Adversarial Search II: Alpha-Beta Pruning

CSE 415: Introduction to Artificial Intelligence
University of Washington
Winter, 2020

Ss
State-space Search

Alpha-Beta Pruning

Enhance minimax search with two extra values at each tree node that represent the interval in which the "solution" value must lie.

$$[\alpha, \beta]$$

Initialize the root's to $[-\infty, \infty]$.
Update these at the current node, when possible.
If any node gets $\alpha \geq \beta$, then it is "finished", so "prune off" any of its children that remain.

Ss
State-space Search

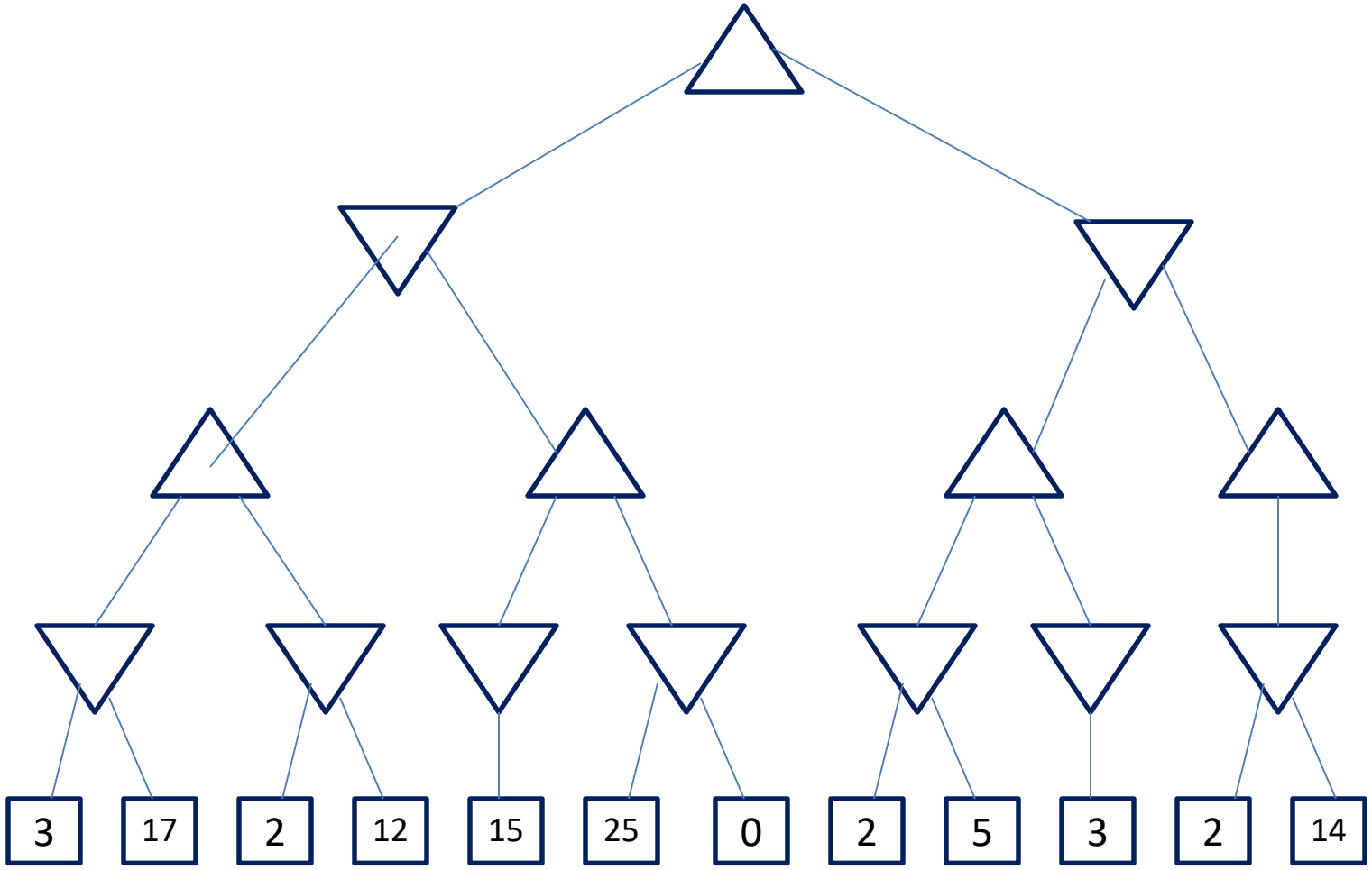
Alpha-Beta Cutoffs

An alpha (beta) cutoff occurs at a Maximizing (minimizing) node when it is known that the maximizing (minimizing) player has a move that results in a value alpha (beta) and, subsequently, when an alternative to that move is explored, it is found that the alternative gives the opponent the option of moving to a lower (higher) valued position.

Any further exploration of the alternative can be canceled.

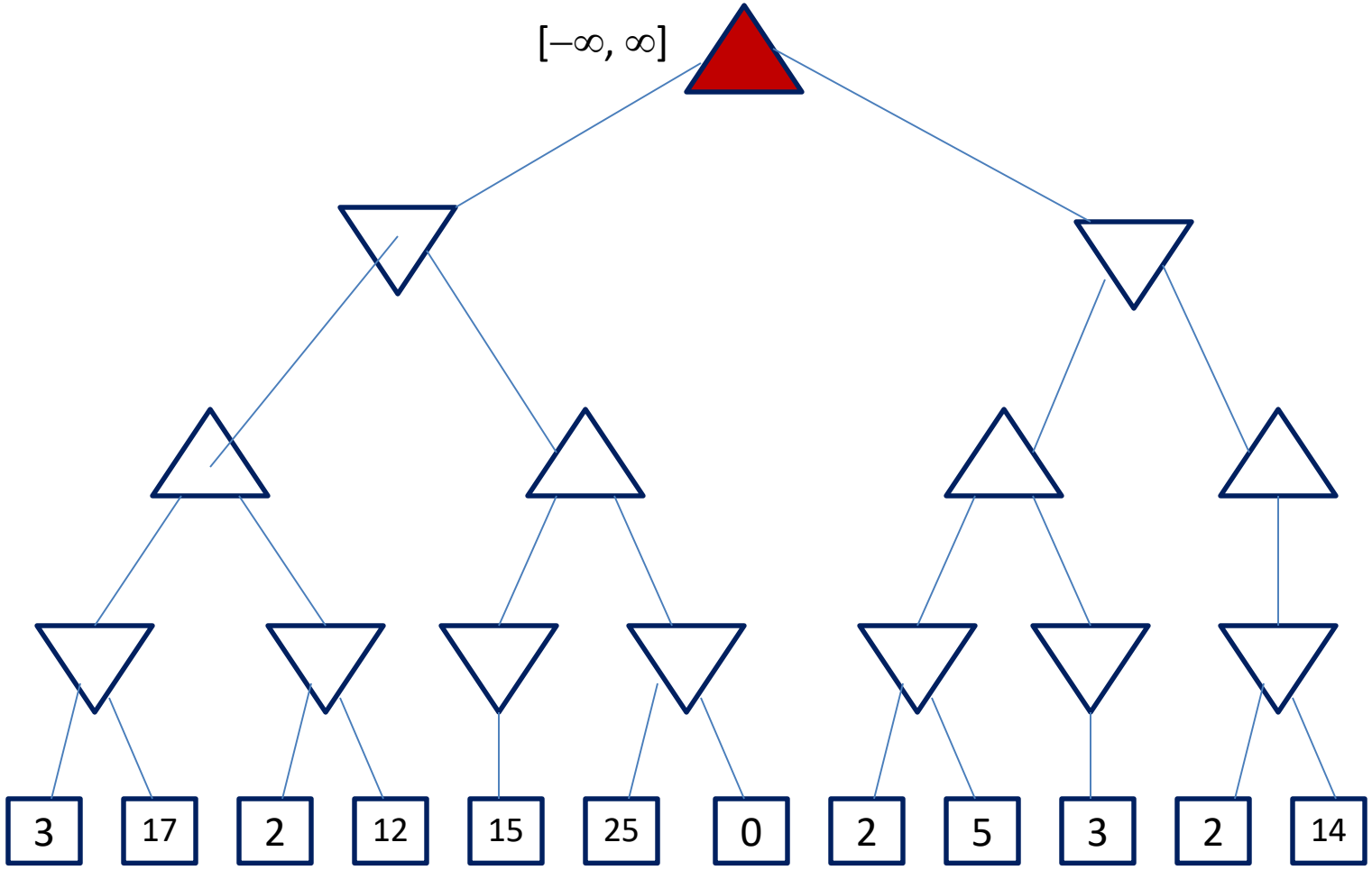
Ss
State-space Search

Alpha-Beta Pruning



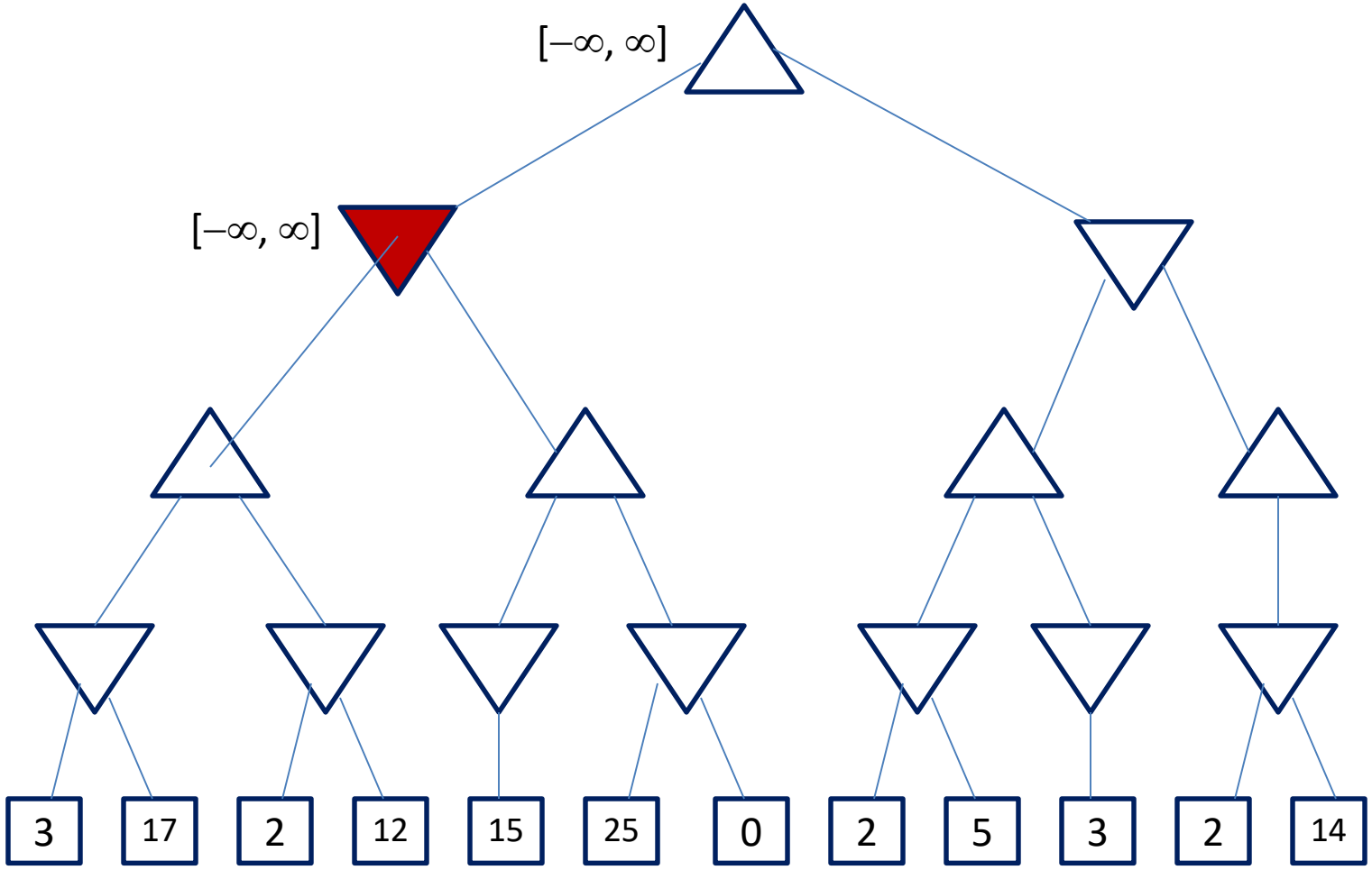
Ss
State-space Search

Alpha-Beta Pruning



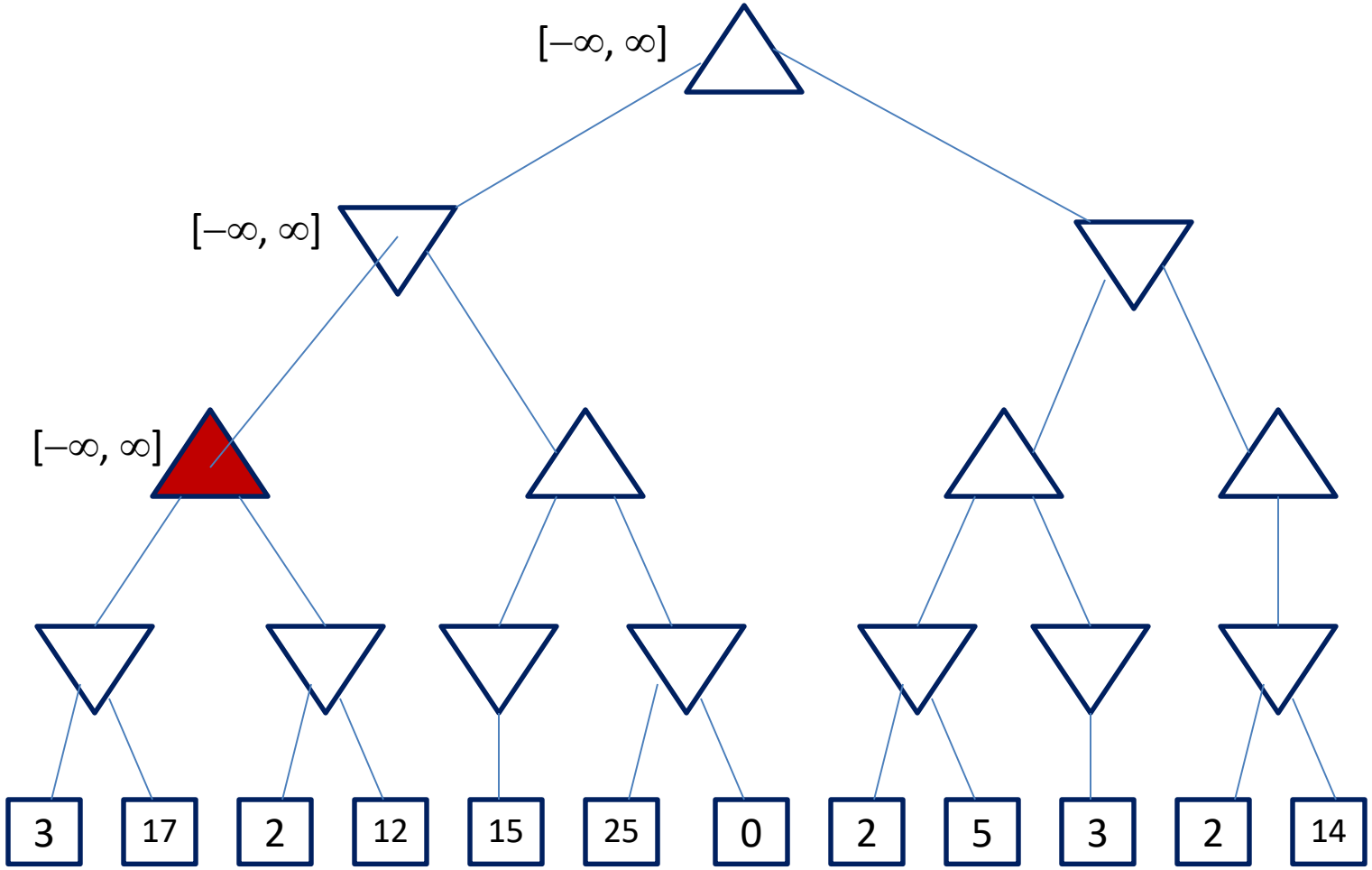
Ss
State-space Search

Alpha-Beta Pruning



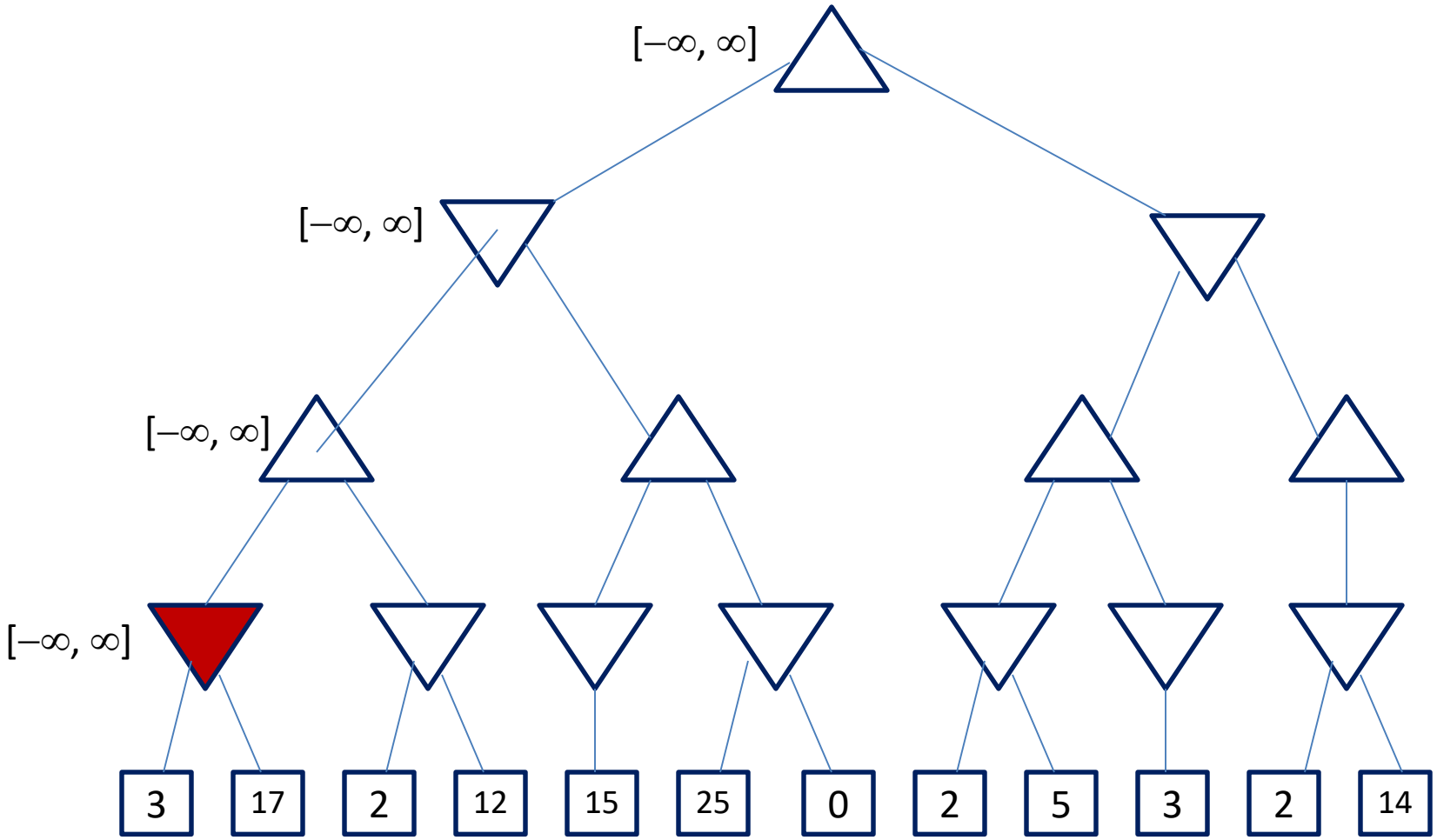
Ss
State-space Search

Alpha-Beta Pruning



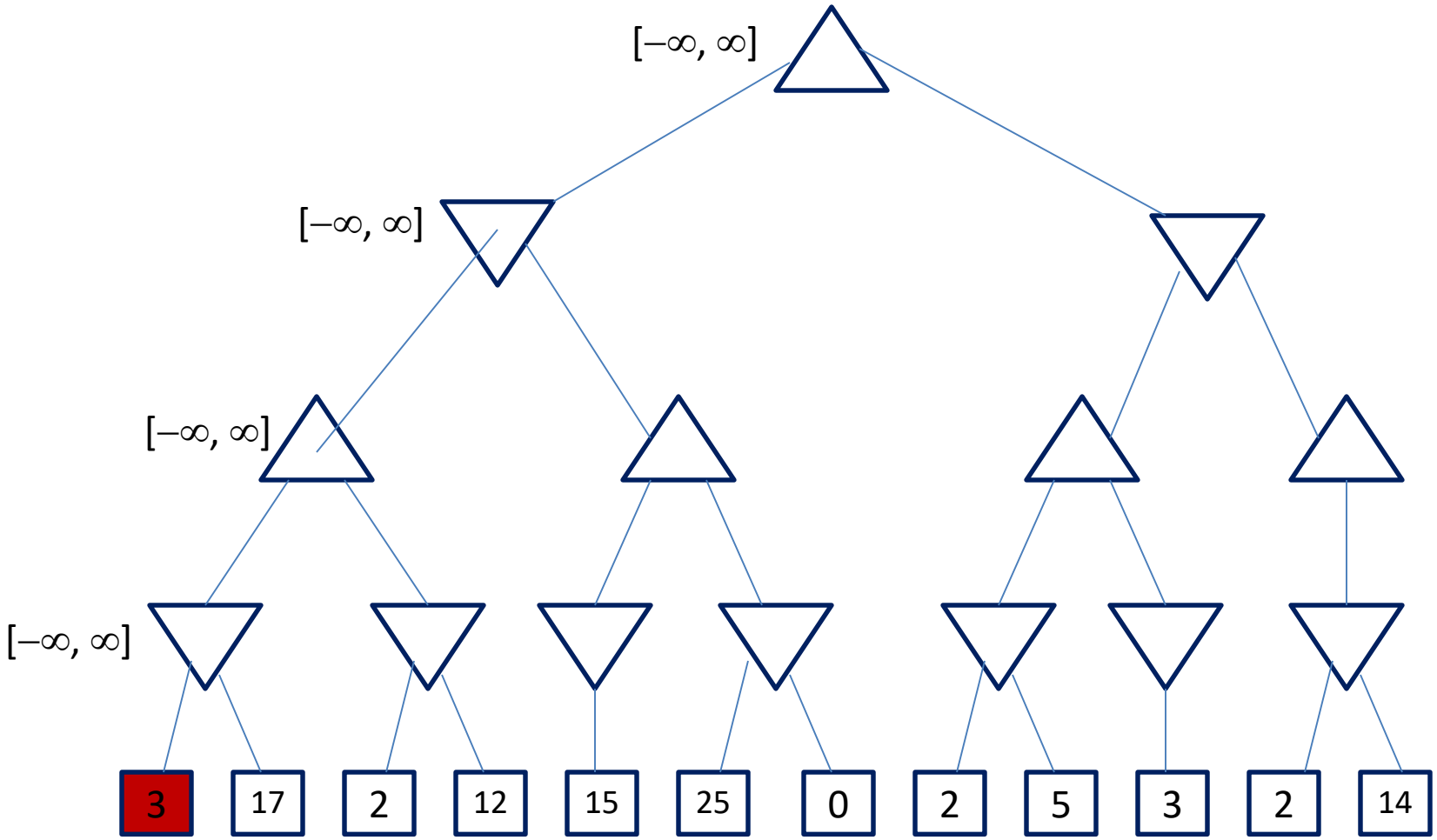
Ss
State-space Search

Alpha-Beta Pruning



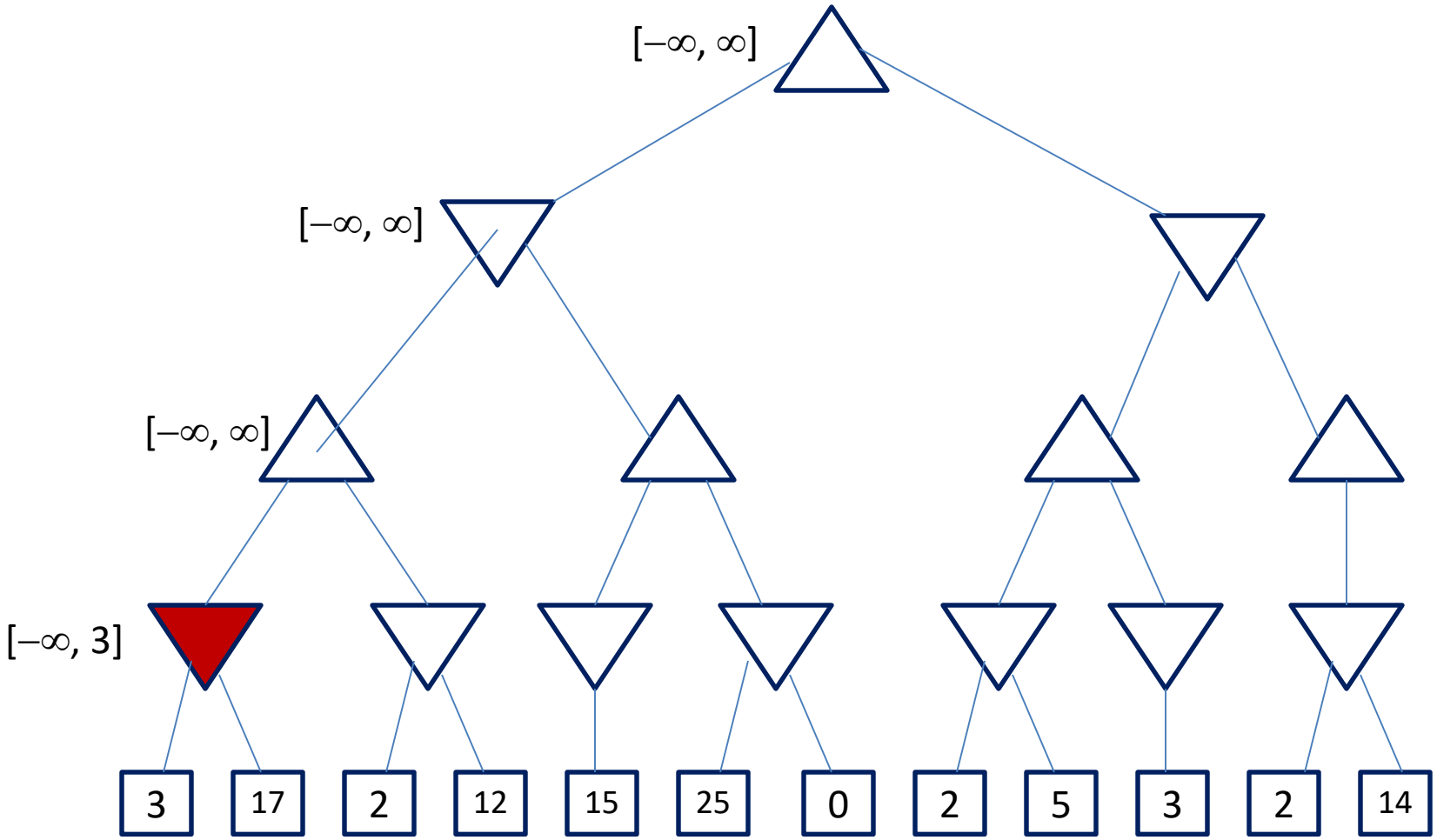
Ss
State-space Search

Alpha-Beta Pruning



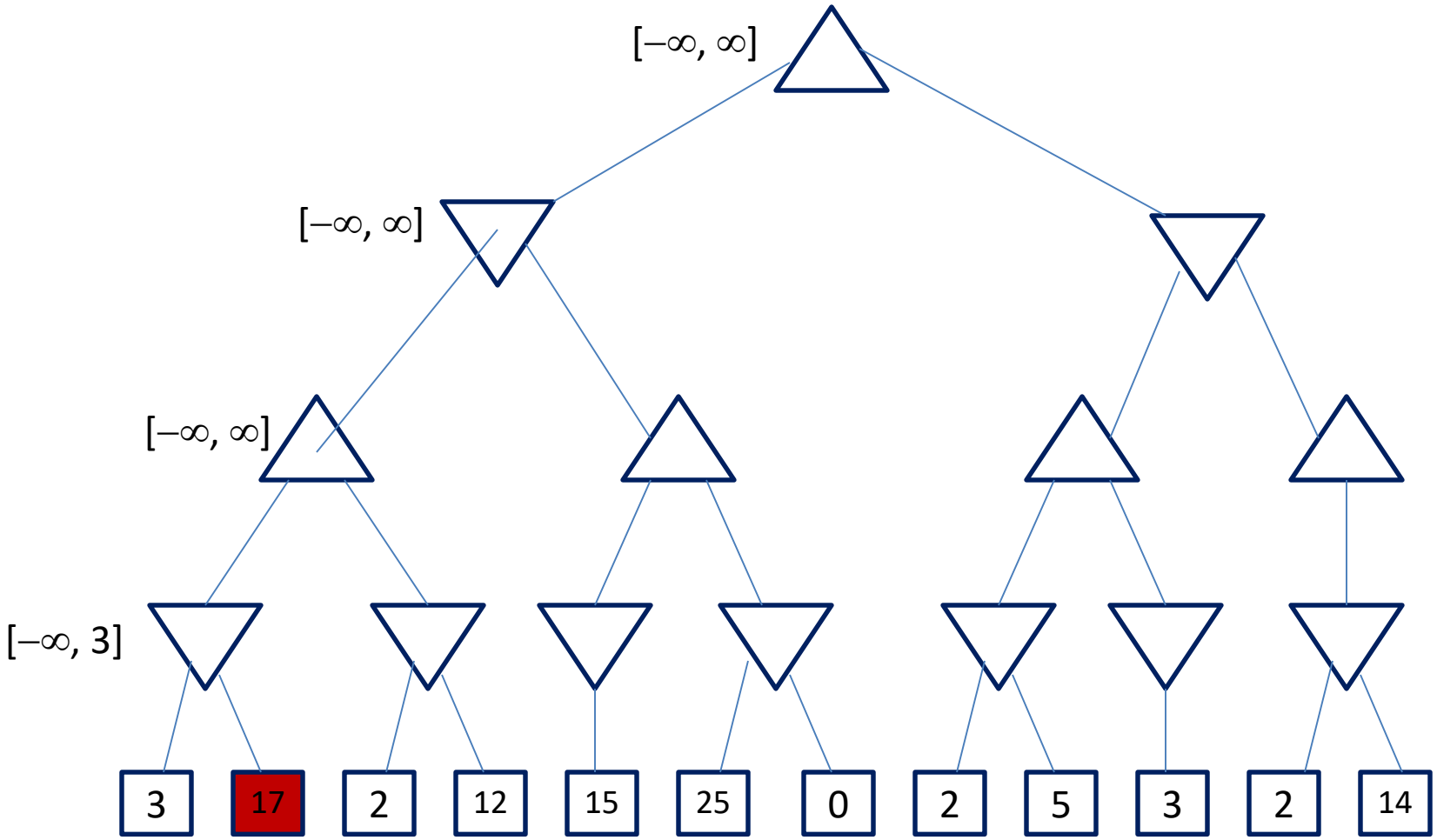
Ss
State-space Search

Alpha-Beta Pruning



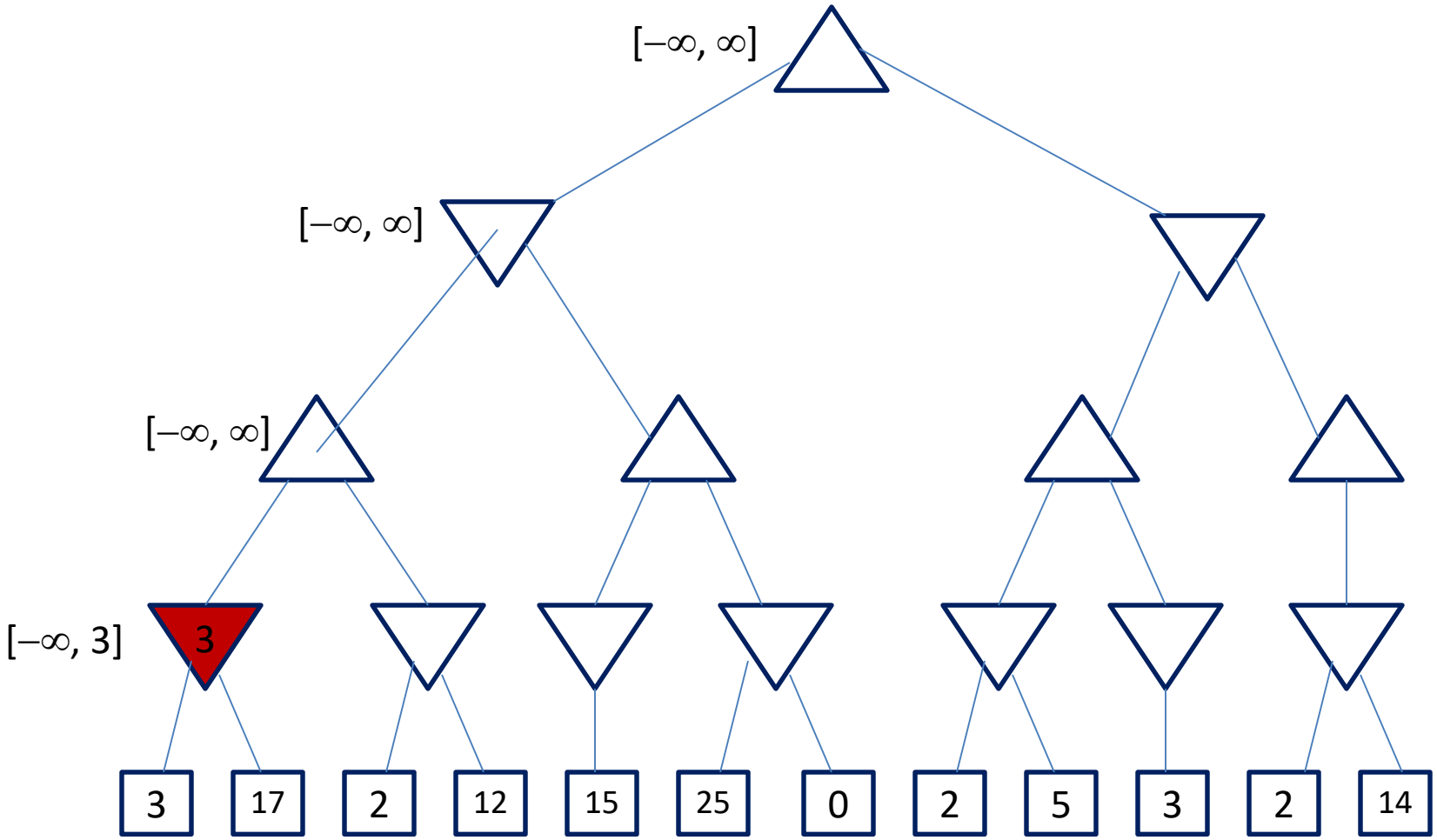
Ss
State-space Search

Alpha-Beta Pruning



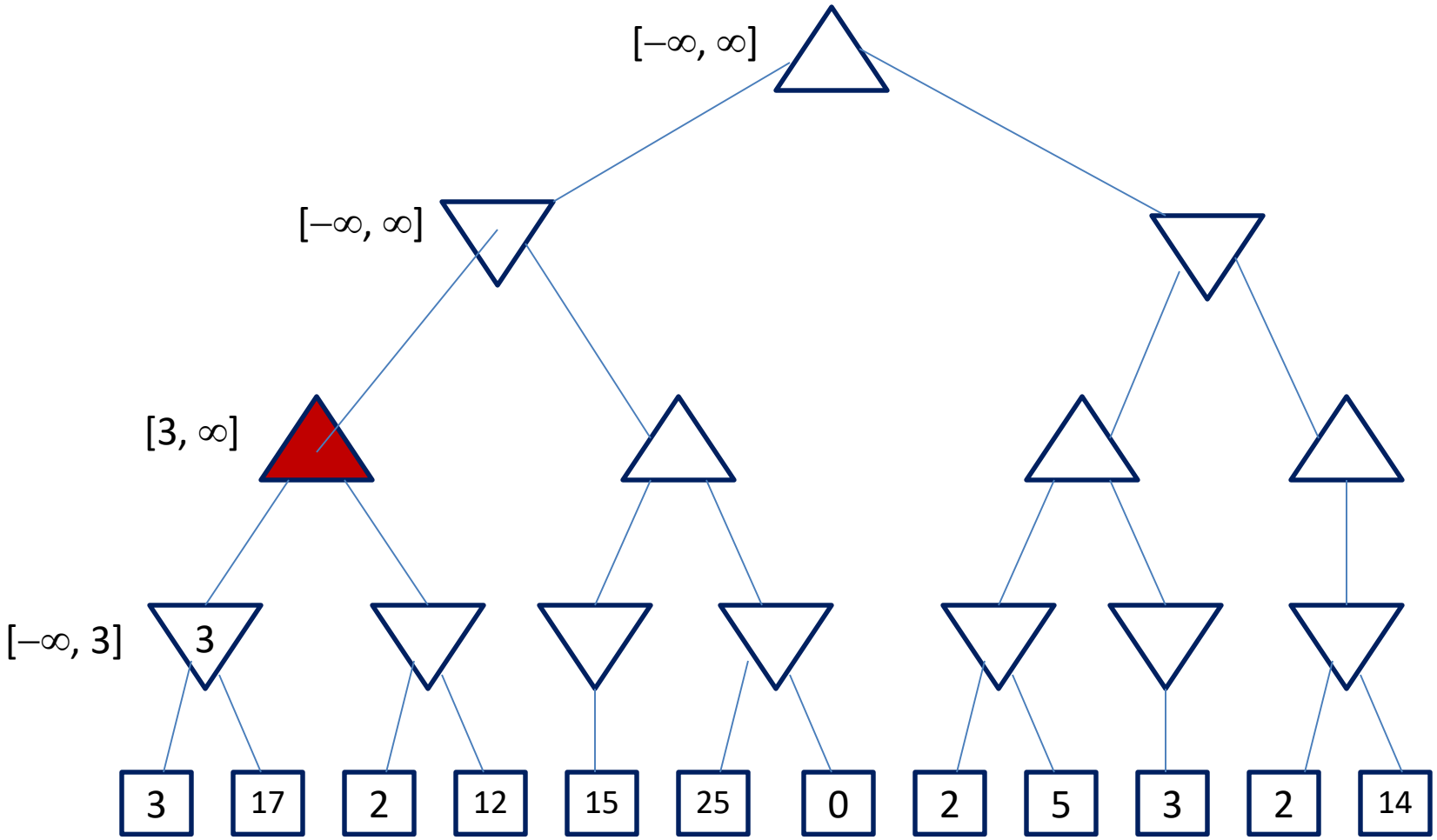
Ss
State-space Search

Alpha-Beta Pruning



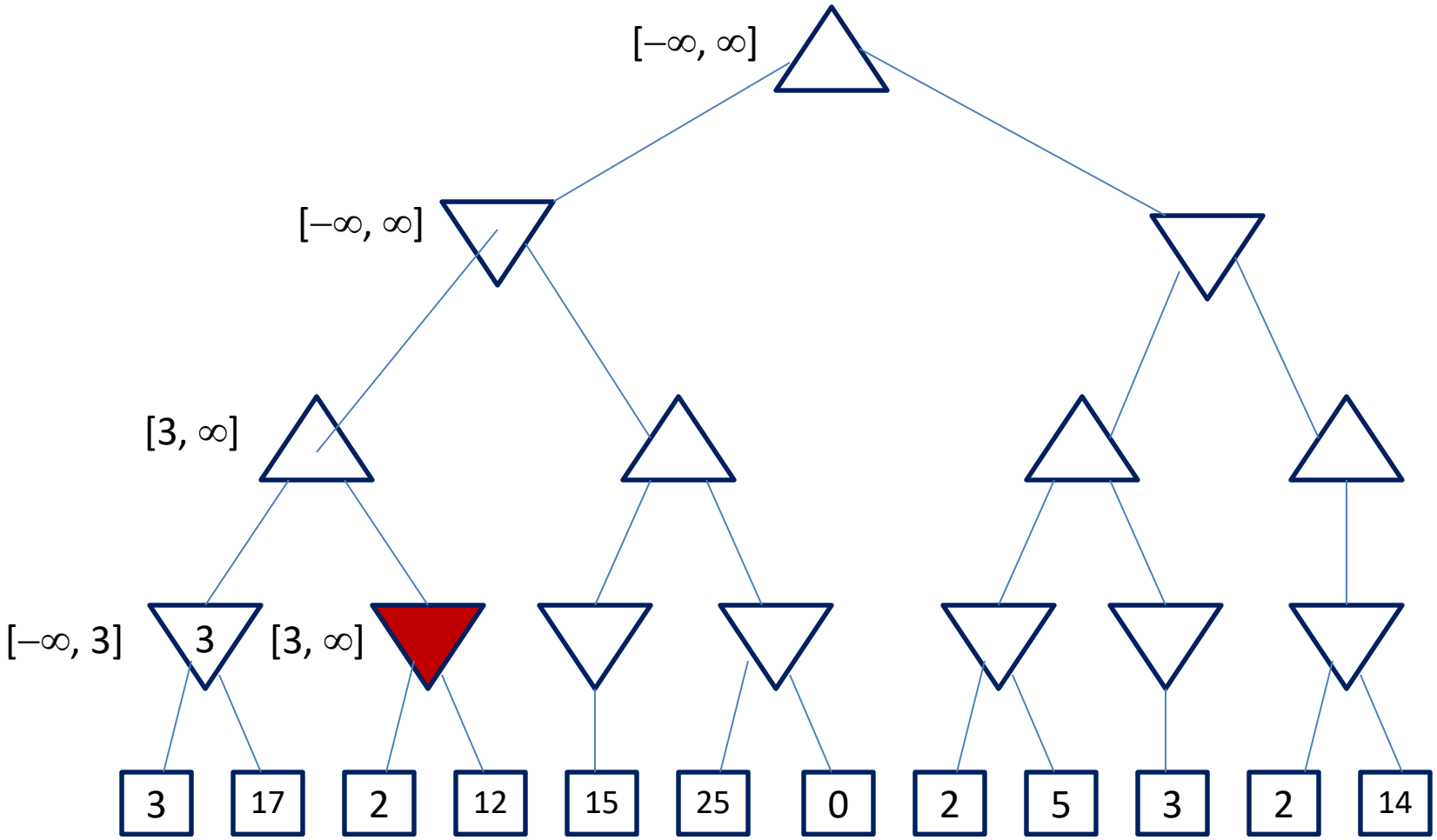
Ss
State-space Search

Alpha-Beta Pruning



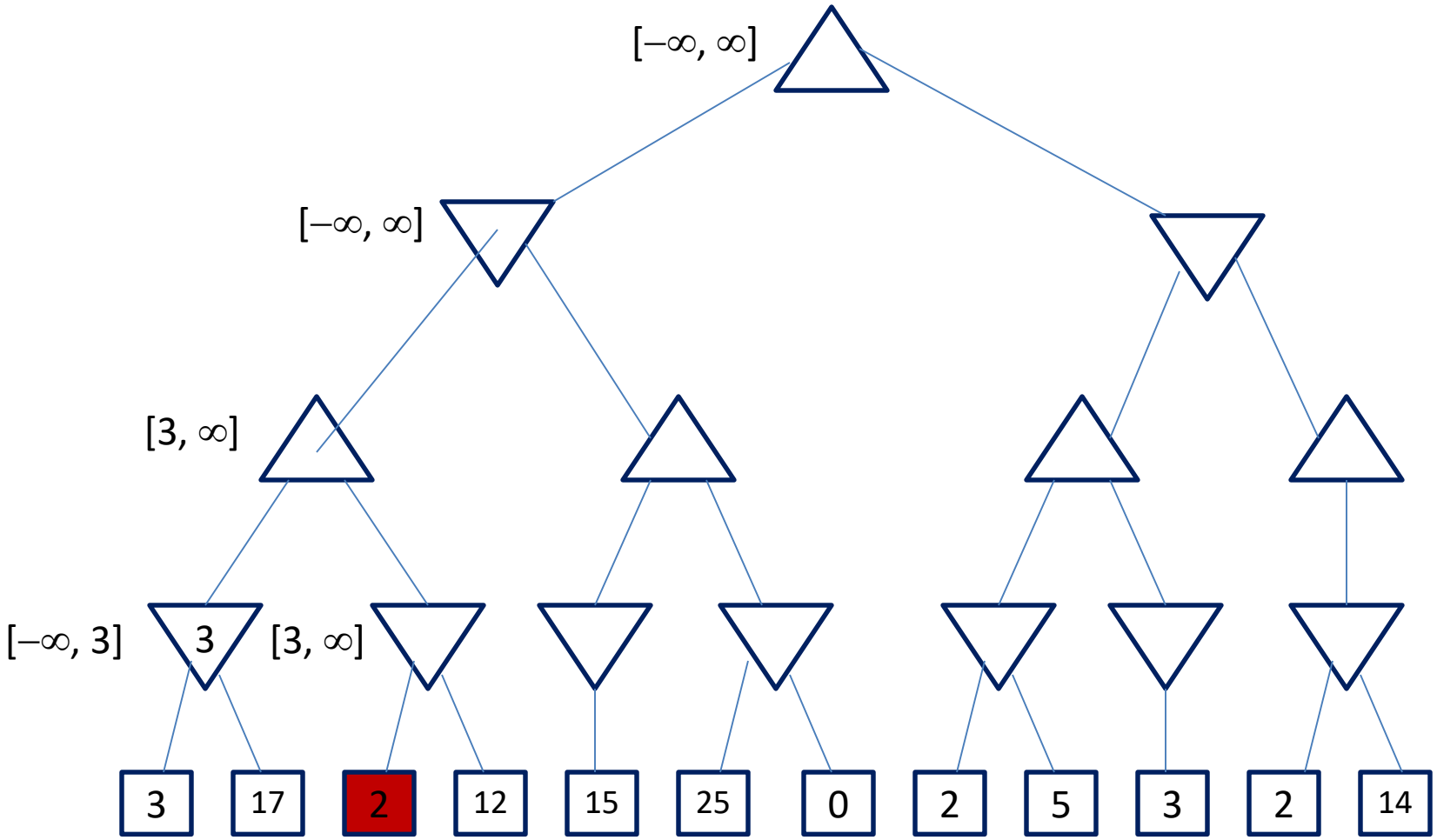
Ss
State-space Search

Alpha-Beta Pruning



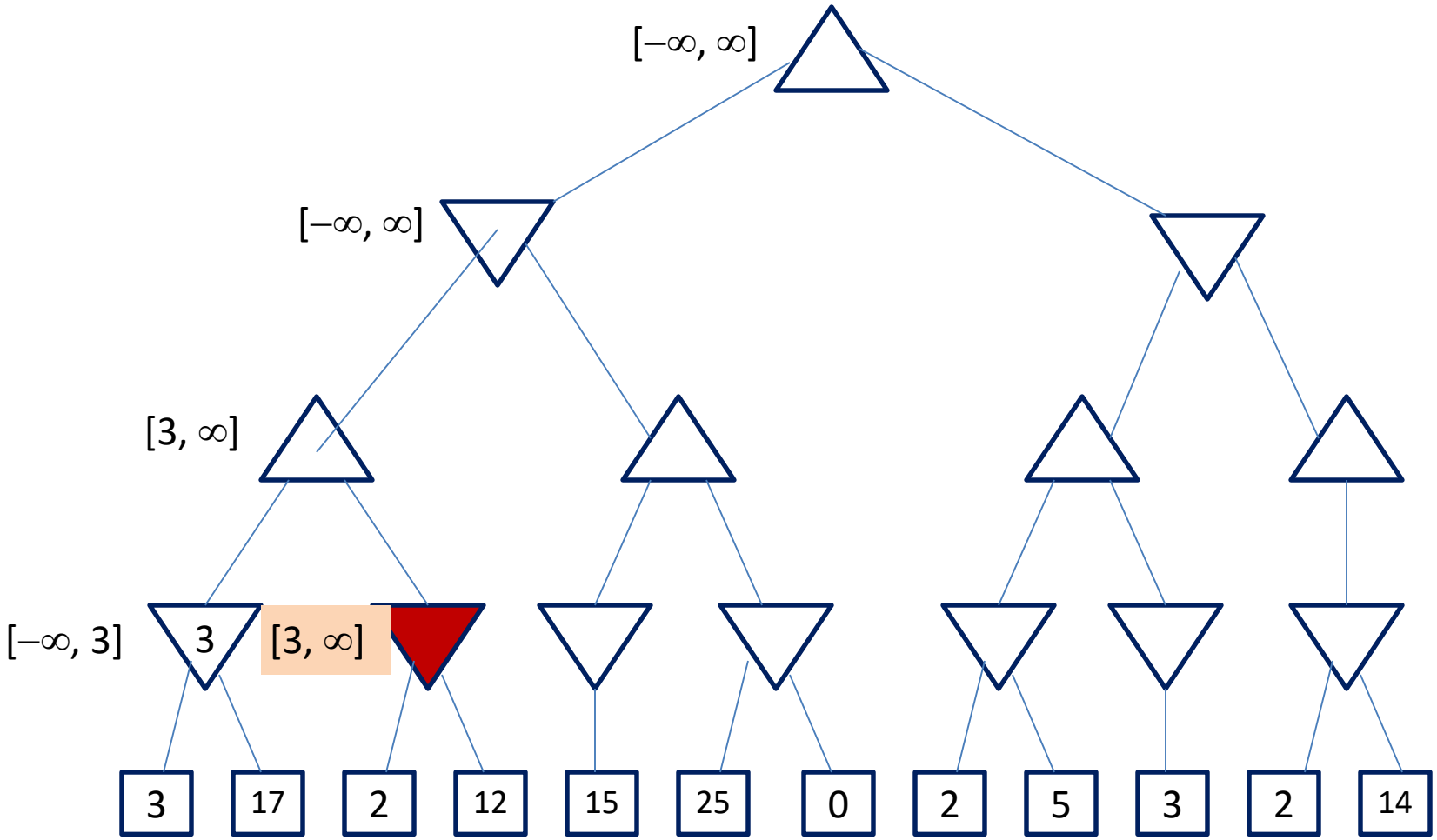
Ss
State-space Search

Alpha-Beta Pruning



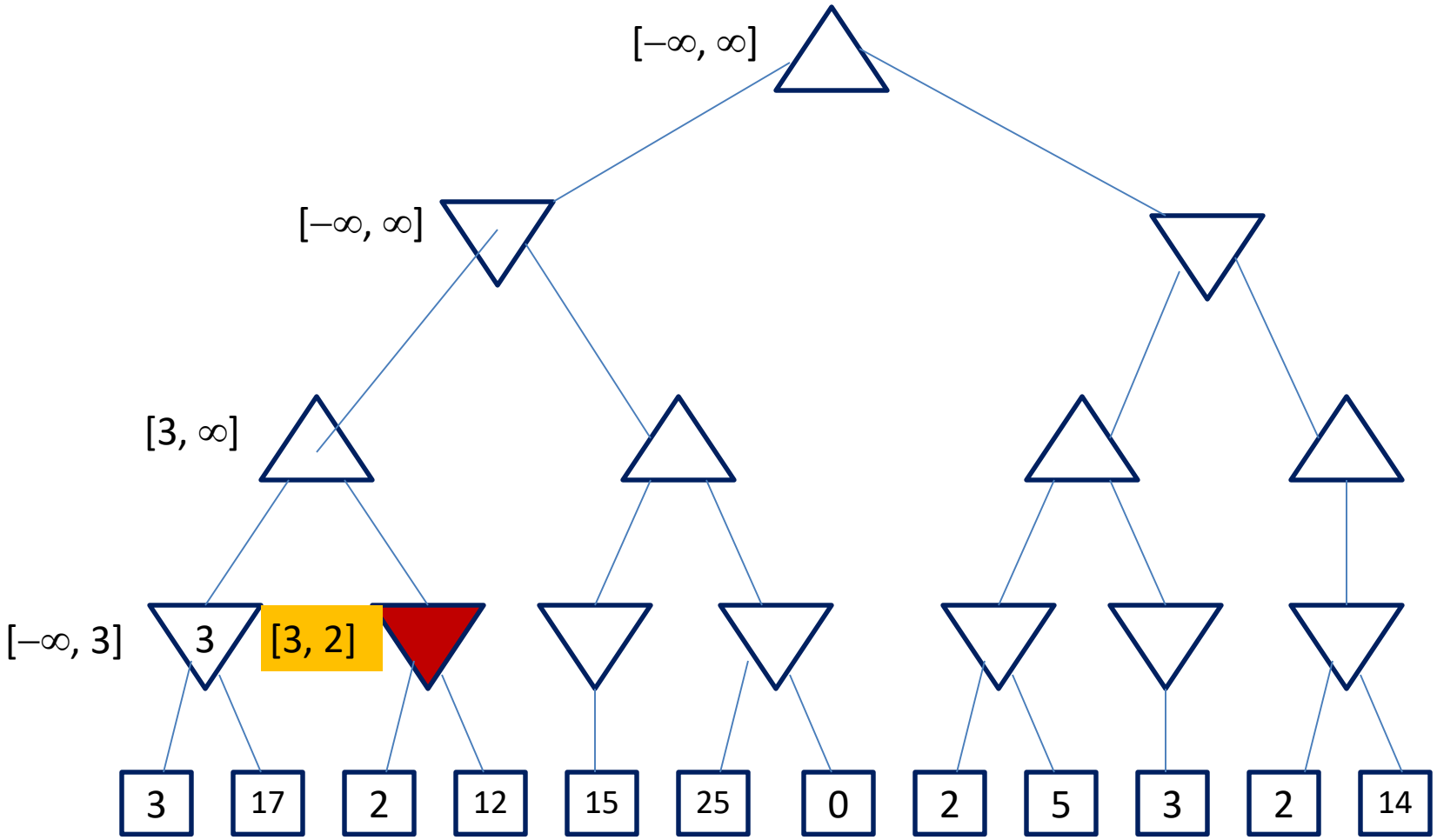
Ss
State-space Search

Alpha-Beta Pruning



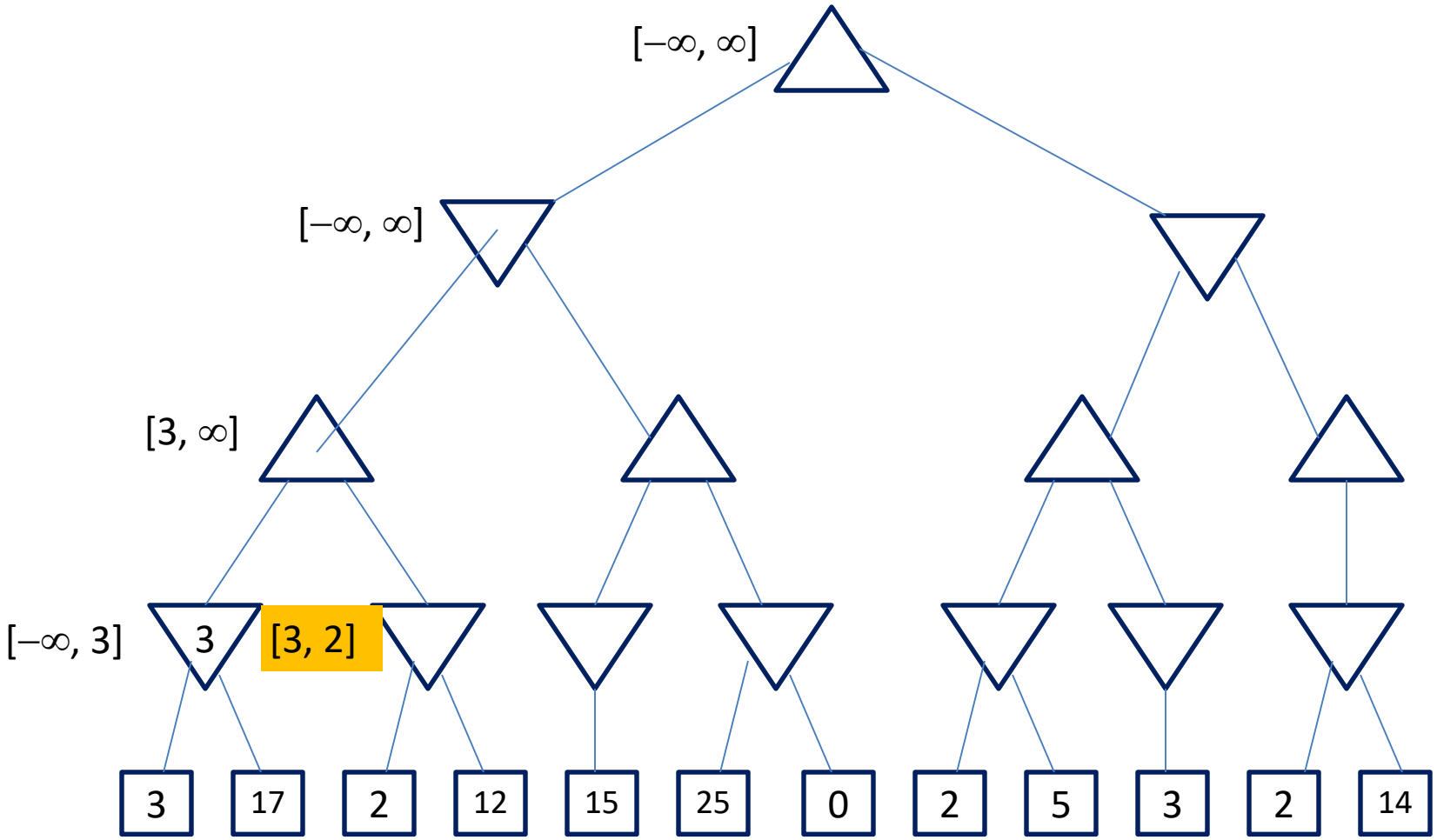
Ss
State-space Search

Alpha-Beta Pruning



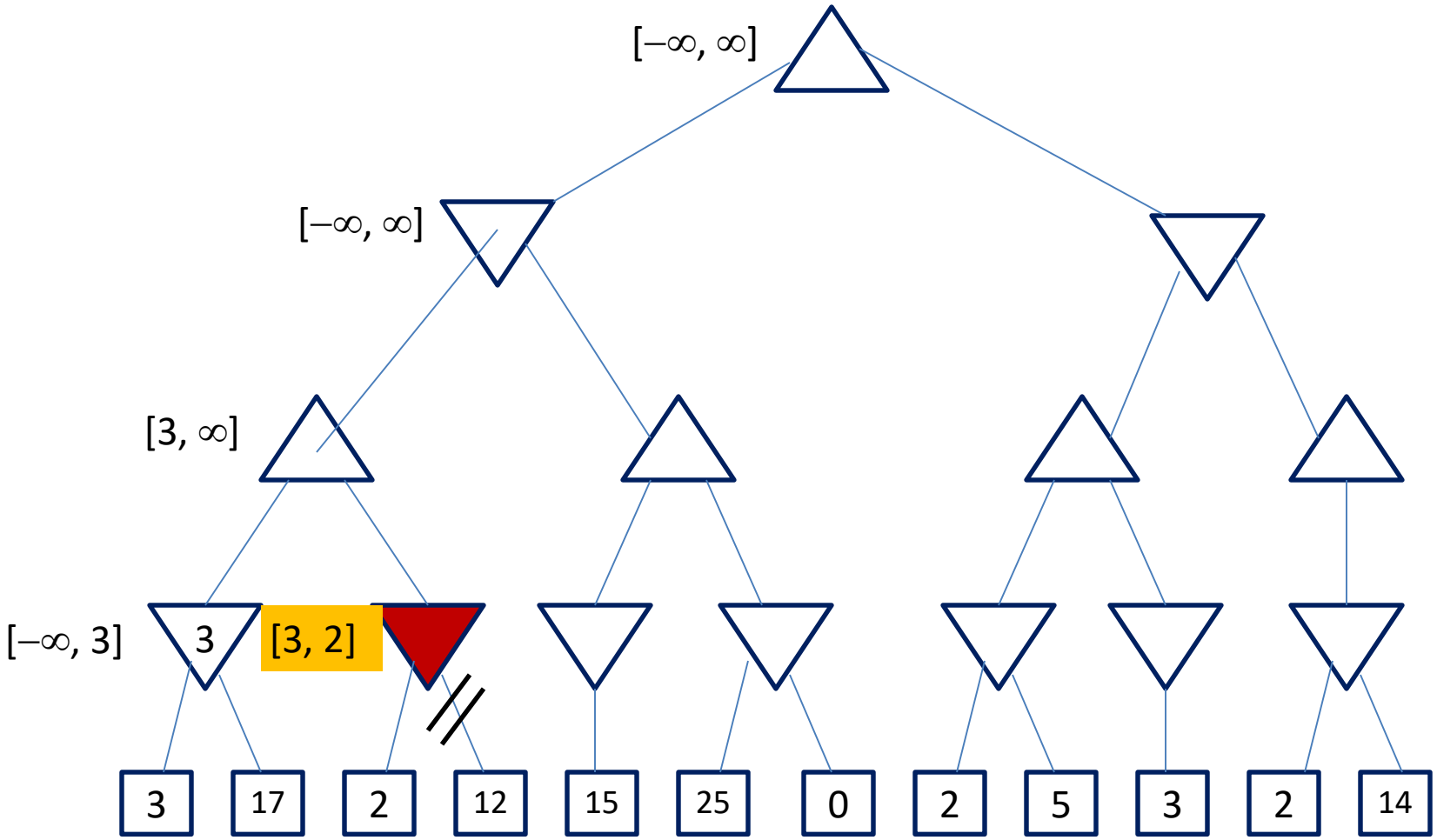
Ss
State-space Search

Alpha-Beta Pruning



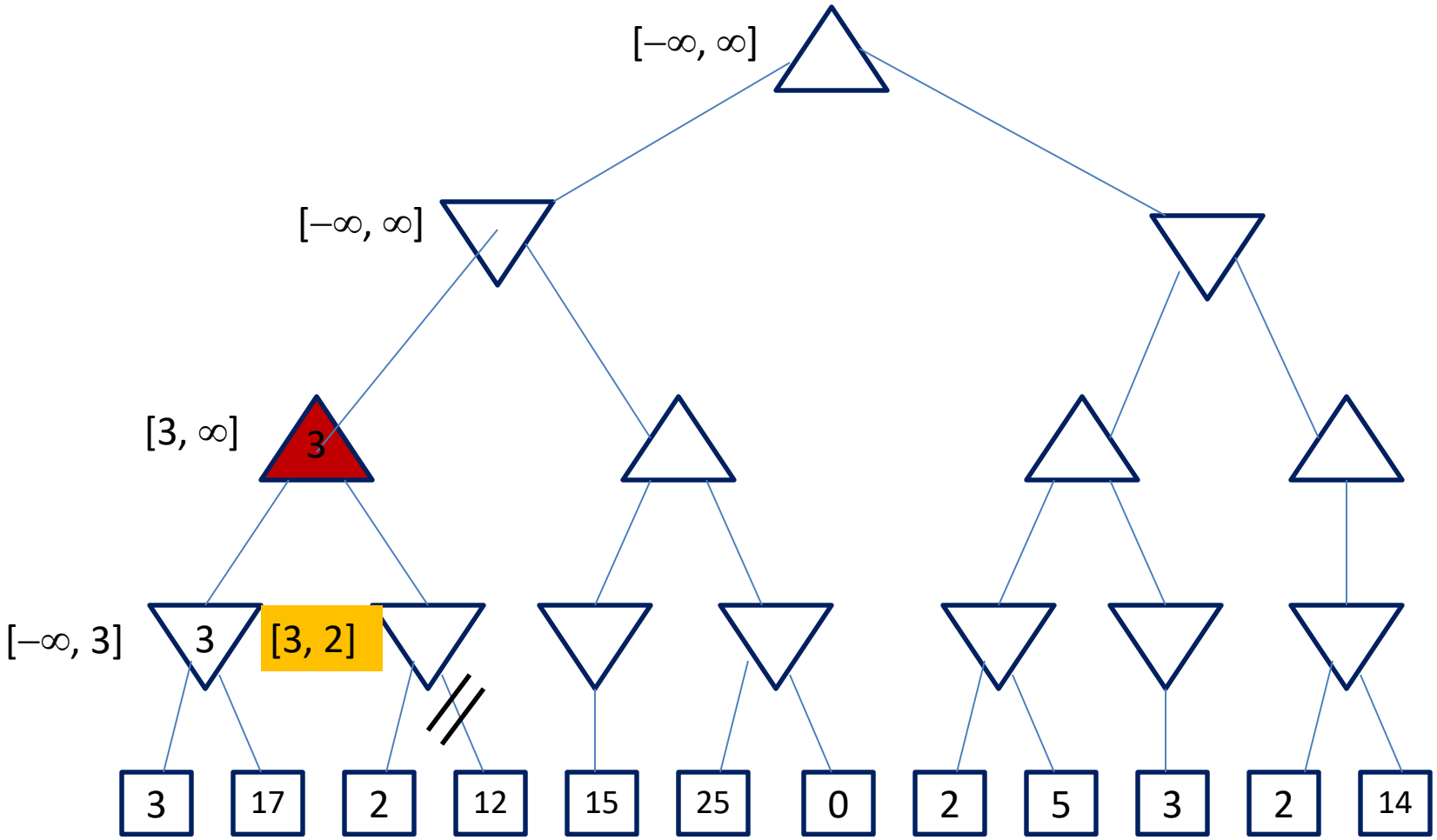
Ss
State-space Search

Alpha-Beta Pruning



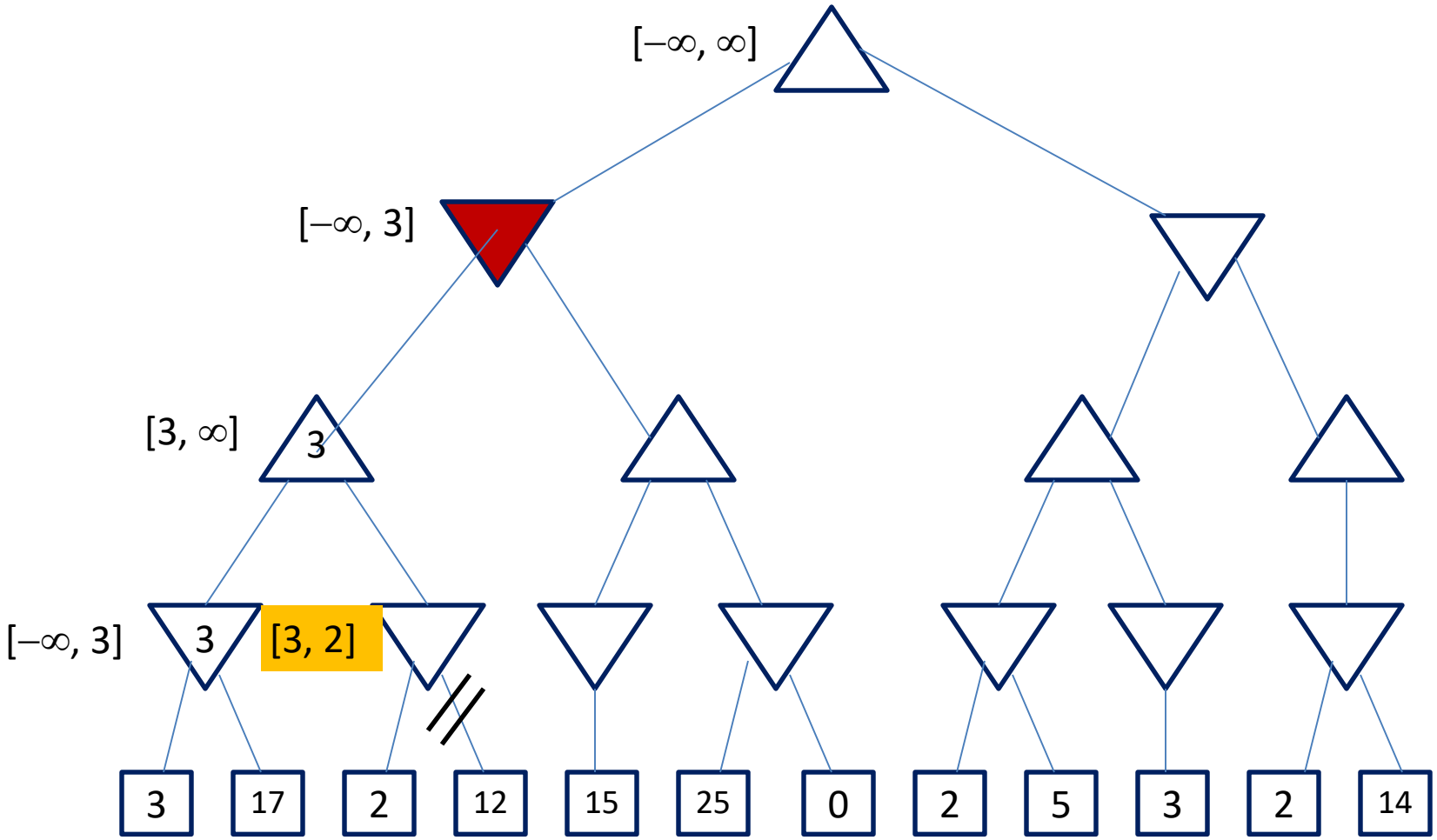
Ss
State-space Search

Alpha-Beta Pruning



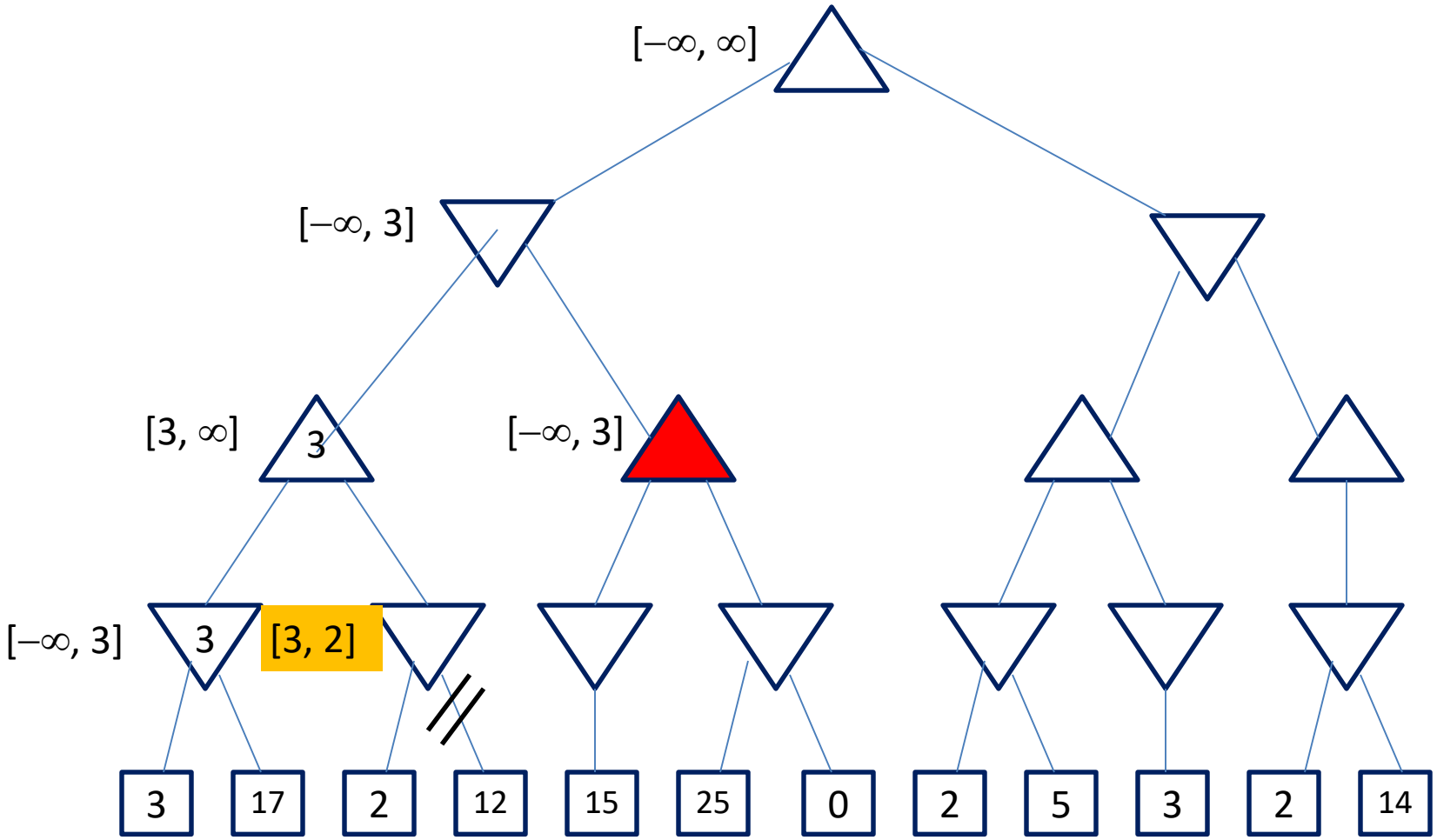
Ss
State-space Search

Alpha-Beta Pruning



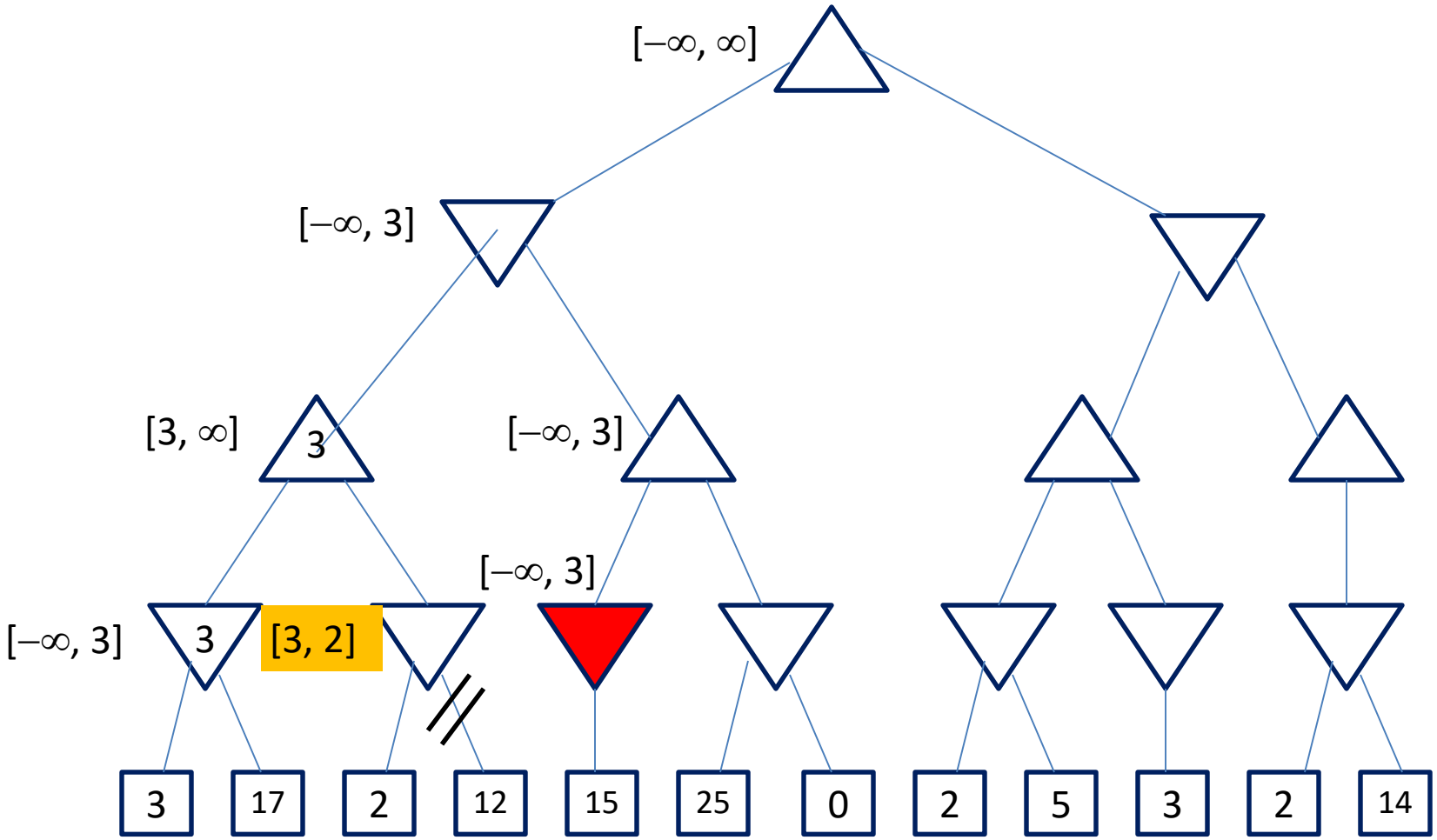
Ss
State-space Search

Alpha-Beta Pruning



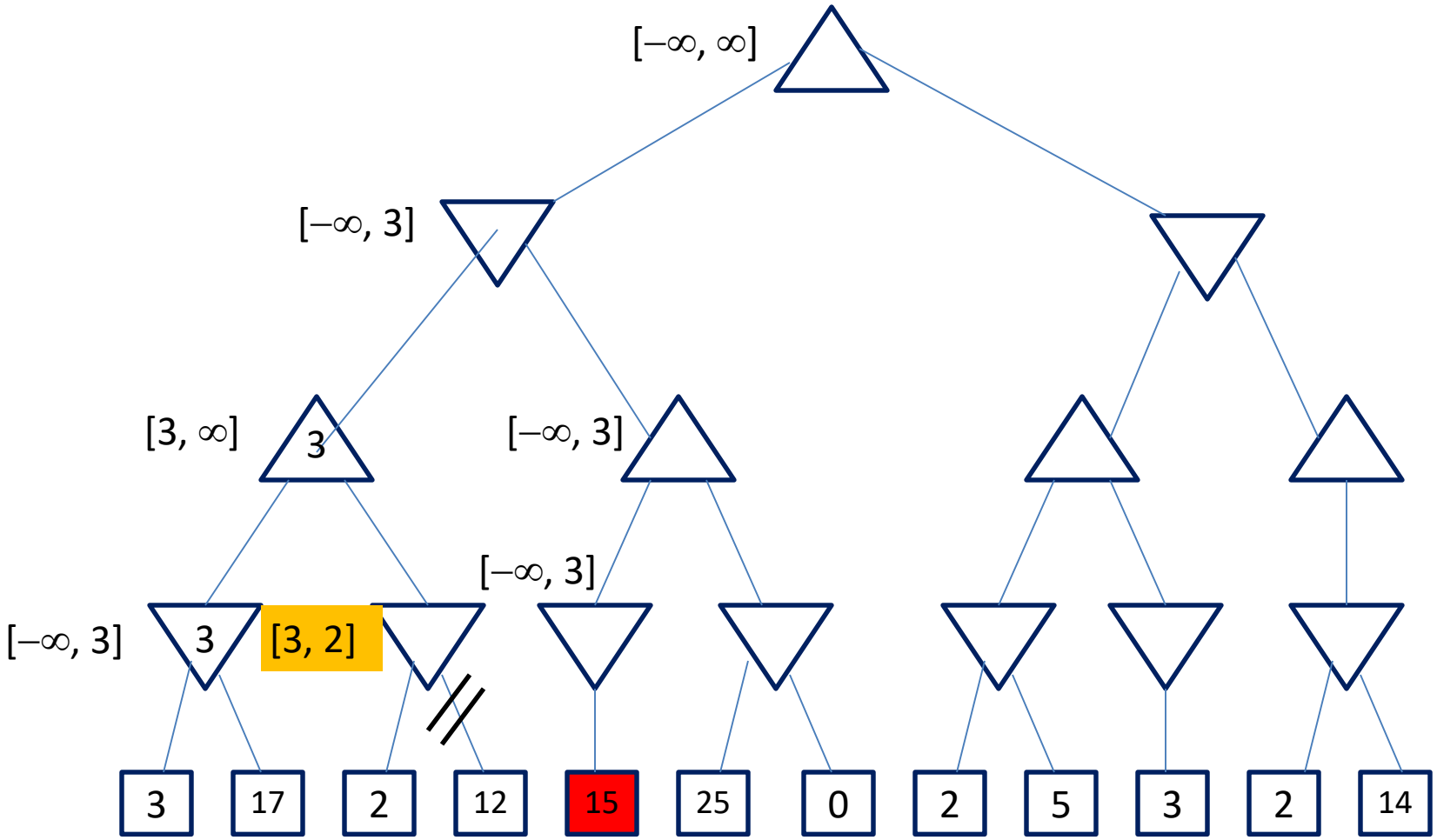
Ss
State-space Search

Alpha-Beta Pruning



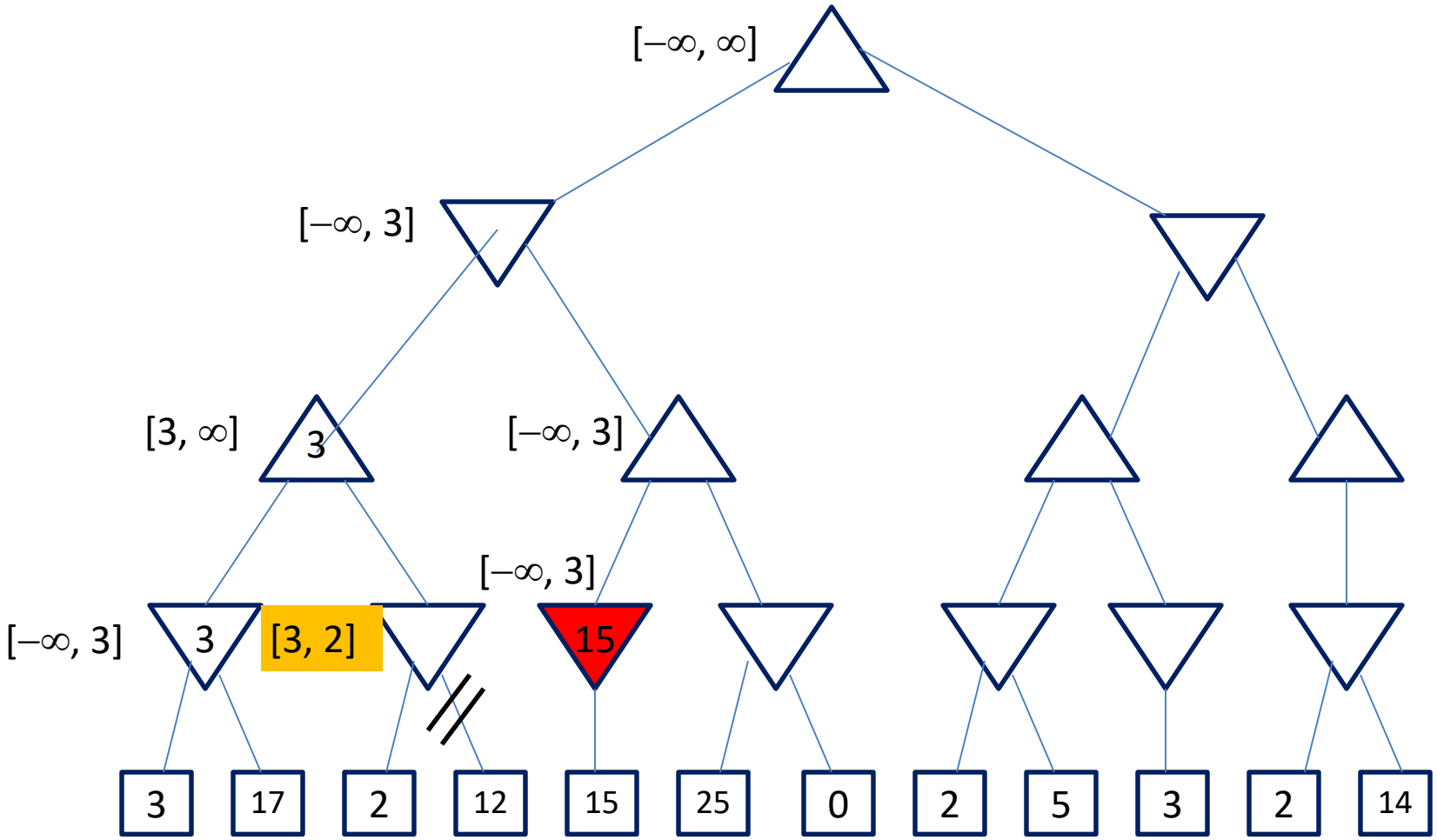
Ss
State-space Search

Alpha-Beta Pruning



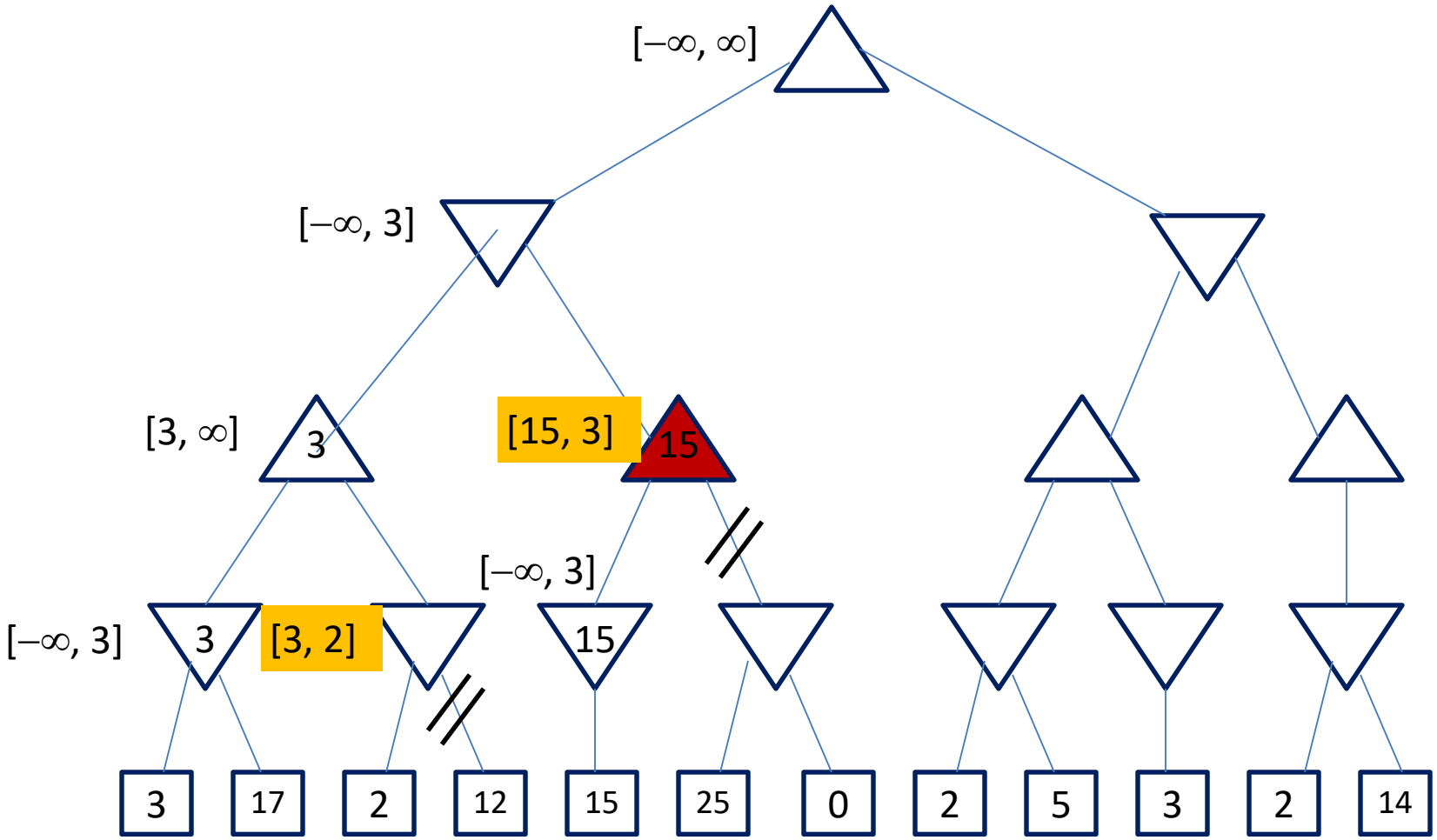
Ss
State-space Search

Alpha-Beta Pruning



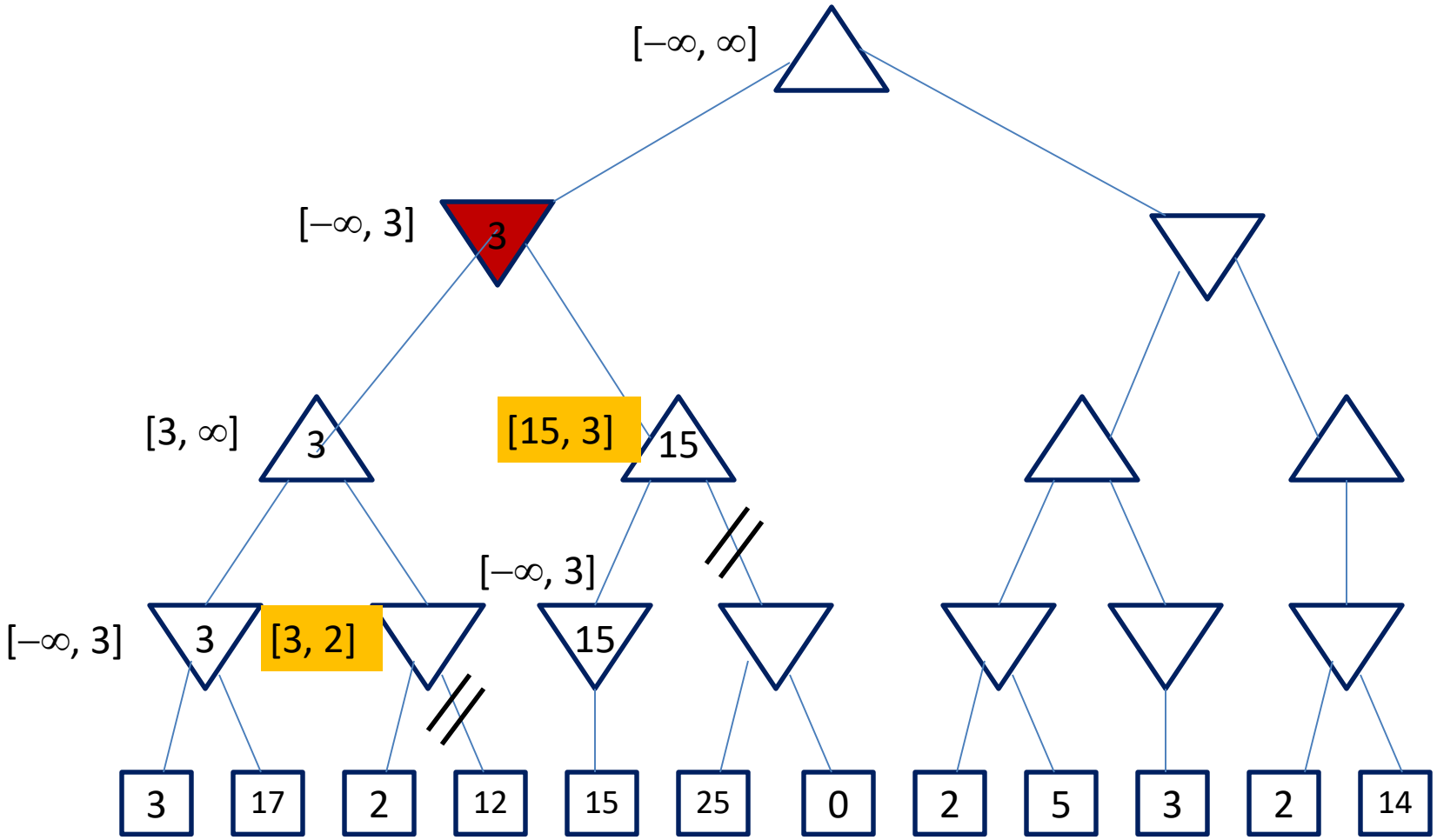
Ss
State-space Search

Alpha-Beta Pruning



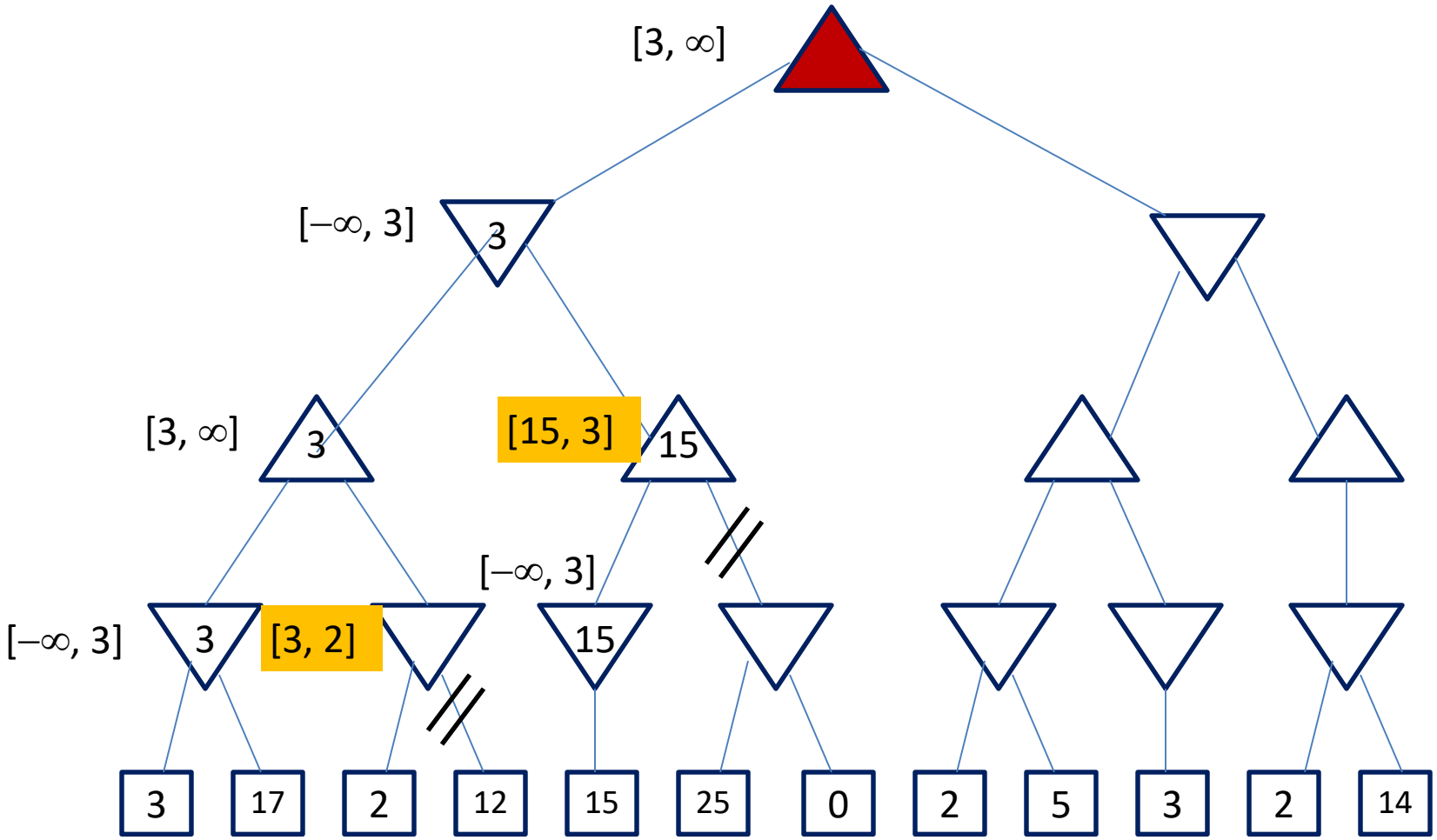
Ss
State-space Search

Alpha-Beta Pruning



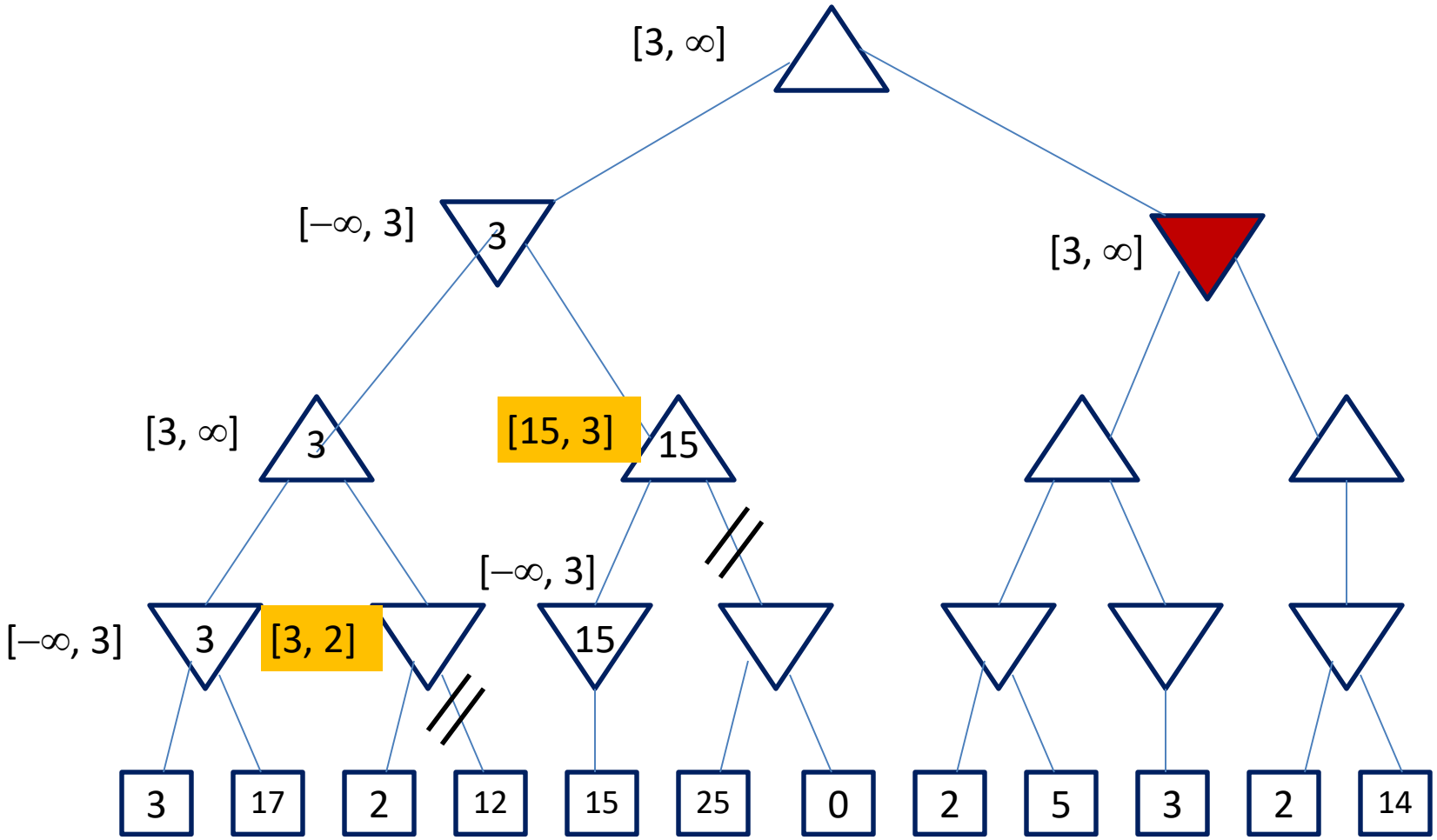
Ss
State-space Search

Alpha-Beta Pruning



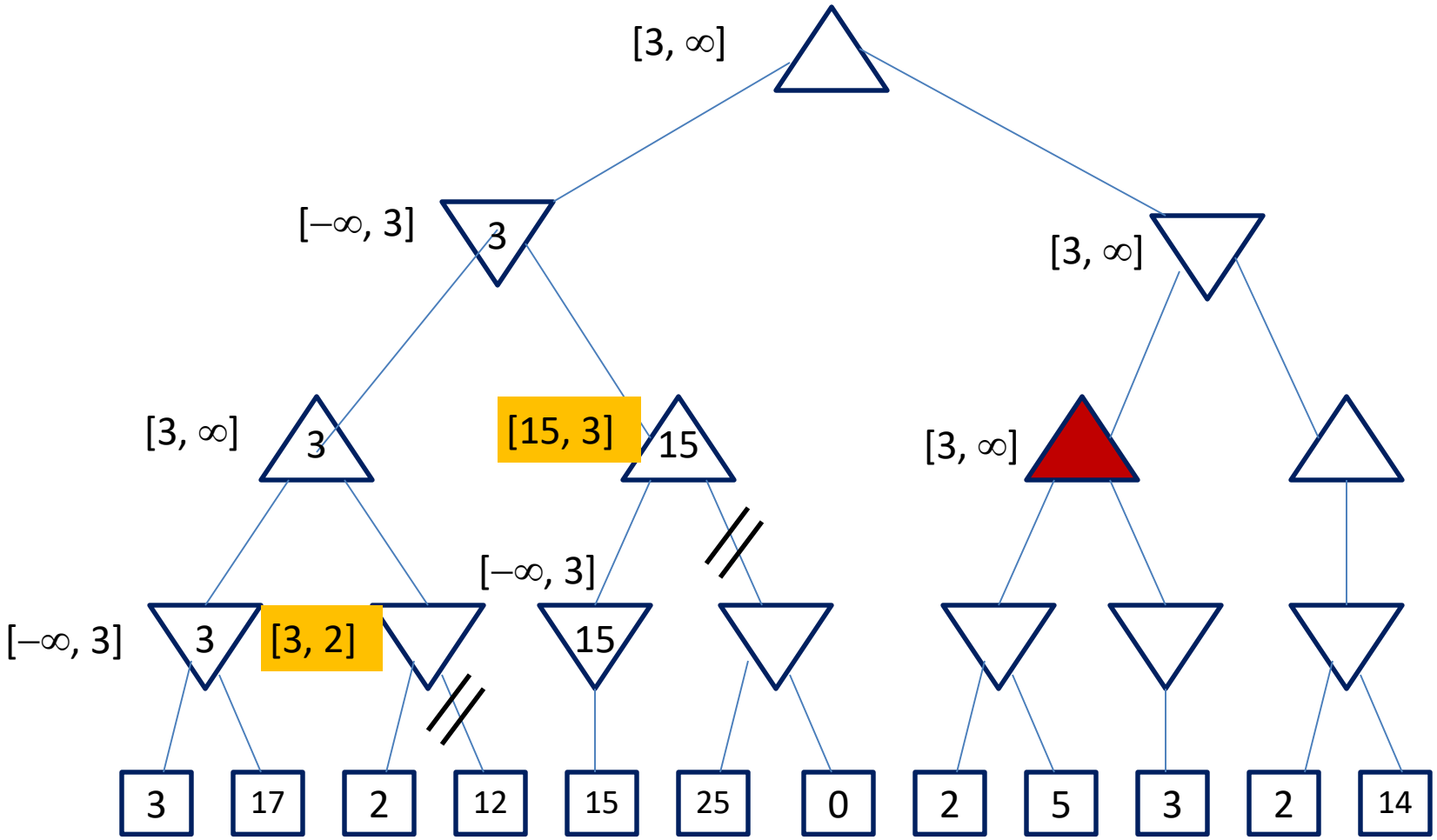
Ss
State-space Search

Alpha-Beta Pruning



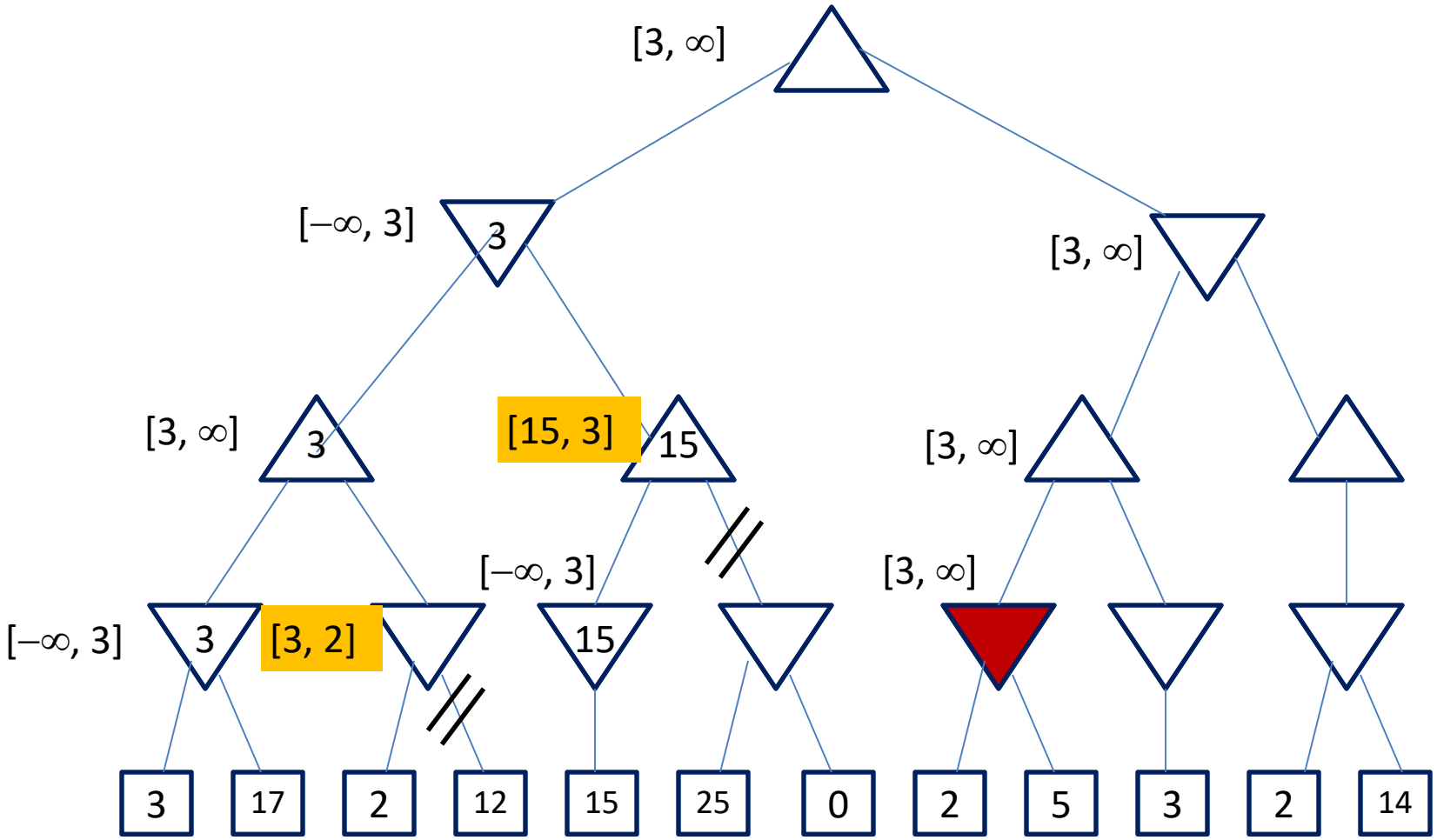
Ss
State-space Search

Alpha-Beta Pruning



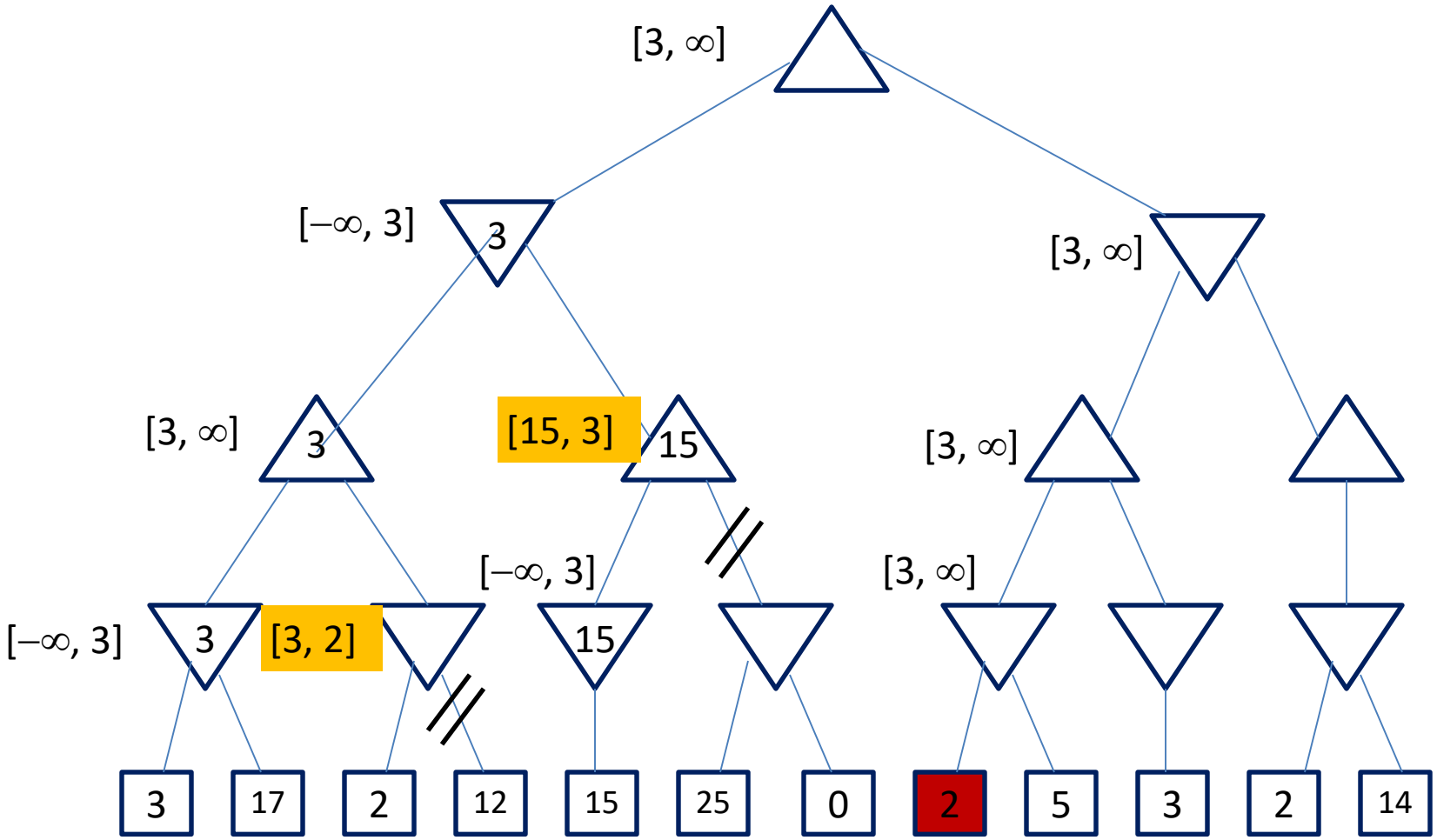
Ss
State-space Search

Alpha-Beta Pruning



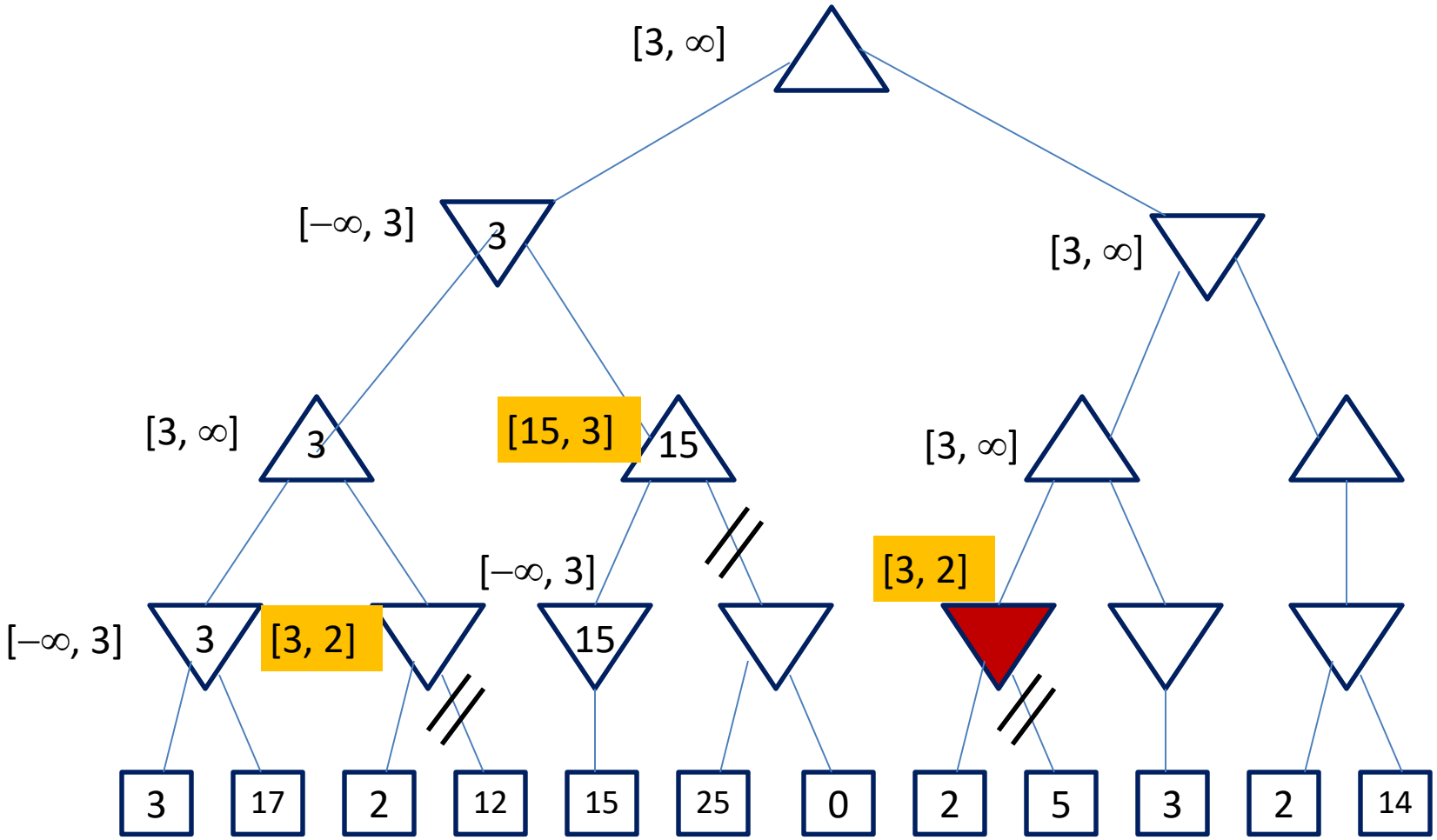
Ss
State-space Search

Alpha-Beta Pruning



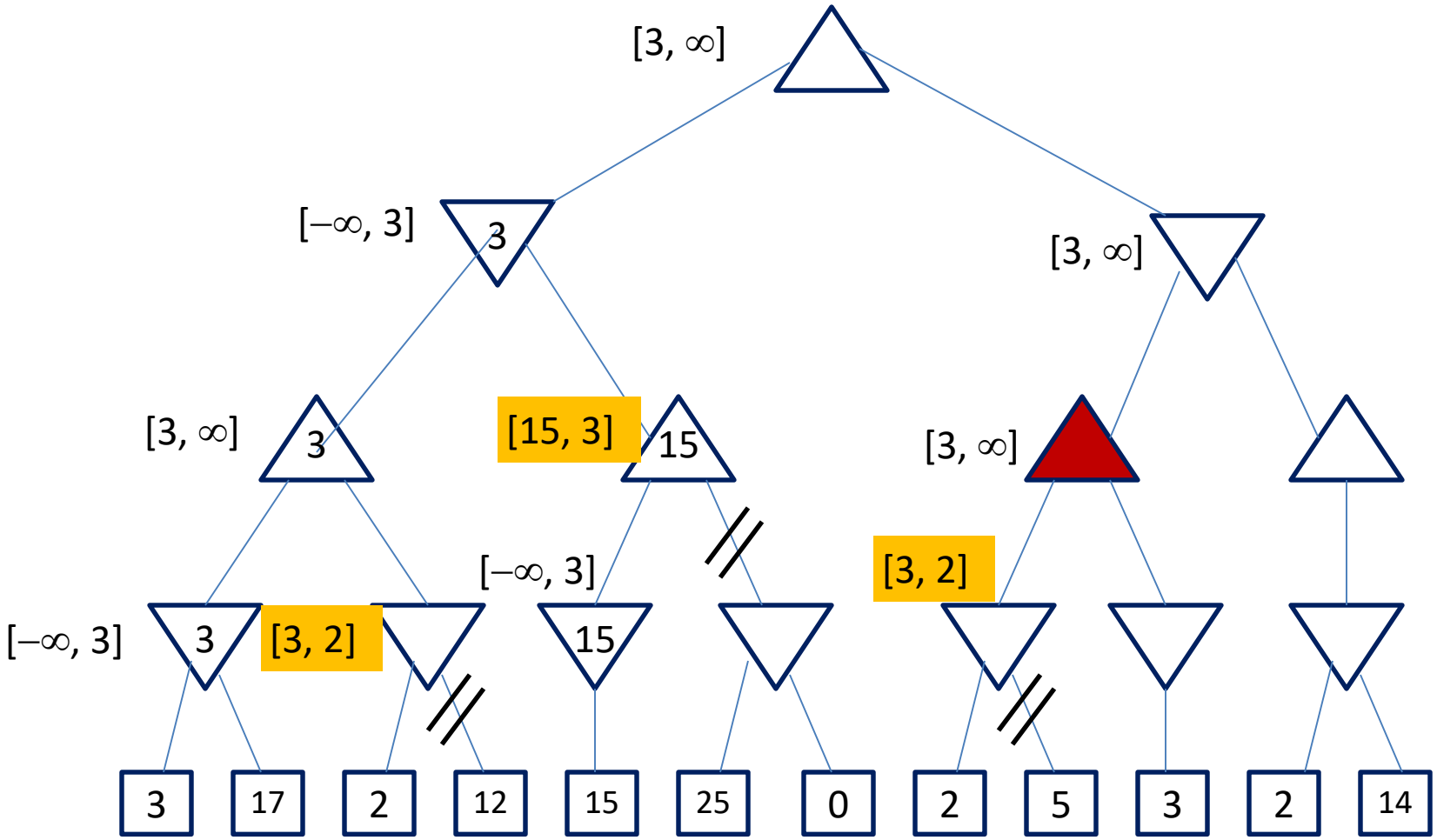
Ss
State-space Search

Alpha-Beta Pruning



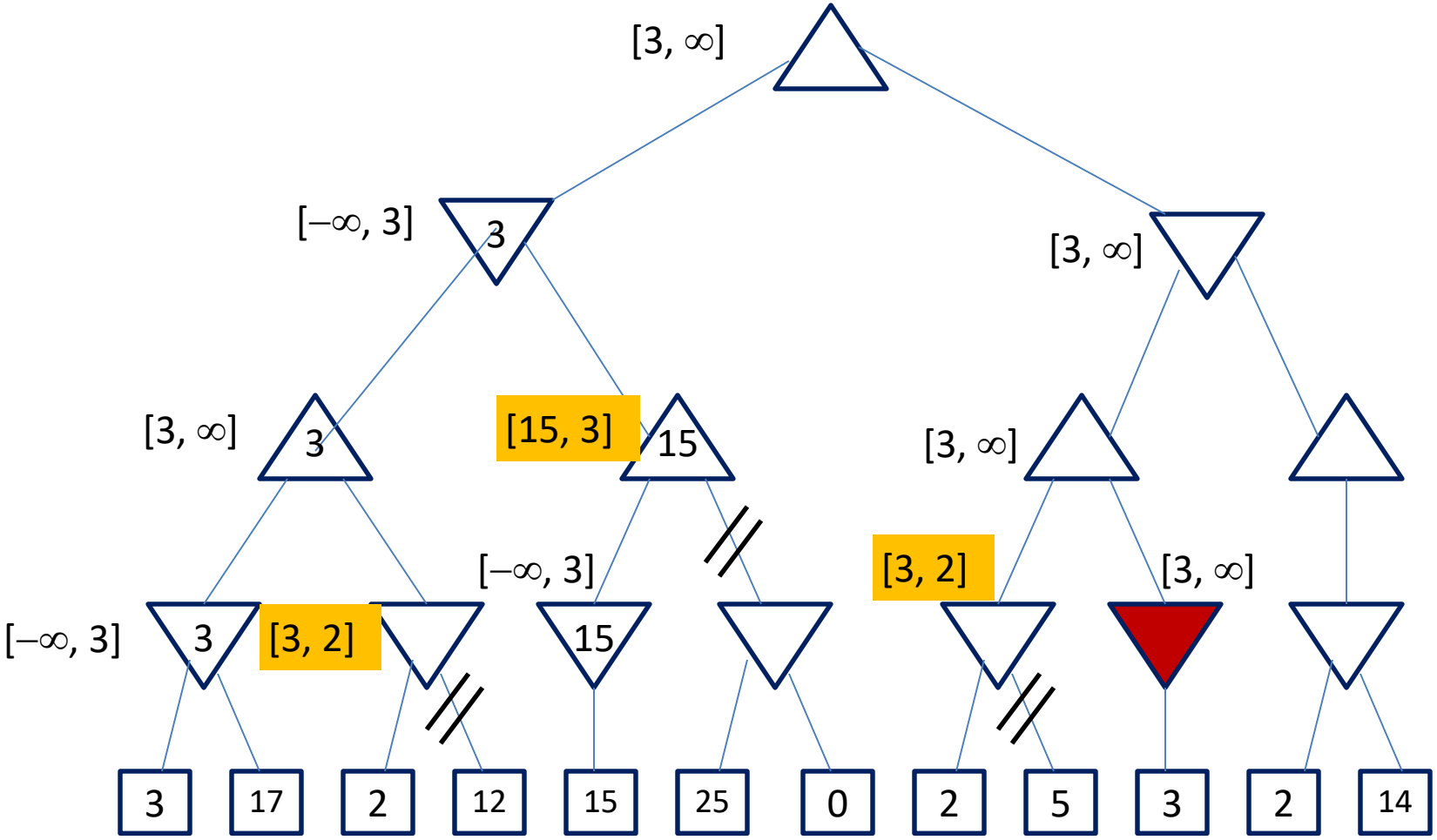
Ss
State-space Search

Alpha-Beta Pruning



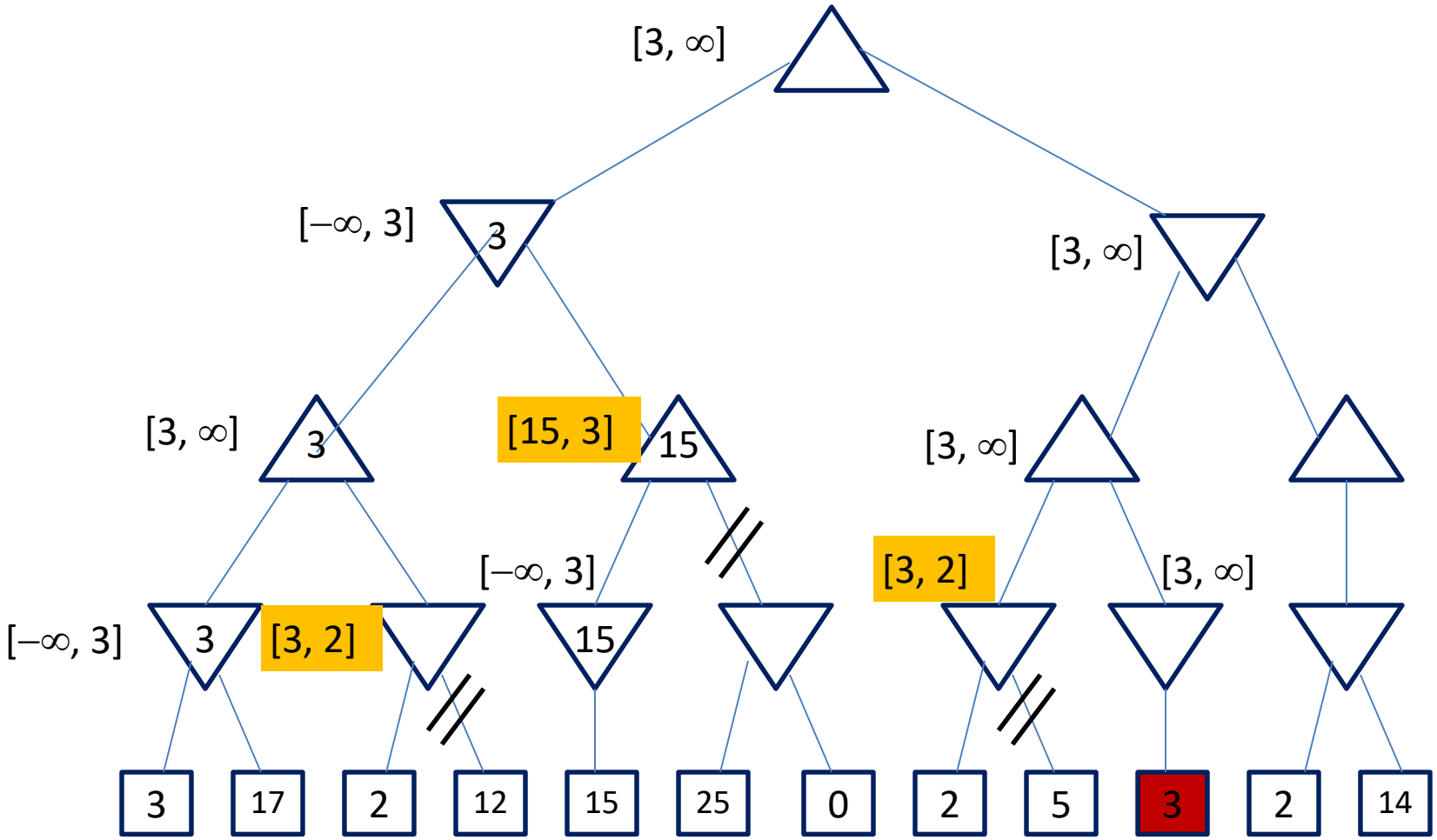
Ss
State-space Search

Alpha-Beta Pruning



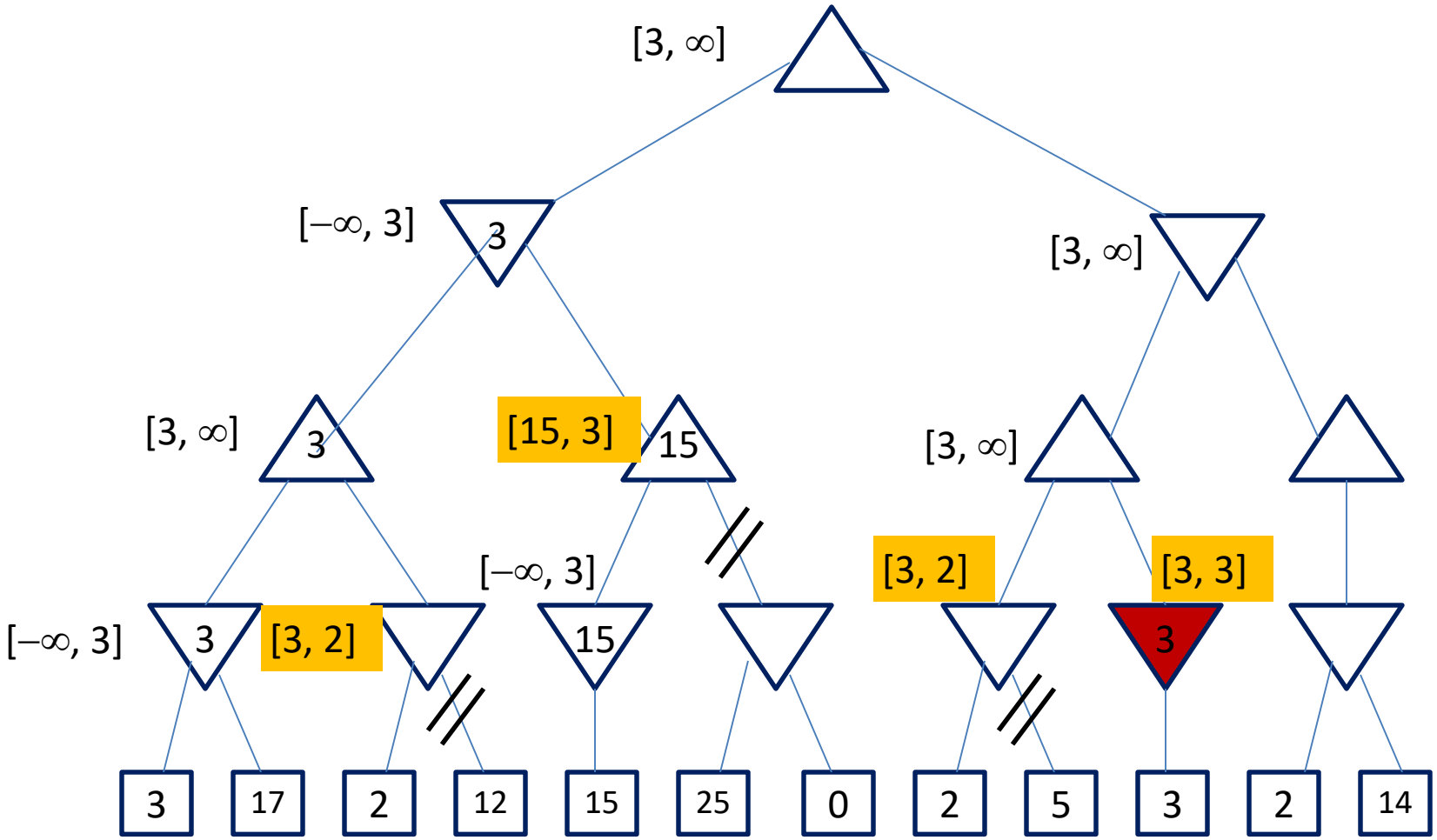
Ss
State-space Search

Alpha-Beta Pruning



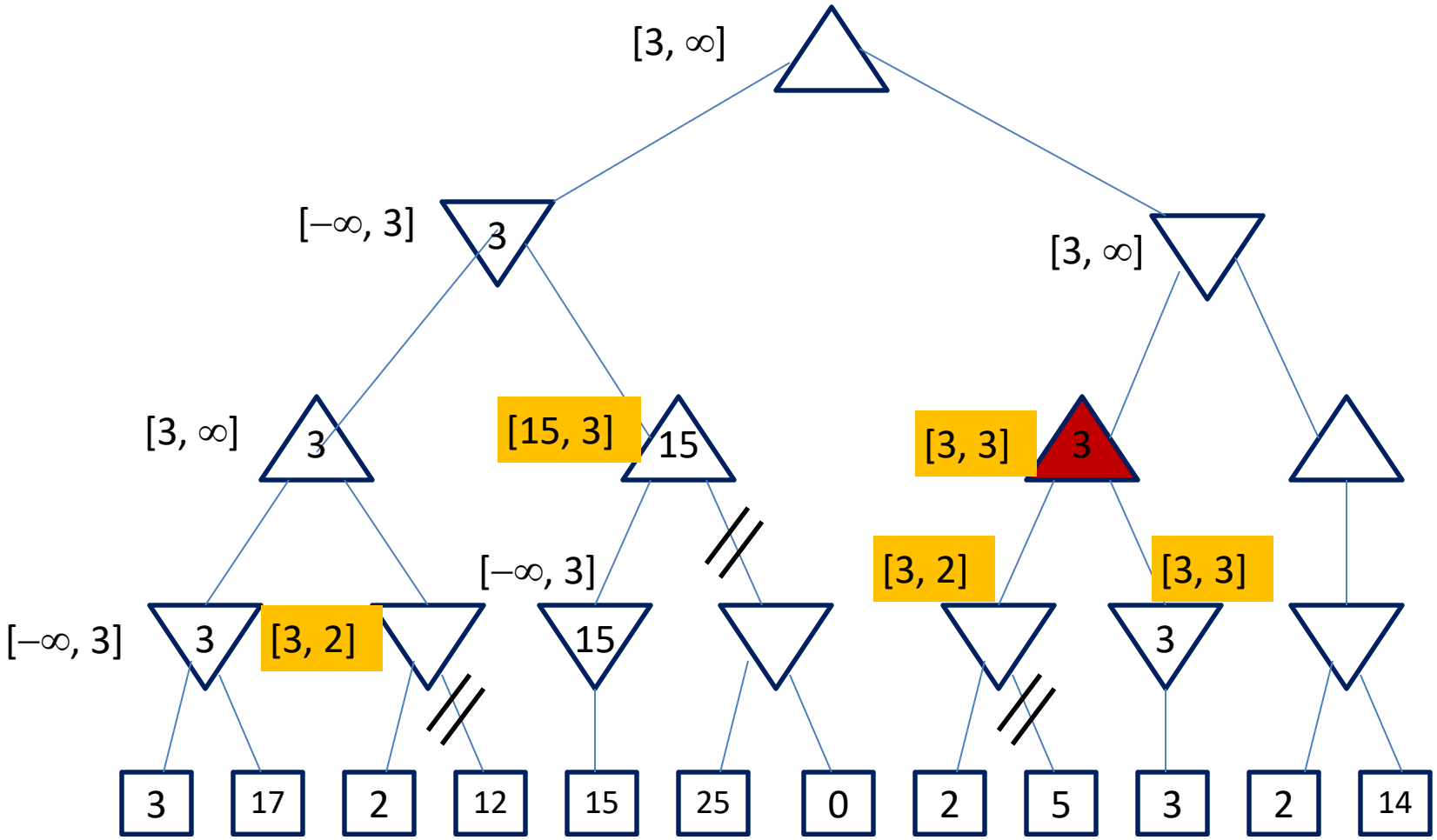
Ss
State-space Search

Alpha-Beta Pruning



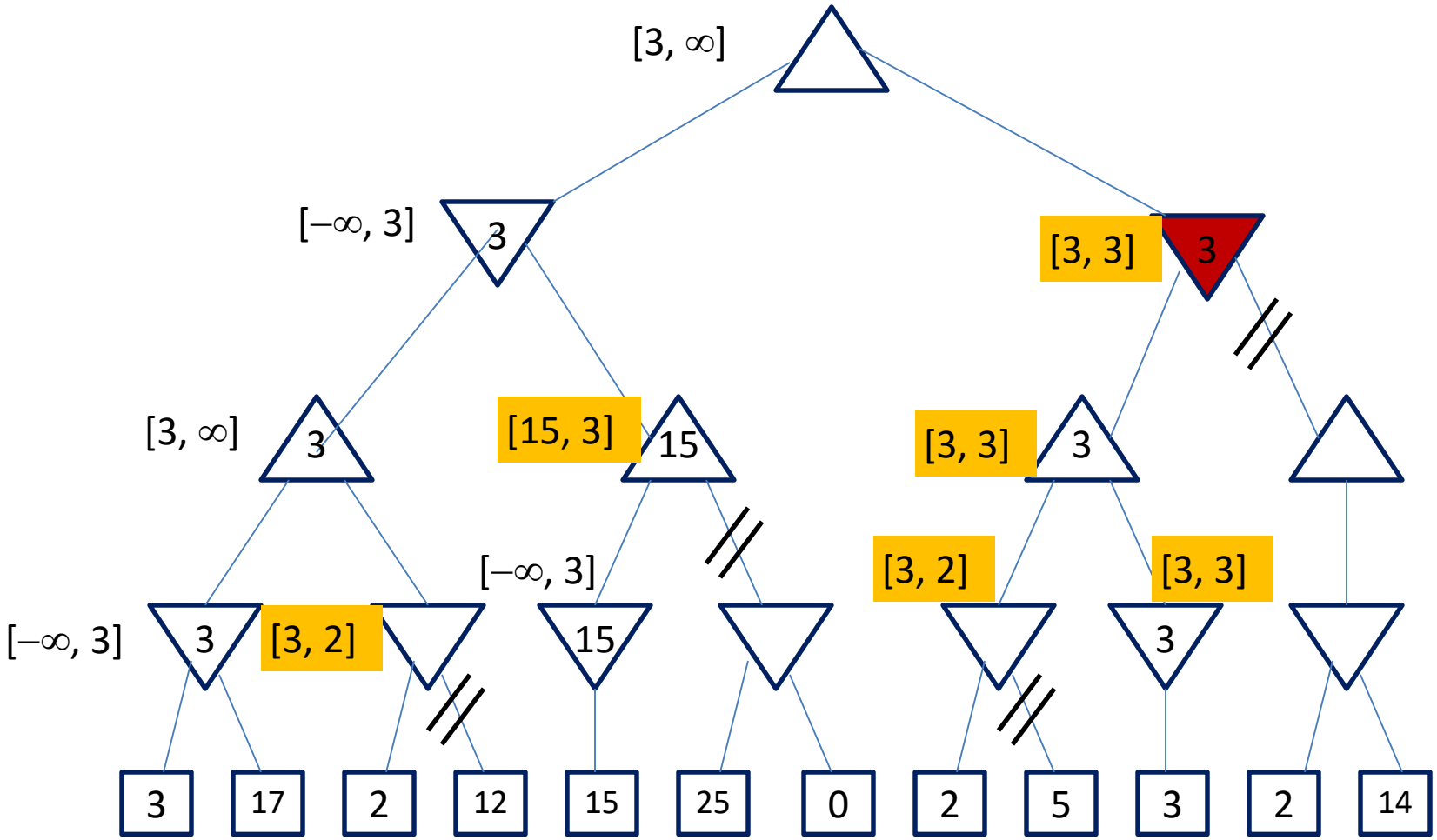
Ss
State-space Search

Alpha-Beta Pruning



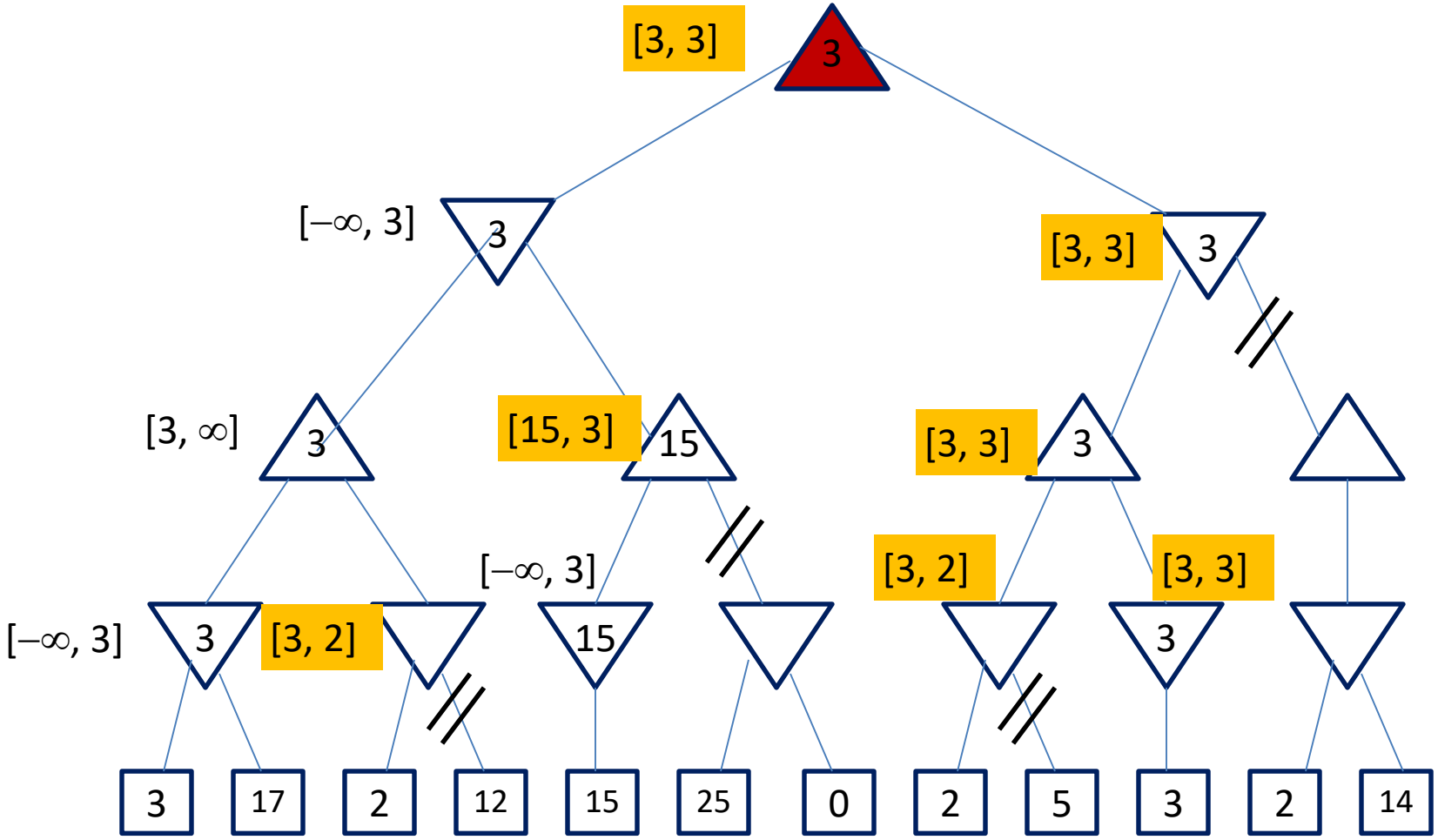
Ss
State-space Search

Alpha-Beta Pruning



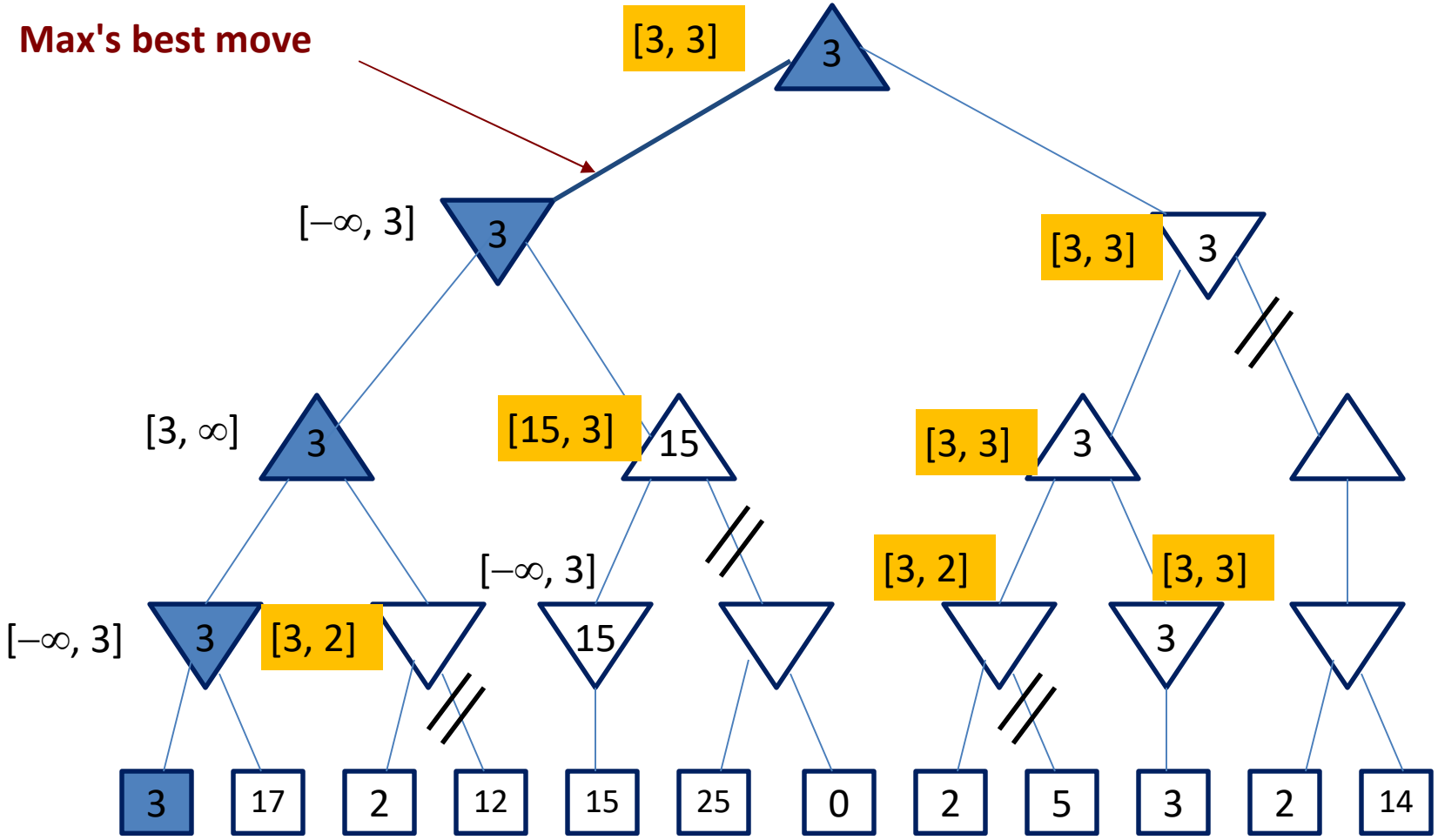
Ss
State-space Search

Alpha-Beta Pruning



Ss
State-space Search

Alpha-Beta Pruning



Ss
State-space Search

Strategy to Increase the Number of Cutoffs

At each non-leaf level, perform a static evaluation of all successors of a node and order them best-first before doing the recursive calls. If the best move was first, the tendency should be to get cutoffs when exploring the remaining ones.

Or, use Iterative Deepening, with ply limits increasing from, say 1 to 15. Use results of the last iteration to order moves in the next iteration.

In games like chess, α-β pruning typically allows searching ahead 2 times as deep. It tends to reduce the effective branching factor from d to approx. sqrt(d).

Ss
State-space Search

Strategy to Increase the Number of Cutoffs

At each non-leaf level, perform a static evaluation of all successors of a node and order them best-first before doing the recursive calls. If the best move was first, the tendency should be to get cutoffs when exploring the remaining ones.

Or, use Iterative Deepening, with ply limits increasing from, say 1 to 15. Use results of the last iteration to order moves in the next iteration.