Automatica 45 (2009) 2471–2482

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Natural actor–critic algorithms[☆]

Shalabh Bhatnagar[a,*], Richard S. Sutton[b], Mohammad Ghavamzadeh[c], Mark Lee[b]

[a] Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560 012, India
[b] The RLAI Laboratory, Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada T6G 2E8
[c] INRIA Lille - Nord Europe, Team SequeL, France

ARTICLE INFO

Article history:
Received 23 May 2007
Received in revised form
1 January 2009
Accepted 3 July 2009
Available online 27 August 2009

Keywords:
Actor–critic reinforcement learning algorithms
Policy-gradient methods
Approximate dynamic programming
Function approximation
Two-timescale stochastic approximation
Temporal difference learning
Natural gradient

ABSTRACT

We present four new reinforcement learning algorithms based on actor–critic, natural-gradient and function-approximation ideas, and we provide their convergence proofs. Actor–critic reinforcement learning methods are online approximations to policy iteration in which the value-function parameters are estimated using temporal difference learning and the policy parameters are updated by stochastic gradient descent. Methods based on policy gradients in this way are of special interest because of their compatibility with function-approximation methods, which are needed to handle large or infinite state spaces. The use of temporal difference learning in this way is of special interest because in many applications it dramatically reduces the variance of the gradient estimates. The use of the natural gradient is of interest because it can produce better conditioned parameterizations and has been shown to further reduce variance in some cases. Our results extend prior two-timescale convergence results for actor–critic methods by Konda and Tsitsiklis by using temporal difference learning in the actor and by incorporating natural gradients. Our results extend prior empirical studies of natural actor–critic methods by Peters, Vijayakumar and Schaal by providing the first convergence proofs and the first fully incremental algorithms.

1. Introduction

Many problems of scientific and economic importance are optimal sequential decision problems and as such can be formulated as Markov decision processes (MDPs) (Bertsekas & Tsitsiklis, 1996; Rust, 1996; White, 1993). In some cases, MDPs can be solved analytically, and in many cases they can be solved iteratively by dynamic programming or linear programming. However, in other cases these methods cannot be applied either because the state space is too large, a system model is available only as a simulator, or no system model is available. It is in these cases that the techniques and algorithms of reinforcement learning (RL) may be helpful.

Reinforcement learning (Bertsekas & Tsitsiklis, 1996; Sutton & Barto, 1998) can be viewed as a broad class of sample-based methods for solving MDPs. In place of a model, these methods use sample trajectories of the system and the controller interacting, such as could be obtained from a simulation. It is not unusual in practical applications for such a simulator to be available when an explicit transition-probability model of the sort suitable for use by dynamic or linear programming is not (Crites & Barto, 1998; Tesauro, 1995). Reinforcement learning methods can also be used with no model at all, by obtaining sample trajectories by direct interaction with the system (Kohl & Stone, 2004; Ng et al., 2004).

One of the biggest challenges to solve MDPs with conventional methods is handling large state (and action) spaces. This is sometimes known as the "curse of dimensionality" because of the tendency of the size of a state space to grow exponentially with the number of its dimensions. The computational effort required to solve an MDP thus increases exponentially with the dimension and cardinality of the state space. A natural and venerable way of addressing the curse is to approximate the value function and policy parametrically with a number of parameters much smaller than the size of the state space (Bellman & Dreyfus, 1959). However a straightforward application of such function-approximation methods to dynamic programming has not proved effective on large problems. Some work with RL and function approximation has also run into problems of convergence and instability (Baird, 1995; Boyan & Moore, 1995), but about a decade ago it was established that if trajectories were sampled according to their distribution under the target policy (the on-policy distribution) then convergence could be assured for linear feature-based function approximators (Sutton, 1996; Tadic, 2001; Tsitsiklis & Van Roy, 1997).

[☆] This paper was not presented at any IFAC meeting. This paper was recommended for publication in revised form by Associate Editor Giuseppe De Nicolao under the direction of Editor Ian R. Petersen.

[*] Corresponding author. Tel.: +91 80 2293 2987; fax: +91 80 2360 2911.
E-mail addresses: shalabh@csa.iisc.ernet.in (S. Bhatnagar), sutton@cs.ualberta.ca (R.S. Sutton), mohammad.ghavamzadeh@inria.fr (M. Ghavamzadeh), mlee@cs.ualberta.ca (M. Lee).

0005-1098/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.automatica.2009.07.008

Reinforcement learning's most impressive successes have in fact been on problems with extremely large state spaces that could not have been solved without function approximation (Crites & Barto, 1998; Ng et al., 2004; Tesauro, 1995). The ability of sample-based methods to use function approximation effectively is one of the most important reasons for interest in RL within the engineering disciplines.

Policy-gradient methods are some of the simplest RL algorithms and provide both a good illustration of RL and a foundation for the actor–critic methods that are the primary focus of this paper. In policy-gradient methods, the policy is taken to be an arbitrary differentiable function of a parameter vector $\theta \in \mathcal{R}^d$. Given some performance measure $J : \mathcal{R}^d \rightarrow \mathcal{R}$, we would like to update the policy parameter in the direction of the gradient:

$$\Delta\theta \propto \nabla_\theta J(\theta). \tag{1}$$

The gradient is not directly available of course, but sample trajectories can be used to construct unbiased estimators of it, estimators that can be used in a stochastic approximation of the actual gradient. This is the basic idea behind all policy-gradient methods (Aleksandrov, Sysoyev, & Shemeneva, 1968; Baxter & Bartlett, 2001; Bhatnagar, 2005, 2007; Ghavamzadeh & Mahadevan, 2003; Ghavamzadeh & Engel, 2007a,b; Glynn, 1990; Konda & Tsitsiklis, 2003; Marbach & Tsitsiklis, 2001; Peters & Schaal, 2008; Sutton, McAllester, Singh, & Mansour, 2000; Williams, 1992). Theoretical analysis and empirical evaluations have highlighted a major shortcoming of these algorithms, namely, the high variance of their gradient estimates, and thus the slow convergence and sample inefficiency.

One possible solution to this problem, proposed by Kakade (2002) and then refined and extended by Bagnell and Schneider (2003) and by Peters, Vijayakumar, and Schaal (2003), is based on the idea of *natural* gradients previously developed for supervised learning by Amari (1998). In the application to RL, the policy gradient in (1) is replaced with a natural version. This is motivated by the intuition that a change in the policy parameterization should not influence the result of the policy update. In terms of the policy update rule (1), the move to natural gradient amounts to linearly transforming the gradient using the inverse Fisher information matrix of the policy. In empirical evaluations, natural policy gradient has sometimes been shown to outperform conventional policy-gradient methods (Bagnell & Schneider, 2003; Kakade, 2002; Peters et al., 2003; Richter, Aberdeen, & Yu, 2007). Moreover, the use of natural gradients can lead to simpler, and in some cases, more computationally efficient algorithms. Three of the four algorithms we introduce in this paper incorporate natural gradients.

In this paper we focus on a sub-class of policy-gradient methods known as actor–critic algorithms. These methods can be thought of as reinforcement learning analogs of dynamic programming's policy iteration method. Actor–critic methods are based on the simultaneous online estimation of the parameters of two structures, called the *actor* and the *critic*. The actor corresponds to a conventional action–selection policy, mapping states to actions in a probabilistic manner. The critic corresponds to a conventional state-value function, mapping states to expected cumulative future reward. Thus, the critic addresses a problem of prediction, whereas the actor is concerned with control. These problems are separable, but are solved simultaneously to find an optimal policy. A variety of methods can be used to solve the prediction problem, but the ones that have proved most effective are those based on some form of temporal difference (TD) learning (Sutton, 1988), in which estimates are updated on the basis of other estimates. Such "bootstrapping methods" (Sutton & Barto, 1998) can be viewed as a way of accelerating learning by trading bias for variance.

Actor–critic methods were among the earliest to be investigated in reinforcement learning (Barto, Sutton, & Anderson, 1983; Sutton, 1984). They were largely supplanted in the 1990s by methods that estimate action-value functions (mappings from states and actions to the subsequent expected return) that are then used directly to select actions without constructing an explicit policy structure. The action-value approach was initially appealing because of its simplicity, but theoretical complications arose when it was combined with function approximation: these methods do not converge in the normal sense, but rather may "chatter" in the neighborhood of a good solution (Gordon, 1995). These complications lead to renewed interest in policy-gradient methods. Policy-gradient methods without bootstrapping can easily be proved convergent, but can suffer from high variance resulting in slow convergence as mentioned above, motivating their combination with bootstrapping temporal difference methods as in actor–critic algorithms.

In this paper we introduce four novel actor–critic algorithms along these lines. For all four methods we prove convergence of the parameters of the policy and state-value function to a small neighborhood of the set of local maxima of the average reward when the TD error inherent in the function approximation is small. Our results are an extension of our prior work (Bhatnagar, Sutton, Ghavamzadeh, & Lee, 2008), and of prior work on the convergence of two-timescale stochastic approximation recursions (Abdulla & Bhatnagar, 2007; Bhatnagar & Kumar, 2004; Konda & Borkar, 1999; Konda & Tsitsiklis, 2003). That work had previously shown convergence to a locally optimal policy for several non-bootstrapping algorithms with or without function approximation. Convergence of general two-timescale stochastic approximation algorithms has been shown under some assumptions in Borkar (1997). Konda and Tsitsiklis (2003) have shown convergence for an actor–critic algorithm that uses bootstrapping in the critic, but our results are the first to prove convergence when the actor is bootstrapping as well. Our results also extend prior two-timescale results by incorporating natural gradients. Our results and algorithms differ in a number of other, smaller ways from those of Konda and Tsitsiklis; we detail these in Section 6 after the analysis has been presented.

Two other aspects of the theoretical results presented here should be mentioned at the outset. First, one of the issues that arises in policy-gradient methods is the selection of a baseline reward level. In contrast to previous work, we show that, in an actor–critic setting when compatible features are used, the baseline that minimizes the estimator variance for any given policy is in fact the state-value function. Second, for the case of a fixed policy we use a recent result by Borkar and Meyn (2000) to provide an alternative, simpler proof of convergence (cf. Tsitsiklis & Van Roy, 1997; Tsitsikis & Van Roy, 1999) in the Euclidean norm of TD recursions.

In this paper we do not explicitly consider the treatment of eligibility traces ($\lambda > 0$ in TD(λ) (Sutton, 1988)), which have been shown to improve performance in cases of function approximation or partial observability, but we believe the extension of all of our results to general λ would be straightforward. Less clear is how or whether our results could be extended to least-squares TD methods (Boyan, 1999; Bradtke & Barto, 1996; Farahmand, Ghavamzadeh, Szepesvári, & Mannor, 2009; Lagoudakis & Parr, 2003). It is not clear how to satisfactorily incorporate these methods in a context in which the policy is changing. Our proof techniques do not immediately extend to this case and we leave it for future work. We do consider the use of approximate advantages as in the works of Baird (1993) and of Peters and Schaal (2008). Because of space limitations, we do not present empirical results obtained from our algorithms in this paper but these can be seen in Section 8 of our technical report (Bhatnagar, Sutton, Ghavamzadeh, & Lee, 2009).