Factoring $N = p^r q^s$ in Polynomial Time for Large $r, s$

Jean-Sébastien Coron

University of Luxembourg

April 14, 2015
Coppersmith’s technique for finding small roots of polynomial equations [Cop97]
- Based on the LLL lattice reduction algorithm
- Numerous applications in cryptography.

Application: factoring with high bits known
- Factor $N = pq$ in polynomial time if 1/2 of the bits of $p$ are known. [Cop97]

Polynomial time factorization of $N = p^r q$ for large $r$ [BDHG99]
- Factor $N = p^r q$ in polynomial time if $1/(r + 1)$ of the bits of $p$ are known
  - Therefore polynomial time for $r \sim \log p$.

Polynomial time factorization of $N = p^r q^s$ for large $r$ or $s$
  (this talk).
Solving $f(x) = 0 \mod N$ when $N = pq$ is of unknown factorization: hard problem.
- For $f(x) = x^2 - a$, equivalent to factoring $N$.
- For $f(x) = x^e - a$, equivalent to inverting RSA.

Coppersmith showed (E96) that finding small roots is easy.
- When $\deg f = \delta$, finds in polynomial time all integer $x_0$ such that $f(x_0) = 0 \mod N$ and $|x_0| \leq N^{1/\delta}$.
- Based the LLL lattice reduction algorithm.

Can be heuristically extended to more variables.
Coppersmith’s algorithm has numerous applications in cryptanalysis:

- Cryptanalysis of plain RSA when some part of the message is known:
  - If $c = (B + x_0)^3 \mod N$, let $f(x) = (B + x)^3 - c$ and recover $x_0$ if $x_0 < N^{1/3}$.
- Breaking RSA for $d < N^{0.29}$

Applications in provable security:

- Improved security proof for RSA-OAEP with low-exponent $e$ (Shoup, C01).
Coppersmith’s Technique

- We want to find a small root $x_0$ of $f(x) \equiv 0 \pmod{N}$.
- Find a small linear integer combination $h(x)$ of the polynomials:
  
  $$q_{ik}(x) = x^i \cdot N^{\ell-k} \cdot f^k(x) \pmod{N^\ell}$$

- for some $\ell$ and $0 \leq i < \delta$ and $0 \leq k \leq \ell$.
- $f(x_0) = 0 \pmod{N} \Rightarrow f^k(x_0) = 0 \pmod{N^k} \Rightarrow q_{ik}(x_0) = 0 \pmod{N^\ell}$.
- Then $h(x_0) = 0 \pmod{N^\ell}$.
- If the coefficients of $h(x)$ are small enough:
  - Then $h(x_0) = 0$ holds over $\mathbb{Z}$.
  - $x_0$ can be found using any standard root-finding algorithm.
Illustration with a polynomial of degree 2:
- Let \( f(x) = x^2 + ax + b \mod N \).
- We must find \( x_0 \) such that \( f(x_0) = 0 \mod N \) and \( |x_0| \leq X \).

We are interested in finding a small linear integer combination of the polynomials:
- \( f(x) \), \( N x \) and \( N \).
- Then \( h(x_0) = 0 \mod N \).

If the coefficients of \( h(x) \) are small enough:
- Then \( h(x_0) = 0 \) also holds over \( \mathbb{Z} \),
- which enables to recover \( x_0 \).
Howgrave-Graham lemma

Given \( h(x) = \sum h_i x^i \), let \( \| h \|^2 = \sum h_i^2 \).

**Howgrave-Graham lemma:**

- Let \( h \in \mathbb{Z}[x] \) be a sum of at most \( \omega \) monomials. If \( h(x_0) = 0 \mod N \) with \( |x_0| \leq X \) and \( \| h(xX) \| < N/\sqrt{\omega} \), then \( h(x_0) = 0 \) holds over \( \mathbb{Z} \).

**Proof:**

\[
|h(x_0)| = \left| \sum h_i x_0^i \right| = \left| \sum h_i X^i \left( \frac{x_0}{X} \right)^i \right|
\leq \sum \left| h_i X^i \left( \frac{x_0}{X} \right)^i \right| \leq \sum \left| h_i X^i \right|
\leq \sqrt{\omega} \| h(xX) \| < N
\]

Since \( h(x_0) = 0 \mod N \), this gives \( h(x_0) = 0 \).
Factoring $N = p^r q^s$ in Polynomial Time for Large $r, s$
The coefficients of $h(xX)$ must be small:

- $h(xX)$ is a linear integer combination of the polynomials

\[
\begin{align*}
  f(xX) &= X^2 \cdot x^2 + aX \cdot x + b \\
  q_1(xX) &= NX \cdot x \\
  q_2(xX) &= N
\end{align*}
\]

We must find a small integer linear combination of the vectors:

- $[X^2, aX, b]$, $[0, NX, 0]$ and $[0, 0, N]$ 

Tool: LLL algorithm.
We must find a small linear integer combination $h(xX)$ of the polynomials $f(xX)$, $xXN$ and $N$.

Let $L$ be the corresponding lattice, with a basis of row vectors:

$$\begin{bmatrix}
X^2 & aX & b \\
xN & b \\
N & N
\end{bmatrix}$$

Using LLL, one can find a lattice vector $b$ of norm:

$$\|b\| \leq 2(\det L)^{1/3} \leq 2N^{2/3}X$$

Then if $X < N^{1/3}/4$, then $\|h(xX)\| = \|b\| < N/2$

Howgrave-Graham lemma applies and $h(x_0) = 0$. 

Jean-Sébastien Coron

Factoring $N = p^r q^s$ in Polynomial Time for Large $r$, $s$
Lattice

Definition:
Let $u_1, \ldots, u_\omega \in \mathbb{Z}^n$ be linearly independent vectors with $\omega \leq n$. The lattice $L$ spanned by the $u_i$'s is

$$L = \left\{ \sum_{i=1}^{\omega} n_i \cdot u_i | n_i \in \mathbb{Z} \right\}$$

If $L$ is full rank ($\omega = n$), then $\det L = |\det M|$, where $M$ is the matrix whose rows are the basis vectors $u_1, \ldots, u_\omega$.

The LLL algorithm:
The LLL algorithm, given $(u_1, \ldots, u_\omega)$, finds in polynomial time a vector $b_1$ such that:

$$\|b_1\| \leq 2^{(\omega-1)/4} \det(L)^{1/\omega}$$
The previous bound gives $|x_0| \leq N^{1/3}/4$.

But Coppersmith’s bound gives $|x_0| \leq N^{1/2}$.

Technique: work modulo $N^k$ instead of $N$.

Let $q(x) = (f(x))^2$. Then $q(x_0) = 0 \mod N^2$.

$q(x) = x^4 + a'x^3 + b'x^2 + c'x + d'$.

Find a small linear combination $h(x)$ of the polynomials $q(x), Nx f(x), N f(x), N^2 x$ and $N^2$.

Then $h(x_0) = 0 \mod N^2$.

If the coefficients of $h(x)$ are small enough, then $h(x_0) = 0$. 
Details when working modulo $N^2$

- **Lattice basis:**
  
  \[
  \begin{bmatrix}
  X^4 & a' X^3 & b' X^2 & c' X & d' \\
  NX^3 & NaX^2 & NbX & NaX & Nb \\
  NX^2 & NaX & N^2 X & N^2 & N^2
  \end{bmatrix}
  \]

- Using LLL, one gets:
  - $\|h(xX)\| \leq 2 \cdot (\det L)^{1/5} \leq 2X^2 N^{6/5}$
  - If $X \leq N^{2/5}/6$, then $\|h(xX)\| \leq N^2/3$ and $h(x_0) = 0$.

- We get $X \sim N^{2/5}$ instead of $X \sim N^{1/3}$
  - By further increasing the lattice dimension, we can get Coppersmith’s bound $X \sim N^{1/2}$.
Let \( N = p \cdot q \). Assume that we know half of the most significant bits of \( p \).

Write \( p = P + x_0 \) for some known \( P \) and unknown \( x_0 \) with \( x_0 < p^{1/2} \).

Consider the system:

\[
\begin{align*}
N &\equiv 0 \pmod{P + x_0} \\
x + P &\equiv 0 \pmod{P + x_0}
\end{align*}
\]

\( x_0 \) is a small root of both polynomial equations.

We can apply Coppersmith’s technique: the only difference is that the modulus is unknown, but this is not a problem for Howgrave-Graham’s Lemma.

We can recover \( x_0 \) if \( x_0 < p^{1/2} \)

Polynomial time factorization of \( N = pq \) if half of the high order (or low order) bits of \( p \) are known.
Factoring $N = p^r q$ in Polynomial Time

- Extension to $N = p^r q$ from [BDHG99]
  - Polynomial-time factorization of $N = p^r q$ when $1/(r + 1)$ of the bits of $p$ are known.

- Polynomial-time factorization of $N = p^r q$ for large $r$
  - When $r \approx \log p$, only a constant number of bits of $p$ need to be known.
  - Exhaustive search of these bits is then polynomial-time
Factoring $N = p^r q^s$ in Polynomial Time

- Polynomial time factorization of $N = p^r q^s$ when $r$ or $s$ is greater than $(\log p)^3$
- Particular case: $N = p^{r+1} q^r$.
  - We can write $N = (pq)^r \cdot p$ and apply [BDHG99] to factor in polytime, again when $r \simeq \log p$
- More generally: $N = p^{\alpha \cdot r + a} \cdot q^{\beta \cdot r + b}$
  - Write $N = (p^\alpha q^\beta)^r \cdot (p^a q^b)$
  - Still factor in polytime if $r \simeq \log p$, for small $\alpha, \beta, a, b$.
- More generally for $N = p^r q^s$.
  - Write: \[
  \begin{cases}
  r = u \cdot \alpha + a \\
  s = u \cdot \beta + b
  \end{cases}
  \]
  for some small enough $\alpha, \beta, a, b$, and large enough $u$.
  - $N = P^u Q$ where $P := p^\alpha q^\beta$ and $Q := p^a q^b$
  - Apply [BDHG99] to factor $N = P^u Q$ in polytime.