02180 Introduction to AI, SP22
BOARD GAME ASSIGNMENT
Due: 21st March 2022 at 23:59

Introduction

This assignment is to be carried out in **groups of 4 students**. Your solution is to be handed in via DTU Learn. You should hand in three *separate* files:

1. A **pdf file** containing the **report**. See instructions for the report below.
2. A **pdf file** containing a short **group declaration** stating who did what in terms of ideas, implementation and report.
3. A **zip file** containing the source code, including a readme for how to install and run the program.

The goal of the assignment is to implement AI for a board game. In order to do so, you must 1) choose a board game, 2) implement the game on a computer, and 3) implement an AI-based player for it using one of the algorithms presented in the course (or variants, combinations or extensions of these algorithms). The assignment is deliberately relatively open-ended and free, so that you get experience in independently choosing an AI project, investigate the relevant AI methods for it, and finally implementing and benchmarking it.

Choice of game

First you need to choose the board game that you want to create an AI-based player for. You have a more or less completely free choice, except you are not allowed to choose a game that is essentially trivial and has a very small state spaces like tic-tac-toe. We have selected a number of games that we suggest to choose from, but you are allowed to choose others. The list of games we have selected is:

1. **Kalaha** (aka **Kalah** or **Mancala**). 2 players.
2. **Ricochet Robots**. ≥ 2 players. 3 or more players are recommended.
3. **Hanabi**. 3-5 players. 4 players are recommended.
4. **Laser Chess** (aka **Khet Laser Game**). 2 players.

6. **Kulibrat**. 2 players. This game was designed by Thomas Bolander, and hence none of you are likely to know it in advance. The rules are in the file `kulibrat_rules.pdf` on DTU Learn.

7. **Taiji**. 2 players.

8. **Chinese Checkers** (aka **Chinese Chequers**). 2-6 players.

9. **Rush Hour Shift**. 2 players.

10. **2048**. 1 player. This is originally a video game, not a board game. However, it can still easily be turned into a board game. It is described here: https://en.wikipedia.org/wiki/2048_(video_game).

All games have distinguishing features making them require different types and levels of AI and illustrating different AI-aspects. Game AI for all games can be implemented using the theories and algorithms introduced in the course. Hanabi is potentially a bit more challenging, since you have to reason about the partial observability of the other players.

The first thing you have to do in your group is to choose a game. It is recommended to spend some time on this, making sure you choose a game where you have a relatively clear view of how to make AI for it. For the games you consider, try to play them a couple of times first, while reflecting on what kind of AI could work for them. When playing the games, try to think about the following:

1. What kind of reasoning are you as a human using when thinking about the next move to make?

2. Which algorithms from the course could possibly be used to create a computer opponent (an AI) for the game?

3. Can any of the human reasoning and intuition used to play the game be implemented in the AI algorithm?

4. How would you represent game states in the computer? And moves?

Some games are clearly much more complicated than others. If you choose a game for which it is very challenging to create good AI, we will of course expect less in terms of an efficient AI-player than if you choose a much simpler game. Conversely, if you choose one of the games that has simple rules (is easy to implement) and is relatively straightforward to create AI for, we will expect you to be able to produce a much more polished and optimised solution. For any of the games, you are allowed to simplify in terms of making the game board smaller, using fewer pieces or omit certain rules, if that is necessary for you to be able to make the AI work.

Implementation

You have to implement the game and an AI player for it, so that it is possible to play against the computer (except the single-player games where the computer plays alone). You don't have to make any fancy graphical user interface, you can just choose a minimal representation that makes it possible to play against the computer (using e.g. a textual representation of a game

state or whatever you find suitable). You can choose whatever programming language you prefer.

Your AI should be based on search methods, so that it builds on the curriculum of the course. You are allowed to extend the search methods with theories and methods from other courses (e.g. using epistemic models for representing states or neural networks for evaluation functions), but this will not be expected and is only recommended if you have special interests in those techniques. It is your own responsibility to avoid choosing a project which is too ambitious, so that you risk never making it work.

If you prefer, you are allowed to use an existing implementation to show a graphical representation of the game, but only for that. Your AI must be a separate entity that borrows no code from anywhere. The separation of your AI and the game engine must be completely obvious and explicitly stated in the documentation. There can be no doubt about what code you wrote and what code belongs to the existing game implementation. It's not acceptable if your AI code is somehow entangled in the game engine code. So, if you want to use a game implementation to render your game, you have to find one which has a clearly defined API that your AI can use to interact with the engine without having to dig into the engine itself.

Report

You should write a concise 4–6 pages report, addressing the questions and issues below that are most relevant to your chosen game, AI and implementation:

1. Game rules, in particular if you choose a game not on the list above or if you make simplifications to the game. Make this part as short as possible. If you have simplified the rules, explain why and explain what it would take to make AI for the full game.

2. What kind of game is it? Relevant questions are e.g.: Single- or multi-player? Competitive or cooperative? Zero-sum? Is it turn-based, concurrent or real-time? Perfect or imperfect information (full or partial observability)? Deterministic or stochastic? And what does this imply for the kind of algorithms that apply to the game?

3. What, approximately, is the size of the state space of the game, that is, the number of reachable states from an initial state of the game? Finding the precise number can often be very difficult, but here it is enough to estimate whether it is e.g. in the ballpark of 10^5, 10^{10}, 10^{15}, 10^{20}, 10^{30}, 10^{40}, 10^{50}, ... You can e.g. try to give an upper bound on the size of the state space (or both a lower and upper bound). What does this imply for the kind of algorithms that apply to the game?

4. Describe the following elements of your game (or the relevant subset): s_0 (initial state), $\textsc{Player}(s)$, $\textsc{Actions}(s)$, $\textsc{Results}(s, a)$, $\textsc{Terminal-Test}(s)$, $\textsc{Utility}(s, p)$. You don't necessarily have to specify these formally, but try to describe them as clearly and precisely as possible. If your chosen game can not be expressed fully in terms of these functions, explain why, and describe the additional components as precisely as possible.

5. How are states and moves represented in the computer? To make AI for this game, was it enough to represent a game state as the position of the pieces or the distribution of the cards? Or was it e.g. necessary to represent game states as belief states, and why? Don't give too implementation-specific answers, but answer in overall terms of algorithms and data structures.

6. Which of the methods and algorithms considered in the course could potentially be used to construct AI for your game, and which have you chosen? If you use one of the standard algorithms without modification, you don't have to describe it, but can just refer to the textbook (or other sources). However, try to give a small concrete example of how the algorithm works in your game, e.g. by illustrating a few steps of the search graph/tree being built (you can do this in a smaller or simplified version of the game, if you prefer). Also, argue of the appropriateness of your chosen algorithm for the chosen game.

7. If you use heuristics or evaluation functions, make sure to explain how these have been designed. Use mathematical notation rather than implementation-specific notation.

8. Are there any parameters that can be adjusted in your implemented AI? Did you try out several different algorithms, search depths, heuristics or evaluation functions? If so, it is a very good idea to include benchmarks (test results) comparing the different versions. It is also interesting to compare the strength of the AIs to human players (yourselves).

9. Comments on future work: How could your solution be improved? Could it be optimised using the same algorithms and data structures that you are currently using? Or would significant improvements of the AI require different methods, and if so, which could be considered?

The format of the report is quite short, so it is important to choose the most important parts to cover in the report. Don't waste space on explaining things that are already covered in the course, all that can be assumed. The report should be as concise as possible, that is, everything should be expressed as short as possible while still being clear and precise. Being mathematically precise and making use of the technical concepts from the course (and other courses) is often a good way to write explanations that are more clear, precise *and* shorter. But of course it requires more time to write this way.

Assessment

The assignment will be assessed based on the the quality of the report and the implementation, weighted more or less equally. The implementation will mainly be assessed on your choice of algorithms and data structures. In the report, we will assess how well you chose the most important aspects to focus on from the list provided above, and how well and precise everything is described. Try to be as mathematically precise as possible and use the technical concepts from the course whenever appropriate. We we also assess the quality of the AI that you have implemented, in terms of the appropriateness of the chosen technique and how well it works in the implementation.