KYBERNETIKA — VOLUME *26* (1990), NUMBER 4

# ON THE BEHAVIOUR OF LEARNING ALGORITHMS IN A CHANGING ENVIRONMENT WITH APPLICATION TO DATA NETWORK ROUTING PROBLEM

ATHANASIOS V. VASILAKOS

In data networks the message has peak and slack periods and the topology may change. A learning automation is situated at each node of the network where a routing decision must be made and directs traffic entering the node onto one of the outgoing links. Using network feedback, an automaton modifies its routing strategy to improve its link selections. This approach has the advantage over existing routing schemes of offering a simple and extremely practical feedback and updating policy. A new model of a nonstationary automaton environment is proposed and the limiting behaviour of this model is analysed. Simulation studies of automata operating in simple networks verify the analytical results.

## 1. INTRODUCTION

Learning algorithms has been suggested in the past in circuit switched telephone networks. In this paper, we focus on the packet-switched routing problem, with learning automata proposed for virtual and datagram networks. The virtual network function is similar to a circuit-switched network: the routing mechanism establish a virtual connection between source and destination, which can then be used to transfer the data packets. Datagram networks on the other hand, treat the data packets as individual entities and the routing algorithm selects any allowable path, spreading the traffic efficiently over the available capacity of the network.

In this paper we develop an abstract model of the network that is functionally dependent on the automaton's strategies in order to study the equilibrium and transient behaviour of the system. A dynamic model in which the environment is characterized by a set of state variables is needed to correctly represent the network's operation. The state of the system converges in distribution and the equilibrium behaviour of the system can be characterized.

The learning algorithms are compared against existing techniques and has been shown to provide a superior strategy when non-stationary network conditions prevail.

## 2. LEARNING AUTOMATA IN NON-STATIONARY ENVIRONMENTS

A learning automaton operates in such a manner as to choose an optimal action from an allowable set and to apply the selected action to a non-stationary environment. In turn, the environment responds with a feedback signal, which initiates an updating of the internal state vector responsible for the future action selection process. A learning automaton environment configuration is shown in Figure 1.

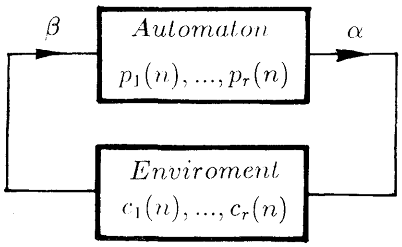

**Fig. 1.** Automaton/Environments interaction.

The automaton is defined by a quadruple $\{\alpha, \beta, p, T\}$ for which $\alpha$ and $\beta$ are the output and input sets, $p$ is the action probability vector, and $T$the learning algorithm.

The output set $\alpha$ is the action set of the automaton $\{\alpha_1, \ldots, \alpha_r\}$, the input set $\beta$ is a continuous random variable in the interval $[0, 1]$ (S-model). By normalizing the queue delay feedback in the network, the input space of an S-automaton and the network feedback can be made to correspond.

The environment is defined by the set

$c = \{c_1, \ldots, c_r\}$ where

$c_i = \mathrm{P}\{\beta(n) = \beta_i \mid \alpha(n) = \alpha_i\}$ , $\beta_i$ a random variable in $[0, 1]$

$n$ is the time of the automaton action

$c_i$ is called the $i$ penalty probability of the environment.

The reinforcement algorithm $T(\alpha, \beta, p)$ provides the necessary means to modify the action probability vector in relation to the performed action and received response.

$$p(n + 1) = T(\alpha, \beta, p(n))$$

The learning algorithm with $r$-actions takes place as follows:

*Reward on* $\alpha_i$

$$\left.\begin{aligned} p_i(n + 1) &= p_i(n) + a(1 - \beta(n))\, p_i(n) \\ p_j(n + 1) &= p_j(n) - a(1 - \beta(n))\, p_j(n) \end{aligned}\right\} \beta(n) = 0$$

*Penalty on* $\alpha_i$

$$\left.\begin{aligned} p_i(n + 1) &= p_i(n) - b\, \beta(n)\, p_i(n) \\ p_j(n + 1) &= p_j(n) + b\, \beta(n)\, [1/(r - 1) - p_j(n)] \end{aligned}\right\} \beta(n) = 1 \qquad (1)$$

The constants $a$ and $b$ are the reward and penalty parameters, $0 < a, b < 1$. The reward to the selection of the action $\alpha_i$ is largest when $\beta$ is close to 0, and the converse is true for $\beta$ close to 1.

We shall refer to the case when $a = b$ as the $\mathrm{SL}_{\mathrm{RP}}$ algorithm (linear reward-penalty).

The linear reward-inaction, $\mathrm{SL}_{\mathrm{RI}}$, algorithm is given by (1) when $b = 0$.

The linear reward-ε-penalty, $\mathrm{SL}_{\mathrm{R\varepsilon P}}$, algorithm is given by (1) when $b = O(a)$ $O(a)$ is such that $O(a)/a \to 0$ as $a \to 0$.

A non-stationary environment model with a continuous functional dependence on the action probabilities of the automaton is presented for the real network representation. The parameters of the environment correspond to normalized performance variables that are monitored in a network. To account for both equilibrium and transient behaviour in the environment, a fully dynamic model is introduced.

$$c_i(n + 1) = k_i\, c_i(n) + (1 - k_i)\, \Phi_i(p_i(n)) \tag{2}$$

Each of the environment parameters is now a state variable so the system state consists of the automaton strategy augmented with the $r$-vector of the environment. We assume $0 < k < 1$. So the environment model is stable.

$\Phi_i(\cdot)$ is defined to have the properties

(i) $\Phi_i(p_i)$ is defined everywhere on $[0, 1]$ and
$\Phi_i$: $[0, 1] \to [0, 1]$ with $\Phi_i(1) = 1$

(ii) $\Phi_i(p_i)$ is analytic of every point of $[0, 1]$ and

(iii) $\Phi_i(p_i)$ is monotonically increasing. (3)

Let us denote the network's queue delay at trial $n$ for the automaton's $i$th link choice by $T_i(n)$. The delay is positive and, in a stable situation, finite. For $T_i(n) \in [\mu, \nu]$ where $\mu, \nu \in \mathbb{N}$ define the normalized queue delay, corresponding to the outputs of the abstract environment models, by

$$c_i(n) = \frac{T_i(n) - \mu}{\nu - \mu} \tag{4}$$

$c_i(n)$ is a random variable in the interval $[0, 1]$, and the normalized delay in the computer network corresponds to the input space of the S-automata described earlier. Hence the automaton in the network is updated with $\beta(n) = c_i(n)$ when $\alpha_i$ ($\mathrm{link}_i$) is chosen.

## 3. ANALYSIS OF THE DYNAMIC ENVIRONMENT

The simple case of a two-action automaton is analyzed in detail and later the results are extended to the more general case in which the automaton has $r > 2$ actions. Under the conditions specified, it is shown that a unique vector $[p^{*\mathrm{T}}, c^{*\mathrm{T}}]^{\mathrm{T}}$ exists.

Let $x$ be the state at the learning model whose state space is $X$. $X$ is the $r$-dimensional simplex $s^r$ where

$$s^r \triangleq \{p \mid p = (p_1, p_2, \ldots, p_r) , \quad \sum_{i=1}^{r} p_i = 1 , \quad 0 \leqq p_i \leqq 1\}$$

Let $\mathscr{E}$ be the space of all possible pairs of automaton actions and inputs to the automaton from the environment, and $e(\alpha, \beta)$ be a particular event at a given trial. $T$ is the transition function defined in the previous section and $p(x, e)$ is the probability distribution of $e$ conditioned on $x$.

$T$ acting in a stationary environment is said to be distance diminishing if, for any initial states $x$ and $x'$, $d(T(x), T(x')) < d(x, x')$ where $d(\cdot, \cdot)$ is a suitably chosen metric. The following criterion of Norman [6] is applied to prove the algorithm is distance diminishing. Define

$$l(T) = \frac{d[T(x, e), T(x', e)]}{d(x, x')} \tag{5}$$

$$m(p) = \sup_{x \neq x'} \frac{|p(x, e) - p(x', e)|}{d(x, x')} \tag{6}$$

where $d$ is the metric

$$d(x, x') = \sum_{i=1}^{r} |x_i - x_i'| .$$

The operation of the algorithm is distance diminishing if $m(p) < \infty$ and $l(T) \leqq 1$ for all $e \in \mathscr{E}$ and all $x \in X$ and if there is some $e \in \mathscr{E}$ such that $l(T) < 1$ and $p(x, e) > 0$.

For the non-stationary dynamic environments of this paper, the state of the system has to be suitably defined to include the environment also. The state space in this case is the product space $Y = s^r \times [0, 1]^r$. The state vector of dimension $2r$ is $Y = [p^{\mathrm{T}}, c^{\mathrm{T}}]^{\mathrm{T}}$. We will demonstrate that the $\mathrm{SL_{RP}}$ automaton is distance diminishing. Similar arguments can be applied for the $\mathrm{SL_{RI}}$ and $\mathrm{SL_{R\varepsilon P}}$ automata. It has been proven by Norman that if the process ((1), (2)) is distance diminishing, it is a compact Markov process and converges to one of its ergodic kernels. If there is a single ergodic kernel, the process converges uniquely from any initial state. Under the conditions specified, it is shown that a unique vector, $[p^{*\mathrm{T}}, c^{*\mathrm{T}}]^{\mathrm{T}}$ exists.

## Existence and Uniqueness of $p^*$

The environment for the $r$-action case is given by (2) and (3):

$$c_i(n + 1) = k_i\, c_i(n) + (1 - k_i)\, \Phi_i(p_i(n)) , \quad i = 1, 2, \ldots, r$$

where $\Phi_i$ is analytic and monotonically increasing on $[0, 1]$ and $\Phi_i$: $[0, 1] \to [0, 1]$ with $\Phi_i(1) = 1$.

*Brouwer's Fixed Point Theorem* [8]: Every convex compact subspace of a Banach space is a fixed point space. Using Brouwer's theorem, it is first shown that $2r$ constants $p_i^*$, $c_i^*$, $i = 1, 2, \ldots, r$ exist such that $p_i^* c_i^* = p_j^* c_j^*$ for $i, j = 1, 2, \ldots, r$. Using the properties of $\Phi_i$, it is later shown that the constants $c_i^*$, $p_i^*$, $i = 1, 2, \ldots, r$ are unique.

We define the following functions

$$w_i = -p_i c_i + \frac{1}{r-1} \sum_{j=1}^{r} p_j c_j\,, \quad j \neq i$$

$$\sigma_i = -p_i c_i + p_i \Phi_i(p_i) \tag{7}$$

It is clear that the sums of all $r$ components of the $w_i$ and $\sigma_i$ are 0.

Define a continuous map $A$ taking $s^r \times [0, 1]^r$ into $s^r \times [0, 1]^r$: $A = \begin{pmatrix} M \\ N \end{pmatrix}$ where $M$ and $N$ are defined by

$$M\colon p_i' = \frac{p_i + w_i^+}{1 + \sum_{j=1}^{r} w_j^+}\,, \quad N\colon c_i' = \frac{c_i}{1 + \sum_{j=1}^{r} \sigma_j^+} \tag{8}$$

where $w_i^+ = \max\{w_i, 0\}$ and $\sigma_i^+ = \max\{\sigma_i, 0\}$. It is clear that $s^r \times [0, 1]^r$ is convex.

There must be some $w_i^+ = 0$, and by the fixed point theorem and (8),

$$p_i \sum_{j=1}^{r} w_j^+ = 0 \quad \text{and}$$

$$c_i \sum_{j=1}^{r} \sigma_j^+ = 0 \tag{9}$$

These equations imply $p_i^* c_i^* = p_j^* c_j^*$, $i \neq j$. Uniqueness of the equilibrium follows by assuming the existence of two equilibrium points $y^*$ and $y^{*\prime}$ and proving a contradiction using the continuity and monotonicity of $\Phi_i$.

## Distance Diminishing Property of the $SL_{RP}$ Automaton

### I. Two-action case

For the two action case, the following equations describe the operation of the automaton into the environment:

$$p_i(n+1) = p_i(n) - a[1 - c_j(n)]\, p_i(n) + a\, c_j(n)\,[1 - p_i(n)]\,, \quad \alpha(n) = \alpha_j$$

$$p_i(n+1) = p_i(n) + a[1 - c_i(n)]\,[1 - p_i(n)] - a\, c_i(n)\, p_i(n)\,, \quad \alpha(n) = \alpha_i \tag{10}$$

The constant $a$ is the learning parameter, $0 < a < 1$

$$c_1(n+1) = k_1\, c_1(n) + (1 - k_1)\, \Phi_1(p_1(n))$$

$$c_2(n+1) = k_2\, c_2(n) + (1 - k_2)\, \Phi_2(p_2(n)) \tag{11}$$

where $i, j \in \{1, 2\}$ and $i \neq j$. The minimal state description is given by $(p_1, c_1, c_2)^T$ since $p_2 = 1 - p_1$, $k_1$, $k_2$, $\Phi_1$ and $\Phi_2$ are unknown and depend on the network.

When an automaton operates in a random environment, distance diminishing behaviour must be established using the entire state $y$. However, because the dependence of $c_1(n+1)$ and $c_2(n+1)$ on $p_1(n)$ is deterministic and the randomness is introduced only in the equations for $p_1$ we can focus on the convergence behaviour in the simplex $S$ determined by the action probabilities of the automaton and show that $p_1$ converges in distribution. Once this is proven, the convergence of $c_i$ follows.

Let $\hat{p}_1(n+1)$, $\hat{c}_1(n+1)$ and $\hat{c}_2(n+1)$ be defined from (10) and (11) in terms of states $(p_1(n), c_1(n), c_2(n))^T$ and $(p'_1(n), c'_1(n), c'_2(n))^T$ as

$$\hat{p}_1(n+1) = p_1(n+1) - p'_1(n+1) = (1-a)\,\hat{p}_1(n) - a\,\hat{c}_1(n)\,,$$
$$\alpha(n) = \alpha_1$$
$$\hat{p}_1(n+1) = p_1(n+1) - p'_1(n+1) = (1-a)\,\hat{p}_1(n) + a\,\hat{c}_2(n)\,,$$
$$\alpha(n) = \alpha_2 \tag{12}$$
$$\hat{c}_i(n+1) = k_i\,\hat{c}_i(n) + (1-k_i)\{\Phi_i(p_i(n)) - \Phi_i(p'_i(n))\}\,,\quad i \in \{1,2\} \tag{13}$$

For the distance diminishing property the equation (5) must be upper bounded by 1. We also consider a lower bound $m < l(T)$, $0 < m < 1$.

$$m < \frac{|\hat{p}_i(n+1)|}{|\hat{p}_i(n)|} = |(1-a)\,\hat{p}_i(n) - ak_i\,\hat{c}_i(n-1) - a(1-k)\,.$$
$$.\,[\Phi(p_i(n-1)) - \Phi(p'_i(n-1)]|\;|\hat{p}_i(n)| < 1 \tag{14}$$

To eliminate the dependence on $\hat{c}_i(n)$ we make the following substitution

$$\hat{c}_i(n) = k_i^N\,\hat{c}_i(n-N) + (1-k_i)\sum_{j=1}^{N}\{\Phi(p_i(n-j)) - \Phi(p'_i(n-j))\}\,k_i^{j-1} \tag{15}$$

where $k_i^N\,\hat{c}_i(n-N)$ can be made less than any positive constant $\delta_i$ by a suitable choice of $N$.

Substituting in (14) and multiplying by $|\hat{p}_i(n)|$ we have:

$$m|\hat{p}_i(n)| < |\hat{p}_i(n+1)| = |(1-a)\,\hat{p}_i(n) - ak_i^N\,\hat{c}_i(n-N) -$$
$$- a(1-k_i)\sum_{j=1}^{N}[\Phi(p_i(n-j)) - \Phi(p'_i(n-j))]\,k_i^{j-1}| < |\hat{p}_i(n)|$$

We assume $N$ sufficiently large so the term $k_i^N$ is negligible. So

$$\sum_{j=1}^{N}\frac{[\Phi(p_i(n-j)) - \Phi(p'_i(n-j))]\,k_i^{j-1}}{\hat{p}_i(n)} < \frac{1-a-m}{a(1-k_i)} \tag{16}$$

If the maximum slope of $\Phi_i$ $(i = 1, 2)$ is $\xi$, it follows that

$$\Phi(p_i(n-h)) - \Phi(p'_i(n-j)) < \Phi(p_i(n)) - \Phi(p'_i(n)) - \xi\,\Delta\hat{p}_i(n-j) \tag{17}$$

where

$$\Delta\hat{p}_i(n-j) = \hat{p}_i(n) - \hat{p}_i(n-j)\,.$$

From (15) we have

$$\sum_{j=1}^{N} k_i^{j-1} \frac{\Phi(p_i(n-j)) - \Phi(p_i'(n-j))}{\hat{p}_i(n)} - \xi \frac{\Delta \hat{p}_i(n-j)}{\hat{p}_i(n)} < \frac{1-a-m}{a(1-k_i)} \qquad (18)$$

Applying $t$ times the lower bound $m < \hat{p}_i(n+1)/\hat{p}_i(n)$ we have $\hat{p}_i(n)/\hat{p}_i(n-t) > m^t$ and

$$\begin{aligned} \Delta \hat{p}_i(n-t) = \hat{p}_i(n) - \hat{p}_i(n-t) &> -(1-m^t)\, \hat{p}_i(n-t)\,, \quad p > 0 \\ &< -(1-m^t)\, \hat{p}_i(n-t)\,, \quad p < 0 \end{aligned}$$

The inequality (17) can be written as

$$\sum_{j=1}^{N+1} k_i^{j-1} \left[ \xi + \xi \frac{1-m^j}{m^j} \right] < \frac{1-a-m}{a(1-k_i)}\,, \quad j = 1, 2, \ldots, N+1 \Rightarrow$$

$$\Rightarrow \xi < \frac{1-a-m}{a(1-k_i)} \frac{m-k_i}{1-\left(\frac{k_i}{m}\right)^N}$$

We assume $k_i < m$ and $N$ sufficiently large so that $(k_i/m)^N \ll 1$. So

$$\xi < \frac{1-a-m}{a(1-k_i)} (m-k_i) \qquad (19)$$

For the value of $m$, $m = (1 + k_i - a)/2$, (18) is written

$$\xi < \frac{[1-a-k_i]^2}{4a(1-k_i)} \qquad (20)$$

Hence our algorithm can be made distance diminishing for any $\xi$ if $a$ is sufficiently small. The choice of $a$ is seen to depend both on the maximum slope of $\Phi_i$ as well as the parameter $k_i$ which determines the dynamics of the environment.

To examine the distribution of the learning model in the equilibrium, we consider the asymptotic mean learning behaviour. The distribution of $p$ specifies the distribution of $c$ through (2). For the $SL_{RP}$ automaton, let $p$ and $c$ be given, and define

$$w(p_1, c) = \mathsf{E} \left[ \begin{array}{c|c} \dfrac{\Delta p_1(n)}{a} & p(n) = p \\ \Delta c & c(n) = c \end{array} \right].$$

$$\begin{bmatrix} c_2(n)\, p_2(n) - c_1(n)\, p_1(n) \\ (1-k_1)\,[\Phi_1(p_1(n)) - c_1(n)] \\ (1-k_2)\,[\Phi_2(p_2(n)) - c_2(n)] \end{bmatrix} \qquad (21)$$

The solution of (21) (the zero of $w$) is at $p^*$, $c^*$. Using the results from Norman [6] we can show that $p^*$ and the mean of the distribution of $p$ are within $O(a)$ of each other. $\{p(n)\}_{n \geq 0}$ changes by small steps, and $p^*$ is approached is the argument of $w$ is continuous. If $\tau$ is the solution of the differential equation $\dot{\tau} = w(\tau)$, then $\tau = (p^*, c^*)$ is a stable equilibrium. In general the zero of $\mathsf{E}[w(p, c)]$ does not coincide with the zero $(p^{*T}, c^{*T})^T$ of $w(\tau)$. But it can be shown that the solution is within $O(a)$ of $p^*$.

Define

$$z(k) = \frac{p(k) - \tau(k)}{\sqrt{a}}$$

If $a \to 0$, $ka \to t < \infty$, then $z(k)$ converges in distribution to $N[0, \sigma(t)]$, where $\sigma(t) = O(a)$. Hence if $a$ is small, the equilibrium strategy is approximated by $p^*$.

## II. $r$-Action Automaton

The proof of convergence for the $r$-action automaton is parallel to that of two-action automaton. To prove $l(T) < 1$,

$$m < \frac{d[T(p(n)), T(p'(n))]}{d[p(n), p'(n)]} =$$

$$= \Big\{ |(1 - a)[p_i(n) - p_i'(n)] - a[c_i(n) - c_i'(n)]| +$$

$$+ \sum_{j \neq i}^{r} \left| (1 - a)[p_j(n) - p_j'(n)] + \frac{a}{r - 1}[c_i(n) - c_i'(n)] \right| \Big\} \div$$

$$\div \sum_{j=1}^{r} |p_j(n) - p_j'(n)| < 1 \qquad (22)$$

As in the two-action case, applying equation (15) for large $N$ we get the bound:

$$\sum_{j=1}^{N} \frac{[\Phi_i(p_i(n - j)) - \Phi_i(p_i'(n - j))]\gamma_i^{j-1}}{(p_i(n) - p_i'(n))} < \frac{1 - a - m}{2a(1 - k_i)} \qquad (23)$$

(23) is identical to (16) except for a factor of $\frac{1}{2}$.

Following the same steps as in previous case we get as upper bound

$$\xi < \frac{[1 - a - k_i]^2}{8a(1 - k_i)} \qquad (24)$$

So, the operation of the $r$-action algorithm is distance diminishing under the same conditions as given for the two-action case.

If we consider $w(p_i, c)$ as defined in (21) we have:

$$\mathrm{E}[\Delta p_i(n) \mid p(n) = p] = -ap_i\beta_i + (a - b)\, p_i\beta_i[1 - \sum_{j \neq i} p_j] + \frac{b}{r - 1} \cdot$$

$$\cdot \sum_{j \neq i} p_j\beta_j + (a - b)\, p_i \sum_{j \neq i} p_j\beta_j \,.$$

In the $SL_{RP}$ algorithm $a = b$ and $p^*$ is given by $p_i^* c_i^* = p_j^* c_j^*$, $i, j = 1, 2, \ldots, r$.

As previously $p(n)$ converges to a normal distribution with variance $O(a)$. The mean of the distribution is within $O(a)$ of $p^*$ and consequently if $a$ is small the routing strategy is approximated by $p^*$.

## 4. SIMULATIONS

The consistency of the simulated behaviour of automata operating in the network with the behaviour predicted by the models it is shown. The need for a fully dynamic environment is pointed out, and the use of this model to predict transient behaviour is demonstrated. The asymptotic behaviour of the $SL_{RP}$, $SL_{RI}$, $SL_{R\varepsilon P}$ algorithms in (5) and (6) is verified, with the forcing function of the environment chosen to be $\Phi_i(p_i) = p_i$. All the simulations are performed in the 4-node network of Figure 2.

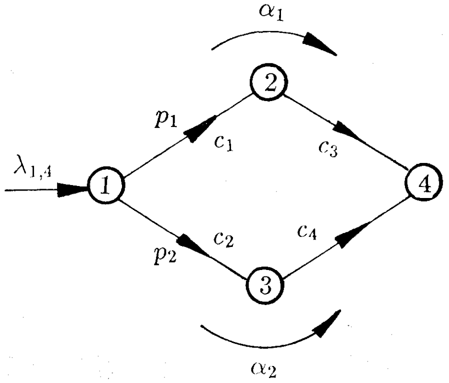

**Fig. 2.** $c_1 = c_2 = c_3 = c_4$ 8000 bits/sec; $\lambda_{1,4}$ variable in packets/byte-time; packet length = 256 bits, byte = 8 bits.

The transient behaviour of the $SL_{RP}$ automaton in the dynamic environment is shown in Figure 3.

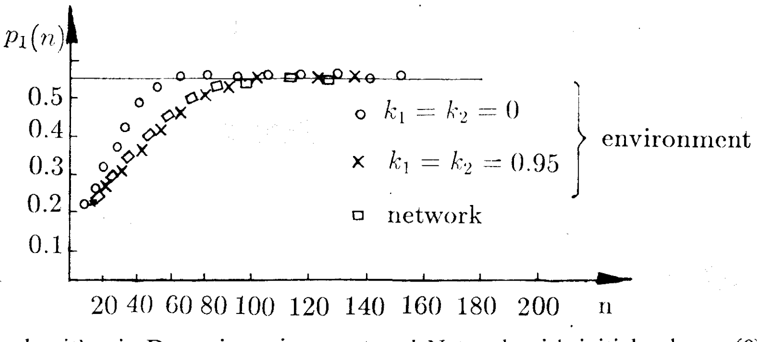

**Fig. 3.** $SL_{RP}$ algorithm in Dynamic environment and Network with initial values $c_1(0) = 0{\cdot}5$, $c_2(0) = 0{\cdot}5$ learning constants $a = b = 0{\cdot}02$ and traffic $\lambda_{1,4} = 0{\cdot}2$ packets/byte-time.

By varying the time constants of the dynamic environment model (2) it is seen in Figure 3 that the transient behaviour of the automaton in the dynamic environment and network nearly match each other when $k_1 = k_2 = 0{\cdot}95$.

The load equalizing behaviour of the $SL_{RP}$, $SL_{RI}$, $SL_{R\varepsilon P}$ algorithms and their comparison with other routing algorithms such as random routing and minimum delay routing is shown in Table 1.

**Table 1.** Performance of the Learning Algorithms Compared to Random and Shortest Queue Algorithms. 1000 packets of length 256 bits have been transmitted in the network. Learning constants $a = b = 0{\cdot}02$ for $SL_{RP}$, $a = 0{\cdot}02$ for $SL_{RI}$ and $a = 0{\cdot}02$, $b = 0{\cdot}002$ for $SL_{R\varepsilon P}$, are experimentally chosen. Perfect information (downchain queue delay) is available to all algorithms.

| | | Delay/path | | Product of Delay-Probability | | Average Delay |
|---|---|---|---|---|---|---|
| Algorithm | Load $\lambda_{1,4}$ | $\alpha_1$ | $\alpha_2$ | $\alpha_1$ | $\alpha_2$ | |
| **$SL_{RP}$** | 0·2 | 1·137 | 1·063 | 0·545 | 0·552 | 1·097 |
| **$SL_{RI}$** | 0·2 | 1·005 | 1·027 | 0·421 | 0·592 | 1·013 |
| $SL_{R\varepsilon P}$ | 0·2 | 1·098 | 1·057 | 0·385 | 0·671 | 1·056 |
| Random | 0·2 | 1·221 | 1·209 | 0·610 | 0·604 | 1·215 |
| Shortest Queue | 0·2 | 0·993 | 1·008 | 0·993 | 1·008 | 1·005 |

From Table 1 it is clear that learning routing is better than random routing and that minimum delay (shortest queue) is the lower bound on the packet delay.

The effect of changes in the network parameters has been considered, by increasing the sampling interval and introducing a transportation lag in the feedback information available to the algorithms.

Thus the performance of minimum delay routing deteriorates as the sampling interval increases and a transport lag is introduced. On the contrary learning algorithms can easily adapt to the changes in the network conditions as is shown in Table 2.

**Table 2.** The effect of Sampling Interval and Transport Lag on the Performance of Minimum Delay and Learning Algorithms. The load $\lambda_{1,4} = 0{\cdot}2$ packets/byte-time and the sampling interval in increased from 0·02 sec to 1 sec.

| Algorithm | Average Before | Delay After |
|---|---|---|
| $SL_{RP}$ | 1·097 | 1·184 |
| Shortest Queue | 1·005 | 1·521 |

Now we illustrate the need of a fully dynamic environment. The instantaneous environment $c_i(n) = \Phi_i\, p_i(n)$, suggested in [9], is memoryless and consequently $p_1(n_1) = p_1(n_2)$ at trial $n_1$, $n_2$ implies that $c_1(n_1) = c_1(n_2)$.

Transient peaks of $p_1(n)$ were generated with two different source rates in Figure 2. Equal values of $p_1(n)$ have been observed and the corresponding values of $c_1(n)$

were found to be significantly different. The numerical values at these points are given in Table 3. It is clear that the network is not memoryless.

By varying the time constants of the dynamic environment model (2) it is seen

**Table 3.**

| $\lambda_{13}$ | $n$ | $p_1(n)$ | $c_1(n)$ |
|---|---|---|---|
| | 70 | 0·463 | 0·440 |
| 0·2 | 120 | 0·461 | 0·531 |
| | 67 | 0·511 | 0·671 |
| 0·4 | 100 | 0·512 | 0·823 |

that the transient behaviour of the automaton in the dynamic environment and network nearly matching when $k_1 = k_2 = 0{\cdot}95$.

## 5. CONCLUSIONS

The principal contribution of this paper is the introduction of a dynamical environment model presenting the characteristics of a network and the applicability of learning methodology to the data network routing problem. Simulations have shown that the dynamic environment can be used to study the behaviour of learning algorithms in data networks. The superiority of learning algorithms over existing algorithms when network conditions are changed is shown.

This work indicates that the use of learning automata is a realistic and useful approach to routing in data communication networks.

The performance of learning algorithms with different updating methods, feedback policies and its advantages over other dynamic routing techniques is a problem for further investigation.

(Received April 26, 1988.)

## REFERENCES

[1] R. G. Gallager: A minimum delay routing algorithm using distributed computation. IEEE Trans. Comm. *25* (1977), 73–84.

[2] L. Kleinrock: Queueing Systems: II-Applications. J. Wiley, New York 1976.

[3] K. S. Narendra, L. G. Mason and S. S. Tripathi: Application of learning automata to telephone traffic routing problems. Preprints of 1974 Joint Automatic Control Conference, Austin, TX, 18–21 June 1974, pp. 753–759.

[4] K. S. Narendra and M. A. L. Thathachar: Learning automata: a survey. IEEE Trans. Systems Man Cybernet. *4* (1974), 323–334.

[5] K. S. Narendra, E. A. Wright and L. G. Mason: Application of learning automata to telephone traffic routing and control. IEEE Trans. Systems Man Cybernet. *7* (1977), 11, 785–792.

[6] M. F. Norman: Markov Processes and Learning Models. Academic Press, New York—London 1972.
[7] A. S. Tanenbaum: Computer Networks. Prentice-Hall, Englewood Cliffs, N. J. 1980.
[8] R. Larsen: Functional Analysis. Marcel-Dekker, New York 1973.
[9] V. A. Vasilakos: Learning algorithm in non-stationary environments, with application to data networks. Working Paper, Univ. of Patras.
[10] V. A. Vasilakos and S. A. Koubias: Learning automata control of data communication networks. In: Proc. 12th IMACS World Congress on Scientific Computation, Paris, July 1988.
[11] V. A. Vasilakos and S. A. Koubias: On routing and performance comparison of techniques for packet-switched networks using learning automata. In: Proc. IEEE Internat. Symposium on Circuits and Systems, Espoo, Finland, June 1988.

*Dr. Athanasios V. Vasilakos, Department of Computer Engineering and C.T. I., University of Patras, 26500 Patras. Greece.*