Safe and Sample-Efficient Reinforcement Learning Algorithms for Factored Environments

Thiago D. Simão
Delft University of Technology, The Netherlands
t.diassimao@tudelft.nl

Abstract

Reinforcement Learning (RL) deals with problems that can be modeled as a Markov Decision Process (MDP) where the transition function is unknown. In situations where an arbitrary policy π is already in execution and the experiences with the environment were recorded in a batch $\mathcal{D}$, an RL algorithm can use $\mathcal{D}$ to compute a new policy π'. However, the policy computed by traditional RL algorithms might have worse performance compared to π. Our goal is to develop safe RL algorithms, where the agent has a high confidence that the performance of π' is better than the performance of π given $\mathcal{D}$. To develop sample-efficient and safe RL algorithms we combine ideas from exploration strategies in RL with a safe policy improvement method.

1 Model-based Exploration in RL

To find the optimal policy quickly, the R-max algorithm [Brafman and Tennenholtz, 2002] incentivizes the agent to explore unknown parts of the environment in early stages of the learning process. To do so, it keeps track of a set of state-action pairs considered known:

$$K_m = \{(s,a) \in S \times A \mid n(s,a) \geq m\}, \qquad (1)$$

where $n(s,a)$ is the number of times the agent has applied action a in state s, and m is a threshold to consider a state-action pair known.

Often, the state space S can be represented by a set of state factors $X = \{X_1, \cdots, X_{|X|}\}$ where each factor has a domain $Dom(X_i)$. When these factors are highly independent, a Factored MDP (FMDP) can compactly represent an MDP, using a dependence function $D : S \times A \times X \to \mathcal{I}$ that indicates the commonalities among different factors, where $\mathcal{I}$ is a set of dependency identifiers [Strehl, 2007]. The probabilistic transition function can be compactly represented:

$$T(s' \mid s,a) = \prod_{i=1}^{|X|} P(s'_i \mid D(s,a,X_i)),$$

where s'_i is the value of X_i in the next state s'.

The factored R-max algorithm is a direct extension of R-max for FMDPs [Guestrin *et al.*, 2003]. It maintains an estimate of each transition component distribution and decides which state-action pairs are known or not according to these estimates. This algorithm only considers as known parts of the environment where the estimate of all transition components have been experienced enough times. In particular, given a minimum number of samples for each factor $\vec{m} = \langle m_1, \ldots, m_{|X|}\rangle$ and the counters of each transition component $n(j)$, the set of known state-action pairs is constructed as follows:

$$K_{\vec{m}} = \{(s,a) \in S \times A \mid \forall X_i : n(D(s,a,X_i)) \geq m_i\}. \qquad (2)$$

2 Safe Policy Improvement

Safe Policy Improvement (SPI) addresses the question of how to compute a new policy π that outperforms the behavior policy π_b with high confidence $1-\delta$, given a batch of previous interactions $\mathcal{D}$ and an admissible error ζ.

The SPI by Baseline Bootstrapping (SPIBB) framework is a model-based approach that guarantees safety by bootstrapping unknown parts of the approximated model with the behavior policy π_b [Laroche *et al.*, 2019]. Formally, the set of bootstrapped state-action pairs $\mathfrak{B}_m$ is the complement of the set K_m (1). This way, the SPIBB algorithms guarantee to perform at least as well as the behavior policy and do not rely on a safety test, in contrast to other SPI algorithms.

The *policy-based* Π_b-SPIBB algorithm attributes the same probability to bootstrapped pairs as the behavior policy, which restricts the policy space to

$$\Pi_b = \{\pi \mid \pi(s,a) = \pi_b(s,a) : \forall \pi \in \Pi, \forall (s,a) \in \mathfrak{B}_m\}. \qquad (3)$$

Laroche *et al.* [2019] prove that if $m = \frac{2}{\epsilon^2} \log \frac{|S||A|2^{|S|}}{\delta}$ then the Π_b-SPIBB algorithm is safe, where ϵ is a bound on the L_1 distance between the estimated transition function and the true transition function, that depends on the precision parameter ζ.

The Π_b-SPIBB algorithm can change the policy if a subset of the state-action pairs is well known, therefore it ca be less conservative than other SPI algorithms. Nevertheless, when the problem is described by a set of factors, m grows exponentially in the number of factors. In the next section we show that, by taking in account the independence between features, it is possible to exploit the factored representation of the problem using a minimum number of samples that is only polynomial in the number of parameters of the FMDP.

3 Main Contributions

To exploit the structure of an FMDP we proposed the Factored Π_b-SPIBB algorithm, an adaptation of Π_b-SPIBB algorithm for factored environments [Simão and Spaan, 2019]. This algorithm takes an extra input: the dependency function D that determines which transition components must be estimated.

First, the algorithm estimates each transition component according to $\mathcal{D}$ using the same counters as the factored R-max algorithm. Second, it redefines the set of state-action pairs to be bootstrapped $\mathfrak{B}_{\vec{m}}$ as the complement of the set of known state-action pairs $K_{\vec{m}}$ (2). In this way, given $\mathfrak{B}_{\vec{m}}$ and the behavior policy π_b, the constrained policy space Π_b can be computed using (3). Finally, the algorithm searches for an optimal policy in Π_b, however, in this case the transition function $\hat{T}(\cdot \mid s, a)$ is estimated according to the estimate of each transition component.

The theoretical analysis shows that it is possible to bound the probability that the Factored Π_b-SPIBB algorithm computes a policy worse than the behavior policy. To do so it is necessary to define $\vec{m}$ such that

$$m_i = \frac{2|X|^2}{\epsilon^2} \log \frac{|\mathcal{Q}|2^{|Dom(X_i)|}}{\delta}, \forall i \in [1, |X|].$$

Therefore, given a desired ϵ, the term $|A||S|$ is replaced by $|\mathcal{Q}|$ and $|S|$ is reduced to $|Dom(X_i)|$. This comes at the lower cost of adding a term polynomial in the number of variables $|X|^2$ (necessary to bound the error of each component distribution). In domains where the features are highly independent from each other these results reduce significantly the number of samples necessary to improve the behavior policy.

Next, we relax the assumption that the dependency function D is known a priori, by proposing the Structure Learning Π_b-SPIBB that integrates different structure learning algorithms [Diuk *et al.*, 2009; Strehl *et al.*, 2007] in the SPIBB framework.

The empirical analysis shows a large difference in the number of samples necessary to change the behavior policy of flat SPI algorithms and their factored counterparts. In the SysAdmin domain (Figure 1), the Factored Π_b-SPIBB algorithm manages to compute policies better than the behavior policy given batches with only 100 trajectories, in contrast to the Π_b-SPIBB algorithm, that only shows improvement when $|\mathcal{D}| \geq 2000$. Finally, we find that even without the prior knowledge about the dependency function D, the Factored Π_b-SPIBB algorithm equipped with the structure learning k-meteorologists algorithm can also exploit the independence between features, finding improved policies with 1000 trajectories.

4 Future Work

Some directions for future work include: extension of other model-based safe RL methods to factored environments, such as the robust approach by Petrik *et al.* [2016]; exploitation of other types of structure in the SPIBB framework such as decision trees and relation between objects; and development of a general SPIBB framework to be applied under different assumptions of prior knowledge.

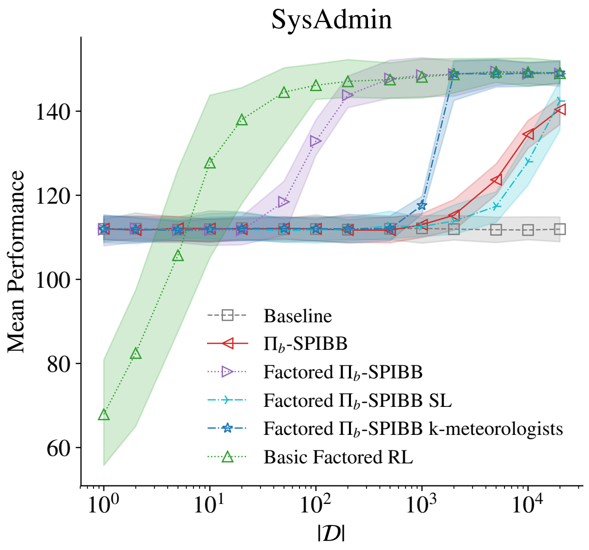

Figure 1: Average performance of the policy computed by different algorithms (y-axis). The x-axis shows the number of trials in $\mathcal{D}$.

Acknowledgments

This research received funding from the Netherlands Organisation for Scientific Research (NWO).

References

[Brafman and Tennenholtz, 2002] Ronen I. Brafman and Moshe Tennenholtz. R-MAX - A General Polynomial Time Algorithm for Near-Optimal Reinforcement Learning. *JMLR*, 3:213–231, 2002.

[Diuk *et al.*, 2009] Carlos Diuk, Lihong Li, and Bethany R. Leffler. The Adaptive k-meteorologists Problem and Its Application to Structure Learning and Feature Selection in Reinforcement Learning. In *ICML*, pages 249–256. ACM, 2009.

[Guestrin *et al.*, 2003] Carlos Guestrin, Daphne Koller, Ronald Parr, and Shobha Venkataraman. Efficient Solution Algorithms for Factored MDPs. *JAIR*, 19:399–468, 2003.

[Laroche *et al.*, 2019] Romain Laroche, Paul Trichelair, and Rémi Tachet des Combes. Safe policy improvement with baseline bootstrapping. In *ICML*, pages 3652–3661. PMLR, 2019.

[Petrik *et al.*, 2016] Marek Petrik, Mohammad Ghavamzadeh, and Yinlam Chow. Safe Policy Improvement by Minimizing Robust Baseline Regret. In *NeurIPS*, pages 2298–2306. Curran Associates, Inc., 2016.

[Simão and Spaan, 2019] Thiago D. Simão and Matthijs T. J. Spaan. Safe Policy Improvement with Baseline Bootstrapping in Factored Environments. In *AAAI*. AAAI Press, 2019.

[Strehl *et al.*, 2007] Alexander L. Strehl, Carlos Diuk, and Michael L. Littman. Efficient Structure Learning in Factored-State MDPs. In *AAAI*, pages 645–650. AAAI Press, 2007.

[Strehl, 2007] Alexander L Strehl. Model-Based Reinforcement Learning in Factored-State MDPs. In *2007 IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning*, pages 103–110. IEEE, 2007.