CS 561 Artificial Intelligence
Lecture

Adversarial Search

Rashmi Dutta Baruah

Dept. of Computer Science & Engineering

IIT Guwahati

Outline

- Introduction : games
- Minimax algorithm
- Alpha-Beta algorithm

Games

- Game Theory (from Economics)
  - Game is a formal model representing strategic interactions (conflicting or cooperating) among multiple agents.
  - Game with only one player : decision problem
  - Game theory: study of such models and applied to many fields including political science, psychology, and computer science.
- Games in AI
  - Adversarial search problems: Games
    - competitive multi-agent environment where, while planning, an agent needs to consider the actions of other agents whose goals are conflicting.
    - deterministic, two-player, turn-taking, zero-sum games of perfect information (the utility values at the end of the game are always equal and opposite, eg. chess)

Games

- Strategic and Extensive form of games
- Strategic form (also called normal form)
  - usually represented by a matrix that shows the players, strategies, and pay offs
  - players choose a strategy simultaneously (without knowing the other player's strategy).
  - players choose the strategy only once and than the game is over, such games are also called static (or simultaneous move) games
- Extensive form (also called game trees)
  - completely describes how the game is played over time.

Games

- Example: Prisoner’s dilemma
- Two players, “prisoners” 1, 2.
- Each is questioned by authorities separately and without communication.
- Each has two strategies.
  - Prisoner 1: Defect, Cooperate
  - Prisoner 2: Defect, Cooperate
  - consequences quantified in prison years (payoff 0,1,2,3)

We will consider games in extensive forms next.

Strategic or Normal Form

Prisoner2

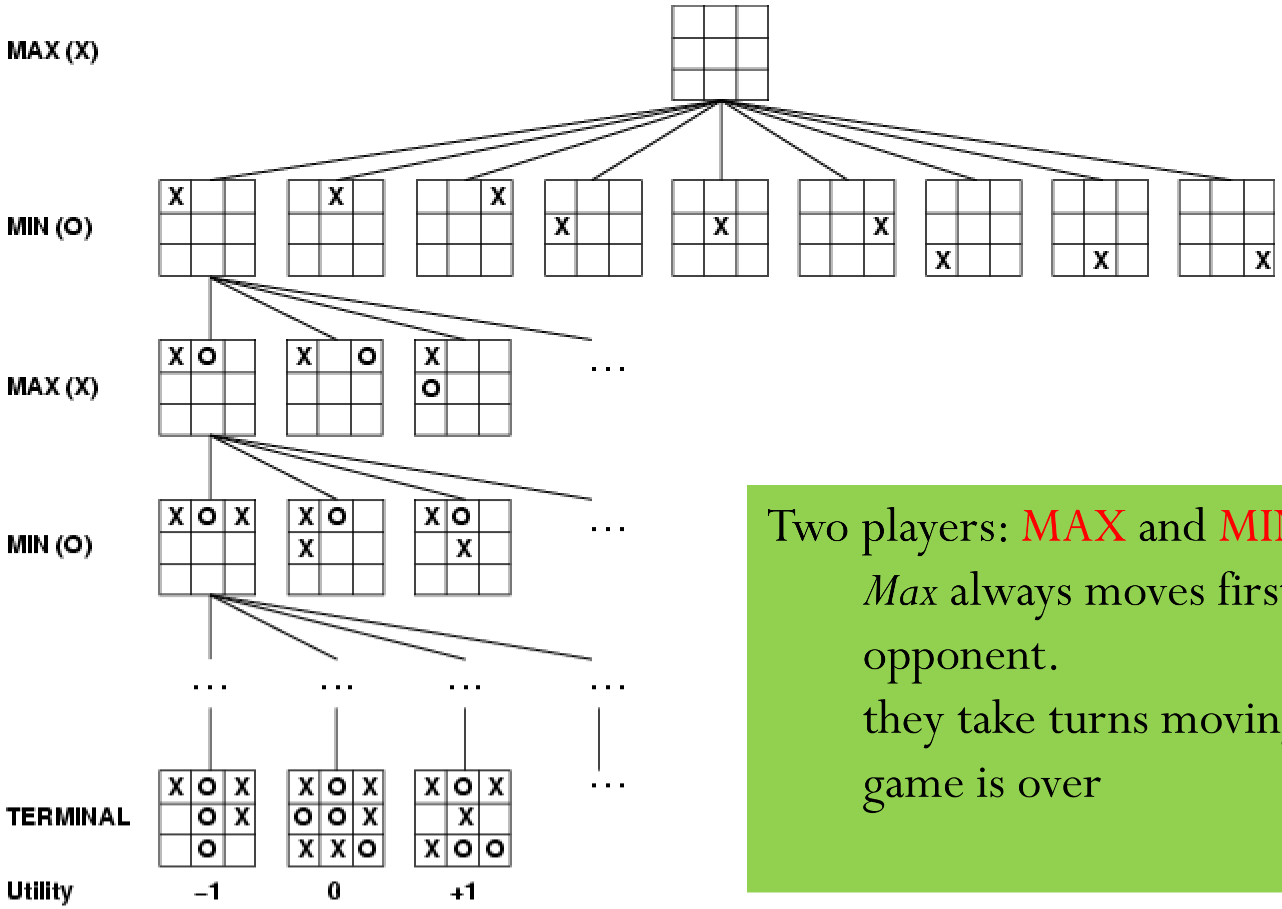
Prisoner1		Cooperate (C)	Defect (D)
	Cooperate (C)	2,2	0,3
	Defect (D)	3,0	1,1

Games

- Game can be formulated as search problem
  - S_0 : initial state
  - $PLAYER(s)$: which player has the move in a state
  - $ACTIONS(s)$: returns the set of legal moves
  - $RESULT(s, a)$: transition model, defines the result of a move
  - $TERMINAL - TEST(S)$: true when the game is over and false otherwise. States where game has ended are called terminal states.
  - $UTILITY(s, p)$: gives final numeric value for a game that ends in terminal state s for a player p. Example, in chess win +1, loss 0, draw 1/2 .

Games

- Game tree: nodes are game states, edges are moves, and defined by initial state, *ACTIONS* function and *RESULT* function.



Two players: MAX and MIN

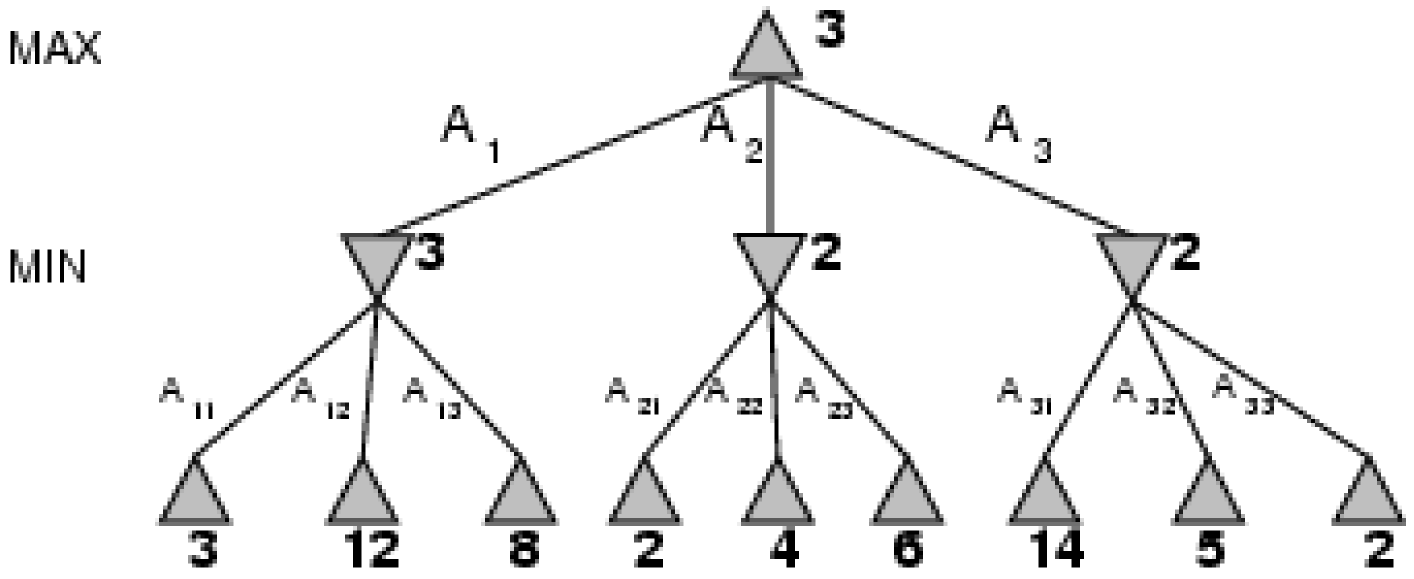
Max always moves first. *Min* is the opponent.

they take turns moving until the game is over

Minimax Algorithm

- Minimax: algorithm to find optimal strategy
- Idea: choose move to a state with highest minimax value

An action by one player is called a *ply*, two ply (a action and a counter action) is called a *move*.



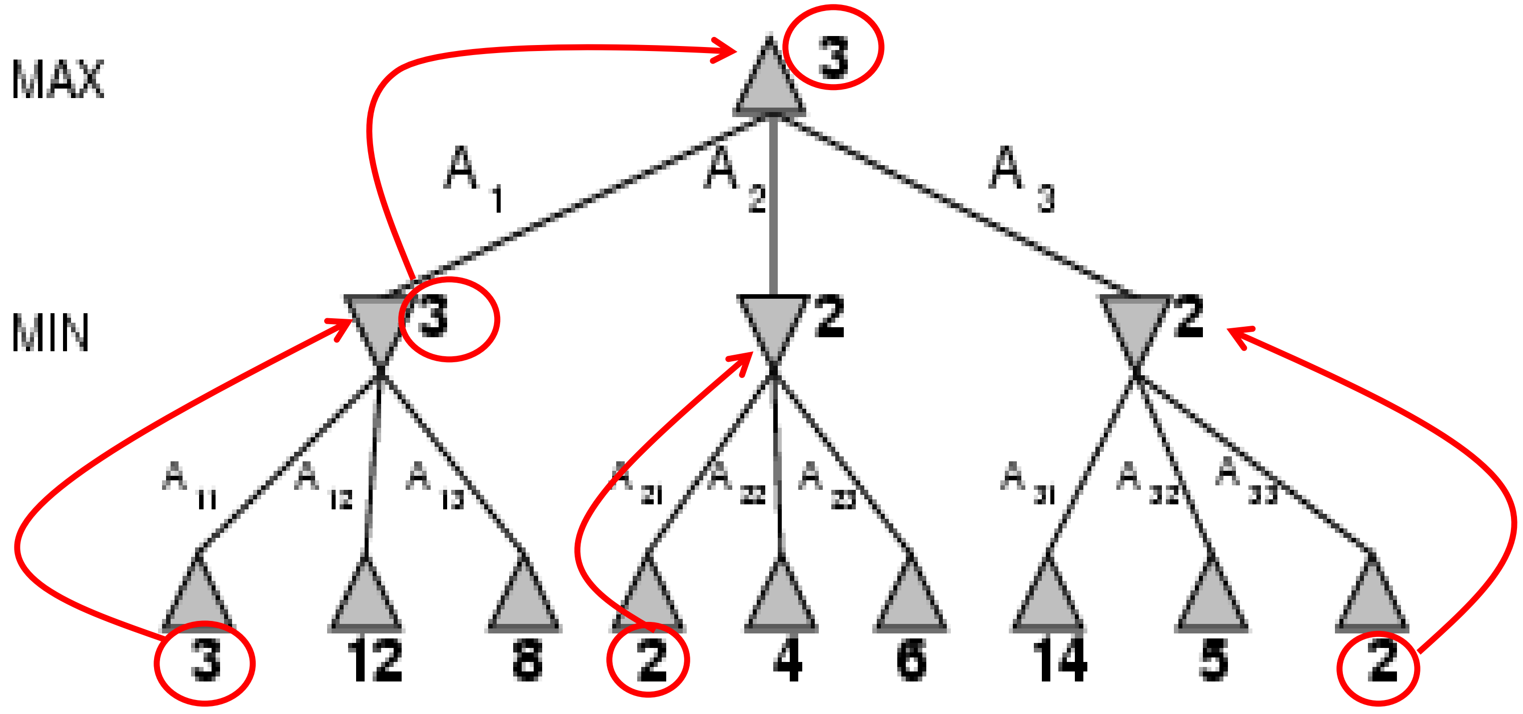
Minimax Algorithm

- Generate the game tree down to the terminal nodes.
- Apply the utility function to the terminal nodes.
- For a set **S** of sibling nodes, pass up to the parent
  - the lowest value in **S** if the siblings are MAX nodes
  - the largest value in **S** if the siblings are MIN nodes
- Recursively do the above, until the backed-up values reach the initial state.
- The value of the initial state is the minimum score for MAX.

minimax value of a node is the utility (for MAX) of being in the corresponding state.

$$MINIMAX(s) = \begin{cases} UTILITY(s) \; if \; TRERMINAL-TEST(s) \\ max_{a \in ACTIONS(s)} MINIMAX(RESULT(s,a)) \; if \; PLAYER\,(s) = MAX \\ min_{a \in ACTIONS(s)} MINIMAX(RESULT(s,a)) \; if \; PLAYER\,(s) = MIN \end{cases}$$

Minimax Algorithm



In this game Max's best move is A_1, because he is guaranteed a score of at least 3.

Properties of minimax

- Complete? Yes (if tree is finite)
- Optimal? Yes (against an optimal opponent)
- Time complexity? O(b^m)
- Space complexity? O(bm) (depth-first exploration)

- For chess, b ≈ 35, m ≈100 for "reasonable" games
  → exact solution completely infeasible

Alpha-Beta pruning

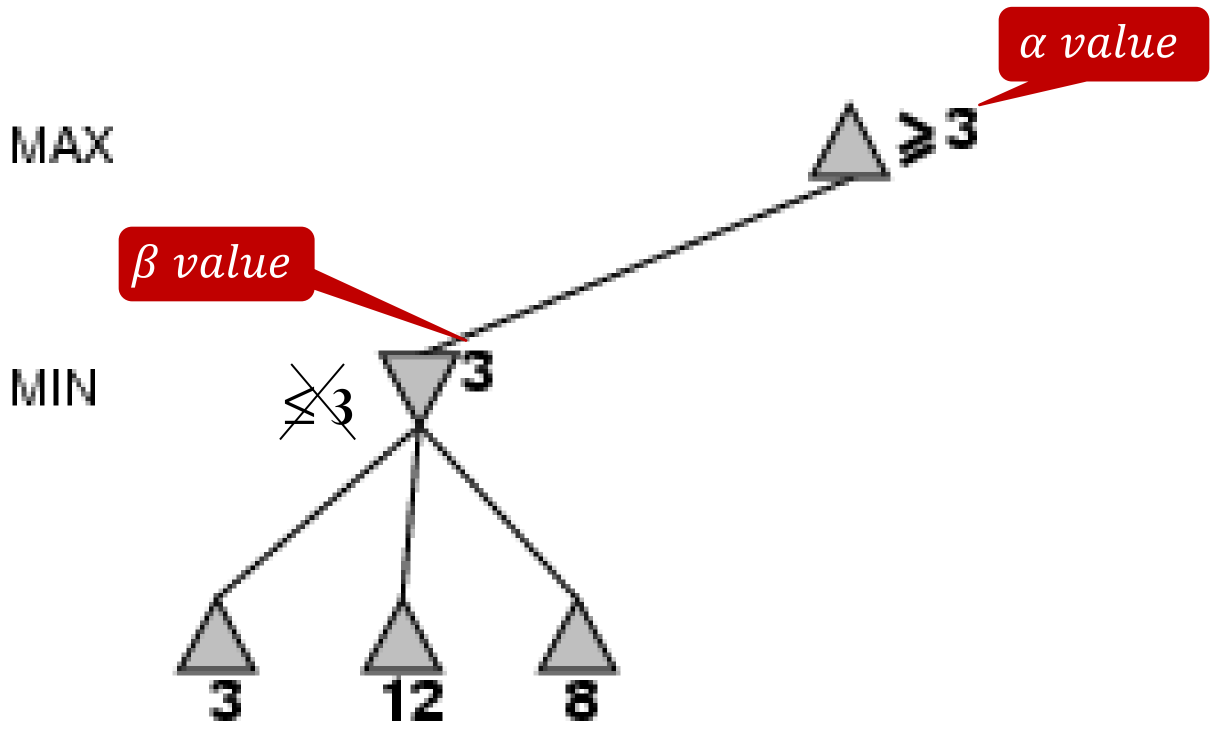
- The exponent part of the time complexity can be reduced to half $O(b^{m/2})$.
- Idea: instead of looking at every node, prune away the branches of the tree that cannot possibly influence the final decision.
- Alpha-beta pruning gets its name from the following two parameters that describe bounds on the backed-up values that appear anywhere along the path:

α = the value of the best (i.e., highest-value) choice found so far at any choice point along the path for MAX
β = the value of the best (i.e. lowest-value) choice found so far at any choice point along the path for MIN

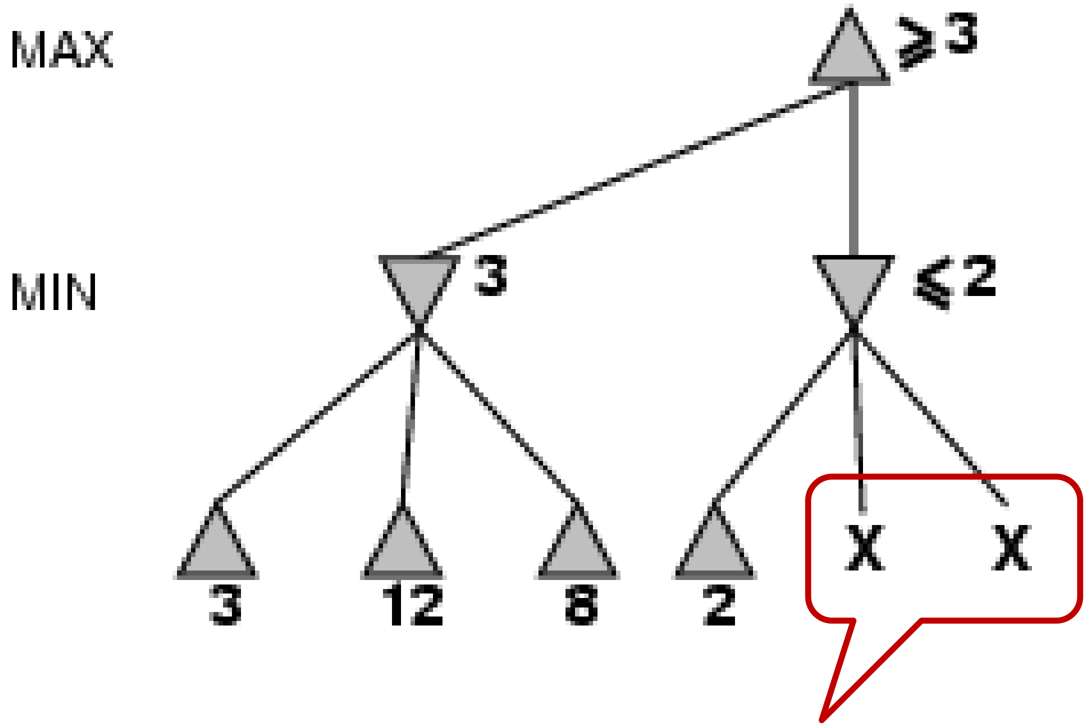
Alpha-Beta pruning

- Alpha-Beta search updates the values of α and β as it goes along and prunes the remaining branches at a node (i.e., terminates the recursive call)
  - when the value of the current node is worse than the current α and β value for MAX or MIN, respectively.
- While the backed-up values are updated we note that:
  - the α values of MAX nodes (including the start node) can never decrease.
  - the β values of MIN nodes can never increase.

α-β pruning example

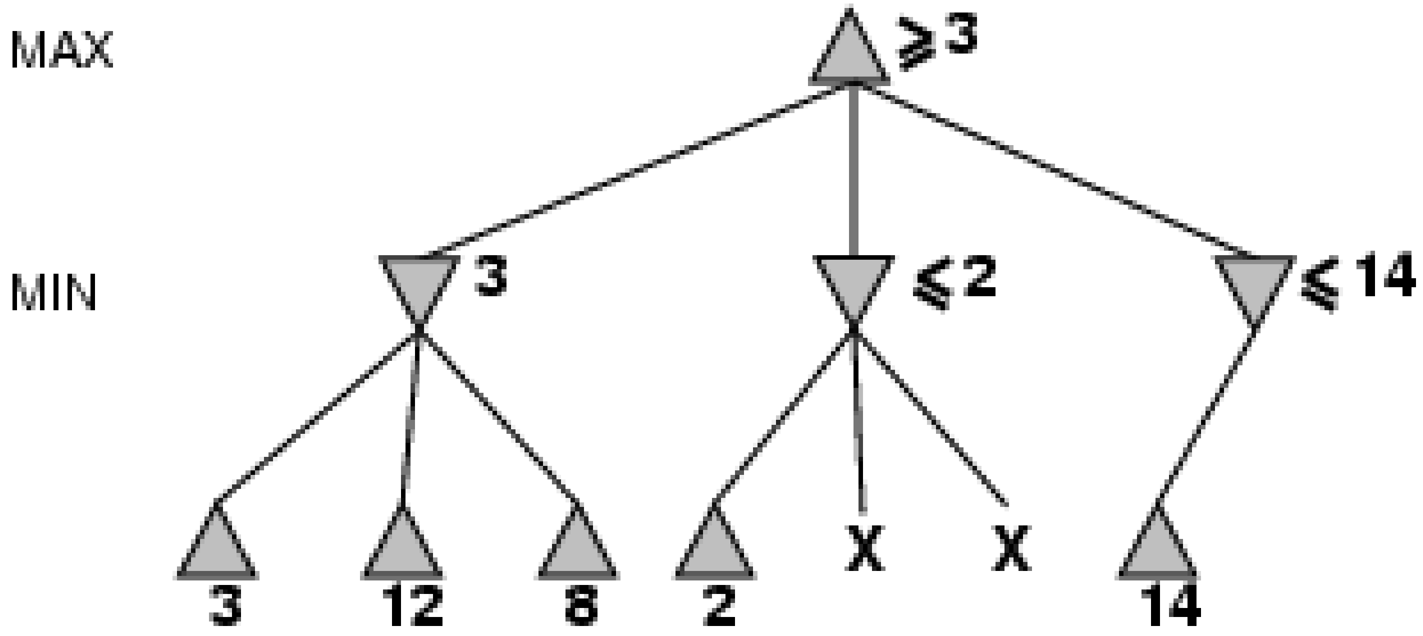


α-β pruning example

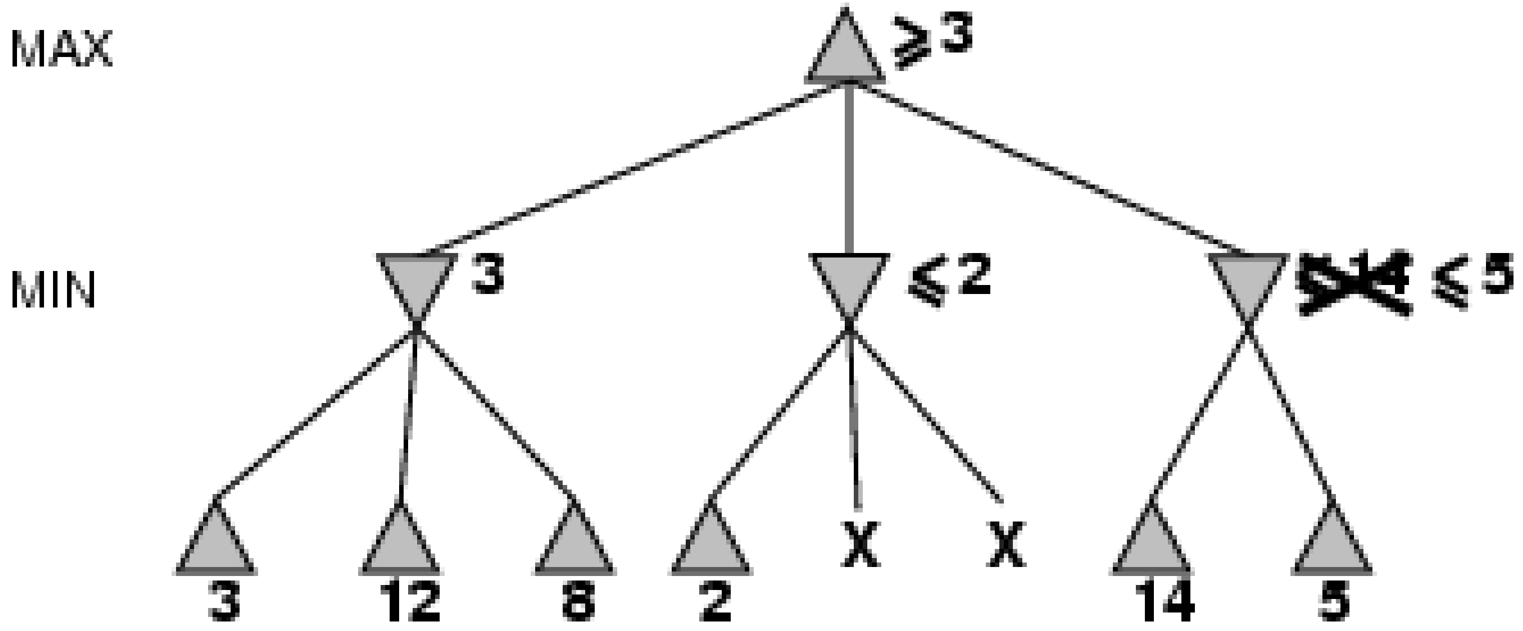


$\alpha\ cutoff$

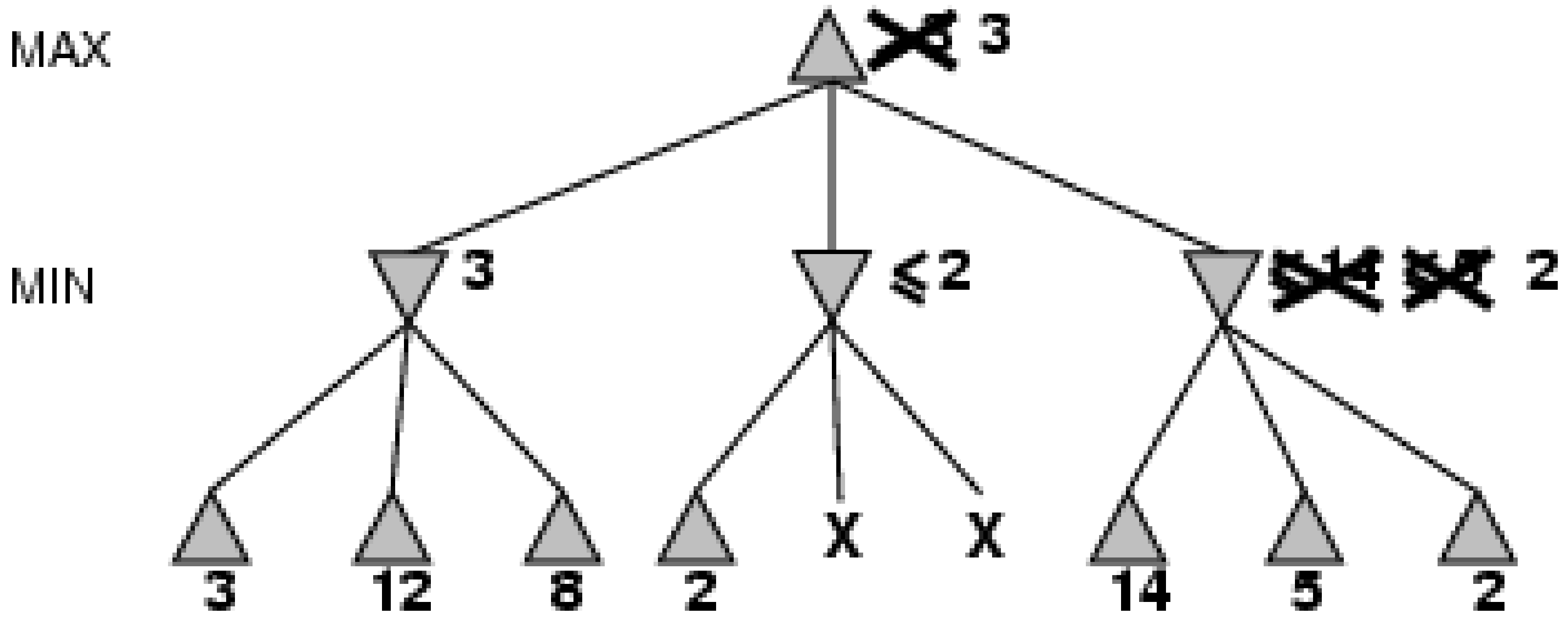
α-β pruning example



α-β pruning example



α-β pruning example



Alpha-Beta pruning

- $\alpha - cutoff$ **:** search can be discontinued below any MIN node having a beta value less than or equal to the alpha value of any of its MAX node ancestors.
- $\beta - cutoff$: search can be discontinued below any MAX node having an alpha value greater than or equal to the beta value of its MIN node ancestors.

Alpha-Beta pruning

- Alpha-Beta is guaranteed to compute the same Minimax value for the root node as computed by Minimax
- In the worst case, Alpha-Beta does NO pruning.
- In the best case, Alpha-Beta will examine only $2b^{m/2}$ leaf nodes.
- The best case occurs when each player's best move is the leftmost alternative (i.e., the first child generated). So, at MAX nodes the child with the largest value is generated first, and at MIN nodes the child with the smallest value is generated first.