Evaluation, interpolation and multivariate multiplication

Éric Schost (ORCCA, UWO)
joint work with Joris van der Hoeven (LIX, X)
The big picture

We have very good algorithms for **univariate polynomials**

- arithmetic operations $+, \times, \div$
- computing in $K \to K[X]/f$

We would like good algorithms for **multivariate polynomials**

- one question (among others):
  efficient arithmetic in $K \to K[X_1, \ldots, X_n]/\langle f_1, \ldots, f_s \rangle$
- application: solving polynomial systems

**Objective:** quasi-linear time, no factor of the form $c^n$ in the cost
(in particular, expansion is forbidden – cf. Canny-Kaltofen-Lakshman’s sparse product)
This talk

No known solution, even for nice assumptions on \( \langle f_1, \ldots, f_s \rangle \) such as

- tdeg or lex Gröbner basis
- triangular set

**This talk:** algorithms for multiplication modulo zero-dimensional monomial ideals, i.e. multiplication of power series.
Main result

Input

- $M$ is a zero-dimensional monomial ideal in $K[X_1, \ldots, X_n]$
- $\delta_M = \dim_K K[X_1, \ldots, X_n]/M$
  this is the input and output size
- $\text{reg}_M = \max \deg(m)$, for $m$ a monomial not in $M$

Theorem

One multiplication modulo $M$ can be done in $O^\sim(\delta_M \text{reg}_M n)$ operations in $K$ (provided $K$ is large enough).

The factor $\text{reg}_M$ is the price to pay to use evaluation and interpolation techniques.
Truncation in total degree is determined by

\[ M = \langle X_1, \ldots, X_n \rangle^d = \langle \text{all monomials of degree } d \rangle. \]

Used in many forms of Hensel lifting; the support is a simplex.
Example: partial degree truncation

Truncation in partial degree is determined by

\[ M = \langle X_{d_1}^{d_1}, \ldots, X_{d_n}^{d_n} \rangle. \]

Used in a few (more marginal?) algorithms. The support of such series is a cube.
Previous work

1 variable

Truncation does not help for “optimal” algorithms (Fiduccia-Zalcstein)

Short product: improvement for algorithms like Karatsuba or Toom-Cook (Schönhage, Mulders, Hanrot-Zimmermann)

2 variables

Upper and lower bounds by Schönhage and Bläser

Total degree truncation

Quasi-linear cost for char $k = 0$ (Lecerf-S.)

Previous work by Griewank; refinements by van der Hoeven
Part I

Review: evaluation and interpolation in one variable
Polynomial multiplication

We let $M$ be such that polynomials of degree less than $n$ can be multiplied in $M(n)$ base ring operations.

Examples

- **Naive** $M(n) = O(n^2)$
- **Karatsuba** $M(n) = O(n^{\log_2(3)})$
- **Toom** $M(n) = O(n^{\log_3(5)})$
- **FFT over nice fields** $M(n) = O(n \log(n))$
- **FFT in general** $M(n) = O(n \log(n) \log \log(n))$

+ the assumptions of Chapter 9.
Already in one variable, the problem comes in many different flavors:

- **polynomial** or rational

- **dense** or sparse

- **Lagrange, Hermite, Birkhoff, …**

- **monomial basis, Newton basis, Bernstein basis, …**
  - \(1, X, \ldots, X^d\)
  - \(1, (X - x_0), \ldots, (X - x_0) \cdots (X - x_{d-1}), x_i\) pairwise distinct
Known results (cf. Chapter 10)

Monomial basis

Newton basis

$M(d) \log(d)$

$1.5M(d) \log(d)$

$2.5M(d) \log(d)$

Values

Borodin-Moenck, Bostan-Lecerf-S.

Bini-Pan, Bostan-S.

Slightly better results for arithmetic progressions (Gerhard); much better results for geometric progressions (Bluestein, Rabiner et al., Mersereau, Bostan-S.).
Known results (cf. Chapter 10)

Borodin-Moenck, Bostan-Lecerf-S.
Bini-Pan, Bostan-S.

Slightly better results for arithmetic progressions (Gerhard); much better results for geometric progressions (Bluestein, Rabiner et al., Mersereau, Bostan-S.).
Part II

Multivariate evaluation and interpolation
Previous work

In multivariate cases, there are many possible views on the problem (see Gasca-Sauer). From the point of view of feasibility, our interpolation problem will be simple.

Tensor product algorithms

- multidimensional FFT
- Pan (1994): simple evaluation / interpolation at a grid

Evaluation algorithms

- Nüsken, Ziegler (2004): bivariate evaluation at arbitrary points, subquadratic time
Choose an initial segment $T \subset \mathbb{N}^n$ for the partial order on $\mathbb{N}^n$

**Monomial support**

- $K[X_1, \ldots, X_n]_T = \{X_1^{i_1} \cdots X_n^{i_n} \mid (i_1, \ldots, i_n) \in T\}$
- bounding box: $d_1, \ldots, d_n$ such that
  \[ T \subset \{0, \ldots, d_1 - 1\} \times \cdots \times \{0, \ldots, d_n - 1\} \]

**Sample set**

- for $i \leq n$, pick pairwise distinct $x_{i,0}, \ldots, x_{i,d_i-1}$
- $V_T = \{(x_1, i_1, \ldots, x_n, i_n) \mid (i_1, \ldots, i_n) \in T\}$
- $V_T$ is contained in the grid
  \[ (x_{1,0}, \ldots, x_{1,d_1-1}) \times \cdots \times (x_{n,0}, \ldots, x_{1,d_n-1}) \]
Examples

Monomial support

Easy sample set

- choose \((x_{i,0}, \ldots, x_{i,d_i-1}) = (0, \ldots, d_i - 1)\)
- in this case \(V_T = T\)
- so we are evaluating polynomials supported on \(K[X_1, \ldots, X_n]_T\) at the set \(T\).

This is the sample set I will choose, even though using a geometric progression would be a bit better.
Previous work

Mora, Sauer (but also Macaulay, Hartshorne . . . )

- the evaluation map is invertible
  (by a Gröbner basis argument)

Werner, 1980

- interpolation in the Newton basis, using divided differences
Multivariate Newton basis

Polynomials in $K[X_1, \ldots, X_n]_T$ can be written on:

- the monomial basis $X_1^{i_1} \cdots X_n^{i_n}$
- the Newton basis $N_{i_1}(X_1) \cdots N_{i_n}(X_n)$, with
  
  $$N_{i_j}(X_j) = (X_j - x_{j,0}) \cdots (X_j - x_{j,i_j-1})$$

**Example:** $T$ is given as

With a grid based on $(0, 1, 2) \times (0, 1, 2)$, the bases are

- 1, $X_1$, $X_1^2$, $X_2$, $X_2X_1$, $X_2^2$
- 1, $X_1$, $X_1(X_1 - 1)$, $X_2$, $X_2X_1$, $X_2(X_2 - 1)$
Theorem: Changes of basis can be done in time

\[ O \left( \left( \frac{M(d_1) \log(d_1)}{d_1} + \cdots + \frac{M(d_n) \log(d_n)}{d_n} \right) |T| \right) \subset O^-(n|T|). \]

Done using a tensored version of the univariate algorithms.

Example: \( T \) is given as
Theorem: Changes of basis can be done in time
\[ O\left(\left(\frac{M(d_1) \log(d_1)}{d_1} + \cdots + \frac{M(d_n) \log(d_n)}{d_n}\right)|T|\right) \subset O^\sim(n|T|). \]

Done using a tensored version of the univariate algorithms.

Example: \( T \) is given as
**Theorem:** Changes of basis can be done in time

\[
O \left( \left( \frac{M(d_1) \log(d_1)}{d_1} + \cdots + \frac{M(d_n) \log(d_n)}{d_n} \right) |T| \right) \subset O^\sim(n|T|).
\]

Done using a tensored version of the univariate algorithms.

**Example:** $T$ is given as

![Diagram of the given example](image URL)
Conversions

**Theorem:** Changes of basis can be done in time

\[
O \left( \left( \frac{M(d_1) \log(d_1)}{d_1} + \cdots + \frac{M(d_n) \log(d_n)}{d_n} \right) |T| \right) \subset O^-(n|T|).
\]

Done using a tensored version of the univariate algorithms.

**Example:** \( T \) is given as

\[
M(d_1) \log(d_1) d_2
\]
Theorem: Changes of basis can be done in time

\[ O \left( \left( \frac{M(d_1) \log(d_1)}{d_1} + \cdots + \frac{M(d_n) \log(d_n)}{d_n} \right) |T| \right) \subset O^\sim(n|T|). \]

Done using a tensored version of the univariate algorithms.

Example: \( T \) is given as

\[ M(d_1) \log(d_1) d_2 \]
Theorem: Changes of basis can be done in time

\[ O \left( \left( \frac{M(d_1) \log(d_1)}{d_1} + \cdots + \frac{M(d_n) \log(d_n)}{d_n} \right) |T| \right) \subset O^\sim(n|T|). \]

Done using a tensored version of the univariate algorithms.

Example: \( T \) is given as

\[ M(d_1) \log(d_1) \, d_2 \]
Conversions

**Theorem:** Changes of basis can be done in time

\[
O \left( \left( \frac{M(d_1) \log(d_1)}{d_1} + \cdots + \frac{M(d_n) \log(d_n)}{d_n} \right) |T| \right) \subset O^-(n|T|).
\]

Done using a tensored version of the univariate algorithms.

**Example:** \( T \) is given as

\[
M(d_1) \log(d_1) d_2 + M(d_2) \log(d_2) d_1
\]
Theorem: Evaluation and interpolation can be done in time

$$O\left(\left(\frac{M(d_1) \log(d_1)}{d_1} + \cdots + \frac{M(d_n) \log(d_n)}{d_n}\right) |T|\right) \subset O^\sim(n|T|).$$

Done in the Newton basis, using a tensored version of the univariate algorithms.

Example:
Theorem: Evaluation and interpolation can be done in time

\[ O \left( \left( \frac{M(d_1) \log(d_1)}{d_1} + \ldots + \frac{M(d_n) \log(d_n)}{d_n} \right) |T| \right) \subset O^\sim(n|T|). \]

Done in the Newton basis, using a tensored version of the univariate algorithms.

Example:
Evaluation and interpolation

Theorem: Evaluation and interpolation can be done in time

\[ O \left( \left( \frac{M(d_1) \log(d_1)}{d_1} + \cdots + \frac{M(d_n) \log(d_n)}{d_n} \right) |T| \right) \subset O^*(n|T|). \]

Done in the Newton basis, using a tensored version of the univariate algorithms.

Example:

\[ p_{0,2} X_2 (X_2 - 1) \]

\[ v_{0,1} X_2 \quad v_{1,1} X_2 \]

\[ v_{0,0} \quad v_{1,0} \quad v_{2,0} \]
Evaluation and interpolation

**Theorem:** Evaluation and interpolation can be done in time

\[
O \left( \left( \frac{M(d_1) \log(d_1)}{d_1} + \cdots + \frac{M(d_n) \log(d_n)}{d_n} \right) |T| \right) \subset O^*(n|T|).
\]

Done in the Newton basis, using a tensored version of the univariate algorithms.

**Example:**

\[
\begin{align*}
&v_{0,2}X_2(X_2 - 1) \\
v_{0,1}X_2 & & v_{1,1}X_2 \\
v_{0,0} & & v_{1,0} & & v_{2,0}
\end{align*}
\]
Theorem: Evaluation and interpolation can be done in time

\[
O \left( \left( \frac{M(d_1) \log(d_1)}{d_1} + \cdots + \frac{M(d_n) \log(d_n)}{d_n} \right) |T| \right) \subset O^c(n|T|).
\]

Done in the Newton basis, using a tensored version of the univariate algorithms.

Example:
Theorem: Evaluation and interpolation can be done in time

\[ O \left( \left( \frac{M(d_1) \log(d_1)}{d_1} + \cdots + \frac{M(d_n) \log(d_n)}{d_n} \right) |T| \right) \subset O^\ast(n|T|). \]

Done in the Newton basis, using a tensored version of the univariate algorithms.

Example:
Evaluation and interpolation

**Theorem:** Evaluation and interpolation can be done in time

$$O \left( \left( \frac{M(d_1) \log(d_1)}{d_1} + \ldots + \frac{M(d_n) \log(d_n)}{d_n} \right) |T| \right) \subset O^-(n|T|).$$

Done in the Newton basis, using a tensored version of the univariate algorithms.

**Example:**

![Diagram](image-url)
**Theorem:** Evaluation and interpolation can be done in time

\[
O \left( \left( \frac{M(d_1) \log(d_1)}{d_1} + \cdots + \frac{M(d_n) \log(d_n)}{d_n} \right) |T| \right) \subset O^*(n|T|) .
\]

Done in the Newton basis, using a tensored version of the univariate algorithms.

**Example:**

\[
\begin{align*}
p_{0,2}X_2(X_2 - 1) \\
p_{0,1}X_2 & \quad p_{1,1}X_1X_2 \\
p_{0,0} & \quad p_{1,0}X_1 & \quad p_{2,0}X_1(X_1 - 1)
\end{align*}
\]
Theorem: Evaluation and interpolation can be done in time
\[ O \left( \left( \frac{M(d_1) \log(d_1)}{d_1} + \cdots + \frac{M(d_n) \log(d_n)}{d_n} \right) |T| \right) \subset O^*(n|T|). \]

Done in the Newton basis, using a tensored version of the univariate algorithms.

Example:

To evaluate \( P \) at \((1, 1)\), we just need the coefficients “under” \((1, 1)\)
Part III

Power series multiplication
Review

Setup

- \( M \): zero-dimensional monomial ideal in \( K[X_1, \ldots, X_n] \)
- \( T \): exponents of the monomials not in \( M \)
- \( \delta_M = \dim K[X_1, \ldots, X_n]/M = |T| \)
  this is the input and output size
- \( \text{reg}_M = \max \deg(m), \text{for } m \text{ not in } M \)

Example

\[ M = \langle X_1^2, X_1X_2, X_2^2 \rangle \]

\[ T = \{(0, 0), (1, 0), (0, 1)\} \]

\[ K[X_1, X_2]_T \text{ generated by } 1, X_1, X_2 \]

\[ \delta_M = 3, \text{reg}_M = 1 \]
Theorem (2005). One multiplication modulo $M$ can be done using

- $O(\text{reg}_M)$ evaluations / interpolations at $T$ of polynomials in $K[X_1, \ldots, X_n]_T$
- $\delta_M$ univariate power series products at precision $O(\text{reg}_M)$

(at the time, I did not know how to do the evaluation / interpolation)

Ingredient: APA-algorithms (as in fast matrix multiplication)

- Bini-Capovani-Romani-Lotti: floating-point products
- Bini: relation to exact computations
- Bini-Lotti-Romani, Schönhage: multiplication modulo $X^2$
The algorithm on an example to multiply modulo
\[ \langle X_1^2, X_1X_2, X_2^2 \rangle \]
The algorithm on an example

to multiply modulo \( \langle X_1^2, X_1X_2, X_2^2 \rangle \)
multiply modulo \( \langle X_1(X_1 - \varepsilon), X_1X_2, X_2(X_2 - \varepsilon) \rangle \)

\[ \varepsilon \]
\[ 0 \]
\[ 0 \]
\[ \varepsilon \]
The algorithm on an example

\[ \langle X_1^2, X_1X_2, X_2^2 \rangle \]

to multiply modulo

\[ \langle X_1(X_1 - \varepsilon), X_1X_2, X_2(X_2 - \varepsilon) \rangle \]

multiply modulo

\[ \langle X_1, X_1X_2, X_2(\varepsilon) \rangle \]

let \( \varepsilon = 0 \)
The algorithm on an example

to multiply modulo $\langle X_1^2, X_1X_2, X_2^2 \rangle$
multiply modulo $\langle X_1(X_1 - \varepsilon), X_1X_2, X_2(X_2 - \varepsilon) \rangle$
(by evaluation / interpolation)

let $\varepsilon = 0$
The algorithm on an example

to multiply modulo $\langle X^2, X_1X_2, X^2_2 \rangle$

multiply modulo $\langle X_1(X_1 - \varepsilon), X_1X_2, X_2(X_2 - \varepsilon) \rangle$
(by evaluation / interpolation)

let $\varepsilon = 0$

• in general:

  in every member $x_1^{i_1}x_2^{i_2}...x_n^{i_n}$ of the basis of $P$ change $x_i^{p_i}(i = 1, 2, ..., n)$ to $x_i(x_i - 1)...(x_i - p_i + 1)$.

• do evaluation / interpolation, with power series coefficients
• correctness (again) from Mora-Sauer-...’s argument
• precision in $\varepsilon = 2 \times$ regularity
1. The factor $\text{reg}_M$ is annoying
   - I should allow expansion: allows product modulo $\langle X_1^d, \ldots, X_n^d \rangle$ in time $O(\delta 4^{\sqrt{\log \delta}})$
   - but still, does not solve

2. Computing modulo a zero-dimensional Gröbner basis?
   - initial ideals obtained through one-parameter deformations
   - homotopy techniques?