Re: Lisp and Scheme with fewer parentheses / Mathematica??

Source: http://coding.derkeiler.com/Archive/Lisp/comp.lang.lisp/2008−01/msg00535.html

- *From*: Xah Lee <xah@xxxxxxxxxx>
- *Date*: Tue, 8 Jan 2008 17:01:44 −0800 (PST)

Dear Rainer,

Any dummy, at our level, knew that Mathematica and lisp have different evalutaion models or computational models. (the at our level , means us, who have years of experience with one or more fringe , functional programing languages. Of course, if you ask a average industrial programer, even with say 10 years of programing experience, chances are, they wouldn't know what the fuck is a rule−based system or lisp macros)

So, of course i'm not arguing that lisp and Mathematica have the same evaluation models. I'm saying, as well as Jon is saying, that programing in Mathematica is like in sense a glorified (or extened) macros system as used in lisp. In fact, one might argue, that the concept of so−called rule−based system or term−rewritting is developed from lisp. (after all, it is almost the case that any advanced functional programing concepts has origin or trace of origin from lisp, because after all, lisp is for all practical purposes the first functional language and one of the oldest language)

Now, what you might be arguing, is that it is not reasonable to say programing mathematica is akin to programing with a advanced macros system because they are different evalution models (more generally, differente computational models). But this is like picking spellings or drilling technicalities. To illustrate my point, if i say lisp's list is far more powerful than Perl's, then the perl fuckheads will argue that no, because Perl has list and array, which is entirely different beast than lisp's cons cells model.

So, in your previous message, there is really nothing i disagree in any technical sense. In fact, your admission that Mathematica is in general higher−level than lisp, is all really me and perhaps Jon was trying to make.

Now, this does not mean, Mathematica is better than lisp in all aspects. Of course, Mathematica's high−level−ness, its ability to computer thousands of advanced mathematical functions and equations, its pattern matching, list manipulation power, is all good. And, don't

get me wrong: Mathematica is not a niche computer−algebra−only language. However, there are certainly many things mathematica cannot do and is not suitable to do as compared to lisp. For example, you can't really do much of network programing with Mathematica (e.g. writing a web application, networked file system, or ssh client, etc.). You can't really use Mathematica to program GUI applications in the general sense. (i.e. write a X−windows window manager, or write a web browser) And, due to M's high−level−ness, it is also not suitable to, say, to use it to write a OS device driver. (note: Mathematica has WebMathematica that function as a general web−application framework, and GridMathematica that functions as general parallel computing system, and the Mathematica Kernel with its MathLink protocol allows several general extension and programing with it in diverse areas)

In short, Mathematica's supreme high−level−ness is not a lisp killer. Meanwhile, the fact it is higher−level than lisp, and in many ways practically similiar to lisp, should not be dimsissed or bent out of shape by that Richard Fateman or any Common Lisper.

PS It is not correct to say that Mathematica relies on pattern matching or term−rewriting as its core computational model. Because, in Mathematica, there are pure function Function (lambda) construct, as well as many imperative language constructs that cannot be explained by pattern matching. (a simple example: x++ or the syntactically equivalent Increment[x]) In fact, it is not possible to write any non−trivial programs in Mathematica by using only pattern matching. In my over a decade programing with Mathematica (mostly active in the 1990s, i actually avoid pattern matching, for reasons that in general pattern matching is somewhat unpredictable, and code based on it is not so portable since other langs don't have it.)

Xah
xah@xxxxxxxxxx
? http://xahlee.org/

On Jan 8, 3:31 pm, Rainer Joswig <jos...@xxxxxxx> wrote:

> In article
> <559d62bb−9e82−44d9−b3b6−4abef91e2...@xxxxxxxxxxxxxxxxxxxxxxxxxxxx>,
> XahLee<x...@xxxxxxxxxx> wrote:
>
> > No Rainer, you don't understand.
>
> > Try to get out of lisp mindset, and view things from general humanity.

> Alternatively, imagine a alien from outer space is visiting earth. He, being a alien, don't understand how computer scientists classifed languages here, nor anything about our hardware such as silicon cpu and memories etc. He, being advanced alien, we assume he understands advanced mathematics far beyond us. In particular, 1+1 is still 2 for him, and any implications.
>
> Now, will this Alien, buy your stuff about how lisp is this and that?

Could be.

> Also, let's be explicit about what we are arguing here, even in general there is no clearly defined or focus of a newsgroup argumentation. But, roughly, in our context, the argument is about, whether, is it reasonable, to say that programing in mathematica is analogous to a far more powerful system of lisp's macros.

I don't think Lisp macros are more 'powerful'. Mathematica
(the language and its implementation) works a bit differently.
In some ways you could say that Mathematica is more
powerful (since it transforms expressions by rewrite rules) and
more 'high−level'. There are some similarities to Lisp,
but the core 'evaluation' model is different.

Let me try to describe the main difference:

* Lisp evaluates functions to values or executes special forms.
Macros do source transformation before the code
gets executed (the created code is then executed).

* Mathematica applies transformations to data
as long as their are applicable transformations.

Lisp has a 'simpler' and more straight forward model evaluation model.
Even Common Lisp. That also has some advantages.
Lisp has a 'Functional' core. Mathematica has a 'rule−based
rewrite engine' at its core. This allows Mathematica to easily
do symbolic manipulations like you see in computer algebra
systems.

In Lisp you need to extend the basic system to do that. The
infrastructure is there. This infrastructure has been designed
such these kind of systems can be 'relatively' easily
be written. So Lisp is a bit more low−level than Mathematica.
Lisp has been used to write systems that work in some ways 'similar'
to Mathematica. Examples are Macsyma, Reduce, Axiom.

If you look at Macsyma, you can easily see where the evaluation
model of Lisp has been extended to manipulate symbolic
expressions (like mathematical formulas). Lisp's role
is more to be a straight−forward implementation language
for systems and languages like Mathematica.

There is one part of Common Lisp that I would think
is also 'high−level' − that is CLOS. The rest is low− to mid−level.

So, coming back to your Subject
'Lisp and Scheme with fewer parentheses / Mathematica??'

No, Lisp and Scheme are sufficiently different from Mathematica
in the way how they do computation, that Mathematica is not
a Lisp with fewer parentheses. Lisp, the language, is in some
ways more low−level.

Lisp with fewer parentheses would be more like Logo or
Dylan. Especially the latter.

> Now, if you agree that this is the subject as i phrased it, then, i
> think you either don't have opion, or, you agree with me. Further, i
> argue, that this phrase, is a reasonable interpretation where the
> thread is from, more specifically, it is a reasonable interpretation
> of what Jon Harrop said, of which Richard Fateman retorts.
>
> Now, if you don't agree that the subject is as i phrased it, then, you
> can rephrase it, or explicitly define a subject that we would argue
> about. Then, i, or others, can voice our opinion. (or not)
>
> Xah
> x...@xxxxxxxxxx
> ?http://xahlee.org/
>
> On Jan 8, 1:50 pm, Rainer Joswig <jos...@xxxxxxx> wrote:
>
>> In article
>> <a4fa6a2d−2eed−4ea9−bb83−d8b27cfe7...@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx>,
>> XahLee<x...@xxxxxxxxxx> wrote:

2008−01−07

I just ran into a newsgroup post, where the
redoubtable Richard J
Fateman (a well−known Mathematica hater)
sullies Mathematica by
sputtering computer sciency obfuscation.

Here's the source of the message:

http://groups.google.com/group/comp.lang.lisp/msg/163eb656becfcab6?dm...

You may want to read Richard's mail again and check what
he is saying.

He is basically right.

Mathematica is a rule−based rewrite−system at its core.
That makes it substantially different from the evaluation
'rules' of (Common) Lisp.

http://www.lispworks.com/documentation/HyperSpec/Body/03_ab.htm

or older:

http://www.cs.cmu.edu/Groups/AI/html/cltl/clm/node55.html#SECTION0090...

Compare that with

http://documents.wolfram.com/mathematica/book/section−2.6.1
http://documents.wolfram.com/mathematica/book/section−2.6.4

That's why I said in another posting that you need to understand
the evaluation / compilation model of Lisp.

.