Reasoning Systems

Backward chaining and the reasoning system is discussed in the previous class, now let we have the same rules for the vehicles.

Vehicles rule base:-

Bicycle: IF vehicleType=cycle
AND num_wheels=2
AND motor= no
THEN vehicle= Bicycle
Tricycle: IF vechicleType=cycle
AND num_wheels=3
AND motor=no
THEN vehicle=Tricycle
Motorcycle: IF vehicleType=cycle
AND numb-wheels=2
AND motor=yes
THEN vechicle=Motorcycle
SportsCar: IF vehicleType=automobile
AND size=small
AND num_doors=2
THEN vechicle=Sports_Car
Sedan: IF vechicleType=automobile
AND size=medium
AND num_doors=4
THEN vehicle =Sedan
MiniVan: IF vechicleType=automobile
AND size=medium
AND num_doors=3
THEN vechicle=MiniVan
SUV: IF vechicleType=automobile
AND size=large
AND num_doors=4
THEN vehicle=Sports_utility_vechicle
Cycle: IF num_wheels < 4
THEN vechicleType= cycle
Automobile: IF num_wheel=4
AND motor=yes
THEN vechicleType=automobile

Backward Channing

It is often called goal-directed inferencing, because a particular consequence or goal clause is evaluated first, and then we go backward through the rules.

- Unlike forward chaining, which uses rules to produce new information,

- *Backward chaining uses rules* ***to answer questions about whether a goal clause is true or not.***

- *Backward chaining is* ***more focused than forward chaining****, because it only processes rules that are relevant to the question. It is similar to how resolution is used in predicate logic.*

- *Backward chaining is used for advisory expert systems, where users ask questions and get asked leading questions to find an answer.* ***Mycin*** *is an example that used the backward chaining of bacterial infections in medical patients (Shortliffe 1976).*

Backward Algorithm

1.Load the rule base into the inference engine, and any facts from the knowledge base into the working memory.

2. Add any additional initial data into the working memory.

3. Specify a goal variable for the inference engine to find.

4.Find the set of rules which refer ***to the goal variable in a consequent clause****. That is, find all rules which set the value of the goal variable when they fire. Put each rule on the goal stack.*

5. If the goal stack is empty, halt.

6. Take the top rule off the goal stack.

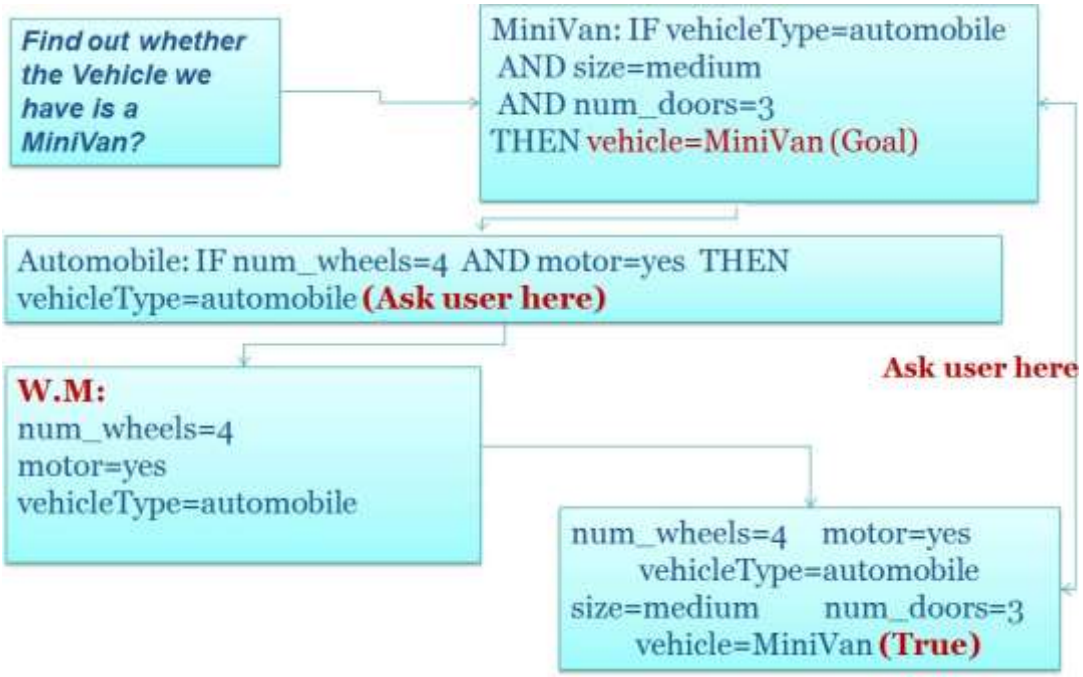
7.Try to prove the rule is true by testing all antecedent clauses to see if they are true. We test each antecedent clause in turn:

(A) If the clause is true, go on to the next antecedent clause.

(B) If the clause is false, then pop the rule off the goal stack; go to step 5.

(C) If the truth value is unknown because the antecedent variable is unknown, go to step 4, with the antecedent variable as the new goal variable.

(D) If all antecedent clauses are true, fire the rule, setting the consequent variable to the consequent value, pop the rule off the goal stack, and go to 5.



Question: which is better: backward or forward chaining?