Rule Based Systems

Adnan Shahzada

Rules for Knowledge Representation

- One way to represent knowledge is by using rules
- Rules express what must happen or what does happen when certain conditions are met
- Example
  - If weather is cold then wear a coat

Rule Base

- A rule is defined as if-then statements

  If

  st1

  st2

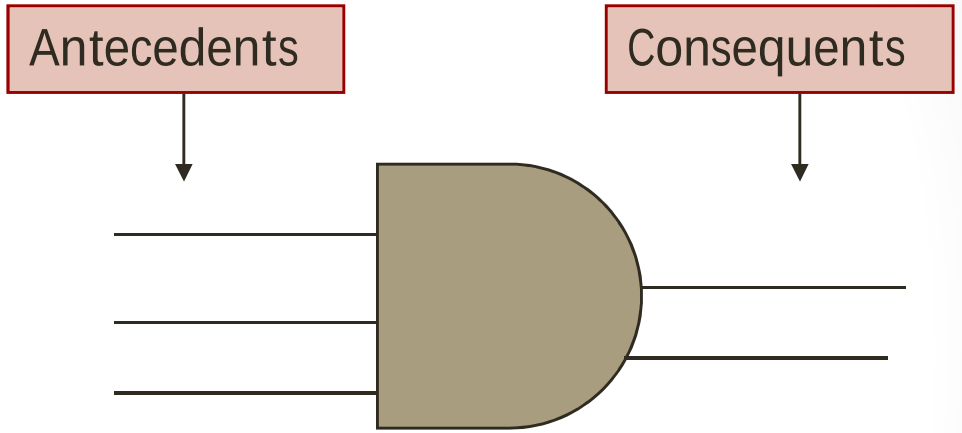
  :

  then

  conc1

  conc2

  :



Rule Based System Architecture

- Rule based system are computer systems that use rules to provide recommendations or diagnoses, or to determine a course of action in a particular situation or to solve a particular problem.

- Components of Rule based system
  - A database of rules
  - A database of facts
  - An interpreter or inference engine

We might want to:

- See what new facts can be *derived*
- *Ask* whether a fact is implied by the knowledge base and already known facts

Rule Based Systems

- A fact or Assertion is something that's true e.g.
  - Weather is cold
- The then pattern often specifies a new assertion to be placed in working memory
- Such a rule-based system is called deduction system.
  - If car color is yellow then it's a taxi
- Sometimes the then pattern specifies an action. Such a rule-based system is called reaction system.
  - If it is hot then switch on the AC

Control Schemes/Reasoning with Rules

- Given a set of rules like these, there are essentially two ways we can use them to generate new knowledge:

- Forward chaining
  - starts with the facts, and sees what rules apply (and hence what should be done) given the facts.
  - data driven reasoning;

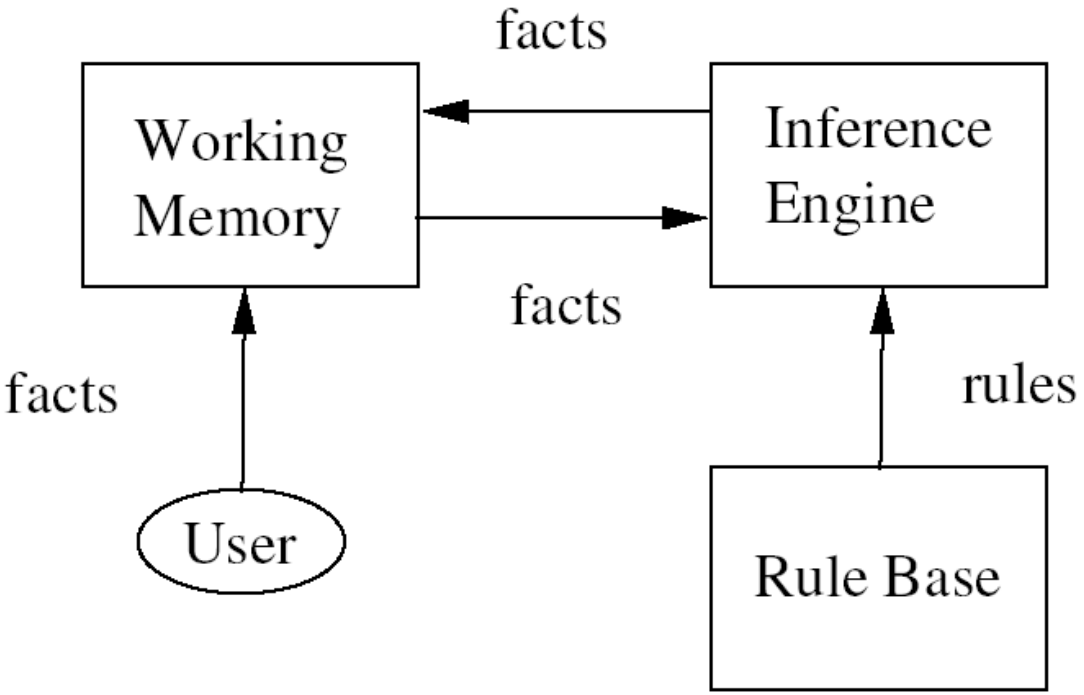
- Backward chaining
  - starts with something to find out, and looks for rules that will help in answering it
  - goal driven.

Forward Chaining

- Take the facts in the fact database and see if any combination of these match all antecedents of a rule
- Rule is triggered if all antecedents of a rule are matched by the facts in the database
- When rule is triggered then its fired
  - Means conclusion is added to the facts database
- In deduction systems generally all triggered rules are fired
- In reactive systems there is a need to decide which possible action is to be taken
- There is a need for conflict resolution

Forward Chaining System

- Facts are held in a *working memory*
- Condition-action rules represent actions to take when specified facts occur in working memory.
- Typically the actions involve adding or deleting facts from working memory.



Simple Example (Forward Chaining)

- R1: IF hot AND smoky THEN ADD fire
- R2: IF alarm_beeps THEN ADD smoky
- R3: If fire THEN ADD switch_on_sprinklers

- F1: alarm_beeps [Given]
- F2: hot [Given]

- F3: smoky [from F1 by R2]
- F4: fire [from F2, F4 by R1]
- F5: switch_on_sprinklers [from F4 by R3]

Properties of Forward Chaining

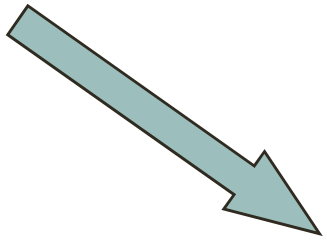
- Note that *all rules which can fire do fire*.
- Can be inefficient
  - leads to spurious rules firing, unfocussed problem
- Set of rules that can fire known as *conflict set*.
- Decision about which rule to fire is *conflict resolution*.

Conflict Resolution

- In some cases all conclusions can be derived
- Rules can be given priority levels
- Longest Matching Strategy

If it is cold
Then wear a coat
If it is cold
Then stay at home
If it is cold
Then turn on heater

If patient has pain
Then prescribe painkillers priority 10
If patient has pain
And pain = chest pain
Then treat for heart disease priority 100



If patient has pain
Then prescribe painkiller

If patient has chest pain
And patient is over 60
And patient has history of heart diseases
Then take to emergency room

Meta Rules

- Knowledge about knowledge is called Meta Knowledge
- Rules that define how conflict resolution will be used and how other aspects of the system itself will run are called meta rules.
- Knowledge engineer building the expert system is responsible for building appropriate meta knowledge into the system
- Meta-knowledge encodes knowledge about how to guide search for solution.
- Explicitly coded in the form of rules

Backward Chaining

- Same rules/facts may be processed differently, using backward chaining interpreter
- Backward chaining means reasoning from *goals* back to *facts*.
- The idea is that this focuses the search.
- Example: Checking *hypothesis*
  - Should I switch the sprinklers on?

Algorithm for Backward Chaining

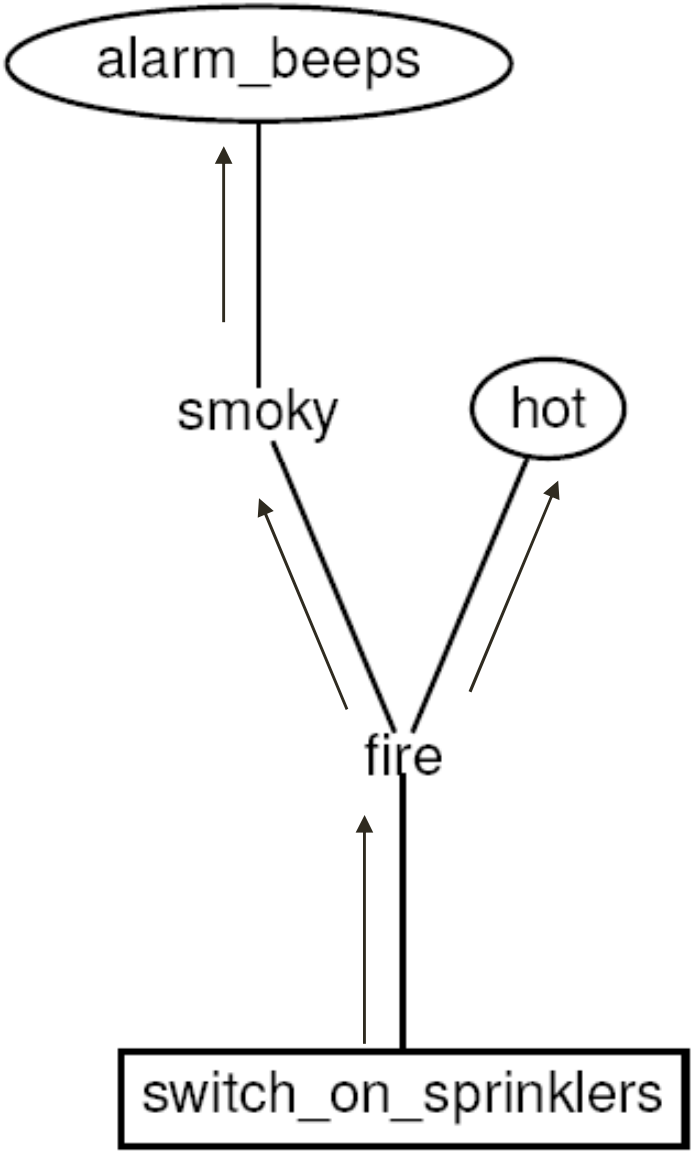
- To prove goal G:
- If G is in the initial facts, it is proven.
- Otherwise, find a rule which can be used to conclude G, and try to prove each of that rule's conditions.

Example of Backward Chaining

- Rules:
- R1: IF hot AND smoky THEN fire
- R2: IF alarm beeps THEN smoky
- R3: If fire THEN switch on sprinklers

- Facts:
- F1: hot
- F2: alarm beeps

- Goal:
- Should I switch sprinklers on?



Another Example

- Goal is Z
  - Rule 1:If *Y & D* then *Z*
  - Rule 2: IF *X & B & E* then *Y*
  - Rule 3: If *A* then *X*
  - Rule 4: If *C* then *L*
  - Rule 5: If *L &M* then *N*

Another Example

- Goal is Z
  - Rule 1:If *Y & D* then *Z*
  - Rule 2: IF *X & B & E* then *Y*
  - Rule 3: If *A* then *X*
  - Rule 4: If *C* then *L*
  - Rule 5: If *L &M* then *N*

Forward Chaining

Cycle 1:First **Rule 3** and then **Rule 4** are fired. New facts that are added to the database: **X** and **L**

Cycle 2: Rule 2 is fired. New facts that are added to the database: Y

Cycle 3: Rule 1 is fired. New facts that are added to the database: Z

Totally 4 rules were used

Backward Chaining

Cycle 1:**Rule 3** is fired. New facts that are added to the database: X

Cycle 2: Rule 2 is fired. New facts that are added to the database: Y

Cycle 3: Rule 1 is fired. New facts that are added to the database: Z

Totally 3 rules were used

Forward Vs Backward Chaining

- Depends on problem, and on properties of rule set.
- If you have clear hypotheses, backward chaining is likely to be better.
  - Goal driven
- Forward chaining may be better if you have less clear hypothesis and want to see what can be concluded from current situation.
  - Data driven

Properties of Rules

- Rules are a natural representation.
- They are inferentially adequate.
- They have representation adequacy for some types of information/environments.
- They can be inferentially inefficient (basically doing unconstrained search)
- They can have a well-defined syntax, but lack a well defined semantics.