# Connect4-README

Gabriele Giacobazzi

## Contents

# 1 Instructions

## 1.1 To compile and run

- Unzip the archive with all the files in a new folder
- Go into the folder using a terminal

```
cd "name" change directory
ls / dir to list the contents of the folder (Linux OS/Windows OS respectively)
```

- Run the following commands from a terminal with Java 8 (and Java set as an environmental variable if run from a Windows OS) installed on the computer:

```
javac *.java
```

```
jar cfm Connect4.jar META-INF/MANIFEST.MF *.class
```

```
java -jar Connect4.jar
```

In case there are problems with the compilation it is possible to use directly the precompiled .jar file situated in the ./out folder

## 1.2 To play

- Use the "Players" menu to select the mode of the game and then click "New" in the "File" menu to initialize the environment. After that click on the "Start" button situated at the bottom of the window to start the game.
  - If a "Human" mode is selected, the moves button will be activated after clicking the start button
  - AI will play alone after clicking on the start button or after a human move.
- Using the "Options" menu it is possible to run simulations of AI games and save the output in a .txt file located in the main folder. The default number of simulations is 100.

### 1.2.1 Miscellaneous

- The first player plays Red tokens, while the second one Yellow ones.
- At the bottom of the window it is possible to track the number of the respective Players' moves and the time elapsed to make a move, in case an AI is playing.
- The simulations process between the agents are not immediate, so it is necessary to wait a little bit, especially when using a pruned minimax agent.

### 1.2.2 Problems

- Sometimes the new game action could get buggy: in that case it is suggested to restart the application.
- If the simulation process completion window does not appear after a while it is suggested to restart the application and retry the operation. It is possible to check the simulation progression from the terminal.
- In "AI vs AI" modes, the tokens are displayed in the grid after the game has finished.

## 2 Assignments

a)The evaluation function counts the number of threats of the maximizing player and the minimizing one and acts accordingly; if it sees that there is a 4-streak of any player it stops immediately and makes that move.
The default depth of the minimax algorithm is 6.
b)Using a $\alpha-\beta$ pruning approach we optimize the normal algorithm, hence we can increase the maximum depth of the trees the algorithm will compute; in this application the default depth of the $\alpha-\beta$ pruning minimax algorithm is 8. At the beginning of the game the compute times are really similar to the normal minimax, so there are no differences between the two; progressing into the game the $\alpha-\beta$ pruning starts to cut off the times by a lot.

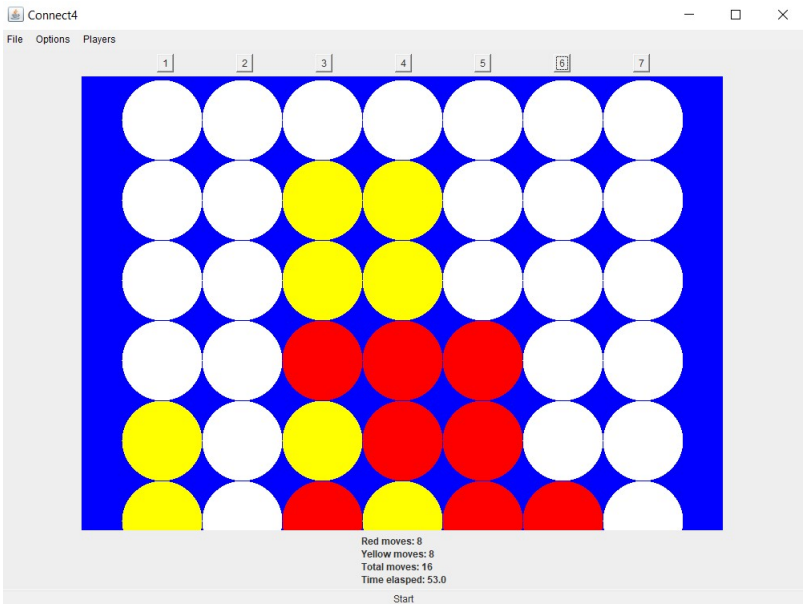


Figure 1: Minimax algorithm compute time

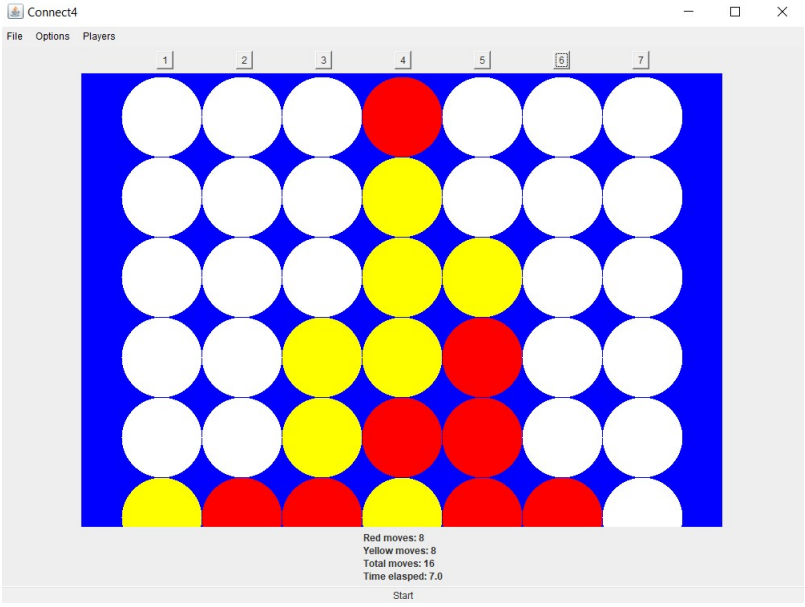


Figure 2: Minimax with $\alpha-\beta$ pruning approach algorithm compute time

c)After running the simulations these are the considerations that can be done about the various playing agents:

- Random agent: it never wins against a minimax based agent, because the latter can strategize a win without any problems.

- Minimax agent: against another default minimax agent the chance of winning is 50%. Against a $\alpha-\beta$ pruning minimax algorithm the latter always wins.
- $\alpha-\beta$ pruning agent: it wins against every kind of agent due to the better strategy and computations it does.

As a human the only possible win we can achieve is against the random agent, because the moves it does are random; against the minimax based agents it is (I think) impossible to win, because they will always try to stop our winning moves.