How Fast Can We Multiply Over $\mathrm{GF}(2)[x]$?

Paul Zimmermann
INRIA Lorraine/LORIA, Nancy, France

(thanks to Richard Brent, Pierrick Gaudry, Samuli Larvala, Emmanuel Thomé)

Followup to Mika's talk (preproceedings, p. 25)

Theorem. *The first digit of* $F_{5 \cdot 10^{87}}$ *is* 1.

Followup to Mika's talk (preproceedings, p. 25)

Theorem. *The first digit of* $F_{5\cdot 10^{87}}$ *is* 1.

Proof:

```
bash-3.00$ time ./fib 5e87
n=5000000000000000000000000000000000000000000\
  000000000000000000000000000000000000000000000
prec=302
length of Fib(n) in base 3 is
2190089397429712060570026179560455216382019945\
  882232014510953225378415664943464622853621
first digit is 1

user    0m0.004s
```

Credits: MPFR (`www.mpfr.org`), MPFI.

Plan of the talk

- **Theory**

Plan of the talk

- **Theory**
- **Algorithms**

Plan of the talk

- **Theory**
- **Algorithms**
- **Numbers**

Motivation: Search for Primitive Trinomials

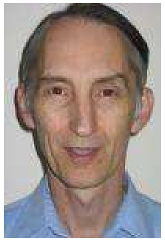
T. Kumada, H. Leeb, Y. Kurita and M. Matsumoto, *New primitive t-nomials $(t = 3, 5)$ over $\mathrm{GF}(2)$ whose degree is a Mersenne exponent*, Math. Comp., (2000):

$$x^{859433} + x^{288477} + 1.$$

Motivation: Search for Primitive Trinomials

T. Kumada, H. Leeb, Y. Kurita and M. Matsumoto, *New primitive t-nomials $(t = 3, 5)$ over $\mathrm{GF}(2)$ whose degree is a Mersenne exponent*, Math. Comp., (2000):

$$x^{859433} + x^{288477} + 1.$$

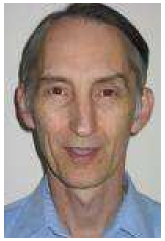


With Richard Brent and Samuli Larvala (Helsinki University of Technology), we started a search in 2000 . . .

Motivation: Search for Primitive Trinomials

T. Kumada, H. Leeb, Y. Kurita and M. Matsumoto, *New primitive t-nomials $(t = 3, 5)$ over $\mathrm{GF}(2)$ whose degree is a Mersenne exponent*, Math. Comp., (2000):

$$x^{859433} + x^{288477} + 1.$$



With Richard Brent and Samuli Larvala (Helsinki University of Technology), we started a search in 2000 . . .

June 26, 2000:

$$x^{859433} + x^{170340} + 1$$

Status so far

$$x^r + x^s + 1$$

r	s	when
756839	215747, 267428, 279695	June 2000
859433	170340, 288477	June 2000
3021377	361604, 1010202	July 2000 to April 2001: 13 GIPS-years
6972593	3037958	Feb 2001 to July 2003: 230 GIPS-years

Status so far

$$x^r + x^s + 1$$

r	s	when
756839	215747, 267428, 279695	June 2000
859433	170340, 288477	June 2000
3021377	361604, 1010202	July 2000 to April 2001: 13 GIPS-years
6972593	3037958	Feb 2001 to July 2003: 230 GIPS-years

Largest known primitive trinomial:

$$x^{6972593} + x^{3037958} + 1$$

(August 31, 2002, while Samuli Larvala and PZ were visiting Richard Brent in Oxford)

Status so far

$$x^r + x^s + 1$$

r	s	when
756839	215747, 267428, 279695	June 2000
859433	170340, 288477	June 2000
3021377	361604, 1010202	July 2000 to April 2001: 13 GIPS-years
6972593	3037958	Feb 2001 to July 2003: 230 GIPS-years

Largest known primitive trinomial:

$$x^{6972593} + x^{3037958} + 1$$

(August 31, 2002, while Samuli Larvala and PZ were visiting Richard Brent in Oxford)

As a comparison, RSA-155 (1999) took 8 GIPS years.

THEORY

GF(2)

Field with two elements: $\{0, 1\}$.

GF(2)

Field with two elements: $\{0, 1\}$.

Addition table:

$+$	0	1
0	0	1
1	1	0

GF(2)

Field with two elements: $\{0, 1\}$.

Addition table:

$+$	0	1
0	0	1
1	1	0

Multiplication table:

$\times$	0	1
0	0	0
1	0	1

$\mathrm{GF}(2)[x]$

Polynomial ring:

$$a(x) = a_d x^d + a_{d-1} x^{d-1} + \cdots + a_1 x + a_0,$$

where $a_i \in \{0, 1\}$.

$\mathrm{GF}(2)[x]$

Polynomial ring:

$$a(x) = a_d x^d + a_{d-1} x^{d-1} + \cdots + a_1 x + a_0,$$

where $a_i \in \{0, 1\}$.

If $a_d \neq 0$, $d = \deg(a)$.

Irreducible Polynomial

Definition. $a(x) \in \mathrm{GF}(2)[x]$ is *irreducible* if

$$a(x) = b(x)c(x)$$

implies $b(x) = 1$ or $c(x) = 1$.

Irreducible Polynomial

Definition. $a(x) \in \mathrm{GF}(2)[x]$ is *irreducible* if

$$a(x) = b(x)c(x)$$

implies $b(x) = 1$ or $c(x) = 1$.

Example 1. $x^3 + x + 1$ is irreducible.

Irreducible Polynomial

Definition. $a(x) \in \mathrm{GF}(2)[x]$ is *irreducible* if

$$a(x) = b(x)c(x)$$

implies $b(x) = 1$ or $c(x) = 1$.

Example 1. $x^3 + x + 1$ is irreducible.

Example 2. $x^3 + 1$ is not:

$$x^3 + 1 = (x^2 + x + 1)(x + 1)$$

Irreducible Polynomial

Definition. $a(x) \in \mathrm{GF}(2)[x]$ is *irreducible* if

$$a(x) = b(x)c(x)$$

implies $b(x) = 1$ or $c(x) = 1$.

Example 1. $x^3 + x + 1$ is irreducible.

Example 2. $x^3 + 1$ is not:

$$x^3 + 1 = (x^2 + x + 1)(x + 1)$$

Example 3. $x^4 + x^2 + x + 1$ is not either (irreducible over $\mathbb{Q}$):

$$x^4 + x^2 + x + 1 = (x^3 + x^2 + 1)(x + 1)$$

Why trinomials?

We search simple irreducible polynomials over $\mathrm{GF}(2)$.

$\Longrightarrow$ efficient way to implement arithmetic over $\mathrm{GF}(2^r)$.

Why trinomials?

We search simple irreducible polynomials over $\mathrm{GF}(2)$.

$\Longrightarrow$ efficient way to implement arithmetic over $\mathrm{GF}(2^r)$.

First try monomials: x^r is divisible by $x\ldots$

Why trinomials?

We search simple irreducible polynomials over $\mathrm{GF}(2)$.

$\Longrightarrow$ efficient way to implement arithmetic over $\mathrm{GF}(2^r)$.

First try monomials: x^r is divisible by $x\ldots$

Next try binomials: $x^r + 1$ is divisible by $x + 1$:

$$x^r + 1 = (x + 1)(x^{r-1} + x^{r-2} + \cdots + x + 1)$$

Why trinomials?

We search simple irreducible polynomials over $\mathrm{GF}(2)$.

$\Longrightarrow$ efficient way to implement arithmetic over $\mathrm{GF}(2^r)$.

First try monomials: x^r is divisible by $x\ldots$

Next try binomials: $x^r + 1$ is divisible by $x + 1$:

$$x^r + 1 = (x + 1)(x^{r-1} + x^{r-2} + \cdots + x + 1)$$

Then try trinomials:

$$x^r + x^s + 1 \qquad \text{with } r > s > 0.$$

Primitive Trinomials

Definition. A polynomial $f(x) \in \mathrm{GF}(2)[x]$ is said *primitive* iff:

$\mathbf{(1)}$ $f(x)$ is *irreducible*;

$\mathbf{(2a)}$ x has order $2^r - 1$ modulo $f(x)$, where $r := \deg(f)$;

Cf Lucas test (Nitin's talk).

Primitive Trinomials

Definition. A polynomial $f(x) \in \mathrm{GF}(2)[x]$ is said *primitive* iff:

$\mathbf{(1)}$ $f(x)$ is *irreducible*;

$\mathbf{(2a)}$ x has order $2^r - 1$ modulo $f(x)$, where $r := \deg(f)$;

Cf Lucas test (Nitin's talk).

Example 1. $x^4 + x + 1$ is primitive:

$$x, x^2, x^3, x+1, x^2+x, x^3+x^2, x^3+x+1, x^2+1, x^3+x, x^2+x+1, x^3+x^2+x,$$

$$x^3+x^2+x+1, x^3+x^2+1, x^3+1, x^{15} \equiv 1.$$

Primitive Trinomials

Definition. A polynomial $f(x) \in \mathrm{GF}(2)[x]$ is said *primitive* iff:

$\mathbf{(1)}$ $f(x)$ is *irreducible*;

$\mathbf{(2a)}$ x has order $2^r - 1$ modulo $f(x)$, where $r := \deg(f)$;

Cf Lucas test (Nitin's talk).

Example 1. $x^4 + x + 1$ is primitive:

$$x, x^2, x^3, x+1, x^2+x, x^3+x^2, x^3+x+1, x^2+1, x^3+x, x^2+x+1, x^3+x^2+x,$$

$$x^3+x^2+x+1, x^3+x^2+1, x^3+1, x^{15} \equiv 1.$$

Example 2. $x^6 + x^3 + 1$ is irreducible but not primitive:

$$x^9 \equiv 1 \bmod (x^6 + x^3 + 1).$$

How to Check Primitivity?

Follow the definition:

1. Check $f(x)$ is irreducible.

How to Check Primitivity?

Follow the definition:

1. Check $f(x)$ is irreducible.
2. Check $x^{2^r-1} = 1 \bmod f(x)$.

How to Check Primitivity?

Follow the definition:

1. Check $f(x)$ is irreducible.
2. Check $x^{2^r-1} = 1 \bmod f(x)$.
3. For each prime divisor p of $2^r - 1$, check

$$x^{(2^r-1)/p} \neq 1 \bmod f(x)$$

How to Check Primitivity?

Follow the definition:

1. Check $f(x)$ is irreducible.
2. Check $x^{2^r-1} = 1 \bmod f(x)$.
3. For each prime divisor p of $2^r - 1$, check

$$x^{(2^r-1)/p} \neq 1 \bmod f(x)$$

Need to factor $2^r - 1 \ldots$

How to Check Primitivity?

Follow the definition:

1. Check $f(x)$ is irreducible.
2. Check $x^{2^r-1} = 1 \bmod f(x)$.
3. For each prime divisor p of $2^r - 1$, check

$$x^{(2^r-1)/p} \neq 1 \bmod f(x)$$

Need to factor $2^r - 1 \ldots$

Easy if $2^r - 1$ is known to be prime:

$$f(x) \text{ irreducible} \Longrightarrow f(x) \text{ primitive}$$

Use Mersenne primes $2^r - 1$

Great Internet Mersenne Prime Search (GIMPS, `www.mersenne.org`).

George

Woltman

	r	date	$r \bmod 8$
M35	1398269	Nov 1996	5
M36	2976221	Aug 1997	5
M37	3021377	Jan 1998	1
M38	6972593	Jun 1999	1
M39	13466917	Nov 2001	5
M40?	20996011	Nov 2003	3
M41?	24036583	May 2004	7
M42?	25964951	Feb 2005	7
M43?	30402457	Dec 2005	1
M44?	32582657	Sep 2006	1

Do such trinomials always exist?

Does an irreducible/primitive trinomial exist for all degree r?

Do such trinomials always exist?

Does an irreducible/primitive trinomial exist for all degree r?

No!

Do such trinomials always exist?

Does an irreducible/primitive trinomial exist for all degree r?

No!

Example. No irreducible trinomial of degree 8:

$$x^8 + x + 1 = (x^6 + x^5 + x^3 + x^2 + 1)(x^2 + x + 1)$$

$$x^8 + x^2 + 1 = (x^4 + x + 1)^2$$

$$x^8 + x^3 + 1 = (x^3 + x + 1)(x^5 + x^3 + x^2 + x + 1)$$

$$x^8 + x^4 + 1 = (x^2 + x + 1)^4$$

Do such trinomials always exist?

Does an irreducible/primitive trinomial exist for all degree r?

No!

Example. No irreducible trinomial of degree 8:

$$x^8 + x + 1 = (x^6 + x^5 + x^3 + x^2 + 1)(x^2 + x + 1)$$

$$x^8 + x^2 + 1 = (x^4 + x + 1)^2$$

$$x^8 + x^3 + 1 = (x^3 + x + 1)(x^5 + x^3 + x^2 + x + 1)$$

$$x^8 + x^4 + 1 = (x^2 + x + 1)^4$$

In general, no irreducible trinomial of degree $r = 8k$.

Swan's Theorem (1962)

Previous work by von zur Gathen (2002), Dalen (1955), Dickson (1906), Stickelberger (1897), Pellet (1878), ...

Theorem. Suppose $r > s > 0$, $r - s$ odd. Then $x^r + x^s + 1$ has an even number of irreducible factors over $\mathrm{GF}(2)$ if and only if one of the following holds:

- r is even, $r \neq 2s$, $rs/2 \bmod 4 \in \{0, 1\}$;
- $2r \neq 0 \bmod s$, $r = \pm 3 \bmod 8$;
- $2r = 0 \bmod s$, $r = \pm 1 \bmod 8$.

Swan's Theorem (1962)

Previous work by von zur Gathen (2002), Dalen (1955), Dickson (1906), Stickelberger (1897), Pellet (1878), ...

Theorem. Suppose $r > s > 0$, $r - s$ odd. Then $x^r + x^s + 1$ has an even number of irreducible factors over $\mathrm{GF}(2)$ if and only if one of the following holds:

- r is even, $r \neq 2s$, $rs/2 \bmod 4 \in \{0, 1\}$;
- $2r \neq 0 \bmod s$, $r = \pm 3 \bmod 8$;
- $2r = 0 \bmod s$, $r = \pm 1 \bmod 8$.

Corollary 1. If r is prime, $r = \pm 3 \bmod 8$, $s \notin \{2, r-2\}$, then $x^r + x^s + 1$ is reducible.

$\Longrightarrow$ need to check only $x^r + x^2 + 1$.

Swan's Theorem (1962)

Previous work by von zur Gathen (2002), Dalen (1955), Dickson (1906), Stickelberger (1897), Pellet (1878), ...

Theorem. Suppose $r > s > 0$, $r - s$ odd. Then $x^r + x^s + 1$ has an even number of irreducible factors over $\mathrm{GF}(2)$ if and only if one of the following holds:

- r is even, $r \neq 2s$, $rs/2 \bmod 4 \in \{0, 1\}$;
- $2r \neq 0 \bmod s$, $r = \pm 3 \bmod 8$;
- $2r = 0 \bmod s$, $r = \pm 1 \bmod 8$.

Corollary 1. If r is prime, $r = \pm 3 \bmod 8$, $s \notin \{2, r-2\}$, then $x^r + x^s + 1$ is reducible.

$\Longrightarrow$ need to check only $x^r + x^2 + 1$.

Corollary 2. A trinomial of degree multiple of 8 cannot be irreducible.

$r = 8k$: pentanomials

Swan's theorem: no trinomial of degree $r = 8k$ can be irreducible.

How to perform efficient arithmetic in $\mathrm{GF}(2^r)$, say $\mathrm{GF}(2^{16})$?

Workaround: use a pentanomial

$$x^{16} + x^5 + x^3 + x + 1.$$

$r = 8k$: almost irreducible trinomials

(Richard Brent, PZ, 2003)

$$x^{19}+x^4+1 = (x^3+x+1)(x^{16}+x^{14}+x^{13}+x^{12}+x^9+x^7+x^6+x^5+x^2+x+1)$$

Perform all arithmetic modulo $x^{19} + x^4 + 1$.

Reduce mod $x^{16} + \cdots + 1$ only when a canonical form is needed.

ALGORITHMS

The Problem

Given a degree r with $2^r - 1$ prime.

The Problem

Given a degree r with $2^r - 1$ prime.

Goal 1. Find all irreducible (thus primitive) trinomials

$$x^r + x^s + 1.$$

The Problem

Given a degree r with $2^r - 1$ prime.

Goal 1. Find all irreducible (thus primitive) trinomials

$$x^r + x^s + 1.$$

Goal 2. (if possible) output a *certifi cate* which can be checked faster than the time to make it.

Certifi cates

Integer multiplication:

$$395718860534 \cdot 193139816415 \Rightarrow 76429068075489748865610$$

Difficult to exhibit a certificate which can be checked faster!

Certifi cates

Integer multiplication:

$$395718860534 \cdot 193139816415 \Rightarrow 76429068075489748865610$$

Difficult to exhibit a certificate which can be checked faster!

Integer factorization:

$$179435405554681543034355 \Rightarrow 224241704651 \cdot 800187484459$$

One factor is a valid certificate.

Do not waste a factor of two!

One of Schönhage's **golden rules**.

$$x^r + x^s + 1 = a(x)b(x) \Longrightarrow 1 + x^{r-s} + x^r = x^r a(1/x)b(1/x)$$

$\Longrightarrow$ can restrict to $s \le r/2$.

Main Theorem

Theorem. The product of $\mathbf{ALL}$ irreducible factors of degree $\mathbf{dividing}$ k is $x^{2^k} + x$.

$$x^{2^1} + x = x(x+1)$$

$$x^{2^2} + x = x(x+1)(x^2+x+1)$$

$$x^{2^3} + x = x(x+1)(x^3+x+1)(x^3+x^2+1)$$

$$x^{2^4} + x = x(x+1)(x^2+x+1)(x^4+x+1)(x^4+x^3+1)(x^4+x^3+x^2+x+1)$$

The old algorithm

1. (sieving) for $k = 2$ to k_0, compute:

$$\gcd(x^{2^k} + x, x^r + x^s + 1)$$

If non trivial, output “divisible by degree k”

(When 2^k exceeds r, reduce mod $x^r + x^s + 1$.)

The old algorithm

1. (sieving) for $k = 2$ to k_0, compute:

$$\gcd(x^{2^k} + x, x^r + x^s + 1)$$

If non trivial, output "divisible by degree k"

(When 2^k exceeds r, reduce mod $x^r + x^s + 1$.)

2. (full test) check whether:

$$x^{2^r} \equiv x \bmod (x^r + x^s + 1).$$

If not, output the low bits from $x^{2^r} \bmod (x^r + x^s + 1)$ as pseudo-certificate.

The old algorithm

1. (sieving) for $k = 2$ to k_0, compute:

$$\gcd(x^{2^k} + x, x^r + x^s + 1)$$

If non trivial, output “divisible by degree k”

(When 2^k exceeds r, reduce mod $x^r + x^s + 1$.)

2. (full test) check whether:

$$x^{2^r} \equiv x \bmod (x^r + x^s + 1).$$

If not, output the low bits from $x^{2^r} \bmod (x^r + x^s + 1)$ as pseudo-certificate.

For $r = 6972593$, we used $k_0 = 26$: 236244 trinomials (7%) survived Step 1.

Complexity: $O(r^2)$ for each full test.

The “new” algorithm

Perform a classical DDF (distinct degree factorization) with the “blocking strategy” (von zur Gathen and Shoup 1992, Kaltofen and Shoup 1998):

0. Partition $\{2, \ldots, \lfloor r/2 \rfloor\}$ into intervals $I_1, \ldots, I_m$.

1. for $j := 1$ to m do

    $a \leftarrow 1$; for k in I_j do

        $b \leftarrow x^{2^k} \bmod (x^r + x^s + 1)$ **[SQR]**

        $a \leftarrow a(b + x) \bmod (x^r + x^s + 1)$ **[MUL]**

    $g \leftarrow \gcd(a, x^r + x^s + 1)$ **[GCD]**

    if $g \neq 1$ then output “reducible with degree in I_j”

Output “irreducible”.

Complexity: $O(dM(r))$ if the smallest factor has degree d, assuming the GCD cost is not dominant.

Complexity Analysis (sketch)

Old Algorithm: we sieve up to $k \approx \log r$.

Complexity Analysis (sketch)

Old Algorithm: we sieve up to $k \approx \log r$.

- it remains $\approx \frac{r}{\log r}$ trinomials

Complexity Analysis (sketch)

Old Algorithm: we sieve up to $k \approx \log r$.

- it remains $\approx \frac{r}{\log r}$ trinomials
- cost $\approx r^2$ per full test, total cost $\approx \frac{r^3}{\log r}$

Complexity Analysis (sketch)

Old Algorithm: we sieve up to $k \approx \log r$.

- it remains $\approx \frac{r}{\log r}$ trinomials
- cost $\approx r^2$ per full test, total cost $\approx \frac{r^3}{\log r}$

New Algorithm: cost $O(dM(r))$ if the smallest factor has degree d.

Complexity Analysis (sketch)

Old Algorithm: we sieve up to $k \approx \log r$.

- it remains $\approx \frac{r}{\log r}$ trinomials
- cost $\approx r^2$ per full test, total cost $\approx \frac{r^3}{\log r}$

New Algorithm: cost $O(dM(r))$ if the smallest factor has degree d.

- $\Pr[\text{no factor of degree} < d] \approx \frac{1}{d}$

Complexity Analysis (sketch)

Old Algorithm: we sieve up to $k \approx \log r$.

- it remains $\approx \frac{r}{\log r}$ trinomials
- cost $\approx r^2$ per full test, total cost $\approx \frac{r^3}{\log r}$

New Algorithm: cost $O(dM(r))$ if the smallest factor has degree d.

- $\Pr[\text{no factor of degree} < d] \approx \frac{1}{d}$
- total cost $r \sum_{d=1}^{r/2} \frac{1}{d} M(r) \approx rM(r) \log r$

Complexity Analysis (sketch)

Old Algorithm: we sieve up to $k \approx \log r$.

- it remains $\approx \frac{r}{\log r}$ trinomials
- cost $\approx r^2$ per full test, total cost $\approx \frac{r^3}{\log r}$

New Algorithm: cost $O(dM(r))$ if the smallest factor has degree d.

- $\Pr[\text{no factor of degree} < d] \approx \frac{1}{d}$
- total cost $r \sum_{d=1}^{r/2} \frac{1}{d} M(r) \approx rM(r) \log r$

New algorithm faster as soon as $M(r) \ll \frac{r^2}{\log^2 r}$.

Complexity Analysis (sketch)

Old Algorithm: we sieve up to $k \approx \log r$.

- it remains $\approx \frac{r}{\log r}$ trinomials
- cost $\approx r^2$ per full test, total cost $\approx \frac{r^3}{\log r}$

New Algorithm: cost $O(dM(r))$ if the smallest factor has degree d.

- $\Pr[\text{no factor of degree} < d] \approx \frac{1}{d}$
- total cost $r \sum_{d=1}^{r/2} \frac{1}{d} M(r) \approx r M(r) \log r$

New algorithm faster as soon as $M(r) \ll \frac{r^2}{\log^2 r}$.

With R. Brent: a faster algorithm in $O(r^2 \log r \sqrt{M(r)/r})$, but no space in the margin. . .

NUMBERS

Binary Polynomials

$a(x) = a_{r-1}x^{r-1} + \cdots + a_1 x + a_0$ is stored in computer by the *binary polynomial*

$$a(2) = a_{r-1} \cdot 2^{r-1} + \cdots + a_1 \cdot 2 + a_0.$$

On a 8-bit computer, the trinomial $x^{19} + x^4 + 1$ is stored as:

$$\underbrace{\boxed{00001000}}_{x^3 \cdot x^{16}}\underbrace{\boxed{00000000}}_{0 \cdot x^8}\underbrace{\boxed{00010001}}_{(x^4+1) \cdot x^0}$$

Addition of Binary Polynomials

$x^{15}+x^{13}+x^{12}+x^{11}+x^{9}+x^{8}+x^{6}+x^{4}+x^{3}+x^{2}$	10111011	01011100
$x^{15}+x^{12}+x^{11}+x^{10}+x^{9}+x^{7}+x^{6}+x^{5}+x^{4}+x^{2}+x$	10011110	11110110
$x^{13}+x^{10}+x^{8}+x^{7}+x^{5}+x^{3}+x$	00100101	10101010

Multiplication by x^k

$a = x^{13} + x^{12} + x^{11} + x^9 + x^8 + x^6 + x^4 + x^3 + x^2$ | 00111011 | 01011100 |

$x^2 a = x^{15} + x^{14} + x^{13} + x^{11} + x^{10} + x^8 + x^6 + x^5 + x^4$ | 11101101 | 01110000 |

Multiplication

$$(x^6 + x^4 + x^3 + x^2)(x^5 + x^4 + x^3 + x + 1)$$

$$\begin{array}{r} \boxed{01011100} \\ \times \ \boxed{00111011} \\ \hline \end{array}$$

Multiplication

$(x^6 + x^4 + x^3 + x^2)(x^5 + x^4 + x^3 + x + 1)$

$$\begin{array}{r} \boxed{01011100} \\ \times\ \boxed{00111011} \\ \hline \boxed{01011100} \end{array}$$

Multiplication

$(x^6 + x^4 + x^3 + x^2)(x^5 + x^4 + x^3 + x + 1)$

$$
\begin{array}{r}
\boxed{01011100} \\
\times \ \boxed{00111011} \\
\hline
\boxed{01011100} \\
\boxed{01011100}
\end{array}
$$

Multiplication

$$(x^6 + x^4 + x^3 + x^2)(x^5 + x^4 + x^3 + x + 1)$$

```
             01011100
          ×  00111011
          -----------
             01011100
            01011100
          01011100
```

Multiplication

$(x^6 + x^4 + x^3 + x^2)(x^5 + x^4 + x^3 + x + 1)$

$$
\begin{array}{r}
\boxed{01011100} \\
\times \boxed{00111011} \\
\hline
\boxed{01011100} \\
\boxed{01011100} \\
\boxed{01011100} \\
\boxed{01011100}
\end{array}
$$

Multiplication

$(x^6 + x^4 + x^3 + x^2)(x^5 + x^4 + x^3 + x + 1)$

$$
\begin{array}{r}
\boxed{01011100} \\
\times\ \boxed{00111011} \\
\hline
\boxed{01011100} \\
\boxed{01011100} \\
\boxed{01011100} \\
\boxed{01011100} \\
+\ \boxed{01011100}
\end{array}
$$

Multiplication

$$(x^6 + x^4 + x^3 + x^2)(x^5 + x^4 + x^3 + x + 1)$$

$$
\begin{array}{r}
\boxed{01011100} \\
\times \ \boxed{00111011} \\
\hline
\boxed{01011100} \\
\boxed{01011100} \\
\boxed{01011100} \\
\boxed{01011100} \\
+ \ \boxed{01011100} \\
\hline
\boxed{0000110001000100}
\end{array}
$$

$$x^{11} + x^{10} + x^6 + x^2$$

Squares are easy:

$$x^t + x^u + \cdots \qquad \Longrightarrow \qquad x^{2t} + x^{2u} + \cdots$$

GCDs reduce to multiplication: $O(M(r) \log r)$

$\Longrightarrow$ We have to improve multiplications!

Multiplication over $\mathrm{GF}(2)[x]$

- naive (quadratic) algorithm
- Karatsuba's algorithm
- Toom-Cook 3-way and higher order
- Fast Fourier Transform: segmentation, Cantor (BiPolAr), Schönhage

Schönhage's Algorithm

Schnelle Multiplikation von Polynomen über Körpern der Charakteristik 2, A. Schönhage, *Acta Inf. 7* (1977), 395–398.

Complexity $O(r \log r \log\log r)$.

High-level description:

one product $\bmod (x^{2N} + x^N + 1) \quad \Longrightarrow \quad 2K$ products $\bmod (x^{2L} + x^L + 1)$

Constraints: K power of 3, $L \geq N/K$, L multiple of K

Variant described here:

one product $\bmod (x^N + 1) \quad \Longrightarrow \quad K$ products $\bmod (x^{2L} + x^L + 1)$

Constraints: K power of 3, $L \geq N/K$, L multiple of $K/3$

Forward and backward transform: $O(K \log K)$ additions/shifts mod $x^{2L} + x^L + 1$.

Pointwise products: K products mod $x^{2L} + x^L + 1$.

The Algorithm

Input: a, b polynomials of degree $< N$

Parameters: K power of 3 dividing N, $M = N/K$, $L \geq M$ multiple of $K/3$.

1. Decompose a, b in base x^M:

$$a(x) = \sum_{i=0}^{K-1} a_i(x) x^{iM}$$

2. Forward transform with $\omega = x^{3L/K}$:

$$\hat{a}_i = \sum_{j=0}^{K-1} a_i(x) \omega^{ij} \bmod (x^{2L} + x^L + 1), 0 \leq i < K$$

3. Pointwise products:

$$\hat{c}_i = \hat{a}_i \hat{b}_i, \quad 0 \leq i < K$$

4. Backward transform:

$$c_\ell = \sum_{i=0}^{K-1} \hat{c}_i(x)\omega^{-\ell i} \bmod (x^{2L} + x^L + 1), 0 \leq \ell < K$$

5. Recomposition:

$$c(x) = \sum_{\ell=0}^{K-1} c_\ell x^{\ell M} \bmod (x^N + 1).$$

An example

Compute $a(x)b(x) \bmod (x^{15}+1)$:

$$a(x) = x^{14}+x^{13}+x^{12}+x^{11}+x^{10}+x^{8}+x^{6}+x^{5}+x^{4}+x^{3}+x^{2}+1,$$

$$b(x) = x^{13}+x^{11}+x^{8}+x^{7}+x^{6}+x^{2}.$$

Take $K=3$, $L=5$:

$$a_2 = x^4+x^3+x^2+x+1, a_1 = x^3+x+1, a_0 = x^4+x^3+x^2+1$$

$$b_2 = x^3+x, b_1 = x^3+x^2+x, b_0 = x^2$$

Forward transform ($\omega = x^5$, mod $x^{10}+x^5+1$):

$$\hat{a}_2 = x^{20}a_2 + x^{10}a_1 + a_0 = x^9+x^7+x^4+x^2+x$$

$$\hat{a}_1 = x^{10}a_2 + x^{5}a_1 + a_0 = x^9+x^7+x$$

$$\hat{a}_0 = a_2 + a_1 + a_0 = x^3+1$$

An example

Forward transform ($\omega = x^5$, mod $x^{10} + x^5 + 1$):

$$\hat{a}_2 = x^9 + x^7 + x^4 + x^2 + x, \hat{a}_1 = x^9 + x^7 + x, \hat{a}_0 = x^3 + 1$$

$$\hat{b}_2 = x^7 + x^3 + x, \hat{b}_1 = x^7 + x^3 + x^2 + x, \hat{b}_0 = 0$$

Pointwise transforms:

$$\hat{c}_2 = x^6 + x^3, \hat{c}_1 = x^7 + x^6 + x^3, \hat{c}_0 = 0$$

Backward transform:

$$c_2 = x^6 + x^3, c_1 = x^7 + x^6 + x^3, c_0 = 0$$

Reconstruction:

$$c_2 x^{10} + c_1 x^5 + c_0 = x^{13} + x^{12} + x^{11} + x^8 + x^2 + x \bmod (x^{15} + 1)$$

Why does it work?

Let $R_L := \mathrm{GF}(2)[x]/(x^{2L} + x^L + 1)$.

$\omega = x^{3L/K} \implies \omega^{K/3} = x^L$ thus in R_L:

$$\omega^{2K/3} + \omega^{K/3} + 1 = 0 \tag{1}$$

From Eq. (1) it follows

$$\omega^K = 1 \quad \text{and} \quad \omega^{-1} = \omega^{K-1} \tag{2}$$

$$\begin{aligned} c_\ell := \sum_{i=0}^{K-1} \hat{c}_i(x)\omega^{-\ell i} &= \sum_{i=0}^{K-1} \omega^{-\ell i} \left(\sum_{j=0}^{K-1} \omega^{ij} a_i\right) \left(\sum_{k=0}^{K-1} \omega^{ik} b_k\right) \\ &= \sum_{j=0}^{K-1} \sum_{k=0}^{K-1} a_j b_k \sum_{i=0}^{K-1} \omega^{i(j+k-\ell)}. \end{aligned}$$

Why does it work?

$$c_\ell = \sum_{j=0}^{K-1} \sum_{k=0}^{K-1} a_j b_k \sum_{i=0}^{K-1} \omega^{i(j+k-\ell)}$$

We have $-K < j + k - \ell < 2K$. If $t := j + k - \ell \neq 0 \bmod K$:

$$\sum_{i=0}^{K-1} \omega^{i(j+k-\ell)} = \frac{\omega^{Kt} + 1}{\omega^t + 1} = 0.$$

Otherwise $j + k - \ell \in \{0, K\}$, and $\omega^{i(j+k-\ell)} = 1$.

Thus $\sum_{i=0}^{K-1} \omega^{i(j+k-\ell)}$ is non-zero only when $j + k - \ell \in \{0, K\}$, in which case it equals $K = 1 \bmod 2$.

It follows:

$$c_\ell = \sum_{j+k=\ell} a_j b_k + \sum_{j+k=K+\ell} a_j b_k \pmod{x^{2L} + x^L + 1}.$$

Why does it work?

$$c_\ell = \sum_{j+k=\ell} a_j b_k + \sum_{j+k=K+\ell} a_j b_k \pmod{x^{2L} + x^L + 1}.$$

Recall $\deg(a_j), \deg(b_k) < M$: if $L \geq M$, then

$$c_\ell = \sum_{j+k=\ell} a_j b_k + \sum_{j+k=K+\ell} a_j b_k.$$

5. Recomposition:

$$c(x) = \sum_{\ell=0}^{K-1} c_\ell x^{\ell M} \bmod (x^N + 1).$$

$c(x)$ is simply the cyclic convolution of $a(x)$ and $b(x)$ mod $x^N + 1$.

Arithmetic Modulo $x^{2L} + x^L + 1$

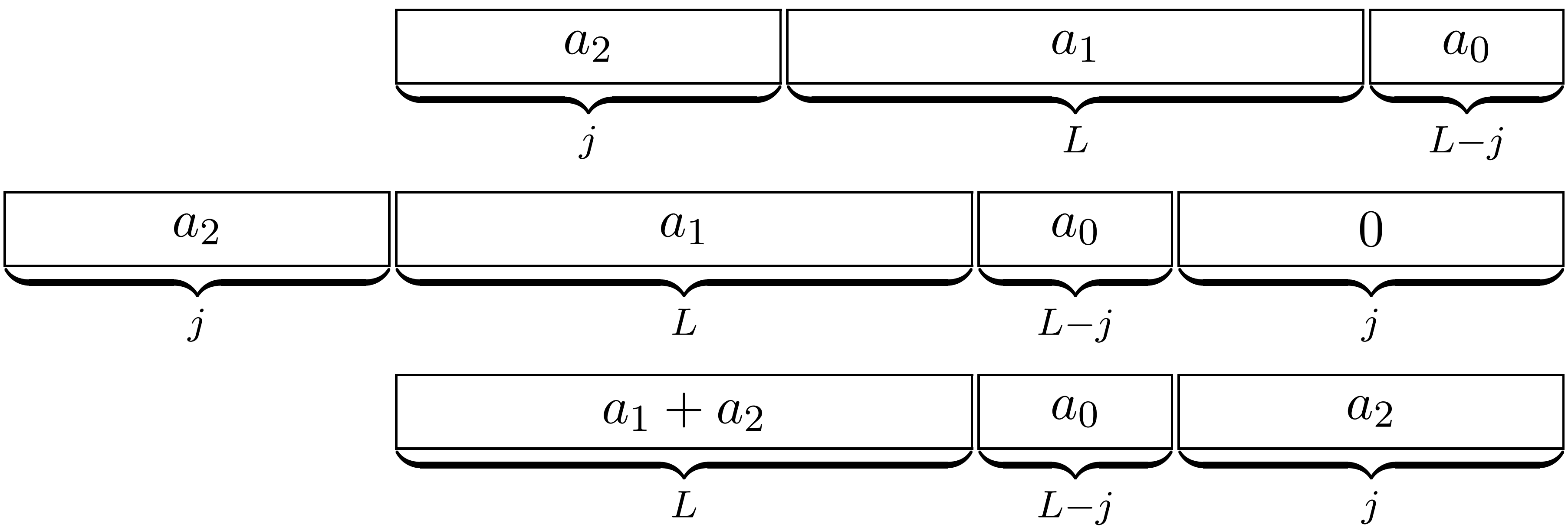
- addition: easy
- shift: multiplication by x^j, $0 \leq j < 3L$
- full multiplication

Shifts Modulo $x^{2L} + x^L + 1$

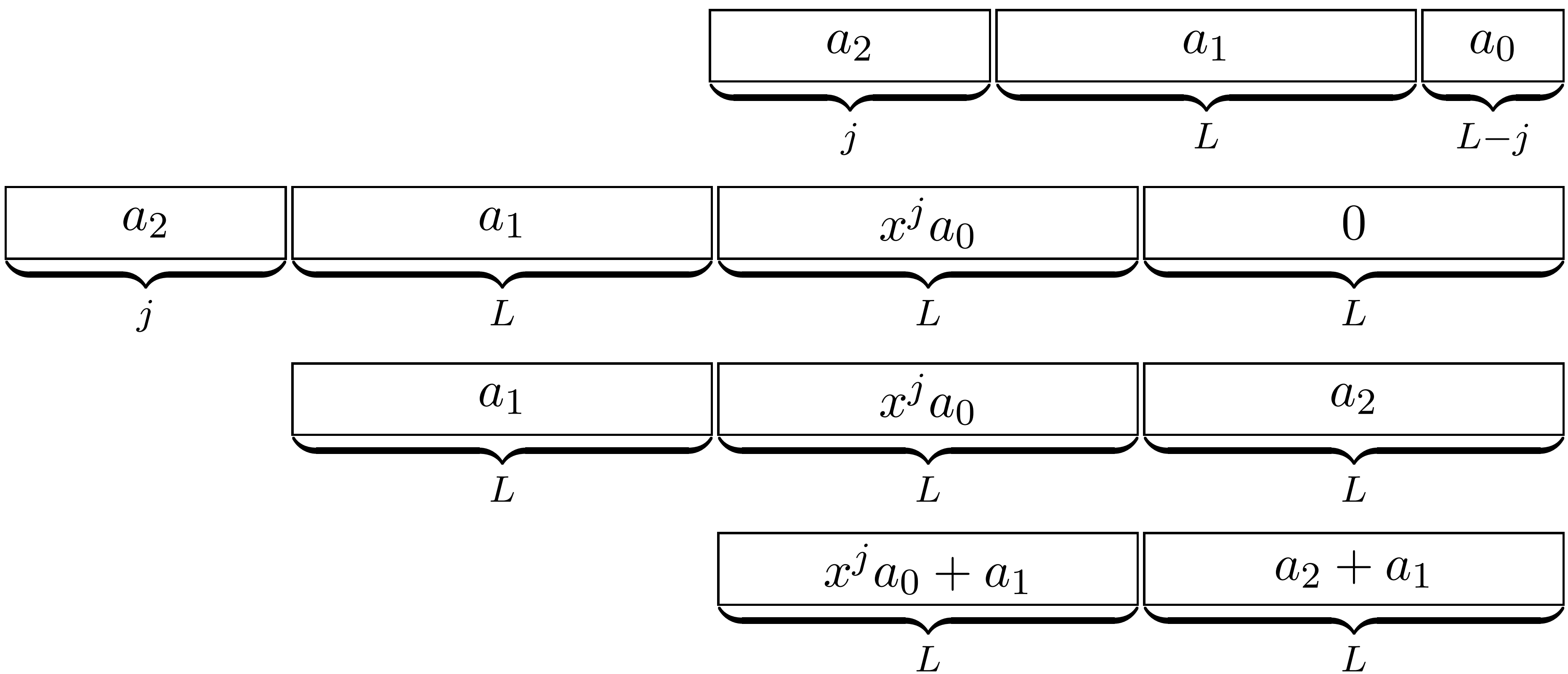
Input: a binary polynomial $a(x)$ of degree $< 2L$, $0 \le j < 3L$

Output: $x^j a(x) \bmod (x^{2L} + x^L + 1)$

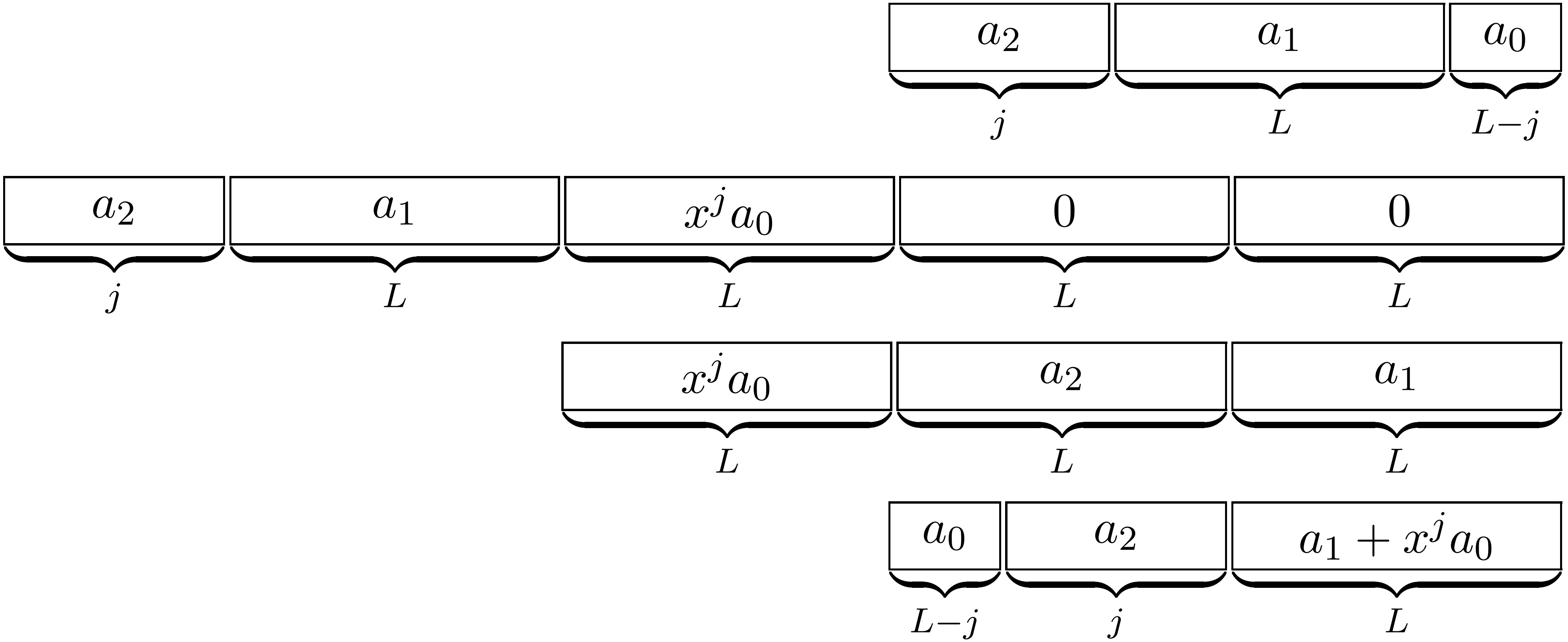
1. Shift of j, $0 \le j < L$:



Case 2: Shift of $L + j, 0 \le j < L$



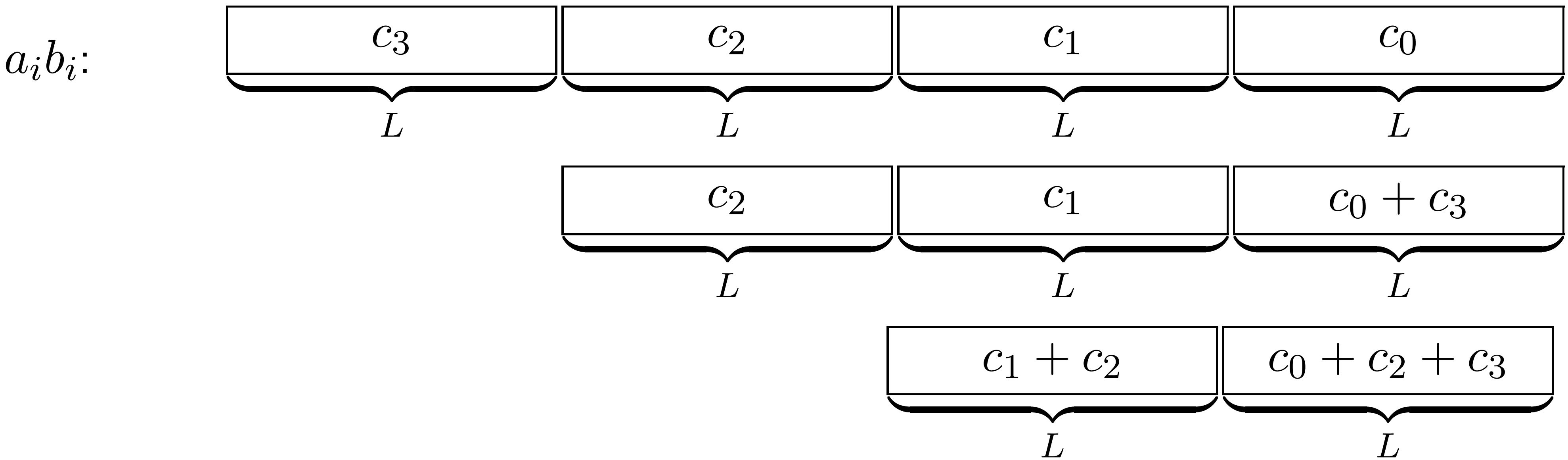
Case 3: Shift of $2L + j, 0 \leq j < L$



Multiplication mod $x^{2L} + x^L + 1$

3. Pointwise products

$$\hat{c}_i = \hat{a}_i \hat{b}_i (\bmod x^{2L} + x^L + 1)$$



Timings

Core 2 processor, 2.66Ghz, 4MB cache, 3GB memory.

r	Toom-Cook 3	Toom-Cook 4	FFTMul(K)	GCD
6972593	1.32s	1.01s	0.27s(6561)	12.1s
24036583	7.89s	6.30s	1.77s(6561)	55.3s
32582657	13.9s	8.11s	2.16s(6561)	78.4s

6972593 again

From April 18 to April 29, 2007, we started the computation of extended logs for $r = 6972593$ using about 25 Opterons (2.2Ghz and 2.4Ghz).

6972593 again

From April 18 to April 29, 2007, we started the computation of extended logs for $r = 6972593$ using about 25 Opterons (2.2Ghz and 2.4Ghz).

It took a total of about 0.44 cpu year, or about 1 GIPS-year.

6972593 again

From April 18 to April 29, 2007, we started the computation of extended logs for $r = 6972593$ using about 25 Opterons (2.2Ghz and 2.4Ghz).

It took a total of about 0.44 cpu year, or about 1 GIPS-year.

And checking all certificates took only 2 hours with Magma!

6972593 again

From April 18 to April 29, 2007, we started the computation of extended logs for $r = 6972593$ using about 25 Opterons (2.2Ghz and 2.4Ghz).

It took a total of about 0.44 cpu year, or about 1 GIPS-year.

And checking all certificates took only 2 hours with Magma!

Speedup of about 230 due to:

6972593 again

From April 18 to April 29, 2007, we started the computation of extended logs for $r = 6972593$ using about 25 Opterons (2.2Ghz and 2.4Ghz).

It took a total of about 0.44 cpu year, or about 1 GIPS-year.

And checking all certificates took only 2 hours with Magma!

Speedup of about 230 due to:

- Schönhage's multiplication (with classical DDF and blocking strategy)

6972593 again

From April 18 to April 29, 2007, we started the computation of extended logs for $r = 6972593$ using about 25 Opterons (2.2Ghz and 2.4Ghz).

It took a total of about 0.44 cpu year, or about 1 GIPS-year.

And checking all certificates took only 2 hours with Magma!

Speedup of about 230 due to:

- Schönhage's multiplication (with classical DDF and blocking strategy)
- new multi-level blocking DDF algorithm (with R. Brent)

6972593 again

From April 18 to April 29, 2007, we started the computation of extended logs for $r = 6972593$ using about 25 Opterons (2.2Ghz and 2.4Ghz).

It took a total of about 0.44 cpu year, or about 1 GIPS-year.

And checking all certificates took only 2 hours with Magma!

Speedup of about 230 due to:

- Schönhage's multiplication (with classical DDF and blocking strategy)
- new multi-level blocking DDF algorithm (with R. Brent)
- faster basecase multiplication (with P. Gaudry and E. Thomé): about 130 cycles for $128 \times 128 \rightarrow 256$ (Core 2)

6972593 again

From April 18 to April 29, 2007, we started the computation of extended logs for $r = 6972593$ using about 25 Opterons (2.2Ghz and 2.4Ghz).

It took a total of about 0.44 cpu year, or about 1 GIPS-year.

And checking all certificates took only 2 hours with Magma!

Speedup of about 230 due to:

- Schönhage's multiplication (with classical DDF and blocking strategy)
- new multi-level blocking DDF algorithm (with R. Brent)
- faster basecase multiplication (with P. Gaudry and E. Thomé): about 130 cycles for $128 \times 128 \to 256$ (Core 2)
- subquadratic GCD (still quite expensive)

24036583

We have started computations for $r = 24036583$ (M41?) on April 25.

Already done more than 10%.

No primitive trinomial so far.

But already found a (smallest) factor of degree almost one million!

Help welcome (preferably Opteron/Core 2)!

Some Conclusions

Some Conclusions

- **slow** algorithms are **slow** as expected

Some Conclusions

- **slow** algorithms are **slow** as expected
- **fast** algorithms are indeed asymptotically **faster**

Some Conclusions

- **slow** algorithms are **slow** as expected
- **fast** algorithms are indeed asymptotically **faster**
- thanks to Moore's law, the **asymptotic** domain is closer and closer...

Some Conclusions

- **slow** algorithms are **slow** as expected
- **fast** algorithms are indeed asymptotically **faster**
- thanks to Moore's law, the **asymptotic** domain is closer and closer...

Thank you for staying awake so far!