Algebraic Simplification, Yet more

Lecture 13

Richardson's class R2

1. rationals and π
2. variable x
3. +, *, /
4. log(u), exp(u), composition

- no sin or abs
- This has a zero equivalence algorithm if we can tell if a constant is zero.

Consider an ordering of subexpressions

- Suppose y is a subexpression of z, then z is more complex than z. Not much more structure is needed.
- Let y be most complex subexpression and suppose it is log(u) for some u.
- Express $F=a_n y^n + \ldots a_1 y + a_0$
- Is a_n, a simpler expression, zero?

a_n is zero

- Let $F_1 = a_{n-1}y^{n-1} + \ldots$, a simpler expression. Test it for zero.

a_n is not zero

- Let $F_1=F/a_n = y^n + \ldots + a_0/a_n$, ok, since a_n is not zero.
- Take the derivative wrt x of F_1, which

looks like $F_2=ny'y^{n-1} + \ldots + (a_n a_0' - a_0 a_n')/a_n^2$.

This expression is simpler because the derivative of log(u) is du/u. If $F_2=0$ then F_1 is a constant. Since it is a constant, test to see if it is zero.

What if y is not log(u)?

- Let y be most complex subexpression and suppose it is exp(u) for some u.
- Express $F=a_n y^n + \ldots a_1 y + a_0$
- Is a_0, a simpler expression, zero?

- If a_0 is not zero, divide through by it to get $F_1=a_n y^n+ \ldots+a_1 y+1$. This might not be simpler, but compute its derivative, $F_2=b_n y^n+..+b_1 y+0$. Now we can divide through by y, and get something like $b_n y^{n-1}+ \ldots+b_1$ which IS simpler. Note that exp(u) is assumed non-zero since u is supposedly defined and not -1 . Is F_2 constant?

What if a_0 is zero?

- divide through by y to get $F_1=a_n y^{n-1}+ a_1$, a simpler expression. Since $F=a_0*F_1$, if F_1 is zero, so is F.

Generalizations and speculations

- What if the only thing we need to add more functions to this list is to have a sufficiently simple defining equation,
- df/dx = f defines exponential
- df/dx = 1/x defines log

- The speculation is that all the properties of special functions of physics can be derived "automatically" by a computer system which could then use these properties to build a simplifier. (Not just zero-equivalence test as given here.) See ESF project, INRIA