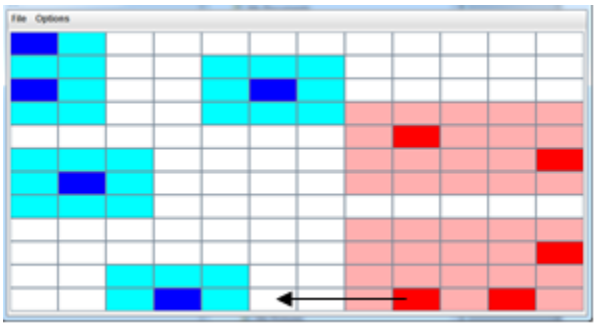
UNIVERSITY OF PRETORIA
DEPARTMENT OF COMPUTER SCIENCE

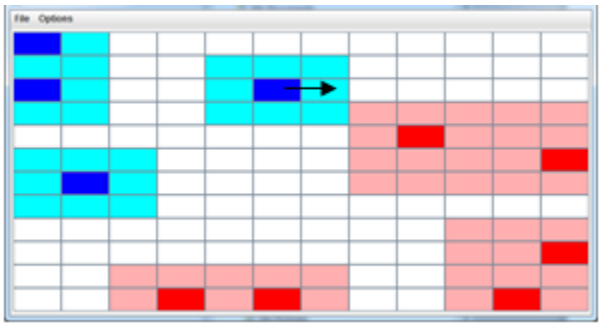
COS314: Artificial Intelligence
Project 1: Game Trees (Cell Wars)

Project 1 is a pair-project. Find a partner to work with and submit one project for both of you. You are permitted to work on your own if you prefer.

Rules of the game

The game is played on a grid of N × N blocks by two players, where N is an even number > 8. Each block can be empty (in white), occupied by a cell (dark colour), or covered by a cell (light colour). Each player is allocated a colour and the cells owned by that player are displayed in their colour. The region around a cell is 'covered' by that cell and the blocks that are covered are displayed in a lighter shade than the colour of the cell. A cell covers the 8 blocks forming a rectangle around the cell. If the covered blocks of a cell touch any block covered by one of its own cells, then the area covered extends to the rectangle including both covered areas. The example screen below on the left shows a scenario where the blue player has five cells in four separate rectangles of covered blocks, while the red player has five cells covering only two rectangles of blocks. Notice that red in the left picture has more covered blocks in total than blue.





Note that it is possible for a block to be covered by more than one cell. In this case the block is covered by both cells.

Moving: A player takes a turn by first selecting one of his/her cells. The player then decides where the cell should move to. A cell can only move up, down, left or right (not diagonally) by the number of blocks proportional to the cells in the rectangle of covered blocks surrounding the cell. For example, the red cells in the bottom right covered rectangle can each move by 1, 2, or 3 blocks, because there are 3 cells in the covered rectangle. In contrast, the blue cell on the bottom edge of the grid is only able to move by 1 block. A cell cannot move onto another cell (of its own colour or of the opponent's colour).

Capturing cells: If a cell of player A moves so that the covered block surrounding the cell shares an edge with a cell of player B, then player B's cell is taken over by player A and it changes colour. This is shown with the black arrow in the left screen above, where the red player moves

its cell by 3 blocks. This results in the red region around the red cell sharing an edge with the blue cell, which is captured (as shown in the screen on the right). Notice how the covered blocks adjust to the new positions of the cells. Note that if a covered block touches a cell only on the corner, then it is not captured. For example, if the blue cell in the screen on the right moves one block as indicated by the arrow, then the red cell will not be captured (although the covered regions will be overlapping).

If a cell captures an opponent's cell, then the captured cell can immediately capture any other cells in the same move.

Initial state: Each player has a set number of initial cells (say 5). The grid is divided into a left and a right half. The cells of player 1 are randomly placed in the left half of the board and the cells of player 2 are placed randomly in the right half of the board. The two middle columns of blocks in the grid do not contain any cells in the initial setup.

Winning / Losing: The game ends when the cells of one of the players become extinct. The other player is then the winner.

Implementation

You are required to implement the game along with the AI to play the game. This involves the following:

- You need to develop at least one heuristic for evaluating nodes of the game tree. Extra marks will be awarded for implemented multiple heuristics.
- You are required to implement the Minimax algorithm with and without alpha-beta pruning.
- You may implement the project in either Java or C++. If you want to use a different language, then you are required to clear it with the lecturer before starting the implementation.
- All source files must contain both project members' names and student numbers in a comment at the top of the file.

GUI

The user interface for the game should have the following features:

- The user should be able to set the size of the grid (an even number > 8) and the number of initial cells.
- The user should be able to set the players of the game (human vs human, human vs AI or AI vs AI). For the AI players, the user should be able to set the ply depth and whether or not alpha-beta pruning should be used.
- A move made by a player should appear in such a way that a user can tell which cell was selected for moving.
- A textual user interface can be used but be aware that this will award you less marks than a GUI.

Notes on Design

The design of your code will be evaluated. The general structure of your code contributes to the design, not only design patterns. An example of where one would gain marks in design is if one uses the Strategy design pattern for the AI algorithms as opposed to big `if` statements.

Have a look at the Model View Controller meta pattern, it offers a way of 'untangling' the GUI, players, and game logic from each other. You are free to choose. Keep in mind that a good design will not only award you more marks but will make your project easier to manage.

Marking guide

Your project mark will be determined using the following table as a guide. Note that 45% of your mark will be awarded at the demonstration. If you cannot run your program or are not present at the demonstration you will get 0 for this part of the mark. The remaining 55% is based on your electronic hand-in. Your code will be inspected for logic and coding style.

	Maximum mark
Demonstration	
• Running of the game: you will be given a mark based on how well your program runs as specified.	30
• User Interface	15
Electronic Hand-in	
• Game logic	10
• Game state	10
• Search algorithms & heuristic(s)	20
• Coding style and internal documentation	15
Total	**100**

Hand-in details

Electronic submission due date: Thursday 19 March, 14h00.
Demos: Thursday 19 March and Thursday 26 March (book your 10 min slot on the COS314 website). Both project partners must be present at the demonstration.

- Hand in all your files as a single compressed file on the COS314 website.
- You will be required to demonstrate your uploaded program in your booked slot in the SIT1 Lab. Both project partners must to present at the demonstration to answer questions.
- It is your responsibility to ensure that your program runs on the computers in the SIT1 Lab.
- Make sure you are there at least 30 minutes before the demonstration time. We will not wait for you if you are not ready at the time of the demonstration.
- In addition to the demonstration, you must submit your files electronically by the due date/time. Hand in only one copy of your project on one project member's account.
- Very important: make sure that the names and student numbers of both partners appear in all source files as a comment at the top of the files.