**Markov Systems with Rewards, Markov Decision Processes**

Manuela Veloso
(Thanks to Reid Simmons and Andrew Moore)
Grad AI, Spring 2012

---

**Search and Planning**

- Planning
  - Deterministic state, preconditions, effects
  - Uncertainty
    - Conditional planning, conformant planning, nondeterministic
- Probabilistic modeling of systems with uncertainty and rewards
- Modeling probabilistic systems with control, i.e., action selection
- Reinforcement learning

---

**Example – Markov System with Reward**

\[ V^*(s_i) = r_i + \gamma \left[ p_{i1} V^*(s_1) + p_{i2} V^*(s_2) + \ldots + p_{in} V^*(s_n) \right] \]

- States
- Rewards in states
- Probabilistic transitions between states
- Markov: transitions only depend on current state

---

**Markov Systems with Rewards**

- Finite set of \( n \) states, \( s_i \)
- Probabilistic state matrix, \( P, p_{ij} \)
- "Goal achievement" - Reward for each state, \( r_i \)
- Discount factor - \( \gamma \)
- Process/observation:
  - Assume start state \( s_i \)
  - Receive immediate reward \( r_i \)
  - Move, or observe a move, randomly to a new state according to the probability transition matrix
  - Future rewards (of next state) are discounted by \( \gamma \)

---

**Solving a Markov System with Rewards**

- \( V^*(s_i) \) - expected discounted sum of future rewards starting in state \( s_i \)
- \( V^*(s_i) = r_i + \gamma [p_{i1} V^*(s_1) + p_{i2} V^*(s_2) + \ldots + p_{in} V^*(s_n)] \)

---

**Value Iteration to Solve a Markov System with Rewards**

- \( V^*(s_i) \) - expected discounted sum of future rewards starting in state \( s_i \) for one step.
- \( V^*(s_i) \) - expected discounted sum of future rewards starting in state \( s_i \) for two steps.
- \( \ldots \)
- \( V^*(s_i) \) - expected discounted sum of future rewards starting in state \( s_i \) for \( k \) steps.
- As \( k \to \infty \), \( V^*(s_i) \) approaches \( V^*(s_i) \)
- Stop when difference of \( k + 1 \) and \( k \) values is smaller than some \( \varepsilon \).
### Markov Decision Processes

- Finite set of states, $s_1, \ldots, s_n$
- Finite set of actions, $a_1, \ldots, a_m$
- Probabilistic state, action transitions:
  
  \[ p_{ij} = \text{prob} (\text{next} = s_j | \text{current} = s_i \text{ and take action } a_k) \]

  - Markov assumption: State transition function only depends on current state, not on the "history" of how the state was reached.
  - Reward for each state, $r_1, \ldots, r_n$
  - Process:
    - Start in state $s_i$
    - Receive immediate reward $r_i$
    - Choose action $a_k \in A$
    - Change to state $s_j$ with probability $p_{ij}$
    - Discount future rewards

### Nondeterministic Example

Reward and discount factor to be decided.
Note the need to have a finite set of states and actions.
Note the need to have all transition probabilities.
Solving an MDP

- Find an action to apply to each state.
- A policy is a mapping from states to actions.
- Optimal policy - for every state, there is no other action that gets a higher sum of discounted future rewards.
- For every MDP there exists an optimal policy.
- Solving an MDP is finding an optimal policy.
- A specific policy converts an MDP into a plain Markov system with rewards.

Value Iteration

- $V^*(s_i)$ - expected discounted future rewards, if we start from state $s_i$ and we follow the optimal policy.
- Compute $V^*$ with value iteration:
  \[ V^*(s_i) = \max_{a_i} \left\{ r_i + \gamma \sum_{j} p_{ij} V^*(s_j) \right\} \]
  - Dynamic programming

Policy Iteration

- Start with some policy $\pi_0(s_i)$.
- Such policy transforms the MDP into a plain Markov system with rewards.
- Compute the values of the states according to the current policy.
- Update policy:
  \[ \pi_{k+1}(s_i) = \arg \max_a \left\{ r_i + \gamma \sum_{j} p_{ij} V^*(s_j) \right\} \]
  - Keep computing
  - Stop when $\pi_{k+1} = \pi_k$.

Markov Models

- Plan is a Policy
  - Stationary: Best action is fixed
  - Non-stationary: Best action depends on time
- States can be discrete, continuous, or hybrid

<table>
<thead>
<tr>
<th>Plan Type</th>
<th>Passive</th>
<th>Controlled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully Observable</td>
<td>Markov Models</td>
<td>MDP</td>
</tr>
<tr>
<td>Hidden State</td>
<td>HMM</td>
<td>POMDP</td>
</tr>
<tr>
<td>Time Dependent</td>
<td>Semi-Markov</td>
<td>SMDP</td>
</tr>
</tbody>
</table>
Tradeoffs

• **MDPs**
  + Tractable to solve
  + Relatively easy to specify
    – Assumes perfect knowledge of state

• **POMDPs**
  + Treats all sources of uncertainty uniformly
  + Allows for taking actions that gain information
    – Difficult to specify all the conditional probabilities
    – hugely intractable to solve optimally

• **SMDO**
  + General distributions for action durations
    – Few good solution algorithms

Summary

• Markov Models with Reward
• Value iteration
• Markov Decision Process
• Value Iteration
• Policy Iteration
• Reinforcement Learning