# AILog

David Poole
Department of Computer Science,
University of British Columbia,
http://www.cs.ubc.ca/~poole

June 2, 2011

AILog2 (formerly CILog) is an open-source purely declarative representation and reasoning system, that includes pure Prolog (including negation as failure) and allows for probabilistic reasoning. It is available from `http://artint.info/code/ailog/ailog2.html`. It is intended as a pedagogical tool to present a simple logic that can be used for AI problems. It is designed for programming in the small, where you can axiomatize a domain, ask questions and debug the knowledge base, without knowing how answers are produced. It implements the logic examples from the books *Artificial Intelligence: foundations of computational agents* [Poole and Mackworth, 2010] and *Computational Intelligence: A Logical Approach* [Poole, Mackworth, and Goebel, 1998]. The probability is based on the Independent Choice Logic [Poole, 2008] which can represent Bayesian networks, Markov decision processes and complex mixes of logic and probability. Many examples from the research literature are provided.

It uses Prolog's syntax for variables, function symbols and predicates, but uses `<-` for "if", `&` for "and" and `~` for negation as failure. This is to emphasize that it is not Prolog. It has minimal built-in predicates to get students to think logically about the problem rather than looking in the user manual. As long as a program is acyclic (so that ground queries eventually terminate), a program means its Clark's completion. So all of the logic programs can be interpreted logically with their normal semantics, even when there are probabilities.

It implements the Independent Choice Logic [Poole, 2008], that allows (independent) probabilistic inputs to the logic program. This turns out to be a powerful and intuitive way to incorporate probabilities into programming languages [Poole, 2010]. It allows for incremental conditioning on arbitrary propositions and queries on arbitrary propositions. The user can explore the explanations (corresponding to the proofs of the observations and queries), as well as compact representations of the possible worlds from which the probabilities can be computed.

AIlog includes:

- a definite clause representation and reasoning system
- a simple tell-ask user interface, where the user can tell the system facts and ask questions of the system
- explanation facilities to explain how a goal was proved, why an answer couldn't be found, why a question was asked, why an error-producing goal was called, and why the depth-bound was reached. These provide knowledge-level debugging tools to let the user debug incorrect answers, missing answers, system errors, and possible infinite loops.
- depth-bounded search, that can be used to investigate potential infinite loops and used to build an iterative-deepening search procedure
- sound negation-as-failure, that interacts appropriately with the depth-bound
- ask-the-user facilities
- assumables for finding conflicts (e.g., in consistency based diagnosis) and for abduction
- probabilistic reasoning, integrated with negation as failure and the depth-bounded search.

# References


Poole, D., Mackworth, A., and Goebel, R. (1998). *Computational Intelligence: A Logical Approach*. Oxford University Press, New York.

Poole, D. (2008). The independent choice logic and beyond. In L. De Raedt, P. Frasconi, K. Kersting, and S. Muggleton (Eds.), *Probabilistic Inductive Logic Programming: Theory and Application*, LNCS 4911. Springer Verlag. URL `http://cs.ubc.ca/~poole/papers/ICL-Beyond.pdf`.

Poole, D. (2010). Probabilistic programming languages: Independent choices and deterministic systems. In R. Dechter, H. Geffner, and J. Halpern (Eds.), *Heuristics, Probability and Causality: A Tribute to Judea Pearl*, pp. 253–269. College Publications.

Poole, D.L. and Mackworth, A.K. (2010). *Artificial Intelligence: foundations of computational agents*. Cambridge University Press, New York, NY. URL `http://artint.info`.