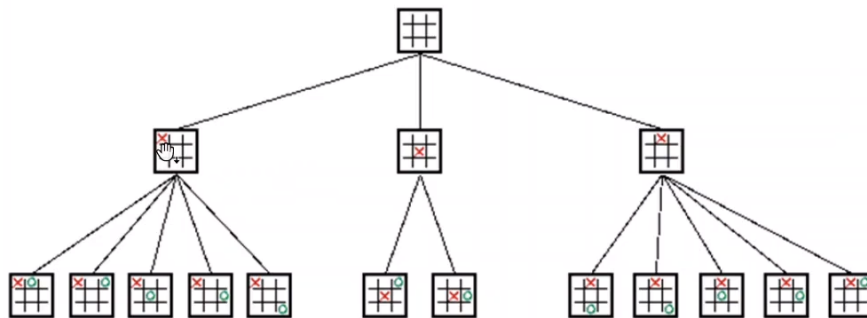
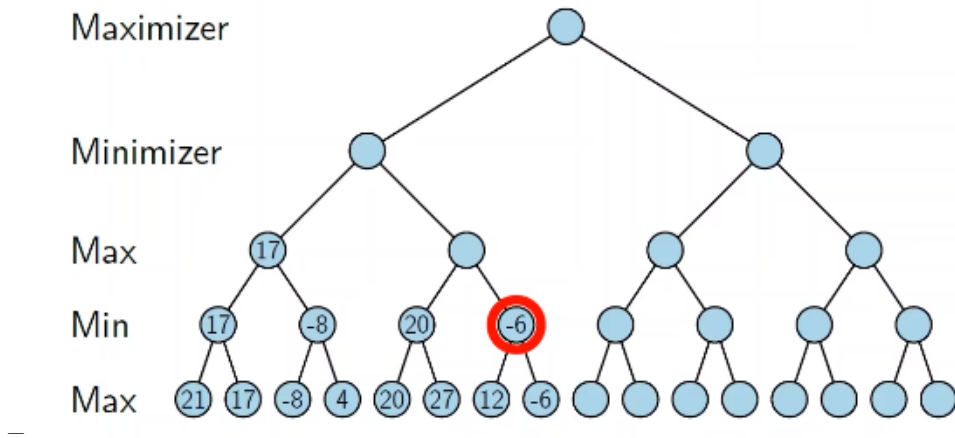


Classical Approach

- starting position as root
 - during game the current position
- states as nodes
- possible moves as edges to child nodes
 - states may be reached by multiple move sequences
 - performance improvement possible
 - * no longer a tree
- tree depth is bounded
 - exponential overhead
 - combinatorial explosion
- half move
 - own move + opponents move = full move
 - must also consider the opponents answer move
 - k half moves
 - * $\lfloor \frac{k}{2} \rfloor (+1)$ moves per player

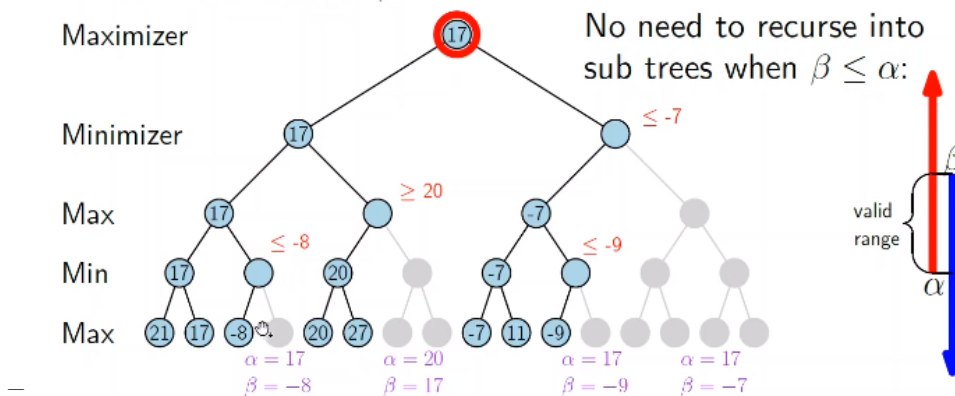


- - not shown possible moves are symmetrical
- leaves nodes are evaluated using heuristics
 - heuristic scores the game state
 - terminal states (win, lose) have extreme values
 - non-leave nodes evaluated by traversing the tree downwards
 - * chooses maximum/minimum score of available moves
- maximizer and minimizer
 - one player wants to maximize the value of all child nodes
 - other player wants to minimize the value of all child nodes



α - β Pruning

- prevents combinatorial explosion
 - reduces states to consider
 - combines states with identical outcome
 - more efficient and equal results as the classical approach
 - stops traversing tree downwards upon reaching
 - a minimum value smaller than current maximum
 - a maximum value larger than current minimum
- α : Lower bound for maximizer: no need to consider states with scores below α .
- β : Upper bound for minimizer: no need to consider states with scores above β .



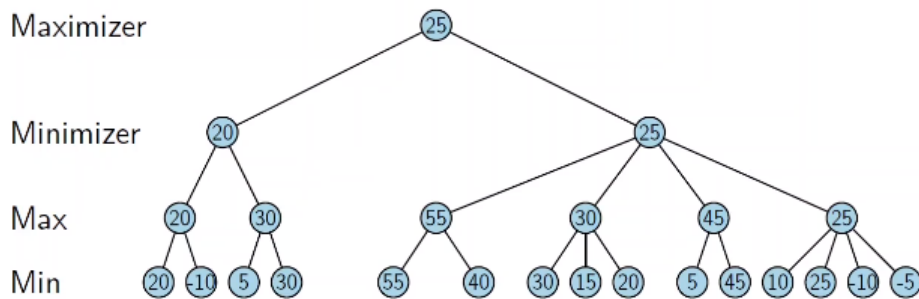
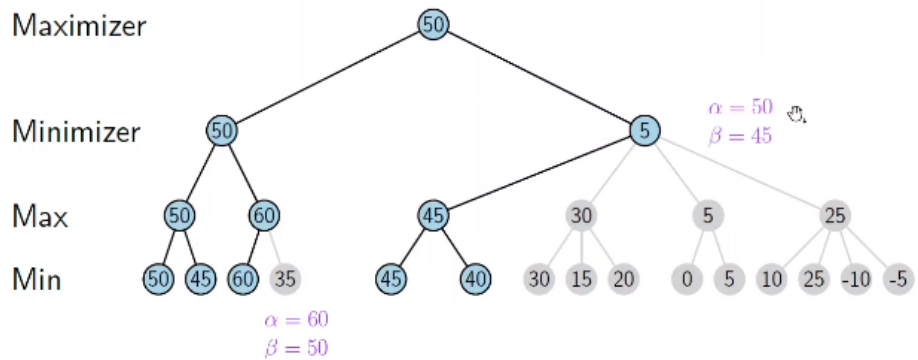
- pseudo code

```

evaluate (node, alpha, beta)
  if node is a leaf
    return (heuristic value of node)
  if node is a minimizing node
    for each child of node
      beta = min (beta, evaluate (child, alpha, beta))
      if beta <= alpha
        return (alpha)
    return (beta)
  if node is a maximizing node
    for each child of node
      alpha = max (alpha, evaluate (child, alpha, beta))
      if beta <= alpha
        return (beta)
    return (alpha)
- evaluate(root,  $-\infty, \infty$ );

```

- examples



How much can we save with α - β pruning?

Sometimes nothing!

- pre-sorted α - β pruning

– efficiency depends on order of states

Consider potentially best states first (large values for maximizer, small (large negativ) for minimizer)