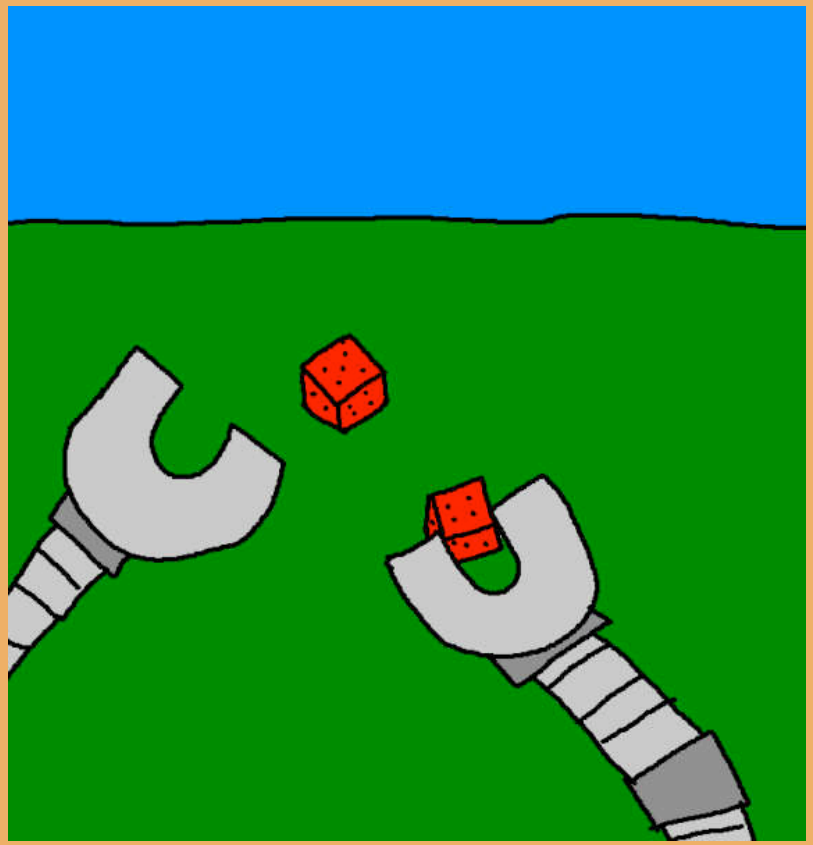
Game Playing

M. Anthony Kapolka III
Wilkes University
CS 340 - Online

Features of Games

- deterministic vs. stochastic
- goal (adversarial vs. cooperative)
- perfect vs. partial information
- number of players (1 or more)
- interaction scheduling

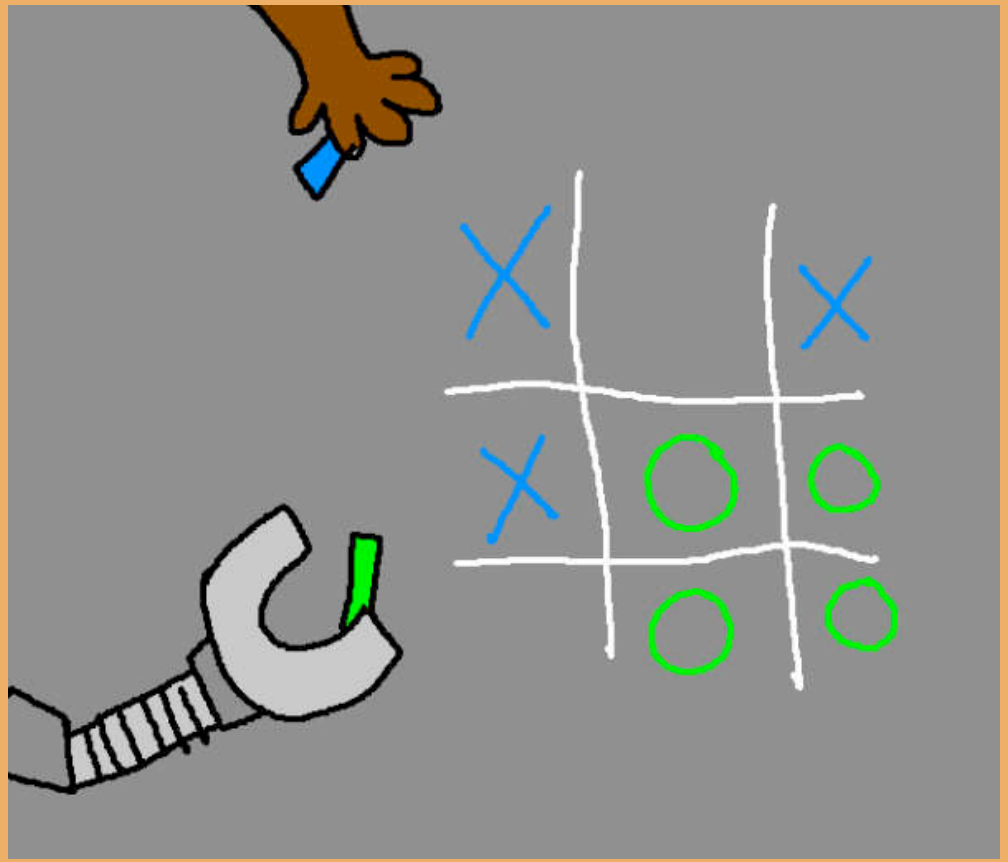


As State Spaces

- State variables describe game features
- Start state(s) define initial setup
- States are legal game positions
- Moves are transitions between states
- Goal determines terminal states
- Two players? bi-partite state space!

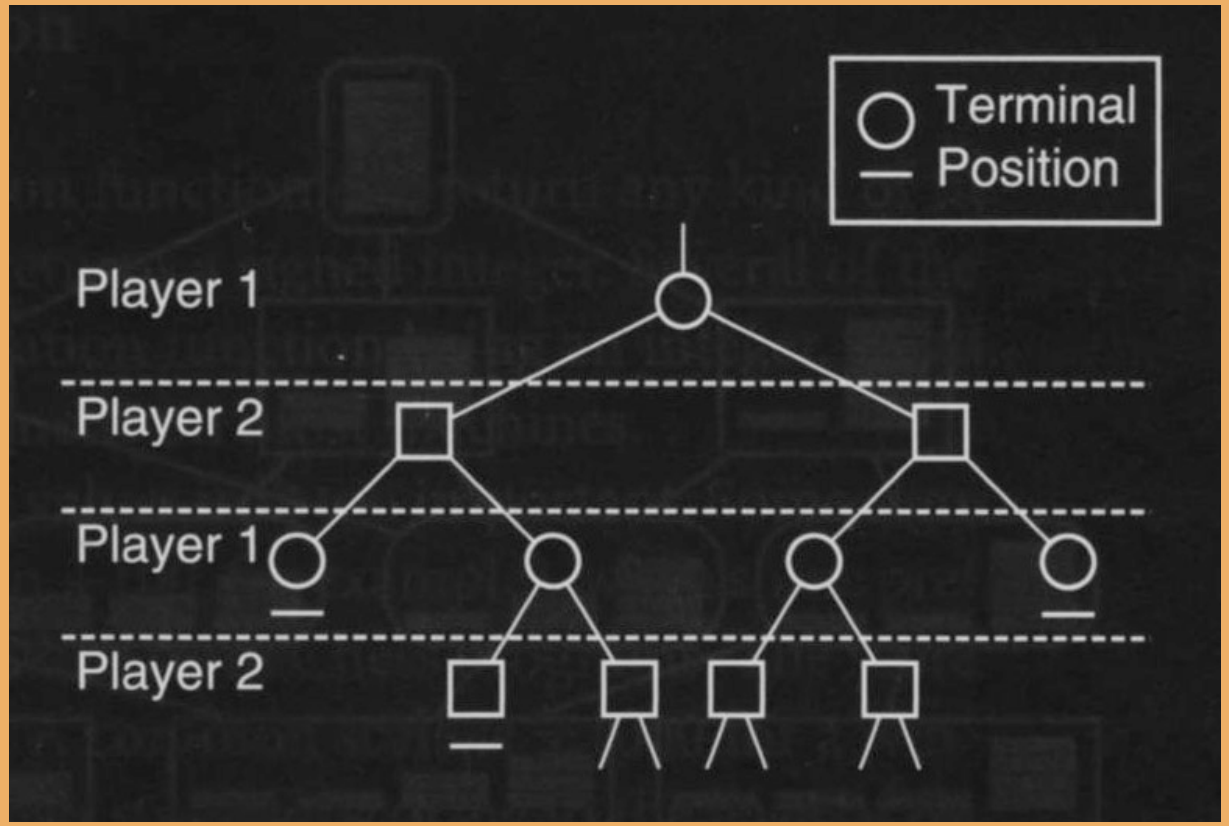
Perfect Information

- Players know everything there is to know
  - No hidden information
  - No random events
- Players need not have same set of moves



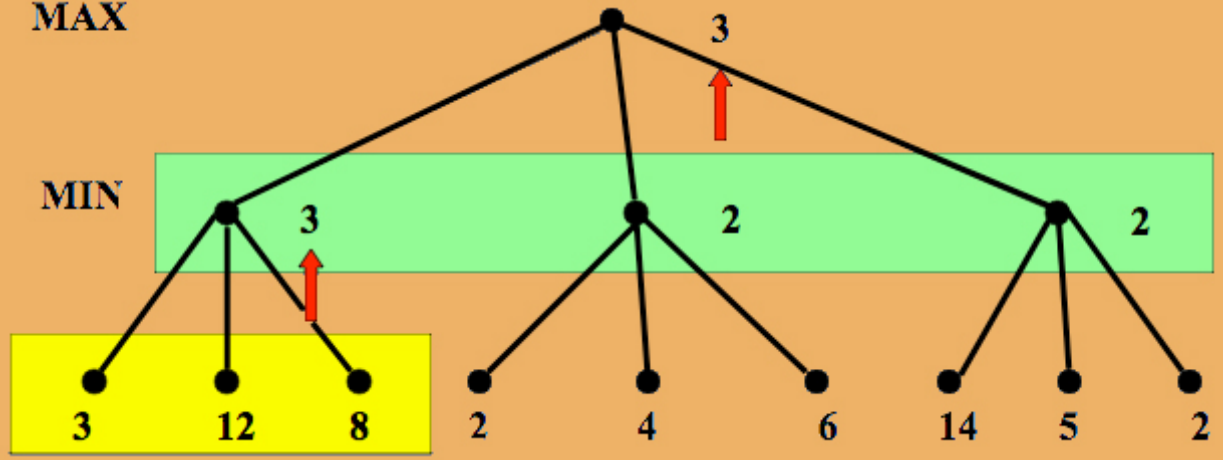
Contingency Problem

- Opponent has many possible moves
- Our solution must "cover" them
- Opponent's behavior introduces uncertainty
- Assume rational opponent
  - maximizes his/her utility (payoff) function



AI for Games, Millington & Funge: Figure 8.2

MiniMax Algorithm



Game Tree showing Two Ply

Token Game

- Two players: Blue and Red.
- Start state: 5 tokens, Blue to play
- Transition: Take 1, 2, or 3 tokens
- Goal state: Take last token

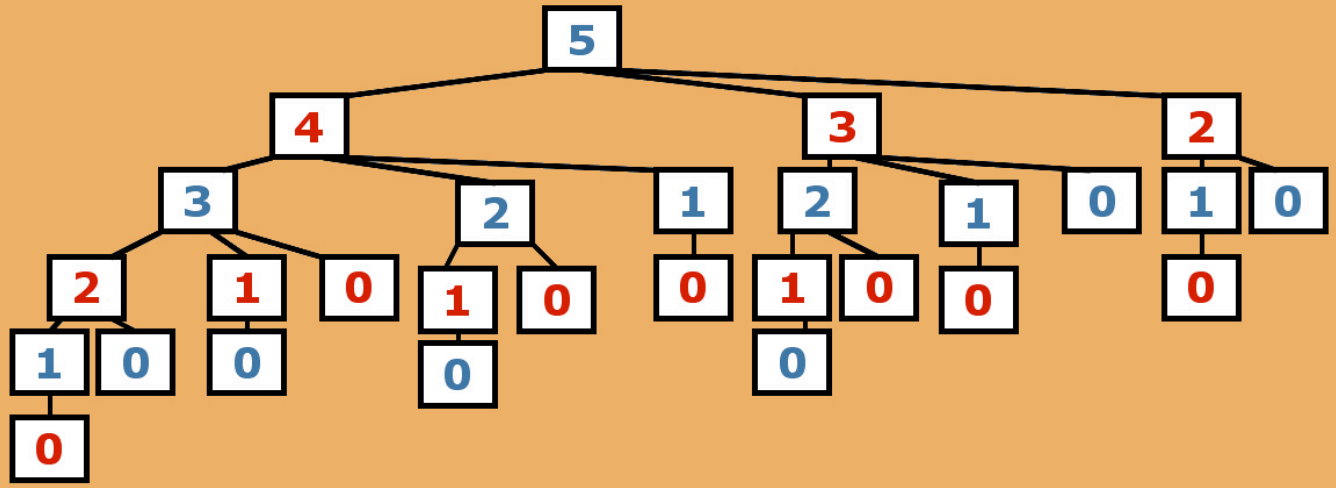
MiniMax

- Assumes both players play perfectly
- Min wants lowest possible score - or chooses the move which will minimizes the score resulting from Max choosing the maximizing move.
- Max does the reverse - choses the move that maximizes the score given Min's minimizing move.

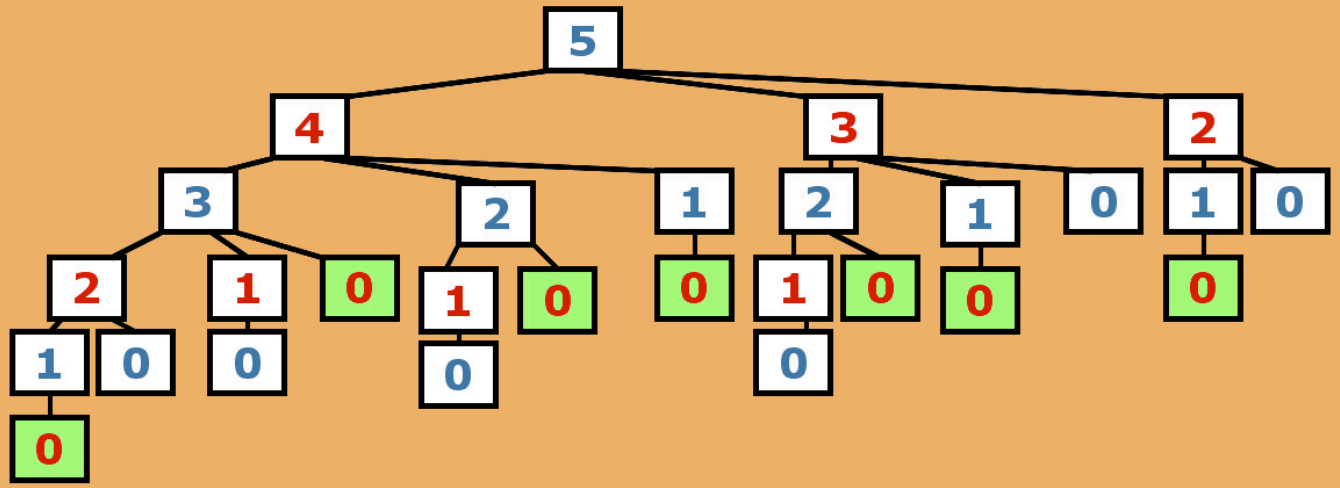
MiniMax Procedure

- Statically evaluate states at depth d
- Work upwards
  - At Max level, choose maximum of child node values
  - At Min level, choose minimum of child node values
- Can code depth first, so space efficient

Token Game Space

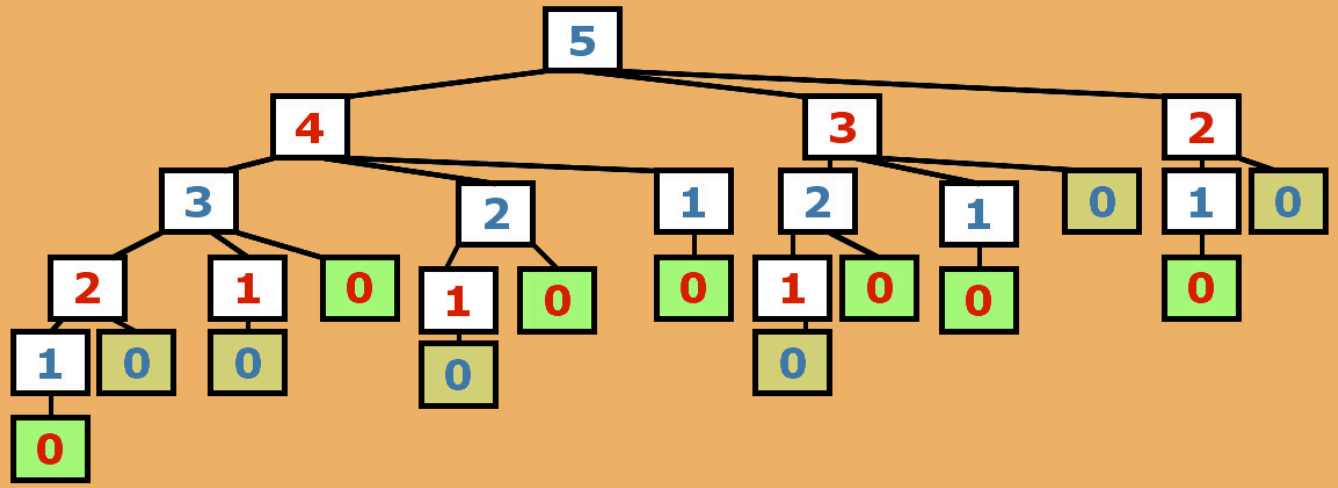


Token Game Space



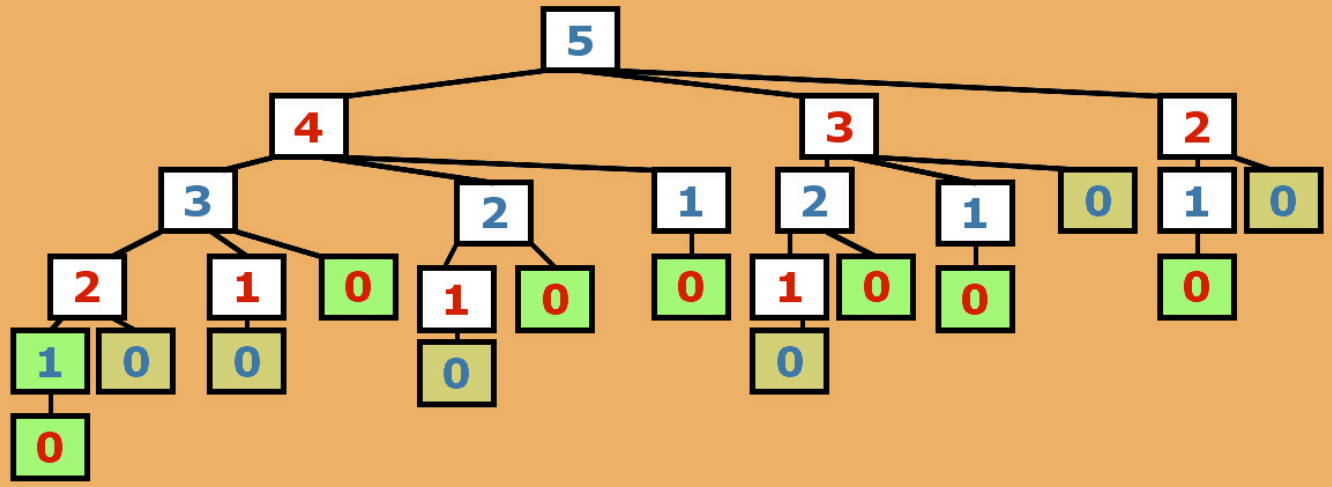
Light Green - Blue Player Wins

Token Game Space



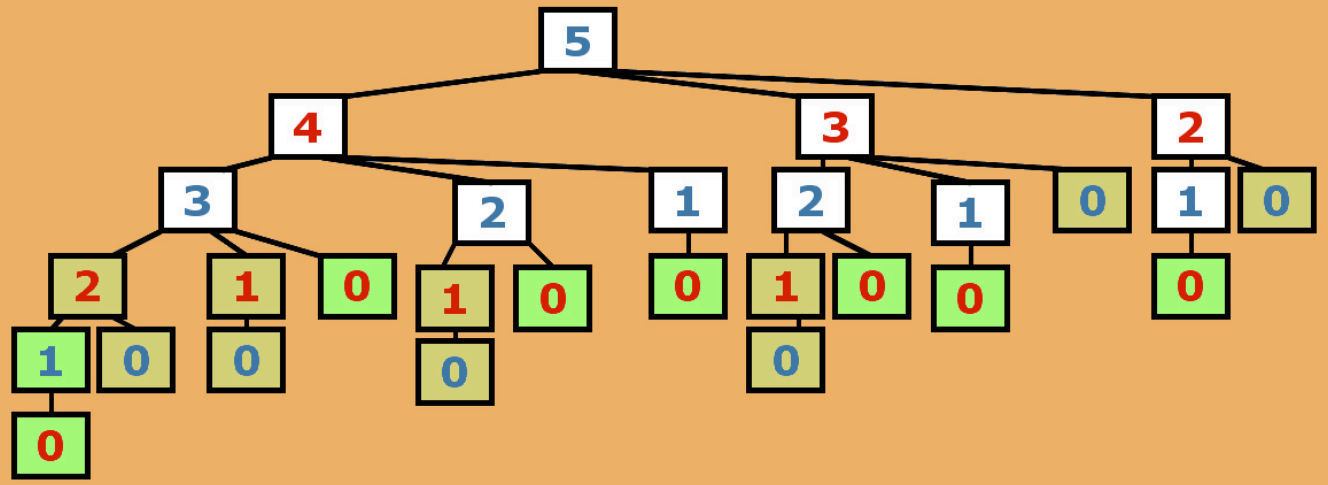
Dark Green - Red Player Wins

Token Game Space



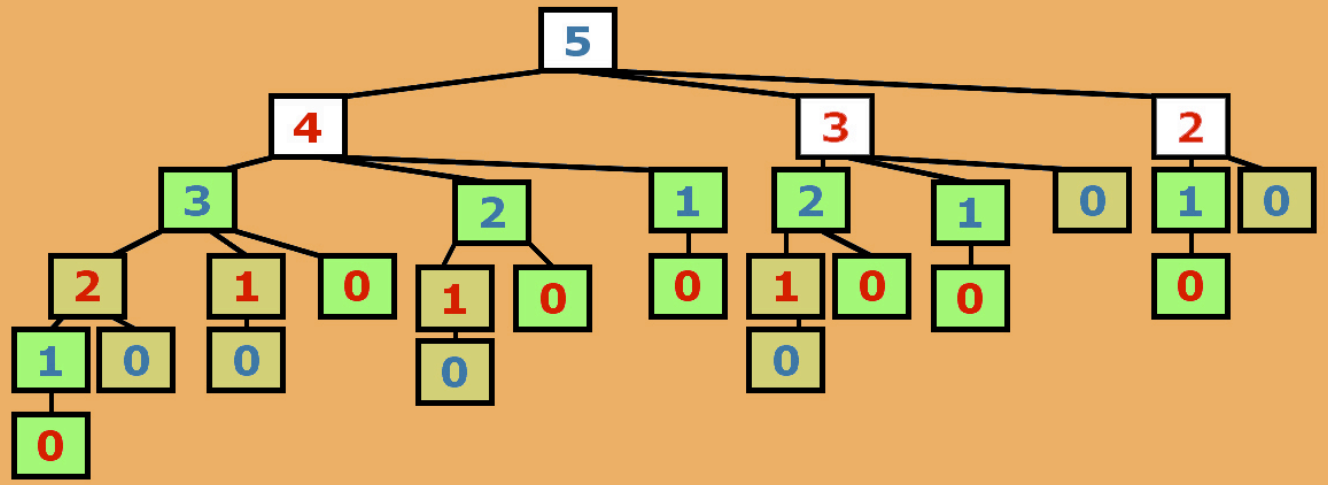
Propagate results up the tree

Token Game Space



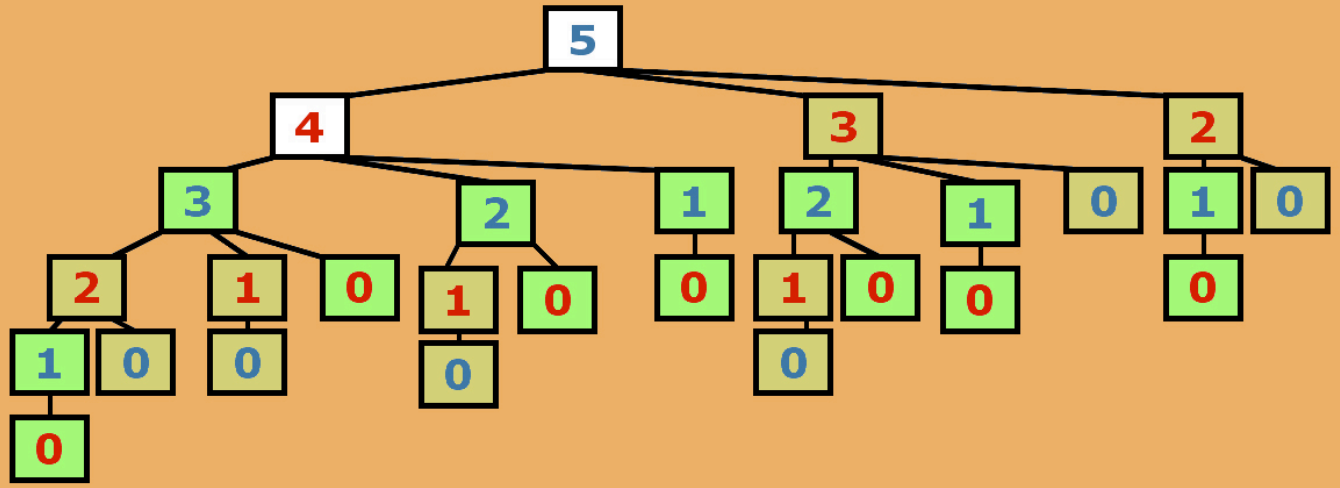
Red player choses winning moves when possible

Token Game Space



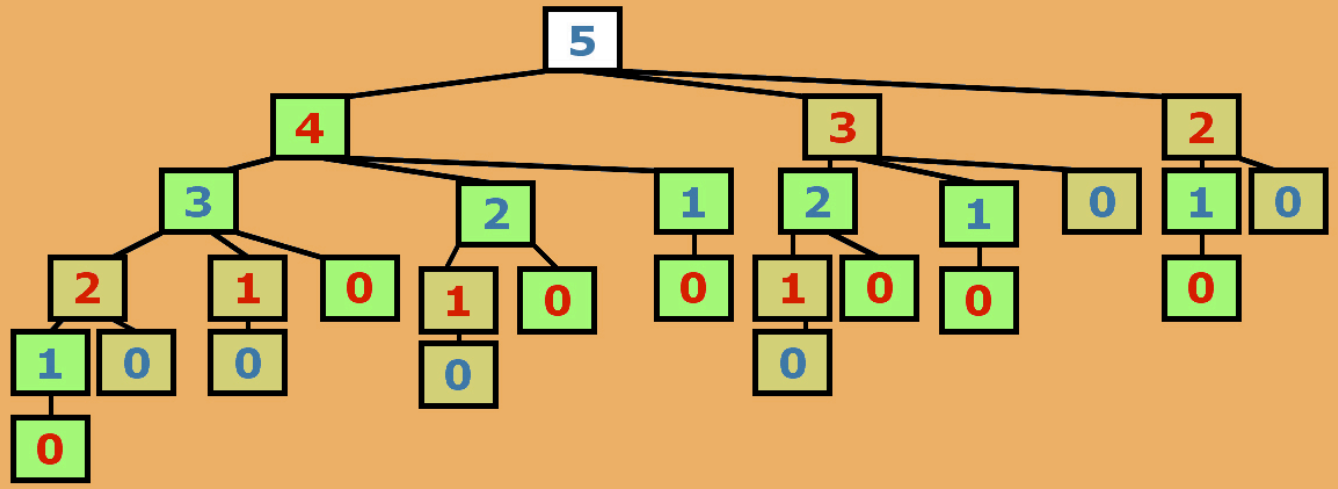
Likewise, Blue player choses winning moves

Token Game Space



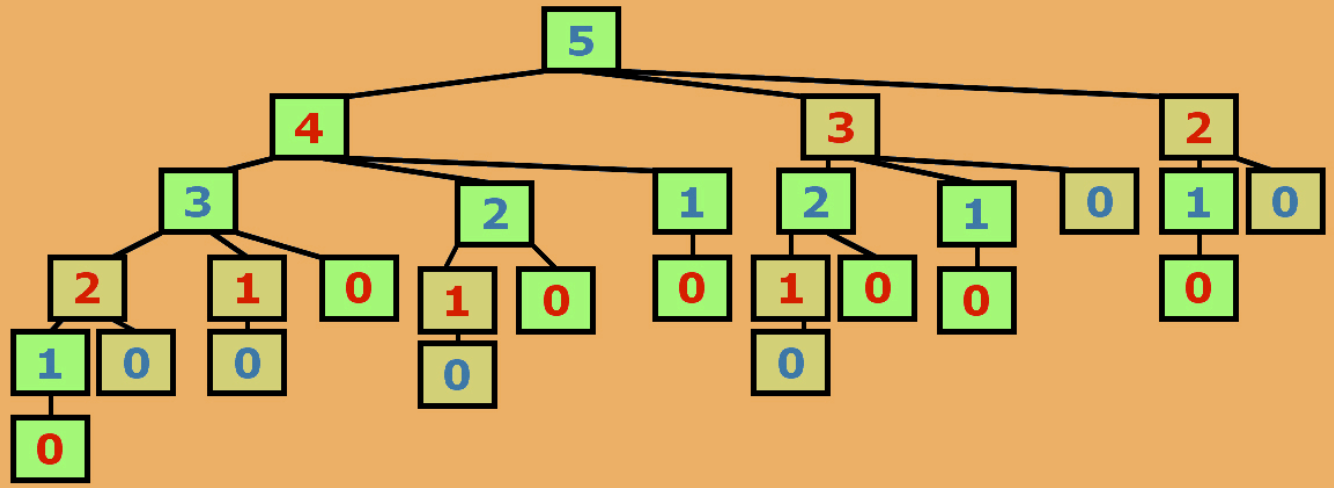
Red player choses… but look…

Token Game Space

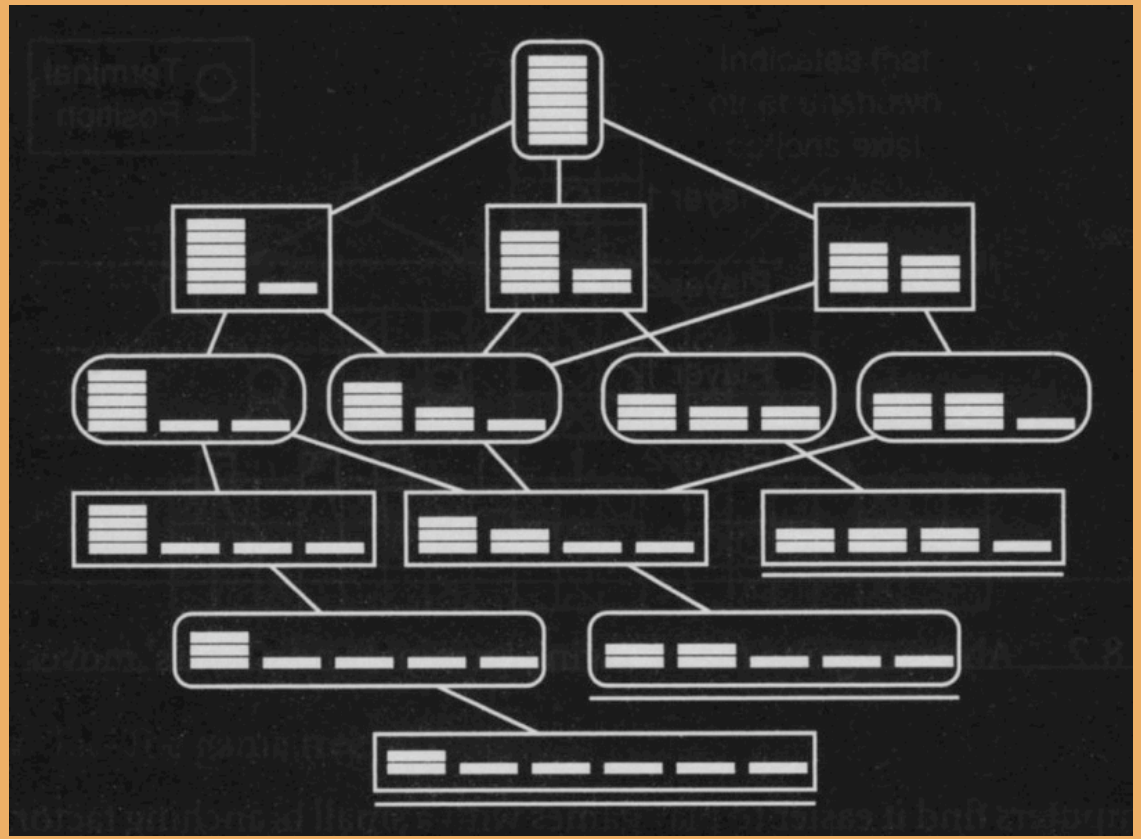


No winning move for Red there…

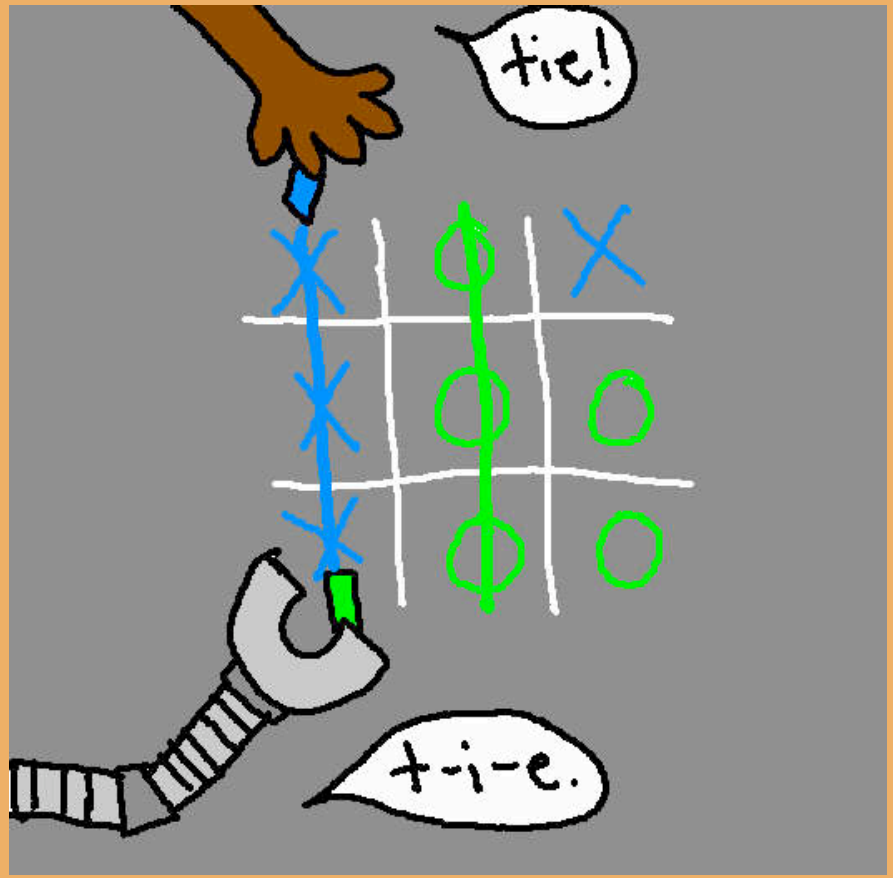
Token Game Space

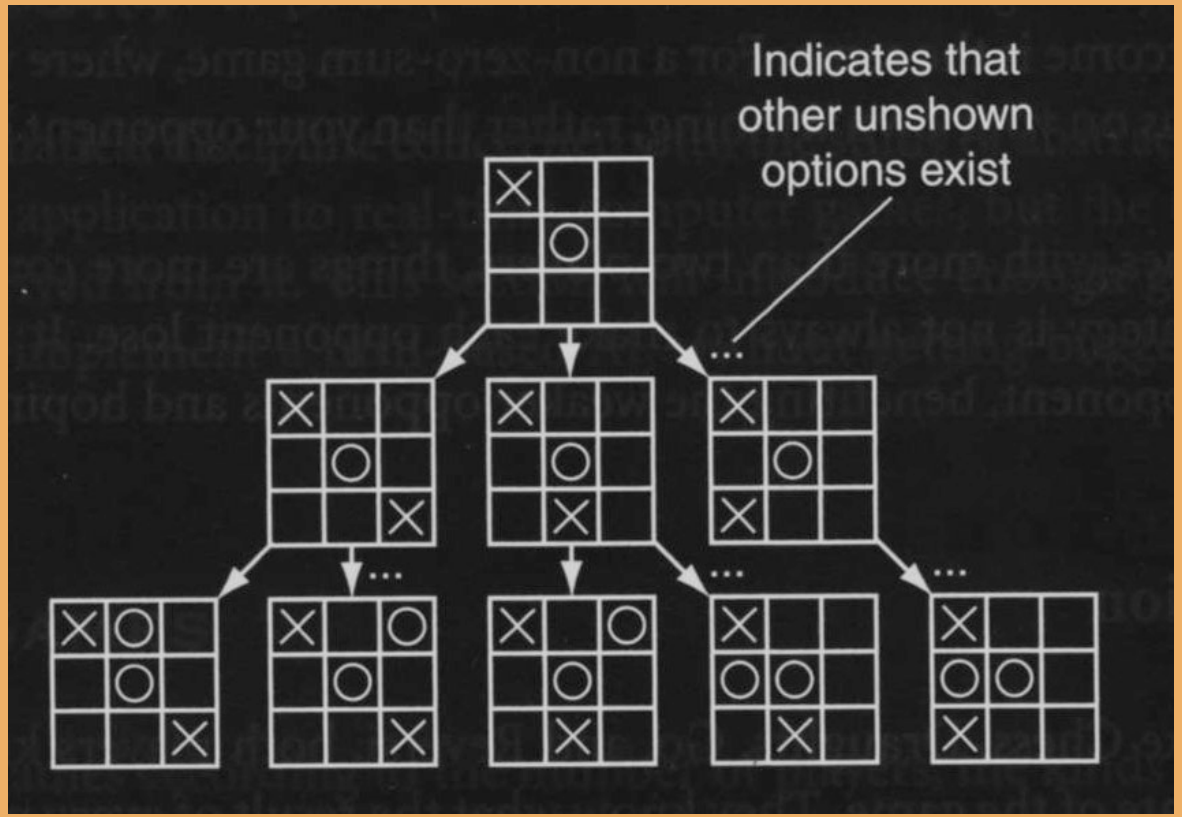


Blue player can force a win in this game



AI for Games, Millington & Funge: Figure 8.3


tie!
t-i-e.



AI for Games, Millington & Funge: Figure 8.1

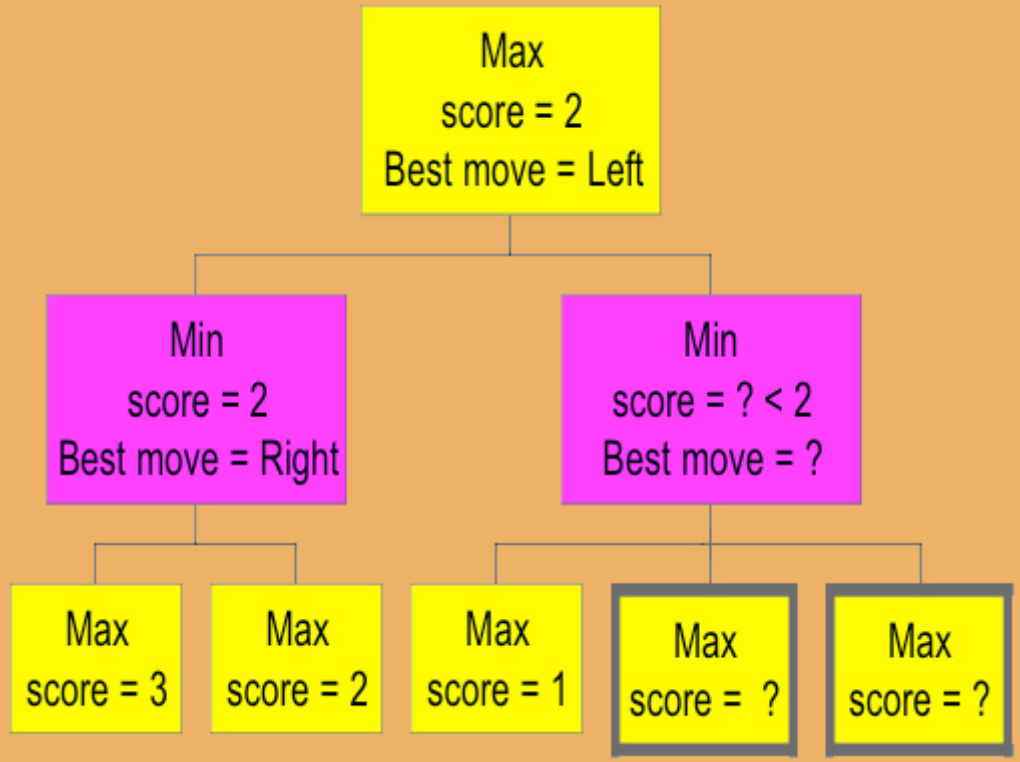
Problems!

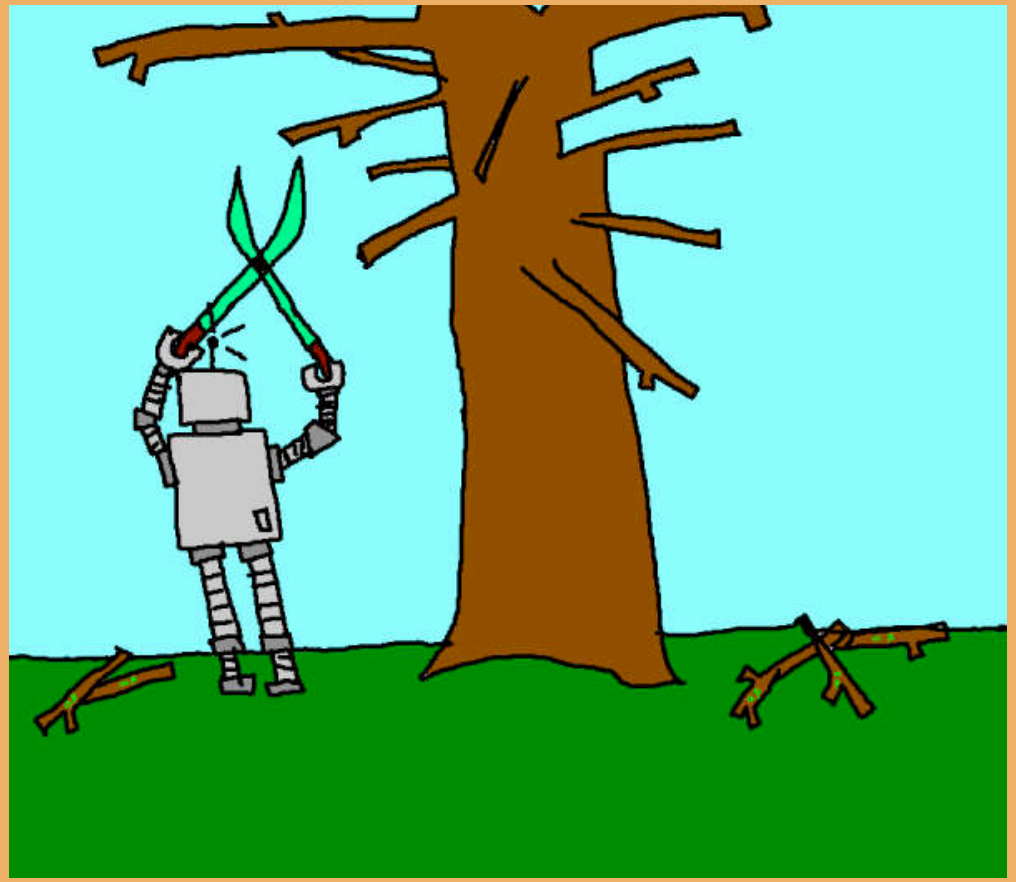
- Tic Tac Toe - 9! or 362,880 states
- Checkers about 10^{40} states
- Chess about 10^{120} states
- Go - 361! or about 10^{750} states!!!!!!

MiniMax Search

- In interesting games, state space is large - too large to reach terminal states
- Use heuristic evaluation of partial paths
- Use estimated cost of internal nodes
- Deeper search gets 'closer' to terminal states
- Also, problem is often an online search

Doing Extra Work!



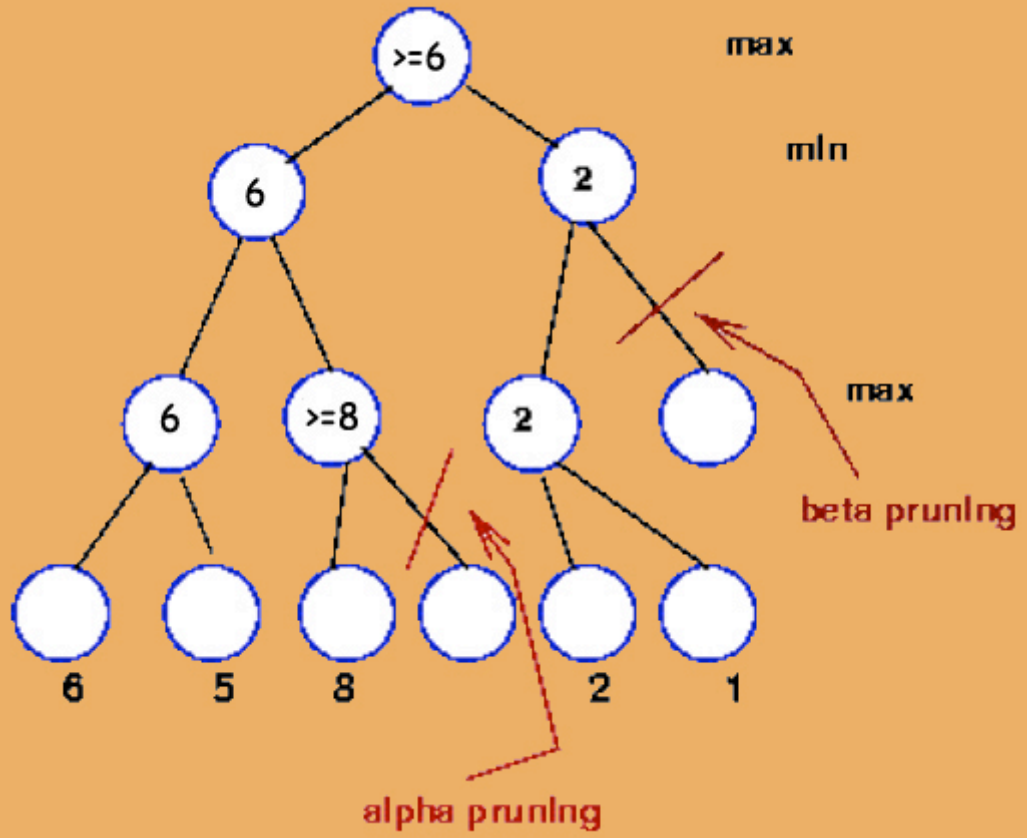


Alpha-Beta Pruning

- Start propagating costs as soon as a leaf node is generated.
- Don't explore nodes which we know are no better than best found so far.

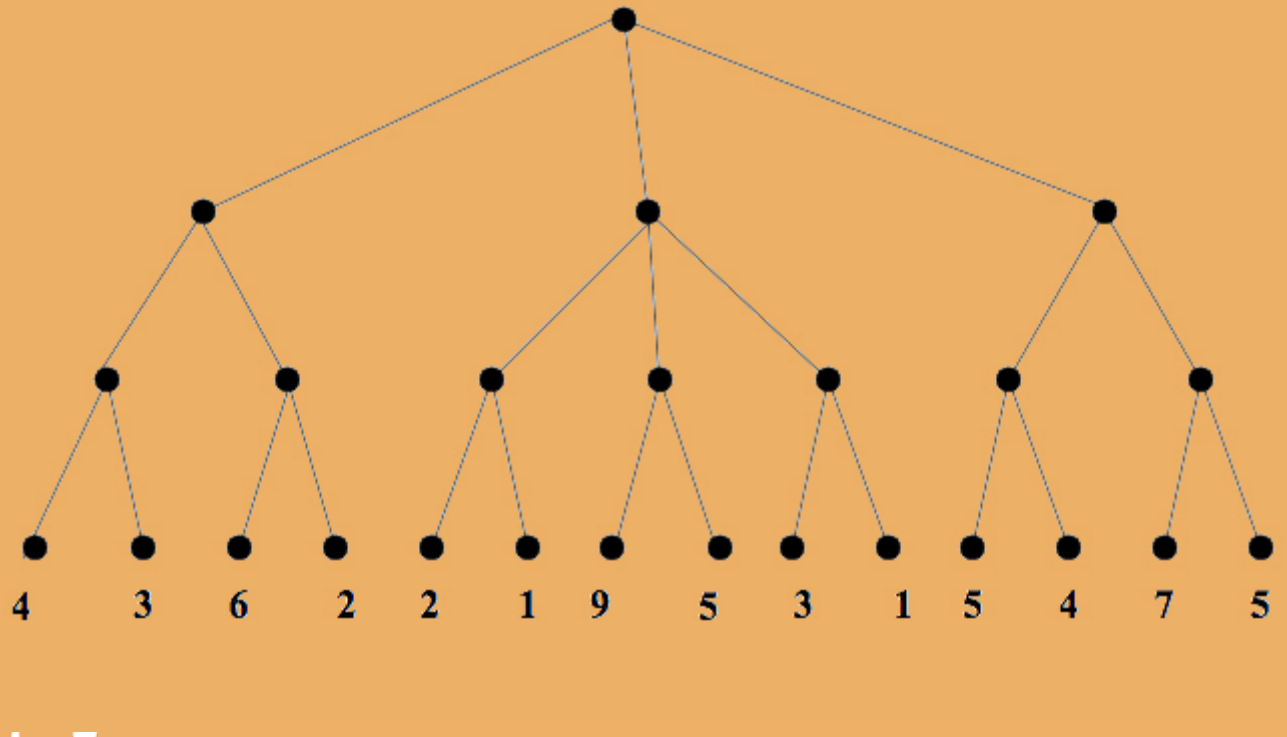
Alpha Beta Values

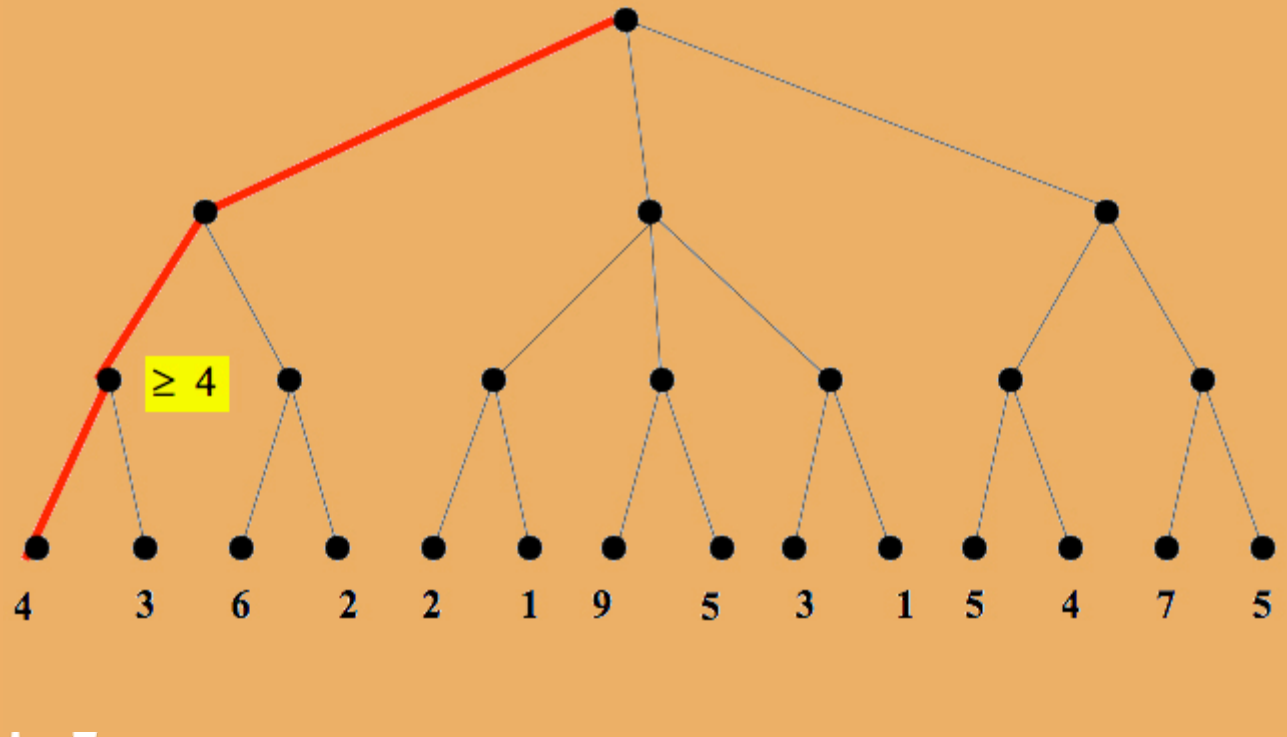
- Alpha value - associated with Max, best score found, never decreases.
- Beta value - associated with Min, best score found, never increases.
- Each used to prune tree

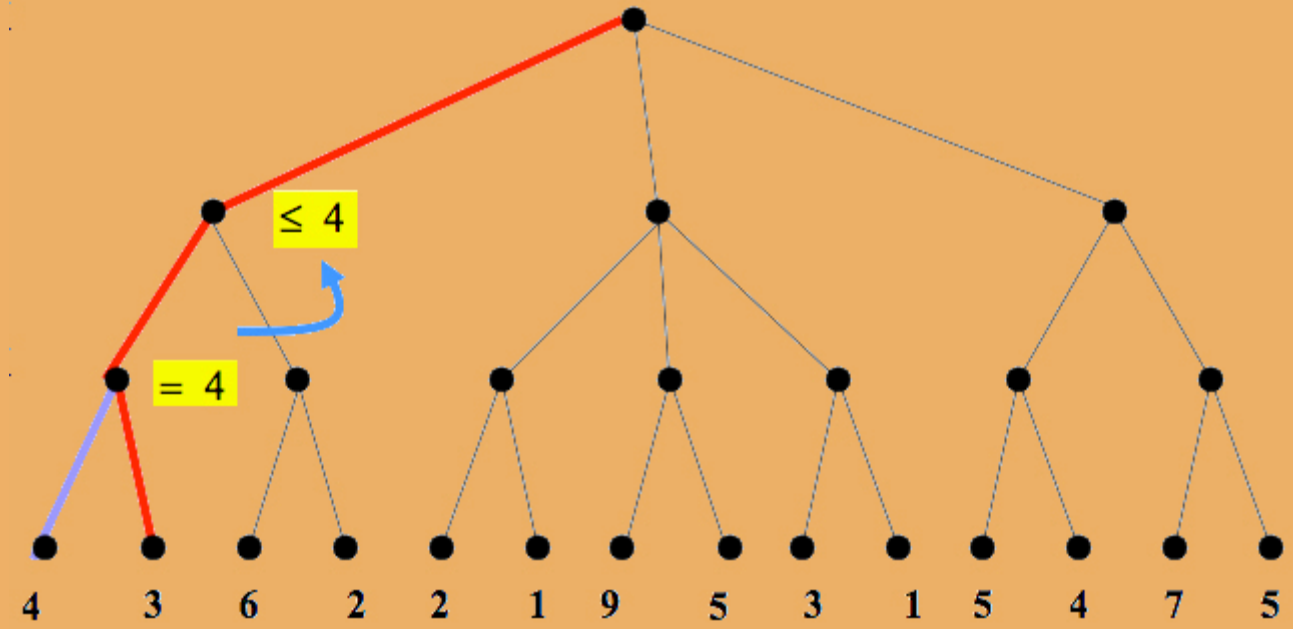

>=6
max
min
6
2
6
>=8
2
max
beta pruning
6
5
8
2
1
alpha pruning

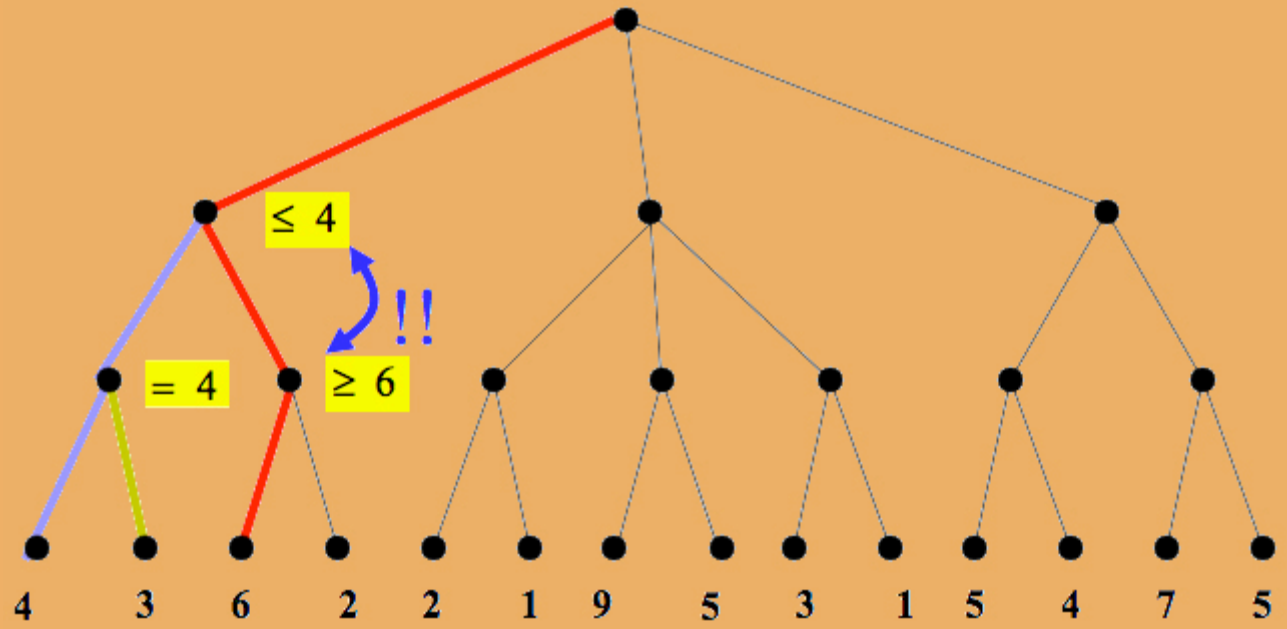
Yet Another Problem!

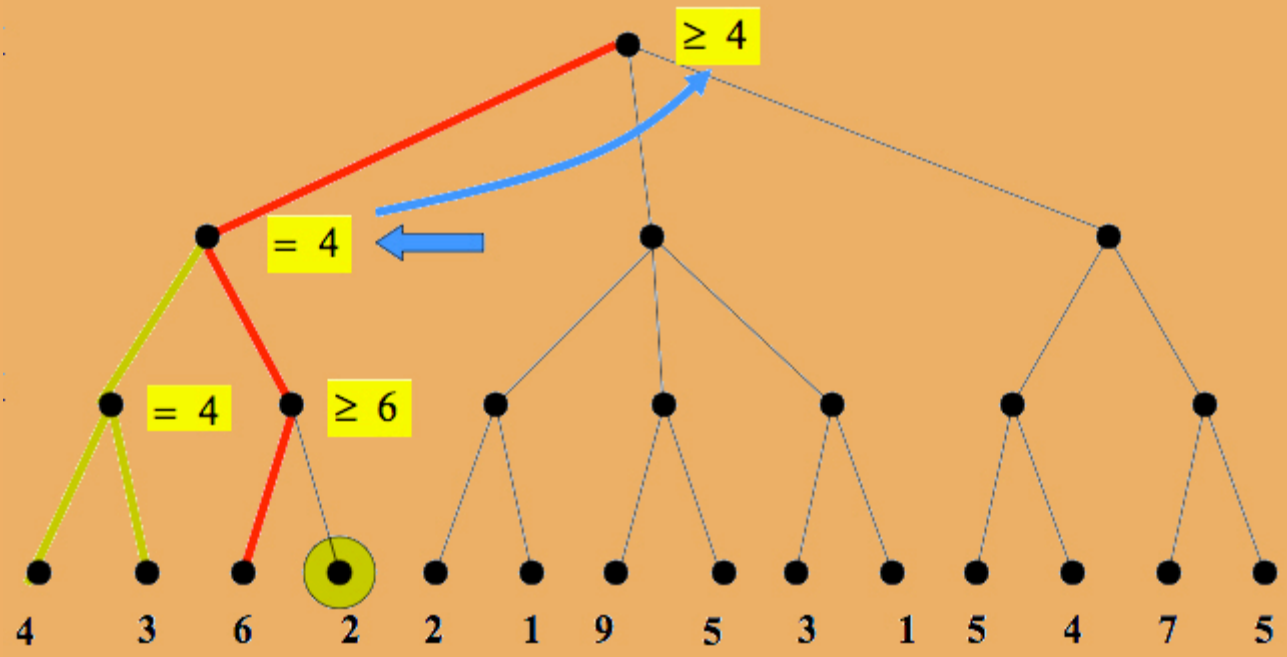
- limited look ahead leads to horizon problem
  - bad move is just beyond search depth
- solution - expand promising nodes to greater depth
- expand until quiescence - no drastic changes occur

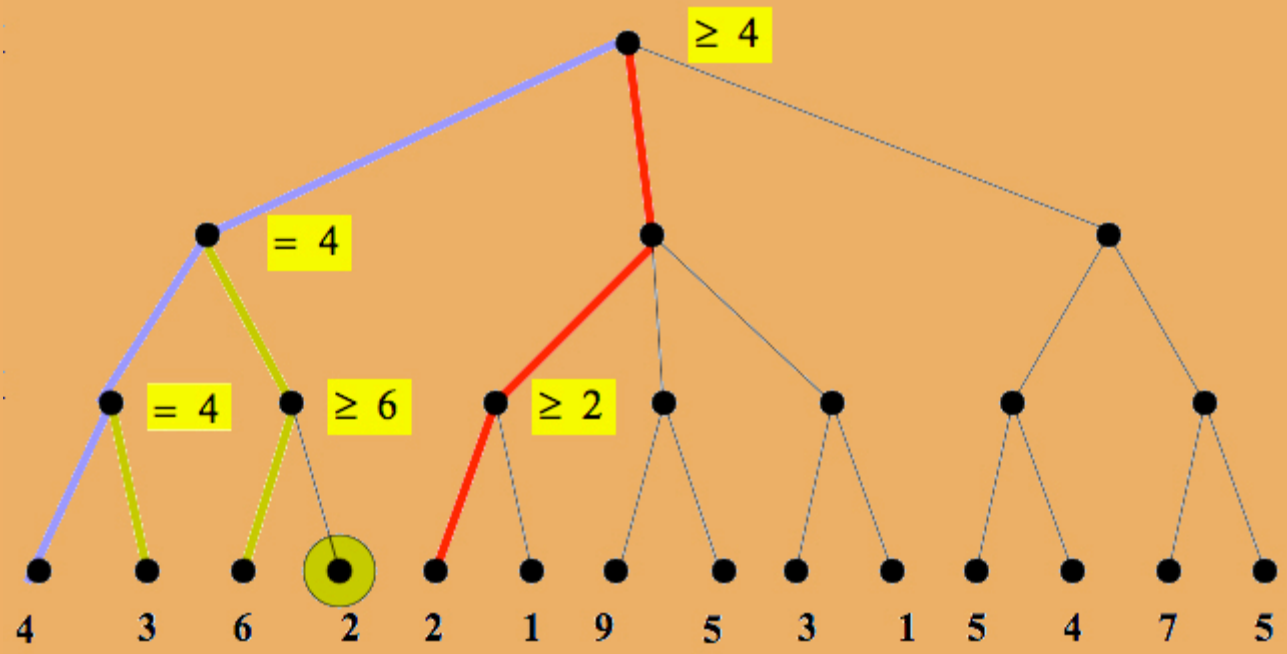

4 3 6 2 2 1 9 5 3 1 5 4 7 5

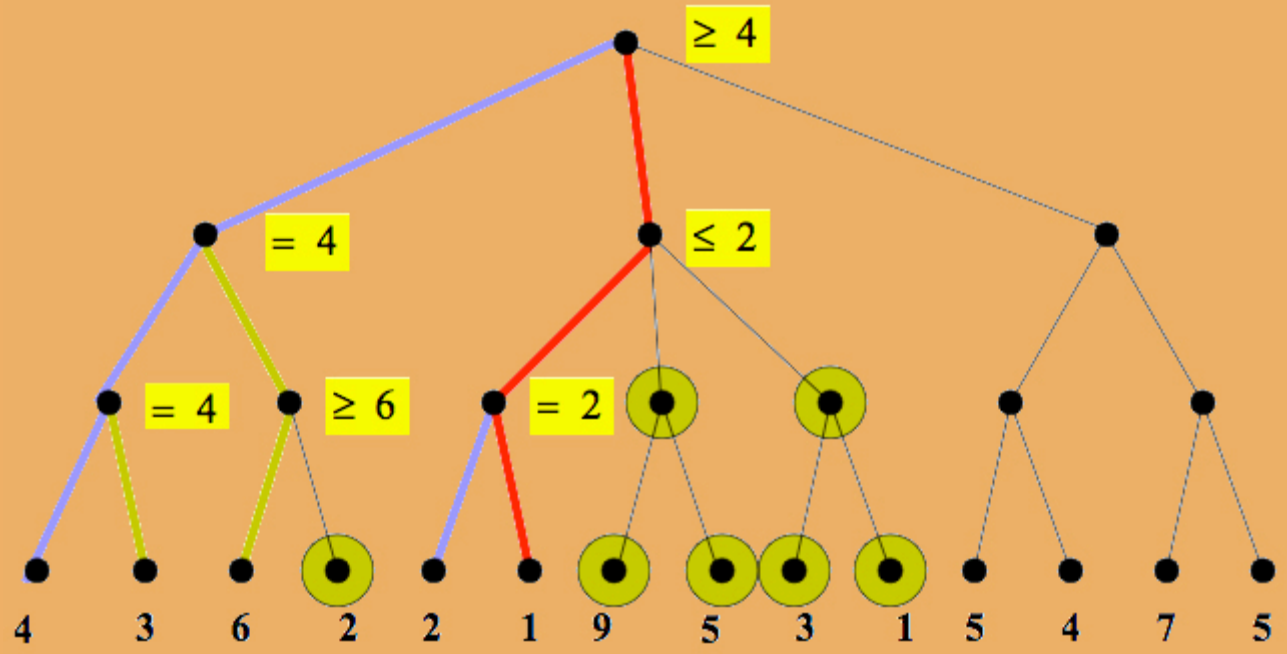

≥ 4
4 3 6 2 2 1 9 5 3 1 5 4 7 5

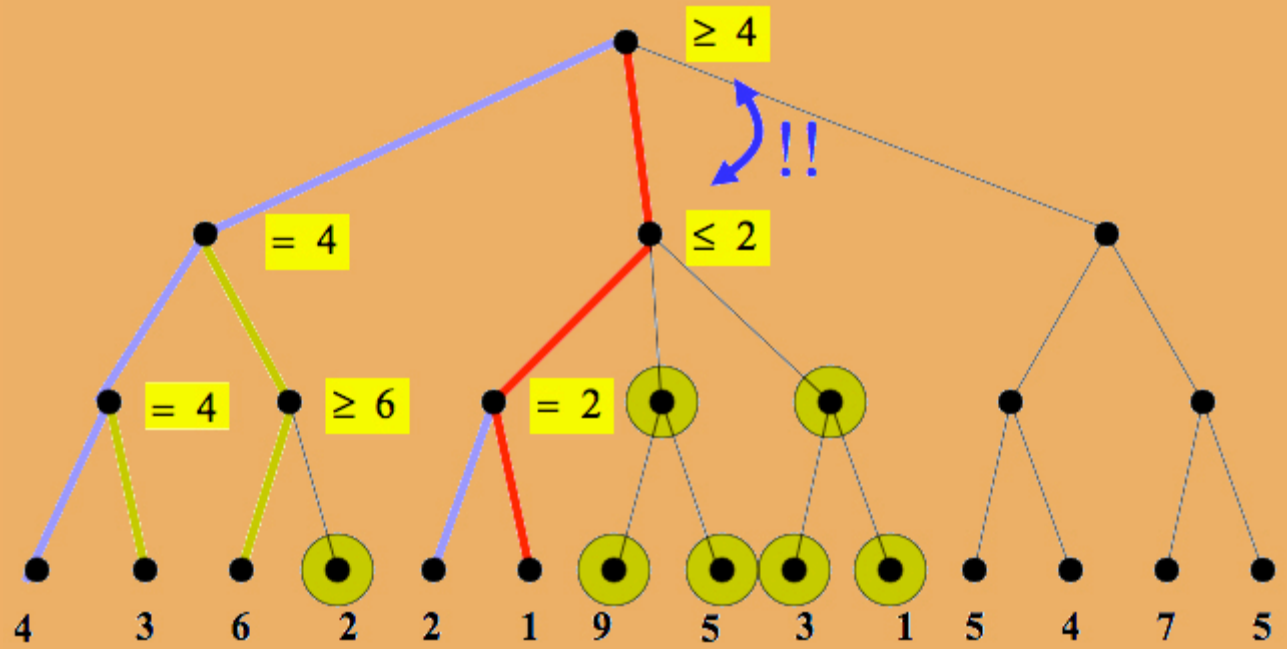

≤ 4
= 4
4 3 6 2 2 1 9 5 3 1 5 4 7 5

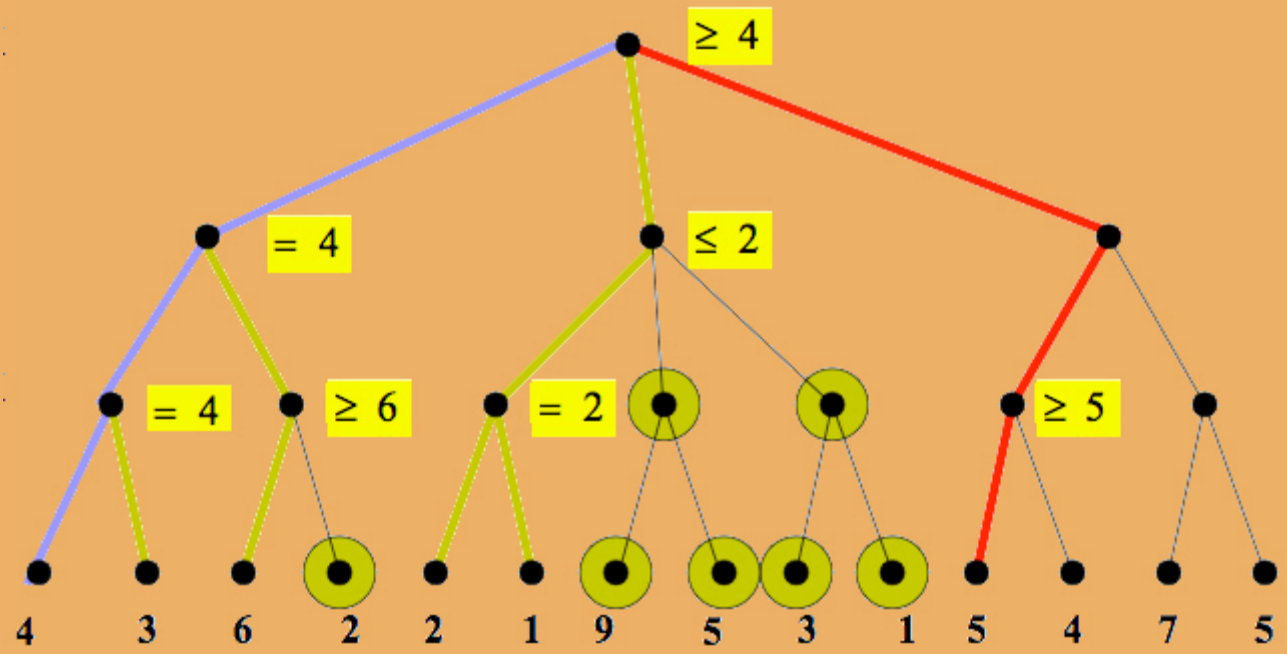

≤ 4
!!
= 4
≥ 6
4 3 6 2 2 1 9 5 3 1 5 4 7 5

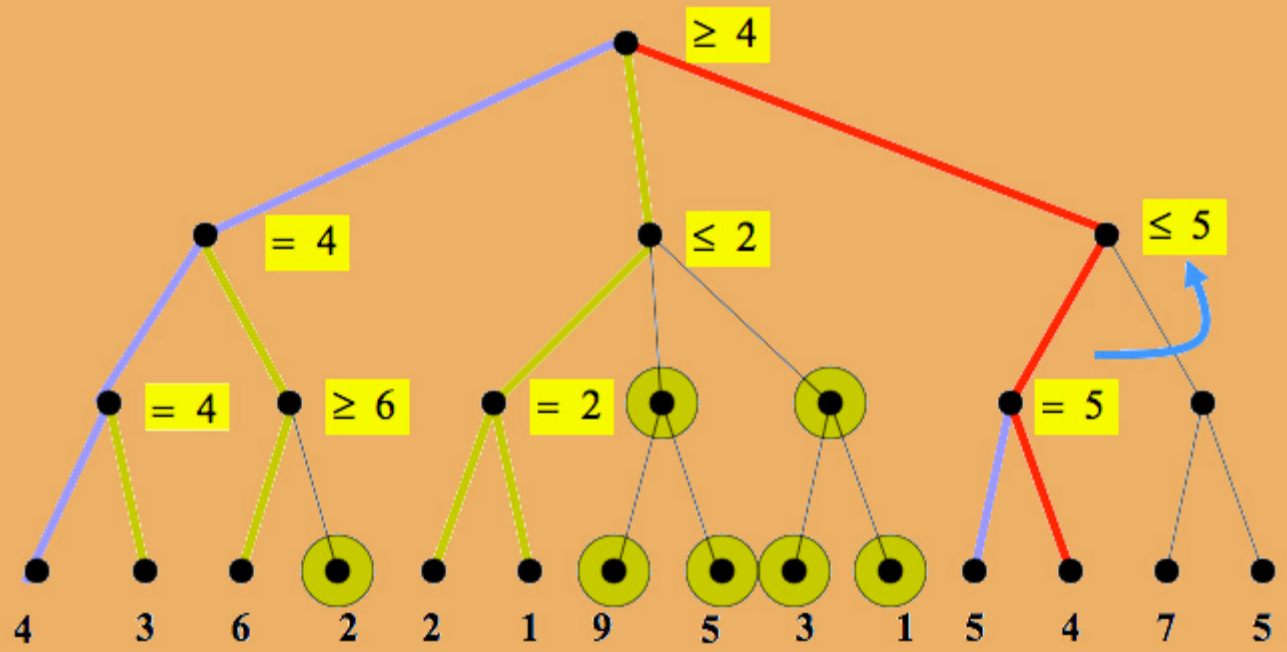

≥ 4
= 4
= 4
≥ 6
4 3 6 2 2 1 9 5 3 1 5 4 7 5

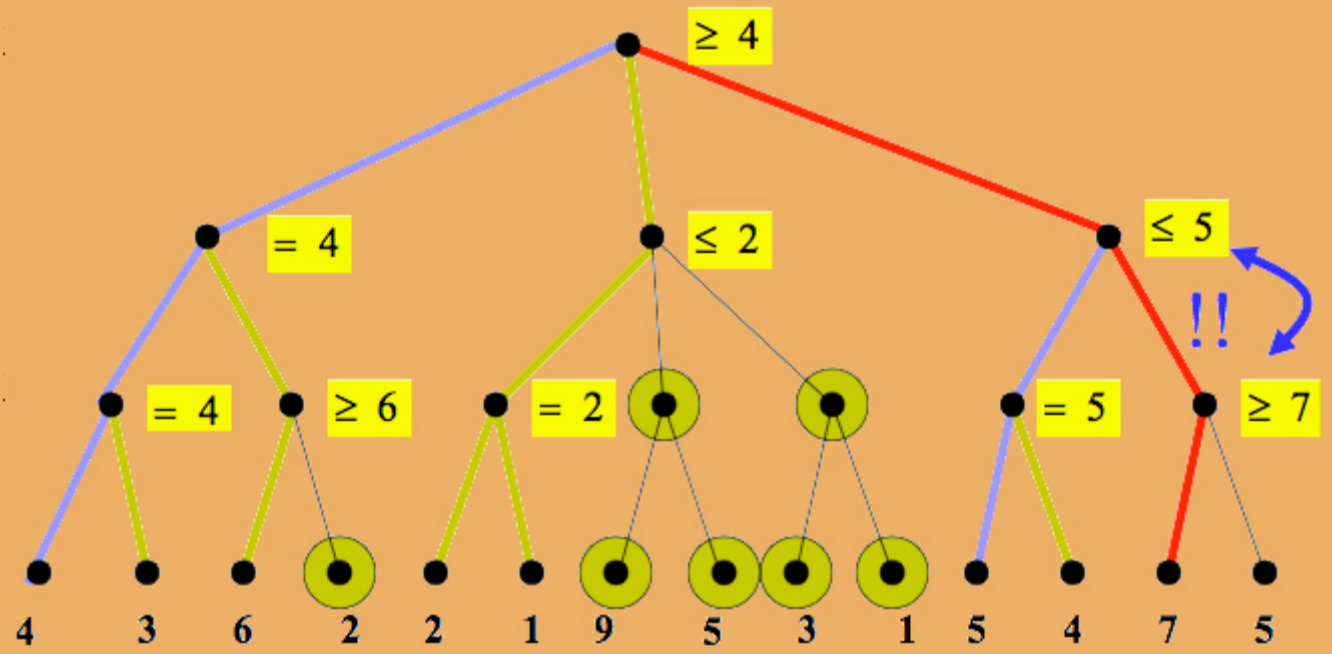

≥ 4
= 4
= 4
≥ 6
≥ 2
4 3 6 2 2 1 9 5 3 1 5 4 7 5

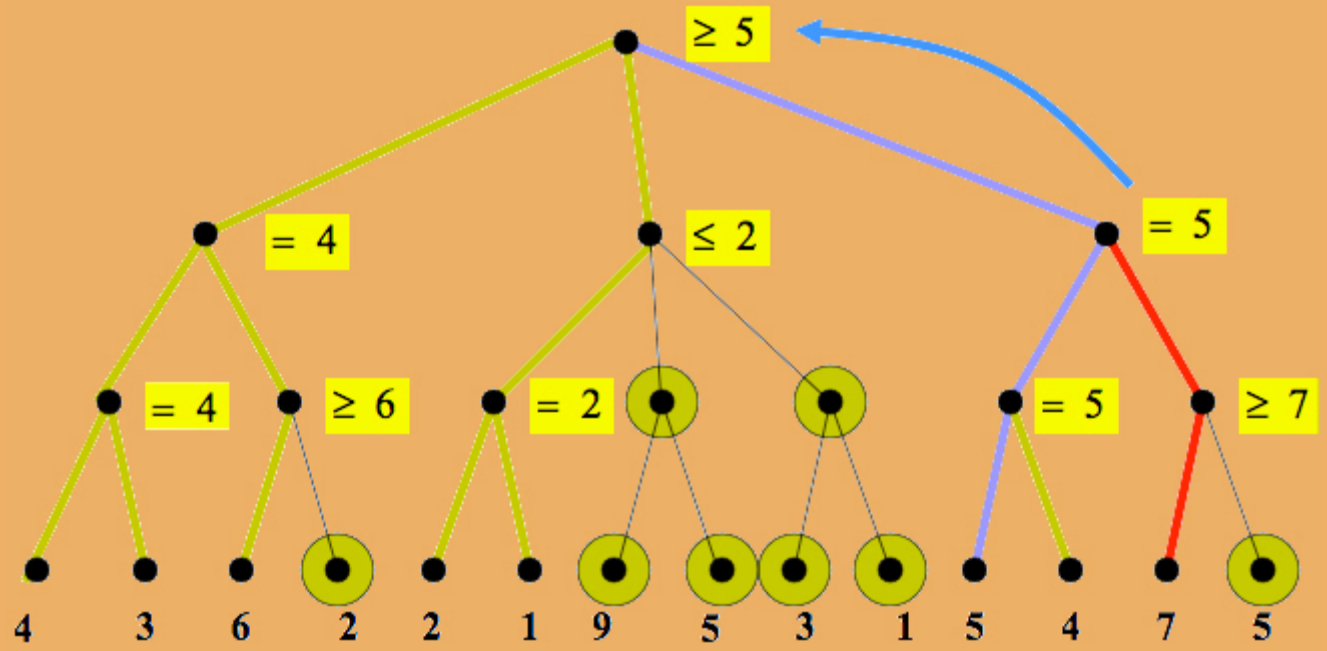

≥ 4
= 4
≤ 2
= 4
≥ 6
= 2
4 3 6 2 2 1 9 5 3 1 5 4 7 5

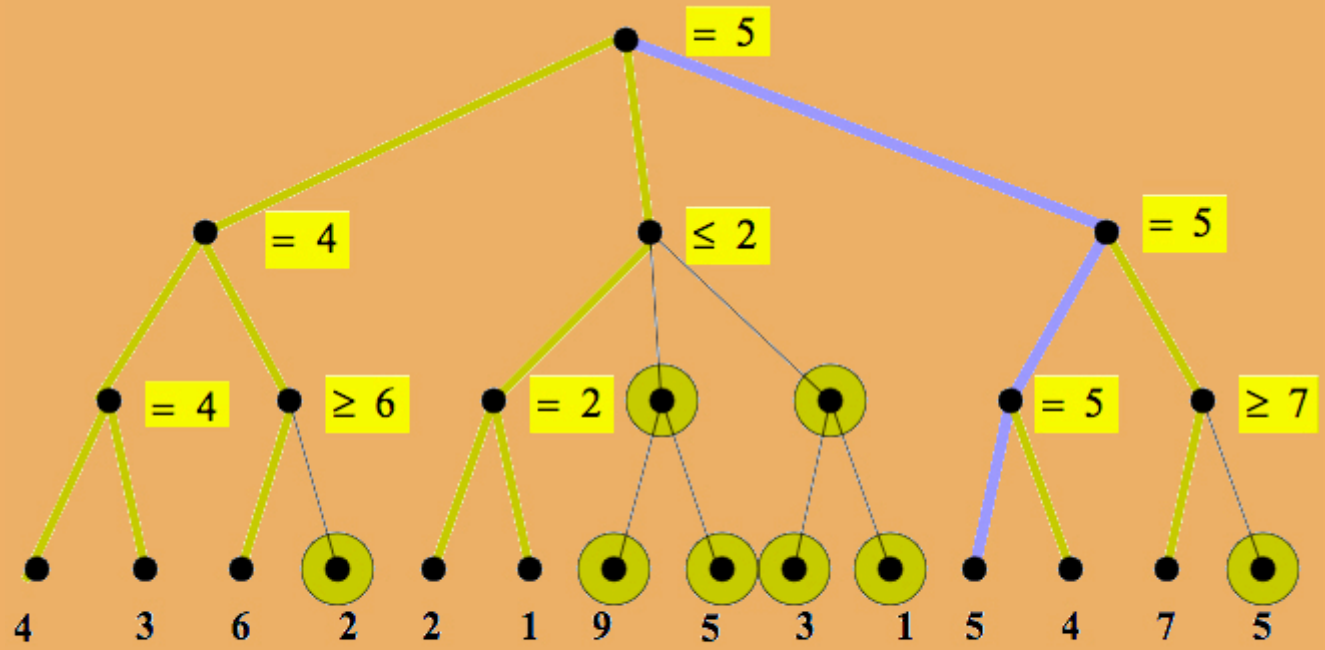

≥ 4
!!
= 4
≤ 2
= 4
≥ 6
= 2
4 3 6 2 2 1 9 5 3 1 5 4 7 5


≥ 4
= 4
≤ 2
= 4
≥ 6
= 2
≥ 5
4 3 6 2 2 1 9 5 3 1 5 4 7 5


≥ 4
= 4
≤ 2
≤ 5
= 4
≥ 6
= 2
= 5
4 3 6 2 2 1 9 5 3 1 5 4 7 5


≥ 4
= 4
≤ 2
≤ 5
!!
= 4
≥ 6
= 2
= 5
≥ 7
4 3 6 2 2 1 9 5 3 1 5 4 7 5


≥ 5
= 4
≤ 2
= 5
= 4
≥ 6
= 2
= 5
≥ 7
4 3 6 2 2 1 9 5 3 1 5 4 7 5



nodes that were never explored !!!

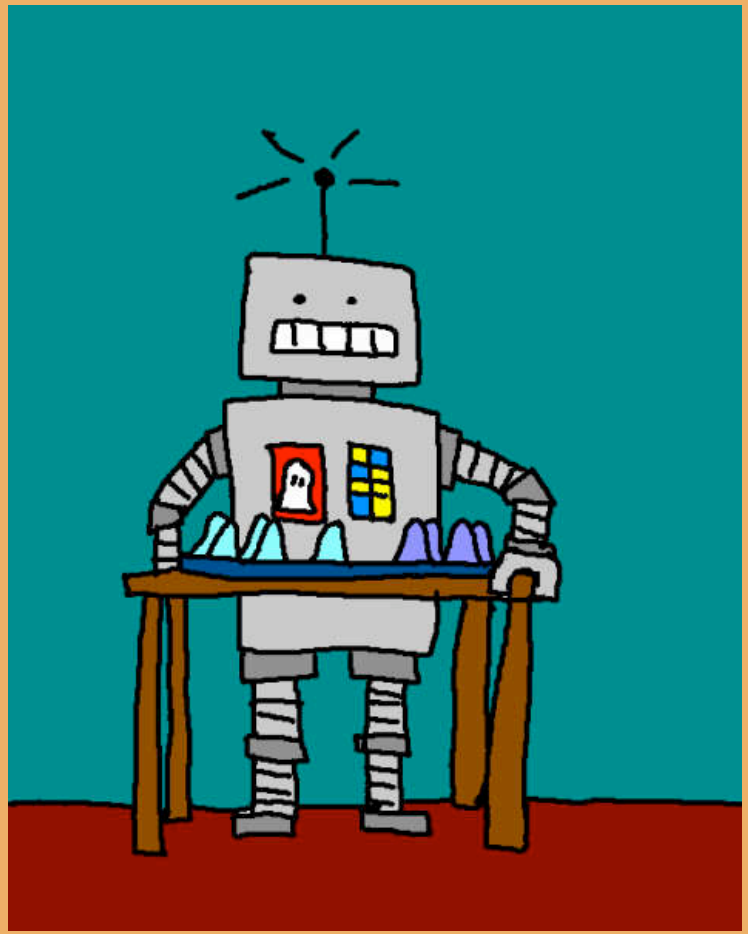
Refinements & Alternatives

- Use book moves (from library)
- Opponent may not choose optimal move
- Iterative deepening (for time constraints)

Estimated Cost - Static Evaluation

- Estimates the true value of a node
  - true value is value from exhaustive search
  - can range over a metric space
- Example: Chess
  - Pawn 1 pt, Knight/Bishop 3 pts, Rook 5 pts, Queen 9 pts.
  - Subtract the sum of each side's pieces.

Ghosts!

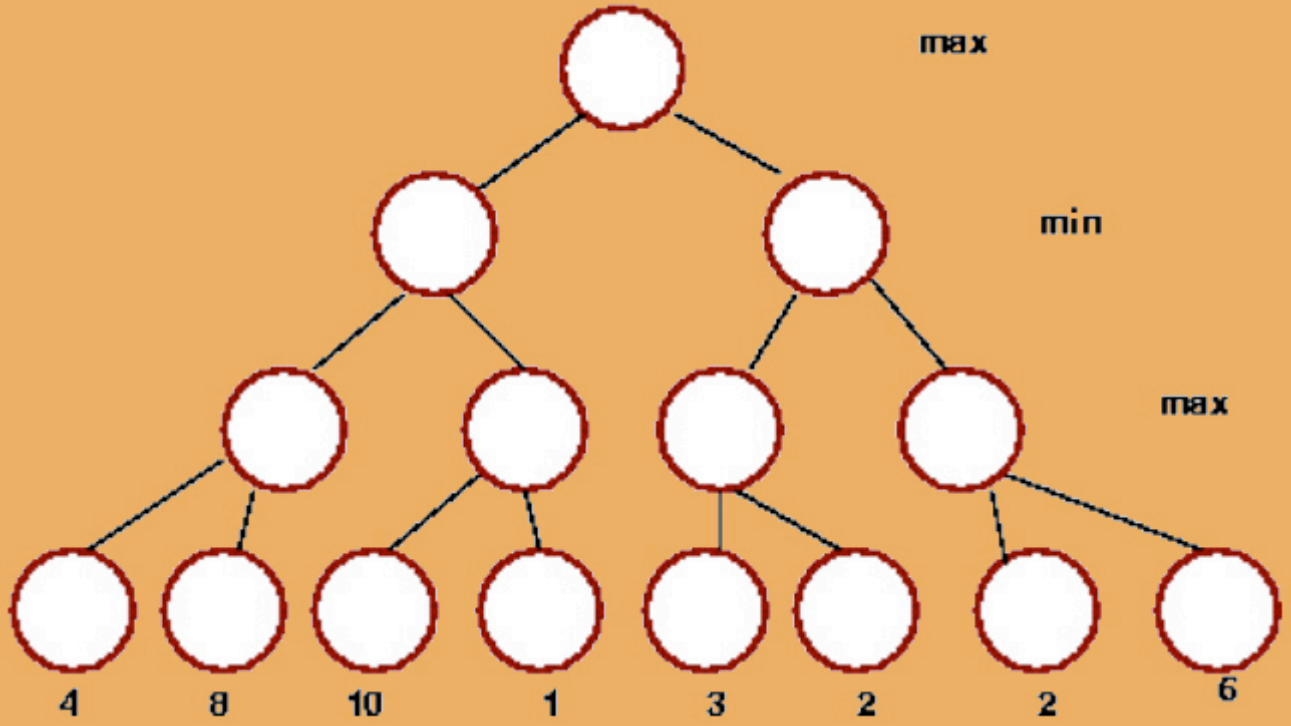


Homework

- Design a static-evaluation function for the game Ghosts!
- May involve a heuristic

Any Questions?

Study Questions



Study Questions

- Use the Alpha-Beta pruning procedure to identify which parts of the tree will be pruned.
- Let the Min player go first (e.g. reverse Max and Min levels) and reapply the Alpha-Beta procedure. How does this change things?