

## Theorem Proving: Additional Web Material

Automated theorem proving with computers has an incredibly long and rich history. Wikipedia, of course, is a great starting point

[http://en.wikipedia.org/wiki/Automated\\_theorem\\_proving](http://en.wikipedia.org/wiki/Automated_theorem_proving)

An alternative take is here:

<http://www.cs.miami.edu/~tptp/OverviewOfATP.html>

There are many, many automated proof systems, and there are many highly “disparate” camps in the theorem proving world, often oriented more around systems that, in principle, focus primarily on one technique, but in practice tend to all embrace a wide range of overlapping techniques. In these communities ideas are shared primarily through conferences and papers, so it tends to be quite academic in its approach, though a good number of industrial applications of theorem proving exist.

Some simple examples of using an F#-like language to do some automatic theorem proving is provided by John Harrison here:

<http://www.cl.cam.ac.uk/~jrh13/atp/index.html>

Functional programming is particularly related to *equational theorem proving* and *term rewriting systems*. One of the famous equational proof engines is “E” (<http://www4.informatik.tu-muenchen.de/~schulz/WORK/e prover.html>).

Functional programming is also closely associated with “interactive” theorem proving. For example, see

HOL: <http://hol.sourceforge.net/>

Isabelle: <http://isabelle.in.tum.de/>

Coq: <http://coq.inria.fr/>

HOL Light: <http://www.cl.cam.ac.uk/~jrh13/hol-light/>

ACL2: <http://www.cs.utexas.edu/users/moore/acl2/>

PVS: <http://pvs.csl.sri.com/>

[Formalizing 100 theorems in HOL Light](#) is a particularly interesting lightweight tutorial introduction.

The paper *Floating Point Verification in HOL Light: The Exponential Function* showing a case study of applying HOL Light to Floating Point Verification is here:

<http://portal.acm.org/citation.cfm?id=349276>