Polynomial Factorization: a Success Story

Erich Kaltofen
North Carolina State University
www.kaltofen.us

Letter by Gödel to John von Neumann 1956

Dr. Hao Wang with best wishes and sorry I then forgott to put it in your place

Princeton 20./III. 1956

"Goedl"

Lieber Herr v. Neumann!

Ich habe mit grösstem Bedauern von Ihrer Erkrankung gehört. Die Nachricht kam mir ganz

Princeton 20./III. 1956

Goedl

Lieber Herr v. Neumann!

Letter by Gödel to John von Neumann 1956

problems erhalten kann; 2. bedeutet ja $\varphi(n) \sim K \cdot n$
(oder $\sim K n^2$) bloss, dass die Anzahl der Schritte gegen-
über dem blossen Probieren von N auf $\log N$ (oder
$(\log N)^2$) verringert werden kann. So starke Verringer-
ungen kommen aber bei andern finiten Problemen
durchaus vor, z. B. bei der Berechnung eines quadra-
tischen Restsymbols durch wiederholte Anwendung des
Reziprozitätsgesetzes. Es wäre interessant zu wissen,

... Such strong speedups [N to $(\log N)^2$] can occur for other finite problems, e.g. when computing the quadratic residuosity by repeated application of the reciprocity law.

Letter by Gödel to John von Neumann 1956

Reziprozitätsgesetzes. Es wäre interessant zu wissen,
wie es damit z.B. bei der Feststellung, ob eine Zahl Prim-
zahl ist, steht u. wie stark <u>im allgemeinen</u> bei finiten
kombinatorischen Problemen die Anzahl der Schritte
gegenüber dem bloßen Probieren verringert werden kann.

... It would be interesting to know, how it is with that, e.g. about the decision if a number is a prime number, a. how much <u>in general</u> for finite combinatorial problems the number of steps can be reduced versus trying all possibilities.

Las Vegas Squareroots Modulo Large p

Problem: given a prime number $p > 2$ and $b \in \mathbb{Z}_p$
factor $x^2 - b \equiv (x+a)(x-a) \pmod{p}$

Algorithm: pick random $u, v \in \mathbb{Z}_p$ and compute

$$\mathrm{GCD}\left(x^2 - b,\ 1 + (ux+v)^{\frac{p-1}{2}} \bmod (x^2 - b)\right)$$

If

$$(ux+v)^{\frac{p-1}{2}} \bmod (x+a) \equiv (-ua+v)^{\frac{p-1}{2}} = -1$$
$$(ux+v)^{\frac{p-1}{2}} \bmod (x-a) \equiv (ua+v)^{\frac{p-1}{2}} \neq -1$$

the factor $x + a$ is found.

Monte Carlo Primality Testing

Problem: given an odd integer $m \neq k^j$, test if m is prime.

Algorithm: pick random $c \in \mathbb{Z}_m$ and factor $x^2 - (c^2 \bmod m)$.

If $\mathrm{GCD}(\ldots) = 1$ for most c, u, v

we are either very unlucky, or m is composite.

If $a \equiv \pm c \bmod m$ for all c,

we are either very unlucky, or m is prime.

Monte Carlo Primality Testing

Problem: given an odd integer $m \neq k^j$, test if m is prime.

Algorithm: pick random $c \in \mathbb{Z}_m$ and factor $x^2 - (c^2 \bmod m)$.

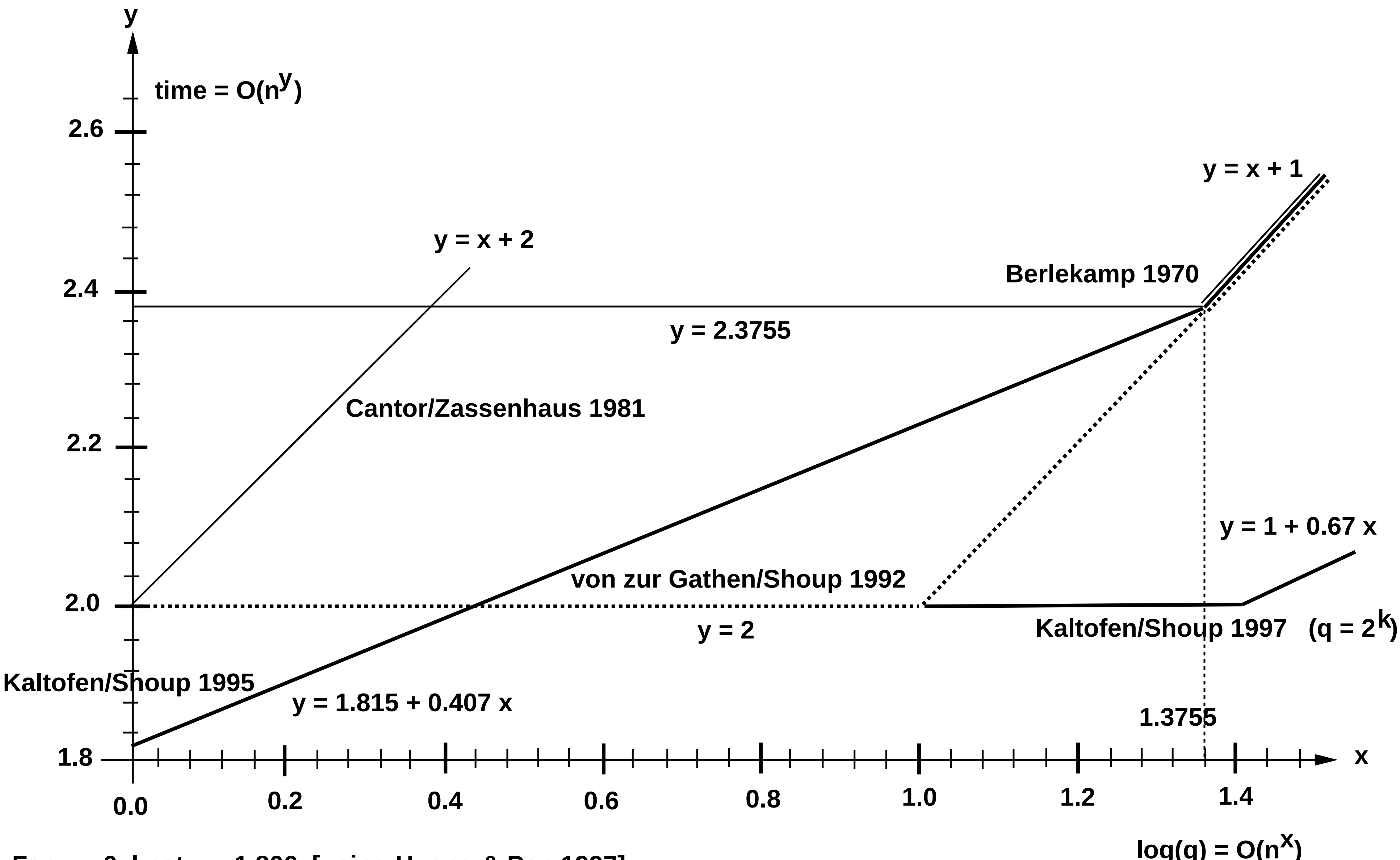
If $\mathrm{GCD}(\ldots) = 1$ for most c, u, v
we are either very unlucky, or m is composite.

If $a \equiv \pm c \bmod m$ for all c,
we are either very unlucky, or m is prime.

Reason: if m is composite,
there are two $a_1, a_2 \neq \pm c$ with $a_1^2 \equiv a_2^2 \equiv c^2 \pmod{m}$

Example: $37 \equiv 10^2 \equiv 17^2 \equiv 46^2 \equiv 53^2 \pmod{63}$

Best algorithms for $\mathbb{F}_q[x]$: $O(n^y)$ arithmetic operations in $\mathbb{F}_q$



For x = 0, best y = 1.806 [using Huang & Pan 1997]

n = degree, q = number of field elements

Factorization in $\mathbb{Z}[x]$

Berlekamp/Zassenhaus 1969 algorithm exponential in worst case

LLL 1982 overcome by lattice basis reduction

Sasaki et al. 1993/van Hoeij 2000 find small low dimen. lattices

Factorization in $\mathbb{Z}[x]$

Sasaki et al. 1993/van Hoeij 2000 find small low dimen. lattices

Idea: Let $f \equiv g \cdot h \cdots w \pmod{p^k}$, $\alpha_j, \beta_j, \ldots, \omega_j$ be the roots of $g, h, \ldots, w$; compute short column space vector of

$$\begin{array}{cl}
\left[\begin{array}{ccccccc}
C & 0 & \ldots & 0 & 0 & 0 & \ldots \\
0 & C & \ldots & 0 & \vdots & \vdots & \\
\vdots & & \ddots & \vdots & & & \\
0 & 0 & \ldots & C & 0 & 0 & \ldots \\
\sum_j \alpha_j & \sum_j \beta_j & \ldots & \sum_j \omega_j & p^k & 0 & \ldots \\
\sum_j \alpha_j^2 & \sum_j \beta_j^2 & \ldots & \sum_j \omega_j^2 & 0 & p^k & \\
\vdots & \vdots & & \vdots & \vdots & & \ddots
\end{array}\right]
&
\begin{array}{l}
\longleftarrow \text{forces } 0 \text{ or } 1 \text{ component} \\
\\
\\
\\
\longleftarrow \text{second highest coefficients} \\
\longleftarrow \text{via Newton identities} \\
\\
\end{array}
\end{array}$$

$$\uparrow \quad \uparrow$$

adjustment modulo p^k

Factorization in $\mathbb{Z}[x]$

From my 1982 survey

"As we will see below, in the worst case step (F5) is the dominant step in our algorithm. Therefore one is advised to test first whether the **second highest coefficient is absolutely smaller than** $\deg(f)\|f\|_2$, the corresponding factor coefficient bound, or whether the constant coefficient of $g(x)$ divides that of $f(x)$."

Factorization in $\mathbb{Z}[x]$

From my 1982 survey

"As we will see below, in the worst case step (F5) is the dominant step in our algorithm. Therefore one is advised to test first whether the **second highest coefficient is absolutely smaller than** $\deg(f)\|f\|_2$, the corresponding factor coefficient bound, or whether the constant coefficient of $g(x)$ divides that of $f(x)$."

Factorization in $\mathbb{Z}_p[y][x]$

Masayuki Noro and Kazuhiro Yokoyama [ISSAC 2002] use Gröbner walk to obtain fantastic practical performance.

Gao's 2000 algorithm in $\mathbb{C}[x,y]$

Let $f \in \mathbb{C}[x,y]$ squarefree

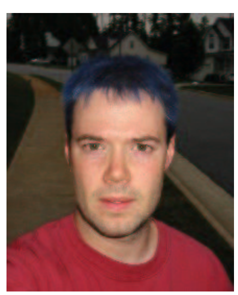
Ruppert's [1986] differential equation

$$\frac{\partial}{\partial y}\left(\frac{g}{f}\right) = \frac{\partial}{\partial x}\left(\frac{h}{f}\right), \begin{cases} \deg_x(g) < \deg_x(f), \deg_x(h) \le \deg_x(f), \\ \deg_y(g) \le \deg_y(f), \deg_y(h) < \deg_y(f). \end{cases}$$

For the factorization $f = f_1 \cdots f_r$ over $\mathbb{C}$ we have

$$g = \lambda_1 \frac{f}{f_1}\frac{\partial f_1}{\partial x} + \cdots + \lambda_r \frac{f}{f_r}\frac{\partial f_r}{\partial x}, \quad \lambda_i \in \mathbb{C},$$

$$f = \prod_{\lambda \in \mathbb{C}} \mathrm{GCD}\left(f, g - \lambda \frac{\partial f}{\partial x}\right).$$

$\longrightarrow$John May's  talk

Hard problems for sparse polynomials $\sum_i c_i z^{e_i} \in \mathbb{Z}[z]$

Plaisted 1977: Let $N = \prod_{i=1}^{n} p_i$, where p_i distinct primes.

Formula	Polynomial	Rootset
x_j	$z^{\frac{N}{p_j}} - 1$	$\{(e^{\frac{2\pi\mathbf{i}}{N}})^a \mid a \equiv 0 \pmod{p_j}\}$
$\neg x_k$	$\dfrac{z^N - 1}{z^{\frac{N}{p_k}} - 1} = \sum_{i=0}^{p_k-1} z^{\frac{iN}{p_k}}$	$\{(e^{\frac{2\pi\mathbf{i}}{N}})^b \mid b \not\equiv 0 \pmod{p_k}\}$
$L_1 \vee L_2$	$\mathrm{LCM}(\mathrm{Poly}(L_1), \mathrm{Poly}(L_2))$	$\mathrm{Roots}(L_1) \cup \mathrm{Roots}(L_2)$
$x_j \vee \neg x_k$	$\dfrac{(z^{\frac{N}{p_j p_k}} - 1)(z^N - 1)}{z^{\frac{N}{p_k}} - 1}$	(is sparse polynomial)

Hard problems for sparse polynomials $\sum_i c_i z^{e_i} \in \mathbb{Z}[z]$

Plaisted 1977: Let $N = \prod_{i=1}^{n} p_i$, where p_i distinct primes.

Formula	Polynomial	Rootset
x_j	$z^{\frac{N}{p_j}} - 1$	$\{(e^{\frac{2\pi \mathbf{i}}{N}})^a \mid a \equiv 0 \pmod{p_j}\}$
$\neg x_k$	$\dfrac{z^N - 1}{z^{\frac{N}{p_k}} - 1} = \sum_{i=0}^{p_k-1} z^{\frac{iN}{p_k}}$	$\{(e^{\frac{2\pi \mathbf{i}}{N}})^b \mid b \not\equiv 0 \pmod{p_k}\}$
$L_1 \vee L_2$	$\mathrm{LCM}(\mathrm{Poly}(L_1), \mathrm{Poly}(L_2))$	$\mathrm{Roots}(L_1) \cup \mathrm{Roots}(L_2)$
$x_j \vee \neg x_k$	$\dfrac{(z^{\frac{N}{p_j p_k}} - 1)(z^N - 1)}{z^{\frac{N}{p_k}} - 1}$	(is sparse polynomial)
$C_1 \wedge C_2$	$\mathrm{GCD}(\mathrm{Poly}(C_1), \mathrm{Poly}(C_2))$	$\mathrm{Roots}(C_1) \cap \mathrm{Roots}(C_2)$

Theorem *$C_1 \wedge \cdots \wedge C_l$ is satisfiable*

$$\iff \mathrm{GCD}(\mathrm{Poly}(C_1), \ldots, \mathrm{Poly}(C_l)) \neq 1.$$

Other hard problems [Plaisted 1977/78]

1. Given sequences $a_1, \ldots, a_m \in \mathbb{Z}$ and $b_1, \ldots b_n \in \mathbb{Z}$ determine whether

$$\prod_{i=1}^{m} (z^{a_i} - 1) \quad \text{is not a factor of} \quad \prod_{i=1}^{n} (z^{b_i} - 1).$$

2. Given a set $\{a_1, \ldots, a_m\} \subset \mathbb{Z}$ determine whether

$$\int_0^{2\pi} \cos(a_1\theta) \cdots \cos(a_m\theta) d\theta \neq 0.$$

Easy problems for sparse polynomials $f = \sum_i c_i x^{e_i} \in \mathbb{Z}[z]$

Cucker, Korian, Smale 1998: Compute root $a \in \mathbb{Z}: \quad f(a) = 0.$

Gap idea: if $f(a) = 0, a \neq \pm 1$ then $g_1(a) = \cdots = g_s(a) = 0$
where $f(x) = \sum_j g_j(x) x^{u_j}$ and $u_{j+1} - u_j - \deg(g_j) > b.$

Easy problems for sparse polynomials $f = \sum_i c_i x^{e_i} \in \mathbb{Z}[z]$

Cucker, Korian, Smale 1998: Compute root $a \in \mathbb{Z}$: $f(a) = 0$.

Gap idea: if $f(a) = 0, a \neq \pm 1$ then $g_1(a) = \cdots = g_s(a) = 0$
where $f(x) = \sum_j g_j(x) x^{u_j}$ and $u_{j+1} - u_j - \deg(g_j) > b$.

Write $f(x) = \underbrace{g(x)}_{\deg(g) < d} + x^{d+b} h(x), \quad \|f\|_1 = |c_1| + \cdots + |c_t|$.

For $a \neq \pm 1$, $h(a) \neq 0$:

$$\begin{aligned} |g(a)| &< \|f\|_1 \cdot |a|^d \\ |a^{d+b} h(a)| &> |a|^{d+b} \end{aligned}$$

Easy problems for sparse polynomials $f = \sum_i c_i x^{e_i} \in \mathbb{Z}[z]$

Cucker, Korian, Smale 1998: Compute root $a \in \mathbb{Z}: \quad f(a) = 0.$

Gap idea: if $f(a) = 0, a \neq \pm 1$ then $g_1(a) = \cdots = g_s(a) = 0$
where $f(x) = \sum_j g_j(x) x^{u_j}$ and $u_{j+1} - u_j - \deg(g_j) > b.$

Write $f(x) = \underbrace{g(x)}_{\deg(g) < d} + x^{d+b} h(x), \quad \|f\|_1 = |c_1| + \cdots + |c_t|.$

For $a \neq \pm 1, h(a) \neq 0:$

$$\begin{aligned} |g(a)| &< \|f\|_1 \cdot |a|^d \\ |a^{d+b} h(a)| &> |a|^{d+b} \end{aligned}$$

$b > \log_2 \|f\|_1 \Longrightarrow |a|^{d+b} > 2^b \cdot |a|^d > \|f\|_1 \cdot |a|^d \Longrightarrow f(a) \neq 0.$

Generalization by H. W. Lenstra, Jr. 1999

Input: a sparse $f(x) = \sum_{i=1}^{t} c_i x^{e_i} \in \mathbb{Z}[z]$
$\varphi(\zeta) \in \mathbb{Z}[\zeta]$ monic irred.; let $K = \mathbb{Q}[\zeta]/(\varphi(\zeta))$
a factor degree bound d

Output: a list of all irreducible factors of f over K of degree $\leq d$

Bit complexity is $\left(\underline{t + \log(\deg f)} + \log\|f\| + \log\|\varphi\|\right)^{O(d \cdot \deg(\varphi))}$

Special case $\varphi = 1, d = 1:$ Algorithm finds all rational roots in polynomial time.

Open Problem: Roots of Trinomials in $\mathbb{Z}_p$

Given a prime number p and integers $b, c \in \mathbb{Z}_p$, $d > e$ compute $y \in \mathbb{Z}_p$ such that

$$y^d + by^e + c \equiv 0 \pmod{p}$$

in time

$$\big(\log(d) + \log(p)\big)^{O(1)}$$

Alternatively, prove that computing a root in $\mathbb{Z}_p$ of a polynomial given by straight-line program over $\mathbb{Z}_p$ is NP-hard.

Status of My ECCAD'98 Challenge Problems

Problem 1: *Nearby multivariate polynomials that factor over* $\mathbb{C}$

<u>Status</u>: Open, but many new numerical algorithms

E.g., Lihong Zhi shows some early success with the algorithm suggested in the paper here with John May

Status of My ECCAD'98 Challenge Problems

Problem 1: *Nearby multivariate polynomials that factor over* $\mathbb{C}$

Status: Open, but many new numerical algorithms

E.g., Lihong Zhi shows some early success with the algorithm suggested in the paper here with John May

Problem 2: *Collins's Zolotarev's problem on a computer*

Status: Unknown

Status of My ECCAD'98 Challenge Problems

Problem 1: *Nearby multivariate polynomials that factor over* $\mathbb{C}$
Status: Open, but many new numerical algorithms
E.g., Lihong Zhi shows some early success with the algorithm suggested in the paper here with John May

Problem 2: *Collins's Zolotarev's problem on a computer*
Status: Unknown

Problem 3: *Characteristic polynomial of a black box matrix*
Status: Progress in [Villard CASC 2000]
Improved bit complexity results in dense case by Kaltofen and Villard 2003.

Status of My ECCAD'98 Challenge Problems

Problem 4: *Lattice basis reduction-safe GGH-like cryptosystems*
<u>Status</u>: See `www.ntru.com`, CRYPTO 2003

Status of My ECCAD'98 Challenge Problems

Problem 4: *Lattice basis reduction-safe GGH-like cryptosystems*
Status: See `www.ntru.com`, CRYPTO 2003

Problem 5: *Gröbner bases via iterative numerical methods*
Status: Ongoing work; see [Traverso and Zanoni ISSAC 2002]

Status of My ECCAD'98 Challenge Problems

Problem 4: *Lattice basis reduction-safe GGH-like cryptosystems*
Status: See `www.ntru.com`, CRYPTO 2003

Problem 5: *Gröbner bases via iterative numerical methods*
Status: Ongoing work; see [Traverso and Zanoni ISSAC 2002]

Problem 6: *Space and time efficient transposition principle*
Status: Substantial progress [Bostan, Lecerf, Schost ISSAC 2003]

Status of My ECCAD'98 Challenge Problems

Problem 7: *Plug-and-play and generic programming methodology for symbolic computation*

<u>Status</u>: Open

Surprises from LinBox project using C++ allocators

```
myAllocator a;
myAllocator::pointer p = a.allocate(1);
a.construct(p,0); // effect: new((void*)p) T(0)
a.destroy(p);     // effect: ((T*)p)->~T()
a.deallocate(p,1);
```

Status of My ECCAD'98 Challenge Problems

Problem 7: *Plug-and-play and generic programming methodology for symbolic computation*
<u>Status</u>: Open

Surprises from LinBox project using C++ allocators

```
myAllocator a;
myAllocator::pointer p = a.allocate(1);
a.construct(p,0); // effect: new((void*)p) T(0)
a.destroy(p);     // effect: ((T*)p)->~T()
a.deallocate(p,1);
```

ANSI/ISO 14882 Section 20.1.5.4

"Implementations of containers ... are permitted to assume that their Allocator template parameter meets the following two additional requirements ...
— the typedef members `pointer`, ... **are required to be** `T*` ..."

Status of My ECCAD'98 Challenge Problems

Problem 8: *Another "killer" application besides education*
Status: 1999 Physics Nobel Prize for SCHOONSCHIP
Subject of ACA 2003 panel discussion

What is an algorithm?

– **finite** unambiguous list of steps ("control, program")

– computes a function from $D \longrightarrow E$ where D is **infinite** ("infinite Turing tape")

Ambiguity through randomization

– Monte Carlo (BPP): "always fast, probably correct". Examples: `isprime`

Lemma *[DeMillo&Lipton'78, Schwartz/Zippel'79]*
Let $f, g \in \mathbb{F}[x_1, \ldots, x_n], f \neq g, S \subseteq \mathbb{F}$.

$$\begin{aligned}\text{Probability}(f(a_1,\ldots,a_n) \neq g(a_1,\ldots,a_n) \mid a_i \in S) \\ \geq 1 - \max\{\deg(f), \deg(g)\}/\text{cardinality}(S)\end{aligned}$$

sparse polynomial interpolation, factorization, minimal polynomial of a sparse matrix

Do we exactly know what the algorithm computes? E.g., in the presence of floating point arithmetic?

– Las Vegas (RP): "always correct, probably fast".
Examples: polynomial factorization in $\mathbb{Z}_p[x]$, where $p \gg 2$.
Determinant of a sparse matrix

De-randomization: conjectured slow-down is within polynomial complexity.

Shuhong Gao, E. Kaltofen, and Lauder, A., "Deterministic distinct degree factorization for polynomials over finite fields," 2001.

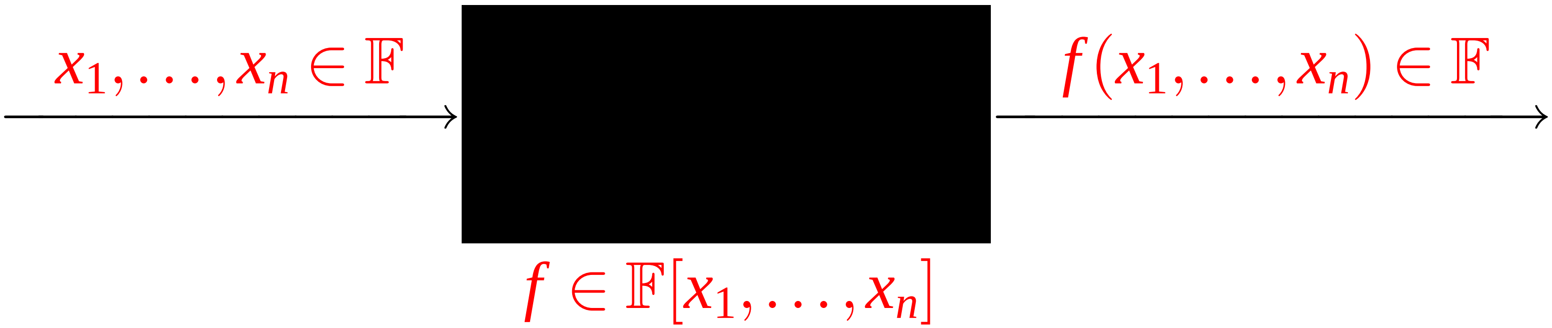
M. Agrawal, N. Kayal, N. Saxena, "PRIMES is in P," 2002.

Kabanets and Impagliazzo [STOC 2003]

If Schwartz/Zippel **can be** de-randomized (subexponentially), then there **do not** exist polynomial-size circuits for NEXP or the permanent.

Efficiency dilemma: the higher the confidence in the result, the more time it takes to compute it.

Black box polynomials

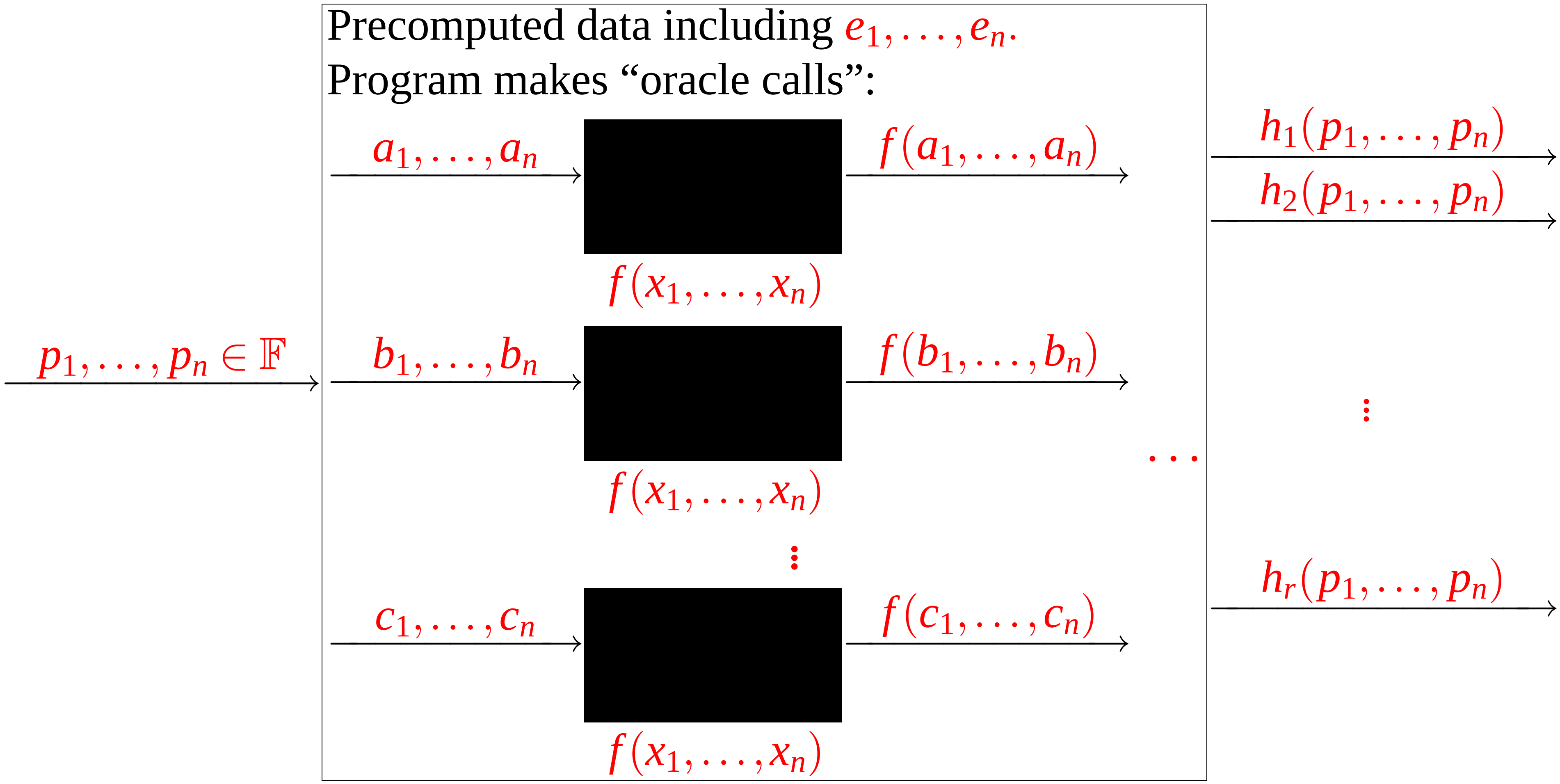


$\mathbb{F}$ an arbitrary field, e.g., rationals, reals, complexes

Perform polynomial algebra operations, e.g., factorization with

$$(n \cdot \deg(f))^{O(1)} \begin{cases} \text{black box calls,} \\ \text{arithmetic operations in } \mathbb{F} \text{ and} \\ \text{randomly selected elements in } \mathbb{F} \end{cases}$$

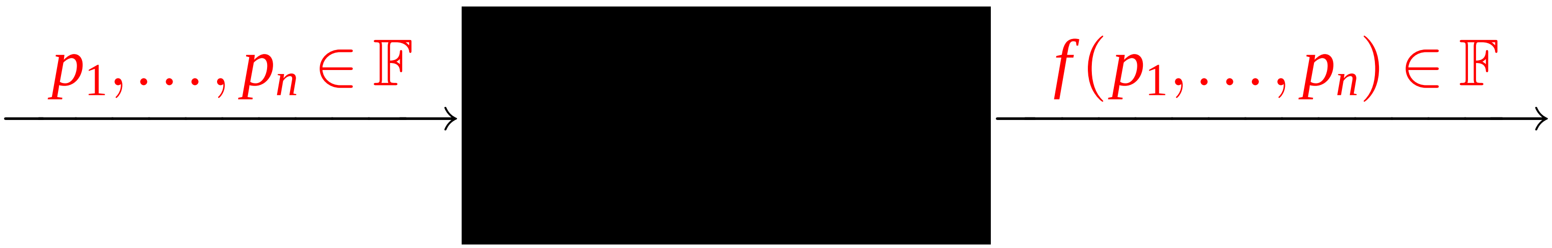
Kaltofen and Trager 1988 efficiently construct the following efficient program:



$$f(x_1, \ldots, x_n) = h_1(x_1, \ldots, x_n)^{e_1} \cdots h_r(x_1, \ldots, x_n)^{e_r}$$

$h_i \in \mathbb{F}[x_1, \ldots, x_n]$ irreducible.

Given a black box



$f(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n]$
$\mathbb{F}$ a field

compute by multiple evaluation of this black box the sparse representation of f

$$f(x_1, \dots, x_n) = \sum_{i=1}^{t} a_i x_1^{e_{i,1}} \cdots x_n^{e_{i,n}}, \quad a_i \neq 0$$

Many algorithms that are polynomial-time in $\deg(f), n, t$:

Zippel 1979, 1988; Ben-Or, Tiwari 1988

Kaltofen, Lakshman, Wiley 1988, 1990

Grigoriev, Karpinski, Singer 1988

Kaltofen, Lee, Lobo 2000, 2003

Mansour 1992; Giesbrecht, Lee, Labahn 2003: numerical method

Sparsity with non-standard basis

In place of x^e use

$(x-a)^e$	shifted basis
$x(x+1)\cdots(x+e-1)$	Pochhammer basis
$T_e(x)$	Chebyshev basis

Algorithms:

Lakshman, Saunders 1992, 1994: Chebyshev, Pochh., shifted
Grigoriev, Karpinski 1993: shifted
Grigoriev, Lakshman 1995: shifted
Lee 2001: Chebyshev, Pochhammer, shifted
Giesbrecht, Kaltofen and Lee 2002, 2003: shifted

FoxBox [Díaz, Kaltofen 1998] example: determinant of symmetric Toeplitz matrix

$$\det\left(\begin{bmatrix} a_0 & a_1 & \dots & a_{n-2} & a_{n-1} \\ a_1 & a_0 & \dots & a_{n-3} & a_{n-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n-2} & a_{n-3} & \dots & a_0 & a_1 \\ a_{n-1} & a_{n-2} & \dots & a_1 & a_0 \end{bmatrix}\right)$$

$$= F_1(a_0, \dots, a_{n-1}) \cdot F_2(a_0, \dots, a_{n-1}).$$

over the integers.

Serialization of **factors box** of 8 by 8 symmetric Toeplitz matrix modulo 65521

15,8,-1,1,2,2,-1,8,1,7,1,1,20752,-1,1,39448,33225,984,17332,53283,
35730,23945,13948,22252,52005,13703,8621,27776,33318,2740,
4472,36959,17038,55127,16460,26669,39430,1,0,1,4,20769,16570,
58474,30131,770,4,25421,22569,51508,59396,10568,4,20769,16570,
58474,30131,770,8,531,55309,40895,38056,34677,30870,397,59131,
12756,3,13601,54878,13783,39334,3,41605,59081,10842,15125,
3,45764,5312,9992,25318,3,59301,18015,3739,13650,3,23540,44673,
45053,33398,3,4675,39636,45179,40604,3,49815,29818,2643,16065,
3,46787,46548,12505,53510,3,10439,37666,18998,32189,3,38967,
14338,31161,12779,3,27030,21461,12907,22939,3,24657,32725,
47756,22305,3,44226,9911,59256,54610,3,56240,51924,26856,52915,
3,16133,61189,17015,39397,3,24483,12048,40057,21323

Serialization of **checkpoint** during sparse interpolation

28, 14, 9, 64017, 31343, 5117, 64185, 47755, 27377, 25604, 6323, 41969, 14, 3, 4, 0, 0, 3, 4, 0, 1, 3, 4, 0, 2, 3, 4, 0, 3, 3, 4, 0, 4, 3, 4, 1, 0, 3, 4, 1, 1, 3, 4, 1, 2, 3, 4, 1, 3, 3, 4, 2, 0, 3, 4, 2, 1, 3, 4, 2, 2, 3, 4, 3, 0, 3, 4, 3, 1, 14, 59877, 1764, 59012, 44468, 1, 19485, 25871, 3356, 2, 58834, 49014, 65518, 15714, 65520, 1, 2, 4, 4, 1, 1

Numerical	*Randomized (Monte Carlo)*
more efficiency, but approximate result	more efficiency, but uncertain result
ill-conditionedness near singular inputs	unfavorable inputs: pseudo-primes, $\sum_i \prod_j (x_i - j)$, Coppersmith's "pathological" matrices
convergence analysis	probabilistic analysis
try algorithms on unproven inputs	try algorithms with limited randomness

Numerical + randomized, e.g., LinBox's matrix preconditioners:
all of the above(?)

Hallmarks of a good heuristic

– Is algorithmic in nature, i.e., always terminates with a result of possibly unknown validity

– Is a proven complete solution in a more stringent setting, for example, by restricting the inputs or by slowing the algorithm

– Has an experimental track record, for example, works on 50% of cases

A Protocol for Spam Prevention [M. Naor et al., CRYPTO 2003]

From: "Dr. Cecilia Samarachi (Mrs)" <C.Samara91Dr@netscape.net>
Date: Sun, 25 May 2003 13:15:39
To: kaltofen@math.ncsu.edu
Dear Friend, VERY URGENT BUSINESS RELATIONSHIP.
...
My Ministry wants to award some major contracts and this contracts have been approved, implementation is on the pipeline and this contract is on supply of AGRICULTURAL CHEMICAL AND DRUGS/INJECTIONS FOR COW TREATMENT.
...
1. I want to use this last opportunity while still in the office to extract some money by inflating this contract to be awarded, and the over-invoiced amount I will use to establish my own hospital in U.K. or Germany after the transaction.

2. The inflated money (over-invoiced) from this contract will be immediately paid (Transfered) to my account in U.K. on confirmation of payment to your Bank.

3. I sincerely promise to approve your quotations on submission at all cost, provided my additional amount in your quotation will be 100% safe, immediately payment is made to your company. We would sign an agreement for the security and safety for my secret commission from the (over-invoiced) contarct.
...
Yours Faithfully,
Dr.(Mrs) Cecilia Samarachi.

Main idea:

1. take the unique message header as numeric data
2. spammer must perform “hard” computation and submit result with message
3. recipient “easily” checks result before accepting message

Main idea:
1. take the unique message header as numeric data
2. spammer must perform "hard" computation and submit result with message
3. recipient "easily" checks result before accepting message

Example: for message data m, compute a and "small" δ such that

$$a^2 \equiv m + \delta \pmod{p} \quad \text{and} \quad 10^5 \text{ divides } a.$$

Note: squareroot modulo p is $(\log p)^{2+o(1)}$, squaring $(\log p)^{1+o(1)}$.
See Maple worksheet.

Dwork, Goldberg, Naor design random table-lookup scheme that causes cache faults

NEEDED: non-localizable algorithmic problems whose results are easy to check

My suggestion: let spammer contribute to common good by spinning on a useful factorization, Gröbner basis,... problem