The Quadratic Sieve

Přemysl Jedlička

Department of Mathematics
Technical Faculty
Czech University of Agriculture, Prague

8 May 2006, Znojmo

Large Numbers Factorisation

The prime number factorisation is the decomposition of a number N to $N = \prod p_i^{b_i}$, for p_i primes.
This problem is believed to be NP.

In RSA algorithm, the central role play two primes p and q and $N = p \cdot q$.
N is a part of the public key.
Primes p, q are the secret key.

Large Numbers Factorisation

The prime number factorisation is the decomposition of a number N to $N = \prod p_i^{b_i}$, for p_i primes.
This problem is believed to be NP.

In RSA algorithm, the central role play two primes p and q and $N = p \cdot q$.
N is a part of the public key.
Primes p, q are the secret key.

Large Numbers Factorisation

The prime number factorisation is the decomposition of a number N to $N = \prod p_i^{b_i}$, for p_i primes.
This problem is believed to be NP.

In RSA algorithm, the central role play two primes p and q and $N = p \cdot q$.
N is a part of the public key.
Primes p, q are the secret key.

Large Numbers Factorisation

The prime number factorisation is the decomposition of a number N to $N = \prod p_i^{b_i}$, for p_i primes.
This problem is believed to be NP.

In RSA algorithm, the central role play two primes p and q and $N = p \cdot q$.
N is a part of the public key.
Primes p, q are the secret key.

Large Numbers Factorisation

The prime number factorisation is the decomposition of a number N to $N = \prod p_i^{b_i}$, for p_i primes.
This problem is believed to be NP.

In RSA algorithm, the central role play two primes p and q and $N = p \cdot q$.
N is a part of the public key.
Primes p, q are the secret key.

Algorithms Used For Factorisation

The best classical algorithms for factorising a product of two primes are based on the Fermat's factorisation scheme. Examples are:

- CFRAC (Continued Fraction Algorithm)
- Quadratic Sieve
- Number Field Sieve

The best (in theory) is quantum factorisation.

Algorithms Used For Factorisation

The best classical algorithms for factorising a product of two primes are based on the Fermat's factorisation scheme. Examples are:

- CFRAC (Continued Fraction Algorithm)
- Quadratic Sieve
- Number Field Sieve

The best (in theory) is quantum factorisation.

Algorithms Used For Factorisation

The best classical algorithms for factorising a product of two primes are based on the Fermat's factorisation scheme. Examples are:

- CFRAC (Continued Fraction Algorithm)
- Quadratic Sieve
- Number Field Sieve

The best (in theory) is quantum factorisation.

Algorithms Used For Factorisation

The best classical algorithms for factorising a product of two primes are based on the Fermat's factorisation scheme. Examples are:

- CFRAC (Continued Fraction Algorithm)
- Quadratic Sieve
- Number Field Sieve

The best (in theory) is quantum factorisation.

Algorithms Used For Factorisation

The best classical algorithms for factorising a product of two primes are based on the Fermat's factorisation scheme. Examples are:

- CFRAC (Continued Fraction Algorithm)
- Quadratic Sieve
- Number Field Sieve

The best (in theory) is quantum factorisation.

Fermat's Factorisation

We want to factorise a large number N. Suppose that we have a set of equations

$$x_i \equiv y_i^2 \pmod{N}.$$

Let us pick up a set $\mathscr{I}$ of indeces such that

$$\prod_{i \in \mathscr{I}} x_i = x^2.$$

Then we have

$$x^2 = \prod_{i \in \mathscr{I}} x_i \equiv \prod_{i \in \mathscr{I}} y_i^2 = y^2 \pmod{N};$$

$$(x - y)(x + y) = kN.$$

Fermat's Factorisation

We want to factorise a large number N. Suppose that we have a set of equations

$$x_i \equiv y_i^2 \pmod{N}.$$

Let us pick up a set $\mathscr{I}$ of indeces such that

$$\prod_{i \in \mathscr{I}} x_i = x^2.$$

Then we have

$$x^2 = \prod_{i \in \mathscr{I}} x_i \equiv \prod_{i \in \mathscr{I}} y_i^2 = y^2 \pmod{N};$$

$$(x - y)(x + y) = kN.$$

Fermat's Factorisation

We want to factorise a large number N. Suppose that we have a set of equations

$$x_i \equiv y_i^2 \pmod{N}.$$

Let us pick up a set $\mathscr{I}$ of indeces such that

$$\prod_{i \in \mathscr{I}} x_i = x^2.$$

Then we have

$$x^2 = \prod_{i \in \mathscr{I}} x_i \equiv \prod_{i \in \mathscr{I}} y_i^2 = y^2 \pmod{N};$$

$$(x - y)(x + y) = kN.$$

Fermat's Factorisation

We want to factorise a large number N. Suppose that we have a set of equations

$$x_i \equiv y_i^2 \pmod{N}.$$

Let us pick up a set $\mathscr{I}$ of indeces such that

$$\prod_{i \in \mathscr{I}} x_i = x^2.$$

Then we have

$$x^2 = \prod_{i \in \mathscr{I}} x_i \equiv \prod_{i \in \mathscr{I}} y_i^2 = y^2 \pmod{N};$$

$$(x - y)(x + y) = kN.$$

Choosing Dependencies

Each x_i has a prime decomposition $x_i = \prod p_j^{b_{ij}}$.

We want to find a set of indeces $\mathscr{I}$, such that, for each j,

$$\prod_{i\in\mathscr{I}} p_j^{b_{ij}}$$

is a square, that means

$$\sum_{i\in\mathscr{I}} b_{ij}$$

is even.

Choosing Dependencies

Each x_i has a prime decomposition $x_i = \prod p_j^{b_{ij}}$.

We want to find a set of indeces $\mathscr{I}$, such that, for each j,

$$\prod_{i \in \mathscr{I}} p_j^{b_{ij}}$$

is a square, that means

$$\sum_{i \in \mathscr{I}} b_{ij}$$

is even.

Choosing Dependencies

Each x_i has a prime decomposition $x_i = \prod p_j^{b_{ij}}$.

We want to find a set of indeces $\mathscr{I}$, such that, for each j,

$$\prod_{i \in \mathscr{I}} p_j^{b_{ij}}$$

is a square, that means

$$\sum_{i \in \mathscr{I}} b_{ij}$$

is even.

Linear System over GF(2)

We want, for each j,

$$\sum_{i \in \mathscr{I}} b_{ij} \equiv 0 \pmod 2.$$

Denote by $\vec{v}$ the incidence vector of I. Then, for each j,

$$\sum \cdot v_i \cdot b_{ij} \equiv 0 \pmod 2,$$

which can be rewritten as

$$\vec{v} \cdot B = \vec{o}$$

in the two element field GF(2).

Linear System over GF(2)

We want, for each j,

$$\sum_{i \in \mathscr{I}} b_{ij} \equiv 0 \pmod 2.$$

Denote by $\vec{v}$ the incidence vector of I. Then, for each j,

$$\sum \cdot v_i \cdot b_{ij} \equiv 0 \pmod 2,$$

which can be rewritten as

$$\vec{v} \cdot B = \vec{o}$$

in the two element field GF(2).

Linear System over GF(2)

We want, for each j,

$$\sum_{i \in \mathscr{I}} b_{ij} \equiv 0 \pmod{2}.$$

Denote by $\vec{v}$ the incidence vector of I. Then, for each j,

$$\sum \cdot v_i \cdot b_{ij} \equiv 0 \pmod{2},$$

which can be rewritten as

$$\vec{v} \cdot B = \vec{o}$$

in the two element field GF(2).

Solving Systems over GF(2)

The matrix B is very sparse, for every row there is only non-zero entries: the primes that are of an odd exponent in the decomposision of x_i.

Algorithms for solving huge sparse linear systems over GF(2):

- Reduced Gaussian elimination
- Block Lanczos algorithm
- Block Wiedemann algorithm

Solving Systems over GF(2)

The matrix B is very sparse, for every row there is only non-zero entries: the primes that are of an odd exponent in the decomposision of x_i.

Algorithms for solving huge sparse linear systems over GF(2):

- Reduced Gaussian elimination
- Block Lanczos algorithm
- Block Wiedemann algorithm

Solving Systems over GF(2)

The matrix B is very sparse, for every row there is only non-zero entries: the primes that are of an odd exponent in the decomposision of x_i.

Algorithms for solving huge sparse linear systems over GF(2):

- Reduced Gaussian elimination
- Block Lanczos algorithm
- Block Wiedemann algorithm

Solving Systems over GF(2)

The matrix B is very sparse, for every row there is only non-zero entries: the primes that are of an odd exponent in the decomposision of x_i.

Algorithms for solving huge sparse linear systems over GF(2):

- Reduced Gaussian elimination
- Block Lanczos algorithm
- Block Wiedemann algorithm

Quadratic Sieve

Let a, b be integers and consider the polynomial

$$Q(x) = (ax + b)^2 - N.$$

Let take an x and denote

$$Q(x) = p_1^{b_1} \cdots p_k^{b_k}.$$

Than we have

$$(ax + b)^2 \equiv p_1^{b_1} \cdots p_k^{b_k} \pmod{N}$$

and these are the relations needed for the Fermat's factorisation.

Quadratic Sieve

Let a, b be integers and consider the polynomial

$$Q(x) = (ax + b)^2 - N.$$

Let take an x and denote

$$Q(x) = p_1^{b_1} \cdots p_k^{b_k}.$$

Than we have

$$(ax + b)^2 \equiv p_1^{b_1} \cdots p_k^{b_k} \pmod{N}$$

and these are the relations needed for the Fermat's factorisation.

Smooth Relations

Let F be a bound on primes acceptable for the factorisation. A relation

$$(ax+b)^2 \equiv p_1^{b_1} \cdots p_k^{b_k} \pmod{N}$$

is called smooth is all the primes p_i are smaller than F.

We collect only smooth relations.

Smooth Relations

Let F be a bound on primes acceptable for the factorisation. A relation

$$(ax+b)^2 \equiv p_1^{b_1} \cdots p_k^{b_k} \pmod{N}$$

is called smooth is all the primes p_i are smaller than F.

We collect only smooth relations.

The Sieve

$Q(x)$ factorisation	2	3	5	$\cdots$	p	$\cdots$	
$\vdots$							
$Q(x)$					p		$\log Q(x) - \log p$
$Q(x+1)$							
$Q(x+2)$							
$\vdots$							
$Q(x+p)$					p		
$\vdots$							
$Q(x+2p)$					p		
$\vdots$							

The Sieve

$Q(x)$ factorisation	2	3	5	$\cdots$	p	$\cdots$	
$\vdots$							
$Q(x)$					p		$\log Q(x) - \log p$
$Q(x+1)$							
$Q(x+2)$							
$\vdots$							
$Q(x+p)$					p		
$\vdots$							
$Q(x+2p)$					p		
$\vdots$							

The Sieve

$Q(x)$ factorisation	2	3	5	$\cdots$	p	$\cdots$	
$\vdots$							
$Q(x)$					p		$\log Q(x) - \log p$
$Q(x+1)$							
$Q(x+2)$							
$\vdots$							
$Q(x+p)$					p		
$\vdots$							
$Q(x+2p)$					p		
$\vdots$							

The Sieve

$$
\begin{array}{r|ccccccc|l}
Q(x)\text{ factorisation} & 2 & 3 & 5 & \cdots & p & \cdots & & \\
\hline
\vdots & & & & & & & & \\
Q(x) & & & & & p & & & \log Q(x) - \log p \\
Q(x+1) & & & & & & & & \\
Q(x+2) & & & & & & & & \\
\vdots & & & & & & & & \\
Q(x+p) & & & & & p & & & \\
\vdots & & & & & & & & \\
Q(x+2p) & & & & & p & & & \\
\vdots & & & & & & & &
\end{array}
$$

Further Improvements

- Large prime vatiation (LPV)
- Double large prime variation (DLPV)
- Multi-polynomial quadratic sieve (MPQS)
- Self-initialization quadratic sieve (SIQS)

Further Improvements

- Large prime vatiation (LPV)
- Double large prime variation (DLPV)
- Multi-polynomial quadratic sieve (MPQS)
- Self-initialization quadratic sieve (SIQS)

Further Improvements

- Large prime vatiation (LPV)
- Double large prime variation (DLPV)
- Multi-polynomial quadratic sieve (MPQS)
- Self-initialization quadratic sieve (SIQS)

Further Improvements

- Large prime vatiation (LPV)
- Double large prime variation (DLPV)
- Multi-polynomial quadratic sieve (MPQS)
- Self-initialization quadratic sieve (SIQS)

An implementation

A multi-polynomial quadratic sieve was implemented in Prague

http://www.karlin.mff.cuni.cz/~krypto/mpqs.php

by M. Kechlibar, J. J. Zvánovec and P. Jedlička.