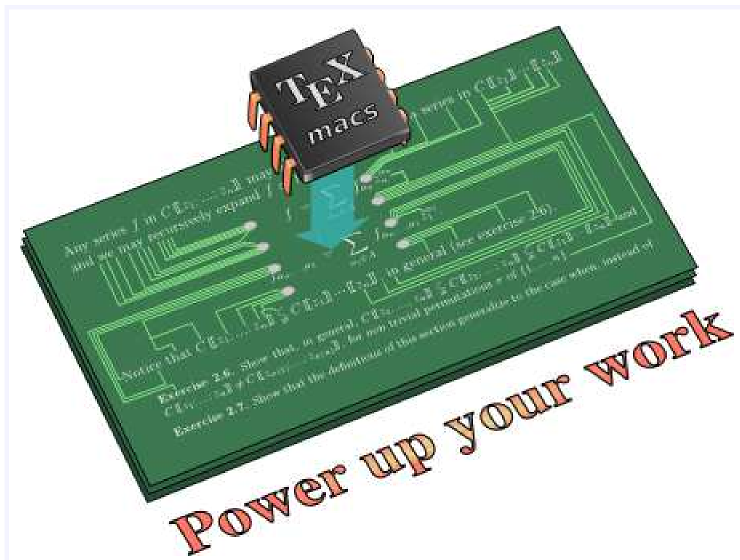
Fast integer multiplication

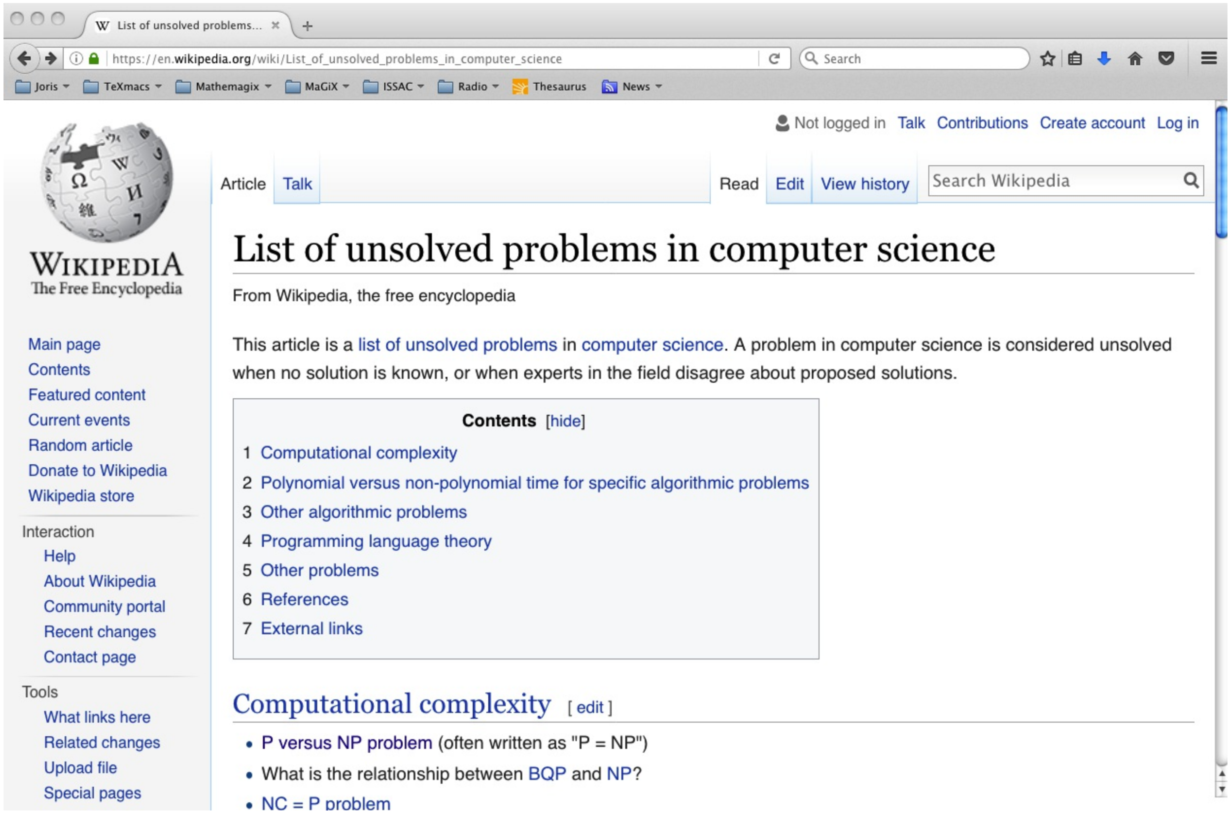
David Harvey, **Joris van der Hoeven**, Grégoire Lecerf

CNRS, École polytechnique



LIX, December 15, 2016

http://www.TeXmacs.org


List of unsolved problems...
https://en.wikipedia.org/wiki/List_of_unsolved_problems_in_computer_science
Search
Joris
TeXmacs
Mathemagix
MaGiX
ISSAC
Radio
Thesaurus
News
Not logged in
Talk
Contributions
Create account
Log in
Article
Talk
Read
Edit
View history
Search Wikipedia
WIKIPEDIA
The Free Encyclopedia
Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store
Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page
Tools
What links here
Related changes
Upload file
Special pages
List of unsolved problems in computer science
From Wikipedia, the free encyclopedia
This article is a list of unsolved problems in computer science. A problem in computer science is considered unsolved when no solution is known, or when experts in the field disagree about proposed solutions.
Contents [hide]
1 Computational complexity
2 Polynomial versus non-polynomial time for specific algorithmic problems
3 Other algorithmic problems
4 Programming language theory
5 Other problems
6 References
7 External links
Computational complexity [edit]
• P versus NP problem (often written as "P = NP")
• What is the relationship between BQP and NP?
• NC = P problem

Computational complexity

- P versus NP problem (often written as “P = NP”)
- What is the relationship between BQP and NP?
- NC = P problem ...

Polynomial vs non-polynomial time for specific algorithmic problems

- Can integer factorization be done in polynomial time?
- Is integer factorization NP-complete?
- Can clustered planar drawings be found in polynomial time? ...

Other algorithmic problems

- What is the fastest algorithm for multiplication of two n-digit numbers?
- What is the fastest algorithm for matrix multiplication?
- Can the Schwartz–Zippel lemma for polynomial identity testing be derandomized?
- Can a depth-first search tree be constructed in NC?
- Does linear programming admit a strongly polynomial-time algorithm?
- What is the lower bound on the complexity of fast Fourier transform algorithms? ...

Programming language theory ...

Other problems ...

Sequential *vs.* parallel

We will consider sequential algorithms

Sequential *vs.* parallel

We will consider sequential algorithms

Turing machines

Turing machines with a finite number of tapes [Papadimitriou 94]

Sequential *vs.* parallel

We will consider sequential algorithms

Turing machines

Turing machines with a finite number of tapes [Papadimitriou 94]

Other bit complexity models

- Operations on $\log n$-bit numbers in time $O(1)$
- Random access machine (RAM)

Sequential *vs.* parallel

We will consider sequential algorithms

Turing machines

Turing machines with a finite number of tapes [Papadimitriou 94]

Other bit complexity models

- Operations on $\log n$-bit numbers in time $O(1)$
- Random access machine (RAM)

« Straight Line Programs » (SLPs)

DAGs, non branching programs [Bürgisser–Clausen–Shokrollahi 97]

Sequential *vs.* parallel

We will consider sequential algorithms

Turing machines

Turing machines with a finite number of tapes [Papadimitriou 94]

Other bit complexity models

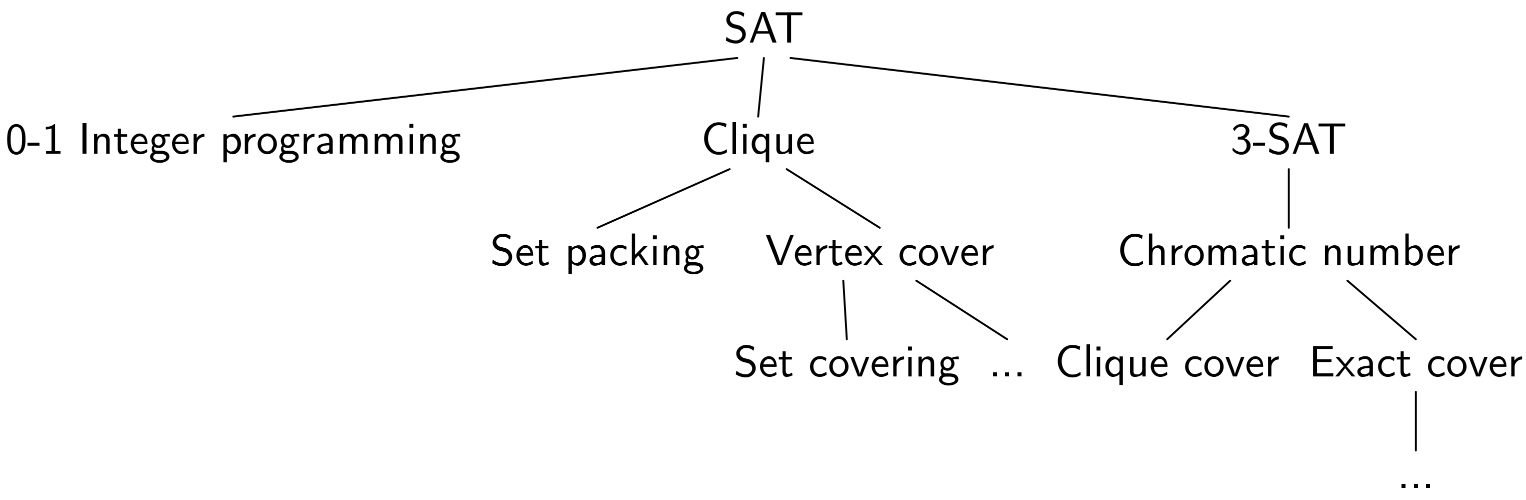
- Operations on $\log n$-bit numbers in time $O(1)$
- Random access machine (RAM)

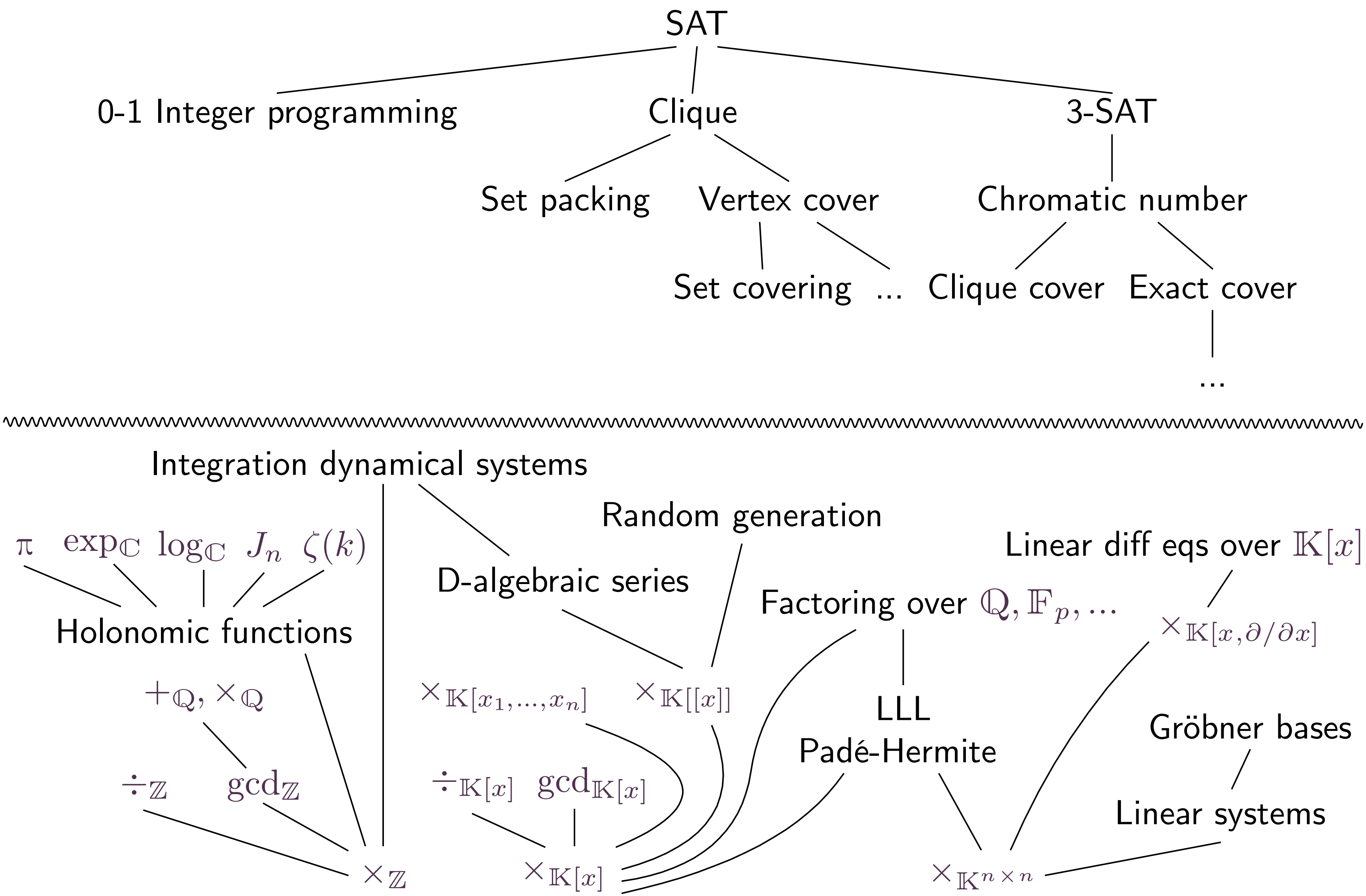
« Straight Line Programs » (SLPs)

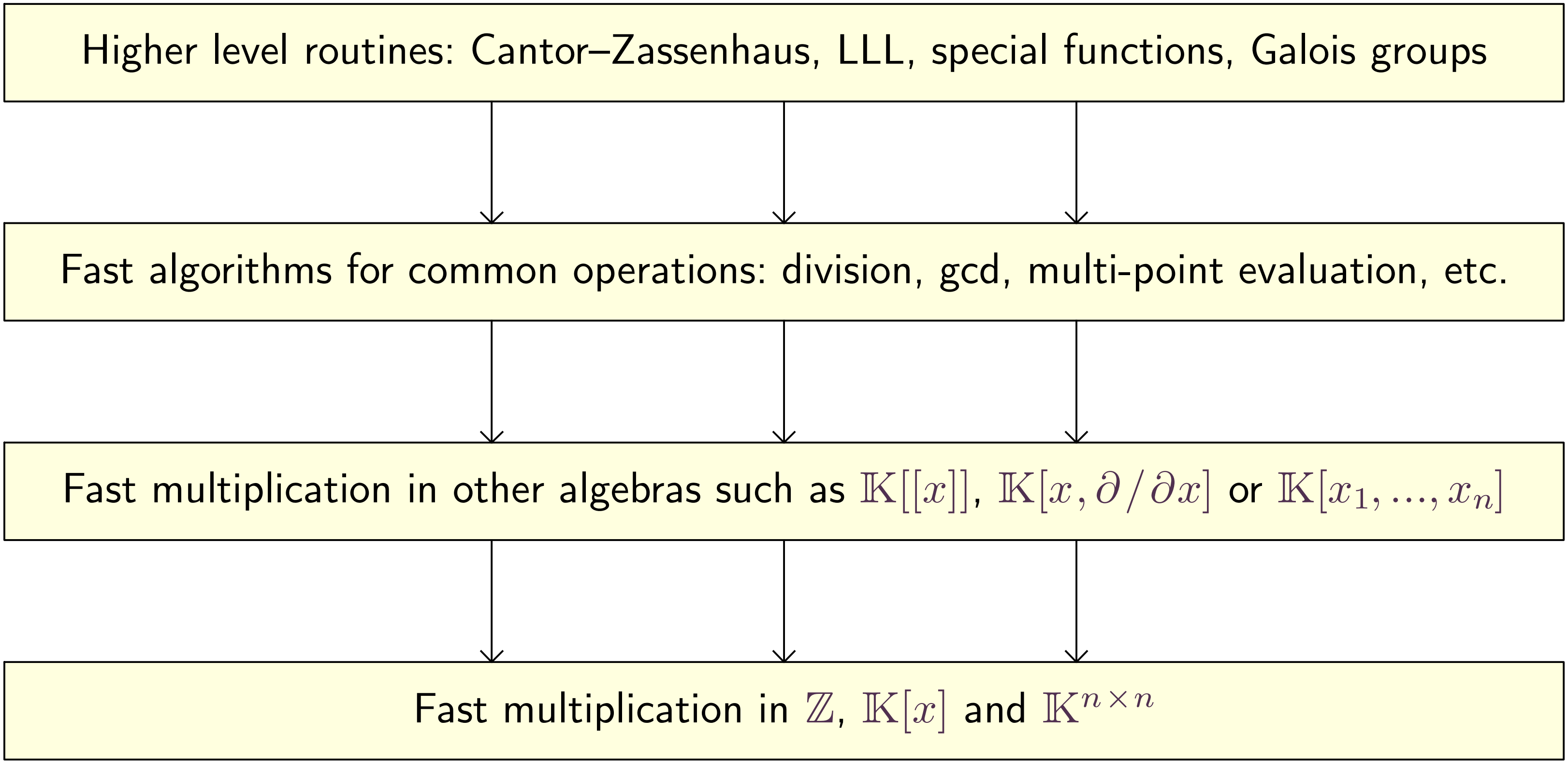
DAGs, non branching programs [Bürgisser–Clausen–Shokrollahi 97]

Other algebraic complexity models

- Turing machines with entries in model-theoretic structures $\mathfrak{S}$ [Friedman 69]
- BSS machines [Blum–Shub–Smale 89]


SAT
0-1 Integer programming
Clique
3-SAT
Set packing
Vertex cover
Chromatic number
Set covering
...
Clique cover
Exact cover
...


SAT
0-1 Integer programming
Clique
3-SAT
Set packing
Vertex cover
Chromatic number
Set covering
...
Clique cover
Exact cover
...
Integration dynamical systems
Random generation
π $\exp_{\mathbb{C}}$ $\log_{\mathbb{C}}$ J_n $\zeta(k)$
D-algebraic series
Linear diff eqs over $\mathbb{K}[x]$
Factoring over $\mathbb{Q}, \mathbb{F}_p, \ldots$
Holonomic functions
$\times_{\mathbb{K}[x,\partial/\partial x]}$
$+_{\mathbb{Q}}, \times_{\mathbb{Q}}$
$\times_{\mathbb{K}[x_1,\ldots,x_n]}$
$\times_{\mathbb{K}[[x]]}$
LLL
Gröbner bases
Padé-Hermite
$\div_{\mathbb{Z}}$
$\gcd_{\mathbb{Z}}$
$\div_{\mathbb{K}[x]}$
$\gcd_{\mathbb{K}[x]}$
Linear systems
$\times_{\mathbb{Z}}$
$\times_{\mathbb{K}[x]}$
$\times_{\mathbb{K}^{n\times n}}$


Higher level routines: Cantor–Zassenhaus, LLL, special functions, Galois groups
Fast algorithms for common operations: division, gcd, multi-point evaluation, etc.
Fast multiplication in other algebras such as $\mathbb{K}[[x]]$, $\mathbb{K}[x, \partial/\partial x]$ or $\mathbb{K}[x_1, \ldots, x_n]$
Fast multiplication in $\mathbb{Z}$, $\mathbb{K}[x]$ and $\mathbb{K}^{n \times n}$

Algebra	Model	Current record	Author(s)
$\mathbb{Z}$	binary	$\mathsf{I}(n) = O(n \log n\, 8^{\log^* n})$	Harvey-vdH-Lecerf 2014
$\mathbb{K}[z]$	algebraic	$\mathsf{M}(d) = O(d \log d \log \log d)$	Cantor-Kaltofen 1981
$\mathbb{F}_p[z]$	binary	$O(t \log t\, 8^{\log^* t})$ $t = d \log p$	Harvey-vdH-Lecerf 2014
$\mathbb{K}[z]$	algebraic char. $p > 0$	$\mathsf{M}(d) = O(d \log d\, 8^{\log^* d})$	Harvey-vdH-Lecerf 2014
$\mathbb{K}^{r \times r}$	algebraic	$O(n^\omega)$, $\omega \leqslant 2.3728639$	Le Gall 2014

$$\begin{aligned} \log^* x &:= \min \{k \in \mathbb{N} : \log^{\circ k} x \leqslant 1\}, \\ \log^{\circ k} &:= \underbrace{\log \circ \cdots \circ \log}_{k \times}. \end{aligned}$$

Algebra	Model	Current record	Author(s)
$\mathbb{K}[x]^{r \times r}$	algebraic	$O(\mathsf{M}(d)\, r^2 + d\, r^{\omega})$	Bostan-Schost 2005
$\mathbb{Z}^{r \times r}$	binary	$O\left(r^2\, \mathsf{M}(n) + r^{\omega} n 2^{\log^* n - \log^* d} \frac{\mathsf{M}(\log r)}{\log r}\right)$	Harvey-vdH 2014
$\mathbb{K}[[z]]$	algebraic relaxed	$O(d \log d\, \mathrm{e}^{\sqrt{2 \log 2} \sqrt{\log \log d}}\, X)$ $X = (\log \log d)^{5/2} \log \log \log d$	vdH 2014
$\mathbb{Z}_p$	binary relaxed	$O(d \log d\, \mathrm{e}^{\sqrt{2 \log 2} \sqrt{\log \log d}}\, X)$ $X = ((\log \log d)^{3/2} \log p)\, \ell \log \ell$ $\ell = \log (\log d + \log p)$	vdH 2014 Bertomieu-vdH-Lecerf 2011
$\mathbb{K}[z_1, \ldots, z_k]$	algebraic dense	$O(\mathsf{M}(s))$	vdH-Schost 2010
$\mathbb{Z}[z_1, \ldots, z_k]$	binary sparse	$O(\mathsf{I}(s\, \nu) \log s + s\, \mathsf{I}(\nu) \log \nu)$ $\nu = n + k \log d$	vdH-Lecerf 2009
$\mathbb{K}[z, \partial]$	algebraic	$O(r^{\omega - 1} d + r\, \mathsf{M}(d) \log d)$ si $d \geqslant r$ $O(d^{\omega - 1} r + d\, \mathsf{M}(r) \log r)$ si $r \geqslant d$	Benoit-Bostan-vdH 2012

Operation	Model	Current record	Author(s)
Euclidean division in $\mathbb{Z}$	binary	$\lesssim {}^5\!/_3\, \mathsf{I}(n)$	vdH 2010*
Square root in $\mathbb{Z}$	binary	$\lesssim {}^4\!/_3\, \mathsf{I}(n)$	Harvey 2009* vdH
G.c.d. in $\mathbb{Z}$	binary	$O(\mathsf{I}(n) \log n)$	Schönhage 1971 Lehmer
Chinese remaindering	binary	$O(\mathsf{I}(n\, d) \log d)$	Borodin-Moenck 1972
Chinese remaindering Fixed moduli	binary	$O\left(\mathsf{I}(n\, d) \frac{\log d}{\log \log (n\, d)}\right)$	vdH 2016

* Adapted from power series analogue

Date	Authors	Complexity
<3000 aJC	Unknown	$O(n^2)$
1962	Karatsuba	$O(n^{\log 3/\log 2})$
1963 (1965)	Toom (Cook)	$O(n\,2^{5\sqrt{\log n/\log 2}})$
1966	Schönhage	$O(n\,2^{\sqrt{2\log n/\log 2}}\,(\log n)^{3/2})$
1969	Knuth	$O(n\,2^{\sqrt{2\log n/\log 2}}\,\log n)$
1971	Schönhage–Strassen	$O(n\log n\log\log n)$
2007	Fürer	$O(n\log n\,2^{O(\log^* n)})$
2014	Harvey–vdH–Lecerf	$O(n\log n\,8^{\log^* n})$

From integers to polynomials and back

$$\begin{aligned} 971362651726262537182735 &= 971362\,X^3 + 651726\,X^2 + 262537\,X + 182735 \\ X &= 1000000 \end{aligned}$$

$$p = 127622142187 \qquad q = 209836129877$$

$$P = 127622\,X + 142187 \qquad Q = 209836\,X + 129877$$

$Q_1 = 209836$	$P_0\,Q_1$	$P_1\,Q_1$
$Q_0 = 129877$	$P_0\,Q_0$	$P_1\,Q_0$
	$P_0 = 142187$	$P_1 = 127622$

$$PQ = P_0\,Q_0 + (P_0\,Q_1 + P_1\,Q_0)\,X + P_1\,Q_1\,X^2$$

$$p = 127622142187 \qquad q = 209836129877$$

$$P = 127622\,X + 142187 \qquad Q = 209836\,X + 129877$$

$Q_1 = 209836$	$P_0\,Q_1$	$P_1\,Q_1$
$Q_0 = 129877$	$P_0\,Q_0$	$P_1\,Q_0$
	$P_0 = 142187$	$P_1 = 127622$

$$P_0\,Q_0 \;=\; 18466820999$$

$$PQ = P_0\,Q_0 + (P_0\,Q_1 + P_1\,Q_0)\,X + P_1\,Q_1\,X^2$$

$$p = 127622142187 \qquad q = 209836129877$$

$$P = 127622\,X + 142187 \qquad Q = 209836\,X + 129877$$

$Q_1 = 209836$	$P_0\,Q_1$	$P_1\,Q_1$
$Q_0 = 129877$	$P_0\,Q_0$	$P_1\,Q_0$
	$P_0 = 142187$	$P_1 = 127622$

$$P_0\,Q_0 = 18466820999$$
$$P_1\,Q_1 = 26779689992$$

$$PQ = P_0\,Q_0 + (P_0\,Q_1 + P_1\,Q_0)\,X + P_1\,Q_1\,X^2$$

$$p = 127622142187 \qquad q = 209836129877$$

$$P = 127622\,X + 142187 \qquad Q = 209836\,X + 129877$$

$Q_1 = 209836$	$P_0\,Q_1$	$P_1\,Q_1$
$Q_0 = 129877$	$P_0\,Q_0$	$P_1\,Q_0$
	$P_0 = 142187$	$P_1 = 127622$

$$P_0\,Q_0 = 18466820999$$
$$P_1\,Q_1 = 26779689992$$
$$(P_0 + P_1)\,(Q_0 + Q_1) = 91657624817$$

$$PQ = P_0\,Q_0 + (P_0\,Q_1 + P_1\,Q_0)\,X + P_1\,Q_1\,X^2$$

$$p = 127622142187 \qquad q = 209836129877$$

$$P = 127622\,X + 142187 \qquad Q = 209836\,X + 129877$$

$Q_1 = 209836$	$P_0\,Q_1$	$P_1\,Q_1$
$Q_0 = 129877$	$P_0\,Q_0$	$P_1\,Q_0$
	$P_0 = 142187$	$P_1 = 127622$

$$P_0\,Q_0 = 18466820999$$
$$P_1\,Q_1 = 26779689992$$
$$A = (P_0 + P_1)\,(Q_0 + Q_1) = 91657624817$$

$$A - P_0\,Q_0$$

$$PQ = P_0\,Q_0 + (P_0\,Q_1 + P_1\,Q_0)\,X + P_1\,Q_1\,X^2$$

$$p = 127622142187 \qquad q = 209836129877$$

$$P = 127622\,X + 142187 \qquad Q = 209836\,X + 129877$$

$Q_1 = 209836$	$P_0\,Q_1$	$P_1\,Q_1$
$Q_0 = 129877$	$P_0\,Q_0$	$P_1\,Q_0$
	$P_0 = 142187$	$P_1 = 127622$

$$\begin{aligned} P_0\,Q_0 &= 18466820999 \\ P_1\,Q_1 &= 26779689992 \\ A = (P_0 + P_1)\,(Q_0 + Q_1) &= 91657624817 \end{aligned}$$

$$A - P_0\,Q_0 - P_1\,Q_1$$

$$PQ = P_0\,Q_0 + (P_0\,Q_1 + P_1\,Q_0)\,X + P_1\,Q_1\,X^2$$

$p = 127622142187 \qquad q = 209836129877$

$P = 127622\,X + 142187 \qquad Q = 209836\,X + 129877$

$Q_1 = 209836$	$P_0\,Q_1$	$P_1\,Q_1$
$Q_0 = 129877$	$P_0\,Q_0$	$P_1\,Q_0$
	$P_0 = 142187$	$P_1 = 127622$

$$\begin{aligned} P_0\,Q_0 &= 18466820999 \\ P_1\,Q_1 &= 26779689992 \\ A = (P_0 + P_1)\,(Q_0 + Q_1) &= 91657624817 \\ A - P_0\,Q_0 - P_1\,Q_1 &= P_0\,Q_1 + P_1\,Q_0 \\ &= 46411113826 \end{aligned}$$

$$PQ = P_0\,Q_0 + (P_0\,Q_1 + P_1\,Q_0)\,X + P_1\,Q_1\,X^2$$

$p = 127622142187 \qquad q = 209836129877$

$P = 127622\,X + 142187 \qquad Q = 209836\,X + 129877$

$Q_1 = 209836$	$P_0\,Q_1$	$P_1\,Q_1$
$Q_0 = 129877$	$P_0\,Q_0$	$P_1\,Q_0$
	$P_0 = 142187$	$P_1 = 127622$

$$\begin{aligned} P_0\,Q_0 &= 18466820999 \\ P_1\,Q_1 &= 26779689992 \\ A = (P_0 + P_1)\,(Q_0 + Q_1) &= 91657624817 \end{aligned}$$

$$\begin{aligned} A - P_0\,Q_0 - P_1\,Q_1 &= P_0\,Q_1 + P_1\,Q_0 \\ &= 46411113826 \end{aligned}$$

$$PQ = 26779689992 + 46411113826\,X + 18466820999\,X^2$$

$$p = 127622142187 \qquad q = 209836129877$$

$$P = 127622\,X + 142187 \qquad Q = 209836\,X + 129877$$

$Q_1 = 209836$	$P_0\,Q_1$	$P_1\,Q_1$
$Q_0 = 129877$	$P_0\,Q_0$	$P_1\,Q_0$
	$P_0 = 142187$	$P_1 = 127622$

$$\begin{aligned} P_0\,Q_0 &= 18466820999 \\ P_1\,Q_1 &= 26779689992 \\ A = (P_0 + P_1)\,(Q_0 + Q_1) &= 91657624817 \end{aligned}$$

$$\begin{aligned} A - P_0\,Q_0 - P_1\,Q_1 &= P_0\,Q_1 + P_1\,Q_0 \\ &= 46411113826 \end{aligned}$$

$$P\,Q = 26779689992 + 46411113826\,X + 18466820999\,X^2$$

$$p\,q = \begin{array}{r} 26779689992000000000000 \\ 00000046411113826000000 \\ 00000000000018466820999 \\ \hline 26779736403132292820999 \end{array}$$

Karatsuba

$$P = 127622\,X + 142187$$
$$Q = 209836\,X + 129877$$

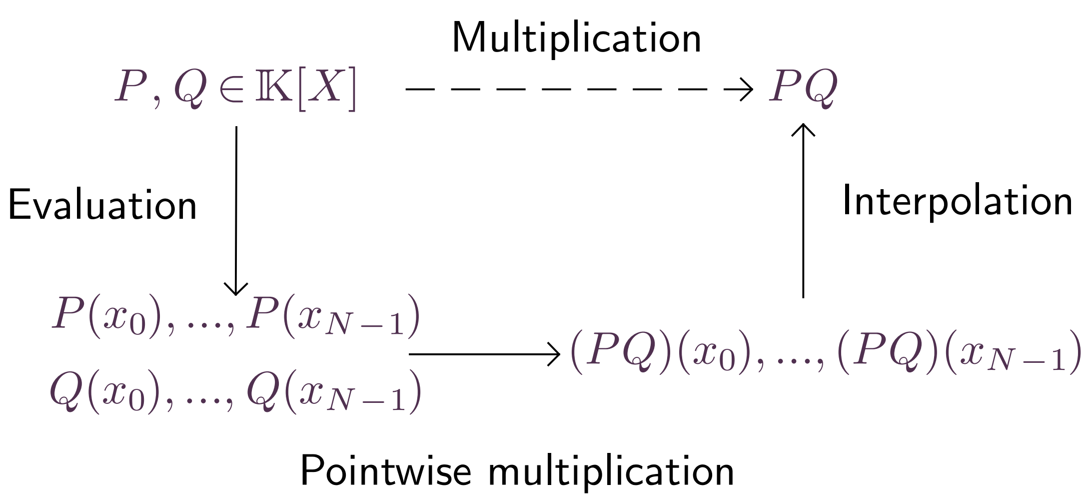
$$P_0\,Q_0 = (PQ)(0) = P(0)\,Q(0)$$
$$P_1\,Q_1\,\infty^2 = (PQ)(\infty) = P(\infty)\,Q(\infty)$$
$$(P_0 + P_1)\,(Q_0 + Q_1) = (PQ)(1) = P(1)\,Q(1)$$

Karatsuba

$$
\begin{aligned}
P &= 127622\,X + 142187\\
Q &= 209836\,X + 129877
\end{aligned}
$$

$$
\begin{aligned}
P_0\,Q_0 &= (PQ)(0) = P(0)\,Q(0)\\
P_1\,Q_1\,\infty^2 &= (PQ)(\infty) = P(\infty)\,Q(\infty)\\
(P_0 + P_1)\,(Q_0 + Q_1) &= (PQ)(1) = P(1)\,Q(1)
\end{aligned}
$$

Evaluation-interpolation



Discrete Fourier transforms

Assume that $\omega \in \mathbb{K}$ is such that $\omega^N = 1$, $N \in 2^{\mathbb{N}}$ and $1, \omega, \omega^2, \ldots, \omega^{N-1}$ all distinct.

$$\mathrm{DFT}_\omega(P_0, \ldots, P_{N-1}) \;=\; (P(1), P(\omega), P(\omega^2), \ldots, P(\omega^{N-1}))$$

Corresponds to evaluating $P = P_0 + \cdots + P_{N-1} X^{N-1}$ at $x_i = \omega^i$

Discrete Fourier transforms

Assume that $\omega \in \mathbb{K}$ is such that $\omega^N = 1$, $N \in 2^{\mathbb{N}}$ and $1, \omega, \omega^2, \ldots, \omega^{N-1}$ all distinct.

$$\mathrm{DFT}_\omega(P_0, \ldots, P_{N-1}) = (P(1), P(\omega), P(\omega^2), \ldots, P(\omega^{N-1}))$$

Corresponds to evaluating $P = P_0 + \cdots + P_{N-1} X^{N-1}$ at $x_i = \omega^i$

Cyclic polynomials

Elements of $\mathbb{K}[x] / (X^N - 1)$. We have

$$X^N - 1 = (X - 1)(X - \omega) \cdots (X - \omega^{N-1})$$

Discrete Fourier transforms

Assume that $\omega \in \mathbb{K}$ is such that $\omega^N = 1$, $N \in 2^{\mathbb{N}}$ and $1, \omega, \omega^2, \ldots, \omega^{N-1}$ all distinct.

$$\mathrm{DFT}_\omega(P_0, \ldots, P_{N-1}) = (P(1), P(\omega), P(\omega^2), \ldots, P(\omega^{N-1}))$$

Corresponds to evaluating $P = P_0 + \cdots + P_{N-1} X^{N-1}$ at $x_i = \omega^i$

Cyclic polynomials

Elements of $\mathbb{K}[x]/(X^N - 1)$. We have

$$\begin{aligned} X^N - 1 &= (X-1)(X-\omega)\cdots(X-\omega^{N-1}) \\ \mathbb{K}[X]/(X^N-1) &\cong K[X]/(X-1) \oplus K[X]/(X-\omega) \oplus \cdots \oplus K[X]/(X-\omega^{N-1}) \end{aligned}$$

Discrete Fourier transforms

Assume that $\omega \in \mathbb{K}$ is such that $\omega^N = 1$, $N \in 2^{\mathbb{N}}$ and $1, \omega, \omega^2, \ldots, \omega^{N-1}$ all distinct.

$$\mathrm{DFT}_\omega(P_0, \ldots, P_{N-1}) = (P(1), P(\omega), P(\omega^2), \ldots, P(\omega^{N-1}))$$

Corresponds to evaluating $P = P_0 + \cdots + P_{N-1} X^{N-1}$ at $x_i = \omega^i$

Cyclic polynomials

Elements of $\mathbb{K}[x]/(X^N - 1)$. We have

$$\begin{aligned} X^N - 1 &= (X-1)(X-\omega)\cdots(X-\omega^{N-1}) \\ \mathbb{K}[X]/(X^N-1) &\cong K[X]/(X-1) \oplus K[X]/(X-\omega) \oplus \cdots \oplus K[X]/(X-\omega^{N-1}) \\ P &\rightsquigarrow (P(1), P(\omega), \ldots, P(\omega^{N-1})) \end{aligned}$$

Discrete Fourier transforms

Assume that $\omega \in \mathbb{K}$ is such that $\omega^N = 1$, $N \in 2^{\mathbb{N}}$ and $1, \omega, \omega^2, \ldots, \omega^{N-1}$ all distinct.

$$\mathrm{DFT}_\omega(P_0, \ldots, P_{N-1}) = (P(1), P(\omega), P(\omega^2), \ldots, P(\omega^{N-1}))$$

Corresponds to evaluating $P = P_0 + \cdots + P_{N-1} X^{N-1}$ at $x_i = \omega^i$

Cyclic polynomials

Elements of $\mathbb{K}[x] / (X^N - 1)$. We have

$$\begin{aligned} X^N - 1 &= (X-1)(X-\omega)\cdots(X-\omega^{N-1}) \\ \mathbb{K}[X]/(X^N-1) &\cong K[X]/(X-1) \oplus K[X]/(X-\omega) \oplus \cdots \oplus K[X]/(X-\omega^{N-1}) \\ P &\leftrightsquigarrow (P(1), P(\omega), \ldots, P(\omega^{N-1})) \end{aligned}$$

Cyclic convolution $\rightsquigarrow$ DFT

Product of $P, Q \in \mathbb{K}[X] / (X^N - 1)$ also called cyclic convolution

$$((PQ)_0, \ldots, (PQ)_{n-1}) = \mathrm{DFT}_\omega^{-1}(\mathrm{DFT}_\omega(P_0, \ldots, P_{n-1})\, \mathrm{DFT}_\omega(Q_0, \ldots, Q_{n-1}))$$

DFT $\rightsquigarrow$ Cyclic convolution [Bluestein 70]

Assume $\eta \in \mathbb{K}$ given with $\eta^2 = \omega$.

$$f_i := \eta^{i^2}, \qquad g_i := \eta^{-i^2}$$

$$f_{i+n} = \eta^{(i+n)^2} = \eta^{i^2+n^2+2ni} = \eta^{i^2} \omega^{(\frac{n}{2}+i)n} = f_i, \qquad g_{i+n} = g_i$$

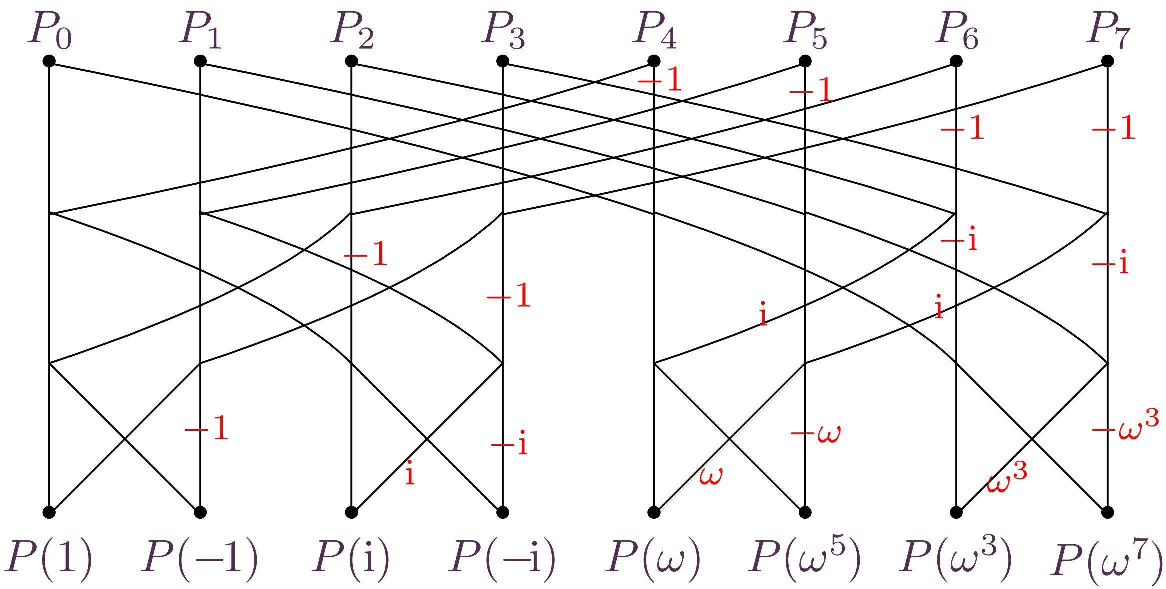
Then $\omega^{ij} = f_i f_j g_{i-j}$, so for all $a \in \mathbb{K}^n$:

$$\hat{a}_i = \mathrm{DFT}_\omega(a)_i = \sum_{j=0}^{n-1} a_j \omega^{ij} = f_i \sum_{j=0}^{n-1} (a_j f_j) \, g_{i-j}$$

One recognizes a cyclic convolution

Cooley–Tuckey method (essentially known to Gauß)

$$
\begin{aligned}
P(X) &= A(X^2) + B(X^2)\,X \\
P(\omega) &= A(\omega^2) + B(\omega^2)\,X \\
(P(1), P(\omega), \ldots, P(\omega^{N-1})) &= \Big(A(1)+B(1), \ldots, A((\omega^2)^{\frac{N}{2}-1}) + B((\omega^2)^{\frac{N}{2}-1})\,\omega^{\frac{N}{2}-1}, \\
&\quad A(1)-B(1), \ldots, A((\omega^2)^{\frac{N}{2}-1}) - B((\omega^2)^{\frac{N}{2}-1})\,\omega^{\frac{N}{2}-1} \Big)
\end{aligned}
$$



Direct transform

$$\mathrm{DFT}_\omega(P_0, \ldots, P_{N-1}) \;=\; (P(1), P(\omega), P(\omega^2), \ldots, P(\omega^{N-1}))$$

Direct transform

$$\mathrm{DFT}_\omega(P_0, \ldots, P_{N-1}) = (P(1), P(\omega), P(\omega^2), \ldots, P(\omega^{N-1}))$$

Inverse transform

$$\mathrm{DFT}_\omega^{-1} = \frac{1}{N}\mathrm{DFT}_{\omega^{-1}}$$

Interpolation $\leftrightsquigarrow$ Evaluation

Direct transform

$$\mathrm{DFT}_\omega(P_0, \ldots, P_{N-1}) \;=\; (P(1), P(\omega), P(\omega^2), \ldots, P(\omega^{N-1}))$$

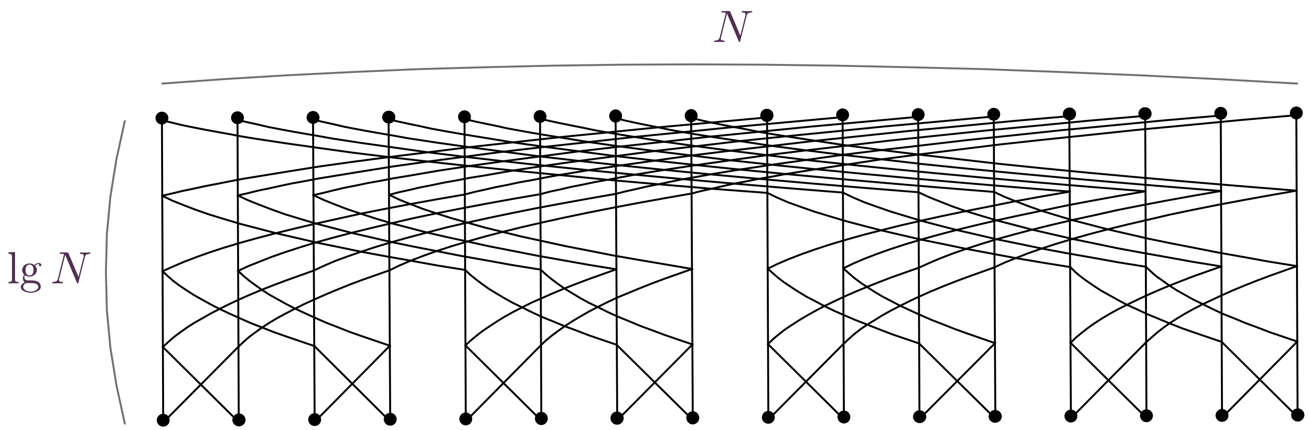
Inverse transform

$$\mathrm{DFT}_\omega^{-1} \;=\; \tfrac{1}{N}\,\mathrm{DFT}_{\omega^{-1}}$$

Interpolation $\leftrightsquigarrow$ Evaluation

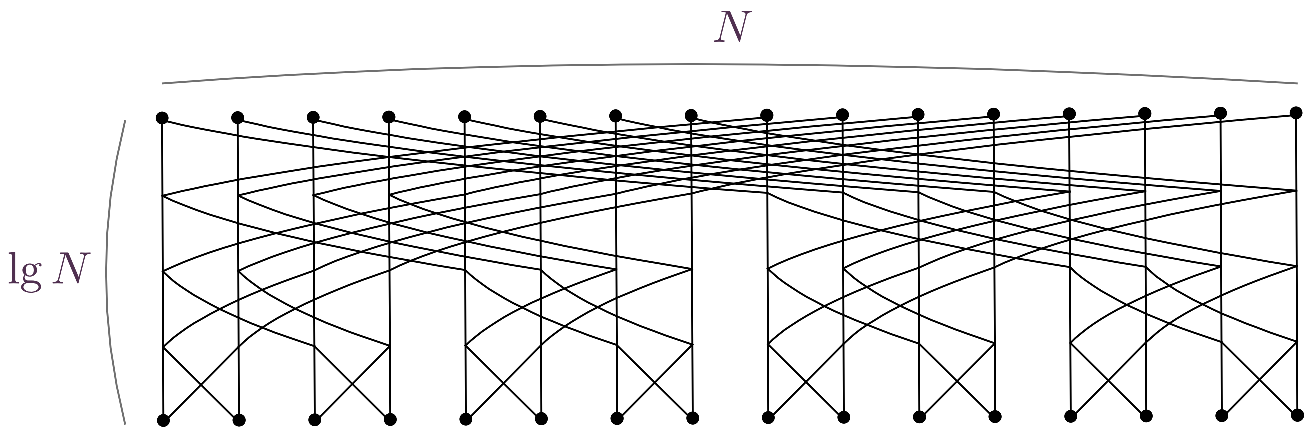
Variants

- $\mathbb{K} = \mathbb{C}_b$: **complex DFT**, complex fixed-point arithmetic with b-bit precision
- $\mathbb{K} = \mathbb{F}_p$, **modular DFT**, with p prime number of the form $k\,2^N \pm 1$ [Pollard 71]
- $\mathbb{K} = \mathbb{L}[Y]/(Y^{2^N} \pm 1)$, **synthetic DFT**, *à la* Schönhage–Strassen



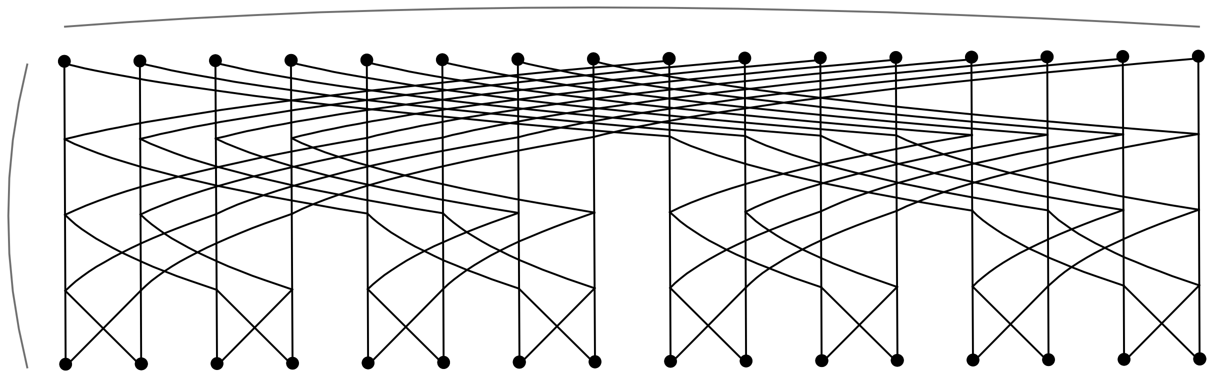
Cost of one DFT : $\frac{1}{2} N \lg N$ "butterflies" $\rightsquigarrow O(N \lg N)$ operations in $\mathbb{K}$

Complex DFT	$N \asymp n/\lg n$	$\mathsf{M}_{\mathbb{K}}(1) = O(\mathsf{I}(\lg n))$	$\mathsf{I}(n) = O(n \lg n\, \mathsf{I}(\lg n) + n \lg n)$
Modular DFT	$N \asymp n/\lg n$	$\mathsf{M}_{\mathbb{K}}(1) = O(\mathsf{I}(\lg n))$	$\mathsf{I}(n) = O(n \lg n\, \mathsf{I}(\lg n) + n \lg n)$
Synthetic DFT	$N \asymp \sqrt{n}$	butterfly $\rightsquigarrow O(\sqrt{n})$	$\mathsf{I}(n) = O(n \lg n \sqrt{n} + \sqrt{n}\, \mathsf{I}(\sqrt{n}))$



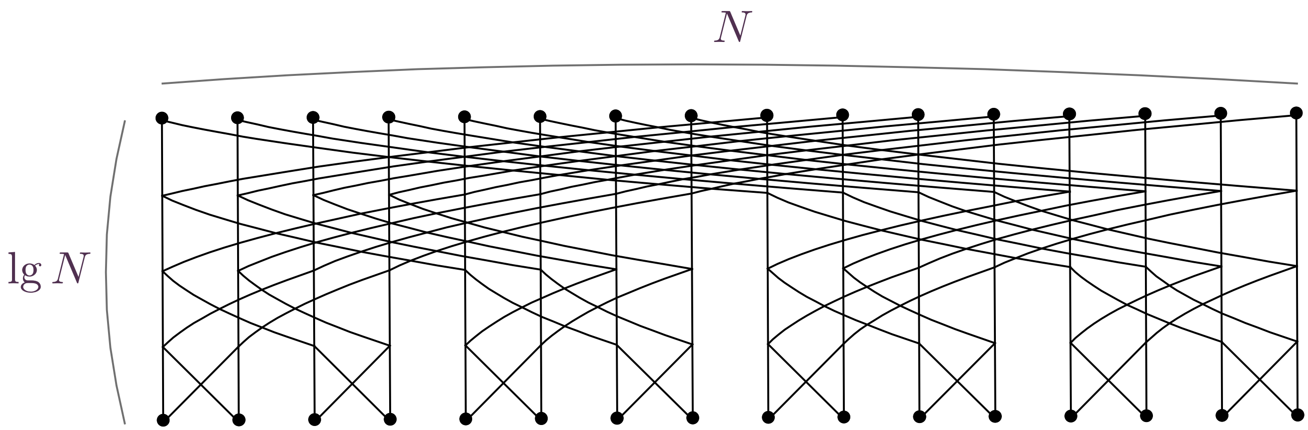
Cost of one DFT : $\frac{1}{2} N \lg N$ "butterflies" $\rightsquigarrow O(N \lg N)$ operations in $\mathbb{K}$

Complex DFT	$N \asymp n/\lg n$	$\mathsf{M}_{\mathbb{K}}(1) = O(\mathsf{I}(\lg n))$	$\mathsf{I}(n) = O(n \lg n\, \mathsf{I}(\lg n) + n \lg n)$
Modular DFT	$N \asymp n/\lg n$	$\mathsf{M}_{\mathbb{K}}(1) = O(\mathsf{I}(\lg n))$	$\mathsf{I}(n) = O(n \lg n\, \mathsf{I}(\lg n) + n \lg n)$
Synthetic DFT	$N \asymp \sqrt{n}$	butterfly $\rightsquigarrow O(\sqrt{n})$	$\mathsf{I}(n) = O(n \lg n \sqrt{n} + \sqrt{n}\, \mathsf{I}(\sqrt{n}))$



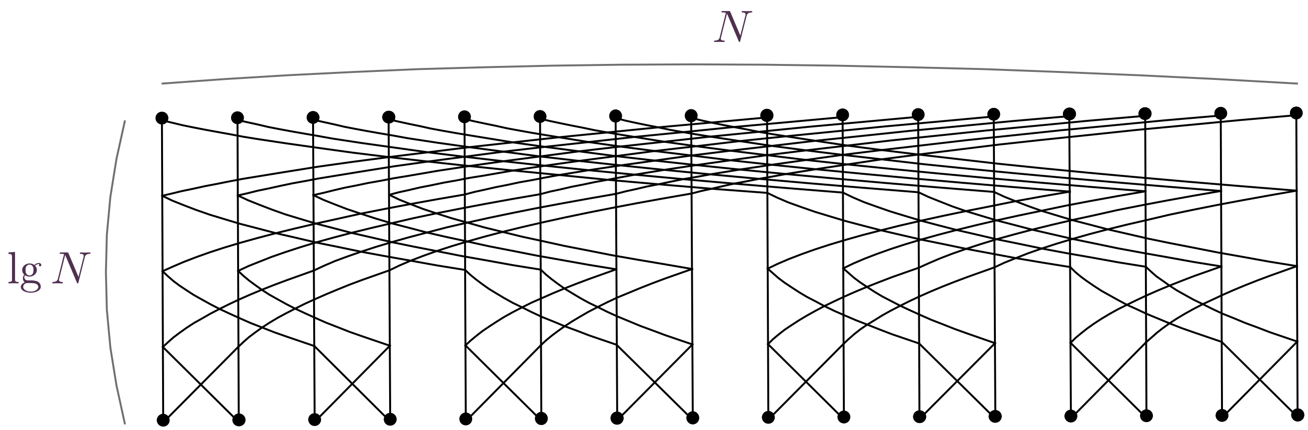
Cost of one DFT : $\frac{1}{2} N \lg N$ "butterflies" $\rightsquigarrow O(N \lg N)$ operations in $\mathbb{K}$

Complex DFT	$N \asymp n / \lg n$	$\mathsf{M}_{\mathbb{K}}(1) = O(\mathsf{I}(\lg n))$	$\mathsf{I}(n) = O(n \lg n\, \mathsf{I}(\lg n) + n \lg n)$
Modular DFT	$N \asymp n / \lg n$	$\mathsf{M}_{\mathbb{K}}(1) = O(\mathsf{I}(\lg n))$	$\mathsf{I}(n) = O(n \lg n\, \mathsf{I}(\lg n) + n \lg n)$
Synthetic DFT	$N \asymp \sqrt{n}$	butterfly $\rightsquigarrow O(\sqrt{n})$	$\mathsf{I}(n) = O(n \lg n \sqrt{n} + \sqrt{n}\, \mathsf{I}(\sqrt{n}))$



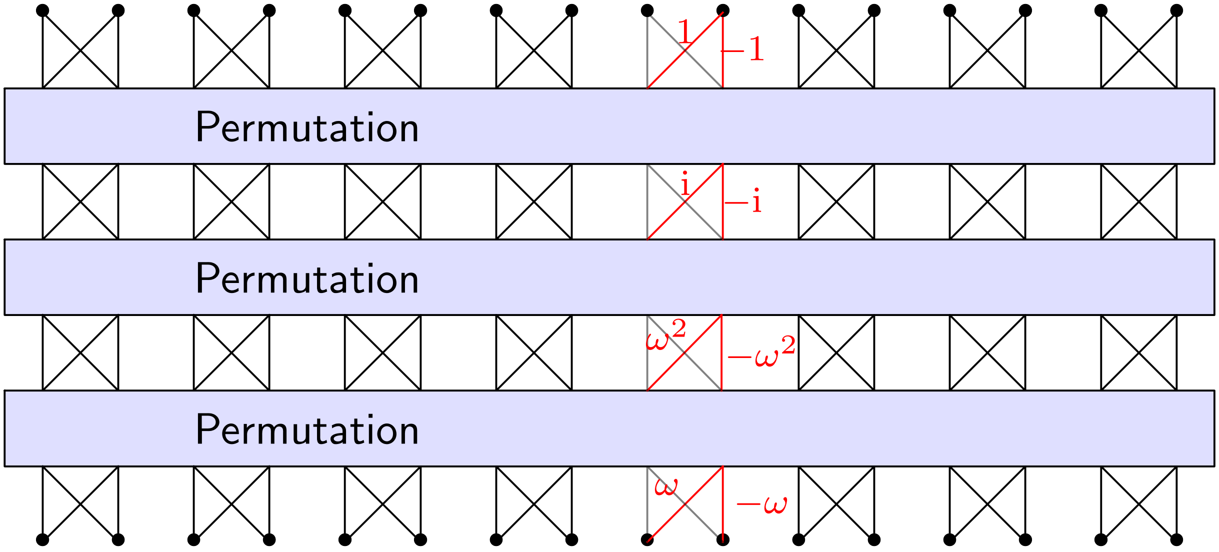
Cost of one DFT : $\frac{1}{2}N \lg N$ "butterflies" $\rightsquigarrow O(N \lg N)$ operations in $\mathbb{K}$

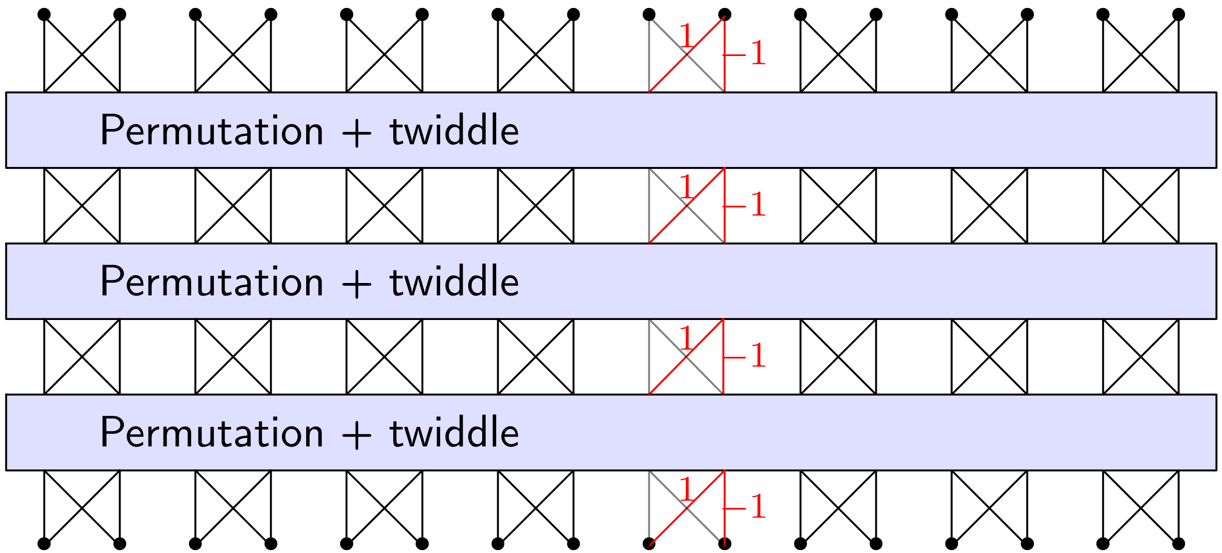
Complex DFT	$N \asymp n/\lg n$	$\mathsf{M}_{\mathbb{K}}(1) = O(\mathsf{I}(\lg n))$	$\mathsf{I}(n) = O(n \lg n\, \mathsf{I}(\lg n) + n \lg n)$
Modular DFT	$N \asymp n/\lg n$	$\mathsf{M}_{\mathbb{K}}(1) = O(\mathsf{I}(\lg n))$	$\mathsf{I}(n) = O(n \lg n\, \mathsf{I}(\lg n) + n \lg n)$
Synthetic DFT	$N \asymp \sqrt{n}$	butterfly $\rightsquigarrow O(\sqrt{n})$	$\mathsf{I}(n) = O(n \lg n \sqrt{n} + \sqrt{n}\, \mathsf{I}(\sqrt{n}))$

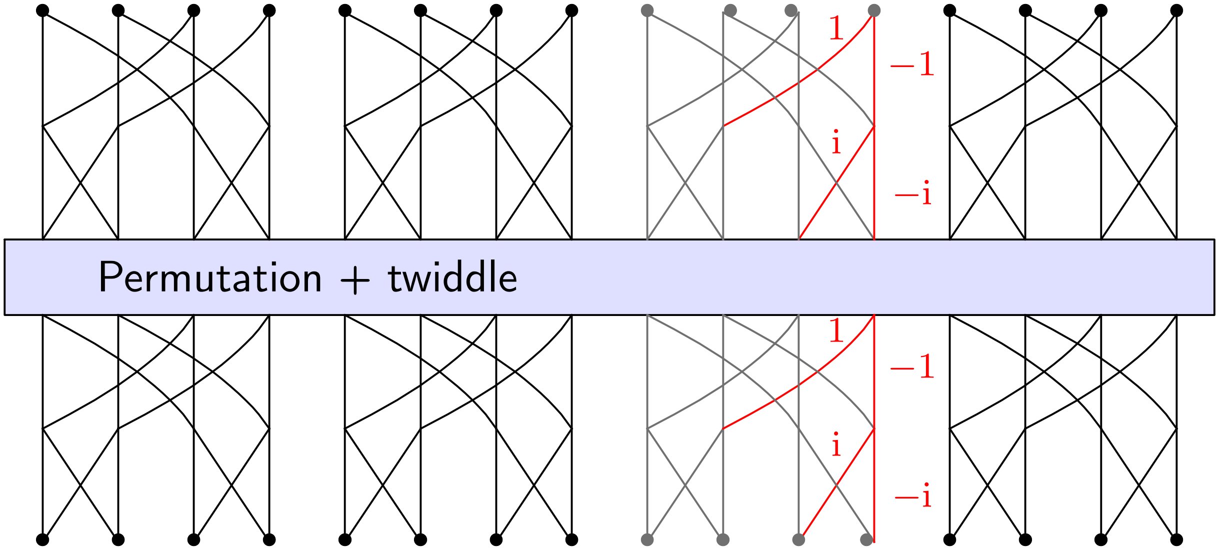


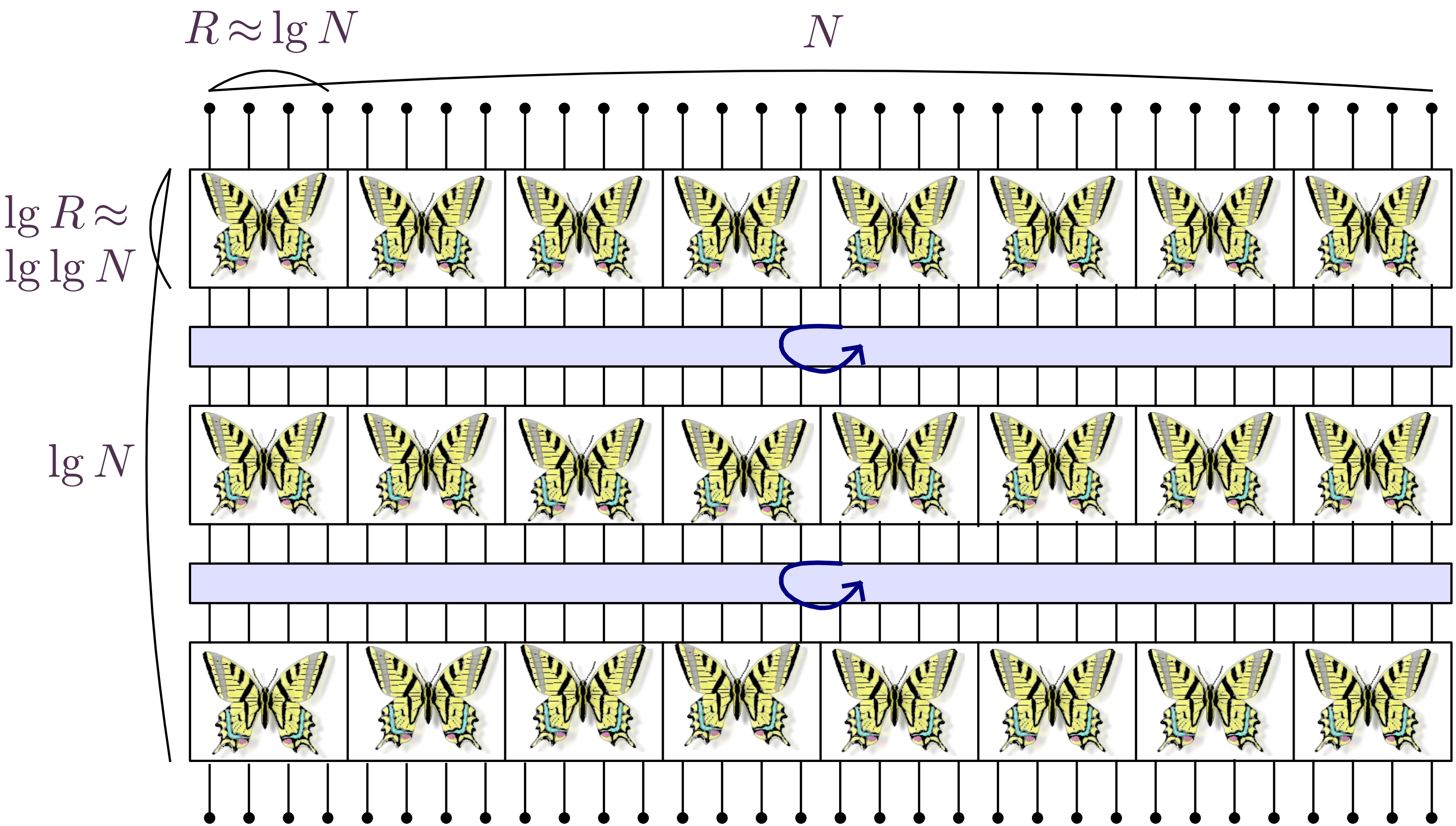
Cost of one DFT : $\frac{1}{2} N \lg N$ "butterflies" $\rightsquigarrow O(N \lg N)$ operations in $\mathbb{K}$

Complex DFT	$N \asymp n / \lg n$	$\mathsf{M}_{\mathbb{K}}(1) = O(\mathsf{I}(\lg n))$	$\mathsf{I}(n) = O(n \lg n \lg\lg n \lg\lg\lg n \cdots)$
Modular DFT	$N \asymp n / \lg n$	$\mathsf{M}_{\mathbb{K}}(1) = O(\mathsf{I}(\lg n))$	$\mathsf{I}(n) = O(n \lg n \lg\lg n \lg\lg\lg n \cdots)$
Synthetic DFT	$N \asymp \sqrt{n}$	butterfly $\rightsquigarrow O(\sqrt{n})$	$\mathsf{I}(n) = O(n \lg n \lg\lg n)$


Permutation
Permutation
Permutation
1
−1
i
−i
ω²
−ω²
ω
−ω


1
−1
Permutation + twiddle
1
−1
Permutation + twiddle
1
−1
Permutation + twiddle
1
−1


Permutation + twiddle
1
−1
i
−i
1
−1
i
−i



Slightly cheating in picture: we should have used 16 butterflies on every line

- Cut n-bit integer in N chunks of $\approx b/2$ bits. Use b-bit working precision.

- Cut n-bit integer in N chunks of $\approx b/2$ bits. Use b-bit working precision.
- Reduce FFT to giant butterflies of size $R \times \lg R$ with $R \approx b \approx \lg n$

- Cut n-bit integer in N chunks of $\approx b/2$ bits. Use b-bit working precision.
- Reduce FFT to giant butterflies of size $R \times \lg R$ with $R \approx b \approx \lg n$
- Reduce giant butterflies back to integer multiplication:

$$\text{DFT of size } R \times \lg R \text{ over } \mathbb{C}_b \overset{\text{Bluestein}}{\rightsquigarrow} O(\mathsf{M}_{\mathbb{C}_b[X]}(R)) \overset{\text{Kronecker}}{\rightsquigarrow} O(\mathsf{I}(R\,b))$$

(with respect to Fürer's method: Slower giant butterflies, but faster *twiddling*)

- Cut n-bit integer in N chunks of $\approx b/2$ bits. Use b-bit working precision.
- Reduce FFT to giant butterflies of size $R \times \lg R$ with $R \approx b \approx \lg n$
- Reduce giant butterflies back to integer multiplication:

  DFT of size $R \times \lg R$ over $\mathbb{C}_b \overset{\text{Bluestein}}{\rightsquigarrow} O(\mathsf{M}_{\mathbb{C}_b[X]}(R)) \overset{\text{Kronecker}}{\rightsquigarrow} O(\mathsf{I}(R\,b))$

  (with respect to Fürer's method: Slower giant butterflies, but faster *twiddling*)
- Cost of giant butterflies:

$$\mathsf{C}_1 = O\left(\frac{N}{R}\frac{\lg N}{\lg R}\mathsf{I}(R\,b)\right).$$

- Cut n-bit integer in N chunks of $\approx b/2$ bits. Use b-bit working precision.
- Reduce FFT to giant butterflies of size $R \times \lg R$ with $R \approx b \approx \lg n$
- Reduce giant butterflies back to integer multiplication:

DFT of size $R \times \lg R$ over $\mathbb{C}_b \overset{\text{Bluestein}}{\rightsquigarrow} O(\mathsf{M}_{\mathbb{C}_b[X]}(R)) \overset{\text{Kronecker}}{\rightsquigarrow} O(\mathsf{I}(R\,b))$

(with respect to Fürer's method: Slower giant butterflies, but faster *twiddling*)

- Cost of giant butterflies:

$$\mathsf{C}_1 = O\left(\frac{N}{R}\frac{\lg N}{\lg R}\mathsf{I}(R\,b)\right).$$

- Cost of twiddle factors:

$$\mathsf{C}_2 = O\left(N\frac{\lg N}{\lg R}\mathsf{I}(b)\right) = O\left(\frac{N}{R}\frac{\lg N}{\lg R}\mathsf{I}(R\,b)\right)$$

- Cut n-bit integer in N chunks of $\approx b/2$ bits. Use b-bit working precision.
- Reduce FFT to giant butterflies of size $R \times \lg R$ with $R \approx b \approx \lg n$
- Reduce giant butterflies back to integer multiplication:

DFT of size $R \times \lg R$ over $\mathbb{C}_b \overset{\text{Bluestein}}{\rightsquigarrow} O(\mathsf{M}_{\mathbb{C}_b[X]}(R)) \overset{\text{Kronecker}}{\rightsquigarrow} O(\mathsf{I}(R\,b))$

(with respect to Fürer's method: Slower giant butterflies, but faster *twiddling*)

- Cost of giant butterflies:

$$\mathsf{C}_1 \;=\; O\!\left(\frac{N}{R}\frac{\lg N}{\lg R}\mathsf{I}(R\,b)\right).$$

- Cost of twiddle factors:

$$\mathsf{C}_2 \;=\; O\!\left(N\frac{\lg N}{\lg R}\mathsf{I}(b)\right) \;=\; O\!\left(\frac{N}{R}\frac{\lg N}{\lg R}\mathsf{I}(R\,b)\right)$$

- Recursion:

$$\frac{\mathsf{I}(n)}{n\lg n} \;\leqslant\; K\frac{\mathsf{I}(R\,b)}{R\,b\lg(R\,b)} + O(1).$$

- Cut n-bit integer in N chunks of $\approx b/2$ bits. Use b-bit working precision.
- Reduce FFT to giant butterflies of size $R \times \lg R$ with $R \approx b \approx \lg n$
- Reduce giant butterflies back to integer multiplication:

  DFT of size $R \times \lg R$ over $\mathbb{C}_b \overset{\text{Bluestein}}{\rightsquigarrow} O(\mathsf{M}_{\mathbb{C}_b[X]}(R)) \overset{\text{Kronecker}}{\rightsquigarrow} O(\mathsf{I}(R\,b))$

  (with respect to Fürer's method: Slower giant butterflies, but faster *twiddling*)
- Cost of giant butterflies:

$$\mathsf{C}_1 = O\left(\frac{N}{R}\frac{\lg N}{\lg R}\mathsf{I}(R\,b)\right).$$

- Cost of twiddle factors:

$$\mathsf{C}_2 = O\left(N\frac{\lg N}{\lg R}\mathsf{I}(b)\right) = O\left(\frac{N}{R}\frac{\lg N}{\lg R}\mathsf{I}(R\,b)\right)$$

- Recursion:

$$\frac{\mathsf{I}(n)}{n\lg n} \leqslant K\frac{\mathsf{I}(R\,b)}{R\,b\lg(R\,b)} + O(1).$$

- Conclusion:

$$\mathsf{I}(n) = O(n\lg n\,K^{\log^* n}).$$

We now have several ways to show that

$$\mathsf{I}(n) \;=\; O(n \lg n \, K^{\log^* n}).$$

What is the best K we can get?

We now have several ways to show that

$$\mathsf{I}(n) \;=\; O(n \lg n \, K^{\log^* n}).$$

What is the best K we can get?

Fürer, after optimisation : $K = 16$ (?)
We, after optimisation : $K = 8$

We now have several ways to show that

$$\mathsf{I}(n) \; = \; O(n \lg n \, K^{\log^* n}).$$

What is the best K we can get?

Fürer, after optimisation : $K = 16$ (?)
We, after optimisation : $K = 8$

Ingredients

- Multiplication in $\mathbb{Z} \rightsquigarrow$ multiplication in $(\mathbb{Z} / (2^n - 1) \, \mathbb{Z})[\mathrm{i}]$.

We now have several ways to show that

$$\mathsf{I}(n) \;=\; O(n \lg n \, K^{\log^* n}).$$

What is the best K we can get?

Fürer, after optimisation : $K = 16$ (?)
We, after optimisation : $K = 8$

Ingredients

- Multiplication in $\mathbb{Z} \rightsquigarrow$ multiplication in $(\mathbb{Z}/(2^n - 1)\,\mathbb{Z})[\mathrm{i}]$.
- One argument shared many times in recursive calls $\rightsquigarrow$ 2 DFTs instead of 3.

We now have several ways to show that

$$\mathsf{I}(n) \;=\; O(n \lg n \, K^{\log^* n}).$$

What is the best K we can get?

Fürer, after optimisation : $K = 16$ (?)
We, after optimisation : $K = 8$

Ingredients

- Multiplication in $\mathbb{Z}$ $\rightsquigarrow$ multiplication in $(\mathbb{Z}/(2^n - 1)\,\mathbb{Z})[\mathrm{i}]$.
- One argument shared many times in recursive calls $\rightsquigarrow$ 2 DFTs instead of 3.
- Convolution of length N with b-bit coefficients $\rightsquigarrow$ output of size $2\,b + O(\lg N)$.

  Taking $b \asymp (\lg n)^2$ instead of $b \asymp \lg n$ improves the ratio $(2\,b + O(\lg N))\,/\,b$.

We now have several ways to show that

$$\mathsf{I}(n) \;=\; O(n \lg n \, K^{\log^* n}).$$

What is the best K we can get?

Fürer, after optimisation : $K = 16$ (?)
We, after optimisation : $K = 8$

Ingredients

- Multiplication in $\mathbb{Z}$ $\rightsquigarrow$ multiplication in $(\mathbb{Z}/(2^n - 1)\,\mathbb{Z})[\mathrm{i}]$.
- One argument shared many times in recursive calls $\rightsquigarrow$ 2 DFTs instead of 3.
- Convolution of length N with b-bit coefficients $\rightsquigarrow$ output of size $2\,b + O(\lg N)$.

  Taking $b \asymp (\lg n)^2$ instead of $b \asymp \lg n$ improves the ratio $(2\,b + O(\lg N))\,/\,b$.
- Increase $R \approx \lg N$ $\rightsquigarrow$ $R \approx (\lg N)^{\lg \lg N + O(1)}$.

  Cost Bluestein–Kronecker $\gg$ cost twiddling and other.

Where does the cost come from?

a) Factor 2 $\leftrightsquigarrow$ Kronecker segmentation ($\mathbb{Z}[\mathrm{i}] \rightsquigarrow \mathbb{C}_b[X]$, cutting into pieces of $\frac{b}{2}$ bits)
b) Factor 2 $\leftrightsquigarrow$ direct and inverse DFT
c) Factor 2 $\leftrightsquigarrow$ Kronecker substitution ($\mathbb{C}_b[X]/(X^R - 1) \rightsquigarrow \mathbb{Z}/(2^{2bR} - 1)\,\mathbb{Z}$)

Where does the cost come from?

a) Factor 2 $\leftrightsquigarrow$ Kronecker segmentation ($\mathbb{Z}[\mathrm{i}] \rightsquigarrow \mathbb{C}_b[X]$, cutting into pieces of $\frac{b}{2}$ bits)
b) Factor 2 $\leftrightsquigarrow$ direct and inverse DFT
c) Factor 2 $\leftrightsquigarrow$ Kronecker substitution ($\mathbb{C}_b[X]/(X^R-1) \rightsquigarrow \mathbb{Z}/(2^{2bR}-1)\mathbb{Z}$)

Fermat primes

And *if*, *if*, *if* there were sufficiently many prime numbers of the form $p = 2^{2^k} + 1$
(Optimized) Fürer approach for $\mathbb{K} = \mathbb{F}_p$ yields $K = 4$
Unfortunately..., $p = 2^{16} + 1$ is the largest known prime number of this form

Where does the cost come from?

a) Factor 2 $\leftsquigarrow$ Kronecker segmentation ($\mathbb{Z}[\mathrm{i}] \rightsquigarrow \mathbb{C}_b[X]$, cutting into pieces of $\frac{b}{2}$ bits)
b) Factor 2 $\leftsquigarrow$ direct and inverse DFT
c) Factor 2 $\leftsquigarrow$ Kronecker substitution ($\mathbb{C}_b[X]/(X^R-1) \rightsquigarrow \mathbb{Z}/(2^{2bR}-1)\,\mathbb{Z}$)

Fermat primes

And *if*, *if*, *if* there were sufficiently many prime numbers of the form $p = 2^{2^k}+1$
(Optimized) Fürer approach for $\mathbb{K} = \mathbb{F}_p$ yields $K = 4$
Unfortunately..., $p = 2^{16}+1$ is the largest known prime number of this form

Mersenne primes

Conjecture 3. *Let* $\pi_m(x) = \{p \leqslant x : p = 2^q - 1, p \textit{ prime}, q \textit{ prime}\}$. *Then* $\exists a < b$,

$$a \log \log x < \pi_m(x) < b \log \log x$$

Where does the cost come from?

a) Factor 2 $\leftsquigarrow$ Kronecker segmentation ($\mathbb{Z}[\mathrm{i}] \rightsquigarrow \mathbb{C}_b[X]$, cutting into pieces of $\frac{b}{2}$ bits)
b) Factor 2 $\leftsquigarrow$ direct and inverse DFT
c) Factor 2 $\leftsquigarrow$ Kronecker substitution ($\mathbb{C}_b[X]/(X^R - 1) \rightsquigarrow \mathbb{Z}/(2^{2bR} - 1)\,\mathbb{Z}$)

Fermat primes

And *if*, *if*, *if* there were sufficiently many prime numbers of the form $p = 2^{2^k} + 1$
(Optimized) Fürer approach for $\mathbb{K} = \mathbb{F}_p$ yields $K = 4$
Unfortunately..., $p = 2^{16} + 1$ is the largest known prime number of this form

Mersenne primes

Conjecture 4. *Let* $\pi_m(x) = \{p \leqslant x : p = 2^q - 1, p \ \mathit{prime}, q \ \mathit{prime}\}$. *Then* $\exists a < b$,

$$a \log \log x < \pi_m(x) < b \log \log x$$

Crandall–Fagin algorithm

Multiplication $\mathbb{F}_p[\mathrm{i}][X]/(X^M - 1) \rightsquigarrow \mathbb{F}_{p'}[\mathrm{i}][X, Y]/(X^M - 1, Y^N - 1)$, $p' \lll p$
Conjecture 4 $\Rightarrow$ $K = 4$