EECS 4/5740 Artificial Intelligence

Term Project

Game Playing through Adversarial Search

Instructions

This is a team project assignment. Teams can have no more than two members. A team must be formed with members being only undergraduate students or graduate students. Students will need to form their own teams unless help is requested from the course instructor. Individual effort, although not recommended, is permitted and does not carry any extra credit.

Team members are expected to contribute to every aspect of the project equally as the grades will be partly but to a significant degree based on individual contributions.

This project has different requirements for undergraduate students and graduate students.

Introduction

This project aims to give you an opportunity to implement an AI agent using a high-level programming language: the AI agent will employ adversarial search with alpha-beta pruning and heuristics-based evaluation function to play a two-person board game, Conga. Here is the official Conga board game webpage: http://www.congaspiel.de/

Conga Game

The Conga is a two-player game played on a square 4×4 board, as shown in Figure 1.

Player 1

(1,4)	(2,4)	(3,4)	(4,4)
(1,3)	(2,3)	(3,3)	(4,3)
(1,2)	(2,2)	(3,2)	(4,2)
(1,1)	(2,1)	(3,1)	(4,1)

Player 2

Figure 1: A Conga board layout for 2 players.

Initially, Player 1 has ten black stones in (1,4) and Player 2 has ten white stones in (4,1). The players alternate turns. On each turn, a player chooses a square with some of her stones in it, and picks a direction to move them, either

horizontally, vertically or diagonally. The move is done by removing the stones from the square and placing one stone in the following square, two in the next one, and the others into the last one. The stones can only be moved in consecutive squares that are not occupied by the opponent; if a direction has less than three squares not occupied by the opponent in a row, then all remaining stones are placed in the last empty square. If a square has no neighboring squares that are not occupied by the opponent, then the stones in that square cannot be moved.

You can move stones from a square to a (series of) neighboring square(s), provided the squares you are moving to are not occupied by the opponent. It makes no difference if the squares are free or occupied by yourself. You do not need any of the squares you are moving to be free; they could all already be occupied by your other stones. The only distinction is between squares that are occupied by the opponent (which block you) and squares that are not occupied by the opponent (which you can move to). For example, let's say you have several stones in square (1,4) and you want to move them right. There can be four possible cases:

1) The opponent occupies square (2,4). You cannot move right. That is the example in Figure 4.
2) The opponent occupies square (3,4), but square (2,4) is either free or occupied by yourself. You move all the stones from (1,4) to (2,4).
3) The opponent occupies square (4,4), but squares (2,4) and (3,4) are either free or occupied by yourself. You move one stone from (1,4) to (2,4), and all other stones in (3,4). That is the example in Figure 3.
4) The opponent doesn't occupy any squares in that row, and all squares are either free or occupied by yourself. You move one stone from (1,4) to (2,4), two stones to (3,4), and all other stones in (4,4). That is the example in Figure 2.

The goal of the game is to block the opponent, so that (s)he has no legal moves. In other words, all the opponents' stones must be trapped in squares that are all surrounded by the player's stones.

Requirements

For this project, you must design and implement a computer program to play the Conga game. The agent implemented must use the Minimax search algorithm and Alpha-Beta pruning, as well as some evaluation function to limit the search. The adversarial search algorithm should be able to make a move within a reasonable time (aka response time) using a desktop computer using standard specs such as i5 CPU, 8 GB RAM, 256 GB HDD etc. (or equivalent). Since it is unlikely that you will discover an optimal evaluation function on the first try, you will have to consider several evaluation functions and present them in your report.

Graduate student teams will, additionally, need to implement an extension/enhancement/improvement of the basic Alpha-Beta agent for increased winning potential. This will require the student to conduct a literature search to identify one or more extensions or enhancements to implement and evaluate for performance against the basic Alpha-Beta agent.

Your implementation should output information about the search, including the search ply/depth, the number of nodes explored, the number of nodes pruned, and the listing of times it took to make a move by the AI agent(s).

In order to assess the performance, you will also need to implement a Random agent that plays the Conga game, or an agent that always plays through a random but legal move.

You may reuse a third party code-level implementation of Minimax, Alpha-Beta (without Conga specific heuristics or evaluation function) or its extensions. However the code which you re-use or borrow, which you must also cite in your report and source file(s), cannot be specialized for the game of Conga: Conga specific heuristics or evaluation function by others are NOT admissible. You are required to write the code for the game of Conga using one of the two high-level programming languages, namely Java or C++. Any other high-level language implementation or Python, MATLAB, or any other scripting language based implementation is not permitted.

As the focus of this project is not necessarily all aspects of software development, you may use existing code libraries for input/output, graphics etc. The human interface can be text based but with appropriate organization for easy interpretation. Again, you must clearly cite any such code written by others in your report. You must give proper credit for code written by others in your program using source file or function/procedure/method headers.

> Warning: Please be advised that using the code by others for any other functionality beyond what is explicitly permitted in this document would be a violation of academic misconduct policy and may result in a grade of F(ail) for the course.

Project Deliverables

You are expected to turn in a written report in a single PDF file which should include the following:

1) Cover page with Project Title, Team Members, Date of Submission and Pledge: use the same pledge for assignments and both team members must separately sign.
2) A description of your entire program for AI agent for Conga using appropriate set of software engineering documentation tools such as UML diagrams.
    a) (This section to be included only if AI agent is not fully functional) Explain the functional and non-functional parts of your program in detail.
3) A description of your implementation of the AI game playing agent using Minimax and Alpha-Beta pruning using appropriate software documentation aids.
    a) Graduate teams will also need to discuss their enhancement(s) along with associated literature survey.
4) Discussion of different evaluation functions and heuristics implemented. This section needs to conclude with your choice of the "best" evaluation function preceded with the rationale for choosing it. Note that in most cases empirical assessment and evaluation of performance of evaluation functions through simulation (aka game playing against each other) may be needed to identify the best performer.
5) Performance evaluation of your AI agent:
    a) Report the outcome of at least 10 games played against the "Random agent" using a student's t-distribution model: use the best evaluation function for this.
        I. Undergraduate students will present for the AI agent vs. Random agent.
        II. Graduate students will need to present for both AI agent (Alpha-Beta) vs. Random agent as well as AI agent (Alpha-Beta) vs. AI agent (Alpha-Beta with enhancement)
    b) Report the time complexity (in terms of total number of nodes visited), amount of pruning and real-time response times. Use the best evaluation function for this. Present the specifications of the hardware platform and the OS on which these measurements are done.
        I. Undergraduate students will present for the AI agent.
        II. Graduate students will need to present for both AI agent (Alpha-Beta) and AI agent (Alpha-Beta with enhancement)

6) Sample output of your agent using the "best" (according to your previous analysis) evaluation function you implemented, playing against the Random agent. Explain and/or justify key moves your agent makes.
7) Detailed contributions by each team member for software development at the level of function/procedure/method, testing & debugging, verification & validation, and report authoring.
8) (*Graduate students only*) MLA/IEEE/etc. style references for Alpha-Beta enhancement(s).
9) Appendix A - Copy of your source code: use landscape mode with small font size to minimize lines wrapping around the end of the page. You must use proper naming, layout, indentation, commenting, file/function header etc. for documenting your code.
10) Appendix B - Link to YouTube video that shows the entire game episode of your AI agent (aka program) playing against the Random agent using the best evaluation function.
    a) Graduate students will need to also present this for the AI agent vs. Enhanced AI agent case.
11) Appendix C - Link to Windows OS "executable" code on UToledo OneDrive with a brief user's manual (in PDF) on how to run your code.
    a) Make sure the OneDrive folder with your code is accessible for download.
    b) Make any input files available in the same folder.
    c) Mention the IDE name & version which you used for development.

Project Evaluation

The project is worth 100 points. Your grade in this project will be determined as follows. Late projects may be denied credit in accordance with the late submission policy in effect for assignments.

Grading Scheme:

1. Implementation of the search technique:
   - Minimax algorithm (10 points)
   - Alpha-beta pruning (10 points)
     - Enhancements to basic MiniMax/Alpha-Beta algorithm (Graduate Student Teams only)
   - Evaluation functions (30 points)
2. Performance evaluation:
   - Game Winning Record (10 points)
   - Efficiency and complexity: (10 points)
     - time complexity: total number of nodes explored during the search,
     - max game tree depth of the search, and number of nodes pruned
   - Real-Time requirement: Agent is fast enough to be considered "interactive" with a human player. (10 points)
3. Written report (20 points): professionalism (layout, organization, presentation etc.), and completeness (documents all aspects of project work).

Please note that unless you present in sufficient detail in your report to make it possible to evaluate your work, credit cannot be awarded for any grading criteria listed above.

Please also note that lack of individual contributions may adversely and significantly affect your grade which may end up being in the bottom quartile for percentage (as an example) if significant and meaningful contributions for many aspects of this project is not reported for a specific team member.

Notes on the game:

While researching the game of Conga, you may realize that there is a second victory condition. A player can win by making a line four squares long containing the same number of stones. This victory condition has been removed from the game for this project, in order to simplify the evaluation function.

A consequence of this simplification is that the game is now much harder to win. In fact, two players of equal strength can play for hours without being able to end the game. However, it is still possible for a player to defeat a player of lesser skill, or for an agent to defeat another agent with a shallower search and a less accurate evaluation function. And it is definitely possible for a rational agent to defeat the Random Agent, in as little as 30 moves.

Given the requirement of making your agent play against the Random Agent, you may come up with the idea of somehow optimizing their agent to play better against that opponent specifically, by exploiting its random, non-optimal play. That is not acceptable for this project. The goal of the project is to design an agent that can play rationally against any opponent, not one that is optimized to play only against the Random Agent.

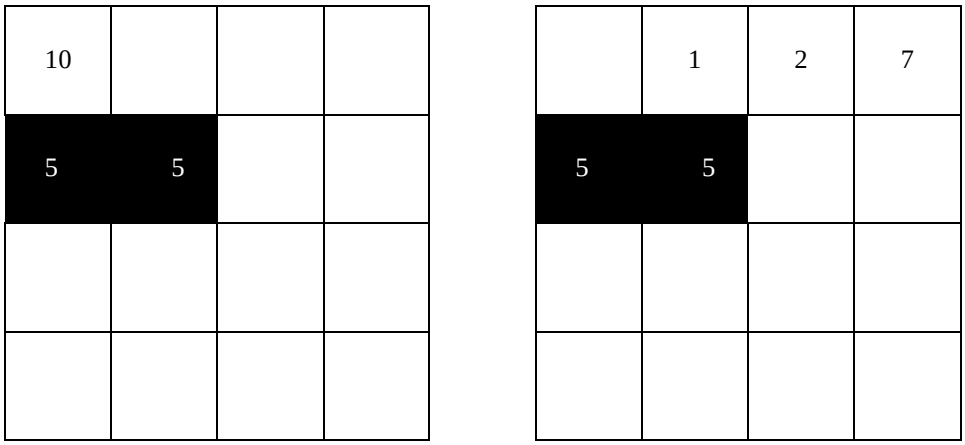

Figure 2: From the setup of the left board, white has only one legal move. The result of that move is shown in the right board.

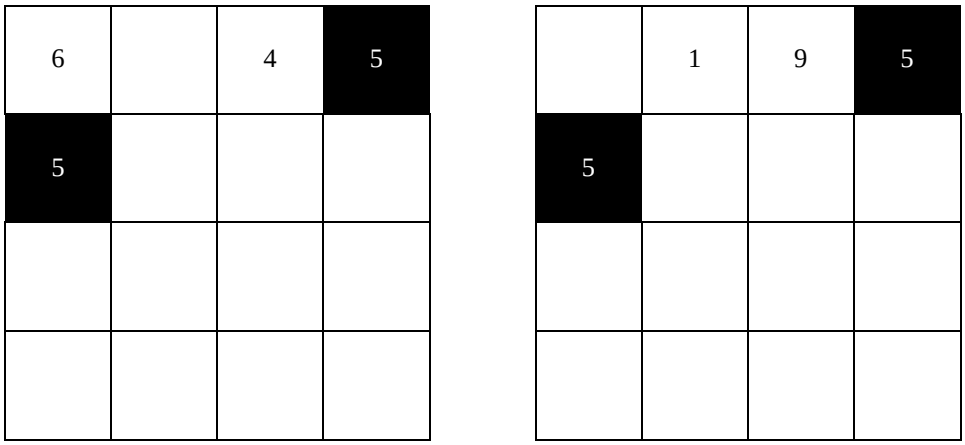

Figure 3: From the setup of the left board, white has six legal moves, and decides to move (1,4) to the right. The result of that move is shown in the right board.

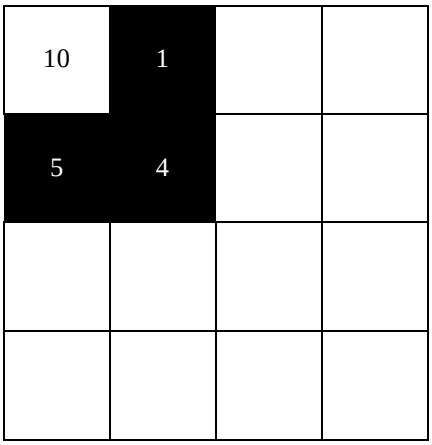

Figure 4: In the setup of this board, white has no legal moves. Black has won.

Credits: This project is based on the original at http://pami.uwaterloo.ca/~basir/ECE457/project2.pdf

Revision Date: 28 September 2020