PROJECT 1 – SEARCH

Subject: Fundamentals of Artificial Intelligence

I. Finding path by search algorithm (50%)

1. Introduction

- In this project, students research and implement the searching algorithm. In addition, students have to visualize the result of the searching algorithm.

2. Requirements

- Programming language: Python (for visualization, we recommend students use turtle library or Pygame of Python)
- Timeline: 3 weeks.
- Final product: find_path_search folder, includes:
  - Code folder: include every coding files.
  - Report folder: include file report.pdf:
    - Student's information
    - Each algorithm, student report:
      - The idea of the algorithm.
      - Example (reference section input/output)
      - Conclusion, pros and cons.
- Evaluation:
  - Implement 5 searching algorithm: 75%.
  - Report: 25%
- Every cheat/copy/lie will be punished with a course score of 0.

3. Problem

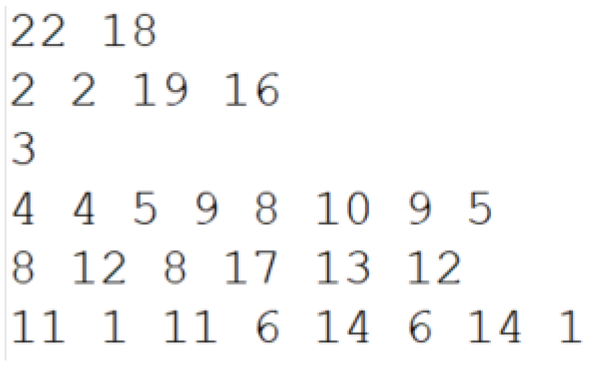
a. Problem description

- The robot has been sent to a maze of size M x N, and the robot has to find the path from the Source (starting position) to the Goal (ending position). The robot allows to move in 4 directions: up, down, left, right. In the maze, there are some obstacles.
- The student as asked to implement 5 search algorithms:
  - **Breadth-first search**
  - **Uniform-cost search**
  - **Iterative deepening search** that uses depth-first tree search as core component and avoids loops by checking a new node against the current path.
  - **Greedy-best first search** using the Manhattan distance as heuristic.
  - **Graph-search A*** using the Manhattan distance as heuristic.

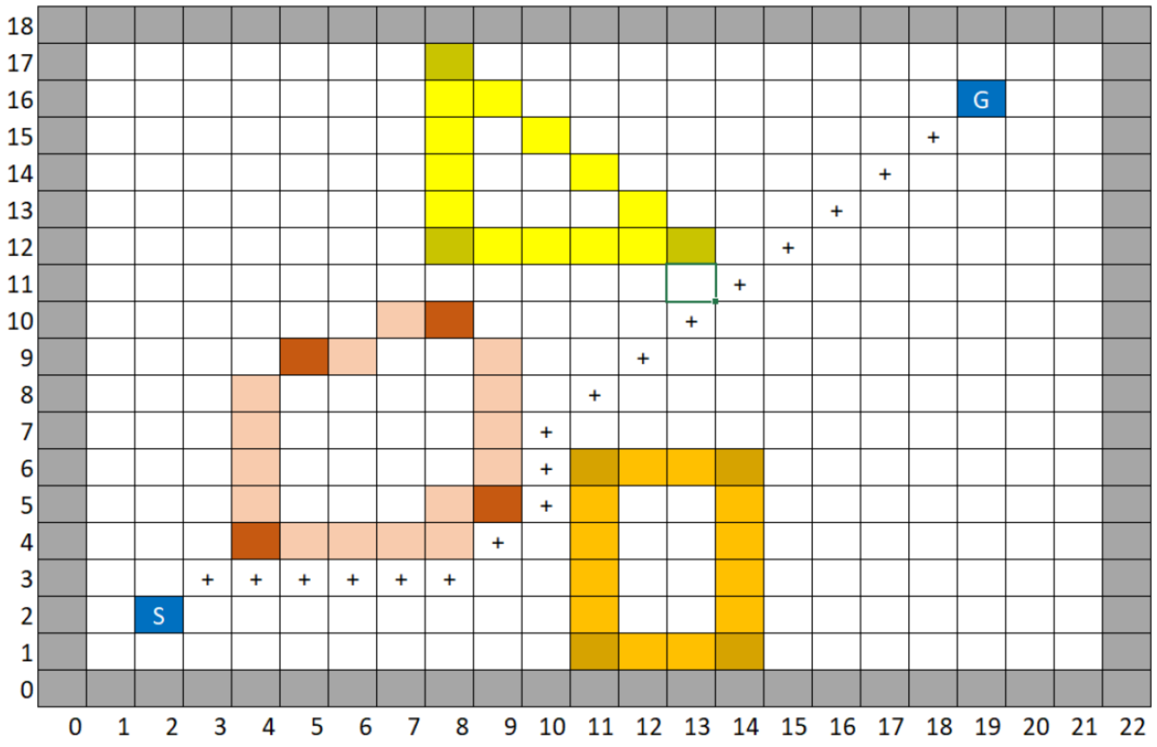
b. Input/output format

- The format of the input file:
    - First line: the size of the maze width, height.
    - Second line: the position of the Source and Goal. For example: 2 2 19 16 meaning source point is (2, 2) and goal point is (19, 16).
    - Third line: the number of the obstacles in the maze.
    - The next following line, defining the obstacle by the rule:
        - The obstacle is a Convex polygon.
        - A polygon is a set of points that are next to each other clockwise. The last point will be implicitly concatenated to the first point to form a valid convex polygon.
- The output:
    - Graphical representation of polygons.
    - Graphical representation of expanded node.
    - Graphical representation of final path.
    - Cost of expanded node (total number of expanded nodes)
    - Cost of final path (total number of nodes in the final path)

- The example of input.txt
  (Everything is relative, depend on your implementation)



```
22 18
2 2 19 16
3
4 4 5 9 8 10 9 5
8 12 8 17 13 12
11 1 11 6 14 6 14 1
```



II. Playing game with adversarial search (50%)

1. Introduction

- In this project, students research and implement the adversarial searching algorithm.
- In addition, students implement an application (tic-tac-toe problem) and apply the adversarial technique to solve that tic-tac-toe.

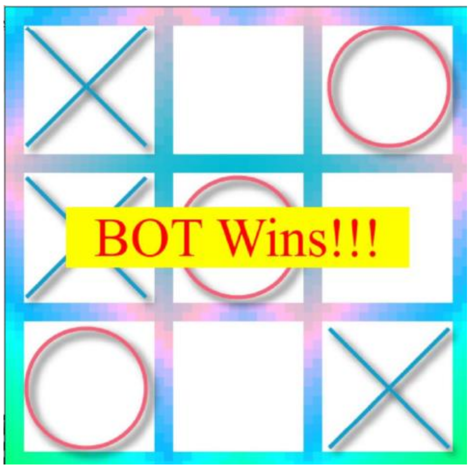
2. Requirement

- Programming language: Python (for visualization, we recommend students use tkinter library or Pygame of Python)
- Final product: play_game_adversarial folder, includes:
  - Code folder: include every coding files.
  - Report folder: include file report.pdf
    - Student’s information.
    - Introduce about the algorithm.
      - Describe the idea of that algorithm.
      - Completeness.
      - Time/space complexity.
    - Link demo application (YouTube or Google drive or One drive).

- Evaluation:
    - Implement adversarial algorithm for tic-tac-toe: game 50%
    - Interface application: 20%.
    - Report: 30%.
- Every cheat/copy/lie will be punished with a course score of 0.

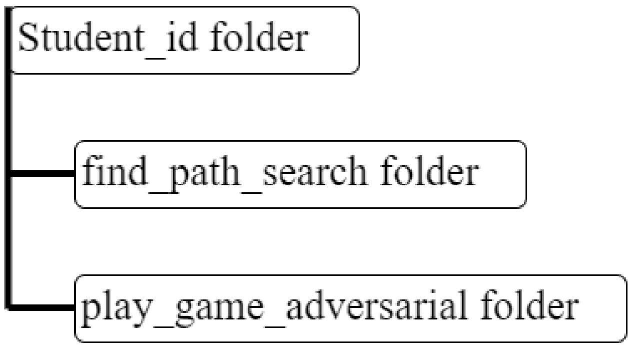
3. Problem

- Students implement a tic-tac-toe game (Vietnamese: trò chơi caro). For simplicity, students just need to implement 3x3, 5x5 and 7x7 maps. Student will control player 1, Computer will control player 2 (and vice versa).
- **Students choose any adversarial search. Using that adversarial search to find the optimal path, which will help the computer to win this game.**
- Notes: students must implement the interface of tic-tac-toe game. The main purpose of this project is learning Adversarial search, please do not focus on application or interface (just easy to look are enough).
- Example of tic-tac-toe game, implementing by Pygame:



III. Submission and References

1. Submission



2. References

- The document in the Computer Science Department at the University of Science, Vietnam National University, Ho Chi Minh City.
- The book: “Artificial Intelligence: A Modern Approach 3th Edition”