Learning Objectives

- To practice running minimax search and alpha-beta pruning.
- To evaluate the properties of a real-world example of adversarial search.

Q1. Adversarial Search

Observe the minimax graph below.

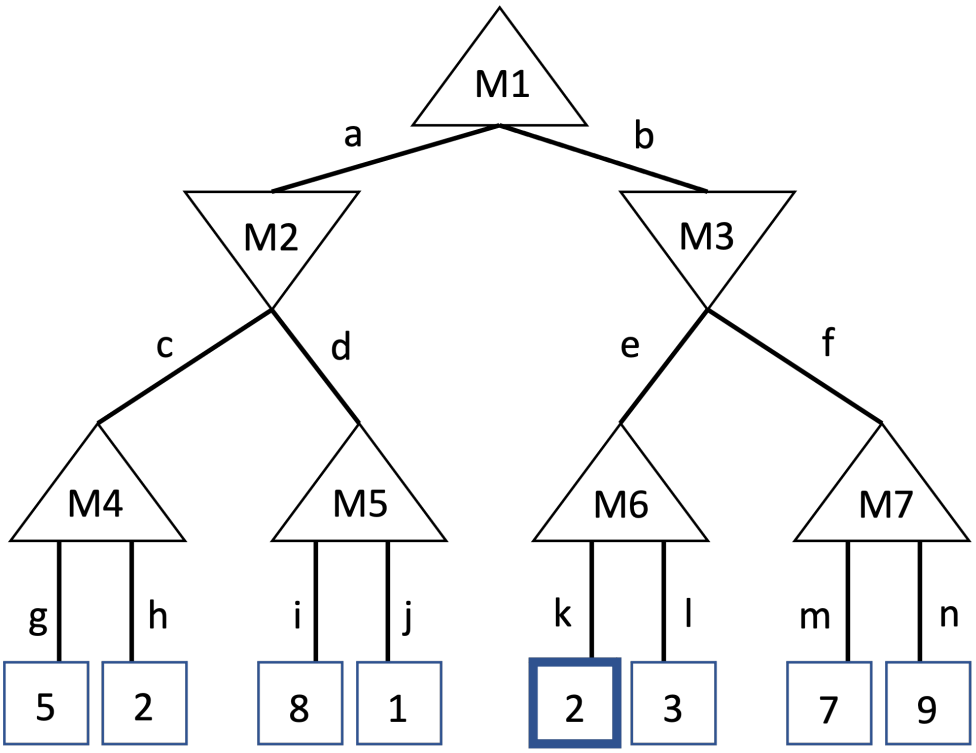

Figure 1: Graph

(a) What is the minimax value of the root (M1)? M1=5, M2=5, M3=3, M4=5, M5=8, M6=3, M7=9.

(b) What is the smallest integer value could you change the highlighted 2 to change the value of M1? In order to change the value of M1, M3 needs to be greater than 5. This will only happen if M6 and M7 are greater than 5. So you could change the 2 to a 6 to satisfy that requirement.

(c) With the original 2 in place again, what edges would be pruned in alpha-beta pruning?

The j edge would be pruned because the min node M2 would not choose anything higher than 5 and the max node M5 would not choose anything lower than 8.

The f branch would be pruned (M7 would not be seen). The max node M6 would be 3, so M3 would be 3. M3 wouldn't choose anything more than 3 but M1 wouldn't pick anything less than 5.

Q2. Connect 4

Connect Four is a two player game in which players take turns dropping colored discs into a column of a 6 row by 7 column grid. The discs stop at the lowest open row in the column. The first player with 4 of their colored discs to appear consecutively (horizontally, vertically, or diagonally) wins.

Figure 2: Connect 4 game

(a) You decide to model each player as an agent. Is the game static or dynamic? Is it stochastic or deterministic? Is it fully observable or partially observable? The environment is static; when both agents stop playing, the board doesn't move. The actions are deterministic; even though the piece moves down the board, you know exactly where it will stop. It is fully observable; both agents can see the whole board.

(b) What is the maximum branching factor for each agent? What is the minimum branching factor? Maximum is 7; there are seven columns, so the agent could put the piece in any of those columns. Minimum is 1; this would happen in the case where all columns but one are filled up

(c) What is the maximum depth of the search tree? $7 * 6 = 42$ levels. Each player fills in one square at a time. The grid is 7 x 6 so there are 42 squares. If the game ends in a tie, it would take 42 moves to get there.

(d) Is it feasible to search through the whole minimax tree to find a best move? If not, what kinds of heuristics could you create to value a non-leaf node?

No it is not feasible. Supposing that most games end after more than 20 moves, that would be more than $8x10^{16}$ leaves to explore.

There are many possible answers to the heuristic question. You could say that the value could be the maximum number of squares in a row. Or the maximum number of squares in a row with the remaining squares open.

Bad examples would be the number of squares remaining or the difference in the number of chips for each player. Both are not helpful for determining if someone will win.

(e) Is the minimax algorithm still optimal if you used a heuristic to value non-leaf nodes?

No it is not optimal. Depending on the heuristic, the new values for non-leaf nodes will not lead to the same min or max nodes as the original minimax algorithm.