Assignment 4 – Adversarial search with learning

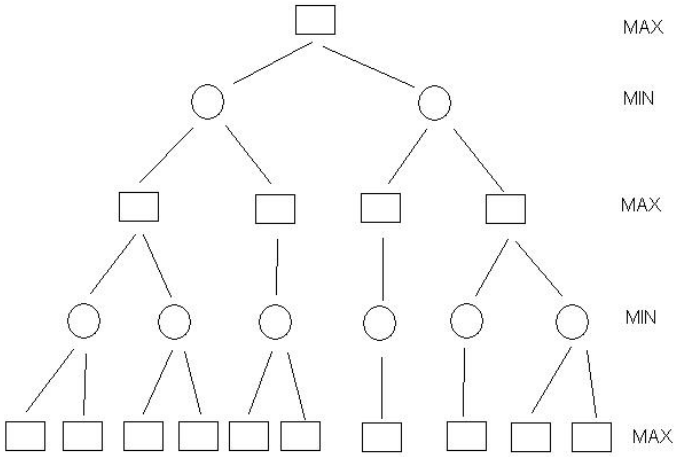
Submit:
- Your group report (within a zip file) to http://www.deei.fct.ualg.pt/IA/Entregas/

  Up to April 30, 2018

Adversarial search

1. Consider a null sum game, of two agents, playing in turn, MIN and MAX (which plays first). The game starts with n pieces. In his turn, a player can take 1, 2 or 3 pieces. The winner being the player who can get the last piece. Using a minimax strategy show that when n = 5, MAX wins the game regardless of what MIN does.

2. Consider the following game tree, where the square nodes maximize and the circles minimize, the minimax algorithm with alpha-beta cuts and ply 2.



Describe how this algorithm works while searching for the first two decisions of the maximizing player. Arbitrate whatever you find appropriate.

Adversarial search with learning for Checkers

As we know from Problem 1, Samuel's Checkers program was the first Machine Learning application [1]. In this assignment we are going to develop a very similar application.

Task

Your task is to develop and heuristic evaluation function with learning to the checkers game. In particular:

1. The development is to be based in the Ashish Prasad's checkers code available from https://github.com/AshishPrasad/Checkers

2. The heuristic evaluation function $\hat{V}(b)$ should be a linear combination of board features such as the number of black or white tiles in the board *b*. See [1][2] for further examples. However, the exact number and type of features is free.

3. The learning method attempts to minimize the following measure of error (*loss function*) *E*:

$$E = \frac{\sum_{b \in B} [V_{train}(b) - \hat{V}(b)]^2}{|B|}$$

   where *b* is a board example from the training set *B* and $V_{train}(b)$ is the target utility value of *b*. The specific optimization method to be used is left as a design choice.

4. Collect a training set *B* with a set of examples of the form (b, $V_{train}(b)$) with $V_{train}(b) = \hat{V}(successor(b))$ where *successor (b)* is the next board state for which it is again the agent's turn to play [2]. Examples:

$1 - \langle 0,0,0,15,0 \rangle$	$\hat{V}_1 = -36.7$	$V_{train}(1) = \hat{V}_3 = -54.1$
$2 - \langle 0,0,0,10,0 \rangle$	$\hat{V}_2 = -24.2$	$V_{train}(2) = \hat{V}_4 = -31.7$
$3 - \langle 0,0,0,22,0 \rangle$	$\hat{V}_3 = -54.1$	$V_{train}(3) = \hat{V}_5 = -803.4$
...		...
$127 - \langle 2330,0,0,27,0 \rangle$		$V_{train}(127) = -1000$

5. Compare the obtained results with those obtained with i) your reactive controller and ii) the original robot player available in the Prasad's code.

References

[1] Samuel, Arthur L (1959) "Some Studies in Machine Learning Using the Game of Checkers", *IBM Journal of Research and Development*, vol. 3, 1959, pp. 201-229

[2] Tom Mitchell, *Machine Learning*, McGraw-Hill, 1997 – chapter 1

Game on

Each group is invited to take part in a competition which will take place in the lab class right after the deadline. In the competition each group's agent will run three times. The competition is won by the group with the highest average score taken over the three runs. Should a draw occurs, the total duration time of the runs will be used as performance measure. The lower the better.

Bonus for the lab class grades will be granted as follow:

Place	1st	2nd	3rd	4th	5th	6th	7th	8th
Points	20	16	12	10	8	6	4	2

Rules

1 – The competition has two stage;
2 – All groups are invited to the first stage of the competition, under the following conditions:

- 3 (three) runs; each one of them not exceeding 2 minutes;
- After 2 minutes the game stops with a draw.

3 – A group ranking will be produced based on the score of the three simulations.
4 – Only the three first ranked groups will have the chance to participate in the second phase (finals), which will take place under to following conditions:

- The finalist agents will play against each other for 3 (three) runs; each one of them not exceeding 2 minutes;
- After 2 minutes the game stops with a draw.

5 – The competition rank will be that of the finals followed by the group rank obtained in the first stage.
6 – No code changes are allowed during the competition. Code should be included within a single file named 'LearningAgent.java' and sent to the instructor before the competition.
8 – Unspecified events will be judged by the instructor;
9 – Appeals, complaints or grievance claims are not accepted.