Kanchana T

2022UG000062

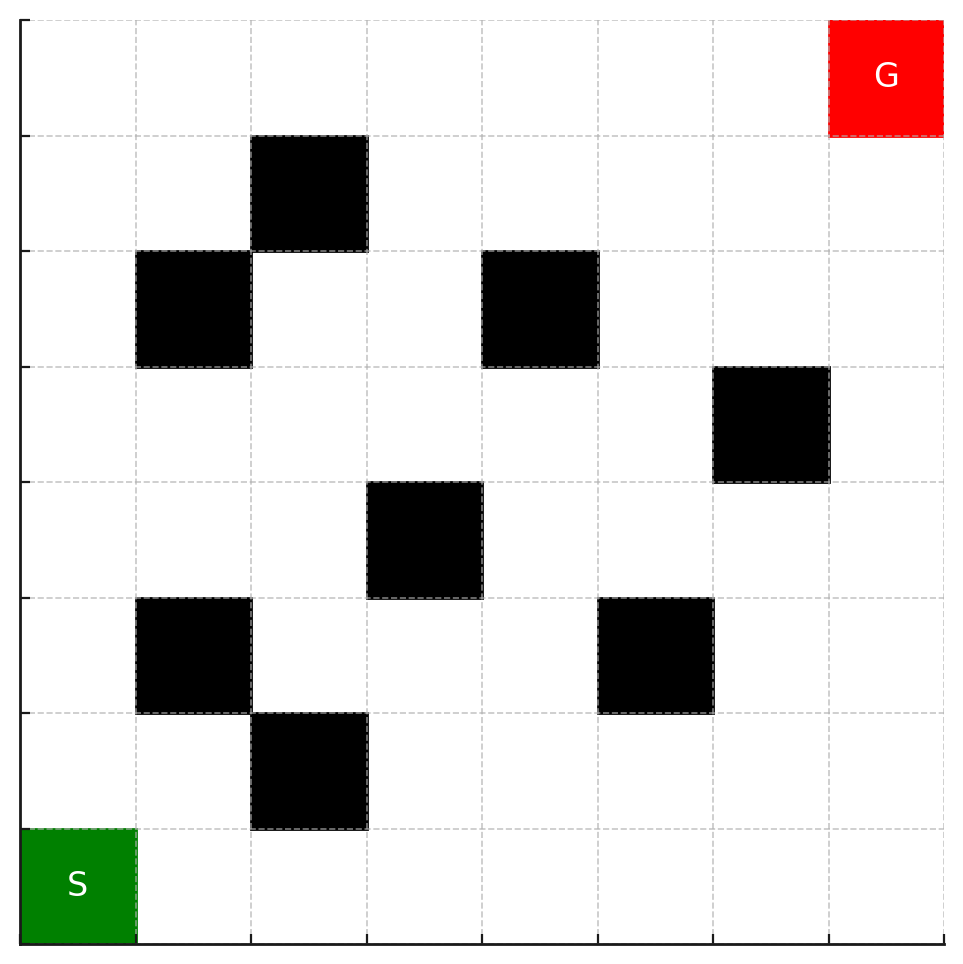
Reinforcement Learning

B.tech(Hons) Computer Science and Data Science

**Reinforcement Learning**

**Assignment 1**

**Design, Analyse, and Simulate a Maze World as an MDP**

****

1. **Formalize the Grid Maze as an MDP**

* **State space S:** Each state is represented by grid coordinates (row,col) where 0≤row, col<8. Walls (blocked cells) are not part of the free state space. The **goal state** is terminal.
* **Action space A:** {U = (+1, 0), D = (-1, 0), L = (0, -1), R = (0, +1)}.
* **Transition function p(s’|s, α) :** If action leads to a **free cell,** move there deterministically. If action leads to a wall or off-grid, agent.
* **Reward function r(s, α, s’):** Normal move (-0.1), Collision with wall (-1) and Reaching goal (+10).
* **Assignment instance**
* Grid size: 8 x 8
* Start state: (0, 0)
* Goal state: (7, 7)
* walls : {(1, 2), (2, 1), (2, 5), (3, 3), (4, 6), (5, 1), (5, 4), (6, 2)}

1. **Implement and Simulate a Random-Policy Agent​**

**Random Policy Agent**

* **Strategy:** Choose each action uniformly at random from {U, D, L, R}.
* **Performance (20 episodes):**
* Successes: **7/20**
* Steps: Min = 47, Max = 200, Avg ~ 172.25
* Reward: Min -91.99, Max ~ -8.10, Avg ~ -61.84

**Sample trajectory (failed, truncated):**

(0, 0) U -> (1, 0) D -> (0, 0)R -> (0, 1) -> (1, 1).....(200 steps, no goal)

**Greedy Policy Agent**

* **Strategy:** At each step, choose the action that **minimizes Manhattan distance** to the goal, avoiding walls.
* **Performance (20 episodes):**
* Successes: **20/20**
* Steps: Min = Max = Avg = **14**
* Reward: Min = Max = Avg = **+8.7**

**Sample trajectory (successful):**

(0, 0)U -> (1, 0)U -> (2, 0)......(7, 0)R -> (7, 7)

1. **Analysis and Exploration**

**1. Effect of Wall Placement**

* **Walls** act as **barriers** that force agents to take detours.
* For the **random agent**, walls make it far more likely to get stuck in loops or bounce between blocked moves, since the agent has no memory or planning.
* For the **greedy agent**, as long as the walls do not completely block the Manhattan path, the agent can still reach the goal efficiently by always moving closer. But if walls created a dead-end, the greedy policy could fail without backtracking.

## **2. Effect of Reward Values**

* **Step cost (−0.1):** pushes the agent to reach the goal in fewer steps. Without it, random wandering would not be penalized, and greedy agents would not be distinguished by efficiency.
* **Wall/off-grid penalty (−1):** discourages illegal moves. For the random agent, frequent penalties explain the very low average reward.
* **Goal reward (+10):** ensures reaching the goal is much better than looping forever. It gives greedy agents a consistent positive reward and motivates exploration in RL settings.

## **3. Simulation Results: Random vs Greedy**

| **Metric** | **Random Policy** | **Greedy Policy** |
| --- | --- | --- |
| Episodes (runs) | 20 | 20 |
| Successes (reached goal) | **7 / 20 (35%)** | **20 / 20 (100%)** |
| Steps (min / max / avg) | 47 / 200 / 172.25 | 14 / 14 / 14.00 |
| Reward (min / max / avg) | -92 / -8.1 / -61.84 | 8.7 / 8.7 / 8.7 |

**Observations:**

* Random agents succeed only sometimes, and even then with very long paths. Average reward is highly negative due to repeated penalties.
* A greedy agent succeeds **every single time**, in the minimum possible steps (14) with consistently positive reward.
* The difference highlights how **policy design** (heuristics) dramatically improves navigation efficiency.

## **4. Strengths & Weaknesses**

**Random Policy**

* Strengths: explores all directions; can eventually stumble upon the goal even in unknown environments.
* Weaknesses: inefficient, high variance in performance, very low rewards, often fails within step limit.

**Greedy Policy**

* Strengths: always finds the shortest path in this environment; highly efficient, consistent performance.
* Weaknesses: not robust to complex mazes — if walls created traps or dead ends, greedy could fail (no backtracking or planning).

## **5. Ways to Improve Navigation**

1. **Backtracking / Memory** → remember visited states to avoid loops.
2. **Wall-aware heuristic** → combine Manhattan distance with knowledge of blocked cells.
3. *Search-based policies (A, Dijkstra)*\* → guarantee optimal paths if the environment map is known.
4. **Reinforcement Learning** → let the agent learn policies through trial-and-error with Q-learning or value iteration.

**Conclusion**

The grid maze environment was successfully formalized as an MDP and tested with two agents. The random policy demonstrated poor efficiency, with low success rates and highly negative rewards due to frequent collisions and looping. In contrast, the greedy policy consistently reached the goal in the shortest possible path with positive rewards, highlighting the power of heuristic-driven strategies. Overall, the results show that wall placement and reward design strongly influence agent behavior, and that informed policies (like greedy or search-based) dramatically outperform uninformed random exploration.