MiniMax and Alpha Beta Pruning

CMSC 250H

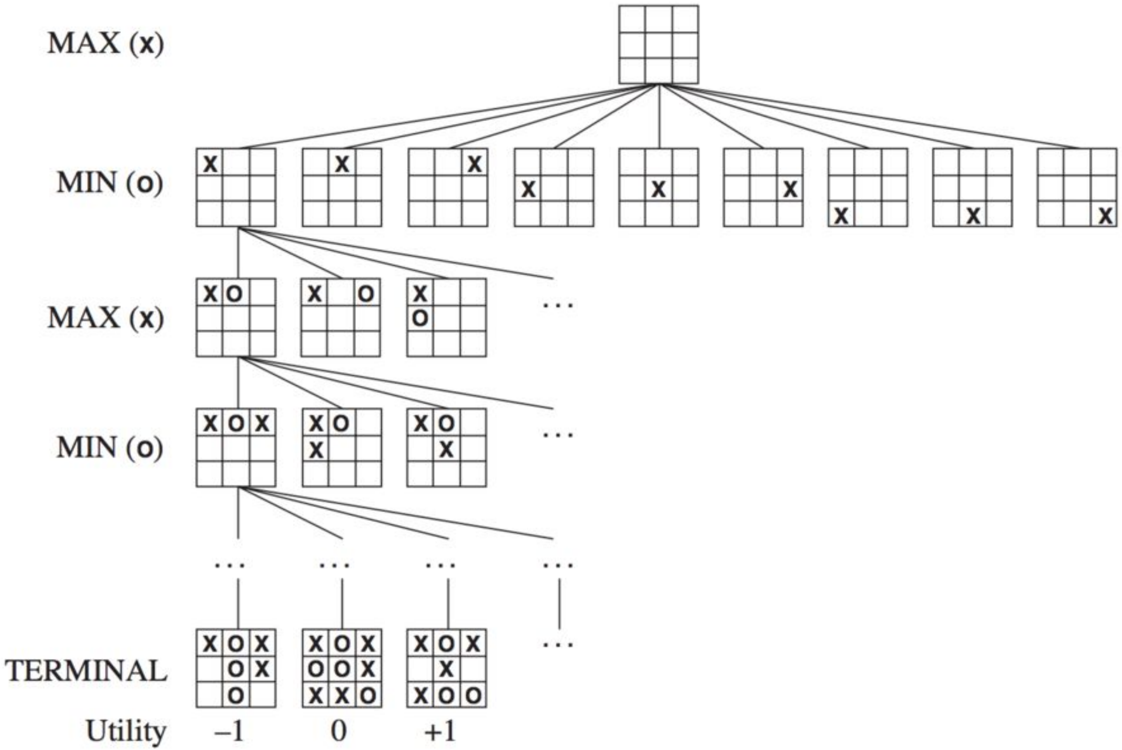
Games

- Games are interesting because they are really hard to solve
- Consider a game with two players: Min and Max
- Max moves first
- At the end of the game, points are awarded to the winning player and penalties are given to the loser
- A game can be defined as a kind of search problem with the following elements:
    - S_0: The initial state, which specifies how the game is set up at the start
    - PLAYER(s): Defines which player has the move in a state
    - ACTIONS(s): Returns the set of legal moves in a state
    - 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**

Game Tree

- A **Game Tree** is a tree where the nodes are game states and the edges are moves
  - Tic-tac-toe the game tree has fewer than 9! = 362,88- terminal nodes
  - Chess has over 10^{40} nodes

Game Tree



Optimal Decisions in Games

- In a normal search problem, the optimal solution would be a sequences of actions leading to a goal state
  - A goal state is a terminal state that is a win
- In adversarial search, Max must find a contingent strategy which specifies Max's move in the initial state
  - Max moves in the states resulting from every possible move Min can make
  - Min then moves in the states resulting from every possible move Max can make

Minimax

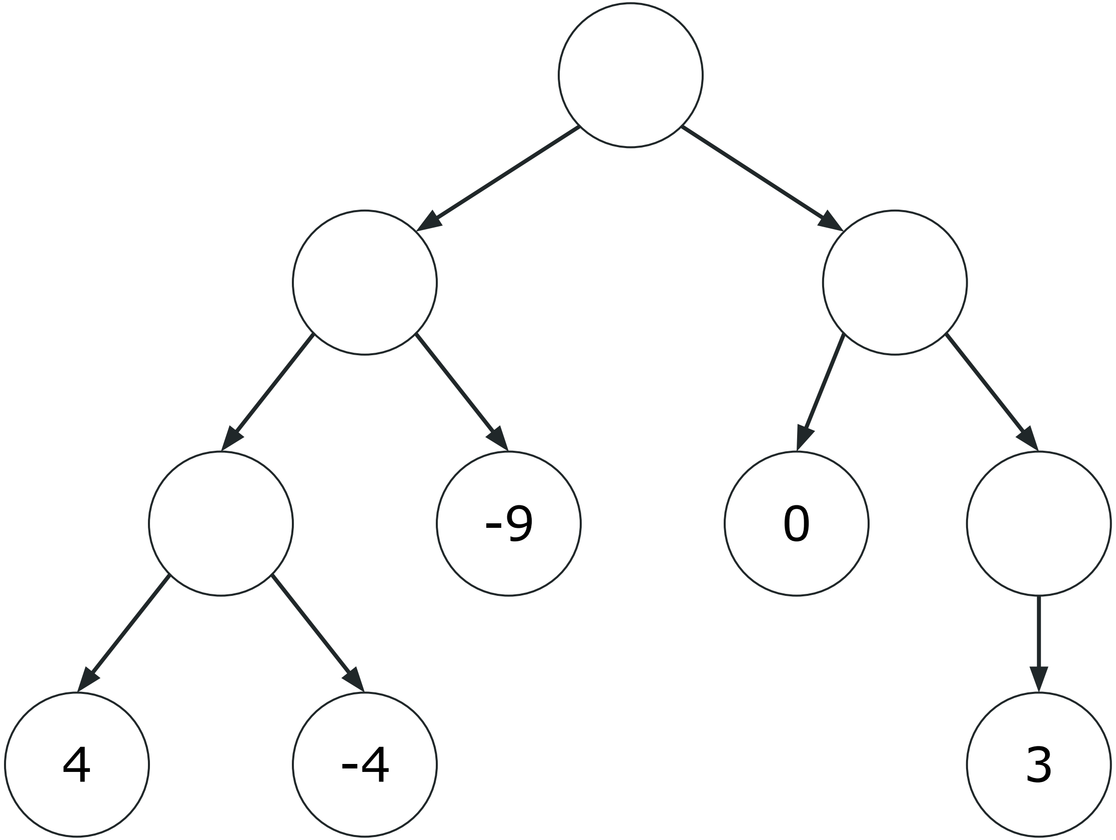
MiniMax Value

- Given a game tree, the optimal strategy can be determined from the **minimax value** of each node
- The minimax value of a node is the utility of being in the corresponding state, assuming that both players play optimally from there to the end of the game
    - Utility defines the final numeric value for a game that ends in terminal state
        - It is common to use 1 for Max winning, -1 for Max loosing, and 0 for Max and Min tying
- Given a choice, Max will always move to a state of a maximum value
    - Min will always move to a state of a minimum value

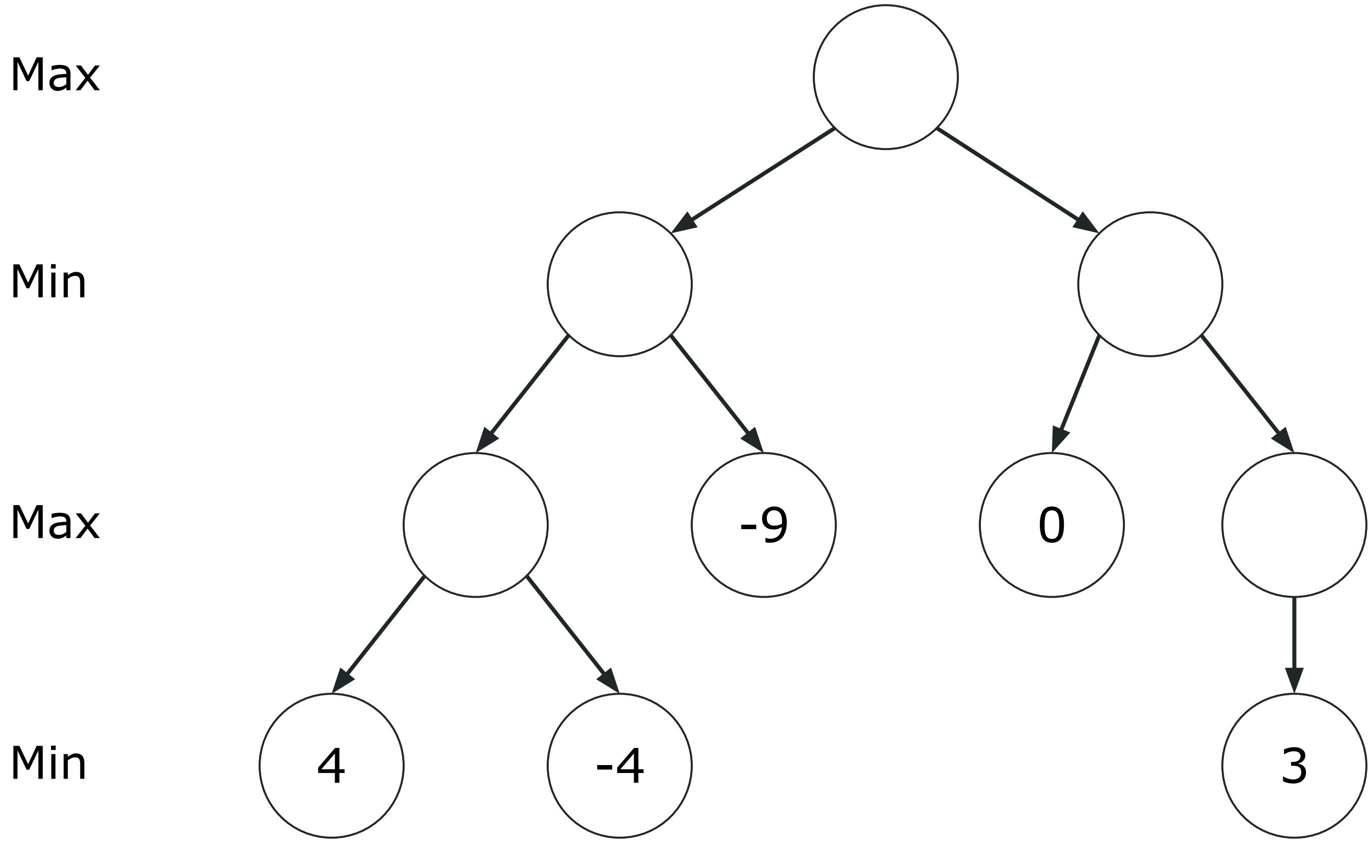
MiniMax

- The minimax algorithm computes the minimax decision from the current state
- It uses recursion of the minimax values of each successor
- The recursion proceeds all the way down to the leaves of the tree, and then the minimax values are backed up through the tree
- The minimax algorithm performs a complete depth-first exploration of the game tree

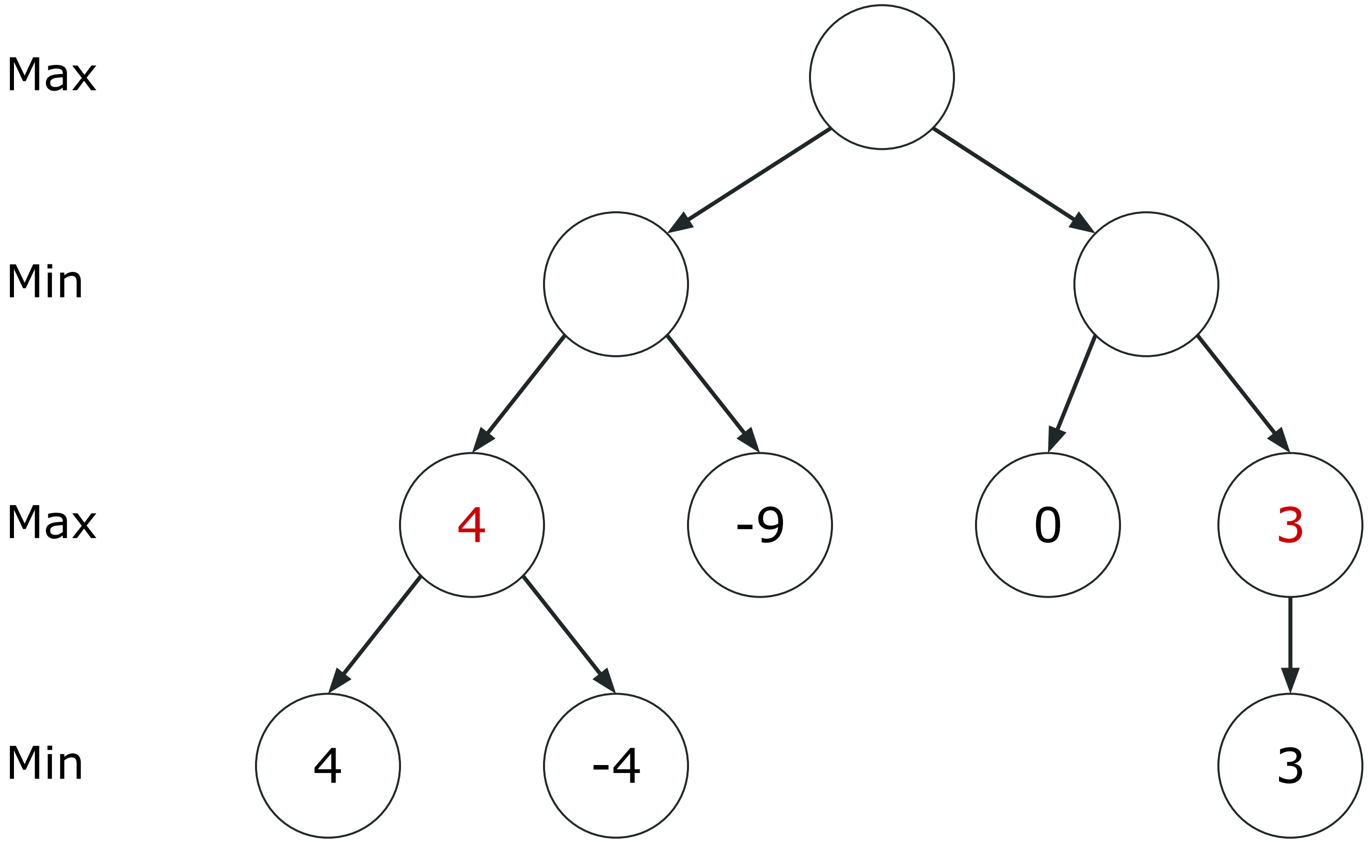
MiniMax Example



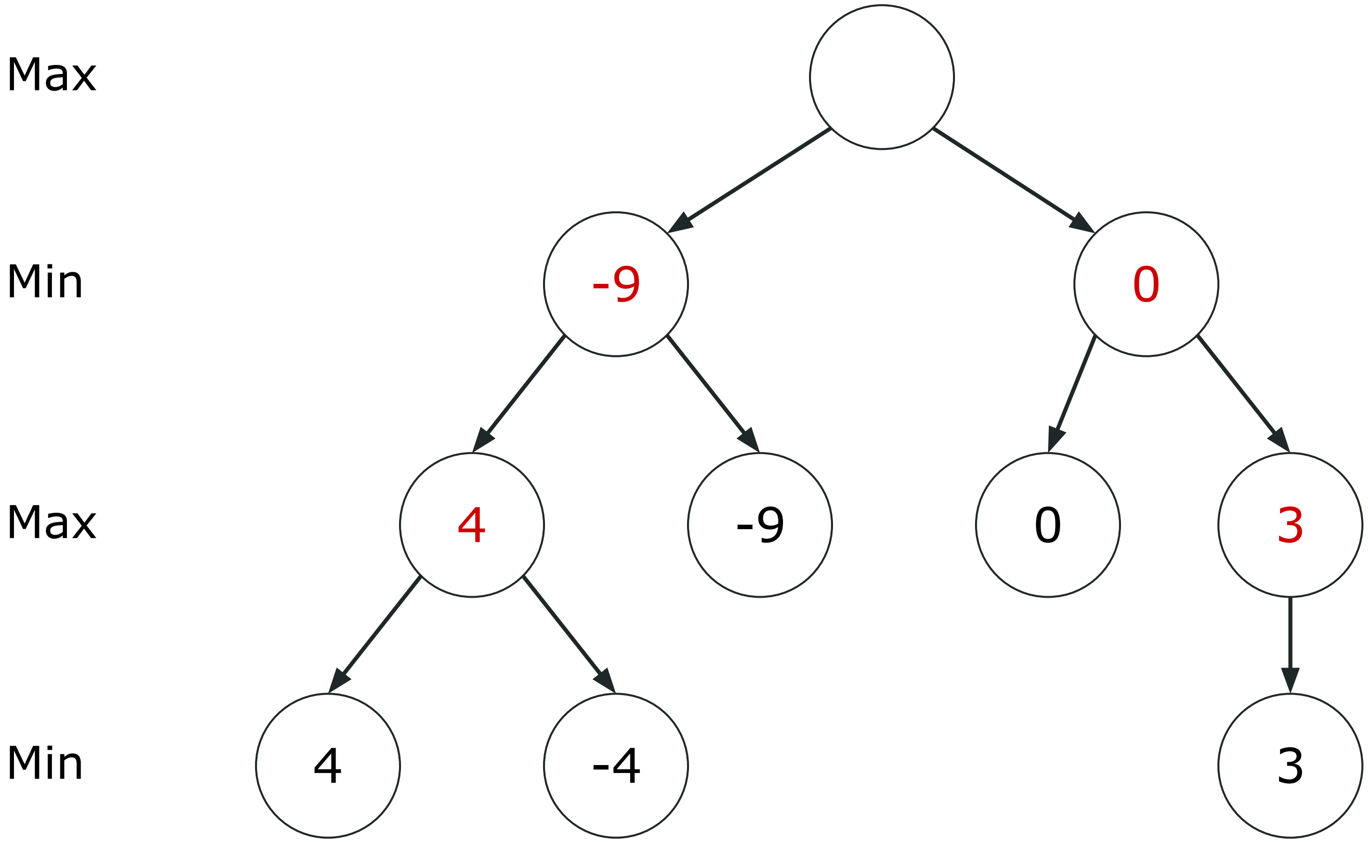
MiniMax Example



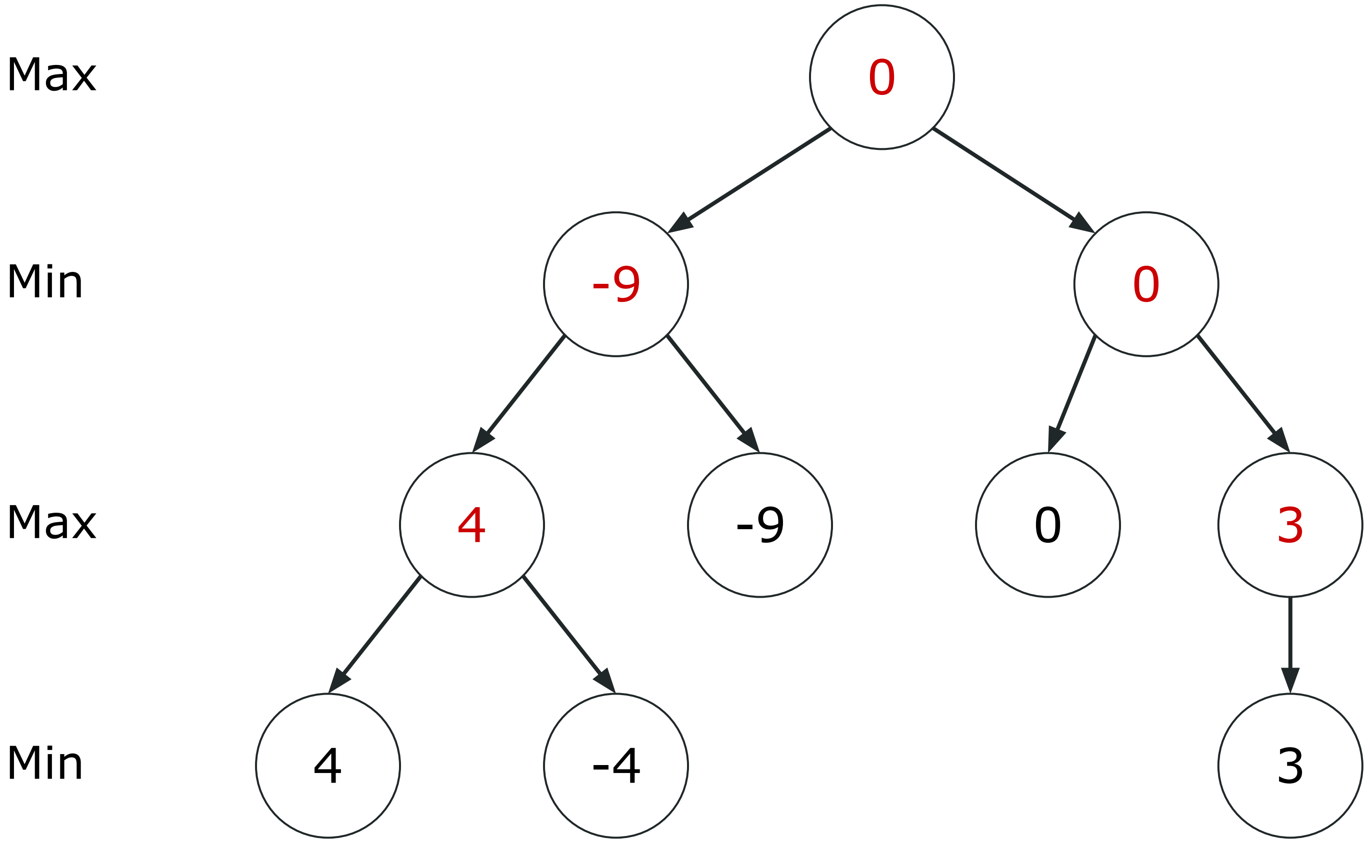
MiniMax



MiniMax



MiniMax



Minimax Analysis

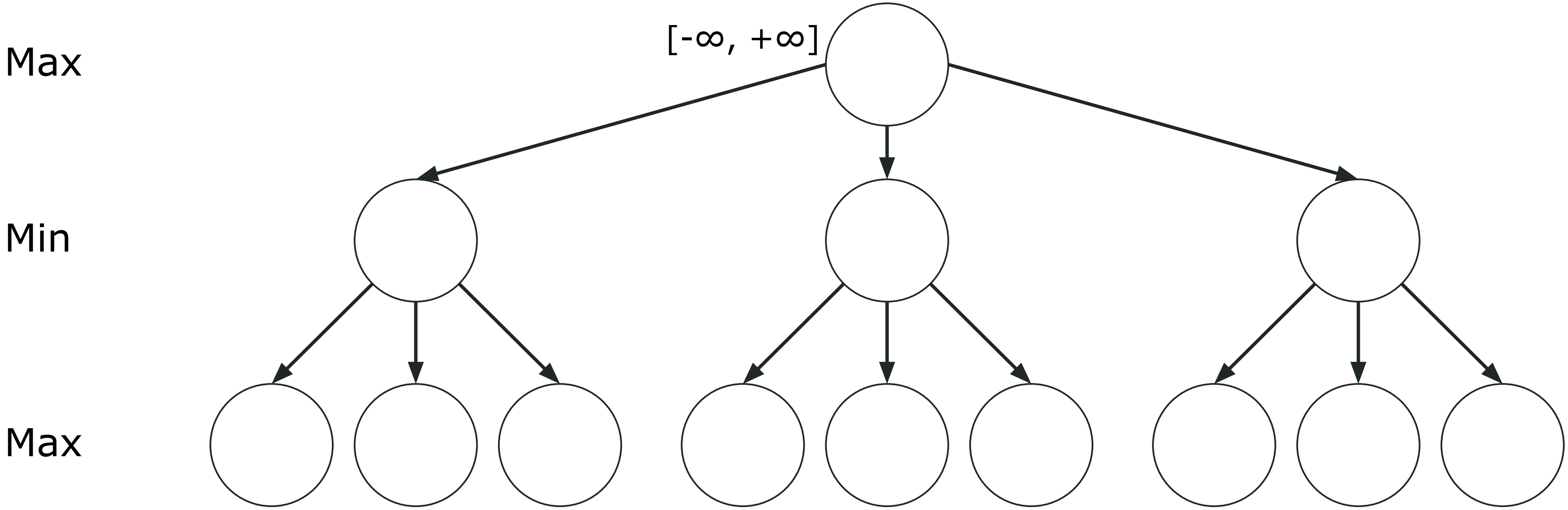
- If the maximum depth of the tree is *h* and there are *b* legal moves at each point, then the time complexity of the minimax algorithm is $O(b^h)$
  - The space complexity is $O(bh)$
- For real games, the time cost is totally impractical, but this algorithm serves as the basis for the mathematical analysis of games and for more practical algorithms

Alpha Beta Pruning

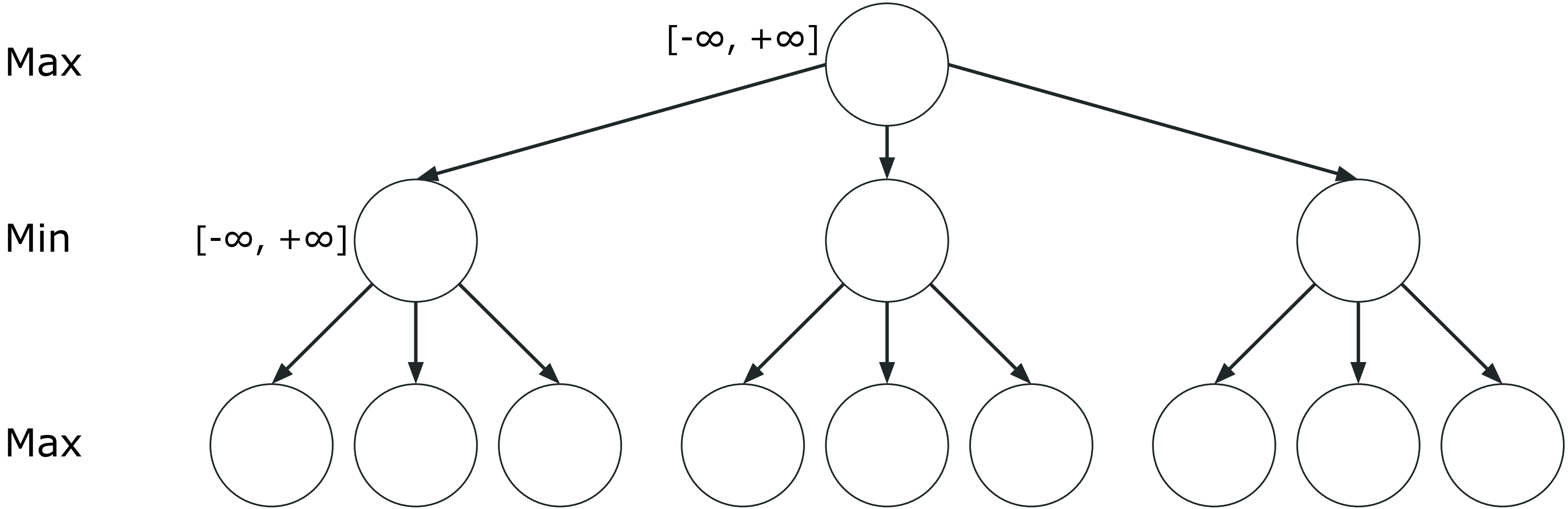
Alpha Beta Pruning

- Makes MiniMax more efficient
- If we search down the whole tree, the number of states is exponential to the depth of the tree
- Alpha Beta Pruning cuts away leaves when traversing tree
- Stops evaluating a state when at least one possibility has been found to prove worse then a previous found move
- Returns the same value that MiniMax would produce
- Prunes away branches that do not influence final decision
- In the tuple [α, β]
  - Maximize α
  - Minimize β

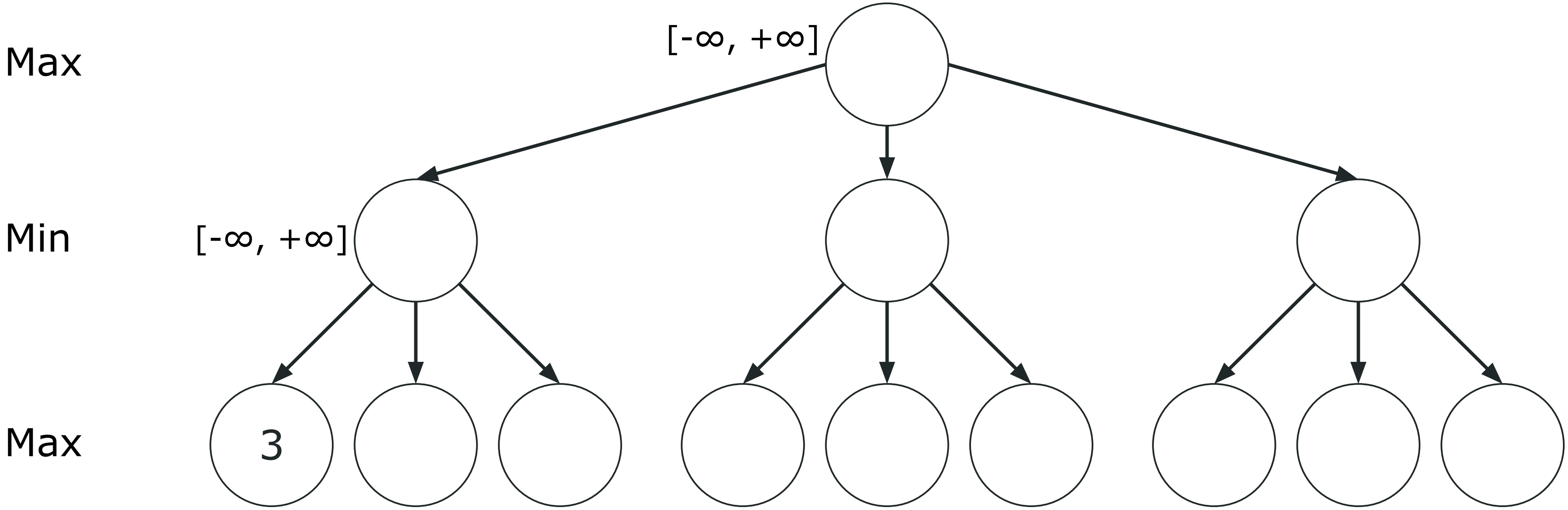
Alpha Beta Pruning Example 1



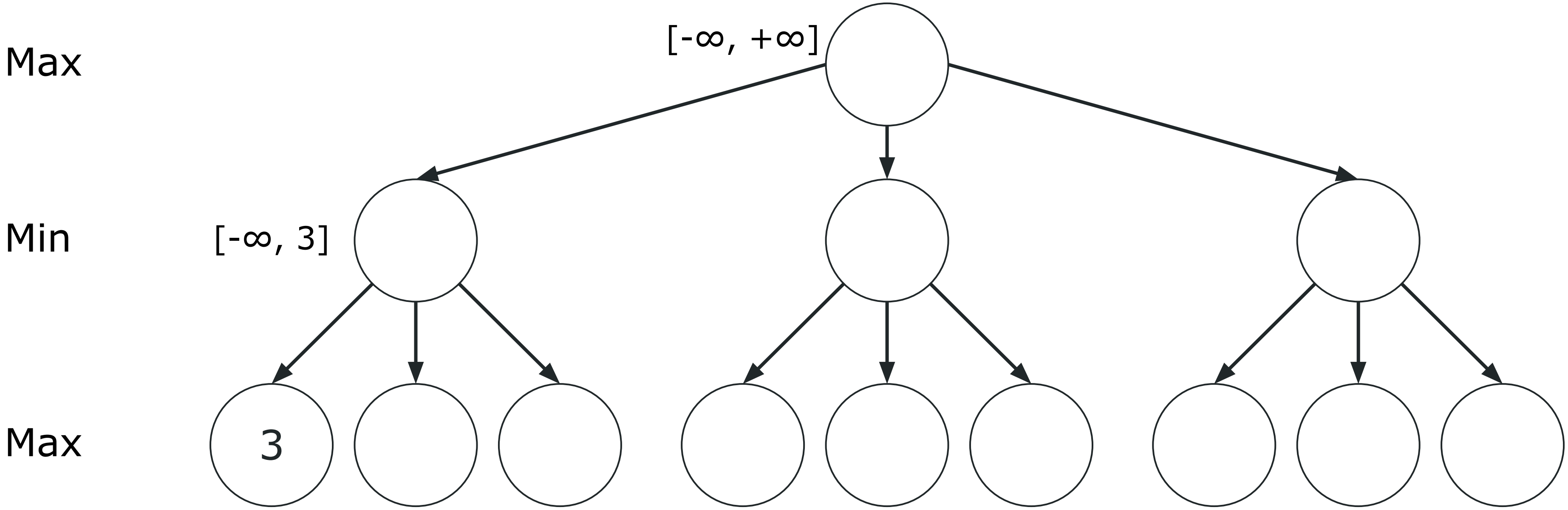
Alpha Beta Pruning Example 1



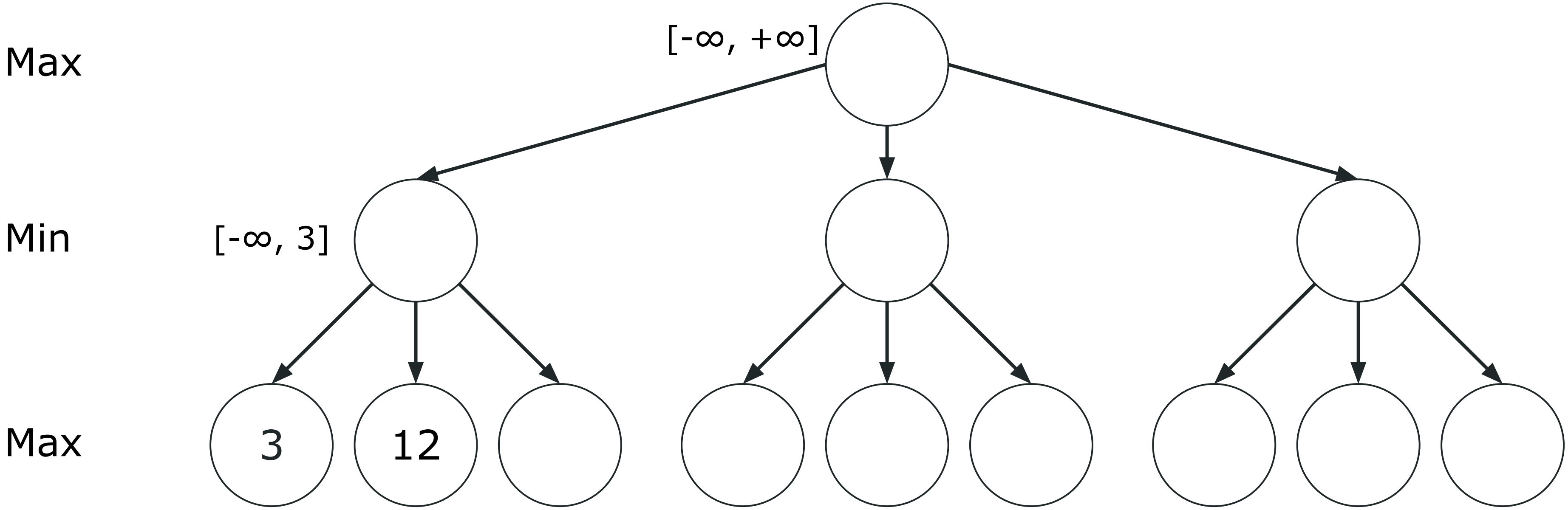
Alpha Beta Pruning Example 1



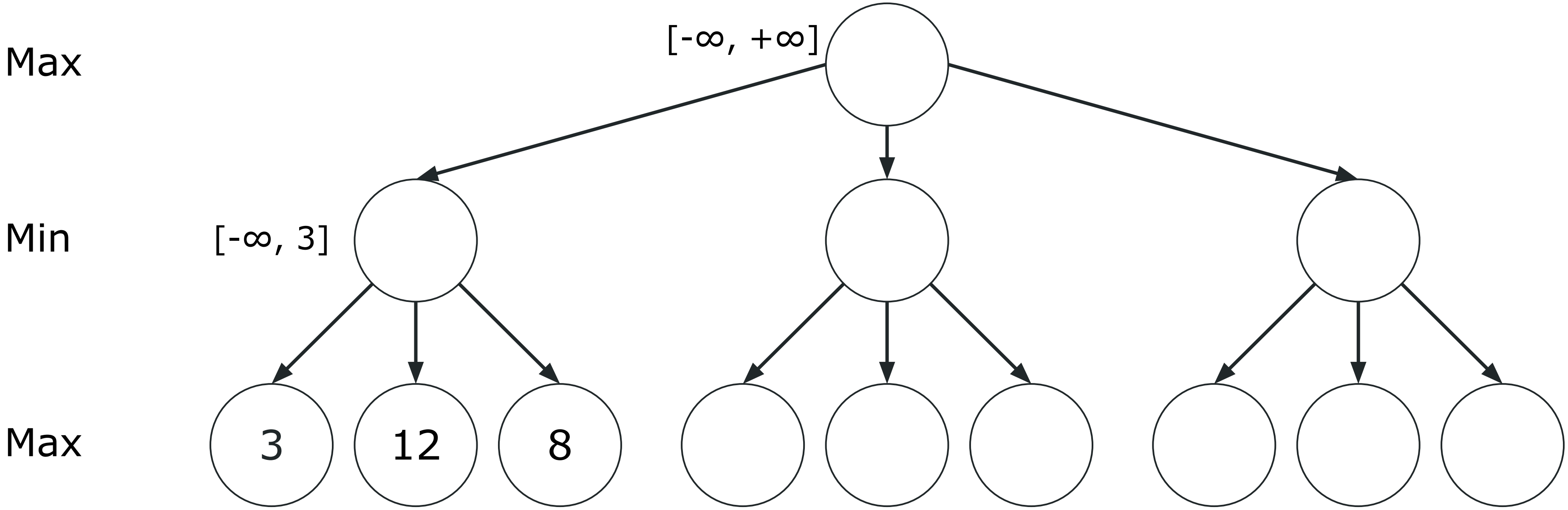
Alpha Beta Pruning Example 1



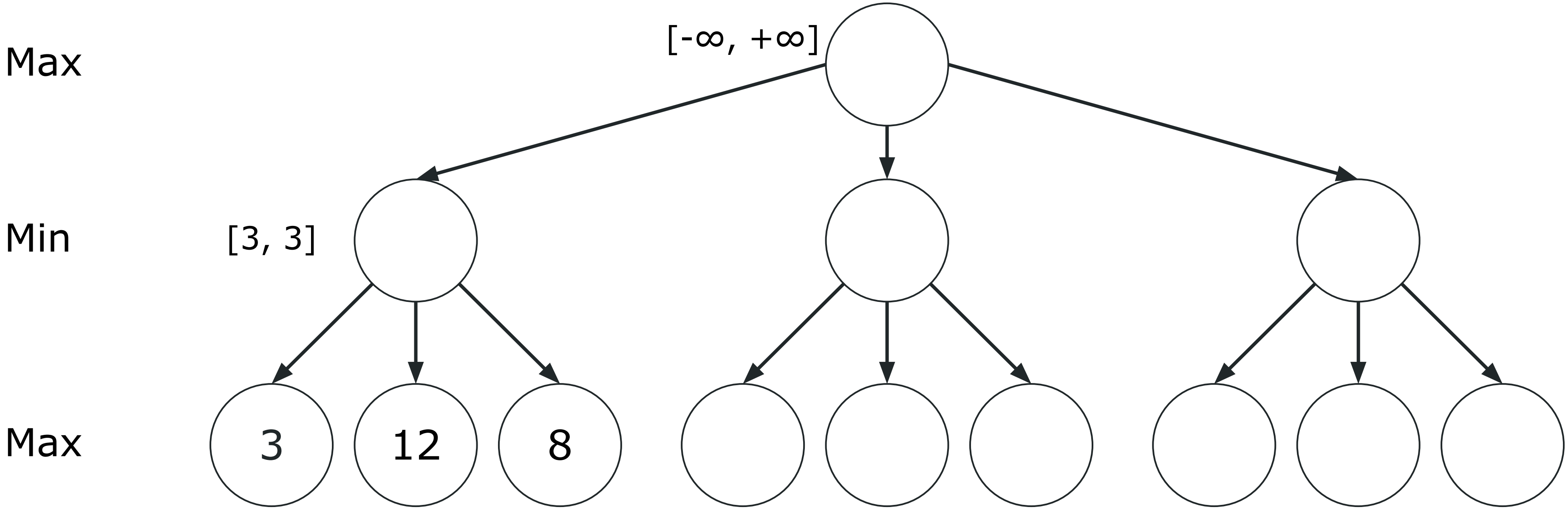
Alpha Beta Pruning Example 1



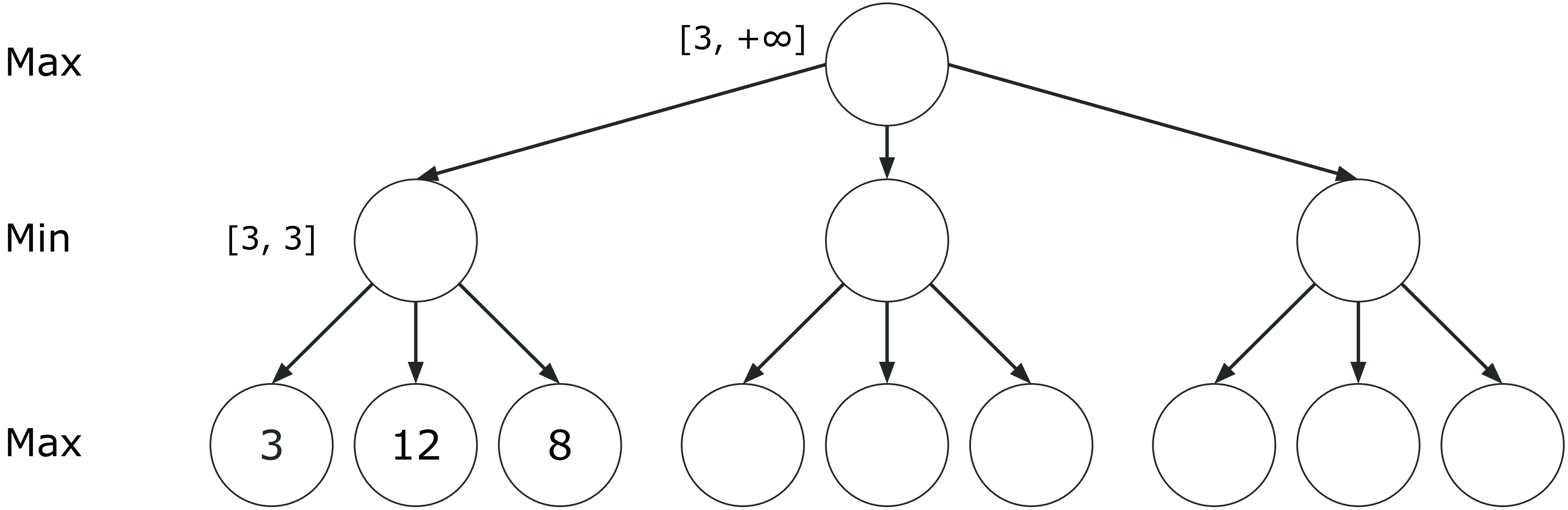
Alpha Beta Pruning Example 1



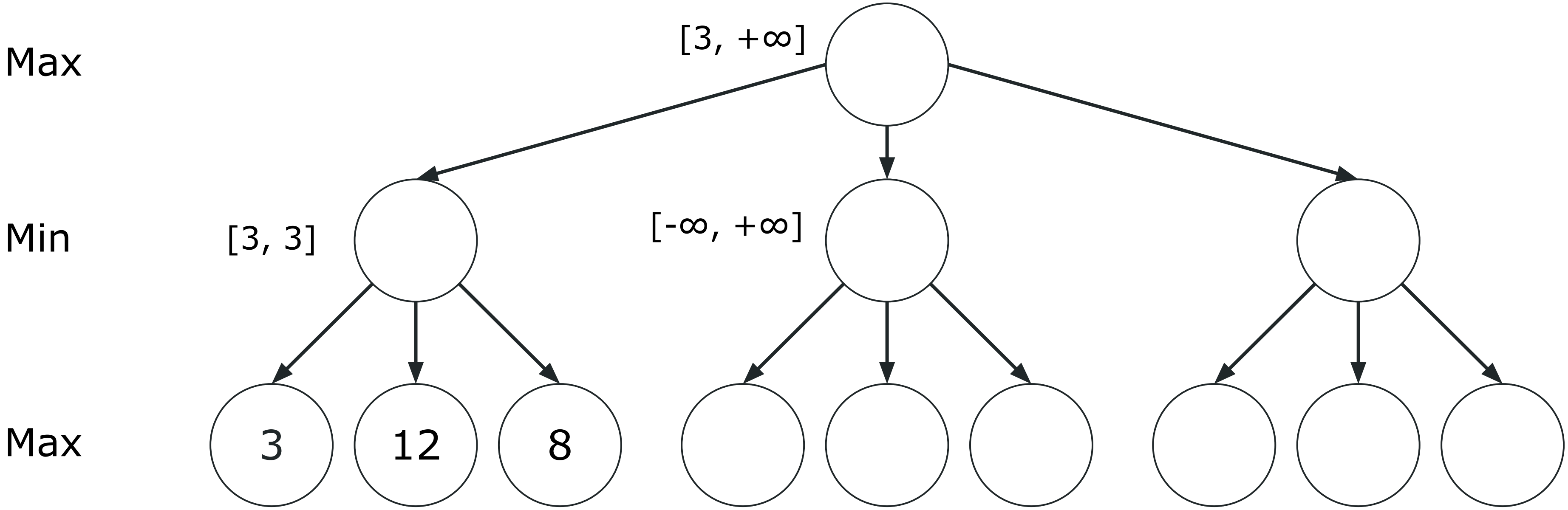
Alpha Beta Pruning Example 1



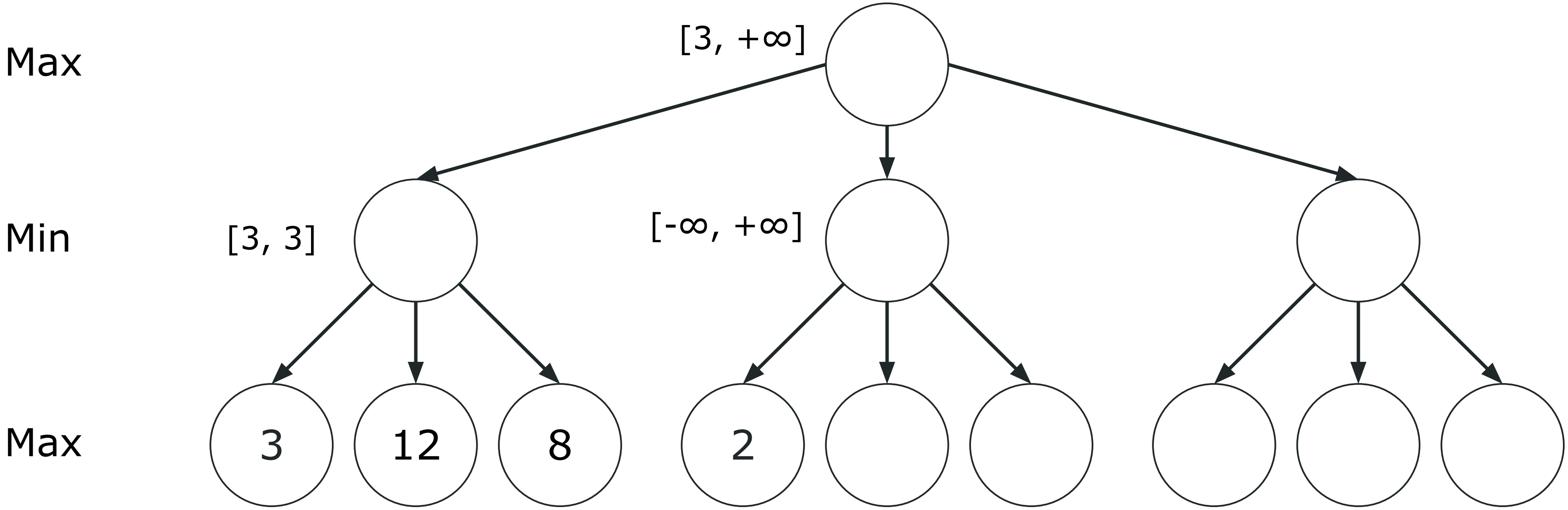
Alpha Beta Pruning Example 1



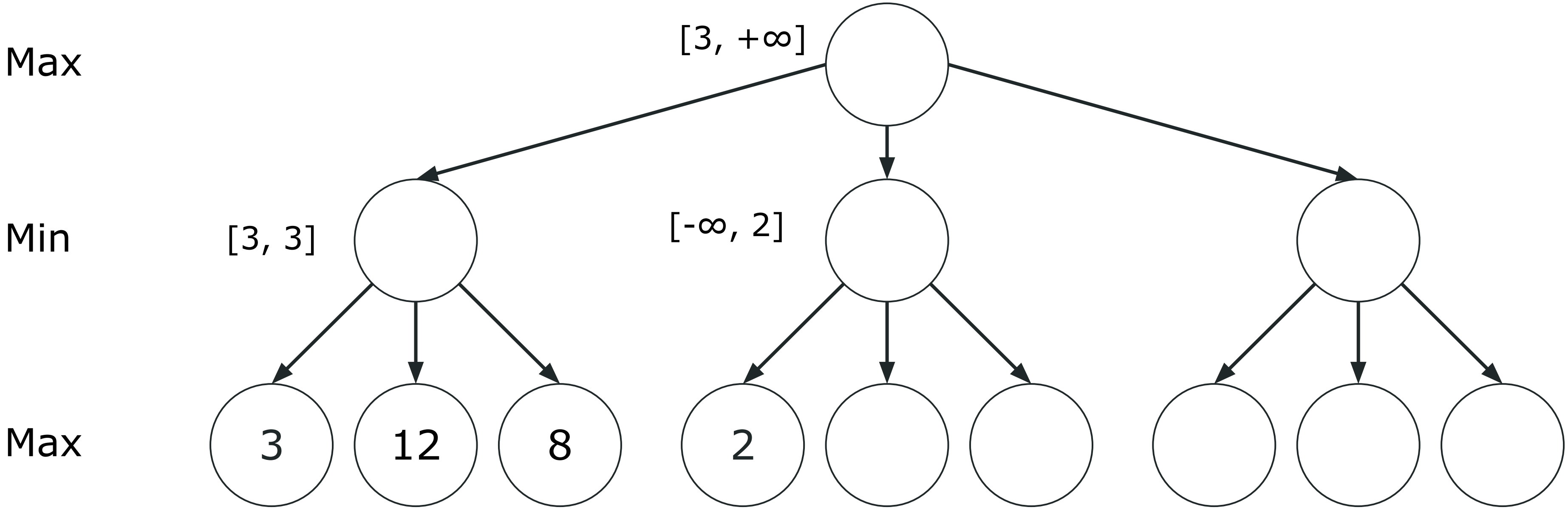
Alpha Beta Pruning Example 1



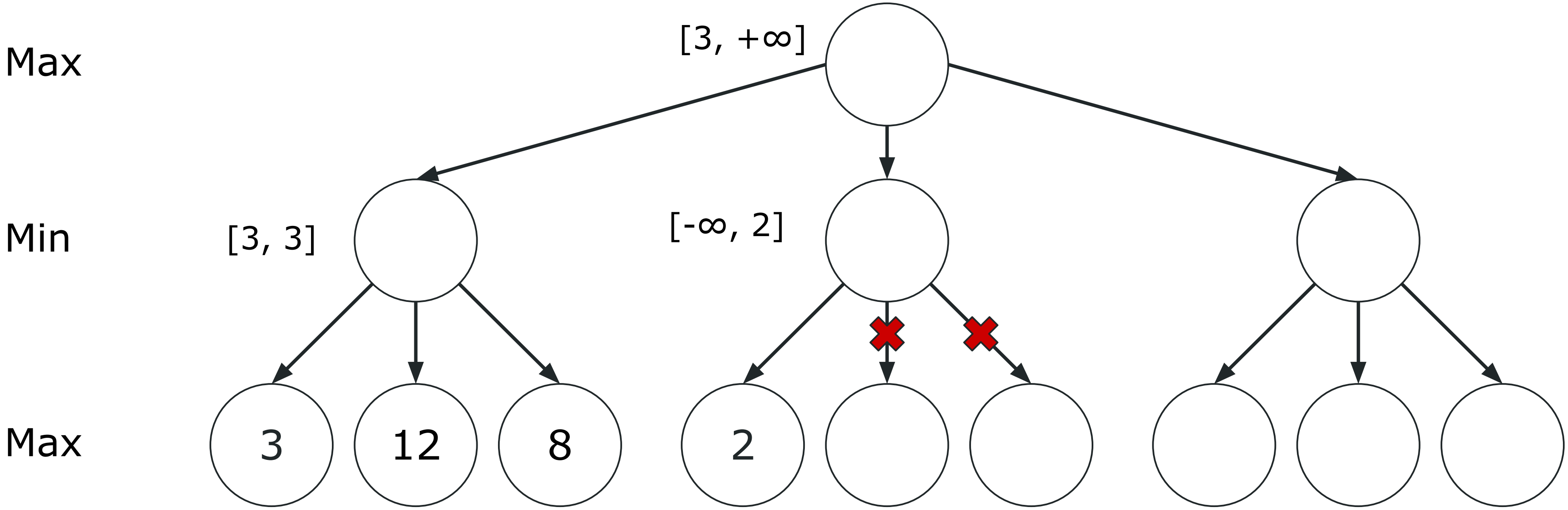
Alpha Beta Pruning Example 1



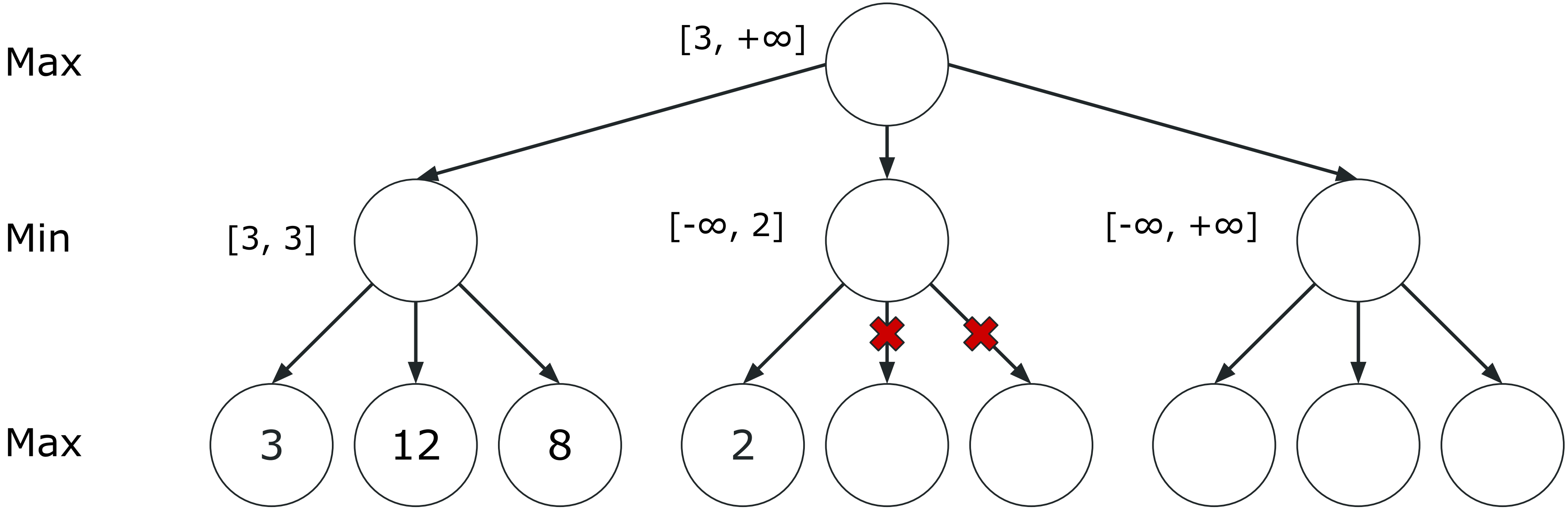
Alpha Beta Pruning Example 1



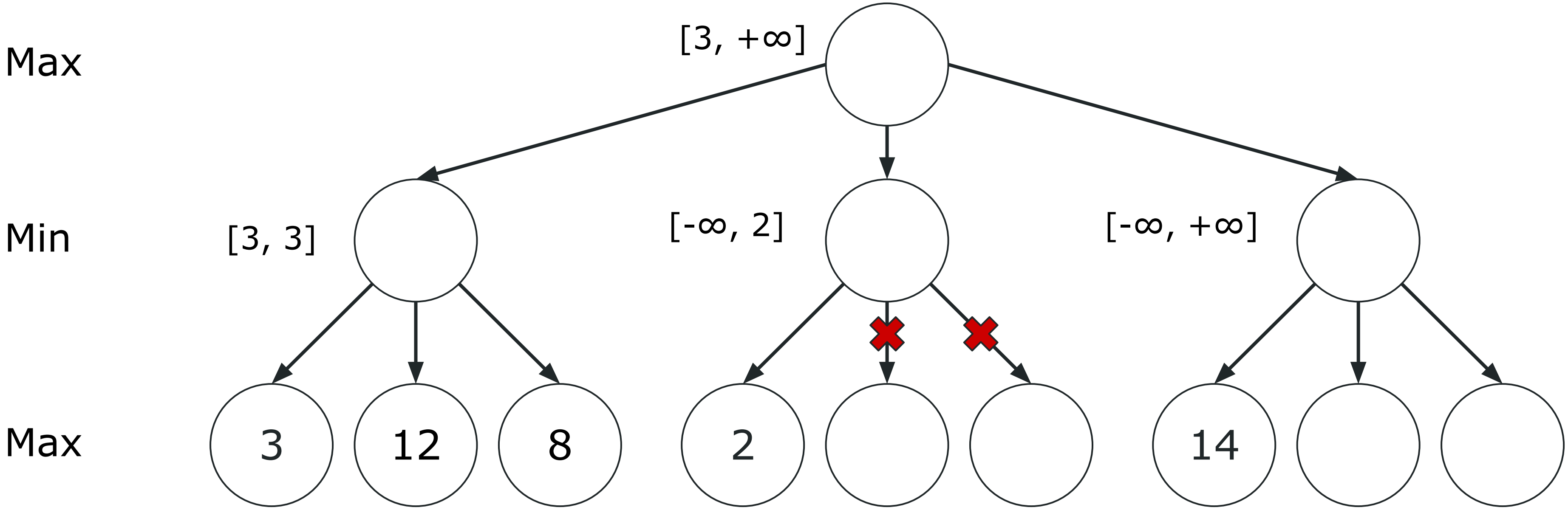
Alpha Beta Pruning Example 1



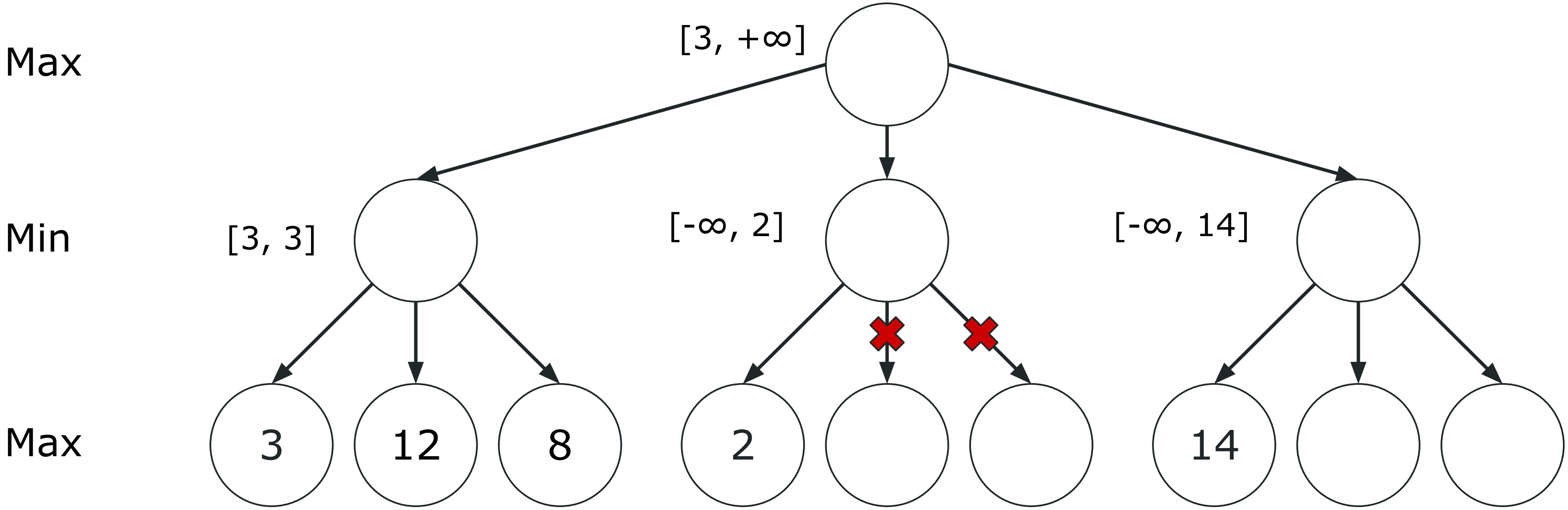
Alpha Beta Pruning Example 1



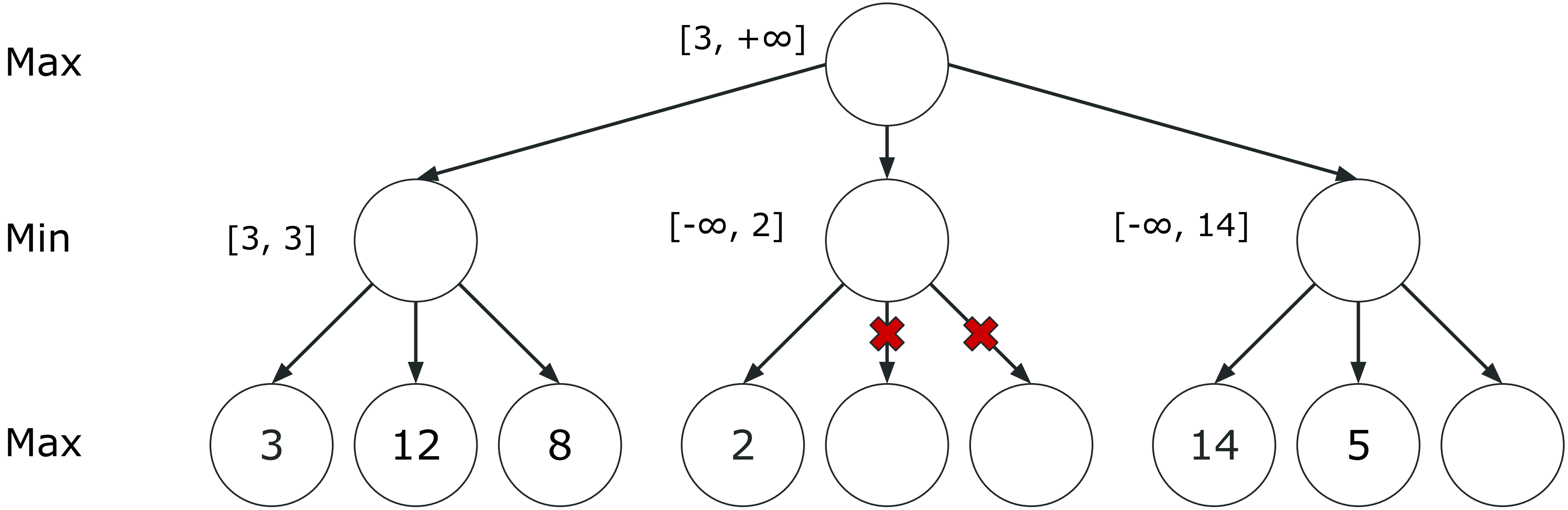
Alpha Beta Pruning Example 1



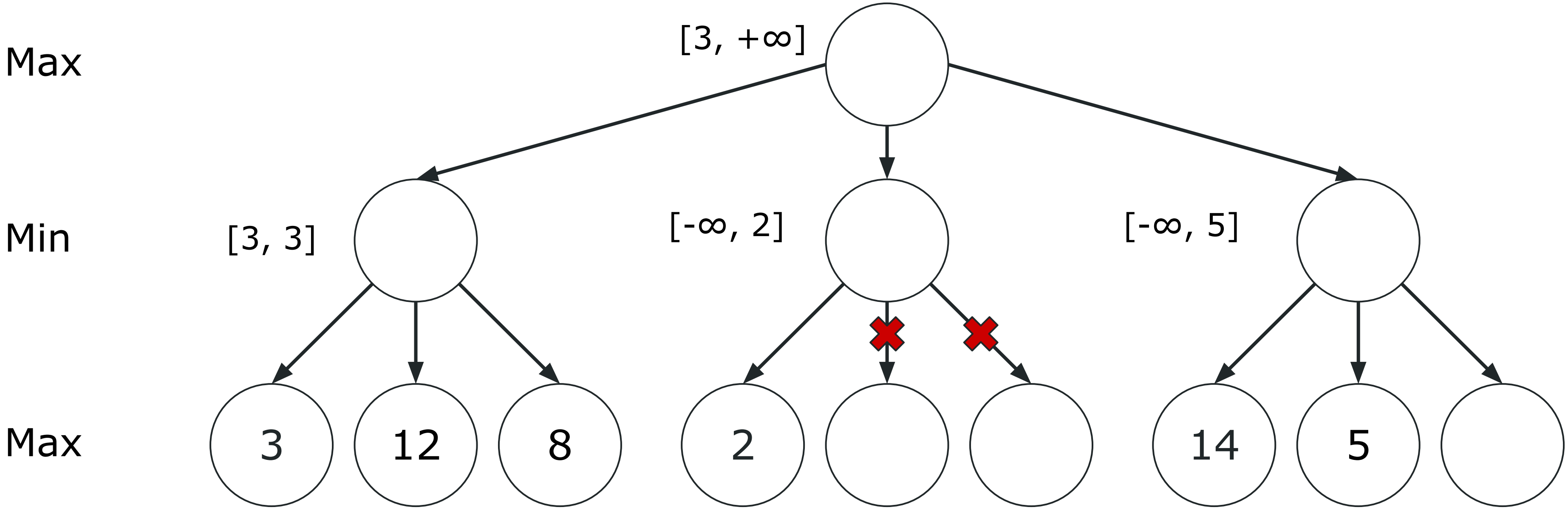
Alpha Beta Pruning Example 1



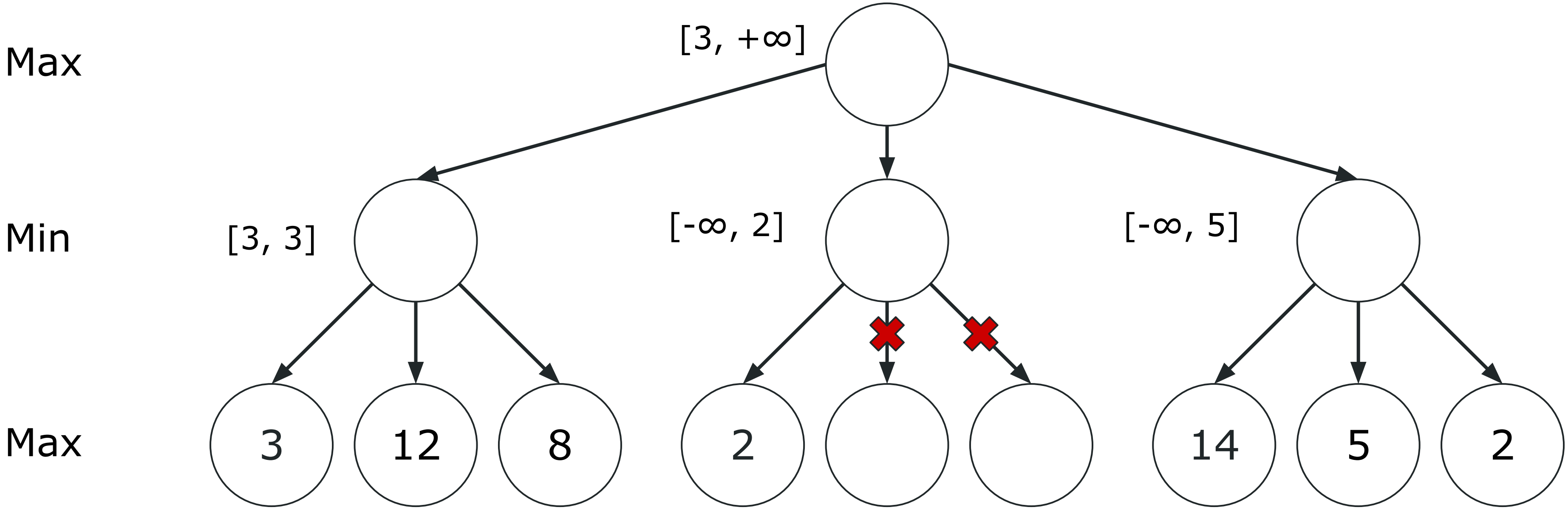
Alpha Beta Pruning Example 1



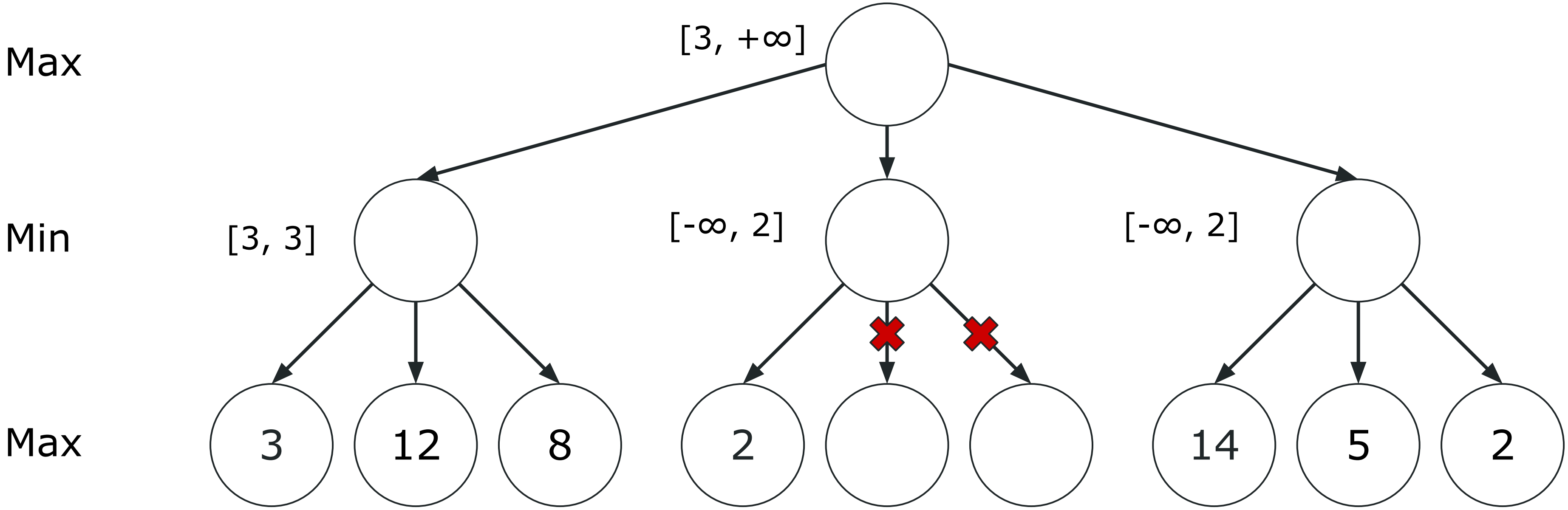
Alpha Beta Pruning Example 1



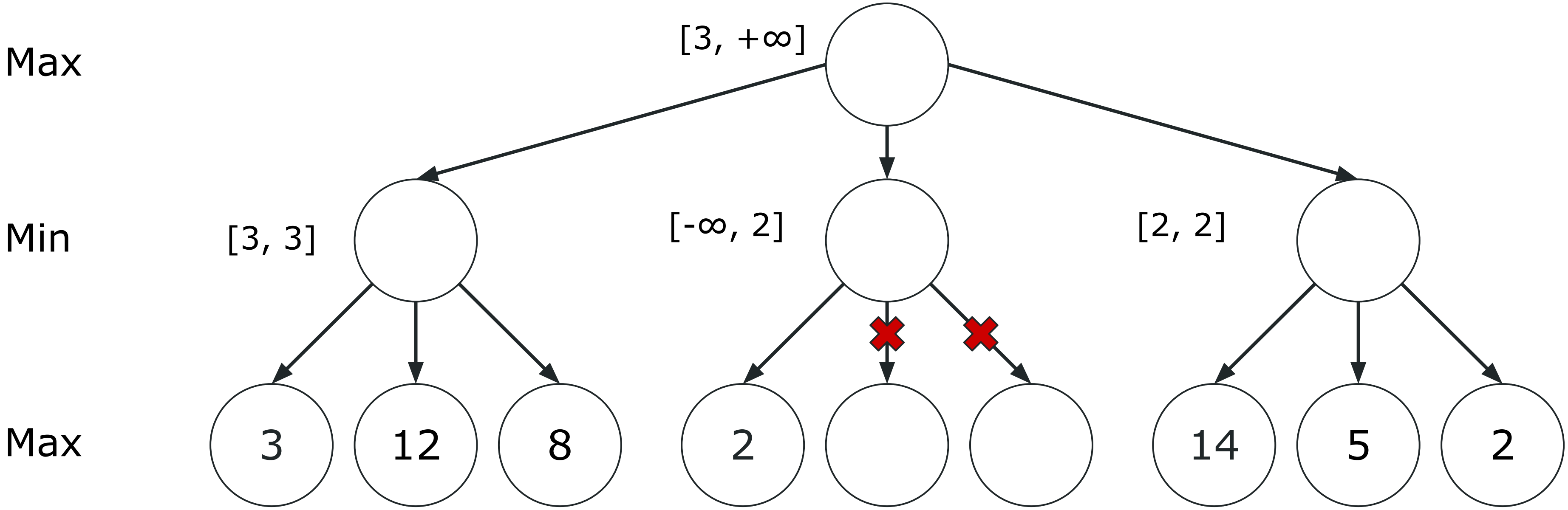
Alpha Beta Pruning Example 1



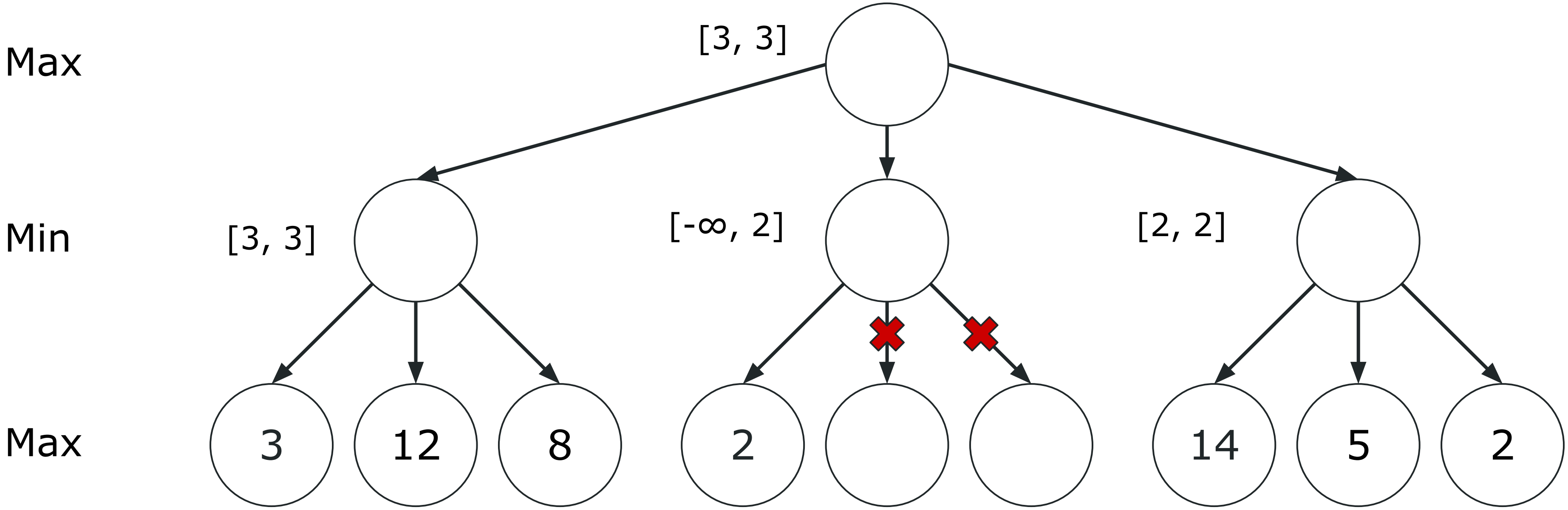
Alpha Beta Pruning Example 1



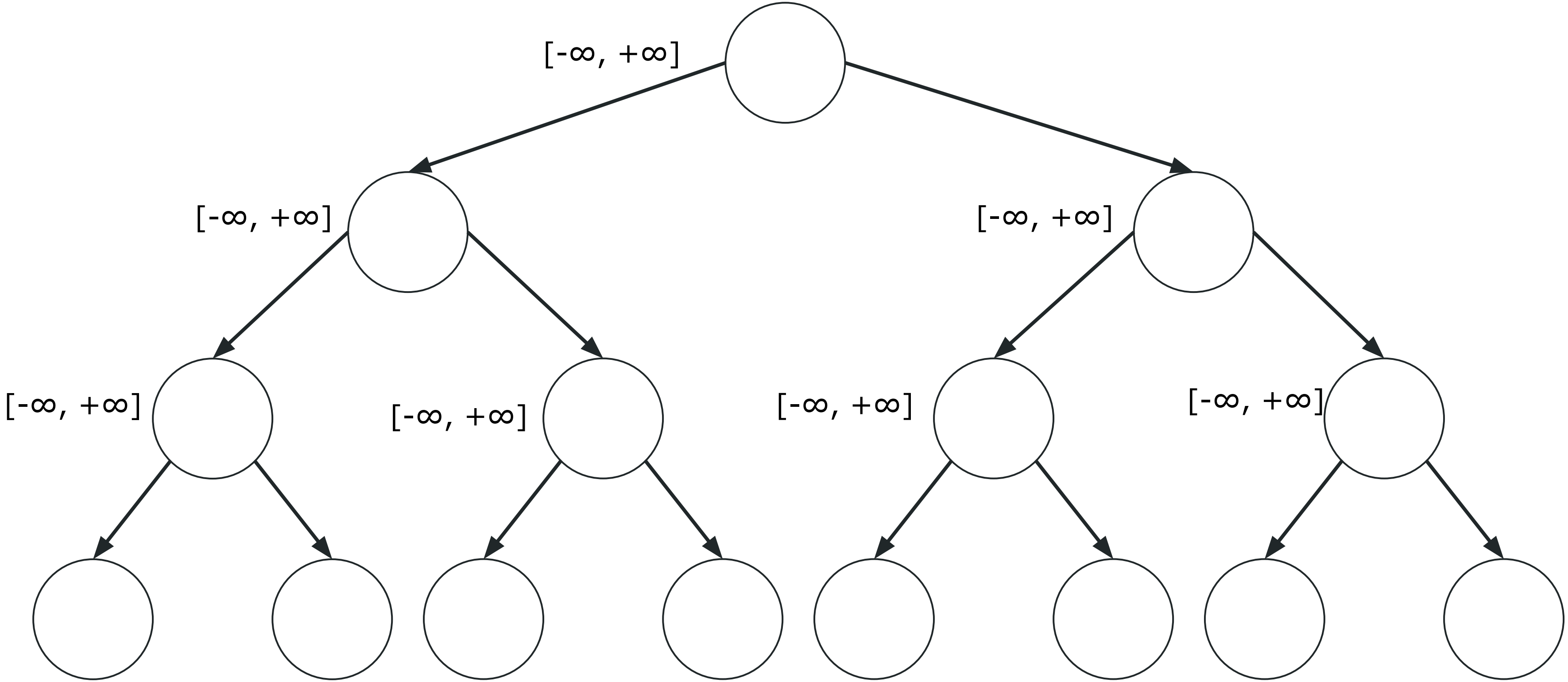
Alpha Beta Pruning Example 1



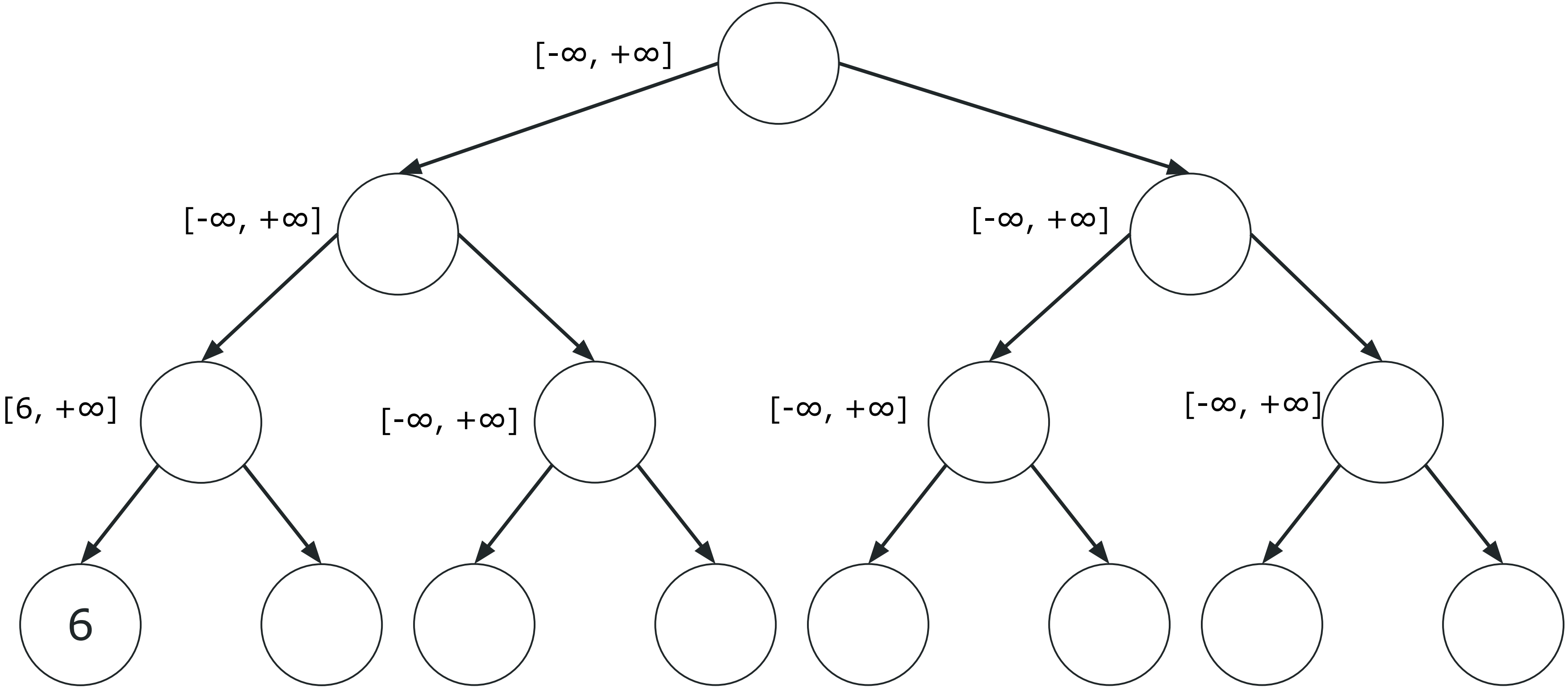
Alpha Beta Pruning Example 1



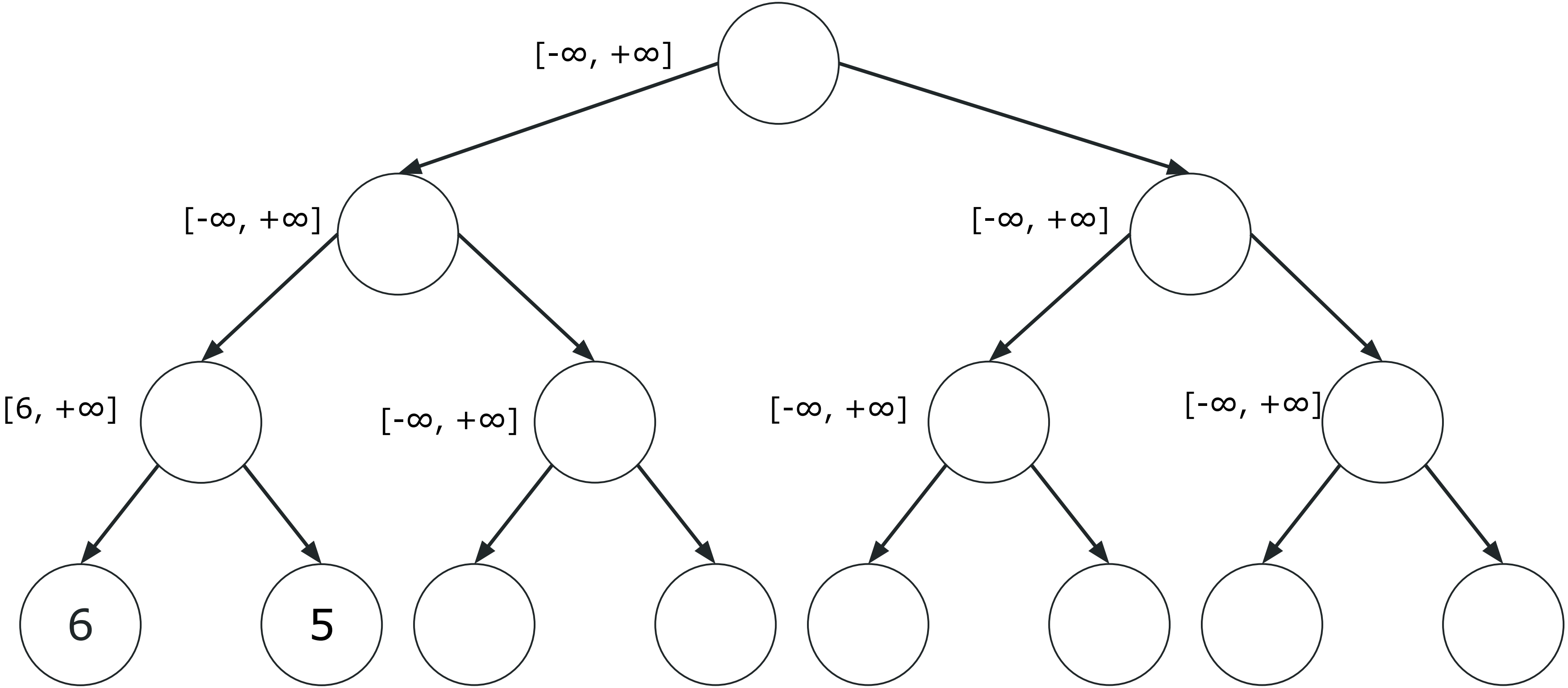
Alpha Beta Pruning Example 2



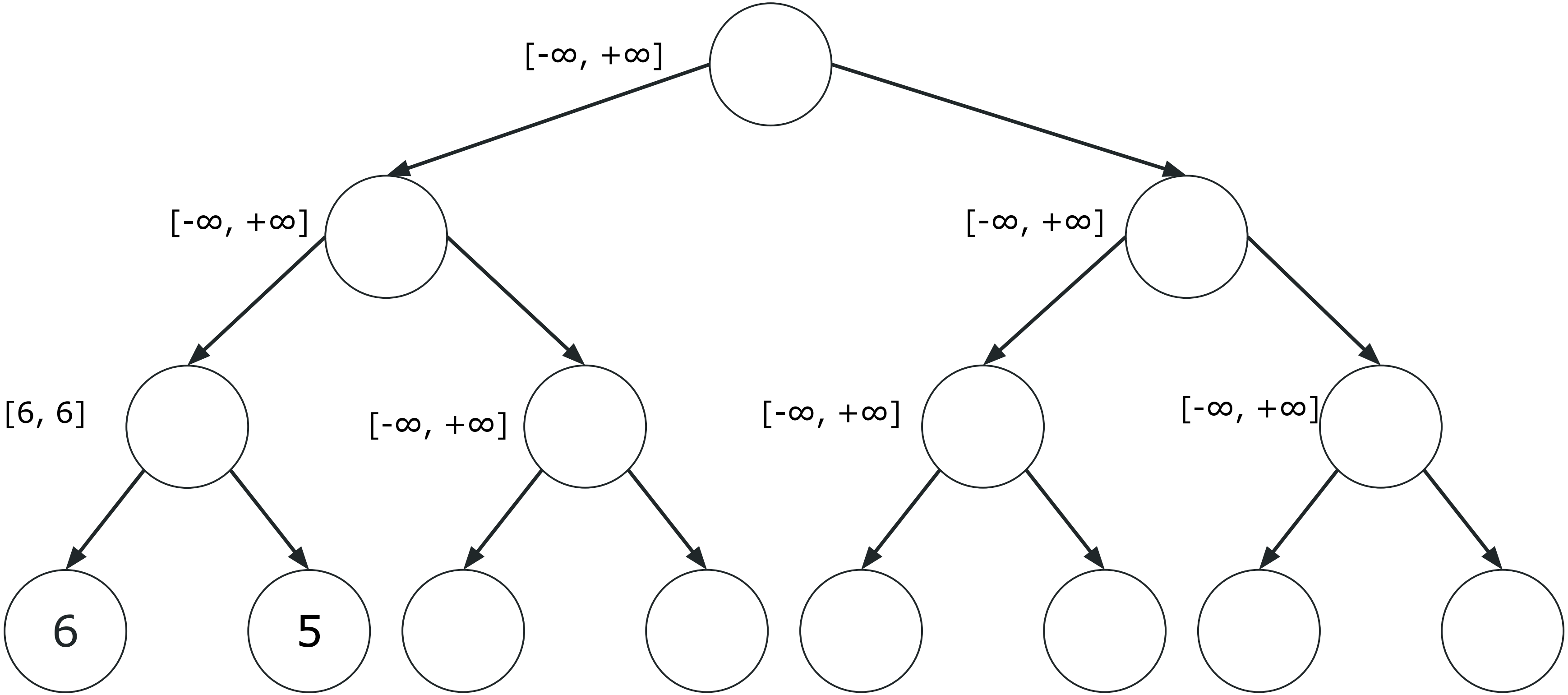
Alpha Beta Pruning Example 2



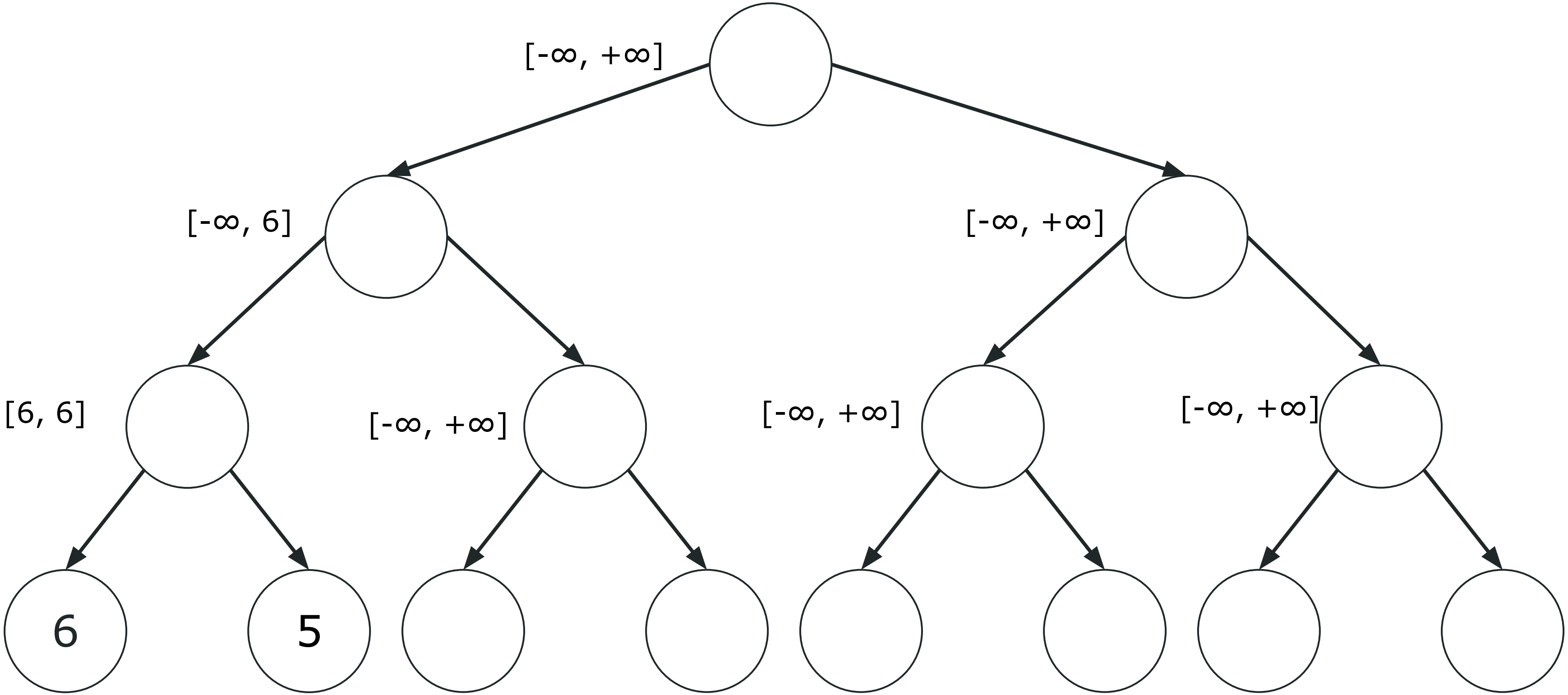
Alpha Beta Pruning Example 2



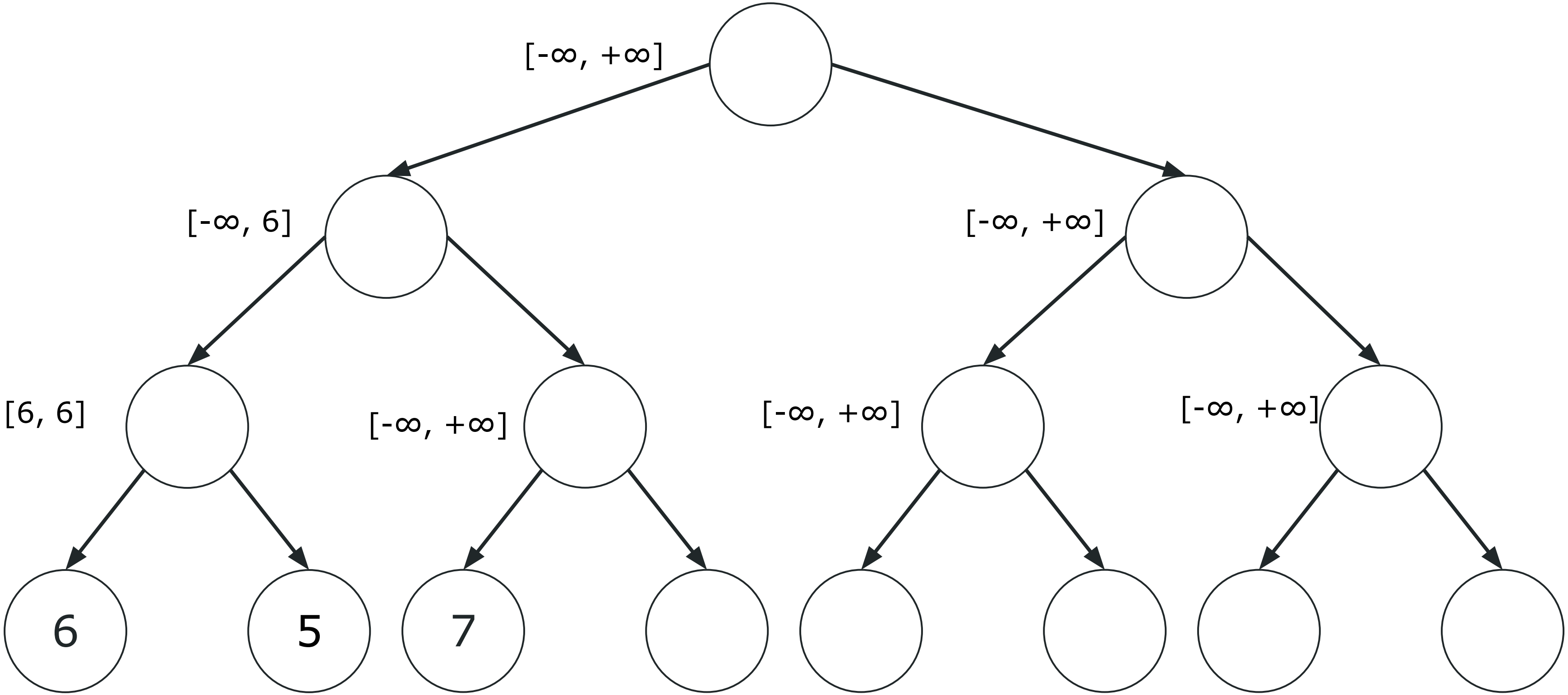
Alpha Beta Pruning Example 2



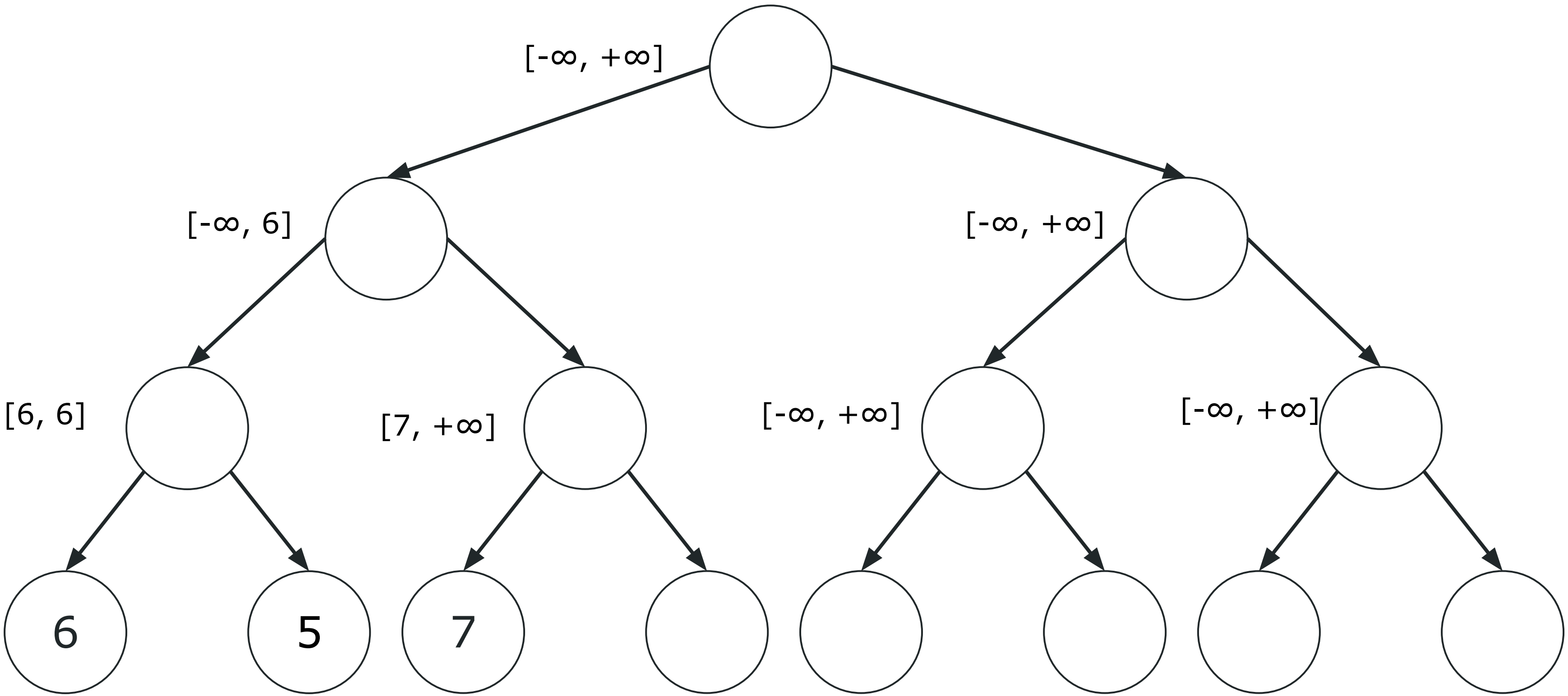
Alpha Beta Pruning Example 2



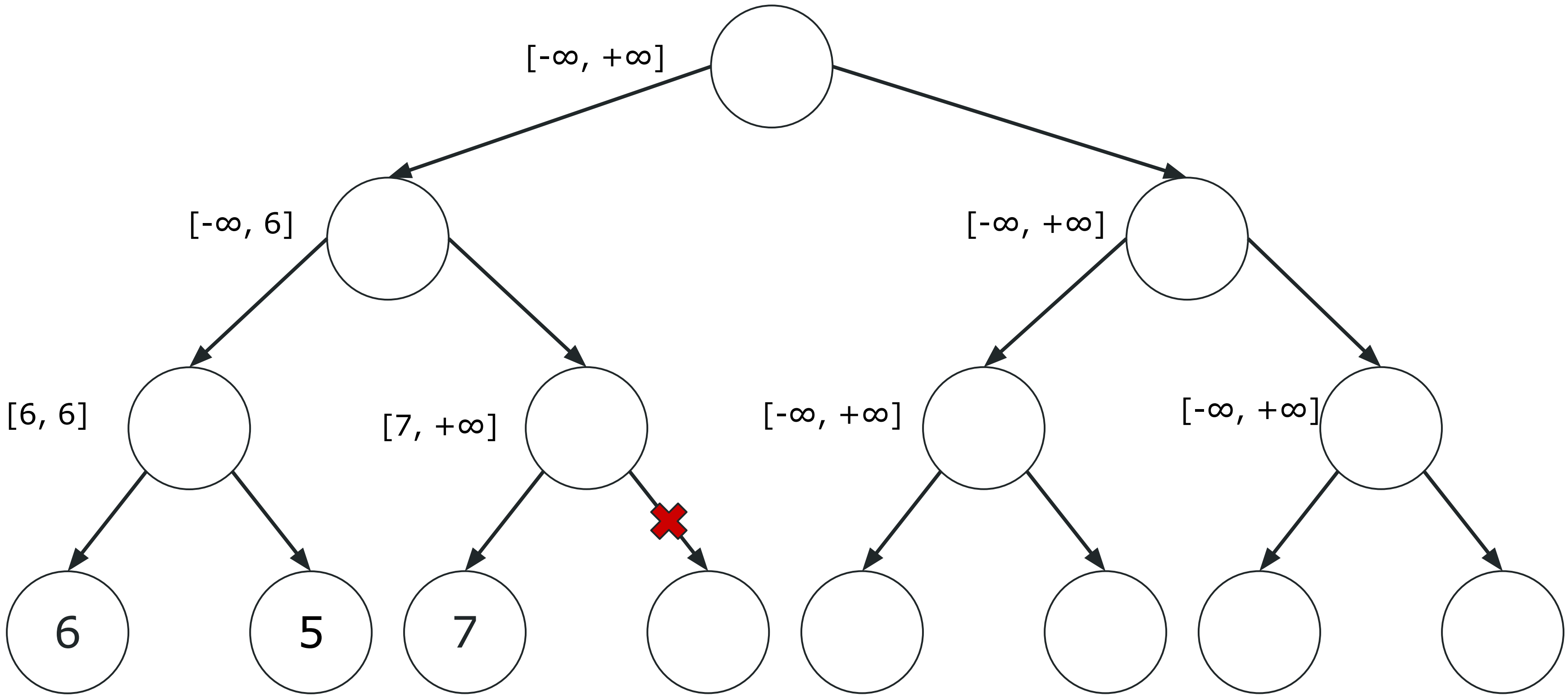
Alpha Beta Pruning Example 2



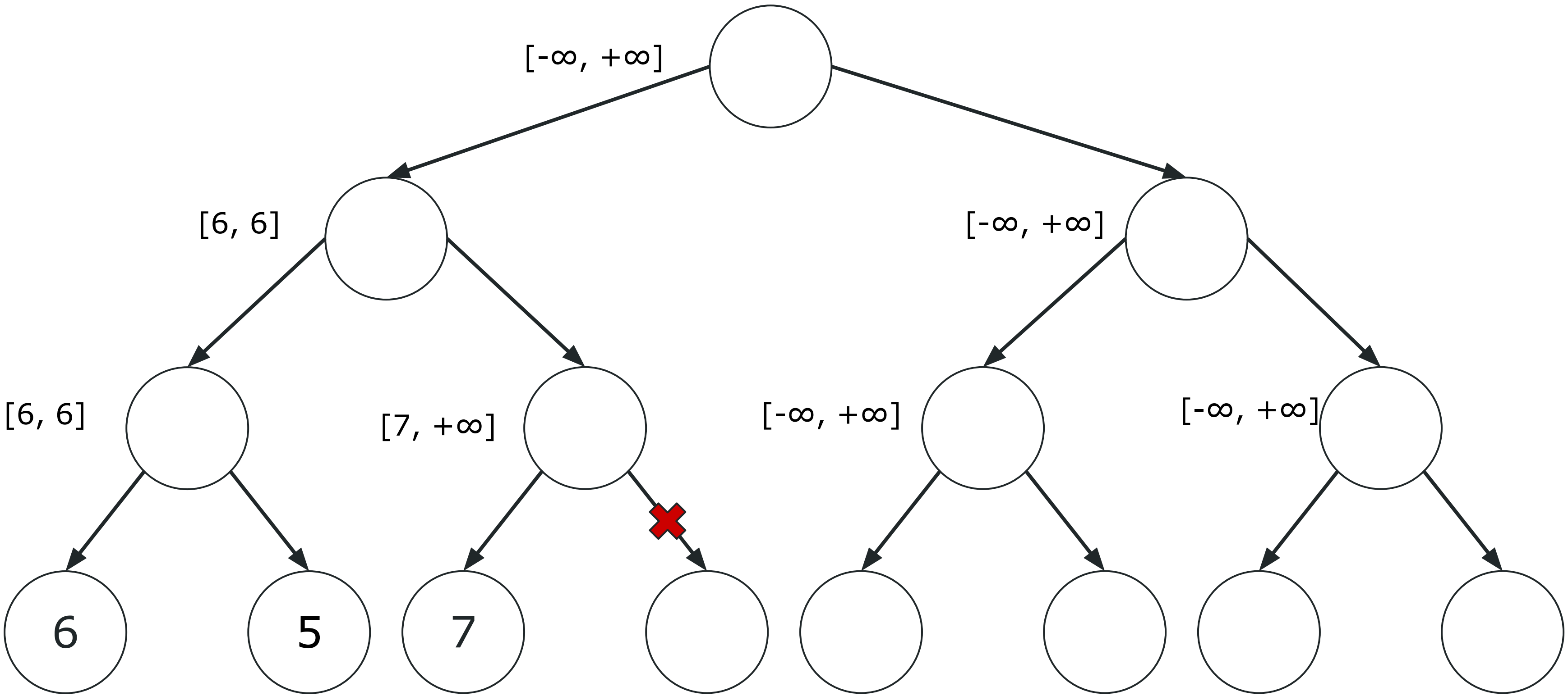
Alpha Beta Pruning Example 2



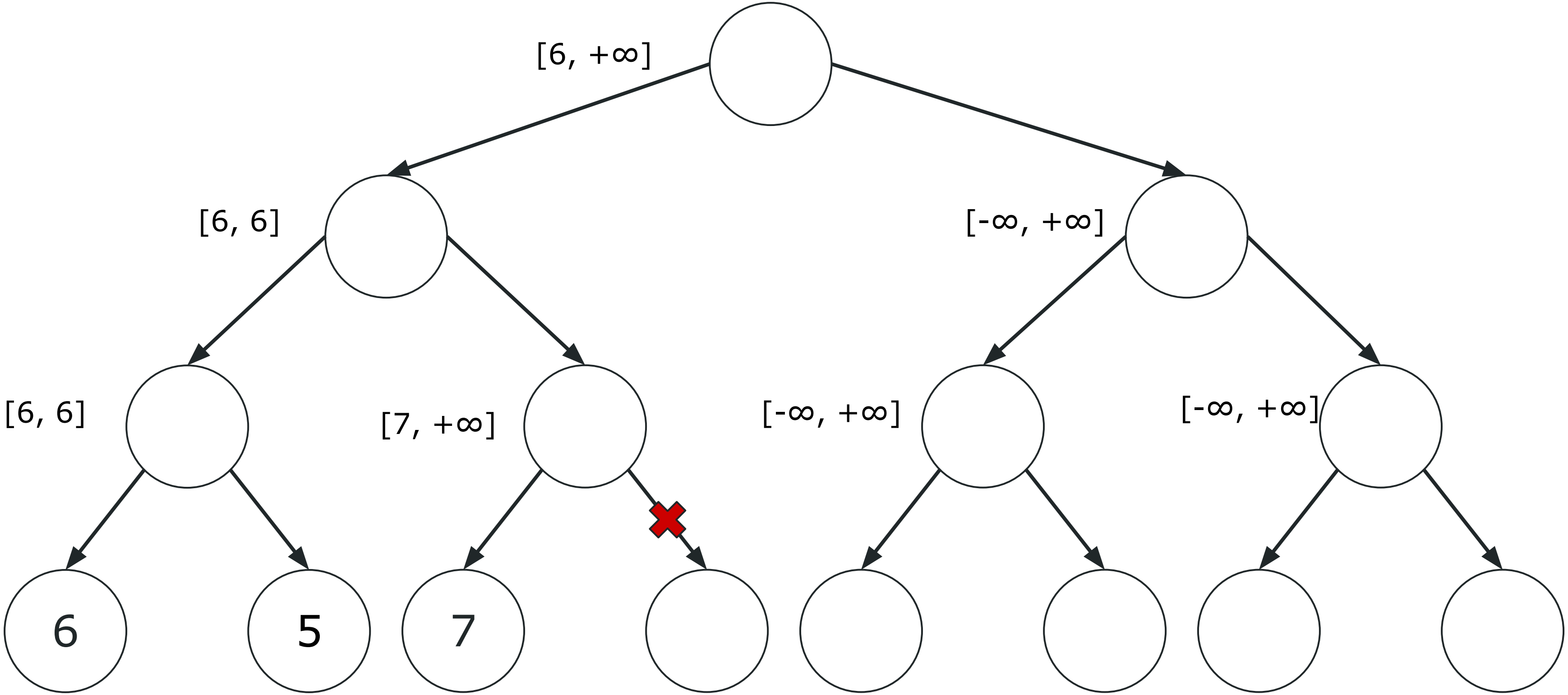
Alpha Beta Pruning Example 2



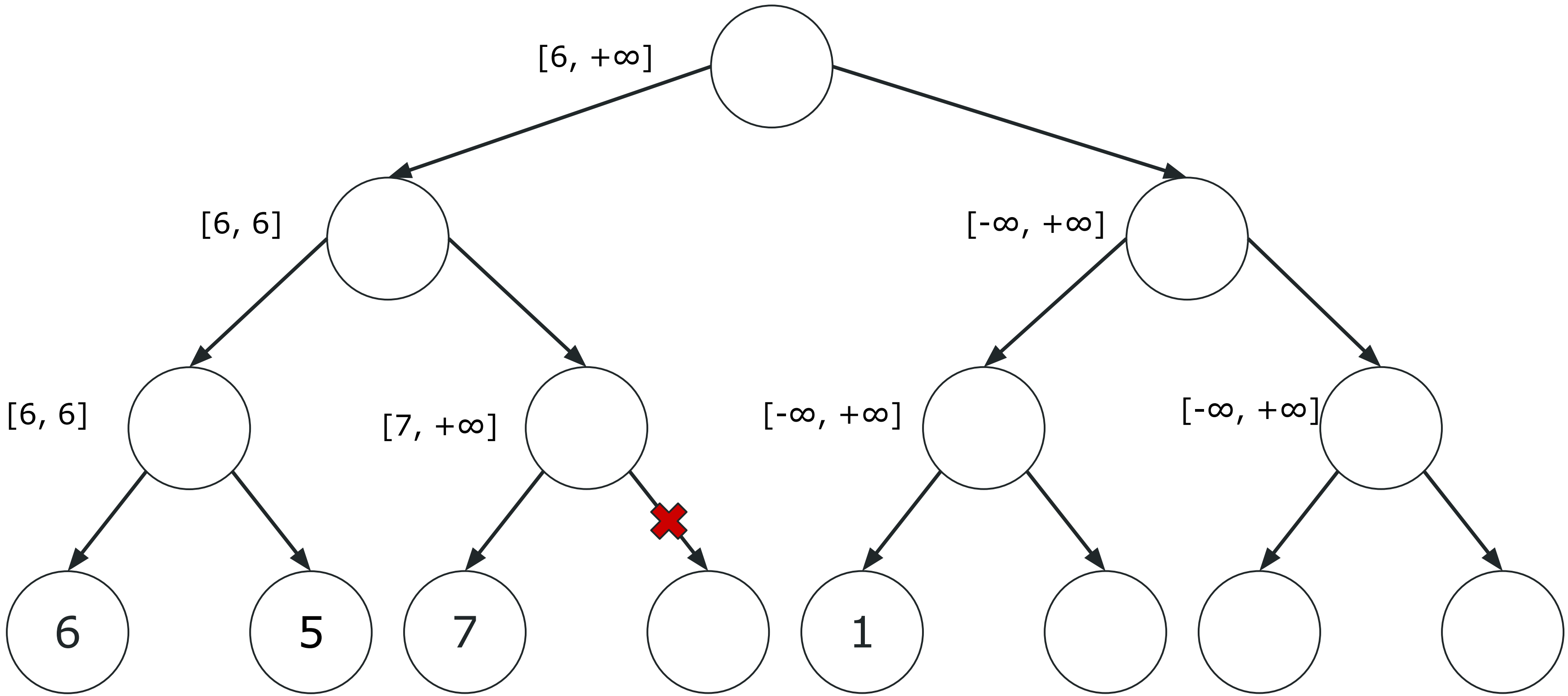
Alpha Beta Pruning Example 2



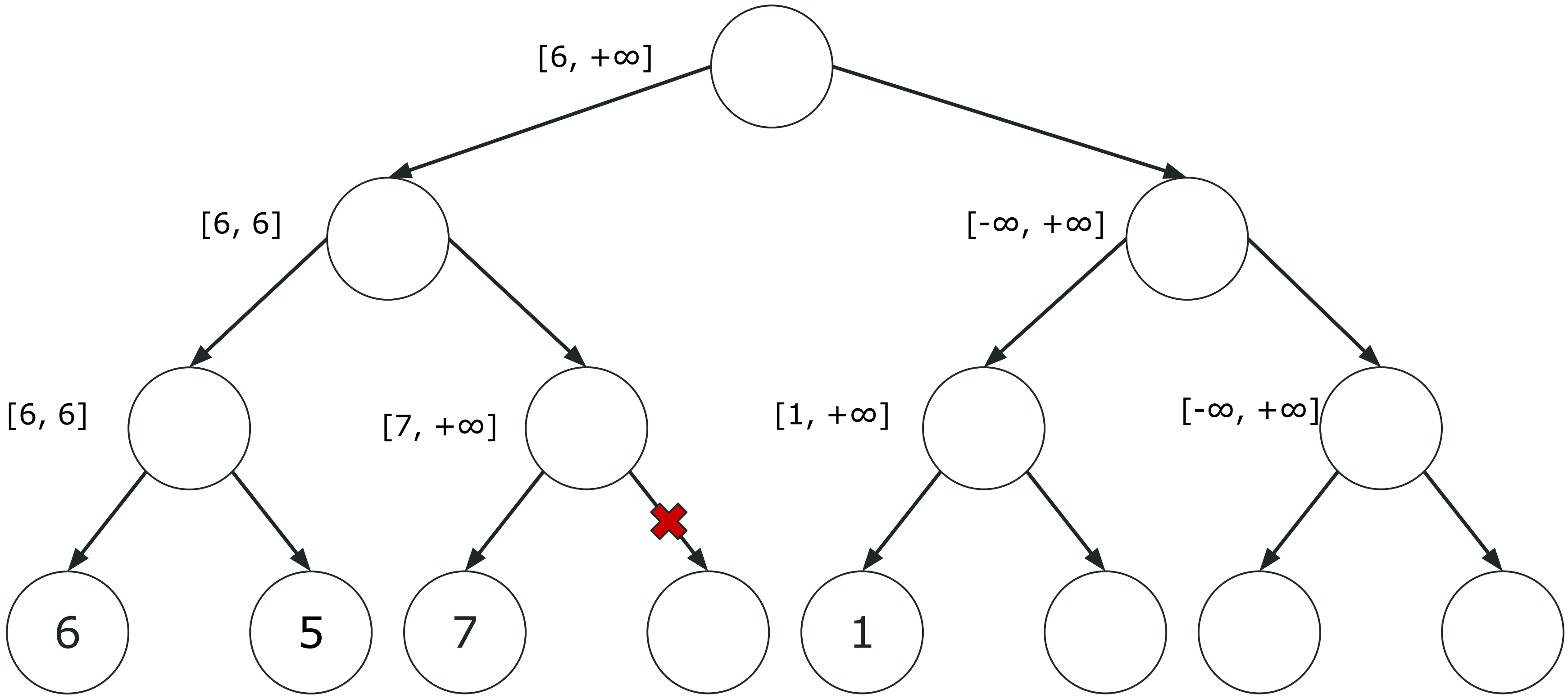
Alpha Beta Pruning Example 2



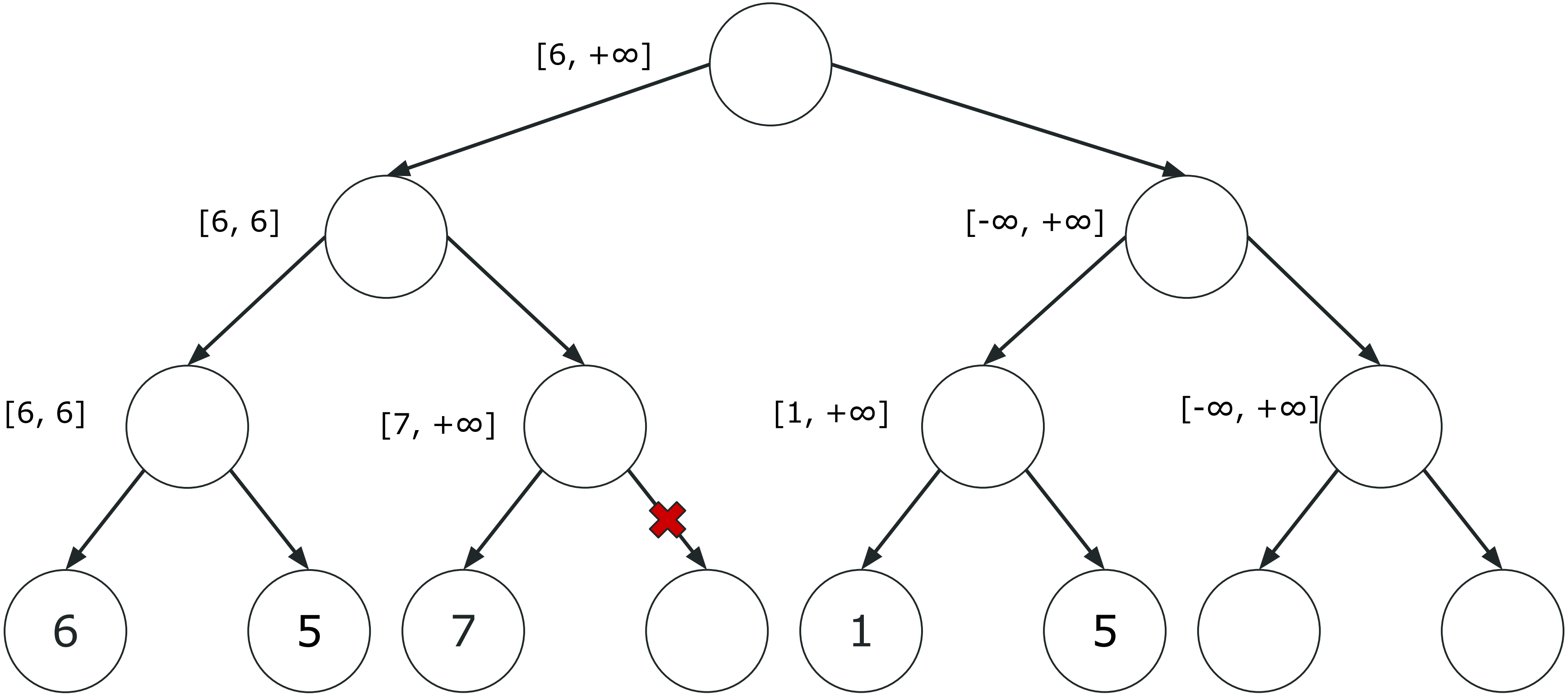
Alpha Beta Pruning Example 2



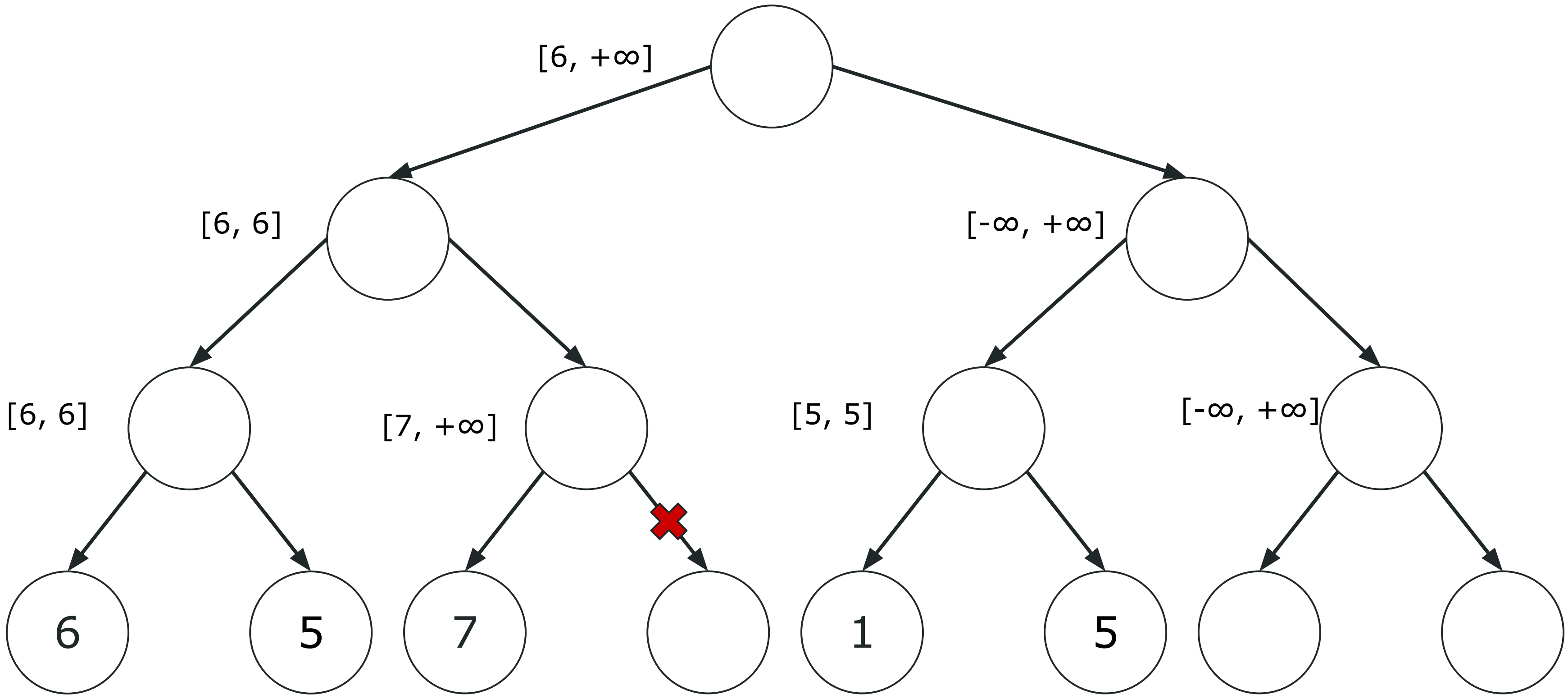
Alpha Beta Pruning Example 2



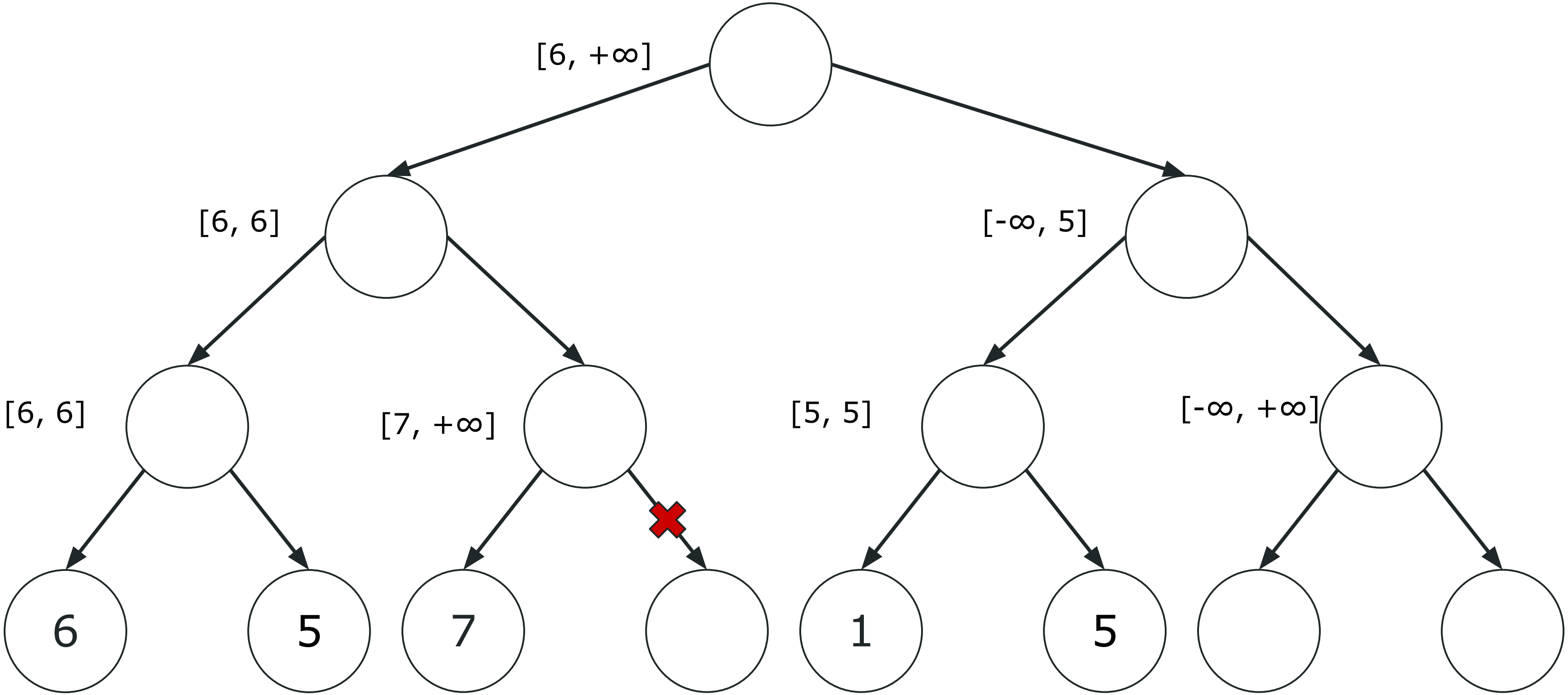
Alpha Beta Pruning Example 2



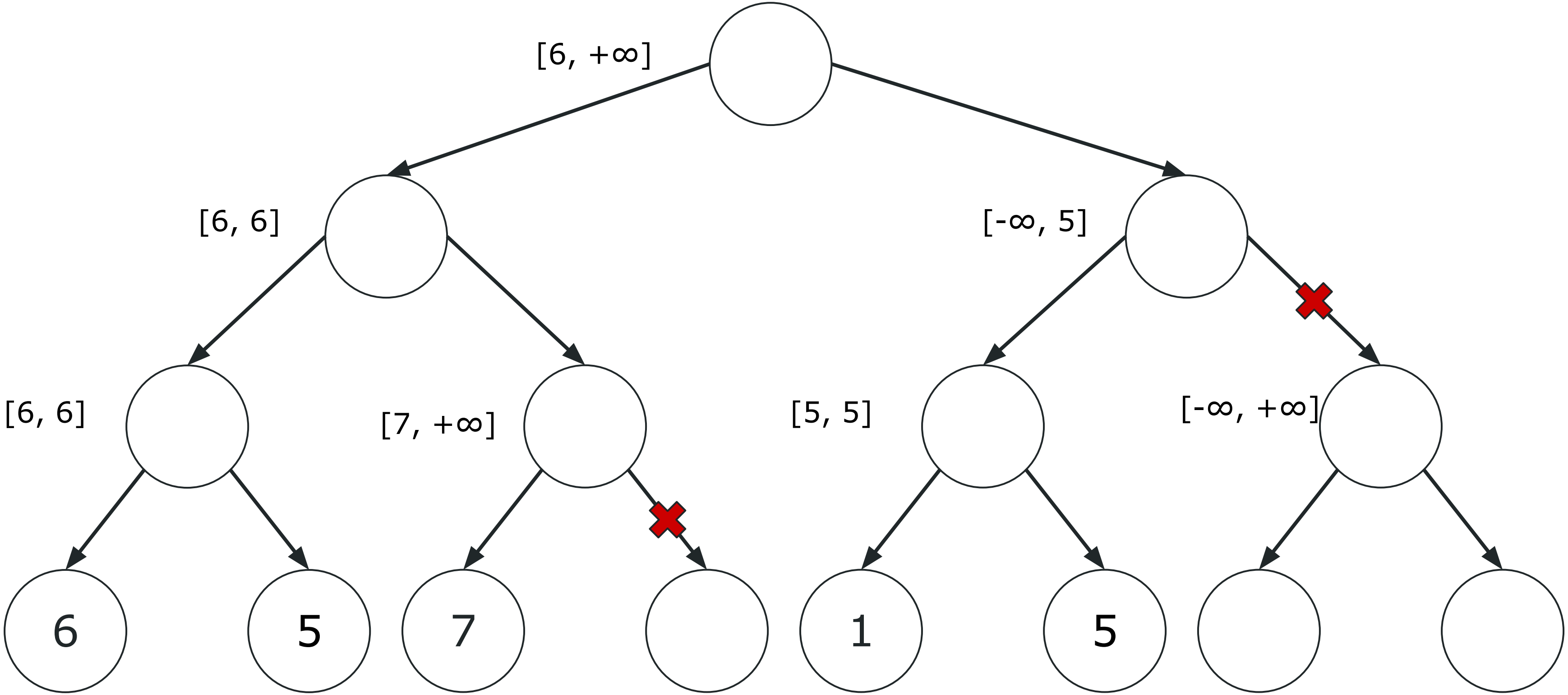
Alpha Beta Pruning Example 2



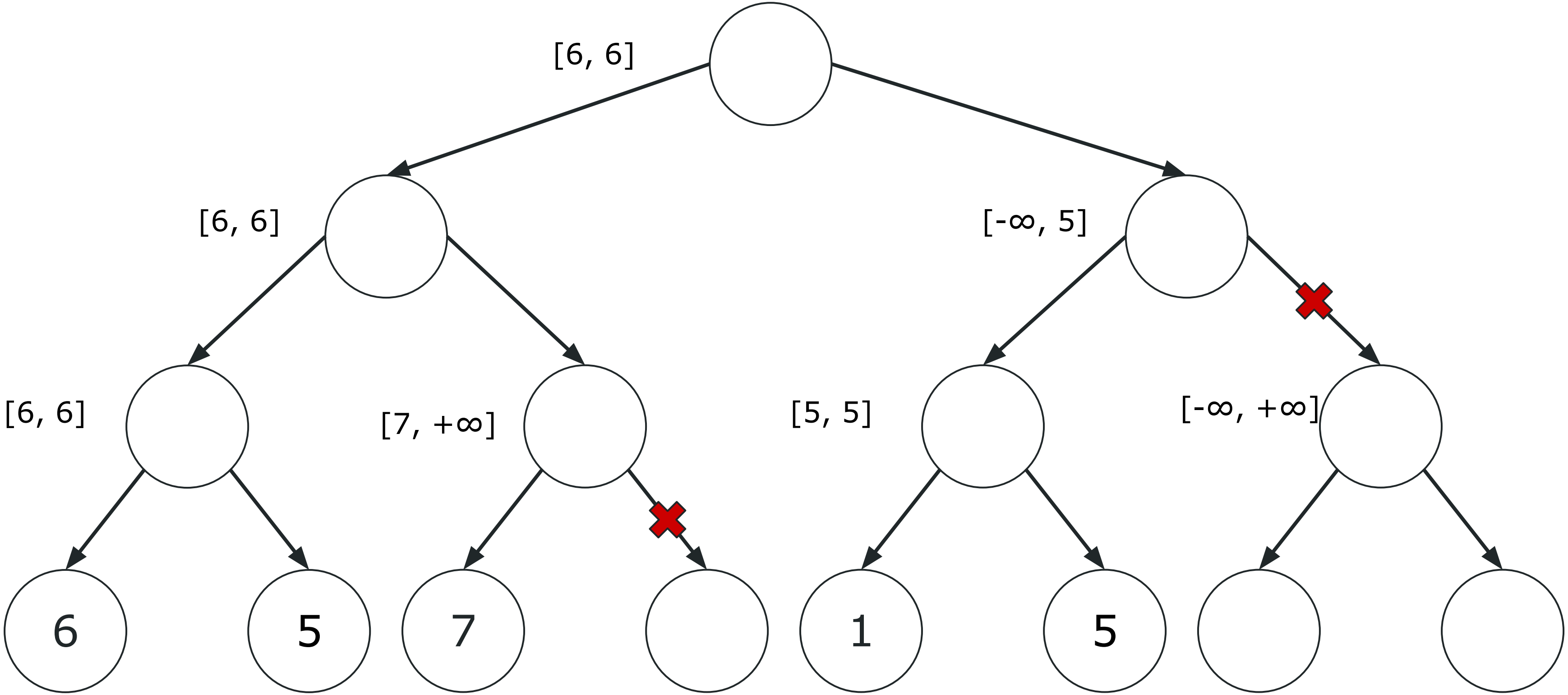
Alpha Beta Pruning Example 2



Alpha Beta Pruning Example 2



Alpha Beta Pruning Example 2



MiniMax vs. Alpha Beta Pruning Runtime

- MiniMax
  - Runtime: $O(b^h)$
  - Space: $O(bh)$
- Alpha Beta Pruning
  - Runtime:
    - Worst-Case: $O(b^h)$
    - Best-Case: $O(b^{h/2})$
  - Space: $O(bh)$

Why is the Worst-Case Runtime equal to MiniMax?

In the Worst-Case, your Alpha Beta is running MiniMax!

b = Branching Factor
h = Height of the Tree

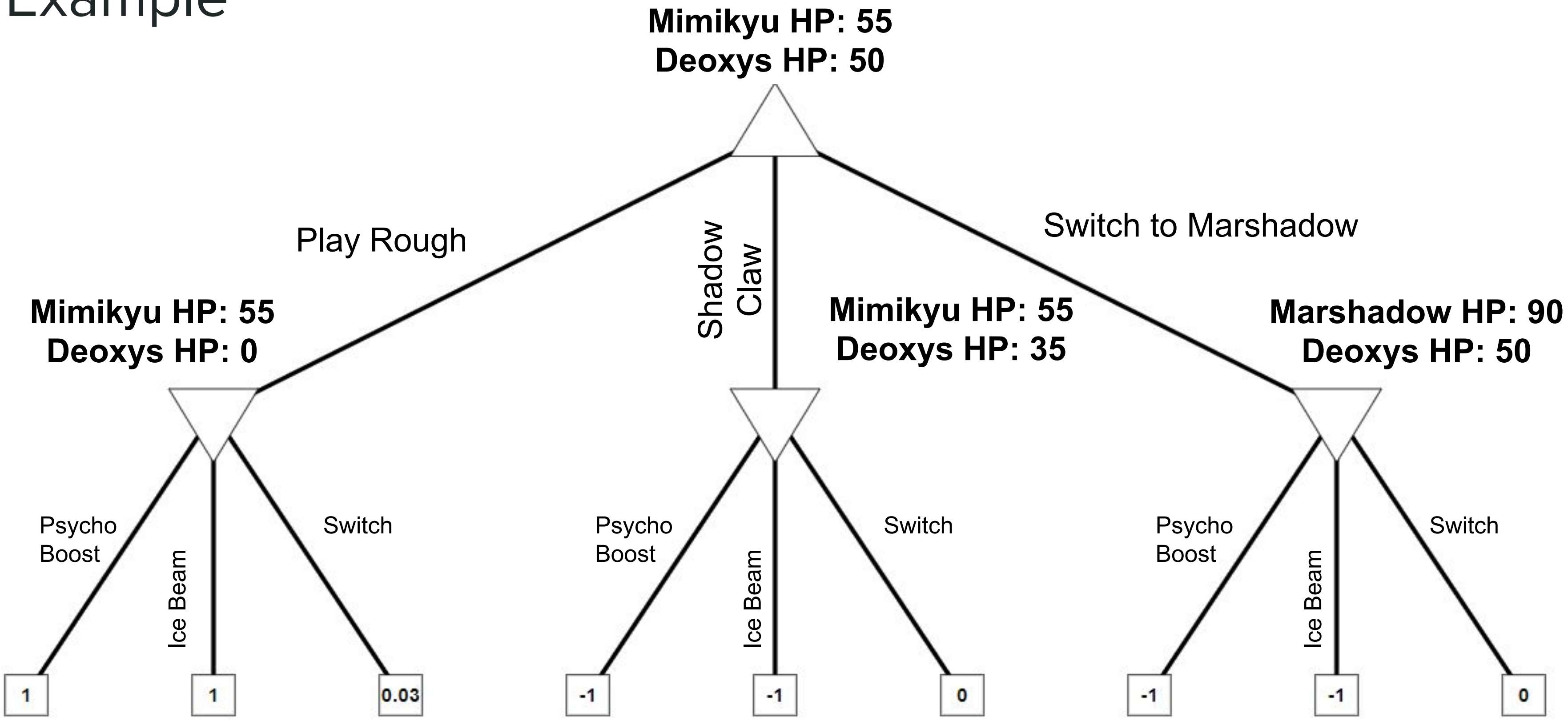
Alpha Beta for 2 Player Games

- Game Trees get really big really fast
  - Grows exponentially
  - Alpha Beta Pruning is more efficient than Minimax
- Used for many games
  - Tic-Tac-Toe
  - Chess
  - Go
- Heuristic is easily incorporated
  - A Heuristic is a mapping from a game state to a value
    - Ex: In Chess, White Pieces - Black Pieces = Value
      - This is a bad heuristic to use
  - We use heuristics when we do not want calculate every end game state

“Real Life” Use: Pokemon

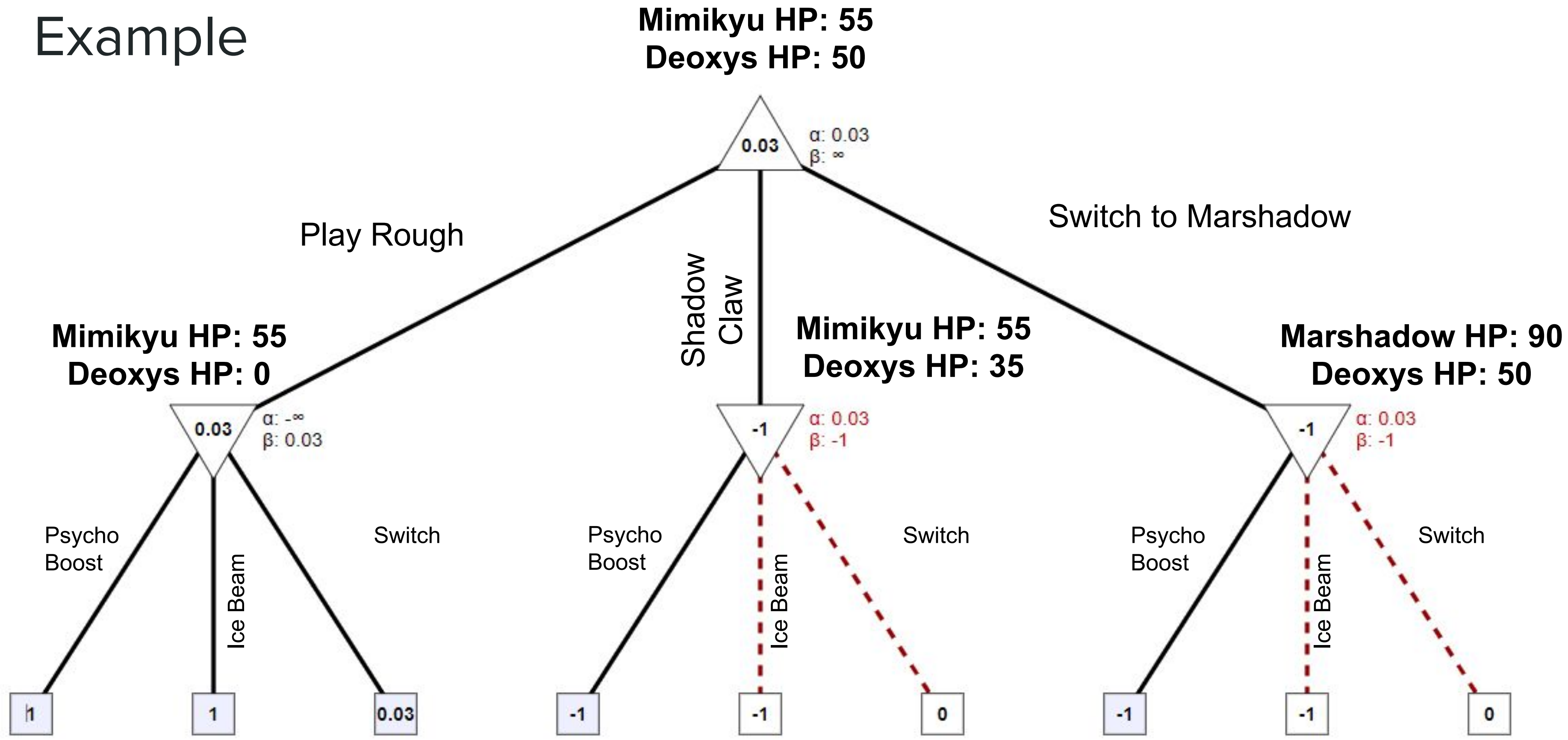
- I created a AI simulation that simulates a competitive battling scenario
  - Used Java
  - Dictionary of Pokemon
  - Dictionary of Moves
  - Battle Game Tree
  - Alpha Beta Pruning to Traverse tree
  - Minimax to Check Alpha Beta
  - 12 different classes

Example



Mimikyu HP: 55
Deoxys HP: 50
Play Rough
Shadow Claw
Switch to Marshadow
Mimikyu HP: 55
Deoxys HP: 0
Mimikyu HP: 55
Deoxys HP: 35
Marshadow HP: 90
Deoxys HP: 50
Psycho Boost
Ice Beam
Switch
Psycho Boost
Ice Beam
Switch
Psycho Boost
Ice Beam
Switch
1
1
0.03
-1
-1
0
-1
-1
0

Example



Mimikyu HP: 55
Deoxys HP: 50
0.03
α: 0.03
β: ∞
Play Rough
Shadow Claw
Switch to Marshadow
Mimikyu HP: 55
Deoxys HP: 0
0.03
α: -∞
β: 0.03
Mimikyu HP: 55
Deoxys HP: 35
-1
α: 0.03
β: -1
Marshadow HP: 90
Deoxys HP: 50
-1
α: 0.03
β: -1
Psycho Boost
Ice Beam
Switch
Psycho Boost
Ice Beam
Switch
Psycho Boost
Ice Beam
Switch
1
1
0.03
-1
-1
0
-1
-1
0