AI Project Report: Pathfinding and Reinforcement Learning

**Team Members:**

# 1. Introduction

This report presents a comprehensive study of classical search algorithms and reinforcement learning techniques applied to grid-based pathfinding problems. The project is divided into two main phases:  
  
- Phase I: Comparative performance analysis of traditional search algorithms (BFS, DFS, UCS, A\*, IDS, Greedy, Hill Climbing, Simulated Annealing, and Genetic Algorithm).  
- Phase II: Implementation of a Q-learning algorithm to train an agent for optimal navigation in a dynamic environment.  
  
We explore execution time, path optimality, and efficiency in Phase I, and learning behavior, rewards, and Q-values in Phase II.

# 2. Phase I: Search Algorithms Performance Analysis

## 2.1 Problem Modeling

**State Space Representation:**

- Robot's position (x, y)  
- Delivery status of multiple goals  
- Grid configuration with obstacles

Grid Size: 30x30  
Number of Goals: 3  
State Space Size: Approx. 3,200 states

**Initial State: Robot starts at position (0, 0)**

**Goal State: All delivery goals have been visited and marked as delivered.**

**Actions:**

- Move(Direction): North, South, East, or West, restricted by obstacles.  
- Deliver(Position): When robot reaches a goal location.

**Transition Function: Updates robot's position or goal delivery status accordingly.**

**Modeling Assumptions: Static grid with fixed obstacles. Random goal placement in obstacle-free cells. Delivery occurs upon reaching the goal.**

## 2.2 Algorithm Performance Summary

Algorithm Performance Summary:  
BFS: Optimal path, moderate execution time, high memory usage.  
DFS: Fastest, low memory usage, suboptimal long paths.  
UCS: Optimal, moderate time and memory.  
A\*: Best balance of optimality and efficiency.  
IDS: Very low memory, high time due to iterative depth.  
Greedy/Hill Climbing: Fast but often suboptimal.  
Simulated Annealing: Balanced but slower.  
Genetic Algorithm: Explorative but computationally expensive.

## 2.3 Analysis and Reflections

A\* delivered the most consistent performance, balancing optimality and efficiency.  
DFS performed the fastest but often produced suboptimal, long paths.  
Simulated Annealing and Genetic Algorithm offered exploration but at a high computational cost.  
Greedy and Hill Climbing were efficient but heavily reliant on the heuristic, often missing the optimal path.  
IDS was theoretically sound but impractical for large grids due to time complexity.

# 3. Phase II: Q-Learning for Grid Navigation

## 3.1 Environment Design

Grid Dimensions: 8x8  
Start Point: Top-left corner (0,0)  
Goals: Randomly placed in obstacle-free cells  
Obstacles: Static  
Agent’s Task: Navigate from start to goal while avoiding obstacles

## 3.2 Q-Learning Setup

State Space: Agent’s position (x, y)  
Actions: Move North, South, East, West  
Rewards: Positive for reaching the goal; negative for hitting obstacles or going off-grid

Hyperparameters:  
Learning Rate: 0.1  
Discount Factor: 0.99  
Exploration Rate: Decaying  
Episodes: 500

Action Selection: Epsilon-greedy

## 3.3 Results and Analysis

Reward Statistics:  
Average Reward/Episode: 201.70  
Max Reward: 285.55  
Min Reward: -258.80  
Std Dev: 122.85  
Final 10 Episodes Avg Reward: 278.47

Q-Value Analysis:  
Average Q-value: 8.61  
Max Q-value: 226.30  
Min Q-value: -2.45  
Std Dev: 31.01

Sample Q-values:  
Start State - N: 0.17, S: -0.08, E: 0.26, W: -0.10  
Goal (6,2) - N: 0.10, S: -0.08, E: 0.25, W: -0.26

Final Path Length: 15  
Path: ['E', 'S', 'S', 'S', 'E', 'S', 'S', 'S', 'E', 'E', 'E', 'E', 'E', 'E', 'S']

## 3.4 Conclusion

The Q-learning model effectively learned to navigate the grid world. With 500 training episodes:  
- The agent improved its cumulative rewards.  
- Q-values reflected improved confidence in optimal actions.  
- Final paths demonstrated efficiency in reaching the goal.

# 4. Overall Conclusion

This project demonstrates the strengths and limitations of both traditional and learning-based pathfinding methods:  
  
- Classical Algorithms (A\*, UCS, BFS) provide deterministic, optimal results but require high computation in large state spaces.  
- Reinforcement Learning (Q-learning) offers adaptability and generalization, showing strong performance after sufficient training.  
  
For static environments with known goals, classical methods like A\* are reliable. For dynamic or unknown environments, Q-learning provides a promising alternative, capable of learning effective strategies through interaction.