INFO 6205 Spring 2025 Team Project

Monte Carlo Tree Search

Kickoff: March 21st, noon.
Milestone: March 31st, midnight: working version of TicTacToe: *runGame*.
Deadline: April 20th, midnight (heavy penalties for late submission).

General Description

Your team will write a program to implement a Monte Carlo Tree Search for two games:

- Tic-tac-toe.
- A game of your own choosing.

Monte Carlo Tree Search is a very successful technique for game-playing. But it's considerably more complex than it looks at first. I urge you not to do any "premature optimization." Start simple and, once you have the basic idea working, you may consider optimizations, provided that you measure their effects (which, of course, you will detail in your report). Please read about the technique of Monte Carlo Tree Search from the references below.

I have created some skeleton code for you in the *DSAIPG* repository. If you can't find it because you've recently refreshed the repository, talk to the TAs.

The skeleton code is in the following package: *com.phasmidsoftware.dsaipg.projects.mcts*. There are two sub-packages: *core* and *tictactoe*. You will add a third sub-package in your repository that will include code specific to your own game. The skeleton code will all compile, but you will need to code the following:

TicTacToe

If you run the main program, it will run a random game and print the winner (or draw) at the end. You can test this with the unit test called *runGame* (which has a predictable winner—you can run different games by varying the seed).

At first, you will run into some test failures. This is because I want you to go in and provide code where it is marked:

```
// TO BE IMPLEMENTED
```

The purpose of getting you to do that coding is to try and get you to understand the way that the *TicTacToe* game is implemented for the generic code. Please be sure to understand how it is coded before you attempt to implement your own game.

In the *core* sub-package, all the code is generic. You will see a lot of Java "generics" in the code. You can ignore any complaints about *Game* being "raw." Everything specific about a *Game, Move, Node,* or *State* is to be found in the sub-package for the particular game. All the other code (in the *core* sub-package) is truly generic. On the whole, you shouldn't need to make any changes to *Move* or *State* (you are allowed to if you must). But I do anticipate that you will need to change *Game* and *Node*. This is particularly true if you decide to code a multi-player game like *Settlers of Catan*.

MCTS

It's important to realize that I have not implemented the Monte Carlo Tree Search logic. That's your job. All I've done is provide some skeleton code that I hope will be somewhat useful. You are required to stick to the general framework of the solution. But you will be making many changes to the existing code, both in the *core* sub-package and, I expect, in the *tictactoe* sub-package.

I have not coded anything about heuristics that you might use to aid the MCTS process. That's up to you. The tic-tac-toe example is sufficiently small, in terms of the number of possible games, that you won't burn up a lot of time even without heuristics. But, if you decide to play chess or go, you'll need every performance enhancement that you can find.

Requirements

You must provide unit tests for your own game and any unit tests that are required to support changes in the tic-tac-toe game. We will be looking at your coverage.

You must provide details of timings of runs, and a reference to explain the rules of your chosen game.

You must detail the steps required to run your games from git. Preferably, you will use maven in order to build and run the program. A video that shows the program running is encouraged but not required.

You will provide a report that includes an executive summary as well as all of the details, graphs, charts, etc. that you need to support your work. Think of it as a peer-reviewed paper that you are submitting—except that I don't need you to worry too much about formatting of citations and things like that. I *do* expect that all citations will be included, however.

References

Monte Carlo tree search (Wikipedia article)
https://arxiv.org/abs/2103.04931 (review paper)
Monte Carlo Tree Search: Beginners guide