Representing Knowledge Using Rules

- Procedural Versus Declarative Knowledge
- Logic Programming
- Forward versus Backward Reasoning
- Matching
- Control Knowledge
Procedural Versus Declarative Knowledge

- A *declarative representation* is one in which knowledge is specified, but the use to which that knowledge is to be put is not given.

- A *procedural representation* is one in which the control information that is necessary to use the knowledge is considered to be embedded in the knowledge itself.
Logic Programming

∀ x: pet(x) ∧ small(x) ->
apartmentpet(x)
∀ x: cat(x) ∨ dog(x) -> pet(x)
∀ x: poodle(x) -> dog(x) ∧ small(x)
poodle(fluffy)

A Representation in Logic

apartmentpet(X) :- pet(X) , small(X) .
pet(X) :- cat(X).
pet(X) :- dog(X).
dog(X) :- poodle(X).
small(X) :- poodle(X).
poodle(fluffy).

A Representation in PROLOG
Forward versus Backward Reasoning

. Are there more possible start states or goal states?

. In which direction is the branching factor greater?

. Will the program be asked to justify its reasoning process to a user?

. What kind of event is going to trigger a problem-solving episode?
Matching

- Indexing
- Matching with Variables
- Complex and Approximate Matching
- Conflict Resolution
  - Preference Based on Rules
  - Preference Based on Objects
  - Preference Based on States
Control Knowledge

1. Knowledge about which states are more preferable to others

2. Knowledge about which rule to apply in a given situation

3. Knowledge about the order in which to pursue subgoals

4. Knowledge about useful sequences of rules to apply
Production System

• Invented in 1943 by Post Used as the basis for many rule-based expert systems Production System consists of 3 components:

• **Rules**  
  An unordered set of user-defined "if-then" rules of the form: \( \text{if } P_1 \land \ldots \land P_m \text{ then Action}_1, \ldots, \text{Action}_n \)  
  where the \( P_i \)s are facts that determine the conditions when this rule is applicable. Each Action adds or deletes a fact from the Working Memory.

• **Working Memory (WM)**  
  A set of “facts” consisting of positive literals defining what's known to be true about the world

• **Inference Engine**  
  Procedure for inferring changes (additions and deletions) to Working Memory.
Inference Engine

while changes are made to Working Memory do:

- **Construct Conflict Set**
  The Conflict Set is the set of all possible \( (\text{rule, list-of-facts}) \) pairs such that rule is one of the rules and list-of-facts is a subset of facts in WM that unify with the antecedent part (i.e., Left-hand side) of the given rule.

- **Apply Conflict Resolution Strategy**
  Select one pair from the Conflict Set.

- **Act Phase**
  Execute the actions associated with the consequent part of the selected rule, after making the substitutions used during unification of the antecedent part with the list-of-facts.
Conflict Resolution Strategy

- Conflict Resolution Strategies The following are some of the commonly used conflict resolution strategies. These are often combined as well to define hybrid strategies.

  - **Refraction**
    A rule can only be used once with the same set of facts in WM. Whenever WM is modified, all rules can again be used. This strategy prevents a single rule and list of facts from be used over and over again, resulting in “infinite firing” of the same thing.

  - **Recency**
    Use rules that match the facts that were added most recently to WM. Hence, each fact in WM has a time-stamp indicating when that fact was added. Provides a kind of “focus of attention” strategy.

  - **Specificity**
    Use the most specific rule, i.e., if one rule’s LHS is a superset of the facts in the LHS of a second rule, then use the first one because it is more specific. In general, select that rule that has the largest number of preconditions.
Example

- Let WM = \{A, D\}
- Let Rules =
  1. if A then Add(B)
  2. if A then Add(C), Delete(A)
  3. if A ^ E then Add(D)
  4. if D then Add(E)
  5. if A ^ D then Add(F)
- Conflict Set = \{\text{Rule1, (A)), (Rule2, (A)), (Rule4, (D)), (Rule5, (A,D))}\}
- Using Specificity Conflict Resolution Strategy, select (Rule5, (A,D)) because it matches two facts from WM while the others match only one fact each.
- “Fire” Rule5 by adding F to WM, so that now WM =\{A, D, F\}