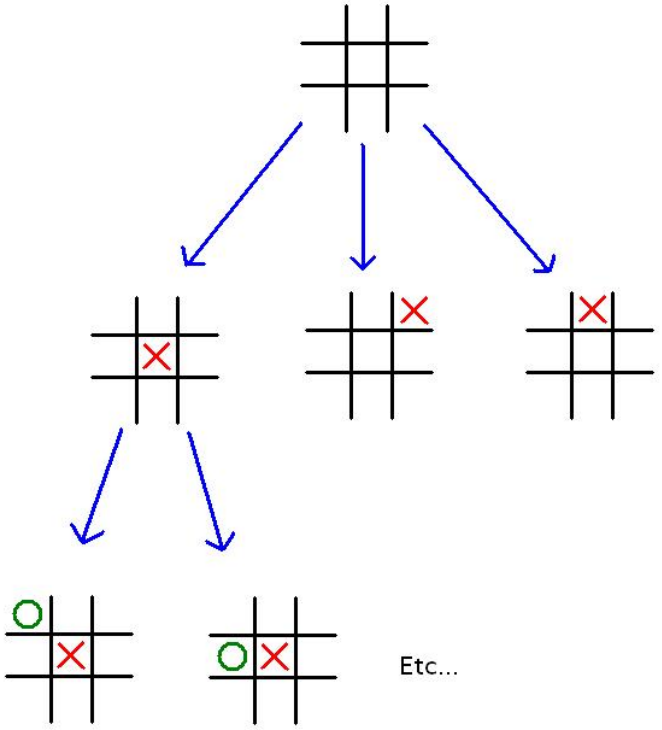
Game Tree Search



Slides By: Evan Ye

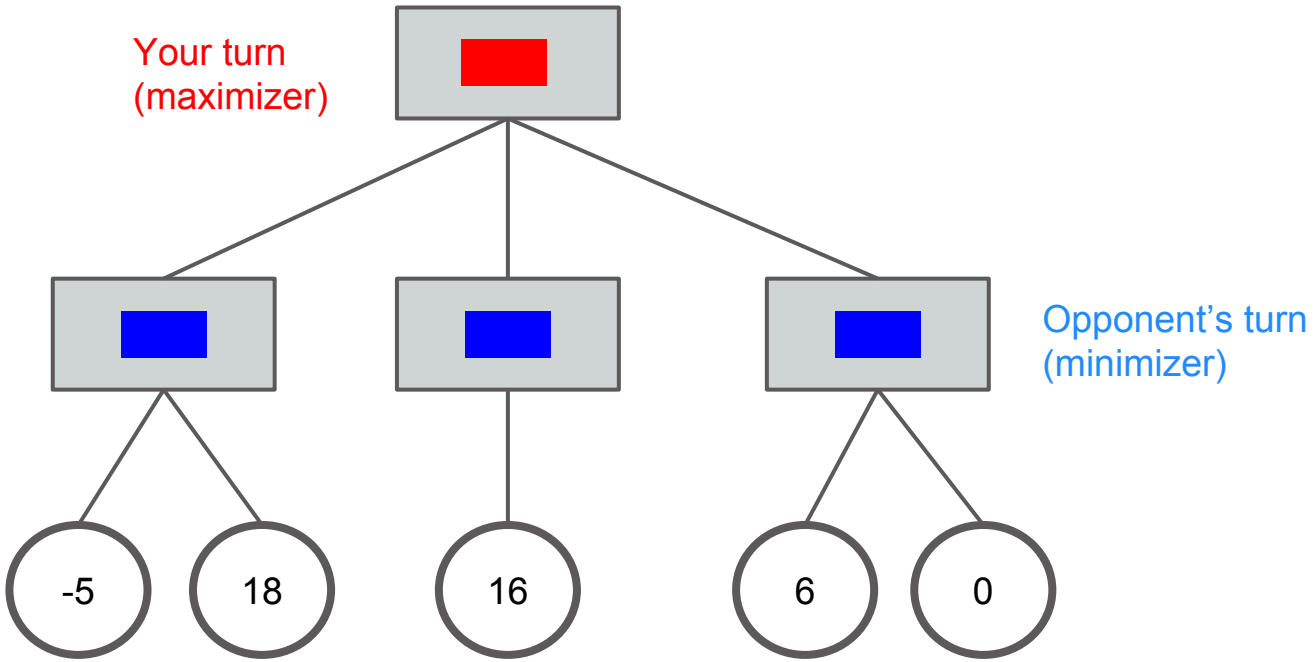
Game Tree Search

- Every position has a score
- The higher the score, the better for you, the worse for your opponent.
- The lower the score, the worse for you, the better for your opponent.

- Game tree search is the process of checking every possible move (tree recursion).

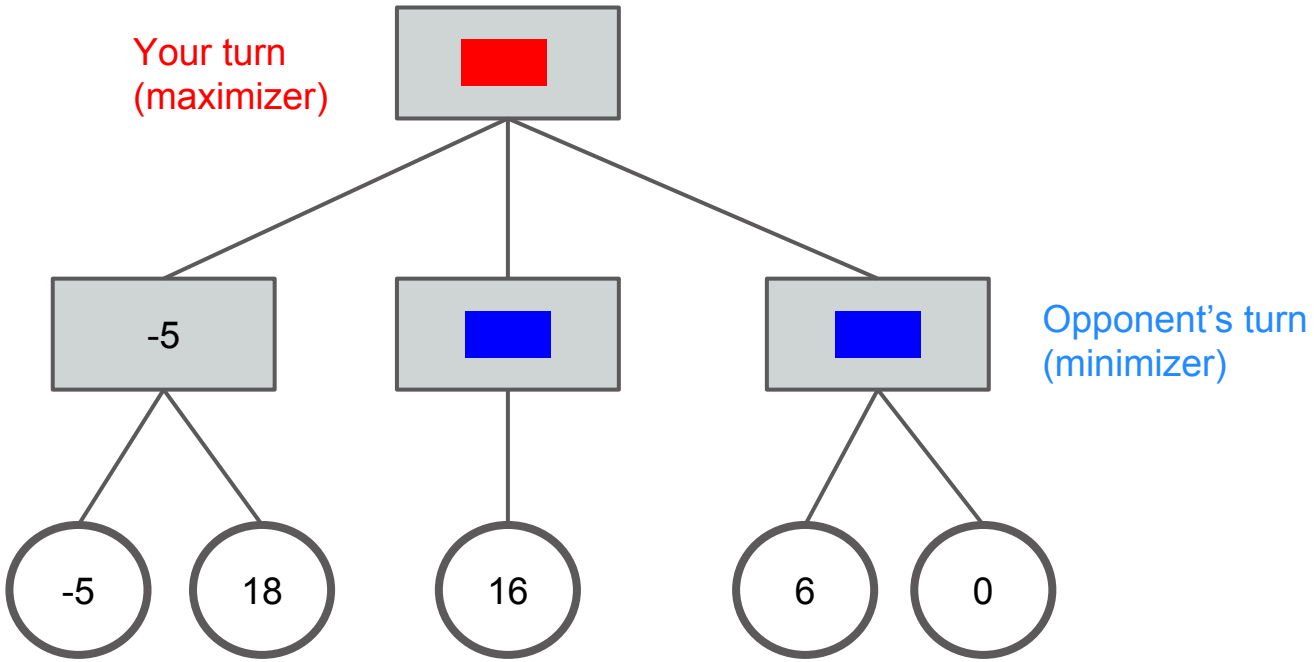
Minimax

Complete minimax on the tree below:



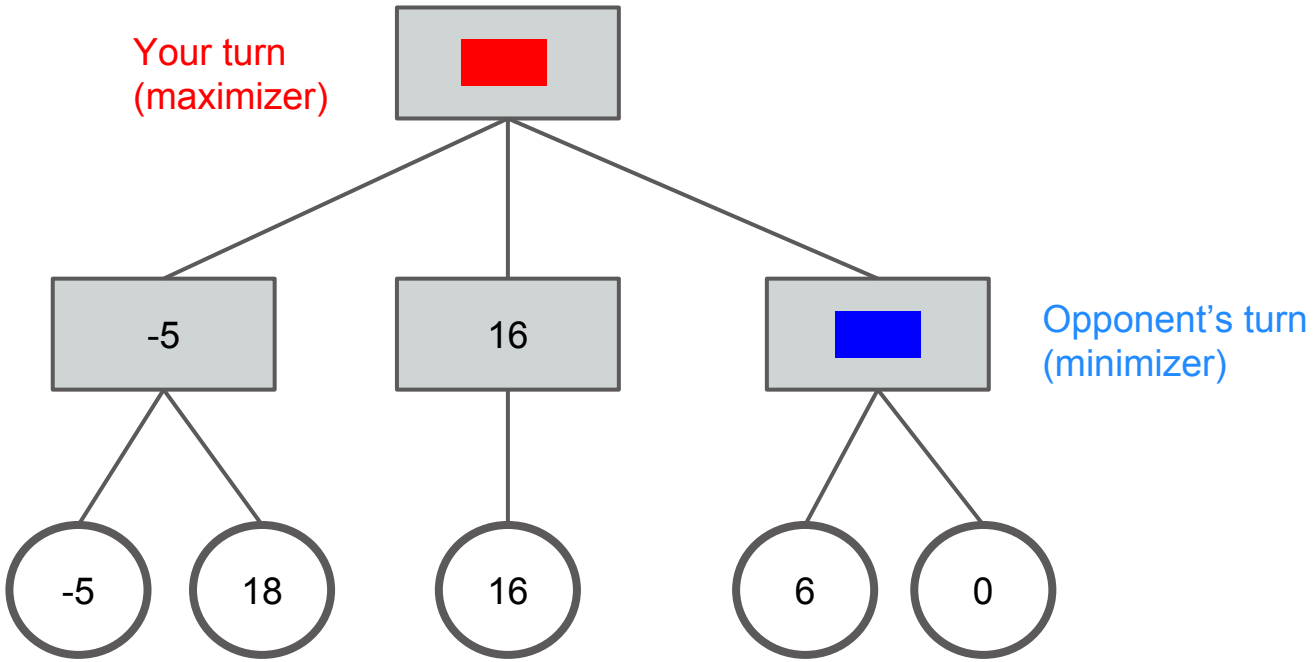
Minimax

Complete minimax on the tree below:



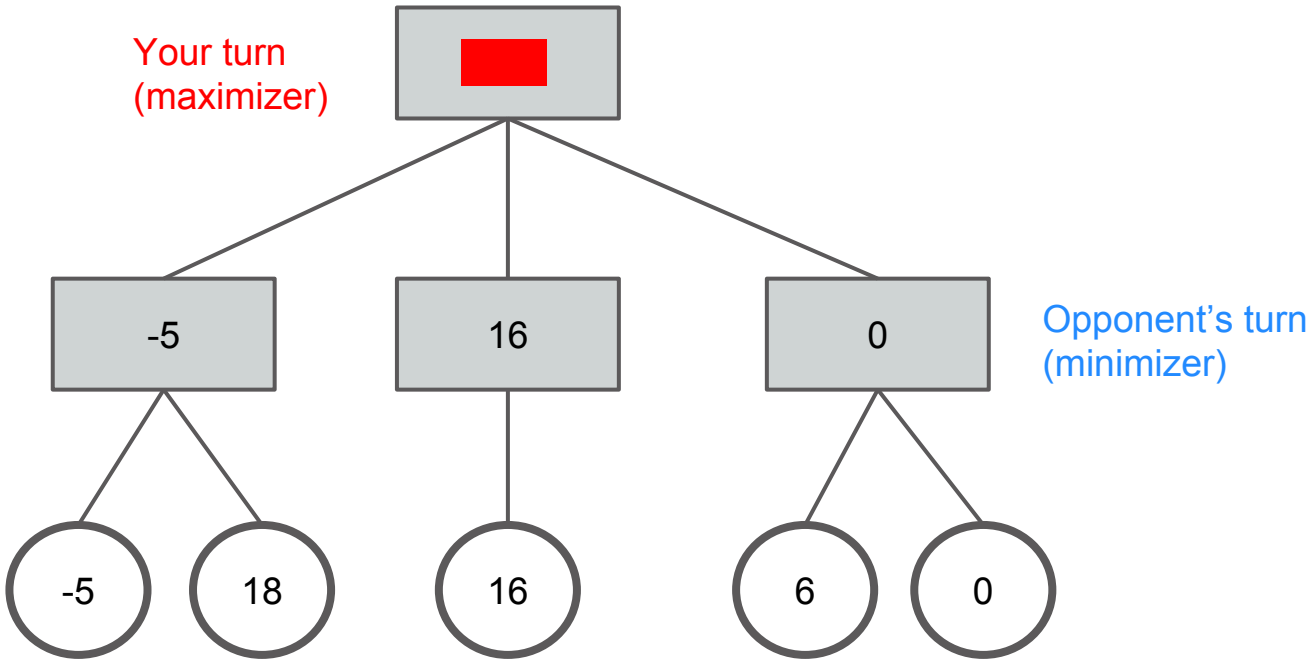
Minimax

Complete minimax on the tree below:



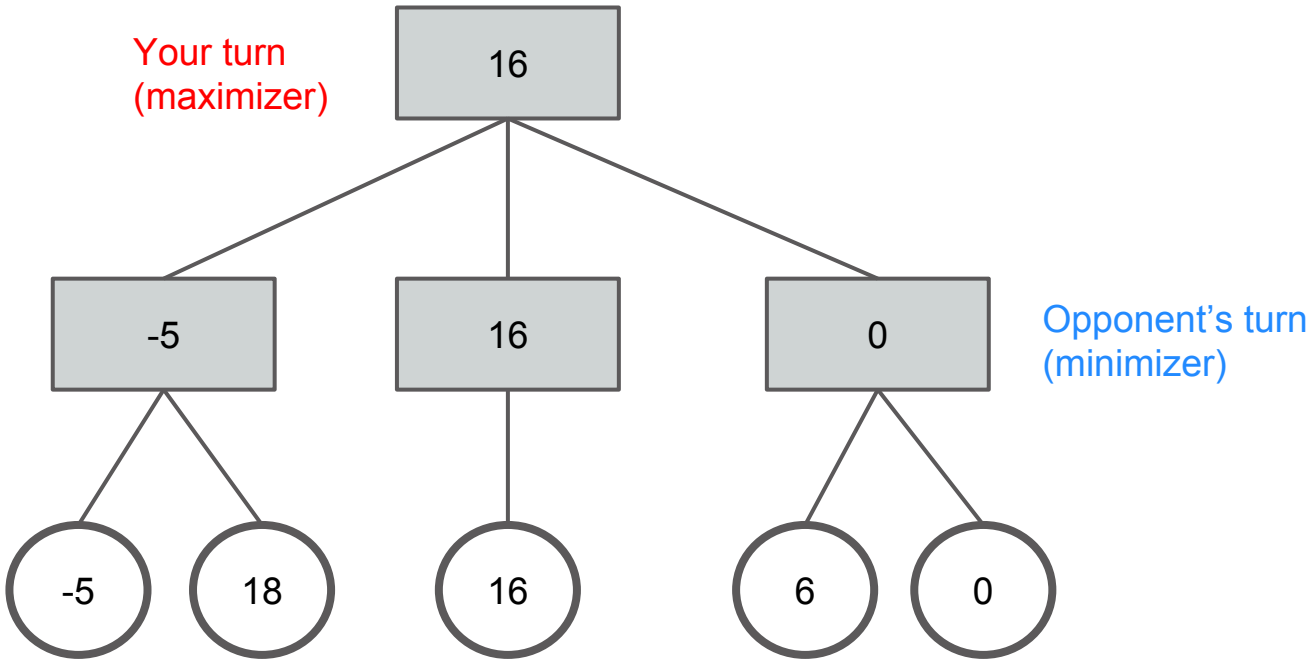
Minimax

Complete minimax on the tree below:



Minimax

Complete minimax on the tree below:



Alpha Beta Pruning

Idea: make minimax faster!

α - best guaranteed score seen so far for YOU

- YOU = maximizer node, YOU search for HIGHER scores
- Starts at negative infinity

β - best guaranteed score seen so far for your OPPONENT

- OPPONENT = minimizer node, searches for LOWER scores
- Starts at positive infinity

Prune Cases

- prune at a minimizer node whose β value is less than or equal to the α value

- prune at a maximizer node whose α value is greater than or equal to the β value

Alpha Beta Pruning

Idea: make minimax faster!

α - best guaranteed score seen so far for YOU

- YOU = maximizer node, YOU search for HIGHER scores
- Starts at negative infinity

β - best guaranteed score seen so far for your OPPONENT

- OPPONENT = minimizer node, searches for LOWER scores
- Starts at positive infinity

Prune Cases

Basically, when $\alpha \geq \beta$

Alpha Beta Pruning

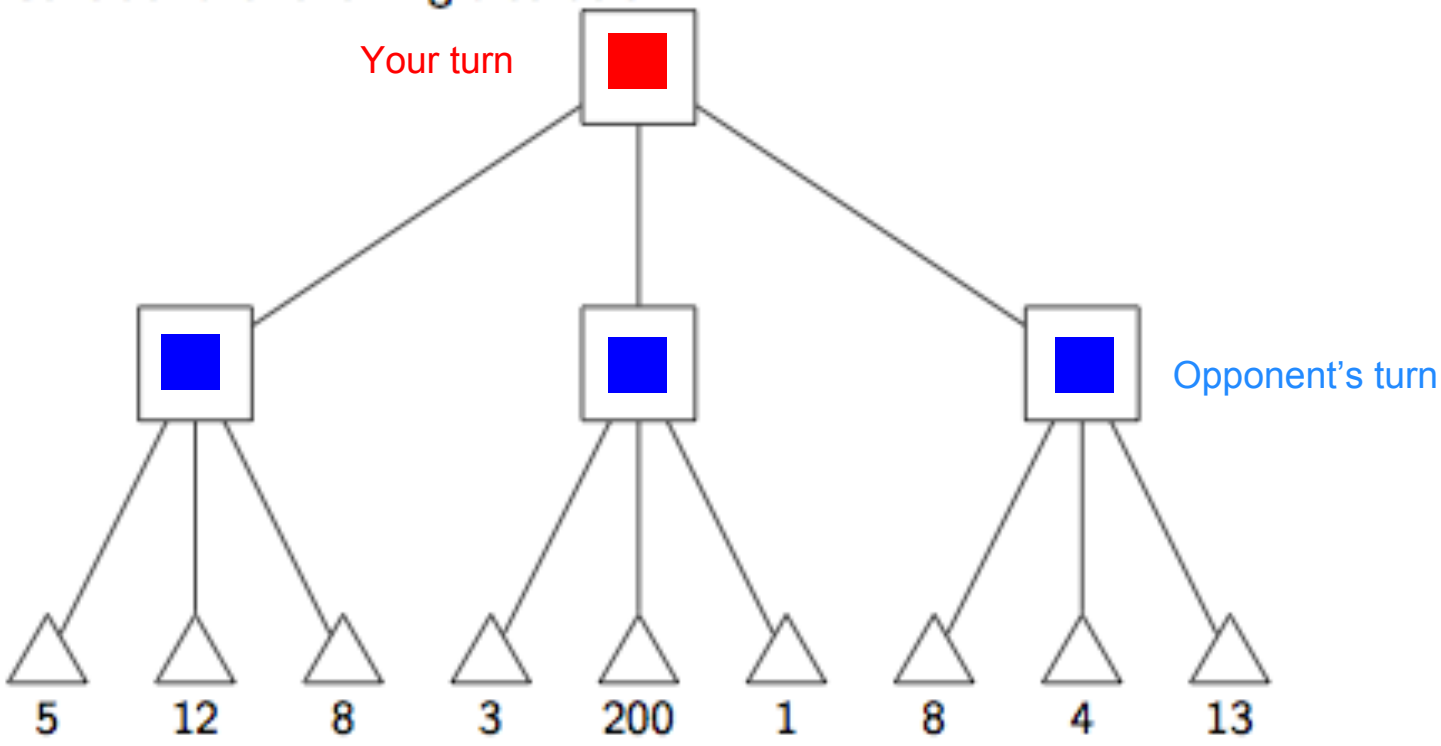
How to:

1. Do the minimax search
2. Keep track of alpha, beta
   a. if you are maximizing, update alpha
   b. if you are minimizing, update beta
3. If you ever see **α** >= **β**, PRUNE

Note*: **α** and **β** are INHERITED from parent nodes.

Alpha Beta Pruning

Consider the following tree below:

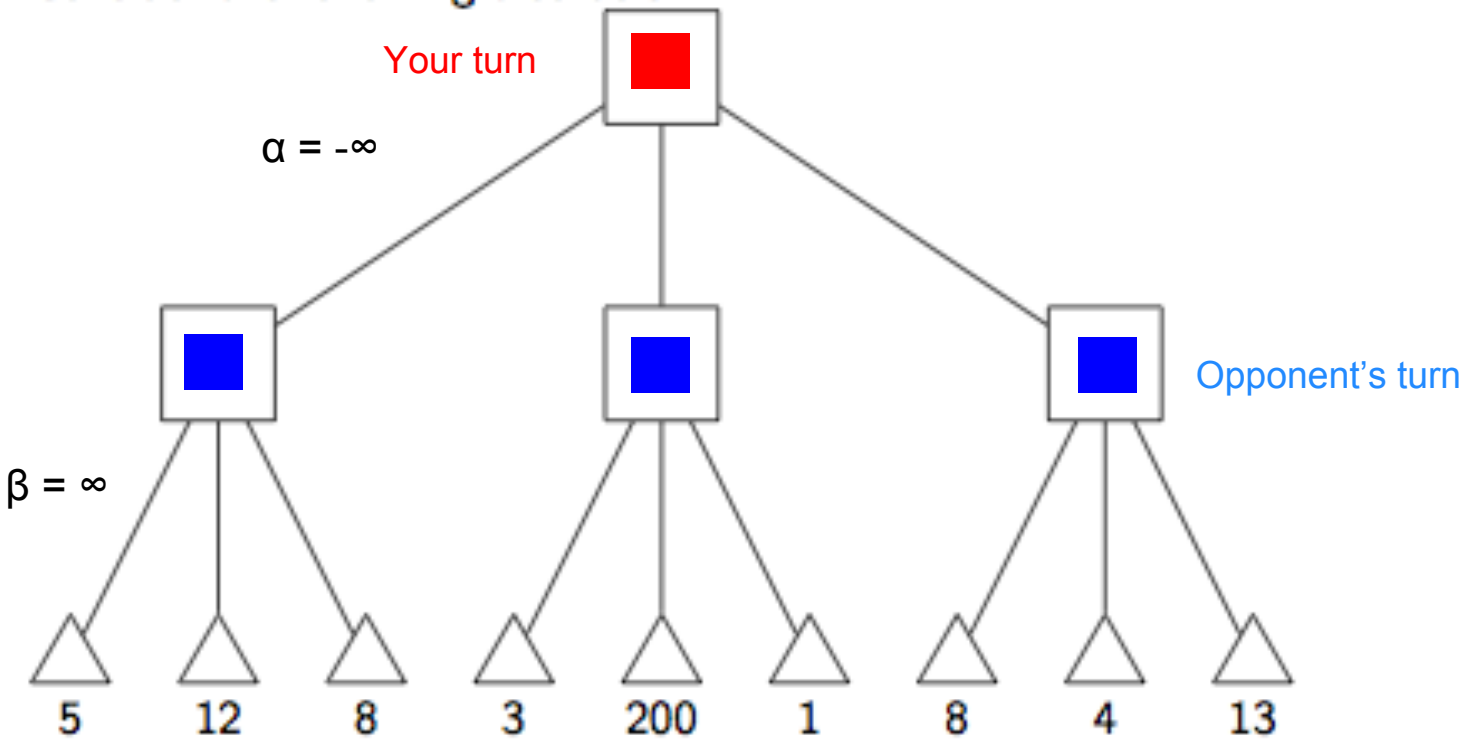


Determine which nodes will be pruned using α-β pruning and the final minimax value of the root. Assume the first player is MAX.

Also assume we traverse children from left to right.

Alpha Beta Pruning

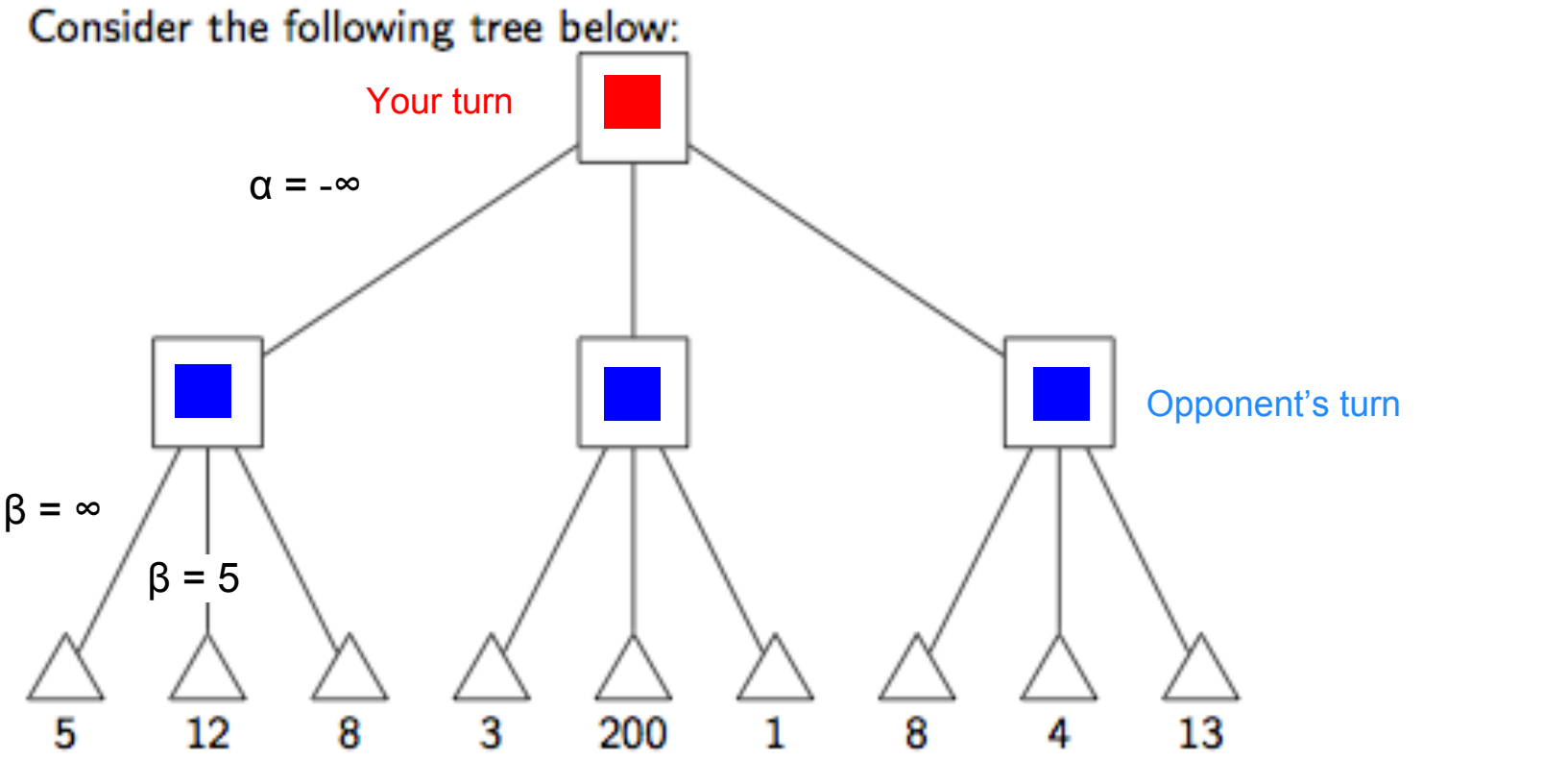
Consider the following tree below:



Determine which nodes will be pruned using α-β pruning and the final minimax value of the root. Assume the first player is MAX.

Also assume we traverse children from left to right.

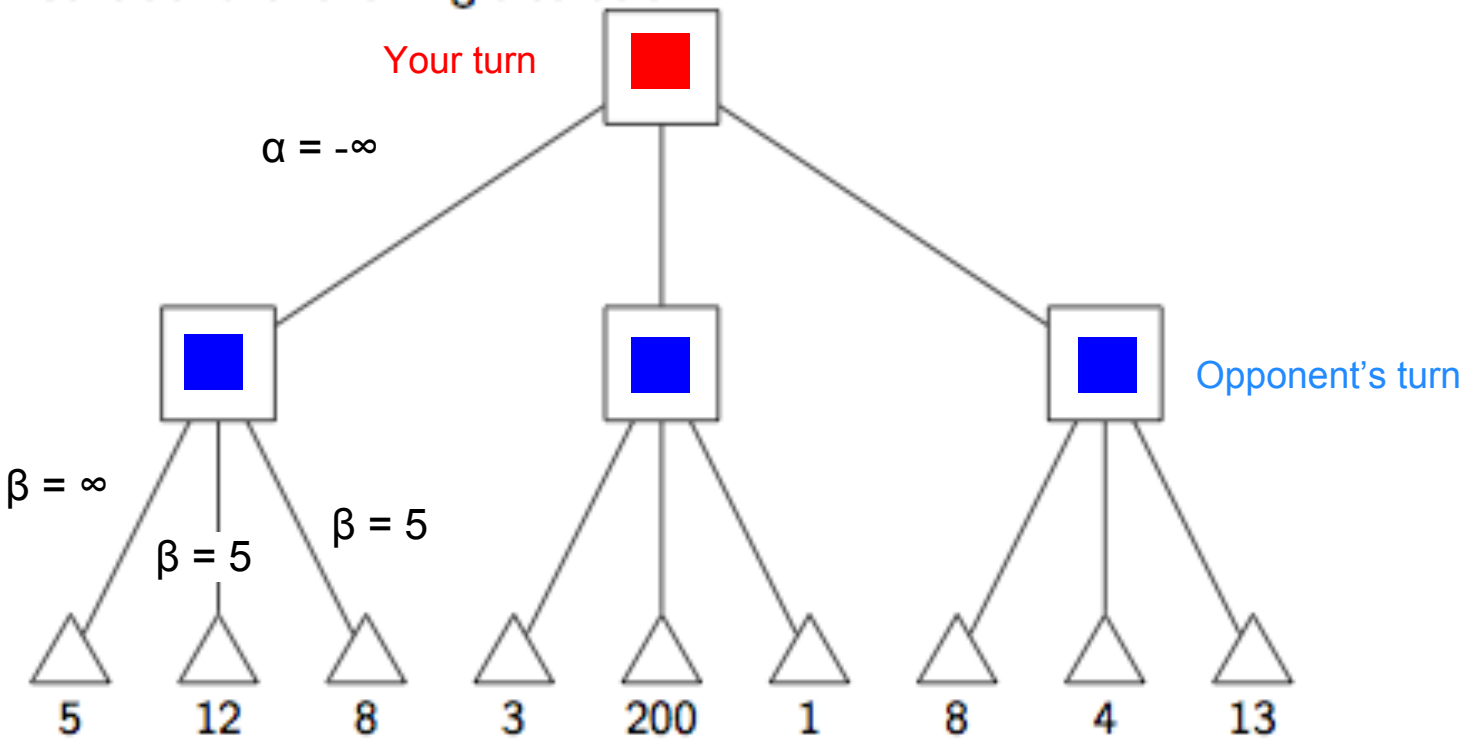
Alpha Beta Pruning



Also assume we traverse children from left to right.

Alpha Beta Pruning

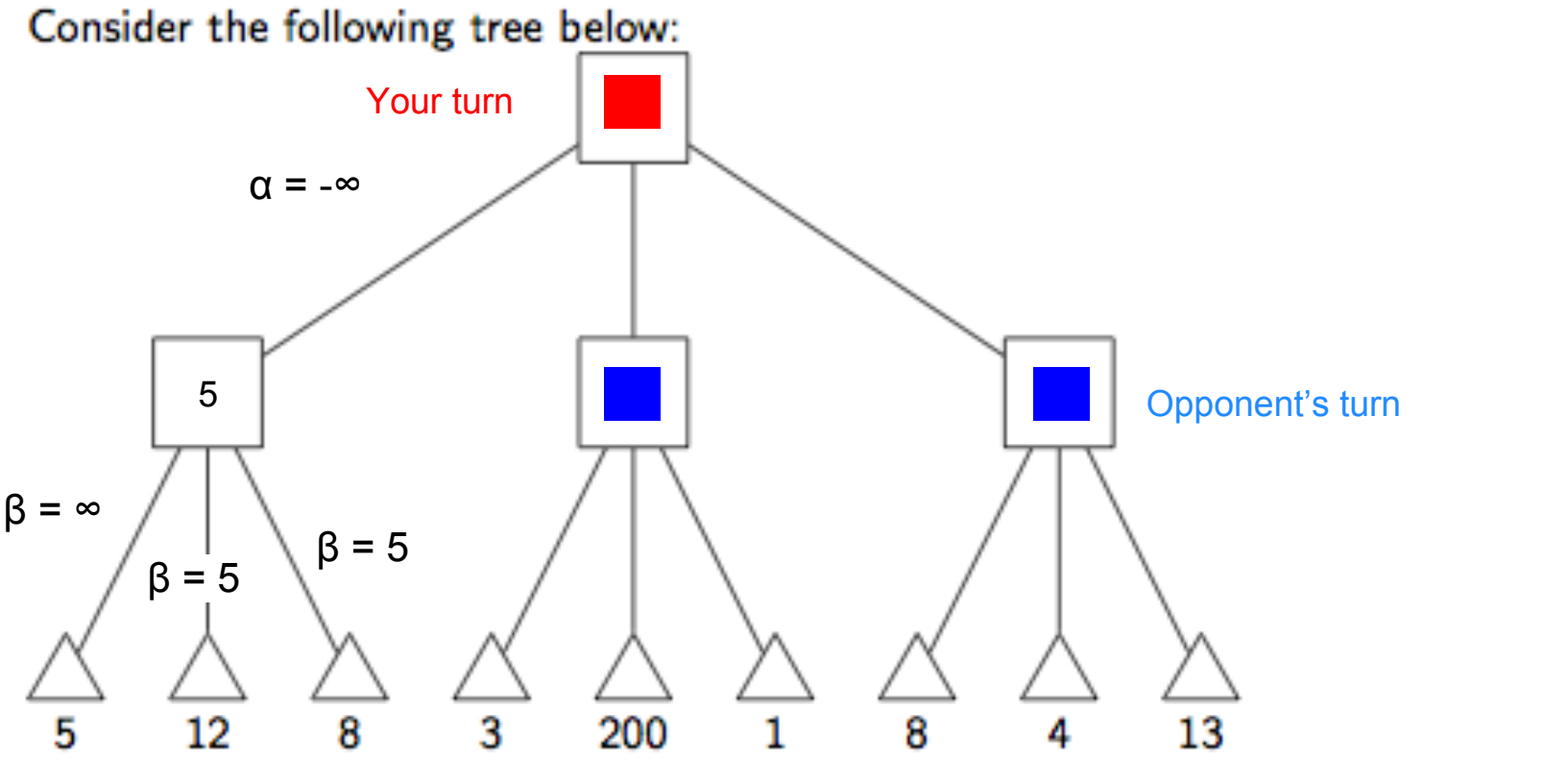
Consider the following tree below:



Determine which nodes will be pruned using α-β pruning and the final minimax value of the root. Assume the first player is MAX.

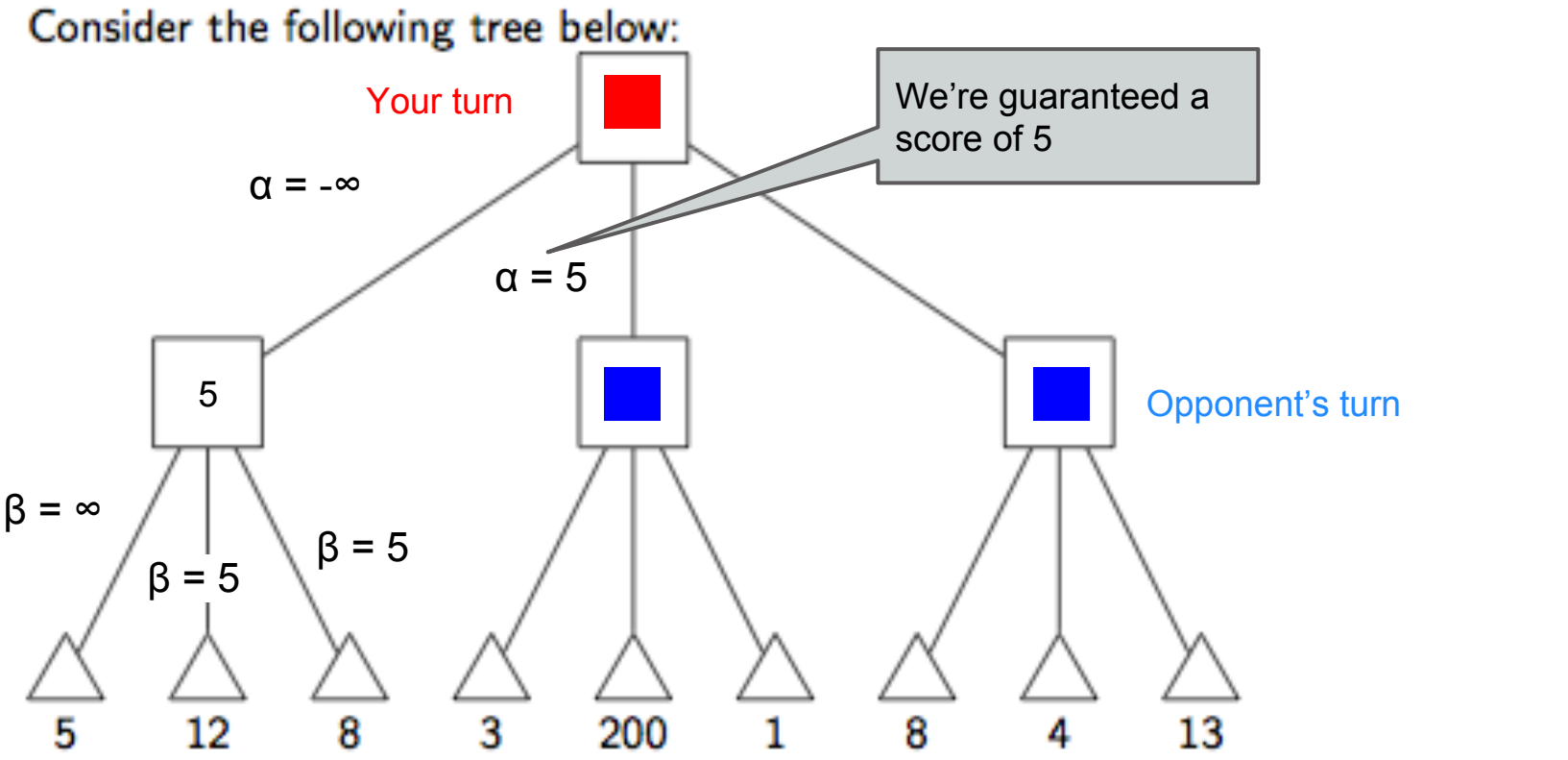
Also assume we traverse children from left to right.

Alpha Beta Pruning



Also assume we traverse children from left to right.

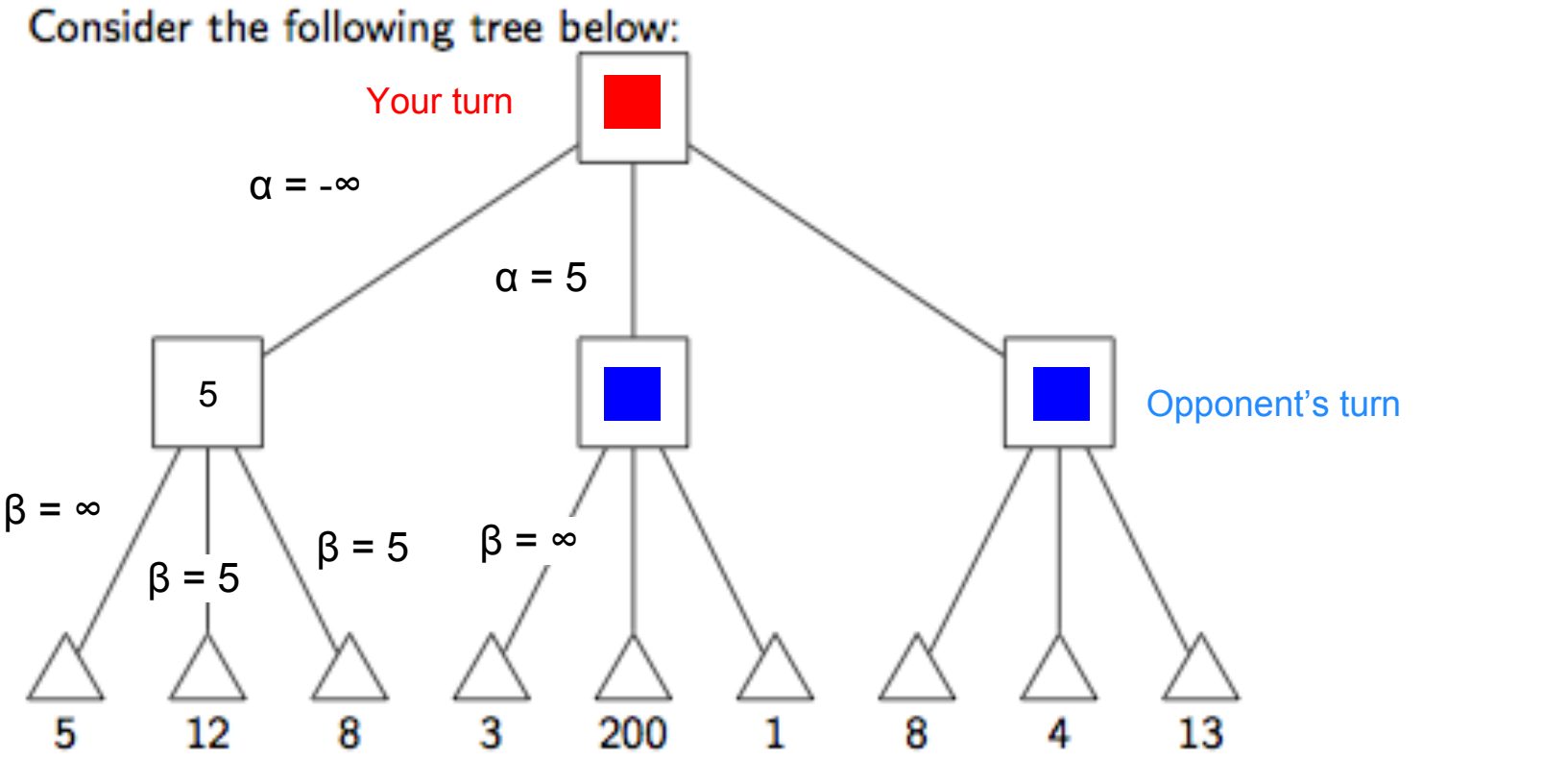
Alpha Beta Pruning



Determine which nodes will be pruned using α-β pruning and the final minimax value of the root. Assume the first player is MAX.

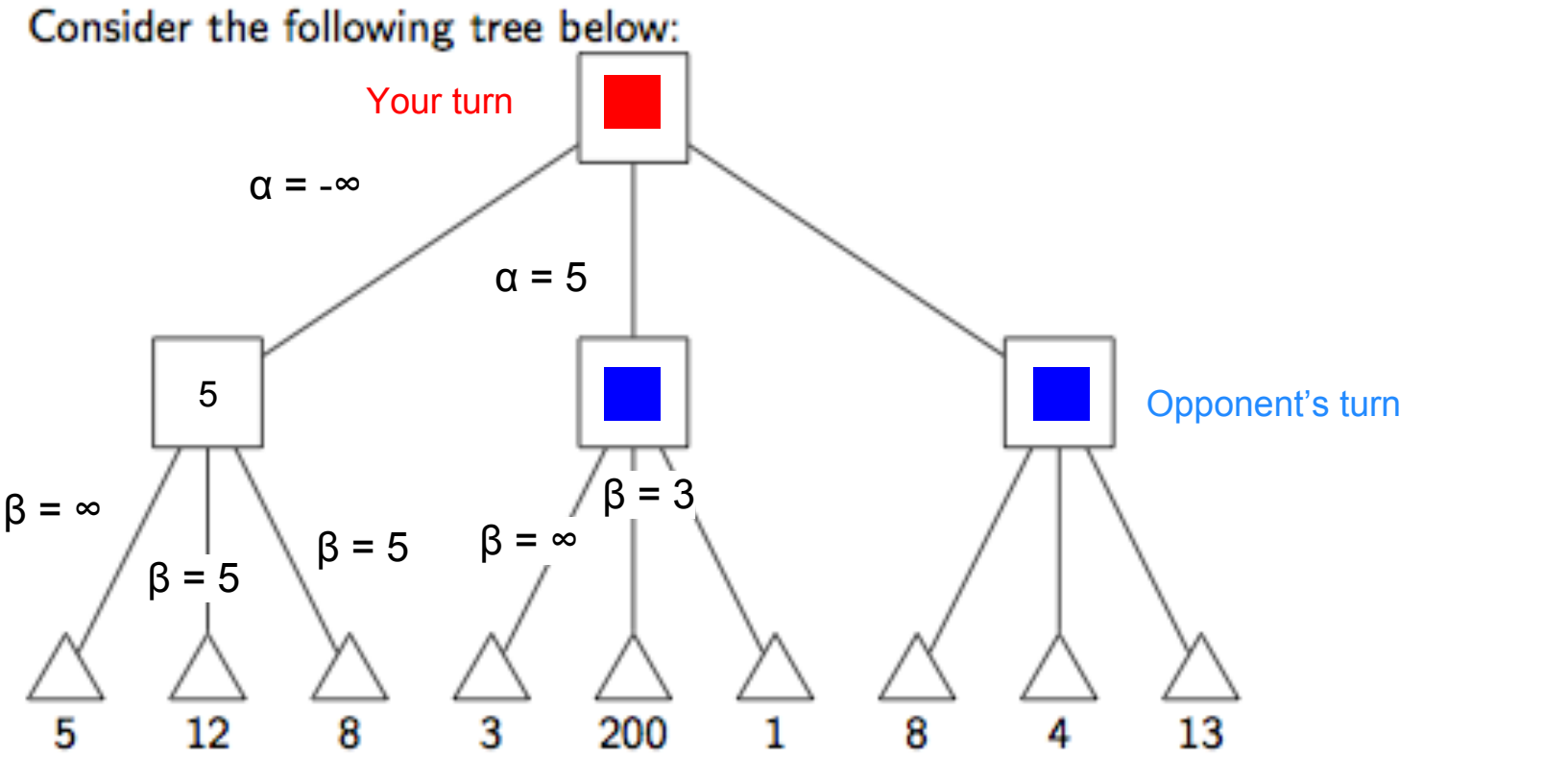
Also assume we traverse children from left to right.

Alpha Beta Pruning



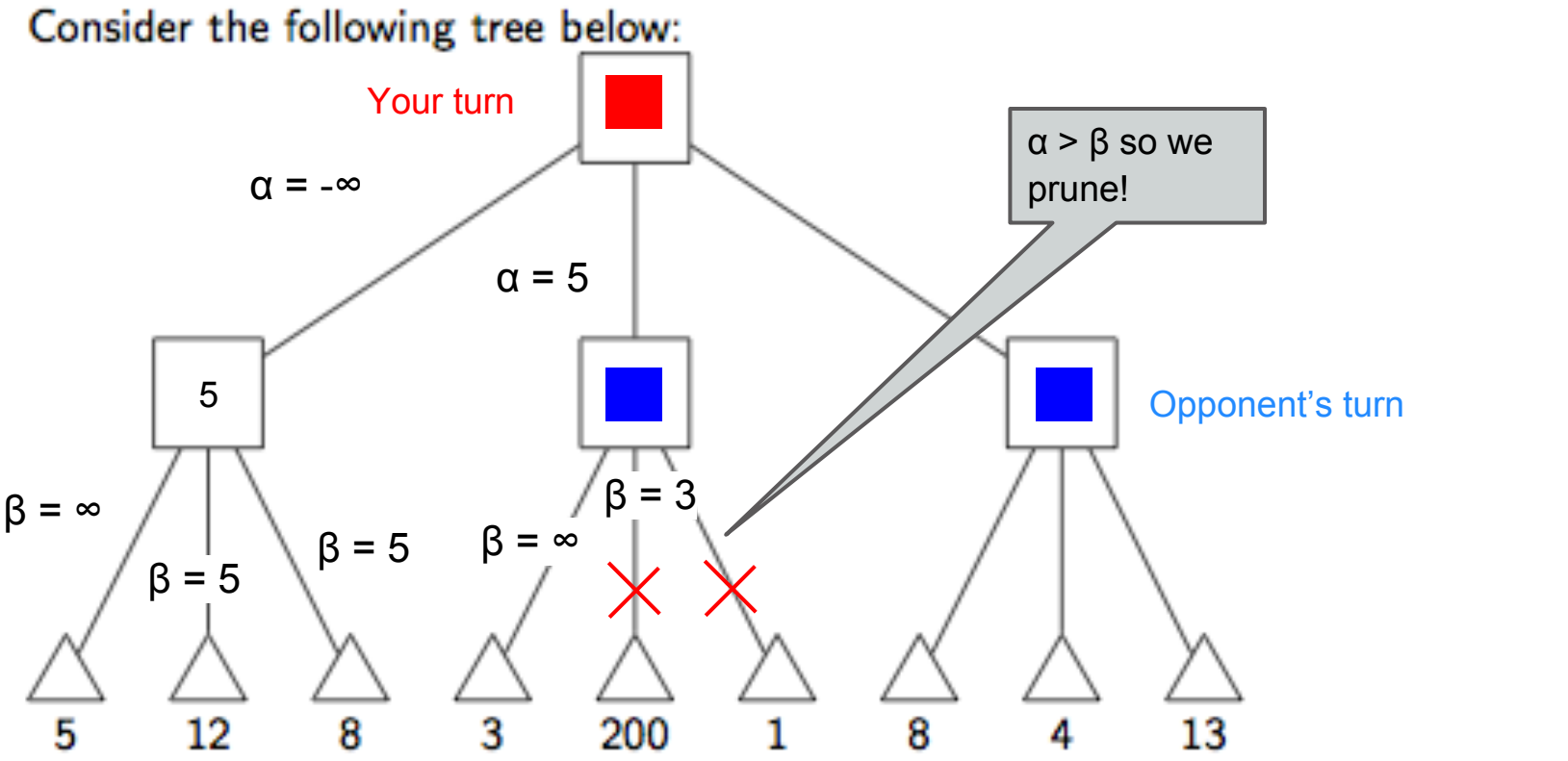
Also assume we traverse children from left to right.

Alpha Beta Pruning



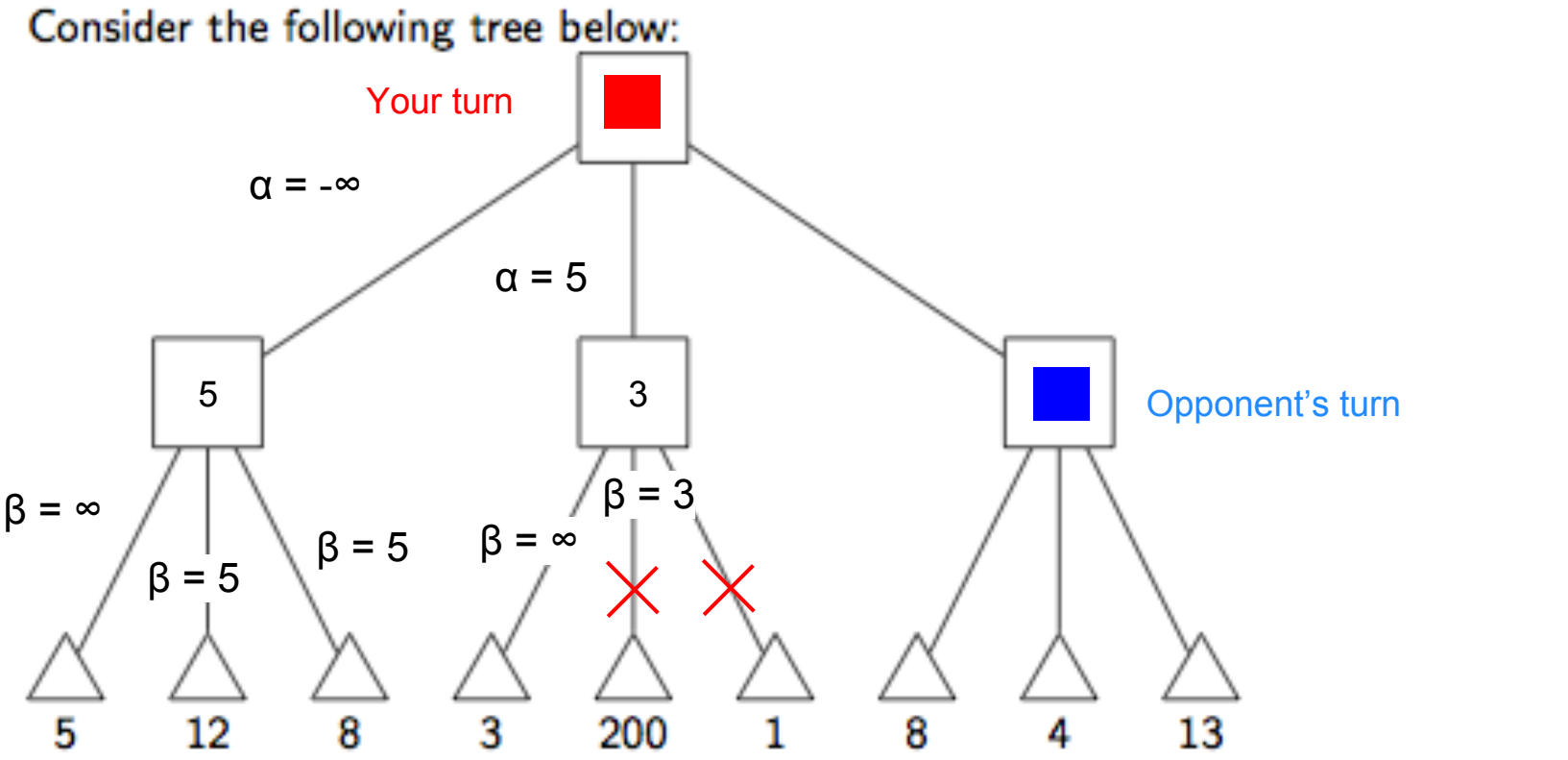
Also assume we traverse children from left to right.

Alpha Beta Pruning



Also assume we traverse children from left to right.

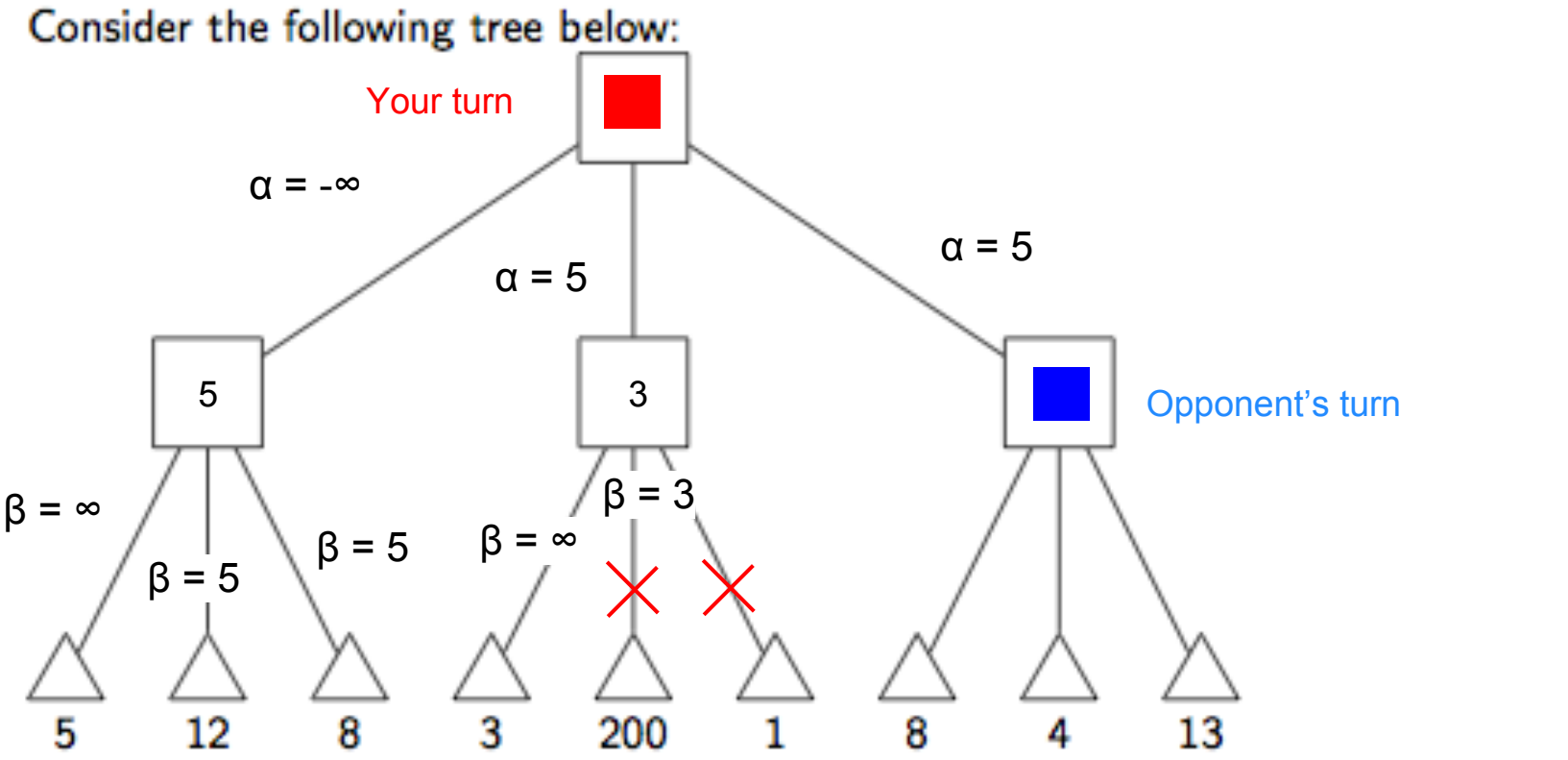
Alpha Beta Pruning



Determine which nodes will be pruned using α-β pruning and the final minimax value of the root. Assume the first player is MAX.

Also assume we traverse children from left to right.

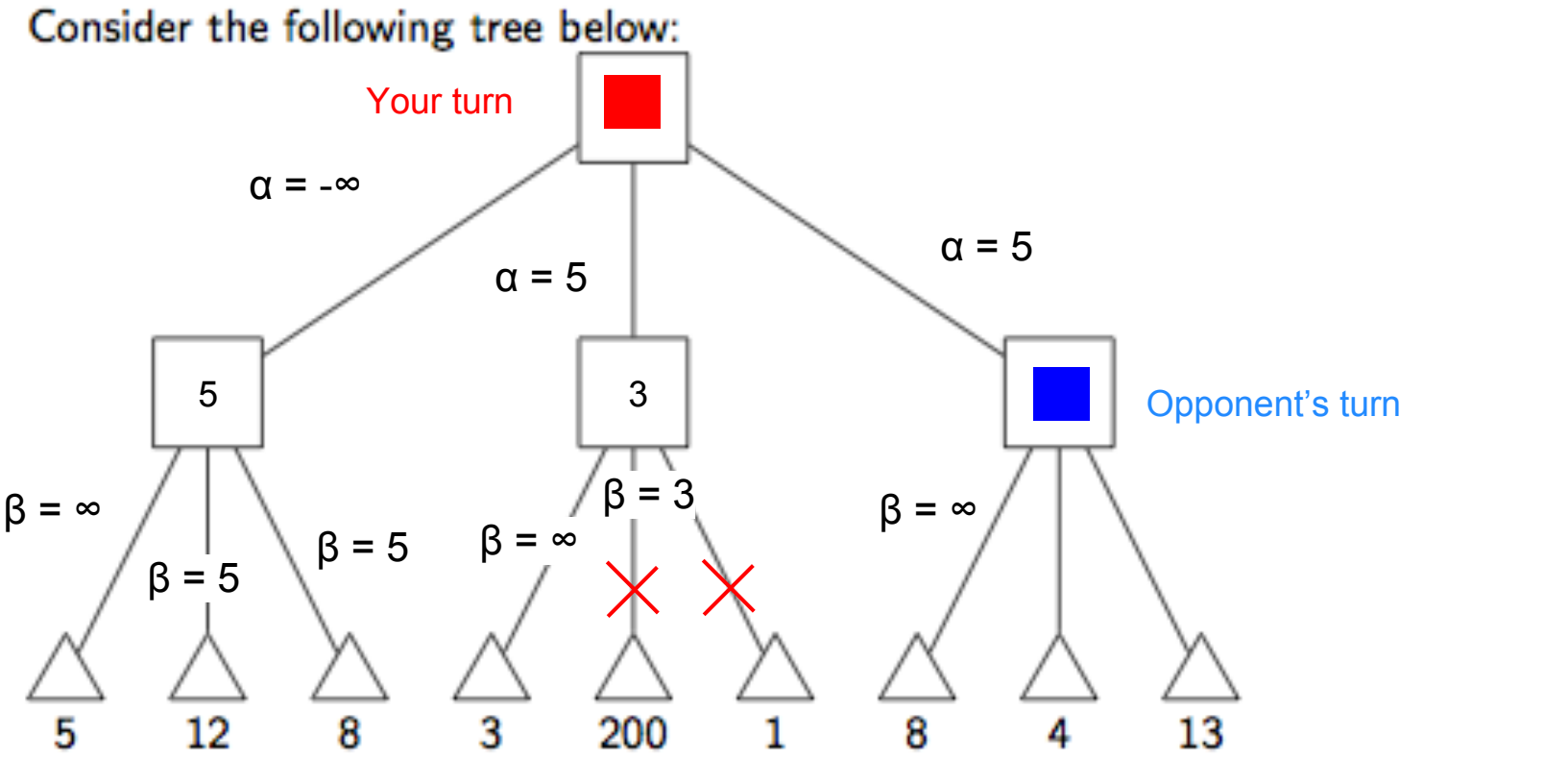
Alpha Beta Pruning



Determine which nodes will be pruned using α-β pruning and the final minimax value of the root. Assume the first player is MAX.

Also assume we traverse children from left to right.

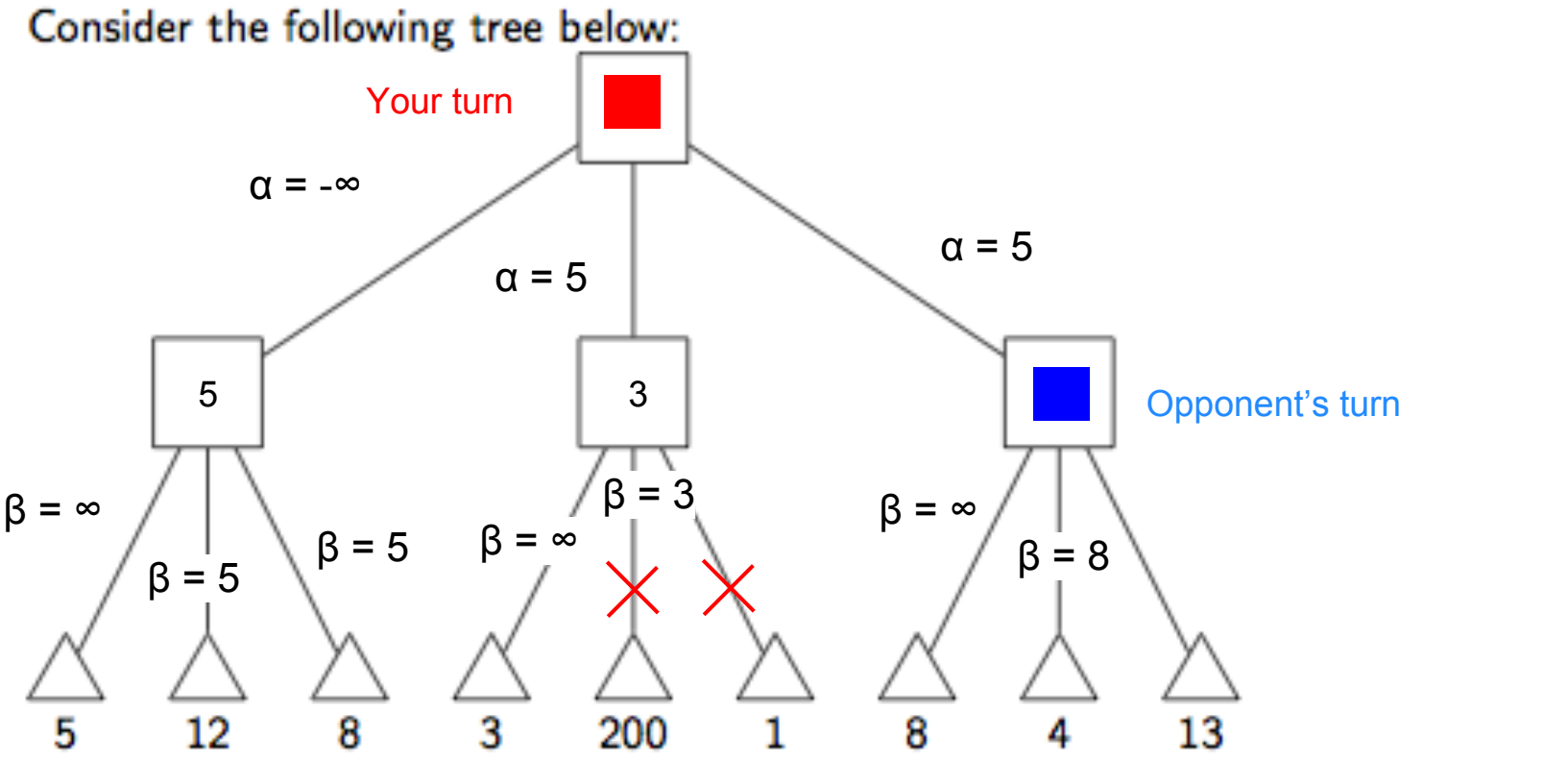
Alpha Beta Pruning



Determine which nodes will be pruned using α-β pruning and the final minimax value of the root. Assume the first player is MAX.

Also assume we traverse children from left to right.

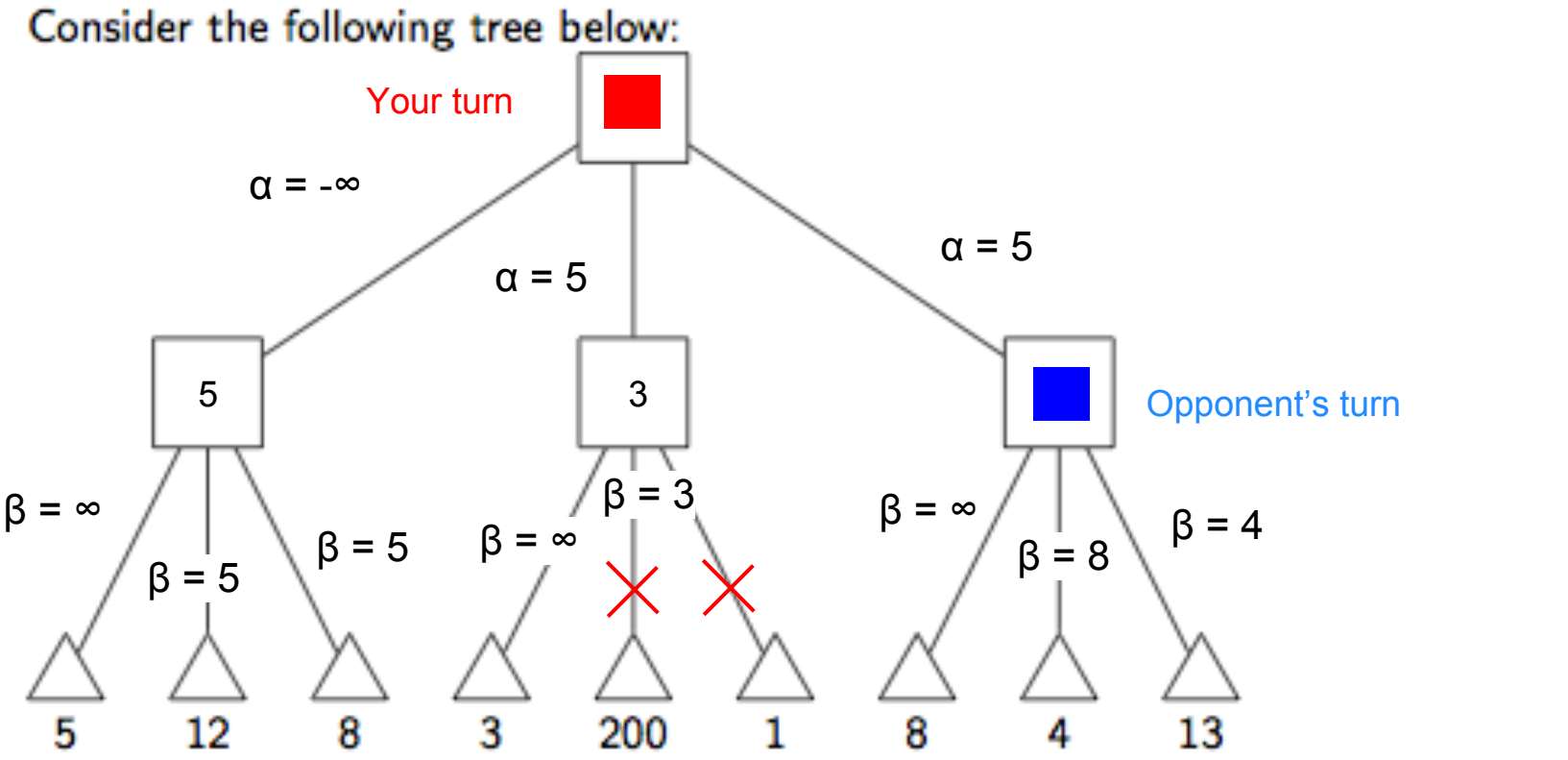
Alpha Beta Pruning



Determine which nodes will be pruned using α-β pruning and the final minimax value of the root. Assume the first player is MAX.

Also assume we traverse children from left to right.

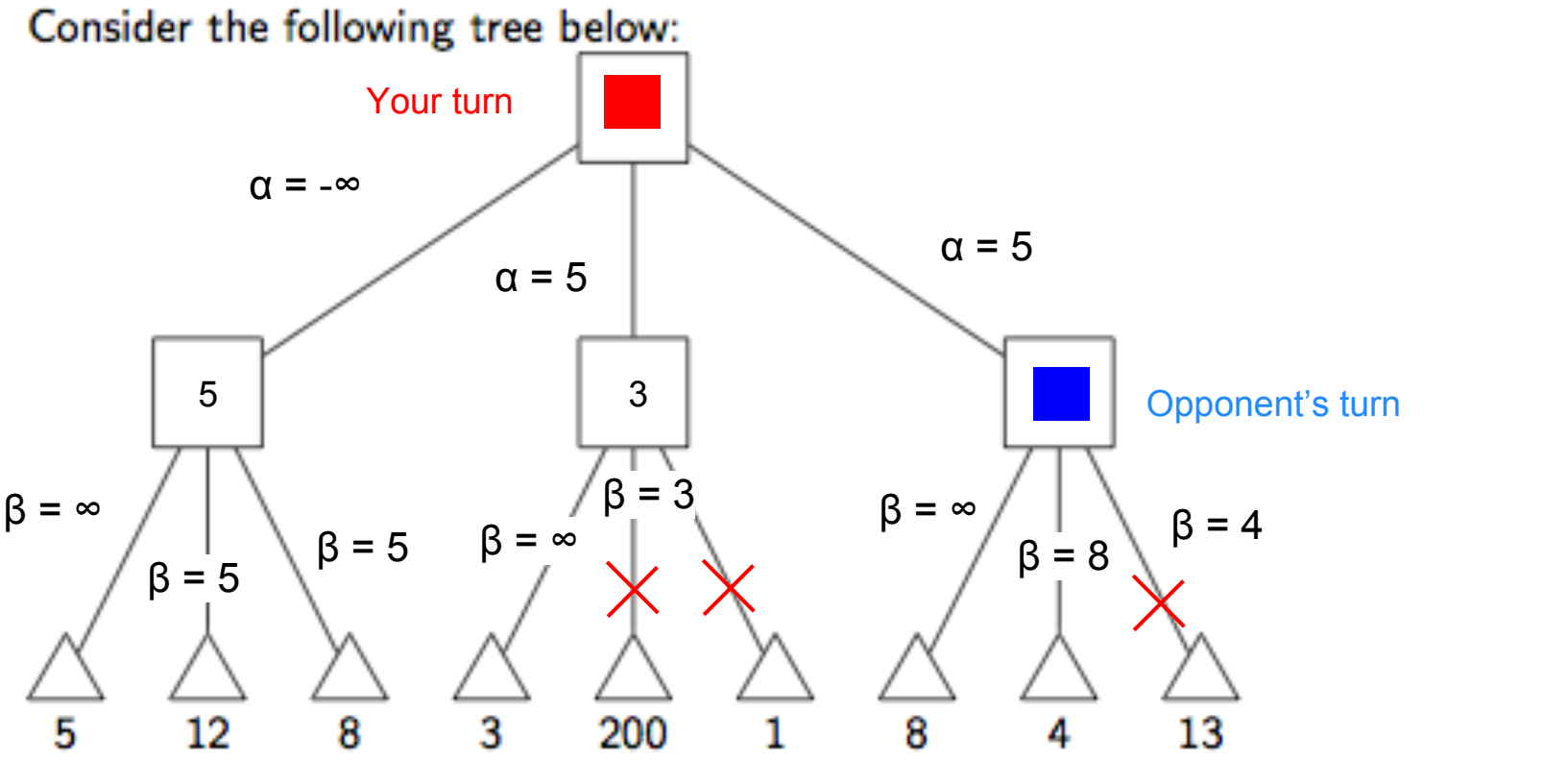
Alpha Beta Pruning



Determine which nodes will be pruned using α-β pruning and the final minimax value of the root. Assume the first player is MAX.

Also assume we traverse children from left to right.

Alpha Beta Pruning

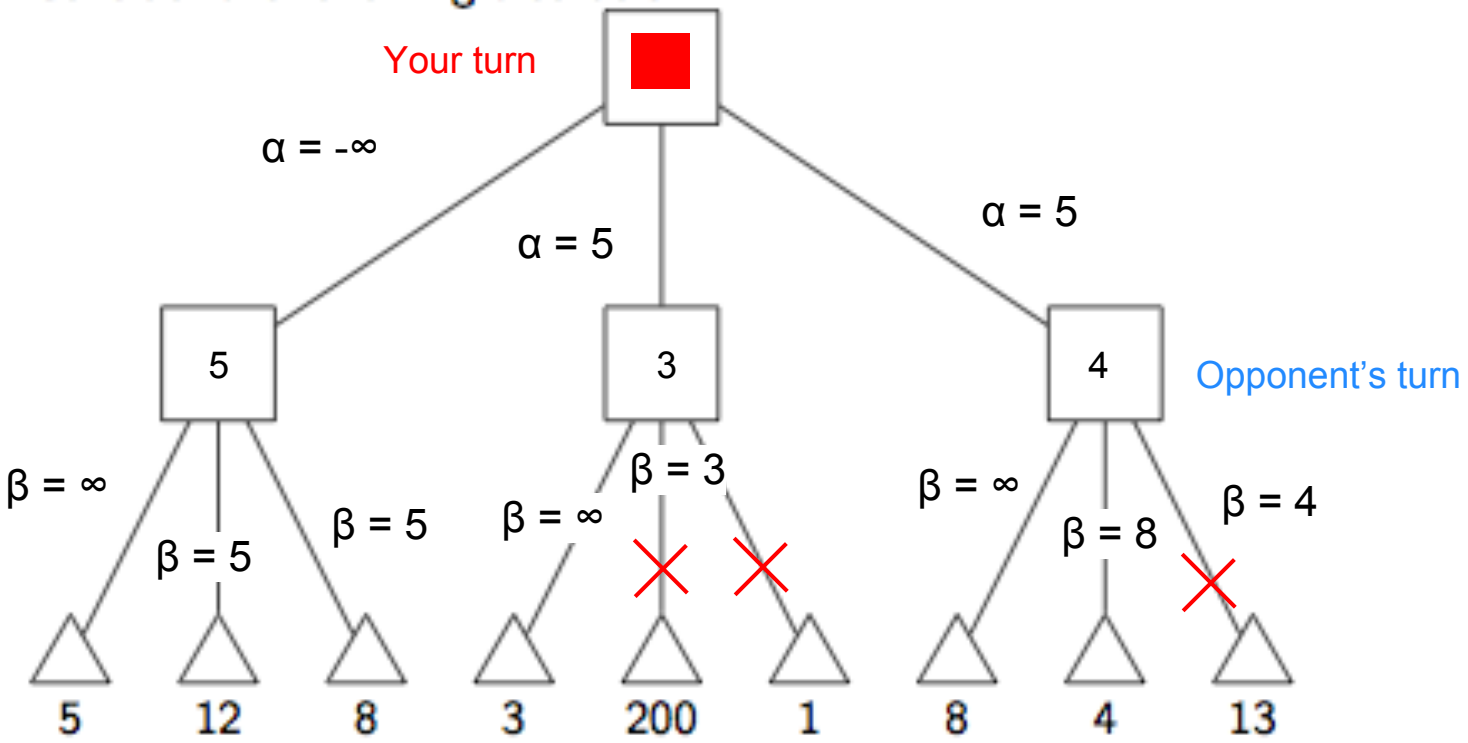


Determine which nodes will be pruned using α-β pruning and the final minimax value of the root. Assume the first player is MAX.

Also assume we traverse children from left to right.

Alpha Beta Pruning

Consider the following tree below:

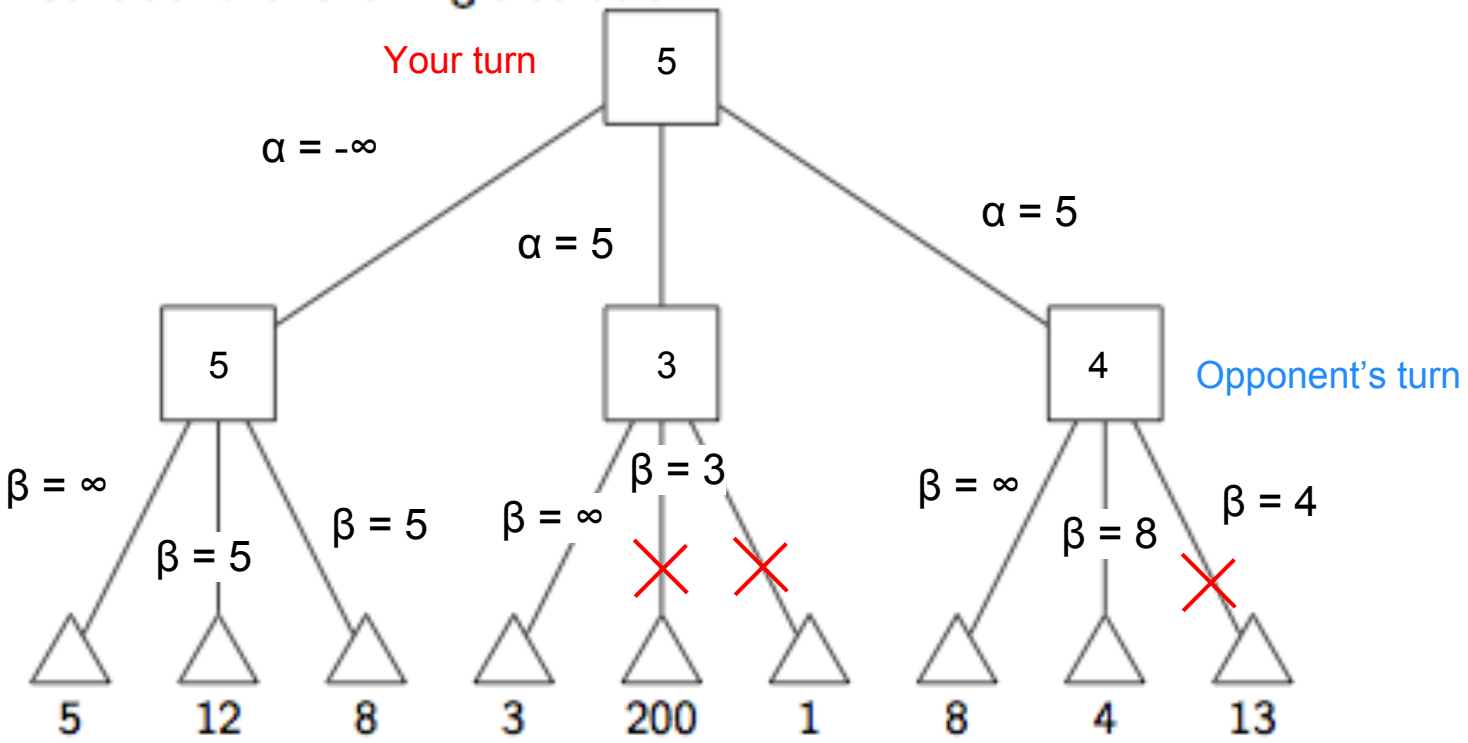


Determine which nodes will be pruned using α-β pruning and the final minimax value of the root. Assume the first player is MAX.

Also assume we traverse children from left to right.

Alpha Beta Pruning

Consider the following tree below:

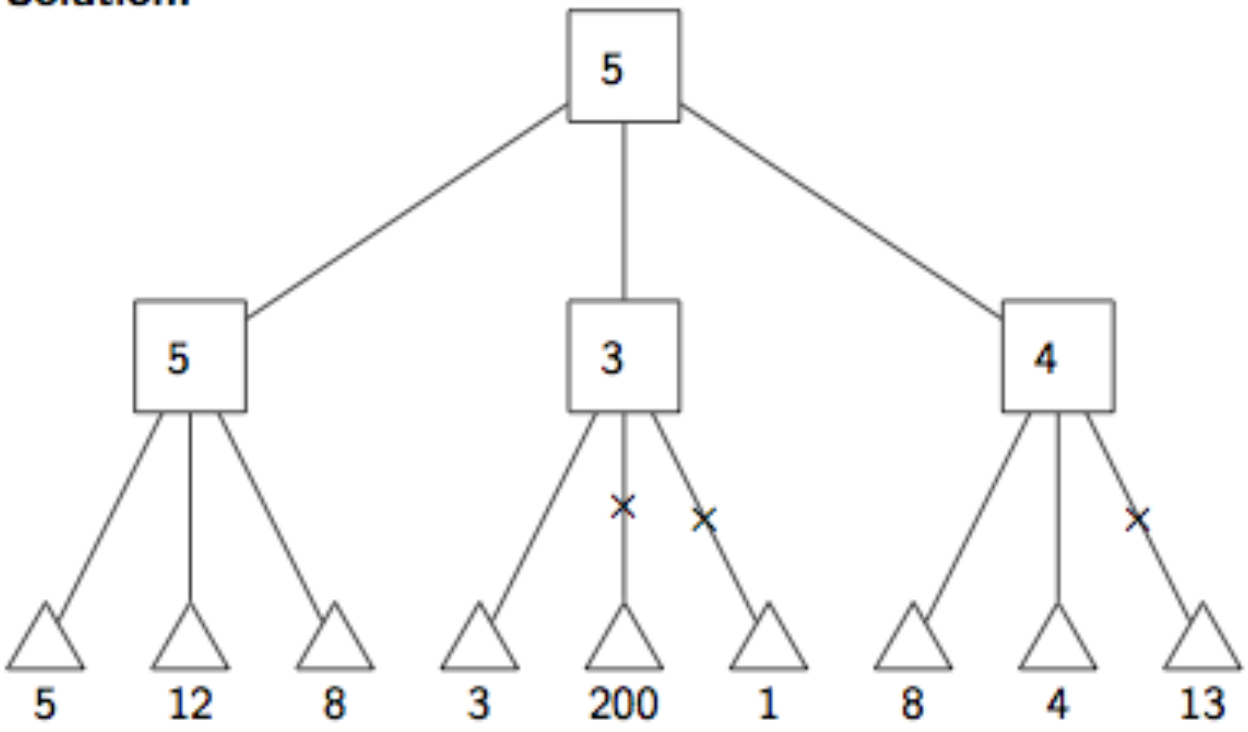


Determine which nodes will be pruned using α-β pruning and the final minimax value of the root. Assume the first player is MAX.

Also assume we traverse children from left to right.

Alpha Beta Pruning

Solution:



Alpha Beta Pruning

True or False:

1. After alpha-beta pruning, the root maximizer node will never have the wrong value.

2. During alpha-beta pruning, none of the minimizer or maximizer intermediate nodes will differ from values found when using the normal minimax algorithm.

3. Alpha-beta pruning can have different prunings based on the order in which the algorithm traverses the tree.

Alpha Beta Pruning

True or False:

1. After alpha-beta pruning, the root maximizer node will never have the wrong value.

True -- try doing minimax on our example

2. During alpha-beta pruning, none of the minimizer or maximizer intermediate nodes will differ from values found when using the normal minimax algorithm.

3. Alpha-beta pruning can have different prunings based on the order in which the algorithm traverses the tree.

Alpha Beta Pruning

True or False:

1. After alpha-beta pruning, the root maximizer node will never have the wrong value.

True -- try doing minimax on our example

2. During alpha-beta pruning, none of the minimizer or maximizer intermediate nodes will differ from values found when using the normal minimax algorithm.

False -- try doing minimax on our example

3. Alpha-beta pruning can have different prunings based on the order in which the algorithm traverses the tree.

Alpha Beta Pruning

True or False:

1. After alpha-beta pruning, the root maximizer node will never have the wrong value.

True -- try doing minimax on our example

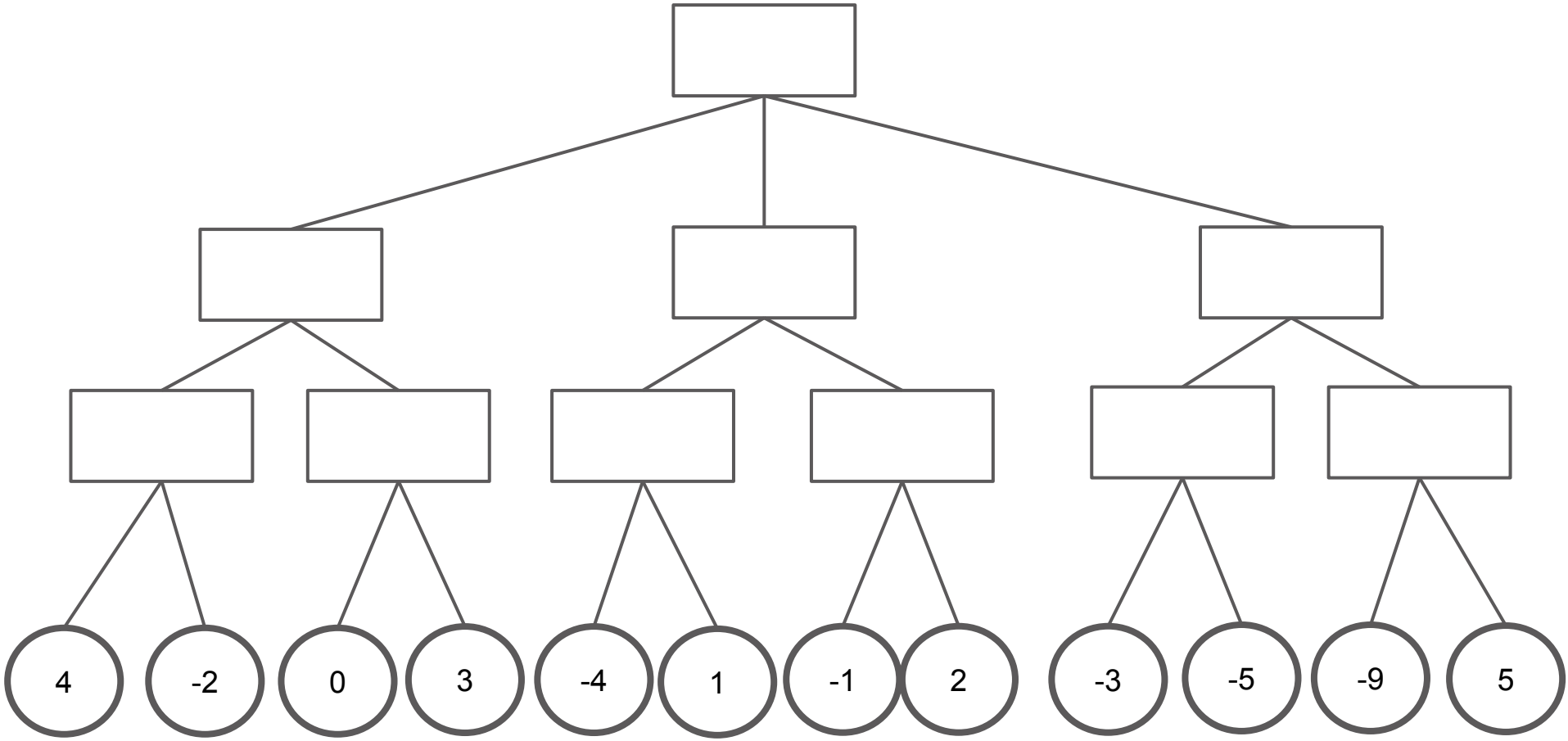
2. During alpha-beta pruning, none of the minimizer or maximizer intermediate nodes will differ from values found when using the normal minimax algorithm.

False -- try doing minimax on our example

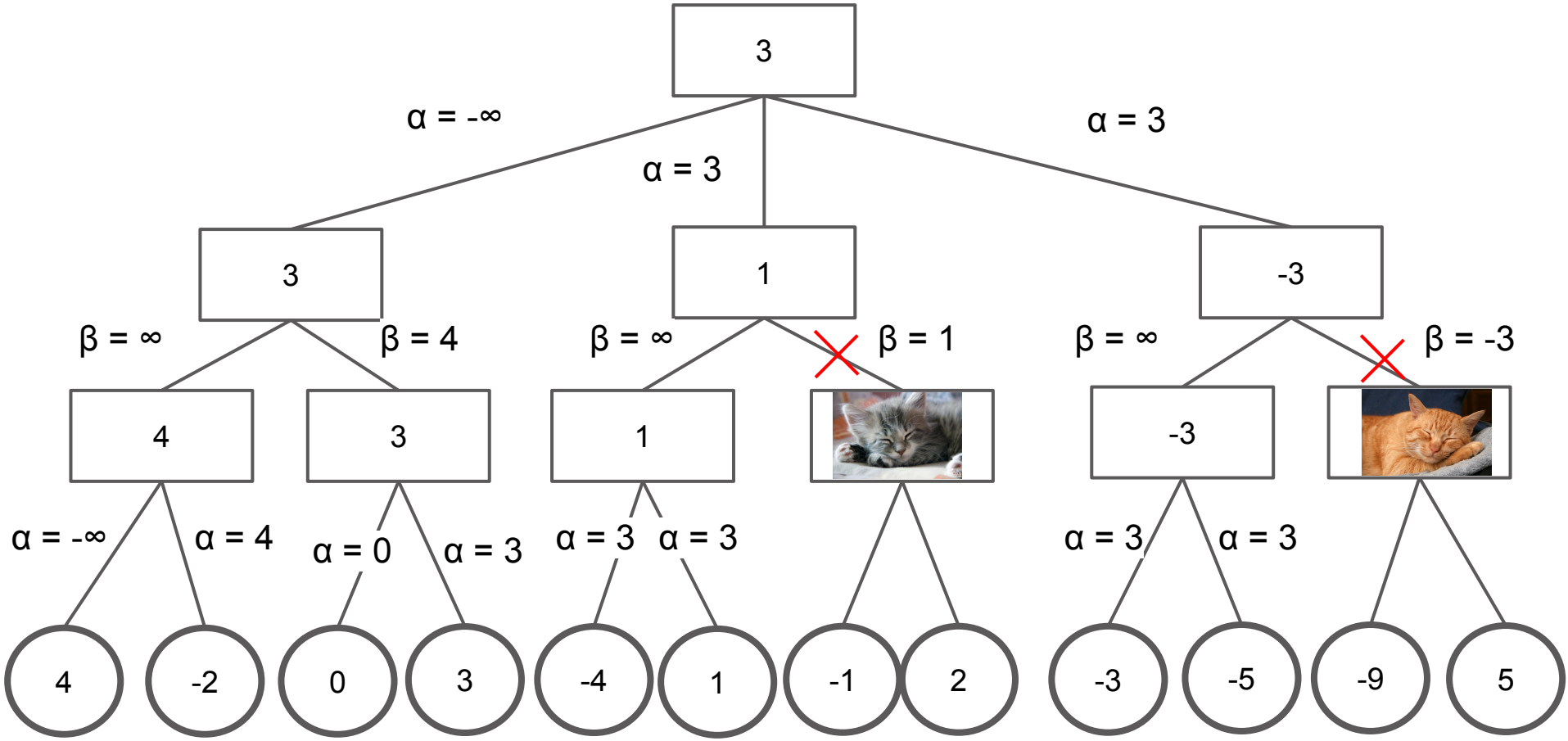
3. Alpha-beta pruning can have different prunings based on the order in which the algorithm traverses the tree.

True -- what would have happened if we started with the middle minimizer's children first?

Alpha Beta Pruning



Alpha Beta Pruning



Alpha Beta Pruning Code

```
public Best chooseMove( boolean side, int
alpha, int beta) {
  Best myBest = new Best(); // My best move
  Best reply; // Opponent's best reply
  if (the current Grid is full or has a win) {
    return a Best with the Grid's score, no
move;
  }
  if (side == COMPUTER) {
    myBest.score = alpha;
  } else {
    myBest.score = beta;
  }
  //code continued on the next page
```

```
for (each legal move m) {
    perform move m; // Modifies "this" Grid
    reply = chooseMove( ! side, alpha, beta);
    undo move m; // Restores "this" Grid
    if ((side == COMPUTER) && (reply.score >=
myBest.score)) {
      myBest.move = m;
      myBest.score = reply.score;
      alpha = reply.score;
    } else if ((side == HUMAN) && (reply.score
<= myBest.score)) {
      myBest.move = m;
      myBest.score = reply.score;
      beta = reply.score;
    }
    if (alpha >= beta) { return myBest; }
  }
  return myBest;
}
```