Developing a Computer Algebra System

Dominic Broadbent

The University of Nottingham

What is a Computer Algebra System?

A computer algebra system (CAS) is a piece of software that mimics the paper and pencil symbolic manipulation of mathematical expressions. This distinguishes them from traditional calculators that deal with equations numerically. In general, the chief objective of a CAS is to replace the need for hand computation of arduous or tedious algebraic tasks.

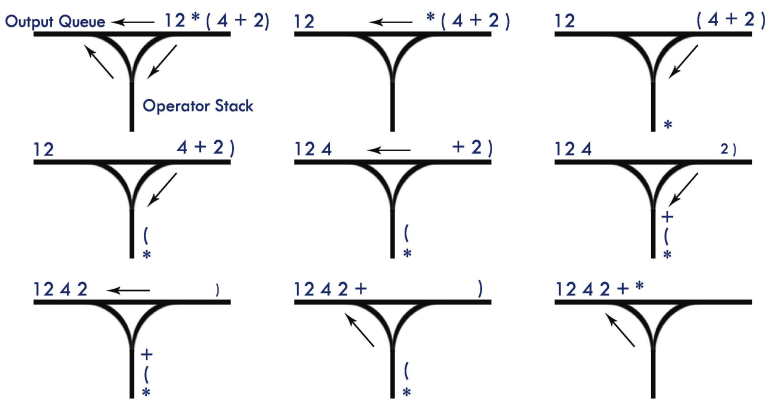
The very first CAS, named 'Schoonschip' after the Dutch expression "schoon schip maken"; literally "to make the ship clean" was developed in 1963 by Nobel laureate Martinus J.G. Veltman. By 1987 CAS had made their way to hand-held calculators and in the present day there are many general-purpose systems both free and commercially available; perhaps the most widely recognised being Wolfram Mathematica.

The Shunting-Yard Algorithm

We write mathematics in *infix* notation; named as such because operators are found *inside* the operands. This notation requires parentheses in order to deviate from the standard order of operations. This notation is very difficult to interpret in a CAS and so we convert to *postfix* notation; named as such because operators are found *after* the operands. Postfix notation doesn't require parentheses as the order of operation is obvious from the order of the expression. An example is as follows:

"12 * (4 + 2) ≡ 12 4 2 + *"

This form is ideal for CAStle as the answer can be computed only working from left to right. The Shunting-Yard algorithm accomplishes this conversion via two data structures; the stack and the queue. These structures hold and release items in specific orders. The queue is First-In-First-Out (FIFO) while the stack is Last-In-First-Out (LIFO). Applying the algorithm to the above example works as follows:



Output expression = 12 4 2 + *

Development of CAStle

Our compter algebra system, called CAStle, was built in the programming language Python. The development process began with a focus on numerical computation. This was in anticipation of calculating composite coefficients of variables. Various algorithms have been implented such as Edsger Dijkstra's Shunting-Yard Algorithm, named as such due to its operation resembling a rail-road shunting-yard. Via these algorithms the system is capable of evaluating numerical expressions, with many available functions.

With computation functionality fully implemented we moved onto the main goal; simplifying compound linear expressions in multiple variables. A compound linear expression is equivalent to a linear expression where the 'variables' can represent simple non-linear expressions with no operations or coefficients. An example is as follows:

"$12z^2xy^{-3} + 4xyz - 7xyxy$"

Many hurdles had to be overcome to accomplish this. Firstly we had to develop a *parser* capable of processing the user input into a standard form. Then we devised algorithms with the capacity to simplify variables and collect alike terms. Most of these algorithms were then adapted to provide contextual step-by-step instructions upon the users request. Below is an example of CAStle's output.

```
Enter your expression:
(4*fac(3)) + (12/3/2)x^2 + 2xyxy + 2x^3yyx^-1

The variable xyxy is equal to: x^2y^2

The variable x^3yyx^-1 is equal to: x^2y^2

The expression has been simplified to: (4*fac(3)) + (12/3/2)x^2 + 2x^2y^2 + 2x^2y^2

The constant term (4*fac(3)) has been interpreted as: (4*fac(3))

fac(3) = 6
Updated expression: (4*6)
4*6 = 24

The expression has been simplified to: 24 + (12/3/2)x^2 + 2x^2y^2 + 2x^2y^2

The coefficient of (12/3/2)x^2 has been interpreted as: ((12/3)/2)

12/3 = 4.0
Updated expression: (4.0/2)
4.0/2 = 2.0

The expression has been simplified to: 24 + 2x^2 + 2x^2y^2 + 2x^2y^2

x^2y^2 term: 2x^2y^2 + 2x^2y^2 = 4x^2y^2

CAStle has simplified your expression to:
24 + 2x^2 + 4x^2y^2
```