Distributed Asynchronous Optimal Routing in Data Networks

JOHN N. TSITSIKLIS, MEMBER, IEEE, AND DIMITRI P. BERTSEKAS, FELLOW, IEEE

Abstract—**In this paper we study the performance of a class of distributed optimal routing algorithms of the gradient projection type under weaker and more realistic assumptions than those considered thus far. In particular, we show convergence to an optimal routing without assuming synchronization of computation at all nodes and measurement of link lengths at all links, while taking into account the possibility of link flow transients caused by routing updates. This demonstrates the robustness of these algorithms in a realistic distributed operating environment.**

I. INTRODUCTION

THE most popular formulation of the optimal distributed routing problem in a data network is based on a multicommodity flow optimization whereby a separable objective function of the form

$$\sum_{(i,j)} \bar{D}^{ij}(F^{ij})$$

is minimized with respect to the flow variables F^{ij} subject to multicommodity flow constraints [1]–[3], [12]. Here (i, j) denotes a generic directed network link, and $\bar{D}^{ij}$ is a strictly convex differentiable, increasing function of F^{ij} which represents in turn the total traffic arrival rate on link (i, j) measured, for example, in packets or bits per second.

We want to find a routing that minimizes this objective. By a routing we mean a set of active paths for each origin–destination (OD) pair (set of paths carrying some traffic of that OD pair), together with the fraction of total traffic of the OD pair routed along each active path.

A typical example of adaptive distributed routing, patterned after the ARPANET algorithm [4], operates roughly as follows.

The total link arrival rates F^{ij} are measured by time averaging over a period of time, and are communicated to all network nodes. Upon reception of these measured rates each node updates the part of the routing dealing with traffic originating at that node. The updating method is based on some rule, e.g., a shortest path method [2], [4], or an iterative optimization algorithm [1], [5], [6].

There are a number of variations of this idea; for example, some relevant function of F^{ij} may be measured in place of F^{ij} [such as average delay per packet crossing link (i, j)], or a somewhat different type of routing policy may be used, but these will not concern us for the time being. The preceding algorithm is used in this paper as an example which is interesting in its own right but also involves ideas that are common to other types of routing algorithms.

Most of the existing analysis of distributed routing algorithms such as the one above is predicated on several assumptions that are to some extent violated in practice. These are as follows.

Manuscript received April 11, 1985; revised November 8, 1985. Paper recommended by Associate Editor, T. L. Johnson. This work was supported by DARPA under Contract ONR-N00014-75-C-1183 and by an IBM Faculty Development Award.

The authors are with the Massachusetts Institute of Technology, Cambridge, MA 02139.

IEEE Log Number 8407380.

1) The *quasi-static assumption*, i.e., the external traffic arrival process for each OD pair is stationary over time. This assumption is approximately valid when there is a large number of user-pair conversations associated with each OD pair, and each of these conversations has an arrival rate that is small relative to the total arrival rate for the OD pair (i.e., a "many small users" assumption). An asymptotic analysis of the effect of violation of this assumption on the stationary character of the external traffic arrival rates is given in [7].

2) The *fast settling time assumption*, i.e., transients in the flows F^{ij} due to changes in routing are negligible. In other words, once the routing is updated, the flows F^{ij} settle to their new values within time which is very small relative to the time between routing updates. This assumption is typically valid in datagram networks but less so in virtual circuit networks where, existing virtual circuits may not be rerouted after a routing update. When this assumption is violated, link flow measurements F^{ij} reflect a dependence not just on the current routing but also on possibly several past routings. A seemingly good model is to represent each F^{ij} as a convex combination of the rates of arrival at (i, j) corresponding to two or more past routing updates.

3) The *synchronous update assumption*, i.e., all link rates F^{ij} are measured simultaneously, and are received simultaneously at all network nodes who in turn simultaneously carry out a routing update. However, there may be technical reasons (such as software complexity) that argue against enforcing a synchronous update protocol. For example, the distributed routing algorithm of the ARPANET [4] is not operated synchronously. Furthermore, in an asynchronous updating environment, the rates F^{ij} are typically measured as time averages that reflect dependence on more than one update.

In this paper we study gradient projection methods, which are one of the most interesting classes of algorithms for distributed optimal routing. A typical iteration in a gradient method consists of making a small update in a direction which improves the value of the cost function, e.g., opposite to the direction of the gradient. A gradient projection method is a modification of this idea, so that constrained optimization problems (such as the multicommodity flow problem of this paper) may be handled as well: namely, whenever an update leads to a point outside the feasible set (which is determined by the constraints of the problem), feasibility is enforced by projecting that point back into the feasible set. The first application of this type of gradient projection method in data communication routing is due to Gallager [1] as explained later in [14]. Gallager's method operates in a space of link flow fractions. Related gradient projection methods which operate in the space of path flows are given in [3], [5], [15], and [16]. This latter class of methods is the starting point for the analysis of the present paper. We conjecture, however, that qualitatively similar results hold for Gallager's method as well as for its second derivative version [6].

Our main result states that gradient projection methods for optimal routing are valid even if the settling time and synchronous update assumption are violated to a considerable extent. Even though we retain the quasi-static assumption in our analysis, we conjecture that the result of this paper can be generalized along the lines of another related study [7] where it is shown that a routing algorithm based on a shortest path rule converges to a neighborhood of the optimum. The size of this neighborhood depends on

0018-9286/86/0400-0325$01.00

the extent of violation of the quasi-static assumption. A similar deviation from optimality can be caused by errors in the measurement of F^{ij}. In our analysis, these errors are neglected.

A practical routing algorithm that nearly falls within the framework of the present paper is the one implemented in the CODEX network [18]. There destination nodes of OD pairs asynchronously assign and reroute virtual circuits to shortest paths with respect to link lengths that relate to first derivatives of link costs. Only one virtual circuit can be rerouted at a time, but several virtual circuits can be rerouted before new measurements are received. More precisely, a destination node assigns (or reroutes) a virtual circuit to a path for which the assignment (rerouting) results in minimum cost. This is equivalent to assignment (rerouting) on a shortest path with respect to link lengths which are first derivatives of link costs evaluated at a flow that lies between the current flow and the flow resulting once the assignment (rerouting) is effected. Another difference is that, in the CODEX network, each virtual circuit may carry flow that is a substantial portion of a link's capacity. This may place a lower bound on the amount of flow that can be diverted to a shortest path at each iteration.

In the next section we provide some background on distributed asynchronous algorithms and discuss the relation of the result of the present paper with earlier analyses. In Section III we formulate our class of distributed asynchronous routing algorithms and present our main results. In Section IV we study a related algorithm. The proofs of our results may be found in the Appendix.

II. Asynchronous Optimization Algorithms

We provide here a brief discussion of the currently available theory and tools of analysis of asynchronous distributed algorithms. An extensive survey may be be found in [17]. In a typical such algorithm (aimed at solving an optimization problem) each processor i has in its memory a vector x^i which may be interpreted as an estimate of an optimal solution. Each processor obtains measurements, performs computations, and updates some of the components of its vector. Concerning the other components, it relies entirely on messages received from other processors. We are mainly interested in the case where minimal assumptions are placed on the orderliness of message exchanges.

There are two distinct approaches for analyzing algorithmic convergence. The first approach is essentially a generalization of the Lyapunov function method for proving convergence of centralized iterative processes. The idea here is that, no matter what the precise sequence of message exchange is, each update by any processor brings its vector x^i closer to the optimum in some sense. This approach applies primarily to problems involving monotone or contraction mappings with respect to a "sup-"norm (e.g., a distributed shortest path algorithm) [8], [9]; it is only required that each processor communicates to every other processor an infinite number of times.

The second approach is based on the idea that if the processors communicate fast enough relative to the speed of convergence of the computation, then the evolution of their solution estimates x^i may be (up to first order in the step-size used) the same as if all processors were communicating to each other at each time instance [10], [11]. The latter case is, however, mathematically equivalent to a centralized (synchronous) algorithm for which there is an abundance of techniques and results. Notice that in this approach, slightly stronger assumptions are placed on the nature of the communication process than in the first one. This is compensated by the fact that the corresponding method of analysis applies to broader classes of algorithms.

The method of analysis of the present paper is close in spirit to the second approach outlined above. Unfortunately, however, the results available cannot be directly applied to the routing problem studied in this paper and a new proof is required. One reason is that earlier results concern algorithms for unconstrained optimization. In the routing problem, the nonnegativity and the conservation of flow introduce inequality and equality constraints. While equality constraints could be taken care of by eliminating some of the variables, inequality constraints must be explicitly taken into account. Another difference arises because, in the routing algorithm, optimization is carried out with respect to path flow variables, whereas the messages being broadcast contain estimates of the link flows (see the next section). In earlier results the variables being communicated were assumed to be the same as the variables being optimized. Finally, the transient behavior of the network (which results from the fact that we do not make the fast settling time assumption) adds a few more particularities to the model and the analysis.

III. The Routing Model

We present here our basic assumptions, our notation, and the model by which the nodes in a communication network may adjust the routing of the flows through that network.

We are given a network described by a directed graph $G = (V, E)$. (V is the set of nodes, E the set of directed links.) For each pair $w = (i, j)$ of distinct nodes i and j (also called on origin-destination, or OD pair) we introduce P_w, a set of directed paths from i to j, containing no loops. (These are the candidate paths for carrying the flow from i to j.)[1] For each OD pair $w = (i, j)$, let r_w be the total arrival rate at node i of traffic that has to be sent to node j (measured, for example, in packets or bits per second). For each path $p \in P_w$, we denote by $x_{w,p}$ the amount of flow which is routed through path p. Naturally, we have the constraints

$$x_{w,p} \geq 0, \qquad \forall\, p \in P_w,\ \forall\, w,$$

$$\sum_{p \in P_w} x_{w,p} = r_w,\ \forall\, w.$$

Let us define a vector x_w with components $x_{w,p}$, $p \in P_w$. Suppose that there is a total of M OD pairs and let us index them so that w takes values in $\{1, \cdots, M\}$. Then, the totality of flows through the network may be described by a vector $x = (x_1, \cdots, x_M)$. Naturally, x is subject to the constraint $x \in G$ where $G = G_1 \times G_2 \times \cdots \times G_M$, and G_w is the simplex defined by (3.1) and (3.2).

For any link (i, j) in the network, let F^{ij} denote the corresponding traffic arrival rate at that link. Clearly,

$$F^{ij} = \sum_{w=1}^{M} \sum_{\substack{p \in P_w \\ (i,j) \in p}} x_{w,p}, \tag{3.3}$$

i.e., the total flow on link (i, j) is the sum of path flows of all paths traversing (i, j). Alternatively, (3.3) may be written as

$$F^{ij} = \langle e^{ij}, x \rangle \tag{3.4}$$

where $\langle \cdot, \cdot \rangle$ denotes the usual inner product and e^{ij} is an appropriate vector with 0 or 1 entries.

A cost function, corresponding to some measure of congestion through the network, is introduced. We assume the separable form

$$\bar{D} = \sum_{(i,j) \in E} \bar{D}^{ij}(F^{ij}). \tag{3.5}$$

We assume that for each link $(i, j) \in E$, the function $\bar{D}^{ij}$ is

[1] A simple choice is to let P_w be the set of all directed paths from i to j. For practical reasons, however, one may wish to consider a smaller set P_w. While such a restriction may increase the optimal value of the cost function, there may be benefits relating to ease of implementation. In any case, only those paths in P_w that carry positive flow will be involved in the calculations of the following algorithm. Furthermore a shortest path algorithm can be used to augment P_w with new paths (see [3], [5], and [15]).

defined on [0, ∞), is real valued (finite), convex, and continuously differentiable. We also assume that the derivative of $\bar{D}^{ij}$ is Lipschitz continuous on any bounded interval. A typical example is when $\bar{D}$ expresses average delay per message based on the Kleinrock independence assumption [12].

We are interested in the case where the nodes in the network adjust the path routing variables $x_{w,p}$ so as to minimize (3.5). Since a set of path flow variables $\{x_{w,p}: p \in P_w,\ w \in \{1, \cdots, M\}\}$ determines uniquely the link flow variables F^{ij} [through (3.3)], it is more convenient to express the cost function in terms of the path flow variables. We are thus led to the cost function

$$D(x) = \sum_{(i,j)\in E} D^{ij}(x), \tag{3.6}$$

where

$$D^{ij}(x) = \bar{D}^{ij}(\langle e^{ij}, x\rangle)$$

[compare to (3.4) and (3.5)]. Clearly, D^{ij} inherits the convexity and smoothness properties of $\bar{D}^{ij}$.

Let us now consider the situation where the flows change with time, due to rerouting decisions made by the nodes in the network. Accordingly, the flows at time n are described by a vector $x(n) = (x_1(n), \cdots, x_M(n)) \in G$. Let us assume that the routing decisions for the flow corresponding to a particular OD pair $w = (i, j)$ are made by the origin node i. In an ideal situation, node i would have access to the exact value of $x(n)$ and could perform the gradient projection update [3], [15]

$$x_w(n+1) = \left[\quad \frac{\partial D}{\partial} (x(n)) \right]^+ \tag{3.7}$$

Here, γ is a positive scalar step-size, μ_w a positive scaling constant, and $[\cdot]^+$ denotes the projection on the simplex G_w with respect to the Euclidean norm. The vector $x_w(n)$ can be used to obtain the fraction of flow that should be directed on each path of the OD pair w between times n and $(n + 1)$. These fractions can form the basis for implementation of the routing algorithm.

For reasons related to the convergence rate of the algorithm, we may also wish to consider the following generalization of (3.7):

$$x_w(n+1) = \left[x_w(n) - \gamma M_w^{-1}(n) \frac{\partial D}{\partial x_w}(x(n)) \right]^+_{M_w(n)}. \tag{3.8}$$

Here, γ is again a positive scalar step-size and $M_w(n)$ is a symmetric positive definite matrix (which is time varying in general). Finally, $[\cdot]^+_{M_w(n)}$ denotes the projection on G_w with respect to the norm induced by $M_w(n)$. More precisely, for a given x, the projection $[x]^+_{M_w(n)}$ is the unique vector which minimizes $\langle (z - x), M_w(n)(z - x)\rangle$ over all $z \in G_w$. (In the special case where $M_w(n) = I$, $[\cdot]^+_{M_w(n)}$ coincides with the usual projection with respect to the Euclidean norm.) An equivalent formulation is to define $x_w(n + 1)$ as the (unique) solution of the constrained optimization problem

$$\min_{x_w \in G_w} \left| \left\langle \frac{\partial D}{\partial x_w}(x(n)),\ (x_w - x_w(n)) \right\rangle \right. \\ \left. + \frac{1}{2\gamma} \langle x_w - x_w(n),\ M_w(n)(x_w - x_w(n)) \rangle \right|. \tag{3.9}$$

Typically, $M_w(n)$ is taken to be some estimate of the Hessian matrix $\partial^2 D/\partial x_w^2$. With such a choice (3.8) becomes an approximation to a projected Newton method. Such methods usually have faster convergence, when compared to (3.7), for roughly the same reasons that Newton methods for unconstrained optimization are better than the ordinary gradient algorithm. Nevertheless, since convergence rates are not studied in this paper, we do not need to be specific on the choice of $M_w(n)$. We will only assume that there exist positive constants δ, Δ such that

$$0 < \delta I \le M_w(n) \le \Delta I, \qquad \forall\ n,\ w,^2 \tag{3.10}$$

where I is the identity matrix of suitable dimension.

The convergence of an algorithm described by (3.7) or (3.8) follows from known results [14], [16]. However, in a practical situation, iteration (3.7) or (3.8) is bound to be unrealistic for several reasons.

1) It assumes perfect synchronization of the computation of all origin nodes.

2) It assumes that $x(n)$ (or, equivalently, the link flows $F^{ij}(n)$ at time n) may be measured exactly at time n and that the measured values are instantly transmitted to every origin node which needs these values. This, in turn presupposes a perfectly synchronized exchange of messages carrying these values.

3) Even if the origin node i is able to compute $x_w(n + 1)$ exactly through (3.7) or (3.8), the actual flows through the network, at time $n + 1$, will be different from the computed ones, unless the settling time is negligible.

The above necessitates the development of a more realistic model, which is done below.

First, because of remark 3) we will differentiate between the actual flows through the network (denoted by $x(n)$, $x_w(n)$, etc.) and the *desired* flows, as determined by the computations of some node; the latter will be denoted by $\bar{x}(n)$ and $\bar{x}_w(n)$. The routing decisions of some node at time n are determined by the desired flows $\bar{x}_w(n)$. However, due to transients, each component $x_{w,p}(n)$ of the actual flow $x(n)$ will take some value between $\bar{x}_{w,p}(n)$ and $x_{w,p}(n - 1)$. It is therefore natural to assume that for each time n and for each path $p \in P_w$, there exists some (generally unknown) $a_{w,p}(n)$ between 0 and 1 such that

$$x_{w,p}(n) = a_{w,p}(n)\bar{x}_{w,p}(n) + (1 - a_{w,p}(n))x_{w,p}(n-1). \tag{3.11}$$

We will also assume that for some $\alpha > 0$,

$$a_{w,p}(n) \ge \alpha, \qquad \forall\ w,\ p,\ n. \tag{3.12}$$

The above assumptions are motivated from a consideration of the way that routing strategies are implemented in actual data networks and is mainly applicable to the case of virtual circuit routing. If a certain path has more virtual circuits $(x_{w,p}(n))$ then desired $(\bar{x}_{w,p}(n))$, then no new virtual circuits will be assigned to it, whereas some of the existing virtual circuits will be deleted when the corresponding conversation terminates. A similar situation prevails if $x_{w,p}(n) < \bar{x}_{w,p}(n)$. Thus, $x_{w,p}(n + 1)$ is expected to take values in the range postulated by (3.11), (3.12). In the more realistic case, however, where arrivals and departures of virtual circuits are random, (3.11), (3.12) will only hold with some probability which converges to one as the violation of the quasi-static assumption becomes smaller and smaller.

From (3.11) and the requirement that x_w belongs to the simplex G_w, we conclude that the coefficients $a_{w,p}(n)$ have to satisfy for every w, n the condition

$$\sum_p a_{w,p}(n)(\bar{x}_{w,p}(n) - x_{w,p}(n-1)) = 0. \tag{3.13}$$

We next introduce an algorithm for updating desired flows, and try to model the effects of asynchronism. We postulate an update rule of the form [cf. (3.8)]

$$\bar{x}_w(n+1) = [\bar{x}_w(n) - \gamma M_w^{-1}(n)\lambda_w(n)]^+_{M_w(n)}. \tag{3.14}$$

Here $\lambda_w(n)$ is some estimate of $\partial D/\partial x_w(x(n))$ which is, in general, inexact due to asynchronism and delays in obtaining measurements. However, it would be unnatural to assume that the computation (3.14) is carried out at each time instance for each

[2] The notation $A \le B$, for matrices A, B, means that $B - A$ is nonnegative definite.

OD pair. We therefore define a set T_w of times for which (3.14) is used. For all $n \notin T_w$, we simply let $\bar{x}_w(n+1) = \bar{x}_w(n)$. We only assume that the time between consecutive updates (equivalently, the difference of consecutive elements of T_w) is bounded, for each w. In particular, we allow the possibility that iteration (3.14) is executed for some OD pairs w several times before being executed even once for some other OD pairs. This captures the uncoordinated character of a realistic distributed environment where the origin nodes of OD pairs carry out a routing update whenever some new information becomes available without regard as to whether this information has reached other nodes. We now describe the process by which $\lambda_w(n)$ is formed.

For each link (i, j), node i estimates from time to time the amount of traffic through that link. Practically, these estimates do not correspond to instantaneous measurements but to an average of a set of measurements obtained over some period of time. Accordingly, at each time n, node i has available an estimate

$$\tilde{F}^{ij}(n) = \sum_{Q}^{n} c^{ij}(n, m) F^{ij}(m). \tag{3.15}$$

Here, $c^{ij}(n, m)$ are (generally unknown) nonnegative scalars summing to one (for fixed n), and Q is a bound on the time over which measurements are averaged. These estimates are broadcast from time to time (asynchronously and possibly with some variable delay). Let us assume that the time between consecutive broadcasts plus the communication delay until the broadcasted messages reach all nodes is bounded by some T. It follows that at time n each node k knows the value of $\tilde{F}^{ij}(m_k)$, for some m_k with $n - T \le m_k \le n$. Combining this observation with (3.15) we conclude that at time n, each node k knows an estimate $\hat{F}^{ij}_k(n)$ satisfying

$$\hat{F}^{ij}_k(n) = \sum_{m=n-C}^{n} d^{ij}_k(n, m) F^{ij}(m) \tag{3.16}$$

where $C = T + Q$ and $d^{ij}_k(n, m)$ are (generally unknown) nonnegative coefficients summing to one, for fixed n.

For each OD pair w, the corresponding origin node (let us denote it by k) uses the values of $\hat{F}^{ij}_k(n)$ to form an estimate $\lambda_w(n)$ of $\partial D/\partial x_w(x(n))$ as follows. Note that

$$\frac{\partial D^{ij}}{\partial x_{w,p}}(x(n)) = \begin{cases} \dfrac{\partial \tilde{D}^{ij}}{\partial F^{ij}}(F^{ij}(n)), & \text{if } (i, j) \in p \\ 0 & \text{otherwise.} \end{cases} \tag{3.17}$$

Accordingly, a natural estimate is given (componentwise) by

$$\lambda_{w,p}(n) = \sum_{(i,j)\in p} \frac{\partial \tilde{D}^{ij}}{\partial F^{ij}}(\hat{F}^{ij}_k(n)). \tag{3.18}$$

The development of our model is now complete. To summarize, the basic equation is (3.14), where $x(n)$ is determined by (3.11), $\lambda_w(n)$ is determined by (3.18), $\hat{F}^{ij}_k(n)$ is given by (3.16), and F^{ij} is related to x by (3.3).

Our main result states that the above described algorithm converges to an optimal routing.

Theorem 3.1: With the algorithm and the assumptions introduced above and provided that the step-size γ is chosen small enough, $D(x(n))$ converges to $\min_{x\in G} D(x)$ and any limit point of $\{x(n)\}$ is a minimizing point. Moreover, $x_w(n) - \bar{x}_w(n)$ converges to zero for all OD pairs w.

Corollary 3.1: Under the Assumptions of Theorem 3.1, if each $\tilde{D}^{ij}$ is strictly convex (as a function of F^{ij}), then the vector of link flows $F^{ij}(n)$ converges to the unique minimizing vector of the cost function (3.5), over all link flow vectors of the form (3.3) with the path flow vector x ranging over the set G.

Let us point out here that Theorem 3.1 and Corollary 3.1 remain valid even if our assumption (3.11) is replaced by the following weaker assumption: there exist nonnegative coefficients $a_{w,p}(n; k)$ and some scalars $B \ge 0$, $\beta \in [0, 1)$ such that

$$\sum_{k=1}^{n} a_{w,p}(n; k) = 1, \ \forall\, n, \ w, \ p \in P_w,$$

$$x_{w,p}(n) = \sum_{k=1}^{n} a_{w,p}(n; k)\bar{x}_{w,p}(k), \ \forall\, n, \ w, \ p \in P_w,$$

$$a_{w,p}(n; k) \le B\beta^{n-k}, \ \forall\, n, \ k, \ w, \ p \in P_w.$$

This assumption basically requires that if $\bar{x}(k)$ is held constant, say equal to $\bar{x}$, then the actual flows $x(n)$ converge to $\bar{x}$ with at least a geometric rate.

Our proof of Theorem 3.1 indicates that convergence is guaranteed if the step size γ is chosen proportional to α where α is the constant of inequality (3.12). Thus, if the settling time of the network is small (α large), the step size can be also relatively large. However, if the network settles slowly, a small step size is used. This is quite reasonable because there is no point in using a rapidly changing routing strategy on a network which can only change slowly.

We close this section with a remark. A distributed asynchronous version of the Bellman algorithm for shortest paths has been shown to converge appropriately [8], [9] even if the time between consecutive broadcasts is unbounded. In our model however, we have assumed a finite bound T. The reason is that otherwise convergence is not guaranteed, as will be shown below. Of course, a boundedness assumption is always observed in practice.

A simple example which demonstrates that without such a bound the algorithm need not converge is the following. Consider the network of Fig. 1. There are three origin nodes (nodes 1, 2, and 3), with input arrival rate equal to 1 at each one of them, and a single destination node (node 6). For each OD pair there are two paths. For each origin node i, let x_i denote the flow routed through the path containing node 4. Let $\tilde{D}^{ij}(F^{ij}) = (F^{ij})^2$ for $(i, j) = (4, 6)$ or $(5, 6)$ and $\tilde{D}^{ij}(F^{ij}) = 0$ for all other links. In terms of the variables x_1, x_2, x_3, the cost becomes

$$D(x_1, x_2, x_3) = (x_1 + x_2 + x_3)^2 + (3 - x_1 - x_2 - x_3)^2. \tag{3.19}$$

We assume that the settling time is zero, so that we do not need to distinguish between actual and desired flows, and that each node i ($i = 1, 2, 3$) knows x_i exactly and is able to transmit its value instantaneously to the remaining origin nodes. Suppose that initially $x_1 = x_2 = x_3 = 1$ and that each origin node executes a large number of gradient projection iterations with a small step size before communicating the current value of x to the other nodes. Then, effectively, node i solves the problem

$$\min_{0\le x_i\le 1} \{(x_i + 2)^2 + (1 - x_i)^2\}$$

thereby obtaining the value $x_i = 0$. At that point the processors broadcast their current values of x_i. If this sequence of events is repeated, each x_i will become again equal to 1. So, (x_1, x_2, x_3) oscillates between (0, 0, 0) and (1, 1, 1) without ever converging to an optimal routing. The same behavior is also observed if the cost function (3.19) is modified by adding a term $\epsilon(x_1^2 + x_2^2 + x_3^2)$, which makes it strictly convex (consistently with the assumptions of this paper), as long as $0 < \epsilon \ll 1$. Clearly, the reason for divergence in this example is that the spirit of the "second approach" for proving convergence, discussed in Section II, is violated.

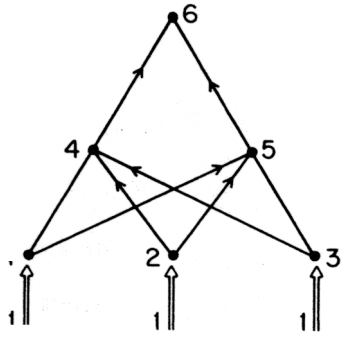

Fig. A simple routing problem.

IV. An Alternative Algorithm

In this section we consider the following variation of the basic update equation (3.14):

$$\bar{x}_w(n+1)=[x_w(n)-\gamma M_w^{-1}(n)\lambda_w(n)]^+_{M_w(n)}. \tag{4.1}$$

The main difference is that $\bar{x}_w(n+1)$, as obtained from (4.1), is a small modification of the *actual* flow $x_w(n)$ rather than the *desired* flow $\bar{x}_w(n)$, as in (3.14). If the settling time in the network is very small or if the step size γ is very small, then $\bar{x}_w(n) \approx x_w(n)$ and (4.1) coincides with (3.14). It is therefore, somewhat surprising that *(4.1) does not lead to a convergent algorithm in general,* as we now explain.

Suppose that we were dealing with an unconstrained problem and with perfect synchronization so that $\lambda_w(n) = \partial D/\partial x(x(n))$. In that case, (4.1) could be combined with (3.11) to yield

$$x_w(n+1)=x_w(n)-\gamma A_w(n+1)M_w^{-1}(n)\frac{\partial D}{\partial x_w}(x(n)) \tag{4.2}$$

where $A_w(n) = \text{diag}\,\{a_{w,p}(n)\}$ is diagonal and positive definite. However, $A_w(n+1)M_w^{-1}(n)$ need not be positive definite and the update (4.2) may be in a direction of cost increase. In the unconstrained case this issue may be taken care of by letting $M_w(n)$ be diagonal, so that $A_w(n+1)M_w^{-1}(n) > 0$. Still, this would not work for constrained problems because the projection introduces a further "rotation" of the updates. This situation may be remedied, however, by appropriately transforming the problem of projecting onto the simplex G_w, to a problem of projecting onto an orthant. More precisely, we consider the following modification of (4.1).

At each time $n \in T_w$, $\bar{x}_w(n+1)$ is computed as follows.

1) Let $\tilde{i}$ be the index of a shortest path for OD pair w with respect to link lengths $\partial\bar{D}^{ij}/\partial F^{ij}$, i.e.,

$$\lambda_{w,\tilde{i}}(n)\le\lambda_{w,p}(n), \qquad \forall\, p \in P_w.^3$$

2) Let $M_w(n)$ be a diagonal matrix with 0 in the $\tilde{i}$th position. The remaining diagonal entries are positive numbers $\mu_{w,p}(n)$, satisfying

$$0<\delta\le\mu_{w,p}(n)\le\Delta \tag{4.3}$$

where δ, Δ are fixed throughout the algorithm.

3) Let

$$\bar{x}_{w,p}(n+1) = \max\left\{0,\ x_{w,p}(n)-\frac{\gamma}{\mu_{w,p}(n)}(\lambda_{w,p}(n)-\lambda_{w,\tilde{i}}(n))\right\}, \quad p\neq\tilde{i} \tag{4.4}$$

and

$$\bar{x}_{w,\tilde{i}}(n+1)=r_w-\sum_{\substack{p\neq\tilde{i}\\ p\in P_w}}\bar{x}_{w,p}(n+1).$$

By our choice of $\tilde{i}$, we have $\bar{x}_{w,p}(n+1) \le x_{w,p}(n)$, $\forall p \neq \tilde{i}$, which implies $\bar{x}_{w,\tilde{i}}(n+1) \ge x_{w,\tilde{i}}(n) \ge 0$. Therefore, the vector $\bar{x}_w(n+1)$ computed by (4.4), (4.5) is feasible.

It can be easily checked that the vector $\bar{x}_w(n+1)$ is the solution of the optimization problem

$$\min_{x\in G_w}\ \langle\lambda_w(n),\ x-x_w(n)\rangle+\frac{1}{2\gamma}\langle x-x_w(n),\ M_w(n)(x-x_w(n))\rangle. \tag{4.6}$$

The main difference with the gradient projection updates of the last section is that now $M_w(n)$ is not positive definite, due to the $\tilde{i}$th diagonal entry which is zero. Nevertheless, the restriction of $M_w(n)$ on the linear manifold containing G_w is positive definite, so that (4.6) has a unique solution.

Theorem 4.1: The conclusions of Theorem 4.1 (convergence to an optimal routing) remain valid if (3.14) is replaced by (4.4), (4.5).

V. Conclusions

Gradient projection algorithms for routing in a data network converge appropriately even in the face of substantial asynchronism and even if the time required for the network to adjust to a change in the routing policies (settling time) is nonnegligible. While convergence is proved under the assumption that the input arrival rates r_w are constant, it is expected that the algorithm will be able to adjust appropriately in the face of small variations. If input variations become substantial, however, and the quasi-static assumption is violated, a more detailed analysis is required, incorporating stochastic effects.

Another idealization in our model arises in the measurement equation (3.13), which assumes that measurements are noiseless. This is a reasonable assumption if the time average runs over a sufficiently long period but may be unrealistic otherwise, necessitating again a more elaborate stochastic model.

Let us mention an important related class of distributed algorithms. In the present model the nodes measure and broadcast messages with their estimates of the link flows F^{ij}. Other nodes receive the broadcasted messages and use them to compute estimates of the expression $\partial\bar{D}^{ij}/\partial F^{ij}(F^{ij})$ which is required in the algorithm. An alternative possibility would be to let, say node j, measure directly or compute the value of $\partial\bar{D}^{ij}/\partial F^{ij}(F^{ij})$ and broadcast that value to the other nodes. For certain special choices of the cost function $\bar{D}^{ij}$ and under certain assumptions, the partial derivative $\partial\bar{D}^{ij}/\partial F^{ij}$ equals the average delay of a packet traveling through link (i, j). In that case, it is very natural to assume that this derivative may be measured directly, without first measuring the flow F^{ij}. Our result may be easily shown to be valid for this class of algorithms as well.

We have not presented any numerical results on the performance of our algorithms, but a simulation of an actual data network, operating in a realistic environment should be the next step in future research.

Appendix

Proof of Theorem 3.1: Let $\langle\cdot,\cdot\rangle$, $\|\cdot\|$ denote the Euclidean inner product on R^n and the associated norm, respectively. Let M be a symmetric positive definite matrix and define a new inner product $\langle\cdot,\cdot\rangle_M$ by

$$\langle x,\ y\rangle_M=\langle x,\ My\rangle.$$

[3] A more precise notation would be $\tilde{i}_w(n)$, but this would be unnecessarily cumbersome.

This inner product induces the norm $\|\cdot\|_M$ given by $\|x\|_M^2 = \langle x, x\rangle_M$. With the notation introduced in Section III, $[a]_M^+$ is the projection of a on the closed convex set $G \subset R^n$, with respect to the inner product $\langle\cdot,\cdot\rangle_M$. Therefore, the projection theorem [13, p. 69] implies that

$$\langle a-[a]_M^+,\ M(x-[a]_M^+)\rangle \le 0, \qquad \forall\, x \in G,\ \forall\, a. \quad \text{(A.1)}$$

Replacing a with $x + a$ in (A.1) we obtain

$$\langle x+a-[x+a]_M^+,\ M(x-[x+a]_M^+)\rangle \le 0, \qquad \forall\, x \in G,\ \forall\, a$$

and

$$\langle Ma,\ [x+a]_M^+ - x\rangle \ge \|x-[x+a]_M^+\|_M^2, \qquad \forall\, x \in G,\ \forall\, a. \quad \text{(A.2)}$$

We define $s(n)$ to be the vector with components

$$s_w(n) = \begin{cases} [\bar{x}_w(n) - \gamma M_w^{-1}(n)\lambda_w(n)]_{M_w}^+ - \bar{x}_w(n), & n \in T_w, \\ 0, & n \notin T_w. \end{cases} \quad \text{(A.3)}$$

Hence,

$$\bar{x}_w(n+1) = \bar{x}_w(n) + s_w(n), \qquad \forall\, n. \quad \text{(A.4)}$$

Using (A.2) with $a = -\gamma M_w^{-1}(n)\lambda_w(n)$, we obtain, for $n \in T_w$:

$$-\langle \gamma\lambda_w(n),\ s_w(n)\rangle \ge \|s_w(n)\|_{M_w(n)}^2 \ge \delta\|s_w(n)\|^2.$$

Clearly, this inequality remains valid for $n \notin T_w$ as well and may be written as

$$\langle \lambda_w(n),\ s_w(n)\rangle \le -\frac{\delta}{\gamma}\|s_w(n)\|^2, \qquad \forall\, n. \quad \text{(A.5)}$$

Using (A.4) and (3.11) we obtain, for any w, $p \in P_w$,

$$\begin{aligned} x_{w,p}(n) - \bar{x}_{w,p}(n) &= (1-a_{w,p}(n))(x_{w,p}(n-1) - \bar{x}_{w,p}(n)) \\ &= (1-a_{w,p}(n))[(x_{w,p}(n-1) \\ &\quad - \bar{x}_{w,p}(n-1)) - s_w(n-1)]. \end{aligned} \quad \text{(A.6)}$$

Using (3.12),

$$\begin{aligned} &|x_{w,p}(n) - \bar{x}_{w,p}(n)| \\ &\quad \le (1-\alpha)|x_{w,p}(n-1) - \bar{x}_{w,p}(n-1)| + |s_{w,p}(n-1)| \end{aligned} \quad \text{(A.7)}$$

which yields, for $\beta = 1 - \alpha < 1$

$$|x_{w,p}(n) - \bar{x}_{w,p}(n)| \le \sum_{k=1}^{n-1} \beta^{n-k-1}|s_{w,p}(k)|. \quad \text{(A.8)}$$

(For convenience we are assuming that the routing algorithm is initialized with $x(1) = \bar{x}(1)$. The proof is easily modified if this is not the case.) Inequality (A.8) shows that for some $A_1 \ge 0$ (independent of γ or n)

$$\|x(n) - \bar{x}(n)\| \le A_1 \sum_{k=1}^{n-1} \beta^{n-k}\|s(k)\|. \quad \text{(A.9)}$$

Compare now (3.17) to (3.18) and use the Lipschitz continuity of $\partial\bar{D}^{ij}/\partial F^{ij}$ to conclude that for some constants $A_2, \cdots, A_7$ (independent of γ or n)

$$\begin{aligned} &\left\|\frac{\partial D}{\partial x_w}(x(n)) - \lambda_w(n)\right\| \\ &\quad \le A_2 \max_{i,j,k} |\hat{F}_k^{ij}(n) - F^{ij}(n)| \\ &\quad \le A_3 \max_{i,j} \max_{n-C\le m\le n} |F^{ij}(m) - F^{ij}(n)| \\ &\quad \le A_4 \max_{n-C\le m\le n} \|x(m) - x(n)\| \\ &\quad \le A_4 \max_{n-C\le m\le n} \{\|x(m) - \bar{x}(m)\| \\ &\qquad + \|\bar{x}(m) - \bar{x}(n)\| + \|\bar{x}(n) - x(n)\|\} \\ &\quad \le A_5 \sum_{k=1}^{n-1} \beta^{n-k}\|s(k)\| + A_6 \sum_{m=n-C}^{n-1} \|s(m)\| \\ &\quad \le A_7 \sum_{k=1}^{n-1} \beta^{n-k}\|s(k)\|. \end{aligned} \quad \text{(A.10)}$$

[The second inequality follows from (3.16), the third from (3.3), the fourth is the triangle inequality, the fifth uses (A.9).] Using Lipschitz continuity once more, (A.9) and (A.10) we finally obtain, for some $A_8 \ge 0$ (independent of n, γ),

$$\left\|\frac{\partial D}{\partial x_w}(\bar{x}(n)) - \lambda_w(n)\right\| \le A_8 \sum_{k=1}^{n-1} \beta^{n-k}\|s(k)\|. \quad \text{(A.11)}$$

Using a first-order series expansion for D, we have

$$\begin{aligned} D(\bar{x}(n+1)) &\le D(\bar{x}(n)) + \sum_w \left\langle \frac{\partial D}{\partial x_w}(\bar{x}(n)),\ s_w(n)\right\rangle + A_9\|s(n)\|^2 \\ &\le D(\bar{x}(n)) + \sum_w \langle \lambda_w(n),\ s_w(n)\rangle \\ &\quad + A_8 \sum_{k=1}^{n-1} \beta^{n-k}\|s(k)\|\|s(n)\| \\ &\quad + A_9\|s(n)\|^2 \le D(\bar{x}(n)) - \frac{A_{10}}{\gamma}\|s(n)\|^2 \\ &\quad + A_{11} \sum_{k=1}^{n} \beta^{n-k}\|s(k)\|^2. \end{aligned} \quad \text{(A.12)}$$

[Here, the second inequality was obtained from (A.11); the third from (A.5).] Summing (A.12) for different values of n and rearranging terms, we obtain

$$D(\bar{x}(n+1)) \le D(\bar{x}(1)) - \sum_{k=1}^{n} \left[\frac{A_{10}}{\gamma} - A_{11}\sum_{i=k}^{n} \beta^{i-k}\right]\|s(k)\|^2. \quad \text{(A.13)}$$

Suppose that γ is small enough so that $A_{10}/\gamma - A_{11}/(1-\beta) > 0$. Note that D is continuous and that $\bar{x}(n)$ takes values in a compact set. Hence, $D(\bar{x}(n+1))$ is bounded below. Let $n \to \infty$ in (A.13) to obtain

$$\sum_{k=1}^{\infty} \|s(k)\|^2 < \infty. \quad \text{(A.14)}$$

In particular, $s(k)$ converges to zero, as $k \to \infty$, and using (A.9) we obtain $\lim_{k\to\infty} \|x(k) - \bar{x}(k)\| = 0$. It also follows from (A.13), (A.14) that $D(\bar{x}(n))$ converges, as $k \to \infty$.

Let us define for any w and for any positive definite symmetric matrix M_w

$$f_w(x, M_w) = \left[x_w - \gamma M_w^{-1} \frac{\partial D}{\partial x_w}(x)\right]_{M_w}^{+} - x_w. \qquad \text{(A.15)}$$

Let x^* be a limit point of $\{x(n)\}$. (At least one exists because G is compact.) Since we have assumed that the difference between consecutive elements of T_w is bounded, for any w, we conclude that x^* is also a limit point of $\{x(n): n \in T_w\}$. Notice also that the set of matrices satisfying (3.10) is compact. It follows that there exists a sequence $\{n_k\} \subset T_w$ such that $x(n_k)$ converges to x^* and $M_w(n_k)$ converges to some M_w^* satisfying (3.10). Finally, notice that (due to (A.10), the convergence of $s(n)$ to zero and the continuity of $\partial D/\partial x_w$)

$$\lim_{k\to\infty} \lambda_w(n_k) = \lim_{k\to\infty} \frac{\partial D}{\partial x_w}(x(n_k)) = \frac{\partial D}{\partial x_w}(x^*).$$

Putting everything together, and comparing (A.15) to (A.3), we conclude that

$$f_w(x^*, M_w^*) = \lim_{k\to\infty} s_w(n_k) = 0.$$

(This step uses the fact that $[a]_M^+$ is jointly continuous as a function of a, M.) Consequently, for each w there is a matrix M_w^* satisfying (3.10) and such that $f_w(x^*, M_w^*) = 0$, $\forall w$. Using the projection theorem [13] and (A.15), we obtain $\langle \gamma(M_w^*)^{-1}\partial D/\partial x_w(x^*), M_w^*(x_w - x_w^*)\rangle \geq 0$, $\forall x_w \in G_w$, $\forall w$. Summing over all w's we obtain $\langle \partial D/\partial x(x^*), x - x^*\rangle \geq 0$, $\forall x \in G$ and, since D is convex, we have $D(x) \geq D(x^*) + \langle \partial D/\partial x(x^*), x - x^*\rangle$, $\forall x \in G$. Therefore, x^* minimizes D over the set G, thus proving part of the theorem.

The above imply that $\min_{x\in G} D(x) = D(x^*)$ is a limit point of $\{D(x(n))\}$. Since $\{D(x(n))\}$ is a convergent sequence, it converges to $\min_{x\in G} D(x)$, thus completing the proof. □

Proof of Corollary 3.1: By Theorem 3.1, any limit point of $\{x(n)\}$ minimizes D. Hence, any limit point of $\{F^{ij}(n)\}$ minimizes $\tilde{D}$ over the convex set consisting of link flows given by (3.4) with x ranging over G. However, due to strict convexity, $\tilde{D}$ has a unique minimum over this set which proves the corollary. □

Proof of Theorem 4.1: Let $s(n)$, $\bar{s}(n)$ be vectors with components

$$s_w(n) = x_w(n+1) - x_w(n) \qquad \text{(A.16)}$$

$$\bar{s}_w(n) = \bar{x}_w(n+1) - x_w(n), \qquad \text{(A.17)}$$

respectively. Using (3.11) we obtain

$$s_w(n) = A_w(n+1)\bar{s}_w(n), \qquad \text{(A.18)}$$

where $A_w(n) = \text{diag}\{a_{w,p}(n)\}$. We therefore have, for some $A \geq 0$,

$$\begin{aligned}\langle \lambda_w(n), s_w(n)\rangle &= \sum_{p\in P_w} a_{w,p}(n+1)\lambda_{w,p}(n)\bar{s}_{w,p}(n) \\ &= \sum_{\substack{p\in P_w \\ p\neq \tilde{i}}} a_{w,p}(n+1)\bar{s}_{w,p}(n)[\lambda_{w,p}(n) - \lambda_{w,\tilde{i}}(n)] \\ &\leq -\frac{A}{\gamma}\sum_{\substack{p\in P_w \\ p\neq \tilde{i}}} a_{w,p}(n+1)|\bar{s}_{w,p}(n)|^2 \\ &\leq -\frac{\alpha A}{\gamma}\sum_{\substack{p\in P_w \\ p\neq \tilde{i}}} |\bar{s}_{w,p}(n)|^2. \qquad \text{(A.19)}\end{aligned}$$

(The first equality follows from (A.18), the second from (3.13); the first inequality follows from (4.4) and a little algebra; the last from (3.12).) Also notice that (4.5) implies

$$\bar{s}_{w,\tilde{i}}(n) = -\sum_{\substack{p\in P_w \\ p\neq \tilde{i}}} \bar{s}_{w,p}(n),$$

which finally yields, for some $A_1 \geq 0$,

$$|\bar{s}_{w,\tilde{i}}(n)|^2 \leq A_1 \sum_{\substack{p\in P_w \\ p\neq \tilde{i}}} |\bar{s}_{w,p}(n)|^2. \qquad \text{(A.20)}$$

Combining (A.19) and (A.20) we conclude that, for some $A_2 \geq 0$ independent of γ or n, we have

$$\langle \lambda_w(n), s_w(n)\rangle \leq -\frac{A_2}{\gamma}\|s_w(n)\|^2. \qquad \text{(A.21)}$$

An argument similar to (A.10) yields

$$\left\|\frac{\partial D}{\partial x_w}(x(n)) - \lambda_w(n)\right\| \leq A_3 \sum_{m=1}^{n-1} \beta^{n-m}\|s_w(m)\|.$$

We then obtain, similarly with (A.12),

$$D(x(n+1)) \leq D(x(n)) - \frac{A_2}{\gamma}\|s(n)\|^2 + A_4 \sum_{m=1}^{n} \beta^{n-m}\|s(m)\|^2.$$

From here on, the proof follows the lines of the proof of Theorem 3.1 and is, therefore, omitted. We only point out some differences. First, $f_w(x, M_w)$ should not be defined via (A.15) but as the (unique) solution of

$$\min_y \left\langle y - x, \frac{\partial D}{\partial x_w}(x)\right\rangle + \frac{1}{2\gamma}\langle y - x, M_w(y - x)\rangle.$$

Second, when we choose a convergent subsequence $x(n_k)$, we should take a further subsequence so that $\tilde{i}$ is the same at all times n_k. □

References

[1] R. G. Gallager, "A minimum delay routing algorithm using distributed computation," *IEEE Trans. Commun.*, vol. COM-25, pp. 73–85, 1977.

[2] M. Schwartz and T. E. Stern, "Routing techniques used in computer communication networks," *IEEE Trans. Commun.*, vol. COM-28, pp. 539–559, 1980.

[3] D. P. Bertsekas, "Optimal routing and flow control methods for communication networks," in *Analysis and Optimization of Systems*, A. Bensoussan and J. L. Lions, Eds. New York: Springer-Verlag, 1982, pp. 615–643.

[4] J. M. McQuillan, I. Richer, and E. C. Rosen, "The new routing algorithm for the ARPANET," *IEEE Trans. Commun.*, vol. COM-28, pp. 711–719, 1980.

[5] D. P. Bertsekas and E. M. Gafni, "Projected Newton methods and optmization of multicommodity flows," *IEEE Trans. Automat. Contr.*, vol. AC-28, pp. 1090–1096, 1983.

[6] D. P. Bertsekas, E. M. Gafni, and R. G. Gallager, "Second derivative algorithms for minimum delay distributed routing in networks," *IEEE Trans. Commun.*, vol. COM-32, pp. 911–919, 1984.

[7] E. M. Gafni and D. P. Bertsekas, "Asymptotic optimality of shortest path routing," Mass. Inst. Technol., Cambridge, MA, LIDS Rep. P-1307, July 1983; also in *IEEE Trans. Inform. Theory*, to be published.

[8] D. P. Bertsekas, "Distributed dynamic programming," *IEEE Trans. Automat. Contr.*, vol. AC-27, pp. 610–615, 1982.

[9] D. P. Bertsekas, "Distributed asynchronous computation of fixed points," in *Mathematical Programming*, vol. 27, pp. 107–120, 1983.

[10] J. N. Tsitsiklis, D. P. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," in *Proc. 1984 Amer. Contr. Conf.*, San Diego, CA, 1984.

[11] J. N. Tsitsiklis, "Problems in decentralized decision making and computation," Ph.D. dissertation, Dep. Elec. Eng. Comput. Sci., Mass. Inst. Technol., Cambridge, MA, 1984.
[12] L. Kleinrock, *Communication Nets: Stochastic Message Flow and Delay.* New York: McGraw-Hill, 1964.
[13] D. G. Luenberger, *Optimization by Vector Space Methods.* New York: Wiley, 1969.
[14] D. P. Bertsekas, "Algorithms for nonlinear multicommodity network flow problems," in *Proc. International Symposium on Systems Optimization and Analysis,* A. Bensoussan and J. L. Lions, Eds. New York: Springer-Verlag, 1979, pp. 210–224.
[15] D. P. Bertsekas, "A class of optimal routing algorithms for communication networks," in *Proc. 5th Int. Conf. Comput. Commun.,* Atlanta, GA, Oct. 1980, pp. 71–76.
[16] D. P. Bertsekas and E. Gafni, "Projection methods for variational inequalities with application to the traffic assignment problem," *Math. Progr. Study,* Vol. 17, D. C. Sorensen and R. J.-B. Wets, Eds. Amsterdam, The Netherlands: North-Holland, 1982, pp. 139–159.
[17] D. P. Bertsekas, J. N. Tsitsiklis, and M. Athans, "Convergence theories of distributed iterative processes: A survey," Mass. Inst. Technol., LIDS Rep. P-1412, Oct. 1984.
[18] P. Humblet, Private communication, 1985.

John N. Tsitsiklis (S'80–M'81) was born in Thessaloniki, Greece in 1958. He received the B.S. degrees in electrical engineering and mathematics in 1980, the M.S. degree in electrical engineering in 1981, and the Ph.D. degree in electrical engineering in 1984, all from the Massachusetts Institute of Technology, Cambridge.

He spent the 1983–1984 academic year at Stanford University, Stanford, CA, as an Assistant Professor of Electrical Engineering. He is currently an Assistant Professor of Electrical Engineering at the Massachusetts Institute of Technology. His reserch interests are in distributed computation, decentralized decision making, and estimation, as well as stochastic control.

Dimitri P. Bertsekas (S'70–SM'77–F'84) was born in Athens, Greece, in 1942. He received the Mechanical and Electrical Engineering Diploma from the National Technical University of Athens, 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 Department of Engineering-Economic Systems, Stanford University, Stanford, CA, from 1971 to 1974, and in the Department of Electrical Engineering, University of Illinois, Urbana, from 1974 to 1979. At present he is Professor of Electrical Engineering and Computer Science at the Massachusetts Institute of Technology. He is the author of *Dynamic Programming and Stochastic Control* (New York: Academic, 1976), *Constrained Optimization and Lagrange Multiplier Methods* (New York: Academic, 1982); and coauthor of *Stochastic Optimal Control: The Discrete-Time Case* (New York: Academic, 1978) and *Data Communication Networks,* which will be published in 1986.