**National University of Computer & Emerging Sciences**

**Karachi Campus**



**PROJECT TITLE: Maze Navigation and Path Finding**

**Project Report**

**Artificial Intelligence [AI]**

**Section: 4C**

**Group Members:**

**22K-4017 Ali Afridi**

**22K-4010 Tanveer Ali**

**22K-4090 Muhammad Talha**

**Abstract:**

In this project, we introduce an AI-driven maze solver implemented in Python using the tkinter library for visualization. The maze solver employs two search algorithms: Breadth-First Search (BFS) and A\* Search, catering to different problem-solving paradigms. The primary goal is to navigate through a randomly generated maze while avoiding obstacles, demonstrating the application of search algorithms in solving real-world problems.

**1. Introduction:**

Our project focuses on developing an AI-driven maze solver, showcasing the practicality of search algorithms in a gaming context. By navigating through a maze environment, users can observe firsthand how AI algorithms strategize and make decisions to reach a goal state efficiently.

**2. Problem Definition:**

The core challenge lies in efficiently navigating a maze while avoiding obstacles. The maze environment comprises a grid of cells, with certain cells marked as obstacles. The AI agent must find the optimal path from a start cell to a goal cell, employing either BFS or A\* Search algorithms based on user selection.

**3. Relevant Method/Model:**

We implement two search algorithms: BFS for uninformed search and A\* Search for informed search. BFS explores all possible paths systematically, while A\* Search uses heuristic information to guide its search toward the goal more intelligently.

**4. Performance Measurement:**

Performance is evaluated based on the algorithm's ability to find a valid path from start to goal while avoiding obstacles. Metrics include the time taken to find the path and the optimality of the generated path compared to the shortest path.

**5. Risks and Dependencies:**

Potential risks include inefficient pathfinding due to algorithmic limitations or maze complexity. Dependencies on the tkinter library for visualization and user interaction underscore the need for seamless integration and compatibility.

**6. Run Performance Checks:**

We conduct rigorous performance evaluations, analyzing the efficiency and correctness of pathfinding algorithms under various maze configurations. Performance checks include time complexity analysis and visual representation of pathfinding processes.

**7. Implementation Details:**

Our implementation utilizes object-oriented programming, with classes for the maze environment, search agents, and GUI components. The maze is randomly generated, and users can interactively mark obstacles using mouse clicks.

**8. Results and Analysis:**

Results showcase the effectiveness of BFS and A\* Search in solving maze navigation problems. We analyze the performance, path optimality, and computational efficiency of each algorithm, providing insights into their strengths and limitations.

**9. Future Directions:**

Future enhancements may include additional search algorithms, maze generation algorithms, and optimizations for real-time pathfinding. Integration with online platforms or multiplayer functionality could further extend the project's scope.

**10. Conclusion:**

In conclusion, our AI-driven maze solver project exemplifies the application of search algorithms in gaming and problem-solving domains. Through systematic methodology and performance evaluation, we aim to showcase the efficacy of AI techniques in navigating complex environments, paving the way for further exploration in AI-driven gaming solutions.

**Reference:**

https://www.youtube.com/watch?v=KEIvXwUm8iE&t=57s