COMP3702 - Tutorial 12, 2021

Q1: alpha-beta pruning

Each node has 3 attributes: V, alpha and beta.

V: current best value (maximum for MAX node and minimum for MIN node) over all children node.

Alpha: the maximum lower bound of possible solutions

Beta: the minimum upper bound of possible solutions.

The initial value for root (MAX node):

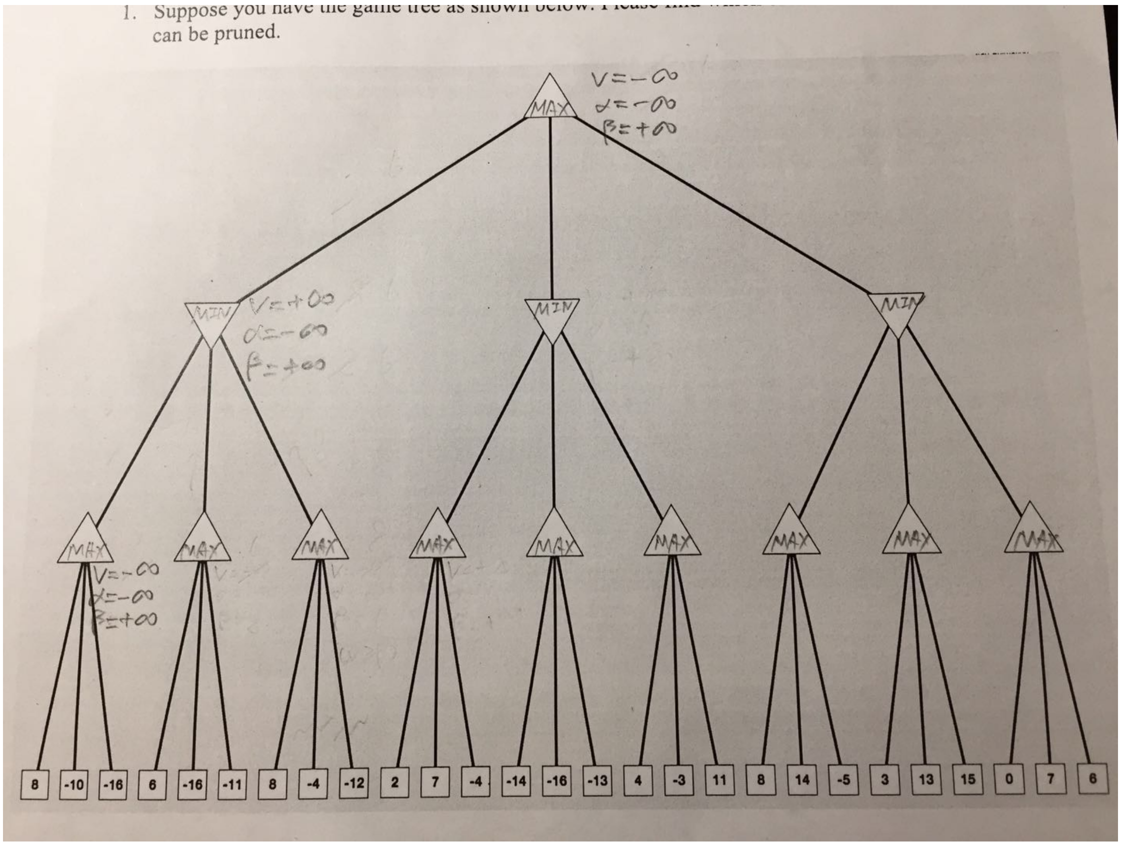
- V = -inf (the worst value)
- Alpha = -inf (the worst value)
- Beta = +inf (the worst value)
- Since V=-inf < beta=+inf, we don’ t prune the tree

The initial value for MIN node:

- V = +inf (the worst value)
- Alpha = copy it from its parent (MAX node)
- Beta = copy it from its parent (MAX node)
- Since V=+inf > alpha=-inf, we don’ t prune the tree

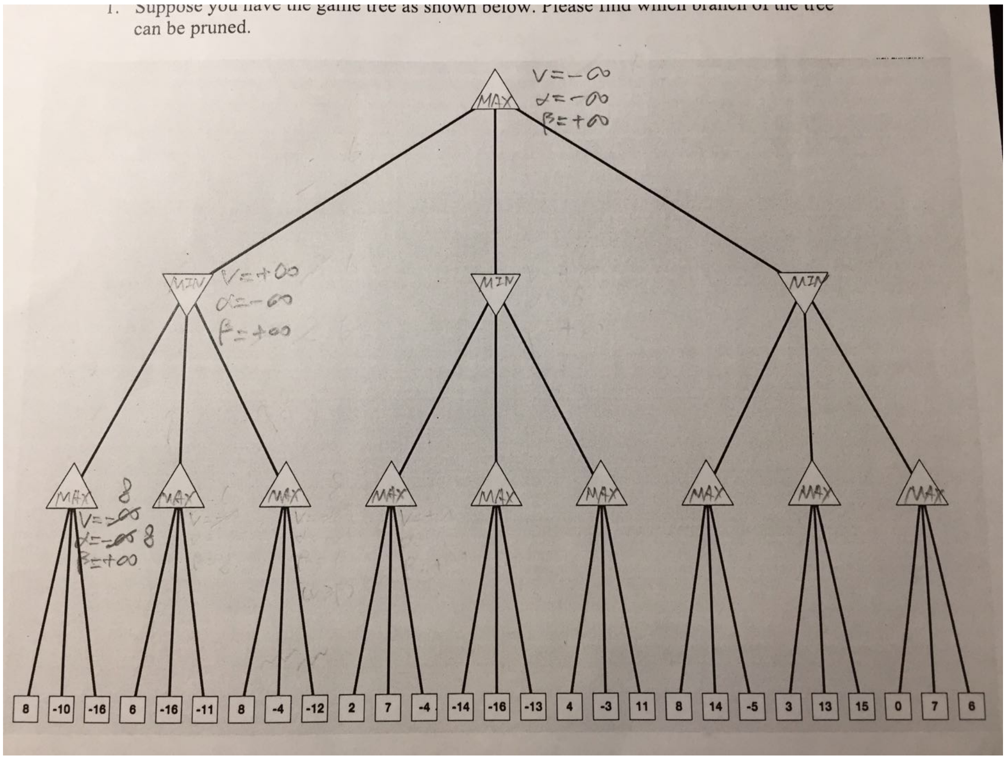
The initial value for other MAX node:

- V = -inf (the worst value)
- Alpha = copy it from its parent (MIN node)
- Beta = copy it from its parent (MIN node)
- Since V=-inf < beta=+inf, we don’ t prune the tree



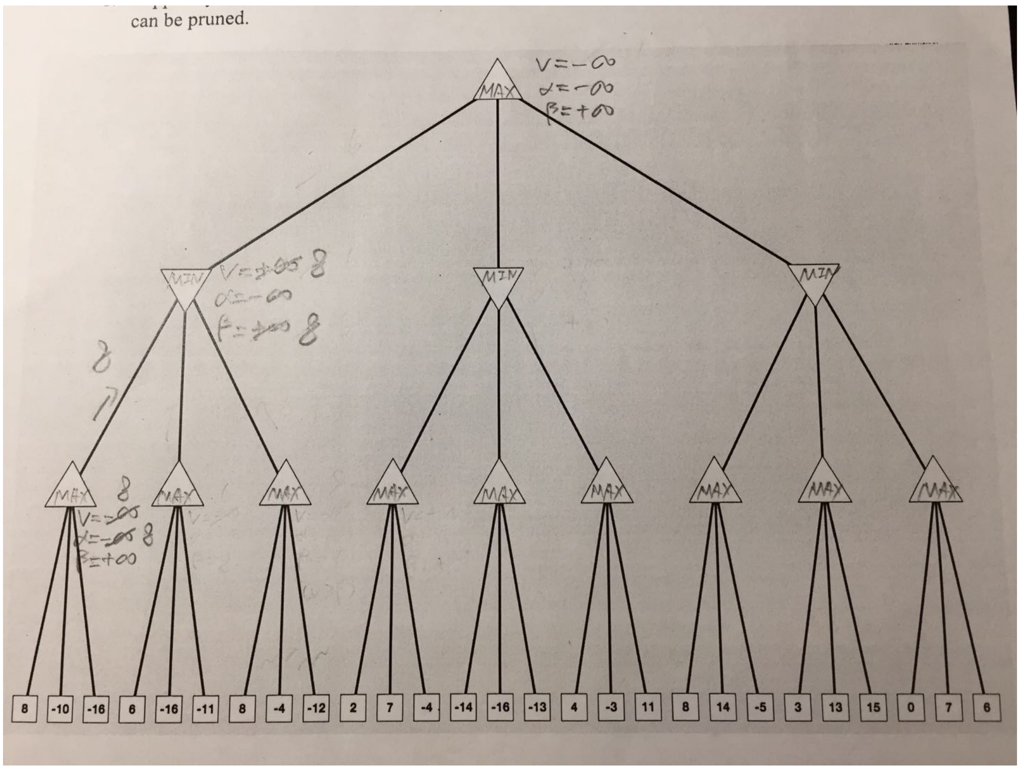
For lower MAX node,

- Update V if a better value is found, that is 8.
- Update alpha value as the maximum V found so far, that is 8.
- If V >= beta, prune the tree; otherwise, check another child node and repeat. Back to current situation, V=8 is less than beta=+inf, which means we don’ t prune the tree.

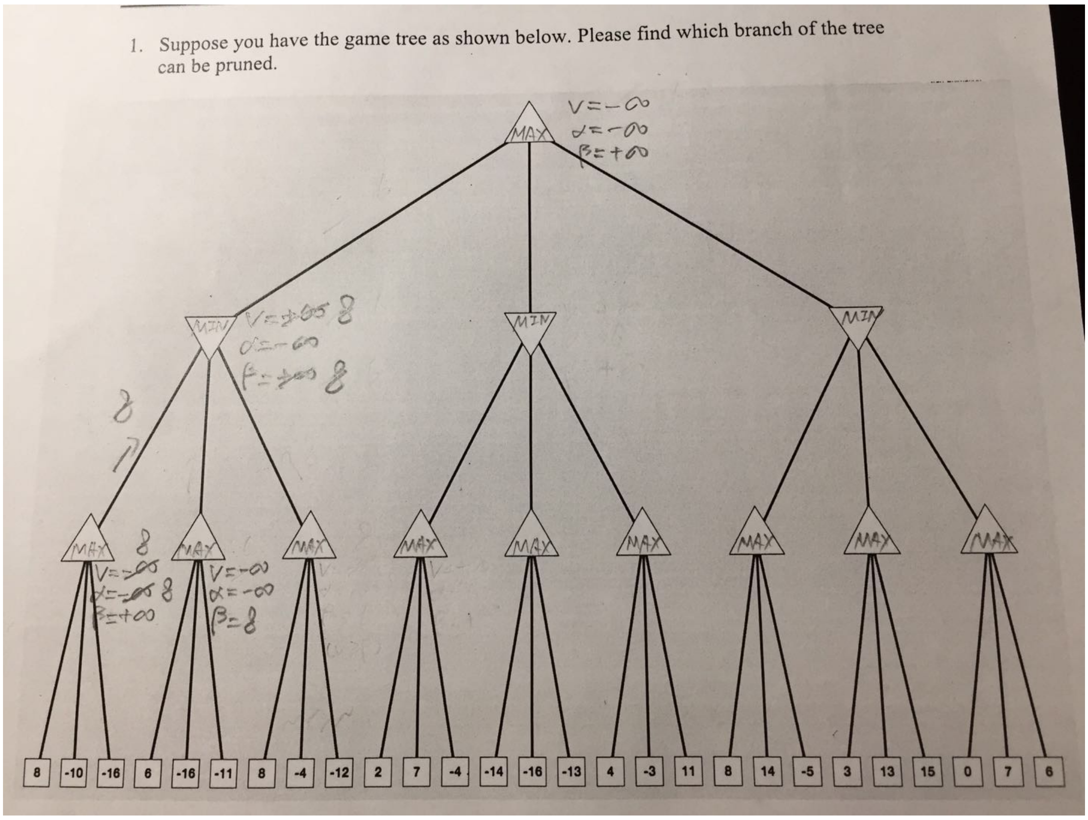


After finishing checking all other children of the lower MAX node,

- Backup the best V=8 found so far.
- For its parent node (MIN node), update V and beta. The beta currently is 8, which means there is a candidate solution for this MIN node costs 8. In other words, the current lowest upper bound value is 8.

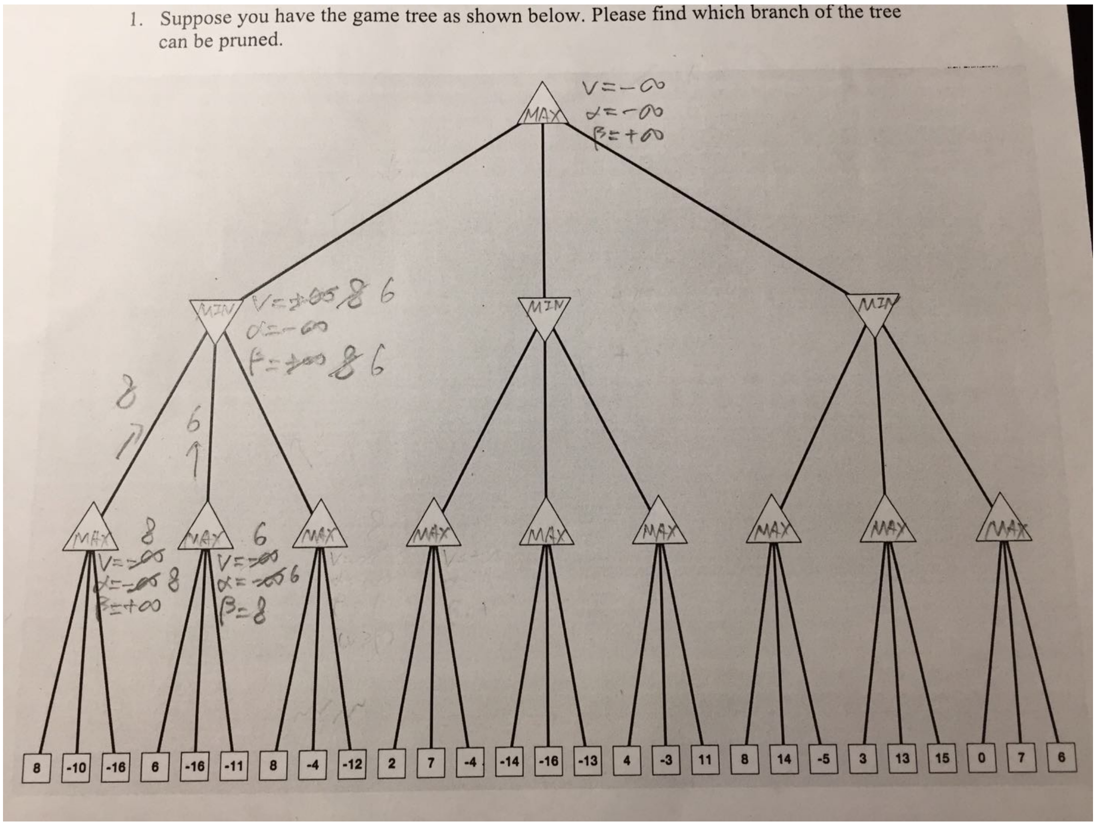


For the new MAX node from current MIN node, the initial beta is set to 8. Beta here will be used as a measure to prune the tree if no solution is likely to be found.



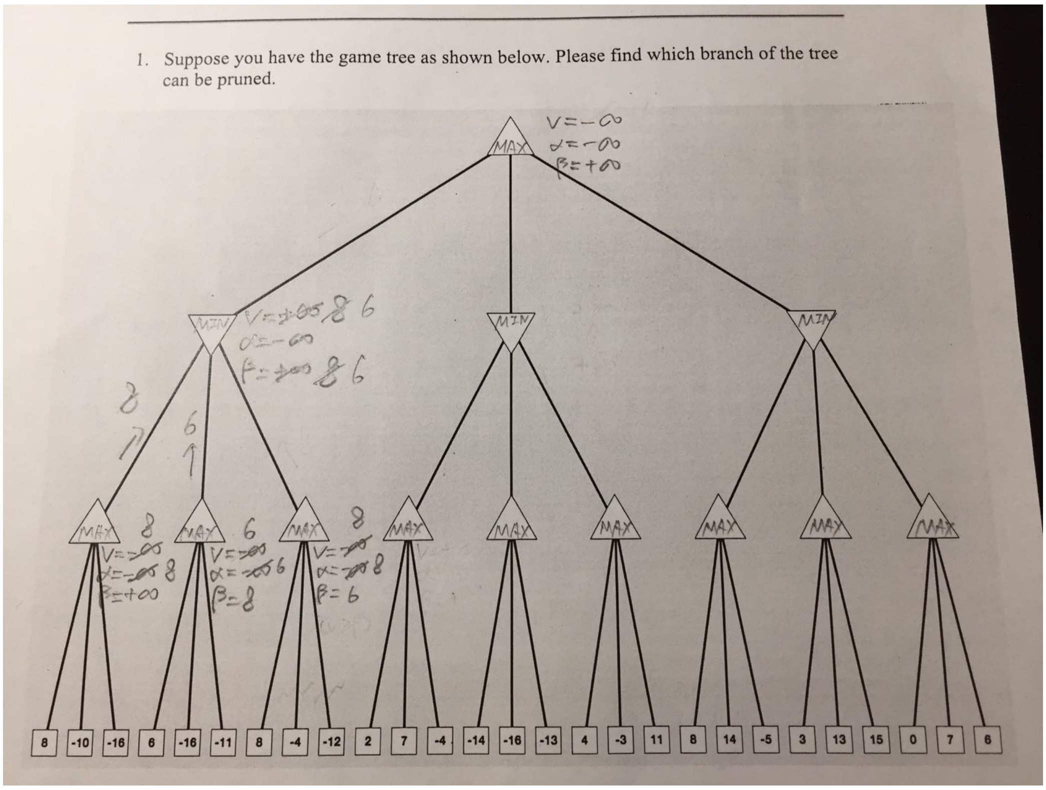
For MAX node,

- Update V if a better value is found, that is 6.
- Update alpha value as the maximum V found so far, that is 6.
- Check if V >= beta, prune the tree; otherwise, check another child node
- After finishing checking, backup the V to its parent. For the parent (MIN node), we update V and beta to 6 because a better solution is found.

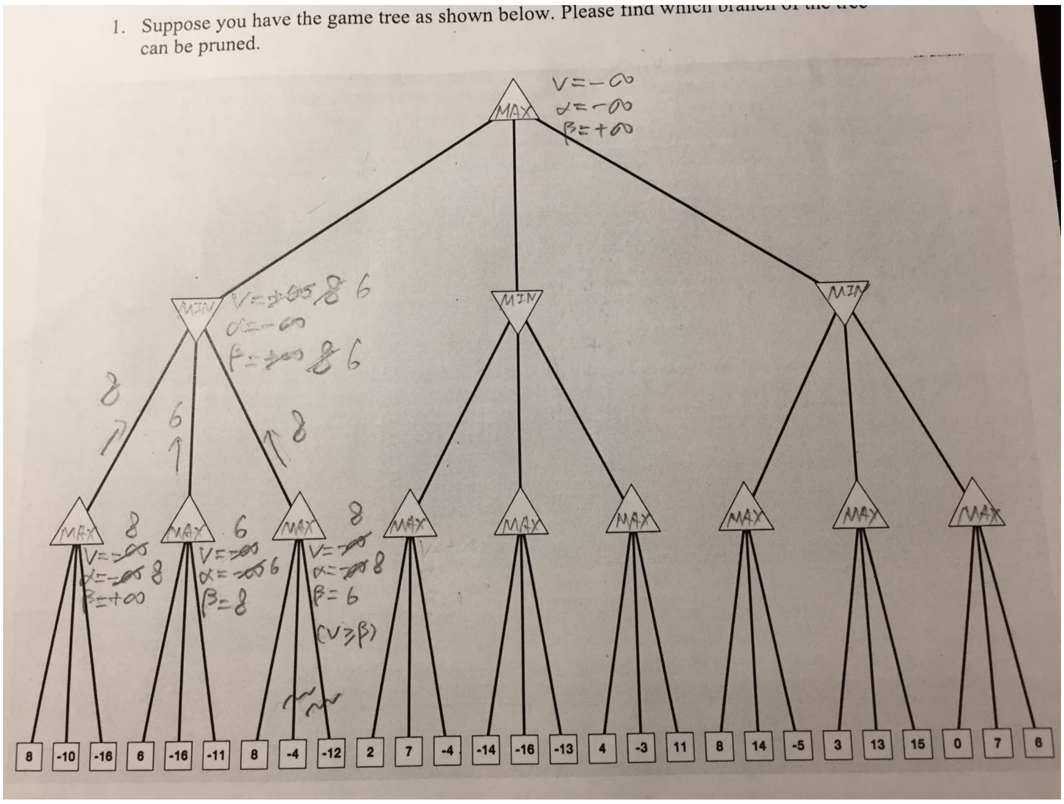


For MAX node,

- Update V if a better value is found, that is 8.
- Update alpha value as the maximum V found so far, that is 8.

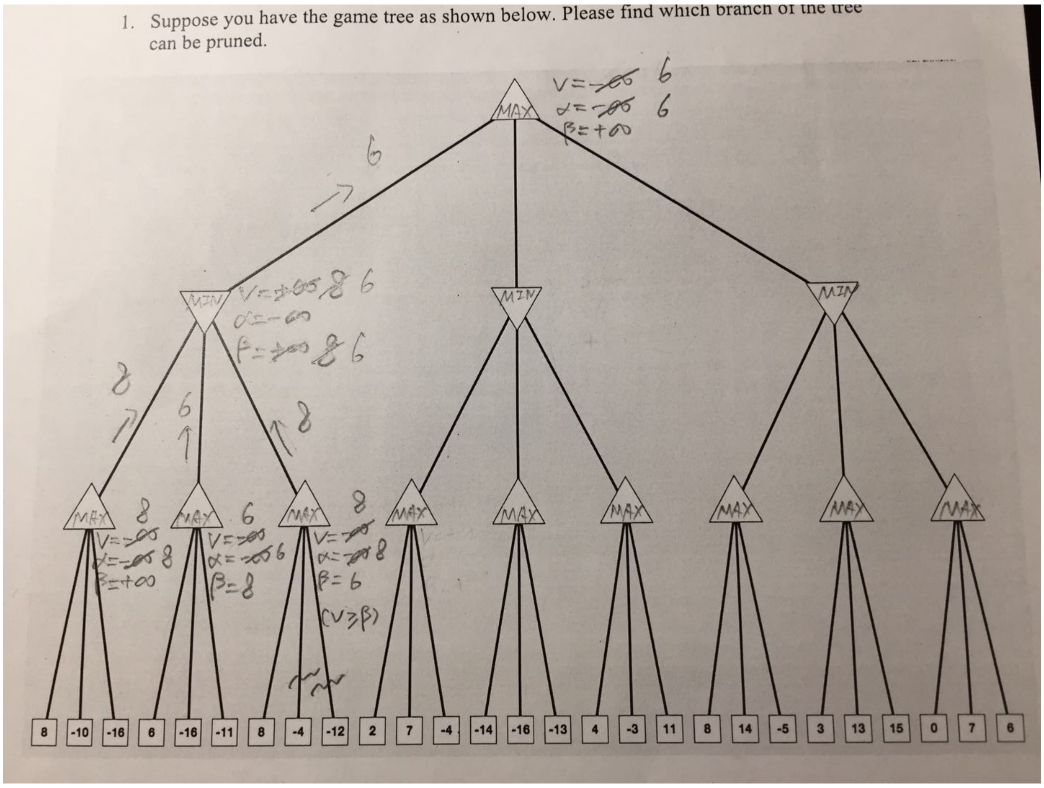


- Currently, V=8 > beta=6, we will prune the tree. Remember this is MAX node, this check condition intuitively says if an arbitrary V obtained from children cannot reduce the beta, then the maximum V of course cannot reduce the upper bound (beta).
- After finishing pruning, backup the V (=8) to its parent. Since current V don’ t improve anything, we just keep the V and beta to be the same.



The MIN node has been checked, therefore we backup V to the root node.

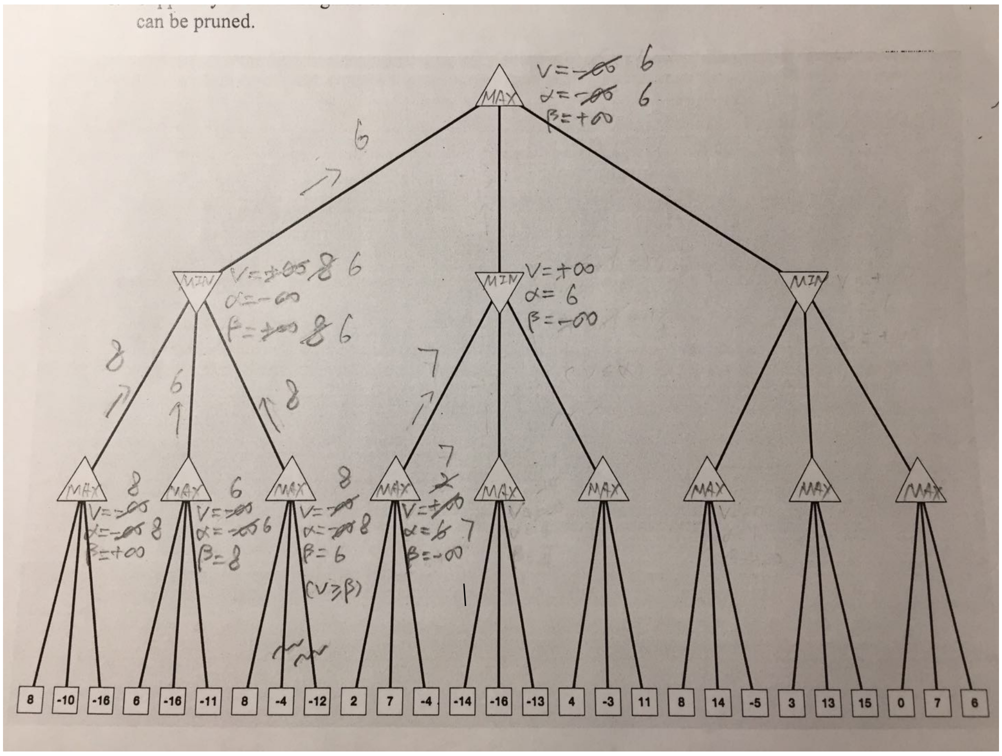
For the root, since a better alpha is found, update alpha to be 6.



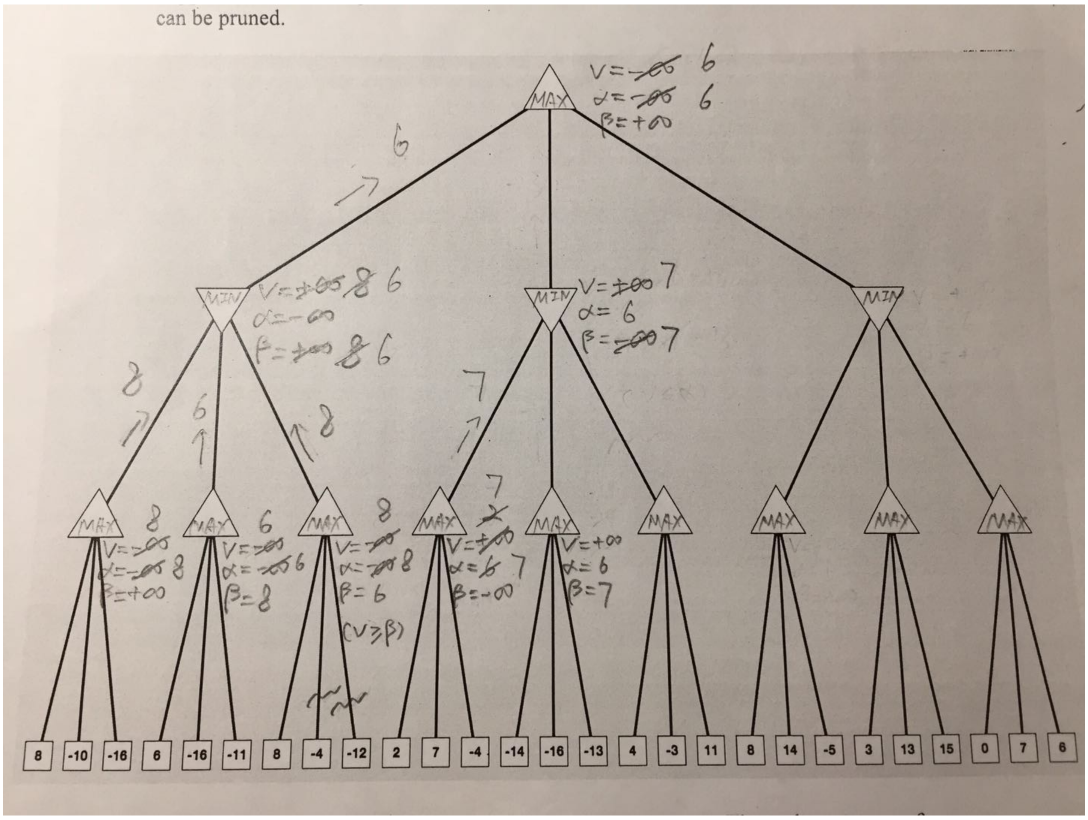
For the middle MIN node, since V=+inf > alpha=6, we don’t prune the tree.

For the lower MAX node, since V=-inf < beta=+inf, we don’t prune the tree. Then, we will

- Update V if a better value is found.
- Update alpha value as the maximum V found so far.
- Check if V >= beta, prune the tree; otherwise, check another child node
- After finishing checking, backup the V and update beta to its parent, that is 7.

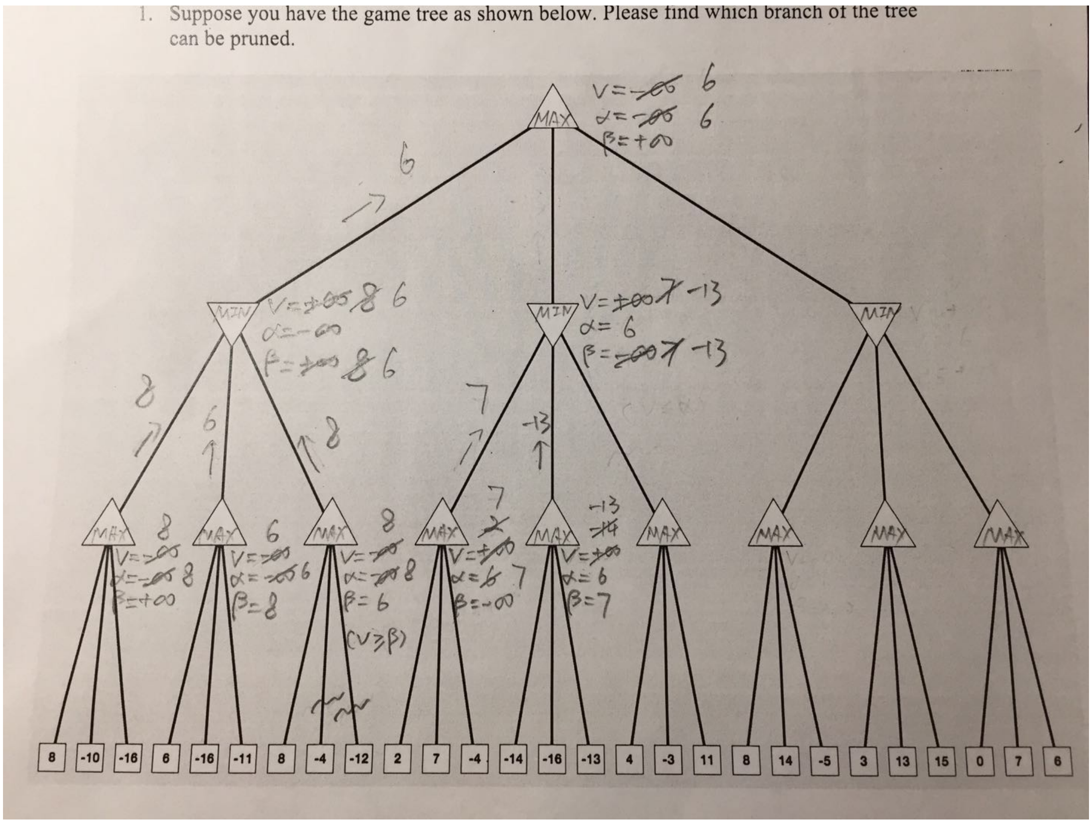


The same.

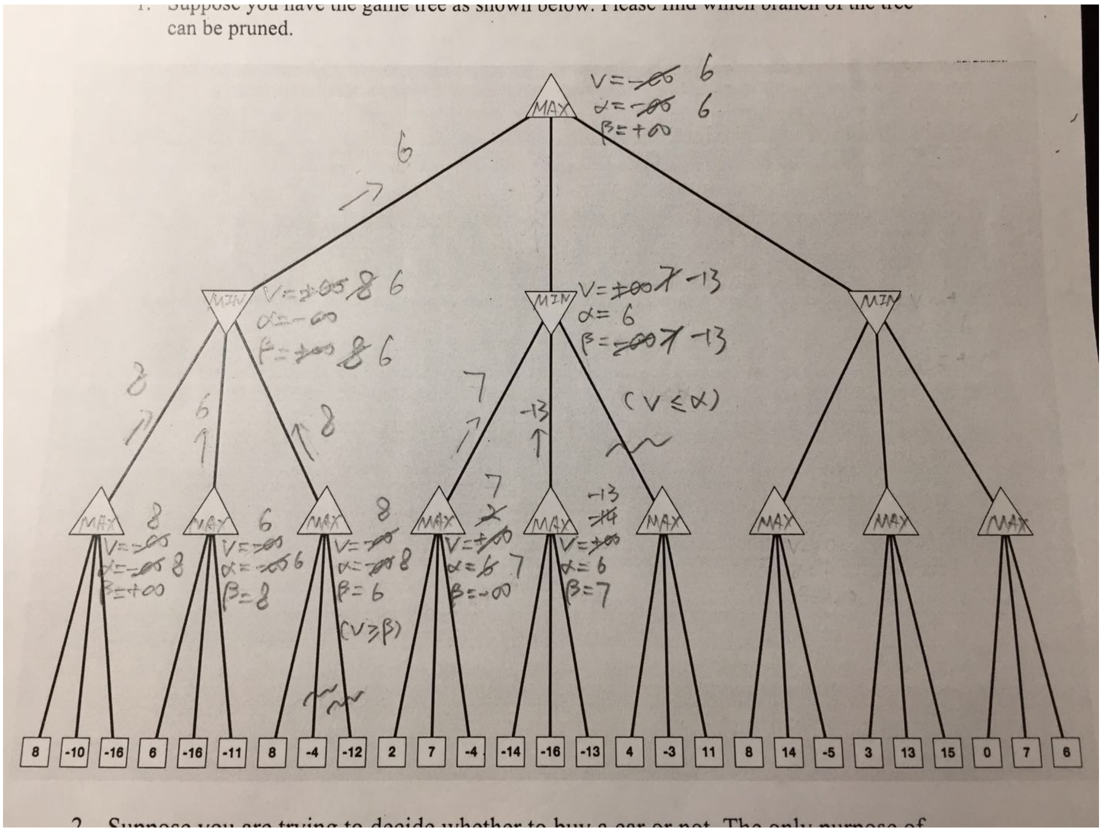


Now, for MIN node, a lower V is found, therefore its V and beta are updated to -13.

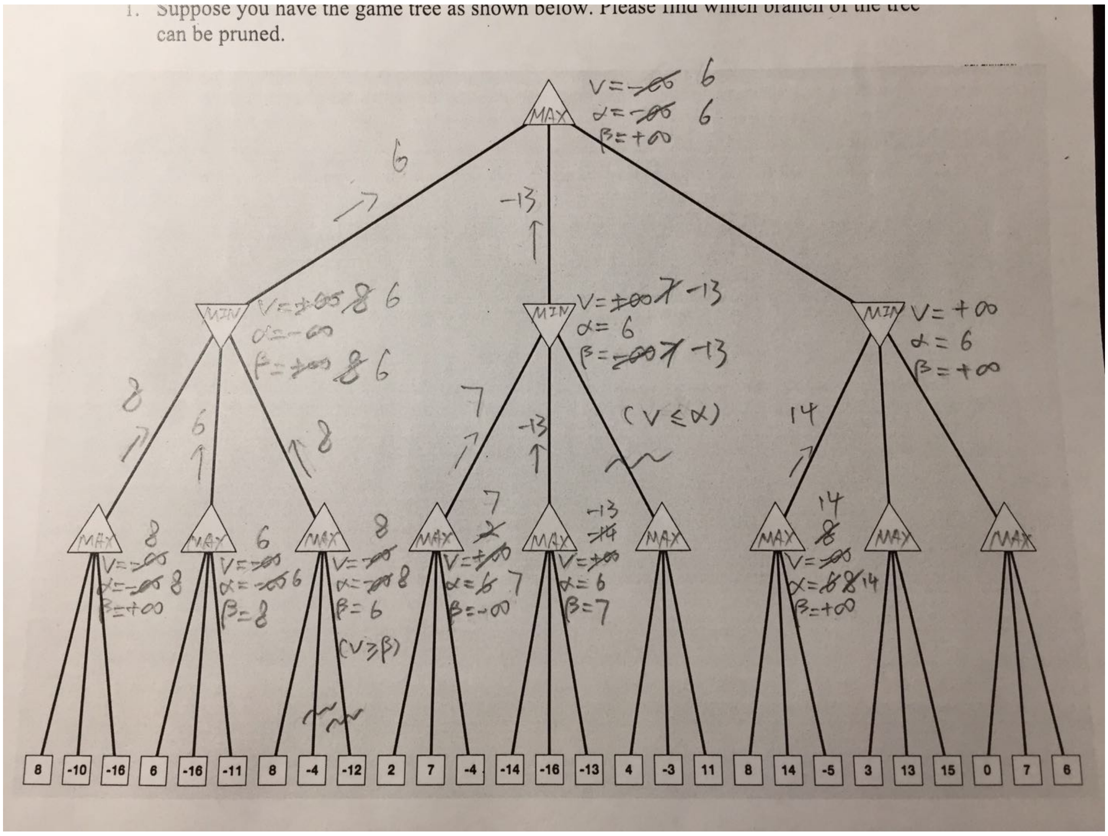
Since V=-13 < alpha=6, intuitively it means an arbitrary V obtained from children cannot increase the alpha, then the minimum V cannot increase the lower bound surely. As a result, we are safe to prune the tree.



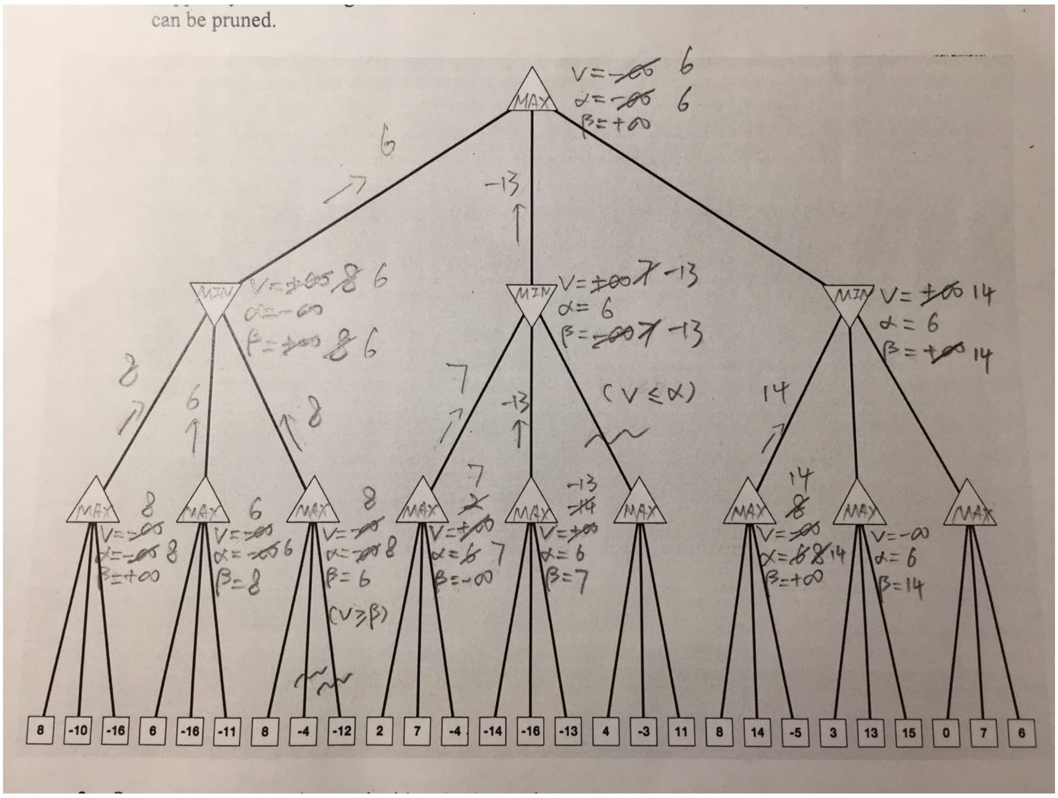
Prune the tree.



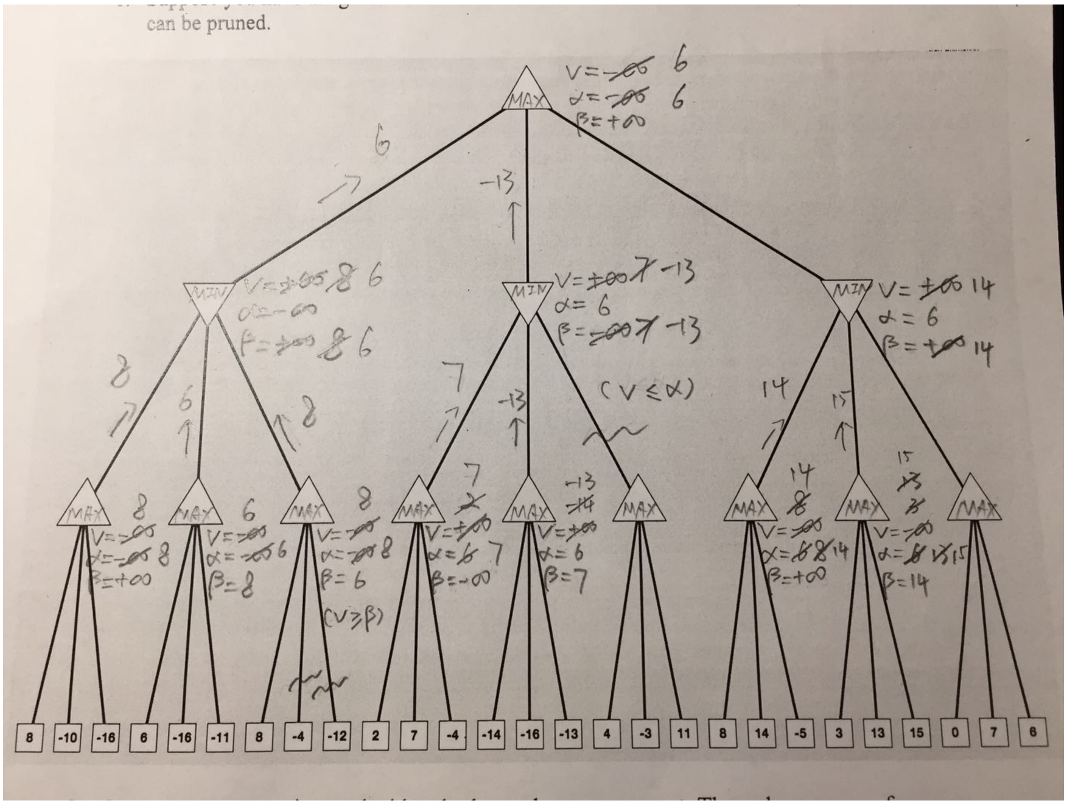
Repeat the same steps.



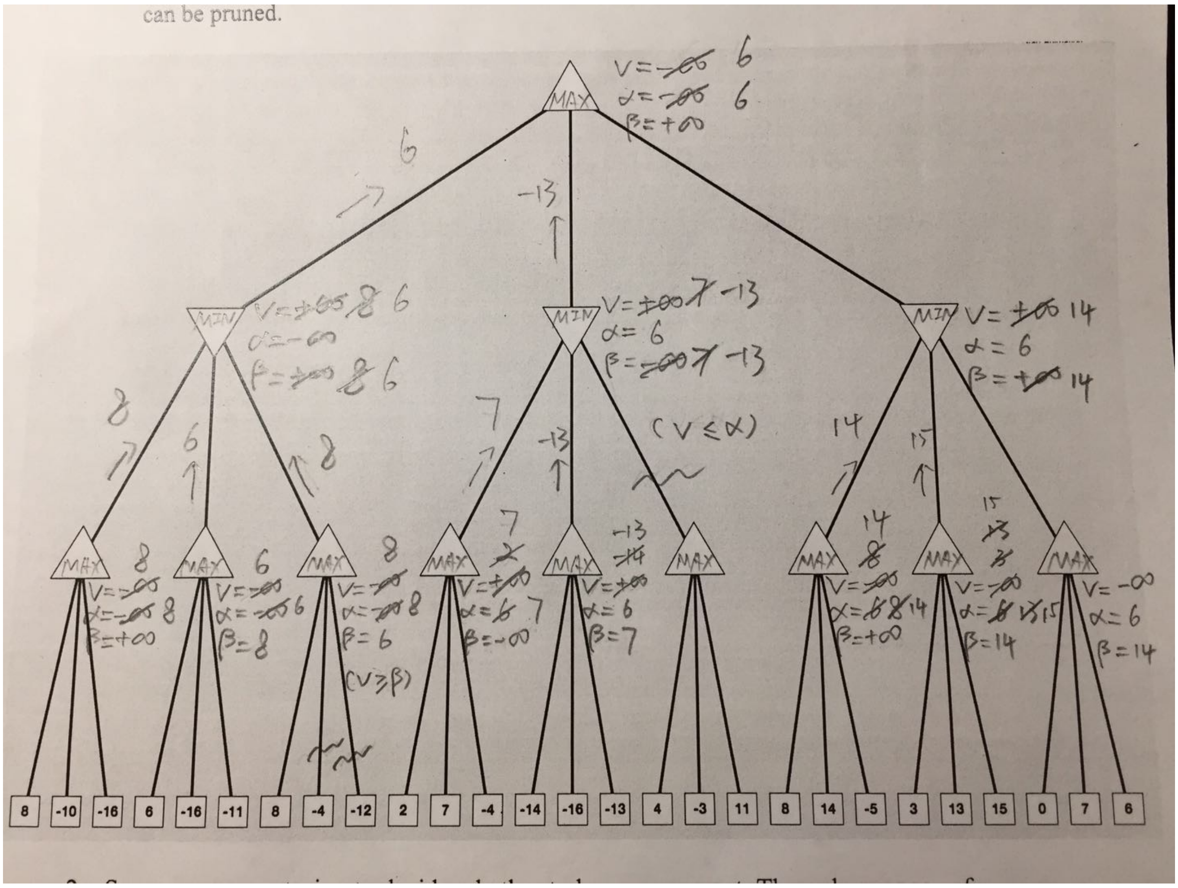
Repeat the same steps.



Repeat the same steps.



Repeat the same steps.



Repeat the same steps. Finally, we get V=7 at the root.