Design Science

Comparing gate and annealing-based quantum computing for configuration-based design tasks

Oliver Schiffmann, James Gopsill and Ben Hicks

Design and Manufacturing Futures Lab, University of Bristol, Bristol, UK

Abstract

Complete exploration of design spaces is often computationally prohibitive. Classical search methods offer a solution but are limited by challenges like local optima and an inability to traverse dislocated design spaces. Quantum computing (QC) offers a potential solution by leveraging quantum phenomena to achieve computational speed-ups. However, the practical capability of current QC platforms to deliver these advantages remains unclear. To investigate this, we apply and compare two quantum approaches – the Gate-Based Grover's algorithm and quantum annealing (QA) – to a generic tile placement problem. We benchmark their performance on real quantum hardware (IBM and D-Wave, respectively) against a classical brute-force search. QA on D-Wave's hardware successfully produced usable results, significantly outperforming a classical brute-force approach (0.137 s vs 14.8 s) at the largest scale tested. Conversely, Grover's algorithm on IBM's gate-based hardware was dominated by noise and failed to yield solutions. While successful, the QA results exhibited a hardware-induced bias, where equally optimal solutions were not returned with the same probability (coefficient of variation: 0.248–0.463). These findings suggest that for near-term engineering applications, QA shows more immediate promise than current gate-based systems. This study's contribution is a direct comparison of two physically implemented quantum approaches, offering practical insights, reformulation examples and clear recommendations on the utilisation of QC in engineering design.

Keywords: Configuration design, Quantum computing, Data-driven design, Engineering design

Received 01 July 2025
Revised 11 September 2025
Accepted 11 September 2025

Corresponding author
Oliver Schiffmann
oliver.schiffmann@bristol.ac.uk

Des. Sci., vol. 11, e40
journals.cambridge.org/dsj
DOI: 10.1017/dsj.2025.10028

1. Introduction

Numerical algorithms, simulations and modelling techniques enable designers to explore vast design spaces and identify optimal solutions. Examples include finite element analysis for thermomechanical analyses (Belhocine & Abdullah 2020), generative design for structural optimisation (Gonzalez-Delgado, Jaen-Sola & Oterkus 2023), and Monte-Carlo cost–benefit analyses for supply chain optimisation (Schiffmann *et al.* 2023). Design processes that make significant use of these techniques are often referred to as data-driven design.

All these methods enable designers to discover key information about their problem and its potential solutions. Examples include sensitivity scores against design parameters, the number of valid options, and scores assessing the 'fit' of a solution to a design problem. The role of the designer therefore shifts from the generation of solutions to the definition of design problems and the evaluation of solutions with design options generated and scored computationally (Biskjaer, Dalsgaard & Halskov 2014).

In this new role, designers continue to push the boundaries of what classical computation can achieve in the time constraints of the design process. Designers often wish to evaluate more options, encode more requirements and represent more complex design problems. Each of these factors increases the computational complexity and time for a classical computer to reach a solution. For example, integrated circuit design used to contain a handful of transistors, and their design could be handled manually using relatively simple computational tools. However, today's integrated circuits can contain trillions of transistors (Lécuyer 2022), and their design requires sophisticated computer-aided design software. This increase in transistor count has also led to a dramatic increase in the number of design variables, constraints, and objectives that must be considered.

However, the benefit of classical computational methods is plateauing as problem complexity increases. This limitation primarily stems from the inherent complexity in computationally representing, resolving, and exploring design spaces. This increasing complexity becomes a limitation as we reach the upper end of our manufacturing abilities for classical processors. Despite increasing numbers of transistors the clock speed of classical computers is capped at around 5GHz (Sutter 2005). This indicates we have almost exhausted our vertical scaling potential. As we work in this processing-speed-limited world, understanding and addressing the challenges with problem complexity is crucial to enhance the design process. Key challenges relevant to engineering design include:

- **High-dimensionality:** Design problems often involve a vast number of variables and constraints, resulting in high-dimensional solution spaces. Navigating and optimising these spaces efficiently is a complex task, requiring advanced algorithms and computational power (Gopsill, Schiffmann & Hicks 2022).
- **Trade-offs and multi-objective optimisation:** In many engineering design scenarios, there are conflicting objectives that must be carefully balanced. Achieving the optimal trade-offs between parameters such as cost, performance, reliability and sustainability presents a significant challenge (Sun *et al.* 2018).
- **Design space exploration:** Traditional design approaches often rely on human intuition and experience, limiting the exploration of alternative design options. Expanding the search space to consider a wider and more diverse range of solutions is essential to discovering globally optimal designs.
- **Computational bottlenecks:** As design complexity increases, so does the computational cost and time required to perform detailed analyses, simulations and optimisations. This leads to longer design cycles and potentially hampers the ability to iterate and explore design alternatives.
- **Incorporating uncertainty:** Engineering design is subject to various sources of uncertainty, including material properties, environmental conditions and manufacturing variations. Effectively addressing and quantifying uncertainty is vital for robust design decisions (Schiffmann *et al.* 2023).

Note that limited processing speed only reduces our ability to handle uncertainty by hampering methods that rely on exploring more solutions – such as Monte Carlo simulation. Other methods, such as probabilistic ones, can help to incorporate uncertainty within processing speed constraints.

Gopsill *et al.* (2022) illustrate these challenges through an example 3-stage gearbox design problem. The design parameters for this system were as follows:

- 17 gear options for the 6 gears;
- 5 materials for the 4 shafts;
- 9 bearing options for the 8 bearings; and,
- 8 electric motors for the motor.

While only a relatively small number of options exist for each component choice, the combinatorial nature of the design space results in solution space featuring 5.18×10^{18} possible design options. Evaluating all these solutions would require 164 years if each design option were able to be evaluated in a single clock cycle on a 1GHz processor. This demonstrates the coupling between computational complexity and the scale of design space.

There are some subtleties, however, in that not every design option takes the same time to compute, and one may not need to compute every option to identify the optimum. Also, this gearbox argument uses the full factorial design space. This was chosen not to overstate the limitations of classical approaches but to keep the argument simple and highlight the speed at which problem spaces can grow with only a few variables.

Having evaluated a set of options, it may be useful to store the results for later analysis or selection. Assuming 5 bits are required to store a gear option, 3 bits for the material, 4 for the bearings and 8 for the motors then a total of 15 bits or 2 bytes is required to store a single design option. Multiplying this by the number of possible options yields the storage required to capture the entire design space – 1.036×10^{7} TB. With modern hard disk drives storage capacity in the 10s of TBs, a designer can quickly create a design problem that cannot be resolved and stored in its entirety under classical computation.

The gearbox selection task exemplifies the challenge of navigating vast design spaces, even when the computation of a single design option remains trivial. The computational time to evaluate a single design option can also pose a significant challenge. Work that exemplifies this problem is concerned with the design of winding strands in electrical machines – specifically their lay and how it affects AC losses (Mellor, Hoole & Simpson 2021; Hoole *et al.* 2021). The authors discuss the development of a conductor placement algorithm. For the algorithm to fill a single slot with approximately 130 conductors, a run time of around 3 hours is required. This is for a single slot, of which there will be multiple in an electric machine. Further, this is for a single set of design parameters. Should the designer wish to explore even a small number of design options this task quickly becomes intractable. Figure 1 demonstrates how the challenges of computing many solutions and computing complex solutions combine to create the classically tractable problem space.

To overcome these challenges, designers use design exploration (often referred to as meta-heuristics) and optimisation algorithms that intelligently navigate the problem space to find optimal solutions. Examples include gradient descent methods, evolutionary algorithms, particle swarm optimisation and generative approaches. Statistical methods also exist, such as those developed by Genichi Taguchi, that focus on eliminating variables of lower impact from the problem to shrink design spaces and make them more manageable (Ghani, Choudhury & Hassan 2004).

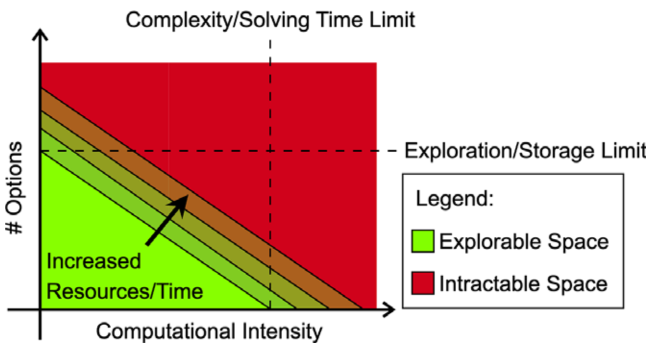


Figure 1. A figure showing how the type of problems we can tackle using classical computation are bounded by the number of options we can compute and the complexity of evaluating a single option.

However, these each face their own issues as design spaces become increasingly complex. For example, gradient descent struggles with convergence to local optima and traversing discontinuous design spaces (Gopsill, Johns & Hicks 2021). At these levels of increased problem complexity, the limitations of classical computational methods become more apparent, particularly as the number of variables, constraints and objectives grows. This presents a challenge in a world where designers seek to continuously improve existing solutions, evaluate trade-offs and integrate the latest knowledge into their design process. As a result, there is a growing need for methods that can more efficiently represent and manipulate design information.

1.1. A quantum computing primer

Quantum computing (QC) uses quantum phenomena to represent and process information, which could overcome some of the barriers faced by designers using classical computation. The field of QC remains an evolving field, within which there are a variety of techniques being developed. Figure 2 provides a simple comparison between two classical and two quantum algorithms, which are the focus of this paper, applied to a layout style design puzzle. This is intended to give readers unfamiliar with QC an idea as to how these quantum phenomena can be leveraged to enable different approaches to computation.

The smallest unit of information represented in a quantum computer is a quantum bit – referred to as a qubit. Classical computers process information in binary states (0 or 1). The power of a qubit lies in its ability to exist in a state of 0, 1, or both simultaneously. This property, known as superposition, is a core principle of quantum mechanics.

An intuitive analogy for superposition is a spinning coin. Whilst spinning, it is neither heads nor tails but has a probability associated with both outcomes. Only when it lands does it settle into a single, definite state (either heads or tails). Similarly, during the execution of a quantum computation a qubit can remain in a superposition of 0 and 1.

Another crucial quantum phenomenon is entanglement. This occurs when two or more qubits become linked in such a way that their fates are intertwined, regardless of the distance separating them. To extend the coin analogy, imagine

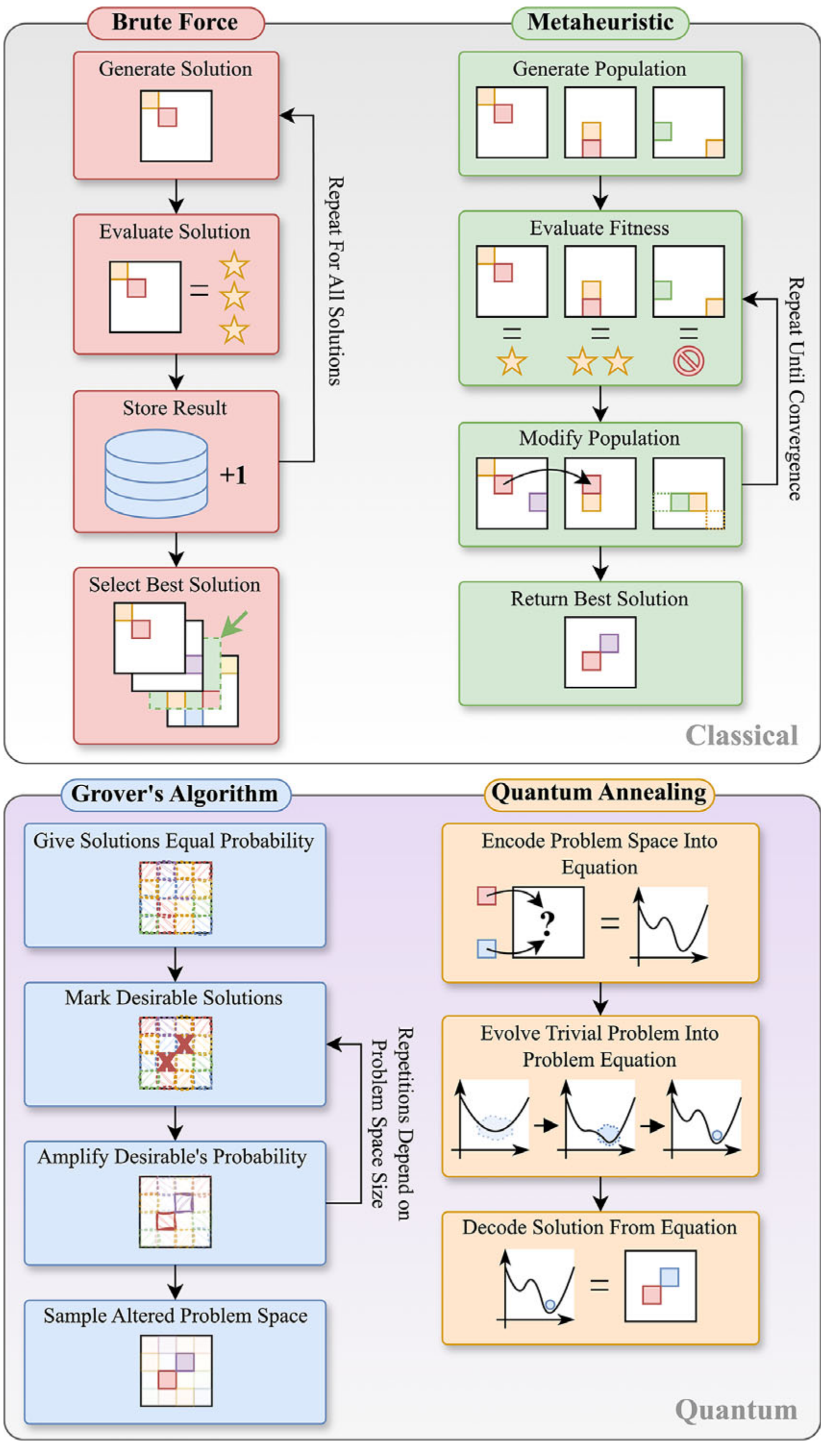


Figure 2. A figure showing the comparison between two classical and two quantum algorithms applied to a design puzzle. Here the puzzle is a layout type problem requiring the placement of entities in a grid.

two “entangled” coins. The outcome of one coin’s flip immediately influences the outcome of the other. This interconnectedness allows for creating complex computational states and is essential for many quantum algorithms.

While these analogies are helpful, the behaviour of qubits is formally described through Dirac notation. The state of a qubit is represented by a vector called a “state vector.” For a single qubit, the two basic states, the “heads” and “tails” of our system, are defined as:

- The state 0, represented by the notation$|0\rangle$
- The state 1, represented by the notation$|1\rangle$

A qubit in superposition is described as a linear combination, or a weighted sum, of these two basic states. This is shown mathematically in Equation (1):

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

where $|\phi\rangle$ represents the superposition state created by superimposing the two binary states $|0\rangle$ and $|1\rangle$, α and β are their linear coefficients, respectively. These are complex numbers called probability amplitudes. Operations can then be performed on a state, $|\phi\rangle$, as part of a computational process. $|\phi\rangle$ can eventually be measured to extract an answer from the quantum computer. This answer would be one of the basis states $|0\rangle$ and $|1\rangle$ and the probability of each being returned would be $|\alpha|^2$ or $|\beta|^2$, respectively (Yingchareonthawornchai, Aporntewan & Chongstitvatana 2012).

A “universal” quantum computer can be used to perform the same tasks as classical computers. However, quantum computer specific algorithms can take advantage of quantum phenomena to provide a speed-up by skipping steps required in classical algorithms. The superposition shown in Equation (1) is one example of such quantum phenomena. Qubits can exist in a state of superposition. This allows a single qubit to represent multiple possibilities at once, increasing the information multiple qubits can hold compared to a classical bit.

Grover’s Algorithm is a classic example of a quantum algorithm. Designed for unstructured search, it provides a theoretical quadratic speed-up. Similarly, Shor’s algorithm, designed for integer factorisation, holds significant implications for cryptography (Nielsen & Chuang 2001). In the Section 3.2.1 a more comprehensive explanation of how we can create an algorithm using the building blocks of qubits is provided, as well as an explanation of Grover’s algorithm since it is a key approach investigated in this work. These quantum algorithms are primarily designed within the framework of gate-based quantum computation. In this paradigm qubits are manipulated through quantum gates, analogous to classical bits and logic gates in classical computing.

An alternative to gate-based quantum computation is quantum annealing (QA). Simply put, QA uses quantum phenomena to find the minimum value of a function, making it well-suited for optimisation problems. It leverages quantum entanglement and quantum tunnelling (Venegas-Andraca *et al.* 2018; Hauke *et al.* 2020) to define an energy landscape (similarly to that used in many classical optimisation techniques) through a combination of penalty functions. In this landscape, more optimal solutions have lower energies (less optimal ones experience higher positive penalties). QA can be thought of as an analogue form of computation (Yang, Zolanvari & Jain 2023) as it involves running a physical process that evolves over time and is described by a Hamiltonian. It is also called analogue as it can only solve problems of a certain form, whereas gate-model computation is often called digital because theoretically it can perform any type of computation.

A Hamiltonian is a mathematical construct that represents the time evolution of quantum states. The Hamiltonian, $H(t)$, that represents the energy landscape during QA is shown in Equation (2):

$$H(t) = A(t)H_0 + B(t)H_1 \tag{2}$$

where $A(t)$ and $B(t)$ are values varied from 0 to 1 and 1 to 0 over the course of the computation, respectively, and H_1 is some initial Hamiltonian in its ground state and H_0 is the Hamiltonian whose ground state encodes an optimal solution (lowest energy in the energy landscape) (Hauke *et al.* 2020). Here, you can think of the process as slowly morphing one energy landscape into another. We start with a simple, known landscape, H_0, whose lowest point is easy to find. We then slowly turn on the Hamiltonian for our actual engineering problem, H_1, which contains the complex landscape we want to explore. The functions $A(t)$ and $B(t)$ act like control knobs, smoothly decreasing the influence of the simple landscape from 1 to 0 while increasing the influence of the problem landscape from 0 to 1.

If this evolution is slow enough, or adiabatic, such that no energy is added then the final solution should be the global optimum. This is an interesting property for engineering designers as it offers the potential to avoid local minima in rugged or discontinuous design spaces (Koshka & Novotny 2020; Koh & Nishimori 2022).

QA is not using past results to generate and investigate a new population of possible solutions, as many popular classical methods do. Instead, the process begins with an initial quantum state that encodes the known optimal solution to a simpler problem. This state is then evolved until it represents a solution to the desired problem. This makes the probability of achieving a globally optimum solution a function of different variables. This means there may be possible problem scenarios that would benefit from a QA approach.

It should be noted that unlike some gate-based algorithms (see the Section 3.2.1), the expected speed-up for QA has not been theoretically quantified. However, there is evidence that QA can outperform classical alternatives for certain classes of problems (Rajak *et al.* 2023). In fact, during the review of this manuscript D-Wave, a major QA company, have claimed to be able to achieve classically impossible results through QA (King *et al.* 2025). This development is a result of QA systems maturing considerably in recent years (Hauke *et al.* 2020). This hardware advancement affirms QA as a suitable area for exploration in this Noisy Intermediate-Scale Quantum (NISQ) era. NISQ devices are those devices that, while capable of outperforming some classical approaches, suffer from substantial error in their results due to quantum noise (Dasgupta & Humble 2020).

1.2. Research opportunity

An important feature of quantum computing research is the recognition that both gate-based and QA methods hold promise in the near and long term. This realisation underpins the importance of exploring all avenues until a dominant method emerges such that we can take full advantage when/if it does. The field is still in its infancy and development of hardware is rapid. Many approaches exist, including those just discussed and hybrid versions – approaches that combine classical and quantum processing. It is unclear exactly how we will gain "advantage" over classical computation using QC. For engineers a central challenge emerges: how to represent problems effectively to exploit the unique strengths of both gate-based and annealing approaches? Further, a gap exists in the current research that specifically addresses how to implement the developing approaches within existing hardware. As such, it is of value to engineering designers to explore a variety of QC methods to evaluate their potential in supporting design processes.

It is timely as recent years have seen an acceleration in the development and maturity of quantum hardware, a trend summarised in Figure 3. This progress has not only increased the number of available qubits and the variety of hardware architectures but has also democratised access to these systems. Researchers can now execute algorithms on various quantum computers through cloud platforms. Furthermore, the development process has been significantly streamlined, with software development kits (SDKs) emerging that provide a unified interface to a multitude of quantum backends, including both physical hardware and increasingly powerful classical simulators. This consolidation simplifies the process of comparing different quantum approaches and hardware, lowering the barrier to entry for engineers.

The contribution of this work provides an insight into the readiness of the different hardware options for a generalised instance of a configuration-based design task involving large numbers of options/combinations and comprising many constraints. Configuration/layout design is a common class of problem in engineering (Khalafallah & El-Rayes 2011; Dimitrova & Maréchal 2015; Chen *et al.* 2021), and this work utilises a generalised (non-context specific) instance of this class.

This insight could be used to eliminate the investigation of certain options for unsuitable problem architectures, as well as provide examples for how problems can be refactored to be quantum-solvable. It also provides readers with a demonstration of how the same problem can be formulated in different ways and the implications of such a formulation. In addition, a methodology is laid out for applying two QC methods to an engineering design problem. This methodology is applied to two case studies, giving a demonstration of the process from problem choice all the way to obtaining results from today's quantum devices. Finally, this methodology and its efficacy are reflected on to suggest improvements.

This paper continues with a Related Work section, which provides the reader with an overview of the ways in which some quantum methods have already been employed by engineering designers and to what extent they have considered the available hardware in their applications. This is followed by the Experimental Setup, covering the steps taken throughout the study to achieve results enabling hardware, results, and problem formulation comparison. The results achieved via the experimental process are then presented, showing how the different quantum approaches perform. The implications of these results are then explored in the Discussion section. Finally, the contribution of this paper is summarised in the Conclusion.

2. Related work

The Related Work provides a summary of the field investigating the applications of QC to justify the timeliness and nature of the investigation. QC application areas is

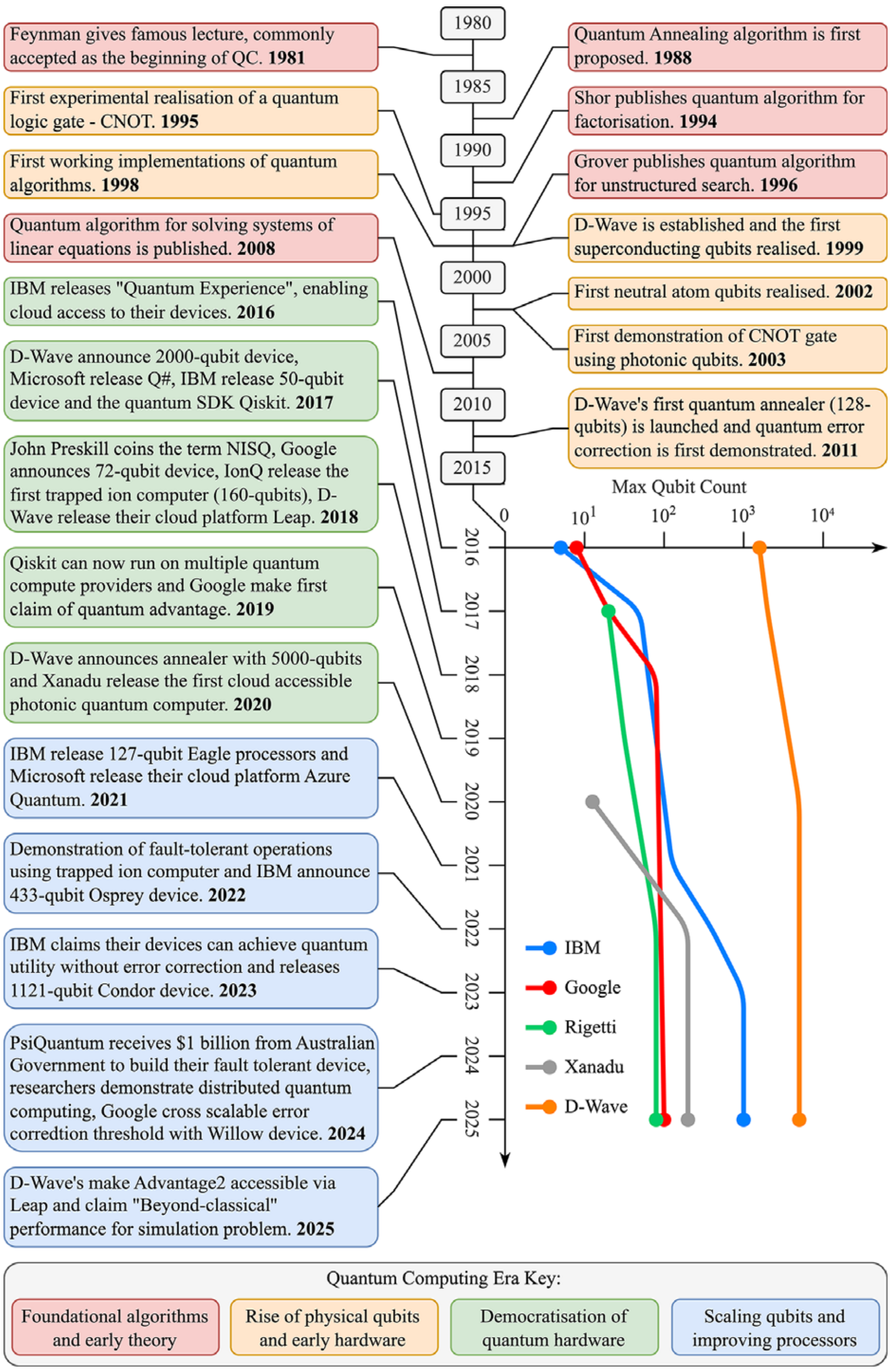

1980
1985
1990
1995
2000
2005
2010
2015
Feynman gives famous lecture, commonly accepted as the beginning of QC. 1981
First experimental realisation of a quantum logic gate - CNOT. 1995
First working implementations of quantum algorithms. 1998
Quantum algorithm for solving systems of linear equations is published. 2008
IBM releases "Quantum Experience", enabling cloud access to their devices. 2016
D-Wave announce 2000-qubit device, Microsoft release Q#, IBM release 50-qubit device and the quantum SDK Qiskit. 2017
John Preskill coins the term NISQ, Google announces 72-qubit device, IonQ release the first trapped ion computer (160-qubits), D-Wave release their cloud platform Leap. 2018
Qiskit can now run on multiple quantum compute providers and Google make first claim of quantum advantage. 2019
D-Wave announces annealer with 5000-qubits and Xanadu release the first cloud accessible photonic quantum computer. 2020
IBM release 127-qubit Eagle processors and Microsoft release their cloud platform Azure Quantum. 2021
Demonstration of fault-tolerant operations using trapped ion computer and IBM announce 433-qubit Osprey device. 2022
IBM claims their devices can achieve quantum utility without error correction and releases 1121-qubit Condor device. 2023
PsiQuantum receives $1 billion from Australian Government to build their fault tolerant device, researchers demonstrate distributed quantum computing, Google cross scalable error corredtion threshold with Willow device. 2024
D-Wave's make Advantage2 accessible via Leap and claim "Beyond-classical" performance for simulation problem. 2025
Quantum Annealing algorithm is first proposed. 1988
Shor publishes quantum algorithm for factorisation. 1994
Grover publishes quantum algorithm for unstructured search. 1996
D-Wave is established and the first superconducting qubits realised. 1999
First neutral atom qubits realised. 2002
First demonstration of CNOT gate using photonic qubits. 2003
D-Wave's first quantum annealer (128-qubits) is launched and quantum error correction is first demonstrated. 2011
Max Qubit Count
0
10^1
10^2
10^3
10^4
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
IBM
Google
Rigetti
Xanadu
D-Wave
Quantum Computing Era Key:
Foundational algorithms and early theory
Rise of physical qubits and early hardware
Democratisation of quantum hardware
Scaling qubits and improving processors

Figure 3. A figure showing the major developments in QC relevant to engineering design. This figure shows how the field is rapidly progressing from creation to developing deployable methods and the required supporting hardware. The sources for the information presented in this figure are detailed in Appendix B: Source for Figure 3 inside Tables B1 and B2.

a young yet rapidly maturing field (Gill *et al.* 2022; Mahmoudi *et al.* 2022) with the earliest works dating 2001 (Williams 2001). The research has been grouped according to types of QC hardware:

- dedicated (meaning only quantum hardware contributes significantly to the computation) gate-based;
- dedicated QA based;
- hybrid (quantum-classical) versions of gate-based computation;
- hybrid QA

Hybrid quantum algorithms can be defined as one requiring non-trivial amounts of both quantum and classical computation and could not be sensibly described without reference to the classical computation (Callison & Chancellor 2022). This definition is necessary as almost all forms of QC currently require at least some amount of classical computing (this could be as small as repeating the quantum algorithm). Variational QC is a subset of hybrid quantum computation that represent a promising avenue for NISQ usefulness. They rely on a classical optimiser to tweak elements of a quantum algorithm, improving the results of the quantum algorithm with each iteration (Cerezo *et al.* 2021). It is also worth noting that QA is not considered by all to truly be QC as QA devices cannot be programmed to tackle any problem. However, it is included in this comparison as it is a novel means of computation that utilises quantum phenomena to potentially overcome some of the barriers faced by classical methods. Additionally, it is a more mature option for utilising quantum phenomena in this way and gives us something to compare other methods against. For simplicity, it will continue to be referred to as QC in this paper.

There is a growing body of research targeting each of these areas as QC remains a rapidly developing field without an established industry standard approach for application. Ullah *et al.* (2022) discuss how QC might help manage the increasing computational complexity of smart grids. The complexity has come from the shift away from fossil fuels to more distributed energy resources (DERs), such as photovoltaic cells, wind turbines and electric vehicles. The non-linearity and uncertain nature of these DERs increases the complexity of smart grid decision. The algorithms discussed include dedicated gate-based approaches such as the quantum Harrow–Hassidim–Lloyd algorithm (for tackling systems of linear equations), hybrid approaches such as the Variational Quantum Eigensolver algorithm (a hybrid quantum-classical computational approach for finding the ground state of a Hamiltonian), QA approaches for minimising objective equations and quantum machine learning (QML). This review highlights the fact that we need a robust methodology for identifying and testing promising QC approach and problem pairings as there are numerous potential options.

2.1. Gate-based applications

Dalal *et al.* (2024) investigate the use of a quantum algorithms to solve NP-hard logistics scheduling problems. The study focuses on two specific use cases: the travelling salesperson problem and an industrial job-shop scheduling problem for an automated laboratory robot. The study investigated a both a pure and hybrid version of the same newly proposed quantum algorithm for solving combinatorial optimisation problems. Their performance was benchmarked against existing quantum methods for solving combinatorial problems like the quantum approximate optimisation algorithm (QAOA) and digitised quantum annealing. These algorithms were implemented and tested on cloud-based quantum hardware, including IonQ's trapped-ion processors and IBM's superconducting processors. This study exemplifies the need to explore across the different avenues for applying QC.

Another example of research investigating a hybrid gate-based approach is the report by Liu *et al.* (2024). Their paper demonstrates an integrated pipeline that combines the classical finite element method (FEM) with a quantum algorithm to solve eigenvalue problems in solid mechanics and structural engineering. The primary goal is to calculate the fundamental natural frequency for three distinct mechanical systems. The quantum algorithm used is the variational quantum Eigensolver (VQE), a hybrid quantum-classical method that finds the minimal eigenvalue of a system. The framework was implemented on the IBM Qiskit platform, with studies conducted on a noise-free simulator and final demonstrations performed on IBM's quantum processors. While the results on real hardware showed larger errors compared to the simulator, the work successfully validates the integrated FEM-VQE methodology for mechanical analysis.

Examples of work exploring the use of simulated methods include Correa-Jullian *et al.* (2022) and Li *et al.* (2022). Both were looking at fault diagnosis in wind turbines and roller bearings, respectively. Their QML approaches improve the ability of machine learning algorithms to process large amounts of data. Correa-Jullian *et al.* (2022) operated on a dataset containing 42.2 million useful data entries and Li *et al.* (2022) used data sets containing 4 and 16 million sample points. Li *et al.* (2022) state that QC hardware with sufficient coherent times are not available at present and so explored a simulated approach to demonstrate the feasibility of their quantum support vector machine approach. Correa-Jullian *et al.* (2022) also explore a simulated quantum approach and compared it to classical approach implemented on a GPU. Correa-Jullian *et al.* (2022) found that the proposed quantum approach was comparable to conventional ML models and outperformed some other models while Li *et al.* (2022) showed that a fault diagnosis model based on a quantum least square support vector machine was feasible.

2.2. Quantum annealing applications

Moving on from gate-based approaches, QA has also shown promise in practical applications, with several successful implementations in industry reported by D-Wave Systems (D-Wave n.d.c). These successes suggest that QA is a viable tool for solving complex engineering problems, even in the absence of a demonstrated quantum speed-up. The relative ease of use of QA algorithms, which do not require deep-domain expertise in the classical algorithms they aim to outperform (Hauke *et al.* 2020), further enhances their appeal for engineering applications.

Morstyn (2023) discusses the practical implementation of annealing-based quantum computing for combinatorial optimal power flow problems. These are problems that involve arranging controllable power sources to meet demand at a minimum cost – a problem complicated by the rise of DERs. In the paper, results obtained from the real D-wave quantum processing unit (QPU) are presented and compared against classical simulated annealing. The author concludes that while current quantum processors are too small for realistic, distribution-scale applications, the required number of qubits scales linearly with the number of electric vehicles and network constraints, suggesting a promising opportunity for the future as quantum hardware continues to develop. This work is an example of a body of literature focused on the application of current quantum annealers to industrial problems. This may indicate that these devices are more mature than alternative quantum options and provide a viable option for gaining a quantum advantage in the near term, an opinion supported by Leymann & Barzen (2020).

The use of hybrid annealing devices has also been explored by Quinton *et al.* (2025). This paper benchmarks the performance of D-Wave's hybrid solver against classical solvers for a real-world mixed-integer linear programming (MILP) problem. The chosen case study is the Unit Commitment problem, a complex optimisation challenge that involves scheduling power generation to meet demand at minimum cost. The problem was tested at three different scales, including a full-scale version with over 44,000 variables and two smaller, reduced versions. The D-Wave constrained quadratic model (CQM) hyrbid solver was utilised and its performance was compared to industry-leading classical solvers. For the full-scale problem, the hybrid solver failed to find any feasible solutions with its default settings and produced highly suboptimal results even with significantly increased run times. On the smaller, reduced-scale problems, the solver found feasible solutions, but they were still far from the optimal values found classically and took longer to compute. The authors conclude that while D-Wave can solve MILP problems, its hybrid solver is not currently effective for this type of large-scale linear problem, suggesting that a computational advantage is likely limited to problems with inherent quadratic structures. This work suggests that we should continue to investigate annealing approaches to find where they can be most effectively applied in engineering design.

These recent works are summarised in Table 1 and show that there is a wealth of recent literature exploring applications for QC to engineering relevant problems. Additionally, these approaches are making use of the entire suite of available hardware. This highlights a need for comparison between approaches and an

Table 1. A table summarising each of the related works discussed in this section

Ref.	Engineering problem	Approach
Ullah *et al.* (2022)	Increased complexity in smart grids due to DERs	Summary of available QC techniques
Dalal *et al.* (2024)	NP-hard logistics and scheduling problems	Dedicated and hybrid versions of gate-based algorithm using real QPUs
Liu *et al.* (2024)	Finding natural frequency for mechanical systems	Hybrid gate-based algorithm using real and simulated QPUs
Correa-Jullian *et al.* (2022)	Fault detection in wind turbines	QML using simulators
Li *et al.* (2022)	Fault detection in roller bearings	QML using simulators
Morstyn (2023)	Optimal power flow problems	Dedicated quantum annealing using real QPUs
Quinton *et al.* (2025)	Scheduling power generation to meet demand	Hybrid quantum annealing using real QPUs

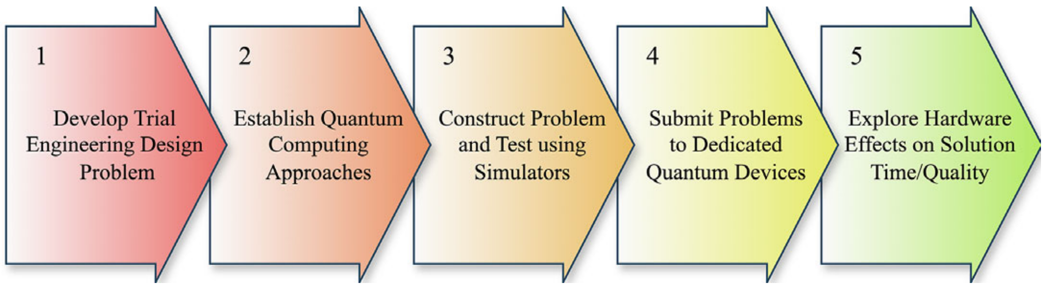

1
Develop Trial Engineering Design Problem
2
Establish Quantum Computing Approaches
3
Construct Problem and Test using Simulators
4
Submit Problems to Dedicated Quantum Devices
5
Explore Hardware Effects on Solution Time/Quality

Figure 4. A figure showing the major steps to be executed as part of the experimental process.

investigation tailored to the engineering design community so that we are not left behind. The works also help to contextualise the approaches explored here by showing which options were left unexplored.

3. Experimental setup

Figure 4 details the five steps of the experimental setup designed to study the ease of translating a constraint-based design optioneering problem to a form solvable via QC and produce results using real-world QC machines. Here optioneering is defined as a design approach focused on systematically generating and evaluating a wide range of valid solutions, or “options,” to a complex problem. Step 1 defined the constraint-based design optioneering problem to be solved. Step 2 explores the identified QC approaches and their requirement for use. Step 3 involves translating the optioneering problem into a form appropriate for solution using the identified method. Step 4 details the analysis used to establish the feasibility of each approach. Step 5 details the analysis used to evaluate the suitability/readiness of the QC methods.

3.1. Step 1: problem design

A problem was generated that was representative of a typical engineering design problem. The selected problem builds on the constraint-based tile placement problem first presented in Gopsill *et al.* (2021) and then developed in Gopsill *et al.* (2022). The problem is analogous to many engineering design problems, including:

- Very large-scale integration **(VLSI) chip design:** In VLSI chip design, engineers need to place numerous electronic components like transistors, logic gates, and interconnects efficiently on a silicon wafer. The objective is to minimise the size, power consumption and maximise the performance of the chip (Held *et al.* 2011).
- **Factory floor layout:** In manufacturing, optimising the layout of machines and workstations on the factory floor is crucial for efficiency and productivity. Engineers need to consider factors such as workflow, material handling and safety (Ripon & Torresen 2014).
- **Optical fibre network design:** In telecommunications, engineers design optical fibre networks by deciding where to lay fibre optic cables to connect cities or

regions efficiently. This involves minimising the total cable length while considering geography (Ranaweera *et al.* 2019).

- **Placement of sensors in environmental monitoring:** In environmental monitoring, the placement of sensors (e.g., for air quality or weather) is critical. Engineers need to determine the best locations to obtain accurate data while minimising costs (Al-Turjman, Hassanein & Ibnkahla 2013).

The problem affords the ability to compare between the processes of implementation for different quantum approaches (an approach being a combination of algorithm, executing hardware and the software required to interface with the hardware) as well as the performance. The problem was also not so complex as to make these comparisons difficult or prohibit the investigation into what is currently an emerging technology.

The examples presented above give an idea of how ubiquitous these layout-style, or configuration design, tasks are within the study of engineering design. Creating a simple abstracted version of these kinds of problems keeps the findings generalisable to the wider engineering design audience. To construct this generalised configuration design problem we can consider the constraints and objectives common across different examples found in literature (such as those already presented). Features common to configuration design problems are:

- The task is to arrange a series of entities (PCB components, sensors, machinery, etc…) within a physical space.
- Certain positions are out of bounds for placement (a component or feature already exists there in the chip or factory, some prohibitive geography prevents laying cable or sensing is not allowed).
- Entities cannot occupy the same space (components, features, facilities cannot occupy the same physical space or use the same resources).
- There is at least one objective with which solutions can be ranked (minimise the total area required for PCB components, maximise the area covered by sensors, keep the centre of gravity for a ships cargo as close to the centre line as possible).

3.1.1. Formal problem statement

These features were used to define the tiling problem depicted in Figure 5. The goal is to find the optimal placement of two tiles on an 8x8 grid according to a defined objective and a no-overlap constraint. The problem is defined by 12 binary variables (6 for each tile, 3 for each tile coordinate). For each tile $t \in \{1,2\}$, its position on the grid, denoted by the coordinate pair (X_t, Y_t), is determined by a set of binary variables:

- X-coordinate variables:$x_{t,1}, x_{t,2}, x_{t,3} \in \{0,1\}$
- Y-coordinate variables:$y_{t,1}, y_{t,2}, y_{t,3} \in \{0,1\}$

The integer value for each coordinate (from 0 to 7) can be derived from these binary variables using the following conversion:

$$\begin{aligned} X_t &= 4x_{t,1} + 2x_{t,2} + x_{t,3} \\ Y_t &= 4y_{t,1} + 2y_{t,2} + y_{t,3} \end{aligned}$$

The objective is to place the tiles as close to the eastern wall as possible. This could correspond to maximising the sum of the integer X-coordinates of both tiles, given below as Z.

$$\text{Maximise} \quad Z = \left(\left(4x_{1,1} + 2x_{1,2} + x_{1,3}\right) + \left(4x_{2,1} + 2x_{2,2} + x_{2,3}\right)\right) \tag{3}$$

There is one primary constraint: the two tiles cannot be placed in the same grid position. This means the coordinate pair for Tile 1 must not be equal to the coordinate pair for Tile 2.

$$(X_1, Y_1) \neq (X_2, Y_2) \tag{4}$$

The grid was varied in size – 16x16, 32x32, and 64x64 – to explore the scaling behaviour of the gate and annealing approach, however, the initial configuration for testing is an 8x8 grid as depicted in Figure 5. An 8x8 grid problem setup results in $64^2 = 4096$ potential solutions with 56 unique valid solutions. The tiles were considered non-identical so a switch of positions would be also considered valid and unique solution.

Since this is a rather universal problem, with some real world examples having nearly unlimited constraints (e.g., city planning has many stakeholders enabling numerous complex constraints/objectives) it is likely that a problem of this form (or a problem that could be formulated this way) has reached a fidelity where the classical methods employed for solution generation begin to struggle.

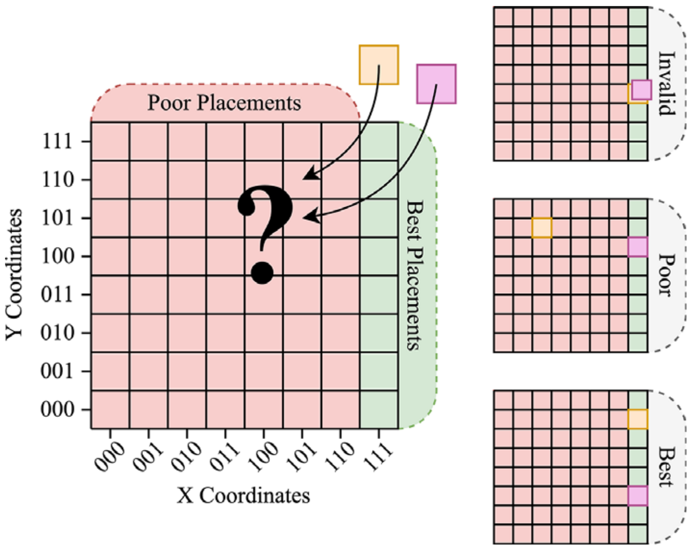


Figure 5. A figure showing the combinatorial problem of placing two tiles (orange and pink) in an 8x8 grid. Grid coordinates are shown represented by binary strings (000, 001, 010…). The tiles preferred positions are shown by the green area. Tiles placed closer to this eastern wall/green area are considered better solutions. Three example solutions are shown. The top example shows both tiles overlapping as representing an invalid solution.

3.2. Step 2: quantum approaches

The study explored the applicability of quantum gate and annealing-based approaches. The two approaches were compared against a classical brute-force approach to provide conclusions about the readiness of QC for use in engineering design. Both a simulated and real-world implementation of the approach were tested. Simulation of QC will not yield any computational benefits but was performed to identify if issues in the results should be attributed to quantum hardware's limitations or problem formulation.

These approaches were selected primarily to demonstrate how problem formulation can vary between two of the leading approaches to QC. The problem formulation for the gate-based approach (Grover's algorithm) was carried over from the authors' previous work (Gopsill *et al.* 2021, 2022). This meant that only one additional formulation needed development for this study, enabling the results to be shared more quickly – which is important in such a fast moving field. QA used a very different problem formulation, therefore this study details 2 reference models for engineering designers.

Further justification for the inclusion of Grover's algorithm is its design for use with a gate-model (or digital) quantum computer. An end goal for the development of this kind of QC hardware is a universal QC, one that can perform any kind of computation that a classical device could. A criticism of approaches developed for NISQ hardware (such as QA) is that they are designed to operate on machines with developmental constraints. This means their code must be continually updated alongside the hardware to remain efficient. The problem formulation discussed for the gate-model approach in this study should be implementable on this future hardware which is free from NISQ quirks. Such a device is often called a fault-tolerant quantum computer (FTQC) as it has been developed to the point where it can tolerate the errors that affect QC. Therefore, these two approaches should provide the study with a balance between near and long-term relevance.

3.2.1. Quantum gate model

Grover's algorithm is a method for searching through an unstructured database, executable on a gate-model quantum computer. This means that it is a set of rules, understandable by the archetypal quantum computer, which finds an entry or entries that possess a certain property in a database which is completely unsorted. As an example, imagine searching for all the books with titles that contain "design methodology" in a bookshelf that has never been organised.

This algorithm is of great interest as the operations required to find the desired solution are reduced versus if you attempted the same problem using a classical computer. This means we would be able to tackle a larger problem in the same space of time. An example of the impressive potential improvement offered by a quantum search algorithm is given in Nielsen & Chuang (2001). If you were given a map of many cities and wished to find the shortest route passing through all of them, one option would be to search all possible routes through every city whilst maintaining a record of the shortest route so far. On a classical computer, if there are N possible routes then it would take $O(N)$ operations. Grover's algorithm requires only $O(\sqrt{N})$ operations.

How can we create an algorithm such as this from the building blocks of qubits? The most common approach to quantum algorithm construction can be understood through the classical analogue of logic gates. In a classical computer bits are used to represent information, perhaps a problem variable and logic gates which are built up from transistors are used to operate on the information. In a quantum computer, a qubit can be used to represent a problem variable and quantum gates perform operations on the qubits. An example that is consistent in classical and quantum computation would be the NOT gate. The difference between the classical and quantum approach is that quantum bits have more properties and quantum gates can perform more operations using these properties. This enables the quantum algorithms to skip steps that would be required in a classical algorithm. This means that the quantum algorithm has the potential to find the solution in fewer steps. Examples of such gates are the Hadamard gate, $\boxed{H}$, which creates quantum superpositions, and the Controlled NOT (CNOT) gate, $\boxed{\oplus}$, which enables entanglement between qubits.

This logic gate style algorithm construction is typically applied using gate-based quantum computers. Algorithms in such devices can be depicted using circuit diagrams, which are graphical models that depict the evolution of qubits through a series of quantum gates. The circuit combines quantum gates, resulting in a procedure that transforms the initial state (of qubits) into a final state, which encodes a potential solution to the problem (Nielsen & Chuang 2001). For more information on the quantum gate model, circuits and their use, see Nielsen & Chuang (2001) for a coverage of the fundamentals. Alternatively, the tutorials available on the IBM Quantum Learning platform (IBM 2023b) offer a more implementation focused explanation using Qiskit. A pictorial representation of Grover's algorithm is presented in Figure 2.

A more detailed explanation as to how this gate model approach can be used to construct such an algorithm can be found in Appendix A. It should be noted that the reader need not fully understand the operation of Grover's algorithm to understand the contribution of this work. The key point to understand from this section is the gate-model approach to QC – utilising quantum gates to encode solutions in the states of qubits before a measurement is applied.

3.2.2. *Problem formulation for gate-based algorithms*

This study continues from the authors previous work where simulated results were achieved for Grover's algorithm applied to the same tiling problem. To solve a problem using a gate-based approach, the problem variables must be represented by a register (collection) of n qubits. This requires the problem to be represented by binary variables. The representation of of this tiling problem via qubits is presented in the Section titled "Grover's Algorithm using Qiskit." A detailed version of the problem formulation can be seen in the authors previous work (Gopsill *et al.* 2021, 2022).

3.2.3. *Quantum annealing*

QA is a heuristic quantum optimisation algorithm for solving complex combinatorial optimisation problems. Adiabatic quantum computation (AQC) is a theoretical framework for quantum computing that utilises the adiabatic theorem (Amin 2009) from quantum mechanics to perform computations. This theorem states that if a quantum system starts in its ground state and the Hamiltonian of the system is changed slowly enough, the system will remain in its ground state at all times – Equation (2). This property of adiabatic evolution allows AQC to solve optimisation problems by transforming the initial Hamiltonian, which represents the starting problem, into a final Hamiltonian, which encodes the solution.

QA is a specific implementation of AQC, it is the generic name for quantum algorithms that use quantum mechanical fluctuations (quantum tunnelling) to search for the solution of an optimisation problem (Morita & Nishimori 2008). QA's classical counterpart, simulated annealing (SA), can be used to help understand the process. "Quantum tunnelling between different classical states replaces thermal hopping in SA" (Morita & Nishimori 2008).

The main challenge for QA is evolving the system slowly enough (adiabatic-ally enough) such that it does not jump from the lowest energy state to a higher energy state. The minimum disparity between the lowest energy state and the next lowest one is called the energy gap. Issues arise when this gap becomes so small such that the time required to avoid crossing it becomes infeasibly long (Hastings 2021). There is not a general predicted speed up for QA as there is for Grover's algorithm, however, the required annealing time does scale inversely with energy gap size (Hauke *et al.* 2020). As the complexity of the problem increases (more constraints, objectives and variables) then the energy landscape can become more "rugged," the energy gap will shrink and the annealing time will need to increase, or more samples will need to be taken.

3.2.4. Problem formulation for QA using D-wave devices

With an understanding of how QA differs from gate-based approaches, problem formulation for QA devices can be discussed. QA can be considered as a form of analogue computation and so the problem formulation is dependent on the hardware chosen (Yang *et al.* 2023). For this work, the QPUs offered by the company D-wave were chosen. D-wave provide instructional guides on how to use, formulate problems, and submit jobs to their devices. This implementation can be done in Python using their Ocean SDK (D-Wave n.d.b).

There are three classes of problem that can be tackled when considering the whole suite of D-wave devices. These are binary quadratic models (BQM), discrete quadratic models and constrained quadratic models (CQM). Each of these can handle a different class of variable, binary, discrete, and integer, respectively (D.-W. Developers 2022). However, to make use of a dedicated device you must formulate the problem as BQM.

BQMs are made up of two classes. These are quadratic unconstrained binary optimisation (QUBO) problems and Ising models. These are mathematically equivalent but some problems may see better results with one option. For this work the QUBO approach was selected. It is also possible to convert between the approaches once the problem has been formulated. The QUBO approach followed for this work was taken from the resources provided by D-wave in their "Problem Formulation Guide" (D-Wave n.d.b). It should be noted that problem formulation is restricted to QUBOs instead of higher-order polynomial unconstrained binary optimisation problems due to experimental devices only being able to handle 2-local interactions (Hauke *et al.* 2020). The general form for a QUBO can be seen in Equation (5)

$$min\left(\sum_i a_i x_i + \sum_i \sum_{j>i} b_{i,j} x_i x_j + c\right) \tag{5}$$

where a_i and $b_{i,j}$ are constants that are chosen to define the problem, x_i and x_j are the binary variables for the problem, and c is a constant term. Note that Equation (5) is minimised because annealing devices are always trying to find the lowest energy state.

This QUBO equation describes the problem as an energy landscape; imagine a 3D surface with hills and valleys. The specific shape of this landscape – the position and depth of its valleys - is determined by translating the formal objectives and constraints into the coefficients (a_i and $b_{i,j}$) of the QUBO. This translation generally conforms to the following:

- The linear terms ($a_i x_i$) represent the problem's primary objective. For instance, in our tiling problem, the goal to place tiles near the eastern wall maps directly onto these terms, creating a general slope towards the optimal region.
- The quadratic terms ($b_{i,j} x_i x_j$) are for enforcing constraints. A large "penalty" value is added to the energy for any invalid solution, such as overlapping tiles, effectively creating high-energy hills that a valid solution must avoid.

If formulated correctly, the combination of binary variables that finds the lowest possible energy (the deepest valley in this landscape) encodes the optimal and valid solution to the original problem. The final Hamiltonian (H_0 in Equation (2)) is the physical implementation of this optimisation problem's energy landscape on the quantum annealer.

The formulation of a problem as a QUBO is needed to programme the problem onto a D-wave device. The D-Wave QPU is a lattice of interconnected qubits. These qubits are connected via couplers – although the qubits are not fully connected. To programme a D-Wave quantum computer is to set values for its qubit biases and coupler strengths. The coefficients for the linear terms in Equation (5) set the values for the bias and the quadratic terms set the values for the coupler strengths (D-Wave n.d.a).

3.3. Step 3: implementation

Step 3 closes the gap between theoretical problem formulation for Grover's algorithm and QA and the physical steps that will need to be taken by engineering designers wishing to make use of these techniques. This subsection covers the implementation of both approaches with specific reference to the SDKs used – Qiskit and Ocean. It also discusses how a classical brute force algorithm was constructed to provide comparative results. The repository that stores the code used for this work can be found at the following address: https://github.com/OliverSchiffmann/comparingGateAndAnnealing.

Once the problem's construction for each approach has been validated using the simulators, it can be submitted to the respective dedicated devices. This step would provide a reader following this method with results that indicate whether the approach shows promise for near-term use. Should the results returned be usable (meaning they can be used to determine valid solutions that meet the constraints) then further investigation into how the quantum methods perform at different problem scales can be conducted. In combination with a scaling investigation, one could begin to optimise their approach for the chosen hardware. For gate-model approaches this could involve finding ways to reduce the circuit depth, or use gates with lower error rates. However, this optimisation is outside the scope of this study.

3.3.1. Grover's algorithm using Qiskit

The algorithm was implemented using the Python SDK Qiskit (IBM n.d.c). Qiskit is an open-source toolkit for developing and compiling quantum circuits. These circuits can then be sent as jobs to be executed using the compute resources supported by the IBM Quantum Platform (IBM n.d.a). Other SDKs exist (cirq, Quantum Development Kit (QDK), TKET…) and there are other suppliers of quantum computing resources (Google, Microsoft, IonQ…) (Ullah *et al.* 2022). The approach adopted for this work was chosen due to the author's prior experience with python, Qiskit and IBM's quantum experience.

Before the construction of any quantum circuit, the representation of the problem variables must be decided. For the tiling problem shown in Figure 5 the variables are the x and y coordinates of each tile. Since quantum gates in IBM's devices operate on the computational basis $|0\rangle$ and $|1\rangle$, which collapse to the binary digits 0 and 1, respectively, we can represent the problem variables in terms of binary digits. Since our placement grid begins as 8 cells long in each dimension we can represent each coordinate of a tile with 3 binary bits ($2^3 = 8$). With 3 digits for each coordinate, and two coordinates per tile, and two tiles total we need a problem register of 12 qubits. Therefore, a solution to the tiling problem will take the form

$$\underbrace{q_1, q_2, q_3,}_{x_1} \underbrace{q_4, q_5, q_6,}_{y_1} \underbrace{q_7, q_8, q_9,}_{x_2} \underbrace{q_{10}, q_{11}, q_{12}}_{y_2} \tag{6}$$

where q_n is the measured value from the nth qubit in the problem qubit register, capable of taking the value 0 or 1.

Now that the representation of variables using qubits has been decided, the circuit that will implement Grover's algorithm can be constructed. This circuit construction was taken from the tutorials provided by Qiskit (Tapia 2024) and is discussed in Gopsill *et al.* (2022).

Once the quantum circuit has been created it can be sent to a device for execution. IBM offers two options – the simulator and the dedicated device. For this work, the IBMQ_QASM_Simulator was chosen to obtain simulated (classically achieved) results and ensure that the quantum circuit was correct. Its IBMs general purpose simulator and it has access to 32 simulated qubits. For real quantum achieved results, the IBMQ_Brisbane device was used. The device has access to 127 qubits and was selected as it was, at the time, the only device large enough to run Grover's algorithm available for free on the IBM Quantum Platform. This is because although only 12 qubits are required to represent the solutions to the tiling problem 6 additional computational qubits and one Oracle qubit are required bringing the total to 19. At the time of writing, two additional devices with 127 qubits have become available – IBMQ_Osaka and IBMQ_Kyoto. These devices differ in their values for Error Per Layered Gate (EPLG) IBM (2023a).

These seven additional qubits form the Oracle workspace shown in Figure A1. These were required for some of the quantum gates that were used in the iterations of Grover's algorithm, G. This can be more easily understood using the classical example or RAM, or short term memory. In a classical computer you need memory for the bits that represent your problem variables, and you must also have enough memory to perform operations (or logic gates to continue the earlier examples) on these problem variables. In a quantum computer this oracle workspace must use quantum bits, not classical bits, as otherwise to use information from the qubits representing the problem variables you would need to measure them, which would destroy any superposition/entanglement between qubits. These 19 qubits are enough to complete the execution of Grover's algorithm on an 8x8 grid. If the problem scale increases more will be needed as more qubits will be required to encode the position of the tile in the grid, since coordinate values above 8 will require more than 3 bits to encode (4 bits up to 16, 5 up to 32 and 6 up to 64).

Devices are often referred to as samplers in the SDK documentation. A sampler is something that runs a solver multiple times and returns the distribution of results. Hence, the number of runs for the entire quantum circuit, not just G, should be chosen. 1000 runs was chosen as this kept the problem below the maximum allowable estimated run time, kept classical simulations of reasonable length, and provided a sufficient number of results to visualise the distribution – as there are only 56 valid solutions for the 8x8 problem scale.

3.3.2. *Quantum annealing using ocean*

The Ocean SDK in Python was used to construct the problem for D-wave devices (D-Wave n.d.a). To submit a problem to a dedicated D-wave device, you must formulate a BQM as either a QUBO or Ising model. The following is a coverage of the key components for creating a BQM using the QUBO approach and Ocean SDK:

1. The implementation begins with the creation of an empty BQM as a binary model: "bqm = BinaryQuadraticModel('BINARY')." This sets up a model where variables (q_{1-12} in Equation (6)) will be binary (0 or 1).
2. Variables must then be added into the empty BQM. This is initially done with 0 as the linear coefficient (a_i in Equation (5)): "bqm.add_variable(qi, 0)." Where "qi" is q_{1-12} from Equation (6).
3. The east wall placement constraint was then added. This was included as an objective equation by modifying the bias for certain variables (those that determine the x coordinate of both tiles). "bqm.add_variable(qi, −east_wall_weight)" was used to modify the bias of the x coordinate variables to a value equal to the negative of the variable "east_wall_weight." The negative is used to help promote (decrease the energy of) solutions that are on the east wall. Choosing this value is a matter of balancing the relative importance of each constraint. A higher value will prioritise solutions which meet this constraint over the other. A value of 3 was found to work well through simulator and dedicated device testing.
4. The no-overlap constraint was incorporated indirectly by defining quadratic interactions between variables. For instance, "bqm.add_interaction(qi, qj, penalty)" adds a quadratic term to the model. The value of "penalty" determines the weighting of this no-overlap constraint. The major challenge with representing this problem as a BQM was implementing the no-overlap constraint. This is because it requires the consideration of more than two variables at once (for example the tiles can share their two digits that make up their y coordinate so long as they do not share the third). Remember that due to manufacturing

constraints problems must be formulated as QUBOs instead of HUBOs. To overcome this, an ancillary variable can be used. An ancillary variable z with a large negative bias to encourage $z = 1$ is used in conjunction with other variables to model these more complex constraints.
5. The BQM can then be solved using D-wave's resources. For this work both a simulated and dedicated approach was utilised. As with the Grover's approach, this required specifying the number of runs. 1000 was chosen for this approach as well. Ocean supports the "SimulatedAnnealingSampler()" for simulating results locally, and "EmbeddingComposite(DwaveSampler())" was used for obtaining quantum achieved results.

An important note is the concept of embedding. The lattice of qubits is not a fully connected lattice and hence the BQM must undergo a process of embedding where qubits are grouped using different topologies so that the interactions between them can be represented. "EmbeddingComposite()" is the default option offered by D-wave and was sufficient for a problem of the complexity considered here.

3.3.3. The classical brute force approach

A classical brute force approach was also created for comparison. The brute-force approach simply iterated through every solution and checking if it meets the two constraints. This way it can be certain the classical approach would have found every valid solution.

Functionally, the script was a for loop that checks each combination of variable values to see if they meet the constraints. First, it checks if both tiles are in the same position. If they are, it appends the combinations of variables to the list of invalid solutions. If they are not, then it checks to see if both tiles are on the eastern wall. If they are, then it appends the variable combination to the list of valid solutions. If they are not, on the eastern wall then, again, it appends the variable combination to the list of invalid solutions.

3.4. Steps 4 and 5: evaluation using dedicated devices

These steps describe a methodology developed and tested during this study for the purpose of comparing approaches in terms of solution time and quality. This comparison protocol should be approach agnostic and therefore applicable to other QC approaches to this tiling problem. It provides results for comparing the performance of approaches; however, comparisons can still be made about problem formulations and implementation challenges by completing the previous steps in this experimental setup.

The performance of QC approaches can be determined by looking at three factors: time-to-solution, error levels in the results and the closeness of the valid results to a uniform distribution. Time to solution is important as a major barrier faced by classical computation is a limitation on the number or in the complexity of solutions that can be explored in a given time. The time to solution will be made up of two distinct components: processing time on the QPU or CPU (for quantum or classical approaches, respectively) and queue time. These can then be summed to find the total time for job completion.

Timing results for a gate-based approach were taken from the job manager on the IBM quantum platform, where Qiskit runtime usage was taken as the QPU time

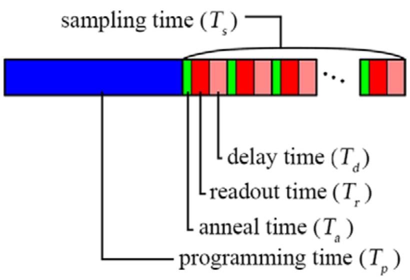


Figure 6. A figure showing the contributors to total time to solution for QA. Taken from the system documentation provided by D-wave (D-Wave n.d.a).

and CPU time is worked out by subtracting queue time from total time. Simulated QA times can be obtained by timing script execution. Real QA timings can be obtained by extracting the timings returned as part of the results. QPU time is taken as QPU_access_time. An explanation of the annealing time components can be seen in Figure 6 which is taken from the operation and timing section of D-Wave (n.d.a). The CPU times for the locally run classical approaches were achieved using an M1 pro chip (approximate clock speed of 3.2GHz) and measured using the timeit module.

Jobs used to investigate time-to-solution, error and uniformity were repeated 50 times, and mean values reported. This is to reduce the impact of the inherent variability in quantum achieved results on any observations made. The classical approach will also be repeated 50 times and averaged to find a mean time.

Error was defined as the percentage of solutions returned that do not meet the constraints. The number of solutions violating each constraint can be divided by the total number of returned solutions. This gave an error level for each constraint, which was then be combined to obtain the total error. Error levels provide an insight into the readiness of the chosen hardware for the scale and type of problem considered in this paper. High levels of error would indicate that the delicate quantum state used to represent information is being interfered with too much.

Finally, the distribution of valid results (those in which both tiles are placed on the eastern wall) will be compared to a uniform distribution. This is done using the coefficient of variation. This is calculated by dividing the standard deviation of the frequencies for valid positions by the mean. A lower value indicates that the results are closer to uniformly distributed. Understanding how close the results returned by a quantum computer are to a uniform distribution is useful for several reasons. In quantum computing, jobs submitted to these devices often undergo multiple executions due to inherent noise affecting the hardware and the probabilistic nature of results. Examining the closeness to a uniform distribution enables us to assess whether there are any systematic biases present within the quantum computer. By analysing this closeness, we gain insights into the reliability and accuracy of the quantum device for specific problem scales. Additionally, it helps us determine the number of times we may need to sample the results to effectively mitigate these biases. Note that coefficient of variation results for the classical approach will not be collected. The classical brute force algorithm iterates through every possible solution and checks if it meets the constraints. It does not return results, as the quantum samplers do, and hence has no coefficient of variation.

4. Results

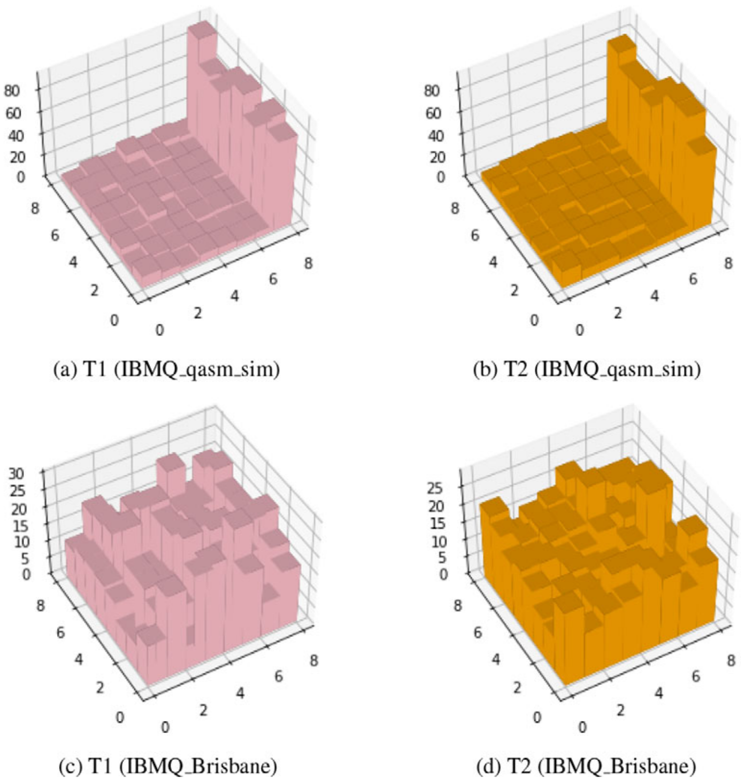
This section presents the results obtained through the use of simulated and real quantum devices for both the Grover's algorithm and the QA approach to solving the tiling problem. These are the results of executing steps 3, 4 and 5 in Figure 4.

4.1. Simulated and feasibility results

Results were collected in two stages, preliminary results for assessing the feasibility of each approach and more detailed results concerning performance at different problem scales. Preliminary results for both approaches are presented.

The results obtained from simulators and for testing the feasibility of each approach consist of 3-D histograms. These show the frequency with which each tile is placed in each position for the simulated and quantum computations of both approaches (Figures 7 and 8). In these Figures "T1" and "T2" refer to results showing the placement of Tile 1 and Tile 2, respectively.

These results consider only 8x8 grids and are used to confirm if further investigation (increasing problem scale) was warranted. Grover's algorithm does



(a) T1 (IBMQ_qasm_sim) (b) T2 (IBMQ_qasm_sim)

(c) T1 (IBMQ_Brisbane) (d) T2 (IBMQ_Brisbane)

Figure 7. Figures showing the results from Grover's Algorithm using both a quantum simulator (IBMQ_qasm_sim) and a real quantum computer (IBMQ_Brisbane). The frequency of tile placement at each position on the grid is indicated by the height of the bar at that coordinate. Figures (a) and (c) show the results for the placement of Tile 1 using simulated and real quantum computation, respectively. (b) and (d) show the same for Tile 2, respectively.

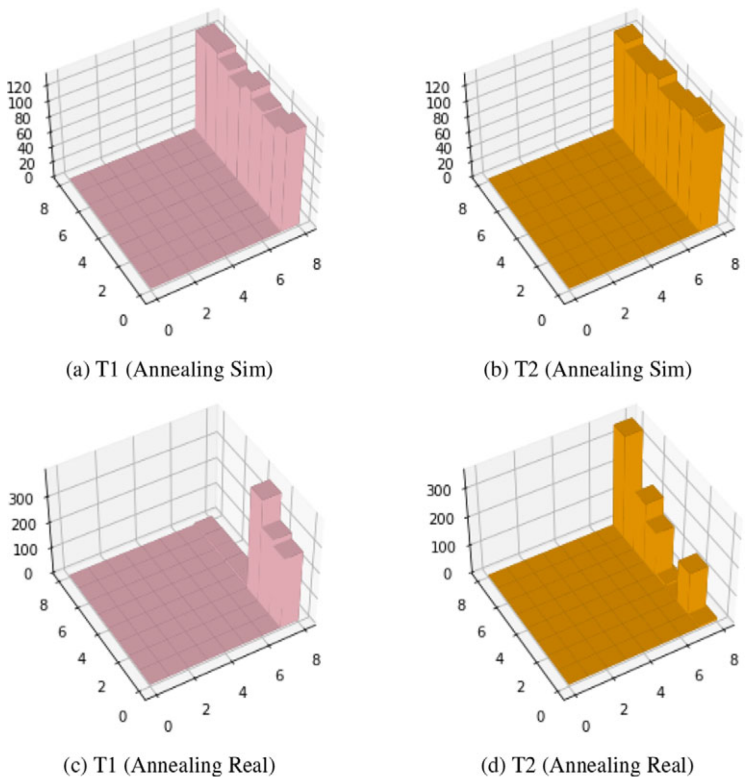


Figure 8. Figures showing the results of both simulated and real QA approaches to the tiling problem. The frequency of tile placement at each position on the grid is indicated by the height of the bar at that coordinate.

not produce any useable quantum achieved results when executed using IBM's device. This was expected as it is not an approach that has been designed for NISQ application. However, the results are presented and discussed to complete the loop for application of this study's methodology. QA did produce useable results when executed on a D-wave QPU. However, the tile positions were subject to a strong bias – a point discussed further in the Section 5.1.

4.1.1. Error

From these histograms, the error present in the returned results could be calculated as described in the Experimental Setup Section. The error levels from these preliminary results are presented in Table 2.

The combination of the 3-D histograms for Grover's approach, Figure 7, and the error levels, Table 2, shows that a simulated approach obtains good results. It is clear that the approach is more likely to return a result that meets the constraints and that there is a roughly uniform distribution of results across the 8 valid tile positions. This provides assurance that the quantum circuit for Grover's algorithm has been constructed appropriately for our problem. Although not benefiting from the potential advantages of QC, the opportunity to model problems using QC principles with the view to one day using a real-world device is beneficial. However,

Table 2. A table containing the percentage error present in both the simulated and real quantum results obtained for Grover's and QA approaches, broken up as they apply to each constraint

	No overlap (%)	Eastern wall (%)	Total (%)
Grover's simulated	1.5	42.0	43.5
Grover's quantum	1.6	97.8	99.4
QA simulated	0.0	0.0	0.0
QA real	0.0	0.0	0.0

when looking at the results obtained from one of IBM's dedicated quantum devices, it is clear that the results become dominated by error.

The 3-D histograms in Figure 8 and error levels in Table 2 for the annealing approach show improved performance over the Grover's approach. The simulated results show that the problem has been formulated correctly. Further, the results achieved using a dedicated D-wave device suffer from 0% error, as opposed to the 99.4% error when using IBM devices. This is promising as it implies that D-wave devices can be used to obtain useful results.

4.1.2. Time to solution

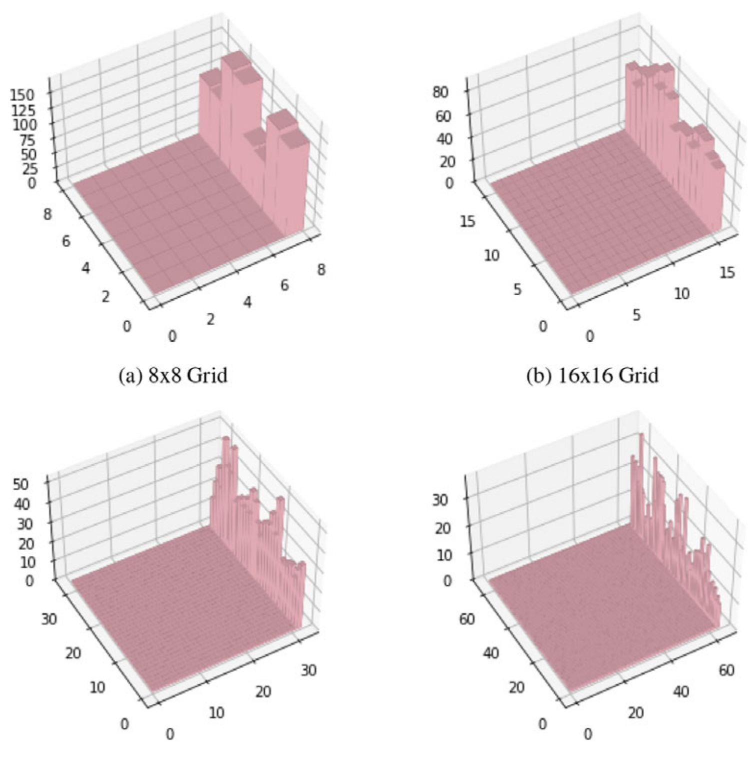
Timing results for the preliminary results were also collected as described in the Experimental Setup Section. Table 3 contains time-to-solution values for the preliminary results. The classical brute force approach is 2 orders of magnitude faster than QA and seven orders faster than the quantum implemented Grover's approach. A quantum speed-up has not been demonstrated; however, this relatively small problem may not fully leverage the promised scaling benefits of QC. Comparing QA and Grover's approach in Table 3, QA proves faster, in both queue time and QPU time. Note that these timings are for single jobs and were not averaged over multiple jobs.

4.2. QA results at varying problem scales

The preliminary results show that only QA has produced feasible results for the chosen problem. It was also of interest to examine if this would change at increased

Table 3. A table containing the times required for obtaining the results presented in Figures 7 and 8. Note that a queue time of N/A is given for locally run approaches

	QPU/CPU time	Queue time	Total time
Grover's simulated	≈ 5 m 45 s	less than 1 s	5 m 45 s
Grover's quantum	1 m 27 s	2 h 17 m 17.5 s	3 h 0 m 25.3 s
QA simulated	0.243 s	N/A	0.243 s
QA real	0.126 s	0.136	0.262 s
Brute force	0.00367 s	N/A	0.00367 s



(a) 8x8 Grid

(b) 16x16 Grid

(c) 32x32 Grid

(d) 64x64 Grid

Figure 9. Figures showing the results for Tile 1 obtained using D-wave's Advantage_system4.1 at varying grid sizes.

problem scales. Figures 9 and 10 show the results returned from a single job for each grid size. Taking these figures in conjunction with the data presented in Table 4 reveals how QA performance changes at increased problem scales. The solutions remain usable at all scales, meaning that almost all solutions returned are valid. However, the coefficient of variation does increase with grid size. This is to be expected, as QA is a sampling method, and so when there are more valid solutions, the results on the eastern wall are less uniformly distributed. The QPU time and percentage error exhibit small increases with problem scale. Since annealing time is specified as a problem parameter, there would be no expected increase in QPU access time aside from an increased time to programme the problem onto the annealing device. The most significant observation from these results is that QA's usability remains relatively constant at increased scales, but the classical brute force approach suffers significantly. Between a 16x16 and 32x32 grid, QA overtakes the brute force approach in time to solution.

5. Discussion

This work set out to investigate two different QC approaches for tackling a generalised configuration-design problem. The aim of this investigation was to develop and test a methodology for assessing the suitability of each approach for engineering designers. The collected results provide insight into the quality of

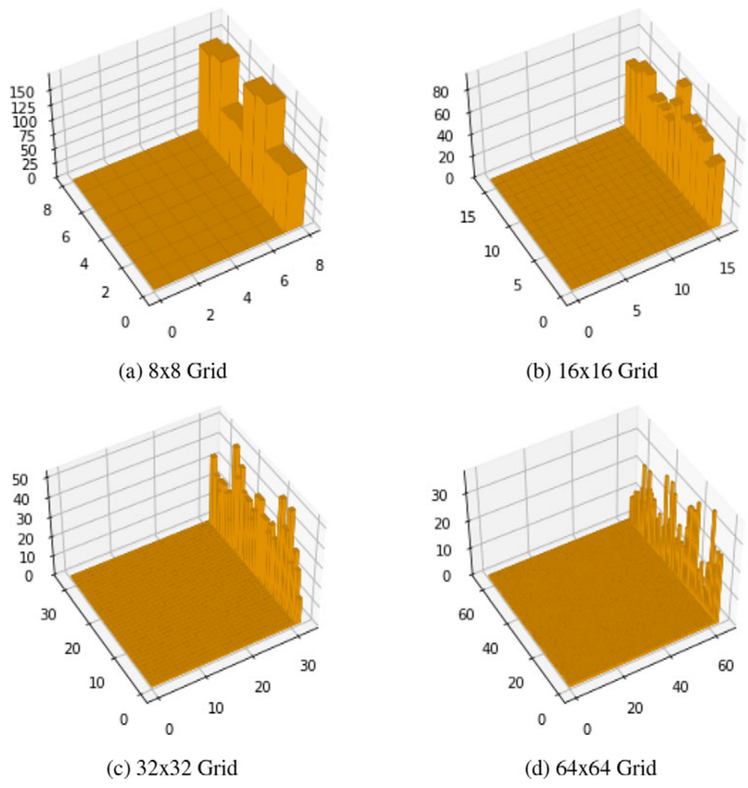


Figure 10. Figures showing the results for Tile 2 obtained using D-wave's Advantage_system4.1 at varying grid sizes.

Table 4. A table containing data comparing the time to solution for the classical brute force approach (CPU time) and the QA approach, as well as error and coefficient of variation values for QA, at different problem scales. Note that a lower value for coefficient of variation indicates less deviation from a uniform distribution. The CPU times for the locally run classical approaches were achieved using an M1 pro chip (approximate clock speed of 3.2GHz)

Grid size	CPU time/s	QPU time/s	Error / %	Coefficient of variation
8x8	0.00292	0.125	0.01	0.248
16x16	0.04512	0.131	0.006	0.291
32x32	0.72754	0.133	0.002	0.370
64x64	14.7632	0.137	0.002	0.463

results through the percentage error, distribution of valid solutions, and the time to solution of the respective approaches. As a result of executing the methodology, observations about the ease of implementation for the respective approaches can also be made. Development of two reference models for QC application also enables comparisons and conclusions focused around problem formulation, whilst expanding the available library of such models.

5.1. Issues with quantum achieved results

The properties of the IBM_Brisbane device can be checked using IBM's Quantum Platform. At the time of writing its circuit layer operations per seconds (CLOPS) is listed as 180,000 (IBM n.d.a). The depth (number of operations/gates applied in sequence) of the Grover's circuit with 9 Oracle repetitions is 289. This means it would take $\approx 1606\mu s$ to complete the circuit. The same device has a median coherence time (called T1 by IBM) of $230.58\mu s$. This indicates that the qubits inside the IBM devices may well be decohering (having their quantum state disrupted by environmental factors) before useful results can be obtained. The coherence time is ≈ 7 times too short. This could explain the error dominated results shown in Figure 7. Furthermore, NISQ devices suffer from other forms of error, such as gate errors caused by poor calibration (Yang *et al.* 2023). This means it is not just the coherence time that must increase but the rate of error must also fall. The depth of the circuit could be reduced by decreasing the number of applied oracles, at the expense of reducing the probability of returning a correct solution. This negative influence may be offset by increasing the number of runs. A small experiment was run that investigated a single oracle version for 8000 runs. However, this was found to be just as dominated by error and so the Grover's approach was not investigated further as part of Step 5 in Figure 4. It is important to note that these values are updated regularly so developing a knowledge of the hardware milestones needed for an algorithm to be executable is valuable.

The histograms in Figure 8 show an inherent bias that is known to affect D-wave's devices (Hauke *et al.* 2020). This means that whilst the annealing approach does return valid results, its solution space exploration capacity might be limited. Furthermore, real-world problems are likely to contain many more constraints and objectives which will complicate the energy landscape. As such, increased error would be expected to appear in results obtained via annealing.

5.2. Speed of QA versus brute force

It is important to note that there are classical algorithms superior to the brute force approach used here. As such, the data in Table 4 are not a demonstration of quantum speed-up. However, these QA annealing results were obtained without using the problem specific knowledge that would be needed to choose and make use of the appropriate classical algorithm. Instead, knowledge of the QA procedure and how to formulate the problem as a BQM was needed.

5.3. Ease of implementation between hardware options

The ease of implementation for the hardware used in each approach can be discussed subjectively as a result of completing this methodology. The classical analogy of logic gates for the gate-based Grover's approach could be argued to aid understanding of the algorithm operation. However, the annealing approach does not require the construction of a circuit. Instead it requires the selection of coefficients in Equation (5). Whilst this is a more abstract approach, it is an approach that remains constant, even for other problem formulations. Further, the energy landscape considered throughout the annealing approach is analogous to the solution space engineering designers are used to exploring. The documentation supporting Qiskit as well as the job manager provided by IBM appeared more developed than D-wave's counterparts. In the authors' opinion, the background theory required for the gate-based approach was more demanding, whilst the problem formulation for annealing was more complex. Considering all these points, the authors would consider the barrier to implementation lower for annealing methods. The formulation of problems as QUBO/Ising models is also studied in other fields like physics, especially the formulation of graph problems (Lucas 2014; Arai & Haraguchi 2021. This research is partly driven by excitement around QC. It may therefore be possible to reduce the burden of problem formulation by building on existing QUBO/Ising formulations published in literature.

5.4. Observations about problem formulation for engineering designers

Observations drawn from this work regarding gate-based versus QA methods for engineering design applications suggest that reducing circuit depth in gate-based approaches may enhance usability. D-wave's QA approach can yield rapid and usable results compared to Grover's method, but users should be cautious of inherent biases. In more complex cases, as with increasing scale and a more intricate energy landscape, error in QA results may increase due to a reduced energy gap. Bias drawbacks may intensify if valid solutions represent a smaller part of the solution space. However, the scaling benefits of QC reduce the time disparity between QA and classical approaches. Notably, most of the QPU time for QA was spent on problem programming, suggesting potential for increases in annealing time and more runs without significantly affecting time to solution. The number of runs should be increased as problem scale increases to maintain even coverage of equally optimal solutions.

Perhaps the most significant observation made during this work concerns the formulation of problem QUBOs for D-wave's annealing devices.

The dedicated QA devices at D-wave accept BQM problems in the QUBO or Ising format. A QUBO formulation was chosen for this work as it appeared more intuitive. To formulate a problem as a QUBO, an equation with all the binary variables needed to represent the problem must be created. The coefficients for those variables and their quadratic combinations in the QUBO equation must also be selected. This should be done such that the equation would reach a minimum value when the variables take values that represent the optimal solution. However, when dealing with many constraints and objectives it is not obvious what the values of these coefficients should be, particularly when considering the weighting of each constraint. Fortunately, each constraint/objective can be considered individually, and their respective QUBOs summed to form the overall BQM (D-Wave 2022). However, when constraints start to concern more than two variables, ancilla variables are required so that no terms become of a higher order than quadratic.

The example seen in D-Wave (2022) is for a small two variable problem. It uses a constraint satisfaction table to generate a system of linear equations which can then be solved for exact QUBO coefficient values representing the constraint. If this method was followed for the problem discussed in this paper, then the number of equations exceeded the number of variables and the system became over-constrained. Instead, the bqm.add_variable() method was used. This method sets the bias value for a variable, meaning it determines the degree to which a variable tends to a particular outcome (D-Wave n.d.b). Essentially, it rewards when the variable takes the value 1 and penalises when it takes the value 0 when the bias is negative (or vise versa if the bias is positive). This method when combined with the binary string problem formulation shown in Equation (6) makes it difficult to create certain constraints and objectives. For example, imagine that the objective was to place a tile as close to the centre of the grid as possible. The number of 1's and 0's in the binary string dictating the solution is roughly equal. How would you penalise a solution closer to the eastern wall? Penalising more 1's would result in the optimum solution being placed on the western wall. Rewarding more 1's would move the optimum towards the eastern wall. Additionally, this binary string formulation makes it hard to create smoothly varying constraints. This is best seen in Figure 11. This figure is a visual representation of the energy landscape for Tile 1's position, where a higher energy represents a more probable returned solution. It can be seen that solutions on the eastern wall are best, and hence they are returned most often (seen in the results as part of Figures 8 and 9). However, it can also be seen (most clearly at x = 5) that not every solution performs better than its westerly neighbour. This is due to the way that counting in binary works, and that as the integer value the string represents increases, the number of 1's within the string does not always increase (011 ⟶ 100).

The impact of this is that engineering designers must consider the types of constraints and objectives they will be creating during the formulation of the problem. Whilst the number of variables (qubits) required for solution representation scales well with problem size for this approach, it is less flexible when creating constraints and objectives. An alternative problem formulation may be better. An example could be one variable per tile per space for each axis. So for an 8x8 grid there would be 8 variables for the x-coordinate of Tile 1, 8 for the y-coordinate of Tile 1, and 16 more for Tile 2. The position of a tile could then be indicated by a single variable taking the value 1 for each coordinate. This would scale less well with the problem space and would come with the added constraint that not all value combinations for the variables constitute a possible solution (when more than one variable is 1, for example). The scaling problem is less of a

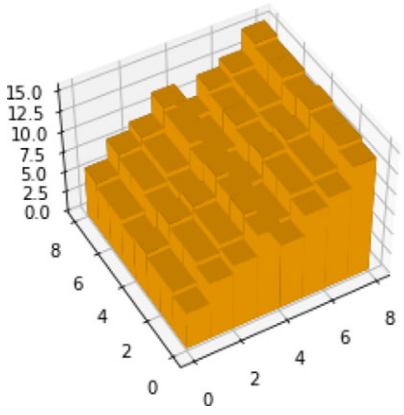


Figure 11. A figure showing how the difficulty in creating a smoothly varying energy landscape using the binary string solution representation. Note that for visual clarity the sign of all energies have been changed to make them positive.

concern since D-wave devices have many more qubits than were used in this study. However, it is important to remember that not all qubits can be used for variables, as some are needed to connect qubits within the device.

5.5. Methodology reflection

An objective of this study was to address a gap in the existing body of research, as highlighted in the Related Work section. Specifically, this gap concerns the evaluation of different hardware options for use by engineering designers.

A key aspect of this investigation was assessing the quality of results achievable with quantum hardware. However, this analysis is inherently limited, as Grover's algorithm was unable to produce usable results on any available device for the given problem formulation. To better evaluate the potential of gate-based approaches for engineering design problems, a more suitable method could have been selected—one that aligns more closely with the capabilities of today's or near-term quantum devices.

In answer to the question "Is the method applied in this study suitable for identifying promising QC approaches for engineers?," it may be improved by modifying Steps 1 and 2 shown in Figure 4. The study initially approached the problem by first identifying a broadly relevant engineering challenge that might benefit from a QC approach, followed by a search for two suitable quantum algorithms. This strategy ensures that, if a successful quantum approach is identified, the findings on problem implementation would be valuable to a wider audience. However, a drawback of this method is that the theoretical advantages of QC may be hampered by current hardware limitations.

An alternative approach that could have improved the experimental setup would be to first identify promising near-term quantum techniques before reviewing existing computational engineering design tasks to determine which align with known applications of those techniques. While this approach might result in selecting an application area that stands to benefit less from QC in the long term, it would increase the likelihood of identifying a realistic and feasible quantum alternative in the near future.

5.6. Future work

The first area identified for further investigation would be to explore a different, more NISQ-suitable (e.g., shallower) gate-model approach. This would enable a more relevant comparison between the most promising gate-based option and QA from a performance perspective. Such a comparison would help direct engineering designers' learning as they begin to adopt this new technology. The author's highlight the QAOA as a promising method (Blekos *et al.* 2024). Combining a QAOA investigation with the research methodology restructuring suggested in the previous subsection should allow strong conclusions to be drawn about suitable avenues for QC to be applied in engineering design. Furthermore, QAOA can solve problems using the same QUBO formulation as QA. This would allow the comparison of performance to be problem formulation agnostic, giving clearer insights into the suitability of different hardware options.

This QAOA investigation could be supplemented by exploring various types of engineering design problem currently addressed with classical computation, providing insights into potential areas for QC. This could involve a review of the literature looking for problems or sub-problems for which there is a direct quantum alternative. This should help to develop approaches that avoid the limitations discussed in this study.

Another avenue for exploration is other gate-based QC hardware options. IBM uses superconducting qubits, but other options with longer coherence times are available. This could involve exploring what options are offered by other hardware developing companies such as Microsoft that use nuclear magnetic resonance QC or IonQ that use trapped ion QC (Hassija *et al.* 2020; Cheng *et al.* 2023) as these are said to have long coherence times.

6. Conclusion

QA facilitated through D-Wave stands out as an appropriate quantum computing platform for near-term use. The robustness of simulators is crucial in building experience that will enable quantum computing principles for engineering problem solving to be leveraged. These can also be used to help decide if a design problem is a good candidate for a quantum approach by enabling experimentation with problem formulation, different algorithms, and even hardware options at zero financial cost. This is particularly useful given the rapid development of QC hardware and its constantly evolving capabilities.

An important future direction involves determining the optimal size of design problems for quantum computing applications, considering scalability and computational limits. It is difficult to say the exact problem scale where QC will outperform classical methods once you implement your approach onto real hardware. This issue is exemplified in recent work investigating D-wave's hybrid solvers published in Nature Quinton *et al.* (2025). They showed that performance varies heavily between binary quadratic problems and non-binary problems, non-quadratic problems, or combinations of the latter two. It is therefore unlikely we will be able to identify universal values for problem scales where QC achieves advantage. This is why it is essential that engineering designers begin to develop, deploy, and test QC approaches to their own problems so that these critical tipping points can start being identified. This requirement provides justification for this study, as engineering designers will need an introduction to the field of QC, and we will need the development of robust methods for comparison across problems and solving methods.

This work demonstrates the relatively straightforward representation of a generalised instance of a configuration-based design problem for both QA and gate-based approaches, with QA already yielding usable results. Whilst the Grover's approach investigated was not suitable for NISQ implementation, the surrounding discussion covering problem formulation could remain useful for engineering designers in an FTQC future, as it is independent from gate-model hardware variations. It has also been shown that engineering designers should pay particular attention to the type of constraints and objectives that they will be using when deciding their problem formulation. Expanding research in quantum computing offers hope for improved hardware, including longer coherence times, reduced bias in annealing devices, increased CLOPS and more qubits. As a result, this work provides a foundation for engineering designers to consider how they can adapt their problems to leverage potential quantum speed-up in the future.

A summary of this studies findings are as follows:

1. The reference models presented accurately translate the layout problem into a form appropriate for their respective quantum solving methods.
2. IBM devices are currently incapable of executing this Grover's algorithm approach for the 8x8 grid problem formulation,
3. D-wave's device was able to achieve correct results with almost no error for every problem scale tested; however, the results were skewed by hardware biases.
4. D-wave device was able to achieve its results for the 64x64 scale problem approximately 100 times faster than the brute force approach.
5. It is non-trivial to transition from classical to quantum methods and a degree of training is required; however, the major difficulty with annealing can be alleviated by making use of problem formulations already published in literature.
6. When formulating a problem for tackling with QA, engineering designers must pay careful attention to the type of constraints and objectives present in their problem. Those with a greater than quadratic order will introduce additional qubit overhead.
7. Our approach for identifying suitable application areas for QC in engineering design could be improved by comparing a NISQ-suitable gate-based method across multiple problem types instead of trying to find a suitable method/algorithm for a problem of wide interest.

Acknowledgements

Funding for this work was provided by the EPSRC, grant: EP/W524414/1.

References

AbuGhanem, M. 2025 Ibm quantum computers: Evolution, performance, and future directions. *The Journal of Supercomputing* **81** (5), 687.

Al-Turjman, F. M., **Hassanein, H. S.** & **Ibnkahla, M. A.** 2013 Efficient deployment of wireless sensor networks targeting environment monitoring applications. *Computer Communications* **36** (2), 135–148.

Amin, M. H. 2009 Consistency of the adiabatic theorem. *Physical Review Letters* **102** (22), 220401.

Apolloni, B., **Cesa-Bianchi, N.** & **De Falco, D.** 1990 A numerical implementation of "quantum annealing". In *Stochastic Processes, Physics and Geometry: Proceedings of the Ascona-Locarno Conference*, edited by Albeverio, S., Casati, G., Merlini, D., Moresi, R. & Cattaneo, G (World Scientific Singapore, 1990), pp. 97–111.

Arai, H. & **Haraguchi, H.** 2021 A study of ising formulations for minimizing setup cost in the two-dimensional cutting stock problem. *Algorithms* **14** (6), 182.

Arute, F., **Arya, K.**, **Babbush, R.**, **Bacon, D.**, **Bardin, J. C.**, **Barends, R.**, **Biswas, R.**, **Boixo, S.**, **Brandao, F. G.**, **Buell, D. A.**, **Burkett, B.**, **Chen, Y.**, **Chen, Z.**, **Chiaro, B.**, **Collins, R.**, **Courtney, W.**, **Dunsworth, A.**, **Farhi, E.**, **Foxen, B.**, **Fowler, A.**, **Gidney, C.**, **Giustina, M.**, **Graff, R.**, **Guerin, K.**, **Habegger, S.**, **Harrigan, M. P.**, **Hartmann, M. J.**, **Ho, A.**, **Hoffmann, M.**, **Huang, T.**, **Humble, T. S.**, **Isakov, S. V.**, **Jeffrey, E.**, **Jiang, Z.**, **Kafri, D.**, **Kechedzhi, K.**, **Kelly, J.**, **Klimov, P. V.**, **Knysh, S.**, **Korotkov, A.**, **Kostritsa, F.**, **Landhuis, D.**, **Lindmark, M.**, **Lucero, E.**, **Lyakh, D.**, **Mandrà, S.**, **McClean, J. R.**,

McEwen, M., Megrant, A., Mi, X., Michielsen, K., Mohseni, M., Mutus, J., Naaman, O., Neeley, M., Neill, C., Niu, M. Y., Ostby, E., Petukhov, A., Platt, J. C., Quintana, C., Rieffel, E. G., Roushan, P., Rubin, N. C., Sank, D., Satzinger, K. J., Smelyanskiy, V., Sung, K. J., Trevithick, M. D., Vainsencher, A., Villalonga, B., White, T., Yao, Z. J., Yeh, P., Zalcman, A., Neven, H. & **Martinis, J. M.** 2019 Quantum supremacy using a programmable superconducting processor. *Nature* **574** (7779), 505–510.

Ball, P. December 2018 *Ion-based commercial quantum computer is a first.* https://physicsworld.com/a/ion-based-commercial-quantum-computer-is-a-first/ (accessed June 25th 2025).

Belhocine, A. & **Abdullah, O. I.** 2020 Thermomechanical model for the analysis of disc brake using the finite element method in frictional contact. *Multiscale Science and Engineering* **2**, 27–41.

Biskjaer, M. M., Dalsgaard, P. & **Halskov, K.** 2014 A constraint-based understanding of design spaces. In *Proceedings of the 2014 Conference on Designing Interactive Systems*, pp. 453–462. Association for Computing Machinery.

Blekos, K., Brand, D., Ceschini, A., Chou, C.-H., Li, R.-H., Pandya, K. & **Summer, A.** 2024 A review on quantum approximate optimization algorithm and its variants. *Physics Reports* **1068**, 1–66.

Callison, A. & **Chancellor, N.** 2022 Hybrid quantum-classical algorithms in the noisy intermediate-scale quantum era and beyond. *Physical Review A* **106** (1), 010101.

Cerezo, M., Arrasmith, A., Babbush, R., Benjamin, S. C., Endo, S., Fujii, K., McClean, J. R., Mitarai, K., Yuan, X., Cincio, L. & **Coles, P. J.** 2021 Variational quantum algorithms. *Nature Reviews Physics* **3** (9), 625–644.

Chen, X., Zhao, X., Gong, Z., Zhang, J., Zhou, W., Chen, X. & **Yao, W.** 2021 A deep neural network surrogate modeling benchmark for temperature field prediction of heat source layout. *Science China Physics, Mechanics & Astronomy* **64** (11), 1.

Cheng, B., Deng, X.-H., Gu, X., He, Y., Hu, G., Huang, P., Li, J., Lin, B.-C., Lu, D., Lu, Y., Qiu, C., Wang, H., Xin, T., Yu, S., Yung, M.-H., Zeng, J., Zhang, S., Zhong, Y., Peng, X., Nori, F. & **Yu, D.** 2023 Noisy intermediate-scale quantum computers. *Frontiers of Physics* **18** (2), 21308.

Choi, C. September 2020 *First photonic quantum computer on the cloud toronto-based xanadu suggests its system could scale up to millions of qubits.* https://spectrum.ieee.org/photonic-quantum (accessed June 25th 2025).

Chuang, I. L., Gershenfeld, N. & **Kubinec, M.** 1998 Experimental implementation of fast quantum searching. *Physical Review Letters* **80** (15), 3408.

Correa-Jullian, C., Cofre-Martel, S., Martin, G. S., Droguett, E. L., de Novaes Pires Leite, G. & **Costa, A.** 2022 Exploring quantum machine learning and feature reduction techniques for wind turbine pitch fault detection. *Energies* **15** (8), 2792.

D.-W. Developers 2022 *Measuring performance of the leap constrained quadratic model solver* Technical report, 14-1065A-A. D-Wave Systems Inc.

Dalal, A., Montalban, I., Hegade, N. N., Cadavid, A. G., Solano, E., Awasthi, A., Vodola, D., Jones, C., Weiss, H. & **Füchsel, G.** 2024 Digitized counterdiabatic quantum algorithms for logistics scheduling. *Physical Review Applied* **22** (6), 064068.

Dasgupta, S. & **Humble, T. S.** 2020 Characterizing the stability of NISQ devices. In *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pp. 419–429. IEEE.

Devitt, S. J. 2016 Performing quantum computing experiments in the cloud. *Physical Review A* **94** (3), 032329.

Dimitrova, Z. & **Maréchal, F.** 2015 Techno-economic design of hybrid electric vehicles using multi objective optimization techniques. *Energy* **91**, 630–644.

Dumke, R., **Volk, M.**, **Müther, T.**, **Buchkremer, F.**, **Birkl, G.** & **Ertmer, W.** 2002 Micro-optical realization of arrays of selectively addressable dipole traps: A scalable configuration for quantum computation with atomic qubits. *Physical Review Letters* **89** (9), 097903.

D-Wave October 2018 *D-wave launches leap, the first real-time quantum application environment.* https://www.dwavequantum.com/company/newsroom/press-release/d-wave-launches-leap-the-first-real-time-quantum-application-environment/ (accessed June 25th 2026).

D-Wave 2022 *Problem formulation guide.* https://www.dwavesys.com/media/bu0lh5ee/problem-formulation-guide-2022-01-10.pdf (accessed June 25th 2025).

D-Wave May 2025 *D-wave announces general availability of advantage2 quantum computer, its most advanced and performant system.* https://www.dwavequantum.com/company/newsroom/press-release/d-wave-announces-general-availability-of-advantage2-quantum-computer-its-most-advanced-and-performant-system/ (accessed June 25th 2025).

D-Wave n.d.a *D-wave documentation.* https://docs.dwavesys.com/docs/latest/doc_getting_started.html (accessed June 25th 2025).

D-Wave n.d.b *Beginner guides.* https://www.dwavesys.com/learn/resource-library/ (accessed June 25th 2025).

D-Wave n.d.c *Real-world quantum applications at business scale.* https://www.dwavesys.com/learn/customer-success-stories/ (accessed June 25th 2025).

Feynman, R. P. 2018 Simulating physics with computers. In *Feynman and Computation*, pp. 133–153. CRC Press.

Ghani, J. A., **Choudhury, I.** & **Hassan, H.** 2004 Application of taguchi method in the optimization of end milling parameters. *Journal of Materials Processing Technology* **145** (1), 84–92.

Gill, S. S., **Kumar, A.**, **Singh, H.**, **Singh, M.**, **Kaur, K.**, **Usman, M.** & **Buyya, R.** 2022 Quantum computing: A taxonomy, systematic review and future directions. *Software: Practice and Experience* **52** (1), 66–114.

Gonzalez-Delgado, D., **Jaen-Sola, P.** & **Oterkus, E.** 2023 Design and optimization of multi-mw offshore direct-drive wind turbine electrical generator structures using generative design techniques. *Ocean Engineering* **280**, 114417.

Google Quantum AI and Collaborators. 2024 Quantum error correction below the surface code threshold. *Nature* **638**, 920–926.

Gopsill, J., **Johns, G.** & **Hicks, B.** 2021 Quantum combinatorial design. *Proceedings of the Design Society* **1**, 2511–2520.

Gopsill, J., **Schiffmann, O.** & **Hicks, B.** 2022 Research questions in applying quantum computing to systems design. In *Design Computing and Cognition'22* (ed. Gero, J. S.), pp. 735–745. Springer International Publishing. doi:10.1007/978-3-031-20418-0_43.

Grover, L. K. 1996 A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, pp. 212–219. ACM.

Harrow, A. W., **Hassidim, A.** & **Lloyd, S.** 2009 Quantum algorithm for linear systems of equations. *Physical Review Letters* **103** (15), 150502.

Hassija, V., **Chamola, V.**, **Saxena, V.**, **Chanana, V.**, **Parashari, P.**, **Mumtaz, S.** & **Guizani, M.** 2020 Present landscape of quantum computing. *IET Quantum Communication* **1** (2), 42–48.

Hastings, M. B. 2021 The power of adiabatic quantum computation with no sign problem. *Quantum* **5**, 597.

Hauke, P., **Katzgraber, H. G.**, **Lechner, W.**, **Nishimori, H.** & **Oliver, W. D.** 2020 Perspectives of quantum annealing: Methods and implementations. *Reports on Progress in Physics* **83** (5), 054401.

Held, S., **Korte, B.**, **Rautenbach, D.** & **Vygen, J.** 2011 Combinatorial optimization in VLSI design. *Combinatorial Optimization* **31**, 33–96.

Hoole, J., **Mellor, P. H.**, **Simpson, N.** & **North, D.** 2021 Statistical simulation of conductor lay and ac losses in multi-strand stator windings. In *2021 IEEE International Electric Machines & Drives Conference (IEMDC)*, pp. 1–8. IEEE.

IBM 2023a *Updating how we measure quantum quality and speed.* https://research.ibm.com/blog/quantum-metric-layer-fidelity (accessed June 25th 2025).

IBM 2023b *IBM quantum learning.* https://learning.quantum.ibm.com/ (accessed June 25th 2025).

IBM n.d.a *Compute resources.* https://quantum.ibm.com/services/resources (accessed June 25th 2025).

IBM n.d.b *Celebrating 7 years of qiskit.* https://www.ibm.com/quantum/qiskit/history#2019 (accessed June 25th 2025).

IBM n.d.c *Qiskit homepage.* https://qiskit.org/ (accessed November 24th 2023).

Johnson, M. W., **Amin, M. H.**, **Gildert, S.**, **Lanting, T.**, **Hamze, F.**, **Dickson, N.**, **Harris, R.**, **Berkley, A. J.**, **Johansson, J.**, **Bunyk, P.**, **Chapple, E. M.**, **Enderud, C.**, **Hilton, J. P.**, **Karimi, K.**, **Ladizinsky, E.**, **Ladizinsky, N.**, **Oh, T.**, **Perminov, I.**, **Rich, C.**, **Thom, M. C.**, **Tolkacheva, E.**, **Truncik, C. J. S.**, **Uchaikin, S.**, **Wang, J.**, **Wilson, B.** & **Rose, G.** 2011 Quantum annealing with manufactured spins. *Nature* **473** (7346), 194–198.

Jones, N. 2013 Google and nasa snap up quantum computer. *Nature* **497** (7449), 16.

Kelly, J. March 2018 *A preview of bristlecone, google's new quantum processor.* https://research.google/blog/a-preview-of-bristlecone-googles-new-quantum-processor/ (accessed June 25th 2025).

Khalafallah, A. & **El-Rayes, K.** 2011 Automated multi-objective optimization system for airport site layouts. *Automation in Construction* **20** (4), 313–320.

Kim, Y., **Eddins, A.**, **Anand, S.**, **Wei, K. X.**, **Van Den Berg, E.**, **Rosenblatt, S.**, **Nayfeh, H.**, **Wu, Y.**, **Zaletel, M.**, **Temme, K.** & **Kandala, A.** 2023 Evidence for the utility of quantum computing before fault tolerance. *Nature* **618** (7965), 500–505.

King, A. D., **Nocera, A.**, **Rams, M. M.**, **Dziarmaga, J.**, **Wiersema, R.**, **Bernoudy, W.**, **Raymond, J.**, **Kaushal, N.**, **Heinsdorf, N.**, **Harris, R.**, **Boothby, K.**, **Altomare, F.**, **Asad, M.**, **Berkley, A. J.**, **Boschnak, M.**, **Chern, K.**, **Christiani, H.**, **Cibere, S.**, **Connor, J.**, **Dehn, M. H**, **Deshpande, R.**, **Ejtemaee, S.**, **Farre, P.**, **Hamer, K.**, **Hoskinson, E.**, **Huang, S.**, **Johnson, M. W.**, **Kortas, S.**, **Ladizinsky, E.**, **Lanting, T.**, **Lai, T.**, **Li, R.**, **MacDonald, A. J. R.**, **Marsden, G.**, **McGeoch, C. C.**, **Molavi, R.**, **Oh, T.**, **Neufeld, R.**, **Norouzpour, M.**, **Pasvolsky, J.**, **Poitras, P.**, **Poulin-Lamarre, G.**, **Prescott, T.**, **Reis, M.**, **Rich, C.**, **Samani, M.**, **Sheldan, B.**, **Smirnov, A.**, **Sterpka, E.**, **Clavera, B. T.**, **Tsai, N.**, **Volkmann, M.**, **Whiticar, A. M.**, **Whittaker, J. D.**, **Wilkinson, W.**, **Yao, J.**, **Yi, T. J.**, **Sandvik, A. W.**, **Alvarez, G.**, **Melko, R. G.**, **Carrasquilla, J.**, **Franz, M.** & **Amin, M. H.** 2025 Beyond-classical computation in quantum simulation. *Science* **388** (6743), eado6285.

Knight, W. November 2017 *IBM raises the bar with a 50-qubit quantum computer.* https://www.technologyreview.com/2017/11/10/147728/ibm-raises-the-bar-with-a-50-qubit-quantum-computer/ (accessed June 25th 2025).

Koh, Y. W. & **Nishimori, H.** 2022 Quantum and classical annealing in a continuous space with multiple local minima. *Physical Review A* **105** (6), 062435.

Koshka, Y. & **Novotny, M. A.** 2020 Comparison of D-wave quantum annealing and classical simulated annealing for local minima determination. *IEEE Journal on Selected Areas in Information Theory* **1** (2), 515–525.

Lécuyer, C. 2022 Driving semiconductor innovation: Moore's law at fairchild and intel. *Enterprise & Society* **23** (1), 133–163.

Leymann, F. & **Barzen, J.** 2020 The bitter truth about gate-based quantum algorithms in the nisq era. *Quantum Science and Technology* **5** (4), 044007.

Li, Y., **Song, L.**, **Sun, Q.**, **Xu, H.**, **Li, X.**, **Fang, Z.** & **Yao, W.** 2022 Rolling bearing fault diagnosis based on quantum ls-svm. *EPJ Quantum Technology* **9** (1), 1–15.

Liu, Y., **Liu, J.**, **Raney, J. R.** & **Wang, P.** 2024 Quantum computing for solid mechanics and structural engineering–a demonstration with variational quantum eigensolver. *Extreme Mechanics Letters* **67**, 102117.

Lucas, A. 2014 Ising formulations of many np problems. *Frontiers in Physics* **2**, 5.

Mahmoudi, Y., **Zioui, N.**, **Belbachir, H.**, **Tadjine, M.** & **Rezgui, A.** 2022 A brief review on mathematical tools applicable to quantum computing for modelling and optimization problems in engineering. *Emerging Science Journal* **7** (1), 289–312.

Main, D., **Drmota, P.**, **Nadlinger, D.**, **Ainley, E.**, **Agrawal, A.**, **Nichol, B.**, **Srinivas, R.**, **Araneda, G.** & **Lucas, D.** 2025 Distributed quantum computing across an optical network link. *Nature* **638**, 1–6.

Mellor, P., **Hoole, J.** & **Simpson, N.** 2021 Computationally efficient prediction of statistical variance in the ac losses of multi-stranded windings. In *2021 IEEE Energy Conversion Congress and Exposition (ECCE)*, pp. 3887–3894. IEEE.

Merali, Z. 2011 First sale for quantum computing. *Nature* **474** (7349), 18.

Microsoft Team December 2017 *Announcing the Microsoft quantum development kit.* https://azure.microsoft.com/en-us/blog/quantum/2017/12/11/announcing-microsoft-quantum-development-kit/ (accessed June 25th 2025).

Monroe, C., **Meekhof, D. M.**, **King, B. E.**, **Itano, W. M.** & **Wineland, D. J.** 1995 Demonstration of a fundamental quantum logic gate. *Physical Review Letters* **75** (25), 4714.

Morita, S. & **Nishimori, H.** 2008 Mathematical foundation of quantum annealing. *Journal of Mathematical Physics* **49** (12).

Morstyn, T. 2023 Annealing-based quantum computing for combinatorial optimal power flow. *IEEE Transactions on Smart Grid* **14** (2), 1093–1102.

Nakamura, Y., **Pashkin, Y. A.** & **Tsai, J.** 1999 Coherent control of macroscopic quantum states in a single-cooper-pair box. *Nature* **398** (6730), 786–788.

Neven, H. December 2017 *Meet willow, our state-of-the-art quantum chip.* https://blog.google/technology/research/google-willow-quantum-chip/ (accessed June 25th 2025).

Nielsen, M. A. & **Chuang, I. L.** 2001 Quantum computation and quantum information. *Physics Today* **54** (2), 60.

O'Brien, J. L., **Pryde, G. J.**, **White, A. G.**, **Ralph, T. C.** & **Branning, D.** 2003 Demonstration of an all-optical quantum controlled-not gate. *Nature* **426** (6964), 264–267.

Pelofske, E., **Bärtschi, A.** & **Eidenbenz, S.** 2022 Quantum volume in practice: What users can expect from NISQ devices. *IEEE Transactions on Quantum Engineering* **3**, 1–19.

Pittman, T., **Fitch, M.**, **Jacobs, B.** & **Franson, J.** 2003 Experimental controlled-not logic gate for single photons in the coincidence basis. *Physical Review A* **68** (3), 032316.

Postler, L., **Heu, S.**, **Pogorelov, I.**, **Rispler, M.**, **Feldker, T.**, **Meth, M.**, **Marciniak, C. D.**, **Stricker, R.**, **Ringbauer, M.**, **Blatt, R.**, **Schindler, P.**, **Müller, M.** & **Monz, T.** 2022 Demonstration of fault-tolerant universal quantum gate operations. *Nature* **605** (7911), 675–680.

Preskill, J. 2018 Quantum computing in the NISQ era and beyond. *Quantum* **2**, 79.

Prickett, N. H. November 2017 *D-wave makes quantum leap with reverse annealing.* https://www.nextplatform.com/2017/11/14/d-wave-makes-quantum-leap-reverse-annealing/ (accessed June 25th 2025).

Quinton, F. A., **Myhr, P. A. S.**, **Barani, M.**, **del Granado, P. C.** & **Zhang, H.** 2025 Quantum annealing applications, challenges and limitations for optimisation problems compared to classical solvers. *Scientific Reports* **15** (1), 12733.

R. Computing January 2024 *Rigetti announces public availability of ankaa-2 system with a 2.5x performance improvement compared to previous qpus.* https://www.globenewswire.com/news-release/2024/01/04/2804006/0/en/Rigetti-Announces-Public-Availability-of-Ankaa-2-System-with-a-2-5x-Performance-Improvement-Compared-to-Previous-QPUs.html (accessed June 25th 2025).

Rajak, A., **Suzuki, S.**, **Dutta, A.** & **Chakrabarti, B. K.** 2023 Quantum annealing: An overview. *Philosophical Transactions of the Royal Society A* **381** (2241), 20210417.

Ranaweera, C., **Monti, P.**, **Skubic, B.**, **Wong, E.**, **Furdek, M.**, **Wosinska, L.**, **Machuca, C. M.**, **Nirmalathas, A.** & **Lim, C.** 2019 Optical transport network design for 5g fixed wireless access. *Journal of Lightwave Technology* **37** (16), 3893–3901.

Reynolds, M. June 2017 *Google on track for quantum computer breakthrough by end of 2017.* https://www.newscientist.com/article/2138373-google-on-track-for-quantum-computer-breakthrough-by-end-of-2017/ (accessed June 25th 2025).

Ripon, K. S. N. & **Torresen, J.** 2014 Integrated job shop scheduling and layout planning: A hybrid evolutionary method for optimizing multiple objectives. *Evolving Systems* **5**, 121–132.

Schiffmann, O., Hicks, B., Nassehi, A., Gopsill, J., and Valero, M. 2023 A cost–benefit analysis simulation for the digitalisation of cold supply chains. *Sensors* **23** (8), 4147. doi: 10.3390/s23084147.

Schindler, P., **Barreiro, J. T.**, **Monz, T.**, **Nebendahl, V.**, **Nigg, D.**, **Chwalla, M.**, **Hennrich, M.** & **Blatt, R.** 2011 Experimental repetitive quantum error correction. *Science* **332** (6033), 1059–1061.

Shankland, S. December 2019 *Quantum computing leaps ahead in 2019 with new power and speed.* https://www.cnet.com/tech/computing/quantum-computing-leaps-ahead-in-2019-with-new-power-and-speed/ (accessed June 25th 2025).

Shor, P. W. 1994 Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pp. 124–134. IEEE.

Sun, G., **Zhang, H.**, **Fang, J.**, **Li, G.** & **Li, Q.** 2018 A new multi-objective discrete robust optimization algorithm for engineering design. *Applied Mathematical Modelling* **53**, 602–621.

Sutter, H. 2005 The free lunch is over: A fundamental turn toward concurrency in software. *Dr. Dobb's Journal* **30** (3), 202–210.

Svore, K. February 2021 *Azure quantum is now in public preview.* https://azure.microsoft.com/en-us/blog/quantum/2021/02/01/azure-quantum-preview/ (accessed June 25th 2025).

Swayne, M. April 2024 *Psiquantum receives $940 million (AUD) from Australian government — May now be world's highest funded independent quantum firm.* https://thequantuminsider.com/2024/04/29/psiquantum-receives-940-million-aud-from-australian-government/ (accessed June 25th 2025).

Tapia, E. P. February 2024 *Grover's algorithm and amplitude amplification.* https://github.com/qiskit-community/qiskit-algorithms/blob/stable/0.2/docs/tutorials/06_grover.ipynb (accessed June 25th 2025).

Trueman, C. March 2025 *D-wave claims 'quantum supremacy,' some experts are Skeptical.* https://www.datacenterdynamics.com/en/news/d-wave-claims-quantum-supremacy-other-experts-are-less-sure/ (accessed June 25th 2025).

Ullah, M. H., **Eskandarpour, R.**, **Zheng, H.** & **Khodaei, A.** 2022 Quantum computing for smart grid applications. *IET Generation, Transmission & Distribution* **16** (21), 4239–4257.

Venegas-Andraca, S. E., **Cruz-Santos, W.**, **McGeoch, C.** & **Lanzagorta, M.** 2018 A cross-disciplinary introduction to quantum annealing-based algorithms. *Contemporary Physics* **59** (2), 174–197.

Wang, B. June 2015 *Dwave commercializes 1152 qubit chip but there are 2048 physical qubits so some chips will have more than 1152 qubits active.* https://www.nextbigfuture.com/2015/06/dwave-commercializes-1152-qubit-chip.html (accessed June 25th 2025).

Williams, C. P. 2001 Quantum search algorithms in science and engineering. *Computing in Science & Engineering* **3** (2), 44–51.

Xanadu September 2020 *Xanadu releases world's first photonic quantum computer in the cloud.* https://www.xanadu.ai/press/xanadu-releases-worlds-first-photonic-quantum-computer-in-the-cloud (accessed June 25th 2025).

Xanadu June 2022 *Beating classical computers with borealis.* https://www.xanadu.ai/blog/beating-classical-computers-with-Borealis (accessed June 25th 2025).

Yang, Z., **Zolanvari, M.** & **Jain, R.** 2023 A survey of important issues in quantum computing and communications. *IEEE Communications Surveys & Tutorials* **25**, 1059–1094.

Yingchareonthawornchai, S., **Aporntewan, C.**, **and Chongstitvatana, P.** 2012 An implementation of compact genetic algorithm on a quantum computer. In *2012 Ninth International Conference on Computer Science and Software Engineering (JCSSE)*, pp. 131–135. IEEE.

Zeng, W. December 2017 *Unsupervised machine learning on Rigetti 19Q with forest 1.2.* https://medium.com/rigetti/unsupervised-machine-learning-on-rigetti-19q-with-forest-1-2-39021339699 (accessed June 25th 2025).

Appendix A: Grover's algorithm and circuit diagrams

Figure A1 presents a high-level representation of Grover's algorithm using a circuit diagram. This means that the lowest level or fundamental quantum gates are not shown, but the functions that they provide when combined are shown instead. These diagrams are read from left to right, with vertically aligned gates being applied simultaneously – much like a music stave. Time flows from left to right, meaning earlier operations appear on the left, and later operations appear on the right. The diagrams are composed of "wires." These are the horizontal lines spanning the width of the diagram and represent a qubit as it passes through/is operated on by each of the gates. Operations on qubits are represented by boxes, such as the one containing "G" on top of the wires in Figure A1.

Figure A1 shows us that Grover's algorithm requires a register of n qubits (problem qubits) and a second register of computational bits (or the oracle workspace). The problem qubits are used to represent the problem variables and encode the solution to our problem when measured. The computational bits allow computations to be performed on the problem qubits throughout the algorithms operation. The circuit begins with all the problem qubits in the ground state $|0\rangle$ then applies a Hadamard gate to each wire, or qubit. This places each qubit in a state of equal superposition between the states $|0\rangle$ and $|1\rangle$, shown in Equation (7) (and shown pictorially in Figure 2).

$$|0\rangle \rightarrow \frac{|0\rangle + |1\rangle}{\sqrt{2}} \tag{A.1}$$

The square of the coefficients of the states represent the probability of that state being returned $(1/\sqrt{2})^2 = 1/2)$. We call this an equal superposition as there is a 50% chance that the qubit, if measured at this stage, would return a 0 and a 50% chance it would return a 1.

Once in this state of superposition, the circuit undergoes its first iteration of Grover's algorithm, *G*. An iteration is shown on the circuit diagram by the rectangle containing the letter G. This iteration can be broken up into four steps (Nielsen & Chuang 2001) (and how these steps might affect the solution space can be seen in Figure 2):

1. Apply the Oracle. The Oracle is another combination of more elementary gates whose purpose is to mark or recognise a solution to the problem. This marking is carried out by flipping a qubit in the Oracle workspace (or in the computational qubit register) from $|0\rangle$ to $|1\rangle$ if a solution is detected.
2. Apply a Hadamard to each qubit in the problem qubits register.
3. Perform a "phase shift" which has the effect of increasing the amplitude of the target state, or the probability of it being returned.
4. Apply a Hadamard to each qubit in the problem qubits register.

This *G* can be repeated many times, and the optimum number is a function of problem size. For problems where there are M valid solutions in a space of size N the Oracle, *O*, should be applied $O\sqrt{N/M}$ times. For our problem that is $\sqrt{4096/56} = 8.5 \approx 9$ times.

Finally, Figure A1 shows that the qubits in the problem qubits register are measured, as indicated by the block containing a dial symbol. This means that state represented by the linear combination of the problem qubit states collapses into a binary string, 1 digit per problem qubit. This binary string can be decoded to return the solution we were searching for.

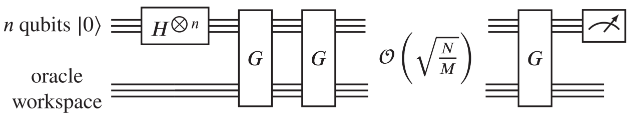


Figure A1. A figure showing the schematic circuit for Grover's algorithm (Nielsen & Chuang 2001).

Appendix B: Sources for Figure 3

Table B1. A table detailing the supporting references used to create Figure 3, organised by the QC era

Quantum computing era	Supporting references
Foundational algorithms and early theory	Feynman (2018), Apolloni, Cesa-Bianchi & De Falco (1990), Shor (1994), Grover (1996), Harrow, Hassidim & Lloyd (2009)
Rise of physical qubits and early hardware	Monroe *et al.* (1995), Chuang, Gershenfeld & Kubinec (1998), Nakamura, Pashkin & Tsai (1999), Dumke *et al.* (2002), Pittman *et al.* (2003), O'Brien *et al.* (2003), Johnson *et al.* (2011), Schindler *et al.* (2011)
Democratisation of quantum hardware	Devitt (2016), Prickett (2017), IBM (n.d.c), Team (2017), Knight (2017), Preskill (2018), Kelly (2018), Ball (2018), D-Wave (2018), Arute *et al.* (2019), IBM (n.d.b), Choi (2020), Trueman (2025)
Scaling qubits and improving processors	AbuGhanem (2025), Svore (2021), Postler *et al.* (2022), Kim *et al.* (2023), Swayne (2024), Main *et al.* (2025), Google Quantum AI and Collaborators (2024), D-Wave (2025), King *et al.* (2025)

Table B2. A table detailing the supporting references for the maximum qubit count achieved by each quantum compute provider, organised by provider

Compute provider	Supporting references
IBM	Knight (2017), AbuGhanem (2025), Devitt (2016)
Google	Kelly (2018), Reynolds (2017), Neven (2017)
Rigetti	Zeng (2017), Shankland (2019), Pelofske, Bärtschi & Eidenbenz (2022), Computing (2024)
Xanadu	Xanadu (2020), Xanadu (2022)
D-wave	Prickett (2017), Trueman (2025), Merali (2011), Jones (2013), Wang (2015), D-Wave (2025)

https://doi.org/10.1017/dsj.2025.10028 Published online by Cambridge University Press