PL Project

PROLOG AND AI

by Imt2018011 - Arem Venkata Karthik Reddy,
Imt2018019 - Chebrolu Suchith Kumar &
Imt2018083 - Venkata Sai Gopal Sundari.
On May 19, 2021.

ARTIFICIAL INTELLIGENCE:

Artificial intelligence is the getting of computers to do things that seem to be intelligent. The hope is that more intelligent computers can be more helpful to us--better able to respond to our needs and wants, and more clever about satisfying them.

But "intelligence" is a vague word. So artificial intelligence is not a well-defined field. One thing it often means is advanced software engineering, sophisticated software techniques for hard problems that can't be solved in an easy way. Another thing it often means is non-numeric ways of solving problems since people can't handle numbers well. Non-numeric ways are often "common sense" ways, not necessarily the best ones. So artificial intelligence programs--like people--are usually not perfect and even make mistakes.

Artificial intelligence includes:

Getting computers to remember complicated interrelated facts, and draw conclusions from them (inference), Getting computers to plan sequences of actions to accomplish goals (planning), Getting computers to offer us advice based on complicated rules for various situations (expert systems).

The key issue in artificial intelligence is reductionism, the degree to which a program fails to reflect the full complexity of human beings. Reductionism includes how often program behavior duplicates human behavior and how much it differs when it does differ. Reductionism is partly a moral issue because it requires moral judgments.

PROLOG:

Prolog is a logical and declarative programming language. The name itself, Prolog, is short for PROgramming in LOGic. Prolog is intended primarily as a declarative programming language. In prolog, logic is expressed as relations (called Facts and Rules). The core heart of prolog lies at the logic being applied. Formulation or Computation is carried out by running a query over these relations. In prolog, We declare some facts. These facts constitute the Knowledge Base of the system. We can query against the Knowledge Base. We get output as affirmative if our query is already in the knowledge base or it is implied by Knowledge

Base, otherwise, we get output as negative. So, Knowledge Base can be considered similar to a database, against which we can query. Prolog facts are expressed in a definite pattern. Facts contain entities and their relation. Prolog extensively uses expressions of logic instead of developing a program by providing a specific sequence of instructions to the computer.

For Example:

```
Facts:                          English meanings
food(burger)                    # burger is a food
food(sandwich)                  # sandwich is a food
food(pizza)                     # pizza is a food
lunch(sandwich)                 # sandwich is a lunch
dinner(pizza)                   # pizza is a dinner

Rules:
meal(X) :- food(X).             #Every food is a meal or Anything is a meal if it is a food

Queries / Goals:
?- food(pizza).                 # Is pizza a food?
Prolog returns >>Yes

?- meal(X), lunch(X).           # Which food is meal and lunch?
Prolog returns >> X := sandwich.

?- dinner(sandwich).            #  Is sandwich a dinner?
Prolog returns >>No
```

PROLOG AND AI:

Prolog has a clear contribution to solving a series of AI problems. There are many features that make Prolog suitable as a programming language for developing AI applications. Some of them are:

Declarative nature:

Logical programming removes the imperative nature of other languages and allows programmers to solve a problem by describing the problem itself rather than defining a solution. The programmer writes programs by declaring the facts and the rules that apply to the problem at hand and then makes queries in order for Prolog to return valid solutions. This high level of abstraction makes Prolog suitable for AI applications where the programmers should give emphasis on the problem. Prolog's built-in search mechanism takes the control of the program execution, leaving the programmer to concentrate on the problem.

Backtracking:

When a search path ends at a dead end, the backtracking mechanism of Prolog retreats back down the search path to try another path. This feature makes Prolog exceptionally suitable for a number of search problems that AI faces.

It also helps in finding more than one solution to the problem. In that case, backtracking is forced after finding the first solution. Because a great number of AI problems can be represented as a problem of finding the right path in the search space, the built-in depth-first mechanism of Prolog, accompanied by backtracking, becomes very powerful.

Pattern matching and Unification:

The use of unification to find the most general common instance of two formulas or patterns makes pattern matching a build-in feature of Prolog. This intelligent feature can assist AI problem solving where in many cases the decisions that are made are based on situation matching. This Prolog ability can be found really helpful in specific areas of AI, such as natural language processing, intelligent database search, and some others.

Including these three features, recursion instead of iteration is also helpful as many AI problems are recursive in nature, increasing the suitability of Prolog and solving problems recursively makes Prolog programs smaller and understandable even when coping with large AI problems.

Prolog is ideal for creating meta-interpreters or interpreters written in Prolog that can interpret subsets of Prolog code. Using this property we are able to write interpreters for expert systems, machine learning using explanation-based learning models, etc, which are all huge contributions in the field of AI.

List handling mechanism:

The data structure of List is very important for handling AI problems (e.g., LISP which is also used for solving AI problems, stands for " LIST Processing"). Lists are built-in in Prolog while in most other languages they are not, making faster and easier the writing of Prolog programs that require list handling. List's recursive nature allows the extensive use of recursion in problem-solving, providing an additional advantage for solving AI problems with Prolog.

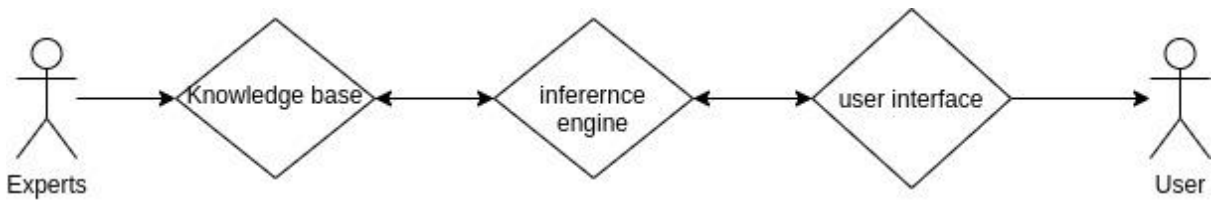
Don't care, and don't know nondeterminism:

In the execution of a Prolog program, the nondeterminism feature is really apparent. Although the rule that will execute is the first one that matches the goal, we can ask for more than one solution. Using the backtracking mechanism, alternative rules will apply and other valid solutions will be found. This introduces a) "don't know nondeterminism" implying that all possible ways to find the solutions will be followed because the execution does not know how to find the solution, or b) "don't care non-determinism" meaning that we just need one solution and we do not care which solution is that among the many that exist. Both of these types of nondeterminism are considered useful in AI applications, especially for those dealing with logic.

Experts systems:

Expert System:- Knowledge + problem-solving strategies

It consists of a knowledge base that can capture the domain-specific knowledge and an inference engine that consists of algorithms for manipulating the knowledge represented in the knowledge base to solve a problem presented to the system.



Knowledge Base:

It is required to exhibit intelligence that contains domain-specific knowledge.
It contains both factual(information widely accepted by the experts) and heuristic(accurate judgment) knowledge.

The success of any Expert system depends upon the collection of a highly accurate and precise knowledge base.

Inference Engine:

Forward chaining: It follows the chain of conditions and derivations and finally deduces the outcome. It considers all the facts and rules and sorts them before concluding with a solution.

Backward Chaining: On the basis of what has already happened, the inference engine tries to find out which conditions could have happened. This strategy is followed for finding out cause or reason.

User Interface:

It provides interaction between Expert systems and the end-users who might not be experts. It helps users to accomplish their goals in the shortest possible way.

Github repository link of the project:
https://github.com/suchithkumarch/Prolog_and_AI

Thank you.