

Chapter 6

New Reinforcement Schemes for Stochastic Learning Automata

As described in Chapter 3, a variable structure stochastic automaton, having received the response of the environment, updates its action probabilities according to a reinforcement scheme. The reinforcement scheme is the mechanism generating the learning behavior of the stochastic automaton. It is of course based on the type of environment, and is generally classified according to two criteria:

- nature of the mapping from the previous probability vector to the new probability vector, e.g.:
 - (i) linear, nonlinear or hybrid or
 - (ii) projectional, nonprojectional¹
- the behavior of the learning automaton using the scheme, e.g., optimal, expedient.

Our efforts of designing an intelligent path controller for an autonomous vehicle using stochastic learning automata led us to employ four different reinforcement schemes. These are:

- Linear reward-penalty L_{R-P}^- (with $a = b$) or Bush-Mosteller scheme [Bush58]
- Multi-teacher general absolutely expedient scheme (MGAE), a nonlinear scheme under S-model environment [Baba83, Baba85]
- Linear reward-penalty L_{R-P}^+ scheme (general L_{R-P} with $a \neq b$) [Narendra89, Ünsal96]
- Nonlinear reinforcement scheme with H function (NL_H), an absolutely expedient scheme [Ünsal95]

The first two algorithms are widely known [Narendra89, Baba85]. Their convergence properties were investigated, and several applications using these schemes were previously presented [Chand69, Baba80]. The third algorithm is less popular, since the behavior of an automaton using this scheme have not been analytically proven [Narendra89]. A special case of this algorithm, $L_{R-\epsilon P}$, is obtained by adding a small penalty term ($0 < b \ll a < 1$) to the L_{R-I}

¹ See Section 3.5 for the definition of these terms.

scheme. However, the proof of ϵ -optimality for the general r -action case is again not complete [Narendra89]. The general linear reward-penalty scheme, L_{R-P} , is linear, nonprojectional. In Section 6.1.1, we will prove that this scheme is optimal in the case where there is an ‘pure optimal’ action.

The last scheme is nonlinear, projectional and is an extension of the general absolutely expedient schemes [Narendra89, Baba85]. It differs from previous schemes by the definition of an additional reward function. We will also show in this chapter that this scheme is absolutely expedient in stationary environments.

The last two schemes are used in this work for their improved behavior in specific cases frequently encountered in our application to intelligent vehicle control. Both schemes are found to be convergent to the optimal solution faster than the first two schemes. They are the direct results of our attempts to create reinforcement schemes with desirable characteristics suitable for this study of learning automata applications to intelligent vehicle control.

In the following section, we will compare the linear reinforcement scheme $L_{R-P}^\#$ to previously existing linear learning algorithms. The nonlinear reinforcement scheme NL_H is also compared to the general nonlinear scheme in Section 6.2. Since these two new schemes are found to be useful, we have investigated their convergence properties. Using the nonlinear stability theorems, we give the proof of optimality of the linear scheme $L_{R-P}^\#$ for a specific case. Furthermore, conditions of absolute expediency are checked with the new nonlinear scheme NL_H , and the scheme is proven to be absolutely expedient.

6.1 A Linear Reinforcement Scheme: Linear Reward-Penalty with Unequal Parameters, $L_{R-P}^\#$

Consider the general linear reward-penalty reinforcement scheme previously given in Section 3.5.1:

$$\begin{aligned} & \text{if } \alpha(n) = \alpha_i \\ & \text{for } \beta(n) = 0 \quad \begin{cases} p_i(n+1) = p_i(n) + a(1 - p_i(n)) \\ p_j(n+1) = (1-a)p_j(n) \end{cases} \quad j \neq i \end{aligned} \quad (6.1)$$

$$\text{for } \beta(n) = 1 \quad \begin{cases} p_i(n+1) = (1-b)p_i(n) \\ p_j(n+1) = \frac{b}{r-1} + (1-b)p_j(n) \end{cases} \quad j \neq i$$

where r is the number of actions. Parameters a and b are associated with the reward and penalty updates respectively. For $b = 0$, the algorithm is called linear reward-inaction scheme, and is known to be ϵ -optimal [Narendra89]. The specific case where $a = b$ is, generally, known as the linear reward-penalty scheme, and is expedient for stationary environments [Bush58, Narendra89].

An interesting phenomenon is seen in the case where $a = 0$ and $b \neq 0$. For an automaton with a single "optimal action" in a stationary environment and using this "linear inaction-penalty scheme" L_{I-P} , although the probability of the optimal action grows higher than others, it can never reach 1. In fact, it approaches $1/(r-1)$. Therefore, nothing can be said about the optimality of the scheme. Since this reinforcement scheme is relatively unimportant, we give the proof of this statement in Appendix B. If the parameter $b = 0$ while $0 < a < 1$, the algorithm is called L_{R-I} , the linear reward-inaction scheme. Again, several researchers studied the behavior and possible applications of this scheme [Viswanathan73, Shapiro69]. The case where the leaning parameters are unequal ($a \neq b$) have not studied rigorously by previous researchers² because the proof of optimality does not exist for this case.

The relative behavior of the L_{R-I} ($b = 0$), L_{R-P}^- ($a = b$) and L_{R-P}^+ ($a \neq b$) schemes is shown in Table 6.1 and Figure 6.1. The data show the convergence behavior of an optimal action in the best case possible (i.e., $c_a = 0$, $c_{j \neq a} = 1$) for different values of parameters a and b . We calculate the time for the optimal action's probability to reach approximately 1. The number of actions is 3; initial probability values are the same. For every parameter set, the reinforcement scheme is tested 500 times. The updating of the probabilities stops when the probability of the optimal action reaches 0.995. The simulation is stopped automatically at time step 1500 even if the convergence is not obtained. The top graph in Figure 6.1 shows the minimum, average, and the maximum number of steps for the parameters space $[0.1, 0.9] \times [0.0, 0.9]$. The bottom graph shows only the minimum and average number of steps; it is also scaled and rotated version of the first plot to emphasize the effects of the parameter b for smaller values of parameter a .

Simulation data show that the time to reach the pure optimal strategy depends on the learning parameters. Although Figures 6.1 and 6.2 show the whole parameter range, values larger than 0.3 are not generally used. As seen from the data in Table 6.1, and Figures 6.1 and 6.2, the convergence to pure optimal strategy is faster with L_{R-P}^- than L_{R-I} . The difference between these two reinforcement schemes is that the first one is expedient, while the latter is ϵ -optimal. Again, the simulation results show that when b is slightly less than a (e.g., L_{R-P}^+ with $[a, b] = [0.2, 0.1]$ or $[a, b] = [0.3, 0.1]$) the convergence to a pure optimal strategy is faster than with L_{R-P}^- .

² Except the case where the learning parameter b is much smaller than the parameter a , i.e., $L_{R-\epsilon P}$ scheme.

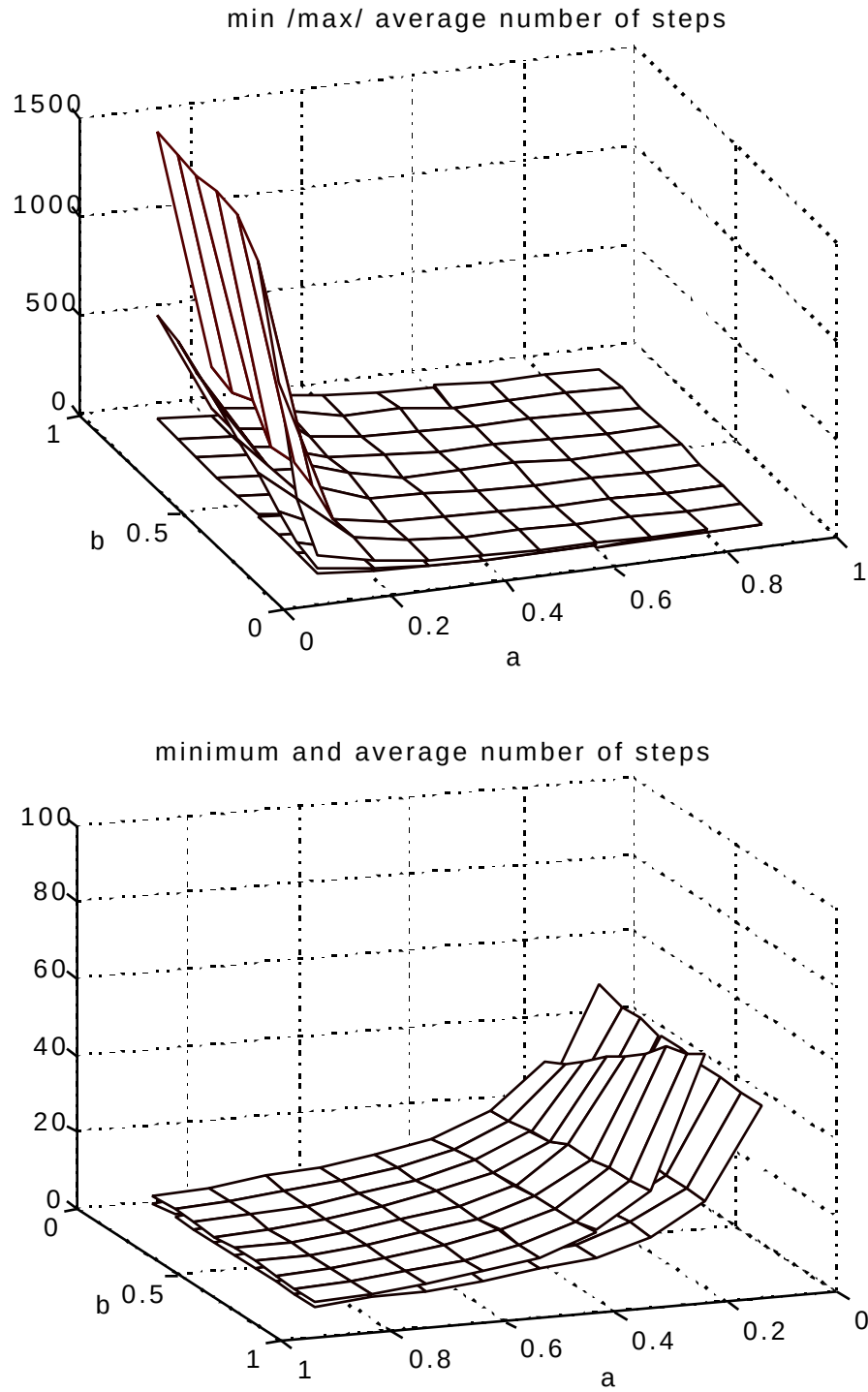


Figure 6.1. Number of steps needed for p_{opt} to reach 0.995 for different values of learning parameters a and b : (top) maximum, average and minimum number of steps; (bottom) average and minimum number of steps (over 500 runs; 3 actions; initial probabilities same; perfect case; note the axis shift and scaling in the second plot).

Scheme ³	Parameters		Number of steps for p_{opt} to reach 0.995					
	a	b	max.	min.	mean	median	std.dev.	Nonconv.
L_{R-I}	0.1	0.0	79	50	59.418	59	4.557	0
L_{R-P}^-	0.1	0.1	142	51	75.336	71	16.978	0
*	0.1	0.2	327	48	108.062	95	47.066	0
*	0.1	0.3	820	49	170.254	134	114.389	0
*	0.1	0.4	1338	48	266.16	204	218.733	0
*	0.1	0.5	1479	48	344.535	273.5	264.416	8
*	0.1	0.6	1497	47	425.219	326	338.649	17
L_{R-I}	0.2	0.0	42	23	29.124	28.5	3.532	0
*	0.2	0.1	76	23	32.548	32	6.328	0
L_{R-P}^-	0.2	0.2	105	23	34.940	33	8.737	0
*	0.2	0.3	96	23	38.594	35	1.744	0
*	0.2	0.4	164	23	43.440	38	1.901	0
*	0.2	0.5	181	23	46.884	39	2.168	0
*	0.2	0.6	169	23	51.136	42.5	2.941	0
L_{R-I}	0.3	0.0	36	15	19.068	18	3.199	0
*	0.3	0.1	39	15	20.326	20	3.659	0
*	0.3	0.2	43	15	21.164	20	5.006	0
L_{R-P}^-	0.3	0.3	47	15	21.966	21	5.754	0
*	0.3	0.4	60	15	22.814	21	6.988	0
*	0.3	0.5	97	15	25.130	22	10.211	0
*	0.3	0.6	72	15	24.964	22	9.943	0
L_{R-I}	0.4	0.0	26	11	14.278	14	2.767	0
*	0.4	0.1	27	11	14.788	14	3.223	0
*	0.4	0.2	29	11	15.090	14	3.233	0
*	0.4	0.3	31	11	15.308	14	3.642	0
L_{R-P}^-	0.4	0.4	51	11	15.618	14	4.469	0
*	0.4	0.5	42	11	16.002	14	4.973	0
*	0.4	0.6	51	11	16.396	15	5.745	0
L_{R-I}	0.5	0.0	24	9	11.978	11	2.533	0
*	0.5	0.1	22	9	11.672	11	2.472	0
*	0.5	0.2	23	9	11.574	11	2.689	0
*	0.5	0.3	26	9	11.844	11	3.024	0
*	0.5	0.4	31	9	11.824	11	3.428	0
L_{R-P}^-	0.5	0.5	30	9	11.772	11	3.243	0
*	0.5	0.6	27	9	11.886	11	3.325	0
L_{R-I}	0.6	0.0	26	7	9.540	9	2.750	0
*	0.6	0.1	24	7	9.536	9	2.349	0
*	0.6	0.2	20	7	9.240	9	2.096	0
*	0.6	0.3	19	7	9.322	9	2.132	0
*	0.6	0.4	23	7	9.446	9	2.519	0
*	0.6	0.5	31	7	9.814	9	2.982	0
L_{R-P}^-	0.6	0.6	29	7	9.588	9	3.104	0

Table 6.1. Behavior of the automata under general L_{R-P} scheme: 500 runs; maximum step =1500; 3 actions with $c = [0 \ 1 \ 1]$ (This data is plotted in Figure 6.1).

³ The mark ‘*’ indicates linear reward-penalty scheme L_{R-P}^- ($a \neq b$).

In some simulation runs, the probability p_{opt} did not reach 0.995. In this case, the statistical values are calculated by discarding the data for these runs, and therefore, do not present the actual behavior of the algorithm. The fact that the algorithm did not converge in 1500 steps does not exactly mean that it would not converge if the limit was increased. Similarly, the fact that we have convergence in all 500 runs does not guarantee convergence at all times

Figure 6.2 shows the result of the simulations for the parameters range $0 \leq b \leq 0.9$, $0.1 \leq a \leq 0.9$. For each parameter pair, 500 simulations are completed. The number of simulations which did not converge in 1500 steps are indicated with 'o'. The marker 'x' indicates the fact that the probability of the pure optimal action had converged to 0.995 in all 500 simulation runs. We know that for $a = 0$, convergence is not obtained no matter what the value of parameter b is (Appendix B). As seen from the plot, if the value of the parameter b is much larger than parameter a for small values of a , convergence again is not guaranteed. The exact limit of the 'non-convergence' region is not known since the number of simulations is 500, and the limit of time steps is 1500. Furthermore, this region is different for different numbers of actions.

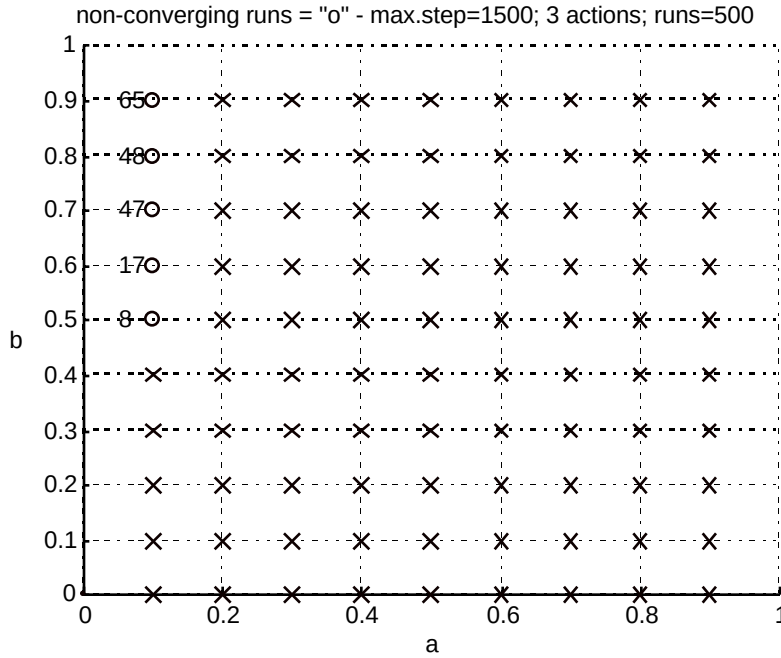


Figure 6.2. For some parameter values, convergence is not obtained for all 500 runs (Number of non-converging runs are shown next to marker 'o').

Table 6.2 gives the maximum, minimum and average number of steps for the optimal action's probability to converge to 0.995, for a smaller range of learning parameters. As seen from the table (and Figure 6.1), a slight change in the value of the parameter affects the speed of convergence to the pure optimal strategy. Especially for smaller values of parameter a , the maximum and consequently the average number steps decreases significantly with a decreasing

value of parameter b . For example, the maximum number of steps decreases from 105 to 48 when we switch from the L_{R-P}^- scheme to the L_{R-P}^+ scheme by decreasing the value of the parameter b from 0.2 to 0 while $a = 0.2$.

Scheme	Parameters		Number of steps for p_{opt} to reach 0.995		
	a	b	max.	min.	mean
L_{R-I}	0.10	0.00	79	50	59.418
L_{R-P}^+	0.10	0.05	122	49	67.080
L_{R-P}^-	0.10	0.10	142	51	75.336
L_{R-P}^+	0.10	0.20	327	48	108.062
L_{R-P}^-	0.10	0.30	820	49	170.254
L_{R-I}	0.20	0.00	42	23	29.124
L_{R-P}^+	0.20	0.05	48	23	31.038
L_{R-P}^-	0.20	0.10	76	23	32.548
L_{R-P}^-	0.20	0.20	105	23	34.940
L_{R-I}	0.30	0.00	36	15	19.068
L_{R-P}^+	0.30	0.10	39	15	20.326
L_{R-P}^-	0.30	0.20	43	15	21.164

Table 6.2. Behavior of the automata under general L_{R-P} scheme:
500 runs; maximum step = 1500; 3 actions with $c_1 = 0$, $c_2 = c_3 = 1$.

Changing the initial values of the action probability does not drastically change the behavior of the automata except to increase the number of time steps for convergence to the pure optimal strategy. Figure 6.3 shows the maximum, average and minimum number of steps for convergence with initial probability vector of $[0.005 \ 0.4975 \ 0.4975]$. Besides the increase in the number of steps needed for convergence, the comparison of Figures 6.1 and 6.3 shows that the effects of the learning parameters are comparable.

Similarly, increasing the number of actions also increases the number of time steps needed for convergence; the effect of the parameters is generally the same. As seen from Tables 6.2 and 6.3, the minimum number of steps for convergence is not affected significantly, but the maximum number of steps, and therefore the average, increases with an increasing number of actions.

Furthermore, if the probabilities of penalty for all non-optimal actions ($c_{j \neq \alpha}$) are less than 1, the convergence is again slower as expected (Figure 6.4). The change in the convergence rate is due to the decrease in the number of ‘correct’ updates on the action probability vector. The effects of the parameters is again similar to previous cases. The effect of the parameter b on the number of non-converging runs and the average number of steps for convergence is significant when learning parameter a is small.

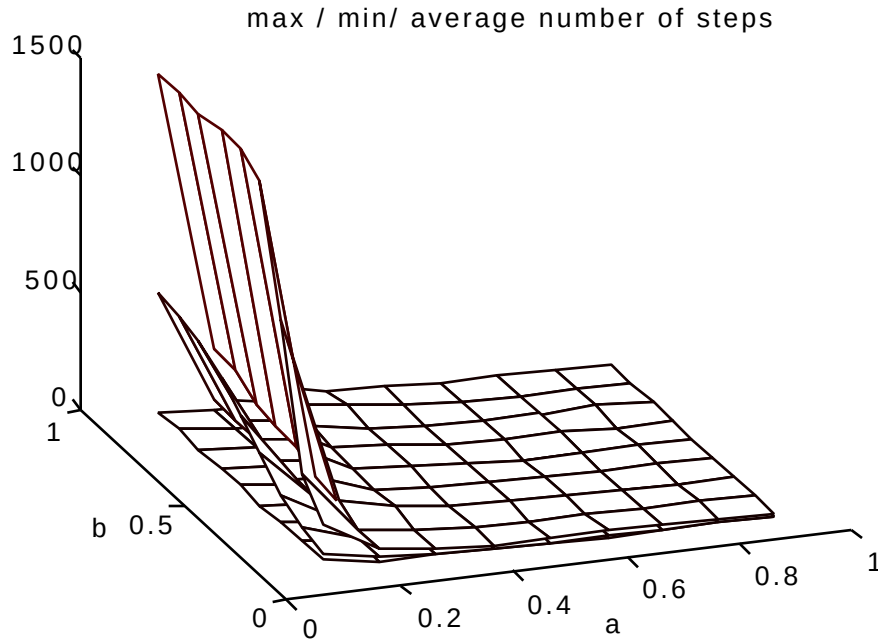


Figure 6.3 Number of steps needed for p_{opt} to reach 0.995 for different values of learning parameters a and b : maximum, average and minimum number of steps (over 500 runs; 3 actions; perfect case; $p(0) = [0.05 \ 0 \ 0.4975 \ 0.4975]$).

# of actions	Parameters		Number of steps for p_{opt} to reach 0.995		
	a	b	max.	min.	mean
4	0.10	0.10	292	50	98.2420
5			503	55	117.2060
7			459	65	153.9420
10			774	64	198.0060
4	0.20	0.00	50	24	31.3400
5			61	24	32.8320
7			85	25	36.1540
10			129	25	39.1120
4	0.20	0.10	70	24	36.2900
5			74	24	40.0940
7			101	26	43.9480
10			114	25	50.1940
4	0.20	0.20	138	24	44.5460
5			135	24	50.8560
7			256	25	64.1420
10			360	25	82.2700
4	0.30	0.30	92	16	27.2540
5			125	16	31.2400
7			117	16	38.5120
10			259	16	49.2380

Table 6.3. Effect of number of actions on behavior of the automata using general L_{R-P} scheme: 500 runs; maximum step = 1500; perfect case; all action probabilities initially equal.

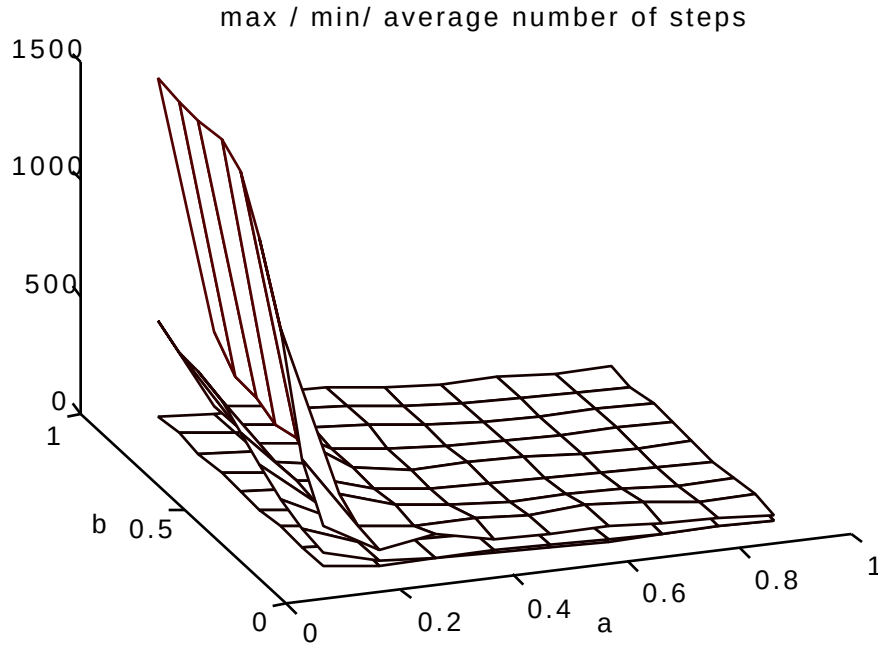


Figure 6.4 Number of steps needed for p_{opt} to reach 0.995 for different values of learning parameters a and b : maximum, average and minimum number of steps (500 runs; 3 actions; ideal case: $c = [0 \ 0.8 \ 0.5]$; initial probabilities are the same).

In simulations described in Chapters 4 and 5, the L_{R-P}^* reinforcement scheme is frequently used. Instead of increasing both learning parameters, decreasing the parameter associated with the penalty (namely b) resulted in a better performance. However, L_{R-P}^* reinforcement scheme is not widely used, nor there are complete results on learning behavior of an automata using this scheme. We will now show that this reinforcement scheme is expedient and ϵ -optimal for a special case.

6.1.1 Proof of Convergence for Linear Reward-Penalty Scheme with Unequal Learning Parameters, L_{R-P}^*

The linear reward-penalty reinforcement scheme with $a \neq b$ has been previously studied to some extent. Narendra and Thathachar defined a special case of L_{R-P}^* , the $L_{R-\epsilon P}$ scheme where the learning parameter associated with penalty (namely b) is much smaller than learning parameter associated with reward. In [Narendra89], this scheme is treated as a special case of the linear reward-inaction scheme L_{R-I} because only small values of the penalty parameter $b = \epsilon a$ where $0 < \epsilon \ll 1$ are considered. The reason for such a definition is that the behavior of the scheme can be studied by using the methods given in [Narendra89] provided that the parameters are small. However, even for small values of the parameter b (as well as a), this “linear approach” does not

result in a complete proof; “the establishment of ε -optimality of the $L_{R-\varepsilon P}$ scheme in the r -action case in general is incomplete” [Narendra89].

Here, we will consider all values of the parameters a and b and use the general stability theorem on nonlinear discrete-time systems [Kalman62] to prove the convergence for a specific region, without limiting the parameter b to small values. However, there is a price we have to pay for this; the use of the stability theorem limits this proof to a specific case where there is an “optimal” action⁴.

To prove the convergence to pure optimal action, we start with the conditional expectation of the action probabilities. From the definition of the $L_{R-P}^{\#}$ scheme (Equation 6.1), we may write the conditional expectation for the probability of an action at the next step as:

$$\begin{aligned}
 E[p_i(n+1)|p_i(n)] &= \sum_{k=1}^r E[p_i(n+1)|p_i(n) \quad \alpha(n) = \alpha_k] p_k(n) \\
 &= \sum_{k=1}^r c_k E[p_i(n+1)|p_i(n) \quad \alpha(n) = \alpha_k \quad \beta(n) = 1] p_k(n) \\
 &\quad + \sum_{k=1}^r (1-c_k) E[p_i(n+1)|p_i(n) \quad \alpha(n) = \alpha_k \quad \beta(n) = 0] p_k(n) \\
 &= c_i p_i(n)(1-b)p_i(n) + \sum_{k \neq i}^r c_k p_k(n) \frac{b}{r-1} + (1-b)p_i(n) \\
 &\quad + (1-c_i)p_i(n)[p_i(n) + a(1-p_i(n))] + \sum_{k \neq i}^r (1-c_k)p_k(n)(1-a)p_i(n)
 \end{aligned} \tag{6.2}$$

We can write this equality as:

$$\begin{aligned}
 E[p_i(n+1)|p_i(n)] &= p_i(n)(1-b) \sum_{k=1}^r c_k p_k(n) + \frac{b}{r-1} \sum_{k \neq i}^r c_k p_k(n) + (1-a)p_i(n) \sum_{k=1}^r p_k(n) \\
 &\quad - (1-a)p_i(n) \sum_{k=1}^r c_k p_k(n) + a p_i(n)(1-c_i)
 \end{aligned} \tag{6.3}$$

Rearranging terms and using the fact that $\sum_{k=1}^r p_k(n) = 1$, we get:

$$\begin{aligned}
 E[p_i(n+1)|p_i(n)] &= [(1-b) - (1-a)] p_i(n) \sum_{k=1}^r p_k c_k + \frac{b}{r-1} \sum_{k \neq i}^r p_k(n) c_k \\
 &\quad + (1-a) p_i(n) [1 + a p_i(n)(1-c_i)] \\
 &= (a-b) p_i(n) \sum_{k=1}^r p_k(n) c_k + \frac{b}{r-1} \sum_{k \neq i}^r p_k(n) c_k + (1-ac_i) p_i(n)
 \end{aligned} \tag{6.4}$$

⁴ The case where there is an “optimal” action is the most common case in stochastic automata applications. The convergence of the probability vector to the pure optimal strategy in this case is proven for almost all linear and nonlinear reinforcement schemes [Najim94, Narendra89].

For $a = b$, this corresponds to difference equation for the L_{R-P}^- scheme [Narendra89]. Now, taking the expected value of both sides, we obtain:

$$E[p_i(n+1)] = (a-b) \sum_{k=1}^r c_k E[p_i(n)p_k(n)] + \frac{b}{r-1} \sum_{k \neq i}^r c_k E[p_k(n)] + (1-ac_i) E[p_i(n)] \quad (6.5)$$

At this point, we will state two theorems for the stability of nonlinear discrete-time systems which will be used in our proof.

Theorem 3.1: Consider the discrete-time system

$$x(n+1) = f(x(n)) \quad (6.6)$$

where x is a vector, f is a vector with the property that $f(0) = 0$. Suppose there exists a scalar function $V(x)$ continuous in x such that:

- (i) $V(x) > 0$ for $x \neq 0$.
- (ii) $\Delta V(x) < 0$ for $x \neq 0$ where $\Delta V(x(n+1)) = V(x(n+1)) - V(x(n)) = V(f(x(n))) - V(x(n))$
- (iii) $V(0) = 0$.
- (iv) $V(x) \rightarrow \infty$ as $\|x\| \rightarrow \infty$.

Then, the equilibrium state $x = 0$ is asymptotically stable in the large and $V(x)$ is a Lyapunov function. Condition (ii) may be replaced with:

- (ii') $\Delta V(x) \leq 0$ for all, and $\Delta V(x)$ does not vanish identically for any solution sequence $\{x(n)\}$ satisfying Equation (6.6).

Theorem 3.2: If the function $f(x)$ defined above is a contraction, e.g.:

$$\|f(x)\| < \|x\| \quad \text{with} \quad f(0) = 0 \quad (6.7)$$

for some set of values $x \neq 0$ and some norm, then the system above is asymptotically stable and one of its Lyapunov functions is:

$$V(x) = \|x\| \quad (6.8)$$

These two theorems and their proofs are given in [Kalman62].

The general form of the difference Equation 6.5 is:

$$p_i(n+1) = \sum_{j \neq i}^r \mu_{ij} (p_i(n) p_j(n) + \sigma_{ij}^2(n)) + \sum_{k=1}^r \eta_{ik} p_k(n) \quad (6.9)$$

where:

$$\begin{aligned}
p_i(n) &\equiv E[p_i(n)] & i = 1, \dots, r \\
\sigma_{ij}^2(n) &\equiv \text{Cov}(p_i(n), p_j(n)) \\
\mu_{ik} &\equiv (a-b) c_k & k = 1, \dots, r \\
\eta_{ii} &\equiv 1 - a c_i \\
\eta_{ij} &\equiv \frac{b}{r-1} c_j & j \neq i
\end{aligned} \tag{6.10}^5$$

We first give the proof for a three-action automaton for the sake of clarity. For a three-action automaton with an “optimal action” ($c_1 = 0$), Equation 6.9 (or Equation 6.5) can be written as:

$$\begin{aligned}
p_1(n+1) &= \eta_{11}p_1(n) + \eta_{12}p_2(n) + \eta_{13}p_3(n) \\
&\quad + \mu_{11}p_1(n)^2 + \mu_{11}\sigma_{11}^2(n) + \mu_{12}p_1(n)p_2(n) + \mu_{12}\sigma_{12}^2(n) + \mu_{13}p_1(n)p_3(n) + \mu_{13}\sigma_{13}^2(n) \\
p_2(n+1) &= \eta_{21}p_1(n) + \eta_{22}p_2(n) + \eta_{23}p_3(n) \\
&\quad + \mu_{21}p_2(n)p_1(n) + \mu_{21}\sigma_{21}^2(n) + \mu_{22}p_2(n)^2 + \mu_{22}\sigma_{22}^2(n) + \mu_{13}p_2(n)p_3(n) + \mu_{23}\sigma_{23}^2(n) \\
p_3(n+1) &= \eta_{31}p_1(n) + \eta_{32}p_2(n) + \eta_{33}p_3(n) \\
&\quad + \mu_{31}p_3(n)p_1(n) + \mu_{31}\sigma_{31}^2(n) + \mu_{32}p_3(n)p_2(n) + \mu_{32}\sigma_{32}^2(n) + \mu_{33}p_3(n)^2 + \mu_{33}\sigma_{33}^2(n)
\end{aligned} \tag{6.11}$$

where:

$$\begin{aligned}
\mu_{j1} &= (a-b) \quad 0=0 & \mu_{jk} &= (a-b) c_k \\
\eta_{j1} &= \frac{b}{3-1} \quad 0=0 & \eta_{jk} &= 1 - ac_k \quad j = 2,3 \quad k = 2,3
\end{aligned} \tag{6.12}$$

Again, $\sigma_{ii}^2(n)$ are variances, and $\sigma_{ij}^2(n)$ are covariances for the action probabilities. Since the first action has zero penalty probability, the difference equations for the last two action probabilities does not include the probability of the first action. Therefore, the difference equations for the probabilities of the last two actions can be written as:

$$\begin{aligned}
p_2(n+1) &= (1 - ac_2)p_2(n) + \frac{b}{2}c_3p_3(n) \\
&\quad + (a-b)c_2p_2(n)^2 + (a-b)c_2\sigma_{22}^2(n) + (a-b)c_3p_2(n)p_3(n) + (a-b)c_3\sigma_{32}^2(n) \\
p_3(n+1) &= \frac{b}{2}c_2p_2(n) + (1 - ac_3)p_3(n) \\
&\quad + (a-b)c_2p_3(n)p_2(n) + (a-b)c_2\sigma_{32}^2(n) + (a-b)c_3p_3(n)^2 + (a-b)c_3\sigma_{33}^2(n)
\end{aligned} \tag{6.13}$$

⁵ We omit the operator $E[\cdot]$ for simplicity.

Let us define a function f of the expected values of the last two probabilities as⁶:

$$f\left(\begin{matrix} \hat{p}_2 \\ \hat{p}_3 \end{matrix}\right) = \begin{matrix} \hat{e}_2 \\ \hat{e}_3 \end{matrix} \left((1-ac_2)p_2 + \frac{b}{2}c_3p_3 + (a-b)c_2p_2p_2 + (a-b)c_2\sigma_{22}^2 + (a-b)c_3p_2p_3 + (a-b)c_3\sigma_{23}^2 \right) \\ + \begin{matrix} \hat{e}_3 \\ \hat{e}_2 \end{matrix} \left((1-ac_3)p_3 + \frac{b}{2}c_2p_2 + (a-b)c_2p_3p_2 + (a-b)c_2\sigma_{32}^2 + (a-b)c_3p_3p_3 + (a-b)c_3\sigma_{33}^2 \right) \quad (6.14)$$

If we can prove that this function is a contraction using the 1-norm⁷, then we can state that the expected value of the probabilities of the last two actions asymptotically converge to 0, by using Theorems 3.1 and 3.2. Thus, the probability p_I of the optimal action asymptotically approaches 1.

The contraction condition is:

$$\left\| \begin{matrix} \hat{e}_2 p_2(n) \\ \hat{e}_3 p_3(n) \end{matrix} \right\|_1 = \left| (1-ac_2)p_2 + \frac{b}{2}c_3p_3 + (a-b)(c_2p_2^2 + c_2\sigma_{22}^2 + c_3p_2p_3 + c_3\sigma_{23}^2) \right| \\ + \left| (1-ac_3)p_3 + \frac{b}{2}c_2p_2 + (a-b)(c_3p_3^2 + c_3\sigma_{33}^2 + c_2p_2p_3 + c_2\sigma_{32}^2) \right| \\ \stackrel{?}{<} |p_2(n)| + |p_3(n)| = \left\| \begin{matrix} \hat{e}_2 p_2(n) \\ \hat{e}_3 p_3(n) \end{matrix} \right\|_1 \quad (6.15)$$

Since, a, b, c_i, p_i are all parameters/variables in $[0, 1]$, and the last terms in the absolute values are always positive⁸ for all n , we can write:

$$\left\| \begin{matrix} \hat{e}_2 p_2(n) \\ \hat{e}_3 p_3(n) \end{matrix} \right\|_1 \leq (1-ac_2)p_2 + \frac{b}{2}c_3p_3 + |a-b| (c_2p_2^2 + c_2\sigma_{22}^2 + c_3p_2p_3 + c_3\sigma_{23}^2) \\ + (1-ac_3)p_3 + \frac{b}{2}c_2p_2 + |a-b| (c_3p_3^2 + c_3\sigma_{33}^2 + c_2p_2p_3 + c_2\sigma_{32}^2) \quad (6.16)$$

Rearranging the terms, we have:

$$\left\| \begin{matrix} \hat{e}_2 p_2(n) \\ \hat{e}_3 p_3(n) \end{matrix} \right\|_1 \leq (1-ac_2)p_2 + \frac{b}{2}c_2p_2 + |a-b| (c_2p_2^2 + c_2\sigma_{22}^2 + c_2p_2p_3 + c_2\sigma_{23}^2) \\ + (1-ac_3)p_3 + \frac{b}{2}c_3p_3 + |a-b| (c_3p_3^2 + c_3\sigma_{33}^2 + c_3p_2p_3 + c_3\sigma_{32}^2)$$

⁶ By omitting the terms (n) for simplification.

⁷ Denoted here as $\|\cdot\|_1$.

⁸ All the terms of the sum are positive: the first term includes the square of the expected value, and the second term, the variance which are always positive; the third and fourth terms can be combined as $c_i(p_i p_j + \sigma_{ij}^2) = c_i E[p_i p_j]$. Expected value of the multiplication of any two action probabilities is always positive since $0 < p_i < 1$.

(6.17)

Replacing $\frac{b}{2}$ by b without invalidating the inequality, and using the identity $E[x - y] = E[x] - E[y] + \sigma_{xy}^2$, we obtain:

$$\begin{aligned} \left\| \begin{matrix} a \\ c \\ e \end{matrix} p_2(n) \right\|_1 &< (1 - ac_2)p_2 + bc_2p_2 + |a - b| c_2(E[p_2^2] + E[p_2p_3]) \\ &+ (1 - ac_3)p_3 + bc_3p_3 + |a - b| c_3(E[p_3^2] + E[p_2p_3]) \end{aligned} \quad (6.18)$$

We know that $E[p_i^2] + E[p_ip_j] = E[p_i^2 + p_ip_j] = E[p_i(p_i + p_j)] < E[p_i]$ since the sum of all probabilities is equal to 1. Therefore:

$$\left\| \begin{matrix} a \\ c \\ e \end{matrix} p_2(n) \right\|_1 < (1 - ac_2)p_2 + bc_2p_2 + |a - b| c_2p_2 + (1 - ac_3)p_3 + bc_3p_3 + |a - b| c_3p_3 \quad (6.19)$$

Assuming $a > b$, we get:

$$\left\| \begin{matrix} a \\ c \\ e \end{matrix} p_2(n) \right\|_1 < (1 - ac_2 + bc_2 + ac_2 - bc_2)p_2 + (1 - ac_3 + bc_3 + ac_3 - bc_3)p_3 = p_2 + p_3 = \left\| \begin{matrix} p_2(n) \\ p_3(n) \end{matrix} \right\|_1 \quad (6.20)$$

Therefore, for the region $a > b$ (Figure 6.5), the expected values of the probabilities of the sub-optimal actions all converge to zero. Since the sum of all probabilities is equal to 1, the expected value of the optimal action must converge to 1. Figure 6.6a shows the changes in the probabilities of the suboptimal actions for a linear reward-penalty scheme with $a = 0.2$ and $b = 0.1$ with initial probabilities of $p(0) = [0.05 \ 0.50 \ 0.45]$. As seen from the plot, probabilities p_2 and p_3 do not asymptotically converge to 0. In 83 steps, the probability p_1 of the optimal action reaches 0.999.

The cumulative averages $\bar{p}_i(n) = \sum_{k=0}^n p_i(k)$ ($n = 0, 1, \dots, 83, i = 1, 2$) are plotted in Figure 6.6b.

They converge asymptotically to 0.

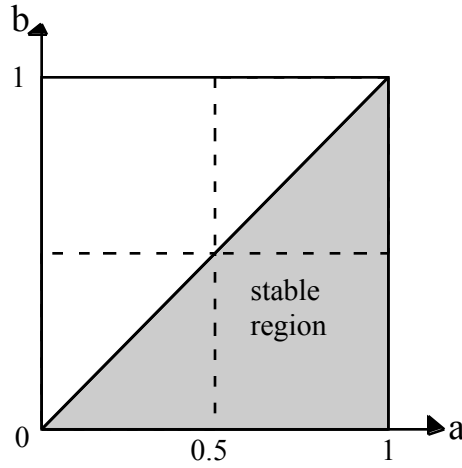


Figure 6.5. The region for asymptotic convergence to pure optimal strategy.

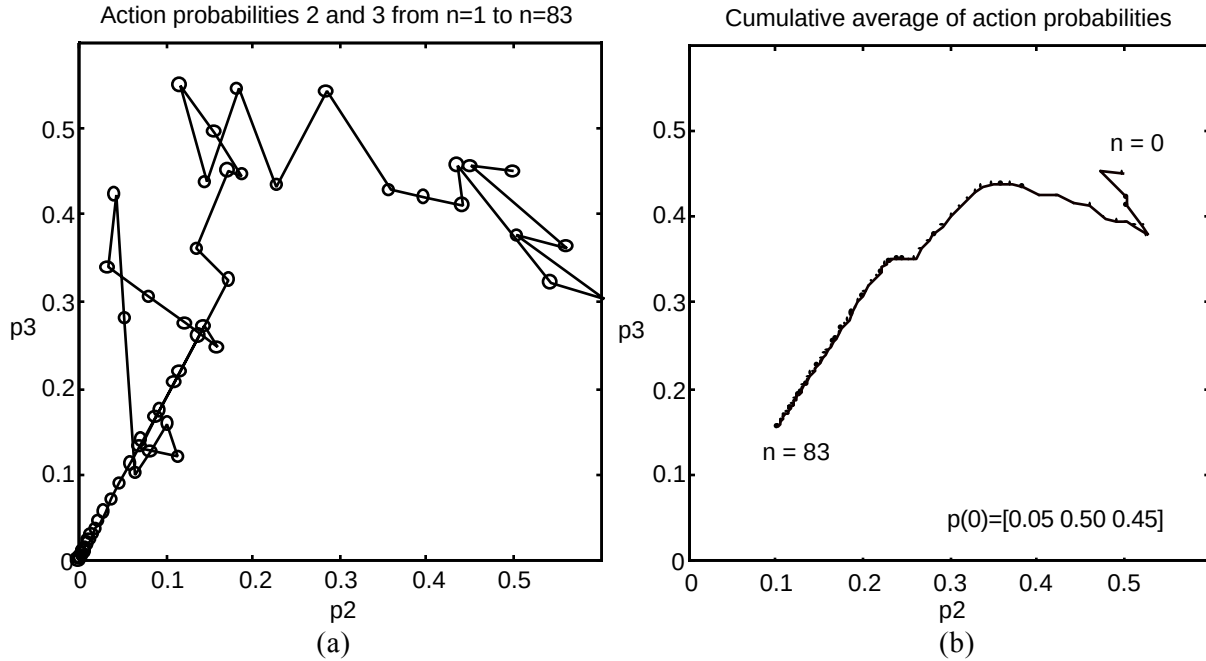


Figure 6.6. Probabilities (a) and cumulative average (b) of the probabilities of the non-optimal actions for a 3-action automata with L_{R-P}^{π} ($a = 0.2$, $b = 0.1$).

For $r = N$ actions with $\bar{p}(n) = [p_1(n) \ p_2(n) \ \dots \ p_r(n)]$, the contraction condition can be written as:

$$\left\| \begin{pmatrix} p_2(n) \\ p_3(n) \\ \vdots \\ p_N(n) \end{pmatrix} \right\| < \left\| \begin{pmatrix} p_2(n) \\ p_3(n) \\ \vdots \\ p_N(n) \end{pmatrix} \right\|_1 \equiv \sum_{i=2}^N |p_i(n)| \quad (6.21)$$

To prove this condition, we start with the standard norm inequality:

$$\begin{aligned} \|f(\bar{p}(n))\| &= \sum_{i=2}^N \left| (1-ac_i)p_i + \sum_{\substack{j=2 \\ j \neq i}}^N \frac{b}{N-1} c_j p_j + \sum_{\substack{j=2 \\ j \neq i}}^N (a-b)c_j (p_i p_j + \sigma_{ij}^2) \right| \\ &\leq \sum_{i=2}^N |(1-ac_i)p_i| + \sum_{i=2}^N \sum_{\substack{j=2 \\ j \neq i}}^N \left| \frac{b}{N-1} c_j p_j \right| + \sum_{i=2}^N \sum_{\substack{j=2 \\ j \neq i}}^N |(a-b)c_j (p_i p_j + \sigma_{ij}^2)| \end{aligned} \quad (6.22)$$

Again, $(1-ac_i)$, $p_i(n)$, c_i , b , $(a-b)$ are all positive values, and the sum of all $(p_i p_j + \sigma_{ij}^2)$ has positive values for all time steps. Therefore:

$$\|f(\bar{p}(n))\| \leq \sum_{i=2}^N (1-ac_i)p_i + \frac{b}{N-1} \sum_{i=2}^N \sum_{\substack{j=2 \\ j \neq i}}^N c_j p_j + (a-b) \sum_{i=2}^N \sum_{\substack{j=2 \\ j \neq i}}^N c_j (p_i p_j + \sigma_{ij}^2) \quad (6.23)$$

More explicitly:

$$\begin{aligned} \|f(\bar{p}(n))\| &\leq \begin{pmatrix} (1-ac_2)p_2 \\ (1-ac_3)p_3 \\ \vdots \\ (1-ac_N)p_N \end{pmatrix} + \frac{b}{N-1} \begin{pmatrix} (c_2 p_2 + c_3 p_3 + \dots + c_N p_N) \\ (c_3 p_2 + c_4 p_4 + \dots + c_N p_N) \\ \vdots \\ (c_2 p_2 + c_3 p_3 + \dots + c_{N-1} p_{N-1} + c_N p_N) \end{pmatrix} \\ &\quad + (a-b) \begin{pmatrix} c_2(p_2^2 + \sigma_{22}^2 + p_2 p_3 + \sigma_{23}^2 + \dots + p_2 p_N + \sigma_{2N}^2) \\ c_3(p_2 p_3 + \sigma_{32}^2 + p_3^2 + \sigma_{33}^2 + \dots + p_3 p_N + \sigma_{3N}^2) \\ \vdots \\ c_N(p_2 p_N + \sigma_{N2}^2 + p_3 p_N + \sigma_{N3}^2 + \dots + p_N^2 + \sigma_{NN}^2) \end{pmatrix} \end{aligned} \quad (6.24)$$

Rearranging, and using $E[x \cdot y] = E[x] E[y] + \sigma_{xy}^2$, we have:

And, since the probability vector converges to the unit vector corresponding to the optimal strategy, the expected value of the average penalty converges to zero. Therefore, such an automaton is expedient and optimal (see the definitions in Chapter 3).

Although I_{R-P}^* is not absolutely expedient like nonlinear schemes given in Chapter 3 and Section 6.2, it is optimal for a specific but widely encountered environment condition. It still remains to extend the proof to the region $\{(a,b) \in [0,1] \times [0,1] \mid a < b\}$ (Figure 6.5) if at all possible. Also, the proof can again be extended to all possible environments by considering values $c_{opt} > 0$. However, in this case, previous linearization approaches [Narendra89] and the contraction theorems 3.1 and 3.2 cannot be used.

6.1.2 Notes on the “Ideal Environment” and Penalty Probabilities

The proof given in the previous section for general the linear reward-penalty reinforcement scheme is valid for an automaton in an “ideal environment.” The word “ideal” here refers to the fact that there is an “optimal action” for which the probability of receiving a penalty is zero. In such a case, the automaton is expected to reach “pure optimal strategy.” In the literature, both the conditions where $\min \{c_i\} = 0$ and $\min \{c_i\} > 0$ are studied. When there is no optimal action, the action probabilities converge to values conversely proportional to probabilities of penalty [Narendra89, Najim94]. We have studied the ideal environment in depth because this case is often encountered in applications. Almost all of the learning automata applications are defined in ideal P-model environments [Baba85, Najim94]. In all the applications discussed in this thesis, the environment is defined so that there is always one action with zero probability of penalty. This action is “the optimal action” for the current situation or the “solution” to the problem at hand.

Throughout this dissertation, we defined the probabilities of penalty for each action as a constant in the range of $[0,1]$ where zero corresponds to no penalty at all, and 1 corresponds to guaranteed penalty. Although these values may be time varying, most of the literature except [Najim94, Baba85] consider these to be deterministic. On the other hand, it is interesting to note that all proofs in this chapter and in the previous literature carry directly to a noisy environment by defining the penalty probabilities as random variables. Assuming that the probability of penalty from the environment for any action is a random variable of mean m_i and variance v_i , all c_i 's in difference equations can be simply replaced by m_i 's to accommodate this new definition¹⁰.

For all the linear reinforcement schemes we mentioned so far, it has been shown here or elsewhere that convergence to optimal or pure optimal strategy is guaranteed, sometimes based on several conditions on the parameters and the environment. However, there is not much to say about the expediency of these schemes, especially absolute expediency. For a learning algorithm, it is important to be able to show that the learning characteristic is kept at every step, *i.e.*, the automaton does better than the previous step at every time iteration. This need has led to a

¹⁰ This is because the probabilities of penalty c_i are independent from action probabilities p_i .

synthesis approach toward reinforcement schemes, and the first result was the absolutely expedient nonlinear scheme of Lakshmivarahan and Thathachar [Lakshamivarahan73].

Definitions of several nonlinear reinforcement schemes were given in Section 3.5.2. In the next section, we will introduce a new nonlinear scheme that works better than the previously defined schemes [Baba83, Baba85] for several applications where fast convergence under certain conditions is necessary.

6.2. A Nonlinear Reinforcement Scheme: NL_H

Consider the general nonlinear reinforcement scheme given in Section 3.5.2 (see Equations 3.20 and 3.27). For a multi-teacher P-model environment the general absolutely expedient scheme can be written as:

if $\alpha(n) = \alpha_i$,

$$\begin{aligned} p_i(n+1) &= p_i(n) + \frac{\beta_i^1 + \beta_i^N}{N} \sum_{\substack{j=1 \\ j \neq i}}^r \varphi_j(\bar{p}(n)) - \frac{\beta_i^1 + \beta_i^N}{N} \sum_{\substack{j=1 \\ j \neq i}}^r \psi_j(\bar{p}(n)) \\ p_j(n+1) &= p_j(n) - \frac{\beta_i^1 + \beta_i^N}{N} \sum_{\substack{j=1 \\ j \neq i}}^r \varphi_j(\bar{p}(n)) + \frac{\beta_i^1 + \beta_i^N}{N} \sum_{\substack{j=1 \\ j \neq i}}^r \psi_j(\bar{p}(n)) \quad \text{for all } j \neq i \end{aligned} \quad (6.28)$$

where the functions φ_i and ψ_i satisfy the following conditions:

$$\begin{aligned} \frac{\varphi_1(\bar{p}(n))}{p_1(n)} &= \frac{\varphi_r(\bar{p}(n))}{p_r(n)} = \lambda(\bar{p}(n)) \\ \frac{\psi_1(\bar{p}(n))}{p_1(n)} &= \frac{\psi_r(\bar{p}(n))}{p_r(n)} = \mu(\bar{p}(n)) \\ p_j(n) + \psi_j(\bar{p}(n)) &> 0 \\ p_i(n) + \sum_{\substack{j=1 \\ j \neq i}}^r \varphi_j(\bar{p}(n)) &> 0 \\ p_j(n) - \varphi_j(\bar{p}(n)) &< 1 \end{aligned} \quad (6.29)$$

for all $i, j = 1, \dots, r$.

The functions $\psi(x)$ and $\varphi(x)$ can be chosen arbitrarily as long as they satisfy the conditions given above. Previous attempts to find absolutely expedient schemes led to nonlinear functions. The only absolutely expedient linear scheme is the L_{R-1} ($b=0$) scheme. In this case, $\psi(x) = -a$ and $\varphi(x) = 0$, and they satisfy the above conditions. The following theorem given in [Narendra89] and [Najim94], states the necessary and sufficient conditions for absolute expediency:

Theorem 3.4: If the functions $\lambda(\bar{p})$ and $\mu(\bar{p})$ satisfy the following conditions:

$$\begin{aligned}\lambda(\bar{p}) &\leq 0 \\ \mu(\bar{p}) &\leq 0 \\ \lambda(\bar{p}) + \mu(\bar{p}) &< 0\end{aligned}\tag{6.30}$$

then the automaton with the reinforcement scheme in Equations 6.28 and 6.29 is absolutely expedient in a stationary environment.

In our applications, we defined a single environment response that is a function of multiple (sensor) responses. However, the theorem is valid for any number of teachers in an environment. A well-known description of functions $\lambda(p)$ and $\mu(p)$ is given in Section 3.5.2. The resulting reinforcement scheme is called multi-teacher general absolutely expedient (MGAE) scheme in [Baba85]. Our reinforcement scheme differs from the one given in Section 3.5.2 (Equations 3.27 and 3.28) by the definition of these two functions. In the form of Equation 6.28, we can write our update algorithm as:

Nonlinear reinforcement scheme NL_H

$$\begin{aligned}\text{if } \alpha(n) = \alpha_i \\ p_i(n+1) &= p_i(n) + f(-k\theta H(n)) [1 - p_i(n)] - [1 - f](-\theta) [1 - p_i(n)] \\ p_j(n+1) &= p_j(n) - f(-k\theta H(n)) p_j(n) + [1 - f](-\theta) p_j(n) \text{ for all } j \neq i\end{aligned}\tag{6.31}$$

i.e.:

$$\begin{aligned}\psi_k(\bar{p}(n)) &\equiv -\theta p_k(n) \\ \phi_k(\bar{p}(n)) &\equiv -k\theta H(n) p_k(n)\end{aligned}\tag{6.32}$$

where learning parameters k and θ are real valued and satisfy:

$$\begin{aligned}0 < \theta < 1 \\ 0 < k\theta < 1\end{aligned}\tag{6.33}$$

The next section introduces the function H used in our reinforcement scheme and the reasoning behind this choice. We also prove that the definitions above satisfy all necessary and sufficient conditions for absolute expediency. The comparison of the new nonlinear scheme with the general absolutely expedient scheme concludes this chapter.

6.2.1 Function H and the Conditions for Absolute Expediency

In Equation 6.31, the feedback f is some combination of teacher outputs¹¹, and the function H is defined as (see also Figure 6.7):

$$H(n) \equiv \min_{\hat{i}} \frac{1}{\hat{i}}; \max_{\hat{i}} \frac{p_i(n)}{k\theta(1 - p_i(n))} - \varepsilon; 0 \leq H(n) \leq 1\tag{6.34}$$

¹¹ It could be the average of all teacher outputs as in Equation X.28 or some nonlinear combination such as the one we define in Section 4.x.x.

Parameter ϵ is an arbitrarily small positive real number. Also note that the function H includes p_i which is the action probability corresponding to the current action.

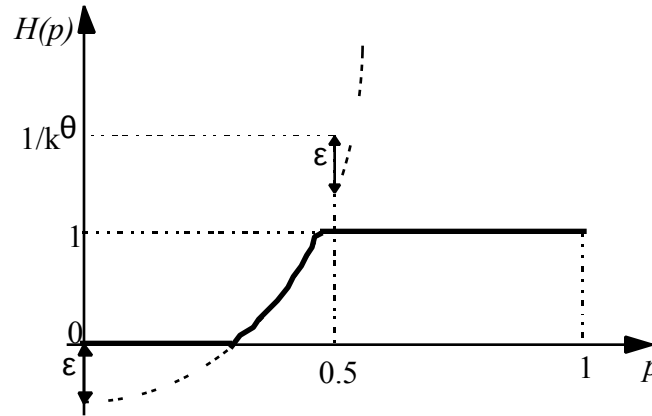


Figure 6.7. Sketch of the function H .

We now show that the defined algorithm (Equations 6.31-6.34) satisfies all the conditions in Equations 6.29. From Equation 6.32, we have:

$$\frac{\varphi_k(\underline{P}(n))}{p_k(n)} = \frac{-k \theta H(n) p_k(n)}{p_k(n)} = -k \theta H(n) \equiv \lambda(\underline{P}(n)) \quad (6.35)$$

and:

$$\frac{\psi_k(\underline{P}(n))}{p_k(n)} = \frac{-\theta}{p_k(n)} = -\theta \equiv \mu(\underline{P}(n)) \quad (6.36)$$

That is, our definition is consistent with the first two conditions of Equation 6.29. There are three remaining necessary and sufficient conditions for absolute expediency. Using Equations 6.31 and 6.32, the rest of the conditions on ϕ_i and ψ_i translates to the following:

$$\begin{aligned} \text{(a)} \quad & p_i(n) + \theta \quad (1 - p_i(n)) < 1 \\ \text{(b)} \quad & p_j(n) - \theta \quad p_j(n) > 0 \\ \text{(c)} \quad & p_i(n) - k \quad \theta \quad H(n) \quad (1 - p_i(n)) > 0 \end{aligned} \tag{6.37}$$

Conditions (a) and (b) are associated with the reward updates while condition (c) is associated with the penalty updates. These conditions guarantee that the probabilities stay in the range $(0, 1)$ at all times (with the assumption that none of the probabilities is initially 0 or 1). Conditions (a) and (b) can be shown to be satisfied using the fact that the sum of all probabilities is 1:

$$(a) \quad p_i + \theta \sum_{\substack{j=1 \\ j \neq i}}^r p_j < p_i + \theta \sum_{\substack{j=1 \\ j \neq i}}^r p_j = 1 \text{ since } 0 < \theta < 1 \quad (6.38)$$

$$(b) \ p_j(n) - \theta \ p_j(n) = p_j \ (1 - \theta) > 0 \text{ since } 0 < p_j < 1 \text{ and } 0 < \theta < 1 \quad (6.39)$$

For the third condition, we have:

$$(c) \quad \begin{aligned} p_i(n) - k \ \theta \ H(n) \ (1 - p_i(n)) > 0 &\Leftrightarrow k \ \theta \ H(n) \ (1 - p_i(n)) < p_i(n) \\ &\Leftrightarrow H(n) < \frac{p_i(n)}{k \ \theta \ (1 - p_i(n))} \end{aligned} \quad (6.40)$$

This condition is already satisfied by the previous definition of the function $H(n)$. For the limiting values of $H = 0$ and $H = 1$ (Equation 6.34), we have the following:

♦ $H = 0$ $p_i(n) > 0$. This is true for all values of the action probabilities p_i .

♦ For $H = 1$, we must have $p_i(n) - k \ \theta \ (1 - p_i(n)) > 0$ to satisfy condition (c). From the definition of the function, we conclude that $H = 1$ implies:

$$\frac{p_i(n)}{k \ \theta \ (1 - p_i(n))} - \varepsilon \geq 1$$

This inequality can be rewritten as:

$$p_i(n) \geq (1 + \varepsilon) \ k \ \theta \ (1 - p_i(n))$$

or, omitting the time step variable and rearranging:

$$p_i - k \ \theta \ (1 - p_i) - \varepsilon \ k \ \theta \ (1 - p_i) \geq 0$$

Since all the factors of the third term are strictly positive real, we may omit this term without affecting the inequality to obtain:

$$p_i - k \ \theta \ (1 - p_i) > 0 \quad (6.41)$$

which is exactly what we must have for $H = 1$.

Thus, condition (c) is satisfied for the limiting values of function H too. With this, we conclude that all five conditions of the Equation 6.29 are satisfied. Therefore, the reinforcement scheme NL_H is a candidate for absolute expediency.

Furthermore, the functions λ and μ for nonlinear scheme NL_H satisfy the following:

$$\begin{aligned} \lambda(\bar{p}(n)) &= -\theta < 0 \\ \mu(\bar{p}(n)) &= -k \ \theta \ H \leq 0 \\ \lambda(\bar{p}(n)) + \mu(\bar{p}(n)) &= -\theta - k \ \theta \ H < 0 \end{aligned} \quad (6.42)$$

because $0 < \theta < 1$, $0 < k \ \theta < 1$, and $0 < H < 1$. Since the conditions above are sufficient for absolute expediency, and we know that the nonlinear reinforcement scheme NL_H satisfies all the conditions listed in the definition of a candidate nonlinear reinforcement scheme, we state the algorithm given in Equations 6.31-6.34 is absolutely expedient in a stationary environment.

6.2.2 Comparison of NL_H with General Absolutely Expedient Scheme

The choice of the function H is also due to another factor besides the conditions given in Equation 6.29. Our definition is found to work better than the general MGAE Algorithm given in [Baba85], considering the convergence to a solution. We compared two reinforcement schemes using three actions and two different initial conditions (Table 6.1). Our definition results in a faster update in general as shown in Table 6.4. The average number of steps for the pure optimal action's probability to reach 0.995 are given for different values of the learning parameters k and θ . Two different cases are considered.

The definition of the function H (Equation 6.34) is based on the fact that the update rate for a penalty response from the environment is higher when the probability of the current action is close to 1. (Note that H is a function of the probability of the current action p_i ; it is not related to the index j in Equation 6.28.) This provides a much faster convergence rate when the actions receiving a penalty from the environment have high probability values associated with them.

Parameters		Average number of steps to reach $p_{opt}(0) = 0.995$			
		(a) 3 actions with $p(0) = [1/3 \ 1/3 \ 1/3]$		(b) 3 actions with $p_{opt}(0)=0.005,$ $p_{j \neq opt}(0)=0.4975$	
		algorithm in [Baba85]	New algorithm with H	algorithm in [Baba85]	New algorithm with H
1	0.010	544.99	304.48	952/94	534.42
	0.035	155.67	155.59	358.46**	159.11
	0.050	108.79	66.22	263.84**	113.98
	0.100	54.11	35.13	184.85	60.13
	0.200	26.85	19.21	141.64	31.16
2	0.050	102.80	48.36	259.77	77.60
	0.100	51.42	27.57	185.83	41.70
	0.200	25.58	15.73	117.84	23.53
* 80 runs did not converge in 1200 steps.		H function is defined in Equation 6.34.			
** 1 run did not converge in 1200 steps.		500 runs for each parameter set.			

Table 6.4. Convergence rates for a single *optimal* action of a 3-action automaton in a stationary environment.

The data shown in Table 6.4 are the results of two different initial conditions: (a) all probabilities are initially the same, only one action is receiving reward, and (b) the only action receiving a reward¹² from the environment has a very small probability value. The difference in convergence rate is more distinct in the situation where the probability of the optimal action is initially very close to 0¹³. As seen in Table 6.4, when $p_{opt} = 0.005$, the number of iteration steps

¹² I.e., the *optimal* action.

¹³ This situation occurs frequently in our application to automated highway systems; for example, while the probability of the lateral action *shift left* is converging to 1, a vehicle may enter the left sensor range. In this case, we need a “strong” penalty update to decrease the probability of this action, while “encouraging” the action *stay in lane*.

to reach $p_{\text{opt}} = 0.995$ is reduced drastically for relatively large values of the learning parameter θ . Especially when both learning parameters are large (see the shaded area in Table 6.4), the difference between the average number of steps for two schemes is threefold or more. In order to have a fast update on the probability vector, the function H is set to the highest possible value (see Equation 6.34) satisfying the conditions for φ_i in Equation 6.29.