Victoria University
of Bangladesh

Final Assessment

Md Bakhtiar Chowdhury

ID: 2121210061

Department: CSE

Semester: Spring -2023

Batch: 21th

Course Title: Artificial Intelligence

Course Code: CSI 341

Submitted To:

Md. Shahin Khan

Lecturer, Dept. of CSE/CSIT

Victoria University of Bangladesh

Submission Date: 07 June, 2023

Answer to the question NO 1(a)

a) Answer:- Blind search algorithms, also known as uniformed search algorithms, are a class of search algorithm that do not use any domain-specific knowledge or heuristic information about the problem being solved. these algorithms explore the search space systematically without considering any additional-information. Here are some commonly used blind search algorithms and their properties.

1. Breadth-First search (BFS):-

⇒ Property:- Expands all nodes at the current depth level before moving to the next depth level.

⇒ completeness: BFS is complete if the search space is finite and their are no infinite paths.

-> Optimality:- BFS guarantees the optimal solution if the path cost is non-decreasing.

2. Depth-First search (DFS):-

-> property:- Explores as far as possible along each branch before backtracking

z

⇒ completeness : DFS is not complete if the search space contains infinite loops or path.

⇒ optimality : DFS does not guarantee the optimal solution as it may find a solution along a deeper path before finding a shallower one.

3. Iterative Deepening Depth-First search (IDDFS)

⇒ Property :- Repeatedly performs depth-limited searches with increasing depth limits.

4. uniform cost search (UCS)

⇒ Property :- Expands the node with lowest path cost first.

5. Depth-Limited search (DLS)

⇒ Property :- performs DFS up to a certain depth limit.

6). Bidirectional search :-

⇒ property :- Explores the search space simultaneously from both the start and goal states.

Its important to note that blind search algorithms may not always be the most efficient or effective for solving complex problem

Answer to the question NO 1(b)

b. Answer:- The four general steps of problem-solving provide a structured approach to tackle and resolve problems effectively. these steps are commonly followed in problem -solving processes across various domains. Here is a brief description of each step:-

1. understand the problem:- The first step is to gain a clear understanding of the problem at hand. the involves identifying and defining the problem, analysing its components, and determining the desired outcome or goal. it is crucial to gather all the necessary information clarify any uncertainties. and establish and context and constraints. of the problem.

2. Devise a plan:- once the problem is understood the next step is to develop a plan or strategy to solve it. this involves considering different approaches and evaluating their feasibility it may be helpfull to breakdown the problem into smaller sub problems or tasks and determine the appropriate method or techniques to address each component. the plan should

outline the sequence of steps to be taken and the resources required for their execution.

3. Implement the plan:- in this step, the devised plan is put into action. It involves executing the steps and actions outlined in the plan to work towards the solution. This may include performing calculations, conducting experiments, gathering data or applying specific algorithms or techniques. It is essential to follow the plan systematically, track progress and make adjustments if necessary.

4. Evaluate the results:- once the plan is implemented, the next step is to assess and evaluate the obtained results. The involves comparing the achieved outcome with the desired goal and determining the effectiveness and efficiency of the solution. It may be necessary to analyze the result, review the process followed and identify and potential areas for improvement or further refinement. This steps provides valuable insights and feedback that can be utilized in future problem-solving endeavors.

Answer to the question NO 2(a)

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 state:- the initial state represents the starting configuring of the robot and the assembly components. it specifies the initial positions and orientations of the robot and the inital arrangements of the assembly components.

Actions:- Actions are the operations or movements that the robot can perform in the assembly process. these actions include picking up components, placing components, moving to different locations, rotating, and any other relavant operations required to assemble the components.

~~Goal Set~~
goal test :- The goal test defines the condition or criteria that determine when the assembly process is considered complete or successfull. it checks whether the desired configuration

of the assembled components has be achieved. the goal test verifies if all the components are in their correct positions and orientations, meeting the required assembly criteria.

Path cost :- The path cost represents the cost associated with performing 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 different paths or solutions.

its 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 constraints of the problem. the above description provides a general framework for understanding the components involved in a robot assembly problem, but the exact details would need to be defined based on the specifi context and requirements of the problem at hand.

Answer to the question no 2(b)

b) Answer:- BFS search strategy :-

⇒ Expand shallowest unexpanded node

⇒ Fringe is a FIFO queue

⇒ New nodes 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

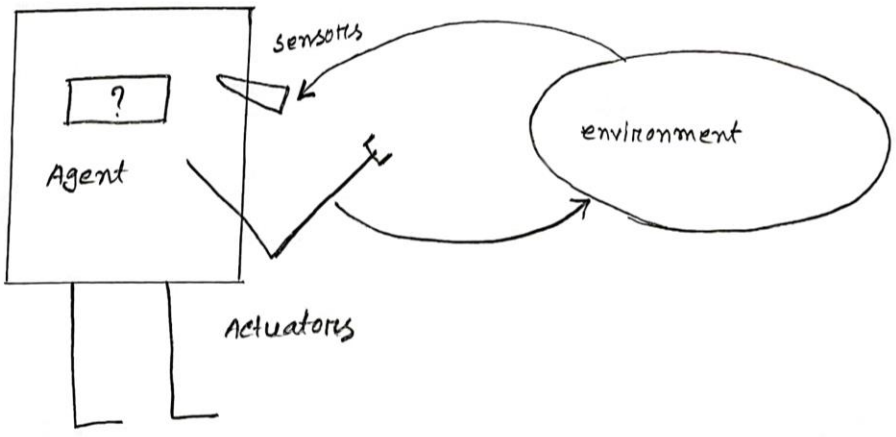
steps 7 = 7, 8, 9

steps 8 = 8, 9

Answer to the question NO 4 (a)

a) Answer:- A problem-solving agent is an intelligent agent that is capable of perceiving and analyzing a given problem or situation formulating a suitable plan or strategy, and taking actions to achieve the desigeed goal. It follows a systematic approach to identify problem, explore potential solutions, and a simple diagram illustrating a problem-solving agent.

A problem solving agent is an intelligent entity that observes it environme identifies problems or goals, generates solutions, and takes actions to transform the current state to a desired state.



Explanation of the diagram:-

Perception:- the agent perceives the environment and gathers information of data about the current state of the problem.

Problem formulation:- Based on the perceived information, the agent formulates the problem by defining the initial state, goal state, available and constraints.

Planning:- The agent generates a plan or strategy to solve the problem. It analyzes the problem, explores different paths or solutions, and determines the sequence of actions to be taken.

Actions Execution:- The agent executes the planned action by interacting with the environment. It performs the necessary operations or manipulations to transform the current state towards the desired goal state.

The problem solving agent continuously observes the environment, reevaluates the problem, and adjust its actions accordingly. It utilizes it knowledge, reasoning and decision-making capabilities to find the most suitable solutions and adapt to changes in the problem or environment.

Answer to the question No 4 (b)

b) Answer:- Depth-First search (DFS) is a popular search algorithm that explores a graph or a tree by traversing as far as possible along each branch before backtracking. while DFS has its advantages, it also comes with certain limitations. Here are some limitations of the DFS strategy.

1. completeness: DFS may not find a solution if the search space contains infinite loops or path. If there is no solution within the search depth limit or if the search space is infinite DFS will run indefinitely without finding a solution.

2. optimality:- DFS does not guarantee finding the optimal solution. It may find a solution along a deeper path before exploring shallower paths that could potentially lead to a better or optimal solution. DFS prioritizes depth over breadth which may reselt in suboptimal solution.

3. memory usage:- DFS can consume a significant amount of memory when exploring deep paths or traversing large graphs. It stores the entire path from the root to the current node, which can be memory-intensive. in cases where memory

resources are limited. DFS may encounter memory constrains or even exhaust are available memory.

4. time complexity:- in certain scenarios, DFS can have a higher time complexity compared to other search algorithms. If the search space is extensive and the branching factor is high, DFS may spend a significant amount of time exploring unnecessary paths and backtracking.

5. Lack of guidance:- DFS is an uninformed search algorithm, meaning it does not use any additional information or heuristics to guide its search. It explores paths solely based on the order in which they are encountred, without considering and knowledge about the problem or the potential quality of the path.

6. Path Length Bias:- DFS tends to favor longer paths over shorter ones, as it explores deeper levels first. this bias towards longer paths may not be desirable in certain situations where shorter paths are preferred.

To overcome some of these limitations, alternatives search algorithm such as Breadth-First search.

Answer to the question NO.5(a)

a) Answer:- To Avoide repeated states in a search algorithm, you can implement a mechanism known as state checking or state tracking. This mechanism ensures that previously visited state are not revisited during the search process. Here's a solution for avoiding repeted states:-

1. Maintain a data structure:- use a data structure, such as a set or a hash table, to store the visited states encountered during the search. the data structure will allow efficient lookup and checking for repeated states.

2. Track visited states:- whenever you generate a new state during the search process, check if it has been visited before. you can do this by comparing the current state with the set of visited state store in the data structure.

3. Avoid Revisiting:- if the generated state has already been visited, skip further exploration of that state and move on the next state. this ensure that you avoid redundant and repeated computations.

4. update visited states:- if the generated state is encountered for the first time, Add it to the set of visited state in the

the data structure. the keeps track of the visited states and prevents revisiting them in subsequent iterations of the search algorithm.

by implementing the solution, you ensure that the search algorithm does not get stuck in loops or revisit states that have already been explored.

Answer to the question no 5(b)

b) Answer:- search algorithms have a wide range of real world applications. Here are some example of problem where searching algorithms can be employed:-

1. web search:- search engines like google, bing, and yahoo use sophisticated search algorithms to retrieve relevant web pages based on user queries. these algorithms analyze the content, relevance, and popularity of web pages to provide accurate research results.

2. Route planning:- navigation systems and map applications utilize searching algorithms to find the optimal routes between two locations. algorithms like Dijkstra's algorithm and A* search are commonly used to determine the shortest or fastest

path based on various factors such as distance, time traffic conditions and road networks.

3. Image and object recognition :- searching algorithms play a crucial role in image and object recognination systems. these algorithms search through large databases of images or patterns to identify and matchs specific objects facts, or features within images. techniques like features extraction template matching, and simplarity search are commonly used

4. Database searching :- search algorithms are used in database to efficiently locate specific records or information. indexing - methods such as B-trees or hash table enable first searching and retrive of data based on specific criteria, such as key value of attributes.

5. Text mining and information Retrival:- searching algorithm are employed in text mining and information retrival system to extract relavant information from the large collection of text data.

6. Genetic and protein sequence analysis:- In bioinformatics, searcing algorithms are used to compare genetic or protein sequences to identify patterns, similarities, and functional relationships

>>>>END<<<<