Fast Reinforcement Learning using Multiple Models

Kumpati S. Narendra† *IEEE Life Fellow*, Yu Wang†, and Snehasis Mukhopadyhay*,
Center for Systems Science, Yale University

Abstract— **Reinforcement Learning aims to find the optimal decision in uncertain environments on the basis of qualitative and noisy on-line performance feedback provided by the environments. During the past four decades, learning theory has grown into a vast field in which a very large number of problems have been studied. One of the primary limitations of reinforcement schemes, acknowledged by workers in the field, is their slow speed of convergence. The principal objective of this paper is to present a new approach, based on the use of multiple models (or estimates), that may alleviate this problem and increase the speed of response.**

In adaptive control theory, multiple model based methods have been proposed over the past two decades, which improve substantially the performance of the system. The authors undertook to apply similar concepts in reinforcement learning as well, and this paper represents the first effort in this direction. Simple situations of learning in feed-forward networks are considered in the paper, and compared to two different schemes. It is shown that convergence speeds that are more than an order of magnitude faster than those of the first scheme, can be achieved in some cases. While the second scheme is comparable to the new approach in many situations, it is seen to exhibit undesirable behaviour in others, where the new approach is more robust. The latter is currently being extended incrementally and systematically to more complex problems that have been discussed in the literature. The ultimate aim of the authors is to apply this approach to learning in discrete and continuous state dynamic environments.

I. Introduction

Learning is defined as any relatively permanent change in behavior resulting form past experience, and a learning system is characterized by its ability to improve its behavior with time in some sense. In mathematical psychology models of learning systems were developed over fifty years ago to explain behavior patterns among living organisms. These, in turn, were used to synthesize engineering systems, which could be considered to exhibit "learning behavior".

In the introduction to their book "Reinforcement Learning: An Introduction" [1], Barto and Sutton state that the history of reinforcement learning consists of two main threads, both of which have evolved over a long period and both of which enjoy a rich literature. The first is learning by trial and error, which as stated earlier, started with the psychology of animal learning. The other thread is optimal control, which is an integral part of control theory and is based on the pioneering work of Pontryagin and his coworkers [2] as well as the classical theories of Hamilton and Jacobi, and the Principle of Optimality of Bellman [3]. Reinforcement learning aims to find the optimal decision (or decision rule) in feedback with an unknown and uncertain teacher or environment, on the basis of qualitative and noisy feedback received from the environment. At the present time there exists a very large variety of learning models and algorithms ([4]-[13]) depending upon the assumptions made concerning the environments. The unifying theme of all these different models is that the resulting learning behaviors can be considered as conducting a probabilistic search over the action/decision probability space to optimize an underlying implicit criterion function.

One of the primary well known limitations of all reinforcement methods proposed in the literature over the years is their slow speed of convergence([14]). Different authors have commented on this fact, and, in spite of the many advances that have taken place in learning theory in recent years, it continues to be true to this day. This slowness can be partially attributed to the underlying assumption that the relevant information concerning the environment has to be learned from complete ignorance, using only the responses of the environment to different actions.

In this paper, we introduce a new approach based on multiple models for improving the speed of response of learning schemes. In the past, the authors have successfully developed numerous approaches based on multiple models for improving the response of dynamical systems in a conceptually related field, i.e. adaptive control ([15]-[21]) Motivated strongly by the success of such approaches, a similar effort is being made in this paper in the area of learning systems. The objective is to investigate whether the speed of convergence of learning schemes can be improved substantially, and the extent to which such improvements depend upon the complexity of the system. Further, it was also decided to test the approach successively on incrementally more complex problems. While the ultimate objective is to apply the method to learning in dynamical environments, only two preliminary stages are reported in this paper. These include the learning automaton discussed in Section II, and feed-forward networks considered in Section IV.

II. The Learning Automaton

The learning automaton is one of the earliest reinforcement learning models in the literature ([22],[23]). An agent can perform one action out of a finite set of r actions at every instant into an environment. The environment responds to each action with either a "reward" or a "penalty". The probability of action α_i yielding a reward is d_i and d_{opt}=Max$_j$ d_j corresponds to the optimal action. The probabilities d_i

†: Professor Kumpati S. Narendra and Yu Wang are with the Department of Electrical Engineering, Yale University, New Haven, 06510; *:Professor Snehasis Mukhopadyhay is with the Department of Computer and Information Science, Indiana University - Purdue University Indianapolis

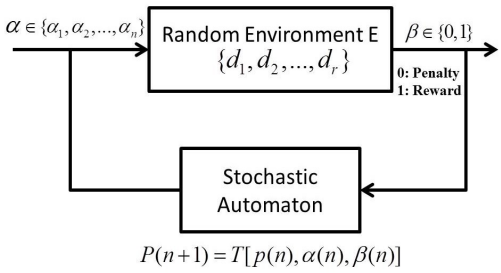


Fig. 1: Learning Automata

are unknown. The objective is to derive a procedure by which an agent learns the "best" action (i.e. α_{opt}, the action corresponds to d_{opt}) by performing actions in the environment.

The learning automaton in its simplest form is shown in Figure (1). The environment E is defined by a set of r reward probabilities $\{d_1, d_2, ..., d_r\}$, corresponding to actions in an input set $\boldsymbol{\alpha}$= $\{\alpha_1, \alpha_2, ..., \alpha_r\}$ and an output set $\boldsymbol{\beta}$=$\{1, 0\}$, where 1 is referred to as a reward and 0 as a penalty.

$$d_i = Prob[\beta = 1 | \alpha = \alpha_i] \qquad \alpha_i \in \boldsymbol{\alpha} \tag{1}$$

A. Automaton:

The automaton is a stochastic decision rule which at every instant n selects an action α_i from the action set with a probability distribution $p_i(n)$, and receives a response β(n). From the action chosen and the response obtained it modifies the action probabilities using the learning rule

$$p(n+1) = T[p(n), \alpha(n), \beta(n)] \tag{2}$$

where T is a mapping.

The choice of T determines the accuracy as well as the speed of the learning rule. A very large number of mappings (or learning rules) have been proposed by different authors([24]-[27])

Comment 1: The principal feature of the learning automaton is that it is static, has only one state in which the best action is to be determined, and that a single first order difference equation describes the learning scheme. In more complex systems as shown later, such choices may have to be made at different states and in dynamic environments.

B. Norms of Behavior:

If two learning schemes operate in an environment, the one that results in favorable responses more often is considered better. To quantify this we define $M(n)$, which is the average reward for a given action probability vector p(n).i.e. $M(n) = E[\beta(n)|p(n)]$

Initially, when learning starts, it is natural to assume that all actions have the same probabilities i.e. $\frac{1}{r}$, and the average reward is $M_0 = \frac{1}{r}\Sigma d_i$.

A learning scheme is said to be expedient if the average reward $M(n) > M_0$, and absolutely expedient if $E[M(n+1)|p(n)] > M(n)$ for all n and all d_i, or $M(n)$ is a submartingale.

A learning scheme is said to be ϵ-optimal if $lim_{n\to\infty} = d_{opt} - \epsilon$ for any arbitrary $\epsilon > 0$ by the choice the parameters of the reinforcement scheme.

Among the simple learning schemes suggested in the literature, the linear reward-inaction scheme suggested by Shapiro and Narendra in 1969 [28] is found to be ϵ-optimal. In this scheme the probability of an action α_i which yields a reward is increased while the probability of an action which yields a penalty is left unchanged. Further, with a reward, all the probabilities of the other actions are suitably modified to conserve probability measure.

Comment 2: Qualitatively, one is interested in choosing only the best action in the limit. However, the fact that one action is better than another can be concluded only by trying both of them. This accounts for many schemes being only ϵ-optimal rather than strictly optimal.

C. The L_{R-I} Scheme

1) Two action case $\{\ \alpha_1,\ \alpha_2\ \}$*:* When $\alpha(n) = \alpha_i\ i = \{1, 2\}$ and result in a reward response ($\beta(n) = 1$), the action probability is updated as

$$p_i(n+1) = p_i(n) + g[p_j(n)] \tag{3}$$

$$p_j(n+1) = p_j(n) - g[p_j(n)] \quad j \neq i \tag{4}$$

If the response by choosing $\alpha(n) = \alpha_i$ is penalty i.e. $\beta(n) = 0$, the action probability is updated as

$$p_i(n+1) = p_i(n) \qquad i \in \{1, 2\} \tag{5}$$

2) r-actions case $\{\ \alpha_1,\ ...,\ \alpha_r\ \}$*:* When $\alpha(n) = \alpha_i$ and results in a reward response ($\beta(n) = 1$), the action probability is updated as

$$p_j(n+1) = p_j(n) - g[p_j(n)] \qquad \beta(n) = 1 \tag{6}$$

$$p_i(n+1) = p_i(n) + \Sigma_{j\neq i} g[p_j(n)] \tag{7}$$

$g[p_j(n)]$ in the L_{R-I} scheme is $ap_j(n)$ so that $\Sigma_{j\neq i} g[p_j(n)] = a[1 - p_i(n)]$

As in the two action case, the probabilities $p_i(n)$ remain the same for a penalty response.

Comment 3: A simple procedure for increasing the speed of response is to increase the step-size in the learning algorithms. However, this also increases the probability of convergence to a wrong action.

Comment 4: Throughout the paper, we shall use learning schemes which have the same form as the L_{R-I} scheme for comparison purposes. The principal difference between the schemes is the manner in which the probabilities are updated based on all the available information at that instant.

III. The Pursuit Algorithm

The Linear Reward-Inaction algorithm described thus far is a direct scheme which uses only the environmental feedback at every iteration. In contrast to this, indirect estimator

algorithms have been proposed in the literature, which use the entire history of the environmental feedback, to order the preference of the actions at stage 'n' [29]. A special case of such an algorithm is the pursuit algorithm proposed by Thathachar and Sastry in 1985 [30]. In this scheme, if action α_i is tried N_i times and results in n_i reward outputs, the ratio $\frac{n_i}{N_i}$(i=1,2,...,r) is maintained by the automaton and at every state, the probability of the action α_{opt} corresponding to $Max_i[\frac{n_i}{N_i}]$ is updated as defined earlier. The convergence speed of this algorithm is found, in practice, to be significantly greater than that of the direct algorithm, and hence comparison of the new approach proposed must also be made with this scheme.

In the following we present several studies in which the multiple-model based approach is compared both with the Linear Reward-Inaction scheme and the pursuit algorithm.

IV. The Multiple Models Approach

In the multiple models approach, several estimates (models) are chosen to determine which of them explains most accurately the observed outputs of the environment. Both fixed and adaptive models can be used for this purpose but only the former are used in this paper.

If d_1, d_2,...,d_r are the reward probabilities, our main objective is in ordering them and determining d_{opt}. Towards this end a finite number of estimates m_j ($j = 9$ in all the simulations presented) are chosen as $\{0.1,0.2,...,0.8,0.9\}$.

If a reward probability (say 0.62) which is unknown produces a sequence of N_1 outputs with n_1 rewards and $N_1 - n_1$ penalties the likelihood function for the j^{th} model is

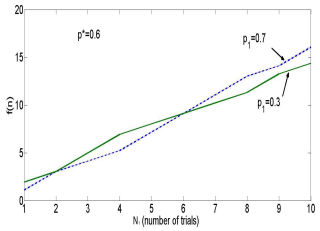
$$f[m_j, N_1] = m_j^{n_1}(1 - m_j)^{N_1 - n_1} \tag{8}$$

At every instant, this is computed for each value of $j = \{1, 2, 3, ...9\}$. Using this information, the likelihood functions can be plotted as functions of n. At any instant N_1, the one that is the maximum can be considered to be closest to the unknown probability (i.e 0.62). In learning schemes, it is necessary to determine at every instant which action has the highest reward probability so that its (action) probability can be increased as shown in equations (6), (7) for the L_{R-I} scheme.

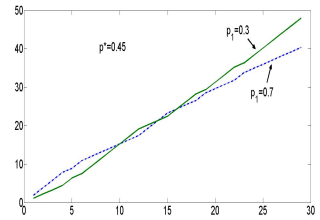
Comment 5: From the foregoing discussion, it is clear that several estimates (models) will be needed for each action, to order them according to their reward probabilities. Further, if the system has many states where many actions can be taken (as discussed in later sections), multiple models would be needed for every one of them also. This implies that improvement in the speed of response is obtained at the cost of additional computation.

Comment 6: In all the simulations presented, as mentioned earlier, nine models (estimates) distributed uniformly between 0 and 1 were used. For greater precision (i.e. to distinguish between probabilities which are closer to each other) more models may be needed.

Simulation 1: The following simulation illustrates how the multiple model approach is used in all the learning schemes. An action α_1 has an unknown reward probability 0.62 and is attempted 10 times and yields the sequence "THTTHHHHTH", where H denotes a reward and T a penalty. Using the estimates $m_1 = 0.7$, $m_2 = 0.3$, two likelihood functions are plotted as shown in Figure (2) as a function of N_1 (Note that -log(f) is plotted).



(a) Evolution of the likelihood function with p_1=0.7, p_2=0.3, $p*$=0.62



(b) Evolution of the likelihood function with p_1=0.7, p_2=0.3, $p*$=0.45

Fig. 2: Evolution of likelihood functions

At every instant, the likelihood estimate that is a maximum is concluded to be closest to 0.62. Similarly, if a second action has a reward probability of 0.45, the same procedure is adopted to choose the estimate whose distance from 0.45 is a minimum.

The two maxima obtained are used to rank order the two reward probabilities, i.e. to conclude that the first action is better than the second.

While only two models (estimates) were used in the discussion for purposes of clarity, nine models were used in all the simulations including Simulation 2 discussed below.

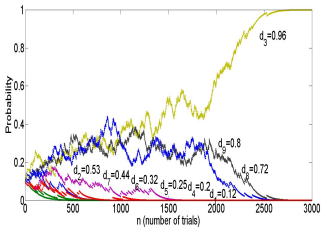
Simulation 2: The simulation results obtained in a simple automaton with nine actions with reward probabilities $\{d_1 = 0.12, d_2 = 0.53, d_3 = 0.96, d_4 = 0.2, d_5 = 0.25, d_6 = 0.32, d_7 = 0.44, d_8 = 0.72, d_9 = 0.8\}$ are shown in Figures (3). The nine models are chosen uniformly distributed in [0,1] as m_i=$\{0.1, ..., 0.9\}$. They show the convergence of the simple L_{R-I} scheme and the scheme based on multiple models. While the L_{R-I} scheme takes approximately 2500 trials, the multiple model approach reaches a probability of 0.95 for the optimal action, in 250 trials (Figure (3b)).

Comments on the Multiple Model Approach:

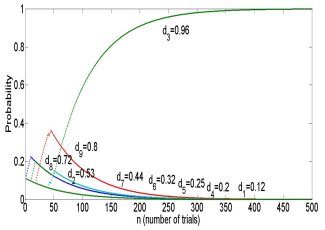
Several theoretical questions need to be addressed while proposing a new approach to learning:

(i) The first concerns the proof that the method is ϵ-optimal.

(ii) The second is a detailed comparison of the multiple-model based approach with the pursuit algorithm. This is particularly important since at every stage the latter



(a) L-RI Scheme with Multiple Actions



(b) Multiple Models with Multiple Actions

Fig. 3: Nine actions with Single Model and Multiple Models

maximizes the value of the likelihood function. It has been shown in [27], that sample paths of the latter may not stay inside a fixed neighbourhood of its target, and this is reflected in simulation studies 3, 4, and 5. The multiple model approach is seen to result in fast convergence without exhibiting such behavior.

(iii) The better performance of the new approach has to be explained on the basis of the prior information assumed of the environment, the number of models chosen, and their effect on the convergence properties of the algorithm. At the present time a detailed study is under-way on all the above issues.

Comment 7: Since the new approach is significantly faster than the Linear Reward-Inaction scheme, all the comparisons made in Simulation 3,4 and 5 are with the pursuit algorithm.

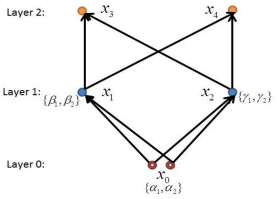
V. Learning in Feed-forward Networks

As stated in the introduction, the number of situations in which learning is encountered is extremely large. The authors decided to study the effectiveness of the multiple model based approach, incrementally, by successively increasing the complexity of the problem. As stated in the introduction, while the ultimate aim is to apply it to learning in discrete state as well as continuous state dynamic environments, in this preliminary paper we confine our attention to learning in feed-forward networks, with multiple layers, where actions at each layer transfers the system to the following layer, and the reward received by an action depends upon a finite number of states that follow.

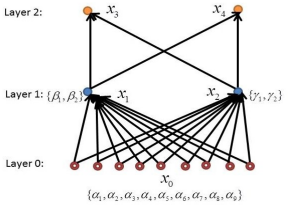
In the following, we describe three simulation studies which deal with the three networks shown in Figure (4).

A. Structure 1 (Figure 4(a)): Two actions at each state at layer 2

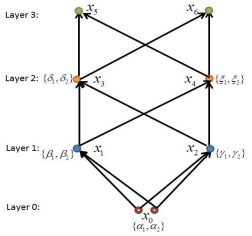
The structure of the network simulated here is shown in Figure (4a), and has three layers (i.e. 0, 1 and 2). Only two actions α_1 and α_2 can be performed at layer 0 and transfer



(a) Structure 1: Two actions at each state at layer 1



(b) Structure 2: Nine actions at layer 0



(c) Structure 3: Three layers

Fig. 4: Three Feed-forward Networks

the system to either state x_1 or x_2. The transition probability matrix is P_1

$$P_1 = \begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{bmatrix} = \begin{bmatrix} 0.5 & 0.5 \\ 0.2 & 0.8 \end{bmatrix} \tag{9}$$

where p_{ij} is the probability that actions α_i transfers the state x_0 to x_j(j=1,2).

Actions $\{\beta_1, \beta_2\}$ at state x_1 and $\{\gamma_1, \gamma_2\}$ at state x_2 transfer the system probabilistically to state x_3 and x_4 respectively. M and N are transition probabilities defined by

$$M = \begin{bmatrix} M_{13} & M_{14} \\ M_{23} & M_{24} \end{bmatrix} = \begin{bmatrix} 0.1 & 0.9 \\ 0.4 & 0.6 \end{bmatrix} \tag{10}$$

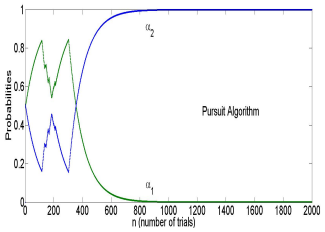
$$N = \begin{bmatrix} N_{13} & N_{14} \\ N_{23} & N_{24} \end{bmatrix} = \begin{bmatrix} 0.8 & 0.2 \\ 0.5 & 0.5 \end{bmatrix} \tag{11}$$

where M_{ij}= {transition probability from x_1 using actions $\beta_i (i = 1, 2)$ to states $x_j (j = 3, 4)$ } and N_{ij}={transition probability from x_2 using actions $\gamma_i (i = 1, 2)$ to states $x_j (j = 3, 4)$ }.

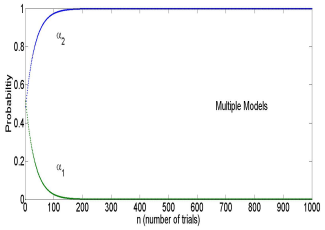
On reaching the states x_3 and x_4 the system receives the reward 10 and 20 respectively. The objective of the learning scheme is to determine the optimal actions α_i, β_j and γ_k $(i, j, k$={$1, 2$}) at states x_0, x_1 x_2 respectively.

Algebraic Part (Optimization):

Before proceeding to simulate the given problem we compute the optimal actions assuming that the matrices P, M and N are known. (In adaptive control, this has been referred to as

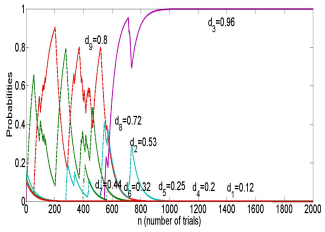


(a) Single Models Response of α_1 and α_2 for Structure 1

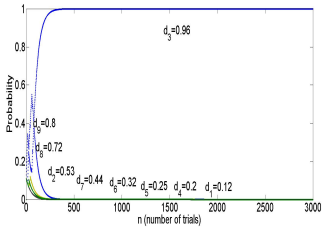


(b) Multiple Models Response of α_1 and α_2 for Structure 1

Fig. 5: Response of actions α_1 and α_2 for Structure 1



(a) Response with Single Model for Structure 2



(b) Response with Multiple Models for Structure 2

Fig. 6: Response of actions $\alpha_1, \alpha_2, \cdots, \alpha_9$ for Structure 2

the algebraic part of the problem. In learning schemes, it becomes a problem of optimization).

Computation of Optimal Actions:

If P, M and N are known, the expected rewards with actions β_1, β_2, γ_1, γ_2 can be computed as 19, 16, 12 and 15 respectively. This implies that β_1 is the best action at state x_1 and γ_2 is the best action at state x_2.

Knowing the expected rewards of the optimal actions β_1 and γ_2, the optimal action at x_0 can be determined as α_2 (The rewards for α_1 and α_2 are 16.2 and 18.2 respectively).

Simulation 3:

In the simulation studies we are interested in, the action probabilities of α_2, β_1 and γ_2 to converge arbitrarily close to 1. Due to space limitations, only the evolution of the probabilities for α_1, α_2 and the time for convergence using both the pursuit algorithm and the multiple model approaches are shown in Figures (5).

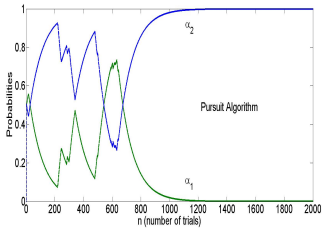
B. More Complex Systems

From Structure 1, the improvement in performance using multiple models is seen to depend on the number of probabilities of reward that has to be estimated (or ordered). Simulations 4 and 5 demonstrate that this is indeed the case, when the number of actions in layer 0 is increased (Simulation 4) and when the network has 3 layers (Simulation 5).

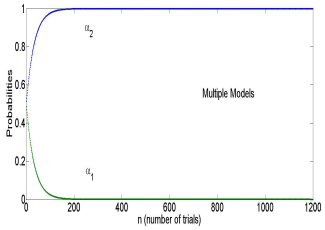
Simulation 4 (Structure 2, Figure 4 (b)): The structure of the system in simulation 4 is similar to that in Simulation 2, with the exception that the action set at layer 0 consists of 9 actions$\{\alpha_1, ..., \alpha_9\}$. The transition probability from x_0 to x_1 using action α_i is d_i (and hence to x_2 is 1-d_i). The value of d_i were chosen as follows:

$\{d_1 = 0.12,\ d_2 = 0.53,\ d_3 = 0.96,\ d_4 = 0.2,\ d_5 = 0.25,\ d_6 = 0.32,\ d_7 = 0.44,\ d_8 = 0.72,\ d_9 = 0.8\ \}$,

Since the transition probabilities from x_1 and x_2 to x_3 and x_4 remain the same, the expected rewards with actions β_1,β_2,γ_1,γ_2 are also the same as in simulation 2 (i.e. 19,16 and 12,15).



(a) Response of Single Model for Structure 3



(b) Response of Multiple Models for Structure 3

Fig. 7: Response of actions α_1, α_2 for Structure 3

From the probabilities d_i, the optimal action can be computed to be α_3 (with probability $d_3 = 0.96$).

The rapid convergence of the multiple model based approach relative to the pursuit algorithm is shown in Figure 6(a) and (b). It is worth noting that the convergence of even a sample path is relatively smooth in the former case. In the sample paths shown in Figure 6(a) for the pursuit algorithm a non-optimal action is chosen with the highest probability for the first 400 trials.

Simulation 5 (Structure 3, Figure 4 (c)): The structure used in simulation 5 (Figure (4)(c)) has three layers with states x_1 and x_2 at layer 1, x_3 and x_4 at layer 2 and x_5 and x_6 at layer 3. The actions in sets $\{\alpha_1, \alpha_2\}$ at x_0, $\{\beta_1, \beta_2\}$, $\{\gamma_1, \gamma_2\}$, $\{\delta_1, \delta_2\}$, $\{\xi_1, \xi_2\}$ in states x_1, x_2, x_3 and x_4 respectively, have to be determined to optimize the expected reward.

The learning algorithms in layers 1 and 2 have to converge before the optimal action in the set α can be determined. In the simulation depicted in Figure 7, the multiple-model based approach is found to be significantly faster than the single model based approach.

Note: It is to be noted that 100 models corresponding to the ten actions had to be used to realize the approximately twenty times faster response realized in the simulation.

Comment 8: While in many simulation studies, the multiple model based approach and that using the pursuit algorithm behave in a similar fashion, situations also exist when a non-optimal action is chosen for a long interval using the latter before the probability of the optimal action tends to a value close to unity. The simulations shown in Figures 5, 6 and 7 are illustrative of such responses. The theoretical work that is currently under-way aims to explain this behavior.

VI. Conclusion

A new approach based on the use of multiple estimates for improving the convergence of learning schemes is proposed in this paper. Simulation studies using simple learning automata, and feed-forward learning schemes, are included. The principal objective is to compare the performance of such learning schemes using either a Linear Reward-Inaction scheme or the pursuit algorithm to that obtained using multiple models. The results clearly indicate that the multiple model based schemes are clearly an order of magnitude faster than the former but comparable in speed to the pursuit algorithm over some sample paths. The latter, however, exhibits convergence behavior which makes it unattractive in learning situations. In contrast to this the multiple model based approach is both fast and robust. Theoretical investigations are currently in progress to explain the differences between the two approaches. The speed and robustness of the convergence of multiple models provide ample evidence that the new approach may also prove significantly better in more complex learning contexts, where learning has to be carried out in general dynamic environments.

Acknowledgement:
The research reported here was supported by NSF Grant No. 1408279.

References

[1] Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. MIT press, 1998.

[2] Pontryagin, Lev S. Mathematical theory of optimal processes. CRC Press, 1987.

[3] Bellman, Richard. "Dynamic programming and Lagrange multipliers." Proceedings of the National Academy of Sciences 42.10 (1956): 767-769.

[4] Barto, Andrew G., Steven J. Bradtke, and Satinder P. Singh. "Learning to act using real-time dynamic programming." Artificial Intelligence 72.1 (1995): 81-138.

[5] Doya, K., Kazuyuki Samejima, Kenichi Katagiri, Mitsuo Kawato. "Multiple model-based reinforcement learning." Neural computation 14.6 (2002): 1347-1369.

[6] Kaelbling, Leslie P., Michael L. Littman, and Andrew W. Moore. "Reinforcement learning: A survey." Journal of artificial intelligence research (1996): 237-285.

[7] Narendra, Kumpati S., and Snehasis Mukhopadhyay. "Associative learning in random environments using neural networks." Neural Networks, IEEE Transactions on 2.1 (1991): 20-31.

[8] Narendra, Kumpati S., and Snehasis Mukhopadhyay. "Intelligent control using neural networks." Control Systems, IEEE 12.2 (1992): 11-18.

[9] Sutton, Richard S. "Learning to predict by the methods of temporal differences." Machine learning 3.1 (1988): 9-44.

[10] Watkins, Christopher JCH, and Peter Dayan. "Q-learning." Machine learning 8.3-4 (1992): 279-292.

[11] Tilak, Omkar, and Snehasis Mukhopadhyay. "Partially decentralized reinforcement learning in finite, multi-agent Markov decision processes." AI Communications 24.4 (2011): 293-309.

[12] Littman, Michael L. "Markov games as a framework for multi-agent reinforcement learning." Proceedings of the eleventh international conference on machine learning. Vol. 157. 1994.

[13] Sutton, Richard S., Doina Precup, and Satinder Singh. "Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning." Artificial intelligence 112.1 (1999): 181-211.

[14] Lin, Long-Ji. "Self-improving reactive agents based on reinforcement learning, planning and teaching." Machine learning 8.3-4 (1992): 293-321.

[15] Narendra, Kumpati S., and Jeyendran Balakrishnan. "Adaptive control using multiple models." Automatic Control, IEEE Transactions on 42.2 (1997): 171-187.

[16] Narendra, Kumpati S., and Cheng Xiang. "Adaptive control of discrete-time systems using multiple models." Automatic Control, IEEE Transactions on 45.9 (2000): 1669-1686.

[17] Narendra, Kumpati S., and Osvaldo A. Driollet. "Stochastic adaptive control using multiple models for improved performance in the presence of random disturbances." International Journal of Adaptive Control and Signal Processing 15.3 (2001): 287-317.

[18] Narendra, Kumpati S., Yu Wang, and Wei Chen. "Stability, robustness, and performance issues in second level adaptation." American Control Conference (ACC), 2014. IEEE, 2014.

[19] Narendra, Kumpati S., Yu Wang, and Wei Chen. "The Rationale for Second Level Adaptation." The 16th Yale Workshop on Adaptive and Learning Systems, 2013

[20] Narendra, Kumpati S., Yu Wang, and Wei Chen. "Extension of Second Level Adaptation using Multiple Models to SISO Systems." 2015 American Control Conference (ACC). IEEE, 2015.

[21] Narendra,Kumpati S., Snehasis Mukhopadhyay, and Yu Wang, "Improving the Speed of Response of Learning Algorithms Using Multiple Models", The 17th Yale Workshop on Adaptive and Learning Systems, 2015

[22] Narendra, Kumpati S., and Mandayam AL Thathachar. Learning automata: an introduction. Courier Corporation, 2012.

[23] Tsetlin, Michael. L. "On behaviour of finite automata in random medium." Avtomat. i Telemekh 22.10 (1961): 1345-1354.

[24] Narendra, Kumpati S., and Mandayam AL Thathachar. "Learning automata-a survey." Systems, Man and Cybernetics, IEEE Transactions on 4 (1974): 323-334.

[25] Nedzelnitsky, Oleg V. and K. S. Narendra. "Nonstationary models of learning automata routing in data communication networks." Systems, Man and Cybernetics, IEEE Transactions on 17.6 (1987): 1004-1015.

[26] Viswanathan Ramanarayanan, and Kumpati S. Narendra. "Stochastic automata models with applications to learning systems." Systems, Man and Cybernetics, IEEE Transactions on 1 (1973): 107-111.

[27] Martin, Ryan, and Omkar Tilak. "On -optimality of the pursuit learning algorithm." Journal of Applied Probability (2012): 795-805.

[28] I. Joseph Shapiro, and Kumpati S. Narendra. "Use of stochastic automata for parameter self-optimization with multimodal performance criteria." Systems Science and Cybernetics, IEEE Transactions on 5.4 (1969): 352-360.

[29] Thathachar, Mandayam AL, and Pidaparty S. Sastry. Networks of learning automata: Techniques for online stochastic optimization. Springer Science & Business Media, 2011.

[30] Thathachar, M. A. L., and P. Shanti Sastry. "A new approach to the design of reinforcement schemes for learning automata." IEEE Transactions on Systems, Man, and Cybernetics 1 (1985): 168-175.