Report LINFO1361: Assignment 2

Group N°1 (Moodle), 39 (INGInious) (Moodle and Inginious)

Student1: Victor Carballes Cordoba (NOMA : 34472100)

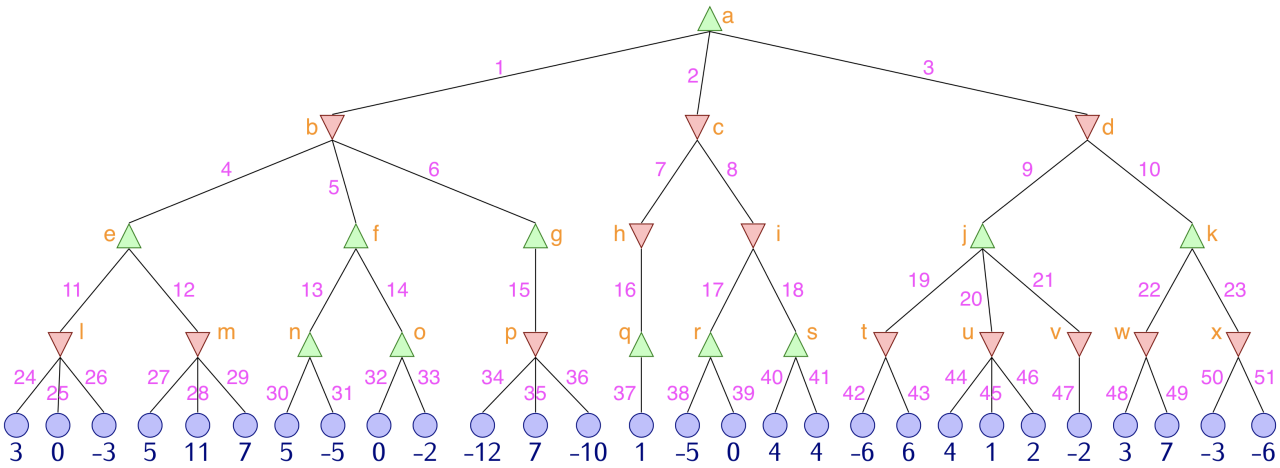
Student2: Krystian Targonski (NOMA : 42942000)

April 7, 2024

Answer to the questions by adding text in the boxes. You may answer in either **French or English**. Do not modify anything else in the template. The size of the boxes indicate the place you **can** use, but you **need not** use all of it (it is not an indication of any expected length of answer). **Be as concise as possible! A good short answer is better than a lot of nonsense!**

1 Exercises (5 pts)

The following figure assigns a unique letter to each node, and a unique number to each branch. Use it to answer the following questions.



1. Perform the MiniMax algorithm on the following tree, i.e. put a value to each node. What move should the root player do? **(1 pt)**

   Assign a numerical value to each node, and indicate the move (i.e. 1, 2, or 3) to perform:

a: 1	**f: 5**	**k: 3**	**p: -12**	**u: 1**
b: -12	**g: -12**	**l: -3**	**q: 1**	**v: -2**
c: 0	**h: 1**	**m: 5**	**r: 0**	**w: 3**
d: 1	**i: 0**	**n: 5**	**s: 4**	**x: -6**
e: 5	**j: 1**	**o: 0**	**t: -6**	**Move: 3**

2. Perform the Alpha-Beta algorithm on the same tree. At each non terminal node, put the successive values of α and β. Cross out the arcs reaching non visited nodes. Assume a left-to-right node expansion. **(1 pt)**

   Indicate the successive α and β values of each node in the table below. Separate successive values by a comma (,). Indicate at the bottom the identifiers of the branches that are cut (in increasing order, separated by a comma) (indicate only the branches where the cuts happen, i.e. don't *indicate the branches that are below a cut).*

Node	α values	β values
a	-inf,-12,0,1	inf,1
b	-inf,-12	inf,5,-12
c	-inf,0	inf,1,0
d	-inf,1	inf,1
e	-inf,-3,5	inf,5
f	-inf,5	inf
g	-inf,-12	inf,-12
h	-inf,1	inf,1
i	-inf,0	inf,0
j	-inf,-6,1	inf,1
k	-inf,3	inf
l	-inf,-3	3,0,-3
m	-inf,5	5
n	5	inf,5
o	/	/
p	-inf,12	-12
q	1	1
r	-5,0	inf,0
s	4	inf,4
t	-inf,-6	-6
u	-inf,1	4,1
v	-2	-2
w	-inf,3	3
x	/	/

Cuts: 14, 51

3. Do the same, assuming a right-to-left node expansion instead. **(1 pt)**

Node	α values	β values
a	-inf,1	inf
b	-inf	inf,-12
c	-inf	inf,0
d	-inf,1	inf,3,1
e	/	/
f	/	/
g	-inf,-12	inf,-12
h	/	/
i	-inf,0	inf,4,0
j	-inf,-2,1	inf,1
k	-inf,-6,3	inf,3
l	/	/
m	/	/
n	/	/
o	/	/
p	-inf,-12	-10,-12
q	/	/
r	0	inf,0
s	4	inf,4
t	-inf,-6	6,-6
u	-inf,1	2,1
v	-2	-2
w	-inf,3	7,3
x	-inf,-6	-6

Cuts: 4, 5, 7

4. Is there a node ordering that can lead to a more optimal pruning of the tree (in the sense where the algorithm prunes more branches than in the two other considered cases)? If no, explain why. If yes, give a new node ordering and the resulting new pruning. **(1 pt)**

   Insert an image below containing the reordered tree, with successive α/β values indicated next to each node, and where the branches that are cut by the algorithm are crossed out. This may either be an edited version of `minimax_empty.png` *(using paint, gimp, etc.), a photograph of a drawing you made by hand, etc. In any case, the image must be* ***clear*** *in order to be graded.*

Yes there is, we can prune a bit more by ordering the nodes in this way and having a right-to-left node expansion

5. How does Alpha-Beta need to be modified for games with more than two players? **(1 pt)**

pour minmax utility -> vecteur de plusieurs valeurs une fois qu'on a le vecteur -> chaque joueur à son tour prends l'action la plus efficace pour lui

2 Shobu (35 pts)

2.1 Alpha-Beta agent (4 points, to be submitted on Inginious)

2.2 Monte-Carlo Tree Search agent (5 points, to be submitted on Inginious)

2.3 Warm-up questions (3 points)

1. What is the branching factor at the start of the game? What is the mean empirical branching factor? (The branching factor is considered here as the number of possible moves that a player can do from a given state). **(1 point)**

First off, we would like to mention that we counted an "action" as a single piece moving, in other words, passive and active moves are both counted ! In the case where no one has moved yet and our definition of an action, the branching factor is 18*6 = 108.

2. What would be one (of the many) drawbacks of the simple heuristic that is imposed for the basic alpha-beta agent above? **(1 point)**

Whenever the agent can clearly take an opponent's piece out, it does not always do it if the heuristic is kept simple

3. Considering the Monte-Carlo Tree Search algorithm, what is the hypothesis supporting the use of random simulation to estimate the win rate? Is this hypothesis always valid? **(1 point)**

the Monte Carlo Hypothesis suggests that even though individual simulations are purely random and may not accurately reflect optimal or realistic gameplay, their aggregated results provide valuable statistical information about the relative merits of different moves in a given game state. This hypothesis underpins the effectiveness of Monte Carlo Tree Search in decision-making processes, particularly in games and domains where exhaustive search is impractical or infeasible. NO. Complexity of the Game: In games or decision-making problems with extremely high branching factors or complex state spaces, the Monte Carlo Hypothesis may break down. The sheer number of possible outcomes and interactions may make it infeasible to obtain accurate estimates through random sampling alone. (our case here) Strategic Depth: In games that require deep strategic thinking and planning, purely random simulations may not capture the nuances of optimal gameplay. Players may employ sophisticated strategies that are not reflected in the results of random simulations, leading to inaccurate win rate estimates.

2.4 Description of your agent (8 points)

Describe in the boxes below what you have implemented for your contest agent. You can also mention things that you tried and did not work but focus on what you have submitted in the end.

Talk abt first : - We thought about going with AB since this is what gave us the best results from the get go, MTCS seemed like too much of a bother g Next : We absly did not ctrl C V the AB agent, first we looked at simple imporvements like "hey this heuristic isn't great, let's change it" which lead us to find find thru trial and error the one we got Next : A way to further imporve the AB algorithm was to make improvements to how it ran as it would allow us to go to further dephts than the classic version would hence having a more broad understanding of where its own actions will lead it Next : This was done thu a binary map, we thought about a bool map but it didn't meet our expectations in performance improvements Next : Something that we also wanted to improve at all costs was the case where our agent would find itself looping thru the same actions over and over again. to do this we added a way to track already explored scenarios which prevented this behaviour Next : using the previous point, we also took the liberty to stop simulating if we ended up on a state already known ! we just could take back the information we stored about it ! Next : given the fact that the order in which the agent tries different actions on a given state play a big role, we also implemented a way to sort those in a way that would overall maximise the performance of the model which lead us to further increase the depth at which we could simulate "interesting cases"

2.5 Comparison of agents (5 points)

Describe here the comparison of the different agents: random, Alpha-Beta, MCTS and your agent (and others if you want!). Remember that it should be a statistical comparison. Describe how you compare the agents. Draw some observations and conclusions based on the results you have obtained.

2.6 Contest (10 points, to be submitted on Inginious)