Practical Quantum Algorithms for Cryptanalysis

Ben Priestley

The University of Edinburgh

November 29, 2023

Problem Motivation

- Cryptographic protocols based on **prime factorisation** (or discrete logarithm) will be compromised by Shor's algorithm when a large fault-tolerant quantum computer is developed.
- New cryptosystems are being developed pre-emptively. The best so far are mostly based on **lattice problems**.
- Without a fault-tolerant quantum computer, it is hard to verify that these proposed cyptosystems will be quantum-safe.
- Today, we are seeing the emergence of **noisy intermediate-scale quantum (NISQ)**, which allow us to implement quantum algorithms that are sufficiently **shallow**.

A Practical Look at Quantum Security

We look at a new class of heuristic quantum algorithm – **variational algorithms (VQA)** – that may work well even on today's NISQ devices to analyse cryptosystems empirically.

- What are the practical implications for prime factorisation?
- What resources might a VQA need to break the RSA cryptosystem? Is this a more immediate threat than realising Shor's algorithm?
- Is lattice-based cryptography theoretically and practically secure against a VQA adversary? Can VQAs be used to test how quantum-safe proposed cryptosystems are?

Preliminaries

Lattices

- A **Euclidean lattice** $\mathcal{L}$ is a *discrete additive subgroup* of $\mathbb{R}^m$.
- Less formally, $\mathcal{L}(\{\mathbf{b_1}, \ldots, \mathbf{b_r}\}) = \{\sum_{i=1}^{r} x_i \cdot \mathbf{b_i} \mid x_i \in \mathbb{Z}\}$ is the set of all integer combinations of r linearly independent vectors $\mathbf{b_1} \ldots, \mathbf{b_r} \in \mathbb{R}^m$.
  - These $\mathbf{b}_i$ for 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*.
- Intuitively, a lattice $\mathcal{L}(B)$ is a regular ordering of points in $\mathbb{R}^m$.

Important Lattice Problems

- **Shortest Vector Problem (SVP)**: Given a lattice $\mathcal{L}(B)$, find the vector $v \neq 0$ whose length is the shortest of any vector in the lattice; that is $\|v\| = \lambda_1(\mathcal{L})$.
  - $\lambda_i(\mathcal{L})$ is shorthand meaning *the length of the i-th shortest vector in* $\mathcal{L}$.
- **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; that is $\|t - v\| = \text{dist}(\mathcal{L}, t)$.
- The hardness of these problems is dictated by how 'nice' the given basis is (how short and mutually orthogonal the $\mathbf{b}_i$ are).
  - An analogue to the idea of public/private key cryptography can be used with these problems, by having two equivalent bases – one nice and one not-so-nice.

Variational Quantum Algorithms (VQA)

- These are hybrid quantum-classical algorithms with three primary components:
  - A *cost function* $C(\vec{\theta})$ describing the problem to be solved in terms of a set of parameters $\vec{\theta}$.
  - An *ansatz* $U_A(\vec{\theta})$ expressing the search space in terms of $\vec{\theta}$.
  - A *(classical) optimiser* that iteratively improves the assignments to $\vec{\theta}$ to minimise $C(\vec{\theta})$.
- We first look at the **quantum approximate optimisation algorithm (QAOA)**. Later we will consider VQE, and then other novel algorithms like AQC-PQC.

An Outrageous Claim of Sub-linear Resources

The Sub-linear Claim

- A major motivation for this work is the claim due to Yan et al. (2022) that prime factorisation can be solved with "sub-linear resources" in the bit-length $\log_2 N$ of the semi-prime N.
  - They use this claim to estimate a requirement of 372 qubits (and a depth of thousands) to break RSA.
- Their claim is based on the claim due to Schnorr et al. (2021) that the prime factorisation problem can be reduced to CVP on the prime lattice with dimension $m \sim 2c \log N / \log \log N$, where c is a lattice parameter.
- Their claims are unlikely to hold at 'large' instances.

A Simulation Analysis of the SQIF

Overview of the SQIF

- Reduce the prime factorisation to the problem of finding $n+1$ smooth-relation pairs, then reduce this problem to CVP on the prime lattice.
- Find an approximate solution to the CVP by a classical algorithm (polynomial-time).
- Define a Hamiltonian to capture the energy (i.e. solution quality) of nearby lattice points.
- Use a VQA to sample low-energy eigenstates representing high-quality solutions.
- Unpack the solution in the sample as $n+1$ smooth relation pairs, and solve the resulting system of linear equations (polynomial-time), thus obtaining non-trivial factors.

Preprocessing Factoring as CVP

- Schnorr's algorithm reduces prime factoring to CVP by encoding pairs of p_n-smooth numbers (u, v) (assuming u and v are co-prime) as vectors on the prime lattice $B_{n,c}$.

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

- The basis $B_{n,c}$ is parameterised by the 'precision parameter' c, and is of dimension $n+1$ and rank n. The $f(i)$ elements are random permutations of elements in $\{\lceil 1/2 \rceil, \ldots, \lceil n/2 \rceil\}$.

Obtaining an (Approximate) Solution Classically

Schnorr and Yan et al. use **Babai's algorithm**, as follows:

- Perform a lattice reduction (e.g. by LLL-reduction) to yield a reduced basis $D = [\mathbf{d}_1 \cdots \mathbf{d}_n]$ and corresponding Gram-Schmidt orthogonal basis $\tilde{D} = [\tilde{\mathbf{d}}_1 \cdots \tilde{\mathbf{d}}_n]$.
- Perform a "size-reduction" of t using $D, \tilde{D}$ to yield n Gram-Schmidt coefficients $\mu_i = \langle \mathbf{d}_i, \tilde{\mathbf{d}}_i \rangle / \langle \tilde{\mathbf{d}}_i, \tilde{\mathbf{d}}_i \rangle$.
- Round the coefficients to the nearest integer to snap to a lattice point; $\mathbf{b}_{op} = \sum_{i=0}^{n} c_i \mathbf{d}_i$, where $c_i = \lceil \mu_i \rfloor$.

The Optimisation Problem: Searching Around $\mathbf{b}_{op}$

- We are looking for a new vector $\mathbf{v}_{new}$ in the unit hypercube centred on Babai's solution $\mathbf{b}_{op}$. This search space is given by either a positive or negative unit step in each reduced basis $\mathbf{d}_i$, or no step at all.
- This can be formulated as the optimisation problem $F : \{\pm 1, 0\}^n \to \mathbb{R}$ defined by

$$F(z_1, \ldots, z_n) = \|\mathbf{t} - \mathbf{v}_{new}\|^2 = \left\| \mathbf{t} - \sum_{i=1}^{n} (c_i + z_i)\mathbf{d}_i \right\|^2$$

- Notice that this is non-binary. This is a problem as it leads to a somewhat ad-hoc Hamiltonian definition.

The Optimisation Problem: QUBO Form

- c_i is obtained by rounding μ_i, so we are only really interested in the two values $\lfloor \mu_i \rfloor$ and $\lceil \mu_i \rceil$.
- However, F is defined to consider the c_i obtained by rounding, the alternative c_i we are interested in (had we rounded the other way), and *another* value that is one more than the rounding. We can remove this extra value from consideration by considering the original rounding in the formulation;

$$F(z_1, \ldots, z_n) = \|\mathbf{t} - \mathbf{v}_{new}\|^2 = \left\| \mathbf{t} - \sum_{i=1}^{n} (c_i + \lceil \mu_i - c_i \rceil z_i)\mathbf{d}_i \right\|^2$$

- Now F is a *binary* optimisation problem – it is a QUBO.

Mapping F to a Hamiltonian

Having formulated F as a QUBO, we may define a corresponding all-to-all connected Ising spin-glass Hamiltonian using the standard mapping $z_i \mapsto \hat{z}_i = (I - \sigma_z^i)/2$, which gives us

$$H_P = \left\| \mathbf{t} - \sum_{i=1}^{n} (c_i + \lceil \mu_i - c_i \rceil \hat{z}_i) \mathbf{d}_i \right\|^2 = \sum_{j=1}^{n+1} \left| t_j - b_{op}^j + \sum_{i=1}^{n} \lceil \mu_i - c_i \rceil \hat{z}_i d_{i,j} \right|^2$$

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

Sampling Low-energy Eigenstates by QAOA

- Open a uniform superposition over n qubits via $H^{\otimes n}|0^n\rangle$.
- Act with the unitaries $U(\gamma, H_P) = e^{-i\gamma H_P}$ parameterised by γ and $U(\beta, B) = e^{i\pi\beta B/2} = \prod_{i=1}^{n} e^{i\pi\beta X_i/2}$ parameterised by β.
  - Since this operation is not diagonal, there will be constructive and destructive interference that we hope will lead to states corresponding to low energy values for H_P.
- Repeat the application of the two U operations p times, with parameters γ_i and β_i in the i-th layer.
- The aim of the algorithm is to find the optimal values for the continuous variables γ and β so that the expectation value $\langle\gamma, \beta|H_P|\gamma, \beta\rangle$ is minimised.