An Analytic Solution to Discrete Bayesian Reinforcement Learning

Pascal Poupart (U of Waterloo)

Nikos Vlassis (U of Amsterdam)

Jesse Hoey (U of Toronto)

Kevin Regan (U of Waterloo)

Motivation

- Automated assistant
  - [Boger et al. IJCAI-05]
- Use RL to adapt to users
  - Learn through user interactions (no simulation)
  - Bear cost of actions
  - Cannot explore too much
  - Real-time response

Model-Based Bayesian RL

- Model-based Bayesian RL:
  - Naturally optimize exploration/exploitation tradeoff
  - Reduce exploration with prior knowledge
  - Mathematically and computationally complex

- **Contributions:**
  - **Optimal value function has simple parameterization**
    - i.e., upper envelope of a set of multivariate polynomials
  - **BEETLE: Bayesian Exploration/Exploitation Tradeoff in LEarning**
    - Exploit polynomial parameterization

Outline

- Bayesian reinforcement learning
- Value function parameterization
- BEETLE algorithm
- Experiments
- Conclusion

Reinforcement Learning

- Markov Decision Process:
  - ***S*: ** set of states
  - ***A*: ** set of actions
  - ***R*: ** set of rewards
  - ***T*(*s*,*a*,*s'*) = Pr(*s'*|*s*,*a*): ** transition function
  - ***U*(*s*,*a*) = *r*: ** reward function

} Reinforcement Learning

- 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

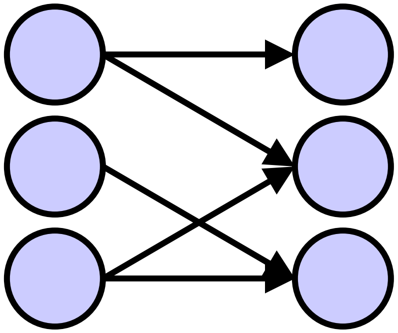
Model Learning

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

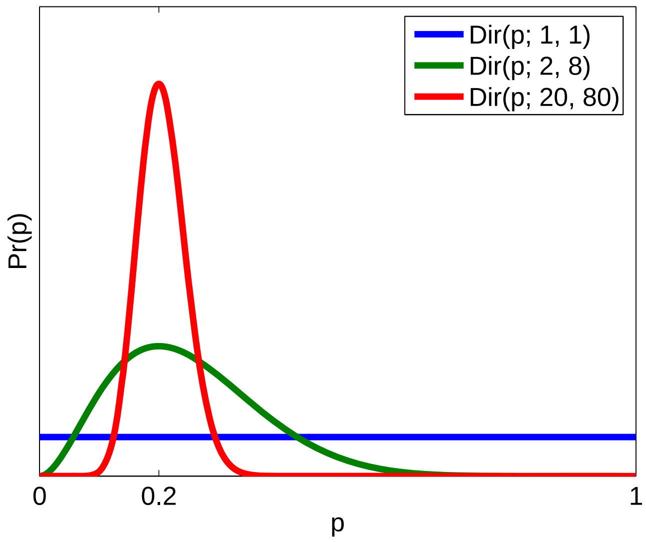
- **Conjugate prior:**
  – Dirichlet prior → Dirichlet posterior
- $b(\theta_{sa}) = \mathrm{Dir}(\theta_{sa};n_{sa}) = k \prod_{s''} (\theta_{sas''})^{n_{sas''} - 1}$
- $b_{sas'}(\theta_{sa}) = k\, \mathrm{b}(\theta_{sa})\ \theta_{sas'}$
  $= k \prod_{s''} (\theta_{sas''})^{n_{sas''} - 1 + \delta(s',s'')}$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

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



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
  - Optimal exploration/exploitation tradeoff
  - [Dearden 98,99], [Strens 00], [Duff 02], [Wang 05]

Value Function Parameterization

- **Theorem:** V^* is the upper envelope of a set of multivariate polynomials ($V_s(\theta) = \max_i poly_i(\theta)$)
- **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
    - $V_s(\theta) = \max_a \underbrace{U(s,a)}_{k_a} + \Sigma_{s'} \underbrace{\Pr(s'|s,a,\theta)}_{\theta_{sas'}}\ \underbrace{V_{s'}(\theta)}_{\max_i poly_i(\theta)}$

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

      $= \max_j\ poly_j(\theta)$

BEETLE Algorithm

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

Polynomials

- Computational issue:
  - # of monomials in each polynomial grows by $O(|S|)$ at each iteration
  - $poly(\theta) = U(s,a^*) + \Sigma_{s'}\, \theta_{sa^*s'}\, poly_{a^*s'}(\theta)$
    $= U(s,a^*) + \Sigma_{s'}\, \theta_{sas'}\, \Sigma_i\, mono_i(\theta)$
    $= U(s,a^*) + \Sigma_{i,s'}\, mono_{i,s'}(\theta)$
- After *n* iterations: polynomials have $O(|S|^n)$ monomials!

Projection Scheme

- Approximate polynomials by a linear combination of a fixed set of monomial basis functions $\phi_i(\theta)$:
  - i.e. $poly(\theta) \approx \Sigma_i \, c_i \, \phi_i(\theta)$

- Find best coefficients c_i by minimizing L_n norm:
  - $Min_c \int_\theta |poly(\theta) - \Sigma_i \, c_i \, \phi_i(\theta)|^n \, d\theta$
- For the Euclidean norm (L_2), this can be done by solving a system of linear equations $Ax = b$ such that
  - $A_{ij} = \int_\theta \phi_i(\theta) \, \phi_j(\theta) \, d\theta$
  - $b_i = \int_\theta poly(\theta) \, \phi_j(\theta) \, \mathrm{d}\theta$
  - $x_i = c_i$

Basis functions

- Which monomials should we use as basis functions?

- Recall that:
  - $b_{sas'}(\theta) = k\, b(\theta)\, \theta_{sas'}$
  - $poly(\theta) \leftarrow U(s,a) + \Sigma_{s'}\, \theta_{sas'}\, poly_{as'}(\theta)$

- Hence we use beliefs as basis functions

Beetle summary

- Offline: optimize policy at sampled belief points
  - Time: minutes to hours
- Online: learn transition model by belief monitoring
  - Time: fraction of a second

- Advantages:
  - Fast enough for online learning
  - Optimizes exploration/exploitation tradeoff
  - Easy to encode prior knowledge in initial belief
- Disadvantage:
  - Policy may not be good for all belief points

Empirical Evaluation

- Comparison with two heuristics

- **Exploit:** pure exploitation strategy
  - Greedily select best action of the mean model at each time step
  - Slow execution: must solve an MDP at each time step

- **Discrete POMDP:** discretize θ
  - Discretization leads to an exponential number of states
  - Intractable for medium to large problems

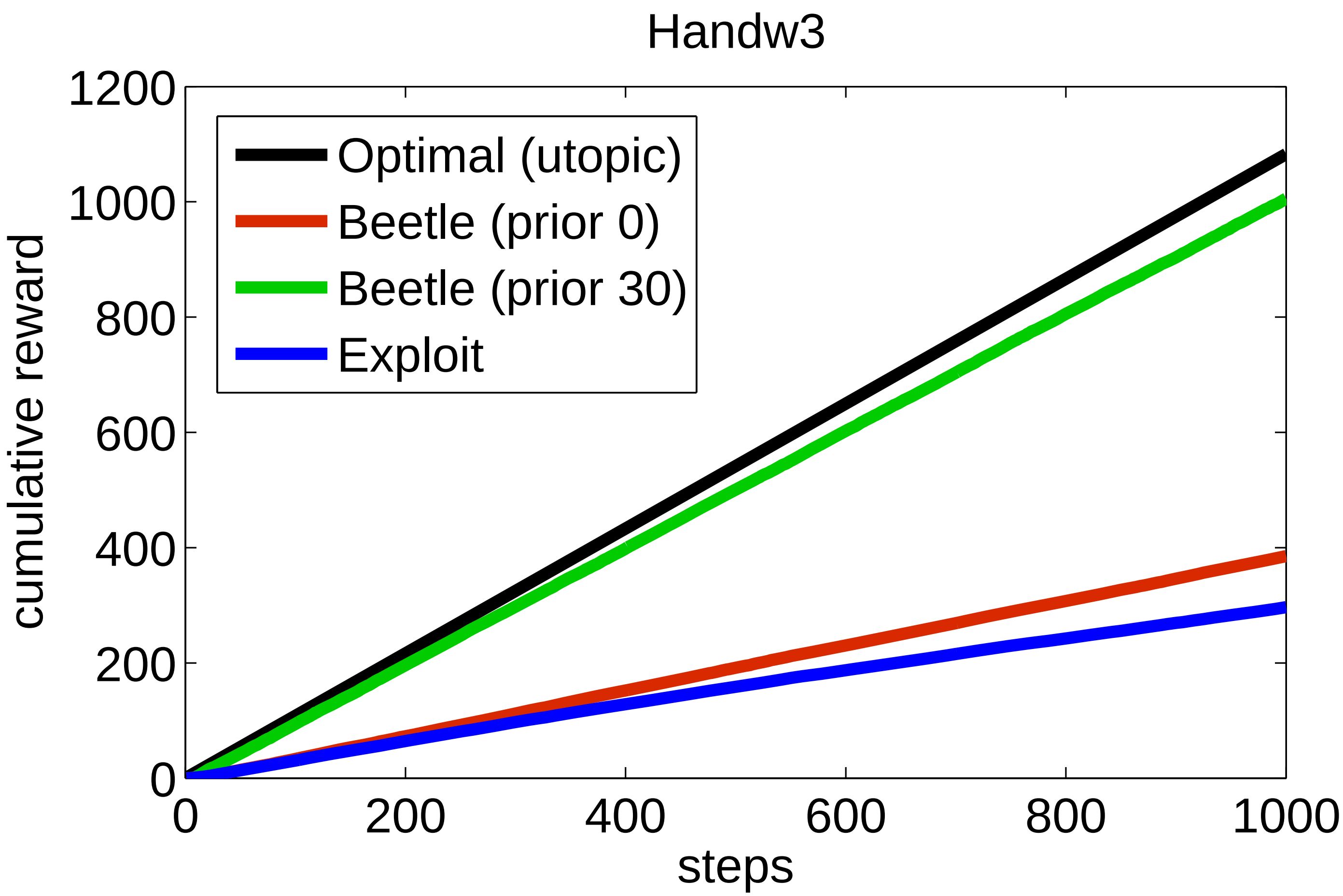
Empirical Evaluation

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

Learning Curves



Conclusion

- Motivation
  - Learning by interaction with environment (no simulation)
  - Bear consequence of actions
  - Minimal exploration
  - Real-time execution
- Bayesian RL
  - Optimizes exploration/exploitation tradeoff
  - Can easily encode prior knowledge to reduce exploration
- Contributions
  - Optimal value function parameterization: as the upper envelope of multivariate polynomials
  - BEETLE algorithm

Future work

- Learn user behaviors for assistive technologies
- Consider partially observable domains
- Learn dynamic transition models
- Consider correlated Dirichlets priors