Artificial Intelligence – Lab Assignment 3

UNIZG FER, academic year 2013/14

Handed out: May 12. Due: May 21 at 23:59

The topic of this lab assignment are rule based systems. The task is to implement an expert system shell that enables inference using backward chaining of production rules. You can implement this task in a programming language of your choice.

Expert system shell

The expert system shell implements an inference engine that uses a knoweldge base (consisting of facts and rules) to derive new knowledge. The antecedent and consequent of each production rule consist of an arbitrary number of attribute-value pairs, conjoined with conjunctions. On the left hand side of a rule there can be more than one value for a given attribute; in that case a disjunction between such values is assumed. E.g.,

```
RULE myRule01 SALIENCE 2
IF atr1 = v11 & atr2 = v21|v22 THEN atr3 = v3 & atr4 = v4
```

You need to represent the production rules in memory using a suitable data structure. You need not implement the generation of this structure from text representation (parsing).

You should implement a conflict resolution strategy based on rule order and rule priorities. Each rule should have a numerical priority (salience) value assigned to it, which should be used for resolving conflicts. The rule with a greater priority value will have precedence over rules with a lower priority value. Additionally, if two rules have the same priority value, then precedence is given to the first listed rule.

At the very start, the user must specify the attribute whose value is to be derived. Queries about values of attributes that cannot be derived using the knowledge base alone are to be answered by the user during program execution. After starting, the program should print out the knowledge base (the rules and the facts). After each step, the program should print out the state of memory, the set of conflicting rules, and the identifiers of rules that fired.

Example

To check that the shell works correctly, use the small diagnostic expert system described next. The system is used to detect the cause of a car malfunction. The system variables are as follows:

```
FuelMeter         =  present | notPresent
ReservoirStatus   =  empty | notEmpty
BatteryStatus     =  OK | weak
BatteryYears      =  <=2 | 2-4 | >=4
BatteryAge        =  new | medium | old
```

```
BatteryVoltage     =  >=8 | <8 | unknown
StarterStatus      =  OK | faulty
CarburetorStatus   =  OK | leaks
StartplugsStatus   =  OK | malfunctioning
FuelSmell          =  yes | no
StartSound         =  normal | squeaking | knocking | notPresent
Lights             =  OK | weak | notPresent
EnvironmentTemp    =  <=-7 | >-7
MotorTurns         =  yes | slowly | no
Problem            =  BatteryWeak | BatteryTooCold | carburetorLeaks |
                      StartplugsFaulty | StarterFaulty |
                      ReservoirEmpty
```

The `Problem` variable points to one of the six possible malfunction causes. The `BatteryVoltage` variable can, apart from numerical values, also take an `unknown` value. If the user can measure the battery voltage (he/she has a voltmeter) then the voltmeter reading is taken as a reliable indication of battery function (`BatteryStatus` variable). However, if the voltage is unknown to the user, the expert system can infer the battery status based on the values of other variables.

The functionality of the expert system is based on the following 20 rules (these rules should be encoded in structures that you use to represent the rules, or read into the program from a text file, if you decide to implement a parser):

```
1   IF    FuelMeter = present
      THEN  ReservoirStatus = notEmpty
2   IF    FuelMeter = notPresent
      THEN  ReservoirStatus = empty
3   IF    BatteryYears = <=2
      THEN  BatteryAge = new
4   IF    BatteryYears = 2-4
      THEN  BatteryAge = medium
5   IF    BatteryYears = >=4
      THEN  BatteryAge = old
6   IF    BatteryVoltage = >=8
      THEN  BatteryStatus = OK
7   IF    BatteryVoltage = <8
      THEN  BatteryStatus = weak
8   IF    EnvironmentTemp = <=-7 & BatteryStatus = OK
      THEN  problem = BatteryTooCold
9   IF    MotorTurns = slowly|no & Lights = weak & BatteryVoltage = unknown
      THEN  BatteryStatus = weak
10  IF    MotorTurns = no & Lights = notPresent & BatteryVoltage = unknown
      THEN  BatteryStatus = weak
11  IF    MotorTurns = slowly|no & Lights = OK & BatteryVoltage = unknown &
            BatteryAge = new|medium
      THEN  BatteryStatus = weak
12  IF    MotorTurns = yes & BatteryStatus = OK & FuelSmell = yes
      THEN  CarburetorStatus = leaks
13  IF    MotorTurns = no & StartSound = knocking
      THEN  StartplugsStatus = malfunctioning
14  IF    MotorTurns = no & StartSound = notPresent & BatteryStatus = OK
      THEN  StartplugsStatus = malfunctioning
15  IF    StartSound = squeaking
      THEN  StarterStatus = faulty
```

```
16  IF    ReservoirStatus = empty
    THEN  Problem = ReservoirEmpty
17  IF    BatteryStatus = weak
    THEN  Problem = BatteryWeak
18  IF    RasplinjačStanje = leaks
    THEN  Problem = CarburetorLeaks
19  IF    StartplugsStatus = malfunctioning
    THEN  Problem = StartplugsFaulty
20  IF    StarterStatus = faulty
    THEN  Problem = StarterFaulty
```