Convergent Reinforcement Learning with Value Function Interpolation

Csaba Szepesvári SZEPES@MINDMAKER.HU
Mindmaker Ltd., Budapest 1121, Konkoly Thege M. u. 29–33, HUNGARY

Abstract

We consider the convergence of a class of reinforcement learning algorithms combined with value function interpolation methods using the methods developed in (Littman & Szepesvári, 1996). As a special case of the obtained general results, for the first time, we prove the (almost sure) convergence of Q-learning when combined with value function interpolation in uncountable spaces.

1. Introduction

Since the early days of dynamic programming researchers interested in solving problems with large or even infinite state spaces combined function approximators and value backups. Some of the first examples include (Bellman & Dreyfus, 1959; Reetz, 1977; Werbös, 1977). The approach at these days was to use some interpolation method and compute the value backups on a set of representative states.

Reinforcement learning (RL) has grown out of the work of (Samuel, 1959) who himself has used a form of value function approximation. Sometimes RL is defined as large-scale approximate dynamic programming combined with learning techniques.

One main stream of RL employs gradient-based methods. Focusing on value function approximation only, examples include $TD(\lambda)$ (Sutton, 1988), Bellman-error (Schweitzer & Seidman, 1985) and residual algorithms (Baird, 1995; Baird & Moore, 1999). Some successful applications of these methods are described in (Tesauro, 1994; Zhang & Dietterich, 1995; Baxter et al., 1998). Another main stream employs contructive (local) methods. Decision tree based methods were used e.g. in (Chapman & Kaelbling, 1991) and more recently in (Wang & Dietterich, 1999). Discretization using triangularization has been advocated in (Munos, 1997), (sometimes adaptive) state aggregation was employed in e.g. (Moore & Atkeson, 1995; Tsitsiklis & Van Roy, 1996; Gordon, 1995). Recently, locally weighted regression models have been used by (Smart & Kaelbling, 2000).

Regression literature teaches us that the advantage of constructive or local methods is that the behavior of the method can be understood much better than that of the gradient-based approaches. As discussed in (Atkeson et al., 1997) advantages of constructive local methods include *(i)* no interference, *(ii)* incremental improvement (no local minima as in gradient methods), *(iii)* one-shot learning and *(iv)* that the incorporation of new data is cheap. The main disadvantage of local methods is that value prediction slows down as the number of training examples increases and their performance degrades as the dimensionality of the space increases.

Unfortunately, it seems that the nice properties of constructive approximations do not carry over seemlessly to the combinations of local methods and reinforcement learning. This has been observed by many researchers (e.g. (Boyan & Moore, 1995; Baird, 1995)) and then researchers were urged to come up with convergent algorithms. In (Gordon, 1995) and independently in (Tsitsiklis & Van Roy, 1996) convergence results were derived for approximate dynamic programming when the "value-fitting operator" was chosen to be a non-expansion[1] More precisely, in (Gordon, 1995) an algorithm of the form $V_{t+1} = \mathcal{G}TV_t$ was considered, where T is the value backup operator and $\mathcal{G}$ is a non-expansion w.r.t. the supremum norm. In discounted problems T is known to be a contraction (w.r.t. the supremum norm) and thus $\hat{\mathcal{T}} = \mathcal{G}T$ is a contraction as well and the algorithm converges to the unique fixed point of $\hat{\mathcal{T}}$. In order to make the algorithm practical one decomposes $\mathcal{G}$ into the product of an operator $\mathcal{P}$ that maps the space of value functions into a low-dimensional space Θ and an operator $\mathcal{E}$ that maps Θ to the space of value functions. Then, by defining $\theta_t = \mathcal{P}V_t$ one gets

$$\theta_{t+1} = \mathcal{P}T\mathcal{E}\theta_t. \tag{1}$$

[1] In (Tsitsiklis & Van Roy, 1996) the authors actually derived a slightly less restrictive condition. We shall discuss this difference later.

Gordon proposed to use "averagers" as $\mathcal{E}$ and discussed the problems with the extension of the method to the learning scenario when T is unknown.

In (Tsitsiklis & Van Roy, 1996), T is still assumed to be known, although these authors mention that their method could be extended to the case when T is replaced by its Monte-Carlo approximation. State aggregation with approximate gradual value iteration is studied. Iteration 1 is replaced by $\theta_{t+1,i} = (1-\alpha_t)\theta_{t,i} + \alpha_t(T(\mathcal{E}\theta_t))(X_t^{(i)})$, where $X_t^{(i)}$ is a sample from the ith "cluster" and $(\mathcal{E}\theta)(x) = \sum_{i=1}^n \theta_i \chi_{A_i}(x)$, where $\{A_i\}$ forms a partition of the state space $\mathcal{X}$. Convergence is shown and tight error bounds are derived. In the same article, another algorithm in the style of Equation 1 is presented that employs an interpolative linear operator $\mathcal{E}$. Convergence and tight error bounds are derived.

Apperantly, the only work known to us which does not make use of a model and which employs local learning is that of (Singh et al., 1995). The authors of these article propose a combination of Q-learning with what they call "soft-state aggregation". The soft-clusters are represented by probability distributions $P(i|x)$: $\sum_{i=1}^n P(i|x) = 1$, $P(i|x) > 0$. A fixed persistently exciting policy is used to generate a stationary state-action sequence (X_t, A_t) over a finite state and action space. In each time-step a random cluster index I_t is generated from $P(\cdot|X_t)$ and the Q-values are updated using the experience samples $< I_t, A_t, R_t, I_{t+1} >$. Amost sure convergence is guaranteed since $< I_t, A_t, R_t, I_{t+1} >$ has a stable statistics (Singh et al., 1995).

Although this algorithm is guaranteed to converge, it is quite inefficient as it updates only one Q-value in each time step even if the clusters "overlap". In an earlier work Q-learning was studied with "multi-state" updates (Ribeiro & Szepesvári, 1996; Szepesvári & Littman, 1999). In the algorithm considered multiple states are updated in each time step (hence the name of it), the original motivation being to share samples across "neighboring" Q-values to speed up the convergence process. Convergence results and bounds on the error introduced were given.

One main motivation of the present article is to extend this algorithm to large or even infinite state spaces. This is achieved by combining the ideas of (Tsitsiklis & Van Roy, 1996) with that of (Ribeiro & Szepesvári, 1996). The key idea is to employ interpolative function approximators. This way a familiar error recursion can be shown to hold for the error in the parameter space. Following the ideas of (Szepesvári & Littman, 1999), instead of showing the convergence of this combined algorithm directly we derive a generic convergence theorem of which the given algorithm will be shown to be a special case. Our results will show convergence of on-line learning algorithms even in infinite state-spaces. The actual form of the function approximator employed is left unspecified apart from the requirement that the corresponding operator must be a non-expansion.

The organization of the paper is as follows: In Section 2 the basic definitions are given and the notation is introduced. In Section 3 the form of the generic algorithm is proposed. The main convergence result along with its application to Q-learning and approximate value iteration is presented in Section 4. Conclusions are drawn and some open issues are discussed in Section 5.

2. Definitions

We assume that the reader is familiar with basic concepts of Markov decision processes (MDPs). In this section we introduce our notation, but do not provide any explanation of the concepts used.

2.1 MDPs

An MDP is a tuple $\mathcal{M} = (\mathcal{X}, \mathcal{A}, p, r, \gamma)$, where $\mathcal{X}$ is a set of states, $\mathcal{A}$ is a set of actions, p is the transition probability density function for non-countable state spaces and the transition probability function for countable state spaces, r is the reward function and γ is the discount factor.

The optimal value function associated with the MDP $\mathcal{M}$ shall be denoted by V^*, the value operator by T, the optimal policy by π^*. The sup-norm shall be denoted by $\|\cdot\|$. The space of real-valued bounded functions over a set $\mathcal{X}$ will be denoted by $B(\mathcal{X})$.

2.2 Interpolative Function Approximation

A *function approximator* F maps a parameter space Θ into the set of functions of interest, in our case $B(\mathcal{X})$, i.e., $F : \Theta \to B(\mathcal{X})$. Usually, Θ is a finite dimensional space, typically $\Theta = \mathbb{R}^k$ for some $k > 0$. For convenience, we shall denote the function $F(\theta)$ by F_θ.

Besides choosing a function approximator, one also needs to choose a *method* to construct the parameter given some data. In this article we will consider *interpolative methods*, by which we mean the following: given the data $\mathcal{D} = ((x_1, \dots, x_n), (v_1, \dots, v_n))$ with $x_i \neq x_j$, an interpolative method $R : \mathcal{X}^n \times \mathbb{R}^n \to \Theta$ chooses a parameter vector θ such that $F_\theta(x_i) = v_i$ for all $i = 1, 2, \dots, n$.

Fix the basis points $\mathbf{x} = (x_1, \dots, x_n)$, $x_i \neq x_j$. Given a function V of $B(\mathcal{X})$ *the interpolative representation of* V as given by the pair (F, R) is F_θ, where $\theta = R(\mathcal{D}_V)$ and $\mathcal{D}_V = (x_1, \dots, x_n), (V(x_1), \dots, , V(x_n))$. Let $\mathcal{G} : B(\mathcal{X}) \to B(\mathcal{X})$ be the interpolative operator defined by $\mathcal{G}V = F(R(\mathcal{D}_V))$.

The mapping that "projects" functions in $B(\mathcal{X})$ onto $\mathbb{R}^n$ given the basis points $\mathbf{x}$ shall be denoted by $\mathcal{P}$. $\mathcal{P} : B(\mathcal{X}) \to \mathbb{R}^n$, $(\mathcal{P}V)_i = V(x_i)$. Further, let us define $\mathcal{E} : \mathbb{R}^n \to B(\mathcal{X})$ by $\mathcal{E}v = \mathcal{G}V$, where V is any function in $B(\mathcal{X})$ satisfying $V(x_i) = v_i$, $i = 1, 2, \dots, n$. $\mathcal{E}$ is well-defined, by the definitions of $\mathcal{G}$ and R.

It is easy to see that these operators satisfy the equations

$$\mathcal{E}\mathcal{P}V = \mathcal{G}V, \quad (2)$$

$$\mathcal{P}\mathcal{E} = \mathrm{id}_{\mathbb{R}^n}, \quad (3)$$

where $\mathrm{id}_{\mathbb{R}^n}$ is the identity over $\mathbb{R}^n$. Here the second equation states that $\mathcal{G}$ is interpolative. Note that $\mathcal{P}$ is linear: $\mathcal{P}(\lambda_1 U + \lambda_2 V) = \lambda_1 \mathcal{P}U + \lambda_2 \mathcal{P}V$, $U, V \in B(\mathcal{X}), \lambda_1, \lambda_2 \in \mathbb{R}$. Further, $P|U| = |PU|$, where $|U|$ is defined by $|U|(x) = |U(x)|$.

In the algorithms and the analysis below, for the sake of generality, we shall take $\mathcal{G}$ as our starting point together with its decomposition into the projection ($\mathcal{P}$) and "expansion" ($\mathcal{E}$) operators satisfying Equations 2 and 3. We shall require below that $\mathcal{G}$ be a non-expansion w.r.t the supremum norm.[2] Given the decomposition above this is equivalent to requiring that $\mathcal{E}$ is a non-expansion ($\mathbb{R}^n$ is taken with the ℓ^∞ norm):

Proposition 2.1. *Let $\mathcal{G}, \mathcal{P}, \mathcal{E}$ satisfy 2 and 3, where $\mathcal{P}$ is the projection operator for some set of basis points. Then $\mathcal{G}$ is a non-expansion if and only if $\mathcal{E}$ is a non-expansion. Further,*

$$\|\mathcal{P}\mathcal{G}U - \mathcal{P}\mathcal{G}V\| = \|\mathcal{P}U - \mathcal{P}V\| \quad (4)$$

$$\|\mathcal{P}U - \mathcal{P}V\| \leq \|U - V\|, \quad (5)$$

$$\|\mathcal{G}U - \mathcal{G}V\| \leq \|\mathcal{P}\mathcal{G}U - \mathcal{P}\mathcal{G}V\|. \quad (6)$$

Proof. Assume that $\mathcal{G}$ is a non-expansion. Let $u, v \in \mathbb{R}^n$ be arbitrary. Choose $U, V \in B(\mathcal{X})$ such that $\mathcal{P}U = u$ and $\mathcal{P}V = v$ and $\|U - V\| = \|u - v\|$. Then $\|\mathcal{E}u - \mathcal{E}v\| = \|\mathcal{E}\mathcal{P}U - \mathcal{E}\mathcal{P}V\| = \|\mathcal{G}U - \mathcal{G}V\| \leq \|U - V\| = \|u - v\|$. This proofs that $\mathcal{E}$ is a non-expansion. Now, assume that $\mathcal{E}$ is a non-expansion. Let $U, V \in B(\mathcal{X})$ be arbitrary. $\|\mathcal{G}U - \mathcal{G}V\| \leq \|\mathcal{P}U - \mathcal{P}V\| \leq \|U - V\|$.

The first two inequalities are trivial. The third one follows since $\|\mathcal{G}U - \mathcal{G}V\| \leq \|\mathcal{P}U - \mathcal{P}V\|$ (since $\mathcal{E}$ is a non-exponasion) and $\|\mathcal{P}U - \mathcal{P}V\|$ is equal to $\|\mathcal{P}\mathcal{G}U - \mathcal{P}\mathcal{G}V\|$ by 4. □

Many well known interpolation operators ($\mathcal{E}$) are non-expansions. Among them are the 0th order spline interpolation with $\mathcal{E}u = \sum_{i=1}^n u_i \chi_{A_i}$, where $\mathcal{X}$ is the disjoint union of the sets $A_1, \dots, A_n$ and χ_A is the characteristic function of A (corresponding to state aggregation), certain higher order spline-interpolations, and radial basis function (RBF) interpolation, which is a special case of kernel-based approximation. All these interpolation architectures assume a linear form: $\mathcal{E}u = u^T \phi$ for a suitably chosen function $\phi : \mathcal{X} \to \mathbb{R}^n$. ϕ is called the basis function and can either be explicitly or implicitly given. For example, if the sets A_i above are given explicitly then $\phi = (\chi_{A_1}, \dots, \chi_{A_n})$. If A_i is given as the sets corresponding to the leafs of a decision tree then the basis function is implicit, – it is encoded into the decision tree algorithm. In the case of RBF networks $\phi = U^{-T}\psi$, where $\psi : \mathcal{X} \to \mathbb{R}^n$ is a mapping such that $\psi_i(x) = K(x_i, x)$ for some radially symmetrical kernel function K, and U is defined by $u_{ij} = \psi_i(x_j) = K(x_i, d(x_i, x_j))$.[3]

2.3 Asynchronous Stochastic Dynamic Programming

We will make use of the notation of the paper (Littman & Szepesvári, 1996) so as we can capture many asynchronous stochastic dynamic programming algorithms with a single algorithm and convergence proof. Let $\mathcal{T}_t : B \times B \to B$ be a sequence of stochastic operators satisfying certain regularity conditions. The algorithms considered in (Szepesvári & Littman, 1999) take the form

$$V_{t+1} = \mathcal{T}_t(V_t, V_t). \quad (7)$$

In order to shed some light on the intuitions behind this notation consider (lookup table based) Q-learning. Take $B = B(\mathcal{X} \times \mathcal{A})$ and let

$$[\mathcal{T}_t(Q, Q')](x, a) = (1 - \alpha_t(x, a))Q(x, a) + \alpha_t(x, a)\left\{R_t + \gamma \max_b Q'(X_{t+1}, b)\right\}, \quad (8)$$

where $\alpha_t(x, a) = 0$ if $(X_t, A_t) \neq (x, a)$. Here (X_t, A_t) is the state-action pair visited at time step t and R_t is the immediate reward received. Note that the operator sequence $\mathcal{T}_t$ depends on the random trajectory $(X_0, A_0, R_0, X_1, \dots)$ and is therefore random itself. The choice of $\mathcal{T}_t$ is motivated by the observation

[2] Later we will relax this condition in the spirit of (Tsitsiklis & Van Roy, 1996).

[3] Conditions under which the RBF interpolation operator of RBF networks exists and is a non-expansion are given in (Tsitsiklis & Van Roy, 1996).)

that each component of $Q_{t+1} = \mathcal{T}_t(Q_t, Q)$ is a standard one-dimensional stochastic approximation process and thus, under the usual conditions, Q_t converges to $\mathcal{T}_Q Q$ w.p.1, where $\mathcal{T}_Q$ denotes the optimal value operator for Q-functions. Therefore algorithm 7 can be thought of as performing a "diagonal" approximation to $\lim_{t\to\infty} \mathcal{T}_Q^t Q_0$.

Although, asynchronicity does not follow directly from the form of 7, using Q-learning as an example, one can see that the form of the algorithm does allow the different components of the value function to be updated at different rates - hence the algorithm allows asynchronous updates.

In (Szepesvári & Littman, 1999) a general convergence result is presented for this algorithm. In the same article it was shown that this convergence result applies to many well known algorithms, including e.g. model-based reinforcement learning, asynchronous dynamic programming, Q-learning, Q-learning with multi-state updates. This last result will play an important role in this paper.

The convergence analysis of the algorithm builds on the comparision of the sequence $U_{t+1} = T_t(U_t, V^*)$ with V_t. In particular, a recursive error equation is derived for their difference.

3. The RLI Algorithm

The proposed RLI ("reinforcement learning with interpolation") algorithm has the form

$$V_{t+1} = \mathcal{G}_t \mathcal{T}_t(V_t, V_t), \tag{9}$$

in the value function space, for a suitably chosen sequence of random operators and non-expansive interpolation operators $\mathcal{G}_t$. This algorithm can be thought of as the extension of the algorithm proposed by Gordon who studied the iteration $V_{t+1} = \mathcal{G}TV_t$. If one thinks of 7 as the generalization of value-iteration then clearly 9 appears analogous to the algorithm studied by Gordon.

Given the decomposition $\mathcal{G}_t = \mathcal{E}_t\mathcal{P}_t$, where $\mathcal{P}_t$ is a projection operator as in the previous section, one readily derives the parameter space recursion

$$\theta_{t+1} = \mathcal{P}_t\mathcal{T}_t(\mathcal{E}_t\theta_t, \mathcal{E}_t\theta_t), \tag{10}$$

where $V_t = \mathcal{E}_t\theta_t$ or $\theta_t = \mathcal{P}_t V_t$. We will be primarily concerned with the convergence of V_t and the quality of approximation of the optimal value function by the limes. Note that if the limes exists and if $\mathcal{P}_t = \mathcal{P}$ for some $\mathcal{P}$ and for all $t > 0$ then by the continuity of $\mathcal{P}$ and since $\theta_t = \mathcal{P}V_t$, θ_t will also converge to some limiting value.

4. Results

4.1 Convergence Analysis

For the simplicity of presentation let $\mathcal{G}_t = \mathcal{G}$, $\mathcal{P}_t = \mathcal{P}$ and $\mathcal{E}_t = \mathcal{E}$. Following the ideas developed in (Szepesvári & Littman, 1999) we compare $U_{t+1} = \mathcal{G}\mathcal{T}_t(U_t, \hat{V}^*)$ with $V_{t+1} = \mathcal{G}\mathcal{T}_t(V_t, V_t)$. Here $\hat{V}^*$ is the fixed point of $\mathcal{G}T$. Let $\delta_t = |V_t - U_t|$ denote the error process.

First, note that by 6,

$$\begin{aligned} \|\delta_{t+1}\| &= \|\mathcal{G}T_t(V_t, V_t) - \mathcal{G}T_t(U_t, \hat{V}^*)\| \\ &\le \|\mathcal{P}\mathcal{G}T_t(V_t, V_t) - \mathcal{P}\mathcal{G}T_t(U_t, \hat{V}^*)\| \\ &= \|\mathcal{P}V_{t+1} - \mathcal{P}U_{t+1}\| = \|\mathcal{P}\delta_{t+1}\|. \end{aligned} \tag{11}$$

This inequality will play a key role in proving the convergence of V_t as it shows that it is sufficient to prove that $\mathcal{P}\delta_t$ converges to zero w.p.1. This way, the problem is reduced to a finite dimensional problem.

Now, by 3 $\mathcal{P}V_{t+1} = \mathcal{P}\mathcal{E}\mathcal{P}\mathcal{T}_t(V_t, V_t) = \mathcal{P}\mathcal{T}_t(V_t, V_t)$. Similarly, $\mathcal{P}U_{t+1} = \mathcal{P}\mathcal{T}_t(U_t, \hat{V}^*)$. Therefore

$$\mathcal{P}\delta_{t+1} = |\mathcal{P}\mathcal{T}_t(V_t, V_t) - \mathcal{P}\mathcal{T}_t(U_t, \hat{V}^*)|. \tag{12}$$

Now, recall that in (Szepesvári & Littman, 1999) the $\mathcal{T}_t$ operators were shown to satisfy the following conditions in most of the applications:[4]

$$|\mathcal{T}_t(U_1, V) - \mathcal{T}_t(U_2, V)| \le G_t|U_1 - U_2|, \tag{13}$$

$$|\mathcal{T}_t(U, V_1) - \mathcal{T}_t(U, V_2)| \le F_t(\|V_1 - V_2\| + \lambda_t), \tag{14}$$

where $U_1, U_2, V_1, V_2 \in B(\mathcal{X})$ are arbitrary, $\lambda_t \ge 0$ is a random sequence converging to zero w.p.1, G_t, F_t are suitable chosen uniformly bounded non-negative random functions satisfying $0 \le F_t, G_t \le 1$, $F_t \le \gamma(1-G_t)$, and $\lim_{n\to\infty} \|\Pi_{t=t_0}^n G_t\| = 0$ w.p.1 for all $t_0 > 0$ and $0 < \gamma < 1$ is a discount factor (a non-random real number).

Note that in the applications F_t typically corresponds to $\alpha_t\gamma$ and G_t to $1-\alpha_t$, where α_t is the "learning rate" at time step t (cf. Equation 8).

Proceeding formally, using 13 and 14 we get

$$\begin{aligned} \mathcal{P}\delta_{t+1} &\le |\mathcal{P}\mathcal{T}_t(V_t, \hat{V}^*) - \mathcal{P}\mathcal{T}_t(U_t, \hat{V}^*)| + \\ &\quad |\mathcal{P}\mathcal{T}_t(V_t, V_t) - \mathcal{P}\mathcal{T}_t(V_t, \hat{V}^*)| \\ &\le \mathcal{P}(G_t|V_t - U_t|) + \\ &\quad \mathcal{P}(F_t(\|V_t - U_t\| + \|U_t - \hat{V}^*\| + \lambda_t)) \\ &= (\mathcal{P}G_t)\,(\mathcal{P}|V_t - U_t|) + \\ &\quad (\mathcal{P}F_t)\,(\|\delta_t\| + \|U_t - \hat{V}^*\| + \lambda_t). \end{aligned}$$

[4]Here and in what follows multiplication, absolute value, equality and inequality of functions should be understood pointwise.

Now, by 11, $\|\delta_t\| \le \|\mathcal{P}\delta_t\|$ and therefore

$$\mathcal{P}\delta_{t+1} \le (\mathcal{P}G_t)(\mathcal{P}\delta_t) + (\mathcal{P}F_t)(\|\mathcal{P}\delta_t\| + \|U_t - \hat{V}^*\| + \lambda_t). \tag{15}$$

Notice that we have reduced the infinite dimensional error recursion to a finite dimensional one. Now, if $\mathcal{P}G_t$, $\mathcal{P}F_t$ satisfy $0 \le \mathcal{P}F_t, \mathcal{P}G_t \le 1$, $\mathcal{P}F_t \le \gamma(1 - \mathcal{P}G_t)$, and $\lim_{n\to\infty} \|\Pi_{t=t_0}^n \mathcal{P}G_t\| = 0$ w.p.1, for all $t_0 > 0$ and if $\|U_t - \hat{V}^*\|$ converges to zero then by Lemma 26 of (Szepesvári & Littman, 1999) $\mathcal{P}\delta_t$ converges to zero w.p.1.

Similarly to the above analysis, the convergence of U_t can be studied by looking at $\mathcal{P}U_t$. Namely, if $\mathcal{P}U_t$ converges to $\hat{\theta}^*$ then by the continuity of $\mathcal{E}$, U_t must also converge w.p.1.[5] Further, if $\hat{V}^*$ is the fixed point of $\mathcal{G}T$ then $\|U_{t+1} - \hat{V}^*\| = \|\mathcal{G}\mathcal{T}_t(U_t, \hat{V}^*) - \mathcal{G}\hat{V}^*\| \le \|\mathcal{P}U_{t+1} - \mathcal{P}\hat{V}^*\|$ by 6.

4.2 The Main Theorem

Theorem 4.1. *Assume that the sequence $U_{t+1} = \mathcal{G}\mathcal{T}_t(U_t, \hat{V}^*)$ converges to $\hat{V}^*$ w.p.1, where $\hat{V}^*$ is the fixed point of $\mathcal{G}T$. Further, assume that the random variables G_t, F_t satisfy $0 \le \mathcal{P}F_t, \mathcal{P}G_t \le 1$, $\mathcal{P}F_t \le \gamma(1 - \mathcal{P}G_t)$, and $\lim_{n\to\infty} \|\Pi_{t=t_0}^n \mathcal{P}G_t\| = 0$ w.p.1 and $\mathcal{T}_t, F_t, G_t$ satisfy inequalities 13 and 14, where $\lambda_t \ge 0$ converges to zero w.p.1. Then $V_{t+1} = \mathcal{G}\mathcal{T}_t(V_t, V_t)$ converges to $\hat{V}^*$ w.p.1.*

A performance bound follows immediately from the bounds derived for the composite of non-expansion and contraction operators (see e.g. (Gordon, 1995) or Proposition 9 in (Szepesvári & Littman, 1999)).

Theorem 4.2. *Let $\mathcal{G}$ be an interpolative non-expansion over B and $\mathcal{T}$ be a contraction with contraction factor $0 < \gamma < 1$, also given over B. Let $\hat{V}^*$ be the fixed point of $\mathcal{G}\mathcal{T}$ and let V^* be the fixed point of $\mathcal{T}$. Then*

$$\|\hat{V}^* - V^*\| = \frac{2\inf_{v\in\mathbb{R}^n} \|\mathcal{E}v - V^*\|}{(1-\gamma)}.$$

Proof. The theorem follows directly e.g. from Proposition 9 of (Szepesvári & Littman, 1999) and the observation that $\{ V : \mathcal{G}V = V \} = \{ \mathcal{E}v : v \in \mathbb{R}^n \}$. □

Note that the infimum is not necessarily taken at $v^* = \mathcal{P}V^*$, but clearly, $\|\mathcal{G}V^* - V^*\|$ provides an upper bound on $\inf_{v\in\mathbb{R}^n} \|\mathcal{E}v - V^*\|$.

[5] Since $\mathcal{E}$ is a non-expansion, it must be continuous.

4.3 Interpolative Q-learning

Let $\mathcal{G} = \mathcal{E}\mathcal{P}$ be any interpolative, non-expansion defined over $B(\mathcal{X} \times \mathcal{A})$.[6] Let $X_0, A_0, R_0, X_1, A_1, R_1, \ldots$ be a random trajectory generated by a suitable stationary exploration policy (X_t is the state visited at time t, A_t is the action taken at time t and R_t is the reward received at time t).

The straightforward combination of Q-learning and interpolative function approximation generates the sequence $Q_{t+1} = \mathcal{G}\mathcal{T}_t(Q_t, Q_t)$, where $\mathcal{T}_t$ is the Q-learning update operator defined in Equation 8. This algorithm has the problem that it does not change Q_t unless $(X_t, A_t) \in W$. Since this has a zero probability in uncountable state spaces the resulting algorithm will be rather uninteresting. We need to update Q-values at state-action pairs other than the currently visited ones!

In (Ribeiro & Szepesvári, 1996) and later in (Szepesvári & Littman, 1999), the process $Q_{t+1} = \mathcal{T}_t(Q_t, Q_t)$ was studied where

$$\begin{aligned}[\mathcal{T}_t(Q, Q')](z, a) &= (1 - \alpha_t(z, a)s(z, a, X_t))Q(z, a) \\ &+ \alpha_t(z, a)s(z, a, X_t)\left\{R_t + \gamma \max_b Q'(X_{t+1}, b)\right\}\end{aligned} \tag{16}$$

where α_t is a suitably chosen sequence of random learning rates and $s(z, a, x) > 0$ is a spatial smoothening factor. Typically, $s(z, a, x) \le s(z, a, z)$ and $s(z, a, x)$ decays to zero as $\|z - x\| \to \infty$. We will require that s is measurable and uniformly bounded. In the updates we allow $\alpha_t(x, a) > 0$ for state-action pairs $(x, a) \ne (X_t, A_t)$.[7] This way other state-action pairs than the currently visited one will be updated as well - just what we wanted to achieve.[8]

In (Szepesvári & Littman, 1999) it was shown that in finite state spaces, under mild conditions Q_t defined above converges w.p.1. The motivation there, however, was to speed up Q-learning rather than making Q-learning work in infinite state spaces. The analysis required that every state action pair be visited infinitely often. Such a condition is never met in uncountable state spaces.

[6] A special case is when the basis points (x_i, a_i) are chosen such that $W = \{(x_1, a_1), \ldots, (x_n, a_n)\} = \{x_1, \ldots, x_n\} \times A$. This corresponds to approximation in the state space only. In what follows we shall assume this.

[7] We will still need, however, $\alpha_t(x, a) = 0$ unless $a = A_t$. This last condition could be removed by extending s to include "action smoothening" as well.

[8] In (Singh et al., 1995) essentially the same was achieved by sampling which component to update. We believe that our method makes use of the samples more efficiently, but this still needs to be proven!

Here we show that this update rule when combined with interpolative function approximation yields a convergent algorithm. Let $Q_{t+1} = \mathcal{G}\mathcal{T}_t(Q_t, Q_t)$, where $\mathcal{T}_t$ is defined by 16. In the parameter space the algorithm takes the form:

$$\begin{aligned}\theta_{(t+1,i)} &= (1 - \alpha_t(x_i, a_i)s(x_i, a_i, X_t))\theta_{(t,i)} + \\ &+\alpha_t(x_i, a_i)s(x_i, a_i, X_t) \\ &\left\{R_t + \gamma \max_b Q_t(X_{t+1}, b)\right\}, \quad i = 1, 2, \ldots, n,\end{aligned} \tag{17}$$

where $Q_t = \mathcal{E}\theta_t$ and we have used $Q_t(x_i, a_i) = \theta_{t,i}$ which follows from the interpolative property of $\mathcal{G}$.

Now, let us consider $\tilde{Q}_{t+1} = \mathcal{T}_t(\tilde{Q}_t, Q)$, where $Q \in B(\mathcal{X} \times \mathcal{A})$ is fixed. In order to analyze this process we need to make some further assumptions on the sample path $\{(X_t, A_t)_t\}$ and the policy π that generates it. First, we require that $\pi(a|x) > 0$ for all $(x, a) \in \mathcal{X} \times \mathcal{A}$. Second, we require that $\{X_t\}$ be stationary, have a unique invariant measure, $\mu_X(\cdot)$, and let it be positive Harris (e.g. (Tadić, 2001)). It is known (see e.g. (Tadić, 2001), Lemma 9) that under these conditions $(1/(t+1)) \sum_{s=0}^{t} f(X_s) \to \int f(x) d\mu_X(x)$, where f is an arbitrary $L^1(\mu_x)$ function.

Now, by the positivity of s, $\int s(x_i, a_i, x) d\mu_X(x) > 0$ and thus there must exist some $\epsilon > 0$ such that $\mu_X(A) > 0$, where $A = \{x : s(x_i, a_i, x) > \epsilon\}$. Let $n_t(x_i, a_i) = 1 + \sum_{s=0}^{t} \chi(s(x_i, a_i, X_t) > \epsilon)$. Clearly, $n_t(x_i, a_i) \to \infty$ w.p.1, since $(1/(t+1)) \sum_{s=0}^{t} \chi(s(x_i, a_i, X_t) > \epsilon) \to \mu_X(A)$. Therefore if $\alpha_t(x_i, a_i) = \chi(s(x_i, a_i, X_t) > \epsilon)/n_t(x_i, a_i)$ then $\sum_{s=0}^{t} \alpha_s(x_i, a_i)s(x_i, a_i, X_s) \geq \epsilon \sum_{k=0}^{n_t(x_i,a_i)} 1/(k+1)$. For the same sequence, by the boundedness of s, $\sum_{s=0}^{t} (\alpha_s(x_i, a_i)s(x_i, a_i, X_s))^2 \to \infty$ w.p.1.

Now, the analysis presented for multi-state Q-learning in (Szepesvári & Littman, 1999) applies to $\tilde{\theta}_t = \mathcal{P}\tilde{Q}_t$ and shows that it converges to $(\mathcal{P}\mathcal{H})(Q)$, where $\mathcal{H} : B(\mathcal{X} \times \mathcal{A}) \to B(\mathcal{X} \times \mathcal{A})$ is given by

$$\begin{aligned}\mathcal{H}(Q)(z, a) = \int\int \hat{s}(z, a, x)\{R(x, a) + \\ \gamma \max_b Q(y, b)\}p(y|x, a)dyd\mu_X(x),\end{aligned} \tag{18}$$

where $\hat{s}(z, a, x) = s(z, a, x)/(\int s(z, a, x)d\mu_X(x))$. By standard arguments $\mathcal{H}$ can be shown to be a contraction operator with contraction factor γ.

Now let $\tilde{\mathcal{T}} : B(\mathcal{X} \times \mathcal{A}) \to B(\mathcal{X} \times \mathcal{A})$ be defined by $\tilde{\mathcal{T}} = \mathcal{G}\mathcal{H}$ and let $\hat{Q}^*$ be the fixed point of it. We claim that $Q_{t+1} = \mathcal{G}T_t(Q_t, Q_t)$ converges to $\hat{Q}^*$ w.p.1. We shall make use of Theorem 4.1. Let $F_t(z, a) = \gamma\alpha_t(z, a)s(z, a, X_t)$ and $G_t(z, a) = (1 - \alpha_t(z, a)s(z, a, X_t))$. $\lim_{n\to\infty} \|\Pi_{t=t_0}^{n} \mathcal{P}G_t\| = 0$ follows iff $\sum_{t=0}^{\infty} \alpha_t(x_i, a_i)s(x_i, a_i, X_t) = \infty$ w.p.1., $i = 1, 2, \ldots, n$. This proves the following theorem:

Theorem 4.3. *Fix a stochastic exploration policy $\pi(a|x)$ such that the sequence of the generated states X_t has a unique invariant measure μ_X, $\{X_t\}$ is stationary and positive Harris. Assume further that $\pi(a|x) > 0$ for all $a \in \mathcal{A}$ and $x \in \mathcal{X}$. Let $\mathcal{E}$ be a non-expansion, let $\alpha_t(x_i, a_i)$ be defined as in the previous paragraph, where $s(z, a, x) > 0$ is a design parameter. Let s be measurable and bounded. Then $\theta_{(t+1,i)}$ converges w.p.1 to θ^* such that $\hat{Q}^* = \mathcal{E}\theta^*$ is the fixed point of $\mathcal{G}\mathcal{H}$, where $\mathcal{H}$ is defined by 18.*

Since $TD(0)$ is a special case of Q-learning (where the cardinality of $\mathcal{A}$ is one), one obtains a convergence result for TD(0) as a corrollary of the above theorem. This result complements those that were obtained by Van Roy and Tsitsiklis (Tsitsiklis & Van Roy, 1997) and more recently by Tadić for *gradient-based* TD(λ) (Tadić, 2001).

4.4 Interpolative Approximate Asynchronous Value Iteration

Let us consider again the iteration $V_{t+1} = \mathcal{G}TV_t$ studied by (Gordon, 1995), or the equivalent form (Equation 1) in the parameter space. Convergence can be proved as the composite operator $\mathcal{G}T$ is a contraction with the same index as that of T.[9] However, this algorithm is not practical if the number of states accessible from x_i (the ith base-point) is very large or infinite. In (Tsitsiklis & Van Roy, 1996) the authors consider the same algorithm was considered for interpolative representations. As another application of Theorem 4.1 the convergence of an extension of the second algorithm consider in (Tsitsiklis & Van Roy, 1996) is derived here.

There are a number of ways to approximate $\mathcal{P}T\mathcal{E}\theta_t$. An obvious approximation is to replace T by a Monte-Carlo approximation of it. If T is "randomized" in this way then one must change θ_t graudally to filter out the effects of noise:

$$\theta_{t+1} = (1 - \alpha_t)\theta_t + \alpha_t \mathcal{P}\hat{\mathcal{T}}_t\mathcal{E}\theta_t.$$

One possibility is to sample K transitions $\{(X_{t,k}^{(i,a)}, R_{t,k}^{(i,a)}\}_{k=1,\ldots,K}$ from each basis point x_i and action a and define $(\mathcal{P}(\hat{\mathcal{T}}_t V))_i = (1/N) \min_{a\in A} \sum_{k=1}^{N} \{R_{t,k}^{i,a} + \gamma V(X_{t,k}^{(i,a)})\}$. As before, the algorithm can be put in the form $V_{t+1} = \mathcal{T}_t(V_t, V_t)$ and analyzed using the tools developed above. Under

[9] At this level of generality one does not need the interpolative requirement.

the usual conditions on α_t, the samples and $\mathcal{E}$, one obtains convergence to the fixed point of $\mathcal{G}T$ w.p.1.

If one does not have a sampling device that can generate samples from any point of interest then one may use the idea of spatial smoothening and rely on a single sample path generated using a persistently exciting policy. Again, the resulting algorithm can be proven to converge under conditions similar to that of in Theorem 4.3.

Many other variants can be proposed. Due to the lack of space we cannot discuss these here.

4.5 Adaptive Representations

Deciding for a good representation for a given MDP requires good insight of the problem to be solved.Therefore it is important to consider methods that refine the representation on the basis of observed data. The iteration $V_{t+1} = \mathcal{G}_t \mathcal{T}_t(V_t, V_t)$ models such changing representations provided that we allow the operator $\mathcal{G}_t$ to become random (since it will depend on random data). Examples that can be cast into the present framework include refinements of triangulations as in (Munos, 1997), decision tree growing (Wang & Dietterich, 1999) and growing kernel-based methods (Smart & Kaelbling, 2000).

If there exist a deterministic bound on the number of changes of the representation (e.g. adaptive discretization is stopped when a given maximal spatial resolution is achieved or when a maximum number of allowable changes is achieved) then V_t will converge to $\hat{V}^*$, the fixed point of $\mathcal{G}_\infty T$, where $\mathcal{G}_\infty = \lim_{t\to\infty} G_t$.[10] Note that in general G_∞ will be random, but the main theorem continues to hold.

4.6 Relaxing the interpolation requirement

The requirement that $\mathcal{G}$ should be a non-expansion can be relaxed if one observes that all what matters is that $\mathcal{G}T$ should be a contraction. For this it is sufficient to require that $\mathcal{G}$ is Lipschitz with a constant $0 < \gamma' \leq 1/\gamma$ (or $\|\mathcal{E}V - \mathcal{E}U\| \leq \gamma' \|U - V\|$, where $U, V \in B(\mathcal{X})$ are arbitrary). This was first observed in (Tsitsiklis & Van Roy, 1996). The results derived here can also be shown to hold under this relaxed condition.

5. Conclusion

We have introduced a class of general interpolative reinforcement learning algorithms and studied their convergence. In particular, an extension of Q-learning was studied in uncountable state spaces. Almost sure convergence was derived and bounds were given to the loss resulting from using the given architecture.

One attractive feature of Q-learning is that it allows the exploration policy to be chosen at will as long as it remains "infinitely exploring" (see (Singh et al., 1997)). Unfortunately, the result of the previous section requires that the exploration policy be fixed during learning. A naive approach to overcome this problem is to multiply $\alpha_t(z,a)s(z,a,X_t)$ in the update equation with a correction factor C_t, where C_t is obtained recursively: $C_{t+1} = C_t(\pi(A_{t+1}|X_{t+1})/\pi_t(A_{t+1}|X_{t+1})$. Here π is a fixed exploration policy and π_t is the policy that is actually followed during learning. Trivial computation shows that for any real-valued, bounded measurable function f, $E[f(X_t)C_t] = \int f(x)d\mu_X(x)$, i.e., C_t successfully compensates for the changes in the policy. Unfortunately, there is a catch: C_t in general cannot be kept bounded and uniformly positive and thus the previous convergence results will not continue to hold. Therefore decoupling exploration and learning remains a very interesting open problem.

If there is no way to decouple learning from the exploration policy then one may hope to use to use approximate policy iteration together with the considered methods. Then the rate of convergence of the method becomes very important.

Finally, in connection to adaptive interpolation methods, some interesting problems are as follows: (i) Derive conditions under which (specific) algorithms let $\mathcal{G}_t$ converge to some deterministic $\mathcal{G}_t$. (ii) Extend the analysis to the case of indefinitely growing representations, derive consistency results.

[10] Convergence is defined w.r.t. the induced sup-norm of operators.

References

Atkeson, C., Moore, A., & Schaal, S. (1997). Locally weighted learning for control. *Artificial Intelligence Review*, *11*, 75–113.

Baird, L. (1995). Residual algorithms: Reinforcement learning with function approximation. *Proceedings of the Twelfth International Conference on Machine Learning* (pp. 30–37). San Francisco, CA: Morgan Kaufmann.

Baird, L., & Moore, A. (1999). Gradient descent for general reinforcement learning. *Advances in Neural Information Processing Systems 11.*

Baxter, J., Tridgell, A., & Weaver, L. (1998). Knightcap: A chess program that learns by combining TD(λ) with game-tree search. *Proceedings of the*

International Conference on Machine Learning (pp. 28–36).

Bellman, R., & Dreyfus, S. (1959). Functional approximation and dynamic programming. *Math. Tables and other Aids Comp.*, *13*, 247–251.

Boyan, J., & Moore, A. (1995). Generalization in reinforcement learning: Safely approximating the value function. *Advances in Neural Information Processing Systems* (pp. 369–376). The MIT Press.

Chapman, D., & Kaelbling, L. (1991). Input generalization in delayed reinforcement learning: An algorithm and performance comparasions. *Proceedings of the International Joint Conference on Artificial Intelligence.*

Gordon, G. J. (1995). Stable function approximation in dynamic programming. *Proceedings of the Twelfth International Conference on Machine Learning* (pp. 261–268). San Francisco, CA: Morgan Kaufmann.

Littman, M., & Szepesvári, C. (1996). A Generalized Reinforcement Learning Model: Convergence and applications. *Int. Conf. on Machine Learning* (pp. 310–318).

Moore, A. W., & Atkeson, C. G. (1995). The partigame algorithm for variable resolution reinforcement learning in multidimensional state spaces. *Machine Learning*, *21*, 199–233.

Munos, R. (1997). Finite-element methods with local triangulation refinement for continuous reinforcement learning problems. In M. Someren and G. Widmer (Eds.), *Machine learning: Ecml'97 (9th european conf. on machine learning, proceedings)*, vol. 1224 of *Lecture Notes in Artificial Intelligence*, 170–183. Springer, Berlin.

Reetz, D. (1977). Approximate solutions of a discounted Markovian decision problem. *Bonner Mathematischer Schriften*, *98: Dynamische Optimierungen*, 77–92.

Ribeiro, C., & Szepesvári, C. (1996). Q-learning combined with spreading: Convergence and results. *Proceedings of ISRF-IEE International Conference: Intelligent and Cognitive Systems, Neural Networks Symposium* (pp. 32–36). Tehran, Iran.

Samuel, A. (1959). Some studies in machine learning using the game of checkers. *IBM Journal on Research and Development*, 210–229. Reprinted in *Computers and Thought*, E.A. Feigenbaum and J. Feldman, editors, McGraw-Hill, New York, 1963.

Schweitzer, P., & Seidman, A. (1985). Generalized polynomial approximations in Markovian decision processes. *J. of Math. Anal. and Appl.*, *110*, 568–582.

Singh, S., Jaakkola, T., & Jordan, M. (1995). Reinforcement learning with soft state aggregation. *Proceedings of Neural Information Processing Systems.*

Singh, S., Jaakkola, T., Littman, M., & Szepesvári, C. (1997). On the convergence of single-step on-policy reinforcement-learning algorithms. *Machine Learning.* accepted.

Smart, W., & Kaelbling, L. (2000). Practical reinforcement learning in continuous spaces. *Proceedings of the Seventeenth International Conference on Machine Learning.*

Sutton, R. S. (1988). Learning to predict by the method of temporal differences. *Machine Learning*, *3.*

Szepesvári, C., & Littman, M. (1999). A unified analysis of value-function-based reinforcement-learning algorithms. *Neural Computation*, *11*, 2017–2059.

Tadić, V. (2001). On the convergence of temporal-difference learning with linear function approximation. *Machine Learning*, *42*, 241–267.

Tesauro, G. (1994). TD-Gammon, a self-teaching backgammon program achieves master-level play. *Neural Computation*, *6*, 215–219.

Tsitsiklis, J., & Van Roy, B. (1997). An analysis of temporal difference learning with function approximation. *IEEE Transactions on Automatic Control*, *42*, 674–690.

Tsitsiklis, J. N., & Van Roy, B. (1996). Feature-based methods for large scale dynamic programming. *Machine Learning*, *22*, 59–94.

Wang, X., & Dietterich, T. (1999). Efficient value function approximation using regression trees. *Proceedings of the IJCAI Workshop on Statistical Machine Learning for Large-Scale Optimization.* Stockholm, Sweden.

Werbös, P. (1977). Advanced forecasting methods for global crisis warning and models of intelligence. *General Systems Yearbook*, *22.*

Zhang, W., & Dietterich, T. G. (1995). A reinforcement learning approach to job-shop scheduling. *Proceedings of the International Joint Conference on Artificial Intellience.*