Odds and Ends

Collberg's ADT

The Prolog 1000

Lots of Prologs

Ruby meets Prolog

Collberg's *Architecture Discovery Tool*

In the mid-1990s Christian Collberg developed a system that is able to discover the instruction set, registers, addressing modes and more for a machine given only a C compiler for that machine.

The basic idea is to use the C compiler of the target system to compile a large number of small but carefully crafted programs and then examine the machine code produced for each program to make inferences about the architecture. The end result is a machine description that in turn can be used to generate a code generator for the architecture.

The system is written in Prolog. What do you suppose makes Prolog well-suited for this task?

See http://www.cs.arizona.edu/~collberg/Research/AutomaticRetargeting/index.html for more details.

The Prolog 1000

The Prolog 1000 is a compilation of applications written in Prolog and related languages. Here is a sampling of the entries:

AALPS

The Automated Air Load Planning System provides a flexible spatial representation and knowledge base techniques to reduce the time taken for planning by an expert from weeks to two hours. It incorporates the expertise of loadmasters with extensive cargo and aircraft data.

ACACIA

A knowledge-based framework for the on-line dynamic synthesis of emergency operating procedures in a nuclear power plant.

ASIGNA

Resource-allocation problems occur frequently in chemical plans. Different processes often share pieces of equipment such as reactors and filters. The program ASIGNA allocates equipment to some given set of processes. (2,000 lines)

The Prolog 1000, continued

Coronary Network Reconstruction

The program reconstructs a three-dimensional image of coronary networks from two simultaneous X-Ray projections. The procedures in the reconstruction-labelling process deal with the correction of distortion, the detection of center-lines and boundaries, the derivation of 2-D branch segments whose extremities are branching, crossing or end points and the 3-D reconstruction and display.

All algorithmic components of the reconstruction were written in the C language, whereas the model and resolution processes were represented by predicates and production rules in Prolog. The user interface, which includes a main panel with associated control items, was developed using Carmen, the Prolog by BIM user interface generator.

DAMOCLES

A prototype expert system that supports the damage control officer aboard Standard frigates in maintaining the operational availability of the vessel by safeguarding it and its crew from the effects of weapons, collisions, extreme weather conditions and other calamities. (> 68,000 lines)

The Prolog 1000, continued

DUST-EXPERT

Expert system to aid in design of explosion relief vents in environments where flammable dust may exist. (> 10,000 lines)

EUREX

An expert system that supports the decision procedures about importing and exporting sugar products. It is based on about 100 pages of European regulations and it is designed in order to help the administrative staff of the Belgian Ministry of Economic Affairs in filling in forms and performing other related operations. (>38,000 lines)

GUNGA CLERK

Substantive legal knowledge-based advisory system in New York State Criminal Law, advising on sentencing, pleas, lesser included offenses and elements.

MISTRAL

An expert system for evaluating, explaining and filtering alarms generated by automatic monitoring systems of dams. (1,500 lines)

The full list is in `fall06/pl/Prolog1000.txt`. Several are over 100,000 lines of code.

Lots of Prologs

For an honors section assignment Maxim Shokhirev was given the task of finding as many Prolog implementations as possible in one hour. His results:

1. DOS-PROLOG
http://www.lpa.co.uk/dos.htm
2. Open Prolog
http://www.cs.tcd.ie/open-prolog/
3. Ciao Prolog
http://www.clip.dia.fi.upm.es/Software/Ciao
4. GNU Prolog
http://pauillac.inria.fr/~diaz/gnu-prolog/
5. Visual Prolog (PDC Prolog and Turbo Prolog)
http://www.visual-prolog.com/
6. SWI-Prolog
http://www.swi-prolog.org/
7. tuProlog
http://tuprolog.alice.unibo.it/
8. HiLog
ftp://ftp.cs.sunysb.edu/pub/TechReports/kifer/hilog.pdf
9. ?Prolog
http://www.lix.polytechnique.fr/Labo/Dale.Miller/lProlog/
10. F-logic
http://www.cs.umbc.edu/771/papers/flogic.pdf
11. OW Prolog
http://www.geocities.com/owprologow/
12. FLORA-2
http://flora.sourceforge.net/
13. Logtalk
http://www.logtalk.org/

DataLog implementations:
---
14. bddbddb
http://bddbddb.sourceforge.net/
15. ConceptBase
http://www-i5.informatik.rwth-aachen.de/CBdoc/
16. DES
http://www.fdi.ucm.es/profesor/fernan/DES/
17. DLV
http://www.dlvsystem.com/
18. XSB
http://xsb.sourceforge.net/

19. WIN Prolog
http://www.lpa.co.uk/
20. YAP Prolog
http://www.ncc.up.pt/~vsc/Yap
21. AI::Prolog
http://search.cpan.org/~ovid/AI-Prolog-0.734/lib/AI/Prolog.pm
22. SICStus Prolog
http://www.sics.se/sicstus/

23. ECLiPSe Prolog
http://eclipse.crosscoreop.com/
24. Amzi! Prolog
http://www.amzi.com/
25. B-Prolog
http://www.probp.com/
26. MINERVA
http://www.ifcomputer.co.jp/MINERVA/
27. Trinc Prolog
http://www.trinc-prolog.com/
28. Strawberry Prolog
http://www.dobrev.com/
29. hProlog
http://www.cs.kuleuven.ac.be/~bmd/hProlog/
30. ilProlog
http://www.pharmadm.com/dmax.asp
31. CxProlog
http://ctp.di.fct.unl.pt/~amd/cxprolog/
32. NanoProlog
http://ctp.di.fct.unl.pt/~amd/cxprolog/
33. BinProlog
http://www.binnetcorp.com/BinProlog/
34. Quintus Prolog
http://www.sics.se/quintus/
35. Allegro Prolog
http://www.franz.com/products/prolog/
36. ALS Prolog
http://www.als.com/
37. W-Prolog
http://ktiml.mff.cuni.cz/~bartak/prolog/testing.html
38. Aquarius Prolog
listserv@acal-server.usc.edu

http://www.exa.unicen.edu.ar/~azunino/javalog.html
54. jProlog
http://www.cs.kuleuven.ac.be/~bmd/PrologInJava/
55. tiny_prolog.rb (Ruby implementation)
http://swik.net/Ruby/Inside+Ruby/A+Basic+Prolog+Implementation+in+Ruby/ofun
56. WAM (Warren Abstract Machine)
http://64.233.187.104/search?q=cache:fgeYFaPXclIJ:www.clip.dia.fi.upm.es/~logalg/slides/PS/8_wam.ps+Prolog+Implementation+list&hl=en&gl=us&ct=clnk&cd=49
57. Allegro Prolog
http://www.franz.com/support/tech_corner/prolog-071504.lhtml
58. NU-Prolog
http://www.cs.mu.oz.au/~lee/src/nuprolog/
59. Mimola Software System (MSS)
citeseer.ist.psu.edu/22901.html
60. LogTalk
http://www.logtalk.org/
61. Palm Toy Language
http://www.geocities.com/willowfung/
62. Qu-Prolog
http://www.itee.uq.edu.au/~pjr/HomePages/QuPrologHome.html
63. K-Prolog
http://prolog.isac.co.jp/doc/en/
64. ProFIT
http://www.coli.uni-sb.de/~erbach/formal/profit/profit.html
65. Arity/Prolog
http://www.arity.com/www.pl/products/ap.htm
66. Brain Aid Prlog
http://www.comnets.rwth-aachen.de/~ost/private.html

39. EZY Prolog
http://www.ezy-software.com/ezyprolog/EZY_Prolog_-_born_to_be_easy.htm
40. Poplog
http://www.cs.bham.ac.uk/research/poplog/freepoplog.html
41. OpenPoplog
http://openpoplog.sourceforge.net/
42. REBOL Prolog
http://www.rebol.org/cgi-bin/cgiwrap/rebol/view-script.r?script=prolog.r
43. Arity Prolog
pgweiss@netcom.com
44. CHIP
http://www.cosytec.com/
45. IF/Prolog
http://www.ifcomputer.com/IFProlog/
46. LPA Prolog
http://www.lpa.co.uk/
47. cu-Prolog
ftp://ftp.icot.or.jp/pub/cuprolog/
48. Tricia
ftp://ftp.csd.uu.se/pub/Tricia/
49. Visual Prolog Personal Edition
http://www.visual-prolog.com/vip/vipinfo/freeware_version.htm
50. JLog
http://jlogic.sourceforge.net/
51. JIProlog
http://www.ugosweb.com/jiprolog
52. CKI-Prolog
http://www.students.cs.uu.nl/~smotterl/prolog/index.htm
53. JavaLog

67. Reform Prolog
http://user.it.uu.se/~thomasl/reform.html
68. CMU Free/Shareware Prolog
http://www-2.cs.cmu.edu/afs/cs/project/ai-repository/ai/lang/prolog/impl/prolog/0.html
69. Babol Prolog
http://zamek.gda.pl/~budyn/
70. INRIA wamcc
ftp://ftp.inria.fr/INRIA/Projects/contraintes/wamcc/
71. PyProlog
http://sourceforge.net/projects/pyprolog/
72. Artiware Prolog
www.artiware.com
73. DGKS Prolog
www.geocities.com/SiliconValley/Campus/7816/

Ruby meets Prolog

http://eigenclass.org/hiki.rb?tiny+prolog+in+ruby describes a "tiny Prolog in Ruby".

From that page, here is `member`:

```
member[cons(:X,:Y), :X] <<= :CUT
member[cons(:Z,:L), :X] <<= member[:L,:X]
```

Here's the standard family example:

```
sibling[:X,:Y] <<= [parent[:Z,:X], parent[:Z,:Y], noteq[:X,:Y]]
parent[:X,:Y] <<= father[:X,:Y]
parent[:X,:Y] <<= mother[:X,:Y]

# facts: rules with "no preconditions"
father["matz", "Ruby"].fact
mother["Trude", "Sally"].fact
...

query sibling[:X, "Sally"]
# >> 1 sibling["Erica", "Sally"]
```