AI: Principles and techniques - Programming assignment 1

The goal of this assignment is to get familiar with some practical applications of Search algorithms by applying them to the game of N in a Row. You must implement a Tree structure[1] to store game states, and then implement the MinMax algorithm with and without alpha-beta pruning. The goal of this assignment is to compare the runtime complexity of both approaches and give you a better insight into search algorithms. As an additional challenge you may experiment by adding your own improved heuristics or try your hand at implementing the more advanced Monte Carlo Tree Search algorithm.

A framework is available in Java 11 and Python 3.12[2], which includes the basic game classes and provides interfaces for you to extend when implementing the assignment. We recommend you use this. Make sure you are using an appropriate runtime measure for your results, using actual time (e.g. ms) is ***not*** *sufficient.*

To help you get an overview, here is the assignment broken down in parts:

0. Become familiar with the code provided. What are the Variables, how is the game state (board) represented and how can you access it?
1. Implement a Tree structure to store subsequent game states, you will need this to implement the algorithms.
2. Implement the minimax/MinMax algorithm as discussed in the lectures, experiment to ensure your implementation is working correctly.
3. Implement alpha-beta pruning, using a runtime measure to compare the complexity between the agent with and without alpha-beta pruning for various N, board sizes and search depths.

Bonus points are offered if you experiment with at least one different heuristic(s) (1 bonus point extra; grade becomes 0.5 higher), exceptionally brave students may try implementing Monte Carlo Tree Search as a 3rd algorithm (1 bonus point extra; grade becomes 0.5 higher).

Be sure to use semantically rich naming for variables, constants, and functions, and to document your code well. If you use any code you did not write yourself, then be sure to attribute this appropriately, failure to do so may result in a plagiarism warning. It is advised to make this assignment in pairs.

Report

Write a short scientific report on your implementation, describing the project, your code, your experiments, their results, and your interpretation & conclusions. You can use the various parts of the assignment to help you structure the report.

In your report, make sure to reflect on your results and why you think you found them, especially if these do not match your initial expectations.

---

[1] We recommend that you use recursive functions to implicitly implement this tree structure. You may also use classes to implement a tree, but this is typically more work to do.

[2] There are some Numba decorators used, you do not have to modify them or implement them yourself for this assignment. They are used to speed up provided functions.

For a full list of elements to include in your report, check out the “writing a programming report” section at the course guide (learning task 0) and the assessment form for this assignment. We expect a formal (!) report of ca. 4 – 8 pages (not counting possible appendices).

Deliverables

Please submit your report as a PDF with the following naming convention: groupNumber_assignment1.pdf, and your code as a zipped folder groupNumber_assignment1.zip. Only one group member needs to submit the deliverables.

To hand in

Both a report (pdf) and the code (zip). Do not forget to list your name(s), student number(s), course, number of the task, date, etc. in your code and report. If you submit as a pair, let one of the students submit the report and code, and the other a text file with the name of the other student for easy reference.

The deadline of this assignment will be communicated through Brightspace.

Programming questions can be sent to the teaching assistants either during the practical sessions (preferably) or by email. Content-wise questions can be sent to the lecturer.