02180 Intro to AI
Exercises for week 5, 2/3-21
SOLUTIONS

Exercise 1

Solve questions (a), (b) and (c) of Exercise 8 in Chapter 5 of R&N: "Consider the two-player game described in Figure 17..."

SOLUTION.

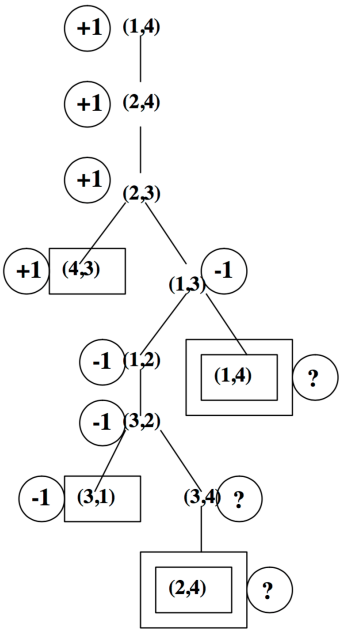
5.8 Consider the two-player game described in Figure 5.17.

a. Draw the complete game tree, using the following conventions:

- Write each state as (s_A, s_B), where s_A and s_B denote the token locations.
- Put each terminal state in a square box and write its game value in a circle.
- Put *loop states* (states that already appear on the path to the root) in double square boxes. Since their value is unclear, annotate each with a "?" in a circle.

b. Now mark each node with its backed-up minimax value (also in a circle). Explain how you handled the "?" values and why.

c. Explain why the standard minimax algorithm would fail on this game tree and briefly sketch how you might fix it, drawing on your answer to (b). Does your modified algorithm give optimal decisions for all games with loops?



a)

b) The "?" values are handled by assuming that an agent with a choice between winning the game and entering a "?" state will always choose the win. That is, $min(-1, ?)$ is -1 and $max(+1, ?)$ is $+1$. If all successors are "?", the backed-up value is "?".

c) Standard minimax would fail by going into an infinite loop and never produce values for the loop states. It can be fixed by keeping track of loop states, mark them by "?" and use the revised min and max functions from the previous questions.

The algorithm can not be guaranteed to give optimal decisions for all games with loops. For example, it is not clear how to compare "?" with a draw position: a draw position is not better or worse for either player, but might at least bring an end to the game, so assigning 0 to "?" states will not work there.

Exercise 2

Solve questions (a), (b), (c) and (d) of Exercise 9 in Chapter 5 of R&N: "This problem exercises the basic concepts of game playing, using tic-tac-toe..."

SOLUTION.

5.9 This problem exercises the basic concepts of game playing, using tic-tac-toe (noughts and crosses) as an example. We define X_n as the number of rows, columns, or diagonals

with exactly n X's and no O's. Similarly, O_n is the number of rows, columns, or diagonals with just n O's. The utility function assigns $+1$ to any position with $X_3 = 1$ and -1 to any position with $O_3 = 1$. All other terminal positions have utility 0. For nonterminal positions, we use a linear evaluation function defined as $Eval(s) = 3X_2(s)+X_1(s)-(3O_2(s)+O_1(s))$.

a. Approximately how many possible games of tic-tac-toe are there?

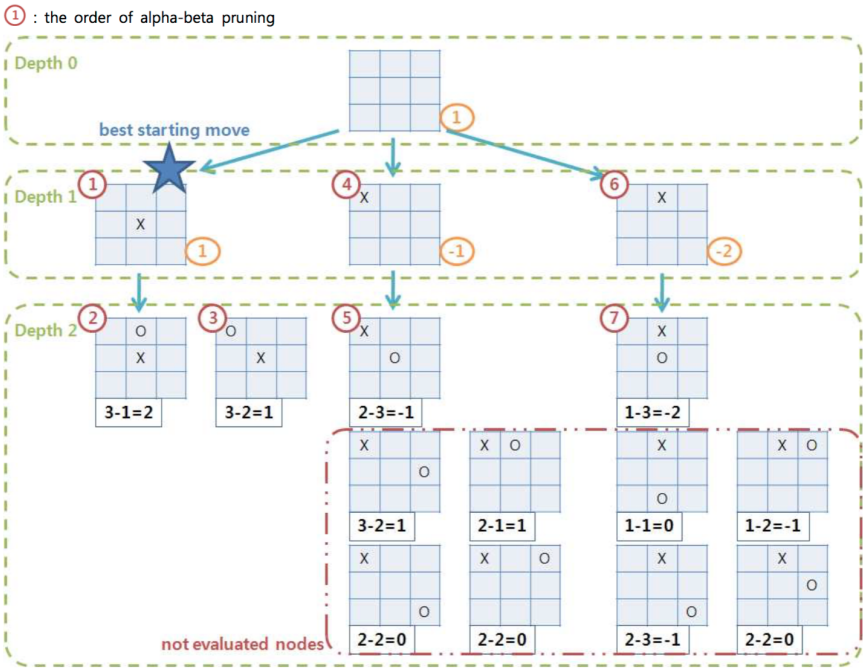
b. Show the whole game tree starting from an empty board down to depth 2 (i.e., one X and one O on the board), taking symmetry into account.

c. Mark on your tree the evaluations of all the positions at depth 2.

d. Using the minimax algorithm, mark on your tree the backed-up values for the positions at depths 1 and 0, and use those values to choose the best starting move.

a) There are 9! ways of placing the Xs and Os on the board (ignoring symmetry). So a simple upper bound on the number of games is $9! = 362.880$.

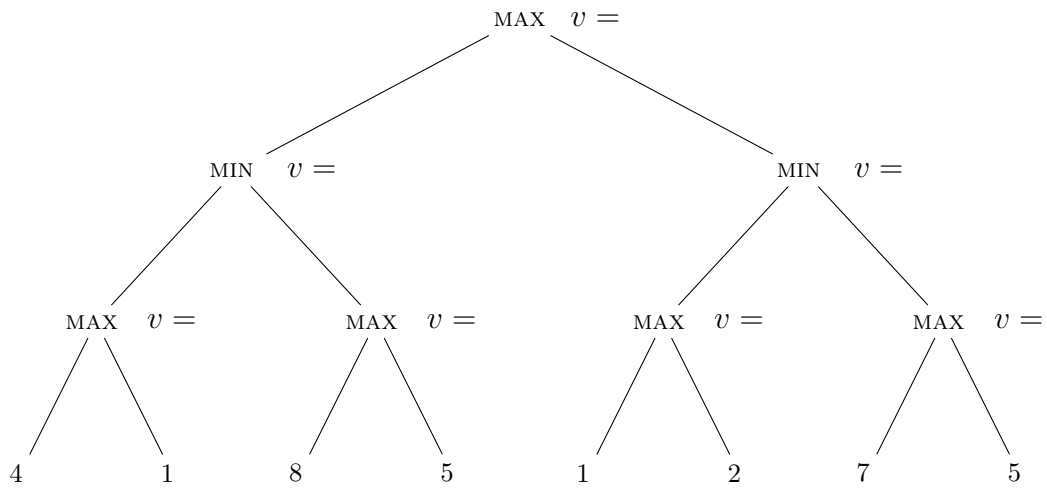
b)



Exercise 3

This is an exam exercise from the course in spring 2018.

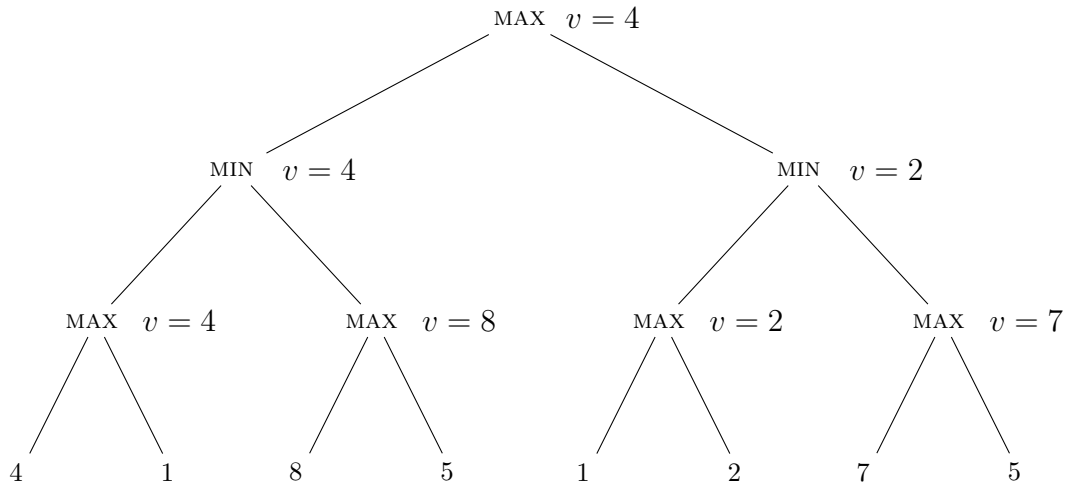
1. Consider the game tree below with MAX and MIN nodes explicitly marked. Add the MINIMAX value of each node to the game tree (after "$v =$"). Then highlight the edge corresponding to the best move of MAX in the initial state of the game.



2. Assume ALPHA-BETA-SEARCH (MINIMAX with α-β pruning) is used to explore the game tree. Assume that the children of a node are visited in order from left to right. Mark any

cuts in the tree (pruned subtrees) above and, for each, indicate whether it is an α-cut or a β-cut.

SOLUTION.



The edge from the root to the left child should be highlighted. There are two cuts: a β-cut to the leftmost leaf with value 5. There is an α-cut on the rightmost MAX-subtree of depth 1.