Practical Quantum Algorithms for Cryptanalysis

Ben Priestley

February 6, 2024

Introduction and Motivation

- Currently used public-key cryptosystems (namely, RSA) are built on hard problems like **integer factorisation**. This makes them classically intractable and, hence, classically secure.
- Shor's seminal algorithm can factor semiprimes in polynomial time, making these cryptosystems **quantumly vulnerable**.
  - Shor's algorithm needs fault-tolerance; or
  - Current estimates call for roughly 20 million (noisy) physical qubits to break RSA-2048.
- Recent work has inspired a **variational approach** to factoring that may escalate the threat to cryptosystems.
- Specifically, bold claims have been made about **lattice-based** factoring methods and whether they can offer a quantum advantage for factoring on modern-day "NISQ" devices.

Preliminaries

Factoring by Sieving

- Suppose we have an integer N to be factored. If we obtain x, y with $x \not\equiv \pm y \bmod N$ satisfying $x^2 \equiv y^2 \bmod N$, then $\gcd(x \pm y, N)$ are non-trivial factors of N.
- Solving this congruence involves a process called **sieving**: searching for "smooth integers". This is the main bottleneck in factoring.

p_m-smooth numbers

A number is p_m-smooth if all of its prime factors belong to the corresponding 'factor basis' $P = \{p_1, \ldots, p_m\}$, where p_i is the i-th prime.

- We need to find $m + 1$ smooth relation (sr)-pairs (u_j, v_j); a pair of p_m-smooth integers u_j and v_j such that

$$u_j = \prod_{i=1}^{m} p_i^{e_{i,j}} \quad \text{and} \quad u_j - v_j N = \prod_{i=0}^{m} p_i^{e'_{i,j}} \quad \text{with} \quad e_{i,j}, e'_{i,j} \in \mathbb{N}$$

Lattices

Euclidean lattice $\mathcal{L}$

A **Euclidean lattice** $\mathcal{L}$ is a *discrete additive subgroup* of $\mathbb{R}^m$. Intuitively, it is a regular ordering of points in $\mathbb{R}^m$.

- $\mathcal{L}(\{\mathbf{b_1}, \ldots, \mathbf{b_r}\}) = \{\sum_{i=1}^{r} x_i \cdot \mathbf{b_i} \mid x_i \in \mathbb{Z}\}$ is the set of all integer combinations of r linearly independent vectors $\mathbf{b_1} \ldots, \mathbf{b_r} \in \mathbb{R}^m$.
  - These $\mathbf{b}_i$ form a *basis* for $\mathcal{L}$.
  - It is common to package the $\mathbf{b_i}$ into a matrix $B \in \mathbb{R}^{m \times r}$.
  - We call m the *dimension* and r the *rank*.

Closest Vector Problem (CVP)

Given a lattice $\mathcal{L}(B)$ and a target vector $t \in \mathbb{R}^m$, find the vector $v \in \mathcal{L}$ that is closest to t. (Note that t is not necessarily in $\mathcal{L}$).

A “Sublinear-resource” Quantum Factoring Algorithm

Schnorr (2021)'s Lattice-based Factoring

- **Sieving is reduced to several CVPs** on the 'prime lattice', with the target vector t determined by $\ln N$;

$$B_{m,c} = \begin{pmatrix} f(1) & 0 & \cdots & 0 \\ 0 & f(2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & f(m) \\ N^c \ln p_1 & N^c \ln p_2 & \cdots & N^c \ln p_m \end{pmatrix}, \; t = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ N^c \ln N \end{pmatrix},$$

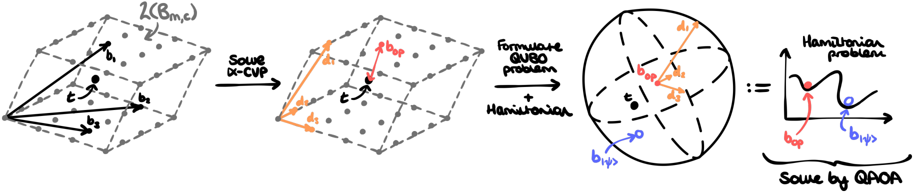
- The $f(i)$ elements are random permutations of the elements in $\{\lceil 1/2 \rceil, \ldots, \lceil m/2 \rceil\}$ to randomise the CVP.
  - By producing different CVPs, we generate different sr-pairs.

The sublinear lattice dimension claim

The claim is that $m \sim 2l \log N / \log\log N$, with l a hyperparameter.

Yan et al. (2022)'s QAOA-accelerated Approach

1. Find an approximate solution b_{op} using a polynomial-time algorithm.
2. Define the unit neighbourhood around b_{op} w.r.t. the LLL-reduced basis $D = \{d_1, \ldots, d_m\}$ for the prime lattice.
3. Formulate the search of this neighbourhood as a *minimising-energy eigenstate problem*; every eigenvector $|\psi\rangle$ with lower energy than $|0\rangle$ produces an enhanced solution $b_{|\psi\rangle} := b_{op} + \sum_{j=1}^{m} \kappa_j \psi_j d_j$.



The sublinear-resource factoring claim

We can thus perform a sieving subroutine via QAOA with "sublinear resources" (making it more bit-saving than Shor's algorithm!).

The Problem Hamiltonian

- We are looking for a vector v_{new} in the unit hypercube, centred on b_{op}, that is closest to the target t, giving the optimisation problem

$$F(z_1, \ldots, z_m) = \|t - v_{new}\|^2 = \left\| t - b_{op} + \sum_{i=1}^{m} \kappa_i z_i d_i \right\|^2 .$$

- Using the standard mapping $z_i \mapsto \hat{z}_i = (I - \sigma_z^i)/2$,

$$H_P = \left\| t - b_{op} + \sum_{i=1}^{m} \kappa_i \hat{z}_i d_i \right\|^2 = \sum_{j=1}^{m+1} \left| t_j - b_{op}^j + \sum_{i=1}^{m} \kappa_i \hat{z}_i d_{i,j} \right|^2 ,$$

where σ_z^i is the Pauli-Z operator $|0\rangle\langle 0| - |1\rangle\langle 1|$ on the i-th qubit.

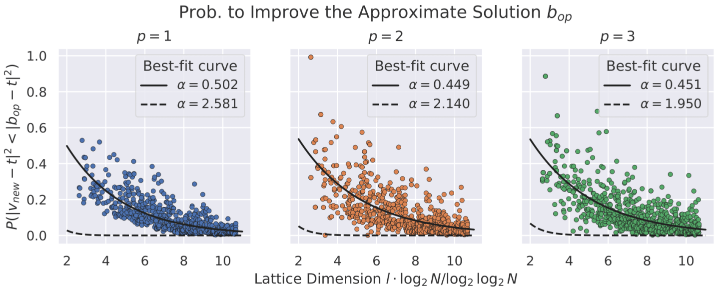
Time-complexity of a QAOA-accelerated CVP Solver

Exploring the Scaling of the CVP Solver

- It is hypothesised that better solutions to the CVP yield sr-pairs with higher likelihood. So, central to the claim is that b_{op} can be reliably improved at scale.

Failing to overcome the CVP's hardness

We show that *the probability to sample an improved solution decays exponentially in the lattice dimension*. However, we should not be pessimistic, because classical solvers are exponential too.

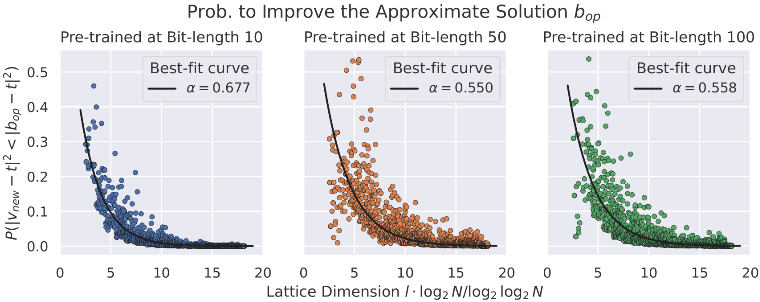


'Pre-trained' QAOA-acceleration

- A substantial cost for the QAOA-acceleration is in finding an optimal set of circuit parameters $\{\beta_i, \gamma_i\}_{i=1,\ldots,p}$ for a given Hamiltonian.

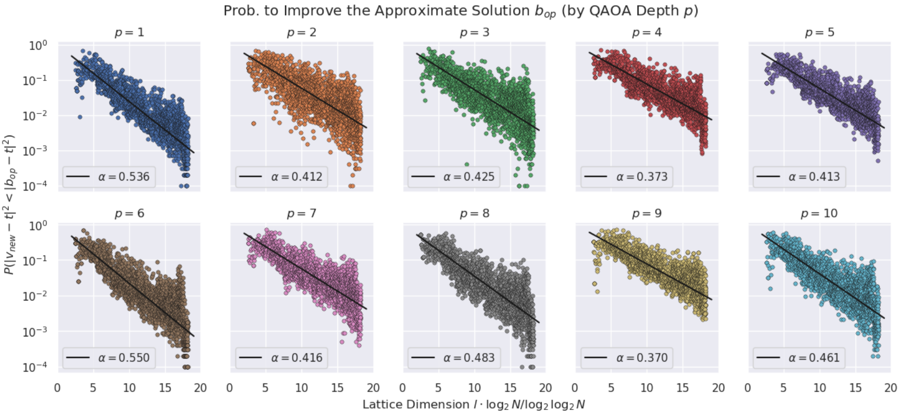
QAOA 'pre-training' (e.g. Boulebnane and Montanaro, 2022)

We show that *the circuit parameters can be 'pre-trained'* on a relatively small problem instance without loss to general performance.



'Pre-trained' QAOA-acceleration (Cont'd)

- In general, increasing the depth p improves the scaling.



What Does This Mean for RSA-2048?

- Obtaining that $P(|b_{|\psi\rangle} - t|^2 < |b_{op} - t|^2) \approx 1/(\sqrt{2})^m$ tells us that our **query complexity** is $(\sqrt{2})^m$ when using a sublinear lattice scheme.
  - We have good reason to believe that higher p would reduce this further, so the query complexity is also dependent on p.
- Yan et al. (2022) claim that 372 qubits suffices to break RSA-2048. This would require $\approx (\sqrt{2})^{372} \approx 9.81 \cdot 10^{55}$ queries *per CVP*.
  - Other work criticising Schnorr's original proposal indicate that exponentially many CVP instances would have to be considered under this claim (... because of the density of smooth numbers ...)

$\therefore$ If we wanted to solve a *single* CVP for RSA-2048 using this method, and our 3-deep QAOA circuit can run in 0.3 ps (*extremely optimistic...*), then it would take a **billion**, **billion**, **billion**, **billion** years.

- So factoring is safe, but this CVP solver may have applications outside of cryptography.