Artificial Intelligence

Game Playing Continued

Lecture 9

CS 444 – Spring 2019
Dr. Kevin Molloy
Department of Computer Science
James Madison University

Outline

- Moving forward with game playing
- How to play games of chance
- Example Problem
- Project 2

Review of α- β Pruning

Pruning is an example of metareasoning – computing about what to compute

This type of pruning does not impact optimality of the decision (but does mean some of the nodes may not have the correct values).

Even with perfect ordering, time complexity = $O(b^{m/2})$ (*m* is max depth)

Chess is still impossible (35^{50} possibilities)

Idea: obtain value of a state (move) without reaching the leaf states

Resource Limits

Standard approach:

- Use **Cutoff-Test** instead of **Terminal-Test** (e.g., depth limited search)
- Use **Eval** instead of **Utility** (heuristic evaluation)

Chess Example:

Suppose we have 100 seconds, explore 10^4 nodes/seconds

10^6 nodes per move $\approx 35^{\,8/2}$

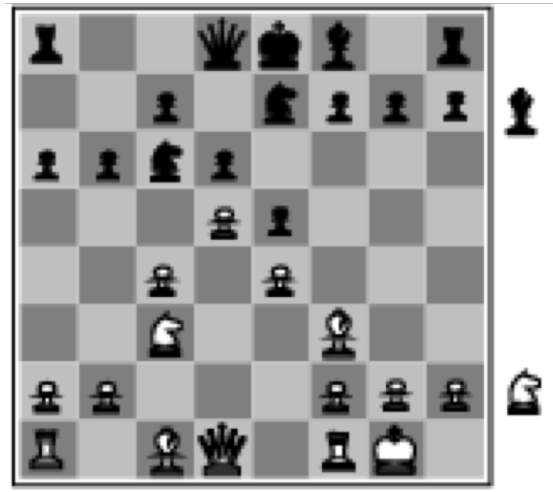
α-β reaches depth 8 $\Longrightarrow$ pretty good chess program

Evaluation Functions

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 $f_1(s)$ (difference in the # of queens)



Black to move

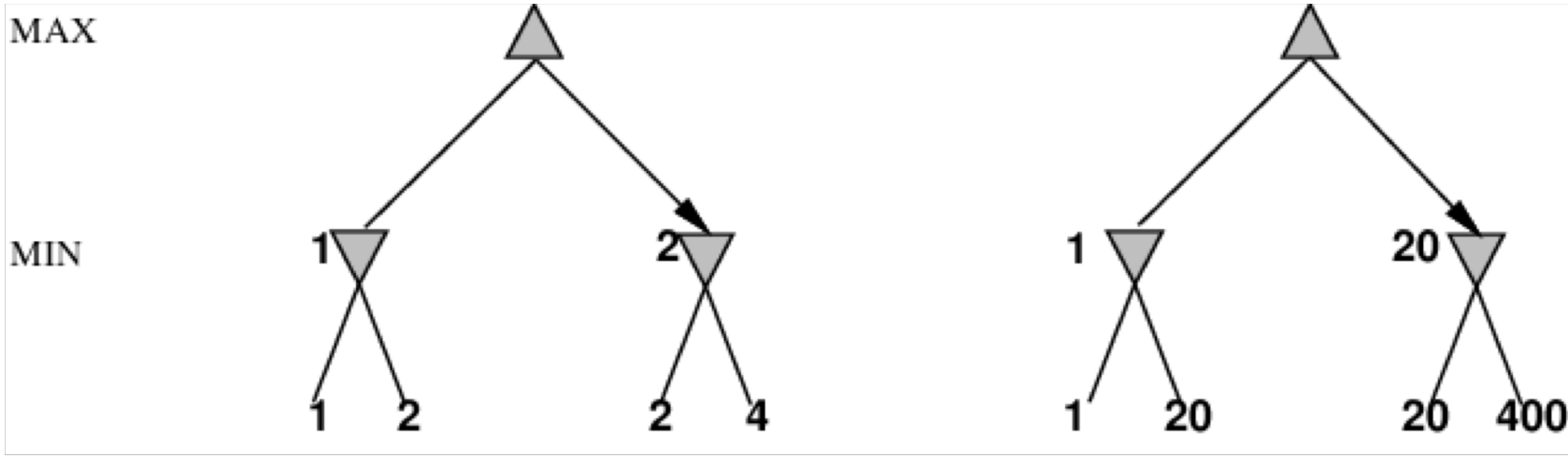
White slightly better

White to move

Black winning

Behavior is preserved under any monotonic transformation of eval.

Only order matters.

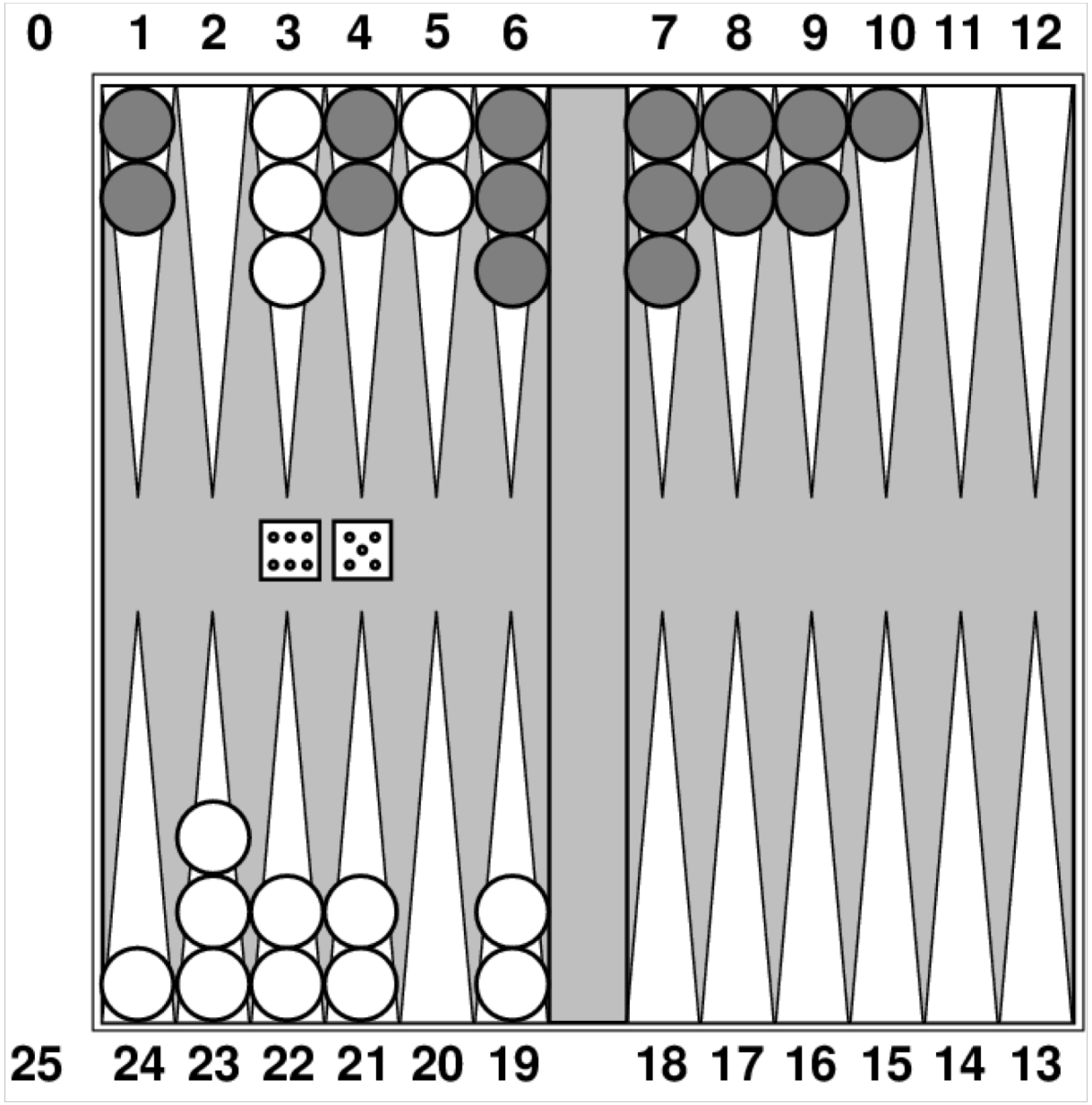


Deterministic games in practice

Chess: Deep blue defeated Gary Kasparov in a six-game match in 1997. The computer searched 200 million positions per second, and was able to extend some search lines up to 40 ply.

Go: branching factor is between 250 and 361 (compared to chess at 35). Pattern knowledge is used instead of search trees. AlphaGo, which employs **deep learning**, is now competing at the world championship level. (https://deepmind.com/research/alphago/).

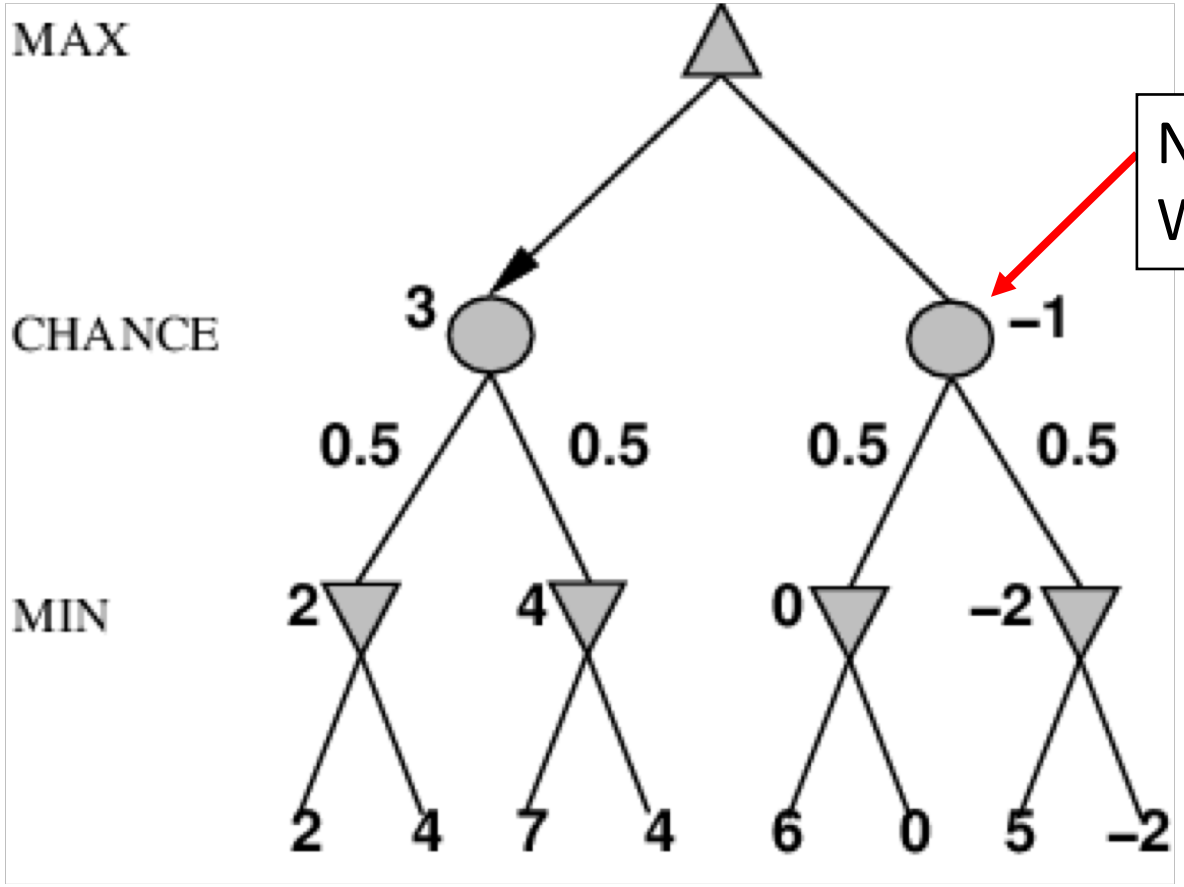
Nondeterministic Games:



Nondeterministic Games

In nondeterministic games, chance is introduced by dice, card-shuffling, etc.

Here is a simplified example with coin-flipping:



New type of node in our tree
We will call this a chance node.

Minimax ⟹ ExpectiMiniMax for Nondeterministic Games

Just like MiniMax, except we must also handle chance nodes:

If *state* is a **Max** node **then**

return highest ExpectiMiniMax-Value of Successor(state)

If *state* is a **Min** node **then**

return lowest ExpectiMiniMax-Value of Successor(state)

If *state* is a **chance** node **then**

return average ExpectiMiniMax-Value of Successor(state)

Nondeterministic Games in Practice

Dice rolls increase *b*: 21 possible distinct rolls with 2 dice.

Backgammon ≈ 20 legal moves (can be 6,000 in 1-1- roll)

Depth 4 = $20 \times (21 \times 20)^3 \approx 1.2 \times 10^9$

Time complexity: $O(b^m n^m)$, where *n* is the number of distinct roll

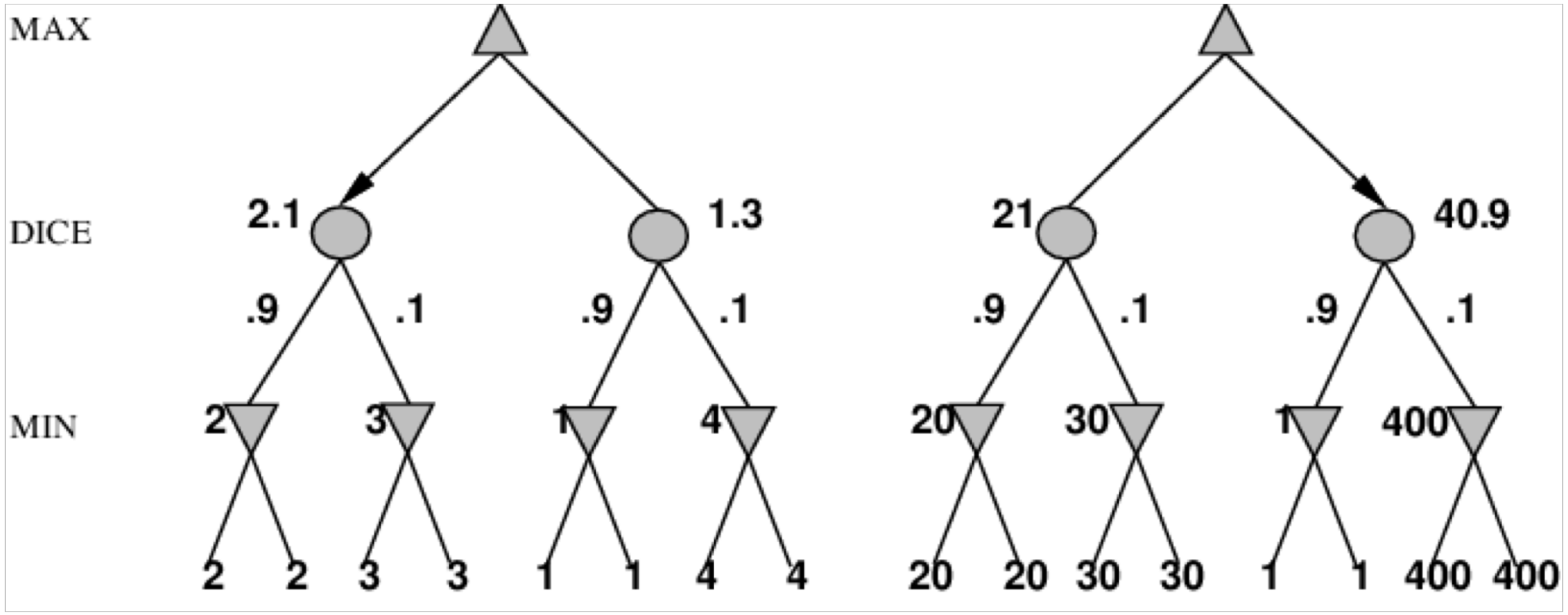
Does it pay to look ahead?

As depth increases, probability of reaching a given node shrinks, thus, its value is diminished. Thus, α-β pruning is not as effective.

TDGammon uses depth-2 search and good EVAL and plays at the

World champion level

Careful EVAL: Exact Values DO matter



- Notice the difference between these two trees.

Monte Carlo Simulation

Monte Carlo simulation can be used to “evaluate” a state

From a start date, have the algorithm play against itself, using random dice rolls

Imperfect Information:

From a start date, have the algorithm play against itself, using random dice rolls.

Ideas for poker?

Generate random hands for your opponent and count how many times you win.

ExpectiMinMax Problem to Solve

Ghost makes the best move 2/3 of the time, but 1/3 of the time makes the worst move.

Fill in the blank nodes