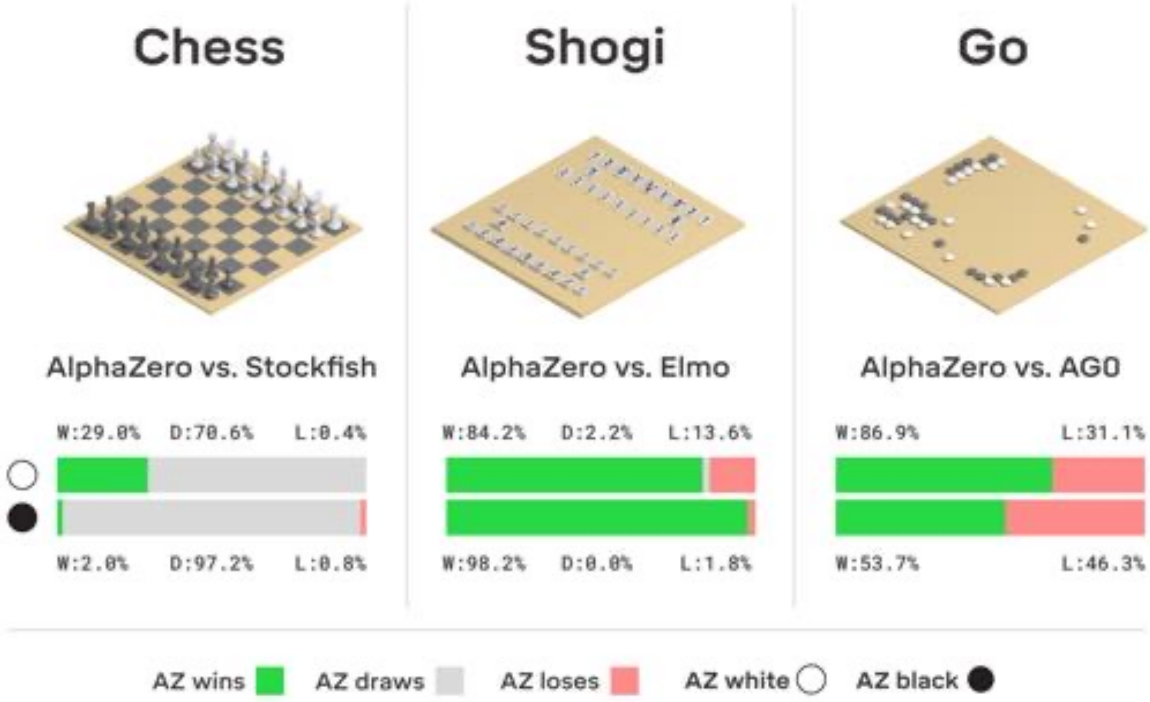
Universitat
de les Illes Balears

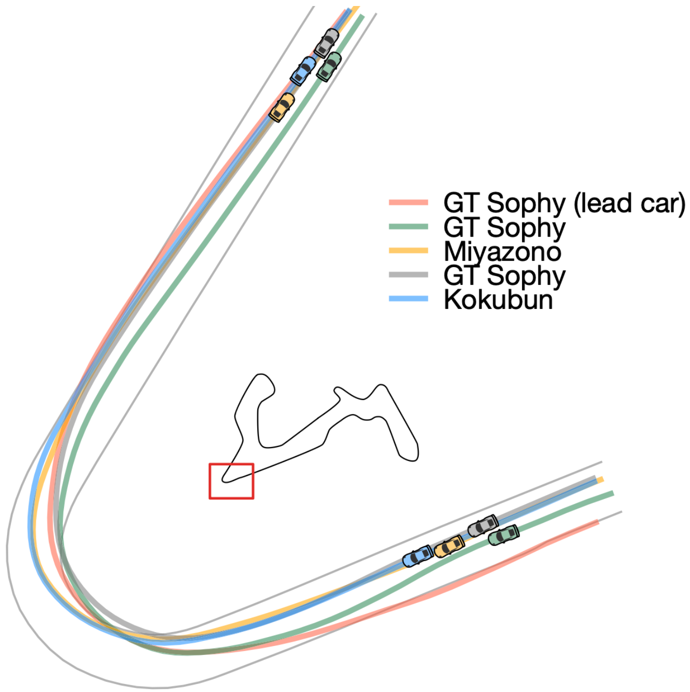
Artificial Intelligence

Lesson 3.3

Games & Adversarial Search

Google DeepMind





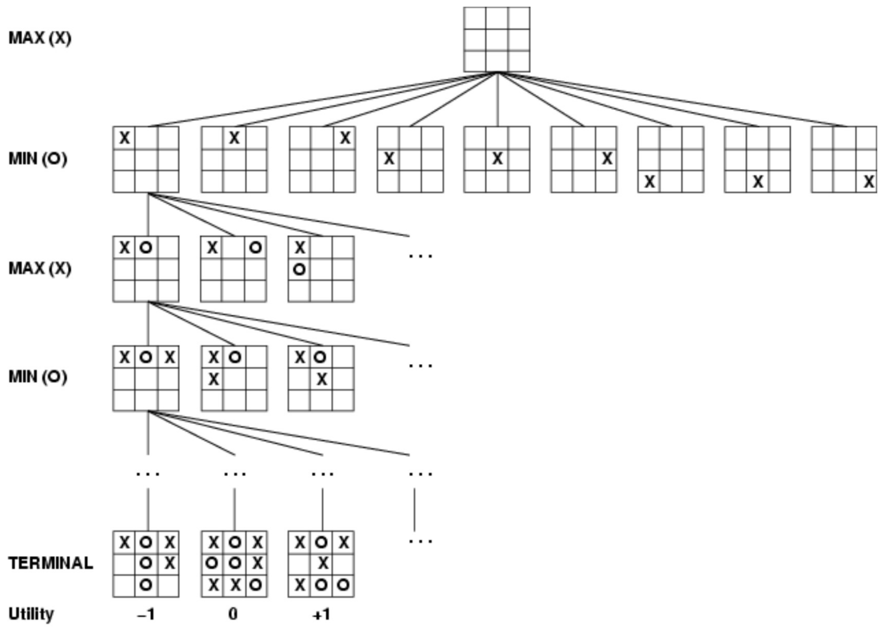
GT Sophy (lead car)
GT Sophy
Miyazono
GT Sophy
Kokubun

Games

	Deterministic	**Stochastic**
Perfect information	Go, Chess, Shogi,..	Backgammon, Monopoly,…
Imperfect information	Battleship,..	Poker,…

- Assumptions
  - Fully observable environments
  - Deterministic, turn-taking, zero-sum, perfect information
  - Non zero-sum: Prisoner’s Dilemma (Nash equilibrium)

Idea: Game tree



How do we search this tree to find the optimal move?

Informed search vs adversarial search

- Informed search
  - Solution is a method for finding goal
  - GBFS & A* algorithms can find a optimal solution
  - Heuristic: estimate cost from start to goal through a given node

- Adversarial search
  - Solution is a strategy (specifies move for every possible opponent reply)
  - Time limits force an approximate solution
  - Heuristic: evaluate “goodness” of game position

Adversarial search

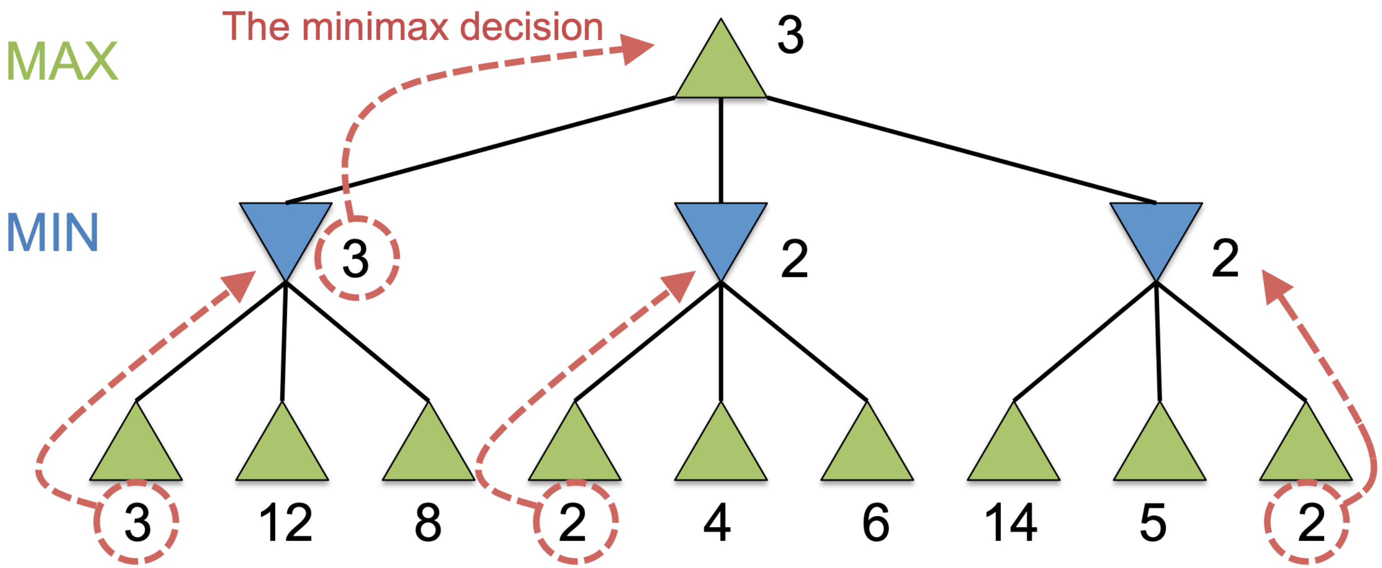
- Formal definition of a game as adversarial search
  - Two players: MAX, MIN
  - Initial state: set-up defined by rules
  - Actions(s): set of legal moves in a state
  - Terminal-Test(s): true if the game is finished; false otherwise
  - Utility(s, p): the numerical value of terminal states for player p
  - MAX uses search tree to determine "best" next move

Minimax algorithm

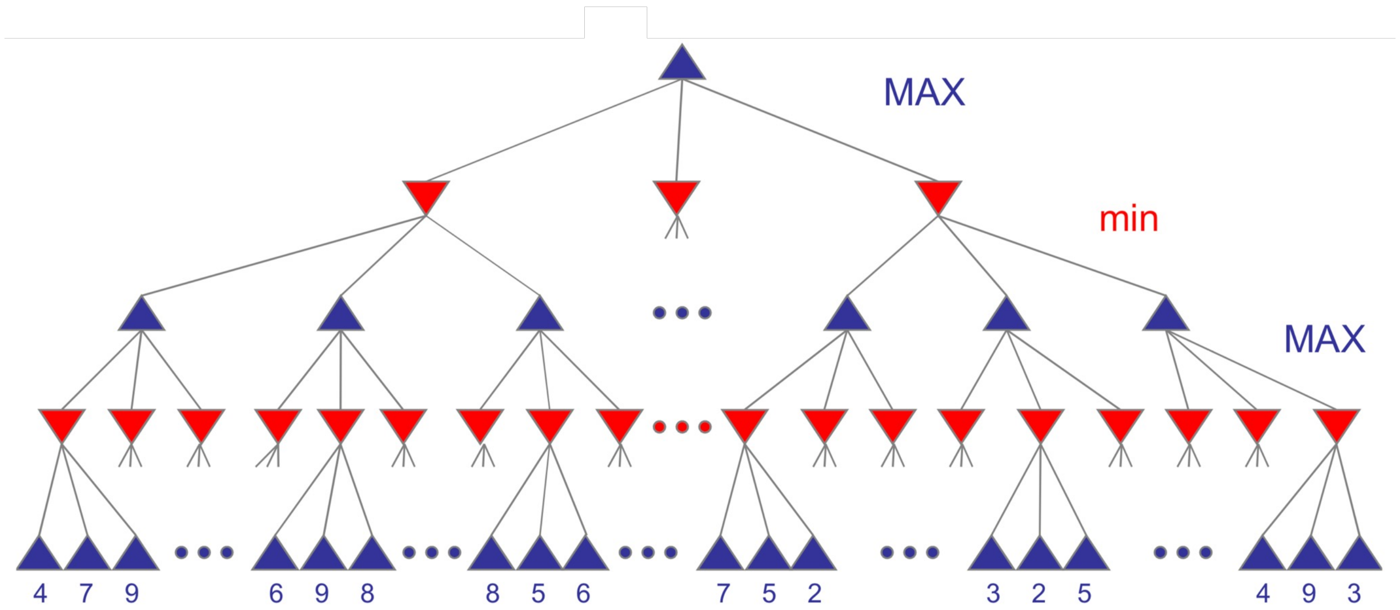
- Optimum algorithm for minimizing the possible loss for a worst case (maximum loss) scenario.

1. Generate the whole game tree to leaves
2. Apply heuristic function to leaves
3. Back-up values from leaves toward the root:
    1. a Max node computes the max of its child values
    2. a Min node computes the min of its child values
4. At root: choose move leading to the child of highest value

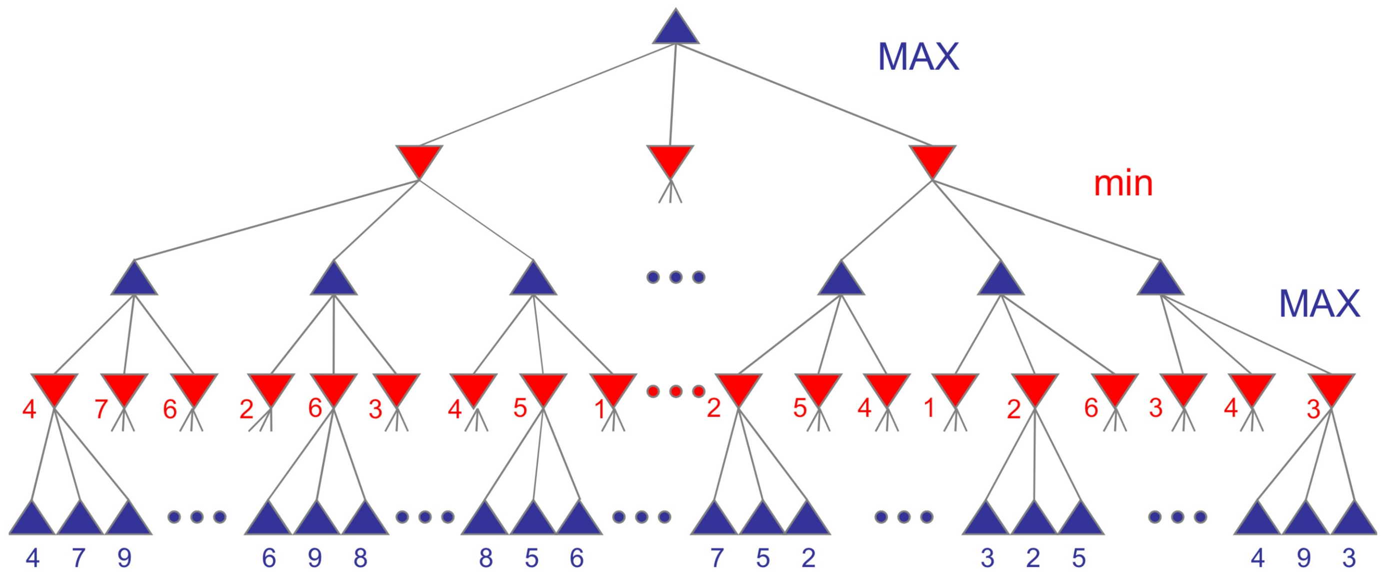
Example: Two player game tree



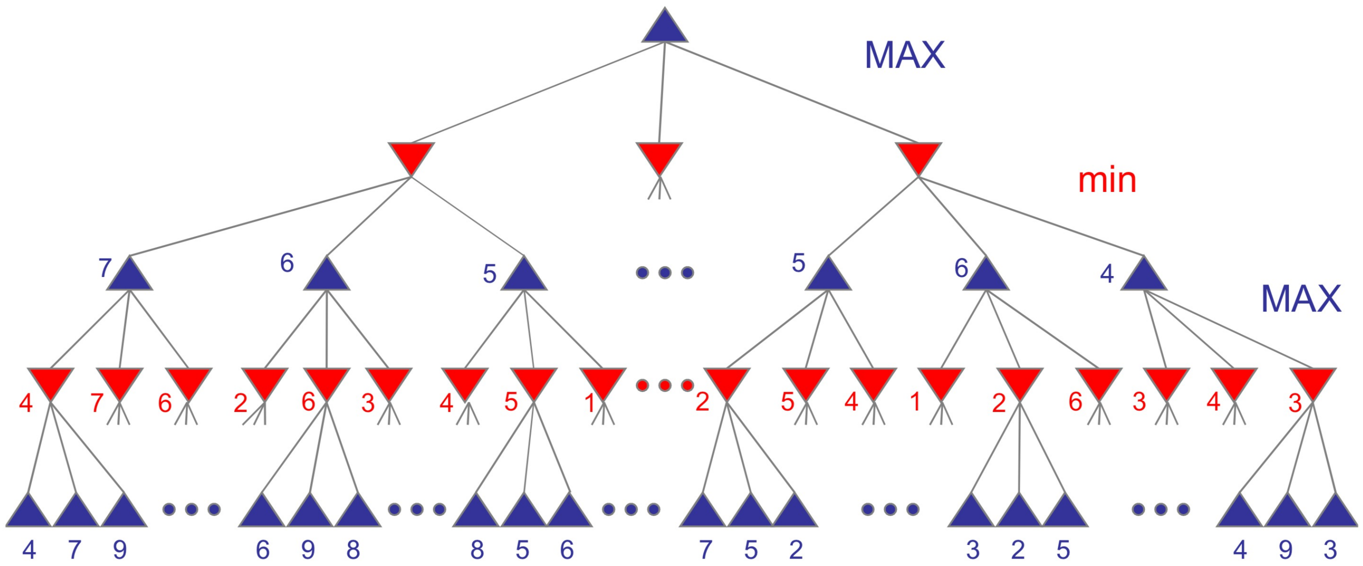
Example



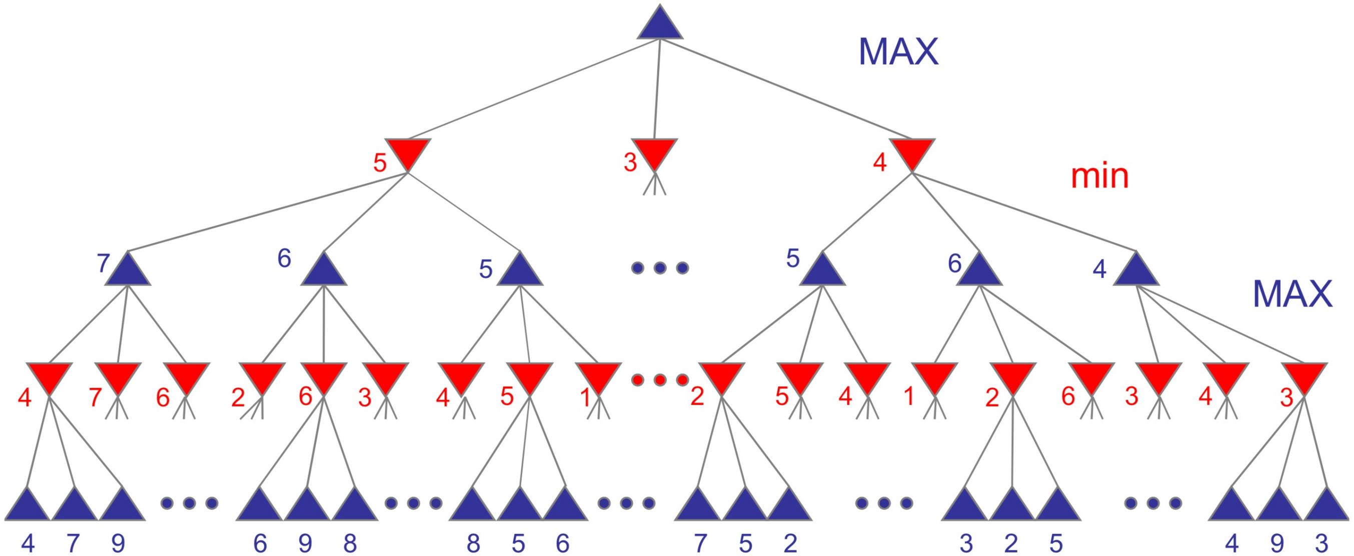
Example



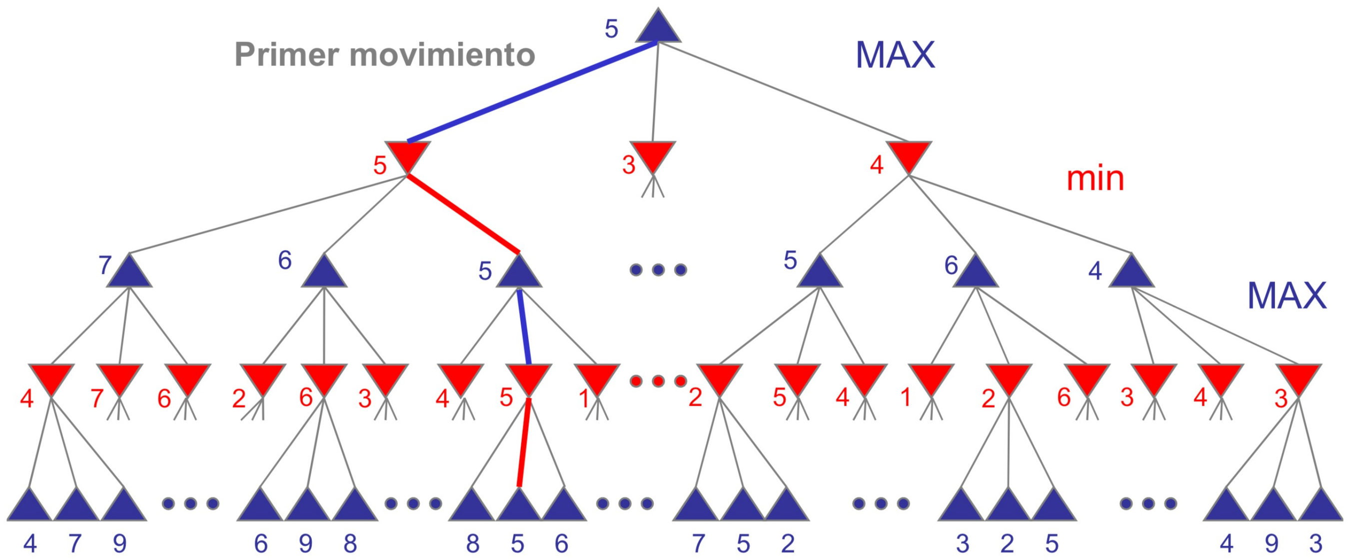
Example


MAX
min
MAX
7
6
5
5
6
4
4
7
6
2
6
3
4
5
1
2
5
4
1
2
6
3
4
3
4 7 9
6 9 8
8 5 6
7 5 2
3 2 5
4 9 3

Example

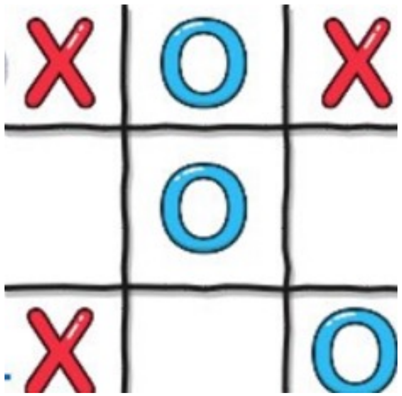

MAX
5
3
4
min
7
6
5
5
6
4
MAX
4 7 6 2 6 3 4 5 1 2 5 4 1 2 6 3 4 3
4 7 9
6 9 8
8 5 6
7 5 2
3 2 5
4 9 3

Example

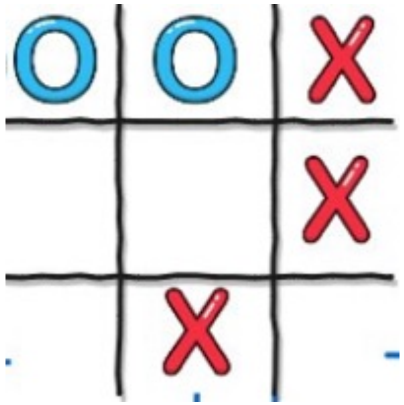


Primer movimiento
MAX
min
MAX
5
5 3 4
7 6 5 5 6 4
4 7 6 2 6 3 4 5 1 2 5 4 1 2 6 3 4 3
4 7 9 6 9 8 8 5 6 7 5 2 3 2 5 4 9 3

Exercise



Best movement for X Player… ?



Who wins?

Game tree size

- Tic-tac-toe
  - Approx branching factor: 5
  - Approx depth: 9
  - $b^d = 5^9$, exact solution is quite reasonable

- Chess
  - Approx branching factor: 35
  - Approx depth: 100
  - $b^d = 35^{100}$, exact solution is infeasible!

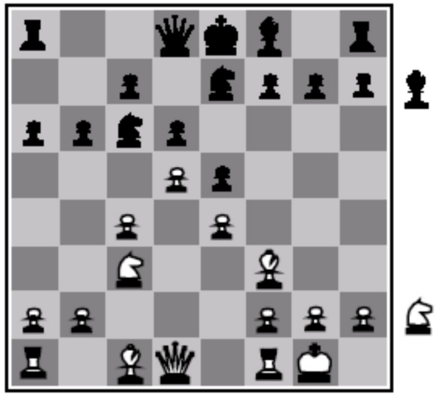
1) Limited depth

2) Alpha–beta pruning!

Static Heuristics

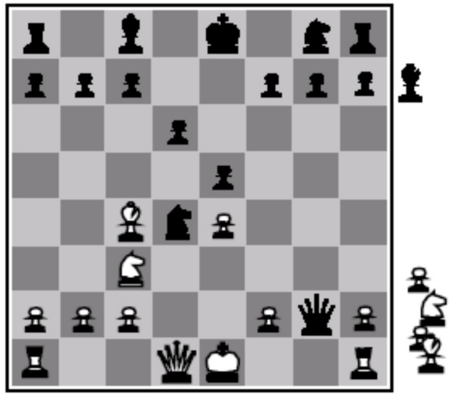
- A static heuristic for adversarial search
  - Estimate how good the current board configuration is for a player.
  - Typically, evaluate how good it is for the player, and how good it is for the opponent, and subtract the opponent's score from the player's.
  - Often called "static" because it is called on a static board position
  - Ex: Othello: Number of white pieces - Number of black pieces
  - Ex: Chess: Value of all white pieces - Value of all black pieces
  - Board evaluation: X for one player -X for opponent

Chess…



Black to move

White slightly better



White to move

Black winning

For chess, typically *linear* weighted sum of features

$$Eval(s) = w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s)$$

e.g., $w_1 = 9$ with
$f_1(s) =$ (number of white queens) – (number of black queens), etc.

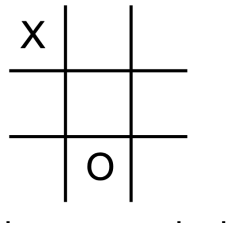
Tic-tac-toe…

- Heuristic?

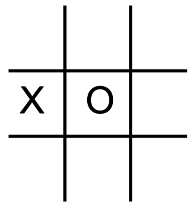
- Best first move?

Tic-tac-toe…

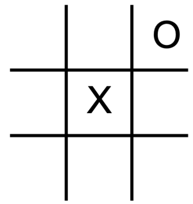
- Heuristic?
  - Number of possible win lines



H= 6-5



H= 4-6



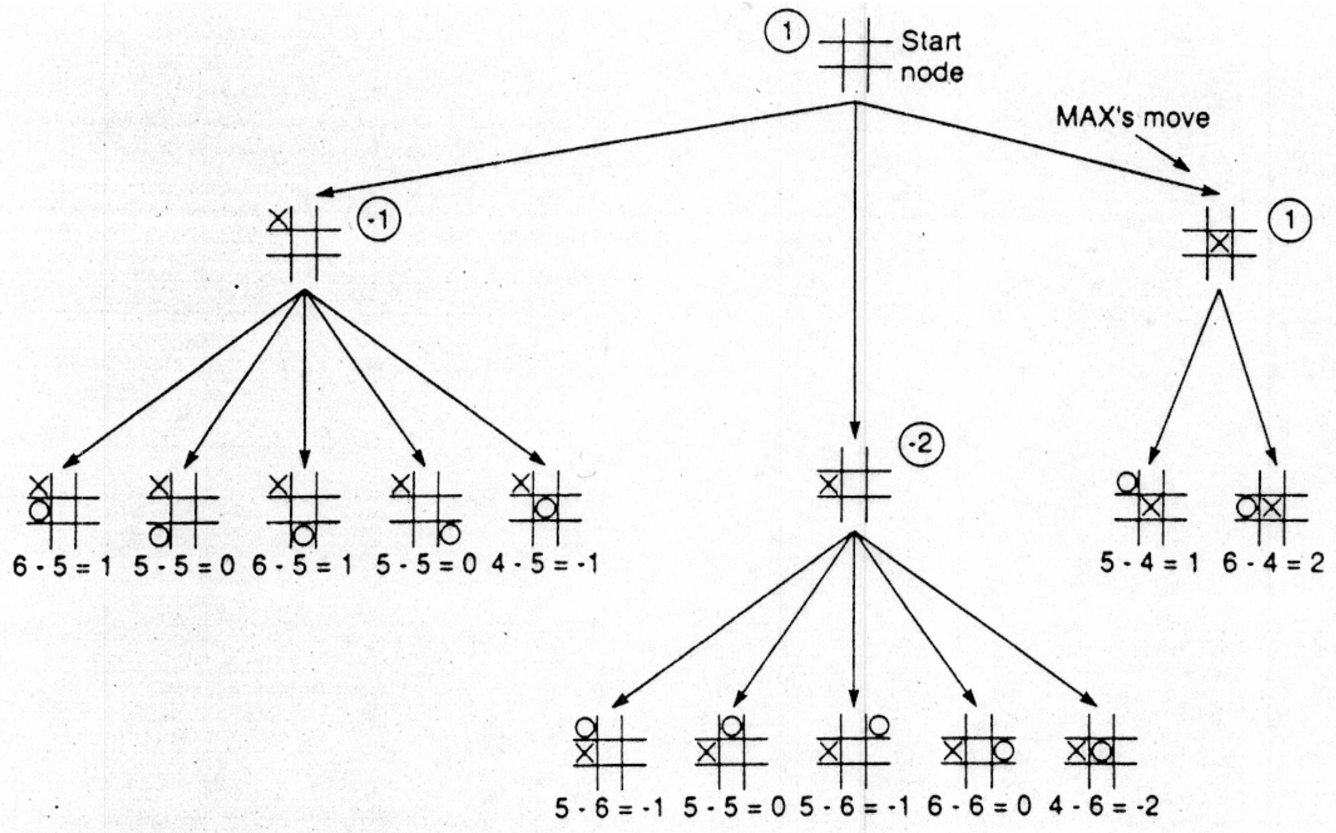
H= 5-4

- Best first move?

Tic-tac-toe…

- Heuristic?

- Best first move?



Alpha-beta pruning

- Minimax problem: exponential states
- **Idea**: it is possible to calculate the correct minimax decision without looking at all the nodes in the tree
  - If a position is provably bad
    - It's no use searching to find out just how bad
  - If the adversary can force a bad position
    - It's no use searching to find the good positions the adversary won't let you achieve

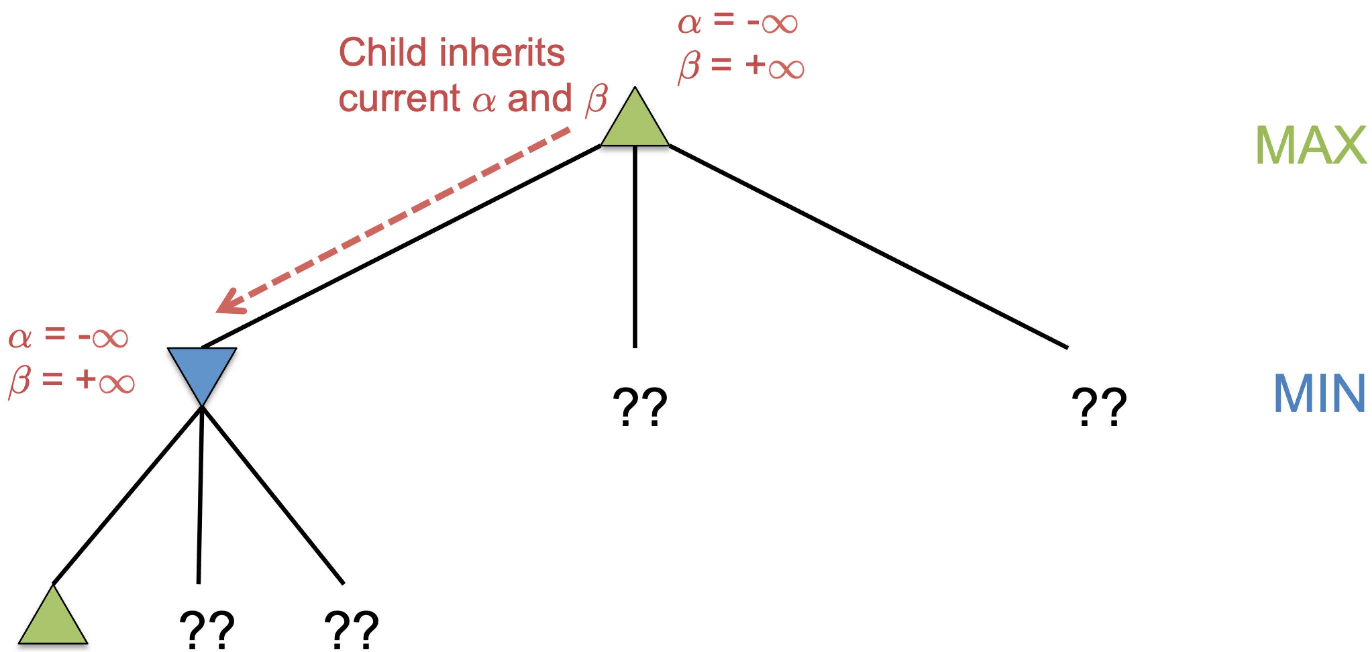
- **Alpha-beta pruning** allows you to remove large parts of the tree, without influencing the final decision
    - An example

Alpha-beta pruning

- Each node is analysed taking into account its value and the value that its parent currently has.
- This determines at each moment one interval (α, β) of possible values that the node could take
- Intuitive meaning of α and β at each moment:
  - MAX nodes: α is the current value of the node (which will have that or more) and β is the current value of the parent (which will have that or less)
  - MIN nodes: β is the current value of the node (which will have that or less) and α is the current value of the parent (which will have that or more)
- Pruning occurs if at some node α ≥ β:
  - There is no need to analyse the remaining successors of the node
  - At MIN nodes, it is called β pruning and at MAX nodes, pruning α

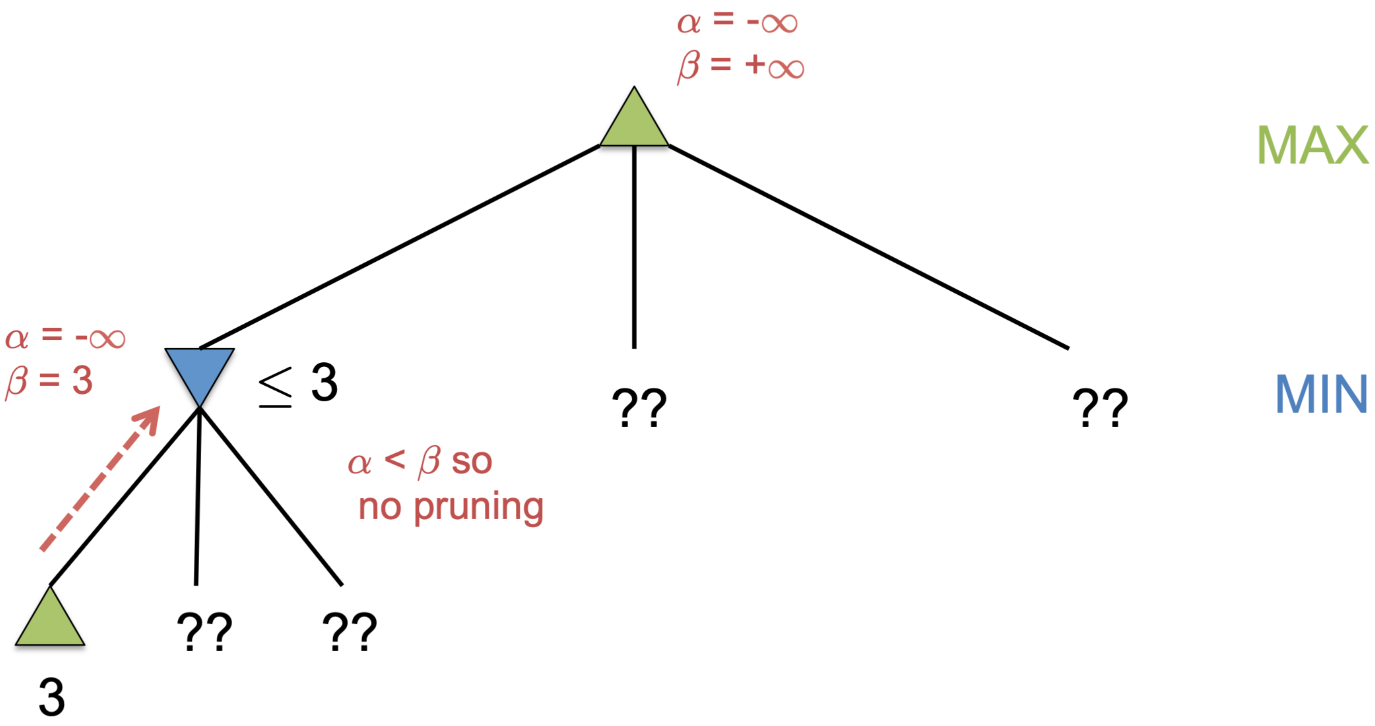
Alpha-beta pruning

- Initially, possibilities are unknown: range $(\propto = -\infty, \beta = \infty)$
- Do a depth-first search to the first leaf



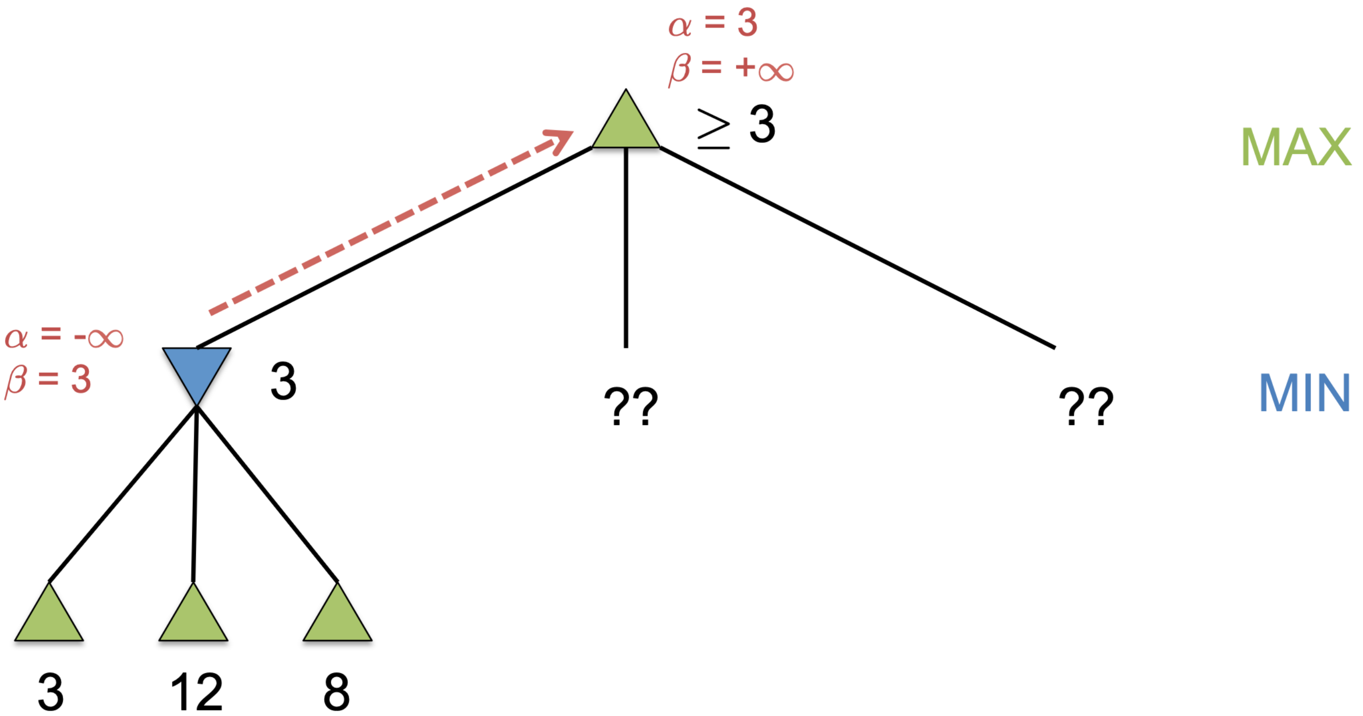
Alpha-beta pruning

- See the first leaf, after MIN's move: MIN updates β



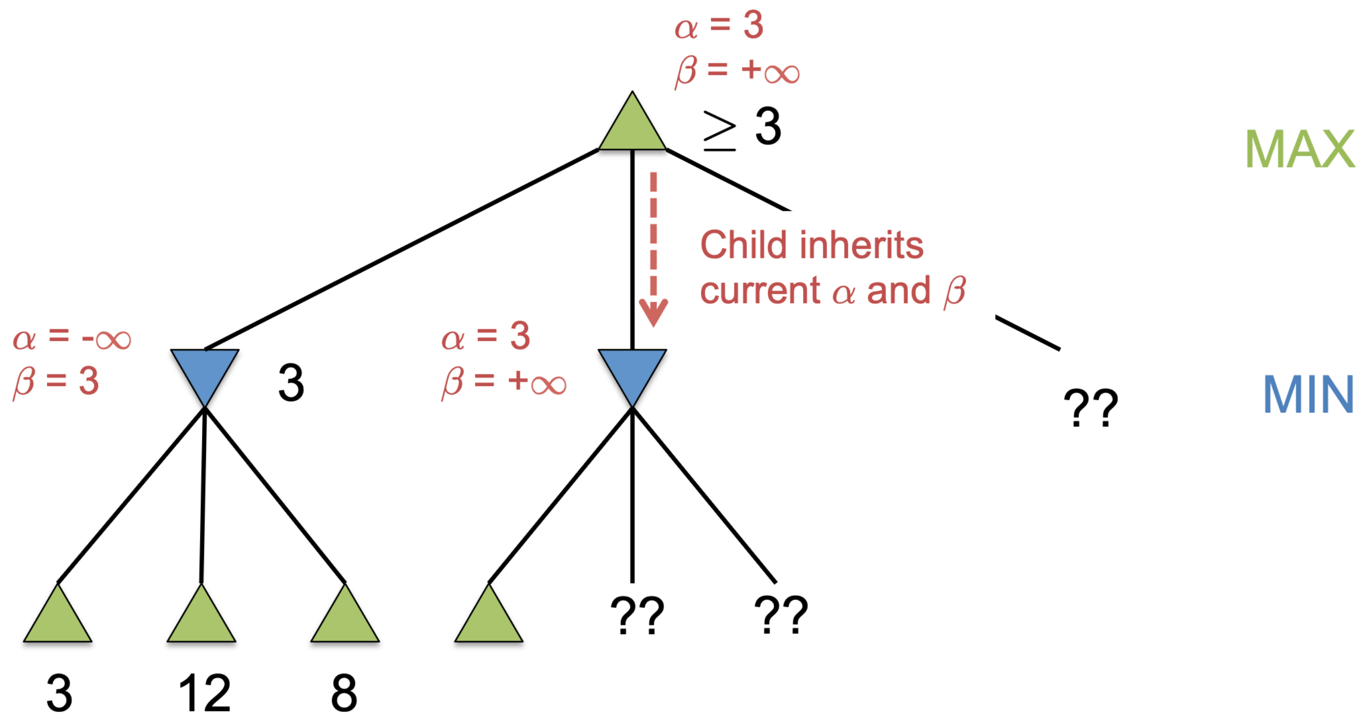
Alpha-beta pruning

- See remaining leaves; value is known
- Pass outcome to caller; MAX updates α



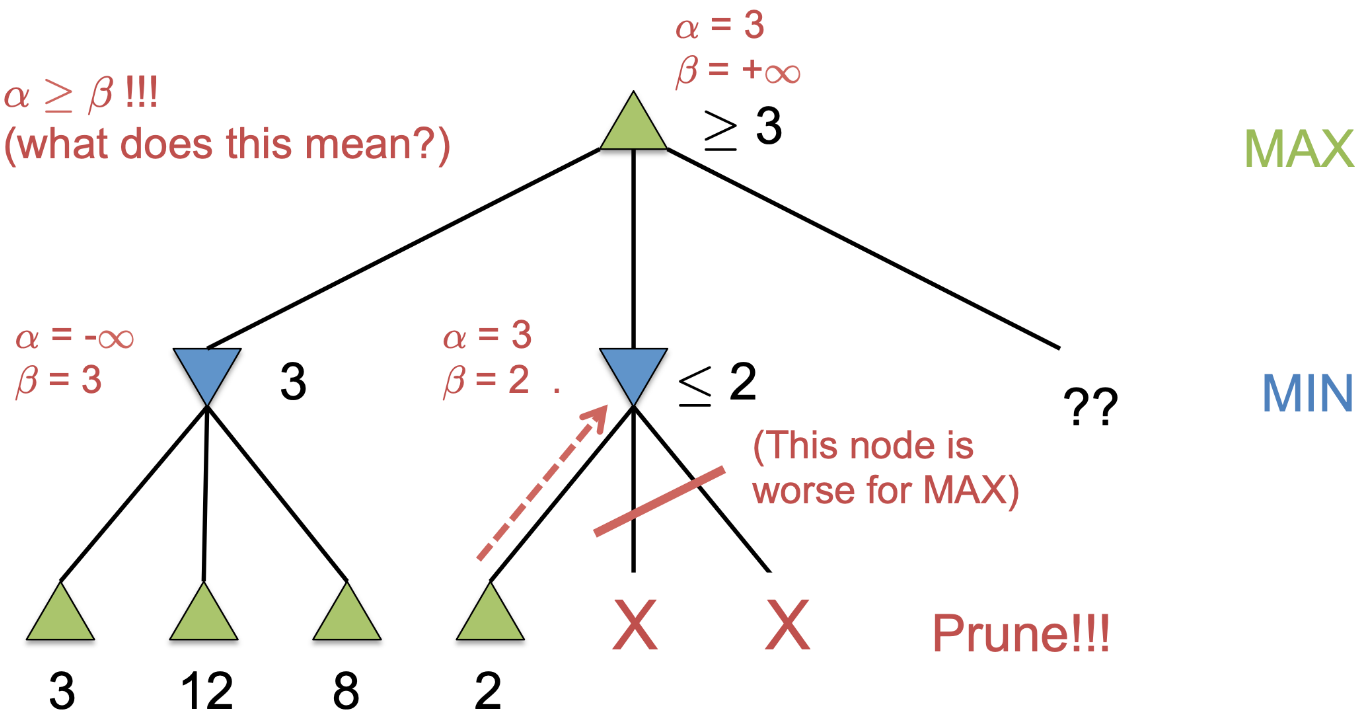
Alpha-beta pruning

- Continue depth-first search to next leaf.
- Pass α, β to descendants

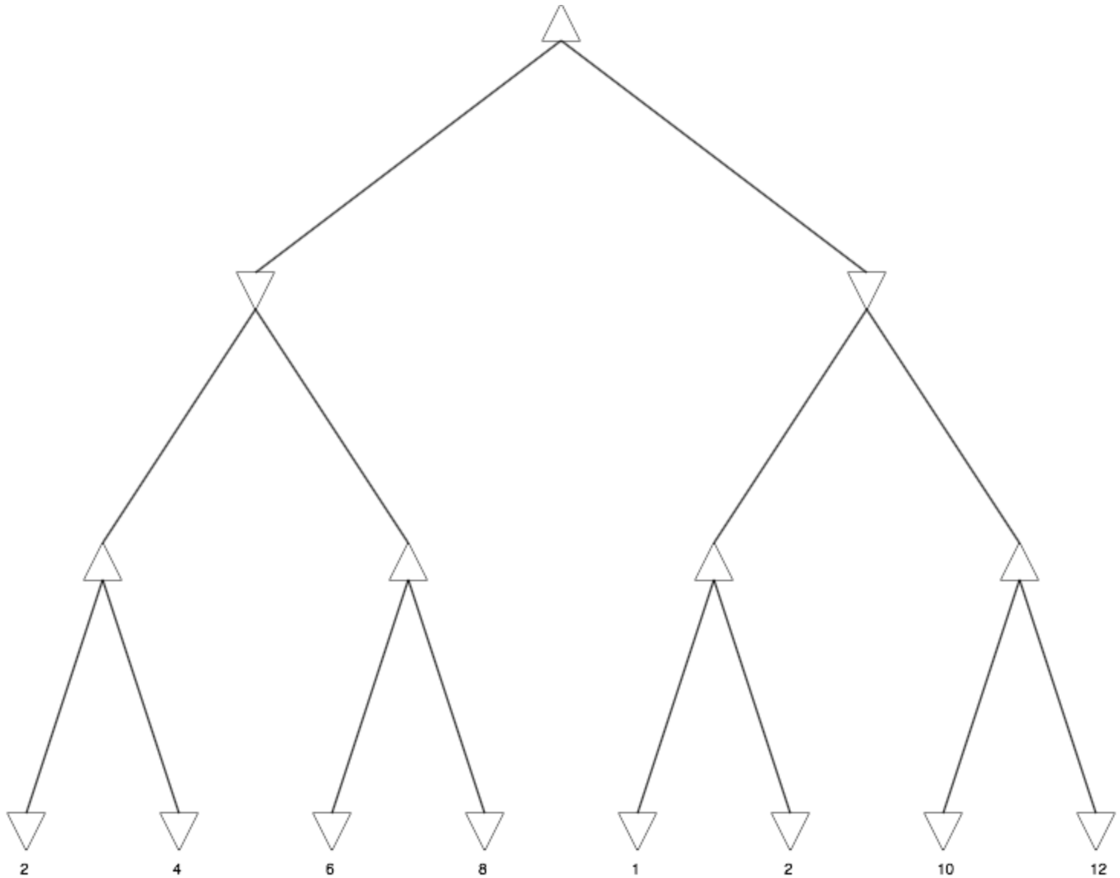


Alpha-beta pruning

- Observe leaf value; MIN's level; MIN updates beta Prune
- play will never reach the other nodes!



Exercise…



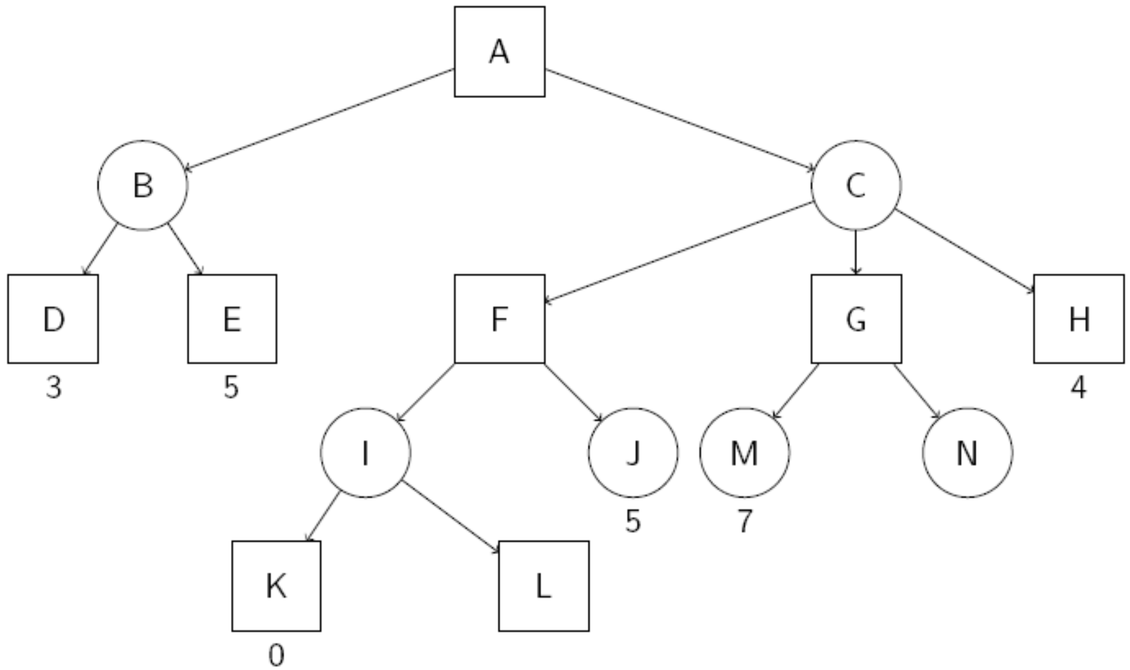
Effectiveness of alpha-beta search

- Worst case: Minimax
- In practice often get $O(b^{(d/2)})$ rather than $O(b^d)$
- This is the same as having a branching factor of sqrt(b),
- In chess go from b~35 to b~6
- Permitting much deeper search in the same amount of time

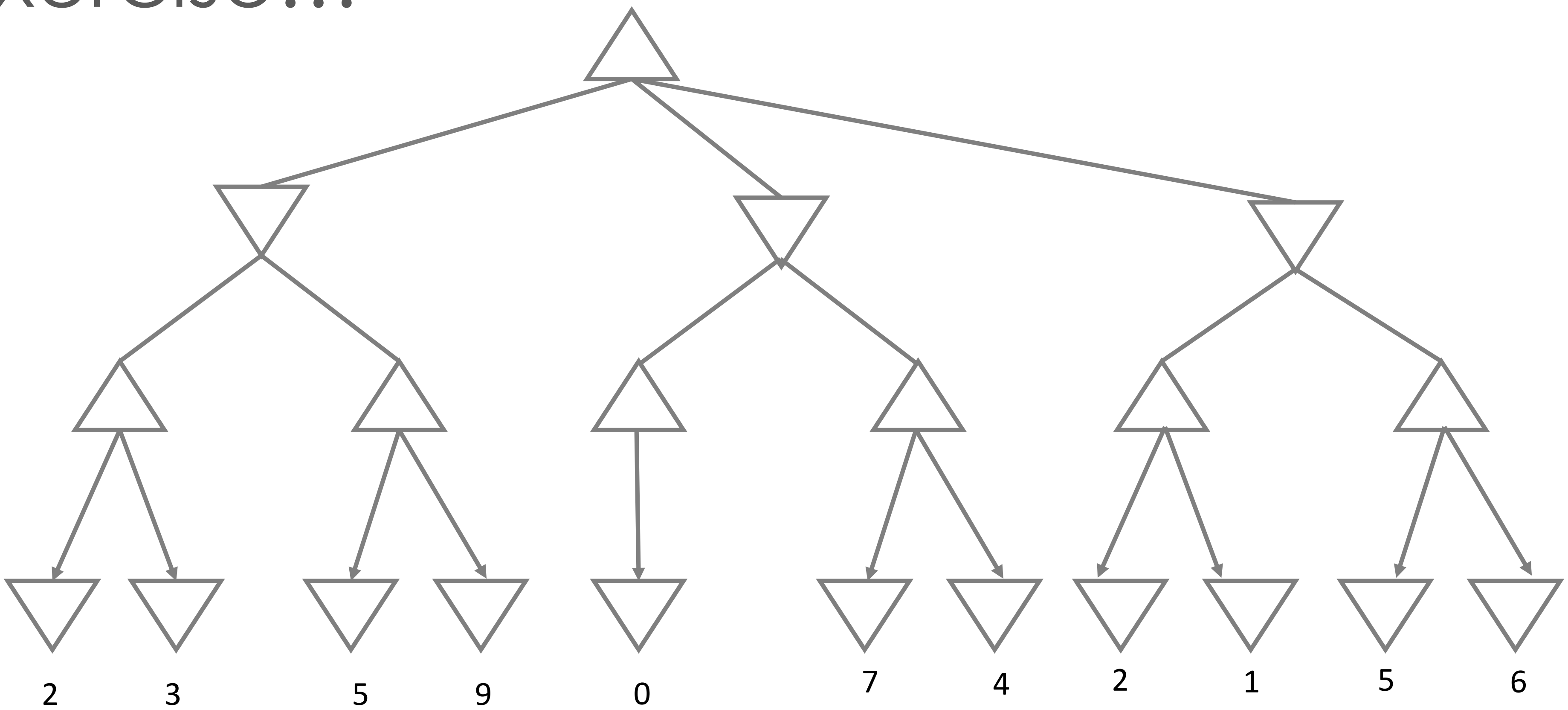
Summary

- Game playing could be modelled as a search problem
- Game trees represent alternate computer/opponent moves
- Heuristic functions estimate the quality of a given board configuration for the MAX player.
- Minimax is a procedure which chooses moves by assuming that the opponent will always choose the move which is best for them
- Alpha-Beta is a procedure which can prune large parts of the search tree and allow search to go deeper
- For many well-known games, computer algorithms based on heuristic search match or out-perform human world experts.

Exercise…



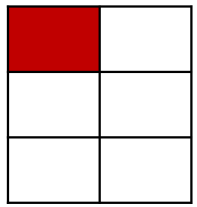
Exercise…



Exercise: Chomp game

Consider the following game:

- Two players have a bar of chocolate with m × n squares.
- The square in the top left corner is known to be poisonous.
- The players play in turn, where the rule is: a player chooses one entire row or column (of 1 or more squares)
- Goal: the player who has to eat the poisonous piece loses.
- Solve for a bar of chocolate with 3 x 2 squares, that is



Hint: possible first actions