

Re: what's it worth to write a short program for polynomial multiplication?

Re: what's it worth to write a short program for polynomial multiplication?

Source: <http://sci.tech-archive.net/Archive/sci.math.symbolic/2008-06/msg00005.html>

- *From:* TimDaly <daly@xxxxxxxxxxxxxxxxxxxx>
 - *Date:* Mon, 2 Jun 2008 17:41:52 -0700 (PDT)
-

On Jun 2, 1:12 pm, rjf <fate...@xxxxxxxx> wrote:

On Jun 2, 8:39 am, hru...@xxxxxxxxxxxxxxxxxxxxxx (Herman Rubin) wrote:

In article
<570bb6f4-6c19-42f1-bc82-f6872d4f6...@xxxxxxxxxxxxxxxxxxxxxxxxxxxx>,

Well, the next sentence in the abstract is...

What is the smallest expression in
a programming language for such a program? (Assuming that the language
does not have "multiplication of polynomials" as a primitive!)

HR:

Obviously, in an appropriate programming language, this

will be a basic operation, so the smallest expression
would be

$z = x * y,$

where the overloaded operator $*$ does what is expected.

This is not really a solution unless the overloaded operator is a lot
smarter than we usually expect.

Re: what's it worth to write a short program for polynomial multiplication?

That is, "*" must not only look at the types of x and y [which are presumably "polynomial"] but at other information, such as the domain of the coefficients, which could be integers, floats, intervals, matrices, elements in a finite field, strings, etc.

It might also look at the sizes, density, number of variables, and it might look at the environment: how many processors are available, how much memory, etc. It certainly should look at both x and y, and perhaps also look at how z is described.

Axiom takes into account the types of the arguments AND the type of the result when choosing the * operator. In addition, the types carry information which describes the polynomials (sparse, univariate, dense, multivariate, distributed, etc) as well as type information for the field of coefficients (integer, fractions, complex, finite, etc).

Thus, there are many polynomial multiplications available which go by the name "*" but are dynamically chosen at runtime to be the most specific (and presumably, most optimal) way of multiplying polynomials.

The algebra can easily be extended to support new representations with most-specific multiplication operations.

There are 58 * operators in Axiom, many of which can be automatically specialized for the polynomial type in question. Thus,

$x * y$

is the shortest, valid way to efficiently multiply two polynomials in the Spad (scratchpad, Axiom's native language).

Tim Daly
Axiom Lead Developer

.