Computer Algebra's Dirty Little Secret

Stephen M. Watt
University of Western Ontario

ORCCA
Ontario Research Centre for Computer Algebra

TRICS Seminar, UWO CSD, 5 November 2008

Western

What This Talk Is About

- Computers and mathematics
- Computer algebra and symbolic computation
- What computer algebra systems *can* do
- What computer algebra systems *cannot* do
- How to get them to do it

Warning!

This talk is X-rated.

It is intended for a mathematically mature audience.

X, other variables and graphical content may appear.

Viewer discretion is advised.

Computers doing mathematics

- Numerical computing `sin(1.02)**2`
- Symbolic computing `diff(sin(x^n)^m, x)`
- Math communication $\frac{\partial}{\partial x}\sin^m(x^n)$
- Automated theorem proving, conjecture generation, ...

Mathematics extending computing

- Algebraic notation `a*d - b*c`
- Arrays
- Garbage collection
- Operator overloading
- Templates and generic programming
- Functional programming, ...

Some of the things I do

- Develop algorithms
  - Algebraic algorithms
  - Symbolic-numeric algorithms

    ```
    gcd(x + 1, x^2 + 2.01*x + .99)
    ```

- Study how to build computer algebra systems
  - Memory management
  - Higher-order type systems
  - Optimizing compilers
- Mathematical knowledge management
  - Representation of mathematical objects
  - Mathematical handwriting recognition

What do computer algebra systems do?

Choose the best answer:

(a) Manipulate expressions and equations.

(b) Do calculus homework.

(c) Give general formulas as answers.

(d) Model industrial mathematical problems.

(e) All of the above.

A Maple session

Maple 11 - [Untitled (1) - [Server 1]]

File Edit View Insert Format Spreadsheet Window Help

```
> limit((2*sin(x)-sin(2*x))/(x - sin(x)), x = 0);
```

$$6$$

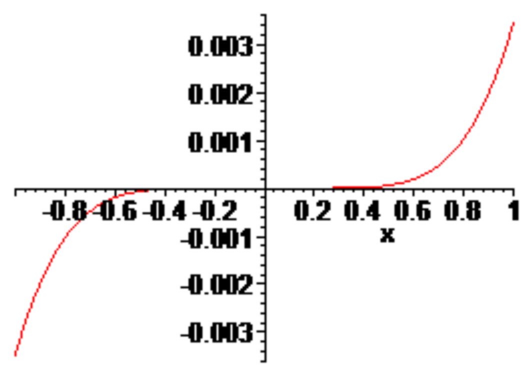
```
> int(sin(x)^5 + exp(-x^2/b), x = 0..Pi);
```

$$\frac{16}{15}+\frac{1}{2}\sqrt{\pi}\,\sqrt{b}\,\mathrm{erf}\left(\frac{\pi}{\sqrt{b}}\right)$$

```
> factor(x^15+1);
```

$$(1+x)(1-x+x^2-x^3+x^4)(1-x+x^2)(1+x-x^3-x^4-x^5+x^7+x^8)$$

```
> plot(sin(tanh(x)) - tanh(sin(x)), x = -1..1);
```

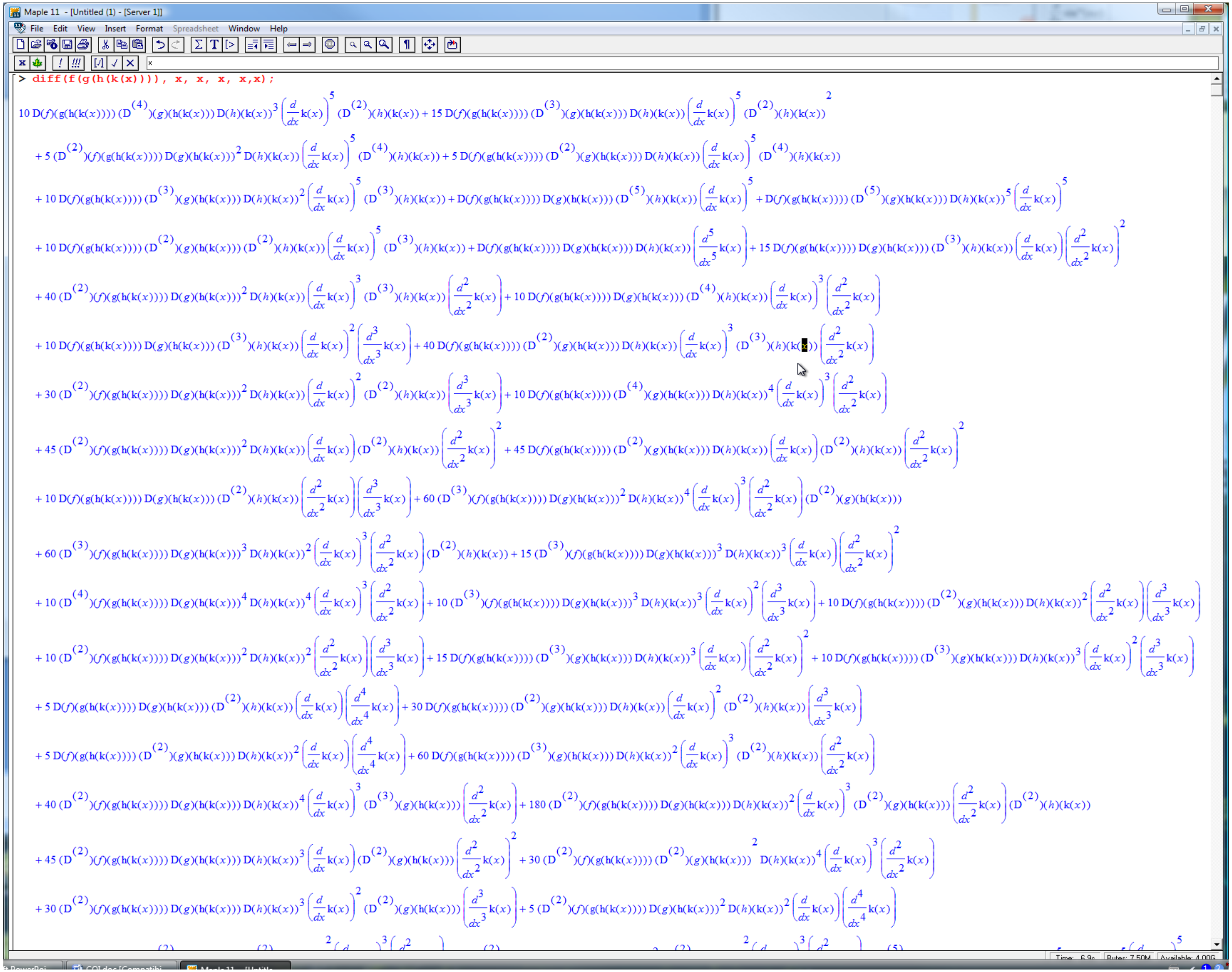



```
> taylor(sin(tanh(x)) - tanh(sin(x)), x = 0, 10);
```

$$\frac{1}{90}x^7-\frac{13}{756}x^9+O(x^{10})$$

Time: 6.6s Bytes: 7.44M Available: 4.00G

Another session (Be careful what you ask!)



Computer Algebra vs Symbolic Computation

- Computer Algebra
  - **Arithmetic** on defined algebraic structures
  - Polynomials, matrices, algebraic functions,…
  - May involve symbols parameters, indeterminates

- Symbolic computation
  - **Transformation** of expression trees
  - Symbols for opns (“+”, “sin”), variables, consts
  - Simplification, expression equivalence

Computer Algebra vs Symbolic Computation

- Computer Algebra
  - Well-defined semantics
  - Compose constructions
  - Algebraic algorithms

- Symbolic computation
  - Alternative forms (factored, expanded, Horner...)
  - Working in partially-specified domains
  - Working symbolically

Computer Algebra

$\mathbb{Q}[\alpha]/\langle\alpha^2 + 3\alpha + 7\rangle$:

$$\frac{1}{5\alpha^2 + 2\alpha - 7} \to \alpha^2 + \frac{3940}{1309}\alpha + \frac{9160}{1309}$$

Symbolic Computation

$$
\begin{aligned}
ax^2 + bx + c &= a\left(x^2 + \frac{bx}{a} + \frac{b^2}{4a^2}\right) - a\left(\frac{b^2}{4a^2} - \frac{c}{a}\right) \\
&= a\left(x + \frac{b}{2a}\right)^2 - a\left(\frac{\sqrt{b^2 - 4ac}}{2a}\right)^2 \\
&= a\left(x - \frac{-b + \sqrt{b^2 - 4ac}}{2a}\right) \cdot \left(x - \frac{-b - \sqrt{b^2 - 4ac}}{2a}\right)
\end{aligned}
$$

Dirty Little Secret 1

- Symbolic mathematics systems have become increasingly "algebratized" over the past 20 years.
- **This is good**:
  Spectacular algorithmic advances allow us to solve problems not even dreamed of in the 80s.
- **This is bad**:
  We are no closer to handling simple problems that are outside the classical algebraic domains.

Algebraic Algorithms

- Problems are solved using methods vastly different than the ones you learned in school or university.

- Examples:
  - Polynomial multiplication: DFT
  - Integration: Risch algorithm
  - Factorization: Cantor-Zassenhaus

Polynomial Multiplication

- Two polynomials

$$P = 3x^3 + 4x^2 - x + 3 \qquad Q = x^3 - 2x^2 + x + 7$$

- School method

$$\begin{array}{ccccccc}
 & & & 3x^3 & + 4x^2 & - x & + 3 \\
 & & \times & x^3 & - 2x^2 & + x & + 7 \\
\hline
 & & & 21x^3 & +28x^2 & -7x & +21 \\
 & & 3x^4 & + 4x^3 & - x^2 & +3x & \\
 & -6x^5 & -8x^4 & + 2x^3 & - 6x^2 & & \\
3x^6 & +4x^5 & - x^4 & + 3x^3 & & & \\
\hline
3x^6 & -2x^5 & -6x^4 & +30x^3 & +21x^2 & -4x & +21
\end{array}$$

- Multiplication costs $O(d^2)$

Polynomial Multiplication

- Point-wise Value Method

Evaluate

$$\begin{aligned} P &= \{(-3,-39),\ (-2,-\ 3),\ (-1,\ 5),\ (0,\ 3),\ (1,\ 9),\ (2,\ 41),\ (3,117)\} \\ Q &= \{(-3,-41),\ (-2,-11),\ (-1,\ 3),\ (0,\ 7),\ (1,\ 7),\ (2,\ \ 9),\ (3,\ 19)\} \end{aligned}$$

$$PQ = \{(-3,1599),\ (-2,\ \ 33),\ (-1,15),\ (0,21),\ (1,63),\ (2,369),\ (3,2223)\}$$

Interpolate

- DFT trick: evaluate at "roots of unity"

$$\omega^0, \omega^1, \omega^2, \ldots$$

$$\omega = \sqrt[n]{1} \qquad \text{like} \quad \exp(2\pi i/n) \quad \text{over} \quad \mathbb{C}, \quad \text{but over } \mathbb{F}_p$$

- Multiplication now *O*(*d* log *d*)

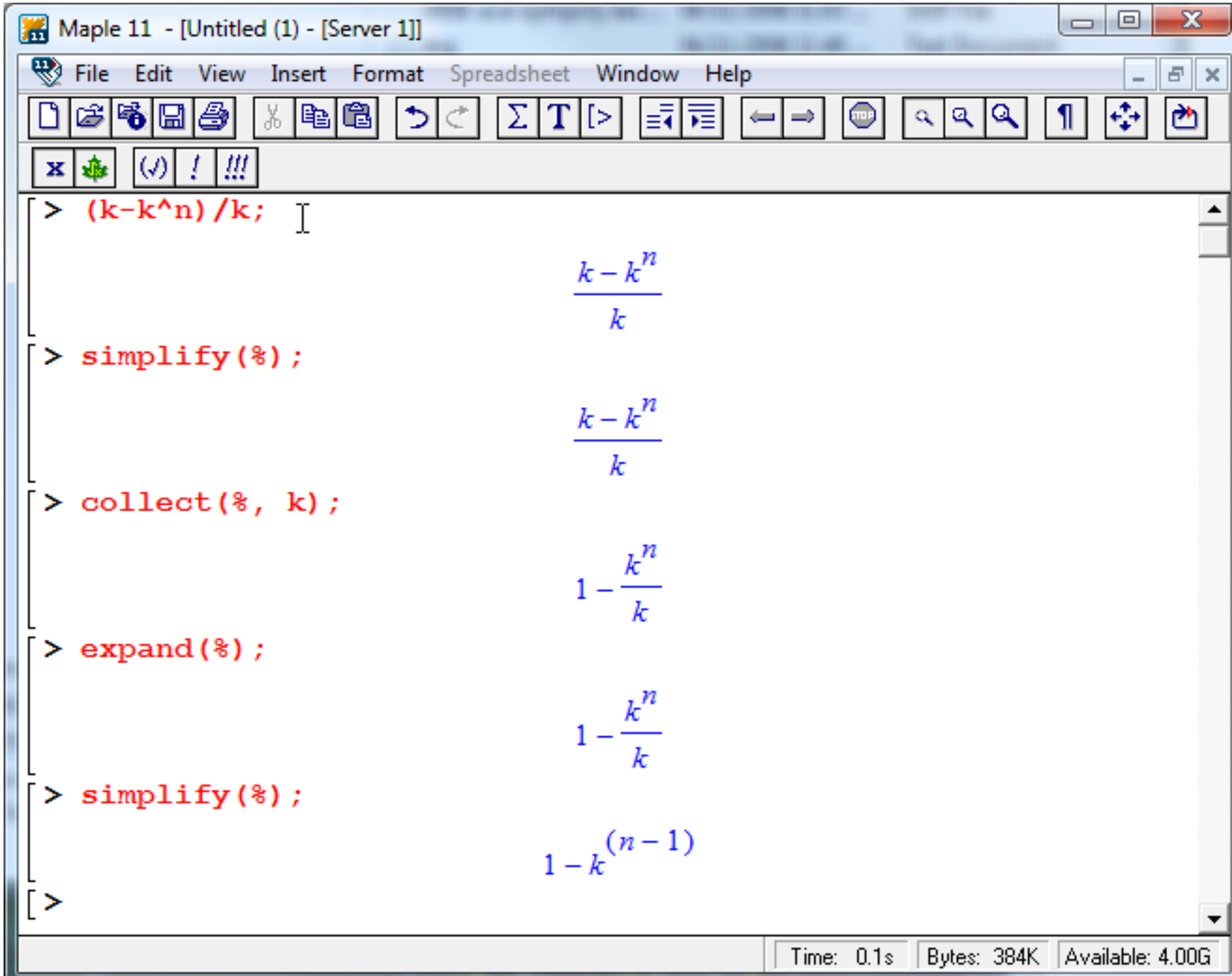
Dirty Little Secret 2

- Computer math systems are presently very bad at computing with *symbolic* values.

- Polynomial of degree d.
- Field of characteristic p.
- Space of dimension n.

- Computer algebra has a hard time representing.
- Symbolic computation has few algorithms.

What we have


Maple 11 - [Untitled (1) - [Server 1]]
File Edit View Insert Format Spreadsheet Window Help
> (k-k^n)/k;
$\frac{k-k^n}{k}$
> simplify(%);
$\frac{k-k^n}{k}$
> collect(%, k);
$1-\frac{k^n}{k}$
> expand(%);
$1-\frac{k^n}{k}$
> simplify(%);
$1-k^{(n-1)}$
>
Time: 0.1s
Bytes: 384K
Available: 4.00G

What we want

$$x^{2n} - y^{2m} = (x^n + y^m)(x^n - y^m)$$

$$x^{n^2+3n} - y^{2m} = \left(x^{n(n+3)/2} + y^m\right)\left(x^{n(n+3)/2} - y^m\right)$$

$$16^n - 81^m = (2^n - 3^m)(2^n + 3^m)(2^{2n} + 3^{2m})$$

Bringing Approaches Together

- Can *algebratize* symbolic computation
  (initial algebras, free algebras, adjoint functors)
- Can *symbolicize* algebraic computation
  (more varied algebraic structures)

- Amount to the same thing
- Need *algorithms* for these formal structures.

Two Steps

- Symbolic polynomials:

  "polynomials" in which the exponents are integer-valued functions.

$$x^{n(n+3)/2} + y^m$$

- Symbolic matrices:

  "matrices" in which the internal structure are of symbolic size.

- We work with these easily by hand but CAS fail.

$$\begin{bmatrix} a_n & & 0 \\ \vdots & \ddots & \\ a_0 & & a_n \\ & \ddots & \vdots \\ 0 & & a_0 \end{bmatrix}$$

Symbolic Polynomials

- Arise frequently in practice.
- Wish to perform as many of the usual polynomial operations as possible.
- Model as
  - monomials with integer-valued polynomials as exponents, and
  - finite combinations of "+" and "×".

Symbolic Polynomials

These are OK

$$x^n - y^m$$

$$(nx^{n^2-2m+2} - x^2y^k) \times (x^n + 1)$$

These are not *OK*

$$x^{\binom{n}{m}} - x^{\mathrm{lcm}(n,m)} + 1$$

$$(x-1) \times \sum_{i=0}^{n} x^i$$

Integer-Valued Polynomials

(OK to ignore if you don't like math)

For an integral domain D with quotient field K, univariate integer-valued polynomials may be defined as

$$\mathrm{Int}_{[X]}(D) = \{f(X) \mid f(X) \in K[X] \text{ and } f(a) \in D, \text{ for all } a \in D\}$$

(Note: In standard notation, the $[X]$ is not written.)

- Example: $\frac{1}{2}n^2 - \frac{1}{2}n \in \mathrm{Int}_{[n]}(\mathbb{Z})$.
- We make use of the obvious multivariate generalization, which we denote $\mathrm{Int}_{[n_1,\ldots,n_p]}(D)$.

Symbolic Polynomials

(also ok to ignore if you don't like math – you've got the idea already)

Definition: *The ring of* symbolic polynomials *in* $x_1, ..., x_v$ *with exponents in* $n_1, ..., n_p$ *over the coefficient ring* R *is the ring consisting of finite sums of the form*

$$\sum_i k_i x_1^{e_{i1}} x_2^{e_{i2}} \cdots x_v^{e_{iv}}$$

where $k_i \in R$ *and* $e_{ij} \in \mathrm{Int}_{[n_1,...,n_p]}(\mathbb{Z})$. *Multiplication is defined by*

$$k_1 x_1^{e_{11}} \cdots x_v^{e_{1v}} \quad \times \quad k_2 x_1^{e_{21}} \cdots x_v^{e_{2v}} = k_1 k_2 x_1^{e_{11}+e_{21}} \cdots x_v^{e_{1v}+e_{2v}}$$

- Write $R[n_1, ..., n_p; x_1, ..., x_v]$.
- $R[; x_1, ..., x_v] \cong R[x_1, ..., x_v, x_1^{-1}, ..., x_v^{-1}]$.
- $R[n_1, ..., n_p; x_1, ..., x_v]$ is the group ring $R[\ \mathrm{Int}_{[n_1,...,n_p]}(\mathbb{Z})^v\ ]$, identifying ${x_1}^{e_1} {x_2}^{e_2} \cdots {x_v}^{e_v} \cong (e_1, \ldots, e_v) \in \mathrm{Int}_{[n_1,...,n_p]}(\mathbb{Z})^v$

Why Insist on Integer-Valued Exponents?

- Otherwise they are not really a symbolic model of polynomials,
- they do not behave like polynomials when exponent vars are evaluated,
- factorizations would never happen

$$x^{n^2+n} - 1,$$

- or they would not have a well-defined end

$$(x^{\frac{n^2+n}{2}} + 1)(x^{\frac{n^2+n}{4}} + 1)(x^{\frac{n^2+n}{8}} + 1)\cdots(x^{\frac{n^2+n}{2^k}} + 1)(x^{\frac{n^2+n}{2^k}} - 1).$$

- So integer valued polnomial exponents are more useful in practice and for the theory.

Symbolic Polynomials

- Algorithms for arithmetic (+, ×) straightforward.

- Q: Can we do more interesting things like factorize or take GCDs of symbolic polynomials?

- A: Yes!

Multiplicative Structure

Theorem: $\mathbb{Z}[n_1, ..., n_p; x_1, ..., x_v]$ *is a UFD.*

- First consider the case when exponents are in $\mathbb{Z}[n_1, ... n_p]$.

  We remove the exponent variables inductively.

  $x, x^n, x^{n^2}, ...$ are algebraically independent so we introduce the new variables x_{kj}

$$x_k^{e_{ik}} = x_k^{\sum_j h_{ij} n_1^j} = \prod_j \left(x_k^{{n_1}^j} \right)^{h_{ij}} = \prod_j {x_{kj}}^{h_{ij}}, \qquad h_{ij} \in \mathbb{Z}[n_2, ..., n_p].$$

Multiplicative Structure

Theorem: $\mathbb{Z}[n_1, ..., n_p; x_1, ..., x_v]$ *is a UFD.*

- With exponents in $\text{Int}_{[n_1,...,n_p]}(\mathbb{Z})$, care must be taken to find the "fixed divisors" of the exponent polynomials.

  For example, $n(n-1)$ is always even.

  This implies the factorization

$$x^2 - y^{n^2-n} = (x - y^{n(n-1)/2})(x + y^{n(n-1)/2})$$

Integer-valued Polynomials and Fixed Divisors

Property: *If f is a polynomial in* $\mathrm{Int}_{[n_1,\ldots,n_p]}(\mathbb{Z}) \subset \mathbb{Q}[n_1,\ldots,n_p]$, *then when f is written in the basis $\binom{n_1}{i_1}\cdots\binom{n_p}{i_p}$, its coefficients are integers*

Note $\binom{n}{j} = n(n-1)\cdots(n-j+1)/j!$.

Property: *If f is a polynomial in $\mathbb{Z}[n_1,\ldots,n_p]$, then the largest fixed divisor of f is the* gcd *of the coefficients of f written in the binomial basis.*

Algorithm Family 1: The Extension Method

- Put exponents in basis $\binom{n_i}{j}$.
- Map to new variables using $\gamma : x_k^{\binom{n_1}{i_1}\cdots\binom{n_p}{i_p}} \mapsto X_{ki_1\ldots i_p}$.
- Solve problem over $\mathbb{Z}[X_{10\ldots0}, \ldots X_{vp\ldots p}]$.
- Map back.

"Solve problem" might be "compute GCD" , "factorize", etc.

Example

$$p = 8x^{n^2+6n+4+m^2-m} - 2x^{2n^2+7n+2mn}y^{n^2+3n} - 3x^{n^2+3n+2mn}y^{n^2+3n} + 12x^{4+m^2-m+2n}$$

$$q = 4x^{n^2+4n+m^2+6m} - 28x^{n^2+8n+m^2+6m+2}y^{4n^2-4n} + 2x^{n^2+4n} - 14x^{n^2+8n+2}y^{4n^2-4n} + 6x^{m^2+6m} - 42x^{m^2+6m+4n+2}y^{4n^2-4n} - 21y^{4n^2-4n}x^{4n+2} + 3$$

GCD will have exponents of *x* as polynomials in *m* and *n* of maximum degree 2.

$$\left\{ \binom{n}{i}\binom{m}{j} \,\middle|\, 0 \le i+j \le 2 \right\} = \left\{ 1, n, m, \frac{n(n-1)}{2}, nm, \frac{m(m-1)}{2} \right\}$$

GCD will have exponents of *y* as polynomials in *n* of maximum degree 2.

$$\left\{ \binom{n}{i} \,\middle|\, 0 \le i \le 2 \right\} = \left\{ 1, n, \frac{n(n-1)}{2} \right\}$$

Example (continued)

Make the change of variables:

$$\gamma = \{x \mapsto A,\ x^n \mapsto B,\ x^{\binom{n}{2}} \mapsto C,\ x^m \mapsto D,\ x^{mn} \mapsto E,\ x^{\binom{m}{2}} \mapsto F,$$
$$y \mapsto G,\ y^n \mapsto H,\ y^{\binom{n}{2}} \mapsto I\}$$

$$p = 8A^4B^7C^2F^2 - 2B^9C^4E^2H^4I^2 - 3B^4C^2E^2H^4I^2 + 12A^4B^2F^2$$

$$q = 4B^5C^2D^7F^2 - 28A^2B^9C^2D^7F^2I^8 + 2B^5C^2 - 14A^2B^9C^2I^8$$
$$+ 6D^7F^2 - 42A^2B^4D^7F^2I^8 - 21A^2B^4I^8 + 3.$$

GCD(p, q) and factorization of p are:

$$g = 2B^5C^2 + 3$$

$$p = B^2 \times (2B^5C^2 + 3) \times (2A^2F - BCEIH^2) \times (2A^2F + BCEIH^2)$$

Example (continued)

Apply the inverse substitution:

$$
\begin{aligned}
g &= 2x^{n^2+4n} + 3 \\
p &= x^{2n} \times \left(2x^{n^2+4n} + 3\right) \\
&\quad \times \left(2x^{1/2\,m^2-1/2\,m+2} - x^{1/2\,n^2+mn+1/2\,n}y^{1/2\,n^2+3/2\,n}\right) \\
&\quad \times \left(2x^{1/2\,m^2-1/2\,m+2} + x^{1/2\,n^2+mn+1/2\,n}y^{1/2\,n^2+3/2\,n}\right).
\end{aligned}
$$

A Problem:

How to Factor $x^{m^{1000}n^{1000}+mn} - 1$

- To handle fixed divisors, extension method converts to binomial basis.
- Gives a polynomial in a million variables.

- Instead, compute the maximum possible fixed divisor C_k over the primitive parts of all exponent polynomials of x_k.
- Change of variables $x_k \to X_k^{C_k}$.

- Exponent polynomials retain their sparsity.
- Number of new variables is at most linear in the size of the input.

Other Symbolic Polynomial Algorithms

- Algorithm Family 2:
  Evaluation/interpolation of exponents.
  (Interpolate symmetric polynomials.)

- Sparse evaluation/interpolation of exponents.

- Exponents on coefficients.

- Exponent variables as base variables.

- Functional decomposition of symbolic polynomials.
  If $f = g \circ h$, find g and h.

Symbolic Matrix Arithmetic

- Earlier work by Alan Sexton and Volker Sorge on determining expressions for regions in matrices, e.g.

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} & & \mathbf{0} & \\ & \ddots & \vdots & & & \\ & & a_{nn} & & & \\ & & & b_{11} & \cdots & b_{1m} \\ & & & & \ddots & \vdots \\ \mathbf{0} & & & & & b_{mm} \end{bmatrix}$$

  where a_{ij} is one function of i and j and b_{ij} is another, and n and m are unknowns.

- For them a general matrix entry is a set of expression/condition pairs.

$$A_{ij} = \begin{cases} 0 & i > j \\ a_{ij} & i \leq j \wedge j \leq n \\ 0 & i \leq n \wedge j > n \\ b_{i-n\ j-n} & i > n \wedge i \leq j \end{cases}$$

- Collaboration: Do arithmetic on these objects

The usual problem with piecewise fns

- The usual problem is that the number of conditions multiply on each arithmetic operation.
  q conditions on each of n operands gives q^n cases.

- Example:

$$U = [u_1, u_2, ..., u_h, u'_1, u'_2, ..., u'_{n-h}]$$

$$V = [v_1, v_2, ..., v_k, v'_1, v'_2, ..., v'_{n-k}]$$

  The general entry for $U + V$ is

$$\begin{array}{ll} u_i + v_i & i \leq \min(h, k) \\ u_i + v'_i & k < i \leq h \\ u'_i + v_i & h < i \leq k \\ u'_i + v'_i & \max(h, k) < i \end{array}$$

- At least one of the two middle cases will be empty, but we can't tell which because h and k are unknown.

- For a sum of n such vectors, we have 2^n cases. Intractable.

Use Basis Functions

- **Definition**

$$\xi(i,y,z) = \begin{cases} 1 & \text{if } y \le i < z \\ -1 & \text{if } z \le i < y \\ 0 & \text{otherwise} \end{cases}$$

- **Properties**

$$\begin{aligned} \xi(i,y,y) &= 0 \\ \xi(i,y,z) &= -\xi(i,z,y) \\ \xi(i,y,x) + \xi(i,x,z) &= \xi(i,y,z) \end{aligned}$$

Then Add General Terms

- Addition, based on general terms:

$$U_i + V_i = \xi(i,1,h) \times u_i + \xi(i,h,n) \times u'_{i-h+1} + \xi(i,1,k) \times v_i + \xi(i,k,n) \times v'_{i-k+1}$$

- Suppose $h < k$, then the above yields a sum of three terms:

$$U_i + V_i = \xi(i,1,h) \times (u_i + v_i) + \xi(i,h,k) \times (u'_i + v_i) + \xi(i,k,n) \times (u'_i + v'_i)$$

- But due to the properties of $\xi(i,j,k)$, this is exactly the same as the expression for $k < h$.

- So adding n vectors requires $n+1$ terms with n terms each instead of 2^n cases with n terms each.

- Same works for matrices and with more complex internal structure.

Conclusions

- Computer algebra researchers should realize that their spectacular success hides an equally spectacular failure.

- There is a practically important, and theoretically rich middle ground between "computer algebra" and "symbolic computation."

- We can and should explore this by
  - 1. Creating new usefull well-defined structures.
  - 2. Inventing algorithms for these structures.
  - 3. Getting our math software to handle them.