An Analytic Solution to Discrete Bayesian Reinforcement Learning

Pascal Poupart, Nikos Vlassis, Jesse Hoey, Kevin Regan

Appearing in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006.

Presented by Xinyun Zou

February 12, 2018

(This presentation is fully based on the authors)

Background

- Reinforcement learning (RL) was proposed to allow agents to learn *online* as they interact with their environment
- Shortage of many existing online RL models:
  - slow convergence -> not realistic when each state transition has a cost / some state transitions may lead to severe losses (e.g., helicopter crash, mobile robot collision)
    - Model-free RL: directly learn an optimal policy (or value function)
  - much more complicated and computationally intensive, despite of mitigating the model-free shortage
    - Model-based RL: incorporate prior knowledge to mitigate severe losses
- As a result, RL was mostly used for *offline* learning in simulated environments.

Overview of the Paper

- POMDP formulation of Bayesian RL & Belief Monitoring
- Offline approximate policy optimization technique
- The BEETLE: effective online learning algorithm
- Experiments on the BEETLE

POMDP formulation of Bayesian RL

- Markov decision process (MDP): <S, A, T, R>
  - S: set of discrete states, A: set of actions, T: $\Pr(s' \mid s, a)$, R: $R(s, a, s')$, $\pi: S \rightarrow A$
- The RL problems consist of finding an optimal policy for an MDP with a partially or completely unknown transition function.
  - Derive a simple parameterization of the optimal value function for the Bayesian model-based approach
  - Bayesian learning
    - Pick a prior distribution encoding the learner's initial belief about the possible values of each unknown parameter.
    - Whenever a sampled realization of the unknown parameter is observed, update the belief to reflect the observed data.
    - Each unknown transition parameter T(s', a, s) becomes an unknown continuous parameter $\theta_a^{s,s'}$ in the [0,1] interval.

POMDP formulation of Bayesian RL

- Bayesian model-based RL as a partially observable Markov decision process (POMDP): $<S_P, A_p, O_p, T_p, Z_p, R_p>$
  - $S_P = S \times \{\theta_a^{s,s'}\}$, $A_p = A$: action space, $O_p = S$: observable MDP state space
  - Transition function: $T_p(s, \theta, a, s', \theta') = \Pr(s', \theta' | s, \theta, a)$, factored in
    - Distribution for MDP states: $\Pr(s' \mid s, \theta_a^{s,s'}, a)$
    - Distribution for the unknown parameters: $\Pr(\theta' | \theta) = \delta_\theta(\theta')$
      - $\delta_\theta(\theta')$: Kronecker delta -> unknown parameters are stationary (i.e., θ does not change)
  - Observation function: $Z_p(s', \theta', a, o) = \Pr(o | s', \theta', a) = \delta_{s'}(o)$
  - Reward function: $R_p(s, \theta, a, s', \theta') = R(s, a, s')$, not depend on θ nor θ'

Belief Monitoring to Learn θ

- Belief (i.e. probability density) monitoring is as simple as incrementing the hyperparameter corresponding to the observed transition.
- At each time step, the belief $b(\theta) = Pr(\theta)$ over all unknown parameters $\theta_a^{s,s'}$ is updated based on the observed transitions s, a, s' using Bayes' theorem:

$$\begin{aligned} b_a^{s,s'}(\theta) &= kb(\theta)\Pr(s'|\theta, s, a) & (1) \\ &= kb(\theta)\theta_a^{s,s'} & (2) \end{aligned}$$

- Since the unknown transition model θ is made up of one unknown distribution θ_a^s per s, a pair, let the prior be $b(\theta) = \prod_{s,a} D(\theta_a^s; n_a^s)$ such that n_a^s is a vector of hyperparameters $n_a^{s,s'}$. The posterior obtained after transition $\hat{s}, \hat{a}, \hat{s}'$ is:

$$\begin{aligned} b_a^{s,s'}(\theta) &= k\theta_a^{s,s'}\Pi_{s,a}\mathcal{D}(\theta_a^s; n_a^s) & (3) \\ &= \Pi_{s,a}\mathcal{D}(\theta_a^s; n_a^s + \delta_{\hat{s},\hat{a},\hat{s}'}(s, a, s')) & (4) \end{aligned}$$

  - Dirichlets are conjugate priors of multinomials. A Dirichlet distribution over a multinomial *p* is parameterized by positive numbers n_i, such that $n_i - 1$ can be interpreted as the number of times that the p_i -probability event has been observed.

$$\mathcal{D}(p; n) = k\Pi_i p_i^{n_i - 1}$$

Overview of the Paper

- POMDP formulation of Bayesian RL & Belief Monitoring
- Offline approximate policy optimization technique
- The BEETLE: effective online learning algorithm
- Experiments on the BEETLE

Bellman's equation for Bayesian RL

- In POMDPs, $\pi(b) = a$. V^{π} is the policy value measured by the discounted sum of the rewards earned while executing it: $V^{\pi}(b) = \sum_{t=0}^{\infty} \gamma^t R(b_t, \pi(b_t), b_{t+1})$.
- The optimal value function satisfies Bellman's equation, piecewise linear and convex:
$$V^*(b) = \max_a \sum_o \Pr(o|b,a)[R(b,a,b_a^o) + \gamma V^*(b_a^o)]. \quad (5)$$
- Following Duff (2002): $$V_s^*(b) = \max_a \sum_o \Pr(o|s,b,a)[R(s,b,a,s',b_a^o) + \gamma V_{s'}^*(b_a^o)]. \quad (6)$$
- Since rewards do not depend on b nor b_a^o and observations correspond to the physical states s' in Bayesian RL, the Bellman's equation can be simplified to:
$$V_s^*(b) = \max_a \sum_{s'} \Pr(s'|s,b,a)[R(s,a,s') + \gamma V_{s'}^*(b_a^{s,s'})]. \quad (7)$$

where b is current belief in θ and $b_a^{s,s'}$ is the revised belief state according to (4)

Exploration/Exploitation Tradeoff

- Recall the simplified Bellman's equation:

$$V_s^*(b) = \max_a \sum_{s'} \Pr(s'|s,b,a)[R(s,a,s') + \gamma V_{s'}^*(b_a^{s,s'})]. \quad (7)$$

- Policy optimization by pure exploitation by replacing $b_a^{s,s'}$ in (7) by b, since pure exploitation selects the action that maximizes total rewards based on b only, disregarding the fact that valuable information may be gained by observing the outcome of the action chosen.

$$V_s^*(b) = \max_a \sum_{s'} \Pr(s'|s,b,a)[R(s,a,s') + \gamma V_{s'}^*(b)] \quad (8)$$

- *Conditional planning* hypothesizes future action outcomes and takes into account
- In (7), all possible updated belief states $b_a^{s,s'}$ are considered with probabilities corresponding to the likelihood of reaching s'.
- (7) optimizes the sum of rewards that can be derived based on
  - exploitation, i.e. information available in b
  - and exploration, i.e. info gained in the future by observing selected actions' outcome
- An optimal policy of the POMDP formulation of Bayesian RL optimizes the exploration / exploitation tradeoff, since such a policy maximizes the expected total return.

Optimal Value Function Parameterization

- In Bayesian RL, the optimal value function corresponds to the upper envelope of a set Γ of linear segments called α-functions due to the continuous nature of θ (i.e. $V_s^*(b) = \max_{\alpha \in \Gamma} \alpha_s(b)$).
  - α can be defined as a linear function of b subscripted by s (i.e., $\alpha_s(b)$)
  - or as a function of θ subscripted by s (i.e., $\alpha_s(\theta)$)
- Several existing algorithms (based on e.g. confidence intervals, Normal-Gamma distributions, linear combinations of hyperparameters and sampling) are computationally intensive or make drastic approximations.
- Theorem 1: α-functions in Bayesian RL are multivariate polynomials.
  - Assume α-functions in Γ^k are multivariate polynomials, then so does $a_{b,s}$ in (15).
  - Multivariate polynomials form a closed representation for α-functions under Bellman backups.
- Review of the Bellman backup operator and the updating of the α-functions
  - Suppose the optimal value function $V_s^k(b) = \max_{\alpha \in \Gamma^k} \alpha_s(b)$ for k steps-to-go
  - Use Bellman's equation to compute by dynamic programming the best set Γ^{k+1} representing the optimal value function V_s^{k+1} with k+1 stages-to-go.

Optimal Value Function Parameterization

- Rewrite Bellman's equation (7) by substituting V^k for the maximum over the α-functions in Γ^k: $V_s^{k+1}(b) = \max_a \sum_{s'} \Pr(s'|s,b,a)[R(s,a,s') + \gamma \max_{\alpha \in \Gamma^k} \alpha_{s'}(b_a^{s,s'})]$
- Decompose Bellman's equation in 3 steps:
  - Find maximal α-function for each a and s': $$\alpha_{b,a}^{s,s'} = \operatorname{argmax}_{\alpha \in \Gamma^k} \alpha_{s'}(b_a^{s,s'}) \quad (9)$$
  - Find the best action a: $$a_b^s = \operatorname{argmax}_a \sum_{s'} \Pr(s'|s,b,a)[R(s,a,s') + \gamma \alpha_{b,a}^{s,s'}(b_a^{s,s'})] \quad (10)$$
  - Perform the actual Bellman backup: $$V_s^{k+1}(b) = \sum_{s'} \Pr(s'|s,b,a_b^s)[R(s,a_b^s,s') + \gamma \alpha_{b,a_b^s}^{s,s'}(b_{a_b^s}^{s,s'})] \quad (11)$$
- Rewrite 3rd step (11) by using α-functions w.r.t. θ and expanding belief state $b_{a_b^s}^{s,s'}$:
$$\sum_{s'} \Pr(s'|s,b,a_b^s)[R(s,a_b^s,s') + \gamma \int_\theta b_{a_b^s}^{s,s'}(\theta) \alpha_{b,a_b^s}^{s,s'}(\theta) d\theta] \quad (12)$$
$$= \sum_{s'} \int_\theta b(\theta) \Pr(s'|s,\theta,a_b^s)[R(s,a_b^s,s') + \gamma \alpha_{b,a_b^s}^{s,s'}(\theta) d\theta] \quad (13)$$
$$= \int_\theta b(\theta) \Big[\sum_{s'} \Pr(s'|s,\theta,a_b^s)[R(s,a_b^s,s') + \gamma \alpha_{b,a_b^s}^{s,s'}(\theta)] \Big] d\theta \quad (14)$$
- For every b, define such an α-function and together they form a set Γ^{k+1}: $$\alpha_{b,s}(\theta) = \sum_{s'} \Pr(s'|s,\theta,a_b^s)[R(s,a_b^s,s') + \gamma \alpha_{b,a_b^s}^{s,s'}(\theta)]. \quad (15)$$
- Since each $a_{b,s}$ was defined by using the optimal action and α-functions in Γ^k, then each $a_{b,s}$ is necessarily optimal at b: $$V_s^{k+1}(b) = \int_\theta b(\theta) \alpha_{b,s}(\theta) d\theta \quad (16)$$ $$= \alpha_{b,s}(b) \quad (17)$$ $$= \max_{\alpha \in \Gamma^{k+1}} \alpha_s(b) \quad (18)$$

Overview of the Paper

- POMDP formulation of Bayesian RL & Belief Monitoring
- Offline approximate policy optimization technique
- The BEETLE: effective online learning algorithm
- Experiments on the BEETLE

Point-based Value Iteration

- Multivariate polynomials form a closed representation for α-functions under Bellman backups
- BEETLE (Bayesian Exploration Exploitation Tradeoff in LEarning)
  - A simple and efficient point-based value iteration algorithm
  - An extension of the Perseus algorithm (Spann & Vlassis, 2005) for Bayesian RL

1. A set of reachable s, b pairs sampled by simulating several runs of a default or random policy
2. For a given s, b pair, the best α-function for each a and s' is computed by (9)
3. The optimal action is computed according to (10)
4. A new α-function is constructed by (15) and represented by the non-negative powers λ of its monomial terms

- BEETLE suffers from intractability
  - At each backup, the number of terms of the multivariate polynomial of the α-function grows significantly

α-function Projection

- To mitigate the exponential growth in the number of monomials,
- Project each new α-function onto a multivariate polynomial with a smaller number of monomials after each Bellman backup ➔ minimize the error at each θ
- Pick basis functions as close as possible to the monomials of α-functions
  - In both equations for belief monitoring (4) and backing up α-functions (11), powers are incremented with each s, a, s' transition ➔ they are made up of similar monomials
  - Use the set of reachable belief states generated at the beginning of the BEETLE algorithm as the fixed basis set
- A fixed basis set allows precomputation of the projection of each backed-up component.
  - α-functions can be presented by a column vector (i.e., coefficients of the fixed basis functions)
  - A projected transition function in matrix form for each s, a, s' can be pre-computed
  - The projection of the reward function can be pre-computed and basis coefficients can be stored
  - Point-based backups can be performed by simple matrix operations. Then (15) becomes

$$\tilde{\alpha}_{b,s} = \sum_{s'} \tilde{T}_a^{s,s'} [\tilde{R}_a^{s,s'} + \gamma \tilde{\alpha}_{b,a}^{s,s'}]. \qquad (23)$$

Parameter Tying

- The transition probabilities are partially unknown and may be encoded with few parameters by tying parameters together or using a factored model
- The amount of interaction for online learning can be reduced by starting with informative priors, i.e., prior distributions skewed to a small range of values
- Beetle can be used directly
  - when unknown parameters are tied → have one θ_i per different unknown distribution
  - for factored transition dynamics → have one θ_i per unknown conditional distribution
- For each observed transition s, a, s', we increment several powers, one per conditional probability table, during belief monitoring and point-based backups
- α-functions remain multivariate polynomials

Reward Function

- The BEETLE can learn reward functions with finite possible values
  - With a factored model, the reward signal r can be treated as a state variable
  - The reward function $R(s, a, s') = r$ can be encoded as a conditional probability distribution $\Pr(r|s, a, s')$ and learnt accordingly
  - Discretization for a continuous reward signal should provide enough accuracy.

Discussion

- For effective online learning, the computation time while executing the policy really matters (instead of the offline optimization time)
  - At run time, actions should be selected in less than a second for realtime execution
  - BEETLE achieves this easily since belief monitoring and action selection are not computationally intensive.
- The policy computed offline consists of a mapping from state-belief pairs to actions, fixed throughout its execution
- The belief states change with each state transition.
  - Recall that belief monitoring is the process by which the unknown transition dynamics are learned.
  - The policy indirectly adapts with each belief update.
- Drawback of offline policy optimization: the precomputed policy should prescribe an optimal action for every belief state, but this is usually intractable.
  - point-based value iteration concentrates its effort on finding good actions at a sample of reachable belief states

Overview of the Paper

- POMDP formulation of Bayesian RL & Belief Monitoring
- Offline approximate policy optimization technique
- The BEETLE: effective online learning algorithm
- Experiments on the BEETLE

Two Heuristic Methods

- EXPLOIT
  - Online method with no offline optimization
  - Purely exploits its current belief at each time step
  - Monitor the belief state online and pick the best action by solving the MDP for the expected model
  - Pro: simple
  - Cons: slow at run time, lack of exploration

- DISCRETE POMDP
  - Discretize the unknown distributions θ in N values and build a discrete POMDP
  - Cons: the exponential explosion of the state space (which consists of the cross product of the physical states with N discrete values for each unknown distributions)

Problem Descriptions

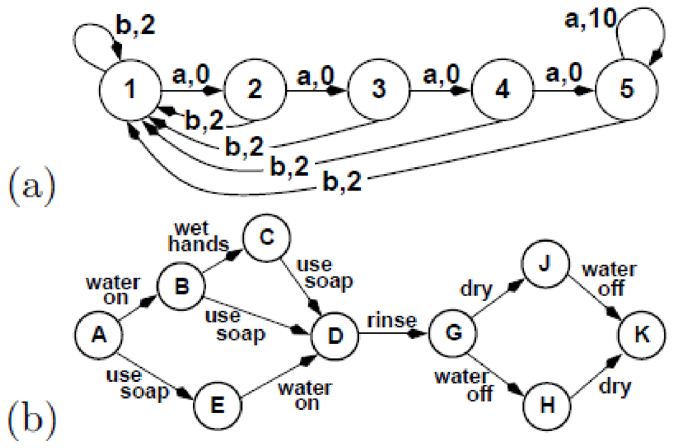

Figure 1. (a) Chain problem showing the action, reward for each transition (b) plansteps for the handwashing problem.

- **The "chain" problem** (Strens 2000; Dearden et al., 1998)
  - The agent has 2 actions a, b that cause transitions between 5 states
  - At each time step, the agent "slips" and
  - performs the opposite action with probability $p_{slip} = 0.2$
- **The handwashing problem** (simplified from Boger et al., 2005)
  - A system gives audio prompts to help persons with cognitive disabilities wash their hands with minimal assistance from a caregiver
  - Major issue: to learn user characteristics that affect their ability to carry out the task
  - Only online learning can be done for those characteristics when interacting with users
  - States can be grouped into 9 plansteps.
  - 2 actions: to do nothing or to issue an audio prompt corresponding to the current planstep
  - Each user has some distribution over behaviors: (1) doing nothing, (2) best possible action at a planstep, (3) second best action (if there are two choices), (4) regressing (e.g. putting soap on their hands after they are clean at planstep=G)

Results

Table 1. Expected total reward for chain and handwashing problems. na-m indicates insufficient memory.

problem	$\|S\|$	$\|A\|$	free params	optimal (utopic)	discrete POMDP	exploit	Beetle	Beetle time (minutes) precomputation	optimization
chain_tied	5	2	1	3677	3661 ± 27	3642 ± 43	3650 ± 41	0.4	1.5
chain_semi	5	2	2	3677	3651 ± 32	3257 ± 124	3648 ± 41	1.3	1.3
chain_full	5	2	40	3677	na-m	3078 ± 49	1754 ± 42	14.8	18.0
handw_tied	9	2	4	1153	1149 ± 12	1133 ± 12	1146 ± 12	2.6	11.8
handw_semi	9	2	8	1153	990 ± 8	991 ± 31	1082 ± 17	3.4	52.3
handw_full	9	6	270	1083	na-m	297 ± 10	385 ± 10	125.3	8.3

Table 2. Expected total reward for varying priors

prior	0	10	20	30
chain_f	1754 ± 42	3453 ± 47	2034 ± 57	3656 ± 32
hand_s	1082 ± 17	1056 ± 18	1097 ± 17	1106 ± 16
hand_f	385 ± 10	540 ± 10	1056 ± 12	1056 ± 12

- Experimented with 3 structural priors
  - Tied: state and action independent
  - Semi-tied: action dependent
  - Full: extreme (rare) case when dynamics are completed unknown
- Optimal return given the true model is reported as a upper bound
  - BEETLE: near optimal policies for the tied and semi-tied, but poor on the full
  - Discrete POMDP: good policies for the tied and semi-tied, but out of memory for the full
  - Exploit: provably optimal policies for the tied (as no exploration required), but sub-optimal for the semi-tied and full
- Table 2 shows test of BEETLE with informative priors (instead of uniform priors)
  - As k increases from 0 to 30, the confidence in the true model increases
  - Increasingly informative priors generally improve BEETLE's performance since it can focus on finding a good policy for a smaller range of likely models

Conclusions

- Optimal value functions for Bayesian RL are parameterized by sets of multivariate polynomials
- This parameterization is exploited to develop an effective algorithm Beetle
- The BEETLE optimizes the exploration/exploitation tradeoff. It focuses only on the truly unknown parts of the dynamics by allowing practitioners to easily encode prior knowledge, reducing the amount of exploration necessary
- Online efficiency is achieved by
  - precomputing offline a policy and
  - doing only action selection and belief monitoring at run time.
- Future directions
  - to extend this work on Bayesian RL in several directions, including continuous state, action and observation spaces, partially observable domains and multi-agent systems.
  - to explore how to handle and possibly learn non-stationary dynamics.

THANK YOU