Complexity of Gradient Projection Method for Optimal Routing in Data Networks

Wei K. Tsai, *Member, IEEE,* John K. Antonio, *Senior Member, IEEE,* and Garng M. Huang, *Senior Member, IEEE*

***Abstract*—In this paper, we derive a time-complexity bound for the gradient projection method for optimal routing in data networks. This result shows that the gradient projection algorithm of the Goldstein–Levitin–Poljak type formulated by Bertsekas converges to within ε in relative accuracy in $O(\varepsilon^{-2}h_{\min}N_{\max})$ number of iterations, where $N_{\max}$ is the number of paths sharing the maximally shared link, and $h_{\min}$ is the diameter of the network. Based on this complexity result, we also show that the one-source-at-a-time update policy has a complexity bound which is $O(n)$ times smaller than that of the all-at-a-time update policy, where n is the number of nodes in the network. The result of this paper argues for constructing networks with low diameter for the purpose of reducing complexity of the network control algorithms. The result also implies that parallelizing the optimal routing algorithm over the network nodes is beneficial.**

***Index Terms*—Algorithm complexity, congestion control, internetworking, routing.**

NOMENCLATURE

OD	Origin Destination.
GP	Gradient Projection.
OSA	One-Source-at-A-time.
AAA	All-At-A-time.
$N_{\max}$	Number of paths sharing the maximally shared link.
$h_{\min}$	Diameter of the network graph.
W	Set of OD pairs in the traffic demand.
$\|A\|_\infty$	$\max_i\{\Sigma_j\|a_{ij}\|,\}$, maximum row sum.
$\|A\|_1$	$\max_j\{\Sigma_i\|a_{ij}\|,\}$, maximum column sum.
$\|A\|_2$	Square root of the largest eigenvalue of A^TA.
$D'_{\min}$	Greatest lower bound of $D'(k)$
$D'_{\max}$	Least upper bound of $D'(k)$
γ	$D'_{\max}/D'_{\min}$
$h_p(k)$	Number of hops in the active path p at iteration k
L	Bound for the spectral radius of the Hessian matrix $\nabla^2\tilde{D}(z(k))$.

I INTRODUCTION

BECAUSE of rapid demand increases for telecommunication, wide-area data networks have become very large. As a result, computation efficiency for congestion-control algorithms has become an important issue. In this paper, we study time complexity for the gradient projection (GP) method for optimal routing in wide-area networks.

There is an extensive literature devoted to numerical methods for optimization problems; yet, far less attention has been paid to complexities of optimization problems or the algorithms for solving them [1]. In their book, Nemirovsky and Yudin [1] have addressed both the problem complexity and algorithm complexity for general convex optimization; in this paper, we focus on a specific gradient projection algorithm for the path-formulated optimal routing problem.

The optimal routing problem and the GP algorithm are important topics. The optimal routing problem is a multicommodity flow problem, which is essential for transportation of commodities over a network. The GP algorithm stands out because of its many advantages over other types of algorithms (Bertsekas [2], [8], [9]). The GP algorithm diagonally scaled[1] by the second derivatives [8] appears to be the best quasistatic optimal routing algorithm. In a deterministic synchronous environment, it converges faster than shortest path methods (used by many networks [4], [8]), and also faster (in terms of computation efforts) than the projected Newton method, when starting far from an optimal routing. These facts make the GP method most attractive [8].

While we have derived an earlier time-complexity bound of the GP algorithm in [13], this paper presents a tighter bound and derives the bounds for the AAA and OSA update policies. The definitions of the AAA and OSA policies will be given in Section V. The result shows that the GP algorithm of the Goldstein–Levitin–Poljak type formulated by Bertsekas converges to within ε in relative accuracy in $O(\varepsilon^{-2}h_{\min}N_{\max})$ number of iterations, where $N_{\max}$ is the number of paths sharing the maximally shared link, and $h_{\min}$ is the diameter of the network. The $O(\cdot)$ notation is explained in the footnote.[2] Thus, the result of this paper argues for constructing networks with low diameter for the purpose of reducing complexity of the network control algorithms. We also show that the OSA[3] update policy has the same overall complexity bound as that

Manuscript received September 2, 1997; revised October 1, 1998; approved by IEEE/ACM TRANSACTIONS ON NETWORKING EDITOR S. H. Low.

W. K. Tsai is with the Department of Electrical and Computer Engineering, University of California at Irvine, Irvine, CA 92697 USA (wtsai@uci.edu).

J. K. Antonio is with the School of Computer Science, University of Oklahoma, Norman, OK 73019 USA (e-mail: antonio@ou.edu).

G. M. Huang is with the Department of Electrical Engineering, Texas A&M University, College Station, TX 77843 USA (e-mail: huang@ee.tamu.edu).

Publisher Item Identifier S 1063-6692(99)09712-5.

[1] In this diagonally scaled GP algorithm, the gradient is first scaled by dividing each of its entries by the corresponding diagonal entry of the Hessian, then a tentative update is made, and then the result is projected onto the positive orthant, as in the equation just before (7).

[2] Let A be some subset of $\Re$ and let f: $A \mapsto \Re$ and g: $A \mapsto \Re$ be some functions. The notation $f(x) = O(g(x))$ [respectively, $f(x) = \Omega(g(x))$] means that there exists a positive constant c and some x_0 such that for every $x \in A$ satisfying $x \geq x_0$, we have $|f(x)| \leq cg(x)$ [respectively, $|f(x)| \geq cg(x)$]. The notation $f(x) = \theta(g(x))$ means that $f(x) = O(g(x))$ and $f(x) = \Omega(g(x))$.

[3] The OSA update policy can be considered as a variation of the coordinate descent algorithm in nonlinear programming [12], or a special case of the Gauss–Seidel algorithm in the general iterative algorithm [11].

of the AAA update policy [2]. The result also implies that parallelizing the optimal routing algorithm over the network nodes is beneficial (see the discussion in Section V).

The complexity result developed in this paper has two important differences from the classical asymptotic convergence rate analysis. First, our result is a time-complexity result, i.e., the computation effort is explicitly expressed as a function of the problem size. In contrast, the classical result is an asymptotic convergence rate estimate in which the dependency on the problem size is not investigated. The asymptotic convergence rate describes the behavior of error of the algorithm solution *as the number of iterations "goes to infinity"*. Thus, even if the rate of convergence is known, this rate cannot be generally converted to a time-complexity bound. Second, our complexity deals with the speed of convergence *outside* a small neighborhood of a stationary point, while the classical convergence rate result deals with the speed of convergence *inside* a neighborhood of a stationary point, e.g., Luenberger [10]. In practice, as the input demands are constantly fluctuating, and the measurements are inherently inaccurate, it is not important for the algorithm to achieve fast convergence to exact optimality; rather, it is more important for the algorithm to achieve fast convergence to a neighborhood of optimality.

Thus, our complexity result is in the same form of the time-complexity bound derived by Nemirovsky and Yudin [1] for strongly convex problems. There are two main differences between our result and that of Nemirovsky and Yudin. The first is that the optimal routing problem is not strongly convex because the Hessian matrix has zero eigenvalues (corresponding to paths with zero flows). The second is that the optimal routing problem is a specific convex optimization problem (as opposed to the general strongly convex problem) and we are able to get stronger results. The time-complexity estimates for the AAA and OSA update policies are the extra results obtained because of the specific structure of the optimal routing problem.

At this point, we would like to comment on the objective function for the optimal routing problem formulated in this paper. While the classical M/M/1 delay function was used as the link-objective function, the entire analysis will still carry through as long as the link-objective function is convex with second derivatives bounded above and away from zero. We chose to use the M/M/1 delay function only for purposes of illustration. Networks today employ a wide variety of scheduling and queuing mechanisms to control the transmission at a link; this implies that the appropriate link-objective function varies and depends on many factors. For practical networks, the cost associated with a link tends to go to infinity as the input rate approaches the link capacity. Such a kind of cost can be quite appropriately modeled by a convex function with bounded second derivatives. As the analysis in this paper can be easily adapted to this large class of link-objective functions, the results are applicable to most networks.

This paper is organized as follows. The optimal routing problem formulation and the GP algorithm are introduced in Section II. The complexity result for the usual GP algorithm, along with the computation model and the implications of the assumptions, are discussed in Section III. The series of technical lemmas leading to the first complexity result is given in Section IV, and the complexity bound for the OSA policy is described in Section V. Section VI concludes the paper and the Appendix contains the proof for Lemma 1.

II. Optimal Routing Problem and the Gradient Projection Algorithm

The optimal routing problem is described as follows. Suppose $G = (\mathcal{N}, \mathcal{L})$ is a directed graph, where $\mathcal{N}$ is the set of nodes, and $\mathcal{L}$ is the set of directed links. Let W be the set of OD pairs, and for each OD pair $w \in W$, let the traffic demand be r_w. The independent variables are the set of path flows $[[x_p]_{p \in P_w}]_{w \in W}$ which satisfies the traffic demand

$$\sum_{p \in P_w} x_p = r_w \quad \forall w \in W \tag{1}$$

where P_w is a given set of paths for an OD pair w, and x_p is the traffic flow on path p for w. In our notation, each path is uniquely identified by the integer p in the entire set of paths. The traffic balance equations, stating that the total traffic flow on any directed link is the sum of all the path flows for all the paths using that directed link, must also be satisfied. Let f_{ij} be the traffic flow on link $(i,j) \in \mathcal{L}$, then the traffic balance equations can be written as

$$f_{ij} = \sum_{w \in W} \sum_{(i,j) \in p,\, p \in P_w} x_p \quad \forall (i,j) \in \mathcal{L} \tag{2}$$

or in matrix form $f = Ex$, where $((i,j),p)$th entry of E is one if link (i,j) is in path p otherwise zero, i.e. E is the *path-link incidence matrix*. All the traffic flows $x = [x_w]_{w \in W}$ must be nonnegative

$$x_p \geq 0, \quad \forall p \in P_w \forall w \in W. \tag{3}$$

Let us consider a standard cost used in optimal routing problem for data networks: the average number of outstanding packets. Using the standard Kleinrock independence approximation and the Jackson network formulation [6], [8], the average number of outstanding packets is given by

$$\sum_{(i,j) \in \mathcal{L}} \frac{f_{ij}}{C_{ij} - f_{ij}} \tag{4}$$

where C_{ij} is the capacity of the link (i,j). The cost function defined in (4) will run into numerical problems if some assigned link flows are greater than or equal to the respective link capacities. It is possible that, in the course of the routing algorithm, a link's total flow exceeds its capacity temporarily. The conventional remedy is to use a quadratic-like cost function which gives the bounded cost at all times while the cost is identical with the original cost when all link flows stay within a fixed (but high) percentage of the respective capacities. This approximation is realistic for two reasons: 1) in real network operations, the final optimal routing always produces link flows satisfying the condition that the link capacities are not exceeded and 2) for real-world optimal routing algorithms, the approximation is routinely applied to avoid a numerical

stability problem. Therefore, the analysis derived in this paper adopts this convention of quadratic-like cost function.

Following [2], let $\rho_{\max}$ be a positive constant, independent of the problem size (to be defined in Section III) and satisfying the condition, $0<\rho_{\max}<1$, $\rho_{\max}\approx 1$. For all $(i,j)\in\mathcal{L}$, let

$$D_{ij}(f_{ij}) \stackrel{\text{def}}{=} \begin{cases} Q_{ij}(f_{ij}), & \text{if } f_{ij}<\rho_{\max}C_{ij}; \\ Q_{ij}(\rho_{\max}C_{ij})+Q_{ij}'(\rho_{\max}C_{ij})(f_{ij}-\rho_{\max}C_{ij}) \\ \quad +\frac{1}{2}Q_{ij}''(\rho_{\max}C_{ij})(f_{ij}-\rho_{\max}C_{ij})^2, \\ \quad \text{otherwise} \end{cases} \tag{5}$$

where

$$Q_{ij}(r) \stackrel{\text{def}}{=} \frac{r}{C_{ij}-r}.$$

The function $D_{ij}(f_{ij})$ will be referred to as the link cost function for each (i,j). The constant $\rho_{\max}$ is actually a replacement for the *maximum link utilization* of the network, which is defined to be $\max\{f_{ij}/C_{ij}\colon (i,j)\in\mathcal{L}\}$ for a link flow vector f. Now the first and second derivatives of the link costs are bounded for bounded flows, independent of the problem size. The cost to minimize is now written as

$$\overline{D}(f)=\sum_{(i,j)\in\mathcal{L}} D_{ij}(f_{ij}). \tag{6}$$

Note that $\overline{D}(\cdot)$ is convex and twice continuously differentiable.

The path-formulated optimal routing problem is that of minimizing (6) subject to constraints (1)–(3). The constraints (2) are eliminated by substituting $f=Ex$ in $\overline{D}(f)$ to get a new cost function $D(x)$; $D(x)$ is obviously convex and twice continuously differentiable. With this new cost and constraints (1) and (3), we obtain a variation of the classical *multicommodity flow problem* (MFP).

The optimality condition for this MFP can be easily derived [2] and is stated below

$$x_p^*>0 \Rightarrow d_p(x^*)-d_{\overline{p}_w}(x^*)=0 \tag{7}$$

where the *-superscript indicates that the corresponding quantity is optimal for the MFP, $d_p(x^*)$ denotes $\partial D(x^*)/\partial x_p$, which is interpreted as the first derivative of the "length" (FDL) of p at x^*, and $\overline{p}_w(k)$ is a shortest path according to FDL

$$d_{\overline{p}_w}(x^*)=\min_{q\in P_w}\{d_q(x^*)\}.$$

For our analysis we will also be concerned with the second derivatives of the "lengths" (SDL) for two paths p and q

$$d_{pq}^2(k) \stackrel{\text{def}}{=} \frac{\partial^2 D(x)}{\partial x_p \partial x_q}.$$

The Goldstein–Levitin–Poljak GP algorithm formulated by Bertsekas [2] for the above MFP is described as follows: let k denote the iteration number. For each $w\in W$, at each iteration $k=1,2,\cdots$, the updated flow $x_p(k+1)$ is computed as follows. The algorithm will choose, for every OD pair at every iteration a particular shortest FDL path; we will refer to these special shortest FDL paths as the *chosen* shortest FDL paths. Let $\overline{p}_w(k)$ be the chosen shortest FDL path in P_w at iteration k. Thus

$$d_{\overline{p}_w(k)}(k)=\min\{d_p(k)\colon p\in P_w\},$$

where $d_p(k) \stackrel{\text{def}}{=} d_p(x(k))$. Due to the complexity of the notation $d_{\overline{p}_w(k)}$, we shall, from now on, use $d_{\overline{p}_w}(k)$ to represent $d_{\overline{p}_w(k)}$, in a slight abuse of notation.

The GP algorithm is based on a transformed cost which is defined below. We will use the notation $s_p(k) \stackrel{\text{def}}{=} x_p(k+1)-x_p(k)$ to denote the *step*, i.e. the change in the path flow vector. The GP update is derived from the standard Taylor series expansion at the cost at iteration k

$$D(x(k+1))=D(x(k))+\sum_{p\in P} d_p(k)s_p(k) + \tfrac{1}{2}\sum_{p\in P}\sum_{q\in P} d_{pq}^2(z(k))s_p(k)s_q(k) \tag{8}$$

where $z(k)$ lies in the line segment formed by joining $x(k)$ and $x(k+1)$.

Substitute

$$x_{\overline{p}_w(k)}(k)=r_w-\sum_{p\in P_w,\, p\neq\overline{p}_w(k)} x_p(k) \quad \forall w\in W$$

into $D(x(k))$, we get the transformed cost $\tilde{D}(\tilde{x}(k))$.[4] The first and second derivatives of the transformed cost are derived

$$\begin{aligned} \tilde{d}_p(k) &\stackrel{\text{def}}{=} \frac{\partial\tilde{D}(\tilde{x}(k))}{\partial x_p}=d_p(k)-d_{w,\overline{p}_w(k)}(k) \\ &=\sum_{(i,j)\in p} D_{ij}'(f_{ij}(k))-\sum_{(i,j)\in\overline{p}_w(k)} D_{ij}'(f_{ij}(k)) \end{aligned} \tag{9}$$

$$\begin{aligned} \tilde{d}_{pq}^2(k) &\stackrel{\text{def}}{=} \frac{\partial^2\tilde{D}(\tilde{x}(k))}{\partial x_p\partial x_q} \\ &=\sum_{(i,j)\in p\cap q} D_{ij}''(f_{ij}(k))-\sum_{(i,j)\in\overline{p}_w(k)\cap q} D_{ij}''(f_{ij}(k)) \\ &\quad+\sum_{(i,j)\in\overline{p}_w(k)\cap\overline{q}_{w'}(k)} D_{ij}''(f_{ij}(k)) \\ &\quad-\sum_{(i,j)\in p\cap\overline{q}_{w'}(k)} D_{ij}''(f_{ij}(k)) \end{aligned} \tag{10}$$

where $D_{(i,j)}'(\cdot)$ [respectively, $D_{(i,j)}''(\cdot)$] denotes the first (respectively, second) derivative of $D_{(i,j)}(\cdot)$, the notation $(i,j)\in p$ means that link (i,j) is part of the path p, $p\cap q$ denotes the set of common links of paths p and q, and $\overline{q}_{w'}$ denotes the chosen shortest path for the OD pair w' to which q belongs.

We will use the symbol $\tilde{s}(k)$ to denote the transformed step, i.e., the change in the transformed path flow vector, and $\tilde{P}(k)$ to denote the set of active paths which are not the chosen shortest path at iteration k

$$\tilde{P}(k) \stackrel{\text{def}}{=} \{p\in P\colon \exists w\in W, s.t. p\in P_w, \quad p\neq\overline{p}_w(k), x_p(k)>0\}.$$

[4] We point out here that the transformed cost $\bar{D}(\bar{x}(k))$ and the actual cost $D(x(k))$ give the same value, even though they are technically two different functions.

In terms of the transformed cost, the Taylor series expansion (8) can be rewritten as

$$\tilde{D}(\tilde{x}(k+1)) = \tilde{D}(\tilde{x}(k)) + \sum_{p\in\tilde{P}(k)} \tilde{d}_p(k)s_p(k) + \tfrac{1}{2}\sum_{p\in\tilde{P}(k)}\sum_{q\in\tilde{P}(k)} \tilde{d}^2_{pq}(z(k))s_p(k)s_q(k). \tag{11}$$

The GP update equations can be stated as follows:

$$x_p(k+1) = [x_p(k) - \alpha(m_p(k))^{-1}(d_p(k) - d_{\overline{p}_w}(k))]^+$$
$$x_{\overline{p}_w(k)}(k+1) = r_w - \sum_{p\in P_w, p\neq\overline{p}_w(k)} x_p(k+1)$$

where $[\cdot]^+$ is defined by the ith element of $[x]^+ = \max(0, x_i)$, α is the scalar step size, and $m_p(k)$ is a scaling factor. In this GP algorithm, $m_p(k)$ is assumed to be bounded in the following sense: let ℓ and Δ be scalar positive constants, independent of the problem size, such that $m_p(k)$ satisfies the condition

$$0 < \ell \le m_p(k) \le \Delta \quad \forall p \in P_w, w \in W. \tag{12}$$

Here, ℓ is an estimate of the minimum of the minimum diagonal elements of the Hessian matrices associated with the set of active paths for all the iterations, and Δ is an estimate of the maximum of the maximum diagonal elements of the Hessian matrices associated with the set of active paths[5] for all the iterations. The upper-bound Δ is guaranteed by our cost function assumption (5), and the lower-bound ℓ is an extra assumption. The lower-bound assumption is always observed in practice, as any practical implementation of GP algorithm will make sure that the division by $m_p(k)$ is bounded away from zero, independent of the problem size.

III. Computation Model and Complexity Result

In this section, we will describe our computation model (our assumptions), the implications of our assumptions, and the complexity bound for the usual GP algorithm.

What constitutes the problem size? Obviously, there are many different ways to parameterize the size of the MFP. The complication here is that the MFP actually depends on many intertwining factors: the network topology, the network link capacities, and the traffic demands for networks, to name just a few. We shall use, in a simplified way, the number of nodes n as the problem size.

As the network size goes to infinity, the cost may also go to infinity. It is obvious that, even if the network size approaches infinity, we should only be concerned with the relative cost performance. That is, the algorithm should be considered to have "converged" if the cost lies within a small percentage of the optimal cost. Thus, in this paper we will consider only relative accuracy of the cost.

We now state more assumptions of the computation model. For each $(i,j) \in \mathcal{L}$, we assume that $C_{ij}(n) = \theta(1)$, and consistent with this assumption, we assume that $r_w(n) = \theta(1)$, for all $w \in W$. Finally, we also assume that at each iteration of the GP algorithm, there are $\theta(1)$ number of active paths per OD pair. The above assumptions are natural and consistent with practical data networks.

Remark: So far, we have defined many symbols and more symbols will be added. To simplify the notation, we will use the following convention. We define $P \stackrel{\text{def}}{=} \cup_{w\in W} P_w$ to be the set of admissible paths. We will use $D^* \stackrel{\text{def}}{=} D(f^*)$ to denote the optimal cost. For any variable written as $v(k)$, we could mean the variable $v(x(k))$—for example, $D(k)$ is the short form of $D(x(k))$. $D(\cdot)$ will be used as the *generic* cost, *independent of the actual arguments.* For example, $D(k)$ denotes the cost at iteration k. We will use $\|\cdot\|$ to denote the Euclidean norm (l_2-norm).

From the standard Jackson network formulation [6], [8], and our basic assumptions, the average number of outstanding packets will grow at a rate proportional to number of OD pairs, given that the average delay remains unchanged. Thus we will also assume the following:

$$D^* = \Omega(|W|). \tag{A}$$

From (5), there exist positive constants $D'_{\min}$, $D''_{\min}$, $D'_{\max}$, and $D''_{\max}$, all independent of n, such that, for all $k \ge 0$ and $(i,j) \in \mathcal{L}$

$$D'_{\min} \le D'_{ij}(f_{ij}(k)) \le D'_{\max}$$
$$D''_{\min} \le D''_{ij}(f_{ij}(k)) \le D''_{\max}. \tag{13}$$

From the GP algorithm, a path p will be active at some iteration $k \ge 0$ only if $x_p(0) > 0$ or p is the chosen shortest FDL path at some iteration $k' \le k$. Because of the optimality condition (7), we assume that, for each OD pair w, at iteration 0, the traffic demand r_w can be assigned to a few (i.e., $O(1)$ in number) minimum hop paths.

From this assumption and (13), the number of hops $h_p(k)$ for any active path p at any iteration $k \ge 0$ is bounded by the inequality

$$h_p(k) \le \gamma h_{\min} \tag{14}$$

where $h_{\min}$ is the diameter of the network, which is the maximum of the minimum hop distance between any two distinct nodes in the network. Then, from (9), (10), and (14), for any active path p at iteration k, its transformed FDL and SDL are bounded by

$$\tilde{d} \le \gamma D'_{\max} h_{\min}, \quad \tilde{d}^2_{pq}(k) \le 2\gamma D''_{\max} h_{\min}. \tag{15}$$

The next lemma is an immediate consequence of the GP algorithm, see [3], [5]; the proof is included in the Appendix for completeness.

Lemma 1: For all $w \in W$, $k \ge 0$

$$\sum_{p\in P_w, p\neq\overline{p}_w(k)} (d_p(k) - d_{w,\overline{p}_w(k)}(k))s_p(k) \le -\frac{\ell}{\alpha}\sum_{p\in P_w, p\neq\overline{p}_w(k)} |s_p(k)|^2.$$

[5] An active path is a path with a positive flow.

From Lemma 1 and (11), we have

$$\tilde{D}(\tilde{x}(k+1)) \leq \tilde{D}(\tilde{x}(k)) - \left[\frac{\ell}{\alpha} - \frac{L}{2}\right]||\tilde{s}(k)||^2 \tag{16}$$

where L is a bound for the spectral radius of the symmetrical positive semi-definite Hessian matrix $\nabla^2\tilde{D}(z(k))$. Thus, by choosing an appropriate stepsize α, say, for some positive constants C_1 and C_2, $0<C_2 \leq C_1<1$

$$C_2\frac{2\ell}{L} \leq \alpha \leq C_1\frac{2\ell}{L} \tag{17}$$

we guarantee that for all $k \geq 0$, $D(x(k+1)) \leq D(x(k))$. This stepsize will depend on n. It turns out that the complexity is intimately related to the ratio of L to ℓ, we define this ratio to be Q

$$Q \stackrel{\text{def}}{=} \frac{L}{\ell} = \theta(\alpha^{-1}) = \theta(L). \tag{18}$$

This ratio of largest to smallest eigenvalues is well known to be an important determinant of the computational effort. For example, see [12].

Let $T_I(n)$ denote the number of iterations needed for the GP algorithm to converge to within $(1+\varepsilon)D^*$. Among all the network parameters, we will highlight $h_{\min}$ and $N_{\max}$, where $N_{\max}$ is the number of paths sharing the maximally shared link.

From our experience with the GP algorithm, we have observed that, in general, the larger the maximum link utilization, the larger the number of iterations needed for the algorithm to converge. However, we will not highlight the dependency of the complexity bounds on $\rho_{\max}$. We do this because, in the quadratic-like cost function (5), $\rho_{\max}$ is typically chosen independent of n. In addition, for realistic data networks, the final maximum link utilization is always reasonably bounded away from unity and our analysis is well suited for such situations. Note also that, from (14), the largest number of links in any active path will be in the order of the network diameter.

Now the complexity result can be stated as follows.

Theorem 1: If the stepsize is chosen according to (17), then for any $\varepsilon>0$, $n \geq 0$, the number of iterations needed for the GP algorithm to converge to within $D^*(1+\varepsilon)$ is bounded by $T_I(n) \leq O(\varepsilon^{-2}Q) = O(\varepsilon^{-2}h_{\min}N_{\max})$.

IV. Proof of the Complexity for the Usual GP Algorithm

The proof of Theorem 1 will be completed by using lemmas 1, 2, and 3.

Lemma 2:

$$L \leq \gamma D''_{\max} h_{\min} N_{\max}.$$

Proof: Let $H_p(k)$ denote the Hessian matrix $\partial^2\tilde{D}(z(k))/\partial x^2$, and let $H_L(k)$ denote the Hessian matrix $\partial^2\overline{D}(f^z(k))/\partial f^2$, where $f^z(k) = E_a(k)\tilde{z}(k)$, $E_a(k)$ denotes the path-link incidence matrix between the set of link flows and the set of path flows for paths in $\tilde{P}(k)$, i.e., active paths which are not the chosen shortest paths. Note that $H_L(k)$ is a diagonal matrix. It is easy to see that $H_p(k) = E_a^T(k)H_L(k)E_a(k)$, where the ^{T}-superscript denotes matrix transpose. Let $\rho(M)$ denote the spectral radius of the square matrix M

$$\begin{aligned}\rho(H_p(k)) \leq& \rho(H_L(k))||E_a^T(k)E_a(k)||_\infty \\ \leq& \rho(H_L(k))||E_a^T(k)||_\infty||E_a(k)||_\infty\end{aligned}.$$

Now use the following relations:

$$\begin{aligned}||E_a^T(k)||_\infty &\leq \gamma h_{\min} \\ ||E_a(k)||_\infty &\leq N_{\max} \\ \rho(H_L(k)) &= D''_{\max}\end{aligned}$$

the lemma is proved. **Q.E.D.**

From the way we choose our stepsize (17) and Lemma 2, we have

$$\frac{1}{\alpha} = \theta(h_{\min}N_{\max}). \tag{19}$$

Lemma 3: There exists a positive constant A_1, independent of n, such that

$$D(x(k)) - D^* \leq A_1\frac{\Delta}{\alpha}\sqrt{|W|}||\tilde{s}(k)||.$$

Proof: By convexity and the fact that $\tilde{d}_p(k) \geq 0$ $x_p^* \geq 0$,

$$\begin{aligned}D(x(k)) - D^* \leq& \sum_{w\in W}\sum_{p\in P_w, p\neq\overline{p}_w(k)}(d_p(k) - d_{\overline{p}_w}(k)) \\ &\cdot (x_p(k) - x_p^*) \\ \leq& \sum_{w\in W}\sum_{p\in P_w, p\neq\overline{p}_w(k)}(d_p(k) - d_{\overline{p}_w}(k))x_p(k) \\ =& \sum_{w\in W}\sum_{p\in P_w, x_p(k+1)=0}(d_p(k) - d_{\overline{p}_w}(k)) \\ &\cdot (x_p(k) - x_p(k+1)) \\ &+ \sum_{w\in W}\sum_{p\in P_w, x_p(k+1)>0, p\neq\overline{p}_w(k)} \\ &\cdot \frac{m_p(k)}{\alpha}(x_p(k) - x_p(k+1))x_p(k) \\ \leq& \sum_{w\in W}\sum_{p\in P_w, x_p(k+1)=0} \\ &\cdot |\tilde{d}_p(k)||x_p(k) - x_p(k+1)| \\ &+ \sum_{w\in W}\sum_{p\in P_w, x_p(k+1)>0, p\neq\overline{p}_w(k)} \\ &\cdot \frac{\Delta}{\alpha}|x_p(k) - x_p(k+1)||x_p(k)|. \qquad (20) \\ \leq& \sum_{w\in W}\sum_{p\in P_w, p\neq\overline{p}_w(k)} \\ &\cdot \max\left\{\tilde{d}_p(k), \frac{\Delta}{\alpha}x_p(k)\right\}|x_p(k) - x_p(k+1)| \\ \leq& \max\left\{\frac{\gamma D'_{\max}}{\Delta}, r_{\max}\right\}\frac{\Delta}{\alpha}\sum_{w\in W} \\ &\cdot \sum_{p\in P_w, p\neq\overline{p}_w(k)}|x_p(k) - x_p(k+1)| \qquad (21)\end{aligned}$$

where in inequality (21), we use the facts $\tilde{d}_p(k) \leq \gamma D'_{\max}h_{\min}$ [c.f. (15)], $(1/\alpha) = \theta(h_{\min}N_{\max})$ [from (19)], $|x_p(k)| \leq$

$r_w = \theta(1)$ for $p \in P_w$, and $r_{\max} \stackrel{\text{def}}{=} \max\{r_w\colon w \in W\}$. The lemma now follows using Holder's inequality and the assumption that there are $\theta(1)$ number of active paths per each OD pair. **Q.E.D.**

Proof of Theorem 1: Combining (16) and Lemma 3, we get

$$D(x(k+1)) - D(x(k)) \le -\left(\frac{\ell}{\alpha} - \frac{L}{2}\right)\frac{(D(x(k)) - D^*)^2}{\left(A_1\dfrac{\Delta}{\alpha}\right)^2 |W|}. \tag{22}$$

From assumption (A), there exists $C_3 > 0$ such that $D^* \ge C_3|W|$. Whenever $D(x(k)) - D^* \ge D^*\varepsilon$

$$D(x(k+1)) - D(x(k)) \le -\alpha\frac{\ell(1-C_1)C_3}{A_1^2\Delta^2}D^*\varepsilon^2. \tag{23}$$

In inequality (23), we have used the fact $C_1 < 1$ and the inequality (17). Now the complexity estimate follows. **Q.E.D.**

Note that the inequality (22) also shows that, in order for $D(k) - D^*$ to be below $\varepsilon(D(x(0)) - D^*)$, it suffices that k be greater than the inverse of $A_2(D(x(0)) - D^*)\varepsilon^2$, where A_2 is a constant depending on n. Hence, the complexity of obtaining a factor of ε improvement scales inversely with the initial cost.

V. Complexity of the OSA Policy

The overall time complexity of the GP algorithm can be estimated by the product of the iteration complexity and the time complexity of each iteration. So far, in this paper, we have focused on the iteration complexity since the complexity of each iteration is trivial to estimate. It is easy to see that the time complexity of each iteration is dominated by the shortest path computation. This complexity is bounded by $O(n^3)$ since the known fastest sequential algorithm for the all-pair shortest path problem requires $O(n^3)$ amount of computation efforts.

According to Bertsekas [2], there exist two variations of the update policy for the GP algorithm. The usual policy is that, at each iteration, the path flows for all the OD pairs are updated; this policy will be referred to as the AAA policy. An alternate policy is that, at each iteration, the path flows for the OD pairs corresponding to only one origin node are updated, and the origin nodes take turn in a round-robin fashion to participate in these iterations; this policy will be referred to as the OSA policy. Bertsekas [2] has commented that the OSA policy converges in less time overall, even though the number of iterations is more for the OSA policy. In fact, a fast sequential code for the GP algorithm developed at MIT [9] uses the OSA policy, and the OSA policy has consistently shown faster convergence from the experience of the authors of [9]. A reason for this phenomenon suggested by Bertsekas [2] is that the OSA policy changes less amount of flows and the Taylor series expansion becomes more accurate, and the algorithm can afford a larger stepsize without inducing divergence.

Let us also assume, in this section, that we will use an explicit bound for $N_{\max}$. Assume that the number of OD pairs is $O(n^2)$ and that each OD pair has $\theta(1)$ number of active paths at each iteration, then

$$N_{\max} = O(n^2).$$

For the AAA policy, the iteration complexity will be $O(n^2 h_{\min})$, while the time complexity for each iteration will be $O(n^3)$. Thus, the overall time complexity will be $O(n^5 h_{\min})$. It turns out that the iteration complexity bound for the OSA policy is $O(n)$ larger than that of the AAA policy (see Theorem 2 below). Since the computation complexity for each iteration of the OSA policy is $O(n^2)$ (only one-source shortest path problem is to be solved per iteration), the overall complexity for the OSA policy is $O(n^5 h_{\min})$, which is the same as that of the AAA policy. This result is not surprising since the difference between the OSA and the AAA policies is similar to the difference between the Gauss–Seidel and Jacobi updates in the general iterative algorithm [11]. It is known that Gauss–Seidel method is twice faster than the Jacobi method for the one-dimensional discretized Poisson equations [11].

In this subsection, we shall use subscript s to denote quantities associated with the OSA policy. Let L_s denote an upper bound for the spectral radii of the *effective* Hessian matrices for the OSA policy, i.e. the matrix $\nabla^2\tilde{D}(z(k))$ restricted to the set of active paths updated at an OSA iteration. Since there are only $\theta(1)$ number of active paths per OD pair, it is obvious that

$$L_s = O(L/n).$$

Choose α_s, the stepsize for the OSA policy

$$C_2\frac{2\ell}{L_s} \le \alpha_s \le C_1\frac{2\ell}{L_s}. \tag{24}$$

Thus, we have

$$\frac{1}{\alpha_s} = \theta(nh_{\min}). \tag{25}$$

Letting Q_s denote the ratio L_s/ℓ, we have $Q_s = O(Q/n)$. Let $T_s(n)$ denote the number of iterations needed for the GP algorithm with OSA update policy to converge to within $D^*(1+\varepsilon)$. To find the iteration complexity for the OSA policy, we will prove a lemma similar to Lemma 3. Fix any $k \ge 0$, we will focus on n consecutive iterations grouped together in the form: $k, k+1, \cdots, k+n-1$. For any active path p updated at iteration $k+i$, let $\tilde{d}_{p,\,s}$ denote the transformed FDL for path p at the iteration $k+i$; we also write

$$x_{p,\,s}(k) \stackrel{\text{def}}{=} x_p(k+i) = x_p(k)$$

and similarly, we write

$$s_{p,\,s}(k) \stackrel{\text{def}}{=} s_p(k+i) = x_p(k+i+1) - x_p(k+i).$$

Lemma 3a: Suppose the OSA policy is adopted. There exists a positive constant A_3, independent of n, such that

$$D(x(k)) - D^* \le \frac{A_3\Delta n}{\alpha_s}\sqrt{|W|}\|\tilde{s}_s(k)\|.$$

Proof: Following the same initial inequalities in the proof of Lemma 3

$$D(x(k)) - D^* \leq \sum_{p \in \tilde{P}(k)} \tilde{d}_p(k) x_p(k) \leq \sum_{p \in \tilde{P}(k)} \tilde{d}_{p,s}(k) x_{p,s}(k) + \sum_{p \in \tilde{P}(k)} (\tilde{d}_p(k) - \tilde{d}_{p,s}(k)) x_{p,s}(k). \quad (26)$$

Now the first term in the right-hand-side of (26) is similar in form to (20) and we can apply the proof technique of Lemma 3 to get

$$\sum_{p \in \tilde{P}(k)} \tilde{d}_{p,s} x_{p,s}(k) \leq \frac{A_1 \Delta}{\alpha_s} \sqrt{|W|} \|\tilde{s}_s(k)\|. \quad (27)$$

The second term in (26) is bounded as follows. By the Mean Value Theorem, we can write, for any p

$$\tilde{d}_{p,s} = \tilde{d}_p(k) + \sum_{q \in \tilde{P}(k),\, q<p} \tilde{d}_{p,q}(z^p(k)) s_{q,s}(k) \quad (28)$$

where $z^p(k)$ lies in the line segment formed by joining $x(k)$ and $x(k+i)$, $x_p(k)$ is assumed to be updated at iteration $k+i$, and the notation $q<p$ means that path q is updated at an iteration before p's iteration, i.e., before iteration $k+i$.

We can express the second term in (26) as

$$\sum_{p \in \tilde{P}(k)} (\tilde{d}_p(k) - \tilde{d}_{p,s}(k)) x_{p,s}(k) = - \sum_{p \in \tilde{P}(k)} \sum_{q \in \tilde{P}(k),\, q<p} \tilde{d}_{pq}(z^p(k)) s_{q,s}(k) x_{p,s}(k) = -\tilde{x}_s(k)' \tilde{H}_s(k) \tilde{s}_s(k),$$

where $\tilde{H}_s(k)$ is the special Hessian matrix $\{\tilde{d}_{pq}(z^p(k)) : q < p\}$. Thus

$$<\tilde{x}_s(k)' \tilde{H}_s(k) \tilde{s}_s(k) = -\tilde{H}_s^T(k) \tilde{x}_s(k)' \tilde{s}_s(k) \leq \|\tilde{H}_s^T(k) \tilde{x}_s(k)\| \, \|\tilde{s}_s(k)\|. \quad (29)$$

Since the number of active paths in $\tilde{P}(k)$ is assumed to be $O(|W|) = O(n^2)$ and $\tilde{d}_{pq}(z^p(k)) \leq A_6 h_{\min}$, where $A_4 = 2\gamma D''_{\max} h_{\min}$ [cf. (15)]

$$\|\tilde{H}_s^T(k) \tilde{x}_s(k)\| \leq A_7 n^3 h_{\min}$$

where A_7 is a positive constant independent of n. Thus

$$\sum_{p \in \tilde{P}(k)} (\tilde{d}_p(k) - \tilde{d}_{p,s}(k)) x_{p,s}(k) \leq A_5 n^3 h_{\min} \|\tilde{s}_s(k)\| \leq A_6 \frac{\Delta n \sqrt{|W|}}{\alpha_s} \|\tilde{s}_s(k)\| \quad (30)$$

where A_6 is some positive constant independent of n. The lemma follows by combining (27) and (30) with $A_3 = A_1 + A_6$ and noting (25). **Q.E.D.**

TABLE I
DATA POINTS $T_I(n)$ versus $F(n)$

n	$T_I(n)$	$F(n)$	R
169	36	9.21E+7	3.91E-07
289	66	4.70E+8	1.41E-07
256	96	8.22E+8	1.17E-07
324	83	8.70E+8	0.95E-07
361	69	1.13E+9	0.61E-07
225	78	1.75E+9	0.46E-07
441	83	1.82E+9	0.46E-07
400	121	2.71E+9	0.45E-07

Theorem 2: If the stepsize is chosen according to (25) and the OSA policy is adopted, then for any $\varepsilon > 0$, $n \geq 0$, the number of iterations needed for the GP algorithm to converge to within $(1+\varepsilon)D^*$ is bounded by $T_I(n) \leq O(\varepsilon^{-2} n^2 Q_s) = O(\varepsilon^{-2} n^3 h_{\min})$.

The proof of Theorem 2 can be followed almost verbatim from the proof of Theorem 1 and we omit the proof.

The result in this section also implies the benefit of parallelizing the optimal routing algorithm over the network nodes. The AAA policy can be easily parallelized since at each iteration, the updating of path flows can be done at each origin of the OD pairs, and each origin node only has to solve a single-source shortest path problem. Thus, this parallelized version of the AAA–GP algorithm will have an overall time complexity of $O(n^4 h_{\min})$, showing a perfect speed-up of $O(n)$ with $O(n)$ parallel processors. This parallelization will not be attractive if the OSA policy has an overall time complexity of $O(n^4 h_{\min})$, because the sequential OSA–GP algorithm would have the same time complexity while using only one processor. The complexity bound derived in this section suggests that this latter scenario is unlikely and the parallelized AAA–GP algorithm is efficient.

VI. NUMERICAL RESULTS

In this section, we present some numerical results which demonstrate the usefulness of our theoretically derived bounds. The symbols used in Table I are:

- n number of node
- $T_I(n)$ number of iteration
- $F(n)$ estimate function $h_{\min} N_{\max}/\varepsilon^2$
- R ratio $T_I(n)/F(n)$

We wrote a program to generate mesh networks with 0.5 probability that a single node will be an OD node. An OD node is a node which is both an origin and a destination. The link capacities and demands are randomly generated.

The data points $(T_I(n), F(n))$, where $F(n) = \varepsilon^{-2} h_{\min} N_{\max}$, for the networks are plotted in Fig. 1 and shown in Table I.

Note that R will be an estimate for the constant in the $O(\varepsilon^{-2} h_{\min} N_{\max})$ notation. From Table I and Fig. 1, it is clear that the ratio remains relatively unchanged. This results show that the theoretically derived bound $T_I(n) = O(\varepsilon^{-2} h_{\min} N_{\max})$ is an excellent predictor of the trend in the number of iterations as the node number increases. The tiny ratio R (hence, a small constant in the $O(\cdot)$ notation) is

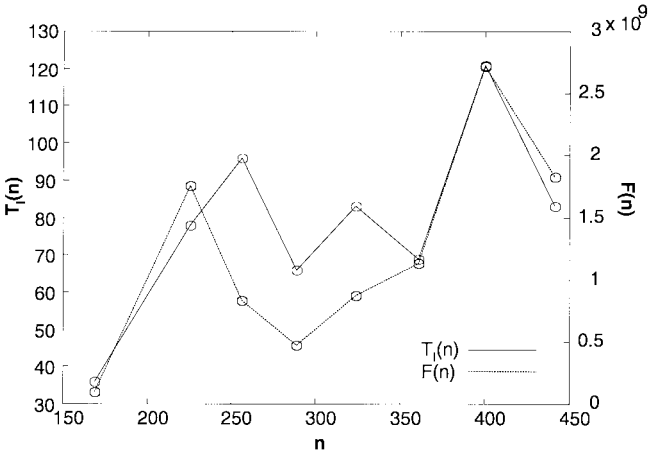


Fig. 1. $T_I(n)$ versus $F(n)$.

attributed to the large value of ε^{-2}. The pleasant surprise is that the trend is well predicted and the trend (or the derivative) is what is the most wanted in an asymptotic complexity analysis.

VII. Conclusion

In this paper, we have studied the time-complexity bounds for the GP method for optimal routing in data networks. Our result shows that the GP algorithm of the Goldstein–Levitin–Poljak type formulated by Bertsekas converges in $O(h_{\min}N_{\max})$ number of iterations. We also show that OSA update policy has the same overall complexity as that of AAA update policy. This result also implies that parallelizing the optimal routing algorithm over the network nodes is beneficial.

One should interpret the results carefully. Complexity bounds are worst-case estimates; it does not mean that these bounds are actually attained in many real-world problems. These bounds do tell us, however, that the number of OD pairs, the network diameters, and the number of active paths per OD pair could be the dominating factors determining the iteration complexity. The main complexity bound can be put into another form, $T_I(n) \leq O(\alpha^{-1}) = O(Q) = O(L\ell^{-1})$, which tells us that the inverse of the stepsize is the critical factor. Often, one can choose a small stepsize and let the GP algorithm runs—if the algorithm does converge, then the computational efforts are proportional to the inverse of the chosen stepsize, otherwise, one can reduce the stepsize and keep on trying. From our implementation experiences, the computational efforts are weakly dependent on the number of OD pairs and more dependent on the network diameters. The fact that the complexity bounds are proportional to the ratio of the maximum eigenvalue estimate to the minimum eigenvalue estimate of the Hessian matrices is also very interesting. This result implies the following: if the active paths are more or less mutually independent, then the ratio will tend to be small, and algorithm will converge faster. On the other hand, if all the active paths are coupled in a complicated way, then the ratio will tend to be large, and the algorithm will tend to be slow. In the former case, the stepsize can be chosen large, while in the latter case, the stepsize should be chosen small. Another obvious observation is that, if the network is heavily loaded, then the stepsize should be chosen small and the algorithm will converge slowly, as some elements of the Hessian matrices will tend to be large.

An interesting application of Lemma 3 is that the Euclidean norm of the transformed step $\tilde{s}(k)$ can be used as a stopping criterion for the GP algorithm. Lemma 3 implies that the l_2 norm of the transformed step can be used to define level sets for the cost function. An actual stopping measure should also be normalized so that the measure is independent of the problem size.

Finally, the result of this paper argues for constructing networks with low diameter for the purpose of reducing complexity of the network control algorithms. This comes from the observation that, among all the terms in the complexity bounds, only $h_{\min}$ is at the disposal of the network designer. In fact, by observing most practical networks, one easily discovers that the network diameter is usually a very slowly increasing function of the network size, showing that the network designers have been making the right decisions.

Appendix

Proof of Lemma 1: From the GP algorithm

$$0 \leq -s_p(k) \leq \frac{\alpha}{m_p(k)}\,\tilde{dl}.$$

Thus

$$-\sum_{p\in P_w,\, p\neq \overline{p}_w(k)} (d_p(k) - d_{w,\overline{p}_w(k)}(k))s_p(k) \geq \sum_{p\in P_w,\, p\neq \overline{p}_w(k)} |s_p(k)|^2 \frac{m_p(k)}{\alpha}$$

with that $\ell \equiv \inf\{m_p(k)\}$ in (7) implies the lemma. **Q.E.D.**

References

[1] A. S. Nemirovsky and D. B. Yudin, *Problem Complexity and Method Efficiency in Optimization.* New York: Wiley, 1983.
[2] 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.
[3] W. K. Tsai, "Convergence of gradient projection routing methods in a distributed asynchronous stochastic quasistatic virtual circuit network," *IEEE Trans. Automat. Contr.*, vol. 34, pp. 20–23, Jan. 1989.
[4] W. T. Tsai, C. V. Ramamoorthy, W. K. Tsai, and O. Nishigushi, "A new adaptive hierarchical routing protocol," *IEEE Trans. Comput.*, vol. 38, pp. 402–407, Aug. 1989.
[5] J. N. Tsitsiklis and D. P. Bertsekas, "Distributed asynchronous optimal routing in data networks," *IEEE Trans. Automat. Contr.*, vol. AC-31, pp. 325–332, 1986.
[6] L. Kleinrock, *Communication Nets: Stochastic Message Flow and Delay*. New York: McGraw-Hill, 1964.
[7] R. G. Gallager, "A minimum delay routing algorithm using distributed computation," *IEEE Trans. Commun.*, vol. COM-25, pp. 73–85, Jan. 1977.
[8] D. P. Bertsekas and R. G. Gallager, *Data Networks.* Englewood Cliffs, NJ: Prentice-Hall, 1987.
[9] D. P. Bertsekas, B. Gendron, and W. K. Tsai, "Implementation of an optimal network flow algorithm based on gradient projection and a path flow formulation," MIT, Cambridge, MA, LIDS Rep. p-1364, Feb. 1984.
[10] D. G. Luenberger, "The gradient projection method along geodesics," *Management Science*, vol. 18, no. 11, pp. 620–631, July 1972.

[11] R. S. Varga, *Matrix Iterative Analysis*. Englewood Cliffs, NJ: Prentice-Hall, 1962.
[12] D. G. Luenberger, *Introduction to Linear and Nonlinear Programming*. Reading, MA: Addison-Wesley, 1984.
[13] J. K. Antonio, W. K. Tsai, and G. M. Huang, "Time complexity of a path-formulated optimal routing algorithm," *IEEE Trans. Automat. Contr.*, vol. 39, pp. 1839–1844, Sept. 1994.
[14] J. M. Ortega, *Numerical Analysis—A Second Course*. New York: Academic, 1972.

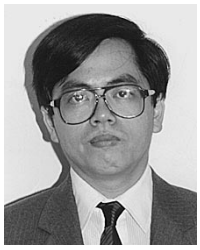
Wei K. Tsai (M'86) received the B.Sc., M.Sc., E.E., and Ph.D. degrees from Massachusetts Institute of Technology, Cambridge, MA.

He is currently an Associate Professor of Electrical and Computer Engineering at the University of California at Irvine. His research interests include flow control, traffic management, and routing for high-speed data networks, digital signal-processing, control, and optimization theory. He has co-authored over 100 publications in the above and related areas.

John K. Antonio (S'88–M'89–SM'98) received the B.S., M.S., and Ph.D. degrees from Texas A&M University, College Station, TX.

He is currently Professor and Director of the School of Computer Science, University of Oklahoma at Norman. His current research interests include: parallel and distributed computing, embedded high-performance computing, reconfigurable computing, and cluster computing. He has co-authored over 60 publications in the above and related areas.

Dr. Antonio was Program Chair for the 1998 Heterogeneous Computing Workshop, sponsored by the IEEE Computer Society, and was General Chair for the 1999 Heterogeneous Computing Workshop. For five years, he has chaired and organized the Industrial Track and Commercial Exhibits portions of the IEEE International Parallel Processing Symposium. He is a member of Tau Beta Pi, Eta Kappa Nu, and Phi Kappa Phi honorary societies.



Garng M. Huang (S'76–M'80–SM'85) received the B.S. and M.S. degrees in electrical engineering from National Chiao Tung University, Hsinchu, Taiwan, R.O.C., in 1975 and 1977, respectively. He received the doctorate degree in systems science and mathematics from Washington University, St. Louis, MO, in 1980.

After teaching at Washington University until 1984, he joined the Department of Electrical Engineering, Texas A&M University, College Station, TX. He is currently a Professor and the Director of Graduate Studies. He has been working on many funded research projects, such as emergency control of large interconnected power system, HVDC systems, restoration of large-scale power systems, on-line detection of system instabilities, on-line stabilization of large power systems, fast parallel/distributed textured algorithms, fast parallel textured algorithms for large power systems, hierarchical aggregation and decomposition algorithm for data network routing problems, and intelligent agents for control of large systems. His current research interest is in large-scale systems theory, large-scale parallel/distributed computing and control, and their applications. He has published more than 100 papers and reports in the areas of nonlinear distributed control systems, parallel/distributed computing and their applications to power systems, data networks, and flexible structures. He is a Registered Professional Engineer of Texas.

Dr. Huang has served as the Technical Committee Chairman of Energy System Control Committee and as an Associated Editor for the IEEE Automatic Control Society. He has also served in a number of committees and subcommittees of the IEEE Power System Society.