CS-208:Artificial Intelligence
Lectures-04
Search preliminaries

Search Methods

What is a Search Process? Every Search Process can be viewed as a traversal through a directed graph in which each node represents a problem state and each directed edge represents a relationship between the states it connects.

What is the Aim of a Search Process? The search process must find a path through the directed graph starting at an initial state and ending in one or more final state.

Five Important Issues that arise in all searching Techniques

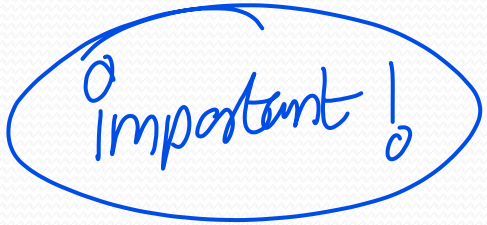
I. The direction in which to conduct the search

II. Topology of the Search Process

III. How each node in the search process will be represented?

IV. Selecting the Applicable Rules.

V. Using a heuristic functions to guide the search.



I. The direction in which to conduct the search

Forward versus backward reasoning: The main objective of a search process is to discover a path through a problem space from an initial state to a goal state.

There are two direction in which a search process could proceed

i. Forward from the start state or
ii. Backward from the goal state

Depending on the topology of the search space, it may be significantly more efficient to conduct the search in one direction rather than the other.

Factors that influence the direction of the search.

a. Are more possible start states or goal state?

It is desirable to move from smaller set of states to a larger set of states.

b. In which direction the branching factor is higher?

It is better to proceed in the direction of lower branching factor.

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

It is important to proceed in the direction that corresponds more closely with the way that user will think.

Bi-directional Search strategy: simultaneous search forward from the start state and backward from the goal state until two paths meet somewhere in-between.

Disadvantage: The bi-directional search may become ineffective when the two searches may pass each other resulting in more work than it would have taken for one of them.

II. Topology of the Search Process

A simple way to implement any search strategy is to perform as a tree traversal:

Problem Tree: Each node of the tree is expanded by the production rules to generate a set of successor nodes and each of which inturn be expanded. This will continue until a node that represents a solution is found.

Problem Graph: A Tree search procedure can be converted to a graph search procedure by modifying the following actions performed at each time when a node is generated:

a) Examine the set of nodes that have been created so far to check if the new successor node is already exists or not.

b) If it does not, simply add the successor to the graph just as in case of a problem tree

c) If it does already exist then do the following two things

i. Set the node that is being expanded to point the already existing node that is same as the new successor. Simply throw away the new successor.

ii. If the tracking of the best path is need to be recorded then check whether new path is better or worse than the old one

- If the new path is worse then do nothing
- If the new path is better then record the new path as correct path to get the node and propagate the corresponding changes down through its successors as necessary. This will create a cycle (a cycle is a path in which a given node appears more than once)

III. How each node in the search process will be represented?

Here three questions that are need to be answered :

1. How can individual objects and facts be represented?
2. How can the representations of objects and facts be combined to form the representation of a complete problem state?
3. How can the sequence of problem state that arise in a search process be represented efficiently?

The first two are the problem of knowledge representation and the third one is particularly important in the context of a search processes.

IV. Selecting the Applicable Rules

A clever search involves choosing the ones that more likely lead to a solution from among the rules that can be applied at a particular point.

Matching: Extracting rules that can be applied from the collection of rules requires some kind matching between the current state and preconditions of the rules.

Indexing: One way to select applicable rules is to search through all the rules and comparing each one's preconditions to the current problem state. This causes two problems:

i. Scanning all the rules at every step of the search process would be hopelessly inefficient. (solution: *Easy way is to use current state an index into the set of rules and select matching ones immediately*)

ii. If there is no ways immediately obvious whether or not a rule's preconditions are satisfied by the particular problem state.

V. Using a heuristic functions to guide the search

A heuristic was defined as a techniques that aids in the discovery of solution to a problem in a most profitable direction even though there is no guarantee that it will never lead in the wrong direction.

There are two major ways in which a domain specific heuristic information can be incorporated in to a rule based search procedure:

- In the rules themselves.
- As a heuristic function that takes a given problem states as an input and determine how promising they are, by providing a value.