SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107
Accredited by NAAC-UGC with 'A' Grade
Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

Department of AI &DS

Course Name – 23ADT201 ARTIFICIAL INTELLIGENCE

II Year / III Semester

UNIT 3

GAME THEORY

Topic:OPTIMAL DECISION MAKING IN GAMES

OPTIMAL DECISION IN GAMES

Case Study: AI in Rock-Paper-Scissors

Background

- **Game**: Rock-Paper-Scissors, a simple hand game where each of the three possible moves (rock, paper, scissors) beats one of the other moves, loses to another, and ties with the same move.
- **AI Challenge**: Developing an AI that can predict the opponent's moves and make optimal decisions to maximize wins.

OPTIMAL DECISION IN GAMES

- Optimal decision-making in games is a fundamental challenge in the field of Artificial Intelligence (AI).
- Various techniques and approaches have been developed to enable AI agents to make optimal decisions in games. Here are some key concepts and methods:

OPTIMAL DECISION IN GAMES

MIN MAX ALGORITHM:

Concept of Minimax Algorithm

1. **Start with MAX:** The game begins with MAX (player 1), who makes the first move based on the current board state.
2. **Expanding Nodes:** The game tree is expanded to a certain depth to explore possible future moves.
3. **Evaluation Function:** At each leaf node (end of a branch in the game tree), an evaluation function is applied to assess the desirability of that outcome.
4. **Backing Up Values:** Values from the leaf nodes are "backed up" through the tree to non-leaf nodes, eventually determining the best move for the root node.
5. **MIN and MAX Nodes:** At MIN (player 2) nodes, the algorithm chooses the minimum value among children; at MAX nodes, it chooses the maximum value.

OPTIMAL DECISION IN GAMES

Properties of Minimax

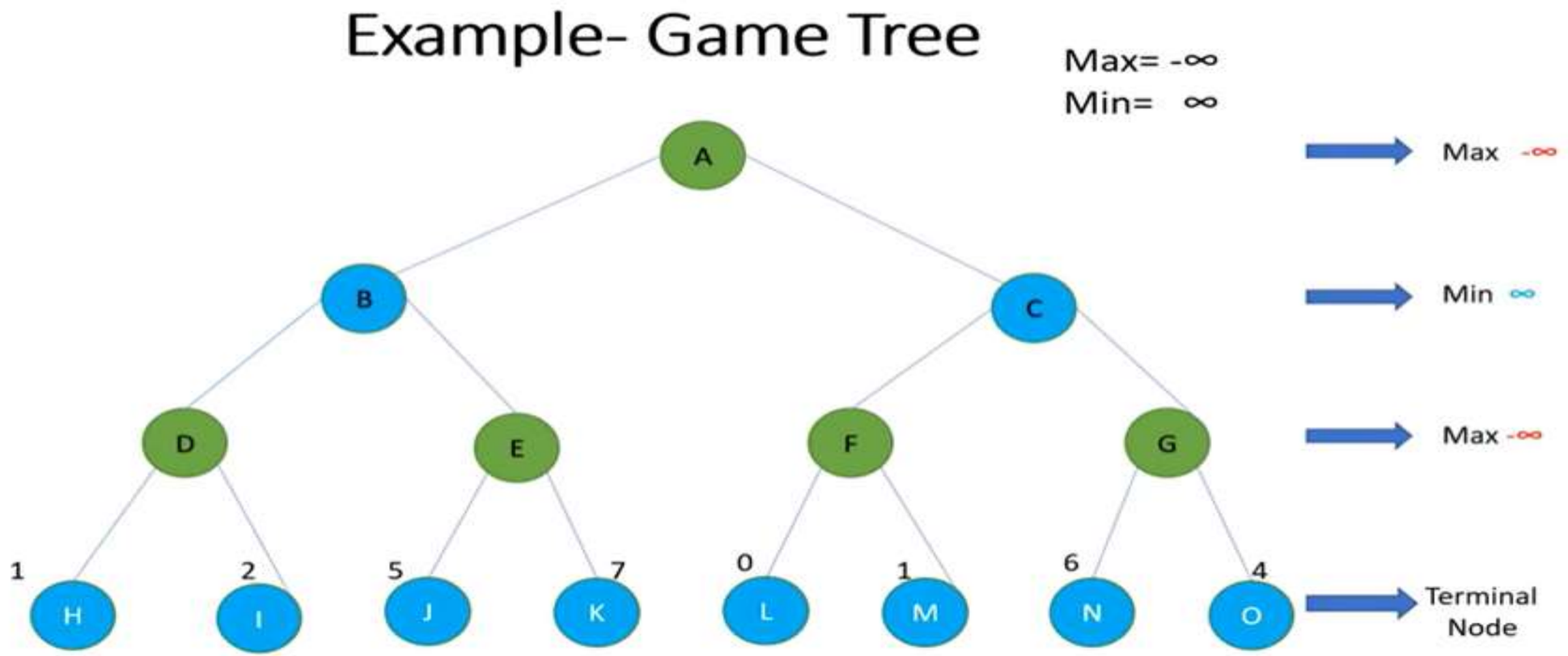
1. **Complete Solution:** Minimax provides a comprehensive solution for games with finite trees, meaning it explores all possible moves and outcomes.
2. **Optimal Strategy:** It guarantees an optimal strategy if both players play perfectly.
3. **Time Complexity:** The time complexity is O(bm)O(b^m)O(bm), where bbb is the branching factor (possible moves per turn) and mmm is the depth of the game tree.
4. **Space Complexity (Full Tree):** The space complexity is O(bm)O(b^m)O(bm) if the algorithm generates all successors at once.
5. **Space Complexity (DFS):** The space complexity can be reduced to O(m)O(m)O(m) using depth-first search, which generates successors one at a time

OPTIMAL DECISION IN GAMES

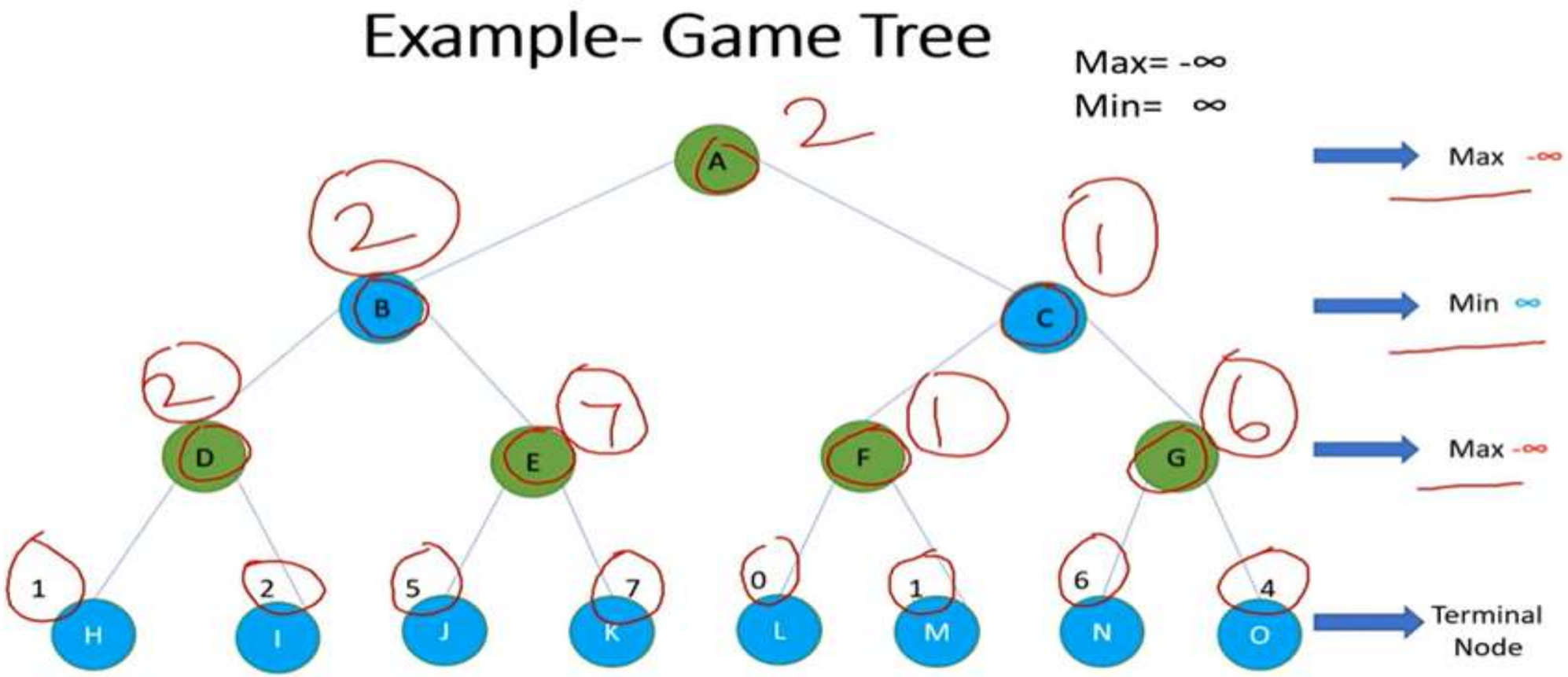
Minimax Algorithm

- Minimax is a kind of backtracking algorithm that is used in game theory to find the optimal move for a player .
- It is widely used in two player turn-based games.
  Example: Chess, Checkers, tic-tac-toe.
- In Minimax the two players are called MAX and MIN.
- MAX ⟶ highest value
- MIN ⟶ lowest value
- The minimax algorithm proceeds all the way down to the terminal node of the tree, then backtrack the tree as the recursion.

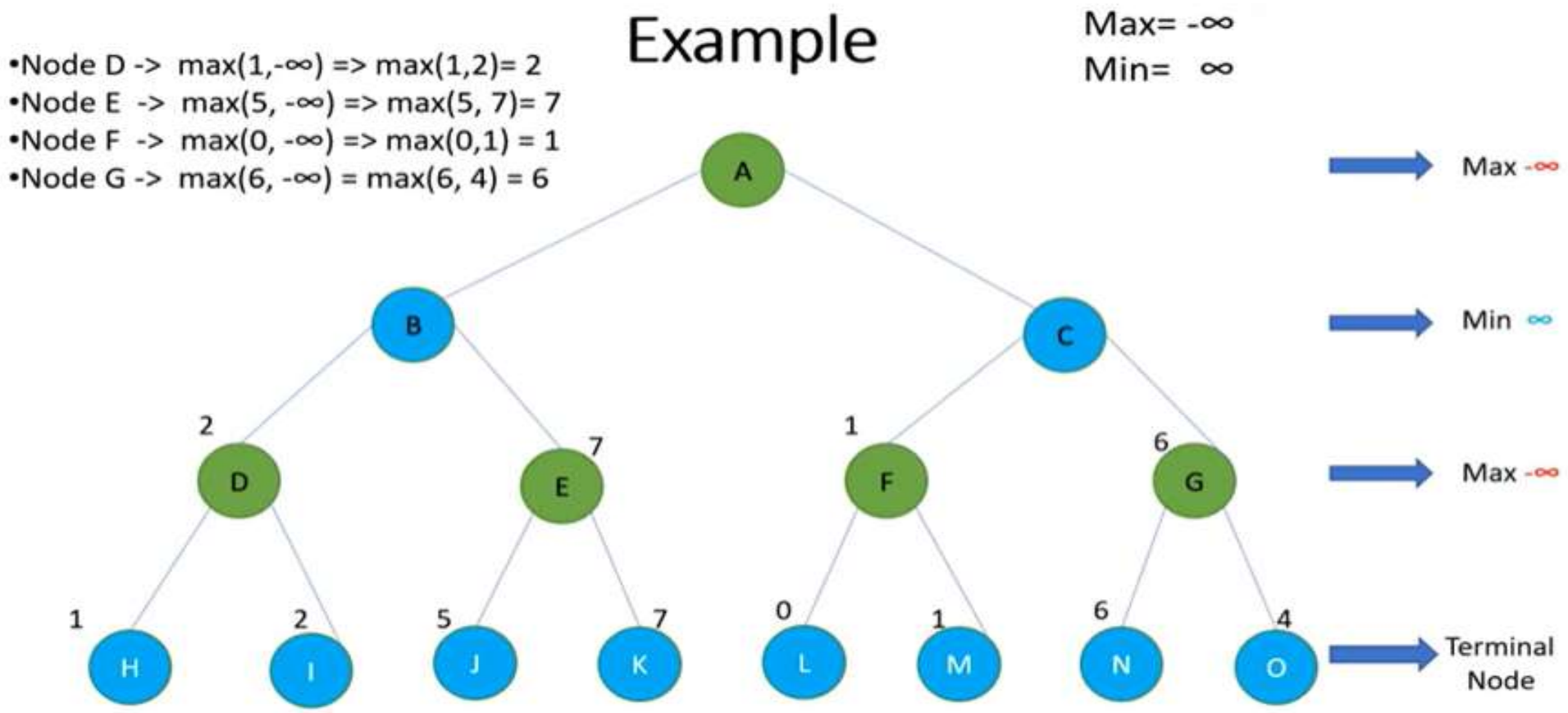
OPTIMAL DECISION IN GAMES



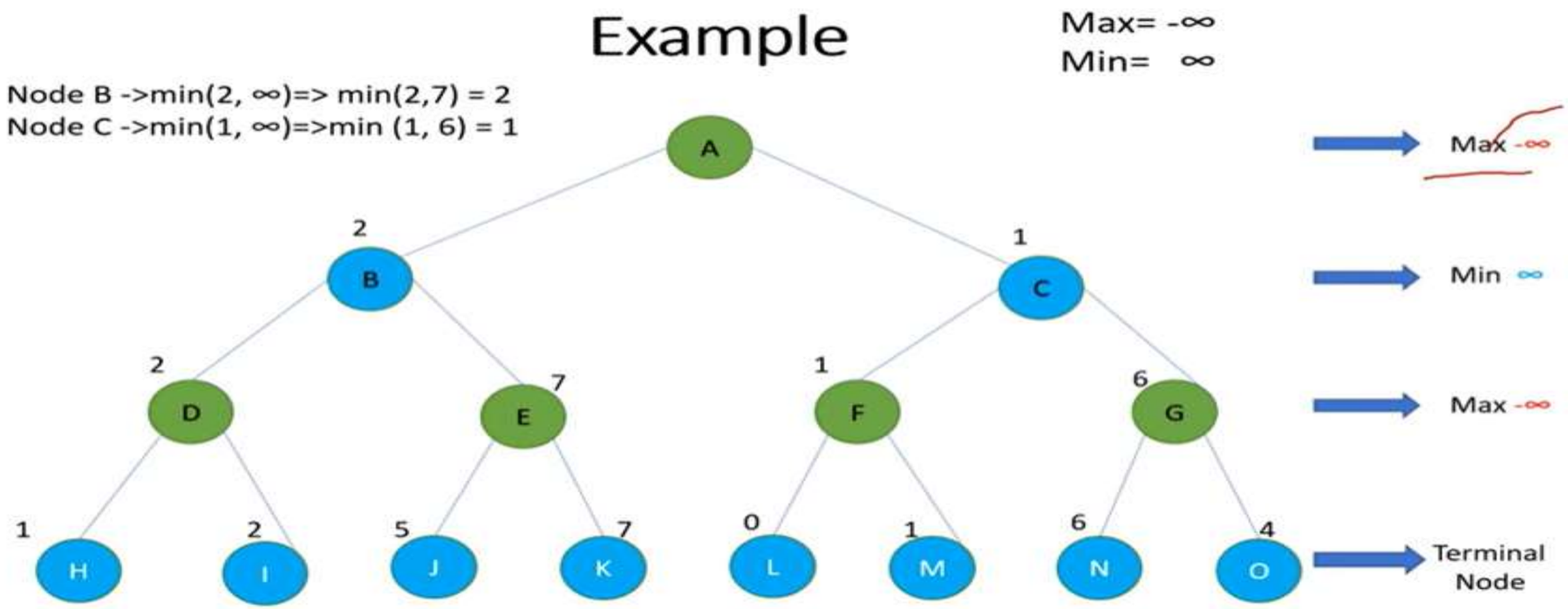
OPTIMAL DECISION IN GAMES



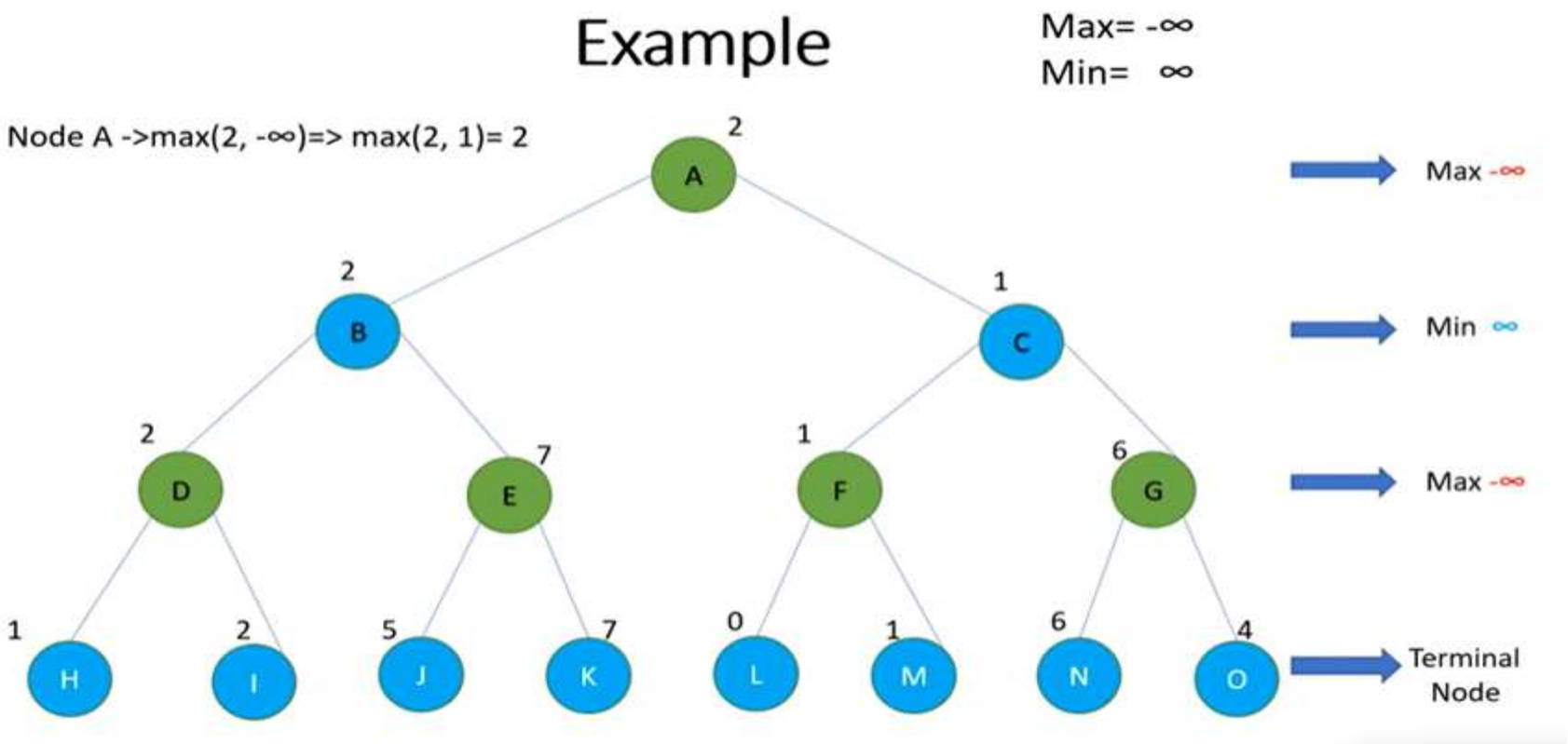
OPTIMAL DECISION IN GAMES



OPTIMAL DECISION IN GAMES



OPTIMAL DECISION IN GAMES



OPTIMAL DECISION IN GAMES

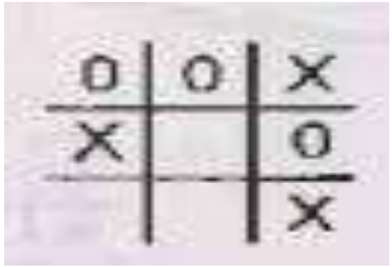
Game Playing with Minimax - Tic-Tac-Toe Example

1. **Two Players:** The game involves two players, MIN and MAX, with MAX starting the game.
2. **Initial Moves:** MAX has nine possible initial moves, corresponding to the nine empty squares.
3. **Alternating Turns:** Players alternate turns until the game reaches a terminal state, such as three in a row or a filled board.
4. **Utility Values:** Each leaf node in the game tree has a utility value indicating the outcome for MAX: positive for a win, negative for a loss, and zero for a draw.
5. **MAX's Strategy:** MAX uses the game tree to determine the optimal move by selecting the branch with the highest utility value, given that MIN will also play optimally.

OPTIMAL DECISION IN GAMES

Consider game of tic-tac-toe with the initial state -

STEP 1



Start: MAX (player 1) moves (MAX is making X)

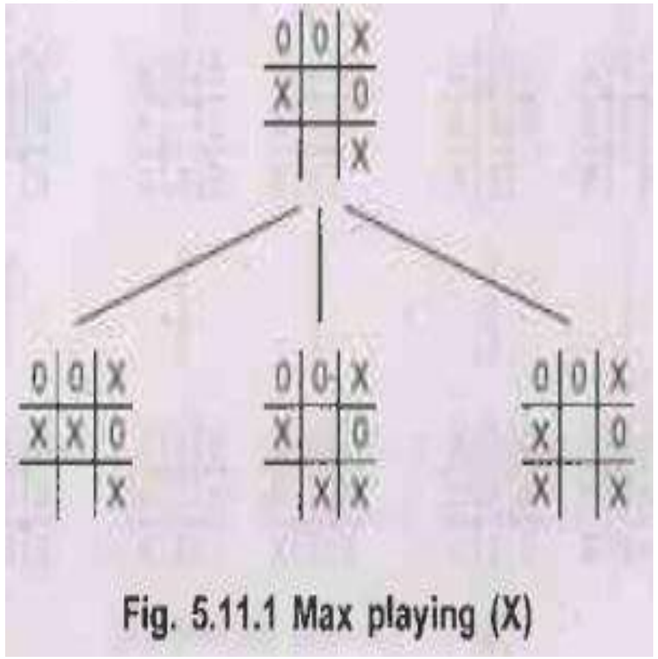


Fig. 5.11.1 Max playing (X)

STEP 2:Next: MIN (player 2) moves.

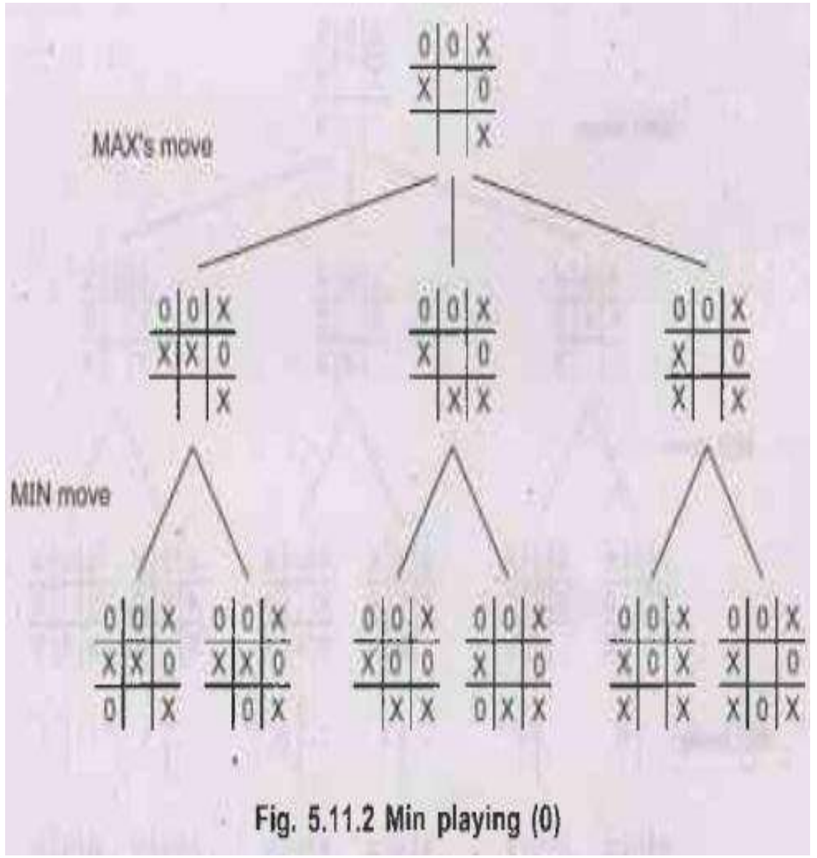


Fig. 5.11.2 Min playing (0)

OPTIMAL DECISION IN GAMES

Step 3: Again: MAX's moves

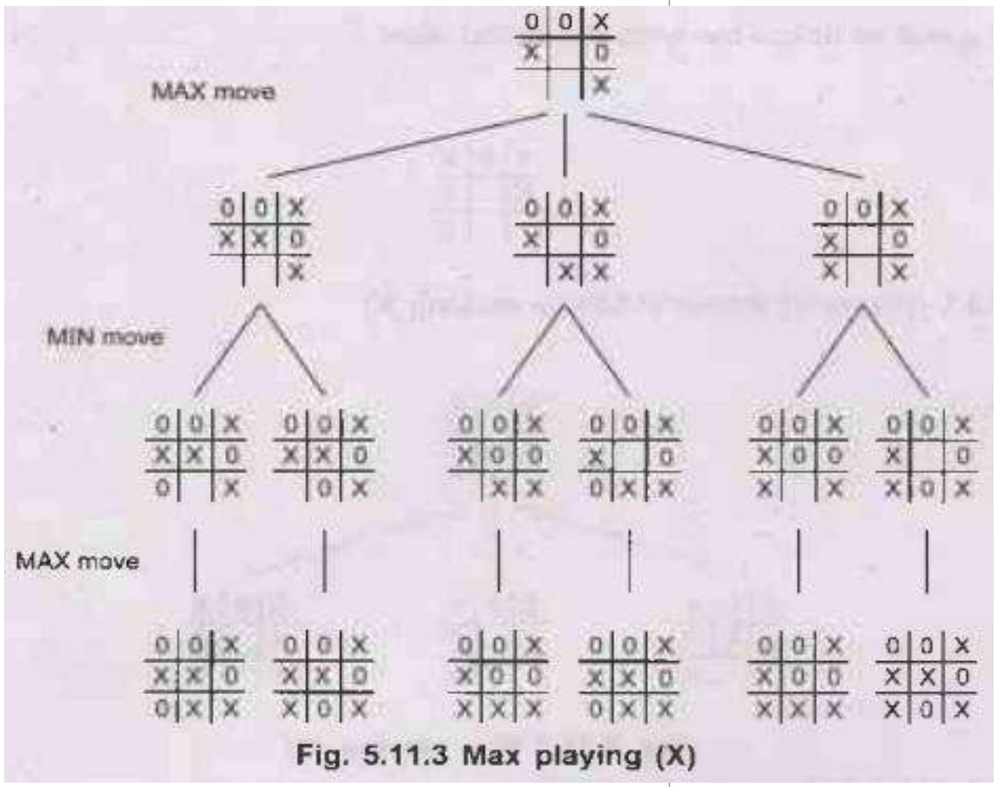


Fig. 5.11.3 Max playing (X)

Step 4: +1' for a win, '0' for a draw. Criteria '+ 1' for a win, '0' for a draw.

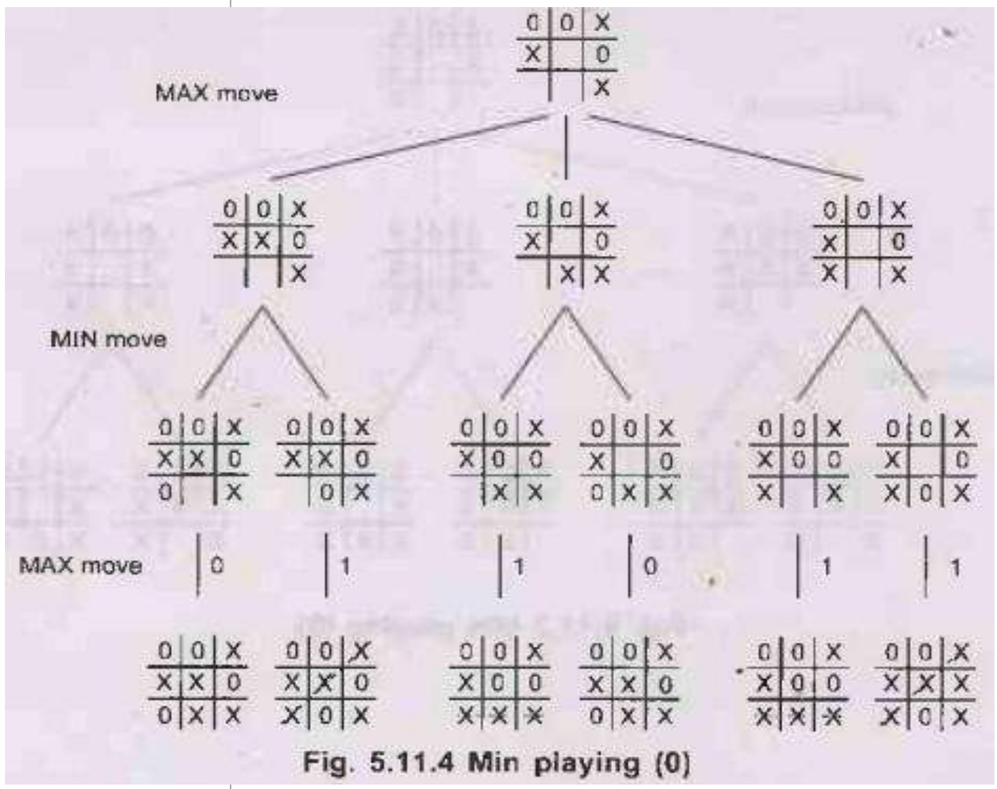


Fig. 5.11.4 Min playing (0)

OPTIMAL DECISION IN GAMES

Step 5: Level by level, on the basis of opponents turn UP: One level

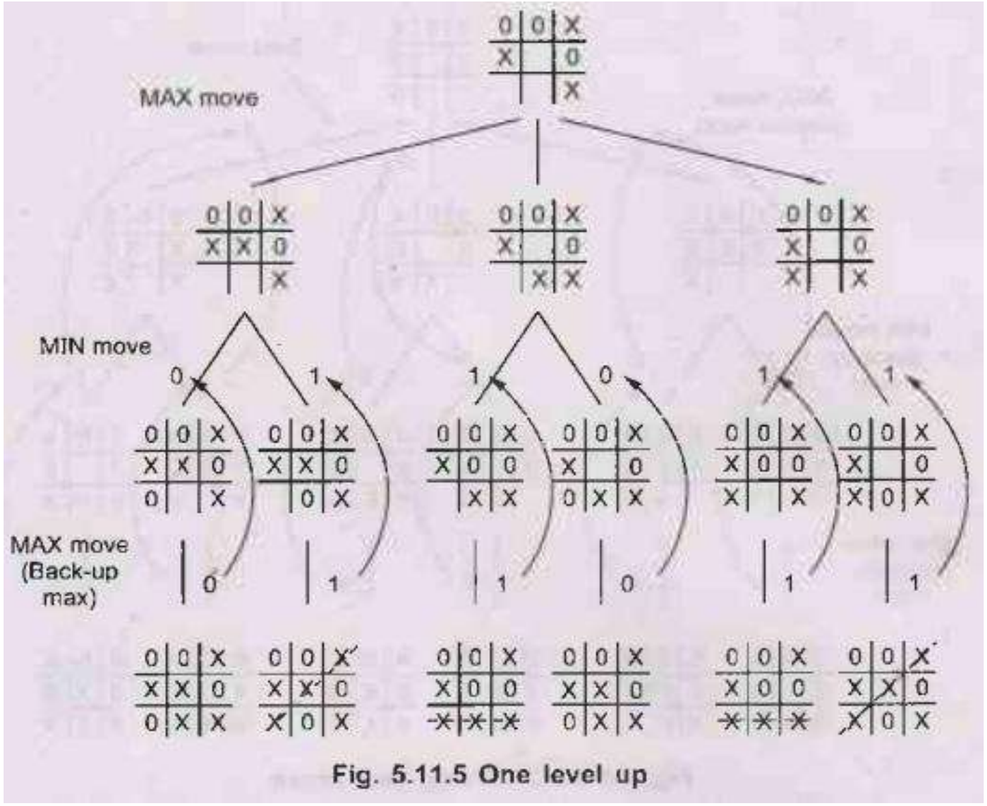


Fig. 5.11.5 One level up

Step 6: UP: Two levels

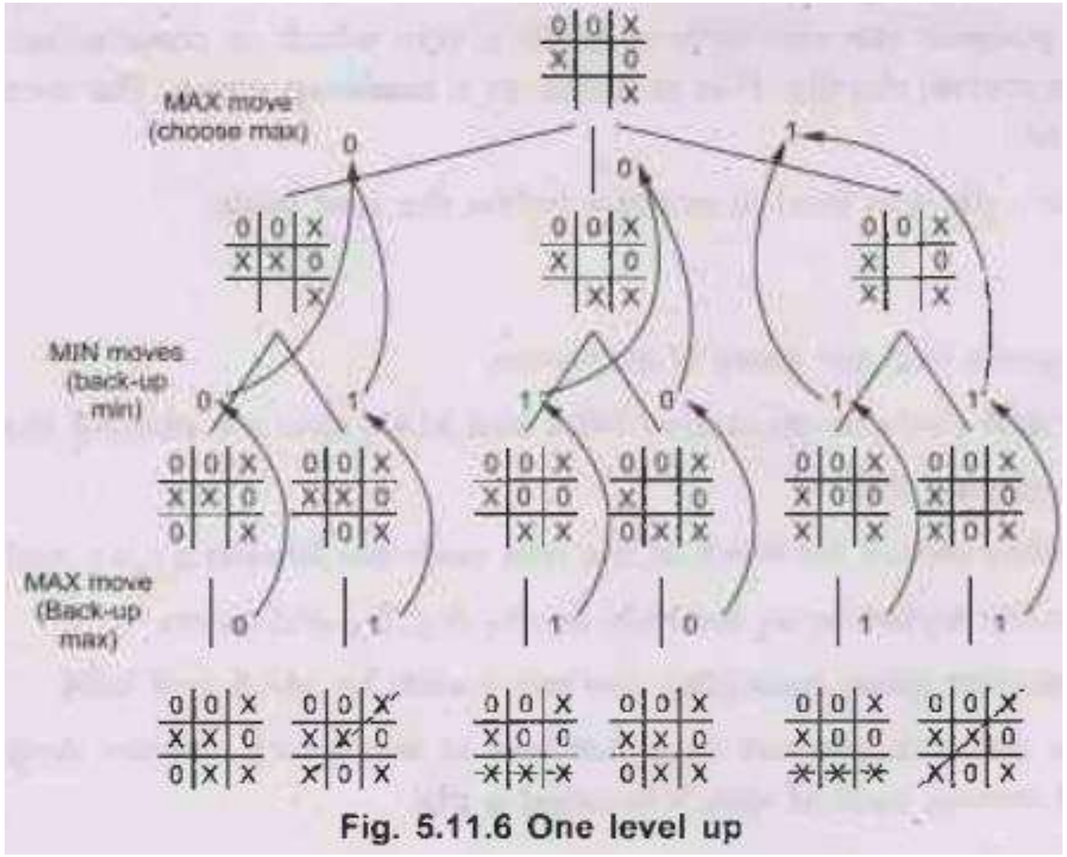


Fig. 5.11.6 One level up

OPTIMAL DECISION IN GAMES

Step 7: Choose best move which is maximum

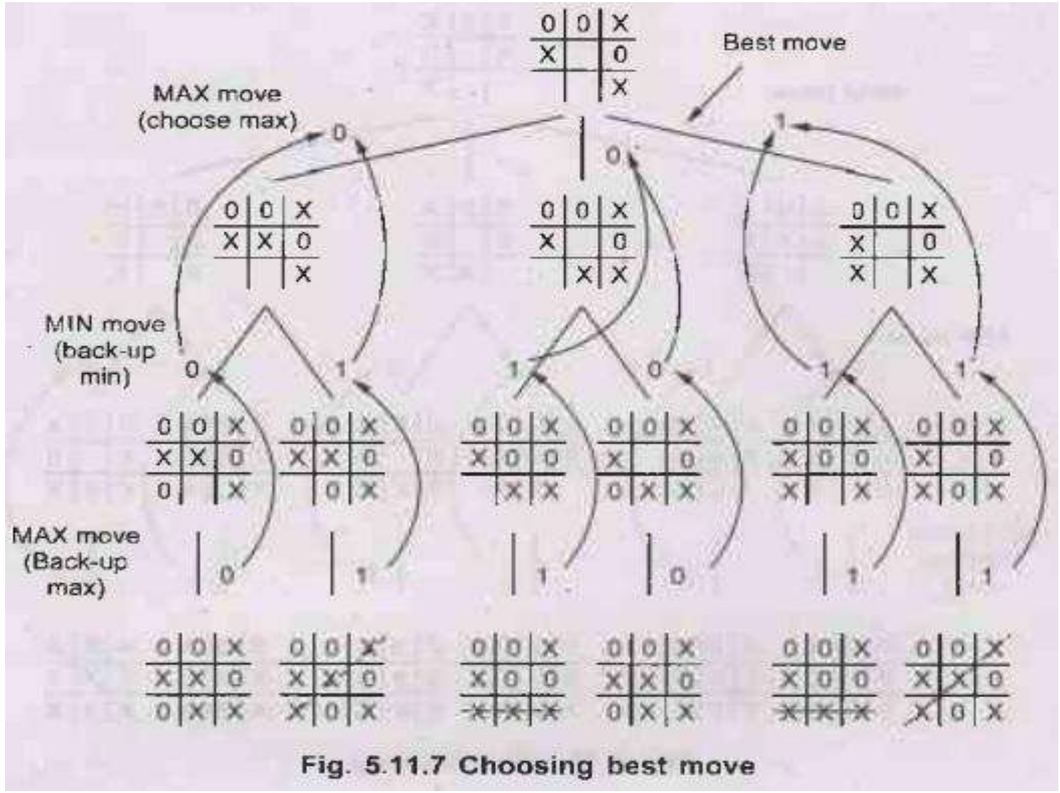


Fig. 5.11.7 Choosing best move

OPTIMAL DECISION IN GAMES

OPTIMAL DECISION MAKING IN MULTIPLAYER GAMES

Mini-Max Algorithm for Playing Multiplayer Games

1. There are many multiplayer games like cricket, football, etc.

2. The multiplayer game can be played with minimax concept.

3. Here the single value for each node is replaced with a vector of values. For example, in a three-player game with players A, B, and C a vector (VA , VB , VC) is associated with each node.

4. For terminal states, this vector gives the utility of the state from each player's viewpoint. (In two player, zero-sum games, the two-element vector can be reduced to a single value because the values are always opposite.

5. In this implementation UTILITY function return a vector of utilities.

OPTIMAL DECISION IN GAMES

6. For non-terminal states we can calculate utility function value as explain below - Consider following diagram,

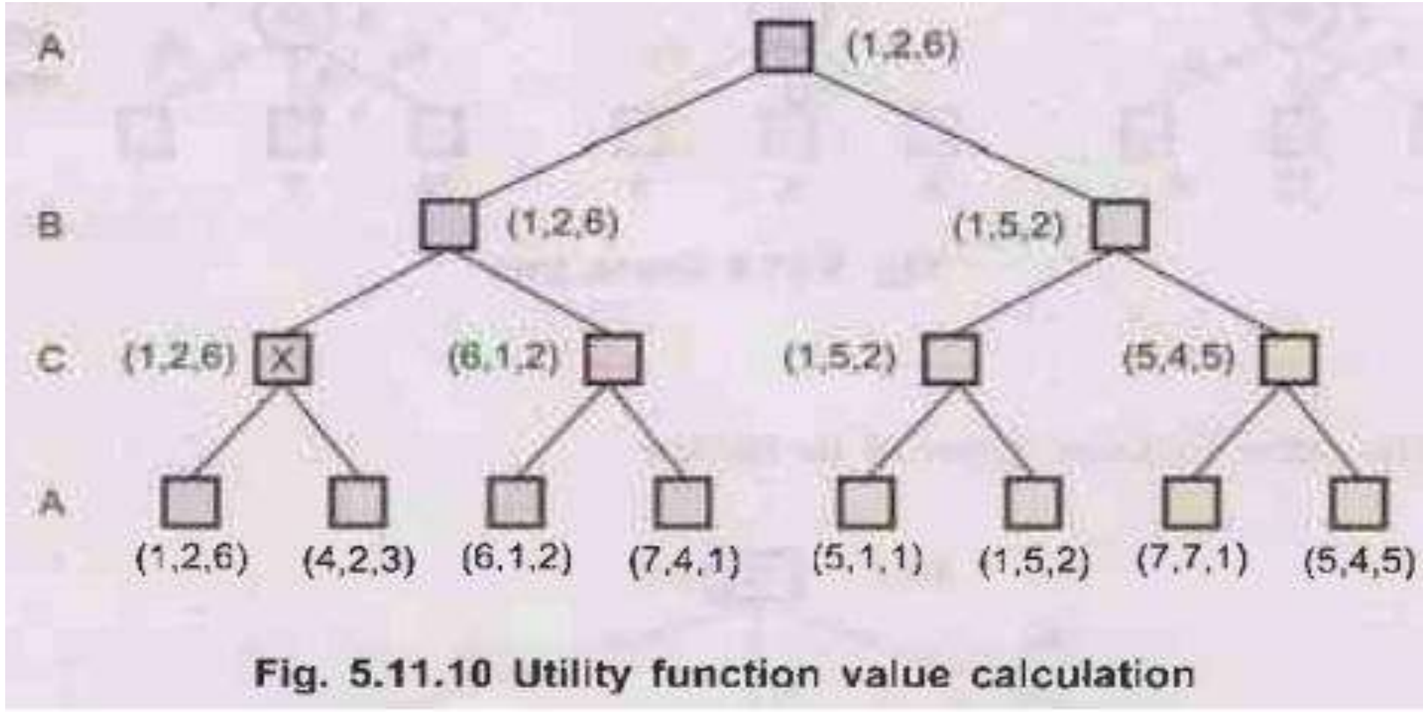


Fig. 5.11.10 Utility function value calculation

OPTIMAL DECISION IN GAMES

7. For non-terminal states vector values should be calculated as explained. Consider the node marked X in the game tree shown in above diagram. In this state, player C choose what to do. The two choices lead to terminal states with utility vector (VA = 1, VB = 2, VC = 6) and (VA = 4, VB = 2, VC = 3). Since 6 is bigger than 3, C should choose the first move. This means that if state X is reached subsequent play will lead to a terminal state with utilities (VA = 1, VB = 2, VC = 6). Hence the backed-up value of X is this vector.

8. In general, the backed-up value of a node 'n' is the utility vector of whichever successor has the highest value for the player choosing at 'n'.

9. Multiplayer games usually involve alliance, whether formal or informal, among the players. Alliances are made and broken as the game proceeds.

OPTIMAL DECISION IN GAMES

THANK YOU