Agent Reasoning

- Which perspective to take? (AAMAS 2009)
  - Logic
    - Theorem provers, Logical languages
  - Game Theory
    - Nash equilibrium
  - Social Theory
    - Voting, Altruism
  - Emergent Behavior
    - Swarm, Mechanism design

Overview

- Background
- Environment Models
- Decision Problem Models
- Multiagent Models

Background | Vocabulary

- States
  - Set of unique descriptions of environment
  - Combination of meaningful attributes

- Actions
  - Set of activities performed by agents

- Observations
  - Information provided by environment
  - Depends on state, possibly actions

State Transitions

- Change in state
- Depends on current state, possibly actions

History

- Sequence of observations
- Sometimes includes states/actions
**Background | Vocabulary**

- **Rewards**
  - Benefit to an agent
  - State/action dependent

- **Costs**
  - Negative effect on agent
  - State/action dependent

- **Utility**
  - Sum of current and future rewards
  - Finite or infinite horizon (if of steps)
  - Often discounted

**Background | Markov Processes**

- **Markov Property**
  - Current state depends only on previous
  - Future state depends only on current
  - 1st order Markov property

- **Markov process**
  - Stochastic process model with Markov assumption

- **Not perfect, but tractable**
  - “All models are wrong, some models are useful” --Dunbar

**Environment | Overview**

- **Environment Modeling**
  - Process independent of the agent

- **Markov Chains**

- **Hidden Markov Models**

**Observeability**

- **Identification of states**
  - Full
    - Agent always knows current state
    - E.g., robot with GPS

  - Partial
    - Current state hidden
    - Estimated by observations
    - E.g., robot with camera

**When to use**

- Want to model environment change
- Actions don’t change state of environment
- Or some changes for all actions
- Rewards/costs tied only to environment state

- **Fully observable**
  - Markov chain

- **Partially observable**
  - Hidden Markov model
**Environment | Markov Chain**

- **Markov Chain**
  - Simplest model of stochastic changes in environment
  - Handles non-determinism in state changes
  - Building block for other models

- **2-tuple \(<S, T>\)**
  - \(S\) = set of states
  - \(T(s, s') = P(s'|s)\) = state transition probabilities
  - Can include reward \(R(s)\) or cost \(C(s)\)

**Environment | HMM**

- **Hidden Markov Model**
  - Model stochastic environment with hidden states
  - Partially observable Markov chain
  - Handles incomplete information on states

- **4 tuple \(<S, \Omega, T, O>\)**
  - \(S\) and \(T\) as before
  - \(\Omega\) = set of observations
  - \(O(s', o) = P(o|s')\) = observation probabilities

---

**Environment | Markov Chain**

- **Wireless Network Modeling (Nguyen et al., 1996)**
  - Loss depends on outcome of previous packet
  - Accounts for "bursty" behavior

**Dishonest Casino Modeling (Durbin et al., 1998)**

- Casino uses two die
  - One fair, one loaded

Source: (Durbin et al., 1998)
Goal 1:
- Predict hidden state sequence from observations
  - Most probable path $p$

Use Viterbi algorithm
1. Initialize $v_i(0)$ values to 0, $v_i(0) = 1$
2. For each position $i$ in the sequence
   1. Calculate $v_i(i) = O(s', x_i) \max_{s \in S} v_{i-1}(i-1)T(s, x_i)$
   2. Calculate $ptr_i(s') = \arg \max_{s \in S} v_{i-1}(i-1)T(s, x_i)$
3. Build $p$ from $ptr_i$

Goal 2:
- Compute sequence probability $P(s)$

Use Forward algorithm
1. Initialize $f_i(0)$ values to 0, $f_i(0) = 1$
2. For each position $i$ in the sequence
   1. Calculate $f_i(i) = O(s', x_i) \sum_{s' \in S} f_i(i-1)T(s, x_i)$
3. Calculate $P(s) = \sum_{i=0}^{|s|} f_i(1)|T(s, x_i)$

Goal 3:
- Compute state probabilities $P(\pi_i = s | x_i) = \frac{f_i(i)b_i(i)}{P(x)}$

Use Backward algorithm
1. Initialize $b_i(|x|)$ values to $T(s, s_0)$
2. For each position $i$ in the sequence backwards
   1. Calculate $b_i(i) = \sum_{i \neq s'} b_i(i+1)T(s, x_i)O(s', x_i)$
3. Calculate $P(s) = \sum_{i \neq s'} b_i(1)T(s, x_i)O(s', x_i)$

NS($s, s'$) = $\sum_{i \neq S} \frac{1}{P(x)} \sum_{s \in S} T(s, x_i)O(s', x_i)f_i^i(i)b_i^i(i+1)$

NO($s', o$) = $\sum_{i \neq S} \frac{1}{P(x)} \sum_{i \neq s'} f_i^i(i)b_i^i(i)$

Decision Problem Modeling
- Depend on actions taken by agent

Markov Decision Process (MDP)

Partially Observable MDP (POMDP)
### Decision | Overview

- **When to use**
  - Want to model effect of agent on environment
  - Need to compute policy of actions
  - Actions do change state of environment
  - Rewards/costs tied to environment state and actions

- **Fully observable**
  - MDP
- **Partially observable**
  - POMDP

### Decision | MDP

- **Markov Decision Process**
  - Model of agent influence on stochastic environment
  - Extends Markov chain with actions

  - **4 tuple** \( <S, A, T, R> \)
    - \( S \) as before
    - \( A = \text{set of actions} \)
    - \( T(s, a, s') = P(s'|s, a) \)
    - \( R(s, a) = \text{reward for performing} \ a \ \text{in} \ s \)

### Decision | MDP

- **Goal**
  - Build a controller for agent actions
  - Generates a policy \( \pi \) mapping states to actions
  - Choose actions which maximize utility

- **Value functions (Bellman equations)**
  - Discounted infinite-horizon
    - \( V_k(s) = R(s, \pi(s)) + \gamma \sum_{s'} P(s'|s, \pi(s))V_{k+1}(s') \)
  - Current reward
  - Discounted future reward
  - Finite-horizon
    - \( V_k(s) = R(s, \pi(s)) + \gamma \sum_{s'} P(s'|s, \pi(s))V_{k+1}(s') \)

  Source: (Kaelbling et al., 1998)

### Decision | MDP

- **Choosing sensing activities with stateful resources**
  - States: energy in sensor
  - 3 actions: Advanced, Basic, Wait
  - Best reward with Advanced in High

<table>
<thead>
<tr>
<th>States</th>
<th>Markov Chain</th>
<th>HMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actions</td>
<td>MDP</td>
<td>POMDP</td>
</tr>
</tbody>
</table>

### Decision | MDP

- **Value iteration algorithm**
  - \( V_0(s) = 0 \) for all \( s \)
  - \( r = 1 \)
  - \( Q^k(s, a) \)
  - \( Q^k(s, a) = T(s, a, r) + \gamma \sum_{s'} P(s'|s, a)Q^k(s', a) \)
  - \( V^k(s) \)
  - \( V^k(s) = \max_a Q^k(s, a) \)
  - \( \epsilon \)

  Until \( |V^k(s) - V^{k-1}(s)| < \epsilon \) for all \( s \) in \( S \)

- **Policy iteration algorithm**
  - Tracks best action for each state

  Source: (Kaelbling et al., 1998)
Learn model
- Model-based reinforcement learning (RL)

RMax algorithm (Brafman and Tennenholtz, 2002)
- Count state transitions as in Markov chains
  - Fully observable
  - Save (count) rewards
  - Assume initial rewards maximal
  - Enforce exploration vs. exploitation

Partially Observable MDP
- Model of agent influence on hidden states
- Mixes HMM with MDP

6 tuple <S, A, Ω, T, O, R>
- S, A, T, R as in MDP
- Ω as in HMM
- O(s', a, o) = P(o | s', a)

User Preference Elicitation (Doshi and Roy, 2008)
- States = User goal (hidden)
- Actions = Query/Confirm/Support
- Observations = User response
- Cost to sensing, rewards for correct support

Goal: similar to MDP
- Build a policy τ(b) mapping belief states to actions
- Maximize expected utility

Belief states b(s)
- Probabilities of being in environment states given observation and last belief state
- Determined by State Estimator
  \[ b'(s') = \frac{O(s', a, o) \sum_{s,a} T(s, a, s')b(s)}{P(\tau(a, b))} \]

Approach: build an MDP on belief states
- S = B
- A
- \[ T(b, a, b') = P(b' | b, a) \]
- Defined by State Estimator
- \[ R(b, a) = \sum_{s} b(s)R(s, a) \]

Better approach: Policy trees
- Choose actions based on observations
  - Conditional plans
- Define value function over trees
  \[ V_{\pi}(s) = R(s, a(p)) + \gamma \sum_{s'} T(s, a(p), s') \sum_{a} O(s', a(p), a)V_{\pi}(s') \]
  - Expected utility of following tree
  - Optimize to build plan
  - One tree per state
  - Still exponential complexity

Agent

Source: (Kaelbling et al., 1998)
Most approaches: approximation algorithms

PBVI (Pineau et. al, 2003)
- Estimate value function for sampled belief states
- Each corresponds to an action
- Find closest belief state, pick best action

Online approaches (Ross et al., 2008)
- Limited depth trees
- Heuristic search (Ross & Chaib-draa, 2007)

Learn model
- Model-based partially observable reinforcement learning (PORL)
- Perceptual Distinctions (Chrisman, 1992)
  - Applied Baum-Welch to POMDPs
- Bayes Adaptive POMDP (Ross et. al, 2007)
  - “Meta-POMDP” approach
  - Possible POMDPs are states
  - Maintain belief state over possible models

Multiagent Processes
- Multiple agents change environment

Decentralized MDP

Stochastic Games

When to use
- Want to model effect of multiple agents on environment
  - Need to compute policy of actions for each agent
  - Each agent’s actions change state of environment
  - Rewards/costs tied to environment state and actions

Cooperative
- Decentralized MDP

Competitive
- Stochastic Games

Decentralized MDP/POMDP
- Model of multiple agents’ influence on stochastic environment
- Extends MDP/POMDP to multiple cooperating agents

Fully observable
- Agents combined know the true state
- Each agent might only have incomplete information
- May require communication

Partially observable
- Combined observations does not yield state
Multiagent | Decentralized MDP

- Difficult problem
  - \( \text{NEXP-hard} \) (Bernstein et al., 2002)

- Heuristic/Approximate solutions
  - Value Function Propagation (Marecki and Tambe, 2007)
  - Single-agent semi-MDPs with communication (Goldman and Zilberstein, 2008)

Multiagent | Stochastic Games

- Stochastic Games
  - Model of competitive agents in a stochastic environment
  - MDP is a single agent stochastic game

- Similar model as Decentralized MDP
  - Agents maximize own rewards (selfish)
  - Don’t share information

- Partially observable: Bayesian Games

Multiagent | Stochastic Games

- Goal: develop a strategy governing behavior
  - Similar to policy in MDPs

- Look for Nash equilibrium
  - No agent can do better with any other choice

- Rely on properties of environment
  - Zero-sum game
  - Discounted rewards
  - Stationarity

Conclusion | Summary

- Markov Processes (discrete state/time)
  - Model stochastic environment
  - Can handle incomplete information

- Environment models
  - Markov chain, HMM

- Decision problem models
  - MDP, POMDP

- Multiagent models
  - Decentralized MDP, Stochastic games
Conclusion | IAMAS Library

- Hidden Markov Models
  - Viterbi, Forward, Backward, Baum-Welch
- MDP
- RMax
- POMDP
  - Policy trees, PBVI, BAPOMDP
- Data structures and tools for other models

Questions?

- Which models might be applicable to final projects?
  - Poker playing agent?
  - Agents on mars?
- Can we incorporate ULM features?
  - Or vice-versa?
- How sufficient is MDP-based reasoning as a theory for agent control?

General References

- Markov Chains
- Hidden Markov Models
- Markov Decision Processes
- Multiagent Models

Other References

- C. Blum, 1992, Reinforcement learning with perceptual aliasing: the perceptual distinctions approach, Proc. of AAAI’92.