

Minimax and Alpha-Beta search in game playing

Saumya Singh - 12 - 070905022

Amitabh Anand - 4 - 070905004

Ankur Sinha - 23 - 070905042

Section A

Department of Computer Science & Engineering

Manipal Institute of technology

August 2010

1 Summary

We use Minimax and Alpha-Beta pruning in a simulated combat game. The project also includes creating an elementary GUI using Curses. We will create two dueling AI's which stomp on little cities. AI one will use Minimax and AI two will use Alpha-Beta pruning.

2 Our Problem Statement / Input - Output

Here is an example question that we'll solve to implement Minimax and Alpha-Beta pruning (taken from project assignments at the University of California) :

The George W. Bush Institute for Modern Warfare and Record Deficits has hired you to implement a war game for their new 5 billion dollar war simulation Computer, the W.O.P.R. Your job is to create two different artificial agents to play a game that simulates a combat situation. Once you are finished, the DoD will classify your research and pull you from the project leaving you a fractured and bitter human being. However, as you do not know this yet, you are full of life and zeal for the project. Remember, todays enemy is tomorrows vapor, or so it says in the entry way into the institute. **The game is a simulation of ground warfare and has the following rules:**

1. The game board is a 5x5 grid representing a city your forces will trample.
2. Each player takes turns like in chess or tic-tac-toe. That is, player A takes a move, then player B, then back to player A and so forth.
3. Each square has a fixed point value between 1 and 99 based upon its computed strategic and resource value.
4. The object of the game is to be the player in the end with the most points derived by adding the point value of squares in their possession. Thus, one wants to capture the squares worth the most points.
5. The game ends when all the squares are occupied by all players since no more moves are left.
6. Movement is always vertical and horizontal but never diagonal.
7. Units can be conquered in the vertical and horizontal direction, but never the diagonal direction.
8. The values of the squares can be changed for each game, but remain constant within a game.
9. On each turn, a player can make one of two moves:

- (a) **Commando Para drop** You can take any open space on the board with a Para Drop. This will create a new piece on the board. This move can be made as many times as one wants to during the game, but only once per turn. However, a Commando Para Drop cannot conquer any pieces. It simply allows one to arbitrarily place a piece anywhere unoccupied on the board. Once you have done a Para Drop, your turn is complete.

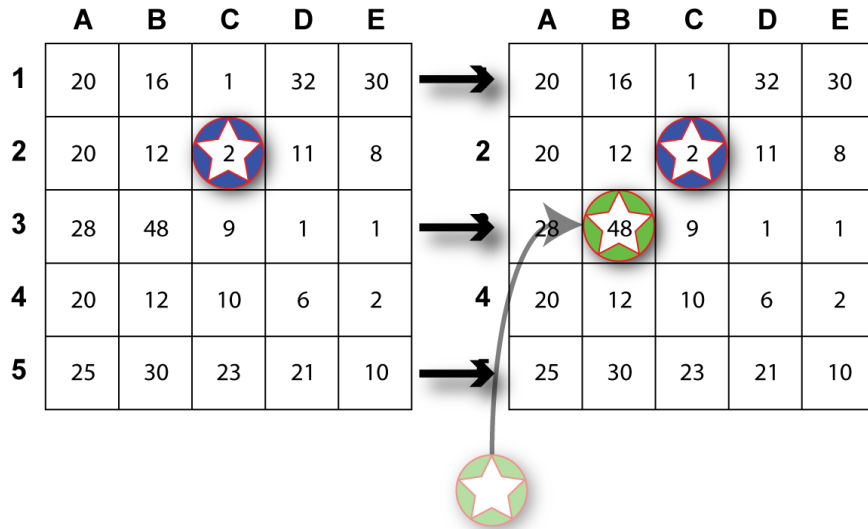


Figure 1: This is a Commando Para Drop. In this case, green drops a new piece on square [B,3]. This square is worth 48, which is a higher number, meaning that it contains some juicy oil wells or other important resources. After that, the score is green 48 : blue 2. A Commando Para Drop could have been carried out on any squares except for [C,2] since blue already occupies it.

- (b) **M1 Death Blitz** From any space you occupy on the board, you can take the one next to it (up, down, left, right, but not diagonally) if it is unoccupied. The space you originally held is still occupied. Thus, you get to create a new piece in the blitzed square. Any enemy touching the square you have taken is conquered and that square is turned to your side (you turn its piece to your side). An M1 Death Blitz can be done even if it will not conquer another piece. Once you have made this move, your turn is over.














	A	B	C	D	E			A	B	C	D	E
1	20	16			30	→		20	16			30
2	20	12			8		2	20	12			8
3	28	48	9	1	1	→		28	48	9		1
4	20	12			2		4	20	12			2
5	25	30	23	21	10	→		25	30	23	21	10

Figure 2: This is an M1 Death Blitz. Green blitzes the piece in [D,4] to [D,3]. This conquers the blue piece in [D,2] since it is touching the new green piece in [D,3]. A blitz always creates a new piece and always moves one square, but it does not conquer another piece unless it is touching it. Thus, another valid move might have been for [D,4] to have blitzed [E,4]. Then the green player would own [D,4] and [E,4] but would have conquered none of blues pieces. Note, the score before the blitz was green 16 : blue 54 but afterwards is green 28 : blue 43.

















	A	B	C	D	E		A	B	C	D	E
1	20	16			30	→	20	16			30
2	20	12				→	20	12			
3	28	48	9		1	→	28	48	9		1
4	20	12			2	→	20	12			2
5	25	30	23	21	10	→	25	30	23	21	10

Figure 3: Here blue blitzes [C,3] from [C,2]. In the process greens pieces at [D,3] and [D,4] are conquered since they touch [C,3]. Notice that in its next move, green will not be able to conquer any of blues pieces and only the piece at [D,4] would be able to execute an M1 Death Blitz since [D,2] has no neighboring unoccupied squares.

3 Modules : Steps we'll take

1. We create two AI's to play the game. The first AI will use the Minimax algorithm to play the game and the second will use Alpha-Beta Pruning.
2. We also create a simple "GUI" using Curses to show the state of the game and allow a human to play the AI.
3. Our GUI will show:
 - (a) The game board with values for each square.
 - (b) Which player owns which square.
 - (c) Note that the purpose of the GUI is twofold. It should allow one to observe the AIs as they play each other as well as allow a human to play the AI.
4. We will read in a new board and the AIs will play each other. This way we can play with boards representing different cities our vicious AIs will ruthlessly smite (See below).
 - (a) The board will be a delimited text file.
 - (b) We only instantiate a Minimax v. Alpha-Beta Pruning game.
 - (c) **(Optional, if time permits)** We will allow a human to play one of the AIs. We can also allow a human to play another human.
5. We will give a summary at the end of the game for each AI.

- (a) We will give its score and whether it won or lost.
- (b) We will print the total number of nodes examined by each AI for the whole game (using a long integer).
- (c) We print the average time to complete each turn. Thus, we will compute the mean value for player A and one for player B.

4 Number of Weeks required : module wise breakup

1. AI Player One : 5 weeks
2. AI Player Two : 5 weeks
3. GUI : 2 weeks

5 Hardware and Software requirements

Linux platform, GNU C/C++, GNU ncurses library, other development tools available for Linux.