Model-Based Bayesian Reinforcement Learning for Real-World Domains

Joelle Pineau

School of Computer Science, McGill University, Canada

March 7 2008
Motivation

We are currently building robotic systems which must deal with:

- noisy sensing of their environments,
- observations that are discrete/continuous, structured,
- poor model of sensors and actuators.

E.g.

- SmartWheeler
- Deep-brain stimulation

[Pineau et al., 2007]

[Guez et al., 2008]

Despite all this, we expect the robot to choose actions which maximize long-term rewards!
Motivation

Typical way of solving such problem:

- **Supervised Learning**: Learn model from data, then plan with the learned model.
  - Hard to learn every possible cases (⇒ poor model).
  - Requires a lot of data.
  - Cost of learning not taken into account.

- **Reinforcement Learning**: Learn directly how to act, through trial-and-error interactions with the environment.
  - Requires a lot of data.
  - Exploration-exploitation problem.
  - Limited success/applicability in partially observable domains.
Motivation

What do we need for tackling real-world problems?

- Ability to input prior knowledge explicitly.
- Learning from few data.
- Online model adaptation.
- Methods that can deal with:
  - partial state observability,
  - continuous data,
  - structured representations.
- Ability to maximize expected return based on current state of information.
General Idea:

- Define prior distributions over all unknown parameters.
- Update posterior via Baye’s rule as experience is acquired.
- Optimize action choice w.r.t. posterior distribution over model.

Allows us to:

- Include prior knowledge explicitly.
- Learn the system as necessary to accomplish the task.
- Consider model uncertainty during planning.
- Trade-off optimally between exploration and exploitation.
Markov Decision Processes

MDP Model Definition:

- $S$: Set of states
- $A$: Set of actions
- $T(s, a, s') = \Pr(s'|s, a)$, transition probabilities
- $R(s, a) \in \mathbb{R}$, the immediate reward
- $\gamma$: discount factor

Optimal policy obtained by solving:

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

How can we solve this if $T$ is unknown?
In Finite MDPs: ([Dearden et al. 99], [Duff 02], [Poupart et al. 06])

Maintain counts $\phi_{ss'}^a$ of number of times the transition $s \xrightarrow{a} s'$ is observed, starting from prior $\phi_0$.

Counts define Dirichlet prior/posterior over $T$.

Planning according to $\phi$ is an MDP problem itself:
- $S'$: physical state ($s \in S$) + information state ($\phi$)
- $T'$: describes probability of update $(s, \phi) \xrightarrow{a} (s', \phi')$

First part of the talk: Extend this to partially observable domains.
Second part of the talk: Extend this to structured domains.
Bayesian RL in Finite MDPs

In Finite MDPs: ([Dearden et al. 99], [Duff 02], [Poupart et al. 06])

Maintain counts $\phi_{ss'}^a$ of number of times the transition $s \xrightarrow{a} s'$ is observed, starting from prior $\phi_0$.

Counts define Dirichlet prior/posterior over $T$.

Planning according to $\phi$ is an MDP problem itself:

- $S'$: physical state ($s \in S$) + information state ($\phi$)
- $T'$: describes probability of update $\langle s, \phi \rangle \xrightarrow{a} \langle s', \phi' \rangle$

First part of the talk: Extend this to partially observable domains.
Second part of the talk: Extend this to structured domains.
Partially Observable Markov Decision Processes

POMDP Model Definition

- $S$: Set of states (*unobservable by the agent*)
- $A$: Set of actions
- $T(s, a, s') = \Pr(s'|s, a)$, transition probabilities
- $R(s, a) \in \mathbb{R}$, immediate rewards
- $\gamma$: discount factor
- $Z$: Set of observations
- $O(s', a, z) = \Pr(z|s', a)$, the observation probabilities
- $b_0(s)$: Initial state distribution

Belief monitoring via Bayes rule:

$$b_t(s') = \eta O(s', a_{t-1}, z_t) \sum_{s \in S} T(s, a_{t-1}, s') b_{t-1}(s)$$

Value function optimization:

$$V^*(b) = \max_{a \in A} \left[ R(b, a) + \gamma \sum_{z \in Z} \Pr(z|b, a) V^*(\tau(b, a, z)) \right]$$
Partially Observable Markov Decision Processes

**POMDP Model Definition**

- **$S$**: Set of states (*unobservable by the agent*)
- **$A$**: Set of actions
- **$T(s, a, s') = \Pr(s'|s, a)$**, transition probabilities
- **$R(s, a) \in \mathbb{R}$**, immediate rewards
- **$\gamma$**: discount factor
- **$Z$**: Set of observations
- **$O(s', a, z) = \Pr(z|s', a)$**, the observation probabilities
- **$b_0(s)$**: Initial state distribution

**Belief monitoring via Bayes rule**:

$$b_t(s') = \eta O(s', a_{t-1}, z_t) \sum_{s \in S} T(s, a_{t-1}, s') b_{t-1}(s)$$

**Value function optimization**:

$$V^*(b) = \max_{a \in A} \left[ R(b, a) + \gamma \sum_{z \in Z} \Pr(z|b, a) V^*(\tau(b, a, z)) \right]$$
**Partially Observable Markov Decision Processes**

<table>
<thead>
<tr>
<th>POMDP Model Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$: Set of states (<em>unobservable by the agent</em>)</td>
</tr>
<tr>
<td>$A$: Set of actions</td>
</tr>
<tr>
<td>$T(s, a, s') = \Pr(s'</td>
</tr>
<tr>
<td>$R(s, a) \in \mathbb{R}$, immediate rewards</td>
</tr>
<tr>
<td>$\gamma$: discount factor</td>
</tr>
<tr>
<td>$Z$: Set of observations</td>
</tr>
<tr>
<td>$O(s', a, z) = \Pr(z</td>
</tr>
<tr>
<td>$b_0(s)$: Initial state distribution</td>
</tr>
</tbody>
</table>

Belief monitoring via Bayes rule:

$$b_t(s') = \eta O(s', a_{t-1}, z_t) \sum_{s \in S} T(s, a_{t-1}, s') b_{t-1}(s)$$

Value function optimization:

$$V^*(b) = \max_{a \in A} \left[ R(b, a) + \gamma \sum_{z \in Z} \Pr(z|b, a) V^*(\tau(b, a, z)) \right]$$
Partially Observable Markov Decision Processes

**POMDP Model Definition**

- **$S$**: Set of states (*unobservable by the agent*)
- **$A$**: Set of actions
- **$T(s, a, s') = \Pr(s'|s, a)$**, transition probabilities
- **$R(s, a) \in \mathbb{R}$**, immediate rewards
- **$\gamma$**: discount factor
- **$Z$**: Set of observations
- **$O(s', a, z) = \Pr(z|s', a)$**, the observation probabilities
- **$b_0(s)$**: Initial state distribution

**Belief monitoring via Bayes rule**:

$$b_t(s') = \eta O(s', a_{t-1}, z_t) \sum_{s \in S} T(s, a_{t-1}, s') b_{t-1}(s)$$

**Value function optimization**:

$$V^*(b) = \max_{a \in A} \left[ R(b, a) + \gamma \sum_{z \in Z} \Pr(z|b, a) V^*(\tau(b, a, z)) \right]$$
In Finite POMDPs \((T, O)\) unknown:

Let:

- \(\phi_{ss'}^a\) : counts of \(s \xrightarrow{a} s'\)
- \(\psi_{sz}^a\) : counts of seeing \(z\) at \(s\) after doing \(a\).

\(\Rightarrow\) Decision problem over \((s, \phi, \psi)\).
Bayes-Adaptive POMDP Model ([Ross et al. NIPS’07])

- $S' = S \times \mathbb{N}^{|S|^2|A|} \times \mathbb{N}^{|S||A||Z|}$
- $A' = A$
- $Z' = Z$
- $Pr(s', \phi', \psi' | s, \phi, \psi, a, z) =$
  $$\sum_{s''' \in S} \sum_{z'' \in Z} \frac{\phi_{ss'}^a}{\psi_{ss'}^a} \frac{\psi_{s'z}^a}{l(\phi', \phi + \delta_{ss'}^a) l(\psi', \psi + \delta_{s'z}^a)}$$
- $R'(s, \phi, \psi, a) = R(s, a)$

**Goal:** Maximize return under partial observability of $(s, \phi, \psi)$. 
A few comments

About the Bayes-Adaptive MDP

- Defines an infinite-state MDP with a known model.
- The state is defined over \((s, \phi)\).
- At every time step, \(s\) is observable, and \(\phi\) is updated.

About the Bayes-Adaptive POMDP

- Defines an infinite-state POMDP with a known model.
- The state is defined over \((s, \phi, \psi)\).
- At every time step, \(s\) is not observable, so neither are \(\phi\) and \(\psi\).
How can we update counters $\phi$ and $\psi$, if we don’t observe $s$?

(Note: this is the basic problem for classical RL in partially observable environments.)
Belief in BAPOMDPs

Let
- $b_0$: initial belief over original state space
- $\phi_0, \psi_0$: initial counts (prior on $T, O$)

Initial belief of the BAPOMDP:

$$b'_0(s, \phi, \psi) = b_0(s)l(\phi, \phi_0)l(\psi, \psi_0)$$

Monitoring the belief:
- The belief defines a mixture of Dirichlets over $T, O$.
- Allows us to learn the unknown POMDP model.
- Computing $b_t$ exactly is in $O(|S|^t+1)$ - VERY LARGE!
We can bound the error introduced in the value function due to differences in model posteriors.

Theorem 1:

\[
\sup_{\alpha \in \Gamma_t, s \in S} |V_t^\alpha(s, \phi, \psi) - V_t^\alpha(s, \phi', \psi')| \leq \frac{2\gamma \|R\|_\infty}{(1 - \gamma)^2} \sup_{s, s', a \in A} \left[ D_s^{sa}(\phi, \phi') + D_s' a(\psi, \psi') \right. \\
\left. + \frac{4}{\ln(\gamma - e)} \left( \sum_{s'' \in S} \frac{\phi_{s''} a - \phi'_{s''} a}{(N_a^{sa} + 1)(N_a^{sa'} + 1)} + \sum_{z \in Z} \frac{\psi_{s''} a - \psi'_{s''} a}{(N_a^{sa} + 1)(N_a^{sa'} + 1)} \right) \right]
\]

where:

\[
N_a^{sa} = \sum_{s' \in S} \phi_{s's'}^a,
\]

\[
D_s^{sa}(\phi, \phi') = \sum_{s' \in S} \frac{\phi_{s's'}^a - \phi'_{s's'}^a}{N_a^{sa}}.
\]

\[
N_a^{sa'} = \sum_{z \in Z} \psi_{sz}^a,
\]

\[
D_s a(\psi, \psi') = \sum_{z \in Z} \frac{\psi_{sz}^a - \psi'_{sz}^a}{N_a^{sa}}.
\]
Now consider the following approximate BAPOMDP:

**Bounded-counts BAPOMDP:**

\[ M_\epsilon = (\tilde{S}_\epsilon, A, Z, \tilde{P}_\epsilon, \tilde{R}_\epsilon) \]

\[ \tilde{S}_\epsilon = S \times \{ \phi \in \mathbb{N}^{|S|^2|A|} \mid \forall s, a \cdot 0 < N^\phi_{sa} \leq N^\epsilon_S \} \times \{ \psi \in \mathbb{N}^{|S||A||Z|} \mid \forall s, a \cdot 0 < N^\psi_{sa} \leq N^\epsilon_Z \} \]

\[ \tilde{P}_\epsilon(s, \phi, \psi, a, s', \phi', \psi', z) = \frac{\phi_{ss'}^a \psi_{s'z'}^a}{\sum_{s''} \phi_{ss''}^a \sum_{z'} \psi_{s'z'}^a} I(\mathcal{P}_\epsilon(s', \phi + \delta_{ss'}^a, \psi + \delta_{s'z'}^a), (s', \phi', \psi')) \]

\[ \tilde{R}_\epsilon(s, \phi, \psi, a) = R(s, a) \]

where \( \mathcal{P}_\epsilon : S' \rightarrow \tilde{S}_\epsilon \) returns the state in \( \tilde{S}_\epsilon \) that minimizes the bound in Theorem 1.
We can bound the error introduced in the value function when we approximate the BAPOMDP by thresholding count vectors.

Theorem 2:

To achieve $|\tilde{V}_t^\alpha(\mathcal{P}_\epsilon(s, \phi, \psi)) - V_t^\alpha(s, \phi, \psi)| < \frac{\epsilon}{1-\gamma}$, where $\tilde{\alpha}_t$ is computed from $M_\epsilon$ and $\alpha_t$ is computed from $M$, define

$\epsilon' = \frac{\epsilon(1-\gamma)^2}{8\gamma||R||_\infty}$,

$\epsilon'' = \frac{\epsilon(1-\gamma)^2 \ln(\gamma^{-\theta})}{32\gamma||R||_\infty}$,

$N_S^\epsilon = \max \left( \frac{|S|(1+\epsilon')}{\epsilon'}, \frac{1}{\epsilon''} - 1 \right)$,

$N_Z^\epsilon = \max \left( \frac{|Z|(1+\epsilon')}{\epsilon'}, \frac{1}{\epsilon''} - 1 \right)$.
Problem: Computing $b_t$ exactly in a BAPOMDP is in $O(|S|^{t+1})$.

Use particle filters for efficient approximation of the belief:

- **Monte Carlo**: Perform belief update by sampling $K$ particles and state transitions.
- **K Most Likely**: After each belief update, keep only the $K$ particles with highest probability.
- **Weighted Distance Metric**: After each belief update, use a greedy algorithm to pick the $K$ particles which best fit the posterior (using the distance metric in Theorem 1).
Problem: Computing $b_t$ exactly in a BAPOMDP is in $O(|S|^{t+1})$.

Use particle filters for efficient approximation of the belief:

- **Monte Carlo**: Perform belief update by sampling $K$ particles and state transitions.
- **K Most Likely**: After each belief update, keep only the $K$ particles with highest probability.
- **Weighted Distance Metric**: After each belief update, use a greedy algorithm to pick the $K$ particles which best fit the posterior (using the distance metric in Theorem 1).
Problem: Computing $b_t$ exactly in a BAPOMDP is in $O(|S|^{t+1})$.

Use particle filters for efficient approximation of the belief:

- **Monte Carlo**: Perform belief update by sampling $K$ particles and state transitions.
- **K Most Likely**: After each belief update, keep only the $K$ particles with highest probability.
- **Weighted Distance Metric**: After each belief update, use a greedy algorithm to pick the $K$ particles which best fit the posterior (using the distance metric in Theorem 1).
Problem: Computing $b_t$ exactly in a BAPOMDP is in $O(|S|^t+1)$.

Use particle filters for efficient approximation of the belief:

- **Monte Carlo**: Perform belief update by sampling $K$ particles and state transitions.
- **K Most Likely**: After each belief update, keep only the $K$ particles with highest probability.
- **Weighted Distance Metric**: After each belief update, use a greedy algorithm to pick the $K$ particles which best fit the posterior (using the distance metric in Theorem 1).
We still need to optimize a policy:

$$Pr(s, \phi, \psi | \phi_0, \psi_0, a_1, z_1, \ldots, a_{t-1}, z_{t-1}) \rightarrow a$$

This involves solving an infinite-state POMDP!

- It can be solved exactly for finite horizons given prior $$(\phi_0, \psi_0)$$.

Monte Carlo Online Planning (Receding Horizon Control):
Experimental Results

Follow:

- A robot has to follow an individual within a known environment.
- There are 2 possible individuals with different motion behaviors. The behaviors are unknown a priori.
- The individual changes at the beginning of each trajectory, and can only be identified by observations of the behavior.
Experimental Results

Expected return:

![Graph showing experiment results]
Experimental Results

**Model Accuracy:**

\[ WL1(b) = \sum_{(s, \phi, \psi) \in S'_b} b(s, \phi, \psi) \sum_{a \in A} \sum_{s' \in S} \left[ \sum_{s \in S} | T^s_{s'} | + \sum_{z \in Z} | O_{s'az} - O_{s'az} | \right] \]
Experimental Results

Planning time:

![Planning Time/Action (ms) Graph](image-url)
Discussion

Bayes-Adaptive POMDP produces a policy trading-off between:
- Exploring to learn the model.
- Identifying the system’s state.
- Gathering rewards.

**Problem**: Many real-world domains (e.g. in robotics) are represented over continuous spaces (states, actions, observations).
Bayes-Adaptive POMDP produces a policy trading-off between:

- Exploring to learn the model.
- Identifying the system’s state.
- Gathering rewards.

**Problem**: Many real-world domains (e.g. in robotics) are represented over continuous spaces (states, actions, observations).
Can’t use Dirichlet counts to learn about the model.

Could discretize the problem and apply our current method, but:
- Combinatorial explosion in number of states (or else poor precision).
- Can require lots of training data (to visit every small cell).

**Solution**: Assume a more appropriate parametric form for the transition and observation model.
Consider the Gaussian case:

- \( S \subseteq \mathbb{R}^m, A \subseteq \mathbb{R}^n, Z \subseteq \mathbb{R}^p \)
- \( s_{t+1} = g_T(s_t, a_t, X_t) \)
- \( z_{t+1} = g_O(s_{t+1}, a_t, Y_t) \)

where \( X_t \sim N(\mu_X, \Sigma_X) \), \( Y_t \sim N(\mu_Y, \Sigma_Y) \), and \( g_T, g_O \) are arbitrary functions (possibly non-linear).

Assume \( g_T, g_O \) are known, but parameters \( \mu_X, \Sigma_X, \mu_Y, \Sigma_Y \) are unknown.

Parameters \( \mu, \Sigma \) can be learned by maintaining sample mean \( \hat{\mu} \) and sample covariance \( \hat{\Sigma} \).

These define a Normal-Wishart posterior over \( \mu, \Sigma \).
Bayesian RL in Continuous POMDP

Bayes-Adaptive Continuous POMDP: ([Ross et al. ICRA’08])

\[
S' = S \times \mathbb{R}^{\lvert X \rvert + \lvert X \rvert^2 + 2} \times \mathbb{R}^{\lvert Y \rvert + \lvert Y \rvert^2 + 2}
\]

\[
A' = A
\]

\[
Z' = Z
\]

\[
P'(s, \phi, \psi, a, s', \phi', \psi', z) =
I(g_T(s, a, x), s')I(g_O(s', a, y), z)I(\phi', \mathcal{U}(\phi, x))I(\psi', \mathcal{U}(\psi, y))f_{X|\phi}(x)f_{Y|\psi}(y)
\]

\[
R'(s, \phi, \psi, a) = R(s, a)
\]

where:

\[
\phi : \text{the posterior over } (\mu_X, \Sigma_X)
\]

\[
\psi : \text{the posterior over } (\mu_Y, \Sigma_Y)
\]

\[
\mathcal{U} : \text{the update function of } \phi, \psi, \text{i.e. } \mathcal{U}(\phi, x) = \phi' \text{ and } \mathcal{U}(\psi, y) = \psi'
\]
Simple Robot Navigation Task:

- **S**: \((x, y)\) position
- **A**: \((v, \theta)\) velocity \(v \in [0, 1]\) and angle \(\theta \in [0, 2\pi]\)
- **Z**: Noisy \((x, y)\) position
- \(g_T(s, a, X) = s + v \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} X\)
- \(g_O(s', a, Y) = s' + Y\)
- \(R(s, a) = I(||s - s_{GOAL}||_2 < 0.25)\)
- \(\gamma = 0.85\)
We choose exact parameters:

$$\mu_X = \begin{bmatrix} 0.8 \\ 0.3 \end{bmatrix} \quad \Sigma_X = \begin{bmatrix} 0.04 & -0.01 \\ -0.01 & 0.01 \end{bmatrix}$$

$$\mu_Y = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \Sigma_Y = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix}$$

And start with a prior based on 10 "artificial" samples:

$$\hat{\mu}_X = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \hat{\Sigma}_X = \begin{bmatrix} 0.04 & -0.01 \\ -0.01 & 0.16 \end{bmatrix}$$

$$\hat{\mu}_Y = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \hat{\Sigma}_Y = \begin{bmatrix} 0.16 & 0 \\ 0 & 0.16 \end{bmatrix}$$

such that $\phi_0 = (\hat{\mu}_X, 10, 9, 9\hat{\Sigma}_X)$, $\psi_0 = (\hat{\mu}_Y, 10, 9, 9\hat{\Sigma}_Y)$.

Each time the robot reaches the goal, a new goal is chosen randomly at 5 distance units from previous goal.
Robot Navigation Task

Average evolution of the return over time:

![Graph showing the average evolution of the return over time for different models: Prior model, Exact Model, and Learning. The x-axis represents training steps, and the y-axis represents the average return.]
Robot Navigation Task

Average accuracy of the model over time:

![Graph showing the average accuracy of the model over training steps. The y-axis represents accuracy ranging from 0 to 1, and the x-axis represents training steps ranging from 0 to 250. The graph shows a curve that starts near 1 and decreases as training steps increase.]
Discussion

Bayesian learning can be extended to continuous state/action/observation spaces.

It is important to choose an appropriate parameterization (but not limited to linear Gaussian).

Monte Carlo methods are needed to achieve tractably.

**Problem**: What if there are dependencies between state variables?
Bayesian RL in Structured Domains

In many domains, there are inherent independencies between state variables.

Factored MDPs ([Boutilier et al. 1999])

- \( S : S_1 \times S_2 \times \cdots \times S_n \)
- \( A \)
- \( T(s, a, s') = \prod_{i=1}^{n} \Pr(s'_i|\text{ParVal}_i(s, G_a), a) \)
  - \( G_a \) is a bipartite graph defining conditional dependencies between state variables
- \( R(s, a) \)

E.g. \( G_a = \)

![Diagram](image-url)
Extending Bayesian RL to structured domains is easy when the graph structure is known a priori.

This is not realistic in many domains (and also much less interesting!)

Consider uncertainty in both model structure, $G_a$, and parameters, $\Pr(s_i'|\text{ParVal}_i(s, G_a), a)$. 
Bayesian RL in Structured Domains

**Bayes-Adaptive Structured MDP Model**

- \( S' = S \times G^{|A|} \), where \( G \) is the set of DBNs \((G, \theta_G)\)
- \( A' = A \)

\[
T'(s, G, \theta_G, a, s', G', \theta_G') = \Pr(s'|s, G, \theta_G, a) \Pr(G', \theta_G'|G, \theta_G, a, s').
\]

\[
= \prod_{i=1}^{n} \theta_{G_a}^{i,s'|\text{ParVal}_i(s,G_a)} l_{(G,\theta_G)}(G', \theta_G').
\]

- \( Z' = S \), i.e. transition to a particular state of the MDP

\[
O(s', G', \theta_G', a, z) = l_{s'}(z).
\]

- \( R' : S' \times A' \rightarrow \mathbb{R} \), where \( R'(s, G, \theta_G, a) = R(s, a) \).

**Goal**: Maximize return under partial observability of \((s, G, \theta_G)\).
Bayesian RL in Structured Domains

Bayes-Adaptive Structured MDP Model

- \( S' = S \times G^{|A|} \), where \( G \) is the set of DBNs \((G, \theta_G)\)
- \( A' = A \)

\[
T'(s, G, \theta_G, a, s', G', \theta_{G'}) = \Pr(s'|s, G, \theta_G, a) \Pr(G', \theta'_{G'}|G, \theta_G, s, a, s').
\]
\[
= \prod_{i=1}^n \theta_{G_a}^{i, s'_i | \text{ParVal}_i(s, G_a)} l_{(G, \theta_G)}(G', \theta'_{G'}). 
\]

- \( Z' = S \), i.e. transition to a particular state of the MDP

\[
O(s', G', \theta'_{G'}, a, z) = l_{s'}(z). 
\]

- \( R' : S' \times A' \to \mathbb{R} \), where \( R'(s, G, \theta_G, a) = R(s, a) \).

Goal: Maximize return under partial observability of \((s, G, \theta_G)\).
Define an initial belief:
- $s_0$ : the initial state of the MDP
- $P(G_a, \theta_{G_a}) \forall a \in A$ : the prior over models

Make the following (standard) assumption:

$$P(G_a, \theta_{G_a}) = P(G_a)P(G_a|\theta_{G_a})$$

- $P(G_a|\theta_{G_a})$ is a product of indep. Dirichlet distributions (one per state variable)
- $\phi_{i,s',|ParVal_i(s,G_a)}$ defines the counts, for state variables $i$, of observed transitions $(s, a, s')$

**Problem**: Cannot track $P(G_a)$ is closed form.
The posterior $P(G_a)$ defines a distribution over possible graphs.

**Problem**: There are $O(n!2^{n/2})$ possible graph structures.

**Solution**: Use an MCMC algorithm to sample graph structures from the posterior ([Friedman & Koller, 2003]).
The goal is to optimize a policy:

\[ Pr(s, G, \theta | G_0, \theta_0, s_1, a_1, ..., s_{t-1}, a_{t-1}) \rightarrow a \]

Apply Monte Carlo online planning again:
Network administration domain: (Guestrin et al. 2003)

- A network is composed of $n$ computers linked together by some topology.
- Each computer is either in *running* or *failure* mode.
  - It has some fixed probability of transition from *running* to *failure*.
  - That probability increases for every neighbor in *failure* mode.
  - A computer remains in failure mode until it is rebooted.
- Actions: *RebootComputer* $i$ (for $i = 1 : n$), *DoNothing*
  - Reward of +1 for every running computer at every step.
  - Reward of -1 for each rebooting action.
- Assume full state observability for now.
Experimental Results - Linear Network

Linear network topology:

![Linear Network Topology](image)

Empirical Return:

![Empirical Return Graph](image)
Experimental Results - Linear Network

Error in model inference:

(a) Parameter Error

(b) Structure Error
Error in model inference:

(a) Parameter Error

(b) Structure Error
Experimental Results - Linear Network

Original topology:

Learned topology:
Experimental Results - Larger Network

Original topology:

Learned topology:

Empirical Return:

![Graph showing empirical return over number of steps for different learning methods: Full Joint, Structure Learning, Known Structure. The graph illustrates the performance improvement with learned structures compared to known structures.](image-url)
Motivation Bayesian RL Bayes-Adaptive POMDPs Learning Structure

Experimental Results - Dense Network

Original topology : Learned topology :

Empirical Return :

![Graph showing empirical return over number of steps for different structures](image-url)
Error in model inference:

(a) Parameter Error

(b) Structure Error
Bayesian learning of factored representations allows powerful generalization between states sharing similar features.

Learning is able to make efficient use of data.

Structure learning is able to accelerate learning even in domains with weak structure.
Conclusion

We extended the model-based bayesian RL framework to handle:
- partially observable domains
- continuous state variables
- structured domains

Optimal policy maximizes long-term return given the prior over model.

Monte Carlo methods can be used to achieve tractable (approximate) belief monitoring and planning.

Interesting future applications in human-computer interaction and robotics.

[Pineau et al., 2007]  [Guez et al., 2008]
Acknowledgments

This work was done in collaboration with Stephane Ross.

Financial support provided NSERC, FQRNT, CFI.
QUESTIONS?