SFBU **Name:** ______________________

Introduction to Artificial Intelligence

Midterm Sample

10/14/2024

Time Limit: 75 Minutes ______________________

1. This exam contains 8 pages (including this cover page) and 28 questions.
2. Total of points is 100.
3. Answer all questions. The marks for each question are indicated at the beginning of each question.
4. This **IS a CLOSED BOOK** exam.
5. Calculators are not allowed.

1 True/False (20 points)

1. (2 points) A* search is guaranteed to find an optimal solution if the heuristic is admissible. ___
2. (2 points) Breadth-First Search (BFS) guarantees finding the shortest path in all types of graphs. ___
3. (2 points) Depth-First Search (DFS) may get stuck in an infinite loop if the graph contains cycles and no visited nodes list is maintained. ___
4. (2 points) The Hill Climbing algorithm always finds the global optimum solution.___
5. (2 points) Simulated Annealing allows occasional uphill moves to escape local optima. ___
6. (2 points) Constraint Satisfaction Problems (CSPs) are best solved using informed search algorithms like A*. ___
7. (2 points) In Iterative Deepening Search, the depth limit is gradually increased until a solution is found. ___
8. (2 points) In A*, the function $f(n) = g(n) + h(n)$, where $g(n)$ represents the cost from the start node to n and $h(n)$ is the estimated cost to the goal.___
9. (2 points) Hill Climbing with random restart can help avoid local optima.___
10. (2 points) The time complexity of BFS is $O(b^d)$, where b is the branching factor and d is the depth of the shallowest solution.___

2 Multiple Choice (20 points, 2pts each)

Select **all** choices that apply.

11. (2 points) **Pathfinding in a Grid**
Scenario: A robot must find the shortest path from the top-left corner to the bottom-right corner of a grid. Some grid cells have obstacles. The robot uses A* with a Manhattan distance heuristic.
What happens if the heuristic underestimates the true cost of reaching the goal?
    A. The robot will fail to find any path.
    B. The robot will find a suboptimal path.
    C. The search will be incomplete.
    D. The robot will find the optimal path.

12. (2 points) **Mountain Climbing Robot**
Scenario: A robot is tasked with finding the highest peak on a terrain using the Hill Climbing algorithm. However, there are multiple peaks of varying heights.
What is the main risk for the robot in using this algorithm?
    A. The robot will always find the global peak.
    B. The robot will take too long to reach a peak.
    C. The robot may get stuck at a local peak.
    D. The robot will explore every possible path.

13. (2 points) **Maze Solving Using BFS**
Scenario: A student writes a BFS algorithm to solve a maze where every path has an equal cost. The goal is to find the shortest path from start to goal.
Which of the following is true about BFS in this case?
    A. BFS guarantees finding the shortest path.
    B. BFS explores the entire maze before returning a solution.
    C. BFS may not find the shortest path if there are cycles.
    D. BFS will run faster than DFS for all mazes.

14. (2 points) **Simulated Annealing for Job Scheduling**
Scenario: A company uses Simulated Annealing to optimize job scheduling. The initial schedule is far from optimal, but over time, the algorithm refines the solution.
Why does Simulated Annealing accept worse solutions early on?
    A. To guarantee the best solution.
    B. To speed up the search.
    C. To escape local optima.
    D. To find the shortest solution.

15. (2 points) Which of the following search strategies is uninformed?
    A. A* Search
    B. Heuristic Search
    C. Greedy Best-First Search
    D. Depth-First Search (DFS)

16. (2 points) Which of the following heuristics is admissible?
    A. A heuristic that always overestimates the cost to the goal.
    B. A heuristic that occasionally underestimates the cost to the goal.
    C. A heuristic that never overestimates the cost to the goal.
    D. A heuristic that is not guaranteed to find the best path.

17. (2 points) In Constraint Satisfaction Problems, which algorithm is often combined with backtracking to make the search more efficient?
    A. Hill Climbing
    B. A* Search
    C. Forward Checking
    D. Depth-First Search

18. (2 points) What is the main difference between Iterative Deepening Search (IDS) and Depth-First Search (DFS)?
    A. IDS uses a queue, while DFS uses a stack.
    B. IDS gradually increases the depth limit.
    C. IDS finds the optimal solution, but DFS does not.
    D. IDS is incomplete, while DFS is complete.

19. (2 points) In Hill Climbing with random restarts, what is the purpose of the random restart?
    A. To guarantee finding the global optimum.
    B. To explore all possible states.
    C. To avoid getting stuck in local optima.
    D. To speed up the search.

20. (2 points) Which of the following is a key feature of Simulated Annealing?
    A. It is a greedy search algorithm.
    B. It guarantees finding the global optimum.
    C. It occasionally accepts worse solutions.
    D. It only accepts solutions that improve the objective function.

3 Pseudecode (40 points, 10 pts each)

21. (10 points) **Breadth-First Search (BFS)**
Fill in the blanks in the pseudocode below.

```
Procedure BFS(graph, start)
    Initialize visited as an empty set
    Initialize queue with start node
    Initialize result as an empty list

    While queue is not empty:
        current_node <-- Dequeue the first element from queue

        If current_node is not in visited:
            Add current_node to result
            Mark current_node as visited

            For each neighbor of current_node in graph[__________]:
                If neighbor is not in visited:
                    Enqueue neighbor to queue

    Return result
End Procedure
```

22. (10 points) **Hill Climbing Algorithm**
Fill in the blanks in the pseudocode below.

```
Procedure Hill_Climbing(start, evaluate, neighbors)
    current <-- start
    current_value <-- evaluate(current)

    While True:
        next_move <-- None
        For each neighbor in neighbors(current):
            neighbor_value <-- evaluate(__________)

            If neighbor_value > current_value:
                next_move <-- neighbor
                current_value <-- __________

        If next_move is None:
            Return current  // Local maximum found

        current <-- __________
End Procedure
```

23. (10 points) **A* Search Algorithm**
Fill in the blanks in the pseudocode below.

```
Procedure A_Star(graph, start, goal, heuristic)
    Initialize open_list as a priority queue with start node
    Initialize came_from as an empty map
    Initialize g_score[start] <-- 0
    Initialize f_score[start] <-- heuristic(start, goal)

    While open_list is not empty:
        current_node <-- Node in open_list with the lowest f_score value

        If current_node = goal:
            path <-- Reconstruct path from came_from
            Return path

        Remove current_node from open_list

        For each neighbor, cost in graph[__________]:
            tentative_g_score <-- g_score[current_node] + cost

            If neighbor is not in g_score OR
            tentative_g_score < g_score[__________]:
                came_from[neighbor] <-- current_node
                g_score[neighbor] <-- tentative_g_score
                f_score[neighbor] <-- g_score[neighbor] + heuristic(neighbor, goal)

                Add or update neighbor in open_list with priority
                f_score[__________]

    Return Failure (No path found)
End Procedure
```

24. (10 points) **Iterative Deepening Search** Write the pseudocode for the Iterative Deepening Search (IDS) algorithm. Your pseudocode should take a graph, a start node, and a goal node as input, and return the path from the start to the goal. You will need to perform a Depth-Limited Search (DLS) in each iteration.

```
Procedure Iterative_Deepening_Search(graph, start, goal)
    For depth_limit from 0 to infinity:
        result <-- Depth_Limited_Search(graph, start, goal, depth_limit)

        If result not equal failure:
            Return result

Procedure Depth_Limited_Search(graph, node, goal, limit)
    If node = goal:
        Return path

    Else If limit = 0:
        Return failure

    Else:
        For each neighbor in graph[node]:
            result <-- Depth_Limited_Search(graph, neighbor, goal, limit - 1)
            If result not equal failure:
                Return result

    Return failure
End Procedure
```

4 Concepts (20 Points)

25. (5 points) Explain the differences between Breadth-First Search (BFS), Depth-First Search (DFS), and Iterative Deepening Search (IDS). When would you prefer one over the other?

26. (5 points) In Simulated Annealing, how does the temperature parameter affect the search process? How does it help the algorithm avoid local optima?

27. (5 points) Discuss how heuristic search algorithms (such as A* or Greedy Best-First Search) differ from uninformed search algorithms. Provide an example where heuristic search performs better.

28. (5 points) In Constraint Satisfaction Problems (CSPs), explain how forward checking and arc consistency can be used to improve the efficiency of backtracking search. Provide examples of when these techniques would be useful.