COMP 424 Final Project Game: *Colosseum Survival!*

Course Instructor: David Meger
Due Date: Dec 5th, 2022, 8:59PM EST

1. Goal

The main goal of the project for this course is to give you a chance to play around with some of the AI algorithms discussed in class, in the context of a fun, large-scale problem. This year we will be working on a game called ***Colosseum Survival!***

General questions should be posted in Ed. **This is a group project, with two students per group**. Working alone is OK if you prefer.

The detailed game rules, starter code, and programming instructions for your agent can be found at `https://github.com/dmeger/Project-COMP424-2022-Fall`.

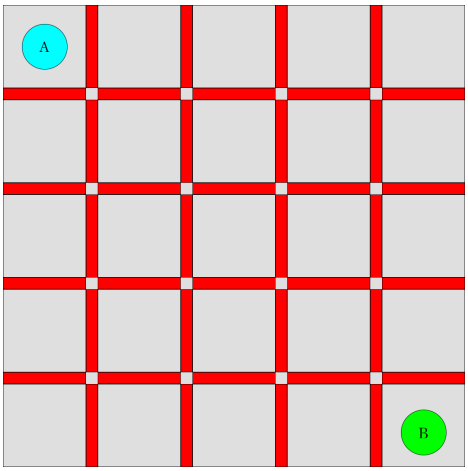

Figure 1: Gameboard

2. Evaluation

We will use several phases to group your agents by strength. A first pass is that you should definitely be able to beat the random agent provided. Next, make sure you can do well against a human player who has not practiced too much. Finally, a tournament against the strongest set of agents within the class will be run to determine the rankings. You do not have to win the tournament to get a satisfactory grade, but the winning team(s) will receive 100% of the performance grade.

In all phases, due to the deterministic nature of the game board, each match will consist of N games, giving both programs equal opportunity to play first. The evaluation phase will require a lot of matches so please be mindful of your program's runtime. We will perform a screening process by pairing your agent with a random agent to ensure the runtimes matches the expectations (check the Tournament Constraints in Section 5.1.1).

2.1 Implementing your own agent

You need to write your own *agent* and submit it for the class project. Detailed instructions of various parts of the game is available in the **README.md** file. Follow the steps there to implement and test your own agent. Be very precise about the instructions regarding your submitted student agent, because we need to be able to run your code automatically. **Important**: You should not modify any other files apart from `student_agent.py`. Any helper variables or functions you need should also be created within the same file. In the event of any update to the game code, the TAs will announce the necessary steps in Ed to pull the starter code.

3. Report

You are required to write a report with a detailed explanation of your approach and reasoning. The report must be a typed PDF file, and should be free of spelling and grammar errors. The suggested length is between 4 and 8 pages, but the most important constraint is that the report be clear and concise. You should use the source of this document [1] as a template to write your report in LaTeX. The report must include the following required components:

- An explanation of how your program works, and a motivation for your approach.
- A brief description of the theoretical basis of the approach (about a half-page in most cases); references to the text of other documents, such as the textbook, are appropriate but not absolutely necessary. If you use algorithms from other sources, briefly describe the algorithm and be sure to cite your source.
- A summary of the advantages and disadvantages of your approach, expected failure modes, or weaknesses of your program.
- If you tried other approaches during the course of the project, such as failed/weaker algorithmic ideas, or dummy agents to test against, summarize them briefly and discuss how they compared to your final approach.
- A brief description (max. half page) of how you would go about improving your player (e.g. by introducing other AI techniques, changing internal representation etc.)

4. Submission

First, fill your and your team members details in `authors.yaml` file in the repository. Then, submit to the My Courses folder. Do not create a zip, just select the multiple files:

- Report as a PDF file (not the latex source)
- student_agent.py

1. You can find the source of this document in Overleaf here: https://www.overleaf.com/read/vnygbjryrxrt. You would need to create an Overleaf account. Copy this project and create a new project to write your report. Replace the `author` information with your own group information.

- authors.yaml

You can make multiple submissions up to the deadline, but in this case we will be very strict about not accepting late code due to McGill's policy about no assessments during study day and final exams.

5. Grading Scheme

40% of the project grade will be allotted for performance in the tournament, and the other 60% will be based on your report.

5.1 Tournament Grading Scheme

The top scoring agent(s) will receive full marks for performance, other top agents in the tournament will will receive marks according to a linear interpolation scheme based on the number of wins/losses they achieve. To get a passing grade on the performance portion, your agent must beat the random player.

5.1.1 Tournament Constraints

During the tournament, we will use the following additional rules:

- **Execution Environment**. We will run the tournament on SOCS teaching Linux environment (as in the mimi.cs.mcgill.ca server and desktops in Trottier 3rd floor). These run Linux, Python Python 3.10.6. We will only install libraries as defined in `requirements.txt`, so you are not allowed to use external libraries. This means built-in libraries for computation are allowed but libraries made specifically for machine learning or AI are not. If you think you would require some external libraries not specified in `requirements.txt` and which is not a AI/ML specific library, please post a question in Ed to get an approval from the TAs, but we will almost always disallow things unless you really convince us we missed something needed.

- **Turn Timeouts**. During each game, your agent will be given no more than 30 seconds to choose its First move, and no more than 2 seconds to choose each subsequent move. The initial 30 second period should be used to perform any setup required by your agent (e.g. any pre-processing step). If your player does not choose a move within the allotted time, a random move will be chosen instead. If your agent exceeds the time limit drastically (for example, if it gets stuck in an infinite loop) then you will suffer an automatic game loss.

- **Illegal moves**. In the game, if your agent attempts to move to positions which are illegal, i.e. which requires more number of steps than K, then we will run a random walk over the game board. If your code fails during execution, then the game will also continue with a random move.

- **Multi-threading**. Your agent must be single threaded, run on only one processor and complete all computation when returning from the step function (as in, you are not allowed to "think" on your opponent's time).

- **File IO**. Your player will not be allowed to read and write files : all file IO is prohibited. In particular, you are not allowed to write files, so your agent will not be able to do any learning from game to game.
- **Memory Usage**. Your agent will run in its own process and will not be allowed to exceed 500 mb of RAM. Exceeding the RAM limits will result in a game loss.

You are free to implement any method of choosing moves as long as your program runs within these constraints and is well documented in both the write-up and the code. Documentation is an important part of software development, so we expect well-commented code. All implementation must be your own.

5.2 Report Grading Scheme

The marks for the write-up will be awarded as follows:

- Technical Approach: 20/60
- Motivation for Technical Approach: 15/60
- Pros/cons of Chosen Approach: 10/60
- Future Improvements: 5/60
- Language and Writing: 5/60
- Organization: 5/60

6. Academic Integrity

This is a group project. The exchange of ideas regarding the game is encouraged, but sharing of code and reports is forbidden and will be treated as cheating. We will be using document and code comparison tools to verify that the submitted materials are the work of the authors only. Please see the syllabus and www.mcgill.ca/integrity for more information.

7. Contact

Please post on Ed with the Projects label with any questions.