# INFO8006 Introduction to Artificial Intelligence

Exercises 3: Games and adversarial search

## Learning outcomes

At the end of this exercise session you should be able to:

- Define formally the search problem associated to a game (IPATTU[1])
- Define and apply the Minimax algorithm
- Define and apply $\alpha - \beta$ pruning for Minimax
- Define H-Minimax, Expectiminimax, Monte-Carlo Tree Search

## Exercise 1: Tic-Tac-Toe (AIMA, Ex 5.9)

*Tic-tac-toe is a paper-and-pencil game for two players, X and O, who take turns marking the cells of a $3 \times 3$ grid. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row wins the game.*
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 $Eval(s) = 3X_2(s) + X_1(s) - (3O_2(s) + O_1(s))$.

1. Define the search problem associated with the Tic-tac-toe game.
   Let us first define a game state $s \in \{-1, 0, 1\}^{3\times 3}$, where the three values $-1, 0, 1$ respectively denote O, empty and X cells.

   - Initial state:
   $$s_0 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$
   - Player function: $player(s) = 1 \quad \text{if} \quad \Delta(s) = 0 \quad \text{and} \quad -1 \quad \text{otherwise, where}$
   $$\Delta(s) = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} s \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}.$$
   - Legal actions: An action is defined by a tuple $a = (a^1, a^2) \in \{0, 1, 2\} \times \{0, 1, 2\}$. $actions(s) = \{a : s_{a^1,a^2} = 0\}$.
   - Transition model: $result(s, a) = s'$ where $s'$ is the same as $s$ except that $s'_{a^1,a^2} = player(s)$.
   - Terminal test: $terminal(s) =$ True if $s$ contains 3 adjacent cells with $+1$, 3 adjacent cells with $-1$ or if the board is full.
   - Utility function: $utility(s, p) = 1$ if 3 adjacent cells with $+1$ else if the board is full $, 0$ else $- 1$.

2. Approximately how many possible games states of Tic-tac-toe are there?
   If we disregard unreachable states, we have $3^{3\times 3} = 19683$ possible states. And how many games (game state sequence)? $9! = 362880$

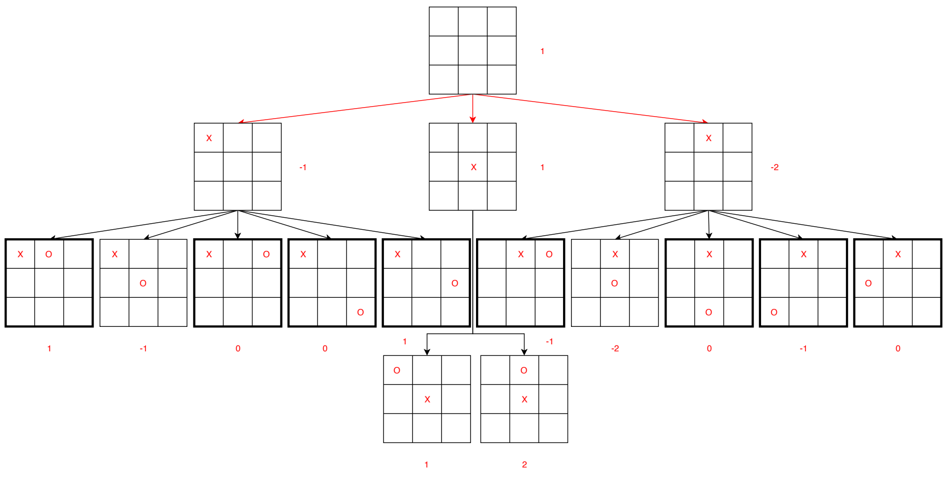
3. Show the whole game tree starting from an empty grid down to depth 2 (i.e., one X and one O on the board), taking symmetry into account.

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

5. 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.

6. Circle the nodes at depth 2 that would not be evaluated if $\alpha - \beta$ pruning were applied, assuming the nodes are generated in the optimal order for $\alpha - \beta$ pruning. See the figure below for the solutions to points 3 to 6.

[1] Initial state - Player function - Actions function - Transition model - Terminal test - Utility function



## Exercise 2: 21 misère game (January 2019)

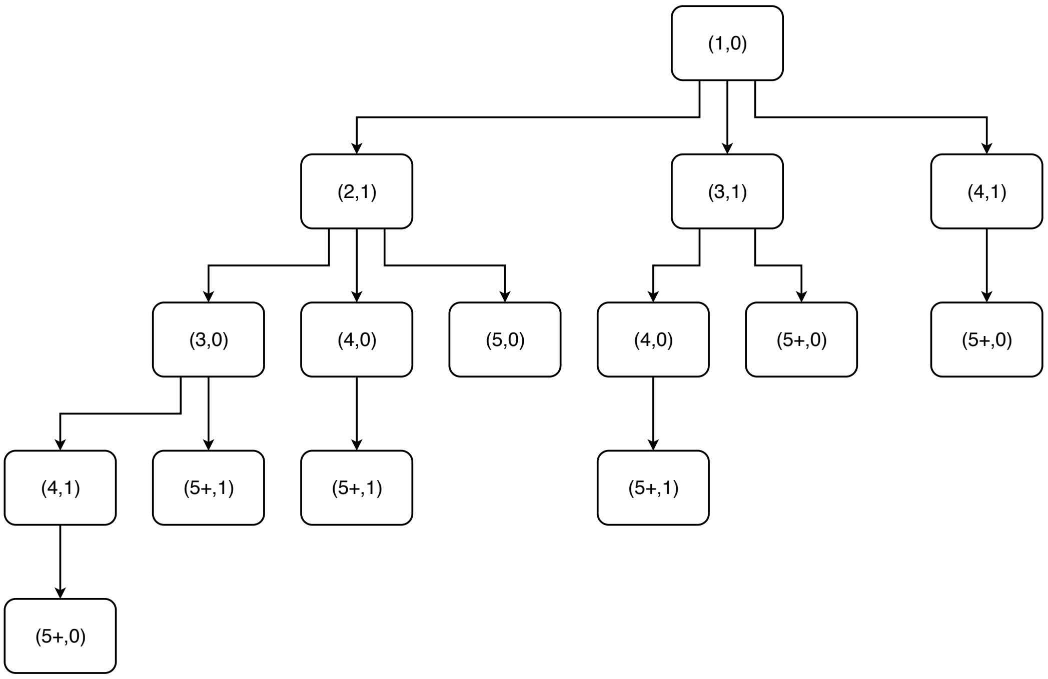
The game ”21” is played as a misère game with any number of players who take turns saying a number. The first player says ”1” and each player in turn increases the number by 1, 2, or 3, but may not exceed 21; the player who says ”21” or a larger number loses.

1. Define the search problem associated with the 2-player version of the ”21” game.

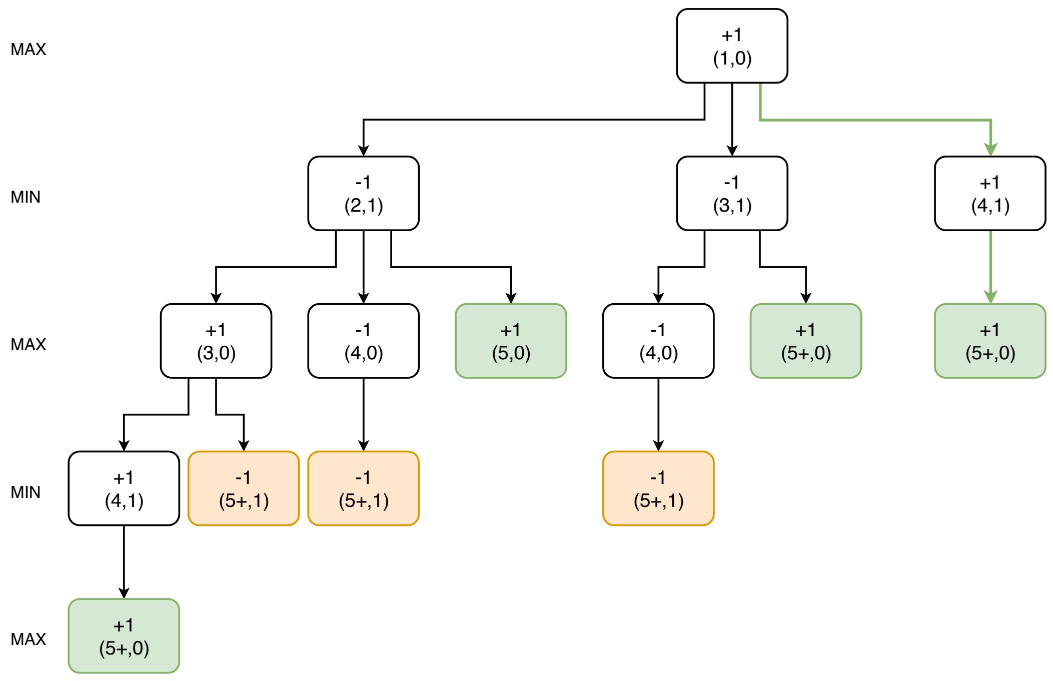
   We define a state $s$ as a pair $s = (s^0, s^1) \in \mathbb{Z} \times \{0,1\}$ such that the first element $s^0$ is the value played by the last player and the second element $s^1$ corresponds to the player to play next. The first player has id $p = 1$ and the second player has id $p = 0$.

   - Initial state: $s_0 := (s_0^0, s_0^1) := (1, 0)$.
   - Player function: $player(\cdot) : \mathbb{Z} \times \{0,1\} \to \{0,1\}$, such that $player(s = (s^0, s^1)) := s^1$.
   - Action function: actions are denoted by the integer values that can be played by the current player. We define the action function $action(\cdot) : \mathbb{Z} \times \{0,1\} \to \mathbb{Z}^3$ such that $action(s = (s^0, s^1)) := \{s^0 + 1, s^0 + 2, s^0 + 3\}$.
   - Transition function: $result(\cdot,\cdot) : (\mathbb{Z} \times \{0,1\}) \times \mathbb{Z} \to \mathbb{Z} \times \{0,1\}$ such that $result(s = (s^0, s^1), a) := (a, (s^1 + 1) \bmod 2)$.
   - Terminal function: $terminal(\cdot) : \mathbb{Z} \times \{0,1\} \to \{0,1\}$ such that $terminal(s = (s^0, s^1)) := s^0 \geq 21$.
   - Utility function: $utility(\cdot,\cdot) : (\mathbb{Z} \times \{0,1\}) \times \{0,1\} \to \{-1,1\}$ such that $utility(s = (s^0, s^1), p) := 1 - 2|p - s^0|$.

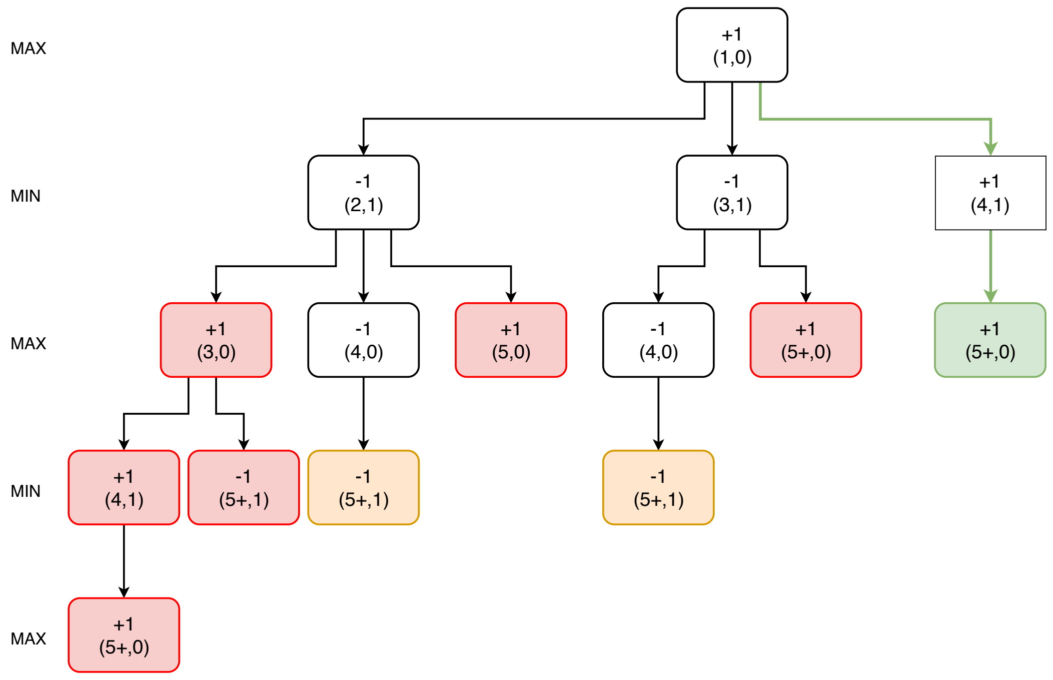
2. For this subquestion and the following, consider the game of ”5” (still in its 2-player version) which has the same rule except that you should not say 5 or more. Show the whole game tree.



3. (a) Using the minimax algorithm, mark on your tree the backed-up values, and use those values to choose the best starting move
From the game tree we conclude that the best move is to play 4. NB: The winning strategy for the game of 21 is to always say a multiple of 4; it is then guaranteed that the other player will ultimately have to say "21" – so in the standard version where the first player opens with "1", they start with a losing move.



(b) Assume alpha–beta pruning were applied in optimal order. Draw the game tree containing only the nodes that would be evaluated.
Nodes that are not visited are colored in red.



## Exercise 3: Quiz

1. In a fully observable, turn-taking, zero-sum game between two perfectly rational players, it does not help the first player to know what strategy the second player is using (that is, what move the second player will make, given the first player's move). It doesn't help the 1st player to know the strategy of the other.
2. What is a quiescent position? A position in which the outcome of a game is unlikely to vary a lot in the near future.

3. In MCTS, what is encouraged by each term of the sum in the formula $\frac{Q(n',p)}{N(n')} + c\sqrt{\frac{2\log N(n)}{N(n'}}$? 1) Exploitation 2) Exploration.

4. Which heuristic is correct (i.e. resulting in a rational agent) (see image below)? Left.

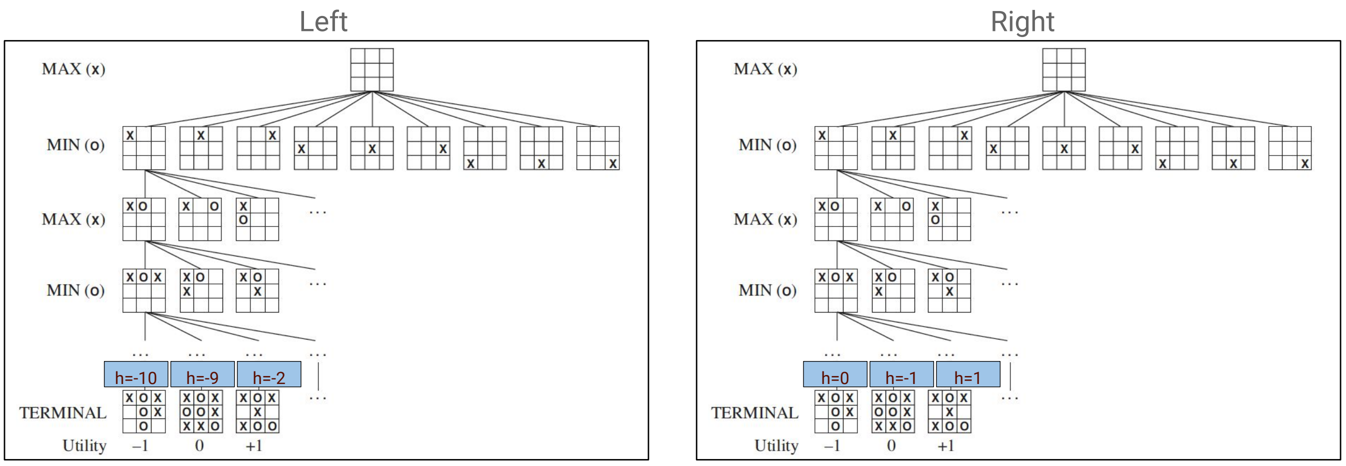


Figure 1: Two possible heuristics

## Exercise 4: Chess and transposition table ⋆ (AIMA, Ex 5.15)

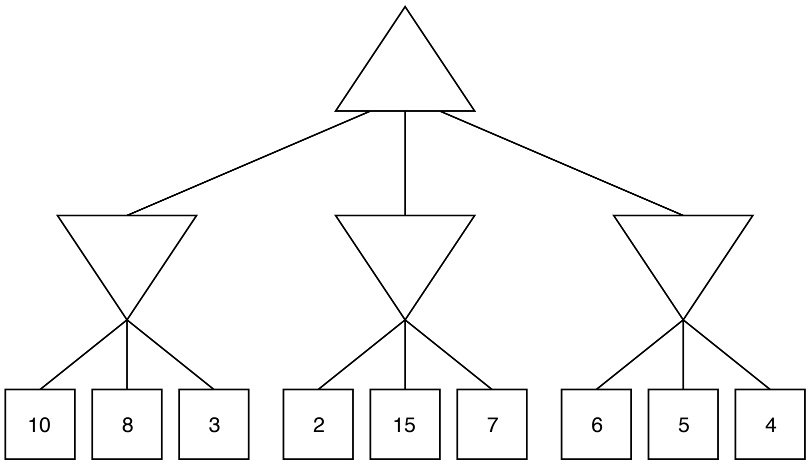
Suppose you have a chess program that can evaluate 16 million nodes per second. Decide on a compact representation of a game state for storage in a transposition table.

1. About how many entries can you fit in a 4-gigabyte in-memory table? With 32 pieces, each needing 6 bits to specify its position on one of 64 squares, we need 24 bytes (6 32-bit words) to store a position, so we can store roughly 160 million positions in the table (ignoring pointers for hash table bucket lists).

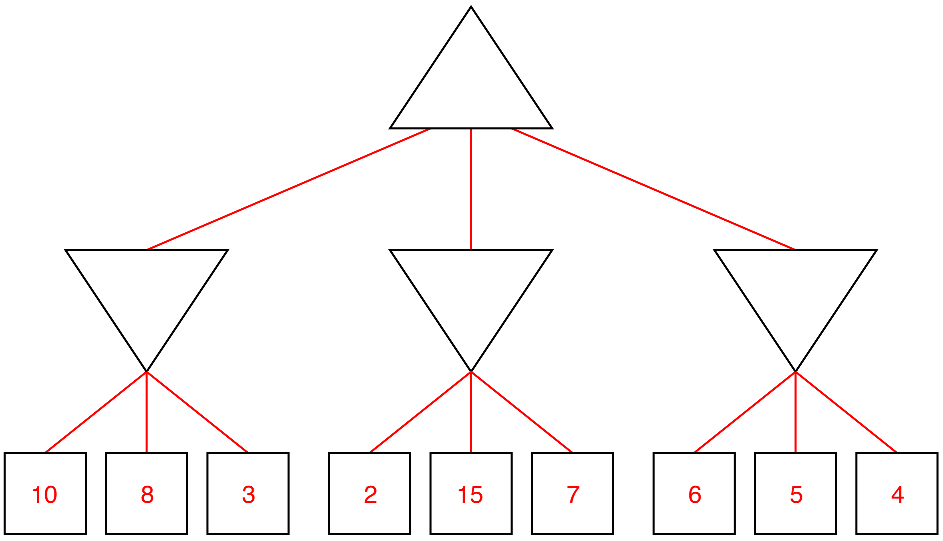
2. Will that be enough for the three minutes of search allocated for one move? No, this is about 1/18 of the 2880 million positions generated during a three-minute search.

3. How many table lookups can you do in the time it would take to do one evaluation? Suppose that you have a 3,2GHz machine and that it takes 20 operations to do one lookup on the transposition table. The number of operations per evaluation is $\frac{3.2\times10^9 \text{op/s}}{16\times10^6 eval/s}$ = 200op/eval, and so 10 lookups can be made for the same amount of time.

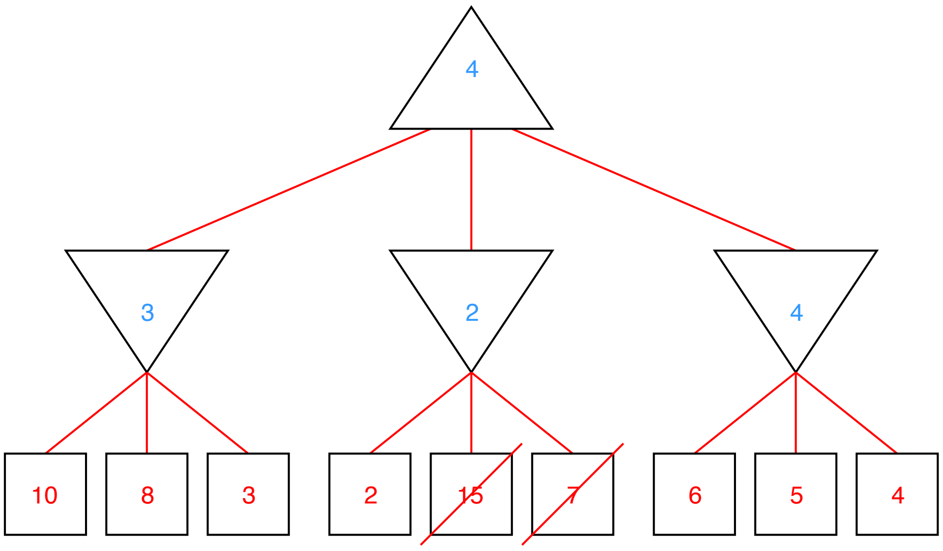
## Exercise 5: Minimax ⋆ (Berkeley CS188 Fall 2019)



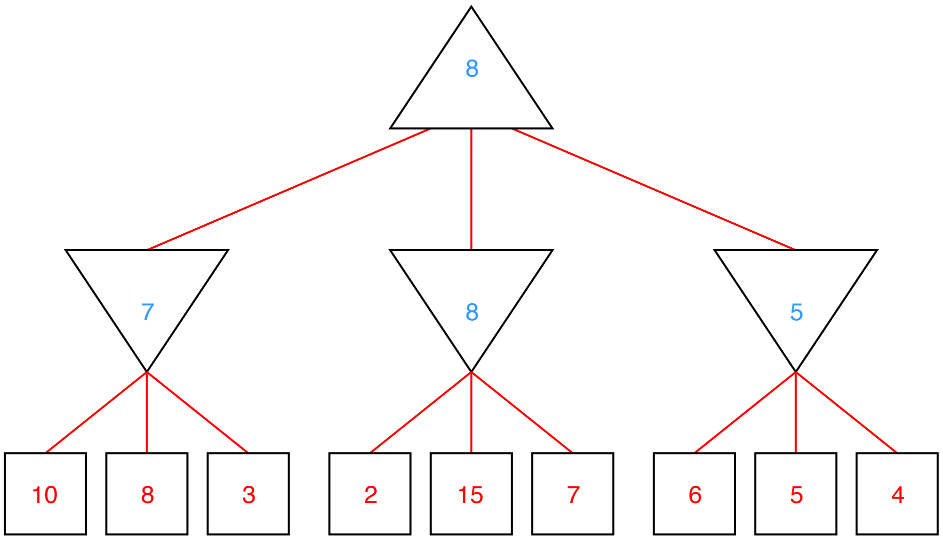
1. Consider the zero-sum game tree shown above. Triangles that point up, such as at the top node (root), represent choices for the maximizing player; triangles that point down represent choices for the minimizing player. Assuming both players act optimally, fill in the minimax value of each node. See the figure below.



2. Which nodes can be pruned from the game tree above through alpha-beta pruning? If no nodes can be pruned, explain why not. Assume the search goes from left to right; when choosing which child to visit first, choose the left-most unvisited child. See the figure below.



3. Again, consider the same zero-sum game tree, except that now, instead of a minimizing player, we have a chance node that will select one of the three values uniformly at random. Fill in the expectimax value of each node. The game tree is redrawn below for your convenience. See the figure below.



4. Which nodes can be pruned from the game tree above through alpha-beta pruning? If no nodes can be pruned, explain why not. No nodes can be pruned. There will always be the possibility that an as-yet-unvisited leaf of the current parent chance node will have a very high value, which increases the overall average value for that chance node. For example, when we see that leaf 4 has a value of 2, which is much less than the value of the left chance node, 7, at this point we cannot make any assumptions about how the value of the middle chance node will ultimately be more or less in value than the left chance node. As it turns out, the leaf 5 has a value of 15, which brings the expected value of the middle chance node to 8, which is greater than the value of the left chance node. In the case where there is an upper bound to the value of a leaf node, there is a possibility of pruning: suppose that an upper bound of $+10$ applies only to the children of the rightmost chance node. In this case, after seeing that leaf 7 has a value of 6 and leaf 8 has a value of 5, the best possible value that the rightmost chance node can take on is $\frac{6+5+10}{3} = 7$, which is less than 8, the value of the middle chance node. Therefore, it is possible to prune leaf 9 in this case.