Reinforcement Learning Robot Control with Progressive State Aggregation

Christos N. Mavridis, Nilesh Suriyarachchi, and John S. Baras
Department of Electrical and Computer Engineering and the Institute for Systems Research, University of Maryland, College Park, USA.

Abstract

While reinforcement learning algorithms based on parametric models can deal with the curse of dimensionality, convergence properties can be difficult to establish and their performance in practice heavily depends on the choice of the basis functions. We propose a reinforcement learning algorithm based on an adaptive aggregation scheme defined by a progressively growing set of codevectors placed in the joint state-action space according to a maximum-entropy vector quantization scheme. The proposed algorithm can be used for robot control and constitutes a two-timescale stochastic approximation algorithm with: (a) a fast component that executes a temporal-difference learning algorithm, and (b) a slow component, based on an online deterministic annealing algorithm, that adaptively partitions the state-action space according to a dissimilarity measure that belongs to the family of Bregman divergences. The proposed online deterministic annealing algorithm constitutes a competitive-learning neural network that shows robustness with respect to the initial conditions, requires minimal hyper-parameter tuning, and provides online control over the performance-complexity trade-off.

Introduction and Problem Definition

Reinforcement learning algorithms are being extensively studied, not only due to their effectiveness in numerous applications [5], but also due to their promise to solve difficult optimal control problems in an online and data-driven fashion. Consider a discrete-time MDP $(\mathcal{X},\mathcal{U},\mathcal{P},C)$ with:

- $\mathcal{X}$ being the state space,
- $\mathcal{U}$ being the action (control) space,
- $\mathcal{P}:(x,u,x')\mapsto \mathbb{P}[x'|x,u]$ being the transition probabilities associated with a stochastic state transition function $f:(x,u)\mapsto x'$, and
- $C:\mathcal{X}\times\mathcal{U}\rightarrow\mathbb{R}_+$, being the immediate cost function, assumed deterministic.

Reinforcement Learning (RL) examines the problem of learning a control policy $u:=(u_0,u_1,\dots)$ that solves the discounted infinite-horizon optimal control problem

$$\begin{aligned} V^*(x_k) :&= \min_u \ \mathbb{E}\left[\sum_{l=k}^{\infty}\gamma^{l-k}C(x_l,u_l)\right] \\ &\overset{(HJB)}{=} \min_u \left\{ C(x_k,u_k)+\gamma\mathbb{E}\left[V^*(x_{k+1})\mid x_k\right] \right\} \\ &= \min_{u_k} Q^*(x_k,u_k) \end{aligned} \tag{1}$$

where $\gamma\in(0,1]$, $V^*:=V^{u^*}$ and $Q^*:=Q^{u^*}$ represent the optimal value and Q functions, respectively. Reinforcement learning algorithms consist mainly of temporal-difference learning algorithms that try to approximate a solution to (1) using iterative optimization methods. The optimization is performed over a finite set of parameters which are used to describe the value (or Q) function. These parameters typically correspond to a parametric model (e.g. a neural network) used for function approximation, or to the different values of the vector $V(\mathcal{X})$ (or $Q(\mathcal{X},\mathcal{U})$), in which case $\mathcal{X}$ and $\mathcal{U}$ are assumed finite either by definition or as a result of discretization. When $\mathcal{X}$ and $\mathcal{U}$ are finite, a widely used approach is the Q-learning algorithm:

$$Q_{j+1}(x,u') = Q_j(x,u') + \alpha_j[C(x,u') + \gamma\min_u Q_j(x',u) - Q_j(x,u')]$$

Extended abstract submitted to the Maryland Robotics Center (MRC) Research Symposium 2021. Research partially supported by the Defense Advanced Research Projects Agency (DARPA) under Agreement No. HR00111990027, by ONR grant N00014- 17-1-2622, and by a grant from Northrop Grumman Corporation. Authors' e-mails: {mavridis, nileshs, baras}@umd.edu.

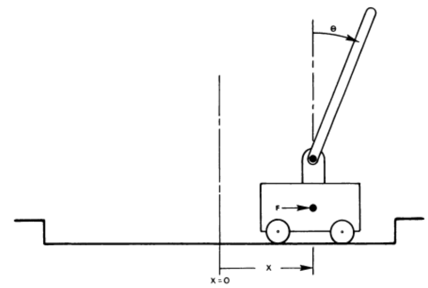

Figure 1: Inverted pendulum configuration to be controlled ([2]).

that provably asymptotically minimizes the mean-squared Bellman error. While reinforcement learning algorithms based on parametric models can deal with the dimensionality issues, convergence properties can be difficult to establish, and their performance in practice heavily depends on the choice of the basis functions [3]. As a middle point between the two approaches, state aggregation has been proposed as a quantization scheme for large or infinite spaces [1, 8], and can be viewed as a special case of linear models with the basis functions being indicator functions of a partition of the state/action space [11]. Although this simplicity of the feature space is often desirable, crude approximation can decrease the overall performance of the algorithm, while state aggregation schemes are typically fixed and ad-hoc [4], which results to a sub-optimal representation of the space.

Methodology and Results

We propose an adaptive state/action aggregation algorithm, based on the results of [8], that updates the space partition with a vector quantization algorithm [7] while implementing a version of Q-learning in the discretized space. This leads to a better representation of the state/action space compared to naive discretization, and with fewer number of discrete states. We consider an MDP $(\mathcal{X},\mathcal{U},\mathcal{P},C)$, where $S\subseteq\mathbb{R}^{d_X}$, $S\subseteq\mathbb{R}^{d_U}$ are compact convex sets. We are interested in the approximation of the quality function $Q:\mathcal{X}\times\mathcal{U}\rightarrow\mathbb{R}_+$. To this end, we define a quantizer $Q_P(x,u)=\sum_{h=1}^K \mu_h \mathbb{1}_{[(x,u)\in P_h]}$, where $\{P_h\}_{h=1}^K$ is a partition of $\mathcal{X}\times\mathcal{U}$. The parameters $\mu_h:=(m_h,v_h)$ define a state-action aggregation scheme with k clusters (aggregate state-action pairs), each represented by $m_h\in\mathcal{X}$ and $v_h\in\mathcal{U}$, for $h=1,\dots,K$. We use the online deterministic annealing (ODA) algorithm (Alg. 1) as an online greedy algorithm that finds an optimal representation of the data space with respect to a trade-off between minimum average distortion and maximum entropy. The online deterministic annealing algorithm is a prototype-based clustering algorithm that is robust with respect to the initial conditions [6], and provides a means to progressively adjust the number of clusters used, via an intuitive bifurcation phenomenon that controls the performance-complexity trade-off created by the interplay of minimum-distortion and maximum-entropy. After convergence, if the representation is meaningful, the finite set $\{\mu_h\}_{h=1}^K$, where $\mu_h\in\mathcal{X}\times\mathcal{U}$, can be used directly for piece-wise constant approximation of the Q function. We stress that the cardinality K of the set of representatives of the space $\mathcal{X}\times\mathcal{U}$ is automatically chosen by Alg. 1 and progressively increases, as needed, with respect to the complexity-accuracy trade-off presented above.

In essence, we are approximating the Q function with a piece-wise constant parametric model with the parameters that define the partition living in the data space and being chosen by the vector quantization algorithm 1. We can design a reinforcement learning algorithm as a two-timescale stochastic approximation algorithm with (a) a fast component that updates the values $Q:=\{Q(h)\}_{h=1}^K$ with a temporal-difference learning algorithm, and (b) a slow component that updates the representation $\mu:=\{\mu_h\}_{h=1}^K$ based on Alg. 1. Such a framework can incorporate different reinforcement learning algorithms, including the proposed algorithm presented in Alg. 2.

In Fig. 2 we compare the average number of timesteps (here $N_t = 1000$) with respect to the number of aggregate states used, for three different state aggregation algorithms. The first one is naive discretization without state aggregation, the second is the SOM-based algorithm proposed in [8], and the last is the proposed algorithm Alg. 2. We initialize the codevectors μ by uniformly discretizing over $\hat{S} \times \{-10, 10\}$, for $\hat{S} = [-1,1] \times [-4,4] \times [-1,1] \times [-4,4]$. We use $K \in \{16, 81, 256, 625\}$ clusters, corresponding to a standard discretization scheme with only $n \in \{2,3,4,5\}$ bins for each dimension. As expected, state aggregation outperforms standard discretization of the state-action space. The ability to progressively adapt the number and placement of the centroids of the aggregate states is an important property of the proposed algorithm, 5 instances of which are presented in Fig. 2 for different parameters T_{min}, which result to $K \in \{56, 118, 136, 202, 252\}$ aggregate states. As shown, the behavior of Alg. 2 depends on the temperature schedule T, as well as on hyperparameters such as the the profile of the stepsizes α_i and β_i.

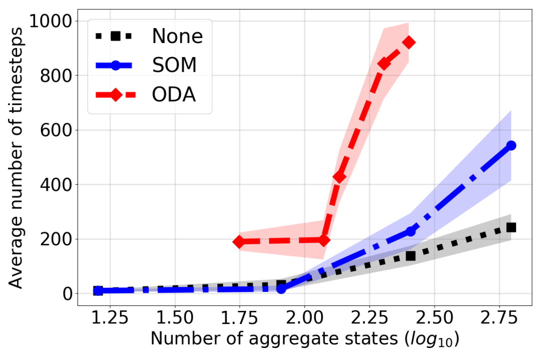

Figure 2: Average number of timesteps ($N_t = 1000$) over number of aggregate states used. (red) the proposed algorithm. (black) Q-learning without state aggregation. (blue) the SOM-based algorithm of [8].

Future Directions

It is natural to seek smoother approximations, that can incorporate the progressively growing nature of the online deterministic annealing algorithm. To this end, Gaussian processes offer a useful candidate, as they constitute non-parametric regressors that allow for the quantification of the uncertainty of the model in each region of the space [9]. However, Gaussian processes are not useful for applications with large datasets, since the time complexity for training is $O(n^3)$, where n is the number of known data points. This also rules out the straightforward use of Gaussian processes in an online fashion. The prediction can be conditioned on just a subset of points, which is, however, typically learned by solving a large optimization problem over the entire dataset [10]. It is possible to use of the codevectors generated by the online deterministic annealing algorithm (Alg. 1) as a training set for Gaussian process regression, which, in turn, can be used in the proposed framework for reinforcement learning robot control.

Algorithm 1 State-Action Aggregation Algorithm (ODA)

Select parameters and initial configuration $\{\mu_i\}$
while $K < K_{max}$ **and** $T > T_{min}$ **do**
  Perturb $\mu^i \leftarrow \{\mu_i + \delta, \mu_i - \delta\}, \forall i$
  Set $n \leftarrow 0$
  repeat
    Observe state x
    for $i = 1, \ldots, K$ **do**
      Update:

$$p(\mu_i|x) \leftarrow \frac{p(\mu_i) e^{-\frac{d_\phi(x,\mu_i)}{T}}}{\sum_i p(\mu_i) e^{-\frac{d_\phi(x,\mu_i)}{T}}}$$

$$p(\mu_i) \leftarrow p(\mu_i) + \beta_n \left[p(\mu_i|x) - p(\mu_i) \right]$$

$$\sigma(\mu_i) \leftarrow \sigma(\mu_i) + \beta_n \left[x p(\mu_i|x) - \sigma(\mu_i) \right]$$

$$\mu_i \leftarrow \frac{\sigma(\mu_i)}{p(\mu_i)}$$

      $n \leftarrow n+1$
    end for
  until Convergence
  Keep effective codevectors
  Lower temperature $T \leftarrow \gamma T$
end while

Algorithm 2 Reinforcement Learning Algorithm with ODA

Initialize $\mu_h, Q_0(h), \forall h \in \{1, \ldots, K\}$
repeat
  Observe x and find $h = \arg\min_{\tau=1,\ldots,k} d_\phi((x,u'), \mu_\tau)$
  Choose $u' = \pi_L(h|\mu)$
  Observe $x' = f(x,u')$ and find $h' = \arg\min_{\tau=1,\ldots,k} d_\phi(x', \mu(\tau))$
  Update $Q(h)$:

$$Q_{i+1}(h) = Q_i(h) + \alpha_i [C(x,u') + \gamma \min_u Q_i(h') - Q_i(h)]$$

  if $i \mod N = 0$ **then**
    Update partition $\mu := \{\mu_h\}_{h=1}^K$ using Alg. 1
  end if
until Convergence
Update Policy: $u^*(x) = \arg\min_u \{ Q(h(x)) \}$

References

[1] JS Baras and VS Borkar. A learning algorithm for markov decision processes with adaptive state aggregation. In *Proceedings of the 39th IEEE Conference on Decision and Control (Cat. No. 00CH37187)*, volume 4, pages 3351–3356. IEEE, 2000.

[2] Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, pages 834–846, 1983.

[3] Christoph Dann, Gerhard Neumann, Jan Peters, et al. Policy evaluation with temporal differences: A survey and comparison. *Journal of Machine Learning Research*, 15:809–883, 2014.

[4] Abraham George, Warren B Powell, and Sanjeev R Kulkarni. Value function approximation using multiple aggregation for multiattribute resource management. *Journal of Machine Learning Research*, 2008.

[5] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[6] Christos Mavridis and John Baras. Online deterministic annealing for classification and clustering, 2021.

[7] Christos N Mavridis and John S Baras. Convergence of stochastic vector quantization and learning vector quantization with bregman divergences. In *21rst IFAC World Congress*. IFAC, 2020.

[8] Christos N Mavridis and John S Baras. Vector quantization for adaptive state aggregation in reinforcement learning. In *2021 American Control Conference (ACC)*. IEEE, 2021.

[9] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.

[10] Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. *Advances in neural information processing systems*, 18:1257–1264, 2005.

[11] John N Tsitsiklis and Benjamin Van Roy. Feature-based methods for large scale dynamic programming. *Machine Learning*, 22, 1996.