CSCI-630: 01 – Foundations of Intelligent Systems

Project report – An expert system for climate control

Author: Shreyas Jayanna (sj7316@rit.edu)

This report details the summary, requirements, specifications, description of the problem domain, feasibility study, implementation and testing involved in the development of the climate control expert system. At the end of the report is a user guide which details the actions to be performed to successfully run the expert system.

1. Executive summary:

The expert system tries to simulate an office environment where each occupant of the room has a different temperature and humidity preferences. The expert system takes every occupant's preferences into consideration and maintains the indoor environment at the best possible condition while being energy efficient in doing so.

The results of the expert system is an interactive prolog shell console which takes inputs and displays the best indoor environment configuration that is being set.

2. Requirements:

The project requires five major aspects:

1. **Identify the problem and analyze the knowledge to be included into the system:**

   The problem is identified as a climate control expert system for an office setting which monitors and maintains the temperature and humidity at the desired levels.

   The knowledge to be included in the system is the rules and decisions that the expert system has to take based on the surrounding environment conditions while being energy efficient.
2. **Choose an implementation tool:**

   The implementation tool chosen is PROLOG. PROLOG is an excellent rule based tool. Although the documentation of PROLOG is a little confusing, once it is learnt, its actual power can be utilized to develop a good expert system.
3. **Design an expert system:**

   Design of the expert system involves designing the rules and knowledge base. Different environment conditions are taken into account while designing the expert system.
4. **Develop an expert system prototype:**

Development of the expert system is carried out in two steps. The first step involved the prototype development. The prototype covered only few basic rules for temperature control.

5. **Expand, test and revise your expert system prototype:**

   In the expansion phase, the expert system prototype was further developed into a complete expert system which monitors the temperature and humidity of the indoor environment which taking into consideration the outdoor environment as well as being energy-efficient in doing so.

3. Specification:

The system monitors and maintains the indoor environment conditions of an office space. The system takes decisions such that minimal energy is used in achieving the desired indoor conditions.

1. **Input**: The input is taken from the user through the shell. The user is presented with options to choose from to alter the indoor conditions such as resetting temperature, humidity. The user is also presented with an option to enter the outdoor temperature as the system takes outdoor temperature into account while making energy efficient decisions to maintain the indoor conditions favorable. In a real-time setting, the outdoor temperature can be recorded through sensors.

   When the user chooses to alter the temperature, either one or multiple temperature preferences can be provided. The system decides what temperature to be set while considering the indoor temperature at that moment and the outdoor temperature.

   The user can also alter the humidity of the indoor environment. The system takes one or more than one preference for the humidity and sets it to the most favorable condition for all occupants.
2. **Decision making**: After taking the inputs, the system calculates the mean and median values of the desired temperatures and humidity values and based on the rules defined, sets the temperature to the most favorable one to all occupants.
3. **Rules**: There are 12 rules and 10 cases that the system evaluates while setting the temperature. The system evaluates 2 rules for setting the humidity of the office space.
4. **Output**:
   a. The system, after evaluating the rules, sets the temperature to the most favorable one to all occupants and displays a message saying what decision it has made and how it is going to achieve it.
   b. While altering the humidity, the system evaluates the rules and displays appropriate messages as to what action is being taken to achieve the desired humid indoor conditions.

4. Description of the problem domain:

Prolog has been chosen to develop the expert system with. Prolog is a logic programming language. It is declarative and is associated mainly with artificial intelligence. The program logic is expressed as facts and rules. The execution of the code involves evaluating the rules to determine the facts. Prolog follows chronological backtracking strategy which makes it more powerful to use for developing expert systems.

5. Feasibility study:

The different domain options considered while developing the project were Jess, CLIPS and Prolog.

1. Jess: Jess is the Java Expert System Shell. Although Jess is somewhat a clone of CLIPS, it brings in the Java flavor in it. It is available free of cost for non-commercial uses which made it one of the contenders for this project. Jess comes with a readme HTML file which describes how to set it up. The file contains few methods as well but the documentation for JESS is quite poor compared to CLIPS and Prolog. In fact, the readme file instructs the user to read the manual of CLIPS before using Jess. Jess does not implement modules which was needed very much in this project. Although the Java language implementation of Jess makes it favorable, it was found to be a tad confusing to use Jess and hence, Jess was dropped.
2. CLIPS: CLIPS stands for C Language Integration Production System. CLIPS is a freely available tool for non-commercial use. The CLIPS user manual describes easily how to use CLIPS and is easy to follow. Upon a small research, it was found that CLIPS is very similar to LISP. CLIPS shell provides basic components of an expert system like facts-list, rules, inference engine, etc.. Although CLIPS seems to be fairly easier to use, it was dropped with the intention of learning Prolog which is new.
3. Prolog: Prolog as mentioned in section 4, is a logic programming language. It mainly follows backward chaining but forward chaining can be achieved too which is intended in this project. Prolog allows distinction of rule facts and just facts. The manual for Prolog is extensive and it seemed interesting to use. Although Prolog can be confusing at times, the fact that it is a powerful tool for developing expert systems overpowered the confusion. SWI Prolog is a Prolog shell which is freely available to work with Prolog. SWI Prolog comes with a rich manual which details the usage as well as a tutorial which comes in handy while using Prolog beginners.

Conclusion: Developing this expert system was an opportunity to learn something new. There was no previous knowledge about Prolog until now. Prolog is chosen as the domain tool to be used for all the advantages mentioned in this section and to learn developing expert systems in Prolog.

6. Implementation:

1. Representation of the knowledge base: The knowledge base is represented basically as rule_fact(s) which are true when certain rules are satisfied. For example, to turn off the AC, the following rules have to be met:
    a. The current indoor temperature is less than the outdoor temperature.
    b. The desired temperature should be greater than the current temperature.
    c. The desired temperature should be lesser than the outdoor temperature.

2. Expert system applications: The expert system can be used to monitor and maintain the indoor environment conditions of a room where multiple occupants desire different temperature and humidity settings. The expert system can be deployed where there is a need for an energy efficient climate control system.
3. Structure: The expert system consists of dynamic predicates, regular predicates, clauses, rules and rule-facts.
4. User-interface: The user interface is an interactive shell. The shell interacts with the user and takes specific inputs to maintain the indoor environment conditions at a favorable level.
5. Limitations: Real-time sensors could not be used in the project because of various constraints and hence every input has to be taken through the user. This is considered one of the main limitations of this project. One more limitation of this project is that it doesn't try to control the indoor luminosity conditions. Maintaining the indoor luminosity conditions can favor the temperature control to some extent. Also, the system does not consider extreme conditions like temperature more than 110F in which case, it should trigger an emergency mechanism to have the place vacated by all the occupants as such conditions are dangerous for the well-being of occupants.
6. Software requirements: Prolog shell is a software requirement to execute this project. SWI Prolog which is used in the project can be downloaded freely from 'http://www.swi-prolog.org/' for non-commercial use. Various versions of the shell are available for use with Linux, Windows and MAC OS.
7. Hardware requirements: The shell requires minimal hardware configurations which can be found on any of today's computer systems in use.

7. Testing description:

The testing involved simulating various conditions and checking how the system responds to user inputs. Various conditions like the ones listed below were tested:

1. Outdoor temperature greater than indoor temperature
2. The desired temperature close to indoor temperature
3. The desired temperature close to outdoor temperature
4. The desired temperature is a range of values
5. Indoor temperature greater than outdoor temperature
6. The mean of the input values close to indoor temperature
7. The median of the input values close to outdoor temperature, etc.

The system performed pretty well for all the tested conditions.

8. User's guide:

The following steps may be followed to execute the expert system.

1. The Prolog shell needs to be installed.
2. On the shell, the following command needs to be executed:
   consult('pr1_shreyas-jayanna.pl').

Make sure to include the dot at the end of the command. All queries or commands need to need with a period for the interpreter to execute it.
This command will compile the prolog code.

3. Type the following command to start the execution:
   go.
   'go' is a predicate in the code which starts the execution.
4. Once the code starts execution, follow the instructions shown on the shell to play around with the expert system.