[13] A. Segall, "The modeling of adaptive routing in data communication networks," this issue, pp. 85–95.
[14] M. Bello, S. M. thesis, Dep. Elec. Eng. and Comput. Sci., Massachusetts Inst. Technol., Cambridge, Sept. 1976.
[15] F. R. Gantmacher, *Matrix Theory*, vol. 2. New York: Chelsea, 1959.
[16] D. G. Luenberger, *Introduction to Linear and Nonlinear Programming*. Reading, MA: Addison Wesley, 1973.

★

Robert G. Gallager (S'58–M'61–F'68) was born in Philadelphia, PA on May 29, 1931. He received the S.B. degree in electrical engineering from the University of Pennsylvania, Philadelphia, in 1953 and the S.M. and Sc.D. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, in 1957 and 1960, respectively.

From 1953 to 1954 he was a member of the technical staff at Bell Laboratories and from 1954 to 1956 was in the signal corps of the U.S. Army. He has been at the Massachusetts Institute of Technology since 1956 and was Associate Chairman of the Faculty from 1973 to 1975. He is currently a Professor of Electrical Engineering and Computer Science and is the Associate Director of the Electronic Systems Laboratory. He is also a consultant to Codex Corporation, Newton, MA. He is the author of the text book *Information Theory and Reliable Communication* (New York: Wiley, 1968), and was awarded the IEEE Baker Prize Paper Award in 1966 for the paper "A Simple Derivation of the Coding Theorem and Some Applications."

Mr. Gallager was a member of the Administrative Committee of the IEEE group on Information Theory, from 1965 to 1970 and was Chairman of the group in 1971. His major research interests are data networks, information theory, and computer architecture.

The Modeling of Adaptive Routing in Data-Communication Networks

ADRIAN SEGALL, MEMBER, IEEE

Abstract—**Basic analytical models for problems of dynamic and quasi-static routing in data-communication networks are introduced. The models are intended to handle the quantities of interest in an algorithmic form, and as such require only a minimal number of assumptions. Control and estimation methods are used to construct algorithms for the solution of the routing problem.**

I. INTRODUCTION

THE problem of obtaining efficient routing procedures for fast delivery of messages to their destinations is of utmost importance in the design of modern data-communication networks. Although a large variety of routing algorithms have been developed and implemented in many existing networks, a lack of basic models and theories able to analyze a large variety of routing procedures has made it necessary to base these algorithms almost solely on intuition, heuristics, and simulation. It is the purpose of this paper to present several analytical models for various types of routing strategies and to indicate methods to analyze their performance.

For the purpose of this paper, we classify the routing procedures according to how dynamic they are, with the ends of the scale consisting of purely static and completely dynamic strategies. In a *purely static* situation, given fractions of the traffic at a node i of the network destined for each of the other nodes $j \neq i$ are directed on each of the links outgoing from node i. These fractions are decided upon before the network starts operating, are *fixed* in time, and depend only on the time and ensemble averages of the message flow requirements in the network. At the other end of the scale is the *completely dynamic* strategy which allows continuous changing of the routes. In particular, the routes can be varied not only as functions of time, but also according to congestion and traffic requirement changes in various portions of the network.

One can immediately see some of the advantages and drawbacks of each of the extreme strategies, but probably the most prominent ones are the following. The static routing procedure is relatively simple to implement, but on the other hand, if links or nodes in the network fail or build congestion, the messages intended to be transmitted over them will be completely blocked. The completely dynamic procedure is supposed to cope with the congestion and failure problems, but on the other hand, it requires large amounts of overhead per message for purposes of addressing, reordering at destinations, etc.

Given the advantages and disadvantages of the two extreme routing procedures, one should try in many practical situations to devise strategies that can possibly have some of the desired properties of both. One possibility is to use a *quasi-static* routing procedure, in which changes of routes will be allowed only at given intervals of time and/or whenever extreme situations occur. The time intervals between routing changes will be relatively long, so that most of the time messages will

Manuscript received March 10, 1976; revised June 30, 1976. This paper was presented at the National Telecommunications Conference, New Orleans, LA, December 1975. This work was supported by the Advanced Research Projects Agency of the Department of Defense (monitored by ONR) under Contract N00014-75-C-1183 and by the National Science Foundation under Grant ENG75-14103. Part of this work was performed at Codex Corporation, Newton, MA 02195.

The author was with the Electronic Systems Laboratory, Research Laboratory of Electronics and the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02195. He is now with the Department of Electrical Engineering, Technion IIT, Haifa, Israel.

be delivered in order and will not need individual addressing, but on the other hand, if a link fails or recovers or if the traffic and delays build up in a particular section of the network, the routing will be changed accordingly.

The best known existing analytical model for routing problems in data-communication networks has been introduced by Kleinrock [1] and addresses the *fixed* routing strategy. We describe it briefly here for reference and comparison with the models introduced in this paper.

1) The messages arrive from outside the network to the nodes according to independent constant rate Poisson processes, and their lengths are assumed to be independent exponentially distributed and independent of the arrival times.

2) At subsequent nodes along the paths of the messages, the lengths of the messages and their interarrival times become dependent, a fact which makes the analysis extremely difficult. To cope with this problem, the famous *"independence assumption"* is introduced, requiring the messages to "lose their identity" at each node and be assigned new independent lengths.

3) Once a message arrives at a node, it is assigned to one of the outgoing lines and waits there in queue until it is sent.

4) Based on $M/M/1$ queue analysis, the delay in *steady state* in each link is *calculated explicitly* as

$$D_{lm} = \frac{f_{lm}}{C_{lm} - f_{lm}} \tag{1a}$$

where

f_{lm} is the flow in link (l,m) in messages/second
C_{lm} is the capacity of link (l,m) in messages/second
D_{lm} is the total delay/unit time experienced by all messages in link (l,m).

5) A routing procedure is obtained to minimize the total delay over the entire network (see [2]):

$$D_t = \sum D_{lm}. \tag{1b}$$

The routing procedure is static, namely, constant in time and a function of the various rates and capacities only.

Since Kleinrock's model was proposed in 1962, researchers in the area have repeatedly expressed desires to find models that will be able to handle other types of routing procedures, in particular, dynamic strategies. Specifically, it is desirable to obtain models

a) in which various assumptions, and in particular, the *"independence assumption"* will not be needed,

b) that will be able to handle *transients and dynamic situations* and not only steady state,

c) that can provide *closed-loop* control strategies, namely, strategies that change dynamically according to the congestion in the various portions of the network.

The issues above are important in systems using completely dynamic routing, as well as those employing quasi-static strategies.

The purpose of the present paper is to introduce models for dynamic and quasi-static routing and present techniques to handle problems a), b), c) above. The models will be described in detail in Sections II-IV, but we may mention already that the main trend of the models is departure from the need of calculating the quantities of interest like delay, throughput, etc., *explicitly in closed form,* and rather developing models directed towards using efficient *algorithms* to handle these quantities. The main idea of the models is to use estimation and optimal control methods in the design of optimization algorithms. In fact, we may mention that the transition from closed-form solutions to algorithmic models has been repeatedly made before in many aspects of control and estimation theory: Kalman filters have replaced the Wiener filter, quadratic linear control replaced the classic root-locus methods, etc.

In Section II we introduce the basic model for dynamic routing, indicate the approach for solving it, and illustrate it by a simple example. Section III describes quasi-static routing, indicates what are the quantities of interest in such a procedure, and proposes schemes to estimate them. Models for analyzing the dynamic behavior of networks using quasi-static routing and for designing optimal feedback quasi-static strategies are introduced then in Section IV.

II. BASIC MODELS FOR DYNAMIC ROUTING

One of the first questions that has to be answered in the process of the analytical modeling of a system is how detailed does the description of the system have to be. The more microscopic the model is, presumably the more accuracy is achieved, but on the other hand, usually the harder it is to handle analytically. In a data-communication network, the usual approach is to look at each message or packet as an entity, and therefore the state of the network will be described by the number and destination of packets (say) in each of the buffers. The difficulty with this approach is that the number of states becomes immense even for the very simplest networks; in a network with N nodes, α outgoing links per node in the average, and buffers of maximum capacity of m packets, the number of states is approximately

$$(\alpha N m)^{N-1}. \tag{2}$$

The way to circumvent this difficulty is to realize that, in fact, any individual packet contributes very little to the overall behavior of the network, so that it is not really necessary to look separately at each of these packets and their lengths (does it really make much difference from the network point of view if at some instant there are 8 or 9 packets in a particular buffer?). It therefore makes sense to regard the network in a more macroscopic way and this is one of the main features of the model described below.

A. The Basic Model

Consider a data-communication network consisting of N nodes and let the collection of nodes $\{1,2,\cdots,N\}$ be denoted

by N. Let L be the collection of all links in the network (all lines are taken to be unidirectional):

$$L = \{(i,k), \text{ such that } i, k \in N \text{ and there is a direct link connecting } i \text{ to } k\}, \tag{3}$$

and for every $i \in N$, denote

$$E(i) = \text{collection of nodes } k \text{ such that } (i,k) \in L$$
$$I(i) = \text{collection of nodes } l \text{ such that } (l,i) \in L. \tag{4}$$

Let us imagine that at each node $i \in N$ we have $N-1$ boxes in which at every time t we store messages, packets, bits, etc., whose destination is $1, 2, \cdots, (i-1), (i+1), \cdots, N$, respectively, disregarding their origin. We call them boxes rather than queues, since any queueing discipline can be accommodated by the model without changing the equations. The number of messages of each type at any time t is measured in some arbitrary units, but we assume that the units are such that after appropriate normalization, the contents of the boxes can be approximated by a continuous variable, called "amount of traffic." Let

$x_i^j(t)$ = amount of traffic at node i at time t whose destination is node j where $i, j, \in N, j \neq i$

$r_i^j(t)$ = instantaneous rate of traffic at time t with destination j arriving at node i from outside the network

C_{ik} = capacity of link (i,k) in units of traffic/unit time where $(i,k) \in L$.

We should point out that in contrast to previous models, the "queues" here are being associated with the *nodes* rather than with the *links*.

Each link diverging from a node i is shared by some or all of the up to $(N-1)$ types of traffic stored at time t at node i (we use the term *type* to indicate *destination*). The problem can be stated now as: *at every time t, given the knowledge of the network congestion* $\{x_i^j(t),\ i, j \in N, j \neq i\}$, *dynamically decide what portion of each link to use for each type of traffic, so that the total delay will be minimized.*

We now give the equations describing the problem.

Dynamics: Let $u_{ik}^j(t)$ be the portion of the capacity of the link (i,k) used at time t for traffic of type j. Then the rate of change of the contents of each box is given by

$$\dot{x}_i^j(t) = r_i^j(t) - \sum_{k \in E(i)} u_{ik}^j(t) + \sum_{\substack{l \in I(i) \\ l \neq j}} u_{li}^j(t), \qquad i,j \in N, \quad j \neq i. \tag{5}$$

We may mention that, in contradistinction to the finite-state model, where the number of states is given in (2), the continuous state model (5) has at most $N(N-1)$ states where N is the number of nodes in the network. This number can sometimes be further reduced if for physical reasons one knows that traffic destined to particular nodes will never arrive at a given node.

Constraints: We clearly have positivity constraints

$$x_i^j(t) \geqslant 0 \tag{6a}$$
$$u_{ik}^j(t) \geqslant 0 \tag{6b}$$

and capacity constraints

$$\sum_{j \neq i} u_{ik}^j(t) \leqslant C_{ik}, \qquad (i,k) \in L, \quad j \in N. \tag{7}$$

In later versions of the model, more constraints may be introduced, for example, to take into account the capacities of the buffers.

Cost Functional: As mentioned in the discussion at the end of Section I, one of the major difficulties of the queueing models is that they require explicit closed-form expressions for the average delays, and these can be found analytically only for very special distributions and dependence relationships that do not always agree with reality. On the other hand, algorithmic-oriented models require only an expression for the delay in terms of the state variables and controls describing the problem. For example, observe that if $x(t)$ is the amount of traffic in some buffer at time t, then the quantity

$$\int_{t_0}^{t_f} x(t)\, dt \tag{8}$$

gives the total time spent in this buffer by the traffic that passed through it during some period of interest $[t_0, t_f]$ when t_f is such that $x(t_f) = 0$. Consequently, expression (8) is exactly the total delay experienced in this buffer, and therefore the total delay in the network during $[t_0, t_f]$ is given by

$$D = \int_{t_0}^{t_f} \left[\sum_{i \neq j} x_i^j(t) \right] dt \tag{9}$$

where t_f is some time at which all boxes are empty. Priorities can also be easily incorporated in the criterion (9) by giving nonequal weights α_{ij} to the appropriate x's, so that the cost functional will be

$$J = \int_{t_0}^{t_f} \sum_{i \neq j} \alpha_{ij} x_i^j(t)\, dt. \tag{10}$$

Optimization Problem: Let us denote by $\boldsymbol{x}(t)$ the vector of all states $x_i^j(t)$ and by $\boldsymbol{u}(t)$ the vector of all controls $u_{ik}^j(t)$. The problem is then to find the set of controls $\boldsymbol{u}$ as a function of time and state

$$\boldsymbol{u}(t) \equiv \boldsymbol{u}(t, \boldsymbol{x}(t)) \tag{11}$$

that will bring the state from $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$ (given) to $\boldsymbol{x}(t_f) = \boldsymbol{0}$ while minimizing the criterion (10) subject to control and state inequality constraints (6), (7), and to the dynamics (5).

B. Solutions to the Optimal Control Problem

There are various ways in which the optimal control problem stated in Section II-A can be attacked. First, one can observe that the model is linear in all variables with linear cost functional, so that after discretizing it in time, it reduces to the solution of a linear programming problem. Its solution will provide a sequence of $\boldsymbol{u}$'s that will bring the state from a given $\boldsymbol{x}(t_0)$ to $\mathbf{0}$ while minimizing the cost (10). However, this brute-force method has many disadvantages; the linear program will be of very large dimensions, this method provides only a nonfeedback (although dynamic) strategy and therefore a new linear program has to be solved for each initial condition, and also the insight obtained from the solution will be minimal. Nonfeedback solutions, in general, are worth considering, however, because they require much less storage than the closed-loop one, and various possible methods are presently under investigation [4]. Our presentation here will be restricted, however, to methods for obtaining feedback solutions, namely, those in which the optimal routing variables are given as a function of the present state of the network $\boldsymbol{x}(t)$.

Feedback solutions are provided by the minimum principle of Pontryagin [3] requiring the formation of the Hamiltonian

$$H(\boldsymbol{x},\boldsymbol{u},\lambda,\mu,t) = \sum_{i\neq j} \lambda_{ij}\dot{x}_i{}^j + \sum_{i\neq j} \mu_{ij}x_i{}^j + \sum_{i\neq j} \alpha_{ij}x_i{}^j \tag{12}$$

where $\{\lambda_{ij}\}$ is the set of costates, $\{\mu_{ij}\}$ is a set of Lagrange multipliers to take into account the state inequality constraints $x_i{}^j(t) \geq 0$ of (6), and $\dot{x}_i{}^j$ is substituted from (5). For readers not familiar with the maximum principle, we may mention that the Hamiltonian is nothing but the dynamic counterpart of the Lagrangian formed for static constrained minimization problems, and λ, μ, α are the appropriate Lagrange multipliers. The costates satisfy [3]

$$-\dot{\lambda}_{ij}(t) = \frac{\partial H}{\partial x_i{}^j} = \alpha_{ij} + \mu_{ij}(t), \qquad \lambda_{ij}(t_f) = 0 \tag{13}$$

and the μ_{ij} are such that they satisfy the complementary slackness conditions

$$\mu_{ij}(t)x_i{}^j(t) = 0, \qquad \mu_{ij}(t) \leq 0. \tag{14}$$

The optimal control $\boldsymbol{u}^*(t)$ is then given by the following.

Theorem 1: A necessary and sufficient condition for the control law $\boldsymbol{u}^* \in U$ optimal is that for every t and for some λ, μ, $\boldsymbol{x}$ satisfying (5), (6a), and (13) and (14), it minimizes the Hamiltonian H namely,

$$H(\boldsymbol{x},\boldsymbol{u}^*,\lambda,\mu,t) = \min_{\boldsymbol{u}\in U} H(\boldsymbol{x},\boldsymbol{u},\lambda,\mu,t) \tag{15}$$

where U is the set of allowable controls given by

$$U = \left\{\boldsymbol{u}: u_{ik}{}^j \geq 0, \sum_{j\neq i} u_{ik}{}^j \leq C_{ij}\right\}. \tag{16}$$

The necessity of the minimum principle (15) is classical [3]. It turns out that because of the linearity of the problem, (15) is also sufficient, and this fact is proved in [4]. The sufficiency of the minimum principle is important, since it assures that if a path can be found from $\boldsymbol{x}(t_0)$ to $\boldsymbol{x}(t_f) = \mathbf{0}$ such that (15) is satisfied with *any* set of costates and Lagrange multipliers λ, μ with properties (13), (14), then this path is *optimal.* Therefore the solution of (15) will provide not only a local minimum, but also a global one.

Now observe that since the last two summations of (12) are independent of $\boldsymbol{u}$ and so are the terms containing the inputs r when substituting $\dot{x}$ from (5) into (12), minimizing the Hamiltonian (12) is equivalent to minimizing over $\boldsymbol{u} \in U$ the quantity

$$\hat{H} = \sum_{i\neq j} \lambda_{ij}(t)\left[-\sum_{k\in E(i)} u_{ik}{}^j(t) + \sum_{\substack{l\in I(i)\\ l\neq j}} u_{li}{}^j(t)\right]. \tag{17}$$

Now observe that to minimize (17) with respect to $\boldsymbol{u}$ subject to the constraints (6b), (7) is again a linear programming problem (of very small dimensionality), provided that we know the costates $\lambda(t)$. The problem is that these costates are given by (13), which is a differential equation running *backward,* whereas (5) are running *forward,* so that a two-point boundary value problem has to be solved. Its solutions will indeed provide feedback strategies because the λ's are dependent on the $\boldsymbol{x}$'s through (13) and (14), which makes the solution $\boldsymbol{u}$ of (17) $\boldsymbol{x}$-dependent. The usual technique of solving two-point boundary value problems is to guess some values of $\lambda_{ij}(t_0)$, run (13) forward while satisfying (14) and minimizing (17), and see whether the $\lambda_{ij}(t_f)$ that are obtained are all 0 as required by (13). If yes, the problem is solved, and if not, one finds an algorithm to correct the guess and repeats the procedure until it converges. It is clear that such an algorithm would be quite cumbersome to our problem, and also that, after all these iterations, one only obtains the solution for states along the obtained path. If another initial state has to be considered, the procedure must be repeated. It is exactly this point that makes it difficult to obtain solutions of general two-point boundary value and optimal control problems. The problem considered here, however, is linear and convex, and by heavily exploiting these facts, it is possible to construct an algorithm which, using a relatively small number of linear programs of dimensions not larger than the number of control variables, will construct the feedback control law. The main idea of the algorithm is that, by solving one linear program, it is possible, because of the linearity and convexity of the problem, to obtain the control law for an entire conical portion of the state space, as will be illustrated in the example of Section II-C.

C. The Backward Construction of the Control

We shall consider here only the case for which there are no inputs ($r_i{}^j(t) = 0$) over the period of interest $t \in [t_0, t_f]$ and we also take all weights in (10) to be unity:

$$\alpha_{ij} = 1. \tag{18}$$

The goal is then to deliver the existing traffic in the network $\{x_i^j(t_0)$ for all $i, j \in N, i \neq j\}$ to its destination in such a way as to minimize total delay (9). In the case of no external inputs, the states x_i^j will deplete from their initial values and sequentially empty out (contact the boundary $x_i^j = 0$) until at t_f all the states are zero. It is certainly possible for optimality to dictate that two or more states reach the boundary simultaneously or that some states will first increase and only then empty out.

Suppose, in the spirit of dynamic programming [5], we were to view the trajectories backward in time beginning at t_f. We would then see a sequence of states leaving the boundary (perhaps two or more at a time) and varying with piecewise constant slopes. Since we know the value of the costates at t_f, we can construct backward in time the costates associated with those states off the boundary ($\mu_{ij} = 0$). We then have a sequence of linear programs (17), (16) with known coefficients corresponding to each time a state (or states) leaves the boundary. By solving a comprehensive set of such programs corresponding to all possible combinations of states leaving the boundary, we may construct the optimal feedback control law $\boldsymbol{u}^*(\boldsymbol{x})$ on the entire $N(N-1)$-dimensional $\boldsymbol{x}$-space. This technique will be illustrated in the Appendix by investigating the simple network of Fig. 1. The main point to observe is that the optimal routing strategy for each of the regions in Fig. 2 is obtained by just solving one small linear program, and this will be the case for arbitrary networks. In addition, the regions in the $\boldsymbol{x}$-space within which the same control is optimal are always separated by *hyperplanes* passing through the origin, a fact which makes both their computation and the decision on the value of the control to be used during the operation of the network relatively simple (checking several linear inequalities). Moreover, since the controls are "bang-bang", i.e., go from one extreme point of the $\boldsymbol{u}$-space to another, the areas underneath the functions $x_i^j(t)$ are always trapezoidal, so that the value of the optimal cost J in (10) is always quadratic in the initial condition $\boldsymbol{x}(t_0)$.

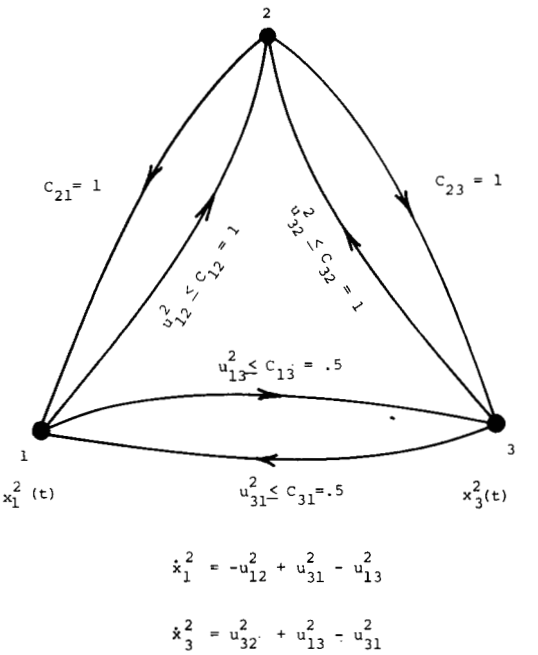


Fig. 1. An example of a data-communication network.

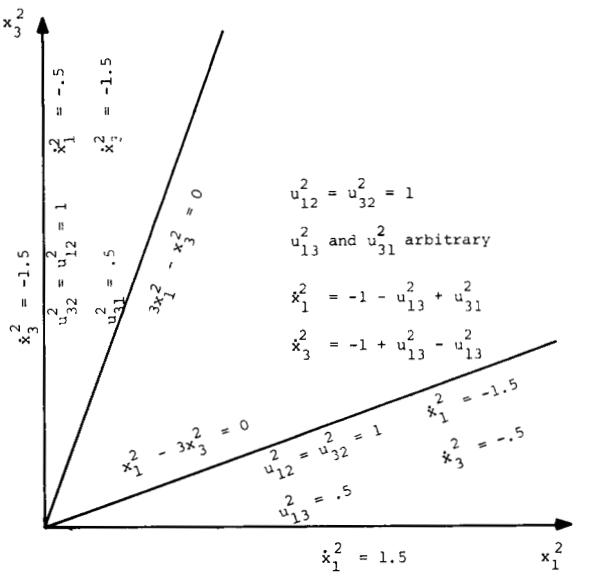


Fig. 2. Optimal controls corresponding to $x_1^2 \geq 0, x_3^2 \geq 0$ and all other states zero.

D. Stochastic Inputs

In an actual network, the inputs are, of course, stochastic. One approximation is to take into account only the (ensemble) average rates of the inputs $[r_i^j(t)$ in (5)], find the optimal controls using the deterministic methods of Sections II-A, B, and C, and use these controls in the operation of the network. This may or may not work, depending essentially on the variance and the higher moments of the actual inputs. For this reason it is of interest to investigate possible stochastic control schemes. When the inputs are stochastic, (5) becomes

$$dx_i^j(t) = \left[-\sum u_{ik}^j(t) + \sum u_{li}^j(t) \right] dt + dY_i^j(t)$$

$$\equiv -g_i^j(\boldsymbol{u})dt + dY_i^j(t) \tag{19}$$

where $Y_i^j(t)$ is the process describing the traffic with destination j arriving at node i from outside the network. Clearly, the processes $Y_i^j(t)$ must have nondecreasing paths and usually can be modeled as mutually independent jump processes with positive jumps only. (It is very tempting for control theoreticians to model Y_i^j as Gaussian processes, but this cannot be done here, for instance, because Y_i^j must have nondecreasing paths.) The time of the jumps corresponds to the time of message arrivals and their size to the message lengths. Such a process $Y_i^j(t)$ is characterized (see [7]) by the quantities $\Pi_{ij}(t,y)$ defined for every $y \geq 0$ *as the rate of arrival at time t of messages of size less than or equal to y given the past history of all processes of interest.* For example, one can take, as is usually done in the literature, Y_i^j to be a compound Poisson process, in which case $\Pi_{ij}(t,y)$ is deterministic given by

$$\Pi_{ij}(t,y) = f_{ij}(t) \cdot B_{ij}(y) \tag{20}$$

where $B_{ij}(\cdot)$ is the distribution of message lengths and $f_{ij}(\cdot)$

is the message arrival rate. It is important, however, if possible, to allow $\Pi(\cdot,y)$ to be random, i.e., dependent on the past and present development in the network. For example, if the network has finite buffers, then one has to refuse traffic of a size larger than the difference between the buffer size and its present contents. This makes the $\Pi_{ij}(t,dy) = 0$ at the appropriate size y, and this says essentially that $\Pi_{ij}(t,y)$ is, in fact, $x_i^j(t)$-dependent:

$$\Pi_{ij}(t,y) = \Pi_{ij}(t,y,x_i^j(t)). \tag{21}$$

The cost to be minimized can be taken as

$$K = E\left\{\int_0^T \sum_{i,j} \alpha_{ij} x_i^j(t)dt + J(\boldsymbol{x}(T))\right\} \tag{22}$$

where $J(\boldsymbol{x}(T))$ is the value of the cost J of (10) with T identified with t_0 and $\boldsymbol{x}(T)$ being the initial condition in the problem considered in Sections II-A, B, and C. Equation (22) includes the expected total delay during the period of operation $[0,T]$ plus the extra delay one has to incur to get rid of the traffic left in the system at time T. We are now interested in the optimal control of the form (11) that will minimize K subject to dynamics (19) and constraints (6) and (7). A sufficient condition for a control $\boldsymbol{u}^*$ to be optimal is provided by the following.

Theorem 2: Let U be the set of all admissible controls, i.e., feedback controls satisfying the constraints (6b), (7). Suppose there exists a control $\boldsymbol{u}^* \in U$ and a function $V(\boldsymbol{x},t)$ such that for all t and $\boldsymbol{x}$ and for any other control $\boldsymbol{u} \in U$, we have

$$\begin{aligned} 0 = \frac{\partial V}{\partial t} &- \sum_{i,j} g_i^j(\boldsymbol{u}^*) \frac{\partial V}{\partial x_i^j} + \sum_{i,j} \alpha_{ij} x_i^j \\ &+ \sum_{i,j} \int_0^\infty \left[V(x_i^j + z_i^j, t) - V(\boldsymbol{x},t) \right] \Pi_{ij}(t,dz_i^j) \\ \leqslant \frac{\partial V}{\partial t} &- \sum_{i,j} g_i^j(\boldsymbol{u}) \frac{\partial V}{\partial x_i^j} + \sum_{i,j} \alpha_{ij} x_i^j \\ &+ \sum_{i,j} \int_0^\infty \left[V(x_i^j + z_i^j, t) - V(\boldsymbol{x},t) \right] \Pi_{ij}(t,dz_i^j) \end{aligned} \tag{23a}$$

with terminal condition

$$V(\boldsymbol{x},T) = J(\boldsymbol{x}). \tag{23b}$$

(The integrals in (23a) are ordinary Stieltjes integrals.) Then $\boldsymbol{u}^*$ is optimal in U and $V(\boldsymbol{x},0)$ gives the optimal cost (22) whenever $\boldsymbol{x}(0) = \boldsymbol{x}$. Here $V(x_i^j + z_i^j)$ says that one adds z_i^j only to the x_i^j component of $\boldsymbol{x}$, while keeping all the others fixed.

To limit the length of the paper, we omit the proof of Theorem 2 and indicate only that the proof follows essentially the same lines as in [8] and [9]. The function $V(\boldsymbol{x},t)$ will then provide the expected remaining cost at time t given that the network is in state $\boldsymbol{x}(t) = \boldsymbol{x}$. In principle, $V(\boldsymbol{x},t)$ and $u(\boldsymbol{x},t)$ must be found for every $\boldsymbol{x}$ and t by solving (23) backward in time, while also minimizing its right-hand side. However, solution by numerical integration is out of the question here because it necessitates a computation at each point of a grid over the $\boldsymbol{x}$-space, so that alternate solutions are being presently investigated.

III. QUASI-STATIC ROUTING

As mentioned in the Introduction, in many data-communication networks it will be advisable to use a quasi-static routing procedure, under which the portions of traffic sent by each node to each destination on each outgoing link will remain constant over relatively long time periods. At preassigned times and/or when need arises, the routes can be changed to improve the overall network performance, which in the present paper is taken to be *average delay*. Regarding this procedure, we shall address and elaborate on the following issues.

1) Identify the quantities of interest for the procedure and indicate how to use them.

2) From the available measurements, estimate these quantities.

3) Analyze the behavior of the network under the quasi-static routing procedure.

A. Quantity of Interest—Incremental Delay

With the notation of (3) and (4), let $D_{ik}(f_{ik})$ be the *total delay* faced by all messages passing through link (i,k) per unit time. For simplicity, we assume here that for each link (i,k), the delay D_{ik} is a function only of the total flow f_{ik} passing through this link. Then the total delay over the network per unit time is

$$D_t = \sum_{(i,k)\in L} D_{ik}(f_{ik}). \tag{24}$$

If there are different types of traffic through a link so that the delay is a function of the individual rates and not only of the *total* rate, (24) can be easily changed accordingly.

Suppose now that some nominal flows $\bar{f}_{ik}$ exist in the network satisfying some nominal flow requirements $\{\bar{r}_i^j, i, j \in N, i \neq j\}$ and that the flow requirement from some node i to some destination j increases by an incremental amount δr_i^j. The question is what path should be chosen for this extra traffic. The classical answer to this question (say the one implemented in the ARPANET [10]) is to choose the path over which the total delay is minimal. Clearly, such a path will be the best for the extra traffic δr_i^j itself, but it disregards the fact that this choice may hurt everybody else, i.e., the existing traffic. If the quantity to be optimized is indeed the *average* delay [which is proportional to D_t in (24)], both effects must be taken into consideration: the delay incurred by the extra traffic itself, as well as the extra delay suffered by the existing traffic. This can be done if one observes that if one chooses a path P from i to j for the extra traffic, then from (24) the extra total delay δD_t will be (up to first order)

$$\delta D_t = \sum_{(l,m)\in P} \frac{\partial D_t}{\partial f_{lm}} \cdot \delta r_i{}^j = \sum_{(l,m)\in P} \frac{dD_{lm}}{df_{lm}} \cdot \delta r_i{}^j. \qquad (25)$$

Equation (25) essentially says that the flow in each of the links of the chosen path P will increase by $\delta r_i{}^j$.

The following decentralized routing algorithm is suggested by expression (25):

1) estimate the *incremental delay* over each link in the network (the estimation can be done locally and the procedure will be described presently) and

2) use these quantities to update the routing tables essentially in the same way the estimated *delay* is used in ARPANET [10].

Other strategies can be designed as well, depending on the particular network under design. In [11] a recursive algorithm has been proposed to divide the traffic in an optimal way over each of the outgoing links, so that the total delay will be minimized. Also, in a network controlled from a central site, the router can periodically collect the estimated incremental delays and use them to find the gradient of the delay whose projection on the flow requirement subspace will provide the steepest descent direction for change of the flows. Such a strategy has been proposed in [13]. No matter which of the strategies indicated above is used, the point is that all methods need as a basic quantity the *incremental delay* dD_{lm}/df_{lm}. One way to find it is using Kleinrock's formula (1), but as stated before, this expression involves a certain number of assumptions which one would like to avoid if possible. It is therefore of importance to estimate the incremental delay directly, thus reducing the dependence of the algorithms on various assumptions. In fact, it will be seen in the next subsection that the only necessary assumption for the estimation algorithms to make sense is stationarity over the intervals between routing changes.

Before describing the specific estimation procedures, we may also mention that the same kind of strategies as indicated above can be used in much the same way in other kinds of networks like transportation or air traffic nets. It is only the estimation procedure that will depend on the particular application and specific transmission mechanisms of the network. In fact, the two types of routing procedures described above have been considered before in the transportation literature. Minimizing each vehicle's own delay is referred to as user optimization or Wardrop's first principle, whereas optimizing the average delay is called system optimization or Wardrop's second principle [14].

B. Estimation of Incremental Delay

Once the quantity of interest has been identified and routing procedures using it have been indicated, the next problem is to find (i.e., estimate) this quantity. In the present subsection, we shall describe algorithms to estimate the incremental delay over a given link. Observe that the total delay over a link (which we now denote by D for notational convenience) is very easy to estimate by just adding up the individual delays of the messages. It is not so obvious, however, how to estimate the incremental delay $dD(f)/df$ *without actually increasing the flow.* The problem is to estimate $dD(f)/df$ over a given link from the traffic record of arrivals and departures measured over the interval between routing changes. In addition, it would be useful to have a procedure that will keep a current estimate and will recursively update it, so that it will not be necessary to memorize the entire record and to do all the computation at the end.

Clearly, the estimation procedure will depend on the mechanism that is used to combine the data at the nodes. In the following, we briefly describe recursive algorithms to obtain estimates of the incremental delay dD/df over a given link for two different mechanisms: 1) addressed packets or messages and 2) character multiplexing.

1) Incremental Delay for Packet or Message Switched Networks: Consider a network operating according to a packet or message switching strategy. One is interested then to find the effect on the arrival-departure record over a given link of a hypothetical increase (or decrease) δf in traffic flow rate. To reduce the effective traffic rate, one can, for example, use the following procedure which is similar to the so-called "jackknife" algorithm in statistics for estimating the variance of a distribution from a sequence of independent samples [15]: suppose, again hypothetically, that each packet arriving to the link would be accepted with probability $(1 - \epsilon)$ and rejected with probability ϵ independently from packet to packet. The effective rate will then be reduced in the average by

$$\delta f = \frac{M\epsilon}{T} \qquad (26)$$

where

T = period of interest over which the estimate is calculated
M = number of arrivals during T.

Also, the probability of removing two or more packets at a time is of order ϵ^2 and therefore of second order in δf, so that one only has to consider the effect of removing only one packet at a time. It is also easy to see that removing a packet from one given busy period has no effect on packets served in other busy periods, so that one only has to consider the effect of the removal on packets from the same busy period. For a given busy period, let $c_m{}^n$ be the amount of system time that the mth packet would save if the nth packet were to be removed. If we denote by d_n and a_n, respectively, the departure and arrival time of the nth packet relative to the beginning of the busy period, say, then one can easily obtain the following recursive formulas:

$$c_m{}^n = 0, \qquad \text{for } m < n$$

$$c_m{}^n = d_n - a_n, \qquad \text{for } m = n$$

$$c_{n+1}{}^n = d_n - \max(a_{n+1}, d_{n-1}) \text{ where we take } d_0 = 0 \qquad (27a)$$

and for $m > n + 1$

$$c_m{}^n = \min(c_{m-1}{}^n, d_{m-1} - a_m). \qquad (27b)$$

We assume here first-come first-serve discipline. We may mention that since the algorithm is recursive, very little memory is

necessary to implement it. Equations (27) hold, of course, for all packets from the same busy period. Over B busy periods, containing each $\{N_i,\ i = 1, \cdots, B\}$ packets, the total effect will therefore be

$$\delta D = \frac{1}{T} \sum_{i=1}^{B} \sum_{n=1}^{N_i} \sum_{m=n}^{N_i} \epsilon\ {c_m}^n \tag{28}$$

and the quantity $\delta D/\delta f$ will be

$$\frac{\delta D}{\delta f} = \frac{\sum_{i=1}^{B} \sum_{n=1}^{N_i} \sum_{m=n}^{N_i} {c_m}^n}{\sum_{i=1}^{B} N_i} \tag{29}$$

A slightly different algorithm will be to add a new hypothetical packet and calculate its influence on the total delay. This algorithm is described in detail in [12] where properties of the estimates–like bias and consistency–are also investigated. One can note here, however, that both algorithms require no assumptions whatsoever about the nature of the statistics or other parameters of the traffic, but rather use the measured arrival-departure record to estimate the quantities of interest. Also, since the estimation is done link by link, the procedure can be implemented locally in a completely decentralized manner.

2) Character Multiplexing: Suppose each link (l,k) operates with the following scheme: node l assembles into blocks all characters (disregarding the destination) that are supposed to go over the link (l,k), attaches error check bits and other necessary overhead characters, and sends the block over; node k checks for errors, disassembles the block, and forwards the characters to the subsequent links; node l has buffers where characters from each of the incoming links intended to go on link (l,k) are stored, and at the time is is ready to form a new block, it empties all buffers; the characters are sent in one block together with overhead, and when the transmission of the block is finished, the buffers are emptied again and a new block is formed. We assume that the overhead per block is constant and let

B = number of blocks observed,
n_i = length of ith block, both overhead and information characters (in characters), $i = 1, \cdots, N$
C = capacity of link (in characters/second)
ν = length of overhead (in characters)
f = average flow (in characters/second).

The problem is to estimate the first-order increase in total delay δD of the characters that are transmitted over the link caused by a hypothetical increase δf of the flow rate, given observations of $\{n_1, n_2, \cdots, n_B\}$ over some period of interest. Observe that with the present scheme, the average delay (on the link under consideration) of a character that enters the ith block is

$$\frac{1}{C}(\tfrac{1}{2}n_{i-1} + n_i) \tag{30}$$

where $n_0 = 0$. This is because such a character waits on the average $n_{i-1}/2C$ seconds before it is sampled and exactly n_i/C seconds until it is released at the other end. Therefore, since there are $(n_i - \nu)$ information characters in the ith block, the total delay per unit time over B blocks being sent during a period of length T is

$$D = \frac{1}{C} \sum_{i=1}^{B} (n_i - \nu)(\tfrac{1}{2}n_{i-1} + n_i). \tag{31}$$

Next, we analyze what would happen if we were to increase the flow f through the link by an incremental amount δf. Equation (31) shows that the change in the delay will be caused by changes δn_i in the block lengths; in turn the change δn_i in the length of the ith block will be caused both directly by the increase of the flow and indirectly by the change of the length of the $(i-1)$th block. On the average, therefore,

$$\delta n_i = \frac{n_{i-1}}{C}\,\delta f + \frac{f}{C}\,\delta n_{i-1} \tag{32}$$

and (32) and (31) suggest a recursive procedure to estimate the quantity of interest $\delta D/\delta f$:

$$\frac{\delta n_i}{\delta f} = \frac{n_{i-1}}{C} + \frac{f}{C}\frac{\delta n_{i-1}}{\delta f}, \qquad \frac{\delta n_0}{\delta f} = 0 \tag{33a}$$

$$\frac{\delta D}{\delta f} = \frac{1}{C} \sum_{i=1}^{B} \left[\frac{\delta n_i}{\delta f} \left(\frac{1}{2} n_{i-1} + n_i \right) + (n_i - \nu) \left(\frac{1}{2} \frac{\delta n_{i-1}}{\delta f} + \frac{\delta n_i}{\delta f} \right) \right]. \tag{33b}$$

One should observe that the quantities needed in the estimation procedure here are C, f, ν, and the measured data $\{n_0, n_1, \cdots, n_N\}$. Clearly, the procedure should be duplicated over each link, and the estimates be used in updating the routing tables as indicated in the beginning of this section. Observe, moreover, that here also there are no assumptions needed in the estimation procedure and that the estimation can be done in a completely decentralized way. Moreover, it is a recursive procedure so that the amount of memory needed to implement it is minimal.

IV. DYNAMIC ANALYSIS OF QUASI-STATIC ROUTING

In Section III we indicated various routing procedures for networks working in a quasi-static mode and proposed ways to obtain the incremental delay, which is the basic quantity used in those procedures. Another question that remains to be answered is what will be the dynamic behavior of the network under these quasi-static procedures. Specifically, due to finite propagation time of routing changes, the network will not respond instantaneously to these changes, but rather in a dynamic way. Moreover, dynamically changing input rates will have

certain effects on the behavior of the network. One would like then to have models and analysis techniques that allow investigating *basic* questions about the behavior of the network, like dynamic stability and time constants, ability to control and "observe" its dynamics (controllability and observability in system theoretic language), and the possibility to improve the routing procedures that were proposed before on an intuitive basis to obtain "good" system behavior. For completeness we propose here such a dynamic model for quasi-static routing whose analysis is, however, only in early stages, and hence it is left to subsequent presentations.

Clearly, the systems under consideration—data-communication networks— are highly nonlinear and system theory is not developed enough at this stage to efficiently handle such systems. What we shall use here, therefore, is a common technique in system theory, and this is to linearize the system around a nominal operation point. The linearized model will describe then the fluctuations of the network around such a point, thereby indicating its stability or instability and its approximate behavior. Moreover, it is important to realize that since a quasi-static routing procedure is under consideration here, rather than a completely dynamic one as in Section II, a much more macroscopic description of the system is necessary. For example, we shall now disregard the instantaneous contents of the queues, and use only the flows through the queues as our basic variables.

To obtain the dynamic model for quasi-static routing, consider first a network operating in steady state at a nominal point and let

$$\begin{aligned} F_i^j &= \text{total rate of traffic destined for node } j \text{ at node } i \\ \phi_{ik}^j &= \text{fraction of the traffic } F_i^j \text{ sent over link } (i,k) \\ f_{ik}^j &= \text{rate of traffic destined for } j \text{ over link } (i,k) \text{ and} \\ r_i^j &= \text{input flow at node } i \text{ of traffic destined for } j. \end{aligned} \tag{34}$$

Traffic equilibrium gives for nominal values (denoted by overbars)

$$\bar{f}_{ik}^j = \bar{\phi}_{ik}^j \cdot \bar{F}_i^j \tag{35a}$$

$$\bar{F}_i^j = \sum_{\substack{l \in I(i) \\ l \neq j}} \bar{f}_{li}^j + \bar{r}_i^j = \sum_{\substack{l \in I(i) \\ l \neq j}} \bar{\phi}_{li}^j \bar{F}_i^j + \bar{r}_i^j \tag{35b}$$

$$\sum_{k \in E(i)} \bar{\phi}_{ik}^j = 1 \tag{35c}$$

where $I(i)$ and $E(i)$ are defined in (3).

We are now interested in small dynamic changes δF_i^j, etc., around these nominal values. The time unit will be taken here as the time it takes for a change of routing at some node to be felt at the adjacent nodes in the network. We assume this quantity to be fixed throughout the network. Assuming also that changes in input rates are felt immediatly at the source node, the dynamic behavior of the network is described by the following dynamic equations [obtained from (35b) and (35c)]:

$$\begin{aligned} \delta F_i^j(t+1) = &\sum_{\substack{l \in I(i) \\ l \neq j}} \bar{\phi}_{li}^j \delta F_l^j(t) + \\ &+ \sum_{\substack{l \in I(i) \\ l \neq j}} \bar{F}_l^j \delta \phi_{li}^j(t) + \delta r_i^j(t+1), \\ &i, j = 1, \cdots, N \end{aligned} \tag{36a}$$

$$\sum_{k \in E(i)} \delta \phi_{ik}^j(t) = 0. \tag{36b}$$

The above relationship essentially says that it takes one time unit for changes occuring at node l, say, to be felt at an adjacent node i, but changes in the input rate at node i influence the flow there immediately. The system-theoretic interpretation of the various quantities in (36) is clear: as said before, the situation in the network is described now by the set of node flows, so that $\delta F_i^j(t)$ are the states, the fractions of traffic over each outgoing link are the quantities that control the network, therefore $\delta \phi_{ik}^j(t)$ represent the controls, and the stochastic fluctuations $\delta r_i^j(t)$ in the inputs represent the "noise."

Several other versions of model (36) can also be considered. For example, instead of the changes occuring at one node "being felt" at subsequent nodes after exactly one time unit, one can have a "smoother" description by modeling each node i as a first-order Markov system with time constant α_i^j. Then the model (36) becomes

$$\begin{aligned} \frac{1}{\alpha_i^j} \cdot \frac{d}{dt} (\delta F_i^j)) = \delta F_i^j(t) + \sum_{\substack{l \in I(i) \\ l \neq j}} \bar{\phi}_{li}^j(t) \\ \sum_{\substack{l \in I(i) \\ l \neq j}} \bar{F}_l^j \delta \phi_{li}^j(t) + \delta r_i^j(t) \end{aligned} \tag{37}$$

where now $t \in [t_0, \infty)$. In this model, the time constants α_i^j will, of course, depend on the nominal point around which we are working and will have to be determined by measurements or simulation.

There are various analytically attractive features of the above model. It is a linear stochastic model and such systems have been widely studied in the literature. One can therefore hope that its analysis will give much insight in the dynamic behavior of networks. Even more, the equations corresponding to different destinations are completely decoupled, so that they reduce to the study of N systems, each with $(N-1)$ states. This tremendous reduction of dimension will be of much help in the analysis of the model.

V. CONCLUSIONS

New models for dynamic and quasi-static routing have been introduced in this paper. These models have the property that they do not require explicit closed-form expressions for the

quantities of interest, but rather are intended to handle them in an algorithmic form. Optimal control and estimation methods to construct efficient algorithms for the problems of interest have been indicated.

APPENDIX

CONSTRUCTION OF OPTIMAL ROUTING STATEGY FOR A SIMPLE EXAMPLE

Consider the network of Fig. 1 with indicated capacities. Suppose x_1^2 leaves the boundary last (first backward in time), and therefore the Hamiltonian along that segment is

$$\lambda_{12}x_1^2 = \lambda_{12}(-u_{12}^2 + u_{31}^2 - u_{13}^2). \tag{A1}$$

This is minimized at u^* (12) defined by

$$u^*(12) = \{u_{12}^2 = 1, u_{31}^2 = 0, u_{13}^2 = 0.5\} \tag{A2}$$

and the requirement that $x_3^2 = x_3^2 = 0$ implies $u_{32}^2 = 0.5$. We then know that along the x_1^2 axis of Fig. 2, x_1^2 travels at the rate $x_1^2 = -1.5$. We may similarly determine the optimal control on the x_3^2 axis by allowing x_3^2 to leave last.

Next, we allow x_1^2 to leave last and x_3^2 beforehand. The Hamiltonian is now

$$\lambda_{12}(-u_{12}^2 + u_{31}^2 - u_{13}^2) + \lambda_{32}(-u_{32}^2 - u_{31}^2) \tag{A3}$$

and is minimized at u^* (32, 12) defined by

$$u^*(32, 12) = \{u_{12}^2 = u_{32}^2 = 1, u_{13}^2 = 0.5, u_{31}^2 = 0\} \tag{A4}$$

since $\lambda_{32} < \lambda_{12}$. Therefore, x_1^2 and x_3^2 travel at rates $\dot{x}_1^2 = -1.5$, $\dot{x}_3^2 = -0.5$, namely, parallel to the line $x_1^2 - 3x_3^2 = 0$.

This says that within the region of the plane (x_1^2, x_3^2) between this line and the x_1^2 axis, the optimal control is u^* (32, 12). We may similarly determine the optimal control between the x_3^2 axis and the line $3x_1^2 - x_3^2 = 0$ by allowing x_3^2 to leave last and x_1^2 beforehand.

Finally, we allow x_1^2 and x_3^2 to leave the boundary simultaneously. The Hamiltonian

$$\lambda_{12}(-u_{12}^2 + u_{31}^2 - u_{13}^2) + \lambda_{32}(-u_{32}^2 + u_{13}^2 - u_{31}^2) \tag{A5}$$

is minimized over the infinitely nonunique set u^* (12 simult. 32) defined by

$$u^*(12 \text{ simult. } 32) = \{u_{12}^2 = u_{32}^2 = 1, u_{13}^2 = [0, 0.5], \; u_{31}^2 = [0, 0.5]\}$$

since $\lambda_{12} = \lambda_{32}$. This allows us to stipulate the nonunique control in the region lying between $3x_1^2 - x_3^2 = 0$ and $x_1^2 - 3x_3^2 = 0$, thus completing the characterization of the two-dimensional space x_1^2, x_3^2. The procedure is then continued until the entire six-dimensional x-space is filled up.

In general, each time we solve a linear program with m states assumed off the boundary, we have the additional $N(N-1) - m$ constraints $\dot{x}_i^j = 0$ corresponding to those states on the boundary. The solution provides us with an optimal direction (or directions) in an m-dimensional subspace of x-space which may be used to extend the region of applicability of the optimal control to a convex region in the m-space. The union of all such regions produced by a comprehensive set of linear programs covers the entire $N(N-1)$-dimensional space with optimal controls.

This algorithm can be made to operate efficiently by virtue of the fact that each linear program in a given sequence is parametrically related to the previous one (addition of a column and perturbation of cost coefficients). The problem of finding the complete set of nonunique solutions which is needed to specify all optimal directions is handled by a variation of the algorithm suggested by Chernikova [6]. A computer program is currently being designed and implemented to construct the feedback space by the method indicated above. Hopefully this technique may be extended to treat the case with inputs.

ACKNOWLEDGMENT

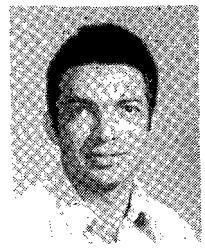
The author thanks his students F. Moss and M. Bello for many stimulating discussions regarding the models introduced in this paper. Thanks are also extended to an anonymous reviewer for many useful suggestions.

REFERENCES

[1] L. Kleinrock, *Communication Nets: Stochastic Message Flow and Delay.* New York: McGraw Hill, 1964.
[2] D. G. Cantor and M. Gerla, "Optimal routing in a packet-switched computer network," *IEEE Trans. Comput.*, vol. C-23, pp. 1062-1069, Oct. 1974.
[3] L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko, *The Mathematical Theory of Optimal Processes.* New York: Wiley-Interscience, 1962.
[4] F. Moss, Ph.D. dissertation, Massachusetts Inst. Technol., Cambridge, 1976.
[5] R. Bellman, *Adaptive Control Processes: A Guided Tour.* Princeton, NJ: Princeton Univ. Press, 1961.
[6] N. V. Chernikova, "Algorithm for finding a general formula for the nonnegative solutions of a system of linear inequalities," *U.S.S.R. Computational Math. and Math. Phys.*, vol. 4, pp. 151-158, 1964.
[7] A. Segall and T. Kailath, "The modeling of randomly modulated jump processes," *IEEE Trans. Inform. Theory*, vol. IT-21, pp. 135-143, Mar. 1975.
[8] W. M. Wonham, "On the separation theorem of stochastic control," *SIAM J. Contr.*, vol. 6, no. 2, pp. 312-326, 1968.
[9] A. Segall, "Optimal control of noisy finite state Markov processes," submitted to *IEEE Trans. Automat. Contr.*
[10] J. M. McQuillan, "Adaptive routing algorithms for distributed networks," Bolt, Beranek and Newman, Inc., Rep. 2831, May 1974.
[11] R. G. Gallager, "An optimal routing algorithm using distributed computation," this issue, pp. 73-85.
[12] M. Bello, S. M. thesis, Dep. Elec. Eng. and Comput. Sci., Massachusetts Inst. Technol., Cambridge, Sept. 1976.
[13] M. Schwartz and C. K. Cheung, "The gradient projection algorithm for multiple routing in message-switched networks," *IEEE Trans. Commun.*, vol. COM-24, pp. 449-456, Apr. 1976.

[14] H. J. Payne and W. A. Thompson, "Traffic assignment on transportation networks with capacity constraints and queueing," in *Proc. 47th Nat. ORSA Meeting*, 1975.
[15] R. G. Miller, "The jackknife–A review," *Biometrica*, vol. 61, no. 1, pp. 1-15, 1974.

★



Adrian Segall (S'71-M'74) was born in Bucharest, Romania, on March 15, 1944. He received the B.Sc. and M.Sc. degrees from the Technion–Israel Institute of Technology, Haifa, in 1965 and 1971, respectively, both in electrical engineering, and the Ph.D. degree from Stanford University, Stanford, CA, in 1973 also in electrical engineering, with a minor in statistics.

From 1965 to 1968 he served in the Israel Defense Army as an Electronics Engineer. From 1968 to 1971 he was a Research Engineer in the Scientific Department, Israel Ministry of Defence. From 1971 to 1973 he worked on his Ph.D. dissertation at Stanford University, where he held positions of Research Assistant and Acting Instructor in Electrical Engineering. From 1973 to 1974 he was a Research Engineer at Systems Control, Inc., Palo Alto, CA, and a Lecturer in Electrical Engineering at Stanford University. In 1974 he joined the faculty of the Massachusetts Institute of Technology, Cambridge, where he was an Assistant Professor of Electrical Engineering and Computer Science and a consultant to Codex Corporation. He is now with the Department of Electrical Engineering, Technion IIT, Haifa, Israel. His research interests are in applications of the theory of stochastic processes and optimization to computer communication networks, estimation, detection, and automatic control.

Throughput in the ARPANET–Protocols and Measurement

LEONARD KLEINROCK, FELLOW, IEEE, AND HOLGER OPDERBECK, MEMBER, IEEE

Abstract–**The speed at which large files can travel across a computer network is an important performance measure of that network. In this paper we examine the achievable sustained throughput in the ARPANET. Our point of departure is to describe the procedures used for controlling the flow of long messages (multipacket messages) and to identify the limitations that these procedures place on the throughput. We then present the quantitative results of experiments which measured the maximum throughput as a function of topological distance in the ARPANET. We observed a throughput of approximately 38 kbit/s at short distances. This throughput falls off at longer distances in a fashion which depends upon which particular version of the flow control procedure is in use; for example, at a distance of 9 hops, an October 1974 measurement gave 30 kbit/s, whereas a May 1975 experiment gave 27 kbit/s. The two different flow control procedures for these experiments are described, and the sources of throughput degradation at longer distances are identified, a major cause being due to a poor movement of critical limiting resources around in the network (this we call "phasing"). We conclude that flow control is a tricky business, but in spite of this, the ARPANET throughput is respectably high.**

I. INTRODUCTION

THE ARPANET, which was the world's first large-scale experimental packet-switching network, needs little introduction; it has been amply documented (see, for example, [5] and the extensive references therein). Our interest in this paper is to describe the message-handling protocols and some experimental results for the achievable throughput across the ARPANET. These experiments were conducted at the UCLA Network Measurement Center (NMC) and show that the network can support roughly 38 kbit/sec between HOST computers which are a few hops apart; for more distant HOST pairs, the throughput falls off to a level dependent upon the particular version of message processing used, as discussed in detail below.

An earlier NMC experiment reported upon the behavior of actual user traffic in the ARPANET (and also described the NMC itself) [4]. More recent NMC experiments identified, explained, and solved some deadlock and throughput-degradation phenomena in the ARPANET [11] and also measured the effect of network protocols and control messages on line overhead [4]. The experiments reported upon herein consisted of throughput measurements of UCLA-generated traffic (using our PDP 11/45 HOST in a dedicated mode) which was sent through the ARPANET to "fake" HOST's at various topological distances (hops) from UCLA. Each experiment ran for 10 min during which time full (8-packet) multipacket traffic was pumped into the ARPANET as fast as the network would permit. Both throughput (from the UCLA HOST to the destination HOST) and delay (as seen by the UCLA HOST) were measured, along with some other statistics described below.

This paper is organized as follows. We describe the message-handling procedure for multipacket messages in Section II, identify the limitations this procedure imposes on the throughput in Section III, and then quantitatively report upon the October 1974 throughput experiments in Section IV. The issue of looping in the adaptive routing procedure and its erratic effect on throughput is discussed in Section V. Some

Manuscript received May 11, 1976; revised July 22, 1976. This paper has been presented at the Fourth Data Communications Symposium, Quebec City, P.Q., Canada, October 7-9, 1975. This work was supported by the Advanced Research Projects Agency of the Department of Defense under Contract DAHC-15-73-C-0368.

L. Kleinrock is with the Department of Computer Science, University of California, Los Angeles, CA 90024.

H. Opderbeck was with the Department of Computer Science, University of California, Los Angeles, CA 90024. He is now with the Telenet Corporation, Washington, DC.