Computer Algebra and the three 'E's: Efficiency, Elegance and Expressivenes

James H. Davenport & John Fitch
Department of Computer Science
University of Bath
Bath BA2 7AY England
{J.H.Davenport,J.P.Fitch}@bath.ac.uk

June 29, 2007

We all want (as users) or claim to provide (as designers) the three ‘E’s

- Elegance
- Expressiveness
- Efficiency

Elegance (of input)

$$\frac{-b+\sqrt{b^2-4ac}}{2a} \tag{1}$$

```
\frac{-b+\sqrt{b^2-4ac}}{2a}
```

```
(-b+SQRT(b^2-4*a*c))/(2*a)
```

```
(/ (+ (- b) (SQRT (- (^ b 2) (* 4 a c)))) (* 2 a))
```

```
(divide (plus (minus b) (sqrt (minus (power b 2)
                                     (times 4 a c))))
        (times 2 a))
```

But is this a real issue?

1. There is so much going on (MathUI) that the visual should cease to be a problem.

   - "I don't mind editing XML as long as I don't have to look at it".

2. It *is* nice to have automatic n-arisation, especially with lists:
   `'gcd'/[content(p,x) for p in l]>` is nice.

3. Especially if the system can do 'early abort' on finding 1, as in Axiom.

   - Rest becomes 'expressiveness'.

Elegance (of output)
This is a real issue.

Who can wade through the 100s of pages our system can produce at the drop of a hat?

Users This is a *system* issue, not a *language* issue.

Programmers Do need proper support in the language to support debugging, with I/O in *their* types, not the macine types in which they are implemented. Interpreted languages tend to provide this, compiled ones not (but Axiom did!).

Expressiveness

Of course, we really want

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}. \tag{2}$$

- Easy — just extend the operators.
- Often appropriate: $\mathbf{v}/||\mathbf{v}||$.
- But not the panacea it seems.

$$\frac{1}{6}\sqrt[3]{-108\,c+12\sqrt{12\,b^3+81\,c^2}}$$

$$-\frac{2b}{\sqrt[3]{-108\,c+12\sqrt{12\,b^3+81\,c^2}}},$$

is apparently 36-valued. Even

$$\left(\lambda x.\frac{1}{6}x-\frac{2b}{x}\right)\sqrt[3]{-108\,c+12\sqrt{12\,b^3+81\,c^2}}$$

is apparently six-valued.

Expressiveness needs types
(JHD only; JPff disagrees)

- If the elements of my matrix come from a commutative ring, I want you to multiply the matrices ...

- and calculate the determinant.

- What do you mean: "division by a zero divisor"!

No known type system is powerful enough!

Efficiency: what is special about us?

- There's no credit for being the second to do a computation.

* But the same is true of the rest of computational science.

- My data are so large.

* Bet Google's eigenvalue problem is bigger than yours!

The dynamic range

Gaussian elimination in sparse matrices

- Dodgson/Bareiss fraction-free
- With special sparsity hacks
- The entries might be very large
- or they might be integers, mostly very small

At one extreme, I'll tolerate *any* overhead, at the other I want byte-packing for most of the entries.

How does this manifest itself?

- Early Maple's 'polynomial gcd by evaluation'.

* Integers are fast, $\mathbf{Z}[x]/(p)$ isn't.

- Code bloat.

- Axiom's 'special case compilation'.

- Singular's hack for exponent packing.

* But they're safe!

Questions to think about
(almost all related!)

- Where is the kernel boundary?
- How will I get efficiency when the objects are small/fast?
- Are my efficiency hacks safe?

* If not, should I be in this game at all?

- Are there efficiency hacks that *could* be safe/semi-safe?

- Now, where *was* that swamp I was menat to drain?

* (with thanks to Fred Brooks)