Multi-criteria Reinforcement Learning*

Zoltán Gábor, Zsolt Kalmár and Csaba Szepesvári
Associative Computing Ltd.
Budapest 1121, Konkoly Thege M. út 29–33
e-mails: {gzoli,kalmar,szepes}@mindmaker.kfkipark.hu

Abstract

We consider multi-criteria sequential decision making problems where the vector-valued evaluations are compared by a given, fixed total ordering. Conditions for the optimality of stationary policies and the Bellman optimality equation are given for a special, but important class of problems when the evaluation of policies can be computed for the criteria independently of each other. The analysis requires special care as the topology introduced by pointwise convergence and the order-topology introduced by the preference order are in general incompatible. Reinforcement learning algorithms are proposed and analyzed. Preliminary computer experiments confirm the validity of the derived algorithms. These type of multi-criteria problems are most useful when there are several optimal solutions to a problem and one wants to choose the one among these which is optimal according to another fixed criterion. Possible application in robotics and repeated games are outlined.

1 Introduction

Scalar-valued reinforcement learning (RL) algorithms are capable of solving difficult multi-step decision problems when the decision criteria can be expressed in a recursive way as a function of the immediate scalar sreinforcement. However, there are some important cases when there is no simple way to express the optimization criteria as a function of a single scalar reinforcement value. Consider, for example, the dilemma of Leibniz's ass. This poor animal is placed at equal distances away from two platefuls of food. He is hungry so he feels like going to one of the plates. However, if he goes to one plate then there is a chance that the dish from the other one gets stolen. Since the ass is greedy (he does not want any dish to be stolen away) he will never move and will, eventually, die.

In this example the ass has two different objectives competing with one another. The first one is to eat so that the ass can stay alive, the second one is to prevent the dishes from being stolen. A reasonable compromise, which could be termed the "watchmen's compromise", is to minimize the number of dishes stolen per unit time such that the ass manages to stay alive: $\lim_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T} S_t \to \min$ s.t. $\lim_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T} R_t \geq R_{\mathrm{crit}}$. Here $S_t \in \{0,1\}$ is the indicator of whether a plate was stolen at time t, $R_t = \{0,1\}$ is the indicator of whether the ass was consuming at time t, and R_{crit} is the critical amount of food per unit time needed for staying alive. We can use a Tauberian approximation to the above criterion [Ross, 1970]:

$$\sum_{t=0}^{\infty} \gamma^t S_t \to \min \quad \text{s.t.} \quad \sum_{t=0}^{\infty} \gamma^t R_t \geq R'_{\mathrm{crit}}, \qquad (1)$$

where $0 < \gamma < 1$ is a value close to 1, $R'_{\mathrm{crit}} = R_{\mathrm{crit}}/(1-\gamma)$, i.e., the discounted total cost (reward) criterion replaces the average cost (reward) criterion.[1] If γ is sufficiently close to 1 then optimal solutions to the second criterion will be close to optimal measured by the first criterion. Since the decision should be made on the basis of both the amount of food eaten and

*Revised on 05/04/2004. Thanks to Peter Geibel pointing out a problem with transforming the constrained multi-criteria DP problem to an unconstrained multi-criteria problem in the original version of this paper.

[1] In order to simplify the presentation we implicitly assume here that the decision process is deterministic. However, this assumption is in no way essential to the subsequent developments and will be abandoned later.

the number of plates stolen, and both of these should be computed separately, this calls for a vector-valued representation of reinforcement values, i.e., in the case of Leibniz's ass, the reinforcement at time t will be (R_t, S_t).

Another reasonable compromise is to maximize the weighted sum of protected plates and the amount eaten: $\sum_{t=0}^{\infty} \gamma^t (w_1(1-S_t) + w_2 R_t) \to \max$, where $w_1, w_2 > 0$. This reduces the problem to the case of scalar-valued reinforcement values. Here, we do not want to argue against this or other reductions, but we want to show that under certain conditions reinforcement learning algorithms can be extended to the vector-valued case in a sensible way.

If the immediate reinforcement is vector-valued then so will be the long-term reinforcement, and, specifically, the evaluation of policies. Then the comparison of policies becomes problematic. The requirements are the following: we want to compare any pairs of policies and, in particular, we want a transitive and reflexive comparison operator. Several approaches will be shown below. No matter how the policies are compared the notion of an optimal policy can be defined at this point: an optimal policy is one which compares favorably with any other policy.

The comparison methods are best illustrated by the above problem. Let $v_\pi(x) \in \mathbf{R}^2$ denote the evaluation of policy π in state x with $v_\pi(x)^T = (v_{\pi,1}(x), v_{\pi,2}(x))$, where $v_{\pi,1}(x)$ is the maximum of the amount of food eaten *and* R_{crit}, while $v_{\pi,2}(x)$ is the number of plates stolen when policy π is being used, both being computed when policy π is being used beginning from state x. Then the criterion considered above suggests to compare any pair of policies (π_1, π_2) by first comparing the first components of their respective evaluation functions: π_1 is better than π_2 if $v_{\pi_1,1}(x) > v_{\pi_2,2}(x)$. Since the evaluations are cut at R_{crit} we may expect that $v_{\pi_1,1}(x)$ and $v_{\pi_2,2}(x)$ will be equal in a large number of cases. Then, we compare the second components: π_1 is better than π_2 if $v_{\pi_1,2}(x) < v_{\pi_2,2}(x)$ (note the reversed relational symbol). That is, among policies which let Leibniz's ass staying alive, the ones with a smaller number of stolen plates are preferred. In this case there is an *ordering among the vector-components* (the two criteria) and so this problem is one example of *ordinal multi-criteria decision problems.* Ordinal multi-criteria decision problems which was considered a long time ago by Mitten [1964] and Sobel [1975] in terms of preference relations over "partial policies". In order the subordinate criteria to be useful at all, the optimization problem corresponding to the main criterion should have multiple solutions. This can be achieved using reduced reinforcement-spaces. As an interesting example note that Asimov's robots use such an ordered multi-criteria decision problem representation: the "laws of robotics" claims that robots have to *i)* defend human beings, *ii)* defend themselves unless this conflicts with rule *i)*; and *iii)* serve human beings unless this conflicts with rules *i)* or *ii)*. Also this type of criterion is related to solving MDPs in parallel, a problem similar to that of considered by Singh and Cohn [1997] and empirically in mobile robot learning domains by Asada et al. [1994].

Criterion (1) can also be viewed as one that defines a discounted optimization problem subject to a discounted constraint. Structural properties of such problems were studied extensively in the control and operations research literature, e.g. by Frid [1972], Heyman and Sobel [1984], Altman and Schwartz [1991].

Another approach is to compare any pair of policies, (π_1, π_2), by comparing the weighted sum of the components of theirs evaluation functions, e.g. $w_1 v_{\pi_1,1}(x) + w_2 v_{\pi_1,2}(x)$ and $w_1 v_{\pi_2,1}(x) + w_2 v_{\pi_2,2}(x)$ $(w_1, w_2 \in \mathbf{R})$. Note that this criterion, often called the weighted criterion (see Feinberg and Schwartz [1995] and the references therein), is different from the one obtained by the linear combination of the immediate reinforcement values iff the discount factors of the two components are different.

If there is no natural weighing of components then one can still use the canonical ordering over the return space. In this case, however, not all policies will be comparable and so the notion of optimality needs to be adjusted. The natural choice is then *Pareto-optimality*: a policy π is called Pareto-optimal in state x if no other policy can majorize π at x, i.e., if there is no policy π' s.t. $v_{\pi'}(x) \geq v_\pi(x)$. A policy is called Pareto-optimal iff it is Pareto-optimal in each state. It turns out, that Pareto-optimality is equivalent to weighted optimality with appropriately chosen weights and if each component of the evaluation is computed as the total discounted reward for some reward function [Feinberg and Schwartz, 1995, Lemma 7.4]. In the above example, assuming that the amount of consumed food is not truncated, a Pareto-optimal policy would be one for which there is no other policy that would allow the ass to consume more (than the amount ensured by the Pareto-optimal policy) while assuring a smaller number of stolen plates. Pareto-optimality has been studied by many researchers who usually studied conditions which ensured the existence of optimal policies of certain forms.

Apparently the earliest result for dynamic vector-valued models are those of Brown and Strauch [1965], who considered abstract return spaces having a general multiplicative lattice structure and who showed that the "principle of optimality" holds for finite-horizon problems. The results were later extended to infinite horizon problems in many special cases (see, e.g. [Feinberg, 1982, Henig, 1983, Feinberg and Schwartz, 1994]).

In this article we present a general framework based on abstract dynamic programming models, and which is a mixture of the above approaches [Denardo, 1967, Bertsekas, 1977, Littman and Szepesvári, 1996, Szepesvári, 1998]. Namely, we suggest an approach based on the notion of reinforcement-propagating operators, just now these operators will work on function spaces defined over general return spaces with general orderings. In this way we can address constrained problems, lexicographic criteria, lattice return spaces and different reinforcement propagation scenarios within the same framework.

The article is organized as follows: in Section 2 we introduce the concepts necessary for the development and list some basic results concerning the Bellman-optimality equation and the existence of optimal stationary policies. Reinforcement learning algorithms are introduced in Section 3. Some computer experiments, illustrating the theory, are given in Section 4 and conclusions are drawn in Section 5.

2 Abstract ordinal dynamic programming

An Abstract Dynamic Programming (ADP) problem can be given as a 5-tuple $(\mathcal{R}, X, A, \mathcal{A}, \mathcal{Q})$, where X is the state-space of the decision problem, A is the set of actions, $\mathcal{A} : X \to A$, $\mathcal{A}(x)$ are the actions feasible in state x, $\mathcal{R}$ is the return space and $\mathcal{Q} : \mathcal{R}^X \to \mathcal{R}^{X\times A}$ is the so-called reinforcement-propagator operator [Szepesvári, 1998]. In order to explain the meaning of these components consider the problem of Leibniz's ass once again. A simplified representation of that problem could be the following : the ass's state assumes three values: being in the middle, at the left plate, or at the right plate. The plates can be full or empty. One state of the decision problem is composed of the position of the ass, and the state of the plates. So the state space (X) has 12 elements. The actions taken by the ass can be to stay at that position, move left, or move right, so the action space (A) has three elements. The dynamics is given by the following (stochastic) rules: the move actions work as intended. If the ass chooses to stay at a full plate then that plate becomes empty (consuming), if the ass stays at an empty plate then food may appear at that plate according to some fixed stochastic rule and if the ass stays at a plate (either full or empty) then the state of the other plate can change according to some other fixed (stochastic) rule. If the ass is in the middle then none of the plates can become empty in the next step (the ass is guarding the food). The dynamics can be summarized by a random mapping $t : X \times A \to X$ (or, equivalently, as a set of transition probabilities). The ass is considered to be consuming a unit food if it chooses to stay at a full plate. If x_t is the state at time t then the reinforcement streams $\{R_t, S_t\}$ of Equation (1) can be given by $R_t = 1$ if in state x_t the ass is at a full plate and the chosen action, a_t, is "stay", $R_t = 0$, otherwise. Therefore, $R_t = R(x_t, a_t)$ for some function R. Further, $S_t = 1$, if the food disappears from a plate while the ass is at the other plate, otherwise $S_t = 0$. That is, $S_t = S(x_t, a_t, x_{t+1})$, where $x_{t+1} = t(x_t, a_t)$. Let us define the evaluation of a (deterministic, stationary) policy, $\pi : X \to A$, by

$$\begin{aligned}
\tilde{v}_{\pi,1}(x) &= \min\Big(R_{\text{crit}},\, E\big[\sum_{t=0}^{\infty}\gamma^t R_t \,\big|\, x_0 = x\big]\Big), \\
\tilde{v}_{\pi,2}(x) &= E\big[\sum_{t=0}^{\infty}\gamma^t S_t \,\big|\, x_0 = x\big]
\end{aligned}$$

where $E[\cdot]$ is the expectation operator underlying the decision process. Since $\min(R, E[\xi + \eta]) = \min(R, E[\xi] + E[\eta]) = \min(R, E[\xi] + \min(R, E[\eta]))$ holds if $R > 0$ and ξ, η are nonnegative random variables, it follows that

$$\begin{aligned}
\tilde{v}_{\pi,1}(x) &= \min\Big(R_{\text{crit}},\, R(x,\pi(x)) + \\
&\qquad \min\big(R_{\text{crit}}, \gamma\sum_{y\in X} p(x,\pi(x),y)\hat{v}_{\pi}(y)\big)\Big),
\end{aligned}$$

where $\hat{v}_\pi(x)$ is just the expected discounted total cost of executing policy π from state x and $p(x, a, y) = P(y = t(x, a))$. Since $\hat{v}_\pi \geq \tilde{v}_{\pi,1}$, it then follows that $\tilde{v}_{\pi,1}$ satisfies the inequality

$$\begin{aligned}
\tilde{v}_{\pi,1}(x) &\geq \min\Big(R_{\text{crit}}, R(x,\pi(x)) + \\
&\qquad \min\big(R_{\text{crit}}, \gamma\sum_{y\in X} p(x,\pi(x),y)\tilde{v}_{\pi,1}(y)\big)\Big).
\end{aligned}$$

It then follows that if we define v_π as the solution of the fixed point equation

$$v_{\pi,1}(x) = \min\Big(R_{\mathrm{crit}}, R(x,\pi(x)) + \min\big(R_{\mathrm{crit}}, \gamma \sum_{y\in X} p(x,\pi(x),y) v_{\pi,1}(y)\big)\Big), \tag{2}$$

then $v_\pi \le \tilde{v}_\pi$, i.e. v_π provides a lower bound for $\tilde{v}_\pi$. It also follows that if π is such that $v_\pi(x) = R_{\mathrm{crit}}$ then also $\tilde{v}_\pi(x) = R_{\mathrm{crit}}$. Thus if π is such that $v_\pi(x) = R_{\mathrm{crit}}$ then π is also a "satisfactory" solution of the original problem. Since in general $\tilde{v}_{\pi,1}$ cannot be obtained as the solution of a natural fixed point equation, in what follows we shall work with this approximation within the realm of this example. Note that as a result of this approximation we may loose the optimal solutions to the original problem.

It should be clear that $\tilde{v}_{\pi,2}$ can be written recursively:

$$\tilde{v}_{\pi,2}(x) = \sum_{y\in X} p(x,\pi(x),y)\left\{S(x,\pi(x),y) + \gamma \tilde{v}_{\pi,2}(y)\right\}. \tag{3}$$

Now, if one defines $\mathcal{Q}$ by

$$\begin{aligned}(\mathcal{Q}v)(x,a)_1 &= \min\Big(R_{\mathrm{crit}}, R(x,a) + \min\big(R_{\mathrm{crit}}, \gamma \sum_{y\in X} p(x,a,y) v_1(y)\big)\Big),\\ (\mathcal{Q}v)(x,a)_2 &= \sum_{y\in X} p(x,a,y)\left\{S(x,a,y) + \gamma v_2(y)\right\}\end{aligned}$$

and $T_\pi : \mathcal{R} \to \mathcal{R}$ by $(T_\pi v)(x) = (\mathcal{Q}v)(x,\pi(x)), \quad x \in X,$ then we see that v_π is just the fixed point of T_π. Note that the definition of $\mathcal{Q}$ is obtained from (2) and (3) by systematically replacing $\pi(x)$ by a, and v_π ($\tilde{v}_\pi$) by v everywhere in the equations. Observe that $\mathcal{Q}$ provides a concise summary of *both* the state- and reinforcement-dynamics of the decision process. Note that using $\mathcal{Q}$ alone it is possible to define the evaluation for general classes of policies and it is possible to show that the corresponding fixed point will still hold true [Szepesvári, 1998].

Policies are compared on the basis of their evaluations. Since now $v_\pi(x) \in \mathcal{R} = \mathbf{R}^2$ is vector-valued we need a way to compare pairs of vectors. Therefore, we will assume that a binary relation $\le$ over $\mathcal{R}$ is given which is reflexive, transitive and trichotomous (i.e., $\le$ is an *ordering*, or $\mathcal{R} = (\mathcal{R}; \le)$ is a *lattice*) [2] In our example we can take a "reverse-2nd" *lexicographic ordering*: $r \le r'$ if $r_1 < r'_1$ or if $r_1 = r'_1$ then $r_2 \ge r'_2$ (here the components of r and r' were denoted by lower indices). This finishes the construction of the ADP describing the problem-structure of Leibniz's ass. This "reverse-2nd" lexicographic ordering differs from lexicographic ordering only by the condition on the second components: we wrote $r_2 \ge r'_2$ instead of $r_2 \le r'_2$. For convenience, we will continue with considering lexicographic ordering. Lexicographic ordering (and also "reverse-2nd" ordering) satisfies the above properties, i.e., it is an ordering.

Now, what are the optimal policies in an ADP? In order to facilitate the connection with RL we will define the notion of optimal reinforcement function (instead of relying on Pareto-optimality), but first we need to assign a meaning to the supremum of subsets of $\mathcal{R}$: for $A \subset \mathcal{R}$, $a = \mathrm{s.u.p.}\, A$ is a value such that for all $c \ge A$, also $c \ge a$ ($a \ge b$ is defined by $b \le a$, and $a \ge A$ is defined as $a \ge a'$ for all $a' \in A$). The infimum of sets is defined similarly. A lattice $(\mathcal{R}; \le)$ is said to be *complete* if for all bounded subsets A, both the infimum and the supremum of the set exist. Lexicographic ordering is complete: for example, the supremum a^* of $A \subset \mathbf{R}^2$ can be defined in a standard way as follows: $a^*_1 = \sup\{a_1 : a = (a_1, a_2)^T \in A\}$ and $a^*_2 = \inf\{a_{2n} : a_n = (a_{1n}, a_{2n})^T \in A \,\mathrm{s.t.}\, a_{1n} \to a^*_1\}$. In order to deal with the supremum of arbitrary subsets of $\mathcal{R}$ we need to extend the return space $\mathcal{R} = \mathbf{R}^2$ to $\hat{\mathbf{R}}^2$, where $\hat{\mathbf{R}} = \{-\infty, +\infty\} \cup \mathbf{R}$ is the set of extended reals with the natural topology. The ordering $\le$ of $\mathcal{R}$ can be extended to functions with values in $\mathcal{R}$ in the natural way: for $v, w \in \mathcal{R}^Y$ we say that $v \le w$ iff for all $y \in Y$, $v(y) \le w(y)$ holds. Note that $\le$ over $\mathcal{R}^Y$ is only a partial ordering (i.e., it is not total).

Equipped with the notion of supremum we can define the optimal reinforcement function:

$$v^*(x) = \underset{\pi\in\Pi}{\mathrm{s.u.p.}}\, v_\pi(x), \quad x \in X. \tag{4}$$

Here Π denotes a fixed set of policies. We will consider the case when Π equals to the set of all stationary policies. A policy in the class Π is said to be *optimal* if $v_\pi = v^*$.

The following theorem, proven in the appendix, gives a characterisation of optimal deterministic stationary policies. For further discussions related to various forms of optimal policies see [Gábor et al., 1998].

[2] A binary relation $\le$ over $\mathcal{R}$ is called *i) reflexive* if $r \le r$ for any $r \in \mathcal{R}$; *ii) transitive* if r, r' and r'' are such that $r \le r'$ and $r' \le r''$ then $r \le r''$ ($r, r', r'' \in \mathcal{R}$); and *iii) trichotomous* if for any pairs $(r, r') \in \mathcal{R}$ either $r \le r'$ or $r' \le r$ (the ordering is *total*) and if both relations hold then $r = r'$.

Theorem 2.1 *Consider a finite*[3] *ADP $(\mathcal{R}, X, A, \mathcal{A}, \mathcal{Q})$, where (i) $(\mathcal{R}; +, \lambda\cdot, \|\cdot\|_{\mathcal{R}})$ is a Banach-space and $\mathcal{R}$ is equipped with (ii) a complete ordering $\le$ which satisfies the following* countable transitivity property*: (iii) if r_n is weakly convergent*[4] *in $\mathcal{R}$, and $r_0 \le r_1 \le r_2 \le \dots r_n \le r_{n+1} \le \dots$ then $r_0 \le \lim_{n\to\infty} r_n$. Further, assume that (iv) $\mathcal{Q} : \mathcal{R}^X \to \mathcal{R}^{X\times A}$ is monotone: $\mathcal{Q}v \le \mathcal{Q}w$ whenever $v \le w$, $v, w \in \mathcal{R}^X$, continuous in the topologies induced by pointwise convergence over $\mathcal{R}^X$ and $\mathcal{R}^{X\times A}$, (v) and that it is a contraction w.r.t. the induced max-norm*[5] *$\|\cdot\|_{\infty,\mathcal{R}}$. (vi) Assume that $T : \mathcal{R} \to \mathcal{R}$, defined by*

$$(Tv)(x) = \underset{a\in\mathcal{A}(x)}{\mathrm{m.a.x.}} (\mathcal{Q}v)(x, a) \tag{5}$$

has a unique fixed point v^+, and $\lim_{n\to\infty} T^n v = v^+$ for all $v \in \mathcal{R}^X$ s.t. $\|v\|_{\infty,\mathcal{R}} < \infty$. Let $\Pi = A^X$ be the space of stationary policies. Then (a) *$v^+ \ge v_\pi$ for all π (π is a deterministic stationary policy) and $v^+ = v^*$, so $Tv^* = v^*$ (Bellman optimality equation);* (b) *if $T_\pi v^+ = Tv^+$, i.e., if π is myopic w.r.t. v^+, then $v_\pi = v^*$ (myopic policies are optimal);* (c) *if $T_{\pi'} v_\pi > v_\pi$ then $v_{\pi'} \ge v_\pi$ (Howard's policy improvement routine is valid).*

Operator T, as defined by (5), is called the *optimal value operator.*

It is easy to check that countable transitivity holds for sequences of $\mathbf{R}^n$ and the lexicographic ordering. Note that contraction arguments cannot be used since there is no norm over $\mathbf{R}^n$ with the lexicographic ordering for which the m.a.x. operator would be a non-expansion. For a further discussion of this and additional peculiarities related to lexicographic orderings see [Gábor et al., 1998].

Note that if $\mathcal{R} = \mathbf{R}^n$ with the lexicographic ordering then the actions at which the maximum is reached in Eq. (5) can be computed by first computing the sets

$$A_{i+1} = \{\, a \in A_i(x) \mid \max_{b\in A_i(x)} (\mathcal{Q}f)(x, b)_i = (\mathcal{Q}f)(x, a)_i \,\}$$

recursively for $i = 0, 1, 2, \dots, n-1$ with $A_0 = \mathcal{A}(x)$. For convenience, we will denote the action sets as defined above by $A_i(Q, x)$ when $\mathcal{Q}f$ is replaced by any function $Q \in \mathcal{R}(X \times A)$:

$$\begin{aligned} A_0(Q, x) &= \mathcal{A}(x) \\ A_{i+1}(Q, x) &= \{\, a \in A_i(Q, x) \mid \\ &\qquad \max_{b\in A_i(Q,x)} Q(x, b)_i = Q(x, a)_i \,\}, \end{aligned}$$

where $i = 0, 1, 2, \dots, n-1$. Then $(Tv)(x)_{i+1} = \max_{a\in A_i(\mathcal{Q}v,x)} (\mathcal{Q}v)(x, a)_{i+1}$. Now we show that T has a unique fixed point and $T^n v$ converges to this fixed point for all bounded $v \in \mathcal{R}^X$ provided that $\mathcal{Q}$ satisfies the conditions of the theorem and if $\mathcal{Q}$ acts *componentwise*, i.e., if $(\mathcal{Q}v)_i = (\mathcal{Q}w)_i$ whenever $v_i = w_i$. Fix v and consider the first component of $T^n v$. Define $T_1 : \mathbf{R}^X \to \mathbf{R}^X$ by $T_1 f = (T\hat{f})_1$, where $\hat{f} = (f, f_2, \dots, f_n)$ with $f_2, \dots, f_n$ being arbitrary. T_1 is well defined and is a contraction. Moreover, $(T^n v)_1 = T_1^n v_1$ holds for all $n \in \mathbf{N}$, and therefore $(T^n v)_1$ converges to the unique fixed point of T_1. Similarly, if u and w are both fixed points of T then $u_1 = w_1$. Let us denote this common value by v_1^+. Now, consider $(T^n v)_2$. Since $(T^{n+1}v)_2(x) = \max_{a\in A_1(\mathcal{Q}T^n v,x)} (\mathcal{Q}T^n v)(x, a)_2$, and since $\mathcal{Q}$ is componentwise, $A_1(\mathcal{Q}T^n v, x)$ depends only on $(T^n v)_1$ which is known to converge. Therefore, because of the finiteness of A, for n large enough $A_1(\mathcal{Q}T^n v, x)$ will stabilize at some set $A_1^*(v, x)$. Now, since the operator $u(x) \mapsto \max_{a\in A_1^*(v,x)} (\mathcal{Q}\hat{u})(x, a)_2$ is a contraction, where $\hat{u} = (v_1^+, u, u', \dots)$, also $(T^n v)_2$ converges to some value (the operator is well defined since $\mathcal{Q}$ is componentwise). Moreover, if u and w are both fixed points of T then $u_1 = w_1$ and thus $A_1(\mathcal{Q}u, x) = A_1(\mathcal{Q}v, x) (= A_1^*(x))$ for all $x \in X$, and so u_2 and w_2 are both the fixed points of the contraction $z \mapsto \max_{a\in A_1^*(x)} (\mathcal{Q}\hat{z})(x, a)_2$ and are therefore equal. Continuing in this way for the higher indices we get the proof of the required statement. Note that this argument shows the problem of Leibniz's ass is indeed in the realm of Theorem 2.1.

Theorem 2.1 is just one example of how the existence of optimal stationary policies can be ensured in multi-criteria problems. There are many possible extensions of it, but these are outside of the scope of the present article.

3 Learning optimal policies

Since most convergence proofs for RL algorithms rely on contraction arguments the generalization of results like the convergence of such as the Adaptive Real-Time Dynamic Programming [Barto et al., 1991], Q-learning [Watkins, 1990], TD(λ) [Sutton, 1988] are easy to obtain for vector-valued MDPs *provided* that T is a

[3] An ADP $(\mathcal{R}, X, A, \mathcal{A}, \mathcal{Q})$ is called finite if both X and A are finite. The finiteness assumption in this theorem can be relaxed by some extra work.

[4] A sequence r_n is said to be weakly convergent in $\mathcal{R}$ if it is convergent in the topology induced by the vector space structure of $\mathcal{R}$.

[5] The induced maximum-norm $\|\cdot\|_{\infty,\mathcal{R}}$ is defined by $\|v\|_{\infty,\mathcal{R}} = \sup_{z\in Z} \|v(z)\|_{\mathcal{R}}$.

contraction[6]. Unfortunately, this holds rarely. Nevertheless a successive componentwise analysis, like the one presented at the end of the previous section will in general yield the desired convergence result.

As a particular example consider the case of Q-learning. Let $Q^* = \mathcal{Q}v^*$ be the optimal action-value function. Q-learning solves the fixed point equation $Q^* = \mathcal{Q}SQ^*$, $(SQ)(x) = \text{m.a.x.}_{b\in\mathcal{A}(x)} Q(x,b))$, by relaxation and without ever estimating $\mathcal{Q}$. In the case of an MDP with the expected discounted total cost Q-learning takes the form

$$\begin{aligned} Q_{t+1}(x_t,a_t) &= (1-\alpha_t(x_t,a_t))Q_t(x_t,a_t) + \\ &\quad \alpha_t(x_t,a_t)\big\{R_t(x_t,a_t,x_{t+1}) + \\ &\quad \gamma \max_{b\in\mathcal{A}(x_t)} Q_t(x_{t+1},b)\big\}, \end{aligned}$$

with $Q_{t+1}(x,a) = Q_t(x,a)$ for pairs $(x,a) \neq (x_t,a_t)$. The relaxation factor (learning rate) $0 < \alpha_t(x_t,a_t) < 1$ is gradually decreased towards zero so that the variance of the estimates are reduced and (probability one) convergence can be achieved.

The raw generalization of Q-learning to vector-valued Q-learning replaces the immediate-reward scalars (R_t) in the above equation by immediate-reward vectors and "max" by "m.a.x.". For simplicity, consider a two-dimensional return space with the lexicographic ordering. The update equation for the first component remains unchanged, but the update of the second component becomes

$$\begin{aligned} Q_{t+1,2}(x_t,a_t) &= (1-\alpha_t(x_t,a_t)Q_{t,2}(x_t,a_t) + \\ &\quad \alpha_t(x_t,a_t)\big\{R_{t,2}(x_t,a_t,x_{t+1}) + \\ &\quad \gamma \max_{b\in A_1(Q_t,x_t)} Q_{t,2}(x_{t+1},b)\big\}. \end{aligned}$$

The raw componentwise generalization of Q-learning would employ (erronously) $\mathcal{A}(x_t)$ instead of $A_1(Q_t,x_t)$.

The analogue of Q-learning for MDPs with the maximin criterion, proposed by Heger [Heger, 1994, 1996], is the Q-hat algorithm defined as

$$\begin{aligned} Q_{t+1}(x_t,a_t) &= \min\big\{Q_t(x_t,a_t), R_t(x_t,a_t,x_{t+1}) + \\ &\quad \gamma \max_{b\in A} Q_t(x_{t+1},b)\big\}. \end{aligned}$$

This algorithm will converge to the optimal Q-function if $Q_0 \geq Q^*$ (the initial estimate is optimistic). The raw generalization replaces "min" and "max" with "m.i.n." and "m.a.x.", respectively. Unfortunately, this iteration may fail to converge to Q^* since the convergence of Q-hat exploits $Q_t \geq Q^*$ $(t \geq 0)$ and this may fail in this case.[7] In order to surmount this problem one has to update the second and larger index components by some means other than Q-hat learning.

It is natural then to consider adaptive real-time dynamic programming algorithms. For maximin problems this algorithm builds an estimate of the transition sets $T(x,a) = \{\, y \in X \,|\, p(x,a,y) > 0 \,\}$ and another estimate of the rewards $R(x,a,y)$. Since there is no "optimistic initialization" condition here, one may show (using successive componentwise analysis) that the composite algorithm converges to optimality if some other conditions, basically ensuring "sufficient exploration", hold. Further discussions related to action-selection strategies ensuring "sufficient exploration" in minimax problems can be found in [Gábor et al., 1998].

4 Computer simulations

The purpose of the computer simulations was twofold: to demonstrate that the theory works in practice, and to provide some hint on the rate of convergence of different algorithms.The ARTDP algorithm were tried out for tic-tac-toe with lexicographic ordering and the first criterion prescribing the desire to win (or make a draw) and the second to finish the game as soon as possible[8]. The action selection procedure was the greedy policy in all of the cases. Several opponents were tried whose stategy was a mixture of the optimal-policy and a totally randomized one. The degree of

[6] In fact, since the convergence of the vast majority of RL algorithms follows from the general asynchronous contraction-mapping theorem of [Littman and Szepesvári, 1996] (see also [Szepesvári and Littman, 1998]), it is sufficient to reproduce the proof of that theorem. It turns out, that the raw generalization of that proof will work without any problems *for contractions*. However, this is out of the scope of this article.

[7] This can be shown in the following way: Consider again $\mathcal{R} = \mathbf{R}^2$ with the lexicographic ordering. Then $Q_{t+1,2}(x_t,a_t) = \min\big\{Q_{t,2}(x_t,a_t), R_{t,2}(x_t,a_t,x_{t+1}) + \gamma \max_{b\in A_1} Q_{t,2}(x_{t+1},b)\big\}$, where $A_1 = A_1(Q_t,x_t)$. Notice that $Q_{t+1,2}(x,a) \leq Q_{t,2}(x,a)$ for all $(x,a) \in U$ so if once $Q_{t,2}(x,a) < Q_2^*(x,a)$ then $Q_{t,2}(x,a)$ cannot converge to $Q_2^*(x,a)$. Here, $A_1(Q_t,x_t)$ may be quite different from $A_1(Q^*,x_t)$ which means that $Q_{t+1,2}(x_t,a_t)$ may become smaller than $Q^*(x_t,a_t)$ even if $Q_{t,2} = Q_2^*$, depending only on the values of $Q_{t,1}$.

[8] The first component of the reinforcement-vector was $+1$ if the learner won, 0 if the game was a draw and -1 if he lost the game. The second component was unity in each step. We used the well known minimax representation of alternating games [see e.g. Littman and Szepesvári, 1996]. Note that by a simple change to the lexicographic ordering one may consider another criterion when the learner minimizes the number of steps only when starting from winning states, otherwise trying to mark time.

	0	0.25	0.5	0.75	1
ARTDP	0.73	0.74	0.74	0.76	0.74
	3.55	4.2	4.18	4.18	4.19
MC-ARTDP	0.85	1	0.96	1	1
	3.59	3.28	3.29	3.28	3.28

Table 1: Results of exhaustive testing. Percent of optimal moves learnt, and average number of steps to the end of the game for cases when the learner won are shown for both learners learning with ARTDP and MC-ARTDP. In the first raw the degree of randomness of the opponents are shown: a randomness of 0 means an optimal opponent, while the randomness of 1 means a perfectly random opponent. The results suggest that since the learners do not explore, a complete optimal policy cannot be learned against the perfect opponent (just part of the game-tree is explored). The number of steps until the end of the game are consistently smaller for MC-ARTDP than that of for ARTDP. Also MC-ARTDP can win a larger percent of games.

randomness was set to 0, 0.25, 0.5, 0.75 and 1, so that the first opponent, corresponding to randomness 0, is the optimal one, while the last one is the totally randomized one. For comparison both the multi-criteria and single criterion ARDTP algorithms were tried (called MC-ARTDP and ARTDP, respectively.) The learner started the game in each trial. The percent of wins and draws, and the number of steps in the cases of won or drew games are shown in Table 1. The percents are computed by employing an exhaustive search, i.e., we measured the percent of those leaves in the *full reachable* game-tree when our learner did not lose the game. It is clear that MC-ARTDP performs better than the ARTDP algorithm in all of the cases, i.e., it could explore a larger part of the game-tree. The reason for this is that MC-ARTDP uses more information than ARTDP. In particular, since the second components of its evaluation function are initialized to zero, initially unexplored actions will look more favourable than explored ones, meaning that dependence on the second component will facilitate exploration. To confirm this observation we ran another set of experiments using the ARTDP algorithm and when actions were chosen based on one of the following two well-known exploration stategies: the Boltzmann-exploration and the ϵ-greedy strategy with decaying exploration[9]. In this case ARTDP yielded comparable results to that of MC-ARTDP, thus confirming the hypothesis.[10]

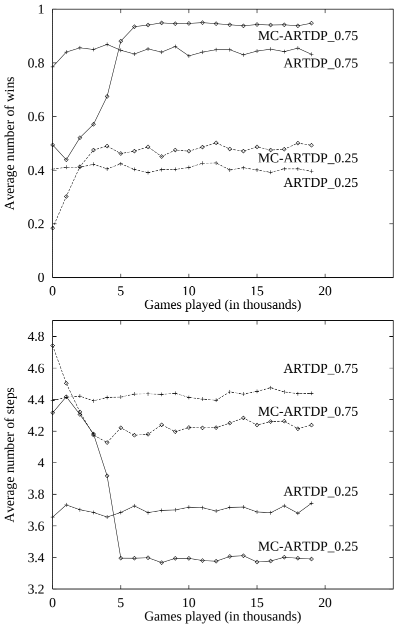

Figure 1: Results of learning with the one-criterion and multi-criteria ARTDP algorithms against opponents of different strengths. MC-ARTDP-0.25 and MC-ARTDP-0.75 label the curves of MC-ARTDP for an opponent with randomness 0.25 and 0.75 , respectively. corresponding to the raw ARTDP algorithms. The figures clearly show that it is easier to win against weaker opponents

Exploration has a price, though. The more exploratory actions the player tries the larger is the number of games lost during the learning trials. In order to get a more complete picture about the performances of the two algorithms we have measured on-line (or during-learning) performance. Results are shown in Figures 1. The l.h.s. subfigure shows the percent of plays won or drew. The larger the convergence speed to 1 is the smaller is the cost of exploration. The r.h.s. subfigure depicts the number of steps until the end of the game, for the games when our learner actually won. Both figures show results for the opponents with randomness 0.25 and 0.75 (results for the other cases can be roughly obtained by intra- and extrapolations and

[9]The ϵ-greedy exploration stategy chooses the best-looking (greedy) action with probability $1 - \epsilon$ and chooses an action uniformly randomly from the rest with probability ϵ [Thrun, 1992].

[10]In theory, as time goes to infinity both algorithms will converge to optimality. So the worse than optimal results should not be considered as cases when the algorithms got stuck in "local minima".

are not shown). Note that both the ARTDP and MC-ARTDP learn faster against weaker opponents which could be accounted for the small average depth of game tree when playing against a weak opponent. Note that this the learner trained against a weak opponent will probably fail to win over a strong one, and the reverse may hold, too: in order to learn the optimal minimax strategy the opponents should not be restricted[11]. Also, in the case of both opponents MC-ARTDP learns slightly slower (in the short-term) but results in a better policy in the medium-term. More experiments are needed to confirm these findings.

5 Conclusions

We have considered multi-criteria decision problems using the framework of abstract dynamic programming. The reinforcements were assumed to be vector-valued and were compared by a given total ordering defined over the corresponding vector space. A result, showing the existence of optimal policies was derived and it was shown that it applies to lexicographic ordering with "componentwise reinforcement propagation". Next, reinforcement learning algorithms were derived and we have argued that, in the case of lexicographic ordering, their convergence can be proven by a method which we termed "successive componentwise analysis". Experimental results were presented to illustrate the working of the algorithms. In the future we plan to extend the results and run other simulations to reinforce the utility of multi-criteria learning.

Acknowledgements

This work was partially done while Cs.Sz. was with the Research Group of Artificial Intelligence. This work was partially supported by OTKA Grant No. F20132 and the Hungarian Ministry of Education Grant No. FKFP 1354/1997.

Appendix

Here we prove Theorem 2.1, the text of which is not repeated here because of lack of space. Firstly, we shall prove that v^+, the unique fixed point of T, majorizes the optimal value function, v^*. Fix an arbitrary policy π and observe that $Tv_\pi \geq T_\pi v_\pi$. Since $T_\pi v_\pi = v_\pi$, also $Tv_\pi \geq v_\pi$. From this, and because of the monotonicity of T (which holds because A is finite), we obtain $T^2 v_\pi \geq Tv_\pi \geq v_\pi$. Iterating this indefinitely, we get that $T^{n+1} v_\pi \geq T_\pi^n v_\pi \geq \ldots \geq v_\pi$ holds for all $n \in \mathbf{N}$. Thus, $T^n v_\pi$ is monoton increasing and thus (by the countable transitivity assumption) $\lim_{n\to\infty} T^n v_\pi \geq v_\pi$. Now, since $\lim_{n\to\infty} T^n v_\pi = v^+$, so $v^+ \geq v_\pi$. Since π was arbitrary, it follows that $v^+ \geq v^*$ by the definition of the s.u.p. operator. Now, let π be a policy which is myopic w.r.t. v^+: $T_\pi v^+ = Tv^+$. Since $Tv^+ = v^+$, so $T_\pi v^+ = v^+$. Now, since v_π is the unique fixed point of T_π (T_π is a contraction since $\mathcal{Q}$ is a contraction), we get that $v^+ = v_\pi$. This shows that $v^+ = v^*$ and that π is optimal. In order to prove the third part consider a pair of policies (π, π') s.t. $T_{\pi'} v_\pi > v_\pi$. By the first train of thoughts, we get that $T_{\pi'}^n v_\pi \geq v_\pi$ is a monotone increasing sequence, so that $v_{\pi'} = \lim_{n\to\infty} T_{\pi'}^n v_\pi \geq v_\pi$ holds, too, thus finishing the proof.

[11] Since the opponents are randomized (except the optimal opponent) the algorithms would converge to optimality. However, the convergence rate will still depend on the degree of randomness of the opponent. The convergence rate will depend on how fast can the part of the game-tree which is accessible for an optimal player be fully explored. For opponents with higher randomness deep parts can hardly be accessed, for opponents with small randomness parts that follow an initial sub-optimal choice will be hard to explore.

References

E. Altman and A. Schwartz. Adaptive control of constrained Markov chains: Criteria and policies. *Annals of Operations Research*, 28:101–134, 1991.

M. Asada, E. Uchibe, S. Noda, S. Tawaratsumida, and K. Hosoda. Coordination of multiple behaviors acquired by a vision-based reinforcement learning. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robot and Sytems*, volume 2, pages 917–924, 1994.

A.G. Barto, S.J. Bradtke, and S.P. Singh. Real-time learning and control using asynchronous dynamic programming. Technical report 91-57, Computer Science Department, University of Massachusetts, 1991.

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

T.A. Brown and R.E. Strauch. Dynamic programming on multiplicative lattices. *J. Math. Anal. and App.*, 12:364–370, 1965.

E.V. Denardo. Contraction mappings in the theory underlying dynamic programming. *SIAM Rev.*, 9:165–177, 1967.

E.A. Feinberg. Controlled Markov decision process with arbitrary numerical criteria. *Theory of Probability and Applications*, 27:486–503, 1982.

E.A. Feinberg and A. Schwartz. Markov decision models with weighted discounted rewards. *Mathematics of Operations Research*, 19:152–168, 1994.

E.A. Feinberg and A. Schwartz. Constrained Markov decision models with weighted discounted rewards. *Mathematics of Operations Research*, 20(2):302–320, 1995.

E.B. Frid. On optimal strategies in control problems with constraints. *Theory of Probability and Applications*, 17: 188–192, 1972.

Z. Gábor, Zs. Kalmár, and Cs. Szepesvári. Multi-criteria reinforcement learning. Technical report 98-115, Research Group on Artificial Intelligence, JATE-MTA, 1998.

M. Heger. Consideration of risk in reinforcement learning. Revised submission to the 11th International Machine Learning Conference ML-94, 1994.

M. Heger. The loss from imperfect value functions in expectation-based and minimax-based tasks. *Machine Learning*, 22:197–225, 1996.

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

D. Heyman and M. Sobel. *Stochastic Models in Operations Research: Stochastic Optimization*, volume 2. McGraw-Hill, New York, 1984.

M.L. Littman and Cs. Szepesvári. A Generalized Reinforcement Learning Model: Convergence and applications. In *Int. Conf. on Machine Learning*, pages 310–318, 1996.

L.G. Mitten. Composition principles for synthesis of optimum multi-stage processes. *Operations Research*, 12: 610–619, 1964.

S.M. Ross. *Applied Probability Models with Optimization Applications.* Holden Day, San Francisco, California, 1970.

S. Singh and D. Cohn. How to dynamically merge Markov decision processes. In *Advances in Neural Information Processing Systems 11*, Cambridge, MA, 1997. MIT Press. in press.

M.J. Sobel. Ordinal dynamic programming. *Management Science*, 21:967–975, 1975.

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

Cs. Szepesvári. Non-Markovian policies in sequential decision problems. *Acta Cybernetica*, 1998. accepted.

Cs. Szepesvári and M.L. Littman. A unified analysis of value-function-based reinforcement-learning algorithms. *Neural Computation*, 1998. accepted.

S.B. Thrun. *The role of exploration in learning control.* Van Nostrand Rheinhold, Florence KY, 1992.

C.J.C.H. Watkins. *Learning from Delayed Rewards.* PhD thesis, King's College, Cambridge, 1990.