Knowledge Representation Laboratory 1: Getting Started with Prolog

CITS3005

There are a number of ways to set up a prolog environment. The aim of today's laboratory is to run try some different environments and find one that is suitable for your circumstances.

1 Flavours of Prolog

There are several flavours of prolog to consider:

1. SWI-prolog is probably the most mature implementation, and contains new features like dictionaries. It is probably the most well known version, and has useful integrations for later in the unit.
2. GNU-prolog is a simpler implementation, and conforms with the standard. It's probably the easiest to get started with.
3. Problog and other variations are semantic variations of Prolog. Problog is written in python, and associates probabilities with clauses.

2 Setting up Prolog

1. **Prolog in the laboratories**. SWI-Prolog is installed on Laboratory machines. You can find it in the development tools menu on windows, and start a console running.
2. **SWISH Prolog in the browser**. A prolog environment has been built in JavaScript to allow you to run simple programs. It also comes with a notebook environment, similar to Jupyter.
3. **Run SWI-Prolog in a Docker container**. Docker presents a convenient way to build and run prolog programs in a stable environment. This can be useful if you want to build a stable environment is vscode for example. The initial set-up can be a little tricky, so please feel free to ask questions advice.
4. **Install SWI-Prolog on your own machine**. The SWI-Prolog website https://www.swi-prolog.org/ has instructions for downloading and building SWI-prolog.
5. **Install GNU-Prolog on your own machine**. The build instructions for Gnu-Prolog are less complex than for SWI-Prolog, and GNU-prolog will be adequate for the first half of the unit.

3 A Tutorial

There is a good introductory tutorial, written for Gnu-Prolog, but will work for SWI and other variations as well. *Aim to work through the first five pages of the tutorial this week.*

4 A first program

Develop a simple scenario based on your academic record (or some fictitious academic record). The knowledge base should contain:

- a unary predicate for students (e.g. jane, tim, cris);
- a unary predicate for units (e.g. cits3005, cits2211);
- a binary predicate for prerequisite (e.g. cits2211 is a prerequisite for cits3005);
- a mark predicate giving the mark for a student in a unit;
- and a rule indicating whether a student is eligible to take a unit.