How fast can we multiply and divide polynomials?

JSSAC Special Interest Group on Computer Education,

December 19th, 2009.

Michael Monagan
Center for Experimental and Constructive Mathematics,
Simon Fraser University,
Vancouver, British Columbia,
CANADA.

Joint work with Roman Pearce.

Examples of what CAS can do.

> factor$(2\,x^3 + x^2y^2 - x^3y + x^2 - 5\,yx - 3\,y^3 + 3\,y^2x - 3\,y - 1)$;

> solve$(\{x^2 + y^2 + z^2 - 4\ ,\ x\,y\,z + 2\ ,\ x\,y + z^3 - 1\})$;

> Determinant$\left(\begin{bmatrix} t & 1-2\,t & 1 \\ t^2 & t & 1 \\ 1+t & 1 & 1+t+t^2 \end{bmatrix}\right)$;

> $\int x^2\ln(x)\mathrm{e}^{-x} + (1-x)\ln(x)\mathrm{e}^{-x} - 2x\mathrm{e}^{-x}\,\mathrm{d}x$;

Examples of what CAS can do.

> $\mathrm{factor}(2\,x^3 + x^2y^2 - x^3y + x^2 - 5\,yx - 3\,y^3 + 3\,y^2x - 3\,y - 1);$

> $\mathrm{solve}(\{x^2 + y^2 + z^2 - 4\ ,\ x\,y\,z + 2\ ,\ x\,y + z^3 - 1\});$

> $\mathrm{Determinant}\left(\begin{bmatrix} t & 1-2\,t & 1 \\ t^2 & t & 1 \\ 1+t & 1 & 1+t+t^2 \end{bmatrix}\right);$

> $\int x^2\,\ln(x)\mathrm{e}^{-x} + (1-x)\,\ln(x)\mathrm{e}^{-x} - 2x\mathrm{e}^{-x}\,\mathrm{d}x;$

Risch

$\mathrm{e}^{-x} \to \theta_1$
$\ln x \to \theta_2$

$$\int \overbrace{x^2\theta_2\theta_1 + (1-x)\theta_2\theta_1 - 2x\theta_1}^{\text{a polynomial}}\,\mathrm{d}x \quad \text{where} \quad \begin{matrix}\theta_1' = -\theta_1 \\ \theta_2' = 1/x.\end{matrix}$$

Polynomials are the key!

Talk Outline:

- Maple demo on solving polynomials.
- How do CAS represent polynomials?
- How do CAS multiply and divide polynomials?
- Our new representation and algorithms.
- Benchmarks comparing us with CAS.
- Maple demo of the graph theory package.

How do CAS *represent* polynomials?

Recursive and distributed polynomial representations.

The distributed represention: monomials $x^i y^j z^k$ are sorted in *lexicographical order* (Magma, Mathematica):

$$f = -6x^3 + 9xy^3z - 8xy^2z + 7y^2z^2 + 5$$

or *graded lex order* (Singular, Maple 15):

$$f = 9xy^3z - 8xy^2z + 7y^2z^2 - 6x^3 + 5.$$

Key property: if X, Y, Z are monomials then $Y > Z \implies XY > XZ$.

Recursive and distributed polynomial representations.

The distributed represention: monomials $x^i y^j z^k$ are sorted in *lexicographical order* (Magma, Mathematica):

$$f = -6x^3 + 9xy^3z - 8xy^2z + 7y^2z^2 + 5$$

or *graded lex order* (Singular, Maple 15):

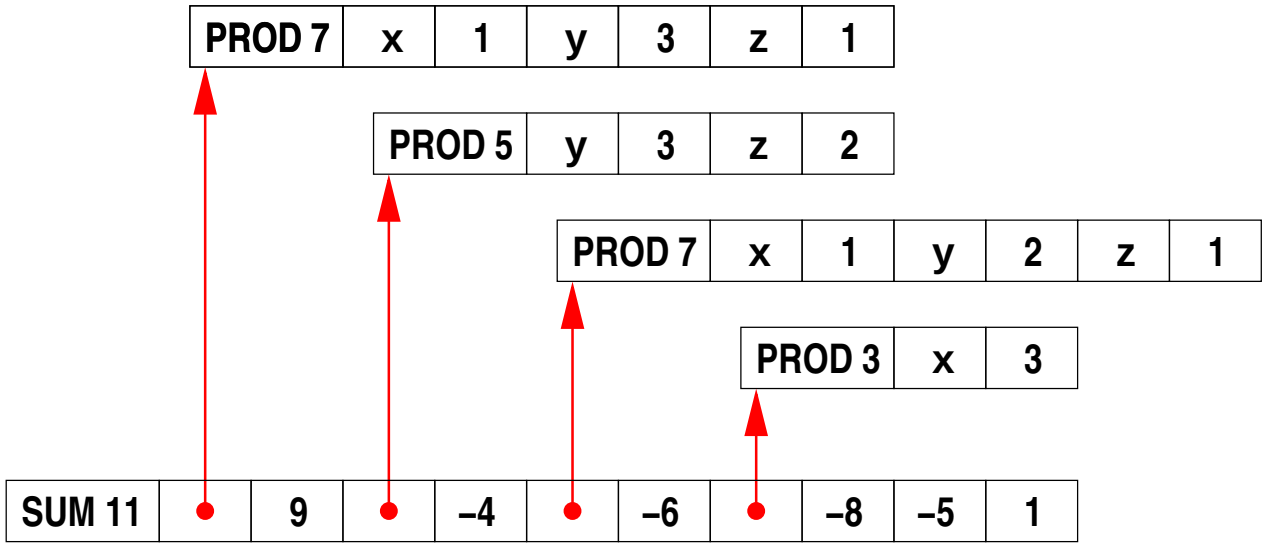
$$f = 9xy^3z - 8xy^2z + 7y^2z^2 - 6x^3 + 5.$$

Key property: if X, Y, Z are monomials then $Y > Z \implies XY > XZ$.

The recursive representation (Macsyma, REDUCE, Derive, Pari):

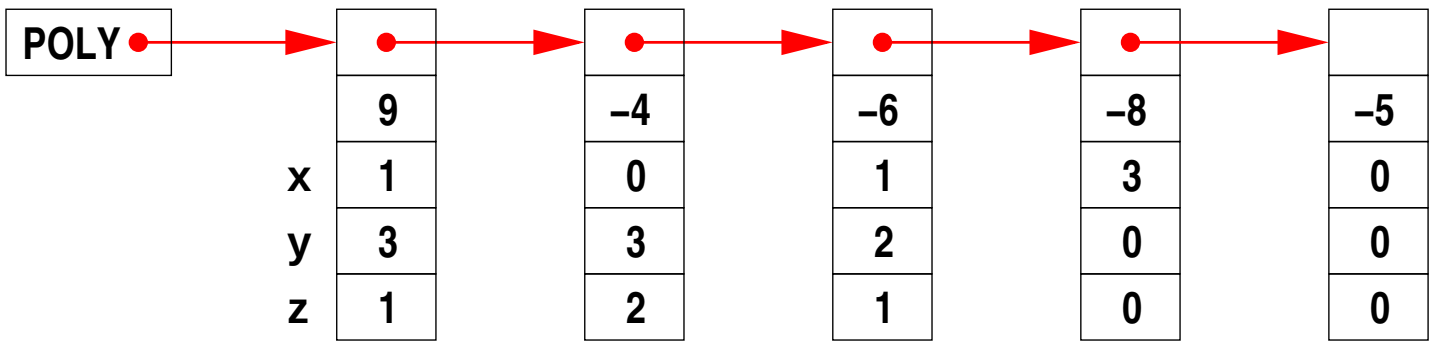
$$f = (-6)x^3 + ((9z)y^3 + (-8z)y^2)x^1 + ((7z^2)y^2 + 5y^0)x^0.$$

Maple's sum of products representation.

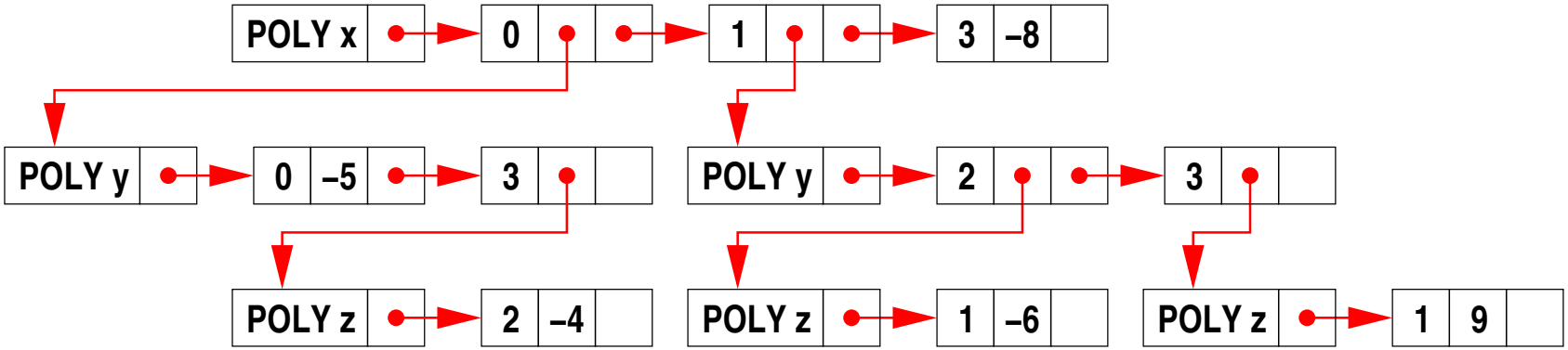


$$9xy^3z - 4y^3z^2 - 6xy^2z - 8x^3 - 5$$

Singular's distributed representation.

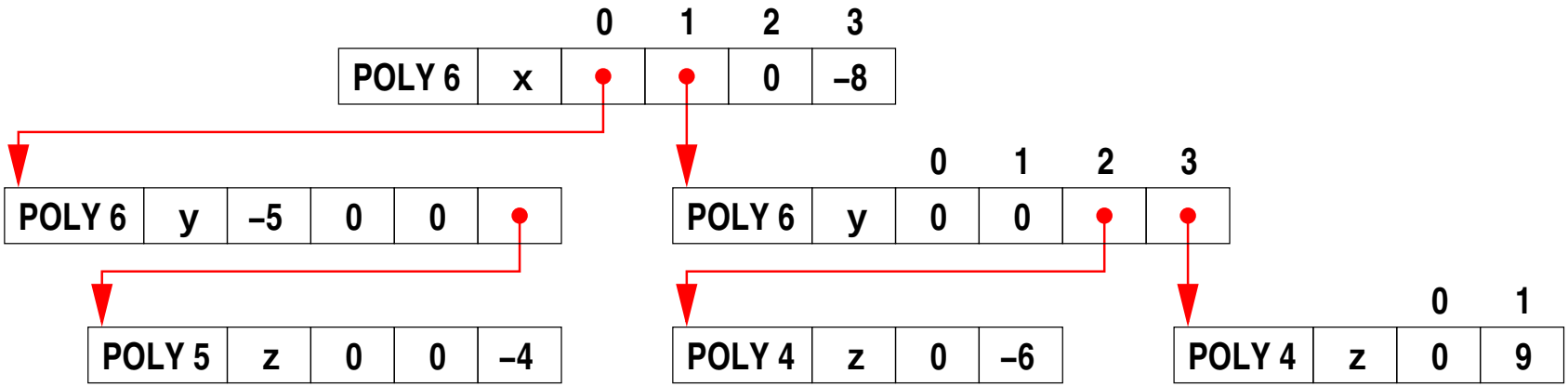


Trip's recursive sparse representation.



$$(-5y - 4z^2y^3) + (-6zy^2 + 9zy^3)x - 8x^3$$

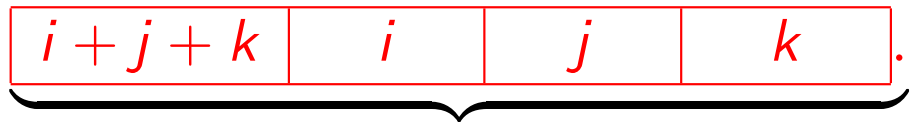
Pari's recursive dense representation.



Our sdmp representation uses packed monomials.

Packing for $x^i y^j z^k$ in graded lex order with $x > y > z$:

One 64 bit word :



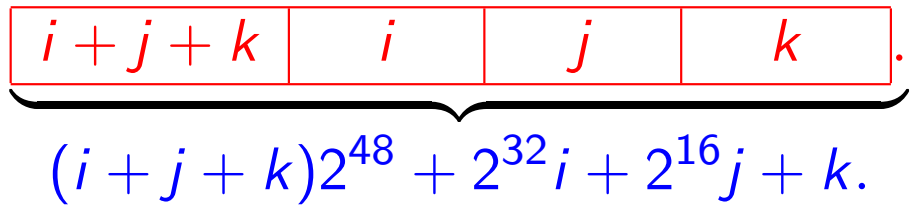
$$(i+j+k)2^{48} + 2^{32}i + 2^{16}j + k.$$

Why?

Our sdmp representation uses packed monomials.

Packing for $x^i y^j z^k$ in graded lex order with $x > y > z$:

One 64 bit word :



Why? Because monomial $>$ and $\times$ are one machine instruction.

Our sdmp representation uses packed monomials.

Packing for $x^i y^j z^k$ in graded lex order with $x > y > z$:

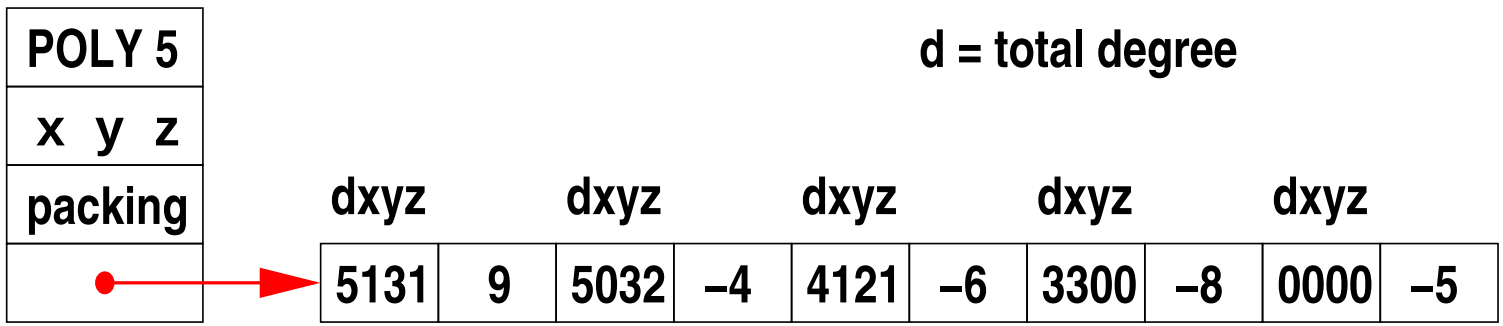
One 64 bit word :

$i+j+k$	i	j	k

$$(i+j+k)2^{48} + 2^{32}i + 2^{16}j + k.$$

Why? Because monomial $>$ and $\times$ are one machine instruction.

Our packed array for $9xy^3z - 4y^3z^2 - 6xy^2z - 8x^3 - 5$.

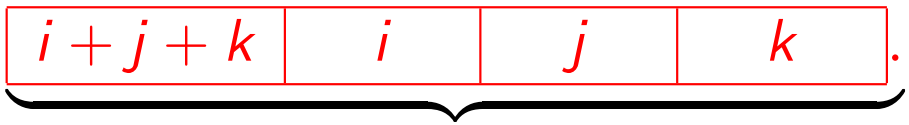


Why graded lex order?

Our sdmp representation uses packed monomials.

Packing for $x^i y^j z^k$ in graded lex order with $x > y > z$:

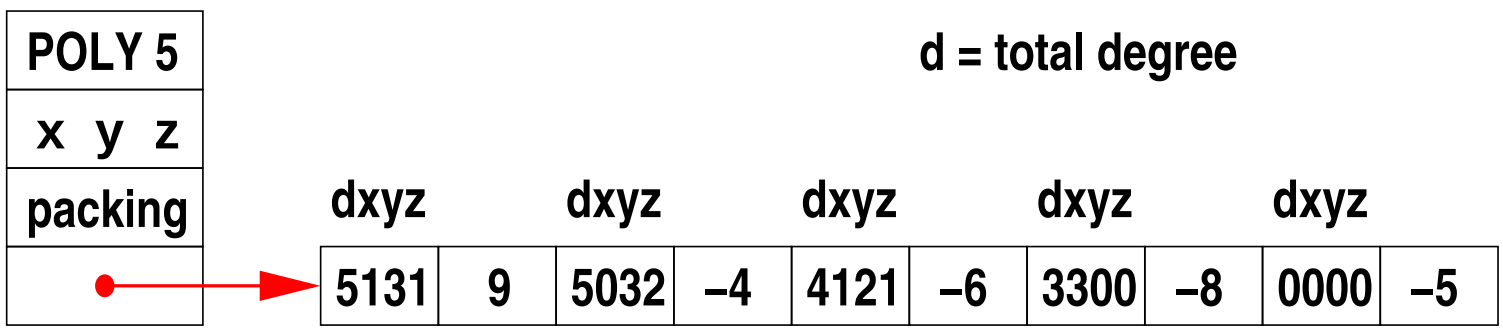
One 64 bit word :



$$(i+j+k)2^{48} + 2^{32}i + 2^{16}j + k.$$

Why? Because monomial $>$ and $\times$ are one machine instruction.

Our packed array for $9xy^3z - 4y^3z^2 - 6xy^2z - 8x^3 - 5$.



Why graded lex order? No exponent overflow in division.

How do CAS multiply and divide polynomials?

Let $f = f_1 + f_2 + \cdots + f_n$ and $g = g_1 + g_2 + \cdots + g_m$
where $f_1 > f_2 > \cdots > f_n$ and $g_1 > g_2 > \cdots > g_m$.

Using

$$h = f \times g = ((f_1 g + f_2 g) + f_3 g) + \cdots + f_n g \quad \text{and}$$
$$h \div g = f \ : \ (((h - f_1 g) - f_2 g) - f_3 g) - \cdots - f_n g$$

Let $f = f_1 + f_2 + \cdots + f_n$ and $g = g_1 + g_2 + \cdots + g_m$
where $f_1 > f_2 > \cdots > f_n$ and $g_1 > g_2 > \cdots > g_m$.

Using

$$h = f \times g = ((f_1 g + f_2 g) + f_3 g) + \cdots + f_n g \quad \text{and}$$
$$h \div g = f \ : \ (((h - f_1 g) - f_2 g) - f_3 g) - \cdots - f_n g$$

takes $O(n^2 m)$ comparisons of monomials
and $O(nm)$ multiplications of coefficient and monomials.

Example:
$f = x^n + x^{n-1} + \ldots + x$ and $g = y^n + y^{n-1} + \ldots + y$.

Our algorithms for multiplication and division use heaps.

Heaps

A binary heap H with n entries is a partially ordered array satisfying

$$H_i \geq H_{2i} \quad \text{and} \quad H_i \geq H_{2i+1}.$$

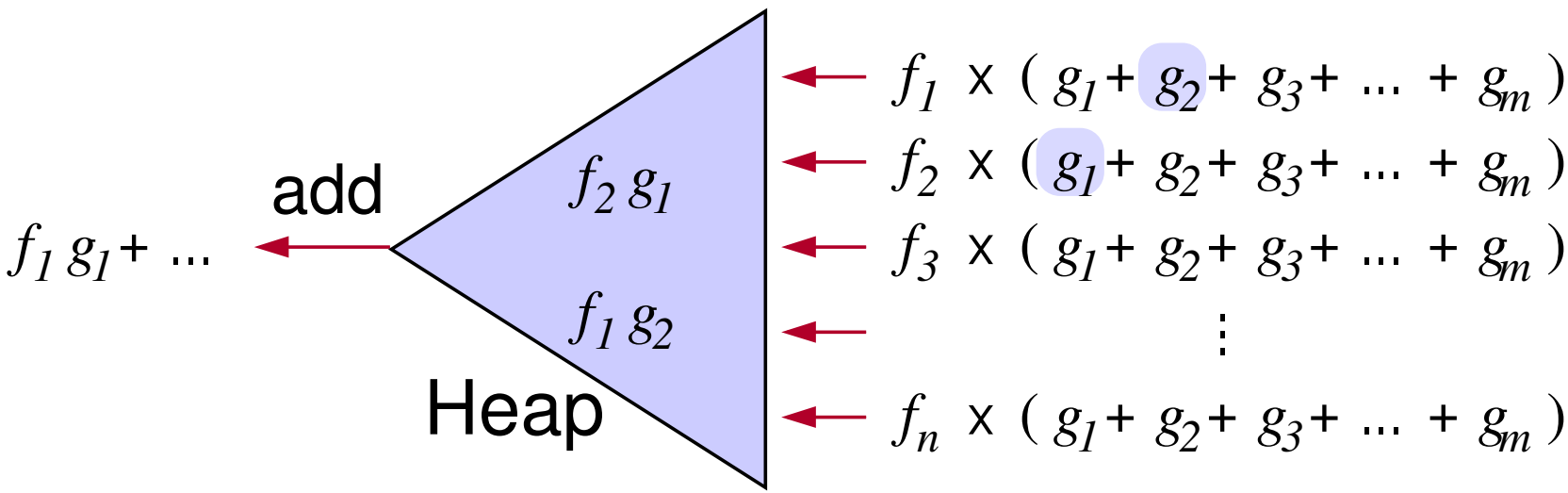
H_1	H_2	H_3	H_4	H_5	H_6	H_7	H_8
10	9	8	3	5	4	7	–

- H_1 is the biggest entry in a heap.
- We can extract the maximum entry in $O(\log_2 n)$ comparisons.
- We can insert a new entry in $O(\log_2 n)$ comparisons.

Multiplication using a binary heap.

Johnson, 1974, a simultaneous n-ary merge:

$$f = a_1 X_1 + a_2 X_2 + \cdots + a_n X_n$$
$$g = b_1 Y_1 + b_2 Y_2 + \cdots + b_m Y_m \quad \text{(sorted)}$$



- $O(nm \log n)$ comparisons.
- Space for $\leq n$ monomials in the heap.
- Can pick $n \leq m$.

Division using a heap.

Johnson's quotient heap algorithm.

$$\text{Dividing } f \div g = q \text{ compute } \boxed{f - \sum_{i=1}^{\#q} q_i \times g}$$

- $O(\#f + \#q\#g \log \#q)$ comparisons
- $O(\#q)$ working memory

Our divisor heap algorithm.

$$\text{Dividing } f \div g = q \text{ compute } \boxed{f - \sum_{i=2}^{\#g} g_i \times q}$$

- $O(\#f + \#q\#g \log \#g)$ comparisons
- $O(\#g)$ working memory

Minimal heap division (Monagan & Pearce, 2008)

Problem: we don't know if $\#q > \#g$ when starting a division.
E.g. $(x^7 - y^7) \div (x - y) = x^6 + yx^5 + y^2x^4 + \cdots + y^6$.

Start with quotient heap, switch to divisor heap when $\#q = \#g$.

$$f \; - \underbrace{\sum_{i=1}^{\min(\#q,\#g)} q_i \times g}_{\text{quotient heap}} \; - \underbrace{\sum_{i=2}^{\#g} g_i \times (q_{\#g+1} + \cdots)}_{\text{divisor heap}}$$

- $O(\#f + \#q\#g \log \min(\#q, \#g))$ comparisons
- $O(\min(\#q, \#g))$ working memory

Which CAS is fastest?

Benchmark 1: A dense Fateman problem.

$$f = (1 + x + y + z + t)^{20} \quad g = f + 1$$

- f and g have 39 bit coefficients and 10,626 terms
- $h = f \cdot g$ has 83 bit coefficients and 135,751 terms

Intel Core2 3.0 GHz	multiply $p = f \times g$	divide $q = p/f$
Maple 12	289.23 s	187.72 s
Maple 13	187.35 s	159.12 s
Singular 3-0-4	62.00 s	20.00 s
Magma V2.14-7	23.02 s	22.76 s
Pari 2.3.3 (w/ GMP)	32.43 s	14.76 s
Trip v0.99	5.93 s	-
sdmp	2.26 s	2.77 s
Maple 14	3.33 s	4.46s

Benchmark 2: A sparse 10 variable problem.

$$
\begin{aligned}
f &= (x_1x_2 + x_2x_3 + x_3x_4 + x_4x_5 + x_5x_6 + x_6x_7 + x_7x_8 + x_8x_9 + x_9x_{10} + x_1x_{10} \\
&\quad + x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + 1)^4 \\
g &= (x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2 + x_7^2 + x_8^2 + x_9^2 + x_{10}^2 \\
&\quad + x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + 1)^4
\end{aligned}
$$

$6,746 \times 8,361 =$ $3,157,883$ terms	multiply $p = f \times g$ seconds	divide $q = p/f$ secs
Maple 12	305.76s	280.65s
Maple 13	293.74s	312.29s
Singular 3-0-4	31.00s	18.00s
Magma V2.14-7	17.43s	197.72s
Pari 2.3.3 (w/ GMP)	7.06s	7.05s
Trip v0.99 (rationals)	8.13s	–
sdmp	2.46s	2.61s
Maple 14	11.74s	14.45s

Demo of the GraphTheory Package.