Routing in multidomain networks — **Source link**

Dragomir Dimitrijević, B. Maglaris, Robert R. Boorstyn

Institutions: National Technical University of Athens, New York University

Related papers:

- Impact of complex network properties on routing in backbone networks

- Domain-disjoint routing based on topology aggregation for survivable multidomain optical networks

- Geographical Cluster-Based Routing in Sensing-Covered Networks

- Self-routing permutation networks for communications and computer systems

- End-to-end shared restoration algorithms in multi-domain mesh networks

Routing in Multidomain Networks

Dragomir D. Dimitrijević, *Member, IEEE*, Basil Maglaris, *Member, IEEE*, and Robert R. Boorstyn, *Fellow, IEEE*

Abstract— We investigate the problem of management and control in a large and, for simplicity, homogeneous packet-switched network. Specifically we focus on routing, an important function of network management. The network consists of several individually controlled domains. Domains are interconnected via gateway links. Each domain is controlled by its own Network Control Center, while the overall network performance is managed by an Integrated Network Control Center. Each Center has only a portion of the information required for global routing. We investigate the impact of the reduced information available at each center on network performance (average delay in our case). We present a general approach to designing a hierarchical algorithm for routing in multidomain networks. We propose a heuristic procedure suitable for packet-switched networks. Several numerical examples will illustrate the impact of incomplete information on the network. Performance is compared with a lower bound obtained, which is not differentiating destinations in other domains. Therefore, for this bound, each domain is perceived as a single node in a simplified model of the network.

I. INTRODUCTION

A. Multidomain Networks

IN THIS PAPER we investigate the problem of management and control in a large and, for simplicity, homogeneous packet-switched network. We focus on routing, an important issue in network management. We shall use average end-to-end delay over the entire network as a measure of network performance. This delay is assumed to be a separable function of the link delays. We model all links as independent $M/M/1$ queues, as commonly assumed in packet-switched networks [8]. The primary concern of early researchers was to design an efficient, static or dynamic, centralized or distributed, routing algorithm. However, management of today's computer communication networks involves different kinds of problems as shown in the following discussion.

1) Network Heterogeneity: Computer communication vendors and users are faced with growing communication systems that must handle voice, data, and soon video, on an integrated basis, eventually through an *Integrated Services Digital Network* (ISDN). A modern computer communication network is actually a complex interconnection of many constituent,
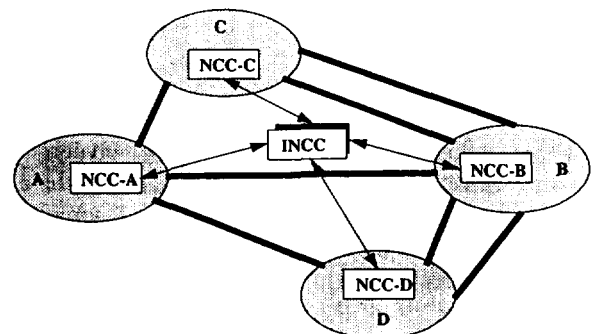
Fig. 1. A multidomain network.

loosely coupled subnetworks or domains. Generally, there can be different types of subnetworks such as packet-switched, *System Network Architecture* (SNA), $T1$ or voice networks. Those subnetworks usually are provided by different vendors. Each of the subnetworks must interface with other networks such as long-haul services from long distance carriers.

The management and control of these networks of interconnected heterogeneous domains is an increasingly important area. In such a heterogeneous environment, it is difficult to provide all the information needed without placing a burden on communication bandwidth and introducing communication, processing, and storage overhead. On the other hand, vendors and users must somehow monitor and control the operation of the entire system to guarantee an acceptable grade of service.

For that reason, each domain is controlled by its own *Network Control Center* (NCC). The management and control of such a system involves cooperation among different NCC's. The distributed approach involves peer-to-peer coordination of the various NCC's involved, while the hierarchical approach introduces an *Integrated Network Control Center* (INCC) that supervises NCC's and their interaction. AT&T's *Unified Network Management Architecture* (UNMA) [9] uses the evolving *International Organization for Standardization—Open Systems Interconnection,* (ISO-OSI) protocols for passing management information between a network element and an NCC and extends to the NCC-INCC interface.

2) Size of the Networks: Network routing algorithms generally attempt to provide communication among nodes by sending data messages along the best or shortest paths between them. Unfortunately, in large communication networks it is difficult to maintain knowledge of such paths due to the cost in storage, computation, and communication bandwidth.

A multidomain network, Fig. 1, consists of several individually controlled domains. Domains are interconnected via (possibly multiple) gateway links (heavy lines) and controlled by their own NCC's. We assume coordination among different

NCC's. For that reason, we allow a limited amount of information exchange between different NCC's. Due to the network size, we wish to reduce the amount of information exchange while still preserving reasonable network performance. Limitation on the available information raises the problem of myopic decision making. Coordination and information exchange between NCC's is achieved via the INCC. We assume that the INCC has overall, but still condensed, information about the entire network. It determines some guidelines as the basis on which each of the NCC's manages its own network.

The consequences of the reduced information exchange are:

1) Reduced communication overhead
2) Reduced processing and memory requirements in each NCC
3) Suboptimal network performance

B. Previous Work

Various network routing algorithms have been suggested in the literature as a means of reducing computational and storage complexity and communication overhead at the expense of network performance. Kamoun proposed [14], [15] using clustering and a suboptimal shortest path (nonbifurcated) routing algorithm. Even a large homogeneous network may be divided into several loosely coupled domains. The analysis has been generalized to k levels of clustering [11], [12]. Another algorithm, suitable for virtual circuits, was proposed by Baratz and Jaffe [1]. It finds optimal path lengths by introducing a limited broadcast of call-request packets during establishment of a session.

A practical example of hierarchical routing is *Digital Network Architecture* (DNA) [4]. A two-level hierarchical network can consist of up to 63 areas (domains in this paper's terminology) and each area can have up to 1022 nodes. The same distributed shortest path algorithm is performed in each of the two levels of hierarchy: area-to-area, and node-to-node (within an area). The link costs are predefined by a network implementor. A set of suggested link costs is provided by the vendor.

None of the aforementioned papers have analyzed overall network performance when the information about topology and traffic requirement is incomplete. Unlike in [4], [14], [15], we allow that packets originated from one, and destined to another node, may be bifurcated across multiple routes. A lower bound to the average delay in a network consisting of two individually controlled domains was analyzed in [5]. This algorithm was later generalized [6] to handle more than two domains. We shall use this algorithm as a lower bound to the network performance subject to the constraint that the interdomain traffic must be routed via gateway links independently of the particular destination node. It was shown that despite the additional constraints, the entire requirement matrix is needed to achieve a lower bound of network performance while keeping link flows consistent. By "inconsistency" we mean discrepancy between expected and actual link flows enforced by actual network operation. Inconsistency is caused by the lack of information about the network. An alternative, heuristic approach to this problem was proposed in [3]. This algorithm starts from a tentative routing, and attempts to improve the network performance after additional measurements of gateway costs are obtained. However, it does not propose how to establish the initial routing. Unlike [3] and [5], in the procedure we shall develop here, NCC's will configure their routes based on inconsistent estimates for the internet flows. We shall compare three estimation policies used to estimate the amount of traffic that each node can expect from each gateway link coming to the domain. The initial routing is improved after the network acquires more information about actual network flows. A more detailed study of the problem considered in this paper can be found in [6]. A condensed version of this paper was published in [7].

C. Paper Outline

Section I of this paper contains the introduction. Section II contains a description of operations performed by the INCC. A procedure for routing in multidomain networks is outlined. It involves the *High Level Routing* (HLR) performed by the INCC as well as the *Low-Level Routing* (LLR) performed by the NCC's in each domain individually. Section III concentrates on the LLR and local performance optimization subject to the constraints imposed by the INCC and HLR and with incomplete information available. A way to simplify the model of a domain through *equivalent capacities* associated with each gateway link will be given. The equivalent capacities are a way to convey information about local congestions in a domain without giving too many internal details, and thus introducing an excessive overhead. It will be shown how a domain can improve its local performance after acquiring more information in the learning period, which we call *a posteriori routing*. Section IV contains numerical examples and illustrates the impact of the unavailable information on the network performance. Section V contains the conclusion.

II. OPERATIONS PERFORMED BY THE INCC

A. Properties of Routing in Multidomain Networks

In the following section we shall describe a hierarchical routing algorithm and determine the amount of information available from a practical point of view to each of the network parts. In order to design an applicable routing algorithm, we impose some a priori constraints on available information, as well as on some properties of the algorithm.

1) Node Independent Internet Information: Any information on traffic across domains is independent of specific nodes. This means that the amount of information about the internet is independent of the size of individual domain. Each domain does not distinguish nodes in other domains, i.e., all messages destined to the same domain are considered as a single commodity. This property implies destination independent routing and more compact routing tables in each node.

2) Loop-Free Routing: The resulting routing must be loop-free. To prevent loops, we impose an additional constraint, since the detailed internal structure of a domain is not globally known. We insist that each domain on a route can be visited

only once. This means that the routing viewed on a domain-to-domain basis is also loop-free, although a loop on a domain-to-domain basis does not necessarily mean a loop in the actual path. Since each domain knows all the details about its internal structure, a loop-free route within the domain can be provided.

3) Destination Dependent Statistics: Each entity in the network keeps statistics necessary for the routing algorithm and reports to a higher level. For example, nodes keep statistics about generated traffic from each internal node to an external domain and reports them to the NCC, NCC's report the condense information about the internet requirements to the INCC.

4) Limited Internet Information Exchange: We allow only a small amount of internet information exchange. To achieve this goal, we allow a small number of iterations that involve changes in the overall routing. Excessive amount of global information exchange may not be manageable from a practical point of view.

5) Parallel Work of NCC's: All NCC's should work independently of each other as much as possible to achieve maximum parallelism.

6) Processing and Memory Efficiency: The algorithm must be provably more efficient than the centralized routing scheme. This involves computational efficiency and the size of routing tables in all nodes.

7) Memoryless Routing: We assume that all messages are handled based on their destination, regardless of the previous nodes that are visited. Many systems use this kind of routing. Other systems (e.g., bridged local area networks) apply source routing implementations [13] where the source of a packet specifies the entire route from the source to the destination.

B. A Procedure for Routing in Multidomain Networks

The following section gives a brief description of a hierarchical algorithm for routing in multidomain networks. We divide the problem of designing the hierarchical routing algorithm into different parts to be studied separately.

1) **Initialization:** The INCC initializes the internet (domain-to-domain) routing.

2) **High-Level routing:** Based on a simplified configuration of the network, the INCC determines routes for all internet requirements. After the high-level routing is determined, each of the NCC's is informed about the INCC's decision concerning the routing in the corresponding domain.

3) **Low-Level routing:** Based upon the information about the states of its gateway links received from the INCC, each NCC decides on its local routing. This part of the algorithm will be described in Section III.

4) **Computation of gateway penalties:** Each of the NCC's computes penalties for each of its gateway links in order to avoid congested areas in the network. Note that no matter how the penalties are obtained, each of the gateway links is characterized by a single number which is sent to the INCC. A heuristic approach to the computation of the penalties will be given in Section III.

5) **Stopping criterion:** Steps 2)–4) are repeated until a stopping criterion is satisfied. In our case, the INCC terminates the procedure when the relative difference between two consecutive values of its objective function becomes smaller than 1%. Since the algorithm is a heuristic one and there is no proof that it converges, an upper limit to the number of iterations must be imposed.

6) **A posteriori routing:** The initial routing is readjusted after a learning period when more information about the network is acquired. This part of the algorithm will be discussed in Section III.

C. Computational and Memory Complexity

Let n be the number of nodes in a multidomain network and α be a constant between 0 and 1. We shall denote the number of domains in the network as $d = O(n^\alpha)$, and the average number of nodes per domain as $s = O(n^{1-\alpha})$ ($n = sd$). Generally, the order of computational complexity of a routing algorithm (shortest path, flow deviation, extermal flow) is $O(n^3)$ [2], and the size of the routing tables is $O(n)$. The order of complexity of the high-level routing is $O(d^3)$. The order of complexity of the low-level routing performed in each domain is $O(s^3)$. Since all NCC's perform their low level routing algorithm simultaneously the overall computational complexity is equal to the sum of the two parts. The minimum complexity, $O(\sqrt{n^3})$, is achieved when $\alpha = 0.5$. This is significantly better than $O(n^3)$ for the global optimum. Similarly, the minimum memory complexity, $O(\sqrt{n})$, is achieved for $\alpha = 0.5$.

D. High-Level Routing

We assume that the INCC has some topological information about the domains. The information is restricted to existence of the domains and their connectivity via gateway links. This means that the topology of the internet is known. In the graph that describes the internet topology, each node corresponds to a domain in the network, while each directed arc corresponds to a gateway link.

To be more specific, we allow the following information about the network:

1) Each gateway link g is characterized by its *equivalent capacity* (EC) C'_g obtained from the domains that are adjacent to gateway link g. The equivalent capacity is a measure of congestion on both sides of the corresponding gateway link. In our case, we reduce the original gateway capacity to a valve depending on the severity of congestion. The reduction is a function of congestion in the domain. Gateway capacities leading to more congested areas are reduced more heavily, thus forcing traffic to avoid those areas.

2) We also assume availability of the average data rates for the domain-to-domain requirements. In a packet-switched network, it will simply be r_{IJ}, defined as the total amount of traffic rate that has to be sent from domain I to domain J. Each r_{IJ} is obtained from the NCC in the source domain I.

The INCC determines routes for all domain-to-domain requirements. To simplify the problem, we wish to make use of some of the well-known routing algorithms using the C'_g and r_{IJ} as input data. In this work, we apply the Flow Deviation Algorithm [8] by using the equivalent capacities of the gateway links instead of their real capacities. We used the actual gateway capacities as an initial value for the equivalent capacities.

The INCC's objective is to minimize "the average delay" using the equivalent capacities instead of real ones. The result of this algorithm is a set of f_{gJ} defined as the flow of messages in gateway link g that are destined for domain J. This number is reported to the domains adjacent to gateway link g and is used as a constraint in the low-level routing optimization. Note that the high-level (domain-to-domain) routing produced using this objective function is loop-free. By "loop-free high-level routing" we mean that none of the domains is visited more than once by the route for any source-destination pair. The loop-free property is a consequence of the Flow Deviation Algorithm, and the way the objective function is defined.

III. OPERATIONS PERFORMED BY NCC's

A. Formulation of the Problem

In this section we formulate the problem of low level routing. The low level routing algorithm optimizes a local objective function subject to the constraints imposed by the INCC. Based on the information received from the INCC, each NCC iteratively determines routes to the best of its knowledge. The routes are determined for each of the following four types of traffic:

1) Internal Traffic: This type of traffic is generated at and destined for a node within a domain, say domain I. The information about the internal traffic is local and known to NCC I.

2) Outgoing Traffic: This type of traffic is generated in the domain and destined for a node outside of the domain. The flows in each gateway link of the domain and destined for other domains are determined by the INCC.

3) Transit Traffic: This type of traffic is generated and destined outside of a domain and passes through the domain. The information about the total traffic in each gateway link destined for each domain is received from the INCC. Basically, this type of traffic can be handled as outgoing traffic generated in the incoming gateway node. Therefore, we can assume that there is no transit traffic. The outgoing requirements for the gateway nodes are modified to incorporate the amount of traffic coming from their incoming gateway links.

4) Incoming Traffic: This type of traffic is generated in a source node outside of a domain and destined for a node within the domain. It is the only type of traffic for which insufficient information is received from the INCC. The INCC provides to each domain only the total amount of traffic destined to it on each of the gateway links adjacent to the domain. This is not enough for the local routing since the incoming traffic must be routed to its actual destination node. Therefore, it is necessary to assume some breakdown per destination node and do a locally optimal routing for the assumed incoming traffic

pattern. We shall present three different estimation policies about the incoming traffic in Section III-C.

In the proposed algorithm, flows in the gateway links are controlled by the INCC and cannot be changed by NCC's. Flows in the gateway links are used as constraints for the low-level routing. For that reason we assume that the domain being considered, domain I, consists of its nodes and *internal links only*. The topology of such a graph and the capacities of the links are knows to NCC I. Furthermore, NCC I knows the following:

— the node-to-node requirements within the domain, r_{ij}
— the node-to-domain requirements, $r_{iJ}, I \neq J$
— the total incoming traffic destined to each node, r_{iI}
— the flows of messages, f_{gJ}, in all gateway links g entering domain I and destined for domain J. By $g \rightarrow i$ and $g \leftarrow i$ we denote gateway links g in and out of gateway node i, respectively.

The low-level routing problem can be formulated as follows. Minimize the local average delay, or more precisely, the average number of packets, D_I, defined as

$$D_I = \sum_{l \in I} \frac{f_l}{C_l - f_l} \tag{1}$$

such that the total link flows f_l belong to the set of feasible flows Ω, and $l \in I$ denotes the set of internal, i.e., non-gateway links l in domain I. We shall assume that the link flows always satisfy the capacity constraints ($f_l < C_l$). Note that now C_l denotes a real link capacity (not the equivalent capacity). The capacity constraints can be relaxed by extending each term in the sum (1) beyond a certain link utilization (99% in our case) using its second order Taylor series approximation.

The total flow T_i^J that a node i in I sends to domain J (including both locally and externally generated traffic) appears like local traffic generated in i and is computed as

$$T_i^J = \begin{cases} r_{iJ} & \text{internal node } i, J \neq I \\ r_{iJ} + \sum_{g \rightarrow i} f_{gJ} & \text{gateway node } i, J \neq I \\ \sum_{g \rightarrow i} f_{gI} & \text{gateway node } i, J = I \\ 0 & \text{internal node } i, J = I \end{cases} \tag{2}$$

By "gateway node" we mean a node adjacent to a gateway link. Note that we did not include the local requirements r_{iJ} of domain I in the third and fourth line of the right hand side of (2). This separates the traffic with uncertain destination (T_i^I) from that with certain destination (r_{ij}).

The total flow R_i^J that a node i in I receives and relays to domain J (that has not been generated in I) appears like local traffic destined to i. It is computed as

$$R_i^J = \begin{cases} R_i^I & \text{any node } i, J = I \\ 0 & \text{internal node } i, J \neq I \\ \sum_{g \leftarrow i} f_{gJ} & \text{gateway node } i, J \neq I \end{cases} \tag{3}$$

We define the following notation:

— P_{ij} is the set of paths between nodes i and j
— π_{ij}^P is the fraction of the internal traffic between nodes i and j on path p

— $\phi_{ij}^{p(J)}$ is the fraction of the internet traffic between nodes i and j on path p destined to domain J

— x_{ij}^J is the amount of the internet traffic that node i sends to node j and node j relays to domain J

The global flow in each link l can be determined as

$$f_l = \sum_{i,j} r_{ij} \sum_{p;l\in p} \pi_{ij}^p + \sum_J \sum_{i,j} x_{ij}^J \sum_{p;l\in p} \phi_{ij}^{p(J)} \tag{4}$$

The linear constraints are

$$\sum_{p\in P_{ij}} \pi_{ij}^p = 1 \qquad \forall i,j \tag{5}$$

$$\sum_{p\in P_{ij}} \phi_{ij}^{p(J)} = 1 \qquad \forall i,j,J \tag{6}$$

$$\sum_j x_{ij}^J = T_i^J \qquad \forall i,J \tag{7}$$

$$\sum_i x_{ij}^J = R_j^J \qquad \forall j,J \tag{8}$$

It can be shown [16] that if (i) the global minimum exists, and (ii) the objective is convex, then the global minimum can be found using a descent direction procedure such as the Flow Deviation Algorithm [8].

B. The Low-Level Routing Algorithm

After the formulation of the problem in the previous paragraph, we can outline the low-level routing algorithm as a constrained version of the Flow Deviation Algorithm. A formal proof that the algorithm converges to the global minimum of (1) subject to constraints (2–8) can be found in [6].

1) Initialize variables so they satisfy the constraints (2–8).
2) Find the shortest paths and their lengths d_{ij}. The length of a path is defined as a sum of lengths of the constituent links. As defined in the Flow Deviation Algorithm, link lengths are calculated as the first order derivatives of link delays (average number of packets) with respect to their link flows.
3) Establish the extremal flows:

   a) Establish the extremal flows due to the local traffic r_{ij} by sending it along the shortest paths
   b) Establish the extremal flows due to the transit traffic. We shall denote these flows as $\hat{x}_{ij}^J$. Find the set of $\hat{x}_{ij}^J$, for all $J \neq I$, that minimizes

$$\sum_{i,j} \hat{x}_{ij}^J d_{ij} \tag{9}$$

   subject to (7) and (8) as a linear transportation problem. Send the determined $\hat{x}_{ij}^J$ along the shortest paths.
   c) Establish the extremal flows due to the incoming traffic, i.e., find $\hat{x}_{ij}^I$ according to some estimation policy. Send the obtained $\hat{x}_{ij}^J$ along the shortest paths. This issue will be discussed in the next paragraph.

4) Add all flow components above [a), b), and c)] and obtain overall extremal flows v_l. Find an optimal λ such that the convex combination between old link flows f_l and extremal link flows v_l produces new link flows $f_l \leftarrow (1 - \lambda)f_l + \lambda v_l$ that minimizes the total number of packets in the local links.
5) Update x_{ij} as $x_{ij} \leftarrow (1 - \lambda)x_{ij} + \lambda\hat{x}_{ij}$ where $x_{ij} = \sum_J x_{ij}^J$.
6) Go to step 2 unless a stopping criterion is satisfied.

C. Estimation Policies

The following section explains how to determine $\hat{x}_{ij}^I$ and establish the extremal flows for the incoming traffic in step c) in the LLR. This step depends on the estimation policy. We consider three cases:

1) Best Case: We assume that the breakdown of the total incoming traffic will be the best possible for the domain's local performance. We wish to find x_{ij}^J such that

$$\min_{x_{ij}^I} \min_{x_{ij}^I; I\neq J} D_I^{\text{opt}} = \min_{x_{ij}^I} D_I^{\text{opt}} \tag{10}$$

where D_I^{opt} denotes minimum number of packets (minimum delay) in domain I for a particular set of x_{ij}^J. In the best case estimation policy, steps b) and c) of the LLR are equivalent, i.e., we have to minimize (9) subject to (7) and (8) for $J = I$ in step c).

2) Proportional Case: In this case, we assume that the incoming traffic from each of the gateway nodes i that is destined for a particular destination node j is proportional to the total incoming traffic from that gateway node

$$x_{ij}^I = \hat{x}_{ij}^I = \frac{T_i^I R_j^I}{S} \qquad i,j \in I \tag{11}$$

where S is the total incoming traffic

$$S = \sum_i T_i^I = \sum_j R_j^I \tag{12}$$

Therefore, the x_{ij}^I and $\hat{x}_{ij}^I$ are kept fixed during the execution of the LLR. The maximum entropy principle [17] can be used as a justification [6] for the proportional assumption. The experiments with randomly generated traffic showed that the proportional estimate generally gives better estimate since the best and worst case distributions of traffic requirements are not likely to occur.

3) Worst Case: In this case, we assume that the breakdown of the total incoming traffic from each of the gateway nodes will be the worst possible for the domain's local performance. We wish to find x_{ij}^J such that

$$\max_{x_{ij}^I} \min_{x_{ij}^J; I\neq J} D_I^{\text{opt}} \tag{13}$$

In this case, x_{ij}^I must be determined in a different way than the other two cases. The problem can be formulated as a nonlinear transportation problem that can be solved using the following algorithm.

Algorithm W: To find an optimal routing using the worst case assumption, we use the computational procedure for solving a transportation problem ([10, pp. 339–340]). In each iteration, the cost coefficients d_{ij} are linearized using the LLR and are assumed fixed.

1) Initialize x_{ij}^I by finding a basic solution B to the constraints (7) and (8) for $J = I$. If there are n nodes in the domain, there will be at most $2n - 1$ non-zero variables x_{ij}^I since the number of constraints (7) and (8) is $2n$, one of them being linearly dependent of the other $2n - 1$ constraints. We say that they belong to the basis B.

2) Perform the LLR while keeping x_{ij}^I fixed and $\hat{x}_{ij}^I = x_{ij}^I$.

3) Compute the shortest path lengths d_{ij} between all pairs of nodes i and j in I.

4) Solve a system of $2n - 1$ linear equations with $2n$ variables, a_i, b_i $(i = 1, 2, \ldots, n)$.

$$a_i + b_j = -d_{ij} \qquad \forall (i,j) \in B$$

5) Find p and q that will give

$$\max \langle (p, q) \in \bar{B} | a_p + b_q = d_{pq} \rangle$$

where $\bar{B}$ denotes the set of variables x_{ij}^I that are not in basic B.

6) If $a_p + b_q + d_{pq} \leq 0$ stop the algorithm. Otherwise go to the next step.

7) Introduce x_{pq}^I into the basis by giving it the maximum value so that the non-negativity of the x_{ij}^I is still preserved. A basic variable that becomes equal to zero is eliminated from the basis.

8) Go to step 2.

At the end of Algorithm W, the routing will be optimal assuming that the actual breakdown of the incoming traffic is the worst possible for the domain's local performance.

D. Computation of Equivalent Capacities

The purpose of equivalent capacities is to penalize usage of paths leading to congested areas in the network. Let g be a gateway link connected to gateway node i. Let C_g be the capacity of g. In our implementation, we compute the equivalent capacities C_g' from the form

$$\frac{C_g'}{(C_g' - f_g)^2} = \frac{C_g}{(C_g - f_g)^2} + \sum_j \alpha_{ij} d_{ij} \qquad (14)$$

as a heuristic function of estimated traffic in the domain. A coefficient α_{ij} in (14) is the fraction of the total internet traffic from gateway node i to any other node j. Due to lack of information, these coefficients have to be estimated by the corresponding NCC. The NCC calculates equivalent capacities of its incoming gateway links and sends the results to the INCC.

E. A Posteriori Routing

The initial routing is produced based on an estimate of the missing information about the states of the gateway links. However, the estimated parameters are not exact and therefore

the local performance of the produced routing is not optimal. The determined routing can be used for a period of time until steady state flows are established and each domain learns about the missing information. After the learning period, each domain can readjust its routing. This a posteriori routing can be done in two different ways:

1) Unchanged Use of the Gateway Links: Each domain keeps the previously determined use of the gateway links unchanged and readjusts the use of its internal paths. A new, locally optimal routing is computed after a domain learns about traffic coming from each of its gateway links. This approach requires only one measurement of the actual incoming requirements since the local changes will not affect other domains. Therefore, the stability of this routing algorithm is guaranteed.

2) Changed Use of the Gateway Links: In this version of the a posteriori routing, each domain is allowed to change its use of the gateway links by transit and outgoing traffic subject to the constraints imposed by the INCC in the last iteration of the routing. However, this adjustment can affect the rest of the network and therefore more than one iteration may be necessary. Note that there is no guarantee that this process converges. For that reason, we limited the number of iterations to 10. However, all examples that we tried, three to five iterations were necessary for convergence.

In the a posteriori phase of the routing algorithm, the INCC is not involved and the total flows in the gateway links remain unchanged. Therefore, all domains can measure and adjust their routing asynchronously.

IV. NUMERICAL EXAMPLES AND DISCUSSION

We have carried out a number of examples in order to evaluate the proposed heuristic algorithm. In this section we present some of them. All results were obtained analytically with all links modeled as independent $M/M/1$ queues. The performance of the algorithm is compared with the global optimal solution (e.g. obtained from the Flow Deviation for the global network) and a lower bound generalized from [5]. The lower bound is determined using global knowledge and additional constraints, that is: (i) each node routes messages destined to another domain as a single commodity regardless of the particular destination node, and (ii) a message never enters a particular domain more than once.

A. A Simple Example

We first assess the impact of the lack of information (internet traffic breakdown and global topology on our algorithm. Recall that we accounted for this information by assuming some estimated traffic breakdown and evaluating equivalent gateway capacities.

To understand what can be achieved with the amount of information available we use a simple example, Fig. 2. The gateway links are depicted with heavy lines. We will vary the estimation policy and the requirements and compare actual and estimated network performance. Let the network consist of two domains A and B as in Fig. 2. Domain A contains nodes 0 and 1, and domain B contains nodes 2 and 3. The
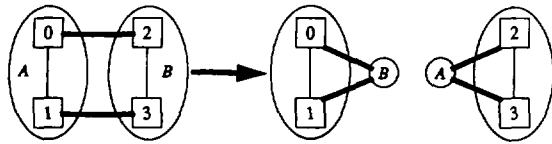
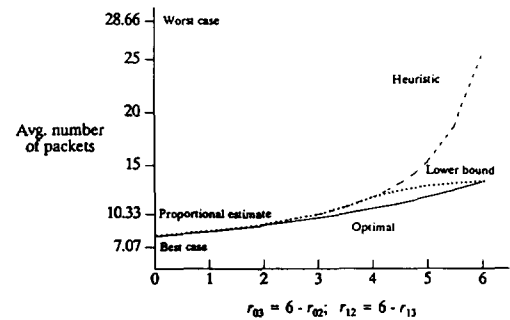Fig. 2. A simple two domain network and its condensed representation.

Fig. 3. Example 4.1: Average number of packets vs. requirement breakdown.

Fig. 4. Example 4.1: Maximum link utilization versus requirement breakdown.

capacity of each of 4 full duplex links is 10 units. Let the requirements be defined in such a way so the network is symmetric along the horizontal line. As a result the algorithm will always determine the same routing independently of the estimation policy requirements and equivalent capacities.

The right-hand side of Fig. 2 depicts a condensed representation of the network from the point of view of each NCC. Let us consider what is the result of the high level routing. Recall that in our heuristic algorithm each gateway link is characterized by a single number. Since the information available is symmetric along the horizontal line, the equivalent capacities of either gateway link in one direction must be equal. Therefore, flows in both gateway links in that direction must be equal and independent of how the equivalent capacities are computed.

Local routing will be independent of the estimation policy for the incoming traffic (best, worst, and proportional cases) since there is a single path between two nodes in each domain. We know that the low level routing is locally optimal. the internet requirements will be routed along the physically closest gateway link since each node has the same amount of outgoing traffic as its closest gateway link is allowed to carry. Therefore, the algorithm will determine the same routing for this particular example independent of the actual breakdown of the internet requirements and estimation policy. In what follows, we vary internet traffic and estimation policy and compare the actual and estimated network performance of the heuristic algorithm, and the global and constrained optimal routing.

The incompletely defined requirements and the constraints on them are derived from the available information :

$$
\mathbf{r} = \begin{bmatrix} 0 & 1 & r_{02} & r_{03} \\ 1 & 0 & r_{12} & r_{13} \\ 3 & 3 & 0 & 3 \\ 3 & 3 & 3 & 0 \end{bmatrix} \quad \begin{aligned} r_{02} + r_{03} &= 6 \\ r_{12} + r_{13} &= 6 \\ r_{02} + r_{12} &= 6 \\ r_{03} + r_{13} &= 6 \end{aligned} \quad (15)
$$

We will vary the unspecified elements (the outgoing traffic of domain A) in order to determine the sensitivity of different algorithms. The total internet requirements out of nodes 0 and 1 and into nodes 2 and 3 are fixed and equal to 6 units. The total throughput of the network is equal to 32 units. Since each node sends its outgoing requirements along the closest gateway link, the best case will be if the entire outgoing traffic is destined to the node on the other side of the gateway link.

In the proportional case the actual breakdown of the outgoing requirements is split evenly among the two destination nodes. The worst case is when the entire outgoing requirements from each node are destined to the node diagonally opposite. The average number of packets in the network for the three cases is 7.07, 10.33, and 28.66 respectively. Recall that in all three cases the routing will be the same. Fig. 3 shows
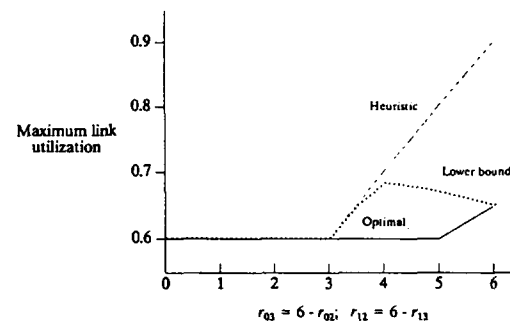
the average number of packets as a function of the variable requirements $r_{03} = r_{12}$ for the three different algorithms. Fig. 4 shows the maximum link utilization for the same parameters and routing algorithms. We see that, when $r_{03} \leq 3.5$ the lower bound on network performance can be achieved by our heuristic algorithm (and any algorithm that provides a single number per gateway link as its cost). The constrained optimal algorithm used for the lower bound [5], requires one gateway length for each source node-outgoing gateway link pair which in this example doubles the amount of information. The produced routing is nonbifurcated below a threshold which is approximately equal to 3.5. Above the threshold nodes 0 and 1 use the internal link 0–1 for the outgoing traffic and the constrained optimum algorithm performs better than the heuristic one. After r_{03} exceeds four units, the lower bound algorithm stops bifurcating and each node uses the opposite gateway link to send its entire outgoing traffic. As r_{03} increases above four units, a lower bound decreases since traffic breakdown becomes more suitable to already fixed routing. After r_{03} exceeds five units the network performance for the heuristic algorithm deteriorates significantly.

Note that nodes 0 and 1 begin to bifurcate their internet traffic after r_{03} exceeds 3.5 units. Above this threshold there exist flows of messages destined to domain B in either directions on link 0–1. If the messages are routed based on their destination domain exclusively, a loop will occur and cause problems in the network. The loop is a consequence of inconsistent perception of the rest of the network by nodes 0 and 1. A careful implementation should consider messages emanating in different nodes as different commodities. The messages should be routed based on their source node and
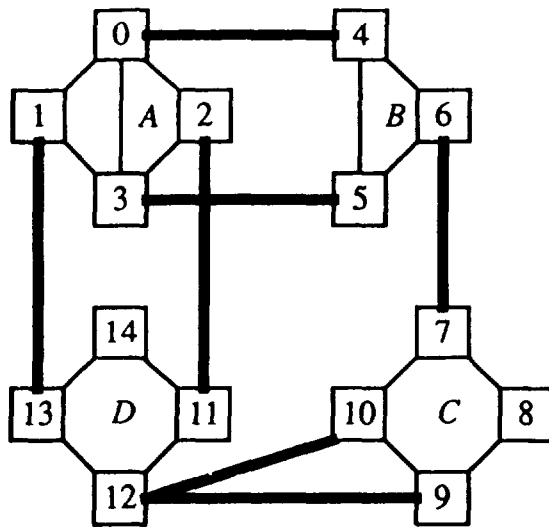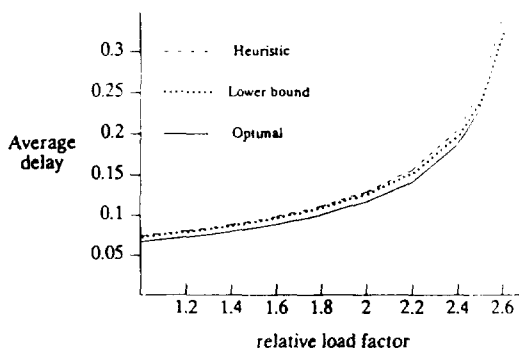
Fig. 5. A four-domain network topology.

Fig. 6. Example 4.2: Average delay versus relative load factor.
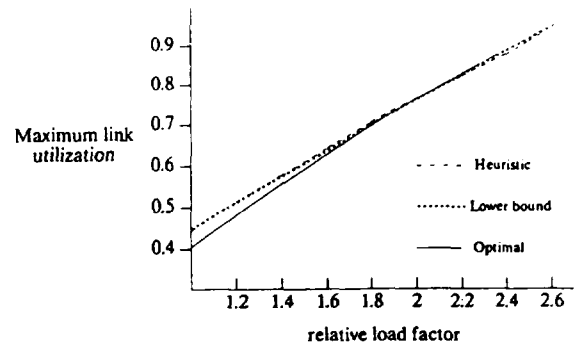

Fig. 7. Example 4.2: Maximum link utilization versus relative load factor.
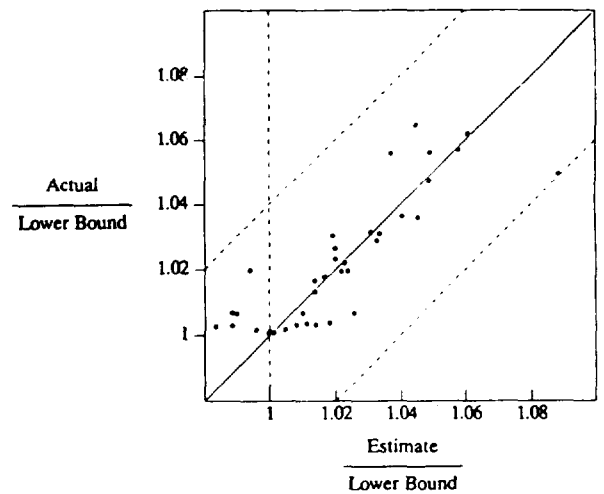
Fig. 8. Example 4.3: Correlation between estimated and actual network performance.

destination domain. However, the proposed heuristic algorithm guarantees that loop-free paths and the messages can be routed based on their destination domain exclusively. This simplifies the implementation and reduces the size of the routing tables as the expense of network performance.

B. A Gradually Loaded Network

Fig. 5 depicts the topology of a four-domain network. Capacities of all links are equal to 50 units. The initial end-to-end requirements are generated as exponentially distributed random numbers with the average value equal to 1. The network is gradually loaded by multiplying the initial requirements by the relative load factor ranging from 1 to 2.6. Fig. 6 shows that the average delay per packet in the network is close to the lower bound even when the maximum link utilization exceeds 90%. The maximum link utilization is also close for all three algorithms as shown in Fig. 7.

C. Statistical Performance of the Multidomain Routing Algorithms

In this example we study the performance of the proposed routing algorithm statistically for two different networks. One network is shown in Fig. 5. The other network topology can

be found in [3], Fig. 9 (Net6). We randomly generated several requirements profiles.

Fig. 8 shows the correlation between estimated and actual network performance after the initial routing terminates. Both values are normalized by the lower bound achievable. The points closer to the diagonal solid line denote a better estimate of the actual network performance. The points above this line give an optimistic estimate as opposed to pessimistic estimates below the line. Points on the left of the vertical dashed line are too optimistic and give an estimate lower than a lower bound. We see that all points lie within the band bounded by the diagonal dashed lines in which the difference between an estimated and an actual network performance is less than 4% of the lower bound. This means that we have a realistic picture of the network although we do not have complete information. We see also that all except two points have the vertical coordinate below 1.06. It means that in the majority of cases the initial, heuristic routing gives a good network performance which is less than 6% above the lower bound.

Fig. 9 statistically compares the heuristic (with a posteriori adjustments) and the optimal network performance with the lower bound. The horizontal axis gives the lower bound for average delay in the networks studied subject to the constraint that all messages destined to the same domain are routed as a single commodity. The vertical axis gives the
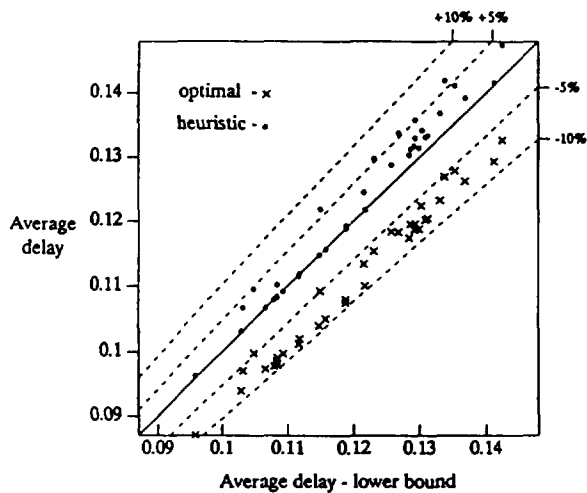
Fig. 9. Example 4.3: Heuristic (with *a posteriori* phase) and optimal routing versus lower bound.

optimal network delay (crosses) and the delay achieved by the heuristic algorithm (bullets). We see that the optimal network performance is 5–10% below the lower bound. In the majority of cases the heuristic algorithm (with *a posteriori* adjustments) gives network performance which is not more than 5% worse than the lower bound.

D. Comparison of Estimation Policies

There are two instances within our algorithm that an estimation policy is needed. The first concerns the evaluation of equivalent capacities of gateway links. The second involves the optimization of the local routing, once gateway flows are determined.

In this example we compare the local and global performance of each of the proposed estimation policies for the gateway flows, i.e., best case, proportional case, and worst case. We use the topology shown in Fig. 5. The end-to-end requirements are random numbers uniformly distributed between 0 and 4. The experiments showed that the proportional estimate generally gives better network performance since the best and worst case are not very likely to occur. For that reason we used the best, proportional, and worst case assumptions to determine flows in gateway links in the first phase of the heuristic routing algorithm. The local routing algorithm always applies the proportional estimate because we found that it gives better initial routing in most of the cases. After the routing is determined in such a way, the learning period starts. The domains measure the actual breakdown of the incoming traffic and adjust their local routing.

Tables I, II, and III show the network performance in each iteration of the initial and a posteriori routing algorithm for the three estimation policies. Column labeled as INCC shows the INCC's objective function i.e., "the average number of packets in the gateway links" when the equivalent capacities are used instead of the real capacities. Columns A-D shows the estimated (upper number) and the actual (lower number) average number of packet in queues for the internal links in each domain. Column labeled as "gateways" shows the actual

TABLE I
Best Case for the Network Topology in Fig. 5

	INCC	A	B	C	D	gateway	total
1	27.5644	3.16169	1.75832	4.14746	5.69480	27.5644	42.3267
		6.21117	3.40325	8.80252	11.2008		57.1822
2	30.0007	3.15284	1.81568	3.94671	5.42692	27.7122	42.0543
		6.31313	3.44869	8.50479	11.6842		57.6630
3	29.9756	3.13747	1.81806	3.66704	5.38233	27.7246	41.7295
		6.29486	3.46336	7.92027	11.9938		57.3969
1		6.33383	3.44451	7.06022	10.2089	27.7246	55.4552
2		6.25555	3.46152	7.42008	10.1761	27.7246	55.0829
3		6.30001	3.46305	7.42155	10.1761	27.7246	55.0847
4		6.30001	3.46318	7.42100	10.1759	27.7246	55.0847

TABLE II
Proportional Case for the Network Topology in Fig. 5

	INCC	A	B	C	D	gateway	total
1	27.5644	6.32043	3.37675	7.78775	10.7090	27.5644	55.7583
		6.17330	3.40014	8.20656	11.1504		56.4948
2	34.9651	6.37167	3.62951	7.37370	9.86683	27.8530	55.0948
		6.24655	3.67280	7.61420	10.2989		55.6855
3	35.0598	6.31845	3.86112	6.65135	9.52959	28.0678	54.4283
		6.24185	3.88459	6.07833	9.94699		55.1195
1		6.31845	3.86112	6.65135	9.52959	28.0678	55.1195
2		6.23389	3.88459	6.92739	9.61373	28.0678	54.7818
3		6.29230	3.88603	6.92748	9.61310	28.0678	54.7858
4		6.29235	3.88508	6.92756	9.61306	28.0678	54.7858

TABLE III
Worst Case for the Network Topology in Fig. 5

	INCC	A	B	C	D	gateway	total
1	27.5644	8.99462	5.41490	12.7383	21.5997	27.5644	76.3119
		6.22870	3.77115	8.99679	17.6812		64.2423
2	43.3148	9.20249	6.21072	12.2571	17.4210	28.1994	73.2907
		7.05506	4.40877	8.12391	18.6583		66.4455
3	43.1533	9.25908	6.86509	11.8632	16.3244	28.8530	73.1648
		7.05985	4.94674	7.10541	17.1800		65.1451
1		6.88327	4.40428	6.83204	8.87331	28.8530	56.5141
2		6.83263	4.43794	7.06488	8.97439	28.8530	56.2282
3		6.90508	4.43894	7.06511	9.97385	28.8530	56.2355
4		6.90502	4.43853	7.06478	8.97410	28.8530	56.2355

average number of packets in the gateway links. Recall that this number is controlled by the INCC. It is known since the high level routing has enough information to compute it. The column "total" shows the total estimated (higher) and actual (lower) average number of packets in the entire network. The high level routing stops when the relative difference between values of the INCC's objective function in two consecutive iterations is less than 1%.

All three examples have the same high level routing in the first iteration since they start with the same equivalent capacities which are equal to the actual capacities. Each domain produces the locally optimal routing subject to different assumptions about their incoming traffic. For example the optimal routing for domain A in the first iteration gives 3.16169, 6.32043, and 8.99462 packets on average in the internal links for the best, proportional, and worst case respectively. If the learning period starts after the first iteration, the actual average number of packets in the internal links would be 6.21117, 6.17330, and 6.22870 respectively. After the first iteration, the equivalent capacities differ and therefore the produced high level routing is different. After 3 iterations the algorithms terminate. The low level routing is adjusted according to the proportional estimate and the learning period starts. Rows in

Tables I, II, and III below the heavy line show iterations of the a posteriori routing. Note that the average number of packets in the gateway links remains unchanged since the INCC is not involved in the *a posteriori* routing. In most of the cases, 3 to 5 iterations are needed in this phase of the algorithm although there is no proof that the procedure converges when changed use of the gateway links is allowed. The average number of packets in the network is 48.9746 for the globally optimal routing and 52.8198 for the lower bound. We see that the proportional estimate produces a better routing (54.7858) than the best case (55.0847) and the worst case estimate (56.2355)). It was the case in all examples studied except some fabricated on purpose.

E. A Posteriori Routing

In this example we discuss *a posteriori* routing. We define the requirements in the network shown in Fig. 5 so the proportional estimate of the breakdown of incoming internet traffic is far from the actual in one of the domains. The produced initial routing is far from the optimal, but the network is able to improve its performance after it learns the actual breakdown of the incoming traffic.

Recall that the capacities of all links are equal to 50 units. Let the internal requirements out of node $0(r_{01}, r_{02},$ and $r_{03})$ be equal to 25 units. Let the internal requirements out of node $7(r_{7,10},$ and $r_{78})$ be equal to 20 units. This makes gateway links $4 \rightarrow 0$ and $6 \rightarrow 7$ unattractive. Let the requirement r_{43} be equal to 22 units and all other requirements be equal to 2 units.

We know that the low level routing is locally optimal subject to the assumption that the breakdown of the incoming traffic is correct. Therefore, most of the flow in the gateway link $4 \rightarrow 0$ determined by the high level routing which is destined to domain A should be used by node 4. Since most of its traffic is destined to node 3, there should be a great difference between estimated and actual traffic coming from gateway link $4 \rightarrow 0$ and destined to node 3. Due to the wrong estimation, the produced routing is far from the optimal.

Table IV shows the results of the heuristic algorithm in each iteration. Due to the chosen requirements, domain A underestimates the average number of packets in its internal links. The estimate of the other three domains is exact since their internet requirements are uniform. Due to a large error in estimation, the heuristic routing gives 85.8765 packets in the network on the average which is far from the lower bound 74.3062 and the optimal value 66.9001. The utilization of the most heavily utilized link $0 \rightarrow 3$ is 92.9%. Utilizations of links $0 \rightarrow 1$ and $0 \rightarrow 2$ are about 75%.

Rows in Table IV below the heavy line show iterations of the a posteriori routing. After the learning period, domain A learned the actual breakdown of traffic in its gateway links and was able to improve its local routing, while the routing in the other three domains remained unchanged. Only two iterations were needed. The typical number of iterations is 3 to 5 although there is no proof that the a posteriori routing converges at all. The average number of packets in the network is reduced to 79.2409 which is less than 7% above the lower

TABLE IV
EXAMPLE 4.5: BEHAVIOR OF THE *a posteriori* ROUTING

	INCC	A	B	C	D	gateway	total
1	30.8222	16.9397	3.49689	14.7537	12.1295	30.8222	78.1420
		29.4838	3.49689	14.7537	12.1295		90.6861
2	42.1395	17.9922	3.78546	13.1760	12.6346	31.2398	78.8281
		28.8869	3.78547	13.1760	12.6346	89.7228	
3	43.0218	17.6943	3.80675	13.3239	12.1794	31.5059	78.5103
		25.3804	3.80675	13.3239	12.1794		86.1963
4	43.2684	17.8230	3.80230	12.7538	12.2492	31.6558	78.2858
		25.4137	3.80399	12.7539	12.2492		85.8765
1		18.7780	3.80399	12.7538	12.2492	31.6558	79.2409
2		18.7780	3.80399	12.7538	12.2492	31.6558	79.2409

bound. The most heavily utilized link is still link $0 \rightarrow 3$, but its utilization is reduced to 81.7%. Utilizations of links $0 \rightarrow 1$ and $0 \rightarrow 2$ increased to about 81%. This means that node 0 was forced to reroute the traffic along the alternative paths in order to clear a direct path toward node 3 for the traffic destined to that node. The average number of packets in the gateway links is unchanged during the a posteriori routing since the high level routing remains fixed in this phase.

V. CONCLUSION

A general approach to designing a hierarchical algorithm for routing in multidomain networks is presented and a heuristic routing algorithm is developed in this paper. A simplified model of a packet-switched network is used. The overall network performance is managed through the high-level routing that is performed by an Integrated Network Control Center. The computational complexity of the proposed algorithm grows as $O(\sqrt{n^3})$ for an appropriately partitioned network and the size of the routing tables is $O(\sqrt{n})$. A number of examples is presented to illustrate the proposed heuristic algorithm. Generally, the algorithm provides network performance that is no more than 10% above the lower bound although some special examples can be made up where the algorithm performs much worse. In many cases, the network performance was less than 5% above the lower bound.

REFERENCES

[1] A. E. Baratz and J. Jaffe, "Establishing vitrual circuits in large computer networks," IBM Res. Rep. RC10350(46174), Jan. 1984.
[2] D. Bertsekas and R. Gallager, *Data Networks*. Englewood Cliffs, NJ: Prentice Hall, 1987.
[3] C. Chao, P. Sarachik, B. Maglaris, R. Boorstyn, and D. Dimitrijević, "Control of multi-domain networks," *Proc. Network Manag. and Contr. Workshop*, Tarrytown, NY:, Sept. 1989.
[4] *DECnet, Digital Network Architecture, Routing Layer Functional Specification*, Version 2.0.0, Digital Equipment Corporation, Maynard, MA, May, 1983.
[5] D. Dimitrijević, B. Maglaris, and R. Boorstyn, "Routing in multiple-domain networks," *Proc. of IEEE INFOCOM' 89*, Ottawa, Canada, pp. 1047–1056, Apr. 1989.
[6] D. Dimitrijević, "Routing in multi-domain networks," Ph.D. dissertation, Dep. of EE/CS, Polytechnic Univ., 1990.
[7] D. Dimitrijević, B. Maglaris, and R. Boorstyn, "Routing in multi-domain networks," *Proc. IEEE INFOCOM' 91*, Bal Harbour, FL, April 1991, pp. 257–264.
[8] L. Fratta, M. Gerla, and L. Kleinrock, "The flow deviation method: An approach to store-and-forward communication network design," *Networks*, vol. 3, no. 2, pp. 97–133, 1973.
[9] K. T. Fung and D. P. Ko, "X.25 PAD concentrator and gateway network management towards a UNMA environment," *Proc. Network Manag. and Contr. Workshop*, Tarrytown, NY: Sept. 1989.

[10] S. I. Gass, *Linear Programming*. New York: McGraw-Hill, 1984.
[11] J. Hagouel, "Issues in routing for large and dynamic networks," IBM Res. Rep. RC9942(44055), Apr. 1983.
[12] _____, "Issues in routing for large and dynamic networks," Ph.D. dissertation, Columbia Univ., May 1983.
[13] M. C. Hamner and G. R. Samsen, "Source routing bridge implementation," *IEEE Network*, vol. 2, Jan. 1988.
[14] F. Kamoun, "Design considerations for large computer communication networks," UCLA-ENG-7642, 1976.
[15] L. Kleinrock and F. Kamoun, "Hierarchical routing for large networks—Performance evaluation and optimization," in *Computer Networks*. North-Holland, The Netherlands: 1977, vol. 1, pp. 155–174.
[16] D. Luenberger, *Linear and Nonlinear Programming*. Reading, MA: Addison-Wesley, 1984.
[17] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*. New York: McGraw-Hill, 1984.

Dragomir D. Dimitrijević (S'87-M'88) was born in Belgrade, Serbia, Yugoslavia, in 1963. He received the B.S. and M.S. degrees from the Department of Electrical Engineering, University of Belgrade, Serbia, Yugoslavia, in 1986 and 1987, respectively, and the Ph.D. degree in computer science from Polytechnic University, Brooklyn, NY, in 1990.

He joined Contel Technology Center, Chantilly, VA, in 1990 as a Member of the Technical Staff. After the merger in between Contel and GTE, he joined GTE Laboratories, Waltham, MA, in 1991. Currently, as a Senior Member of the Technical Staff, he is involved with development of performance analysis techniques and tools for computer and communication systems. His areas of interest include routing in multidomain networks, protocol verification and evaluation, modeling and performance evaluation of communication and computer systems, channel allocation in cellular networks, and traffic engenering and planning of SS7 networks.

Dr. Dimitrijevic has received several awards such as GTE Laboratories Performance Recognition Award granted in 1993, Pearl Brownstein Doctoral Research Award for his doctoral research granted in 1990 by Polytechnic University; Special Research Fellowship granted in 1988 by the Foldes Fellowship Foundation at Polytechnic University; The Best Paper Award in the area of digital communications at the 13th ETAN Conference in 1987; Research Fellowship granted in 1986 by the Serbian Academy of Sciences and Arts; International Prize at Competition in Mathematics for Students of Balkan Countries granted in 1984 by the Union of Balkan Mathematicians. Dr. Dimitrijevic is a member of the IEEE Communications and Computer Societies.

Basil S. Maglaris (S'74-M'79) received the undergraduate diploma in Electrical Engineering from the National Technical University of Athens in 1974, the M.Sc. degree from the Polytechnic Institute of Brooklyn in 1975, and the Ph.D. degree in Electrical Engineering from Columbia University in 1979.

From 1979 to 1981, he was with the Network Analysis Corp., Great Neck, NY. In 1981 he joined the Polytechnic Institute of New York, Brooklyn, NY where he is Associate Professor of Electrical Engineering and Computer Science. Since 1990, he is on an extended leave. Currently, he teaches at the National Technical University of Athens, and conducts research in the Network Management and Optimal Design Laboratory. His research interests focus on the analysis, performance evaluation, network management, and optimization of data, voice and integrated networks. He teaches courses on network protocols, performance analysis and design.

Robert R. Boorstyn (S'58-M'59-SM'82-F'86) received the B.E.E. degree from the City College of New York in 1958, and the M.S. (EE) and Ph.D. (EE) degrees, both from the Polytechnic Institute of Brooklyn, in 1963 and 1966, respectively.

He is currently professor of Electrical Engineering and Computer Science at the Polytechnic University in New York. He has been a member of the faculty of the Polytechnic since 1961. He spent a year's leave at Bell Telephone Laboratories in 1977-1978 performing research on dynamic routing and other topics in communication networks. He spent the summer of 1987, and subsequently consulted for Bell Communication Research, where he worked on the architecture of multicast broadband switches. For the calendar year 1993 he was a visiting scientist at IBM T. J. Watson Research Center, performing research on performance and design of high speed networks. In 1994 he was a visiting scholar at the Univeristy of California in Berkeley. Professor Boorstyn has written over 50 journal and conference publications—mostly in the area of communication networks. His main research has been in network design, algorithms, and performance, dynamic routing, multihop packet radio networks, network management, and broadband networks.

Dr. Boorstyn has been a frequent consultant, and has been the principal investigator on numerous multi-year grants and contracts, for both government agencies and industry. Prof. Boorstyn has supervised over 30 Ph.D. dissertations. He is one of the founders of Polytechnic's Center for Advanced Technology in Telecommunications (CATT), which was created by New York State in 1983, and where he has been director of research. In 1986 he was elected Fellow of the IEEE for contributions to the theory and development of packet radio networks.