Quant Club Selections 2025-26: Coding Task

Developing an Optimal Algorithm for the game 2048

Objective: The goal of this task is to design and implement an algorithm that can reach the 2048 tile in the popular game 2048 using the minimum number of moves. Participants are required to understand the rules of the game and develop an efficient Python-based strategy.

Reference: https://en.wikipedia.org/wiki/2048_(video_game)

Game Rules:

1. **Grid and Tiles:** 2048 is played on a 4x4 grid with numbered tiles that slide in one of four directions: up, down, left, or right.

2. **Merging Mechanism:** When two tiles of the same number collide during a move, they merge into a single tile with a value equal to their sum.

3. **New Tile Generation:** After each move, a new tile (2 or 4, **chosen randomly**) appears at an empty **random** position on the board.

4. **Movement Restrictions:**

    ○ Tiles move as far as possible in the chosen direction until they are blocked by another tile or the edge of the board.

    ○ If a move does not change the board state, it is considered invalid.

5. **Winning Condition:** The game is won when a tile with a value of 2048 appears on the board.

6. **Losing Condition:** The game ends when no valid moves remain (i.e., the grid is full and no adjacent tiles can merge).

Task Requirements:

1. **Understanding the Game Mechanics:** Participants should first experiment with the game manually to understand its mechanics and build a manually playable version of the game. It may be a command line version or using a GUI library, you are free to experiment.

2. **Algorithm Design:**

    - Develop a strategy that minimizes the number of moves required to reach the 2048 tile.
    - Consider heuristics such as prioritizing merges, maintaining a structured board, or maximizing high-value tile placement.

3. **Implementation in Python:**

    - Write a Python script that simulates the 2048 game and implements the designed strategy.
    - Use an appropriate data structure to represent the board (e.g., a 2D list or NumPy array).

4. **Performance Evaluation:**

    - Measure the number of moves taken to reach 2048.
    - Compare different strategies and their effectiveness.
    - Come up with a few parameters of your own that help us determine the performance and efficiency of your algorithm.

Submission Guidelines:

- Provide the Python notebook implementing the algorithm.
- Make sure the notebook is well commented to explain the algorithm you used. Treat your notebook as code file cum report. Try to include as much explanation as you can in the comments.

Good luck!