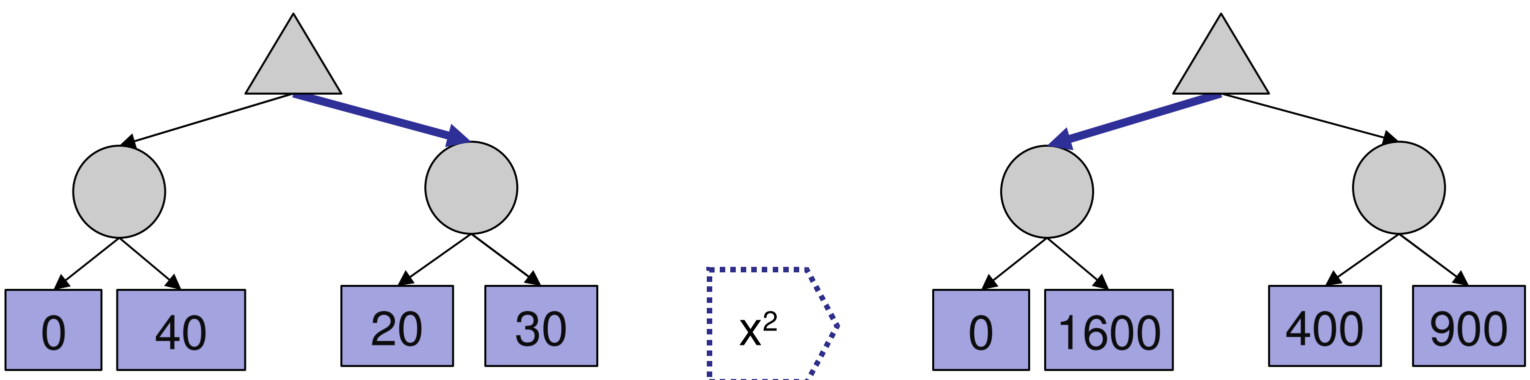
Expectimax Evaluation

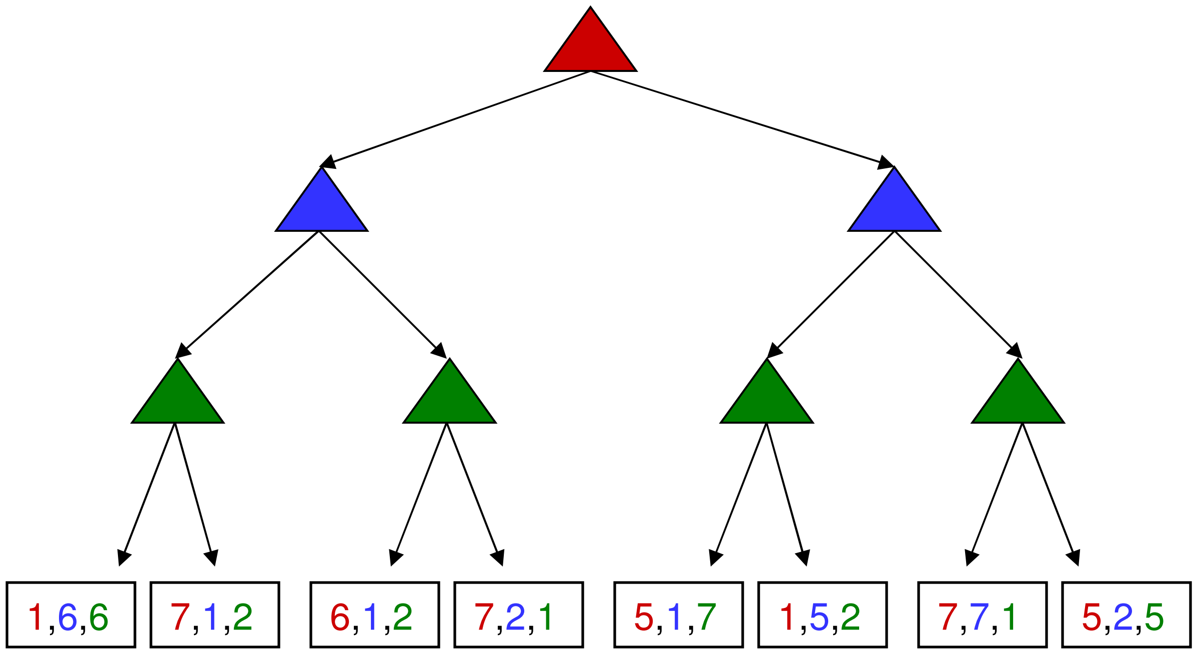
- Evaluation functions quickly return an estimate for a node's true value (which value, expectimax or minimax?)
- For minimax, evaluation function scale doesn't matter
  - We just want better states to have higher evaluations (get the ordering right)
  - We call this insensitivity to monotonic transformations
- For expectimax, we need *magnitudes* to be meaningful



This slide deck courtesy of Dan Klein at UC Berkeley

Multi-Agent Utilities

- Similar to minimax:
  - Terminals have utility tuples
  - Node values are also utility tuples
  - Each player maximizes its own utility
  - Can give rise to cooperation and competition dynamically…



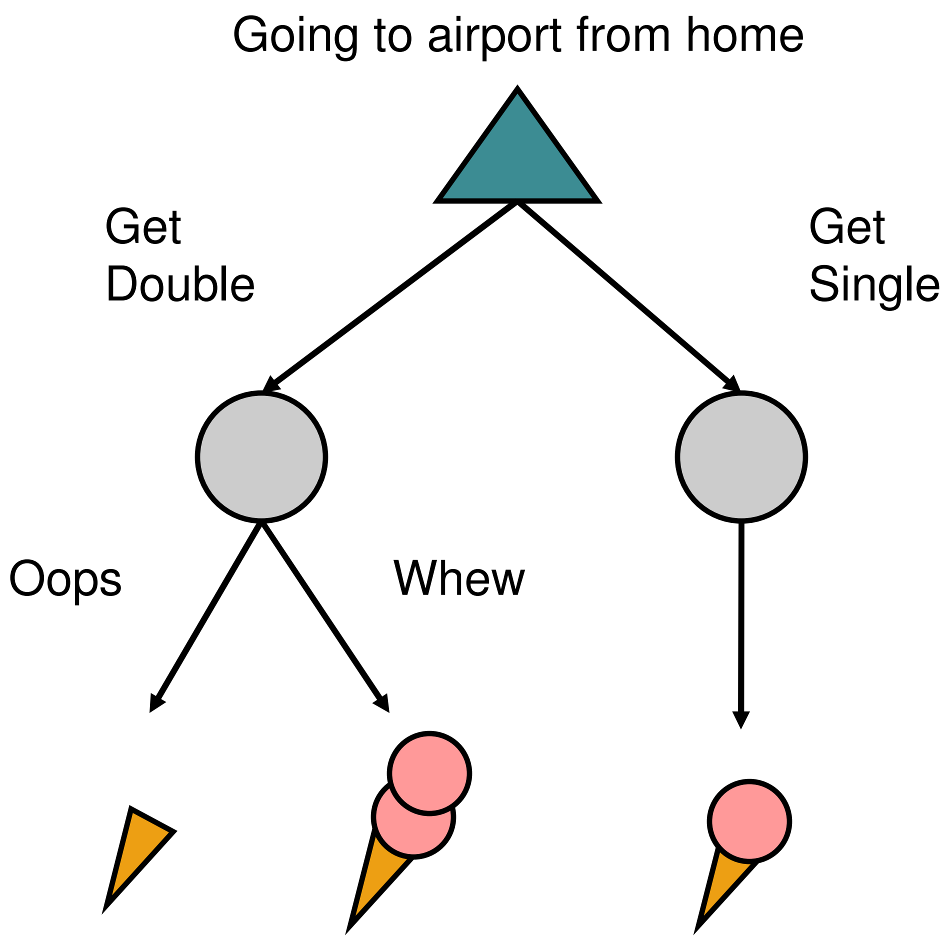
Maximum Expected Utility

- Why should we average utilities? Why not minimax?

- Principle of maximum expected utility:
  - A rational agent should chose the action which maximizes its expected utility, given its knowledge

- Questions:
  - Where do utilities come from?
  - How do we know such utilities even exist?
  - Why are we taking expectations of utilities (not, e.g. minimax)?
  - What if our behavior can't be described by utilities?

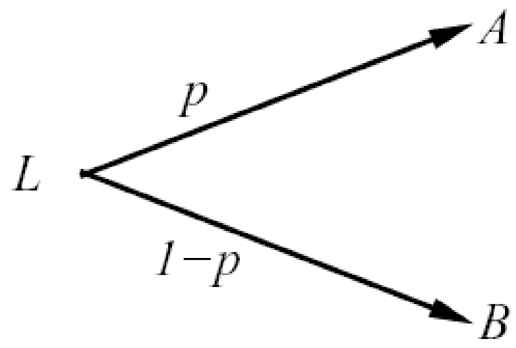
Utilities: Uncertain Outcomes



Preferences

- An agent chooses among:
  - Prizes: A, B, etc.
  - Lotteries: situations with uncertain prizes

$$L = [p, A;\ (1-p), B]$$



- Notation:

$A \succ B$ $\quad$ A preferred over B

$A \sim B$ $\quad$ indifference between A and B

$A \succeq B$ $\quad$ B not preferred over A

Rational Preferences

- Preferences of a rational agent must obey constraints.
  - The axioms of rationality:

Orderability

$$(A \succ B) \vee (B \succ A) \vee (A \sim B)$$

Transitivity

$$(A \succ B) \wedge (B \succ C) \Rightarrow (A \succ C)$$

Continuity

$$A \succ B \succ C \Rightarrow \exists p\ [p, A;\ 1-p, C] \sim B$$

Substitutability

$$A \sim B \Rightarrow [p, A;\ 1-p, C] \sim [p, B; 1-p, C]$$

Monotonicity

$$A \succ B \Rightarrow$$
$$(p \geq q \Leftrightarrow [p, A;\ 1-p, B] \succeq [q, A;\ 1-q, B])$$

- Theorem: Rational preferences imply behavior describable as maximization of expected utility

MEU Principle

- Theorem:
  - [Ramsey, 1931; von Neumann & Morgenstern, 1944]
  - Given any preferences satisfying these constraints, there exists a real-valued function U such that:

$$U(A) \geq U(B) \;\Leftrightarrow\; A \succeq B$$

$$U([p_1, S_1;\ \ldots\ ;\ p_n, S_n]) = \sum_i p_i U(S_i)$$

- Maximum expected utility (MEU) principle:
  - Choose the action that maximizes expected utility
  - Note: an agent can be entirely rational (consistent with MEU) without ever representing or manipulating utilities and probabilities
  - E.g., a lookup table for perfect tictactoe, reflex vacuum cleaner

Utility Scales

- Normalized utilities: $u_+ = 1.0$, $u_- = 0.0$

- Micromorts: one-millionth chance of death, useful for paying to reduce product risks, etc.

- QALYs: quality-adjusted life years, useful for medical decisions involving substantial risk

- Note: behavior is invariant under positive linear transformation

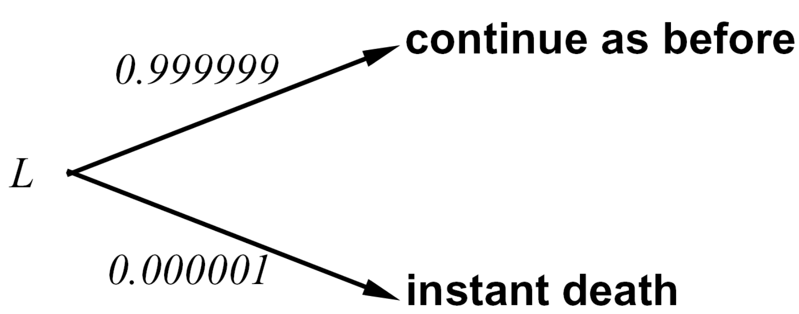
$$U'(x) = k_1 U(x) + k_2 \quad \text{where } k_1 > 0$$

- With deterministic prizes only (no lottery choices), only ordinal utility can be determined, i.e., total order on prizes

Human Utilities

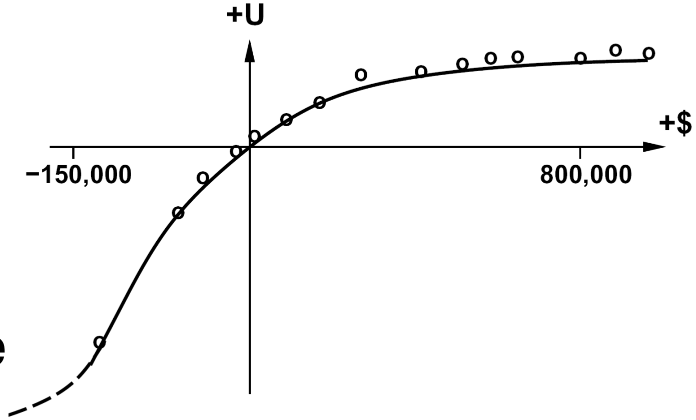
- Utilities map states to real numbers. Which numbers?
- Standard approach to assessment of human utilities:
  - Compare a state A to a standard lottery L_p between
    - "best possible prize" u_+ with probability p
    - "worst possible catastrophe" u_- with probability 1-p
  - Adjust lottery probability p until A ~ L_p
  - Resulting p is a utility in [0,1]

pay $30 ~



Money

- Money <u>does not</u> behave as a utility function, but we can talk about the utility of having money (or being in debt)
- Given a lottery L = [p, $X; (1-p), $Y]
  - The expected monetary value EMV(L) is p*X + (1-p)*Y
  - U(L) = p*U($X) + (1-p)*U($Y)
  - Typically, U(L) < U(EMV(L)): why?
  - In this sense, people are risk-averse
  - When deep in debt, we are risk-prone



- Utility curve: for what probability p am I indifferent between:
  - Some sure outcome x
  - A lottery [p,$M; (1-p),$0], M large

Example: Insurance

- Consider the lottery [0.5,$1000; 0.5,$0]
  - What is its expected monetary value? ($500)
  - What is its certainty equivalent?
    - Monetary value acceptable in lieu of lottery
    - $400 for most people
  - Difference of $100 is the insurance premium
    - There's an insurance industry because people will pay to reduce their risk
    - If everyone were risk-neutral, no insurance needed!

Example: Human Rationality?

- Famous example of Allais (1953)
  - A: [0.8,\$4k; 0.2,\$0]
  - B: [1.0,\$3k; 0.0,\$0]

  - C: [0.2,\$4k; 0.8,\$0]
  - D: [0.25,\$3k; 0.75,\$0]

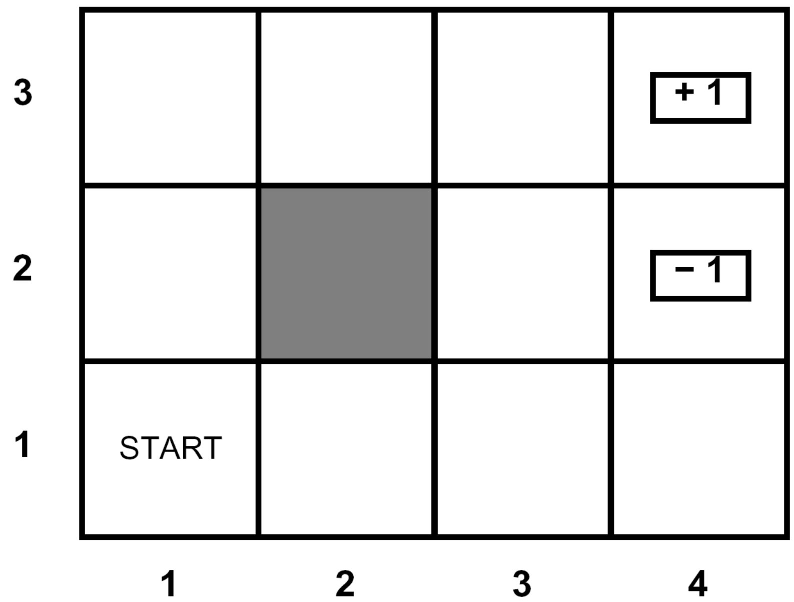
- Most people prefer B > A, C > D
- But if U(\$0) = 0, then
  - $B > A \Rightarrow U(\$3k) > 0.8\, U(\$4k)$
  - $C > D \Rightarrow 0.8\, U(\$4k) > U(\$3k)$

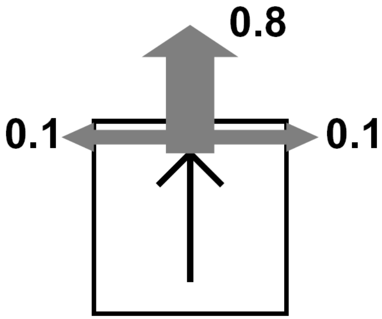
Non-Deterministic Search

How do you plan when your actions might fail?

Example: Grid World

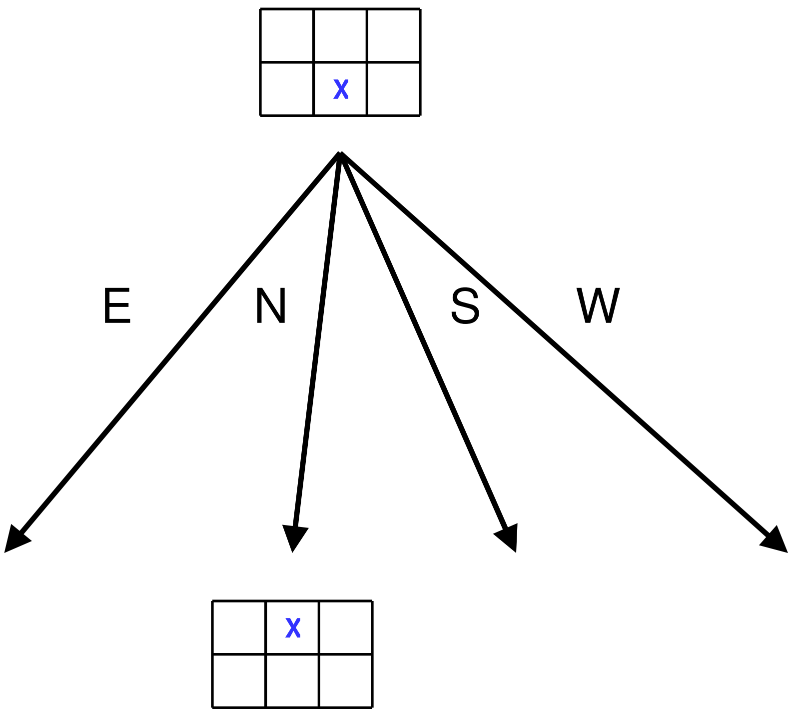
- The agent lives in a grid
- Walls block the agent's path
- The agent's actions do not always go as planned:
  - 80% of the time, the action North takes the agent North
    (if there is no wall there)
  - 10% of the time, North takes the agent West; 10% East
  - If there is a wall in the direction the agent would have been taken, the agent stays put
- Small "living" reward each step
- Big rewards come at the end
- Goal: maximize sum of rewards*



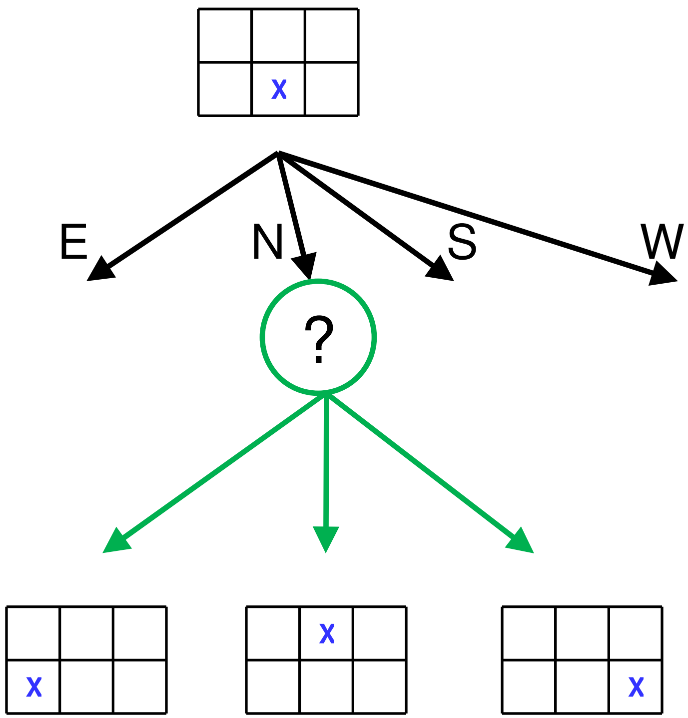


Action Results

Deterministic Grid World



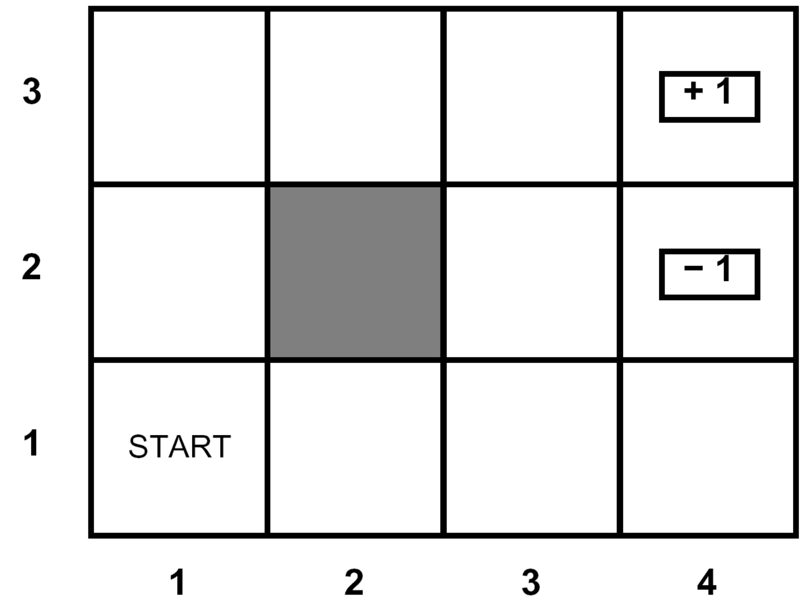
Stochastic Grid World

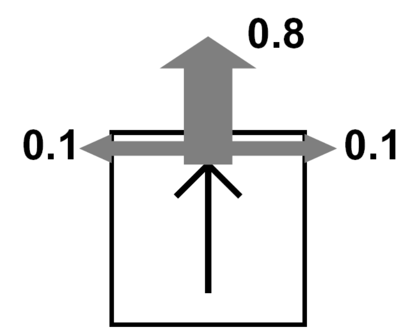


Markov Decision Processes

- An MDP is defined by:
  - A set of states $s \in S$
  - A set of actions $a \in A$
  - A transition function T(s,a,s')
    - Prob that a from s leads to s'
    - i.e., P(s' | s,a)
    - Also called the model
  - A reward function R(s, a, s')
    - Sometimes just R(s) or R(s')
  - A start state (or distribution)
  - Maybe a terminal state

- MDPs are a family of non-deterministic search problems
  - One way to solve them is with expectimax search – but we'll have a new tool soon





What is Markov about MDPs?

- Andrey Markov (1856-1922)
- “Markov” generally means that given the present state, the future and the past are independent
- For Markov decision processes, “Markov” means:

$$P(S_{t+1} = s' | S_t = s_t, A_t = a_t, S_{t-1} = s_{t-1}, A_{t-1}, \ldots S_0 = s_0)$$

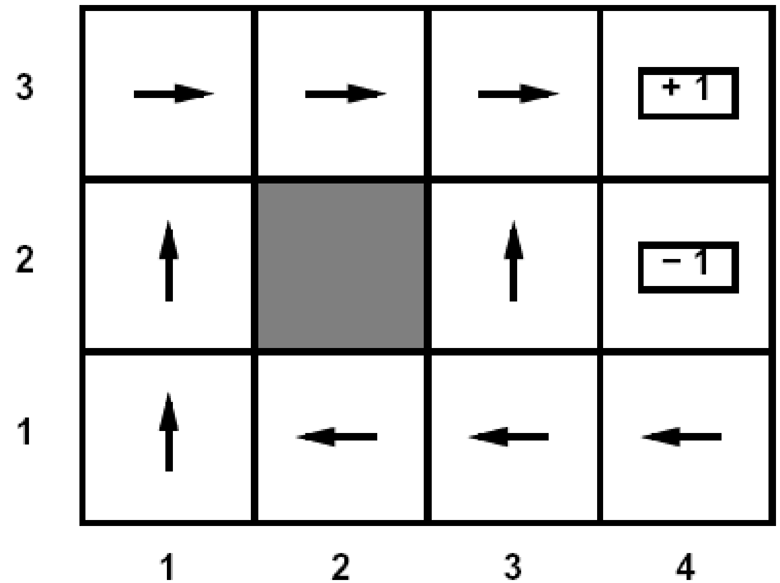
$$=$$

$$P(S_{t+1} = s' | S_t = s_t, A_t = a_t)$$

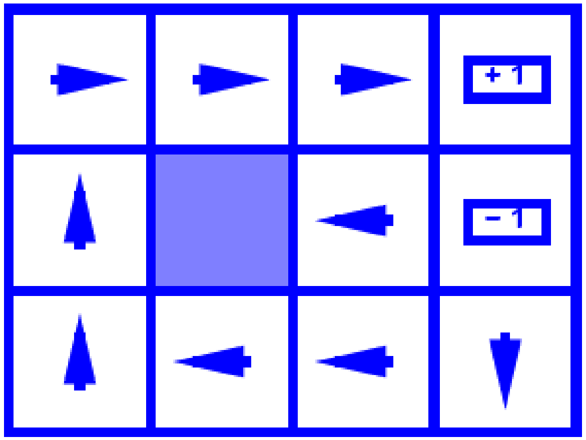
Solving MDPs

- In deterministic single-agent search problems, want an optimal plan, or sequence of actions, from start to a goal
- In an MDP, we want an optimal policy π *: S → A
  - A policy π gives an action for each state
  - An optimal policy maximizes expected utility if followed
  - Defines a reflex agent (if precomputed)

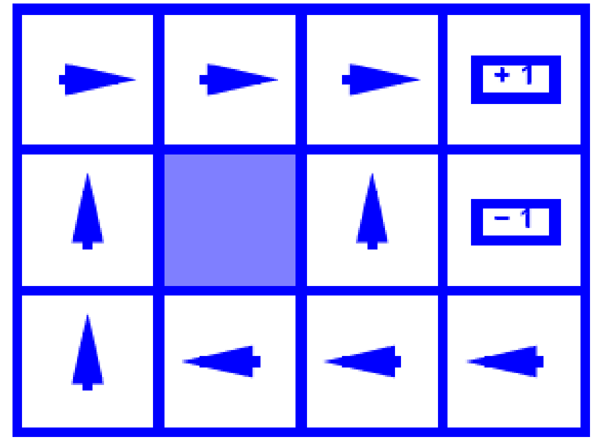
Optimal policy when R(s, a, s') = -0.03 for all non-terminals s



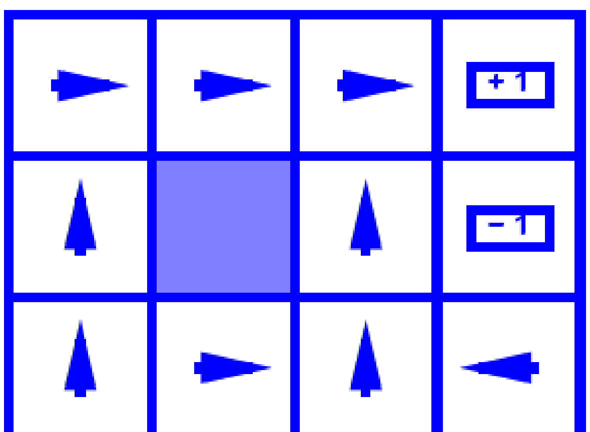
Example Optimal Policies



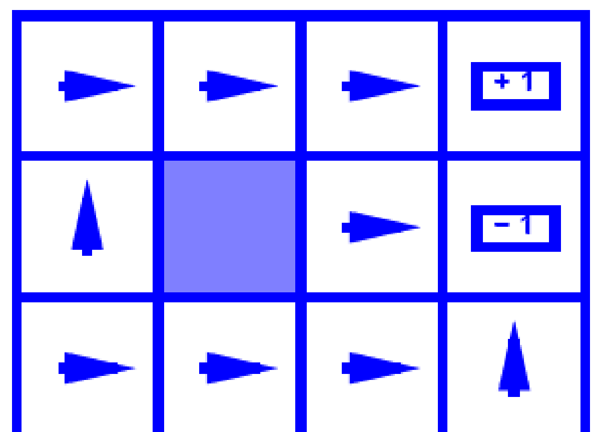
R(s) = -0.01



R(s) = -0.03



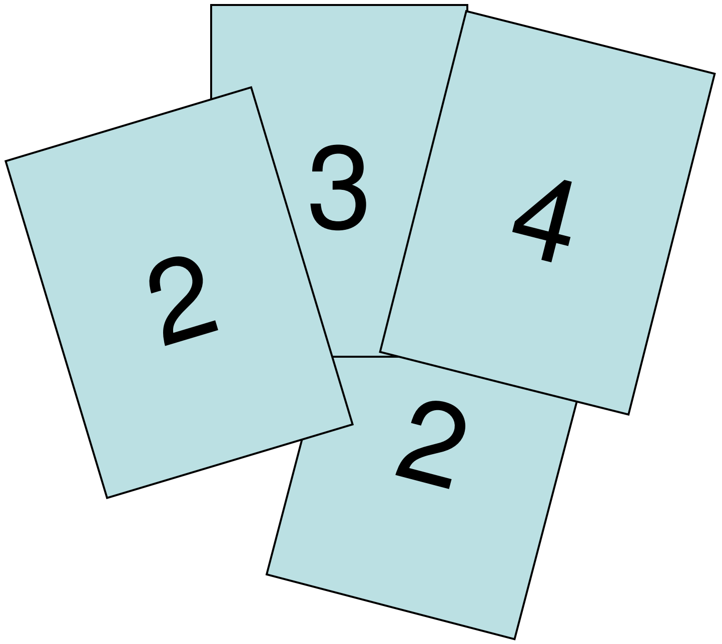
R(s) = -0.4



R(s) = -2.0

Example: High-Low

- Rules
  - Three card types: 2, 3, 4
  - Infinite deck, twice as many 2’s
  - Start with 3 showing
  - After each card, you guess the next card will be “high” or “low”
  - New card is flipped
  - If you’re right, you win the points shown on the new card
  - Ties are no-ops
  - If you’re wrong, game ends

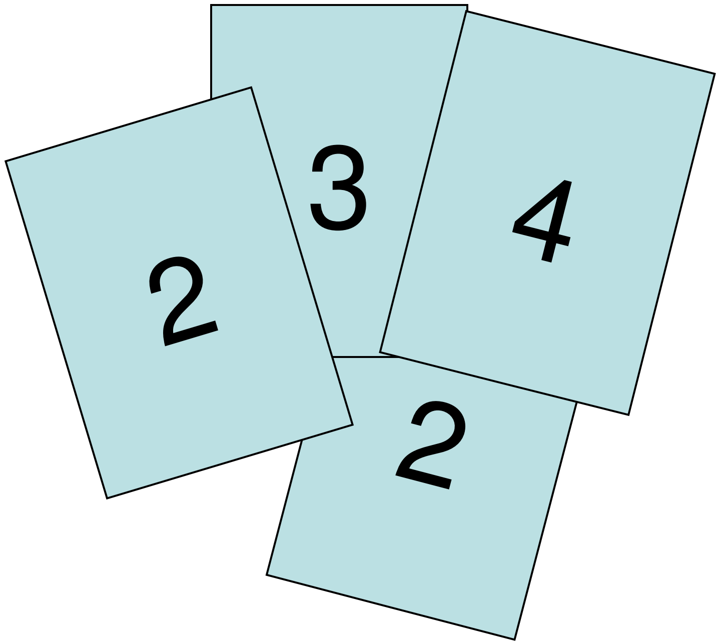


- How is this different from the expectimax games?
  - #1: get rewards as you go
  - #2: you might play forever!

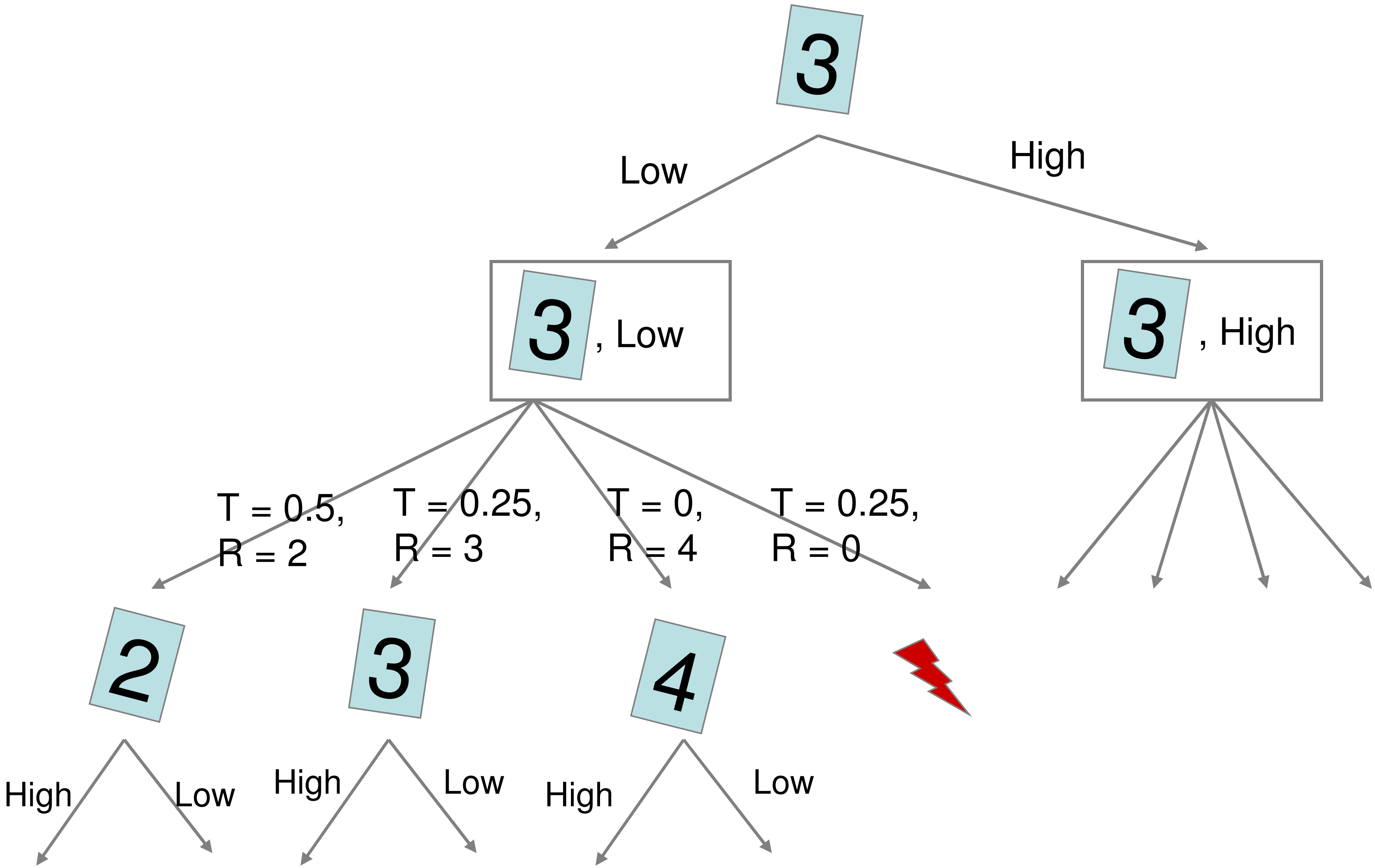
You can patch expectimax to deal with #1, but not #2…

High-Low as an MDP

- States: 2, 3, 4, done
- Actions: High, Low
- Model: T(s, a, s’):
  - P(s’=4 | 4, Low) = 1/4
  - P(s’=3 | 4, Low) = 1/4
  - P(s’=2 | 4, Low) = 1/2
  - P(s’=done | 4, Low) = 0
  - P(s’=4 | 4, High) = 1/4
  - P(s’=3 | 4, High) = 0
  - P(s’=2 | 4, High) = 0
  - P(s’=done | 4, High) = 3/4
  - …
- Rewards: R(s, a, s’):
  - Number shown on s’ if s $\neq$ s’
  - 0 otherwise
- Start: 3

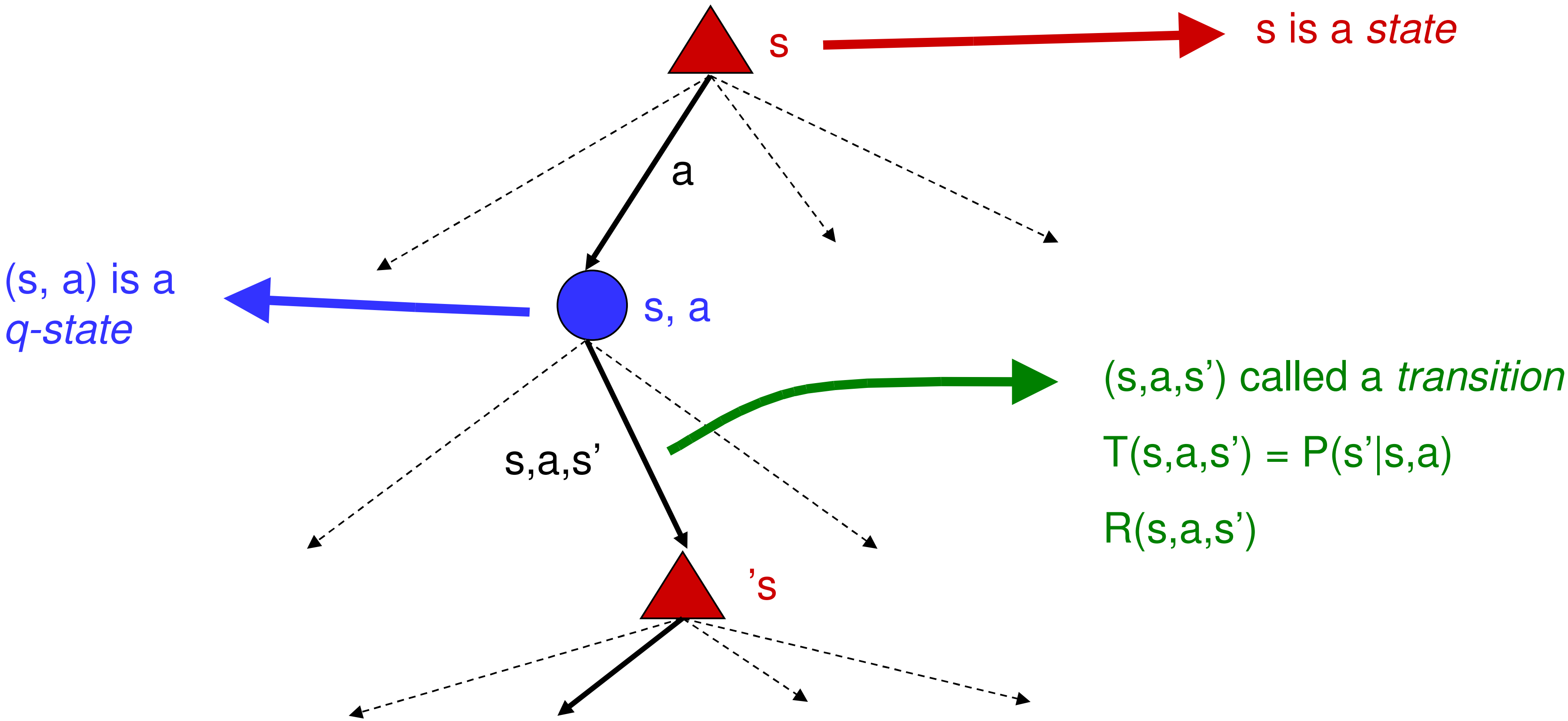


High-Low: Outcome Tree



MDP Search Trees

- Each MDP state gives an expectimax-like search tree



Utilities of Sequences

- What utility does a sequence of rewards have?

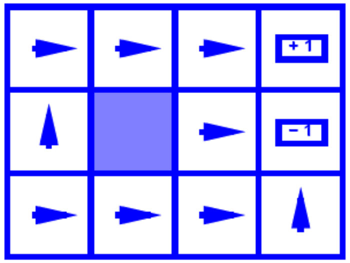
- Formally, we generally assume stationary preferences:

$$[r, r_0, r_1, r_2, \ldots] \succ [r, r_0', r_1', r_2', \ldots]$$
$$\Leftrightarrow$$
$$[r_0, r_1, r_2, \ldots] \succ [r_0', r_1', r_2', \ldots]$$

- Theorem: only two ways to define stationary utilities
  - Additive utility:
  $$U([r_0, r_1, r_2, \ldots]) = r_0 + r_1 + r_2 + \cdots$$
  - Discounted utility:
  $$U([r_0, r_1, r_2, \ldots]) = r_0 + \gamma r_1 + \gamma^2 r_2 \cdots$$

Infinite Utilities?!

- Problem: infinite state sequences have infinite rewards



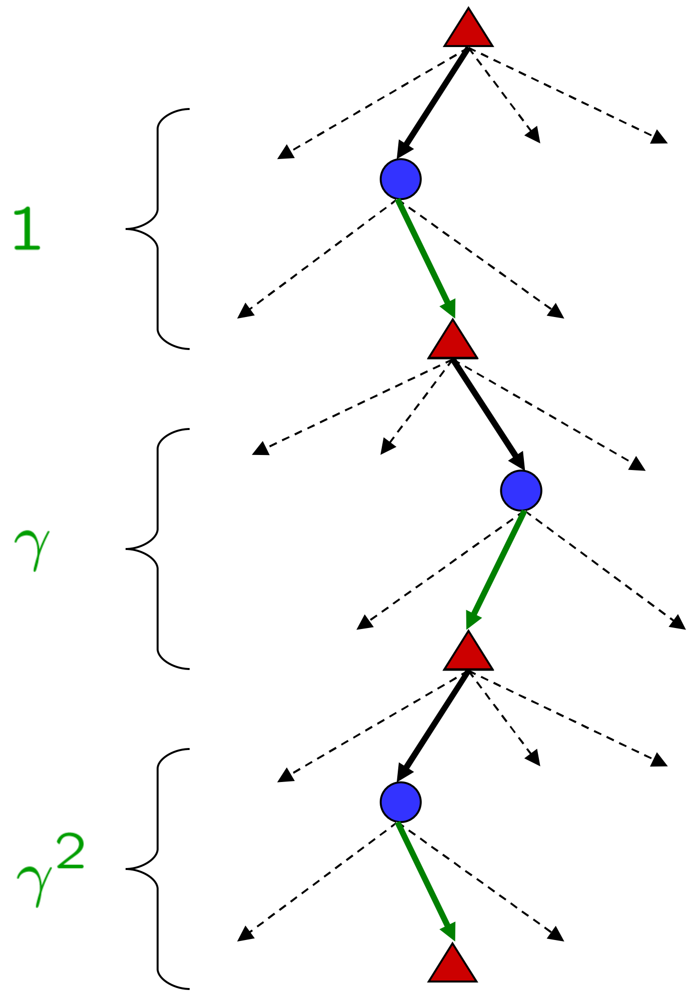
- Solutions:
  - Finite horizon:
    - Terminate episodes after a fixed T steps (e.g. life)
    - Gives nonstationary policies (π depends on time left)
  - Absorbing state: guarantee that for every policy, a terminal state will eventually be reached (like "done" for High-Low)
  - Discounting: for $0 < \gamma < 1$

$$U([r_0, \ldots r_\infty]) = \sum_{t=0}^{\infty} \gamma^t r_t \leq R\text{max}/(1-\gamma)$$

    - Smaller γ means smaller "horizon" – shorter term focus

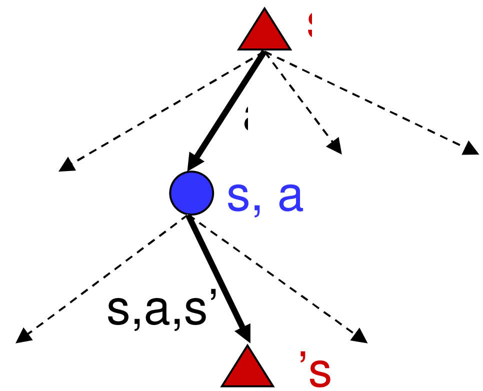
Discounting

- Typically discount rewards by $\gamma < 1$ each time step
  - Sooner rewards have higher utility than later rewards
  - Also helps the algorithms converge
- Example: discount of 0.5
  - U([1,2,3]) = 1*1 + 0.5*2 + 0.25*3
  - U([1,2,3]) < U([3,2,1])



Recap: Defining MDPs

- Markov decision processes:
  - States S
  - Start state s_0
  - Actions A
  - Transitions P(s'|s,a) (or T(s,a,s'))
  - Rewards R(s,a,s') (and discount γ)



- MDP quantities so far:
  - Policy = Choice of action for each state
  - Utility (or return) = "expectimax value" of a state

Optimal Utilities

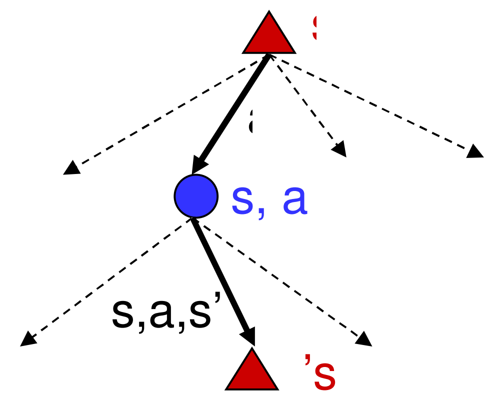
- Fundamental operation: compute the values (optimal expectimax utilities) of states s

- Why? Optimal values define optimal policies!

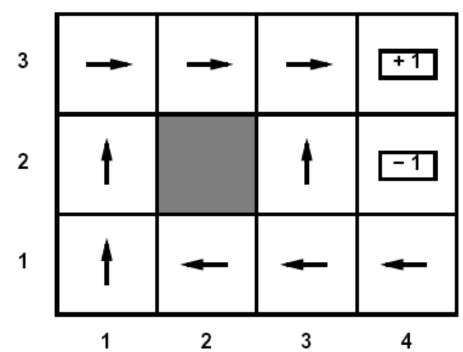
- Define the value of a state s:
  $V^*(s)$ = expected utility starting in s and acting optimally

- Define the value of a q-state (s,a):
  $Q^*(s,a)$ = expected utility starting out having taken action a from state s and (thereafter) acting optimally

- Define the optimal policy:
  $\pi^*(s)$ = optimal action from state s

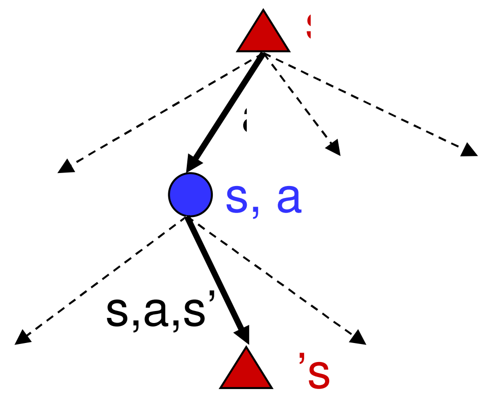


3	0.812	0.868	0.912	+ 1
2	0.762		0.660	− 1
1	0.705	0.655	0.611	0.388
	1	2	3	4



The Bellman Equations

- Definition of “optimal utility” leads to a simple one-step lookahead relationship amongst optimal utility values:



Optimal rewards = maximize over first action and then follow optimal policy

- Formally:

$$V^*(s) = \max_a Q^*(s,a)$$

$$Q^*(s,a) = \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V^*(s')\right]$$

$$V^*(s) = \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V^*(s')\right]$$

Why Not Search Trees?

- Why not solve with expectimax?
- Problems:
  - This tree is usually infinite (why?)
  - Same states appear over and over (why?)
  - We would search once per state (why?)
- Idea: Value iteration
  - Compute optimal values for all states all at once using successive approximations
  - Will be a bottom-up dynamic program similar in cost to memoization
  - Do all planning offline, no replanning needed!