Soft Dynamic Programming Algorithms: Convergence Proofs

Satinder P. Singh

Department of Computer Science
University of Massachusetts
Amherst, MA 01003
singh@cs.umass.edu

Algorithms based on dynamic programming (DP) find optimal solutions to finite-state optimal control tasks by iterating a "backup" operator that only considers the consequences of executing the "best" action in a state. In many problem domains, the optimal solution may be "brittle" and it may be desirable to find robust, if suboptimal, solutions that prefer states that have *many* "good" actions to choose from, over states that have a *single* good action. I present a family of iterative approximation algorithms constructed by replacing the "hard" max operator in classical DP algorithms by a "soft" *generalized means* of order p operator (Rivest [8]) (e.g., a non-linearly weighted l_p norm). The soft DP algorithms converge to solutions that are more robust than solutions found by classical DP algorithms. I prove that for each index $p \geq 1$, the corresponding soft DP algorithm converges to a unique fixed point, and that the approximate but robust solution gets uniformly better as the index p is increased, converging in the limit ($p \rightarrow \infty$) to the optimal solution determined by classical DP algorithms.

Based on Poster at CLNL-93

Soft Dynamic Programming Algorithms: Convergence Proofs

Satinder P. Singh

Department of Computer Science
University of Massachusetts
Amherst, MA 01003
satinder@cs.umass.edu

1 Introduction

Optimal control problems are characterized by the fact that control decisions taken at the present time affect the behavior of the controlled system at future times. Therefore, the solution is not just a decision at the present time instant, but a sequence of decisions over the entire duration of control. The goal is to find an optimal *control policy*, i.e., a function assigning actions (or decisions) to states, that optimizes an objective functional, such as minimum time, or minimum cost. In many optimal control problems, the optimal solution may be *brittle* in that it may not leave the controller any room for even the slightest error. For example the minimum time solution in a navigation problem may take an expensive robot over a narrow ridge where the slightest deviation from the optimal action can lead to disaster.

In this paper we propose new algorithms that find solutions in which states that have *many* "good" actions to choose from are preferred over states that have a *single* good action choice. This *robustness* is achieved by potentially sacrificing optimality. Robustness can be particularly important if the there is mismatch between the model and the real physical system, or if the real system is non-stationary, or if availability of control actions varies with time.

I thank Peter Dayan and Andrew Barto for their contributions to this paper. This work was supported by the Air Force Office of Scientific Research, Bolling AFB, under Grant AFOSR-89-0526 and by the National Science Foundation under Grant ECS-8912623.

2 Dynamic Programming Algorithms

Dynamic programming (DP) [2] provides an approach to solving optimal control problems under general conditions. The DP solution specifies the optimal control at every state of the system at every instant of time, and is often implemented as a feedback (closed-loop) controller in which the state of the system is constantly measured and the corresponding control applied. The DP approach is based on Bellman's [2] Principle Of Optimality which states that "an optimal policy has the property that whatever the initial state and decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision." This converts the simultaneous determination of the entire optimal control policy into a sequential determination. Subsequently, a number of iterative improvement algorithms have been defined that organize their computation more efficiently than exhaustive search through the space of closed-loop policies.

In searching for the optimal control policy, DP-based algorithms employ the *max* operator that is a "hard" operator because it only considers the consequences of executing the best action in a state and ignores the fact that all the other actions could be disastrous (see Equation 2). This paper introduces a family of iterative approximation algorithms constructed by replacing the hard max operator in DP-based algorithms by "soft" *generalized means* [5] of order p (e.g., a non-linearly weighted l_p norm). These soft DP algorithms converge to solutions that are more robust than those of classical DP. I prove that for each index $p \geq 1$, the corresponding iterative algorithm converges to a unique fixed point, and that the approximation gets uniformly better as the index p is increased, converging in the limit ($p \to \infty$) to the DP solution. The main contribution of this paper is the new family of approximation algorithms and their convergence results. The implications for neural network researchers are also discussed.

Markovian Decision Tasks (MDTs) (e.g., [3]) are a subset of optimal control tasks important to learning researchers because many interesting sequential learning tasks can be formulated as multi-stage MDTs. Let S be the set of states of the system and A be the set of actions available to the controller. Executing action $a \in A$ in state $x \in S$ results in payoff $r(x, a)$ and causes a transition to state y with probability $P_{xy}(a)$. The objective functional, $J_n = E[\sum_{t=0}^{n} \gamma^t r_t]$, where $0 < \gamma < 1$ is the discount factor, n is the horizon or the number of stages in the task, E is the expectation operator, and r_t is the payoff received at time step t.

In this paper, I will develop approximations to DP algorithms for solving finite state, infinite-horizon MDTs. Implications of relaxing these constraints are discussed in Section 4. For stationary policy $\pi : S \to A$, the return $V_\pi(x) = E[\sum_{t=0}^{\infty} \gamma^t r_t(\pi)]$, where $r_t(\pi)$ is the payoff received by the controller at time t when the start state is x and the policy π is followed forever. An optimal policy π^* maximizes the return for each state. I will denote V_{π^*} simply as V^*.

For MDTs, Bellman's Optimality Principle can be stated as follows; $\forall\, x \in S$:

$$V^*(x) \;=\; \max_{a \in A} [r(x,a) + \gamma \sum_{y \in S} P_{xy}(a) V^*(y)]. \tag{1}$$

Value Iteration (e.g., [3]) is an iterative improvement algorithm, $V_{t+1} = T(V_t)$, for computing the optimal value function V^*. V_t is the estimate of V^* at the t^{th} iteration, and the operator $T : (\Re^+)^{|S|} \rightarrow (\Re^+)^{|S|}$ is defined as follows:

$$T(V_{t+1})(x) \;=\; \max_{a \in A} [r(x,a) + \gamma \sum_{y \in S} P_{xy}(a) V_t(y)] \tag{2}$$

Note that the operator T in Equation 2 is "hard". Following Rivest[8], who replaced the min-max operators in game tree search with generalized means, I replace the max operator with the generalized mean for the more general class of DP-based algorithms.

2.1 Facts about Generalized Means

Let $A = \{a_1, a_2, \ldots, a_n\}$, and $A' = \{a'_1, a'_2, \ldots, a'_n\}$. Define $A_{\max} = \max\{a_1, a_2, \ldots, a_n\}$, and $||A||_\infty = \max\{|a_i|, |a_2|, \ldots, |a_n|\}$. Define $A_p = [\frac{1}{n}\sum_{i=1}^{n}(a_i)^p]^{\frac{1}{p}}$, called a generalized mean of order p. The following facts are proved in Hardy *et al.* [5] under the conditions, $1 \leq i \leq n$, $a_i, a'_i \in \Re^+$:

Fact 1. (Convergence) $\lim_{p \rightarrow \infty} A_p = A_{\max}$.

Fact 2. (Differentiability) While $\frac{\partial A_{\max}}{\partial a_i}$ is not defined, $\frac{\partial A_p}{\partial a_i} = \frac{1}{n}[\frac{a_i}{A_p}]^{p-1}$ for $0 < p < \infty$.

Fact 3. (Uniform Improvement) $0 < p < q \Rightarrow A_p \leq A_q \leq A_{\max}$; further if $\exists\, i, j$, s.t. $a_i \neq a_j$, then $0 < p < q \Rightarrow A_p < A_q < A_{\max}$.

Fact 4. (Monotonicity) if $\forall\, i, a_i \leq a'_i$, then $A_p \leq A'_p$. In addition, if $\exists\, i$, s.t. $a_i < a'_i$, then $A_p < A'_p$.

Fact 5. (Boundedness) If $p \geq 1$, and if $||A - A'||_\infty \leq M$, i.e., the two different sequences of n numbers differ at most by M, then $|A_p - A'_p| \leq M$. In addition, if $p > 1$, and $A \neq A'$, then $||A - A'||_\infty \leq M \Rightarrow |A_p - A'_p| < M$. See Appendix for a proof.

3 New Iterative Fixed Point Algorithms

A family of iterative improvement algorithms, indexed by a scalar parameter p, can be defined as $V_{t+1} = T_p(V_t)$, where the update operator $T_p : (\Re^+)^{|S|} \rightarrow (\Re^+)^{|S|}$:

$$T_p(V_{t+1})(x) \stackrel{\text{def}}{=} \left\{ \frac{1}{|A|} \sum_{a \in A} [r(x,a) + \gamma \sum_{y \in S} P_{xy}(a) V_t(y)]^p \right\}^{\frac{1}{p}}. \quad (3)$$

Note that even though in this abstract the new algorithms are developed by modifying the Value Iteration algorithm, similar modifications can be made to most other DP algorithms as well as reinforcement learning algorithms [1] that involve the max operator and are related to DP, such as Watkins's popular *Q-learning* [11] algorithm (see Appendix).

Fact 7. By the "Convergence" (Fact 1) property of the generalized mean operator, $\lim_{p \rightarrow \infty} T_p = T$.

3.1 Convergence Results

Fact 8. For a discrete MDT the finite set of stationary policies form a partial order under the relation $>$: $\pi > \pi' \Rightarrow \forall\, x \in S,\, V_\pi(x) > V_{\pi'}(x)$. If $0 \leq \gamma < 1$, and a finite constant $\Delta \in \Re$ is added uniformly to all the payoffs, the partial order of the policies does not change.

Assumption 1. $0 < \gamma < 1$

Assumption 2. $\forall\, x \in S, \forall\, a \in A,\, r(x,a) \geq 0$. This is not a restriction for MDTs with $0 \leq \gamma < 1$, because of Fact 8.

The development of the convergence proofs follows Bertsekas and Tsitsiklis [3] closely. Throughout, I will assume that assumptions 1 and 2 hold. Note that assumption 2 guarantees that the optimal value function will be non-negative.

Proposition 1. For all $p \geq 1$, the following hold for the mapping T_p:

(a) (Monotonicity) T_p is monotone in the sense that $\forall\, V, V' \in (\Re^+)^{|S|}$:

$$V \leq V' \;\; \Rightarrow \;\; T_p(V) \leq T_p(V'),$$

Proof: Follows trivially from the monotonicity of the generalized mean (Fact 4).

(b) (Contraction Mapping) For all finite $V, V' \in (\Re^+)^{|S|}$,

$$||T_p(V) - T_p(V')||_\infty \;\; \leq \;\; \alpha ||V - V'||_\infty,$$

where $\alpha < 1$.

Proof: Clearly, $\exists\, 0 \leq M < \infty$, s.t. $||V - V'||_\infty \leq M$. Then $\forall\, x \in S, -M \leq (V(x) - V'(x)) \leq M$. Substituting $V'(y) + M$ for $V(y)$ in Equation 3, we get

$$T_p(V)(x) \leq \left\{ \frac{1}{|A|} \sum_{a \in A} [r(x,a) + \gamma \sum_{y \in S} P_{xy}(a)(V'(y) + M)]^p \right\}^{\frac{1}{p}},$$

and using the fact that P is a stochastic matrix,

$$T_p(V)(x) \leq \left\{ \frac{1}{|A|} \sum_{a \in A} [r(x,a) + \gamma M + \gamma \sum_{y \in S} P_{xy}(a)V'(y)]^p \right\}^{\frac{1}{p}}. \tag{4}$$

By symmetry, it is also true that:

$$T_p(V')(x) \leq \left\{ \frac{1}{|A|} \sum_{a \in A} [r(x,a) + \gamma M + \gamma \sum_{y \in S} P_{xy}(a)V(y)]^p \right\}^{\frac{1}{p}}. \tag{5}$$

Using the boundedness of the generalized mean (Fact 5), Equations 4 and 5 imply that $\forall\, x \in S$, $T_p(V)(x) \leq T_p(V')(x) + \gamma M$, and that $T_p(V')(x) \leq T_p(V)(x) + \gamma M$. That proves Proposition 1(b).

Theorem 1: Under Assumptions 1 and 2, if the starting estimate $V_0 \in (\Re^+)^{|S|}$, then $\forall\, p \geq 1$, the iteration $V_{t+1} = T_p(V_t)$ converges to a unique fixed point V_p^*.

Proof: Using Proposition 1(b) and the contraction mapping theorem, the iterative algorithm defined by Equation 3 converges to a unique fixed point.

Corollary 1(a): Let V^* be the optimal value function. Then $\lim_{p \to \infty} V_p^* = V^*$.

Proof: Bertsekas and Tsitsiklis [3] show that the iteration $V_{t+1} = T(V_t)$, where the operator T is as defined in Equation 2, converges to the optimal value function V^*. Therefore, $\lim_{p \to \infty} T_p = T \Rightarrow \lim_{p \to \infty} V_p^* = V^*$.

Theorem 2: $1 \leq p \leq q \Rightarrow V_p^* \leq V_q^* \leq V^*$.

Proof: Consider the iteration by iteration estimates for a parallel implementation of the two algorithms: $V_{p,t+1} = T_p(V_{p,t})$ and $V_{q,t+1} = T_q(V_{q,t})$, where the successive estimates have been subscripted with the additional symbols p and q in order to distinguish between the two algorithms. Assume that $V_{p,0} = V_{q,0} \leq V_p^* \leq V^*$, and $V_{p,0} = V_{q,0} \leq V_q^* \leq V^*$.

$$
\begin{array}{lcll}
V_{p,0} & = & V_{q,0} & \text{by construction,} \\
V_{p,1} & \leq & V_{q,1} & \text{by Uniform Improvement (Fact 3); applied to each state,} \\
V_{p,2} & \leq & V_{q,2} & \text{by Monotonicity (Fact 4); applied to each state,} \\
. & . & . & ., \\
. & . & . & ., \\
V_{p,t} & \leq & V_{q,t} & \text{by Monotonicity (Fact 4); applied to each state.}
\end{array}
$$

We know that $V_{p\infty} = V_p^*$, and $V_{q\infty} = V_q^*$. As shown above, $\exists\, V_0 \in (\Re^+)^{|S|}$, s.t. $\forall\, t > 0, V_{p,t} \leq V_{q,t} < V^*$. By Theorem 1, we know that the fixed point is independent of V_0. Therefore, $V_p^* \leq V_q^* \leq V^*$ $(\forall\, V_0 \in (\Re^+)^{|S|})$.

Corollary 2(a)**:** For any $\epsilon > 0$, $\exists\, p \geq 1$, such that $\forall\, q > p$, $||V_q^* - V^*||_\infty < \epsilon$.

Proof: From Theorems 1 and 2 the sequence of vectors $\{V_p^*\}$ are bounded and converge to V^*. Corollary 2(a) is a property of bounded and convergent sequences.

Thus, the operator $\{T_p\}$ defines a family of iterative approximation algorithms to the value iteration algorithm. As the parameter p is increased, the approximation to the optimal value function becomes uniformly better. However, the true measure of interest is not how closely the optimal value function is approximated, but how good is the greedy policy derived from the approximations.

3.2 How good are the approximations in policy space ?

Fact 9. For any given finite action MDT, $\exists\, \delta > 0$, such that $\forall\, \tilde{V} \in (\Re^+)^{|S|}$, s.t. $||\tilde{V} - V^*||_\infty < \frac{\delta}{2}$, any policy that is greedy with respect to $\tilde{V}$ is optimal. See Appendix for proof.

Fact 9 implies that as long as the estimated value function is within $\frac{\epsilon}{2}$ of the optimal value function, the policy derived from the approximation will be optimal. Define Π_p to be the set of stationary policies that are greedy with respect to V_p^*. If $\pi \in \Pi_p$, then $\forall\, x \in S$, and $\forall a \in A$, the immediate payoff for executing action $\pi(x)$ summed with the expected discounted value of the next state is $\geq$ that for any other action $a \in A$, i.e.,:

$$R(x, \pi(x)) + \gamma \sum_{y \in S} P_{xy}(\pi(x)) V_p^*(y) \quad \geq \quad R(x, a) + \gamma \sum_{y \in S} P_{xy}(a) V_p^*(y).$$

Let Π^* be the set of stationary optimal policies. Define $I_p = \Pi_p \cap \Pi^*$.

Theorem 3**:** For any finite MDT, $\exists\, p$, where $1 \leq p < \infty$, s.t. $\forall\, q > p, \Pi_q \subset \Pi^*$.

Proof: Follows directly from Corollary 2(a) and Fact 9.

Theorem 3 implies that in practice there is no need for $p \uparrow \infty$ for the algorithm defined by T_p to yield optimal policies.

4 Discussion

DP-based learning algorithms defined using the max operator update the value of a state based on the estimate derived from the "best" action from that state. Algorithms based on the generalized mean, on the other hand, update the value of a state using some non-linearly weighted average of the estimates derived from all the actions available in that state. Thus, the latter will assign higher values to states that have many good actions over states that have just one good action, and conversely will also penalize states for having any really bad action at all. This may be advantageous for many tasks where there is a great deal of uncertainity. Learning algorithms that increase the index p as more information accrues can smoothly interpolate between considering the estimates from all the actions to considering the estimate from the best action alone.

Another advantage of using the operator T_p instead of T is that unlike T, T_p is differentiable, which makes it possible to compute the following derivative for neighboring states x and y:

$$\frac{\partial V(x)}{\partial V(y)} = \frac{\gamma}{|A|} \sum_{a \in A} \left\{ \left[\frac{r(x,a) + \sum_{y' \in S} P_{xy'}(a) V(y')}{V(x)} \right]^{p-1} \sum_{z \in S} P_{xz}(a) \frac{\partial V(z)}{\partial V(y)} \right\}. \quad (6)$$

Note that one can use the *chain rule* to compute the above derivatives for states that are not neighbors in the state graph of the MDT in much the same way as the backpropagation [9] algorithm for multi-layer connectionist networks. Being able to compute derivatives allows sensitivity analysis and may lead to some new ways of addressing the difficult exploration versus control issue [10] in optimal control tasks. Indeed, the motivation for Rivest's work [8], which inspired this paper, was to use sensitivity analysis to address the analogous exploration issue in game tree search. Note that derivatives of the values of states with respect to the transition probabilities and the immediate payoffs can also be derived.

While the algorithms developed in this paper are restricted to discrete MDTs, similar algorithms can be derived for the continuous case by using the integral form of the generalized mean:

$$T_p(V)(x) = \left[\int_{a \in A} \rho(a) [r(x,a) + \gamma \int_{y \in S} P_{xy}(a) V(y) dy]^p da \right]^{\frac{1}{p}},$$

where $\int_{a \in A} \rho(a) da = 1$ is a weighting function analogous to the $\frac{1}{|A|}$ in the discrete (see Equation 3) version. It is conjectured that convergence results similar to the ones derived for the discrete case can be derived for continuous state MDTs.

As discussed by Rivest [8], other forms of generalized means exist, e.g., for any continuous monotone increasing function, f, we can consider mean values of the form $f^{-1}(\frac{1}{n}\sum_{i=1}^{n} f(a_i))$. In particular, the exponential function can be used to derive an interesting alternative sequence of operators, $H_\lambda = \frac{\ln(\frac{1}{n}\sum_{i=1}^{n} e^{\lambda a_i})}{\lambda}$, where $\lambda > 0$. As the parameter λ is increased in value the approximation to the max gets strictly better. Using H_λ, a family of alternative iterative fixed point algorithms can be defined: $V_{t+1} = H_\lambda(V_t)$. An advantage of using H_λ is that it requires less computation than the operator T_p. The operator H_λ is similar in spirit to the "soft-max" function used by Jacobs *et al.* [6] and may provide a probabilistic framework for action selection in DP-based algorithms.

5 Conclusion

Several researchers are investigating the advantages of combining nonlinear neural network based function approximation methods and traditional adaptive control techniques (e.g., [7, 4]). The algorithms presented in this paper have the dual advantages of leading to more robust solutions and of employing a differentiable backup operator. It is hoped that these changes will pave the way for further progress in adapting DP algorithms and nonlinear neural network techniques for *learning* to solve optimal control tasks.

Appendix

Proof **for Fact** 5**:**

It is known that if $p \geq 1$, $[\sum_{i=1}^{n}(a_i + \Delta_i)^p]^{\frac{1}{p}} \leq [\sum_{i=1}^{n}(a_i)^p]^{\frac{1}{p}} + [\sum_{i=1}^{n}(\Delta_i)^p]^{\frac{1}{p}}$. Thus,

$$\begin{aligned}
[\frac{1}{n}\sum_{i=1}^{n}(a_i)^p]^{\frac{1}{p}} &= \frac{1}{n^{\frac{1}{p}}}[\sum_{i=1}^{n}(a_i)^p]^{\frac{1}{p}} \\
&\leq \frac{1}{n^{\frac{1}{p}}}[\sum_{i=1}^{n}(a'_i + M)^p]^{\frac{1}{p}} \\
&\leq \frac{1}{n^{\frac{1}{p}}}[\sum_{i=1}^{n}(a'_i)^p]^{\frac{1}{p}} + \frac{1}{n^{\frac{1}{p}}}[\sum_{i=1}^{n}(M)^p]^{\frac{1}{p}} \\
&= A'_p + M
\end{aligned}$$

Thus $A_p \leq A'_p + M$, and by symmetry, $A'_p \leq A_p + M$. Therefore, $|A_p - A'_p| \leq M$.

Proof **for Fact** 9**:**

$\forall x \in S$, let $A - \{\pi^*(x)\}$ be the set of non-optimal actions available in state x.
If $A - \{\pi^*(x)\} \neq \emptyset$, then

$$m(x) \;=\; \min_{a \in A - \{\pi^*(x)\}} |(R(x,\pi^*(x)) + \gamma \sum_{y \in S} P_{xy}(\pi^*(x)) V^*(y)) - (R(x,a) + \gamma \sum_{y \in S} P_{xy}(a) V^*(y))|,$$

else, $m(x) = \infty$. Let $\delta = \min_{x \in S}\{m(x)\}$. Note that $\delta = 0$ only if all actions are optimal in every state; in such a case, δ can be set to any non-zero value. Clearly, by perturbing the optimal value function by less than $\frac{\delta}{2}$, the greedy action in any state will still be optimal.

Differentiable Approximations to Q-learning

Q-learning [11] is a DP-based algorithm that estimates Q-values, $Q(x,a)$ that are defined via the following equations:

$$Q(x,a) \;=\; r(x,a) + \gamma \sum_{y \in S} P_{xy}(a)[\max_{a' \in A} Q(y,a')]$$

Thus, Q-values associate scalars to state-action pairs instead of just the states. Given the Q-values the optimal value function is easily defined as $V^*(x) = \max_{a \in A} Q(x,a)$. Q-learning is an asynchronous iterative improvement algorithm:

$$Q_{t+1}(x,a) \;=\; (1 - \alpha_t(x,a)) Q_t(x,a) + \alpha_t(x,a)[r(x,a) + \gamma \max_{a' \in A} Q(y,a')],$$

where y is a sampled next state chosen with probability $P_{xy}(a)$. Note that Q-learning does require knowledge of the state transition probabilities P to learn the Q-values. Generalized means of order p can be used to define Q-values without using the non-differentiable max operator:

$$Q(x,a) \;=\; r(x,a) + \gamma \sum_{y \in S} P_{xy}(a) \left[\frac{1}{|A|} \sum_{a' \in A} Q(y,a')^p \right]^{\frac{1}{p}}.$$

The corresponding learning algorithm is:

$$Q_{t+1}(x,a) \;=\; (1 - \alpha_t(x,a)) Q_t(x,a) + \alpha_t(x,a)\{r(x,a) + \gamma \left[\frac{1}{|A|} \sum_{a' \in A} Q(y,a')^p \right]^{\frac{1}{p}}\}$$

References

[1] A.G. Barto, R.S. Sutton, and C. Watkins. Sequential decision problems and neural networks. In D.S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 686–693, San Mateo, CA, 1990. Morgan Kaufmann.

[2] R.E. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.

[3] D.P. Bertsekas and J.N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, Englewood Cliffs, NJ, 1989.

[4] W. Fun and M.I. Jordan. The moving basin: Effective action-search in adaptive control, 1992. submitted to Neural Computation.

[5] G.H. Hardy, J.E. Littlewood, and G. Polya. *Inequalities*. University Press, Cambridge, England, 2 edition, 1952.

[6] R.A. Jacobs, M.I. Jordan, S.J. Nowlan, and G.E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1), 1991.

[7] M.I. Jordan and R.A. Jacobs. Learning to control an unstable system with forward modeling. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, San Mateo, CA, 1990. Morgan Kaufmann.

[8] R.L. Rivest. Game tree searching by min/max approximation. *Artificial Intelligence*, 34:77–96, 1988.

[9] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. In D.E. Rumelhart and J.L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol.1: Foundations*. Bradford Books/MIT Press, Cambridge, MA, 1986.

[10] R.S. Sutton. Integrating architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proc. of the Seventh International Conference on Machine Learning*, pages 216–224, San Mateo, CA, 1990. Morgan Kaufmann.

[11] C.J.C.H. Watkins. *Learning from Delayed Rewards*. PhD thesis, Cambridge Univ., Cambridge, England, 1989.