Imperial College London

Imperial College London
Electrical and Electronic Engineering

A suite of quantum algorithms for the shortest vector problem

Author:
David Joseph

Supervisors:
Dr. Cong Ling,
Dr. Florian Mintert

Submitted in partial fulfilment of the requirements for the degree of
Doctor of Philosophy in Electrical and Electronic Engineering
Imperial College London, 10 August 2021

Copyright Declaration

Abstract

Crytography has come to be an essential part of the cybersecurity infrastructure that provides a safe environment for communications in an increasingly connected world. The advent of quantum computing poses a threat to the foundations of the current widely-used cryptographic model, due to the breaking of most of the cryptographic algorithms used to provide confidentiality, authenticity, and more. Consequently a new set of cryptographic protocols have been designed to be secure against quantum computers, and are collectively known as post-quantum cryptography (PQC). A forerunner among PQC is lattice-based cryptography, whose security relies upon the hardness of a number of closely related mathematical problems, one of which is known as the shortest vector problem (SVP).

In this thesis I describe a suite of quantum algorithms that utilize the energy minimization principle to attack the shortest vector problem. The algorithms outlined span the gate-model and continuous time quantum computing, and explore methods of parameter optimization via variational methods, which are thought to be effective on near-term quantum computers. The performance of the algorithms are analyzed numerically, analytically, and on quantum hardware where possible. I explain how the results obtained in the pursuit of solving SVP apply more broadly to quantum algorithms seeking to solve general real-world problems; minimize the effect of noise on imperfect hardware; and improve efficiency of parameter optimization.

Acknowledgements

I am grateful for the many opportunities this PhD course has afforded me. I would like to extend my sincere thanks to Cong and Florian for guiding me in their respective areas. The support of Cong's group, and his inspiration to work hard and fast have undoubtedly helped me throughout my time at Imperial, and Florian's patience in discussing quantum mechanical concepts on the whiteboard at length was keenly appreciated.

Thank you to Adam Callison for being a constant point of reference and with whom writing my second paper was a pleasure. To Chris Porter and Charlie Grover, the shared experiences and humour have made the whole process go a lot quicker. Michael Lawson's perpetual tea breaks were a constant hindrance and important source of amusement. To Ahmad, Pepe, Bruno and Ilya (and Neil!), thank you for great craic and your shared interest in all things ML and finance. To everyone at Sandbox it was a fantastic experience learning from you all about quantum machine learning, and I benefited enormously from Antonio Martinez's expertise while we coauthored my third paper.

On a personal note, thank you to my parents for their patient support (and two years of lodging), and to my wonderful girlfriend Eve for being around whenever I needed to vent. Thanks to Simon and Peter for helping to make our diverse and dangerous corner of London home.

I am indebted to the Academic Centres of Excellence in Cyber Security Research for their funding of the project, and to Andrew E. David M. and Dan S. for their friendly guidance.

Statement of Originality and Publications

The work contained in this thesis is my own, apart from in places which this is explicitly stated and appropriately referenced. The original work of this thesis has not been used to qualify for another academic award. Chapter 2 contains a literature review, Chapters 3, 5, 6 are based upon theory, simulations and experiments presented in [JCLM21, JMLM21] which are subject to Creative Commons 4.0 International license and the arXiv.org perpetual non-exclusive license respectively. These chapters use some text and figures from these papers. Chapter 4 is based on theory and results presented in [JGLM20] under Creative Commons 4.0 International license, and reuses some text and figures from this paper. All published manuscripts are also hosted on arXiv under the arXiv.org perpetual, non-exclusive license.

Contents

List of Figures

Acronyms

API Application programme interface

AQC Adiabatic quantum computing

BDD Bounded distance decoding

BKZ Block Korkine Zolotarev

CVP Closest Vector Problem

DGS Discrete Gaussian Sampler

ETSI European Telecommunications Standards Institute

FHE Fully homomorphic encryption

GGH Goldreich, Goldwasser and Halevi

HKZ Hermite Korkine Zolotarev

HNF Hermite normal form

ISO International Standards Organization

LBC Lattice-based cryptography

LLL Lenstra Lenstra Lovasz

LWE Learning with errors

MLWE Module learning with errors

NISQ Noisy intermediate-scale quantum

NIST National Institute of Standards of Technology

NTRU N^{th} degree truncated polynomial rings

PQC Post-quantum cryptography

QAOA Quantum Approximate Optimization Algorithm

QMA Quantum Merlin Arthur

QML Quantum machine learning

QMV Quantum mean value

QUBO Quadratic unconstrained binary optimization

RLWE Ring learning with errors

RSA Rivest, Shamir, and Adelman cryptosystem

SIS Short integer solution

SIVP Shortest independent vectors problem

SNDL Store Now Decrypt Later

SVP Shortest Vector Problem

VQA Variational Quantum Algorithm

Chapter 1

Introduction

1.1 Cryptography

Cryptography is the name for the set of techniques used to communicate securely in the presence of adversaries. While methods to thwart adversaries have been devised over millenia, it is over the past century that the information revolution and computer technology have allowed cryptography to flourish. In particular it is the development of protocols that depend on the hardness of a range of mathematical problems that have enabled the plethora of uses cryptography now serves, such as: confidentiality, authenticity, non-repudiation, and homomorphic encryption.

The earliest form of modern cryptography was symmetric-key cryptography, and this took off around the end of the first world war with the invention of the Enigma machine in 1918 [Ell05], and the formation of the Government Code and Cypher School in Britain in 1919 [Hea91]. In these early years of modern cryptography, the techniques centred around confidentiality, that is, allowing two parties (Alice and Bob) to communicate plaintexts securely between one another over an insecure channel, which an eavesdroper (Eve) may be listening in to. This is known as symmetric-key (or secret-key) cryptography [PP09] because it is necessary that Alice and Bob both share some secret $\mathbf{s}$ which is used to both encrypt the plaintext, and also decrypt the ensuing ciphertext to recover the plaintext. Any adversary who learns $\mathbf{s}$ could also then read

encrypted messages. Until 1976 this remained the only publicly known form of encryption.

1.2 Public-key cryptography

The mid-Seventies was a fruitful time for cryptology, as a revolutionary new method for exchanging secrets securely over insecure channels meant that Alice and Bob could set up their secure channel (i.e. establish a common secret $\mathbf{s}$) without having to physically exchange it [DH76]. Soon after followed a technique called RSA (after the authors Rivest, Shamir and Adelman) [RSA78] which allowed arbitrary messages to be encrypted and communicated from Alice to Bob without the requirement of a pre-shared secret key $\mathbf{s}$. These protocols became known as public-key cryptography, as to implement them, a party generates a public key which is known to the world, and a private key which is kept to themselves. These key pairs generally are intimately linked according to some mathematical structure, but in order to derive the private key from the public key one must solve some intractably hard mathematical problem, such as finding discrete logarithms in the case of Diffie-Hellman key exchange [DH76], or the closely related integer factorization problem in the case of RSA cryptography [RSA78]. Public-key cryptography is more expensive computationally than symmetric cryptography, as sending a given message requires more computation and larger ciphertexts and public keys to be communicated. Public-key cryptography thus came to be used for very specific but powerful usecases which complement symmetric cryptography, such as for exchanging keys prior to initiating a symmetric cryptography session, or for authenticating messages.

Public-key cryptography is now pervasive across the many different forms of media used for modern communication. It lies at the heart of the transport layer security (TLS) protocol [DA99] as well as internet key exchange (IKE) [HC98]. It is used throughout all layers of the software stack, in low level protocols such as media access control security (MACsec) [Com99] and internet protocol security (IPsec) [KA98]. Standards bodies at this moment are incorporating public key cryptography into 5G networks and it is expected to be mandatory across 6G networks. The internet of things (IoT) [MLLL15] and edge computing [Ham19] herald an

age of increasingly distributed computation, but this in turn implies more attack points for adversaries and so secure and efficient cryptographic protocols are necessary for these devices, which are often characterized by constraints on computation power due to their size. The vast array of public-key cryptography applications make its reliability and security more important than ever, and particularly in the face of the threat posed by quantum computers.

1.3 Quantum computing

The concept of quantum computation is much more recent than that of cryptography, and was first proposed as a possibility in the Eighties [Ben80, Fey99]. Initially it was used to demonstrate toy examples of how information could be manipulated in ways not possible with classical machines, such as the Deutsch-Jozsa algorithm [DJ92] and quantum teleportation [BBC+93]. Meanwhile other uses for quantum devices (other than computation) were being conjured, such as quantum key distribution [BB20] which can perform key exchange, but instead of using the conjectured hardness of mathematical problems to guarantee security, it uses fundamental laws of physics.

The first algorithm which demonstrated an exponential speed-up in the computation of a materially interesting problem was developed by mathematician Peter Shor in 1994 [Sho99], whereby the factorization of integers was shown to be possible in polynomially many quantum operations. This was an important step as it was the first demonstration that quantum computers could have a significant impact on the real world, if only large and fault-tolerant quantum devices could be constructed to implement the algorithms. The key implication of Shor's algorithm was that the public-key cryptosystems that were then around two decades old would become obsolete in the face of a quantum adversary.

While Shor's algorithm factored integers, the mathematical structure of many public-key cryptosystems of the time were very similar. Consequently, due to the period finding nature of Shor's algorithm, it meant that cryptosystems based on the hardness of finding discrete logarithms (i.e. Diffie-Hellman key exchange [DH76]) and performing certain operations on elliptic

curves (i.e. elliptic curve cryptography (ECC) [HMV06]) would also be broken. The net result was that the majority of public key cryptosystems in use at the time would eventually have to be replaced by ones which relied upon totally new flavours of mathematical problems.

In the meantime, the study of how to manipulate extremely fragile quantum systems to perform computations blossomed into a wide and varied field characterized by engineering limitations. In the early years research centred around gate model algorithms which are closely analogous to the classical computational architecture, but eventually new approaches were discovered such as discrete time quantum walks [AAKV01], adiabatic quantum computing [FGGS00], quantum annealing [KN98], continuous time quantum walks [FG98], and others. Some of these approaches offered advantages such as more scalable systems at the cost of less control over individual components, such as with annealing-style algorithms.

As well as the computational architecture, the very building blocks of such systems have been investigated from a variety of angles. To create a qubit, which upon measurement achieves one of two basis states $|0\rangle, |1\rangle$ and before then remains in a linear superposition of those states $\alpha|0\rangle + \beta|1\rangle$, one simply requires a two-level quantum system. Consequently qubits composed of photons [KLM01], electrons [BDEK04], quantum dots [LD98], optical lattices [DBJ00], and more have all been considered. So far there is no clear winner, and there are many competing firms with quantum computers with astonishingly varied constructions.

The extraordinary diversity amongst quantum computing devices makes comparison extremely difficult, especially as there are many axes along which progress must be made to obtain useful machines. The aspects to be improved include single and two-qubit gate fidelities, coherence times, error-correcting codes, achieving high levels of connectivity, and high depth circuits. Some of the above are closely related, and some represent independent challenges. At present the largest gate model computers are of the order of 100 qubits but are not fault tolerant: Google made the first serious claim to quantum supremacy with their 53 qubit machine in 2019 [AAB+19]. Competitors such as IBM and Amazon have laid out roadmaps over the next decade [Gam20, CNAA+20, Cha20] which outline ambitious pathways to larger and larger quantum computers, with IBM's $1,121$ qubit chip called Condor due in 2023.

A key milestone for the industry will be developing the first logical qubit (a system behaving like a single perfect qubit), which is expected to require around 1,000 error-corrected qubits. It has been estimated that around 4,000 such logical qubits would be required to break RSA-2048 [RNSL17], (i.e. factoring 2048 bit semi-primes), though with the current rate of progress, this may be reached within a decade [PM20], according to a survey of leading experts in quantum computing.

1.4 Post-quantum cryptography

In order to preserve information security, it is necessary to find new mathematical problems upon which to base cryptosystems, and through concentrated cryptanalysis, convince ourselves that such schemes are resistant to quantum attacks. Cryptography that is thought to be resistant to quantum attacks is collectively known as post-quantum cryptography (PQC). PQC encompasses both old cryptography which is believed to be resistant to quantum attacks, as well as newer cryptography designed with mitigation of quantum attacks in mind.

Two of the oldest families of PQC are hash-based and code-based cryptography. Cryptographic hash functions date back to the Seventies, partly in response to the need for one way hash functions identified in Diffie and Hellman's seminal paper on public key cryptography [DH76]. Accordingly, the first constructions began to appear a few years later [Yuv79, Rab79, Mer79]. Hash functions remain an integral part of cryptography, both traditional and post-quantum. They are well suited to digital signature schemes, and while not the most efficient of post-quantum signatures, they boast strong security guarantees due to the many decades of cryptanalysis they have undergone. Code-based cryptography is another family of cryptosystems [OS09], based on the hardness of decoding random codes. The oldest of this category, named after the author, is the McEliece cryptosystem of 1978 [McE78] which remains unbroken and is a finalist in the foremost standardization process (discussed in a later section). This scheme provides similar functionality to RSA (and was published around the same time) but is less efficient and so has not seen widespread adoption. Nevertheless, the age of this family of cryptography lends strong

security guarantees to systems based on the hardness of decoding random codes.

More recent branches of what is now considered post-quantum cryptography include lattice-based [Pei16], multivariate-based [DY09], and isogeny-based cryptography [DF17], all of which are reviewed in [BBD09] and [BL17]. The most well-balanced in terms of efficiency appears to be lattice-based constructions, largely due to the impressive efficiency gains from embedding lattices in algebraic number fields, at the cost of some confidence in the security of such systems.

1.4.1 Near-term threats

Despite the fact that quantum computers capable of factorizing 2048 bit semi-primes may still be over a decade away, there are strong motivations for upgrading cryptography in the near term. The first is known as the Store Now; Decryt Later (SNDL) attack. In this, an adversary (Eve) can exfiltrate all communications between Alice and Bob, which are bulk encrypted using symmetric cryptography, but where Alice and Bob's secret keys are exchanged using Diffie-Hellman key exchange or similar (RSA, ECC, etc). Eve can then store this encrypted data for a number of years until she has access to a large fault-tolerant quantum computer, and can then use this to break the key exchange, obtaining the secret key that Alice/Bob have used to bulk encrypt, at which point Eve can then trivially decrypt the entire communication session. This is a serious threat where valuable data being exchanged now is expected to still be sensitive by the time large quantum computers are available – in $5, 10$ or 15 years' time. This could include medical records, trade/military secrets, and government communications, and furthermore, this type of attack is practicable on today's data. The SNDL attack is a strong motivator to upgrade to post-quantum key exchange.

Another compelling reason to transition is that many applications utilize cryptographic hardware which is difficult to upgrade. This could include vehicles [LPW06] such as ships and planes which generally have multi-decade lifespans, which means that devices being fitted now into brand new vehicles would be expected to be operational well into the post-quantum era, and retrofitting cryptographic devices at a later stage in many cases would be significantly more troublesome than deploying quantum-safe hardware at launch. Similarly, industrial control

systems also rely on cryptographic hardware and with some systems requiring 99.999% availability (or six minutes downtime per year), upgrading hardware or even software represents an unacceptable cost.

For these reasons, and the natural complexity of transitioning from a widely used set of cryptographic algorithms to a completely new set, it is important that work progresses rapidly towards enabling this migration. The first step in this process is to standardize a set of safe algorithms, which is a considerable undertaking in itself.

1.4.2 Standardization

With the threat of the quantum computing age looming and with a number of cryptography families reaching maturity, organizations worldwide have taken to standardizing a set of quantum-safe public-key cryptosystems for the quantum age. Already the National Institute for Standards and Technology (NIST) has standardized two stateful hash-based signature schemes [HBG+18, MCF19] in a process completed in 2020 [CAD+20], and in Europe, the European Telecommunications Standards Institute (ETSI) is leading efforts towards standardization of quantum key distribution [LL09].

NIST, at the time of writing, is also in the final round of a process to standardize between two and four cryptosystems (key encapsulation mechanisms and digital signature schemes). This process is generally regarded as the most inclusive and thorough standardization process for PQC, having commenced in 2017, considering 82 initial submissions, of which 69 were considered in the first round, 26 in the second round [AAAS+19], and 15 in the third round [AASA+20]. Of the 15 in the third round, there are seven finalist schemes and eight alternate track schemes. The reasoning is that there should be a key encapsulation mechanism and a digital signature algorithm standardized reasonably quickly that can be an all-rounder performance wise, (picked from the finalists), and the alternate track allows for a more diverse set of schemes to continue to be analysed, as well as allowing for more use case-specific schemes to be considered, such as isogeny-based cryptosystems which suffer large computational workloads,

but at the gain of very small ciphertext and public key sizes, which could be helpful in scenarios where communication bandwidth is the primary bottleneck.

Of the finalist schemes, there are four key encapsulation mechanisms (of which three are lattice-based) and three digital signature algorithms (of which two are lattice-based), putting lattice-based cryptography (LBC) firmly in the lead due to the all-round efficiency of its more structured variants which are embedded in number fields. It is imperative that all schemes considered for standardization face intense scrutiny, but with lattice-based cryptosystems looking like strong favourites for the post-quantum era, the need for cryptanalysis of this family of cryptography is even stronger. The content of this thesis analyzes how fully quantum devices might attack a central mathematical problem – the shortest vector problem – that underpins lattice-based cryptography. While previous works incorporate quantum theory into attacks by adapting classical algorithms, the subject of this thesis can be interpreted as fully quantum approaches, unencumbered by the traditional algorithmic approaches of previous cryptanalysis.

1.5 Contribution

In Chapter 2, I review the relevant literature for both lattice-based cryptography and quantum computing which are both necessary to develop a holistic understanding of subsequent chapters. The chapter lends particular focus to the mathematical problems arising from lattices, as well as the best classical algorithms for solving them. For quantum computing I cover the main architectures and their advantages/drawbacks, as well as the variational algorithms which are likely to be central to near-term implementations.

In Chapter 3, I outline in classical terms the way the shortest vector problem can be mapped to a classical Hamiltonian, and provide a theoretical upper bound on the search space required to guarantee existence of the desired vector within the solution set. This is important for Chapters 5 and 6, and provides helpful context to Chapter 4.

In Chapter 4, I detail a quantum algorithm for the shortest vector problem using ultra-cooled gases trapped in an optical lattice for which long range interactions can be arbitrarily tuned by

a practitioner. I subsequently provide a numerical analysis of the algorithm for low dimensional lattice instances.

In Chapter 5, I build upon the foundations of Chapter 3 to define a quantum Ising formulation of the shortest vector problem, and provide an analysis of two variants of the algorithm simulated numerically, and as implemented on a D-Wave 2000Q quantum annealer. These results are then used to demonstrate how different choices of qudits can affect robustness to noise in near term devices.

In Chapter 6, I provide a circuit-based quantum algorithm for the shortest vector problem, utilizing Hamiltonians defined in the work of Chapter 5. The algorithm of this chapter is low depth and admits more thorough mathematical analysis than previous algorithms, and this analysis is used to derive approximations that enable more efficient optimization of quantum algorithms not just for the problem at hand, but for a wide range of algorithms including integer optimization problems, which have strong relevance to real-world scenarios.

in Chapter 7, I summarise the findings and achievements described in the thesis and provide an outlook for future work, including general problems to be overcome by the cryptography and quantum computing communities at large. I then suggest natural next steps and investigations to supplement and improve the findings of my research.

Chapter 2

Background Theory

The contents of this thesis covers a wide range of technical topics, spanning many variants of quantum computing (both gate-based and continuous time models), the principles of public key cryptography, and the field of mathematical lattices upon which one family of cryptosystems – conjectured to be quantum-secure – are based. In this section I define the foundational concepts that will be necessary for full understanding of the ensuing technical chapters.

2.1 Cryptography

The inner workings of a cryptosystem will not be essential to understanding the original contributions presented in later chapters, but an understanding of the mechanisms by which cryptosystems protect data and communications helps one put into context the importance of the underlying mathematical problems which provide the security, which in turn provides context for the valuable role quantum cryptanalysis plays in the protection of the next generation of information technological systems.

Symmetric cryptography, or private key cryptography has been around for a long time and behaves in a rather different way to public key cryptography, relying on many simple mathematical functions. Public key cryptography relies on hard to invert mathematical problems

which are highly structured and involve advanced mathematics such as number theory and group theory. Public key cryptography has a number of applications, the two most common of which are secure key exchange and authentication of communications. Two commonly used methods of key exchange are described below.

2.1.1 Public key cryptography

2.1.1.1 Diffie-Hellman key exchange

In the Diffie Hellman key exchange, which is of the form of Step 1 of Fig. 2.1, Alice and Bob agree on a number field which is usually the multiplicative group of integers modulo some prime p. They also agree on a generator of this group called g, which is a primitive root. Alice and Bob each generate their key pair (secret key sk and public key pk) by each picking a private key in the multiplicative group a, b respectively, and obtaining the public keys by raising g to the power of their private key

$$\begin{aligned} (sk, pk)_A &= (a, g^a), \\ (sk, pk)_B &= (b, g^b). \end{aligned} \tag{2.1}$$

Alice sends Bob her public key g^a and Bob sends Alice his public key g^b. They reach an agreed shared secret by raising their partner's public key to the power of their own private key

$$(g^a)^b = (g^b)^a = g^{ab}. \tag{2.2}$$

For an eavesdropper to deduce the key, they must find out one of either a or b. But this is equivalent to finding x given g^x modulo p, which is not efficiently computable on classical computers, and so for large enough p the key exchange is secure.

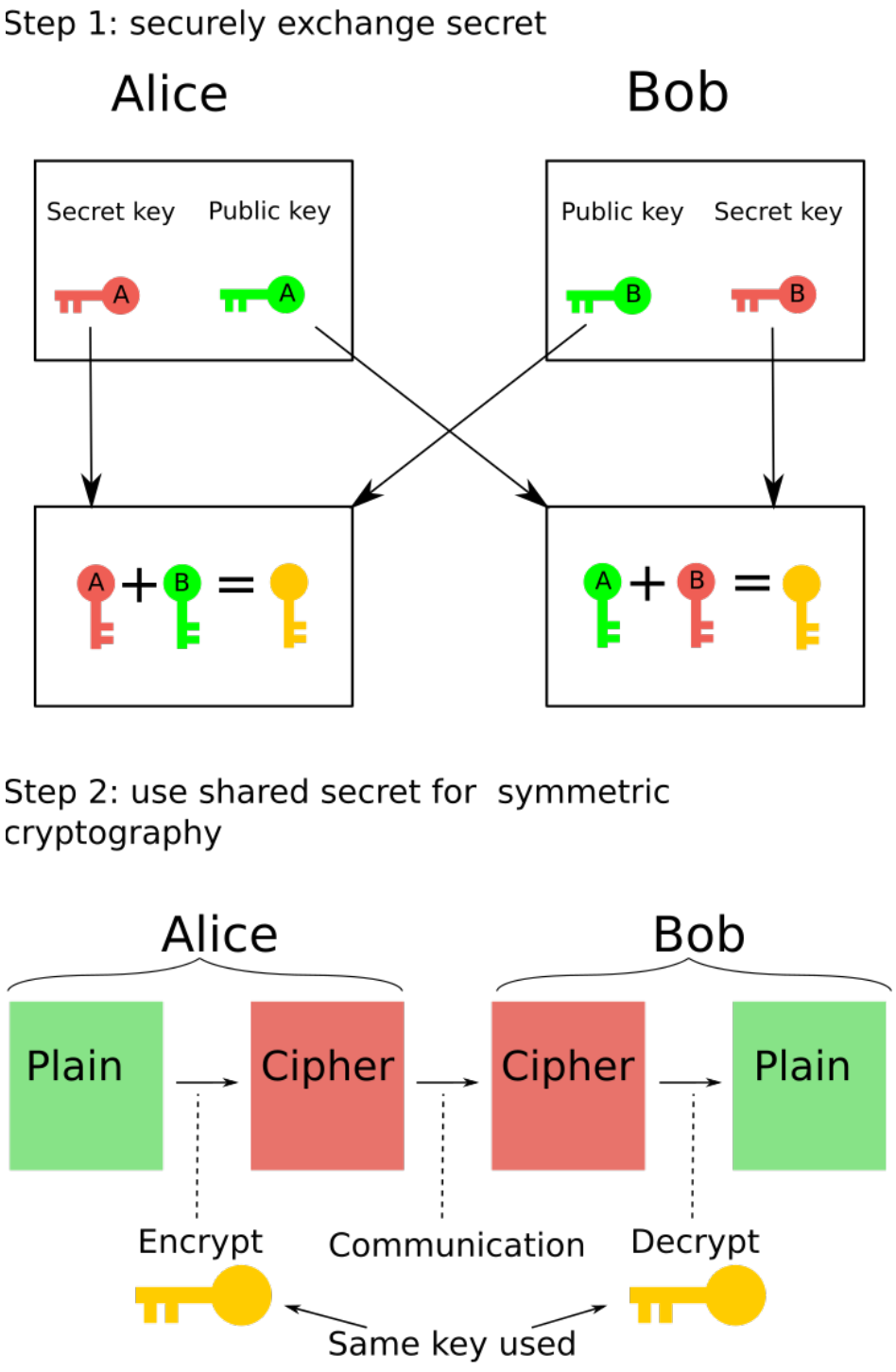

Figure 2.1: Alice and Bob must securely exchange a shared secret (golden key) over an insecure channel (Step 1), and then they can proceed to communicate confidentially by exchanging messages encrypted symmetrically using their shared secret (Step 2).

2.1.1.2 Rivest-Shamir-Adelman (RSA) cryptography

In RSA encryption, Alice generates a secret/public keypair (sk, pk) by first selecting two large primes p, q with $n = pq$, and picking an integer $0 < e < n$. The number n is public and the primes p, q must be kept secret. Then,

$$\begin{aligned} pk &= (n, e), \\ sk &= d = e^{-1} \text{ modulo } n. \end{aligned} \tag{2.3}$$

Bob encrypts message M by firstly hashing it into the number field being used $m = H(M)$, then raising it to the power of the public key

$$c = m^e \text{ modulo } n, \tag{2.4}$$

and Alice decrypts by raising c to the power of the secret key d

$$c^d = (m^e)^d = m \text{ modulo } n. \tag{2.5}$$

In the RSA algorithm, breaking the system means finding the e^{th} root of c, which is intractable for large n on classical computers. Equivalently, factorizing $n = pq$ into p, q also breaks the system as it reveals Euler's totient function $\phi(n) = (p-1)(q-1)$ which is the order of the group of units $\mathbb{Z}_N^\times$. When Alice chooses e, she must check that it is coprime with $\phi(n)$, then to generate the secret key she should find an integer to satisfy the relation $de \equiv 1$ modulo n, which is efficiently computable given $\phi(n)$. While this is an algorithm describing Bob's encryption of message M so that only Alice can decrypt it, it can be interpreted as a key exchange algorithm by considering M as a secret generated by Bob, and transmitted securely to Alice so that they can then both perform symmetric cryptography using M.

2.2 Lattices

Mathematical lattices possess many appealing properties from a cryptographic standpoint. Lattice algorithms use simple linear algebra operations on vectors, and most importantly, provides strong average-case security guarantees [Ajt96]. This is key, because for real life implementation, parameters must be drawn from some distribution, and so the problem must be hard on average, and not just in the worst case, which is usually considered in the theory of algorithms.

LBC boasts other attractive traits too, that have helped it to become a frontrunner to replace current cryptosystems. One can make efficient constructions which use the properties of algebraic fields to increase efficiency of LBC schemes significantly [LPR10, AD17], at the cost of an extra security assumption. It can also be adapted to serve a wide variety of cryptographic functionalities, and is the only flavour of cryptographic algorithms yet to provide Fully Homomorphic Encryption (FHE) [Gen09].

Definition 2.1 (Mathematical lattice)**.** *A mathematical lattice is a repeating pattern of points in N-dimensional space. It can be described as the set of integer combinations of the basis vectors of some basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_N\}$, so can be written*

$$\mathcal{L}(\mathbf{B}) = \sum_i x_i \mathbf{b}_i = \{\mathbf{xB} \colon \mathbf{x} \in \mathbb{Z}^N\}, \tag{2.6}$$

where $\mathbf{B}$ can be interpreted as a matrix whose rows are the basis vectors $\mathbf{b}_i$.

An important tool in the construction of algorithms in thesis is the Gram matrix.

Definition 2.2 (Gram matrix)**.** *The Gram matrix of a lattice $\mathcal{L}(\mathbf{B})$ is*

$$\mathbf{G} = \mathbf{B}\mathbf{B}^T, \quad \mathbf{G}_{ij} = \mathbf{b}_i \mathbf{b}_j^T, \tag{2.7}$$

where the elements of $\mathbf{G}$ are the overlap of basis vectors $\mathbf{b}_i, \mathbf{b}_j$.

A lattice of dimension N exists within an M-dimensional ambient space $\mathbb{R}^M$ where $N \leq M$, and is said to be full rank if $N = M$. For this thesis, only full rank lattices are considered and

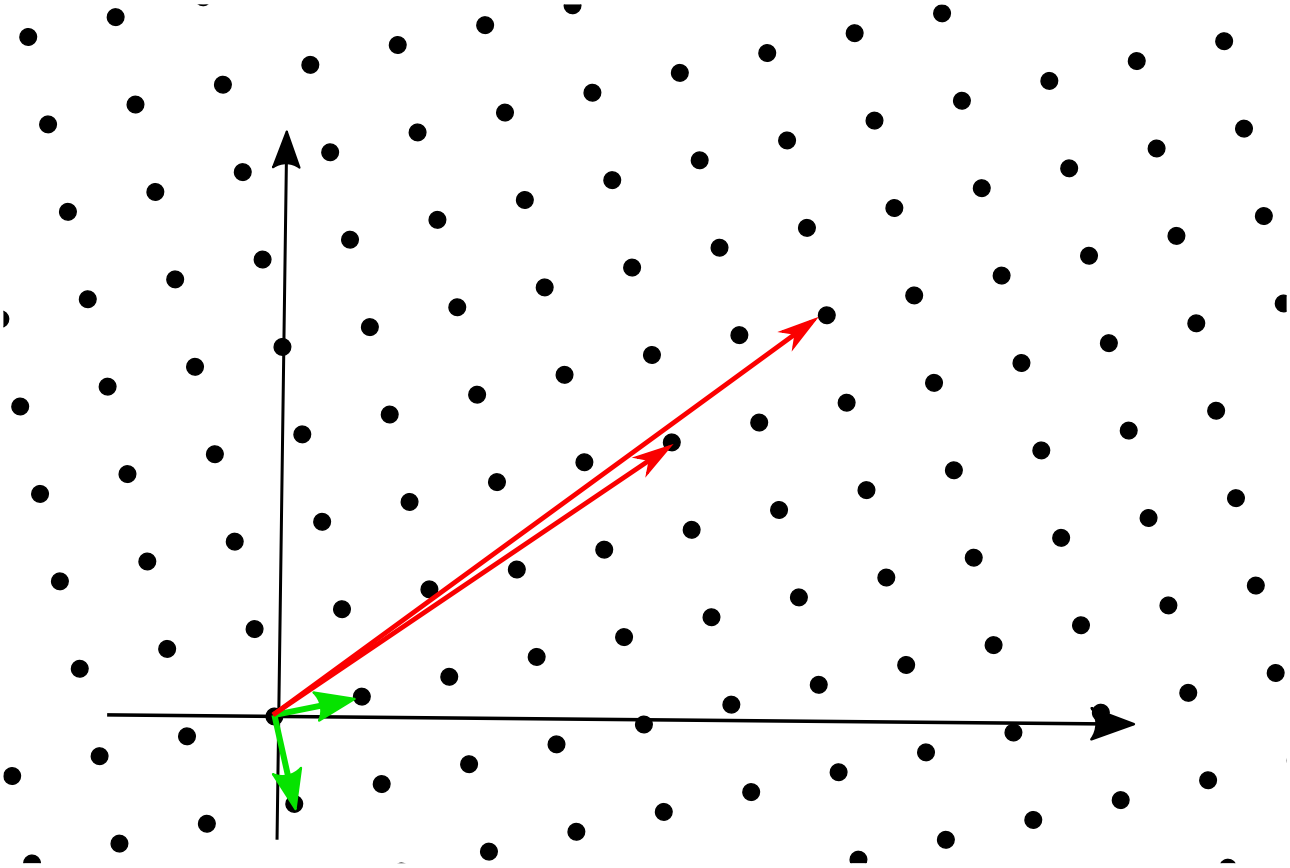

Figure 2.2: Points of a lattice, along with a good basis (short green arrows), and a worse basis (long red arrows). Every point in the lattice can be written as an integer combination of the red arrows, or as a different integer combination of the green arrows.

so in the following the dimension of the lattice N is also the dimension of the ambient space. A lattice can have infinitely many bases, but only a small number of good bases. Good bases are those which have vectors which are somewhat short and are highly orthogonal to one another. Each basis for a lattice can be transformed into any other basis of the lattice via a unimodular transformation, which means multiplying the basis matrix with a unimodular matrix $\mathbf{U}$ (an integer matrix with determinant ± 1)

$$\begin{aligned} \mathbf{B}_2 &= \mathbf{U}\mathbf{B}_1 \\ \mathcal{L}(\mathbf{B}_1) &= \mathcal{L}(\mathbf{B}_2). \end{aligned} \tag{2.8}$$

Lattices are geometric objects, and an important concept is the fundamental parallelepiped, which describes the 'shape' of a particular lattice basis.

Definition 2.3 (Fundamental parallelepiped)**.** *The fundamental parallelepiped of a lattice* $\mathcal{L}(\mathbf{B})$ *is*

$$\mathcal{P}(\mathcal{L}) = \{\mathbf{xB} : \mathbf{x} \in [0,1)^N\}, \tag{2.9}$$

and the covolume of $\mathcal{L}$ is the volume of $\mathcal{P}(\mathcal{L})$. While the geometry and covolume of a lattice is invariant under basis transformations, the fundamental parallelepiped has a unique shape for each different basis. All fundamental parallelepipeds will provide a tiling on the ambient space.

It is in a difficult problem to ascertain whether two lattices are isomorphic $\mathcal{L}_1 \cong \mathcal{L}_2$, requiring $N^{O(N)}$ time [HR14], but it is not hard to deduce whether two lattices, described by different bases, are the same as one another $\mathcal{L}(\mathbf{B}_1) = \mathcal{L}(\mathbf{B}_2)$. This is because each lattice posseses a unique Hermite Normal Form (HNF), which is a specific type of basis. The HNF can be computed efficiently, and so to establish whether two bases describe the same lattice, one simply checks if $HNF(\mathcal{L}(\mathbf{B}_1)) = HNF(\mathcal{L}(\mathbf{B}_2))$.

Definition 2.4 (Hermite Normal Form)**.** *The row HNF (where basis vectors are the rows of the matrix) of an* N*-dimensional lattice* $\mathcal{L}$ *is an upper triangular matrix* $\mathbf{H}$ *which satisfies the following conditions:*

- $\mathbf{H}_{ij} = 0$ *for* $i > j$*.*
- *The first nonzero term in a row from the left is called the pivot, is positive, and is strictly to the right of the pivot of the row above.*
- *Elements directly below a pivot are* 0 *and elements directly above are reduced modulo the pivot.*

A lattice $\mathcal{L}$ has *optimal* HNF if all-but-one pivots are 1. As a consequence there exists only one column – excluding pivots – of nonzero terms (the the elements above pivots of value 1 are reduced modulo 1, hence are all zeros). HNF bases are considered to be bad bases (if they were good bases then they should not be efficiently computable). An HNF basis $\mathbf{B}$ (with rows

being basis vectors) in optimal form would take the form

$$\mathbf{B} = \begin{bmatrix} 1 & 0 & 0 & \dots & d_1 \\ 0 & 1 & 0 & \dots & d_2 \\ 0 & 0 & 1 & \dots & d_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & D \end{bmatrix}, \tag{2.10}$$

where $0 < d_i \leq D$ for all $1 \leq i < N$.

2.2.1 Lattice problems

The properties of mathematical lattices that make them so attractive to cryptographers is the nature of the central lattice problems: their simplicity, their average case hardness, and the operations required to perform lattice manipulations, which fit well into current systems.

2.2.1.1 Short integer solution

The short integer solution (SIS) problem is an average case problem used to underpin the very earliest lattice-based cryptosystems [Ajt96].

Definition 2.5 (Shortest integer solution)**.** *Given m integer vectors $\mathbf{a}_i \in \mathbb{Z}_q^N$ which are together the rows of the matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times N}$, then find an integer vector $\mathbf{x} \in \mathbb{Z}^m$ of bounded length $\|\mathbf{x}\| \leq \beta$ for some real β such that*

$$f_{\mathbf{A}}(\mathbf{x}) = \mathbf{x}\mathbf{A} = \sum x_i \mathbf{a}_i = \mathbf{0} \in \mathbb{Z}_q^N. \tag{2.11}$$

The parameter β must be bounded above to prevent trivial but legitimate solutions. For example $\beta < q$ is necessary, otherwise $\mathbf{x} = (q, 0, \dots, 0)$ is an admissible solution. Parameter β (and number of vectors m) must also be large enough to ensure that some nontrivial solution does in fact exist. With increasing m the problem becomes easier, because one can always ignore rows of $\mathbf{A}$ if they are not needed, so extra $\mathbf{a}_i$ make the problem strictly easier. The

problem can also be generalized (known as the inhomogeneous form) so that instead of finding a combination of $\mathbf{a}_i$ that add up to $\mathbf{0}$ modulo q, they must sum to a nonzero vector $\mathbf{v}$ modulo q.

2.2.1.2 Shortest vector problem

Perhaps the simplest lattice problem is the shortest vector problem (SVP).

Definition 2.6 (Shortest vector problem)**.** *Given $\mathcal{L}(\mathbf{B})$, find the shortest nonzero lattice vector, which has length*

$$\lambda_1(\mathcal{L}) = \min\{\|\mathbf{v}\| \colon \mathbf{v} \in \mathcal{L}\backslash\{\mathbf{0}\}\}. \tag{2.12}$$

If one is given a good basis of $\mathcal{L}$ the problem is computationally easy, but if given a bad basis where the basis vectors are long and highly non-orthogonal, the problem is difficult.

The problem formulation can even be relaxed yet maintain its hardness. In the relaxed version, known as the approximate shortest vector problem (SVP_η), one is tasked with finding a short *enough* vector. Here short enough means that the vector should be less than a factor of η longer than the exact shortest vector, where η is some value which is polynomial in N, so the task is to find $\mathbf{v}$ where

$$\|\mathbf{v}\| \leq \eta\lambda_1(\mathcal{L}), \tag{2.13}$$

and $\eta = \text{poly}(N)$. One of the earliest researchers of lattices, Hermann Minkowski developed a crude bound for the maximum length of the shortest vector based only upon the dimension of the lattice and its covolume $D(\mathcal{L})$

Definition 2.7 (Minkowski's bound)**.** *An N-dimensional lattice $\mathcal{L}$ with covolume D has a shortest vector of length*

$$\lambda_1(\mathcal{L}) \leq \sqrt{N}|D|^{1/N}. \tag{2.14}$$

Proof of Minkowski's bound. Minkowski's convex body theorem states that any convex, centrally symmetric body S of volume $V(S) > 2^N det(\mathcal{L})$ contains at least one nonzero lattice vector. The proof of Minkowski's bound follows from this.

Let $v = \min\{\|\mathbf{x}\|_\infty : \mathbf{x} \in \mathcal{L}\backslash\{\mathbf{0}\}\}$, and assume $v > det(\mathcal{L})^{1/N}$. Consider the open hypercube C of side length $2v$ centred at the origin. This has volume

$$V(C) = 2^N v^N > 2^N det(\mathcal{L}),$$

and so being a symmetric convex open set centred at the origin, must contain at least one nonzero lattice point $\mathbf{l}$. But due to the definition of C, the lattice point contained in C must have $\|\mathbf{l}\|_\infty < v$, which contradicts the definition $v = \min\{\|\mathbf{x}\|_\infty : \mathbf{x} \in \mathcal{L}\backslash\{\mathbf{0}\}\}$, and so the assumption is contradicted. □

Consequently, for any lattice $\mathcal{L}$ there must exist a point $\mathbf{x}$ with infinity norm $\|\mathbf{x}\|_\infty \leq det(\mathcal{L})^{1/N}$. Due to the fact that the Euclidean norm of an N-dimensional vector is bounded by $\|\mathbf{v}\| \leq \sqrt{N}\|\mathbf{v}\|_\infty$, one can say that $\lambda_1(\mathcal{L}) \leq \sqrt{N} det(\mathcal{L})^{1/N}$.

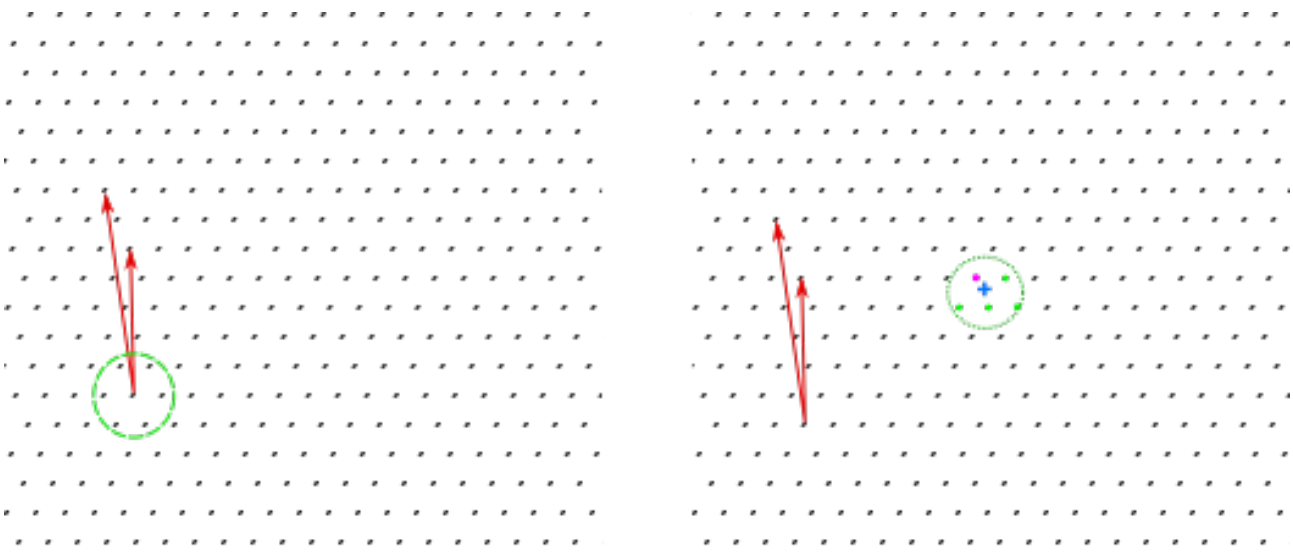

Figure 2.3: Illustration of SVP (left) and CVP (right) over a two-dimensional lattice. Any of the points in the green circle on the left would solve exact SVP (though typically there would just be one unique shortest vector, up to sign). On the right hand diagram, the blue cross represents a target vector, with the pink lattice point solving exact CVP, and green points solving approximate CVP where the appoximation factor is indicated by the green dashed ball around the target vector.

2.2.1.3 Closest vector problem

A closely related problem to SVP is the closest vector problem (CVP).

Definition 2.8 (Closest vector problem)**.** *Given a basis for the lattice as well as a point in the ambient space* $\mathbf{x} \in \mathbb{R}^M$*, and tasked with finding the closest lattice vector* $\mathbf{v}$ *to* $\mathbf{x}$*, so that*

$$\|\mathbf{v} - \mathbf{x}\| = \min\{\|\mathcal{L} - \mathbf{x}\|\}. \tag{2.15}$$

As with SVP, CVP also has a more relaxed format, CVP_η, where one must find a lattice vector $\mathbf{v}$ that is close-enough to $\mathbf{x}$, again with close-enough meaning within some polynomial multiplicative factor η

$$\|\mathbf{v} - \mathbf{x}\| = \gamma \min\{\|\mathcal{L} - \mathbf{x}\|\}. \tag{2.16}$$

A special case of the closest vector problem is the bounded distance decoding problem (BDD), where one is required to solve exact CVP, under the understanding that the target vector $\mathbf{x}$ is quite close to the lattice,

$$\|\mathcal{L} - \mathbf{x}\| < \alpha\lambda_1 L, \tag{2.17}$$

and α is usually taken to be in $(0, 1/2)$ so as to ensure that in a ball of diameter $\alpha\lambda_1(\mathcal{L})$ around $\mathbf{x}$ there is exactly one lattice point.

2.2.1.4 Shortest independent vectors problem

The shortest independent vectors problem (SIVP) is similar to SVP, but requires more information about a lattice.

Definition 2.9 (Shortest independent vectors problem)**.** *Given a lattice* $\mathcal{L}(\mathbf{B})$*, output a set of* N *linearly independent vectors* $\mathbf{v}_i$ *such that* $\max\{\|\mathbf{v}_i\|\} \leq \max\{\|\mathbf{b}_i\|\}$ *where all bases are considered.*

Note that outputting such a series of vectors does not guarantee that it is still a basis for the lattice, only that the set of vectors $\{\mathbf{v}_1, \dots \mathbf{v}_N\}$ is a basis for a sublattice $\mathcal{L}' \subseteq \mathcal{L}$.

In the η approximate version, it is just necessary that $\max\{\|\mathbf{v}_i\|\} \leq \eta\lambda_d(\mathcal{L})$ with $\lambda_d(\mathcal{L})$ being the d^{th} successive minima of the lattice.

2.2.2 Lattice algorithms

Due to the highly interconnected nature of the central lattice problems, and owing to many decades of hardness reductions between them [Mic08], many of the best algorithms can be tweaked subtly to address a related problem. This is advantageous because, while strong reductions between problems increases the attack surface for cryptanalysts to undermine the security of lattice-based cryptography, it means that security of the field can be succinctly described and analysed with just a few approaches. The ability to find short lattice vectors is important to all problems, and there are two main approaches to this: enumeration and sieving. Lattice enumeration algorithms can be considered brute force in their simplest sense, as they involve drawing a large ball around the origin, then enumerating every lattice vector within the ball, under the knowledge that so long as the radius of the ball is larger than a certain bound (e.g. Minkowski's bound of Eq. (2.14)), the shortest vector will deterministically reside within the set of enumerated vectors. Sieving, meanwhile, is probabilistic in nature. This approach gathers a large list of lattice vectors, sampled from some appropriate distribution, then iteratively combines the vectors to generate new lattice vectors, with the output vectors getting shorter and shorter with each pass. In most cases, however, preprocessing algorithms are performed in order to attain an appropriate basis from which to start.

2.2.2.1 Basis reduction

Basis reduction is at the core of lattice algorithms, which display a duality in that strong basis reduction implies short vectors, and short vector algorithms allow for better basis reduction. One of the most celebrated basis reduction algorithms is the Lenstra-Lenstra-Lovasz (LLL) algorithm [LLL82], whose output is called an LLL-reduced basis.

The Gram-Schmidt values are $\mu_{ij} = \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle}$. For a basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_N\}$, its orthonormal basis is $\mathbf{B}^* = \{\mathbf{b}_1^*, \ldots, \mathbf{b}_N^*\}$, determined by the Gram-Schmidt orthonormalization process, in which vectors are iteratively projected onto a subspace orthogonal to those processed before them.

A basis is said to be LLL-reduced if it is

- Size reduced: for $1 \leq j < i \leq N$ Gram-Schmidt coefficients $|\mu_{ij}| < 0.5$
- Lovasz condition: for $k = 2, \ldots N$, $\delta \|\mathbf{b}_{k-1}^*\|^2 < \|\mathbf{b}_k^*\|^2 + \mu_{k,k-1}^2 \|\mathbf{b}_{k-1}^*\|^2$

The LLL algorithm, described in Algorithm 1 outputs an LLL-reduced basis, and proceeds by performing two steps iteratively, until both conditions are satisfied between all pairs of basis vectors. Aside from the input basis, the practitioner just has to decide on parameter $\delta \in (1/4, 1]$ which determines the trade-off between the speed of the algorithm and how well reduced the final basis is. The LLL algorithm runs in polynomial time, and so is highly practical, though it achieves only an exponential approximation to the shortest vector, which can be shown to be $\gamma(N) = \left(\frac{2}{\sqrt{3}}\right)^N$. Picking $\delta = 1$ enables the algorithm to output a polynomial approximation to the shortest vector, though it runs in exponential time, making it impractical. Such is the efficiency and simplicity of LLL, it has been adapted to a variety of use cases [LMHG10, LMHG13] and expanded to include variants over number fields [Sim09] and complex numbers [GLM09].

Algorithm 1: LLL basis reduction

Input : Basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_N\}$, parameter $\delta \in (1/4, 1]$ often taken to be $\delta = 3/4$

Output: LLL-reduced basis $\mathbf{B}_{LLL}$

`/* Step 1: size reduce */`

for $i = 1$ *to* N **do**
  for $k = i - 1$ *to* 1 **do**
    $m \leftarrow \lfloor \mu_{ki} \rceil$
    $\mathbf{b}_i \leftarrow \mathbf{b}_i - m\mathbf{b}_k$

`/* Step 2: Lovasz condition */`

for $i = 1$ *to* $N - 1$ **do**
  if $\delta \|\mathbf{b}_i^*\|^2 > \|\mathbf{b}_{i+1}^* + \mu_{i,i+1}\mathbf{b}_i^*\|^2$ **then**
    Swap $\mathbf{b}_i, \mathbf{b}_{i+1}$ `// Go to Step 1`

While the LLL algorithm reduces bases by considering adjacent vectors in the basis and pairwise comparison, there are stronger forms of reduction which consider many vectors at once. One such technique is Hermite Korkine Zolotarev (HKZ) reduction [KZ73, Her50] which reduces the entire basis simultaneously, or Block Korkine Zolotarev (BKZ) reduction which can be seen as an interpolation between the two. In BKZ, a practitioner chooses a block size which determines how many vectors to reduce at once. Larger block sizes give stronger reductions, but are more costly, and BKZ with a block size of 2 is just the LLL algorithm: BKZ can be thought of as a generalization of LLL.

2.2.2.2 Discrete Gaussian sampling

The ability to generate lattice vectors is essential to many algorithms, some of which are described in the following. In particular, the abililty to control the distribution from which one samples is important. For example, if one could sample from a distribution of short-enough lattice vectors efficiently, then this could be used on its own to generate solutions to problems such as SVP in a short amount of time. The discrete Gaussian sampler (DGS) is an area subject to much research. As well as being used for cryptanalytical purposes, DGS is used in the implementation of cryptographic algorithms, as randomized lattice instances must be used for each new instantiation of a cryptosystem. This opens up DGS to a new range of analyses, as it's use in cryptographic implementations requires extra properties such as side channel resilience. A discrete Gaussian distribution over the integer lattice in two dimensions is illustrated in Fig. 2.4, with the z axis indicating the probability for a particular point (x, y) to be sampled.

A discrete Gaussian over a lattice $\mathcal{L}$ can be written $D_{\mathcal{L},s,\mathbf{c}} = \exp\left\{\frac{-\pi\|\mathbf{v}-\mathbf{c}\|^2}{s^2}\right\}$ where the distribution is centred at $\mathbf{c} \in \mathbb{R}^N$, and the parameter s is the standard deviation. Importantly such a distribution does not depend on geometry of the input basis, and the first DGS algorithm, known as the Klein sampler [Kle00] was applied cryptographically in 2008 [GPV08]. Peikert's sampler [Pei10] is more parallelizable, at the cost of outputting longer vectors which is a disadvantage in a cryptographic context.

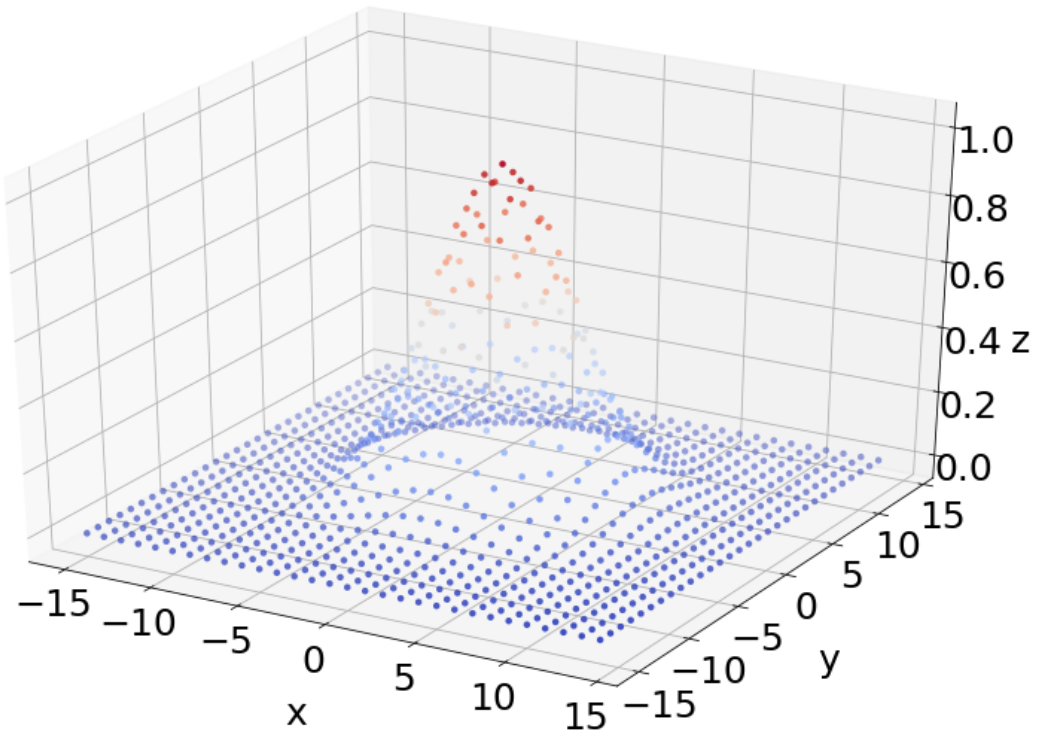

Figure 2.4: Visualization of a discrete Gaussian distribution over $\mathbb{Z} \times \mathbb{Z}$. In rejection sampling, the (x, y) coordinates would be uniformly random sampled over the feasible space, and accepted with probability given by the z coordinate.

In order to sample from DGS, it is necessary to use some algorithm which samples from a one-dimensional Gaussian distribution over the integers. The simplest of these is rejection sampling, where one picks an integer uniform randomly over the feasible options, and then accepts the integer with probability proportional to the value of the Gaussian at that integer. This method of sampling, however, is extremely vulnerable to side channel attacks. Inversion sampling requires precomputed distribution tables, whereas the Knuth-Yao [KDA76] method is very space efficient but requires extra processing in order to become time-independent and so cryptographically safe to use.

2.2.2.3 Enumeration

Enumeration is a powerful weapon in the cryptanalyst's toolbox, and particularly so when combined with cutting edge techniques, to be discussed shortly. Enumeration is typically

performed on a preprocessed basis, which means a basis which has already been reduced using some choice of algorithm (e.g. LLL, BKZ, HKZ, or more sophisticated variants).

Given such a reduced basis, a good bound on the length of the shortest vector R, and the assurance that the shortest vector is unique up to sign, solving SVP is simply the task of finding a vector $\mathbf{w} \in \mathcal{L}$ suc that $\|\mathbf{w}\| < R$, which makes checking for correctness easy.

An enumeration tree is formed from the vectors of the projected lattices $\pi_N(\mathcal{L}), \ldots, \pi_1(\mathcal{L})$ with norm less than R. The enumeration tree has depth N, and at depth d, it contains all the vectors in the rank d projected lattice $\pi_{N+1-d}(\mathcal{L})$ with norm less than R. The root of the tree is the zero vector, and the leaves are the lattice vectors of $\mathcal{L}$ which are shorter than R. The nodes at each level are ordered by Euclidean norm, and as the depth increases, the children of a given node are *at least* as long as the parent.

Once this is done, in the Schnorr-Euchner algorithm [SE94], a Depth First Search is performed on the tree (depth first due to the fact that a shorter length at the lowest levels of the tree is likely to persist through to the leaf nodes, without any information to the contrary).

This version of enumeration runs in exponential time, but this can be improved. The way to improve efficiency is to reduce the number of nodes evaluated, and so the question arises of how to select which nodes to discard. It was proposed to consider a different ball of radius R_k for each level of the tree where $R_1 \leq \ldots \leq R_N$ [SH95], meaning that the enumeration zone is no longer a ball. This method is called enumeration with pruning and the logic behind this is that since nodes at each subsequent layer of the enumeration tree will have longer Euclidean norm, there is little point considering elements which are already close to the edge of the ball (i.e. vectors of $\pi_1(N)$ with norm close to R) as they will be discarded at a later stage with high probability, and this method is shown in Fig. 2.5, along with classical enumeration, both demonstrated on a two-dimensional lattice. Extreme pruning developed the field [GNR10] by taking drastically smaller R_i compared with R for small i. The interesting aspect to this technique is that when considering such a small enumeration area, the probability of finding the shortest vector actually goes to zero, but by a factor smaller than the reduction in computation. Accordingly, one randomizes the input basis using transformations of small unimodular matrices and repeats

the enumeration with extreme pruning many times. Thus one can achieve the same probability of success as pruned enumeration, only with vastly fewer computations, and repeating the entire algorithm with randomized bases.

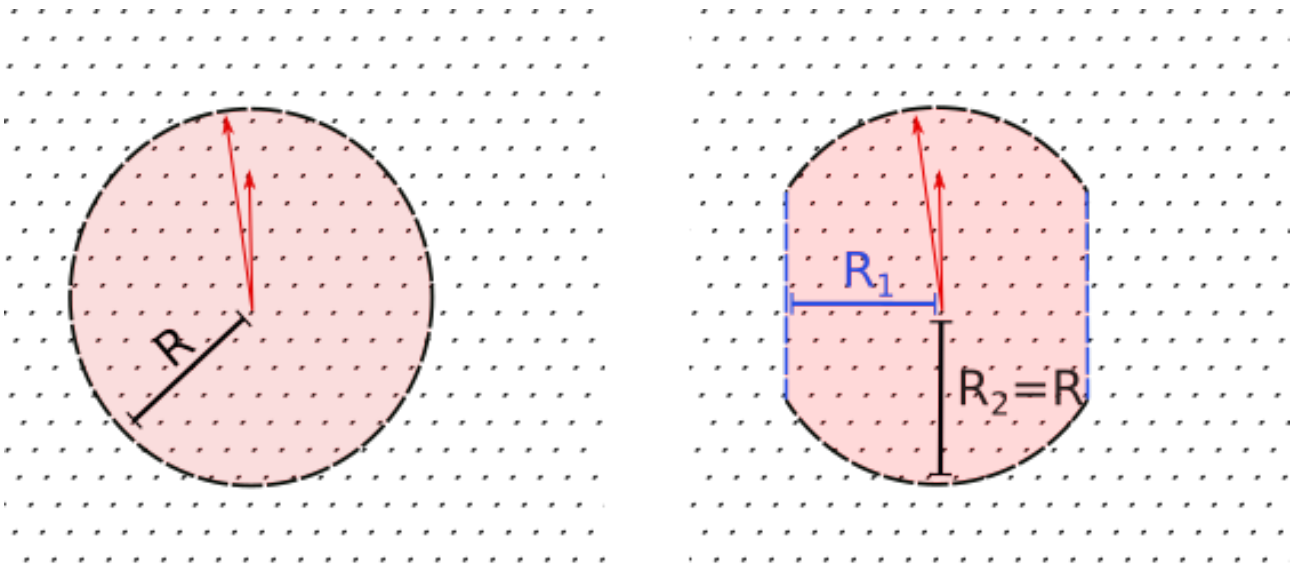

Figure 2.5: Diagram of enumeration (left) and pruned enumeration (right) on a two-dimensional lattice. Whereas in basic enumeration all points within a ball of radius R are evaluated, in pruned enumeration, only points in the projected lattices $\pi_i(\mathcal{L})$ with norm less than R_i are enumerated.

2.2.2.4 Sieving

The first sieving algorithm, known as the AKS sieve, was proposed at the turn of the century [AKS01] and was the first algorithm to provably solve SVP in single exponential time 2^{cN}. The AKS sieve initializes a long list of vectors sampled from a large ball of dimension R around the origin, and calls this list Y. At the next iteration of the sieve Y is reduced to C, in which points $\mathbf{y}, \mathbf{y}'$ which are within distance $R/2$ of each other are replaced be a single point $\mathbf{y}$ in the sieved set C.

The list sieve [MV10], meanwhile, begins with an empty list and adds vectors to it. In this variant, a new lattice vector $\mathbf{v}$ is generated and then reduced with respect to the other vectors in the list by subtracting them from it. After $\mathbf{v}$ can be reduced no more, it is added to the list. Considerations in the list sieve are to make sure that the list vectors maintain large enough angles between one another, and to ensure that one does not obtain collisions, as this

represents wasted effort. The fastest variant, ListSieve-Birthday, uses the Birthday Paradox to avoid collisions [PS09].

Other combinatorial algorithms are closely related to sieving, such as the $2^{O(N)}$ time SVP algorithms based on discrete Gaussian sampling (DGS) [ADRSD15, ASD17]. In these algorithms DGS is used to generate a large list of vectors distributed around the origin. In contrast to earlier algorithms which consider the differences of lattice vectors, this algorithm takes the average of lattice vectors, provided that they reside in the same coset modulo $2\mathcal{L}$, of which there are 2^N cosets.

When weighing up the relative merits of sieving algorithms and enumeration algorithms there are a number of factors to consider. Enumeration is asymptotically slower, generally taking around $N^{O(N)}$ time whereas sieving achieves single exponential runtime, with the combinatorial algorithms based on DGS achieving $2^{O(N)}$ runtime. For low dimensions, however, enumeration runs faster, and the point of crossover is an area of research with the sieving superiority having now been demonstrated for dimensions as low as 70 [ADH+19]. Furthermore, sieving algorithms, while fastest for cryptographic-sized lattices, are impractical in high dimensions due to their exponential space requirements.

2.2.2.5 Quantum adaptations

In order for lattice-based cryptography to be considered quantum-safe, it must stand up to quantum cryptanalysis. While there are a small number of quantum algorithms for which known speed-ups are available, there is much that is still unknown about the potential of quantum computers to solve various computationally hard problems. In the quest to determine the quantum resilience of LBC, one intuitive method is to investigate speed-ups of classical algorithms such as enumeration and sieving, by applying Grover's search of an unstructured database. In this case one seeks to apply Grover's search as a quantum subroutine wherever it is appropriate. Due to Grover's quadratic speed-up of unstructured search, it is not possible to break LBC in this manner, but nevertheless, important cutting edge asymptotic scaling can be derived using these techniques [LMVDP13] which has an impact on parameter choices for

potential cryptosystems.

In 2018 quantum speed-ups of enumeration were obtained [ANS18] by applying quantum tree algorithms [Mon18] to reduce T operations of a classical enumeration tree to $\sqrt{T}$ quantum operations.

Quantum period finding algorithms have been used to demonstrate quantum polynomial time reductions from the learning with errors problem (LWE) – discussed in the following – to the extended dihedral coset problem [BKSW18], and to find short vectors in number fields in quantum polynomial time [CDW17].

2.2.3 Lattice cryptosystems

The first lattice-based cryptographic constructions were proposed in the late Nineties [Ajt96, AD97]. The security proofs demonstrated that breaking the cryptosystems were at least as hard as solving SVP and SIS. In particular, a crucially important result was showing that SIS is hard *on average* if SVP_η is hard in the *worst case*. The significance of this is that breaking a cryptosystem should be hard on average (as one picks different instances at each instantiation), but average case hardness is generally harder to prove than worst case hardness, so the reduction to the shortest vector problem played a large role in cementing lattices as appropriate vehicles for cryptography, and the first fully-fledged lattice cryptosystem became known as the Ajtai-Dwork cryptosystem.

While highly theoretically significant, an analysis demonstrated that cryptosystems such as Ajtai-Dwork would require impractically large keys [NS98]. A signature scheme by Goldreich, Goldwasser, and Halevi (GGH) was released around the same time [GGH97] but was later cryptanalyzed [Ngu99].

Shortly afterwards, a cryptosystem based on N^{th} degree truncated polynomial rings (NTRU) was proposed, and named NTRUEncrypt. The scheme is based upon the NTRU assumption, which is that factoring polynomials in a truncated polynomial ring, into polynomials with small coefficients, is difficult. Breaking the assumption is closely related to lattice reduction. A

related signature scheme called NTRUSign was based upon the GGH scheme, and the hardness of solving CVP in NTRU lattices [HHGP+03] but was broken when it was found that repeated calls to the system leaked information about the geometry of the private basis [NR06]. A fix for this was developed that utilized DGS to allow unlimited calls to the scheme without leaking any information about the geometry of the basis [SS11].

2.2.3.1 Learning with errors

Perhaps the most celebrated of all lattice-based cryptographic constructions is the learning with errors problem [Reg05]. It was the first cryptosystem with security proven under worst case hardness assumptions, and in the intervening years work has been done to improve the security proof [BLP+13, Pei09].

There are two variants of the learning with erros problem, which have been shown to be equivalently hard. One is called search-LWE: given M samples of $(\mathbf{a}, b = \mathbf{as}+e)$, find the secret $\mathbf{s}$, where $\mathbf{a} \in \mathbb{Z}_q^N$, $\mathbf{s} \in \mathbb{Z}_q^N$, and e is some small error $e \in \mathbb{Z}_q$, find $\mathbf{s}$. This can be equivalently stated as finding $\mathbf{s}$ given the matrix and vector pair $(\mathbf{A}, \mathbf{b} = \mathbf{As}+\mathbf{e})$ where $\mathbf{A} \in \mathbb{Z}_q^{M\times N}$, $\mathbf{s} \in \mathbb{Z}_q^N$ and the error vector has small coefficients $\mathbf{e} \in \mathbb{Z}_q^M$. The dimensions and calculation of $(\mathbf{A}, \mathbf{b})$ are shown in Fig. 2.6. It is easy to see that with no error at all, it would be trivial to find $\mathbf{s}$ by computing $\mathbf{A}^{-1}\mathbf{b}$ which is efficiently computable. However the addition of a small error makes the problem extremely computationally challenging.

In order to interpret LWE as a lattice problem, one should consider the search variant as an instance of CVP. The matrix $\mathbf{A}$ is a lattice basis in column format, $\mathbf{s}$ is some linear combination of the columns of $\mathbf{A}$, and so $\mathbf{As}$ is a lattice point. The perturbation due to $\mathbf{e}$ means that one is given point $\mathbf{b}$ in the ambient space $\mathbb{Z}_q^N$, and finding the closest lattice vector (which with error drawn from an appropriate distribution, should be unique, making it an instance of the BDD problem) means finding $\mathbf{As}$, from which it is easy to deduce the vector $\mathbf{s}$.

Algorithm 2: Sample LWE encryption

Input : Modulus q, usually prime with $N^2 \leq q \leq 2N^2$, number of samples M, error distribution χ

`/* Private key */`

1 $priv = \mathbf{s} \leftarrow \mathbb{Z}_q^N$ uniform random

`/* Public key */`

2 $\mathbf{A} \leftarrow \mathbb{Z}_q^{M \times N}$ uniform random

3 $\mathbf{e} \leftarrow \chi^M$

4 $pub = (\mathbf{A}, \mathbf{b} = \mathbf{As} + \mathbf{e})$

`/* Encryption */`

5 Encryption of bit $x \in \{0, 1\}$

6 $\mathbf{t} \leftarrow \{0, 1\}^M$ represents a random subset of samples

7 $(\mathbf{a}, b) = (\mathbf{tA}, \frac{x}{2} + \mathbf{tb})$

`/* Decryption */`

8 Decrypt $(\mathbf{a}, b)$ as 0 if $b - \frac{\mathbf{as}}{q}$ rounds to 0 and 1 otherwise.

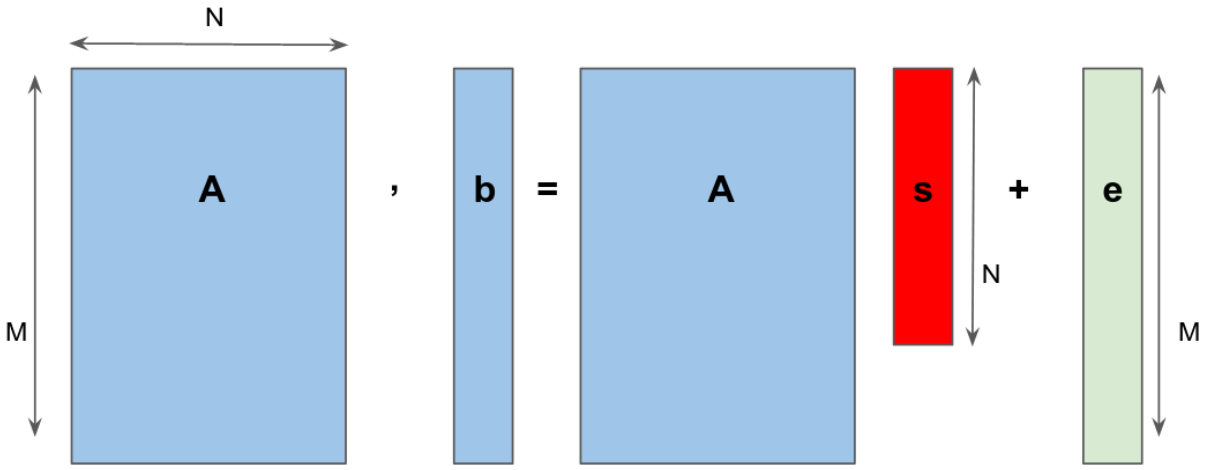

Figure 2.6: Graphical illustration of linear algebra calculation of the public key in the learning with errors problem, the hardness of which provides the security of LWE-based cryptosystems. Blocks in blue are publicly available, the block in red is the private key which should remain secret, and the block in green is a small error vector which should not be disclosed. The public key is $(\mathbf{A}, \mathbf{b})$ where $\mathbf{b}$ is calculated as shown, using the secret key and some small error.

A great strength of the LWE framework is that it generalizes well to more sophisticated number fields, yielding improved efficiencies. In the Ring learning with errors (RLWE) format, one

replaces integers with polynomials whose coefficients are drawn from some finite field which is typically taken to be $\mathbb{Z}_q$ from some prime q. The set of polynomials over $\mathbb{Z}_q$ with the operations of addition and multiplication are the infinite polynomial ring $\mathbb{Z}_q[x]$, and RLWE takes place in a quotient ring of this given by $\mathbb{Z}_q[x]/\phi(x)$. Typically the quotient polynomial is $\phi(x) = x^N + 1$ (often with $N = 2^k$). A polynomial in this ring takes the form

$$a(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{N-1} x^{N-1},$$

which means matrix $\mathbf{A}$ in the LWE problem can be represented by a circulant matrix of the form

$$\mathbf{A} = \begin{bmatrix} a_1 & -a_N & \ldots & -a_2 \\ a_2 & a_1 & \ldots & -a_3 \\ \vdots & \vdots & \ddots & \vdots \\ a_N & a_{N-1} & \ldots & a_1 \end{bmatrix}, \tag{2.18}$$

which can in turn be represented by the single element of the polynomial ring $a(x)$. While this introduces more structure to the LWE problem, this structure has not yet been successfully attacked, but yet it yields an $O(N)$ improvement in communication efficiency, because the public key can be represented by a single polynomial $a(x)$ of degree $N - 1$ (hence N coefficients), as opposed to a dense matrix $\mathbf{A}$.

Further developments have been on LWE which include generalizing the efficiency/security tradeoff that RLWE offers [AD17], and generalizing to non-commutative rings [GLV20].

2.3 Quantum Computing

Quantum computing is essentially the manipulation of a quantum state to carefully choreograph an interference pattern which corresponds to some pre-encoded problem. The interference pattern should then exhibit tall peaks at eigenstates which represent a good solution. Upon measurement of the final state, the tall peaks at desired solutions manifest as high probabilities of finding correct solutions. Put simply, a quantum algorithm has three components:

1. Preparation of an initial state $|\psi_0\rangle$,
2. Manipulation of $|\psi_0\rangle$ through application of unitary operators,
3. Measurement of final state $|\psi\rangle$.

While classical computers all use the same digital architecture, quantum computing uses many different approaches, both analogue and digital, continuous time and discrete time.

2.3.1 Adiabatic quantum computing

Adiabatic quantum computing (AQC) is a form of continuous time quantum computing (CTQC) proposed in [FGGS00]. In AQC one initializes a system in state $|\psi_0\rangle$ which is an eigenstate of some simple driver Hamiltonian H_D. One then gradually perturbs the Hamiltonian to a new problem Hamiltonian H_P into which a computational problem is encoded. The canonical way that problems are encoded to H_P is that low energy eigenstates correspond to good solutions, and often the optimal solution is given by the ground state of H_P.

The adiabatic theorem states that if a system is perturbed slowly enough relative to eigenspectrum of the Hamiltonian, the system remains in its instantaneous eigenstate. This theorem means that if one initializes the system in a certain eigenstate of H_D and interpolates slowly enough, one can measure the system at completion to be in the same eigenstate of H_P with probability $\simeq 1$. Fig. 2.7 demonstrates the evolution of the energy eigenspectrum of Hamiltonian H_t as it interpolates between H_0 and H_P, where the speed of the interpolation depends

sensitively on the minimum energy gap between the ground (blue) and first excited (red) states, as explained in the following.

For an adiabatic algorithm of total sweep time T, the Hamiltonian at time $t \in [0, T]$ can thus be written

$$H_t = (1 - f(t))H_D + f(t)H_P, \tag{2.19}$$

subject to the boundary conditions

$$f(0) = 0, \quad f(T) = 1. \tag{2.20}$$

While the problem Hamiltonian encodes the computational problem, it is down to the driver Hamiltonian H_D to induce quantum dynamics, so that the entire feasible solution space is explored.

In classical computing and gate model quantum computing, algorithmic performance is assessed by the number of operations required. Because AQC is analogue, the equivalent performance metric is the time sweep length T. Sweep length T depends on the minimum eigenspectrum gap between the ground and first excited states (for AQC where one initializes in the ground state and aims to preserve the system in the ground state) given by

$$\Delta = \min_t \{E_1 - E_0\}. \tag{2.21}$$

Then the time scaling is

$$T = O\Big(\frac{1}{\Delta^2}\Big). \tag{2.22}$$

This result reduces complexity analysis of AQC to lower bounding of the minimum gap size in the eigenspectrum of the intermediate Hamiltonian H_t, though this is generally a difficult problem, making performance assessment of AQC challenging. It has been demonstrated, however, that AQC is equivalent (up to a polynomial factor) to the better-analyzed gate model [AvDK+04], and that it is universal and equivalent to standard quantum computation under certain conditions [KKR06].

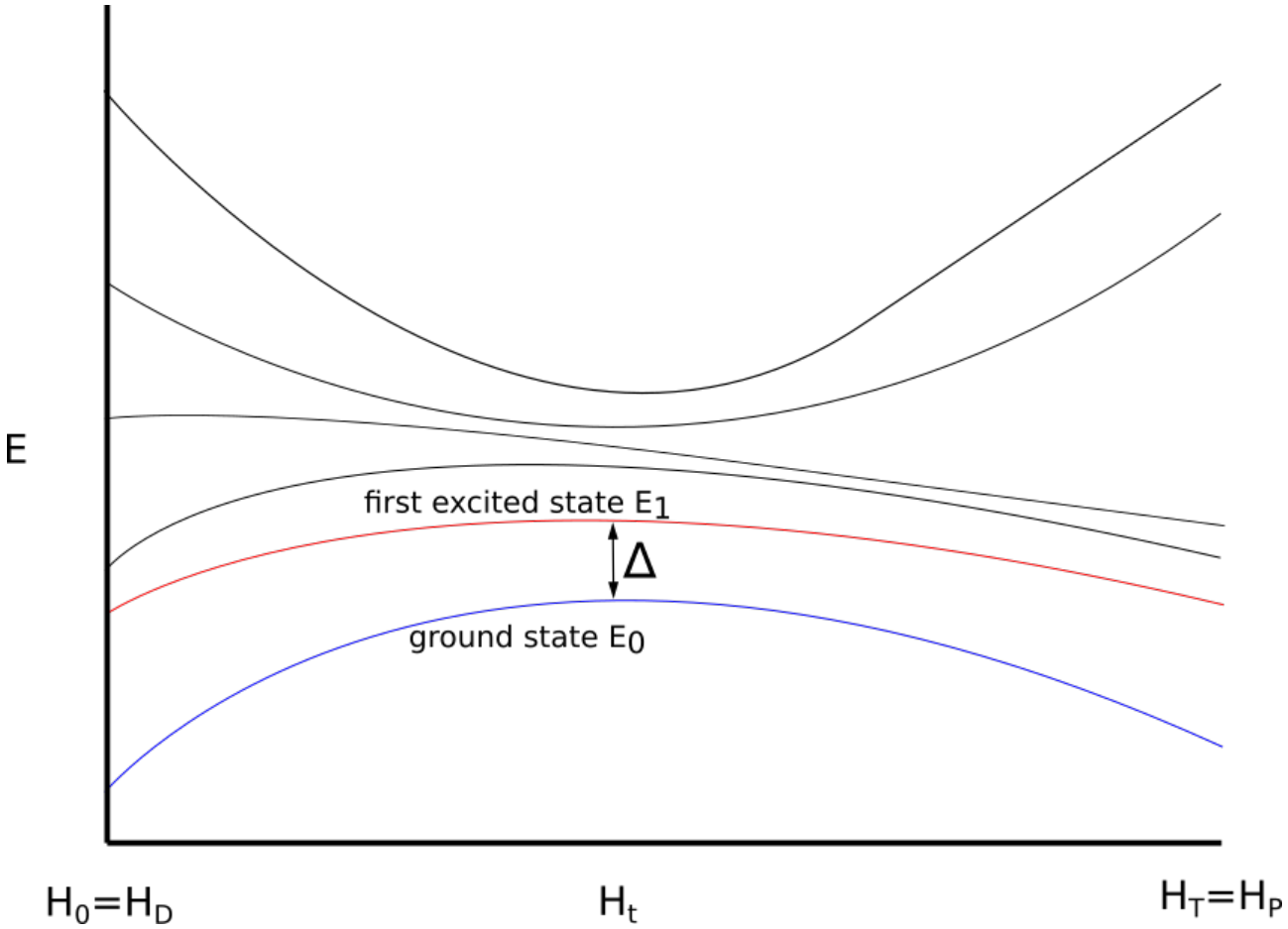

Figure 2.7: Diagram illustrating the evolution of the eigenspectrum of H_t from $t = 0$ to $t = T$. Having initialized $|\psi_0\rangle$ to be the ground state of H_D, to remain in the ground state (blue) the sweep length T must be long enough relative to Δ – the minimum gap between the red and blue lines – in order to prevent a thermal transition to the first excited state shown by the red line.

2.3.2 Transverse Ising model

The transverse Ising model is a quantum adaptation of the classical Ising model, which was originally conceived as a way to understand the properties of ferromagnetism in materials. The classical Ising model is written

$$H = -\sum_{\langle ij \rangle} J_{ij} s_i s_j - \mu \sum_i h_i s_i, \tag{2.23}$$

where the s_i are spins taking values in $\{-1, +1\}$, the J_{ij} are the interactions between spin i and spin j, and $\langle ij \rangle$ indicates that the interaction terms only occur between nearest neighbours. The h_i represent the local magnetic field at spin i, and μ is the magnetic moment. Materials with $J > 0$ are called ferromagnetic as their lowest energy states are those for which spins align with their neighbours.

In the quantum setting, the s_i are replaced by Z_i operators, which operate on the system by applying a Pauli-Z operator to the i^{th} qubit, where the Pauli operators are defined by the matrices

$$\mathbb{1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{2.24}$$

The Z_j operator can be expressed on the full Hilbert space of n qubits as

$$Z_j = \otimes_1^{j-1} \mathbb{1} \otimes Z \otimes_{j+1}^{n} \mathbb{1}. \tag{2.25}$$

The quantum Ising Hamiltonian is thus expressed

$$H_{Ising} = -\sum_{ij}^{n} J_{ij} Z_i Z_j - \sum_i^n h_i Z_i, \tag{2.26}$$

where the J_{ij} interaction terms and h_i local energies can be chosen according to the computational problem under consideration. So long as the quantum device is capable of all-to-all connections, J_{ij} terms can be included between all i, j, though many quantum processors do not offer this property at present.

While the quantum Ising Hamiltonian is used to express the computational problem in a Pauli-Z eigenbasis, the transverse field is applied via the transverse Hamiltonian

$$H_{transverse} = \nu \sum_{i}^{n} X_i, \tag{2.27}$$

where X_i are defined identically to Z_i (only with Pauli-X operators), and ν is field strength. This Hamiltonian represents the application of a local field to each qubit perpendicular to the Z direction. It has the effect of relaxing the system and hence driving the quantum exploration of the feasible solution space. According to the AQC framework, one can construct an adiabatic quantum Ising algorithm according to

$$H_t = (1 - f(t))H_{transverse} + f(t)H_{Ising}. \tag{2.28}$$

The appeal of the transverse Ising model is that a great number of combinatorial optimization problems can be expressed in the form of H_{Ising} [Luc14], and it lends itself naturally to a number of systems capable of implementing the the Hamiltonians of Eq. (2.28), such as the D-Wave processor discussed (subject to certain adaptations, discussed later), and the coherent Ising machine of [HHF+16, HIM+19].

2.3.3 The Bose-Hubbard model

One physical system proposed for computation is the Bose-Hubbard model [GK63] which describes the behaviour of an ultra-cooled bosonic gas in a periodic potential landscape, known as an optical lattice. The periodic landscape has a series of minima known as sites, and it is proposed that by manipulating the system, and subsequently measuring the number of bosonic particles present at the various sites, one can obtain candidate solutions to computational problems. One popular technique is sweeping between superfluid and Mott insulator phases [SUXF06].

The Bose-Hubbard Hamiltonian consists of a quantum tunneling term and an onsite interaction

term, which are defined via the annihilation (a_i), creation ($a_i^\dagger$), and number ($\hat{n}_i$) operators

$$\begin{aligned} a_i \left|n_i\right\rangle &= \sqrt{n_i}\left|n_i - 1\right\rangle, \\ a_i^\dagger \left|n_i\right\rangle &= \sqrt{n_i + 1}\left|n_i + 1\right\rangle, \\ a_i^\dagger a_i &= \hat{n}_i. \end{aligned} \tag{2.29}$$

The most basic version of the model Hamiltonian takes the form

$$H = A\sum_{\langle i,j\rangle}(a_i^\dagger a_j + a_j^\dagger a_i) + U\sum_i \hat{n}_i(\hat{n}_i - 1), \tag{2.30}$$

where the sites are given an undirected graph structure with the sites as vertices and tunnelling terms between vertices which share an edge on the graph.

The Bose-Hubbard model has been shown to be Quantum Merlin Arther (QMA) complete for certain parameter choices [CGW14], meaning it cannot be efficiently classically computed, and has been demonstrated to offer universal quantum walk-based computation [UF12].

2.3.4 The circuit model

In the circuit model (or gate model) of quantum computing one designs a circuit of quantum logic gates analogous to an electrical circuit, and applies the quantum logic gates to a register of n qubits. Each of the gates creates a reversible unitary transformation on the state $|\psi\rangle$. Gates can act on any number of qubits, but for practical purposes, most circuits are devised using single and two-qubit gates. The space of single-qubit gates is generated by the Pauli matrices of Eq. (2.24). A quantum computer needs a library of quantum gates in order to approximate general unitary transformations, and it has been shown that such a universal library can be constructed using the space of single qubit gates and virtually any fixed two-qubit gate [BBC+95], though typically controlled NOT (CNOT) gates are used [Sle06], which

is

$$U_{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{2.31}$$

For the contribution of this thesis, however, the only two-qubit gates required are the ZZ gate given by

$$ZZ = Z \otimes Z = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.32}$$

2.3.5 Shor's algorithm

Potentially the most significant quantum algorithm so far discovered is that of Shor for factorizing integers, and it is presented here due to its cryptographic significance and motivation of the field of PQC.

Assume $N = pq$ is an odd integer and p, q are distinct primes. The aim is to find a square root b of 1 modulo N that is not either $+1$ or -1. This means solving for a nontrivial solution to

$$\begin{aligned} b^2 &= mN + 1, \\ (b+1)(b-1) &= mN, \end{aligned} \tag{2.33}$$

as the left hand side of the equation is the difference of two squares. This gives nontrivial divisors $gcd(b+1, N)$ and $gcd(b-1, N)$ where gcd denotes the greatest common divisor. Shor's algorithm can be broken into a classical part described in Algorithm 3 and a quantum period finding subroutine.

For example, with $N = 21 = 7 \times 3$, the period $r = 6$ and with $a = 2$ we have $gcd(2^3 + 1, 21) = gcd(9, 21) = 3$ and $gcd(2^3 - 1, 21) = gcd(7, 21) = 7$.

In Algorithm 4, the initialization $|\psi_0\rangle$ can be performed by the application of Hadamard gates,

Algorithm 3: Shor's algorithm

Input : Semiprime integer N
Output: Distinct primes p, q where $N = pq$

1 Pick random a where $1 < a < N$
2 Compute $gcd(a, N)$ using Euclidean algorithm (efficiently computable)
3 **if** $gcd(a, N)$ *is non-trivial* **then**
4 $\quad$ $gcd(a, N) = p$ or $gcd(a, N) = q$ so algorithm terminates successfully ▷ Vanishingly unlikely
5 Use the quantum period finding subroutine of Algorithm 4 to compute period r of function

$$f(x) \equiv a^x modulo N$$

`/* By Euler's theorem, period` r `divides Euler's totient function` $\Phi(N) = (p-1)(q-1)$ `which is the order of the group of units` $\mathbb{Z}_N^\times$ `and is the smallest integer` r `for which` $f(x+r) = f(x)$. `*/`

6 **if** r *is odd* **then**
7 $\quad$ Return to step 1
8 **if** $a^{r/2} = -1$ **then**
9 $\quad$ Return to step 1
10 **else**
11 $gcd(a^{r/2} + 1, N)$ and $gcd(a^{r/2} - 1, N)$ are both non-trivial factors of N so algorithm terminates successfully.

which when applied to a qubit in state $|0\rangle$ results in an equal probability of measuring (in the computational basis) either of the eigenstates $|0\rangle$ or $|1\rangle$. The Hadamard matrix is

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{2.34}$$

In Step 4 of Algorithm 4 the final state can be reordered as

$$\frac{1}{Q} \sum_{z=0}^{N-1} \sum_{y=0}^{Q-1} \sum_{x \in \{0,\ldots,Q-1\}, f(x)=z} \omega^{xy} |y, z\rangle, \tag{2.35}$$

which is a superposition of between Q and Q^2 states, as there are $\approx Q/r$ distinct values of $f(x)$. As defined in Step 5, m represents the number of distinct x which share $f(x) = f(x_0)$, and let b index these, so that $x = x_0 + rb$ for $b = [0, \ldots m-1]$. Then the coefficient of the state $\frac{1}{Q}|y, z\rangle$ is

$$\sum_{x \in \{0,\ldots,Q-1\}, f(x)=z} \omega^{xy} = \sum_{b=0}^{m-1} \omega^{y(x_0+rb)} = \omega^{x)y} \sum_{b=0}^{m-1} \omega^{yrb}, \tag{2.36}$$

which is greatest when ω^{yrb} are closely aligned to one another, which given b is integer valued, means that ω^{yr} must lie on (or extremely close to) the positive real axis. Because $\omega^{yr} = e^{2\pi yri/Q}$, for ω^{yr} to be positive and real $\frac{2\pi yri}{Q}$ should be an integer multiple of $2\pi i$, hence why $\frac{yr}{Q}$ will be close to an integer c.

2.3.6 Variational quantum algorithms

Computing in the noisy intermediate-scale quantum (NISQ) era [Pre18] will be challenging due to a number of constraints, which include qubit errors, small qubit counts, short decoherence times, qubit idling errors, and more. A leading family of algorithms to cope with these difficulties while still achieving quantum advantage is the class of variational quantum algorithms (VQA), which can be interpreted as a quantum analogue of machine learning.

In VQA, one constructs a parametrized ansatz $|\psi(\theta)\rangle$, where $|\psi(\theta)\rangle$ is determined by the gate sequence of the quantum circuit, and can be expressed generally as a sequence of unitaries

$$|\psi(\theta)\rangle = U_M(\theta_M) \dots U_1(\theta_1) |\psi_0\rangle . \tag{2.37}$$

The parameter vector θ is usually influenced by the choice of ansatz. Sometimes the ansatz is chosen to fit the problem at hand (problem-inspired) and sometimes it is a generic ansatz that can fit to any problem (problem-agnostic). In order to evaluate the performance of the algorithm for a set of parameters θ, a cost function $C(\theta)$ should be defined. Similarly to classical machine learning, this cost function should satisfy certain properties. The cost should be a scalar, and its minimum should correspond to the optimal solution. It is desirable that lower cost scores imply better solutions across the entire solution space (this is good for approximate optimizations). It should also not be possible to efficiently classically simulate $C(\theta)$, otherwise the algorithm would not be a good candidate for quantum advantage. One common choice of cost function is the expectation of the final state $|\psi\rangle$ with respect to the problem Hamiltonian

$$C(\theta) = \langle\psi(\theta)| H_P |\psi(\theta)\rangle . \tag{2.38}$$

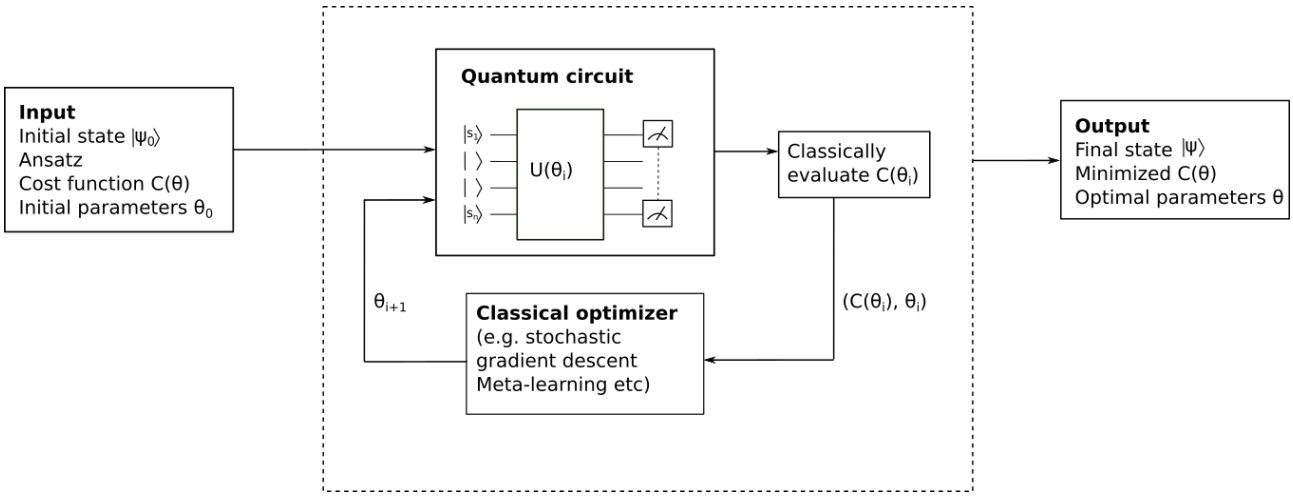

Figure 2.8: A schematic of a variational quantum algorithm. One inputs an initial state (or more generally, a training set of quantum data), as well as an ansatz, cost function for minimization, and initial parameters. Inside the dashed box one iteratively runs the quantum algorithm with the latest set of parameters θ_i, calculates $C(\theta_i)$, and uses the classical outer loop to optimize θ over many calls to the quantum circuit. Once θ converges with an acceptably low $C(\theta)$, one outputs the optimized parameters and cost, allowing the practitioner to then apply the optimized circuit to solve computational problems.

The expectation is a scalar, and has a natural interpretation as the mean eigenenergy of $|\psi(\theta)\rangle$ when measured with respect to H_P.

VQA utilizes a classical outer loop of machine learning in order to optimize $|\psi(\theta)\rangle$ with respect to the parameter vector θ by minimising $C(\theta)$, as illustrated in Fig. 2.8. Similarly to with classical machine learning, one would train the VQA on a training set, and then use the final trained set of parameters to execute the algorithm on unseen problems. Methods for training VQA fall into two buckets: those which utilize gradient descent, such as stochastic gradient descent [KB14] and imaginary time evolution [MJE+19], and those which do not rely on gradient descent, such as meta-learning [WSW+21], simultaneous perturbation stochastic approximation [Spa98] and Bayesian optimization [Moč75, Moc12].

A major obstacle for VQA in achieving quantum advantage is that of barren plateaus. The problem is that the partial derivatives of the cost function parameter landscape exponentially vanish with the system size, meaning that the parameter landscape becomes flat [MBS+18]. Consequently any optimization algorithm would require exponentially higher precision and the extra cost would negate any potential quantum advantage. Thus, analysing problems to deduce

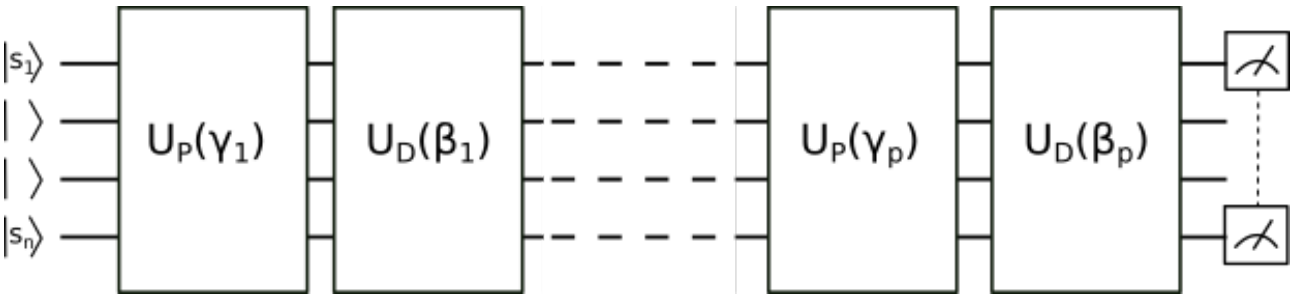

Figure 2.9: QAOA ansatz, where initial state is prepared on the left as $|\psi_0\rangle = \otimes_{i=1}^{n} |s_i\rangle$, then unitaries U_D, U_P are applied p times, before measurement of final state $|\psi\rangle$. The ansatz is parametrized by the $2p$-dimensional vector θ which prescribes the Rabi angles γ_i, β_i.

whether or not they are susceptible to barren plateaus will be an important task for VQA.

2.3.6.1 Quantum approximate optimization algorithm

One member of the VQA family is the quantum approximate optimization algorithm (QAOA) which was initially proposed as an adaptation of the adiabatic quantum algorithm to the gate model of computation [FGG14]. As with AQC, the goal of QAOA is to take input state $|\psi_0\rangle$ and map it to the ground state of some problem Hamiltonian H_P by applying a sequence of unitaries that approximately emulate the adiabatic transition from H_D at $t = 0$ to H_P at $t = T$. QAOA is in fact a coarse Trotterization of the adiabatic quantum algorithm, and the unitaries applied are induced by the driver and problem Hamiltonians H_D and H_P

$$\begin{aligned} U_D(\beta) &= e^{-i\beta H_D}, \\ U_P(\gamma) &= e^{-i\gamma H_P}. \end{aligned} \tag{2.39}$$

Fig. 2.9 demonstrates the repeated application of unitaries U_D, U_P to an initial register of qubits before measurement at the end of the QAOA. The pair of unitaries U_D, U_P are applied p times during QAOA, with higher p corresponding to better approximations to AQC (p is a central parameter to QAOA). For lower p the approximation is more coarse, and this comparison between QAOA and AQC is visualized in Fig. 2.10, with AQC taking a continuous path and QAOA proceeding in discrete jumps.

Algorithm 5: Quantum approximate optimization algorithm

Input : Steps p, driver Hamiltonian H_D, problem Hamiltonian H_P

Output: Candidate solution

1 Prepare initial state $|\psi_0\rangle$

2 Apply ansatz $U_{QAOA}|\psi_0\rangle = U_D(\beta_p)U_P(\gamma_p)\dots U_D(\beta_1)U_P(\gamma_1)|\psi_0\rangle$

3 Measure final state $|\psi\rangle = U_{QAOA}|\psi_0\rangle$ to obtain candidate solution.

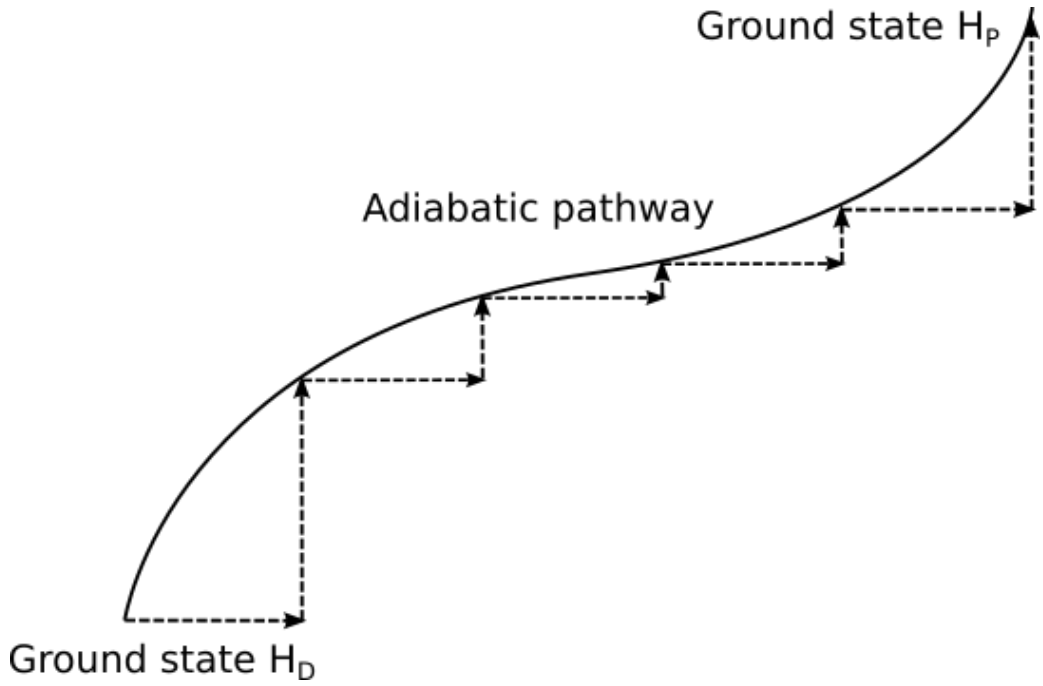

Figure 2.10: A visualization of the smooth adiabatic pathway, compared with the discretized pathway of QAOA which proceeds in jumps. The higher p is, the more closely the QAOA pathway approximates the adiabatic pathway.

In the limit $p \to \infty$, QAOA perfectly approximates the adiabatic quantum algorithm, though its most interesting application is likely to be in low depth circuits with small p. The reason for this is that in the NISQ regime only low depth algorithms will be practicable (and even afterwards, it is reasonable to expect that low depth circuits will be more efficient and straightforward to implement). Low depth QAOA also presents interesting alternative paths to solutions due to its ability to exploit diabatic transitions [ZWC+20], in contrast to AQC. In the adiabatic limit, there appear to be strong patterns for optimal parameters γ_i, β_i, in which γ_i begin very small and increase with i, and β_i begin large and decrease with i [Cro18]. This seems intuitive as considering the Rabi angles γ_i, β_i as times for which the Hamiltonians H_D, H_P are

applied, this represents the driver Hamiltonian being switched on more in the early stages of the algorithm and turned down gradually, while the problem Hamiltonian is applied very little at the beginning, and is turned up throughout the algorithm. Using such patterns to guide parameter initialization, strong results have been demonstrated for efficient optimization of $\theta = (\gamma_1, \ldots \gamma_p, \beta_1, \ldots, \beta_p)$ [ZWC+20]. For low p, however, due to factors such as diabatic pathways and the coarseness of the approximation, finding patterns that can be used for parameter initialization is more difficult.

2.3.6.2 Quantum mean value problem

Classical simulation of quantum circuits is known to be difficult, and indeed must be difficult for a quantum circuit to be a candidate for quantum advantage. Nevertheless, it is necessary to develop methods that allow greater understanding of the output of quantum circuits numerically. As indicated in Eq. (2.38), the expectation of the state $|\psi\rangle$ with respect to a problem Hamiltonian is frequently used to score VQA, and this usually requires full numerical simulation of the state $|\psi\rangle$. The exponential cost as system size increases make expectation an appealing target for efficiency-improving methods.

Speedups have been achieved for very specific circuits, such as a polynomial time algorithm for evaluation of the two-qubit observable $\langle\psi| Z_a Z_b |\psi\rangle$ where $|\psi\rangle$ is the final state of a $p = 1, 2$ QAOA algorithm with Ising-like cost function on planar two-dimensional graphs [BGM21], but such results are only applicable to constant depth circuits.

Tensor networks are another method for more efficient quantum simulation. In essence, tensor networks create a graphical representation of an ansatz, where nodes represent operators and states, and use what are known as contractions to absorb simply connected nodes into one another. The end result should be a simplified graph for which the linear algebra computations are severely reduced. Tensor network methods, however, are only applicable in locally connected systems, and offer no advantage to the highly entangled ansatze which are expected to stand the best chances at achieving quantum advantage on NISQ devices.

2.3.7 Numerical methods

Due to limited availability and quality of quantum hardware, many of the results presented in this thesis were obtained via numerical simulation in the Python programming language. In particular the following Python libraries were used extensively: Bose-Hubbard simulations were performed using QuSpin [WB17], simulations of low depth QAOA circuits were done using TensorFlow Quantum [BVM+20], and analytical formulae were evaluated using the NumPy library [Oli06]. Data processing was performed using Pandas [McK11], and visualizations were performed primarily using Matplotlib [Hun07], Seaborn [Was21], and also Pandas on occasion. The QuTip library [JNN12] was used in auxiliary exploratory research, and the D-Wave software development kit was used as an interface to manage and submit problems to the D-Wave quantum chip.

Algorithm 4: Quantum period finding algorithm

Input : N

Output: Period r of the function $f(x) = a^x \, modulo \, N$

Identify $Q = 2^q$ such that $N^2 < Q < 2N^2$ ▷ Implying $\frac{Q}{r} > N$

Initialise $|\psi_0\rangle$ to be the equal superposition over feasible states $\frac{1}{\sqrt{Q}} \sum_{i=0}^{Q-1} |i\rangle$

`/* Entangle qubits using quantum` $f(x)$ `function */`

Map $f(x)$ to a quantum function U_f, to create the unitary $U_f \, |x, 0^q\rangle = |x, f(x\rangle$ so that

$$U_f \frac{1}{\sqrt{Q}} \sum_{x=0}^{Q-1} |x, 0^q\rangle = \frac{1}{\sqrt{Q}} \sum_{x=0}^{Q-1} |x, f(x)\rangle$$

Apply the inverse quantum Fourier transform to input qubits. This uses a Q^{th} root of unity $\omega = e^{2\pi i/Q}$ to distribute amplitude of $|x\rangle$ states among $|y\rangle$ states equally. $U_{QFT}(|x\rangle) = \frac{1}{\sqrt{Q}} \sum_{y=0}^{Q-1} \omega^{xy}$ gives final state

$$\frac{1}{Q} \sum_{x=0}^{Q-1} \sum_{y=0}^{Q-1} \omega_{xy} \, |y, f(x)\rangle$$

Measure the final state, observing y in the input register and some z in the output register.

$$P(|y, z\rangle) = \left| \frac{1}{Q} \sum_{x \in \{0,\ldots,Q-1\}, f(x)=z} \omega^{xy} \right|^2 = \frac{1}{Q^2} \frac{\sin^2(\frac{\pi r m y}{Q})}{\sin^2(\frac{\pi r y}{Q})},$$

where x_0 is the smallest value of x such that $f(x) = z$, and $m - 1 = \lfloor \frac{Q - x_0 - 1}{r} \rfloor$ This is highest when $\frac{yr}{Q}$ is close to an integer which we call c

$\frac{y}{Q} \approx \frac{c}{r}$, so find integers d, s to approximate this value such that $s < N$ and

$$\left| \frac{y}{Q} - \frac{d}{s} \right| < \frac{1}{2Q}$$

Then s is likely to be the period r, if not a factor of r

if $f(x) = f(x + s)$ **then**

  Period $r = s$ has been found. Algorithm terminates successfully

else

  Try other candidates d, s to obtain more candidates for r. If this does not work, repeat the algorithm.

Chapter 3

Shortest Vector Problem in the Quantum Realm

In this chapter I lay the groundwork for the combinatorial and spatial parts of the mapping from SVP into quantum algorithms. This material is common to the algorithms of Chapters 5 6. The content of this chapter pertains mainly to the algorithm setup and the interpretation of measurements as N-dimensional vectors, and so does not directly require any quantum understanding. Theorem 3.1 and its proof were presented in [JCLM21].

The first task in designing a suite of algorithms to search for short lattice vectors is to formulate SVP in terms of quantum operators on a set of qubits (or on some alternative physical system, such as in Chapter 4). To do this, I use the description of a general lattice vector, and write it in the form of a quadratic unconstrained binary optimization (QUBO) problem. This enables the definition of a Hamiltonian, the eigenstates of which are to be interpreted as lattice vectors, and whose corresponding eigenenergies are the vector lengths.

The next step is to ascertain spatial bounds as while a mathematical lattice is infinite, for the purposes of search algorithms one must narrow down the search to a finite space of linear combinations of lattice vectors.

3.1 QUBO Representation of SVP

For any given basis $\mathbf{B}$ of a lattice $\mathcal{L}$, one can write every lattice vector $\mathbf{v}$ as a linear combination of the basis vectors

$$\mathbf{v} = x_1\mathbf{b}_1 + \ldots + x_N\mathbf{b}_N, \tag{3.1}$$

or, written in matrix format,

$$\mathbf{v} = \mathbf{x}\mathbf{B}, \tag{3.2}$$

where $\mathbf{x} \in \mathbb{Z}^N$, and $\mathbf{B}$ is a row basis, meaning the rows of $\mathbf{B}$ are the basis vectors of $\mathcal{L}$. Expressing $\mathbf{v}$ in full, one can write

$$\mathbf{v} = \begin{bmatrix} x_1\mathbf{b_1}_1 + \ldots x_N\mathbf{b_N}_1 \\ x_1\mathbf{b_1}_2 + \ldots x_N\mathbf{b_N}_2 \\ \vdots \\ x_1\mathbf{b_1}_N + \ldots x_N\mathbf{b_N}_N \end{bmatrix}, \tag{3.3}$$

and so one can write the Euclidean norm of $\mathbf{v}$ as a function of the coefficients x_i and the basis vector coordinates $\mathbf{b_i}_j$

$$\|\mathbf{v}\| = \sqrt{(x_1\mathbf{b_1}_1 + \ldots x_N\mathbf{b_N}_1)^2 + \ldots + (x_1\mathbf{b_1}_N + \ldots x_N\mathbf{b_N}_N)^2}. \tag{3.4}$$

Expanding the terms inside of the square root,

$$\|\mathbf{v}\| = \sqrt{\sum_{ij}^{N} x_i x_j \sum_{m=1}^{N} \mathbf{b_i}_m\mathbf{b_j}_m} = \sqrt{\sum_{ij}^{N} x_i x_j \mathbf{b_i}\mathbf{b_j}}, \tag{3.5}$$

which can be condensed by observing that the scalar $\mathbf{b_i}\mathbf{b_j}$ is the $(i,j)^{th}$ coefficient of the Gram matrix of Eq (2.7)

$$\|\mathbf{v}\| = \sqrt{\sum_{ij}^{N} x_i x_j \mathbf{G_{ij}}}. \tag{3.6}$$

For the algorithms considered in this thesis, the values

$$\|\mathbf{v}\|^2 = \sum_{ij}^{N} x_i x_j \mathbf{G_{ij}}, \tag{3.7}$$

are the eigenenergies of the problem Hamiltonians H_P, where the x_i are the variables subject to quantum optimization. In order to optimize the x_i in a quantum setting, the x_i must be mapped from integers (in the classical regime) to qudits (in the quantum regime), which remain in a superposition of different integer values until measurement, when they will each output a classical integer. The set of N integers x_i describe a vector $\mathbf{v} \in \mathcal{L}(\mathbf{B})$, which corresponds to an eigenstate of the problem Hamiltonian whose eigenenergy is the Euclidean norm squared of the lattice vector, as given by Eq. (3.7).

The form of Eq. (3.7) is not yet a QUBO problem, which is necessary for the algorithms of Chapters 5, 6, as the x_i are integral and not binary variables. To complete the mapping to the QUBO setting, it is just necessary to specify how the x_i should be constructed from a set of binary variables, and the spatial analysis depends on these constructions.

3.2 Qudit definitions in the gate model

Having defined the mapping from SVP to a classical problem Hamiltonian in terms of the set of N integers x_i to optimize, the next step is to define limits on the search space. The mathematical lattice extends to infinity, yet algorithms for practical implementation can only consider a finite range for each x_i. In this section I define the qubit-to-qudit mappings used for later algorithms, and derive analytical bounds on the number of qubits required to ensure that the exact shortest vector of a given integer lattice resides within the solution space considered. Put differently, what is the minimum range $[a, b]$ in which to look for the x_i, such that one can ensure that there exists a vector $\mathbf{x} = (x_1, \ldots, x_N)$ with $a \leq x_i \leq b$ and $\|\mathbf{xB}\| = \lambda_1(\mathcal{L})$? The qubit-to-qudit mappings are defined in an entirely classical setting as one simply needs to interpret the output of a measurement of qubits (a binary string) as an integer. The quantum qudit operators that correspond to these mappings are given in later chapters.

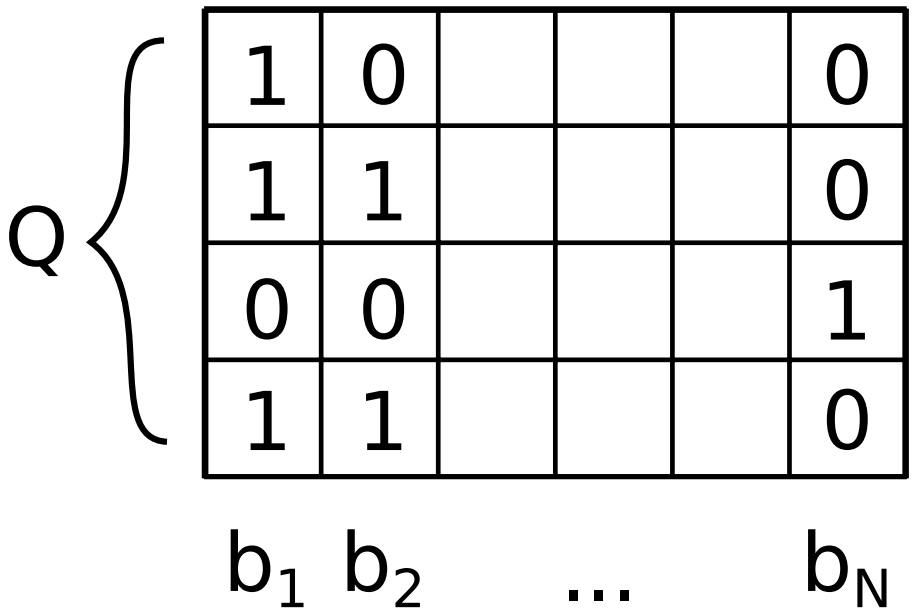

Figure 3.1: Diagram of a two-dimensional array of qubits, which can be interpreted as a one-dimensional array of integers by using a number of different qubit-to-qudit mappings. Each column of qubits is a binary string, which can be interpreted as an integer. The vector of integers is matrix multiplied with the lattice basis to return a lattice vector, as per Eq. (3.2).

3.2.1 Hamming qudits

The Hamming-encoded qudits are a simple and space-inefficient mapping. If one considers a binary string of length j, consisting of 1's and 0's, the Hamming weight of the string is simply the number of 1's. This number can take any value between 0 and j inclusive. Consequently j qubits, using this Hamming encoding, can represent $j + 1$ integers.

It is important to consider that in the Hamming case, the $j + 1$ integers range from 0 to j, but it is easy to shift this range. The shift in range is necessary, as when searching for short vectors it is most appropriate to search in a symmetric interval about the origin.

The Hamming-encoded qudit is spatially inefficient, as there is a considerable amount redundancy: the strings '1100' and '0011' both yield the same integer (2). But an important aspect of the Hamming-encoded qudit is that each qubit is as significant as the next, which, as explained in Chapter 5, can be extremely useful due to the realities of dealing with quantum hardware.

3.2.2 Binary qudits

For binary-encoded qudits, one interprets the binary string as a binary number in the canonical way. This is the most spatially efficient mapping, and an array of j qudits can return 2^j distinct integers, and each unique bit string represents a different integer.

In order to realise this efficiency, each bit has a different 'significance'. The consequence of this is that the qubits have exponentially increasing prefactors (the most significant bits have the largest prefactors), and this can make some aspects of implementation troublesome, as investigated in Chapter 5.

Fig 3.1 shows an example measurement of a $N \times 4$ array of qubits, measured in the $\{0, 1\}$ basis (in the Z eigenbasis this is $\{-1, 1\}$ but it is easy to map to $\{0, 1\}$ via the operator $\frac{1}{2}(\mathbb{1} - \mathbb{Z})$). Fig 3.1 illustrates how the two-dimensional array of qubits can be read as a one-dimensional array of integers upon measurement, with each column of qubits making up a single qudit. In the Hamming-encoded setting without any shift in range, where one takes the measured value of the qudit to be the Hamming weight of the binary string, the qudit of the first column of the qubit array in Fig 3.1 has a value upon measurement of 3 (Hamming weight of 1101), and before measurement, is in a superposition of values between 0 and 4 inclusive.

In the binary-encoded setting without any shift in range, and taking the top row to be the most significant bits and the bottom row to be the least significant bits, the qudit takes the value of 13 (1101 in binary), and prior to measurement, would have been in a superposition of values between 0 and 15.

3.3 Spatial Requirements for SVP representations

Having defined the mapping from qubits to qudits, it is now established that to search a range of integers of size 2^j, it is necessary to use j qubits per qudit in the binary encoding, and $2^j - 1$ qubits per qudit in the Hamming encoding. In order to ascertain full spatial requirements in the Ising lattice, it is necessary to evaluate what range of integers should be searched in order

to guarantee that the shortest vector is a feasible solution within the search space.

I prove this by using the HNF of a lattice, and restrict the proof to the case of lattices with optimal HNF. This is a reasonable assumption because lattices with optimal HNF are both common, with $\simeq 40\%$ of random integer lattices having optimal HNF [RPS11], and also cryptographically desirable [PSW08], as lattices with optimal HNF can be communicated with $O(N)$ space, versus other bases, where $O(N^2)$ space is needed to communicate the entire $N \times N$ matrix, despite describing the exact same lattice. Furthermore, it is easy to obtain the HNF of a lattice, taking time polynomial in N [KB79].

Theorem 3.1 (Spatial bounds)**.** *For an N-dimensional lattice $\mathcal{L}(\mathbf{B})$ with covolume $det(\mathbf{B}) = D$ with optimal HNF, at most $\frac{3N}{2}\log N + N + \log D$ qubits are required in order to ensure that a quantum algorithm to solve the exact shortest vector problem can succeed.*

Proof. The central equation to be optimized is Eq (3.2), where the x_i of $\mathbf{x}$ must be bounded in $-2^k \leq x_i \leq 2^k$ for $1 \leq i \leq N$. An optimal HNF lattice basis is written

$$\begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_N \end{bmatrix} = \begin{bmatrix} 1 & & & b_1 \\ & \ddots & & \vdots \\ & & 1 & b_{N-1} \\ & & & D \end{bmatrix}. \tag{3.8}$$

Because optimal HNF mandates that only one pivot is greater than 1, and in this format the determinant of the matrix (which is the covolume of $\mathcal{L}$) is the product of the diagonal elements, the singular nontrivial pivot takes value D.

The lattice vector can therefore be written explicitly as

$$\begin{bmatrix} x_1, & \ldots, & x_N \end{bmatrix} \begin{bmatrix} 1 & & & b_1 \\ & \ddots & & \vdots \\ & & 1 & b_{N-1} \\ & & & D \end{bmatrix} = \begin{bmatrix} x_1, & \ldots & x_{N-1}, & (x_1 b_1 + \ldots + x_N D) \end{bmatrix}. \tag{3.9}$$

Minkowski's theorem [Min16] provides an upper bound on the length of the shortest vector of a lattice as a function solely of the covolume of the lattice and its dimension

$$\lambda_1(\mathcal{L}) \leq \sqrt{N} D^{1/N}. \tag{3.10}$$

Geometrically, Eq (3.10) prescribes a spherical region around the origin within which the shortest vector must lie. A more relaxed interpretation would be to observe that each coordinate in isolation must satisfy the bound. This means that instead of searching within the N-sphere of radius $\sqrt{N} D^{1/N}$, the region is expanded to the N-cube of side length $2\sqrt{N} D^{1/N}$ centred at the origin. This region is strictly larger than the N-sphere and contains it entirely, so is a weaker condition than Minkowski's bound, but allows for a straightforward coordinate-wise bound: each element of $\mathbf{v}$ must have absolute value less than $\sqrt{N} D^{1/N}$

$$|x_1|, \ldots, |x_{N-1}|, |x_1 b_1 + \ldots + x_N D| \leq \sqrt{N} D^{1/N}. \tag{3.11}$$

Eq (3.11) immediately bounds $x_1, \ldots x_{N-1}$. Concerning the final coordinate x_N, however, it follows that

$$|x_N D| \leq \sqrt{N} D^{1/N} + |x_1 b_1 + \ldots + x_{N-1} b_{N-1}|. \tag{3.12}$$

Next, apply the triangle inequality to isolate the individual terms inside the modulus

$$|x_N D| \leq |x_1 b_1| + \ldots + |x_{N-1} b_{N-1}| + \sqrt{N} D^{1/N}. \tag{3.13}$$

From the properties of HNF bases, the b_i are nonnegative and reduced modulo D, meaning D can be used to upper bound the b_i

$$|x_N D| \leq |x_1 D| + \ldots + |x_{N-1} D| + |\sqrt{N} D^{1/N}|, \tag{3.14}$$

and substituting in the bounds on $x_i, 1 \leq i \leq N-1$ from Eq (3.11) gives an expression in just

the lattice dimension N and its covolume D,

$$|x_N D| \leq \left|\sqrt{N} D^{1/N} D\right| + ... + \left|\sqrt{N} D^{1/N} D\right| + \left|\sqrt{N} D^{1/N}\right|. \tag{3.15}$$

Collecting terms gives a an expression in x_N, D, N

$$|x_N D| \leq N^{3/2} D^{1+1/N}, \tag{3.16}$$

which prescribes an interval over which to search for x_N of size $2N^{3/2}D^{1/N}$. This bound is larger than the bound for the other x_i, whose interval requires only $2N^{1/2}D$, so taking the larger quantity, this range of integers can be represented by $\log_2(2N^{3/2}D^{1/N})$ qubits using a binary-encoded qudits. Consequently, at most $N \log_2(2N^{3/2}D^{1/N})$ qubits are required to run a quantum algorithm to solve exact SVP where variables are optimized in an Ising lattice. This simplifies to an asymptotic spatial requirement of $O(N \log N + \log D)$ qubits. □

Corollary 3.1. *Using Hamming-encoded qudits, an exact SVP algorithm on an N-dimensional lattice $\mathcal{L}$ with covolume $D = \det(\mathcal{L})$ and optimal HNF, requires at most $2N^{5/2}D^{1/N}$ qubits.*

Proof. Most of the proof is identical to that of Theorem 3.1 right the way down to Eq (3.16). The difference in size requirements derives from the fact that after obtaining $x_N \leq N^{3/2}D^{1/N}$, no logarithm is taken. After multiplying by the number of qudits required, N, one arrives at the bound $N^{5/2}D^{1/N}$. □

Having derived and presented the bounds of Theorem 3.1 and Corollary 3.1, one should take a moment to consider how these results should be incorporated for practical use in algorithms:

1. These bounds could likely be tightened in a number of places, though whether this has an effect on the asymptotic scaling is another question. For example,
   - The relaxation to use an N-cube instead of the N-sphere from Minkowski's bound is extremely loose for large N,
   - While x_N requires a search interval which is a factor of N larger than the other x_i, for simplicity we take the interval of x_N and present the results of using this bound for all the x_i which gives a rectangular Ising lattice.
2. We have provided an upper bound on spatial requirements for quantum devices to identify the shortest vector of a lattice. While these upper bounds are useful from an attacker's perspective, it would be insightful for future work to derive lower bounds which would help from a defensive perspective, in that one could understand the minimum number of qubits an attacker would need in order to attempt to solve an SVP instance.
3. While the spatial results presented here guarantee the existence of the exact shortest vector within the feasible solution space, the quantum algorithm remains inherently probabilistic, and one may find that picking a smaller interval for qudits serves to increase the probability of finding short vectors sufficiently that it is worth the tradeoff to lose the guarantee of the existence of the shortest vector in the feasible solution space. This reasoning is analogous to the rigorously analysed enumeration with extreme pruning whereby randomized but smaller search areas can provide significantly improved time to solution [GNR10].
4. Due to variants of SVP such as SVP_α where one aims to find some 'approximately' short vector (as opposed to the exact shortest vector), it may not be necessary at all to ensure that the shortest vector resides within the feasible solution space.

Nevertheless, the spatial bounds provided give an indication that quantum algorithms to solve

SVP of the flavour presented in this thesis should not be limited by space requirements, and offer a baseline from which improvements and alterations can be made.

Chapter 4

Bose-Hubbard search for short vectors

4.1 Introduction

In this chapter I present the findings of [JGLM20], where the properties of an ultra-cooled bosonic gas are programmed to help identify short vectors in a lattice $\mathcal{L}(\mathbf{B})$ via an AQC-style algorithm where in contrast to AQC, fast evolutions are employed. The chapter is structured to be self-contained, though references other chapters where appropriate.

The most widely recognised framework for quantum computing is the circuit model, likely because it is strongly analogous to the classical architecture of computing: instead of using building blocks of bits, one uses qubits which have the same properties as their classical counterparts upon measurement, but until that point the qubits remain in some quantum state that is a superposition of the possible outcomes. However there are a number of other models for quantum computation which do not possess a close relative in the classical realm, and this is particularly true of some forms of continuous time quantum computing.

In Chapter 6 a circuit model is described, which utilizes the problem Hamiltonian of Bin algorithm of Chapter 5, which despite being a continuous time quantum algorithm, still uses a qubit-based architecture and a series of two-qubit gate-like operators. The physical system of this chapter does not use a qubit architecture, meaning it exhibits a different set of advantages

and disadvantages.

In the quantum algorithms for SVP of this chapter, the physical system used is based on a Bose-Hubbard Hamiltonian which describes the energy of a configuration of bosonic particles in a one-dimensional periodic potential landscape with a number of minima, which are called sites. By tuning the depth of the potential wells (i.e. how pronounced the sites are), one can transition between a superfluid and a Mott insulator, where no quantum tunneling takes place. The number of bosons located at site i in the Mott insulator is the output of a qudit.

4.2 Quantum Algorithm

This section defines the driver Hamiltonian, the SVP problem Hamiltonian, and the basic algorithm, as well as the combinatoric considerations necessary to ensure the right number of bosons are used.

4.2.1 Hamiltonian definitions

The quantum algorithm follows the usual AQC method of interpolating between an initial driver Hamiltonian H_D and a final problem Hamiltonian H_P according to a continuous schedule. The choice of Hamiltonian evolution schedule has been subject to scrutiny, but for simplicity, in this chapter a linear sweep is used

$$H_t = \Big(1 - \frac{t}{T}\Big) H_D + \Big(\frac{t}{T}\Big) H_P, \tag{4.1}$$

where T is the total length of the sweep and $0 \leq t \leq T$. The driver Hamiltonian, responsible for exploration of the periodic landscape is determined by the annihilation (a_i) and creation

$(a_i^\dagger)$ operators

$$\begin{aligned} a_i \left|n_i\right\rangle &= \sqrt{n_i}\left|n_i - 1\right\rangle, \\ a_i^\dagger \left|n_i\right\rangle &= \sqrt{n_i + 1}\left|n_i + 1\right\rangle, \\ a_i^\dagger a_i &= \hat{n}_i, \end{aligned} \tag{4.2}$$

which increment or decrement the particle number at site i. The driver Hamiltonian, is then

$$H_D = -\sum_i (a_i a_{i+1}^\dagger + a_{i+1} a_i^\dagger) \tag{4.3}$$

where the first term describes a particle tunneling from site i to site $i+1$, and the second term describes tunneling in the opposite direction. This means that a particle at any site can tunnel to the site to its left or right while preserving the total particle number, as each term of H_D consists of a convolution of a creation operator with an annihilation operator or vice versa.

A thorough discussion of the mapping from lattice vectors into a classical Hamiltonian can be found in Chapter 3. Eq (3.7) must then be adapted from a classical Hamiltonian to a quantum Hamiltonian, for which integers are replaced by qudits. For the Bose-Hubbard SVP algorithm, H_P is written

$$H_P = \sum_{ij}^{N} \hat{n}_i \hat{n}_j \mathbf{G}_{ij}, \tag{4.4}$$

where $\hat{n}_i$ is the number of bosons recorded at site i, when the system is in Mott insulator form at completion of the algorithm at time $t = T$. This is convenient as one does not have to artificially construct the qudit from an array of qubits, as is the case for the algorithms of Chapters 5, 6, which incurs a cost of many two-qubit ZZ gates which are a significant source of error in the NISQ era.

While the Hamiltonian of Eq (4.4) defines the mapping of lattice vectors into the quantum realm, it does not adequately describe how this might be realised in a Bose-Hubbard Hamiltonian. To do this, one can decompose the problem Hamiltonian into an onsite chemical

Hamiltonian H_S, and an interaction energy Hamiltonian H_I. The interaction Hamiltonian

$$H_I = \sum_i v_{ii}\hat{n}_i(\hat{n}_i - 1) + \sum_{i \neq j} v_{ij}\hat{n}_i\hat{n}_j, \tag{4.5}$$

determines the strength of interaction between each particle and every other particle in the potential landscape. It is written as two summations because for particles at the same site, each of the $\hat{n}_i$ particles interacts with the $\hat{n}_i - 1$ other particles at site i, but for particles at distinct sites, each of the $\hat{n}_i$ particles at site i interact with each of the $\hat{n}_j$ particles at site j. It should be noted that for the SVP application, it is necessary to consider long range interactions between particles at different sites $\sum_{i \neq j} v_{ij}\hat{n}_i\hat{n}_j$. This represents an experimental challenge, but that is a consideration outside the scope of this work. The onsite Hamiltonian

$$H_S = \sum_i \mu_i \hat{n}_i \tag{4.6}$$

assigns a fixed energy μ_i to each particle at site i.

Next, to reconcile H_I, H_S, and H_P, let the onsite Hamiltonian compensate for the missing factor of $\hat{n}_i$ in the interaction Hamiltonian between particles at the same site, by setting $\mu_i = v_{ii}$. Then

$$H_P = H_I + H_S = \sum_{ij} \tilde{v}_{ij}\hat{n}_i\hat{n}_j, \tag{4.7}$$

and to complete the formulation of H_P as an SVP Hamiltonian, it just remains to set

$$\tilde{v}_{ij} = \mathbf{G}_{ij}. \tag{4.8}$$

Because each site can host any nonnegative number of particles, the number of Fock states in the Hilbert space is

$$D = \frac{(K + N - 1)!}{K!(N - 1)!}, \tag{4.9}$$

where K is the number of bosonic particles, and N is the number of sites in the potential landscape. The energy of a single particle at site i should then be equivalent to the length of

basis vector $\mathbf{b}_i$.

4.2.2 Illustrative example

The following example demonstrates the implementation of the algorithm as defined up to this point. The physical system consists of two particles in a periodic landscape of two sites. The Hilbert space has dimension three and the Fock states can be enumerated

$$|02\rangle, |11\rangle, |20\rangle. \tag{4.10}$$

It is important to remember that states $|0,0\rangle, |0,1\rangle, |1,0\rangle, |2,1\rangle, |1,2\rangle, |2,2\rangle$ are not within the search space because the system has two particles and in this theoretical framework particles cannot be created or annihilated. Consider a good basis $\mathbf{B}_G$, and a bad basis $\mathbf{B}_B$ for the same lattice. Using, Definition 2.2 derive the Gram matrix $\mathbf{G}_B$ from $\mathbf{B}_B$. The Gram matrix determines the coefficients of the problem Hamiltonian H_P

$$\mathbf{B}_G = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix} \quad \mathbf{B}_B = \begin{pmatrix} 1 & 2 \\ 0 & -2 \end{pmatrix} \quad \mathbf{G}_B = \begin{pmatrix} 5 & -4 \\ -4 & 4 \end{pmatrix}. \tag{4.11}$$

Calculating coefficients for H_D using Eq (4.3), and calculating H_P from Eq (4.4) yields matrices for the driver and problem Hamiltonians of

$$H_D = \begin{pmatrix} 0 & -\sqrt{2} & 0 \\ -\sqrt{2} & 0 & -\sqrt{2} \\ 0 & -\sqrt{2} & 0 \end{pmatrix}, \quad H_P = \begin{pmatrix} 16 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 20 \end{pmatrix}. \tag{4.12}$$

Combining H_D and H_P to obtain the time-dependent Hamiltonian matrix, using the linear time evolution as per Eq. (4.1),

$$H(t) = \begin{pmatrix} 16\frac{t}{T} & -\sqrt{2}\left(1-\frac{t}{T}\right) & 0 \\ -\sqrt{2}\left(1-\frac{t}{T}\right) & 1\frac{t}{T} & -\sqrt{2}\left(1-\frac{t}{T}\right) \\ 0 & -\sqrt{2}\left(1-\frac{t}{T}\right) & 20\frac{t}{T} \end{pmatrix}. \tag{4.13}$$

The ground state at $t = 0$ is thus the ground state of H_D

$$|\psi_0\rangle = \frac{1}{2}(|02\rangle + \sqrt{2}\,|11\rangle + |20\rangle). \tag{4.14}$$

Initialise the system in state $|\psi_0\rangle$ and let it evolve. The probabilities of measuring the system to be in each of the Fock states of Eq (4.10) during the evolution is shown in Fig 4.1.

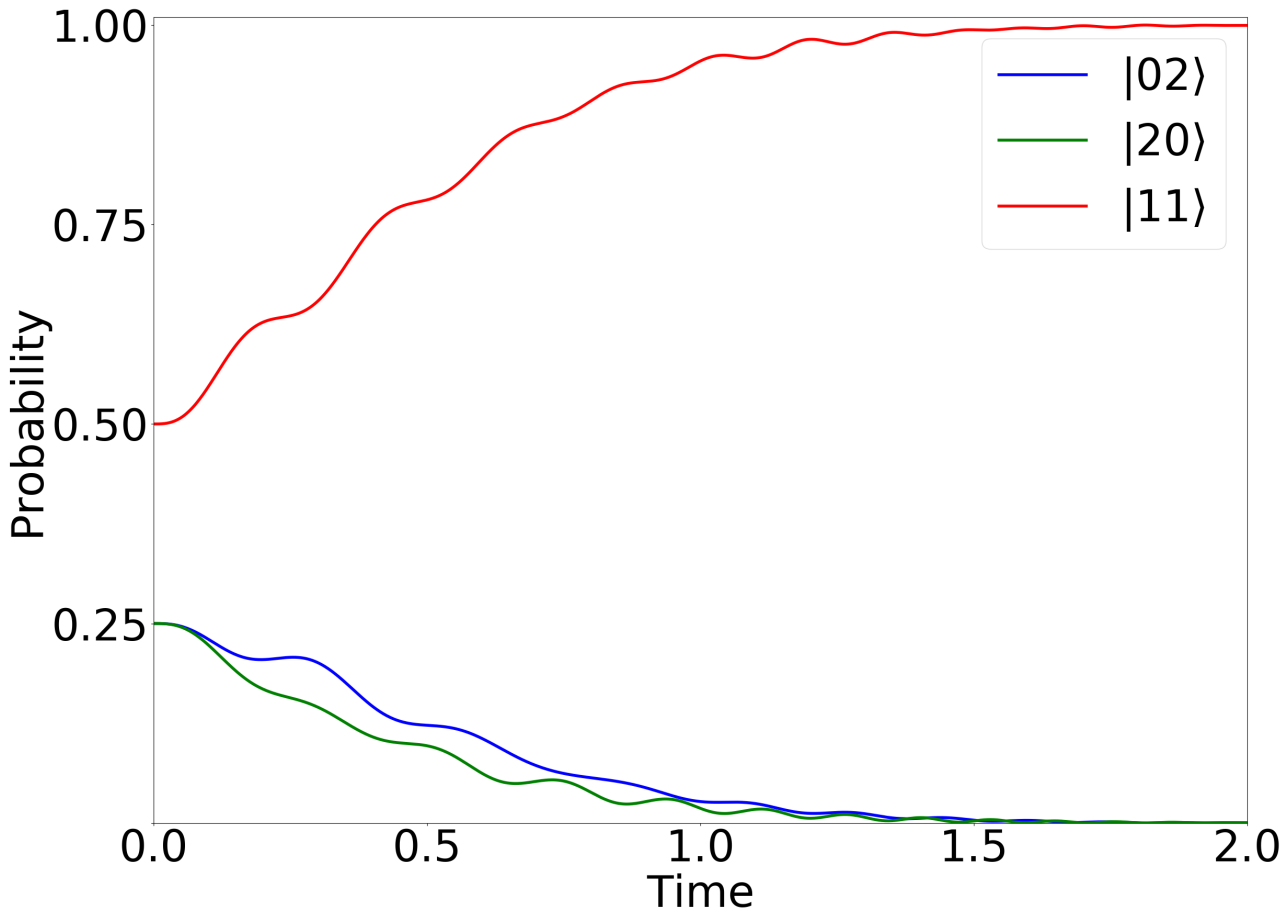

Figure 4.1: Each line represents the probability of finding the system in the corresponding Fock state as outlined in Eq (4.10) as time progresses in the evolution.

The Hamiltonian ground state for $t = T = 2$ is

$$|\psi_T\rangle = |11\rangle\,, \tag{4.15}$$

which is easy to see because summing the two rows of $\mathbf{B}_B$ gives a vector of unit length.

$$\begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 0 & -2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \end{pmatrix}, \tag{4.16}$$

which is of course the shortest vector in the lattice. This can also be seen by looking at $\mathbf{B}_G$, where $(0, 1)$ is the basis vector given by the top row of $\mathbf{B}_G$. This algorithm, however, only allows for Fock states with a fixed particle sum of 2. If the coefficient vector of the shortest vector $\mathbf{x}_{\text{min}}$ has a coefficient sum different from 2, for example $x_1 + x_2 = 3$, then the shortest vector does not present in the solution set. The following sections detail how to craft an algorithm that overcomes this constraint by performing multiple runs, and analyzes the number of bosons necessary to run such algorithms successfully.

4.2.3 Enabling negative coefficients

At present, running the algorithm with K particles and measuring the final state to ascertain the number of particles by measuring $\hat{n}_i$ at each site, one would obtain a coefficient vector $\mathbf{x} = (x_1, \ldots, x_N)$ where the $x_i > 0$ for all i. This is quite limiting, however, as one has no guarantee that a short vector can be found via such a nonnegative combination. Similarly to how a shift in range is applied to obtain the qudit operators in Chapters 5 and 6, an offset must be applied so that the x_i are returned from a roughly symmetric interval about the origin.

The solution proposed to is increase the number of particles in the system by Nm (on average, m particles for each site), and then apply a change of variables in order to use these particles as an offset. This enables negative coefficients x_i, because the x_i can now be as low as $-m$, which is recorded if no particles are observed at site i. Observing m particles at site i results

in $x_i = 0$ and so on. Write the new particle number

$$n_i = \hat{n}_i + m, \tag{4.17}$$

and denote the adjusted problem Hamiltonian H'_P. Having applied this change of variables to Eq (4.4), the new problem Hamiltonian H'_P becomes

$$H'_P = \sum_{ij} \mathbf{G}_{ij}(\hat{n}_i + m)(\hat{n}_j + m) \tag{4.18}$$

which can be expressed in terms of the original problem Hamiltonian of Eq (4.4) as

$$H'_P = H_P + 2m \sum_i \hat{n}_i \Big(\sum_j \mathbf{G}_{ij} \Big) + m^2 \sum_{ij} \mathbf{G}_{ij}. \tag{4.19}$$

In order to minimize H_P and not H'_P, however, the onsite energy must be reduced by a function of the row sum $\sum_j \mathbf{G}_{ij}$ for each site i. The final term of Eq (4.19) is constant, meaning that it affects the eigenspectrum of H_P only by a constant shift, therefore it does not alter the ordering of the lattice vectors represented and so the optimal configuration of bosons in the periodic landscape is not affected by the $m^2 \sum_{ij} \mathbf{G}_{ij}$.

To ensure that the shortest vector lies in the solution set (and so occupies the first excited state E_1 of H_P) the offset m must be larger than the infinity norm of the coefficient vector $\mathbf{x}_{\text{min}}$. That is to say,

$$\begin{aligned} m &\geq \|\mathbf{x}_{\text{min}}\|_\infty, \\ \|\mathbf{x}_{\text{min}}\mathbf{B}\| &= \lambda_1(\mathcal{L}). \end{aligned} \tag{4.20}$$

4.2.4 Multi-run

Running the algorithm as described thus far with $K = Nm$ particles would result in a coefficient vector $\mathbf{x}$ where the $x_i > -m$, and $\sum_{i=1}^{N} = 0$. To find the shortest vector, it is necessary to try different numbers of particles, as the shortest vector can only be found when the algorithm

is run with $K = Nm + r$ particles because $\sum_{i=1}^{N} x_i^{\min} = r$. In this case, many repeats of the algorithm will be required, and in one of them the ground state of the algorithm will yield the shortest vector. As such, one would run the sweep for $K = Nm, Nm+1, Nm+2, \ldots, Nm+c$ particles where c is some heuristic or analytic upper bound

$$c \gtrsim \left| \sum_{i}^{N} x_i^{\min} \right|, \tag{4.21}$$

and let $K_{\max} = Nm + c$. So long as c is chosen correctly, and the sweeps are performed adiabatically, one is guaranteed to find the shortest vector. In this algorithm (called Multi-run), c sweeps are therefore required which can be performed in parallel, and the length of sweeps T_r remains an open question due to the difficulty in analyzing intermediate eigenspectra of adiabatic quantum algorithms. Fig. 4.2 shows $\|\mathbf{x}_{\min}\|_\infty$ as lattice dimension grows for lattices

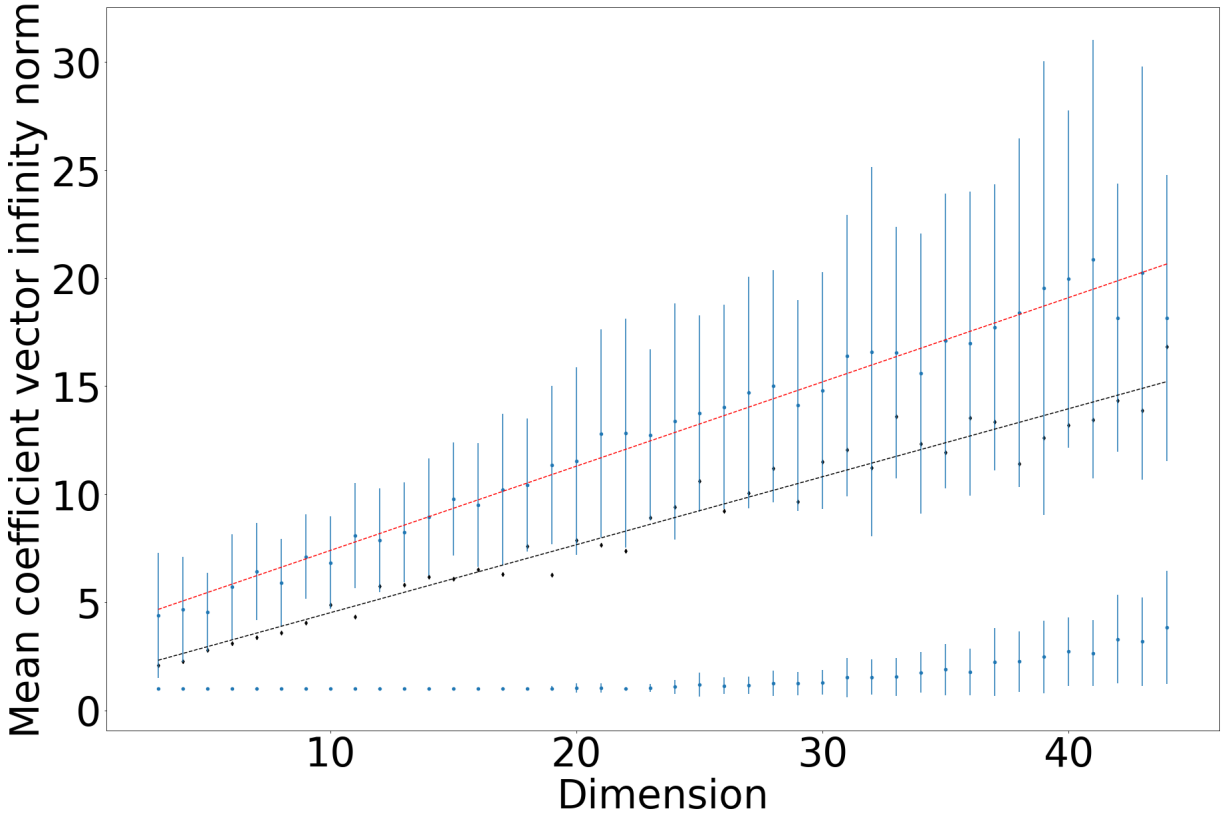

Figure 4.2: Upper distribution (with red linear best fit line) shows average $\|\mathbf{x}_{\min}\|_\infty$ for HNF bases, the black linear best fit shows $\left|\sum_i^N x_i^{\min}\right|$, and the lower distribution shows $\|\mathbf{x}_{\min}\|_\infty$ for LLL-reduced bases.

preprocessed to obtain HNF bases (with red dashed best fit) and LLL-reduced bases (lower

distribution) in dimensions $3 \leq N \leq 45$. Also shown is the value $\left|\sum_i^N x_i^{\min}\right|$ for the coefficient vectors $\mathbf{x}_{\min}$ (with black dashed best fit). One can see that $\left|\sum_i^N x_i^{\min}\right|$ grows linearly with lattice dimension, implying $c = O(N)$ runs would be required to execute Multi-run.

In the next section, Single-run eloquently combines the above $O(N)$ sweeps into a single sweep which brings certain benefits, but at the cost of $\lambda_1(\mathcal{L})$ no longer occupying the ground state.

4.2.5 Single-run

The purpose of the algorithm described in the following is to generalise Multi-run into a single sweep for which all Multi-run solutions are attainable. In Multi-run the coefficient vectors for a two-dimensional lattice $(0, 1), (1, 1)$ would be candidate solutions for separate sweeps because $|\sum x_i|$ are 1 and 2 respectively. In Single-run, however, the aim is for all such potential solutions from the various repetitions of Multi-run to exist within a single solution space.

The proposed solution is to introduce an extra site to the potential landscape, which represents the zero vector, meaning there are now $N + 1$ sites, and let the site of the zero vector be site $N + 1$. The extra site behaves as a particle reservoir and should not directly influence the energy of the system (as adding or subtracting multiples of the zero vector do not affect vector length). Classically, this is equivalent to performing a single sweep of Multi-run on the rank-N lattice $\mathcal{L}(\mathbf{B})$ embedded in $N + 1$ dimensional space, as denoted in Eq (4.23). Let the particle number for Single-run be K_S. If the process is run with K_S total particles, then the set of configurations where no particles are at site $N + 1$ represent the solution set of a one sweep of Multi-run with K_S particles; and the set of configurations where one particle is recorded at site $N + 1$ represent the solution set of a one sweep of Multi-run with $K_S - 1$ particles, and so on. A consequence of this adaptation is that the ground state is not the shortest vector, but is now the zero vector, which is observed when each of the first N sites host m particles (which yields coefficient 0 due to the offset), and all remaining particles reside at site $N + 1$. In this case the eigenstate measured has eigenvalue $\|\mathbf{0}\|^2 = 0$, and so an adiabatic evolution is no longer required. The implications of this are considered in later sections.

The Single-run problem Hamiltonian H'_P can thus be represented

$$H'_P = \sum_{i,j}^{N+1} \mathbf{G}'_{ij}(\hat{n}_i + m)(\hat{n}_j + m), \tag{4.22}$$

and $\mathbf{G}'_{ij}$ is $\mathbf{B}'\mathbf{B}'^T$, where

$$\mathbf{B}' = \left[\begin{array}{c} \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_N \\ \hline \mathbf{0} \end{array}\right]. \tag{4.23}$$

The problem Hamiltonian of Eq (4.22) is used for the remainder of the chapter, including for all numerical simulations.

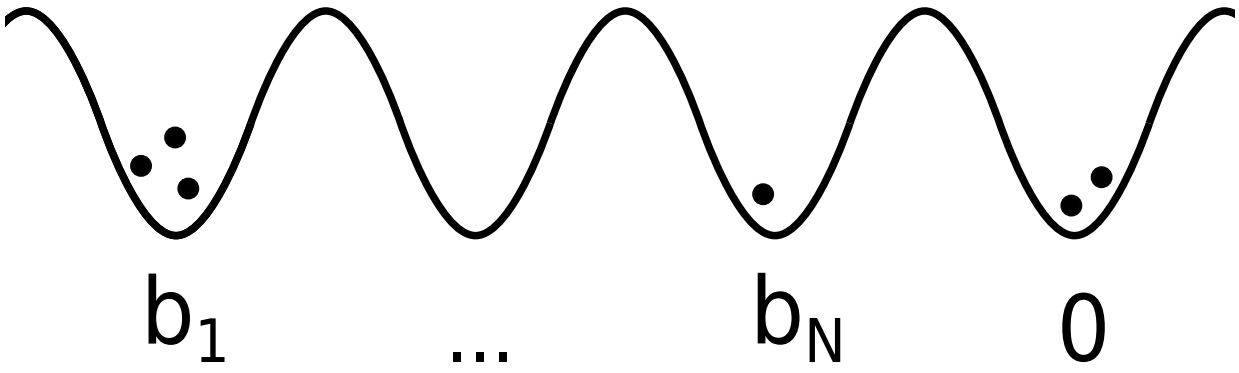

Figure 4.3: The energy of a system of ultracooled bosons trapped in a potential landscape depicted above is the squared Euclidean norm of $\mathbf{v} = (3, ..., 1, 2)\mathbf{B}'$.

The size of the Hilbert space D_S for the Single-run algorithm (letting $K_S = K_{\max} = Nm + c$ to make Single-run and Multi-run directly comparable) is, using either the hockey-stick identity or Eq (4.9)

$$D_S = \frac{(K_S + N)!}{K_S!N!} = \sum_{K=0}^{K_{max}} \frac{(K + N - 1)!}{K!(N - 1)!}. \tag{4.24}$$

Having added an extra particle reservoir site at site $N + 1$ and appended $\mathbf{0}$ to the basis, the ground state of the problem Hamiltonian H'_P is the trivial $\mathbf{0}$ vector, and the first excited state E_1 yields the shortest vector, of length $\lambda_1(\mathcal{L})$.

4.3 Asymptotic space and time

The time scaling properties of adiabatic quantum algorithms are not well understood, except for dependence on the minimum energy gap between the ground state and first excited state, the bounding of which is a hard problem in itself. In some circuit model algorithms, such as those of Shor [Sho99] and Grover [Gro96], algorithmic speed-ups over classical algorithms have been proven, but apart from a proof that AQC is equivalent (up to some polynomial factor) to the circuit model [AVDK+08], no such speed-ups are guaranteed, and it is not yet known if adiabatic quantum computation can offer any speed-up over classical optimization [Luc14]. For problems of size M, the dependence on the minimum energy gap means that usually

$$T = O(\exp(\alpha M^{\beta})), \tag{4.25}$$

though good values of α, β could still yield dramatically reduced run times versus classical counterparts. While estimating time complexity is difficult, it is possible to get a strong sense of the spatial requirements (in different architectures, this might be qubit requirements).

4.3.1 Spatial requirements

While the Bose-Hubbard model as applied in the above algorithms does not conform to the qubit model, it is nevertheless comparable via analysis of the Hilbert space size (and to find the qubit equivalent, take the $\log_2$ of the Hilbert space size). The system size is decided by the number of sites and the number of particles K_S. The number of sites is predetermined by the lattice instance at hand, and is $N+1$ for Single-run, but it is down to the practitioner to decide the number of particles K_S. Naturally one aims to pick K_S to be just large enough so that $\mathbf{x}_{\min}$ resides in the solution set with high likelihood.

The solution set for Single-run should include all of the solution sets for Multi-run up to $K_{\max} = Nm + c$, and so picking the number of particles for Single-run K_S should be based off of $K_{\max}$. Single-run has $N+1$ sites, so taking m offset particles per site gives $m(N+1)$

particles to begin with, so it is now left to decide if any extra are needed. From Fig. 4.2 one can see that for HNF bases (a conservative choice as these are generally bad bases, especially compared to LLL-reduced bases) m and c are both $O(N)$, and $c < m$. The implication is that heuristically

$$K_{\max} = Nm + c < (N+1)m, \tag{4.26}$$

and so c can be ignored from hereon in, taking $K_S = (N+1)m$. Using that $m = O(N)$ one can then deduce that the particle requirements of Single-run are $K_S = O(N^2)$.

By considering the size of the solution space as determined by the heuristically-influenced parameter choices above, the qubit requirements can be analytically shown to grow as $O(N \log N)$ as follows. The offset m required to obtain the shortest vector has been shown to grow linearly, so let $m \approx rN$ for some constant r. There are $m(N+1)$ particles in the system, so the particle number is $K_S = rN^2 + rN$. Since the Hilbert space for a system of P particles among Q sites is

$$D = \frac{(P+Q-1)!}{(P)!(Q-1)!}, \tag{4.27}$$

one can substitute in the parameters derived above, giving

$$D = \frac{(K_S+N)!}{K_S!N!} = \frac{(rN^2+rN+N)!}{(rN^2+rN)!N!}. \tag{4.28}$$

By Stirling's approximation, which is an accurate approximation for factorials, this is close to

$$D \approx \frac{1}{\sqrt{2\pi N}}\left(\frac{e}{N}\right)^N \left(\frac{(rN^2+rN+N)!}{(rN^2+rN)!}\right), \tag{4.29}$$

where e is Euler's number. Cancel out the denominator factorial with the numerator factorial,

$$D \approx \frac{1}{\sqrt{2\pi N}}\left(\frac{e}{N}\right)^N \prod_{i=1}^{N}(rN^2+rN+i), \tag{4.30}$$

then using the largest value in the product term means D can be bounded above

$$D < \frac{1}{\sqrt{2\pi N}}\left(\frac{e}{N}\right)^N (rN^2+(r+1)N)^N = \frac{1}{\sqrt{2\pi N}}(e(rN+(r+1))^N. \tag{4.31}$$

Taking $\log_2 D$ gives a qubit-equivalent of $O(N \log N)$. While this is based on heuristic analyses of some parameters which were used to derive this scaling, it agrees with the rigorous analytical spatial requirement of $O(N \log N)$ demonstrated using HNF bases in Chapter 3 and so one can have confidence in the parameters selected for the coefficient offset m and the number of runs r in Multi-run.

4.4 Results

While the analysis of Section 4.3.1 gives a strong sense of the spatial efficiency of the algorithms in this chapter, it would be nice to do the same for time scaling. To analyse that heuristically would require full quantum simulation, however, which is known to be a hard problem, and is intractable for even relatively small systems due to the exponential blow-up in Hilbert space for increasing system size. In this section I present an analysis of the algorithms run on lattice instances in low dimensions.

To gain a sense of how time sweep length T influences performance, Single-run was simulated on $200, 150$, and 100 lattices in two, three, and four dimensions respectively. In such low dimensions it is necessary to construct our own bad bases for the experiments, as HNF and LLL bases tend to be optimally reduced in two, three and four dimensions. Accordingly, I created a problem set by generating good bases then scrambling them by post-multiplication with some random unimodular matrix to obtain a worse basis. The basis vector lengths increased by a factor of $12.06, 10.08$ and 10.07 on average between a lattice's good basis, and the bad one post-scrambling. Because of the constraints of the classical simulators used, basis vectors could not be increased by too much otherwise the instances would become intractable, and the Hamiltonians were scaled (according to the largest element) to fit roughly the same eigenspectrum, as otherwise the minimum energy gaps would vary wildly meaning the sweep times would not be comparable across lattices. This Hamiltonian scaling is similar to that performed in order to run the experiments of Chapter 5 on the D-Wave quantum annealer. This scaling means that the effect of varying sweep time T is isolated, and prevents the lattice geometry from

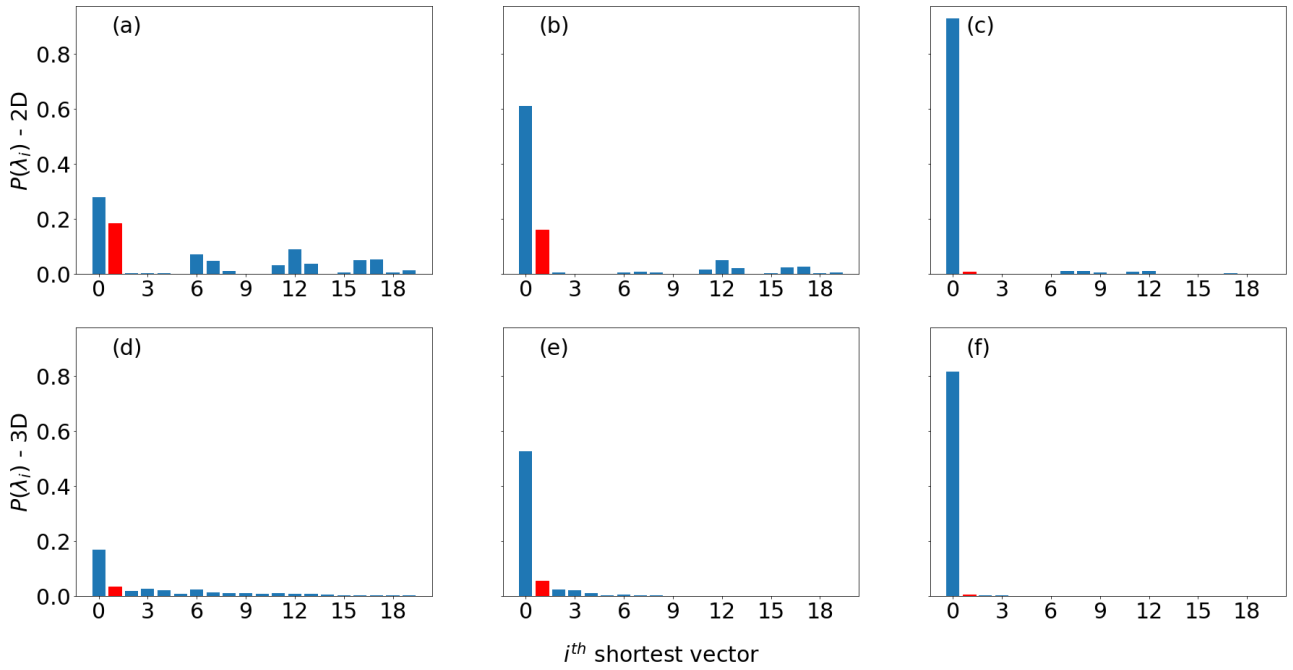

Figure 4.4: Probability of returning i^{th} lowest energy eigenstate/shortest vector averaged, averaged over 200 instances of two-dimensional lattices for $T = 1, 10, 100$ in subplots (a), (b) and (c); and averaged over 50 instances of three-dimensional lattices for $T = 1, 10, 100$ in subplots (d), (e) and (f).

significantly warping results. In all simulations, natural units were used and $\hbar$ set to 1.

Fig. 4.4 shows the averaged results for Single-run of Section 4.2.5. Each subplot corresponds to a different choice of parameters, and shows the mean probability of observing the final state to give the i^{th} shortest vector (equivalently the i^{th} eigenstate). The tallest bars on the left-hand side of each subplot represent the zero vector, and the red bars are the shortest nonzero lattice vector, of eigenenergy $\lambda_1(\mathcal{L})$. The top row is for two-dimensional lattices and the bottom row is for three-dimensional lattices. Subplots (a,d) show sweep times $T = 1$, (b,e) are $T = 10$, and (c,f) are $T = 100$. As one moves from the left-hand subplots to the right-hand plots, the concentration of probability density around the zero vector increases, meaning the algorithms are more likely to output the zero vector. This is undesirable from an SVP perspective, but is in line with adiabatic theory, in that slower sweeps correspond to better approximations of the final state $|\psi\rangle$ to the ground state E_0 of H_P.

The red bars are the optimal output of Single-run, and in an ideal scenario, just a single red bar would be observed of height 1.0, meaning that the shortest vector was found with certainty.

Looking closely at the red bars in Fig. 4.4, it is clear that while ground state probability monotonically increases with slower sweep times T, this is not the case for the shortest vector, as illustrated by subplots (d, e, f). The red bar is highest in subplot (e), indicating that $T = 1$ is too fast, and $T = 100$ is too slow, and this is analyzed further with more results later in the section.

The top row in Fig. 4.4, showing results for parameters $N = 2$, $m = 3$, and $T = 1, 10, 100$ means the systems simulated contain nine particles in three sites, which by Eq. (4.9) gives a Hilbert space of 55 eigenstates. The bottom row, shows results for $N = 3$, $m = 4$, and $T = 1, 10, 100$, and so contains sixteen particles across four sites, with a Hilbert space of size 969. By $T = 100$, both systems are nearing the adiabatic limit, with negligible probability of measuring the final state $|\psi\rangle$ to be anything other than the ground state. It should also be noted that increasing system size, keeping T constant (i.e. looking at the subplots vertically below), one can see that the probability of recording the i^{th} eigenstate decreases almost monotonically, indicating that increasing system size dilutes the probability density of low energy eigenstates for fixed T, meaning slower sweeps are required to achieve the same quality of results in higher dimensions.

Bounding the time required to achieve adiabatic sweeps for general problems is a difficult and much-researched problem, but there are new methods being investigated that may yield much more appealing time bounds. It has been demonstrated that for MAX 2-SAT there are instances for which slow sweeps are outperformed by fast sweeps [CFL+14], and in QAOA which was inspired by the adiabatic quantum algorithm, low depth circuits have been shown to find ground states by taking diabatic paths [ZWC+20]. Next I demonstrate how faster sweeps are advantageous for the quantum SVP algorithm of this chapter.

While most attention has been focused on the complexity of finding ground states, the question of finding arbitrarily low energy eigenstates is less well studied. In Single-run it is the first excited eigenstate that is targeted, with the next $\gamma = \text{poly}(N)$ lowest energy eigenstates of keen interest for the approximate version SVP_γ.

In Fig 4.5 the solid lines are mean probabilities, and the dashed lines are the 10^{th} percentile probabilities. Lines in blue represent $\mathbf{0}$ probabilities (ground state), red shows shortest vec-

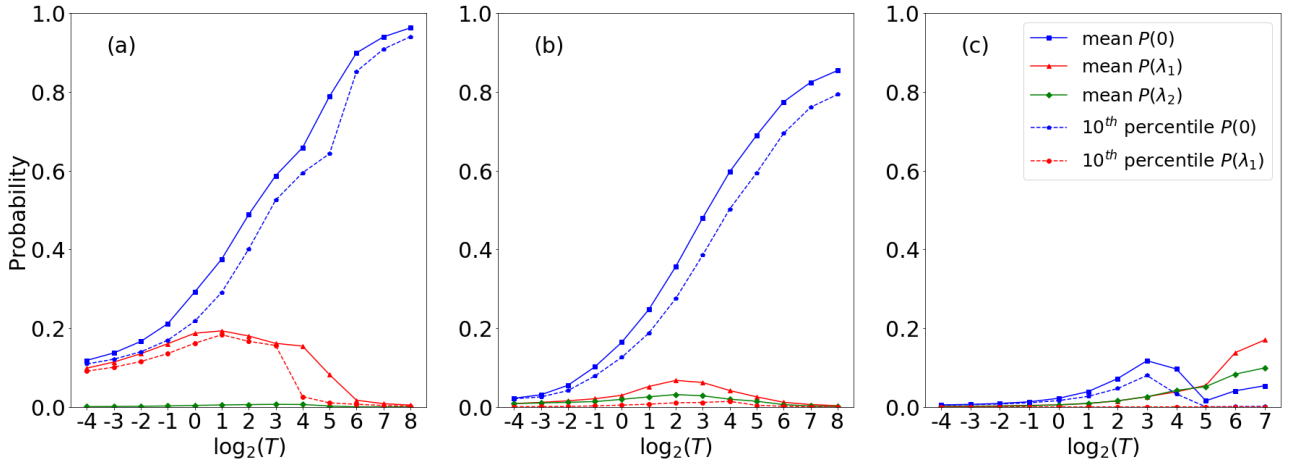

Figure 4.5: Numerical simulations of Single-run on (a): 200 two-dimensional lattices, (b): 150 three-dimensional lattices, and (c): 150 four-dimensional lattices. The mean (solid) and 10^{th} percentile (dashed) probabilities of returning the i^{th} shortest vector are depicted, in blue for the ground state (zero vector), red for the shortest vector (of length $\lambda_1(\mathcal{L})$), and green for the second shortest vector (of length $\lambda_2(\mathcal{L})$).

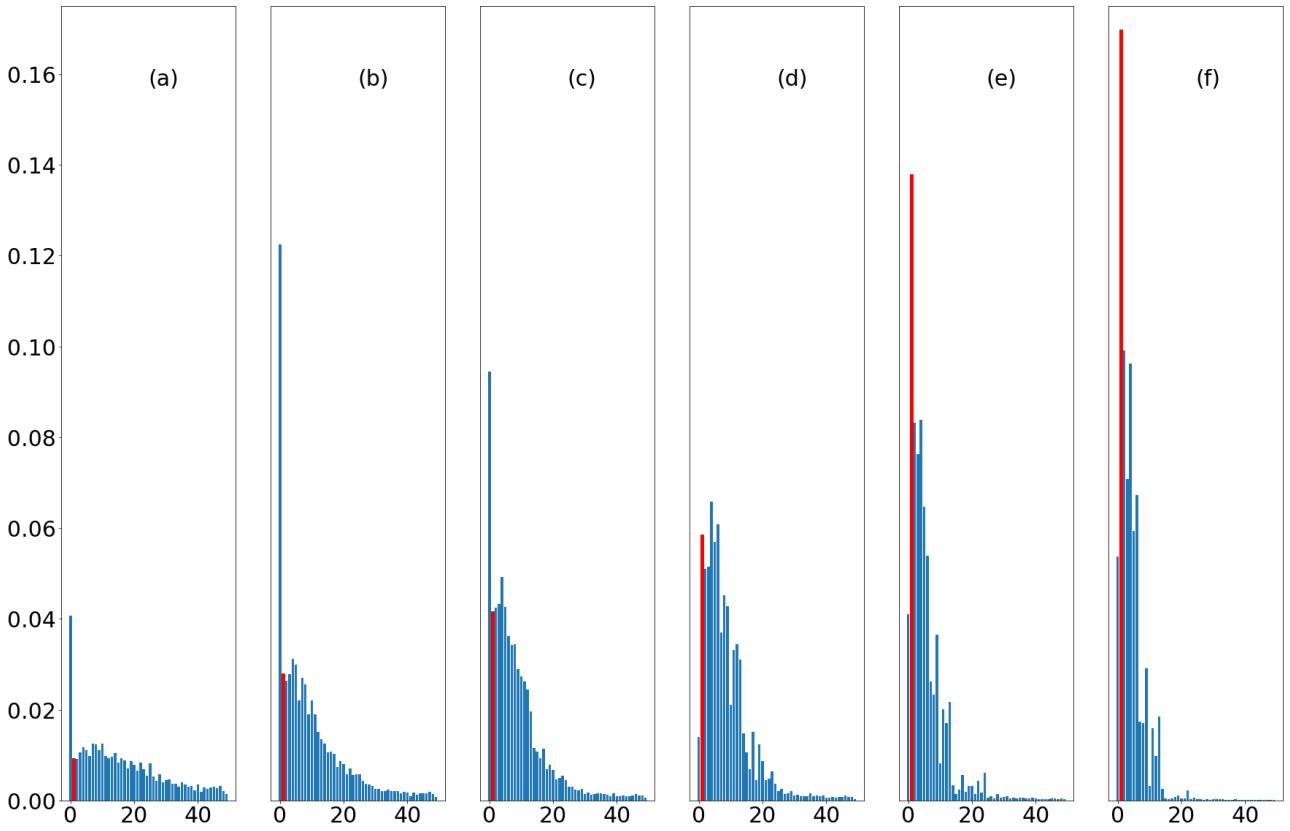

Figure 4.6: Probability of observing i^{th} shortest vector averaged across 150 instances of four-dimensional lattices, with sweep durations $T = 4, 8, 16, 32, 64, 128$ in subplots (a), (b), (c), (d), (e), and (f), respectively.

tor probabilities (first excited eigenstate of H_P), and green depicts second shortest vector probabilities (second excited eigenstate of H_P). Subplot (a) is for numerical simulations on two-dimensional lattices, (b) is for three-dimensional lattices, and (c) is for four-dimensional lattices.

In subplots (a) and (b) it can be seen from the smooth curve of the blue lines that exponentially slower sweeps give higher probabilities of recording the ground state, approaching probability 1.0 asymptotically to the right of the subplots, and approaching probability 0.0 asymptotically to the left of the subplots with exponentially faster sweeps. For the sought-after first excited state in red, however, different behaviour is observed. In the adiabatic limit to the right of the subplots the red line approaches 0.0 due to the probability density concentrating on the zero vector as the algorithm becomes adiabatic. But for exponentially faster sweeps one observes a peak in success probability, and then a gentle decrease as the faster and faster sweeps induce even more excitations beyond E_1. This peak in probability of observing the shortest vector occurs at $T \approx 2^0$ in two dimensions, $T \approx 2^2$ in three dimensions, and $T \gtrsim 2^7$ in four dimensions. Call this area the Goldilocks zone, as this is when the sweeps are not too fast (meaning there are too many excitations) and not too slow (i.e. the adiabatic limit).

Subplot (c) in Fig. 4.5 shows a slightly different picture, however, for the four-dimensional lattices. At $T = 2^5$, the ground state probability is overtaken by the green and red lines for the first and second excited states. Understanding the results presented in subplot (c) of Fig. 4.4 can be aided by looking at the results in a more granular form, which is what Fig. 4.6 depicts. Each subplot in Fig. 4.6 shows the distribution over the i^{th} excited states for a different time sweep, in a similar fashion to Fig. 4.5, with the shortest vector highlighted in red. From Fig 4.5 (c) shows that the ground state probabilities at first increase moving from faster sweeps like $T = 2^{-4}$ to $T = 2^3$, before decreasing and then rising again, Fig. 4.6 demonstrates that even as the ground state probabilities decrease in the range $T = 2^3$ to $T = 2^5$, probability density is still continuing to concentrate more and more on low energy states for slower sweeps. Eventually, way past the right hand limit of Fig. 4.5 (c) the ground state probability would again approach 1.0 asymptotically and the first excited state would approach probability 0.0. In the subplots shown of Fig. 4.6 one does see an extremely encouraging distribution over very

short vectors that nevertheless are not the ground state vector, and from subplot (a) through (e) the probability of finding the exact shortest vector increases from less than 0.02 to over 0.12 (and will decrease again for much slower sweeps). The concentration of probability in the twenty shortest vectors out of a Hilbert space of $\gtrsim 10,000$ indicates strong potential for a quantum sampler of short vectors, or solver of approximate SVP. The curious behaviour of the ground state probabilities which dip before increasing again has been observed elsewhere, and is sometimes known as a diabatic bump, as illustrated in Fig. 6 (c) of [ZWC+20].

As has been discussed previously, the probability distribution across the eigenspectrum of H_P depends sensitively on the eigenspectrum of H_t, in particular at the points where the gaps between intermediate eigenstates are small, known as avoided crossings. To add more context to the system evolution, Fig. 4.7 shows the evolution of energy levels for the five lowest energy levels, and the probability of observing the system to be in those energy levels during the evolution. The five energy levels, due to symmetry of the lattice, correspond to the unique zero vector, plus/minus the shortest vector, and plus/minus the second shortest vector. These were recorded with $T = 0.25, m = 3$ on a representative two-dimensional lattice in subplots (a,b), and $T = 4, m = 4$ on a representative three dimensional lattice in subplots (c,d).

In Fig. 4.7 on the left-hand subplots (a,b) for the two-dimensional lattice, one sees in subplot (b) that the minimum energy gap between E_0 and E_1 occurs around $t = 0.025$, and up to this point, the probability of observing E_1 is increasing. For the three-dimensional example the spectrum is more tightly packed, and this induces more volatile dynamics on the probability distribution across the lowest few eigenstates, as evidenced by the jumpy plots shown in (c). The seemingly discontinuous jumps and falls in probability in subplot (c) occur around avoided crossings, when there is a high probability of exciting to the next energy level up.

For an SVP solver it is necessary that there is at least one excitation from the ground state, and so this opens the door to sub-adiabatic sweeps as a tool to achieve optimal solutions. So while this chapter has assumed the framework of adiabatic evolution, one can obtain higher success probabilities *and* faster execution than AQC for the Single-run SVP algorithm. Furthermore, due to the symmetry of the lattice, E_1 and E_2 both yield the shortest vector (up to sign) and

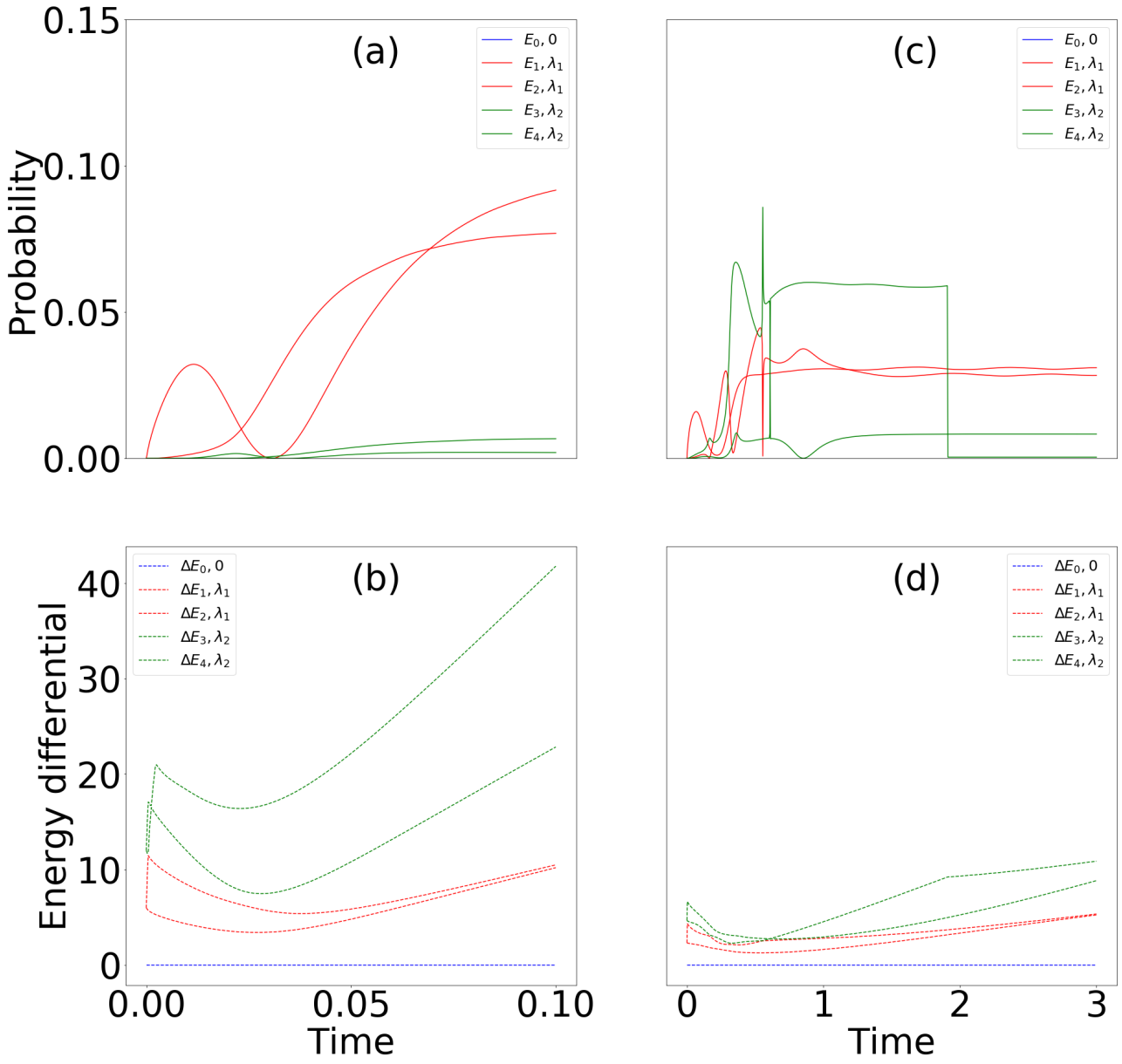

Figure 4.7: Algorithm sweep on a two-dimensional (left; $t \in [0, 0.1]$ of a $T = 0.25$ sweep) and three-dimensional (right; $t \in [0, 3]$ of a $T = 4$ sweep) lattice instance showing probability of measuring the system in an eigenstate (top) and the corresponding energy differential (bottom) —both for the lowest five energy levels —defined as $\Delta E_i^t = E_i^t - E_0^t$. Degeneracies should not occur during the intermediate Hamiltonian H_t for $t \in [0, T)$, though H_P will contain degeneracies because lattices are symmetric about the origin, meaning for every possible vector length there are at least two configurations. Pairs of lines in the same colour give vectors of equal length upon measurement (at final time $t = T$, which is off the right edge of the subplots), because one vector is the negative of the other. In (b) and (d), the lower/upper red lines are $\Delta E_1, \Delta E_2$ respectively, and the lower/upper green lines are $\Delta E_3, \Delta E_4$ respectively.

E_3, E_4 give the second shortest vector etc. This implies that one can allow for another factor of two in the tolerance of excitations from the ground state.

4.5 Conclusion

The examination of AQC-style algorithms in the context of approximate optimization (i.e. *not* targeting the ground state) opens an interesting avenue of research for computational problems which can be mapped to physical systems where low-but-not-lowest energy levels are sought-after, such as the SVP setting investigated in this chapter. The existence of the Goldilocks zone, where sweeps are neither too slow nor too fast, hints at a specific sub-adiabatic range of sweeps which could be of interest for a variety of different problems, but without the sometimes catastrophic dependence on the minimum gap $\Delta = \min_t\{E_1^t - E_0^t\}$. Demonstrating the utility of sub-adiabatic algorithms is particularly appealing for the NISQ era, as maintaining the high coherence times necessary for slow sweeps is likely to remain a prominent challenge for hardware. The numerical findings of this chapter agree closely with the simulations of Chapter 5 which presents a complimentary set of algorithms for the shortest vector problem, and thus adds to the body of evidence that AQC-style algorithms for SVP may have an important role to play in the future.

Hopefully the more generalized mapping of SVP into a Bose-Hubbard Hamiltonian will inspire other creative algorithms utilizing such periodic landscapes with ultra-cooled bosonic gases, as the avoidance of the qubit-based architecture is likely to enable scaling systems faster due to the lack of overheads from high gate counts, which are necessary for algorithms in the circuit model.

Going forward, there is much work to do in generalizing the understanding of time constraints for approximate solutions to quantum annealing algorithms, so that the sub-adiabatic range of algorithms can be better compared with their circuit model counterparts. For instance, analytical work to explain the distributions of Fig. 4.6 would go a long way to explaining what results practitioners could expect to see from fast sweeps. On the hardware front, work

has to be done to realize systems to which one can apply the Bose-Hubbard model, and with interesting complexities such as long-range interactions necessary for implementation of the SVP algorithm presented here.

Chapter 5

Quantum Ising algorithms via D-Wave

5.1 Introduction

In this chapter I describe the findings from [JCLM21] and is designed to be more or less self-contained, though some of the theoretical results of [JCLM21] have already been presented in Chapter 3 and are referenced where appropriate. The content of the chapter consists of descriptions of two variations of a quantum Ising algorithm for SVP, and an analysis of results of both numerical simulation and implementing them on a D-Wave quantum annealer. This is a pertinent experiment for a couple of reasons:

1. Firstly, while a number of quantum devices have recently been made accessible to the general public via cloud interfaces, most have extremely limited resources and low gate fidelities. This issue is particularly acute in the gate model options because of the high level of control over each qubit required. The D-Wave quantum annealer has many drawbacks, but it does allow for implementation of fully quantum algorithms to solve problem instances larger than is possible on other devices.

2. The NIST process for standardization of PQC is now in its late stages, with full standards expected to be released around 2024. Given that lattice-based cryptography is a hot

favourite in the competition, accounting for 5 of 7 finalist algorithms, it is important to pave the way for fully quantum cryptanalyses.

There are reasons why quantum annealing algorithms are a natural fit for SVP such as the fact that SVP is formulated easily as an optimization problem, making it an appropriate candidate to tackle with algorithms which aim to exploit the minimum energy principle to find low energy eigenstates.

Implementing the algorithms of this chapter on the D-Wave 2000Q quantum chip gives a flavour of what might be possible with more advanced hardware, and the likely problems to overcome and adjust to in the NISQ era. It should be emphasized however that the quantum processor used to generate the following results is highly error-prone and should not be used to extrapolate to problem instances of cryptographic significance.

For completeness, the entire formulation of the algorithm is provided in this chapter, though a more expansive explanation of some parts are available in the wider background section of Chapter 2 and the original contribution mathematical foundations in Chapter 3.

5.2 Quantum Ising SVP

The algorithm proceeds via interpolation between a driver Hamiltonian H_D and a problem Hamiltonian H_P. The problem Hamiltonian is written in a Pauli-Z eigenbasis, taking the form of an Ising model and is defined in the following, and the driver Hamiltonian is the local Hamiltonian of transverse field generated by Pauli-X operators on each qubit

$$H_D = \sum_j X_j. \tag{5.1}$$

The algorithm proceeds via the linear interpolation between H_D and H_P

$$H_t = \Big(1 - \frac{t}{T}\Big) H_D + \Big(\frac{t}{T}\Big) H_P, \tag{5.2}$$

and this framework is expounded upon further in Chapter 2.

In the rest of this section I describe the mapping of SVP into the Ising coefficients J_{ij}, h_i that define the energy of the system. The problem instance is a lattice defined by a basis $\mathcal{L}(\mathbf{B})$. The coefficients are derived directly from the input basis $\mathbf{B}$.

Any lattice point can be written in the form

$$\mathbf{v} = \mathbf{xB} = x_1\mathbf{b}_1 + ... + x_N\mathbf{b}_N, \tag{5.3}$$

and the Euclidean norm of $\mathbf{v}$ is written

$$||\mathbf{v}||^2 = \sum_{i,j} x_i x_j \mathbf{b}_i \mathbf{b}_j \ . \tag{5.4}$$

The aim is to find the integer vector $\mathbf{x} = (x_1, ..., x_N)$ that minimises this sum over $i, j = 1, ..., N$, given the fixed scalar values $\mathbf{b}_i\mathbf{b}_j$ which are determined by the lattice basis with which the algorithm is run. It would be straightforward to map this minimisation into a device consisting of coupled qudits rather than qubits, via the Hamiltonian

$$H_P = \sum_{i,j}^{N} \hat{Q}^{(i)}\hat{Q}^{(j)}\mathbf{G}_{ij}, \tag{5.5}$$

where $\mathbf{G}_{ij} = \mathbf{b}_i\mathbf{b}_j$ is the $(i,j)^{th}$ element of the Gram matrix for the lattice basis, and $\hat{Q}^{(j)}$ is a qudit operator that outputs an integer in some arbitrary range $[a, b]$. In this case, measurement of $\hat{Q}^{(j)}$ yields an integer candidate for coordinate x_j in $\mathbf{x}$. Ising models do not provide pre-synthesized qudits, however, and so it is left to the practitioner to define qudits as a function of qubits. On the D-Wave processor, measurements of qubits are performed in the Pauli-Z eigenbasis, meaning results on n-qubit algorithms are returned as strings of $\{\pm 1\}^n$ which can be read as binary strings in $\{0, 1\}^n$.

The eigenstates of H_P in Eq. 5.5 all correspond to vectors in the lattice for which every component x_j of coefficient vector $\mathbf{x}$ is within the range of values taken by the qudit $\hat{Q}^{(j)}$ (how big this range should be is addressed in Chapter 3 where an upper bound is provided, though

the question of the optimal range remains open).

The eigenvalue of the eigenstate which determines $\mathbf{x}$ is simply the squared Euclidean norm of that vector, given by Eq (5.4). Thus, the ground state of the problem Hamiltonian H_P corresponds to the (undesirable) zero vector, while the first-excited manifold will consist of states that correspond to vectors with length $\lambda_1(\mathcal{L})$ (the shortest vectors); there are usually at least two such shortest vectors, since $\mathcal{L}$ is a discrete additive group, so

$$\mathbf{v} \in \mathcal{L} \implies -\mathbf{v} \in \mathcal{L}, \tag{5.6}$$

and because the range of $\hat{Q}^{(i)}$ is chosen to be symmetric about the origin, if $\mathbf{v}$ is in the feasible solution space then so is $-\mathbf{v}$. Accordingly, there are many degeneracies in H_P, but this is not an issue because thermal transitions towards the end of the algorithm between eigenstates with vanishing gap (as they are degenerate in H_P) do not affect the quality of the solution, just the sign of the vector. Solving exact SVP is thus equivalent to finding a state in the first-excited manifold of the problem Hamiltonian H_P.

In the next part I describe two encodings of the qudit operators $\hat{Q}^{(j)}$. These can be understood as the quantum realizations of the Hamming and Binary encodings of Chapter 3. They demonstrate how to parse n-bit strings of Pauli-Z eigenstates $\{\pm 1\}^n$ returned from the D-Wave API into coefficient vectors $\mathbf{x}$.

5.2.1 Hamming-encoded qudits

This qudit mapping is simple and is not optimal in terms of space because it leads to redundancies, with multiple spin configurations corresponding to the same qudit value as discussed in Chapter 3. The reason for presenting this mapping is that it is more robust to noise (explained in Section 5.4) than the spatially optimal encoding presented next. The comparison of robustness to noise between qudit definitions is discussed further in Section 5.6. While it is not expected that even today's cryptosystems will be broken in the NISQ era, the quantum computing community must contend with poor quality qubits in the near to mid term. Con-

sequently, the straightforward Hamming qudit has the potential to be a useful tool in the near future. Despite the spatial inefficiency of this mapping, the D-Wave 2000Q provides sufficient qubits to run the quantum SVP algorithm with Hamming qudits for problem instances of interesting size. As shown in the following, the superior performance of Hamming qudits in this application makes this version the quantum Ising SVP algorithm *for now*.

The Hamming qudit operator is defined as a sum of qubit operators

$$\hat{Q}^{(j)}_{Ham} = \sum_{p=0}^{2^{k+1}} \frac{\hat{Z}_{pj}}{2}. \tag{5.7}$$

This qudit operator assigns a value to each computational basis state by counting the number of qubits in the the $+1$ state, its *Hamming weight*, shifted so that the possible values are symmetric about zero. The qudit outputs an integer in $[-2^k, 2^k]$ upon measurement and is comprised of $2^{k+1} + 1$ qubits. From here on, when referring to the quantum Ising SVP algorithm with Hamming-weight-encoded qudits, I use the term *Ham*.

5.2.2 Binary-encoded qudits

This qudit mapping reads off a measurement of register of k qubits as a binary number. Spatially, this mapping is optimally efficient because each Ising spin configuration relates to a distinct coefficient vector. This qudit mapping induces a wider range of qubit-qubit interaction strengths, however, which requires high quality quantum hardware as shown in Section 5.6, so this is the quantum Ising SVP algorithm *for later*.

To read a Pauli-Z eigenstate in a $\{0, 1\}$ basis, one can measure a qubit using the operator

$$\hat{O} = \frac{(\mathbb{1} - Z)}{2}, \tag{5.8}$$

though for practical implementation one can measure using a Z gate and perform the rest of the transformation classically. To interpret a set of qubits read in the $\hat{O}$ basis as a binary

number, one can define the qudit on the j^{th} operator with output in the range $[0, 2^{k+1} - 1]$ as

$$\hat{Q}^{(j)} = \sum_{p=0}^{k} 2^p \hat{O}_{pj}. \tag{5.9}$$

Finally, shifting the range down so that the output is symmetric about 0, define the Binary-encoded qudit as

$$\begin{aligned} \hat{Q}^{(j)}_{Bin} &= \sum_{p=0}^{k} 2^p \hat{O}_{pj} - 2^k \mathbb{1} \\ &= \frac{1}{2}\Big(\sum_{p=0}^{k} 2^p \mathbb{1} - \sum_{p=0}^{k} 2^p Z \Big) - 2^k \mathbb{1}. \end{aligned} \tag{5.10}$$

One can collect the $\mathbb{1}$ terms by applying the identity

$$\sum_{p=0}^{k} 2^p = 2^{k+1} - 1, \tag{5.11}$$

so that the qudit operator can be expressed fully as

$$\hat{Q}^{(j)}_{Bin} = -\sum_{p=0}^{k} 2^{p-1} \hat{Z}_{pj} - \frac{1}{2}\mathbb{1}, \tag{5.12}$$

which maps measurements $Z_{0j}, ..., Z_{kj}$ on qubits $(0, j), ..., (k, j)$ to integers in the range $[-2^k, 2^k - 1]$, which is roughly symmetric range around the origin. The values can be read as a binary number in $[0, 2^{k+1} - 1]$ which is shifted down by 2^k, to give the required range. From here on, when referring to the quantum Ising SVP algorithm with binary-encoded qudits, I will use the term *Bin*.

5.3 Qubit implementation on D-Wave processor

The quantum Ising model assumes full qubit-qubit connectivity such as in the Sherrington-Kirkpatrick model [Pan12], whereas D-Wave processors do not yet natively offer full connectivity. The D-Wave 2000Q uses a Chimera topology in which there are many clusters of eight

highly (but not fully) connected qubits, and each cluster is connected to eight external clusters. The recently released D-Wave 5000Q processor offers a substantially more connected graph of qubits known as the Pegasus topology [MHRB19]. The algorithms of this chapter were implemented on the D-Wave 2000Q hence used the Chimera topology.

Each highly connected cluster of eight qubits is represented as a diamond formation of dots in Fig. 5.1. A model requiring full connectivity can be mapped into D-Wave's Chimera topology at quadratic cost in the number of qubits. This is done by creating what are known as qubit chains, which are illustrated in Fig. 5.1 with separate qubit chains in different colours.

The qubit chain works by strongly incentivizing all constituent qubits to return the same value, doing so by assigning strongly ferromagnetic couplings between all connected qubits in the chain. The strength of such ferromagnetic couplings are double that of the maximum permitted tunable couplings which are used to encode the problem. If not all qubits in the chain return the same value, this is called a chain break, and the chain is error corrected via a democratic majority vote to decide on the overall output of the qubit chain (which represents one qubit in a fully-connected graph).

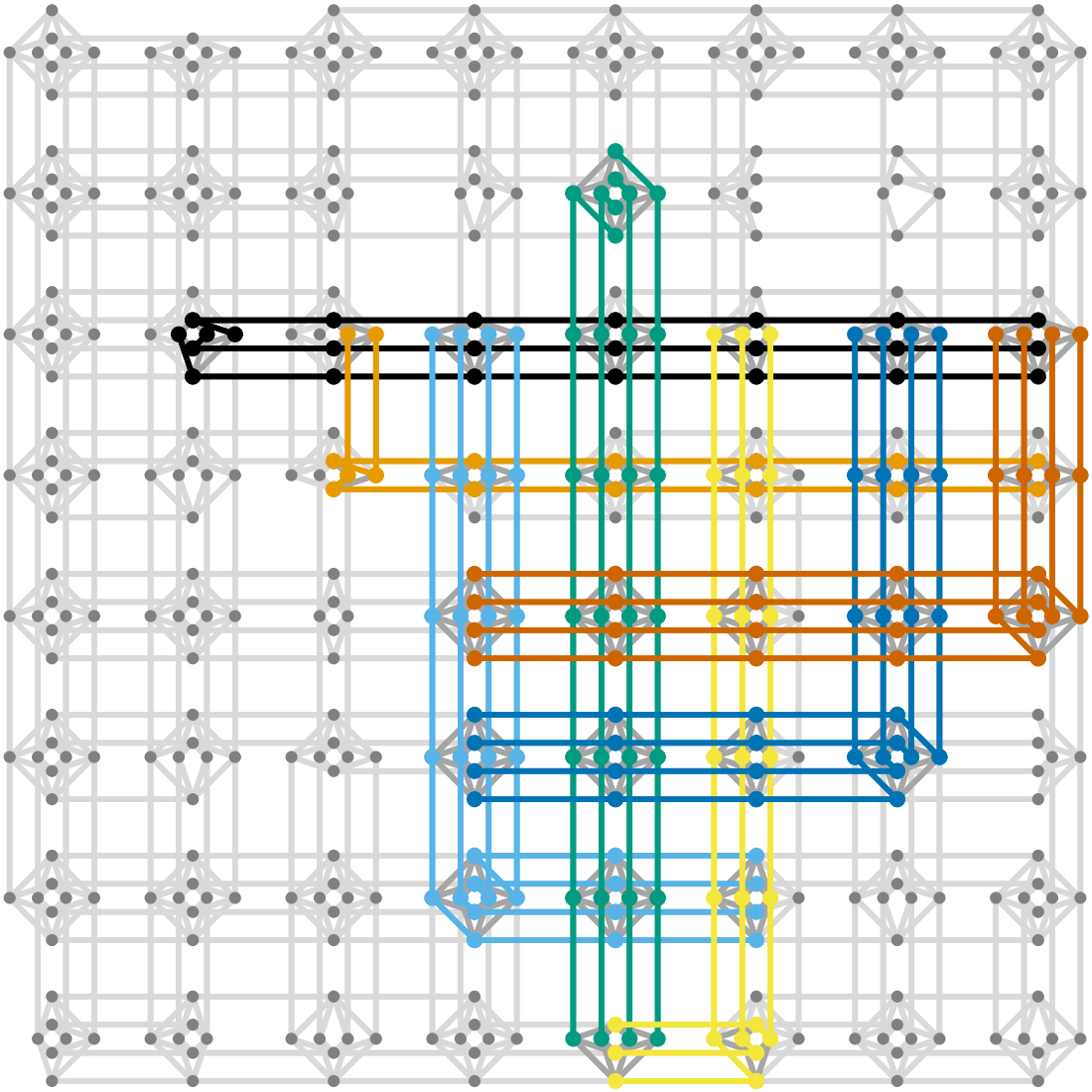

Figure 5.1: Diagram from [BKR16] showing fully-connected qubits, each represented by qubit chains of a different colour, in the Chimera topology. Qubits of the same colour have strong ferromagnetic interactions with other qubits with which it is connected in the same chain.

5.4 Algorithm experimentation setup

Ham and Bin of Section 5.2 were implemented on the D-Wave 2000Q quantum annealer to gain understanding of how such algorithms can be expected to perform on NISQ era hardware. To complement these results, numerical simulations of ideal closed-system versions of the algorithms were performed. The aim of this is to compare near-term performance (D-Wave trials) with long term potential, which is when the algorithms of this paper would be most likely to present a material threat to lattice cryptography. Due to computational constraints the prob-

lem instances examined numerically are smaller than the largest instances implemented on the D-Wave 2000Q.

To create a set of problem instances, random full rank integer matrices were first generated (the good bases) with coefficients in $\{0, 1\}$, and then multiplied by random unimodular matrices with coefficients in $\{-6, \ldots, 6\}$ to get bad bases. This method was used in order to emulate a cryptographically realistic scenario, whereby the existence of good bases (i.e. containing short vectors) is guaranteed. The bad bases were not in HNF, as would be required in order to apply Theorem 3.1. This is not an issue as HNF of low dimensional lattices tends to give highly reduced bases which is not of use for the experiments in this work.

The qudit range was fixed as $[-4, 4]$ for Ham and $[-4, 3]$ for Bin. With this choice of range a shortest vector is represented in all of the three-dimensional lattice instances used, and approximately half of the instances in each of the higher dimensions, as I did not post-select instances where a $\lambda(\mathcal{L})$ solution is possible. The reason this choice of range (slightly too small to guarantee existence of the shortest vector in the solution set) was made is due to the trade-off of obtaining significantly better results for analysis. While analysis of this degree of freedom would be interesting, fixing the qudit range allowed for a concentrated and thorough analysis of other metrics.

To map the fully connected Ising SVP graph of H_P into D-Wave's Chimera topology (thereby generating the qubit chains of Section 5.3), the D-Wave Ocean API's minorminer library was used, whereby an input edge list (representing the fully connected Ising SVP graph) is mapped to a target edge list (representing the D-Wave 2000Q qubit graph).

The D-Wave API offers a feature called 'spin-reversal transforms'. This mitigates biases in the results due to the D-Wave chip's asymmetric application of ferromagnetic and antiferromagnetic couplings. The biases are mitigated by randomizing the Pauli-Z operator signs and adjusting adjacent signs on ZZ gates in the problem fields accordingly. This feature was not used in experiments conducted in this chapter, but after subsequent discussions it is thought that this may potentially improve results.

A final aspect to mention concerns the energy spectrum provided: the range of interaction strengths must be normalized to fit the absolute spectral range available on the quantum processor. D-Wave provide a constraint of $|J_{ij}| < 1.0$ (and ferromagnetic couplings to implement qubit chains take interaction value 2.0). This means that all ZZ gates must have their strengths adjusted accordingly, and due to the exponential prefactors of H_P in Bin, the strongest gates are significantly stronger than the gates between least-significant qubits. Consequently these gates take very low interaction strengths and this makes them highly vulnerable to any kind of noise, which is believed to adversely affect Bin relative to Ham results discussed in the following section.

5.5 Numerical results

Both *Ham* and *Bin* algorithms were numerically simulated for three-dimensional lattices over a range of sweep times to investigate what performance could be expected in a post-NISQ regime.

Fig 5.2 shows the probabilities of measuring the final state $|\psi\rangle$ to be in the ground state (meaning zero-vector, blue circles), the first excited state (shortest vector, red diamonds) and the second excited state (second-shortest vector, green crosses) for increasing sweep durations along the x-axis. Subplot (a) is for Ham and subplot (b) is for Bin. For high T, (i.e. slow sweeps) one sees high probabilities of observing the ground state of H_P which is consistent with adiabatic theory. Ham approaches the adiabatic limit for much smaller T values than for Bin, which approaches the adiabatic limit past the right edge of subplot (b) as the blue line will continue to increase asymptotically approaching a probability of 1.0.

For small T (that is, fast sweeps) Ham and Bin show similar probabilities for each of E_0, E_1, E_2. The sum of $P(E_0) + P(E_1) + P(E_2) << 1$ for fast sweeps due to high probability of further excitations to even higher eigenstates, which imply longer vectors. This means that very fast sweeps are not much use because the algorithm returns nonzero lattice vectors which are too long, and very slow sweeps give high probability of returning the zero vector which is equally useless. In between, there is a regime of intermediate sweep durations in which the probability

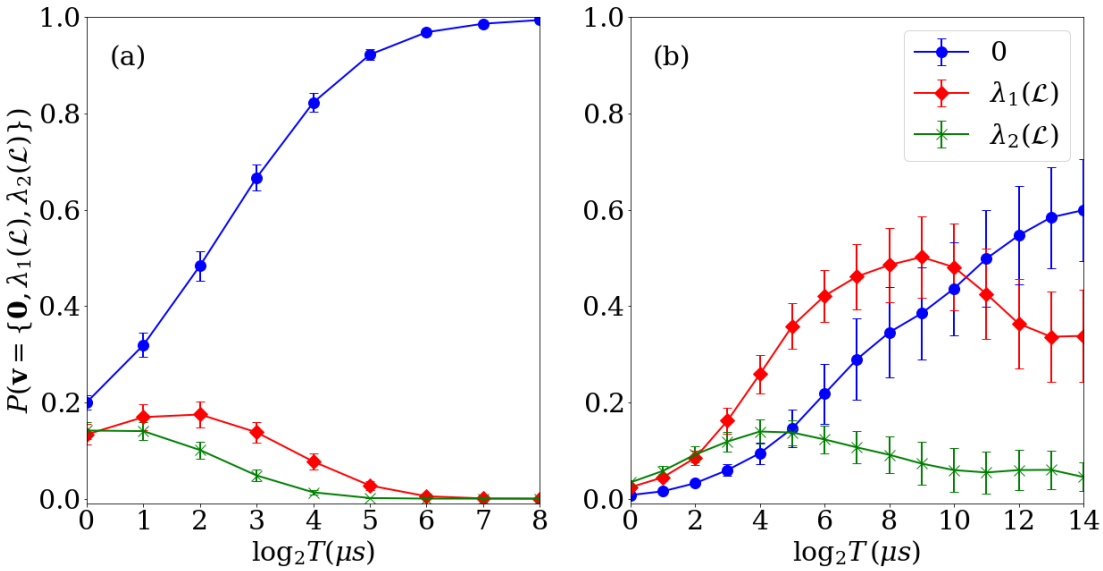

Figure 5.2: Numerical experiments carried out on 20 3D lattices. The lines show mean final probabilities for measuring the system to be in the ground (blue circles), first (red diamonds) and second (green crosses) excited eigenstates —grouped by degeneracy —at completion for sweep lengths T, increasing in powers of 2, for the quantum Ising algorithm implemented with (a) Hamming-weight-encoded qudits (*Ham*) and (b) binary-encoded qudits (*Bin*). The *Ham* algorithm nears the adiabatic limit at $T = 2^8$, after the shortest vector success probability peak at around $T = 2^2$, which is considerably lower than the peak time for *Bin*. The *Bin* algorithm nears the adiabatic limit past the rightmost data point but not before an encouraging peak in shortest vector success probability at $T = 2^9$. Standard errors across the 20 samples are shown in both subplots by vertical error bars.

to obtain the desired shortest nonzero vector is maximized. In subplot (a), this regime is around $1 \lesssim \log T \lesssim 2$, and in subplot (b) it is around $7 \lesssim \log T \lesssim 10$. In this 'Goldilocks zone' [JGLM20], the sweep is fast enough to induce excitations from the ground state, but also sufficiently slow to avoid excessive excitations to undesirably high energy eigenstates.

Comparing subplots (a) and (b) in Fig. 5.2 highlights that the probability of obtaining the first excited state (red diamonds) in the Goldilocks zone is substantially higher for Bin encoded qubits than for Ham encoded qubits. This implies the Bin algorithm is clearly preferable under the conditions simulated, which are low noise, long coherence time and low error environments.

The superior solution probabilities demonstrated by Bin are also attributable to the adiabatic theorem. In Bin, the minimum eigenenergy gap

$$\Delta = \min_t (E_1 - E_0) \tag{5.13}$$

between the ground state and first excited grows very small due to the large range of interaction magnitudes present in the qubit coupling coefficients. A representative example of the gap evolution over the course of the algorithm for both Ham and Bin is shown in Fig. 5.3. The initial ($t/T = 0$) and final ($t/T = 1$) gap coincides for both algorithms, but the gap for the Bin (blue solid) decreases rapidly in the early stage of the dynamics. For $t/T \lesssim 0.2$ the gap is much smaller than the minimal gap for Ham (red dashed lines) which reaches a minimum at $t/T \lesssim 0.6$. The smaller value of Δ for Bin means that to achieve the same probability of observing the ground state, much longer sweeps are required (which in the SVP application is advantageous because the ground state is not desired). This is because of the time complexity of AQC, in which a smaller minimum energy gap means that excitations to the first excited state are more common. Bin, furthermore, has higher spatial efficiency. Thus, Bin is certainly the algorithm *for later*.

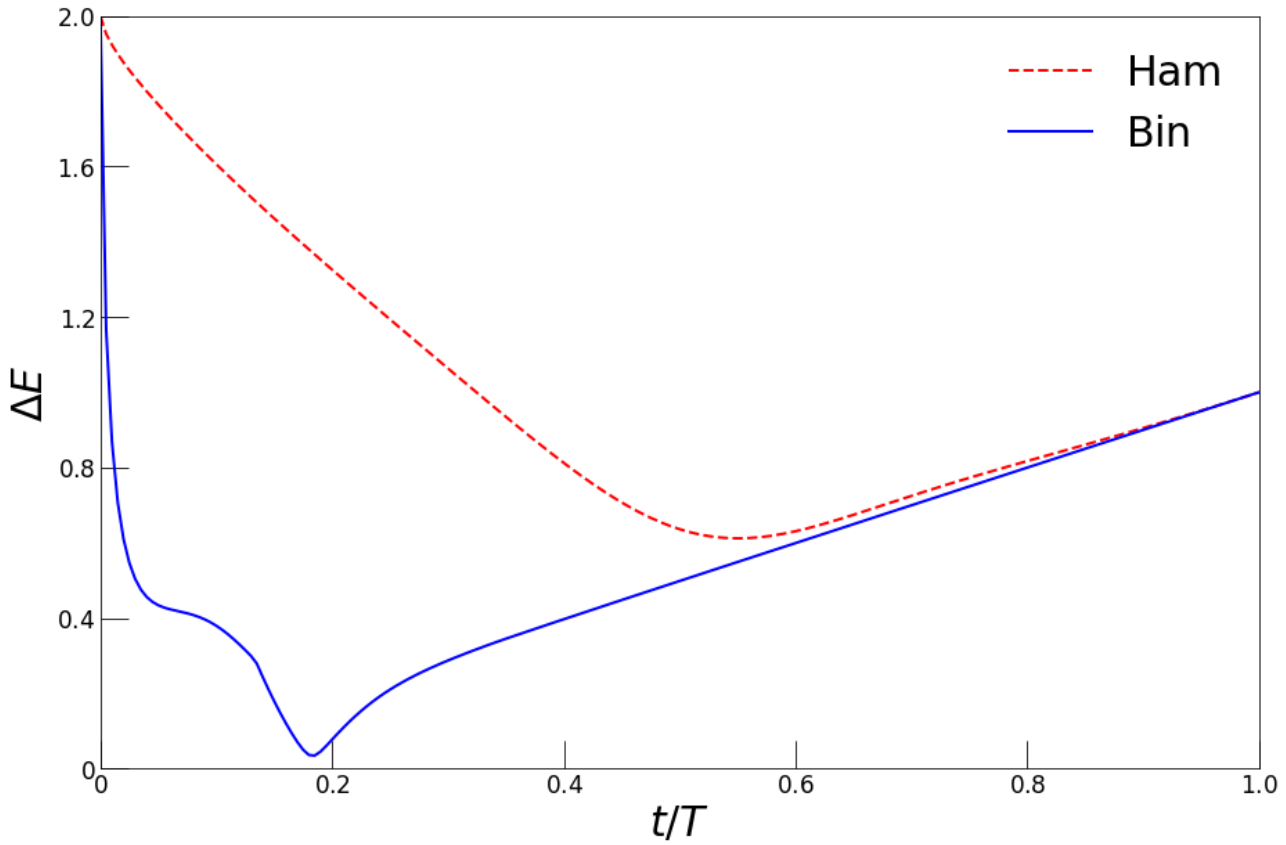

Figure 5.3: Energy gap ΔE, in arbitrary units, between the ground state E_0 and the first excited state E_1 at each time t along a linear sweep of total duration T for the Ham algorithm (red dashed) and the Bin algorithm (blue solid). The minimum gap occurs earlier in the evolution for Bin and is much smaller implying higher excitation probabilities.

5.6 Quantum hardware-generated results

The rest of the results presented are the product of implementation of Ham and Bin on a quantum annealer.

5.6.1 D-Wave Quantum Annealer

The quantum SVP algorithms Bin and Ham can be performed on the D-Wave 2000Q quantum annealer because of their formulation as a transverse field Ising model algorithm [MHRB19]. This section provides an analysis of the results recorded on the D-Wave quantum processor. Implementing Bin and Ham in such a way as to gain insightful results requires a certain amount

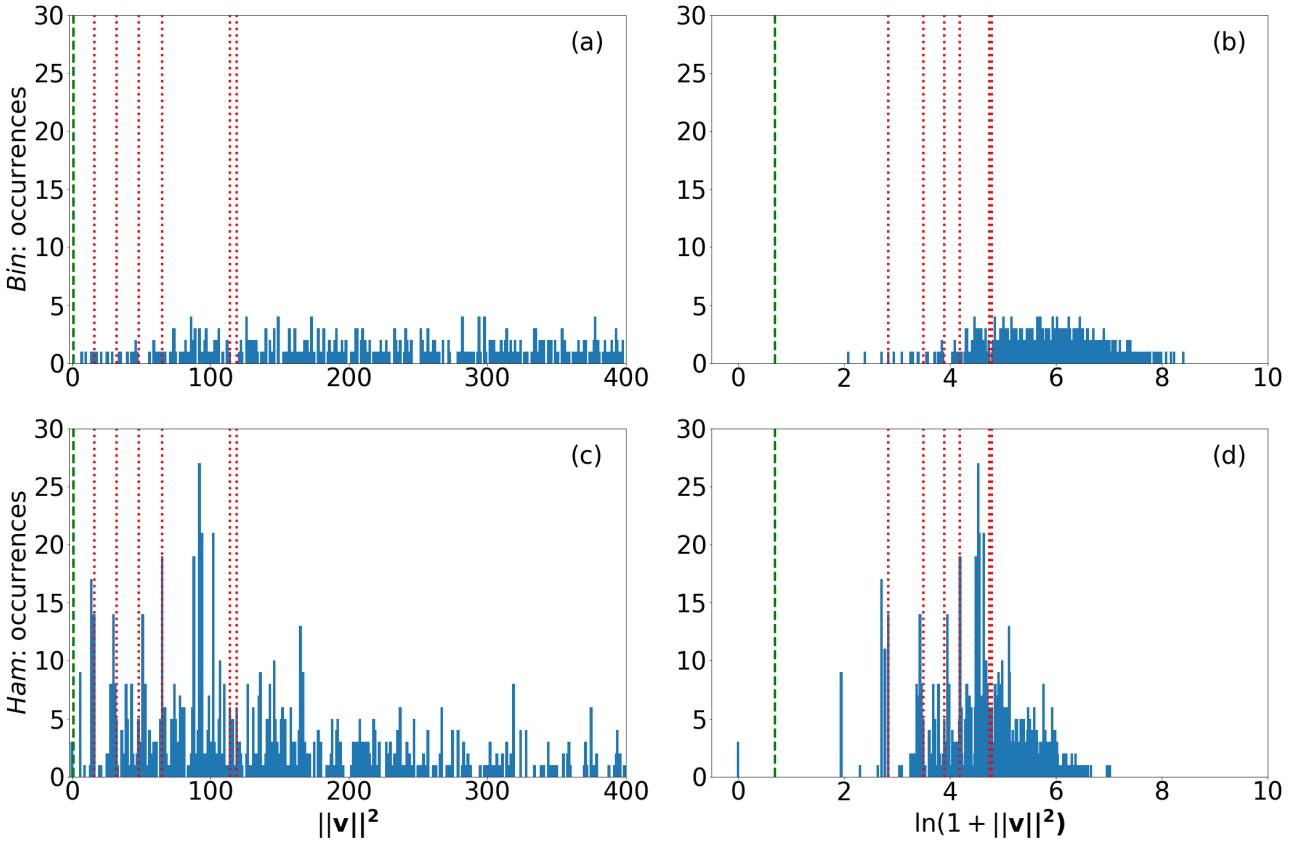

Figure 5.4: Illustrative results from D-Wave for Bin (a,b) and Ham (c,d) algorithms showing vector length squared on the x-axis on the left (a,c), and the natural logarithm of (1 plus vector length squared) on the right (b,d), with bar height indicating the number of occurrences. The right hand subplots show the same data as the left hand subplots, but allow an easier comparison due to the natural logarithm transform. This is a sample experiment of 900 runs of both Ham and Bin on the same 6-dimensional lattice, all with a sweep length of $T = 64\mu s$. The green dashed verticals indicate the length $\lambda_1(\mathcal{L})$ on the x-axis, and the red dotted verticals indicate the length of the 6 input basis vectors (from which $\mathbf{G}$ is derived) on the x-axis. The heights of the green dashed and red dotted verticals have no significance.

of tinkering with the D-Wave API and the hyperparameters of the algorithms in question as described in Section 5.4. The quadratic cost of mapping from the fully connected Ising graph to the Chimera topology means that while the maximum problem size (Ham on seven-dimensional lattices) was 56 qubits, this translates (non-deterministically) to over $1,000$ physical qubits on the D-Wave processor, using the Ocean package tools for the mapping.

5.6.1.1 Representative example

Fig. 5.4 shows the results obtained for representative (six-dimensional) lattice. The x-axis shows the vector norms squared in subplots (a) and (c) and on a natural logarithmic scale

in (b) and (d). The height of the blue bars show the number of occurrences (y-axis) of the corresponding vector lengths over 900 samples of the algorithms. The green dashed vertical indicates the lengths of the shortest nonzero vector and the red dotted vertical lines show the length of the six basis vectors used given as input via the matrix $\mathbf{G}$.

Subplots (a,b) show the results for Bin, and (c,d) show the results for Ham. Subplots (a) and (c) show the same data as (b) and (d), but (a) and (c) show shorter vectors well, whereas (b) and (d) show the full distribution including the longest vectors returned.

Comparing the subplots of (b) and (d) one sees that for Bin the majority of vectors returned are longer than all of the input basis vectors, whereas Ham performs substantially better, giving much shorter vectors. This superior performance of Ham can also be seen from subplots (a) and (c) too, as in (c) one observes occurrences of order 10 for vectors shorter than the shortest basis vector, whereas occurrences of vectors of similar lengths are between 0 and 3 in Bin, as seen in (b).

Perfect performance (for all subplots in Fig. 5.4) would be a single tall blue bar of height 900, located on the x-axis at the same location as the green dashed vertical. This would mean the output of the algorithm was the shortest vector with certainty. The farther to the right (longer) a blue bar is, the less desirable the vectors it represents are. Bars further right than *all* of the red dotted vertical lines can be immediately discarded as they are *longer* than all of the input basis vectors $\{\mathbf{b}_1, \ldots, \mathbf{b}_N\}$. Showing these results too, however, offers useful context as it helps to compare the Ham with Bin, and in doing so one can see that Ham outputs considerably shorter vectors on the D-Wave 2000Q than Bin.

Though Fig. 5.4 provides a helpful picture of the distribution of the quantum annealing results for a representative example and allows for a qualitative comparison of Ham and Bin, a more rigorous statistical analysis of Bin and Ham is given in the following where results across many lattices in dimensions three to seven.

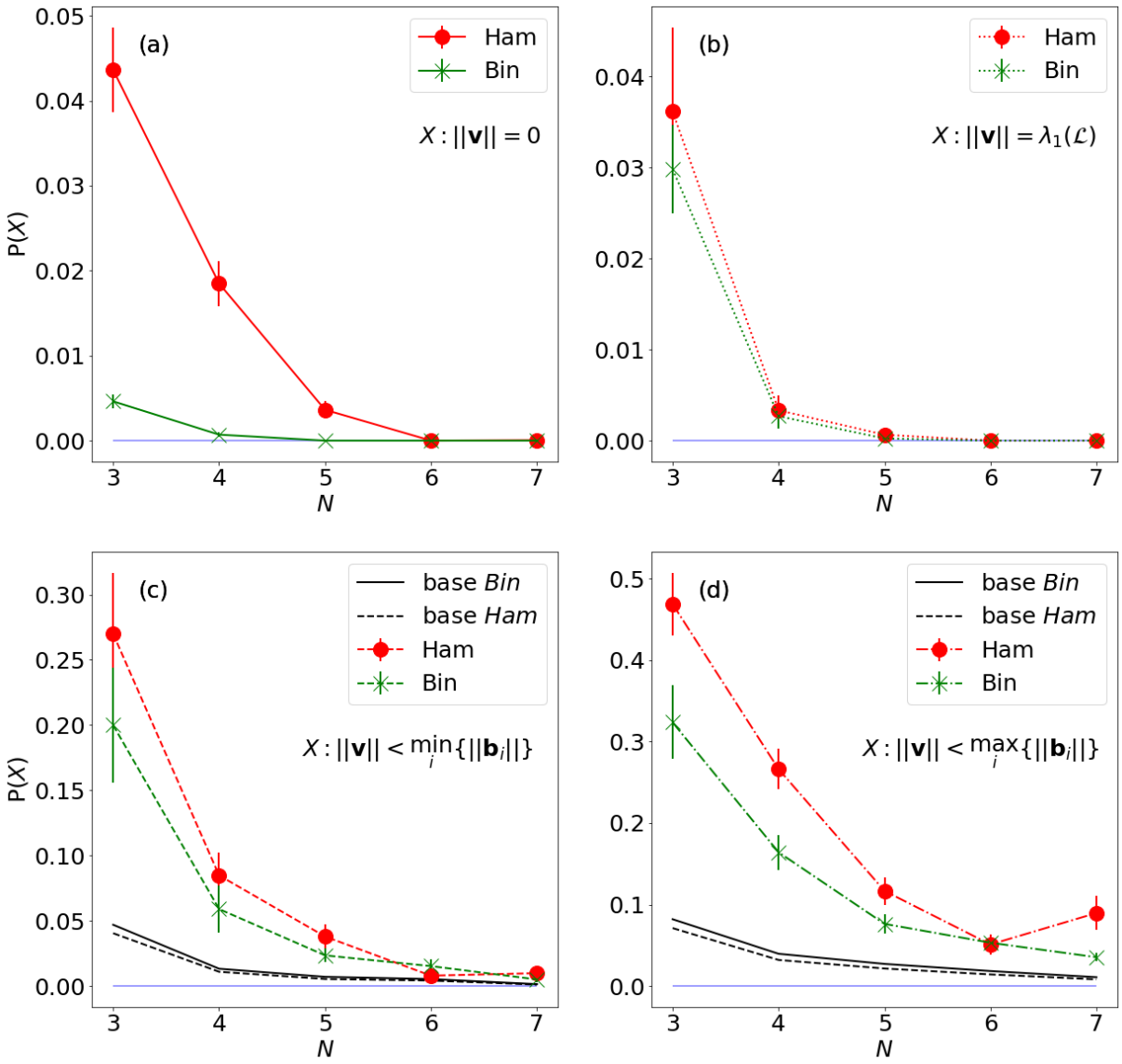

Figure 5.5: $T = 1\mu s$ for Ham (red circles) and Bin (green crosses) algorithms performed on 19 randomly generated 'bad' lattice bases in dimensions $[3, 4, 5, 6, 7]$, and for four different figures of merit: ground state/zero vector ((a) not desired), shortest vector ((b) ideal), shorter than minimum input basis vector (c), and shorter than median input basis vector (d). The black lines of subplot (c,d) are baselines indicating the results from uniform random sampling of the feasible solution space (dotted for *Ham*, solid for *Bin*), i.e. the proportion of lattice vectors in the solution space that are shorter than shortest (c), median (d) input basis vectors.

5.6.1.2 Aggregated results

Fig 5.5 summarizes results from 19 lattices in dimensions $N = \{3, 4, 5, 6, 7\}$, according to four key figures of merit (FoM). The experiments consisted of 900 samples per lattice instance with the minimum sweep time available on D-Wave 2000Q of $1\mu s$. Lattice dimension is along the x-axis, and probability of achieving the figure of merit corresponding to the specific subplot is along the y-axis. In each lattice dimension, probabilities are averaged over the 19 instances, and standard error bars are shown.

Ham is illustrated in red circles and Bin in green crosses across all subplots. In subplots (c) and (d) the black solid and dotted lines indicate baseline probabilities (for Ham and Bin respectively) and are separate due to the different (but asymptotically close) Hilbert spaces of the two algorithms, because Ham searches over a coefficient space of $[-2^k, 2^k]^N$ and Bin searches over a space of $[-2^k, 2^k - 1]^N$.

Both Ham and Bin suffer decreasing success probability for all FoM as lattice dimension grows. This is expected due to the exponentially larger Hilbert space to search over. In Fig. 5.5 the lines of red circles sit higher than the green crosses for all subplots, indicating higher success probabilities for Ham on the quantum annealer across all lattice dimensions and FoM, which is consistent with the findings of the representative example of Fig. 5.4.

Subplot (a) shows the probability of the algorithms being measured in their ground states, which in this case is the undesired zero vector $\mathbf{0}$. While unhelpful in the case of SVP this would constitute the strongest metric of success for most AQC applications. While Ham finds zero vectors in $\approx 5\%$ of samples for three-dimensional lattices, decreasing to $\approx 0\%$ in higher dimensions, Bin displays much lower probabilities even for the lowest dimensions. This is primarily due to the smaller minimum energy gap Δ between the ground and first excited states of H_t for Bin versus Ham. This point has been discussed in more detail in Section 5.5 with the aid of Fig. 5.3.

The zero vector probability of subplot (a) might seem pointless to analyze due to the fact it is of no use in the lattice context, but it should be interpreted with respect to the other FoM,

as it is a metric of success for traditional AQC. This indicates that for similarly formulated computation problems where the ground state is sought, Ham far outperforms Bin on present quantum annealing hardware.

In Fig. 5.5, subplot (b) is the probability of the algorithms solving the exact shortest vector problem, known to be NP-hard. Unlike the distinct performance profiles of Ham and Bin on the ground state FoM, in subplot (b) one can see that both Ham and Bin behave extremely similarly, measuring the shortest vector in $\approx 3{-}5\%$ in three dimensions, with no shortest vectors observed in six and seven dimensions. High probabilities in this FoM are most important when considering a potential attack on SVP.

As individual eigenstates represent an exponentially small proportion of the Hilbert space, and quantum hardware is not yet highly performant, it is more straightforward to analyze Ham and Bin by looking at ensembles of vectors. This allows for easier analysis within the constraints of 900 repetitions per lattice instance for each of Ham and Bin. Subplots (c,d) display the results for two such collections of vectors.

In Fig. 5.5, subplot (c) shows the probability of observed vectors being shorter than the shortest basis vector, and (d) shows the probability of observed vectors being shorter than the median basis vector. The more relaxed conditions for these figures of merit than in (a,b) mean higher probabilities are recorded. Ham and Bin start with success probabilities of order 10% in both (c) and (d) followed by a smooth decay in success probability with increasing lattice dimension. The decay of success in (c) and (d) is, encouragingly, more gentle than for (a) and (b). The baseline probabilities in black demonstrate the proportion of the Hilbert space which satisfy the respective FoM. These are equivalent to the success probability for a naive classical uniform sampler over the coefficient space $[-2^k, 2^k]$ (dotted) and $[-2^k, 2^k - 1]$ (solid).

The vectors that constitute successful runs in subplots (c) and (d) do not solve exact SVP but have important utility in lattice algorithms. Firstly, the approximate version of SVP, known as SVP_α simply requires a vector of length less than $\alpha\lambda_1(\mathcal{L})$, which is to say, an eigenstate of H_P of energy less than $\alpha^2 E_1$. Furthermore, basis reduction algorithms [LLL82, Her50] proceed by iteratively decreasing the lengths of basis vectors, and a quantum algorithm that can

reliably output vectors shorter than the minimum/median input basis vectors could potentially be a valuable tool. And lastly, combinatorial algorithms, such as sieving algorithms rely on some initial distribution of vectors [ADRSD15, ASD17] before iteratively combining to produce shorter vectors. A quantum lattice vector sampler which could be shown to draw vectors from a somewhat short distribution could be valuable in these cases too.

The results of Fig. 5.5 are a collection of 19 datasets just like that displayed in Fig. 5.4 for each of the five lattice dimensions, so it is important to understand how vectors visualized in Fig. 5.4 relate to Fig. 5.5. Success in Fig. 5.5 (c) means outputting a vector of length shorter than the minimum input basis vector. In Fig. 5.4 (any of the subplots) this would correspond to the sum of the heights of blue bars which are to the left of the leftmost red dashed vertical, divided by 900 to get a probability. Fig. 5.5 (d) shows the sum of the heights of blue bars to the left of the middle red dashed vertical, then divided by 900. These would give a success probabilities for a single lattice for the FoM of subplots (c) and (d) of Fig. 5.5. The points in Fig. 5.5 (c) and (d) are then simply the average over these 19 values, with standard error bars displayed.

It is important to remember that in Fig. 5.5, subplot (b) it is not necessarily a problem that probability of obtaining $\lambda_1(\mathcal{L})$ goes to zero as dimension increases. What is important is the rate of decay. For example, a polynomial rate of decay would in fact break lattice cryptography and would represent a catastrophic setback to cryptographic preparations for the post-quantum era. An exponential decay could even equal the current state of the art among SVP algorithms. The results presented here are a first taste of what is possible and do not provide enough data to heuristically extrapolate what this scaling would be for high lattice dimensions. A key point however, is that Ham demonstrates superior performance to Bin at almost every data point across Fig. 5.5, in return for a linear overhead in qubit requirements. While in the near term high quality quantum devices offer extremely low qubit counts, more imperfect machines such as the quantum annealer used in this work can offer very high numbers of poor quality qubits meaning that such a linear overhead is not a major concern for the annealing algorithms presented here. For the reasons described above, Ham is thus the algorithm *for now*.

5.7 Conclusion

Ham and Bin, the two quantum SVP algorithms studied in this chapter, provide a strong foundation from which fully quantum cryptanalyses of lattice cryptosystems can progress. The algorithms adapt the problem Hamiltonian of [JGLM20] into one which fits a model of quantum computation which is more accessible with today's hardware. Even with highly imperfect machines, as are available at the time of writing, such early analyses are important as classical lattice algorithms often perform much better than their provable upper bounds, so although no rigorous analysis of Ham and Bin is yet available, it is reasonable to believe that a heuristic analysis will ultimately prove essential to ensure the security of lattice-based cryptosystems. It will be encouraging to see quantum hardware improve to the point that larger experiments like the ones described above can be performed in order to extract reliable performance bounds for Ham, Bin and their offspring. Already since these experiments were originally performed, D-Wave has released the 5000Q with a significantly more connected qubit topology, and doubtless other improvements too. As circuit-based hardware devices rapidly expand their qubit count over the next decade, it will be insightful to trial adapted versions of Ham, Bin in this model, particularly seeing as for Bin, the $O(N \log N)$ space requirement is a reasonably forgiving bound on qubits.

The problem Hamiltonians presented here are atypical by comparison with most adiabatic quantum algorithms due to the fact that optimal solutions reside at non-ground low energy eigenstates. Whereas typical AQC suffers poor time-scaling due to the inverse dependence on the minimum energy gap Δ, the quantum SVP framework presented here may have the potential to circumvent this limitation, which is a motivator for further theoretical analysis.

Specifically, in tackling SVP_α one searches for a vector $\mathbf{v}$ of length

$$\|\mathbf{v}\| < \alpha\lambda_1(\mathcal{L}), \tag{5.14}$$

where $\alpha = \mathrm{poly}(N)$, implying polynomially many excitations are tolerable. Analysis of this more relaxed version of the adiabatic algorithm requires significant generalization from the

presently available results, which pertain to simply remaining in the instantaneous eigenstate with high probability. As shown here, such algorithms have the potential to provide significant algorithmic speedup. These theoretical results would find application not just to post-quantum cryptography, but to a wide plethora of approximate optimization problems.

One of the main takeaways from the analysis of Ham and Bin is to keep an open mind when considering strategies to deal with NISQ limitations. Originally only Bin was proposed for in-depth analysis due to its more efficient spatial requirements, but ultimately it was found that Ham was more robust to noise and that its $O(N/\log N)$ overhead in qubit count (versus Bin) was not actually too problematic.

Chapter 6

Expectation approximation for hard integer value problems

6.1 Introduction

While quantum computers theoretically should eventually be able to solve problems which are conjectured to remain intractable on classical computers, fault tolerant large scale quantum devices remain beyond practical reach for now. Between this era and the present lies the NISQ era [Pre18, BCLK+21], during which simple quantum algorithms can be implemented on small quantum chips, allowing for high noise levels, idling errors, and low-fidelity gates and qubits. The result is that research in the near-mid term must address a different set of challenges to those challenges of the fully quantum future. These challenges include identifying good candidates for quantum advantage on imperfect devices, engineering efficient error correcting codes, and devising strategies to lighten the burden of classical quantum simulation and optimization. One promising avenue of research for NISQ era quantum computation is into the quantum approximate optimization algorithm (QAOA) [FGG14], which is a coarse approximation to the adiabatic quantum algorithm. This chapter describes the findings of [JMLM21], and in it I present a low depth QAOA circuit to solve the shortest vector problem (SVP), and via analytical means, provide efficient approximations to the expectation of such a circuit and demonstrate

how the principles of this work can be generalized to a variety of other problems of real world significance.

6.2 QAOA for SVP

QAOA can be thought of as a member of the family of variational quantum eigensolvers (VQE), in that it consists of a parametrized ansatz $|\psi(\theta)\rangle$ for which the aim is to minimize (or without loss of generalization, maximize) some cost function $C(\theta)$, before running the final optimized algorithm with the ultimate choice of θ.

The adiabatic quantum algorithm, from which QAOA derives, is defined by the continuous interpolation between a driver Hamiltonian H_D responsible for exploring the solution space, and a problem Hamiltonian H_P into which the computational problem is encoded

$$H_t = (1 - f(t))H_D + f(t)H_P, \tag{6.1}$$

where $t \in [0, T]$, and f is subject to the constraints

$$\begin{aligned} f(0) &= 0, \\ f(T) &= 1. \end{aligned} \tag{6.2}$$

The general QAOA emulates this continuous evolution in a discrete manner by alternately applying H_D and H_P for p steps

$$|\psi(\theta)\rangle = e^{-i\beta_p H_D} e^{-i\gamma_p H_P} \dots e^{-i\beta_1 H_D} e^{-i\gamma_1 H_P} |\psi_0\rangle , \tag{6.3}$$

where the parameter space to optimize over is $\theta = (\beta_1, \dots, \beta_p, \gamma_1, \dots, \gamma_p)$. In the limit that $p \to \infty$, the algorithm approaches the adiabatic quantum algorithm and θ can be derived from first principles so that $|\psi(\theta)\rangle$ approximates the ground state of H_P to within an arbitrarily small error. However it is the low depth variants that are of most interest in the NISQ era, with $p = O(\log n)$ for systems of n qubits with locally connected problem Hamiltonians [FGG20],

and $p = O(1)$ for systems with fully connected Hamiltonians. In these cases $|\psi(\theta)\rangle$ gives only a rough approximation to the ground state of H_P, and furthermore depends sensitively on θ, hence requiring optimization via the VQE framework. In this chapter the problem in question is SVP, and the corresponding problem Hamiltonian is fully connected, hence it is a strong candidate to analyze via QAOA's lowest depth variant, with $p = 1$. The $p = 1$ variant of QAOA is the most widely studied (due to the analytical difficulties presented by circuits of higher depth) and has been shown to be capable of impressive speed-ups versus classical algorithms in some cases [FGG15, FGGZ19].

6.2.1 Parameter and Hamiltonian specification

The qubit layout is in the form of a two-dimensional Ising lattice of qubits, of dimensions $N \times k$ where N is fixed by the mathematical lattice dimension of the problem instance, and k is the number of qubits per qudit, chosen by the practitioner. Selection of k can be picked according to the analytical bounds derived in Theorem 3.1, or picked otherwise to improve results as was done in Chapter 5. The final measurement of the Ising lattice, once transformed to a $\{0, 1\}$ basis would look like Fig. 3.1

The driver and problem Hamiltonians are identical to those used in Chapter 5. The local driver Hamiltonian consists of Pauli-X operators on each qubit

$$H_D = \sum_j X_j. \tag{6.4}$$

The problem Hamiltonian takes the form

$$H_P = \sum_{i,j} \hat{Q}_i \hat{Q}_j \mathbf{G}_{ij}, \tag{6.5}$$

where $\hat{Q}_i$ is the qudit operator which outputs the integer coefficient for the i^{th} basis vector. The qudits are defined according to the binary encoding of Chapter 3, and are identical to those

applied in the Bin algorithm of Chapter 5. I define the qudit operators as

$$\hat{Q}_i = \sum_{p=0}^{k} 2^{p-1} \hat{Z}_{ip} + \frac{1}{2}\mathbb{1}, \tag{6.6}$$

meaning that the full problem Hamiltonian can be expressed in Z and ZZ operators on the two-dimensional Ising lattice of qubits as

$$H_P = \frac{1}{4} \sum_{ij} \mathbf{G}_{ij} \Big[\sum_{p,q}^{k} 2^{p+q} Z_{ip} Z_{jq} + \sum_{p=0}^{k} 2^{p} (Z_{ip} + Z_{jp}) + \mathbb{1} \Big], \tag{6.7}$$

where the indices (i, p) give the coordinates of the qubit in the two-dimensional Ising lattice. Index i specifies the basis vector, and p specifies the significance of the qubit.

The lowest depth variant was analyzed meaning the driver and problem Hamiltonians are applied once each, and the Rabi angle of the driver Hamiltonian was fixed to be $\beta = \pi/4$, a choice which simplifies trigonometric analysis and has been used elsewhere for lowest depth QAOA to obtain algorithmic speed-up over classical algorithms [FGG15]. Consequently only one parameter remains over which to optimize, which is the Rabi angle of the problem Hamiltonian γ. The input state $|\psi_0\rangle$ is taken to be the equal superposition over all feasible solutions, which can be implemented via a single Hadamard gate on each qubit, and results are interpreted in the $\{0, 1\}$ basis which corresponds to the measurement operator

$$O = \frac{(\mathbb{1} - Z)}{2}, \tag{6.8}$$

which, without loss of generality can be performed via a single Z gate to give a Pauli-Z eigenstate in $\{-1, 1\}$ which can be linearly transformed to $\{0, 1\}$. Summarizing the above, the algorithm under consideration can be expressed in the following three steps:

1. Initialise $|\psi_0\rangle = \otimes^{Nk} |+\rangle$

2. Apply unitaries to get final state

$$|\psi(\gamma)\rangle = e^{\frac{-i\pi}{4} H_D} e^{-i\gamma H_P} |\psi_0\rangle \tag{6.9}$$

3. Measure in the Pauli-Z eigenbasis on each qubit.

Fig. 6.1 visualizes a sample QAOA SVP circuit as shown using the Cirq Python package, for a two-dimensional lattice, with two qubits-per-qudit (in total, a four qubit system), which corresponds to the algorithm just described.

6.3 Expectation evaluation

As a cost function, the expectation μ of H_P was used, which is equivalent to the mean Euclidean length squared of all the vectors returned when $|\psi\rangle$ is measured with respect to H_P, given by

$$\mu = \langle\psi| H_P |\psi\rangle . \tag{6.10}$$

Applying this to the problem Hamiltonian in full, of Eq. (6.7), I can express this as

$$\mu = \frac{1}{4}\sum_{ij}^{N} \mathbf{G_{ij}} \left[1 + \sum_{p,q}^{k} 2^{p+q} \Gamma_{ijpq} + \sum_{p=0}^{k} 2^{p} \Big(\Omega_{ip} + \Omega_{jp} \Big) \right], \tag{6.11}$$

in terms of expectation values $\Gamma_{ijpq} = \langle\psi| Z_{ip} Z_{jq} |\psi\rangle$, and $\Omega_{ip} = \langle\psi| Z_{ip} |\psi\rangle$ of the two-body interaction and single-qubit energy terms of H_P. In this section I give explicit formulae for the dependencies of Γ, Ω on $-\gamma$ with the specific choice $\beta = \pi/4$. The formulae are given for $-\gamma$ to reduce the number of minus signs later in the analysis. Due to the common occurrence of such trigonometric terms, I denote the shorthand

$$\begin{aligned} \mathcal{S}_{ip} &= \sin\left(2^{p}\gamma \sum_{v}^{N} \mathbf{G_{iv}} \right) = \sin\left(2^{p}\gamma \sum_{v}^{N} \mathbf{G_{vi}} \right), \\ \mathcal{C}_{ip} &= \cos\left(2^{p}\gamma \sum_{u}^{N} \mathbf{G_{iv}} \right) = \cos\left(2^{p}\gamma \sum_{u}^{N} \mathbf{G_{vi}} \right). \end{aligned} \tag{6.12}$$

For a particular SVP instance, H_P is determined by the Gram matrix $\mathbf{G}$ and the number of qubits per qudit k, as seen in Eq. (6.7).

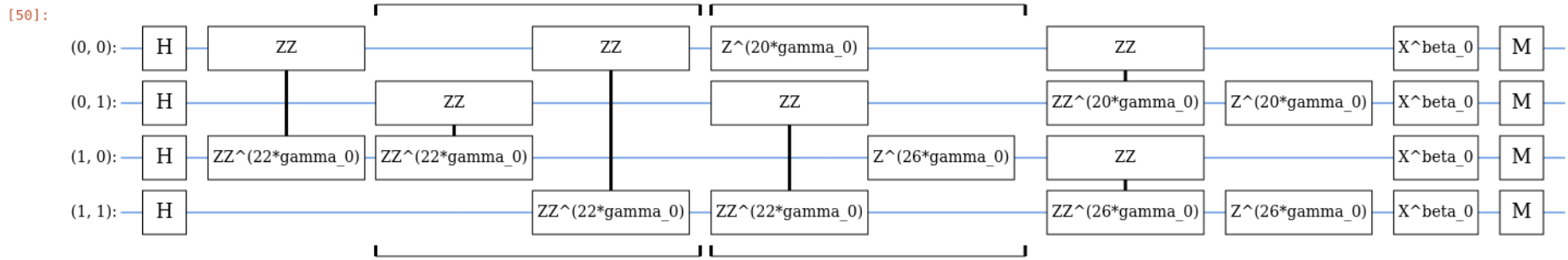

Figure 6.1: Example minimal depth QAOA circuit for a lattice $\mathcal{L}(\mathbf{B})$ with $\mathbf{B} = [[2, 4], [1, 5]]$, and corresponding Gram matrix $\mathbf{G} = \mathbf{B}\mathbf{B}^T = [[20, 22], [22, 26]]$. Lattice dimension is $N = 2$, qudit size is $k = 2$, the Hadamard gates, denoted H, initialise $|\psi_0\rangle$ to be the equal superposition over all feasible solutions, and measurement is denoted M. The circuit was generated using the Cirq application programme interface within a Jupyter notebook. Parameters γ, β are shown by the sympy symbols 'gamma0' and 'beta0'

6.3.1 Single-qubit operator expectation

I calculated the single-qubit operator expectation to be

$$\Omega_{ip} = \frac{-\mathcal{S}_{ip}}{\cos(2^{2p}\gamma \mathbf{G_{ii}})} \prod_{v=1}^{N} \prod_{s=0}^{k} \cos\left(\mathbf{G_{iv}} 2^{p+s} \gamma\right) , \tag{6.13}$$

where indices (i, p) describe the location of the qubit, with i denoting the relevant basis vector, and p denoting significance of the qubit within the qudit $\hat{Q}_i$ as shown in Eq. (6.6).

The expectation of Z_{ip} with respect to the final state $|\psi\rangle$ is

$$\Omega_{ip} = \langle \psi_0 | \, e^{-i\gamma H_P} e^{i\beta H_D} Z_{ip} e^{-i\beta H_D} e^{i\gamma H_P} \, |\psi_0\rangle \, , \tag{6.14}$$

which is the standard way to indicate the expectation of an operator with respect to an initial state which has been evolved by unitary H_P for time $-\gamma$, then by unitary H_D for time β. Here $|\psi\rangle$ is expressed in the form of Eq. (6.9) (but with $-\gamma$ instead of γ). In the following I derive a formula for Ω_{ip} by applying the standard linear algebra and trigonometric techniques associated with Bloch sphere rotations, which are induced by the unitaries H_D and H_P. Conjugation of Z_{ip} with the unitary induced by H_D and $\beta = \pi/4$ transforms Z_{ip} to Y_{ip}. After conjugating Y_{ip} with the single-qubit part of H_P, the transformed operators of Ω_{ip} can be written

$$e^{-i\gamma H_P} e^{i\beta H_D} Z_{ip} e^{-i\beta H_D} e^{i\gamma H_P} = \exp\left(\frac{-i\gamma}{4}\Delta_{ip}\right) \exp\left(\frac{-i\gamma}{4}\Theta_{ip}\right) Y_{ip} \exp\left(\frac{i\gamma}{4}\Theta_{ip}\right) \exp\left(\frac{i\gamma}{4}\Delta_{ip}\right), \tag{6.15}$$

where Θ and Δ are shorthand I have introduced, given by

$$\begin{aligned} \Theta_{ip} &= \sum_{u,v}^{N} \mathbf{G_{uv}} \sum_{q=0}^{k} 2^q (Z_{qu} + Z_{qv}), \\ \Delta_{ip} &= 2 \sum_{v=1}^{N} \sum_{s=0}^{k} \mathbf{G_{iv}} 2^{p+s} Z_{ip} Z_{vs}. \end{aligned} \tag{6.16}$$

Removing parts of Θ_{ip} which commute with Y_{ip} enables the substitution of Θ with Θ', which is

$$\Theta'_{ip} = 2^{p+1} \sum_{u=1}^{N} \mathbf{G_{iu}} Z_{ip}. \tag{6.17}$$

Replacing Θ' with Θ in Eq. (6.15) and applying the transformation $R_Z(\Theta')Y R_Z(\Theta')^\dagger$ (with R_Z denoting the Bloch sphere rotation along the Z axis) leaves

$$e^{-i\gamma H_P} e^{i\beta H_D} Z_{ip} e^{-i\beta H_D} e^{i\gamma H_P} = \exp\left(\frac{-i\gamma}{4}\Delta_{ip}\right)\left[\mathcal{C}_{ip} Y_{ip} - \mathcal{S}_{ip} X_{ip}\right] \exp\left(\frac{i\gamma}{4}\Delta_{ip}\right). \tag{6.18}$$

The final operation to perform is to conjugate Eq. (6.18) with the two-qubit gates of H_P. After conjugation with initial state $|\psi_0\rangle$, which is a Pauli-X eigenstate, the only nonzero results derive from tensor products of X operators. Accordingly, one can see that the Y_{ip} term of Eq. (6.18) disappears. The remaining nonzero part of Eq. (6.14) is

$$\Omega_{ip} = -\mathcal{S}_{ip} \langle\psi_0| \exp\left(\frac{-i\gamma}{4}\Delta_{ip}\right) X_{ip} \exp\left(\frac{i\gamma}{4}\Delta_{ip}\right) |\psi_0\rangle\,, \tag{6.19}$$

Using the relation $\langle\psi_0| \otimes_t X_t |\psi_0\rangle = 1$, Ω_{ip} evaluates to the expression given in Eq. (6.13), with the denominator deriving from the fact that there is no $Z_{ip}Z_{ip}$ operator (because $Z_{ip}Z_{ip} = \mathbb{1}_{ip}$) thus it should not be included in the product term.

6.3.2 Two-qubit operator expectation

The expectation of the two-qubit operators Γ_{ijpq} is the expectation of the two-qubit $Z_{ip}Z_{jq}$ interaction operator with respect to the evolved state $|\psi\rangle$, which I evaluated to be

$$\begin{aligned}
\Gamma_{ijpq} &= \zeta_{ijpq}\left[\mathcal{S}_{ip}\mathcal{S}_{jq}\left(\sum_{m=2x}^{Nk-2}\sum_{g}^{\binom{Nk-2}{m}} \chi_{S_g}\right) + \mathcal{C}_{ip}\mathcal{C}_{jq}\left(\sum_{m=2x+1}^{Nk-2}\sum_{g}^{\binom{Nk-2}{m}} \chi_{S_g}\right)\right], \\
\zeta_{ijpq} &= \frac{\prod_{v=1}^{N}\prod_{r=0}^{k}\cos(2^{p+r}\gamma\mathbf{G_{iv}})\cos(2^{q+r}\gamma\mathbf{G_{vj}})}{\cos(2^{2p}\gamma\mathbf{G_{ii}})\cos(2^{2q}\gamma\mathbf{G_{jj}})\cos^2(2^{p+q}\gamma\mathbf{G_{ij}})}.
\end{aligned} \tag{6.20}$$

Function χ will be discussed in the following, and depends on a series of combinatorial arguments. The indices (i, j, p, q) specify a qubit pair, with one qubit indexed by (i, p) and the other by (j, q). In the construction of the QAOA for SVP there are Nk qubits in the Ising qubit lattice, which is made of N qudits, each of which is comprised of k qubits. In the formula for Γ in Eq. (6.20), S denotes a subset of m qubits, chosen without replacement, from the $Nk-2$ remaining qubits (i.e. *not* qubits $(i, p), (j, q)$, of which there are $\binom{Nk-2}{m}$ distinct subsets, indexed by the subscript g in S_g). Write the qubits of a subset S of order m as

$$S = \{(t_1, w_1), \ldots, (t_m, w_m)\}. \tag{6.21}$$

Then χ_S is given in Eq. (6.25) and is explained further later in the section. In the formula for Γ in Eq. (6.20), the summations over χ can be addressed in two parts. The first part, with prefactor $\mathcal{S}_{ip}\mathcal{S}_{jq}$ is a summation of χ_S for all (nonempty) subsets S of qubits *other than* $(i, p), (j, q)$ containing an even number of qubits. The second part, with prefactor $\mathcal{C}_{ip}\mathcal{C}_{jq}$ is a summation of χ for all subsets S of qubits (also not including $(i, p), (j, q)$) containing an odd number of qubits. For a given m, there are $\binom{Nk-2}{m}$ distinct subsets of m qubits, indexed by subscript g. Considering both parts together (m odd and m even), there are, in total, $2^{Nk-2}-1$ terms. The exponential increase in computation is the main reason for computational difficulty in classically evaluating μ, and even the approximators μ_A which are outlined in the next section.

To gain an expression for Γ in Eq. (6.20), the expectation of $Z_{ip}Z_{jq}$ must be evaluated with respect to the evolved state $|\psi\rangle$ (again with $-\gamma$ instead of γ)

$$\Gamma_{ijpq} = \langle\psi_0| e^{-i\gamma H_P} e^{i\beta H_D} Z_{ip}Z_{jq} e^{-i\beta H_D} e^{i\gamma H_P} |\psi_0\rangle . \tag{6.22}$$

The driver Hamiltonian H_D, applied for time $\beta = \pi/4$ transforms $Z_{ip}Z_{jq}$ into $Y_{ip}Y_{jq}$. The next part is to conjugate the $Y_{ip}Y_{jq}$ operators with the single-qubit part of H_P, given by Θ in Eq.

(6.16). Consequently, the operators inside the expectation of Eq. (6.20) are

$$e^{-i\gamma H_P}e^{i\beta H_D}Z_{ip}Z_{jq}e^{-i\beta H_D}e^{i\gamma H_P} = \exp\left(\frac{-i\gamma}{4}(\Delta_{jq}+\Delta_{ip})\right)\exp\left(\frac{-i\gamma}{4}(\Theta_{jq}+\Theta_{ip})\right)Y_{ip}Y_{jq} \\ \exp\left(\frac{i\gamma}{4}(\Theta_{ip}+\Theta_{jq})\right)\exp\left(\frac{i\gamma}{4}(\Delta_{ip}+\Delta_{jq})\right). \quad (6.23)$$

Applying Bloch sphere rotations induced by Θ, this leaves

$$e^{-i\gamma H_P}e^{i\beta H_D}Z_{ip}Z_{jq}e^{-i\beta H_D}e^{i\gamma H_P} = \exp\left(\frac{-i\gamma}{4}(\Delta_{jq}+\Delta_{ip})\right)\Big[\mathcal{C}_{ip}\mathcal{C}_{jq}Y_{ip}Y_{jq} \\ -\mathcal{S}_{ip}\mathcal{C}_{jq}X_{ip}Y_{jq}-\mathcal{C}_{ip}\mathcal{S}_{jq}Y_{ip}X_{jq}+\mathcal{S}_{ip}\mathcal{S}_{jq}X_{ip}X_{jq}\Big]\exp\left(\frac{i\gamma}{4}(\Delta_{jq}+\Delta_{ip})\right). \quad (6.24)$$

The only terms to consider in Eq. (6.24) are those that do not go to zero after conjugation by the two-qubit part of H_P (induced by δ) when measured in the H_P basis. Due to the fact that $|\psi_0\rangle$ is an X_l eigenstate of eigenvalue 1, and $\langle\psi_0|Y_l|\psi_0\rangle, \langle\psi_0|Z_l|\psi_0\rangle = 0$ for all l, the only surviving terms are products of X_l's with no trailing Y_l, Z_l.

Next is to analyze how two-qubit interaction ZZ gates flip Y operators to X and vice versa. As all-X tensor products are necessary, each operator must have been flipped the correct number of times modulo 2, and furthermore, there must be no trailing Z, Y operators in the final term. Applying all two-qubit ZZ gates of H_P to each of the terms in Eq. (6.24) yields the following results case by case, where a single index in (a, b, c, d) is used to distinguish qubits:

Case 1 - X_aX_b Only XX terms give a nonzero expectation after conjugation with the initial state $|\psi_0\rangle$, so after applying all two-qubit interaction ZZ gates of H_P, one should only consider XX operators with no trailing Z, Y. After applying all the ZZ gates, only one XX term remains with a coefficient which is a product of all cos's. Applying Z_aZ_c and Z_bZ_c results in a Y_aY_b term, then applying Z_aZ_d and Z_bZ_d gives another X_aX_b term with no trailing Y, Z operators, and has a coefficient which is a product of sin's. After applying all other two-qubit gates on this XX operator, a nonzero term consisting of four sin's and $2(Nk-4)$ cos's is found. Qubits c and d were used to flip the X_aX_b to Y_aY_b then back again to X_aX_b, and there are $\binom{Nk-2}{2}$ ways to pick qubits c, d. This can be repeated iteratively, as the same result holds when

using qubits c, d, e, f to perform the flipping, so there is a corresponding $X_a X_b$ term (with no trailing Y, Z) for each distinct combination of ω qubits which do not include qubits a and b, where ω is even. This means there are $\binom{Nk-2}{\omega}$ XX terms for each even $\omega \leq Nk-2$.

Cases 2,3 - $X_a Y_b, Y_a X_b$ These return no nonzero terms because only one of X_a, Y_b needs to be flipped, but that would result in a trailing Z, therefore no $X_a X_b$ terms with no trailing Y, Z are left within the final expression.

Case 4 - $Y_a Y_b$ This case utilizes the same reasoning as in *Case 1*. The only difference is that instead of requiring even numbers of qubits other than a, b, an odd number is required, so for $\omega = 1, 3, \ldots$ there are $\binom{Nk-2}{\omega}$ nonzero terms, which are each composed of a product of $2(Nk-2) - 2\omega$ cos terms, and 2ω sin terms.

Using the above rationale, as an illustrative example, deduce the expectation of an XX operator conjugated by the two-qubit part of H_P. For reference, this is the same as evaluating the final term of the central square brackets in Eq. (6.24), but without the trigonometric prefactors $\mathcal{S}_{ip}, \mathcal{S}_{jq}$. The system is fully connected, and the only two-qubit gates to adjust for are $Z_{ip}Z_{ip}, Z_{jq}Z_{jq}$ which are both the identity, and $Z_{ip}Z_{jq}$, which commutes with $X_{ip}X_{jq}$. Overall, there are 2^{Nk-2} nonzero terms, and the term where the $X_{ip}X_{jq}$ is not flipped at all has value ζ_{ijpq} from Eq. (6.20), where the denominator is there to cancel the gates (which are identities) $Z_{ip}Z_{ip}, Z_{jq}Z_{jq}, Z_{ip}Z_{jq}$, from the product term. The value ζ is utilized as a stem function for the remaining nonzero terms, which are a product over the same qubit indices, and with the same angles, but with sin's instead of cos's.

Next update ζ for a specified combination of qubits responsible for the operator flips (from XX to YY and back again). Then the nonzero part of the expectation due to this combination of operator flips is $\zeta_{ijpq}\chi_S$, where S is given by Eq. (6.21) and I define χ_S as

$$\chi_S = \prod_{h=1}^{d} \tan\left(2^{p+w_h}\gamma \mathbf{G_{it_h}}\right) \tan\left(2^{q+w_h}\gamma \mathbf{G_{t_h j}}\right). \tag{6.25}$$

Considering all possible combinations of qubits, and applying the rationale of *Cases 1, 4* for Eq. (6.24), one arrives at the formula for Γ in Eq. (6.20).

6.3.3 Expectation simulation

Having evaluated formulae for the expectation of the minimal depth QAOA algorithm for SVP for a given choice of parameters, the logical next step is to ascertain whether such information could be of use. To this end, numerical simulations of the expectation were performed on a set of two-dimensional lattices, with varying numbers of qubits-per-qudit k, and these results are shown in Fig. 6.2.

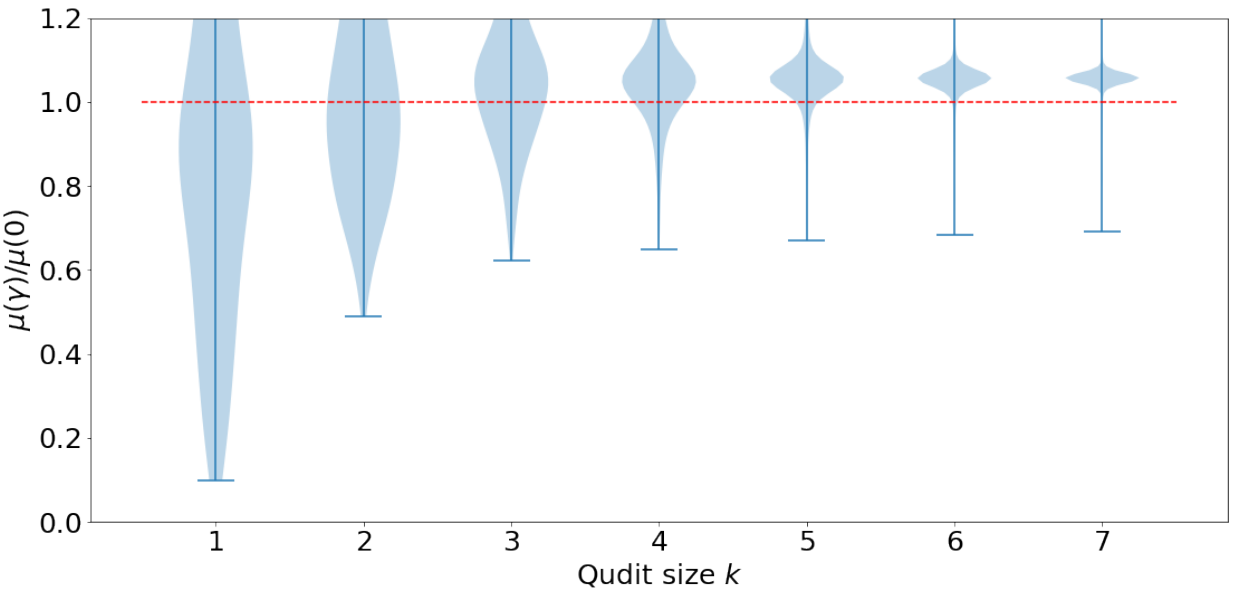

Figure 6.2: Each violin visualizes the distribution of $\mu(\gamma)/\mu(0)$ over varying γ taken from 45 two-dimensional lattices, and simulated numerically via TensorFlow Quantum [BVM+20]. As k increases system size grows, and the distributions become increasingly concentrated around the median. The centers of the violins lie above the classical baseline (red dashed line) for $k \geq 3$. The lower end of the distributions, however extend substantially lower than this baseline, indicative of a clear improvement over the naive classical sampler for well chosen but rare values of γ.

The violin plots of Fig. 6.2 show the probability density of $\mu(\gamma)/\mu(0)$ for uniformly distributed γ over the range $[0, \pi]$ for two-dimensional lattices and qudit sizes $k = 1$ to $k = 7$ giving a total system size Nk running up to fourteen qubits. Each violin shows results averaged over 45 lattice instances, and the expectation was evaluated at the same $1,000$ values of γ for each one. The importance of $\mu(0)$ is that the problem Hamiltonian is not applied to the system at all, and so the ultimate distribution of $|\psi\rangle$ with respect to H_P is simply the uniform

random distribution over all possible vectors in the solution space, which can be thought of as a naive classical uniform random sampler for the coefficient vector, thus providing a natural benchmark with which to compare the results of the QAOA. This classical baseline is depicted by the red dashed line at 1.0, and the reason all results are divided through by $\mu(0)$ is that the coefficients of different lattice bases mean that the absolute eigenvalues of H_P vary widely between instances, and so normalizing by dividing through by $\mu(0)$ allows for easier comparison of the algorithmic performance across lattice instances.

The distributions become increasingly narrowly concentrated around the median (fattest part of the violin) for higher values of k, and furthermore, the median lies *above* the classical baseline, meaning that for larger system sizes, picking some random γ with which to run the algorithm defined above offers no improvement over even a naive classical sampler. One can see, however, that the lower end of the distributions lie consistently and significantly below the baseline, and level off at ≈ 0.7 times the value of the classical sampler. This means that for well chosen values of γ, the low depth quantum algorithm can return vectors with squared Euclidean norm on average $\approx 30\%$ shorter than is possible from random guessing, and that this effect appears to hold well even for higher qudit sizes. This means that the simple and low depth quantum algorithm already defined has strong potential, but in order to realise this potential it is necessary to identify a good value of γ with which to run the algorithm, which is a problem (parameter optimization) broadly addressed by the VQE framework. The following sections outline techniques for speeding up identification of appropriate γ.

6.4 Expectation approximators

Due to the analytical evaluation of $\mu(\gamma)$ in the previous section and the explicit formulae for Γ, Ω, it is possible to choose terms from μ which are likely to make a more significant contribution than others. If successful, this would enable estimation of $\mu(\gamma)$ without full computation

of $|\psi\rangle$ or even μ. Let μ be approximated

$$\mu \simeq \mu_A = \frac{1}{4}\sum_{ij}^{N} \mathbf{G_{ij}} \sum_{k-A<p,q\leq k} 2^{p+q}\Gamma_{ijpq} \ , \tag{6.26}$$

where $A \gtrsim 0$ is the order of approximation. The coarsest approximation is for $A = 1$, and this just considers the expectation of the two-qubit operators between the most significant qubits in each qubit, i.e. the fully connected graph on N qubits out of a system of Nk qubits. The most accurate approximation would be $A = k$ and in this case all two-qubit operators of H_P are considered, and only the Ω terms are dropped from the exact expectation, though the Ω contribution to μ is negligible to evaluate by comparison with the two-qubit operators.

The choice of which terms to approximate μ by is driven primarily by the integer prefactors, due to the observation that the binary encoding of qudits leads to 2^{p+q} multipliers on the expectation of individual operators Γ and Ω. The exponential nature of such prefactors means that the interaction between the most significant qubits is $\approx 2^{2k}$ times as strong as the interactions between the least significant qubits, yet evaluation of each Γ incurs roughly the same computational cost. The Ω are dropped due to the fact that even for the most significant qubits the prefactors of Ω are $\approx 2^k$ which is quadratically smaller than the dominant terms of μ_A.

Due to the comparable cost in classically evaluating each Γ, and the asymptotically negligible cost of evaluating Ω, the efficiency saving of μ_A is roughly proportional to the number of Γ to compute in μ_A versus the number of Γ to compute in μ. Due to the fact that μ_A considers the complete graph on the A most significant qubits this involves $NA(NA-1)/2$ terms of Γ, and for full μ there are $Nk(Nk-1)/2$ terms of Γ to evaluate, leading to a quadratic efficiency saving of $O(A^2/k^2)$.

6.4.1 Hamiltonian formulation of approximators

One can equivalently formulate the approximators μ_A directly as a truncation from the problem Hamiltonian H_P to approximator Hamiltonians H_A. This follows from the expression

$$\mu_A = \langle\psi| H_A |\psi\rangle , \tag{6.27}$$

where H_A consists of the fully connected graph of two-qubit ZZ interaction gates of H_P on the A most significant qubits, analogous to the formulation of μ_A in the previous part. The truncated Hamiltonians take a similar form to Eq. (6.26)

$$H_A = \frac{1}{4}\sum_{ij} \mathbf{G_{ij}} \sum_{k-A<p,q\leq k}^{k} 2^{p+q} Z_{ip} Z_{jq}. \tag{6.28}$$

In this expression for μ_A it should be noted that the expectation is taken of an approximate Hamiltonian, but the full H_P is used to evolve the system, and so $|\psi\rangle$ is in fact the exact final state. Using the H_A in the propagator to generate some approximate final state $|\psi_A\rangle$ would oversimplify the classical problem, and I confirmed this numerically with poor approximations to the exact expectation.

6.5 Results

Having defined the μ_A which intuitively should be good approximators to the expectation μ, it remains to observe how well they are able to approximate μ. A first and general measure of the closeness of the approximators to the exact expectation is to take the Pearson correlation [FPP10]

$$r_A = \frac{\mathrm{cov}(\mu_A, \mu)}{\sigma_{\mu_A}\sigma_\mu} , \tag{6.29}$$

where the covariance is $\mathrm{cov}(x, y) = \mathbb{E}(xy) - \mathbb{E}(\mathbb{E}(x)\,\mathbb{E}(y))$ and variance is $\sigma_x^2 = \mathrm{cov}(x, x)$. The average of $x(\gamma)$ for γ between 0 and π is denoted $\mathbb{E}(x)$.

In Fig. 6.3 one can see the Pearson correlation coefficients r_A averaged over 45 two-dimensional

lattices plotted for a range of different qudit sizes k. Approximating μ using only the most significant qubits of the system is quite rough, as r_1 (in red triangles) does not rise above around 0.6. Increasing the approximation to include the two most significant qubits, however, gives a marked improvement, as r_2 (in blue pluses) reaches a correlation of ≈ 0.9. The pattern continues, with each extra most significant qubit improving the approximation, and $r_{k/2}$ and r_k (in purple dashes and yellow diamonds respectively) converge rapidly, thereby guaranteeing a good approximation to μ using only a small fraction of the total number of qubits.

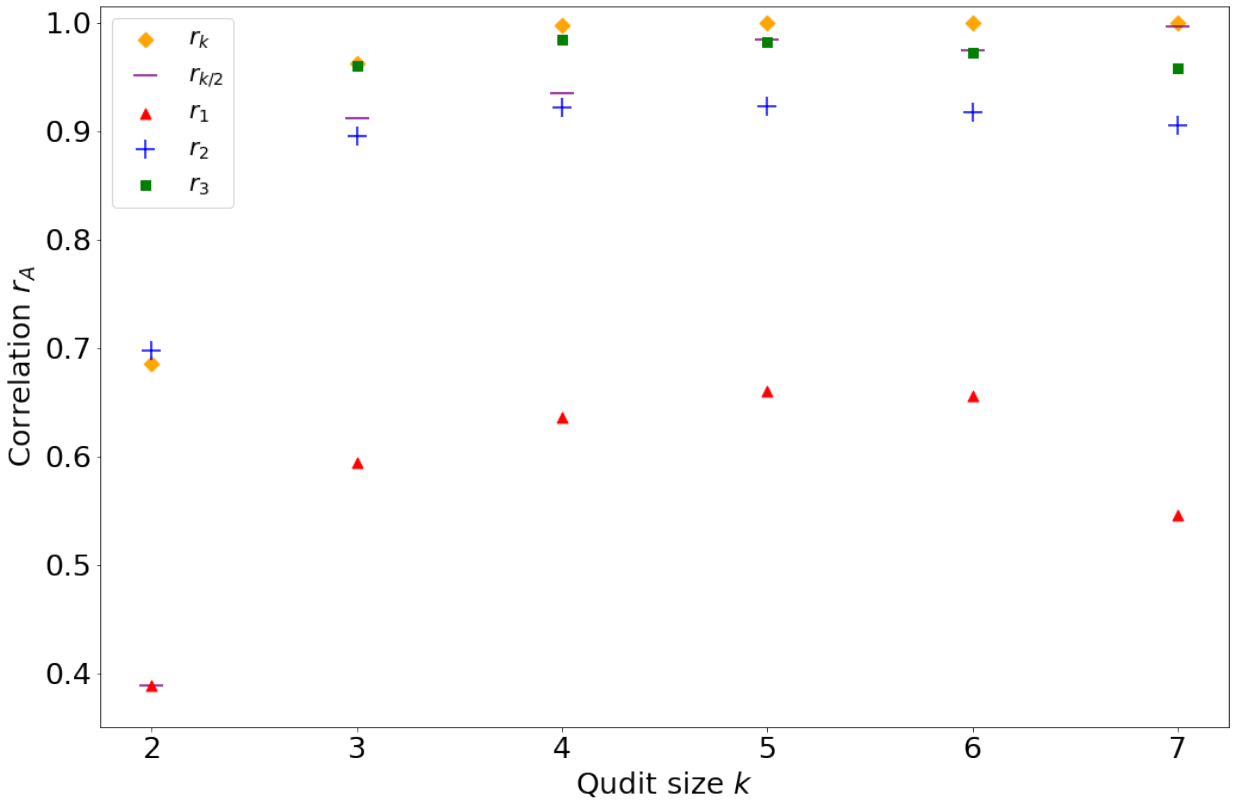

Figure 6.3: Pearson correlation coefficient r_A for $A = 1, 2, 3, k/2$ and k averaged over 45 two-dimensional lattices. Approximations μ_A using only a few most-significant qubits (small values of A) give an excellent approximation of μ, indicated by values of r_A close to 1.

While the correlations of Fig. 6.3 give a scalar indication of how well the series $\mu_A(\gamma)$ correlates with $\mu(\gamma)$, it aids understanding to visualize the relationship between the two. Fig. 6.4 shows, for $A = 1, 2, 3$ the distributions of errors between the the approximators and the exact expectation. The ratio μ_A/μ is shown, so better approximators would show a tightly concentrated distribution around 1.0. Qualitatively, it is clear that lower A lead to higher systematic errors

as evidenced by the distance of the distribution centers from 1.0, with $A = 1$ displaying a large 0.25 displacement from 1.0, and $A = 2, 3$ showing drastically reduced systematic errors of 7% and 2% respectively. The systematic error is a nuance not captured by the Pearson correlation coefficient r_A, but nevertheless is not a major concern as correcting for systematic errors in practice should be relatively straightforward.

More important is the shape of the distributions, and all three appear to broadly adhere to a normal distribution, though with very small secondary peaks shown for $A = 2, 3$. It is clear that $A = 1$ has the broadest distribution, with $A = 2$ having lower variance and $A = 3$ lower variance again. This is complemented by the increasing height of the central peaks for higher A. The implication of the width of the distribution (for example at $A = 3$) is that errors are less than a few percent on average for a given value of γ. This is across the entire range of γ, however, but the main data points of interest are the γ which give very low μ, so a more focused analysis is still required, despite the encouraging results displayed in Figs. 6.3, 6.4.

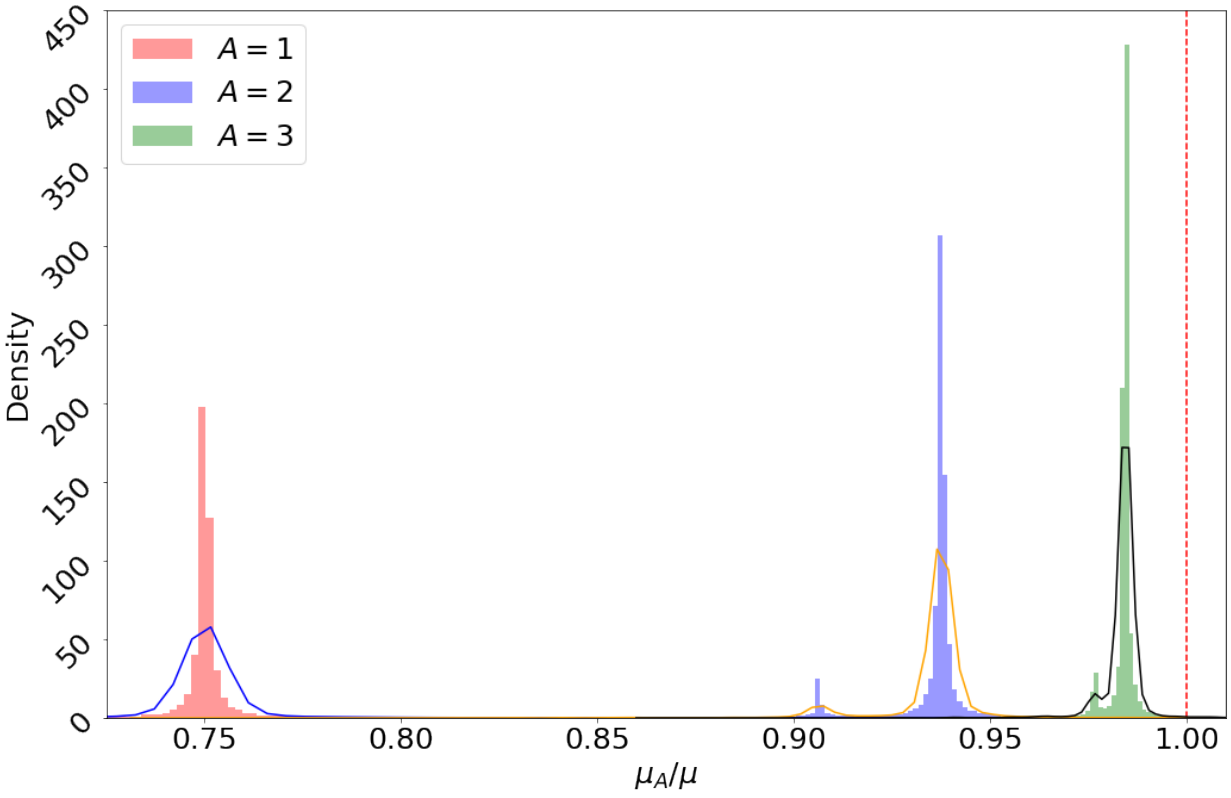

Figure 6.4: Distribution of the ratio between the approximated expectation and the exact expectation for a fixed γ. Low A gives a wider spread of errors and a large systematic error, whereas higher A give sharper peaks centred closer to the ideal value of 1.0

Strong correlations r_A and small errors μ_A/μ for small A imply that minimization of μ_A could be used to find values of γ that give low expectation values, and therefore which would be well suited to a low depth QAOA SVP circuit. For an approximator to be of use, however, it must do more than just correlate well with μ and exhibit small errors *on average*; the μ_A should also agree closely with μ on the minima of the parameter landscape – in this case the one-dimensional landscape over γ. This is because the minima are the most important part of the landscape for achieving success in the QAOA, yet as found from Fig. 6.2, the good minima exist for increasingly few γ as system size increases. Fig. 6.5 illustrates how well the μ_A agree on the location of the minima. Fig. 6.5 shows histograms of $\mu(\gamma_A)/\mu(\gamma_{opt})$, with γ_A being the γ that gives the lowest value of μ_A, and γ_{opt} is the γ which gives the minimum value of μ. Separate histograms are shown for μ_1, μ_2, μ_3, and each includes the results for all 45 lattice instances for $k = 5, 6, 7$ together. The reason $\mu(\gamma_A)/\mu(\gamma_{opt})$ is insightful to analyze is because it tells how well a γ obtained via classical optimization might perform when plugged into the full quantum algorithm. For a perfect approximator, a single bar at 1.0 would be seen, indicating that the approximately-optimized version of γ also gives the exact minimum over μ in all cases, and so the results are indistinguishable from exact optimization therefore meaning any savings from the approximator come at no cost in performance.

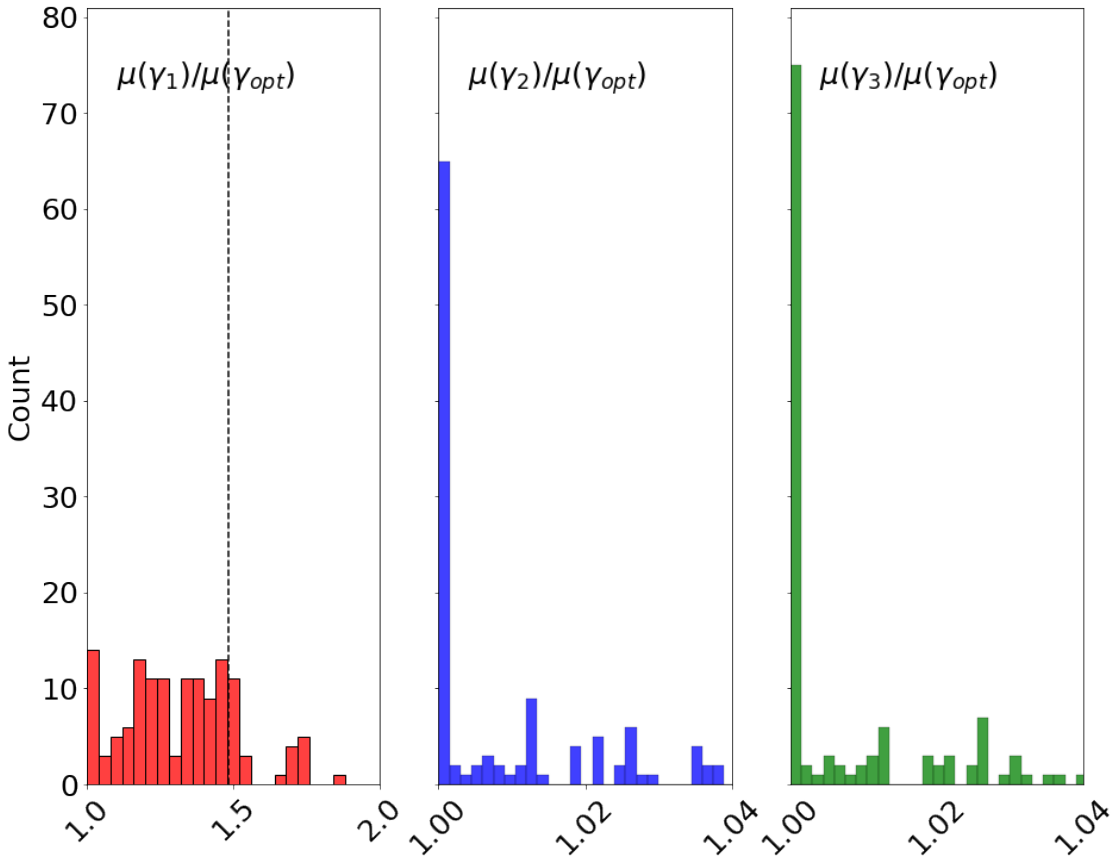

Figure 6.5: Histogram for the ratio $\mu(\gamma_A)/\mu(\gamma_{opt})$ where γ_A and γ_{opt} are the values of γ that minimze μ_A and μ respectively. Each distribution contains data points for 45 lattices with $k = 5, 6, 7$. The bin width is 0.04 for $A = 1$ and 1.5 10^{-3} for $A = 2$ and $A = 3$. The black dashed vertical in left hand plot denotes $\mu(0)/\mu(\gamma_{opt})$, but this value is too large to be shown for $A = 2, 3$. The actual minimum of μ is approximated increasingly well by γ_A with increasing value of A, and even coarse approximations ($A \ll k$) yield excellent results.

In Fig. 6.2 one notices that the classical baseline sits, on average, ≈ 1.5 times higher than the minimum recorded value of μ. This is demonstrated in the $A = 1$ subplot of Fig. 6.5 by the black dashed vertical line (not shown for $A = 2, 3$ subplots as 1.5 is too high to show). Even for $A = 1$ most of the density lies to the left of the black dashed vertical demonstrating that even the coarsest approximation helps to identify γ values which perform substantially better than the naive classical sampler. In the $A = 2, 3$ subplots the distance of $\mu(\gamma_A)/\mu(\gamma_{opt})$ from 1.0 is no more than a few percent, and the vast majority of probability density is much lower than this. The trade-off in performance due to optimizing an approximation μ_A versus the exact μ is

thus much smaller than benefit of using an exactly-optimized QAOA over the classical sampler.

6.6 Conclusion

The approximation techniques of this chapter have been demonstrated to be effective in the concrete case of low depth QAOA for the shortest vector problem, but in principle they can be applied to a wide variety of quantum algorithms. The only requirement is that such algorithms possess problem Hamiltonians of widely ranging interaction magnitudes, so that the smaller terms can be dropped without greatly affecting the resulting expectation. The most natural problems to generalize to are those of integer optimization, for example integer programming problems [PS98, WN99] and the knapsack problem [Mat96]. The reason for this is that any Hamiltonian that applies a binary qudit encoding will be appropriate for such approximators due to the exponentially increasing interaction prefactors, and similarly to the algorithm of this chapter, the problems can be expressed entirely by the qudit size k and matrix $\mathbf{G}$, which in the specific case of SVP is the Gram matrix of a lattice $\mathcal{L}(\mathbf{B})$.

Because unoptimized QAOA with low circuit depth fails to offer any improvement over even very basic classical samplers, but optimized versions can perform significantly better, the techniques of this chapter will help to bring low depth quantum algorithms closer to practicability. As the quantum computing community progresses through the NISQ era and increasingly lengthy gate sequences become possible, the optimization generalizes in a straightforward manner to one over higher-dimensional parameter spaces. Cost functions for such circuits are likely to fast become intractable to simulate classically even with such efficiency-saving approximations as presented here. The applicability of these techniques extend beyond classical simulation however, because they imply that the final state $|\psi\rangle$ of such algorithms could be measured with respect to H_A, as described in Section 6.4.1. Similarly to how the approximations reduce classical computation, they could in this way be used to reduce the gate count during measurement of the fully quantum algorithm.

Chapter 7

Conclusion

7.1 Summary of Thesis Achievements

In Chapter 2, I outlined the key concepts from both classical cryptography and quantum computing required in order to understand the contribution of later chapters. The subsequent material described a number of quantum algorithms for the shortest vector problem which explored a wide range of quantum hardware and device architecture, analyzing heuristic results from both simulation and quantum implementation along the way.

In Chapter 3, I presented a mapping from mathematical lattice vectors to a classical Hamiltonian H_P where the shortest nonzero vector occupied the first excited state. To find such a low energy state, H_P then required optimization with respect to an integer coefficient vector $\mathbf{x}$. Due to the qubit-based architecture of many quantum devices, I then reformulated this as an optimization over an array of binary variables which could be interpreted as the output of measurement of a qubit. Having defined these mappings, I provided a bounding of the search space for $\mathbf{x}$, which is necessary in order to guarantee the existence of the shortest vector at the first excited state of H_P. To prove that the spatial scaling of the algorithm was no larger than $O(N \log N)$, a weak form of the Minkowski bound (on the length of the shortest vector) was combined with the Hermite Normal Form basis of a lattice.

In Chapter 4, I described an SVP algorithm in optical lattices using an adiabatic-style transition of a bosonic gas from superfluid to Mott insulator, whence a vector of integers $\mathbf{x}$ is observed from the configuration of bosons in the optical lattice. To formulate H_P in this algorithm, the onsite energy at each site in the optical lattice, and the interaction energy between bosons were combined. When simulated numerically for low dimensional instances of SVP, a Goldilocks zone was identified which signified some intermediate range of time sweeps for which finding low-but-not-ground eigenstates of H_P was optimal. Sweeps which are too fast give a random assortment of vectors which is no better than guessing, but adiabatic sweeps return the trivial zero vector. This represents a significant advantage over adiabatic algorithms because physically achieving the long coherence times required for fully adiabatic quantum computation is a significant challenge, so an algorithm that achieves optimality at sub-adiabatic sweep lengths should make implementation much easier.

In Chapter 5, I presented two variants (based on different qudit definitions) of a quantum annealing algorithm for finding short vectors, using an Ising formulation of SVP. A major advantage of this representation is that many quantum devices are designed to handle quantum Ising optimization problems, including the D-Wave quantum annealer. The opportunity to investigate this further was seized upon, and consequently the first fully quantum cryptanalysis of lattice-based cryptography was prototyped. A number of challenges pertaining to quantum implementation presented: results were noisy due to imperfect hardware; a fully connected Ising graph had to be embedded in the Chimera topology of the D-Wave 2000Q quantum chip; sweep times were available only for $T \in [1\mu s, 1,000\mu s]$; and more. Nevertheless, interesting results were obtained which demonstrated that the less spatially efficient qudit definition Ham provided more robustness to noise, which is particularly useful for near-term implementation, and that qualitatively the quantum hardware did in fact effectively achieve an output of lower energy vectors with respect to some classical equivalent baseline samplers. These experimental results were contrasted with numerical simulations (in lower dimensions) which reproduced the Goldilocks zone first identified in Chapter 4, and illustrated the superiority of the spatially optimal qudit mapping Bin on ideal hardware, which is a long term scenario.

In Chapter 6, I described a low depth QAOA method for sampling short vectors. The advantage

of this algorithm is that it can be executed on circuit-model hardware using a series of single and two-qubit quantum gates. The reasons for picking QAOA were firstly that it is a natural progression from adiabatic-style algorithms, having been derived from the adiabatic algorithm via Trotterization. Secondly, QAOA's low depth variants are both analytically accessible, and have been demonstrated to offer algorithmic speed-up for certain problems [FGG15, FGGZ19]. Thirdly, QAOA fits into the category of variational quantum algorithms, which are thought to be prime candidates for achieving quantum advantage in the NISQ era. Initial numerical simulations showed that for unoptimized QAOA parameters, the lowest depth QAOA algorithm for SVP did not perform any better than a classical equivalent sampler. Furthermore, finding optimized parameters becomes increasingly difficult with system size. The simulations did reveal, however, that for a well chosen parameters, the algorithm in question could significantly outperform the classical equivalent, raising the question of how to find such a good choice of parameters. An analytical evaluation of the expectation of vector length output by the QAOA showed that the expectation (used as a loss function for optimization) could be efficiently approximated by a small number of terms, so long as the Hamiltonian had widely ranging terms. Demonstrating that such approximators give results extremely close to the exact expectation, I showed that such approximations have the potential to significantly increase efficiency of training variational quantum algorithms, and reduce the burden of final state measurement by reducing the number of measurement gates in a fully quantum instantiation.

7.2 Future Work

Perhaps the most pressing work that would strengthen the contributions of this thesis is more analytical results. A key difficulty in the study of adiabatic algorithms is that time scaling relies upon the minimum gap between the ground and first excited state of the intermediate Hamiltonian, which is in general a hard problem. This means that for the algorithms of Chapters 4, 5 time scaling is not yet known. In particular, while much focus is upon the time scaling required to achieve the ground state of a problem Hamiltonian, the algorithms presented above show that an analysis of time scaling to achieve $O(1)$ probability of achieving

an arbitrarily low eigenstate would also be extremely valuable, and potentially more useful for real-world problems. It is conceivable that modeling excitations and transitions between energy levels could be performed without detailed knowledge of the intermediate eigenspectra. If this were possible, it would unlock time scaling for the algorithms of Chapters 4, 5 for solving approximate SVP, which would be a huge milestone for quantum cryptanalysis.

Other theoretical work which would improve understanding of how well quantum devices could attack lattice cryptography would be to understand the distribution of the final state of the minimal depth QAOA described in Chapter 6. While the expectation is a well respected figure of merit for variational quantum algorithms due to its scalar description of the goodness of the final state, there are other metrics which potentially shed more light on the output. For example one could minimize the expectation by simply reducing the occurrence of very long vectors as opposed to (much more valuably) increasing the occurrence of very short vectors. One could look to optimize with respect to a different loss function, such as the Gibbs objective function [LFC+20] which gives greater weight to short vectors, or try to model the thermal distribution of eigenstates upon measurement.

On the implementation side, it would be interesting to repeat experiments from the D-Wave 2000Q with the D-Wave 5000Q. Its ability to run on higher dimensional lattice instances, as well as its increased connectivity (meaning less overhead due to qubit chains) and other hardware improvements would be likely to give a clearer picture of algorithmic potential than what is presented in Chapter 5. Other gate-based quantum hardware will hopefully soon be able to run the QAOA of Chapter 6, though the problem of long range bosonic interactions is likely to mean that the algorithm of Chapter 4 will be the last practicable implementation relating to this thesis.

The approximations of Chapter 6 enable useful efficiencies in evaluating the expectation of a quantum circuit for a certain set of Hamiltonians. Based on the analysis of Chapter 6 I believe they have the potential to improve training of variational quantum eigensolvers significantly at little cost to the accuracy of the optimized parameters. While actually performing such optimizations with commonly used machine learning outer loops, and higher dimensional pa-

rameter spaces was out of scope for this work, it would be a natural next step to show that such approximations can and do improve the variational quantum algorithm workflow.

Bibliography

[AAAS+19] Gorjan Alagic, Gorjan Alagic, Jacob Alperin-Sheriff, Daniel Apon, David Cooper, Quynh Dang, Yi-Kai Liu, Carl Miller, Dustin Moody, and Rene Peralta. *Status report on the first round of the NIST post-quantum cryptography standardization process*. US Department of Commerce, National Institute of Standards and Technology . . . , 2019.

[AAB+19] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando G S L Brandao, and David A Buell. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.

[AAKV01] Dorit Aharonov, Andris Ambainis, Julia Kempe, and Umesh Vazirani. Quantum walks on graphs. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 50–59, 2001.

[AASA+20] Gorjan Alagic, Jacob Alperin-Sheriff, Daniel Apon, David Cooper, Quynh Dang, John Kelsey, Yi-Kai Liu, Carl Miller, Dustin Moody, and Rene Peralta. Status report on the second round of the NIST post-quantum cryptography standardization process. *US Department of Commerce, NIST*, 2020.

[AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 284–293, 1997.

[AD17] Martin R Albrecht and Amit Deo. Large Modulus Ring-LWE $$\ge $$ Module-LWE. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 267–296. Springer, 2017.

[ADH+19] Martin R Albrecht, Léo Ducas, Gottfried Herold, Elena Kirshanova, Eamonn W Postlethwaite, and Marc Stevens. The general sieve kernel and new records in lattice reduction. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 717–746. Springer, 2019.

[ADRSD15] Divesh Aggarwal, Daniel Dadush, Oded Regev, and Noah Stephens-Davidowitz. Solving the shortest vector problem in 2n time using discrete Gaussian sampling. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 733–742, 2015.

[Ajt96] M. Ajtai. Generating hard instances of lattice problems (extended abstract). *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96*, pages 99–108, 1996.

[AKS01] Miklós Ajtai, Ravi Kumar, and Dandapani Sivakumar. A Sieve Algorithm for the Shortest Lattice Vector Problem. *Proc. 33rd ACM Symp. on Theory of Computing (STOC)*, 2146:601–610, 2001.

[ANS18] Yoshinori Aono, Phong Q. Nguyen, and Yixin Shen. Quantum lattice enumeration and tweaking discrete pruning. In *ASIACRYPT*, 2018.

[ASD17] Divesh Aggarwal and Noah Stephens-Davidowitz. Just Take the Average! An Embarrassingly Simple 2ˆn-Time Algorithm for SVP (and CVP). *arXiv preprint arXiv:1709.01535*, 2017.

[AvDK+04] Dorit Aharonov, Wim van Dam, Julia Kempe, Zeph Landau, Seth Lloyd, and Oded Regev. Adiabatic Quantum Computation is Equivalent to Standard Quantum Computation. *SIAM Journal of Computing*, 37(1):166–194, 2004.

[AVDK+08] Dorit Aharonov, Wim Van Dam, Julia Kempe, Zeph Landau, Seth Lloyd, and Oded Regev. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM review*, 50(4):755–787, 2008.

[BB20] Charles H Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. *arXiv preprint arXiv:2003.06557*, 2020.

[BBC+93] Charles H Bennett, Gilles Brassard, Claude Crépeau, Richard Jozsa, Asher Peres, and William K Wootters. Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels. *Physical review letters*, 70(13):1895, 1993.

[BBC+95] Adriano Barenco, Charles H Bennett, Richard Cleve, David P DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John A Smolin, and Harald Weinfurter. Elementary gates for quantum computation. *Physical review A*, 52(5):3457, 1995.

[BBD09] Daniel Bernstein, Johannes Buchmann, and Erik Dahmen. *Introduction to post-quantum cryptography*. Springer-Verlag, 2009.

[BCLK+21] Kishor Bharti, Alba Cervera-Lierta, Thi Ha Kyaw, Tobias Haug, Sumner Alperin-Lea, Abhinav Anand, Matthias Degroote, Hermanni Heimonen, Jakob S Kottmann, and Tim Menke. Noisy intermediate-scale quantum (NISQ) algorithms. *arXiv preprint arXiv:2101.08448*, 2021.

[BDEK04] C W J Beenakker, D P DiVincenzo, C Emary, and M Kindermann. Charge detection enables free-electron quantum computation. *Physical review letters*, 93(2):20501, 2004.

[Ben80] Paul Benioff. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. *Journal of statistical physics*, 22(5):563–591, 1980.

[BGM21] Sergey Bravyi, David Gosset, and Ramis Movassagh. Classical algorithms for quantum mean values. *Nature Physics*, 17(3):337–341, 2021.

[BKR16] Tomas Boothby, Andrew D King, and Aidan Roy. Fast clique minor generation in Chimera qubit connectivity graphs. *Quantum Information Processing*, 15(1):495–508, 2016.

[BKSW18] Zvika Brakerski, Elena Kirshanova, Damien Stehlé, and Weiqiang Wen. Learning with Errors and Extrapolated Dihedral Cosets. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10769 LNCS:702–727, 2018.

[BL17] Daniel J. Bernstein and Tanja Lange. Post-quantum cryptography. *Nature*, 549:188–194, 2017.

[BLP+13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 575–584, 2013.

[BVM+20] Michael Broughton, Guillaume Verdon, Trevor McCourt, Antonio J Martinez, Jae Hyeon Yoo, Sergei V Isakov, Philip Massey, Murphy Yuezhen Niu, Ramin Halavati, and Evan Peters. Tensorflow quantum: A software framework for quantum machine learning. *arXiv preprint arXiv:2003.02989*, 2020.

[CAD+20] David A Cooper, Daniel C Apon, Quynh H Dang, Michael S Davidson, Morris J Dworkin, and Carl A Miller. Recommendation for stateful hash-based signature schemes. *NIST Special Publication*, 800:208, 2020.

[CDW17] Ronald Cramer, Léo Ducas, and Benjamin Wesolowski. Short stickelberger class relations and application to ideal-SVP. *Lecture Notes in Computer Science*, 10210:324–348, 2017.

[CFL+14] Elizabeth Crosson, Edward Farhi, Cedric Yen-Yu Lin, Han-Hsuan Lin, and Peter Shor. Different Strategies for Optimization Using the Quantum Adiabatic Algorithm. *arXiv preprint arXiv:1401.7320*, 2014.

[CGW14] Andrew M Childs, David Gosset, and Zak Webb. The Bose-Hubbard model is QMA-complete. In *International Colloquium on Automata, Languages, and Programming*, pages 308–319. Springer, 2014.

[Cha20] Peter Chapman. Scaling IonQ's Quantum Computers: The Roadmap. *IonQ web blog*, 2020.

[CNAA+20] Christopher Chamberland, Kyungjoo Noh, Patricio Arrangoiz-Arriola, Earl T Campbell, Connor T Hann, Joseph Iverson, Harald Putterman, Thomas C Bohdanowicz, Steven T Flammia, and Andrew Keller. Building a fault-tolerant quantum computer using concatenated cat codes. *arXiv preprint arXiv:2012.04108*, 2020.

[Com99] IEEE Computer Society L A N M A N Standards Committee. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. *ANSI/IEEE Std. 802.11-1999*, 1999.

[Cro18] Gavin E Crooks. Performance of the quantum approximate optimization algorithm on the maximum cut problem. *arXiv preprint arXiv:1811.08419*, 2018.

[DA99] Tim Dierks and Christopher Allen. Rfc2246: The TLS protocol version 1.0, 1999.

[DBJ00] Ivan H Deutsch, Gavin K Brennen, and Poul S Jessen. Quantum computing with neutral atoms in an optical lattice. *Fortschritte der Physik: Progress of Physics*, 48(9 -11):925 – 943, 2000.

[DF17] Luca De Feo. Mathematics of isogeny based cryptography. *arXiv preprint arXiv:1711.04062*, 2017.

[DH76] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976.

[DJ92] David Deutsch and Richard Jozsa. Rapid solution of problems by quantum computation. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, 439(1907):553–558, 1992.

[DY09] Jintai Ding and Bo-Yin Yang. Multivariate public key cryptography. In *Post-quantum cryptography*, pages 193–241. Springer, 2009.

[Ell05] Claire Ellis. Exploring the Enigma. *http://plus. maths. org/content/os/issue34/features/ellis/index¿. Acesso em*, 5:1939–1945, 2005.

[Fey99] Richard P Feynman. Simulating physics with computers. *Int. J. Theor. Phys*, 21(6/7), 1999.

[FG98] Edward Farhi and Sam Gutmann. Quantum computation and decision trees. *Physical Review A*, 58(2):915, 1998.

[FGG14] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A Quantum Approximate Optimization Algorithm. *arXiv:1411.4028*, pages 1–16, 2014.

[FGG15] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A Quantum Approximate Optimization Algorithm Applied to a Bounded Occurrence Constraint Problem. *preprint arXiv:1412.6062*, 2015.

[FGG20] Edward Farhi, David Gamarnik, and Sam Gutmann. The quantum approximate optimization algorithm needs to see the whole graph: a typical case. *arXiv preprint arXiv:2004.09002*, 2020.

[FGGS00] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Michael Sipser. Quantum Computation by Adiabatic Evolution. *arXiv preprint arXiv: quant-ph/0001106*, 2000.

[FGGZ19] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Leo Zhou. The quantum approximate optimization algorithm and the sherrington-kirkpatrick model at infinite size. *preprint arXiv:1910.08187*, 2019.

[FPP10] David Freedman, Robert Pisani, and Roger Purves. *Statistical models and causal inference: a dialogue with the social sciences*. Cambridge University Press, 2010.

[Gam20] Jay Gambetta. Ibm's roadmap for scaling quantum technology. *IBM Research Blog*, 2020.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 169–178, 2009.

[GGH97] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-Key Cryptosystems from Lattice Reduction Problems. In *CRYPTO*, London, 1997. Springer-Verlag.

[GK63] H. A. Gersch and G. C. Knollman. Quantum Cell Model for Bosons. *Physical Review*, 129(2):959–967, 1963.

[GLM09] Ying Hung Gan, Cong Ling, and Wai Ho Mow. Complex lattice reduction algorithm for low-complexity full-diversity MIMO detection. *IEEE Transactions on Signal Processing*, 57(7):2701–2710, 2009.

[GLV20] Charles Grover, Cong Ling, and Roope Vehkalahti. Non-Commutative Ring Learning With Errors From Cyclic Algebras. *arXiv preprint arXiv:2008.01834*, 2020.

[GNR10] Nicolas Gama, Phong Q Nguyen, and Oded Regev. Lattice Enumeration Using Extreme Pruning. *EUROCRYPT 2010, Lecture Notes in Computer Science*, 6110:257–278, 2010.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for Hard Lattices and New Cryptographic Constructions. *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 197–206, 2008.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. *Proceedings, 28th Annual ACM Symposium on the Theory of Computing*, pages 212–219, 1996.

[Ham19] Eric Hamilton. What is edge computing: The network edge explained, 2019.

[HBG+18] Andreas Hülsing, Denis Butin, Stefan-Lukas Gazdag, Joost Rijneveld, and Aziz Mohaisen. XMSS: eXtended Merkle signature scheme. In *RFC 8391*. IRTF, 2018.

[HC98] Dan Harkins and Dave Carrel. The internet key exchange (IKE). Technical report, RFC 2409, november, 1998.

[Hea91] Daniel R Headrick. *The invisible weapon: Telecommunications and international politics, 1851-1945*. Oxford University Press on Demand, 1991.

[Her50] Charles Hermite. Extraits de lettres de M. Ch. Hermite à M. Jacobi sur différents objects de la théorie des nombres. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, 1850(40):261–278, 1850.

[HHF+16] Ryan Hamerly, Yoshitaka Haribara, M. M. Fejer, Carsten Langrock, Takahiro Inagaki, Kazuyuki Aihara, Hideo Mabuchi, Hiroki Takesue, Alireza Marandi, Peter L. McMahon, Yoshihisa Yamamoto, Shoko Utsunomiya, Shuhei Tamate, and Robert L. Byer. A fully programmable 100-spin coherent Ising machine with all-to-all connections. *Science*, 354(6312):614–617, 2016.

[HHGP+03] Jeffrey Hoffstein, Nick Howgrave-Graham, Jill Pipher, Joseph H Silverman, and William Whyte. NTRUSIGN: Digital signatures using the NTRU lattice. In *Cryptographers' track at the RSA conference*, pages 122–140. Springer, 2003.

[HIM+19] Ryan Hamerly, Takahiro Inagaki, Peter L McMahon, Davide Venturelli, Alireza Marandi, Tatsuhiro Onodera, Edwin Ng, Carsten Langrock, Kensuke Inaba, and Toshimori Honjo. Experimental investigation of performance differences between coherent Ising machines and a quantum annealer. *Science advances*, 5(5):eaau0823, 2019.

[HMV06] Darrel Hankerson, Alfred J Menezes, and Scott Vanstone. *Guide to elliptic curve cryptography*. Springer Science & Business Media, 2006.

[HR14] Ishay Haviv and Oded Regev. On the lattice isomorphism problem. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 391–404. SIAM, 2014.

[Hun07] John D Hunter. Matplotlib: A 2D graphics environment. *IEEE Annals of the History of Computing*, 9(03):90–95, 2007.

[JCLM21] David Joseph, Adam Callison, Cong Ling, and Florian Mintert. Two quantum Ising algorithms for the shortest-vector problem. *Physical Review A*, 103(3):32433, 2021.

[JGLM20] David Joseph, Alexandros Ghionis, Cong Ling, and Florian Mintert. Not-so-adiabatic quantum computation for the shortest vector problem. *Physical Review Research*, 013361:1–13, 2020.

[JMLM21] David Joseph, Antonio J Martinez, Cong Ling, and Florian Mintert. Quantum mean value approximator for hard integer value problems. *arXiv preprint arXiv:2105.13106*, 2021.

[JNN12] J Robert Johansson, Paul D Nation, and Franco Nori. QuTiP: An open-source Python framework for the dynamics of open quantum systems. *Computer Physics Communications*, 183(8):1760–1772, 2012.

[KA98] Stephen Kent and Randall Atkinson. RFC2406: IP encapsulating security payload (ESP), 1998.

[KB79] Ravindran Kannan and Achim Bachem. Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. *siam Journal on Computing*, 8(4):499–507, 1979.

[KB14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[KDA76] Donald E Knuth, KNUTH DE, and Y A O AC. THE COMPLEXITY OF NONUNIFORM RANDOM NUMBER GENERATION. 1976.

[KKR06] Julia Kempe, Alexei Kitaev, and Oded Regev. The complexity of the local Hamiltonian problem. *SIAM Journal on Computing*, 35(5):1070–1097, 2006.

[Kle00] Philip Klein. Finding the closest lattice vector when it's unusually close. In *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 937–941, 2000.

[KLM01] Emanuel Knill, Raymond Laflamme, and Gerald J Milburn. A scheme for efficient quantum computation with linear optics. *nature*, 409(6816):46–52, 2001.

[KN98] Tadashi Kadowaki and Hidetoshi Nishimori. Quantum annealing in the transverse Ising model. *Physical Review E - Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, 58(5):5355–5363, 1998.

[KZ73] Alexandr Korkin and Egor Zolotarev. Sur les formes quadratiques. *Math. Ann.*, 6:366–389, 1873.

[LD98] Daniel Loss and David P DiVincenzo. Quantum computation with quantum dots. *Physical Review A*, 57(1):120, 1998.

[LFC+20] Li Li, Minjie Fan, Marc Coram, Patrick Riley, and Stefan Leichenauer. Quantum optimization with a novel gibbs objective function and ansatz architecture search. *Physical Review Research*, 2(2):23074, 2020.

[LL09] Thomas Länger and Gaby Lenhart. Standardization of quantum key distribution and the ETSI standardization initiative ISG-QKD. *New Journal of Physics*, 11(5):55051, 2009.

[LLL82] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[LMHG10] Cong Ling, Wai Ho Mow, and Nick Howgrave-Graham. Variants of the lll algorithm in digital communications: Complexity analysis and fixed-complexity implementation. *arXiv preprint arXiv:1006.1661*, 2010.

[LMHG13] Cong Ling, Wai Ho Mow, and Nick Howgrave-Graham. Reduced and fixed-complexity variants of the LLL algorithm for communications. *IEEE Transactions on Communications*, 61(3):1040–1050, 2013.

[LMVDP13] Thijs Laarhoven, Michele Mosca, and Joop Van De Pol. Solving the shortest vector problem in lattices faster using quantum search. In *PQCrypto*, pages 83–101, 2013.

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 1–23. Springer, 2010.

[LPW06] Kerstin Lemke, Christof Paar, and Marko Wolf. *Embedded security in cars*. Springer, 2006.

[Luc14] Andrew Lucas. Ising formulations of many NP problems. *Front. Phys*, 2(5), 2014.

[Mat96] George B Mathews. On the partition of numbers. *Proceedings of the London Mathematical Society*, 1(1):486–490, 1896.

[MBS+18] Jarrod R McClean, Sergio Boixo, Vadim N Smelyanskiy, Ryan Babbush, and Hartmut Neven. Barren plateaus in quantum neural network training landscapes. *Nature communications*, 9(1):1–6, 2018.

[McE78] Robert J McEliece. A public-key cryptosystem based on algebraic. *Coding Thv*, 4244:114–116, 1978.

[MCF19] David McGrew, Michael Curcio, and Scott Fluhrer. Leighton-Micali hash-based signatures, 2019.

[McK11] Wes McKinney. pandas: a foundational Python library for data analysis and statistics. *Python for High Performance and Scientific Computing*, 14(9):1–9, 2011.

[Mer79] Ralph Charles Merkle. *Secrecy, authentication, and public key systems*. Stanford university, 1979.

[MHRB19] Catherine C McGeoch, Richard Harris, Steven P Reinhardt, and Paul I Bunyk. Practical annealing-based quantum computing. *Computer*, 52(6):38–46, 2019.

[Mic08] Daniele Micciancio. Efficient reductions among lattice problems. In *SODA*, pages 84–93. Citeseer, 2008.

[Min16] Hermann Minkowski. *Geometrie der zahlen*, volume 2. Ripol Classic, 2016.

[MJE+19] Sam McArdle, Tyson Jones, Suguru Endo, Ying Li, Simon C Benjamin, and Xiao Yuan. Variational ansatz-based quantum simulation of imaginary time evolution. *npj Quantum Information*, 5(1):1–6, 2019.

[MLLL15] Somayya Madakam, Vihar Lake, Vihar Lake, and Vihar Lake. Internet of Things (IoT): A literature review. *Journal of Computer and Communications*, 3(05):164, 2015.

[Moč75] Jonas Močkus. On Bayesian methods for seeking the extremum. In *Optimization techniques IFIP technical conference*, pages 400–404. Springer, 1975.

[Moc12] Jonas Mockus. *Bayesian approach to global optimization: theory and applications*, volume 37. Springer Science & Business Media, 2012.

[Mon18] Ashley Montanaro. Quantum walk speedup of backtracking algorithms. *Theory Comput.*, 14(15):1–24, 9 2018.

[MV10] Daniele Micciancio and Panagiotis Voulgaris. Faster exponential time algorithms for the shortest vector problem. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 1468–1480. SIAM, 2010.

[Ngu99] Phong Nguyen. Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from crypto'97. In *Annual International Cryptology Conference*, pages 288–304. Springer, 1999.

[NR06] Phong Q Nguyen and Oded Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 271–288. Springer, 2006.

[NS98] Phong Nguyen and Jacques Stern. Cryptanalysis of the Ajtai-Dwork cryptosystem. In *Annual International Cryptology Conference*, pages 223–242. Springer, 1998.

[Oli06] Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[OS09] Raphael Overbeck and Nicolas Sendrier. Code-based cryptography. In *Post-quantum cryptography*, pages 95–145. Springer, 2009.

[Pan12] Dmitry Panchenko. The Sherrington-Kirkpatrick model: an overview. *Journal of Statistical Physics*, 149(2):362–383, 2012.

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem. *Proceedings of STOC*, pages 333–342, 2009.

[Pei10] Chris Peikert. An efficient and parallel Gaussian sampler for lattices. In *Annual Cryptology Conference*, pages 80–97. Springer, 2010.

[Pei16] Chris Peikert. A Decade of Lattice Cryptography. *Foundations and Trends in Theoretical Computer Science*, 10(4):283–424, 2016.

[PM20] Marco Piani and Michele Mosca. Quantum Threat Timeline Report 2020. *evolutionQ website*, 2020.

[PP09] Christof Paar and Jan Pelzl. *Understanding cryptography: a textbook for students and practitioners*. Springer Science & Business Media, 2009.

[Pre18] John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, 2018.

[PS98] Christos H Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1998.

[PS09] Xavier Pujol and Damien Stehlé. Solving the Shortest Lattice Vector Problem in Time 22.465 n. *IACR Cryptol. ePrint Arch.*, 2009:605, 2009.

[PSW08] Thomas Plantard, Willy Susilo, and Khin Than Win. A digital signature scheme based on CVP∞. In *International Workshop on Public Key Cryptography*, pages 288–307. Springer, 2008.

[Rab79] Michael O Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical report, Massachusetts Inst of Tech Cambridge Lab for Computer Science, 1979.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Proceedings of STOC*, pages 84–93, 2005.

[RNSL17] Martin Roetteler, Michael Naehrig, Krysta M Svore, and Kristin Lauter. Quantum resource estimates for computing elliptic curve discrete logarithms. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 241–270. Springer, 2017.

[RPS11] Michael Rose, Thomas Plantard, and Willy Susilo. Improving BDD cryptosystems in general lattices. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6672 LNCS:152–167, 2011.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[SE94] C. P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical Programming*, 66(1-3):181–199, 1994.

[SH95] Claus-Peter Schnorr and Horst Helmut Hörner. Attacking the Chor-Rivest cryptosystem by improved lattice reduction. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 1–12. Springer, 1995.

[Sho99] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[Sim09] Denis Simon. Selected applications of LLL in number theory. In *The LLL Algorithm*, pages 265–282. Springer, 2009.

[Sle06] Alexander Slepoy. Quantum gate decomposition algorithms. Technical report, Sandia National Laboratories, 2006.

[Spa98] James C Spall. Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Transactions on aerospace and electronic systems*, 34(3):817–823, 1998.

[SS11] Damien Stehlé and Ron Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 27–47. Springer, 2011.

[SUXF06] Ralf Schützhold, Michael Uhlmann, Yan Xu, and Uwe R Fischer. Sweeping from the superfluid to the Mott phase in the Bose-Hubbard model. *Physical review letters*, 97(20):200601, 2006.

[UF12] Michael S Underwood and David L Feder. Bose-Hubbard model for universal quantum-walk-based computation. *Physical Review A*, 85(5):52314, 2012.

[Was21] Michael L Waskom. Seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021.

[WB17] Phillip Weinberg and Marin Bukov. QuSpin: a Python package for dynamics and exact diagonalisation of quantum many body systems part I: spin chains. 2017.

[WN99] Laurence A Wolsey and George L Nemhauser. *Integer and combinatorial optimization*, volume 55. John Wiley & Sons, 1999.

[WSW+21] Max Wilson, Rachel Stromswold, Filip Wudarski, Stuart Hadfield, Norm M Tubman, and Eleanor G Rieffel. Optimizing quantum heuristics with meta-learning. *Quantum Machine Intelligence*, 3(1):1–14, 2021.

[Yuv79] Gideon Yuval. How to swindle Rabin. *Cryptologia*, 3(3):187–191, 1979.

[ZWC+20] Leo Zhou, Sheng-Tao Wang, Soonwon Choi, Hannes Pichler, and Mikhail D Lukin. Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices. *Physical Review X*, 10(2):21067, 2020.