MARKOV PROCESSES TUTORIAL

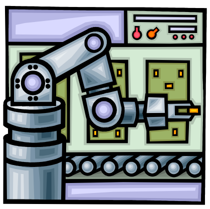
Nebraska Lincoln
Adam Eck
January 25, 2011

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

Agent Reasoning

- Problems with reasoning
  - Stochastic environments
  - Limited information
- Real-world applications
  - Autonomous robots
  - E-commerce
  - Decision support systems
  - Industrial control



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

Background | Vocabulary

- 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 (# of steps)
  - Often discounted

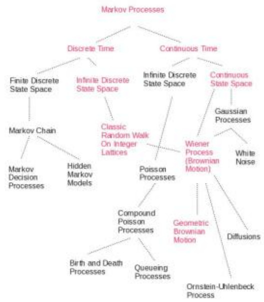
Background | Vocabulary

- Observability
  - 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

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

Background | Markov Processes



Source: (Dunbar, 2010)

Environment | Overview

- Environment Modeling
  - Process **independent** of the agent
- Markov Chains
- Hidden Markov Models

Environment | Overview

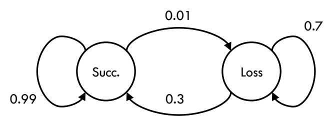
- When to use
  - Want to model environment change
  - Actions don't change state of environment
    - Or same changes for all actions
  - Rewards/costs tied only to environment state
- Fully observable
  - Markov chain
- Partially observable
  - Hidden Markov model

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 | Markov Chain

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



	Success	Loss
Success	0.99	0.01
Loss	0.3	0.7

Environment | Markov Chain

- Goal
  - Compute P(s' | s) for future states
    - Can be more than one step in the future
- Chapman-Kolmogorov Equations
  - Use dynamic programming

$$T^n(s,s') = \sum_{s^* \in S} T^m(s,s^*) T^{n-m}(s^*,s')$$

Environment | Markov Chain

- Learn model
  - Count state transitions
    - NS(s, s') = # of transitions from s to s'
    - Fully observable
  - Dirichlet distribution
  - Probability from proportions

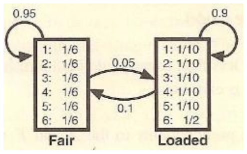
$$T(s,s') = NS(s,s') \Big/ \sum_{s^* \in S} NS(s,s^*)$$

Environment | HMM

- Hidden Markov Model
  - Model stochastic environment with **hidden** states
  - Partially observable Markov chain
    - Handles incomplete information on states
- 4 tuple <S, Ω, T, O >
  - S and T as before
  - Ω = set of observations
  - O(s', o) = P(o | s') = observation probabilities

Environment | HMM

- Dishonest Casino Modeling (Durbin *et. al*, 1998)
  - Casino uses two die
    - One fair, one loaded



Source: (Durbin *et. al*, 1998)

Environment | HMM

- Goal 1:
  - Predict hidden state sequence from observations
    - Most probable path p
- Use Viterbi algorithm

1. Initialize $v_s(0)$ values to 0, $v_{s0}(0) = 1$
2. For each position i in the sequence
   1. Calculate $v_{s'}(i) = O(s', x_i) \max_{s \in S} v_s(i-1) T(s, s')$
   2. Calculate $ptr_i(s') = \arg\max_{s \in S} v_s(i-1) T(s, s')$
3. Build p from *ptr*

Environment | HMM

- Goal 2:
  - Compute sequence probability $P(x)$
- Use Forward algorithm

1. Initialize $f_s(0)$ values to 0, $f_{s0}(0) = 1$
2. For each position i in the sequence
   1. Calculate $f_{s'}(i) = O(s', x_i) \sum_{s \in S} f_s(i-1) T(s, s')$
3. Calculate $P(x) = \sum_{s \in S} f_s(|x|) T(s_0, s)$

Environment | HMM

- Goal 3:
  - Compute state probabilities $P(\pi_i = s \mid x) = \frac{f_s(i) b_s(i)}{P(x)}$
- Use Backward algorithm

1. Initialize $b_s(|x|)$ values to $T(s, s_0)$
2. For each position i in the sequence backwards
   1. Calculate $b_s(i) = \sum_{s' \in S} b_{s'}(i+1) T(s, s') O(s', x_{i+1})$
3. Calculate $P(x) = \sum_{s' \in S} b_{s'}(1) T(s_0, s') O(s', x_1)$

Environment | HMM

- Learn model:
  - Baum-Welch algorithm

1. Initialize a random model
2. While not converged
   1. For each sequence x^j in X
      1. Run Forward algorithm on x^j
      2. Run Backward algorithm on x^j
      3. Update NS(s, s') and NO(s', o)
   2. Compute new model
   3. Calculate model likelihood

Environment | HMM

$$NS(s, s') = \sum_{x^j \in X} \frac{1}{P(x^j)} \sum_i T(s, s') O(s', x_i) f_s^j(i) b_{s'}^j(i+1)$$

$$NO(s', o) = \sum_{x^j \in X} \frac{1}{P(x^j)} \sum_{\{i \mid x_i^j = o\}} f_{s'}^j(i) b_{s'}^j(i)$$

Decision | Overview

- 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 | Overview

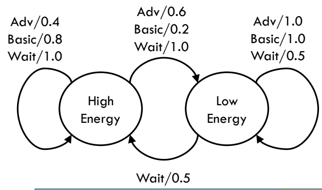
	Fully Observable	**Partially Observable**
States	Markov Chain	HMM
Actions	MDP	POMDP

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 = set of actions
  - T(s, a, s') = P(s' | s, a)
  - R(s,a) = reward for performing a in s

Decision | MDP

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



	High	Low
High/Adv	0.4	0.6
High/Basic	0.8	0.2
High/Wait	1.0	1.0
Low/Adv	0.0	1.0
Low/Basic	0.0	1.0
Low/Wait	0.5	0.5

Decision | MDP

- Goal
  - Build a controller for agent actions
    - Generates a policy π mapping states to actions
  - Choose actions which maximize utility
- Value functions (Bellman equations)
  - Discounted infinite-horizon

$$V_\pi(s) = \underbrace{R(s,\pi(s))}_{\text{Current reward}} + \underbrace{\gamma \sum_{s' \in S} P(s'|s,\pi(s)) V_\pi(s')}_{\text{Discounted future reward}}$$

  - Finite-horizon

$$V_{\pi,t}(s) = R(s,\pi_t(s)) + \gamma \sum_{s' \in S} P(s'|s,\pi_t(s)) V_{\pi,t-1}(s')$$

Source: (Kaelbling *et. al*, 1998)

Decision | MDP

- Value iteration algorithm

$V_1(s) := 0$ for all s
$t := 1$
loop
  $t := t + 1$
  loop for all $s \in \mathcal{S}$
    loop for all $a \in \mathcal{A}$
      $Q_t^a(s) := R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s,a,s') V_{t-1}(s')$
    end loop
    $V_t(s) := \max_a Q_t^a(s)$
  end loop
until $|V_t(s) - V_{t-1}(s)| < \varepsilon$ for all $s \in \mathcal{S}$

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

Source: (Kaelbling *et. al*, 1998)

Decision | MDP

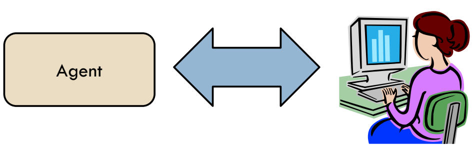
- Learn model
  - Model-based reinforcement learning (RL)
- RMax algorithm (Brafman and Tennenholtz, 2002)
  - Count state transitions as in Markov chains
    - Fully observable
  - Save (count) rewards
  - Assume initial rewards maximal
    - Enforce exploration vs. exploitation

Decision | POMDP

- 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)

Decision | POMDP

- 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



Decision | POMDP

- Goal: similar to MDP
  - Build a policy π(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

$$b'(s') = \frac{O(s',a,o)\sum_{s\in S} T(s,a,s')b(s)}{\Pr(o \mid a,b)}$$

Source: (Kaelbling *et. al*, 1998)

Decision | POMDP

- 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) = $\sum_{S_{POMDP}} b(s)R(s,a)$
- Problem: continuous state MDP
  - States are probability distributions
  - Very difficult to solve (uncountably infinite number of states)
    - State space is $\mathbb{R}^{|S_{POMDP}|}$

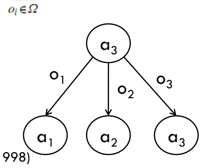
Source: (Kaelbling *et. al*, 1998)

Decision | POMDP

- Better approach: Policy trees
  - Choose actions based on observations
    - Conditional plans
  - Define value function over trees

$$V_p(s) = R(s,a(p)) + \gamma \sum_{s'\in S} T(s,a(p),s') \sum_{o_i\in\Omega} O(s',a(p),o_i)V_{o_i(p)}(s')$$

    - Expected utility of following tree
    - Optimize to build plan
  - One tree per state
  - **Still exponential complexity**



Source: (Kaelbling *et. al*, 1998)

Decision | POMDP

- 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)

Decision | POMDP

- 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 | Overview

- Multiagent Processes
  - **Multiple** agents change environment
- Decentralized MDP
- Stochastic Games

Multiagent | Overview

- 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

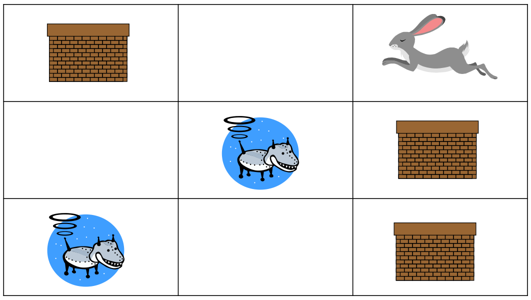
Multiagent | Decentralized MDP

- Decentralized MDP/POMDP
  - Model of **multiple** agents' influence on stochastic environment
  - Extends MDP/POMDP to multiple **cooperating** agents
- Similar model as MDP/POMDP
  - Shared S, T, Ω between agents
  - Same or different A, R for each agent
  - T, O, R depend on each agent's actions

Multiagent | Decentralized MDP

- 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



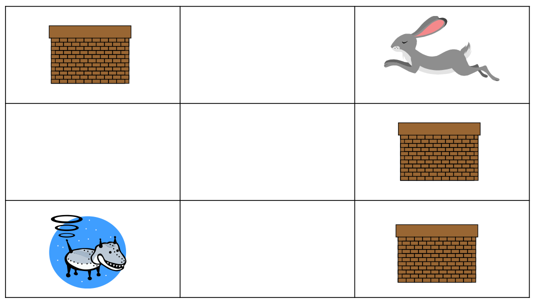
Multiagent | Decentralized MDP

- Difficult problem
  - 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



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?

Conclusion | Discussion

- 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
  - R. Durbin, S. Eddy, A. Krogh, and G. Mitchison, 1998, *Biological Sequence Analysis,* Cambridge University Press.
  - F.S. Hillier and G.J. Lieberman, 2005, *Introduction to Operations Research*, 8th Edition, McGraw Hill.
- Hidden Markov Models
  - R. Durbin, S. Eddy, A. Krogh, and G. Mitchison, 1998, *Biological Sequence Analysis,* Cambridge University Press.
- Markov Decision Processes
  - L.P. Kaelbling, M.L. Littman, and A.R. Cassandra, Planning and acting in partially observable stochastic domains, *Artificial Intelligence*, vol. 101, pp. 99-134, 1998.
- Multiagent Models
  - D.S. Berstein, R. Givan, N. Immerman, S. Zilberstein. 2002. The complexity of decentralized control of Markov decision processes. Mathematics of Operations Research. 27(4). 819-840.
  - J. Filar and K. Vrieze, 1997, *Competitive Markov decision processes*, Springer.
  - Y. Shoham and K. Leyton-Brown, 2009, *Multiagent Systems: Algorithmic, game-theoretic, and logical foundations*, Cambridge University Press.

Other References

- R.I. Brafman and M. Tennenholtz, 2002, R-max – A general polynomial time algorithm for near-optimal reinforcement learning, *Journal of Machine Learning Research*, 3, 213-231.
- L. Chrisman, 1992, Reinforcement learning with perceptual aliasing: the perceptual distinctions approach. *Proc. of AAAI'92*.
- F. Doshi and N. Roy, 2008, The permutable POMDP: fast solutions to POMDPs for preference elicitation, *Proc. of AAMAS'08*, 493-500.
- S. Dunbar, 2010, Stochastic Processes, http://www.math.unl.edu/~sdunbar1/MathematicalFinance/Lessons/Background/StochasticProcesses/stochasticprocesses.xml, accessed on January 21, 2011.
- C.V. Goldman and S. Zilberstein, 2008, Communication-based decomposition mechanisms for decentralized MDPs, *JAIR*, 32, 169-202.
- J. Marecki and M. Tambe, 2007, On opportunistic techniques for solving decentralized Markov decision processes with temporal constraints, *Proc. of AAMAS'07*.
- G.T. Nguyen, R.H. Katz, B. Nobel, and M. Satyanarayanan, A trace-based approach for modeling wireless channel behavior, *Proc. 1996 Winter Simulation Conf.,* ed. J.M. Charnes, D.J. Morrice, D.T. Brunner, and J.J. Swain, Coronado, CA, pp. 597-604, Dec. 8-11, 1996.
- J. Pineau, G. Gordon, and S. Thrun, 2003, Point-based value iteration: An anytime algorithm for POMDPs, *Proc. of IJCAI'03,* 1025-1032.
- S. Ross and B. Chaib-draa, Aems: An anytime online search algorithm for approximate policy refinement in large POMDPs, *Proc. of IJCAI '07*, pp. 2592-2598, 2007.
- S. Ross, B. Chaib-draa, and J. Pineau, 2007, Bayes-adaptive POMDPs, *Proc. of NIPS'07*.
- S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa, Online planning algorithms for POMDPs, *Journal of Artificial Intelligence Research*, vol. 32, pp. 663-704, 2008.