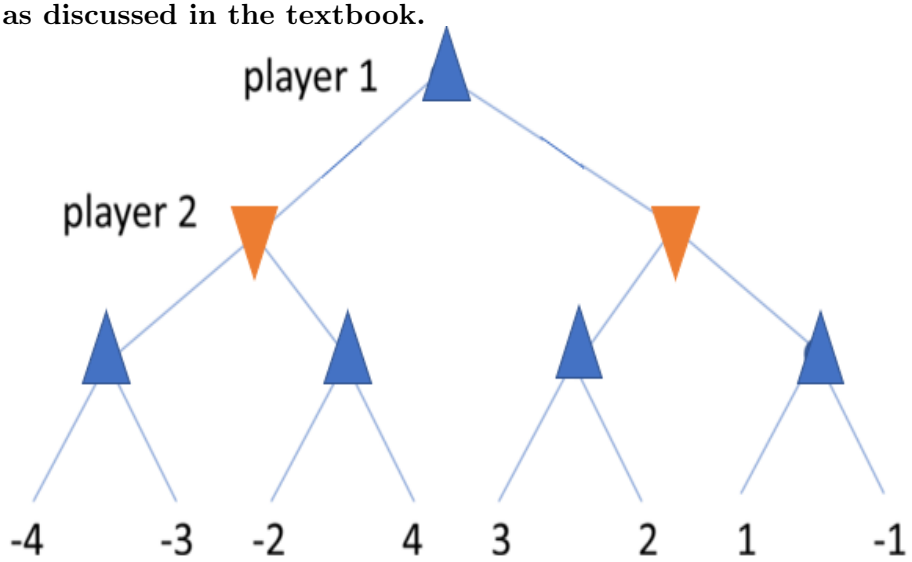
Homework 1
Daniel Ortiz-Chaves, 128009829

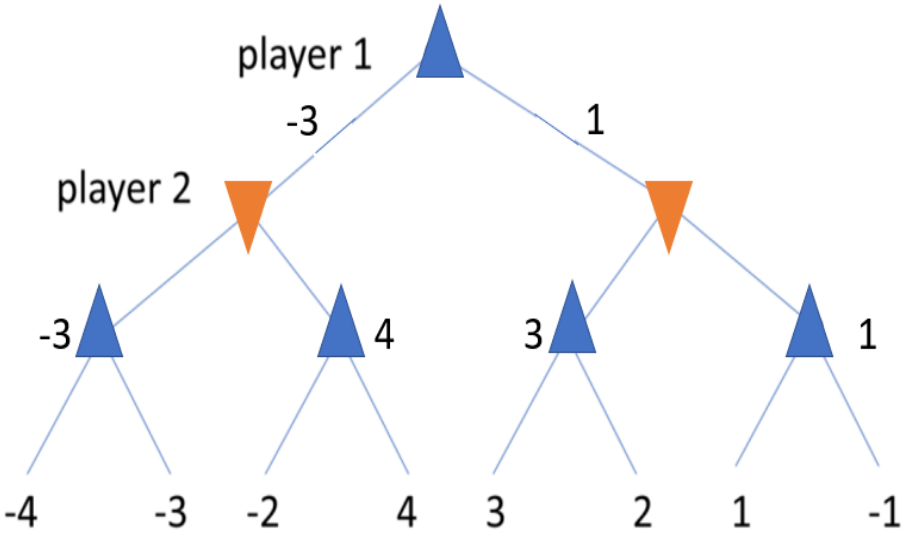
CSCE 420 Spring 2023

Problem 1.

Given the simple game tree (binary, depth 3) below, label the nodes with up or down arrows, as discussed in the textbook.



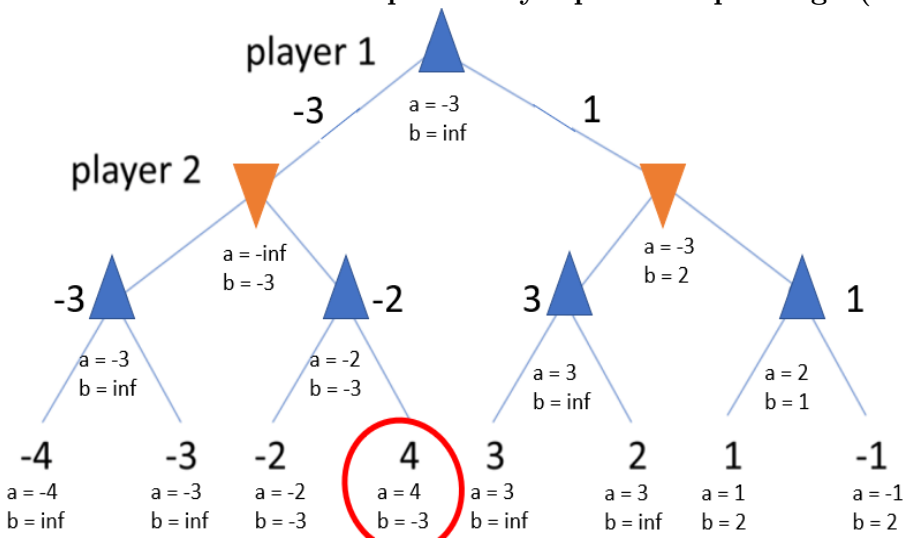
Compute the minimax values at the internal nodes (write the values next each node).



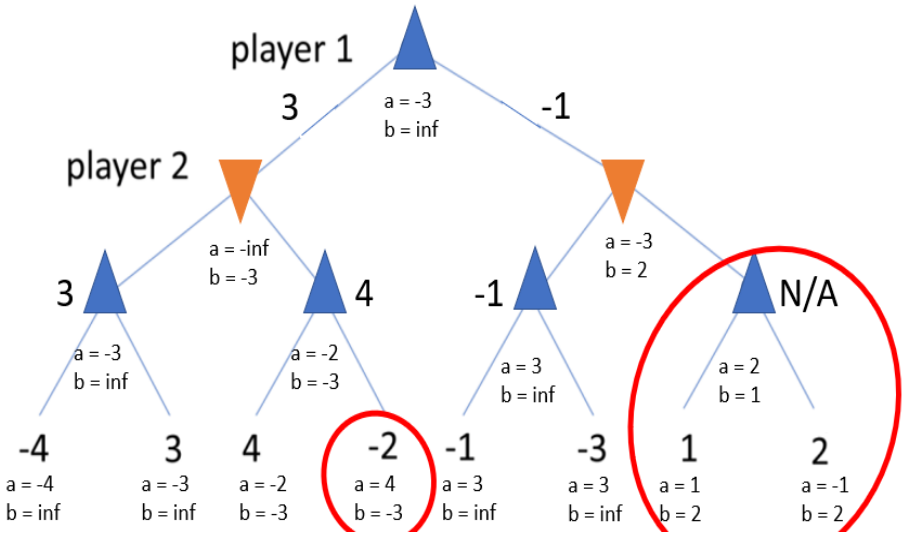
Should the player 1 take action A or B at the root?
Based on these computed utility values, I believe that player 1 should take action B as between A & B, B has the highest utlitiy value as the action player 2 will most likely be the -1 subtree. That action will allow player 1 to obtain a win.

What is the expected outcome (payoff at the end of the game)?
Based on these computed utility values, I believe that the expected payoff will be 1.

Which branches would be pruned by alpha-beta pruning? (circle them)

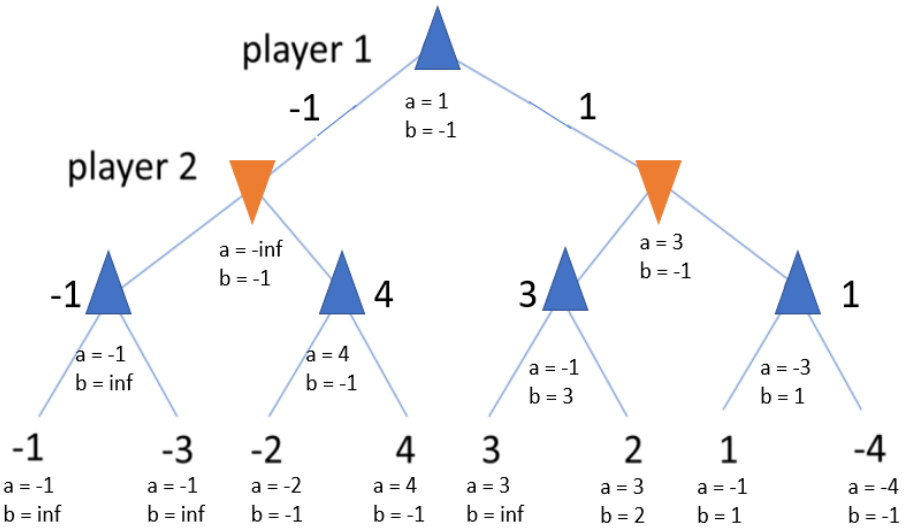


How could the leaves be relabeled to maximize the number of nodes pruned? (you can move the utilities around arbitrarily to other leaves, but you still have to use -4,-3,-2,-1,+1,+2,+3,+4)



I achieved 4 nodes pruned, by ordering the leaves [left to right] (-4,3,4,-2,-1,-3,1,2)

How could the leaves be relabeled to eliminate pruning?



Essentially, switch the -4 and -1 leaf nodes (on the far left and far right) and pruning is eliminated.

Problem 2.

- **Could the player at the root force a win?**

  From the trials I have run, I do not believe the player at the root can force a win. Due to the two turns each players take. The additional min layer, negates any possibility for the 1000 win state to progress up the tree, and the preceding max layer prevents a -1000 win state to progress up the tree. Thus, I believe that in this configuration, then the only possible outcome is a draw (0).

- **Does it matter where the 2 non-zero states are located in the tree? (e.g. adjacent or far apart)**

  From those trials I ran, I do not believe the position of the 2 non-zero states will affect the outcome. Due to the two turns each players take. The additional min layer, negates any possibility for the 1000 win state to progress up the tree, and the preceding max layer prevents a -1000 win state to progress up the tree.

- **If this question was changed to have a different depth, would it change the answers to the two questions above? If yes, how do the answers change? If no, explain why no change.**

  From those trials I ran, I do not believe the depth of the tree will affect the outcome. As increasing the depth simply continues the same issue as the previous answers have listed. For decreasing the depth, it will also not affect the outcome as long as each player gets 1 turn, then the min player will neutralize the win (1000) state and the max player player will neutralize the losee (-1000) state.

  Hence, due to the set-up of the problem, it will always end in a draw regardless of depth or the position of the 2 non-zero states.

Problem 3.

What are the variables, domains, and constraints?

Variables: -

Set of Across Words $\in \{A_1 \cdots A_n\}$

Set of Down Words $\in \{D_1 \cdots D_m\}$

Set of Words $\in \{w_1 \cdots w_z\}$, z being the finite number of words in a dictonary

Domains: -

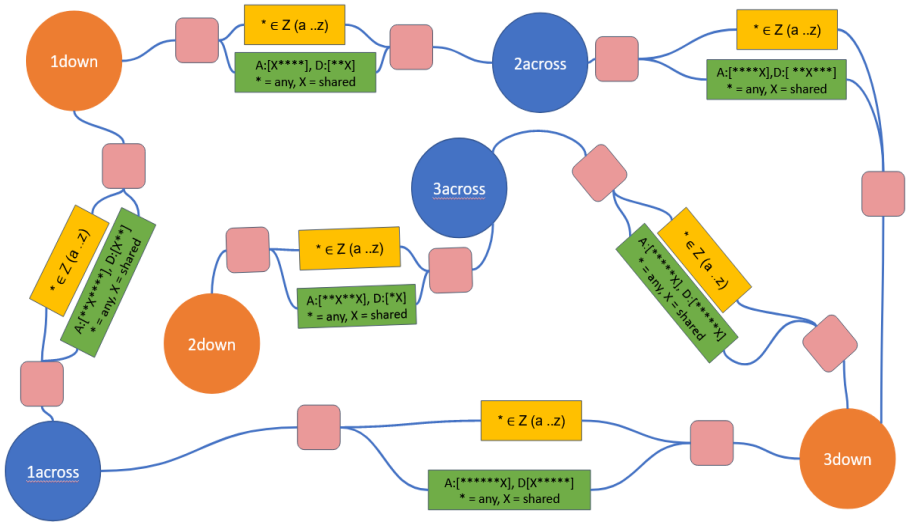
$\text{dom(W)} = \{\ w_i \text{ from Words} \mid < L_1, \cdots > \in w_i \mid L_i \in (a \cdots z)\}$

$\text{dom(A)} = \text{dom(D)} = \{\ w_i \text{ from Words} \mid \text{len}(w_i) = \text{len}(A_i \text{ or } D_i \text{ segement})\}$

Constraint: -

$c_i = \{cells(A_1) =< a_1, a_2, \cdots > |cells(A_i) \cap cells(D_i)|value(a_i) = value(d_i)\}$

Draw the constraint graph.



I drew the constraint graph using the n and m of the given example.

Describe the first couple of steps of how standard backtracking search would work

I complie these steps using the n and m of the given example. I assume acroynoms don't count as words, eg. APA

1. Start with 1across, say the first 7 letter word is abandon, nothing else in the crossword puzzle, so move on
2. Move to 2across, say the first 5 letter word is aback. no conflict, move on
3. Move to 3across, say the first 6 is abbots. no conflict, move on.
4. Move to 1down, Conflict as no 3-letter word starts with a and ends with a. So back-track back to 2across and remove abate from potential words.
5. Back at 2across, say the 2nd 5-letter word is abend. No conflict, so move on.
6. Move to 1down, say the 1st viable 3-letter word is Ape. No conflict, move on.
7. Move to 2down, etc.$\cdots$

Describe how using the MRV would change the search

Based on the example crossword and the above back-track steps. MRV would have changed the search, specifically when it reaches step 2. It would have removed all the words from our Word domain from our Word variable that didn't have *a* at the start and end.

Thus, our search would have realized then there was an inconsistency with choosing aback as the word for 2across. As there would be no remaing words left in the Word variable from this action, the search would have chosen another 5-letter word there and then. Therefore, MRV essentially prevented a back-track operation, saving us memory and reducing the overall need for continually keeping the past states; as those were primarily for back-tracks.