CSE 473

Lecture 8

Adversarial Search: Expectimax and Expectiminimax

Based on slides from CSE AI Faculty + Dan Klein, Stuart Russell, Andrew Moore

Where we have been and where we are headed

- Blind Search
  - DFS, BFS, IDS
- Informed Search
  - Systematic: Uniform cost, greedy best first, A*, IDA*
  - Stochastic: Hill climbing, simulated annealing, GAs
- **Adversarial Search**
  - Mini-max
  - Alpha-beta pruning
  - Evaluation functions for cut off search
  - Expectimax & Expectiminimax

Modeling the Opponent

- So far assumed

  Opponent = rational, optimal (always picks MIN values)

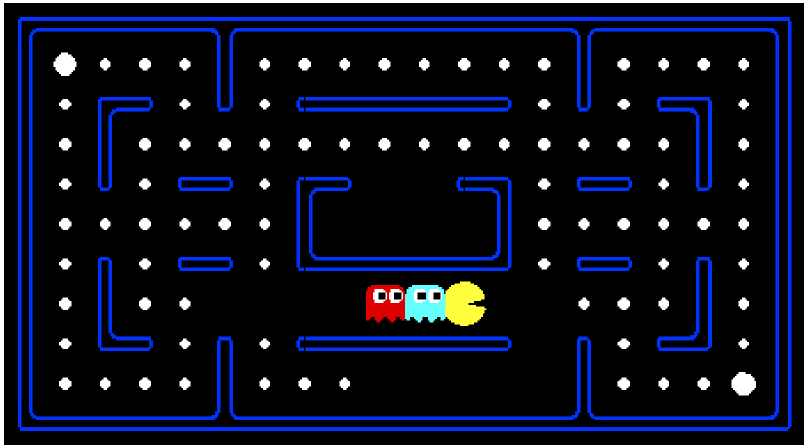
- What if

  Opponent = random? (picks action randomly)

  2 player w/ random opponent = 1 player stochastic

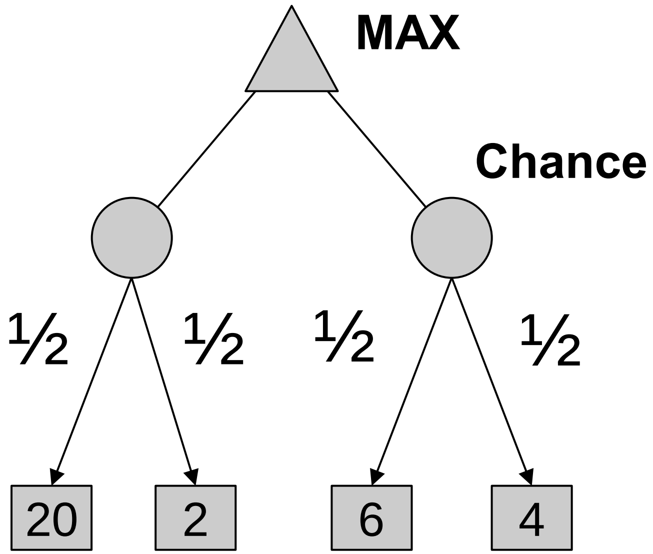
Stochastic Single-Player

- Don't know what the result of an action will be. E.g.,
  - In backgammon, don't know result of dice throw; In solitaire, card shuffle is unknown; in minesweeper, mine locations are unknown
  - In Pac-Man, suppose the ghosts behave randomly



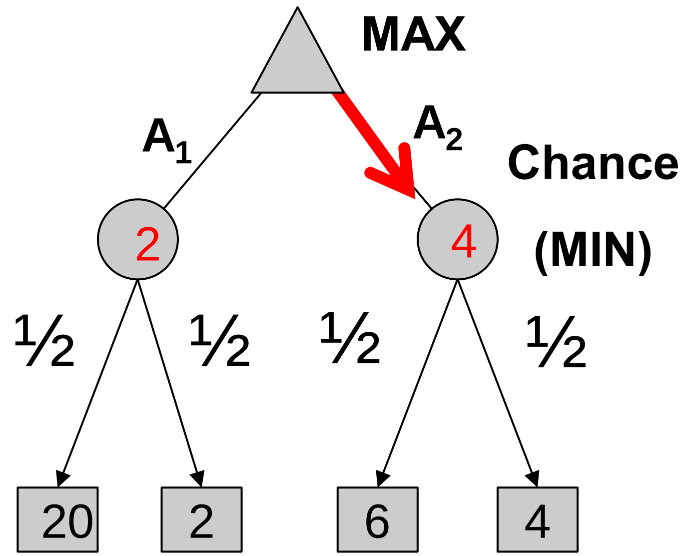
Game Tree for Stochastic Single-Player Game

- Game tree has
  - MAX nodes as before
  - Chance nodes: Environment selects an action with some probability



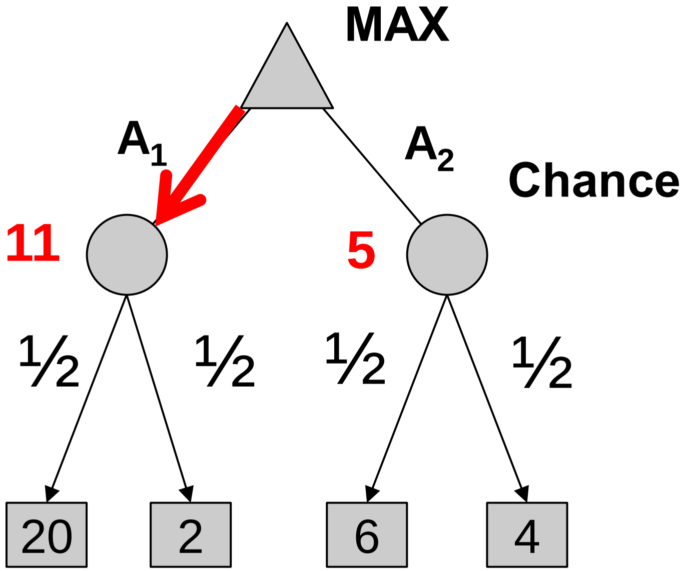
Should we use Minimax Search?

- Minimax strategy: Pick MIN value move at each chance node
- Which move (action) would MAX choose?
- MAX would always choose A_2
  - Average utility = $6/2+4/2 = 5$
- If MAX had chosen A_1
  - Average utility = 11



Expectimax Search

- **Expectimax search:**
  Chance nodes take average (expectation) of value of children
- MAX picks move with *maximum expected value*



Maximizing Expected Utility

- Principle of maximum expected utility:
  An agent should chose the action which maximizes its expected utility, given its knowledge
  - General principle for decision making
  - Often taken as the definition of *rationality*
  - **We will see this idea over and over in this course!**

- Let's decompress this definition…

Review of Probability

- A random variable represents an event whose outcome is unknown
  - Example:
  - Random variable T = Traffic on freeway?
  - Outcomes (or values) for T: {none, light, heavy}

- A probability distribution is an assignment of weights to outcomes
  - Example: P(T=none) = 0.25, P(T=light) = 0.55, P(T=heavy) = 0.20

Review of Probability

- Laws of probability (more later):
  - Probabilities are always in [0, 1]
  - Probabilities (over all possible outcomes) sum to one

- As we get more evidence, probabilities may change:
  - $P(T=heavy) = 0.20$
  - $P(T=heavy \mid Hour=8am) = 0.60$
  - We'll talk about conditional probabilities, methods for reasoning, and updating probabilities later

What are Probabilities?

- **Objectivist / frequentist answer**:

  Probability = average over repeated experiments

  - Examples:
  - Flip a coin 100 times; if 55 heads, 45 tails,
    P(heads)= 0.55 and P(tails) = 0.45
  - P(rain) for Seattle from historical observation
  - PacMan’s estimate of what the ghost will do based on what it has done in the past
  - P(10% of class will get an A) based on past classes
  - P(100% of class will get an A) based on past classes

What are Probabilities?

- Subjectivist / Bayesian answer:

  Degrees of belief about unobserved variables

  - E.g. An agent's belief that it's raining based on what it has observed
  - E.g. PacMan's belief that the ghost will turn left, given the state
  - Your belief that a politician is lying
  - Often agents can *learn* probabilities from past experiences (more later)
  - New evidence *updates beliefs* (more later)

Uncertainty Everywhere

- Not just for games of chance!
  - Robot rotated wheel three times, how far did it advance?
  - Tooth hurts: have cavity?
  - At 45^{th} and the Ave: Safe to cross street?
  - Got up late: Will you make it to class?
  - Didn’t get coffee: Will you stay awake in class?
  - Email subject line says “I have a crush on you”: Is it spam?

Where does uncertainty come from?

- Sources of uncertainty in random variables:
  - Inherently random processes (dice, coin, etc.)
  - Incomplete knowledge of the world
    - Ignorance of underlying processes
    - Unmodeled variables
  - Insufficient or ambiguous evidence, e.g., 3D to 2D image in vision

Expectations

- We can define a function f(X) of a random variable X
- The expected value of a function is its average value under the probability distribution over the function's inputs

$$E\big(f(X)\big) = \sum_x f(X = x) P(X = x)$$

Expectations

- Example: How long to drive to the airport?
  - Driving time (in mins) as a function of traffic T:

    D(T=none) = 20, D(T=light) = 30, D(T=heavy) = 60

  - What is your expected driving time?
    - Recall: P(T) = {none: 0.25, light: 0.5, heavy: 0.25}
    - E[D(T)] = D(none) * P(none) + D(light) * P(light) + D(heavy) * P(heavy)
    - E[D(T)] = (20 * 0.25) + (30 * 0.5) + (60 * 0.25) = 35 mins

Example 2

- Example: Expected value of a fair die roll

X	P	f
1	1/6	1
2	1/6	2
3	1/6	3
4	1/6	4
5	1/6	5
6	1/6	6

$$1 \times \frac{1}{6} + 2 \times \frac{1}{6} + 3 \times \frac{1}{6} + 4 \times \frac{1}{6} + 5 \times \frac{1}{6} + 6 \times \frac{1}{6}$$

$$= 3.5$$

Utilities

- Utilities are *functions* from states of the world to real numbers that describe an agent's preferences

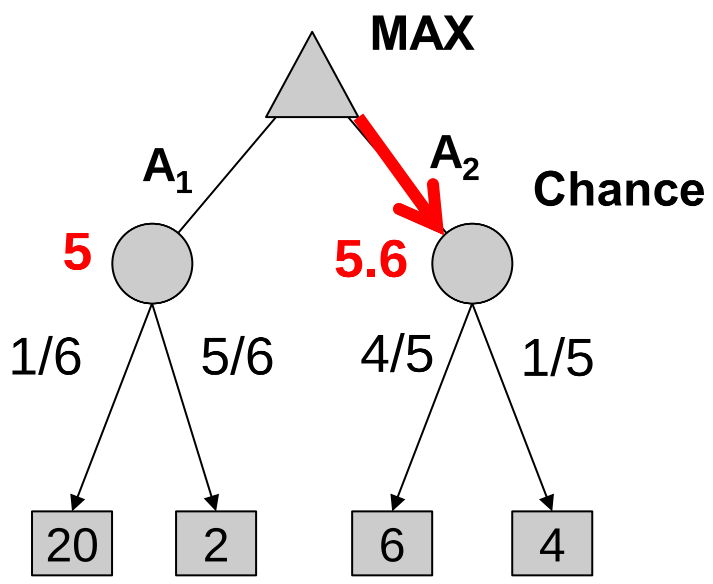
- Where do utilities come from?
  - In a game, may be simple (+1/0/-1 for win/tie/loss)
  - Utilities summarize the agent's goals

- In general, we hard-wire utilities and choose actions to maximize *expected utility*

Back to Expectimax

Expectimax search

- Chance nodes have uncertain outcomes
- Take average (expectation) of value of children to get expected utility or value
- Max nodes as in minimax search but choose action with max expected utility

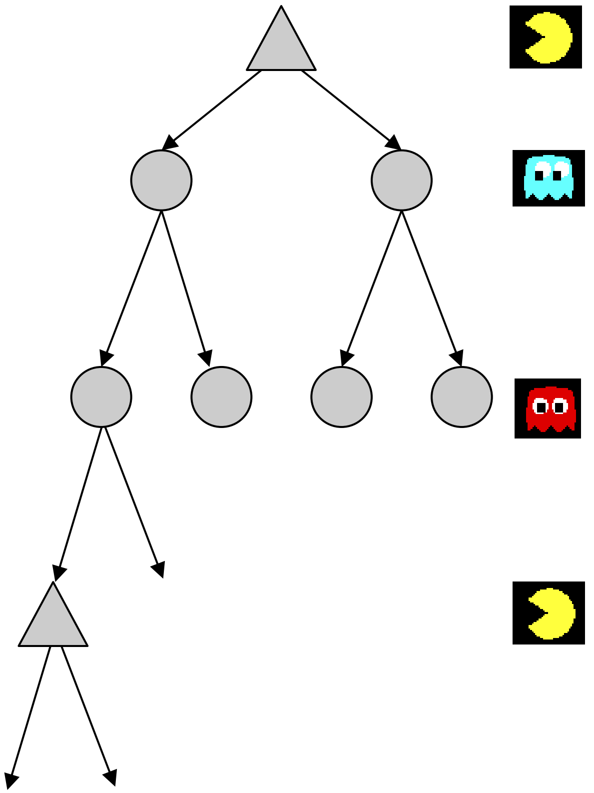


Later, we'll formalize the underlying problem as a

Markov Decision Process

Expectimax Search

- In expectimax search, we have a probabilistic model of how the opponent (or environment) will behave in any state
  - Node for every outcome out of our control: opponent or environment
  - Model can be a simple uniform distribution (e.g., roll a die: 1/6)
  - Model can be sophisticated and require a great deal of computation
    - The model might even say that adversarial actions are more likely! E.g., Ghosts in PacMan

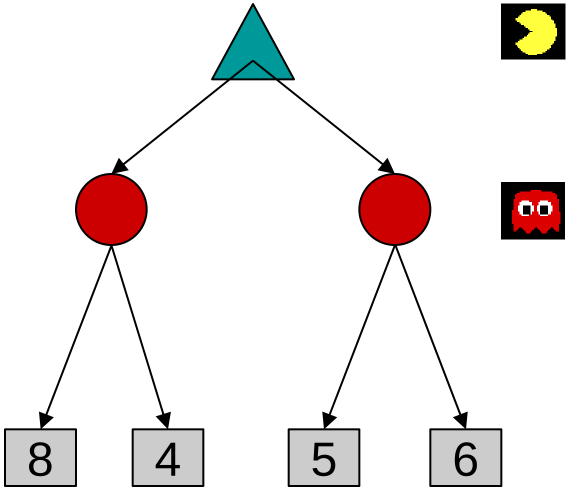


Expectimax Pseudocode

```
def value(s)
    if s is a max node return maxValue(s)
    if s is an exp node return expValue(s)
    if s is a terminal node return evaluation(s)

def maxValue(s)
    values = [value(s') for s' in successors(s)]
    return max(values)

def expValue(s)
    values = [value(s') for s' in successors(s)]
    weights = [probability(s, s') for s' in successors(s)]
    return expectation(values, weights)
```



Minimax versus Expectimax

PacMan with ghosts moving randomly

3 ply look ahead

Minimax: Video

Forgettaboutit...

Minimax versus Expectimax

PacMan with ghosts moving randomly

3 ply look ahead

Expectimax: Video

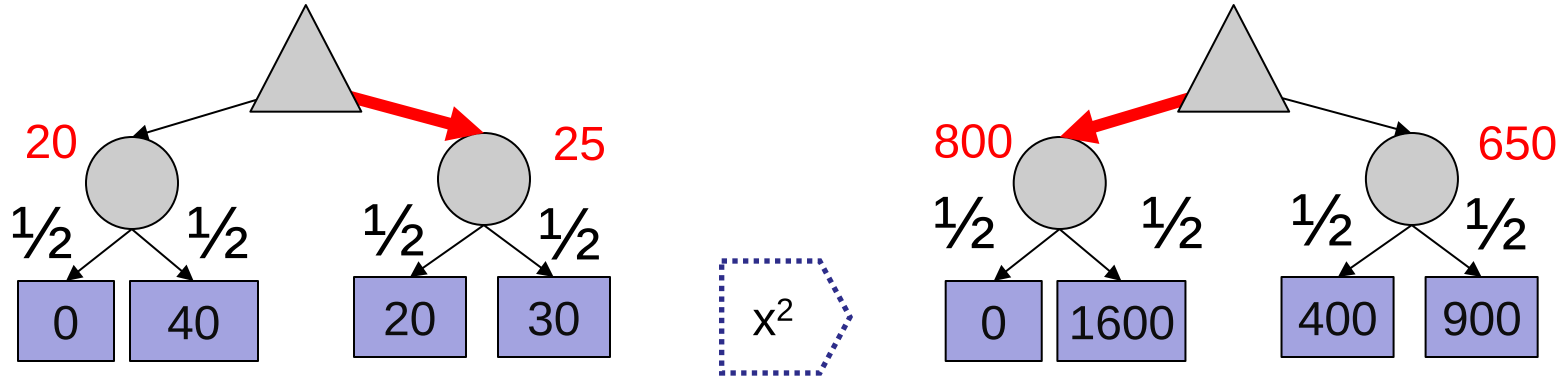
Wins some of the time

Expectimax for Pacman

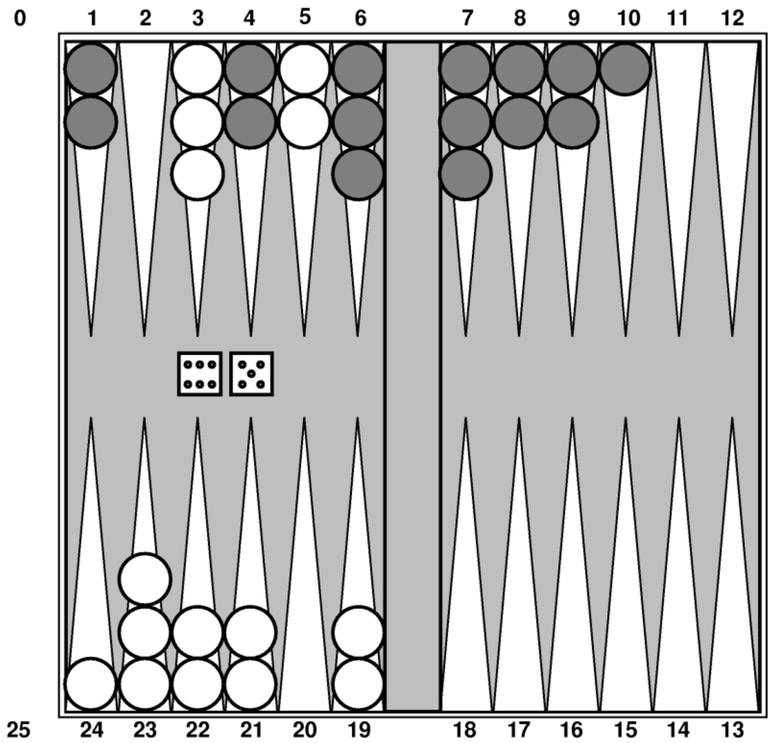
- Ghosts not trying to minimize PacMan's score but moving at random
- They are a part of the environment
- Pacman has a belief (distribution) over how they will act

What about Evaluation Functions for Limited Depth Expectimax?

- Evaluation functions quickly return an estimate for a node's true value
- For minimax, *evaluation function scale* doesn't matter
  - We just want better states to have higher evaluations (using MIN/MAX, so just get the relative value right)
  - We call this insensitivity to monotonic transformations
- For expectimax, *magnitudes* matter!



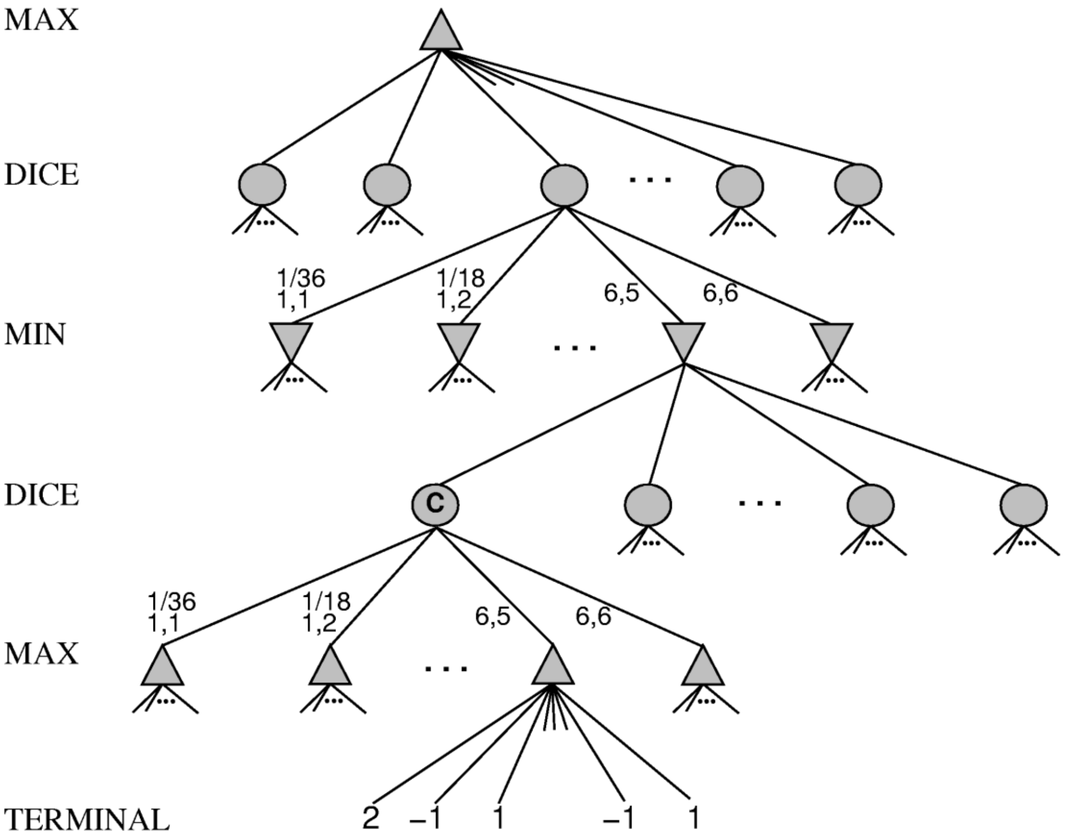
Extending Expectimax to Stochastic Two Player Games



White has just rolled 6-5 and has 4 legal moves.

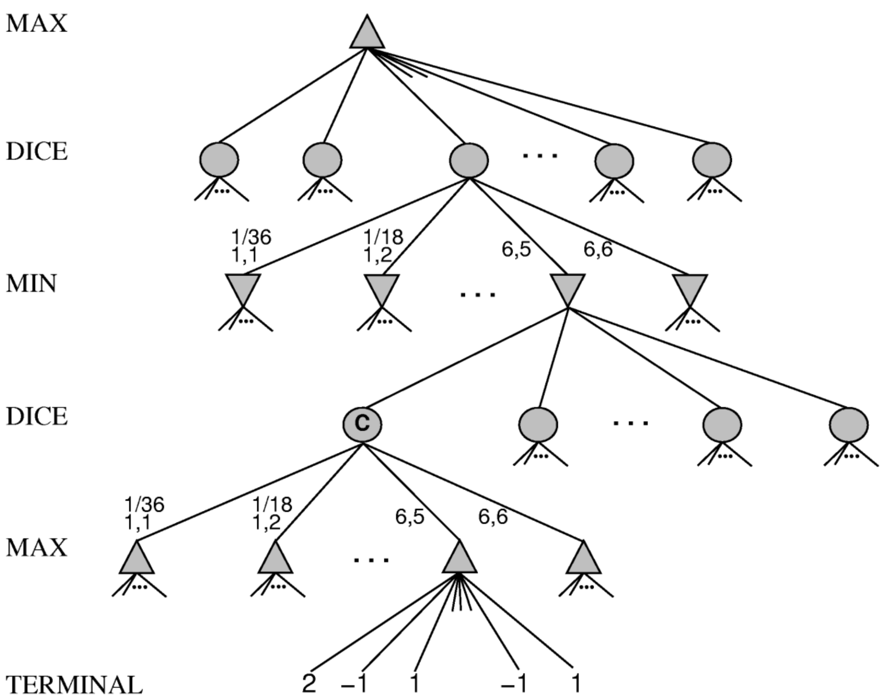
Expectiminimax Search

- In addition to MIN- and MAX nodes, we have chance nodes (e.g., for rolling dice)
- Chance nodes take expectations, otherwise like minimax



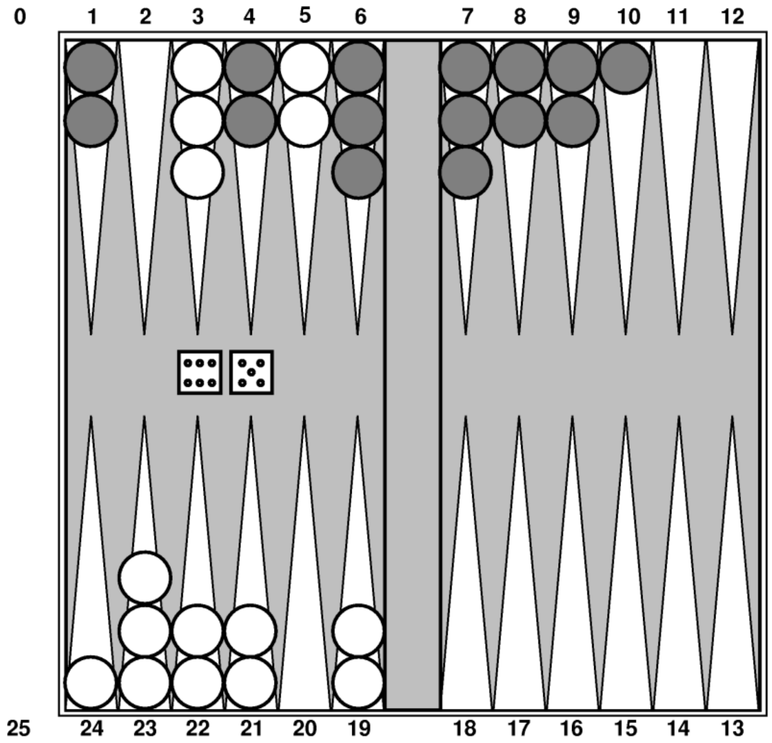
Expectiminimax Search

if *state* is a MAX node **then**
  return the highest EXPECTIMINIMAX-VALUE of SUCCESSORS(*state*)
if *state* is a MIN node **then**
  return the lowest EXPECTIMINIMAX-VALUE of SUCCESSORS(*state*)
if *state* is a chance node **then**
  return average of EXPECTIMINIMAX-VALUE of SUCCESSORS(*state*)



Search costs increase: Instead of $O(b^d)$, we get $O((bn)^d)$, where n is the number of chance outcomes

Example: TDGammon program



TDGammon uses depth-2 search + very good eval function + reinforcement learning (playing against itself!)
→ world-champion level play

Summary of Game Tree Search

- Basic idea: Minimax
  - Too slow for most games
- Alpha-Beta pruning can increase max depth by factor up to 2
- Limited depth search necessary for most games
- Static evaluation functions necessary for limited depth search; opening game and end game databases can help
- Computers can beat humans in some games (checkers, chess, othello) but not yet in others (Go)
- Expectimax and Expectiminimax allow search in stochastic games

To Do

- Finish Project #1: Due Sunday before midnight
- Finish Chapter 5; Read Chapter 7