Maple Lecture 12. Representation of Expressions

In this lecture we take a detailed look at the internal representations of expressions. The material in this lecture is inspired on [1, Section 6.1 and 6.2]. See [2] for more specific details. The use of directed acyclic graphs to represent expressions has implications on the effect of a substitution, one of the major tools to manipulate expressions.

12.1 Internal representation of polynomials

There are three levels of selecting data from a polynomial. At the highest level we have **coeffs** and **coeff**. The commands **nops** and **op** work at a lower level and allow us to draw the expression tree of the polynomial. Finally, at the lowest level, we use **dismantle** to get to the Directed Acyclic Graph representation of a polynomial.

This Directed Acyclic Graph representation is motivated by the memory efficiency of Maple: every object is stored only once. So leaves in the expression tree with the same content are stored as the same end node in a directed acyclic graph. But also the substitution command can be executed more efficiently: the replacement of the same symbol (or number) in an expression occurs only once, even though the symbol (or number) may appear in several places in the expression.

```
[> p := x^10 + 2*x^4 + 9;
[> coeffs(p);              # sequence of coefficients
[> whattype(p);            # same as op(0,p);
```

While p is a polynomial, it is also a general expression of type + on which the usual selectors work:

```
[> nops(p);                # number of terms
[> op(p);                  # sequence of terms
[> op(1,p);                # select first term of p
[> op([2,1],p);            # select 1st term of 2nd term of p
```

With the **op** command, we can construct the expression tree, shown in Figure 1.

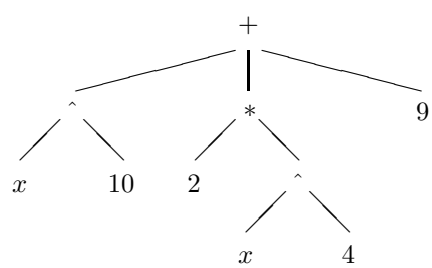

Figure 1: Expression tree of $x^{10} + 2x^4 + 9$.

The expression tree is still very much a logical view on the polynomial. Internally, Maple stores an expression as a *Directed Acyclic Graph*. From the output of **dismantle** we draw this graph in Figure 2.

```
[> dismantle(p);
```

Notice: in storing the terms in the sum, the monomials come first, followed by the coefficients. So the polynomial $x^{10} + 2x^4 + 9$ is actually stored like $x^{10} \times 1 + x^4 \times 2 + 9 \times 1$.

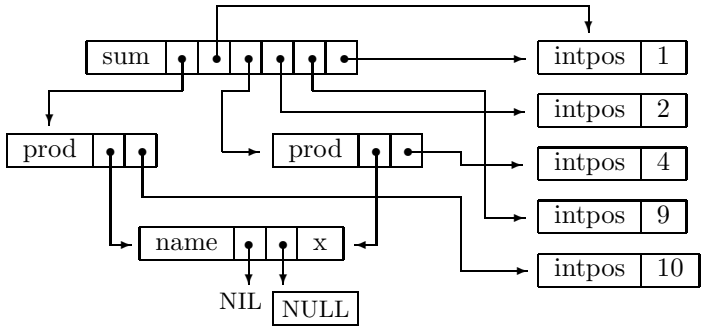

Figure 2: Directed Acyclic Graph of $x^{10} + 2x^4 + 9$.

12.2 The side effects of a substitution

Now we understand the structure, and knowing that Maple stores every object only once, we can explain the effects of a substitution like this (see Figure 3):

```
[> q := subs(1=Pi,p);
[> dismantle(q);
```

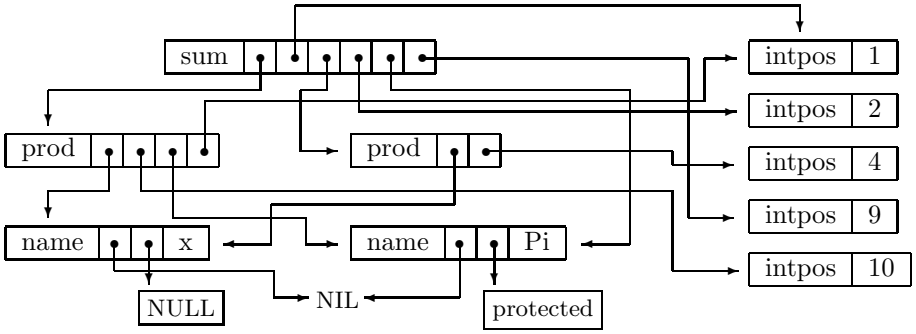

Figure 3: Directed Acyclic Graph of $x^{10}\pi + 2x^4 + 9\pi$.

Observe that `protected` is an attribute to the name `Pi`. When a variable has no attribute (like `x`), then this field of the name simply points to `NULL`.

Since the polynomial `p` $= x^{10} + 2x^4 + 9$ is stored like $x^{10} \times 1 + x^4 \times 2 + 9 \times 1$ and every object is stored only once, we now see why the command `subs(1=Pi,p)` creates the polynomial $x^{10} \times \pi + x^4 \times 2 + 9 \times \pi$.

12.3 Generalized rational expressions

Maple can work with polynomials where the variable is replaced by, for example a cosine:

```
[> pcos := subs(x=cos(x),p);
[> whattype(pcos); type(pcos,polynom);
```

While the expression is no longer a true polynomial, we can still ask for the coefficients :

```
[> coeffs(pcos);
```

But we can no longer factor the polynomial directly :

```
[> factor(pcos,complex);
```

We have to work indirectly, first factor p and then substitute x by fp in the factored polynomial:

```
[> fp := factor(p,complex);
[> fpcos := subs(x=cos(x),fp);
```

General rational expressions are rational polynomial in disguise:

```
[> rp := ((sin(y))^2-1)/(sin(y)-1);
[> type(rp,ratpoly);
[> factor(numer(rp));
[> normal(rp);
```

It is interesting to see the internal structure (see Figure 4):

```
[> dismantle(rp);
```

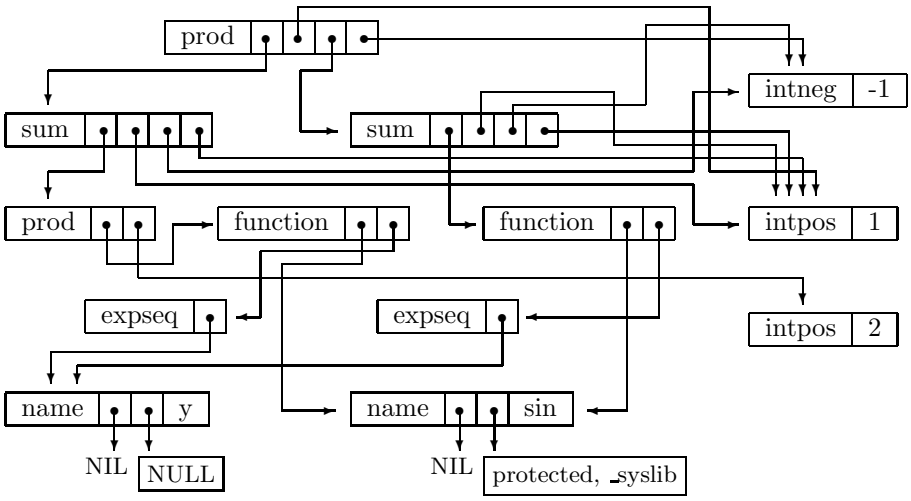

Figure 4: Directed Acyclic Graph of $\frac{\sin(y)^2-1}{\sin(y)-1}$.

The dismantle procedure is one of the tools in the "hackware package" of Maple. Like we could do in a low-level programming language as C, we can manipulate addresses. For example:

```
[> dismantle[dec](p);          # see the addresses in decimal notation
[> addressof(p);               # verify the address of p
```

12.4 Assignments

1. For the polynomial $p = x^3 - 8x + 1$, give the Maple command (only one command!) to select the coefficient -8. Draw the expression tree for p and the directed acyclic graph of its internal representation.

2. Use dismantle to investigate the internal representation of the polynomial $2x(y^2+1)^2$. From the output of dismantle, draw the directed acyclic graph of this polynomial.

3. Consider the following instructions

```
[> x^2 - x + 1/x - 1/x^2;
[> subs(-1=1,%);
```

   Explain the result of this substitution.

4. Compare the internal representations of $p = x^8 + 2x^4 + 3$ with $q = (x^4 + 2)x^4 + 3$.
   Draw the directed acyclic graphs of p and q based on the output of dismantle.

5. Do **dismantle[dec](p);** after $p := x^8 + 2x^4 + 3$.
   Explain how you can see from the output of dismantle that the name x is represented only once.

References

[1] A. Heck. *Introduction to Maple*. Springer-Verlag, third edition, 2003.

[2] M.B. Monagan, K.O. Geddes, K.M. Heal, G. Labahn, S.M. Vorkoetter, J. McCarron, and P. DeMarco. *Maple 9 Advanced Programming Guide*. Maplesoft, 2003.