



Batch: B2
Roll No.: 16014022050

Experiment No. 2

Title: To perform VLAB on Artificial Intelligence

Course Outcome:

CO1: Define and explain the fundamental concepts, scope, and types of Artificial Intelligence and Machine learning.

Vlab Links: <https://ai1-iiith.vlabs.ac.in>List%20of%20experiments.html>

Books/ Journals/ Websites referred:

1. http://en.wikipedia.org/wiki/List_of_artificial_intelligence_projects
2. http://www.cs.cornell.edu/courses/cs478/2002sp/mllinks/interesting_ai_demos_and_project.htm
3. <http://homepages.inf.ed.ac.uk/rbf/AIMOVIES/AImovai.htm>
4. “Artificial Intelligence: a Modern Approach” by Russell and Norving, Pearson education Publications
5. “Artificial Intelligence” By Rich and knight, Tata McGraw Hill Publications

Pre Lab/ Prior Concepts:

History and evolution of AI, Artificial intelligence: definitions and theories.

Historical Profile:

AI research is highly technical and specialized and is also divided by several multidisciplinary technical issues. So far there are many projects those have been developed and are in progress to work on those issues. Students must learn the applications of intelligent robots by studying various such projects to know the depth and complexity of the course.

Introduction:

Vlab On

- 1. AI Depth First Search**
- 2. Greedy Best First Search**
- 3. Min Max Search**

Implementation:

1. AI DEPTH FIRST SEARCH:

Choose difficulty: Beginner Intermediate

What kind of data structure does DFS use?
 a. FIFO Queue
 b. LIFO Queue (Stack)
 c. Linked List
 d. Heap

Does DFS find the shortest path to the goal node?
 a. Yes
 b. No
 c. Sometimes [Explanation](#)
 d. Never

What advantage does DFS have over BFS (Breadth First Search)?
 a. Optimality
 b. Completeness
 c. Less Memory Usage (Space Complexity)
 d. Less Time Complexity

Does DFS keep in memory all of the visited nodes?
 a. Yes
 b. No [Explanation](#)

When is DFS not complete?
 a. When the goal node is not reachable from the source vertex
 b. When the state space is infinite
 c. Either a or b
 d. It is always complete

[Submit Quiz](#)
Score: 5 out of 5

Tasks

Depth First Search

Instructions

Controls

Start Vertex: 0
Start DFS
Next
AutoPlay
Toggle Forces
Clear
Clear Visited
Note: Overrides Graph Tree Controls:
Tree Depth: 4

Information

Frontier: []
Visiting: undefined
To Visit: undefined



The screenshot shows a quiz interface for Depth-First Search (DFS). On the left, a sidebar lists navigation options: Theory, Procedure, Pretest, Demo, Practice, Posttest, References, Contributors, and Feedback. The main content area displays several questions with multiple-choice answers and explanations. At the bottom, there is a 'Submit Quiz' button and a score indicator 'Score: 4 out of 4'. The top right features a 'Rate Me' button and a 'Report a Bug' button.

What happens when visited nodes are not kept track of in DFS?

- a. The DFS algorithm will still find the optimal solution.
- b. The DFS algorithm may get stuck in an infinite loop. Explanation
- c. The DFS algorithm will terminate but may not find the optimal solution.
- d. The DFS algorithm will be faster but may miss some nodes.
- e. The DFS algorithm will not work at all and fail to find any solution.

Assuming that nodes at the same depth as the goal node have no successors, what is the space complexity of DFS? (Here b is the branching factor and d is the depth of the goal node)

- a. $O(b^d)$ Explanation
- b. $O(bd)$
- c. $O(b \cdot d)$
- d. $O(b \log bd)$

What if a FIFO queue is used instead of a LIFO queue (stack) in DFS?

- a. The algorithm becomes more memory efficient.
- b. The algorithm would become equivalent to BFS (Breadth First Search). Explanation
- c. The algorithm would not work properly, i.e. it will not traverse the graph properly and/or completely
- d. No change in the algorithm, i.e. it remains unaffected

Iterative Deepening Depth First Search (IDDFS) is a variation of DFS in which search is performed using a series of depth-limited DFS iterations. It starts with a depth limit of 0 and performs a DFS with that limit. If the solution is not found, it incrementally increases the depth limit by 1 and performs another DFS and so on. Which of the following is true about IDDFS?

- a. Not complete, not optimal, $O(bd)$ space complexity, $O(b^d)$ time complexity
- b. Complete, not optimal, $O(b^d)$ space complexity, $O(b^d)$ time complexity
- c. Not complete, optimal, $O(bd)$ space complexity, $O(bd)$ time complexity
- d. Complete, optimal, $O(bd)$ space complexity, $O(bd)$ time complexity Explanation

2. GREEDY BEST FIRST SEARCH

The screenshot shows a simulation of Greedy Best First Search (GBFS) on a graph. The graph has seven nodes labeled 0 through 6. Node 0 is the start vertex, indicated by a black circle. Nodes 1, 2, 5, and 6 are orange circles, while nodes 3 and 4 are green circles. Edge weights are: 0-1 (135), 0-2 (295), 1-3 (221), 1-4 (224), 2-5 (273), and 2-6 (201). The 'Controls' panel on the left includes fields for 'Start Vertex' (0) and 'Goal Vertex' (5), and buttons for 'Start BEFS' and 'Next'. It also has checkboxes for 'AutoPlay' and 'Toggle Forces', and buttons for 'Clear' and 'Clear Visited'. A note says 'Note: Overwrites Graph Tree Controls:' followed by a 'Tree Depth' field set to 3. The 'Information' panel on the right shows: Goal Vector: 5, Frontier: [6, 1], Estimated Cost(E): [219, 352], Visiting: undefined, To Visit: 6. The 'Observation' panel at the bottom right shows 'Found Goal!'.



ai1-iiith.vlabs.ac.in/exp/greedy-best-first-search/practice.html

Virtual Labs

Greedy Best First Search

★★★★★

Rate Me Report a Bug

Controls

Start Vertex:
 Goal Vertex:

AutoPlay

Note: Overwrites Graph

Tree Controls:
| Tree Depth:

Information

- Goal Vector: 6
- Frontier: [5, 1]
- Estimated Cost(E): [219, 562]
- Visiting: undefined
- To Visit: 5

Observation

- Found Goal!

Computer Science and Engineering > Artificial Intelligence I > Experiments

Aim Theory Procedure Pretest Demo Practice Posttest References Contributors Feedback

Choose difficulty: Beginner Intermediate

Greedy Best First Search

What does the term 'best' refer to in Best First Search (BFS)?
 a: The best path found so far in the search space.
 b: The best heuristic function used for evaluation.
 c: The best node to expand based on the evaluation function.
 d: The best order of exploring nodes in the search space.

Which factor influences the efficiency of Best First Search (BFS)?
 a: The choice of the heuristic function
 b: The size of the search space
 c: The ordering of expanded nodes
 d: The initial state of the problem

What is the time complexity of Best First Search (BFS) in the worst-case scenario?
 a: $O(m^n)$
 b: $O(2^n m)$
 c: $O(b^n m)$
 d: $O(bm)$

Which of the following heuristic functions is admissible in Best-First Search?
 a: Overestimating the cost from the current node to the goal node
 b: Underestimating the cost from the current node to the goal node Explanation
 c: Always assigning a heuristic value of zero to all nodes
 d: Assigning a random heuristic value to each node

Which property does Best First Search (BFS) guarantee to find?
 a: Completeness
 b: Optimality
 c: Admissibility
 d: None of the above Explanation

Score: 5 out of 5

Virtual Labs
An ISO 9001:2015 Certified Institute

Greedy Best-First Search

★★★★★ Rate Me Report a Bug

Theory Procedure Pretest Demo Practice Posttest References Contributors Feedback

Choose difficulty: Beginner Intermediate

Consider a Best-First Search algorithm applied to a search space with a branching factor of 3. If the maximum depth of the search space is 5, how many nodes will be expanded in the worst case scenario?

a: 15
 b: 243
 c: 45
 d: 121 Explanation

If h_1 and h_2 are two admissible heuristics for a given problem, then heuristic $h_3(n) = \max(h_1(n), h_2(n))$ for all states, n , must also be admissible.

a: True Explanation
 b: False

Consider the problem of getting from Iasi to Fagaras in the simplified map of Romania (Figure 1). Which of the following will occur using BFS?

a: An optimal path will be found from Iasi to Fagaras.
 b: A non-optimal path will be found from Iasi to Fagaras.
 c: Fagaras will never be reached. Explanation
 d: None of the above

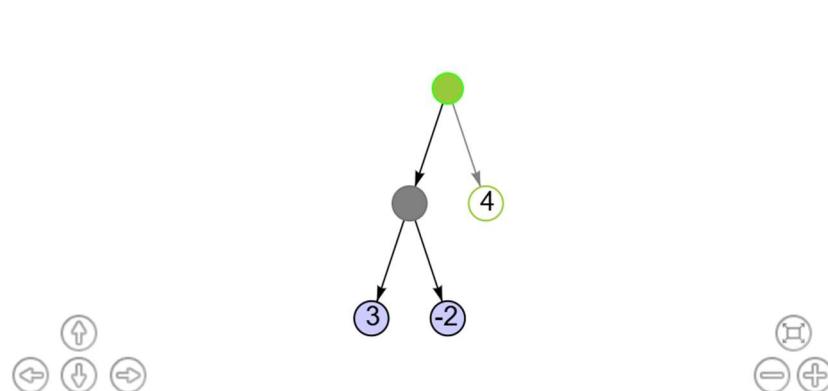
In a Best-First Search algorithm, which of the following evaluation functions is not suitable for solving a maze navigation problem?

a: Number of steps taken from the start to the current node
 b: Manhattan distance from the current node to the goal node
 c: Euclidean distance from the current node to the goal node
 d: Randomly generated value assigned to each node Explanation

Submit Quiz

Score: 4 out of 4

3. MIN MAX SEARCH

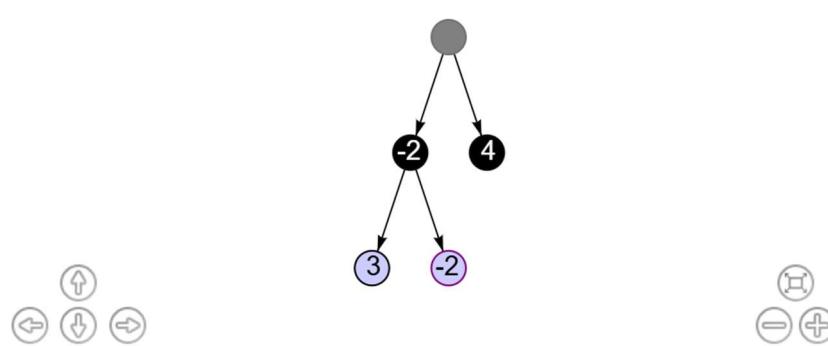


Observations

Select the correct node

Settings

First Player : Maximizer
Graph Number : 3
Random Graph
Random Values

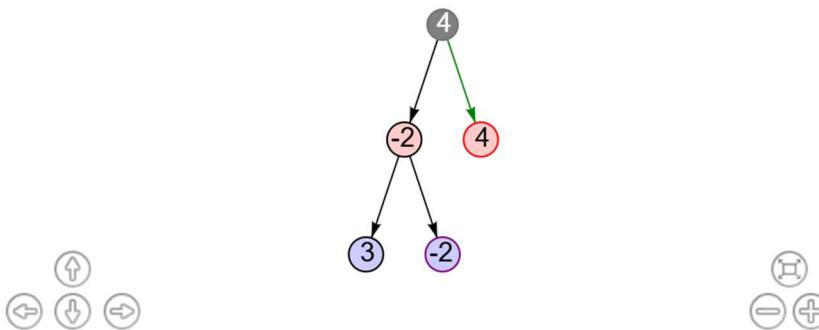


Observations

Since it was minimizer's turn -2, which is min of child node values is picked

Settings

First Player : Maximizer
Graph Number : 3
Random Graph
Random Values



Observations
Green arrows highlight the decision taken in each turn

Settings
First Player : Maximizer
Graph Number : 3
Random Graph
Random Values

Minimax Search

Choose difficulty: Beginner Intermediate Advanced

Can there be more than 1 final path in a graph using minimax?
 a. Yes [Explanation](#)
 b. No

To solve minimax algorithm on a graph what kind of traversal do we use?
 a. Depth First Traversal [Explanation](#)
 b. Breadth First Traversal
 c. None of these

Where can we use minimax algorithm?
 a. Chess
 b. Go
 c. Tic Tac Toe
 d. All of these [Explanation](#)

What is the primary objective of the minimax search algorithm in game theory?
 a. To maximize the score for the maximizing player [Explanation](#)
 b. To minimize the score for the minimizing player [Explanation](#)
 c. To maximize the score difference between the maximizing player and the minimizing player [Explanation](#)
 d. To minimize the number of nodes explored during the search. [Explanation](#)

Which of the following best describes the concept of alpha-beta pruning in the minimax search algorithm?
 a. A technique used to improve the efficiency of the minimax algorithm by discarding irrelevant subtrees. [Explanation](#)
 b. A strategy employed by the maximizing player to maximize their score in a game.
 c. A heuristic function that evaluates the quality of a move in the minimax algorithm.
 d. A method to randomly select moves in order to explore different paths in the game tree.

Submit Quiz
Score 5 out of 5

In Tic-Tac-Toe, if we don't use depth as parameter while mapping to graph, then which of the following will be true?

- a. There will be only 1 path [Explanation](#)
- b. We will terminate quickly as compared to when we use depth [Explanation](#)
- c. Both a and b
- d. None of these

Alpha beta pruning is used for?

- a. Reducing number of nodes that are considered
- b. Optimize minimax algorithm
- c. Mostly useful when large number of states such as in chess
- d. All of these

What is the main disadvantage of the Minimax algorithm?

- a. It is computationally intensive for large game trees [Explanation](#)
- b. It does not guarantee finding the optimal solution
- c. It is difficult to implement
- d. It only works for zero-sum games [Explanation](#)

What is the main advantage of using a Minimax tree in game theory?

- a. It guarantees finding the optimal solution [Explanation](#)
- b. It is computationally efficient for large game trees
- c. It is easier to implement than other algorithms
- d. It works well for non-zero-sum games

What is the time complexity of the Minimax algorithm for a complete tree where n is the number of moves?

- a. O(n)
- b. O(log n)
- c. O(n²)
- d. O(2ⁿ) [Explanation](#)

Feedback:



The screenshot shows a survey form titled "VLabs Outreach" from "Virtual Lab Outreach portal IIT Bombay An initiative by Ministry of Education". The form includes questions about the virtual lab experience and a success message at the bottom.

Questions:

- a. The degree to which the actual lab environment is simulated
- b. The manuals were to be found helpful
- c. The results of experiment were easily interpretable
- d. What was your experience while performing the experiment in virtual lab?
- e. Do you think performing experiments through Virtual Labs gives scope for more innovative and creative research work
- f. Did you go through the manual / step by step method before performing the live experiments

Rating Scale:

Excellent	Very Good	Good	Fair	Poor
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Excellent	Very Good	Good	Fair	Poor
<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Excellent	Very Good	Good	Fair	Poor
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
Yes	Maybe	To some extent	Don't know	No
<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
Yes	No			
<input checked="" type="radio"/>	<input type="radio"/>			

Buttons:

- Submit (purple button)
- Success (green button)

Success Message:

Feedback submitted successfully ×

Post Lab Descriptive Questions:

1. How can Depth First Search be applied to navigate a maze with multiple dead ends to find a possible escape path?

Depth First Search (DFS) is a fundamental graph traversal algorithm that can be applied effectively to navigate a maze with multiple dead ends. The algorithm explores as far as possible along each branch of the maze before backtracking. In the context of a maze, where the **nodes represent junctions or intersections**, and the **edges represent possible paths**, DFS will systematically explore one path to its deepest point before retreating and trying another direction if a dead end is encountered.

To apply DFS in navigating a maze, we typically **begin at the maze's entrance**, marking it as visited. Then, the algorithm **moves forward along an unvisited path until it reaches a dead end** or a junction where no further unvisited paths exist. **Once a dead end is reached, the algorithm backtracks to the last junction and explores an alternative path**, repeating the process until it finds an exit or exhausts all possible paths. If the maze contains multiple dead ends, DFS is particularly suited because it does not require knowledge of the maze's layout in advance. It is capable of exploring the entire structure and returning to previously explored areas. This exhaustive search ensures that if there is an escape route, DFS will eventually find it.

However, a potential downside of DFS is that it can be inefficient in mazes with large branching factors or dead ends scattered throughout, since it might explore long and non-optimal paths before backtracking. It can also get stuck in loops or revisit already explored areas if not properly managed, which is why modern implementations often use a stack to keep track of visited nodes or rely on marking cells to prevent revisiting.

2. In a GPS navigation system, how does Greedy Best First Search help in choosing the next city to visit based on straight-line distance to the destination?

Greedy Best First Search (GBFS) is an informed search algorithm used to find an optimal path based on a **heuristic that estimates the cost to reach the goal**. In the context of GPS navigation systems, GBFS **can help in choosing the next city or node to visit by evaluating the straight-line distance** (also known as the "heuristic") from the current city to the destination city. The primary goal of GBFS is to make decisions that seem the best in the short term, based solely on the heuristic, with the hope that this leads to a globally optimal path.

When a GPS system uses GBFS, **it starts at the current location and computes the straight-line distance from all neighboring cities to the destination**. The city with the smallest heuristic value (i.e., the one that appears to be closest to the goal) is selected as the next city to visit.

For instance, in a road network, **the GPS system would evaluate neighboring cities based on the straight-line distance to the destination, potentially choosing a city that seems close to the goal even if the actual roads make the path longer**. This approach works well when the heuristic is reliable, such as in cases where cities are spread out in a relatively predictable manner (like in a grid or planar network). However, if the heuristic is misleading (for example, due to road detours or traffic), GBFS can lead to suboptimal routes. While this makes GBFS an efficient algorithm for quick decisions, it may not always yield the most cost-effective path when compared to algorithms that consider both the distance and the cost to travel.

3. How is Min-Max Search used in a chess game to choose the best move while anticipating the opponent's counter moves?

Min-Max Search is a decision-making algorithm used extensively in two-player, zero-sum games like chess, where one player's gain is another player's loss. **The algorithm aims to choose the best possible move by simulating all possible future moves**, both for the player and the opponent, and **selecting the one that maximizes the player's chances of winning while minimizing the opponent's opportunities**.

In the context of a chess game, Min-Max works by constructing a game tree where **each node represents a state of the chessboard, and each edge represents a possible move**. Starting from the current game state (the root of the tree), the algorithm **recursively simulates all potential moves for both the player and the opponent**. At the terminal nodes (the leaves of the tree), the algorithm assigns a value that indicates the game's outcome (for example, +1 for a win, 0 for a draw, and -1 for a loss).

The "Min" part of the algorithm refers to the opponent's optimal strategy to minimize the player's advantage, while the "Max" part refers to the player's strategy to maximize their chances of winning. The algorithm evaluates the game tree by alternating between "Max" and "Min" levels, going from the leaves to the root. At each level, the player picks the

maximum value (best move), while the opponent picks the minimum value (to minimize the player's gain).

To choose the best move, Min-Max evaluates all possible moves and counter-moves, assuming that the opponent will always play optimally to counter the player's choices. For example, if it is the player's turn to move, Min-Max will evaluate all the moves that could be made, and for each of those moves, it will evaluate the subsequent moves the opponent might make. This evaluation continues recursively for all future moves and counter-moves, considering the depth of the game tree and the number of possible scenarios.

While Min-Max provides a robust strategy, it can be computationally expensive, as it may require evaluating a vast number of possible moves, especially in complex games like chess. To improve efficiency, techniques like Alpha-Beta Pruning are often used to eliminate branches of the game tree that don't need to be explored, significantly reducing the computational overhead.

Thus, Min-Max Search is crucial for making decisions in a chess game, as it takes into account both the current move and all subsequent moves, helping to anticipate the opponent's counter-moves and ensuring that the player selects the optimal strategy.