

# Re: Computer Algebra Algorithms

---

*Source:* <http://sci.tech-archive.net/Archive/sci.math.symbolic/2005-04/msg00023.html>

---

- *From:* JohnCreighton @xxxxxxxxxxx
  - *Date:* 7 Apr 2005 22:55:19 -0700
- 

mehran gupta wrote:

>Hi,  
>  
>  
>Can anyone please point me to a good basic textbok or reference on  
>computer algebra algorithms. I am mainly interested in writing C >code  
  
>for simple algebraic simplication.  
>  
>  
>Thank You,  
>  
>  
>M. G.

I've been trying to follow this thread somewhat but there is many more posts here then I want to toughly dissect. I don't really think in the above posts enough motivation was given behind what you want to do. If you want to learn CAS, learn lisp because that is what the open source CAS are programmed in and a large part of the academic work in AI uses lisp.

If you want to play around with a few simplification algorithms I see no reason why you cannot use C. I agree with an above poster that C++ would be better if you wanted your C code to give a CAS the C code does not have to look anything like the language the CAS uses. In your C++ code you could overload each operator for type mathEquation and then these operators would construct expression trees based upon the rules you define. This would suffer from the batch limitation as Richard suggested. Each time you wanted to try some simplification you would have to run the compiler and look at your results.

If you want to use C dynamically you are going to need some kind of parser. The parser could be written in C or any other language. It should return to you a tree structure representing the expression the user typed at the command line. You could then perform some algebra on it. In this case you may not care if you can overload the plus

## Re: Computer Algebra Algorithms

operator. You could just call it as a C function `pluss(a,b)`, since everything would be done by your custom interpreter (would this really be C?).

Java was mentioned as a language that wouldn't be a good choice because it didn't support overloading of operators. Richard said there is ways around this. I know in Java you can overload the concatenation operator by creating a method called `toString`, but I don't think when I learned it you could do something equivalent for times.

Richard mentioned a limitation of overload. It is well known that the world is not made up of objects but objects are just a model we use at an instance to dissect a problem. It is difficult to generalize functions in object oriented programming because the functions must be written for members of the same class or that implement the same interface. Many programming languages such as Haskell have a much better type checking system that allows better generalization and more flexible polymorphism.

[http://en.wikipedia.org/wiki/Polymorphism\\_%28computer\\_science%29](http://en.wikipedia.org/wiki/Polymorphism_%28computer_science%29)

If you want to use C and you want to do something useful with CAS I would suggest learning the external function interface of MAPLE. This will allow you to Manipulate maple data structures in the Maple workspace with C functions. I haven't got into it but perhaps you could even use it to prevent the automatic simplification in Maple. Apparently the best reference for this is the Maple advanced programming guide. It is about 54 dollars.

---

• ***Follow-Ups:***

◆ ***Re: Computer Algebra Algorithms***

◇ *From:* Bernard Parisse

- Prev by Date: ***Re: Animation of rotating axes***
- Next by Date: ***Re: Computer Algebra Algorithms***
- Previous by thread: ***Animation of rotating axes***
- Next by thread: ***Re: Computer Algebra Algorithms***
- Index(es):
  - ◆ ***Date***
  - ◆ ***Thread***