

Re: Symbolic Math as Computer Science

Source: <http://sci.tech-archive.net/Archive/sci.math.symbolic/2005-12/msg00151.html>

- *From:* Hatto von Aquitanien <abbot@xxxxxxxxxxxxxxxx>
 - *Date:* Fri, 09 Dec 2005 15:47:21 -0500
-

Richard J. Fateman wrote:

>
>
> Hatto von Aquitanien wrote:
>
> ...
>>
>> In my view there are really only a small number of basic ideas in
>> mathematics. They are all founded in our naive notions of left-right,
>> front-back, up-down, past, present, future, before, after, between,
>> counting, measuring, true, and false. From such basic ideas, I believe we
>> can determine a set of fundamental entities which can be used to
>> construct all mathematical ideas.
>
> See
> <http://www-unix.mcs.anl.gov/qed/>
>
> or
> <http://imps.mcmaster.ca/na-mkm-2004/>
>
> You will have to judge for yourself how much has been accomplished
> versus how much has been proposed.

What your referenced with the links seem to be encyclopedic efforts at cataloging existing mathematical knowledge.

I'm interested in modeling the process of mathematical reasoning. I'm not sure there has really been an effort to implement what I'm envisioning. I would not implement every feature strictly in terms of absolutely fundamental concepts, but I would implement everything in such a way that the implementation would, in the users view, be indistinguishable from a system built on such primitive principles (allowing for the finite nature of computer hardware, of course.)

I can't say I'm qualified for the task to accomplishing what I'm talking about. I don't know to what degree it has been attempted. Mathematica comes close in some ways, but does many things which I might do differently. For one, I don't believe complex numbers would be primitive

Re: Symbolic Math as Computer Science

objects in the way they are in Mathematica. Overall, I would probably try to establish a better type system. I'm still not sure why Mathematica lacks a real data type mechanism. I certainly think of mathematical objects appearing in pencil and paper expressions as having specific types. Those are usually the first things I define when solving a problem.

> .. re — JSC Review of Mathematica
>
> . I wouldn't describe it as flattering to Mathematica.
> Right
>> I understand that it is a critical review, and can expect this to some
>> extent. Some of the criticisms are a bit pedantic/nit-picky from the
>> perspective of a "real user".
> If a user is lead to expect that the answer will be mathematically
> correct, and it is incorrect even .1% of the time, you might consider
> that to be a "nit". I consider it a serious flaw, especially if the
> correct behavior is possible.

I was intending some of the issues of syntax, etc.

> OTOH, Mathematica is very difficult to
>> learn to use well,
> I agree

>> I believe the reluctance to expose the concepts which make it work is a
>> big contributor to the difficulty in learning to use the product.
> I think this is debatable. Some people would, I think, just be confused
> by the "inside" story. Serious users might benefit more, though.

I'm not really talking about the whole of the implementation details. I really mean a more coherent presentation of how the system works. Perhaps I'm wrong, but the more I learn about the system, the more it seems the documentation tends to avoid directly explaining features which the user really should understand, but which might also reveal how the program is implemented. Perhaps it really is an idiosyncrasy of the Mathematica culture not to explain certain aspects of the program and language in what I consider to be a unified and coherent fashion. Perhaps it's just my limitations which have prevented me from seeing the big picture.

If I look at K&R, The C++ Programming Language by Stroustrup, The C++ Standard Library by Josuttis, The Java Programming Language by Gosling, et al, The GNU Emacs Lisp Reference Manual, Introduction to Programming using SML, by Hansen & Rischel, lex & yacc by Levine, Mason and Brown, Learning the Bash Shell by Newham & Rosenblatt, JavaScript: The Definitive Guide, or any number of other books teaching how to use a programming language I will find some kind of diagram illustrating the runtime structure of some part of a program written in that language. I honestly cannot think of one example of such a thing in any Mathematica book I have seen. Maeder's Computer Science with Mathematica comes closest, but the diagrams represent generic data structures. Is that just coincidence? What's truly amazing is that some of these books have hundreds of pages dedicated to the

graphical representation of conceptual information.

And, no, I have not read all the books mentioned above cover to cover.

—

Nil conscire sibi

.

• *Follow-Ups:*

◆ *Re: Symbolic Math as Computer Science*

◇ *From:* Paul Abbott

◆ *Re: Symbolic Math as Computer Science*

◇ *From:* Martin Rubey

• *References:*

◆ *Symbolic Math as Computer Science*

◇ *From:* Hatto von Aquitanien

◆ *Re: Symbolic Math as Computer Science*

◇ *From:* Rouben Rostamian

◆ *Re: Symbolic Math as Computer Science*

◇ *From:* Richard Fateman

◆ *Re: Symbolic Math as Computer Science*

◇ *From:* Hatto von Aquitanien

◆ *Re: Symbolic Math as Computer Science*

◇ *From:* Richard J. Fateman

• Prev by Date: *Re: complexity of numerical software(2)*

• Next by Date: *Re: complexity of numerical software(2)*

• Previous by thread: *Re: Symbolic Math as Computer Science*

• Next by thread: *Re: Symbolic Math as Computer Science*

• Index(es):

◆ *Date*

◆ *Thread*