Journal of Machine Learning Research 5 (2004) 325-360 Submitted 8/02; Revised 12/03; Published 4/04

A Geometric Approach to Multi-Criterion Reinforcement Learning

Shie Mannor SHIE@MIT.EDU
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Nahum Shimkin SHIMKIN@EE.TECHNION.AC.IL
Department of Electrical Engineering
Technion, Israel Institute of Technology
Haifa 32000, Israel

Editor: Sridhar Mahadevan

Abstract

We consider the problem of reinforcement learning in a controlled Markov environment with multiple objective functions of the long-term average reward type. The environment is initially unknown, and furthermore may be affected by the actions of other agents, actions that are observed but cannot be predicted beforehand. We capture this situation using a stochastic game model, where the learning agent is facing an adversary whose policy is arbitrary and unknown, and where the reward function is vector-valued. State recurrence conditions are imposed throughout. In our basic problem formulation, a desired target set is specified in the vector reward space, and the objective of the learning agent is to *approach* the target set, in the sense that the long-term average reward vector will belong to this set. We devise appropriate learning algorithms, that essentially use multiple reinforcement learning algorithms for the standard scalar reward problem, which are combined using the geometric insight from the theory of approachability for vector-valued stochastic games. We then address the more general and optimization-related problem, where a nested class of possible target sets is prescribed, and the goal of the learning agent is to approach the smallest possible target set (which will generally depend on the unknown system parameters). A particular case which falls into this framework is that of stochastic games with average reward constraints, and further specialization provides a reinforcement learning algorithm for constrained Markov decision processes. Some basic examples are provided to illustrate these results.

1. Introduction

Agents that operate in the real world often need to consider several performance criteria simultaneously. In this paper we address the problem of RL (Reinforcement Learning) in a dynamic environment, where the controlling agent's goals are formulated in terms of multiple objective functions, each one corresponding to a long-term average reward functional. Furthermore, we deviate from the strictly Markovian environment model by allowing the presence of additional agents whose policies may be arbitrary. Our goal then is to formulate performance objectives that are meaningful in such a setting, and develop learning algorithms that attain these objectives.

Multi-criterion decision making is a well established research area with a wide range of solution concepts and methods (for an overview see, for example, Steuer, 1986; Ehrgott and Gandibleux, 2002). Most basic to multi-criterion optimization is the concept of efficient (or Pareto optimal) solutions, namely those solutions that cannot be improved upon in all coordinates (with strict improvement is at least one coordinate) by another solution. Unfortunately efficient solutions are essentially non-unique, and various methods have been developed to single out one "optimal" solution" — for example, by forming appropriate scalar combinations of the different objective functions. Another formulation of multi-criterion optimization which leads to a well-defined solution is the constrained optimization problem, where one criterion is optimized subject to explicit constraints on the others. In the context of Markov decision problems (MDPs), several papers have developed dynamic programming algorithms to compute efficient solutions (see White, 1982; Henig, 1983; Carraway et al., 1990), while constrained MDPs have received more extensive attention — see Altman (1999) and references therein. Constrained Stochastic Games were considered by Shimkin (1994) and Altman and Shwartz (2000). Learning schemes for constrained MDPs were devised by Pozniyak et al. (1999), based on the theory of stochastic learning automata. A Q-learning approach to constrained MDPs was considered by Gábor et al. (1998).

In the present paper, we shall use the notion of a *target set* as the basic concept which specifies the controller's goals with respect to its multiple objective functions. This set specifies the range in which the vector of objective functions is required to reside. More precisely, since each objective function is in the form of a long-range average reward, the controller's objective is to ensure that the average-reward vector converges to the given target set. Note that in its basic form the target set concept does not correspond to an optimization problem but rather to constraint satisfaction. However, the optimization aspect may be added by looking for target sets that are minimal in some sense; indeed, we will employ this approach in the latter part of the paper to obtain a learning algorithm for constrained stochastic games (and, in particular, constrained MDPs).

As the target set concept is basic to our problem formulation, it is worth taking some time to highlight its relevance and utility in dynamic decision problems. In various engineering applications, performance requirements are given in the form of bounds on variables of interest, that need to hold under varying system conditions (parameters) and external disturbances. A temperature regulator, for example, may be required to keep the temperature with a certain range (say $\pm 1^o$C) of the nominal operating point. The corresponding target set is then simply the interval $[\underline{T}, \overline{T}]$. In communication networks which provide quality of service (QoS) guarantees, such as certain service classes of ATM networks, a user may be given guarantees in terms of a Minimal Cell Rate (MCR), which lower bounds the available bandwidth, and a Cell Loss Ratio (CLR) that upper bounds the allowed fraction of lost cells (e.g., ATM Forum Technical Committee, 1999). The controller's goal is to keep $\mathrm{BW} \geq \mathrm{MCR}$ and $\mathrm{LR} \leq \mathrm{CLR}_0$, where BW is the available bandwidth and LR is the fraction of lost cells. This translates to a target set T of the form $T = [\mathrm{MCR}, \infty) \times [0, \mathrm{CLR}]$, which resides in the two-dimensional objective space $\mathbb{R}^2$. A more elaborate service guarantee may ensure a certain loss ratio up to a given cell rate, and allow for a higher one when the cell rate is higher; the corresponding target set can still be easily constructed in $\mathbb{R}^2$, although it will no longer be rectangular due to the coupling between the two criterions. We note that regulating mechanisms are equally prevalent in biological systems, where such quantities as temperature, pressure and concentration need to be maintained within certain bounds.

We observe, in passing, that goal setting and their attainment play a central role in prominent descriptive models of human decision making. Simon's theory of *satisfying* decisions suggests that actual decision makers are rarely optimizers, but rather accept "good enough" alternatives that meet previously determined aspiration levels (Simon, 1996). Kahaneman and Tversky's prospect theory

(Kahaneman and Tversky, 1979) evaluates gain or loss relative to a preset reference point, which can also be interpreted as an aspiration level for the decision maker.

A second challenging ingredient in our problem formulation is that of the arbitrarily varying environment. The decision problem is considered from the viewpoint of a particular controlling agent, whose environment may be affected by other, noncooperative, agents. We make no assumptions regarding the choices of these other agents, and allow them to choose their actions arbitrarily. Such agents may also represent non-stationary moves of Nature, or account for non-Markovian dynamics over a partially-observed state. For modelling simplicity we collect all the additional agents as a single one, the *adversary*, whose actions may affect both the state transitions and obtained rewards. The model then reduces to a two-person stochastic game with vector-valued rewards. The adversary is free to choose its actions according to any control policy, subject only to causality limitations (both agents are assumed to fully observe the state and action sequences). Accordingly, we shall require that the controlling agent will achieve the required goal (convergence of the average reward vector to the target set) *for any policy of the adversary*.

The proposed problem formulation is inspired by the theory of approachability, introduced by Blackwell (1956) in the context of repeated matrix games with vector payoffs. This theory provides geometric conditions, and matching control policies, for a set in the reward space to be *approachable* by a player, in the sense that the average reward vector approaches this target set for any policy of the adversary. Essentially, Blackwell approaching strategies are based on reviewing the current average reward vector, and whenever it is outside the target set, the reward is "steered" in the direction of the target set. More precisely, the controlling agent computes the direction of closest distance from current reward vector to the target set, and plays his maximin policy in the matrix game obtained by projecting the vector-valued reward function along this direction. Among its many extensions and applications, approachability theory has been extended to stochastic (Markov) games by Shimkin and Shwartz (1993) under appropriate state recurrence conditions; these conditions are enforced throughout the present paper. The relevant results are reviewed in Section 3.1 and some new results are provided in Section 3.2. In this paper we add the learning aspect, and consider the problem of learning approaching policies on-line. Accordingly, we refer below to the controlling agent as the *learning* agent.

It will be useful at this point to illustrate some of the geometric ideas behind Blackwell's approaching policies via a simple example. Note that this example is strictly illustrative. Consider first a scalar temperature regulation problem, where the performance objective of interest is to regulate the *long-term average* temperature, and guarantee that it is in some prescribed interval $[\underline{T}, \overline{T}]$. The agent may activate a cooler or a heater at will, although when activated it must remain so for a certain period of time. When the cooler is activated the temperature drops to a range $[a,b]$, with $b < \underline{T}$, and when the heater is on the temperature reaches a range $[c,d]$, with $c > \overline{T}$. The precise temperature reached in the allocated range varies due to environmental influence. Given this uncertainty, the simplest way to regulate the average temperature would be to keep note of the current average temperature (over the time interval between the initial and present time), and then activate the cooler whenever the current average increases above $\overline{T}$, and the heater whenever it decreases below $\underline{T}$: see Figure 1 for an illustration. In between these levels, any (reasonable) activation rule may be used. It should be noted that the proposed policy is not stationary in the usual sense, since it depends on the current *average* temperature, which is not part of a standard definition of a state for this system. We also note the similarity with the common "bang-bang" (or threshold) feedback policy for temperature regulation: the goal there is of course to regulate the *instantaneous* temperature, which is accordingly used as the feedback variable. It is not possible (due to the assumed relationships between the temperature intervals) to maintain the instantaneous temperature in the specified interval.

Figure 1: The single dimensional temperature regulation example. If the temperature is higher than $\overline{T}$ the control policy is to cool, and if the temperature is lower than $\underline{T}$ the control policy is to heat.

This one-dimensional example already contains one of the key elements of the approaching policies. Namely, the average reward vector is "steered" in the direction of the target set by using direction dependent sub-policies. While each sub-policy may itself be stationary, the overall approaching policy is not. Except for the special case when the adversary is trivial (so that the model reduces to a single-controller one), the use of such non-stationary policies is essential, in the sense that target sets that are approachable with general polices need not be so with stationary policies.

Consider next a multi-objective version of the temperature regulation problem. The controller's first objective, as before, is to maintain the average temperature in a certain range. One can consider other criteria of interest such as the average humidity, frequency of switching between policies, average energy consumption and so on. This problem may be characterized as a multi-criterion problem, in which the objective of the controller is to have the average reward in some target set. Suppose that the additional variable of interest is the average humidity. In a controlled environment such as a greenhouse, the allowed level of humidity depends on the temperature. An illustrative target set is shown in Figure 2. A steering policy now relies on more elaborated geometry. In place of the two directions (left/right) of the one-dimensional case, we now face a continuous range of possible directions, each associated with a possibly different steering policy. For the purpose of the proposed learning algorithm, it will be useful to consider only a finite number of steering directions (and associated policies). We will show that this can always be done, with negligible effect on the attainable performance.

The analytical basis for this work relies on three elements: the theory of approachability for vector-valued dynamic games; RL algorithms for (scalar) average reward problems; and stochastic game models. As approachability was already introduced above, we next briefly address the other two topics.

Reinforcement Learning has emerged in the last decade as a unifying discipline for learning and adaptive control. Comprehensive overviews may be found in Bertsekas and Tsitsiklis (1995); Sutton and Barto (1998), and Kaelbling et al. (1996). It should be mentioned that reinforcement learning has come to encompass a wide range of learning methods, which surpasses the restricted sense of adapting actions directly in response to a (scalar) reinforcement signal; our method belongs in this wider class. RL for average reward Markov Decision Processes (MDPs) was suggested by Schwartz (1993) and Mahadevan (1996) and later analyzed in Abounadi et al. (2002). The analysis for the average reward criterion is considerably more difficult than the discounted case, as the dynamic programming operator is no longer a contraction. Several methods exist for average reward RL,

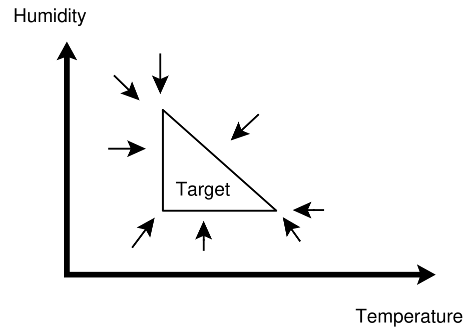

Figure 2: The two dimensional temperature-humidity example. The steering directions, which point to the closest point in the set, are denoted by arrows.

including Q-learning (Abounadi et al., 2002), the E^3 algorithm (Kearns and Singh, 1998), and actor-critic schemes (Sutton and Barto, 1998). RL for multi-objective problems was considered, to the best of our knowledge, only by Gábor et al. (1998). This paper considers discounted constrained MDPs, that is, the objective is to find a maximizing policy for the discounted reward problem, given that some additional discounted constraints are satisfied. The algorithm of Gábor et al. (1998) searches for the best *deterministic* policy. In general, however, the optimal policy for constrained MDPs is not deterministic (e.g., Altman, 1999).

Stochastic Games (SGs) present a flexible model of dynamic conflict situations, with extensive theory and various applications in economics, operations research, and more. For recent overviews see Filar and Vrieze (1996) and Mertens (2002). In particular, existence of a state-independent value and stationary optimal policies is established for average-reward stochastic games under appropriate recurrence conditions, which are assumed in this paper. Stochastic games provide a natural generalization of the single-controller Markov decision problem to the multi-agent setting, and are highly relevant to various RL applications, starting with board games. The naive learning approach ignores possible non-stationarity of the opponent and simply applies standard RL algorithms as used for MDPs. This approach has shown some notable success (Baxter et al., 1998), but provides no performance guarantees. The seminal work of Tesauro (1996) is a notable example of an algorithm that learns under the assumption that the opponent play is stationary and adversarial, but provides no performance guarantees. Learning algorithms for zero-sum stochastic games with discounted rewards have been introduced by Littman (1994) and shown to converge to the max-min optimal policies (Littman and Szepesvári, 1999). Corresponding algorithms for the average reward problem have emerged only recently, and include model-based learning (Brafman and Tennenholtz, 2002), and Q-learning (Mannor and Shimkin, 2002).

Our main results are as follows. We present two learning algorithms for approaching a prescribed target set. The first (Multiple Directions Reinforcement Learning) algorithm is based on discretizing the set of possible steering directions, using a (dense) grid of direction vectors, and maintaining a learning algorithm for the stochastic games induced by each grid direction; the (scalar) reward function for each game is obtained by projecting the vector reward function on the corresponding direction. The second (Current Direction Reinforcement Learning) algorithm focuses on a single direction at a time, which is frozen for a long enough period so that both learning and steering are assured. Both algorithms are efficient, in the sense that they approach (to required accuracy) any target set that is approachable by steering policies when the model is known.

Equipped with the two basic algorithms, we consider the important extension to variable target sets. Here the target set to be approached is not fixed *a-priori*. Indeed, the target set of interest may depend on the model parameters, which are initially unknown. Our starting point here will be a *nested* family of target sets, with the goal of the learning agent naturally defined as approaching the smallest possible set in this family (note that approaching a smaller set automatically implies that any larger set in the nested family is also approached). We present a suitable learning algorithm which combines the basic algorithms with an on-line estimate of the minimally approachable target set. We will then show that this framework can be specialized to give a solution to the important problems of constrained SGs and MDPs.

We shall also briefly consider an alternative algorithm for approaching a (fixed) target set, which is based on the adversarial multi-armed bandit problem (e.g., Auer et al., 2002). This algorithm is suitable for the case where there is only a small set of sub-strategies that the controlling agent may alternate between. The idea here is to treat each of the possible strategies as an arm, and make sure that after enough rounds the reward is as high as that of the best arm (policy). Again, we show that the algorithm attains the same performance as in the known model case.

The paper is organized as follows. In Section 2 we describe the stochastic game model and formulate the control objective. In Section 3 we recall basic results from approachability theory, and prove that it suffices to use finitely many steering directions in order to approach a target set with a given precision. We also discuss RL for scalar-valued SGs and define two notions of optimality that will be used in the sequel. The next two sections describe the Multiple Directions RL algorithm and the Current Direction RL algorithm, while the bandit-based algorithm is described in Section 6. The extension to variable target sets is discussed in Section 7, which also discusses constrained problems. In Section 8 we briefly discuss the specialization of the previous results to (single-controller) MDPs with multiple objectives. Two examples are described in Section 9, followed by concluding remarks. An appendix contains a refined algorithm and its convergence proof.

2. Multi-Criterion Stochastic Games

In this section we present the multi-criterion SG model. We describe the learning agent's objective and define a related scalar projected games which will be useful in the sequel.

2.1 Model Definition

We consider a two-person average reward SG, with a vector-valued reward function. We refer to the players as P1 (the learning agent) and P2 (the arbitrary adversary). The game is defined by the following elements:

1. $\mathcal{S}$, a finite state space.

2. $\mathcal{A}$ and $\mathcal{B}$, finite sets of actions for P1 and P2, respectively. To streamline our notation it is assumed that P1 and P2 have the same action sets available in each state.

3. $P = P(s'|s,a,b)$, the state transition function.

4. $m : \mathcal{S} \times \mathcal{A} \times \mathcal{B} \to \mathbb{R}^k$, a vector-valued reward function.

At each time epoch $n \geq 0$, both players observe the current state s_n, and then P1 and P2 simultaneously choose actions a_n and b_n, respectively. As a result P1 receives the reward vector $m_n = m(s_n, a_n, b_n)$ and the next state is determined according to the transition probability $P(\cdot|s_n, a_n, b_n)$. More generally, we allow the actual reward m_n to be random, in which case $m(s_n, a_n, b_n)$ denotes its mean and we assume the reward is bounded.[1] We further assume that both players observe the previous rewards and actions (however, in some of the learning algorithms below, the assumption that P1 observes P2's action may be relaxed). A policy $\pi \in \Pi$ for P1 is a mapping which assigns to each possible observed history a mixed action in $\Delta(\mathcal{A})$, namely a probability vector over P1's action set $\mathcal{A}$. A policy $\sigma \in \Sigma$ for P2 is defined similarly. A policy of either player is called *stationary* if the mixed action it prescribes depends only on the current state s_n. The set of stationary policies of P1 is denoted by F, and that of P2 by G. Let $\hat{m}_n$ denote the average reward by time n:

$$\hat{m}_n \triangleq \frac{1}{n} \sum_{\tau=0}^{n-1} m_\tau \,.$$

Note that we do not define a reward for P2. Since we allow P2 to use an arbitrary policy, such reward function, even if it exists, is irrelevant to our formulation.

The following recurrence assumption will be employed. Let state s^* denote a specific reference state to which a return is guaranteed. We define the renewal time of state s^* as

$$\tau \triangleq \min\{n > 0 : s_n = s^*\}\,. \tag{1}$$

Assumption 1 (Recurrence) *There exist a state $s^* \in \mathcal{S}$ and a finite constant N such that*

$$\mathbb{E}^s_{\pi,\sigma}(\tau^2) < N \quad \text{for all } \pi \in \Pi, \sigma \in \Sigma \text{ and } s \in \mathcal{S},$$

where $\mathbb{E}^s_{\pi\sigma}$ is the expectation operator when starting from state $s_0 = s$ with policies π and σ for P1 and P2, respectively.

For finite state and action spaces this assumption is satisfied if state s^* is accessible from all other states under any pair of stationary deterministic policies (Shimkin and Shwartz, 1993). We note that Assumption 1 holds if the game is irreducible (as defined in Hoffman and Karp, 1966) or ergodic (as defined in Kearns and Singh, 1998).

Remark 1 *Assumption 1 may be relaxed in a similar manner to Mannor and Shimkin (2000). There, it is assumed that state s^* is* reachable*, that is, either player has a policy that guarantees a return to s^*. The approachability result that is obtained under this related assumption is somewhat more complicated, and we adhere here to Assumption 1 for simplicity.*

Remark 2 *In the absence of any recurrence conditions, tight approachability conditions and corresponding policies are extremely hard to obtain. This may be attributed to the dependence of the minimax value of such games (with scalar reward) on the initial state. Still, non-tight sufficient conditions and corresponding learning algorithms may be obtained by requiring the approachability conditions below to hold for all initial states. Again, we shall not pursue this direction here.*

1. Bounded reward is required essentially only for the SPRL algorithm from Section 6. This assumption can be relaxed for the algorithms of Sections 5,4, and 7 to requiring bounded second moments. The result in the appendix is also proved using the weaker assumption of bounded second moment.

2.2 P1's Objective

P1's task is to approach a target set T, namely to ensure convergence of the average reward vector to this set irrespectively of P2's actions. Formally, let $T \subset \mathbb{R}^k$ denote the target set. In the following, d is the Euclidean distance in $\mathbb{R}^k$. The set-to-point distance between a point x and a set T is $d(x,T) \stackrel{\triangle}{=} \inf_{y\in T} d(x,y)$. We let $P^s_{\pi,\sigma}$ denote the probability measure on the states and rewards sequences when P1 plays the policy π, P2 plays policy σ, and the initial state is s.

Definition 3 *A policy* π^* *of P1* approaches *a set* $T \subset \mathbb{R}^k$ *(from initial state s) if*

$$\lim_{n\to\infty} d(\hat{m}_n, T) = 0 \quad P^s_{\pi^*,\sigma}\text{-a.s., for every } \sigma \in \Sigma .$$

A policy $\sigma^* \in \Sigma$ *of P2* excludes *a set T (from initial state s) if for some* $\delta > 0$*,*

$$\liminf_{n\to\infty} d(\hat{m}_n, T) > \delta \quad P^s_{\pi,\sigma^*}\text{-a.s. for every } \pi \in \Pi ,$$

where a.s. stands for almost surely.

The policy π^* (σ^*) will be called an approaching (excluding) policy for P1 (P2). A set is approachable if there exists an approaching policy from all states. Noting that approaching a set and its topological closure are the same, we shall henceforth suppose that the set T is closed.

Remark 4 *The original definition of approachability by Blackwell (1956) required uniformity of convergence with respect to P2's policy. Some of our results do hold uniformly with respect to P2's policy and some do not. Specifically, the rate of convergence in the algorithms described in Sections 5 and 6 is uniform with respect to P2's policy. The convergence rate of the algorithm described in Section 4 is not uniform unless further assumptions are made. The issue of uniformity of convergence was not pursued here.*

Remark 5 *While the main focus in this paper is on the long-term average criterion, the target set concept does allow some consideration of instantaneous quantities. For example, recall the temperature regulation problems presented in the Introduction. Suppose that the instantaneous temperature (*in addition *to its average) is required to stay in some different range* $[T_{\min}, T_{\max}]$*. We can add a component to the reward vector, which indicates deviation from the required range: i.e., it takes the value '1' if the instantaneous temperature is outside this range, and '0' otherwise. We can now define the target set so that the long-term average of this component is required to approach 0, thereby assuring that the temperature stays in the required range save for a negligible fraction of times. Thus, both instantaneous quantities and their time averages can be considered in a unified framework.*

2.3 The Projected Scalar Game

Let u be a unit vector in the reward space $\mathbb{R}^k$. We often consider the *projected game in direction u* as the zero-sum stochastic game with the same dynamic as above, and *scalar* rewards $r_n \stackrel{\triangle}{=} m_n \cdot u$. Here "$\cdot$" stands for the standard inner product in $\mathbb{R}^k$. Denote this game by $\Gamma_s(u)$, where s is the initial state. The scalar stochastic game $\Gamma_s(u)$, has a *value*, denoted $\mathrm{v}\Gamma_s(u)$, if

$$\begin{aligned} \mathrm{v}\Gamma_s(u) &= \sup_\pi \inf_\sigma \liminf_{n\to\infty} \mathbb{E}^s_{\pi\sigma}(\hat{m}_n \cdot u) \\ &= \inf_\sigma \sup_\pi \limsup_{n\to\infty} \mathbb{E}^s_{\pi\sigma}(\hat{m}_n \cdot u) . \end{aligned} \tag{2}$$

The value is known to exist for any game with finite state and action spaces (Mertens and Neyman, 1981). Furthermore, under Assumption 1 the value is independent of the initial state and can be achieved in stationary policies (Filar and Vrieze, 1996). We henceforth simply write $\Gamma(u)$ for the average reward zero-sum game which reward is $r_n = m_n \cdot u$, and denote its value by $v\Gamma(u)$.

3. Preliminaries

In this section we recall the basic results of approachability theory for SGs. In Subsection 3.2 we extend approachability theory by considering approximate approachability. A target set is said to be *approximately approachable* if for every $\varepsilon > 0$ there exists a policy such that the distance between the average vector-valued reward vector and the target set is asymptotically less than ε. We prove that it suffices to consider finitely many steering directions in order to approximately approach a target set. We discuss relevant types of optimality of RL algorithms in scalar games in Subsection 3.3.

3.1 Approachability for Stochastic Games

We recall the basic results from Shimkin and Shwartz (1993) regarding approachability for known stochastic games, which generalize Blackwell‘s conditions for repeated matrix games. Let

$$\phi(\pi,\sigma) \triangleq \frac{\mathbb{E}^{s^*}_{\pi,\sigma}(\sum_{n=0}^{\tau-1} m_n)}{\mathbb{E}^{s^*}_{\pi,\sigma}(\tau)}$$

denote the *average per-cycle* reward vector, which is the expected total reward over the cycle that starts and ends in the reference state, divided by the expected duration of that cycle. For any $x \notin T$, denote by C_x a closest point in T to x, and let u_x be the unit vector in the direction of $C_x - x$, which points from x to the goal set T. The following theorem requires, geometrically, that there exists a policy $\pi(x)$ such that the set of all possible (vector-valued) expected rewards is on the other side of the hyperplane supported by C_x in direction u_x. See Figure 3 for an illustration, the polygon represents the set of all possible rewards when $\pi(x)$ is used.

Theorem 6 (Shimkin and Shwartz, 1993) *Let Assumption 1 hold. Suppose that for every point $x \notin T$ there exists a policy $\pi(x)$ such that*

$$(\phi(\pi(x),\sigma) - C_x) \cdot u_x \geq 0, \quad \forall\sigma \in \Sigma. \tag{3}$$

Then T is approachable by P1. An approaching policy is given as follows: If $s_n = s^*$ and $\hat{m}_n \notin T$, play $\pi(\hat{m}_n)$ until the next visit to state s^*; otherwise, play arbitrarily until the next return to s^*.

Intuitively, the condition in Equation (3) means that P1 can ensure, irrespectively of P2’s policy, that the average per-cycle reward will be on the other side (relative to x) of the hyperplane that passes through C_x and is perpendicular to the line segment that points from x to C_x. Note that this hyperplane actually separates T and x when T is convex. We shall refer to the direction u_x as the *steering direction* from point x, and to the policy $\pi(x)$ as the *directional steering policy* from x. The approaching policy uses the following rule: between successive visits to the reference state, a fixed (possibly stationary) policy is used. When in the reference state, the current average reward vector $\hat{m}_n$ is inspected. If this vector is not in T, then the steering policy that satisfies Equation (3) with

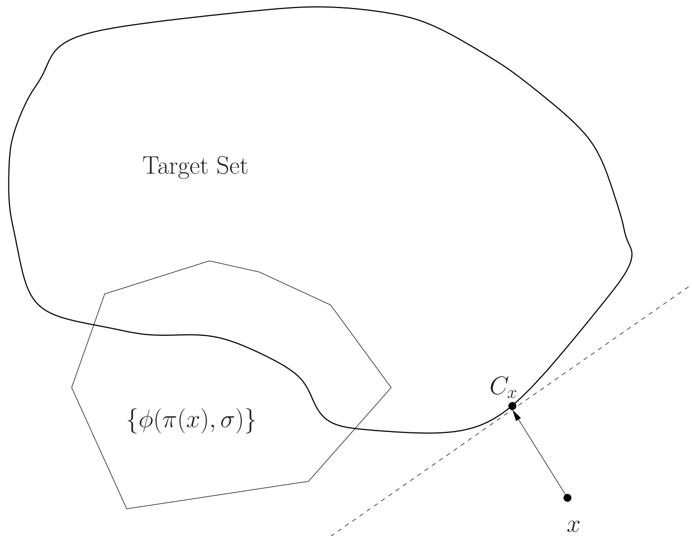

Figure 3: An illustration of the approachability condition. The location of the current average reward is x. The closest point to x in the target set is C_x. The polygon represents the set of all possible expected reward when $\pi(x)$ is played. It can be observed that $\pi(x)$ ensures that P1's reward will be on the other side of the hyperplane perpendicular to the line segment between C_x and x.

$x = \hat{m}_n$ is selected for the next cycle. As a consequence the average reward is "steered" towards T, and eventually converges to it.

Recalling the definition of the projected game in direction u and its value $\mathrm{v}\Gamma(u)$, the condition in Equation (3) may be equivalently stated as $\mathrm{v}\Gamma(u_x) \geq C_x \cdot u_x$. Furthermore, the policy $\pi(x)$ can always be chosen as the stationary policy which is optimal for P1 in the projected game $\Gamma(u_x)$. In particular, the directional steering policy $\pi(x)$ depends only on the corresponding steering direction u_x.

For *convex* target sets, the condition of the last theorem turns out to be both sufficient and necessary. Moreover, this condition may be expressed in a simpler form, which may be considered as a generalization of the minimax theorem for scalar games (Blackwell, 1956). Given a stationary policy $g \in G$ for P2, let $\Phi(F,g) \stackrel{\triangle}{=} \mathrm{co}(\{\phi(f,g)\}_{f\in F})$, where co is the convex hull operator. The Euclidean unit sphere in $\mathbb{R}^k$ is denoted by $\mathbb{B}^k$.

Theorem 7 (Shimkin and Shwartz, 1993) *Let Assumption 1 hold. Let T be a closed convex set in $\mathbb{R}^k$.*

(i) T is approachable if and only if $\Phi(F,g) \cap T \neq \emptyset$ for every stationary policy $g \in G$.

(ii) If T is not approachable then it is excludable by P2. In fact, any stationary policy g that violates (i) is an excluding policy.

(iii) T is approachable if and only if $\mathrm{v}\Gamma(u) \geq \inf_{m\in T} u \cdot m$ for every $u \in \mathbb{B}^k$.

3.2 Approachability with a Finite Set of Steering Directions

Standard approaching policies, as outlined above, may involve an infinite number of steering directions. The corresponding set of directional steering policies may turn out to be infinite as well. For the purpose of our learning scheme, we shall require an approaching policy which relies on a *finite* set of steering directions and policies. In this section we establish appropriate extensions of the results surveyed above. We prove that using a finitely many steering directions to approach a target set can indeed be done, possibly requiring to slightly expand the target set. Specifically, we establish that if the condition in Equation (3) is satisfied for the set T, then for any $\varepsilon > 0$ there exists a finite partition $\{U_1, U_2, \ldots, U_J\}$ of the unit sphere $\mathbb{B}^k$, and a corresponding set $\{\pi_1, \pi_2, \ldots, \pi_J\}$ of policies for P1, such that the required separation condition in Equation (3) may be satisfied within ε precision for any $u \in U_j$ by using the policy π_j. In the following, let M be an upper bound on the magnitude of the expected one-stage reward vector, so that $\|m(s,a,b)\| < M$ for all (s,a,b) ($\|\cdot\|$ denotes the Euclidean norm).

Proposition 8 *Let Assumption 1 hold and suppose that the target set $T \subset \mathbb{R}^k$ satisfies the condition in Equation (3). Fix $\varepsilon > 0$. Let $U = \{u_1, u_2, \ldots, u_J\}$ be a set of unit vectors which are the centers of open balls of radius $\varepsilon/2M$ that cover the unit sphere $\mathbb{B}^k$. Let π_j be the (stationary) policy which is optimal in the projected game $\Gamma(u_j)$. Then for any $x \notin T$, if the unit vector u_x is closest to u_j in U, then*

$$(\phi(\pi_j, \sigma) - C_x) \cdot u_x \geq -\varepsilon, \quad \forall \sigma \in \Sigma. \tag{4}$$

The cardinality J may be chosen so that $J \leq C(2M/\varepsilon)^k$, where C is a constant that depends only on the dimension k.

Proof Let $U = \{u_1, \ldots, u_J\}$ be the prescribed set of unit vectors. A standard sphere packing argument shows that J need not be larger than the stated upper bound (see, e.g., Anthony and Bartlett, 1999). Let π_j be an optimal policy for the projected game $\Gamma(u_j)$. By assumption, Equation (3) holds for any $x \notin T$ and associated unit vector u_x. Let u_j be the closest vector to u_x in U. Rewriting Equation (3) we have that

$$\begin{aligned} u_x \cdot C_x &\leq \inf_{\sigma} \{\phi(\pi(x), \sigma) \cdot u_x\} \\ &\leq \sup_{\pi} \inf_{\sigma} \{\phi(\pi, \sigma) \cdot u_x\} \\ &= \sup_{\pi} \inf_{\sigma} \left\{\phi(\pi, \sigma) \cdot (u_j + (u_x - u_j))\right\}. \end{aligned}$$

By construction, $\|u_j - u_x\| < \varepsilon/2M$ so that for every pair $\sigma \in \Sigma$ and $\pi \in \Pi$ we have that $\|\phi(\pi, \sigma) \cdot (u_i - u_x)\| \leq \varepsilon/2$.

$$\begin{aligned} u_x \cdot C_x &\leq \sup_{\pi} \inf_{\sigma} \left\{\phi(\pi, \sigma) \cdot u_j\right\} + \varepsilon/2 \\ &= \inf_{\sigma} \left\{\phi(\pi_j, \sigma) \cdot u_j\right\} + \varepsilon/2 \\ &\leq \inf_{\sigma} \left\{\phi(\pi_j, \sigma) \cdot u_x\right\} + \varepsilon. \end{aligned}$$

The optimality of π_j in $\Gamma(u_j)$ was used in the first equality and the bound on $\|u_j - u_x\|$ was used again in the last inequality. ∎

For a set $T \subset \mathbb{R}^k$ we define its ε-expansion, T^ε, as $T^\varepsilon \triangleq \{x \in \mathbb{R}^k : d(x, T) \leq \varepsilon\}$. It is now easy to verify that Equation (4) suffices for approaching T^ε, which leads to the following result.

Theorem 9 *Let Assumption 1 hold and suppose that T satisfies Equation (3). For any $\varepsilon > 0$ let $\{u_1,\ldots,u_J\}$ be the unit vectors in Proposition 8, with corresponding directional steering polices $\{\pi_1,\pi_2,\ldots,\pi_J\}$. Then the following policy approaches T^ε, the ε-expansion of T: If $s_n = s^*$ and $\hat{m}_n \notin T^\varepsilon$, then choose j so that $u_{\hat{m}_n}$ is closest to u_j (in Euclidean norm) and play π_j until the next visit to state s^*; otherwise, play arbitrarily.*

Proof We aim to show that the stated policy satisfies the requirements of Theorem 6 with respect to T^ε. That is, for every $x \notin T^\varepsilon$ the policy π_j which is optimal for the u_j which is closest to x satisfies

$$(\phi(\pi_j,\sigma) - C_x^\varepsilon)\cdot u_x^\varepsilon \geq 0\,, \quad \forall\sigma\in\Sigma\,, \tag{5}$$

where C_x^ε is the closest point in T^ε to x and u_x^ε is the unit vector from x to C_x^ε. We first claim that $C_x = C_x^\varepsilon + \varepsilon u_x^\varepsilon$, and hence $u_x = u_x^\varepsilon$. Indeed, if this is not true then there is a point z in T such that $d(z,x) < d(C_x^\varepsilon + \varepsilon u_x^\varepsilon, x) = d(C_x^\varepsilon, x) + \varepsilon$. Let u' denote the unit vector from x to z, we have that $z^\varepsilon = z - \varepsilon u'$ is in T^ε. It follows that $d(z^\varepsilon,x) = d(z,x) - \varepsilon$, combining this with the above, we get that $d(z^\varepsilon,x) < d(C_x^\varepsilon,x)$ which contradicts the assumption that C_x^ε is a closest point to x in T^ε.

Next, observe that by Proposition 8 we have that

$$(\phi(\pi_j,\sigma) - C_x)\cdot u_x \geq -\varepsilon\,, \quad \forall\sigma\in\Sigma\,,$$

but since $C_x = C_x^\varepsilon + \varepsilon u_x^\varepsilon$ and $u_x = u_x^\varepsilon$, (5) follows. ∎

Remark 10 *It follows immediately from the last theorem that the set T itself (rather than its ε-expansion) is approachable with a finite number of steering directions if $T^{-\varepsilon}$, the ε shrinkage of T satisfies Equation (3). Equivalently, T is required to satisfy Equation (3) with the 0 on the right-hand side replaced by ε.*

For convex target sets the sufficient condition Equation (3) for approachability is also necessary. This implies that every ε-expansion is approachable using a finite number of directions as per Theorem 9. The following corollary is an immediate result of the above theorem and Theorem 7.

Corollary 11 *Let Assumption 1 hold. A convex target set T is approachable if and only if for every $\varepsilon > 0$ the set T^ε can be approached by considering a finite number of directions as in Theorem 9.*

3.3 Reinforcement Learning in Scalar Games

In this subsection we review RL for zero-sum stochastic games with scalar rewards. We define two notions of approximate optimality in these games that will be useful in the sequel. The first type of optimality is optimality of the expected reward in finite time and the second type is asymptotic optimality of the learned policy. In both cases, the reference level for optimal performance is the value of the game. The first type of optimality requires that the average reward is not much less than the value of the game. The second type of optimality requires that the learned policy asymptotically achieves an expected reward which is not much worse than the value of the game. We note that a more ambitious goal may be to consider the Bayes envelope as in Mannor and Shimkin (2003) instead of the value, but this is not pursued here.

Consider a scalar zero-sum stochastic game. Denote the scalar reward at time n by r_n and the value by v (which is defined as in Equation (2)). Similarly to the vector-valued case, $\hat{r}_n$ denotes the average reward by time n. Our first notion of optimality is in expectation, after finitely many epochs.

Definition 12 *A learning algorithm* π *is* ε-optimal in expectation *from state s if there exists a deterministic finite time* $N(\varepsilon)$ *such that the expected average reward by time* $N(\varepsilon)$ *satisfies*

$$\mathbb{E}^{s}_{\pi,\sigma}\hat{r}_{N(\varepsilon)} \geq \mathrm{v} - \varepsilon\,, \tag{6}$$

for every policy σ *of P2.*

There are several algorithms that are ε-optimal in expectation. For example, by carefully choosing the parameters of the R–MAX algorithm of Brafman and Tennenholtz (2002) the optimality requirement in Equation (6) is satisfied. Indeed, Brafman and Tennenholtz (2002) suggested a model-based learning policy π of P1 that satisfies that for every $\varepsilon, \delta > 0$ there exists a time $N(\varepsilon, \delta)$ such that the expected average reward satisfies at time $N(\varepsilon, \delta)$ with probability at least $1 - \delta$

$$\mathbb{E}_{\pi,\sigma}\hat{r}_{N(\varepsilon,\delta)} \geq \mathrm{v} - \varepsilon\,,$$

for every policy σ of P2. By proper choice of parameters, the R–MAX algorithm yields an ε-optimal in expectation algorithm. This can be done by running the R–MAX algorithm with $\varepsilon' = \varepsilon/2$ and $\delta' = \varepsilon/2M$, where M is a bound on the expected immediate reward. Another algorithm that is ε-optimal in expectation is the Q-learning algorithm of Mannor and Shimkin (2002). An additional assumption that is required there is that P2 plays every action infinitely often (the analysis follows Kearns and Singh, 1999).

The next definition concerns the optimality of the learned policy and is of asymptotic nature. It implies that the learned policy of P1 is almost optimal in the limit. Formally, a policy of P1 is a mapping from all possible histories to the set of mixed actions of P1. Let π_n denote the policy of P1 from time n onwards (in the definition below this policy depends on the entire history up to time n, and the time axis is moved back n steps so that π_n starts at $t = 0$).

Definition 13 *A learning policy* π *of P1 is* asymptotically ε-optimal *in a scalar game if, for every policy* σ *of P2,* π_n *satisfies*

$$\liminf_{n\to\infty} \frac{\mathbb{E}^{s^*}_{\pi_n,\sigma}\sum_{t=0}^{\tau-1} r_t}{\mathbb{E}^{s^*}_{\pi_n,\sigma}\tau} \geq \mathrm{v} - \varepsilon \quad P_{\pi,\sigma}\text{-}a.s. \tag{7}$$

The renewal time τ *is as defined in Equation (1).*

It should be noted that the requirement in Equation (7) is on the policy π_n and not on the actual reward obtained. Intuitively, the condition means that after enough time P1 cannot be surprised anymore, namely no policy of P2 can decrease his (per cycle) payoff below the value of the game.

RL for average reward zero-sum stochastic games was devised along similar lines as RL for average reward Markov decision processes. A Q-learning based algorithm which combines the ideas of Littman (1994); Littman and Szepesvári (1999) with those of Abounadi et al. (2002) was devised by Mannor and Shimkin (2002). An additional assumption that was needed for the convergence analysis there is that all actions of both players are used infinitely often in order to satisfy Equation (7). A model-based algorithm that is ε-optimal asymptotically can be suggested. By assuming that all actions are played infinitely often a model of the stochastic game may be calculated and an ε-optimal policy played, similarly to the R-MAX algorithm of Brafman and Tennenholtz (2002). It should be noted that P2 may refrain from playing certain actions. Consequently, P1 may never observe these actions and never learn the exact model of the game. This implies that unless we impose some restrictions on P2's policy, the policy learned by P1 may be suboptimal. However, P1 can only benefit if P2 does not play certain actions, since in that case P2's play is constrained. The topic of learning to play optimally in SGs when no restrictions are made on P2 calls for further study.

In the special case of MDPs there are several algorithms that are ε-optimal in expectation (e.g., Kearns and Singh, 1998, 1999). Algorithms that are asymptotically optimal include ε-greedy Q-learning (Singh et al., 2000) and actor-critic (Konda and Tsitsiklis, 2001).

4. The Multiple Directions Algorithm

In this section we introduce and prove the convergence of the MDRL (Multiple-Directions Reinforcement Learning) algorithm. We consider the model of Section 2, but here we assume that P1, the learning agent, does not know the model parameters, namely the state transition probabilities and reward functions. A policy of P1 that does not rely on prior knowledge of these parameters will be referred to as a *learning* policy.

The proposed learning algorithm relies on the construction of the previous section of approaching policies with a finite number of steering directions. The main idea is to apply a (scalar) learning algorithm for each of the projected games $\Gamma(u_j)$ corresponding to these directions. Recall that each such game is a standard zero-sum stochastic game with average reward. The required learning algorithm for game $\Gamma(u)$ should secure an average reward that is not less than the value $v\Gamma(u)$ of that game.

We now describe the MDRL algorithm that approximately approaches any target set T that satisfies Equation (3). The parameters of the algorithm are ε and M, where ε is the approximation level and M is a known bound on the norm of the expected reward per step. The goal of the algorithm is to approach T^{ε}, the ε expansion of T. There are J learning algorithms that are run in parallel, denoted by $\pi_1,\ldots\pi_J$. Each of the J algorithms is responsible for learning to steer the reward in a different direction. Each of the J algorithms is assumed to be an asymptotically $\varepsilon/2$-optimal learning algorithm. The MDRL is described in Figure 4 and is given here as a meta-algorithm (the scalar RL algorithms π_i are not specified).

The algorithm is based on the idea of changing the control policy on arrivals to the reference state s^*. When arriving to s^*, the learning agent checks if the average reward vector is outside the set T^{ε}. In that case, it switches to an appropriate policy that is intended to "steer" the average reward vector towards the target set. The steering policy (π_j) is chosen so that the steered direction (u_j) is close to the exact direction towards T. The intuition behind the algorithm and the proof is that instead of steering in the current direction into T, a "close" direction, quantized out of a finite set is used. By employing the same quantized directions often enough, the decision maker eventually learns how to steer in those directions. Once the steering policy in each of the quantized directions is learned, the decision maker can almost steer in the desired direction at every return to s^*, by choosing a close enough quantized steering direction.

Theorem 14 *Suppose that Assumption 1 holds and that the target set satisfies Equation (3). Then the MDRL algorithm approaches T^{ε}.*

Proof Using the construction of Theorem 9 there are finitely many directions $u_1,\ldots,u_J$ that induce J projected SGs. At every return to state s^*, one of the J projected games is played and algorithm π_i is run. If the i-th game is not played infinitely often then there exists τ_i after which the i-th

Input: Target set T; desired approximation level ε; a bound M on the expected one step reward; a recurrent state s^*.
0. Divide the unit vector ball $\mathbb{B}^k$ as in Proposition 8 for approaching the $\varepsilon/2$ expansion of T. Let $u_1, \ldots u_J$ be the chosen directions there ($J = C(4M/\varepsilon)^k$). Initialize J different $\varepsilon/2$-optimal asymptotically scalar algorithms, $\pi_1, \ldots, \pi_J$. Set $n = 0$. 1. If $s_n \neq s^*$ play arbitrarily until $s_n = s^*$. 2. ($s_n = s^*$) If $\hat{m}_n \in T^\varepsilon$ goto step 1. Else let $i = \arg\min_{1 \leq i \leq J} \|u_i - u_{\hat{m}_n}\|_2$. 3. While $s_n \neq s^*$ play according to π_i, the reward π_i receives is $u_i \cdot m_n$. 4. When $s_n = s^*$ goto step 2.

Figure 4: The MDRL algorithm.

algorithm is not played. Let τ^1 be the maximum of all the times τ_i for i-s that were played finitely often. Let $I \subseteq \{1, \ldots, J\}$ be the indices of algorithms that are played infinitely often. Since each π_i is eventually almost optimal it follows that after some finite time $\tau'(\varepsilon/2)_i$ each of the algorithms almost achieves the maximal reward, that is for $t > \tau'(\varepsilon/2)_i$ (7) holds with $\varepsilon/2$ instead of ε. Let $\tau^2 = \max_{i \in I} \tau'(\varepsilon/2)_i$. Let $\tau^3 = \max\{\tau^1, \tau^2\}$. From time τ^3 onward P1's play is $\varepsilon/2$ optimal. Thus, we can apply Theorem 9 so that T^ε is approached starting from τ^3. Fix $\delta > 0$. Let τ^4 be a large enough time such that the effect of the first τ^3 time epochs is not more δ in norm, i.e., $|\frac{1}{\tau^4} \sum_{n=0}^{\tau^3 - 1} m_n| < \delta$. Since δ is arbitrary, by Theorem 9 the result follows. ∎

Remark 15 *In the algorithm below, when π_j is not played, its learning pauses and the process history during that time is ignored. Note however that some "off-policy" algorithms (such as Q-learning) can learn the optimal policy even while playing a different policy. In that case, a more efficient version of the MDRL may be suggested, in which learning is performed by all learning policies π_j continuously and concurrently.*

5. The Current Direction Learning Algorithm

In this section we introduce and prove the convergence of the CDRL (Current Direction Reinforcement Learning) algorithm. P1's goal is the same as in Section 4 - to approach a given target set T. P1 does not know the model parameters and aims to ensure the convergence of the average reward vector to this set irrespectively of P2's actions.

Instead of using several fixed steering directions while employing a scalar learning algorithm for each of these directions separately, a single learning algorithm is used with the reward function changing over time. The idea is to learn to play a scalar game whose reward is the projected average reward on the direction towards the target set. Once a direction is fixed, the policy is to steer in that direction towards the target set for a while. After a long enough time, the direction is replaced by the new steering direction into the target set, and a new scalar learning algorithm is initiated.

We now describe the CDRL algorithm for approaching any target set T that satisfies Equation (3). The CDRL is described in Figure 5 and is given here as a meta-algorithm. The algorithm is based on the idea of changing the control policy after a long enough time. Whenever a switch occurs the new policy is intended to "steer" the average reward vector towards the target set.

Input: Target set T; Required precision ε; ε-optimal learning algorithm; $N(\varepsilon)$ from Definition 12.
0. Set $\hat{m}_0$ arbitrarily; $n = 0$. 1. Repeat forever: 2. If $\hat{m}_n \in T^\varepsilon$ play arbitrarily for $N(\varepsilon)$ steps; $n := n + N(\varepsilon)$. 3. Else let $u = u_{\hat{m}_n}$. 4. Play an ε-optimal in expectation learning algorithm with reward at time n given by $r_n = u \cdot m_n$ and parameter ε. Play the algorithm for $N(\varepsilon)$ time epochs; $n := n + N(\varepsilon)$.

Figure 5: The CDRL algorithm.

Theorem 16 *Fix $\varepsilon > 0$. Suppose that for every $u \in \mathbb{B}^k$ there exists an ε-optimal in expectation algorithm for each projected SG, $\Gamma(u)$. Assume that the target set satisfies Equation (3). Then T^ε is approached using the CDRL algorithm.*

Proof (outline) The proof follows similarly to the more general Theorem 26 that is provided in the appendix. Consider the time epochs $\tau_i = iN(\varepsilon)$, $i = 1, 2, \ldots$. If $\hat{m}_{\tau_i} \notin T^\varepsilon$ it follows that

$$\mathbb{E}\left(\left(\frac{1}{N(\varepsilon)}\sum_{n=\tau_i}^{\tau_{i+1}-1}(m_n - C_{\hat{m}_{\tau_i}})\right)\cdot u_{\hat{m}_{\tau_i}}\Big|F_{\tau_i}\right) \geq -\varepsilon\,,$$

where F_t is the sigma algebra generated until time t. Consequently,

$$\mathbb{E}\left(\left(\frac{1}{N(\varepsilon)}\sum_{n=\tau_i}^{\tau_{i+1}-1}(m_n - C^\varepsilon_{\hat{m}_{\tau_i}})\right)\cdot u_{\hat{m}_{\tau_i}}\Big|F_{\tau_i}\right) \geq 0\,,$$

where C^ε_x is the closest point to x in T^ε. A similar argument to that of Theorem 26 can be used to show that $d(\hat{m}_{\tau_i}, T^\varepsilon) \to 0$ (a.s.). The result for all t follows by the boundedness of the immediate reward. ∎

Remark 17 *The CDRL algorithm was described above in extreme generality. There are several possible refinements and improvements which depend on the specific scalar algorithm used and on the domain of the problem. Specifically, knowledge is not assumed to be preserved from one activation of the scalar algorithm to the other. One may be interested to transfer knowledge between activations. This is especially natural for model-based algorithm in the spirit of Brafman and Tennenholtz (2002), where estimation of the model is performed.*

Remark 18 *A variant of the CDRL algorithm in which the precision level ε is decreased with time is described in the Appendix. It is shown there that ε can be slowly reduced to 0 so that the target set T itself, rather than its ε-expansion, is approached.*

6. Policy Mixing

Up to now we did not impose any restrictions on the directional steering policies that P1 can employ. Suppose now that P1 may choose between a finite and given set of policies $\pi_1, \ldots, \pi_J$. For simplicity we may restrict the policy changes to occur in renewal epochs to some reference state s^*. To motivate this problem, one may consider an agent who has a few pre-programmed control policies. These policies may have been found useful for different tasks in this environment or are just intelligent guesses of reasonable policies. The agent wants to mix the policies in such a way so that the target set is approached. The idea is to mix between the prescribed policies so that when steering in a direction u the best policy among the J available policies is used. To choose the best such policy, a Multi-Armed Bandit (MAB) algorithm is employed. We describe an algorithm in the spirit of CDRL which we call SPRL (Specified Policies Reinforcement Learning).

If P1 knew the model parameters, P1 could choose the policy that is best in Equation (3) in the current steering direction (out of the set of given policies). In that case it follows from Theorem 6 that the set T is approachable if for every point $x \notin T$ there exists a policy π_i, $1 \leq i \leq J$, such that

$$(\phi(\pi_i, \sigma) - C_x) \cdot u_x \geq 0\,, \quad \forall \sigma \in \Sigma. \tag{8}$$

In that case we say that T is approachable with $\pi_1, \ldots, \pi_J$.

We set out to show that if T is approachable with $\pi_1, \ldots, \pi_J$ when the model is known then it is approachable with $\pi_1, \ldots, \pi_J$ when the model is unknown. The following algorithm resembles the CDRL algorithm in that every once in a while P1 changes the steering direction to best approach T. In order to steer in the current direction we use appropriate algorithms for the adversarial MAB problem, such as the Exp3.1 algorithm of Auer et al. (2002), as a policy selection mechanism. Recall that in the adversarial MAB problem a decision maker is repeatedly required to decide between J arms. When choosing arm i at time epoch n a reward of r_n^i is obtained. The decision maker tries to make sure that its cumulative reward is close to the one obtained at the best arm. We assume that a MAB arm selection algorithm, π, is defined by the following operations:

1. $Init_{\mathrm{MAB}}(J)$ - Initialize the parameters of MAB with J arms.
2. $GetArm_{\mathrm{MAB}}()$ - Returns the arm a that MAB wants to sample next.
3. $Update_{\mathrm{MAB}}(i, r)$ - Informs MAB the latest reward r of arm i.

The dynamics of every algorithm for the MAB problem are described as follows. First, each MAB algorithm is initialized. At every time epoch n, $GetArm_{\mathrm{MAB}}$ is used for picking an arm i_n and a reward $r_n = r_n^{i_n}$ is obtained. $Update_{\mathrm{MAB}}(i_n, r_n)$ is used to update the algorithm with the knowledge of the obtained reward. In the adversarial MAB formulation, there are two players - P1 and P2. P1 chooses which arm to play and simultaneously (not knowing P1's choice) P2 chooses how much reward to give to P1 for every possible choice of P1. The basic requirement from a MAB algorithm is that for every $\varepsilon > 0$ there exists $N(\varepsilon)$ such that

$$\mathbb{E}_{\pi,\sigma} \frac{1}{N(\varepsilon)} \sum_{n=0}^{N(\varepsilon)-1} r_n \geq \max_{1 \leq j \leq J} \sum_{n=0}^{N(\varepsilon)-1} r_n^j - \varepsilon \quad \text{for every policy } \sigma \text{ of P2}, \tag{9}$$

where the expectation is taken over both player's policies, as the resulting policy (for both players) may be stochastic. Equation (9) means that the expected average reward by time $N(\varepsilon)$ is almost as high as the average reward that would have been obtained if P1 would have known the reward of each arm in advance. Note that it is assumed that P2 decides on P1's reward for every action (arm) chosen at every time epoch. The Exp3.1 algorithm satisfies Equation (9). Moreover, $N(\varepsilon)$ depends only on ε, J and a bound on the immediate reward.

To state the SPRL algorithm we shall require the concept of ε-*mixing time* (see Brafman and Tennenholtz, 2002). The ε-mixing time K is a (large enough) number so that for every $1 \leq i \leq J$, every policy σ of P2, and every initial state s,

$$\mathbb{E}^{s}_{\pi_i,\sigma} \frac{1}{K} \sum_{n=0}^{K-1} r_n \geq \inf_{\sigma'} \frac{\mathbb{E}^{s^*}_{\pi_i,\sigma'} \sum_{t=0}^{\tau-1} r_t}{\mathbb{E}^{s^*}_{\pi_i,\sigma'} \tau} - \varepsilon,$$

where τ is the stopping time defined in Equation (1). Having an ε-mixing time implies that the expected reward starting from any initial state is at least as high (up to ε) as the worst case average reward starting from s^*. It follows by Lemma 3.4 from Shimkin and Shwartz (1993) that when Assumption 1 holds then there exists K_0 such that the expected return time to s^* from every state and every pair of policies is bounded by some K_0. By choosing $K \geq K_0 M/\varepsilon$ the above condition is satisfied (M is a bound on the expected one stage reward). The basic requirement of the SPRL algorithm is that K, or an upper bound on it, is provided.

Input: Required precision ε; set of policies $\pi_1, \ldots, \pi_J$; $\varepsilon/2$-mixing time K;
  a recurrent state s^*.

0. $n = 0$; play arbitrarily until $s_n = s^*$.
1. Repeat forever
2. If $\hat{m}_n \in T^\varepsilon$ play arbitrarily for K steps.
3. Else let $u = u_{\hat{m}_n}$.
4. $Init_{MAB}(J)$.
5. Repeat $N(\varepsilon/2)$ times:
6. $i = GetArm_{MAB}$.
7. Play π_i for K epochs, let the total reward over the K epochs be m_{tot}, set $r = m_{tot} \cdot u$; $n := n + N(\varepsilon/2)$.
8. $Update_{MAB}(i, r)$.

Figure 6: The SPRL algorithm - switching between a given small set of policies.

Theorem 19 *Suppose that T is approachable with $\pi_1, \ldots \pi_J$ when the model is known. Then the SPRL algorithm described in Figure 6 approaches T^ε.*

Proof The proof is based on showing that by using the SPRL algorithm Equation (3) holds for the expanded target set T^ε. Consider the reward in the epochs $\tau_i = iKN$ $(i = 1, 2, \ldots)$, for $N = N(\varepsilon/2)$. We claim that the vector valued reward at those points satisfies Blackwell's condition. By using the SPRL algorithm it follows that either $\hat{m}_{\tau_i} \in T^\varepsilon$ or that

$$\mathbb{E}\left(\left(\frac{\sum_{n=\tau_i}^{\tau_{i+1}-1} m_n}{KN} - C_{\hat{m}_{\tau_i}} \right) \cdot u_{\hat{m}_{\tau_i}} \middle| F_{\tau_i} \right) \geq -\varepsilon \quad \text{a.s.}, \tag{10}$$

where $C_{\hat{m}_{\tau_i}}$ is the closest point to $\hat{m}_{\tau_i}$ in T and F_n is the sigma algebra up to time n. Fix i and let $r_n \stackrel{\triangle}{=} (m_n - C_{\hat{m}_{\tau_i}}) \cdot u_{\hat{m}_{\tau_i}}$ for $\tau_i \leq n \leq \tau_{i+1} - 1$. By our choice of playing an $\varepsilon/2$ MAB algorithm it follows that

$$\mathbb{E}_{\pi,\sigma}^{s_{\tau_i}}\left(\frac{1}{KN}\sum_{n=\tau_i}^{\tau_{i+1}-1} r_n\right) = \mathbb{E}_{\pi,\sigma}^{s_{\tau_i}}\left(\frac{1}{N}\sum_{\ell=0}^{N-1}\frac{1}{K}\sum_{n=\tau_i+\ell K}^{\tau_i+(\ell+1)K-1} r_n\right)$$
$$\geq \max_{1\leq j\leq J}\frac{1}{N}\sum_{\ell=0}^{N-1}\frac{1}{K}\mathbb{E}_{\pi_j,\sigma}^{s_{\tau_i+\ell K}}\left(\sum_{n=\tau_i+\ell K}^{\tau_i+(\ell+1)K-1} r_n\right) - \varepsilon/2\,. \tag{11}$$

Since K is an $\varepsilon/2$-mixing time we have that for every $1 \leq j \leq J$ and $1 \leq \ell < N$,

$$\mathbb{E}_{\pi_j,\sigma}^{s_{\tau_i+\ell K}}\frac{1}{K}\sum_{n=\tau_i+\ell K}^{\tau_i+(\ell+1)K-1} r_n \geq \inf_{\sigma'}\frac{\mathbb{E}_{\pi_j,\sigma'}^{s^*}\sum_{t=0}^{\tau-1} r_t}{\mathbb{E}_{\pi_i,\sigma'}^{s^*}\tau} - \varepsilon/2\,.$$

Since this holds for every $1 \leq j \leq J$, we substitute this in Equation (11) to obtain

$$\mathbb{E}_{\pi,\sigma}^{s_{\tau_i}}\left(\frac{1}{KN}\sum_{n=\tau_i}^{\tau_{i+1}-1} r_n\right) \geq \max_{1\leq j\leq J}\inf_{\sigma'}\frac{\mathbb{E}_{\pi_j,\sigma'}^{s^*}\sum_{t=0}^{\tau-1} r_t}{\mathbb{E}_{\pi_i,\sigma'}^{s^*}\tau} - \varepsilon\,.$$

Using the definitions of r_n and $C_{\hat{m}_{\tau_i}}$, Equation (10) follows. The rest of the proof follows almost exactly as Theorem 16 and is therefore omitted. ∎

Several comments are in order.

Remark 20 *We note that even if π_i are learning policies rather than a-priori fixed ones the algorithm still works. The only thing that matters in this case is that the strategies converge after some finite stochastic time. This suggests the following algorithm. Learn the optimal policy for several directions, and use the SPRL algorithm for mixing the learned strategies.*

Remark 21 *It is possible to obtain Theorem 19 with $\varepsilon = 0$. This can be done by a limiting procedure similar to Theorem 26 with the additional complication that both the ε-mixing time and the time required by the MAB algorithm increase. It can be shown that for certain MAB algorithms $N(\varepsilon) \sim \frac{1}{\varepsilon^2}$ and that the behavior of the mixing time as a function of ε is $K(\varepsilon) \sim \frac{1}{\varepsilon}$. The total time is approximately $O(\frac{1}{\varepsilon^3})$, and the technique of Theorem 26 may be employed.*

7. On-line Selection of Target Sets

In many problems of interest the target set that we wish to approach may itself depend on the unknown model parameters, so that it cannot be explicitly prescribed *a-priori* to the learning agent. The simplest and most common instance of this situation is the simple maximization of a scalar objective function: this corresponds to a target set of the form $[a,\infty)$, with a required to be as large as possible. We generalize this situation to the multi-objective case by considering a *nested* family of possible target sets, and assume that the agent's task it to approach the minimal target set, namely the smallest possible one for the actual model at hand. The major difficulty in applying the steering approach now is that the required steering direction is not known, as it may be different for each set in the given family. We address this difficulty by maintaining an estimate of the smallest set that can be approached, which is used as a target for steering, and updating this estimate according to observations. In the latter part of this section we apply these results to solve the problem of learning optimal policies in constrained stochastic games. Throughout this section we assume that all target sets are convex.

7.1 Nested Target Sets

Consider an increasing class of convex sets $\{T_\alpha\}_{\alpha\in\mathcal{L}}$, indexed by the ordered set of indices $\mathcal{L}$ such that $\alpha < \beta$ implies $T_\alpha \subseteq T_\beta$. For simplicity, we may consider $\mathcal{L} = \mathbb{R}$ and assume that $\{T_\alpha\}$ is continuous.[2] The objective is to approach the smallest possible set in this class, that is to approach the smallest possible T_α. Recall that the third part of Theorem 7 claims that a convex target set is approachable if and only if for every $u \in \mathbb{B}^k$, $\mathrm{v}\Gamma(u) \geq \inf_{m\in T} u\cdot m$. Consider a direction u, and let $\mathrm{v}(u) = \mathrm{v}\Gamma(u)$ denote the value of the game in this direction. Let $\alpha(u, \mathrm{v}(u))$ denote the smallest (or infimal) value α for which $\mathrm{v}(u) \geq \inf_{m\in T_\alpha} u\cdot m$. (Note that we define $\alpha(u, \mathrm{v}(u))$ for any value $\mathrm{v}(u)$, since below we shall use an estimate to $\mathrm{v}\Gamma(u)$ rather than the exact value). Then, according to Theorem 7 the set T_α may be approachable only if $\alpha \geq \alpha(u, \mathrm{v}\Gamma(\alpha))$, and the smallest approachable set is

$$T^* = \bigcap_{u\in\mathbb{B}^k} T_{\alpha(u,\mathrm{v}\Gamma(u))} = T_{\alpha^*}\,,$$

where $\alpha^* = \sup_{u\in\mathbb{B}} \alpha(u, \mathrm{v}\Gamma(u))$. See Figure 7 for a geometric illustration of the problem.

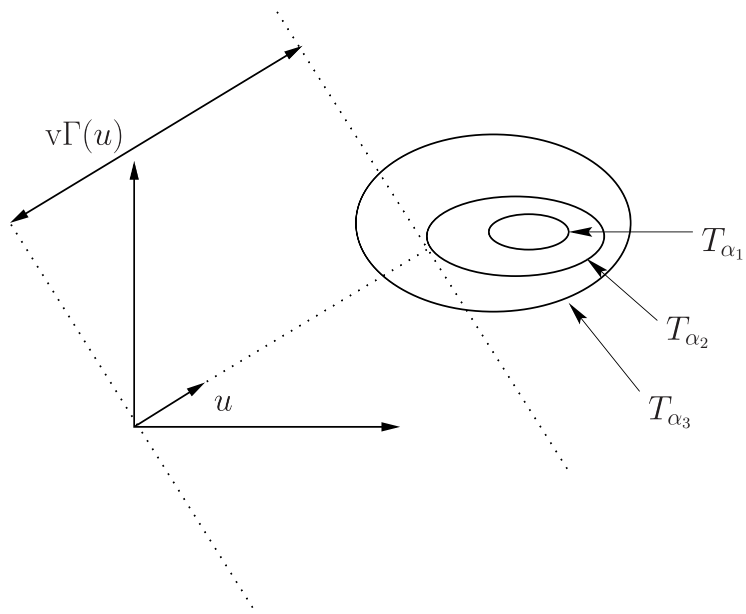

Figure 7: A two dimensional Illustration of nested sets. In the illustration $\alpha_1 < \alpha_2 < \alpha_3$ so that $T_{\alpha_1} \subset T_{\alpha_2} \subset T_{\alpha_3}$. For the direction u the value of the projected game $\Gamma(u)$ is such that $\alpha(u, \mathrm{v}\Gamma(u)) = \alpha_2$. As a result a target T_α set may be approachable if $\alpha \geq \alpha_2$. In this case T_{α_3} may be approachable, but T_{α_1} is not approachable.

We now suggest a modification of the MDRL algorithm for approximately approaching T^*. Recall that the MDRL algorithm is based on switching between a grid of J steering directions $u_1,\ldots,u_J$. For each direction u_j an asymptotically ε-optimal learning algorithm π_j is applied. In

2. Continuity here is induced by the metric, implying that the distance between sets is small when the indices are close. Formally, we require that if $\alpha_k \to \alpha$ then $\sup_{x\in T_\alpha\cup T_{\alpha_k}} \inf_{y\in T_\alpha\cap T_{\alpha_k}} d(x,y) \to 0$.

the original MDRL algorithm some directions may be encountered finitely many times so that the learned directional steering policy may be suboptimal. Since T is not prescribed and should be estimated on-line, we will impose exploration in each of the J steering directions. Fix a direction u_j. Let $\tilde{\mathrm{v}}(u_j)_n$ be the estimate by time n of $\mathrm{v}\Gamma(u_j)$. We require that the estimate is not too pessimistic. That is,

$$\liminf_{n\to\infty} \tilde{\mathrm{v}}(u_j)_n \geq \mathrm{v}\Gamma(u_j) - \varepsilon \quad P_{\pi_j,\sigma} - \text{a.s}\,, \tag{12}$$

where π_j is the learning policy when steering in direction u_j, σ is P2's policy, and Equation (12) holds for all σ. We also require that for every π_j,

$$\lim_{n\to\infty} \tilde{\mathrm{v}}(u_j)_n \quad \text{exists } P_{\pi_j,\sigma} - \text{a.s.} \tag{13}$$

This requirement is not essential, but simplifies the analysis and is fulfilled by all the learning algorithms we are aware of. We also assume that the estimate of the value is not too optimistic and matches P2's policy in the following sense:

$$\liminf_{n\to\infty} \left(\frac{\mathbb{E}^{s^*}_{\pi_n,\sigma} \sum_{t=0}^{\tau-1} r_t}{\mathbb{E}^{s^*}_{\pi_n,\sigma} \tau} - \tilde{\mathrm{v}}(u_j)_n \right) \geq -\varepsilon \quad P_{\pi_j,\sigma}\text{-a.s}\,, \tag{14}$$

for every policy σ of P2 (τ is the renewal time from Equation (1)). Intuitively, Equation (14) means that P1's policy guarantees an expected reward which is not much worse than the estimated value $\tilde{\mathrm{v}}(u_j)_n$. Without Equation (14), the target set may not be estimated correctly, and the steering direction may be wrong. All three assumptions Equation (12)-(14) are satisfied by Q-learning for average reward games of Mannor and Shimkin (2002) under mild additional assumptions on P2's play (the assumption there was that every action is either played a non vanishing fraction of the times or finitely many times, but the fraction of times may not alternate between 0 and something bigger than 0). The above requirements may also be seen to be satisfied by model-based learning algorithms under the assumption that all actions by P2 are played infinitely often. We note that $\tilde{\mathrm{v}}(u_j)_n$ in Equation (12) may be larger than the true value $\mathrm{v}\Gamma(u_j)$. The reason for that is that $\tilde{\mathrm{v}}(u_j)$ is an estimate which is based on the observations. If P2 refrains from playing certain actions (which are worst-case in $\Gamma(u_j)$) then P1 cannot, and need not, take these action into consideration.

In Figure 8 we describe a modified version of the MDRL algorithm. Essentially we add to the MDRL algorithm a small probability of exploration on each return to the reference state s^*. Exploration is made uniformly over the J available directions. For each direction u_i (all directions are chosen comparatively often due to the exploration) an index is maintained for the value of the projected game. This index represents the current estimate of $\alpha(u, \mathrm{v}\Gamma(u))$. The goal is to approach an "almost" minimal set. The steps of the algorithm are very similar to the MDRL algorithm from Figure 4. The differences are the persistent exploration that is performed continually (step 2 of the algorithm) and the fact that the estimated target set replaces the true (unknown) target set. Exploration is made when either $\hat{m}_n$ is inside the estimated target set (and therefore the decision maker does not need to steer in any direction), or with some small probability when returning to s^*. The latter exploration is needed in order to guarantee that the estimate of the target set is eventually consistent. The estimate of the target set is based on estimating the index that each of the J projected games and taking the maximal index to be the current estimate of the index of the target set. We wish to approach the set T_α with minimal index that is approachable. The estimate of the index by time n is denoted by $indmax_n$ and equals

$$indmax_n \stackrel{\triangle}{=} \max_{1\leq i\leq J} \alpha(u_i, \tilde{v}(u_i)_n).$$

We take the maximal index (over the J available directions), because if $\alpha < \alpha(u_i, \tilde{v}(u_i)_n)$ for some i it implies that, according to the current estimate, T_α is not approachable for $\alpha < \alpha(u_i, \tilde{v}(u_i)_n)$. The estimate of the target set by time n is

$$\tilde{T}_n = T_{indmax_n}. \tag{15}$$

For every $\varepsilon > 0$ define the maximal index $indmax(\varepsilon) \stackrel{\triangle}{=} \max_{1\leq i\leq J} \alpha(u_i, v\Gamma(u_i) - \varepsilon)$. The target set P1 tries to approach is $T_{indmax(\varepsilon)}$. The modified MDRL algorithm from Figure 8 approaches the set $T^\varepsilon_{indmax(\varepsilon)}$, which is the ε-expansion of $T_{indmax(\varepsilon)}$.

Input: Desired approximation level ε; a set class $\{T_\alpha\}_{\alpha\in\mathcal{L}}$; a bound M on the expected one step reward; a bound N on the expected renewal time to s^*.
0. Divide the unit vector ball $\mathbb{B}^k$ as in Proposition 8 for approaching the $\varepsilon/3$ expansion of any target set. Let $U = u_1, \ldots, u_J$ be the chosen directions. Initialize J different asymptotically $\varepsilon/3$-optimal scalar algorithms, $\pi_1, \ldots, \pi_J$ that satisfy Equation (14). 1. Choose a random direction $u_i \in U$ and play π_i until $s_n = s^*$. 2. ($s_n = s^*$) If $\hat{m}_n \in \tilde{T}_n$ goto step 1. Else with probability $\delta = \varepsilon/(3MN)$ goto 1. 3. Let $u = \frac{\hat{m}_n - C(\tilde{T}_n, \hat{m}_n)}{\|\hat{m}_n - C(\tilde{T}_n)\|}$. ($C(T,x)$ denotes the closest point in T to x.) 4. Let $i = \text{argmin}_{1\leq i\leq J} \|u - u_i\|$. 5. While $s_n \neq s^*$ play according to π_i, the reward π_i receives is $u_i \cdot m_n$. 6. When $s_n = s^*$ goto step 2.

Figure 8: The modified MDRL for unprescribed target sets.

The following Theorem claims that $T^\varepsilon_{indmax(\varepsilon)}$ is approached using the modified MDRL. Since we assume that the class $\{T_\alpha\}$ is continuous in α, by taking ε small enough we can practically get as close as we wish to T.

Theorem 22 *Suppose that Assumption 1 holds and that the target set satisfies Equation (3). Further assume that each scalar learning algorithm satisfies Eqs. (12), (13), and (14), where Equation (12) and Equation (14) are satisfied with $\varepsilon/3$ instead of ε. Then $T^\varepsilon_{indmax(\varepsilon)}$ is approached using the modified MDRL algorithm from Figure 8.*

Proof (outline) Exploration is persistent in all directions. By the definition of the algorithm, exploration cannot cause more that $\varepsilon/3$ drift away from the target set. Since each algorithm satisfies Equation (14) and since the estimate $\tilde{v}(u_j)_n$ converges to a limit by Equation (13) it follows that from some (random) time on P2's expected reward when steering in direction j is up to $\varepsilon/3$ as high as $\lim_{n\to\infty} \tilde{v}(u_j)_n$. By Equation (12) and Equation (13) the limit of $\tilde{v}(u_j)_n$ is not less than $v\Gamma(u_j) - \varepsilon/3$. It follows that from some (random) time on the estimate of $\tilde{T}_n$ almost does not change (more specifically, we assume that Equation (13) holds so that $\tilde{v}(u_j)_n$ converges. From some random time on, all the J estimates $\tilde{\mathrm{v}}(u_j)_n$ converge. By the continuity of T_α in α, it follows that the set $\tilde{T}_n$ converges to some, random, $\tilde{T}_\infty$). After that time - the same analysis as in Theorem 14 may be performed. ∎

There is a special case in which there is no need to perform the exploration for estimating $\tilde{v}(u_i)_n$. This is the case where the sets in the target set class are balls around a target point. Namely, if there is some point x_0 such that for every α in $\mathcal{L}$ there is a number γ such that T_α is a ball with radius γ around x_0. The steering directions is always towards x_0 and is the same for all the candidate target sets. Consequently, the MDRL algorithm does not require any changes, and exploration is not required.

7.2 Constrained Stochastic Games

A specific case of interest is the case of constrained optimization which has been extensively explored for MDPs (see Altman, 1999, and references therein). In this setup the decision maker obtains in every time step both reward and additional variables. The objective of the decision maker is to maximize its average reward $\hat{r}_n$ given that some additional average variables $\hat{c}_n^1, \ldots, \hat{c}_n^k$ satisfy the constraints $\hat{c}_n^i \leq \bar{c}^i, i = 1, \ldots, k$. The policy of P2 remains unconstrained and arbitrary. The immediate reward vector in this game is comprised of one coordinate which is the actual scalar reward and the remaining k coordinates that are the constraints. The maximization of the reward is performed on a sample path, so the average reward should be optimal almost surely. For zero-sum stochastic games where P1 tries to maximize the average reward subject to the constraints it was shown by Shimkin (1994) that under Assumption 1, and provided that the constraints can be satisfied by some policy, there exists a value v. This implies that P1 can guarantee that $\liminf_{n\to\infty} \hat{r}_n \geq \mathrm{v}$ and $\limsup_{n\to\infty} \hat{c}_n \leq \bar{c}$ (with the last inequality a vector inequality) almost surely. It was also shown there that P1 cannot guarantee a reward higher than v while satisfying the constraints.

We now show that the constrained optimization problem may be solved by the framework of the nested target sets. Let the candidate set class $\{T_\alpha\}$ be defined in the reward-constraints space:

$$T_\alpha = \{(r, c^1 \ldots, c^k) : r \geq -\alpha, c^1 \leq \bar{c}^1, \ldots, c^k \leq \bar{c}^k\} \subseteq \mathbb{R}^{k+1}.$$

The α belongs to the index set $\mathcal{L} = \mathbb{R}$. It obviously holds that if $\alpha < \beta$ then $T_\alpha \subset T_\beta$ and that T_α is convex for all α. Continuity of the class $\{T_\alpha\}$ holds as well. Using the MDRL algorithm with persistent exploration we can find the minimal set to be approached. Since we only use a finite grid of directions as in the MDRL algorithm, the guaranteed reward is $\mathrm{v} - \varepsilon$, with constrains satisfied up to ε. But since ε is arbitrary we can effectively get as close as we want to the value. We summarize the results in the following proposition.

Proposition 23 *Assume that some policy satisfies* $\limsup_{n\to\infty} \hat{c}_n \leq \bar{c}$. *Suppose that the modified MDRL algorithm is used for finding the optimal policy in a constrained SG, and that the assumptions of Theorem 22 are satisfied. Then* $\liminf_{n\to\infty} \hat{r}_n \geq \mathrm{v} - \varepsilon$ *and* $\limsup_{n\to\infty} \hat{c}_n \leq \bar{c} + \varepsilon e$ *where the last inequality is in vector notations, and e is a vector of ones.*

We now describe more explicitly the estimation of the target set for the constrained optimization problem. We denote the first coordinate of a unit vector in $\mathbb{B}^{k+1}$ by u^r and the rest of the k coordinates by u^c. Suppose that at time n we are given an estimate $indmax_n$ of the value v. By the geometry of the problem it follows that for any point (r, c), the closest point in $\tilde{T}_n$ is $(r', c'_1, c'_2, \ldots, c'_k)$ where $r' = \max\{r, indmax_n\}$ and $c'_i = \min\{c_i, \bar{c}_i\}$. Because of the special structure of the target set and its convexity, the set of directions which may possibly be steering directions (in the $k+1$ unit ball) is given by

$$SD = \{(u^r, c^1, \ldots, c^k) \in \mathbb{B}^{k+1} : u^r \geq 0, c^1 \leq 0, \ldots, c^k \leq 0\}. \tag{16}$$

To observe that this is the set of possible steering directions, we first note that $u^r \geq 0$ since there is no point in trying to reduce the reward. Indeed, it is easily seen that otherwise $\inf_{m \in T_\alpha} u \cdot m = -\infty$ for all α, so that $\alpha(u, \mathrm{v}(u)) = -\infty$. Second, if a constraint is satisfied then the steering direction will not attempt to violate it. It therefore suffices to consider steering directions, $u_1, \ldots, u_J$, that belong to *SD*. We now show how to estimate the index of the target set given current estimates of the value for each of the projected games.

Proposition 24 *Given directions $u_1, \ldots, u_J$ and estimates $\tilde{\mathrm{v}}(u_j)$, $1 \leq j \leq J$, the estimated index of the target set $\tilde{T}_n$ is given by*

$$indmax_n = -\min_{1 \leq j \leq J, u_j^r > 0} \frac{\tilde{\mathrm{v}}(u_j)_n - u_j^c \cdot (\bar{c}^1, \ldots, \bar{c}^k)}{u_j^r},$$

Proof By the geometry of the problem each direction u_j and corresponding estimate of the value $\tilde{\mathrm{v}}(u_j)$ define a "half space" which must intersect with the estimated target set (by Theorem 7). This half space is given by: $\{(r, c) \in \mathbb{R} \times \mathbb{R}^k : (r, c) \cdot (u_j^r, u_j^c) \geq \tilde{\mathrm{v}}(u_j)\}$. The target set is of the form $\{(r, c) : r \geq r_{max}(u), c \leq \bar{c}\}$, where $r_{max}(u)$ the maximal scalar reward which can be achieved by a specific steering direction u while satisfying the constraints, for an illustration see Figure 9. It follows that r_{max} is in fact the index α in the construction of the candidate target set class. Since $u_r \geq 0$ and $u_c \leq 0$ it follows that

$$(u^r, u^c) \cdot (r_{max}(u), \bar{c}^1, \ldots, \bar{c}^k) = \tilde{\mathrm{v}}(u),$$

so that (assuming $u^r \neq 0$)

$$r_{max}(u) = \frac{\tilde{\mathrm{v}}(u) - u^c \cdot (\bar{c}^1, \ldots, \bar{c}^k)}{u^r}.$$

Overall, the maximal reward that can be achieved is

$$r_{max} = \inf_{\{u \in \mathbb{B}^{k+1} : u^r > 0, u^c \leq 0\}} r_{max}(u). \tag{17}$$

The result follows by considering finitely many directions and recalling that the index is minus r_{max}. ∎

The last two propositions imply that we can solve constrained SGs, and in particular, constrained MDPs. To the best of our knowledge this is the first RL algorithm that solves constrained SGs. Regarding constrained MDPs, we note that one can envision a model-based algorithm (in the spirit of the E^3 algorithm of Kearns and Singh, 1998), however such an algorithm would work only for small problems. Our algorithm may be applied in a model-free setup, as long as underlying model-free scalar RL algorithms can be used. We also comment that the Q-learning style algorithm of Gábor et al. (1998) searches for the optimal deterministic policy, while the optimal one may be stochastic. An example for the solution of constrained MDPs is provided in Section 9.2.

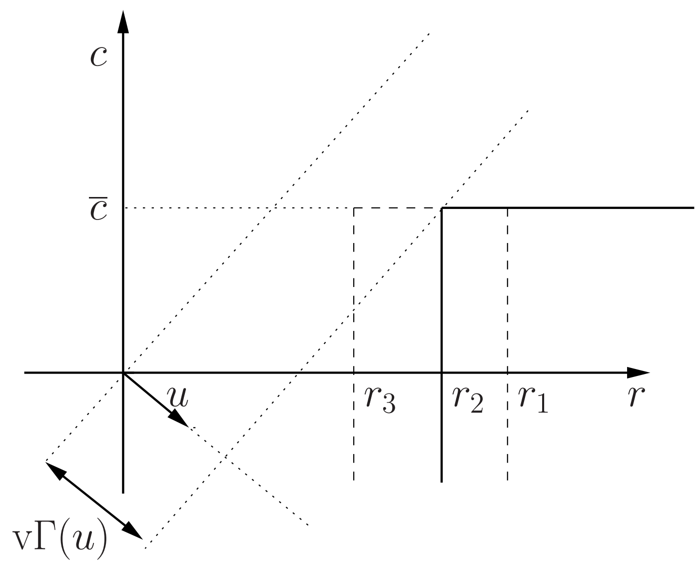

Figure 9: A two dimensional Illustration of finding the value for a constrained game. The x axis is the reward and the y axis is the constraint. The value of the game projected on direction u is $v\Gamma(u)$ which implies that the achievable reward is at most r_2, and possibly less. An average reward of r_1 is not achievable since the set $\{(r,c) : r \geq r_1, c \leq \overline{c}\}$ does not satisfy Theorem 6 for the direction u.

Remark 25 *For general target sets it is possible to devise a similar algorithm to the above where the target set is estimated on-line. For example, one can estimate the game parameters and derive a candidate for the target set based on that estimate. We note that as long as the estimate is consistent (i.e., the final estimate is the actual set to be approached) then schemes which are based on MDRL or CDRL will approach the target set. For example, one may consider approaching the convex Bayes envelope (Mannor and Shimkin, 2003) to arrive at a optimal on-line algorithm for stochastic games. For convex sets there is a particularly interesting possibility of learning the support function (Rockafellar, 1970). Recall that a convex set T can be written as*

$$T = \bigcap_{u \in \mathbb{B}^k} \{x : u \cdot x \geq f_T(u)\},$$

where the function $f_T(u) : \mathbb{B}^k \to \mathbb{R}$ *is the* support function *of T, which is defined as* $f_T(u) \stackrel{\triangle}{=} \inf_{y \in T} u \cdot y$. *The steering direction can be characterized as the maximizer of* $f_T(u) - u \cdot x$ *(Mannor, 2002). Now, if* $f_T(u)$ *can be learned somehow then instead of the true unknown steering direction an approximate steering direction which the maximizer of the* $\tilde{f}_T(u) - u \cdot x$, *where* $\tilde{f}_T$ *is the estimate of* f_T, *can be used. This approach may prove useful when the support function has a simple representation.*

8. Multi-Criterion MDPs

In this section we briefly discuss multi-criterion MDPs. These models may be regarded as a special case of the stochastic game model that was considered so far, with P2 eliminated from the problem. We shall thus only outline the main distinctive features of the MDP case.

Both the MDRL algorithm from Section 4 and the CDRL algorithm from Section 5 remain essentially the same. Their constituent scalar learning algorithms are now learning algorithms for average-reward MDPs. These algorithms are generally simpler than algorithms for the game problem. Examples of optimal or ε-optimal algorithms are Q-Learning with persistent exploration (Bertsekas and Tsitsiklis, 1995), actor-critic schemes (Sutton and Barto, 1998), the E^3 algorithm (Kearns and Singh, 1998), and others. Additionally, it should be noted that the steering policies learned and used within the MDRL algorithm may now be *deterministic* stationary policies, which simplifies the implementation of this algorithm.

Working with a finite number of specified policies as in Section 6 is particularly simple. In this case all that P1 needs to do is to estimate the average reward vector that each policy yields. P1 can attain every point in the convex hull of the average per policy reward vectors by mixing the strategies appropriately. In the absence of an adversary, the problem of approaching a set in a known model becomes much simpler. In this case, if a set is approachable then it may be approached using a stationary (possibly mixed) policy. Indeed, any point in the feasible set of state-action frequencies may be achieved by such a stationary policy (Derman, 1970). Thus, alternative learning schemes that do not rely on the steering approach may be applicable to this problem.

9. Examples

Two brief examples will be used to illustrate the above framework and algorithms. In Subsection 9.1 we show what sample paths of the MDRL algorithm look like for the two dimensional temperature-humidity problem from the introduction. In Subsection 9.2 we solve on-line a queuing problem which is modelled as a constrained MDP.

9.1 The Two-Dimensional Temperature-Humidity Problem

Recall the humidity-temperature example from the introduction. Suppose that the system is modelled in such a way that first P1 chooses a temperature-humidity curve by, say, deciding if to activate heater/cooler or humidifier/dehumidifier a-priori. Then Nature (modelled as P2) chooses the exact location on the temperature-humidity curve by choosing the weather that affect the conditions in the greenhouse. Essentially, P1 chooses the set where the immediate rewards lie in and P2 chooses the exact location of the reward in that set. In Figure 10(a) we show the target set T and three different temperature-humidity curves (these curves are representative curves). Geometrically, P1 chooses the curves where the reward belongs to (each defined by a certain policy of P1 - f_0, f_1, f_2), and P2 chooses the exact location. Additional zero mean with unit variance noise was added to each reward entry. It so happens that there is no stationary policy for P1 which guarantees that the average reward is in T, regardless of P2's actions. We implemented an MDRL algorithm with a "dense" grid of only nine directions. In each direction a version of Littman‘s Q-learning (Littman, 1994), adapted for average cost games, was used (Mannor and Shimkin, 2002). P2's policy was adversarial, and it tried to get the average reward away from the target set. Three sample paths of the average reward for 100,000 epochs that were generated by the MDRL algorithm are shown in

Figure 10(b). Each sample path starts at 'S' and finishes at 'E'. It can be seen that the learning algorithm pushes towards the target set so that the path is mostly on the edge of the target set. Note that in this example a small number of directions was quite enough for approaching the target set. Since the convergence was so fast it made no sense to use the CDRL algorithm, though we would expect it to work well in this case.

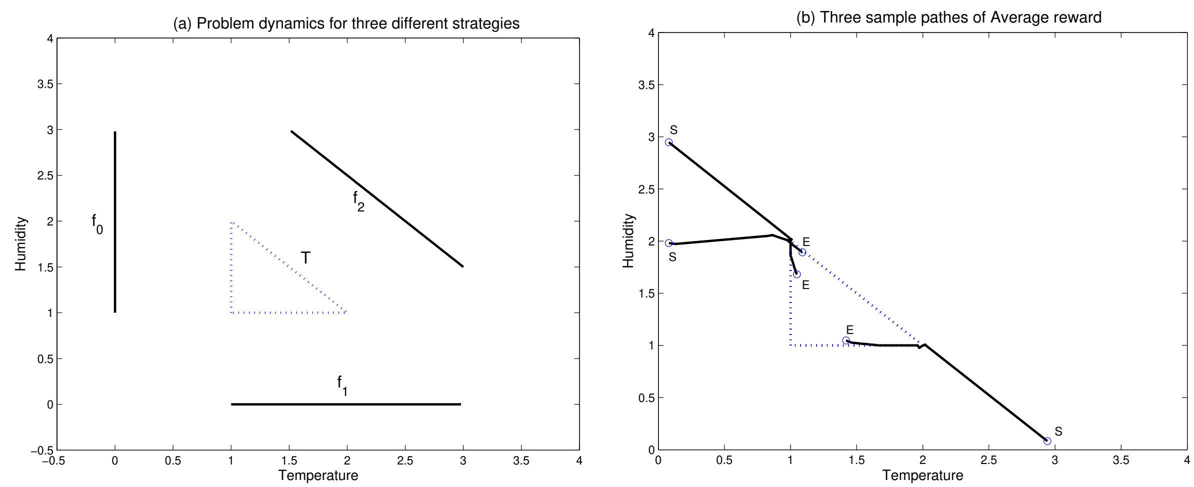

Figure 10: **(a)** Greenhouse problem dynamics. **(b)** Three sample paths from 'S' to 'E'.

9.2 Constrained Optimization of a Queue

In this section we consider the model of a queue with varying service rate. Suppose that there is a queue of buffer length L. Messages arrive to the queue at a rate of μ. P1 controls the service rate which means that if the queue is not empty then a message leaves the queue with probability a_n. The state of the system is the number of messages in the queue $\mathcal{S} = \{0, 1, 2, \ldots, L\}$. The state transition probability satisfies

$$P(s_{n+1}|s_n, a) = \begin{cases} (1-\mu)a & \text{if } L \geq s_n \geq 1 \text{ and } s_{n+1} = s_n - 1, \\ \mu a + (1-\mu)(1-a) & \text{if } L > s_n \geq 1 \text{ and } s_{n+1} = s_n, \\ \mu + (1-\mu)(1-a) & \text{if } s_{n+1} = s_n = L, \\ \mu(1-a) & L-1 \geq \text{if } s_n \geq 0 \text{ and } s_{n+1} = s_n + 1, \\ 1 - \mu(1-a) & \text{if } s_{n+1} = s_n = 0, \\ 0 & \text{otherwise.} \end{cases}$$

The reward of P1 is a function of the number of vacant places in the queue. For simplicity we assumed that $r_n = c(L - s_n)$ where c is a positive constant. P1's objective is to maximize the reward it obtains. This is equivalent to minimizing the average queue length. P1 may affect the state transitions through change of the service rate $a_n \in \{a_L, a_H\}$. The service constraint of P1 is to have the average service rate $\hat{a}_n$ lower than some pre-specified constant c_0. The optimization problem we have is

$$\max \hat{r}_n$$

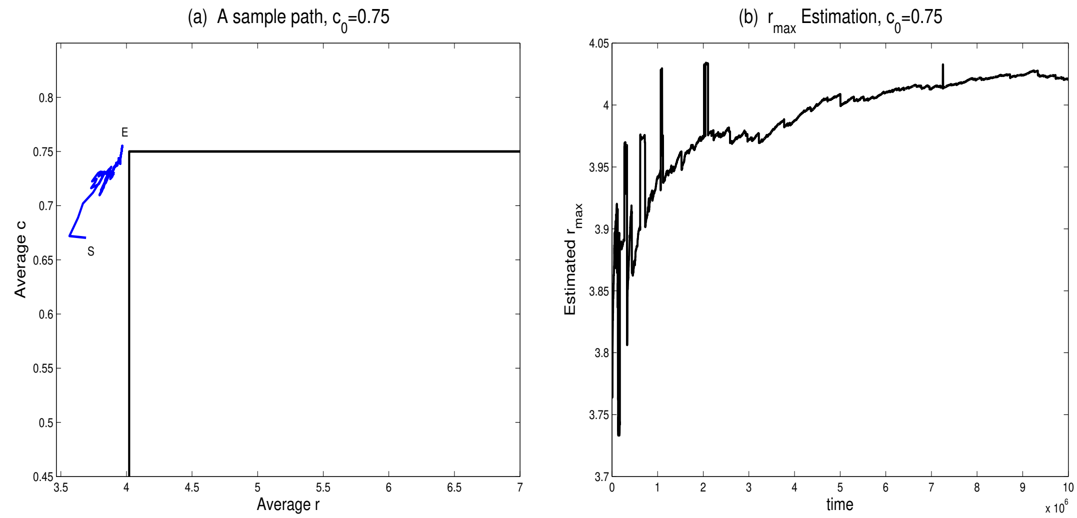

Figure 11: A sample path (a) and r_{max} estimation (b) for a run of the modified MDRL for the constrained queue problem, $c_0 = 0.75$. Note that r_{max} converged to 4.02.

$$\text{s.t. } \hat{a}_n \leq c_0 \,.$$

Note that the maximum should be attained per sample path and may be achieved by a stationary policy (e.g., Altman, 1999). Moreover, it can be shown that there exists an optimal policy with at most one randomization. We used the modified MDRL algorithm for solving the above problem for different values of c_0. The parameters of the problems were $\mu = 0.5$, $L = 9$, $c = 0.5$, $\delta = 0.01$ (exploration rate in stage 2 of the modified MDRL algorithm from Figure 8), $a_L = 0.5$, $a_H = 0.8$, and the number of steering directions was 10. For three values of c_0 we show in Figures 11-13 the sample path (average reward and average constraint) on the left side and the estimation of r_{max} over time as per Equation (17) on the right side. The final estimate of the target set is also drawn for each sample path. The average vector-valued reward typically converges to the corner of the target set and pretty much stays there with random fluctuations caused by the exploration. The estimation of r_{max} is quite stable and tends to converge. As expected, the reward increases as the constraint becomes less stringent.

10. Conclusion

We have presented learning algorithms that address the problem of multi-objective decision making in a dynamic environment, which may contain arbitrarily varying elements, focusing on the long-term average reward vector. The proposed algorithms learn to approach a prescribed target set in multi-dimensional performance space, provided this set satisfies a certain geometric condition with respect to the dynamic game model. For convex sets, this sufficient condition is also necessary. The algorithms we suggested essentially rely on the theory of approachability for stochastic games, and are based on the idea of steering the average reward vector towards the target set. We then extended the algorithms to the case of a target set which is not prescribed but rather may depend on

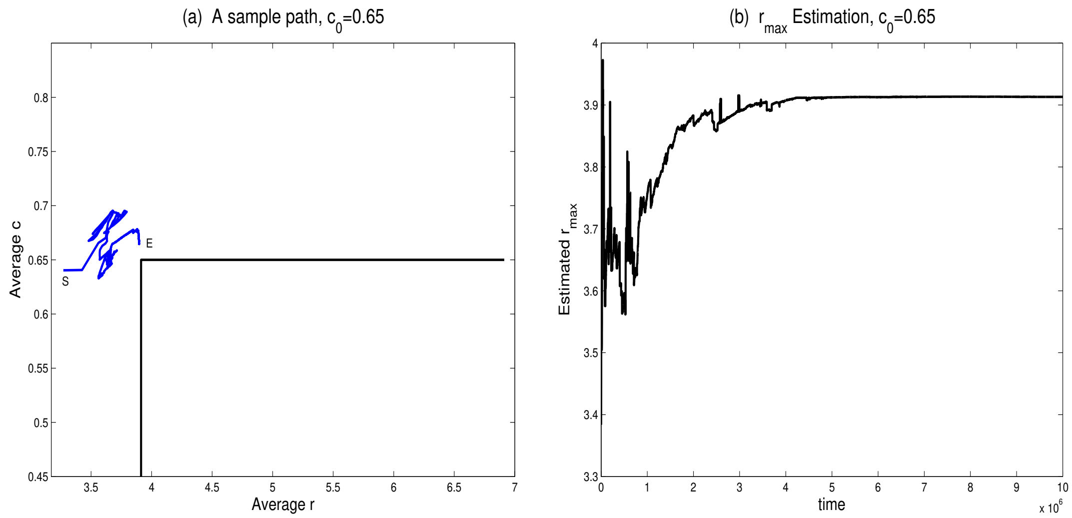

Figure 12: A sample path (a) and r_{max} estimation (b) for a run of the modified MDRL for the constrained queue problem, $c_0 = 0.65$. Note that r_{max} converged to 3.92.

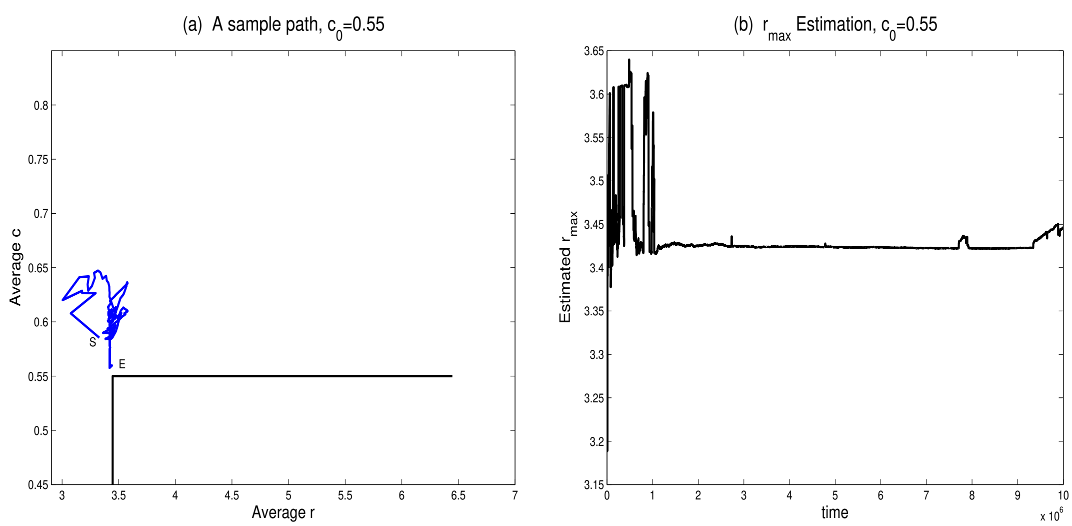

Figure 13: A sample path (a) and r_{max} estimation (b) for a run of the modified MDRL for the constrained queue problem, $c_0 = 0.55$. Note that r_{max} converged to 3.45.

the parameters of the game. This was done by requiring an order relation between candidate target sets or by assuming that the set can be learned consistently. Specifically, we showed that optimal learning policies in constrained MDPs and constrained games may be learned using the proposed schemes. It should be noted that the proposed schemes are in fact algorithmic frameworks, or *meta-*

algorithms, in which one can plug-in different scalar learning algorithms, provided they satisfy the required convergence properties. All results are developed under the single-state recurrence assumption (Assumption 1), with possible relaxations outlined in Remarks 1 and 2.

Several issues call for further study. First, the algorithmic framework we presented can undoubtedly be improved in terms of convergence speed and memory complexity. For example, a reduction of the number of steering directions used in the MDRL algorithm is of practical interest for both convergence speed (less things to learn) and memory requirement (less policies to store in memory). In some cases, especially when the requirements embodied by the target set T are not stringent, this number may be quite small compared to the conservative estimate used above. A possible refinement of the MDRL algorithm is to eliminate directions that are not required, perhaps doing so on-line. Second, specific algorithms may be devised for the special problems of constrained MDPs and constrained SGs which are simpler and with improved convergence rates. Third, the performance criteria considered here were solely based on the long-term average reward. It should be of practical interest to extend this framework to an appropriately windowed or discounted criterion, that gives more emphasis to *recent* rewards. Finally, it should be challenging to apply the proposed framework and algorithms to realistic learning problems. An interesting application class is related to communication networks, with their multiple QoS requirements and averaged performance criteria. We note that the average reward is the criterion of interest is such applications, and that the high rate of data makes our algorithms particularly useful (see Brown, 2001). Such applications would undoubtedly require further attention to issues of computational complexity and convergence rates.

Acknowledgments

This research was supported by the fund for the promotion of research at the Technion. We are indebted to three anonymous referees for thoughtful remarks and for significantly improving the presentation of this work. S.M. would like to thank Yishay Mansour, Eyal Even-Dar, and Ishai Menache for helpful discussions.

Appendix A.

In this appendix we discuss a generalization of the CDRL algorithm and prove that it approaches T rather than the ε expansion of T. Recall that in the CDRL algorithm (see Figure 5) an accuracy parameter ε was provided. The algorithm progressed in blocks where in each block a direction is fixed for a duration of $N(\varepsilon)$. In this $N(\varepsilon)$ duration the vector-valued reward is steered towards the target set T using an ε-optimal in expectation learning algorithm. The following algorithm is a modification of the CDRL algorithm where we slowly take ε to 0. As before, we denote by $N(\varepsilon)$ the time required for the learning algorithm to achieve an ε optimal reward in expectation (see Definition 12).

Theorem 26 *Suppose that for every $\varepsilon > 0$ there exists an ε-optimal in expectation algorithm for each projected SG. Suppose further that $N(\varepsilon)$ has polynomial dependence in $1/\varepsilon$, and assume that the target set satisfies Equation (3). Then T is approached using the modified CDRL algorithm.*

Proof We note that we cannot use Theorem 6 or any of its extensions from Mannor and Shimkin (2000) since a stopping time (in which steering strategies are switched) which is uniformly bounded is not provided. We therefore use a direct argument. It suffices to prove that T^η is approached for

Input: Target set T; ε-optimal learning algorithm with polynomial dependency N.
0. Set $\hat{m}_0$ arbitrarily; $i = 1$; $n = 0$. 1. Repeat forever: 2. Set $\varepsilon_i = \frac{1}{i}$. 3. If $\hat{m}_n \in T$ play arbitrarily for $N(\varepsilon_i)$ steps; $n := n + N(\varepsilon_i)$. 4. Else let $u_i = u_{\hat{m}_n}$. 5. Play an ε_i-optimal in expectation learning algorithm with reward at time n given by $r_n = u_i \cdot m_n$ and parameter ε_i. Play the algorithm for $N(\varepsilon_i)$ time epochs; $n := n + N(\varepsilon_i)$. 6. Set $i = i + 1$.

Figure 14: The modified CDRL algorithm.

every $\eta > 0$ (Mannor and Shimkin, 2000). Fix $\eta > 0$.
Let $\tau_0 = 0$ and define $\tau_i = \tau_{i-1} + N(\varepsilon_i)$ for $i > 0$. Let the polynomial dependence of the execution time of each stage of the algorithm be denoted by $N(\varepsilon_i) = Ci^{\ell}$. It follows that $\tau_i = C\sum_{j=1}^{i} j^{\ell} = O(i^{\ell+1})$.
By the definition of the algorithm either $\hat{m}_{\tau_i} \in T^{\eta}$ or each algorithm produces an average reward which is higher than the value of the projected game (minus ε). This implies that the expected progress in direction u_i is positive:

$$\frac{\mathbb{E}(\sum_{t=\tau_i}^{\tau_{i+1}-1} m_t \,|F_{\tau_i})}{\tau_{i+1} - \tau_i} \cdot u_i \geq v\Gamma(u_i) - \varepsilon_i\,, \tag{18}$$

where F_t is the sigma algebra generated by all observations up to time t. Let i be large enough so that $\varepsilon_i < \eta$. Let d_i denote the distance from $\hat{m}_{\tau_i}$ to T^{η}, i.e., $d_i = d(\hat{m}_{\tau_i}, T^{\eta})$. We begin by showing that $d_i \to 0$ almost surely, and then show that $d(\hat{m}_n, T^{\eta}) \to 0$ for all n.

Consider the square of the distance, d_i^2. Let y_i denote the closest point in T^{η} to $\hat{m}_{\tau_i}$, for an illustration see Figure 15. Equation (18) implies that for i large enough (so that $\varepsilon_i < \eta$),

$$\mathbb{E}\Big(\frac{(\sum_{t=\tau_i}^{\tau_{i+1}-1} m_t \,|F_{\tau_i})}{\tau_{i+1} - \tau_i} - y_i\Big) \cdot u_i \geq 0. \tag{19}$$

It follows that

$$\begin{aligned}
\mathbb{E}(d_{i+1}^2|F_{\tau_i}) &= \mathbb{E}(\|\hat{m}_{\tau_{i+1}} - y_{i+1}\|_2^2\,|F_{\tau_i}) \\
&\leq \mathbb{E}(\|\hat{m}_{\tau_{i+1}} - y_i\|_2^2\,|F_{\tau_i}) \\
&= \mathbb{E}(\|\hat{m}_{\tau_{i+1}} - \hat{m}_{\tau_i} + \hat{m}_{\tau_i} - y_i\|_2^2\,|F_{\tau_i}) \\
&= \|\hat{m}_{\tau_i} - y_i\|_2^2 + \mathbb{E}(\|\hat{m}_{\tau_{i+1}} - \hat{m}_{\tau_i}\|_2^2\,|F_{\tau_i}) + \\
&\qquad \mathbb{E}(2(\hat{m}_{\tau_i} - y_i)\cdot(\hat{m}_{\tau_{i+1}} - \hat{m}_{\tau_i})\,|F_{\tau_i}),
\end{aligned} \tag{20}$$

where the last equality is just the result of simple algebraic manipulation. The first element in Equation (20) is simply d_i^2. To bound the second element in Equation (20) note that

$$\Big\|\hat{m}_{\tau_{i+1}} - \hat{m}_{\tau_i}\Big\|_2^2 = \Big\|\frac{1}{\tau_{i+1}}\sum_{t=\tau_i}^{\tau_{i+1}-1} m_t + (\frac{1}{\tau_{i+1}} - \frac{1}{\tau_i})\sum_{t=0}^{\tau_i-1} m_t\Big\|_2^2$$

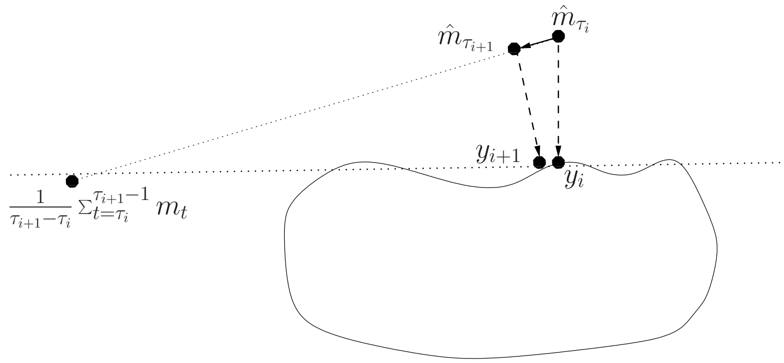

Figure 15: Illustration of one step of the algorithm. The closest point in T to $\hat{m}_{\tau_i}$ is y_i and similarly the closest point to $\hat{m}_{\tau_{i+1}}$ is y_{i+1}. Between "switching times" the reward obtained by P1 lies on the other side of the hyperplane perpendicular to the segment $(y_i, \hat{m}_{\tau_i})$.

$$
\begin{aligned}
&\leq \frac{C_1}{(i+1)^{2\ell+2}} \Big\| \sum_{t=\tau_i}^{\tau_{i+1}-1} m_t \Big\|_2^2 + C_2 \Big(\frac{i^{\ell+1}-(i+1)^{\ell+1}}{(i+1)^{\ell+1} i^{\ell+1}} i^{\ell+1} \Big)^2 \Big\| \hat{m}_{\tau_i} \Big\|_2^2 \\
&\leq C_3 \frac{1}{i^2} \Big(\frac{1}{\tau_{i+1}-\tau_i} \Big)^2 \Big\| \sum_{t=\tau_i}^{\tau_{i+1}-1} m_t \Big\|_2^2 + C_4 \frac{1}{i^2} \Big\| \hat{m}_{\tau_i} \Big\|_2^2,
\end{aligned}
$$

where we used the fact that $\|a+b\|_2^2 \leq 2\|a\|_2^2 + 2\|b\|_2^2$ for the first inequality and C_1, C_2, C_3, C_4 are constants independents of i. Taking the expectation conditioned on F_{τ_i} we can bound $\mathbb{E}(\|\hat{m}_{\tau_{i+1}} - \hat{m}_{\tau_i}\|_2^2 | F_{\tau_i})$ by $\frac{C}{i^2}$. Note that the second term is deterministic (conditioned on F_{τ_i} and vanishes from some point on.

The third element of Equation(20) is more tricky. Since

$$
\begin{aligned}
\hat{m}_{\tau_{i+1}} - \hat{m}_{\tau_i} &= \frac{1}{\tau_{i+1}} \sum_{t=0}^{\tau_{i+1}-1} m_t - \frac{1}{\tau_i} \sum_{t=0}^{\tau_i-1} m_t \\
&= \big(\frac{1}{\tau_{i+1}} - \frac{1}{\tau_i}\big) \sum_{t=0}^{\tau_i-1} m_t + \frac{1}{\tau_{i+1}} \sum_{t=\tau_i}^{\tau_{i+1}-1} m_t \\
&= \frac{\tau_i - \tau_{i+1}}{\tau_{i+1}} \hat{m}_{\tau_i} + \frac{1}{\tau_{i+1}} \sum_{t=\tau_i}^{\tau_{i+1}-1} m_t,
\end{aligned}
$$

we have that

$$
\begin{aligned}
(\hat{m}_{\tau_i} - y_i) \cdot (\hat{m}_{\tau_{i+1}} - \hat{m}_{\tau_i}) &= (\hat{m}_{\tau_i} - y_i) \cdot (\hat{m}_{\tau_{i+1}} - \frac{\tau_i - \tau_{i+1}}{\tau_{i+1}} y_i + \frac{\tau_i - \tau_{i+1}}{\tau_{i+1}} y_i - \hat{m}_{\tau_i}) \\
&= (\hat{m}_{\tau_i} - y_i) \cdot \Big(\frac{\tau_i - \tau_{i+1}}{\tau_{i+1}} \hat{m}_{\tau_i} - \frac{\tau_i - \tau_{i+1}}{\tau_{i+1}} y_i + \\
&\qquad \frac{\tau_i - \tau_{i+1}}{\tau_{i+1}} y_i + \frac{1}{\tau_{i+1}} \sum_{t=\tau_i}^{\tau_{i+1}-1} m_t \Big) \\
&= \frac{\tau_i - \tau_{i+1}}{\tau_{i+1}} (\hat{m}_{\tau_i} - y_i) \cdot (\hat{m}_{\tau_i} - y_i)
\end{aligned}
$$

$$+\frac{\tau_i-\tau_{i+1}}{\tau_{i+1}}(\hat{m}_{\tau_i}-y_i)\cdot(\frac{1}{\tau_{i+1}-\tau_i}\sum_{t=\tau_i}^{\tau_{i+1}-1}m_t-y_i).$$

Since $\tau_i=Ci^{\ell+1}$ the fraction $\frac{\tau_{i+1}-\tau_i}{\tau_{i+1}}=O(1/i)$. The first term is therefore approximately $-d_i^2\frac{C}{i}$. We get the following inequality:

$$\mathbb{E}(d_{i+1}^2|F_{\tau_i})\leq(1-\frac{C'}{i})d_i^2+\frac{C''}{i^2}+C'''\mathbb{E}\left(2\frac{(\hat{m}_{\tau_i}-y_i)\cdot(\sum_{t=\tau_i}^{\tau_{i+1}-1}m_t-y_i)}{i+1}\Big|F_{\tau_i}\right), \tag{21}$$

where C', C'', and C''' are positive constants. Now according to Equation (19) the last term in (21) is negative in expectation. We therefore have that $\mathbb{E}(d_{i+1}^2|F_{\tau_i})\leq(1-\frac{C_1}{i})d_i^2+\frac{C_2}{i^2}$. Using Lemma 27 we have that $d_i^2\to 0$ almost surely. To conclude the proof one should show that $d(\hat{m}_n,T^\eta)\to 0$ for all n. Let i denote the maximal τ_i smaller than n. By the triangle inequality,

$$d(\hat{m}_n,T^\eta)\leq d(\hat{m}_{\tau_i},T^\eta)+\|\hat{m}_{\tau_i}-\hat{m}_n\|_2.$$

The first term converges to 0 by the above. To bound the second term note that

$$\begin{aligned}
\|\hat{m}_{\tau_i}-\hat{m}_n\|_2 &= \Big\|\frac{1}{n}\sum_{t=0}^{n-1}m_t-\frac{1}{\tau_i}\sum_{t=0}^{\tau_i-1}m_t\Big\|_2\\
&= \Big\|\frac{1}{n}\sum_{t=\tau_i}^{n-1}m_t+\frac{\tau_i-n}{n\tau_i}\sum_{t=0}^{\tau_i-1}m_t\Big\|_2\\
&\leq \frac{1}{n}\Big\|\sum_{t=\tau_i}^{n-1}m_t\Big\|_2+\frac{n-\tau_i}{n\tau_i}\Big\|\sum_{t=0}^{\tau_i-1}m_t\Big\|_2\\
&= \frac{n-\tau_i}{n}\frac{1}{n-\tau_i}\Big\|\sum_{t=\tau_i}^{n-1}m_t\Big\|_2+\frac{n-\tau_i}{n}\frac{1}{\tau_i}\Big\|\sum_{t=0}^{\tau_i-1}m_t\Big\|_2,
\end{aligned} \tag{22}$$

where the inequality is due to the triangle inequality. By our construction of τ_i it follows that $\lim_{n\to\infty}\frac{n-\tau_i}{n}=0$. Since the random vector reward m_t has finite expectation and bounded second moment (uniformly in t, since there are finitely many states and actions) it follows that $\frac{1}{\tau_i}\|\sum_{t=0}^{\tau_i-1}m_t\|_2$ is asymptotically contained in some ball of finite radius with probability one.[3] Consequently, the second term in Equation (22) converges to 0 almost surely. The first term in Equation (22) converges to 0 as well by a standard probabilistic argument.[4] ∎

Lemma 27 *Assume e_t is a non-negative random variable, measurable according to the sigma algebra F_t ($F_t\subset F_{t+1}$) and that*

$$\mathbb{E}(e_{t+1}|F_t)\leq(1-d_t)e_t+cd_t^2. \tag{23}$$

Further assume that $\sum_{t=1}^{\infty}d_t=\infty$, $d_t\geq 0$, and that $d_t\to 0$. Then $e_t\to 0$ P-a.s.

3. To see that, take the centralized version of that random variable which converges to 0, almost surely by an appropriate version of the strong law of large numbers. Note that the center may not converge, but is guaranteed to be in some bounded region.
4. The result follows by Chebyshev's inequality on the centralized version of $\sum_{t=\tau_i}^{n-1}m_t$ and the Borel-Cantelli Lemma, noticing that $\sum_{t=\tau_i}^{n-1}m_t$ is multiplied by a deterministic series that converges to 0.

Proof First note that by taking the expectation of Equation (23) we get

$$\mathbb{E}e_{t+1} \leq (1-d_t)\mathbb{E}e_t + cd_t^2.$$

According to Bertsekas and Tsitsiklis (1995, page 117) it follows that $\mathbb{E}e_t \to 0$. Since e_t is non-negative it suffices to show that e_t converges. Fix $\varepsilon > 0$, let

$$V_t^\varepsilon \stackrel{\triangle}{=} \max\{\varepsilon, e_t\}.$$

Since $d_t \to 0$ there exists $T(\varepsilon)$ such that $cd_t < \varepsilon$ for $t > T$. Restrict attention to $t > T(\varepsilon)$. If $e_t < \varepsilon$ then

$$\mathbb{E}(V_{t+1}^\varepsilon|F_t) \leq (1-d_t)\varepsilon + cd_t^2 \leq \varepsilon \leq V_t^\varepsilon.$$

If $e_t > \varepsilon$ we have

$$\mathbb{E}(V_{t+1}^\varepsilon|F_t) \leq (1-d_t)e_t + d_t e_t \leq V_t^\varepsilon.$$

V_t^ε is a super-martingale, by a standard convergence argument we get $V_t^\varepsilon \to V_\infty^\varepsilon$.

By definition $V_t^\varepsilon \geq \varepsilon$ and therefore $\mathbb{E}V_t^\varepsilon \geq \varepsilon$. Since $\mathbb{E}[\max(X,Y)] \leq \mathbb{E}X + \mathbb{E}Y$ it follows that $\mathbb{E}V_t^\varepsilon \leq \mathbb{E}e_t + \varepsilon$. So that $\mathbb{E}V_\infty^\varepsilon = \varepsilon$. Now we have a positive random variable, with expectation ε which is not smaller than ε with probability 1. It follows that $V_\infty^\varepsilon = \varepsilon$.

To summarize, we have shown that for every $\varepsilon > 0$ with probability 1,

$$\limsup_{t\to\infty} e_t \leq \limsup_{t\to\infty} V_t^\varepsilon = \lim_{t\to\infty} V_t^\varepsilon = \varepsilon.$$

Since ε is arbitrary and e_t non-negative it follows that $e_t \to 0$ almost surely. ∎

References

J. Abounadi, D. Bertsekas, and V. Borkar. Stochastic approximation for non-expansive maps: Application to Q-learning algorithms. *SIAM Journal on Control and Optimization*, 41:1–22, 2002.

E. Altman. *Constrained Markov Decision Processes*. Chapman and Hall, 1999.

E. Altman and A. Shwartz. Constrained Markov games: Nash equilibria. In V. Gaitsgory, J. Filar, and K. Mizukami, editors, *Annals of the International Society of Dynamic Games*, volume 5, pages 303–323. Birkhauser, 2000.

M. Anthony and P. L. Bartlett. *Neural Network Learning; Theoretical Foundations*. Cambridge University Press, 1999.

ATM Forum Technical Committee. Traffic management specification version 4.1. www.atmforum.org, March 1999.

P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The non-stochastic multi-armed bandit problem. *SIAM Journal on Computing*, 32:48–77, 2002.

J. Baxter, A. Tridgell, and L. Weaver. TDLeaf(λ): Combining temporal difference learning with game-tree search. In *Proceedings of the 9th Australian Conference on Neural Networks (ACNN-98)*, 1998. URL `http://cs.anu.edu.au/~Lex.Weaver/pub_sem/publications/ACNN98.pdf`.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1995.

D. Blackwell. An analog of the minimax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6(1):1–8, 1956.

R. I. Brafman and M. Tennenholtz. R-MAX, a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3:213–231, 2002.

T. X. Brown. Switch packet arbitration via queue-learning. In *Advances in Neural Information Processing Systems 14*, pages 1337–1344, 2001.

R. L. Carraway, T. L. Morin, and H. Moskowitz. Generalized dynamic programming for multicriteria optimization. *European Journal of Operational Research*, 44:95–104, 1990.

C. Derman. *Finite state Markovian decision processes*. Academic Press, 1970.

M. Ehrgott and X. Gandibleux. *Multiple Criteria Optimization. State of the Art Annotated Bibliographic Surveys*. Kluwer, 2002.

J. Filar and K. Vrieze. *Competitive Markov Decision Processes*. Springer Verlag, 1996.

Z. Gábor, Z. Kalmár, and C. Szepesvári. Multi-criteria reinforcement learning. In *Proc. of the 15th Int. Conf. on Machine Learning*, pages 197–205. Morgan Kaufmann, 1998.

M. I. Henig. Vector-valued dynamic programming. *SIAM Journal on Control and Optimization*, 21 (3):490–499, 1983.

A. J. Hoffman and R. M. Karp. On nonterminating stochastic games. *Management Science*, 12(5): 359–370, 1966.

L. P. Kaelbling, M. Littman, and A. W. Moore. Reinforcement learning - a survey. *Journal of Artificial Intelligence Research*, 4:237–285, May 1996.

D. Kahaneman and A. Tversky. Prospect theory: an analysis of decision under risk. *Econometrica*, 47:263–291, 1979.

M. Kearns and S. Singh. Near-optimal reinforcement learning in polynomial time. In *Proc. of the 15th Int. Conf. on Machine Learning*, pages 260–268. Morgan Kaufmann, 1998.

M. Kearns and S. Singh. Finite-sample convergence rates for Q-learning and indirect algorithms. In *Neural Information Processing Systems 11*, pages 996–1002. Morgan Kaufmann, 1999.

V. R. Konda and J. N. Tsitsiklis. Actor-critic algorithms. Submitted to the SIAM Journal on Control and Optimization, an earlier version appeared in NIPS 1999, February 2001.

M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In Morgan Kaufman, editor, *Eleventh International Conference on Machine Learning*, pages 157–163, 1994.

M. L. Littman and C. Szepesvári. A unified analysis of value-function-based reinforcement-learning algorithms. *Neural Computation*, 11(8):2017–2059, 1999.

S. Mahadevan. Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine Learning*, 22(1):159–196, 1996.

S. Mannor. *Reinforcement Learning and Adaptation in Competitive Dynamic Environments*. PhD thesis, Department of Electrical Engineering, Technion, April 2002.

S. Mannor and N. Shimkin. Generalized approachability results for stochastic games with a single communicating state. Technical report EE- 1263, Faculty of Electrical Engineering, Technion, Israel, October 2000. Appeared in ORP3, 2001.

S. Mannor and N. Shimkin. Reinforcement learning for average reward zero-sum games. Technical report EE- 1316, Faculty of Electrical Engineering, Technion, Israel, May 2002.

S. Mannor and N. Shimkin. The empirical Bayes envelope and regret minimization in competitive Markov decision processes. *Mathematics of Operations Research*, 28(2):327–345, May 2003.

J. F. Mertens. Stochastic games. In Robert J. Aumann and Sergiu Hart, editors, *Handbook of Game Theory with Economic Applications*, volume 3, chapter 47. Elsevier Science Publishers, 2002.

J. F. Mertens and A. Neyman. Stochastic games. *International Journal of Game Theory*, 10(2): 53–66, 1981.

A. S. Pozniyak, K. Najim, and E. Gomez-Ramirez. *Self Learning Control of Finite Markov Chains*. Marcel Decker, 1999.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

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

N. Shimkin. Stochastic games with average cost constraints. In T. Basar and A. Haurie, editors, *Advances in Dynamic Games and Applications*, pages 219–230. Birkhauser, 1994.

N. Shimkin and A. Shwartz. Guaranteed performance regions in Markovian systems with competing decision makers. *IEEE Transactions on Automatic Control*, 38(1):84–95, January 1993.

H. A. Simon. *The Sciences of the Artificial*. MIT Press, 1996.

S. Singh, T. Jaakkola, M. Littman, and C. Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning*, 38(3):287–308, 2000.

R. Steuer. *Multiple Criteria Optimization: Theory, Computation and Application*. Wiley, 1986.

R. S. Sutton and A. G. Barto. *Reinforcement Learning*. MIT Press, 1998.

G. J. Tesauro. TD-gammon, a self-teaching backgammon program, achieves a master-level play. *Neural Computation*, 6:215–219, 1996.

D. White. Multi-objective infinite-horizon discounted markov decision processes. *Journal of Mathematical Analysis and Applications*, 89:639–647, 1982.