Two quantum Ising algorithms for the shortest-vector problem

David Joseph,[1,2] Adam Callison [ORCID],[2] Cong Ling,[1] and Florian Mintert[2]

[1]*Electrical and Electronic Engineering Department, Imperial College London, London SW7 2BU, United Kingdom*
[2]*Physics Department, Imperial College London, London SW7 2BU, United Kingdom*

Quantum computers are expected to break today's public key cryptography within a few decades. New cryptosystems are being designed and standardized for the postquantum era, and a significant proportion of these rely on the hardness of problems like the shortest-vector problem to a quantum adversary. In this paper we describe two variants of a quantum Ising algorithm to solve this problem. One variant is spatially efficient, requiring only $O(N \log_2 N)$ qubits, where N is the lattice dimension, while the other variant is more robust to noise. Analysis of the algorithms' performance on a quantum annealer and in numerical simulations shows that the more qubit-efficient variant will outperform in the long run, while the other variant is more suitable for near-term implementation.

I. INTRODUCTION

The concept of quantum computing (QC) was first conceived of in the early 1980s [1,2] and has slowly grown to become a major field within modern computer science and physics. Utilizing intrinsic properties of quantum mechanics allows some computations to be sped up beyond what is classically possible. Some offer exponential speedup, such as integer factorization and the discrete logarithm [3], whereas others offer polynomial, but still impressive, improvements like Grover's algorithm for searching unsorted lists [4].

As the field of quantum computing has blossomed, multiple paradigms and families of algorithms have emerged. The gate model of quantum computing most closely resembles classical computers, with directly programmable qubit architectures, whereas quantum annealer-style algorithms can be seen as an analog version, in which after system initialization in some eigenstate of a Hamiltonian, the Hamiltonian is gradually altered until at completion the system is measured to be in some eigenstate of a new Hamiltonian which offers a solution to the problem under consideration.

While some forms of quantum computing, such as the gate model and adiabatic quantum computing (AQC) [5,6], are universal, near-term quantum annealing devices are likely to be more suited to specific problem types. Nonetheless, these near-term devices have compelling use cases—from simulating quantum chemistry to developing medicines [7,8]—though some are more practical in the near future than others. In 2021 the QC community finds itself at a turning point, with the first credible claim to quantum supremacy [9] having been made in late 2019, though performing useful computations of this size is still some way off for quantum computers.

A. Postquantum cryptography

One area subject to much disruption is that of cryptography. Classical cryptography will be a victim—once quantum hardware reaches maturity—of the exponential speedup due to Shor's algorithm for integer factorization and discrete logarithm computation. This is because the security of public key cryptography relies upon the existence of a problem that is intractable without a certain piece of information (known as a key) but is efficiently computable when in possession of the key. An example of such a problem is factorization of a large semiprime number $n = pq$, which has only two nontrivial prime factors, p and q. If either factor is known, one can divide n by that factor to ascertain the other. If, however, one knows neither factor, then one must resort to a much more computationally expensive approach, such as attempting to divide n by every integer up to $\sqrt{n}$ (naive) or applying one of the family of number field sieves [10], which in the best cases take superpolynomial time.

The security of the Rivest Shamir and Adelman (RSA) cryptosystem [11] relies on integer factorization, while the security of Diffie-Hellman key exchange, ElGamal, and more rely on closely related problems [12,13], all efficiently computable in the QC domain. To preserve the security of communications and information storage moving forward into a postquantum world, a new set of cryptographic primitives must be developed and demonstrated to be invulnerable to quantum attacks. This replacement field is known as postquantum cryptography.

The search for quantum-safe primitives centers around five families of problems, as outlined in [12]: lattice-based cryptography (LBC), code-based cryptography, isogenies, and multivariate-based and hash-based cryptography. Candidate systems are being assessed in the NIST Post-Quantum Cryptography Standardization process, and there are 26 systems being analyzed in round 2 of the process, of which 12 derive from lattice-based primitives.

LBC has spawned the celebrated learning with errors (LWE) problem [14], later adapted for efficiency (at the risk of as yet unknown security trade-offs) into Ring-LWE, in which computations are performed in algebraic number fields, and also Module-LWE. LWE has even served as the basis for

the first fully homomorphic encryption scheme [15]. Other notable lattice-based cryptosystems include NTRU [16] and Goldreich–Goldwasser–Halevi (GGH) [17]. The central problems in LBC tend to revolve around minimization of distances in high-dimensional spaces.

Two closely related problems are finding the shortest distance between two points in a lattice, known as the shortest-vector problem (SVP), and finding the closest lattice point to any given vector in the ambient space (CVP). Under a guarantee that said point is *at most* a certain distance from the nearest lattice point CVP becomes a bounded distance decoding problem, upon which the security proof of LWE is based.

B. Quantum algorithms for LBC

Up to this point most quantum algorithms for lattice problems have focused on application of a preexisting algorithm to gain a quantum speedup in a primarily classical approach. The two main approaches to lattice problems are enumeration and sieving. Sieving takes as input a selection of vectors from some distribution over the lattice and iteratively combines them in order to output short solutions probabilistically [18]. On the other hand, enumeration literally enumerates all vectors within a certain ball around the origin, which if picked carefully is guaranteed to contain the best possible solution [19,20], though, recently, significant speedups have been obtained by moving into the probabilistic domain (using a technique called extreme pruning), randomizing the input, and repeating many times [21].

Grover's algorithm has been applied to sieving and saturation algorithms to achieve speedups of roughly 25% in the exponent [22]. It seems unlikely that Grover's algorithm can be applied to lattice enumeration, but building on developments for quantum tree algorithms [23,24] a quadratic speed-up has been obtained [25]. The quantum Fourier transform (QFT) plays a part in many quantum algorithms, such as those for solving variants of the hidden subgroup problem (HSP)—Shor's is an example. The dihedral coset problem is another type of HSP; a relaxed form, the extrapolated dihedral coset problem, has been shown to be equivalent to LWE [26]. Lattice problems in certain algebraic number fields can be solved using quantum HSP algorithms that compute unit groups [27] and principal ideals [28]. A recent work [29] proposed a different approach to finding short vectors, encoding vector norms into a Hamiltonian of a system of ultracold bosons trapped in a potential landscape. While broadly in the adiabatic quantum optimization regime, sweeps are performed subadiabatically to obtain results from a distribution over low-energy eigenstates (consequently, "short" vectors).

C. Contribution

Section II contains preliminaries; then in Sec. III we detail a derivative quantum shortest-vector algorithm based on the quantum Ising model. In particular, two variants are presented with provable asymptotic space requirements. In Sec. IV these two algorithms are analyzed in a noiseless setting numerically and are also implemented on the D-Wave quantum annealer [30], providing a fully quantum analysis of lattice
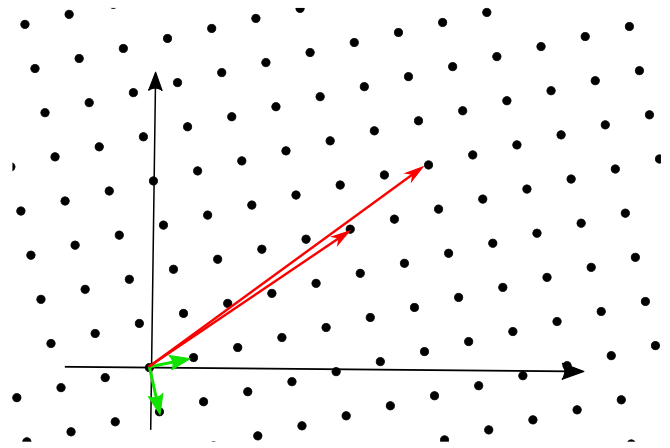

FIG. 1. An example of a two-dimensional lattice with lattice points depicted in black. Red arrows represent a bad basis with long vectors that are far from orthogonal (though still linearly independent); green arrows are a good basis for the same lattice, with short and highly orthogonal vectors. Both bases (by definition) span the lattice.

problems in up to seven-dimensional instances utilizing 56 logical qubits (and over 1000 physical qubits).

The two variants of the algorithm presented relate to different ways of encoding qudits in the quantum Ising model, and we offer an analysis of both implementations, including the circumstances in which each is superior. This last contribution has wider relevance for the QC community, especially when looking at algorithms to optimize over integral combinations of more general vectors.

II. PRELIMINARIES

Vectors and matrices are denoted by boldface lowercase and uppercase letters, respectively, while Hamiltonians are denoted by H. Throughout the paper two vector norms are of interest: the l^2 (or Euclidean) norm of a d-dimensional vector $\mathbf{x}$ is described by $\|\mathbf{x}\|^2 = x_1^2 + \cdots + x_d^2$, and the infinity norm is $\|\mathbf{x}\|_\infty = \max\{|x_i|, 1 \leqslant i \leqslant d\}$. The length of the shortest vector is denoted λ_1; there are at least two vectors of this length in a lattice, as any vector can be reflected about the origin to produce another of identical length. Where log is used, it is in base 2.

A. Lattices

Lattices are simply a repeating pattern of points in N-dimensional space. Figure 1 shows an example of a lattice in two dimensions. Lattices have two attractive mathematical properties: they all contain the origin, and adding any two lattice vectors together with integer coefficients gives a point that is also in the lattice. The concept can be formalized as follows:

Definition 1. A lattice $\mathcal{L} \in \mathbb{R}^N$ is the discrete set of all integer combinations of a set of N linearly independent basis vectors $\mathbf{B} = \{\mathbf{b}_0, \ldots, \mathbf{b}_N\}$:

$$\mathcal{L} = \left\{ \sum_{i=1}^{N} x_i \mathbf{b}_i \right\} = \{\mathbf{B} \cdot \mathbf{x} : \mathbf{x} \in \mathbb{Z}^N\}.$$

Every lattice contains the zero vector, denoted **0**, and this is considered a trivial lattice vector, as $\mathbf{0} \cdot \mathbf{B} = \mathbf{0}$. A set of N linearly independent basis vectors together are known as a "basis." Each lattice has infinitely many bases, and every basis can be mapped to every other basis by a unimodular transformation. There is a notion of good and bad bases, where good means that the basis vectors are quite orthogonal and quite short, where "quite" varies according to the context but can be defined in terms of bounds on the angles between basis vectors or ratios of their lengths. Knowledge of the specifics of these conditions will not be important to the work, but it is essential to appreciate the concept that *short and close to orthogonal are good*.

Necessarily, there are only a small number of good bases (a finite number by any definition of good) and infinitely many bad bases. The subject of turning arbitrary bad lattice bases into "good enough" bases is a widely studied one and is at the center of lattice cryptanalysis [31–33].

Definition 2. The fundamental parallelepiped $\mathcal{P}(\mathcal{L})$ of a lattice described by a basis **B** is the set of points in $\mathbb{R}^N$

$$\mathcal{P}(\mathcal{L}) = \{\mathbf{x} \cdot \mathbf{B} \mid \mathbf{x} \in [0, 1)^N\},$$

and the covolume of $\mathcal{L}$ is defined to be the volume of this N-dimensional polyhedron.

The shape of the fundamental parallelepiped depends on the geometry of the particular basis, but all fundamental parallelepipeds for a given lattice will have the same covolume and provide a tiling on the ambient space $\mathbb{R}^N$. One can compute the covolume of a lattice by taking its determinant, which is obtained by taking the determinant of the basis $\det(\mathcal{L}) = \det(\mathbf{B})$.

One important family of bases for this work is that of Hermite normal form (HNF) bases. The reasons these are of interest are as follows:

(1) They can be described (and therefore communicated) efficiently.

(2) Each lattice has a unique HNF.

(3) The HNF of a lattice can be efficiently computed from any provided basis.

Definition 3. The Hermite normal form (row-basis version) of a full-rank lattice $\mathcal{L} \subset \mathbb{Z}^N$ is an upper-triangular matrix **H** that satisfies

(1) $\mathbf{H}_{ij} = 0$ for $i > j$.

(2) The first nonzero term from the left (the pivot) is positive and strictly to the right of the first nonzero term of the row above.

(3) Elements directly below the pivot are zero, and those directly above are reduced modulo the pivot.

The HNF of a lattice is described as *optimal* if there exists only one nonunit pivot. This means (for a row-basis HNF) that there is only one column of nonzero values. In this work, we examine only full-rank integer lattices, so we take $\mathbf{b}_i \in \mathbb{Z}^N$ from this point onward. In general, the HNF of a lattice is a bad basis by any canonical measure, though it is an efficient means of representing the lattice as it contains fewer nonzero entries than a general basis.

The lattice problem that is the focus of this work is the shortest-vector problem.

Definition 4. For the shortest-vector problem, given a basis $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ describing a lattice $\mathcal{L}$, find the closest (nonzero) lattice point to the origin,

$$\lambda_1(\mathcal{L}) = \min(||\mathbf{x}|| : \mathbf{x} \in \mathcal{L} \backslash \{\mathbf{0}\}).$$

This can alternatively be stated as finding the shortest distance between any two distinct points in the lattice. A major stepping-stone in the field was the reduction from GapSVP (a close relative to SVP) to the LWE cryptosystem [14].

B. Continuous-time quantum computing

Continuous-time quantum computing (CTQC) refers to a group of quantum computational strategies in which a specially engineered Hamiltonian $H(t)$ is applied to a physical system in order to drive it from an initial state toward a state from which the solution to a problem can be read. Typically, the Hamiltonian $H(t)$ is a linear combination (which may or may not be time dependent) of a problem-independent driver Hamiltonian H_0 (often referred to as an initial Hamiltonian when appropriate) and a problem Hamiltonian H_P which encodes the problem to be solved. The eigenstates of the problem Hamiltonian H_P correspond to solutions to a problem (often the ground state is desired), and targeting these eigenstates is where the crux of CTQC lies.

CTQC encompasses a number of quantum algorithmic families, for example, AQC [5], quantum annealing [34], continuous-time quantum walk computing [35,36], and others. Furthermore, the quantum approximate optimization algorithm [37] in the discrete-time gate model is inspired by continuous-time methods.

For the purposes of this work, we will consider systems which are initialized in the ground state of the initial Hamiltonian and are evolved according to a linear time sweep of length T, leaving the system in a Hamiltonian which at any time $t \in [0, T]$ can be described as

$$H(t) = \left(1 - \frac{t}{T}\right)H_0 + \left(\frac{t}{T}\right)H_P. \tag{1}$$

In the following section, the initial Hamiltonian H_0, the problem Hamiltonian H_P, and, consequently, the full Hamiltonian $H(t)$ are defined in the context of the quantum Ising model. The key to constructing the algorithm is in defining the problem Hamiltonian H_P such that low-energy eigenstates relate to good solutions, which reduces in the Ising model to the setting of appropriate fields on and coupling between spins.

C. Quantum annealing in the Ising model

The Ising model originated as a tool for modeling ferromagnetism in materials. In a given material each magnetic domain has a dipole, or spin, denoted s_i. These spins interact with their neighbors in a manner dependent on the properties of the material. The system achieves its lowest-energy state when the spins align so as to minimize the total interaction energies. The energy of such a system is described by the Hamiltonian $H = -\sum_{i,j} J_{ij}s_i s_j - \sum_i h_i s_i$, where J_{ij} are *coupling coefficients* and h_i are *field strengths*.

The transverse Ising model was introduced to quantum computing back in 1998 [34]. A transverse magnetic field can represent temperature, and reducing the strength of this field brings about "quantum cooling." The ground state of

a system modeled by an Ising Hamiltonian can be found if the system is cooled sufficiently slowly. In a material, the coefficients J_{ij} and h_i of the Ising model are determined by the properties of the material. In a quantum computing device that implements the Ising model, however, the programmer chooses the coefficients in order to encode their problem. The programmer sets the coefficients to ensure that the ground state and other low-energy eigenstates encode good solutions to the problem they wish to solve.

A *classical* Ising Hamiltonian

$$H(s_1, \ldots, s_N) = -\sum_{i<j}^N J_{ij} s_i s_j - \sum_{i=1}^N h_i s_i \quad (2)$$

can be written as a quadratic function of N spins $s_i = \pm 1$, which can be mapped to a quantum setting by replacing the spins s_i with Pauli Z operators σ_i^z,

$$H_P = H(\sigma_1^z, \ldots, \sigma_N^z). \quad (3)$$

Pauli operators can be expressed as 2×2 matrices,

$$\sigma_0 = \mathbb{1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad (4)$$

$$\sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad (5)$$

where $\sigma_i^{0,x,y,z}$ represents a Pauli operator acting only on the *i*th qubit of the system.

As is typical for quantum annealing in the Ising model, for the algorithms we consider in this work, we choose the initial Hamiltonian H_0 to be a simple transverse-field Hamiltonian composed of Pauli X operators

$$H_0 = -h_0 \sum_{i=1}^N \sigma_i^x. \quad (6)$$

The system is initialized in the ground state of the transverse-field Hamiltonian H_0, which is an equal superposition of all eigenstates of the problem Hamiltonian H_P, before evolving the Hamiltonian toward the problem Hamiltonian H_P according to Eq. (1).

III. QUANTUM ISING SVP

In this section we describe how the encoding of the SVP problem into the Ising coefficients J_{ij}, h_i that define the energy of the system is performed. The coefficients are derived directly from the input basis, following a process similar to that for the Bose-Hubbard quantum SVP algorithm [29].

Any lattice point can be written as the vector

$$\mathbf{v} = \mathbf{x} \cdot \mathbf{B} = x_1 \mathbf{b}_1 + \cdots + x_N \mathbf{b}_N. \quad (7)$$

The l_2 norm of the vector $\mathbf{v}$ can be written as

$$||\mathbf{v}||^2 = \sum_{i,j} x_i x_j \mathbf{b}_i \cdot \mathbf{b}_j. \quad (8)$$

The aim is to find the integer combination $\mathbf{x} = (x_1, \ldots, x_N)$ that minimizes this sum over $i, j = 1, \ldots, N$, given the fixed scalar values $\mathbf{b}_i \cdot \mathbf{b}_j$ which are determined by the lattice basis
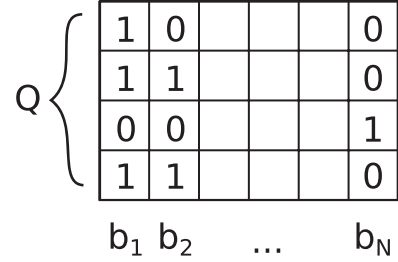

FIG. 2. Representation of qudits as a collection of several qubits. Each column corresponds to a basis vector, and the qubits in that column are interpreted as an integer, according to the qudit definitions, which are set out in Secs. III A and III B. A binary-encoded qudit in the above can take 1 of 16 values, whereas a Hamming-encoded qudit can take 1 of 5 values.

with which the algorithm is run. It would be rather straightforward to map this minimization into a device consisting of coupled qudits rather than qubits, that is, if the device implemented a generalized Ising model Hamiltonian with the spins $s_i = \pm 1$ generalized to take integer values. Since this is not the case, it is necessary to construct an ersatz qudit by combining multiple qubits. This can be visualized, as in Fig. 2, as a grid of spins, where each column represents a qudit for a different basis vector.

Assuming such qudits are available, we can write $\hat{Q}^{(j)}$ to mean the qudit operator acting on qudit j. Then, following [29], the problem Hamiltonian can be written as

$$H_P = \sum_{i,j}^N \hat{Q}^{(i)} \hat{Q}^{(j)} \mathbf{G}_{ij}, \quad (9)$$

where $\mathbf{G}_{ij} = \mathbf{b}_i \cdot \mathbf{b}_j$ is the (i, j)th element of the Gram matrix for the lattice basis.

The eigenstates of H_P in Eq. (9) all correspond to vectors in the lattice for which every component x_j is expressible within the range of values taken by the qudits (how big this range should be is a separate question, which we address later in this section). The corresponding eigenvalues are simply the squared Euclidean length of those vectors. Thus, the ground state of the problem Hamiltonian H_P will correspond to the uninteresting 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 applying the transformation $x_j \to -x_j$ to each vector coordinate of $\mathbf{v}$ produces $-\mathbf{v}$, which has the same length. Solving SVP thus becomes equivalent to finding a state in the first excited manifold of the problem Hamiltonian H_P.

In the following, we describe two different ways to encode the qudits into these column bitstrings, as well as present bounds describing how big the range of qudit values must be, and therefore how many physical qubits are needed, to ensure that the problem Hamiltonian H_P actually expresses at least one shortest vector.

A. Hamming-weight-encoded qudits

This qudit mapping is extremely simple and is not optimal in terms of space. The reason is because it leads to redundancies, with multiple spin configurations corresponding to the same qudit value in $[-2^k, 2^k]$. The reason for presenting

this mapping is that it is more robust to noise (explained in Appendix B) than the more spatially efficient approach we present subsequently, which is demonstrated in Sec. IV. While it is not expected that even today's cryptosystems will be broken with noisy intermediate-scale quantum (NISQ) [38,39] computers, for the foreseeable future the quantum computing community must contend with poor-quality qubits. Consequently, this simple qudit may become a useful tool in the near term. For the purposes of the analysis in Sec. IV, the D-Wave 2000Q provides ample qubits to compensate for the inefficiencies of this mapping, and so what follows is the quantum Ising SVP algorithm *for now*.

Here, we define a qudit operator by a simple sum of qubit operators,

$$\hat{Q}_{\text{Ham}}^{(j)} = \sum_{p=0}^{2^{k+1}} \frac{\hat{Z}_{pj}}{2}. \tag{10}$$

This qudit operator assigns to each computational basis state a value by counting the number of qubits in the $+1$ state, its *Hamming weight*, shifted so that the possible values are symmetric about zero. From here on, when referring to the quantum Ising SVP algorithm with Hamming-weight-encoded qudits, we will use the term Ham.

B. Binary-encoded qudits

This qudit mapping assigns values to the states of its qubit register by combining the constituent qubits into a binary number. This is maximally efficient in space as each Ising spin configuration results in a distinct coefficient vector. This optimal efficiency, however, necessitates high-quality qubits, as we will show in Sec. IV, and so this is the quantum Ising SVP algorithm *for later*.

We define the qudit operator on the jth qudit as

$$\hat{Q}_{\text{Bin}}^{(j)} = -\sum_{p=0}^{k} 2^{p-1}\hat{Z}_{pj} - \frac{1}{2}\mathbb{1}, \tag{11}$$

which maps the operators $\sigma_Z^{0,j}, \ldots, \sigma_Z^{k,j}$ on qubits $(0, j), \ldots, (k, j)$ to integers in the range $[-2^k, 2^k - 1]$, which is a roughly symmetric range around the origin. The values are to be interpreted as a binary number in $[0, 2^{k+1} - 1]$ which is then 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, we will use the term Bin.

C. Space analysis

Examination of the space requirements for implementation of this quantum Ising SVP algorithm (Bin) leads to the following theorem, and a similar analysis is conducted for Ham in Corollary 1:

Theorem 1. For any N-dimensional lattice $\mathcal{L}$ with covolume $\det(\mathcal{L}) = D$ and optimal Hermite normal form, there exists a quantum SVP algorithm that can be run on a system of size at most $(\frac{3N}{2}\log N + N + \log D)$ qubits.

We point out here that the optimal HNF is particularly desirable for cryptography applications [40], and lattices with

optimal HNF are common, with approximately 40% of lattices selected at random having optimal HNF [41].

Proof. The point of the proof is to ascertain how many qubits are required per qudit in order to guarantee that the shortest vector in $\mathcal{L}$ is represented in the Hilbert space explored by the algorithm. This question can be reduced to finding a bound on the range in which to search for coefficients for the basis vectors. That is, the algorithm seeks to minimize the length of vectors $\mathbf{v} = \mathbf{x} \cdot \mathbf{B}$, where $-2^k \leqslant x_i < 2^k$ for $1 \leqslant i \leqslant N$. The HNF basis of $\mathcal{L}$ looks as follows:

$$\begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_N \end{bmatrix} = \begin{bmatrix} 1 & & & b_1 \\ & \ddots & & \vdots \\ & & 1 & b_{N-1} \\ & & & D \end{bmatrix}. \tag{12}$$

The covolume of $\mathcal{L}$ is D as, for bases of this form, the determinant is just the product of the diagonal entries. The form of a general lattice vector, written as a linear combination of the basis vectors, is

$$[x_1 \quad \cdots \quad x_N] \cdot \begin{bmatrix} 1 & & & b_1 \\ & \ddots & & \vdots \\ & & 1 & b_{N-1} \\ & & & D \end{bmatrix} \tag{13}$$

$$= [x_1, \quad \ldots, \quad x_{N-1}, \quad (x_1 b_1 + \cdots + x_N D)].$$

Minkowski's theorem [42] provides the bound

$$\lambda_1(\mathcal{L}) \leqslant \sqrt{N} D^{1/N} \tag{14}$$

on the length of the shortest vector, and we will use a weaker version of this to bound the coefficients on the right-hand side of Eq. (13). Minkowski's bound prescribes a sphere about the origin, with radius equal to the bound in Eq. (14), in which to search for the shortest vector. Relaxing this constraint slightly, it can be asserted that every coordinate of $\mathbf{v}$ must be less than or equal to $\sqrt{N} D^{1/N}$, which now prescribes a (larger) N cube around the origin in which to search.

This means that each coordinate on the right-hand side of Eq. (13) must be smaller than the bound, so for the first $N - 1$ entries in the coefficient vector,

$$|x_1|, \ldots, |x_{N-1}| \leqslant \sqrt{N} D^{1/N}. \tag{15}$$

For the final coordinate x_N, we have that

$$|x_1 b_1 + \cdots + x_N D| \leqslant \sqrt{N} D^{1/N}, \tag{16}$$

and so

$$|x_N D| \leqslant \sqrt{N} D^{1/N} + |x_1 b_1 + \cdots + x_{N-1} b_{N-1}|. \tag{17}$$

Applying the triangle inequality,

$$|x_N D| \leqslant |x_1 b_1| + \cdots + |x_{N-1} b_{N-1}| + \sqrt{N} D^{1/N}, \tag{18}$$

and using the HNF property that b_i are reduced modulo D,

$$|x_N D| \leqslant |x_1 D| + \cdots + |x_{N-1} D| + |\sqrt{N} D^{1/N}|. \tag{19}$$

Then by applying Eq. (15)

$$|x_N D| \leqslant (N - 1)\sqrt{N} D^{1/N} D + \sqrt{N} D^{1/N} \leqslant N^{3/2} D^{1+1/N}. \tag{20}$$

This gives an interval to search for x_N bounded in size by

$$|x_N| \leqslant 2N^{3/2}D^{1/N} \tag{21}$$

and an interval to search for x_i $(1 \leqslant i < N)$ bounded in size [from Eq. (15)] by $(2N^{1/2}D^{1/N})$. The algorithm therefore requires

$$\log(2N^{3/2}D^{1/N}) = 1 + \frac{3}{2}\log N + \frac{1}{N}\log D \tag{22}$$

qubits per qudit, meaning the total system requires $O(N \log N + \log D)$ qubits. ∎

Based on the above analysis, we can state the space requirements for Ham:

Corollary 1. Ham can be implemented on any N-dimensional lattice $\mathcal{L}$ with covolume $\det(\mathcal{L}) = D$ and optimal Hermite normal form, using a system of at most $2N^{5/2}D^{1/N}$ qubits. The first excited eigenstate, modulo degeneracies, of the system at completion solves SVP on $\mathcal{L}$.

Sketch of proof. Recycling working from the proof of Theorem 1, the number of qubits in the grid is determined by the dimension of the lattice (N) and the interval over which to search for coefficients, which is given in Eq. (21) as $2N^{3/2}D^{1/N}$, leaving total qubit scaling as $2N^{5/2}D^{1/N}$.

The bounds given in Theorem 1 and Corollary 1 are upper bounds that guarantee the existence of a vector with length $\lambda_1(\mathcal{L})$ (a shortest vector) within the Hilbert space explored by the quantum Ising algorithms described in Secs. III B and III A. In practice, an attacker may choose to reduce the complexity of the system by reducing the qubits-per-qudit parameter in order to give stronger probabilities of finding short vectors at the expense of the assurance of being able to find a shortest vector.

It is also the case that Theorem 1 and Corollary 1 give upper bounds for HNF input bases. Using different, (classically) better-reduced bases is also likely to require lower qubit-per-qudit values and so yield better results, but giving a bound for these is not straightforward, and the HNF has the advantage that each lattice has a unique HNF that can be efficiently obtained from any other basis.

IV. RESULTS

To analyze the performance of the two algorithms described in Sec. III, Ham and Bin were implemented on the D-Wave 2000Q quantum annealer to shed light on what is possible in the NISQ era of quantum computing. We also performed numerical simulations of ideal, closed-system versions of these algorithms to give an indication of what may be possible in the future, though we were limited to smaller experiment sizes due to the computational constraints of simulating quantum systems on classical hardware.

In our experiments we generated random full-rank integer lattices and obtained a "bad" input basis by postmultiplying with randomly generated unimodular matrices. These bad bases were not in HNF, and we did not use the upper bounds given in Theorem 1 and Corollary 1 but instead fixed our qudits to the ranges $[-4, 4]$ and $[-4, 3]$ for Ham and Bin, respectively. This choice was made in order to improve overall

results by a range of metrics discussed in this section while also fixing a degree of freedom to make the analysis easier to perform.

This means that it is not guaranteed that a shortest vector [of length $\lambda_1(\mathcal{L})$] is represented in the Hilbert space for every instance. In fact, a shortest vector is represented in all of the three-dimensional lattice instances we used and approximately half of the instances in each of the higher dimensions. We have not postselected on instances where shortest vectors are obtainable with coefficients in the specified range.

A. Numerical simulation

We numerically simulated results for both the Ham and Bin algorithms for three-dimensional lattices for a range of sweep times to investigate what performance could be expected in the limit of perfect hardware.

Figure 3 depicts the probabilities of obtaining the zero-vector (blue circles), the shortest nonzero vector (red diamonds), and the second-shortest nonzero vector (green crosses) as a function of the sweep duration for Ham encoded qudits [Fig. 3(a)] and Bin encoded qudits [Fig. 3(b)]. For slow sweeps, there is a high probability of obtaining the zero vector, as one would expect for adiabatic dynamics. The adiabatic limit is achieved well for the displayed sweep durations in Fig. 3(a), but Fig. 3(b) shows only the onset of adiabaticity, even though the x axis extends to substantially slower sweeps than in Fig. 3(a).

For fast sweeps, both Figs. 3(a) and 3(b) show comparable probabilities for each of the three vectors. The three displayed probabilities do not add up to the value of 1 in the regime of fast sweeps, which indicates finite probabilities of also obtaining longer vectors.

Both fast and slow sweeps are thus of little use since slow sweeps favor returning the zero vector and fast sweeps result in essentially random results. There is, however, a regime of intermediate sweep durations in which the probability of obtaining the desired shortest nonzero vector is enhanced. In Fig. 3(a), this regime is found for $1 \lesssim \log T \lesssim 2$, and in Fig. 3(b) it is found for $7 \lesssim \log T \lesssim 10$. In this "Goldilocks zone" [29], the sweep is sufficiently fast to excite the system from its ground state but also sufficiently slow to avoid excitations to highly excited states.

A comparison of Figs. 3(a) and 3(b) also 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 makes the Bin encoded qubits clearly the preferable choice under the conditions simulated.

The superior solution probabilities demonstrated by Bin can also be explained in terms of the adiabatic theorem. For Bin, the minimum gap $\min_t(\Delta E)$ between the ground state and first excited state can become extremely small due to the large range of energy scales present in the qubit coupling coefficients. An example of such a gap for both types of encoding is shown in Fig. 4. The initial ($t/T = 0$) and final ($t/T = 1$) gaps coincide for both encodings, but the gap for the Bin encoded qubits (blue solid line) decreases very rapidly in the early stage of the dynamics. For $t/T \lesssim 0.2$ the gap becomes
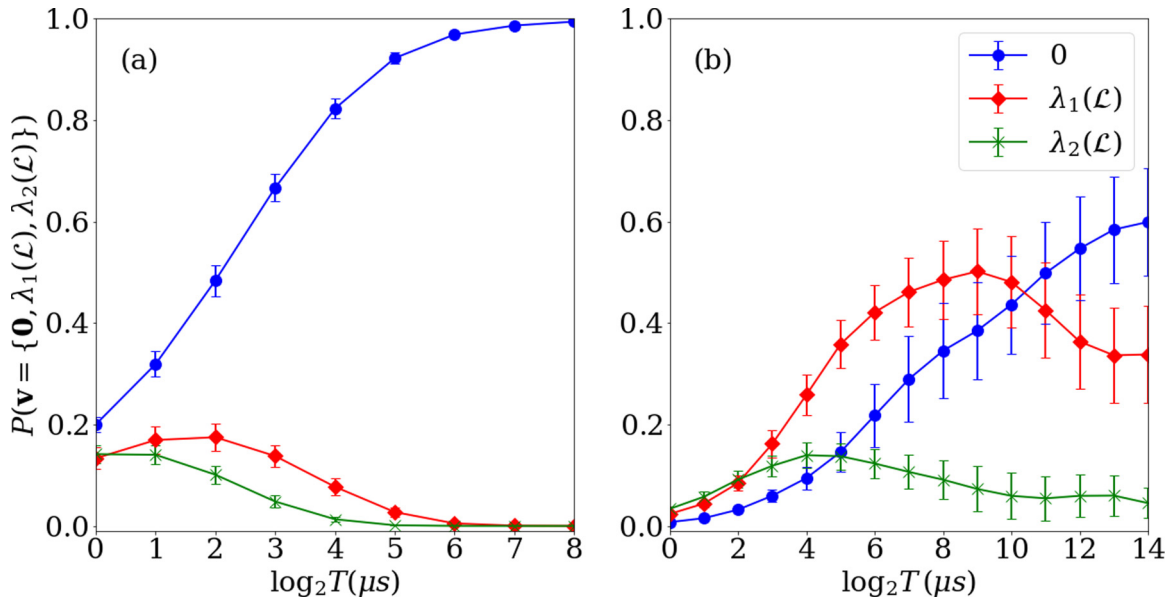
FIG. 3. Numerical experiments carried out on 20 three-dimensional lattices. The lines show mean final probabilities for measuring the system to be in the ground (blue circles) and first (red diamonds) and second (green crosses) excited eigenstates, grouped by degeneracy, at completion for sweep lengths 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 plots by vertical error bars.

much smaller than the minimal gap for Ham encoded qudits (red dashed line) which is obtained for $t/T \lesssim 0.6$. The smaller minimal gap for the Bin encoded qudits means that to achieve the same probability of attaining the ground state significantly slower sweeps are required (which for SVP purposes is advantageous, as we do not seek the ground state). This is because, as per adiabatic theory, a smaller minimum-energy gap means that excitations to the first excited state are more common. An algorithm based on Bin encoded qudits, furthermore, has lower space requirements. Thus, Bin is certainly the algorithm *for later*.

B. D-Wave quantum annealer

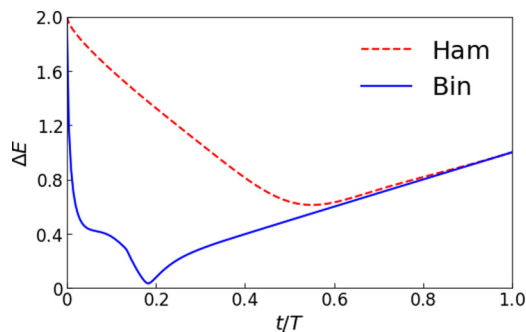The algorithms Bin and Ham, due to their Ising formulation, can be performed on the D-Wave 2000Q quantum

FIG. 4. Energy gap ΔE, in arbitrary units, between the ground state and the first excited state at each time t along a linear sweep of total duration T for the Ham algorithm (red dashed line) and the Bin algorithm (blue solid line). The minimum gap occurs much earlier for Bin and is much smaller.

annealer [30]. We examined the performance of the algorithms presented here experimentally using the D-Wave quantum processor. Interpreting these results requires a subtle understanding of the interplay between features of the algorithm and the mechanics of the quantum processing unit (QPU), discussed in Appendix B. In particular, we used the default embedding provided with the application program interface to map from the fully connected qubit graph specified by the theoretical model described in Sec. III (the logical qubits) and the QPU qubit graph (the physical qubits). In general, this embedding incurs a quadratic cost (logical qubits to physical qubits), resulting in a maximum system size for the experiments listed of 56 logical qubits, which maps (nondeterministically) to over 1000 physical qubits.

1. Representative example

Figure 5 depicts the result obtained for one specific (six-dimensional) lattice. The x axis depicts the lengths of vectors on a linear scale in Figs. 5(a) and 5(c) and on a logarithmic scale in Figs. 5(b) and 5(d). The blue bars represent the number of occurrences (y axis) of the corresponding lengths after 900 iterations of the algorithms. The green dashed vertical line indicates the lengths of the shortest nonzero vector, and the red dotted vertical lines show the length of the six basis vectors used to define the lattice.

Figures 5(a) and 5(b) show data obtained with Bin encoded qudits, and Figs. 5(c) and 5(d) show data obtained with Ham encoded qudits. In fact, Figs. 5(a) and 5(c) show the same data as Figs. 5(b) and 5(d), but Figs. 5(a) and 5(c) resolve the results of shorter vectors better, while Figs. 5(b) and 5(d) show the full distribution including the results for longer data.
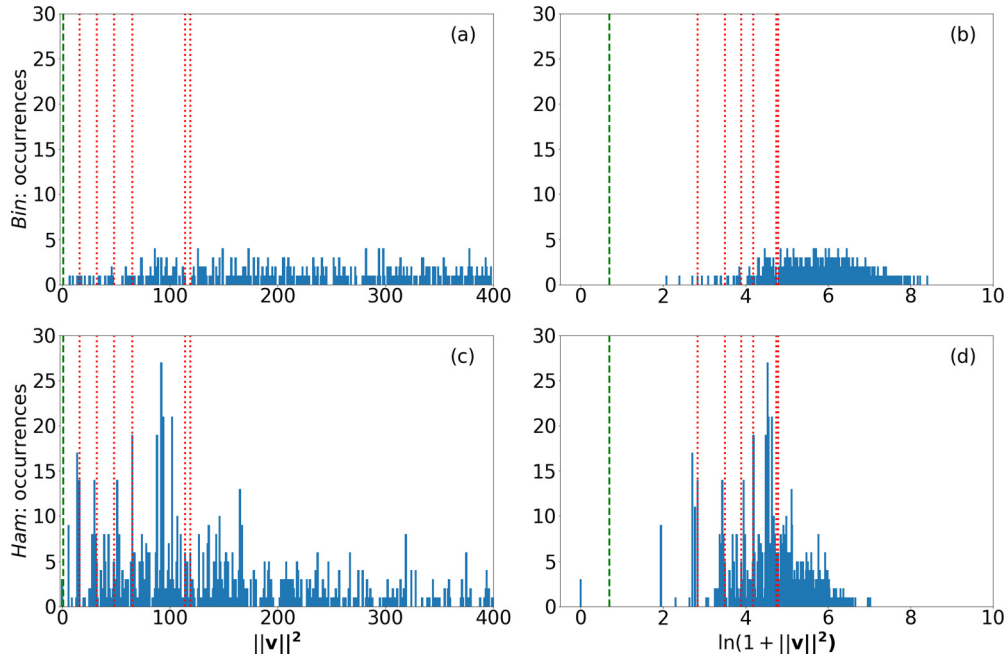
FIG. 5. Illustrative results from D-Wave for the (a) and (b) Bin and (c) and (d) Ham algorithms showing vector length squared on the x axis on the left and the natural logarithm of 1 plus vector length squared on the right, with bar height equal to the number of occurrences. The right-hand plots show the same data as the left-hand plots but allow an easier comparison at a glance. This is a sample experiment comprising 900 runs of each algorithm on the same six-dimensional lattice, all with a sweep length of $T = 64\ \mu$s. The green dashed verticals represent the length $\lambda_1(\mathcal{L})$ on the x axis, and the red dotted verticals represent the length of the six input basis vectors (from which **G** is generated) on the x axis. The heights of the green dashed and red dotted verticals have no significance.

A comparison of Figs. 5(b) and 5(d) shows that the majority of vector lengths obtained with the algorithm based on Bin encoded qudits are longer than all of the basis vectors, while the algorithm based on Ham encoded qudits returns substantially shorter vector lengths. This behavior is also manifest in Figs. 5(a) and 5(c), which show occurrences of the order of 10 for lengths shorter than the median basis vector, while the corresponding lengths in Bin are obtained between 1 and 3 times.

An ideal scenario (for all plots in Fig. 5) would be a single blue bar, with height 900, located on the x axis at the same location as the green dashed vertical. This scenario would indicate that the output of the algorithm was the shortest vector with probability 1. The farther to the right (longer) a blue bar is, the less useful the vectors it represents are. Bars farther right than *all* of the red dotted vertical lines are of no use as they are *longer* than all of the basis vectors which were input to the algorithm. Showing these results too, however, is useful as they help to compare the Ham algorithm with Bin, and in doing so one can see that Ham produces significantly shorter vectors on D-Wave hardware than Bin.

While Fig. 5 helps us to understand what the quantum annealing results look like for a representative example, as well as to qualitatively compare Ham with Bin, for rigorous performance analysis other visualizations of results are appropriate, which we look at next.

2. Aggregated results

Figure 6 shows a summary of results obtained for 19 lattices in dimensions $N = \{3, 4, 5, 6, 7\}$, according to four key

figures of merit (FOM). The experiments took 900 samples for each lattice instance with sweep times of $1\ \mu$s. The x axis shows the lattice dimension, while the y axis shows the probability of attaining the figure of merit corresponding to the particular plot. Probabilities for each lattice dimension are averaged over 19 lattices with standard error bars shown.

The red lines with circles show the results from Ham, and the green lines with crosses are for Bin in all plots. In Figs. 6(c) and 6(d) baseline probabilities from uniform random sampling of the Hilbert spaces are shown by black solid and dotted lines (Ham and Bin have different, but asymptotically close, baselines due to the slight difference in coefficients searched over).

Both algorithms suffer decreasing probabilities of success for all FOMs as lattice dimension increases, which is to be expected due to the increased complexity and larger Hilbert spaces to search over. One can see from Fig. 6 that the red lines with circles sit above the green lines with crosses in all plots. This indicates higher probabilities of success for Ham on the quantum annealing hardware for all FOMs and over all lattice dimensions.

Figure 6(a) shows the probability of the annealer returning the zero vector **0**. While Ham returns zero vectors in 4%–5% of runs for three-dimensional lattices, decreasing to almost 0% in higher dimensions, Bin displays much lower probabilities even in the lowest dimensions. The reason is the smaller minimum energy gap $\min_t \Delta E$ for Bin between the ground state and first excited state, a typical example of which is depicted in Fig. 4 and explained in Sec. IV A.
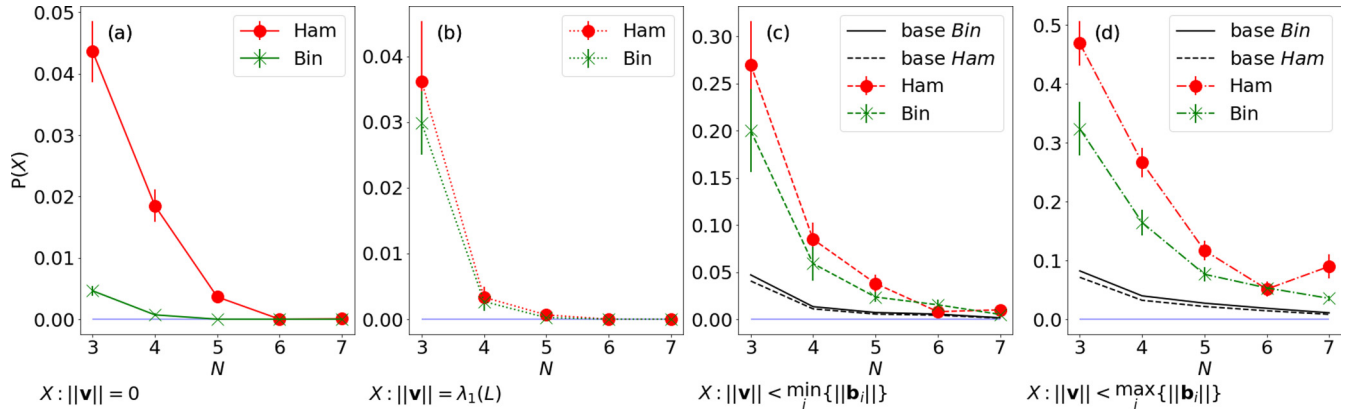
FIG. 6. $T = 1 \, \mu$s for the Ham (red circles) and Bin (green crosses) algorithms performed on 19 randomly generated "bad" lattice bases in dimensions 3 to 7 inclusive and for four different figures of merit: (a) ground state (not desired), (b) shortest vector (preferable), (c) shorter than minimum input basis vector, and (d) shorter than median input basis vector. The black lines are baselines indicating the results from uniform random sampling from the searched-over Hilbert space (dotted lines for Ham, solid lines for Bin), i.e., what proportion of lattice vectors in the solution space is shorter than minimum [in (c)] and median [in (d)] input basis vectors.

Zero vector probability may initially seem to be of no use because the zero vector is of no use computationally. It is best understood, however, in relation to other FOMs and the performance of the algorithm overall, as **0** results can be interpreted as a measure of "adiabaticity." Through this lens, it can be seen that Ham is far superior to Bin in returning ground states on the noisy quantum annealing hardware.

Figure 6(b) shows the probability of obtaining the shortest vector, which is the most important FOM. In Fig. 6(b), in contrast to Fig. 6(a), both algorithms perform similarly, though again with Ham producing slightly higher probabilities. Here, while in three dimensions both Ham and Bin return the shortest vector with 2.5%–5% chance, in dimensions 6 and 7 no shortest vectors were observed. This FOM is the one for which high probabilities are most desired, as these probabilities represent the chances that we can solve exact SVP.

It is easier to analyze ensembles of vectors as opposed to individual results, as these give probabilities which can be more easily analyzed within the constraints of 900 repetitions per lattice sample per algorithm per time sweep. This is where Figs. 6(c) and 6(d) offer insight.

The FOM in Fig. 6(c) is the probability of returned vectors having a length shorter than the shortest basis vector, and Fig. 6(d) gives probabilities of vectors having a length shorter than the median basis vector. Because the length requirements to satisfy these criteria are more relaxed than in Figs. 6(a) and 6(b), higher probabilities are observed for all points. Both Ham and Bin in Figs. 6(c) and 6(d) begin with an order of 10% success probability for three-dimensional lattices, followed by a smooth decay in mean success probabilities as lattice dimension increases. In Figs. 6(c) and 6(d), the decay in success probability is encouragingly less drastic than for the results in Figs. 6(a) and 6(b). The baselines in black show the proportion of the solutions in the entire Hilbert space that are shorter than the shortest and median basis vectors for Figs. 6(c) and 6(d), respectively.

While good results from Figs. 6(c) and 6(d) will not help us directly to solve SVP, they can be used to find basis vectors with which to update the input basis, resulting in a "shorter ba-

sis." This technique is known as "lattice basis reduction" [32] and iteratively finds shorter vectors by improving the basis and vice versa.

To understand how the two FOMs in Figs. 6(c) and 6(d) relate back to the representative example shown in Fig. 5, the *shorter than shortest basis vector* probability would be computed from the results in Fig. 5 by summing the heights of all the blue bars to the *left* of the leftmost red vertical, then dividing by 900 (total number of runs), and the *shorter than median basis vector* probability would be computed by summing the heights of the blue bars to the left of the middle (or median) red dotted vertical, then dividing by 900.

In Fig. 6(b) we stress that it is not important that the probability of obtaining λ_1 tends to zero as dimension increases (this is inevitable), but how fast this occurs is important. A polynomial decay, for example, would be catastrophic for LBC, whereas an exponential decay could even match the current state of the art for lattice algorithms. At this stage, not enough data points are available to heuristically estimate the rate of decay—for heuristic estimation, better hardware is required. The key takeaway from Fig. 6 is that Ham outperforms Bin at almost every data point across the four FOMs, demonstrating a significant improvement on Bin for the present regime (many qubits available but not high quality), cementing it as the choice *for now*.

V. CONCLUSION

The algorithms described in Sec. III go a way to establishing the vector optimization framework first proposed in [29], and the work described in Sec. IV signals emphatically that quantum cryptanalysis is drawing into the empirical realm and is no longer purely a theoretical endeavor. With an ever-increasing pace of development in quantum hardware, coupled with worst-case asymptotic scaling of $O(N \log N)$ for Bin, it is foreseeable that in the near future much larger experiments could be carried out to put quantum lattice algorithms to the test, be it on annealers such as D-Wave's as described here or gate architectures.

AQC usually suffers poor time scaling due to its dependence on $1/\Delta$ (where Δ is the minimum spectral gap between the ground and first excited eigenstates along the Hamiltonian path), which can grow very quickly as system size increases. The long sweep times of AQC are necessary in order to preserve the system in its ground state throughout the evolution. The existance of the Goldilocks zone far from the adiabatic limit raises in this work is encouraging as it hints that algorithms such as the ones described in Sec. III may achieve much more appealing time scaling.

In fact, this raises the curious question of how to approach extracting asymptotic time scaling for an algorithm where success is defined to be measuring the system in its first excited eigenstate. More generally, the approximate form of SVP, SVP_γ, requires only an attacker to find a vector of length polynomially [poly(N)] larger than $\lambda_1(\mathcal{L})$. This form of the problem means that poly(N) excitations are admissible during the evolution, which could potentially be traded off against significant speedups. Answering this question would doubtless be of interest well beyond postquantum cryptography, as the answer would unlock solutions to many approximate optimization problems in QC.

One key area of progress bearing significance for QC is that of improving the fidelity of qubits both in the annealer and gate architectures. In the meantime, as demonstrated above, it is down to theorists to think carefully about how their mathematical constructions might best make use of the hardware available to them as the significant improvements we extracted from Ham qudits were nearly dropped from consideration due to the asymptotic inefficiency in space of $O(N^2)$ versus $O(N \log N)$ for Bin qudits. In this way, it is possible for experimental and theoretical work to meet in the middle.

APPENDIX A: SVP INSTANCE GENERATION

Here we talk through the exact procedure for generating the lattice bases used to perform the quantum SVP algorithms described in Sec. III on.

For each dimension, we generated 20 random matrices with coefficients in $\{0, 1\}$ (ensuring nonzero determinants, so that corresponding lattices are full rank). These are the good bases. We then generated 20 random *unimodular* matrices with coefficients in $[-6, \dots, 6]$. The good bases (which are just a matrix with binary entries, interpreted as a row basis) are then postmultiplied by a unimodular matrix. The result is the set of bad bases. These are used as inputs to the algorithm (i.e., the bad bases are postmultiplied with their transpose to get the Gram matrix, which is used to define interqubit interactions).
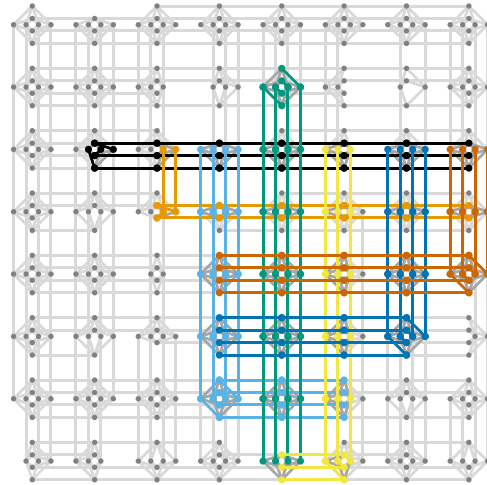

FIG. 7. Diagram, from [43], showing logical qubits embedded as qubit chains (of physical qubits) into the CHIMERA topology, where each qubit with the same color has strong ferromagnetic interactions with other qubits to which it is connected in the same chain.

The reason we generated good bases with short vectors before transforming them into bad bases to input into the algorithms in this paper is to ensure the existence of short vectors and to make them more closely resemble the SVP instances one is likely to encounter in the wild, by which we mean approaching bases for which short vectors are known to exist.

APPENDIX B: D-WAVE

The quantum Ising model assumes full qubit-qubit connectivity, whereas D-Wave 2000Q is constructed according to a CHIMERA topology, whereby each qubit is connected to a handful of nearby qubits that form a sort of cluster and each cluster is connected to a few others. Each cluster is represented as the diamond formation of dots in Fig 7. A model requiring full connectivity can be mapped into D-Wave's CHIMERA topology, incurring a quadratic cost in the number of qubits required. This is done by creating qubit "chains," which are illustrated in Fig 7. In a qubit chain, all qubits are strongly incentivized to return the same value by assigning qubits in the same chain with stronger qubit-qubit ferromagnetic interactions than between qubits in different chains. By way of error correction, when not all qubits in a chain return the same value (called a chain break), a simple majority vote is taken to decide on the final value.

Ham qudit advantage

The reason the Ham qudits trump the Bin qudits despite the larger system size required to search the same solution space is that they allow for greater utilization of the energy spectrum available. In the Bin setting, the strength of qubit-qubit interactions decreases on average by a factor of 2 for each step away from the most significant qubit. This result means that despite the instability of SVP solutions (by which we mean that a small error in the coefficient vector makes a large difference to the output eigenenergy) the least significant

qubits have relatively very weak interactions with the rest of the system, as *all* J, h interaction values must be scaled down to the energy spectrum provided. Crucially, this also means that errors in the interaction energies become much larger relative to the problem Hamiltonian, in effect meaning that it is much more likely that the QPU is solving *the wrong problem* [44], leading to poorer performance relative to Ham.

In Ham, however, each qubit contributes the same amount to the output of the qudit, and so small differences (± 1 in the value of a coefficient) are effected by stronger forces, increasing the likelihood of attaining low-energy solutions. We believe this effect is quite significant but tempered somewhat by the effects of having a larger system size: longer chains are required, which leads to higher chain-break probabilities, and thus, more errors occur.

[1] P. Benioff, The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines, J. Stat. Phys. **22**, 563 (1980).

[2] R. P. Feynman, Simulating physics with computers, Int. J. Theor. Phys **21**, 467 (1999).

[3] P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM Rev. **41**, 303 (1999).

[4] L. K. Grover, A fast quantum mechanical algorithm for database search, in *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA* (ACM, New York, 1996), pp. 212–219.

[5] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, Quantum computation by adiabatic evolution, arXiv:quant-ph/0001106.

[6] D. Aharonov, W. van Dam, J. Kempe, Z. Landau, S. Lloyd, and O. Regev, Adiabatic quantum computation is equivalent to standard quantum computation, SIAM J. Comput. **37**, 166 (2004).

[7] A. Aspuru-Guzik, R. Lindh, and M. Reiher, The matter simulation (R)evolution, ACS Cent. Sci. **4**, 144 (2018).

[8] R. Babbush, N. Wiebe, J. McClean, J. McClain, H. Neven, and G. K.-L. Chan, Low-Depth Quantum Simulation of Materials, Phys. Rev. X **8**, 11044 (2018).

[9] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, Fernando G. S. L. Brandao, and D. A. Buell, Quantum supremacy using a programmable superconducting processor, Nature (London) **574**, 505 (2019).

[10] A. K. Lenstra and W. Hendrik, Jr., *The Development of the Number Field Sieve*, Vol. 1554 of Lecture Notes in Mathematics (Springer, Berlin, Heidelber, 1993).

[11] R. L. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Commun. ACM **21**, 120 (1978).

[12] D. Bernstein, J. Buchmann, and E. Dahmen, *Introduction to Post-quantum Cryptography* (Springer, Berlin, Heidelberg, 2009).

[13] W. Diffie and M. Hellman, New directions in cryptography, IEEE Trans. Inf. Theory **22**, 644 (1976).

[14] O. Regev, On lattices, learning with errors, random linear codes, and cryptography, in *Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing*, Vol. 56 (ACM, New York, 2009), pp. 1–40.

[15] C. Gentry, Fully homomorphic encryption using ideal lattices, in *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing, Bethesda, Maryland* (ACM, New York, 2009), pp. 169–178.

[16] H. Jeffrey, J. Pipher, and J. H. Silverman, International Algorithmic Number Theory Symposium, in *Proceedings of the third International Symposiun, ANTS-III Portland, Oregon, USA,*

Lecture Notes in Computer Science Vol. 1423 (Springer-Verlag, Berlin, Heidelberg, 1998), pp. 267–288.

[17] O. Goldreich, S. Goldwasser, and S. Halevi, Public-Key Cryptosystems from Lattice Reduction Problems, in *Advances in Cryptology – CRYPTO '97*, Lecture Notes in Computer Science Vol. 1294 (Springer, Berlin, Heidelberg, 1997), pp. 112–131.

[18] M. Ajtai, R. Kumar, and D. Sivakumar, A sieve algorithm for the shortest lattice vector problem, in *Proceedings of the 33rd ACM Symposium on Theory of Computing (STOC)*, Lecture Notes in Computer Science Vol. 2146 (ACM, New York, 2001), pp. 601–610.

[19] U. Fincke and M. Pohst, Improved methods for calculating vectors of short length in a lattice, including a complexity analysis, Math. Comput. **44**, 463 (1985).

[20] R. Kannan, Improved algorithms for integer programming and related lattice problems, in *Proceedings of 15th ACM Symposium on Theory of Computing, STOC* (ACM, New York, 1983), pp. 193–206.

[21] N. Gama, P. Q. Nguyen, and O. Regev, Lattice enumeration using ExtremePruning, in *EUROCRYPT 2010*, Lecture Notes in Computer Science Vol. 6110 (Springer-Verlag, Berlin, Heidelberg, 2010), pp. 257–278.

[22] T. Laarhoven, M. Mosca, and J. Van De Pol, Solving the shortest vector problem in lattices faster using quantum search, in *PQCrypto* (Springer-Verlag, Berlin, Heidelberg, 2013), pp. 83–101.

[23] A. Montanaro, Quantum walk speedup of backtracking algorithms, Theory Comput. **14**, 1 (2018).

[24] A. Montanaro, Quantum speedup of branch-and-bound algorithms, Phys. Rev. Research **2**, 13056 (2020).

[25] Y. Aono, P. Q. Nguyen, and Y. Shen, Quantum lattice enumeration and tweaking discrete pruning, in *Advances in Cryptology ASIACRYPT 2018* (Springer, Cham, 2018).

[26] Z. Brakerski, E. Kirshanova, D. Stehlé, and W. Wen, Learning with errors and extrapolated dihedral cosets, *Public-Key Cryptography – PKC 2018*, Lecture Notes in Computer Science Vol. 10769 (Springer, Cham, 2018), pp. 702–727.

[27] K. Eisenträger, S. Hallgren, A. Kitaev, and F. Song, A quantum algorithm for computing the unit group of an arbitrary degree number field, in *Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing* (ACM, New York, 2014), pp. 293–302.

[28] R. Cramer, L. Ducas, and B. Wesolowski, Short Stickelberger class relations and application to ideal-SVP, *Advances in Cryptology – EUROCRYPT 2017 Part I*, Lecture Notes in Computer Science Vol. 10210 (Springer, Cham, 2017), p. 324.

[29] D. Joseph, A. Ghionis, C. Ling, and F. Mintert, Not-so-adiabatic quantum computation for the shortest vector problem, Phys. Rev. Research **2**, 013361 (2020).

[30] C. C. McGeoch, R. Harris, S. P. Reinhardt, and P. I. Bunyk, Practical annealing-based quantum computing, Computer **52**, 38 (2019).

[31] B. Helfrich, Algorithms to construct Minkowski reduced and Hermite reduced lattice bases, Theor. Comput. Sci. **41**, 125 (1985).

[32] A. K. Lenstra, H. W. Lenstra, and L. Lovász, Factoring polynomials with rational coefficients, Math. Ann. **261**, 515 (1982).

[33] C. P. Schnorr and M. Euchner, Lattice basis reduction: Improved practical algorithms and solving subset sum problems, Math. Program. **66**, 181 (1994).

[34] T. Kadowaki and H. Nishimori, Quantum annealing in the transverse Ising model, Phys. Rev. E **58**, 5355 (1998).

[35] A. Callison, N. Chancellor, F. Mintert, and V. Kendon, Finding spin glass ground states using quantum walks, New J. Phys. **21**, 123022 (2019).

[36] A. M. Childs and J. Goldstone, Spatial search by quantum walk, Phys. Rev. A **70**, 022314 (2004).

[37] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm, arXiv:1411.4028.

[38] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, W.-K. Mok, S. Sim, L.-C. Kwek, and A. Aspuru-Guzik, Noisy intermediate-scale quantum (nisq) algorithms, arXiv:2101.08448.

[39] J. Preskill, Quantum computing in the nisq era and beyond, Quantum **2**, 79 (2018).

[40] T. Plantard, W. Susilo, and K. T. Win, A digital signature scheme based on cvp, in *International Workshop on Public Key Cryptography* (Springer-Verlag, Berlin, Heidelberg, 2008), pp. 288–307.

[41] M. Rose, T. Plantard, and W. Susilo, Improving BDD cryptosystems in general lattices, *Information Security Practice and Experience*, Lecture Notes in Computer Science Vol. 6672 (Springer, Berlin, Heidelberg, 2011), pp. 152–167.

[42] H. Minkowski, *Geometrie der zahlen* (Ripol Classic, Moscow, 2016), Vol. 2.

[43] T. Boothby, A. D. King, and A. Roy, Fast clique minor generation in Chimera qubit connectivity graphs, Quantum Inf. Process. **15**, 495 (2016).

[44] K. C. Young, R. Blume-Kohout, and D. A. Lidar, Adiabatic quantum optimization with the wrong Hamiltonian, Phys. Rev. A **88**, 062314 (2013).