MPRI – Cours 2.12.2

F. Morain

Quantum factorization

2019/10/08

The slides are available on http://www.lix.polytechnique.fr/Labo/Francois.Morain/MPRI/2019

Contents

I. Brief quantum history

- Feynman (1982): simulation of quantum systems;
- Bennett & Brassard (1985): quantum key distribution;
- Shor (1994): factorization and discrete logarithm;
- Grover (1996): quantum search;
- Hallgren (2002): Pell's equation.

Basics: bit $\rightsquigarrow$ qubit.

Good reading:

- Science & Vie.
- *Comm. ACM*, 08/2015.

II. Shor's algorithm

Algorithm 1: Factoring with Shor's algorithm

input : N
output: (N_1, N_2) such that $N_1 N_2 = N$, $1 < N_i < N$

repeat
  $x \leftarrow \mathrm{Random}(2, N-2)$;
  $g \leftarrow \gcd(x, N)$;
  if $g \neq 1$ **then**
    return $(g, N/g)$;
  $r \leftarrow \mathrm{QOFA}(N, x)$;
until r *is even and* $x^{r/2} \not\equiv -1 \pmod N$;
$g \leftarrow \gcd(x^{r/2} - 1, N)$;
return $(g, N/g)$;

QOFA = Quantum Order Finding Algorithm (= oracle).

Shor's algorithm: example

$$N = 15$$

$$x = 2$$

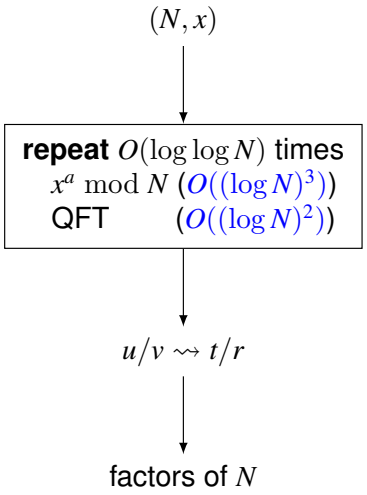
$$r = 4$$

$$2^{4/2} \equiv 4 \bmod N$$

$$\gcd(2^2 - 1, N) = \gcd(3, N) = 3$$

Rem. Works well for all possible x...! Not the case for ordinary random numbers (properties of orders $\bmod N$, etc.).

Shor's algorithm: QOFA(N, x)



QOFA in more details

Rough idea:

1. Find $N^2 \leq q = 2^\ell < 2N^2$; N and q are built in the quantum machine.
2. Initialize the first register (of ℓ qubits) to

$$\frac{1}{q^{1/2}} \sum_{a=0}^{q-1} |a\rangle|0\rangle.$$

3. Compute $x^a \bmod N$ in the 2nd register (of $\log N$ qubits) as

$$\frac{1}{q^{1/2}} \sum_{a=0}^{q-1} |a\rangle|x^a \bmod N\rangle.$$

4. Perform Fourier transform on the 1st register:

$$|a\rangle \mapsto \frac{1}{q^{1/2}} \sum_{c=0}^{q-1} \exp(2\pi iac/q)|c\rangle.$$

When QCM stops

The QC machine is left in state

$$\frac{1}{q} \sum_{a=0}^{q-1} \sum_{c=0}^{q-1} \exp(2\pi iac/q)|c\rangle|x^a \bmod N\rangle.$$

Observe the machine: some $|c\rangle|x^a \bmod N\rangle$.

Proba that it is in state $|c, x^k \bmod N\rangle$ for $0 \leq k < r$:

$$\begin{aligned} \varpi(c) = \varpi(|c\rangle) &= \left| \frac{1}{q} \sum_{a, x^a \equiv x^k \bmod N} \exp(2\pi iac/q) \right|^2 \\ &= \left| \frac{1}{q} \sum_{a \equiv k \bmod r} \exp(2\pi iac/q) \right|^2 \end{aligned}$$

Classical (1/2)

Write $a = br + k$:

$$\begin{aligned}\varpi(c) &= \left|\frac{1}{q}\sum_{b=0}^{\lfloor (q-k-1)/r\rfloor} \exp(2\pi i(br+k)c/q)\right|^2 \\ &= \left|\frac{1}{q}\sum_{b=0}^{\lfloor (q-k-1)/r\rfloor} \exp(2\pi ib(rc)/q)\right|^2\end{aligned}$$

Replace rc/q by $\{rc\}_q$ where $-q/2 < \{rc\}_q \le q/2$.

Prop. When $|\{rc\}_q| \le r/2$ (*)

$$\varpi(c) \ge 1/(3r^2)$$

for N large enough.

Classical (2/2)

Proof: (*) $\Longleftrightarrow$ there exists d s.t. $-\frac{r}{2} \le rc - dq \le \frac{r}{2}$ or

$$\left|\frac{c}{q} - \frac{d}{r}\right| \le \frac{1}{2q} < \frac{1}{2r^2}.$$

Since $q > N^2$, there is at most one d/r with $r < N$ for which this is true. □

Coro. (*) $\Longleftrightarrow$ there exists d s.t. d/r is a convergent of the continued fraction expansion of c/q.

Final analysis: there are $\varphi(r)$ possible values for d;
there are r possible values for $x^k \bmod N$.
$\Rightarrow r\varphi(r)$ states $|c, x^k \bmod N\rangle$ give r.
Proba $\ge r\varphi(r)/3r^2$.
Since $\varphi(r)/r > \delta/\log\log r$, we have to repeat the experiment $O(\log\log r)$ times. □

Factoring 15

$225 \le q = 2^\ell = 256 < 450$, $r = 4$, $1/(3r^2) = 0.02083333$

```
c     pi(c)    {rc}_q u/v
============================
 64 0.062500 0.00000 1/4
128 0.062500 0.00000 1/2
192 0.062500 0.00000 3/4
min=1.652108649839085629086591123 44E-57
```

Rem. $\lambda(15) = 4$; all elements $\ne 1$ have even order.

Factoring 55

$3025 \le q = 2^\ell = 4096 < 6050$; $r = 20$, $1/(3r^2) = 0.0008333$

```
# pi          {rc}_q  u/v
[...]
0.0021915 0.00097656 1/20
0.0021915 0.00097656 11/20
0.0021915 0.00097656 3/10
0.0021915 0.00097656 4/5
0.0021915 0.99902    1/5
0.0021915 0.99902    19/20
0.0021915 0.99902    7/10
0.0021915 0.99902    9/20
0.0024805 0.00000    1/2
0.0024805 0.00000    1/4
0.0024805 0.00000    3/4
0.0025049 0.00000    1/2
0.0025049 0.00000    1/4
0.0025049 0.00000    3/4
min=0.0006394
```

Towards a real implementation

Beauregard (2002): factoring a n-bit number by Shor's algorithm using

- $2n + 3$ (logical) qubits,
- $O(n^3 \log n)$ elementary quantum gates,
- depth $O(n^3)$.

...

Gidney & Ekerå (2019):

- $3n + 0.002n \log n$ qubits,
- $0.3n^3 + 0.0005n^3 \log n$ Toffoli,
- depth $500n^2 + n^2 \log n$ depth,

For $n = 2048$, $20 \cdot 10^6$ physical qubits.

Physical implementations of Shor's QOFA

Building the circuit $(N, x, a) \mapsto |a\rangle|x^a \bmod N\rangle$ for any (x, a) (let alone N) is too difficult $\Rightarrow$ compiled circuit for fixed x.

Some experimentations:

- Vandersypen *et al.*: nuclear magnetic resonance;
- Lu *et al.*: photonic qubits;
- Lanyon *et al.*: quantum entanglement;
- Politi *et al.*: photonic chip;
- Martin-Lopez *et al.*: using qubit recycling;
- Lucero *et al.*: Josephson phase qubit quantum processor.
- Kitaev, Griffiths/Niu, Parker/Plenio, Mosca/Ekert $\rightsquigarrow$ Monz *alii.* (2015). Scalable Shor?

More references

- Beckman *et al.* (arXiv:quant-ph/9602016): ion traps, 15.
- Cooper (arXiv:quant-ph/0612077): numerical example, numerical problems, etc.
- Ray Hill/Viamontes (arXiv:0804.3076): imprecision errors will kill you.
- Game/James (arXiv:1310.6446): 21.

...but all current *implementations* seem to require knowing the factors in advance!

Improving the post-processing

Joint work with F. Grosshans, T. Lawson, B. Smith,
`arxiv.org/abs/1511.04385`

(Ideal) QOFA: given N and x prime to N, return $r = \mathrm{ord}_N(x)$.

Pb. given $u/v = t/r$, can we recover r? We may have to test polynomially many multiples of v (Odlyzko) or rerun the experiment (which is **highly costly**). $\Rightarrow$ can we reduce the number of (physical) trials?

Pb. given r can we always split N in deterministic poly time?

The most favorable case: RSA keys with strong primes!

III. Concluding remarks

QC is a co-processor in its own right, in a precise computing model.
Probably not easy to program (loops? exact computations?).

Why factoring makes no progress:

- Hard to get funding for building a QC for factoring!!!
- Factoring is hard: scaling will probably teach you more things (can you seriously interpolate between 6 and 2048 bits!).
- Strange world where people (students) are diverted from real life (postquantum, etc.).

$\Rightarrow$ very few people in academia are really factoring interesting numbers.

Rem. DLP can also be solved in quantum polynomial time.