

Minimax	start time:
---------	-------------

Before you start, share this document with your team member(s) and then complete the form below to assign the role of speaker.

Team Role	Team Member
Speaker: shares your team's ideas with the class.	Nic, Arogya, Kelii

A. Game	start time:
---------	-------------

- S_0 : The **initial state**, which specifies how the game is set up at the start.
- $\text{PLAYER}(s)$: Defines which player has the move in a state.
- $\text{ACTIONS}(s)$: Returns the set of legal moves in a state.
- $\text{RESULT}(s, a)$: The **transition model**, which defines the result of a move.
- $\text{TERMINAL-TEST}(s)$: A **terminal test**, which is true when the game is over and false otherwise. States where the game has ended are called **terminal states**.
- $\text{UTILITY}(s, p)$: A **utility function** (also called an objective function or payoff function), defines the final numeric value for a game that ends in terminal state s for a player p .

Q1. List at least five games you are familiar with.

Chess
 tic tac toe
 Starcraft 2
 checkers
 Go

Q2. A game with two players is “zero-sum” if what is good for one player is just as bad for the other. There is no “win-win” outcome. Please review the games in Q1 and check if they are zero-sum games or not.

Chess
 Starcraft 2
 Checkers
 Go

Q3. Based on the formulation of the game above, please describe each item for the game [Tic-Tac-Toe](#).

So: Empty Tic-Tac-Toe board (3x3 matrix)

Player: Player with O or X

Actions: List of empty spaces

Result: new state of the board after the action

Terminal Test: If three X or O in row/column/diagonal or no more moves

Utility: 1 for win 0 for tie -1 for lose. Can also be a tie.

Q4. Describe all items for the game [connect 4](#).

So: Empty board matrix

Player: Red or yellow

Actions: non full columns

Result: New state of the board after the action

Terminal Test: Board full or 4 position in a row, col, diagonal

Utility: 1 for win 0 for tie -1 for lose. Can also be a tie.

Q5. Describe all items for [chess](#).

So: Board matrix with pieces in starting positions

Player: White or black

Actions: Valid moves for each piece based on their individual rules and board state.

Result: New state of the board after the action including removal of pieces

Terminal Test: Checkmate, Stalemate, Draw

Utility: 1 for win 0 for tie -1 for lose. Can also be a Draw or Stale mate.

Q6. What is the difference between the game problem and the searching problem?

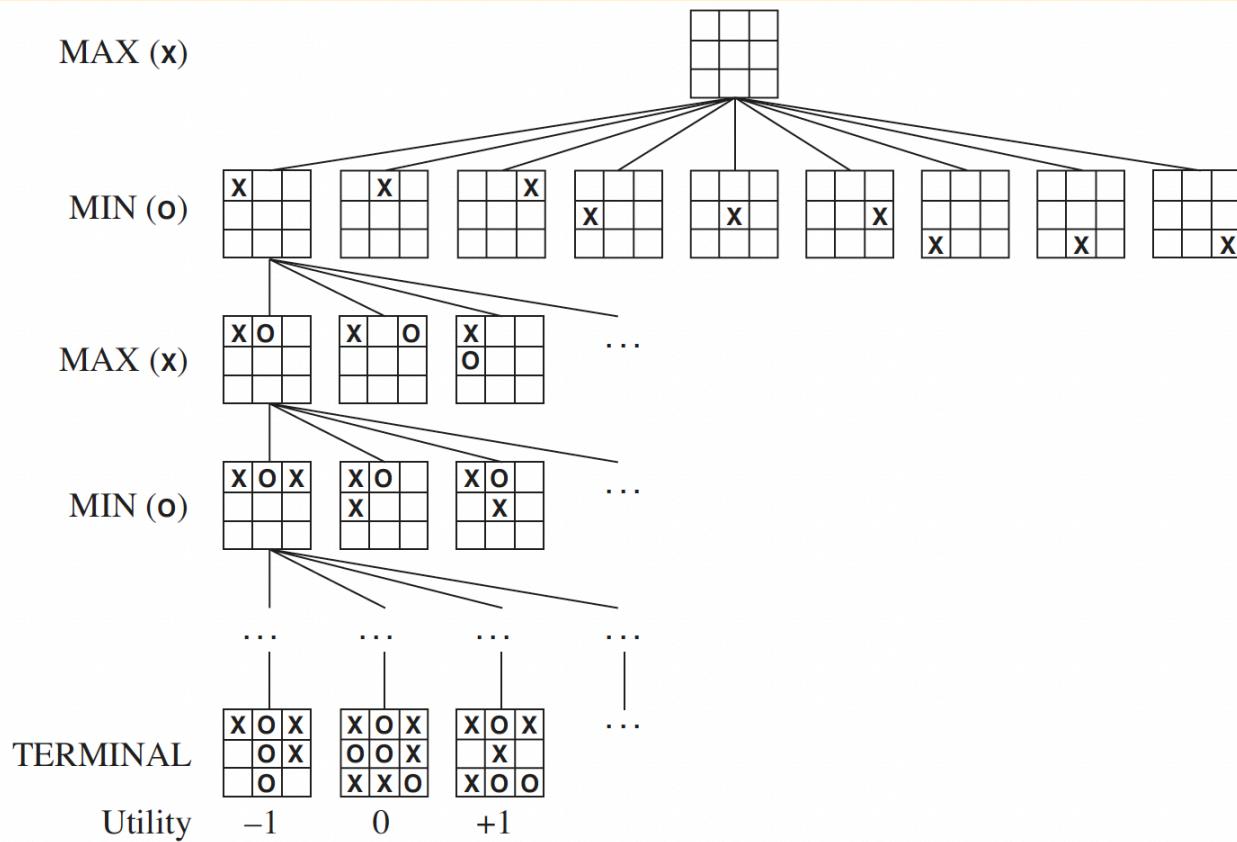
Game can change based on the opponent's move so the next move is not known. The game can still be searched accounting for every possible move the opponent makes, however many games have such a large set of possible moves (chess, go, especially games like Starcraft 2) that only a small subset of moves can be searched through.

A game like tic tac toe is small enough that all possible states can be searched.

We don't need to think about roles in searching problem, in game problem we have to make the next best move for the player and then switch player.

B. Game Tree

start
time:



The initial state, ACTIONS function, and RESULT function define the game tree for the game—a tree where the nodes are game states and the edges are moves. The figure above shows part of the game tree for tic-tac-toe (noughts and crosses). From the initial state, MAX has nine possible moves. Play alternates between MAX's placing an X and MIN's placing an O until we reach leaf nodes corresponding to terminal states such that one player has three in a row or all the squares are filled. The number on each leaf node indicates the utility value of the terminal state from the point of view of MAX; high values are assumed to be good for MAX and bad for MIN (which is how the players get their names).

Q7. What is the height of this game tree?

- 9 maximum

Q8. How long is the shortest path from the root to one terminal?

- 6

Q9. How many terminal nodes in the tree?

- At most 9!

Q10. If you are the MAX player that goes first in the game, what is your strategy to move downward in the game tree to win the game?

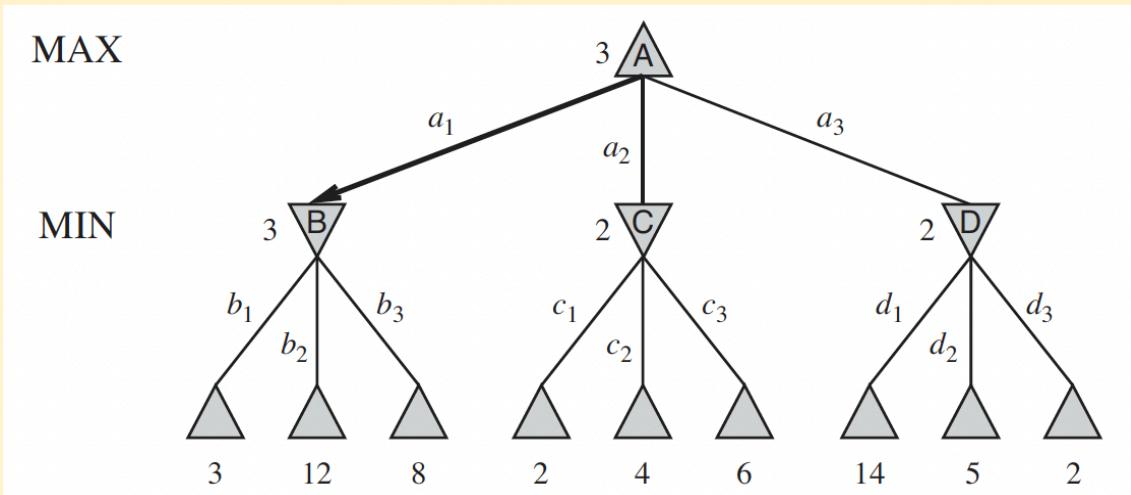
- See the winning path and follow that path

Q11. What if you are the MIN player that goes after MAX?

- Minimize the chance of path that goes to the positive path

C. Minimax

start time:



A two-player game. The upward triangle means MAX player and the downward triangle means MIN player. The number below the terminal means the utility value for the MAX.

Q12. What is the utility value for MAX if MAX chooses action/move a_1 and MIN chooses action/move b_3 ?

- 12

Q13. What is the best action for MIN if MAX chooses a_1 ?

- b_1

Q14. Answer the same questions for MIN if MAX chooses a_2 or a_3 .

- if a_2 : c_1
- if a_3 : d_3

Q15. Based on Q13 and Q14, what is the best action for MAX at the initial step?

- a_1

Q16. Given a game tree, the optimal strategy can be determined from the **minimax** value of each node, which we write as **MINIMAX(state)**. The minimax value of a node is the utility (for MAX) of being in the corresponding state, assuming that both players play **optimally** from there to the end of the game.

What is the minimax value at the terminal state?

same as utility value

Q17. What is the minimax value if MAX chooses action a1?

3

Q18. What is the minimax value of MAX at the initial/root state?

3

Q19. Please only use min and max operations on utility values at terminals to calculate the minimax value of the root.

if min player, then choose min on all minimax values

if max player, then choose max on all minimax values

Q20. Can we use the same kind of formula to calculate the minimax value at any state in the tree?

yes

Q21. Please propose a formula to calculate the minimax value for any game tree.

minimax of the root is:

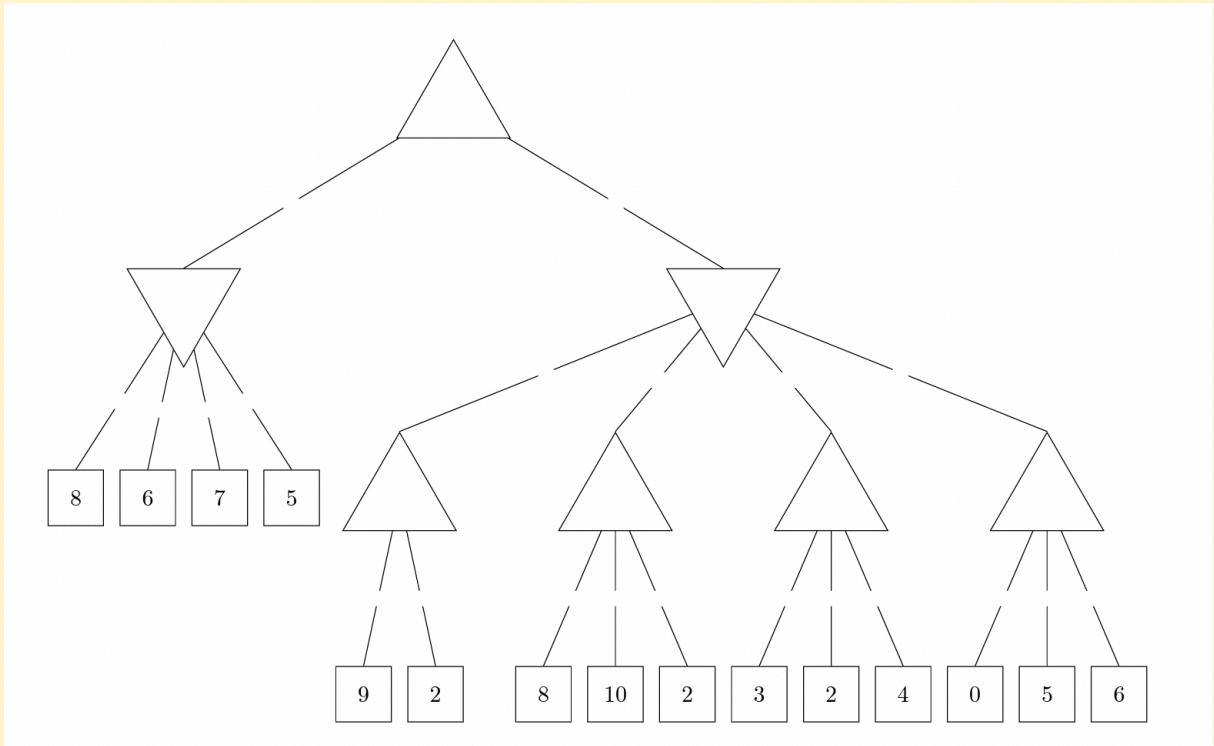
if root is terminal, minimax is utility value

if player is max, max of the utility values ; call minimax on actions

if player is min, min of the utility values ; call minimax on the actions

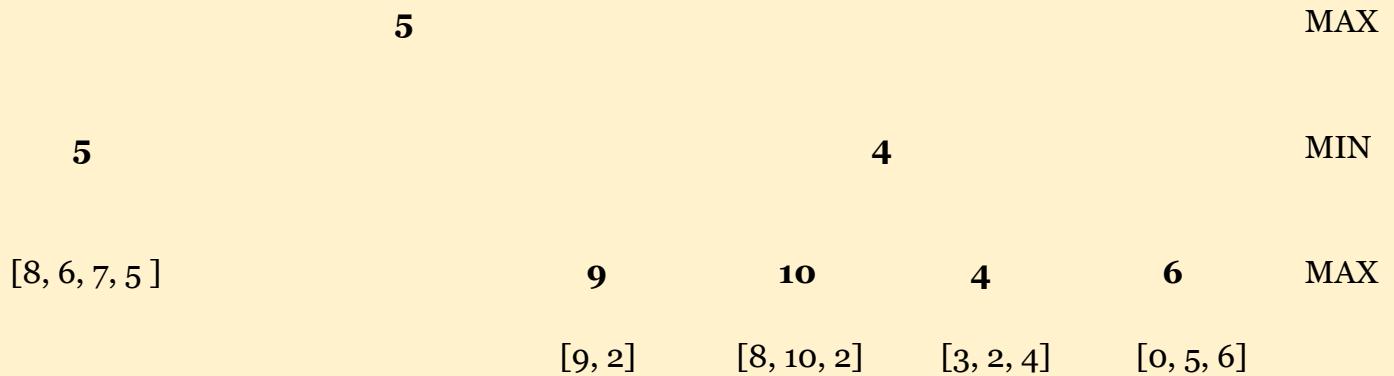
Q22. Based on your answer to Q21, please propose a strategy to play two-player and zero-sum games.

Choose the action with max minimax value for max player, and min minimax for min player.



Q23. Practice

Complete the game tree shown above by filling in values on the maximizer and minimizer nodes.



D. Minimax Simple Application

start time:

Let's consider a simple Tic-Tac-Toe scenario where it is X's turn, and there are 3 empty positions left. This exercise will guide you through solving a Tic-Tac-Toe Minimax problem manually. The goal is for you to construct the Minimax **tree** step by step, calculate **utility** at terminal states, and **backpropagate** values to calculate **minimax values** at internal states, and finally make the next move for X.

X | O | X

O | O | X

6 | 7 | 8

Q24. Understanding the Initial State. Given the following Tic-Tac-Toe board, where it is X's turn, what are the possible moves?

- 6, 7, OR 8

Q25. Constructing the Minimax Tree. Draw the first level of the Minimax tree where X makes a move.

X

CHOOSES 6

1

CHOOSE 7

0

CHOOSE 8

1

Q26. Constructing the Minimax Tree. After X moves, it is O's turn. What are O's possible responses for each case? Continue to draw the tree to the next level.

X

1

7

1

O:

6 8

Q27. Expanding the Game Tree to Full Size. After O makes a move, it is X's turn again. What are X's final possible moves? Complete drawing the tree.

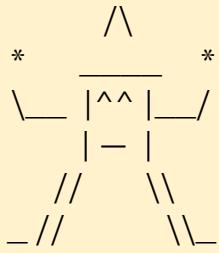
X: 1 (8)

6: -1	7:0	8: 1
O:	7:-1 8:0	6:1 8:0
X:	7:0	8:1 6:0

Q28. Evaluating Terminal States. Evaluate the final board states at the leaf nodes. What are the outcomes?

- 1 or 0. since x maximizes, x wins. during x's turn x will always choose 1

Q29. Backpropagating Minimax Values. Backpropagate values from the leaf nodes to the parent nodes (O's turn).



answered in question 27.

Q30. Choosing the Best Move. What is the best move for X based on the Minimax values?

6 (or 8)

E. Pseudocode of Minimax Search	start time:
--	----------------

```

function MINIMAX-DECISION(state) returns an action
    return  $\arg \max_{a \in \text{ACTIONS}(s)} \text{MIN-VALUE}(\text{RESULT}(s, a))$ 

function MAX-VALUE(state) returns a utility value
    if TERMINAL-TEST(state) then return UTILITY(state)
     $v \leftarrow -\infty$ 
    for each a in ACTIONS(state) do
         $v \leftarrow \text{MAX}(v, \text{MIN-VALUE}(\text{RESULT}(s, a)))$ 
    return v

function MIN-VALUE(state) returns a utility value
    if TERMINAL-TEST(state) then return UTILITY(state)
     $v \leftarrow \infty$ 
    for each a in ACTIONS(state) do
         $v \leftarrow \text{MIN}(v, \text{MAX-VALUE}(\text{RESULT}(s, a)))$ 
    return v

```

Q31. Minimax search always starts building a searching tree from the initial state. Does minimax need to expand the tree to full size, which generates all possible states, to determine the best action from the initial state?

- Yes it needs to expand to largest size and then only we can assign utility values for applying minimax.

Q32. When implementing minimax search, is it similar to DFS or BFS?

- DFS

Q33. If the maximum depth of the tree is m and there are b legal moves at each point, what is the time complexity of the search?

- $O(b^m)$

Q34. Based on the answer in Q31, what is the approximate time complexity of solving tic-tac-toe using minimax search?

- $O(9!)$

Q35. In chess, the branch factor is about 35, which means the player has about 35 possible moves each time. The depth is about 80 ply, which means the game usually ends in 80 steps. What is the time complexity of solving chess using minimax search? Is minimax a feasible algorithm for solving chess? if 80^{35} : $4.0564819e+66$