Games and Logic

In this homework, there are 3 questions about the topics you covered in the class.

Questions 1 and 2 are about the logic topic. In first question you need to represent in first-order logic. In the second question, you are going to use resolution inference algorithm.

In the third question, you need to implement a solution to a described decision problem. To do so, you are going to implement **minimax** algorithm that has been covered in the class.

Problem 1 - First-Order Logic Representation (10 points)

Represent the following sentences in first-order logic (FOL). You may define predicates as you like.

- All cats are animal.
- Everyone who owns a car also has a bicycle.
- There is a student registered to AI class who talks to other students that are registered to AI class.

Problem 2 - FOL, Inference via Resolution (30 points)

English, French, Russian and Turkish belong to language category (Same as "English, French, Russian and Turkish are languages").

Arda, Cihan and Gamze all are students in the same university. Each student in the university speaks **at least** two languages.

Fish and hamburger belong to food category.

Classic, jazz and rock belong to music category.

Students who speak French like jazz music and dislike rock music. All students who speak Russian like rock music. All students who like hamburger speak English but do not speak Turkish. All students who like fish and classic music speak Turkish.

Arda likes jazz music, hamburger, fish but dislikes classic music, rock music.

Cihan dislikes whatever music Arda likes, and he likes whatever music Arda dislikes. He likes hamburger and dislikes fish.

Gamze likes fish, classic music but dislikes hamburger, jazz music and rock music.

Using first-order logic:

(a) Construct a knowledge-base using the given facts.

(b) Use resolution to answer the following queries:

  (1) Which student(s) speak(s) French

  (2) Which student(s) speak(s) both English and Turkish

Please show all of your work. While constructing the knowledge base, try including FOL representations of each sentence in the question. While performing inference via resolution, show necessary steps to obtain answers of the queries.

Problem 3 - Optimal Decision in Games (60 points)

Kamar likes to play games. One of his favorite games is a turn-based game that is played on a grid area. In this game, in each turn, a player picks one of the empty edges on the grid (figure 1). If a player encloses a 1x1 box with the recently added edge, the player gains a point and an additional turn. Enclosing two 1x1 box with a single action allows the player to gain two points but still a single additional turn (an example gameplay is given in next section). Although he likes to play this game, he is not good at it. You need to help him by creating a **minimax** agent that finds the best move to play in a given game state. Your program is going to read initial game state from an input file. Some edges on the grid may already be occupied.

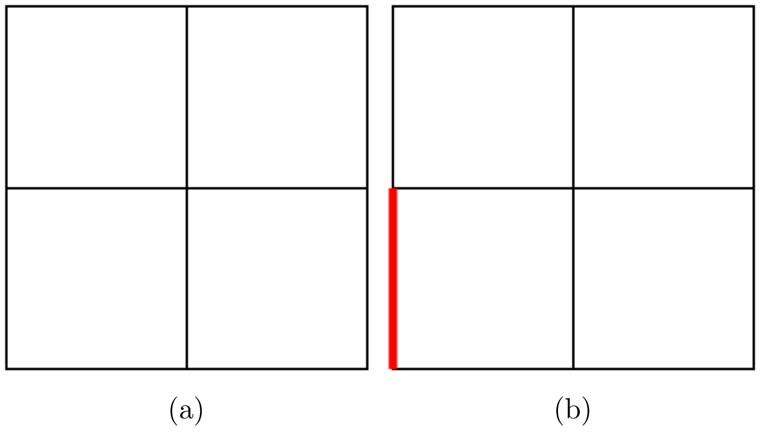

Figure 1: An example 2x2 game grid. (a) Starting position, none of the edges has been picked yet. (b) First player (red) picks lower-left vertical edge.

The game has the following rules:

- A player can choose any horizontal or vertical edge to pick.
- The red player always starts first, independent of the initial grid configuration.
- Enclosing a 1x1 box in the grid allows player to gain a point. A player can gain two points with a single action as well by enclosing two 1x1 boxes.
- If a player gains a score with its last move, the player gets to play an additional turn.

You need to:

(a) Write a program that reads the game grid from an input file.

(b) Implement a basic minimax agent to solve the game.

(c) Implement a minimax agent with alpha-beta pruning to solve the game.

(d) Compare two agents in terms of:

- The number of nodes evaluated in the search tree
- The running time

Some important aspects about your implementation:

- For the assignment, you are required to implement the standard minimax algorithm, that is, your evaluation function will always evaluate a given position from the perspective of the player max. If the player max is going to win the game from the given state, the utility of that state must be higher compared to utilities of losing states. Any other form of the minimax algorithm is not allowed and you will not get any grade.
- Do not use any heuristic functions for state evaluation. Value of states must be calculated exactly by searching down in game tree and propogating terminal state values appropriately.
- Your program **must** accept a command line argument, input file to read initial game state.

Input File Format

First line contains the number of rows in the grid (R). Second line denotes the number of columns in the grid (C). Each of next $C*(R+1)$ lines contains initial status of the horizontal edges: 1 if edge is occupied, 0 otherwise. Each of next $R*(C+1)$ lines contains the initial status of the vertical edges: 1 if edge is occupied, 0 otherwise. Next line contains initial score of player 1, and last line contains initial score of player 2. Edges are ordered from top-left to bottom-right.

Output Format

You will print two lines. The first line corresponds to the score of player 1 after optimal play, and the second line corresponds to the score of player 2 after optimal play.

Constraints

$1 \leq \mathrm{R} \leq 3$
$1 \leq \mathrm{C} \leq 3$
$1 \leq$ #empty edges in the given grid ≤ 16

Example Input Outputs

Example input 0:
1
2
0
0
1
0
0
1
1
0
0

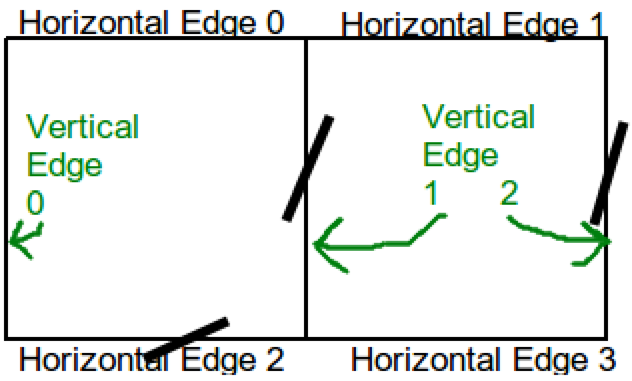

Figure 2: Visualization of grid from example input 0.

Example output 0:
1
1

Explanation of example 0:
The first player must either choose left or right side of the grid. The box at the chosen side will be enclosed by the second player, after that since the second player gets to play an additional move, he/she loses other side. Both players gain a score and the game results in a draw.

Example input 1:

```
2
2
0
0
0
0
0
0
0
0
0
0
0
0
0
0
```

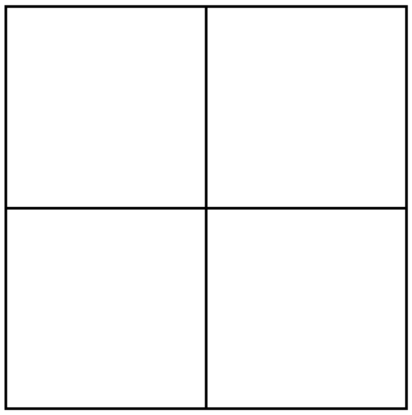
Figure 3: Visualization of grid from example input 1.

Example output 1:

```
3
1
```

Explanation of example 1:
If first player plays optimally, second player can only obtain at most 1 point.

An Example Full Gameplay

Here is an example of a full gameplay in a 1x2 grid, to make sure you understand the rules of the game correctly. Note that players in this example do not necessarily play optimally.

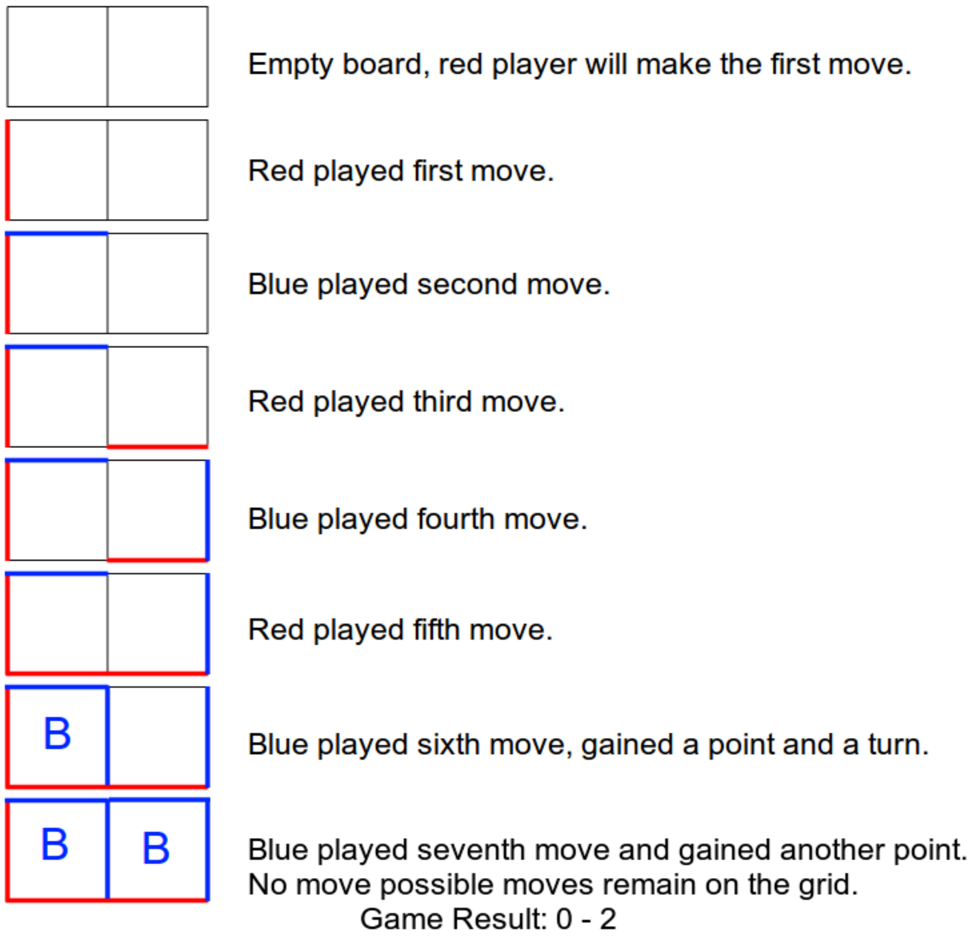
Empty board, red player will make the first move.
Red played first move.
Blue played second move.
Red played third move.
Blue played fourth move.
Red played fifth move.
B
Blue played sixth move, gained a point and a turn.
B
B
Blue played seventh move and gained another point.
No move possible moves remain on the grid.
Game Result: 0 - 2

Figure 4: A full gameplay example starting from initially empty grid with size 1x2

Submission

Submit your homework files through Ninova. Please zip and upload all your files using filename BLG435E_HW_2_STUDENTID.zip. You are going to submit:

1. All your code files for implementation of Q3.
2. A pdf file containing answers of the first two questions of the homework, required explanations/analyses about the third question of the homework and instructions to compile/run your code.

Your code must be able to compile and run on ITU's Linux Servers without any errors, otherwise your code may not be evaluated.

There is no restriction on the programming language.

Pay attention to deadline of the homework, including hour.

Important: In Q3, your solution can rely on existing minimax algorithms. However, you need to explain how the algorithms work in this problem and perform the requested analyses with sufficient explanations in your report. **Code usage without relevant references will be considered as plagiarism.**

Note that submitted homeworks may also be tested with different initial grids other than those that are given to you.

In case of any questions, feel free to send an e-mail to unlut@itu.edu.tr.