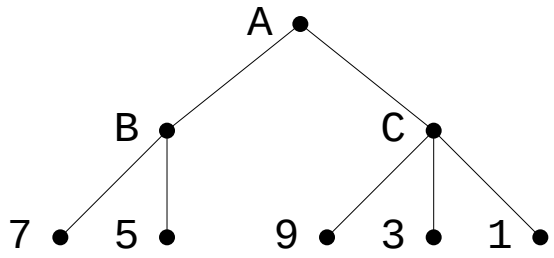
alpha-beta minimax pruning

- two-player game tree minimax computation:
  alpha/cutoffs allow subtree pruning
- worst case: no subtrees pruned, all leaves examined
- for each node, on the path from that node to the root,
  - *alpha* best already-seen option for MAX
  - *beta* best already-seen option for MIN
- example on next page:
  - compute minimax with DFS
  - update alpha/beta at each node
  - use these values to compute minimax at each node
- Pieter Abbeel video `https://www.youtube.com/watch?v=xBXHtz4Gbdo`

computing minimax, alpha, beta at a node

- for all internal MAX nodes, initialize mmx val, α to $-\infty$
- for all internal MIN nodes, initialize mmx val, β to $+\infty$
- traverse nodes DFS order
  - arrive at node: update v, α/β
- backing up to node:
  - update v, α (if MAX), β (if MIN)
  - before descent to next child: can we prune?
  - prune if $\alpha \geq \beta$
    (at MAX: MIN has option elsewhere at least as good)
    (at MIN: MAX has option elsewhere at least as good)

- **exercise.** trace on this tree. show each update.
  check answer with `alphabeta/alphabeta.py`



- **answer.** from A, descend to B. from B, descend to leaf 7. from leaf 7, backup: $v(B) \leftarrow \beta(B) \leftarrow 7$. **can we prune?** no: $v(B) = \beta(B)$ so descend to leaf 5. from leaf 5, backup: $v(B) \leftarrow \beta(B) \leftarrow 5$. from B, backup: $v(A) \leftarrow \alpha(A) \leftarrow 5$. $v(A) = \alpha(A)$ so descend to C: $\alpha(C) \leftarrow 5$. from C, descend to leaf 9. from leaf 9, backup: $v(C) \leftarrow \beta(C) \leftarrow 9$. **can we prune?** no: $v(C) = 9 > \alpha(C) = 5$ so descend to leaf 9. from leaf 3, backup: $v(C) \leftarrow \beta(C) \leftarrow 3$. **can we prune?** yes: $v(C) = 3 < \alpha(C) = 5$ so backup now (effectively pruning C's remaining children leaf 3, leaf 1). from C, backup to A: $v(A) \leftarrow \alpha(A) \leftarrow 5$. from A, backup. done.

prune: $\alpha \geq \beta$ **test better than** $\alpha > \beta$ **test**

- $\alpha \geq \beta$ or $\alpha > \beta$?
- on path-to-root, **as good** or **better** opponent move elsewhere
- at current node, score can only get worse for opponent

  so pointless to continue search: backup
- both tests work but $\alpha \geq \beta$ prunes more nodes than $\alpha > \beta$

  so prefer $\alpha \geq \beta$ test
- e.g. what if all leaf nodes have same MAX-score?
  - how much pruning with cutoff $\alpha > \beta$?
  - answer: none
  - how much pruning with cutoff $\alpha \geq \beta$?
  - answer: maximum amount of pruning possible `:)` **woo hoo**