

## Re: Mathematica vs. Lisp

**Source:** <http://coding.derkeiler.com/Archive/Lisp/comp.lang.lisp/2004-06/1236.html>

---

**From:** Richard Fateman (*rfateman\_at\_sbcglobal.net*)

**Date:** 06/21/04

Date: Sun, 20 Jun 2004 23:54:32 GMT

I agree with the comments by Albert Reiner.

Wolfram wrote SMP because he felt that Macsyma, which he used extensively, was too slow. Wolfram thought that he was cleverer than anyone else and could read Knuth's books and then write a better computer algebra system. He wrote it with

S. Cole, Geoffrey C. Fox, Jeffrey M. Greif, Eric D. Mjolsness, Larry J. Romans, Timothy Shaw, and Anthony E. Terrano.

SMP was terrible.

It had some bad designs, naive and buggy.

For example, Wolfram thought he could use double precision to approximate rational numbers, and no one would notice.

Comments on its successor, Mathematica:

Since the fundamental paradigm for the user-level programming language is pattern matching, what you usually get is a simulation of functional programming using patterns and replacements. If you look at this too closely, the analogy breaks. Wolfram would like to claim that Mathematica combines (the best?) ideas from many languages, including Lisp.

At least in the past, the time to add an element to the front of a list, or the back of the list, was the same, but it depended on the length of the list. How would you like "cons" to depend on the size of the inputs? I haven't timed this Mathematica feature recently.

Also, the evaluation model is quite bizarre. A value of a variable V is re-evaluated whenever it is touched, iff something that V depends upon has been changed since V was last touched. This is approximated by keeping a collection of pages of items which have values on which V depends.

Since side effects are possible, you can have values change

non-deterministically. Sorry I can't give you a reproducible example:) Note also, that if one value in an array  $a[n,m]$  is changed, then the time to access any other value of that array increases, because they have to be re-evaluated. This makes some algorithms that one would think were (say) quadratic, into cubic or even exponential time.

Mathematica is interesting, but the programming language provided to the user violates many rules of good behavior. But then, this is true of other systems catering to scientific applications programmers who are just fiddling around, whose answers probably don't matter, or can be checked by physical experiment, whose programs are either single commands or short programs, and never heard of big-Oh notation, or floating-point error analysis.

I wrote a Mathematica parser and cut-down pattern matcher/evaluator in Common Lisp. Wolfram made noises about copyright infringement, but eventually stopped. Search on google for "mockmma" to find a copy to download.

Of course another difference is that Mathematica is proprietary and definitely not free.

RJF

ps. for the source of the "runic" comment, see

<http://www.cs.berkeley.edu/~fateman/papers/mma.review.pdf>

David Golden wrote:

> *Sashank Varma wrote:*

>

>

>> *There are stories somewhere on the net that when Wolfram was  
>>a wunderkind at Caltech, he used Macsyma -- the original  
>>mathematical software package, written in Mac Lisp -- and  
>>talked to some of the programmers about it.*

>

>

> *Worth noting that Common Lisp Maxima is a descendant  
> of Macsyma, still around thanks to William Schelter  
> getting it open sourced before he died. Pretty cool,  
> and in debian with the rather understated package description  
> "A fairly complete computer algebra system".*

>

> <http://maxima.sourceforge.net/>

>