Lattices, Gaussians and Quantum Factoring Algorithms

Lecturer: Vinod Vaikuntanathan

1 Introduction

Lattice problems and quantum computation have had an interesting bidirectional relationship ever starting from Oded Regev's work [Reg02] which showed that (an important variant of) the classic shortest vector problem on lattices reduces, via a quantum reduction, to solving a hidden subgroup problem on *dihedral groups*. Many quantum algorithms at that time which achieved superpolynomial speedup (including Shor's algorithm for factoring) were solving hidden subgroup problem on Abelian groups; the dihedral group is non-Abelian. We will see this reduction later on in the course.

Since then, lattices have emerged as a powerful tool in quantum computation and cryptography for several reasons:

- On the destructive side, lattices have recently played a key role in improving quantum algorithms that solve the factoring and discrete logarithm problems, via Regev's result [Reg24] and its followups [RV24, EG24].
- Lattices give us computationally hard problems that seem to be (average-case) hard even for quantum algorithms, giving us a basis of *post-quantum* public-key cryptography. Indeed, lattice-based public-key encryption is essentially the only game in town when it comes to post-quantum secure public-key encryption.
- Thirdly, lattices and the cryptographic objects coming out of them have played a key role in proofs of quantumness, methods of generating certified randomness, as well as protocols that enable a classical algorithm to verify that a quantum computation was done correctly. This has happened both directly as in [BCM$^+$21] and indirectly through the use of techniques originally developed in the context of lattices [YZ22].

We will explore all these facets in the next few lectures. This lecture will start with an introduction to lattices and a description of Shor's factoring algorithm viewed through the lattice lens. In turn, this view will help us seamlessly derive Regev's improved quantum factoring algorithm.

2 Lattices

An n-dimensional lattice $\mathcal{L} \subseteq \mathbb{R}^n$ is a discrete additive subgroup of $\mathbb{R}^n$. The lattice is full-rank if $\mathsf{Span}_{\mathbb{R}}(\mathcal{L}) = \mathbb{R}^n$. A (full-rank) n-dimensional lattice is generated by n basis vectors $b_1, \ldots, b_n \in \mathbb{R}^n$ and is the set of all *integer* linear combination of $b_1, \ldots, b_n$.

For example, in one dimension, every lattice is the set of all multiples of some $r \in \mathbb{R}$. In two dimensions, a canonical lattice is $\mathbb{Z}^2$, generated by the the basis vectors $b_1 = (1 \;\; 0)^T$ and $b_2 = (0 \;\; 1)^T$. We will often write them in a matrix the columns of which are the basis vectors. That is,

$$B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Then,

$$\mathcal{L} = \mathcal{L}(B) = B \cdot \mathbb{Z}^2 = \{Bz \, : \, z \in \mathbb{Z}^2\}$$

It turns out that the same lattice can be generated by multiple (indeed, infinitely many) bases. For example,

$$B' = \begin{pmatrix} 100 & 101 \\ 99 & 100 \end{pmatrix}$$

generates the same lattice: $\mathcal{L}(B') = \mathcal{L}(B) = \mathbb{Z}^2$.

There is a simple and efficient test to determine if two bases generate the same lattice.

Lemma 1. *$B \in \mathbb{R}^{n\times n}$ and $B' \in \mathbb{R}^{n\times n}$ generate the same lattice if there is a matrix $U \in\in \mathbb{Z}^{n\times n}$ such that (a) $|\mathsf{det}(U)| = 1$ and (b) $B' = BU$. (Such a matrix U is called a unimodular matrix.)*

The Dual Lattice. Given a lattice $\mathcal{L}$, its dual, denoted $\mathcal{L}^\perp$ or $\mathcal{L}^*$, is the set of all vectors in $\mathbb{R}^n$ whose inner product with every lattice vector is an integer. That is,

$$\mathcal{L}^* = \{x \in \mathbb{R}^n \, : \, \forall y \in \mathcal{L}, \langle x, y\rangle \in \mathbb{Z}\}$$

For example, if $\mathcal{L}$ is the one-dimensional lattice which is the set of all multiples of z, $\mathcal{L}^*$ is the set of all multiples of $1/z$. As another example, the dual of $\mathbb{Z}^n$ is $\mathbb{Z}^n$ itself. If $\mathcal{L}$ is generated by (the columns of) a basis matrix $\mathbf{B} \in \mathbb{R}^{n\times n}$, $\mathcal{L}^*$ is generated by (the columns of) B^{-T}.

3 Computational Problems on Lattices

The length of the shortest non-zero vector in the lattice is denoted $\lambda_1(\mathcal{L})$. (Throughout this course, we will use the Euclidean norm to measure lengths.)

As we saw above, a lattice can have many bases. Given an arbitrary basis of a lattice, potentially one with long vectors, can we find the shortest vector in the lattice? how about a nearly shortest one? how about just the length of the shortest vector? All these problems, to the best of our knowledge, are computationally hard. Indeed, their hardness seems to grow exponentially with the dimension n.

The shortest vector problem SVP. Given a lattice basis B, find a non-zero vector v such that $||v|| = \lambda_1(\mathcal{L}(B))$.

The approximate shortest vector problem SVP_α, $\alpha \geq 1$. Given a lattice basis B, find a non-zero vector v such that $||v|| \leq \alpha \cdot \lambda_1(\mathcal{L}(B))$. If $\alpha = 1$, this is the (exact) shortest vector problem.

The gap shortest vector problem gapSVP_α, $\alpha \geq 1$. This is a promise problem wherein you are given a lattice basis B, and you should output YES if $\lambda_1(\mathcal{L}(B)) \leq 1$ and NO if $\lambda_1(\mathcal{L}(B)) > \alpha$. (If the promise doesn't hold, you are free to output either YES or NO.)

Other problems of interest include the closest vector problem and the bounded distance decoding problem which we will see in later lectures.

For the exact shortest vector problem, the best known algorithms run in time $2^{O(n)}$. It is also NP-hard, and even ETH-hard for odd norms. On the other hand, the celebrated Lenstra-Lenstra-Lovasz algorithm solves the $2^{n/2}$-approximate shortest vector problem in time $\mathsf{poly}(n, ||B||)$ where $||B||$ is the description length of the input basis B. A smooth tradeoff between the approximation factor and running time is known, and is due to Schnorr: one can find a 2^k-approximate shortest vector in time $2^{\tilde{O}(n/k)}$.

4 Gaussians and the Smoothing Lemma

The Gaussian function $\rho : \mathbb{R}^n \to \mathbb{R}$ with center (mean) $c \in \mathbb{R}^n$ and standard deviation $\sigma \in \mathbb{R}$ is given by

$$\rho_{c,s}(x) = e^{-\pi(x-c)^2/s^2}$$

This can be turned into a probability distribution over $\mathbb{R}^n$ by normalizing appropriately:

$$\mathcal{N}_{c,s}(x) = \frac{1}{s^n} \cdot e^{-\pi(x-c)^2/s^2}$$

Gaussian random variables are tightly concentrated around their mean. For example, if $x \sim \mathcal{N}_{c,s}$,

$$\mathbb{E}[\|x-c\|_2] = s\sqrt{n} \quad \text{and} \quad \Pr[\|x-c\| > \alpha s\sqrt{n}] = e^{-\Omega(n)} \text{ for any } \alpha > 1/\sqrt{2\pi}.$$

A *discrete* Gaussian probability distribution over a lattice $\mathcal{L}$ is defined as

$$\mathcal{D}_{\mathcal{L},c,s}(x) = \begin{cases} 0 & \text{if } x \notin \mathcal{L} \\ \frac{\rho_{c,s}(x)}{\sum_{x\in\mathcal{L}} \rho_{c,s}(x)} & \text{if } x \in \mathcal{L} \end{cases}$$

(Note that a pre-requisite for this function being well-defined is that the denominator is finite, but this turns out to be the case as its upper-bounded by the Gaussian integral which is finite.)

The Smoothing Lemma. This is a very important lemma which deals with the following process: imagine a sum of Gaussians, one centered at every lattice point. In other words, the function $\mathcal{S}_s : \mathbb{R}^n \to \mathbb{R}$ defined as

$$\mathcal{S}_s(x) = \sum_{z\in L} e^{-\pi\|x-z\|^2/s^2}$$

This is clearly a periodic function, which can be equivalently described simply by looking at its action on a fundamental parallelopiped of the lattice, namely

$$\mathcal{P}(B) := B \cdot [0,1)^n$$

(As an example, the fundamental parallelopiped of the one-dimensional lattice $\mathbb{Z}$ is the "torus" $[0, 1)$.) So, we will henceforth let $\mathcal{S}_s : \mathcal{P}(B) \to \mathbb{R}$ defined as above.

It seems intuitive that the larger s is, the more uniform S_s is. The smoothing lemma gives us a quantitative bound on when this happens.

Lemma 2. *For any n-dimensional lattice $\mathcal{L}$, $k \geq 1$, and $s \geq k\sqrt{n} \cdot \lambda_n(\mathcal{L})$,*

$$\mathsf{TV}(\mathcal{S}_s, \mathcal{U}_{\mathcal{P}(B)}) = 2^{-\Omega(k^2 n)}$$

where TV *is the total variation distance.*

The smoothing lemma has several nice consequences. For example, consider a discrete Gaussian distribution over a lattice (with center at 0) and keep increasing its standard deviation. This morally has the same effect as keeping the standard deviation the same and shrinking the lattice by the same factor. At some point, one expects the discrete Gaussian to start "behaving like" a continuous Gaussian. That point happens to be the standard deviation defined by the smoothing lemma.

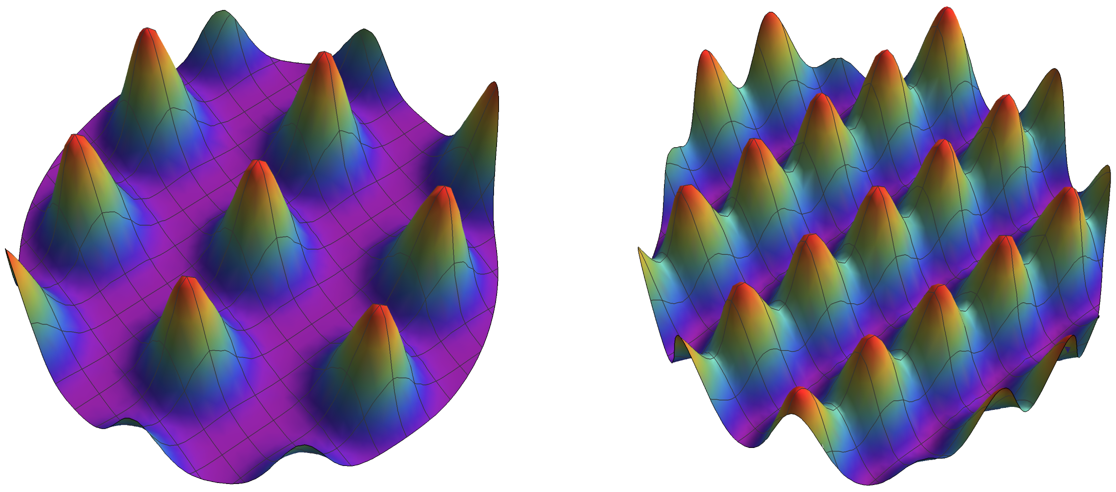

Figure 1: The periodic Gaussian function $\mathcal{S}_s(x)$ for two different two-dimensional lattices $\mathcal{L}$.

Lemma 3. *For any n-dimensional lattice $\mathcal{L}$, $k \geq 1$, and $s \geq \sqrt{kn} \cdot \lambda_n(\mathcal{L})$ and any $c \in \mathbb{R}^n$,*

$$\frac{\sum_{x\in\mathcal{L}} e^{-\pi\|x\|^2/s^2}}{\sum_{x\in\mathcal{L}} e^{-\pi\|x-c\|^2/s^2}} \in \left[1 - 2^{-\Omega(kn)}, 1 + 2^{-\Omega(kn)}\right]$$

and

$$\frac{\sum_{x\in\mathcal{L}} e^{-\pi\|x\|^2/s^2}}{\int_{\mathbb{R}^n} e^{-\pi\|x\|^2/s^2}\,dx} \in \left[1 - 2^{-\Omega(kn)}, 1 + 2^{-\Omega(kn)}\right]$$

Furthermore, since the Gaussian integral $\int_a^b e^{-\pi x^2/s^2}\,dx$ is efficiently approximable (e.g. using an appropriate expansion of the error function erf), we can also compute approximations of $\sum_{x\in\mathcal{L}} e^{-\pi\|x\|^2/s^2}$.

Preparing a discrete Gaussian State. We will be interested in preparing the discrete Gaussian state

$$\sum_{x\in\mathbb{Z}^n} \sqrt{\rho_s(x)}\,|x\rangle \tag{1}$$

Slightly more precisely, we will be interested not in the entirety of $\mathbb{Z}^n$, rather the ball $\mathbb{Z}^n \cap [-s\sqrt{kn}, s\sqrt{kn}]^n$:

$$\sum_{x\in\mathbb{Z}^n\cap[-s\sqrt{kn},s\sqrt{kn}]^n} \sqrt{\rho_s(x)}\,|x\rangle$$

These two states are close in trace distance, so we will consider them the same going forward. The Grover and Rudolph [GR02] algorithm helps us prepare exponentially good approximations of the discrete Gaussian state.

Lemma 4. *For any $n, k \geq 1$ and $s \geq k\sqrt{n}$, there is a $\mathsf{poly}(n, k)$-time quantum algorithm that prepares the discrete Gaussian state (1).*

Proof. While the Grover-Rudolph algorithm prepares a quantum state encoding a fairly general class of distributions, we now give a sense of how it works for the discrete Gaussian. To prepare the n-dimensional state (1), it suffices to prepare n one-dimensional discrete Gaussian states

$$\sum_{x\in\mathbb{Z}^n\cap[-sk,sk]} \sqrt{\rho_s(x)}\,|x\rangle = \sum_{x\in\mathbb{Z}^n\cap[-sk,sk]} \rho_{s\sqrt{2}}(x)\,|x\rangle$$

Consider preparing a state

$$\sum_{i\in[2^\ell]} \sqrt{p(i)}\,|i\rangle$$

for some ℓ and probability distribution $p : [2^\ell] \to [0,1]$. We will do this one qubit at a time for a total of ℓ qubits. Start with $|0\rangle$. Compute

$$\theta := \mathsf{arccos}(\sqrt{\alpha})$$

where $\alpha = \sum_{i\in[1\dots 2^{\ell-1}]} p(i)$. By assumption, α can be computed quickly, in time $\mathsf{poly}(\ell)$. Perform a rotation by θ to get

$$\cos\theta\,|0\rangle + \sin\theta\,|1\rangle = \sqrt{\alpha}\,|0\rangle + \sqrt{1-\alpha}\,|1\rangle$$

which is exactly the right "marginal" state of the first qubit. In general, after the k-th iteration, one has

$$\sum_{j\in[2^k]} \sqrt{p_k(j)}\,|j\rangle$$

where $p_k(j) = \sum_{i=j\star} p(i)$. That is, sum up $p(i)$ over all i whose length-k prefix is j. Indeed, looking at the first iteration, $p_1(0) = \alpha$ and $p_1(1) = 1-\alpha$. Now, compute in superposition $\theta_j := \mathsf{arccos}(\sqrt{p_{k+1}(j0)/p_k(j)})$ to get

$$\sum_{j\in[2^k]} \sqrt{p_k(j)}\,|j\rangle\,|\theta_j\rangle\,|0\rangle$$

Do a θ_j-rotation on the last qubit and uncompute the θ_j register. What's left is precisely

$$\sum_{j\in[2^k]} \sqrt{p_{k+1}(j0)}\,|j0\rangle + \sqrt{p_k(j) - p_{k+1}(j0)}\,|1\rangle = \sum_{j\in[2^{k+1}]} \sqrt{p_{k+1}(j)}\,|j\rangle$$

Continuing this until $k = \ell$ gets us precisely the state we want. $\square$

5 Fourier Transform

The Fourier transform of a (nice)[1] function $f : \mathbb{R}^n \to \mathbb{R}$ is a function $\hat{f} : \mathbb{R}^n \to \mathbb{C}$ defined as

$$\hat{f}(y) = \int_{\mathbb{R}^n} f(x)\,e^{-2\pi i\langle x,y\rangle}dx$$

Consider functions f that are periodic over $\mathbb{Z}^n$; that is for every $x \in \mathbb{R}^n$ and $z \in \mathbb{Z}^n$, $f(x+z) = f(x)$. Then, and only then, its Fourier transform $\hat{f}$ is supported on $\mathbb{Z}^n$. This is not hard to see:

$$\begin{aligned}
\hat{f}(y) = \int_{\mathbb{R}^n} f(x)\,e^{-2\pi i\langle x,y\rangle}dx &= \int_{x\in\mathbb{T}^n} \sum_{z\in\mathbb{Z}^n} f(x+z)\,e^{-2\pi i\langle x+z,y\rangle}dx \\
&= \int_{x\in\mathbb{T}^n} \sum_{z\in\mathbb{Z}^n} f(x)\,e^{-2\pi i\langle x+z,y\rangle}dx \\
&= \int_{x\in\mathbb{T}^n} f(x)\,e^{-2\pi i\langle x,y\rangle}dx \cdot \left(\sum_{z\in\mathbb{Z}^n} e^{-2\pi i\langle z,y\rangle}\right)
\end{aligned}$$

The latter sum is non-zero only when $y \in \mathbb{Z}$.

[1] "Nice" means Lebesgue integrable, that is, $\int_{\mathbb{R}^n} |f(x)|\,dx < \infty$.

The Fourier transform of the Gaussian function turns out to be a Gaussian function itself. In particular,

$$\hat{\rho}_s(x) = s^n \rho_{1/s}(x)$$

The Fourier transform of a *discrete* Gaussian function over a lattice is a superposition of Gaussians, one centered on each dual lattice point. An intuitive way to see it is the following: a discrete Gaussian function is the pointwise product of a continuous Gaussian function ρ_s and the indicator function $I_{\mathcal{L}}$ of the lattice (i.e. a function that evaluates to 1 on all lattice points and 0 elsewhere). Its Fourier transform, therefore, is the convolution of the Fourier transform of the Gaussian, which is $s^n \rho_{1/s}$, and the Fourier transform of $I_{\mathcal{L}}$, which turns out to be $I_{\mathcal{L}^*}$. This is precisely a sum of many Gaussians, one centered on each lattice point, that is, the function $\mathcal{S}_{1/s}$ we encountered above (scaled appropriately).

6 Shor's Algorithm for Factoring

Shor's algorithm for factoring consists of a quantum step followed by a classical post-processing. Our presentation of Shor's algorithm is nearly identical to conventional presentations in the quantum step, but will differ in the classical post-processing. In particular, we will formulate the classical post-processing as a lattice problem, leading us naturally to Regev's algorithm in the next lecture.

For the rest of this exposition, we focus on factoring numbers $N = PQ$ which are a product of two safe prime numbers, that is, $P' := (P-1)/2$ and $Q' := (Q-1)/2$ are themselves prime. Such numbers N are the common moduli that appear in cryptographic applications. This restriction is not important, but is only for the ease of presentation.

Reduction to Order Finding.

- Order of $\mathbb{Z}_N^*$ is $\varphi(N) = (P-1)(Q-1)$, the Euler totient function of N.
- If you find $\varphi(N)$, you can factor. Indeed,

  $$\varphi(N) = (P-1)(Q-1) = N - P - Q + 1 = N - P - N/P + 1$$

  which is a quadratic equation in P, solving which gives us P. (This reduction from factoring to finding $\varphi(N)$ is true for general N too.)
- Roughly a quarter of the elements in $\mathbb{Z}_N^*$ have order $\varphi(N)/4$. Pick a random $a \in \mathbb{Z}_N^*$ and let r be its order. Finding the order of a allows us to factor N.

The Quantum Algorithm.

1. Prepare a discrete Gaussian state

   $$|\psi_1\rangle \propto \sum_{x\in\mathbb{Z}} e^{-\pi x^2/s^2} |x\rangle$$

   for a value of s to be determined. As long as s is moderately large (which it will be), this can be done using the Grover-Rudolph algorithm (see Lemma 2).

   It is worth noting that this state cannot be prepared as-is since its domain is infinite. In reality, we will prepare $|\psi_1'\rangle$ where

   $$|\psi_1'\rangle \propto \sum_{x\in\mathbb{Z}\cap[-(D-1)/2,(D-1)/2]} e^{-\pi x^2/s^2} |x\rangle$$

for a large enough $D \approx s\sqrt{k}$. This is $2^{-\Omega(k)}$-close in trace distance to $|psi_1\rangle$. We will identify $[-(D-1)/2, (D-1)/2]$ with $\mathbb{Z}_D$. However, for the rest of this exposition, we will ignore this inconvenient fact.

2. Compute $a^x \bmod N$ in a second register (in superposition)

$$|\psi_2\rangle \propto \sum_{x\in\mathbb{Z}} e^{-\pi x^2/s^2} |x\rangle |a^x \bmod N\rangle$$

3. Measure the second register to get

$$|\psi_3\rangle \propto \sum_{x\in j+r\mathbb{Z}} e^{-\pi x^2/s^2} |x\rangle$$

   for some j (indeed, if measuring the second register resulted in $b \in \mathbb{Z}_N^*$, we know that $a^j = b \bmod N$).

4. Do a Quantum Fourier transform (over $\mathbb{R}$) to get

$$|\psi_4\rangle \propto \sum_{y\in\mathbb{R}} \Big(\sum_{z\in\mathbb{Z}} e^{-\pi s^2 (y-\frac{z}{r})^2} \Big) |y\rangle$$

   The j factor vanishes as it goes into the global phase. In reality, you have to do a QFT over $\mathbb{Z}_D$ but again, we will conveniently ignore it for now.

5. Measure to get a point close to $\frac{z}{r}$, for some $z \in \mathbb{Z}$. Indeed, the distance of the measured point to the closest multiple of $1/r$ will likely be $\approx 1/s$, so we will set $s \gg r$.

Classical Post-Processing. Do the quantum procedure several times (it turns out $O(1)$ times suffice) to get a noisy integer multiple of $1/r$. That is,

$$w = \frac{z}{r} + \eta \quad \text{or equivalently} \quad rw - z = r\eta := \delta$$

where $|\eta| \approx 1/s$. The post-processing question now is: given such points, can one determine $r \in \mathbb{Z}$? And how?

We will set this up naturally as a lattice problem. Consider the lattice generated by the columns of the matrix

$$M := \begin{pmatrix} \varepsilon & 0 \\ w & -1 \end{pmatrix}$$

for a value of ε we will set later.

What we are asking for is the shortest vector in this lattice which is defined by a coefficient vector v such that the norm of Mv is small. Recall that since the lattice is two-dimensional, the shortest vector can be found efficiently. We will set

- $s > N^4$; and
- $\varepsilon = 1/N^4$.

Indeed, when $v := (r, z)^T$

$$\|Mv\| = r\sqrt{\varepsilon^2 + \eta^2} \leq r/N^4 \cdot \sqrt{2} \leq \sqrt{2}/N^3 \ .$$

What if there are spurious short vectors? Consider a coefficient vector $v' = (r' \ \ z')^T$ where $r' \neq r$ and

$$\|Mv'\| \leq \sqrt{2}/N^3$$

First of all, it must be the case that

$$\varepsilon r' \leq \sqrt{2}/N^3$$

so that $r' \leq \sqrt{2}N$. Setting $r'w - z' = \delta'$, so that $|\delta'| \leq \sqrt{2}/N^3$ as well, we have

$$\frac{z_i'}{r'} - \frac{z_i}{r} = \frac{\delta_i}{r} - \frac{\delta_i'}{r'}$$

Assuming that r and r' are relatively prime, the absolute value of the LHS is

$$\left|\frac{z_i'}{r'} - \frac{z_i}{r}\right| = \left|\frac{z_i' r - z_i r'}{rr'}\right| \geq \frac{1}{rr'} \geq \frac{1}{\sqrt{2}N^2}$$

On the other hand, the absolute value of the RHS is

$$\left|\frac{\delta_i}{r} - \frac{\delta_i'}{r'}\right| \leq \left|\frac{\delta_i}{r}\right| + \left|\frac{\delta_i'}{r'}\right| \leq |\delta_i| + |\delta_i'| \leq (1 + \sqrt{2})/N^3$$

which is a contradiction.

References

[BCM+21] Zvika Brakerski, Paul Christiano, Urmila Mahadev, Umesh Vazirani, and Thomas Vidick. A cryptographic test of quantumness and certifiable randomness from a single quantum device, 2021.

[EG24] Martin Ekerå and Joel Gärtner. Extending regev's factoring algorithm to compute discrete logarithms, 2024.

[GR02] Lov Grover and Terry Rudolph. Creating superpositions that correspond to efficiently integrable probability distributions, 2002.

[Reg02] Oded Regev. Quantum computation and lattice problems. In *43rd Symposium on Foundations of Computer Science (FOCS 2002), 16-19 November 2002, Vancouver, BC, Canada, Proceedings*, pages 520–529. IEEE Computer Society, 2002.

[Reg24] Oded Regev. An efficient quantum factoring algorithm, 2024.

[RV24] Seyoon Ragavan and Vinod Vaikuntanathan. Space-efficient and noise-robust quantum factoring, 2024.

[YZ22] Takashi Yamakawa and Mark Zhandry. Verifiable quantum advantage without structure, 2022.