University of Sussex

Applications and Hardware Considerations for Quantum Computing

Mark Webber

Submitted for the degree of Doctor of Philosophy

University of Sussex

September 2021

Declaration

I hereby declare that this thesis has not been and will not be submitted in whole or in part to another University for the award of any other degree.

Signature:

Mark Webber

UNIVERSITY OF SUSSEX

Mark Webber, Doctor of Philosophy

Applications and Hardware Considerations for Quantum Computing

Abstract

Quantum computers are expected to one day be able to solve a set of problems which are practically impossible with classical super computers, even with their projected continued improvement. As the field of quantum computing has continued to evolve the somewhat disparate research areas of algorithms and hardware have improved in their integration. Fully optimizing quantum algorithms requires a solid understanding of the quantum hardware, and metering experimental hardware priorities requires an understanding of the general algorithm requirements. This thesis initially provides an overview of quantum hardware and applications and discusses their interplay in both the near term and fault tolerant regime. A greater focus is placed on trapped ion architectures in this thesis and in particular the shuttling based approach of the Ion Quantum Technology group at the University of Sussex.

A routing algorithm is provided which can efficiently enable all to all connectivity for the shuttling based trapped ion design without positional swaps. A simulation tool was created and used to develop and characterize routing algorithms. The cost of enabling connectivity in Noisy-Intermediate-Scale-Quantum devices is an important factor in determining computational power. The core ideas of this routing algorithm are currently being integrated into a software compiler stack that will control real quantum hardware. An error model for the shuttling based design is presented which makes use of the time cost for connectivity results from the simulation tool. The error model is used to estimate the computational power (quantum volume) of the design as a function of experimental parameters. The error model can be used to help meter experimental priorities by identifying the most impactful parameters across particular regimes. A comparison is performed using metrics such as Quantum Volume, between the shuttling based trapped ion design and a superconducting grid which uses logical swaps to enable connectivity, and it is found that the trapped ion design has a substantially lower cost associated with connectivity. Large scale trapped ion devices are considered and the total time required to enable all to all connectivity is estimated for both the modular shuttling approach and for the approach that uses small scale modules connected via photonic interconnects.

A review of fault tolerant methods for quantum chemistry is presented. Resource estimations are provided all the way down to the required wall-clock time and number of physical qubits, for ground state energy calculations for molecules across different basis set sizes. The basis set size at which a quantum computer can meaningfully outperform a classical supercomputer is estimated. Determining the point at which a quantum advantage may be realised can help the field progress by setting realistic expectations and by having a device size to aim for. The impact of hardware considerations such as the code cycle time is investigated by including a wider range of possible surface code error correction configurations. Two distinct methods are investigated which allow one to incrementally speed up the rate of computation until the time optimal limit is reached by introducing additional qubits. The number of physical qubits required to reach a desirable run time is estimated as a function of the hardware's code cycle time, for problems such as the ground state estimation of the FeMoco molecule, and for breaking the encryption of the Bitcoin network. It is found that for the quantum advantage problems investigated in this work, hardware with considerably slower code cycle times than the more usually considered $1\mu s$ of superconducting qubits, will still be able to reach desirable run times provided enough physical qubits are available.

Publications

Articles

M. Webber, V. Elfving, S. Weidt and W.K. Hensinger, *The Impact of Hardware Specifications on Reaching Quantum Advantage in the Fault Tolerant Regime.* In journal submission process. ArXiv ID: 2108.12371. 2021

V. Elfving, B.W Broer, **M. Webber**, J. Gavartin, M.D. Halls, K.P. Lorton and A. Bochevarov, *How will quantum computers provide an industrially relevant computational advantage in quantum chemistry?* Available on ArXiv, ID: 2009.12472. 2020

M. Webber, S. Herbert, S. Weidt and W.K. Hensinger, *Efficient Qubit Routing for a Globally Connected Trapped Ion Quantum Computer.* Advanced Quantum Technologies, vol. 3, p. 2000027, 2020

Conference contributions

M. Webber, S. Herbert, S. Weidt, and W.K. Hensinger, *Efficient qubit routing for a globally connected trapped ion quantum computer.* Quantum Technology International Conference 2020 (Virtual conference presentation)

M. Webber, S. Herbert, S. Weidt, and W.K. Hensinger, *Enabling global connectivity in a shuttling based trapped ion quantum computer with efficient routing.* IEEE International Conference on Quantum Computing and Engineering 2020 (Virtual conference poster presentation)

M. Webber, S. Herbert, S. Weidt, and W.K. Hensinger, *Enabling global connectivity in a shuttling based trapped ion quantum computer with efficient routing.* Conference on

Quantum Machine Learning Plus 2018 (Conference poster and flash talk)

M. Webber, S. Weidt, B. Lekitsch, J. Randall, S.C. Webster, E.D. Standing, A.E. Webb, T. Navickas, I. Cohen, K. Lake, N. Johnson, R. LeBrun-Ricalens, A.G. Fowler, K. Molmer, S.J. Devitt, Ch. Wunderlich, A. Retzker, and W.K. Hensinger, *Roadmap for the construction of a large-scale trapped ion quantum computer.* Quantum Machine Learning and Biomimetic Quantum Technologies 2018 (Conference poster)

Acknowledgements

A deep thank you to my supervisor, Professor 'Winni' Hensinger, for the advice and direction you have given me throughout my PhD, and for your can-do attitude which has influenced me greatly. Thank you to Dr. 'Seb' Weidt for setting high expectations from the beginning, especially with the external collaborations and outreach. Our regular meetings kept me on track and grounded. Thank you to Dr. Steven Herbert for our discussions on the routing algorithm which eventually resulted in my first published work. I'd like to thank Dr. Vincent Elfving for our long lasting collaboration, it has been a pleasure working with you and I'm glad to have now expanded my knowledge base to include the fault-tolerant side of things. Thank you to Jansen, whom I met at my first international conference; I fondly remember our conversations and the following adventures. A big thank you to Mitch for our meaningful friendship throughout our PhD, if were it not for the recent disruption, I bet we would be enjoying a plate of nachos at the moment - and maybe even a bubble tea. Thanks to Zak for all of your help with outreach, and our continued work together. Thank you David for your help with gate decomposition and for shooting me with your gun. Thank you to all my IQT colleagues past and present for making work such a great place to be, I wish I could have been around more over the final year. Thank you to all of my new UQ colleagues, I'm looking forward to what we will build together. Thanks to my grandparents whom are no longer with us for their love and support. Thanks to my Nan for providing me with an office space when I needed it, and for our games of scrabble. I am deeply grateful to my mother for the unconditional love she has always shown me and for teaching me how to use a spoon. Thank you Robert for all of the support you have given to me and mum over the years. Dad, a deep thank you for always tending my curiosity and for our meaningful adventures. Thank you to Edyta for always being so welcoming both here and in Poland. I really appreciate my little sister Matilda, for bringing so much fun into my life. I am grateful to Ginny's family for making me feel so at home whenever I visit. Finally, thank you Ginny for being my partner in life, I don't think I would have got this far without you.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Before the advent of electronic computing technology, a "computer" generally referred to a person whose profession involved routine calculations. In the last 50 years our world has been revolutionized by digital computing. Computers have been used to produce rich models of the physical universe and were instrumental to the creation of the internet, which has provided unprecedented connectivity between humans. The free flow of information between individuals and the ability to self navigate the near-entirety of human knowledge can be seen as a magnifying force upon our underlying tendencies, for better and for worse. Quantum computers are expected to one day drastically push forward the boundaries of what is possible for us to simulate and investigate through computation.

1.1 Classical computing

In the broadest sense, a computer transforms information from one state to another by the use of a physical process. Modern computers use transistors to represent a binary state (bit). Any general function which defines a transformation of information (input to output) can be decomposed into a set of simple logical operations, or gates. An example universal set of logic gates for irreversible classical computation is the AND, OR, and NOT gate. The NAND gate alone represents a universal gate set, it takes two inputs and gives a single output, and therefore is a clear example of irreversible logic. The additional bit is effectively erased, which has an associated change in entropy of $\ln 2$. This corresponds to an energy increase of $kT \ln 2$ where k is the Boltzman's constant and T is the temperature, although this degree of heating pales in comparison to the actual dissipation of a transistor at $\sim 10^{10} kT$ [1]. Despite this, in the 1970s there was was a strong motivation to identify whether computation can be done in a reversible manner. The Toffoli gate [2] is a three bit reversible logical operation capable of universal computation, which is a property that no classical two bit operation can have. The Toffoli gate is able to condense the information of two bits into one without erasing information because of the additional bit.

Moore's law predicts the exponential rate of improvement of classical computing; it started as a forecast by Gordon Moore in 1965 and has since proved itself accurate. The prediction was originally motivated as an economic argument, where the "the cost per component is nearly inversely proportional to the number of components" (per chip) [3]. Some of the first computers developed utilized vacuum tubes, 1000's of which would fill an entire room. The Intel 4004 is considered the world's first microprocessor and it was released in 1971 with transistor width of 10 μm. Now in 2021, transistors have a width of $\sim 10nm$. It is possible to physically produce smaller sizes with silicon, however, below a size of $7nm$ the transistors are so close together that electrons may regularly experience quantum tunneling, which introduces errors. The end of Moore's law has been proclaimed numerous times over the last 20 years, but at each instance new developments have enabled progress. A proof of concept has been demonstrated using carbon nanotubes and molybdenum disulfide for transistor sizes of $\sim 1nm$ but further development is required for mass production. We must imagine that eventually a point will be reached where transistors can no longer be made smaller, with the size of atoms at $\sim 0.1nm$. Managing heat build up is also a primary concern for the continued improvement of classical computing, for example, increasing the clock cycle time leads to a cubic increase in power consumption for the chip. In part due to this, clock cycle times have entirely plateaued in recent years, where instead speedups are sought by increasing the number of cores and utilizing parallelization.

There are classes of problems which may be extremely academically or commercially interesting, but that have an unfavourable run time scaling with a classical computer. For example, factoring large numbers scales superpolynomially with problem size for classical computers, and encryption techniques such as RSA rely upon this. In many areas of computing, from simulating chemistry to big data analysis, there is an upper limit to the problem size that can be feasibly tackled with classical computers. There exist quantum algorithms which for some problems, such as the ones listed above, can provide an exponential or near-exponential improvement in the scaling with problem size.

1.2 Quantum computing

Richard Feynman was perhaps the first to clearly envisage a quantum computer and its application to the simulation of physics and chemistry [4]. The first major work towards a quantum model for computation was published by Deutsch in 1985 [5] and in 1994 Shor presented a quantum algorithm for factoring large numbers with an exponential speed up relative to classical computing [6]. Quantum computing is an extension to classical computing and a quantum algorithm can be simulated by a classical computer albeit with an exponential overhead.

1.2.1 The qubit

The quantum bit, or qubit, can have numerous physical implementations. The spin of an electron, the energy levels of a fabricated superconducting circuit, and the polarization of a photon to name a few. Superposition is the archetypal property that qubits possess that distinguishes them from a classical bit. The classical bit can only be in one of its two states at any moment in time, whereas in contrast a qubit can be in any superposition (linear combination) of those two states. A general single qubit state can be expressed in the form $|\psi\rangle = a\,|0\rangle + b\,|1\rangle$, where a and b are probability amplitudes, and the probability of observing the qubit in the $|0\rangle$ state is equal to $|a|^2$, with $|a|^2 + |b|^2 = 1$. The probability amplitudes can be complex numbers, and this notation is an efficient way of capturing the wave like nature of quantum mechanics, such as constructive and destructive interference. A single qubit can be helpfully visualized with the Bloch sphere representation where its state is described by a point on the surface of a unit sphere, see figure 1.1.

1.2.2 Superposition

In the Bloch sphere model the north pole represents the $|0\rangle$ state, and the south pole represents the $|1\rangle$ state, but at one moment in time the state can exist anywhere on the surface. The only valid single qubit operations in this picture correspond to rotations of the sphere. In classical computing we can measure the state of any qubit freely without affecting anything, whereas in quantum computing the measurement process is destructive and collapses the single qubit superposition into one of the basis states. For example, a state described by a point on the equator, $1/\sqrt{2}\,(|0\rangle + |1\rangle)$ of the sphere is in an equal superposition, and if a measurement is performed in the standard basis then the state will collapse with equal probability into either the $|0\rangle$ or $|1\rangle$ state. Following the Bloch sphere representation we can define an arbitrary single qubit state as $|\Psi\rangle = \cos(\theta/2)\,|0\rangle +$

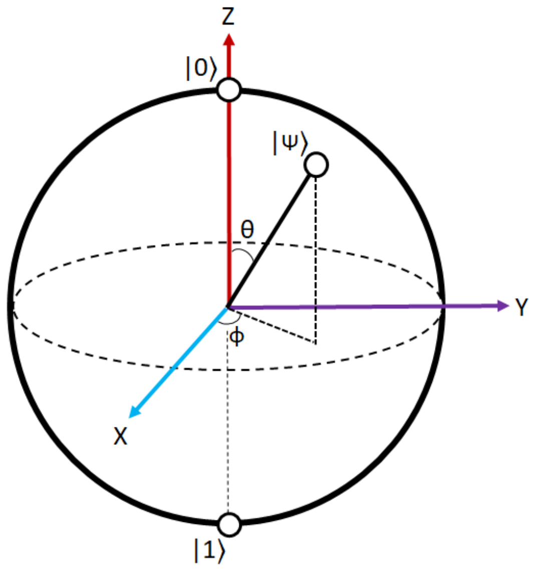


Figure 1.1: The Bloch sphere representation for a qubit with basis states $|0\rangle$ and $|1\rangle$, the three rotational axes, X, Y and Z. The arbitrary state $|\Psi\rangle$ can be described by two angles, the angle from the Z axis, θ, and the angle from the X axis, ϕ.

$(\cos\phi + i\sin\phi)\sin(\theta/2)|1\rangle$ where $0 \leq \theta \leq \pi$ and $0 \leq \phi < 2\pi$. An n qubit state defines a vector space that can be described via a column vector of length 2^n upon which quantum gates operate. A quantum gate that operates on n qubits is described by a $2^n \times 2^n$ unitary matrix. The single qubit Pauli X, Y, and Z matrices correspond to π rotation around the respective axis shown in figure 1.1 and are listed below:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \; Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \; Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{1.1}$$

Another two important quantum gates are the Hadamard (H) and the controlled not (CNOT). The H gate takes the $|0\rangle$ state to a superposition state $1/\sqrt{2}\,(|0\rangle + |1\rangle)$, and the CNOT gate is a two qubit gate where the operation (NOT) is dependent on the state of the controlling qubit. Their corresponding matrices are:

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \; CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{1.2}$$

A two qubit state in superposition can be generated by applying the Hadamard gate on each qubit as shown in figure 1.2:

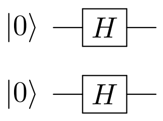


Figure 1.2: Circuit for preparing a two qubit superposition state

The state space of a quantum system with n qubits can be represented as the tensor product, $\otimes$, of the respective state spaces of all of the individual qubits. We can abbreviate: $|0\rangle \otimes\ |1\rangle$ as $|01\rangle$. This circuit has an output with a combined description of both qubits, $|\psi\rangle\ =\ 1/2\ (|00\rangle + |01\rangle + |10\rangle + |11\rangle)$, but one can also consider the separable (tensor product) description where each qubit is in the state $1/\sqrt{2}\ (|0\rangle + |1\rangle)$. This will be a relevant consideration when comparing against an entangled state.

1.2.3 Entanglement

Entanglement follows as a consequence of superposition, and was described as "spooky action at a distance" by Einstein. Entanglement is sometimes mistakenly prescribed to enabled faster than light communication, and while the correlation between two particles can be independent of distance, knowledge of the correlation requires ordinary information transfer. We can generate a Bell pair, which is an entangled two qubit state, with a simple circuit consisting of just a H gate, and a CNOT, as shown in figure 1.3:

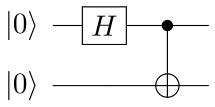


Figure 1.3: Circuit for preparing an entangled state with a Hadamard gate followed by a CNOT where the black dot represents the control.

Just as qubits can be in superposition, so can the application of logical operations. After the H gate the first qubit is in superposition, and following the controlled operation the output state is $|\psi\rangle\ =\ 1/\sqrt{2}\ (|00\rangle + |11\rangle)$. There is no separable (tensor product) description for this state, i.e. there is no valid description for the individual qubit states that also satisfies the combined description. This is contrary to the previous example of two qubit superposition and the lack of a separable description is one of the fundamental indicators of entanglement.

1.2.4 Universal quantum computation

The CNOT gate in combination with arbitrary single qubit rotations are sufficient for universal quantum computation, and so any arbitrary algorithm could be decomposed into such a fixed gate set. Any arbitrary algorithm on n qubits may be described by a unitary matrix of size $2^n \times 2^n$. Quantum gates are implemented via the action of a Hamiltonian over a specific time which results in a unitary time evolution according to the Schrödinger equation. Valid quantum gates are always unitary as it is this property which ensures that the sum of probabilities for all possible outcomes of a quantum state remains exactly 1. The Clifford gates are those that map Pauli operators onto other Pauli operators, and the set can be generated by various combinations of the set $\{H, CNOT, S\}$ where the S gate is $Z^{1/2}$. The Gottesman-Knill theorem [7] states that any Clifford circuit of finite size can be simulated in polynomial time with a classical computer. The Clifford gate set in combination with any non-Clifford gate is sufficient for universal quantum computation; two of the most commonly considered non-Clifford gates are the T gate ($Z^{1/4}$) and the Toffoli gate. This restricted gate set is more often considered in the context of fault tolerant quantum computing. For example, the surface code has easy (low-overhead) access to the Clifford gate set, but more costly methods are necessary to get access to a non-Clifford gate, such as magic state distillation. The Clifford S gate is not available transversely in the surface code but may be made available with relatively low overhead code manipulation techniques [8]. The Toffoli gate can be decomposed in terms of Clifford gates and T gates, and any arbitrary angle single qubit rotation can be decomposed into sequences of T gates and H gates, as per the Solovay-Kitaev theorem [9].

1.2.5 The DiVincenzo criteria

In 2000 David DiVincenzo proposed a list of criteria that are necessary for the construction of a functional quantum computer [10].

1. *A scalable physical system with well characterized qubit.*

   The current state of the art involves using 10s of qubits, and it will be necessary to increase the size of these devices to tackle larger problems. Error correction techniques incur a large physical qubit overhead and so an architecture must indeed be very scalable to realise a fault tolerant device.

2. *The ability to initialize the state of the qubits to a simple fiducial state*

   For example we must be able to prepare the state $|0...0\rangle$ to enable a consistent starting point for algorithms.

3. *Long relevant decoherence times*
   The decoherence times will need to be long relative to the time duration required for gates, so that multiple gates can be applied before coherency is lost.

4. *A universal set of quantum gates*
   The available gate set for a particular architecture may be quite restricted at the physical level, but with a universal set (such as CNOT and arbitrary single qubit rotations) any arbitrary algorithm can be performed with an appropriate decomposition.

5. *A qubit-specific measurement capability*
   In the case of multi qubit measurement we must be able to identify which measurement result corresponds to which qubit.

Two further criteria were later added by DiVincenzo which are necessary for quantum communication. Quantum communication may be a necessary feature for particular modular quantum computing designs, such as the use of photonic-interconnects between small modules of trapped ion processors.

1. The ability to inter-convert stationary and flying qubits

2. The ability to faithfully transmit flying qubits between specified locations

The following chapter provides an overview for some of the leading quantum computing platforms with a focus on trapped ion architectures. The chapter also introduces many of the most prominent quantum algorithms and characterizes them by their required stage of hardware development, near term or fault tolerant.

1.3 Summary of thesis

The aim of this thesis is to help bridge together quantum hardware and applications. As a theoretical research student within an experimental ion trapping group, I initially focused on problems relevant to our shuttling based design. This included developing a routing algorithm to enable global connectivity, and developing error models to assess the design and meter experimental priorities. The latter half of my work was more hardware agnostic, and I investigated fault tolerant quantum chemistry algorithms and resource estimation within the surface code. I then brought in broader hardware considerations, such as the how the clock rate (code cycle time) of the hardware can influence the ability to achieve a quantum advantage.

Chapter 2 provides an overview of quantum computing applications and hardware. The NISQ and fault tolerant regime are distinguished, and relevant hardware considerations for each are introduced.

In chapter 3 we present a routing algorithm for enabling global connectivity in a shuttling based trapped ion architecture, and quantify its properties against device considerations using a developed simulation tool. We compare the routing algorithm, which uses no positional swaps, against a positional swap based routing algorithm, and also investigate the effect of increasing the ion density for a given device.

Chapter 4 will introduce an error model which uses the cost of connectivity results of the previous chapter. The error model is used to estimate the achievable circuit depth as a function of experimental parameters such as gate fidelity. We use the Quantum Volume metric to assess the shuttling based design, and focus on the cost of enabling global connectivity. We compare the shuttling based design against a model for a superconducting device which enables global connectivity through sequences of nearest neighbour swap interactions.

Chapter 5 presents methods for calculating the logical resource requirements for fault tolerant quantum chemistry algorithms. We use quantum algorithm construction tools, such as Microsoft's SDK, *Q#*, and compare the resources for different Hamiltonian simulation techniques. We present physical qubit and run time requirements for threshold quantum advantage chemistry applications by defining the characteristics of a fault tolerant device, and briefly investigate some space and time optimization choices that are available.

Chapter 6 thoroughly investigates the available time optimization choices available within a surface code error corrected quantum computer. The code cycle time (base unit of operation in the surface code) may vary by orders of magnitude between different architectures, and so we investigate to what extent slower code cycle times can be mitigated by utilizing additional qubits. We parallelize layers of T gates which enables one to linearly trade physical qubits for run time. We investigate the physical resource requirements of applications in chemistry and cryptography which have strong commercial incentives.

1.4 Contributions

Chapter 3 and 4 follow the results and narrative of the paper titled “Efficient Qubit Routing for a Globally Connected Trapped Ion Quantum computer” [11] which was published in Advanced Quantum Technologies in August 2020. The simulation tool used to investigate routing and the overarching form of the routing algorithm were created by myself. Dr Steven Herbert provided helpful early discussions on the routing and later provided a unique routing feature for ions assigned to interior gate zones. I wrote the manuscript with the aid of invaluable feedback, provided by the co-authors Dr Steven Herbert, Dr Sebastian Weidt, and Prof. Winfried Hensinger. The two chapters also include the results of some investigations that were performed after the publication of the paper.

Chapter 5 follows the paper “How will quantum computers provide an industrially relevant computational advantage in quantum chemistry?” [12]. The paper is of a broad perspective type and involved collaboration with experienced classical-quantum chemists. I contributed by using Microsoft’s *Q#* software to calculate the resources required for ground state energy calculations of particular molecules. The work involved adapting existing *Q#* code for our purposes and using the chemistry software NWCHEM to generate the electron integrals for particular molecules. I contributed small sections to the manuscript, mostly relating to methodology as opposed to the perspective, and provided multiple rounds of feedback. The results provided by *Q#* were used by lead author Dr. Vincent Elfving to calculate the final resource estimations. The final results presented for the Qubitization method made use of alternative analysis which was performed by Dr. Vincent Elfving.

Chapter 6 follows the paper ”The Impact of Hardware Specifications on Reaching Quantum Advantage in the Fault Tolerant Regime” [13], which at the time of writing is in the journal submission process. I developed the general resource calculator tool which utilizes the error correction methods as laid out by Litinski [14]. I used the tool to assess a wide range of considerations, including the code cycle time of the hardware, and the measurement depth of the logical algorithm. I wrote the manuscript and was aided by multiple rounds of feedback from co-authors, Dr Vincent Elfving, Dr Sebastian Weidt, and Prof. Winfried Hensinger.

Chapter 2

An overview of hardware and applications

This chapter provides an overview of the available hardware types, and introduces a variety of quantum algorithms and their applications. Considerations for running quantum algorithms on realistic hardware are discussed.

2.1 NISQ and fault tolerance

The quality of logical operations in a near term device sets an upper limit on the achievable circuit depth (the number of sequential logical operations). Many of the original quantum algorithm proposals require a circuit depth which greatly exceeds the capabilities of a noisy device, and so error correction techniques must be utilized to enable their use. The quantum threshold theorem states that a quantum computer using error correction schemes, and a physical error below a certain threshold, can suppress the logical error rate to arbitrarily low levels and therefore could run an algorithm with an arbitrarily long circuit depth provided enough physical qubits are available [15, 16, 17]. There is a large overhead of physical qubits associated with error correction, and depending on the degree of error suppression, the ratio of logical qubits to physical qubits may be of the order of 1:1000. The number of qubits in state-of-the-art quantum computers is in the range of 10-100 at the moment, and it will be many years before a quantum computer has sufficient qubits to be fully fault tolerant. This long estimated timeframe motivated researchers to ask whether there is anything useful that can be done with small scale noisy quantum computers, and in 2018 Preskill coined the term "NISQ" (noisy intermediate scale quantum) [18]. Algorithms have now been designed specifically for NISQ quantum computers, and they generally consist of a low circuit depth which must be iterated many times. In this chapter we will provide an overview of the prominent algorithms for these two regimes and their potential applications.

2.1.1 NISQ and computational power

Numerous research teams and companies are developing quantum platforms and it is of interest to compare the strengths and weaknesses of the different approaches. When assessing near term quantum computers, the concept of "quantity hype" is helpful. Quantity hype refers to the occasional over emphasis on the number of qubits within a NISQ device, and while this is an important figure of merit, when stated alone it fails to reflect on the quality of the control over the qubits. The computational power of a NISQ device will be a function of both the number of qubits, and the achievable circuit depth. A higher degree of control on the qubits, which can relate to the coherence time, or the fidelity of logical operations, implies the ability to reach longer circuit depths. The metric quantum volume was proposed by IBM [19, 20]. Quantum volume is a hardware agnostic measure of computational power for NISQ devices and it achieves this by defining a standardized random circuit to benchmark against. The metric places equal importance between the number of physical qubits and the achievable circuit depth. It includes many relevant factors such as the power of the native logical gate set, the cost of enabling connectivity between qubits, the fidelity of operations. We will cover quantum volume in greater detail in chapter 4.

2.1.2 Fault tolerance and error correction

The most prominent error correction technique is the surface code which relies only on nearest neighbour interactions at the physical level, and has one of the most favourable threshold error rates at $\sim 1\%$ [21]. This threshold error implies that if the base physical error can be kept below 1% then the device would be able to run algorithms with arbitrary circuit depth as long as the sufficient number of physical qubits can be reached to encode the problem and cover the associated overhead. The error rates here refer to a depolarizing error channel where there is a probability (error rate) of introducing a random Pauli error per operation, in addition to the desired operation. One cannot confidently convert experimentally achieved gate fidelities into error rates of this form without further characterization procedures, and we expand on this issue in more detail in chapter 6.

There is a well developed theory of classical error correction; the underlying principle is redundancy, where the degree of protection can be increased by increasing the number of bits used to represent a single bit. There is no straightforward method for adapting the classical error correction techniques over to a quantum system, and the clearest challenge is the no-cloning theorem [22]. The no-cloning theorem states that it is impossible to construct a general unitary operator U_{clone} which can perform the operation (for an arbitrary unknown state):

$$U_{clone}(|\psi\rangle \otimes |0\rangle) \rightarrow |\psi\rangle \otimes |\psi\rangle \ . \tag{2.1}$$

An additional complication for quantum error correction is that there is more than one potential error type, in additional to the standard bit flip error (X-error in our Bloch sphere picture) there is also the potential for phase flip errors (Z-errors). Finally, a quantum error correction code must be very careful in its use of measurement operations for the detection, because a general quantum measurement collapses the quantum information into the logical basis. These issues were originally feared to be insurmountable but in 1995 Shor proposed the first quantum error correction scheme [23]. Following this, the arbitrary suppression of error was shown in the threshold theorem and these results strongly motivated continued research into physically realising a quantum computer.

The quantum error correction codes introduce redundancy by distributing the quantum information of the initial state into an entangled multi-qubit logical state, effectively expanding the Hilbert space. An example of a two-qubit encoder is [24]:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \rightarrow |\psi\rangle_L = \alpha\,|00\rangle + \beta\,|11\rangle = \alpha\,|0\rangle_L + \beta\,|1\rangle_L \tag{2.2}$$

Note that the above procedure is not equivalent to cloning the initial state. An ancillary qubit can be used to control particular operators on the logical qubit and it can then be measured without disturbing the encoded information to identify errors. This process is called syndrome extraction. The redundancy of this two qubit encoding is sufficient to detect the existence of an error but does not reveal which qubit the error occurred on and therefore is insufficient for correcting errors. The three qubit repetition code is the smallest instance of an error correction code. The code distance, d, is defined as equal to the numbers of errors that would convert one logical state to another, in the case of the three qubit repetition code this is 3, and it can be related to the number of errors that can be corrected, t, by $t = \lfloor((d-1)/2)\rfloor$. Generally, the greater the redundancy in the error correction code the more errors that can be corrected. In the surface code the number of physical qubits per logical qubit scales as $2d^2$. The surface code is estimated to have an approximate threshold of 1%, and the logical error rate per logical qubit p_L, per code cycle, as a function of the base physical error rate, p, can be approximated by [25]

$$p_L = 0.1(100p)^{(d+1)/2}. \tag{2.3}$$

Equation 2.3 defines a functional form but the specific coefficients (here 0.1 and 100) are determined through numerics with inherent assumptions which include the type of error model and the efficiency of the decoder. The act of encoding qubits in an error correction scheme can often make the available logical gate set more restricted. In the case of the surface code, only the Clifford gates can be applied transversally (or with low overhead), where transversal pragmatically means easy or cheap to apply. More specifically, a transversal gate implies that each qubit in a code block is acted on by at most a single physical gate and each code block is corrected independently when an error occurs. The Clifford S gate is not available transversely in the surface code but may be made available with relatively low overhead code manipulation techniques [8]. As mentioned previously, the Clifford gate set is insufficient for universal quantum computation, and so a non-Clifford gate must be introduced by more costly methods. The T gate can be constructed by consuming a magic state, $|m\rangle = (|0\rangle + e^{i\pi/4}|1\rangle)/\sqrt{2}$ [26], which can be produced with an error proportional to the physical error, independent of the code distance [27]. To create a sufficiently high quality magic state, a distillation protocol [28, 29] can be used which essentially involves converting many low fidelity states into fewer higher fidelity states. The time cost for a logical operation on an error corrected quantum computer will be many orders of magnitude greater than the cost for a logical operation on a classical computer. The first instances of a quantum advantage are likely to come from problems for which there exists an exponential speed up as opposed to a merely polynomial one.

When choosing an error correction protocol, perhaps the most important figure of merit is the threshold of the code, i.e. the error rate below which it is desirable to increase the degree of encoding. If experimental effective error rates are above the threshold of the code, then no benefit can be gained by using the encoding. The most well studied error correction scheme is perhaps the surface code which benefits from the highest threshold found so far at approximately 1% [25] (this value comes with many assumptions on the error model and decoder performance). Color codes are another well studied family, in short they have a more favourable encoding rate as a function of code distance, but suffer from a worse threshold than the surface code. Most codes require only nearest neighbour stabilizer measurements (for syndrome extraction), but hardware which has a low cost associated with physical connectivity may be suitable for codes with more demanding stabilizer measurements. Error correction codes which rely on global interactions at the physical level have favourable encoding rates as a function of code distance [30] but to determine their effective thresholds more research is needed, and it must include the effective cost of enabling that connectivity for a particular hardware.

2.1.3 Repetition code example

In this section we will demonstrate perhaps the simplest example of error correction, the three qubit repetition code. In this code the logical qubit is distributed across three entangled physical (data) qubits, and error information is extracted via interactions between the data qubits and two additional ancilla qubits. The ancilla qubits can then be measured "mid-circuit", and this "syndrome" is used to determine which corrections should be applied, if any. When measuring these syndromes we are measuring multi-qubit symmetries (the parity of stabiliser operators) that are common for both logical basis states, hence measuring these symmetries does not collapse the computational wave function and yet can still be used to detect errors. For this code the logical 0 is defined as $|000\rangle$ and the logical 1 is defined as $|111\rangle$; this repetition code can only protect against bit flip type errors (not phase) and therefore is not truly functional. Shor's first code to correct both phase and bit type errors involves concatenating this bit flip repetition code with itself in the conjugate basis, i.e. an alternative version which instead only protects against phase flips, requiring a total of 9 data qubits. Using figure 2.1 we can see how the repetition code protects against X type errors.

This repetition code uses 2 ancilla qubits to extract error information. When the qubits are pictured as arranged in a linear string of D_1, A_1, D_2, A_2, D_3, (this is a different orientation to figure 2.1 but it is functionally equivalent) each ancilla qubit interacts with the two neighbouring data qubits via a CNOT interaction. If the two data qubits are both in the $|1\rangle$ state, then both CNOTs are applied, if they are both in the $|0\rangle$ state, then neither CNOTs are applied, and therefore in both situations the middle ancilla qubit will be measured as 0. When the data qubits do not agree, i.e. 01 or 10, then only one CNOT is applied and the ancilla qubit will be measured as 1. The ancilla qubits then measure the parity of the two neighbouring data qubits, and by considering the value of both ancilla qubits after the syndrome extraction we can identify the most likely location of error. 00 implies no error, 01 implies $D_1 = D_2 \neq D_3$ therefore D_3 is the most likely source of error, and so on. Error correction codes can account for continuous errors as well (it does not have to be a pauli X (bit flip) error). When the data qubit is in superposition between $|0\rangle$ and $|1\rangle$ then the CNOT would be applied in superposition, but this combined state would

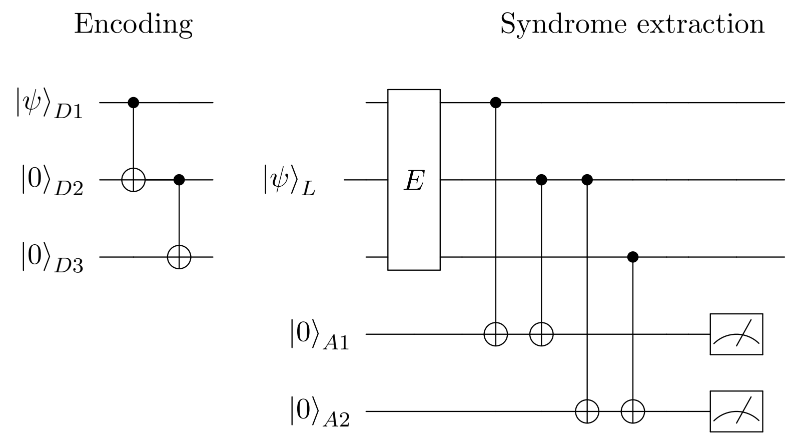


Figure 2.1: The circuit diagram for the three (data) qubit repetition code. First in the encoding stage, the arbitrary single qubit state $|\psi\rangle$ is entangled across three data qubits. Next the error block is applied, which for the code to be successful, will consist of a single arbitrary angle X rotation on any single data qubit. Next CNOTs with control on the data qubits and target on the ancilla qubits are used to extract error information. Finally the ancilla qubits can be measured mid-circuit, which can then be used to determine the necessary corrections.

then collapse upon ancillary measurement, effectively collapsing a continuous X error into either having occurred (full bit flip error) or not, with probability proportional to the magnitude of the error. When an error is introduced on two qubits, the 3 qubit repetition code will correct towards the majority and therefore fail. We can increase the distance of the code by increasing the redundancy, the 5 data qubit (4 ancilla) repetition code can correct errors on up to 2 qubits. The general repetition code of N qubits can correct for $\lfloor (N-1)/2 \rfloor$ errors.

2.2 Overview of hardware types

There are numerous physical systems being used to realize qubits, among them include: superconducting devices [31], trapped ions [32], neutral atoms [33], quantum dots [34], nuclear magnetic resonance [35], linear and nonlinear optical devices [36], and diamond NV centres [37]. In this section and in section 2.3 we provide an overview of some of the most developed hardware platforms, with a particular focus on trapped ions. The primary characteristics that determine the potential for a given hardware type include, gate times, coherence times, gate fidelities, scalability, and the cost for connectivity.

2.2.1 Superconducting devices

In 1999 the first superconducting qubit was developed [38] and now superconducting devices are one of the leading platforms; they are the primary focus of giants such as IBM and Google. Superconducting devices have high designability and rely on existing semiconductor micro-fabrication processing techniques. Each qubit must be fabricated with a non-perfect process which necessitates individual qubit calibration and characterization for near term devices. There are many types of superconducting qubits, but they all use a Josephson junction which is a break of weak connection between an otherwise continuous wire where current flows by quantum tunneling. This element introduces a non-linear inductance which transforms the otherwise harmonic oscillator into an an-harmonic oscillator. The an-harmonic oscillator has a non-constant gap between energy levels which allows a two level system to be defined. Charge, flux, and phase qubits are the three major archetypes of qubit type. Each uses a different feature as the qubit energy levels, but hybridisation between these three types exists. The transmon qubit was proposed in 2007 [39] and it is now one of the most popular categories of superconducting qubits. The transmon qubit benefits from a reduced sensitivity to charge noise as compared to other approaches. The qubits are generally designed to be compatible with microwave control which again leverages an existing mature technology. Superconducting devices must be operated at the $\sim 10mK$ level [40] which is a stringent cooling requirement for which a dilution refrigerator must be used . Dilution refrigerators generally require He-3 for this degree of cooling which in recent years is becoming increasingly scarce [41]. The requirement of a dilution refrigerator also presents serious engineering challenges for scaling superconducting devices, such as the co-location of the control and readout electronics which generally cannot withstand the low temperature. It is an active area of research to enable the control electronics to exist within the dilution refrigerator [42].

The lifetime of superconducting qubits has improved greatly since their inception. In 2010 the T_1 (depolarisation/relaxation) time of a transmon qubit was demonstrated to be $1.2\mu s$ [43], whereas in 2020 a T_1 time of $84\mu s$ was shown [44]. For more exotic superconducting qubit types even higher T_1 times have been demonstrated, such as $240\mu s$ for a 3D transmon [45], and $1600\mu s$ for a $0 - \pi$ qubit [46].

There are numerous methods of effectuating a two qubit interaction in superconducting devices, and the options will depend on the superconducting qubit type. One of the more commonly used two qubit interactions is the $iSWAP$ gate which when combined with arbitrary single qubit gates constitutes a universal gate set [47]. The CNOT gate can be constructed with two applications of the $iSWAP$ gate with additional single qubit operations. The two qubit gate interactions for superconducting devices are very fast (in comparison to trapped ion gates), which is a necessity due to the considerably lower coherence times. In 2016 McKay demonstrated a $\sqrt{iSWAP}$ gate with 98.2% fidelity taking 183ns [48]. More recently, in 2020, a two qubit gate fidelity of 99.5% was demonstrated with superconducting qubits [49].

Qubits within a superconducting device generally exist on a two dimensional grid. The qubits are static, and the two qubit interaction can be realized only between nearest neighbours. Long distance interactions between qubits must be decomposed into sequences of swap operations, for which the routing and scheduling can be optimized [50]. The costs for enabling global connectivity between a shuttling based trapped ion architecture and a superconducting square grid are compared in chapter 4 [11].

2.2.2 Photonics

A qubit can be defined within two modes of some degree of freedom of the optical field, which includes: orthogonal polarizations, propagation paths, frequency modes, and temporal bins. The numerous number of modes within a particular degree of freedom can be used to define qudits (a quantum state with greater than 2 levels), and in principle a single photon with many modes can represent multiple qubits. The scheme for performing n-dimensional unitary transformations on the mode space was originally outlined by Reck et al [51], but encoding multiple qubits into a single photon leads to an exponential number of optical components and therefore it is not a viable method for a scalable quantum computer [36]. A dual rail qubit is more often considered where each photon has two modes, with one photon per qubit.

Single qubit operations are easily implemented with interferometry but conditional two qubit operations are more challenging since they involve a nonlinear optical interaction. Early proposals for two qubit interactions simulate the nonlinear operations with linear optics and measurements which results in a probabilistic gate. In 2001 Knill, Laflamme and Milburn (KLM) proposed a scalable photonic scheme that enables non-deterministic two qubit interactions requiring only linear optical components [52]. There is a large overhead of ancilla qubits with the KLM scheme but the scaling of total components is linear in contrast to the exponential scaling of encoding a single photon across many modes [36].

Measurement-based quantum computing (MBQC) [53, 54], aka cluster state quantum computing, is an alternative method to the circuit model. It first prepares an entangled resource state and then the specifics of the computation are performed by single qubit measurements, which is a one-way process in contrast to quantum logical operations. There is a polynomial overhead associated with MBQC relative to the circuit model where the size of the cluster state scales with the circuit size (qubits x depth), and the number of single qubit measurement time steps scales with the depth of the circuit based approach. The MBQC approach is very appealing for photonic quantum computers because there are efficient means to generate the initial cluster state, such as with entangled photon sources, without the need for deterministic two qubit interactions. The cluster state is prepared first, which can be done probabilistically (with non-deterministic gates), and it is then consumed with measurements specific to the desired algorithm. The MBQC method may be the most reasonable path forward for photonic devices [55].

For a more detailed review of photonic devices and their history of development see the review [36].

2.3 Trapped ions

Trapped ions have many features which make them an appealing choice for a quantum computing platform, such as the fact that each ion of the same species is naturally identical, in contrast to the manufactured approaches such as superconducting qubits. Ions are trapped with electric fields inside an utlra-high vacuum environment and so they can be well isolated from sources of noise. The laser cooling of ions was first demonstrated in 1978 [56] which was soon followed by the trapping of a single ion [57]. Examples of now commonly used ions include $^{43}\mathrm{Ca}^{+}$, $^{171}\mathrm{Yb}^{+}$, $^{137}\mathrm{Ba}^{+}$, and $^{25}\mathrm{Mg}^{+}$. The first proposal of a quantum computer consisting of trapped ions was in 1995 by Cirac and Zoller [58]. Multiple ions can exist within a single potential well and individual addressability can be accomplished with lasers. Entanglement can be enabled by mapping the logical state onto the motional mode that the ions share in the harmonic potential. Following the original proposal, an experimental realisation of the CNOT was demonstrated [59], but the gate scheme required the ions to be in the motional ground state and the cooling required for this is experimentally difficult. A new gate scheme was proposed by Mølmer and Sørensen [60] which requires considerably less cooling for high fidelity gates.

It is not possible to create a static electric field which can trap an ion in three dimensions, which is a result of Gauss' law which states the divergence of an electric field in free space must be zero. There are two common trapping schemes that get around this issue, the Penning trap uses both a static electric field and a magnetic field, and the Paul trap uses a time-dependent electric field [61]. In the case of the Paul trap, a saddle shaped potential can be generated which is then oscillated effectively forming a confining bowl potential [61].

There is a great deal of variation between different trapped ion platforms. In this section some of the main differing themes will be covered in addition to how they can impact upon applications and feasibility.

2.3.1 Optical and hyperfine qubits

There are two main approaches of transition type for trapped ion quantum computers, optical and hyperfine. The logical states of optical qubits are energy levels separated by an amount corresponding to a photon in the optical band. The transitions (logical operations) are driven by lasers, and stabilising these lasers to a sufficient line width is challenging. The number of highly stabilized lasers required for a device is linearly proportional the number of qubits, and therefore this may limit the ability to scale to large devices. The logical states of a hyperfine qubit corresponds to otherwise degenerate energy levels which undergo small shifts due to an interaction between the nucleus and electron clouds. The transition can be then be driven by a field in the microwave frequency (generally a few GHz), which can be emitted across a wide area which potentially makes scaling to larger devices easier. Hyperfine qubits benefit from extremely long lifetimes and high fidelity single qubit gates. The trapped ion two qubit gate requires the qubit internal state to be coupled to the shared motional mode of the ion string in the potential well, and the strength of this coupling is characterized by the Lamb-Dicke parameter. The microwave frequencies associated with the hyperfine qubits results in a Lamb-Dicke paramater which is orders of magnitude lower than for optical qubits which would result in the necessary gate times exceeding that of the coherence time. This issue with the hyperfine qubits can be alleviated by making use of a two photon Raman transition which necessitates two laser beams. High two qubit gate fidelities have been achieved using this scheme [62], but the requirement of lasers may make scaling to large devices more challenging. Another method of using the hyperfine qubits was proposed in 2001 which relies only on the microwave field to drive the transition [63]. The scheme is enabled by making use of a static magnetic field gradient which creates a greater coupling between the qubit states and the associated motional states and has been experimentally demonstrated [64, 65, 66, 67].

2.3.2 Laser based gates and laser free gates

Laser based entangling gates can generally be performed more quickly than laser free gates and historically have achieved higher two qubit gate fidelities. However, an architecture with access to logical gates that do not require lasers may be easier to scale to large device sizes. Laser free gates have been demonstrated with a time cost of 3400 μs [68], 3250 μs (99.7% fidelity) [65] and 2938 μs [69] and laser based gates have been demonstrated in 30 μs [70] and 115 μs (99.9% fidelity) [71]. A two qubit gate has been enabled with lasers as fast as 1.6 μs with a fidelity of 99.8% [72]. A recent work has demonstrated laser free gates that reach a comparable fidelity to the laser based approaches with a time cost of 740 μs [73]. The speed of the laser free gates is dependent on the strength of the magnetic field gradient, and in the Sussex blueprint for a trapped ion architecture the two qubit gate is estimated to require only 10 μs [74].

2.3.3 Scaling beyond a single ion string

Arbitrary connectivity can be accomplished between a string of ions in a single trap with particular gate schemes but this design has a practical upper limit to the size of the device (approximately 50-100). As the ion count within a single trap increases, the inter-ion spacing reduces which increases the difficulty of targeting particular qubits with laser controllers. An additional difficulty is related to implementations of the two qubit gate, where the coupling strength between a pair of ions scales with distance, d, as $1/d^{\alpha}$, where α ranges from 1 to 3 [75, 76, 77], which results in the required time to perform gates increasing with larger chains. Finally, the density of the motional modes increases with chain length which increases the chance of cross-talk for the two qubit gate thereby lowering fidelity. This is because the motional modes of the chain are used within the two qubit interaction.

Beyond increasing the size of a single trap, there are alternative methods to scale. The Quantum Charge Coupled Device (QCCD) was proposed by Wineland et al [78] and expanded by Kielpinski et al [79]; the scheme involves 2D arrays of linear traps connected by junctions in which the ions can move around. In this design the ions are physically transported (shuttled) between specialized regions for storage and logical operations. A device can have numerous traps dedicated to logical operations (gate-zones) which avoids the downsides of performing logical operations on large ion strings. The ion traps are produced from a silicon wafer, with manufacturing difficulties setting an upper limit on their possible size. Thus, to continue to scale it is necessary to connect multiple modules of finite size.

There are two primary methods of connecting trapped ion modules, the first which we will discuss is the use of photonic interconnects which was proposed by Monroe et al [80]. The photonic interconnects can be used to connect modules which are in separate vacuum chambers. These optical connections are probabilistic but heralded which contributes to defining the connection rate. The protocol relies on entangling the ion with a photon which can be transported via a fiber. Two photons from separate modules interfere at a 50/50 beam splitter which effectuates entanglement between the respective ions. The ions typically chosen for quantum computing do not have favourable wavelengths for transmission along the fiber. An additional ion species can be introduced to improve transmission, and this would necessitate further interactions between the data ions and transmitting ions [81]. In 2020 Stephenson et al generated Bell pairs between modules at a rate of 182Hz with a fidelity of 94% [82]. Entanglement distillation can be used to increase the effective fidelity of this process which would reduce the rate of interaction, and it essentially involves combining many poor quality states into fewer higher quality states. The process of distillation is itself probabilistic and hence the required number of rounds is not always the same and this can cause further slow-down [83]. The rate of 182Hz is of a comparable order to the rate of microwave based gates, but considerably slower than state-of-the-art laser based local two qubit interactions.

The alternative approach was proposed by our group and outlined in Lekitsch et al [74]; it involves constructing a larger device by aligning modules together within a single vacuum system. The edges of each module are carefully designed so that the modules can be brought together in such a way that the electric field lines connect, thereby allowing ions to be shuttled across the gap.

2.3.4 Shuttling

Within the QCCD design, ion traps are segmented by individually controllable electrodes and the ions can be moved by changing the applied electric field. In 2012 Walther et al demonstrated fast ion shuttling with very low levels of motional heating [84] and in 2018 high state fidelity shuttling [85] was demonstrated where a distance of $280\mu m$ was traversed in $12.8\mu s$ with a state fidelity of 99.9994%. The associated state fidelity of shuttling is expected to be very high relative to two qubit gate fidelities. This implies that distant connectivity can be enabled efficiently relative to architectures which require two qubit logical operations to enable beyond nearest neighbour connectivity, such as superconducting devices. The cost of connectivity is compared between these two architecture types in chapter 4. Junctions are necessary to construct a connected trapping array, and shuttling across junctions is generally a more difficult task than linear shuttling. The first successful junction shuttling was demonstrated in 2006 by Hensinger et al through a T shaped device [86]. It has since been demonstrated in both Y [87] and X [88, 89] junctions. Separation and combination are necessary operations for the QCCD scheme, where ions are individuated away from a string, or merged into a string within a single potential well respectively. Ion separation and combination have been experimentally demonstrated in a fast manner and with low motional heating [90]. Ion string rearrangement can be accomplished with positional swaps which have been demonstrated within $25\mu s$ and fidelity of 99.8% [91]. The degree to which swaps will be favorable to utilize will depend on the underlying design of the QCCD architecture. If the ion string is within a gate zone then logical swap operations can also be used for reordering but the associated cost may be higher than positional swaps. The effectiveness of ion routing algorithms both with and without positional swaps is compared in chapter 3.

2.3.5 Blueprint for a microwave trapped-ion quantum computer

This section concentrates on the modular design of our research group [74] which is the primary focus of chapter 3 and 4. As discussed in the previous section, the design uses QCCD modules which consist of a 2D array built up by an iterated X-Junction (shown in figure 2.2). The limitation of the maximum micro-fabrication size of modules is overcome by carefully aligning modules with piezoelectric actuators so that shuttling across modules is possible. The primary consideration behind this design was scalability, with an aim to be able to one day reach the very large number of physical qubits necessary for fault tolerant computation (which with the surface code may be in excess of 1 million qubits).

Under each gate zone is a high-current carrying wire which generates a local magnetic field gradient enabling laser free gates. The static magnetic field and microwave based scheme has been demonstrated experimentally by the group [64]. The magnetic field subjects the ion ($^{171}Yb^{+}$) to a Zeeman splitting within which a qubit can be defined, and the degree of the splitting is dependent on the magnitude of the field. Therefore in this scheme the magnetic field gradient enables one to choose the microwave transition frequency by positioning it within specific locations in the gatezone. The microwave fields can be broadcast over a large 2D array which compares favourably (in scalability terms) to the alternative of requiring precisely aligned lasers for each gate zone.

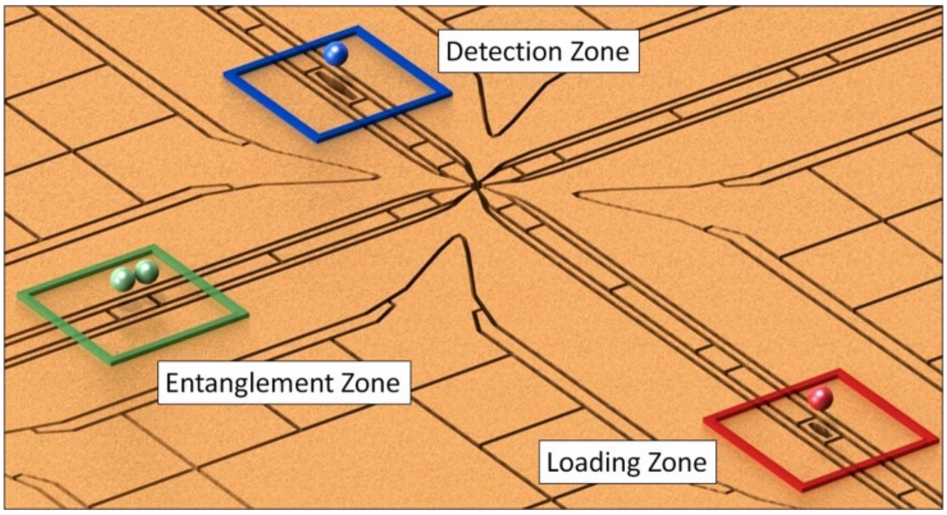


Figure 2.2: A single X-Junction within the QCCD array presented by Lekitsch et al [74]. Each X-Junction contains 3 specialized regions which ions can be transported (shuttled) between. The gate zone in which either one or two qubit logical operations can be performed, the readout zone for measurement, and the loading zone where new ions enter the device (for initialization or to replace a lost ion).

In the NISQ regime global range interactions can be mediated by shuttling operations. Shuttling cannot be performed concurrently with gates and so a near term algorithm would be decomposed into an alternating sequence of shuttling and gates. The ions experience a certain degree of motional heating while in the trap. Shuttling process fidelities reduce with total accumulated heating, as can the quality of two qubit interactions. Cooling techniques which are routinely used for ion initialization, such as Doppler cooling, and sideband cooling, remove quantum information and so they cannot be used mid-circuit. Sympathetic cooling is a technique that preserves quantum information; an alternative ion species is introduced which does not store quantum information and can be directly cooled. The cooling can then be passed onto the data ions when they are stored in the same potential well.

2.4 NISQ algorithms and applications

State of the art quantum computers have around 10-100 noisy qubits, where noisy implies imperfect logical operations and coherence times which limit the achievable circuit depth. A fault tolerant quantum computer with physical qubit numbers in excess of a million would be able to run algorithms with proven exponential or near exponential speedups, but the realization of such a device may be 5-10 years away. Researchers, motivated to make use of the hardware that is currently available, have developed algorithms which are designed specifically with noisy qubits in mind. The most common and general form of a near term algorithm are variational in nature, meaning the parameters associated with a quantum circuit are updated over the course of the algorithm. These Variational Quantum Algorithms (VQA) are sometimes called hybrid algorithms, because they also make use of a classical computer. The algorithm consists of a parameterized low depth circuit (low enough for noisy qubits), the results of which are fed into a classical optimizer which updates the relevant parameters. This process is then iterated to converge towards a desired solution. The workflow for a VQA can be broken into four steps [92]:

1. The objective function encodes the problem to be solved
2. The parameterized quantum circuit is run, with parameters which are later tuned to minimize the objective function
3. A measurement scheme computes expectation values used to evaluate the objective function
4. The classical optimizer proposes new parameters to minimize the objective function

There are many possible methods that could be used for the classical optimizer, but the most generally considered is gradient descent where one moves in the opposite direction to the steepest gradient in search of the global minimum. Variational quantum algorithms can suffer from barren plateaus, i.e. a vanishing gradient and this problem is more likely to arise as more qubits are used [93]. Wang et al found that noise induced barren plateaus are guaranteed to occur given a sufficiently high circuit depth, and importance is placed on identifying viable low depth circuit ansatzes (aka hardware inspired) for VQAs [93].

2.4.1 Variational Quantum Eigensolver

The Variational Quantum Eigensolver (VQE) [94, 95, 96] is a type of VQA used in the context of quantum chemistry, such as estimating the ground state energy of a molecule. In general the objective function is the expectation value of the energy of the Hamiltonian, which is guaranteed by the Rayleigh-Ritz variational principle to be upper-bounded by the true ground state energy [94, 95]. The VQE algorithm requires $\mathcal{O}(1/\epsilon^2)$ iterations of the $\mathcal{O}(1)$ depth paramaterized circuit for an accuracy ϵ which is in contrast to the exponential scaling of exact classical methods and the $\mathcal{O}(1/\epsilon)$ circuit depth for the fault tolerant quantum phase estimation algorithm. The accelerated VQE algorithm interpolates between the near-term and fault tolerant runtime scaling by utilizing the maximum achievable circuit depth [97]. The quality of the initial state starting guess (ansatz) will determine the rate of convergence; in quantum chemistry a usual choice is the Hartree-Fock approximation [98]. The capabilities of the hardware often limits the available ansatz either by the achievable circuit depth or by the cost associated with connectivity [99]. In superconducting architectures, distant connectivity between qubits is very costly relative to nearest neighbour interactions, which leads to a balancing act between the benefits of a higher quality ansatz and the feasibility of its implementation on noisy hardware. The VQE algorithm has been demonstrated for small molecules on various architectures [100, 101] but serious development will be required on both the algorithmic and hardware capability side for hopes of a quantum advantage.

2.4.2 Quantum Approximate Optimization Algorithm

The Quantum Approximate Optimization Algorithm (QAOA), another VQA algorithm, potentially provides approximate solutions to combinatorial optimization problems in polynomial time. The QAOA was first constructed and implemented by Farhi et al and used for MaxCut problems [102], which was one of the first problems ever found to be NP-Complete. Following the initial work of Farhi, QAOA was used on a similar problem type (MAX-3XOR) and was found to outperform the state of the art classical alternative [102]. The result motivated an improvement to the classical algorithm which enabled it to outperform the QAOA approach. This highlights the fact that the goalposts of quantum advantage are non-static; classical approaches can often benefit from quantum inspiration [103]. Methods now exist for extending the applicability of the QAOA algorithm to a wider class of problems, such as graph colouring, and the travelling salesman problem [104]. As with all of the variational type algorithms it is challenging to concretely determine their effective speed up and therefore the size at which they can outcompete classical alternatives. Problems that can be cast as a Quadratic Unconstrained Binary Optimization (QUBO), which includes a wide range of applications from finance, economics and machine learning, can be addressed with the QAOA algorithm [105]. A recent work estimated that several hundreds of qubits will be necessary for a speed up on representative combinatorial problems [106].

2.5 Fault tolerant algorithms and applications

Most of the initial quantum algorithm development was designed with a fault tolerant quantum computer in mind which implies there is no limit on the achievable circuit depth.

In this section an overview of some of the most prominent quantum algorithms and subroutines is provided.

2.5.1 Quantum Fourier Transform

The Quantum Fourier Transform (QFT) is the quantum analogue of the discrete Fourier transform. The Fast Fourier transform (FFT) [107] enables a discrete Fourier transform of a vector of size N in $\mathcal{O}(N \log N)$ which is in contrast to the brute force calculation requiring $\mathcal{O}(N^2)$ operations (involving direct matrix multiplication). The scaling improvement of the FFT enabled a wider range of applications such as signal processing and data compression, and furthermore increased the maximum viable problem size. By using a quantum computer and the QFT algorithm, the computational complexity can be reduced to $\mathcal{O}((\log N)^2)$ [108, 109] which is an instance of a near-exponential speedup. Although the QFT can be implemented efficiently on a quantum computer, the final result of the transform is encoded within the probability amplitudes of the basis states which is not easily accessible. To build up an accurate picture of the probability amplitudes the quantum speedup would be lost, and so the QFT cannot be generally used as direct replacement for the classical FFT. Having said that, the QFT is a core subroutine within many quantum algorithms and so it is indeed possible to make use of the exponential speed up in certain situations.

2.5.2 Quantum Phase estimation

The Quantum Phase estimation (QPE) algorithm is a prominent subroutine used in many quantum algorithms including, Shor's algorithm, quantum chemistry algorithms, and algorithms for solving systems of linear equations. Given a unitary operator U, the QPE algorithm estimates the value θ from $U|\psi\rangle = e^{2\pi i\theta}|\psi\rangle$, where ψ is an eigenvector of U and $e^{2\pi i\theta}$ is an eigenvalue with magnitude 1. The standard circuit for the QPE algorithm requires two qubit registers, the first register is prepared in the state $|\psi\rangle$ and the second register will be used to store the phase. The first step involves applying the Hadamard gate on all qubits in the second register which creates a uniform superposition. Next the qubits within the second register are used as controls for successive powers of U operations on the $|\psi\rangle$ register. The combination of the initial Hadamard gates and the following controlled operations result in a phase kickback on the second register, which effectively encodes θ onto the second register in the Fourier basis. Finally the inverse QFT is used to translate the second register back into the computational basis which can be measured. In this scheme each additional ancilla qubit within the second register adds another binary digit of accuracy on the captured phase. There are now numerous variants of the Quantum Phase estimation algorithm, for example, in the iterative phase estimation algorithm [110] the ancillary register is replaced with a single qubit which is measured repetitively throughout the algorithm while maintaining the same precision. Chapter 5 provides more details on the QPE algorithm and some of its variants in the context of fault tolerant quantum chemistry.

2.5.3 Shor's factoring algorithm

The hardness of factoring large numbers is the backbone of many encryption techniques, such as RSA. In 1994 Shor proposed a quantum algorithm for factoring primes in polynomial time [111], which is in contrast to the best classical method, the General Number Field Sieve with superpolynomial scaling with the size of the prime, n, $\mathcal{O}(\exp{[(c(\ln n^{\frac{1}{3}}(\ln \ln n)^{\frac{2}{3}}])}$ [112]. A key insight used in Shor's algorithm is to recast the problem of factoring as a period finding problem. In particular, finding the period of the modular exponential function, where the aim is to find the smallest positive integer r such that $a^r - 1$ is a multiple of N, or recast in modular arithmetic, the smallest r such that $a^r = 1 \pmod N$. Shor's algorithm constructs a modular multiplication function $x \rightarrow ax \pmod N$, the eigenvalues of which are closely related to the period of a and can be extracted using the quantum phase estimation algorithm. To use Shor's algorithm on problem sizes that are classically difficult will require a circuit depth that greatly exceeds the capabilities of NISQ devices, and so a fault tolerant quantum computer would be required. In the work of Gidney et al, it was estimated that a quantum computer using the surface code with 20 million physical qubits, and a base physical error of 10^{-3} could break RSA encryption in 8 hours [113]. In chapter 6 we perform fault tolerant resource estimation for a related problem, breaking Elliptic Curve encryption, which is used to secure the Bitcoin network.

2.5.4 Grover's search algorithm and amplitude amplification

Grover devised an algorithm for unstructured search problems in 1996 which provides a quadratic speedup as compared to classical alternatives [114]. Grover's algorithm, alongside Shor's, has become one of the most well known quantum algorithms. Amplitude amplification [115, 116, 117] is a generalisation of they key method behind Grover's algorithm, and it has now seen wide ranging use as a subroutine for many quantum algorithms. The original context of the algorithm was to find the unique input to a black box function that produces a specified output value which can be considered similar (with some caveats) to the classical search of an unstructured database for a particular unique item. A classical algorithm for unstructured search of a database of size N requires on average $N/2$ evaluations whereas Grover's algorithm (with the appropriate oracle) requires just $\mathcal{O}(\sqrt{N})$ evaluations. Grover's algorithm assumes access to a function, U_w, that can be applied in superposition to all possible inputs that only applies a negative phase to the desired state, w, and otherwise applies nothing. The following step is the Grover diffusion operator, which acts to invert the probability amplitude of each basis state around the mean of every probability amplitude. The diffusion operator amplifies the amplitude of the marked state while minimising the amplitudes for all other states, and this combination of operators must be repeated $\mathcal{O}(\sqrt{N})$ times to reach the optimal probability (i.e. 50%) of measuring the correct state.

A wide range of NP-complete problems contain a subroutine of exhaustive search, for which amplitude amplification can provide a quadratic speed up [118]. Symmetric-key cryptography is not vulnerable to Shor's algorithm [119] because it does not contain the same type of exploitable structure as in factoring primes, but amplitude amplification would be able to provide a quadratic speedup [119]. The required circuit depth for such applications is likely to be too large for quantum computers without error correction, however low-depth variational alternatives have been proposed [120]. The time cost per logical operation (at the error corrected level) for a fault tolerant quantum computer will likely be many orders of magnitude larger than classical logical operations [121]. Therefore it will be challenging (require larger problem instances) to achieve a quantum advantage with only a quadratic algorithmic speed up, relative to problems with exponential speed ups.

2.5.5 HHL and solving systems of linear equations

Linear algebra is a core component of many machine learning tasks and the HHL algorithm (named after the authors Harrow, Hassidim and Lloyd) was proposed in 2008 to solve systems of linear equations with an exponential speed up [122] (with some caveats). The development of the HHL algorithm catalyzed the then nascent field of quantum machine learning and numerous extensions have been proposed which apply similar methods to a wider range of machine learning problems, such as classification [123], clustering [124], and pattern finding. The goal of the HHL algorithm is to solve the following equation

$$Ax = b \tag{2.4}$$

for x in $\log n$ steps, where n is the number of unknowns and equations in the system. With a classical computer tackling this problem one would assume that it would scale as at best n^2, corresponding to the number of entries in A that must be evaluated. To achieve this impressive degree of speedup, the HHL algorithm comes with some caveats (see [125] for more details), perhaps the most important of which is the form of the output answer of x. The HHL algorithm does not output $x = (x_1 ..., x_n)$ but rather, a quantum state $|x\rangle$ across $\log_2 n$ qubits where each x_i is encoded into the probability amplitudes of the corresponding quantum basis state, i.e. $|x\rangle = \sum_{i=1}^{n} x_i |i\rangle$. To produce the value of a specific x_i from the state $|x\rangle$ will in general require $\mathcal{O}(n)$ preparations of $|x\rangle$, each involving a fresh iteration of the entire algorithm, and therefore would nullify the exponential speedup. There is still interesting information that can be efficiently extracted from $|x\rangle$, such as the value of the inner product $\langle x|z\rangle$ for some vector z. The aforementioned extensions to the HHL algorithm make functional use of the output state $|x\rangle$.

Another important caveat behind the logarithmic run time scaling of the HHL algorithm is known as the data input problem. The HHL algorithm requires that it is possible to quickly (logarithmic in n) load the vector $b = (b_1, ..., b_n)$ into the quantum computer in the form of a state $|b\rangle$, where again each b_i is encoded as a probability amplitude of the respective basis state. It may be possible to efficiently perform this data input problem when b has a special structure, such as when it is described by an explicit formula. For more general instances of b, QRAM (Quantum Random Access Memory) is the primary proposed solution, but assessing (and improving) its feasibility is still an active area of research. In a recent work, Matteo et al estimated that an 8 GB QRAM, requiring fault tolerant quantum error correction, would need quadrillions of qubits to achieve fast (milliseconds) query times [126]. Alternative error correction schemes (in this work the surface code was considered) could offer considerable improvements to the qubit overhead, in addition to the possible improvements to the QRAM procedure.

2.6 Applications in Finance

There are three main categories of computational problems regularly used in the finance industry, optimization, simulation, and machine learning. Optimization problems in finance involve decision making subject to particular constraints, and a canonical example is portfolio optimization, where the goal is to find the best investment strategies with some capital and a set of assets to choose from. Portfolio optimization can be reduced to a Quadratic unconstrained binary optimization (QUBO) problem and the feasibility of the

QAOA algorithm has been investigated [105, 127]. A core component of portfolio optimization is linear algebra, and so the HHL algorithm could potentially provide a substantial speed up. The usual caveats to the HHL algorithm still apply here which we covered in the previous section and so it is unclear whether an exponential speed up would be achievable. Rigorous performance guarantees of a polynomial speedup have been derived for convex portfolio optimization [128] but again, the required effective error rate will necessitate quantum error correction.

Simulation problems deal with predicting potential outcomes and typical examples include derivatives pricing and risk management. Monte Carlo methods are often utilized for these problems which are based on randomized sampling. For example, one starts with a stochastic model of the market and estimates the desired quantity by taking an empirical average over many random starting seeds. It requires $\mathcal{O}(1/\epsilon^2)$ samples to achieve an accuracy, ϵ with classical methods and it was shown by Montanaro that there is a quantum algorithm which instead requires $\mathcal{O}(1/\epsilon)$, leading to a quadratic speed up [129]. The quantum algorithm is based on amplitude amplification [117], which is a generalisation of Grover's algorithm [114] for searching an unstructured database. The required circuit depth of these methods is greatly in excess of what will be possible with noisy (not error corrected) qubits, suggesting that its full realisation will require a fault tolerant device with physical qubits numbers potentially greater than a million. A method of interpolating between the speedup of the classical and full depth quantum algorithms while lowering the required circuit depth has been proposed [130]. This new method opens up the possibility of speed ups for Monte Carlo on NISQ devices because of the tunable circuit depth. In practice one must consider more than just the operational speed up, in this case the number of required samples, because the rate of logical operations can vary greatly between classical and quantum approaches. Classical computers operate at roughly 1GHz which is approximately a factor $20\times$ faster than superconducting devices, which can perform 2-qubit gates in $\sim 20ns$. The quality of the qubits and the cost for distant connectivity will determine the maximum circuit depth that can be utilized, which in turn determines the operational speedup. A fault tolerant quantum computer could reap the full quadratic speedup but the overheads associated with quantum error correction further reduce the effective clock rate.

Machine learning based problems in finance include predicting future events from historical data, pattern detection and classification of samples into categories. Fraud detection is one of the canonical examples and variational quantum approaches have been investigated [131]. Machine learning problems often reduce to solving systems of linear equations for which we again come back to the HHL algorithm and its potential for a substantial speed up, albeit in the fault tolerant regime.

Chapter 3

Trapped Ion Connectivity and Routing

The following two chapters follow the paper titled “Efficient Qubit Routing for a Globally Connected Trapped Ion Quantum computer” [11] which was published in Advanced Quantum Technologies in August 2020. The primary aim of this work was to find an effective routing algorithm to enable global connectivity between ions within an arbitrary sized device following the framework outlined by Lekitsch et al [74] for a scalable trapped-ion quantum computer. This was a necessary component of the second aim, which we cover in chapter 4, to estimate the quantum computational power of devices as a function of experimental parameters. In this chapter the focus is the connectivity problem itself, the developed routing algorithm, and the characterisation of our solution.

3.1 Global connectivity

Connectivity here refers to enabling interactions between the qubits of a given device, and the method by which this is accomplished varies a lot between architecture types. In the shuttling based trapped ion design which we focus on in this chapter, connectivity between distant qubits involves physically shuttling the respective ions (qubits) into the same region of space for local logical operations. For contrast, a superconducting device has stationary qubits which can only interact with their nearest neighbours, and so distant connectivity must be enabled via sequences of logical swap operations (between the physical qubits). Global connectivity refers to enabling interactions between all qubits (where the pairing may be chosen at random for bench-marking purposes) and it is a requirement for the metric Quantum Volume [20]. The cost for enabling global connectivity is an important

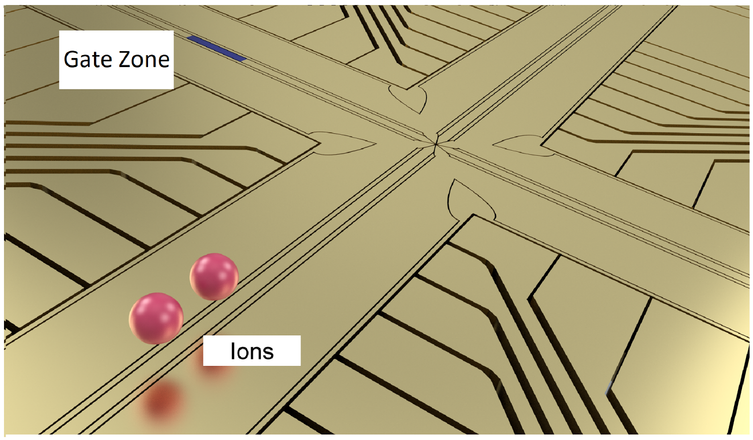


Figure 3.1: A depiction of a single X-Junction which is repeated to form a grid which the ions are restricted to, with zones dedicated to specific tasks.

factor in determining the computational power of near term devices. The most researched error correction technique, the surface code [132, 25, 14] relies only on nearest neighbour interactions. We provided a brief overview of fault tolerant error correction in chapter 2 and in chapter 5 and 6 we show detailed resource estimates using the surface code. Some low-density parity-check codes (LDPC is a family of codes in which the surface code is a member) rely on global connectivity [30] at the physical level but hardware considerations need to be taken into account to determine any practical gain arising from their implementation. The shuttling based architecture we examine here is particularly well suited in enabling mid-range connectivity and we hope to motivate research into error correction codes which would make use of this mid-range connectivity. By mid-range connectivity we intend greater than nearest neighbour but not necessarily global (depending on device size); the larger the device the greater the overhead of movement associated with enabling global connectivity.

In this chapter we present a routing algorithm which efficiently enables global connectivity for a shuttling based trapped ion design [74] and we focus on NISQ-sized devices where it should be feasible to enable global connectivity without error correction. Many of the features within our routing algorithm will be relevant for developing new methods which specialise in enabling nearest-neighbour and mid-range connectivity for error correction.

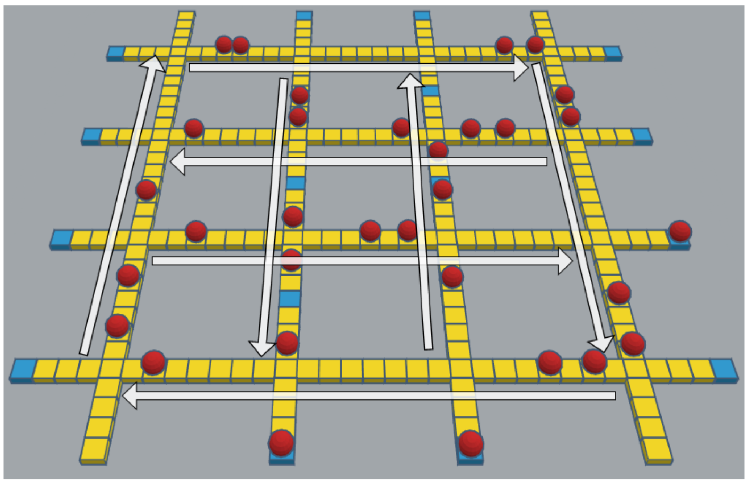


Figure 3.2: 3D representation of a quantum computing device using our proposed routing algorithm, where the yellow grid represents the X-Junctions, which the ions (red spheres) are restricted to move within, and the blue squares represent gate zones. The digitisation of the simulation can be seen with a resolution of 7 positions between adjacent X-Junctions. Arrows represent the lane priority of the routing algorithm. Here there are 4 by 4 X-Junctions (M by M) and 2 ions initialized per X-Junction for a total of 32 (N), where $N = 2M^2$.

3.2 Simulation tool

In the design being investigated here, ions (each encoding a single qubit) are restricted to a square grid (see Figure 3.2) which consists of an array of repeated X-Junctions (see Figure 3.1), each containing a single gate zone. Ions must first be shuttled (physically moved) into the gate zones for gates to be performed. The X-Junctions have a defined spatial resolution, which arises from the fixed number of electrodes on each arm but ions may be moved continuously, i.e. an ion may be shuttled across multiple electrodes in a smooth manner as opposed to only moving one step at a time. The gate zones enable both single and two qubit gates. A hardware-agnostic quantum algorithm must first be decomposed into the native gate set of the device, and this process can be optimised, such as the work by Maslov [133] which focuses on a trapped-ion gate set. A decomposed quantum algorithm is defined by multiple rounds (the circuit depth) of gates. Ideally all the required gates of an individual round will be applied in parallel, however the qubit number, gate density of the algorithm, and the number of gate zones will dictate the gate round overhead (i.e. how many rounds of gate applications are required for a depth-1 circuit). In this architecture, each gate round is further broken into two parts, a

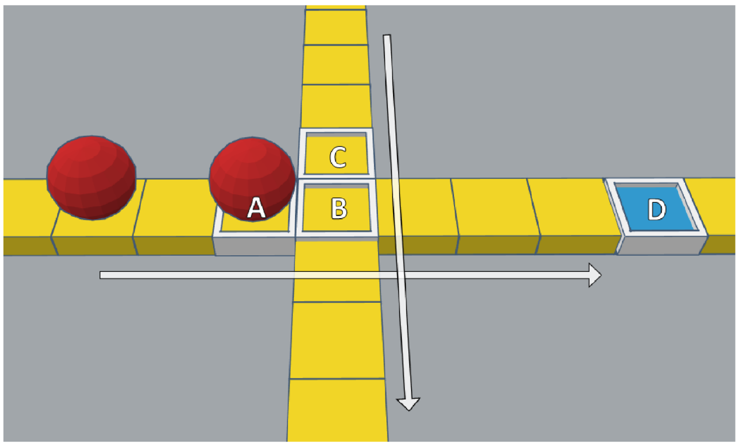


Figure 3.3: A close up of an X-Junction from figure 3.2. The routing logic used to decongest X-Junction centres involves occasionally ignoring the lane priority shown by white arrows. Ions assigned to interior gate zones (blue square labelled D) have the closest X-Junction centre (labelled B) as their destination (one space off the centre because it is an area of lower trap stability (labelled A and C)). The ion in square A has been assigned to the local gate zone and it will travel back and forth between positions A and C directly, by ignoring the lane priority, to decongest for ions still travelling to their destination.

routing sequence, where ions are shuttled into gate zones, which is then followed by the application of gates. We use the terminology "shuttling" to refer to the act of moving ions in the device, and "routing" to refer to the higher order logic of the reconfiguration. In this design, gates cannot be applied concurrently with shuttling. When the required number of gates in an individual round exceeds the number of available gate zones it is necessary to have multiple rounds of shuttling and gates, e.g. a gate round overhead of 2 would imply the need for: shuttle, apply gates, shuttle, apply gates. The shuttling round, which enables the connectivity, is our focus for now. When designing the routing algorithm, we optimized for the total time taken to enable global connectivity.

To investigate how to enable connectivity in this quantum computing design, we have created a simulation tool [134] which represents the devices as a square grid consisting of iterated X-Junctions (see Figure 3.2). The simulation tool was developed using python and visualized using the pygame package (see figure 3.4). The visualisation was particularly useful during the developmental stage of the routing algorithm. The simulation is digitized to a variable resolution, where each position may either be empty or contain an ion (or ion pair). The ions are distributed evenly across the grid near the centre of each X-Junction and a quantum circuit (list of required two-qubit gates, i.e., ions that must be connected) defines the routing problem. Ions which are assigned to the same gate zone

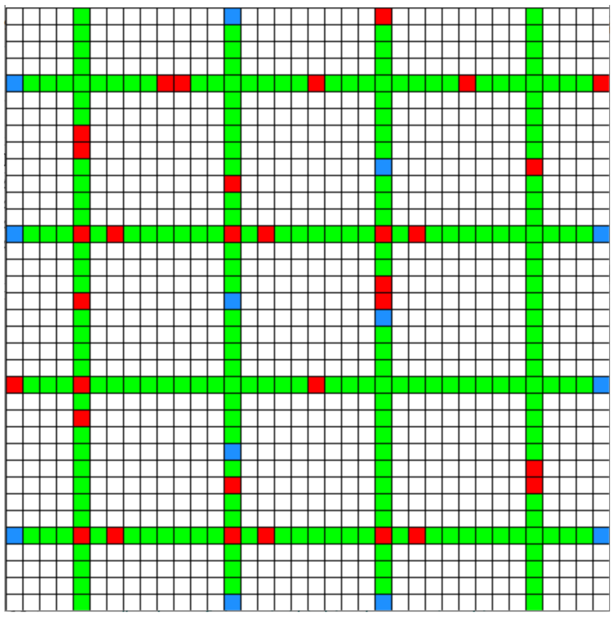


Figure 3.4: The direct visualisation output of the routing simulation through the pygame module. Here the X-Junction is shown as the background green squares, the gate-zones are shown as the blue squares, and ions as the red squares.

are able to combine as a pair. Naive routing algorithms would not converge on a solution (i.e. all desired interactions enabled) as ions with opposite travelling directions meet and cause permanent blockages. Positional swaps between ions have been demonstrated experimentally [135] and their usage would simplify the required routing algorithm. Here we present a solution that does not use positional swaps, and in section 3 we compare the effectiveness of routing both with and without swap operations. Positional swaps are distinct to that of logical swaps (both here between physical qubits), for example a trapped ion device may have access to both operations (but one may have a preferable fidelity or time-cost), in contrast, superconducting devices only have access to logical swaps (which are performed using sequences of the native gate set). When bench-marking the routing algorithms, a randomly generated and globally connected, quantum circuit was used.

3.3 Routing Algorithm logic

In order to assign ion (qubit) pairs to gate zones, we employ a greedy approach (i.e. the method begins optimal but does not necessarily end optimal), assigning each pair to the nearest available gate zone (i.e. minimum combined distance of travel for the two ions), and addressing the pairs in an arbitrary order. This greedy approach is sufficient for a proof of principle using this prototype ion-routing algorithm, however we note that it may not yield the optimum gate-zone designations overall. To this end, a more sophisticated optimisation may be considered in future work, but we note that such combinatorial optimizations are generally hard problems themselves.

Alternating lane priority

At each time step in the simulation, each ion is evaluated and moved sequentially according to the routing algorithm, which involves assessing its location, local environment and destination. The routing algorithm we have developed assigns alternating direction priorities to each lane of the square grid. The top-most horizontal lane is a right-only lane, the lane below it is left-only, and so on, and this also applies to vertical lanes (see Figure 3.2). We ensure that the outer perimeter of the device is a clockwise loop regardless of the number of lanes, so that all gate zones can be reached, which means that odd size devices, e.g. one which consists of 3 by 3 X-Junctions, will not have fully alternating lane directions and instead will have right, left, left, and up, down, down.

Perimeter considerations

We define a square grid device formed from M by M X-Junctions to be of device size M. We preferentially position gate zones on the exterior of the device where possible (on the outer arms of the perimeter X-junctions). Exterior gate zones are more favourable for routing as ions waiting for their gate do not block the movement of other ions. For square devices the number of interior gate zones scales with device size as $(M-2)^2$ and the number of exterior gate zones scale as $4M-4$, which results in a cross over point at device size 7 (98 qubits at 2 per X-Junction).

Decision process

The centres of X-Junctions are decision points, where an ion will follow the lane priority towards its destination. Ions can enter the outer arms into the exterior gate zones only when it allows them to reach their assigned destination. Ions which are not destined to a gate zone during a given shuttling round have their destination set to their current location, and therefore only move to decongest. During development of the routing algorithm, a major bottleneck identified was congestion at interior gate zones. Devices larger than 2 by 2 have interior gate zones, and the ions waiting there can cause permanent blockages or unnecessary movement depending on how they are handled. To remedy this problem an additional feature was included, in which ions assigned to interior gate zones wait at the closest available X-Junction centre, where they are able to decongest efficiently by temporarily ignoring the lane priority (see Figure 3.3). Depending on the ion density, the gate round overhead may be greater than 1, to illustrate an example, if there are twice as many ion-pairs as gate zones, a depth-1 algorithm is broken into two rounds of shuttling and gates. In the first round of shuttling, half of the ions have a gate-destination while the other half have no destination and only move to decongest. For the following round their roles switch over so that now the ions which have already performed their interaction will only move to decongest and the remaining ions proceed towards their assigned gate zone. The movements available to each ion are dependent on multiple assigned parameters. The following binary questions determine these parameters: Does this ion have a destination for this round of gates? Is this a single ion or pair? Is this a waiting ion assigned to interior X-Junctions? The valid moves are then determined by using these parameters in combination with the location and local environment of the ion. At any particular time step an ion may have multiple valid moves available to it, hence there is a hierarchical list of decisions which are explained in the following pseudo code.

Pseudo code

```
Initialize device with size, resolution, number of ions
Pair all ions randomly
Calculate the number_of_rounds required (from number of ions and gate zones)
Until all pairs have visited a gate-zone:
    Loop over number_of_rounds:
        Loop over pairs:
            Assign destination of each pair in this round to closest
            combined-distance available gate zone
        Loop over all ions Until round complete:
            if current_ion is a single move from it's assigned pair:
                Combine as a pair
            else-if current_ion has a destination and is not there yet:
                suggested_move = Identify single step move that is
                unoccupied and minimises the distance to
                the destination and follows the lane-priority (see fig 3.2)
                if suggested_move has a valid result:
                    Move
                else:
                    Be stationary this round
            else-if current_ion has a destination and is already there and
```

```
the assigned destination is an interior gate zone:
    if current_ion is currently blocking another ion:
        Perform lane-priority ignoring move to the other arm
        of the X-junction (see fig 3.3)
    else:
    Be stationary this round
else-if current_ion has a destination and is already there
and the assigned destination is an exterior gate zone:
    Be stationary this round
else-if current_ion has no destination:
    if current_ion is currently blocking another ion:
        Move following lane priority
    else:
        Be stationary this round
```

3.4 Characterising the routing algorithm

In this section we characterize the performance and flexibility of our routing algorithm, which we refer to as lane priority routing, within the framework of our abstract simulation tool. In chapter 4 we will introduce more practical considerations, allowing us to quantify the expected fidelity associated with enabling connectivity.

3.4.1 Time taken to route

Randomly generated depth 1 circuits on N qubits consisting of $N/2$ two qubit gates were iterated sufficiently to represent the requirement of global connectivity (we investigate the choice of iteration number in figure 3.12). After each iteration we count the total number of time steps (τ) required to enable the global connectivity, which can be converted into a total time (in seconds (s)) by considering the estimated speed at which one can shuttle between adjacent X-Junctions. The number of time steps (τ) is scaled to the resolution of the model so that $\tau = 1$ can be considered equivalent to the time it takes to shuttle between adjacent X-Junctions. We include experimentally achieved shuttling speeds later in this chapter. As a brief aside, it is important to note that increasing the speed of an individual shuttling operation may not always lead to an increase in the final fidelity, as the quality of the shuttling operation may impact on subsequent operations by factors

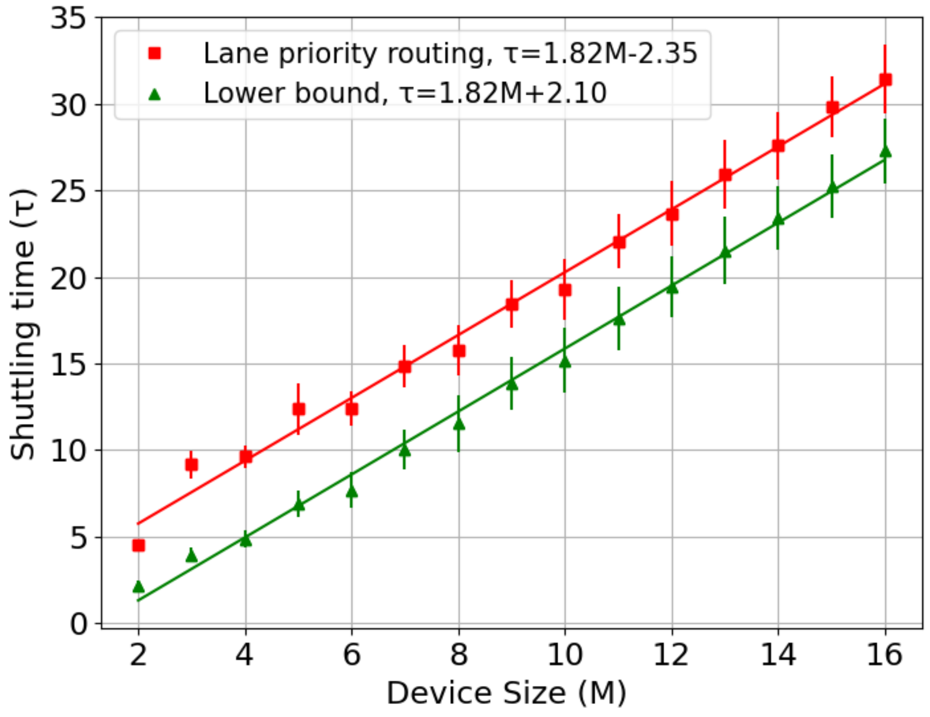


Figure 3.5: Shuttling time, τ (scaled by the resolution of the model so that a time of 1 is equal to the time it takes to shuttle between two adjacent X-Junctions), taken to enable connectivity as a function of device size (defined as a square grid consisting of M by M X-Junctions where M is the device size). There are two qubits initialized per X-Junction (plotted here for a range of 8 to 512 qubits). Red squares: Shuttling time for the routing algorithm. Green triangles: The lower bound (shortest route) shuttling time. The trend lines were generated using linear regression analysis and they both have a gradient of 1.82. Vertical lines represent one standard deviation. The results are the average value over 300 iterations of randomly selected pairings. The iteration number was chosen after investigating the mean and standard deviation convergence rate.

such as accumulated heating or ion loss. At each iteration, a lower bound is calculated for that particular set of pairings, which is equal to the minimum number of time steps that will enable connectivity. To calculate the lower bound it is assumed that, qubits (ions) take the shortest path towards their destination and swap with no time penalty (i.e. the time required for an ion to move one discrete step is independent of whether a swap is performed or not). For a particular iteration, the ion with the greatest distance to travel is identified, and the number of spatial steps between its starting location and its destination is equal to the lower bound.

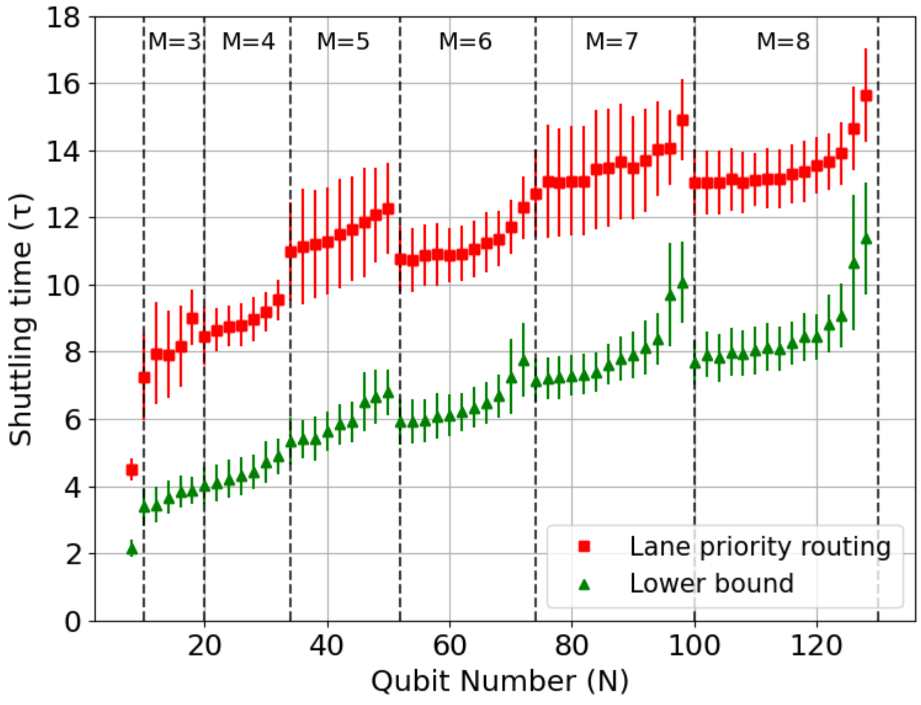


Figure 3.6: Shuttling time as a function of qubit number. Two qubits are sequentially added to a given device and the device size increases when the device can no longer accommodate two qubits per X-Junction. Red squares: Shuttling time for our routing algorithm. Green triangles: The lower bound (shortest route) shuttling time. Vertical lines represent one standard deviation and the dashed lines mark where the device size is increased to accommodate the additional qubits. The results are the average value over 300 iterations of randomly selected pairings.

The average number of time steps (τ) required to enable the global connectivity can then be compared to the lower bound as shown in figure 3.5. These results are for devices with perfect two qubit gate parallelizability, i.e. there are two qubits initialized per X Junction. We conjecture that the total shuttling time would at best scale linearly with device size, M, because randomly selected distances in a square scales linearly with the length of the square. Our lane priority routing and the lower bound scale linearly with the device size and with an equal gradient of 1.82. There is a constant overhead (independent of device size) associated with the lane priority routing. The minimal distance path length should have the same scaling with device size when following the lane priority as compared to taking a direct route but certain journeys will result in an overhead of at most 4τ to conform to the lane movement directions (ignoring congestion). The distance between two particular X-Junction centers is the Manhattan distance (distance in the first dimension plus the distance in the second dimension, which is in contrast to a straight line in Euclidean space), and there can be numerous unique paths with equal lengths between two points. Using a simple numerical model for choosing M^2 random positions within a square grid we find that the average maximum Manhattan distance per iteration scales with square size (comparable to device size M) as $1.94M - 2$, which is similar to the lower bound scaling within our simulation tool, and the difference will be due to the process within the simulation tool of choosing the closest available gate-zones for each pair. For these results there are two ions per X-Junction (per gate zone) and so the scaling for total shuttling time of our lane priority routing, τ, as a function of qubits, N, where $N = 2 \times M^2$ is $\tau = 1.3\,(3)\,\sqrt{N} + 2\,(5)$, and the fit and standard error were calculated using linear regression. An oscillating pattern on the lane priority routing is noticeable with its relative magnitude decreasing with device size, which results from even sized devices outperforming odd sized devices. Odd sized devices (for example a device of 3 by 3 X-Junctions) cannot fully realize the alternating lane priority because we ensure that the outer perimeter lane is always a clockwise path.

The routing algorithm can function for a wide range of qubit numbers for a given device size. Figure 3.6 shows the shuttling dependence on qubit number for qubit densities less than or equal to 2 per X-Junction, i.e. with full gate parallelizability. The oscillating pattern resulting from odd and even device sizes is more notable because here we investigate a smaller range of device sizes than in figure 3.5. Shuttling time increases for both the lane priority routing and lower bound as more qubits are added to a device of static size, and peaks at a density of two qubits per X-Junction.

3.4.2 Counting passes through X-Junction centres

The main criteria we optimized for when creating the routing algorithm was the total time. However, to calculate the achievable circuit depth at which a device can run, the total error will not just be a function of the total time, but will also include factors such as gate fidelity and ion loss. Traversing an X-Junction will have a corresponding ion loss rate which may be higher than the loss associated with linear shuttling or idling. In order to quantify the associated error we have used our simulation to count the number of times qubits are expected to move through an X-Junction centre. The implications of these results for achievable depth will be explored in the following chapter. In figure 3.7 the mean number of passes through an X-Junction, X_{count}, is plotted as a function

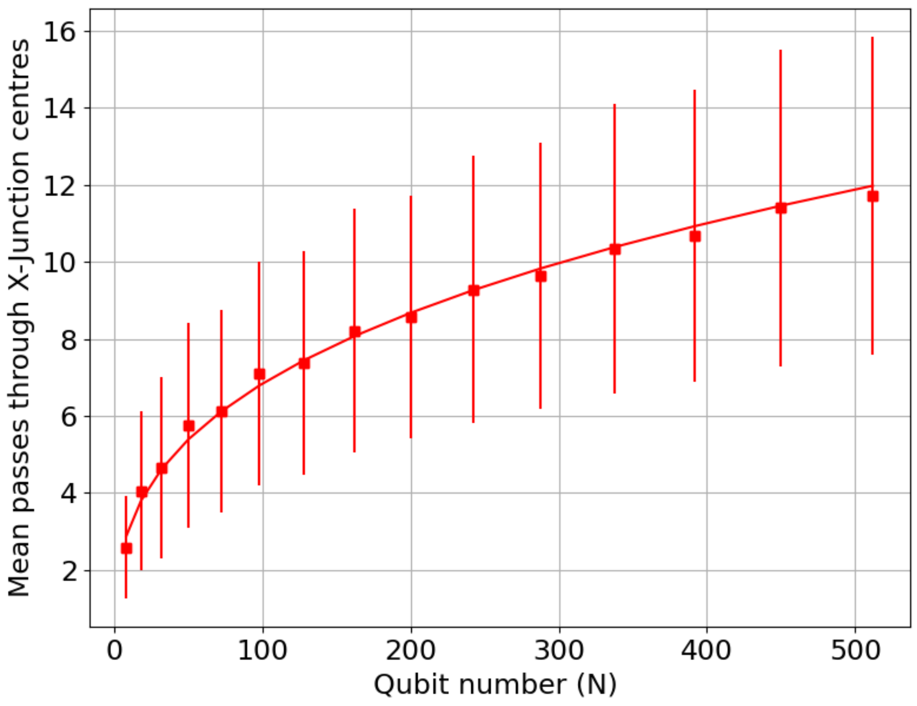


Figure 3.7: The mean number of passes through an X-Junction centre per ion as a function of qubit number for square devices with two qubits per X-Junction. Vertical lines represent a single standard deviation. The results are the average value over 300 iterations of randomly selected pairings.

of qubit number with vertical lines corresponding to a single standard deviation, and the dependence is well described by the following equation, $X_{count} = 0.4\,(1)\,\sqrt{N} + 2\,(2)$. The distribution of passes is investigated in 3.8 for four different device sizes, 4, 6, 8 and 10. The qubits are separated into two data sets, according to whether they are assigned to an interior or exterior gate zone. Across all device sizes investigated the maximum passes did not exceed $4\times$ the stated mean. As expected we can see that ions assigned to interior gate zones pass through X-Junction centres more frequently which is because they must move out of the way of other passing ions (see Figure 3.3). As the device size increases the ratio of interior gate zones to exterior gate zones increases, so it may be worth investigating rectangular shaped devices to minimise the number of interior gate zones (although this would be at the cost of a longer average travel distance). The simulation tool is flexible enough to allow for rectangular shaped devices and a brief investigation was performed into the relevant performance but is not included here. To determine the actual performance benefit it would require introducing the experimentally realistic costs of X-Junction center

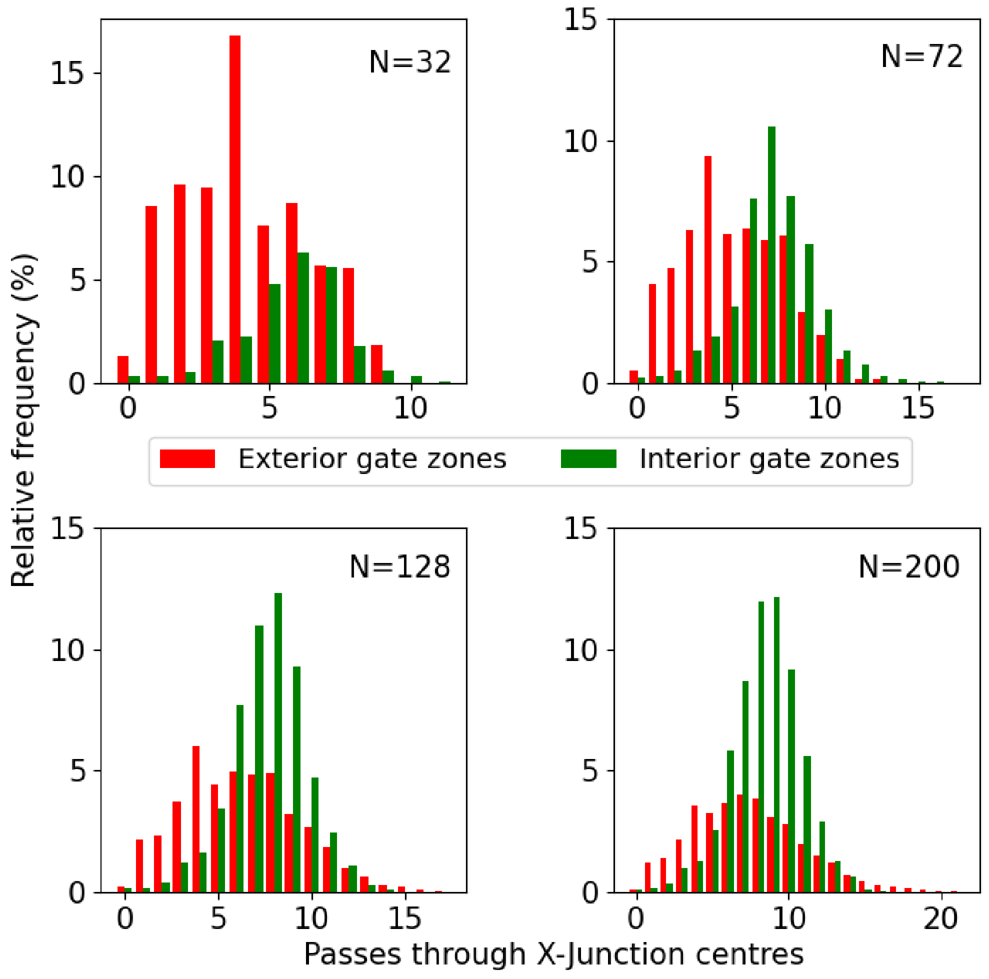


Figure 3.8: The relative frequency distribution of passes through X-Junction centres for four different device sizes, 4x4 (N=32), 6x6 (N=72), 8x8 (N=128), 10x10 (N=200). Red bars: Qubits assigned to exterior gate zones. Green bars: Qubits assigned to interior gate zones. 300 iterations of the globally connected depth-1 algorithm were used to generate a representative sample, and the frequency is scaled accordingly.

crosses relative to total time required.

3.4.3 Increasing ion density

It may be desirable to increase the qubit density beyond 2 per X-Junction despite the potential loss of gate parallelizability as additional X-Junctions are experimentally costly to implement. Figure 3.9 shows the efficiency of the routing protocol for three different qubit densities. We can see that the total shuttling time increases substantially with

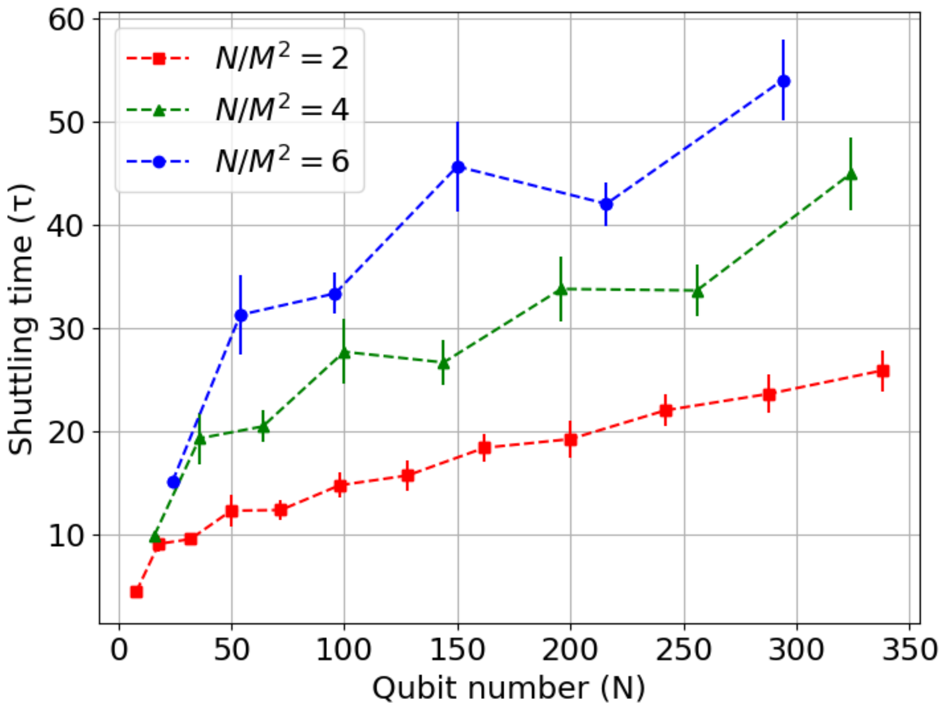


Figure 3.9: Required shuttling time as a function of qubit number for three different qubit densities. Red squares: 2 qubits per X-Junction. Green triangles: 4 qubits per X-Junction. Blue circles: 6 qubits per X-Junction. Vertical lines represent one standard deviation. Gate density is set to 100% meaning that qubit densities higher than 2 per X-Junction are decomposed into multiple rounds of shuttling and gates. The results are the average value over 300 iterations of randomly selected pairings.

increasing ion density. The increase in shuttling time is predominantly attributed to the multiple rounds of shuttling (and gates) which are required for the 100% gate density (where gate density is the percentage of qubits involved in gates per algorithm step (layer)) algorithm which we are assessing against. With a density of four qubits per X-Junction, a 100% two qubit gate density algorithm would be completed by two full rounds of shuttling and gate applications. The oscillating pattern attributed to odd and even devices becomes more apparent with increasing qubit density. The total shuttling time defined here only includes the additional time associated with the multiple rounds of shuttling and does not include the gate time. The overall cost of increasing qubit density will depend on the gate density of the desired algorithm.

We also investigated the effect of increasing qubit density while equally reducing the gate density of the required algorithm as can be seen in figure 3.10. For each device size

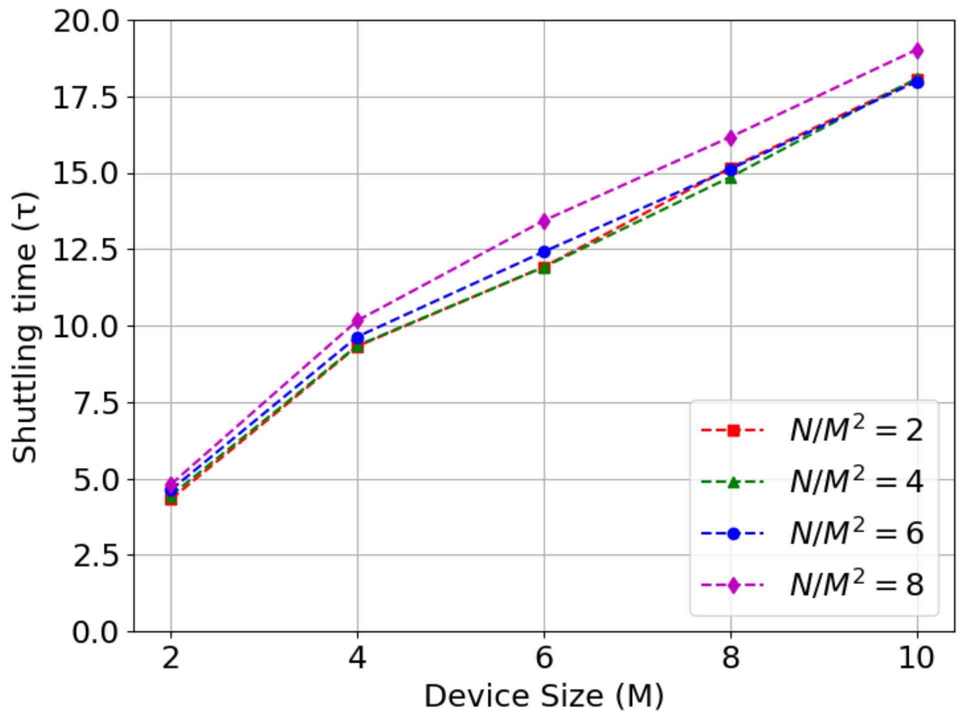


Figure 3.10: Required shuttling time as a function of device size for four different qubit densities for even-sized devices where red squares represent 2 qubits per X-Junction, green triangles 4 qubits per X-Junction, blue circles 6 qubits per X-Junction, and magenta diamonds 8 qubits per X-Junction. In each case the gate density is reduced accordingly so that the algorithm is performed in one round of shuttling. For example with a qubit density of 2 per X-Junction, the gate density is 100%, with 4 per X-Junction, the gate density is 50%, because each X-Junction has one gate zone.

the numbers of qubit pairs required to visit a gate zone was equal to the total number of gate zones, regardless of the total number of ions initialized on the device. The shuttling time for densities 2, 4, and 6 are nearly identical, highlighting the effectiveness of the lane priority routing. A density of 8 sees an approximately 5% increase as compared to a density of 2 (ions per X-Junction). Through comparison of figure 3.9 and 3.10 we can determine that the large differences noted in figure 3.9 are almost entirely due to the additional rounds of shuttling that are required for the 100% gate density algorithm as opposed to the congestion that may result from the increased ion density.

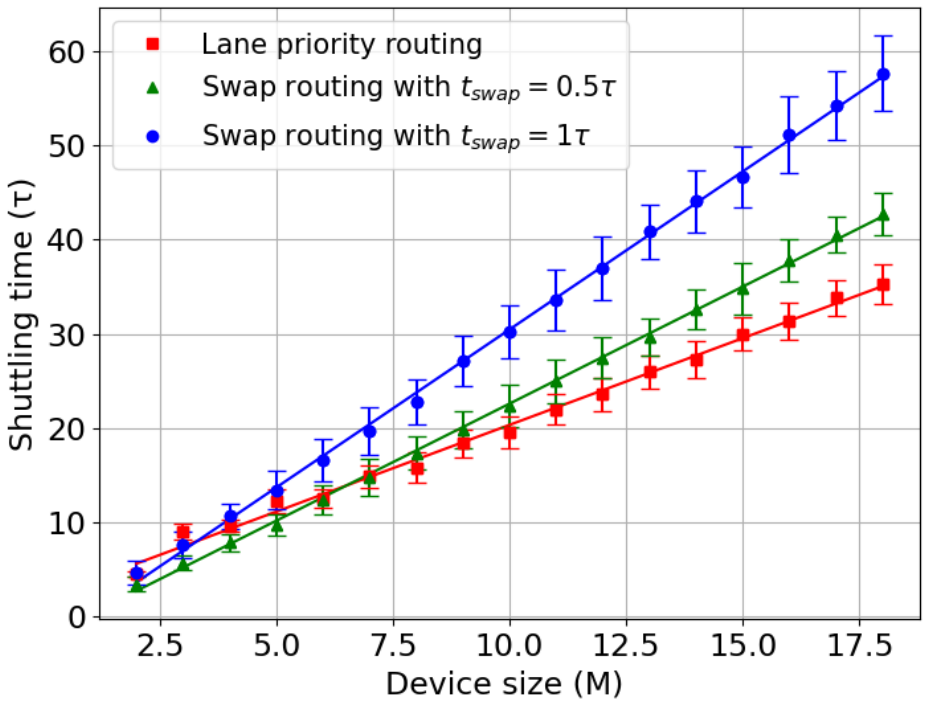


Figure 3.11: Required shuttling time as a function of device size (defined as a square grid consisting of M by M X-Junctions where M is device size), with 2 ions per X-Junction, comparing swap based routing to our lane priority routing algorithm. Red squares: Lane priority routing. Green Triangles: Swap based routing with a swap time penalty equivalent to half the time it takes to shuttle between two adjacent X-Junctions. Blue Circles: Swap based routing with a swap time penalty equivalent to the time it takes to shuttle between two adjacent X-Junctions. Vertical lines represent one standard deviation and trend lines are fit using linear regression.

3.4.4 Positional swaps

We created a second routing algorithm which relies on positional swaps where qubits take the shortest available route (ignoring the previously mentioned lane priority routing) and swap to decongest. We have compared the total shuttling time of the swap routing against the lane priority routing, for two different swap time penalties, shown in figure 3.11. The time penalties were chosen based on early experimental results. H, Kaufmann *et al* demonstrated fast ion swapping of $42\mu s$ at a process fidelity of 99.5% [135]. Van Mourik *et al* demonstrated positional ion swapping in $25\mu s$ with an associated coherence loss of 0.2(2)% [91]. For ion shuttling speed, Walther *et al* demonstrated fast shuttling of cold ions, over a distance of $280\mu m$ in $3.6\mu s$ [84] and P. Kaufmann *et al* demonstrated a state fidelity of 99.9994%, for shuttling over a distance of $280\mu m$ in $12.8\mu s$ [85]. For a wide range of device sizes the lane priority routing outperforms the swap based routing on total time taken, for the swap time penalties used here. The swap routing would be equivalent to the lower bound of figure 3.5 when there is no time penalty on the swap, i.e. when the time cost of a swap is equal to shuttling the same distance. We characterized the average number of swaps, n_{swap}, per qubit for each connectivity run and found that for 18 qubits, the average was 1 swap, and for 50 qubits the average was 1.7 swaps. The dependence was well described by the following equation, $n_{swap} = 0.23(2)\sqrt{N} + 0.1(2)$, where the fit and standard error were calculated using linear regression. The average number of swaps per ion which was required to enable connectivity was found to be only weakly dependent on the swap time cost penalty, therefore doubling the time penalty results in minimal change to the number of swaps. The analysis associated with figure 3.11 suggests that for efficient routing in this 2D trap design, it will not be necessary to perform positional swap operations. However some combination of the lane priority routing and positional swaps may be favorable, depending on the expected costs associated with these operations. In the following chapter we will bring in more practical considerations to define the expected fidelity of a globally connected algorithm using our lane priority routing algorithm.

3.4.5 Justifying iteration number choice

For all of the graphs plotted in this chapter an iteration number of 300 was chosen to average over. In each iteration a new set of random pairings of qubits was chosen. To justify that this iteration number is large enough to well represent the requirement of global connectivity, we plot the dependence of the mean and standard deviation as a function of iteration number. In figure 3.12 A we show the convergence of both for a device size of 8 and in figure 3.12 B we show for a device size of 12 where at each iteration the mean and standard deviation are recalculated. In both scenarios it appears that an iteration number of approximately 200 would be sufficient. The early shot behaviour varies from run to run of a data set such as this due to the inherent randomness, but each converges to the same value for high enough shot counts.

3.5 Concluding remarks

We have created a routing algorithm to enable global connectivity for a shuttling based architecture. The relevance of this work is independent of the specific choice of gate operation, ion species and transition. Finding the optimum instruction set for each individual

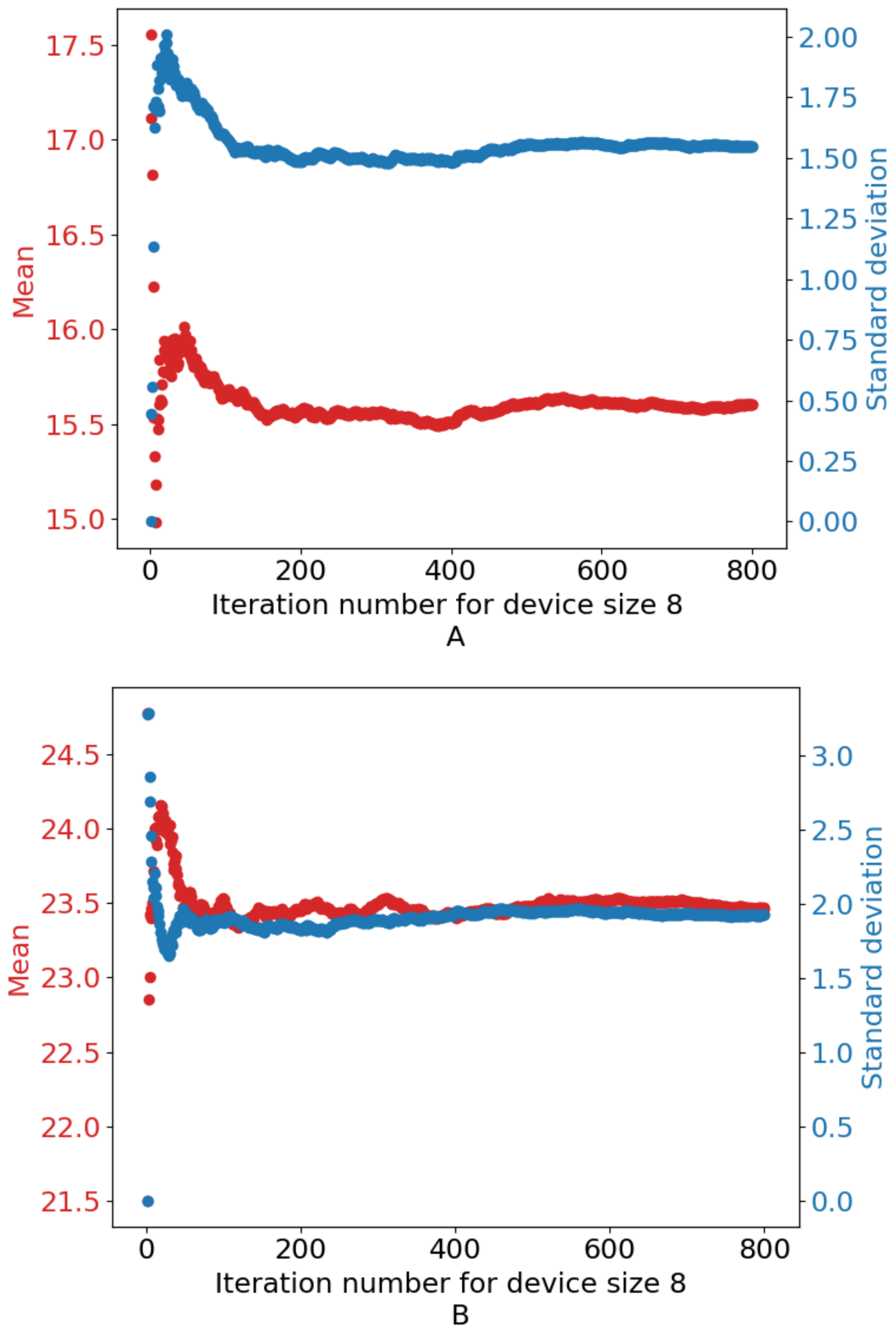


Figure 3.12: The convergence rate of the mean (red) and standard deviation (blue) for shuttling time for a device of size (A) 8 (128 qubits) and size (B) 12 (288 qubits). At each iteration the mean and standard deviation were recalculated.

ion in real time is intractable and so we have solved the problem in a heuristic manner. We have assessed the effectiveness of our routing algorithm by quantifying the total time to enable connectivity, and compared against a strict lower-bound and an alternative positional-swap based routing algorithm. In the following chapter we will introduce practical considerations and produce an error model which enables one to estimate the computational power of near term devices following this shuttling based design.

Chapter 4

Prediction of Computational Power for Near Term Devices

The following chapter continues to follow the paper titled "Efficient Qubit Routing for a Globally Connected Trapped Ion Quantum computer" [11]. The aim of this chapter is to predict the computational power of near term devices as a function of experimental parameters. This analysis can then be used to determine experimental priorities. We develop an error model for the shuttling based trapped ion design and utilize the routing results of the previous chapter to estimate the computational power as defined by the quantum volume. We compare the quantum volume of the shuttling based design to a superconducting architecture as a function of the two qubit gate fidelity.

4.1 Achievable depth and quantum volume

For comparison between near term quantum computers, one must consider more than just the number of qubits. Quantum volume (QV) is a conceived metric for quantum computational power designed to enable sincere comparison between architectures [19, 20]. QV includes factors such as gate fidelity, qubit number, connectivity and the available gate set. Below we present the definition as given by Moll et al [19] (which has some differences to the definition given by Cross et al [20], which is provided later in this section).

$$QV = \max_N \left[\min \left[N, \frac{1}{N \times \epsilon_{eff}(N)} \right] \right]^2, \tag{4.1}$$

for the number of qubits within the device N, and effective error rate ϵ_{eff}, which typically depends on N. QV reflects the limiting factor of the device, which is either the qubit number or the achievable depth D, where

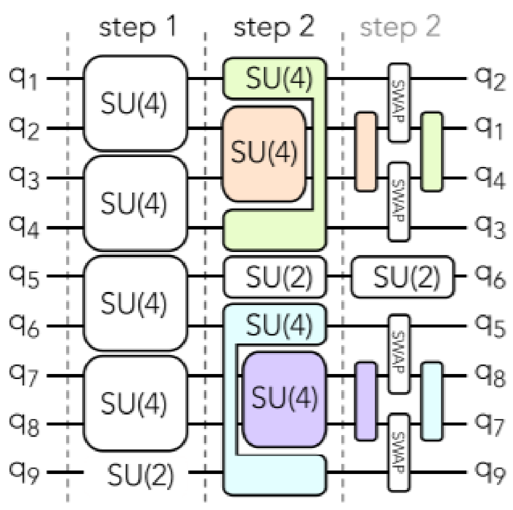


Figure 4.1: An example circuit for calculating quantum volume in an architecture with linear string connectivity. In step 1 the qubit pairing selection conforms to the device's natural connectivity, whereas in the case of step 2, the desired connectivity requires swap gates. Many randomly generated depth-1 circuits should be used and averaged over to calculate the effective error. Image by Moll et al [19]

$$D = 1/(N \times \epsilon_{eff}(N)) \tag{4.2}$$

To compute QV, a randomly generated depth-1 circuit on N qubits with $N/2$ arbitrary (SU(4)) two qubit gates is used. The achievable depth represents the circuit depth at which the device can run before coherence is lost, specifically, the depth at which at least one qubit error is statistically likely. A greater number of qubits requires a lower effective error to reach the same circuit depth because each additional qubit increases the likelihood of at least one error occurring across the device, which is captured by the $1/N$ dependence. The achievable depth is a useful concept which can be used separately from QV to estimate the feasibility of running an algorithm on a NISQ device.

4.1.1 The latest Quantum Volume definition

The latest definition for Quantum Volume was proposed by Cross et al [20], and the metric has received a greater adoption in the community since the first iteration [19]. In this section we will provide an overview of the new definition.

The aim of the metric remains the same, to provide a single value number that represents the computational power of near term devices by considering the qubit number, the qubit quality, and the available connectivity. This aim is different to some other commonly used metrics, which generally specialise in assessing the quality of operations on small numbers of qubits, such as, randomized benchmarking [136], state process tomography, and gateset tomography [137].

The new definition for quantum volume has many similarities to the previous one, and can be seen below.

$$log_2\, QV = \underset{N}{\text{argmax}} \Big[\min \Big[N, D(N) \Big] \Big], \tag{4.3}$$

In the new metric the heavy output generation problem [138] is used to quantify the quality of the implementation of the randomly generated model circuit U, which can then be used to define the achievable depth. The main advantage of the first definition by Moll [19], which directly defines the achievable depth as a function of the effective error and qubit number, is that it enables one to begin estimating the quantum volume without the need for a real device or a a full quantum state simulation. In either situation, a full quantum state simulation should provide greater insights than the estimations generated using effective errors and it will be worthwhile to develop. The routing logic of the previous chapter would then be one layer within this larger simulation. In the following we highlight the method by which the new version of quantum volume is calculated [20].

The perfect output distribution of the circuit U, such as the one shown in figure 4.1, is

$$p_U(x) = |\, \langle x \,|\, U \,|\, 0 \rangle \,|^2 \tag{4.4}$$

where x is the observable bit-string for the N qubit device. The heavy outputs for this distribution are H_U and are defined as those which are greater than the median of the distribution. To calculate the median, the set of output probabilities are sorted in ascending order and then $p_{med} = (p_{2^{(N-1)}} + p_{2^{(N-1)}-1})/2$. The output probabilities of an arbitrary quantum circuit should be exponentially distributed random variables which implies that approximately 85% ($\frac{1+ln(2)}{2}$) of them should exceed the median value [138]. If the device is completely depolarized this percentage will fall to 50%.

The device implementation of U is U', the observed distribution is $q_U(x)$ and the probability of sampling a heavy output is

$$h_U = \sum_{x \in H_U} q_U(x) \tag{4.5}$$

The heavy output probability for the device implementation can then be calculated (by performing numerous iterations of the device implementation) as a function of the depth of the circuit, and then the achievable depth is defined as the largest value such that the probability remains greater than $2/3$. The set H_U must be calculated classically, directly from U which scales exponentially with the depth. The output shot distribution of the noisy circuit (U') are collected and compared to the ideal case to define the heavy output probability. For large enough circuits computing the set H_U classically will not be possible and at this point the metric will need to be redesigned so that the classical simulations are not necessary. QV is now defined to scale exponentially with the (quality) qubit number as opposed to quadratically so as to better represent the relative power of quantum computers to classical computers.

4.1.2 How we use Quantum Volume

In the results presented in this chapter and the associated paper, we estimate the achievable depth directly as defined in equation 4.2, by calculating the effective error as a function of hardware parameters, e.g. gate error, coherence time, time taken to enable connectivity. We provide more details on this process and on the specifics of our error model in section 4.2. We will often present the results in the form “Square circuit depth” by which we mean the largest achievable square circuit depth, which is equivalent to $\log_2(QV)$ as per equation 4.3. The square circuit depth is a more tangible quantity than quantum volume, i.e. it directly communicates the circuit depth and the number of qubits utilized. The reason this quantity is exponentiated to define quantum volume is to represent the potential for quantum computers to be exponentially more powerful than their classical counterpart (for a fixed circuit volume).

For some of the results presented in this chapter we have altered the circuit requirement of the quantum volume metric to instead be just the native two qubit gate of the architecture, which focuses the analysis entirely on the cost of enabling global connectivity. The arbitrary two qubit gate requirement of quantum volume can be very functional for comparing architectures, particularly for those with drastically differing gate sets. For example, some trapped ion architectures have access to global entangling operations [139]. For architectures with similar gate sets, a large difference in the final fidelity can still arise due to differing decomposition methods. Approximate decomposition methods (which we cover in the following section) can considerably improve the final fidelity, particularly when starting with low two qubit gate fidelities. In the following section we provide more details on the gate requirement for quantum volume and provide a circuit for performing arbitrary two qubit gates for a Mølmer-Sørensen (MS) (trapped-ion) native gate set.

4.1.3 Gate requirements of Quantum Volume

The QV metric requires application of arbitrary, randomly generated SU(4) two qubit gates. The purpose of this requirement is to capture the power of the architecture's native gate set. There is an upper bound circuit which can express any arbitrary U(4) using 3 CNOTs and 15 elementary single qubit gates [140], with a native gate set consisting of $R_x(\theta), R_z(\theta)$, and the CNOT. We have translated this upper bound circuit into the native gate set of the shuttling based architecture analysed in chapter 3, which is $R_x(\theta), R_y(\theta)$ and the Mølmer-Sørensen (MS) two qubit gate [60]. The gate count of the new upper bound circuit is 3 MS gates and 18 elementary single qubit gates. We reduced the initial single qubit gate count from 29 to 18 by utilising basic commutation relations and the degrees of freedom which are available [133]. The upper bound circuit represents a worst case and optimal circuits can be found for particular SU(4)s using analytical techniques [141]. Most exact decompositions of arbitrary two-qubit gates will require the three native two qubit gates of the upper bound. A new technique demonstrated by IBM can considerably improve the fidelity of decomposing these gates [20]; Cross *et al* instead start with an allowable error on the decomposition, which allows one to identify cases which require less than the upper bound of three two-qubit gates. This can result in a considerable improvement to the final fidelity, particularly when working with lower two qubit gate fidelities.

Decomposing arbitrary two qubit gates

An upper bound circuit for expressing arbitrary two qubit gates in terms of $R_x(\theta), R_z(\theta)$, and the CNOT, was found by Vatan *et al* [140].

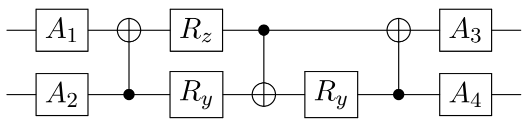


Figure 4.2: A circuit for implementing any transform in U(4) with a gate set consisting of $R_x(\theta), R_z(\theta)$, and the CNOT, where the gate A_i here represents an arbitrary single qubit transform, for a total gate count of 15 elementary single qubit gates and 3 CNOTs.

An arbitrary single qubit gate U_1, can be expressed in the form,

$$U_1 = e^{i\alpha} R_{\hat{n}}(\beta) R_{\hat{m}}(\gamma) R_{\hat{n}}(\delta) \tag{4.6}$$

for appropriate choices of $\alpha, \beta, \gamma, \sigma$, where $\hat{n}$, and $\hat{m}$ are non-parallel real unit vectors in three dimensions [142]. We have converted the circuit of figure 4.2 into the native gate set of the architecture investigated here, which is, $R_x(\theta), R_y(\theta)$ and the Mølmer-Sørensen gate $U_{MS}(\chi)$ [60] which has the form,

$$U_{MS}(\chi) = \begin{pmatrix} cos(\chi) & 0 & 0 & -isin(\chi) \\ 0 & cos(\chi) & -isin(\chi) & 0 \\ 0 & -isin(\chi) & cos(\chi) & 0 \\ -isin(\chi) & 0 & 0 & cos(\chi) \end{pmatrix} \tag{4.7}$$

where χ can be set between $-\pi/4$ and $\pi/4$. The new converted circuit is shown in figure 4.3, and has a gate count of 3 MS gates and 18 single qubit gates. The single qubit gate count was reduced by combining superfluous sequences of single qubit gates, utilising commutation relations and the available degrees of freedom. The MS gate commutes with any $R_x(\theta)$. When decomposing the CNOT and $R_z(\theta)$ gate, there is an available degree of freedom where one may choose the direction of rotation on certain R_y gates, which can then be used to eliminate some R_y gates from the circuit [133].

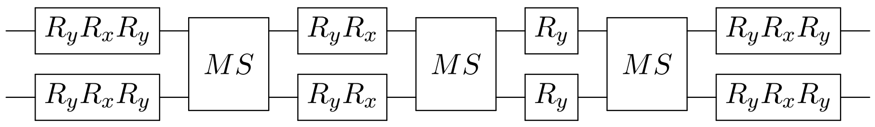


Figure 4.3: A circuit for implementing any transform in U(4) with a gate set consisting of $R_x(\theta), R_y(\theta)$, and the Mølmer–Sørensen gate [140], for a total gate count of 18 elementary single qubit gates and 3 MS gates.

After the publication of the paper that this chapter is primarily based upon, we investigated the approximate decomposition techniques [20] for the MS gate using the Qiskit python libraries. We compare three methods, first the exact decomposition for which virtually all randomly generated two qubit gates require the upper bound of three native two qubit gates. Next we performed approximate decompositions for which the native two qubit gate fidelity is an input; the final circuit error is estimated for decompositions using 1, 2, and 3 native gates, and the optimal circuit is chosen. The final method uses the approximate decompositions in addition to considering the decomposition of the target two qubit gate multiplied by the SWAP gate. This increases the frequency at which a high quality decomposition can be found using less than three native gates, and the SWAP can easily be undone in the back-end by switching the qubit indices following the gate.

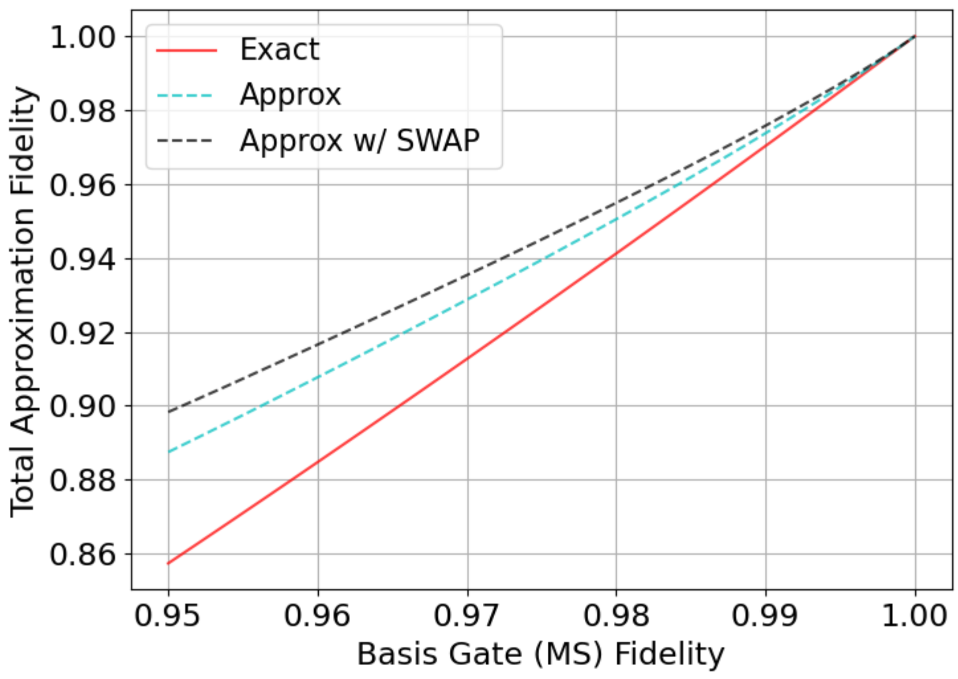


Figure 4.4: The total approximation fidelity as a function of the native (MS) two qubit gate fidelity for different decomposition methods. In red the exact decomposition is used where every randomly generated two qubit gate requires the upper bound of three applications of the native two qubit gate. In cyan the approximate decomposition method is used where that native two qubit gate fidelity is considered and when desirable a decomposition using fewer than three applications of the native gate is used. In black the approximate decomposition is again used but it also considers the option of combining the target two qubit gate with a logical swap which increases the frequency that a desirable approximate solution can be found.

In figure 4.4 we plot the three methods as a function of the base native two qubit gate fidelity. As expected we can see that these techniques are most powerful when the native two qubit gate fidelity is low, and converge towards no discernible benefit as the native two qubit gate approaches 100%. With a native two qubit gate fidelity of 95%, the average fidelity of a random two qubit gate decomposition is 86% with the exact decomposition, whereas it is 90% with the approximate decomposition with optional swaps. With two qubit gate fidelities close to or exceeding 99.9%, which is a fidelity that has already been demonstrated by some trapped ion systems, the benefit of this approximate decomposition method is negligible.

4.2 Error model

The effective error ϵ_{eff} for each depth-1 circuit includes gate error, and errors associated with gate decomposition, connectivity and parallelizability. The effective error can be used to calculate the achievable depth. Many iterations of the randomly generated circuit should be used to best capture the properties of the device. In this section we present an error model for the quantum computing design analysed in chapter 3 and present results for a range of experimental parameters that may be achievable. In the following analysis we assume linear propagation of errors, which represents a worst-case outcome, as it does not account for the possibility of a new error reducing a previous error. We combine the errors associated with connectivity and gates, as opposed to a full simulation of the quantum states and associated noise model. The advantage of this methodology is that we are able to make estimations on effective error (and therefore achievable depth) for a wide range of qubit numbers and device sizes, without a full state simulation. A full stack quantum state simulation is something we are working towards and it may be able to provide greater insights. The effective error ϵ_{eff} for this design and circuit requirement is, $\epsilon_{eff} = \epsilon_{gate} + \epsilon_{conn}$, where ϵ_{gate} is the two qubit gate error and ϵ_{conn} is the error associated with enabling the required global connectivity. We decompose ϵ_{conn} into two components $\epsilon_{conn} = \epsilon_{deco} + \epsilon_{loss}$ where ϵ_{deco} is the quantum decoherence associated with the total time taken to enable connectivity where $\epsilon_{deco} = 1 - e^{-t/c}$ for time t and coherence time c. Recent work by Kaufmann *et al* [85] demonstrated high state fidelity shuttling (99.9994%), where the coherence time associated with shuttling was extrapolated to be $2.13s$. A coherence time of $50s$ has been demonstrated for stationary ions in the atomic clock states of calcium [143]. In figure 3.5 we quantify the average time required to enable connectivity as a function of device size (qubit number). The stated dimensionless time τ can be converted to a real time by multiplying it with the expected time to shuttle an ion between two adjacent X-Junctions. For ion shuttling speed, a distance of $280\mu m$ has been demonstrated with adiabatic techniques in $3.6\mu s$ [84] and $12.8\mu s$ [85]. Diabatic techniques [144, 145, 146] can enable much greater shuttling speeds, and $\sim 80\ ms^{-1}$ has been demonstrated [145]. Lau and James calculate that the maximum speed a $^{40}Ca^{+}$ ion can be transported across a 100 μm trap without excessive error is 10,000 ms^{-1} [147]. There will be an additional time cost associated with performing a single combination and a separation of ions, per iteration of the depth-1 circuit, which have been performed in 80μs [148, 149]. ϵ_{loss} represents the likelihood for an ion to be lost to the vacuum per iteration of the depth-1 circuit. Investigations of ion loss for routing across X-Junction centres [89] found continuously Doppler cooled ion survive more than 10^5 round trips whereas uncooled ions survive at least 65 round trips. This experimental demonstration was for corner X-Junction crosses in contrast to linear X-Junction crosses. We may expect that linear X-Junction crosses would result in lower ion-loss rates due to the simpler waveforms that are required, but in this analysis we treat them as identical capturing a worst case situation. Ion loss can occur when its motional energy exceeds the trap depth, which can be remedied by increasing the trapping potentials and by cooling techniques. Ion loss may also result from temperature independent collisions which instead directly related to the quality of the vacuum. Significant work is carried out in order to allow the application of large trapping voltages in order to increase the effective trapping potential; recently trapping voltages as large as 1000V have been demonstrated [150]. In figure 3.7 we quantify the average number of X-Junction crosses, X_{count}, as a function of device size (qubit number), which can be combined with an ion loss per shuttle rate, X_{loss}, for ϵ_{loss}. This can all be combined into a single equation defining the effective error in this design

$$\epsilon_{eff} = \epsilon_{gate} + (1 - e^{-t/c}) + (X_{count} \times X_{loss}) \tag{4.8}$$

This error model can be used to estimate the achievable depth for a wide variety of device sizes and experimental parameters for devices following this design. Ion loss is indeed a very different class of error as compared to the state fidelity errors associated with gates and decoherence. Here we are making the simplifying assumption for the sake of analysis that ion loss results in a complete computational failure and so can be combined into a single figure of merit (total error). The gate error will depend on the requirement of the algorithm we are assessing against, which in the case of QV is the arbitrary two qubit gate, but at times we will consider the native two qubit gate to focus on the cost of enabling global connectivity.

4.3 Using Quantum Volume to compare architectures

We use our error model to quantify the largest achievable square circuit depth, equivalent to $\log_2(QV)$ as per equation 4.3 as a function of the native two qubit gate fidelity for this architecture with two different assumptions on experimental shuttling parameters, shown in figure 4.5. These can be compared to the upper bound of this metric which corresponds to a hypothetical architecture with free, all to all connectivity. To demonstrate an example, a device with free all to all connectivity and a two qubit gate fidelity of 99.9% has a square circuit depth of 32. This implies that one could effectively run a globally connected native

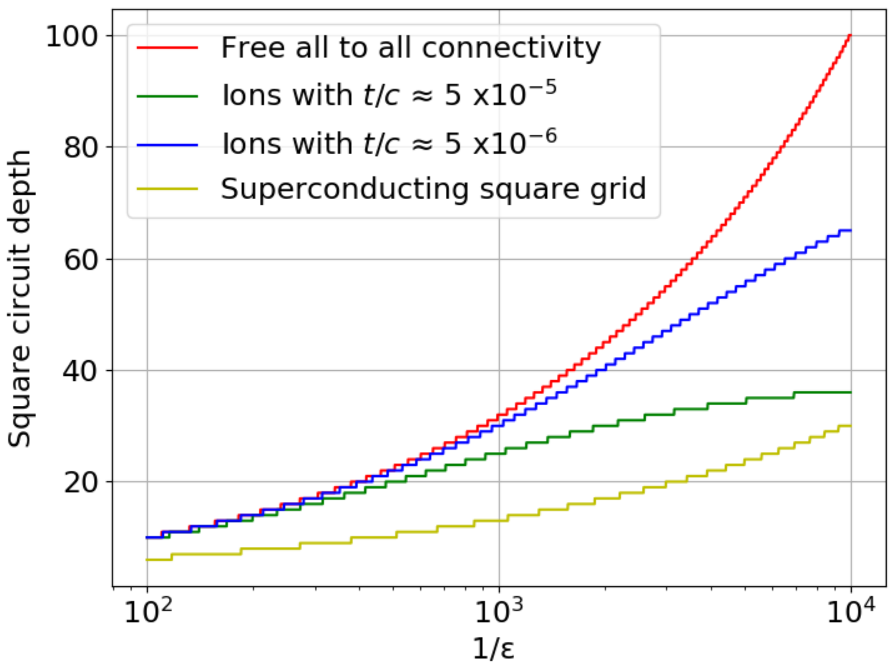


Figure 4.5: The largest achievable square circuit depth, equivalent to $\log_2(QV)$ but with a native two qubit gate requirement, as a function of inverse gate error, $1/\epsilon$. Red: An architecture with all to all connectivity where the square circuit depth is solely defined by the native two qubit gate fidelity using equation 4.2; this represents the upper bound. Green: The shuttling based trapped ion architecture with our proposed error model and the routing results of the previous chapter. The coherence time and the time taken to shuttle between adjacent X-Junctions is extrapolated from work by Kaufmann *et al* [85]. We assume a distance between adjacent X-Junctions of 2500μm [74] which implies a shuttling time, t, of 114μs, and we use the demonstrated state fidelity of shuttling [85] to infer a coherence time, c, of 2.13s, and so $t/c \approx 5 \times 10^{-5}$. We assume an ion loss rate of 10^{-5} per X-Junction pass [89]. We assume each iteration of the depth-1 circuit requires one combination and one separation operation, each of which have a duration of 80μs [148, 149], and we assume the state fidelity of the operation can be inferred from the coherence time. Blue: All the assumptions are identical to the above except for the coherence time (for shuttling) which has been increased by a factor 10 [143]. Yellow: A model of a square grid superconducting architecture where connectivity is enabled through sequences of nearest neighbour swap operations which require 3 native two qubit gates (the CNOT). The depth overhead was found to scale as a function of qubit number N as $2.77\sqrt{N} - 4.53$ using the publicly available quantum compiling software, CQC's $t\,|ket\rangle$; improvements to the connectivity compiler would reduce this overhead.

two qubit gate algorithm with 32 qubits at depth 32. We investigate up to a two qubit gate fidelity of 99.99%; this analysis indicates that without error correction techniques, chasing high qubit numbers will be futile even with considerable improvement to the current state of the art two qubit gate fidelities. The trapped ion plots of figure 4.5 have an ion loss rate of 10^{-5} [89]; we found that increasing this rate substantially decreases the square circuit depth, which seriously emphasises the importance of achieving an ion loss rate of this order. The ion loss rate can be decreased by improving the vacuum quality (which reduces collisions), deeper trapping potentials and by techniques such as sympathetic cooling [151]. Ions may also be automatically reloaded from a filled reservoir trap section. We investigate the impact of the rate of ion loss in section 4.4.1.

We also quantify this metric for a model of a superconducting architecture, which is a square grid with only nearest neighbour interactions. In superconducting square grid systems, connectivity is enabled by sequences of swap operations, and the best known method has an overhead of $\Theta(N^{0.5})$ [152] for the random complete graph (global connectivity). IBM provide an equation to estimate the depth overhead, of the form $(a\sqrt{N} + b)$, for a square grid but it includes their gate decomposition costs of arbitrary two qubit gates [20]. Cowtan *et al* developed a compiler to map quantum circuits to devices with restricted qubit connectivity and provides results on the depth overhead for nearest neighbour square devices [50]. Using the publicly available software, CQC's $t\,|ket\rangle$ and its recently improved connectivity compiler, the depth overhead was found to scale with qubit number N as, $2.77\sqrt{N} - 4.53$. This overhead corresponds to a depth-1, 100% gate density, native two qubit gate (CNOT) algorithm with $10N$ iterations. A SWAP gate is implemented with three CNOTs and no advantageous initial qubit mapping was utilised.

The native two qubit gate of this trapped ion design is the Mølmer-Sørensen [60] and although it does not directly depend on the motional state, it is affected by the heating rate and experimental offsets whereby it is favourable to begin in a low motional state. Therefore to reach the high two qubit gate fidelities used in figure 4.5, it will be necessary to use cooling techniques. Techniques such as Doppler and sideband cooling are only suitable for the beginning of a quantum algorithm as they do not preserve quantum information. Sympathetic cooling is a way of actively cooling throughout a quantum algorithm, whereby the qubit is sympathetically cooled via a different laser cooled ion species. It is likely to be a critical technique for the use of trapped ion devices, particularly in the fault tolerant regime [74]. Shuttling based designs may benefit from multi-species shuttling. The relative difference between the upper bound of free all to all connectivity, and the plots for trapped ions, increases with the two qubit fidelity due to the independent cost associated with shuttling. We find a substantial difference in the achievable square circuit depth between the superconducting plot and the all to all, particularly at higher two qubit gate fidelities. Superconducting square grids have a slower growth rate with two qubit gate fidelity because the associated depth overhead of swaps increases with the number of qubits (the size of the grid). The trapped ion plots can be seen to plateau with increasing two qubit gate fidelity which is due to the static (independent of gate-fidelity) error contributions associated with connectivity, which transition to becoming the dominating error source as the gate fidelity increases. In this model, the trapped ion design outperforms the superconducting square grid for this set of experimental shuttling parameters. The total shuttling time required to enable global connectivity, τ, scales as $\tau = 1.3\,(3)\,\sqrt{N} + 2\,(5)$, where $\tau = 1$ is the shuttling time between adjacent X-Junctions. This scaling is comparable to the depth overhead for swapping on the superconducting square grid. Extrapolating from the high state fidelity shuttling of Kaufmann *et al* [85], it implies a fidelity per shuttling operation ($2500\mu m$) of 99.995% which is significantly higher than the two qubit gate fidelities achieved so far by superconducting systems. In order to facilitate further work with our error model by others, we have made it open access [134]. To experimentally implement the work presented here, a key challenge is to build and operate such a trap as shown in Figure 3.2. A trap needs to be fabricated for which a number of approaches are being perused [153].

4.3.1 Quantum volume comparison with recent experimental results

The quantum volume metric has seen a wider adoption in the community in recent years and now various teams are announcing their best achievements in the spirit of constructive competition. In figure 4.6 we have recast figure 4.5 using quantum volume as opposed to square circuit depth. Here we have also amended the circuit requirement back to the original arbitrary two qubit gate requirement for which we make the simplifying assumption that for these architectures the arbitrary two qubit gate is achieved with three applications of the native gate. We include recently achieved quantum volume values from IBM, of 64 (6 qubits square depth circuit) and Honeywell 128 (7 qubits square depth circuit). We can see that the exponential scaling of the quantum volume can lead to very large values for sufficiently high two qubit gate fidelities. IonQ recently announced that they expect a quantum volume of ~ 4 million on their 32 qubit chip which has a 99.93% two qubit gate fidelity, but this claim has not yet been substantiated. The IonQ architecture does

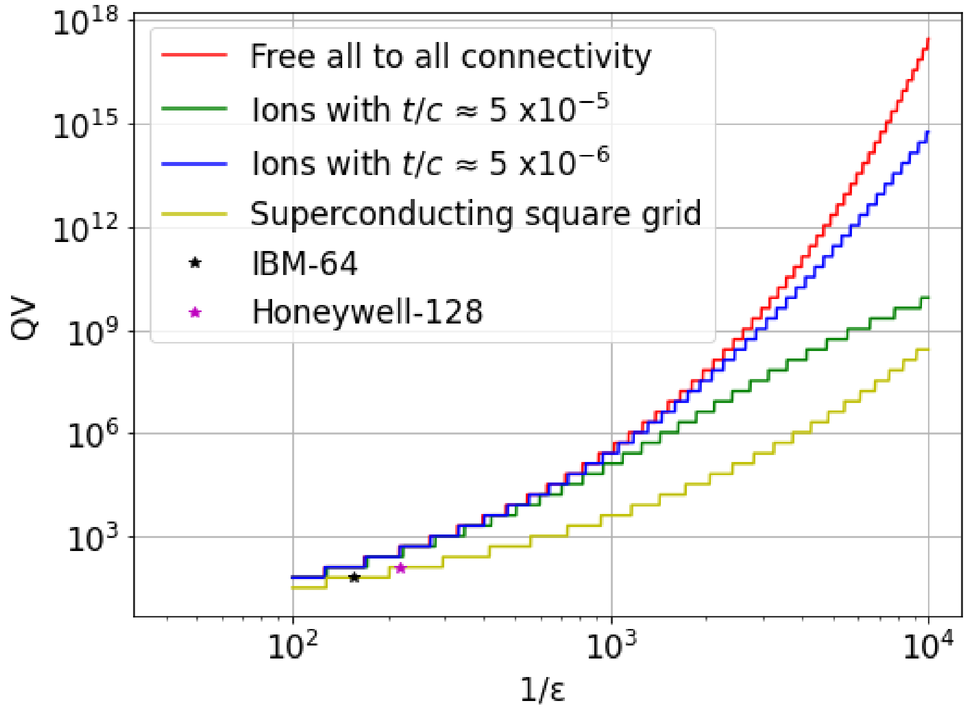


Figure 4.6: Quantum volume as a function of inverse gate error, $1/\epsilon$ with the same error model assumptions as described previously. Red: An architecture with all to all connectivity where the square circuit depth is solely defined by the native two qubit gate fidelity using equation 4.2; this represents the upper bound. Blue: The trapped ion architecture investigated in this manuscript using our proposed error model and the routing results of the previous section, with the ratio of shuttling time to coherence time equal to 5×10^{-5}. Green: All the assumptions are identical to the red plot except for the coherence time which has been increased by a factor 10 [143]. Yellow: A model of a square grid superconducting architecture where connectivity is enabled through sequences of nearest neighbour swap operations. The depth overhead was found to scale as a function of qubit number N as $2.77\sqrt{N} - 4.53$. Black star: IBM achieved a quantum volume of 64 in 2020 with 6 qubits (square circuit depth). Magnenta star: Honeywell achieved a quantum volume of 128 in 2020 with 7 qubits (square circuit depth).

have access to arbitrary connectivity two qubit gates without the need for shuttling and their claimed result would fit within the free-all-to-all model of figure 4.6; some questions for this architecture type would be whether the high two qubit gate fidelity can be maintained while performing the maximum number of gates in parallel (in this case 16), and the general vision for continuing to scale to higher qubit numbers.

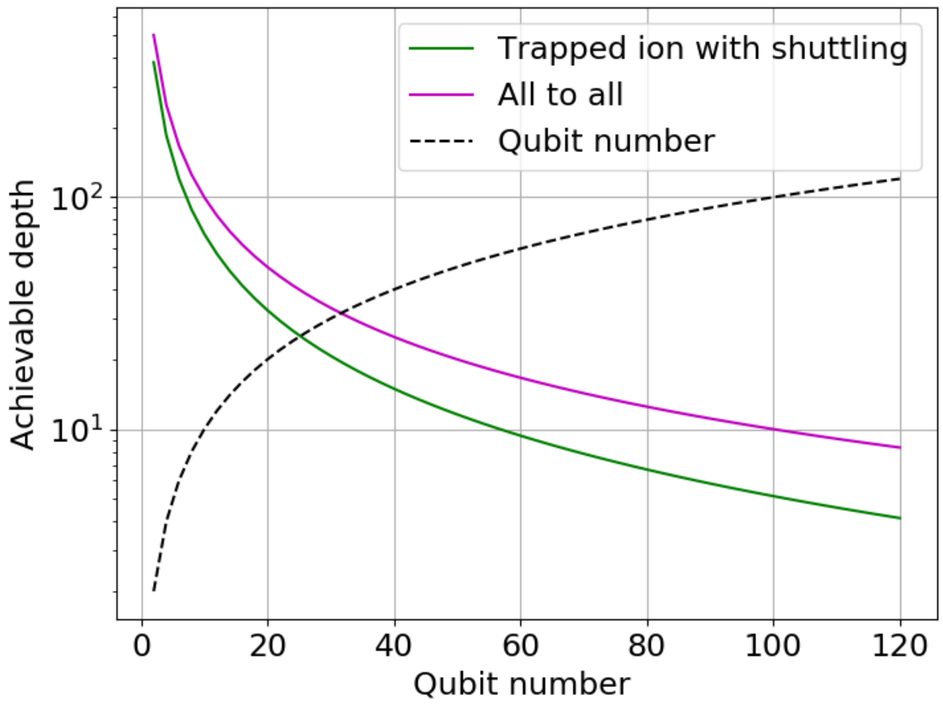


Figure 4.7: Achievable depth for a globally connected native two qubit gate circuit as a function of qubit number with a fixed two qubit gate fidelity of 99.9%. All other components of the error model are the same as described in the caption of figure 4.5 such as an ion loss rate of 10^{-5} and a coherence time of 2.13s.

4.4 Using quantum volume to meter experimental priorities

In this section we present results that can be used to understand the relative importance of various experimental parameters.

Quantum volume (and largest square circuit depth) is an obviously useful metric but some information is lost in the calculation. In figure 4.7 we plot achievable depth as a function of qubit number. The depth on low numbers of qubits starts high (for 2 qubits it is equal to 500 for the all to all case with the two qubit gate fidelity of 99.9%) and then falls rapidly with qubit number because of the $1/N$ factor, and for realistic architectures the effective error term will also depend on N. The intersection of the achievable depth with the dashed qubit number line is equal to the quantum volume when exponentiated, as defined by equation 4.3.

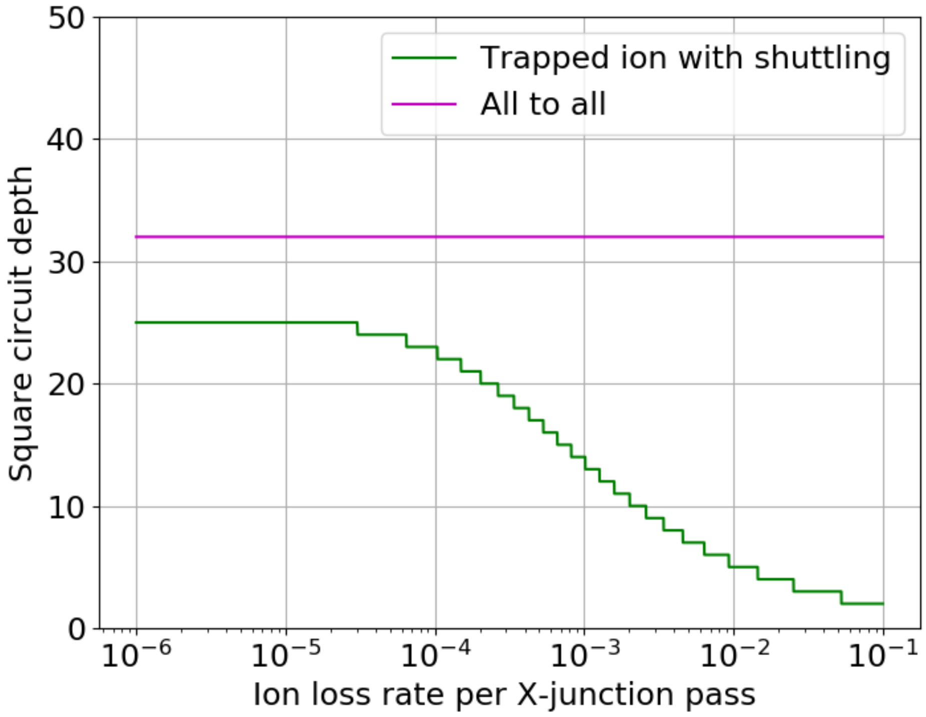


Figure 4.8: The largest achievable square circuit depth, equivalent to $\log_2(QV)$ but with a native two qubit gate requirement, as a function of the ion loss rate with a fixed two qubit gate fidelity of 99.9%. All other components of the error model are the same as described in the caption of figure 4.5 such as a coherence time of 2.13s associated with shuttling operations.

4.4.1 The dependence on ion loss

We now resort back to using the quantum volume metric as in figure 4.5 but instead fix the two qubit gate fidelity (here to 99.9%) and vary the ion loss rate. Within the error model this parameter is the rate at which ions are lost to the vacuum per journey past the centre of an X-Junction.

Investigations of ion loss for routing across X-Junction centres [89] found continuously Doppler cooled ion survive more than 10^5 round trips whereas uncooled ions survive at least 65 round trips. Ion loss occurs when its motional energy exceeds the trap depth, which can be remedied by increasing the trapping potentials and by cooling techniques. The large difference of the ion loss rate between cooled and uncooled ions places importance on cooling techniques which do not disturb the quantum information, such as sympathetic cooling (explained in more detail in section 4.3). In figure 4.8 we plot the dependence of quantum volume versus ion loss across a range of $10^{-6} - 10^{-1}$. Using this graph we can

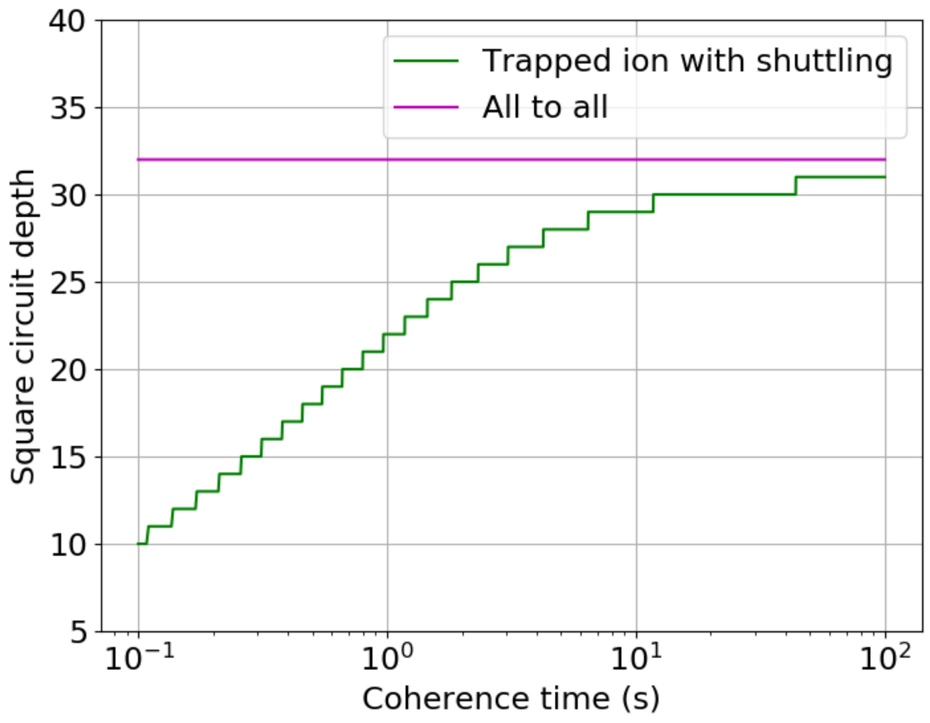


Figure 4.9: The largest achievable square circuit depth, equivalent to $\log_2(QV)$ but with a native two qubit gate requirement, as a function of the coherence time with a fixed two qubit gate fidelity of 99.9%. All other components of the error model are the same as described in the caption of figure 4.5 such as an ion loss rate of 10^{-5}.

see that when the ion loss rate is kept below approximately 10^{-4} then the fidelity loss from the upper bound (all to all) is dominated by the decoherence associated with the time spent shuttling and the 0.1% gate error.

4.4.2 The dependence on coherence time

In this section we use our model to assess the importance of the coherence time with a two qubit gate fidelity of 99.9%. In figure 4.9 we can see that once the coherence time reaches values of approximately 10s there are diminishing returns and the fidelity starts to be dominated by the two qubit gate fidelity. In our error model the coherence time corresponds to both static and moving ions which is corroborated by the high state fidelity shuttling results of the Siegen group [85] where the coherence time was 2.13s.

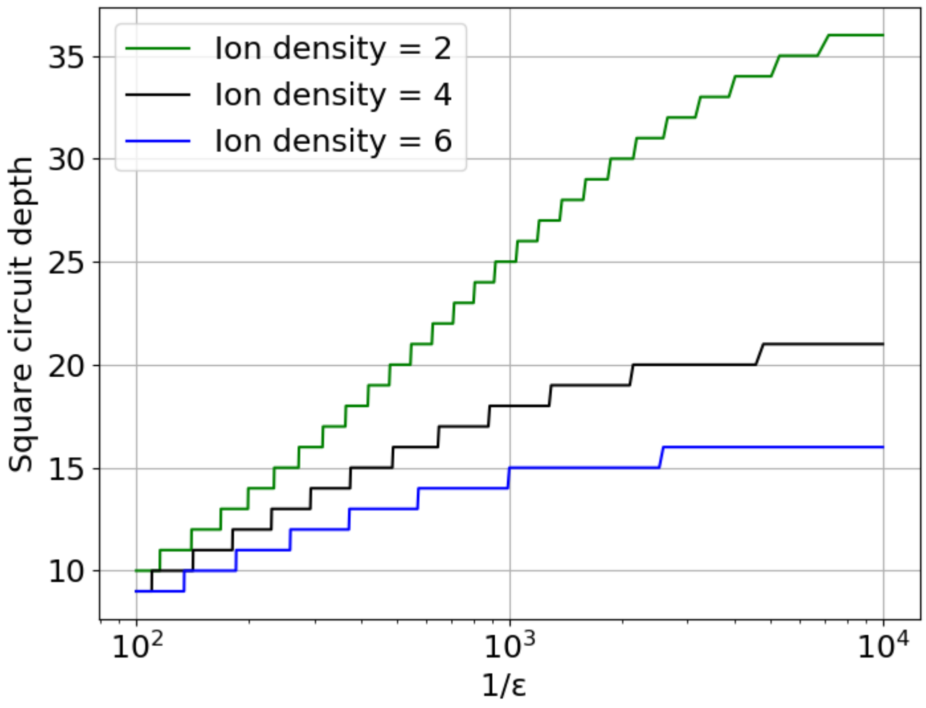


Figure 4.10: The largest achievable square circuit depth, equivalent to $\log_2(QV)$ but with a native two qubit gate requirement, as a function of inverse gate error and a fixed two qubit gate time of $1000\mu s$. All other components of the error model are the same as described in the caption of figure 4.5 such as an ion loss rate of 10^{-5}.

4.4.3 The dependence on ion density

So far this chapter has assumed that there are always two ions initialized per available gate zone which enables the 100% gate density algorithm to be executed in one round. Increasing the number X-Junctions will be experimentally expensive and it will therefore be desirable to increase the ion density at the various stages of development. In figure 3.9 we characterize the effect of increasing the ion density on the time taken to route. When considering the effective error we must define the total time taken which will now include the additional rounds of shuttling and gates. Now the gate time will also be a parameter of the error model where previously it was only considered by the abstraction of the two qubit gate error. Laser free gates have been demonstrated with a time cost of 3400 μs [68], 3250 μs [65] and 2938 μs [69] and laser based gates have been demonstrated as fast as 30 μs [70] and 115 μs [71].

In figure 4.10 we fix the gate time to 1000 μs and compare the QV_{native} for three different ion densities, 2, 4 and 6. The relative difference between the plots is increased as

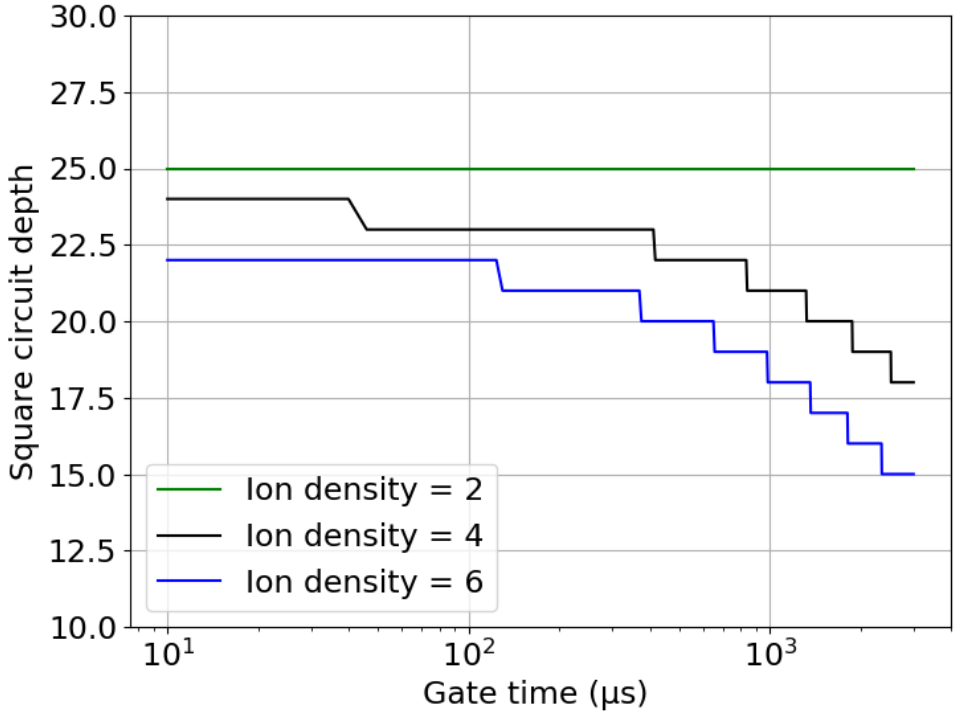


Figure 4.11: The largest achievable square circuit depth, equivalent to $\log_2(QV)$ but with a native two qubit gate requirement, as a function of gate time and a fixed two qubit gate fidelity of 99.9%. All other components of the error model are the same as described in the caption of figure 4.5 such as an ion loss rate of 10^{-5}.

we increase the two qubit gate fidelity, as it transitions away from being the dominating error. The relative importance of ion density with a constant coherence time is dependent on the gate time, and therefore laser-based gates (which are generally considerably faster) may be more suitable for near-term, high ion-density devices. In figure 4.11 we fix the two qubit gate fidelity to be 99.9% and vary the gate time instead. For this gate fidelity the improvement to the square circuit depth begins to plateau once the gate time reaches approximately $500\mu s$ and below. This analysis will be helpful in determining the appropriate ion density to use in a given device, and can help meter experimental priorities, such as how urgently should one rush to larger devices, as opposed to loading more ions onto a smaller device.

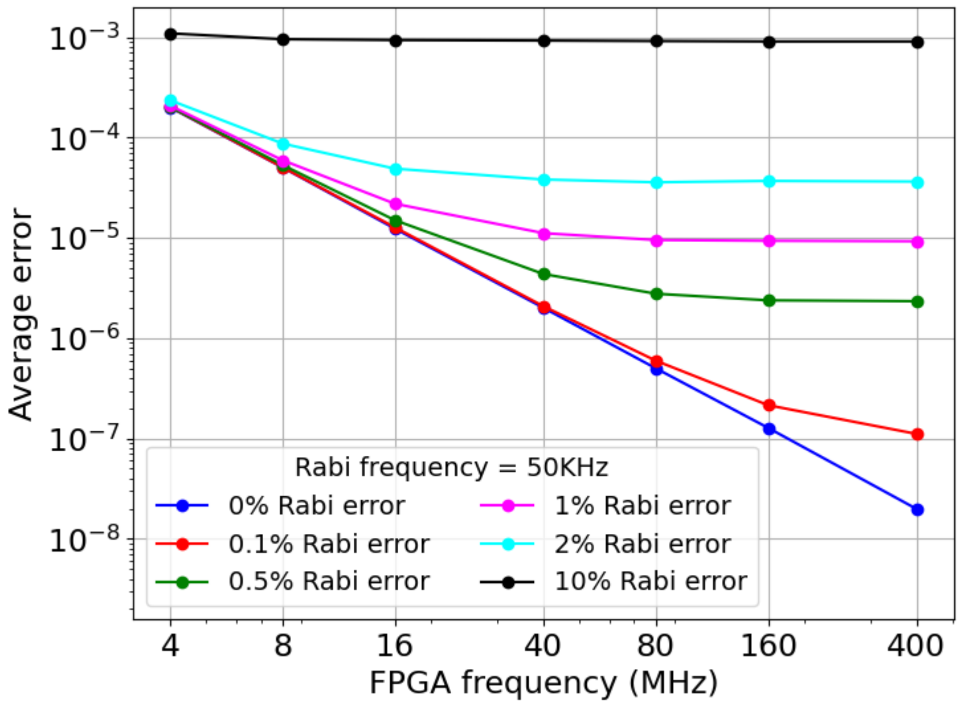


Figure 4.12: The average single qubit gate error with a fixed Rabi frequency of 50KHz, as a function of the FPGA frequency and the uncertainty in the Rabi frequency ω_ϵ. The Rabi frequency was chosen from a normal distribution with mean = 50KHz and $3\sigma = \omega_\epsilon \times 50KHz$.

4.5 Single qubit gate fidelity as a function of experimental parameters

There is a resolution of control on the fields that perform the logical operations. The angle of rotation, θ, for the single qubit gate is determined by the duration of the applied field, T, and its associated Rabi frequency, ω_R, where $\theta = 2\pi\omega_R T$. The frequency of the FPGA, F_{FPGA}, determines the time resolution of the microwave field application duration so that the maximum error on the duration of the application is equal to $T_{error} = 1/F_{FPGA}$. This finite resolution will lead to over or under rotations, ϕ relative to the desired angles. We can define the operational error that evaluates the distance between a target unitary U_{target} and an error affected unitary U_{err} as

$$\epsilon = 1 - \left| \frac{1}{D} \Big\langle U_{\text{target}}, U_{\text{err}} \Big\rangle_F \right|^2 \tag{4.9}$$

where D is the dimension of the Hilbert space, 2 for single qubit gates, and $\langle \cdot, \cdot \rangle_F$ is

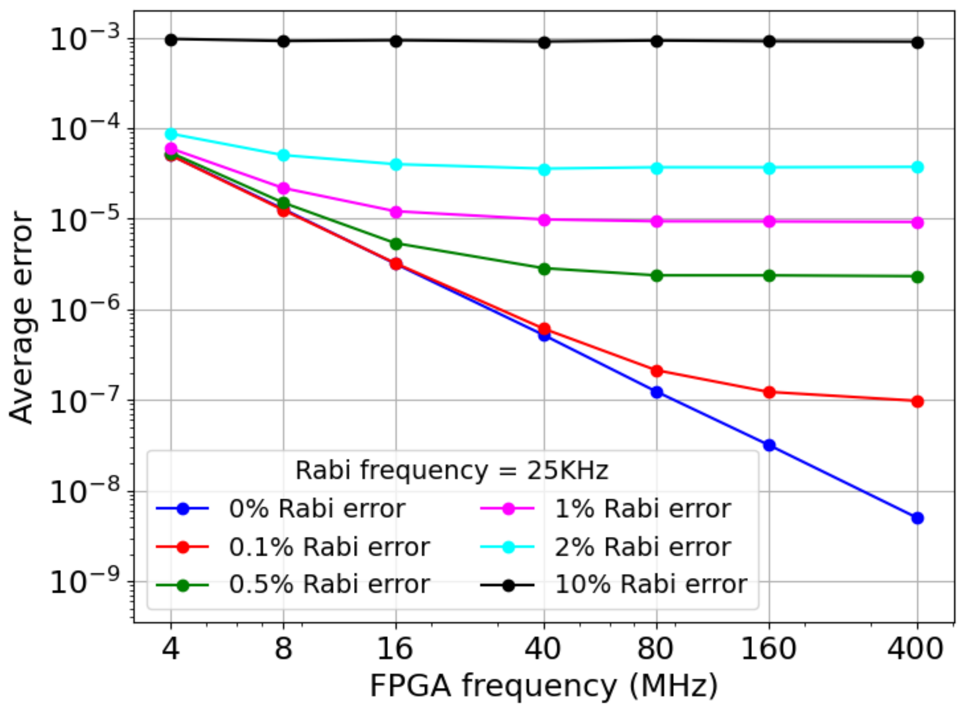


Figure 4.13: The average single qubit gate error with a fixed Rabi frequency of 25KHz, as a function of the FPGA frequency and the uncertainty in the Rabi frequency ω_ϵ. The Rabi frequency was chosen from a normal distribution with mean = 25KHz and $3\sigma = \omega_\epsilon \times 25KHz$.

the Frobenius inner product which can be calculated with

$$\langle A, B \rangle_F = \sum_{i,j} A^*_{ij} B_{ij} = \mathrm{Tr}\left(A^\dagger B\right) \tag{4.10}$$

There is an additional error source associated with an uncertainty in the produced Rabi frequency ω_ϵ, which may result from background noise or from distortions on the original production of the field. We calculate the average expected rotation error ϕ by sampling from normal distributions set by $3\sigma = T_{error}$ and $3\sigma = \omega_\epsilon$ respectively, where ω_ϵ is defined as a fractional error on the Rabi frequency. For example, $\omega_\epsilon = 0.05$ implies the applied field may be at most $\sim 1.05\omega_R$. We then convert the expected rotation error ϕ into an error using equation 4.9 and plot for a fixed Rabi frequency, as a function of the FPGA frequency, and the uncertainty on the Rabi frequency. In figure 4.12 we plot the average error for a Rabi frequency of 50KHz which is a value that is regularly used within the lab, and across a wide range of FPGA frequencies, the centre of which, 40MHz, is currently used within our lab. We can see that a large Rabi error, e.g. 10%, dominates the average error relative to the time resolution of the FPGA. It will be necessary to have a Rabi error of around 3% or less to achieve single qubit gate fidelities of at least 99.99%. In figure 4.13 we instead plot with a Rabi frequency of 25KHz, which results in a lower error for a given time resolution. The time resolution and Rabi frequency are not the only sources of error in the single qubit gate however, for example, the total time duration of the gate will determine the error from decoherence (for a given coherence time). Although lowering the Rabi frequency may reduce the error associated with the time resolution of the FPGA, it will increase the total duration of the gate, and so increase the error from decoherence. The Rabi frequency uncertainty may be greater for larger devices with conflicting fields, and varied distances between gate zones and emitters. This preliminary analysis serves as a best case error analysis for a given FPGA frequency and Rabi frequency uncertainty (i.e. the analysis only includes these contributions). Provided the FPGA frequency is of around 16-40MHz or greater, the associated timing error is unlikely to represent a bottleneck in achievable fidelities. In future work, a more detailed error model should be able to meter experimental priorities.

4.6 Comparing photonic interconnects and shuttling

In this section we will compare photonic interconnects and shuttling, by quantifying the time to enable global connectivity for large devices.

As described in chapter 2, there is a large variety within the category of trapped ion architectures. One of the main ways in which they vary is in how connectivity is enabled. The connectivity graph that can be enabled with photonic interconnects is highly flexible as the cost of the connectivity operation is independent of any spatial distance, and each module may be connected to numerous other modules. In this section we focus on a particular design consisting of modules containing a small number of data qubits (e.g. 1 data qubit) which are connected via photonic interconnects.

4.6.1 Connectivity for large scale devices

Large scale devices will need to use error correction in order to run at a functional circuit depth, and it will incur a potentially large physical qubit to logical qubit overhead. The surface code is the most researched error correction technique, particularly in regards to end to end physical resource estimation. There are many other error correction techniques available and the best choice will likely depend on the underlying architecture's characteristics, such as it's available connectivity. The surface code relies only on nearest neighbour interactions and so it is well suited to superconducting architectures. Higher dimensional error correction codes can have access to a greater range of transverse (cheap to apply) gates [154, 155] which may considerably improve final run-times but few architecture types will be able to realise this 3D (or greater) connectivity; photonic-interconnected modules may be the most flexible architecture with regards to it's possible connectivity graph. Error correction codes which rely on global interactions at the physical level have favourable encoding rates as a function of code distance [30] but enabling this global connectivity on large devices may be challenging, or the connectivity overheads may outweigh the benefits relative to closer-range-connectivity codes.

These alternative error correction techniques should motivate investigation into the cost of enabling global connectivity for large scale devices.

4.6.2 The time cost for global connectivity

For the shuttling based design we use the routing results of chapter 3, paired with achieved experimental results for high state fidelity shuttling [85]. For a photonic interconnected based design consisting of many modules each containing one data qubit [83], we rely on the butterfly network routing results of Brierley [156] and Herbert [157]. Herbert defines the number of operations required for a single round of global connectivity as a function of the number of connections per module; his results represent a best case, in that the scaling is strictly bounded below, but not above, and so the actual operation overhead may be greater than estimated here. In this analysis we abstract away the details of communication ions required for the photonic interconnects. The assumption is that each module contains one data ion and the necessary number of communication ions required to enable the stated number of photonic interconnects.

For the time cost of an individual photonic interconnect operation we use a recent experimental result which represents the state of the art, where remote Bell pairs are generated at a rate of 182Hz and fidelity of 94% [82]. We assume that to reach a satisfactory gate fidelity, entanglement distillation will be utilized and we rely on the purification protocols presented by Nigmatullin et al [83]. In this referenced work each module is assumed to have a total of 5 ions, which includes 1 application (data) ion (Ca^+), 2 ions to generate ion-photon entanglement (Sr^+), 1 shuttling envoy ion (Ca^+), and 1 final Ca^+ ion to mediate the interaction between the data ion and the ion-photon entanglement ions. There are three stages of purification presented with each stage further improving the output fidelity, and the number of stages would be chosen based on the raw entanglement fidelity and the desired output fidelity. Assuming a raw entanglement fidelity of 94%, stage

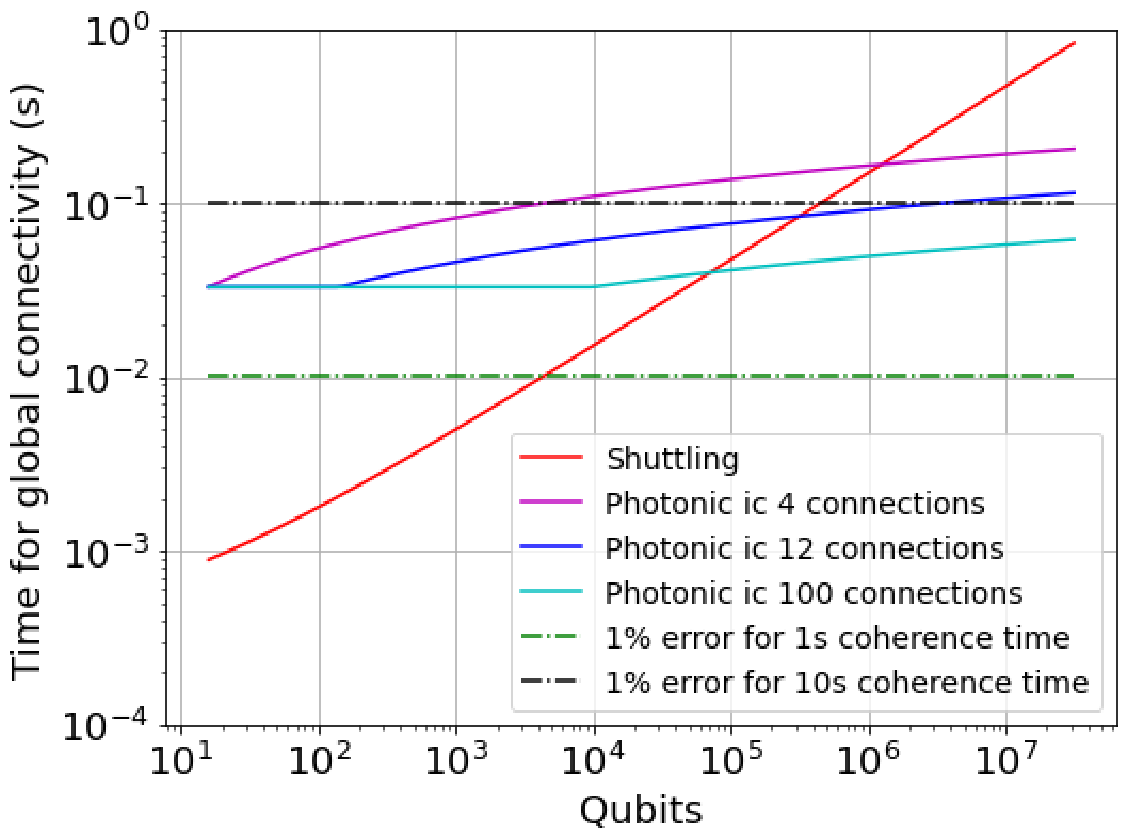


Figure 4.14: The total time for global connectivity as a function of the number of qubits in the device for different architectures. The shuttling based approach (red) uses the lane priority routing algorithm of chapter 3 with shuttling speeds motivated by recent experimental results [85]. The photonic based approach uses the butterfly network with different numbers of connections per module [157], here listed with 4 (magenta), 12 (blue), 100 (cyan). Horizontal dashed lines correspond to the time required for a 1% error with a coherence time of 1 second (green) and 10 seconds (black). The time for an individual photonic interconnect operation is motivated by the recent state of the art, where entanglement was generated at a frequency of 182Hz and fidelity of 94% [82], and we assume that to reach a satisfactory gate fidelity, entanglement distillation will be utilized which would reduce the rate by a factor 6 with 3 stage purification [83].

three purification would improve the output fidelity to 99.7% and reduce the effective rate by a factor 6 [83]. When an accurate error model is known for all operations in the device, distillation protocols can be further specialized to improve performance.

Commercially available switches offer 1-12 branching with a loss of 0.9dB and alternative micro-mechanical mirror arrays can offer higher levels of switching at the cost of a greater loss [83]. Greater switching can be achieved by combining the switches in series, but the associated loss compounds. The usual technique to deal with loss in a classical network is to utilize amplifiers, however this is not currently possible for single photon transmission as is required in quantum networks. Due to this, the total loss in the quantum network should be considered as an effective slow-down factor, where the signal will have to be resent a number of times to account for the loss. The most effective switches are designed for telecom wavelengths and so down conversion may be necessary for their use, alternatively an additional ion species may be utilized for the transmission, but the mixed species ion interactions are in themselves difficult to mediate. In figure 4.14 we plot the total time required to enable a single round of global connectivity as a function of the number of qubits within the device. We investigate a wide range of physical qubit numbers, up to 10^7; fault tolerant applications using the surface code often require in excess of 10^6 qubits [113]. We can see that for device sizes smaller than 10^5 physical qubits, the shuttling based design can enable global connectivity more quickly than the photonic based design with 12 connections per module. We include horizontal lines corresponding to the time required for a 1% error with coherence times corresponding to 1 and 10 seconds, which may be useful for estimating the upper bound device size at which it is reasonable to consider global connectivity. With a coherence time of 10 seconds this upper-bound is the same for both the shuttling design and the photonic design with 12 connections, at approximately 10^5. This work constitutes a preliminary investigation, to make it more rigorous for the shuttling model, one should consider the costs associated with mid-circuit sympathetic cooling. For the photonic interconnects model one should include the costs of switches and further research is needed to more concretely define the operation overhead for a given network structure. Furthermore, for future-looking comparisons such as these, one should consider the relative potential for improvements of the shuttling speed and photonic interconnect rate.

4.7 Summary

In this chapter we have proposed an error model for a shuttling based trapped ion quantum computer that incorporates the results of the routing simulations of the previous chapter. Using the model we have estimated the computational power (quantum volume) of near term devices as a function of experimental parameters. We have found that the shuttling based design will have a low cost associated with enabling distant connectivity relative to superconducting devices, and that the major bottleneck to quantum volume will be the two qubit gate fidelity. We have investigated the impact of ion density (number of ions initialized per X-Junction) by considering the reduction in gate parallelizability as a function of the two qubit gate duration and fidelity. We have estimated the expected error on single qubit gates that results from the time resolution of control on the microwave fields and local amplitude variation. Finally, we investigated the time cost for global connectivity for the shuttling based trapped ion design and for a trapped ion design using small modules connected via photonic interconnects. We used the state of the art rates for photonic interconnects and high fidelity shuttling and consider a wide range of device sizes from 10 to 10 million; we found that the shuttling based approach can enable global connectivity more quickly for device sizes with fewer than 10^5 qubits.

In the future we intend to develop more detailed error models which can be incorporated into a state simulator; such a tool could be used to investigate the suitability of error correction protocols or for near term algorithm optimization. The following chapter focuses on fault tolerant resource estimation for quantum chemistry.

Chapter 5

Fault Tolerant Resource Estimation for Quantum Chemistry

In this chapter we provide an overview of quantum algorithms for quantum chemistry, differentiate NISQ and fault tolerant techniques, and provide gate count resource estimates for estimating ground state energies for a variety of small molecules. We use some of the latest surface code developments to translate these gate counts into physical qubit number requirements and clock time estimates. The work presented in this chapter originally contributed towards the paper "How will quantum computers provide an industrially relevant computational advantage in quantum chemistry?" [12]. The paper is of a broad perspective type and involved collaboration with experienced classical-quantum chemists; in this chapter we take a pedagogical approach and focus more upon the work that I was personally involved with. This chapter will be a change of pace from the previous one, chapter 4, which was very hardware centric. In the following chapter, chapter 6, we will bring in a wider set of hardware considerations by investigating how the rate of physical-logical operations can impact the feasibility of reaching a quantum advantage.

5.1 Classical quantum chemistry

One of the fundamental goals of quantum chemistry is to solve the time-independent, non-relativistic Schrodinger equation for molecular systems. Classical computation power becomes more and more available every year and the algorithms for chemistry are improving rapidly, and yet on the fundamental level all classical techniques involve some degree of approximation to avoid the two prohibitively expensive features. These are: (i) storing the full many-body quantum wave function with 2^N entries and (ii) propagating the wave function in time by general matrix-vector multiplication. These features require exponential scaling in time and space on a classical computer and therefore approximations are inevitable.

To numerically simulate quantum chemistry problems a set of independent functions, known as a basis set, is introduced to describe the physical wave function of the system. This introduction does not remedy the exponential increase of parameters with system size but enables one to balance the achievable accuracy against the required computational resources. Richard Feynman was perhaps the first to envisage a quantum computer and its application to the simulation of physics and chemistry [4]. It is now expected that quantum computers will eventually be able to perform electronic structure calculations with the accuracy typical of rigorous classical methods and the run time associated with approximate methods.

A common simplifying assumption in both classical and quantum methods is the Born-Oppenheimer approximation (BOA) which treats nuclei as stationary point charges due to their relative size. Relativistic effects are also often neglected, particularly for small-medium sized molecules. The classical technique of exact diagonalization can provide an exact answer for the wave function and energy (i.e. the full configuration interaction (FCI) answer). However, the exact method comes at the cost of storing an exponential number of coefficients which means that it is prohibitive for even medium sized molecules and it is instead used mostly as a bench-marking method. The self-consistent field method (SCF), aka Hartree-Fock method, assumes that the exact wave function of the system can be approximated by a single Slater determinant and the method is often used as an ansatz for quantum algorithms. The coupled cluster technique extends the Hartree-Fock method by accounting for electron correlation by constructing multi-electron wave functions with an exponential cluster operator. The coupled cluster has been used to generate some of the most accurate calculations to date for small molecules. Density functional theory is a low accuracy, high speed method in which all of the degrees of freedom of the electronic system are integrated out, except for the density.

5.1.1 Precision vs accuracy

A commonly used standard of accuracy within the quantum chemistry field is known simply as chemical accuracy, for which the error must be no greater than 1 kcal/mol relative to the hypothetical exact energy or an experimental measurement [158]. This concept is distinct to that of precision which in this context is instead the computational error relative to a computational reference. For example, one can achieve chemical precision within a minimal basis set for a particular molecule, but even the exact answer within the minimal basis set would not necessarily be sufficient to achieve chemical accuracy. This misconception between accuracy and precision has appeared in several papers over the last few years [159, 160, 94, 161, 162, 163, 164, 165] within the quantum computing quantum chemistry field (as opposed to the classical computing quantum chemistry field). Chemical accuracy has been claimed at times when in reality the achievement is instead chemical precision relative to the FCI method within a very small basis set. The distinction is important because achieving chemical precision does not at all guarantee any useful predictive power, such as chemical reactivity.

In figure 5.1 we plot the number of spin-orbitals mapped to qubits (the size of the used basis set) for a number of actual quantum computing calculations done to date. The figure makes it clear that for these quantum computing demonstrations, the number of qubits used is considerably smaller than would be necessary to achieve chemical accuracy. The limited quantum computational resources are certainly a reason for this but the subsequent claims should reflect the achievements with regards to the meaning of accuracy and precision.

5.2 NISQ and fault tolerant techniques

In this section we highlight the differences between NISQ and fault tolerant approaches to quantum chemistry. As described in more detail in chapter 2, NISQ devices correspond to devices in which each physical qubit corresponds to a logical qubit. The limited coherence times imply a maximum achievable circuit depth, which can define the feasibility of running a particular algorithm. Fault tolerant devices have not yet been built and their realization is expected to take several years. There are large resource overheads associated with error correction techniques which means that a fault tolerant device will likely require upwards of 100,000 physical qubits for useful applications.

We will focus on the calculations of ground state energies because extensions to excited state energies and Hamiltonian dynamics follow directly from similar principles, for examples see Refs [166, 167, 168]. There are two main paradigms for performing energy estimation, Hamiltonian averaging (tomography), and Quantum Phase estimation.

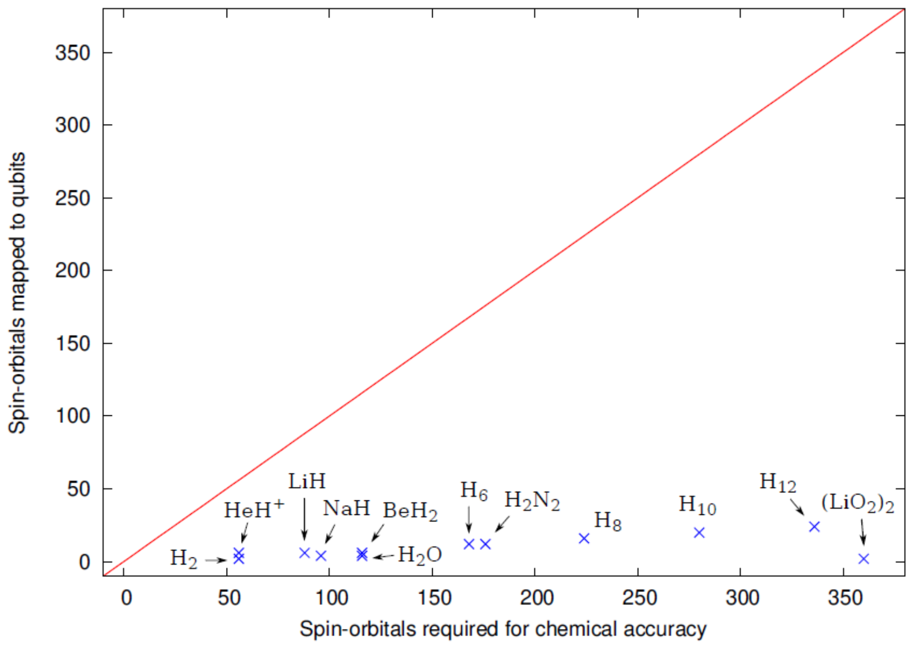


Figure 5.1: A comparison of the number of spin-orbitals required to achieve chemical accuracy for ground state energy calculations (red line) and the number of spin-orbitals which have been utilized in some actual quantum computing calculations conducted to date [12]. Here it is assumed that the cc-pVTZ basis set would be sufficient for chemical accuracy if used within an extrapolation scheme.

5.2.1 NISQ and VQE

The primary method of performing quantum chemistry on a NISQ device involves Hamiltonian averaging. In Hamiltonian averaging, first a quantum state is prepared over a qubit register and the Hamiltonian operator's expectation value is measured by sampling the qubits many times. The value can then be determined by averaging over all measurements. For an accuracy ϵ this procedure needs to be repeated as $\mathcal{O}(1/\epsilon^2)$ [166]. This results in an expectation value for a single operator within the Hamiltonian. To acquire the expectation value of the entire Hamiltonian this procedure needs to be repeated for every operator. Due to the linearity of the expectation operator the final answer is a sum of each individual result. In the second quantization, the general chemistry Hamiltonian has $\mathcal{O}(N^4)$ terms which plays a big part in defining the final run time [166]. There are some techniques to reduce this dependence [169, 170]. To acquire the ground state energy using this procedure it is necessary for the initial prepared quantum state (ansatz) to have sufficient overlap with the ground state. The variational principle [171] can be used to consistently produce the ground state, it states that the expectation value of the Hamiltonian for a prepared trial state is always an upper bound to the ground state energy. Using the variational principle in combination with Hamiltonian tomography is known as the Variational Quantum Eigensolver (VQE) [94] and it is the primary technique for chemistry simulation on NISQ devices. For more details on the NISQ-variational approach see section 2.4.

5.2.2 Fault tolerance and QPE

The Quantum Phase Estimation (QPE) algorithm generates eigenvalues for a general unitary operator and it can be applied to quantum chemistry to find the eigenenergies of chemistry Hamiltonians to FCI precision. Unlike VQE which involves many iterations ($\mathcal{O}(1/\epsilon^2)$ with accuracy ϵ) of low depth circuits, the QPE algorithm requires a $\mathcal{O}(1)$ iterations of a circuit with a depth scaling as $\mathcal{O}(1/\epsilon)$. The large depth required in the QPE algorithm means that it will only be possible with error corrected devices, because NISQ devices would lose their coherence long before the end of the circuit. For a given unitary operator $\hat{U}$, the eigenvalues can be represented as a phase $\lambda_j = e^{i\phi_j}$, and phase estimation can be used to extract the eigenvalues in an efficient manner. To utilize the QPE algorithm for quantum chemistry, Hamiltonian simulation is used as a subroutine, where for a Hamiltonian $\hat{H}$, the operator $\hat{U} = e^{-i\hat{H}t}$ is constructed. When using the standard QPE algorithm, the eigenenergy is stored as a binary approximation in a secondary qubit register where each ancilla qubit adds another digit of accuracy. The phase (which can be converted into an energy) is imprinted (via phase-kickback) onto the ancilla register by using the ancilla qubits for controlled operations onto the state preparation register repeatedly. To calculate the ground state energy it is necessary for there to be sufficient overlap with the state preparation register. If there is a superposition of multiple states in the preparation register then the measurement procedure will collapse it into a single eigenstate with probability corresponding to its overlap with the prepared state. The simple Hartree-Fock product state typically provides sufficient overlap for most systems, but unitary variants of the coupled-cluster methods offer a more accurate (and classically costly) alternative.

5.3 Hamiltonian simulation

Hamiltonian simulation involves constructing a quantum circuit which approximates the evolution of the input state with respect to a Hamiltonian $\hat{H}$, which corresponds to the action of the operator $e^{-i\hat{H}t}$. For arbitrary Hamiltonians the number of elementary gates needed to construct the time evolution operator grows exponentially with the number of qubits. In the pioneering work of Lloyd [172], a method for polynomial scaling simulation is shown for Hamiltonians with a special structure (ones which describe only local interactions), which laid the foundation for further research into Hamiltonian simulation [173, 174, 175]. Cao et al provide a detailed review of quantum computing and quantum chemistry including the sequential development of Hamiltonian simulation techniques [166]. A major research focus since Lloyd's work has been to identify special structure within quantum chemistry Hamiltonians to enable efficient simulation. There are two main paradigms for Hamiltonian simulation: product formula algorithms (Trotterization), and quantum signal processing algorithms (Qubitization).

5.3.1 Trotterization

There have been numerous developments on the theory of Trotter decompositions and the resulting error [176, 177, 178, 179]. One of the most favoured decompositions of this form are now known as the Trotter-Suzuki (TS) formulas [180]. The general idea behind using Trotter-Suzuki formulas is to first express the Hamiltonian as a sum of easier to simulate Hamiltonians and then the total evolution is approximated by the combination of these simpler evolutions [181], i.e. let $\hat{H} = \sum_{j=1}^{m} \hat{H}_j$ then the first order TS formula is

$$e^{-i\sum_{j=1}^{m}\hat{H}_j t} = \prod_{j=1}^{m} e^{-i\hat{H}_j t} + O(m^2 t^2), \tag{5.1}$$

The error here results from the fact that the exponential is actually an operator exponential and in general the $\hat{H}_j$ terms do not commute. The above formula has a negligible error for $t << 1$. To accurately simulate for larger times it is necessary to further break up the dynamics into a number (r) of shorter time steps i.e. each time step would run for a time t/r

$$e^{-i\sum_{j=1}^{m}\hat{H}_j t} = \left(\prod_{j=1}^{m} e^{-i\hat{H}_j t}\right)^{r} + O(m^2 t^2/r), \tag{5.2}$$

Higher-order formulas can be constructed and generally provide a more favourable approximation error at the cost of larger gate counts. The methods above rely on being able to efficiently simulate $e^{-i\hat{H}_j t}$ terms, which can be accomplished by defining the Hamiltonian as a sum of Pauli operators and the method to perform this is known as the Jordan-Wigner decomposition.

5.3.2 Qubitization

Qubitization [182, 183] is perhaps the most favored method for simulating quantum chemistry Hamiltonian dynamics. It achieves the provably optimal scaling in query complexity and approximation error albeit while requiring more logical qubits than other methods. Using a pair of reflection operations, the qubitization method first constructs an operator

$$W = e^{\pm \cos^{-1}(\hat{H} \,/\, |h|_1)}, \tag{5.3}$$

from a Hamiltonian of the form $\hat{H} = \sum_j h_j \hat{H}_j$. In the case of calculating the ground state energy and other static quantities, it is sufficient to apply phase estimation directly to the W operator and take the cosine of the result to convert it into the appropriate form. The reflection subroutines within Qubitization rely on each $\hat{H}_j$ being Hermitian and unitary, which fortunately applies to Pauli operators, making this method a good fit for quantum chemistry simulation.

5.4 Q# and resource estimation

Q# is Microsoft's software development kit for quantum computing, and its ultimate goal is to manage execution on large-scale quantum hardware. In the meantime it supports simulation on both short-term "NISQ" scale as well as large-scale fault tolerant devices. As complete simulation of large-scale devices is unachievable for classical computers, *Q#* offers resource estimation features, which allow for detailed constructions of various algorithms and the following gate counts. We used *Q#* to calculate the required number of logical operations to perform the Hamiltonian simulation with various methods, and we explain our methodology in the following section.

5.4.1 Methodology

The first step to use *Q#* for resource estimation in quantum chemistry [184] is to acquire a .YAML file (a human-readable data-serialization language comparable to XML) which contains information on the ansatz to be used within the QPE algorithm. There are some sample YAMLs provided by Microsoft and there are multiple methods to generate new ones. The *Q#* documentation recommends NWChem [185] which is most easily utilized through the docker image provided by PNNL (Pacific Northwest National Laboratory). We had some success using NWChem but eventually moved onto Python's Pyscf module to produce the one and two electron integrals with the self-consistent field method. We converted the integral data using the Broombridge schema 2.0 (YAML-based schema for $Q\#$), resulting in a YAML file with sufficient information to perform resource estimation.

There are numerous provided example programs within the $Q\#$ documentation. We predominantly used (and adapted) the "GetGateCount" file for this investigation. The YAML starts with the Hamiltonian in the second quantization formulation, which is then converted into a qubit Hamiltonian once it is passed into "GetGateCount" using the optimized Jordan-Wigner representation. The target machine of $Q\#$ is set to "CreateAndConfigureTraceSim" which functions as a resource estimator which we use to calculate the costs of a single oracle application. The "GetGateCount" program enables three oracle types for Hamiltonian simulation, which are Trotterization, Qubitization and Optimized Qubitization. By default the program provides resource estimation for T gates, Rotations, and CNOTs, as well as calculating the 1-Norm for the Qubitization oracles. To acquire information regarding the number of qubits utilized, the width counter must be enabled and called. Inputs to the Trotterization oracle are the trotter order, for which the oracle cost scales with exponentially, and the Trotter step size, for which the oracle cost is independent.

The program can be further altered to perform resource estimation on the full energy calculation by using a phase estimation algorithm such as robust phase estimation. The final gate count and accuracy are determined by the input parameter "bits of precision". This in turn sets the standard deviation on the calculated phase as $\sigma \leq 2\pi/2^{bitsPrec}$ and it effectively determines the number of applications (# of queries) of the oracle where the standard deviation satisfies $\sigma \geq 2\pi/\#$ of queries. There are more detailed optimizations that can be performed to determine the optimal number of queries which are specific to the type of oracle used and we will highlight this in more detail later.

5.4.2 Oracle costs for small molecules

In figures 6.2-6.4 we demonstrate the use of Q# for resource estimation. We plot the dependence of T gate count, CNOT count and logical qubit count for the three available Hamiltonian simulation oracle types, Trotterization, Qubitization, and Optimized Qubitization. These resource estimates correspond to a single application of the Hamiltonian simulation oracle as opposed to a full energy estimation which will come later. In figure 5.2 it can be seen that the T gate count of oracles from best to worst is Optimized Qubitization, Trotterization, and Qubitization. It is only when the 'Qubitization' oracle is optimized for chemistry with the techniques of Ref. [186], denoted as 'Optimized Qubit-

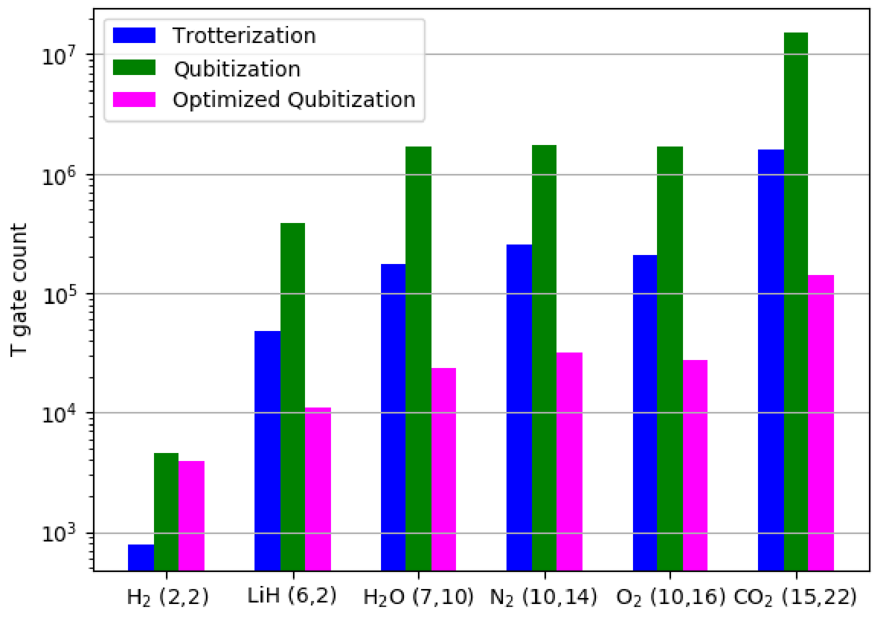


Figure 5.2: T gate cost for a single application of three different oracle types (Trotterization, Qubitization, and Optimized Qubitization), for 6 different small molecules within the minimal STO-3G basis set with orbital and electron number denoted in brackets by (orbitals,electrons).

ization', that the T gate costs are lower than that of Trotterization. In the surface code the T gate count is considered to be the determining factor of the algorithm length. This is because it must be produced via costly techniques such as magic state distillation as opposed to the natively available Clifford gate set. The Trotterization oracle does not directly use any T gates, it instead requires rotation gates which within the surface code need to be synthesized. Long sequences of the Hadamard gate and T gate can be used to synthesize an arbitrary angle rotation, where the chain length scales with the desired accuracy. Several efficient proposals exist and they typically have an upper bound on the cost of S in terms of the number of T gates of the form

$$S = \gamma_1 \log_2(1/\Delta_{synth}) + \gamma_2 \tag{5.4}$$

for a single rotation with error Δ_{synth}, and Ref. [187] found $\gamma_1 = 4$ and $\gamma_2 = 11$. In Q# the default value for Δ_{synth} is $0.001/R$, with number of rotations R, which is what we used to define the T gate cost for the Trotterization oracle.

Trotterization benefits from the lowest logical qubit requirement, which is equal to the spin orbital number of the molecule investigated. Optimized Qubitization has the highest logical qubit requirement which is in part a result of the T-gate optimization process for

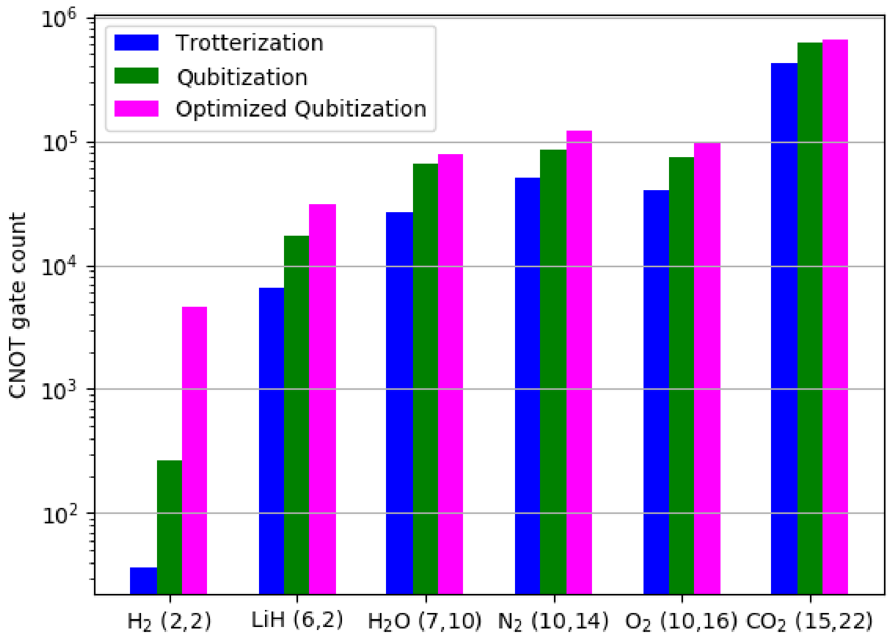


Figure 5.3: CNOT gate cost for a single application of three different oracle types (Trotterization, Qubitization, and Optimized Qubitization), for 6 different small molecules within the minimal STO-3G basis set with orbital and electron number denoted in brackets by (orbitals, electrons).

chemistry.

In figure 5.5 we plot the T cost dependence on orbital number for the H2O molecule in the def2-TZVP basis (which has a maximum of 48 orbitals) for Trotterization and Optimized Qubitization. We can see that regardless of the orbital number the Optimized Qubitization has a T-gate count approximately an order of magnitude lower than that of Trotterization. Without rigorous circuit optimization and at larger system sizes the T-gate oracle costs are directly proportional to the number of Jordan-Wigner terms, which scales as $O(N^4)$. The fits in figure 5.5 were calculated using the first 7 data points (up to orbital number 17) with a fixed form of $A\,N^4$ and show good correspondence to the final two data points at orbital number 48. This plot indicates that extrapolation schemes to higher orbital numbers can be reliable for resource estimation.

While the resource estimations using Q# presented so far may be qualitatively useful, they do not directly entail a wall-clock time (aka elapsed real time) or the required physical qubit numbers for energy estimation. There are three main reasons for this: first, the overall cost is determined not only by the oracle cost, but also by the number of its applications, the state-preparation, and the particular phase estimation scheme that is used. Second, the oracle construction input parameters are critical to the final estimates,

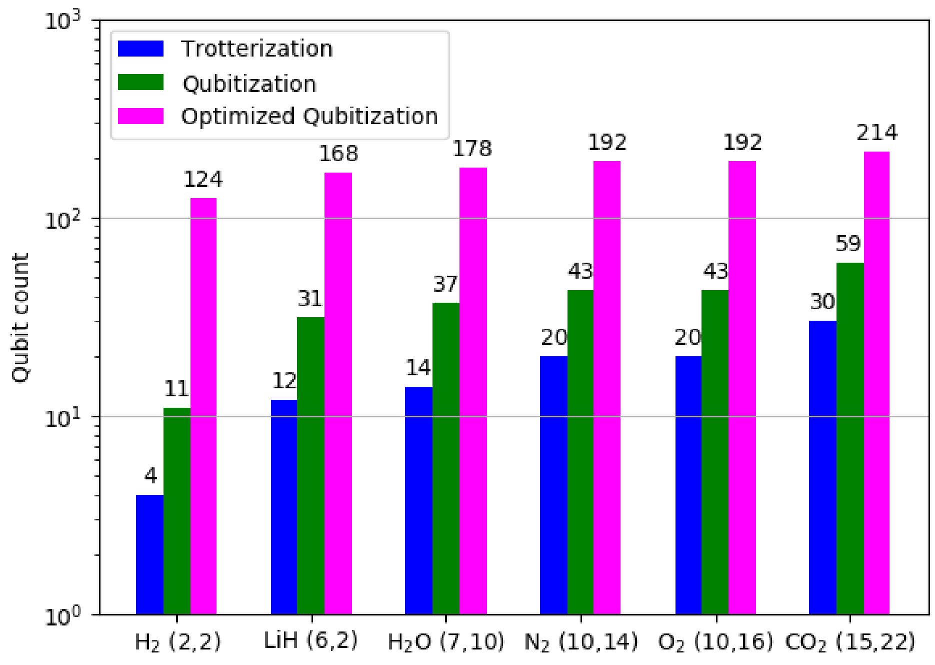


Figure 5.4: Qubit count required for a single application of three different oracle types (Trotterization, Qubitization, and Optimized Qubitization) for 6 different small molecules within the minimal STO-3G basis set with orbital and electron number denoted in brackets by (orbitals,electrons).

but they can not be determined trivially and they depend significantly on the specific simulated system Hamiltonian. For Trotterization these include the Trotter step size and Trotter order [188], and for Qubitization the optimal parameters depend on the Hamiltonian 1-norm and sparsity. Third, the impact of fault-tolerant error-correcting codes on the resource estimate is significant, i.e. orders of magnitude, and needs to be incorporated or added to the calculation. In the following two sections we fill in these gaps and then provide more rigorous resource estimates in terms of physical wall-clock time and qubit number.

5.5 Optimizing gate count and error budgets

In this section we will detail the process of optimizing gate counts for full energy estimation, and the methods unique to each oracle type. First we will briefly overview the variants of the phase estimation algorithm and their advantages. We will also motivate the choices of molecules and basis sets within the context of the limits of classical quantum chemistry.

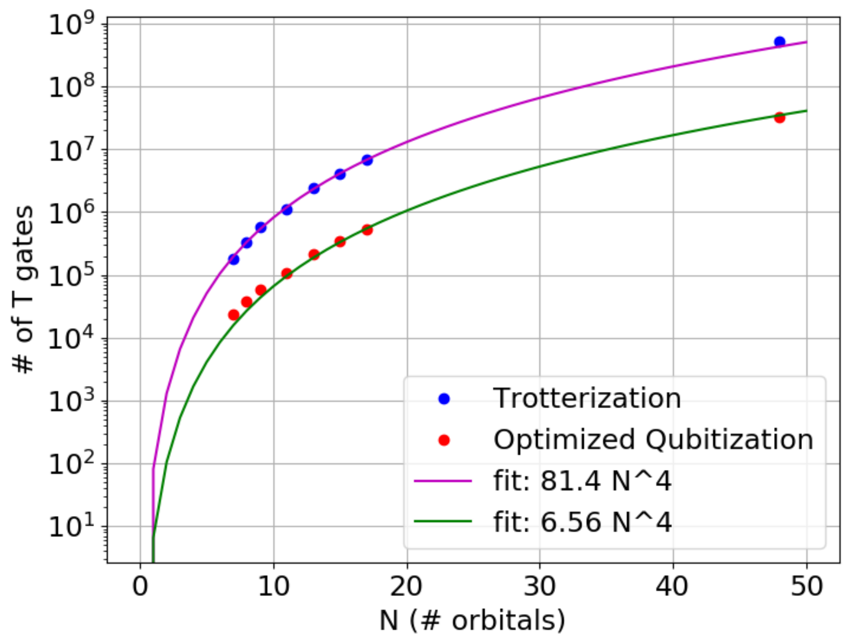


Figure 5.5: The number of T gates required for a single application of an oracle as a function of the number of orbitals in the basis set for H2O in the def2-TZVP basis set (which has a maximum of 48). Blue circles correspond to the Trotterization oracle with results calculated using Q#. Red circles correspond to the Optimized Qubitization oracle and were also calculated using Q#. The fits were calculated using the first 7 data points for each oracle with a fixed form of $A\,N^4$ and show good correspondence to the final data points at 48 orbitals.

5.5.1 Phase estimation variants

We gave a high level description of the phase estimation algorithm earlier in this chapter, corresponding to one of the initial proposals in which a register of k ancillary qubits is required to store the phase with precision dependent on k. In the iterative phase estimation algorithm [110] the ancillary register is replaced with a single qubit which is measured repetitively throughout the algorithm while maintaining the same precision. Bayesian inference techniques [189, 190] can be used to improve the query complexity of the QPE algorithm by extracting the maximum amount of information from each measurement (with the measurement procedure of the iterative approach). The classical processing associated with Bayesian inference has a large computational cost dependent on the desired accuracy, so that exact inference is in practice intractable. The limited accuracy required in quantum chemistry applications means that Bayesian inference methods are an available option. Unlike Bayesian inference methods, the Robust Phase estimation algorithm [191] uses classical post-processing that scales polynomially with the number of measurements making it a more feasible choice for many applications. Robust phase estimation uses only one qubit in the ancilla register and the procedure is non-adaptive, meaning the required sequence of gates is independent of the intermediate measurement outcomes. These features enables confident resource estimation through circuit construction (without execution).

5.5.2 Motivating which molecules to investigate

In following sections we provide resource estimates for two particular molecules. First we consider the ground state energy for the hydrogen molecule H_2 to chemical accuracy. Much of the quantum computing resource estimation work of recent years has focused on Hydrogen and at times the achievement of chemical accuracy has been claimed, when in reality it was chemical precision within the small basis sets (e.g. STO-3G), see figure 5.1. 'STO-nG' are the minimal basis sets where n Gaussian functions are fitted to represent the behaviour of each electron orbital. We aim towards actual chemical accuracy by investigating a successively increasing basis set size for Hydrogen, in the order of STO-3G, 3-21G, 6-31G, cc-pVTZ and cc-pVQZ which have a corresponding orbital numbers of 2, 4, 10, 28, 60. 'cc-p' stands for correlation-consistent polarized and this basis set type is considered the current state of the art for post-Hartree-Fock calculations, the following 'V' indicates they are valence-only basis sets, and the following 'DZ', 'TZ', 'QZ' correspond to double, triple, and quadruple -Zeta and indicate the number of basis functions in increasing order.

We also consider the ground state energy of the chromium dimer, Cr_2, which represents a serious challenge for classical many-body electronic structure methods. We expect the chromium dimer to be a good candidate for a quantum algorithm to achieve a higher accuracy than has been possible with classical algorithms, because of its size and the significant non-dynamic correlation. Instead of increasing the basis set size we use a so-called Complete Active Space (CAS) approach, where the number of active orbitals and active electrons within the large cc-PVTZ basis are sequentially increased. In both the case of Hydorgen and the CAS Chromium dimer, the Hamiltonian matrix elements were calculated with the RHF-SCF method [192]. For further comparison using the Chromium dimer we constructed a denser starting Hamiltonian by performing CASSCF and using the final rotated-basis. This is a better indicator of the final CI steps of a CASSCF calculation.

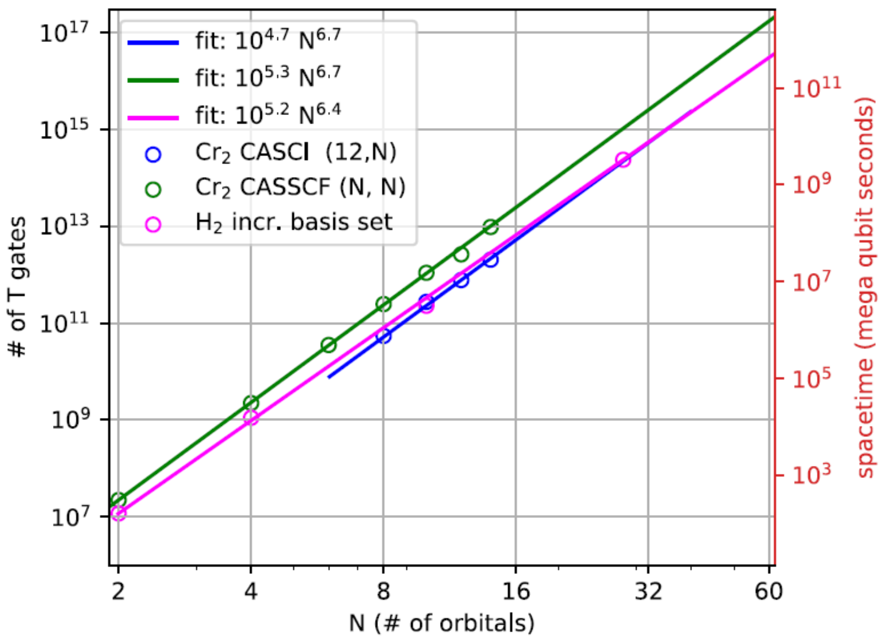


Figure 5.6: The number of T gates required for ground state energy calculations using the Trotterization method, for Cr_2 in increasing CAS size and H_2 in increasing basis set size to chemical accuracy within that basis, as a function of the number of orbitals [12]. The cost has been optimized individually for each data point. The space-time cost as a function of the number of T gates is shown, where we assume a space-time complexity of 14 qubit-seconds per T gate [121]. Qubit-seconds refers to the space-time volume cost, in the form of number of physical qubits required $\times$ seconds.

5.5.3 Trotterization

In this section we provide resource estimates for ground state energy calculation for hydrogen and the chromium dimer. We detail the optimization process that was performed to arrive at these results, which was led by co-author Dr. Vincent Elfving and involved utilizing the oracle cost results from $Q\#$ in combination with the methods of Ref. [188]. We consider the class of iterative phase estimation techniques which only require a single ancilla logical qubit in addition to the $2N$ logical qubits required by the Trotterization oracle. We aim to minimize the total T gate count because this effectively determines the wall-clock time (aka elapsed real time) within the surface code paradigm [193], and we will explain the overheads associated with the surface code in more detail in a following section. As mentioned previously the Trotterization oracle is decomposed into only rotation gates and CNOT gates, and the total T gate count arises from the synthesis of those rotation gates. We use the same derivation for Trotter resource costs as in Ref [188] (for more details see their Supplementary material), in combination with the oracle costs generated from Q#. There are three additive sources of error within this scheme, and we desire their sum ϵ_t to be less than the value of chemical accuracy (0.1 mHartree),

$$\epsilon_P + \epsilon_{TS} + \epsilon_S \leq \epsilon_t := 10^{-4} Ha \tag{5.5}$$

with the error from phase estimation, ϵ_P, the error from the Trotter-Suzuki decomposition, ϵ_{TS}, and the error from rotation gate synthesis, ϵ_S. The total T-gate cost, C, as a function of these errors is drived in Ref. [188] and is,

$$C = 2M\left\lceil\frac{\alpha}{\epsilon_P}\right\rceil\left\lceil\beta\sqrt{\frac{\epsilon_t}{\epsilon_{TS}}}\right\rceil\left(\gamma_1 \log_2\left(\frac{2M}{\epsilon_S}\left\lceil\beta\sqrt{\frac{\epsilon_t}{\epsilon_{TS}}}\right\rceil\right) + \gamma_2\right). \tag{5.6}$$

The minimal T-gate count can then be found by solving the constrained optimization problem. As in equation 5.4, γ_1 and γ_2 relate to the T cost scaling for rotation gate synthesis as a function of error per rotation gate, and we choose $\gamma_1 = 4$ and $\gamma_1 = 11$ as found in Ref. [187]. As in [188] we let $\alpha = \pi/2$ which relates to the scaling of the phase estimation with the desired accuracy. The Trotter number, β, would ideally be calculated by a full extrapolation simulation to determine the threshold Trotter number given a desired error on the Trotter-Suzuki decomposition but we use the same factor as Ref. [188] $\beta \approx 0.0178N^{2.5}$ for a calibration error equal to chemical accuracy. The factor of $2M$ comes from the number of exponentials in the Trotter-Suzuki decomposition. We found that in most cases the relative ratios of the errors in equation 6.5 are approximately $20 : 10 : 1$ corresponding to the following errors respectively, ϵ_P, ϵ_{TS}, ϵ_S. This hierarchy is due to the differing costs to reduce these errors as a function of the increase in T gate count. For example the T-cost of gate synthesis has only a logarithmic dependence on the synthesis error, therefore it can be suppressed more cheaply.

In figure 5.6 we plot the T gate cost as a function of the number of orbitals for H_2 and Cr_2. We also show the resulting space-time cost from a surface code error correction perspective, of 14 qubit-seconds per T gate. Qubit-seconds refers to the space-time volume cost, in the form of number of physical qubits required $\times$ seconds, which is applicable due to the flexible space-time trades that can be made through parallelization. In a following section we will provide more rigorous error correction scheme optimizations for determining the wall-clock time and the required number of physical qubits. In this figure we can see that the total T gate count (which is directly related to the wall-clock time) scales polynomially with orbital number. Comparable accuracy methods on a classical computer scale exponentially with orbital number.

5.5.4 Qubitization

In this section we describe the method that was performed to acquire the logical algorithm requirements for ground state estimation with Qubitization, which was carried out by co-author Dr. Vincent Elfving, using the methods from Ref. [183] and applying them to the particular molecules in question. In Ref. [183] an efficient molecular chemistry simulation method was proposed, which involves performing phase estimation on a quantum walk generated using qubitization oracles [182]. The qubitization paper presents several variants on the method, but in this work we focus on the sparse method. The sparse method includes a large overhead in the number of logical qubits, but it is optimized for the T gate complexity.

The sparse method relies first on reducing the number of contributing terms in the Coulomb operator within the Hamiltonian, down from the normal assumption of $L = N^4$. This is done by setting a cutoff threshold c, meaning that if an element within the Coulomb operator is smaller than c then it is set to 0. The aim is to set c as large as possible while maintaining a sufficient representation accuracy, which can be done by incrementally increasing c from zero and classically calculating the ground state error, until the error surpasses chemical accuracy. For the small basis sets ($N < 10$) we performed FCI calculations, while for the larger basis sets we chose the more computationally efficient Møller-Plesset second order perturbation theory method (MP2). Once the best value of c is ascertained one must recalculate the Hamiltonian 1-norm which is an important parameter for the qubitization method. The qubitization method relies mostly on Toffoli gates (which can be synthesized from T gates). We follow the methods of Ref. [183] and give our results for the Toffoli gate dependence on orbital number in figure 5.7. We include the space-time estimation as well and by comparing figure 5.6 and figure 5.7 we can see that the Qubitization method has a considerably lower space-time cost (approximately 4 orders of magnitude). In figure 5.8 we plot the logical qubit requirement for both Trotterization and Qubitization, and we can see that the Qubitization method does indeed require significantly more logical qubits (upwards of an order of magnitude).

In the following section we will provide more details on the error correction techniques, and the choices and optimizations that need to be made for rigorous estimates.

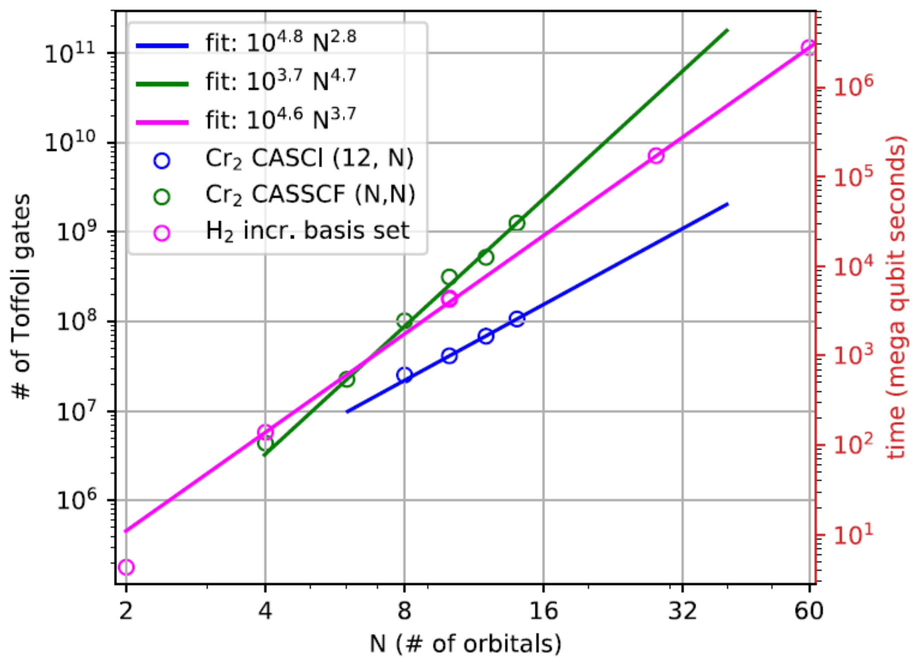


Figure 5.7: The number of Toffoli gates required for ground state energy calculations, using the Qubitization method, for Cr_2 in increasing CAS size and H_2 in increasing basis set size to chemical accuracy within that basis, as a function of the number of orbitals [12]. The cost has been optimized individually for each data point. The space-time cost as a function of the number of Toffoli gates is shown, where we assume a space-time complexity of 24 qubit-seconds per T gate [121]

.

5.6 Error correction and physical resource estimation

The quantum algorithms we have investigated above have a favourable scaling with the chemical system size. However there is still a significant pre-factor which makes the total gate count large even for small-medium systems. To compare these methods to classical run times, there is an even larger overhead associated with quantum error correction that must be taken into account. There is a substantial body of work covering various fault-tolerant strategies, here we have considered the lattice surgery method within the surface code paradigm [193, 121, 14].

The surface code relies on 2D nearest-neighbour qubit connectivity, which is well suited to the superconducting qubit quantum-processors which have a relatively high cost for performing distant two qubit gates.

An important universal gate set is the Clifford + T set ($T = diag(1, e^{i\pi/4})$). The Clifford gates are generated by the Hadamard (H), CNOT, and $S = diag(1, e^{i\pi/2})$ gates.

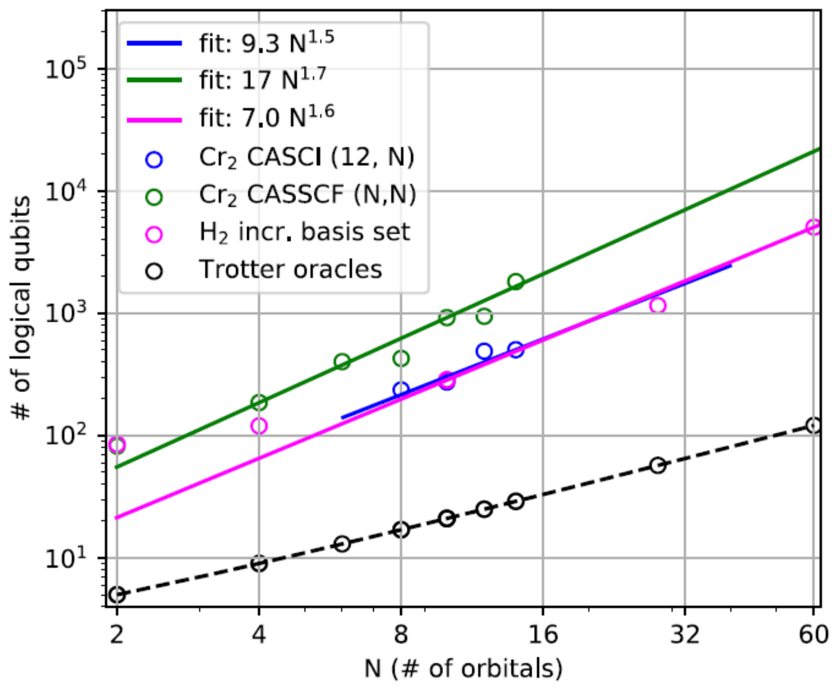
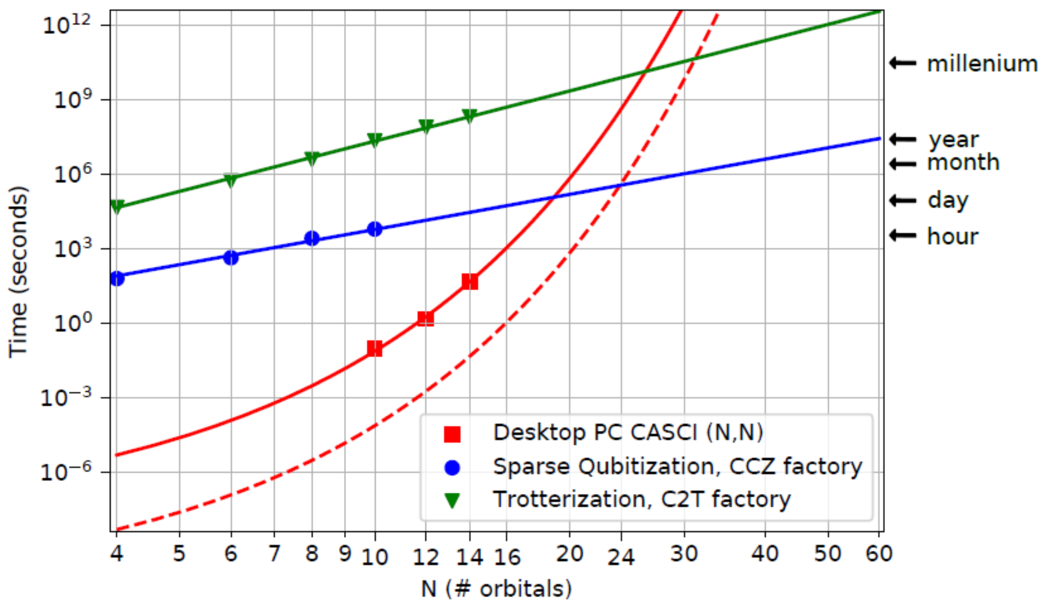


Figure 5.8: The number of logical qubits required for ground state energy calculations as a function of the number of orbitals [12]. For Cr_2 in increasing CAS size and for H_2 in increasing basis set size, calculated to chemical accuracy (precision) within that basis. The cost has been optimized individually for each data point. The Trotter oracle qubit requirement is solely dependent on the orbital number (2N+1) and is shown in black.

The surface code has transversal (or low-overhead) access to the Clifford gate set but not the T gate, where transversal pragmatically means easy or cheap to apply. A transversal gate implies that each qubit in a code block is acted on by at most a single physical gate and each code block is corrected independently when an error occurs. The Clifford S gate is not available transversely in the surface code but may be made available with relatively low overhead code manipulation techniques [8]. As the T-gate is not transversal it has to be generated with other (more costly) methods, the best known of which is magic state distillation. The high cost associated with magic state distillation means that the T-gate count alone effectively determines the final run time of an algorithm. To produce the following physical resource estimates we use lattice surgery based methods, which among other improvements reduces the space-time cost of distillation protocols by up to 90% in comparison to braiding-based implementations [14].

The error corrected quantum computer is partitioned into two sections: the data blocks, the purpose of which is to effectuate T gates by consuming magic states, and,



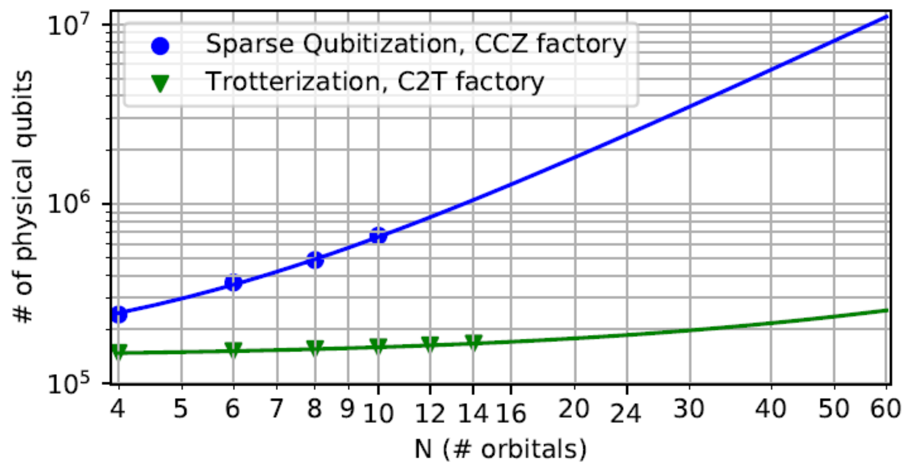


Figure 5.9: Wall-clock time scaling (top) and total physical qubit requirements (bottom) as a function of the number of orbitals for ground state energy calculations on the chromium dimer CR_2 CASCI (N,N), using both sparse Qubitization and Trotterization [12]. Classical run time is included for comparison with the full red line corresponding to a an Intel i9-10980XE with ∼1.2 TFLOPS, and dashed red line corresponding to a $10^5\times$ faster extrapolation (a top-5 HPC at ∼125PFLOPS). Lines represent curve fitting while markers represent numerical instance-specific data.

the distillation blocks, which produce the high fidelity magic states. One generally wants to match the rate at which the T gates are effectuated to the rate of magic state production. This can be performed by varying the number of distillation blocks, or the type of data block.

We first focus on a minimal physical qubit scenario (i.e. space optimized at the cost of time), and evaluate the total run time for performing a CASCI (Complete Active space Configuration Interaction) (N,N) simulation of the chromium dimer. For Qubitization we use the same sparse protocol discussed above, and fix the code distance at $d = 31$ (which should be sufficient for hardware error rates up to 10^{-3}). Note that this code distance should be fine tuned given a non-Clifford gate count requirement, which we do perform later in this section. We assume a CCZ factory [121] (which produces a magic state that can be consumed to effectuate Toffoli gates) produces a state every $5.5d$ cycles. This would correspond to about $170\mu s$ when we assume a $1\mu s$ code cycle (which may correspond to future superconducting devices). For the Trotterization protocol we assume the same code distance, and a C2T catalyzed T gate factory from Ref. [121] which produces T states in 6.5d code cycles.

In figure 5.9 we plot wall-cock time and physical qubit estimates for performing CASCI (N,N) simulation of the chromium dimer for both the Trotterization and Qubitization approaches. We include an estimation for classical run times with processing speeds at 1.2 TFLOPS (corresponding to a desktop PC) and 125PFLOPS (corresponding to a top-5 HPC). The quantum computing approach begins to outperform the classical approach at a CAS size of around $N = 19-32$, which corresponds to a physical qubit count of $\sim 10^5$ for Trotterization and $\sim 3\times10^6$ for Qubitization. In the case of the Trotterization algorithm, the crossover point occurs at a total run-time approaching a thousand years. The T gate complexity of the Trotterization approach make it an unappealing choice relative to the Qubitization method here, but further algorithmic improvements may be possible.

5.6.1 Detailed estimate for Cr_2 CASCI(26,26) and methodology

In this section we will provide a more rigorous estimate for Cr_2 CASCI(26,26), which appears to be on the threshold for staying infeasible for the foreseeable future on classical supercomputers. We will first explain some of the available choices to make with regards to the error correction and distillation schemes. The following represents a practical summary of the key takeaways for performing resource estimation from Litinski's Game of Surface Codes [14]. We do not motivate the size, speed, and quality of the various objects presented (data blocks and distillation blocks), for more information on this see Ref. [14]. In section 6.2.7 we provide a more detailed overview of the Game of Surface Codes method and how to optimize for time beyond what is presented in this chapter.

The code distance d, refers to the number of errors that are required to convert one logical state into another and it is a measure of the degree of protection provided by the code. In the surface code each logical qubit consists of d^2 data qubits and d^2 ancilla qubits, for a total of $2d^2$ physical qubits per logical qubit, and this square arrangement of physical qubits is sometimes referred to as a tile. Larger constructions like distillation blocks and data blocks can be constructed from a certain number of tiles arranged in a particular way. The base operation in the surface code is referred to as the code cycle, and it involves measuring all of the stabilizers via 4 controlled operations between the nearest neighbour data and ancilla qubits. The syndrome (error information) extraction process is imperfect because it consists of error prone physical operations, and so it must be repeated a number of times before one can be confident on the likely locations of errors. For a tile with code distance d, the code cycle must be repeated d times before corrections can be applied, and this quantity, d code cycles, is referred to as a time step, or sometimes as a "beat".

Data blocks

There are different data block types that we can choose based on our priorities with regards to time or space optimization [14]. The first quantum computers to have the required number of physical qubits to run fault tolerant algorithms will presumably be space limited at first, but depending on the scalability of the design, it may quickly become preferable to trade space (physical qubits) for time optimizations.

The Compact data block encodes N qubits in $1.5N + 3$ tiles. The compact data block can consume a magic state every 9 time steps, where each individual time step, requires d, code cycles. The code cycle is the basic operation of the surface code and its duration is determined by sum of the required quantum hardware operations, which includes single and two qubit gates, and quantum measurements. The code cycle may vary by orders of magnitudes between different quantum computing platforms.

The Fast data block encodes N qubits in $2N + \sqrt{8N} + 1$ and can consume a magic state every 1 time step. There are other data block types available, but for our purposes we will only consider these two.

Distillation schemes

Magic state distillation involves converting many low fidelity magic states into fewer higher fidelity states. There are numerous distillation protocols, each of which varies with regards to the output fidelity, required number of tiles, and rate of production. We will cover three distillation protocols here but it is to be noted that many more exist, and they can also be combined to form new strategies. Each high fidelity magic state can be consumed within the data blocks to effectuate one T gate.

The 15-1 distillation block outputs a magic state with an error of leading order $35p^3$, as a function of the base physical error rate p. It requires 11 tiles and 11 time steps per produced magic state. The 116-12 outputs a higher fidelity state at the cost of more tiles, with an error of $41.25p^4$, 57 tiles and 8.25 time steps per magic state. The 225-1 block is a two level approach and it is constructed by using 15 instances of the 15-1 bock as the input into a final 15-1 block, and outputs a state with error $1500625p^9$, requiring 176 tiles and 15 time steps. One must consider the probability of success P which is equal to $(1-p)^N$, which effectively reduces the rate of production by a factor $1/P$ (where N here is the number of input states, e.g. 15, 116). The code distance associated to the distillation factories may be calibrated separately to the data blocks, this is a technique we make use of in chapter 6, furthermore this method and a range of distillation factories are presented in Ref. [27].

Steps for determining the error correction overhead

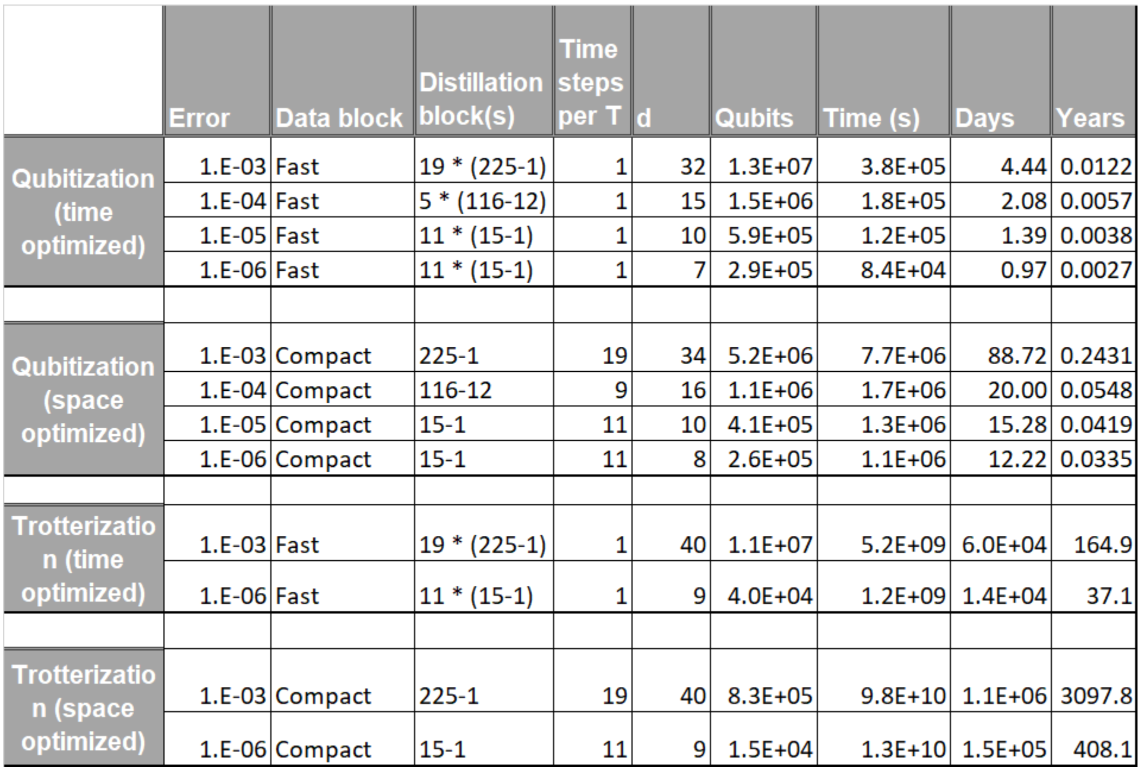
Using the results of figures 5.6-5.8 we estimate that for a CAS size of N=26 on Cr_2, the Trotterization protocol requires 53 logical qubits and $1.3 \cdot 10^{14}$ T gates, and the sparse Qubitization protocol (assuming 4 T gates can synthesize a single Toffoli gate) requires 1366 logical qubits and $1.2 \cdot 10^{10}$ T gates.

The first step to determine the error correction overhead for these algorithms is to choose a distillation protocol which is capable of producing a magic state with a satisfactory error. This is the case when the error is some degree lower than $(1/T_{count})$, with T_{count} being the total number of T gates in the algorithm. We set the acceptable error to $1/(T_{count} \times 100)$ which implies that the probability of a T error throughout the entire algorithm is 1%.

The minimal qubit setup will then consist of the compact data block in combination with the appropriate (single) distillation block. The bottleneck in T gate production should be identified between the data and distillation blocks, which will define how long the algorithm takes to run. We can optimize for time rather than space by choosing the fast data block and running multiple distillation blocks in parallel, enough to match the 1 T gate per time step rate of the fast block.

The next step is to determine the appropriate code distance. The logical error rate per logical qubit per code cycle can be approximated [25] as :

Table 5.1: Error correction strategy and resource estimates for Cr_2 (26,26)



	Error	Data block	Distillation block(s)	Time steps per T	d	Qubits	Time (s)	Days	Years
Qubitization (time optimized)	1.E-03	Fast	19 * (225-1)	1	32	1.3E+07	3.8E+05	4.44	0.0122
	1.E-04	Fast	5 * (116-12)	1	15	1.5E+06	1.8E+05	2.08	0.0057
	1.E-05	Fast	11 * (15-1)	1	10	5.9E+05	1.2E+05	1.39	0.0038
	1.E-06	Fast	11 * (15-1)	1	7	2.9E+05	8.4E+04	0.97	0.0027
Qubitization (space optimized)	1.E-03	Compact	225-1	19	34	5.2E+06	7.7E+06	88.72	0.2431
	1.E-04	Compact	116-12	9	16	1.1E+06	1.7E+06	20.00	0.0548
	1.E-05	Compact	15-1	11	10	4.1E+05	1.3E+06	15.28	0.0419
	1.E-06	Compact	15-1	11	8	2.6E+05	1.1E+06	12.22	0.0335
Trotterization (time optimized)	1.E-03	Fast	19 * (225-1)	1	40	1.1E+07	5.2E+09	6.0E+04	164.9
	1.E-06	Fast	11 * (15-1)	1	9	4.0E+04	1.2E+09	1.4E+04	37.1
Trotterization (space optimized)	1.E-03	Compact	225-1	19	40	8.3E+05	9.8E+10	1.1E+06	3097.8
	1.E-06	Compact	15-1	11	9	1.5E+04	1.3E+10	1.5E+05	408.1

$$p_L(p,d) = 0.1(100p)^{(d+1)/2} \ . \tag{5.7}$$

To keep the probability of a single logical error below 1% we need a code distance, d, that satisfies the following equation

$$N_L \cdot TS_T \cdot T_{count} \cdot d \cdot p_L(p,d) \leq 0.01 \tag{5.8}$$

with N_L, the number of logical qubits (data blocks and distillation blocks), TS_T, the number of time steps per effectuated T gate. Finally, the physical resource estimates can be calculated: the number of physical qubits is equal to $N_L \cdot 2d^2$ and the number of code cycles is equal to $d \cdot TS_T \cdot T_{count}$. In this section we assume a code cycle time of $1\mu s$, which is an estimate for a future superconducting architecture [14].

Results for Cr_2 CASCI(26,26)

We aggregate our results in table 5.1 for Cr_2 (26,26) for both Qubitization and Trotterization and follow either a time optimization or a space optimization strategy. For a physical error of 10^{-3} the qubitization protocol finishes $\sim 10^4$ faster while requiring a comparable number of physical qubits. We showed previously that at the logical level, the Qubitization protocol requires both considerably less T gates and considerably more logical qubits. However the difference in the required number of physical qubits between the two approaches is minimized at the error corrected level, due to the lower T-gate cost of the Qubitization approach, allowing for a lower code distance.

We plot the required time as a function of inverse physical gate error in figure 5.10 for the Qubitization protocol. We include the classical run time for a top 5 classical HPC (125PFLOPS) for comparison which was estimated using the same extrapolation technique of figure 5.9. For a physical error of 10^{-3} the time-optimized protocol finishes by a factor $\sim 20\times$ faster than the space-optimized protocol, while for the other error values it is by a factor $\sim 10\times$. We also plot in figure 5.11 the required number of physical qubits for the same scenario as above. We can see that the relative reduction in qubit number is most notable for low error rates. At an error of 10^{-3} the space-optimized approach uses a factor $2.5\times$ less physical qubits, while at an error of 10^{-6} the factor is only $1.07\times$ less. Using both of these figures we can conclude that the time optimized approach generally has a lower ($\sim 8\times$) space-time volume , and should be the favoured protocol when the number of physical qubits is not strictly limited. The degree of scalability of the underlying physical architecture may play a role in determining whether it is feasible to run the time-optimized approach.

In the following chapter we will focus in more detail on the surface code physical resource estimation process and investigate the extent to which slower hardware code cycle times can be mitigated by utilizing additional qubits.

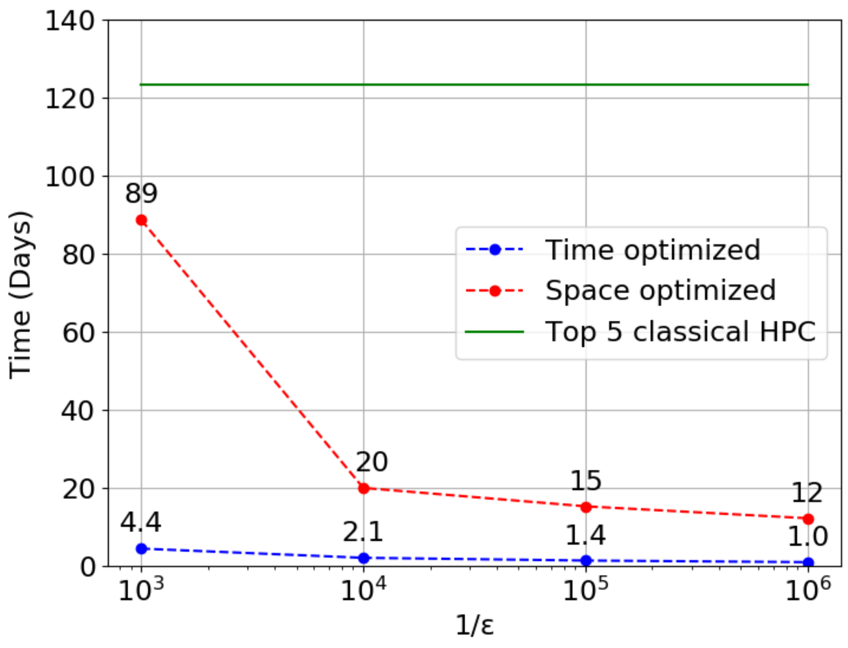


Figure 5.10: The required time (in days) for a ground state energy calculation, for Cr_2 at a CAS size of (26,26) to chemical accuracy with the sparse Qubitization algorithm, all as a function of inverse physical gate error. We include the required time to solve this problem on a top 5 HPC with ∼125PFLOPS (using the same extrapolation from figure 5.9)

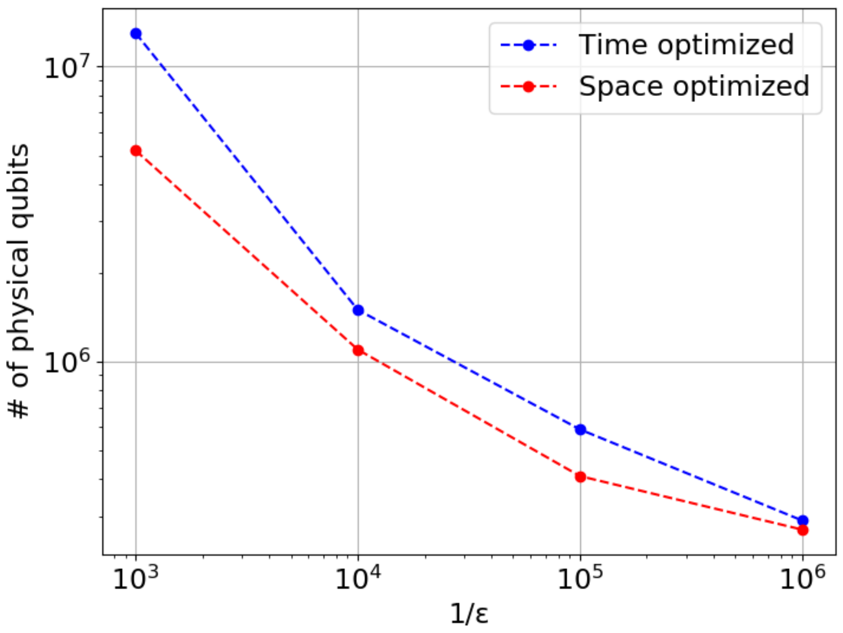


Figure 5.11: The required number of physical qubits for a ground state energy calculation, for Cr_2 at a CAS size of (26,26) to chemical accuracy with the sparse Qubitization algorithm, all as a function of inverse physical gate error.

Chapter 6

The Impact of Hardware Specifications on Reaching Quantum Advantage in the Fault Tolerant Regime

The following chapter closely follows the paper titled "The Impact of Hardware Specifications on Reaching Quantum Advantage in the Fault Tolerant Regime" [13] which at the time of writing is in the process of journal submission.

We investigate how hardware specifications can impact the final run time and the required number of physical qubits to achieve a quantum advantage in the fault tolerant regime. Within a particular time frame, both the code cycle time and the number of achievable physical qubits may vary by orders of magnitude between different quantum hardware designs. We start with logical resource requirements corresponding to a quantum advantage for a particular chemistry application, simulating the FeMoco molecule, and explore to what extent slower code cycle times can be mitigated by using additional qubits. We show that in certain situations architectures with considerably slower code cycle times will still be able to reach desirable run times, provided enough physical qubits are available. We utilize various space and time optimization strategies that have been previously considered within the field of error-correcting surface codes. In particular, we compare two distinct methods of parallelization, Game of Surface Code's Units, and AutoCCZ factories, both of which enable one to incrementally speed up the computation until the reaction limited rate is reached. Finally we calculate the number of physical qubits which would be required to break the 256 bit elliptic curve encryption of keys in the Bitcoin network, within the small available time frame in which it would actually pose a threat to do so. It would require approximately 317 million physical qubits to break the encryption within one hour using the surface code, a code cycle time of 1 μs, a reaction time of 10 μs, and physical gate error of 10^{-3}. To break the encryption instead within one day it would require 13 million physical qubits.

6.1 Introduction

With the advent of quantum computers, the race to a quantum computational advantage has gained serious traction in both the academic and commercial sectors. Recently, quantum supremacy has been claimed [194, 195] on quantum devices with tens of qubits. However, the targeted problems solved were theoretical in nature, and not relevant industrial applications. Quantum advantage, conversely, is a stronger form of supremacy that shows an industrially relevant computational problem solved in a reasonable time-frame that would be practically impossible to do using any classical supercomputer. There is a large physical qubit overhead associated with quantum error correction, which is required to run some of the most powerful algorithms. There are many factors which will determine the ability for different quantum computing architectures to scale. Consequently, within a given time frame the maximum size (qubit count) of a device could vary by orders of magnitude. The code cycle time (base unit of operation in the surface code) will also vary considerably between platforms. Therefore it is of interest to investigate the interplay between the code cycle time and the number of achievable qubits, and the resulting impact on the feasibility for a particular device to achieve a quantum advantage. We calculate the physical qubit and run time requirement for relevant problems in chemistry and cryptography with a surface code error corrected quantum computer, comparing parameters typical to various types of hardware realizations.

Algorithms which are tailored to NISQ devices generally consist of a hybrid approach, where a low depth circuit is parameterized, and iterated through a classical optimizer. These NISQ algorithms are more often heuristic in nature than their fault tolerant counterparts and so providing rigorous resource estimates can be more challenging. Many of the most powerful quantum algorithms require a circuit depth which greatly exceeds the capabilities of NISQ era devices, and for some applications the number of required logical qubits and operations are known. The quantum threshold theorem states that a quantum computer using error correction schemes, and a physical error below a certain threshold, can suppress the logical error rate to arbitrarily low levels [15, 17, 16]. Therefore one could run an algorithm with an arbitrarily long circuit depth provided enough qubits are available to perform the required level of error correction. There is a large time overhead for performing logical operations at the error corrected level relative to operations performed on physical qubits. For the foreseeable future, classical computers will have a clock rate that is orders of magnitude faster than error corrected quantum computers. To determine the problem size at which a quantum computer will outperform a classical computer one must consider both the algorithmic speed up as well as the relative difference between their associated clock rates. By making use of parallelization schemes, quantum computers can speed up the effective rate of logical operations at the cost of additional qubits, and so the ability to scale to large enough device sizes will also play a role in determining the feasibility of reaching desirable run times.

The surface code [196, 197, 198] is the most researched error correction technique, particularly in regards to end-to-end physical resource estimation. There are many other error correction techniques available and the best choice will likely depend on the underlying architecture's characteristics, such as the available physical qubit-qubit connectivity. Superconducting qubits are one of the leading quantum computing platforms and these devices generally consist of static qubits arranged on a grid where only nearest neighbour interactions are natively available. Long distance connections must be enabled by sequences of nearest neighbour swap operations, which in the context of a NISQ device may limit their computational power [20, 11]. The limited connectivity of superconducting qubits in part motivated the continued research into the surface code which relies only on nearest neighbour interactions between physical qubits arranged on a 2D grid. In the following we briefly introduce some of the alternative error correction techniques to the 2D surface code. Error correction codes which rely on global interactions at the physical level have favourable encoding rates as a function of code distance [30] but enabling this global connectivity on large devices may be challenging, or the connectivity overheads may outweigh the benefits relative to closer-range-connectivity codes. Entanglement distribution may be a viable method of enabling distant connectivity for large scale devices with limited physical connectivity in the limit of large reserves of quantum memory [199]. Higher dimensional error correction codes can have access to a greater range of transversal gates [154, 155] which may considerably improve final run-times, where transversal implies that each qubit in a code block is acted on by at most a single physical gate and each code block is corrected independently when an error occurs. Realising this 3D (or greater) physical connectivity could be challenging for many of the current quantum platforms; photonic-interconnected modules may be the most flexible architecture with regards to its possible connectivity graph [200], however, currently achieved connection speeds would present a considerable bottleneck [82]. A variant of the 3D surface code may still be realisable with hardware that is only scalable in two dimensions because the thickness (extra dimension) can be made relatively small and independent of code distance [201]. In section 6.2 we highlight the surface code in more detail and include relevant considerations for physical resource estimation.

Here we highlight some of the leading quantum computing platforms, their relevant error correction strategies, and contrast their rate of operations. The surface code is the front-running proposal for error correction in superconducting devices and their associated code cycle times (the base sequence of hardware operations) have been estimated to be in the range of 0.2 μs-10 μs [193, 14]. There is a great variety within different implementations of trapped ion architectures, particularly with regards to the method of enabling connectivity. A proposed scalable trapped ion design that relies on shuttling to enable connectivity, and microwave based gates has estimated the code cycle time to be 235 μs [74]. For this shuttling based design alternative error correction protocols may be more effective than the surface code due to the variable-mid range connectivity that is possible, but in this work we constrain ourselves to the surface code. Small trapped ion modules connected via photonic interconnects have been envisaged with the surface code [83], but due to their extremely flexible connectivity, higher dimensional error correction codes may one day be utilized. The code cycle time for trapped ions with photonic interconnects would depend on how the physical qubits are distributed across modules; one approach advocates for two tiers of encoding to minimize the use of the slower interconnects [202]. A fault tolerant Silicon based architecture has been proposed using the surface code with code cycles estimated to be 1.2 ms [203]. The error correction choice for photonic devices will depend on the underlying design; the primary candidate for a particular fault tolerant proposal [204] is the RHG lattice [205, 206]. The degree to which an architecture is scalable will vary greatly between architecture types and within particular stages of development.

In this work we provide physical resource estimates to achieve a quantum advantage with a quantum computer using the surface code. Using some of the latest time optimization strategies [14, 113, 207], we investigate the interplay between the code cycle time and the number of achievable physical qubits in the underlying architecture. In the following sections, we focus on quantum algorithms which have been hypothesized to offer a disruptive quantum advantage for industry-relevant applications. We first provide a brief overview of quantum computing for chemistry, and highlight the logical resource requirements for a quantum advantage use case, simulating the ground state of the FeMoco molecule, which we use as the starting point for our investigation. We then calculate the number of physical qubits that are required to break the elliptic curve encryption of Bitcoin keys within the time frame that it actually poses a threat to do, as a function of the code cycle time.

6.1.1 Fault tolerant quantum chemistry

When numerically simulating quantum chemistry problems, typically a set of independent functions, known as a basis set, are introduced to describe the physical wave function of the system. This introduction does not remedy the exponential increase of parameters with system size but enables one to balance the achievable accuracy against the required computational resources. Richard Feynman was perhaps the first to envisage a quantum computer and its application to the simulation of physics and chemistry [4]. It is now expected that quantum computers will eventually be able to perform electronic structure calculations with a quality of solution typical to the most accurate classical methods but with run times comparable to the approximate techniques, such as density functional theory.

The Quantum Phase Estimation (QPE) algorithm generates eigenvalues for a general unitary operator and it can be applied to quantum chemistry to find the eigenenergies of chemistry Hamiltonians to FCI (full configuration interaction, i.e., exact) precision. Unlike the Variational Quantum Eigensolver (VQE) [94] which involves many iterations ($\mathcal{O}(1/\epsilon^2)$ with accuracy ϵ) of low depth circuits, the QPE algorithm requires $\mathcal{O}(1)$ iterations of a circuit with a depth scaling as $\mathcal{O}(1/\epsilon)$. The large depth required in the QPE algorithm means that it will only be possible with error corrected devices, because NISQ devices would lose their coherence long before the end of the circuit.

Hamiltonian simulation is used as a subroutine in the quantum phase estimation algorithm, and it involves constructing a quantum circuit which approximates the evolution of the input state according to the Hamiltonian. Two of the main paradigms for Hamiltonian simulation are Trotterization and Qubitization. Qubitization [182, 183] can be used to simulate the Hamiltonian evolution by using quantum signal processing [208] but more commonly it is used to generate a quantum walk [209] upon which one can directly perform phase estimation. Qubitization is perhaps the most favored method for simulating chemistry Hamiltonian dynamics because it achieves the provably optimal scaling in query complexity and approximation error albeit while requiring more logical qubits than other methods.

Previous work has investigated the potential for quantum computers to provide a quantum advantage by performing ground state energy estimations on the catalytic complex known as FeMo-co [188, 210, 211, 212]. FeMo-Co is a large molecule expressed in biology which reduces N_2 from the atmosphere, and a better understanding of this process could provide a significant commercial advantage by improving the efficiency of nitrogen fixation for the production of ammonia for fertilizer. In this work we start our investigation with the latest logical resource requirements for simulating FeMo-co and calculate the number of physical qubits required to reach a desirable run time as a function of the code cycle time of the hardware.

6.1.2 Breaking Bitcoin's encryption

Bitcoin, the first decentralized cryptocurrency is continuing to grow in popularity. Bitcoin has properties which make it desirable as a hedge against inflation, for example, the rate of supply is known, decreases with time, and is entirely independent of demand. The decentralized nature of the blockchain makes it censor resistant and it can operate in a trustless manner. There are two main ways in which a quantum computer may pose a threat to the Bitcoin network [213, 214]. The first and least likely is the threat to the proof of work mechanism (mining) for which a quantum computer may achieve a quadratic speedup on the hashing of the SHA256 protocol with Grover's algorithm [114]. The algorithmic speedup is unlikely to make up for the considerably slower clock cycle times relative to state of the art classical computing for the foreseeable future [214]. The second and more serious threat would be an attack on the elliptic curve encryption of signatures. Bitcoin uses the Elliptic Curve Digital Signature Algorithm (ECDSA) which relies on the hardness of the Elliptic Curve Discrete Log Problem (ECDLP) and a modified version of Shor's algorithm [111, 215, 216] can provide an exponential speedup using a quantum computer for solving this problem. Bitcoin uses ECDSA to convert between the public and private keys which are used when performing transactions. With best practices (using new addresses for each transaction), the only point at which a public key is available and relevant to an eavesdropper, is after a transaction has been broadcast to the network but prior to its acceptance within the blockchain. In this window, transactions wait in the "mem pool" for an amount of time dependent on the fee paid, the time taken for this process is on average 10 minutes but it can often take much longer. Gidney and Ekerå estimated that it would require 20 million noisy qubits and 8 hours to break the 2048 RSA encryption [113] which is of a comparable difficulty to the EC encryption of Bitcoin. The maximum acceptable run time for breaking Bitcoin's encryption makes it particularly well suited to our investigation into the cost of parallelization and the interplay between the code cycle time and scalability, which we present in section 6.3.

In the following section we introduce considerations for error correction and provide an overview of the space and time optimizations within the surface code that we make use of in this work.

6.2 Space and time optimizations in the surface code

In this section we briefly introduce quantum error correction in the context of resource estimation and explain some of the available strategies within a surface code setup which are selected based upon a preference for space (physical qubit requirement) or time (final run time of the algorithm).

6.2.1 The available gate set

An important consideration for quantum error correction is the available logical gate set which is generally more restricted than the underlying physical gate set. The Clifford gates are those that map Pauli operators onto other Pauli operators, and the set can be generated by various combinations of the set $\{H, CNOT, S\}$ where the S gate is the Pauli $Z^{1/2}$. The Gottesman-Knill theorem [7] states that any Clifford circuit of finite size can be simulated in polynomial time (efficiently) with a classical computer. The Clifford gate set in combination with any non-Clifford gate is sufficient for universal quantum computation, and two of the most commonly considered non-Clifford gates are the T gate ($Z^{1/4}$) and the Toffoli gate (control-control-not). Any arbitrary angle single qubit gate can be decomposed into long sequences of the fixed angle H and T gates with chain length scaling with desired precision, as per the Solovay-Kitaev theorem [9].

$$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}, \quad S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \quad H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{6.1}$$

The surface code has transversal access to the CNOT, and the H and S gates can be realized with a low overhead using other techniques [8]. The T-gate is not transversal in the surface code and it must be effectuated using methods which incur a large space-time volume overhead relative to the other gates listed here. The T gate can be constructed by consuming a magic state, $|m\rangle = (|0\rangle + e^{i\pi/4}|1\rangle)/\sqrt{2}$ [26], where the magic state can be produced with an error proportional to the physical error, independent of the code distance [27]. To create a sufficiently high quality magic state, a distillation protocol [28, 29] can be used which essentially involves converting multiple low fidelity states into fewer higher fidelity states. Due to the high time cost associated with magic state distillation (the production and consumption), we can make a simplifying assumption that the time required to perform the T-gates effectively determines the final run time of an algorithm, as the relative cost of performing the Clifford gates fault-tolerantly is negligible. Some algorithms more naturally lend themselves to being expressed in terms of Toffoli gates, and there exist distinct specialized distillation factories to produce the magic states required to effectuate both of these non-Clifford operations. A Toffoli gate can be decomposed using 4 T gates [217], whereas the CCZ states normally produced for the Toffoli gate can be efficiently catalyzed into two T states [121].

6.2.2 Error correction and logical error rate

A logical qubit in the surface code consists of data qubits, which store the quantum information, and ancillary qubits, which are used for stabilizer measurements that non-destructively extract error information from the data qubits. The distance, d, of a code represents the minimum size of physical error that can lead to a logical error and in the surface code the number of physical qubits per logical qubit scales as $2d^2$. The logical error rate per logical qubit, p_L, per code cycle as a function of the base physical error rate, p, can be approximated by [25]

$$p_L = 0.1(100p)^{(d+1)/2}. \tag{6.2}$$

The efficiency of the error protection decreases as the base physical error rate approaches from below the threshold of the code (here assumed to be 1%). For feasible final resource estimates the base physical error rate will need to be close to or below 10^{-3}, where the necessary code distance is chosen based upon both the base physical error rate and the length of the desired computation. The physical error model here is the assumption that each physical operation has a probability p of also introducing a random Pauli error.

The achieved gate fidelity (the go-to metric for experimentalists) cannot be directly converted with confidence to the physical error rate, without further information. The cause of the gate infidelity, and where it exists on the spectrum between the two extremes of depolarizing error (decoherent noise) and unitary error (coherent noise) will determine the corresponding gate error rate. The best case situation is the one to one mapping between gate error (1-fidelity) and depolarizing error rate, and this has often been an assumption in the experimentally-focused literature. A measure of gate fidelity alone cannot determine the unitarity of the noise, i.e. the relative contribution of coherent and decoherent errors. Coherent errors, such as an over rotation of an intended gate, can positively interfere with each other and therefore cause worse case errors than those that are decoherent. The worst case scaling of the physical error rate, p, with gate fidelity F, and dimension of gate D, is $p = \sqrt{D(D+1)(1-F)}$ [218]. To illustrate an example of this worst case scenario on noise quality, to guarantee an error rate of below 1%, a gate fidelity of 99.9995% would be required [218]. To determine where on the unitarity spectrum the actual hardware noise exists, protocols based on randomized bench marking can be used [219, 220]. This information can then be used to estimate the base physical error rate with confidence [218, 221].

6.2.3 Code cycle, reaction time and measurement depth

The code cycle is the base unit of operation in the surface code and it involves performing a full round of stabilizer measurements. As all operations in the computer are assumed to be subject to errors, including the stabilizer measurement process, the code cycle needs to be repeated d times before corrections can be applied. We will refer to the time it takes to perform these d rounds of code cycles as a "*beat*", where many surface code operations will have a time cost measured in *beats*. A fault tolerant quantum computer using the surface code can be envisaged as partitioned into two sections, data-blocks which consume magic states to effectuate T gates for the desired algorithm, and distillation-blocks which produce high fidelity magic states. Each of these constructs in the surface code consist of a number of logical qubits, sometimes referred to as tiles, and each tile contains $2d^2$ physical qubits. The data block has a size scaling with the number of abstract qubits required for the algorithm and this then sets the minimum required number of physical qubits when paired with a single magic state factory and given the code distance.

The T (or Toffoli) gates of an algorithm can be arranged into layers of gates (measurement layers), where all of the gates within a layer could potentially be executed simultaneously. Measurement layers are sometimes instead referred to as T layers when the relevant non-Clifford gate is the T gate, and the measurement (T) depth is the number of measurement layers in the algorithm. When a magic state is consumed by the data block, a Pauli product measurement is performed to determine whether an error has occurred so that a (Clifford) correction can be applied if required. The algorithm cannot proceed to the next measurement layer until all of the necessary corrections have been applied in the current layer, and this process requires a classical computation (decoding and feed-forward). The characteristic time cost that includes the quantum measurement, classical communication, and classical computation is referred to as the "reaction time". It is conjectured that the fastest an error corrected quantum algorithm can run, i.e. the time optimal limit [222], is by spending only one reaction time per measurement layer, independent of code distance, and we will refer to this as reaction limited. In the case of superconducting devices which have relatively fast physical gates and measurements, the reaction time may be dominated by the classical communication and computation. A reaction time of 10 μs has been used in recent resource estimation work [113], as compared to the 1 μs code cycle time, which requires both physical two qubit operations and measurements. In this work we have defined the reaction time (RT) as a function of the code cycle time (CC) with $RT = (CC/4) + 10\mu s$ unless otherwise stated. This assumption is motivated by the fact that generally the quantum measurement within the code cycle represents a fraction of the total time, for example with the shuttling based trapped ion architecture with a code cycle time of 235 μs, the quantum measurement is estimated to represent $\sim 10\%$ of that total time. We include a code cycle independent additional cost of $10\mu s$ to represent the (somewhat) hardware independent classical processing. With our resource estimation tool, which is available upon request, one could recreate the results of this chapter with a different relationship between the code cycle time and reaction time. If particular surface code set up contained only a single distillation factory, then the rate of computation would likely not be limited by the reaction time but instead by the rate of magic state production. We refer to the regime of being limited by magic state production as "*tick*" limited. There are then three relevant regimes for surface code strategies which are separated by the limiting factor of computation rate. *Beat* limited implies the limiting factor is the rate of magic state consumption by the data block, *tick* limited, the rate of magic state production by the distillation blocks, and *reaction* limited, where the conjectured time-optimal limit is reached and one reaction time is spent per measurement layer.

In this work we utilize and compare two distinct strategies of incrementally trading qubits for run-time up to the reaction limit, and introduce them later in this section.

6.2.4 Distillation and topological errors

When choosing a surface code set up one must decide upon the acceptable final success probability, where in principle a success probability greater than 50% would be sufficient to reach the desired precision by repeating the computation. The acceptable success probability then allows one an additional method of trading space for time, as the lower the acceptable success probability, the lower the various code distances would need to be. There are two contributions to the probability of failure, the topological error associated with the data block, and the total distillation error. The topological error is the chance for at least a single logical error within the data block across the entirety of the algorithm, which can be calculated with the product of the number of logical qubits, the number of code cycles required for the algorithm, and the logical error rate. Where the logical error rate is defined in equation 6.2 by the base physical error rate and the code distance on the data block. The total distillation error corresponds to the probability of at least a single faulty magic state which is calculated by the product of the total required number of required magic states and the error rate per state. The error rate per state is determined by the particular factory chosen and its associated code distances. We can then consider the final failure probability as the linear sum of the topological error and total distillation error.

In this work we set the allowable total distillation error at 5% implying that across the entire algorithm the probability that a magic state is generated with an error is 5%. We choose the appropriate distillation protocol to achieve this error rate, by selecting between T factory protocols of Litinski [27], and by adjusting the level 1 and level 2 code distances of the AutoCCZ factory. The choice of 5% is in part motivated by the capacity of the AutoCCZ factory to reach a sufficient fidelity given the number of magic states required in the quantum advantage cases we address in this work. We set the final topological error to be 1% and choose the appropriate code distance to achieve this by considering the total number of code cycles that the algorithm must run for. This leads to a total final error (failure probability) of $\sim 6\%$. In Litinski's work a final error of 2% is chosen [14], whereas in Gidney and Ekerå's work of breaking RSA encryption the final error is 33.4%. The best choice of final error tolerance will depend on the type of problem being solved, in the case where the result being correct is heralded (e.g. factoring), one can accept large probabilities of failure, leading to a flexible trade-off between space and effective run time (including retries). Therefore in this case the choice should be framed as an optimization problem as opposed to an arbitrary threshold decision. Algorithms which require statistical accumulation from multiple runs may need the failure rate to be considerably lower than in the heralded type algorithms. Using our resource estimation tool, one could investigate the impact of different final failure probabilities.

6.2.5 Routing at the error corrected level

It is necessary to move logical information from one area of the device to another, and perhaps the most common requirement is the transport of magic states from the distillation factories to the data blocks. Analogous to physical qubits in superconducting devices which use logical operations to perform swaps, one can imagine performing swap operations between logical qubits in the surface code. However, alternative techniques are more often considered for enabling long range interactions with topological error correcting codes. Using lattice surgery based methods, long range interactions (CNOTs) can be enabled in d code cycles (1 *beat*), essentially independent of the distance between the two points, so long as there is a chain of free ancillary qubits between them. We refer to this method as entanglement swapping. An ancillary logical qubit can only contribute to one routing chain at a time (per beat). When defining the layout of an error correction set up, which involves choosing the number of distillation factories and orienting them with respect to the data blocks, one must also ensure there is sufficient ancillary routing space to enable the required degree of data transfer between the relevant areas. The degree of data transfer, or alternatively stated the degree of parallelization in the execution of the algorithm, is often characterized by the number of magic states consumed per beat across the entire data block. With a data block arranged into rows with an access hallway between each row (consisting of ancillary logical qubits), there are two unique ways of touching each data qubit, if this is deemed insufficient, entangled copies of the data rows can be created to ensure there are enough unique paths between the factories and the data qubits. In section 6.2.8 we present our choice of the routing overhead factor for entanglement swapping as a function of the degree of parallelization using AutoCCZ factories.

6.2.6 Considering physical mid-range connectivity

In this section we briefly consider the potential impacts on the required resources when the underlying architecture has access to flexible mid-range connectivity between physical qubits. For superconducting devices two qubit operations can only be performed between nearest neighbour physical qubits, and so long range interactions must be enabled by sequences of costly logical swap operations. Alternative hardware may have access to low overhead long range interactions between physical qubits, for example, photons are readily transported long distances which is relevant to both photon-only hardware, and for connecting small modules of trapped ions with photonic interconnects. In the blueprint for a shuttling based trapped ion architecture [74], a single system is envisaged, comprised of iterable micro-fabricated chips, connected via electric fields, which allow for physical shuttling between modules. High state fidelity adiabatic shuttling has been demonstrated with a speed of $\sim 20\ ms^{-1}$ [85]; diabatic techniques [144, 145, 146] can enable much greater shuttling speeds, and $\sim 80\ ms^{-1}$ has been demonstrated [145]. Lau and James calculate that the maximum speed a $^{40}Ca^{+}$ ion can be transported across a 100 μm trap without excessive error is 10,000 ms^{-1} [147]. For NISQ size devices all to all connectivity should be achievable with high fidelity (relative to two qubit operations) using the shuttling-only approach [11], but physically shuttling completely across a device with over a million physical qubits is unlikely to be feasible due to the associated time cost. Utilization of mid-range connectivity may still enable a reduction in the routing overhead associated with entanglement swapping.

While very long range shuttling operations may be protected from error by periodic cooling operations and mid-circuit syndrome extraction and correction, the total time cost must be considered. With entanglement swapping, long range interactions can be enabled between logical qubits in the surface code in a single beat, provided there are sufficient available ancilla qubits between the locations. To contrast this capability we estimate the range at which physical shuttling may remain competitive with entanglement swapping. Assuming a code distance of 30, and logical qubits distributed across a 2D square grid, we estimate that a logical qubit qubit could interact via physical shuttling with another logical qubit in the range of 3-30 grid spaces away within a single beat (d code cycles), depending on physical ion density and shuttling speed. While this is indeed unlikely to be sufficient for mediating all long range interactions between logical qubits, the capability of low cost mid-range physical connectivity could make the transversal CNOT preferable to the more usually considered lattice surgery based methods. Gutiérrez et al have investigated the experimental regimes at which the transversal CNOT may outperform the lattice surgery based methods for trapped ions [223], and for particular values of error contributions, the transversal CNOT may be performed a factor 10× faster. If the transversal CNOT is expected to require less time than the lattice surgery based CNOT, then this would directly impact the rate of magic state production of distillation factories and therefore could reduce the total qubit overhead in the highly parallelized regime. Furthermore, mid range physical connectivity could considerably reduce the qubit footprint of distillation factories by eliminating the need for interior ancillary routing space for entanglement swapping. Alternative error correction strategies to the 2D surface code [30, 201] may be achievable on hardware with flexible mid-range connectivity. We leave a more detailed analysis of the potential benefit of mid-range connectivity as future work.

In the next subsection we introduce the Game of Surface Codes method of trading space for time [14], and following that, the AutoCCZ method [113] where we also include some more detailed assumptions on the necessary routing overhead for entanglement swapping.

6.2.7 Game of Surface Codes

In the work of Litinski [14], various data blocks are presented which vary in their rate of T-gate effectuation (magic state consumption) and the number of required physical qubits. There are numerous distillation protocols each of which varies with regard to the output fidelity, required number of physical qubits, and the rate of production. Distillation blocks can be parallelized and this enables further space-time trade-offs.

We make use of distillation strategies presented by Litinski [27] where the distance associated with the distillation blocks is fine tuned and separate from the distance associated with the data blocks. The data blocks have a required code distance set by the total number of logical qubits and the total number of code cycles required to run the entire algorithm. With the total number of T gates in the algorithm, T_{count}, the distillation blocks only need a code distance sufficient to produce magic states with an error at least lower than $1/T_{count}$. In this chapter we choose to set the acceptable error at $1/(20 \times T_{count}$, corresponding to a $\sim 5\%$ probability of distillation error across the algorithm. The distillation blocks use a certain number of qubits and only need to be protected for a certain number of code cycles (corresponding to one full round of distillation), and these are generally both small relative to the requirements of the data blocks. This method of individual calibration of distance for the data and distillation blocks is in contrast to a prior method [14] where both block types are attributed the same code distance.

In the GoSC scheme, Clifford gates are addressed explicitly via Clifford tracking, i.e. all Clifford gates are commuted to the end of the circuit and absorbed into measurements. This turns T gates into Pauli product rotations, and measurements into Pauli product measurements. In general these Pauli product rotations can be big multi-qubit operations. To account for the generalized worst-case algorithm, which may have these multi-qubit operations, the maximum rate at which the data block can consume a magic state is defined as one state per *beat* (d code cycles). If the input circuit is known then it will sometimes be possible to arrange the data blocks in such a way so that more than one state can be consumed per *beat*. In this investigation we do not consider the details of the input circuit and instead rely only meta details such as, abstract qubit count, total T count, and the measurement depth. Our utilization of the GoSC method should then be considered an upper-bound configuration, i.e., guaranteed to be able to support any algorithm configuration given the meta details. The number of distillation factories can be chosen to match the production rate to the consumption rate of the data block, at which point we may describe the strategy as *beat* limited.

In the work of Litinski [14] further time optimizations are presented, which can be utilized to reach reaction limit where one reaction time is spent per measurement (T) layer. The reaction limited strategy is set by the total number of measurement layers, i.e. the measurement depth (T_{depth}), as opposed to the total T gate count (T_{count}) of the *beat* limited strategy. The average number of parallel-executable T gates per layer ($T_{layer} = T_{count}/T_{depth}$) varies across particular algorithms and there exist methods to optimize circuits to minimize either the T_{count}, T_{depth}, or the circuit width [224, 225]. Litinski's method of speeding computation up beyond the *beat* limited case utilizes "*units*" which combine the previous constructions of data and distillation blocks. The number of units can be increased until the time optimal limit (henceforth reaction-limit) is reached, where each unit can work in parallel to address a set of measurement layers. This does not contradict the previously provided definition of measurement layers, as although the units can parallelize aspects of the work, the layers must still be stitched back together requiring one reaction time per measurement layer for corrections. The reaction limit then defines the maximum number of units that can be utilized. For an algorithm requiring n abstract (logical) qubits, with an average of, T_{layer}, T gates per measurement layer, a single unit will consist of $4n + 4\sqrt{n} + 1$ tiles (logical qubits) for the data block, and 2 T_{layer} storage tiles. Each unit requires an amount of time, t_u, to process a single measurement layer, where it is measured in *beats* and scales as $T_{layer} + \sqrt{n} + 3$. Each unit will need a number of distillation factories to match the required production rate, which is T_{layer} magic states per unit completion time, t_u. With a linear arrangement of units as is considered here, and a reaction time, RT, the reaction (time-optimal) limit is reached with a number of units n_u equal to $t_u/RT + 1$. The GoSC scheme of going beyond the *beat* limited rate initially incurs a space-time overhead but then allows one to trade linearly (by increasing the number of units) until the reaction limit is reached.

6.2.8 AutoCCZ factories

In the work of Gidney and Ekerå [113] detailed surface code layouts along with the logical algorithmic developments for breaking RSA encryption are provided. The surface code strategy utilizes AutoCCZ factories [207] which create a magic state (the CCZ state) that can be consumed to effectuate the non-Clifford Toffoli gate. The "Auto" refers to the fact that these factories create auto corrected CCZ states, meaning that the potential correction operation associated with magic state consumption is decoupled and can be performed far away from the data block.

Using this scheme with the AutoCCZ factories, the *beat* limited rate can be surpassed without introducing units as is performed in GoSC. Provided the routing overhead is accounted for, one can continue to add AutoCCZ factories until the production rate is equal to the number of Toffoli gates per measurement layer per reaction time, at which point the reaction limit is reached.

The AutoCCZ factory is characterized by two code distances, corresponding to the two levels of the protocol. The factory is a tiered distillation scheme where the output magic states of the first level are the input states to the second level. We calculate the final output fidelity following the ancillary files of Gidney and Ekerå [113] as a function of the two code distances and base physical error rate. To choose the optimal two code distances we assess a wide range of possible values and select the setup that reaches the desired final distillation error rate while minimizing the space-time volume of the factory. While technically possible to maintain a reasonably low distillation error rate with base physical error rates near the threshold of the code, 10^{-2}, the associated code distances required would result in an infeasible physical qubit overhead. Assuming any code distance is acceptable, the final distillation error per state, p_D, can be described as a function of the base logical error, p, by $p_D = 34300p^6$, until the threshold of the code is reached. This relationship was found by numerical fitting. A limit on the allowable code distance breaks away from this trend before the threshold, and the lower the limit the sooner it breaks upwards. In figure 6.3 we investigate the impact of the base physical error rate for the final qubit overhead to reach a desirable run time with the AutoCCZ method. If the final output fidelity of the AutoCCZ factory is insufficient given the desired length of an algorithm and base physical error rate, then alternative T gate factories may still be viable [27].

To reach a particular desired run time, assuming it is below the reaction limit, the number of AutoCCZ factories are chosen as necessary. Once this footprint arrangement is settled (the combination of the data block and the number of factories), the last stage of the calculation is to determine the necessary routing overhead to account for the degree of parallelization. As described in section 6.2.5, long distance interactions between the distillation block and data block can be enabled in one beat provided there is an available chain of ancillary logical qubits (routing space) between them. As each ancillary logical qubit can only contribute to one routing chain per beat, there may need to be additional unique paths to account for the degree of parallelization. All of the necessary routing overhead is strictly accounted for in GoSC, with the construction and arrangement of the data blocks, which are then duplicated and distributed across units. In our utilization of AutoCCZ factories we define the degree of parallelization as the number of magic states consumed across the data block per beat. We then ensure there is a routing hallway per data block row for every state consumed per beat, to exceed two hallways per data row, entangled copies of the data block can be made (which is comparable to the entangled copies across Units in GoSC). Finally we multiply the entire area (number of logical qubits) by a factor of 1.2 to ensure the distillation factories are surrounded by hallways and for some work space around the data block for arranging routing-chains. Our utilization of AutoCCZ factories should not be considered a true upper bound for any generalized circuit, unlike GoSC Units. In the following section we contrast the two methods further and state our relative contribution.

6.2.9 Problem specification

We start our investigation with the logical resource requirement set out by Lee et al [212] to simulate FeMoco to chemical accuracy, and assess the feasibility of reaching desirable run times as a function of the code cycle time and number of achievable physical qubits. In the work of Lee et al, a detailed surface code strategy is presented with algorithm specific optimizations, such as minimizing the number of data qubits that are stored for working. In contrast, the surface code strategies we utilize, do not necessarily require detailed knowledge of the input circuit, and are instead only a function of the logical qubit count, the T (or Toffoli) count and the measurement depth. This approach may not yield optimal results for specific algorithms but it enables one to effectively estimate physical resource requirements for a particular algorithm as a function of strategy (the time-space optimization spectrum) and the code cycle time. We contrast two strategies, the first which can be considered to provide upper-bound resource estimates for any general circuit input, and closely follows the work of Litinski's Game of Surface Codes (GoSC) [14]. We go beyond the pedagogical examples provided by Litinski [14] by creating an automatic tool that calculates the physical resources across the space-time optimization spectrum and apply it to the algorithmic requirements of quantum advantage use cases. Furthermore we use the tool to calculate the number of physical qubits required to reach a specified run time as a function of the code cycle time of the hardware by utilizing the necessary degree of parallelization. We investigate the impact of varying the measurement depth with a fixed total T count, on the efficiency of the GoSC unit approach and identify the optimal value in particular regimes. In addition to the Game of Surface Codes method of parallelization, we incorporate a method which uses AutoCCZ factories [207]. For this method we relied upon the ancillary files of Gidney and Ekerå [113] and adapted them to be flexible enough for our broader (circuit agnostic) considerations. Our intention with the routing overhead with the AutoCCZ factories is to cover a wide range of possible circuit characteristics, while it may be an over estimation for some specific circuits, it may also represent an under estimation for some worst-case situations. We accomplish our aim of quick and general resource estimation as a function of algorithm meta information, and hardware characteristics, by contrasting the upper-bound scenario of GoSC units, with a more heuristic utilization of AutoCCZ factories. With detailed knowledge of the input circuit, further optimizations of the footprint configuration are possible, but generally these must be performed on a case by case basis and are non trivial to automate. The tool used to generate the results presented in this paper (chapter) is available upon request. We use the latest logical resource requirements for breaking elliptic curve encryption [215, 216] and estimate the number of physical qubits required to break the encryption of Bitcoin keys in the small amount of time it would actually pose a threat to do so, all as a function of both the code cycle time and base physical error rate.

6.3 Results

To calculate the results presented in this section we use various surface code strategies including the Game of Surface Codes scheme which uses units to parallelize layers of T gates [14], and AutoCCZ factories [207, 113], which are both highlighted in the previous section.

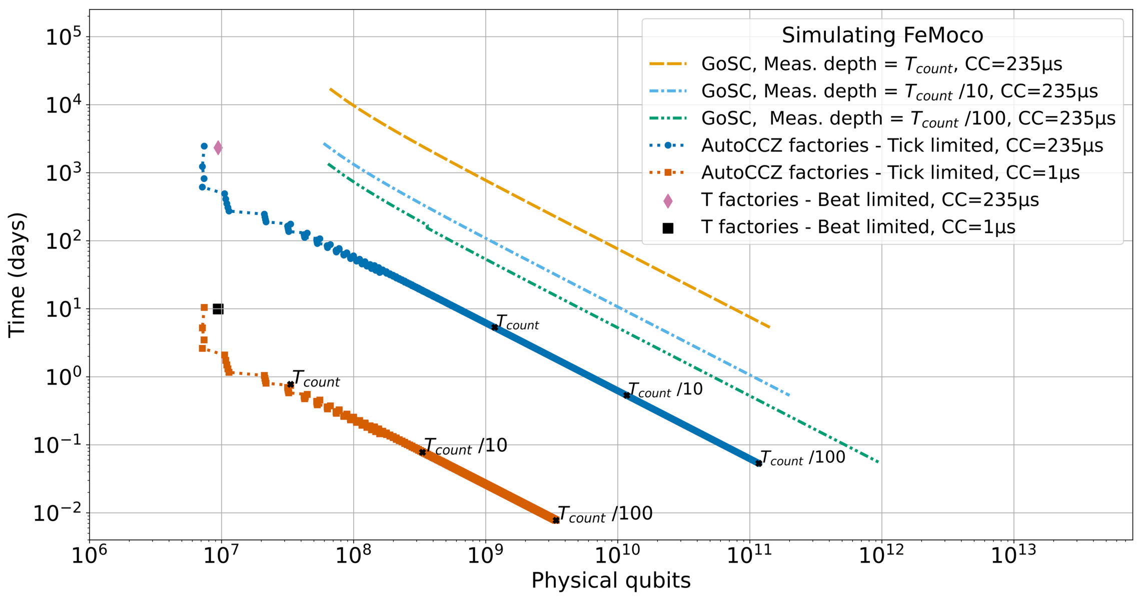


Figure 6.1: Physical resource estimates for a ground state energy calculation of the FeMoco molecule to chemical accuracy as per the logical resource requirements of Lee et al [212]. The associated logical resources required are 2196 logical qubits and 6.7 billion Toffoli gates. The run time and associated physical qubit count is plotted for different surface code strategies and code cycle times. A code cycle time of 1 μs and reaction time of 10 μs is considered, which may correspond to future superconducting devices, in addition to a code cycle time of 235 μs with reaction time of 70 μs which may correspond to a future shuttling based trapped ion architecture [74]. The base physical error rate is set to 10^{-3}, the final distillation error probability is at most 5%, and the final topological error probability is at most 1%. The method from A Game of Surface Codes (GoSC) [14] is utilized where layers of T gates are parallelized using "Units" and is shown for the code cycle time of 235 μs. The three distinct trends all with dashed lines correspond to different measurement depths given as a fraction of total non-Clifford gate count, T_{count}. A *beat* limited approach (i.e. limited by the magic state consumption rate of the single data block) through the GoSC lens are plotted as a square and diamond. With the GoSC approach the improved distance T gate factories of Litinski [14] were used, where their distance is calibrated separately to the data blocks. A distinct method of parallelization is utilized here, which uses AutoCCZ factories [207, 113], enabling all of the Toffoli gates within a given measurement layer to be potentially performed in parallel. The AutoCCZ approach is plotted for the two code cycle times of 1 μs and 235 μs as joined markers where the trend starts with 1 AutoCCZ factory and sequentially adds factories, each represented by a data point, up until the reaction limit is reached.

6.3.1 Simulating FeMoco as a function of the code cycle time

There has been extensive research into both algorithmic development and resource estimation in the field of fault tolerant quantum chemistry, and one of the focus points has been the FeMo-co catalyst [188, 210, 211, 212]. An improved understanding of the FeMo-co catalyst could provide considerable efficiency improvements to nitrogen fixation which currently represents around 2% of the worlds energy usage. We start with some of the latest algorithmic developments by Lee et al [212] and investigate the feasibility of achieving a reasonable run time for different code cycle times and different surface code strategies. The associated logical resources required are 2196 logical qubits and 6.7 billion Toffoli gates.

In figure 6.1 we compare two distinct methods of trading space for run time up to the reaction limit (the conjectured time optimal limit [222]). The two scatter trends utilize AutoCCZ factories with different code cycle times of 1 μs, corresponding to a future superconducting device, and 235 μs, corresponding to a future shuttling based trapped ion device [74]. Each trend starts with one AutoCCZ factory, from there the number of factories is incremented and at each step the code distance is calibrated and the resulting run time and physical qubits are plotted. Initially the total qubit footprint is dominated by the data blocks whereas the run time is bottle necked by the magic state production, i.e. going from one factory to two halves the expected run time. Therefore while this is the case, adding factories results in an improvement to the space-time volume and this can sometimes allow for a reduction in code distance, which may result in an actual reduction in total qubit count. Each time a factory is added, the magic state consumption rate per beat is defined to determine whether the routing overhead needs to be increased as described in section 6.2.8. What is considered to be a desirable run time will largely depend on the importance of the problem being solved and by the speed and quality of the classical alternatives. With one AutoCCZ factory the superconducting device completes in around 10 days with 7.5 million qubits, whereas the trapped ion device requires 2450 days and the same number of qubits. Where 10 days may be considered a quantum advantage for this use case where classical computers stand no chance of providing a meaningful answer, perhaps 2450 days would not. By parallelizing the magic state production the trapped ion device can reach the run time of 10 days requiring 600 million qubits. The factor difference between the physical qubit count here is less than the factor difference between the code cycle time, because initially adding factories is a favourable space time trade until the total qubit footprint becomes dominated by the factories at which point the trade becomes linear. The good news for hardware with slower code-cycle times is that it will often be possible to still reach desirable run times provided enough physical qubits are available. However, the associated qubit overhead may appear daunting, and implies that hardware with slower code cycle times will have to be more scalable to compete, assuming equal error rates and physical connectivity. We plot for a range of possible measurement depths, labeled as a fraction of the total Toffoli count, as this was not provided along with the other logical requirements [212]. In the AutoCCZ scheme the measurement depth does not directly impact the efficiency of the approach, instead it only determines in combination with the reaction time, what the time optimal (reaction) limit is. The labels then indicate the reaction limit, the point at which the trend would end, given that measurement depth.

In figure 6.1 we also include the Game of Surface Codes (GoSC) approach to trading space for time [14], where measurement layers are parallelized with constructs called "Units". Each unit contains its own copy of the data block and enough factories to produce the number of magic states within the measurement layer within the time it takes to prepare the unit, which scales with both the number of magic states per layer and the number of abstract qubits. Units can be incrementally added, each one added reduces the final run time up until the reaction limit is reached. The dashed lines in figure 6.1 use units along with improved T gate factories [27] where it is assumed 4 T gates are required to decompose a Toffoli gate [217, 14]. The GoSC approach is plotted only for the code cycle time of 235 μs but three trends are included for the different measurement depths as a fraction of the total Toffoli count. The efficiency of the GoSC approach (in addition to the reaction limit) is dependent on the measurement depth as can be seen. In section 6.3.3 we investigate the impact of the measurement depth on the final qubit requirement to reach a fixed run time for the GoSC approach.

It can be seen in the figure that the AutoCCZ approach provides more favourable final resource estimates than GoSC units in this scenario. The largest contributing factor to this difference appears to be the initial set up cost for the unit approach, which is nearly an order of magnitude increase in qubits for no appreciable speed up (with measurement depth = $T_{count}/10$). This is in stark contrast to the AutoCCZ approach which initially takes very favourable space-time trades by increasing the number of factories. As the rate of parallelization increases in the AutoCCZ approach, eventually entangled copies of the data block are made to maintain sufficient access hallways between the data block and distillation blocks. Both the AutoCCZ and GoSC approach converge towards an equal linear trade between space and time. It should be restated that the GoSC approach can be considered a true upper-bound estimate, functional for any general circuit, whereas our utilization of the AutoCCZ factories is more heuristic and may represent an underestimation for some specific circuit inputs, see section 6.2.9 for more discussion on this.

In figure 6.1 a *beat* limited method is included for both code cycle times as diamond and square points i.e. no units and the computation rate is limited by the data blocks magic state consumption rate. These points use T gate factories and the fast data blocks from GoSC, which have size scaling as $2n+\sqrt{8n}+1$ for n logical qubits, and can effectuate T gates at a maximum rate of one per *beat* (d code cycles). The number of T factories is chosen to match the rate of magic state consumption of the fast data blocks. The *beat* limited situation provides comparable run times to the single AutoCCZ factory and requires a similar number of physical qubits.

To conclude this section, it appears that the AutoCCZ factories are the favourable approach to trade space for time up to the reaction limit, but a more detailed investigation into the underlying assumptions of both methods is warranted. These resource estimates are solely a function of algorithm meta information such as total T count, measurement depth, and the number of logical qubits. The surface code configuration can be optimized when paired with detailed knowledge of the input algorithm, but this process is non trivial to automate, as we have done in this work as a function of code cycle time. Future hardware that expects to have considerably slower code cycle times than the superconducting devices may still be able to reach desirable run times provided enough physical qubits are available, which further emphasises the importance of scalability. The associated qubit overhead factor will range between less than 1 (here ~ 0.3), to 1, times the difference in the code cycle time depending on the relative degree of parallelization in the comparison. Algorithms should be optimized by minimizing the measurement depth if the reaction limit is restrictive. The physical qubit requirement may be reduced if the underlying hardware has access to low overhead mid-range physical connectivity, as discussed in section 6.2.6

6.3.2 Breaking Bitcoin's EC encryption

Breaking encryption has received a lot of attention in the quantum computing community since Shor's breakthrough algorithm [6] which provides a near exponential speedup for prime factoring which has direct implications for breaking RSA encryption. Gidney and Ekerå provide algorithmic improvements in addition to the surface code strategies for breaking RSA encryption and they estimate that 20 million qubits running for 8 hours

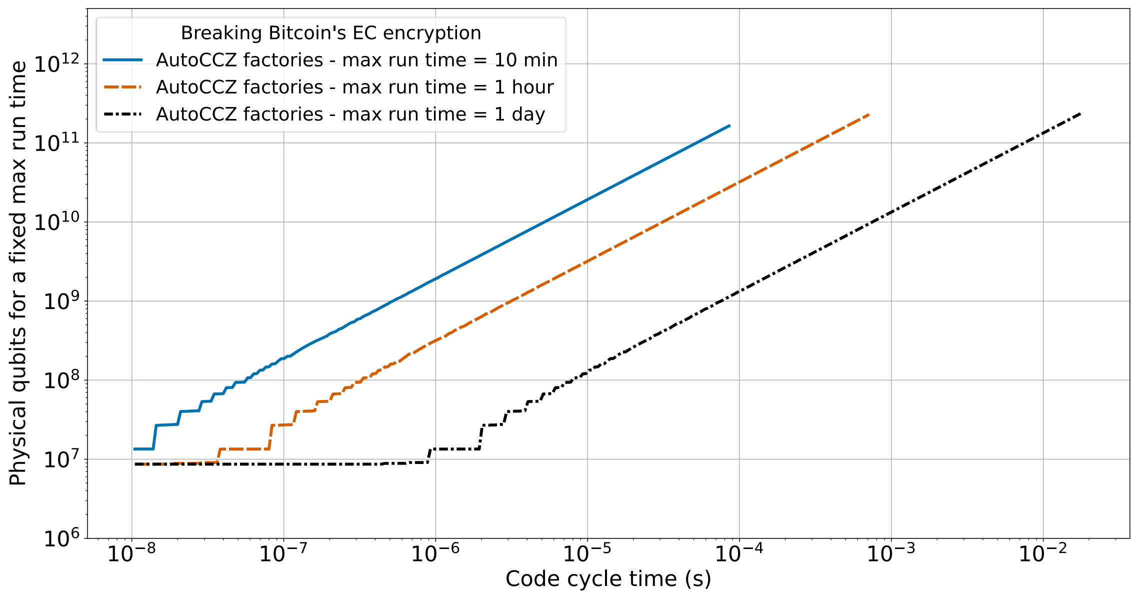


Figure 6.2: The number of physical qubits required to break Bitcoin's 256 elliptic curve encryption with a fixed maximum run time as a function of the code cycle time for maximum run times of 10 minutes, 1 hour, and 1 day. Using the latest algorithmic development for quantum circuits for elliptic curve encryption [215]; the depth optimized approach is chosen requiring $5.76 \cdot 10^9$ T gates, 2871 logical qubits, and a T (measurement) depth of $1.88 \cdot 10^7$. These trends utilize AutoCCZ factories to trade space for time to reach the desired run time and assume that a CCZ state can be efficiently traded for 2 T gates [121]. Assuming the relationship between reaction time (RT) and code cycle time (CC) of $RT = CC/4 + 10\ \mu s$ which is motivated in section 6.2. The trends start from the chosen code cycle time of 10^{-8} and end at the right due to reaching the reaction limit before reaching the desired run time. We use a base physical error rate of 10^{-3} and a final output error of $\sim 6\%$. .

could break it with a code cycle time of 1 μs [113]. In a blueprint for a shuttling based trapped ion device, which estimated the code cycle time to be 235 μs [74], it was originally estimated that breaking RSA encryption would require 110 days and 2 billion qubits with a base error of 10^{-3}, implying a device occupying an area of 103.5×103.5 m^2. With the latest algorithmic and surface code strategy improvements we can reduce this estimate to requiring instead a run time of 10 days (a factor 10$\times$ faster) and 650 million qubits, which would imply a device size of area 60×60 m^2. With a base physical error of 10^{-4} the device size would reduce to 18×18 m^2 and further reductions may be possible if one were to make use of the flexible mid-range connectivity that is available. Using the same relative improvement factor from mid-range connectivity as in the design blueprint [74],

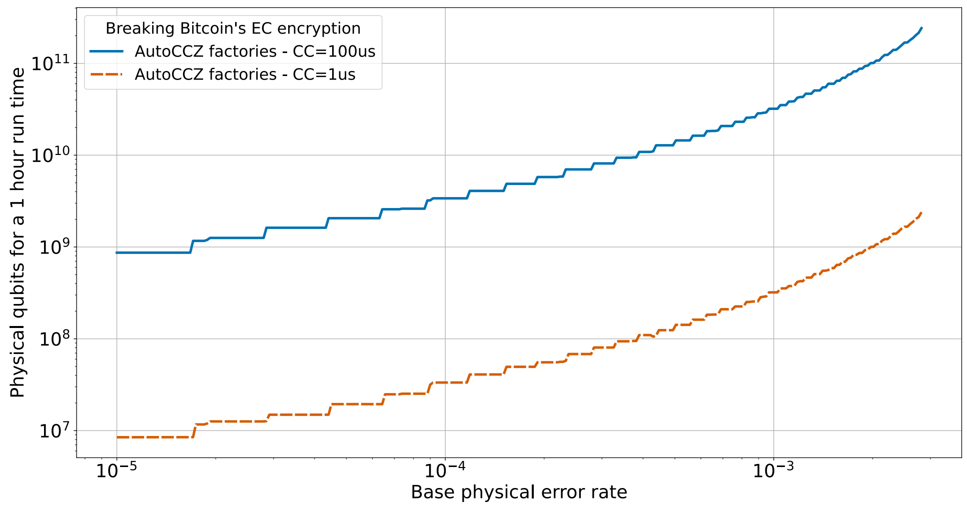


Figure 6.3: The number of physical qubits required to break Bitcoin's 256 elliptic curve encryption with a fixed maximum run time as a function base physical error rate for a maximum run time of 1 hour and code cycle time of both 1 μs and 100 μs. Using the latest algorithmic development for quantum circuits for elliptic curve encryption [215]; the depth optimized approach is chosen requiring $5.76 \cdot 10^9$ T gates, 2871 logical qubits, and a T (measurement) depth of $1.88 \cdot 10^7$. These trends utilize AutoCCZ factories to trade space for time to reach the desired run time and assume that a CCZ state can be efficiently traded for 2 T gates [121]. Assuming the relationship between reaction time (RT) and code cycle time (CC) of $RT = CC/4 + 10\ \mu s$ which is motivated in section 6.2. The trends start from the chosen base physical error rate of 10^{-5} and end at the right due to the AutoCCZ factory being no longer able to reach the desired distillation fidelity given the base physical error and number of required states.

starting from the 10^{-4} case, we might expect a device of size 2.5×2.5 m^2 to be sufficient. This is indeed a rough estimate and a more rigorous understanding of how to best make use of mid-range connectivity would be required to confidently provide the performance improvements relative to a nearest neighbour approach.

Bitcoin uses the Elliptic Curve Digital Signature Algorithm (ECDSA) which relies on the hardness of the Elliptic Curve Discrete Log Problem (ECDLP) and a modified version of Shor's algorithm [111, 215, 216] can provide an exponential speedup using a quantum computer for solving this problem. The encryption of keys in the Bitcoin network are only vulnerable for a short window of time, around 10 minutes to an hour depending on the fee paid, as described in more detail in section 6.1.2, and this makes it a well suited problem for our investigation. In figure 6.2 we plot the number of physical qubits required to break the elliptic curve encryption of Bitcoin within a run time of 1 day, 1 hour and 10 minutes, as a function of the code cycle time. We use the logical resources of the depth optimized approach provided by Haner et al [215] for 256 bit encryption, which corresponds to $5.76 \cdot 10^9$ T gates, 2871 logical qubits, and a measurement depth (T_{depth}) of $1.88 \cdot 10^7$. The measurement depth of this algorithm is low relative to the T_{count}, at $\sim T_{count}/300$, implying that there is a lot of room for parallelization before the reaction limit is reached. In figure 6.2 it can be seen that it would require 317 million physical qubits to break the encryption within one hour with a code cycle time of 1 μs. To break it within 10 minutes with the same code cycle time it would require 1.9 billion physical qubits whereas to break it in 1 day would require only 13 million physical qubits. The horizontal period most evident in the dash-dotted black trend for a run time of 1 day is because the desirable run time can reached for those code cycle times with only 1 AutoCCZ factory. Once the code cycle time increases sufficiently it is then necessary to begin adding AutoCCZ factories to maintain the desired run time. Hardware with considerably slower code cycle times than 1 μs will need to be able to reach larger device sizes to break the encryption within the allotted time. Even for code cycle times of 1 μs, this large physical qubit requirement implies that the Bitcoin network will be secure from quantum computing attacks for many years. High value transactions are likely to pay high fees ensuring they are processed with higher priority, and therefore would require considerably more physical qubits to break the encryption in time. The Bitcoin network could nullify this threat by performing a soft fork onto an encryption method that is quantum secure, where Lamport signatures [226] are the front-running candidate, but such a scheme would require much more memory per key. The bandwidth of Bitcoin is one of the main limiting factors in scaling the network and so changing the encryption method in this way could have serious drawbacks.

In figure 6.3 we plot the required number of physical qubits to break the 256 bit elliptic curve encryption within 1 hour as a function of the base physical error rate. In section 6.2.2 the relationship between the base physical error rate and an experimentally achieved gate fidelity is explained in more detail. We plot two trends for code cycle times equal to 1 μs and 100 μs which begin at the selected error rate of 10^{-5}. The trends end at the right at a physical error rate of $\sim 2.8 \cdot 10^{-3}$ when the AutoCCZ factory can no longer produce a state with sufficient fidelity given the error rate and number of required states. High code distance and multi tiered T gate factories would be able to continue further, but at a value of 1% error rate the (approximate) threshold of the surface code is reached. With a physical error rate higher than this threshold, increasing the code distance actually results in a larger logical error. Three distinct distances are calibrated according to the base physical error rate, first the final topological error is maintained below 1% by adjusting the code distance associated with the data block. Next there is a level 1 code distance and a level 2 code distance associated with the AutoCCZ factory that are calibrated to ensure the final distillation error is maintained below 5%, as per the ancillary files of Gidney and Ekerå [207]. We assess a wide range of code distances for the AutoCCZ factory and choose the set that minimizes the factory volume (i.e. number of qubits $\times$ duration per cycle) while maintaining the desired error rate. With a code cycle time of 1 μs it requires 317 million physical qubits to reach the 1 hour run time with a base error of 10^{-3}, this is reduced down to 33 million for a base error of 10^{-4}, i.e. a factor 10 reduction. The relative reduction in the qubit overhead, associated with an order of magnitude improvement in the base error, is greater when the comparison is performed closer to the threshold of the code. For example from $2.8 \cdot 10^{-3}$ to $2.8 \cdot 10^{-4}$ the qubit reduction is instead a factor 30 (in contrast to the factor 10 of the previous comparison). This highlights the importance of reaching base physical error rates of 10^{-3} and lower.

6.3.3 Finding the optimal measurement depth

Logical algorithms may be optimized for particular properties, for example, either the total T gate count, T_{count}, the number of measurement layers, T_{depth}, or logical qubit count may be minimized [224, 225]. In the elliptic curve encryption breaking algorithm of Haner et al [215] logical requirements are stated for each of these three possible optimizations, where in the previous section the measurement depth minimized approach was chosen. The reaction limit (conjectured to be the fastest an algorithm can be run) is determined solely by the measurement depth and reaction time (i.e. independent of total gate count and code distance), and so the depth optimized approaches are the most suitable when room for parallelization is desired. The ratio of the measurement depth to total gate count is the inverse of the number of T gates per layer, T_{layer} (when considering T gates as opposed to some other non Clifford operation). In the GoSC method of parallelization with units, all aspects of the cost depend on the number of T gates per layer, including the footprint of the unit, the time it takes to prepare a unit, and the number T gates that are effectuated within the preparation time.

In figure 6.4 we plot the efficiency of the GoSC method as a function of this measurement depth with a fixed total gate count, and it can be seen that there is a measure-

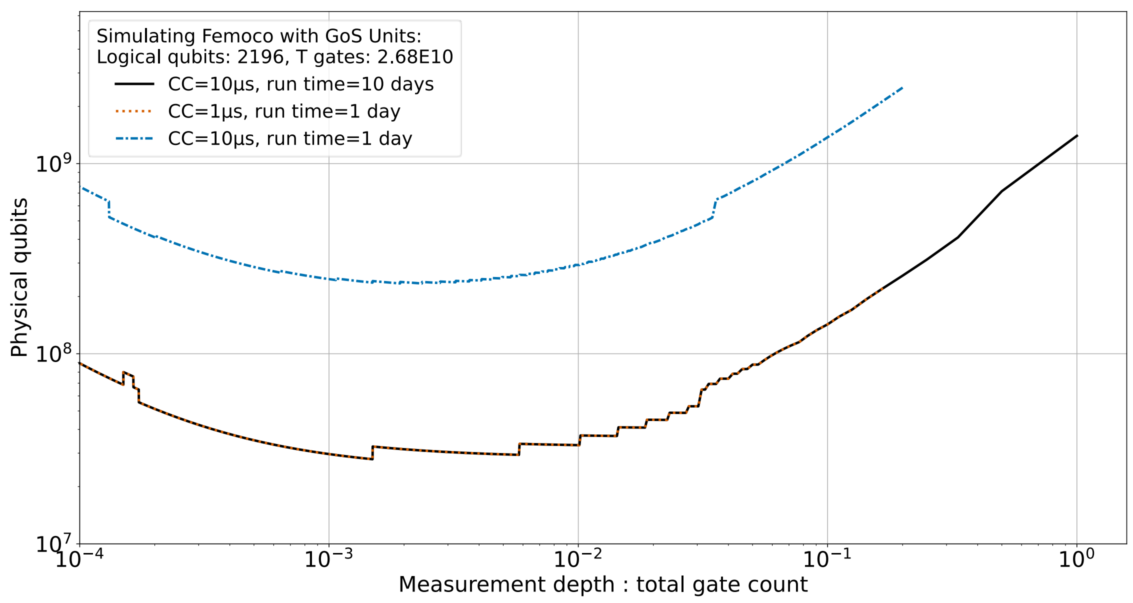


Figure 6.4: Investigating the efficiency of the Game of Surface Code Units approach [14] as a function of the ratio of the measurement depth to total gate count. The total qubit footprint is plotted for a particular desirable run time and code cycle time, with a base error of 10^{-3} and final output error of 6%. The trends start at the chosen measurement depth ratio of 10^{-4} and end when the reaction limit is reached before the desired run time. Plotting the required number of physical qubits for a ground state energy calculation of FeMoCo with the THC Qubitization method of Lee et al [212] to have a maximum run time as stated for code cycle times of 10 μs and 1 μs.

ment depth that leads to a minimum physical qubit footprint. This is in contrast to the AutoCCZ method where instead the measurement depth only plays a role in determining the reaction limit, and it would appear in these figures as a horizontal line. In figure 6.4 we plot the required number of physical qubits to reach a desired run time for the logical resources required to simulate FeMoco [212]. We include three plots for different code cycle times and maximum run times. As expected in this regime prior to reaction-limited, it can be seen that it is the ratio of the code cycle time and maximum run time that determines the physical qubit requirements. We show that the optimal measurement depth ratio is not necessarily "as small as possible", and for this situation it lies between 10^{-3} and 10^{-2}. The discontinuous movements result from an increase in the required number of units as the measurement depth ratio becomes larger. A less demanding run time requirement necessitates fewer units for a given measurement depth which results in less frequent and larger relative downward movements. The two distinct discontinuous movements on the blue trend (for a code cycle time of 10 μs and run time of 1 day) are due to the topological

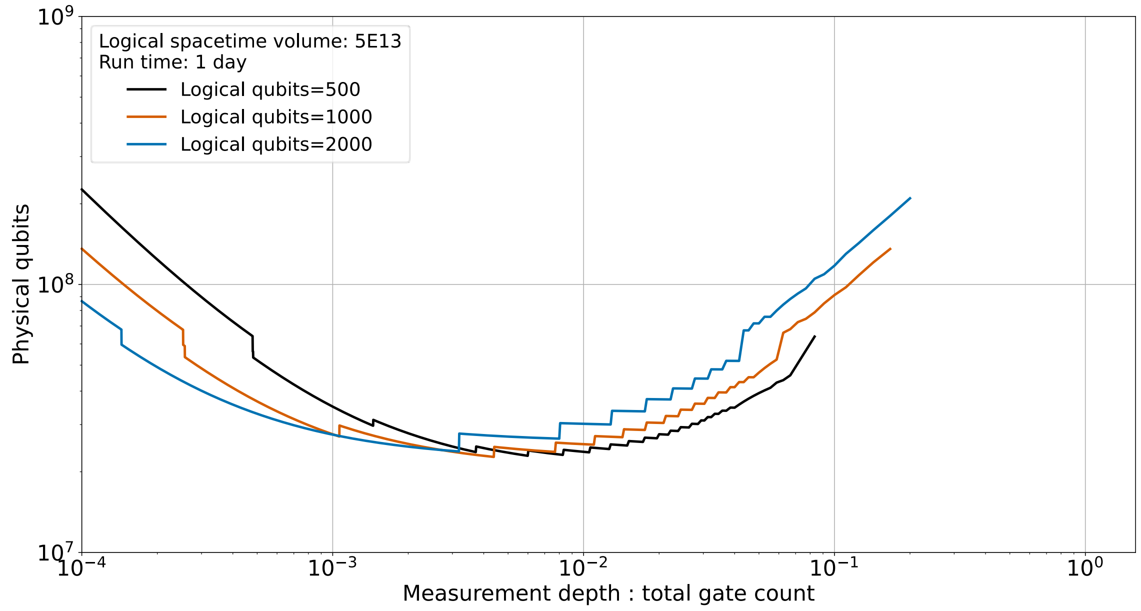


Figure 6.5: Investigating the efficiency of the Game of Surface Code Units approach [14] as a function of the ratio of the measurement depth to total gate count. The total qubit footprint is plotted for a particular desirable run time and code cycle time, with a base error of 10^{-3} and final output error of 6%. The trends start at the chosen measurement depth ratio of 10^{-4} and end when the reaction limit is reached before the desired run time. Plotting the required number of physical qubits for an algorithm with fixed space-time volume of $5 \cdot 10^{13}$, where the number of logical qubits is labeled and the total T gate count is varied to maintain the stated volume. The space-time volume was chosen to be representative of the quantum advantage cases assessed in this work.

code distance changing as the space time volume changes. The black trend with a run time of 10 days can maintain the desired run time for larger measurement depths than the dotted orange line because the proportional difference in the desired run time (a factor 10) is greater than the proportional difference in their associated reaction times. The relationship between the code cycle time and reaction time that we assume in this work is explained in section 6.2.3.

In figure 6.5 we again investigate the efficiency of the parallelization but now for an abstract algorithmic requirement with fixed space-time volume. Three trends are shown with logical qubits, n, corresponding to 500, 1000, and 2000, where the total T gate of the algorithm is set to maintain the fixed space time volume ($n \times T_{count}$) of $5 \cdot 10^{13}$. The trends end at the right when the reaction limit is reached where 2000 is the last to end because it has the lowest total T gate count, which in turn relates to a lower reaction limit for a fixed measurement depth ratio. The optimal measurement depth ratio is larger for lower logical

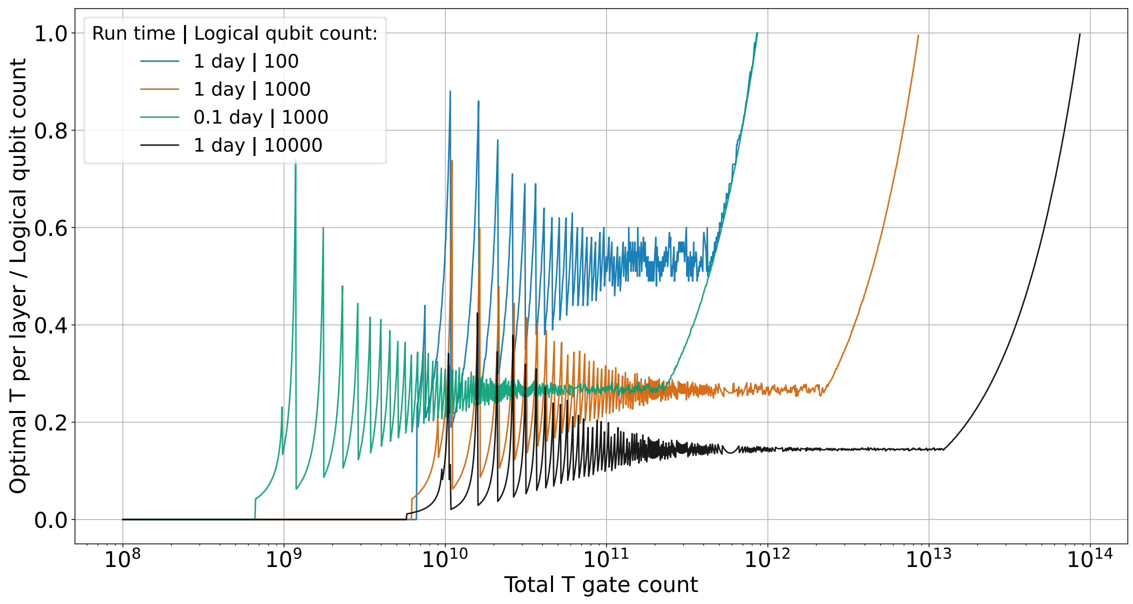


Figure 6.6: The optimal value of the number of T gates per measurement layer, shown as a fraction of the number of logical qubits, plotted as a function of the T gate count of the algorithm. Base physical error of 10^{-3} and code cycle time of $1\mu s$. Different plots for varied desired run times and logical qubit count. The maximum value of T gates per layer is limited to the number of logical qubits for each trend. For logical qubit numbers (N) in excess of 100, the equilibrium value of the optimal number of T gates per layer is well described by the following equation $T_{layer} = 1.9(5)N^{0.70(5)}$. The final phase starts when the algorithmic requirement becomes too demanding for the minimum run time and so the T_{layer} rises despite the loss in efficiency until the number of logical qubits is reached which here we define as the cut off point, where finally even the reaction limit (one reaction time per T layer) is not sufficient.

qubit counts; for 500 the optimal is $0.6 \cdot 10^{-3}$, whereas for 2000 the optimal is $0.3 \cdot 10^{-3}$. In the following we investigate the relationship between the optimal measurement depth and logical qubit requirement of the algorithm in more detail.

Optimal measurement depth and logical qubit requirement

We have shown that the optimal measurement depth given a fixed total gate count for the GoSC approach is non trivial and can be a function of both the number of logical qubits, and on how demanding the desired run time is (i.e. the number of units needed). In this section we investigate the relationship between the optimal measurement depth and the number of logical (abstract) qubits of the algorithm. The average number of T gates per measurement layer, T_{layer} is the inverse of the previously stated ratio of measurement depth to total gate count, i.e. $T_{layer} = T_{count}/T_{depth}$. In figure 6.6, for a particular value of the total T gate count, T_{count}, we calculate the optimal value of T gates per layer, T_{layer}, which is then converted into a ratio of the number of logical qubits. We include multiple trends for different numbers of logical qubits and plot as a function of the total T gate count. This involves identifying the value of T_{layer} at which the physical qubit requirement is minimized as in figure 6.4, for each particular logical resource requirement. We choose different qubit numbers and minimum run time values to highlight the various dependencies. We can see 4 distinct behavioural phases for each trend, first at low T gate counts the minimum run time can be reached with no parallelization and so we state the optimal T per layer as 0 (whereas truly it is independent). The next phase is an oscillating pattern where the optimal T per layer ratio varies widely with a decreasing amplitude as the T gate count increases. The beginning of this phase is determined by the end of the previous one, where the no-parallelization (*beat* limited) method can no longer reach the desired run time, which is determined by the ratio of the code cycle time and minimum run time. The amplitude of the oscillation is largest at the start of parallelization because only a small number of units are initially required and an increase in T count requirement can mean the number of required units increases, for example the optimal number of units may change from 3 to 4, but the associated T_{layer} for this minimum can vary widely, and the magnitude of that variation reduces as the overall number of required units increases. The large variation in the optimal ratio does not imply a large variation in the required number of physical qubits. In the next phase the optimal T per layer ratio is relatively constant at an equilibrium value which is solely determined by the number of logical (abstract) qubits in the algorithm, where the greater the number of logical qubits, the fewer T gates per layer (as a percentage of the logical qubits) are required for the optimal physical qubit overhead. For logical qubit numbers (N) in excess of 100, the equilibrium value of the optimal number of T gates per layer is well described by the following equation $T_{layer} = 1.9(5)N^{0.70(5)}$ which we estimated by taking an average of the points within the equilibrium phase and fit with linear regression. The final phase starts when the algorithmic requirement becomes too demanding for the minimum run time and so the T_{layer} rises despite the loss in efficiency until the number of logical qubits is reached which here we define as the cut off point, where finally even the reaction limit (one reaction time per T layer) is not sufficient.

The potential degree of control over the average number of T gates per layer during algorithm construction and optimization will determine whether it is beneficial to consider the optimal value as calculated with the techniques used in figure 6.6. There are optimization techniques that can minimize either the T_{depth} and T_{count} [224, 225], but they generally trade off with one another, where minimizing one may increase the other. Furthermore, this analysis is specific to the GoSC approach of parallelization with units, and earlier we have shown that the AutoCCZ method of parallelization produces considerably more favourable final resource estimates. As mentioned, the AutoCCZ method does not display this rich behaviour with the efficiency dependence on the measurement depth, and we believe further research is warranted to compare the underlying assumptions of these two methods of parallelization.

6.4 Summary

Within a particular time frame, the code cycle time and the number of achievable physical qubits may vary by orders of magnitude between hardware types. When envisaging a fault tolerant implementation, there are numerous decisions to be made based on a preference for either space or time. In this work we compare surface code strategies of parallelization that allow one to speed up the computation until the reaction limit is reached. Most of the fault tolerant resource estimation work has focused on code cycle times corresponding to superconducting architectures. A space optimized quantum advantage case study translated for hardware with slower code cycle times may lead to run times in excess of 1000 days, and so parallelization would have to be performed to reach desirable run times. In this work we have calculated the required number of physical qubits to reach a given desirable run time for two representative quantum advantage cases (chemistry and encryption) across a range of code cycle times. The feasibility of using these time optimization strategies will depend upon the number of physical qubits achievable within a device, therefore the scalability of an architecture will play an important role in determining whether a quantum advantage is achievable. We contrast two methods of parallelization to simulate the FeMo-Co cataylst, first a Game of Surface Codes approach which should be considered an upperbound, and second a more heuristic utilization of AutoCCZ factories. We find in this situation that the AutoCCZ factories produces more favourable resource estimates and the difference is mostly due to the high initial set up cost of parallelization with Game of Surface Code Units. With a single AutoCCZ factory a superconducting device with a 1 μs code cycle time would require 7.5 million qubits to simulate FeMo-co in ∼10 days, whereas a shuttling based trapped ion device with a 235 μs code cycle time would take 2450 days. By increasing the number of factories the space-time trade is initially favourable (as opposed to linear), and the trapped ion device can reach the same 10 day run-time with 600 million qubits. In this comparison the factor difference between the physical qubit requirement is $\sim 3\times$ less than the factor difference in the code cycle time.

We have investigated the effect of varying the ratio of the T gate count and T depth (the average T gates per layer) and identified the optimal value for a constrained run time against general algorithmic requirements. Here we focused on the GoSC Unit method of parallelization as it was unique in displaying rich T depth dependence.

We apply our methods to the logical resources required to break 256 elliptic curve encryption, which is used to secure public keys in the Bitcoin network. We use the logical resource requirements of the latest algorithmic developments which improve on the previous state of the art by ~ 2 orders of magnitude. There is a small window of time, approximately 10-60 minutes, in which the public keys are available and vulnerable after the initiation of a transaction. We quantify the number of physical qubits required to break the encryption in one hour as a function of code cycle time, and base physical error rate. It would require approximately 317 million physical qubits to break the encryption within one hour using the surface code and a code cycle time of 1 μs, a reaction time of $10\mu s$, and physical gate error of 10^{-3}. To instead break the encryption within one day, it would require *only* 13 million physical qubits. If the base physical error rate was instead the more optimistic value of 10^{-4}, 33 million physical qubits would be required to break the encryption in 1 hour. This large physical qubit requirement implies that the Bitcoin network will be secure from quantum computing attacks for many years (potentially over a decade). Alternative error correction techniques, in particular those which benefit from a more flexible physical qubit connectivity as often found in trapped ion based quantum computers, could potentially offer considerable improvements to the requirements but the slower rate of logical operations must also be factored in. The Bitcoin network could nullify this threat by performing a soft fork onto an encryption method that is quantum secure, but there may be serious scaling concerns associated with the switch. We hope to motivate continued research into end-to-end resource estimation for alternative error correction schemes to the surface code, and to determine how best to make use of the available physical connectivity of different quantum hardware platforms.

Chapter 7

Conclusion

7.1 Summary and future work

Over the years the research areas of quantum algorithms, applications, and hardware have to some extent been disparate. As the field as a whole has continued to evolve, so too have these areas continued to integrate. The work described in this thesis is a contribution towards this integration.

A routing algorithm for a shuttling based trapped ion design was presented which can effectively enable global connectivity without the use of positional swaps. A simulation tool was created to represent devices of variable size following this design, and it was used to aid in the development and characterization of the routing algorithms. The routing algorithm was thoroughly characterized by quantifying the time requirement for connectivity as a function of device size and ion density, and by quantifying the distribution of X-Junction center passes. The routing algorithm was found to compare favourably to an alternative method which instead made extensive use of positional swaps; the analysis incorporated state of the art experimental shuttling and positional swap speeds. The device sizes considered during this work were in general much larger than those that were experimentally available within the research group at the time. Now some years later, the core principles of the routing algorithm are being incorporated into a compiler which will control real quantum hardware. The creation of a full stack compiler that takes a high level quantum algorithm and decomposes it into machine level instructions is a monumental task that will require collaboration between a wide range of skill sets. Alternative optimization methods for routing may be investigated, such as machine learning techniques, but our simulation results show that there is not a large scope for improvement relative to the lower bound.

An error model for the shuttling based design was proposed that incorporates the cost for connectivity results of the routing simulation. The error model was then used to estimate the computational power (quantum volume) of near term devices as a function of experimental parameters. Quantum volume was used to compare the shuttling based design to a superconducting device as a function of two qubit gate fidelities. Across the range investigated the shuttling based design had a higher quantum volume due to the lower cost associated with enabling connectivity as compared to logical swap operations. Arbitrary two qubit gate decompositions into a native gate set were discussed and an approximate decomposition technique was investigated as a function of native two qubit gate fidelity. Quantum volume (actually $\log_2(QV)$, i.e. the maximum achievable square circuit depth) was used to meter experimental priorities by considering the impact of parameters such as ion loss, ion density, and coherence time. Single qubit gate fidelities were estimated as a function of the FPGA frequency (which determines the time resolution of control on the field), and the amplitude variation. The total time required to enable a single round of a globally connected algorithm was assessed across a wide range of device sizes (up to 10^7 physical qubits) for the shuttling based design, and for an alternative trapped ion design that uses small modules which are connected by a photonic interconnect. State of the art rates for photonic interconnects and high fidelity shuttling were used as well as favourable assumptions on the nature of the network of modules. Due to the high time cost for a single photonic interconnect operation, the shuttling based design was found to enable global connectivity in less time for devices of size up to 10^5 physical qubits, and over an order of magnitude faster for physical qubit numbers less than 1000. The metric quantum volume has gained in popularity since the original publication of this work and it is now being used by several commercial endeavours to compare experimental progress. In parallel with the continued advancement of the experimental hardware it would be desirable to develop a state simulator with a hardware inspired error model. Such a tool could be used to estimate the quantum volume of the design in a more rigorous manner, or to investigate the feasibility of small scale error correction techniques, or to optimize quantum algorithms for the hardware. Although some gate decomposition techniques were explored in this chapter, it will be necessary to develop automated gate decomposition and circuit optimization tools that can be applied to any arbitrary quantum circuit and which are specialized for this shuttling based design.

A pedagogical review for quantum chemistry was provided with the aim of answering the question “How will quantum computers provide an industrially relevant computational advantage in quantum chemistry?". In this work the focus was on algorithms which would require a fault tolerant quantum computer. The two primary Hamiltonian simulation techniques were compared for estimating the ground state energy for molecules of various sizes. The software development kit, Q#, was used to calculate the logical resource requirements for a single oracle application. Classical computing techniques were utilized to generate the initial starting guess for particular molecules for the quantum algorithm. End to end resource estimation was performed by considering the overheads associated with surface code error correction, which involved comparing different configurations of data blocks and distillation blocks. The approximate basis set size at which a quantum computer is expected to outperform a classical computer was identified for the molecule Cr_2. For this molecule, the physical resource requirements as a function of the base physical error were estimated. The physical resource estimations in this work required choosing hardware code cycle times; in this work $1\mu s$ was used which may correspond to future superconducting devices.

In the following chapter a broader range of hardware considerations were investigated. With the same error correction configurations a lower code cycle time will linearly increase the final run time requirement. The code cycle time of trapped ion devices may be on the order of $200\times$ slower than superconducting devices. By setting an upper limit to the desirable final run time and varying the error correction configuration we estimate the final physical qubit requirement as a function of the code cycle time. The focus was on problems with a strong commercial narrative such as ground state estimation of FeMoco and breaking the encryption of private keys in the Bitcoin network. Two distinct methods of parallelization in the surface code were compared, Game of Surface Code Units and AutoCCZ factories. The dependence of the T gate parallelization method on the ratio of the T gate depth and T gate count was investigated. Some of the latest developments in the surface code were utilized, such as improving the efficiency of distillation blocks by separately calibrating their required code distance. The potential for quantum computers to threaten the Bitcoin network was discussed. In the future it will be desirable to perform full stack resource requirements for alternative error correction techniques, such as Bacon-Shor, higher dimensional codes, or codes which require longer range physical interaction. For the shuttling based design of the research group, it will be important to consider the strengths of the hardware and how that may best align with the available error correction codes.

7.2 Outlook

The commercial interest behind quantum computing has radically increased since the start of my PhD. It remains to be seen whether a commercially relevant quantum advantage can be achieved with a NISQ type device, but as the hardware continues to develop so too will the understanding behind their potential computational power. It seems clear that a fault tolerant device with over a million physical qubits could run quantum algorithms that provide a serious advantage in areas such as quantum chemistry and machine learning. There are numerous platforms being pursued for quantum computing and it appears that each will have its own strengths and weaknesses relevant at different time scales. To achieve a fault tolerant device, the scalability of the underlying hardware is of primary importance, and this has been the main focus of the shuttling based trapped ion design of our research group. Trapped ion quantum computers are now seen as a major contending platform in the race to a powerful quantum computer, with several commercial endeavours including Honeywell, IonQ and Universal Quantum. A full stack compiler for the shuttling based design would take a high level quantum circuit, decompose (and optimize) into the native gate set, manage the ion's routing, cooling, and phase, all the way down to machine level instructions. This full stack compiler represents a significant undertaking and I look forward to contributing towards it in the future. I hope to help the field continue to develop by providing an algorithmic perspective to the hardware specialists and a hardware perspective to the algorithm specialists.

Bibliography

[1] R. P. Feynman, "Quantum mechanical computers," *Foundations of Physics*, vol. 16, pp. 507–531, 6 1986. 1

[2] T. Toffoli, "Reversible computing," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 85 LNCS, pp. 632–644, Springer Verlag, 1980. 1

[3] G. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, 4 1965. 2

[4] R. P. Feynman, "Simulating physics with computers," *International Journal of Theoretical Physics*, vol. 21, pp. 467–488, 6 1982. 3, 77, 106

[5] D. Deutsch, "Quantum Theory, The Church-Turing Princinple and the Universal Quantum Computer," *Proceedings of The Royal Society of London, Series A: Mathematical and Physical Sciences*, vol. 400, pp. 97–117, 7 1985. 3

[6] P. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," pp. 124–134, Institute of Electrical and Electronics Engineers (IEEE), 12 2002. 3, 123

[7] D. Gottesman, "The Heisenberg Representation of Quantum Computers," 7 1998. 6, 108

[8] D. Litinski and F. von Oppen, "Lattice surgery with a twist: Simplifying clifford gates of surface codes," *Quantum*, vol. 2, 2018. 6, 13, 93, 109

[9] C. M. Dawson and M. A. Nielsen, "The Solovay-Kitaev algorithm," *Quantum Information and Computation*, vol. 6, pp. 081–095, 1 2006. 6, 108

[10] D. P. DiVincenzo and IBM, "The Physical Implementation of Quantum Computation," *Fortschritte der Physik*, vol. 48, pp. 771–783, 2 2000. 6

[11] M. Webber, S. Herbert, S. Weidt, and W. K. Hensinger, “Efficient Qubit Routing for a Globally Connected Trapped Ion Quantum Computer,” *Advanced Quantum Technologies*, vol. 3, p. 2000027, 8 2020. 9, 17, 32, 51, 104, 114

[12] V. E. Elfving, B. W. Broer, M. Webber, J. Gavartin, M. D. Halls, K. P. Lorton, and A. Bochevarov, “How will quantum computers provide an industrially relevant computational advantage in quantum chemistry?,” 9 2020. 9, 76, 79, 89, 92, 93, 94

[13] M. Webber, V. Elfving, S. Weidt, and W. K. Hensinger, “The impact of hardware specifications on reaching quantum advantage in the fault tolerant regime,” *AVS Quantum Science*, vol. 4, p. 013801, 8 2022. 9, 102

[14] D. Litinski, “A game of surface codes: Large-scale quantum computing with lattice surgery,” *Quantum*, vol. 3, 2019. 9, 33, 92, 93, 95, 96, 98, 105, 112, 115, 116, 119, 120, 122, 128, 129

[15] D. Aharonov and M. Ben-Or, “Fault-tolerant quantum computation with constant error rate,” *SIAM Journal on Computing*, vol. 38, pp. 1207–1282, 7 2008. 10, 104

[16] A. Y. Kitaev, “Fault-tolerant quantum computation by anyons,” *Annals of Physics*, vol. 303, pp. 2–30, 1 2003. 10, 104

[17] E. Knill, R. Laflamme, and W. H. Zurek, “Resilient Quantum Computation: Error Models and Thresholds,” *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 454, pp. 365–384, 2 1997. 10, 104

[18] J. Preskill, “Quantum computing in the NISQ era and beyond,” *Quantum*, vol. 2, p. 79, 8 2018. 10

[19] N. Moll, P. Barkoutsos, L. S. Bishop, J. M. Chow, A. Cross, D. J. Egger, S. Filipp, A. Fuhrer, J. M. Gambetta, M. Ganzhorn, A. Kandala, A. Mezzacapo, P. Müller, W. Riess, G. Salis, J. Smolin, I. Tavernelli, and K. Temme, “Quantum optimization using variational algorithms on near-term quantum devices,” *Quantum Science and Technology*, vol. 3, 10 2018. 11, 51, 52, 53

[20] A. W. Cross, L. S. Bishop, S. Sheldon, P. D. Nation, and J. M. Gambetta, “Validating quantum computers using randomized model circuits,” *Physical Review A*, vol. 100, no. 3, 2019. 11, 32, 51, 52, 53, 55, 56, 61, 104

[21] D. S. Wang, A. G. Fowler, and L. C. Hollenberg, "Surface code quantum computing with error rates over 1%," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 83, p. 020302, 2 2011. 11

[22] W. K. Wootters and W. H. Zurek, "A single quantum cannot be cloned," *Nature*, vol. 299, no. 5886, pp. 802–803, 1982. 12

[23] P. W. Shor, "Scheme for reducing decoherence in quantum computer memory," *Physical Review A*, vol. 52, p. R2493, 10 1995. 12

[24] J. Roffe, "Quantum error correction: an introductory guide," *Contemporary Physics*, vol. 60, no. 3, pp. 226–245, 2019. 12

[25] A. G. Fowler and C. Gidney, "Low overhead quantum computation using lattice surgery," *arXiv: 1808.06709*, 8 2018. 13, 33, 97, 109

[26] S. Bravyi and A. Kitaev, "Universal quantum computation with ideal Clifford gates and noisy ancillas," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 71, p. 022316, 2 2005. 13, 109

[27] D. Litinski, "Magic State Distillation: Not as Costly as You Think," *Quantum*, vol. 3, 5 2019. 13, 97, 109, 112, 115, 118, 122

[28] S. Bravyi and J. Haah, "Magic-state distillation with low overhead," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 86, p. 052329, 11 2012. 13, 109

[29] A. G. Fowler, S. J. Devitt, and C. Jones, "Surface code implementation of block code state distillation," *Scientific Reports*, vol. 3, pp. 1–6, 6 2013. 13, 109

[30] J. Roffe, D. R. White, S. Burton, and E. T. Campbell, "Decoding Across the Quantum LDPC Code Landscape," *Physical Review Research*, vol. 2, p. 043423, 5 2020. 14, 33, 72, 104, 115

[31] G. Wendin, "Quantum information processing with superconducting circuits: A review," *Reports on Progress in Physics*, vol. 80, 9 2017. 15

[32] C. D. Bruzewicz, J. Chiaverini, R. McConnell, and J. M. Sage, "Trapped-ion quantum computing: Progress and challenges," *Applied Physics Reviews*, vol. 6, p. 021314, 6 2019. 15

[33] M. Saffman, "Quantum computing with neutral atoms," *National Science Review*, vol. 6, pp. 24–25, 1 2019. 15

[34] C. Kloeffel and D. Loss, “Prospects for spin-based quantum computing in quantum dots,” *Annual Review of Condensed Matter Physics*, vol. 4, pp. 51–81, 4 2013. 15

[35] T. Xin, B. X. Wang, K. R. Li, X. Y. Kong, S. J. Wei, T. Wang, D. Ruan, and G. L. Long, “Nuclear magnetic resonance for quantum computing: Techniques and recent achievements,” *Chinese Physics B*, vol. 27, p. 020308, 2 2018. 15

[36] S. Slussarenko and G. J. Pryde, “Photonic quantum information processing: A concise review,” 2019. 15, 17, 18

[37] L. Childress and R. Hanson, “Diamond NV centers for quantum computing and quantum networks,” *MRS Bulletin*, vol. 38, pp. 134–138, 2 2013. 15

[38] Y. Nakamura, Y. A. Pashkin, and J. S. Tsai, “Coherent control of macroscopic quantum states in a single-Cooper-pair box,” *Nature*, vol. 398, pp. 786–788, 3 1999. 16

[39] J. Koch, T. M. Yu, J. Gambetta, A. A. Houck, D. I. Schuster, J. Majer, A. Blais, M. H. Devoret, S. M. Girvin, and R. J. Schoelkopf, “Charge-insensitive qubit design derived from the Cooper pair box,” *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 76, no. 4, 2007. 16

[40] E. Il’ichev and G. Oelsner, “Superconducting Qubits,” in *Wiley Encyclopedia of Electrical and Electronics Engineering*, pp. 1–13, 2018. 16

[41] H. Zu, W. Dai, and A. T. de Waele, “Development of dilution refrigerators—A review,” *Cryogenics*, vol. 121, p. 103390, 1 2022. 16

[42] L. Geck, A. Kruth, H. Bluhm, S. van Waasen, and S. Heinen, “Control Electronics For Semiconductor Spin Qubits,” *Quantum Science and Technology*, vol. 5, 3 2019. 16

[43] L. Dicarlo, M. D. Reed, L. Sun, B. R. Johnson, J. M. Chow, J. M. Gambetta, L. Frunzio, S. M. Girvin, M. H. Devoret, and R. J. Schoelkopf, “Preparation and measurement of three-qubit entanglement in a superconducting circuit,” *Nature*, vol. 467, no. 7315, pp. 574–578, 2010. 16

[44] G. Park, G. Choi, J. Choi, J. Choi, S. G. Lee, K. W. Lee, W. Song, and Y. Chong, “Observation of a Strongly Enhanced Relaxation Time of an In-situ Tunable Transmon on a Silicon Substrate up to the Purcell Limit Approaching 100 μs,” *Journal of the Korean Physical Society*, vol. 76, pp. 1029–1034, 6 2020. 16

[45] I. Tsioutsios, K. Serniak, S. Diamond, V. V. Sivak, Z. Wang, S. Shankar, L. Frunzio, R. J. Schoelkopf, and M. H. Devoret, “Free-standing silicon shadow masks for transmon qubit fabrication,” *AIP Advances*, vol. 10, no. 6, 2020. 16

[46] A. Gyenis, P. S. Mundada, A. Di Paolo, T. M. Hazard, X. You, D. I. Schuster, J. Koch, A. Blais, and A. A. Houck, “Experimental Realization of a Protected Superconducting Circuit Derived from the $0-\pi$ Qubit,” *PRX Quantum*, vol. 2, no. 1, 2021. 16

[47] E. Ferraro, D. Rei, M. Paris, and M. De Michielis, “Universal set of quantum gates for the flip-flop qubit in the presence of 1/f noise,” *EPJ Quantum Technology*, vol. 9, no. 1, 2022. 16

[48] D. C. McKay, S. Filipp, A. Mezzacapo, E. Magesan, J. M. Chow, and J. M. Gambetta, “Universal Gate for Fixed-Frequency Qubits via a Tunable Bus,” *Physical Review Applied*, vol. 6, no. 6, 2016. 17

[49] Y. Xu, J. Chu, J. Yuan, J. Qiu, Y. Zhou, L. Zhang, X. Tan, Y. Yu, S. Liu, J. Li, F. Yan, and D. Yu, “High-Fidelity, High-Scalability Two-Qubit Gate Scheme for Superconducting Qubits,” *Physical Review Letters*, vol. 125, p. 240503, 12 2020. 17

[50] A. Cowtan, S. Dilkes, R. Duncan, A. Krajenbrink, W. Simmons, and S. Sivarajah, “On the qubit routing problem,” in *Leibniz International Proceedings in Informatics, LIPIcs* (W. van Dam and L. Mancinska, eds.), vol. 135, (Maryland, USA), p. /, 6 2019. 17, 61

[51] M. Reck, A. Zeilinger, H. J. Bernstein, and P. Bertani, “Experimental realization of any discrete unitary operator,” *Physical Review Letters*, vol. 73, pp. 58–61, 7 1994. 17

[52] E. Knill, R. Laflamme, and G. J. Milburn, “A scheme for efficient quantum computation with linear optics,” *Nature*, vol. 409, pp. 46–52, 1 2001. 17

[53] R. Raussendorf and H. J. Briegel, “A one-way quantum computer,” *Physical Review Letters*, vol. 86, pp. 5188–5191, 5 2001. 17

[54] M. A. Nielsen, “Optical quantum computation using cluster states,” *Physical Review Letters*, vol. 93, p. 040503, 7 2004. 17

[55] T. Rudolph, “Why i am optimistic about the silicon-photonic route to quantum computing,” *APL Photonics*, vol. 2, p. 30901, 3 2017. 18

[56] D. J. Wineland, R. E. Drullinger, and F. L. Walls, "Radiation-pressure cooling of bound resonant absorbers," *Physical Review Letters*, vol. 40, pp. 1639–1642, 6 1978. 18

[57] W. Neuhauser, M. Hohenstatt, P. E. Toschek, and H. Dehmelt, "Localized visible Ba+ mono-ion oscillator," *Physical Review A*, vol. 22, pp. 1137–1140, 9 1980. 18

[58] J. I. Cirac and P. Zoller, "Quantum computations with cold trapped ions," *Physical Review Letters*, vol. 74, pp. 4091–4094, 5 1995. 18

[59] C. Monroe, D. M. Meekhof, B. E. King, W. M. Itano, and D. J. Wineland, "Demonstration of a fundamental quantum logic gate," *Physical Review Letters*, vol. 75, pp. 4714–4717, 12 1995. 18

[60] K. Mølmer and A. Sørensen, "Multiparticle entanglement of hot trapped ions," *Physical Review Letters*, vol. 82, pp. 1835–1838, 10 1999. 18, 55, 56, 61

[61] R. Brédy, J. Bernard, L. Chen, G. Montagne, B. Li, and S. Martin, "An introduction to the trapping of clusters with ion traps and electrostatic storage devices," *Journal of Physics B: Atomic, Molecular and Optical Physics*, vol. 42, p. 154023, 7 2009. 19

[62] C. J. Ballance, T. P. Harty, N. M. Linke, and D. M. Lucas, "High-fidelity two-qubit quantum logic gates using trapped calcium-43 ions," *Physical Review Letters*, vol. 117, 6 2014. 19

[63] F. Mintert and C. Wunderlich, "Ion-trap quantum logic using long-wavelength radiation," *Physical Review Letters*, vol. 87, pp. 257904–1, 4 2001. 19

[64] S. Weidt, J. Randall, S. C. Webster, K. Lake, A. E. Webb, I. Cohen, T. Navickas, B. Lekitsch, A. Retzker, and W. K. Hensinger, "Trapped-Ion Quantum Logic with Global Radiation Fields," *Physical Review Letters*, vol. 117, 3 2016. 19, 22

[65] T. P. Harty, M. A. Sepiol, D. T. Allcock, C. J. Ballance, J. E. Tarlton, and D. M. Lucas, "High-Fidelity Trapped-Ion Quantum Logic Using Near-Field Microwaves," *Physical Review Letters*, vol. 117, p. 140501, 9 2016. 19, 20, 67

[66] C. Piltz, T. Sriarunothai, S. S. Ivanov, S. Wölk, and C. Wunderlich, "Versatile microwave-driven trapped ion spin system for quantum information processing," *Science Advances*, vol. 2, 9 2016. 19

[67] R. Srinivas, S. C. Burd, R. T. Sutherland, A. C. Wilson, D. J. Wineland, D. Leibfried, D. T. Allcock, and D. H. Slichter, "Trapped-Ion Spin-Motion Coupling with Microwaves and a Near-Motional Oscillating Magnetic Field Gradient," *Physical Review Letters*, vol. 122, p. 163201, 4 2019. 19

[68] A. E. Webb, S. C. Webster, S. Collingbourne, D. Bretaud, A. M. Lawrence, S. Weidt, F. Mintert, and W. K. Hensinger, "Resilient Entangling Gates for Trapped Ions," *Physical Review Letters*, vol. 121, no. 18, 2018. 20, 67

[69] G. Zarantonello, H. Hahn, J. Morgner, M. Schulte, A. Bautista-Salvador, R. F. Werner, K. Hammerer, and C. Ospelkaus, "Robust and Resource-Efficient Microwave Near-Field Entangling Be+ 9 Gate," *Physical Review Letters*, vol. 123, p. 260503, 12 2019. 20, 67

[70] J. P. Gaebler, T. R. Tan, Y. Lin, Y. Wan, R. Bowler, A. C. Keith, S. Glancy, K. Coakley, E. Knill, D. Leibfried, and D. J. Wineland, "High-Fidelity Universal Gate Set for Be 9 + Ion Qubits," *Physical Review Letters*, vol. 117, p. 060505, 8 2016. 20, 67

[71] C. J. Ballance, T. P. Harty, N. M. Linke, M. A. Sepiol, and D. M. Lucas, "High-Fidelity Quantum Logic Gates Using Trapped-Ion Hyperfine Qubits," *Physical Review Letters*, vol. 117, p. 060504, 8 2016. 20, 67

[72] V. M. Schäfer, C. J. Ballance, K. Thirumalai, L. J. Stephenson, T. G. Ballance, A. M. Steane, and D. M. Lucas, "Fast quantum logic gates with trapped-ion qubits," *Nature 2018 555:7694*, vol. 555, pp. 75–78, 3 2018. 20

[73] R. Srinivas, S. C. Burd, H. M. Knaack, R. T. Sutherland, A. Kwiatkowski, S. Glancy, E. Knill, D. J. Wineland, D. Leibfried, A. C. Wilson, D. T. C. Allcock, and D. H. Slichter, "High-fidelity laser-free universal control of two trapped ion qubits," 2021. 20

[74] B. Lekitsch, S. Weidt, A. G. Fowler, K. Mølmer, S. J. Devitt, C. Wunderlich, and W. K. Hensinger, "Blueprint for a microwave trapped ion quantum computer," *Science Advances*, vol. 3, 2 2017. 20, 21, 22, 23, 32, 33, 60, 61, 105, 114, 120, 121, 124

[75] P. Murali, D. M. Debroy, K. R. Brown, and M. Martonosi, "Architecting Noisy Intermediate-Scale Trapped Ion Quantum Computers," in *Proceedings - International Symposium on Computer Architecture*, vol. 2020-May, pp. 529–542, 2020. 20

[76] J. Zhang, G. Pagano, P. W. Hess, A. Kyprianidis, P. Becker, H. Kaplan, A. V. Gorshkov, Z. X. Gong, and C. Monroe, "Observation of a Many-Body Dynamical Phase Transition with a 53-Qubit Quantum Simulator," *Nature*, vol. 551, pp. 601–604, 8 2017. 20

[77] P. H. Leung and K. R. Brown, "Entangling an arbitrary pair of qubits in a long ion crystal," *Physical Review A*, vol. 98, 8 2018. 20

[78] D. J. Wineland, C. Monroe, W. M. Itano, D. Leibfried, B. E. King, and D. M. Meekhof, "Experimental issues in coherent quantum-state manipulation of trapped atomic ions," *Journal of Research of the National Institute of Standards and Technology*, vol. 103, pp. 259–328, 10 1997. 20

[79] D. Kielpinski, C. Monroe, and D. J. Wineland, "Architecture for a large-scale ion-trap quantum computer," *Nature*, vol. 417, pp. 709–711, 6 2002. 20

[80] C. Monroe and J. Kim, "Scaling the ion trap quantum processor," *Science*, vol. 339, pp. 1164–1169, 3 2013. 21

[81] S. Kasture, F. Lenzini, B. Haylock, A. Boes, A. Mitchell, E. W. Streed, and M. Lobino, "Frequency conversion between UV and telecom wavelengths in a lithium niobate waveguide for quantum communication with Yb+ trapped ions," *Journal of Optics (United Kingdom)*, vol. 18, p. 104007, 9 2016. 21

[82] L. J. Stephenson, D. P. Nadlinger, B. C. Nichol, S. An, P. Drmota, T. G. Ballance, K. Thirumalai, J. F. Goodwin, D. M. Lucas, and C. J. Ballance, "High-Rate, High-Fidelity Entanglement of Qubits Across an Elementary Quantum Network," *Physical Review Letters*, vol. 124, no. 11, 2020. 21, 72, 73, 105

[83] R. Nigmatullin, C. J. Ballance, N. D. Beaudrap, and S. C. Benjamin, "Minimally complex ion traps as modules for quantum communication and computing," *New Journal of Physics*, vol. 18, p. 103028, 10 2016. 21, 72, 73, 105

[84] A. Walther, F. Ziesel, T. Ruster, S. T. Dawkins, K. Ott, M. Hettrich, K. Singer, F. Schmidt-Kaler, and U. Poschinger, "Controlling fast transport of cold trapped ions," *Physical Review Letters*, vol. 109, p. 080501, 8 2012. 21, 47, 58

[85] P. Kaufmann, T. F. Gloger, D. Kaufmann, M. Johanning, and C. Wunderlich, "High-Fidelity Preservation of Quantum Information during Trapped-Ion Trans-

port," *Physical Review Letters*, vol. 120, no. 1, 2018. 21, 48, 58, 60, 62, 66, 72, 73, 114

[86] W. K. Hensinger, S. Olmschenk, D. Stick, D. Hucul, M. Yeo, M. Acton, L. Deslauriers, C. Monroe, and J. Rabchuk, "T-junction ion trap array for two-dimensional ion shuttling, storage, and manipulation," *Applied Physics Letters*, vol. 88, no. 3, pp. 1–3, 2006. 22

[87] J. M. Amini, H. Uys, J. H. Wesenberg, S. Seidelin, J. Britton, J. J. Bollinger, D. Leibfried, C. Ospelkaus, A. P. VanDevender, and D. J. Wineland, "Scalable ion traps for quantum information processing," *New Journal of Physics*, vol. 12, 9 2009. 22

[88] R. B. Blakestad, C. Ospelkaus, A. P. Vandevender, J. M. Amini, J. Britton, D. Leibfried, and D. J. Wineland, "High-fidelity transport of trapped-ion qubits through an X-junction trap array," *Physical Review Letters*, vol. 102, 4 2009. 22

[89] K. Wright, J. M. Amini, D. L. Faircloth, C. Volin, S. Charles Doret, H. Hayden, C. S. Pai, D. W. Landgren, D. Denison, T. Killian, R. E. Slusher, and A. W. Harter, "Reliable transport through a microfabricated X-junction surface-electrode ion trap," *New Journal of Physics*, vol. 15, p. 033004, 3 2013. 22, 59, 60, 61, 65

[90] M. Palmero, S. Martínez-Garaot, U. G. Poschinger, A. Ruschhaupt, and J. G. Muga, "Fast separation of two trapped ions," *New Journal of Physics*, vol. 17, no. 9, p. 93031, 2015. 22

[91] M. W. Van Mourik, E. A. Martinez, L. Gerster, P. Hrmo, T. Monz, P. Schindler, and R. Blatt, "Coherent rotations of qubits within a surface ion-trap quantum computer," *Physical Review A*, vol. 102, no. 2, 2020. 22, 47

[92] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, W.-K. Mok, S. Sim, L.-C. Kwek, and A. Aspuru-Guzik, "Noisy intermediate-scale quantum (NISQ) algorithms," 2021. 24

[93] S. Wang, E. Fontana, M. Cerezo, K. Sharma, A. Sone, L. Cincio, and P. J. Coles, "Noise-Induced Barren Plateaus in Variational Quantum Algorithms," *Arxiv: 2007.14384*, 2020. 24

[94] A. Peruzzo, J. McClean, P. Shadbolt, M. H. Yung, X. Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, "A variational eigenvalue solver on a photonic quantum processor," *Nature Communications*, vol. 5, pp. 1–7, 4 2014. 24, 78, 80, 106

[95] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, "The theory of variational hybrid quantum-classical algorithms," *New Journal of Physics*, vol. 18, p. 23023, 2 2016. 24

[96] D. Wecker, M. B. Hastings, and M. Troyer, "Progress towards practical quantum variational algorithms," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 92, p. 042303, 10 2015. 24

[97] D. Wang, O. Higgott, and S. Brierley, "Accelerated variational quantum eigensolver," *Physical Review Letters*, vol. 122, no. 14, 2019. 25

[98] C. Froese Fischer, "General Hartree-Fock program," *Computer Physics Communications*, vol. 43, no. 3, pp. 355–365, 1987. 25

[99] D. A. Fedorov, B. Peng, N. Govind, and Y. Alexeev, "VQE Method: A Short Survey and Recent Developments," *arXiv: 2103.08505*, 2021. 25

[100] Y. Nam, J. S. Chen, N. C. Pisenti, K. Wright, C. Delaney, D. Maslov, K. R. Brown, S. Allen, J. M. Amini, J. Apisdorf, K. M. Beck, A. Blinov, V. Chaplin, M. Chmielewski, C. Collins, S. Debnath, K. M. Hudek, A. M. Ducore, M. Keesan, S. M. Kreikemeier, J. Mizrahi, P. Solomon, M. Williams, J. D. Wong-Campos, D. Moehring, C. Monroe, and J. Kim, "Ground-state energy estimation of the water molecule on a trapped-ion quantum computer," *npj Quantum Information*, vol. 6, 2 2020. 25

[101] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, S. Boixo, M. Broughton, B. B. Buckley, D. A. Buell, B. Burkett, N. Bushnell, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, S. Demura, A. Dunsworth, D. Eppens, E. Farhi, A. Fowler, B. Foxen, C. Gidney, M. Giustina, R. Graff, S. Habegger, M. P. Harrigan, A. Ho, S. Hong, T. Huang, W. J. Huggins, L. Ioffe, S. V. Isakov, E. Jeffrey, Z. Jiang, C. Jones, D. Kafri, K. Kechedzhi, J. Kelly, S. Kim, P. V. Klimov, A. Korotkov, F. Kostritsa, D. Landhuis, P. Laptev, M. Lindmark, E. Lucero, O. Martin, J. M. Martinis, J. R. McClean, M. McEwen, A. Megrant, X. Mi, M. Mohseni, W. Mruczkiewicz, J. Mutus, O. Naaman, M. Neeley, C. Neill, H. Neven, M. Y. Niu, T. E. O'Brien, E. Ostby, A. Petukhov, H. Putterman, C. Quintana, P. Roushan,

N. C. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, D. Strain, K. J. Sung, M. Szalay, T. Y. Takeshita, A. Vainsencher, T. White, N. Wiebe, Z. J. Yao, P. Yeh, and A. Zalcman, “Hartree-Fock on a superconducting qubit quantum computer,” *Science*, vol. 369, pp. 1084–1089, 4 2020. 25

[102] E. Farhi, J. Goldstone, and S. Gutmann, “A Quantum Approximate Optimization Algorithm,” *ArXiv: 1411.4028*, 11 2014. 25

[103] B. Barak, A. Moitra, R. O’Donnell, P. Raghavendra, O. Regev, D. Steurer, L. Trevisan, A. Vijayaraghavan, D. Witmer, and J. Wright, “Beating the random assignment on constraint satisfaction problems of bounded degree,” *Leibniz International Proceedings in Informatics, LIPIcs*, vol. 40, pp. 110–123, 5 2015. 25

[104] S. Hadfield, Z. Wang, B. O’Gorman, E. G. Rieffel, D. Venturelli, and R. Biswas, “From the Quantum Approximate Optimization Algorithm to a Quantum Alternating Operator Ansatz,” *Algorithms*, vol. 12, 9 2017. 25

[105] A. Bouland, W. van Dam, H. Joorati, I. Kerenidis, and A. Prakash, “Prospects and challenges of quantum finance,” *ArXiv: 2011.06492*, 2020. 25, 30

[106] G. G. Guerreschi and A. Y. Matsuura, “QAOA for Max-Cut requires hundreds of qubits for quantum speed-up,” *Scientific Reports*, vol. 9, pp. 1–7, 12 2019. 25

[107] J. W. Cooley and J. W. Tukey, “An Algorithm for the Machine Calculation of Complex Fourier Series,” *Mathematics of Computation*, vol. 19, p. 297, 4 1965. 26

[108] D. Coppersmith, “An approximate Fourier transform useful in quantum factoring,” *ArXiv: quant-ph/0201067*, 1 2002. 26

[109] D. Camps, R. Van Beeumen, and C. Yang, “Quantum Fourier transform revisited,” *Numerical Linear Algebra with Applications*, vol. 28, no. 1, 2021. 26

[110] M. Dobšíček, G. Johansson, V. Shumeiko, and G. Wendin, “Arbitrary accuracy iterative quantum phase estimation algorithm using a single ancillary qubit: A two-qubit benchmark,” *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 76, no. 3, 2007. 27, 87

[111] P. W. Shor, “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer,” *SIAM Journal on Computing*, vol. 26, pp. 1484–1509, 7 1997. 27, 107, 125

[112] C. Pomerance, "A Tale of Two Sieves," in *Biscuits of Number Theory*, pp. 85–104, 2009. 27

[113] C. Gidney and M. Ekerå, "How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits," *Quantum*, vol. 5, pp. 1–31, 5 2021. 27, 74, 105, 108, 111, 115, 117, 119, 120, 124

[114] L. K. Grover, "A fast quantum mechanical algorithm for database search," *Proceedings of the Annual ACM Symposium on Theory of Computing*, vol. Part F1294, pp. 212–219, 5 1996. 27, 30, 107

[115] G. Brassard, "An Exact Quantum Polynomial-Time Algorithm for Simon's Problem," *arXiv: 9704027*, pp. 12–23, 1997. 27

[116] L. K. Grover, "Quantum computers can search rapidly by using almost any transformation," *Physical Review Letters*, vol. 80, pp. 4329–4332, 12 1997. 27

[117] G. Brassard, P. Høyer, M. Mosca, and A. Tapp, "Quantum amplitude amplification and estimation," pp. 53–74, 5 2002. 27, 30

[118] A. Montanaro, "Quantum algorithms: An overview," *npj Quantum Information*, vol. 2, no. 1, 2016. 28

[119] W. Buchanan and A. Woodward, "Will quantum computers be the end of public key encryption?," *Journal of Cyber Security Technology*, vol. 1, no. 1, pp. 1–22, 2017. 28

[120] K. Plekhanov, M. Rosenkranz, M. Fiorentini, and M. Lubasch, "Variational quantum amplitude estimation," 2021. 28

[121] C. Gidney and A. G. Fowler, "Efficient magic state factories with a catalyzed —CCZi ? 2—Ti transformation," *Quantum*, vol. 3, 12 2019. 28, 89, 92, 95, 109, 124, 125

[122] A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum algorithm for solving linear systems of equations," *Physical Review Letters*, vol. 103, 11 2008. 28

[123] P. Rebentrost, M. Mohseni, and S. Lloyd, "Quantum support vector machine for big data classification," *Physical Review Letters*, vol. 113, p. 130503, 9 2014. 28

[124] S. Lloyd, M. Mohseni, and P. Rebentrost, "Quantum algorithms for supervised and unsupervised machine learning," *arXiv: 1307.0411*, 7 2013. 28

[125] S. Aaronson, "Read the fine print," *Nature Physics*, vol. 11, pp. 291–293, 4 2015. 29

[126] O. D. Matteo, V. Gheorghiu, and M. Mosca, "Fault-Tolerant Resource Estimation of Quantum Random-Access Memories," *IEEE Transactions on Quantum Engineering*, vol. 1, pp. 1–13, 2 2021. 29

[127] M. Hodson, B. Ruck, H. Ong, D. Garvin, and S. Dulman, "Portfolio rebalancing experiments using the Quantum Alternating Operator Ansatz," *arXiv: 1911.05296*, 11 2019. 30

[128] I. Kerenidis, A. Prakash, and D. Szilágyi, "Quantum algorithms for portfolio optimization," in *AFT 2019 - Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, pp. 147–155, 2019. 30

[129] A. Montanaro, "Quantum speedup of Monte Carlo methods," in *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 471, Royal Society of London, 9 2015. 30

[130] T. Giurgica-Tiron, I. Kerenidis, F. Labib, A. Prakash, and W. Zeng, "Low depth algorithms for quantum amplitude estimation," *arXiv: 2012.03348*, 12 2020. 30

[131] D. J. Egger, C. Gambella, J. Marecek, S. McFaddin, M. Mevissen, R. Raymond, A. Simonetto, S. Woerner, and E. Yndurain, "Quantum Computing for Finance: State-of-the-Art and Future Prospects," *IEEE Transactions on Quantum Engineering*, vol. 1, pp. 1–24, 2021. 31

[132] A. G. Fowler, A. M. Stephens, and P. Groszkowski, "High-threshold universal quantum computation on the surface code," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 80, p. 052312, 11 2009. 33

[133] D. Maslov, "Basic circuit compilation techniques for an ion-trap quantum machine," *New Journal of Physics*, vol. 19, no. 2, 2017. 34, 55, 56

[134] M. Webber, "Depth model - Available at: https://github.com/mawebber1/Calculataing-achievable-depth-for-Trapped-Ions." 35, 62

[135] H. Kaufmann, T. Ruster, C. T. Schmiegelow, M. A. Luda, V. Kaushal, J. Schulz, D. Von Lindenfels, F. Schmidt-Kaler, and U. G. Poschinger, "Fast ion swapping for quantum-information processing," *Physical Review A*, vol. 95, p. 052319, 5 2017. 36, 47

[136] E. Magesan, J. M. Gambetta, and J. Emerson, "Characterizing quantum gates via randomized benchmarking," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 85, p. 042311, 4 2012. 53

[137] S. T. Merkel, J. M. Gambetta, J. A. Smolin, S. Poletto, A. D. Córcoles, B. R. Johnson, C. A. Ryan, and M. Steffen, "Self-consistent quantum process tomography," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 87, p. 062119, 6 2013. 53

[138] S. Aaronson and L. Chen, "Complexity-theoretic foundations of quantum supremacy experiments," *Leibniz International Proceedings in Informatics, LIPIcs*, vol. 79, 12 2017. 53

[139] D. Maslov and Y. Nam, "Use of global interactions in efficient quantum circuit constructions," *New Journal of Physics*, vol. 20, 7 2017. 54

[140] F. Vatan and C. Williams, "Optimal quantum circuits for general two-qubit gates," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 69, p. 032315, 3 2004. 55, 56

[141] M. Blaauboer and R. L. De Visser, "An analytical decomposition protocol for optimal implementation of two-qubit entangling gates," *Journal of Physics A: Mathematical and Theoretical*, vol. 41, 9 2008. 55

[142] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge University Press, 2000. 56

[143] T. P. Harty, D. T. Allcock, C. J. Ballance, L. Guidoni, H. A. Janacek, N. M. Linke, D. N. Stacey, and D. M. Lucas, "High-fidelity preparation, gates, memory, and readout of a trapped-ion quantum bit," *Physical Review Letters*, vol. 113, no. 22, 2014. 58, 60, 63

[144] E. Torrontegui, S. Ibáñez, X. Chen, A. Ruschhaupt, D. Guéry-Odelin, and J. G. Muga, "Fast atomic transport without vibrational heating," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 83, no. 1, 2011. 58, 114

[145] A. Walther, F. Ziesel, T. Ruster, S. T. Dawkins, K. Ott, M. Hettrich, K. Singer, F. Schmidt-Kaler, and U. Poschinger, "Controlling fast transport of cold trapped ions," *Physical Review Letters*, vol. 109, no. 8, 2012. 58, 114

[146] R. Bowler, J. Gaebler, Y. Lin, T. R. Tan, D. Hanneke, J. D. Jost, J. P. Home, D. Leibfried, and D. J. Wineland, "Coherent diabatic ion transport and separation in a multizone trap array," *Physical Review Letters*, vol. 109, no. 8, 2012. 58, 114

[147] H. K. Lau and D. F. James, "Decoherence and dephasing errors caused by the dc Stark effect in rapid ion transport," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 83, no. 6, 2011. 58, 114

[148] T. Ruster, C. Warschburger, H. Kaufmann, C. T. Schmiegelow, A. Walther, M. Hettrich, A. Pfister, V. Kaushal, F. Schmidt-Kaler, and U. G. Poschinger, "Experimental realization of fast ion separation in segmented Paul traps," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 90, 5 2014. 58, 60

[149] A. Bermudez, X. Xu, R. Nigmatullin, J. O'Gorman, V. Negnevitsky, P. Schindler, T. Monz, U. G. Poschinger, C. Hempel, J. Home, F. Schmidt-Kaler, M. Biercuk, R. Blatt, S. Benjamin, and M. Müller, "Assessing the progress of trapped-ion processors towards fault-tolerant quantum computation," *Physical Review X*, vol. 7, 5 2017. 58, 60

[150] R. C. Sterling, H. Rattanasonti, S. Weidt, K. Lake, P. Srinivasan, S. C. Webster, M. Kraft, and W. K. Hensinger, "Fabrication and operation of a two-dimensional ion-trap lattice on a high-voltage microchip," *Nature Communications*, vol. 5, pp. 1–6, 4 2014. 59

[151] T. Sriarunothai, G. S. Giri, S. Wölk, and C. Wunderlich, "Radio frequency sideband cooling and sympathetic cooling of trapped ions in a static magnetic field gradient," *Journal of Modern Optics*, vol. 65, pp. 560–567, 3 2018. 61

[152] D. Cheung, D. Maslov, and S. Severini, "Translation Techniques Between Quantum Circuit Architectures," *Workshop on Quantum Information Processing*, pp. 1–3, 2007. 61

[153] Z. D. Romaszko, S. Hong, M. Siegele, R. K. Puddy, F. R. Lebrun-Gallagher, S. Weidt, and W. K. Hensinger, "Engineering of microfabricated ion traps and integration of advanced on-chip features," *Nature Reviews Physics*, vol. 2, no. 6, pp. 285–299, 2020. 62

[154] E. T. Campbell, B. M. Terhal, and C. Vuillot, "Roads towards fault-tolerant universal quantum computation," *Nature*, vol. 549, pp. 172–179, 9 2017. 72, 104

[155] M. Vasmer and D. E. Browne, “Three-dimensional surface codes: Transversal gates and fault-tolerant architectures,” *Physical Review A*, vol. 100, no. 1, 2019. 72, 104

[156] S. Brierley, “Efficient implementation of Quantum circuits with limited qubit interactions,” *ArXiv: 1507.04263*, 7 2015. 72

[157] S. Herbert, “On the depth overhead incurred when running quantum algorithms on near-term quantum computers with limited qubit connectivity,” *Quantum Information and Computation*, vol. 20, pp. 787–806, 5 2020. 72, 73

[158] S. McArdle, S. Endo, A. Aspuru-Guzik, S. C. Benjamin, and X. Yuan, “Quantum computational chemistry,” *Reviews of Modern Physics*, vol. 92, no. 1, 2020. 78

[159] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, “Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets,” *Nature*, vol. 549, no. 7671, pp. 242–246, 2017. 78

[160] G. AI Quantum, “Hartree-Fock on a superconducting qubit quantum computer,” *Science*, vol. 369, pp. 1084–1089, 4 2020. 78

[161] P. J. O’Malley, R. Babbush, I. D. Kivlichan, J. Romero, J. R. McClean, R. Barends, J. Kelly, P. Roushan, A. Tranter, N. Ding, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, A. G. Fowler, E. Jeffrey, E. Lucero, A. Megrant, J. Y. Mutus, M. Neeley, C. Neill, C. Quintana, D. Sank, A. Vainsencher, J. Wenner, T. C. White, P. V. Coveney, P. J. Love, H. Neven, A. Aspuru-Guzik, and J. M. Martinis, “Scalable quantum simulation of molecular energies,” *Physical Review X*, vol. 6, no. 3, 2016. 78

[162] J. I. Colless, V. V. Ramasesh, D. Dahlen, M. S. Blok, M. E. Kimchi-Schwartz, J. R. McClean, J. Carter, W. A. De Jong, and I. Siddiqi, “Computation of Molecular Spectra on a Quantum Processor with an Error-Resilient Algorithm,” *Physical Review X*, vol. 8, no. 1, 2018. 78

[163] I. G. Ryabinkin, T. C. Yen, S. N. Genin, and A. F. Izmaylov, “Qubit Coupled Cluster Method: A Systematic Approach to Quantum Chemistry on a Quantum Computer,” *Journal of Chemical Theory and Computation*, vol. 14, pp. 6317–6326, 9 2018. 78

[164] V. Armaos, D. A. Badounas, and P. Deligiannis, “Computational chemistry on quantum computers: Ground state estimation,” *arXiv: 1907.00362*, 2019. 78

[165] V. E. Elfving, M. Millaruelo, J. A. Gámez, and C. Gogolin, "Simulating quantum chemistry in the seniority-zero space on qubit-based quantum computers," *Physical Review A*, vol. 103, 1 2021. 78

[166] Y. Cao, J. Romero, J. P. Olson, M. Degroote, P. D. Johnson, M. Kieferová, I. D. Kivlichan, T. Menke, B. Peropadre, N. P. Sawaya, S. Sim, L. Veis, and A. Aspuru-Guzik, "Quantum Chemistry in the Age of Quantum Computing," *Chemical Reviews*, vol. 119, pp. 10856–10915, 10 2019. 78, 79, 81

[167] P. J. Ollitrault, A. Kandala, C.-F. Chen, P. K. Barkoutsos, A. Mezzacapo, M. Pistoia, S. Sheldon, S. Woerner, J. M. Gambetta, and I. Tavernelli, "Quantum equation of motion for computing molecular excitation energies on a noisy quantum processor," *Physical Review Research*, vol. 2, 10 2020. 78

[168] O. Higgott, D. Wang, and S. Brierley, "Variational quantum computation of excited states," *Quantum*, vol. 3, p. 156, 7 2019. 78

[169] V. Verteletskyi, T. C. Yen, and A. F. Izmaylov, "Measurement optimization in the variational quantum eigensolver using a minimum clique cover," *Journal of Chemical Physics*, vol. 152, 7 2020. 79

[170] W. J. Huggins, J. R. McClean, N. C. Rubin, Z. Jiang, N. Wiebe, K. B. Whaley, and R. Babbush, "Efficient and noise resilient measurements for quantum chemistry on near-term quantum computers," *npj Quantum Information*, vol. 7, 7 2021. 79

[171] S. T. Epstein, *The Variation Method in Quantum Chemistry*. Elsevier Science, 1974. 79

[172] S. Lloyd, "Universal quantum simulators," *Science*, vol. 273, pp. 1073–1078, 8 1996. 81

[173] D. Aharonov and A. Ta-Shma, "Adiabatic quantum state generation and statistical zero knowledge," *Conference Proceedings of the Annual ACM Symposium on Theory of Computing*, pp. 20–29, 1 2003. 81

[174] D. W. Berry, G. Ahokas, R. Cleve, and B. C. Sanders, "Efficient quantum algorithms for simulating sparse hamiltonians," *Communications in Mathematical Physics*, vol. 270, pp. 359–371, 8 2007. 81

[175] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari, and R. D. Somma, "Exponential improvement in precision for simulating sparse Hamiltonians," *Proceedings of the Annual ACM Symposium on Theory of Computing*, pp. 283–292, 12 2014. 81

[176] S. Descombes and M. Thalhammer, "An exact local error representation of exponential operator splitting methods for evolutionary problems and applications to linear Schrödinger equations in the semi-classical regime," *BIT Numerical Mathematics*, vol. 50, pp. 729–749, 12 2010. 81

[177] I. D. Kivlichan, C. Gidney, D. W. Berry, N. Wiebe, J. McClean, W. Sun, Z. Jiang, N. Rubin, A. Fowler, A. Aspuru-Guzik, H. Neven, and R. Babbush, "Improved fault-tolerant quantum simulation of condensed-phase correlated electrons via trotterization," *Quantum*, vol. 4, 2 2020. 81

[178] J. Huyghebaert and H. De Raedt, "Product formula methods for time-dependent Schrodinger problems," *Journal of Physics A: General Physics*, vol. 23, no. 24, pp. 5777–5793, 1990. 81

[179] A. M. Childs, Y. Su, M. C. Tran, N. Wiebe, and S. Zhu, "Theory of Trotter Error with Commutator Scaling," *Physical Review X*, vol. 11, no. 1, 2021. 81

[180] M. Suzuki, "General theory of fractal path integrals with applications to many-body theories and statistical physics," *Journal of Mathematical Physics*, vol. 32, pp. 400–407, 2 1991. 81

[181] "Simulating Hamiltonian Dynamics - Microsoft Quantum — Microsoft Docs - www.docs.microsoft.com/en-us/quantum/user-guide/libraries/chemistry/concepts/algorithms." 81

[182] G. H. Low and I. L. Chuang, "Hamiltonian simulation by qubitization," *Quantum*, vol. 3, 10 2019. 82, 91, 106

[183] D. W. Berry, C. Gidney, M. Motta, J. R. McClean, and R. Babbush, "Qubitization of arbitrary basis quantum chemistry leveraging sparsity and low rank factorization," *Quantum*, vol. 3, 2019. 82, 91, 106

[184] Microsoft Quantum, "Simulating nature with the new Microsoft Quantum Development Kit chemistry library - www.cloudblogs.microsoft.com/quantum/2018/12/04/simulating-nature-with-the-new-microsoft-quantum-development-kit-chemistry-library/," 2018. 82

[185] "NWChem - High performance computational chemistry software - www.nwchem-sw.org/index-php/Compiling_NWChem.html." 82

[186] R. Babbush, C. Gidney, D. W. Berry, N. Wiebe, J. McClean, A. Paler, A. Fowler, and H. Neven, "Encoding Electronic Spectra in Quantum Circuits with Linear T Complexity," *Physical Review X*, vol. 8, no. 4, 2018. 83

[187] P. Selinger, "Efficient Clifford+T approximation of single-qubit operators," *Quantum Information and Computation*, vol. 15, pp. 159–180, 12 2014. 84, 90

[188] M. Reiher, N. Wiebe, K. M. Svore, D. Wecker, and M. Troyer, "Elucidating reaction mechanisms on quantum computers," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 114, pp. 7555–7560, 7 2017. 86, 89, 90, 107, 121

[189] K. M. Svore, M. B. Hastings, and M. Freedman, "Faster phase estimation," *Quantum Information and Computation*, vol. 14, no. 3-4, pp. 306–328, 2014. 87

[190] N. Wiebe and C. Granade, "Efficient Bayesian Phase Estimation," *Physical Review Letters*, vol. 117, 8 2016. 87

[191] S. Kimmel, G. H. Low, and T. J. Yoder, "Robust calibration of a universal single-qubit gate set via robust phase estimation," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 92, 12 2015. 87

[192] Q. Sun, T. C. Berkelbach, N. S. Blunt, G. H. Booth, S. Guo, Z. Li, J. Liu, J. D. McClain, E. R. Sayfutyarova, S. Sharma, S. Wouters, and G. K. L. Chan, "PySCF: the Python-based simulations of chemistry framework," *Wiley Interdisciplinary Reviews: Computational Molecular Science*, vol. 8, 1 2018. 88

[193] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 86, 8 2012. 89, 92, 105

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

D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandrà, J. R. McClean, M. McEwen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. C. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. C. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. D. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven, and J. M. Martinis, "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, pp. 505–510, 10 2019. 103

[195] A. Deshpande, A. Mehta, T. Vincent, N. Quesada, M. Hinsche, M. Ioannou, L. Madsen, J. Lavoie, H. Qi, J. Eisert, D. Hangleiter, B. Fefferman, and I. Dhand, "Quantum Computational Supremacy via High-Dimensional Gaussian Boson Sampling," 2 2021. 103

[196] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 86, no. 3, p. 32324, 2012. 104

[197] S. B. Bravyi and A. Y. Kitaev, "Quantum codes on a lattice with boundary," 11 1998. 104

[198] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, "Topological quantum memory," *Journal of Mathematical Physics*, vol. 43, p. 4452, 8 2002. 104

[199] N. de Beaudrap and S. Herbert, "Quantum linear network coding for entanglement distribution in restricted architectures," *Quantum*, vol. 4, pp. 1–38, 10 2019. 104

[200] C. Monroe, R. Raussendorf, A. Ruthven, K. R. Brown, P. Maunz, L. M. Duan, and J. Kim, "Large-scale modular quantum-computer architecture with atomic memory and photonic interconnects," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 89, 8 2014. 105

[201] T. R. Scruby, D. E. Browne, P. Webster, and M. Vasmer, "Numerical Implementation of Just-In-Time Decoding in Novel Lattice Slices Through the Three-Dimensional Surface Code," *arXiv: 2012.08536*, 2020. 105, 115

[202] Y. Li and S. C. Benjamin, "Hierarchical surface code for network quantum computing with modules of arbitrary size," *PHYSICAL REVIEW A*, vol. 94, p. 42303, 2016. 105

[203] J. O'gorman, N. H. Nickerson, P. Ross, J. J. Morton, and S. C. Benjamin, "A silicon-based surface code quantum computer," *npj Quantum Information*, vol. 2, pp. 1–14, 2 2016. 105

[204] J. Eli Bourassa, R. N. Alexander, M. Vasmer, A. Patil, I. Tzitrin, T. Matsuura, D. Su, B. Q. Baragiola, S. Guha, G. Dauphinais, K. K. Sabapathy, N. C. Menicucci, and I. Dhand, "Blueprint for a scalable photonic fault-tolerant quantum computer," tech. rep., 2021. 105

[205] R. Raussendorf, S. Bravyi, and J. Harrington, "Long-range quantum entanglement in noisy cluster states," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 71, 7 2004. 105

[206] R. Raussendorf, J. Harrington, and K. Goyal, "A fault-tolerant one-way quantum computer," *Annals of Physics*, vol. 321, pp. 2242–2270, 9 2006. 105

[207] C. Gidney and A. G. Fowler, "Flexible layout of surface code computations using AutoCCZ states," *arXiv: 1905.08916*, 5 2019. 105, 117, 119, 120, 127

[208] G. H. Low and I. L. Chuang, "Optimal Hamiltonian Simulation by Quantum Signal Processing," *Physical Review Letters*, vol. 118, 6 2016. 106

[209] D. Poulin, A. Kitaev, D. S. Steiger, M. B. Hastings, and M. Troyer, "Quantum Algorithm for Spectral Measurement with a Lower Gate Count," *Physical Review Letters*, vol. 121, no. 1, 2018. 106

[210] Z. Li, J. Li, N. S. Dattani, C. J. Umrigar, and G. K.-L. Chan, "The electronic complexity of the ground-state of the FeMo cofactor of nitrogenase as relevant to quantum simulations," *Journal of Chemical Physics*, vol. 150, 9 2018. 107, 121

[211] V. von Burg, G. H. Low, T. Häner, D. S. Steiger, M. Reiher, M. Roetteler, and M. Troyer, "Quantum computing enhanced computational catalysis," *Physical Review Research*, vol. 3, no. 3, 2021. 107, 121

[212] J. Lee, D. W. Berry, C. Gidney, W. J. Huggins, J. R. McClean, N. Wiebe, and R. Babbush, "Even More Efficient Quantum Computations of Chemistry Through Tensor Hypercontraction," *PRX Quantum*, vol. 2, no. 3, 2021. 107, 118, 120, 121, 122, 128

[213] D. Aggarwal, G. Brennen, T. Lee, M. Santha, and M. Tomamichel, "Quantum Attacks on Bitcoin, and How to Protect Against Them," tech. rep., 2018. 107

[214] L. Tessler and T. Byrnes, "Bitcoin and quantum computing," *arXiv:1711.04235*, 2017. 107

[215] T. Häner, S. Jaques, M. Naehrig, M. Roetteler, and M. Soeken, "Improved Quantum Circuits for Elliptic Curve Discrete Logarithms," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12100 LNCS, pp. 425–444, 2020. 107, 119, 124, 125, 126, 127

[216] M. Roetteler, M. Naehrig, K. M. Svore, and K. Lauter, "Quantum resource estimates for computing elliptic curve discrete logarithms," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10625 LNCS, pp. 241–270, 2017. 107, 119, 125

[217] C. Jones, "Low-overhead constructions for the fault-tolerant Toffoli gate," *Physical Review A - Atomic, Molecular, and Optical Physics*, vol. 87, p. 022328, 2 2013. 109, 122

[218] Y. R. Sanders, J. J. Wallman, and B. C. Sanders, "Bounding quantum gate error rate based on reported average fidelity," *New Journal of Physics*, vol. 18, no. 1, 2016. 110

[219] J. Wallman, C. Granade, R. Harper, and S. T. Flammia, "Estimating the coherence of noise," *New Journal of Physics*, vol. 17, no. 11, 2015. 110

[220] S. T. Flammia and J. J. Wallman, "Efficient Estimation of Pauli Channels," *ACM Transactions on Quantum Computing*, vol. 1, no. 1, pp. 1–32, 2020. 110

[221] R. Kueng, D. M. Long, A. C. Doherty, and S. T. Flammia, "Comparing Experiments to the Fault-Tolerance Threshold," *Physical Review Letters*, vol. 117, no. 17, 2016. 110

[222] A. G. Fowler, "Time-optimal quantum computation," *arXiv: 1210.4626*, 2012. 111, 121

[223] M. Gutiérrez, M. Müller, and A. Bermúdez, "Transversality and lattice surgery: Exploring realistic routes toward coupled logical qubits with trapped-ion quantum processors," *Physical Review A*, vol. 99, no. 2, 2019. 114

[224] M. Amy, D. Maslov, and M. Mosca, "Polynomial-time T-depth optimization of Clifford+T circuits via matroid partitioning," *IEEE Transactions on Computer-Aided*

Design of Integrated Circuits and Systems, vol. 33, no. 10, pp. 1476–1489, 2014. 116, 127, 132

[225] N. Abdessaied, M. Amy, M. Soeken, and R. Drechsler, "Technology mapping of reversible circuits to Clifford+T quantum circuits," 2016. 116, 127, 132

[226] G. M. Abdullah, Q. Mehmood, and C. B. A. Khan, "Adoption of Lamport signature scheme to implement digital signatures in IoT," *2018 International Conference on Computing, Mathematics and Engineering Technologies: Invent, Innovate and Integrate for Socioeconomic Development, iCoMET 2018 - Proceedings*, vol. 2018-January, pp. 1–4, 4 2018. 126