

Problem Solving

Dan Rochowiak
drochowi@cs.uah.edu

Basic Feature Set

- Solve problems using:
 - declarative and semantic knowledge to reason about the world and system state
 - procedural knowledge to act when needed under new conditions
 - episodic knowledge to modify their actions
- Through mechanisms for
 - search
 - knowledge representation
 - inference
 - learning

Problem Solving

- Pragmatic problem solvers
- Problem solving has
 - an input phase in which a problem is perceived and an attempt is made to understand the situation or problem;
 - a processing phase in which alternatives are generated and evaluated and a solution is selected;
 - an output phase which includes planning for and implementing the solution; and
 - a review phase in which the solution is evaluated and modifications are made, if necessary.

Structure of Problems

- Following Kahney H. (Problem Solving: a cognitive approach, Open University Press 1986), a well-structured problem is one in which the problem solver (agent) is provided with all the information needed to solve the problem.

Structure of Problems

- In well-structured problems there are four “types” of information:
 - “initial state” data on the question (the situation as is)
 - “goal state” data (where to go)
 - “legal operators” (steps allowed)
 - “operator restrictions” (constraints on the application of the operators)

Structure of Problems

- An ill-structured problem lacks clear details of the “initial” or “goal” state or or does not include a complete specification of “operators” that can be employed in solving the problem.

Problem solving

- Problem solving generally proceeds by selective (informed) search through large sets of possibilities, using rules of thumb (heuristics) to guide the search.
- Expert problem solving relies on large amounts of domain information.
- Many problems fail to be well-structured.

Simple Case

- Suppose that there is a description of something, and the solution is to find something that it matches. The focus is on the processing phase of problem solving.
- Typically the source and target descriptions are keys to the problem solution.
- The solution might be found in the data of the node stored in a library.

Simple Case

- Descriptions provide the schema.
 - The object description is to be matched to the library description
- The filling instruction is simply to find a match
- The classification defines how the match is to be accomplished.

Describe and Match

1. Create an object description, OD, in some specific and well-defined representation
2. Select the first unexamined library description, LD
3. If LD is empty (n is nil)
then
 report there are no matching items
4. If LD matches OD
then
 stop
 report a match
else
 return to 2

Match Mechanisms

- Establish the collection of matching functions that can be employed in the technique. These functions provide a classification. There are many possibilities
 - use the whole of OD and LD as the target for the match
 - use some features of OD and LD.
 - simply access OD and LD
 - evaluate OD and LD.

Well-defined Representation

- There will need to be a common representation scheme for the source (LD) and the target (OD)
- The representation scheme should cover the basic concepts needed to represent the problems and solutions in the domain
- Representation will be domain specific

Generate and Test

1. Generate a hypothesis, H
2. If H is empty
then
report failure
3. Test H
4. If H is satisfactory
then
report H
else
return to 1

Satisfaction and Generation

- Satisfaction may mean satisficing.
 - Not necessary to find an exact or optimal answer. One that is “good enough” may do.
- Notice that a “good” hypothesis generator should
 - be able to generate all possible solutions,
 - should never propose the same solution twice, and
 - should use the available information to limit hypothesis generation.

Search Space

- In both ‘describe and match’ and ‘generate and test’ there is a space that is being searched.
- The search space may be given in the problem or there may be functions that generate or evolve the search space.
- What might a search space be like?

Semantic Nets

- A semantic net is a representation
- In which
 - Lexical description: there are nodes, links, and domain specific labels
 - Structural description: each link connects a head node to a tail node
 - Semantic description: nodes and links denote domain specific items
- With constructors that
 - Construct nodes
 - Given a link label and two nodes, construct a link
- With readers that
 - Produce a list of all links departing a node
 - Produce a list of all links arriving at a node
 - Given a link, produce a tail node
 - Given a link, produce a head node
 - Given a link, produce a link label

Semantic Tree

- A semantic tree is a semantic net
- In which
 - Some links are branches: branches connect two nodes the parent (head) and the child (tail)
 - One node has no parent: the root node
 - Some nodes have no children: the leaf nodes
- With constructors that
 - Connect a parent node to a child node with a branch
- With readers that
 - Produce a list of the given node's children
 - Produce a given node's parents

Links, Nodes, and Labels

- The early development of semantic nets resulted in criticisms that the semantics of particular nets or trees was not well-structured.
- Nodes, arcs and their labels could be used very freely and ambiguously which led to differing interpretations.
- Good labels promote good problem solving and effective search.
- The labels, by themselves, may not carry semantic knowledge.

Knowledge Representation

- There is a need for well-structured knowledge representation techniques.
- There is a need for a well-structured vocabulary (ontology) for the domain.
- Without these, the problem that the system is attempting to solve may become too ill-structured for solution.

Goals and Problem Solving

- Problem solving systems are goal oriented.
- Suppose that one knows the current state of a process and the desired state of a process.
- The desired state is the goal.
- A state space is a semantic net in which nodes denote states and links denote transitions.
- Means-ends analysis can generate a solution. The basic idea is to reduce from each side to a solution.

Means-ends Analysis

1. Describe current state, goal state, and the difference between them
2. If the current state description satisfies the goal state description
 - then
 - report solution
 - else
 - use the difference to select a promising procedure
3. If there is no procedure selected
 - then
 - report failure
 - else
 - apply the selected procedure
 - make the resulting state the current state
 - return to 1

Goal Tree

- A goal tree is a semantic tree where nodes represent goals and branches indicate how you can achieve goals by solving one or more sub-goals.
- And goals: All sub-goals must be solved
- Or goals: Any sub-goals must be solved

Goal Reduction

- The idea is to identify the subgoals for a problem that would, if satisfied, produce the satisfaction of the ultimate goal.
- Intuitively, this is the sort of reasoning that a person goes through when she does the following: "I want to solve for A, but I don't know the solution. If I know the solution to B and C, then I could solve for A. I don't know the solution for B or C, but if I know the solution to D or E, then I would know the solution to B. Thus, if I can solve for D or E and C, then I can solve for A."

Designing for Goal Reduction

- To perform goal reduction, one will need to be able to do a reduction of conjunctive goals and a reduction of disjunctive goals.
- Thus, if and-reduction and or-reduction can be performed, then goal reduction can be performed.

Global Goal Reduction

```
1 If current goal is satisfied
  then
    report goal satisfied
  else
    if current goal is an And goal
      then
        perform and-reduction
      else
        perform Or reduction
```

And Reduction

```
1. Apply goal reduction to each immediate subgoal
2. If goal reduction finds a subgoal that is not satisfied
  then
    report subgoal not satisfied
  else
    report satisfaction
```

Or Reduction

```
1. Apply goal reduction to the first subgoal, SG,
   in the goal list, GL
2. If goal reduction reports satisfaction of the subgoal
  then
    report satisfaction
  else
    remove SG from GL
    return to 1
3. If GL is empty
  then
    report subgoal not satisfied
```

Explanation

- Goal trees enable explanation (introspection).
- How and why actions have been taken
- How did you (mover) do X? Answer: list And-sub-goals or the Or-sub-goal that was achieved
- Why do you do X? Answer: list X's immediate super-goal

Summary

- At this point it would seem possible to build problem solving, goal-driven program
- There are identifiable procedures for problem solving and goal oriented reasoning that seem well-structured enough to be implemented
- The procedures examined so far require searching a space of possibilities.
