SAARLAND UNIVERSITY — ARTIFICIAL INTELLIGENCE
Prof. Dr. Jörg Hoffmann, Dr. Daniel Fišer, Daniel Höller, Dr. Sophia Saller

Theoretical Exercise Sheet 4

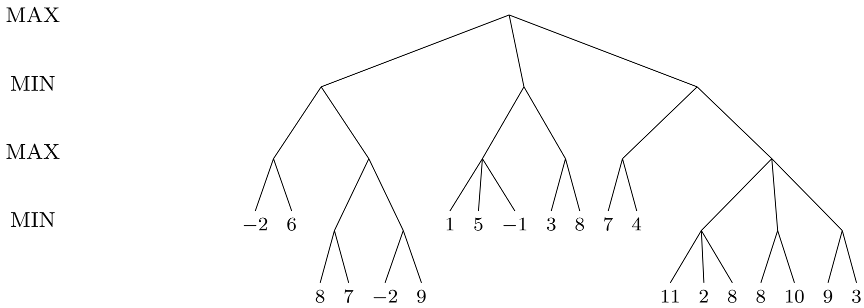
Deadline Friday, May 13, 23:59

About the submission of this sheet.

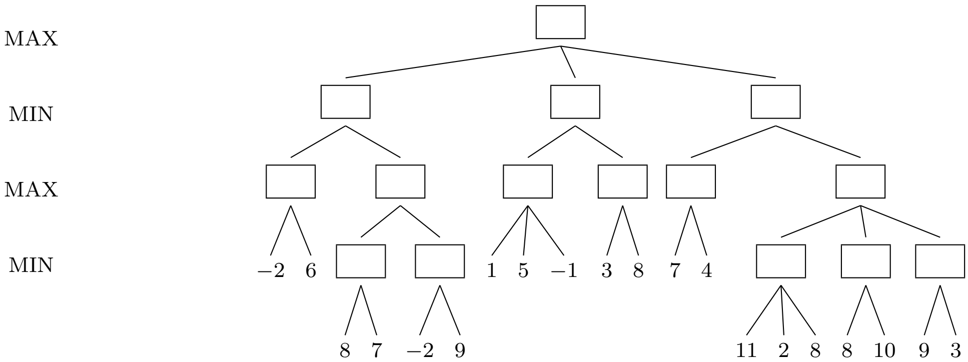
- You might submit the solutions to exercises in groups of up to 3 students.
- All students of a group need to be in the same tutorial.
- Write the names of **all** students of your group on your solution.
- Hand in the solution **in CMS**. We have activated "Team Groupings" in CMS.
  - Go to your personal page in CMS. Here you find the entry "Teams".
  - When you click "Create team", you get an invite code.
  - Please share this with your team mates, who need to click on "Join team" and enter the code.

(Please turn the page)

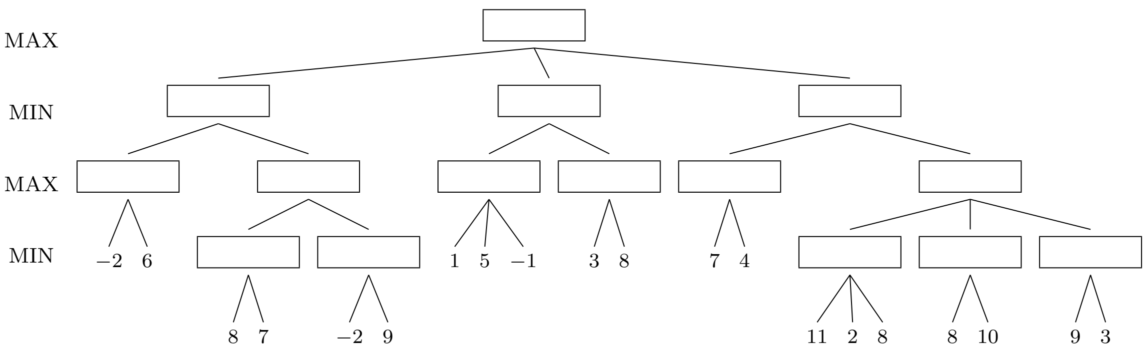
1. (32 points) Consider the following game tree corresponding to a two-player zero-sum game as specified in the lecture. As usual, **Max is to start in the initial state (i.e., the root of the tree)**. For the following algorithms, the expansion order is **from left to right, i.e., in each node the left-most branch is expanded first.**



1. In the following tree, perform Minimax search, i.e., annotate all internal nodes with the correct Minimax value. Which move does Max choose?



2. In the following tree, perform Alpha-Beta search. Annotate all internal nodes (that are not pruned) with the value that will be propagated to the parent node as well as the final $[\alpha, \beta]$ window before propagating the value to the parent. Mark which edges will be pruned. How many leaf nodes are pruned?



3. Consider, again, our game tree given above. Max chooses the action with the highest utility in the root of the tree. What can you say about the utility he will receive against:
   - An optimally playing opponent
   - A non-optimal playing opponent

   Note: Max always plays optimally.
4. It is possible to prune even more nodes by reordering the nodes of the tree. Develop a general recipe to reorder the nodes of a Minimax tree such that a maximal number of leaf nodes is pruned in Alpha-Beta pruning. **No** formal algorithm is required.

2. (32 points) Max and Minnie are playing a classic children's game called Tic-tac-toe in which players take turns marking the cells in a 3×3 grid. Max marks Xs and Minnie marks Os. Minnie wins with utility -1 if **any line (horizontal, vertical, or diagonal)** fills up with three Os, whereas Max wins with utility $+1$ if **any** line fills up with three Xs. If there are no empty cells left and no one has won so far, the game ends in a draw with utility 0.

   (a) Consider the state depicted below. Here, it is Max's turn to play.

X		O
X	O	
		O

   Draw the **full** Minimax tree and annotate every node with its utility.

   (b) Consider the evaluation function $f(x) :=$ the number of (horizontal) rows which contain at most one O. For example, in the initial state $f(x) = 3$.

   Draw the Minimax tree with this evaluation function and a depth of 2. Annotate every node with its utility.

   (c) Assuming a perfectly playing opponent, Minimax search without a depth limit will always guarantee a draw. However, it is not obvious whether this guarantee can still be made when imposing certain depth limits in combination with certain evaluation functions. As an example, consider the simple evaluation function g aimed to detect situations in which the opponent can win in one step (if it is their turn).

   $$g(x) := \left\{ \begin{array}{rl} -1, & \text{if there is a line (horizontal, vertical, or diagonal) with} \\ & \text{two opponent marks and an empty field} \\ 0, & \text{otherwise} \end{array} \right\}$$

   Now prove or disprove the following claim:

   Running Minimax search with a depth limit of 3 and evaluation function g is sufficient to guarantee a draw against a perfectly playing opponent in a 3×3 Tic-tac-toe game. You may **not** assume that you are allowed to start the game.

3. (16 points) Please decide for each of the following statements whether it is true or false and justify your answer (1-3 sentences per statement).

   1. In Alpha-Beta pruning, all unexplored successors of a Min-node n will be pruned if one of the explored successors of n has a higher utility score than the lowest Min-node utility already found on the path to n.
   2. There exists a Minimax tree in which Alpha-Beta pruning will not prune any leaf nodes.
   3. A two-player zero-sum game has exactly three possible outcomes in terms of its utility function.
   4. Full Minimax search always yields the best possible outcome in terms of its utility, no matter how the opponent plays.