**Chapter 6**

Adversarial Search

**The chapter consist of Short type Questions & Answers , Descriptive Question & Answer and MCQs & answers**

Contents

[Short type Questions & Answers 1](#_Toc14943591)

[Define game. 1](#_Toc14943592)

[What is alpha-beta pruning? 2](#_Toc14943593)

[How many Move Ordering can be done in Alpha-Beta pruning. 2](#_Toc14943594)

[What are various Terminology in game playing (Adversarial Search) 2](#_Toc14943595)

[Define Offline search. 3](#_Toc14943596)

[What are Rules to find good ordering in alpha-beta pruning 3](#_Toc14943597)

[Define the term backtracking search. 4](#_Toc14943598)

[When a problem is called commutative? 4](#_Toc14943599)

[What do you mean by minimum remaining values? 4](#_Toc14943600)

[Descriptive Question & Answer 4](#_Toc14943601)

[Explain Minimax Algorithm and draw game tree for Tic Tac Toe Game. 4](#_Toc14943602)

[What is Alpha-Beta Pruning. Demonstrate with example 8](#_Toc14943603)

[Write a short note on : Types of Games in AI 14](#_Toc14943604)

[Why is game theory important to AI? 15](#_Toc14943605)

[MCQs & answers 16](#_Toc14943606)

# Short type Questions & Answers

## Define game.

A game can be defined by the initial state, the legal actions in each state, a terminal test and a utility function that applies to terminal states.

## What is alpha-beta pruning?

The problem with minimax search is that the number of games states it has to examine is exponential in the number of moves. We can’t eliminate the exponent, but we can effectively cut it in half. The trick is that is possible to compute the correct minimax decision without looking at every node in the game tree. This technique is called alpha-beta pruning.

## How many Move Ordering can be done in Alpha-Beta pruning.

The effectiveness of alpha-beta pruning is highly dependent on the order in which each node is examined. Move order is an important aspect of alpha-beta pruning.

It can be of two types:

* **Worst ordering:** In some cases, alpha-beta pruning algorithm does not prune any of the leaves of the tree, and works exactly as minimax algorithm. In this case, it also consumes more time because of alpha-beta factors, such a move of pruning is called worst ordering. In this case, the best move occurs on the right side of the tree. The time complexity for such an order is O(bm).
* **Ideal ordering:** The ideal ordering for alpha-beta pruning occurs when lots of pruning happens in the tree, and best moves occur at the left side of the tree. We apply DFS hence it first search left of the tree and go deep twice as minimax algorithm in the same amount of time. Complexity in ideal ordering is O(bm/2).

## What are various Terminology in game playing (Adversarial Search)

* **Game Tree**: It is a structure in the form of a tree consisting of all the possible moves which allow you to move from a state of the game to the next state.

A game can be defined as a search problem with the following components:

* **Initial state**: It comprises the position of the board and showing whose move it is.
* **Successor function**: It defines what the legal moves a player can make are.
* **Terminal state**: It is the position of the board when the game gets over.
* **Utility function**: It is a function which assigns a numeric value for the outcome of a game. For instance, in chess or tic-tac-toe, the outcome is either a win, a loss, or a draw, and these can be represented by the values +1, -1, or 0, respectively. There are games that have a much larger range of possible outcomes; for instance, the utilities in backgammon varies from +192 to -192. A utility function can also be called a payoff function.

## Define Offline search.

Offline search algorithms compute a complete solution before setting in the real world and then execute the solution without recourse to their percepts.

## What are Rules to find good ordering in alpha-beta pruning

Following are some rules to find good ordering in alpha-beta pruning:

* Occur the best move from the shallowest node.
* Order the nodes in the tree such that the best nodes are checked first.
* Use domain knowledge while finding the best move. Ex: for Chess, try order: captures first, then threats, then forward moves, backward moves.
* We can bookkeep the states, as there is a possibility that states may repeat.

## Define the term backtracking search.

Backtracking search is used for a depth first search that chooses values for one variable at a time and backtracks when a variable has no legal values left to assign.

## When a problem is called commutative?

A problem is commutative if the order of application of any given set of actions has no effect on the outcome.

## What do you mean by minimum remaining values?

Choosing the variable with the fewest “legal” values is called the minimum remaining values heuristic. Otherwise called as “most constraint variable” or “fail first”

# Descriptive Question & Answer

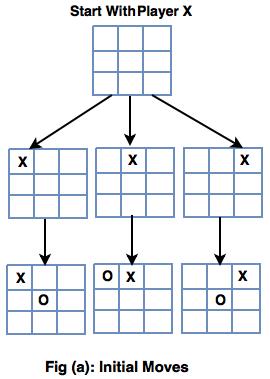
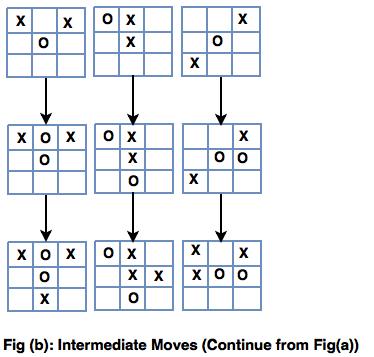
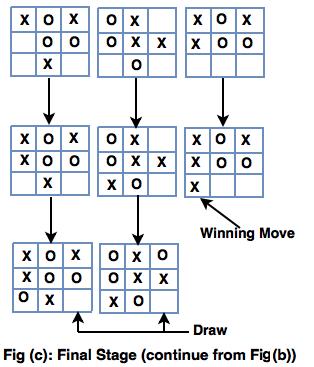
## Explain Minimax Algorithm and draw game tree for Tic Tac Toe Game.

#### Minimax Algorithm

* The Min-Max algorithm is generally used for a game consisting of two players such as tic-tac-toe, checkers, chess etc.
* All these games are logical games, so they can be described by set of rules. It is possible to determine the next available moves from a given point in the game.
* Consider that the game has two players MAX and MIN. MAX will move first and then MIN. A game can be defined as a kind of search problem with the following components.  
  **i. Initial State:** It includes the board position and indicates the move made by one of the two players.  
  **ii. Set of Operators:** It defines the legal moves that a player can make.  
  **iii. Terminal Move:** It determines the end of the game.  
  **iv. Utility function:** It gives a numeric value for the result of the game.   
  **For example:** In Chess or Tic-tac-toe game, the result is declared in the form of points such as +1 for win, -1 for loose, 0 for draw.

**Example: Tic Tac Toe Game.**

* The first Game State will show nine moves, one for each of the empty spaces on its board.
* Similarly, the next level of Game States will show eight moves and continues for each Game State.
* The computer evaluates each of its current possible moves by representing the game as a game tree. This also helps to determine whether it will result into a win or a loss. This game tree representation is shown in fig (a,b and c).

  
  
  
  
  
  
**Concepts for defining a Game Tree:**

* A Game Tree is a structure for organizing all possible (legal) game states by the moves which allow transition from one game state to the next.
* This structure helps the computer to evaluate which moves to make because, by traversing the game tree, a computer (program) can easily see the outcome of a move and  can decide whether to take it or not.

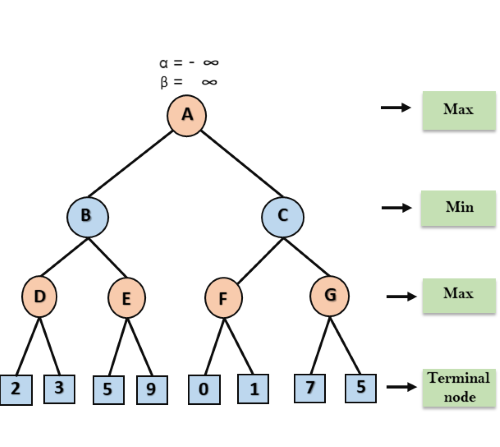
**The following states are used to represent a game tree.**  
  
**1. The board state:** This is an initial stage.  
**2. The current player:** It refers to the player who will be making the next move.   
**3. The next available moves:** For humans, a move involves placing a game token while the computer selects the next game state.  
**4. The game state:**It includes the grouping of the three previous concepts.   
**5.Final Game States**  
In final game states, AI should select the winning move in such a way that each move assigns a numerical value based on its board state.   
  
**The ranking should be given as:**  
  
a) Win: 1  
b) Draw: O  
c) Lose: -1

* It is important to consider the aspects related to winning with the highest ranking, losing to the lowest, and a draw between the two players.
* The Max part of Minimax algorithm states that the user has to select the move with the highest value.
* Final Game States are ranked on the basis of their status of   winning, losing or a draw.
* Ranking of Intermediate Game States is based on the turn of player to make available moves.
* If it's X's turn, set the rank to that of the maximum available move. If a move results into a win, X can take it.
* If it's O's turn, set the rank to that of the minimum available move. If a move results into a loss, X can avoid it.

## What is Alpha-Beta Pruning. Demonstrate with example

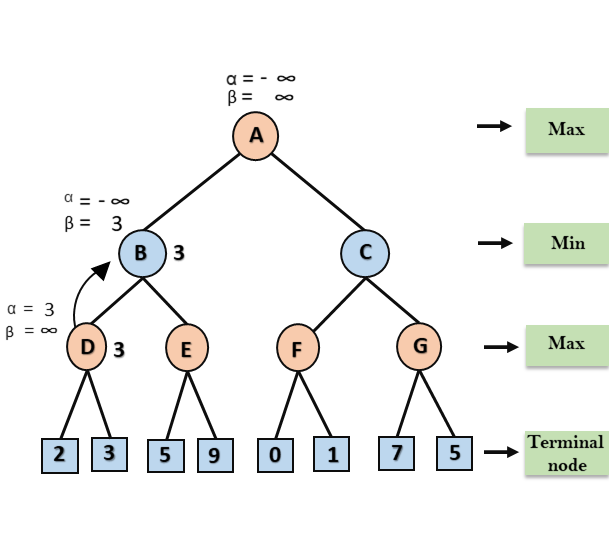
* Alpha-beta pruning is a modified version of the minimax algorithm. It is an optimization technique for the minimax algorithm.
* As we have seen in the minimax search algorithm that the number of game states it has to examine are exponential in depth of the tree. Since we cannot eliminate the exponent, but we can cut it to half. Hence there is a technique by which without checking each node of the game tree we can compute the correct minimax decision, and this technique is called **pruning**. This involves two threshold parameter Alpha and beta for future expansion, so it is called **alpha-beta pruning**. It is also called as **Alpha-Beta Algorithm**.
* Alpha-beta pruning can be applied at any depth of a tree, and sometimes it not only prune the tree leaves but also entire sub-tree.
* The two-parameter can be defined as:
  1. **Alpha:** The best (highest-value) choice we have found so far at any point along the path of Maximizer. The initial value of alpha is **-∞**.
  2. **Beta:** The best (lowest-value) choice we have found so far at any point along the path of Minimizer. The initial value of beta is **+∞**.
* The Alpha-beta pruning to a standard minimax algorithm returns the same move as the standard algorithm does, but it removes all the nodes which are not really affecting the final decision but making algorithm slow. Hence by pruning these nodes, it makes the algorithm fast.

**Step 1:** At the first step the, Max player will start first move from node A where α= -∞ and β= +∞, these value of alpha and beta passed down to node B where again α= -∞ and β= +∞, and Node B passes the same value to its child D.



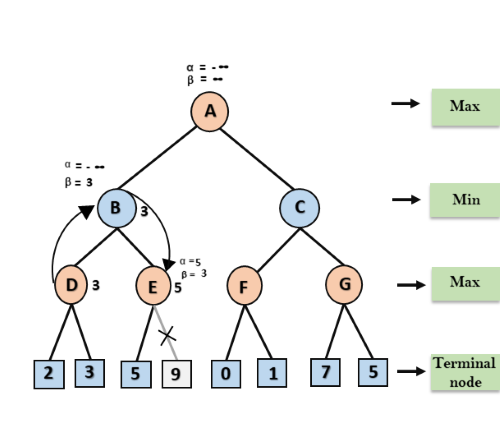
**Step 2:** At Node D, the value of α will be calculated as its turn for Max. The value of α is compared with firstly 2 and then 3, and the max (2, 3) = 3 will be the value of α at node D and node value will also 3.

**Step 3:** Now algorithm backtrack to node B, where the value of β will change as this is a turn of Min, Now β= +∞, will compare with the available subsequent nodes value, i.e. min (∞, 3) = 3, hence at node B now α= -∞, and β= 3.



In the next step, algorithm traverse the next successor of Node B which is node E, and the values of α= -∞, and β= 3 will also be passed.

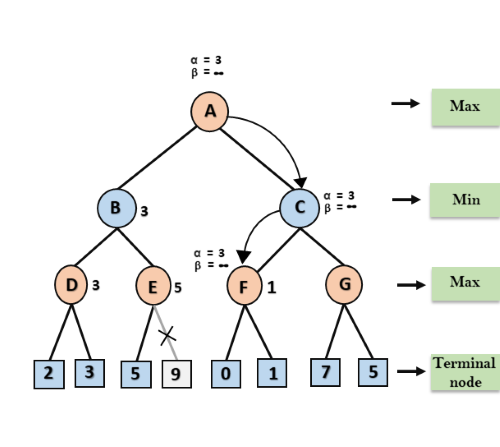
**Step 4:** At node E, Max will take its turn, and the value of alpha will change. The current value of alpha will be compared with 5, so max (-∞, 5) = 5, hence at node E α= 5 and β= 3, where α>=β, so the right successor of E will be pruned, and algorithm will not traverse it, and the value at node E will be 5.



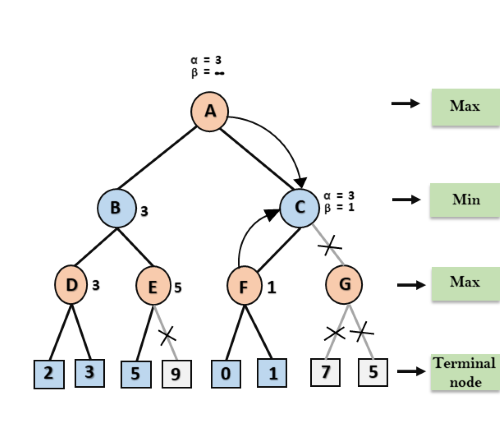
**Step 5:** At next step, algorithm again backtrack the tree, from node B to node A. At node A, the value of alpha will be changed the maximum available value is 3 as max (-∞, 3)= 3, and β= +∞, these two values now passes to right successor of A which is Node C.

At node C, α=3 and β= +∞, and the same values will be passed on to node F.

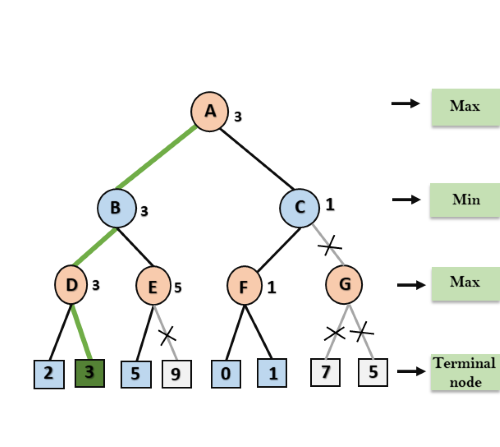
**Step 6:** At node F, again the value of α will be compared with left child which is 0, and max(3,0)= 3, and then compared with right child which is 1, and max(3,1)= 3 still α remains 3, but the node value of F will become 1.



**Step 7:** Node F returns the node value 1 to node C, at C α= 3 and β= +∞, here the value of beta will be changed, it will compare with 1 so min (∞, 1) = 1. Now at C, α=3 and β= 1, and again it satisfies the condition α>=β, so the next child of C which is G will be pruned, and the algorithm will not compute the entire sub-tree G.



**Step 8:** C now returns the value of 1 to A here the best value for A is max (3, 1) = 3. Following is the final game tree which is the showing the nodes which are computed and nodes which has never computed. Hence the optimal value for the maximizer is 3 for this example.



## Write a short note on : Types of Games in AI

|  |  |  |
| --- | --- | --- |
|  | **Deterministic** | **Chance Moves** |
| **Perfect information** | Chess, Checkers, go, Othello | Backgammon, monopoly |
| **Imperfect information** | Battleships, blind, tic-tac-toe | Bridge, poker, scrabble, nuclear war |

* **Perfect information:** A game with the perfect information is that in which agents can look into the complete board. Agents have all the information about the game, and they can see each other moves also. Examples are Chess, Checkers, Go, etc.
* **Imperfect information:** If in a game agents do not have all information about the game and not aware with what's going on, such type of games are called the game with imperfect information, such as tic-tac-toe, Battleship, blind, Bridge, etc.
* **Deterministic games:** Deterministic games are those games which follow a strict pattern and set of rules for the games, and there is no randomness associated with them. Examples are chess, Checkers, Go, tic-tac-toe, etc.
* **Non-deterministic games:** Non-deterministic are those games which have various unpredictable events and has a factor of chance or luck. This factor of chance or luck is introduced by either dice or cards. These are random, and each action response is not fixed. Such games are also called as stochastic games.  
  Example: Backgammon, Monopoly, Poker, etc.

## Why is game theory important to AI?

Game theory, developed by American mathematician Josh Nash, is essential to AI because it plays an underlying role in how these smart algorithms improve over time.

At its most basic, AI is about algorithms that are deployed to find solutions to problems. Game theory is about players in opposition trying to achieve specific goals. As most aspects of life are about competition, game theory has many meaningful real-world applications.

These problems tend to be dynamic. Some game theory problems are natural candidates for AI algorithms. So, whenever game theory is applied, multiple AI agents that interact with each other will only care about utility to itself.

Data scientists within this space should be aware of the following games:

* Symmetric vs. asymmetric
* Perfect vs. imperfect information
* Cooperative vs. non-cooperative
* Simultaneous vs. sequential
* Zero-sum vs. non-zero-sum

# MCQs & answers

**1.** General games involves

(a) Single-agent

(b) Multiagent

(c) Neither single-agent nor multiagent

(d) Only single-agent and multiagent

**2.** Adversarial search problems uses

(a) Competitive environment

(b) Cooperative environment

(c) Neither competitive nor cooperative

environment

(d) Only competitive and cooperative

environment

**3.** Which search is equal to minimax search but

eliminates the branches that can’t influence the

final decision?

(a) Depth-first search

(b) Breadth-first search

(c) Alpha–beta pruning

(d) None of the above

**4.** Which values are independent in minimax

search algorithm?

(a) Pruned leaves *x* and *y*

(b) Every states are dependent

(c) Root is independent

(d) None of the above

**5.** To which depth does the alpha–beta pruning

can be applied?

(a) 10 states

(b) 8 states

(c) 6 states

(d) Any depth

**Answers**

1. **(d) 2. (a) 3. (c) 4. (a) 5. (d)**