Computing with Polynomials Given by Black Boxes for their Evaluations: Greatest Common Divisors, Factorization, Separation of Numerators and Denominators

Erich Kaltofen

Rensselaer Polytechnic Institute
Department of Computer Science

Joint Work with: Barry Trager

IBM Research at Yorktown Heights

Sparse Multivariate Interpolation Problem

Given a black box

$f(x_1, \ldots, x_n) \in \mathsf{K}[x_1, \ldots, x_n]$
K a field of characteristic 0

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

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

Several solutions that are polynomial in n and t (some even in $\mathcal{NC}$)

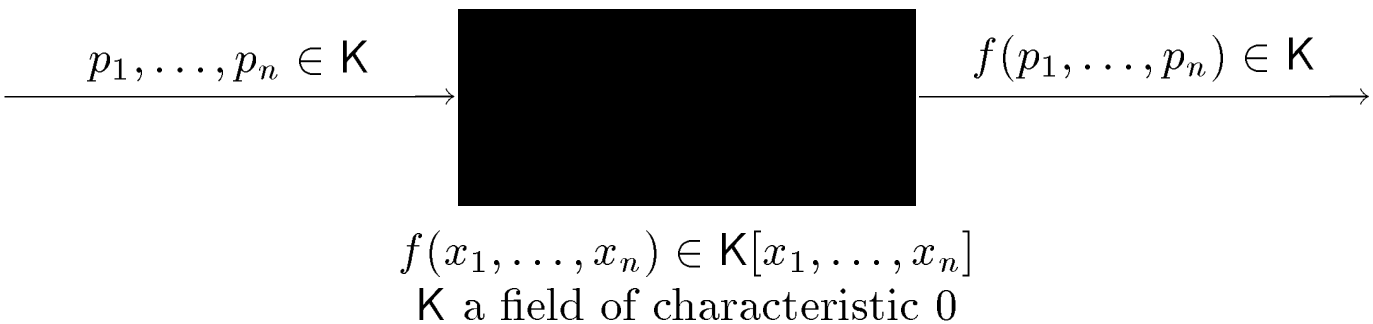
Zippel [EuroSAM 1979, January 1988]
Ben-Or, Tiwari [STOC 1988]
Kaltofen, Lakshman [ISSAC 1988]
Grigoryev, Karpinski, Singer [May 1988]

⋮

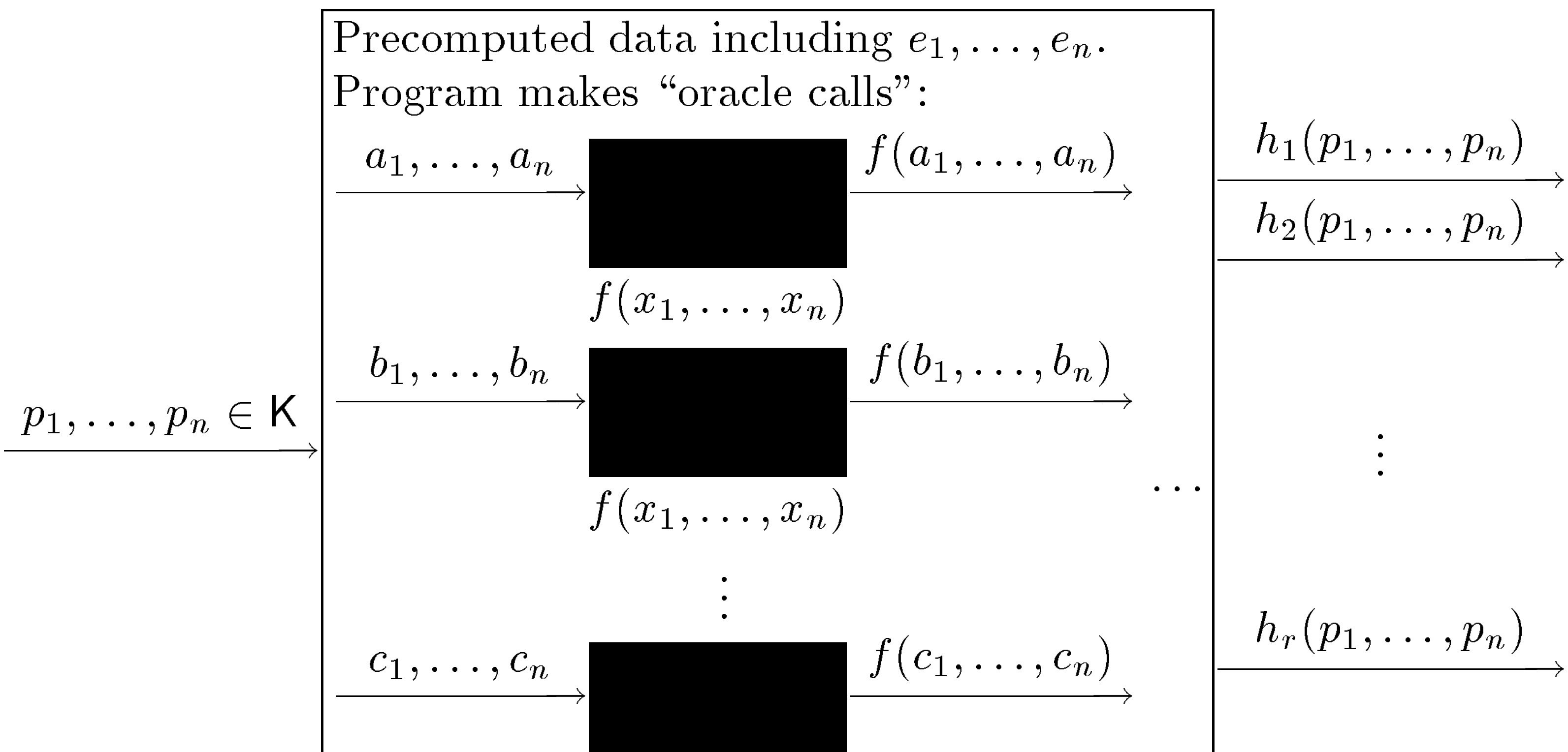
Our solution has the best running time so far

Black Box Factorization Problem

Given a black box

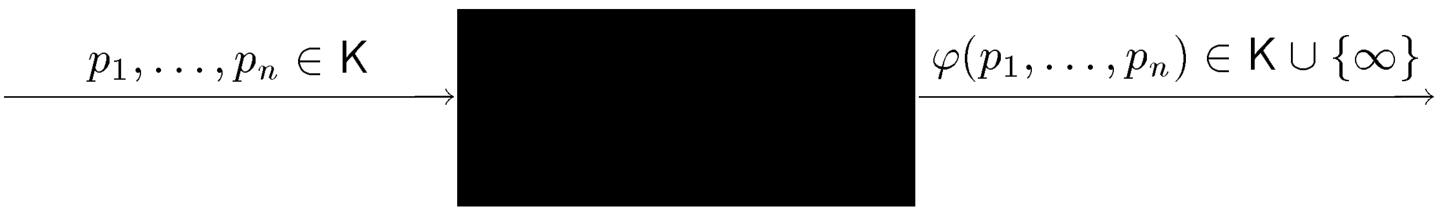


efficiently construct the following feasible program



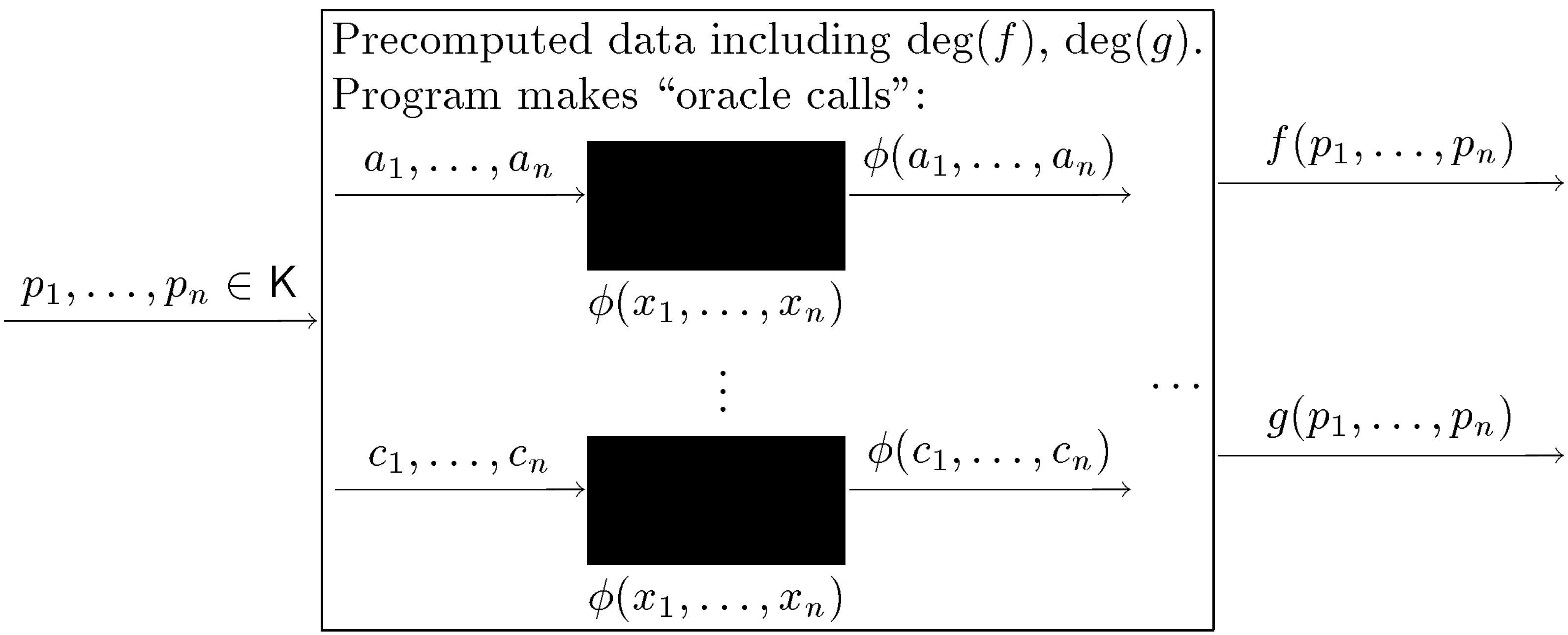
Numerator/Denominator Separation Problem

Given a black box



$\varphi(x_1, \ldots, x_n) \in \mathsf{K}(x_1, \ldots, x_n)$
K a field of characteristic 0

efficiently construct the following feasible program



$\phi(x_1, \ldots, x_n) = \frac{f(x_1, \ldots, x_n)}{g(x_1, \ldots, x_n)}, f, g \in \mathsf{K}[x_1, \ldots, x_n], \mathrm{GCD}(f, g) = 1.$

Characterization of Factor Evaluation Program

- Always evaluates the same associate of each factor

$$x\ y \quad \text{vs.} \quad (\frac{1}{2}x)\ (2y)$$

- Construction of program is Monte-Carlo (might produce incorrect program with probability $\le \epsilon$), and requires a factorization procedure for $\mathsf{K}[y]$, but the program itself is deterministic
- Program contains positive integer constants of value bounded by

$$\frac{2^{\deg(f)^{1+o(1)}}}{\epsilon}$$

- Program makes

$$O(\deg(f)^2) \text{ oracle calls,}$$

  none of whose inputs depends on another one's output, $\rightarrow$ parallel version

  Furthermore, program performs $\deg(f)^{2+o(1)}$ arithmetic operations in K

Characterization of Numerator/Denominator Evaluation Program

- Always evaluates the same associate of the numerator and denominator
$$\frac{f}{g} \quad \text{vs.} \quad \frac{2f}{2g}$$
- Construction of program is Monte-Carlo (might produce incorrect program with probability $\le \epsilon$), but the program itself is deterministic (this makes things much more difficult)
- Program contains positive integer constants of value bounded by
$$\frac{\deg(f)\deg(g)}{\epsilon}$$
- Program makes
$$O\big(\deg(f)\deg(g)(\deg(f)+\deg(g))\big) \text{ oracle calls,}$$
none of whose inputs depends on another one's output
and about the same amount of arithmetic operations (with fast extended Euclidean algorithm)

Homotopy Method for Solving $F(X) = 0$

Known: Solution to $G(X) = 0$	Wanted: Solution to $F(X) = 0$
$x_1(0)$ •	• $x_1(1)$
$x_2(0)$ •	• $x_2(1)$
$x_3(0)$ •	• $x_3(1)$
$\vdots$	$\vdots$
$x_n(0)$ •	• $x_n(1)$

Follow from $y = 0$ to $y = 1$ the solutions of

$$H(X(y)) = (1 - y)G(X(y)) + yF(X(y))$$

Our Homotopy

For $f(x_1, \ldots, x_n) \in \mathsf{K}[x_1, \ldots, x_n]$ consider

$$\bar{f}(X, Y) = f(X + b_1, Y(p_2 - a_2(p_1 - b_1) - b_2) + a_2 X + b_2, \\ \ldots, Y(p_n - a_n(p_1 - b_1) - b_n) + a_n X + b_n)$$

The field elements $a_2, \ldots, a_n, b_1, \ldots, b_n$ are pre-chosen ("known")
The field elements $p_1, \ldots, p_n$ are input

Notice: The polynomial $\bar{f}(X, 0)$ is independent of $p_1, \ldots, p_n$ and can be factored into

$$\bar{f}(X, 0) = \prod_{i=1}^{r} g_i(X)^{e_i}, \quad g_i(X) \in \mathsf{K}[X] \text{ irreducible}$$

By an *effective Hilbert Irreducibility Theorem* one can guarantee that the g_i are distinct images of the factors of f

$$g_i(X) = h_i(X + b_1, \ldots, a_n X + b_n),\ f(x_1, \ldots, x_n) = \prod_{i=1}^{r} h(x_1, \ldots, x_n)^{e_i}$$

$\rightarrow$ enters randomization

By *Hensel Lifting* we can follow the factorization to

$$\bar{f}(X, Y) = \prod_{i=1}^{r} \bar{h}_i(X, Y)^{e_i}$$

Lemma Needed for Numerator/Denominator Construction

Let

$$f(X), g(X) \in \mathsf{K}[X], \quad \mathrm{GCD}(f, g) = 1,$$
$$d = \deg(f), e = \deg(g), \quad g = x^e + \cdots$$

Given are distinct elements

$$i_1, \ldots, i_{d+e+1} \in \mathsf{K}, \quad \forall j : g(i_j) \neq 0$$

and a polynomial

$$h(X) \in \mathsf{K}[X] \text{ such that } \forall j : h(i_j) = \frac{f(i_j)}{g(i_j)}$$

Lemma: f appears as the first remainder of degree $\leq d$ in the Euclidean polynomial remainder sequence of

$$h(X) \text{ and } (X - i_1) \cdots (X - i_{d+e+1})$$

→ multiradix Padé approximation, can compute h by interpolation rather than power series approximation

Three Corollaries

Corollary 1: (Parallel Factorization)
For $\mathsf{K} = \mathbb{Q}$, we can compute in Monte Carlo $\mathcal{NC}$ all sparse factors of f of fixed degree and with no more than a given number t terms

Corollary 2: (Sparse Rational Interpolation)
Given a degree bound

$$b \geq \max(\deg(f), \deg(g))$$

and a bound t for the maximum number of non-zero terms in both f and g, we can in *Las Vegas* polynomial-time in b and t compute from a black box for f/g the sparse representations of f and g

Corollary 3: (Greatest Common Divisor)
From a black box for

$$f_1(x_1, \ldots, x_n), \ldots, f_r(x_1, \ldots, x_r) \in \mathsf{K}[x_1, \ldots, x_n]$$

we can efficiently produce a feasible program with oracle calls that allows to evaluate one and the same associate of

$$\mathrm{GCD}(f_1, \ldots, f_r)$$

Previous Results

Kaltofen [STOC 1986]: Could perform the same transformations from *straight-line programs* to *straight-line programs*

Required to transform individual straight-line instructions
→ new idea needed

Not every straight-line result generalized to black box model
e.g., Baur, Strassen's result on partial derivatives

Black Box Matrix Determinant Problem

Given a black box

$A \in \mathsf{K}^{n\times n}$

K a field of cardinality $\geq 50n^2 \log(n)$

compute the determinant of A.

For $\#K \geq 50n^2 \log(n)$, Doug Wiedemann (1986) constructs a Las Vegas randomized algorithm the computes $\mathrm{Det}(A)$ in

$$O(N) \text{ ``}A \times b \text{ steps''}$$

and

$$O(n^2 \log(n)) \text{ additional arithmetic operations.}$$

The algorithm requires $O(n \log(n))$ space.

Toeplitz Matrix × Vector Product

$$\begin{pmatrix} c & b & a \\ d & c & b \\ e & d & c \end{pmatrix} \times \begin{pmatrix} u \\ v \\ w \end{pmatrix} = \begin{pmatrix} cu + bv + aw \\ du + cv + bw \\ eu + dv + cw \end{pmatrix}$$

$$\begin{aligned} (a + bx + cx^2 + dx^3 + ex^4)\,(u + vx + wx^2) = \\ \vdots \\ +(cu + bv + aw)x^2 \\ +(du + cv + bw)x^3 \\ +(eu + dv + cw)x^4 \\ \vdots \end{aligned}$$

One can multiply a Toeplitz matrix into a vector in $O(n \log(n))$ arithmetic steps, using FFT based polynomial multiplication.