

Complexity news:

FFTs and integer multiplication

D. J. Bernstein

University of Illinois at Chicago

# Advertisement

Fix a field  $k$  with  $2 \neq 0$ .

Fix non-square  $d \in k$ .

$$\{(x, y) \in k \times k : \\ x^2 + y^2 = 1 + dx^2y^2\}$$

is a commutative group with

$$(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$$

defined by Edwards addition law:

$$x_3 = \frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2},$$

$$y_3 = \frac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2}.$$

More on Edwards coordinates:

Harold M. Edwards,

“A normal form  
for elliptic curves.”

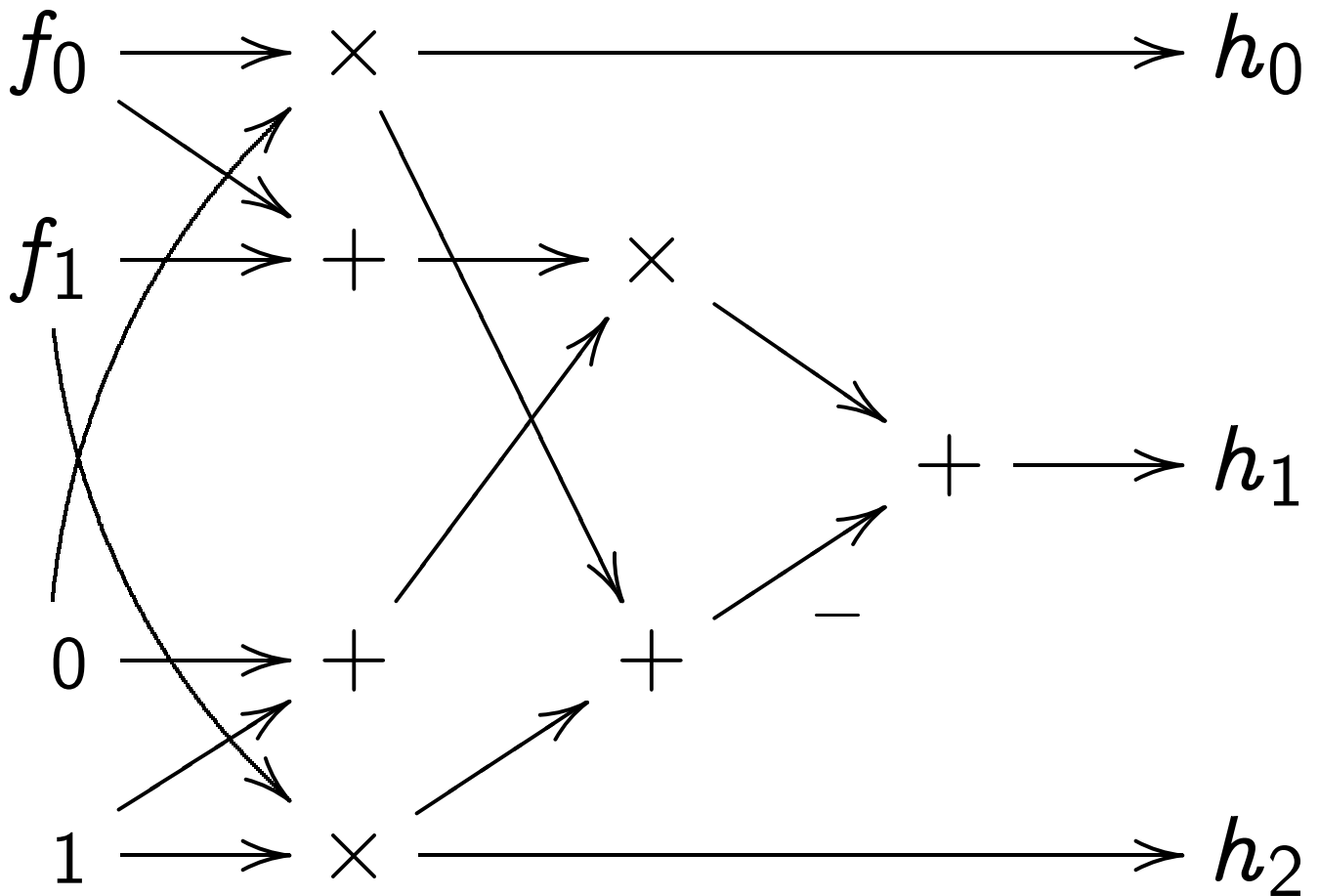
Daniel J. Bernstein, Tanja Lange,

“Faster addition and doubling  
on elliptic curves.”

[http://cr.yp.to  
/newelliptic.html](http://cr.yp.to/newelliptic.html)

# Algebraic complexity

Example (1963 Karatsuba):



This “**R**-algebraic algorithm”  
computes product  $h_0 + h_1x + h_2x^2$   
of  $f_0 + f_1x$ ,  $0 + 1x \in \mathbf{R}[x]$ .

“Total **R**-complexity” 7:

3 mults, 4 adds/subs.

## FFT news

How quickly can we multiply  
in the ring  $\mathbf{R}[x]$ ?

1866 Gauss “FFT”  $\Rightarrow$

algebraic algorithm to compute

$f, g \mapsto fg$  when  $\deg f, \deg g < n$ ;

$\mathbf{R}$ -complexity  $(15 + o(1))n \lg n$ .

1968 R. Yavne: Can do better!

“Split-radix FFT.”

$\mathbf{R}$ -complexity  $(12 + o(1))n \lg n$ .

2004 James Van Buskirk:

Can do better! “Tangent FFT.”

$\mathbf{R}$ -complexity  $(34/3 + o(1))n \lg n$ .

The FFT trick:

Unique  $\mathbf{C}[x]$ -algebra morphism

$$\mathbf{C}[x]/(x^{1024} - 1) \hookrightarrow$$

$$\mathbf{C}[x]/(x^{512} - 1) \oplus \mathbf{C}[x]/(x^{512} + 1).$$

Gauss continuation:

$$\mathbf{C}[x]/(x^{512} + 1) \hookrightarrow$$

$$\mathbf{C}[x]/(x^{256} - i) \oplus \mathbf{C}[x]/(x^{256} + i)$$

where  $i = \sqrt{-1}$ ; etc.

Alternative: Twist.

Unique  $\mathbf{C}$ -algebra morphism

$$\mathbf{C}[x]/(x^{512} + 1) \hookrightarrow \mathbf{C}[y]/(y^{512} - 1)$$

that maps  $x$  to  $\zeta_{1024}y$

where  $\zeta_{1024} = \exp(2\pi i/1024)$ .

Same complexity as Gauss.

Split-radix FFT:

Don't twist  $\mathbf{C}[x]/(x^{512} + 1)$

but do twist  $\mathbf{C}[x]/(x^{256} \pm i)$ .

This maximizes the number of multiplications by 4th roots of 1.

General principle:

“Align roots of 1.”

Use  $\mathbf{C}[x]/(x^{\dots} - \omega)$

if  $\omega$  is “aligned”

for easy multiplications.

Twist to avoid  $\mathbf{C}[x]/(x^{\dots} - \omega)$

if  $\omega$  is not aligned.

The tangent FFT:

Rethink basis of  $\mathbf{C}[x]/(x^n - 1)$ .

Instead of  $1, x, \dots, x^{n-1}$  use

$1/s_{n,0}, x/s_{n,1}, \dots, x^{n-1}/s_{n,n-1}$

where  $s_{n,k} =$

$$\max\left\{\left|\cos\frac{2\pi k}{n}\right|, \left|\sin\frac{2\pi k}{n}\right|\right\}.$$

$$\max\left\{\left|\cos\frac{2\pi k}{n/4}\right|, \left|\sin\frac{2\pi k}{n/4}\right|\right\}.$$

$$\max\left\{\left|\cos\frac{2\pi k}{n/16}\right|, \left|\sin\frac{2\pi k}{n/16}\right|\right\}.$$

$\dots$

Note that  $s_{n,k} = s_{n,k+n/4}$ .

Note that  $\zeta_n^k (s_{n/4,k}/s_{n,k})$  is

$\pm(1 + i \tan \dots)$  or  $\pm(\cot \dots + i)$ .

Complexity  $8.5n - 28$  to split  $n$   
into  $n/4, n/4, n/4, n/8, n/8$ .



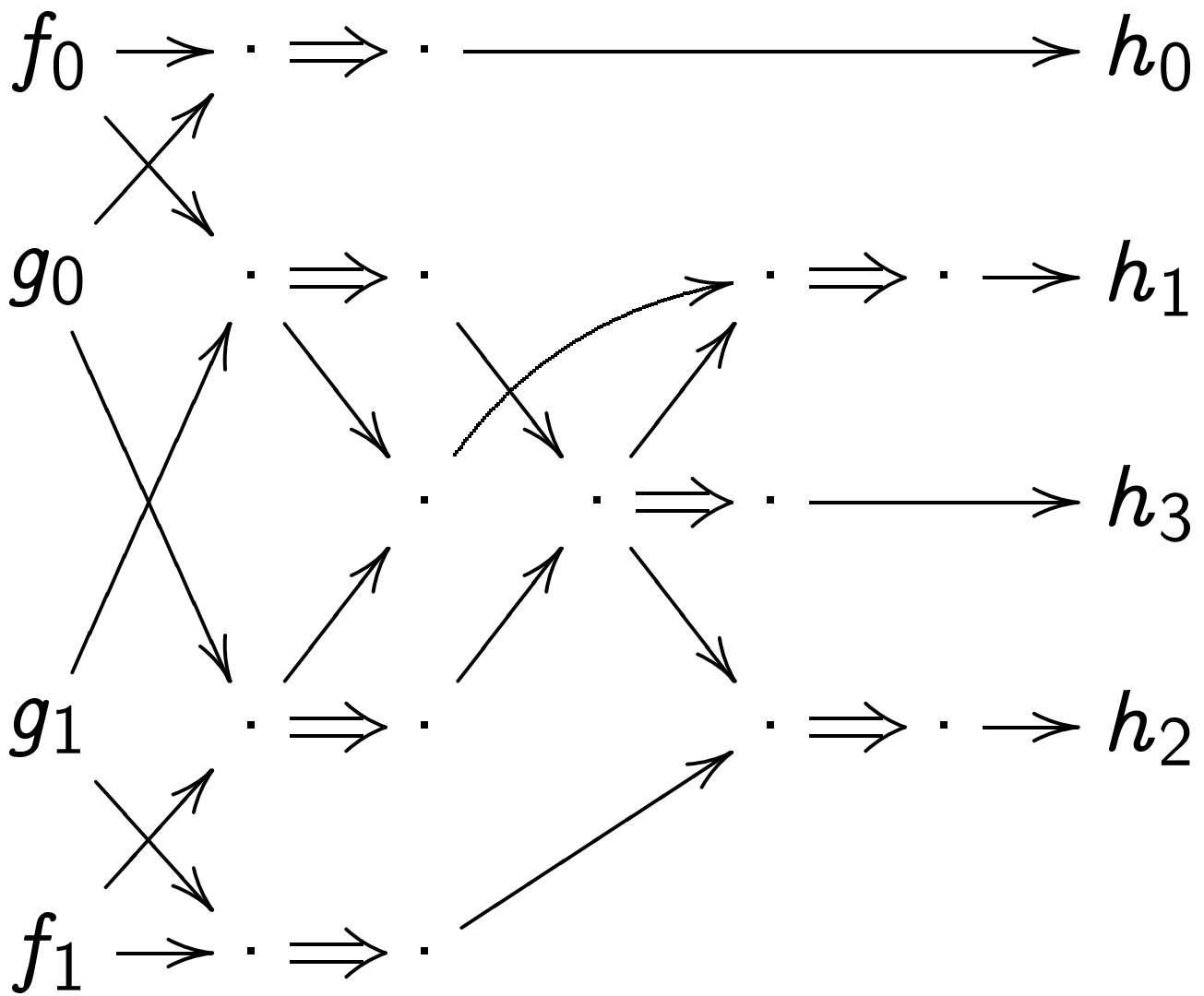
Three expositions of  
how the tangent FFT works:

Frigo/Johnson,  
in *IEEE Trans. Signal Processing*.

Lundy/Van Buskirk,  
in *Computing*.

Bernstein, <http://cr.yp.to/talks.html#2007.04.17>,  
expanding an old idea of Fiduccia.

# Bit complexity



Each  $\cdot$  is a “gate”  $a, b \mapsto 1 - ab$ .

This “circuit” computes

product  $h_0 + 2h_1 + 4h_2 + 8h_3$

of  $f_0 + 2f_1, g_0 + 2g_1 \in \mathbf{Z}$ .

“Bit complexity” 15: 15 gates.

## Integer-multiplication news

How quickly can we multiply  
in the ring  $\mathbf{Z}$ ?

1971 Schönhage/Strassen:

circuit to compute  $f, g \mapsto fg$

when  $f, g \in \{0, 1, \dots, 2^n - 1\}$ ;

bit complexity  $\Theta(n \lg n \lg \lg n)$ .

2007 Fürer: Can do better!

Bit complexity  $(n \lg n) 2^{O(\lg^* n)}$ .

$\lg^* n = 1$  if  $1 \leq \lg n < 2$ ;

$\lg^* n = 2$  if  $1 \leq \lg \lg n < 2$ ;

$\lg^* n = 3$  if  $1 \leq \lg \lg \lg n < 2$ ;

$\lg^* n = 4$  if  $1 \leq \lg \lg \lg \lg n < 2$ ;

etc.

Bit complexity  $n(\lg n)^{O(1)}$ :

View  $n$ -bit integers as

$\Theta(n/\lg n)$ -coeff polynomials

over a prime field  $k$  with

$\lg \#k \in \Theta(\lg n)$ , appropriate  $\zeta$ .

(Or  $k = \mathbf{R}$ , approximating coeffs.)

Bottleneck: FFT has many

mults by constants in  $k$ .

Schönhage/Strassen idea:

View  $n$ -bit integers as

$\Theta(\sqrt{n})$ -coeff polynomials

over the ring  $\mathbf{Z}/(2^{\Theta(\sqrt{n})} + 1)$ .

Mults by  $2^{\dots}$  are easy in this ring.

$\Theta(\lg \lg n)$  levels of recursion.

Fürer idea:

View  $n$ -bit integers as

$\Theta(n/(\lg n)^2)$ -coeff polynomials  
over the ring  $k[y]/(y^{\Theta(\lg n)} + 1)$

with  $\lg \#k \in \Theta(\lg n)$ ,

appropriate  $\zeta$ . (Or  $k = \mathbf{R}$ .)

$\Theta(n/\lg n)$  mults in ring.

Align roots in FFT.

(Recall split-radix speedup!)

Then most mults are easy,

as in Schönhage-Strassen.

$\Theta(n/(\lg n)^2)$  hard mults; ok.

$\Theta(\lg^* n)$  levels of recursion.