Introduction

This is a rule-based expert system shell. It can be used for diagnostics among other things.

It was a university project done by Osama Bodiaf (@OsamaBodiaf), for a sixth semester's AI class.

Components

The software consists of three main components:

- Knowledge Base
  This provides storage for the hypotheses, facts, and rules used by the Inference Engine.

- Inference Engine
  This is the brain of the software, it makes use of the information available in the Knowledge Base via a forward-chaining algorithm to learn new facts, negate or verify existing hypotheses, and through a backward-chaining algorithm to query the truthness of a particular statement.

- Graphical User Interface
  This is the interface through which the user interacts with the software, it contains a built-in text editor for altering the Knowledge Base, plus ways to invoke the Inference Engine with or without arguments, and display the results.

Knowledge Base

- Hypotheses and Facts are referred to as Atoms, and each one of them is expected in one of the following formats:

  <key>

  <key>: <statement>

  <key>: <statement>. <truth value>

  - Each <*key*> should be unique.
  - If <*statement*> is not provided, <*key*> will be used as <*statement*> too, internally.
  - Using a short indicative <*key*>, along with an explicit descriptive <*statement*>, makes the user experience much simpler when providing goals to query, or to learn until achieved.
  - <*truth value*> is not case-sensitive.
  - If <*truth value*> is provided, and its lowercase transformation equals *false*, *faux*, *f*, *no*, *non*, *n*, or *0* then it is considered *False*; Any other value is considered *True*.
  - If <*truth value*> is not provided, it is considered *True*.
  - <*key*> refers to: the <*statement*> and its corresponding <*truth value*>, as a couple.
  - Examples:
    - WH
    - Weather is hot
    - WH: Weather is hot. True
    - # The three examples above are equivalent.
    - B: Busy
    - GB: Go to beach
    - D: AI is dull. 0
    - # The example above is equivalent to: “D: It is false that AI is dull”.
    - PF: Programming is fun!. Ce n’est pas vrai?

- Rules are expected in the following format:

if *<antecedent>* then *<consequent>*

  - *<antecedent>* is a valid logical expression of *<key>*s separated by one of the three main *<logical operator>*s.
  - Supported *<logical operator>*s are: *and, or,* and *not*.
  - *<consequent>* should be a *<key>* optionally preceded by *not*.
  - Examples:
    - if Weather is hot and not B then GB
    - if PF then D
    - if not B and not D then not PF
    - # There are many ways to provide information to the Knowledge Base!

- Comments are possible: each line starting with # is ignored; This is useful when wanting to exclude an Atom or Rule from consideration without actually deleting it.

- Extra white-space is ignored.

Inference Engine

- Forward-chaining algorithm is irrevocable, uses a depth-first search strategy, and favors rules with most premises in selection.

- Backward-chaining algorithm is attemptive, uses a depth-first search strategy, and favors rules filtered first in selection.

- Some decisions are logged, to provide step-by-step explanation about reasoning, when needed.

- The Inference Engine may alter the visual representation of the information included within the Knowledge Base, without modifying its essence.

Graphical User Interface

- The built-in text editor makes it possible to view and directly alter the Knowledge Base, with options to cancel or save the changes.

- The Learn frame provides a way to invoke the Inference Engine, in order to learn as many facts as possible, or until a goal is reached.

- The Query frame can be used to invoke the Inference Engine, for an attempt to deduce the truthness of a given key's statement.

- The GUI also contains frames to show or request additional information, prompt user action, and to provide explanation depending on the situation.