6.2 The Integer Factorization Problem

"Easy": Decide whether a given integer is composite.

"Hard": Find its prime factorization.

Some basics about factoring algorithms:

Pollard's p-1 factoring algorithm

Given composite n. Assume that n has a prime factor p such that $p-1$ has all prime factors $\leq B$.

Algorithm Pollard-(p-1)

Choose $a > 1$ (often $a = 2$)
Compute $b = a^{B!} \bmod n$
Compute $d = \gcd(b-1, n)$
If $1 < d < n$, then d is a nontrivial factor of n

Proof that this algorithm is correct:

Assume p is a prime factor of n s.t. $(p-1)$ has all pr. fctor $\leq B$.
Then $p-1 \mid B!$, i.e., $B! = k(p-1)$, $k \in \mathbb{N}$
By Fermat's little theorem

$$a^{B!} \equiv (a^{p-1})^k \equiv 1 \bmod p$$

Hence, $a^{B!} - 1 \equiv 0 \bmod p$ such that

$\gcd(a^{B!} - 1, n)$ is a factor of n. $\blacksquare$

Remarks

a) Compute $a^{B!} \bmod n$ as follows

$$b_1 = a \bmod n \ , \quad b_j = b_{j-1}^{\,j} \bmod n \ , \ j = 2, \dots, B .$$

b) $B!$ may be substituted by

$$\prod_{\substack{q \le B \\ q \text{ prime}}} q^{\lfloor \ln n / \ln q \rfloor}$$

Note that $q^{\lfloor \ln n / \ln q \rfloor} \le n$

since: $\ell \le \frac{\ln n}{\ln q} \iff \ell \ln q \le \ln n \iff q^{\ell} \le n$

note that $p-1 \,/\, \prod_{q \le B} q^{\lfloor \ln n / \ln q \rfloor}$ still holds.

To protect against Pollard-$(p-1)$ select $n = p \cdot q$ s.t. $p-1$ and $q-1$ have at least one large prime factor. How? $\rightarrow$ Exercise

Improvement of Pollard-$(p-1)$ is "elliptic curve factoring".

Another principle is as follows

Example: Factor 8051.

$$8051 = 8100 - 49 = 90^2 - 7^2$$
$$= (90+7)(90-7) = 97 \cdot 83 \quad .\lrcorner$$

Proposition 6.8. $x \not\equiv \pm y \pmod n$, $x^2 \equiv y^2 \pmod n$
$\Rightarrow \gcd(x-y, n)$ is a nontrivial divisor of n. $\lrcorner$

Proof. $x^2 \equiv y^2 \pmod n$, i.e., $n \mid x^2 - y^2$.
Hence $n \mid (x+y)(x-y)$. By assumption
$n \nmid (x-y)$ and $n \nmid (x+y)$, which shows the assertion. $\lrcorner$

- Prop. 6.8 forms the basis of "quadratic sieve factoring".
  Problem: determine x, y satisfying the assumptions.
  [see Stinson (2002), p. 182–194]
- $\gcd(a,b)$ can be efficiently computed by
  Euclid's algorithm, see sect. 6.3.

Thinking the Future
Zukunft denken

Factoring in practice (→ wikipedia)

Three factoring methods are most successful

- quadratic sieve
- elliptic curve factoring
- ~~nu~~ number field sieve (most powerful)

All are "subexponential", however, non polynomial.

History of factoring:

1994: RSA-129 Atkins, Graff, Lenstra, Leyland: quadr. sieve, 600 workstations.

1996: RSA-130

1999: RSA-155 8400 MIPS-years, 300 PCs

2003: RSA-174

2005: RSA-193 5 months, 80 2.2 GHz Opteron CPU (by BSI)

2010: RSA-232 many hundreds of 2.2 GHz Opteron CPU almost 2 years.

Factoring is considered a one-way function.

- "Easy": Given 2 primes p, q. Compute $n = p \cdot q$
- "Computationally infeasible", "hopeless"
  Given $n = p \cdot q$, p, q prime, unknown
  Determine p and q.

6.3. The Extended Euclidean Algorithm

Known : $\gcd(a,n)=1 \Rightarrow \exists$ inverse s with $a\cdot s \equiv 1 \pmod{n}$

Aim: efficient algorithm to compute $s = a^{-1} \pmod{n}$.

Euclidian algorithm: Let $r_0 > r_1 \in \mathbb{N}$

$(*)$

$r_0 = q_1 r_1 + r_2, \quad 0 < r_2 < r_1$	$r_2 = r_0 - q_1 r_1 = s_2 r_0 + t_2 r_1$
$r_1 = q_2 r_2 + r_3, \quad 0 < r_3 < r_2$	$r_3 = r_1 - q_2 r_2 = s_3 r_0 + t_3 r_1$
$\vdots$	$\vdots$
$r_{k-2} = q_{k-1} r_{k-1} + r_k, \quad 0 < r_k < r_{k-1}$	$r_k = s_k r_0 + t_k r_1$
$r_{k-1} = q_k r_k$	

It holds that $\underline{\gcd(r_0, r_1)} = \gcd(r_1, r_2) = \dots = \gcd(r_{k-1}, r_k) = \underline{r_k}$,

(since $\gcd(a,b) = \gcd(b, a-qb)$)

If $\gcd(r_0, r_1) = 1$ then $s_k r_0 + t_k r_1 = 1$

$\Rightarrow t_k r_1 \equiv 1 - s_k r_0 \equiv 1 \pmod{r_0}$, i.e., $t_k \equiv r_1^{-1} \pmod{r_0}$

Define recursively (according to r.h.s. of $(*)$)

$t_0 = 0, \quad t_1 = 1, \quad t_j = (t_{j-2} - q_{j-1} t_{j-1}) \bmod r_0, \quad j \geq 2$

Theorem 6.8. $r_j \equiv t_j \cdot r_1 \pmod{r_0}$, $j = 0, \dots, k$

Proof. (by induction on j)

$j=0$: $r_0 \equiv t_0 r_1 \pmod{r_0}$ ✓

$j=1$: $r_1 \equiv t_1 r_1 \pmod{r_0}$ ✓

$(j-2, j-1) \Rightarrow j$:

$$r_j \underset{\text{Eucl. alg.}}{\equiv} r_{j-2} - q_{j-1} r_{j-1} \underset{\text{I.H.}}{\equiv} t_{j-2} r_1 - q_{j-1} t_{j-1} r_1$$

$$\equiv (t_{j-2} - q_{j-1} t_{j-1}) r_1 \equiv t_j r_1 \pmod{r_0} \quad \blacksquare$$

Corollary 6.9. $\gcd(r_0, r_1) = 1 \Rightarrow t_k \equiv r_1^{-1} \pmod{r_0}$ ⌟

Number of divisions in the Euclidean algorithm

$$\leq \log_\phi(\sqrt{5}\, r_0) - 2 \quad , \phi = \underbrace{\tfrac{1}{2}(1+\sqrt{5})}_{\text{golden ratio}}$$

(Knuth, II, Chap. 4.5.3., p. 320)

Least favorable if r_0, r_1 are successive Fibonacci numbers.

$$\begin{array}{cccccccc} 1 & 1 & 2 & 3 & 5 & 8 & 13 & 21 \; \dots \\ & & & & & & r_1 & r_0 \end{array}$$

Efficient implementation, see Stinson p. 161. ⌟

6.4. The Chinese Remainder Theorem

Theorem 6.10.

Suppose $m_1, \dots, m_r$ are pairwise relatively prime, $a_1, \dots, a_r \in \mathbb{N}$. The system of r congruences

$$x \equiv a_i \pmod{m_i}, \quad i = 1, \dots, r$$

has a unique solution modulo $M = m_1 \cdots m_r$, given by

$$x = \sum_{i=1}^{r} a_i M_i y_i \mod M,$$

where $M_i = M / m_i$, $y_i = M_i^{-1} \mod m_i$, $i = 1, \dots, r$.

Proof. Stinson (02), p. 162, 163.

Example $r = 3$, $m_1 = 7$, $m_2 = 11$, $m_3 = 13$

$\Rightarrow M = 1001$, $M_1 = 143$, $M_2 = 91$, $M_3 = 77$

$y_1 = 143^{-1} \mod 7 = 3^{-1} \mod 7 = 5$

$y_2 = 4$, $y_3 = 12$

Solution of

$$\begin{aligned} x &\equiv 5 \pmod 7 & (a_1 = 5) \\ x &\equiv 3 \pmod{11} & (a_2 = 3) \\ x &\equiv 10 \pmod{13} & (a_3 = 10) \end{aligned}$$

is $x = 5 \cdot 143 \cdot 5 + 3 \cdot 91 \cdot 4 + 10 \cdot 77 \cdot 12 \mod 1001$

$= 894$