

# **Game Playing**

## **Part 2 Alpha-Beta Pruning**

**Yingyu Liang**

`yliang@cs.wisc.edu`

**Computer Sciences Department**  
**University of Wisconsin, Madison**

[based on slides from A. Moore <http://www.cs.cmu.edu/~awm/tutorials> , C. Dyer, J. Skrentny, Jerry Zhu]

## **alpha-beta pruning**

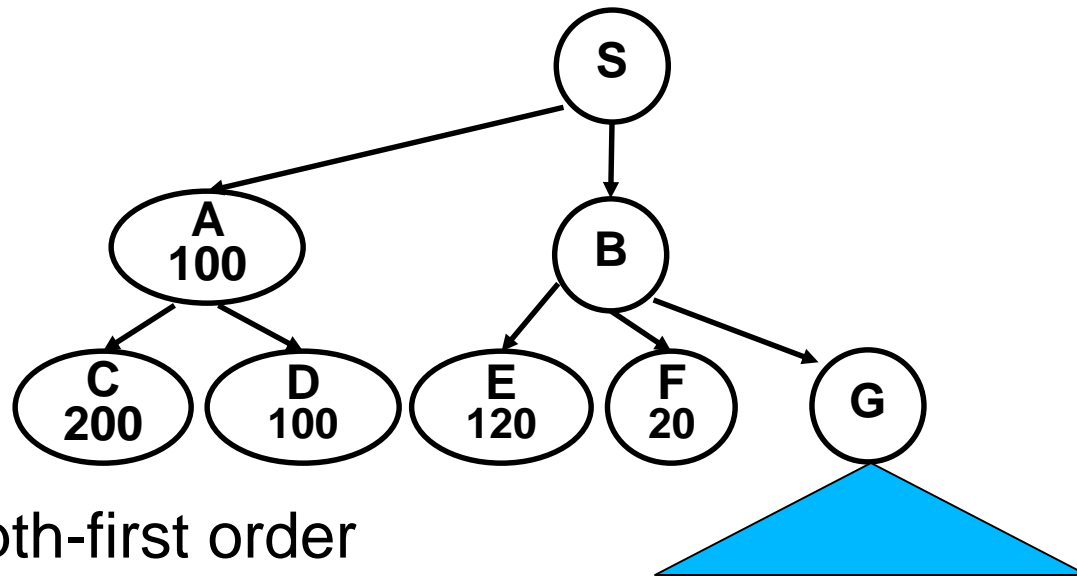
**Gives the same game theoretic values as minimax, but prunes part of the game tree.**

"If you have an idea that is surely bad, don't take the time to see how truly awful it is." -- Pat Winston

# Alpha-Beta Motivation

max

min



- Depth-first order
- After returning from A, Max can get at least 100 at S
- After returning from F, Max can get at most 20 at B
- At this point, Max loses interest in B
- There is no need to explore G. The subtree at G is pruned. Saves time.

# Alpha-beta pruning

function **Max-Value** (s,  $\alpha$ ,  $\beta$ )

inputs:

s: current state in game, Max about to play

$\alpha$ : best score (highest) for Max along path to s

$\beta$ : best score (lowest) for Min along path to s

output: *min( $\beta$  , best-score (for Max) available from s)*

if ( s is a terminal state )

then return ( terminal value of s )

else for each s' in Succ(s)

$\alpha := \max( \alpha , \text{Min-value}(s', \alpha, \beta) )$

if (  $\alpha \geq \beta$  ) then return  $\beta$  /\* alpha pruning \*/

return  $\alpha$

Starting from the root:

Max-Value(root,  $-\infty$ ,  $+\infty$ )

# Alpha-beta pruning

function **Max-Value** (s,  $\alpha$ ,  $\beta$ )

**inputs:**

s: current state in game, Max about to play

$\alpha$ : best score (highest) for Max along path to s

$\beta$ : best score (lowest) for Min along path to s

**output:**  $\min(\beta, \text{best-score (for Max) available from s})$

if ( s is a terminal state )

then return ( terminal value of s )

else for each s' in Succ(s)

$\alpha := \max(\alpha, \text{Min-value}(s', \alpha, \beta))$

if (  $\alpha \geq \beta$  ) then return  $\beta$  /\* alpha pruning \*/

return  $\alpha$

function **Min-Value**(s,  $\alpha$ ,  $\beta$ )

**output:**  $\max(\alpha, \text{best-score (for Min) available from s})$

if ( s is a terminal state )

then return ( terminal value of s )

else for each s' in Succs(s)

$\beta := \min(\beta, \text{Max-value}(s', \alpha, \beta))$

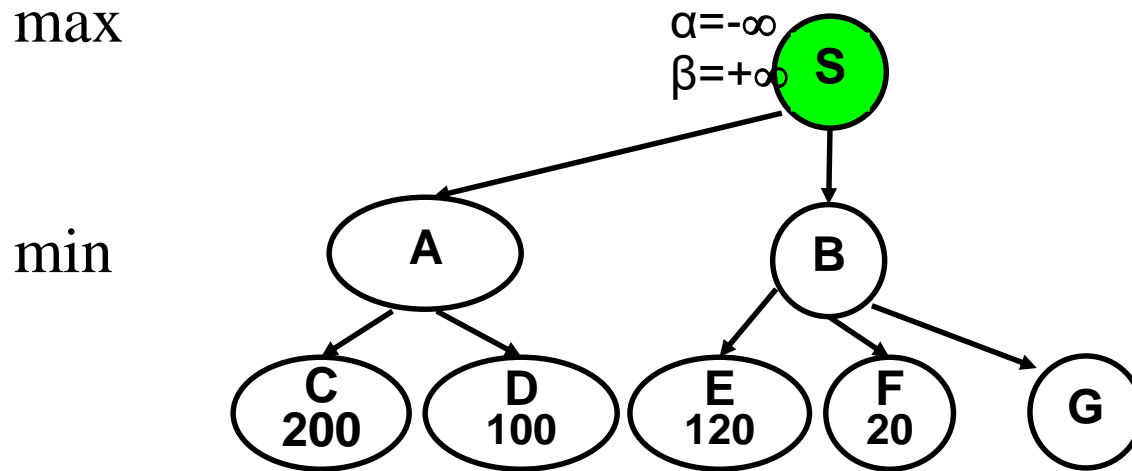
if (  $\alpha \geq \beta$  ) then return  $\alpha$  /\* beta pruning \*/

return  $\beta$

Starting from the root:

Max-Value(root,  $-\infty$ ,  $+\infty$ )

# Alpha-beta pruning example 1

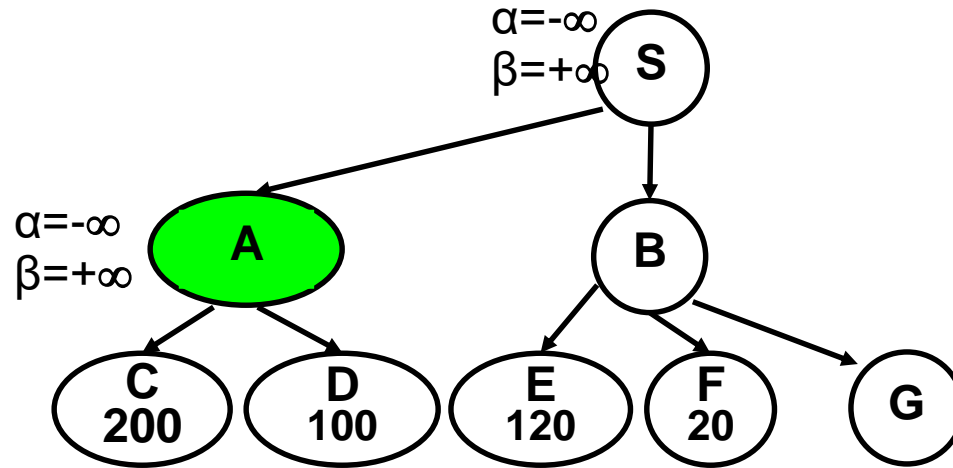


- Keep two bounds along the path
  - $\alpha$ : the best Max can do
  - $\beta$ : the best (smallest) Min can do
- If at anytime  $\alpha$  exceeds  $\beta$ , the remaining children are pruned.

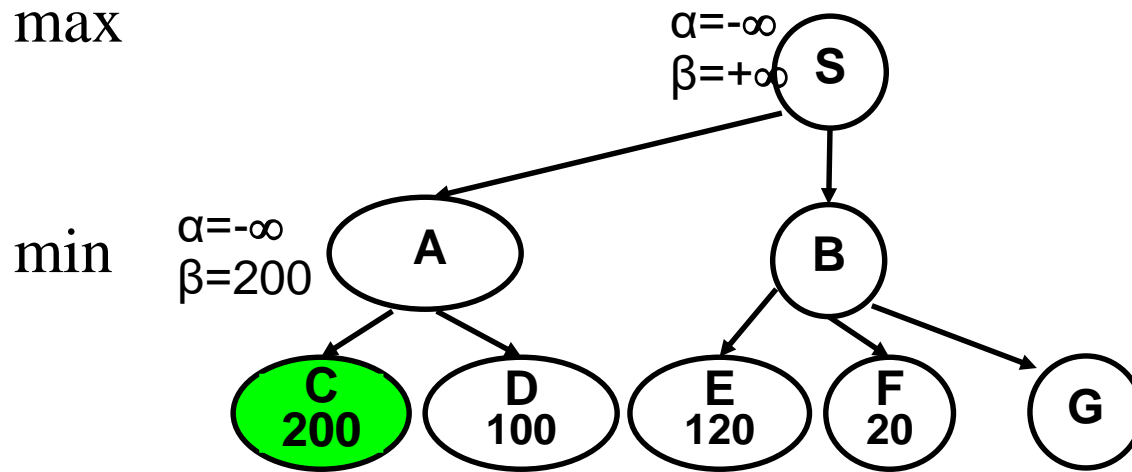
# Alpha-beta pruning example 1

max

min

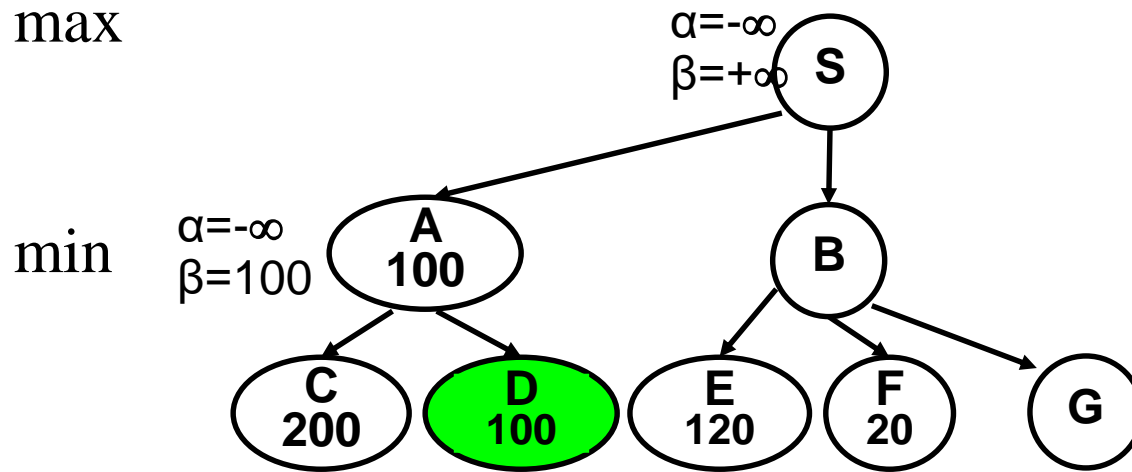


# Alpha-beta pruning example 1

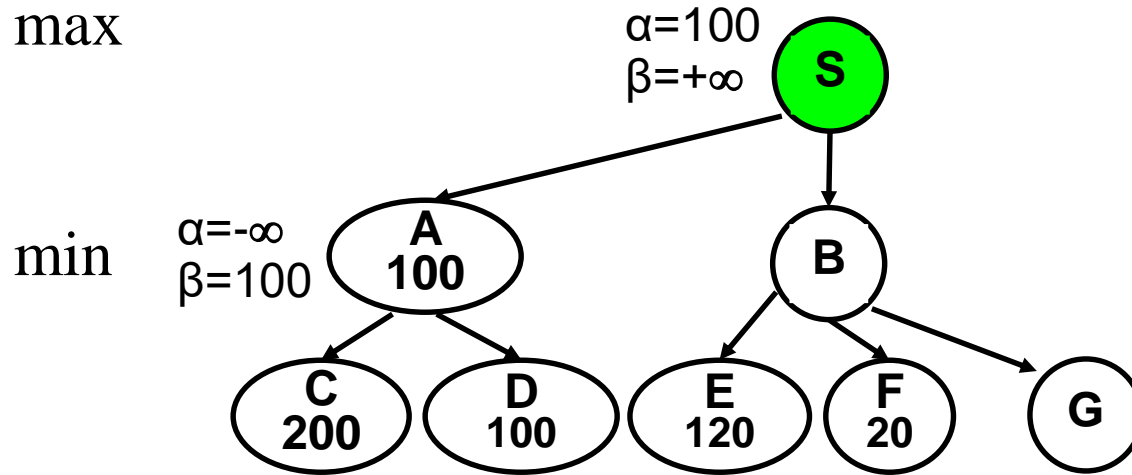




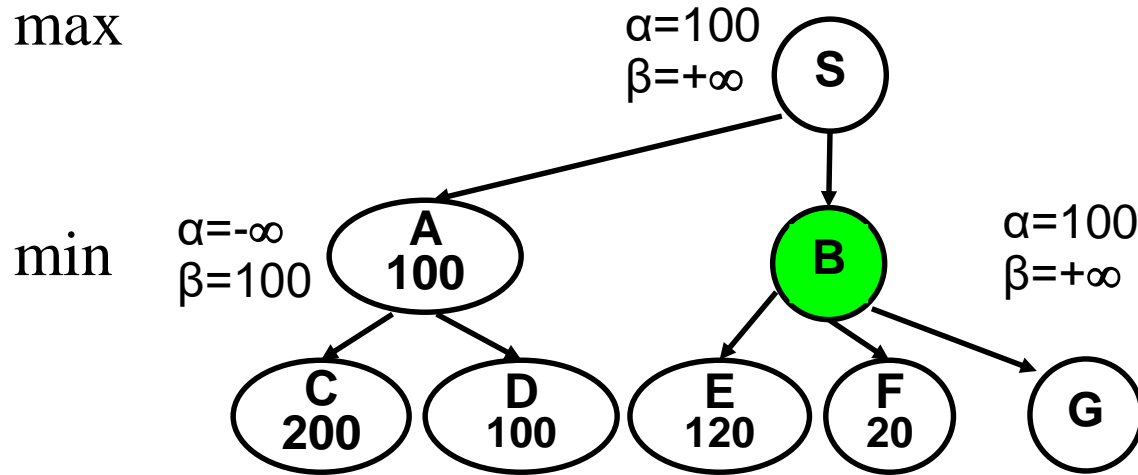
# Alpha-beta pruning example 1



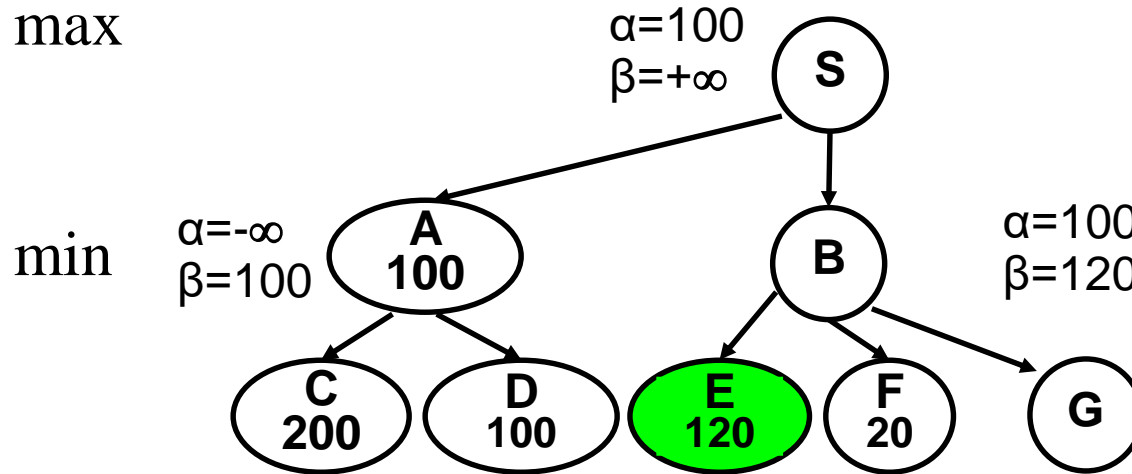
# Alpha-beta pruning example 1



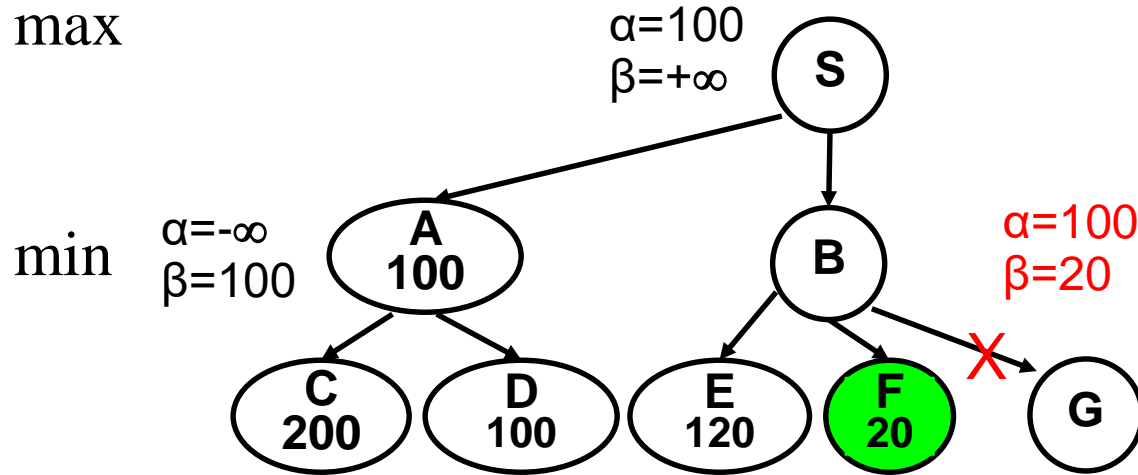
# Alpha-beta pruning example 1



# Alpha-beta pruning example 1



# Alpha-beta pruning example 1



# Alpha-beta pruning

function **Max-Value** (s,  $\alpha$ ,  $\beta$ )

**inputs:**

s: current state in game, Max about to play

$\alpha$ : best score (highest) for Max along path to s

$\beta$ : best score (lowest) for Min along path to s

**output:**  $\min(\beta, \text{best-score (for Max) available from s})$

if ( s is a terminal state )

then return ( terminal value of s )

else for each s' in Succ(s)

$\alpha := \max(\alpha, \text{Min-value}(s', \alpha, \beta))$

if (  $\alpha \geq \beta$  ) then return  $\beta$  /\* alpha pruning \*/

return  $\alpha$

function **Min-Value**(s,  $\alpha$ ,  $\beta$ )

**output:**  $\max(\alpha, \text{best-score (for Min) available from s})$

if ( s is a terminal state )

then return ( terminal value of s )

else for each s' in Succs(s)

$\beta := \min(\beta, \text{Max-value}(s', \alpha, \beta))$

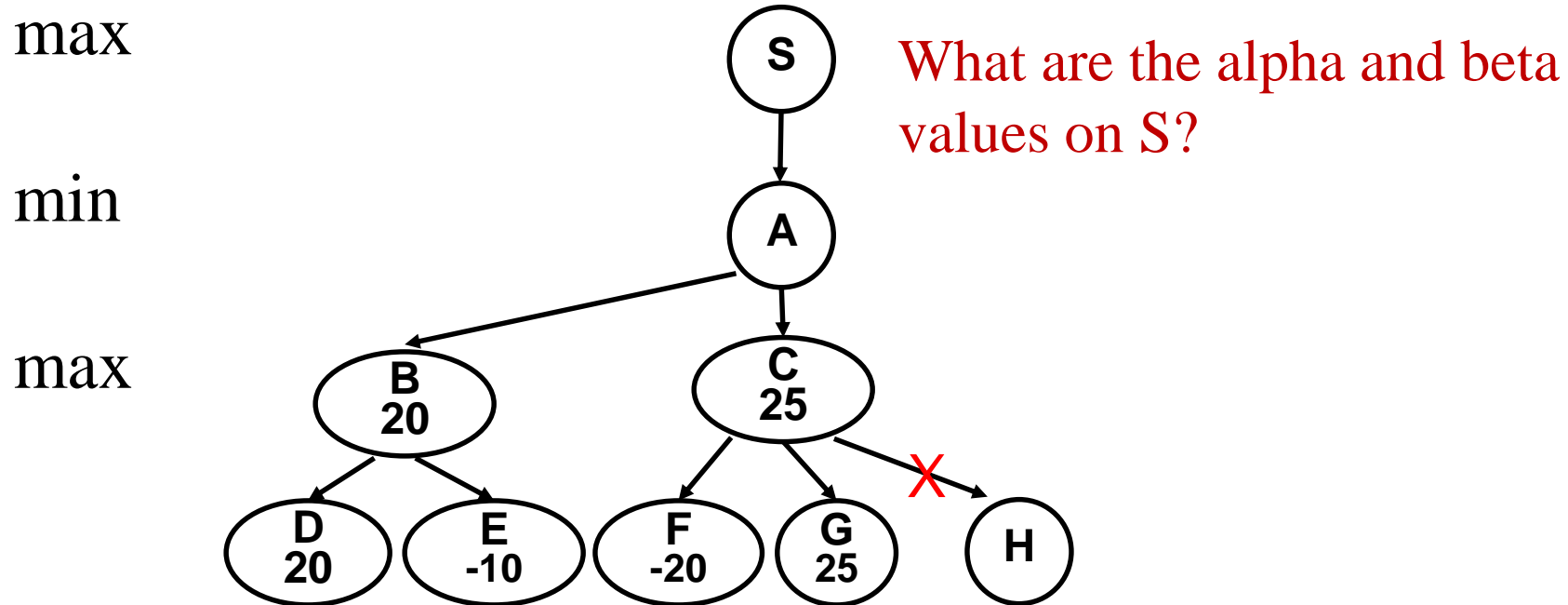
if (  $\alpha \geq \beta$  ) then return  $\alpha$  /\* beta pruning \*/

return  $\beta$

Starting from the root:

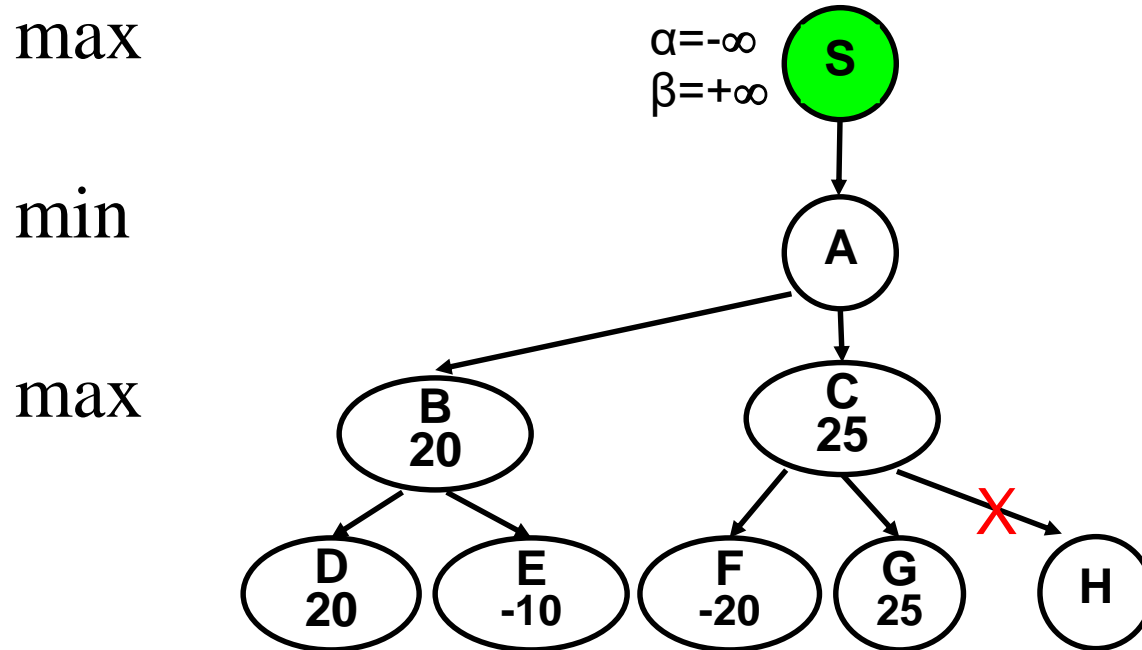
Max-Value(root,  $-\infty$ ,  $+\infty$ )

# Alpha-beta pruning example 2



- Keep two bounds along the path
  - $\alpha$ : the best Max can do
  - $\beta$ : the best (smallest) Min can do
- If at anytime  $\alpha$  exceeds  $\beta$ , the remaining children are pruned.

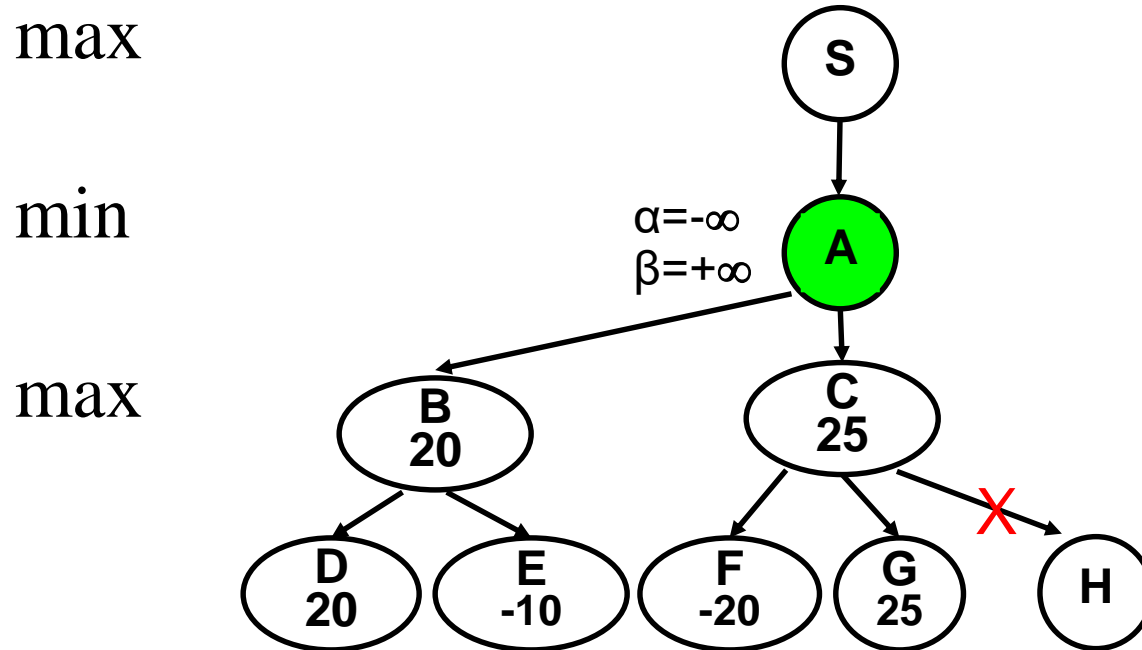
# Alpha-beta pruning example 2



- Keep two bounds along the path
  - $\alpha$ : the best Max can do
  - $\beta$ : the best (smallest) Min can do
- If at anytime  $\alpha$  exceeds  $\beta$ , the remaining children are pruned.

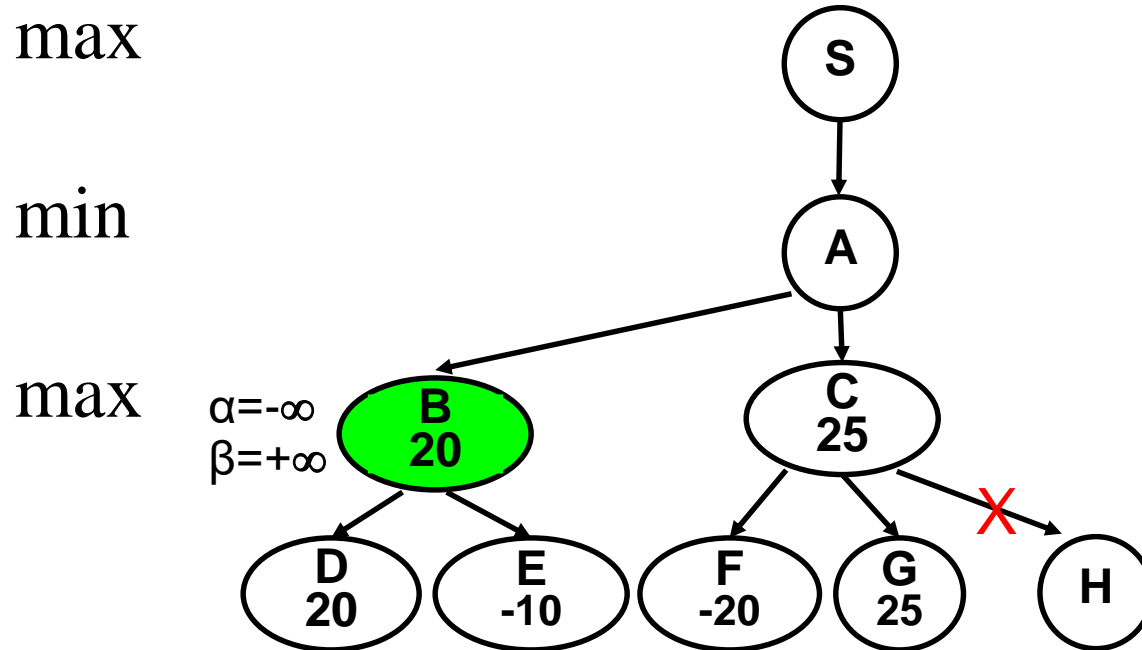


# Alpha-beta pruning example 2



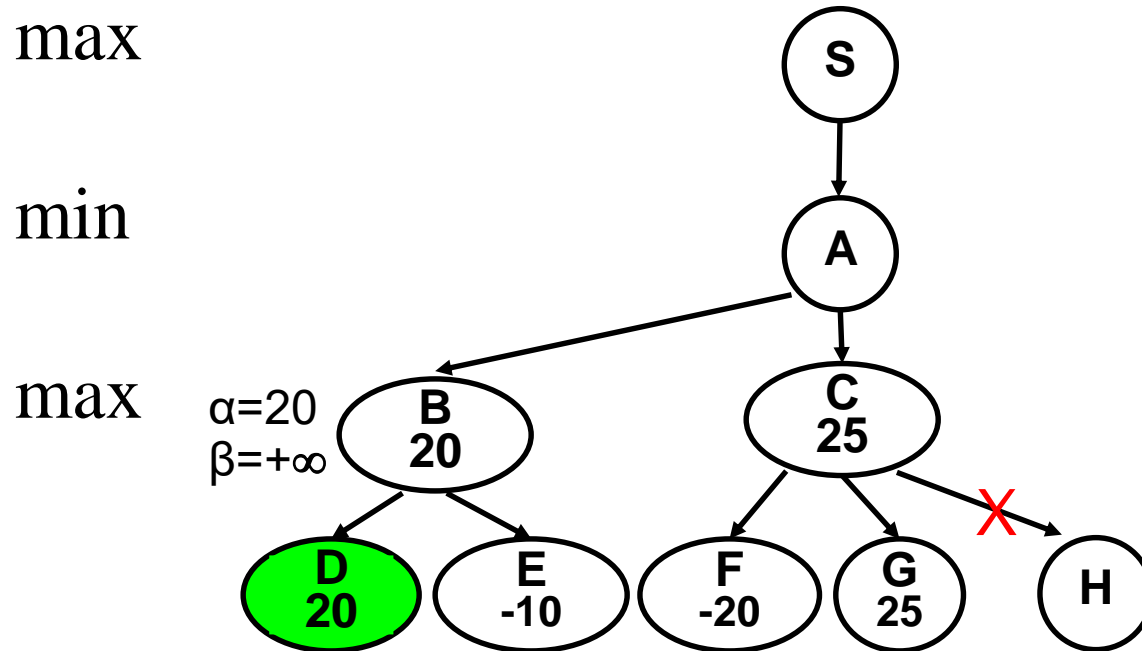
- Keep two bounds along the path
  - $\alpha$ : the best Max can do
  - $\beta$ : the best (smallest) Min can do
- If at anytime  $\alpha$  exceeds  $\beta$ , the remaining children are pruned.

# Alpha-beta pruning example 2



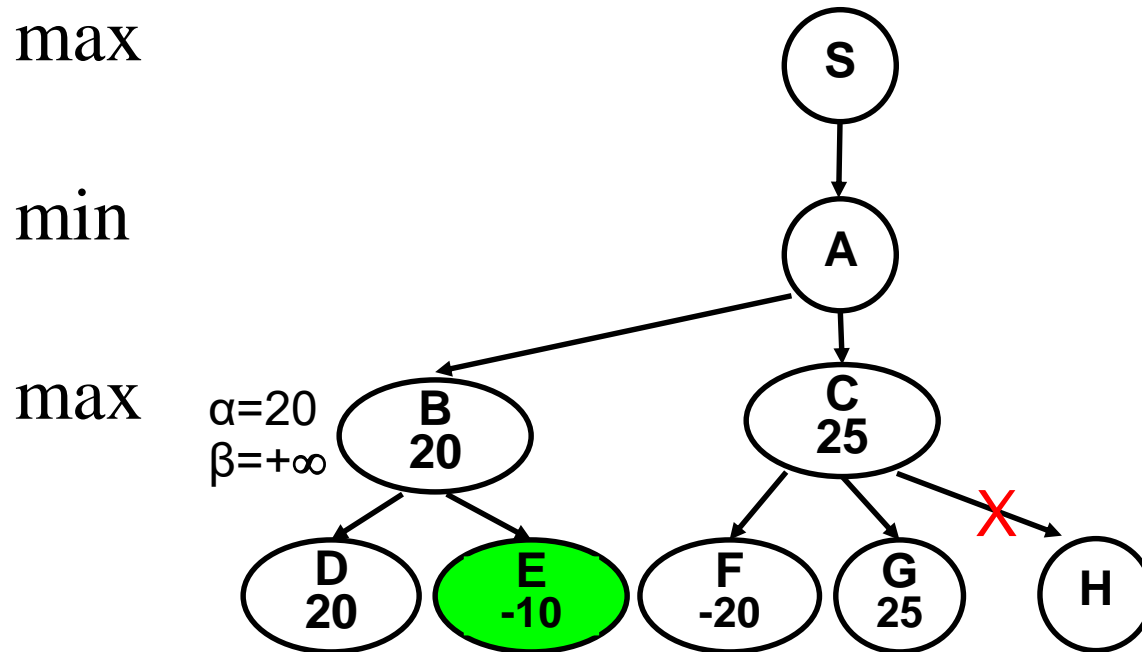
- Keep two bounds along the path
  - $\alpha$ : the best Max can do
  - $\beta$ : the best (smallest) Min can do
- If at anytime  $\alpha$  exceeds  $\beta$ , the remaining children are pruned.

# Alpha-beta pruning example 2



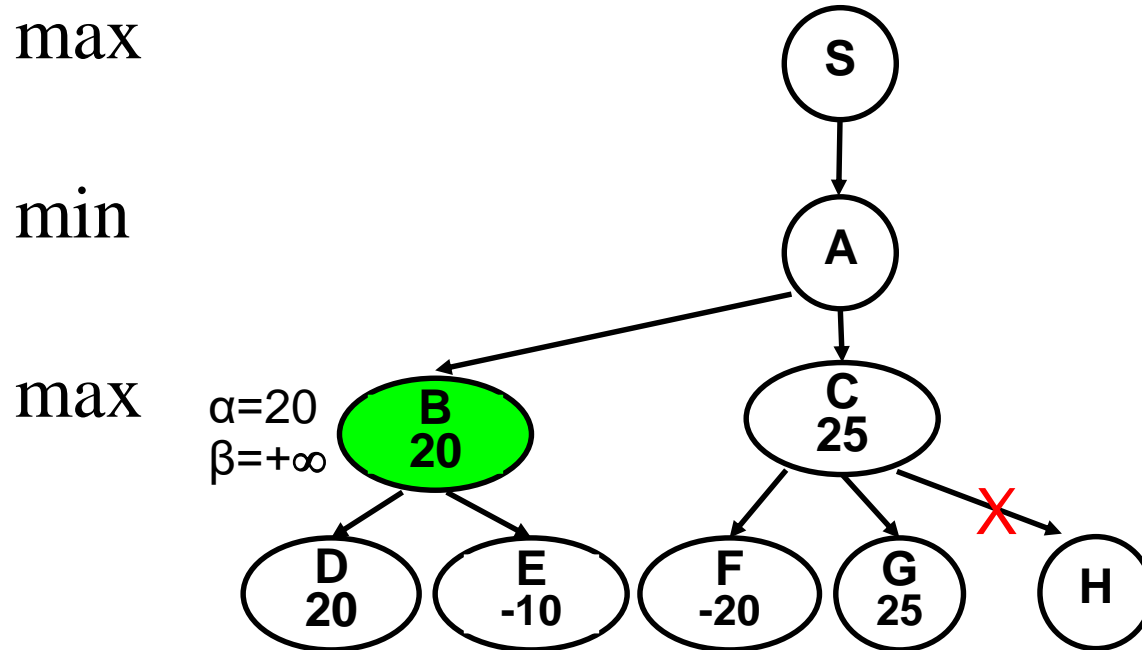
- Keep two bounds along the path
  - $\alpha$ : the best Max can do
  - $\beta$ : the best (smallest) Min can do
- If at anytime  $\alpha$  exceeds  $\beta$ , the remaining children are pruned.

# Alpha-beta pruning example 2



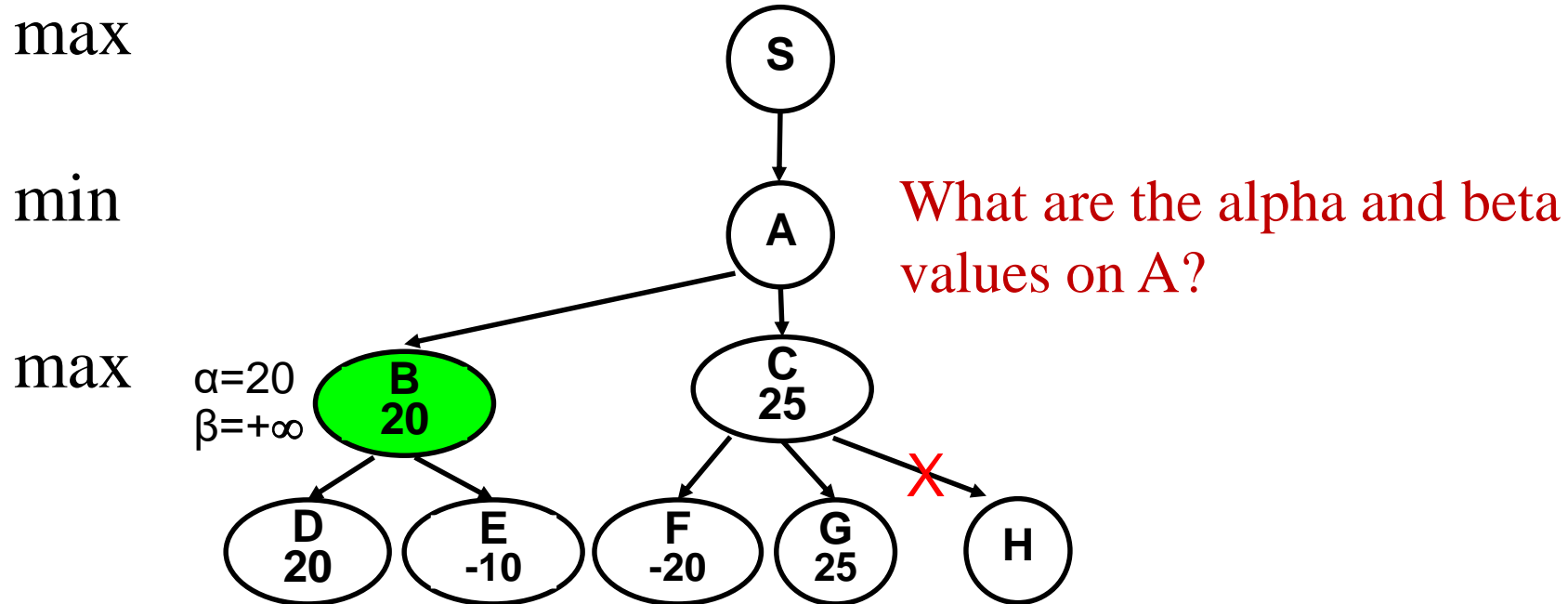
- Keep two bounds along the path
  - $\alpha$ : the best Max can do
  - $\beta$ : the best (smallest) Min can do
- If at anytime  $\alpha$  exceeds  $\beta$ , the remaining children are pruned.

# Alpha-beta pruning example 2



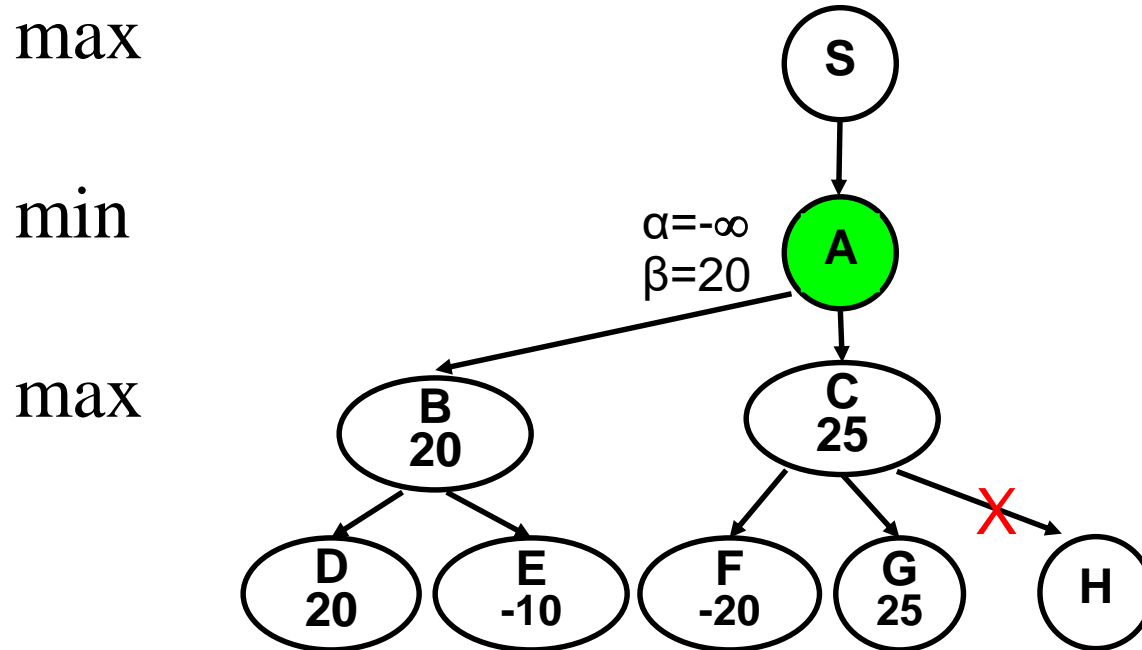
- Keep two bounds along the path
  - $\alpha$ : the best Max can do
  - $\beta$ : the best (smallest) Min can do
- If at anytime  $\alpha$  exceeds  $\beta$ , the remaining children are pruned.

# Alpha-beta pruning example 2



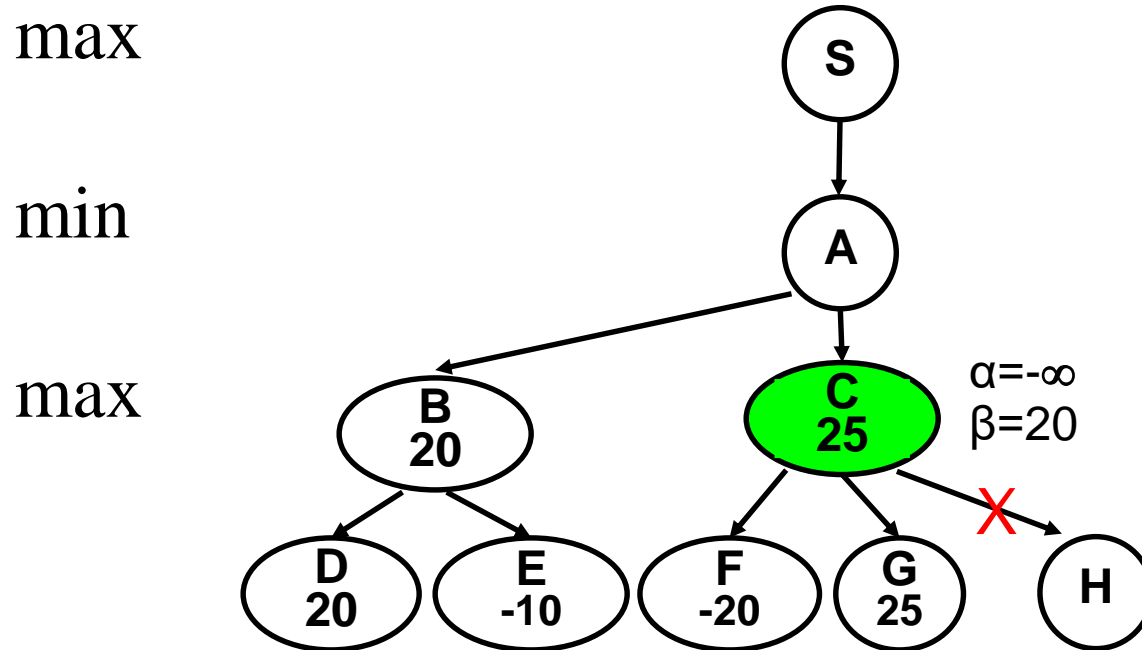
- Keep two bounds along the path
  - $\alpha$ : the best Max can do
  - $\beta$ : the best (smallest) Min can do
- If at anytime  $\alpha$  exceeds  $\beta$ , the remaining children are pruned.

# Alpha-beta pruning example 2



- Keep two bounds along the path
  - $\alpha$ : the best Max can do
  - $\beta$ : the best (smallest) Min can do
- If at anytime  $\alpha$  exceeds  $\beta$ , the remaining children are pruned.

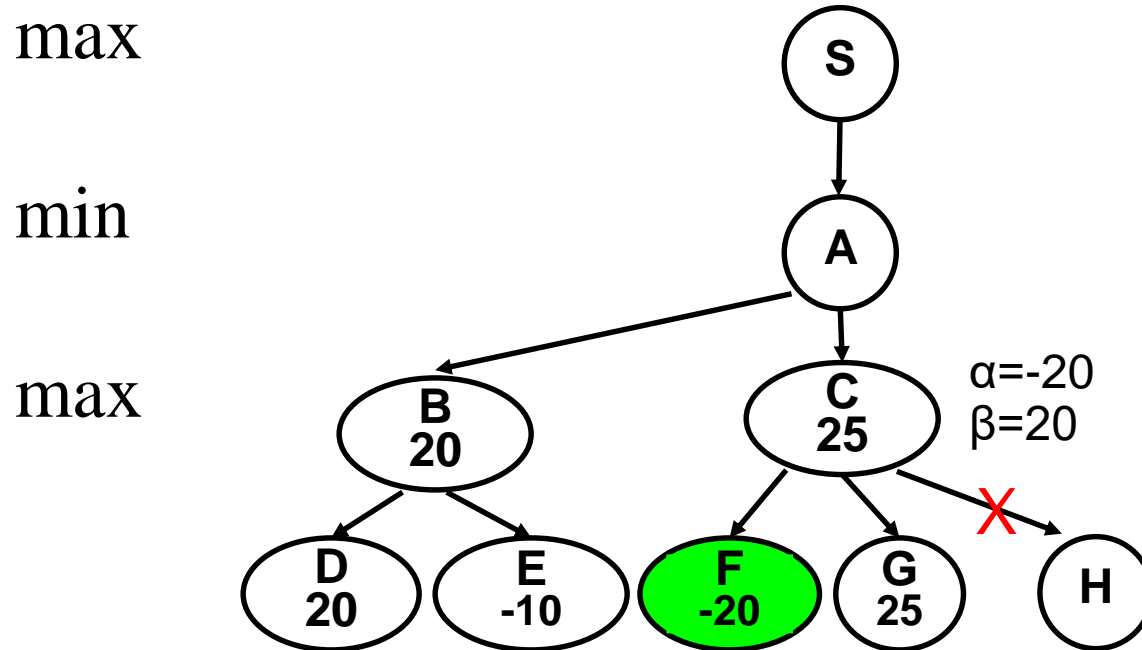
# Alpha-beta pruning example 2



- Keep two bounds along the path
  - $\alpha$ : the best Max can do
  - $\beta$ : the best (smallest) Min can do
- If at anytime  $\alpha$  exceeds  $\beta$ , the remaining children are pruned.

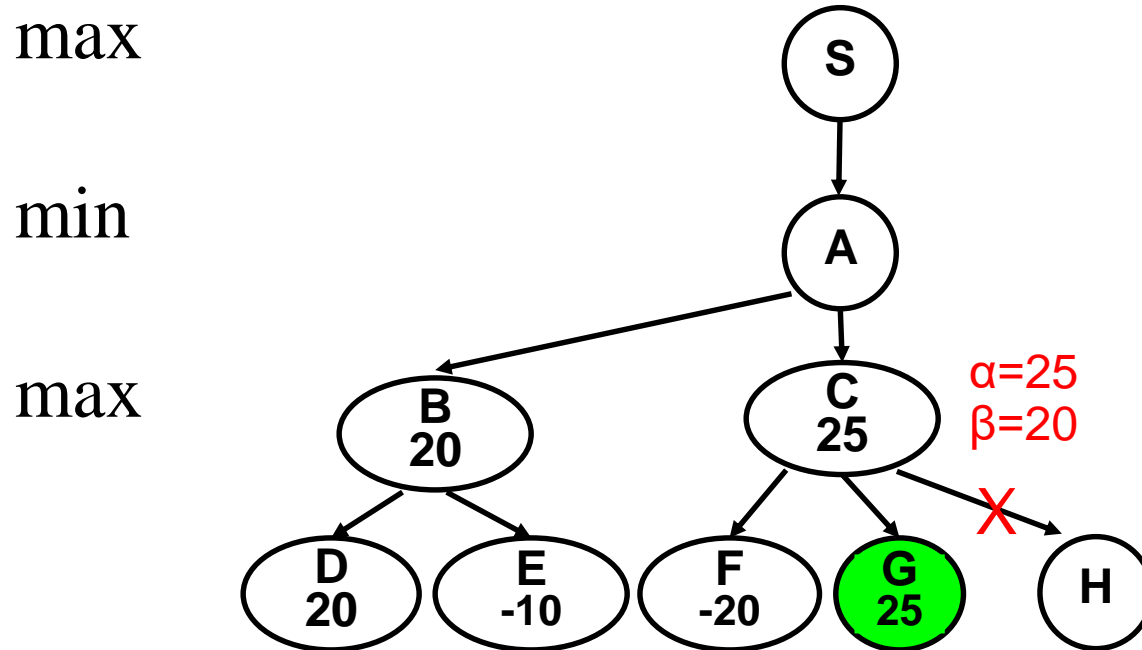


# Alpha-beta pruning example 2



- Keep two bounds along the path
  - $\alpha$ : the best Max can do
  - $\beta$ : the best (smallest) Min can do
- If at anytime  $\alpha$  exceeds  $\beta$ , the remaining children are pruned.

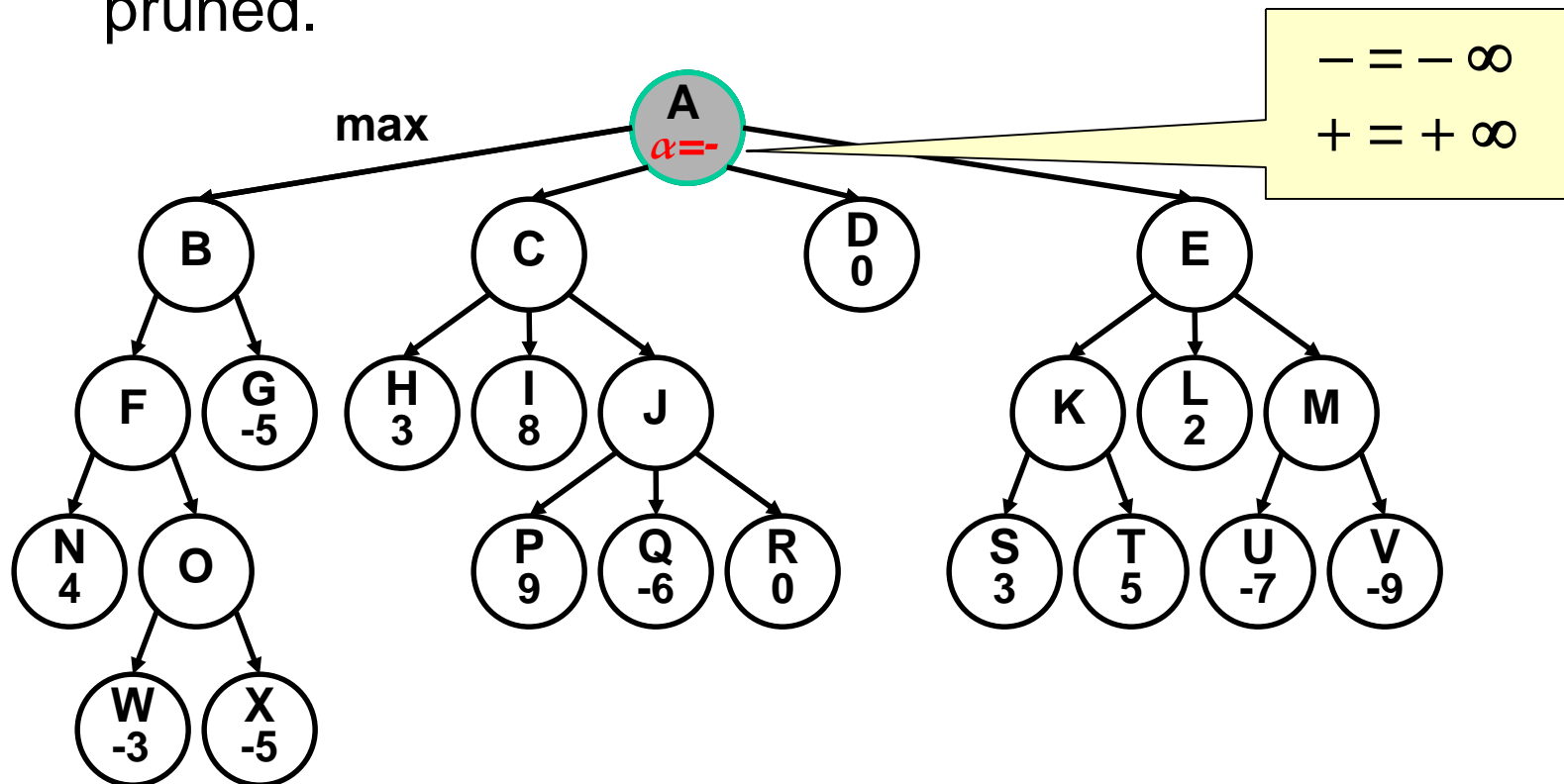
# Alpha-beta pruning example 2



- Keep two bounds along the path
  - $\alpha$ : the best Max can do
  - $\beta$ : the best (smallest) Min can do
- If at anytime  $\alpha$  exceeds  $\beta$ , the remaining children are pruned.

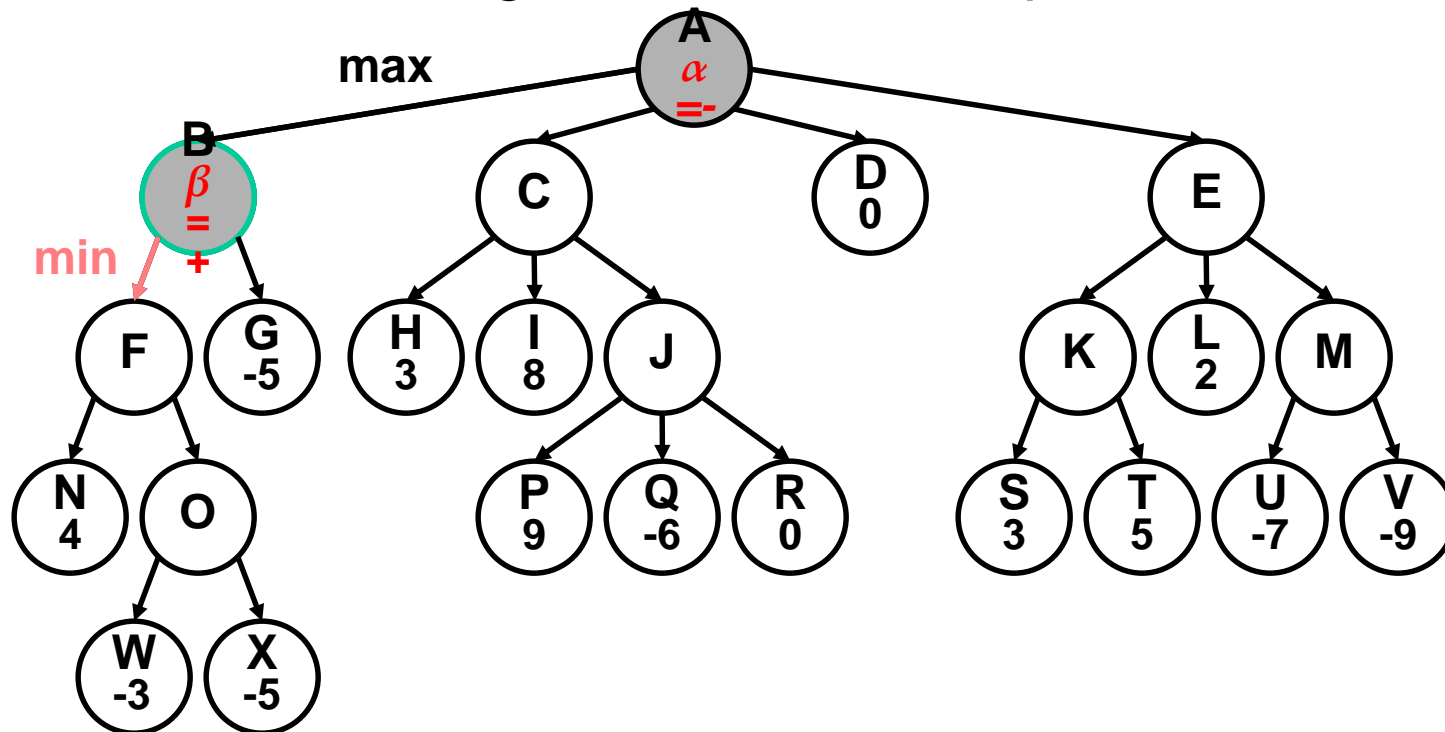
# Yet another alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If at anytime  $\alpha$  exceeds  $\beta$ , the remaining children are pruned.



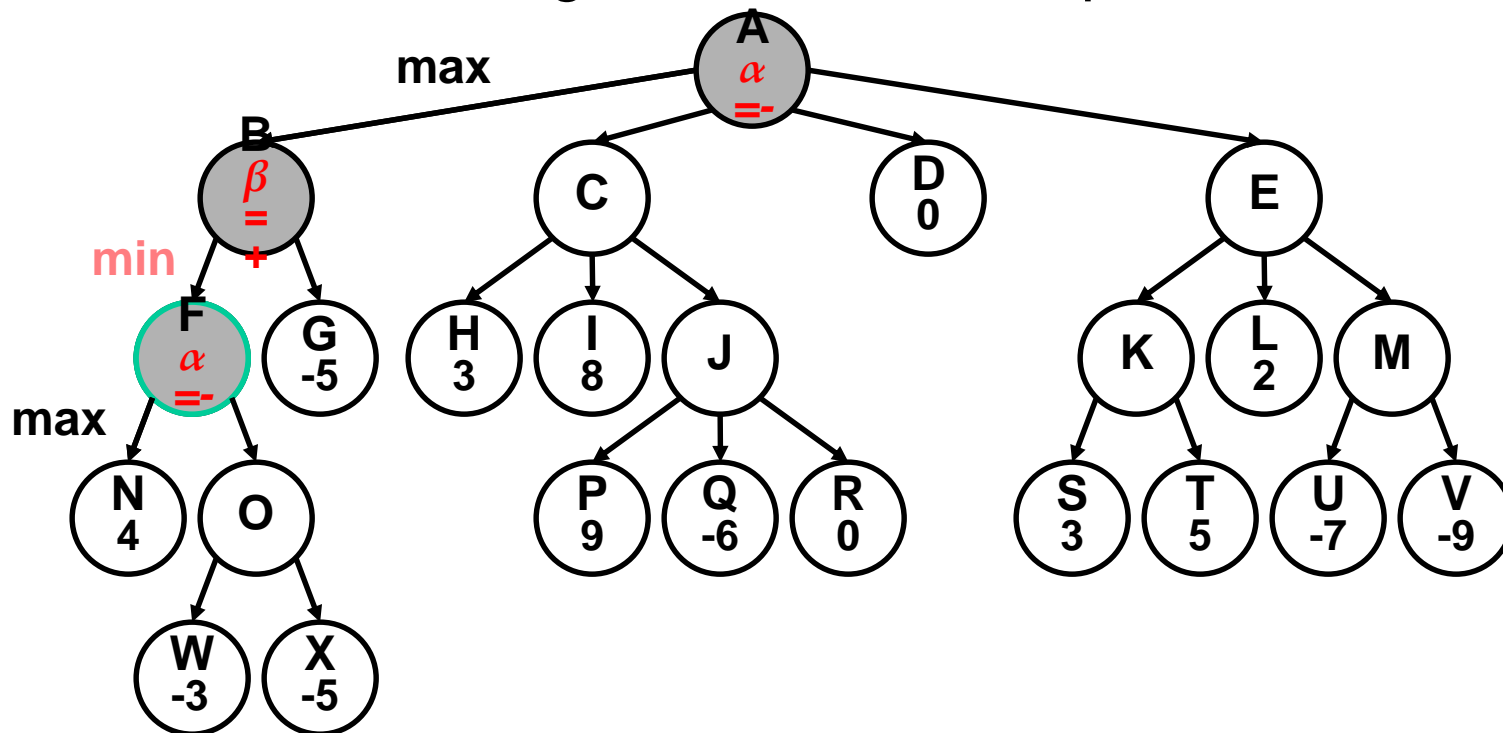
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



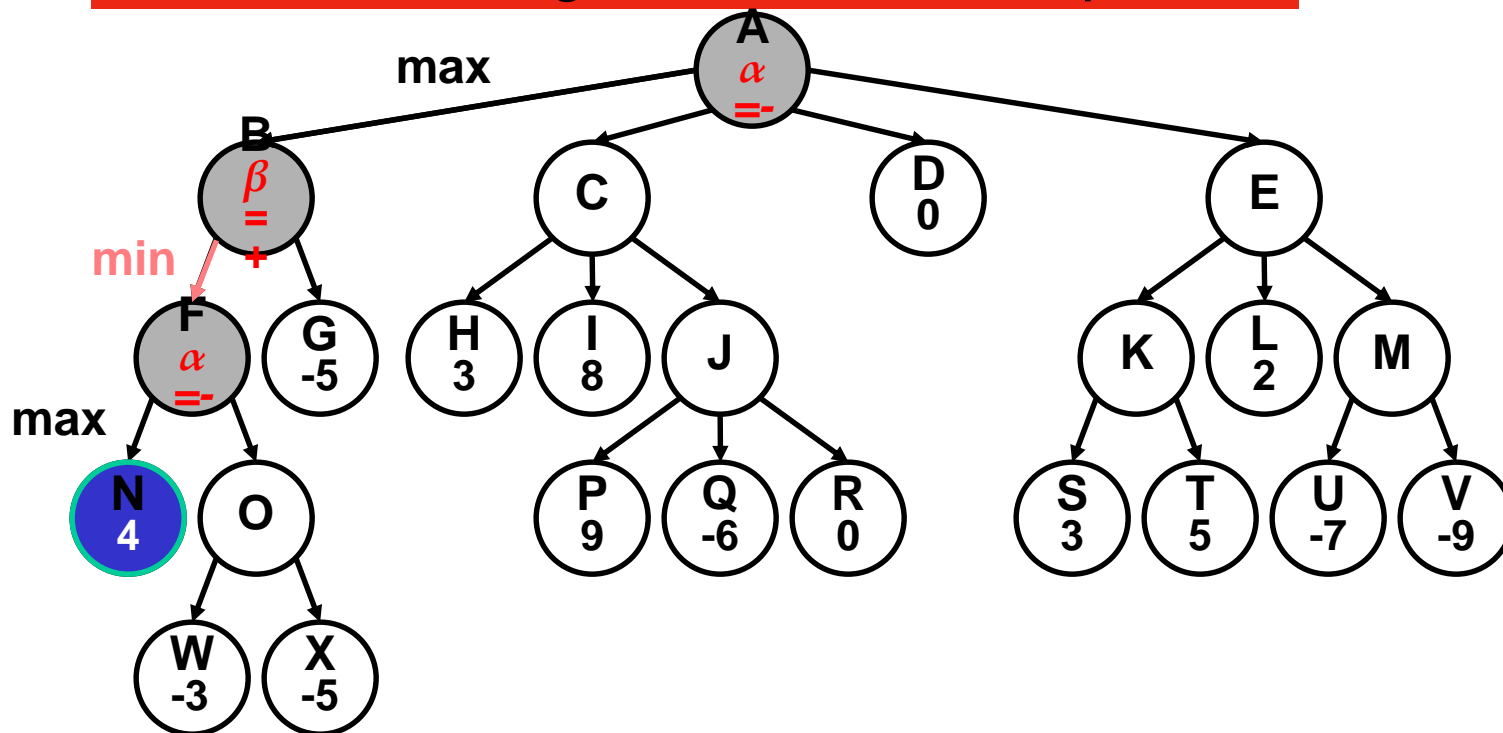
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



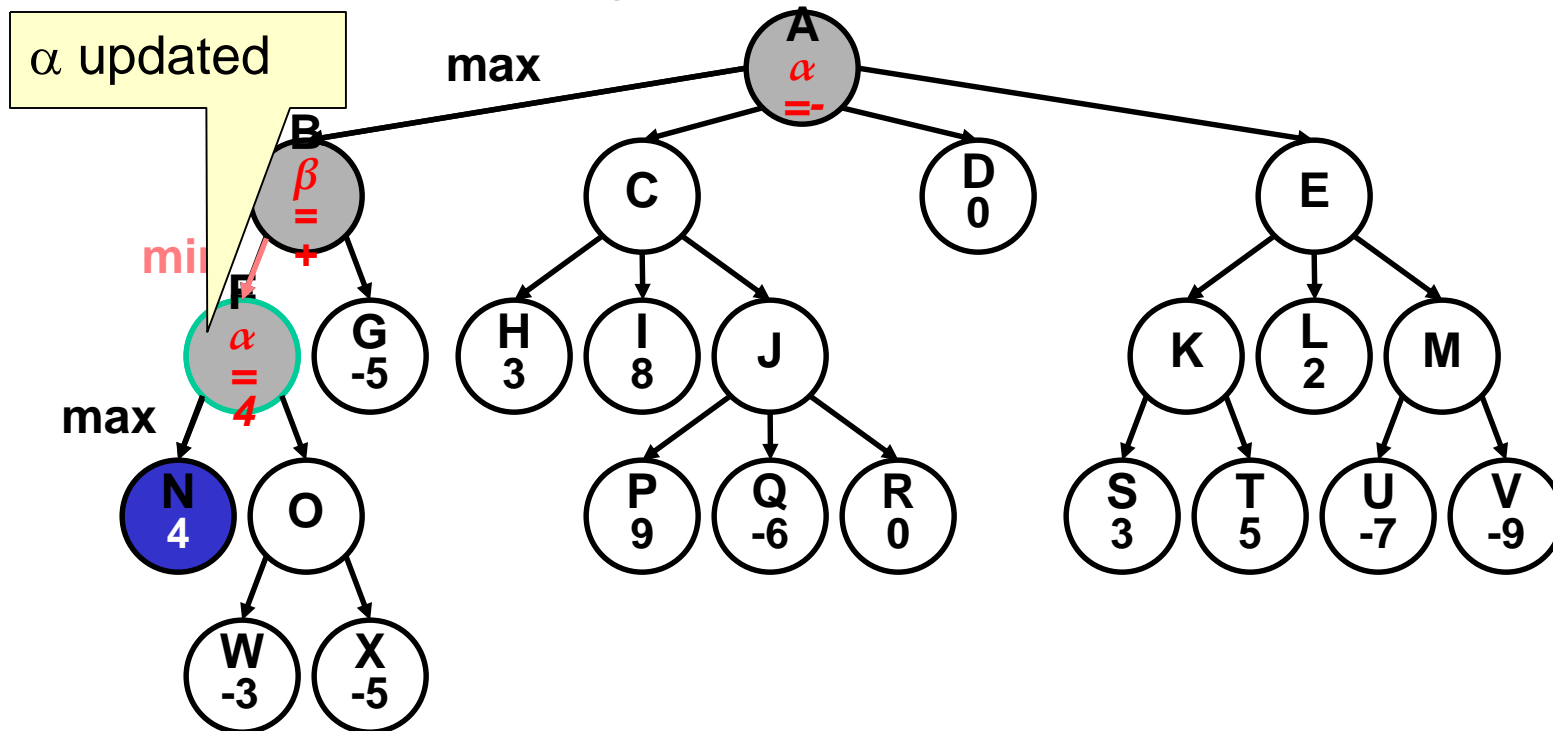
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



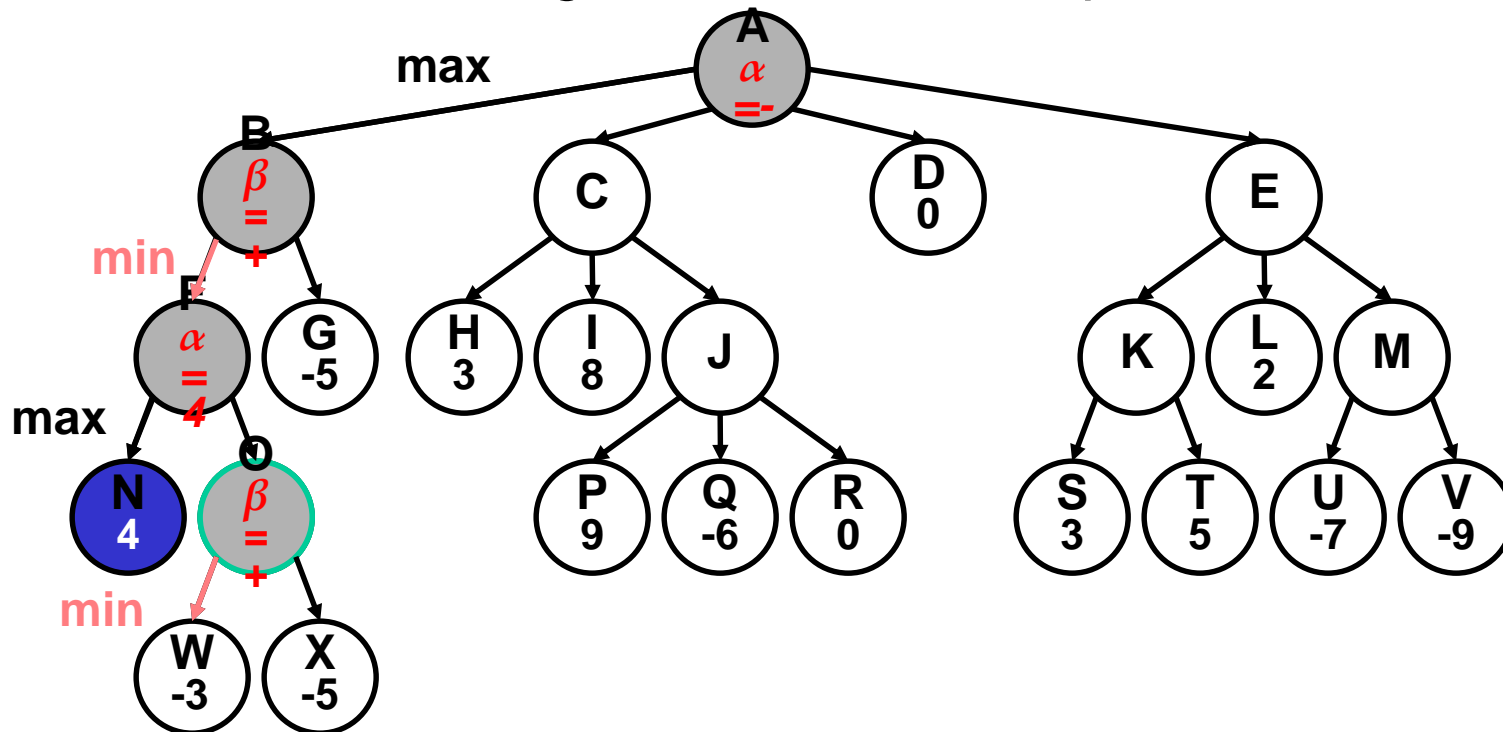
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



# Alpha-beta pruning example

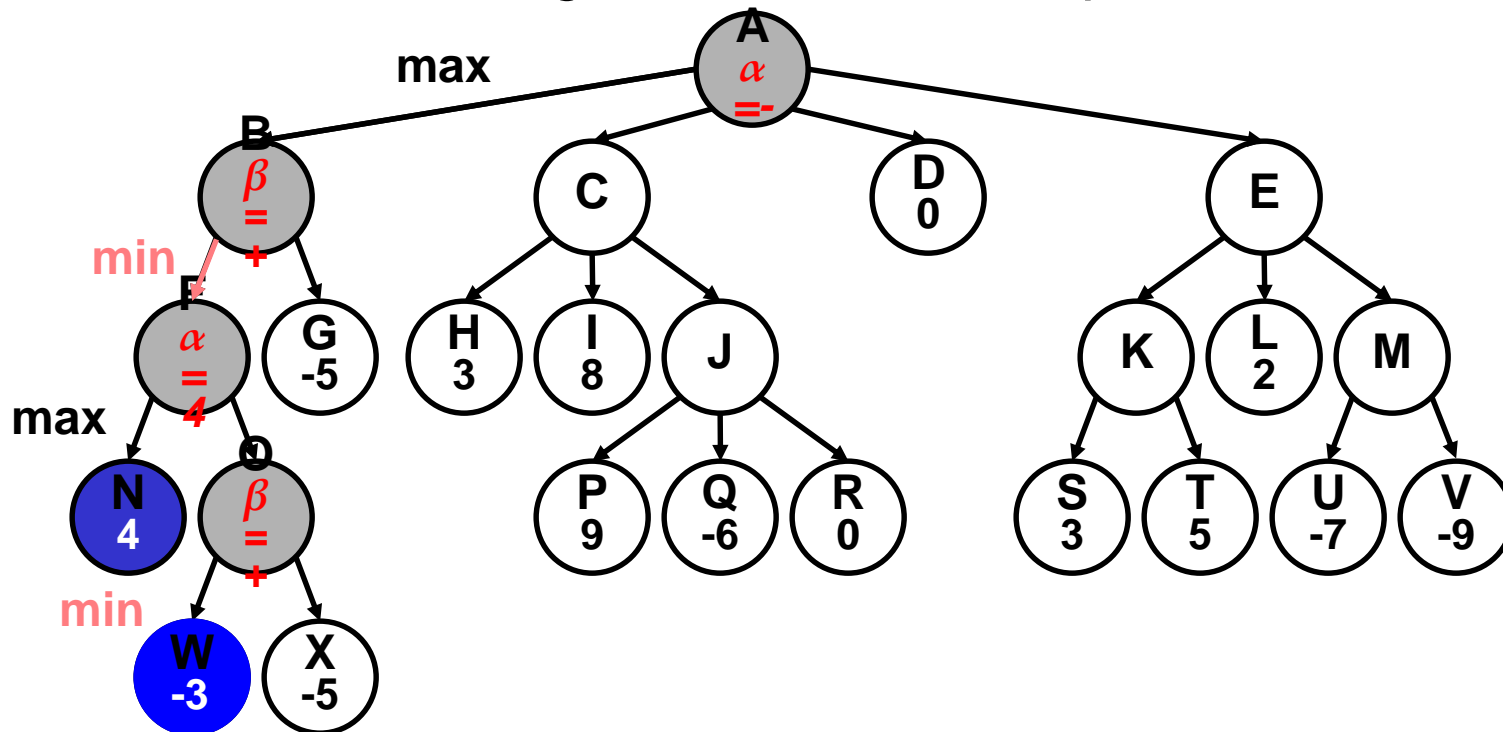
- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.





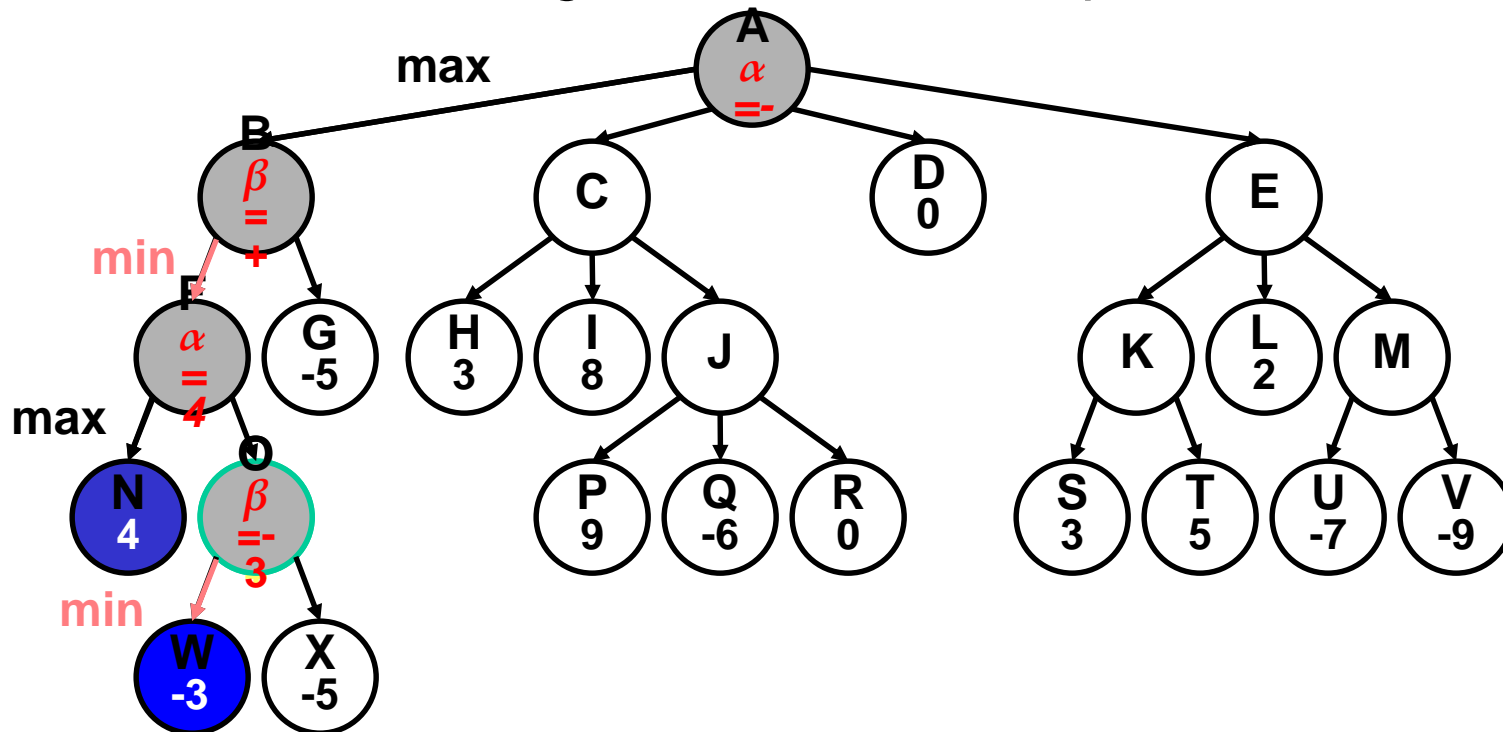
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



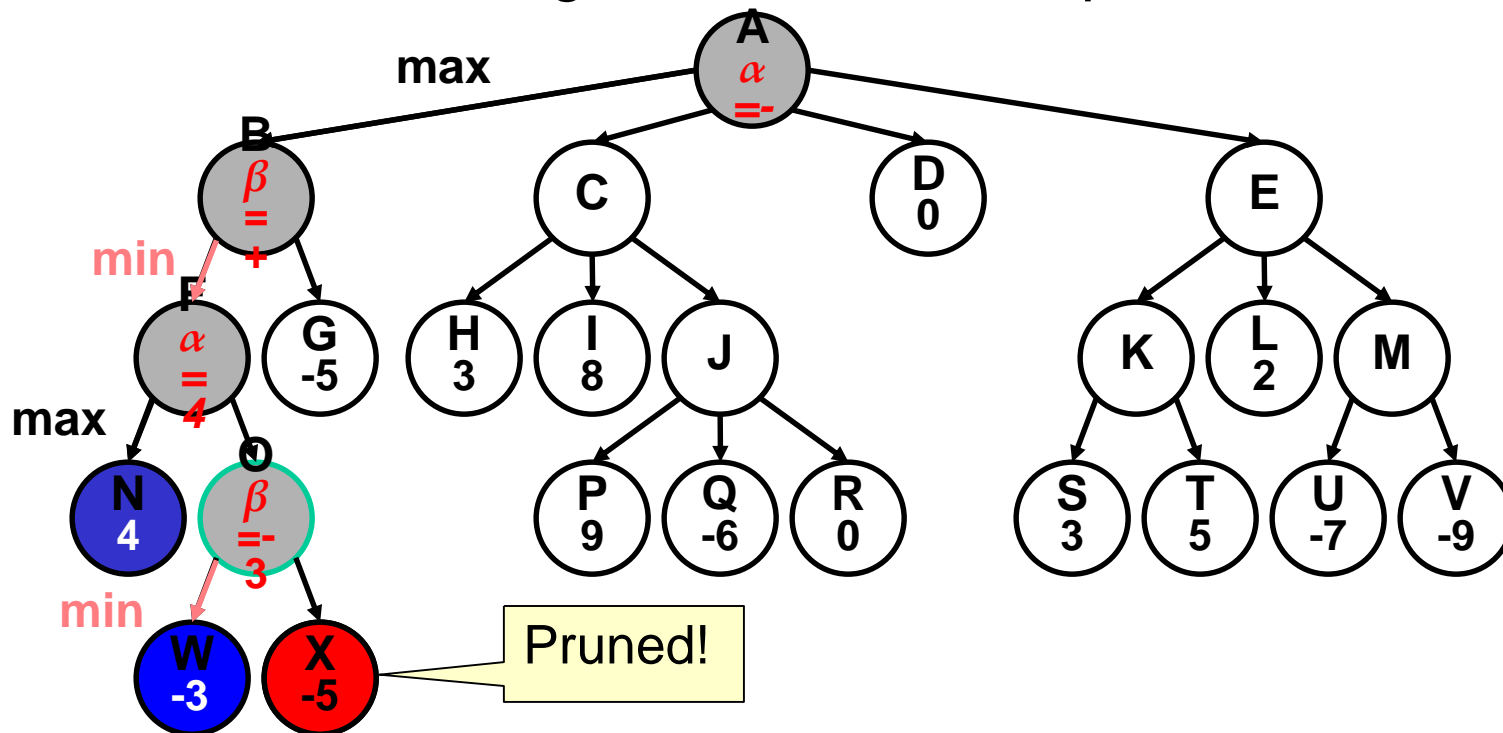
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



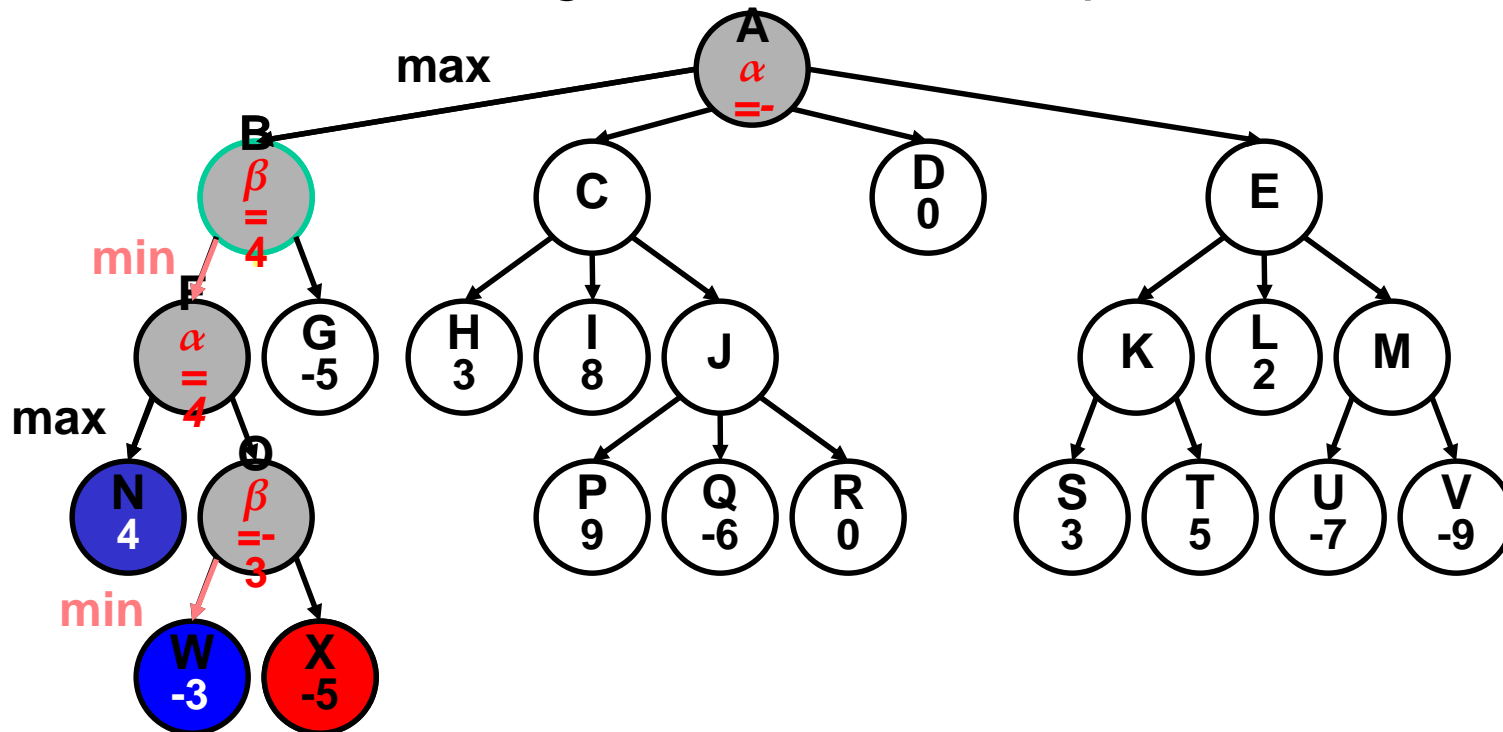
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



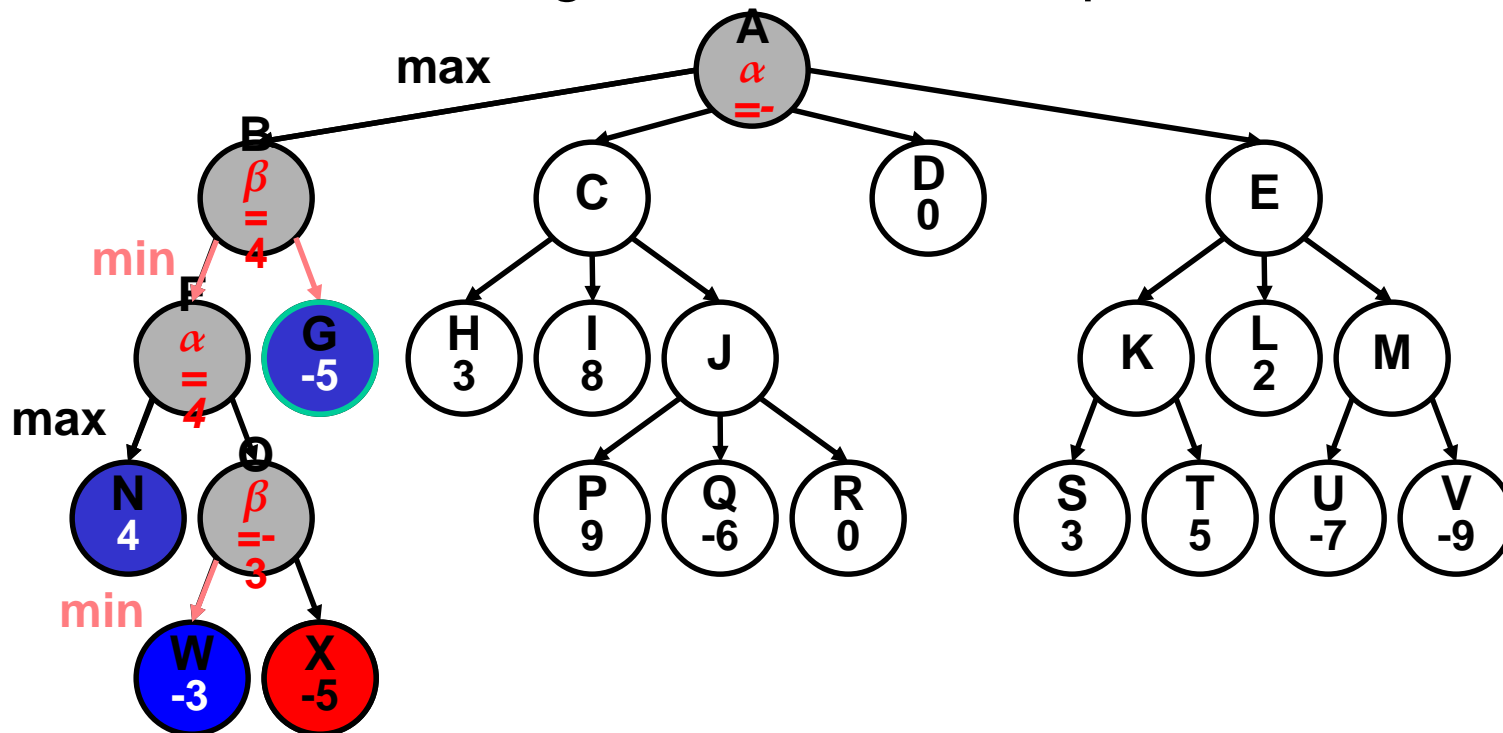
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



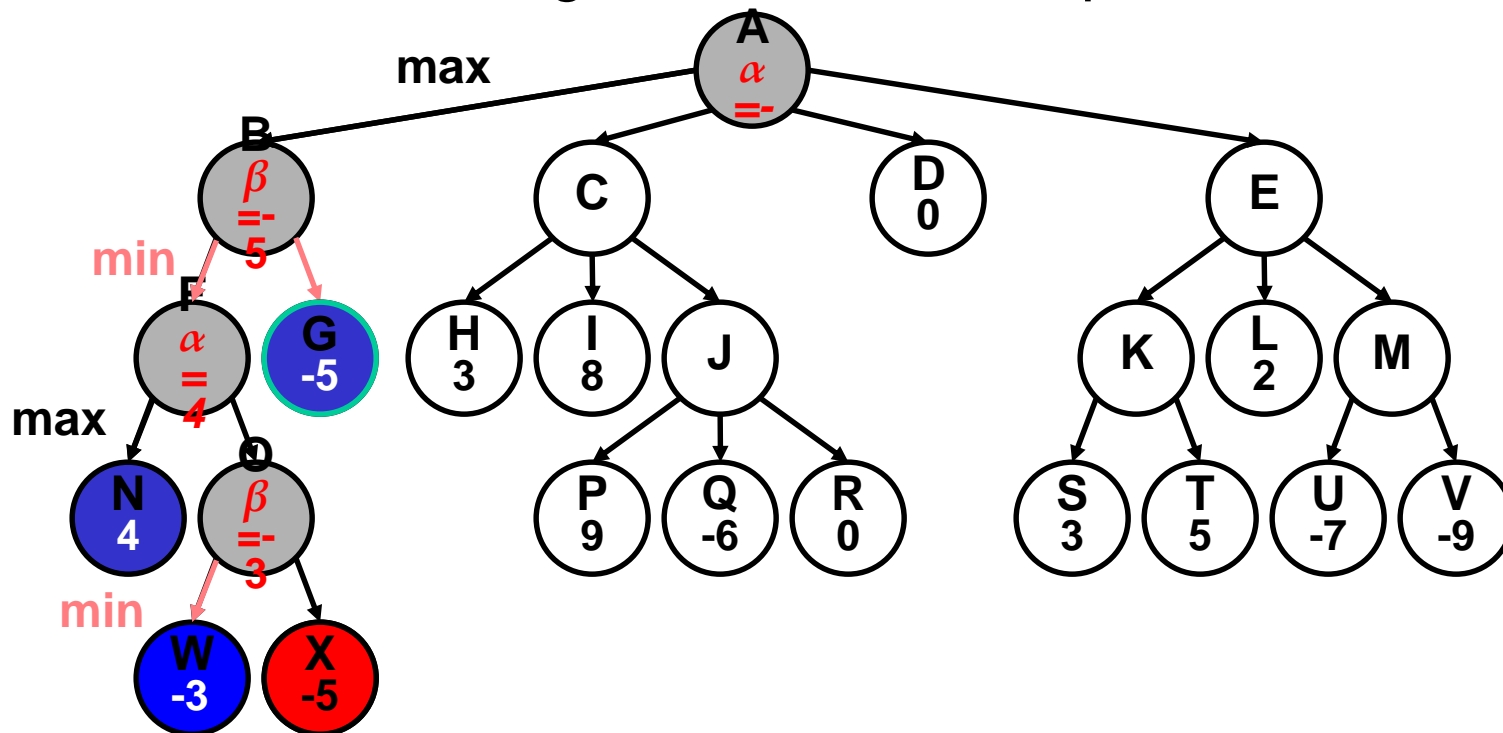
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



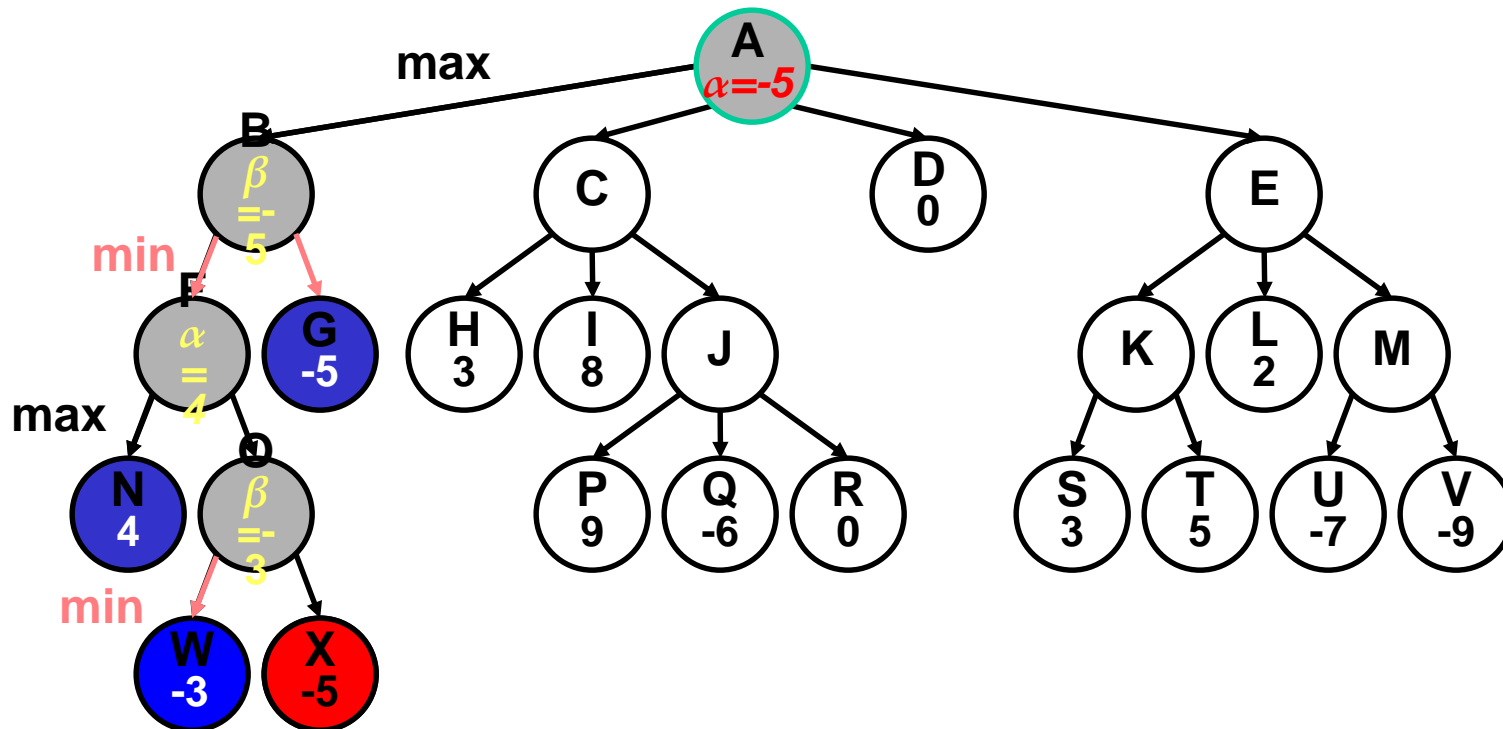
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



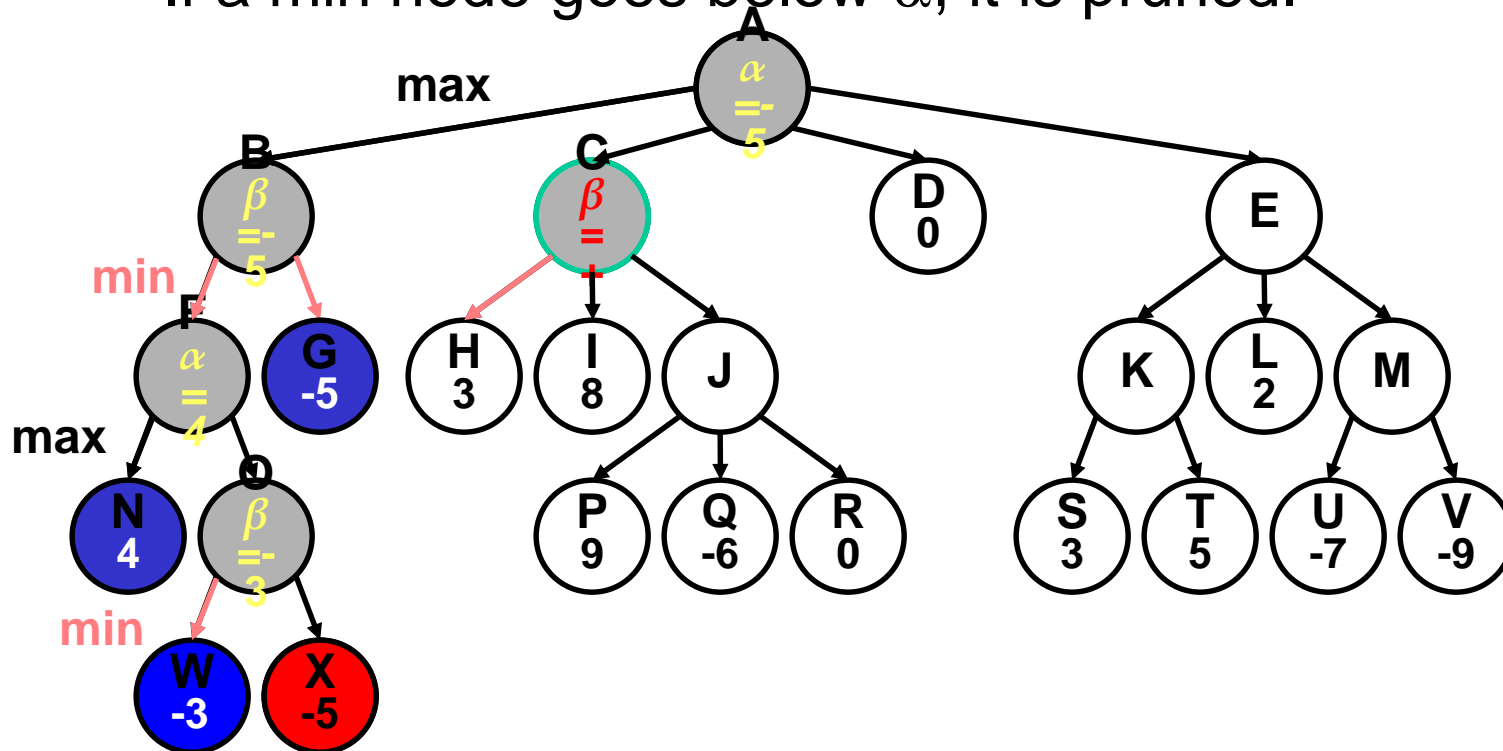
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



# Alpha-beta pruning example

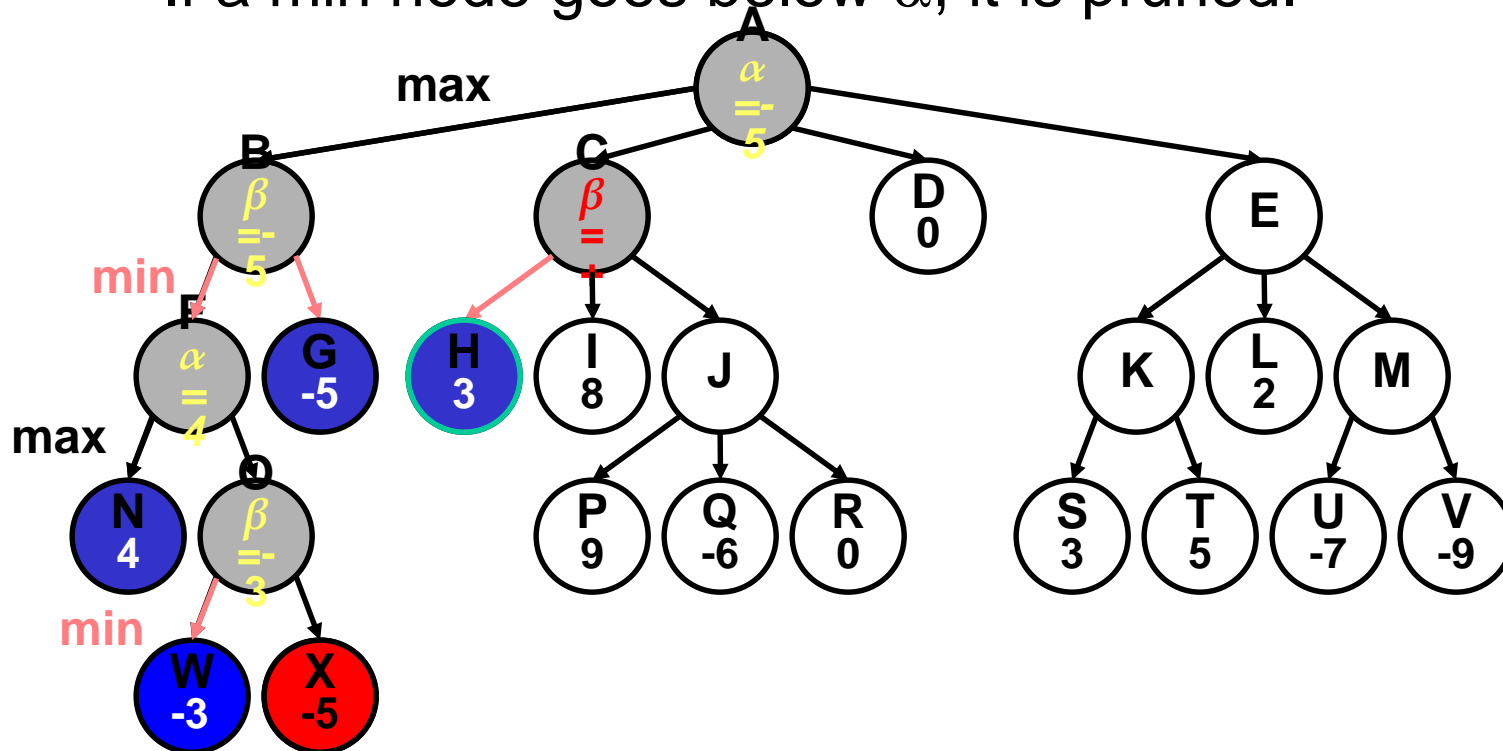
- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.





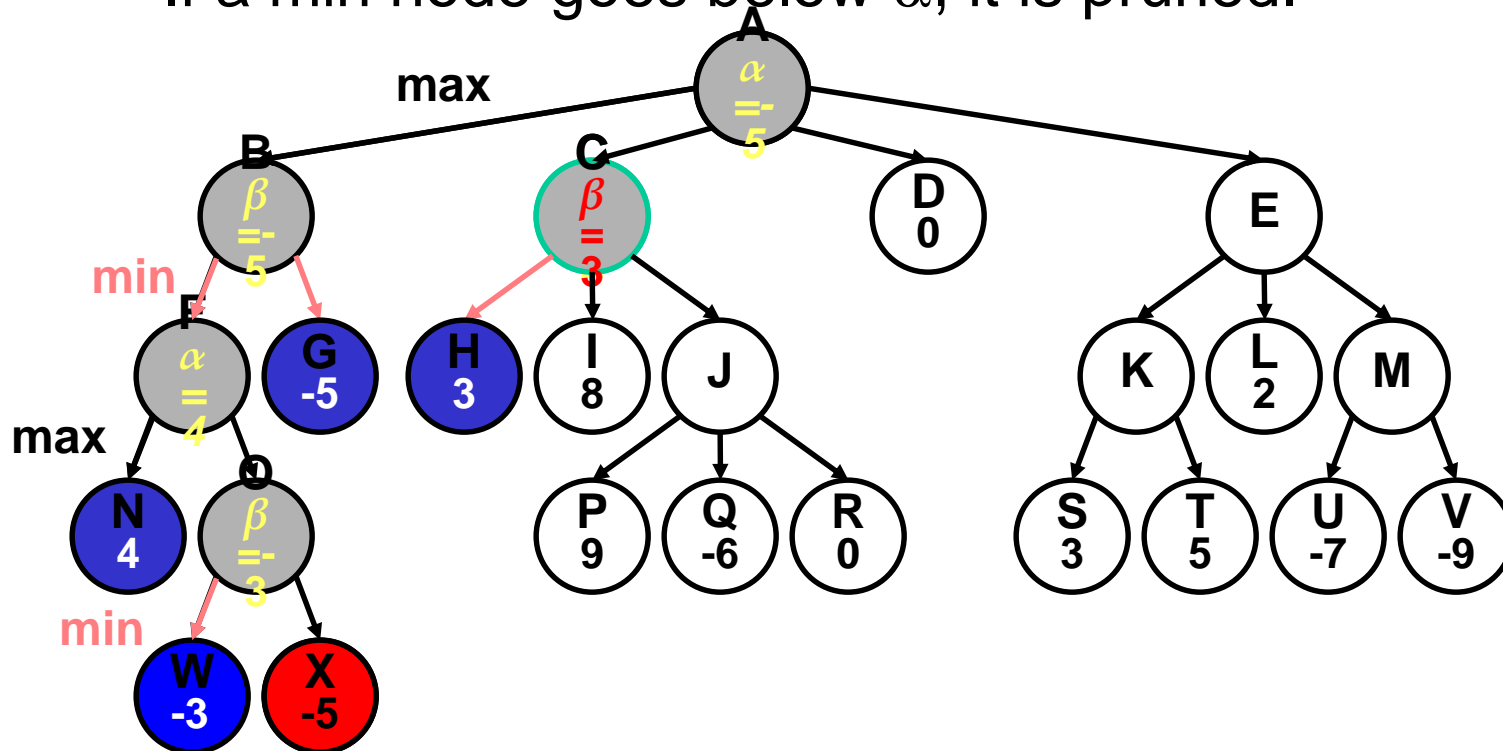
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



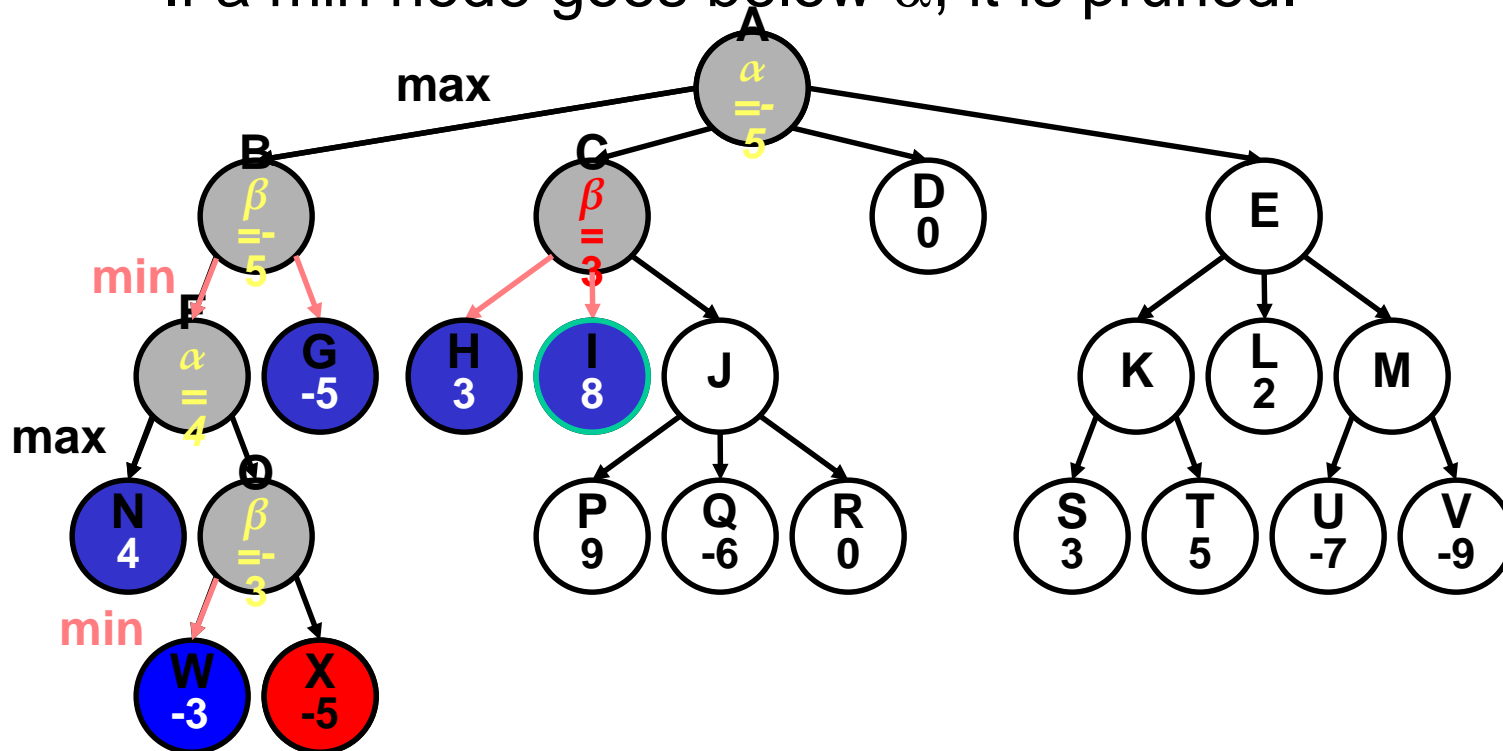
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



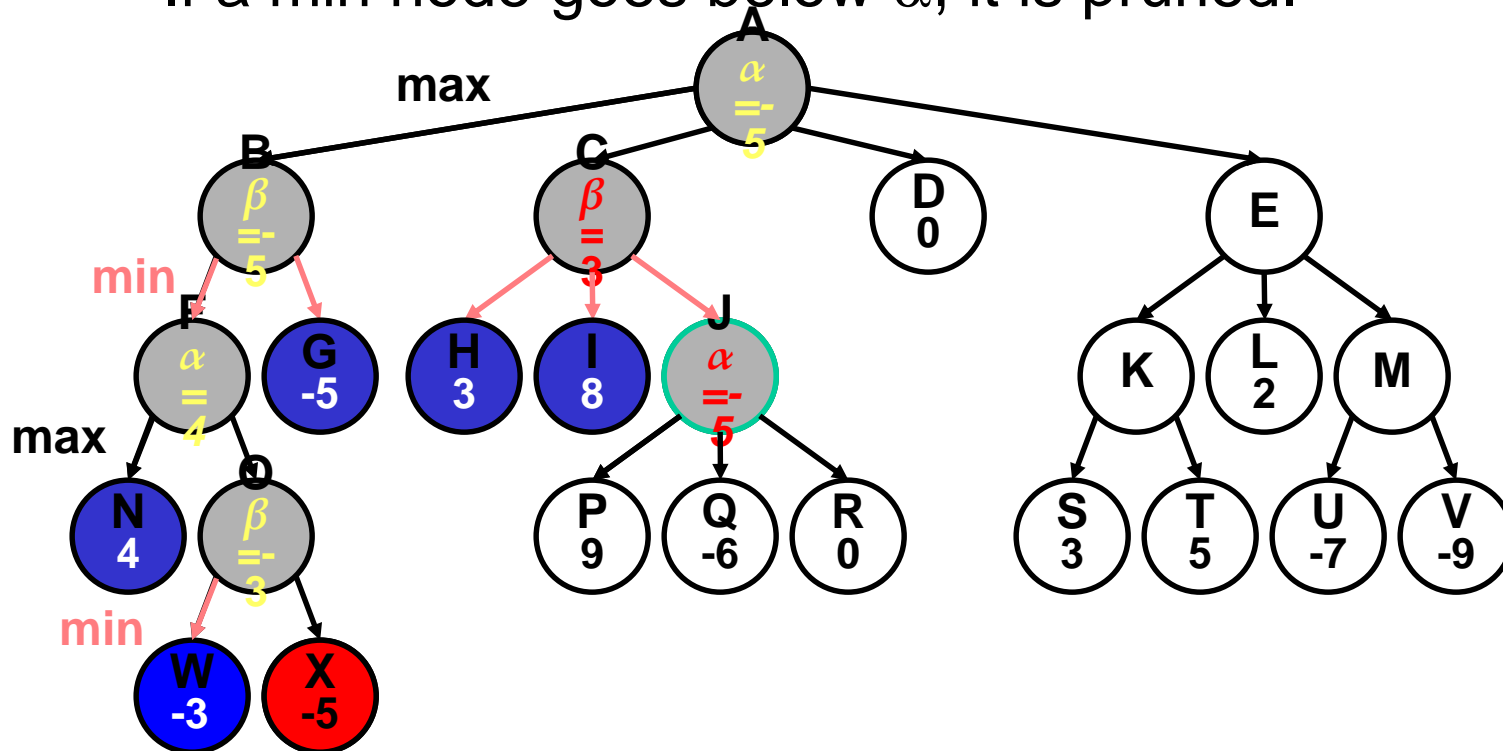
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



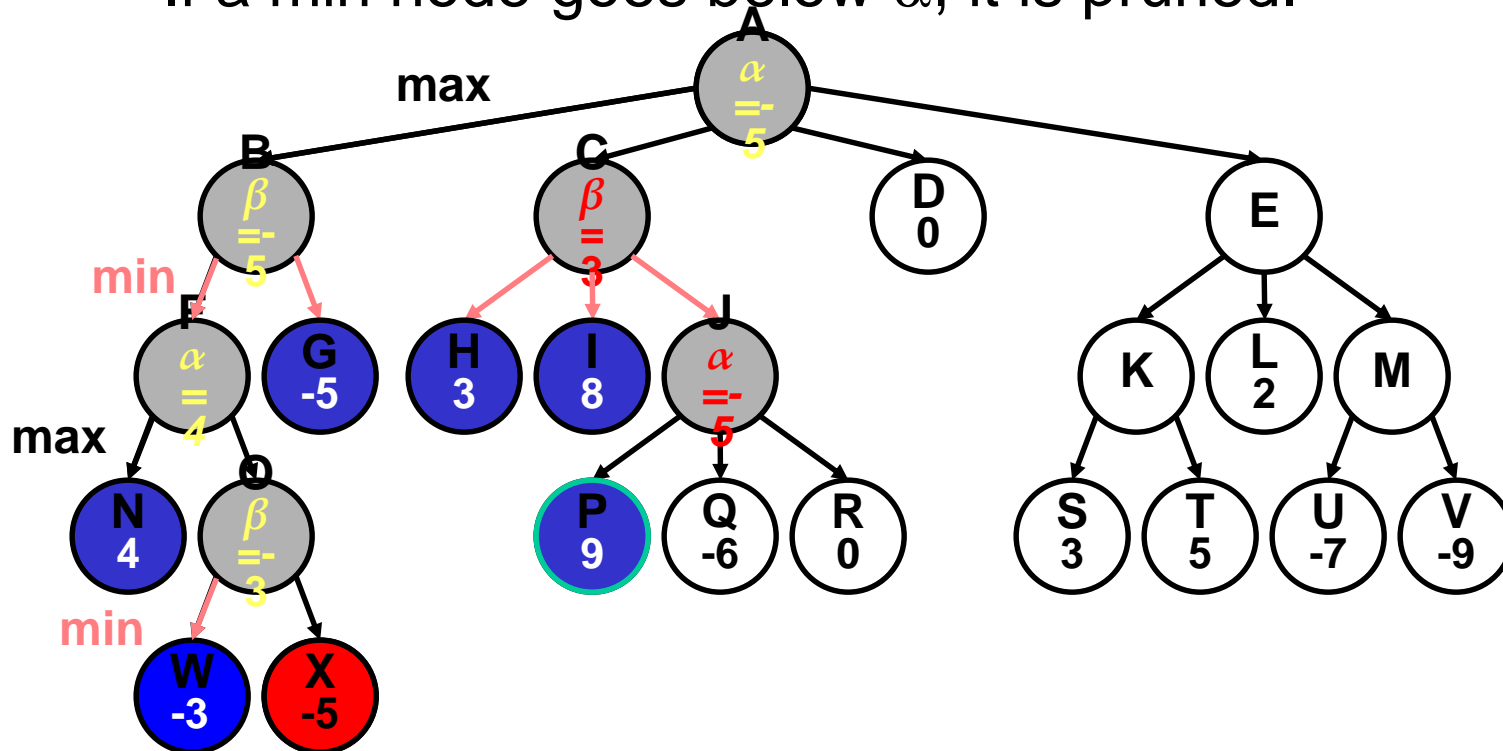
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



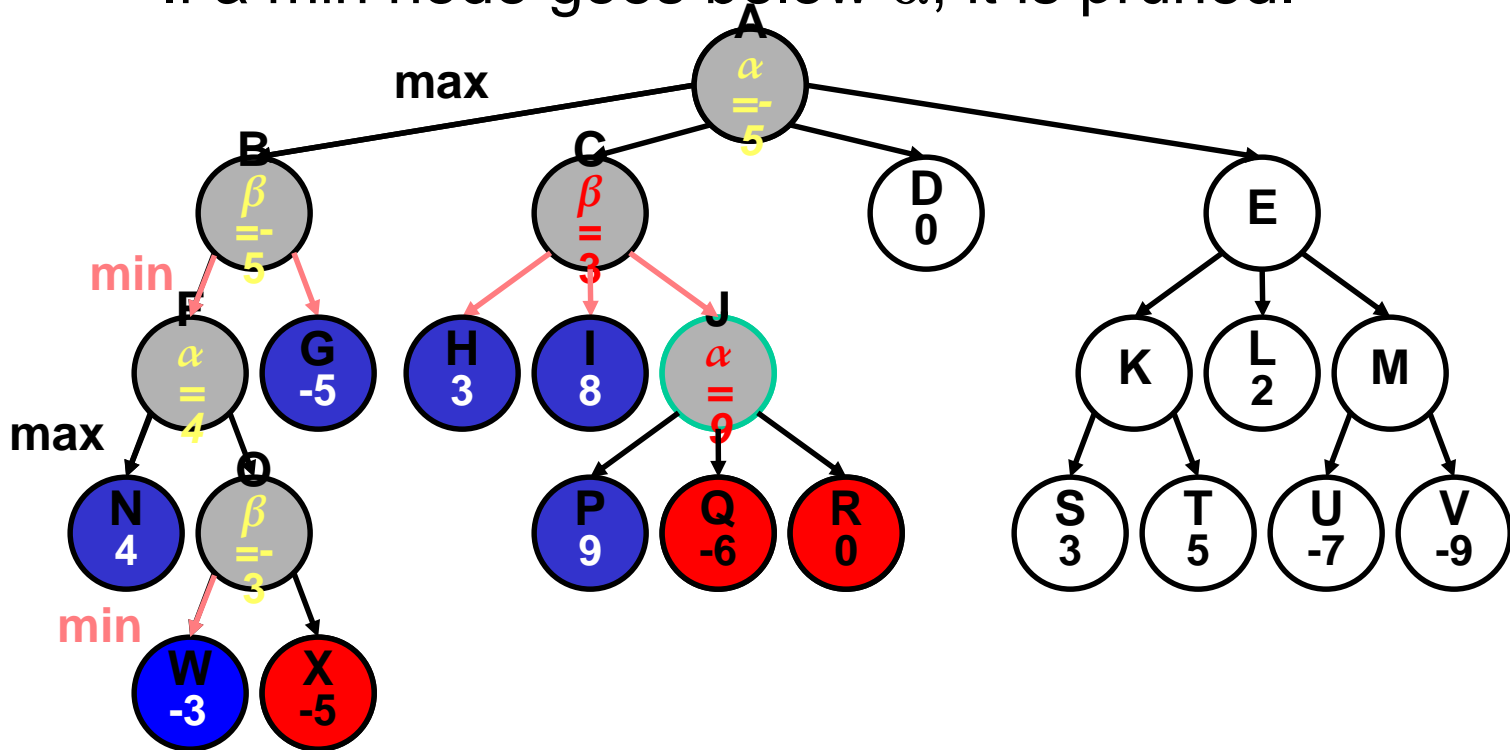
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



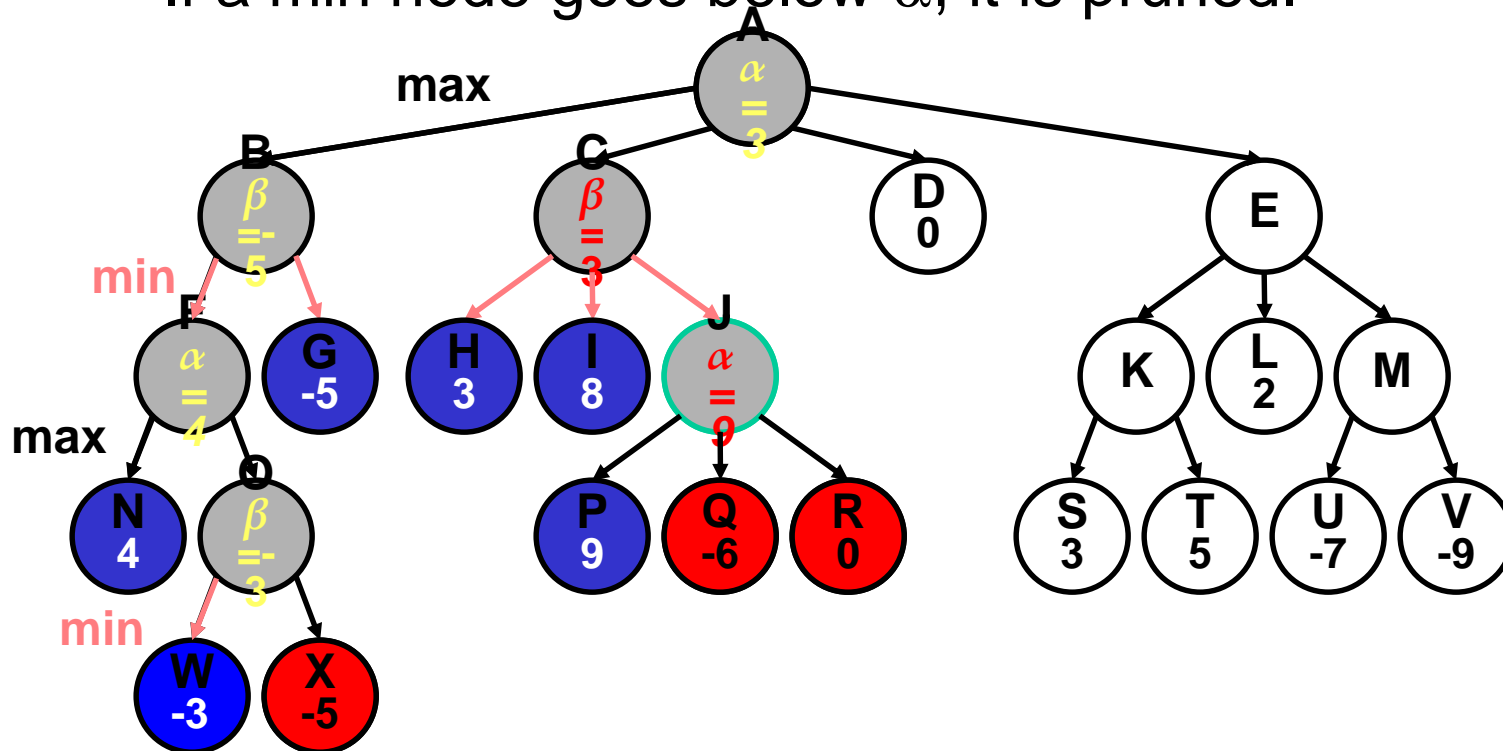
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



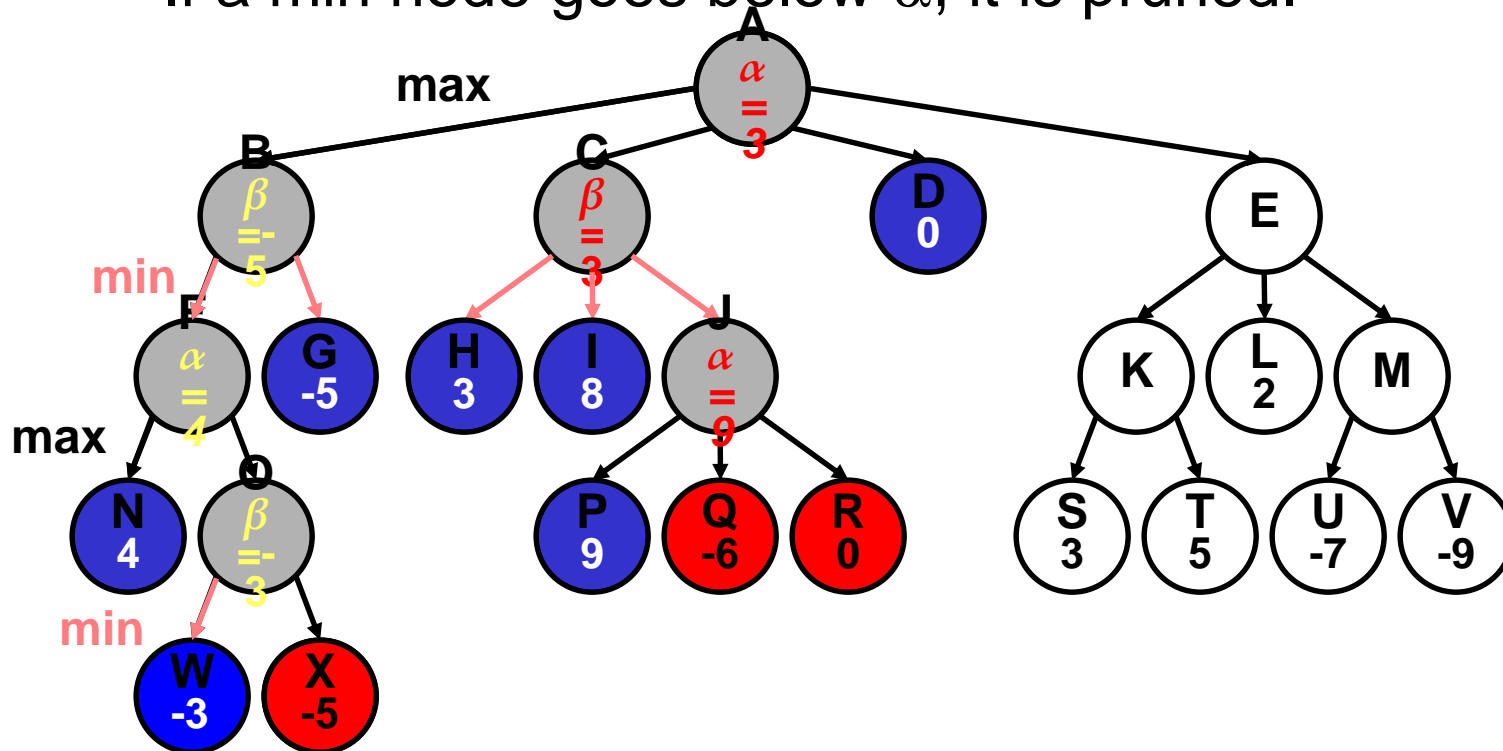
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



# Alpha-beta pruning example

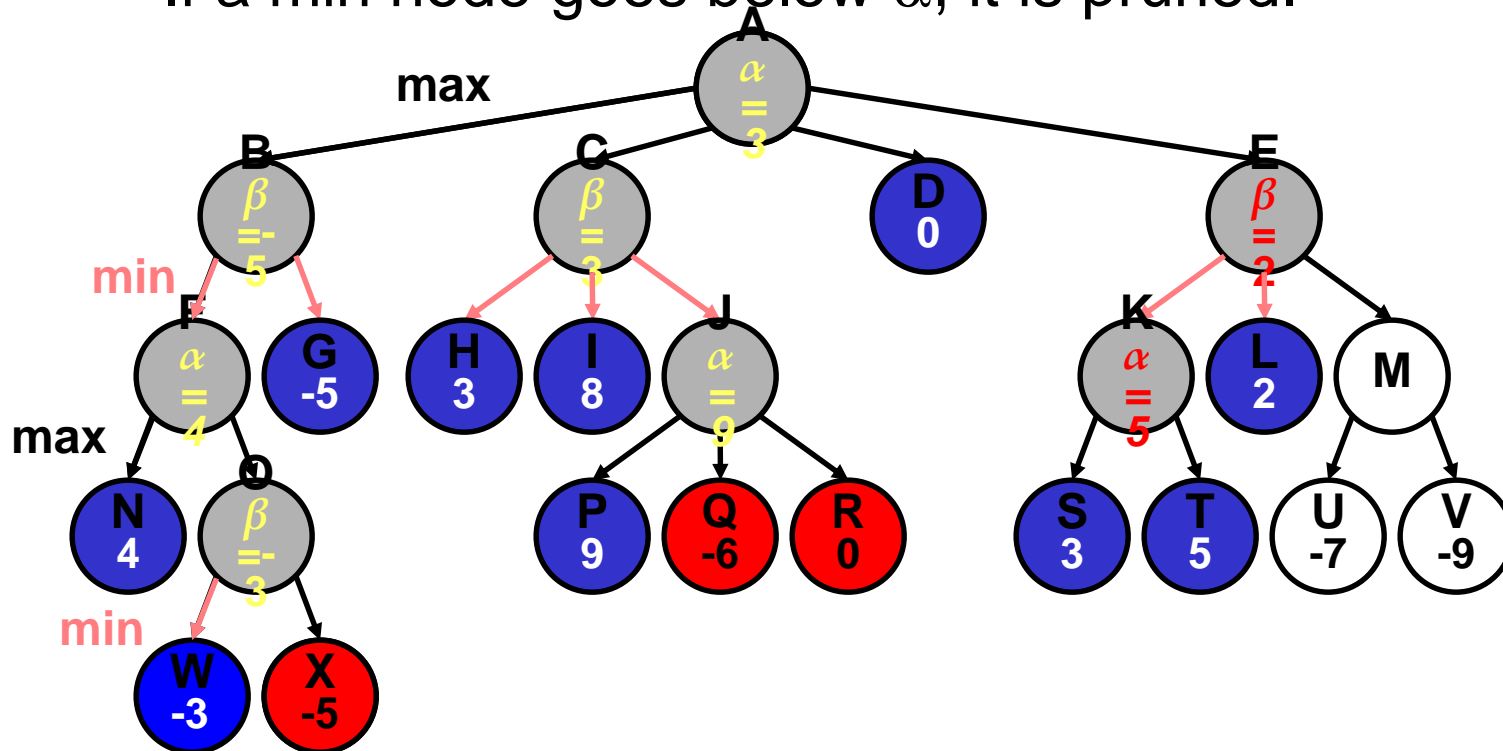
- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.





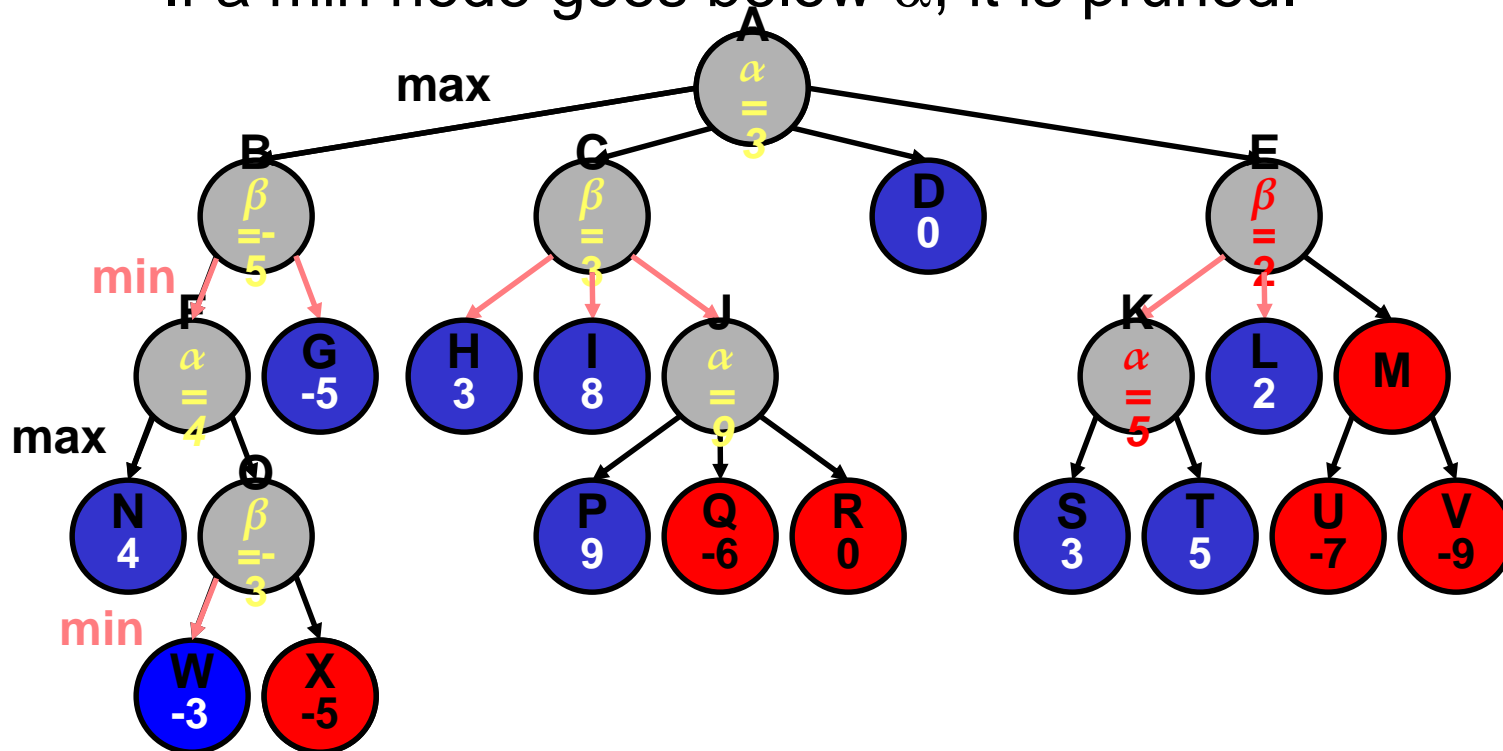
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



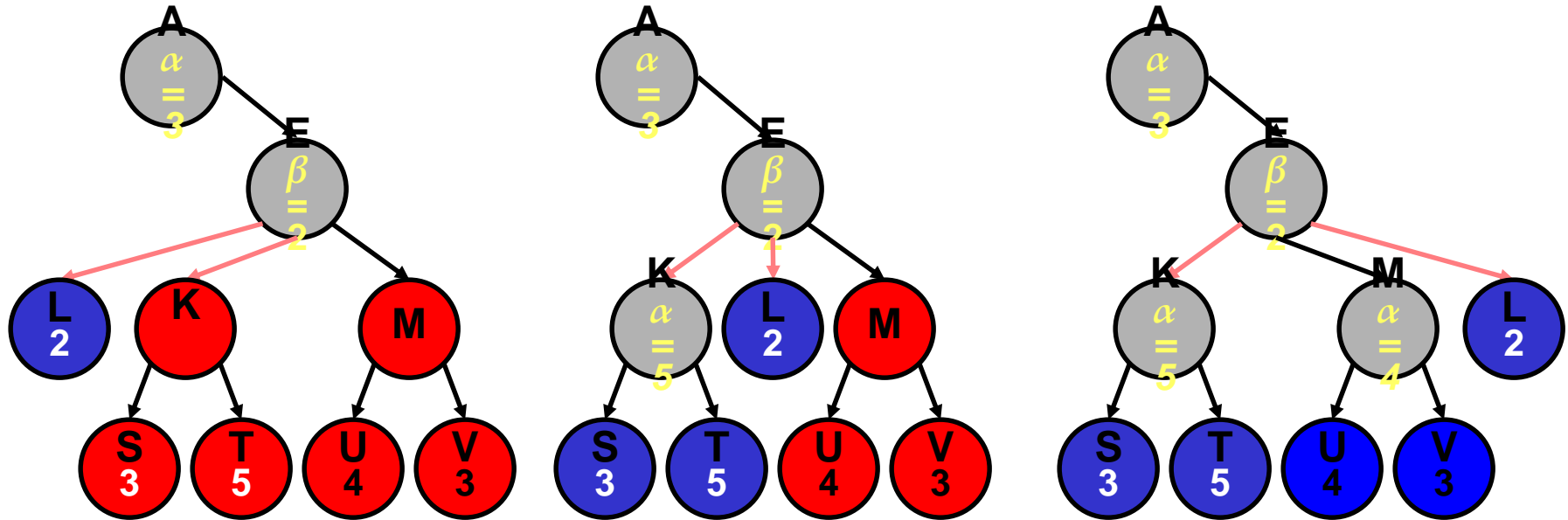
# Alpha-beta pruning example

- Keep two bounds along the path
  - $\alpha$ : the best Max can do on the path
  - $\beta$ : the best (smallest) Min can do on the path
- If a max node exceeds  $\beta$ , it is pruned.
- If a min node goes below  $\alpha$ , it is pruned.



# How effective is alpha-beta pruning?

- Depends on the order of successors!



- In the best case, the number of nodes to search is  $O(b^{m/2})$ , the square root of minimax's cost.
- This occurs when each player's best move is the leftmost child.
- In DeepBlue (IBM Chess), the average branching factor was about 6 with alpha-beta instead of 35-40 without.
- The worst case is no pruning at all.