Dynamic Behavior of Shortest Path Routing Algorithms for Communication Networks

DIMITRI P. BERTSEKAS, SENIOR MEMBER, IEEE

Abstract—Several proposed routing algorithms for store and forward communication networks, including one currently in operation in the ARPANET, route messages along shortest paths computed by using some set of link lengths. When these lengths depend on current traffic conditions as they must in an adaptive algorithm, dynamic behavior questions such as stability, convergence, and speed of convergence are of interest. This paper is the first attempt to analyze systematically these issues. It is shown that minimum queuing delay path algorithms tend to exhibit violent oscillatory behavior in the absence of a damping mechanism. The oscillations can be damped by means of several types of schemes two of which are analyzed in this paper. In the first scheme a constant bias is added to the queuing delay thereby providing a preference towards paths with a small number of links. In the second scheme the effects of several past routings are averaged as for example when the link lengths are computed and communicated asynchronously throughout the network.

I. Introduction

A CENTRAL operational problem of a communication network involves the choice of routes used by messages to travel from origin to destination. It is possible, of course, to choose a fixed route for each origin–destination pair, but this precludes the possibility of adjusting routes to alleviate congestion due to variations in average traffic conditions. For this reason attention has focused on adaptive routing strategies whereby congestion in the network is continuously monitored and routes between origin–destination pairs are modified in real time so as to keep average delay per message at a reasonable level. A routing scheme of this type was implemented in the ARPANET in 1969 and attracted considerable attention. The main idea in this scheme is to compute in real time an estimate of the minimum average delay per message for each origin–destination pair and to route messages along the current minimum estimated delay path. When this scheme was first implemented, it was noticed that it was prone to severe oscillations. This behavior is due to the fact that delay estimates used to choose routes are themselves affected by the route choice with a feedback effect resulting. To remedy this situation it was decided on heuristic grounds to introduce an additive factor, called bias, to the estimated delay of each link, thereby building into the algorithm a preference towards paths with small number of hops to the destination [5]–[7]. This had a stabilizing effect albeit at the expense of considerable loss of sensitivity to traffic congestion.

The implementation of the minimum delay path idea in the original ARPANET algorithm had a number of flaws allowing, for example, the formation of loops. For this reason alternative schemes based on the same idea were studied, and a new algorithm called SPF has been developed and implemented [1], [4], [11]. The present paper is an outgrowth of the author's participation in the design study of this algorithm during the summer of 1978 at Bolt Beranek and Newman (BBN), Inc. However, our analysis does not focus on the ARPANET and the SPF algorithm in particular, but rather is geared towards understanding the effect of feedback and the nature of the dynamic behavior of shortest path algorithms where link lengths depend on current traffic conditions. We note that the algorithms of this paper are far from optimal since they are single path algorithms in the sense that at any given time there is only one path per origin–destination pair along which messages can travel. Better performance can be achieved by allowing multiple paths as for example in the optimization algorithm of Gallager [9] or its second derivative versions [2], [12]. We note also that optimal routing algorithms based on shortest path generation have been given recently in [13]. On the other hand, the hardware limitations of some of the presently existing networks including the ARPANET preclude the use of such more sophisticated algorithms. Furthermore, we feel that the mere fact that the algorithm has been successfully implemented in a network as interesting and influential as the ARPANET makes it worthy of analysis and investigation. This is reinforced by the fact that the behavior exhibited by the algorithm is quite interesting and can pose nontrivial design problems.

The paper is organized as follows.

Manuscript received September 19, 1980; revised January 28, 1981. Paper recommended by L. G. Shaw, Past Chairman of the Networks and Transportation Systems Committee. This work was supported by ARPA under Contract MDA 903-78-C-0129. The research was completed at the University of Illinois, Urbana, with the support of the National Science Foundation under Grant NSF-ENG-77-15949, and the Massachusetts Institute of Technology, Cambridge, with the support of the Office of Naval Research under Grant ONR-N00014-75-C-1183.

The author is with the Department of Electrical Engineering and Computer Science, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139.

In Section II, we provide a deterministic finite-state Markov chain framework for studying a simple version of the algorithm. We show that for ring networks the algorithm may tend to oscillate between poor routing paths and become itself a major contributor to congestion. We also demonstrate how the use of a bias factor can provide a mechanism for damping oscillations as confirmed by experience with the original ARPANET algorithm.

The finite-state model does not lend itself to analysis of more sophisticated routing schemes and more general network topologies. We consequently introduce in Section III a model of a ring network with a continuum of nodes and a single destination. This allows us to employ techniques of stability analysis of discrete-time systems with continuous state space, and enables us to further quantify the relationship between choice of link lengths and algorithmic behavior.

The analysis of Section III focuses primarily on the effect of using a bias factor as a damping mechanism. In Section IV we show that oscillations can also be damped effectively by making the link lengths dependent on several preceding routing paths via some averaging mechanism such as an exponential fading memory scheme or asynchronous link length updating. To our knowledge the fact that averaging can provide a damping mechanism in a shortest path algorithm has not been noticed earlier and in fact when we originally approached this problem at BBN there was considerable concern regarding its effect on algorithmic behavior. It is now believed that the significant degree of averaging inherently present in the SPF algorithm is in large measure responsible for the stable dynamic behavior observed in experiments conducted thus far [11].

The analysis of Sections II–IV focuses on ring networks. The ring topology is central for the extension of our earlier results to more complex network topologies. This extension is carried out in Section V under the assumption that an equilibrium routing exists. However, by contrast with ring networks, an equilibrium routing need not always exist for more complex topologies. We demonstrate via example the mechanism by which such a phenomenon can occur.

The results and analysis of the present paper can be generalized to the case where there are more than one destinations. This analysis is straightforward but considerably more complex technically and may be found in [3]. The continuous node model of Sections III–V may be criticized on the grounds that it is unrealistic. On the other hand, it is very difficult to provide an extensive analysis of a more realistic finite node network model. In particular, it appears impossible to demonstrate the effect of averaging in such a context. Furthermore, we believe that the realism of any algorithmic model must be judged on the basis of the validity of the conclusions it provides regarding the behavior of the related practical algorithm. These conclusions in our case have been verified by extensive numerical experiments with finite-node networks [3], [4]. In particular, the validity of our qualitative results regarding the role of a bias factor and averaging as damping mechanisms have been amply demonstrated.

II. A Finite-State Markov Chain Model

Consider a communication network with modes denoted by $1, 2, \cdots, N$ and directed links denoted by (i, l) where i is the head node and l is the tail node. We consider the following algorithms for periodically updating paths for routing messages.

(A) At the beginning of every time period a nonnegative length D_{il} of every link (i, l) becomes available to each node. Based on these lengths each node computes a shortest path to each destination and routes messages over that path during the period.

The standing assumption for algorithm (A) is that the lengths D_{il} used in computation of a new shortest path depend exclusively on one or more preceding shortest paths. This dependence is *deterministic* via a rule that for the moment we leave unspecified. As an example D_{il} may represent some measure of *average* delay per message on link (i, l) during one or more preceding periods perhaps with an added bias factor—a scheme currently implemented in the ARPANET [1], [11]. By assuming that the dependence of D_{il} on previous shortest paths is deterministic *we also implicitly assume that the input traffic originating at each node is a stationary stochastic process whose ensemble parameters can be adequately measured by time averages.* This assumption is not valid, of course, in practice but is a reasonable approximation to the situation where the time constant of traffic statistic variations is large relative to the shortest path updating period (a *quasi-static* assumption, cf. [9]).

Consider first algorithm (A) applied to a given network for the case where the lengths D_{il} depend exclusively on the preceding shortest path. Assume also that the shortest path algorithm has a fixed rule for breaking ties between equidistant paths. Then each shortest path uniquely determines the next shortest path. There is a finite number of possible shortest paths (also referred to as *routings*) which we denote by $R_1, R_2, \cdots, R_M$ where M is some integer. To any initial routing say R_{i_0}, there corresponds a unique sequence of subsequent routings $R_{i_1}, R_{i_2}, \cdots$. Thus, eventually some routing will be repeated (say $R_{i_k} = R_{i_{k-n}}$), and once this happens the routing sequence will become periodic. Thus, starting at R_{i_0} the algorithm will eventually end up cycling through $R_{i_k}, \cdots, R_{i_{k+n-1}}$. Of course it is possible that R_{i_0} itself is part of the cycle $(k = 0)$, and that the cycle consists of a single routing $(n = 1)$ in which case the algorithm stabilizes at the routing.

The model just described is one of a deterministic finite-state Markov chain with states $R_1, \cdots, R_M$. From Markov chain theory or by elementary reasoning it follows that the set of all routings $\{R_1, \cdots, R_M\}$ can be partitioned into a collection of cycles (or ergodic classes), and a collection of transient routings. If the initial routing is transient it is

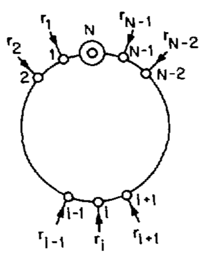


Fig. 1.

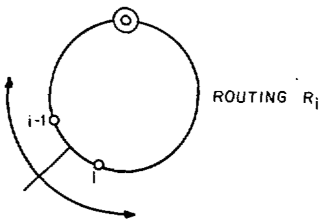


Fig. 2.

never repeated by the algorithm, and if it is part of a cycle the algorithm returns to it periodically. More than one cycles may exist. Furthermore, each transient routing leads to a unique cycle.

When the lengths D_{il} depend on a fixed number (say m) of preceding routings, a finite state model for the algorithm can be similarly constructed whereby the state space of the model is the set of all m-tuples of routings. Similarly, the state space can be partitioned into cycles and transient states. Analysis of such a model is naturally more difficult in view of the increased size of the state space, and this is more so if D_{il} depends on all preceding routings in which case a countable state Markov chain model is necessary.

In what follows in this section we will restrict attention to the case of a ring network with N nodes shown in Fig. 1. Node N is the only destination and all links are bidirectional. By reversing the directions of flow and the role of origins and destinations the subsequent model can be converted to one with a single origin and many destinations. It is recognized that few practical networks can be expected to have a ring topology. However, this topology is not only analytically tractable, but also provides a fundamental building block for analysis of more general topologies as will be seen in Section V. The traffic input originating at node i and destined for N is denoted by r_i. The *routing* R_i, $i=1,\cdots,N$ is the one for which all nodes $j<i$ route their traffic in the clockwise direction and all nodes $j\geq i$ route their traffic in the counterclockwise direction as shown in Fig. 2. Given a routing R_i, the flows on each undirected link $(j-1,j)$ in the clockwise and counterclockwise direction are denoted by $f_j^-(i)$ and $f_j^+(i)$ respectively and are given by

$$f_j^-(i)=\begin{cases}0, & \text{if } i\leq j\\ r_{i-1}+r_{i-2}+\cdots+r_j & \text{if } j<i\end{cases}$$

$$f_j^+(i)=\begin{cases}r_i+r_{i+1}+\cdots+r_{j-1}, & \text{if } i<j\\ 0, & \text{if } j\leq i.\end{cases}$$

We will consider the case where the length D_{il} of a link (i,l) is given by an equation of the form

$$D_{il}=d(f_{il}) \tag{1}$$

where f_{il} is the flow on link (i,l) during the preceding period and d is a real valued, continuously differentiable and monotonically increasing function of flow with $d(0)\geq 0$. For simplicity we assume that the function d is the same for all links but this does not affect materially the analysis that follows. Since the flow f_{il} depends only on the preceding routing the same is true for the length D_{il}. It appears that this simplest of all possible situations is the only one that can be analyzed effectively in a finite-node network context. The practical situation where D_{il} is taken to be the average time delay for a message to traverse link (i,l) can be reasonably modelled by a function d of the form

$$d(f_{il})=P_{il}+T_{il}+Q_{il}(f_{il}) \tag{2}$$

where

P_{il}	average processing plus propagation delay per message,
T_{il}	average transmission delay per message,
$Q_{il}(f_{il})$	average queuing delay per message when the average flow on link (i,l) is f_{il}.

The quantities P_{il} and T_{il} are independent of the flow f_{il} while the dependence of Q_{il} on f_{il} is determined by the statistics of the traffic arriving at i and routed through l. If these statistics can be adequately modeled by an $M/M/1$ queue then Q_{il} takes the form [6], [7]

$$Q_{il}(f_{il})=\frac{f_{il}}{C_{il}-f_{il}} \tag{3}$$

where C_{il} is the transmission capacity of link (i,l). Even though this function is convex and monotonically increasing only on the interval $[0,C_{il})$ rather than the entire real line, our subsequent results apply to it assuming that the average flow in the link lies within $[0,C_{il})$ as it will in a practical network in view of flow-control restrictions. We mention, however, that on the basis of experiments conducted thus far it is unclear whether the average delay per message in the ARPANET can indeed be modelled as in (2). This may be due to peculiarities of the ARPANET hardware and software which are little understood at present.

We now define the shortest path algorithm. Given a routing R_i we define the distances $D_j^-(i)$, $D_j^+(i)$ of node j to the destination in the counterclockwise and clockwise directions respectively by

$$D_j^-(i)=\sum_{l=1}^{j} d[f_l^-(i)]$$

$$D_j^+(i)=\sum_{l=j+1}^{N} d[f_l^+(i)].$$

If $D_i^-(i)=D_i^+(i)$ then the algorithm leaves the routing unchanged at R_i. If $D_i^-(i)\neq D^+(i)$ the algorithm sets the next routing to R_n where the node n is such that

$$D_j^-(i)\geq D_j^+(i)\quad \text{for } j\geq n$$
$$D_j^-(i)< D_j^+(i)\quad \text{for } j< n.$$

It can be easily shown that the next routing is uniquely determined by the relations above. Given an initial routing R^0 we consider the sequence of successive routings $R^1, R^2,\cdots,R^k, R^{k+1},\cdots$, generated by the algorithm.

We note that this algorithm is not claimed to be optimal, or even good. Rather, it approximates the ARPANET algorithm, and it is applied to the simple ring in order to analyze its properties in a special case which is tractable.

The quantity $d(0)$ may be viewed as a *bias factor*. It represents link length at zero flow. The following proposition shows that if $d(0)=0$ and the first two routings are different, i.e., $R^0\neq R^1$ then the algorithm ends up oscillating between the two extreme routings R_1 and R_N which is the worst possible behavior that can occur. In the context of (2) the case $d(0)=0$ corresponds to the situation where the processing and transmission delays P_{il} and T_{il} are negligible relative to the queuing delay Q_{il}.

Proposition 1: Let $d(0)=0$ and assume that $R^0\neq R^1$. Then there exists an index $\bar{k}$ such that for all $k\geq\bar{k}$ either $R^k=R_1$ and $R^{k+1}=R_N$ or $R^k=R_N$ and $R^{k+1}=R_1$.

Proof: Let R_i be a routing and assume that the routing subsequent to R_i is R_n with $n\neq i$. For concreteness assume that $n<i$. We will show that either $i=N$ or else the routing subsequent to R_n is R_j with $j>i$.

If $i\neq N$ then since R_n is the routing subsequent to R_i we have

$$D_{n-1}^-(i)< D_{n-1}^+(i)=D_i^+(i) \tag{4}$$

where the last equality holds because $d(0)=0$ and the links $(n-1,n),\cdots(i-1,i)$ carry no flow when the routing is R_i. We also have

$$D_i^+(i)\leq D_i^+(n) \tag{5}$$
$$D_i^-(n)=D_{n-1}^-(n)\leq D_{n-1}^-(i). \tag{6}$$

From (4)–(6), we have

$$D_i^-(n)\leq D_{n-1}^-(i)< D_i^+(i)\leq D_i^+(n)$$

so finally

$$D_i^-(n)< D_i^+(n).$$

It follows that in the routing R_j which is subsequent to R_n, node i will switch his traffic to the clockwise direction so that $j>i$.

We can show using a very similar argument that if $n>i$ then either $i=1$ or else the routing subsequent to R_n is R_j with $j<i$.

Thus, we have that the number of nodes that lie between two successive routings is increasing at each iteration if none of these routings is R_1 or R_N. On the other hand, if the current routing is R_1 or R_N then the next routing will clearly be R_N or R_1, respectively. This proves the proposition. Q.E.D.

Notice that, if $d(0)=0$, the situation $R^0=R^1$ can only occur if $D_i^-(i)=D_i^+(i)$ where i is the node for which $R^0=R_i$. Thus, if we add any $\epsilon>0$ to any one of the node inputs we will have $R^0\neq R^1$ and the algorithm will again end up oscillating between R_1 and R_N. We provide an example illustrating the result of Proposition 1. Several additional examples involving more general topologies and multiple destinations may be found in [4].

Example: Consider a 16-node ring network where node 16 is the destination. Let $r_i=1$ for $i=1,\cdots,7,9,\cdots,15$ and $r_8=\epsilon\geq 0$. If $\epsilon=0$ and the initial routing is R_8 then by symmetry all subsequent routings equal R_8. If ϵ is very small but positive then for the case where

$$d(f)=f$$

the sequence of generated routings is $R_8, R_{10}, R_3, R_{16}, R_1, R_{16}, R_1,\cdots$. This fact can be verified via a straightforward calculation in Fig. 3 which shows the flow patterns corresponding to successive routings.

We now turn our attention to various notions of equilibria and stability. We say that R_i is an *equilibrium* routing if

$$D_{i-1}^-(i)< D_{i-1}^+(i)\quad\text{and}\quad D_i^+(i)\leq D^-(i).$$

It follows from this definition that R_i is an equilibrium routing if and only if it repeats itself via the shortest path algorithm.

We say that a node i is an *equilibrium node* if

$$D_i^-(i)< D_i^+(i)\quad\text{and}\quad D_i^+(i+1)\leq D_i^-(i+1).$$

In words, a node i is an equilibrium node if, for both cases where the routing is R_i and R_{i-1}, he switches his traffic in the opposite direction at the next routing.

We say that an *equilibrium routing R_i is locally stable* if routing R_{i+1} generates either R_i or R_{i-1} through the algorithm, and routing R_{i-1} generates either R_i or R_{i+1}. We say that an *equilibrium node i is locally stable* if routing R_i generates R_{i+1} via the algorithm, and routing R_{i-1} generates R_i. The definition of local stability is based on the idea that when the algorithm starts "close enough to equilibrium" it should not lead to a "growing" oscillation. The following proposition complements Propositon 1 and suggests that the bias level $d(0)$ should exceed a certain positive value in order for an equilibrium routing or node to be locally stable.

Proposition 2: a) An equilibrium routing R_i is locally stable if

$$d(0)\geq\max\left\{\frac{r_{i-1}}{2}\sum_{\substack{l=1\\ l\neq i}}^{N-1}\hat{m}_l,\ \frac{r_i}{2}\sum_{\substack{l=1\\ l\neq i-1}}^{N-1}\tilde{m}_l\right\}$$

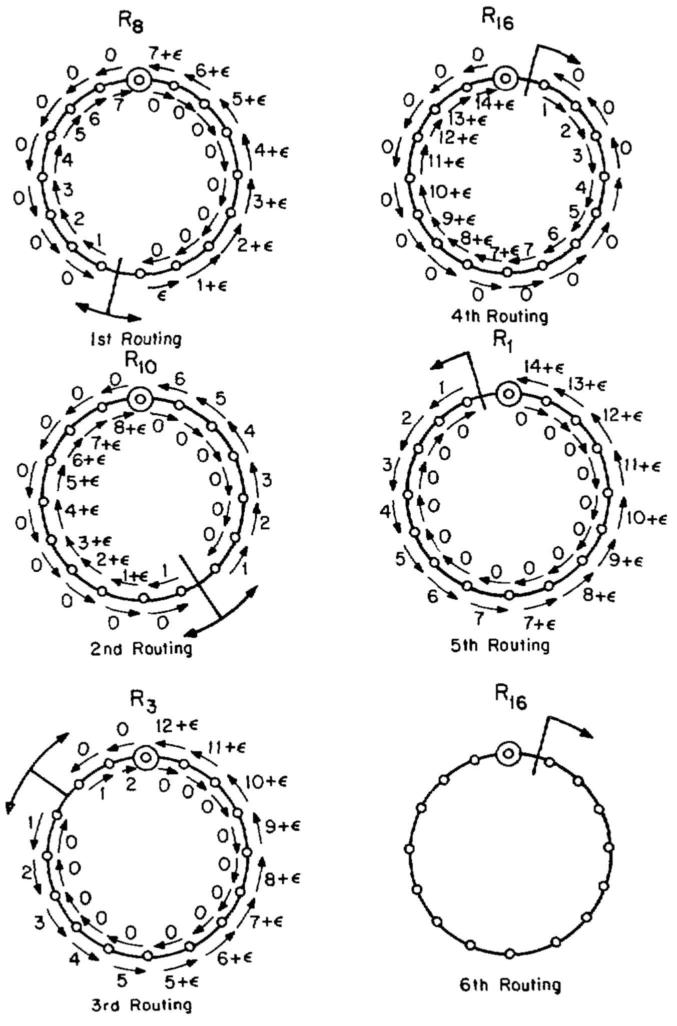


Fig. 3.

where

$$\hat{m}_l = \max\{d'(f) \mid f_l^-(i-1) \leq f \leq f_l^-(i)\}, \quad \text{for } l = 1,\cdots,i-1$$

$$\hat{m}_l = \max\{d'(f) \mid f_{l-1}^-(i) \leq f \leq f_{l-1}^+(i-1)\}, \quad \text{for } l = i+1,\cdots,N-1$$

$$\tilde{m}_l = \max\{d'(f) \mid f_l^-(i) \leq f \leq f_l^-(i+1)\}, \quad \text{for } l = 1,\cdots,i-2$$

$$\tilde{m}_l = \max\{d'(f) \mid f_{l+1}^+(i+1) \leq f \leq f_{l+1}^+(i)\} \quad \text{for } l = i,\cdots,N-1$$

where $d'(f)$ denotes the first derivative of d at f.

b) An equilibrium node i is locally stable if

$$d(0) \geq \frac{r_i}{2} \sum_{l=1}^{N-1} \bar{m}_l$$

where

$$\bar{m}_l = \max\{d'(f) \mid f_l^-(i) \leq f \leq f_l^-(i+1)\}, \quad \text{for } l = 1,\cdots,i$$

$$\bar{m}_l = \max\{d'(f) \mid f_l^+(i+1) \leq f \leq f_l^-(i)\}, \quad \text{for } l = i+1,\cdots,N-1.$$

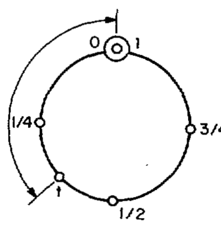


Fig. 4.

The proof of Proposition 2 involves a straightforward but lengthy argument and will be omitted. It can be found in [4].

Proposition 2 implies that in order to ensure local stability the bias $d(0)$ should exceed a level that depends strongly on the traffic conditions. This level is proportional to the input at or near the equilibrium and to a global measure of the derivative d' along the ring. Thus, it may be necessary to choose a value of $d(0)$ which is large relative to r and d' in order to ensure stability for a broad range of input traffic conditions. This can be accomplished by adding a large constant to d. On the other hand, this would introduce a tendency in the algorithm to generate routings close to the min-hop routing (i.e., one that selects routes according to minimum number of links to the destination). As a result, the algorithm would tend to be insensitive to congestion. This tradeoff will be reencountered in the next section.

The point of view that has been adopted in this section is one whereby the algorithm is viewed as a dynamic system with a finite number of states (the finite collection of possible routings). Unfortunately, the study of dynamic behavior and stability properties of such systems is notoriously difficult. To begin with there is no accepted definition of equilibrium, and in fact we saw that in the ring network context there are two types of "equilibria" that are of interest—equilibrium routings and equilibrium nodes. Furthermore, there are no established methodological tools that can be helpful in a finite state system framework. As a result, our progress has been limited to the results just discussed. We are thus motivated to consider approximation of the discrete system with a continuous system having a continuum of states. For such systems there is an effective and well developed stability theory that can be utilized for analysis. We take this approach in the following two sections where we introduce a network with a continuum of nodes. Despite the radical nature of this step the analysis provides informative results and clarifies the role of averaging the effects of several past routings as a means of damping oscillatory behavior. The validity of our approach is supported by the fact that qualitative conclusions drawn from the continuous node model have been verified computationally in finite node models [3], [4].

III. A Continuous Model of a Ring Network

We consider a continuum of nodes arranged in a ring and sending traffic to a single destination as shown in Fig. 4. Points on the ring are identified with their distance t from the destination in the counterclockwise direction, where t is normalized to take values in the interval $[0,1]$. Traffic can move on the ring in both directions.

For every t in $[0,1]$ we denote by $r(t)$ the *input density* at t. The meaning of the function r is that for an subinterval $[t_1, t_2]$ of $[0,1]$ the total input traffic originating at nodes in $[t_1, t_2]$ is

$$\int_{t_1}^{t_2} r(t)\,dt.$$

We assume that r is continuous on $[0,1]$ *and* $r(t)>0$ *for at least one* $t\in(0,1)$. Note that a network with a finite number of nodes can be modeled by a function r containing impulses and such a function can be approximated by a continuous function consisting of narrow triangular pulses of finite height. We are interested in routings specified by points y in $[0,1]$, where the flow splits, i.e., points larger than y send their flow counterclockwise (or in the positive direction) and points smaller than y send their flow clockwise (or in the negative direction). To a given function r and routing y, there corresponds at every point t *a flow in the positive direction* $f^+(y,t)$, and *a flow in the negative* $f^-(y,t)$ given by

$$f^+(y,t)=\begin{cases}\int_y^t r(\tau)\,d\tau, & \text{if } y\le t\\ 0 & \text{if } t\le y\end{cases} \tag{7}$$

$$f^-(y,t)=\begin{cases}0 & \text{if } y\le t\\ \int_t^y r(\tau)\,d\tau & \text{if } t\le y.\end{cases} \tag{8}$$

In order to introduce an algorithm such as (A) in the framework of the continuous model we consider a function d mapping flows into the nonnegative real numbers. The meaning of d is that given a routing y and any point t, *the distances* D^- *and* D^+ *from* t *to the destination in the negative and positive direction* are given by

$$D^-(y,t)=\int_0^t d\left[f^-(y,\tau)\right]d\tau \tag{9}$$

$$D^+(y,t)=\int_t^1 d\left[f^+(y,\tau)\right]d\tau. \tag{10}$$

We will assume that d is a monotonically increasing function of f with everywhere continuous derivative. We further assume that $d(0)>0$. As Proposition 1 shows, the case where $d(0)=0$ is not interesting from a practical point of view.

We consider the following algorithm $(A1)$ for generating routing sequences $\{y_k\}$.

$(A1)$ Given a routing y_k, the next routing y_{k+1} is the solution of the equation

$$D^-(y_k, y_{k+1})=D^+(y_k, y_{k+1}). \tag{11}$$

It will be shown as part of Proposition 3 that (11) has a unique solution for every $y_k\in[0,1]$. Note that since we have

$$D^-(y_k,t)\le D^-(y_k,y_{k+1}) = D^+(y_k,y_{k-1})\le D^+(y_k,t), \quad \text{if } t\le y_{k-1}$$

and

$$D^-(y_k,t)\ge D^-(y_k,y_{k+1}) = D^+(y_k,y_{k+1})\ge D^+(y_k,t), \quad \text{if } t\ge y_{k+1}$$

it follows that a routing y_{k+1} determined from (11) is such that every point t routes its flow in the positive or negative direction according as $D^-(y_k,t)\ge D^+(y_k,t)$ or $D^-(y_k,t)\le D^+(y_k,t)$, i.e., according to minimum distance to the destination.

We say that $y^*\in[0,1]$ is an *equilibrium* if

$$D^-(y^*,y^*)=D^+(y^*,y^*). \tag{12}$$

We first show some preliminary results relating to existence and optimality properties of equilibria.

Proposition 3: There exists a unique equilibrium $y^*\in(0,1)$. Furthermore, (11) has a unique solution y_{k+1} for every y_k.

Proof: Using (9) and (10) we have for all y and t

$$\frac{\partial D^-(y,t)}{\partial t}=d\left[f^-(y,t)\right], \quad \frac{\partial D^+(y,t)}{\partial t}=-d\left[f^+(y,t)\right]. \tag{13}$$

We have $d(0)>0$ and d is monotonically increasing, so $\partial D^-(y,t)/\partial t>0$ and $\partial D^+(y,t)/\partial t<0$. Thus, for fixed y, the function $D^-(y,\cdot)$ is continuous, monotonically increasing and satisfies $D^-(y,0)=0$, while the function $D^+(y,\cdot)$ is continuous, monotonically decreasing, and satisfies $D^+(y,1)=0$. Hence, the equation $D^-(y,t)=D^+(y,t)$ has a unique solution in t lying within $(0,1)$. Denote by $g(y)$ the solution corresponding to y. By using the implicit function theorem, the function $g:[0,1]\to[0,1]$ can be easily shown to be continuous and, by Brower's fixed point theorem [8, p. 161], g has a fixed point y^*. This y^* is an equilibrium. If there exist two equilibria y_1^* and y_2^* with $y_1^*<y_2^*$, then since $d(f)>0$ for all $f\ge 0$, we must have

$$D^-(y_1^*,y_1^*)<D^-(y_1^*,y_2^*)\le D^-(y_2^*,y_2^*)=D^+(y_2^*,y_2^*)$$
$$D^+(y_2^*,y_2^*)<D^+(y_2^*,y_1^*)\le D^+(y_1^*,y_1^*)=D^-(y_1^*,y_1^*)$$

which is impossible. Hence, the equilibrium is unique.

Q.E.D.

Proposition 4: The equilibrium minimizes over all $y\in[0,1]$ the expression

$$J(y)=\int_0^1 p\left[f^+(y,t)\right]dt+\int_0^1 p\left[f^-(y,t)\right]dt$$

where p is any function satisfying for all f

$$p'(f)=d(f) \tag{14}$$

and p' denotes the first derivative of p.

Proof: The first derivative $J'(y)$ of J is given by

$$J'(y)=\int_0^1 p'\left[f^+(y,t)\right]\frac{\partial f^+(y,t)}{\partial y}dt + \int_0^1 p'\left[f^-(y,t)\right]\frac{\partial f^-(y,t)}{\partial y}dt. \tag{15}$$

It can be seen from (7) and (8) that

$$\frac{\partial f^+(y,t)}{\partial y}=\begin{cases}-r(y), & \text{if } y<t\\ 0, & \text{if } t<y\end{cases} \tag{16}$$

$$\frac{\partial f^-(y,t)}{\partial y}=\begin{cases}0, & \text{if } y<t\\ r(y), & \text{if } t<y.\end{cases} \tag{17}$$

Combining (14)–(17) we obtain

$$J'(y)=r(y)\left[-\int_y^1 d[f^+(y,t)]\,dt+\int_0^y d[f^-(y,t)]\,dt\right]$$

or equivalently

$$J'(y)=r(y)[D^-(y,y)-D^+(y,y)].$$

If y^* is an equilibrium it can be seen that we have

$$D^-(y,y)\le D^+(y,y),\quad \text{if } y\le y^*$$
$$D^-(y,y)\ge D^+(y,y),\quad \text{if } y\ge y^*.$$

Thus, $J'(y)\le 0$ if $y\le y^*$, $J'(y)\ge 0$ if $y^*\le y$, and $J'(y^*)=0$. It follows that y^* minimizes J. Q.E.D.

Proposition 4 shows that one can minimize the integral of average delay over the ring by choosing the function d to be marginal delay *and* by guaranteeing that the algorithm converges to an equilibrium. The need to use marginal delays as link lengths in order to minimize total average delay has been pointed out earlier in a different algorithmic context [9]. Proposition 1, however, casts doubt as to whether the algorithm will converge to an equilibrium when the d is chosen to be marginal delay, since in this case value of d at zero flow will typically be near zero ($d(0)\simeq 0$). In any case, Proposition 4 suggests that convergence of the algorithm to an equilibrium is desirable since a function p satisfying (14) is monotonically increasing and convex and hence penalizes at an increasing rate large link flows. As a result, an equilibrium will at least be a reasonably good routing even if it is suboptimal in terms of a different objective function.

We now consider the convergence properties of the algorithm. For any $y\in[0,1]$ we denote by $g(y)$ the unique solution in t of the equation $D^-(y,t)=D^+(y,t)$ (cf. Proposition 1). Thus, Algorithm (A1) can be written

$$y_{k+1}=g(y_k). \tag{18}$$

We have for all $y\in[0,1]$

$$\begin{aligned}D^-[y,g(y)]&=\int_0^{g(y)} d[f^-(y,t)]\,dt\\&=\int_{g(y)}^1 d[f^+(y,t)]\,dt=D^+[y,g(y)].\end{aligned} \tag{19}$$

We evaluate the first derivative $g'(y)=dg(y)/dy$ for $y\in(0,1)$. Differentiation in (19) yields

$$\begin{aligned}&\int_0^{g(y)} d'[f^-(y,t)]\,dt+d[f^-(y,g(y))]g'(y)\\&\quad=\int_{g(y)}^1 d'[f^+(y,t)]\,dt-d[f^+(y,g(y))]g'(y)\end{aligned}$$

or

$$g'(y)=\frac{\int_{g(y)}^1 d'[f^+(y,t)]\,dt-\int_0^{g(y)}d'[f^-(y,t)]\,dt}{d[f^-(y,g(y))]+d[f^+(y,g(y))]}. \tag{20}$$

We have for $t\ne y$

$$\frac{\partial d[f^+(y,t)]}{\partial y}=d'[f^+(y,t)]\frac{\partial f^+(y,t)}{\partial y} \tag{21}$$

$$\frac{\partial d[f^-(y,t)]}{\partial y}=d'[f^-(y,t)]\frac{\partial f^-(y,t)}{\partial y}. \tag{22}$$

Combining (20)–(22) with (16), (17) we obtain

$$g'(y)=-\frac{r(y)\left[\int_0^{\min\{y,g(y)\}}d[f^-(y,t)]\,dt+\int_{\max\{y,g(y)\}}^1 d'[f^-(y,t)]\,dt\right]}{d[f^-(y,g(y))]+d[f^+(y,g(y))]}. \tag{23}$$

At the equilibrium y^*, we have $y^*=g(y^*)$ and $f^-(y^*,y^*)=f^+(y^*,y^*)=0$, so (23) yields

$$g'(y^*)=-\frac{r(y^*)\left[\int_0^{y^*}d'[f^-(y^*,t)]\,dt+\int_{y^*}^1 d'[f^+(y^*,t)]\,dt\right]}{2d(0)}. \tag{24}$$

By using a theorem of Ostrowski [8, pp. 300–301] we can state the following local convergence and rate of convergence result for Algorithm (*A*1).

Proposition 5: Let y^* be the equilibrium. Then if $|g'(y^*)|<1$ or equivalently

$$d(0)>\frac{r(y^*)\left[\int_0^{y^*}d'[f^-(y,t)]\,dt+\int_{y^*}^1 d'[f^+(y^*,t)]\,dt\right]}{2} \tag{25}$$

there exists an open interval I containing y^* such that if $y_0\in I$ the sequence $\{y_k\}$ generated by Algorithm (*A*1) remains in I and converges to y^*. Furthermore, if $y_k\ne y^*$ for all k there holds

$$\limsup_{k\to\infty}\frac{|y_{k+1}-y^*|}{|y_k-y^*|}=\limsup_{k\to\infty}|y_k-y^*|^{1/k}=|g'(y^*)|. \tag{26}$$

When the equilibrium y^* has the property specified in the first conclusion of Proposition 5 we say that it is *locally stable*. If $|g'(y^*)|>1$ then the linearized system corresponding to $y_{k+1}=g(y_k)$ is unstable, so the algorithm tends to diverge from y^* when started close to it. Notice the similarity of (25) with corresponding local stability conditions for finite node networks (cf. Proposition 2).

A sufficient condition for global convergence of Algorithm (*A*1) can be obtained by requiring that g be a contraction mapping, i.e., for some $\rho\in(0,1)$ there holds

$$|g(y)-y^*|\le\rho|y-y^*|,\quad \forall y\in[0,1]. \tag{27}$$

From Taylor's theorem and the fact $g'(y)\leq 0$ we have

$$|g(y)-y^*|=\left|\int_{y^*}^{y} g'(z)\,dz\right|.$$

Let

$$\beta=\max_{0\leq f\leq\int_0^1 r(t)\,dt} d'(f).$$

From (23) we obtain for all z

$$|g'(z)|\leq\frac{\beta r(z)[1-|z-g(z)|]}{2d(0)}.$$

Thus, (27) is satisfied if

$$\sup_{\substack{y\in[0,1]\\ y\neq y^*}}\frac{\beta}{2d(0)}\left|\frac{\int_{y^*}^{y} r(z)[1-|z-g(z)|]\,dz}{y-y^*}\right|<1$$

or equivalently if

$$d(0)>\frac{\beta}{2}\sup_{\substack{y\in[0,1]\\ y\neq y^*}}\left|\frac{\int_{y^*}^{y} r(z)[1-|z-g(z)|]\,dz}{y-y^*}\right|. \tag{28}$$

This will be true in particular if

$$d(0)>\frac{\beta R}{2} \tag{29}$$

where

$$R=\max_{0\leq t\leq 1} r(t).$$

The conclusions of the preceding discussion are summarized in the following proposition.

Proposition 6: If condition (28) or the stronger condition (29) holds, every sequence $\{y_k\}$ generated by algorithm $(A1)$ converges to the equilibrium y^*.

When the equilibrium y^* has the property specified in Proposition 6 we say that it is *globally stable.*

In order to put the results obtained thus far in better perspective let us write $d(f)$ as

$$d(f)=\alpha+\hat{d}(f)$$

where $\alpha=d(0)$ represents the bias factor. For fixed input density r we have that to each positive value of bias α there corresponds an equilibrium y_α^*. The equilibrium is locally stable for α satisfying [cf. (25)]

$$\alpha>\frac{r(y_\alpha^*)\left[\int_0^{y_\alpha^*} d'[f^-(y_\alpha^*,t)]\,dt+\int_{y_\alpha^*}^{1} d'[f^+(y_\alpha^*,t)]\,dt\right]}{2} \tag{30}$$

and globally stable for α satisfying [cf. (29)]

$$\alpha>\frac{\beta R}{2}. \tag{31}$$

As α increases the corresponding equilibria tend to become stable. Furthermore, from (24) and (26) it can be seen that the speed of convergence of the algorithm is accelerated as α increases. On the other hand, it is easy to see that $y_\alpha^*\to\frac{1}{2}$ as $\alpha\to\infty$, which in the context of the routing problem means that the algorithm becomes increasingly insensitive to congestion as $\alpha\to\infty$.

Since in a practical situation we are interested in the stability properties of the algorithm for a broad range of inputs let us consider input densities of the form

$$r_\lambda(t)=\lambda r(t) \tag{32}$$

where λ is a positive parameter. Then it is clear that as λ increases a larger value of bias is necessary in order to stabilize the algorithm.

For example if d is of the form

$$d(f)=\alpha+\beta f^n \tag{33}$$

where $\beta>0$, $n>0$ then from (30) and (31) we see the if r is changed to λr as in (32), then the stability threshold level of the bias is multiplied by λ^n. Thus, for fixed α and r there is a choice of λ for which the corresponding equilibrium is unstable. Incidentally, the expression (33) for d has an interesting property, namely, that the set of all possible equilibria $\{y_\alpha^*|\alpha>0\}$ as well as the set of all locally or globally stable equilibria is independent of the level of input λ and depends only on r. This is straightforward to verify using (33) and the fact that if r is changed to λr and α is changed to $\lambda^n\alpha$ then the routing sequences generated by the algorithm are unaffected.

Choosing the Bias as a Function of the Current Routing

Since stability of the algorithm depends strongly on the level of bias and the level of input we are motivated to consider schemes where the bias is not held fixed but is rather adjusted adaptively on the basis of currently available information. An interesting scheme is to use a length function of the form

$$d(f,y)=\alpha(y)+\hat{d}(f)$$

where $\hat{d}$ is a continuously differentiable, monotonically increasing function with $\hat{d}(0)=0$, and $\alpha(y)$ is taken to be some monotonically nondecreasing function of $D_T(y)$ given by

$$D_T(y)=\int_0^1\hat{d}[f^+(y,t)]\,dt+\int_0^1\hat{d}[f^-(y,t)]\,dt.$$

For example a quadratic function of the form

$$\alpha(y)=\gamma_0+\gamma_1 D_T(y)+\gamma_2[D_T(y)]^2 \tag{34}$$

where γ_0, γ_1, γ_2 are some experimentally determined nonnegative constants seems suitable. In the context of a finite node network with not necessarily a ring structure a scheme like this can be very easily implemented. In this case $D_T(y)$ can be calculated as the sum of all reported link "delays" $\hat{d}(f_{il})$. The bias $\alpha(y)$ can be computed by each node via a formula such as (34) and the link length can be computed as $D_{il} = \alpha(y) + \hat{d}(f_{il})$.

A scheme of the type just described can be analyzed along similar lines as earlier in this section. It has been tested in quite extensive numerical experiments involving finite node networks and it was shown to have very satisfactory performance [3], [4]. This can be attributed to the fact that the level of bias increases or decreases with the level of input thus providing *automatic scaling with respect to input level.* In fact, it can be easily seen that if $\hat{d}$ has the form $\hat{d}(f) = \beta f^n$ where $\beta > 0$, $n > 0$ and we choose $\alpha(y) = \lambda D_T(y)$ where $\lambda > 0$, then for every input density function of the form $\lambda r(t)$, $\lambda > 0$, the sequences generated by the algorithm do not depend on λ.

IV. Averaging the Effect of Several Routings

In this section, we show that the stability properties of the shortest path algorithm of the preceding section can be improved if link lengths suitably depend on flows corresponding to several past routings. There are several possibilities along these lines. Some examples are as follows.

A. Averaging Over the Present and the Past n Routings

Given a sequence of past routing $y_k, y_{k-1}, \cdots$, we define for any t in [0, 1] "averaged" distances to 0 and 1 by

$$\tilde{D}^-(y_k, y_{k-1}, \cdots, y_{k-n}, t) = \int_0^t \frac{1}{n+1} \sum_{i=0}^{n} d[f^-(y_{k-i}, \tau)]\, d\tau \tag{35}$$

$$\tilde{D}^+(y_k, y_{k-1}, \cdots, y_{k-n}, t) = \int_t^1 \frac{1}{n+1} \sum_{i=0}^{n} d[f^+(y_{k-i}, \tau)]\, d\tau. \tag{36}$$

Thus, distances are calculated by intergrating

$$\frac{1}{n+1} \sum_{i=0}^{n} d[f(y_{k-}, \tau)]$$

which is an averaged length over the routings $y_k, \cdots, y_{k-n}$, in place of $d[f(y_k, \tau)]$ which is the length corresponding to the last routing.

The new routing y_{k+1} is obtained from the equation

$$\tilde{D}^-(y_k, y_{k-1}, \cdots, y_{k-n}, y_{k+1}) = \tilde{D}^+(y_k, y_{k-1}, \cdots, y_{k-n}, y_{k+1}). \tag{37}$$

It is easily seen that this defines uniquely y_{k+1} in terms of $y_k, y_{k-1}, \cdots, y_{k-n}$. As earlier we write the corresponding equation as

$$y_{k+1} = g(y_k, y_{k-1}, \cdots, y_{k-n}). \tag{38}$$

A routing y^* is said to be an *equilibrium* if

$$y^* = g(y^*, y^*, \cdots, y^*).$$

It is clear that y^* *is an equilibrium in this sense for a given bias level if and only if it is an equilibrium in the sense given in the preceding section.*

We can define local stability of y^* in the obvious way. We have that y^* is locally stable if it is also a stable equilibrium of equation (38) linearized around y^* (see [8, p. 353]). It is a known fact that this is true if all roots of the characteristic polynomial

$$C(p) = p^{n+1} - \frac{\partial g(y^*)}{\partial y_k} p^n - \frac{\partial g(y^*)}{\partial y_{k-1}} p^{n-1} - \cdots - \frac{\partial g(y^*)}{\partial y_{k-n-1}} p - \frac{\partial g(y^*)}{\partial y_{k-n}}$$

lie inside the unit circle, (i.e., have modulus less than unity). We calculate the derivatives $\partial g / \partial y_{k-i}$.

We have for $\alpha > 0$ similarly as earlier for every i

$$\frac{\partial g(y^*)}{\partial y_{k-i}} = -\frac{r(y^*)}{2d(0)} \frac{1}{n+1} \cdot \left\{ \int_0^{y^*} d'[f^-(y^*, t)]\, dt + \int_{y^*}^1 d'[f^+(y^*, t)]\, dt \right\}.$$

Define

$$\mu = \frac{r(y^*)}{2d(0)} \left\{ \int_0^{y^*} d'[f^-(y^*, t)]\, dt + \int_{y^*}^1 d'[f^-(y^*, t)]\, dt \right\}. \tag{39}$$

Note that, from Proposition 5, y^* is locally stable for algorithm $(A1)$ if $\mu < 1$. The characteristic polynomial can be written as

$$C(p) = p^{n+1} + \frac{\mu}{n+1} p^n + \frac{\mu}{n+1} p^{n-1} + \cdots + \frac{\mu}{n+1} p + \frac{\mu}{n+1}. \tag{40}$$

We now use the following fact.

Lemma: Let ξ be a positive scalar and n be a positive integer. The roots of the polynomial

$$p^{n+1} + \xi p^n + \xi p^{n-1} + \cdots + \xi p + \xi$$

lie inside the unit circle if and only if $\xi < 1$.

Proof: This result can be shown by straightforward application of Jury's stability test [10, p. 97–98]. Q.E.D.

We now apply the result of the lemma to our problem. We have that the equilibrium y^* will be locally stable if

$$\mu < n + 1.$$

It follows using (39) that in the averaged algorithm the bias level must satisfy

$$d(0) > \frac{r(y^*) \left\{ \int_0^{y^*} d'[f^-(y^*, t)]\, dt + \int_{y^*}^1 d'[f^-(y^*, t)]\, dt \right\}}{2(n+1)}$$

in order for the corresponding equilibrium y^* to be stable. If we compare this with the earlier algorithm [cf. (25)] we see that *in the averaged algorithm the bias threshold level for stability is reduced by the factor* $1/(n+1)$ *over the one of algorithm* ($A1$). For a given traffic input, and any given bias level the corresponding equilibrium can be made stable by averaging delays over a sufficiently large number of periods.

Regarding rate of convergence, Ostrowski's theorem again applies. We have from the proof of [8, Theorem 10.1.3] that given any $\epsilon > 0$ there exists a norm $\|\cdot\|$ on R^{n+1} such that, if $y_k \neq y^*$ for all k, then

$$\limsup_{k\to\infty} \frac{\|(y_{k+1}-y^*,\cdots,y_{k-n+1}-y^*)\|}{\|(y_k-y^*,\cdots,y_{k-n}-y^*)\|} \leq \rho(\mu,n)+\epsilon$$

where $\rho(\mu,n)$ is the maximum root modulus of the characteristic polynomial $C(p)$ of (40). It can be seen that for fixed n we have $\rho(\mu,n)\to 0$ as $\mu\to 0$. If $\rho_1,\cdots,\rho_{n+1}$ are the roots of $C(p)$ we have $|\rho_1\cdots\rho_{n+1}|=\mu/(n+1)$, so that $\rho(\mu,n)\geq(\mu/(n+1))^{1/(n+1)}$. It follows that for fixed μ we have $\rho(\mu,n)\to 1$ as $n\to\infty$, so that the rate of convergence deteriorates as $n\to\infty$. Thus, too much damping can slow down the speed of convergence of the algorithm.

B. *Fading Memory Scheme*

This scheme is similar to the preceding one except that the lengths corresponding to all past routings are averaged via a fading memory scheme. Given the sequence of all past routings $\{y_k, y_{k-1},\cdots\}$, the next routing y_{k+1} is determined as the solution of the equation

$$\int_0^{y_{k+1}} \delta_k^-(t)\,dt = \int_{y_{k+1}}^1 \delta_k^+(t)\,dt \tag{41}$$

where δ_k^- and δ_k^+ are obtained by the following exponential fading memory scheme with decay factor $\beta\in[0,1)$

$$\delta_k^-(t)=\beta\delta_{k-1}^-(t)+(1-\beta)d[f^-(y_k,t)]$$
$$\delta_k^-(t)=\beta\delta_{k-1}^+(t)+(1-\beta)d[f^+(y_k,t)].$$

Alternatively, we can write

$$\delta_k^-(t)=(1-\beta)\sum_{i=-\infty}^{k}\beta^{k-i}d[f^-(y_i,t)] \tag{42}$$

$$\delta_k^+(t)=(1-\beta)\sum_{i=-\infty}^{k}\beta^{k-i}d[f^+(y_i,t)]. \tag{43}$$

Let us write the solution of (46) as

$$y_{k+1}=g(y_k,y_{k-1},\cdots). \tag{44}$$

Let us also consider the linear system obtained by formal linearization of (44) around the equilibrium y^*. Similarly, as mentioned earlier, this linearized system is

$$y_{k+1}=-\mu(1-\beta)[y_k+\beta y_{k-1}+\beta^2 y_{k-2}+\cdots] \tag{45}$$

where μ is given by (39). Let us denote

$$z_{k+1}=y_k+\beta y_{k-1}+\beta^2 y_{k-2}+\cdots.$$

Then we have for all k

$$y_{k+1}=-\mu(1-\beta)y_k-\mu(1-\beta)\beta z_k \tag{46}$$

$$z_{k+1}=y_k+\beta z_k \tag{47}$$

and it follows that the linearized system (45) is in effect the two-dimensional system described by (46) and (47). This latter system is stable if both eigenvalues of the system matrix

$$\begin{bmatrix} -\mu(1-\beta) & -\mu(1-\beta)\beta \\ 1 & \beta \end{bmatrix}$$

lie within the unit circle. These two eigenvalues can be calculated to be 0 and $\beta-\mu(1-\beta)$. It follows that the linearized system is stable if

$$\mu<\frac{1+\beta}{1-\beta}.$$

Although we do not provide a proof, it is possible to establish rigorously that stability of the linearized system (45) implies local stability of the algorithm (44) and thus we have the result that *the threshold value of bias for stability in the fading memory scheme is reduced by the factor* $(1-\beta)/(1+\beta)$ *over the one of algorithm* ($A1$). The optimal speed of convergence is obtained when the eigenvalue $\beta-\mu(1-\beta)$ equals zero in which case a superlinear rate of convergence is obtained. This is so when $\beta=\mu/(1+\mu)$. For other values of β in the interval $((\mu-1)/(\mu+1),1)$ the rate of convergence is linear, and for $\beta<(\mu-1)/(\mu+1)$ the equilibrium is unstable. As β is increased from the optimal value $\mu/(1+\mu)$ towards unity the rate of convergence deteriorates.

C. *Asynchronous Length Reporting*

This type of scheme is patterned after a shortest path routing algorithm where nodes report asynchronously the lengths of their outgoing links and the shortest paths are updated after each report. The set of nodes $[0,1]$ is partitioned into n subsets which we call $S_1, S_2,\cdots,S_n$. At some time, say 0, the nodes in S_1 report their lengths averaged over the flows corresponding to the preceding n routings and a routing update takes place. Then at time $\sigma_1>0$ the nodes in S_2 do the same thing. Similarly, for $i=1,\cdots,n-1$, at time $(\sigma_1+\sigma_2+,\cdots,+\sigma_i)$ the nodes in S_{i+1} do the same thing. At time $(\sigma_1+\sigma_2+\cdots+\sigma_n)$ the nodes in S_1 again report their lengths, an updating takes place and the process is repeated. This type of asynchronous operation is currently in use in the ARPANET [4] where, in a finite-node network context, S_i consists of a single node for all i. There are also other variations of asynchronous operation involving for example averaging over all preceding routings via a fading memory scheme. This type of algorithm is described and tested computationally in [3] and [4]. The analysis of all these schemes is very similar as that of the averaging schemes described earlier in this section. The details are quite messy and may be found in [4], where it is shown, via analysis and computational experiment, that asynchronous operation has a substantial beneficial effect on the stability properties of the shortest path algorithm.

V. The Case of a Network with an Arbitrary Topology

The extension of the continuous model to the case of a network with arbitrary topology is quite straightforward. However, the notation required for a precise mathematical description is very cumbersome and tends to cloud the main ideas. For this reason our presentation will be somewhat informal.

Consider the case of an undirected network with a single destination. Let r be the input density function mapping points on the undirected links of the network to the nonnegative real numbers. The meaning of r again is that, given any interval I on a link, the total traffic input originating at this interval is the integral of r over I. We view the set of points on the network as a subset of a Euclidean space of dimension 2 or 3 (depending on whether the network is planar or not), and assume that r is a continuous function. In order to consider notions of length we associate with each undirected link (i,l) two directions $i \to l$ and $l \to i$. (There may be more than one links connecting a pair of nodes within our framework. When we refer to a link (i,l) we mean a particular link connecting i and l and specify further when there is danger of confusion.) A *length function* δ is a function which assigns to each point on an undirected link (i,l) two nonnegative numbers one associated with the direction $i \to l$ and the other associated with the direction $l \to i$. We assume that δ is piecewise continuous along every link in each direction. The meaning of δ is that given any two points on a link (i,l) their distance in the direction $i \to l$ is obtained by integrating δ as defined in that direction between the two points. The distance in the opposite direction $l \to i$ is defined analogously. Similarly, we can consider paths between points on possibly different links and define their length in one or the other direction.

We now associate to a given length function δ a shortest path of every point, and an associated routing. *We assume that δ is everywhere positive.* Given any point we consider the collection of paths to the destination and their associated distances specified by the function δ. A path of minimum distance is referred to as a shortest path from the point to the destination, and the corresponding distance is referred to as the shortest distance of the point to the destination. The *routing* corresponding to δ is the set of points for which there are more than one equidistant paths to the destination. A routing is said to be *regular* if it does not contain any nodes of the network, otherwise it is said to *singular*.

For a given δ, a shortest path of each point and the corresponding routing can be constructed in a simple manner along similar lines as for usual networks. We first construct a shortest path tree for the network in the usual manner by using as (directed) link lengths those specified by the length function δ. (The length of the directed link (i,l) is the integral of δ along (i,l) in the direction $i \to l$.) This gives us a shortest path and the associated shortest distance for every point on the shortest path tree including all the nodes of the network. A shortest path for points on links that are not part of the shortest path tree can be obtained as follows.

Let (i,l) be a link that is not on the tree. Let D_i and D_l be the shortest distances of nodes i and l. The shortest distance of a point t on (i,l) is

$$D(t) = \min\left\{D_i + \int_i^t \delta_{li}(\tau)\,d\tau,\; D_l + \int_t^l \delta_{il}(\tau)\,d\tau\right\}$$

where δ_{li} is δ in the direction $l \to i$ and δ_{il} is δ in the direction $i \to l$. It can be seen that the routing corresponding to δ is regular if and only if each (ordinary) node of the network has only one shortest path associated with it. If a routing is regular then every one of its points lies in the "interior" of some link. Notice that the preceding construction shows that a routing (regular or not) consists of $(L-N+1)$ points where L and N are the number of undirected links and nodes, respectively.

Given a shortest path tree and the corresponding routing constructed as just described, we can define the flow corresponding to it. At each point, say t, of a link (i,l) there are two flows to consider (one of which is zero); the flow in the direction $i \to l$ and the flow in the direction $l \to i$. Each is defined in the natural way by integrating the input density function r over the portion of the network that lies "upstream" from the point t, i.e., over the set of points the shortest paths of which meet t on their way to the destination. At the points of a regular routing the flow is zero in either direction. Notice that if δ is such that the corresponding routing is regular the flow is uniquely determined by δ. Otherwise, the flow will depend not only on δ but also on the shortest path tree selected.

Suppose we are given a monotonically increasing, continuously differentiable function d mapping flow into the positive numbers. Given a shortest path tree T corresponding to a length function δ with routing Y we can define a new length function $\bar{\delta}$ which assigns to points t in any one of the two possible directions the length $\bar{\delta}(t) = d[f(t)]$ where $f(t)$ is the flow at t corresponding to δ and T in the appropriate direction. The corresponding routing is denoted $\bar{Y}$. Note that if Y is singular then $\bar{\delta}$ and $\bar{Y}$ depend not only on δ but also on T. If Y is regular than $\bar{Y}$ is uniquely determined by δ.

We are now in a position to define an algorithm similar to the one of Section III. Given a length function δ_0 and a corresponding shortest path tree T_0 and routing Y_0, the next length function is $\delta_1 = \bar{\delta}_0$ with corresponding routing $Y_1 = \bar{Y}_0$. A shortest path tree T_1 corresponding to δ_1 is selected and is used to define similarly δ_2, Y_2, and T_2. Similarly, the algorithm generates δ_k, Y_k, and T_k for all k.

We say that a routing Y^* corresponding to a length function δ^* and shortest path tree T^* is an *equilibrium routing* if $\bar{\delta}^* = \delta^*$ and $\bar{Y}^* = Y^*$.

Contrary to the case of a ring network where we were able to prove existence of an equilibrium, in general there need not exist an equilibrium. This fact is demonstrated in the following example and provides an indication of the complexity of the dynamic phenomena that we are investigating.

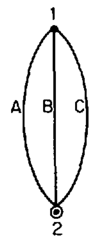


Fig. 5.

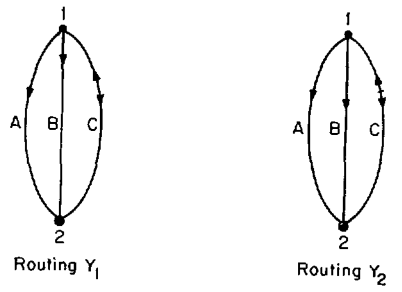


Fig. 6.

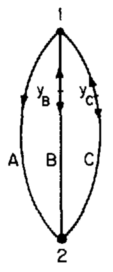


Fig. 7.

Example: Consider the network shown in Fig. 5. There are two nodes 1 and 2 and three links connecting them denoted by A, B, C. Node 2 is the destination. Points on A, B, and C are parameterized by their Euclidean distance to the destination. The Euclidean lengths of A, B, and C are all taken equal to unity. Let the input density function be as follows:

For link A:	$r(t)\equiv 1,$	$\forall t\in[0,1]$
For link B:	$r(t)\equiv r_B,$	$\forall t\in[0,1]$
For link C:	$r(t)\equiv r_C,$	$\forall t\in[0,1].$

We assume that $1\le r_B\le r_C$, $1<r_C$. Let

$$d(f)=\alpha+f$$

where $\alpha>0$ is the bias factor.

In view of the fact $1\le r_B\le r_C$, $1<r_C$, it is clear that an equilibrium routing cannot contain a point in the interior of link A, while it must contain a point in the interior of link C. We consider the following two cases.

Case 1: $r_B=1$. Then an equilibrium routing cannot contain a point in the interior of link B so the only candidate for equilibrium are the two types of singular routings shown in Fig. 6. In routings Y_1 and Y_2 the incoming traffic at node 1 is routed through link A and link B, respectively. None of the two routings can be an equilibrium. In routing Y_1 there will be points in the interior of link A which will have a shorter distance to the destination (corresponding to Y_1) through link B rather than through A, and the reverse situation occurs in routing Y_2. Notice that this argument makes use only of the magnitude of r_B and r_C and is independent of the form of the function d.

Case 2: $1<r_B$. Then it can be seen that the only candidates for equilibria are routings of the form shown in Fig. 7. Each equilibrium routing candidate is specified by the points $y_B, y_C\in[0,1]$ where the flow separates on links B and C. We have that the distances $D^+(y_B)$, $D^-(y_B)$ of y_B corresponding to routing (y_B, y_C) along the counterclockwise the clockwise paths, respectively, are given by

$$D^-(y_B)=\alpha y_B+r_B\int_0^{y_B}(y_B-t)\,dt$$

$$D^+(y_B)=(2-y_B)\alpha+r_B\int_{y_B}^1(t-y_B)\,dt$$
$$+\int_0^1\left[r_B(1-y_B)+r_C(1-y_C)+(1-t)\right]dt.$$

If (y_B, y_C) is an equilibrium we must have

$$D^-(y_B)=D^+(y_B)$$

which after some calculation can be written as

$$2(\alpha+r_B)(1-y_B)+r_C(1-y_C)=\frac{r_B-1}{2}. \tag{48}$$

By symmetry the equation $D^-(y_C)=D^+(y_C)$ can be written as

$$r_B(1-y_B)+2(\alpha+r_C)(1-y_C)=\frac{r_C-1}{2}. \tag{49}$$

Equations (48) and (49) are in fact necessary and sufficient conditions for (y_B, y_C) to be an equilibrium routing. Thus, there exists an equilibrium routing if and only if the solution (y_B^*, y_C^*) to these equations satisfies $y_B^*\in[0,1]$, $y_C^*\in[0,1]$. After some calculation, this condition can be shown to be equivalent to

$$\alpha\ge-\frac{r_C(2r_B-r_C-1)}{2(r_B-1)}. \tag{50}$$

If $2r_B\ge r_C+1$ then for every level of bias there exists an equilibrium routing (y_B, y_C). If, however, $2r_B<r_C+1$, then there exists an equilibrium only for α above the threshold level indicated in (50).

The preceding example shows that existence of an equilibrium can depend on both the level of bias and the input density function. Furthermore, it may happen that, for a given input density function, no value of bias can be found for which an equilibrium exists. This last phenomenon is of a singular nature and is due to the fact that the Euclidean lengths of links A, B, and C are all equal to unity. To see this consider the routing Y_∞ corresponding to the length function $\delta^+(t)\equiv 1$, $\delta^-(t)\equiv 1$. The routing Y_∞ is analogous to the min-hop routing in discrete node networks, and can be associated with infinite level of bias. It is an equilibrium routing for the case $d(f)\equiv 1$. If Y_∞ is a regular routing, i.e., each node has a unique minimum Euclidean distance path to the destination, then it is clear that, for any given input function r, there exists a threshold level of bias $\bar{\alpha}$ such that for all $\alpha \geq \bar{\alpha}$ a regular equilibrium routing exists.

Characterizing the dynamic behavior of the algorithm in the absence of an equilibrium is certainly an interesting problem but we have been unable to make much progress in this direction. Computational results for finite node networks given in [3] suggest that the stability properties of the algorithm are improved by high level of bias and averaging similarly as in the presence of an equilibrium. In what follows in this section we restrict attention to the case where a regular equilibrium routing exists.

Given a regular equilibrium routing $Y^* = \{y_1^*, y_2^*, \cdots, y_n^*\}$ consider for $j = 1, 2, \cdots, n$ the link (i_j, l_j) containing y_j^* and the two shortest paths from y_j^* to the destination. A simple but fundamental observation is that *these two paths join at some point thereby forming a ring of the type considered in Section III.* The zero point on this ring is the point where the two paths join. Let e_j be the Euclidean length of the ring containing y_j^*. For $j = 1, 2, \cdots, n$ we parameterize points on the ring containing y_j^* by the number in $[0, e_j]$ going from smaller to larger numbers as we traverse the ring in a chosen direction similarly as in the previous two sections. Thus, points y_j on the link (i_j, l_j) can and will be identified by the number in $[0, e_j]$ specifying their position on the ring corresponding to y_j^*. It is easy to see now that given Y^*, any collection $Y = \{y_1, y_2, \cdots, y_n\}$ such that y_j lies in the interior of (i_j, l_j) specifies a flow f_Y through each point in the network that follows the (ordinary) shortest path tree corresponding to δ^* and Y^* and separates on each link (i_j, l_j) in the two opposite directions at the point y_j. This flow defines a length function δ_Y via the relation $\delta_Y(t) = d[f_Y(t)]$ in the direction of the flow, and δ_Y yields in the manner described earlier a shortest path tree and a routing denoted by $g(Y)$. It is easy to show (using the regularity of Y^*) that if Y is sufficiently close to Y^* then the (ordinary) shortest path tree corresponding to δ_Y is the same as the one corresponding to Y^* and that the elements of the routing $g(Y)$ lie on the links (i_j, l_j).

The algorithm described earlier can now be redefined as

$$Y_{k+1} = g(Y_k). \tag{51}$$

The definition is local within a sufficiently small neighborhood of Y^* and is associated with the (ordinary) shortest path tree corresponding to Y^* and the associated parameterization of the ring subnetworks containing the links (i_j, l_j).

Similarly, as in the preceding section we say that an equilibrium Y^* is *locally stable* if there is a neighborhood of Y^* (defined in terms of the parameterization of the rings associated with Y^* as discussed earlier), such that the sequence $\{g(Y_k)\}$ generated by (50) is well defined and converges to Y^* for every choice of Y_0 within this neighborhood.

In order for Y^* to be locally stable it is sufficient that the $n \times n$ matrix $\partial g(Y^*)/\partial Y$ be defined and have all its eigenvalues within the unit circle. The computation of $\partial g(Y^*)/\partial Y$ is straightforward along the lines of Section III. We first introduce some notation. For $j = 1, 2, \cdots, n$ let $R^+_{y_j, e_j}$ denote the set of points $t \in [y_j, e_j]$ on the jth ring, and $R^-_{y_j, e_j}$ denote the set of points $t \in [0, y_j]$ on the same ring. Note that for every $j, m = 1, \cdots, n$ the direction of flow on $R^+_{y_j, e_j}$ and $R^+_{y_m, e_m}$ (or $R^-_{y_m, e_m}$) must coinside if these sets have intersection with positive Lebesgue measure. This implies that at least one of the sets $R^+_{y_j, e_j} \cap R^+_{y_m, e_m}$ and $R^+_{y_j, e_j} \cap R^-_{y_m, e_m}$ is either empty or has Lebesgue measure zero. Similarly, at least one of the sets $R^-_{y_j, e_j} \cap R^-_{y_m, e_m}$ and $R^-_{y_j, e_j} \cap R^+_{y_m, e_m}$ is either empty or has Lebesgue measure zero. The equations defining $g(Y)$ can be written as

$$\int_{R^-_{g_j(Y), e_j}} d[f^-(Y, t)]\, dt = \int_{R^+_{g_j(Y), e_j}} d[f^+(Y, t)]\, dt, \qquad j = 1, \cdots, n.$$

By differentiation with respect to y_m we obtain similarly as earlier at the equilibrium Y^*

$$\frac{\partial g_j(Y^*)}{\partial y_m} = \frac{r(y_m^*)}{2d(0)} \theta_{jm}$$

where

$$\begin{aligned}\theta_{jm} = &-\int_{R^+_{y_j^*, e_j} \cap R^+_{y_m^*, e_m}} d'\left[f_j^+(Y^*, t)\right] dt \\ &-\int_{R^-_{y_j^*, e_j} \cap R^-_{y_m^*, e_m}} d'\left[f_j^-(Y^*, t)\right] dt \\ &+\int_{R^+_{y_j^*, e_j} \cap R^-_{y_m^*, e_m}} d'\left[f_j^-(Y^*, t)\right] dt \\ &+\int_{R^-_{y_j^*, e_j} \cap R^+_{y_m^*, e_m}} d'\left[f_j^-(Y^*, t)\right] dt\end{aligned} \tag{52}$$

and $f_j^+(Y^*, t), f_j^-(Y^*, t)$ are the flows on the jth ring in the positive and negative directions. In view of the preceding discussion, at least two of the integrals in (52) are zero for every j and m.

Let R be the diagonal matrix having $r(y_j^*)$ as jth diagonal element, and let Θ be the $n \times n$ matrix having as elements the scalars θ_{jm}. Then we have

$$\frac{\partial g(Y^*)}{\partial Y} = \frac{1}{2d(0)} \Theta R.$$

We can show that the *matrix Θ is negative semidefinite.* Indeed the matrix $-\Theta$ is the Gram matrix associated with the functions

$$\chi_{R^+_{y_j^*, e_j}}(t) d'\left[f_j^+(Y^*, r)\right] - \chi_{R^-_{y_j^*, e_j}}(t) d'\left[f_j^-(Y^*, t)\right], \qquad j = 1, \cdots, n$$

where χ_S is the characteristic function of a set S ($\chi(t) = 1$ if $t \in S$, $\chi(t) = 0$ otherwise). By using the fact that R is diagonal it can be shown that the eigenvalues $\lambda_1, \cdots, \lambda_n$ of

$\partial g(Y^*)/\partial Y$ are real and nonpositive. Consider the spectral radius

$$\mu = \max\{|\lambda_1|, \cdots, |\lambda_n|\}.$$

Then the equilibrium Y^* is locally stable for

$$\mu < 1 \tag{53}$$

and hence there exists a threshold level for $d(0)$ above which the corresponding equilibrium is stable. Similarly, as in the preceding section, we can show that if a fading memory scheme with decay factor β is used to average the effects of past routings the equilibrium Y^* is locally stable if

$$\mu < \frac{1+\beta}{1-\beta} \tag{54}$$

and there is a value of β which optimizes the rate of convergence. It is also possible to show that the other forms of averaging the effects of several past routings improve the stability properties of the algorithm.

For the purpose of aiding the reader in understanding the method of calculation of the matrix $\partial g(Y^*)/\partial Y$ we provide an example.

Example: Consider the network shown in Fig. 8 where node 4 is the destination, and assume that the regular routing $\{y_1^*, y_2^*, y_3^*\}$ shown is an equilibrium. The figure shows also the chosen positive direction on the ring corresponding to each y_i^*.

We calculate the symmetric matrix Θ with elements θ_{jm} given by (52). The interval between any two nodes i and l is denoted $[i, l]$. The interval between some y_i^* and a node l is denoted $[y_i^*, l]$. We have

$$\theta_{11} = -\int_{[y_1^*,1]\cup[1,3]\cup[3,4]} d'[f_1^+(Y^*,t)]\,dt - \int_{[y_1^*,4]} d'[f_1^-(y^*,t)]\,dt$$

$$\theta_{22} = -\int_{[y_2^*,2]\cup[2,3]} d'[f_2^+(Y^*,t)]\,dt - \int_{[y_2^*,1]\cup[1,3]} d'[f_2^-(Y^*,t)]\,dt$$

$$\theta_{33} = -\int_{[y_3^*,2]\cup[2,3]\cup[3,4]} d'[f_3^+(Y^*,t)]\,dt - \int_{[y_3^*,4]} d'[f_3^-(Y^*,t)]\,dt$$

$$\theta_{12} = -\int_{[1,3]} d'[f_1^-(Y^*,t)]\,dt$$

$$\theta_{23} = -\int_{[2,3]} d'[f_2^-(Y^*,t)]\,dt$$

$$\theta_{13} = -\int_{[3,4]} d'[f_1^+(Y^*,t)]\,dt.$$

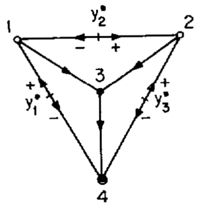


Fig. 8.

VI. Summary and Conclusions

The analysis of this paper shows that adaptive shortest path routing algorithms for packet communication networks exhibit complex dynamic behavior that should be taken into account in their design. This is particularly so if link lengths are chosen to be a measure of delay on the corresponding queue, and the delay per packet due to processing and transmission is small relative to average queuing delay. A stability analysis based on a discrete and a continuous-node model shows that oscillatory behavior can be damped by making use of a bias factor at the expense of reduced sensitivity of the routing algorithm to traffic congestion. Oscillations can also be damped by a scheme that averages the effects of several past routings such as the one introduced by asynchronous link length reporting by nodes (e.g., the one used in the ARPANET [11]). This is particularly fortunate since asynchronous length reporting offers significant practical implementation advantages. Most of the analysis given relates to a ring topology but, as shown in Section V, the ring structure is a fundamental building block for the analysis and extension of our results to more complex topologies. The qualitative conclusions drawn from the analysis of single destination networks with continuum of nodes have been extended to multiple destination networks in [3]. Their validity for finite-node networks has been verified by extensive computational experimentation the results of which are given in [3] and [4].

Acknowledgment

This investigation was conducted while the author participated in a design study of the new ARPANET routing algorithm at Bolt Beranek and Newman, Inc., Cambridge, MA. The collaboration and discussions with J. McQuillan, I. Richer, and E. Rosen had a substantial influence on the ideas of this paper.

References

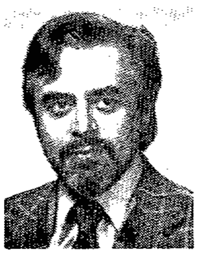
[1] J. M. McQuillan, I. Richer, and E. C. Rosen, "ARPANET routing algorithm improvements first semiannual technical report," Bolt Beranek and Newman, Inc., BBN Rep. 3803 prepared for ARPA and DCA, Apr. 1978.

[2] D. P. Bertsekas, "Algorithms for optimal routing of flow in networks," Coordinated Science Lab., Working Paper, Univ. Illinois, Urbana, June 1978; also in *International Symposium on Systems Optimization and Analysis*, A. Bensoussan and J. L. Lions, Eds. New York: Springer-Verlag, 1979, pp. 210–224.

[3] ——, "Dynamic models of shortest path routing algorithms for communication networks with multiple destinations," in *Proc. 1979 IEEE Conf. Decision and Control*, Ft. Lauderdale, FL, pp. 127–133, Dec. 1979.

[4] J. M. McQuillan, I. Richer, E. C. Rosen, and D. P. Bertsekas,

"ARPANET routing algorithm improvements, second semiannual Rep.," Bolt, Beranek, and Newman, Inc., BBN Rep. prepared for ARPA and DCA, Oct. 1978.
[5] J. McQuillan, G. Falk, and I. Richer, "A review of the development and performance of the ARPANET routing algorithm," *IEEE Trans. Commun.*, vol. COM-26, Dec. 1978.
[6] M. Schwartz, *Computer-Communication Network Design and Analysis.* Englewood Cliffs, NJ: Prentice-Hall, 1977.
[7] L. Kleinrock, *Queuing Systems*, Vol. I and II. New York: Wiley, 1976.
[8] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables.* New York: Academic, 1970.
[9] R. Gallager, "A minimum delay routing algorithm using distributed computation," *IEEE Trans. Commun.*, vol. COM-25, pp. 73–85, 1977.
[10] E. Jury, *Theory and Application of the z-Transform Method.* New York: Wiley, 1964.
[11] J. McQuillan, I. Richer, and E. Rosen, "The new routing algorithm for the ARPANET," *IEEE Trans. Commun.*, vol. COM-28, pp. 711-719, 1980.
[12] D. P. Bertsekas, "A class of optimal routing algorithms for communication networks," in *Proc. 5th Int. Conf. Computer Communication (ICCC'80)*, Atlanta, GA, pp. 71–76, Oct. 1980.
[13] D. P. Bertsekas, E. Gafni, and R. G. Gallager, "Second derivative algorithms for minimum delay distributed routing in networks," L.I.D.S. working paper, Massachusetts Inst. Technol., Cambridge, Jan. 1981 (submitted for publication).



Dimitri P. Bertsekas (S'70–SM'77) was born in Athens, Greece, in 1942. He received the Mechanical and Electrical Engineering Diploma from the National Technical University of Athens, Greece, in 1965, the M.S.E.E. degree from George Washington University, Washington, DC, in 1969, and the Ph.D. degree in system science from the Massachusetts Institute of Technology, Cambridge, in 1971.

He has held faculty positions with the Engineering-Economic Systems Department, Stanford University, Stanford, CA, and the Electrical Engineering Department of the University of Illinois, Urbana. He is currently in the faculty of the Department of Electrical Engineering and Computer Science of the Massachusetts Institute of Technology. He consults regularly with private industry and he is an Associate Editor for the *SIAM Journal on Control and Optimization*. He has done research in the areas of estimation and control of stochastic systems, nonlinear and dynamic programming, and data communication networks, and has written numerous papers in each of these areas. He is the author of *Dynamic Programming and Stochastic Control* (Academic, 1976), *Constrained Optimization and Lagrange Multiplier Methods* (Academic, 1981), and the coauthor of *Stochastic Optimal Control: The Discrete-Time Case* (Academic, 1978).

Short Papers

Reconstruction of Atmospheric Pollutant Concentrations from Remote Sensing Data—An Application of Distributed Parameter Observer Theory

MASATO KODA AND JOHN H. SEINFELD

Abstract—**The reconstruction of a concentration distribution from spatially averaged and noise-corrupted data is a central problem in processing atmospheric remote sensing data. Distributed parameter observer theory is used to develop reconstructibility conditions for distributed parameter systems having measurements typical of those in remote sensing. The relation of the reconstructibility condition to the stability of the distributed parameter observer is demonstrated. The theory is applied to a variety of remote sensing situations, and it is found that those in which concentrations are measured as a function of altitude satisfy the conditions of distributed state reconstructibility.**

I. Introduction

In the remote sensing of tropospheric species, a ground-, aircraft-, or satellite-based platform performs an instantaneous scan of a region of the atmosphere and measures the species burden within the field of view. With aircraft or satellite remote sensing the platform is in motion and the field of view is constantly changing. An object of remote sensing of the atmosphere is to enable reconstruction of the concentration distribution of trace species over an entire region based on the data available from the instrument.

Manuscript received September 9, 1980; revised September 22, 1980 and March 9, 1981. Paper recommended by A. Ephremides, Past Chairman of the Estimation Committee. This work was supported by NASA under Research Grant NAG-1-71.

M. Koda is with the Department of Chemical Engineering, California Institute of Technology, Pasadena, CA, on leave from the Department of Aeronautics, University of Tokyo, Hongo, Bunkyo-ku, Tokyo 113, Japan.

J. H. Seinfeld is with the Department of Chemical Engineering, California Institute of Technology, Pasadena, CA 91125.

The reconstruction of a concentration distribution from spatially averaged and possibly noise-corrupted data is a central problem in processing remote sensing data. In the absence of a mathematical model describing the spatial and temporal concentration distributions, the reconstruction can be carried out by standard data interpolation methods. However, when a mathematical model exists, the problem becomes one of matching the remote sensing data to the model solution in such a way that the incomplete data can be used in conjunction with the model to produce an estimate of the region-wide concentration distribution. This problem of the matching or assimilation of remote sensing data into mathematical models for atmospheric constituents is the subject of this paper.

There exist a few recent studies that assess the capabilities of remote sensing for monitoring regional air pollution episodes. For example, Barnes *et al.* [1] conducted a comparative analysis of satellite visible channel imagery in ground-based aerosol measurements. For three cases, each of which represented a significant pollution episode based on low surface visibility and high sulfate levels, the results show that the extent and transport of the haze pattern can be monitored from satellite data. The study demonstrated the potential of the satellite to monitor both magnitude and aerial extent of pollution episodes. In a related study, Lyons *et al.* [2] reported on a demonstration project showing that currently available synchronous satellite data can detect the aerial extent of large-scale hazy air masses associated with sulfate and ozone episodes.

A study related to that of the present work was reported by Diamonte *et al.* [3] in which they considered the comparison of remote and in situ data on pollutant concentrations from point sources. They considered typical remote sensing geometries to provide insight on estimation of plume properties from these measurements. In a study also related to the present, Kibbler and Suttles [4] considered the estimation of unknown parameters in a pollutant dispersion model by comparing model predictions with remotely sensed air-quality data. A ground-based sensor provided relative pollutant concentration measurements as a function of space and time. The measured data were compared with the dispersion model output through a numerical estimation procedure to yield parameter estimates that best fit the data.