

Re: Pattern Matching and Transform Rules

Source: <http://sci.tech-archive.net/Archive/sci.math.symbolic/2005-11/msg00229.html>

- *From:* Richard Fateman <fateman@xxxxxxxxxxxxxxxx>
 - *Date:* Mon, 28 Nov 2005 06:31:29 GMT
-

I don't know the full context of Michael Trott's statement.

Perhaps he was not aware that the mathematica pattern matcher was based on one written in Lisp? (I think it was first available in Reduce).

Perhaps he was unaware that a fairly complete duplicate of the exact syntax and semantics of the pattern matcher is given in open-source lisp code in MockMMA. Perhaps he was unaware of the large number of pattern matchers and unification programs written in Lisp, Prolog, ML, A good example of one is in Paradigms of AI Programming by Peter Norvig. Another is Schatchen, used by Moses' symbolic integration program, which does a number of tricks that Mathematica's pattern matcher cannot do. Or defmatch in Macsyma.

Some of them are not as elaborate as mathematica's, but to dismiss them ALL, probably sight unseen, is, as Hatto von Aquitanien says, hazardous.

If Mathematica's language is so great, why has so much of the code been written in C? Hundreds of thousands of lines of code. And new code continues to be written, in version after version.

Re: Pattern Matching and Transform Rules

fjolsvit@xxxxxxxxxxxxx wrote:

Can you provide an example of how, say, C++ has superior pattern matching than Mathematica? We are talking about a recursive application of the language which is used to examine and transform expressions written in the language under consideration.