Artificial Intelligence Programming

Markov Decision Processes

Chris Brooks

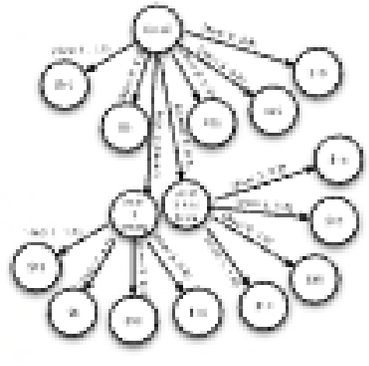
Department of Computer Science
University of San Francisco

Making Sequential Decisions

- Previously, we've talked about:
  - Making one-shot decisions in a deterministic environment
  - Making sequential decisions in a deterministic environment
    - Search
    - Inference
    - Planning
  - Making one-shot decisions in a stochastic environment
    - Probability and Belief Networks
    - Expected Utility
- What about sequential decisions in a stochastic environment?

Sequential Decisions

- We've thought a little bit about this in terms of value of information.
- We can model this as a state-space problem.
- We can even use a minimax-style approach to determine the optimal actions to take.



Expected Utility

- Recall that the expected utility of an action is the utility of each possible outcome, weighted by the probability of that outcome occurring.
- More formally, from state s, an agent may take actions $a_1, a_2, \ldots, a_n$.
- Each action a_i can lead to states $s_{a_1}, s_{a_2}, \ldots, s_{a_m}$, with probability $p_{a_1}, p_{a_2}, \ldots, p_{a_m}$
- $$EU(a_i) = \sum p_{ij} s_{aj}$$
- We call the set of probabilities and associated states the state transition model.
- The agent should choose the action a' that maximizes EU.

Markovian environments

- We can extend this idea to sequential environments.
- Problem: How to determine transition probabilities?
  - The probability of reaching state s given action a might depend on previous actions that were taken.
  - Reasoning about long chains of probabilities can be complex and expensive.
- The Markov assumption says that state transition probabilities depend only on a finite number of parents.
- Simplest: a first-order Markov process. State transition probabilities depend only on the previous state.
  - This is what we'll focus on.

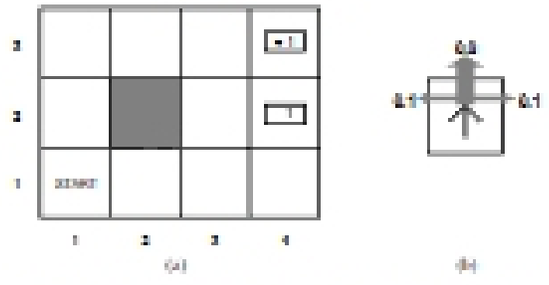
Stationary Distributions

- We'll also assume a ***stationary distribution***
- This says that the probability of reaching a state s' given action a from state s with history M does not change.
- Different histories may produce different probabilities
- Given identical histories, the state transitions will be the same.
- We'll also assume that the utility of a state does not change throughout the course of the problem.
  - In other words, our model of the world does not change while we are solving the problem.

Solving sequential problems

- If we have to solve a sequential problem, the total utility will depend on a sequence of states $s_1, s_2, \ldots, s_n$.
- Let's assign each state a reward $R(s_i)$.
- Agent wants to maximize the sum of rewards.
- We call this formulation a Markov decision process.
  - Formally:
  - An initial state s_0
  - A discrete set of states and actions
  - A Transition model: $T(s, a, s')$ that indicates the probability of reaching state s' from s when taking action a.
  - A reward function: $R(s)$

Example grid problem



- Agent moves in the "intended" direction with probability 0.8, and at a right angle with probability 0.2
- What should an agent do at each state to maximize reward?

MDP solutions

- Since the environment is stochastic, a solution will not be an action sequence.
- Instead, we must specify what an agent should do in any reachable state.
- We call this specification a *policy*
  - "If you're below the goal, move up."
  - "If you're in the left-most column, move right."
- We denote a policy with π, and $\pi[s]$ indicates the policy for state s.

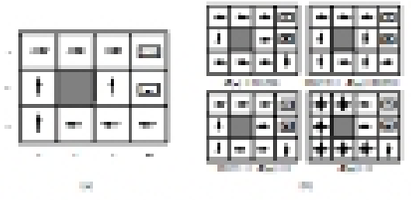
MDP solutions

- Things to note:
  - We've wrapped the goal formulation into the problem
    - Different goals will require different policies.
  - We are assuming a great deal of (correct) knowledge about the world.
    - State transition models, rewards
    - We'll touch on how to learn these without a model.

Comparing policies

- We can compare policies according to the expected utility of the histories they produce.
- The policy with the highest expected utility is the *optimal policy*.
- Once an optimal policy is found, the agent can just look up the best action for any state.

Example grid problem



Left figure: $R(s) = -0.04$. Note: there are typos in this figure; all non-zero rewards should be negative.

- As the costs for nonterminal states change, so does the optimal policy.
- Very high cost: Agent tries to exit immediately
- Middle ground: Agent tries to avoid bad exit
- Positive reward: Agent doesn't try to exit.

More on reward functions

- In solving an MDP, an agent must consider the value of future actions.
- There are different types of problems to consider:
- Horizon - does the world go on forever?
  - Finite horizon: after N actions, the world stops and no more reward can be earned.
  - Infinite horizon; World goes on indefinitely, or we don't know when it stops.
    - Infinite horizon is simpler to deal with, as policies don't change over time.

More on reward functions

- We also need to think about how to value future reward.
- $100 is worth more to me today than in a year.
- We model this by *discounting* future rewards.
  - γ is a *discount factor*
- $U([s_0, s_1, s_2, s_3, \ldots) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \gamma^3 R(s_3) + \ldots, \gamma \in [0, 1]$
- If γ is large, we value future states
- If γ is low, we focus on near-term reward
- In monetary terms, a discount factor of γ is equivalent to an interest rate of $(1/\gamma) - 1$

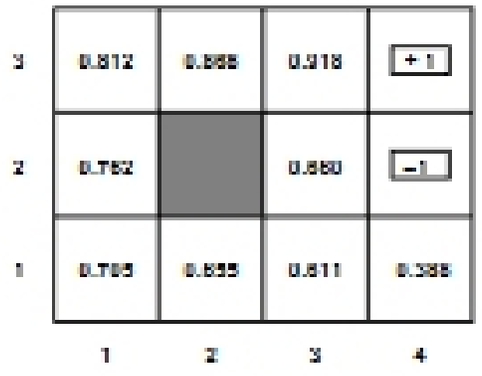
More on reward functions

- Discounting lets us deal sensibly with infinite horizon problems.
  - Otherwise, all EUs would approach infinity.
- Expected utilities will be finite if rewards are finite and bounded and $\gamma < 1$.
- We can now describe the optimal policy π^* as:
- $$\pi^* = \arg\max_\pi EU(\sum_{t=0}^{\infty} \gamma^t R(s_t) | \pi)$$

Value iteration

- How to find an optimal policy?
- We'll begin by calculating the expected utility of each state and then selecting actions that maximize expected utility.
- In a sequential problem, the utility of a state is the expected utility of all the state sequences that follow from it.
- This depends on the policy π being executed.
- Essentially, $U(s)$ is the expected utility of executing an optimal policy from state s.

Utilities of States



- Notice that utilities are highest for states close to the +1 exit.

Utilities of States

- The utility of a state is the immediate reward for that state plus the expected discounted utility of the next state, assuming that the agent chooses the optimal action.
- $$U(s) = R(s) + \gamma \max_a \sum_{s'} T(s, a, s') U(s')$$
- This is called the Bellman equation
- Example:

$$U(1,1) = -0.04 + \gamma \max(0.8U(1,2) + 0.1U(2,1) + 0.1U(1,1$$
$$0.9U(1,1) + 0.1U(1,2)$$
$$0.9U(1,1) + 0.1U(2,1)$$
$$0.8U(2,1) + 0.1U(1,2) + 0.1U(1,1)$$