Extending Regev's factoring algorithm to compute discrete logarithms

Martin Ekerå [1,2] and Joel Gärtner [1,2]

[1] KTH Royal Institute of Technology, SE-100 44 Stockholm, Sweden

[2] Swedish NCSA, Swedish Armed Forces, SE-107 85 Stockholm, Sweden

PQCrypto 2024, Oxford, England, United Kingdom, 14 June, 2024

SWEDISH ARMED FORCES

KTH VETENSKAP OCH KONST

Factoring and discrete logarithm problems

Integer Factoring Problem (IFP)

- Given an integer N, find non-trivial factors p, q such that $N = pq$.

Discrete Logarithm Problem (DLP)

- Given a generator g of a cyclic group and $x = g^e$, find e.

- Historically the basis for virtually all widely deployed asymmetric cryptography.
- Algorithms that solve the IFP can often be adapted to solve the DLP, and vice versa.

- In this presentation, we consider the DLP in cyclic subgroups of $\mathbb{Z}_N^*$.

Quantum algorithms for the IFP and DLP

Algorithm	Problem	#Multiplications	#Runs	Space usage
[Shor94]	IFP	$O(n)$	$O(1)$	$O(n)$
[Shor94]	DLP	$O(n)$	$O(1)$	$O(n)$

- The circuit size is given by the number of multiplications modulo n-bit integers N.

Quantum algorithms for the IFP and DLP

Algorithm	Problem	#Multiplications	#Runs	Space usage
[Shor94]	IFP	$O(n)$	$O(1)$	$O(n)$
[Shor94]	DLP	$O(n)$	$O(1)$	$O(n)$
[Regev23]	IFP	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(n^{3/2})$

- The circuit size is given by the number of multiplications modulo n-bit integers N.

Quantum algorithms for the IFP and DLP

Algorithm	Problem	#Multiplications	#Runs	Space usage
[Shor94]	IFP	$O(n)$	$O(1)$	$O(n)$
[Shor94]	DLP	$O(n)$	$O(1)$	$O(n)$
[Regev23]	IFP	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(n^{3/2})$
[Regev23] with [RV23]	IFP	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(n)$

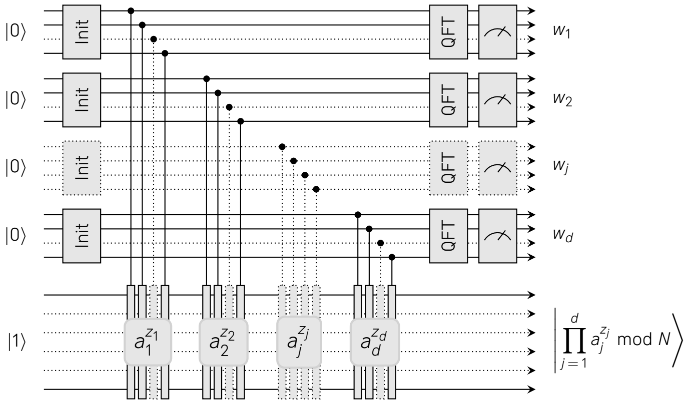
- The circuit size is given by the number of multiplications modulo n-bit integers N.

Quantum algorithms for the IFP and DLP

Algorithm	Problem	#Multiplications	#Runs	Space usage
[Shor94]	IFP	$O(n)$	$O(1)$	$O(n)$
[Shor94]	DLP	$O(n)$	$O(1)$	$O(n)$
[Regev23]	IFP	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(n^{3/2})$
[Regev23] with [RV23]	IFP	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(n)$
Our work	DLP	$O(\sqrt{n})$	$O(\sqrt{n})$	$O(n)$

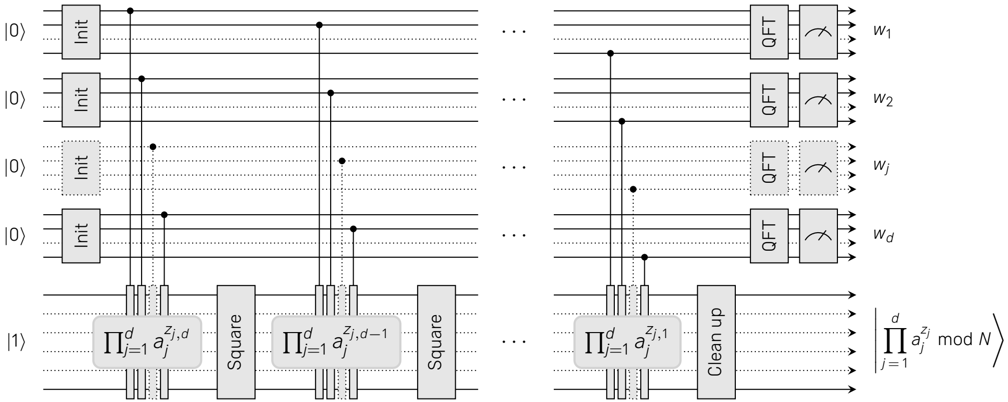
- The circuit size is given by the number of multiplications modulo n-bit integers N.

The quantum circuit



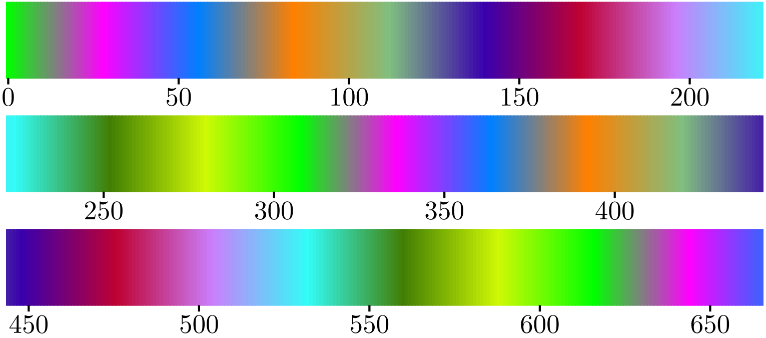
- The circuits for all of the aforementioned algorithms follow the same design pattern.

The quantum circuit



- By letting the a_j be small integers, and re-arranging the order of the multiplications, [Regev23] is able to reduce the circuit size at the expense of using more space.

Shor's factoring algorithm — one-dimensional period finding



Example: $f(z) = 73^z \bmod 667$

- Factors by finding the period of $f(z) = a^z \bmod N$ for random a.

Regev's factoring algorithm — d-dimensional period finding

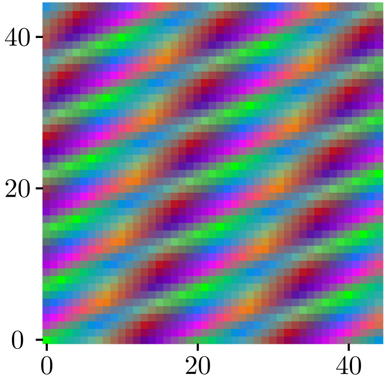
- Considers the function

$$f(z_1, \ldots, z_d) = \prod_{j=1}^{d} a_j^{z_j} \bmod N,$$

the period of which forms a lattice

$$\mathcal{L} = \{(z_1, \ldots, z_d) \mid f(z_1, \ldots, z_d) = 1\}.$$

- Under a heuristic assumption, it suffices to perform $\approx d$ runs to factor N.



Example: $f(x, y) = 4^x 9^y \bmod 667$

Contents

Our extension to computing discrete logarithms

The quantum algorithm

- Each runs of the quantum algorithm gives information on the periodicity of

$$f(z_1, \ldots, z_{d+2}) = x^{z_{d+1}} g^{z_{d+2}} \prod_{j=1}^{d} a_j^{z_j} \bmod N$$

where $x = g^e \bmod N$ and the a_j are small integers.

- Essentially the same algorithm as in [Regev23] but g and x need not be small.

Our extension to computing discrete logarithms

The classical post-processing

- Given the outputs from $O(d)$ runs, the post-processing recovers vectors in the lattice

$$\mathcal{L} = \left\{ (z_1, \ldots, z_{d+2}) \;\middle|\; x^{z_{d+1}} g^{z_{d+2}} \prod_{j=1}^{d} a_j^{z_j} \bmod N = 1 \right\}.$$

- Under a new heuristic assumption, the vectors recovered yield a basis for $\mathcal{L}$.
- Given a basis for $\mathcal{L}$, we can easily recover e by finding the vector

$$(0, \ldots, 0, 1, -e) \in \mathcal{L}.$$

Our new heuristic assumption

- Our new assumption is stronger than the assumption made in [Regev23].
- Both assumptions are essentially that small primes behave as random elements in $\mathbb{Z}_N^*$.
- [Pilatte24] recently proved a variant of our assumption with worse parameters.

Other extensions

More efficient factoring

- Under our new heuristic assumption, we can recover a basis for the lattice

$$\mathcal{L} = \left\{ (z_1, \ldots, z_d) \;\middle|\; \prod_{j=1}^{d} a_j^{z_j} \bmod N = 1 \right\}.$$

  Given a basis for $\mathcal{L}$ with the a_j small primes, we can efficiently factor N completely.
- In [Regev23], the a_j must be squares. In our algorithm, we can avoid the squaring.
- Thus, we can use a_j of half the bit length, which improves the efficiency.

Contents

On the need for robustness

- Quantum computers as currently envisaged may fail to correctly execute the circuit.
- [Regev23] requires $\Theta(\sqrt{n})$ good runs, so only a tiny failure probability is acceptable.

Two approaches to robustness

Our work

- The post-processing succeeds even if some runs are bad.

Ragavan and Vaikuntanathan

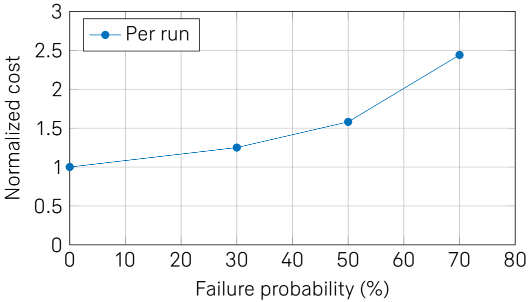
- [RV23] develops a method to filter out bad runs.

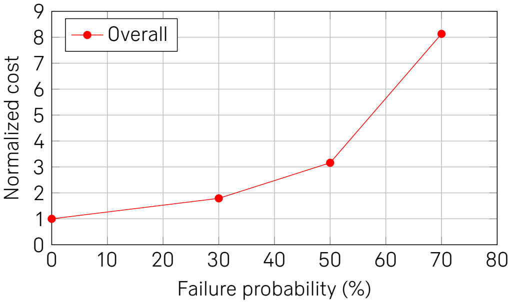
Further details on the two approaches

	Our work	[RV23]
Requirements	New heuristic assumption.	Special property for distribution of outputs from bad runs.
Efficiency	Somewhat larger parameters.	Significantly larger parameters.
Error tolerance	Arbitrary constant percentage.	Constant percentage.

- Natural that we achieve better efficiency since we rely on a heuristic analysis.

Quantifying the robustness through simulations





- [EG24sim] samples the distribution induced by the quantum algorithm.
- Motivates our new assumption and allows us to estimate parameter requirements.

- Simulator only efficient for classically tractable special-form problem instances.

Contents

Regev with space savings *vs.* existing variations of Shor

From our recent cost comparison [EG24] (arXiv:2405.14381)

Per-run advantage of existing variations of Shor						
		Problem size				
Algorithm	Problem	2048	3072	4096	6144	8192
[EH17, E20]	RSA IFP	3.16	2.46	2.04	1.58	1.33
[E19]	General DLP	1.71	1.31	1.08	0.83	0.69
[EH17, E20]	Short DLP	12.6	13.1	12.1	12.2	12.1
[E19]	Schnorr DLP	13.6	14.0	13.1	13.1	13.0

The advantage, defined as (cost of Regev) / (cost of Shor), in a cost model biased in favor of Regev.

▶ Performance for cryptographically relevant problem instances is of key interest.

Conclusion

Open questions

- Optimize Regev's algorithm to make it more competitive in practice.
- Provide optimizations for the special cases of short DLP and DLP in Schnorr groups.
- Extend the algorithm to the elliptic curve DLP.

Conclusion

Open questions

- Optimize Regev's algorithm to make it more competitive in practice.
- Provide optimizations for the special cases of short DLP and DLP in Schnorr groups.
- Extend the algorithm to the elliptic curve DLP.

Summary of our contribution

- We have extended Regev's factoring algorithm to compute discrete logarithms.
- We have provided slightly more efficient variants for factoring completely.
- We have analyzed and argued for the robustness of the post-processing.

KTH
VETENSKAP
OCH KONST