Generalized Markov Decision Processes: Dynamic-programming and Reinforcement-learning Algorithms

Csaba Szepesvári
Bolyai Institute of Mathematics
"József Attila" University of Szeged
Szeged 6720 / Aradi vrt tere 1.
HUNGARY
szepes@math.u-szeged.hu

Michael L. Littman
Department of Computer Science
Brown University
Providence, RI 02912-1910
USA
mlittman@cs.brown.edu

November 25, 1997

Abstract

The problem of maximizing the expected total discounted reward in a completely observable Markovian environment, i.e., a Markov decision process (MDP), models a particular class of sequential decision problems. Algorithms have been developed for making optimal decisions in MDPs given either an MDP specification or the opportunity to interact with the MDP over time. Recently, other sequential decision-making problems have been studied prompting the development of new algorithms and analyses. We describe a new generalized model that subsumes MDPs as well as many of the recent variations. We prove some basic results concerning this model and develop generalizations of value iteration, policy iteration, model-based reinforcement-learning, and Q-learning that can be used to make optimal decisions in the generalized model under various assumptions. Applications of the theory to particular models are described, including risk-averse MDPs, exploration-sensitive MDPs, sarsa, Q-learning with spreading, two-player games, and approximate max picking via sampling. Central to the results are the contraction property of the value operator and a stochastic-approximation theorem that reduces asynchronous convergence to synchronous convergence.

1 INTRODUCTION

One particularly well-studied sequential decision-making problem is that of a single agent maximizing expected discounted total reward in a finite-state, completely observable environment. A discount parameter $0 \leq \gamma < 1$ controls the degree to which future rewards are significant compared to immediate rewards.

The theory of Markov decision processes can be used as a theoretical foundation for important results concerning this decision-making problem [2]. A (finite) Markov decision process (MDP) [31] is defined by the tuple $\langle X, A, P, R \rangle$, where X represents a finite set of states, A a finite set of actions, P a transition function, and R a reward function. The optimal behavior for an agent depends on the optimality criterion; in an MDP with the infinite-horizon discounted criterion, the optimal behavior can be found by identifying the optimal value function, defined recursively by

$$V^*(x) = \max_a \left(R(x,a) + \gamma \sum_y P(x,a,y) V^*(y) \right),$$

for all states $x \in X$, where $R(x,a)$ is the immediate reward for taking action a from state x, $0 \leq \gamma < 1$ is a discount factor, and $P(x,a,y)$ is the probability that state y is reached from state x when action $a \in A$ is chosen. These simultaneous equations, known as the *Bellman equations*, can be solved using a variety of techniques ranging from successive approximation [3] to linear programming [11].

In the absence of complete information regarding the transition and reward functions, reinforcement-learning methods can be used to find optimal value functions. Both model-free (direct) methods, such as Q-learning [59, 60], and model-based (indirect) methods, such as prioritized sweeping [29] and DYNA [46], have been explored and many have been shown to converge to optimal value functions under the proper conditions [60, 53, 19, 13].

Not all sequential decision-making problems of interest can be modeled as MDPs; in one form of two-player game, for example, one or the other player chooses an action in each state with one player striving to maximize the total reward and the other trying to minimize it. A great deal of reinforcement-learning research has been directed to solving games of this kind [51, 52, 37, 7], Algorithms for solving MDPs and their convergence proofs do not apply directly to these problems.

There are deep similarities between MDPs and games; for example, it is possible to define a set of Bellman equations for the optimal minimax value of a two-player zero-sum game,

$$V^*(x) = \begin{cases} \max_{a \in A} \left(R(x,a) + \gamma \sum_y P(x,a,y) V^*(y) \right), & \text{if maximizer moves in } x \\ \min_{a \in A} \left(R(x,a) + \gamma \sum_x P(x,a,y) V^*(y) \right), & \text{if minimizer moves in } x, \end{cases}$$

where $R(x,a)$ is the reward to the maximizing player. When $0 \leq \gamma < 1$, these equations have a unique solution and can be solved by successive-approximation methods [39]. In addition, we show in Section 4.1 that the natural extension of several reinforcement-learning algorithms for solving MDPs converge to optimal value functions in two-player games.

In this paper, we introduce a generalized Markov decision process model with applications to reinforcement learning, and list some of the important results concerning the model.

Generalized MDPs provide a foundation for decision making in MDPs and games, as well as in risk-sensitive models [15], exploration-sensitive models [20, 36], simultaneous-action games [39], and other models. The common feature of these decision problems is that the reward function is based on the total, discounted cost—this latter property enables us to apply arguments based on the properties of contraction mappings, which makes the analysis tractable. Our main theorem addresses conditions on the convergence of asynchronous stochastic processes and shows how this problem can be reduced to determining the convergence of a corresponding synchronous one; it can be used to prove the convergence of model-free and model-based reinforcement-learning algorithms in a variety of different sequential decision-making models.

In the remainder of this section, we present the generalized MDP model and motivate it using two detailed examples; in Section 2, we present several algorithms for solving generalized MDPs that are extensions of classic algorithms for solving MDPs; in Section 3, we describe our main theorem and how it can be used for solving generalized MDPs in a reinforcement-learning setting; and in Section 4 we show several applications of our framework to other sequential decision-making problems. Most of the proofs are deferred to the appendix to increase the readability. We tried to make the appendix as self-contained as we could; however, at some places it is necessary to read the main body of the text before reading the appendix.

1.1 MARKOV DECISION PROCESSES

To provide a point of departure for our generalization of Markov decision processes, we begin by describing some results concerning MDPs. These results are well established; proofs of the unattributed claims can be found in Puterman's MDP book [31].

The ultimate target of a decision-making algorithm is to find an optimal policy. A *policy* is some function that tells the agent which actions should be chosen under which circumstances. Of course, since the agent that applies the policy is not clairvoyant, the action prescribed by a policy cannot depend on future states or actions, i.e., a policy maps the history of a process to an action. A policy π is *optimal* under the expected discounted total reward criterion if, with respect to the space of all possible policies, π maximizes the expected discounted total reward from all states.

Maximizing over the space of all possible policies is practically infeasible. However, MDPs have an important property that makes it unnecessary to consider such a broad space of possibilities. We say a policy π is *stationary* and *deterministic* if it maps the actual state directly to an action, ignoring everything else from the history of the decision process, and we write $\pi(x)$ as the action chosen by π when the current state is x. In expected discounted total reward MDP environments, there is always a stationary deterministic policy that is optimal; we will therefore use the word "policy" to mean stationary deterministic policy, unless otherwise stated.

The value function for a policy π, V^π, maps states to their expected discounted total reward under policy π. It can be defined by the simultaneous equations

$$V^\pi(x) = R(x,a) + \gamma \sum_y P(x,a,y)V^\pi(y).$$

It is also possible to condition the immediate rewards on the state y as well; this is somewhat more general, but complicates the presentation. The optimal value function V^* is the value function of an optimal policy; it is unique for $0 \leq \gamma < 1$. The *myopic policy* with respect to a value function V is the policy π_V such that

$$\pi_V(x) = \operatorname*{argmax}_a \left(R(x,a) + \gamma \sum_y P(x,a,y) V(y) \right).$$

Any myopic policy with respect to the optimal value function is optimal.

The Bellman equations can be operationalized in the form of the dynamic-programming operator T, which maps value functions to value functions:

$$[TV](x) = \max_a \left(R(x,a) + \gamma \sum_y P(x,a,y) V(y) \right).$$

For $0 \leq \gamma < 1$, successive applications of T to a value function bring it closer and closer to the optimal value function V^*, which is the unique fixed point of T: $V^* = TV^*$. The algorithm derived from successive applications of T is known as *value iteration*.

In reinforcement-learning applications, R and P are not known in advance. They can be learned from experience by keeping statistics on the expected reward for each state-action pair, and the proportion of transitions to each next state for each state-action pair. In model-based reinforcement learning, R and P are estimated on-line, and the value function is updated according to the approximate dynamic-programming operator derived from these estimates; this algorithm converges to the optimal value function under a wide variety of choices of the order states are updated [13].

The method of Q-learning [59] uses experience to estimate the optimal value function without ever explicitly approximating R and P. The algorithm estimates the optimal Q function

$$Q^*(x,a) = R(x,a) + \gamma \sum_y P(x,a,y) V^*(y),$$

from which the optimal value function can be computed via $V^*(x) = \max_a Q^*(x,a)$. Given an agent's experience at step t $\langle x_t, a_t, y_t, r_t \rangle$ and the current estimate $Q_t(x,a)$ of the optimal Q function, Q-learning updates

$$Q_{t+1}(x_t,a_t) := (1 - \alpha_t(x_t,a_t)) Q_t(x_t,a_t) + \alpha_t(x_t,a_t)(r_t + \gamma \max_a Q_t(y_t,a)),$$

where $0 \leq \alpha_t(x,a) \leq 1$ is a learning rate that controls how quickly new estimates are blended into old estimates as a function of the state-action pair and the trial number. Q-learning converges to the optimal Q function under the proper conditions [60, 53, 19].

1.2 ALTERNATING MARKOV GAMES

In alternating Markov games, two players take turns issuing actions to try to maximize their own expected discounted total reward. We now describe this model to show how closely it parallels MDPs. The model is defined by the tuple $\langle X_1, X_2, A, B, P, R \rangle$, where X_1 is the set of states in which player 1 issues actions from the set A, X_2 is the set of states in which player 2 issues actions from the set B, P is the transition function, and R is the reward function for player 1. Note that it is not assumed that player 1's actions always follow player 2's actions and vice versa. In the zero-sum games we consider, the rewards to player 2 (the minimizer) are simply the additive inverse of the rewards for player 1 (the maximizer). Markov decision processes are a special case of alternating Markov games in which $X_2 = \emptyset$; Condon [9] proves this and the other unattributed results in this section.

A common optimality criterion for alternating Markov games is discounted minimax optimality. Under this criterion, the maximizer should choose actions so as to maximize its reward in the event that the minimizer chooses the best possible counter-policy. An equivalent definition is for the minimizer to choose actions to minimize its reward against the maximizer with the best possible counter-policy. A pair of policies is said to be in *equilibrium* if neither player has any incentive to change policies if the other player's policy remains fixed. The value function for a pair of equilibrium policies is the optimal value function for the game; it is unique when $0 \leq \gamma < 1$, and can be found by successive approximation. For both players, there is always a deterministic stationary optimal policy. Any myopic policy with respect to the optimal value function is optimal, and any pair of optimal policies is in equilibrium.

Dynamic-programming operators, Bellman equations, and solution algorithms can be defined for alternating Markov games by starting with the definitions used in MDPs and changing the maximum operators to either maximums or minimums conditioned on the state. In Section 4.1, we show that the resulting algorithms share their convergence properties with the analogous algorithms for MDPs. A key difference between MDPs and alternating Markov games is that the former can be solved (i.e., an optimal policy can be found) in polynomial time using linear programming; no such algorithm is known for solving alternating Markov games [10][1].

1.3 GENERALIZED MDPS

In alternating Markov games and MDPs, optimal behavior can be identified by solving the Bellman equations; any myopic policy with respect to the optimal value function is optimal. In this section, we generalize the Bellman equations to define optimal behavior for a broad class of reinforcement-learning models. The objective criterion used in these models is additive in that the value of a policy is some measure of the *total* reward received.

[1]An algorithm that solves MDPs is *strongly polynomial* if the number of arithmetical operations needed by the algorithm is polynomial in $|X|$ and $|A|$. At this time, there is no known algorithm that solves MDPs in strongly polynomial time. Using linear programming, however, MDPs can be solved in a number of operations polynomial in $|X|$, $|A|$ and b, where b measures the number of bits needed to write down the transition, rewards, and discount factor. Value iteration converges in time bounded above by a polynomial in $|X|, |A|, b$, and $1/(1-\gamma)$—this algorithm is called pseudo-polynomial because of the appearance of the factor $h = 1/(1-\gamma)$. Value iteration takes a number of iterations proportional to $h \log(h)$ in the worst case. The worst-case time complexity of policy iteration is not known, although it requires no more iterations than value iteration. Value iteration and also policy iteration can be used to solve alternating Markov games in pseudo-polynomial time. For further information on these topics the interested reader is referred to the PhD Thesis of Michael Littman [26].

As a first step towards a general model, we will express the Bellman equations for MDPs and alternating Markov games in a unified notation. For succinctness, we will use an operator-based notation in which addition and scalar multiplication are generalized in a natural way. For example, if $V : X \to \Re$ is a value function and $R : X \times A \to \Re$ or $R : X \times A \times X \to \Re$ (allowing reward to depend on the resulting state as well) is the reward function, we define $(R + \gamma V) : X \times A \times X \to \Re$ to be:

$$(R + \gamma V)(x, a, y) = R(x, a, y) + \gamma V(y),$$

for $x \in X$, $a \in A$, and $y \in Y$.

If we define the operator $\bigoplus : (X \times A \times X \to \Re) \to (X \times A \to \Re)$ to be an expectation operator according to the transition function P,

$$(\bigoplus (R + \gamma V))(x, a) = \sum_y P(x, a, y)(R(x, a, y) + \gamma V(y)),$$

and $\bigotimes : (X \times A \to \Re) \to (X \to \Re)$ to maximize over A[2],

$$(\bigotimes \bigoplus (R + \gamma V))(x) = \max_a \sum_y P(x, a, y)(R(x, a, y) + \gamma V(y)),$$

then $V^* = \bigotimes \bigoplus (R + \gamma V^*)$ is simply an alternate form of the Bellman equations for MDPs. Here, the big plus ($\bigoplus$) is intended to remind us that the operator is a weighted sum and the big x ($\bigotimes$) reminds us that the operator is a type of ma$\underline{\text{x}}$imization.

The $\bigotimes$ operator here takes a Q function (mapping states and actions to values) and returns the value of the best action in each state. Now, by changing the meaning of $\bigotimes$ to be

$$(\bigotimes Q)(x) = \begin{cases} \max_a Q(x, a), & \text{if } x \in X_1, \\ \min_b Q(x, b), & \text{if } x \in X_2, \end{cases}$$

$V^* = \bigotimes \bigoplus (R + \gamma V^*)$ is also a representation of the Bellman equations for alternating Markov games.

Our generalized MDP model is defined by the generalized Bellman equations $V^* = \bigotimes \bigoplus (R + \gamma V^*)$, where different models are obtained using different definitions for the $\bigotimes$ and $\bigoplus$ operators. As we've seen, optimal value functions for MDPs and alternating Markov games can be expressed in this way; a variety of other examples are discussed in Section 4.

The value functions defined by the generalized MDP model can be interpreted as the total value of the rewards received by an agent selecting actions in a non-deterministic environment. The agent begins in state x, takes action a, and ends up in state y. The $\bigoplus$ operator defines how the value of the next state should be used in assigning value to the current state. The $\bigotimes$ operator defines how an optimal agent should choose actions.

When $0 \leq \gamma < 1$ and $\bigotimes$ and $\bigoplus$ are non-expansions, the generalized Bellman equations have a unique optimal solution, and therefore, the optimal value function is well defined (see Appendix A). Recall that an operator T which maps a normed space B_1 to another normed space B_2 is a non-expansion if for all $f, g \in B_1$ $\|Tf - Tg\| \leq \|f - g\|$, where $\|\cdot\|$ denotes

[2]The definitions of $\bigoplus$ and $\bigotimes$ are $(\bigoplus S)(x, a) = \sum_y P(x, a, y) S(x, a, y)$ and $(\bigotimes Q)(x) = \max_a Q(x, a)$, where $S : (X \times A \times Y) \to \Re$ and $Q : (X \times A) \to \Re$.

model/example reference	$(\bigotimes f)(x)$	$(\bigoplus g)(x,a)$
disc. exp. MDPs [60]	$\max_a f(x,a)$	$\sum_y P(x,a,y)g(x,a,y)$
exp. return of π [45]	$\sum_a \pi(x,a)f(x,a)$	$\sum_y P(x,a,y)g(x,a,y)$
alt. Markov games [7]	$\max_a$ or $\min_b f(x,b)$	$\sum_y P(x,a,y)g(x,a,y)$
risk-sensitive MDPs [15]	$\max_a f(x,a)$	$\min_{y:P(x,a,y)>0} g(x,a,y)$
expl.-sens. MDPs [20]	$\max_{\pi\in P_0} \sum_a \pi(x,a)f(x,a)$	$\sum_y P(x,a,y)g(x,a,y)$
Markov games [24]	$\max_A \min_b \sum_a A(a)f(x,(a,b))$	$\sum_y P(x,(a,b),y)g(x,(a,b),y)$

Table 1: Some models and their specification as generalized Markov decision processes.

the norm on the appropriate spaces[3]. In accordance with the above definition we say that the $\bigotimes$ operator is a non-expansion if

$$\left\|\bigotimes f - \bigotimes g\right\| \le \|f-g\|$$

for all $f,g : X \times A \to \Re$, and $x \in X$, here $\|\cdot\|$ denotes the max norm over the appropriate function space. An analogous condition defines when $\bigoplus$ is a non-expansion.

Many natural operators are non-expansions, such as max, min, midpoint, median, mean, and fixed weighted averages of these operations (see Appendix B). Mode and Boltzmann-weighted averages are not non-expansions (see Littman's thesis [26] for information on Boltzmann-weighted averages). Several previously described sequential decision-making models are special cases of this generalized MDP model—Table 1 gives a brief sampling. For more information about the specific models listed, see the associated references.

As with MDPs, we can define a dynamic-programming operator

$$TV = \bigotimes \bigoplus (R+\gamma V) \tag{1}$$

such that, for $0 \le \gamma < 1$, the optimal value function V^* is the unique fixed point of T. The operator T is a contraction mapping as long as $\gamma < 1$. Recall that an operator T is a contraction mapping if $\|TV_1 - TV_2\| \le \gamma\|V_1 - V_2\|$ where V_1 and V_2 are arbitrary functions and $0 \le \gamma < 1$ is the index of contraction. Here, $\|\cdot\|$ is the max norm. It is easy to see that T is a contraction mapping using the non-expansion properties of $\bigotimes$ and $\bigoplus$.

We can define a notion of stationary *myopic policies* with respect to a value function V; it is any (stochastic) policy π for which $T^\pi V = TV$ where

$$[T^\pi V](x) = \sum_a \pi(x,a)\left(\left(\bigoplus(R+\gamma V)\right)(x,a)\right).$$

[3]If $B_1 = B_2$ are spaces of function over X, i.e., if $B_1 = B_2 = (X \to \Re)$ then we say that $T : B_1 \to B_2$ is a pointwise non-expansion if for all $x \in X$ and $f,g \in B_1$ there holds that $|(Tf)(x) - (Tg)(x)| \le |f(x) - g(x)|$. Taking the maximum of both sides over X we see that if T is a pointwise non-expansion then T is also a non-expansion for the max norm. It is easy to extend the notion of being a pointwise non-expansion to spaces when $B_1 = (X \times A \to \Re)$ and $B_2 = (X \to \Re)$. We say that the operator $T : B_1 \to B_2$ is a pointwise non-expansion over X if $|(Tf)(x) - (Tg)(x)| \le \|f(x,\cdot) - g(x,\cdot)\|$ for all $f,g \in B_1$ and $x \in X$. Here, $f(x,\cdot)$ and $g(x,\cdot)$ are understood as functions from $A \to \Re$, and $\|\cdot\|$ denotes a norm over $A \to \Re$, usually the max norm. Usually it is much easier to check if an operator is a pointwise non-expansion. The operators we consider are, indeed, pointwise non-expansions. Our statements, however, do not exploit this feature.

Here, $\pi(x, a)$ represents the probability that an agent following π would choose action a in state x. For later use, it is convenient to introduce the operator $\otimes^\pi$ defined by $(\otimes^\pi f)(x) = \sum_a \pi(x, a) f(x, a)$. To be certain that every value function possesses a myopic policy, we require that the operator $\otimes$ satisfy the following property: for all functions $f : X \times A \to \Re$ and states x,

$$\min_a f(x, a) \leq (\bigotimes f)(x) \leq \max_a f(x, a). \tag{2}$$

In an alternate formulation, Inequality (2) is replaced by the restriction that $(\otimes f)(x) = f(x, a_f)$ for all f, where $a_f \in A$ is an action that may depend on f. In other words, $\otimes$ must select an action. This has the price that stochastic actions must be explicitly introduced (the action set of the new model would be the set of probability distributions over A, $\Pi(A)$), but has the advantage that "deterministic" policies suffice (since each stochastic policies in the present model would have a corresponding deterministic action). To put this another way, Inequality (2) is just an extension of the selection condition to stochastic actions.

The *value function* with respect to a policy π, V^π is defined by the simultaneous equations $V^\pi = T^\pi V^\pi$; it is unique.

We say a policy π is *optimal* if it is myopic with respect to its own value function. A better term for such policies might be "self-consistent"; we use the optimization-oriented term "optimal" because the most common applications make use of a $\otimes$ operator that selects extremals. Even in non-optimization settings, it is reasonable to call these policies optimal since they share important properties of optimal policies of MDPs. The first such property is that the evaluation of optimal policies is a particular function: the fixed point of T. To see this, let π^* be an optimal policy. Then V^{π^*} is the fixed point of T because $V^{\pi^*} = T^{\pi^*} V^{\pi^*} = T V^{\pi^*}$. Thus, $V^{\pi^*} = V^*$ when $\gamma < 1$, because T has a unique fixed point by the Banach fixed-point theorem [44]. All the statements of this section and some other basic facts about generalized MDPs are proved in Appendices A through D.

1.4 SOLVING GENERALIZED MDPS

The previous subsections have motivated and described our generalization of Markov decision processes. We showed how MDPs and alternating Markov games, two popular models of sequential decision making, could be viewed as examples of the generalized model. In Section 4, we will examine other examples including games in which players make their action choices simultaneously, MDPs with a risk-sensitive performance criterion, MDPs with an exploration-sensitive performance criterion, and the use of sampling to replace the computation of the maximum action in MDPs.

Formulating a problem as an instance of a formal model is rarely an end unto itself, however. We address algorithms for *solving* generalized MDPs; that is, we would like to identify optimal policies for specific instances of the model. One class of algorithms assumes access to a complete description of the model instance. Examples of dynamic-programming algorithms in this class are described in Section 2.

A second class of algorithms assumes that the only information available to the agent on the specific problem instance being solved is via "experience": state-to-state transitions with their associated rewards. Problems couched this way are known as *reinforcement-learning* *problems* and algorithms for solving them are called *reinforcement-learning algorithms*[4]. Section 3 describes reinforcement-learning algorithms for generalized MDPs.

Because of the asynchronous manner in which information arrives in a reinforcement-learning problem, the contraction assumption (that is, that $\gamma < 1$) becomes critical for smoothing across pieces of information that arrive separately. We derive a powerful theorem concerning the convergence of asynchronous learning processes that depends on little other than the assumption of contraction; this makes it applicable to a wide variety of models. Full generality of the theorem is achieved by stating the results in terms of general normed spaces. The theorem will be presented and discussed in Section 3.1.

The next section extends the standard value-iteration and policy-iteration algorithms to the generalized model. Section 3 describes a general theorem that can be used to prove the convergence of several reinforcement-learning algorithms in the generalized MDP framework.

2 SOLVING GENERALIZED MDPS VIA A MODEL

The most basic algorithms for solving MDPs are value iteration [3] and policy iteration [18]; both date back the late 1950s. This section describes how these algorithms can be applied to solve generalized Markov decision processes.

2.1 VALUE ITERATION

The method of *value iteration*, or successive approximations [3, 39], is a way of iteratively computing arbitrarily good approximations to the optimal value function V^*.

A single step of the process starts with an estimate V_{t-1} of the optimal value function, and produces a better estimate $V_t = TV_{t-1}$. We show that applying T repeatedly causes the value function to become as close as desired to optimal. Again, the notation $\|\cdot\|$ refers to the maximum norm.

Lemma 1 *Let V_t be the value function produced in the tth iteration of value iteration. After t steps of value iteration on a generalized* MDP, $\|V_t - V^*\| \leq \gamma^t \|V_0 - V^*\|$.

Proof: We proceed by induction. The base case, $\|V_0 - V^*\| \leq \gamma^0 \|V_0 - V^*\|$, is self evident. By the inductive hypothesis we see

$$\|V_t - V^*\| = \|TV_{t-1} - TV^*\| \leq \gamma \|V_{t-1} - V^*\| \leq \gamma\gamma^{t-1} \|V_0 - V^*\| = \gamma^t \|V_0 - V^*\|.$$

Q.E.D.

Since $0 \leq \gamma < 1$, we have that $V_t \to V^*$ at a geometric rate as t increases. In some circumstances, it is helpful to state this result without reference to the details of the initial value function V_0. Let $M = \sup_x \max_a |R(x,a)| = \|R\|$. If the agent received a reward of M

[4]Traditionally, it was the field of adaptive control that considered such "learning" problems [23]. Adaptive-control researchers, however, usually considered linear models only, i.e., when the evolution equation of the controlled object is linear. Nonetheless, the results and emerged problems of adaptive control can be instructive for reinforcement-learning researchers.

on every step, its total expected reward would be $\sum_{i=0}^{\infty} \gamma^i M = M/(1-\gamma)$. Thus, the zero value function, $V_0 = 0$ cannot differ from the optimal value function by more than $M/(1-\gamma)$ at any state. This also implies that the value function for any policy cannot differ from the optimal value function by more than $2M/(1-\gamma)$ at any state. This allows us to restate Lemma 1 in a form that bounds the number of iterations needed to find an ϵ-optimal value function.

Theorem 1 *Let V_0 be any value function such that $\|V_0\| \leq M/(1-\gamma)$, and let*

$$t^* = \left\lceil \frac{1 + \log(M) + \log(\frac{1}{\epsilon}) + \log(\frac{1}{1-\gamma})}{\log(\frac{1}{\gamma})} \right\rceil .$$

Running value iteration for t^* or more steps results in a value function V such that $\|V - V^*\| \leq \epsilon$.

Proof: This follows from simple algebraic manipulation of the bounds given in this section. Q.E.D.

2.2 COMPUTING NEAR-OPTIMAL POLICIES

Thus, we know that the value function estimates converge to the optimal value function. But when should we stop this iteration? The following result shows that if $\|V_{t+1} - V_t\|$ is small then $\|V_t - V^*\|$ is small, too. The next question is which policy to use after we have stopped the iteration. The natural choice is the myopic policy with respect to the latest estimate of the value function. Below we show that the value of such a policy is also close to the optimum, i.e., it can be bounded as a function of $\|V_{t+1} - V_t\| = \|TV_t - V_t\|$. More generally, we will show that for an arbitrary function V, the distance between the value function for any myopic policy with respect to V and the optimal value function can be bounded as a function of the *Bellman error magnitude* of V, defined as $\|TV - V\|$. These results rely entirely on the contraction property of the involved generalized MDPs. Note that one must be careful when applying these estimates in practice since the meaning of "small error" is highly problem dependent[5].

First, we establish a few basic results.

Lemma 2 *Let V be a value function, V^π be the value function for the myopic policy with respect to V, and V^* be the optimal value function. Let ϵ be the Bellman error magnitude for V, $\epsilon = \|V - TV\|$. Then, $\|V - V^\pi\| \leq \epsilon/(1-\gamma)$ and $\|V - V^*\| \leq \epsilon/(1-\gamma)$.*

Proof: This result follows easily from the contraction property of T and the triangle inequality.

First, $\|V - V^\pi\| \leq \|V - TV\| + \|TV - V^\pi\| = \|V - TV\| + \|T^\pi V - T^\pi V^\pi\| \leq \epsilon + \gamma\|V - V^\pi\|$. Grouping like terms gives $\|V - V^\pi\| \leq \epsilon/(1-\gamma)$.

[5]McDonald and Hingston [28] pointed out that optimal values can be exponentially small in the number of states for special classes of MDPs.

Similarly, $\|V-V^*\| \leq \|V-TV\|+\|TV-V^*\| = \|V-TV\|+\|TV-TV^*\| \leq \epsilon+\gamma\|V-V^*\|$. Grouping like terms gives $\|V - V^*\| \leq \epsilon/(1-\gamma)$. Q.E.D.

We next bound the distance between V^π and V^* in terms of ϵ, the Bellman error magnitude (related arguments have been made before [6, 42, 62, 16][6]).

Theorem 2 *Let V be a value function, V^π be the value function for the myopic policy with respect to V, and V^* be the optimal value function. Let ϵ be the Bellman error magnitude for V, $\epsilon = \|V - TV\|$. Then, $\|TV - V^\pi\| \leq \epsilon\gamma/(1-\gamma)$, $\|TV - V^*\| \leq \epsilon\gamma/(1-\gamma)$, and $\|V^\pi - V^*\| \leq 2\epsilon\gamma/(1-\gamma)$.*

Proof: The third statement follows from an application of the triangle inequality to the first two statements, which we prove now. First,

$$\|TV - V^\pi\| \;=\; \|T^\pi V - T^\pi V^\pi\| \leq \gamma\|V - V^\pi\| \leq \epsilon\gamma/(1-\gamma).$$

Similarly,

$$\|TV - V^*\| \;=\; \|TV - TV^*\| \leq \gamma\|V - V^*\| \leq \epsilon\gamma/(1-\gamma),$$

completing the proof. Q.E.D.

This result is concerned with *values* and not immediate rewards, so the total reward earned by a myopic policy is not too far from optimal. The significance of the result is that a value-iteration algorithm that stops when the Bellman error magnitude is less than or equal to $\epsilon \geq 0$ will produce a good policy with respect to ϵ.

This result can be refined further for a subclass of generalized MDPs. In generalized MDPs in which there is a finite set of policies such that every value function has a myopic policy in that set, any myopic policy with respect to V_t is optimal for large enough t. This is in no way related to the contraction property of the value iteration operator, i.e., it holds for arbitrary monotone and continuous operators [48]. This means that value iteration can be used to find optimal value functions in finite time for generalized MDPs in this subclass. A further refinement, which relies on the contraction property of the dynamic-programming operator, puts a pseudo-polynomial bound on the number of iterations required to find an optimal policy [26]. This requires that γ, P and R are expressed with a polynomial number of bits.

2.3 POLICY ITERATION

In this section, we define a generalized version of policy iteration. Applied to MDPs, it is equivalent to Howard's policy-iteration algorithm [18] and applied to alternating Markov games, it is equivalent to Hoffman and Karp's policy-iteration algorithm [17]. Policy iteration for MDPs proceeds as follows: Choose an initial policy π_0 and evaluate it. Let the next policy, π_1, be the greedy policy with respect to the value function V^{π_0}. Continue in this way until $\pi_{t+1} = \pi_t$. The traditional proof of convergence relies on the following facts [18]:

[6]The most general of these arguments is due to Bertsekas and Shreve [6] (Proposition 4.5) for extremization problems (although, the authors do not exploit this property). They also consider value iteration when the precision of computation is limited. Williams and Baird [62] have proved these bounds tight for MDPs, and this should hold for generalized MDPs, as well.

(i) $V^{\pi_{t+1}} \geq V^{\pi_t}$, and the inequality is strict for at least one state if π_t is not optimal,

(ii) there are a finite number of policies (since X and A are finite), and

(iii) the fixed point of T is unique.

Unlike value iteration, the convergence of policy iteration seems to require that value is maximized (or minimized in a cost-based setting) with respect to some set of possible actions (this is because we require Condition (i) in the above paragraph). To capture this, we will restrict our attention to generalized MDPs in which $\bigotimes$ can be written

$$[\bigotimes Q](x) = \max_{\rho \in \mathcal{R}} \left([\bigotimes^{\rho} Q](x) \right) \tag{3}$$

where $\mathcal{R}$ is a compact set and $\bigotimes^{\rho}$ is a non-expansion operator mapping functions over $X \times A$ to functions over X for all $\rho \in \mathcal{R}$. Note that the conditions that $\mathcal{R}$ be compact and $\bigotimes^{\rho}$ be a non-expansion for all ρ ensure that the maximum in the above equation is well defined. Note that any operator $\bigotimes$ can be written this way by defining $\mathcal{R} = \{\rho_0\}$ and $\bigotimes^{\rho_0} = \bigotimes$; the choice of parameterization ultimately determines the efficiency of the resulting policy-iteration algorithm. A generalized MDP satisfying Equation (3) and satisfying a monotonicity property discussed in Appendix C is called a *maximizing* generalized MDP.

As a concrete example, MDPs can be viewed as type of maximizing generalized MDP. Let $\mathcal{R} = A$ and $[\bigotimes^{\rho} Q](x) = Q(x, \rho)$. Then, $[\bigotimes Q](x) = \max_{a \in A} Q(x, a) = \max_{\rho \in \mathcal{R}} Q(x, \rho) = \max_{\rho \in \mathcal{R}} [\bigotimes^{\rho} Q](x)$ as required by Equation (3). Similarly, alternating Markov games can be viewed as maximizing generalized MDPs. Again, $\mathcal{R} = A$ and define

$$[\bigotimes^{\rho} Q](x) = \begin{cases} Q(x, \rho), & \text{if } x \in X_1, \\ \min_b Q(x, b), & \text{if } x \in X_2. \end{cases}$$

The maximization and minimization operators have been separated so that they can be treated independently. To understand the importance of this, note that the essence of policy iteration is that in every step the new policy is an *improvement* over the previous policy. For alternating Markov games this would mean that $V^{\pi_{t+1}}(x) \geq V^{\pi_t}(x)$ for $x \in X_1$ and $V^{\pi_{t+1}}(x) \leq V^{\pi_t}(x)$ for $x \in X_2$. However, as a careful analysis of an example by Condon [10] shows, the additive structure of rewards is incompatible with this condition. To be able to work with additive rewards, we need to separate the minimumization and maximumization operators.

A more complex example is Markov games, in which $\mathcal{R}$ is not finite; it will be described in Section 4.2.

The term *ρ-myopic policy* refers to a mapping $\omega : X \to \mathcal{R}$ such that

$$[\bigotimes^{\omega(x)} Q](x) = \max_{\rho \in \mathcal{R}} \left([\bigotimes^{\rho} Q](x) \right) = [\bigotimes Q](x),$$

for all $x \in X$. Here ω is myopic with respect to Q. If $Q = \bigoplus(R + \gamma V)$ then the policy which is myopic for Q is called myopic for V as well. The value function for a ρ-myopic policy ω, V^{ω}, is defined as the optimal value function for the generalized MDP where $\bigotimes^{\omega(x)}$ is used as the summary operator in state x; it is well defined.

If the condition $\min_a Q(x,a) \leq [\bigotimes^\rho Q](x) \leq \max_a Q(x,a)$ is satisfied for all $Q : X \times A \to \Re$ and $x \in X$ (the last inequality is automatically satisfied), then to every function $\omega : X \to \mathcal{R}$ we could assign one (or more) policy $\pi_\omega : X \to \Pi(A)$ with the property that $[\bigotimes^{\omega(x)} Q](x) = \sum_a \pi_\omega(x,a)Q(x,a)$. Then, every mapping ω can be identified with an equivalent stochastic stationary policy. This definition is in harmony with the definitions of the value functions V^ω and V^π, and the definition of greediness.

We characterize policy iteration as follows. Start with a value function V and compute its ρ-myopic policy ω and ω's value function V^ω. If $\|V - V^\omega\| \leq \epsilon$, terminate with V^ω as an approximation of the optimal value function. Otherwise, start over, after assigning $V := V^\omega$.

Note that if $\mathcal{R}$ contains a single element, then this policy-iteration algorithm terminates after two steps since $\omega^{\rho_0} = \omega$ and thus $V^{\omega_0} = V^*$ with $\omega_0(x) = \rho_0$ for all $x \in X$. This illustrates the tradeoff between determining the optimal value function of a given mapping ω and determining the optimal value function. The following two examples are also instructive. We can apply the generalized policy-iteration algorithm to MDPs by taking $\mathcal{R}$ to be the set of actions and $\bigotimes^\rho$ to return $Q(s, rho)$. Because computing V^ω is equivalent to evaluating a fixed policy and can be solved by, e.g., Gaussian elimination, the resulting policy-iteration algorithm (which is just standard policy iteration) is useful. In alternating Markov games, we take $\bigotimes^\rho$ to return $Q(s,\rho)$ for states in which value is maximized, and to pick out the minimum value $\min_a Q(s,a)$ otherwise. Computing V^ω is equivalent to solving an MDP, which is conceptually easier than finding V^* directly.

To show that policy iteration converges, we appeal to two important results. The first is that, for maximizing generalized MDPs,

$$V^*(x) = \max_{\omega : X \to \mathcal{R}} V^\omega(x),$$

meaning that the optimal value function dominates or equals the value functions for all possible values of ω. The second is a generalization of a result of Puterman [31] that shows that the iterates of policy iteration are bounded below by the iterates of value iteration. ¿From these two facts, we can conclude that policy iteration converges to the optimal value function, and furthermore, that its convergence is at least as fast as the convergence of value iteration. This result can also be proved for continuous and monotone value-iteration operators of maximizing type without assuming the contraction property [48].

Theorem 3 *Let*

$$V^*(x) = \max_{\rho \in \mathcal{R}} \left(\left(\bigotimes^\rho \bigoplus (R + \gamma V^*) \right)(x) \right)$$

and, for all $\omega : X \to \mathcal{R}$,

$$V^\omega(x) = \left(\bigotimes^{\omega(x)} \bigoplus (R + \gamma V^\omega) \right)(x)$$

where $\bigotimes^\rho$ and $\bigoplus$ are non-expansions and monotonic and $\mathcal{R}$ is compact. Then, for all $x \in X$,

$$V^*(x) = \max_{\omega : X \to \mathcal{R}} V^\omega(x).$$

Proof: This result is proven in Appendix D. Q.E.D.

Lemma 3 *Let U_t be the iterates of value iteration and V_t be the iterates of policy iteration, starting from the same initial value function, $U_0 = V_0$. If U_0 and V_0 are underestimates of the optimal value function, then for all t and $x \in X$, $U_t(x) \leq V_t(x) \leq V^*(x)$.*

Proof: The proof is in Appendix D. Q.E.D.

According to Lemma 3, policy iteration converges at least geometrically and a bound on convergence time can be given by

$$t^* = \left\lceil \frac{\log\left(\|V_0 - V^*\|/(\epsilon(1-\gamma))\right)}{\log(1/\gamma)} \right\rceil$$

or in terms of the Bellman error magnitude of V_0:

$$t^* = \left\lceil \frac{\log\left(\|V_0 - TV_0\|/(\epsilon(1-\gamma)^2)\right)}{\log(1/\gamma)} \right\rceil .$$

If $t \geq t^*$ then $\|V_t - V^*\| \leq \epsilon$ ($\epsilon \geq 0$).

It is worth noting that the implementation of policy evaluation in generalized MDPs depends on the definition of $\bigoplus$. When the expected-reward objective is used, as it is in MDPs, policy evaluation can be implemented using a linear-equation solver. When $\bigoplus$ is maximization or minimization, as it is in some games or under a risk-sensitive criterion, policy evaluation is equivalent to solving an MDP and can be accomplished using linear programming (or policy iteration!).

With a little change, the above framework is also capable of expressing asynchronous policy-iteration algorithms. Most of the previous results on asynchronous policy iteration can be repeated since those proofs depend only on the monotonicity and the contraction properties of the involved operators [61, 40]. The work of Bertsekas and Shreve [6] is also worth mentioning here: they have considered a version of policy iteration in which both myopic policies and the evaluation of these policies are determined with a precision geometrically increasing in time. Such an approximate policy-iteration scheme is useful of the state or the action spaces are infinite (such as a compact subset of a Euclidean space).

3 REINFORCEMENT-LEARNING ALGORITHMS

In this section, we describe methods for solving MDPs that make use of "experience" instead of direct access to the parameters of the model. We begin by introducing a powerful stochastic-approximation theorem.

3.1 COMPUTING THE FIXED POINT OF A CONTRACTION BY MEANS OF RANDOMIZED APPROXIMATIONS

Iterative approaches to finding an optimal value function can be viewed in the following general way. At any moment in time, there is a set of values representing the current approximation of the optimal value function. On each iteration, we apply some dynamic-programming operator, perhaps modified by experience, to the current approximation to generate a new approximation. Over time, we would like the approximation to tend toward the optimal value function.

In this process, there are two types of approximation going on simultaneously. The first is an approximation of the dynamic-programming operator for the underlying model, and the second is the use of the approximate dynamic-programming operator to find the optimal value function. Both Q-learning and model-based reinforcement learning work in this way. This section presents a theorem that gives a set of conditions under which this type of simultaneous stochastic approximation converges to an optimal value function.

First, we need to define the general stochastic process. Let the set X be the states of the model, and the set $\mathbf{B}(X)$ of bounded, real-valued functions over X be the set of value functions. Let $T : \mathbf{B}(X) \to \mathbf{B}(X)$ be an arbitrary contraction mapping and V^* be the fixed point of T.

To apply the value-iteration algorithm, the contraction mapping T is applied directly to successively approximate V^*. In other algorithms, especially reinforcement-learning algorithms, T is not available and we must use our experience to construct approximations of T. Consider a sequence of random operators $T_t : (\mathbf{B}(X) \times \mathbf{B}(X)) \to \mathbf{B}(X)$ and define $U_{t+1} = T_t(U_t, V)$ where V and $U_0 \in \mathbf{B}(X)$ are arbitrary value functions. We say T_t approximates T at V with probability one uniformly over X, if U_t converges to TV uniformly over X [7]. The basic idea is that T_t is a randomized version of T in some sense; it uses U_t as "memory" to help it approximate TV. Here, one may think of V as a "test function," as in physics[8].

The following theorem shows that, under the proper conditions, we can use the sequence T_t to estimate the fixed point V^* of T.

THEOREM 3.1 *Let T be an arbitrary mapping with fixed point V^*, and let T_t approximate T at V^* with probability one uniformly over X. Let V_0 be an arbitrary value function, and define $V_{t+1} = T_t(V_t, V_t)$. If there exist functions $0 \le F_t(x) \le 1$ and $0 \le G_t(x) \le 1$ satisfying the conditions below with probability one, then V_t converges to V^* with probability one uniformly over X:*

1. *for all U_1, and $U_2 \in \mathbf{B}(X)$ and all $x \in X$,*

$$|T_t(U_1, V^*)(x) - T_t(U_2, V^*)(x)| \le G_t(x)|U_1(x) - U_2(x)|;$$

2. *for all U and $V \in \mathbf{B}(X)$, and all $x \in X$,*

$$|T_t(U, V^*)(x) - T_t(U, V)(x)| \le F_t(x) \sup_{x'} |V^*(x') - V(x')|;$$

[7] A sequence of random functions f_n converges to f^* with probability one uniformly over X if, for almost all events ω for which $f_n(\omega, x) \to f^*$, the convergence is uniform in x. This should be contrasted to uniform almost sure (or probability one) convergence, when we consider a sequence of random functions, f_n, and we require that the speed of convergence of $f_n(\omega) - f(\omega)$ to zero should be independent of ω.

[8] In physics, the effect of electric fields is determined using the concept of "test charges," which are imagined unit-charge, no-mass particles subject to the field. The strength of the field (effect of the operator) is determined as the force acting on the test charge. The situation here is analogous since we have an imagined object subject to a transformation.

3. *for all $k > 0$, $\Pi_{t=k}^{n} G_t(x)$ converges to zero uniformly in x as n increases; and,*

4. *there exists $0 \le \gamma < 1$ such that for all $x \in X$ and large enough t,*

$$F_t(x) \le \gamma(1 - G_t(x)).$$

Proof: To prove this, we will define a sequence of auxiliary functions, U_t, that is guaranteed to converge, and relate the convergence of V_t to the convergence of U_t. Let U_0 be an arbitrary value function and let $U_{t+1} = T_t(U_t, V^*)$. Since T_t approximates T at V^*, U_t converges to $TV^* = V^*$ with probability one uniformly over X. We will show that $\|U_t - V_t\|$ converges to zero with probability one, which implies that V_t converges to V^*. Let

$$\delta_t(x) = |U_t(x) - V_t(x)|$$

and let

$$\Delta_t(x) = |U_t(x) - V^*(x)|.$$

We know that $\Delta_t(x)$ converges to zero because U_t converges to V^*.

By the triangle inequality and the constraints on T_t, we have

$$\begin{aligned}
\delta_{t+1}(x) &= |U_{t+1}(x) - V_{t+1}(x)| \\
&= |T_t(U_t, V^*)(x) - T_t(V_t, V_t)(x)| \\
&\le |T_t(U_t, V^*)(x) - T_t(V_t, V^*)(x)| + |T_t(V_t, V^*)(x) - T_t(V_t, V_t)(x)| \\
&\le G_t(x)|U_t(x) - V_t(x)| + F_t(x)\|V^* - V_t\| \\
&\le G_t(x)\delta_t(x) + F_t(x)\|V^* - V_t\| \\
&\le G_t(x)\delta_t(x) + F_t(x)(\|V^* - U_t\| + \|U_t - V_t\|) \\
&\le G_t(x)\delta_t(x) + F_t(x)(\|\delta_t\| + \|\Delta_t\|) \qquad (4)
\end{aligned}$$

If it were the case that $\|\Delta_t\| = 0$ for all t, then δ_t would converge to zero as shown in Lemma 10 of Section E.1. Using this, one may show that the perturbation caused by Δ_t diminishes. The main difficulty of the proof is that an inequality similar to Inequality (4) does not hold for $\|\delta_t\|$, i.e., different components of δ_t may converge at different speeds and, moreover, because of the disturbance term, $\|\Delta_t\|$, $\|\delta_t\|$ may even increase sometimes. Even more, we do not have an *a priori* estimate of the convergence rate of Δ_t to zero, which would enable a traditional treatment. However, the idea of homogeneous perturbed processes [19] can be used to show that the effect of this perturbation can be neglected.

To use Inequality (4) to show that $\delta_t(x)$ goes to zero with probability one, we use some auxiliary results proven in Appendix E. Q.E.D.

Note that from the conditions of the theorem and the additional condition that T_t approximates T at every function $V \in \mathbf{B}(X)$, it follows that T is a contraction operator at V^* with index of contraction γ (i.e., $\|TV - TV^*\| \le \gamma\|V - V^*|$ for all V)[9]. We next describe some of the intuition behind the statement of the theorem and its conditions.

[9]The proof of this goes as follows. Let $V, U_0, V_0 \in \mathbf{B}(X)$ be arbitrary and let $U_{t+1} = T_t(U_t, V)$ and $V_{t+1} = T_t(V_t, V^*)$. Let $\delta_t(x) = |U_t(x) - V_t(x)|$. Then, using Conditions (1) and (2) of Theorem 3.1 we

The iterative approximation of V^* is performed by computing $V_{t+1} = T_t(V_t, V_t)$, where T_t approximates T with the help of the "memory" present in V_t. Because of Conditions (1) and (2), $G_t(x)$ is the extent to which the estimated value function depends on its present value and $F_t(x) \approx 1 - G_t(x)$ is the extent to which the estimated value function is based on "new" information (this reasoning becomes clearer in the context of the applications in Section 4).

In some applications, such as Q-learning, the contribution of new information needs to decay over time to insure that the process converges. In this case, $G_t(x)$ needs to converge to one. Condition (3) allows $G_t(x)$ to converge to 1 as long as the convergence is slow enough to incorporate sufficient information for the process to converge (this is discussed in some detail in Section 4.8).

Condition (4) links the values of $G_t(x)$ and $F_t(x)$ through some quantity $\gamma < 1$. If it were somehow possible to update the values synchronously over the entire state space, the process would converge to V^* even when $\gamma = 1$ provided that $\prod_{t=1}^{T}(F_t(x) + G_t(x)) \to 0$ uniformly in x as T increases. In the more interesting asynchronous case, when $\gamma = 1$, the long-term behavior of V_t is not immediately clear; it may even be that V_t converges to something other than V^* or it may even diverge depending on how strict Inequality (4) and the inequality of Condition (4) are. If these were strict, then $\|\delta_t\|$ might not decrease at all. The requirement that $\gamma < 1$ insures that the use of outdated information in the asynchronous updates does not cause a problem in convergence.

One of the most noteworthy aspects of this theorem is that it shows how to reduce the problem of approximating V^* to the problem of approximating T at a particular point V (in particular, it is enough that T can be approximated at V^*); in many cases, the latter is much easier to achieve and also to prove. For example, the theorem makes the convergence of Q-learning a consequence of the classical Robbins-Monro theorem [34].

In many problems we do not have full access to the operator $\bigoplus$ or the immediate rewards R [10]. Basically, there are two ways to deal with this: we can build an estimate of $\bigoplus$ and R, or we can estimate a function (without ever building a model of $\bigoplus$ and R) from which an optimal policy is easily be determined. In the next section, we discuss a particular generalized Q-learning algorithm which provides an interesting insight into how Q-learning-like algorithms should be constructed.

3.2 GENERALIZED Q-LEARNING

A defining attribute of the generalized Q-learning algorithm is that we exchange the ordering of the $\bigoplus$ and $\bigotimes$ operators in the update equation relative to the defining generalized Bellman

get that $\delta_{t+1}(x) \le G_t(x)\delta_t(x) + \gamma(1 - G_t(x))\|V - V^*\|$. By Condition (3), $\prod_{t=0}^{\infty} G_t(x) = 0$, and thus, $\limsup_{t\to\infty} \delta_t(x) \le \gamma\|V - V^*\|$ (see, e.g., the proof of Lemma 10 of Section E.1). Since T_t approximates T at V^* and also at V, we have that $U_t \to TV$ and $V_t \to TV^*$ with probability one. Thus, δ_t converges to $\|TV - TV^*\|$ with probability one and thus $\|TV - TV^*\| \le \gamma\|V - V^*\|$ holds with probability one. Since any probability space must contain at least one element, the above inequality, which contains only deterministic variables, is true. Note that if Condition (1) were not restricted to V^*, this argument would imply that T is a contraction with index γ.

[10] It is reasonable to assume complete access to $\bigotimes$, since this determines how the agent should optimize its choice of actions. In Section 4.7, we will discuss cases when this assumption is relaxed.

equations. Remember that the fixed-point equation of V^* is

$$V^* = \bigotimes \bigoplus (R + \gamma V^*).$$

Now, if we let $Q^* = \bigoplus(R + \gamma V^*)$ then we arrive at the fixed point equation

$$Q^* = \bigoplus (R + \gamma \bigotimes Q^*),$$

i.e., Q^* is the fixed point of the operator $K : ((X \times A) \to \Re) \to ((X \times A) \to \Re)$, with $KQ = \bigoplus(R + \gamma \bigotimes Q)$. The reason for exchanging summary operators is that

(i) we have access to $\bigotimes$, not $\bigoplus$, but

(ii) we have a "consistent" method for estimating $\bigoplus$ which we will apply to estimate Q^*.

Once we have a good estimate of Q^*, say Q, a near optimal policy is easily determined: one takes a policy π, which is myopic for Q, i.e., $\bigotimes^\pi Q = \bigotimes Q$. Then, using the technique of Section 2.2, it can be shown that $\|V^\pi - V^*\| \le 2\|Q - Q^*\|/(1-\gamma)$, i.e., we can bound the suboptimality of the resulting policy.

Now, what do we mean by a "consistent" method? Let B_1, B_2 be arbitrary normed spaces and let $T : B_1 \to B_2$ be a mapping. By a method which estimates T, we mean any procedure which to every $f \in X$ assigns a sequence $m_t(f) \in B_2$. The method is said to be consistent with T if, for any f, the sequence $m_t(f)$ converges to Tf with probability one. Usually, we will consider iterative methods of the form $\mathcal{M} = (M_0, M_1, \ldots, M_t, \ldots)$, where $M_t : B_2 \times B_1 \to B_2$ and

$$m_{t+1}(f) = M_t(m_t(f), f), \quad t \ge 0, \tag{5}$$

$m_0(f)$ being arbitrary. The first argument of M_t can be viewed as the internal "memory" of the method.

The most well-known example corresponds to the estimating of averages of functions. Let $B_1 = (X \to \Re)$ for some finite X, $B_2 = \Re$, and let $T : (X \to \Re) \to \Re$ be given by

$$Tf = \sum_{x \in X} \Pr(x) f(x),$$

where $\Pr(x)$ is a probability distribution over X. If x_t is a sequence of identically distributed independent random variables with underlying probability distribution $\Pr(\cdot)$, then the iterative method with

$$M_t(m, f) = (1 - \alpha_t)m + \alpha_t f(x_t),$$

where $\alpha_t \ge 0, \sum_{t=0}^\infty \alpha_t = \infty$ and $\sum_{t=0}^\infty \alpha_t^2 < \infty$ is consistent with T. Indeed, since

$$m_{t+1}(f) = (1 - \alpha_t)m_t(f) + \alpha_t f(x_t),$$

we see that this is the simplest Robbins-Monro process (iterated averaging) and $m_t(f)$ converges to Tf with probability one.

To present the following example, we need to refine the definition of consistent iterative methods. We say that an iterative method $\mathcal{M}$ is consistent with T for the initial set $Y_0(f)$, if for each $f \in B_1$ and $m_0(f) \in Y_0(f)$ the process $m_t(f)$ defined by Equation (5) converges to Tf with probability one. The next example shows why we must restrict the set of initial values—also as a function of f.

Let $B_1 = (X \rightarrow \Re)$ for some finite X, $B_2 = \Re$, and let $T : (X \rightarrow \Re) \rightarrow \Re$ be given by $(Tf) = \min_{y \in X_0} f(y)$. Now, if x_t is a sequence of random variables such that $\{x_1, x_2, \ldots, x_t, \ldots\} = X_0$ with probability one, then the iterative method with

$$M_t(m, f) = \min(m, f(x_t))$$

is consistent with T and the initial set $Y_0(f) = \{y \,|\, y \geq Tf\}$.

Consistency of a method with an n-dimensional operator follows from componentwise convergence of the estimates. That is, let $T = (T_1, T_2, \ldots, T_n) : B \rightarrow B_1 \times B_2 \times \ldots \times B_n$ and let $m_t(f) \in B_1 \times B_2 \times \ldots \times B_n$ be a sequence generated by some method. Then it is clear that $m_t(f)$ converges to Tf if and only if $m_t(f)_i$ converges to $(Tf)_i = T_i f$. From this it follows that if $\mathcal{M}_i$ is an iterative method which is consistent with T_i, then the method $\mathcal{M} : (B_1 \times B_2 \times \ldots \times B_n) \times B \rightarrow B_1 \times B_2 \times \ldots \times B_n$ defined by

$$M_t(m, f)_i = M_{t,i}(m_i, f)$$

will be consistent with T. That is, consistent methods for a multidimensional operator T can be constructed by the composition of one-dimensional methods consistent with T_i, $i = 1, 2, \ldots, n$. This is useful since the $\bigoplus$ operator is usually multidimensional.

How a consistent method of estimating $\bigoplus$ results in a Q-learning algorithm is discussed next. In general, if the iterative method $\mathcal{M} = (M_1, M_2, \ldots, M_t, \ldots)$ with $M_t : (X \times A \times Y \rightarrow \Re) \times (X \times A \rightarrow \Re) \rightarrow (X \times A \rightarrow \Re)$ is consistent with $\bigoplus$ then the appropriate Q-learning algorithm is given by

$$Q_{t+1} = M_t(Q_t, R + \gamma \bigotimes Q_t).$$

Note that such an algorithm would need explicit knowledge of R. To avoid this we introduce a new operator $\mathcal{Q}$ which maps $X \rightarrow \Re$ to $X \times A \rightarrow \Re$ and which is defined by $(\mathcal{Q}V) = \bigoplus(R + \gamma V)$. Now, if M_t is consistent with the operator $\mathcal{Q}$ then the corresponding generalized Q-learning rule takes the form

$$Q_{t+1} = M_t(Q_t, \bigotimes Q_t).$$

All the Q-learning algorithms which we will discuss are of this form.

How is the convergence of such an algorithm ensured? Defining $T_t(Q, Q') = M_t(Q, \otimes Q')$, we arrive at an operator sequence T_t, which, in many cases, satisfies the conditions of Theorem 3.1. It is immediate that T_t approximates K at any Q' since $Q_{t+1} = T_t(Q_t, Q') = M_t(Q_t, \otimes Q')$ converges to $\mathcal{Q} \otimes Q' = R + \gamma \otimes Q' = KQ'$ by assumption. The other conditions on T_t (Condition (1)–(4)) result in completely similar conditions on M_t, which we do not list here since it will be equally convenient to check them directly for T_t. The aim of this discussion was to give an explanation of how Q-learning algorithms are constructed.

4 APPLICATIONS

This section uses Theorem 3.1 to prove the convergence of various decision-making algorithms.

4.1 GENERALIZED Q-LEARNING FOR EXPECTED VALUE MODELS

In this section, we will consider a model-free algorithm for solving the family of *finite* state and action generalized MDPs defined by the Bellman equations $V^* = \bigotimes \bigoplus (R + \gamma V^*)$ where $\bigoplus$ is an expected value operator, $(\bigoplus g)(x,a) = \sum_y P(x,a,y) g(x,a,y)$, and the definition of $\bigotimes$ does not depend on R or P.

A Q-learning algorithm for this class of models can be defined as follows. Given experience $\langle x_t, a_t, y_t, r_t \rangle$ at time t and an estimate $Q_t(x,a)$ of the optimal Q function, let

$$Q_{t+1}(x_t, a_t) := (1 - \alpha_t(x_t,a_t)) Q_t(x_t,a_t) + \alpha_t(x_t,a_t) \left(r_t + \gamma (\bigotimes Q_t)(y_t) \right). \qquad (6)$$

We can derive the assumptions necessary for this learning algorithm to satisfy the conditions of Theorem 3.1 and therefore converge to the optimal Q function. The randomized approximate dynamic-programming operator that gives rise to the Q-learning rule is

$$T_t(Q', Q)(x,a) = \begin{cases} (1 - \alpha_t(x,a)) Q'(x,a) + \alpha_t(x,a)(r_t + \gamma(\bigotimes Q)(y_t)), & \text{if } x = x_t \text{ and } a = a_t \\ Q'(x,a), & \text{otherwise.} \end{cases}$$

If

- y_t is randomly selected according to the probability distribution defined by $P(x_t, a_t, \cdot)$,
- $\bigotimes$ is a non-expansion,
- r_t has a finite variance and expected value given x_t, a_t and y_t equal to $R(x_t, a_t, y_t)$,
- the learning rates are decayed so that $\sum_{t=1}^{\infty} \chi(x_t = x, a_t = a) \alpha_t(x,a) = \infty$ and $\sum_{t=1}^{\infty} \chi(x_t = x, a_t = a) \alpha_t(x,a)^2 < \infty$ uniformly with probability one[11],

then a standard result from the theory of stochastic approximation [34] states that T_t approximates K at Q^* with probability one. That is, this method of using a decayed, exponentially weighted average correctly computes the average one-step reward.

Let

$$G_t(x,a) = \begin{cases} 1 - \alpha_t(x,a), & \text{if } x = x_t \text{ and } a = a_t; \\ 1, & \text{otherwise,} \end{cases}$$

and

$$F_t(x,a) = \begin{cases} \gamma \alpha_t(x,a), & \text{if } x = x_t \text{ and } a = a_t; \\ 0, & \text{otherwise.} \end{cases}$$

These functions satisfy the conditions of Theorem 3.1 (Condition (3) is implied by the restrictions placed on the sequence of learning rates α_t).

Theorem 3.1 therefore implies that this generalized Q-learning algorithm converges to the optimal Q function with probability one uniformly over $X \times A$. The convergence of Q-learning for discounted MDPs and alternating Markov games follows trivially from this. Extensions of this result for a "spreading" learning rule [32] are given in Appendix 4.8.

[11]Here, χ denotes the characteristic function. A common choice for learning rates is $\alpha_t(x,a) = 1/(1 + n_t(x,a))$, where $n_t(x,a)$ is the number of times (x,a) has been visited before t. For this learning-rate function, the condition on learning rates requires that every state-action pair is updated infinitely often. If a central decreasing learning rate, e.g. $\alpha_t(x,a) = 1/t$, is used, then the learning-rate condition additionally requires that the update rate of any given (x,a) pair should not decrease faster than the decrease of learning rates. More results on this can be found in Appendix F.

4.2 Q-LEARNING FOR MARKOV GAMES

Markov games are a generalization of MDPs and alternating Markov games in which both players simultaneously choose actions at each step. The basic model was developed by Shapley [39] and is defined by the tuple $\langle X, A, B, P, R \rangle$ and discount factor γ. As in alternating Markov games, the optimality criterion is one of discounted minimax optimality, but because the players move simultaneously, the Bellman equations take on a more complex form:

$$V^*(x) = \max_{\rho \in \Pi(A)} \min_{b \in B} \sum_{a \in A} \rho(a) \left(R(x,(a,b)) + \gamma \sum_{y \in X} P(x,(a,b),y) V^*(y) \right). \tag{7}$$

In these equations, $R(x,(a,b))$ is the immediate reward for the maximizer for taking action a in state x at the same time the minimizer takes action b, $P(x,(a,b),y)$ is the probability that state y is reached from state x when the maximizer takes action a and the minimizer takes action b, and $\Pi(A)$ represents the set of discrete probability distributions over the set A. The sets X, A, and B are finite.

Once again, optimal policies are policies that are in equilibrium, and there is always a pair of optimal policies that are stationary. Unlike MDPs and alternating Markov games, the optimal policies are sometimes stochastic; there are Markov games in which no deterministic policy is optimal. The stochastic nature of optimal policies explains the need for the optimization over probability distributions in the Bellman equations, and stems from the fact that players must avoid being "second guessed" during action selection. An equivalent set of equations can be written with a stochastic choice for the minimizer, and also with the roles of the maximizer and minimizer reversed.

To clarify the connection between this model and the class of generalized MDPs, define $Q : (X \times (A \times B)) \to \Re$ to be an arbitrary Q function over pairs of simultaneous actions,

$$(\bigotimes Q)(x) = \max_{\rho \in \Pi(A)} \min_{b \in B} \sum_{a \in A} \rho[a] Q(x,(a,b)),$$

and

$$(\bigoplus V)(x,(a,b)) = \sum_{y \in X} P(x,(a,b),y) V(y),$$

then Equation (7) can be expressed in the familiar form $V^* = \otimes \oplus (R + \gamma V^*)$. Note that both $\oplus$ and $\otimes$ defined this way are non-expansions and monotonic (see Appendices B and C).

The Q-learning update rule for Markov games [24] given step t experience $\langle x_t, a_t, b_t, y_t, r_t \rangle$ has the form

$$Q_{t+1}(x_t,(a_t,b_t)) := (1 - \alpha_t(x_t,(a_t,b_t))) Q_t(x_t,(a_t,b_t)) + \alpha_t(x_t,(a_t,b_t)) \left(r_t + \gamma (\bigotimes Q_t)(y_t) \right).$$

This is identical to Equation (6), except that actions are taken to be simultaneous pairs for both players. The results of the previous section prove that this rule converges to the optimal Q function under the proper conditions.

In general, it is necessary to solve a linear program to compute the update given above. We hypothesize that Theorem 3.1 can be combined with the results of Vrieze and Tijs [57] on solving Markov games by "fictitious play" to prove the convergence of a linear-programming-free version of Q-learning for Markov games.

4.3 CONVERGENCE UNDER ERGODIC SAMPLING

In most of the sequential decision problems that arise in practice, the state space is huge. The most sensible way of dealing with this difficulty is to generate compact parametric representations that approximate the Q function. One form of compact representation, as described by Tsitsiklis and Van Roy [55], is based on the use of feature extraction to map the set of states into a much smaller set of feature vectors. By storing a value of the optimal Q-function for each possible feature vector, the number of values that need to be computed and stored can be drastically reduced and, if meaningful features are chosen, there is a chance of obtaining a good approximation of the optimal Q-function. This approach is extended by Singh et al. [43], where the authors consider learning Q-values for "softly aggregated" states, i.e., for any given aggregated state s there is a probability distribution over the states which determines to which extent a given state from X belongs to s (this can also be viewed as fuzzy sets over the state space and is also related to the spreading rule described in Section 4.8). In this section, we describe a lemma which provides general conditions under which the raw generalization of Q-learning for such aggregate models converges.

Assume that the sequence of experience tuples is an arbitrary stochastic process, $\xi_n =< x_n, a_n, y_n, r_n >$, that satisfies the following criterion. For a given state x and action a let $\xi'_n(x,a)$ be the subprocess for which $x_n = x$ and $a_n = a$. Assume that X and A are finite, $r_n \le B$ for some fixed number B and

$$\lim_{K\to\infty} \frac{1}{K} \sum_{n=N}^{N+K} r'_n(x,a) = R(x,a) \tag{8}$$

$$\lim_{K\to\infty} \frac{1}{K} \sum_{n=N}^{N+K} \chi(y'_n(x,a) = y) = P(x,a,y) \tag{9}$$

and both converge to their limit values with a speed that is independent of N. Here $\chi(y'_n(x,a) = y) = 1$ if $y'_n(x,a) = y$ and $\chi(y'_n(x,a) = y) = 0$, otherwise. A real-valued function f that satisfies

$$\lim_{T\to\infty} \frac{1}{T} \int_k^{k+T} f(s)ds = F$$

with a convergence speed independent of k is said to admit the "uniform averaging" property. Thus, we may say that a process ξ_n can be averaged uniformly if the above conditions hold.

Lemma 4 *Q-learning applied to a sequence ξ_n that admits the uniform averaging property converges to the optimal Q-function of the* MDP *determined by rewards R and transition probabilities P given by the averages in Equations (8) and (9).*

Proof: We immediately see that the conditions of Theorem 3.1 are satisfied except that T_t approximates T, the value operator of the MDP given by $\langle X, A, R, P\rangle$. However, this follows from standard stochastic-approximation results. Q.E.D.

The above lemma can be used to show that if the sampling of states and actions comes from a fixed distribution, then an aggregate model will converge. That is, if you have a Q function represented by an m-entry table E and a mapping $G : X \times A$ to T, and you update entry e of E (according to the Q-learning rule) whenever $G(x_t, a_t) = e$, and, you are sampling

x_t and a_t according to some probabilistic laws, then the values in your table will converge. Section 4.8 discusses this in more detail. Lemma 4 is concerned with the case when the x_t states are sampled asymptotically according to a distribution function p^∞ defined over X ($\Pr(x_t = x)$ converges to $p^\infty(x)$).

4.4 RISK-SENSITIVE MODELS

Heger [15] described an optimality criterion for MDPs in which only the *worst* possible value of the next state makes a contribution to the value of a state[12]. An optimal policy under this criterion is one that avoids states for which a bad outcome is possible, even if it is not probable; for this reason, the criterion has a risk-averse quality to it. This can be expressed by changing the expected value operator $\bigoplus$ used in MDPs to

$$(\bigoplus g)(x,a) = \min_{y:P(a,x,y)>0} g(x,a,y).$$

The argument in Section 4.6 shows that model-based reinforcement learning can be used to find optimal policies in risk-sensitive models, as long as $\bigotimes$ does not depend on R or P, and P is estimated in a way that preserves its zero vs. non-zero nature in the limit.

For the model in which $(\bigotimes f)(x) = \max_a f(x,a)$, Heger defined a Q-learning-like algorithm that converges to optimal policies without estimating R and P online [15]. In essence, the learning algorithm uses an update rule analogous to the rule in Q-learning with the additional requirement that the initial Q function be set optimistically; that is, $Q_0(x,a) \geq Q^*(x,a)$ for all x and a [13]. Like Q-learning, this learning algorithm is a generalization of Korf's [22] LRTA* algorithm for stochastic environments. The algorithm and its convergence proof can be found in Appendix G.

4.5 EXPLORATION-SENSITIVE MODELS

A major practical difficulty with Q-learning in MDPs is that the conditions needed to ensure convergence to the optimal Q function and optimal policy make it impossible for a learning agent to ever adopt the optimal policy. In particular, an agent following the optimal policy will not visit every state and take every action infinitely often, and this is necessary to assure that an optimal policy is learned.

John [20, 21] devised an approach to this problem based on the idea that any learning agent must continue to explore forever. Such an agent should still seek out actions that result in high expected discounted total reward, but not to the exclusion of taking exploratory actions. He found that better learning performance can be achieved if the Q-learning rule is changed to incorporate the condition of persistent exploration. More precisely, in some domains, John's learning rule performs better than standard Q-learning when exploration is

[12]Such a criterion was also analyzed by Bertsekas and Shreve [6].

[13]The necessity of this condition is clear since in this Q-learning algorithm we need to estimate the operator $\min_{y:P(x,a,y)>0}$ from the observed transitions, and the underlying iterative method–as discussed in Section 3.2—is consistent only if the initial estimate is overestimating. Since we require only that T_t approximates T at Q^*, it is sufficient if the initial value of the process satisfies $Q_0 \geq Q^*$. Note that $Q_0 = M/(1-\gamma)$ satisfies this condition, where $M = \max_{(x,a,y)} R(x,a,y)$.

retained, i.e., the discounted cumulated reward during learning was higher for his learning rule. However, this does not mean that his learning rule converges to a better estimate of the optimal Q-function: if exploration were stopped at some point late in the run, then it is likely that the myopic policy with respect to the Q-function learned by standard Q-learning would perform better than the myopic policy with respect to the Q-function learned by his rule.

One concrete implementation of this idea is the following *metapolicy*. Given a Q function and a small value $e > 0$, when in state x, take the action $\text{argmax}_a Q(x, a)$ with probability $1 - e$ and a random action from A with probability e. Assuming that an agent will select actions according to this metapolicy (instead of, for example, the greedy metapolicy, which always selects the action with the highest Q value), which is a reasonable Q function to use?

John shows empirically that the optimal Q function for the MDP is not always the best choice here. So instead of using the standard Q-learning update rule, he updates Q values by

$$Q_{t+1}(x_t, a_t) := \\ (1 - \alpha_t(x_t, a_t))Q_t(x_t, a_t) + \alpha_t(x_t, a_t) \left(r_t + \gamma \left(e \frac{1}{|A|} \sum_a Q_t(y_t, a) + (1 - e) \max_a Q_t(y_t, a) \right) \right).$$

This update rule tries to learn the value of the exploring metapolicy instead of the value of the optimal MDP policy.

It is not difficult to apply the arguments of Section 4.1 to this variation of Q-learning to show that the learned Q function converges to Q^* defined by

$$Q^*(x, a) = R(x, a) + \gamma \sum_y P(x, a, y) \left(e \frac{1}{|A|} \sum_a Q^*(y, a) + (1 - e) \max_a Q^*(y, a) \right). \qquad (10)$$

(The operator of the corresponding generalized MDP are given as follows: $\bigotimes$ operator takes the bizarre form $(\bigotimes Q)(x, a) = e(1/|A|) \sum_a Q(y, a) + (1 - e) \max_a Q(y, a)$ which is a non-expansion by the results of Appendix B, and $\bigoplus$ is the usual averaging operator underlying the transition probabilities P). In addition, we can show that using this Q^* in the metapolicy results in the best possible behavior over the space of all policies generated by this metapolicy. The conclusion is that John's learning rule converges to the optimal Q function for this type of exploration-sensitive MDP. These results are discussed in a forthcoming technical note [25].

This update rule was also described by Rummery [35] in the context of variations of the TD(λ) rule. In addition, Rummery explored a related update rule:

$$Q_{t+1}(x_t, a_t) := (1 - \alpha_t(x_t, a_t))Q_t(x_t, a_t) + \alpha_t(x_t, a_t) \left(r_t + \gamma Q_t(y_t, b_t) \right),$$

where b_t is chosen as the action in state y_t stochastically according to the exploring metapolicy. It can be viewed as an action-sampled version of John's update rule. This rule has also been studied by John [21], and under the name "SARSA" by Sutton [47] and Singh and Sutton [41]. Once again, it is possible to apply Theorem 3.1 to show that Q_t converges to Q^* as defined in Equation (10) [25]. (In Section 4.7 we describe a related algorithm in which $\bigotimes$ is estimated by computing randomized maximums.)

4.6 MODEL-BASED LEARNING METHODS

The defining assumption in reinforcement learning is that the reward and transition functions, R and P, are not known in advance. Although Q-learning shows that optimal value functions can be estimated without ever explicitly learning R and P, learning R and P makes more efficient use of experience at the expense of additional storage and computation [29]. The parameters of R and P can be learned from experience by keeping statistics for each state-action pair on the expected reward and the proportion of transitions to each next state. In model-based reinforcement learning, R and P are estimated on-line, and the value function is updated according to the approximate dynamic-programming operator derived from these estimates. Theorem 3.1 implies the convergence of a wide variety of model-based reinforcement-learning methods.

The dynamic-programming operator defining the optimal value for generalized MDPs is given in Equation (1). Here we assume that $\bigoplus$ may depend on P and/or R, but $\bigotimes$ may not. It is possible to extend the following argument to allow $\bigotimes$ to depend on P and R as well. In model-based reinforcement learning, R and P are estimated by the quantities R_t and P_t, and $\bigoplus^t$ is an estimate of the $\bigoplus$ operator defined using R_t and P_t. As long as every state-action pair is visited infinitely often, there are a number of simple methods for computing R_t and P_t that converge to R and P. A bit more care is needed to insure that $\bigoplus^t$ converges to $\bigoplus$, however. For example, in expected-reward models, $(\bigoplus g)(x, a) = \sum_y P(x, a, y)g(x, a, y)$ and the convergence of P_t to P guarantees the convergence of $\bigoplus^t$ to $\bigoplus$. On the other hand, in a risk-sensitive model, $(\bigoplus g)(x, a) = \min_{y:P(x,a,y)>0} g(x, a, y)$ and it is necessary to approximate P in a way that insures that the set of y such that $P_t(x, a, y) > 0$ converges to the set of y such that $P(x, a, y) > 0$. This can be accomplished easily, for example, by setting $P_t(x, a, y) = 0$ if no transition from x to y under a has been observed.

Assuming P and R can be estimated in a way that results in the convergence of $\bigoplus^t$ to $\bigoplus$ and that $\bigotimes^t$ is a non-expansion (more precisely we need that the product of the "expansion index" of $\bigotimes^t$ and γ is smaller than one), the approximate dynamic-programming operator T_t defined by

$$T_t(U, V)(x) = \begin{cases} \bigotimes \bigoplus^t (R_t + \gamma V), & \text{if } x \in \tau_t \\ U(x), & \text{otherwise,} \end{cases}$$

converges to T with probability one uniformly. Here, the set $\tau_t \subseteq X$ represents the set of states whose values are updated on step t; one popular choice is to set $\tau_t = \{x_t\}$.

The functions

$$G_t(x) = \begin{cases} 0, & \text{if } x \in \tau_t; \\ 1, & \text{otherwise,} \end{cases}$$

and

$$F_t(x) = \begin{cases} \gamma, & \text{if } x \in \tau_t; \\ 0, & \text{otherwise,} \end{cases}$$

satisfy the conditions of Theorem 3.1 as long as each x is in infinitely many τ_t sets (Condition (3)) and the discount factor γ is less than 1 (Condition (4)).

As a consequence of this argument and Theorem 3.1, model-based methods can be used to find optimal policies in MDPs, alternating Markov games, Markov games, risk-sensitive MDPs, and exploration-sensitive MDPs. Also, if $R_t = R$ and $P_t = P$ for all t, this result implies that asynchronous dynamic programming converges to the optimal value function [2, 1].

4.7 SAMPLED MAX

The asynchronous dynamic-programming algorithm uses insights from the reinforcement-learning literature to solve dynamic programming problems more efficiently. At time step $t+1$, the algorithm has an estimate V_t of the optimal value function and is given a state x_t at which to improve its estimate. It executes the update rule

$$V_{t+1}(x_t) = \max_{a \in A} \left(R(x_t, a) + \gamma \sum_y P(x_t, a, y) V_t(y) \right). \tag{11}$$

The state x_t is typically selected by following a likely trajectory through the state space, which helps the algorithm focus its computation on parts of the space that are likely to be important. The convergence of asynchronous dynamic programming to the optimal value function (under the assumption that all states are visited infinitely often) follows from the work of Gullapalli and Barto [13] and the results in this paper.

When the set of actions is extremely large, computing the value of the maximum action in Equation (11) becomes impractical. An alternative that has been suggested is to use the update rule

$$V_{t+1}(x_t) = (1 - \alpha_t(x_t))V_t(x_t) + \alpha_t(x_t) \left(\max_{a \in A_t} \left(R(x_t, a) + \gamma \sum_y P(x_t, a, y) V_t(y) \right) \right), \tag{12}$$

where A_t is a random subset of A and $\alpha_t(x)$ is the learning rate at time t for state x. The idea behind this rule is that, if A_t is big enough, Equation (12) is just like Equation (11) except that estimates from multiple time steps are blended together. Making A_t small compared to A allows the update to made more efficiently, at the expense of being a poor substitute for the true update. For the purposes of the analysis presented here, we assume each A_t is generated independently by some fixed process. We assume the learning rates satisfy the standard properties (square summable but not directly summable).

We can show that the update rule in Equation (12) converges and can express (indirectly) what it converges to. The basic approach is to notice that choosing the maximum action over a random choice of A_t corresponds to a particular probability distribution over ranks, that using this probability distribution directly would result in a convergent rule, and that estimating it indirectly converges as well. Once the proper definitions are made, the analysis mirrors that of Q-learning quite closely. The main difference is that Q-learning is a way to average over possible choices of next state whereas Equation (12) is a way of averaging over possible choices of action.

The first insight we will use is as follows. Consider the effect of selecting a random set A_t and then computing $\text{argmax}_{a \in A_t} f(a)$. It is not hard to see that f induces a probability distribution over the elements of A. Taking this a step further, note that this probability distribution is exactly the same if we replace f with any order-preserving transformation of f. In fact, the method for selecting A_t induces a fixed probability distribution on the rank positions of A: there is some probability (independent of f) that the selected a will result in the largest value of $f(a)$, some probability that it will result in the second largest value of $f(a)$, and so on. Let $\rho(i)$ be the probability that the a with the ith largest value of $f(a)$ is selected; this function can be derived (in principle) from the method for selecting A_t.

(Note that it is possible for ties to exist in the rank ordering. We imagine these are broken arbitrarily.)

An a concrete example, we will quickly derive the ρ function for the case in which all a have probability p of being included in A_t. In this case, the maximum valued action will be included in A_t, and will therefore be selected as the maximum element in A_t, with probability p. This implies that $\rho(1) = p$. The action with the second largest value will be chosen as the max if and only if it is included in A_t while the maximum valued action is not: $\rho(2) = (1 - p)p$. Continuing in this way, we find that $\rho(i) = (1 - p)^{i-1}p$ in general.

Using the concepts introduced in the previous paragraphs, we can see that Equation (12) is equivalent to

$$V_{t+1}(x_t) = (1 - \alpha_t(x_t))V_t(x_t) + \alpha_t(x_t)\left(R(x_t, a_t) + \gamma \sum_y P(x_t, a_t, y)V_t(y)\right), \qquad (13)$$

where action a is selected as a_t with probability $\rho(i)$ and i is the rank position of action a under one-step lookahead on V_t.

Let $I(i, V)$ be the action with the ith largest value as computed by one-step lookahead on V [14]. Define

$$V^*(x) = \sum_{i=1}^{|A|} \rho(i)\left(R(x, I(i, V^*)) + \gamma \sum_y P(x, I(i, V^*), y)V^*(y)\right), \qquad (14)$$

for all x. Because V^* is defined by taking a fixed probability-weighted average of a rank-based selector function, it is a form of generalized MDP (see Appendix B). It follows from this that V^* is well defined (for $\gamma < 1$).

If $\rho(1) = 1$, Equation (14) is precisely the Bellman equations defining the optimal value function for an MDP. In general, any (non-metric) sampling method for estimating the best action will result in a different rank-position probability function ρ.

We next show that the update rule in Equation (12) results in the convergence of V_t to V^* as defined in Equation (14) (i.e., not the optimal value function, in general). To do this, we need to first define a dynamic-programming operator T that captures a value-iteration method for finding V^*. This is a straightforward translation of Equation (14). We next need a sequence of dynamic programming operators T_t that capture the update rule in Equation (12). This is a simple translation of the equivalent Equation (13).

To apply our stochastic-approximation theorem, we next need to define functions F_t and G_t and show that they satisfy a set of conditions. As the necessary definitions are precisely the same as in our proof of the convergence of Q-learning (Section 4.1), we will not repeat them here.

The final step is to show that T_t approximates T at V^*. In other words, we need to show that

$$V_{t+1}(x_t) = (1 - \alpha_t(x_t))V_t(x_t) + \alpha_t(x_t)\left(R(x_t, a_t) + \gamma \sum_y P(x_t, a_t, y)V^*(y)\right) \qquad (15)$$

converges to V^* if a_t is selected according to $\rho(i)$ as described above and every state is visited infinitely often. Equation (15) is a variation of Equation (13) in which V^* is used in place of

[14]For those who have already read Appendix B, we note that $I(i, V) = \text{argord}_a^i(R(x, a) + \gamma \sum_y P(x, a, y)V)$.

V_t for one-step lookahead. Proving the convergence of V_t to V^* under Equation (15) parallels the analogous result for Q-learning.

The ease with which this final condition can be checked follows directly from the fact that we only require that the update rule emulate the true dynamic-programming operator at a fixed value function, namely V^*.

In conclusion, the sampled max update rule, as defined in Equation (12), converges to the value function V^* as defined in Equation (14). Whether V^* is a good approximation of the true value function depends on the sampling method used and the degree to which suboptimal action choices in the underlying MDP result in near optimal values.

4.8 Q-LEARNING WITH SPREADING

Ribeiro [33] argued that the use of available information in Q-learning is inefficient: in each step it is only the actual state and action whose Q-value is reestimated. The training process is local both in space and time. If some *a priori* knowledge of the "smoothness" of the optimal Q-value is available then one can make the updates of Q-learning more efficient by introducing a so-called "spreading mechanism," which updates the Q-values of state-action pairs in the vicinity of the actual state-action pair, as well.

The rule studied by Ribeiro is as follows: let Q_0 be arbitrary and

$$Q_{t+1}(z,a) := (1-\alpha_t(z,a)s(z,a,x_t))Q_t(z,a)+\alpha_t(z,a)s(z,a,x_t)\left(r_t+\gamma\max_a Q_t(y_t,a)\right), \quad (16)$$

where $\alpha_t(z,a)$ is the local learning rate of the state-action pair (z,a) which is 0 if $a \neq a_t$, $s(z,a,x)$ is a fixed "similarity" function satisfying $0 \leq s(z,a,x)$, and $\langle x_t, a_t, y_t, r_t\rangle$ is the experience of the agent at time t.

The difference between the above and the standard Q-learning rule is that here we may allow $\alpha_t(z,a) \neq 0$ even if $x_t \neq z$, i.e., states different from the actual may be updated, too. The similarity function $s(z,a,x)$ weighs the relative strength at which the updates occur. (One could also use a similarity which extends spreading over actions. For simplicity we do not consider this case here.)

Our aim here is to show that under the appropriate conditions this learning rule converges and also we will be able to derive a bound on how far the converged values of this rule are from the optimal Q function of the underlying MDP. These results extend to generalized MDPs when $\max_a$ is replaced by any non-expansion operator $\otimes$.

Theorem 4 *If*

1. *X, A are finite,*
2. *$\Pr(y_t = y|x = x_t, a = a_t) = P(x,a,y)$,*
3. *$E[r_t|x = x_t, a = a_t, y = y_t] = R(x,a,y)$ and $Var[r_t|x_t,a_t,y_t]$ is bounded,*
4. *y_t and r_t are independent,*
5. *the states, x_t, are sampled from a probability distribution $p^\infty \in \Pi(X)$, with $p^\infty(x) > 0$,*

6. *$s(z,a,\cdot) \geq 0$,*

7. *$\alpha_t(z,a) = 0$ if $a \neq a_t$,*

8. *$\alpha_t(z,a)$ is independent of x_t, y_t and r_t,*

9. *$0 \leq \alpha_t(z,a)$, $\sum_{t=0}^{\infty} \alpha_t(z,a)s(z,a,x_t) = \infty$ and $\sum_{t=0}^{\infty} \alpha_t^2(z,a)s^2(z,a,x_t) < \infty$, both hold uniformly with probability one.*

Then Q_t as given by Equation (16) converges to the fixed point of the operator $T : ((X \times A) \to \Re) \to ((X \times A) \to \Re)$,

$$(TQ)(z,a) = \sum_{x \in X} \hat{s}(z,a,x) \sum_{y \in X} P(x,a,y) \left(R(x,a,y) + \gamma \max_b Q(y,b) \right), \tag{17}$$

where

$$\hat{s}(z,a,x) = \frac{s(z,a,x)}{\sum_y s(z,a,y)p^{\infty}(y)}.$$

Proof: Note that by definition T is a contraction with index γ since $\sum_x \hat{s}(z,a,x) = 1$ for all (z,a). We use Theorem 3.1. Let

$$T_t(Q',Q)(z,a) = (1 - \alpha_t(z,a)s(z,a,x_t))Q'(z,a) + \alpha_t(z,a)s(z,a,x_t) \left(r_t + \gamma \max_a Q(y_t,a) \right).$$

It can be checked that T_t approximates T at any fixed function Q. Moreover, T_t satisfies Conditions (1) through (3) of Theorem 3.1 with $G_t(z,a) = 1 - \alpha_t(z,a)s(z,a,x_t)$ and $F_t(z,a) = \gamma\alpha_t(z,a)s(z,a,x_t)$. Q.E.D.

It is interesting and important to ask how close Q_0, the fixed point of T where T is defined by (17), is to the true optimal Q^*. By Theorem 6.2 of Gordon [12] we have that

$$\|Q_0 - Q^*\| \leq \frac{2\epsilon}{1-\gamma},$$

where

$$\epsilon = \inf\{\, \|Q - Q^*\| \mid FQ = Q \,\},$$

where $(FQ)(z,a) = \sum_x \hat{s}(z,a,x)Q(x,a)$. This helps us to define the spreading coefficients $s(z,a,x)$. Namely, let $n > 0$ be fixed and let

$$s(z,a,x) = \begin{cases} 1, & \text{if } i/n \leq Q^*(z,a), Q^*(x,a) < (i+1)/n \text{ for some } i; \\ 0, & \text{otherwise,} \end{cases}$$

then we get immediately that $\epsilon \leq 1/n$. Of course, the problem with this is that we do not know in advance the optimal Q-values. However, the above example gives us a guideline, how to define a "good" spreading function: $s(z,a,x)$ should be small (zero) for states z and x if $Q^*(z,a)$ and $Q^*(x,a)$ differ substantially, otherwise $s(z,a,x)$ should take on larger values. In other words, it is a good idea to define $s(z,a,x)$ as the degree of expected difference between $Q^*(z,a)$ and $Q^*(x,a)$.

Note that the above learning process is closely related to learning on aggregated states (if $X = \cup_i X_i$ is a partition of X let $s(z,a,x) = 1$ if and only if $z, x \in X_i$ for some i, otherwise $s(z,a,x) = 0$) and also to learning using interpolator function approximators. In order to understand this, let us reformulate the model suggested by Gordon [12]. Let us fix a subset of X, say X_0. This is the "sample space", which should be much smaller than X. Let $\mathcal{A} : (X_0 \times A \to \Re) \to (X \times A \to \Re)$ be a "function approximator". The motivation behind this notion is that to each sample $\{(x_1, a, y_1), (x_2, b, y_2), \ldots, (x_n, c, y_n)\}$, where $\{x_1, x_2, \ldots, x_n\} = X_0$, and $a, b, \ldots, c \in A$, $y_i \in \Re$, $\mathcal{A}$ assigns a function defined on $X \times A$. We assume that $\mathcal{A}$ is a non-expansion. Now, consider the process

$$Q_{t+1}(z,a) := (1 - \alpha_t(z,a)s(z,a,x_t))Q_t(z,a) + \alpha_t(z,a)s(z,a,x_t)\left(r_t + \gamma \max_a [\mathcal{A}\mathcal{P}Q_t](y_t,a)\right), \tag{18}$$

where $\mathcal{P}$ projects $(X \times A \to \Re)$ to $(X_0 \times A \to \Re)$, i.e., $[\mathcal{P}Q](x,a) = Q(x,a)$ for all $(x,a) \in X_0 \times A$. As we noted above, Rule (16) works equally well if $\max_a$ is replaced by any non-expansion. In this particular case, this non-expansion is given by $(\bigotimes Q)(x) = \max_a [\mathcal{A}\mathcal{P}Q](x,a)$ (it is a non-expansion, since $\bigotimes$ is a composition of non-expansions). Thus, under the conditions of Theorem 4, this rule converges to the fixed point of the operator

$$(TQ)(z,a) = \sum_{x \in X} \hat{s}(z,a,x) \sum_{y \in X} P(x,a,y)\left(R(x,a,y) + \gamma \bigotimes Q(y,b)\right).$$

Note that in Equation (18) if $z \in X_0$ then the update of (z,a) depends only on the values of $Q_t(x_0, a)$, where $x_0 \in X_0$. This means that it is sufficient to store these values during the update process—all the other values can be reproduced using $\mathcal{A}$. Also, operator T can be restricted to $X_0 \times A$.

If $s(z,a,x) = 0$ for all $z \neq x$ (in this case the "reduced Q-table" is updated only if $x_t \in X_0$, which is somewhat wasteful[15]), then the above argument shows that Q-learning combined with a non-expansive function approximator converges to the fixed point of the underlying contraction, $T : ((X_0 \times A) \to \Re) \to ((X_0 \times A) \to \Re)$, where

$$(TQ)(x,a) = \sum_{y \in X} P(x,a,y)\left(R(x,a,y) + \gamma \max_a [\mathcal{A}Q](y_t,a)\right).$$

Using standard non-expansion and contraction arguments Gordon proves that $\|\mathcal{A}Q_\infty - Q^*\| \leq 2\epsilon/(1-\gamma)$, where $TQ_\infty = Q_\infty$ and $\epsilon = \inf\{\|Q - Q^*\| \mid \mathcal{A}Q = Q\}$. We note that these results rely only on the non-expansion and contraction properties of the involved operators.

The above convergence theorem can be extended to the case when the agent follows a given exploration "metapolicy" (e.g., by using the results from stochastic-approximation theory [4]) which ensures that every state-action pair is visited infinitely often and that there exists a limit probability distribution over the states X. For example, persistently exciting (exploring) policies satisfy these conditions. A stochastic policy $\pi = \pi(x,a)$ is persistently

[15]Gordon also considered briefly the other case, when $s(z,a,x)$ can be non-zero for $z \neq x$, and stated that this is equivalent to introducing hidden states into the derived MDP and concluded, pessimistically, that we then run the risk of divergence. The above argument shows that, under appropriate conditions, this is not the case.

exciting if the Markov chain with state set $S = X \times A$ and given by the transition probabilities $p((x,a),(y,b)) = P(x,a,y)\pi(y,b)$ is strongly ergodic[16]. This means that if the agent uses π then it will visit every state-action pair infinitely often; moreover, given any initial state there exists a limit distribution, $p^\infty(x,a)$, of $\Pr(x = x_t, a = a_t)$ which satisfies $p^\infty(x,a) > 0$ for all (x,a). Since $\Pr(x_t = x, a_t = a) = \Pr(a_t = a | x_t = x) P(x_t = x) = \pi(x,a)\Pr(x_t = x)$, under the above conditions the limiting distribution (let us denote it by $p^\infty(x)$) of $\Pr(x_t = x)$ exists as well, and satisfies

$$p^\infty(x) = p^\infty(x,a)/\pi(x,a),$$

(a is arbitrary!) and thus $p^\infty(x) > 0$ [17].

All this shows that, under a persistently exciting policy, there exists a probability distribution p^∞ over X such that x_t is sampled asymptotically according to p^∞ . As a consequence, we have that the conclusion of Theorem 4 still holds in this case.

Ribeiro and Szepesvári studied the above process when $s(z,a,x)$ is replaced by a time dependent function which is also a function of the actual action, that is, the spreading coefficient of (z,a) at time t is given by $s_t(z,a,x_t,a_t)$ [32]. By using Theorem 3.1 they have shown that if $s_t(z,a,x_t,a_t) - \chi(z = x_t, a = a_t)$ converges to zero no more slowly than does $\alpha_t(z,a)$, and the expected time between two successive visits of all state-action pairs is bounded, then Q_t, as defined by the appropriately modified Equation (16), converges to the true optimal Q function, Q^*.

This algorithm, therefore, can make more efficient use of experience than Q-learning does, and still converge to the same result.

5 CONCLUSIONS

We have presented a general model for analyzing dynamic-programming and reinforcement-learning algorithms and have given examples that show the broad applicability of our results. This section provides some concluding thoughts.

5.1 RELATED WORK

The work presented here is closely related to several previous research efforts. Szepesvári [50, 48] described a generalized reinforcement-learning model that is both more and less general than the present model. His model enables more general value propagation than $\bigoplus(R+\gamma V)$ with $\gamma < 1$ but is restricted to maximization problems, i.e., when $\bigotimes = \max$. He proves that, under mild regularity conditions such as continuity and monotonicity of the value propagation operator, the Bellman optimality equation is satisfied and policy and value iteration are valid algorithms. He also treats non-Markovian policies. The main difficulty of this approach is that one has to prove fixed-point theorems without any contraction assumption and for

[16]Some authors call a policy π persistently exciting if the Markov chain over X with transition probabilities $p(x,y) = \sum_a \pi(x,a)p(x,a,y)$ is strongly ergodic. These two definitions are equivalent only if $\pi(x,a) > 0$ for all (x,a).

[17]Another way to arrive at these probabilities is to consider the Markov chain with states X and transition probabilities $p(x,y) = \sum_a \pi(x,a)P(x,a,y)$.

infinite state and action spaces. His model can be viewed as the continuation of the work of Bertsekas [5] and Bertsekas and Shreve [6], who proved similar statements under different assumptions.

Waldmann [58] developed a highly general model of dynamic-programming problems, with a focus on deriving approximation bounds. Heger [15, 16] extended many of the standard MDP results to cover the risk-sensitive model. Although his work derives many of the important theorems, it does not present these theorems in a generalized way to allow them to be applied to any other models. Verdu and Poor [56] introduced a class of abstract dynamic-programming models that is far more comprehensive than the model discussed here. Their goal, however, was different from ours: they wanted to show that the celebrated "Principle of Optimality" discovered by Bellman relies on the fact that the order of selection of optimal actions and the computation of cumulated rewards can be exchanged as desired: in addition to permitting non-additive operators and value functions with values from any set (not just the real numbers), they showed how, in the context of finite-horizon models, a weaker "commutativity" condition is sufficient for the principle of optimality to hold. For infinite models they have derived some very general results[18] that are too general to be useful in practice.

Jaakkola, Jordan, and Singh [19] and Tsitsiklis [53] developed the connection between stochastic-approximation theory and reinforcement learning in MDPs. Our work is similar in spirit to that of Jaakkola, et al. We believe the form of Theorem 3.1 makes it particularly convenient for proving the convergence of reinforcement-learning algorithms; our theorem reduces the proof of the convergence of an asynchronous process to a simpler proof of convergence of a corresponding synchronized one. This idea enables us to prove the convergence of asynchronous stochastic processes whose underlying synchronous process is not of the Robbins-Monro type (e.g., risk-sensitive MDPs, model-based algorithms, etc.) in a unified way.

5.2 FUTURE WORK

There are many areas of interest in the theory of reinforcement learning that we would like to address in future work. The results in this paper primarily concern reinforcement-learning in contractive models ($\gamma < 1$), and there are important non-contractive reinforcement-learning scenarios, for example, reinforcement learning under an average-reward criterion [38, 27]. Extending Theorem 3.1 to all-policies-proper MDPs should not be too difficult. Actor-critic systems and asynchronous policy iteration would also worth the study. It would be interesting to develop a TD(λ) algorithm [45] for generalized MDPs; this has already been done for MDPs [30] and exploration-sensitive MDPs [35]. Theorem 3.1 is not restricted to finite state spaces, and it might be valuable to prove the convergence of a finite reinforcement-learning algorithm for an infinite state-space model. A proof of convergence for modified policy iteration [31] in generalized MDPs should not be difficult.

[18]Here is an example of their statements translated into our framework. They first show that from their commutativity condition it follows that $T^n V = V_n^*$, where V_n^* is the n-step optimal value function, V is the terminal reward function. Now, the statement which concerns infinite-horizon models goes like this: if V_n^* converges to V^* (their Condition 3 [56]) then $T^n V$ converges to V^*. The problem is that, in practice, it is usually clear that $V_n^* = T^n V$, but it is much harder to show that V_n^* converges to V^* [5, 48].

Another possible direction for future research is to apply to modern ODE (ordinary differential equation) theory of stochastic approximations. If one is given a definite exploration strategy then this theory may yield results about convergence, speed of convergence, finite sample size effects, optimal exploration, limiting distribution of Q-values, etc.

5.3 CONCLUSION

By identifying common elements among several sequential decision-making models, we created a new class of models that generalizes existing models in an interesting way. In the generalized framework, we replicated the established convergence proofs for reinforcement learning in Markov decision processes, and proved new results concerning the convergence of reinforcement-learning algorithms in game environments, under a risk-sensitive assumption, and under an exploration-sensitive assumption. At the heart of our results is a new stochastic-approximation theorem that is easy to apply to new situations.

A OPTIMAL VALUE FUNCTION IS UNIQUE

We consider a generalized MDP defined by $\langle X, A, R, P, \otimes, \oplus \rangle$, where $\otimes$ and $\oplus$ are non-expansions. We use $Q : X \times A \to \Re$ to stand for Q functions and $V : X \to \Re$ to stand for value functions. We define $TV = \otimes \oplus (R + \gamma V)$, $KQ = \oplus(R + \gamma \otimes Q)$, $V^* = TV^*$, and $Q^* = KQ^*$.

It is the non-expansion property of T and K that will be most convenient for proving results about them. Here is the first.

Lemma 5 *The T and K operators are contraction mappings if $\gamma < 1$. In particular, if V_1 and V_2 are value functions and Q_1 and Q_2 are Q functions, $\|TV_1 - TV_2\| \leq \gamma \|V_1 - V_2\|$, and $\|KQ_1 - KQ_2\| \leq \gamma \|Q_1 - Q_2\|$.*

Proof: We address the T operator first. By the definition of T, we have

$$\begin{aligned} \|TV_1 - TV_2\| &= \|\bigotimes \bigoplus (R + \gamma V_1) - \bigotimes \bigoplus (R + \gamma V_2)\| \\ &\leq \|\bigoplus (R + \gamma V_1) - \bigoplus (R + \gamma V_2)\| \\ &\leq \|(R + \gamma V_1) - (R + \gamma V_2)\| \\ &\leq \gamma \|V_1 - V_2\|. \end{aligned}$$

The definition of K give us

$$\begin{aligned} \|KQ_1 - KQ_2\| &= \|\bigotimes \bigoplus (R + \gamma Q_1) - \bigotimes \bigoplus (R + \gamma Q_2)\| \\ &\leq \|\bigoplus (R + \gamma Q_1) - \bigoplus (R + \gamma Q_2)\| \\ &\leq \|(R + \gamma Q_1) - (R + \gamma Q_2)\| = \gamma \|Q_1 - Q_2\|. \end{aligned}$$

Another way to prove this is the following: the composition of a contraction with a non-expansion (in arbitrary order) is a contraction with the same index. Since the mapping $V \mapsto R + \gamma V$ is a contraction with index γ the desired result follows. Q.E.D.

Because the operator T is guaranteed to bring two value functions closer together, and the operator K is guaranteed to bring two Q functions closer together, they are called *contraction mappings*.

Because all the results in this chapter are stated in terms of norms, they apply to any update rule as long as the dynamic-programming operator under consideration is a contraction mapping. (See recent work by Tsitsiklis and van Roy [54] for the use of another important and interesting norm for reinforcement learning.) The fact that the optimal value functions are well defined does not imply that they are meaningful; that is, it may be the case that the optimal value function is not the same as the value function for some appropriately defined optimal policy. The results in this section apply to value functions *defined* by Bellman equations; to relate the Bellman equations to a notion of optimality, it is necessary to put forth arguments such as are given in Puterman's book [31].

Theorem 5 *For any generalized Markov decision process, if $\gamma < 1$ then there is a unique V^*, called the* optimal value function*, such that $V^* = TV^*$; a unique Q^*, called the* optimal Q function*, such that $Q^* = KQ^*$; and an optimal (possibly stochastic) policy, π^*, such that $V^*(x) = \sum_a \pi^*(x,a)Q^*(x,a)$.*

Proof: ¿From Lemma 5, the T and K operators for the generalized MDP are contraction mappings with respect to the max norm. The existence and uniqueness of V^* and Q^* follow directly from the Banach fixed-point theorem [44].

We can define the optimal value function and the optimal Q function in terms of each other:

$$V^* = \bigotimes Q^*, \tag{19}$$

and $Q^* = \bigoplus(R + \gamma V^*)$. These equations can be shown to be valid from the definitions of K and T and the uniqueness of Q^* and V^*.

By Condition (2) of $\otimes$ and Equation (19),

$$\min_a Q^*(x,a) \le V^*(x) \le \max_a Q^*(x,a).$$

Therefore, it is possible to define a stochastic policy π^* such that

$$V^*(x) = \sum_a \pi^*(x,a)Q^*(x,a).$$

Q.E.D.

The use of the word *optimal* is somewhat strange since V^* need not be the largest or smallest value function in any sense; it is simply the fixed point of the dynamic-programming operator T. This terminology comes from the Markov decision process model, where V^* is the largest value function of all policies and is retained for consistency.

B SOME NON-EXPANSION SUMMARY OPERATORS

In this section, we prove several properties associated with functions that summarize sets of values. These summary operators are important for defining generalized Markov decision processes, which involve summaries over the action set A and the set of next states (we need the results presented here when discussing simultaneous Markov games and exploration-sensitive models).

Let I be a finite set and $h : I \to \Re$. We define a *summary operator* $\bigodot$ over I to be a function that maps a real-valued function over I to a real number. The maximum operator $\max_{i \in I} h(i)$ and the minimum operator $\min_{i \in I} h(i)$ are important examples of summary operators.

Let h be a real-valued function over I. We say a summary operator $\bigodot$ is a *conservative non-expansion* if it satisfies two properties: it is conservative

$$\min_{i \in I} h(i) \le \bigodot h \le \max_{i \in I} h(i), \tag{20}$$

and it is a non-expansion

$$\left| \bigodot h - \bigodot h' \right| \le \max_{i \in I} |h(i) - h'(i)|. \tag{21}$$

We will show that the max and min summary operators are both conservative non-expansions, after proving a series of related results.

Usually, in MDPs, we deal with multidimensional operators, i.e., operators of the form $T : I \to \Re^n$. Define $T_i : (I \to \Re) \to \Re$ as $T_i x = (Tx)_i$, i.e., $Tx = (T_1 x, \ldots, T_n x)$; the T_is are the coordinate-wise components of T. Non-expansion operators have the nice property that if they are non-expansions componentwise then they are non-expansions as well. The same is true for the conservativeness of operators. This is our first theorem.

Theorem 6 *Let $T : (I \to \Re) \to \Re^n$ be an arbitrary operator. If T_i is non-expansion/conservative ($i = 1, 2, \ldots, n$) then T is non-expansion/conservative.*

Proof: For brevity let $\|\cdot\|$ denote the max norm, $\|h\| = \max_i |h(i)|$. Let $h, h' \in (I \to \Re)$ be two functions. Then, since T_i is a non-expansion, $|(Th - Th')_i| = |(Th)_i - (Th')_i| = |T_i h - T_i h'| \le \|h - h'\|$. Then, $\|Th - Th'\| = \max_i |(Th - Th')_i| \le \max_i |h - h'| = \|h - h'\|$. That T is conservative follows immediately since $h \le h'$ if and only if $h(i) \le h'(i)$ for all $i \in I$. Q.E.D.

Note that if $|J| = n$ then the set $J \to \Re$ can be identified with $\Re^n$.

Let h and h' be real-valued functions over I. For $i \in I$, let $\bigodot^i$ be the summary operator $\bigodot^i h = h(i)$ ($\bigodot^i$ is the projection operator).

Theorem 7 *The summary operator $\bigodot^i$ is a conservative non-expansion.*

Proof: Condition (20) requires that $\bigodot^i h = h(i)$ lie between $\min_{i' \in I} h(i')$ and $\max_{i' \in I} h(i')$. This holds trivially.

To see that Condition (21) holds, note that $|\bigodot^i h - \bigodot^i h'| = |h(i) - h'(i)| \le \max_{i' \in I} |h(i') - h'(i')|$. Q.E.D.

We next examine a more complicated set of non-expansions. For real-valued function h over I, let $\mathrm{ord}^n h$ be the nth largest value in $\{h(i) | i \in I\}$ ($1 \le n \le |I|$). According to this definition, $\mathrm{ord}^1 h = \max_i h(i)$ and $\mathrm{ord}^{|I|} h = \min_i h(i)$. We will show that the ord^n summary operator is a conservative non-expansion for all $1 \leq n \leq |I|$. To do this, we show that pairing two sets of numbers in their sorted order minimizes the largest pairwise difference between the sets of numbers.

Lemma 6 *Let h_1, h_2, g_1, g_2 be real numbers. If $h_1 \leq h_2$ and $g_1 \leq g_2$ then*

$$\max_{i=1,2} |h_i - g_i| \leq \max_{i \neq j, i,j=1,2} |h_i - g_j|.$$

Proof: Two bounds can be proven separately:

$$\begin{aligned} |h_1 - g_1| &= \max(h_1 - g_1, g_1 - h_1) \\ &\leq \max(h_2 - g_1, g_2 - h_1) \\ &\leq \max_{i \neq j, i,j=1,2} |h_i - g_j|. \end{aligned}$$

and

$$\begin{aligned} |h_2 - g_2| &= \max(h_2 - g_2, g_2 - h_2) \\ &\leq \max(h_2 - g_1, g_2 - h_1) \\ &\leq \max_{i \neq j, i,j=1,2} |h_i - g_j|. \end{aligned}$$

Combining these two inequalities proves the lemma. Q.E.D.

We use Lemma 6 to create a bound involving the ord^n summary operator.

Lemma 7 *Let h_1 and h_2 be real-valued functions over I. Then*

$$\max_{1 \leq n \leq |I|} |\mathrm{ord}^n h_1 - \mathrm{ord}^n h_2| \leq \max_{i \in I} |h_1(i) - h_2(i)|.$$

Proof: Both quantities in the inequality involve taking a maximum over differences between matched pairs of values. This lemma states that, of all possible matchings, pairing values with the same position in a sorted list of values gives the smallest maximum difference.

To prove this, we argue that, from any matching that violates the sorted order we can produce a matching that is "more sorted" without increasing the maximum difference (and perhaps decreasing it). The idea is that we can find a pair of pairs of values that are matched out of order, and swap the matching for that pair. By Lemma 6, the resulting matching has a maximum difference no larger than the previous matching. After generating pairings that are more and more sorted, we eventually reach the totally sorted matching. Since the initial matching was arbitrary, the lemma follows. Q.E.D.

That ord^n is a conservative non-expansion follows easily from Lemma 7.

Theorem 8 *The ord^n operator is a conservative non-expansion for all $1 \leq n \leq |I|$.*

Proof: Condition (20) is satisfied easily since it is always the case that $\text{ord}^n h = h(i)$ for some $i \in I$.

To verify Condition (21), let h_1 and h_2 be real-valued functions over I. It follows from Lemma 7 that

$$\begin{aligned} |\text{ord}^n h_1 - \text{ord}^n h_2| &\leq \max_n |\text{ord}^n_{i\in I} h_1(i) - \text{ord}^n_{i\in I} h_2(i)| \\ &\leq \max_{i\in I} |h_1(i) - h_2(i)|. \end{aligned}$$

Since n was arbitrary, the theorem is proved. Q.E.D.

Theorems 7 and 8 state that two very specific classes of summary operators are conservative non-expansions. The next theorem makes it possible to create more complex conservative non-expansions by blending conservative non-expansions together.

Theorem 9 *If $\odot^1$ and $\odot^2$ are conservative non-expansions, then for any $0 \leq \nu \leq 1$, the summary operator*

$$\bigodot^{(1+2),\nu} h = \nu \bigodot^1 h + (1-\nu) \bigodot^2 h$$

is a conservative non-expansion.

Proof: Once again, Condition (20) is not difficult to verify since the operators are being combined using a convex weighted average.

Condition (21) follows from

$$\begin{aligned} &\left|\bigodot^{(1+2),\nu} h - \bigodot^{(1+2),\nu} h'\right| \\ &= \left|\nu \bigodot^1 h + (1-\nu)\bigodot^2 h - \left(\nu \bigodot^1 h' + (1-\nu)\bigodot^2 h'\right)\right| \\ &= \left|\nu \left(\bigodot^1 h - \bigodot^1 h'\right) + (1-\nu)\left(\bigodot^2 h - \bigodot^2 h'\right)\right| \\ &\leq \nu \left|\bigodot^1 h - \bigodot^1 h'\right| + (1-\nu)\left|\bigodot^2 h - \bigodot^2 h'\right| \\ &\leq \nu \max_{i\in I} |h(i) - h'(i)| + (1-\nu) \max_{i\in I} |h(i) - h'(i)| = \max_{i\in I} |h(i) - h'(i)|. \end{aligned}$$

The proof is easily extended to weighted averages of more than two operators. Q.E.D.

The previous theorem demonstrated one way of making conservative non-expansions out of other conservative non-expansions by averaging. The next theorem shows a more sophisticated method for constructing conservative non-expansions.

If $\odot^1$ is a summary operator over I_1, and $\odot^2$ is a summary operator over I_2, we define the *composition* of $\odot^1$ and $\odot^2$ to be a summary operator over $I_1 \times I_2$,

$$(\bigodot^1 \circ \bigodot^2) h = \bigodot^1 \bigodot^2 h.$$

Theorem 10 *Let $\odot = \odot^1 \circ \odot^2$ for conservative non-expansions $\odot^1$ over I_1 and $\odot^2$ over I_2. Then $\odot$ over $I = I_1 \times I_2$ is a conservative non-expansion.*

Proof: Let h and h' be real-valued functions over I. For Condition (20), we see that

$$\begin{aligned}
\bigodot h &= \left(\bigodot^1 \circ \bigodot^2\right) h \\
&= \bigodot^1 \bigodot^2 h \\
&\le \max_{i_1 \in I_1} (\bigodot^2 h)((i_1, i_2)) \le \max_{i_1 \in I_1} \max_{i_2 \in I_2} h((i_1, i_2)) \le \max_{(i_1,i_2) \in I} h((i_1, i_2)).
\end{aligned}$$

The argument that $\bigodot h \ge \min_{(i_1,i_2)\in I} h((i_1, i_2))$ is similar.

For Condition (21),

$$\begin{aligned}
&\left|\bigodot h - \bigodot h'\right| \\
&= \left|(\bigodot^1 \circ \bigodot^2)h - (\bigodot^1 \circ \bigodot^2)h'\right| \\
&= \left|\bigodot^1 \bigodot^2 h - \bigodot^1 \bigodot^2 h'\right| \\
&\le \max_{i_1 \in I_1} \left|(\bigodot^2 h)((i_1, i_2)) - (\bigodot^2 h')((i_1, i_2))\right| \\
&\le \max_{i_1 \in I_1} \max_{i_2 \in I_2} |h((i_1, i_2)) - h'((i_1, i_2))| = \max_{(i_1,i_2) \in I} |h((i_1, i_2)) - h'((i_1, i_2))|.
\end{aligned}$$

This proves that $\bigodot$ is a conservative non-expansion. Q.E.D.

As a non-trivial application of the preceding theorems, we will show that the minimax summary operator, used in Markov games, is a conservative non-expansion. Let A_1 and A_2 be finite sets. The *minimax* summary operator over $A_1 \times A_2$ is defined as

$$\text{minimax}\, h = \max_{\rho \in \Pi(A_1)} \min_{a_2 \in A_2} \sum_{a_1 \in A_1} \rho[a_1] h((a_1, a_2)).$$

Let $\rho \in \Pi(A_1)$ and let h_1 be a real-valued function over A_1. Define

$$\bigodot_{a_1 \in A_1}^{\rho} h_1(a_1) = \sum_{a_1 \in A_1} \rho[a_1] h_1(a_1);$$

by Theorem 9 and Theorem 7, $\bigodot^\rho$ is a conservative non-expansion. Let h be a real-valued function over $A_1 \times A_2$. By Theorem 8, the minimum operator is a conservative non-expansion. Rewrite

$$\text{minimax}_{(a_1,a_2) \in A_1 \times A_2} h((a_1, a_2)) = \max_{\rho \in \Pi(A_1)} \left(\min \circ \bigodot^\rho\right)_{(a_2,a_1) \in A_2 \times A_1} h((a_1, a_2));$$

minimax is a conservative non-expansion by Theorem 10. The compactness of the set $\Pi(A_1)$ of probability distributions over A_1 ensures that the above operator is well defined.

The class of conservative non-expansions is quite broad. It is tempting to think that any operator that satisfies Condition (20) will be a non-expansion. Boltzmann averaging is an example where this is not the case [26]. It is also easy to construct summary operators that are non-expansions, but not conservative: $\bigodot h = 1 + \max_i h(i)$.

C POLICY ITERATION AND MAXIMIZING MODELS

Appendix B describes a collection of important non-expansion operators based on element selection, ordering, convex combinations, and composition. All of these operators obey an additional monotonicity property as well.

Operator $\bigodot$ is *monotonic* if, for all real-valued functions h and h' over a (finite) set I, $h(i) \geq h'(i)$ for all $i \in I$ implies

$$\bigodot h \geq \bigodot h'.$$

Theorem 11 *The following summary operators are monotonic: $\bigodot^i$ for all $i \in I$,* ord^n *for all $1 \leq n \leq |I|$, $\bigodot^{(1+2),\nu}$ for all $0 \leq \nu \leq 1$ if $\bigodot^1$ and $\bigodot^2$ are monotonic, and $\bigodot^1 \circ \bigodot^2$ if $\bigodot^1$ and $\bigodot^2$ are monotonic. An operator $T : \Re^n \rightarrow \Re^m$ is monotonic if and only if for all $i = 1, 2, \ldots, m$ T_i is a non-expansion. Moreover, if $T : \Re^n \rightarrow \Re^m$ and $S : \Re^m \rightarrow \Re^k$ are monotonic then $ST : \Re^n \rightarrow \Re^k$ is monotonic, too.*

Proof: The monotonicity of $\bigodot^i$, $\bigodot^{(1+2),\nu}$, and $\bigodot^1 \circ \bigodot^2$ follow immediately from their definitions. The monotonicity of ord^n can be proven by considering the effect of increasing $h(i)$ to $h'(i)$ for each $i \in I$, one at a time. A simple case analysis shows that each increase in $h(i)$ cannot decrease the value of ord$^n h$. The rest follows since comparisons are performed componentwise. Q.E.D.

D POLICY-ITERATION CONVERGENCE PROOF

In this section, we develop the necessary results to show that the generalized policy-iteration algorithm of Section 2.3 converges to the optimal value function. We will first prove several simple lemmas that illuminate the fundamental properties of value functions in maximizing generalized MDPs.

First, for maximizing generalized MDPs, a single step of value iteration on a value function associated with a mapping ω results in a value function that is no smaller.

Lemma 8 *For all $\omega : X \rightarrow \mathcal{R}$, $TV^\omega \geq V^\omega$.*

Proof: ¿From Equation (1), the constraints on $\bigotimes$, and the definition of V^ω,

$$\begin{aligned} [TV^\omega](x) &= (\bigotimes \bigoplus (R + \gamma V^\omega))(x) \\ &= \max_{\rho \in \mathcal{R}} (\bigotimes^\rho \bigoplus (R + \gamma V^\omega))(x) \\ &\geq \left(\bigotimes^{\omega(x)} \bigoplus (R + \gamma V^\omega) \right)(x) = V^\omega(x). \end{aligned}$$

Q.E.D.

Let T^ω be the dynamic-programming operator associated with the mapping ω

$$[T^\omega V](x) = \left(\bigotimes^{\omega(x)} \bigoplus (R + \gamma V) \right)(x).$$

The next lemma says that the monotonicity properties of $\bigotimes$ and $\bigoplus$ carry over to T and T^ω.

Lemma 9 *The mappings T and T^ω are monotonic for maximizing generalized* MDP*s.*

Proof: For value functions V and V', we want to show that if $V \geq V'$, then $TV \geq TV'$ and $T^\omega V \geq T^\omega V'$. This follows easily from the definitions and the monotonicity of the operators involved and because composition of operators preserves monotonicity. Q.E.D.

Theorem 3 (of Section 2.3) states that the optimal value function dominates the value functions for all ω. We will now prove this using Lemmas 8 and 9.

¿From Lemma 8, we have that $V^\omega \leq TV^\omega$ for all ω. Combining this with the result of Lemma 9, we have $TV^\omega \leq T(TV^\omega)$. By induction and transitivity, $V^\omega \leq (T)^k V^\omega$ for all integers $k \geq 0$ where $(T)^k$ corresponds to the application of T repeated k times. Because $\lim_{k\to\infty}(T)^k V^\omega = V^*$, it follows that $V^\omega \leq V^*$, proving the first part of Theorem 3, i.e., that $V^* \geq \max_\omega V^\omega$. (This last inequality is easily proved if we assume that $\min_a Q(x,a) \leq [\bigotimes^\rho Q](x) \leq \max_a Q(x,a)$ holds for all Q and $x \in X$. Then, as it was noted in Section 8, to every ρ-policy ω we may assign a stochastic policy π with $\bigotimes^\omega = \bigotimes^\pi$ and thus with $T^\pi = T^\omega$. From this, the desired inequality follows immediately.) That $V^* = \max_\omega V^\omega$, follows from Lemma 3, proved next.

The final result we need relates the convergence of policy iteration to that of value iteration. Let U_t be the iterates of value iteration and V_t be the iterates of policy iteration, starting from the same initial value function. Let $\omega_t : X \to \mathcal{R}$ be the sequence of mappings such that $V_t = V^{\omega_t}$.

Lemma 3 states that, for all t and $x \in X$, $U_t(x) \leq V_t(x) \leq V^*(x)$. We proceed by induction. Clearly $U_0(x) \leq V_0(x)$, because they are defined to be equal. Now, assume that $U_t(x) \leq V_t(x) \leq V^*(x)$. By Lemma 9, $TU_t(x) \leq TV_t(x)$. By definition, $TU_t(x) = U_{t+1}(x)$, by Lemma 8, $V_t \leq TV_t$, and by definition $TV_t = T^{\omega_t}V_t$. Now by an argument similar to the proof of Theorem 3,

$$TV_t = T^{\omega_{t+1}}V_t \leq (T^{\omega_{t+1}})^k V_t \leq V^{\omega_{t+1}} = V_{t+1}.$$

Therefore, $U_{t+1}(x) \leq V_{t+1}(x)$. By Theorem 3, $V_{t+1}(x) = V^{\omega_{t+1}} \leq V^*(x)$, completing the proof of Lemma 3.

Lemma 3 and Lemma 1 (which stated that value iteration converges) together imply the convergence of policy iteration. Lemma 3 also provides a bound on the convergence rate of the algorithm; it is no slower than value iteration, but perhaps faster.

E REDUCTION OF SOME PARALLEL ITERATIVE PROCEDURES TO SIMPLER ONES

Jaakkola et al. [19] proved (Lemma 2) that if

$$\delta_{t+1}(x) \leq G_t(x)\delta_t(x) + F_t(x)\|\delta_t\| \tag{22}$$

and $F_t \leq \gamma(1 - G_t)$ for some $0 \leq \gamma < 1$ and $\lim_{n\to\infty}\prod_{t=k}^n G_t = 0$ with probability one uniformly over X for all $k > 0$ then the process of Equation (22) converges to zero with probability one uniformly over X. To be precise, the conditions of their Lemma are not exactly the same as the above conditions. Particularly, they assume that condition $F_t \leq \gamma(1-G_t)$ holds only in the conditional mean with respect to the history of the process and make some additional assumptions concerning G_t.

We may expect that, for \$$Delta_t \to 0$, a $\delta_t(x)$ which satisfies

$$\delta_{t+1}(x) \leq G_t(x)\delta_t(x) + F_t(x)(\|\delta_t\| + \|\Delta_t\|)$$

(see Equation (4)) still converges to zero since it is just a perturbed version of the process of Equation (22) where the perturbation converges to zero. Indeed, according to Lemma 12 stated and proved below, this process converges to zero with probability one uniformly over X.

Before proving Lemma 12 we prove some additional statements that are required for the proof. The proof of Lemma 12 follows along the same lines as the proof of Lemma 2 of Jaakkola et al. [19]. First, we prove a simplified version of Lemma 12 then a rather technical lemma follows. (It may be considered as an extension to Jaakkola et al.'s Lemma 1 [19].) This lemma is about the convergence of homogeneous processes. We will use this result to show that the perturbation caused by Δ_t can be neglected. Finally, the proof of Lemma 12 follows. We would like to emphasize that the main result of this section is Lemma 10 since *this is the very point in the proofs when we must take into account that different components of $\delta_t(x)$ change independently of each other.*

E.1 THE MAIN CONVERGENCE LEMMA

Now, we prove our version of Jaakkola et al.'s [19] Lemma 2. Note that both our assumptions and our proof are slightly different from theirs.

Lemma 10 *Let Z be an arbitrary set and consider the sequence*

$$x_{t+1}(z) = g_t(z)x_t(z) + f_t(z)\|x_t\|, \tag{23}$$

where $z \in Z$ and $\|x_1\| < C < \infty$ with probability one for some $C > 0$. Assume that for all k $\lim_{n\to\infty} \prod_{t=k}^{n} g_t(z) = 0$ uniformly in z with probability one and $f_t(z) \leq \gamma(1-g_t(z))$ with probability one. Then $\|x_t\|$ converges to 0 with probability one.

Proof: We will prove that for each $\epsilon, \delta > 0$ there exist an index $T = T(\epsilon,\delta) < \infty$ (possibly random[19]) such that

$$\Pr(\sup_{t\geq T} \|x_t\| < \delta) > 1-\epsilon. \tag{24}$$

Fix arbitrary $\epsilon, \delta > 0$ and a sequence of numbers $p_1, \ldots, p_n, \ldots$ satisfying $0 < p_n < 1$ to be chosen later.

[19]Note that in probability textbooks usually T is not allowed to be random. However, the following short train of thoughts justifies that T can be random and almost sure convergence still holds. First, note that $\Pr\left(\sup_{k\leq t}|x_t| \geq \delta\right) \leq \Pr((\sup_{T\leq t}|x_t| \geq \delta, (T > k)) \,\text{or}\, (T > k)) \leq \Pr(\sup_{T\leq t}|x_t| \geq \delta) + \Pr(T > k)$. Now, fix an arbitrary $\delta, \eta > 0$ and let $T_0 = T(\delta, \eta/2)$, and let $k = k(\epsilon, \eta)$ be a natural number, large enough so that $\Pr(T_0 > k) < \eta/2$. Such a number exists since $T_0 < \infty$. Then, $\Pr(\sup_{k\leq t}|x_t| < \delta) \leq \Pr(\sup_{T_0\leq t}|x_t| \geq \delta) + \Pr(T_0 > k) < \eta$ which was the desired result.

We have that

$$\begin{aligned} x_{t+1}(z) &= g_t(z)x_t(z) + f_t(z)\|x_t\| \\ &\leq g_t(z)\|x_t\| + f_t(z)\|x_t\| \\ &\leq (g_t(z) + f_t(z))\|x_t\| \\ &\leq \|x_t\|, \end{aligned}$$

since, by assumption, $g_t(z) + f_t(z) \leq g_t(z) + \gamma(1 - g_t(z)) \leq 1$. Thus, we have that $\|x_{t+1}\| \leq \|x_t\|$ for all t and, particularly, $\|x_t\| \leq C_1 = \|x_1\|$ holds for all t. Consequently, the process

$$y_{t+1}(z) = g_t(z)y_t(z) + \gamma(1 - g_t(z))C_1 \tag{25}$$

with $y_1 = x_1$ estimates the process $\{x_t\}$ from above: $0 \leq x_t \leq y_t$ holds for all t. The process y_t converges to γC_1 with probability one uniformly over Z and, thus,

$$\limsup_{t\to\infty} \|x_t\| \leq \gamma C_1$$

with probability one. Thus, there exist an index, say M_1, for which if $t > M_1$ then $\|x_t\| \leq (1+\gamma)/2\, C_1$ with probability p_1. Assume that up to some index $i \geq 1$ we have found numbers M_i such that when $t > M_i$ then

$$\|x_t\| \leq \left(\frac{1+\gamma}{2}\right)^i C_1 = C_{i+1} \tag{26}$$

holds with probability $p_1 p_2 \dots p_i$. Now let us restrict our attention to those events for which Inequality (26) holds. Then we see that the process

$$\begin{aligned} y_{M_i} &= x_{M_i} \\ y_{t+1}(z) &= g_t(z)y_t(z) + \gamma(1 - g_t(z))C_{i+1},\ t \geq M_i \end{aligned}$$

bounds x_t from above from the index M_i. Now, the above argument can be repeated to obtain an index M_{i+1} such that Inequality (26) hold for $i+1$ with probability $p_1 p_2 \dots p_i p_{i+1}$.

Since $(1+\gamma)/2 < 1$, there exists an index k for which $((1+\gamma)/2)^k C_1 < \epsilon$. Then we get that Equation (24) is satisfied when we choose $p_1, \dots, p_k$ in a way that $p_1 p_2 \dots p_k \geq 1 - \epsilon$ and we let $T = M_k (= M_k(p_1, p_2, \dots, p_k))$. Q.E.D.

When Equation (23) is subject to decaying perturbations, say ϵ_t, then the proof does not apply any more. The problem is that $\|x_t\| \leq \|x_1\|$ (or $\|x_{T+t}\| \leq \|x_T\|$, for large enough T) can no longer be ensured without additional assumptions. For $x_{t+1}(z) \leq \|x_t\|$ to hold, we would need that $\gamma\epsilon_t \leq (1-\gamma)\|x_t\|$, and if $\liminf_{t\to\infty} \|x_t\| = 0$ then we could not check this relation *a priori*. Thus we choose another way to prove Lemma 12. Notice first, that the key idea in the above proof is to bound x_t by y_t. This can be done if we assume that x_t is kept bounded artificially, e.g., by scaling. The next subsection shows that such a change of x_t does not effect the convergence properties.

E.2 SCALED HOMOGENEOUS PROCESSES

The next lemma is about homogeneous processes, that is about processes of form

$$x_{n+1} = G_n(x_n, \epsilon_n), \tag{27}$$

where G_n is a random function which is homogeneous, i.e.,

$$\beta G_n(x, \epsilon) = G_n(\beta x, \beta \epsilon) \tag{28}$$

holds for all $\beta > 0$, x and ϵ. We are interested in the question whether x_n converges to zero or not. Note that δ_t defined by Inequality (4), when the inequality is replaced by equality, is a homogeneous process. The lemma below says that, under additional conditions, it is enough to prove the convergence of a modified process *which is kept bounded by scaling* to zero, that is, for the process

$$y_{n+1} = \begin{cases} G_n(y_n, \epsilon_n), & \text{if } \|G_n(y_n, \epsilon_n)\| \le T; \\ T\, G_n(y_n, \epsilon_n)/\|G_n(y_n, \epsilon_n)\|, & \text{otherwise,} \end{cases} \tag{29}$$

where $T > 0$ is an arbitrary fixed number.

Let us denote the solution of Equation (27) corresponding to the initial condition $x_0 = w$ and the sequence $\epsilon = \{\epsilon_k\}$ by $x_n(w, \epsilon)$. Similarly, let us denote the solution of Equation (29) corresponding to the initial condition $y_0 = w$ and the sequence ϵ by $y_n(w, \epsilon)$.

We say that the process x_n is *insensitive to finite perturbations of* ϵ if it holds that if $x_n(w, \epsilon)$ converges to zero then so does $x_n(w, \epsilon')$, where ϵ' is an arbitrary sequence that differs only in a finite number of terms from ϵ. Further, we say that the process x_n is *insensitive to scaling of* ϵ *by numbers smaller than* 1, if for all $0 < c < 1$ there holds that if $x_n(w, \epsilon)$ converges to zero then so does $x_n(w, c\epsilon)$.

Lemma 11 *Let us fix an arbitrary positive number T and an arbitrary w_0 and sequence ϵ. Then, a homogeneous process $x_n(w_0, \epsilon)$ converges to zero with probability one, provided that x_n is insensitive to finite perturbations of ϵ and also x_n is insensitive to the scaling of ϵ by numbers smaller than one and $y_n(w_0, \epsilon)$ converges to zero.*

Proof: Let c_k be an arbitrary sequence of reals. For convenience, we will denote the product sequence $\{c_k \epsilon_k\}$ by $c\epsilon$. We state that

$$y_n(w, \epsilon) = x_n(d_n w, c_n \epsilon) \tag{30}$$

for some sequences $\{c_n\}$ and $\{d_n\}$ satisfying $0 < d_n \le 1$ and $c_n = (c_{n0}, c_{n1}, \ldots, c_{nk}, \ldots)$, with $0 < c_{nk} \le 1$ and $c_{nk} = 1$ for $k \ge n$. Note that $y_n(w, \epsilon)$, and also $x_n(w, \epsilon)$ depends only on $\epsilon_0, \ldots, \epsilon_{n-1}$. Thus, it is possible to prove Equation (30) by constructing the appropriate sequences c_n and d_n.

Set $c_{0i} = 1$ for all $i = 0, 1, 2, \ldots$ and let $d_0 = 1$. Then, Equation (30) holds for $n = 0$. Let us assume that Equation (30) holds for n. Let S_n be the "scaling coefficient" of y_n at step $(n+1)$ ($S_n = 1$ if there is no scaling, otherwise $0 < S_n < 1$ with $S_n = T/\|G_n(y_n, \epsilon_n)\|$):

$$\begin{aligned} y_{n+1}(w, \epsilon) &= S_n G_n(y_n(w, \epsilon), \epsilon_n) \\ &= G_n(S_n y_n(w, \epsilon), S_n \epsilon_n) \\ &= G_n(S_n x_n(d_n w, c_n \epsilon), S_n \epsilon_n). \end{aligned}$$

We claim that

$$Sx_n(w,\epsilon) = x_n(Sw, S\epsilon) \tag{31}$$

holds for all w, ϵ and $S > 0$.

For $n = 0$, this obviously holds. Assume that it holds for n. Then

$$\begin{aligned} Sx_{n+1}(w,\epsilon) &= SG_n(x_n(w,\epsilon),\epsilon_n) \\ &= G_n(Sx_n(w,\epsilon), S\epsilon_n) \\ &= G_n(x_n(Sw,S\epsilon), S\epsilon_n) \\ &= x_{n+1}(Sw, S\epsilon). \end{aligned}$$

Thus,

$$y_{n+1}(w,\epsilon) = G_n(x_n(S_n d_n w, S_n c_n \epsilon), S_n \epsilon_n)$$

and we see that Equation (30) holds if we define $c_{n+1,\cdot}$ through $c_{n+1,i} := S_n c_{ni}$, $i = 0, \ldots, n$ and we let $c_{n+1,i} = 1$ for $i > n+1$ and $d_{n+1} = S_n d_n$.

Thus, we find that with the sequences

$$c_{n,i} = \begin{cases} \prod_{j=i}^{n-1} S_j, & \text{if } i < n; \\ 1, & \text{otherwise,} \end{cases}$$

$d_0 = 1$, and

$$d_{n+1} = \prod_{j=0}^{n} S_j$$

Equation (30) is satisfied.

Now, assume that we want to prove for a particular sequence ϵ and initial value w that

$$\lim_{n\to\infty} x_n(w,\epsilon) = 0 \tag{32}$$

holds with probability one. It is enough to prove that Equation (32) holds with probability $1-\delta$ when $\delta > 0$ is an arbitrary, small enough number.

We know that $y_n(w,\epsilon) \to 0$ with probability one. We may assume that $T > \delta$. Then, there exist an index $M = M(\delta)$ such that if $n > M$ then

$$\Pr(\|y_n(w,\epsilon)\| < \delta) > 1 - \delta.$$

Now, let us restrict our attention to those events for which $\|y_n(w,\epsilon)\| < \delta$ for all $n > M$. Since $\delta < T$, we get that there is no rescaling after step M: $S_n = 1$ if $n > M$. Thus, $c_{n,i} = c_{M+1,i}$ for all $n \geq M+1$ and i, and specifically $c_{n,i} = 1$ if $i, t \geq M+1$. Similarly, if $n > M$ then $d_{n+1} = \prod_{i=0}^{M} S_i = d_{M+1}$. By Equation (30), we have that if $n > M$ then

$$y_n(w,\epsilon) = x_n(d_{M+1}w, c_{M+1}\epsilon).$$

Thus, we have that $x_n(d_{M+1}w, c_{M+1}\epsilon)$ converges to zero and by Equation (31), $x_n(w, c_{M+1}\epsilon/d_{M+1})$ converges to zero. Since x_n is insensitive to finite perturbations (in c_{M+1} only a finite number of entries differs from 1), $x_n(w, \epsilon/d_{M+1})$ also converges to zero, and further since $d_{M+1} < 1$, $x_n(w,\epsilon)$ converges to zero, too (x_n is insensitive to scaling of ϵ by d_{M+1}). All these hold with probability at least $1-\delta$. Since δ was arbitrary, the lemma follows. Q.E.D.

Now, we are in the position to prove that Lemma 10 is immune against decaying perturbations.

Lemma 12 *Assume that the conditions of Lemma 10 are satisfied but Equation (23) is replaced by*

$$x_{t+1}(z) = g_t(z)x_t(z) + f_t(z)(\|x_t\| + \epsilon_t), \tag{33}$$

where $\epsilon_t \geq 0$ and ϵ_t converges to zero with probability one. Then $x_t(z)$ still converges to zero with probability 1 uniformly over Z.

Proof: We follow the proof of Lemma 11. First, we show that the process of Equation (33) satisfies the assumptions of Lemma 11 and, thus, it is enough to consider the version of Equation (33) that is kept bounded by scaling.

First, note that x_t is a homogeneous process. Let us prove that x_t is immune against finite perturbations of ϵ. To this end, assume that ϵ'_t differs only in a finite number of terms from ϵ_t, and let

$$y_{t+1}(z) = g_t(z)y_t(z) + f_t(z)(\|y_t\| + \epsilon'_t).$$

Take

$$k_t(z) = |x_t(z) - y_t(z)|.$$

Then,

$$k_{t+1}(z) \leq g_t(z)k_t(z) + f_t(z)(\|k_t(z)\| + |\epsilon_t - \epsilon'_t|).$$

For large enough t,

$$k_{t+1}(z) \leq g_t(z)k_t(z) + f_t(z)\|k_t(z)\|,$$

which is known to converge to zero by Lemma 10. Thus, x_t and y_t both converge or not converge and if one converges then the other must converge to the same value.

The other requirement that we must satisfy to be able to apply Lemma 11 is that x_n is insensitive to scaling of the perturbation by numbers smaller than one; let us choose a number $0 < c < 1$ and assume that $x_n(w, \epsilon)$ converges to zero with probability one. Then, since $x_n(w, c\epsilon) \leq x_n(w, \epsilon)$, $x_n(w, c\epsilon)$ converges to zero with probability one, too.

Now, let us prove that the process that is obtained from x_t by keeping it bounded converges to zero. The proof is the mere repetition of the proof of Lemma 10 except a few points that we discuss now. Let us denote by $\hat{x}_t$ the process that is kept bounded and let the bound be C_1. It is enough to prove that $\|\hat{x}_t\|$ converges to zero with probability one. Now, Equation (25) is replaced by

$$y_{t+1}(z) = g_t(z)y_t(z) + \gamma(1 - g_t(z))(C_1 + \epsilon_t).$$

Now, y_t still converges to γC_1 by Lemma 3.5 of Szepesvári [49] and also $0 \leq \hat{x}_t \leq y_t$. Thus, the whole argument of Lemma 10 can be repeated for the process $\hat{x}_t$, and we get that $\|\hat{x}_t\|$ converges to zero with probability one and consequently so does $\|x_t\|$. Q.E.D.

F CENTRALIZED LEARNING RATES

Note that the learning rate of the on-line Q-learning procedure is given by $\chi(x = x_t, a = a_t)\alpha_t(x, a)$ for a given state-action pair (x, a). Thus, in order to ensure that Q_t converges to Q^* uniformly we have to be certain that $\sum_{t=1}^{\infty} \chi(x = x_t, a = a_t)\alpha_t(x, a)$ is infinity. In practice, instead of a separate learning rate for each state-action pair, often a single learning rate is used (this choice is especially reasonable if the spreading version of Q-learning or soft state aggregation is used). Bradtke conjectured that if this single "centralized" learning rate, α_t, satisfies the criteria $\sum_{t=1}^{\infty} \alpha_t = \infty$ and $\sum_{t=1}^{\infty} \alpha_t^2 < \infty$ then for each (x, a) $\sum_{t=1}^{\infty} \alpha_t \chi(x = x_t, a = a_t) = \infty$ and $\sum_{t=1}^{\infty} \alpha_t^2 \chi(x = x_t, a = a_t) < \infty$ will still hold [8]. This second condition is satisfied since $\alpha_t^2 \chi(x = x_t, a = a_t) \leq \alpha_t^2$ for all $t \geq 1$. The following propositions give some conditions under which the first condition is still met.

Lemma 13 *If the state-action inter-arrival times in $\chi(x = x_t, a = a_t)$ have a common upper bound then $\sum_{t=1}^{\infty} \chi(x = x_t, a = a_t)\alpha_t(x, a) = \infty$ holds with probability one provided that $\sum_{t=1}^{\infty} \alpha_t(x, a) = \infty$ and $\alpha_t(x, a)$ is a decreasing sequence.*

Proof: It is enough to prove the statement for an arbitrary decreasing numerical sequence, α_t. Note that $\sum_{t=1}^{\infty} \chi_t \alpha_t$ can be rewritten as $\sum_{t=1}^{\infty} \alpha_{n_t}$ with an appropriate increasing random sequence n_t. Assume that $n_{t+1} - n_t \leq d$, i.e., that the state-action inter-arrival times are bounded. Since $\alpha_{n_t} \geq \alpha_{n_t+i}$, $i = 0, 1, \ldots, n_{t+1} - n_t - 1$,

$$(n_{t+1} - n_t)\alpha_{n_t} \geq \sum_{i=0}^{n_{t+1}-n_t-1} \alpha_{n_t+i}$$

and

$$\alpha_{n_t} \geq \frac{1}{d} \sum_{i=0}^{n_{t+1}-n_t-1} \alpha_{n_t+i}$$

holds for all t with probability one. Summing this with respect to t, we get

$$\sum_{t=1}^{\infty} \alpha_{n_t} \geq \frac{1}{d} \sum_{t=1}^{\infty} \sum_{i=0}^{n_{t+1}-n_t-1} \alpha_{n_t+i} = \frac{1}{c} \sum_{n=1}^{\infty} \alpha_n = \infty$$

holds with probability one. Q.E.D.

An important extension of the above proposition is the following.

Lemma 14 *Assume that α_n is a decreasing sequence and let d_t be random numbers. Assume that there exists a sequence R_t such that*

$$\sum_{t=1}^{\infty} \Pr(d_t \geq R_t) < \infty$$

and the R_t thinning of $\sum_n \alpha_n$ (that is $\sum_t \alpha_{m_t}$, where $m_1 = 1$ and $m_{t+1} = m_t + R_t$) still converges to infinity. Then

$$\sum_{t=1}^{\infty} \alpha_{n_t} = \infty$$

with probability one, where $n_1 = 1$ and $n_{t+1} = n_t + d_t$.

Proof: It is enough to prove that there exists a fixed T such that $d_t \le R_t$ holds *for all* $t > T$ with probability one, since then $m_t \ge n_{t-\tau}$ and, thus, $\alpha_{m_t} \le \alpha_{n_{t-\tau}}$ for $\tau = \max\{s | n_{T-s} \le m_T\}$. Let $A_t = \{\omega | d_t(\omega) \ge R_t\}$. The required statement is equivalent to

$$\Pr(\cup_{n=1}^{\infty} \cap_{t=n}^{\infty} A_t) = 1. \tag{34}$$

According to the Borel-Cantelli Lemma, in order to prove Equation (34) it is enough to show that $\sum_t \Pr(A_t) < \infty$ However, this holds by assumption. Q.E.D.

The main result of this section is the following.

Theorem 12 *Let $\alpha_n = 1/n$ and assume that the d_ts are exponentially distributed with common parameters, that is there exist numbers p and q such that $\Pr(d_t \ge k) \le qp^k$. Then*

$$\sum_{t=1}^{\infty} \alpha_{n_t} = \infty$$

with probability one, where $n_1 = 1$ and $n_{t+1} = n_t + d_t$.

Proof: We make use of Lemma 14. Let $R_t = (2/c)\log t$, where $c = \log(1/p)$. Then, $\Pr(d_t \ge R_t) = q/t^2$ and, thus, $\sum_{t=1}^{\infty} \Pr(d_t \ge R_t) < \infty$. On the other hand, let $m_1 = 1$ and $m_{t+1} = \lfloor m_t + R_t \rfloor = \lfloor m_t + (2/c)\log t \rfloor$. Then, $m_t \le \lceil (2/c) t \log t \rceil$ can be proved by induction, giving us

$$\sum_t a_{m_t} = \sum_t \frac{1}{m_t} \ge \left\lceil \frac{c}{2} \right\rceil \sum_t \frac{1}{\lfloor t \log t \rfloor} = \infty$$

and proving the theorem. Q.E.D.

The implication of this result for Q-learning is this. If the policy followed by the learner is eventually strongly ergodic, then the distribution of the visits of a state becomes exponential. This means that it in Q-learning it is sufficient to consider a unique sequence of learning rates together with sufficient exploration to ensure convergence to the optimal Q-function.

G CONVERGENCE OF Q-LEARNING FOR RISK-SENSITIVE MODELS

Assume that both X and A are finite. Heger studied the risk-sensitive criterion defined by $(\bigoplus g)(x,a) = \min_{y:P(a,x,y)>0} g(x,a,y)$ and $(\bigotimes f)(x) = \max_a f(x,a)$. In this section, we will prove the following theorem (also proven by Heger [14]).

Theorem 13 *Let*

$$Q_{t+1}(x,a) = \begin{cases} \min\left(r_t + \gamma[\bigotimes Q_t](y_t), Q_t(x,a)\right); & \textit{if } (x,a) = (x_t,a_t); \\ Q_t(x,a); & \textit{otherwise,} \end{cases} \tag{35}$$

where $\langle x_t, a_t, y_t, r_t \rangle$ is the experience of the agent at time t, y_t is selected according to $P(x,a,\cdot)$, and r_t is a random variable with $\liminf_{(x,a,y)=(x_t,a_t,y_t)} r_t = R(x,a,y)$ with probability one. Then, Q_t converges to Q^*, the fixed point of operator K, $KQ = \bigoplus(R + \gamma \bigotimes Q)$ provided that $Q_0 \ge Q^*$ and every state-action pair is updated infinitely often.

Proof: We prove this theorem in two parts. First, we assume that $r_t = R(x_t, a_t, y_t)$, i.e., r_t is non-random. We want to use Theorem 3.1. Let our random operator sequence be defined in the customary way:

$$[T_t(Q', Q)](x,a) = \begin{cases} \min\left(r_t + \gamma[\bigotimes Q_t](y_t), Q(x,a)\right); & \text{if } (x,a) = (x_t, a_t); \\ Q'(x,a); & \text{otherwise,} \end{cases}$$

It is immediate that T_t approximates T at Q^* *provided* that $Q_0 \geq Q^*$. However, the definition of an appropriate function F_t seems to be impossible since $F_t(x,a) \leq \gamma$ should hold if $(x,a) = (x_t, a_t)$. But, if $r_t + \gamma[\bigotimes Q](y_t) \geq Q(x,a)$, then $T_t(Q', Q^*)(x,a) - T_t(Q', Q)(x,a) = Q^*(x,a) - Q(x,a)$, and there is no guarantee that $|Q^*(x,a) - Q(x,a)| \leq \gamma \|Q - Q^*\|$. In the other case, when $r_t + \gamma[\bigotimes Q](y_t) < Q(x,a)$, $T_t(Q', Q^*)(x,a) - T_t(Q', Q)(x,a) = Q^*(x,a) - (R(x_t, a_t, y_t) + \gamma[\bigotimes Q](y_t))$, which seems much more promising since $Q^*(x,a) = R(x,a,y) + \gamma[\bigotimes Q^*](y)$ for $y = \text{argmin}_z R(x,a,z) + \gamma[\bigotimes Q^*](z)$. If $y = y_t$ (by chance), then it follows that $|T_t(Q', Q^*)(x,a) - T_t(Q', Q)(x,a)| \leq \gamma \|Q - Q^*\|$. Since the case when $(x,a) \neq (x_t, a_t)$, i.e., when there is no update, is pleasant ($G_t(x,a) = 1$ and $F_t(x,a) = 0$), the idea is to restrict the updates to the other tractable case when $y_t = y$.

Let the set of critical states for a given (x,a) pair be given by

$$M(x,a) = \{y \in X \mid P(x,a,y) > 0,\ Q^*(x,a) = R(x,a,y) + \gamma[\bigotimes Q^*](y)\}.$$

$M(x,a)$ is non-empty since X is finite. Let us consider the artificial operators

$$T'_t(Q', Q)(x,a) = \begin{cases} \min\left(r_t + \gamma[\bigotimes Q](y_t), Q(x,a)\right); & \text{if } (x,a) = (x_t, a_t) \text{ and } y_t \in M(x,a); \\ Q'(x,a); & \text{otherwise} \end{cases}$$

and the sequence $Q'_0 = Q_0$ and $Q'_{t+1} = T'_t(Q'_t, Q'_t)$. Now, the question is whether it is sufficient to consider the convergence of Q'_t. Fortunately, it is. Since there are no more updates (decreases of value) in the sequence defined by T'_t, we have that $Q^* \leq Q_t \leq Q'_t$ and, thus, if Q'_t converges to Q^* then necessarily so does Q_t. It is again immediate that T'_t still approximates T at Q^* and also that

$$G_t(x,a) = \begin{cases} 0; & \text{if } (x,a) = (x_t, a_t) \text{ and } y_t \in M(x,a), \\ 1; & \text{otherwise.} \end{cases}$$

Let us show that we can also define a suitable F_t function for T'_t. Assume first that $(x,a) = (x_t, a_t)$ and $y_t \in M(x,a)$. Note that we may assume that all the test functions are overestimating, in particular in the inequality below we may assume that $Q' \geq Q^*$.

$$\begin{aligned} |T'_t(Q', Q)(x,a) \quad - \quad & T'_t(Q', Q^*)(x,a)| = \min(r_t + \gamma[\bigotimes Q](y_t), Q(x,a)) - (r_t + \gamma[\bigotimes Q^*](y_t)) \\ \leq \quad & r_t + \gamma[\bigotimes Q](y_t) - (r_t + \gamma[\bigotimes Q^*](y_t)) \\ \leq \quad & \gamma([\bigotimes Q](y_t) - [\bigotimes Q^*](y_t)) \leq \gamma \|Q - Q^*\|. \end{aligned}$$

In the other case, when $(x,a) \neq (x_t, a_t)$ or $y_t \notin M(x,a)$ then

$$|T'_t(Q', Q)(x,a) - T'_t(Q', Q^*)(x,a)| = |Q'(x,a) - Q'(x,a)| = 0.$$

Thus,

$$F_t(x,a) = \begin{cases} \gamma; & \text{if } (x,a) = (x_t,a_t) \text{ and } y_t \in M(x,a), \\ 0; & \text{otherwise,} \end{cases}$$

Condition (3) of Theorem 3.1 is satisfied if and only if the event $\{(x,a) = (x_t,a_t), y_t \in M(x,a)\}$ occurs infinitely often. By assumption, $\{(x,a) = (x_t,a_t)\}$ occurs infinitely many times and, since if $y \in M(x,a)$ then $P(x,a,y) > 0$ and since y_t is sampled according to $P(x,a,\cdot)$, $(x,a) = (x_t,a_t), y_t \in M(x,a)$ occurs infinitely often, too. Finally, Condition (4) is satisfied since, for all t, $F_t(x) = \gamma(1 - G_t(x))$ holds.

Now, we turn to the second part of the proof which is based on an idea very similar in spirit to the idea on which the proof of the first part was built. Assume that the rewards r_t are random and

$$\liminf_{(x,a,y)=(x_t,a_t,y_t), t\to\infty} r_t = R(x,a,y)$$

with probability one and, thus, for each triple (x,a,y) we may choose a subsequence t_k such that $(x,a,y) = (x_{t_k}, a_{t_k}, y_{t_k})$ for all k and

$$\lim_{k\to\infty} r_{t_k} = R(x,a,y).$$

Let $T(x,a,y)$ be the set of numbers $\{t_k\}$ when $(x_t,a_t,y_t) = (x,a,y)$. Further, let $T(x,a) = \cup_{y\in M(x,a)} T(x,a,y)$. The above proof can be repeated almost exactly, the only exception is that we must further restrict the updates. Now let

$$T'_t(Q',Q)(x,a) = \begin{cases} \min\left(r_t + \gamma[\bigotimes Q](y_t), Q(x,a)\right); & \text{if } t \in T(x,a), \\ Q'(x,a); & \text{otherwise.} \end{cases}$$

We still have that T'_t approximates T at Q^* with probability one provided that $Q_0 \geq Q^*$. As in the previous case, we get that

$$G_t(x,a) = \begin{cases} 0; & \text{if } t \in T(x,a); \\ 1; & \text{otherwise} \end{cases}$$

but the estimation of $|T'_t(Q',Q)(x,a) - T'_t(Q',Q^*)(x,a)|$ must be changed. Assume first that $t \in T(x,a)$. Then,

$$\begin{aligned} & |T'_t(Q',Q)(x,a) - T'_t(Q',Q^*)(x,a)| && (36) \\ & = \min(r_t + \gamma[\bigotimes Q](y_t), Q(x,a)) - (R(x,a,y_t) + \gamma[\bigotimes Q^*](y_t)) \\ & \leq r_t + \gamma[\bigotimes Q](y_t) - (R(x,a,y_t) + \gamma[\bigotimes Q^*](y_t)) \\ & \leq \gamma\|Q - Q^*\| + |r_t - R(x,a,y_t)|. && (37) \end{aligned}$$

Let $\sigma_t(x,a) = |r_t - R(x,a,y_t)|$. Note, that

$$\lim_{t\to\infty, t\in T(x,a)} \sigma_t(x,a) = 0$$

with probability one. In the other case $|T'_t(Q',Q)(x,a) - T'_t(Q',Q^*)(x,a)| = 0$. Because of the appearance of $\sigma_t(x,a)$ in Inequality (37) instead of Condition 2 of Theorem 3.1 we have that

$$|T'_t(Q',Q)(x,a) - T'_t(Q',Q^*)(x,a)| = F_t(x,a)(\|Q - Q^*\| + \lambda_t(x,a)),$$

where

$$F_t(x,a) = \begin{cases} \gamma; & \text{if } t \in T(x,a), \\ 0; & \text{otherwise,} \end{cases}$$

and $\lambda_t(x,a) = \sigma_t(x,a)/\gamma$ if $t \in T(x,a)$, $\lambda_t(x,a) = 0$, otherwise.

But, then the change in the proof of Theorem 3.1 is just superficial, namely, instead of Equation (4) we have that (in the notation of Theorem 3.1)

$$\delta_{t+1}(x) \le G_t(x)\delta_t(x) + F_t(x)\left(\|\delta_t\| + \Delta_t + \lambda_t(x)\right), \tag{38}$$

where $\lambda_t(x)$ converges to zero with probability one. However, this additional decaying perturbation can be blended into Δ_t and, thus, we see that our method still applies and Q-learning converges to Q^* in this case, as well. Q.E.D.

References

[1] Andrew G. Barto, S. J. Bradtke, and Satinder P. Singh. Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72(1):81–138, 1995.

[2] Andrew G. Barto, Richard S. Sutton, and Christopher J. C. H. Watkins. Learning and sequential decision making. Technical Report 89-95, Department of Computer and Information Science, University of Massachusetts, Amherst, MA, 1989. Also published in *Learning and Computational Neuroscience: Foundations of Adaptive Networks*, Michael Gabriel and John Moore, editors. The MIT Press, Cambridge, MA, 1991.

[3] Richard Bellman. *Dynamic Programming.* Princeton University Press, Princeton, NJ, 1957.

[4] A. Benveniste, M. Métivier, and P. Priouret. *Adaptive Algorithms and Stochastic Approximations.* Springer Verlag, New York, 1990.

[5] D.P. Bertsekas. Monotone mappings with application in dynamic programming. *SIAM Journal on Control and Optimization*, 15(3):438–464, 1977.

[6] D.P. Bertsekas and S.E. Shreve. *Stochastic Optimal Control: The Discrete Time Case.* Academic Press, New York, 1978.

[7] Justin A. Boyan. Modular neural networks for learning context-dependent game strategies. Master's thesis, Department of Engineering and Computer Laboratory, University of Cambridge, Cambridge, UK, August 1992.

[8] S.J. Bradtke. Incremental dynamic programming for on-line adaptive optimal control. Technical Report 94-62, Department of Computer and Information Science, University of Massachusetts, Amherst, Massachusetts, 1994.

[9] Anne Condon. The complexity of stochastic games. *Information and Computation*, 96(2):203–224, February 1992.

[10] Anne Condon. On algorithms for simple stochastic games. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 13:51–71, 1993.

[11] Cyrus Derman. *Finite State Markovian Decision Processes.* Academic Press, New York, NY, 1970.

[12] Geoffrey J. Gordon. Stable function approximation in dynamic programming. In Armand Prieditis and Stuart Russell, editors, *Proceedings of the Twelfth International Conference on Machine Learning*, pages 261–268, San Francisco, CA, 1995. Morgan Kaufmann.

[13] Vijaykumar Gullapalli and Andrew G. Barto. Convergence of indirect adaptive asynchronous value iteration algorithms. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 695–702, San Mateo, CA, 1994. Morgan Kaufmann.

[14] Matthias Heger. Risk-averse reinforcement learning. Ph.D. thesis, in preparation.

[15] Matthias Heger. Consideration of risk in reinforcement learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 105–111, San Francisco, CA, 1994. Morgan Kaufmann.

[16] Matthias Heger. The loss from imperfect value functions in expectation-based and minimax-based tasks. *Machine Learning*, 22(1/2/3):197–226, 1996.

[17] A. J. Hoffman and R. M. Karp. On nonterminating stochastic games. *Management Science*, 12:359–370, 1966.

[18] Ronald A. Howard. *Dynamic Programming and Markov Processes.* The MIT Press, Cambridge, Massachusetts, 1960.

[19] Tommi Jaakkola, Michael I. Jordan, and Satinder P. Singh. On the convergence of stochastic iterative dynamic programming algorithms. *Neural Computation*, 6(6):1185–1201, November 1994.

[20] George H. John. When the best move isn't optimal: Q-learning with exploration. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, page 1464, Seattle, WA, 1994.

[21] George H. John. When the best move isn't optimal: Q-learning with exploration. Unpublished manuscript, available through URL `ftp://starry.stanford.edu/pub/gjohn/papers/rein-nips.ps`, 1995.

[22] R. E. Korf. Real-time heuristic search. *Artificial Intelligence*, 42:189–211, 1990.

[23] P.R. Kumar. A survey of some results in stochastic adaptive controls. *SIAM Journal of Control and Optimization*, 23:329–380, 1985.

[24] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 157–163, San Francisco, CA, 1994. Morgan Kaufmann.

[25] Michael L. Littman. Combining exploration and contol in reinforcement learning: The convergence of SARSA. Unpublished manuscript. Available through URL `http://www.cs.duke.edu/~mlittman`, 1996.

[26] Michael Lederman Littman. *Algorithms for Sequential Decision Making.* PhD thesis, Department of Computer Science, Brown University, February 1996. Also Technical Report CS-96-09.

[27] Sridhar Mahadevan. Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine Learning*, 22(1/2/3):159–196, 1996.

[28] Matthew A. F. McDonald and Philip Hingston. Approximate discounted dynamic programming is unreliable. Technical report 94/7, Department of Computer Science, The University of Western Australia, Crawkey, WA, 6009, October 1994.

[29] Andrew W. Moore and Christopher G. Atkeson. Prioritized sweeping: Reinforcement learning with less data and less real time. *Machine Learning*, 13:103–130, 1993.

[30] Jing Peng and Ronald J. Williams. Incremental multi-step Q-learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 226–232, San Francisco, CA, 1994. Morgan Kaufmann.

[31] Martin L. Puterman. *Markov Decision Processes—Discrete Stochastic Dynamic Programming.* John Wiley & Sons, Inc., New York, NY, 1994.

[32] Carlos Ribeiro and Csaba Szepesvári. Spatial spreading of action values in Q-learning. In *Proceedings of ISRF-IEE International Conference: Intelligent and Cognitive Systems, Neural Networks Symposium*, pages 32–36, 1996.

[33] C.H.C. Ribeiro. Attentional mechanisms as a strategy for generalisation in the Q-learning algorithm. In *Proceedings of ICANN'95*, volume 1, pages 455–460, 1995.

[34] Herbert Robbins and Sutton Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.

[35] G. A. Rummery. *Problem solving with reinforcement learning.* PhD thesis, Cambridge University Engineering Department, 1994.

[36] G. A. Rummery and M. Niranjan. On-line Q-learning using connectionist systems. Technical Report CUED/F-INFENG/TR 166, Cambridge University Engineering Department, 1994.

[37] Nicol N. Schraudolph, Peter Dayan, and Terrence J. Sejnowski. Temporal difference learning of position evaluation in the game of Go. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 817–824, San Mateo, CA, 1994. Morgan Kaufmann.

[38] Anton Schwartz. A reinforcement learning method for maximizing undiscounted rewards. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 298–305, Amherst, MA, 1993. Morgan Kaufmann.

[39] L.S. Shapley. Stochastic games. *Proceedings of the National Academy of Sciences of the United States of America*, 39:1095–1100, 1953.

[40] Satinder P. Singh and Vijaykumar Gullapalli. Asynchronous modified policy iteration with single-sided updates. Unpublished manuscript. Available through URL `ftp://ftp.cs.colorado.edu/users/baveja/Papers/single-sided.ps.Z`, 1993.

[41] Satinder P. Singh and Richard S. Sutton. Reinforcement learning with replacing eligibility traces. *Machine Learning*, 22(1/2/3):123–158, 1996.

[42] Satinder Pal Singh and Richard C. Yee. An upper bound on the loss from approximate optimal-value functions. *Machine Learning*, 16:227, 1994.

[43] S.P. Singh, T. Jaakkola, and M.I. Jordan. Reinforcement learning with soft state aggregation. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 361–368, Cambridge, MA, 1995. The MIT Press.

[44] D.R. Smart. *Fixed point theorems.* Cambridge University Press, Cambridge, 1974.

[45] Richard S. Sutton. Learning to predict by the method of temporal differences. *Machine Learning*, 3(1):9–44, 1988.

[46] Richard S. Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the Seventh International Conference on Machine Learning*, pages 216–224, Austin, TX, 1990. Morgan Kaufmann.

[47] Richard S. Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, Cambridge, MA, 1996. The MIT Press.

[48] Cs. Szepesvári. Abstract dynamic programming under monotonicity assumptions: Non-Markovian policies, policy iteration and the optimality equation. Technical Report 96-103, Research Group on Artificial Intelligence, JATE-MTA, August 1996.

[49] Cs. Szepesvári. Synthesis of neural networks: the case of cascaded Hebbians. Technical Report TR-96-102, Research Group on Artificial Intelligence, JATE-MTA, August 1996.

[50] Csaba Szepesvári. A general framework for reinforcement learning. In *Proceedings of ICANN'95*, volume 2, pages 165–170, 1995.

[51] Gerald Tesauro. Temporal difference learning and TD-Gammon. *Communications of the ACM*, pages 58–67, March 1995.

[52] Sebastian Thrun. Learning to play the game of chess. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 1069–1076, Cambridge, MA, 1995. The MIT Press.

[53] John N. Tsitsiklis. Asynchronous stochastic approximation and Q-learning. *Machine Learning*, 16(3):185–202, September 1994.

[54] John N. Tsitsiklis and Benjamin Van Roy. An analysis of temporal-difference learning with function approximation. Technical Report LIDS-P-2322, Massachusetts Institute of Technology, March 1996. Available through URL `http://web.mit.edu/bvr/www/td.ps`. To appear in *IEEE Transactions on Automatic Control*.

[55] John N. Tsitsiklis and Benjamin Van Roy. Feature-based methods for large scale dynamic programming. *Machine Learning*, 22(1/2/3):59–94, 1996.

[56] Sergio Verdu and H. Vincent Poor. Abstract dynamic programming models under commutativity conditions. *SIAM Journal of Control and Optimization*, 25(4):990–1006, July 1987.

[57] O. J. Vrieze and S. H. Tijs. Fictitious play applied to sequences of games and discounted stochastic games. *International Journal of Game Theory*, 11(2):71–85, 1982.

[58] K.-H. Waldmann. On bounds for dynamic programs. *Mathematics of Operations Research*, 10(2):220–232, May 1985.

[59] Christopher J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, UK, 1989.

[60] Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3):279–292, 1992.

[61] Ronald J. Williams and Leemon C. Baird, III. Analysis of some incremental variants of policy iteration: First steps toward understanding actor-critic learning systems. Technical Report NU-CCS-93-11, Northeastern University, College of Computer Science, Boston, MA, September 1993.

[62] Ronald J. Williams and Leemon C. Baird, III. Tight performance bounds on greedy policies based on imperfect value functions. Technical Report NU-CCS-93-14, Northeastern University, College of Computer Science, Boston, MA, November 1993.