Homework 1

Instructions

Points: Please see the points for each problem.
Submission: Submit completed homework as a PDF file. Handwritten work or photos of handwritten work must be neat and legible.

Points Summary

Question Number	Points Possible	Points Earned
1	3	_____
2	3	_____
3	3	_____
4	3	_____
5	3	_____
6	3	_____
7	3	_____
8	3	_____
9	3	_____
Total	**27**	_____

1. It is a training day for a swarm of robotic miners in a hazardous environment. Each of the k miners starts in its own assigned start location s_i in a large maze of size $M \times N$, meaning M columns and N rows. Each miner's goal is to return to its own dumpsite stationed at location g_i. Along the way the miners must gather all chucks of Strontium 90 (think Pacman dots) in the maze. [3]

   At each time step, all k miners move one unit to any open adjacent square. The only legal actions are Up, Down, Left, or Right. It is illegal for a miner to wait in a square, attempt to move into a wall, or attempt to occupy the same square as another miner. To set a record, the miners must find an optimal collective solution.

   Define a minimal state space **representation** for this problem.

   Solution: The state can be represented as a tuple:

   $$S = (P_1, P_2, \ldots, P_k, C)$$

   where:

   - $P_i = (x_i, y_i)$ is the coordinate position of the i-th miner, for $i = 1 \ldots k$.
   - C is a boolean vector (or bitmask) of length Z (where Z is the total number of Strontium chunks), indicating whether each chunk has been collected or not.

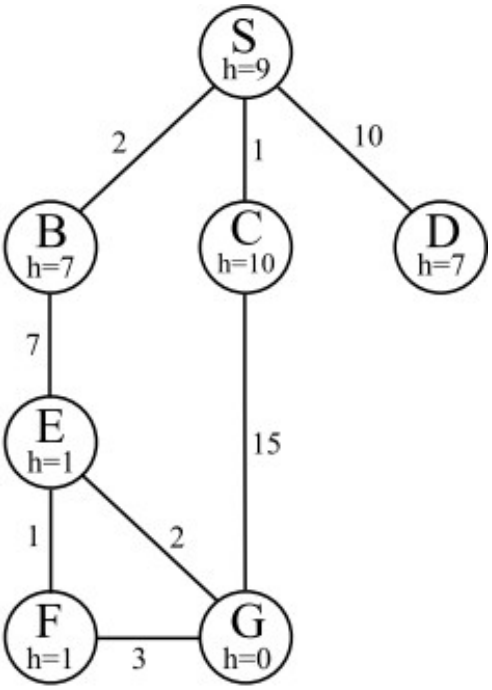
2. Refer to Question 1. How large is the state space? [3]

Solution: Let $M \times N$ be the size of the grid, k be the number of miners, and Z be the number of Strontium chunks.

- There are (MN) possible positions for each miner. For k miners, there are $(MN)^k$ position configurations.
- There are 2^Z possible states for the chunks (collected/uncollected).

Total state space size: $O((MN)^k \cdot 2^Z)$.

3. Consider the search graph shown below. S is the start state and G is the goal state. All edges are bidirectional. [3]



For the following search strategy, give the path that would be returned, or write none if no path will be returned. If there are any ties, assume alphabetical tiebreaking—nodes for states earlier in the alphabet are expanded first in the case of ties.

Depth-first graph search

Solution: Path: $S \to B \to E \to F \to G$
Explanation: DFS explores as deep as possible before backtracking. From S, B is expanded first (alphabetical tiebreaking). From B, E is the only unvisited neighbor. From E, F is expanded before G (alphabetical). From F, G is reached (goal found).

4. Refer to the figure in Question 3. Breadth-first graph search [3]

Solution: Path: $S \to C \to G$
Explanation: BFS explores level by level. Level 0: S. Level 1: B (cost 2), C (cost 1), D (cost 10). C is expanded first (alphabetical tiebreaking). From C, G is reached directly (goal found). Total cost: 1 + 15 = 16.

5. Refer to the figure in Question 3. Uniform cost graph search [3]

Solution: Path: $S \to B \to E \to G$
Explanation: UCS expands nodes by path cost $g(n)$. Frontier: S (g=0). Expand S: add B (g=2), C (g=1), D (g=10). Expand C: add G (g=16). Expand B: add E (g=9). Expand E: add F (g=10), G (g=11). G found via E with cost 11, which is less than 16, so path $S \to B \to E \to G$ is returned. Total cost: 2 + 7 + 2 = 11.

6. Refer to the figure in Question 3. Greedy graph search [3]

Solution: Path: $S \to B \to E \to G$
Explanation: Greedy expands the node with smallest heuristic $h(n)$. From S: B (h=7), C (h=10), D (h=7). B is expanded first (alphabetical tiebreaking). From B: E (h=1). From E: F (h=1), G (h=0). G has the smallest heuristic, so it is expanded and goal is found. Path: $S \to B \to E \to G$.

7. Refer to the figure in Question 3. A* graph search [3]

Solution: Path: $S \to B \to E \to G$
Explanation: A* expands by $f(n) = g(n) + h(n)$. Initial: S (f=0+9=9). Expand S: B (f=2+7=9), C (f=1+10=11), D (f=10+7=17). Expand B (f=9): E (f=9+1=10). Expand E (f=10): F (f=10+1=11), G (f=11+0=11). Expand G (f=11): goal found. Path: $S \to B \to E \to G$. Total cost: 11.

8. List the major characteristics of each search in the table below. [3]

Search	**Main Characteristics**
BFS	
DFS	
UCS	
Greedy	
A*	

Solution:

Search	Main Characteristics
BFS	Complete, Optimal (for unit costs), Time $O(b^d)$, Space $O(b^d)$. Explores layer by layer.
DFS	Incomplete (if cycles/infinite), Not Optimal. Time $O(b^m)$, Space $O(bm)$. Explores deep paths first.
UCS	Complete, Optimal. Expands node with lowest path cost $g(n)$. Time/Space depend on cost contours.
Greedy	Incomplete, Not Optimal. Expands node with lowest heuristic $h(n)$. Can get stuck in loops. Fast but risky.
A*	Complete, Optimal (if $h(n)$ admissible/consistent). Expands lowest $f(n) = g(n) + h(n)$. Large memory requirement.

9. When would an uninformed search outperform a smarter search? [3]

Solution: Uninformed search (like BFS) might outperform informed search (like A*) in scenarios where:

- The heuristic function is computationally expensive to calculate, and the goal is shallow or the branching factor is small.
- The heuristic is of poor quality (misleading), causing the informed search to explore a large number of unnecessary nodes (e.g., a large "garden path").
- The problem space is small enough that the overhead of maintaining a priority queue in A* exceeds the cost of a simple FIFO queue in BFS.