Towards Bayesian Reinforcement Learning

Friday, Dec 8, 2006
NIPS workshop:
Towards a New Reinforcement Learning?

Pascal Poupart

University of Waterloo

ppoupart@cs.uwaterloo.ca

Outline

- A bit of history
- Why Bayesian reinforcement learning?
  - Optimal exploration/exploitation tradeoff
  - Explicit encoding of prior knowledge
- Recent advances
  - Bayesian RL ➔POMDP
  - Beetle algorithm
- Conclusion

A bit of history

- Operations Research: Bayesian control of Markov Chains
  - 1960's: Ronald Howard and his students develop Bayesian techniques to control Markov chains with uncertain probabilities and rewards
  - Martin (1969): Bayesian Decision Problems and Markov Chains, Wiley & Sons

A bit of history

- Exploration/exploitation tradeoff (active learning)
  - Several AI researchers consider Bayesian techniques
    - Kaelbling, Meuleau, Dearden, Strens, etc.
  - Duff (2002):
    - Discovers previous work from OR in the 1960s
    - Model-based Bayesian RL ➔ POMDP
  - 2002-present: renewed interest for Bayesian RL
    - **Model-based Bayesian RL:** Wang, Lizotte et al. (2005); Jaulmes, Pineau et al. (2005); Poupart, Vlassis et al. (2006)
    - **Model-free Bayesian RL:** Engel, Mannor, et al. (2003, 2005)
    - **Bayesian policy gradient algorithms:** Ghavamsadeh, Engel (2006)

Why Bayesian RL?

- Pros
  - Optimal exploration/exploitation tradeoff (given prior)
  - Principled approach to active learning
  - Explicit declaration of assumptions
  - Facilitates encoding of domain knowledge
- Cons
  - Mathematically and computationally complex
- Recent advances:
  - Connection between Bayesian RL and POMDPs
  - Improved POMDP algorithms

Exploration/exploitation tradeoff

- Dilemma:
  - Exploitation: maximize immediate rewards
  - Exploration: maximize information gain
- **Wrong question!**

- Single objective: max expected total rewards
  - $V^{\pi}(s_0) = \Sigma_t \gamma^t E[\, R(s_t, \pi(s_t))\,]_{P(st|\pi)}$
  - Optimal policy π^*: $V^{\pi^*}(s) \geq V^{\pi}(s)$ for all s,π
    - **Optimal exploration/exploitation tradeoff**

Reinforcement Learning

- Markov Decision Process:
  - ***S*: set of states**
  - ***A*: set of actions**
  - ***R*: set of rewards**

  } Reinforcement Learning

  - ***T*(*s*,*a*,*s'*) = Pr(*s'*|*s*,*a*): transition function**
  - ***U*(*s*,*a*) = *r*: reward function**

- Bayesian Model-based Reinforcement Learning
- Encode unknown prob. by random variables θ
  - i.e., $\theta_{sas'}$ = Pr(*s'*|*s*,*a*): random variable in [0,1]
  - i.e., θ_{sa} = Pr(•|*s*,*a*): multinomial distribution

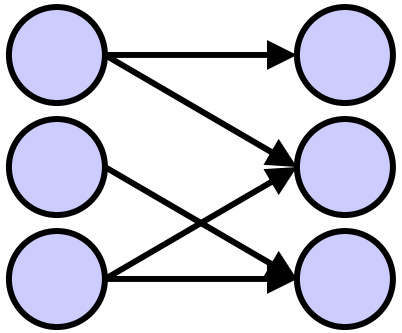
Model Learning

- Assume prior $b(\theta_{sa}) = \Pr(\theta_{sa})$
- Learning: compute posterior given s,a,s'
  - $b_{sas'}(\theta_{sa}) = k \Pr(\theta_{sa}) \Pr(s'|s,a,\theta_{sa}) = k\, b(\theta_{sa})\ \theta_{sas'}$

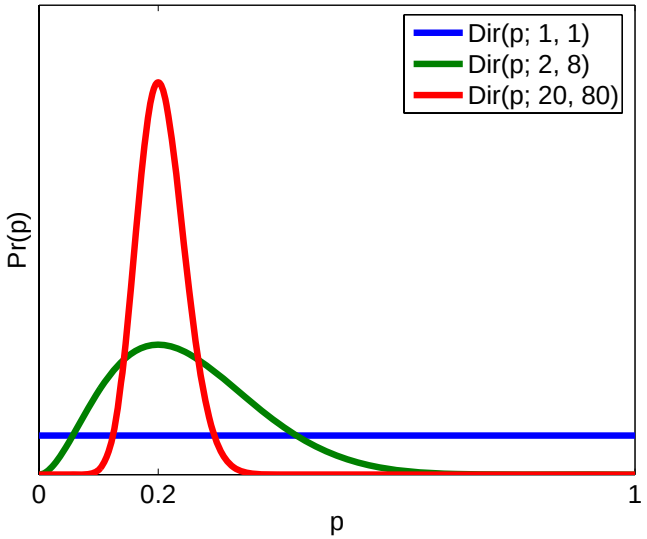
- **Conjugate prior:**
  - Dirichlet prior → Dirichlet posterior
- $b(\theta_{sa}) = \mathrm{Dir}(\theta_{sa};n_{sa}) = k \prod_{s''} (\theta_{sas''})^{n_{sas''} - 1}$
- $b_{sas'}(\theta_{sa}) = k\, \mathrm{b}(\theta_{sa})\ \theta_{sas'}$
  $= k \prod_{s''} (\theta_{sas''})^{n_{sas''} - 1 + \delta(s',s'')}$
  $= k\, \mathrm{Dir}(\theta_{sa}; n_{sa} + \delta(s',s''))$

Prior Knowledge

- Structural priors
  - Tie identical parameters
    - If Pr(•|*s*,*a*) = Pr(•|*s'*,*a'*) then $\theta_{sa} = \theta_{s'a'}$
  - Factored representation
    - DBN: unknown conditional dist.



- Informative priors
  - No knowledge: uniform Dirichlet
  - If $(\theta_1, \theta_2) \sim (0.2, 0.8)$ then set (n_1, n_2) to $(0.2k, 0.8k)$
    - *k* indicates the level of confidence



Misconceptions

- Wouldn't it be better to learn everything from scratch without having to specify any prior?

- **No!**
- There is no such thing as RL without any prior.
- Every learning algorithm has a learning bias
  - Bayesian RL: bias explicit in the prior
  - Other RL techniques: bias implicit but always present
    - Policy search: parameterization of the policy space
    - Value function approximation: type of function approximator

Benefits of Explicit Priors

- Assumptions made can be easily verified
  - Divergence with value function approximators:
    - Implicit (inaccurate) assumption: generalization across states
- Facilitates encoding of domain knowledge
  - Poster: Global Reinforcement Learning (Pavlov & Poupart)
- Prior information simplifies learning
  - Faster training

Policy Optimization

- Classic RL:
  - $V^*(s) = \max_a U(s,a) + \Sigma_{s'} \Pr(s'|s,a) V^*(s')$
  - Hard to tell what needs to be explored
  - Exploration heuristics: ε-greedy, Boltzmann, etc.

- Bayesian RL:
  - $V^*(s,b) = \max_a U(s,a) + \Sigma_{s'} \Pr(s'|s,b,a) V^*(s',b_{sas'})$
  - Belief *b* tells us what parts of the model are not well known and therefore worth exploring

Value Function Parameterization

- **Theorem:** $V^* = \max_i polynomial_i(\theta)$

- **Proof:** by induction
  - Define value function in terms of θ instead of b
    - i.e. $V^*(s,b) = \int_\theta b(\theta)\, V_s(\theta)\, d\theta$
  - Bellman's equation
    - $V_s(\theta) = \max_a \underbrace{U(s,a)}_{k_a} + \Sigma_{s'} \underbrace{\Pr(s'|s,a,\theta)}_{\theta_{sas'}} \underbrace{V_{s'}(\theta)}_{\max_i poly_i(\theta)}$

      $= \max_a \; k_a + \Sigma_{s'} \; \theta_{sas'} \; \max_i poly_i(\theta)$

      $= \max_j \; poly_j(\theta)$

Partially Observable domains

- Beliefs: mixtures of Dirichlets
- Theorem also holds for partially observable domains:
  - Value function: $\max_i$ polynomials$_i(\theta)$

BEETLE Algorithm

- Sample a set of reachable belief points B
- $V \leftarrow \{0\}$
- Repeat
  - $V' \leftarrow \{\}$
  - For each b in B compute multivariate polynomial
    - $poly_{as'}(\theta) \leftarrow \text{argmax}_{poly \in V} \int_\theta b_{sas'}(\theta)\, poly(\theta)\, d\theta$
    - $a^* \leftarrow \text{argmax}_a \int_\theta b_{sas'}(\theta)\, R(s,a) + \Sigma_{s'}\, \theta_{sas'}\, poly_{as'}(\theta)\, d\theta$
    - $poly(\theta) \leftarrow U(s,a^*) + \Sigma_{s'}\, \theta_{sa^*s'}\, poly_{a^*s'}(\theta)$
    - $V' \leftarrow V' \cup \{poly\}$
  - $V \leftarrow V'$

Beetle properties

- Offline: optimize policy at sampled belief points
  - Time: minutes to hours
  - Two approximations:
    - Sampled belief points
    - Polynomials: projection on fixed # of monomials
- Online: learn transition model by belief monitoring
  - Time: fraction of a second

- Fast enough for online learning
- Approx. ➔ suboptimal explor/exploit tradeoff

Empirical Evaluation

Problem	|S|	|A|	Free params	Opt	Discrete POMDP	Exploit	**Beetle**	**Beetle time (minutes)**
Chain1	5	2	1	3677	3661 ± 27	3642 ± 43	**3650 ± 41**	**1.9**
Chain2	5	2	2	3677	3651 ± 32	3257 ± 124	**3648 ± 41**	**2.6**
Chain3	5	2	40	3677	na-m	3078 ± 49	**1754 ± 42**	**32.8**
Handw1	9	2	4	1153	1149 ± 12	1133 ± 12	**1146 ± 12**	**14.0**
Handw2	9	2	8	1153	990 ± 8	991 ± 31	**1082 ± 17**	**55.7**
Handw3	9	6	270	1083	na-m	297 ± 10	**385 ± 10**	**133.6**

Informative Priors

Problem	Opt	Informative priors			
		k = 0	k = 10	k = 20	k = 30
Chain3	3677	1754 ± 42	3453 ± 47	2034 ± 57	3656 ± 32
Handw2	1153	1082 ± 17	1056 ± 18	1097 ± 17	1106 ± 16
Handw3	1083	385 ± 10	540 ± 10	1056 ± 12	1056 ± 12

Conclusion

- Bayesian RL
  - Oldest form of RL
  - Principled solution to exploration/exploitation tradeoff
  - Facilitates encoding of domain knowledge
  - Reduces exploration
- Contributions
  - Optimal value function parameterization:
    - upper envelope of multivariate polynomials
  - BEETLE algorithm

Future work

- Test on robotics problems
  - With Nikos Vlassis, Martin Reidmiller, Russ Tedrake
- Extend Beetle to
  - Partially observable domains
  - Continuous state spaces
- Encoding of domain knowledge
  - Poster: Global Reinforcement Learning (Pavlov and Poupart)