Victoria University of Bangladesh
Name: Md. Ziaul Hoque "Sohel"
Student ID: 2221220031
Course Title: Artificial Intelligence
Code: CSI 341
Batch: 22nd (Evening)
Semester: Spring-2023

## Ans to the Que No 1(A)

**Blind search Algorithms:**

Blind search, also called uninformed search, works with no information about the search space, other than to distinguish the goal state from all the others. The following applets demonstrate four different blind search strategies, using a small binary tree whose nodes contain words. Just enter a word in the text input field (your word doesn't have to be in the tree) and click on the "Start Search" button in order to begin the search. The nodes will change color to red as they are visited by the search.

Information about search strategies

- Breadth-First
- Depth-First
- Depth-Limited
- Iterative Deepening
- javadoc-generated documentation for the applet
- directory containing the source for the applet

Breadth-First Search

Breadth-first search goes through the tree level by level, visiting all of the nodes on the top level first, then all the nodes on the second level, and so on. This strategy has the benefit of being complete, and optimal as long as the shallowest solution is the best solution.

Depth-First Search

Depth-first search goes through the tree branch by branch, going all the way down to the leaf nodes at the bottom of the tree before trying the next branch over. This strategy requires much less memory than breadth-first search, since it only needs to store a single path from the root of the tree down to the leaf node.

Depth-Limited Search

Depth-limited search essentially does a depth-first search with a cutoff at a specified depth limit. When the search hits a node at that depth, it stops going down that branch and moves over to the next one. This avoids the potential problem with depth-first search of going down one branch indefinitely.

Iterative Deepening Search

Iterative deepening does repeated depth-limited searches, starting with a limit of zero and incrementing once each time. As a result, it has the space-saving benefits of depth-first search, but is also complete and optimal, since it will visit all the nodes on the same level first before continuing on to the next level in the next round when the depth is incremented.

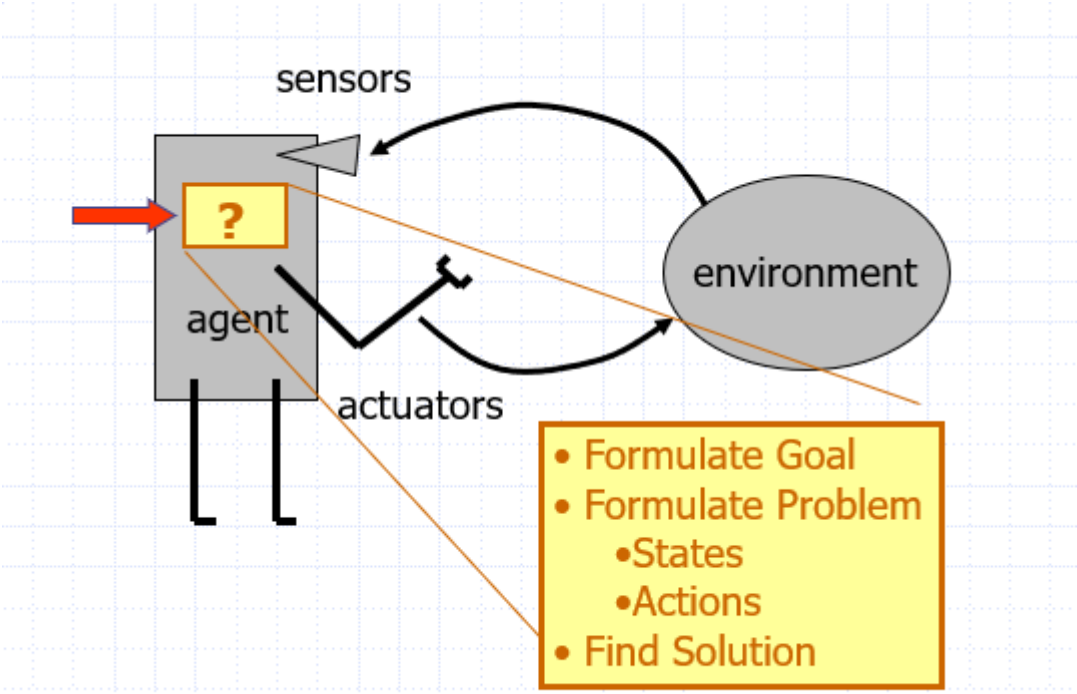
## Ans to the Que No 1(B)

**Four general steps in problem solving:**

- Goal formulation
  - What are the successful world states
- Problem formulation
  - What actions and states to consider given the goal
- Search
  - Determine the possible sequence of actions that lead to the states of known values and then choosing the best sequence.
- Execute
  - Give the solution perform the actions.

## Ans to the Que No 4(A)

**Problem-Solving Agent:**




## Ans to the Que No 4(B)

**Limitation of DFS Strategy:**

- The disadvantage of Depth-First Search is that there is a possibility that it may down the left-most path forever. Even a finite graph can generate an infinite tre One solution to this problem is to impose a cutoff depth on the search. Although ideal cutoff is the solution depth d and this value is rarely known in advance of actually solving the problem. If the chosen cutoff depth is less than d, the algorithm will fail to find a solution, whereas if the cutoff depth is greater than d, a large price is paid in execution time, and the first solution found may not be an optimal one.
- Depth-First Search is not guaranteed to find the solution.
- And there is no guarantee to find a minimal solution, if more than one solution.

## Ans to the Que No 5(A)

**Avoiding Repeated States:**

- Do not return to the parent state.
- Do not create solution paths with cycles.
- Do not generate any repeated states (need to store and check a potentially large number of states)
- This is done by keeping a list of "expanded states" i.e., states whose daughters have already been put on the enqueued list. This entails removing states from the "enqueued list" and placing them on an "expanded list" (In the standard algorithm literature, the list of expanded states is called the "closed list ", thus, we would move states from the open list to the closed list)

## Ans to the Que No 5(B)

**Some Real-World Problems:**

- Route-finding
- Touring
- VLSI layout
- Robot navigation
- Scheduling

## Ans to the Que No 2(A)

**States:**
The states in a robot assembly problem can be represented by the configuration of the robot and the assembly components at any given point in time. Each state describes the position orientation and status of the robot and the assembly component.

**Initial States:**
The Initial state represents the starting configuring of the robot and the assembly component. It specifies the initial position and orientations of the robot and the initial arrangement of the assembly components.

**Action:**
Action s are the operations or movements that the robot can preform in the assembly process. These actions include picking up components. Placing components, moving to different locations, rotating and any other relevant operations required to assemble the components.

**Gola Test:**
The goal test defines the condition or criteria that determine when the assembly process is considered complete or successful. It checks whether the desired configuration of the assembled components has be achieved. The goal test verities if all the components are in their correct position and orientations, meeting the required assembly criteria.

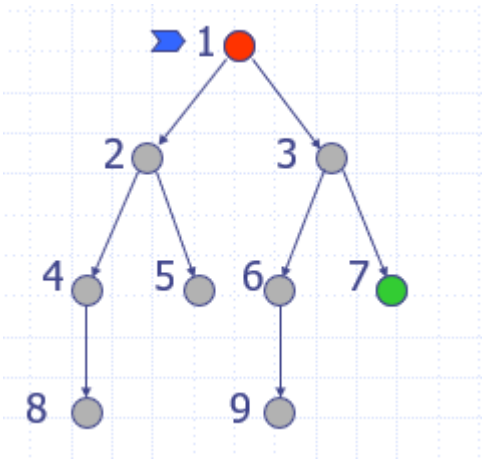
**Path Cost:**
The path cost represents the cost associated with preforming actions or transitions between states in the assembly process. The cost can be measured in term of time, energy, distance, traveled, or any other relevant metric. The path cost determines the efficiency or optimality of the assembly process and can be used to compare differed path or solution.

It's important to note that the specific states, initial state, actions, goal test and path cost in a robot assembly problem would depend on the specific requirements and construction of the problem. The above description provides a general framework for understating the components involved in a robot assembly problem, but the exact details would need to be defined based on the specific context and requirements of the problem at hand.

## Ans to the Que No 2(B)

BFS search strategy:

- ⇨ Expand shallowest unexpended mode
- ⇨ Fringe is a FIFO queue
- ⇨ New nods are inserted at the end of the queue




Steps 1 = 1

Steps 2 = 2 , 3

Steps 3 = 3, 4, 5

Steps 4 = 4, 5, 6, 7

Steps 5 = 5, 6, 7, 8

Steps 6 = 6, 7, 8

Steps 7 = 7, 8, 9

Steps 8 =8, 9

----------END----------