MPRI – Cours 2.12.2

F. Morain

Number theory and quantum factoring

2018/09/18

The slides are available on `http://www.lix.polytechnique.fr/Labo/Francois.Morain/MPRI/2015`

Contents

I. Elementary factoring

A) Removing small primes

Def. A number p is prime iff it has exactly two positive divisors: 1 and p. Composite numbers are not prime.

Ex. Primes: $2, 3, 5, 7, \ldots$. Composites: $4, 6, 8, 9, \ldots$.

Thm. (Hadamard, de la Vallée Poussin)
$\pi(x) = \#\{n \leq x, n \text{ prime }\} \sim x/\log x$.

Prop. any composite N has a prime factor $p \leq \sqrt{N}$.

Algo: make a table of primes (or not) and remove them from N. This takes $O(\sqrt{N}\,\mathsf{M}(\log N))$.

B) Fermat (1/2)

Prop. (Fermat) If N is composite, there exists X and Y s.t. $N = X^2 - Y^2 = (X - Y)(X + Y)$.

Proof: if $N = ab$, take $Y = (b - a)/2$, $X = (b + a)/2$. $\square$

Algorithm 1: Fermat algorithm

Function *Fermat(N)*
- **Input :** N an integer > 1
- **Output:** a pair of factors of N
- $Y \leftarrow -1$;
- **repeat**
  - $Y \leftarrow Y + 1$;
  - $X_2 \leftarrow N + Y^2$;
- **until** X_2 *is a perfect square*;
- $X \leftarrow \sqrt{X_2}$;
- **return** $(X - Y, X + Y)$.

Exo2-1. How to test if X_2 is a square rapidly?

Fermat (2/2)

Prop. If $N = ab$ with $a \leq b$, need $1 + (b-a)/2$ loops.
Proof: write $Y = (b-a)/2$, for which $X_2 = N + Y^2 = ((a+b)/2)^2$. $\square$

Generalization: if $N = ab$ with $a/b \approx \ell/m$, we use

$$4\ell m N = (ma + \ell b)^2 - (ma - \ell b)^2.$$

Prop. (Sherman-Lehman) find k s.t. $4kN = X^2 - Y^2$ with $k = \ell m$ and use preceding remark.

$\Rightarrow$ deterministic $O(N^{1/3})$.

Rem. Heuristic $O(N^{1/4+\epsilon})$ by McKee.

II. A glimpse of quantum factoring

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.

Shor's algorithm

Algorithm 2: Factoring with Shor's algorithm

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

```
repeat
    x ← Random(2, N − 2) ;
    g ← gcd(x, N) ;
    if g ≠ 1 then
        return (g, N/g) ;
    r ← QOFA(N, x) ;
until r is even and x^{r/2} ≢ −1 (mod N);
g ← gcd(x^{r/2} − 1, N) ;
return (g, N/g) ;
```

QOFA $=$ Quantum Order Finding Algorithm ($=$ oracle).
OFA has complexity $O(\sqrt{N})$ in classical complexity (see later).

A) Basic number theory

Thm. (Euler totient function)
$\varphi(N) := \mathrm{Card}((\mathbb{Z}/N\mathbb{Z})^*) = \prod_{i=1}^{k} \varphi(p_i^{\alpha_i}) = \prod_{i=1}^{k} p_i^{\alpha_i - 1}(p_i - 1)$ where $N = \prod_{i=1}^{k} p_i^{\alpha_i}$; $\quad \varphi(N)/N \geq \delta/\log\log N$.

Thm. (Carmichael function) $\lambda(N) := \mathrm{Exp}((\mathbb{Z}/N\mathbb{Z})^*) = \mathrm{lcm}_{i=1}^{k} \lambda(p_i^{\alpha_i})$ where

$$\lambda(p_i^{\alpha_i}) = \begin{cases} \varphi(p_i^{\alpha_i}) = p_i^{\alpha_i - 1}(p_i - 1) & \text{if } p_i \text{ odd or } \alpha_i \leq 2, \\ 2^{e-2} & \text{if } e \geq 3. \end{cases}$$

Properties:
1) $\forall x \in (\mathbb{Z}/N\mathbb{Z})^*$, $\mathrm{ord}_N(x) \mid \lambda(N) \mid \varphi(N)$ (Lagrange);
2) $\lambda(N) = \max\{\mathrm{ord}_N(x),\, x \in (\mathbb{Z}/N\mathbb{Z})^*\}$.

Ex. $\lambda(15) = \mathrm{lcm}(2, 4) = 4$ and $\varphi(15) = 8$: all elements of $(\mathbb{Z}/15\mathbb{Z})^*$ except 1 have even order.

Justification of the algorithm

Prop.
(i) $X^r \equiv 1 \bmod N$ iff $X^r \equiv 1 \bmod p_i^{e_i}$ for all i (use CRT).
(ii) For odd p, $X^r \equiv 1 \bmod p^e$ has $\gcd(r, \varphi(p^e))$ solutions.

Typical use: $X^2 \equiv 1 \bmod (pq)$ has four solutions: ± 1 and $\pm x_0$, where $\gcd(x_0 - 1, N) \in \{p, q\}$.

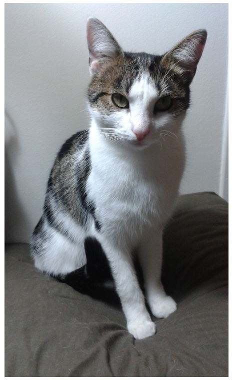
Thm: r is enough to factor N in random polynomial time using a bounded number of calls to QFOA.

Lem. If $\ell \mid r$, then $\gcd(x^{r/\ell} - 1, N)$ might reveal a factor of N.

Proof: we cannot have $x^{r/\ell} \equiv 1 \bmod p$ for all $p \mid N$. $\square$

But we can have $x^{r/2} \equiv -1 \bmod p$ for all p. This happens for random x with proba $1/2^{k-1}$ (for $N \neq p^e$).

B) Shor's algorithm: QFOA(N, x)



(N, x)

↓

repeat $O(\log\log N)$ times
$x^a \bmod N$ ($O((\log N)^3)$)
QFT ($O((\log N)^2)$)

↓

$u/v \rightsquigarrow t/r$

↓

factors of N

QFOA 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 second 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 first register, mapping $|a\rangle$ to

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

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 $a = br + k$:

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

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

Prop. When $|\{rc\}_q| \leq r/2$

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

for N large enough.

Classical (2/2)

Proof: This happens when there exists d s.t. $-\frac{r}{2} \leq rc - dq \leq \frac{r}{2}$ or

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

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

Coro. 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 $\geq r\varphi(r)/3r^2$.
Since $\varphi(r)/r > \delta/\log\log r$, we have to repeat the experiment $O(\log\log r)$ times. $\square$

Factoring 15

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

```
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=1.6521086498390856290865911234E-57
```

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

Factoring 55

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

```
# pi           {r*c}_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
```

Physical implementations of Shor's QFOA

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?

Caveat: quantum computers do not exist (yet?). Largest numbers "factored": 15 and 21; 51, 85.

III. Algorithms for generic groups

Why groups? finite groups are used everywhere in crypto (and elsewhere).

Which tasks?

- representing elements;
- drawing elements at random;
- efficient group laws;
- computation of cardinality;
- structure (with generators);
- etc.

Some groups

- $(\mathbb{Z}/N\mathbb{Z})^*$;
- finite fields $\mathbb{F}_{q^n}$ and subfields;
- algebraic curves (elliptic, hyperelliptic, any genus) over finite fields;
- class groups;
- etc.

Theoretical results

$(G, \circ, 1_G)$, Abelian, finite, of order N; computable $\circ$.

Def. $\mathrm{ord}_G(a) = \min\{k > 0, a^k = 1_G\}$.

Thm. (Lagrange) $\mathrm{ord}_G(a) \mid N$.

Coro. $a^{-1} = a^{N-1}$.

Def. $\mathrm{Exp}(G) = \min\{k > 0, \forall\, a \in G, a^k = 1_G\}$.

Prop.
1. $\mathrm{Exp}(G) \mid N$;
2. $\mathrm{Exp}(G) = \mathrm{lcm}(\mathrm{ord}_G(a), a \in G)$.

It can happen that $\mathrm{Exp}(G) < N$, see later.

Finding the order of an element

Pb. $G = \langle g \rangle$, $N = \mathrm{ord}(g)$; what is the order ω of a in G?

Thm. (Lagrange) $\omega \mid N$.

Rem. If N is small, we can enumerate in $O(N)$ or its divisors.

Prop. a is of order ω if and only if
i) $a^\omega = 1_G$;
ii) for all prime $p \mid \omega$, $a^{\omega/p} \neq 1_G$.

Proof:

In practice, if N *and* its factorization are known, easy.

What if we don't know N (completely)? E.g., (hyper)elliptic curves.

Baby-steps giant-steps

Fundamental algorithm in ANT/crypto; due to Shanks.

Write:

$$\omega = cu + d, \quad 0 \leq d < u, \quad 0 \leq c < N/u.$$

$$a^\omega = 1_G \Leftrightarrow (a^{-u})^c = a^d.$$

Number of group operations: $C_o = u + N/u$ minimized for $u = \sqrt{N}$, hence $2\sqrt{N}$ group operations.

Set operations: u insertions in $\mathcal{B}$ and N/u membership tests in the worst case.

$\Rightarrow$ $\mathcal{B}$ must be a hash table, where both operations take $O(1)$.

Complexity: $O(\sqrt{N})$ in time and space.

Algorithm 3: Baby steps giant steps

Function *BSGS*(G, g, N, a)
- **Input** : $G \supset \langle g \rangle$, g of order N
- **Output:** $\omega = \mathrm{ord}_g(a)$
- $u \leftarrow \lceil \sqrt{N} \rceil$;
- `// Step 1 (baby steps)`
- initialize a table $\mathcal{B}$ for storing u pairs (elt of G, int $< N$);
- store($\mathcal{B}$, (1_G, 0));
- $H \leftarrow a$; store($\mathcal{B}$, (H, 1));
- **for** $d := 2$ **to** $u - 1$ **do**
  - $H \leftarrow H \circ a$; store($\mathcal{B}$, (H, d));
- `// Step 2 (giant steps)`
- $H \leftarrow H \circ a; f \leftarrow 1/H = a^{-u}$;
- $H \leftarrow 1_G$;
- **for** $c := 0$ **to** N/u **do**
  - // $H = f^c$
  - **if** $\exists (H', d) \in \mathcal{B}$ *such that* $H = H'$ **then**
    - // $H = f^c = a^d$ hence $\omega = cu + d$
    - **return** $cu + d$;
  - $H \leftarrow H \circ f$;

Exercises

Exo1-0. Fill in the missing details in function BSGS (numerical trials and/or think).

Exo1-1. Decrease the average time by remarking that $c \approx N/(2u)$ on average.

Exo1-2. What if computing $1/x$ is free?

Exo1-3. Design a variant which takes $O(\max(c, d))$ operations. What is its average running time?