in the problem scenarios.

Furthermore, all Φ values are non-negative, provided that the biases $\Phi(0)$ and $\Phi(1)$ are also non-negative. Negative Φ values can only occur when a negative bias Φ^0 or Φ^{n_x-1} is applied. However, even in such cases, the subtraction and b_{N+1} are determined solely by the chosen problem. Thus, we conclude that for any Laplace problem, s_0, b_1, and b_{N+1} are equal to zero, resulting in the subtraction being simplified to the addition table 5.4.

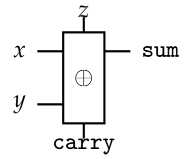
		Φ_{N-1}	Φ_{N-2}	Φ_1	Φ_0	
			Φ^-_{N-1}	Φ^-_2	Φ^-_1	Φ^-_0
$+$			Φ^+_{N-1}	Φ^+_2	Φ^+_1	Φ^+_0
$+$			c_{N-1}	c_2	c_1	
$-$		c_N	s_{N-1}	s_2	s_1	ρ_0
$-$	0	b_N	b_{N-1}	b_2	ρ_0	
	0	ρ_N	ρ_{N-1}	ρ_2	ρ_1	

Table 5.4: Addition table with fixed variables stated explicitly.

When considering the encoding of $\boldsymbol{\rho}$, it is necessary to determine the number of bits needed, given a solution $\boldsymbol{\Phi}$ that requires at least N bits to be encoded. In the case of unsigned integer binary encoding, the value of ρ^i is maximal if Φ^i is maximal and $\Phi^{i-1} = \Phi^{i+1} = 0$, as shown in equation (2.24). In this scenario, $\rho^i = 2 \times \Phi^i$. Thus, ρ is identical to Φ, except for the addition of a trailing zero. Therefore, it is possible to encode the $\boldsymbol{\rho}$-components using at most $N+1$ bits for a given $\boldsymbol{\Phi}$ with N-bit components.

[5] Baugh and Wooley, "A Two's Complement Parallel Array Multiplication Algorithm" (1973).

5.3 Logical gates



x	y	z	carry	sum
0	0	0	0	0
1	0	0	0	1
0	1	0	0	1
1	1	0	1	0
0	0	1	0	1
1	0	1	1	0
0	1	1	1	0
1	1	1	1	1

Figure 5.1: (*Top*) Circuit diagram of full-adder gate. (*Bottom*) Truth table of full-adder gate.

x, y and z are the input bits. The outputs bits are `sum` $= (x \oplus y) \oplus z$ and the `carry` $= (x \wedge y) \vee (y \wedge z)$.

Equation (5.15) defines a full-adder gate algebraically. It can be implemented by cascading two half-adder gates and connecting the `carry` outputs with an `OR`-gate.[6] A half-adder gate computes the two outputs `carry` and `sum` in parallel

$$\texttt{carry} = x \wedge y = \texttt{AND}(x, y) \qquad \texttt{sum} = x \oplus y = \texttt{XOR}(x, y). \tag{5.36}$$

A QUBO formulation can be derived by rearranging the terms in (5.15) to one side, resulting in

$$0 \overset{!}{=} x + y + z - s - 2c, \tag{5.37}$$

and then squaring the expression:

$$0 = (x + y + z - s - 2c)^2. \tag{5.38}$$

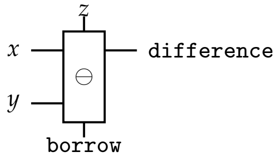
The minimum of the squared expression corresponds to valid values for x, y, z, `sum`, and `carry`. The QUBO form

$$0 = \begin{pmatrix} x & y & z & s & c \end{pmatrix} Q_\oplus \begin{pmatrix} x \\ y \\ z \\ s \\ c \end{pmatrix} \tag{5.39}$$

$$Q_\oplus = \begin{pmatrix} 1 & 2 & 2 & -2 & -4 \\ 0 & 1 & 2 & -2 & -4 \\ 0 & 0 & 1 & -2 & -4 \\ 0 & 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 0 & 4 \end{pmatrix} \tag{5.40}$$

is derived by multiplying out the expression. The process for obtaining the QUBO form of a full subtractor gate is the same, except for a different sign, resulting in

$$Q_\ominus = \begin{pmatrix} 1 & -2 & -2 & -2 & 4 \\ 0 & 1 & 2 & 2 & -4 \\ 0 & 0 & 1 & 2 & -4 \\ 0 & 0 & 0 & 1 & -4 \\ 0 & 0 & 0 & 0 & 4 \end{pmatrix}. \tag{5.41}$$



x	y	z	borrow	diff.
0	0	0	0	0
1	0	0	0	1
0	1	0	1	1
1	1	0	0	0
0	0	1	1	1
1	0	1	0	0
0	1	1	1	0
1	1	1	1	1

Figure 5.2: (*Top*) Circuit diagram of full-subtractor gate. (*Bottom*) Truth table of full-subtractor gate.

x is the bit representing the minuend and y and z are the input bits representing the subtrahends. `difference` $= (x \ominus y) \ominus z$ and `borrow` are the output bits.

The aim of this section is to design a circuit that performs arithmetic operations for each line in D of (2.27). Let us consider row i.

[6] Nielsen and Chuang, *Quantum Computation and Quantum Information* (2010).

Doubling Φ^i is straightforward in binary encoding,

$$\left(2\Phi^i\right)_\alpha = \begin{cases} 0 & \text{for } \alpha = 0 \\ \left(\Phi^i\right)_{\alpha-1} & \text{for } \alpha > 0 \end{cases}. \tag{5.42}$$

Next, we need to sum the Φ^i-neighbors, namely Φ^{i-1} and Φ^{i+1}, and then subtract them from $2\Phi^i$. The circuit idea follows the schematic explanation provided in section 5.2 and is illustrated in Fig. 5.3. The resulting circuit consists of full adder and full subtractor gates, as illustrated in Fig. 5.4.

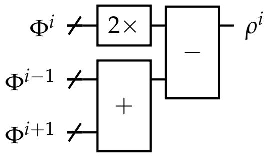


Figure 5.3: Sketch of circuit that represents the CSP. $2\times$-gate is merely a shift in indices. $+$ and $-$ are adder and subtractor gates acting on registers. It depicts row i of the discrete Poisson equation $-\Phi^{i-1} + 2\Phi^i - \Phi^{i+1} = \rho^i$.

5.4 QUBO coefficients

The QUBO matrix is the sum of all sub-QUBO matrices. It is represented by the equation

$$H = \sum_i H_i, \tag{5.43}$$

where $H_i = \sum_\alpha (H^i_\oplus(\alpha) + H^i_\ominus(\alpha))$.

Figure 5.5 visualises the QUBO coefficients. The coefficients no longer exhibit exponential scaling. However, the use of ancilla variables is required, resulting in the QUBO matrix having more than $n_h = (n_x - 2)N$ rows and columns. Although this is a disadvantage of the gate-based encoding approach, it is important to note that the ancilla variables do not need to be sampled correctly to solve the original problem. Section 5.8 investigates false ancillas in more detail. It is worth noting that the gate-based QUBO matrix has less connectivity than the binary encoded QUBO matrix. Therefore, gate-based problems require more qubits, but not necessarily more interactions overall. In any case, for a sufficiently high number of bits N, this results in fewer interactions per qubit.

Now that the QUBO coefficients are in a compact range, we will analyse their distribution. We convert the model to an Ising model, which governs the physics on a hardware level, rather than a QUBO model. The non-zero h_i and J_{ij} values can only take a discrete set of values

$$h_i \in \{-1.5, -1.0, -0.5, 0.5, 1.0, 1.5\} \subset [-4, 4] \tag{5.44}$$

$$J_{ij} \in \{-1.0, -0.5, 0.5, 1.0\} \subset [-1, 1]. \tag{5.45}$$

Upon examining Fig. 5.5, it is apparent that the quadratic QUBO coefficients are limited to two shades of blue and two shades of red. The histogram in Fig. 5.6 displays the distribution of these coefficients. Note that according to equation (A.18) in the appendix, coupling strengths $|J_{ij}|$ of $1/2$ and 1 correspond to $|Q^{ij}|$ values of

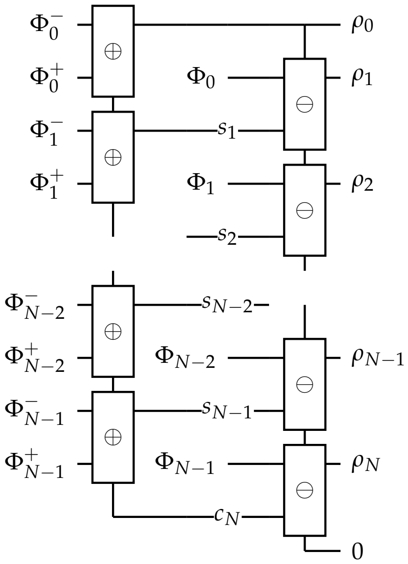


Figure 5.4: Full circuit representing the constraints for row i, namely $-\Phi^- + 2\Phi - \Phi^+ = \rho$, of the discrete Poisson equation. Superscript indices i are ommitted and $i \pm 1$ replaced with $\pm$, since we are exclusively dealing with Φ^i and its adjacent neighbors per row i. Subscript indices refer to the digits α of the unsigned bit string representation belonging to the digit associated with 2^α.

Gates labeled $\oplus$ are full-adder gates explained in Fig. 5.1 and gates labeled $\ominus$ are full-subtractor gates explained in Fig. 5.2. `carry` and `borrow` bits are not labeled to avoid overloading the figure. They belong to the vertical lines on the left below the adders and right below the subtractors respectively.

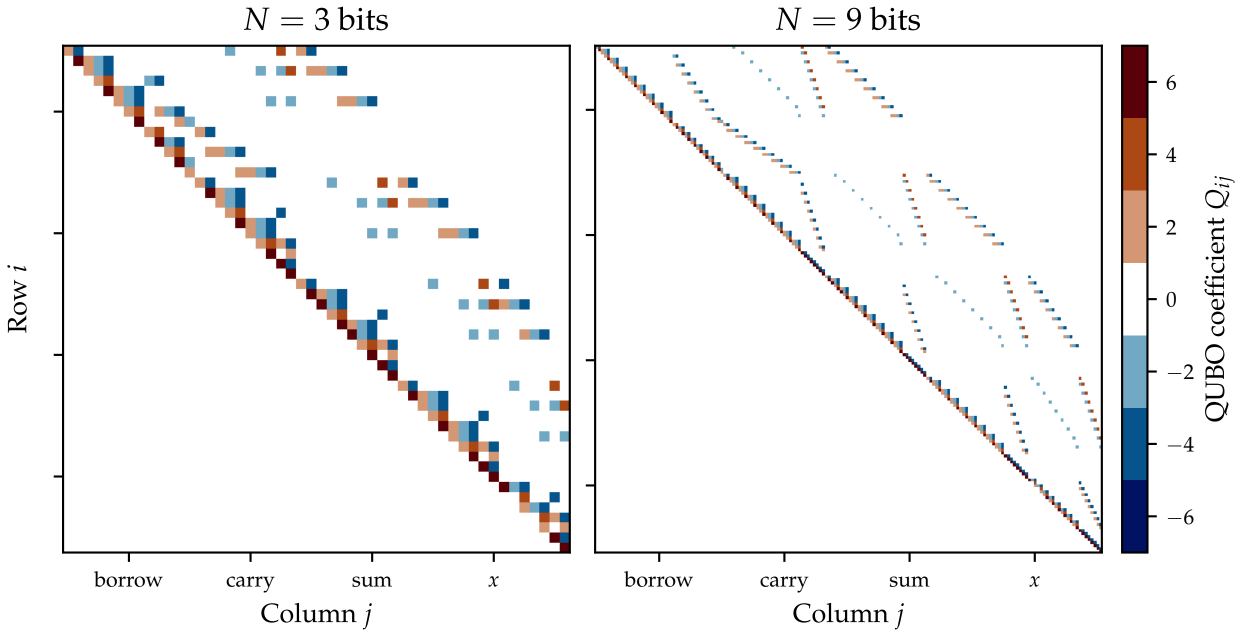


Figure 5.5: Gate-based QUBO matrices Q for a resolution of $n_x = 7$ pixels. Variables encoded in $N = 3$ and 9 bits. Off-diagonal values can only be ±2 or ±4 and only diagonals can have QUBO weights Q_i of ±6. Matrices have $n_h = (n_x - 2)(4N - 2) = 50$ and 170 elements respectively as can be seen in equation 5.34.

2 and 4, respectively. The linear QUBO weights Q^i can also take values of ±6, which corresponds to Ising weights h_i of $\pm 3/2$.

5.5 *Results with default settings*

Ten embeddings were tested and 1000 solution candidates were sampled for each, using default parameters. The embedding with the highest success probability was selected. Figure 5.7 displays the success probabilities in relation to the number of bits N for increasing problem size n_x. For comparison, the binary encoding results from Fig. 3.1 are also included with shaded markers.

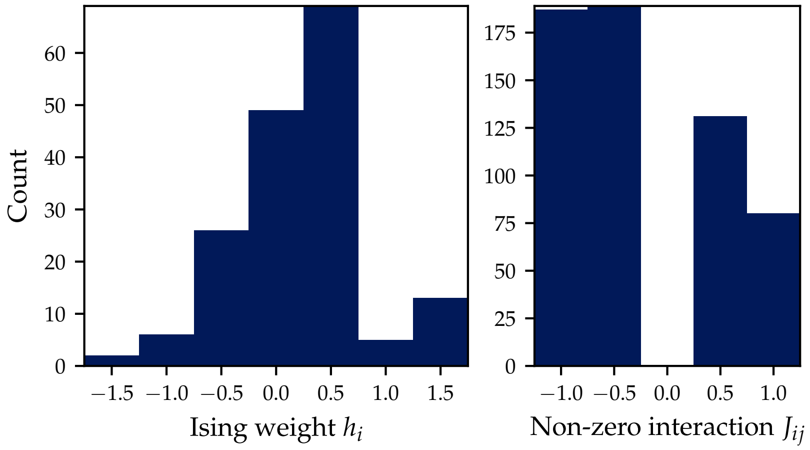


Figure 5.6: Distribution of Ising coefficients h_i and J_{ij} for a resolution of $n_x = 7$ encoded in $N = 9$ bits. No binning was performed as the bars correspond to the possible weights exactly, which can only take those finite values. We do not plot $J_{ij} = 0$, since it would dominate the histogram. These zero quadratic weights do not have any coupling, due to the sparse QUBO matrix Q in Fig. 5.5.

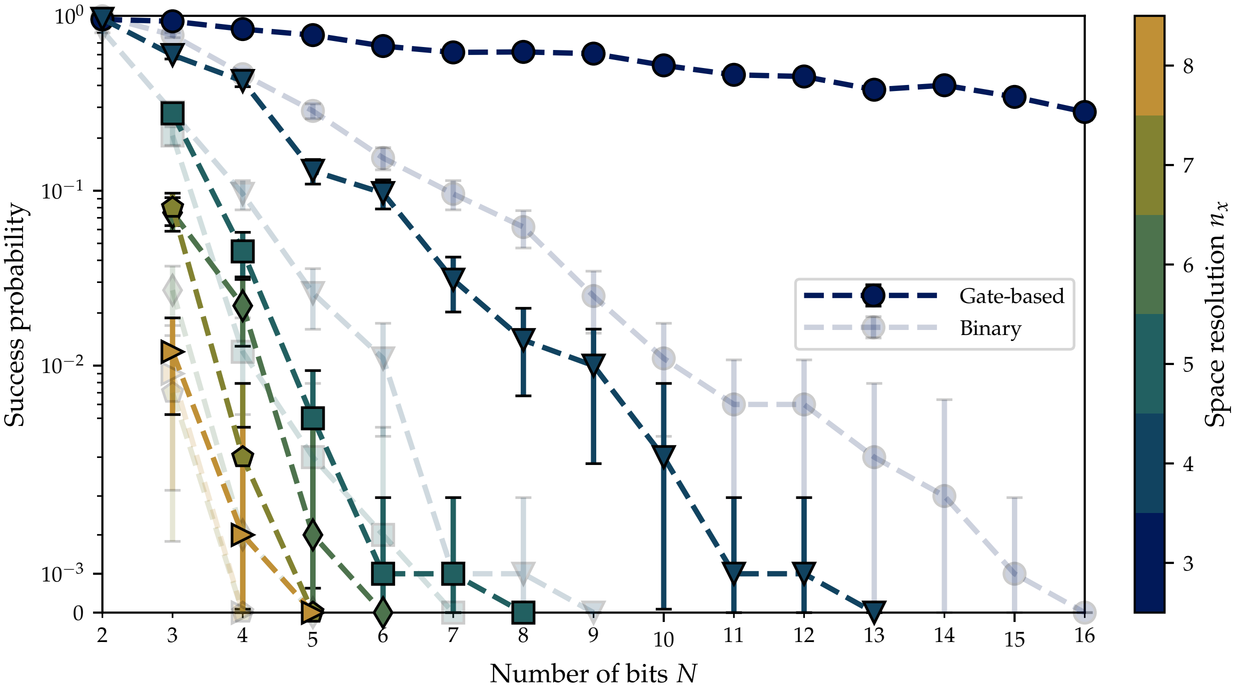


Figure 5.7: Success probabilities in relation to the number of bits N for increasing problem size n_x. 10 embeddings were tested and 1000 solution candidates were sampled for each, using default parameters. The embedding with the highest success probability was selected. We include the data from the binary encoded QUBO formulation in Fig. 3.1 with shaded colors. For the smallest resolution $n_x = 3$ the gate-based data is continued up to very large numbers of bits N and is continued in the appendix in Fig. 5.7 for $N > 16$. The error correction is the same as in equation 3.15: $p \pm z\sqrt{\frac{p(1-p)}{n_{\text{samples}}}}$.

In gate-based encoding, probabilities also decay exponentially with increasing resolution, but they are significantly larger for each resolution compared to binary encoding. It is important to note that gate-based encoding with default values cannot provide a solution for higher resolutions than binary encoding can for the given sample size. However, for low resolutions of $n_x = 3$, we obtain solutions for numbers of bits N that are so large that we have decided not to include them in the figure. To address this, Fig. A.4 in the appendix continues Fig. 5.7 for $N > 16$.

Number of embeddings n_{e}	10
Number of chain strengths n_{cs}	10
Number of samples n_{samples}	100
$\min J_{\text{chain}} / \max \lvert Q_{ij} \rvert$	0.1
$\max J_{\text{chain}} / \max \lvert Q_{ij} \rvert$	1.0

Table 5.5: List of parameters for grid scan tuning of the chain strength J_{chain} for gate-based encoding. The table is identical to Tab. 4.1 that lists tuning parameters for binary encoding.

5.6 *Parameter tuning*

Parameter tuning did not significantly improve binary encoding performance. Although gate-based encoding led to higher success probabilities, this advantage did not translate into higher resolution solutions. To improve the resolution on the same hardware without altering the method, annealing parameters were tuned as described

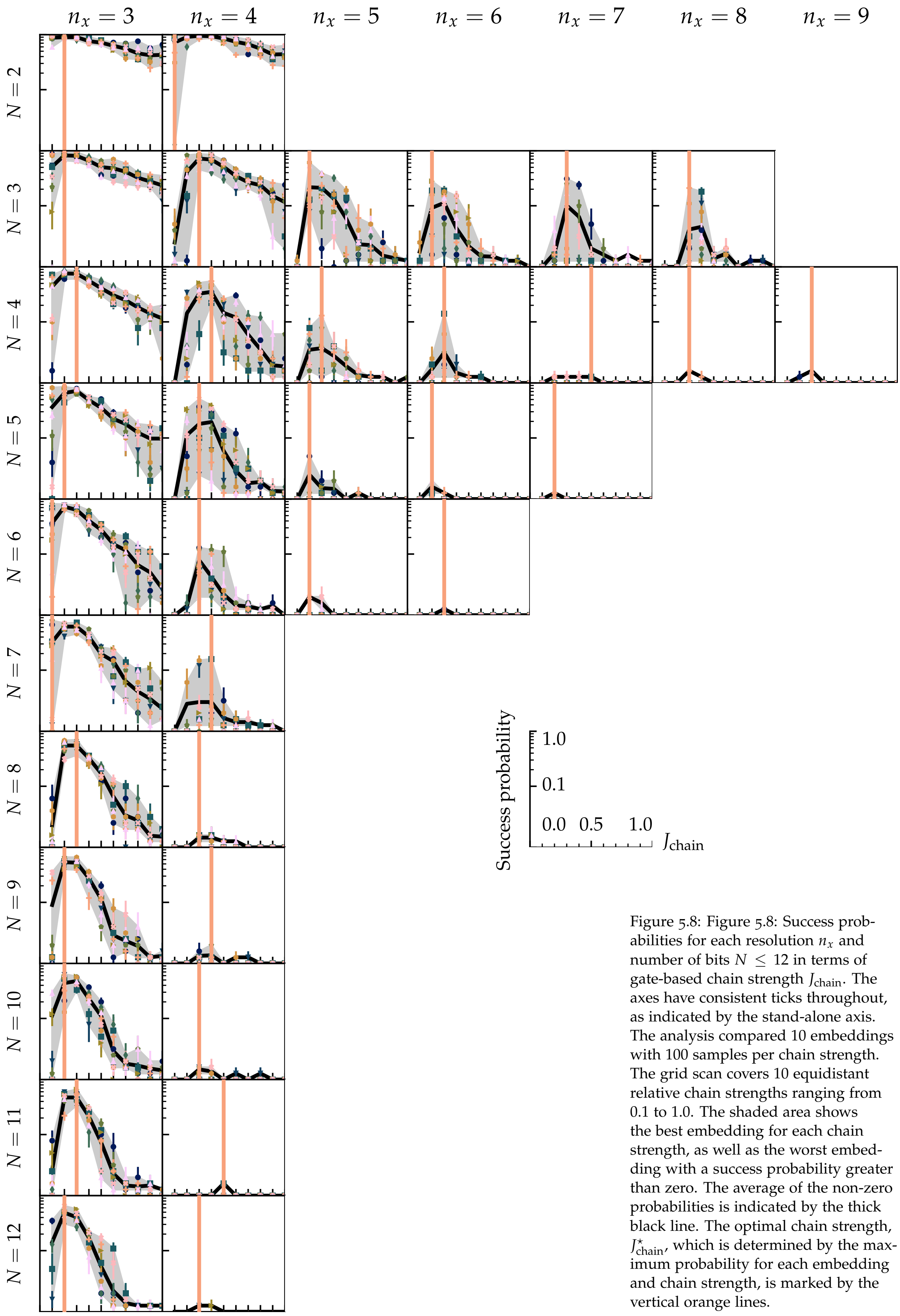


Figure 5.8: Figure 5.8: Success probabilities for each resolution n_x and number of bits $N \leq 12$ in terms of gate-based chain strength J_{chain}. The axes have consistent ticks throughout, as indicated by the stand-alone axis. The analysis compared 10 embeddings with 100 samples per chain strength. The grid scan covers 10 equidistant relative chain strengths ranging from 0.1 to 1.0. The shaded area shows the best embedding for each chain strength, as well as the worst embedding with a success probability greater than zero. The average of the non-zero probabilities is indicated by the thick black line. The optimal chain strength, $J^{\star}_{\text{chain}}$, which is determined by the maximum probability for each embedding and chain strength, is marked by the vertical orange lines.

in chapter 4.

We repeat the tuning procedure by searching for ten minor-embeddings for the given problem at the smallest problem scale. Samples are produced with ten linearly spaced relative chain strengths. The procedure is repeated as the problem scale increases until a solution can no longer be found. Table 5.5 lists the parameters used in the grid scan.

Figure 5.8 shows the success probabilities for various problem scenarios in relation to the relative chain strength J_{chain}. The results differ significantly from those in Fig. 4.1. This time, there is a clear peak around $J_{\text{chain}} = 0.3$, indicating an optimal chain strength $J^{\star}_{\text{chain}}$. This suggests that higher resolution solutions can be achieved compared to default settings.

For each combination of (n_x, N) that results in a non-zero success probability, we estimate the optimal chain strength $J^{\star}_{\text{chain}}$. We assume that the optimal chain strength is independent of both n_x and N. The optimal chain strength for the Laplace model using binary encoding is

$$J^{\star}_{\text{chain}} = 0.30 \pm 0.09. \tag{5.46}$$

The histogram in Fig. 5.9 shows a sharp peak, indicating that the uncertainty for the optimal chain strength $J^{\star}_{\text{chain}}$ is significantly smaller than that discussed in chapter 3.

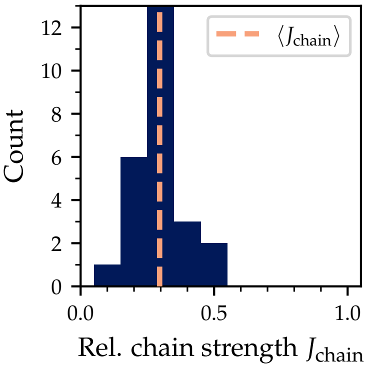


Figure 5.9: Histogram of all optimized chain strengths $J^{\star}_{\text{chain}}$ from tuning procedure. No binning was performed as the tuned chain strengths correspond to the 10 bins. For every problem scenario (n_x, N) with a resolution of $n_x > 3$ pixels, one chain strength was selected. 10 embeddings were tried and 100 samples were produced per chain strength and per embedding. The dashed orange line highlights the arithmetic mean given in equation 5.46.

Figure 5.10 shows the success probabilities achieved by tuning the chain strength according to equation (5.46). The results demonstrate a significant improvement, with correct solutions achieved for resolutions of $n_x = 9$ and $n_x = 11$. However, a correct solution was not obtained for $n_x = 8$, which can be attributed to the sample size. A regression was performed to estimate the scaling factor and compute a UTC. The results are presented in Fig. 5.11. The UTC model is a suitable approximation even for resolutions greater than 3 when using gate-based encoding. The residuals appear to be stochastically distributed.

To conclude the section on tuning, samples that do not solve the problem are evaluated qualitatively by plotting them in comparison with the reference solution. Figure 5.12 displays the electrostatic potential Φ as a function of x against increasing x-resolutions n_x. The solutions are now correct up to a resolution of $n_x = 11$, compared to the previous limit of $n_x = 8$ in Fig. 3.2. The comparison is not the main focus of this chapter and can be found in chapter 6. It is worth noting that for $n_x = 10$, a false solution is obtained, which is, however, a good approximation of the correct solution. Although there are still solution variables with a potential $\Phi > V_0$, the dif-

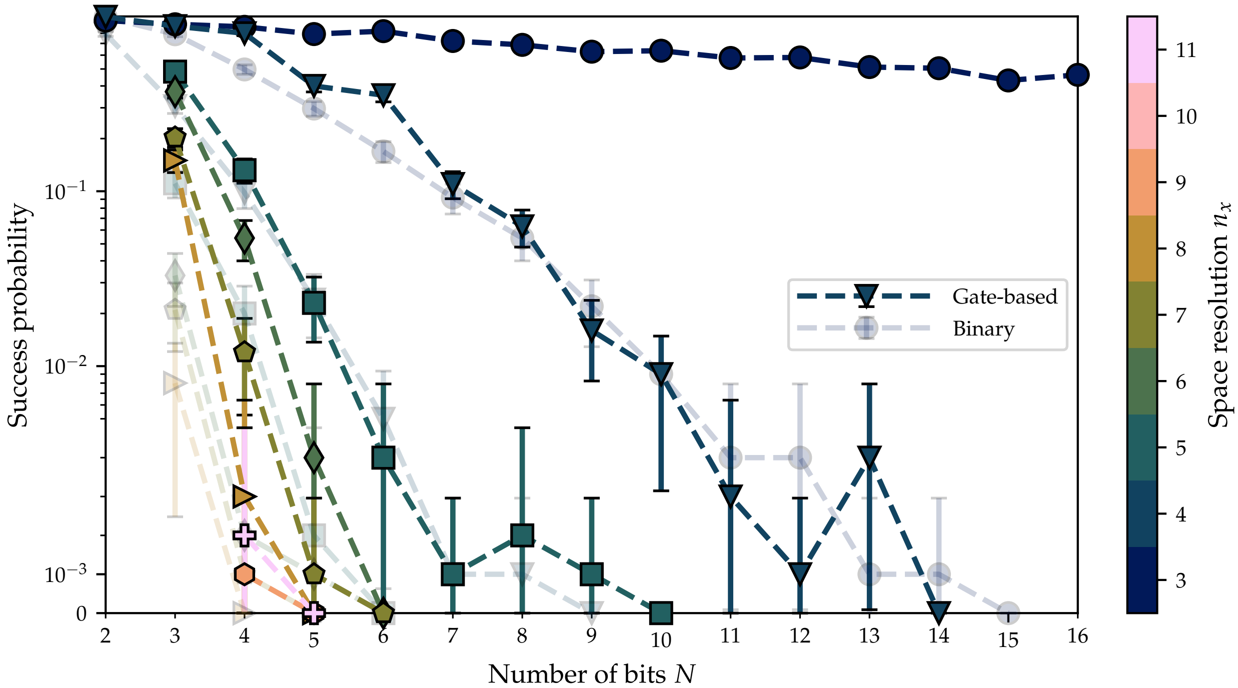


Figure 5.10: Success probabilities in relation to the number of bits N for increasing problem size n_x. 10 embeddings were tested and 1000 solution candidates were sampled for each, using a chain strength J_{chain} of 0.3 taken from equation (5.46). The embedding with the highest success probability was selected. We include the data from the binary encoded QUBO formulation in Fig. A.1 with shaded colors. For the smallest resolution $n_x = 3$ the gate-based data is continued up to very large numbers of bits N. The error correction is the same as in equation 3.15: $p \pm z\sqrt{\frac{p(1-p)}{n_{\text{samples}}}}$.

ference is less significant than was the case with binary encoded QUBOs in Fig. 3.2.

Once again, a clear correlation is observed among the solution variables. The solution values are presented in batches or exhibit a smooth gradient.[7] This is expected since the discrete Laplace operator sets each Φ^k equal to the average of its nearest neighbours, which remains the case regardless of the QUBO formulation. It is worth noting that the sample data seems to be locally correct.

Additionally, the metric for solutions is based on energy, which includes false ancillas. Higher resolutions, such as 12 and 13, may have low energy because most ancilla qubits may be correct. Therefore, there may be higher energy samples in the data that more accurately approximate the correct solutions than the lowest energy samples. Section 5.8 presents a detailed investigation of false ancillas. Fig. 5.13 displays false solutions with the smallest sum of squared differences, a metric that is not affected by ancillary data.

5.7 *The circuit can be simplified*

The circuit formulation was developed from scratch and could potentially be improved in various ways, some of which may not be immediately apparent. The resulting model has evolved organically to its current state. With current knowledge, some simplifications are possible.

[7] See Fig. 3.2.

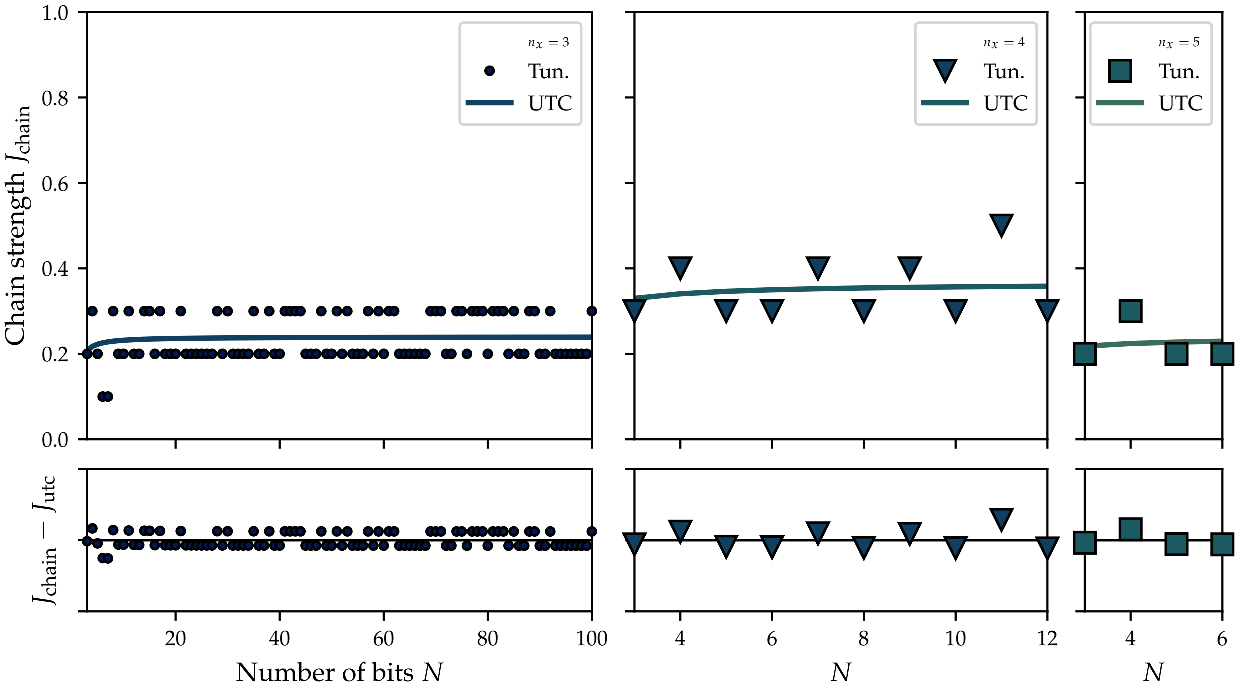


Figure 5.11: Tuned chain strengths J_{chain} with respect to number of bits N for resolutions up to 5 pixels. The lines indicate the UTC model described in section 2.2.1 fitted to the tuned data. The plots on the bottom show the residuals $J^{\star}_{\text{chain}} - J_{\text{UTC}}$.

For every problem scenario (n_x, N), one chain strength was selected. 10 embeddings were tried and 100 samples were produced per chain strength and per embedding.

The circuit can be divided into three components: a trivial component and two non-trivial components.

1. The trivial multiplication by 2,
2. a full-adder gate with `sum` and `carry` bits
3. and a full-subtractor gate with `borrow` bits.

The non-trivial part, which includes the adder and subtractor, can be contracted into a single QUBO. Although the LSE can be split into several sub-QUBOs, merging the adder and subtractor would reduce the final number of QUBOs by half. However, this splitting comes at a cost as the adder and subtractor must be connected using an ancilla qubit, specifically the `sum` ancilla. Unfortunately, combining the gates and reducing the number of ancillas by roughly one third is beyond the already complex scope. However, further research is justified.

Subtractor gates are not necessary to begin with, even though it may seem intuitive to use them, since the equation we are trying to represent with gates is

$$\Phi^i_\alpha - s^i_\alpha = \rho^i_\alpha. \tag{5.47}$$

The investigated cases have a value of $\rho = 0$. It is worth noting that we are not computing the output from an input, i.e. the equation

$$\Phi^i_\alpha - s^i_\alpha = 0 \tag{5.48}$$

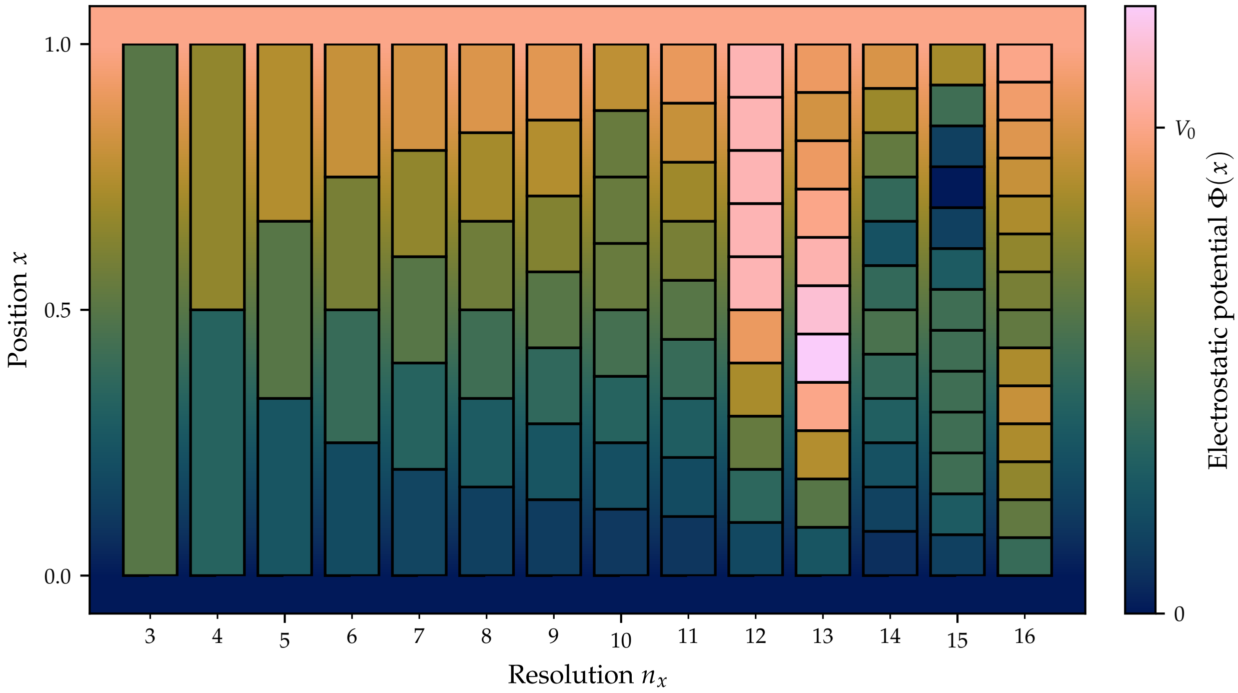


Figure 5.12: Best sample of electrostatic potential Φ^i for gate-based QUBO problem with tuned parameters as evaluated by energy. A chain strength J_{chain} of 0.3 was used with a default annealing time of 20 μs. Plotted are the solutions for increasing resolutions n_x up to 16 which is as high as 4 bits go. Samples up to a resolution n_x of 11 pixels contain correct solutions with the exception of $n_x = 10$. The potential $\Phi(x)$ for a resolution of $n_x = 13$ is takes a maximum value of 1.25 V_0, which is reflected in the adjustments at the color bar.

The background is colored using the continuous Laplace equation. The top and bottom bands for $x \geq 1$ and $x \leq 0$ are fixed using Dirichlet boundary conditions.

is a constraint rather than an input-output relationship.

It is worth noting that the two sub QUBOs per row per digit can be merged into a larger one. However, for the Laplace equation, a larger QUBO is not even necessary, since the equation

$$\Phi^i_\alpha \overset{!}{=} s^i_\alpha \tag{5.49}$$

can be fixed by eliminating the `sum` ancilla and subtractor gate entirely.

Due to time constraints, a detailed investigation of this realization was not possible. Furthermore, the simplifications would make it more challenging to explain the concept because the present circuit directly represents each $[-1, 2, -1]$ row of the discrete Laplace matrix. Despite adhering to the Laplace equation analytically, the constructed *machine* computationally solves the Poisson equation. Charge densities ρ can be introduced into the problem scenario at any time. In theory, a quantum annealer is indifferent to whether the bits are zeroes or ones when solving the Poisson equation.

To elaborate, including non-zero charges a priori does not necessarily increase the difficulty of the problem. In one dimension, the discrete Poisson problem is linear for $\rho = 0$ and piecewise linear between the charges for $\rho \neq 0$. This can even be used to create larger chains without increasing the necessary integer encoding space by alternating positive and negative charges. However, signed encoding, which is implemented based on two's complement and functional, is not used.

There is one important point to make, regardless of whether $\rho = 0$ or not. Subtractor gates are not necessary to model gate-based encoding. If we used signed encoding, we could simply flip the sign bit on adders instead. However, this would waste a valuable bit for simpler problems with $\Phi(0) \geq 0, \Phi(1) \geq 0$ and $\rho \geq 0$.

Even if signed encoding is not used, subtractor gates are not required.[8] The constraint can be rearranged as shown in the equation

$$\Phi^i_\alpha - s^i_\alpha \overset{!}{=} \rho^i_\alpha \quad \rightarrow \quad s^i_\alpha + \rho^i_\alpha \overset{!}{=} \Phi^i_\alpha. \tag{5.50}$$

It is uncertain whether this is an improvement. A full-subtractor and a full-adder QUBO have the same size and coefficients, with only six of the ten off-diagonal signs flipped. Specifically, six coefficients in the 5×5 sub-QUBO are permuted.

5.8 False ancillas

Investigating the alterations to the circuit from the previous section can reveal valuable insights beyond merely improving performance. The `carry` and `borrow` bits are interrelated, as explained below. Converting the subtractor to an adder would eliminate the `borrow` bit and result in two `carry` bits, introducing symmetry to the Hamiltonian. To elaborate, let us focus on the implementation we ended up using. Although the ancilla qubits introduce overhead, they also increase the complexity, which can make matters more challenging. However, this complexity also presents opportunities for further improvement.

We observe 0-energy solutions where the `carry` and `borrow` bits are incorrect. This is because of the symmetries between `carry` and ancilla. Some readers may be familiar with similar intricacies within Feynman diagrams. In some samples, `borrow` and `carry` can exchange roles. A quantitative analysis and mathematical motivation are beyond the scope of this thesis. However, careful validation has demonstrated this to be true. Furthermore, investigating false ancillas, where `carry` and `borrow` swap roles, would likely benefit from circuit simplifications, as outlined in section 5.7.

However, it is possible to observe non-zero energy solutions that yield the correct answer. As the success probabilities approach zero with the given sample sizes, almost correct solutions with a low Hamming distance may be observed. It is worth using the

[8] D-Wave's Ocean-SDK already has adder gates implemented among others. However, subtractor gates are not available.

gate-based formulation to generate samples on an annealer, while manually validating and evaluating the samples and discarding the auxiliary ancilla data altogether. This method can identify valid solutions within the sample that may not be apparent when only considering the energy. Table 5.6 presents the problem scenarios in which correct non-zero energy solutions are found.

Table 5.6: List of x-resolutions that yield solutions for a higher number of bits N when non-zero energy samples are searched. False ancillas are ignored, therefore the performance improves. The x-resolution increases from 8 to 11 pixels for tuned chain strengths.

Resolution n_x	Old N	New N^*	Sampler
7	4	5	Tuned J_{chain} QPU
9	-	4	Tuned J_{chain} QPU
11	-	4	Tuned J_{chain} QPU
20	5	6	Hybrid

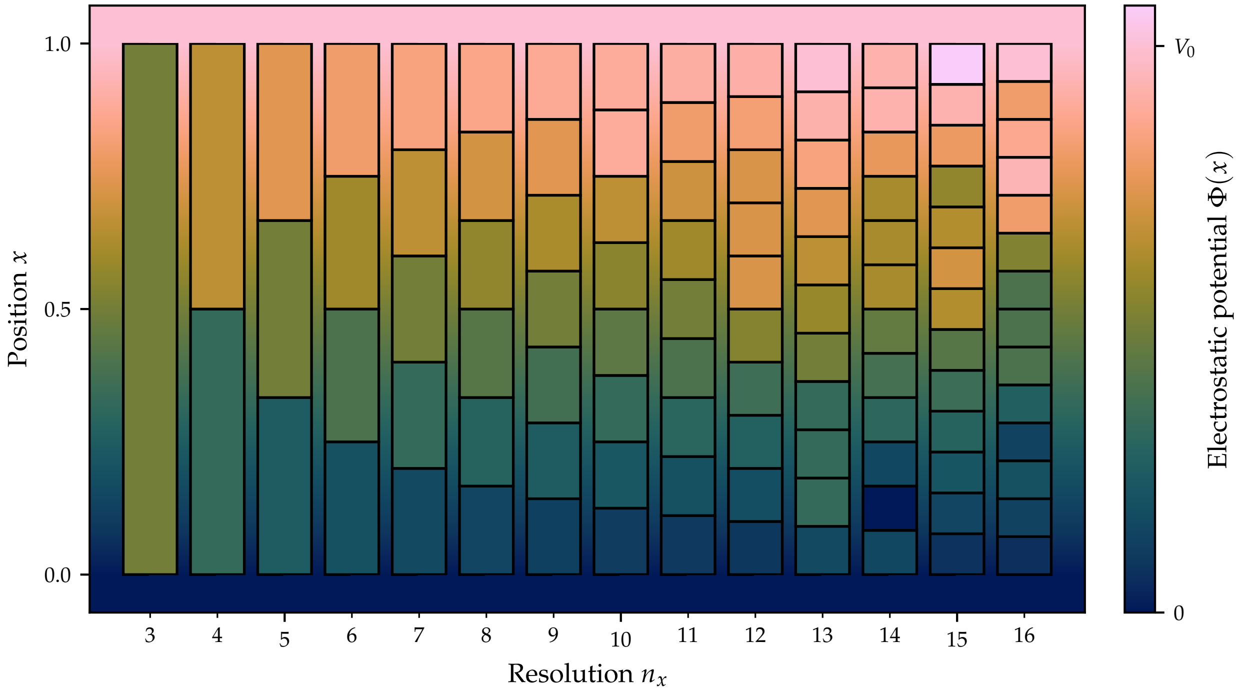


Figure 5.13: Best sample of electrostatic potential Φ^i for gate-based QUBO problem with tuned parameters as evaluated by sum of squared differences χ^2. A chain strength J_{chain} of 0.3 was used with a default annealing time of 20 μs. Samples up to a resolution n_x of 11 pixels contain correct solutions with the exception of $n_x = 10$. The potential $\Phi(x)$ for a resolution of $n_x = 15$ is slightly greater than V_0.

The background is colored using the continuous Laplace equation. The top and bottom bands for $x \geq 1$ and $x \leq 0$ are fixed using Dirichlet boundary conditions. There is no dedicated figure for the binary-encoded counterpart. However, a comparison with the binary-encoded data is given in Fig. 6.4.

Figure 5.13 visualizes false solutions for the smallest sum of squared differences χ^2 as opposed to energy, thus showcasing the best-case potential of the sample set provided by the specific QUBO formulation. It is important to note that the selected solutions remain unchanged for resolutions that we are able to solve. However, for $n_x > 11$, the best samples are much closer to the actual solution than the lowest energy samples. Note that even for binary encoding false solutions show significant improvement when evaluated using the sum of squared differences χ^2 from the correct solution. A fair comparison is presented in Fig. 6.4, as well as a discussion of the caveats.

6 Comparison and discussion

Chapters 3 and 5 analyse binary and gate-based encoding in detail by calculating and comparing the respective success probabilities. In this chapter, the results are combined in summary plots. The focus is on the problem sizes that can be solved given the sample size, rather than the success probabilities for each problem size.

6.1 Benchmark

In addition to the analysis of QPU performance presented in chapters 3 to 5, this section examines random sampling, simulated annealing, and Leap's hybrid solver. Table 6.1 provides a list of the parameters used to benchmark binary and gate-based encoding.

Table 6.1: List of parameters for summary plots. Random sampling, simulated annealing and hybrid annealing are introduced in this chapter. The annealing parameters chain strength and annealing time as well as the concept of embeddings do not apply to them.

Random guesses	Number of samples n_{samples}	1000
	Threshold for success p	>0.5
Simulated	Number of runs n_{runs}	100
	Number of sweeps n_{sweeps}	1000
QPU	Number of embeddings n_{e}	10
	Number of samples n_{samples}	1000
	Binary $J_{\text{chain}} / \max \lvert Q_{ij} \rvert$	0.38
	Gate-based $J_{\text{chain}} / \max \lvert Q_{ij} \rvert$	0.30
	Annealing time $t_{\max} / \mu\text{s}$	20
Hybrid	Number of samples n_{samples}	10

Figure 6.1 illustrates the maximum number of bits N that is able produce correct solution samples for a given resolution n_x and sample size. The hatched area in Fig. 6.1 marks the N-region where solutions are impossible. Gate-based encoding consistently solves larger problems than the binary QUBO formulation. This is true for problems solved with Leap's hybrid solver, simulated annealing, and for problems solved on the QPU, which is the primary focus of the thesis.

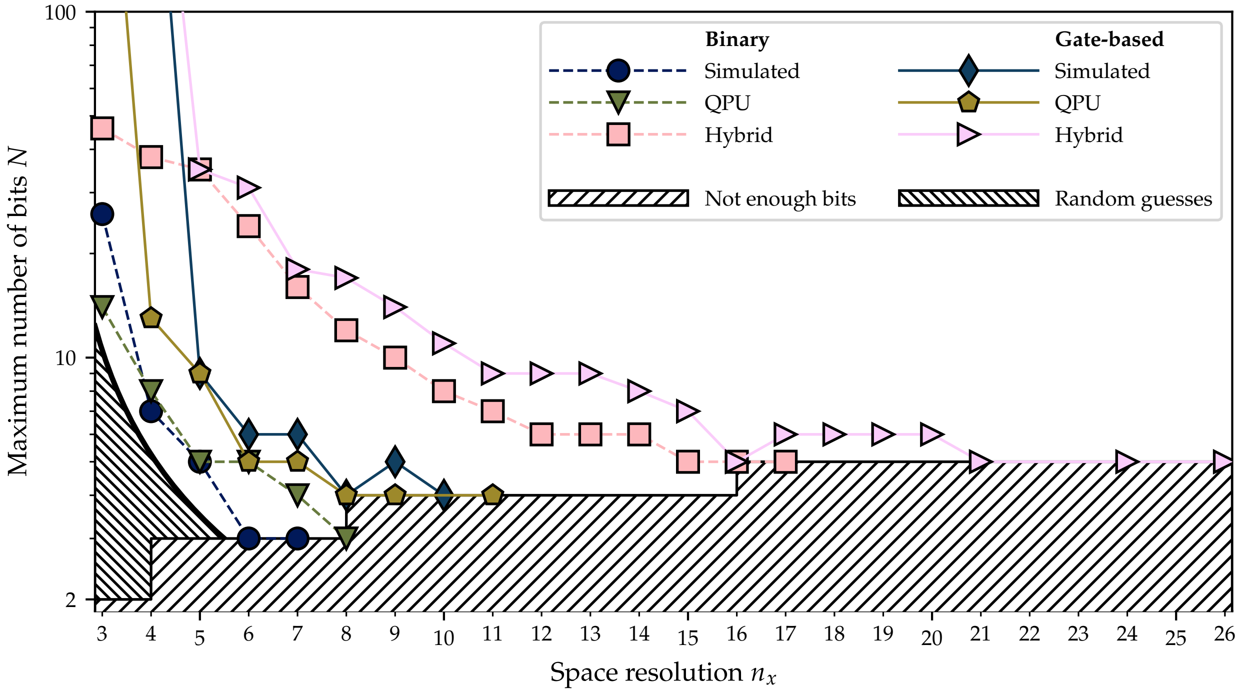


Figure 6.1: Maximum number of bits N that produces successful samples for the given sample size. The QPU plots are based on tuned data with gate-based chain strengths of 0.30 and binary chain strengths of 0.38. Default annealing time of 20 μs was used. Table 6.1 lists the individual settings and sample sizes.

The hatched area on the bottom marks the region, where we lack sufficient bits to solve the problem. The hatched area on the left marks the region where randomly guessing the solution is more likely to succeed than not and is given by equation 6.1. The gate-based QPU marker at $n_x = 10$ and hybrid markers at $n_x \in \{22, 23, 25\}$ are actually missing, as the analysis did not produce a correect solution for those resolutions and the given sample size. Lines are guides to the eye.

Gate-based encoding is successful for almost-trivial problem scenarios $n_x \in \{3, 4\}$ with numbers of bits N so large that we exclude them from the plot. For $n_x = 4$ failure occurred with 2 000 bits, while 1 000 bits succeeded. For $n_x = 3$ failure occured with 60 000 bits, while 50 000 bits succeeded.

In Fig. 6.1, it is not possible to make a direct comparison between simulation, QPU annealing, hybrid annealing and the threshold for random guesses because the respective sample sizes were chosen independently from one another. The QPU annealing analysis consists of $n_{\text{samples}} = 1000$ for each of the $n_{\text{e}} = 10$ embeddings, while the simulated annealing analysis consists of $n_{\text{sweeps}} = 1000$ sweeps for each of the $n_{\text{runs}} = 100$ runs. Increasing the sample size allows for solving problems at higher resolutions. However, it is important to note that larger sample sizes were not analysed. Chapters 3 and 5 demonstrate an exponentially decreasing success probability as the problem scale (n_x, N) increases. The same trend is observed with the random guesses, where $n_{\text{samples}} = 1000$ samples were taken.

Section A4 in the appendix derives the threshold for random guesses, which is

$$N(n_x) = -\frac{\log_2\left(1 - p^{1/n_{\text{samples}}}\right)}{n_x - 2}, \tag{6.1}$$

where n_{samples} is the number of guesses and $p = 1/2$ causes a random guess to be more likely to succeed than not. Due to the exponentially diminishing success probabilities, the impact of this threshold and the sample size is visible but minimal. This observation also applies to simulated annealing and hybrid annealing. Although hybrid annealing is not the main focus of this work, it is necessary to use more than one hybrid sample to avoid artifacting. We have selected a sample set of $n_{\text{samples}} = 10$ hybrid samples to ensure meaningful results.[1]

Although there is a lack of comparability, it is evident that simulated annealing and quantum annealing significantly outperform 1000 random guesses.[2] In addition, the hybrid annealer significantly outperforms a pure QPU approach, regardless of the number of hybrid samples. It demonstrates the potential of current hardware implementations and provides a simple benchmark for comparing binary encoding with the gate-based approach. The hybrid results give weight to the rest of the analysis, since they mitigate potential biases, errors or false assumptions among the complexities of the QPU approach along with all the hyperparameters, solver types and implementation intricacies. Nevertheless, we focus on quantum annealing instead of studying and possibly reverse engineering the black box that is Leap's hybrid solver.[3]

6.1.1 *Performance regardless of bit count*

We include problems with very high numbers of bits N, much higher than necessary to encode solutions. This is because those problems are the main weakness and bottleneck in ordinary binary encoded QUBOs. However, many-bit solutions are sparse and for very small problem sizes it can take a significant amount of time to find a number of bits N that is high enough to result in failure. Additionally, analysing more bits than necessary was done to maintain consistency when comparing performance with different resolutions n_x. If sweeps were performed on problem sizes n_x with variable encoding lengths N, jumps would occur at resolutions $n_x \in 2^{\mathbb{N}}$ which require additional bits.

Nevertheless, the number of bits N, is an implementation detail and not a parameter of the problem at the analytical level. The analytical difficulty is determined by the grid size, boundary conditions and initial conditions. The required number of bits can be determined before encoding the problem, and the actual number is irrelevant as long as it is sufficient to decode a successful solution. Figure 6.2 illustrates the largest problems that can be solved for *any* sufficiently high number of bits N. The six data sets from Fig. 6.1 are flattened by discarding the N-information. The metric shows the highest x-resolution n_x that produces correct samples regardless of N.[4] Gate-based encoding for simulated, QPU and hybrid annealing each significantly outperforms the binary approach for the selected problem scenario.

[1] With one hybrid sample, we could get unlucky with a probability of $1 - p$. With an underlying success probability of 0.5, the probability of 1 sample missing the correct answer is 0.5, whereas the probability of 10 samples missing the correct answer is below 0.001. However, with 10 samples, we can expect to observe failure around success probabilities smaller than 0.1 and practically eliminate the possiblity of success probabilities slightly below 1.0 failing.

[2] This is the case even if we were to assume a different threshold for success than 0.5.

[3] Leap's hybrid annealer has limited options for modifying the behaviour of the black box.

[4] In a way, it indicates the resolution at which the number of bits N crashes into the hatched area - the region where solutions are no longer possible.

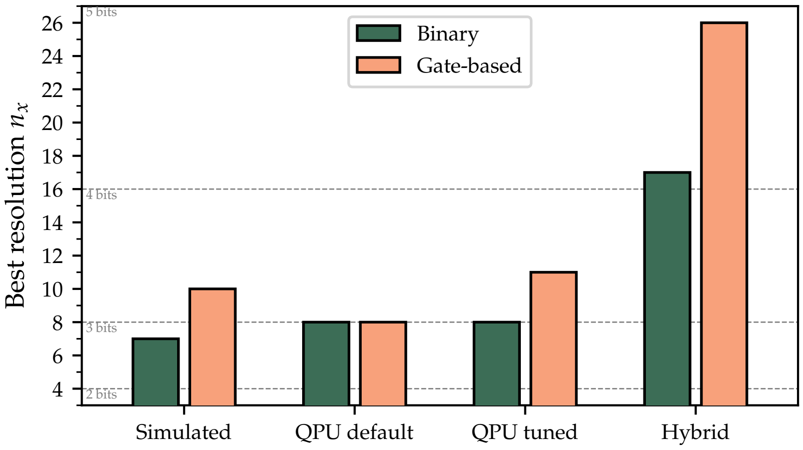


Figure 6.2: Comparison of maximum resolution n_x that can be solved with any amount of bits N given the sample size. The tuned QPU analysis used a gate-based chain strength of 0.30 and a binary chain strength of 0.38. Default annealing time of 20 μs was used. Table 6.1 lists the individual settings and sample sizes. The gate-based tuned QPU succeeds for $n_x = 11$ but fails for $n_x = 10$ and hybrid annealing succeeds for $n_x = 26$ but fails for $n_x \in \{22, 23, 25\}$. The dashed grey lines delimit the resolutions n_x that can produce solutions for the labeled number of bits. 1000 random guesses would more likely succeed for resolutions up to 5 pixels than not.

	Improvement
Simulated	60%
QPU default	0%
QPU tuned	50%
Hybrid	60%

Table 6.2: List of relative improvements of maximum problem size from Fig. 6.2. Note that two pixels are fixed due to Dirichlet boundary conditions. Therefore, $n_x - 2$ is the reference count as it is proportional to the number of variables n_h.

We also included default parameter performance for comparison. In chapter 4 we were unable to improve the performance of binary encoding through parameter tuning. In Fig. 6.2 it can be seen, that both the QPU with default parameters and the QPU with tuned chain strength are unable to exceed a resolution of $n_x = 8$. It is worth noting that these solutions only require $N = 3$ bits, whereas resolutions with $n_x > 8$ require a fourth bit to encode the nine different Φ values in the array of solutions. Figure 6.1 demonstrates this requirement with the binary data around a resolution of $n_x = 8$. The binary model reaches the hatched area as the resolution approaches $n_x = 8$. It can barely solve the 3-bit problem and solving the 4-bit problem seems practically impossible with the given sample sizes. In summary, the binary model faces additional challenges in solving higher resolutions due to the step in the hatched area at $n_x = 8$, further diminishing its success probabilities. On the other hand, gate-based encoding performs significantly better and is able to overcome this 4-bit threshold. With that obstacle overcome, gate-based encoding reaches $n_x = 10$ and $n_x = 11$, since the next threshold is at a resolution of $n_x = 16$.

It is important to note that gate-based encoding benefits from hyperparameter tuning in contrast with the binary formulation. Gate-based encoding produces more successful samples with default parameters. However, tuning of chain strength improves the maximum resolution from $n_x = 8$ to 11 for the given sample size. This is a significant improvement, and without tuning, binary encoding can only solve problems with equal resolution. The default chain strengths based on the UTC model can be described as an educated guess at best. Binary models are not noticeably impacted by the chain strength, while the default values used by D-Wave put gate-based models at a disadvantage. The latter models turn out to be more sensitive to the choice of chain strength J_{chain}.

6.1.2 *Discussion of errors*

To conclude this comparison between binary and gate-based QUBO models, we will now examine the false solution behaviour presented in Fig.3.2 and Fig. 5.12. Figure 6.3 illustrates the errors $\Phi^i - \Phi(x^i)$ of each pixel for 4-bit resolutions. The figure shows that gate-based models can solve resolutions of 9 and 11, whereas binary models cannot.

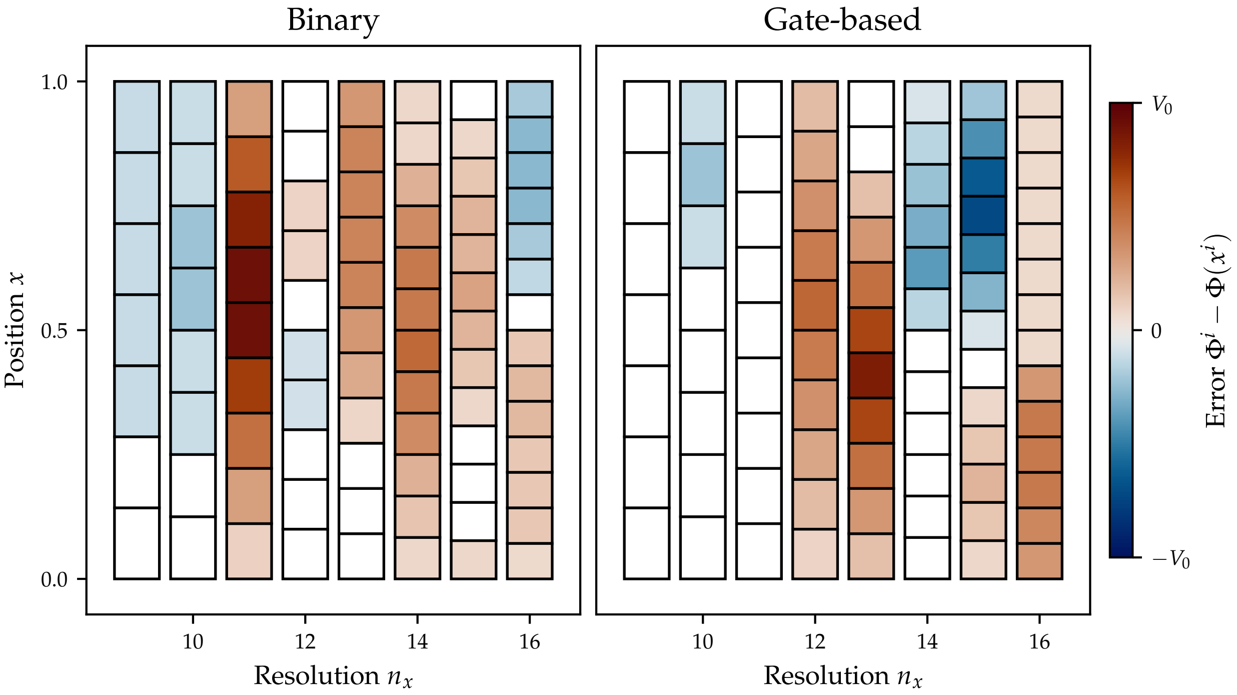


Figure 6.3: Difference between best sample and correct solution $\Phi^i - \Phi(x^i)$ with tuned parameters as evaluated by *energy*. A gate-based chain strength of 0.30 and a binary chain strength of 0.38 were used. Default annealing time of 20 μs was used. The electrostatic potentials are given in Fig. 5.12 and Fig. 3.2.

Regardless of the model used, errors in Figure 6.3 are strongly correlated and can appear as smooth gradients. This is because the discrete Laplace operator solves for piecewise potentials but fails to do so globally. For instance, instead of increasing the potential in integer steps, the potential can increase with a slope of 2 per pixel or more. The discrete Laplace operator yields each value's difference from the average of its neighbouring values, resulting in solutions that are piece-wise linear in one dimension.

Homogeneous batches of errors may occur, especially with gate-based models, as seen in the 16-pixel potential.

$$\mathbf{\Phi} = \frac{1}{15}(0, \cdots, 8, 9, 10, 11, 12, 13, 14, 15, 15)^T \; V_0 \qquad (6.2)$$

In particular, the least significant solution bit Φ_0^{13} contains an error with a value of 1 instead of 0, which propagates and results in the regions in-between becoming local solutions of the discrete Poisson equation. These errors resemble kinks.[5] The potential between the kinks in these regions increases linearly in integer steps, but deviates from the true potential by an offset. Since errors can be identified in specific locations on the QPU chip, there may be a possibility of correction, although this is beyond the scope of this thesis. It is worth noting that correction may not be necessary. The system is described by the physical quantity of the electric field, which remains invariant when a constant is added to the potential.

There is a subtle but important difference between binary and gate-based encoding. Gate-based values are more robustly connected due to the ancillas, which act as a support. Binary encoding also has neighbouring pixels that depend on each other, but the interactions in the QUBO matrices are more dispersed.[6]

Table 6.3 presents the lengths n_{seg} of the longest segments that locally solve the Poisson equation, irrespective of offset or slope.

Table 6.3: Lengths n_{seg} of the longest segments that locally solve the Poisson equation for 4-bit resolutions. n_x is the x-resolution. A relative segment length n_{seg}/n_x of 1 indicates a sample that globally solves the Poisson equation. The right column lists the relative improvement $\Delta n/n_x$ and Δn is the difference between gate-based and binary segment lengths n_{seg}. The bottom row gives the mean values and the standard errors of the mean.

n_x	n_{seg}		$\frac{n_{\text{seg}}}{n_x}$ (%)		$\frac{\Delta n_{\text{seg}}}{n_{\text{seg}}}$ (%)
	Binary	Gate-based	Binary	Gate-based	
9	4	9	44	100	125
10	3	6	30	60	100
11	4	11	36	100	175
12	4	7	33	58	75
13	4	6	31	46	50
14	4	7	29	50	75
15	5	5	33	33	0
16	3	8	19	50	167
Mean	3.9	7.4	32	62	96
	± 1.4	± 2.6	± 11	± 22	± 34

It is important to note that in this context, 'correct' does not necessarily mean a good approximation of the global analytical solution, but instead that they solve the Poisson equation locally. Integer increment regions and linear regions, regardless of their slope, are considered correct. Table 6.3 shows that segments with gate-based encoding are significantly longer.

The disparity is significant, with even the second longest segments of gate-based encoding, i.e. $0, 4, 0, 5, 6, 6, 4$ and 3, are capable of competing with the longest segments of binary encoding. Since the gate-based formulation solves the discrete Poisson equation globally for resolutions n_x of 9 and 11 pixels, there cannot be a secondary segment of any size. Therefore, the corresponding seg-

[5] The indices refer to the pixel $i = 13$ adjacent to the boundary at $x = 1$, which has index 14 for a resolution of 15. The subscript 0 is the least signficant bit α turning what should be a potential of 14 to 15 in equation 6.2.

[6] Fig. 3.3 plots the binary QUBO matrix and Fig. 5.5 plots the gate-based QUBO matrix.

ments have a length of 0. In conclusion, the secondary gate-based segment length is only shorter than the maximum binary segment length for $n_x = 15$ pixels, which is not surprising since a third of the x-domain is already occupied by the primary segment of length 5.

6.2 Prospect

Section 6.1.2 establishes that errors are not stochastic but strongly correlated. As mentioned in section 5.8, errors among the ancilla qubits affect the energy in gate-based encoding. It would be worthwhile to analyse a metric that determines the best sample unaffected by deviating ancilla values. When using a different evaluation metric, the production of samples remains unchanged, only the order of suboptimal samples is altered.

Accurate approximations are correlated with low energy levels. However, it is important to note that the energy level alone is not a reliable indicator of the quality of the solution candidate, even when false ancillas are not present. To address this, a thorough search of the entire sample set was conducted to identify the best candidate based on the sum of squared differences χ^2.[7] Figure 6.4 shows The comparison between binary and gate-based QUBO formulations. The evaluation metric is now independent of ancillary data, which allows correction of any distortion caused by energy-based sampling in Fig. 6.3.

However, determining what constitutes a good solution is not immediately apparent. Although manually evaluated samples may provide a more accurate approximation of the correct solution, it is important to note that the deviations in Fig 6.3 are highly correlated. It is possible that we resorted to brute-force searching for a lucky guess and discarded a seemingly worse solution that is correct over the entire region, albeit with one or two major but correctable kinks which distort the sum of squared differences χ^2. The evaluation of every pixel independently in the sum of squared differences χ^2 discards important correlations. Further investigation is necessary to answer these questions with certainty.

However, the custom evaluation metric can provide an indication of each method's potential in the best-case scenario. Furthermore, when examining the gate-based case for a resolution of $n_x = 10$, it is evident that accounting for false ancillas is crucial. In such cases, the energy-based evaluation produces a significantly inferior solution compared to the least-squares-based evaluation, as shown in Tab.6.4. However, it is worth noting that $8 = (1000)_2$

[7] See Fig. 5.13

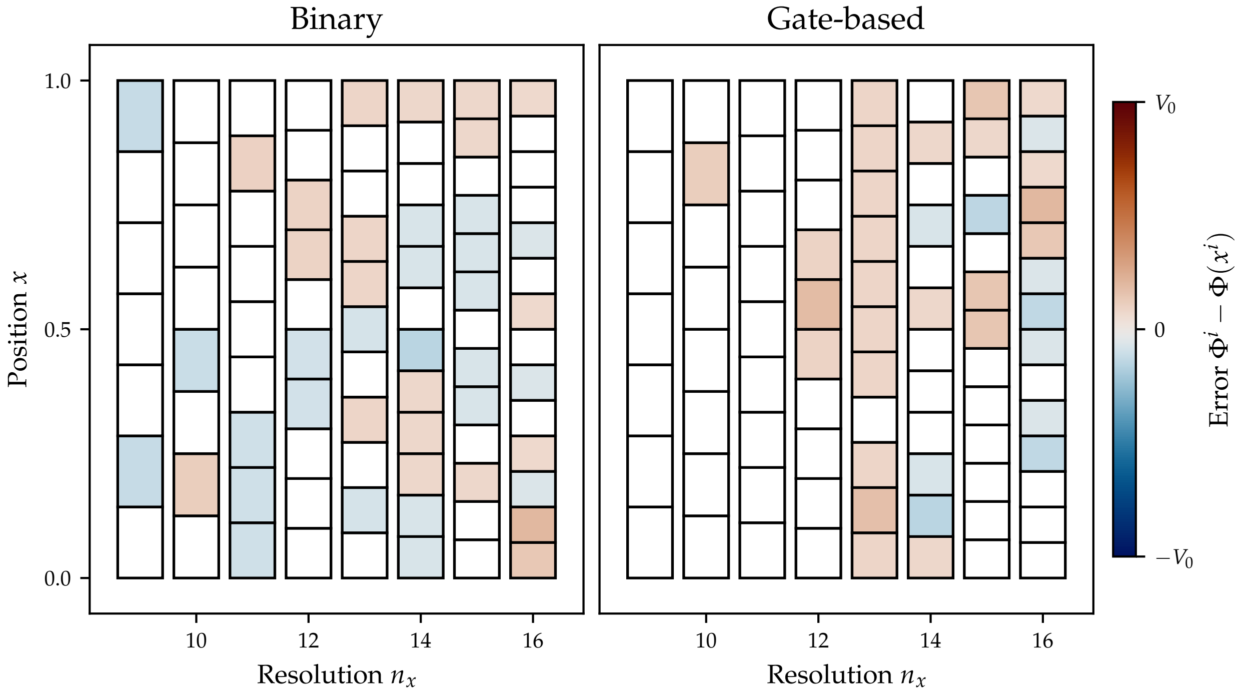


Figure 6.4: Difference between best sample and correct solution $\Phi^i - \Phi(x^i)$ with tuned parameters as evaluated by the *sum of squared differences* χ^2. A gate-based chain strength of 0.30 and a binary chain strength of 0.38 were used. Default annealing time of 20 μs was used. The gate-based electrostatic potentials is given in Fig. 5.13.

and $7 = (0111)_2$ are represented by complementary bit strings. Therefore, the high energy is not necessarily due to false ancillas. As explained in section 5.8, correct solutions within the sample set may have non-zero energy. Therefore, it is imperative to assess the samples using a specific metric.

Table 6.4: Comparison of energy-based and least-squares-based evaluation metrics H and χ^2 respectively for a resolution of 10 pixels. The correct solution is $(\Phi^i) = (0, 1, \cdots, 9)^T$. The bottom sample is more accurate.

H	χ^2	Sample Φ^i									
2	6	0	1	2	3	4	5	5	5	7	9
4	1	0	1	2	3	4	5	6	8	8	9

Before concluding this section, we have three remarks. Firstly, additional analysis is necessary for different QPUs, particularly for Advantage 2. Secondly, a promising area for future research is to apply the concept introduced here to different problem scenarios such as the reduced Navier-Stokes equation (rNSE) in one dimension. By selecting an appropriate time resolution Δt, the rNSE generalises the Poisson problem by replacing the 2s in the diagonal with arbitrary numbers.[8] Finally, in section 6.6, gate-based encoding is discussed in the context of higher-dimensional problem scenarios.

[8] Azzam, "Evaluating Partial Differential Equations on the Quantum Annealer JUPSI" (2023).

6.3 *About gate-based encoding*

The gate-based encoding approach is not a substitute for binary encoding, but rather a new formulation that uses gates to encode the problem. This approach eliminates the issue of exponentially increasing weight of leading digits, which causes auto-scaling to shrink the smaller weights to values smaller than the realizable precision of the magnetic fields on the QPU. However, some may view this approach as defeatist because it prioritises the accuracy of leading digits at the expense of smaller digits, ultimately resulting in rounded values.

Gate-based encoding assigns equal QUBO weights to each bit in the final bit string. The resulting bit string returned by D-Wave is identical in both cases. The bit string is treated like a combination lock, where all digits must be correct for the solution to be considered valid. In this analogy, the lock will not open if even a single digit is incorrect.

However, there is a cost associated with this improvement. For standard encoding, only $(n_x - 2)N$ variables are required, one for each of the N digits of the $n_x - 2$ solution variables. It is impossible to create a QUBO formulation that can produce a $(n_x - 2)N$-bit solution without using $(n_x - 2)N$ variables. Gate-based encoding typically requires three additional variables per solution variable: `sum`, `carry` and `borrow` ancillas. [9]

6.4 *Requirements for gate-based encoding*

The approach is inspired by the factorisation of a number $p \times q$ into p and q, where the product $p \times q$ is represented by a multiplication circuit.[10] At first, we expressed each row of the LSE $D\boldsymbol{\Phi} = \boldsymbol{\rho}$ as a sum of products

$$D^{i0} \times \Phi^0 + D^{i1} \times \Phi^1 + \cdots + D^{i,n_x-1}\Phi^{n_x-1} = \rho^i. \tag{6.3}$$

However, the multiplication of a single product $p \times q$ is already a significant issue and the number of variables increases drastically in the case of matrix multiplication. We are dealing with $O(n_x^2)$ products that must be summed up row by row.

Two critical aspects were observed for gate-based encoding to show promise.

1. Firstly, D is highly sparse, meaning that no multiplications are required for off-tridiagonal i, j with $D^{ij} = 0$.

2. Doubling a binary number is a straightforward process, similar

[9] Note that this is not the case for the first and the last digits.

[10] Jiang et al., "Quantum Annealing for Prime Factorization" (2018).

to multiplying a decimal number by ten (cf. Fig. 5.3).

The discrete Laplace operator has a distinct and highly sparse band structure, consisting of $[-1, 2, -1]$ blocks. This structure can be effectively exploited.

The hypothesis is that arithmetic problems involving sums, differences, and multiplication by powers of 2 can benefit from the gate-based approach, especially when there is only one exact solution. However, if a problem is an approximation that needs to be optimized, not all digits should necessarily carry equal weight.[11]

Although multiplying by powers of 2 is straightforward, it does not necessarily mean that other multiplications are impossible. To multiply a number by a factor of 3, double the number and add it to itself

$$3 \times x = 2 \times x + x. \tag{6.4}$$

In other words, multiplication by a factor of $3 = 2^1 + 1$ results in one extra addition. Multiplication by a factor of $2^k + n$ results in n extra additions.[12]

6.5 Limitations

For simple problems $n_x = 3$ with solution $(0 \cdots 01)$ and $n_x = 4$ with solution $(0 \cdots 001), (0 \cdots 010)$, practical problems arise such as a long wall-clock time to form the binary quadratic model (BQM).

We will focus on larger n_x grids and avoid exploring small problems. The implementation loops through each $\alpha \in \{2, 3, \cdots, N-1\}$ twice and connects the many sub-QUBOs using string labels. It is important to note that current implementations are restricted by the QPU topology and performance.

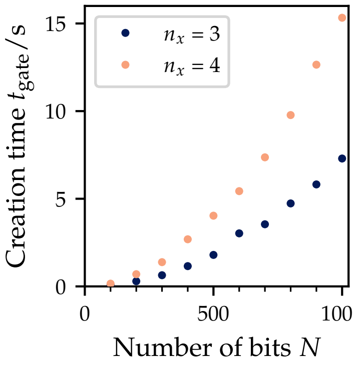


Figure 6.5: Creation time of gate-based models for low resolutions of 3 and 4 pixels. The durations are timed on our development laptop since we are primarily interested in the scaling. No considerations for optimization were made.

6.6 Higher dimensions

The discrete Laplace operator in two dimensions $D_{2\text{d}}$ is highly sparse. However, it consists of 4s on its diagonal and four non-zero values per row as opposed to just two. The resulting blocks are

$$[-1, 0, \cdots, 0, -1, 4, -1, 0, \cdots, 0, -1]. \tag{6.5}$$

The gate-based encoding approach shows promise even in higher dimensions, as it can recursively leverage the one-dimensional implementation

$$[-1, 2, -1] \circ [-1, \cdots, 2, \cdots, -1] \rightarrow [-1, \cdots, -1, 4, -1, \cdots, -1]. \tag{6.6}$$

[11] For instance 999.99 is a good approximation to 1 000 albeit all digits are different. In the gate-based paradigm, 0 is actually closer to 1 000 since all but a single digit match.

[12] If $2^k + n$ is closer to 2^{k+1}, then we can of course multiply by 2^{k+1} and proceed to perform the necessary amount of subtractions.

It is noteworthy that we can significantly[13] increase the number of grid cells without requiring more bits N. This allows us to create non-trivial problems similar to the balanced problem scenarios in various ways. One possible method for adapting the problem to a second dimension is outlined in the schema in Fig. 6.6.

$$\begin{array}{|c|ccc|c|}\hline 0 & 1 & 2 & 3 & 4 \\ \hline\end{array} \rightarrow \begin{array}{|c|ccc|c|}\hline & 1 & 2 & 3 & \\ \hline 0 & 1 & 2 & 3 & 4 \\ 0 & 1 & 2 & 3 & 4 \\ 0 & 1 & 2 & 3 & 4 \\ \hline & 1 & 2 & 3 & \\ \hline\end{array}$$

Figure 6.6: Embedding the problem scenario in two dimensions. Gradient from edge to edge.

Applying a one-sided bias of $V_0 = 4$ does not offer new information, although we can benchmark the more complex problem. It is worth considering applying a one-sided bias to the corner of the topology as shown in Fig. 6.7. In general, higher-dimensional gate-based encoding makes more efficient use of the available encoding space. We can also make more efficient use of the available encoding space in one dimension, by creating problem scenarios that alternate between 0 and $V_0 > 0$. However, this approach requires non-zero charge densities $\rho(\boldsymbol{r}_0)$ for the maxima and minima.

$$\begin{array}{|c|ccc|c|}\hline 0 & 1 & 2 & 3 & 4 \\ \hline\end{array} \rightarrow \begin{array}{|c|ccc|c|}\hline & 0 & 1 & 2 & \\ \hline 0 & 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 6 \\ \hline & 4 & 5 & 6 & \\ \hline\end{array}.$$

Figure 6.7: Two dimensional problem with gradient from corner to corner.

$$\begin{array}{|c|ccc|c|}\hline 0 & 1 & 2 & 3 & 4 \\ \hline\end{array} \rightarrow$$
$$\begin{array}{|c|cccccccccc|c|}\hline 0 & 1 & 2 & 3 & 4 & 3 & 2 & 1 & 0 & 1 & \cdots & 4 \\ \hline\end{array}$$

Figure 6.8: Alternating potential Φ due to presence of charges ρ. Solutions with local minima for $x \in (0, 1)$ require the presence of negative charges. Signed integer encoding is required to represent negative charge densities.

Furthermore, problems can become more interesting in higher dimensions. In one dimension, solutions are necessarily linear. The solution can always be written down[14], even for the Poisson problem with non-zero charge density $\boldsymbol{\rho}$. In two dimensions, the potential $\boldsymbol{\Phi}(\boldsymbol{r})$ induced by a non-zero charge density in cell $\boldsymbol{r}_0$ drops with $1/\|\boldsymbol{r} - \boldsymbol{r}_0\|$, following the Coulomb law.

[13] quadratically

[14] As a rule of thumb, for cells $\boldsymbol{r}_0$ with $\rho(\boldsymbol{r}_0) = 0$, the discrete Laplace operator acts in a way that each cell must be the average of the $2 \times \mathrm{dim}$ adjacent cells.

Appendix

A1 Tables of abbreviations and symbols

	Description
BQM	Binary quadratic model
CCJJ	Compound-compound Josephson-junction
CSP	Constraint satisfaction problem
LHS	Left-hand side
LSE	Linear system of equations
PDE	Partial differential equation
QUBO	Quadratic unconstrained binary optimization
RHS	Right-hand side
rNSE	Reduced Navier-Stokes equation
SDK	Software development kit
SQUID	Superconducting quantum-interference device
UTC	Uniform torque compensation

Table A.1: List of abbreviations.

Symbol	Description
$A(t), B(t)$	Annealing functions that define annealing schedule
$A\boldsymbol{x} = \boldsymbol{b}$	Notation for generic LSE
α	Greek index describing binary digit α in binary representation
b^i_α	Borrow bit α for row i
c^i_α	Carry bit α for row i
$D\boldsymbol{\Phi} = \boldsymbol{\rho}$	Discrete laplace equation
D	Discrete Laplace matrix
Δ	Overhead time
Δx	Spacing between discretized x points
$\boldsymbol{E}(x)$	Electrical field
ϵ_0	Permittivity of free space
$\Phi(x)$	Electrostatic potential
$\boldsymbol{\Phi}$	Discretized electrostatic potential
Φ^i	i-th component of discretized electrostatic potential
$\Phi^{(x)}_i$	External flux on qubit i

Φ_{CCJJ}	Flux applied to all CCJJ structures
H	Hamiltonian, energy function, loss function, objective
H_0	Initial Hamiltonian
H_{problem}	Final / problem Hamiltonian
h_i	Linear Ising weight for qubit i
i	Index enumerating space discretization (latin)
$i\alpha$	Short form for (i, α) describing multi-index i, α
J_{ij}	Quadratic Ising weight for qubits i, j
J_{chain}	Chain strength
$J^{\star}_{\mathrm{chain}}$	Optimal chain strength
J_{def}	Default chain strength
J_{utc}	Uniform torque compensation based chain strength
N	Number of bits or *encoding length*
n_{at}	Number of annealing times
n_{cs}	Number of chain strengths
n_{e}	Number of embeddings
n_h	Number of variables in Q
n_J	Number of interactions in Q
n_{runs}	Number of runs of the simulated annealing algorithm
n_{samples}	Sample size or number of guesses
n_{success}	Number of successes
n_{sweeps}	Number of sweeps used in simulated annealing
n_x	Number of equidistant, discrete x values that represent the interva
$\oplus$	Addition modulo 2
$\ominus$	Subtraction modulo 2
p	Estimator for success probability
p_{fail}	Probability of failure
Q	QUBO matrix
Q^i_α	QUBO coefficient of multi-index $i\alpha$
$Q_\oplus$	QUBO matrix for full-adder gate
$Q_\ominus$	QUBO matrix for full-subtractor gate
$\rho(x)$	Charge density
$\boldsymbol{\rho}$	Discretized charge density
ρ^i	i-th component of discretized charge density
s^i_α	Sum ancilla α for row i
$\sigma^{(x)}$	Pauli-x-matrix
$\sigma^{(z)}$	Pauli-z-matrix
T_{d}	Delay time
$t_{\mathrm{max}}, t_{\mathrm{annealing}}$	Annealing time
T_{p}	Programming time
T_{r}	Readout time
$\boldsymbol{x}$	Variable representing all problem variables

x^i	i-th integer problem variable
x^i_α	Digit α of i-th problem variable
	Solution qubit (i, α)
(x_α)	Bit-string
x_α	Digit α of binary representation of x
V_0	Applied voltage
z	Quantile for 95% confidence interval

Table A.2: List of symbols.

A2 Hyperparameter tuning

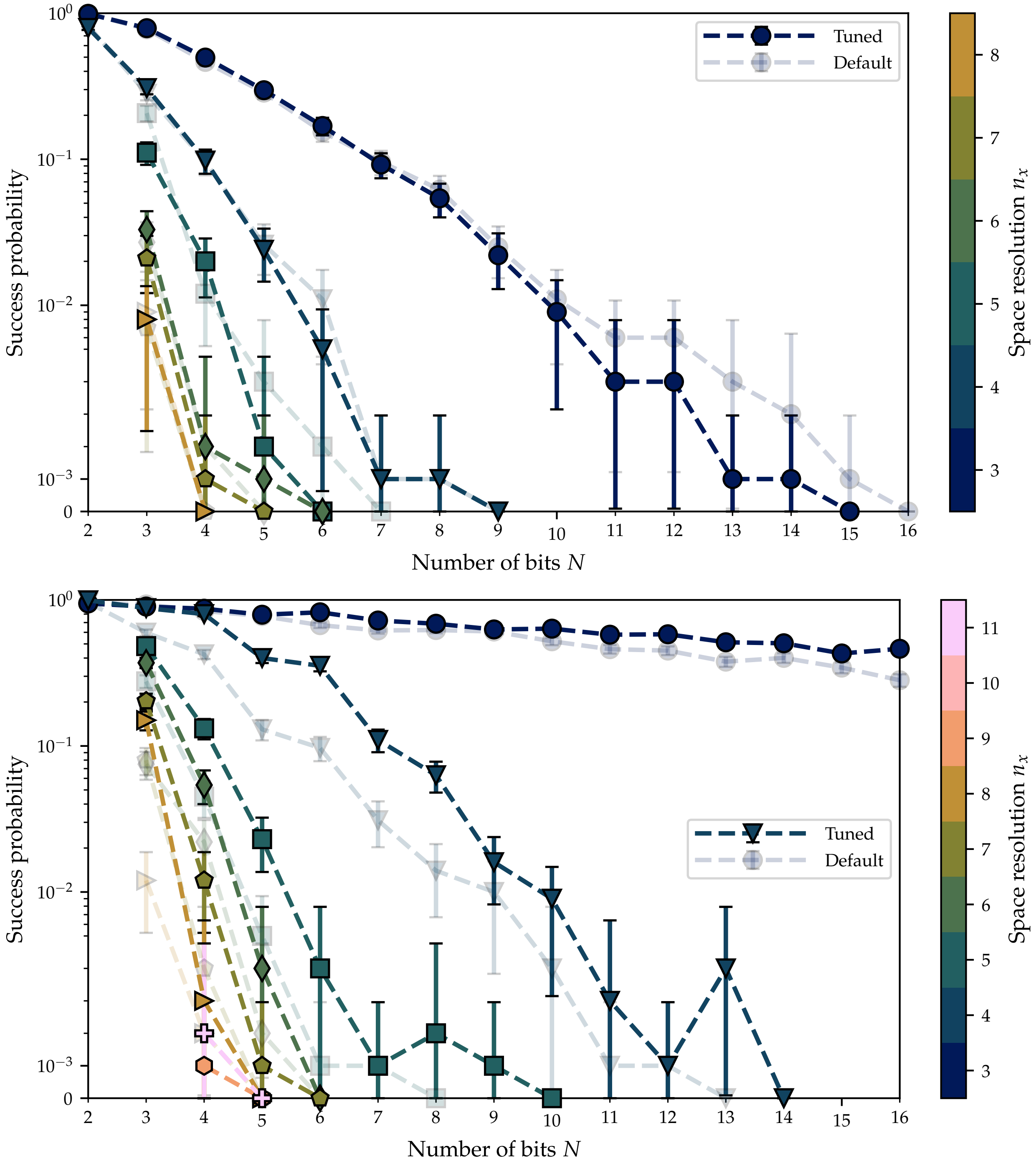


Figure A.1: Success probabilities for tuned chain strength. (*Top*) Binary encoding. A chain strength of 0.38 was used. (*Bottom*) Gate-based encoding. A chain strength of 0.30 was used.

The success probabilities for default values are included with shaded colors.

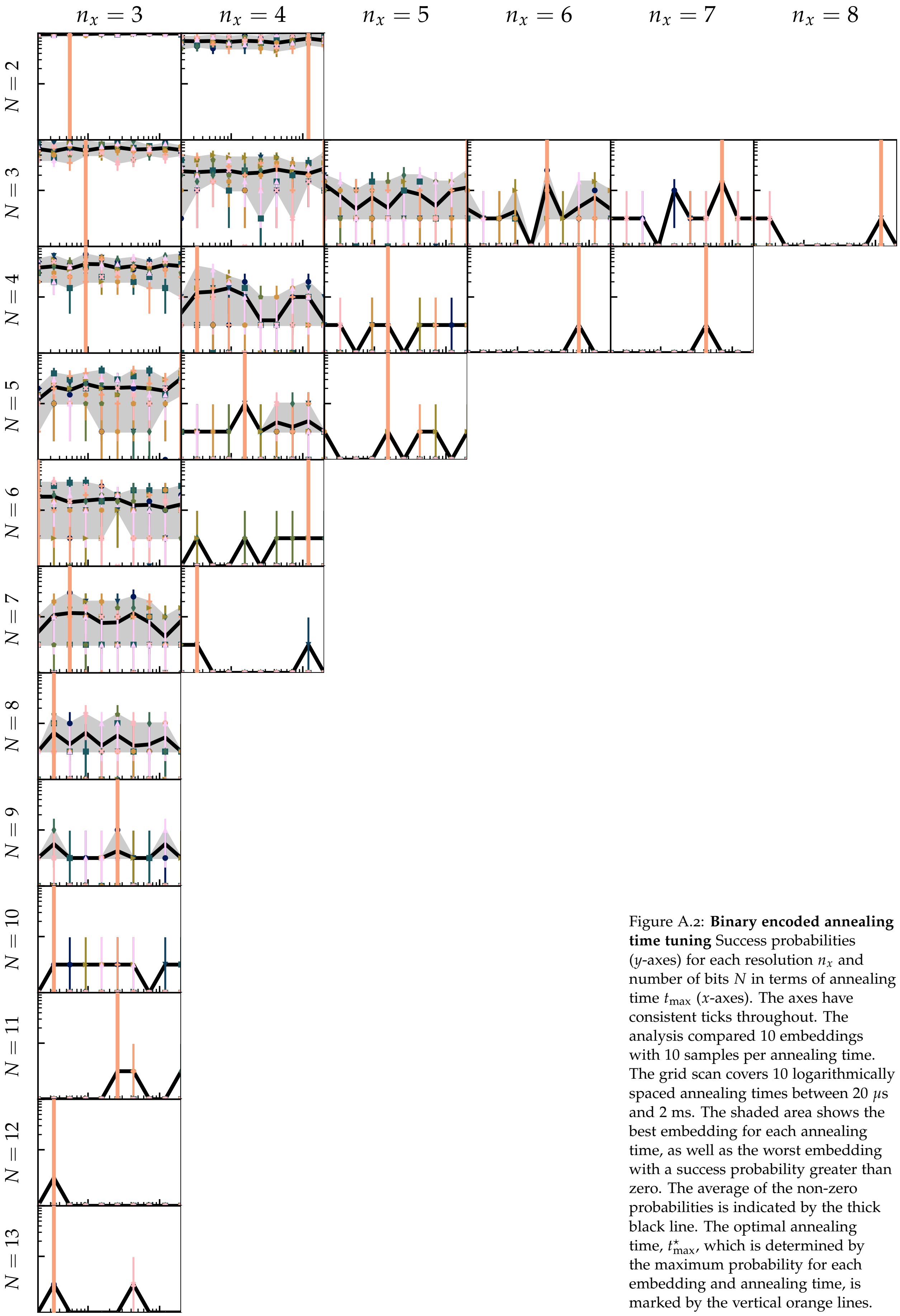


Figure A.2: **Binary encoded annealing time tuning** Success probabilities (y-axes) for each resolution n_x and number of bits N in terms of annealing time $t_{\max}$ (x-axes). The axes have consistent ticks throughout. The analysis compared 10 embeddings with 10 samples per annealing time. The grid scan covers 10 logarithmically spaced annealing times between 20 μs and 2 ms. The shaded area shows the best embedding for each annealing time, as well as the worst embedding with a success probability greater than zero. The average of the non-zero probabilities is indicated by the thick black line. The optimal annealing time, $t^{\star}_{\max}$, which is determined by the maximum probability for each embedding and annealing time, is marked by the vertical orange lines.

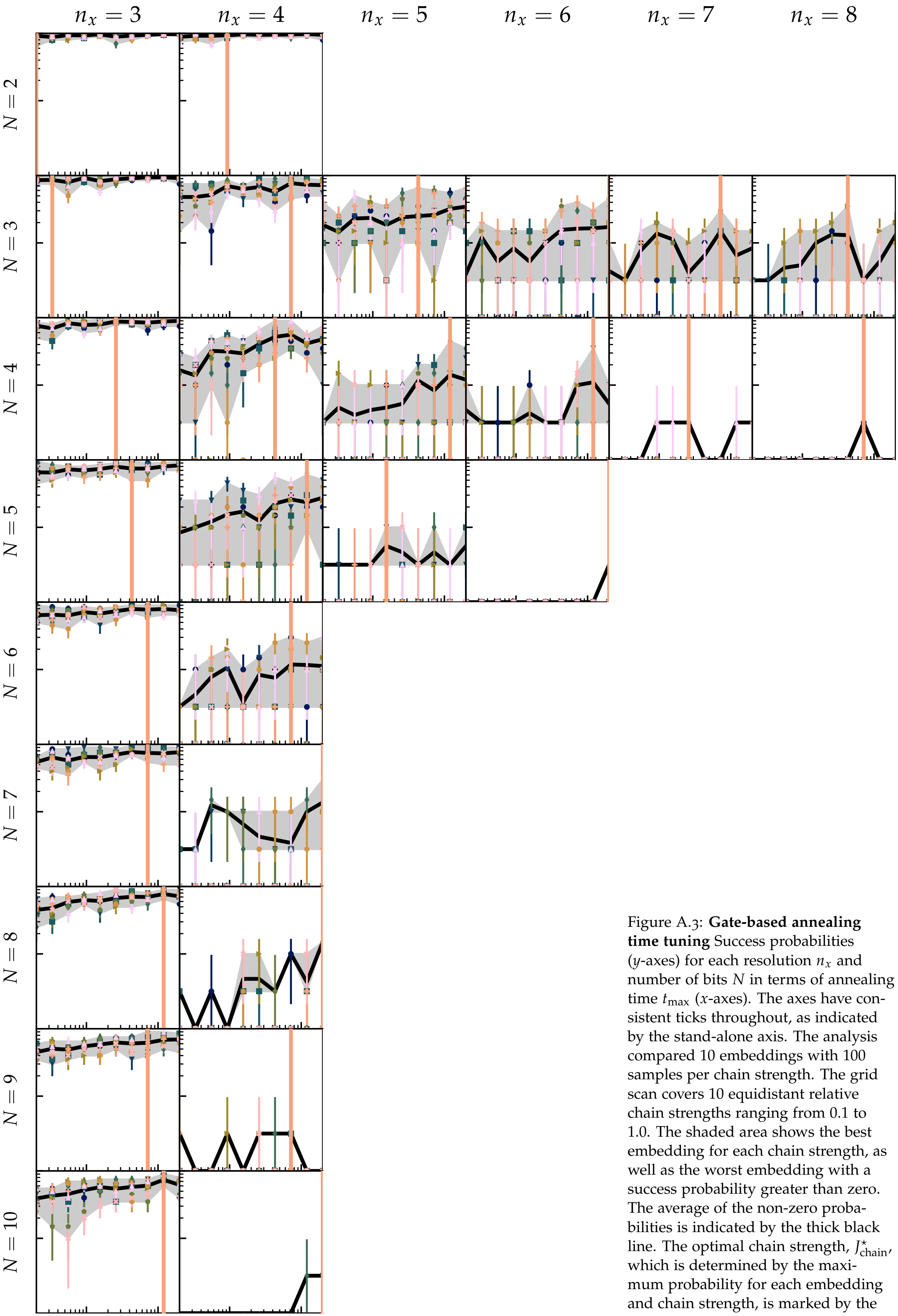


Figure A.3: **Gate-based annealing time tuning** Success probabilities (y-axes) for each resolution n_x and number of bits N in terms of annealing time $t_{\max}$ (x-axes). The axes have consistent ticks throughout, as indicated by the stand-alone axis. The analysis compared 10 embeddings with 100 samples per chain strength. The grid scan covers 10 equidistant relative chain strengths ranging from 0.1 to 1.0. The shaded area shows the best embedding for each chain strength, as well as the worst embedding with a success probability greater than zero. The average of the non-zero probabilities is indicated by the thick black line. The optimal chain strength, $J^{\star}_{\text{chain}}$, which is determined by the maximum probability for each embedding and chain strength, is marked by the vertical orange lines.

A3 Low-resolution problems

For very low-resolutions, gate-based encoding is able to solve the problem even for very high numbers of bits N.

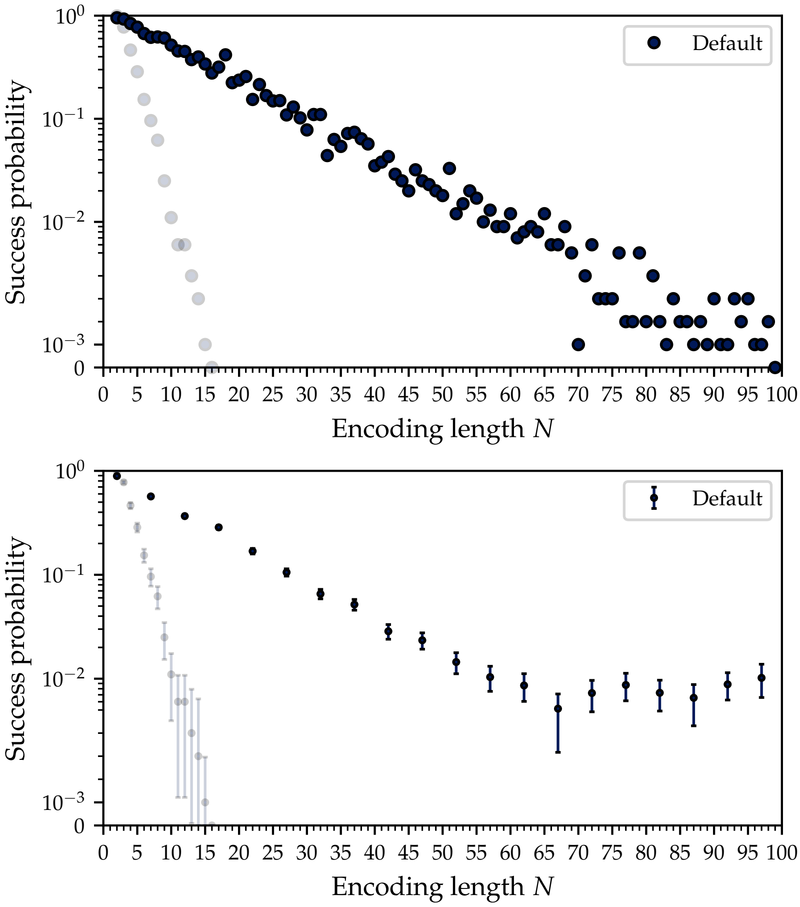


Figure A.4: Success probabilities for gate-based encoding with default values. The probabilities for binary encoding are included in shaded colors. (*Top*) Solutions without evaluating non-zero energy samples. (*Bottom*) Solutions validated by decoding and verifying, if the bit string is correct.

Integer	Φ^1	1
Binary	$(\Phi^1_\alpha)_2$	$(0\cdots01)_2$

Table A.3: Integer solution Φ^1 for $n_x = 3$ and its bit string representation. See equation (A.1).

However, $n_x = 3$ the smallest problem that is not trivial:

$$\begin{pmatrix} 1 & 0 & 0 \\ -1 & 2 & -1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ \Phi^1 \\ 2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix}. \tag{A.1}$$

A4 Random guessing threshold

Guessing $n_x - 2$ problem variables represented by N binary variables each is equivalent to guessing an n_h-bit string with

$$n_h = (n_x - 2)N. \tag{A.2}$$

The probability to guess every bit correctly is $1/2^{n_h}$ per independent guess. The failure probability with $n_{\text{samples}} = 1000$ independent guesses is

$$p_{\text{fail}} = \left(1 - \frac{1}{2^{n_h}}\right)^{n_{\text{samples}}}. \tag{A.3}$$

Since probabilities are non-negative and counts are positive, we take the root and obtain

$$\frac{1}{2^{n_h}} = 1 - p_{\text{fail}}^{1/n_{\text{samples}}}. \tag{A.4}$$

Since $\log 1 = 0$, the base-2 logarithm yields

$$n_h = (n_x - 2)N = -\log_2\left(1 - p_{\text{fail}}^{1/n_{\text{samples}}}\right). \tag{A.5}$$

The result is N as a function of $p_{\text{fail}}, n_{\text{samples}}$ and n_x

$$N(p_{\text{fail}}, n_{\text{samples}}, n_x) = -\frac{\log_2\left(1 - p_{\text{fail}}^{1/n_{\text{samples}}}\right)}{n_x - 2}. \tag{A.6}$$

A failure threshold p_{fail} of 0.5 does not predominantly affect the N-threshold when compared to a threshold of 0.1. Given n_{samples}, the N-threshold where solutions are more likely than not is given by

$$N(n_x) = -\frac{\log_2\left[1 - \left(\frac{1}{2}\right)^{1/n_{\text{samples}}}\right]}{n_x - 2}. \tag{A.7}$$

A5 *Relation between QUBO and Ising coefficients*

We derive the relation between the Q_{ij} from binary models and h_i, J_{ij} from spin models. Binary variables x_i take values 0 and 1, while spin variables s_i take values ± 1. The transformation $x_i \leftrightarrow s_i$ is given by

$$x_i = \frac{s_i + 1}{2} \qquad s_i = 2x_i - 1 \tag{A.8}$$

We insert (A.8) into (2.8) in (A.11).

$$H(\boldsymbol{x}) = \sum_i Q_i x_i + \sum_i \sum_{j>i} Q_{ij} x_i x_j \tag{A.9}$$

$$= \sum_i \frac{Q_i}{2}(s_i + 1) + \sum_i \sum_{j>i} \frac{Q_{ij}}{4}(s_i + 1)(s_j + 1) \tag{A.10}$$

$$= \sum_i \frac{Q_i}{2} s_i + \sum_i \sum_{j>i} \frac{Q_{ij}}{4} s_i s_j + \sum_i \sum_{j>i} \frac{Q_{ij}}{4}(s_i + s_j) + H_{\text{const}} \tag{A.11}$$

The energy offset H_{const} does not affect the Ising coefficients h_i, J_{ij}. For posterity, it is given by

$$H_{\text{const}} = \sum_i \frac{Q_i}{2} + \sum_i \sum_{j>i} \frac{Q_{ij}}{4}. \tag{A.12}$$

We distribute $s_i + s_j$ in (A.11)

$$\sum_i \sum_{j>i} \frac{Q_{ij}}{4}(s_i + s_j) \tag{A.13}$$

$$= \sum_i \left(\sum_{j>i} \frac{Q_{ij}}{4} \right) s_i + \sum_j \left(\sum_{i<j} \frac{Q_{ij}}{4} \right) s_j \tag{A.14}$$

and swap $i \leftrightarrow j$ in the second sum in (A.14) to obtain

$$\sum_i \left(\sum_{k>i} \frac{Q_{ik}}{4} \right) s_i + \sum_i \left(\sum_{k<i} \frac{Q_{ki}}{4} \right) s_i \tag{A.15}$$

$$= \sum_i \left(\frac{\sum_{k>i} Q_{ik} + \sum_{k<i} Q_{ki}}{4} \right) s_i, \tag{A.16}$$

where we re-labelled $j \rightarrow k$ to avoid confusion with column indices. We insert (A.16) into (A.11) and identify

$$h_i = \frac{Q_i}{2} + \sum_{k>i} \frac{Q_{ik}}{4} + \sum_{k<i} \frac{Q_{ki}}{4}. \tag{A.17}$$

This can be written in a more compact way. Q is an upper-triangle-matrix and the terms in (A.17) correspond to the i-diagonal, the non-zero values of row i to the right and the non-zero values of column i to the top. Summing over the lower triangle does not alter the result but allows to write the term in a single sum. We obtain

$$h_i = \sum_k \frac{Q_{ik} + Q_{ki}}{4} \qquad J_{ij} = \frac{Q_{ij}}{4} \tag{A.18}$$

Bibliography

[1]T. Albash and D. A. Lidar, “Adiabatic quantum computation”, Reviews of Modern Physics **90**, 015002 (2018).

[2]E. Andriyash, Z. Bian, F. Chudak, M. Drew-Brook, A. D. King, W. G. Macready, and A. Roy, *Boosting integer factoring performance via quantum annealing offsets* (D-Wave Systems Inc., Dec. 9, 2016).

[3]F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. P. Harrigan, M. J. Hartmann, A. Ho, M. Hoffmann, T. Huang, T. S. Humble, S. V. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, J. Kelly, P. V. Klimov, S. Knysh, A. Korotkov, F. Kostritsa, D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandrà, J. R. McClean, M. McEwen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. C. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. C. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. D. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven, and J. M. Martinis, “Quantum supremacy using a programmable superconducting processor”, Nature **574**, 505–510 (2019).

[4]J. A. Azzam, “Evaluating Partial Differential Equations on the Quantum Annealer JUPSI”, MA thesis (FH Aachen, May 31, 2023).

[5]C. Baugh and B. Wooley, “A Two’s Complement Parallel Array Multiplication Algorithm”, IEEE Transactions on Computers **C-22**, 1045–1047 (1973).

[6]A. Borle and S. J. Lomonaco, “How viable is quantum annealing for solving linear algebra problems?”, version 1, 10.48550/ARXIV.2206.10576 (2022).

[7]P. I. Bunyk, E. M. Hoskinson, M. W. Johnson, E. Tolkacheva, F. Altomare, A. J. Berkley, R. Harris, J. P. Hilton, T. Lanting, A. J.

Przybysz, and J. Whittaker, "Architectural Considerations in the Design of a Superconducting Quantum Annealing Processor", IEEE Transactions on Applied Superconductivity **24**, 1–10 (2014).

[8]D-Wave Systems Inc., *Annealing Implementation and Controls*, D-Wave System Documentation, `https://docs.dwavesys.com/docs/latest/c_qpu_annealing.html`.

[9]D-Wave Systems Inc., *D-Wave QPU Architecture: Topologies*, D-Wave System Documentation, `https://docs.dwavesys.com/docs/latest/c_gs_4.html`.

[10]D-Wave Systems Inc., *D-Wave System Documentation*, `https://docs.dwavesys.com/`.

[11]D-Wave Systems Inc., *Operation and Timing*, D-Wave System Documentation, `https://docs.dwavesys.com/docs/latest/c_qpu_timing.html`.

[12]D-Wave Systems Inc., *Programming the D-Wave QPU: Setting the Chain Strength* (Apr. 14, 2020).

[13]D. P. DiVincenzo, "The physical implementation of quantum computation", Fortschritte Der Physik-progress of Physics **48**, 771–783 (2000).

[14]H. N. Djidjev, "Logical qubit implementation for quantum annealing: augmented Lagrangian approach", Quantum Science and Technology **8**, 035013 (2023).

[15]B. Duan, J. Yuan, C.-H. Yu, J. Huang, and C.-Y. Hsieh, "A survey on HHL algorithm: From theory to application in quantum machine learning", Physics Letters A **384**, 126595 (2020).

[16]D. M. Eigler and E. K. Schweizer, "Positioning single atoms with a scanning tunnelling microscope", Nature **344**, 524–526 (1990).

[17]A. Finnila, M. Gomez, C. Sebenik, C. Stenson, and J. Doll, "Quantum annealing: A new method for minimizing multidimensional functions", Chemical Physics Letters **219**, 343–348 (1994).

[18]D. M. Fox, K. M. Branson, and R. C. Walker, *mRNA codon optimization on quantum computers*, preprint (Bioinformatics, Feb. 20, 2021).

[19]M. F. Gonzalez-Zalba, S. De Franceschi, E. Charbon, T. Meunier, M. Vinet, and A. S. Dzurak, "Scaling silicon-based quantum computing using CMOS technology", Nature Electronics **4**, 872–884 (2021).

[20]J. F. Grcar, "How ordinary elimination became Gaussian elimination", Historia Mathematica **38**, 163–218 (2011).

[21] J. C. Hafele and R. E. Keating, "Around-the-World Atomic Clocks: Observed Relativistic Time Gains", Science **177**, 168–170 (1972).

[22] R. Harris, M. W. Johnson, T. Lanting, A. J. Berkley, J. Johansson, P. Bunyk, E. Tolkacheva, E. Ladizinsky, N. Ladizinsky, T. Oh, F. Cioata, I. Perminov, P. Spear, C. Enderud, C. Rich, S. Uchaikin, M. C. Thom, E. M. Chapple, J. Wang, B. Wilson, M. H. S. Amin, N. Dickson, K. Karimi, B. Macready, C. J. S. Truncik, and G. Rose, "Experimental investigation of an eight-qubit unit cell in a superconducting optimization processor", Physical Review B **82**, 024511 (2010).

[23] E. Ising, "Beitrag zur Theorie des Ferromagnetismus", Zeitschrift für Physik **31**, 253–258 (1925).

[24] T. Ising, R. Folk, R. Kenna, B. Berche, and Yu. Holovatch, "The fate of Ernst Ising and the fate of his model", Journal of Physical Studies **21**, 3002 (2017).

[25] J. D. Jackson, *Classical electrodynamics*, third edition, international adaption (Wiley, Hoboken, NY, 2021), 811 pp.

[26] S. Jiang, K. A. Britt, A. J. McCaskey, T. S. Humble, and S. Kais, "Quantum Annealing for Prime Factorization", Scientific Reports **8**, 17667 (2018).

[27] M. W. Johnson, P. Bunyk, F. Maibaum, E. Tolkacheva, A. J. Berkley, E. M. Chapple, R. Harris, J. Johansson, T. Lanting, I. Perminov, E. Ladizinsky, T. Oh, and G. Rose, "A scalable control system for a superconducting adiabatic quantum optimization processor", Superconductor Science and Technology **23**, 065004 (2010).

[28] T. Kadowaki and H. Nishimori, "Quantum annealing in the transverse Ising model", Physical Review E **58**, 5355–5363 (1998).

[29] Y. Kim, A. Eddins, S. Anand, K. X. Wei, E. Van Den Berg, S. Rosenblatt, H. Nayfeh, Y. Wu, M. Zaletel, K. Temme, and A. Kandala, "Evidence for the utility of quantum computing before fault tolerance", Nature **618**, 500–505 (2023).

[30] S. Klassen, "The Photoelectric Effect: Reconstructing the Story for the Physics Classroom", Science & Education **20**, 719–731 (2011).

[31] G. Kochenberger, J.-K. Hao, F. Glover, M. Lewis, Z. Lü, H. Wang, and Y. Wang, "The unconstrained binary quadratic programming problem: a survey", Journal of Combinatorial Optimization **28**, 58–81 (2014).

[32] G. Leibniz, "Explication de l'arithmétique binaire", Memoires de l'Academie Royale des Sciences (1703).

[33] A. Lucas, "Ising formulations of many NP problems", Frontiers in Physics **2**, 10.3389/fphy.2014.00005 (2014).

[34] R. A. Millikan, "A Direct Determination of " h ."", Physical Review **4**, 73–75 (1914).

[35] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*, 10th anniversary edition (Cambridge university press, Cambridge, 2010).

[36] L. Onsager, "Crystal Statistics. I. A Two-Dimensional Model with an Order-Disorder Transition", Physical Review **65**, 117–149 (1944).

[37] P. M. Pardalos and S. Jha, "Complexity of uniqueness and local search in quadratic 0–1 programming", Operations Research Letters **11**, 119–123 (1992).

[38] S. Resch and U. R. Karpuzcu, "Quantum Computing: An Overview Across the System Stack", version 3, 10.48550/ARXIV.1905.07240 (2019).

[39] J. J. Sakurai and J. Napolitano, *Modern Quantum Mechanics*, 3rd ed. (Cambridge University Press, Sept. 17, 2020).

[40] R. H. Stuewer, "Historical Surprises", Science & Education **15**, 521–530 (2006).

[41] D. Wang and R. Kleinberg, "Analyzing quadratic unconstrained binary optimization problems via multicommodity flows", Discrete Applied Mathematics **157**, 3746–3753 (2009).

[42] D. Willsch, M. Willsch, C. D. Gonzalez Calaza, F. Jin, H. De Raedt, M. Svensson, and K. Michielsen, "Benchmarking Advantage and D-Wave 2000Q quantum annealers with exact cover problems", Quantum Information Processing **21**, 141 (2022).