Virtual Path Control for ATM Networks with Call Level Quality of Service Guarantees

Nikolaos Anerousis, *Member, IEEE*, and Aurel A. Lazar, *Fellow, IEEE*

***Abstract*— The configuration of virtual path (VP) connection services is expected to play an important role in the operation of large-scale asynchronous transfer mode (ATM) networks. A major research challenge is to understand the fundamental tradeoff between the network call throughput and the processing load on the signaling system and to provide an algorithm for VP capacity allocation that achieves an optimal network operating point while guaranteeing quality of service (QoS) at the call level and satisfies *a priori* bounds on the processing load of the call processors. We present a taxonomy of previous approaches to the problem and identify their strengths and weaknesses. Based on these observations, we provide an algorithm for the VP capacity allocation problem that satisfies nodal constraints on the call processing load and blocking constraints for each source–destination (SD) pair. The algorithm maximizes the network revenue under the above set of constraints and is parameterized by the number of traffic classes in the network, the method of representation of networking resources, the admission control policy used in every link and VP, and the network routing scheme. Finally, we apply the algorithm to three sample networks and study several of its performance characteristics. In one case, we applied the calculated VP distribution to the Xunet ATM testbed and verified experimentally the predicted performance.**

I. Introduction

In our previous work [1] we presented a comprehensive management architecture that allows the network manager to configure virtual path (VP) connection services under quality of service (QoS) constraints and evaluate the resulting network performance. VP's in asynchronous transfer mode (ATM) networks provide substantial speedup during the connection establishment phase at the expense of bandwidth loss due to the end-to-end reservation of network resources. Thus, VP's can be used to tune the fundamental tradeoff between the network call throughput and the processing load on the signaling system. They can also be used to provide dedicated connection services to customers such as virtual networks (VN's). The VP connection management architecture that was developed for this purpose was integrated within an OSI management architecture.

The objective of this paper is to provide a supporting algorithmic framework to the network manager for formulating a capacity allocation policy for VP's (also referred to as the *VP distribution policy*), that guarantees QoS both at the cell and the call level. QoS at the cell level can be guaranteed through the concept of the contract region (CR) [14]. At the call level, QoS is guaranteed by bounding the blocking probability of the virtual circuit (VC) service for every source–destination (SD) pair in the network and the average connection setup time. The latter is achieved by modeling the signaling system as a network of queues and bounding the call arrival rates at the signaling processors.

Manuscript received July 10, 1996; revised April 4, 1997; approved by IEEE/ACM Transactions on Networking Editor I. Cidon. This work was supported in part by the National Science Foundation under Grant CDA-90-24735, and in part by a Grant from the AT&T Foundation.

N. Anerousis is with AT&T Laboratories—Research, Florham Park, NJ 07932-0971 USA (e-mail: nikos@research.att.com).

A. A. Lazar is with the Department of Electrical Engineering and Center for Telecommunications Research, Columbia University, New York, NY 10027-6699 USA (e-mail: aurel@ctr.columbia.edu).

Publisher Item Identifier S 1063-6692(98)02848-9.

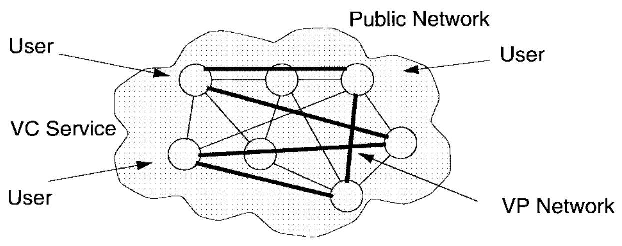

Fig. 1. The network model.

According to our model, users make requests for the VC service at the boundaries of the network (Fig. 1). The network operator establishes VP's within its administrative domain. This VP network is transparent to the users and serves the purpose of carrying user calls through the public network at reduced call processing costs. A separate routing mechanism determines how calls are to be routed through the VP network.

The VP distribution problem is a network control problem with the following formulation: given the network topology, the capacities of network links, the capacities of the signaling processors, and the matrix of offered network load, calculate the routes and capacities of VP's in the network such that the following requirements are satisfied:

1) the sum of VP capacities on each link does not exceed its capacity;
2) the signaling load on each signaling processor is below a given upper bound;
3) the call blocking rate of each SD pair is below a given upper bound (also referred to as the *SD blocking constraint*);
4) the network revenue is maximized under the above constraints.

Fig. 2 shows how the VP distribution algorithm can be embedded in a management cycle. First, information regarding the offered load and the QoS observed by the users is retrieved from the network. If QoS is satisfied, and there is no projected

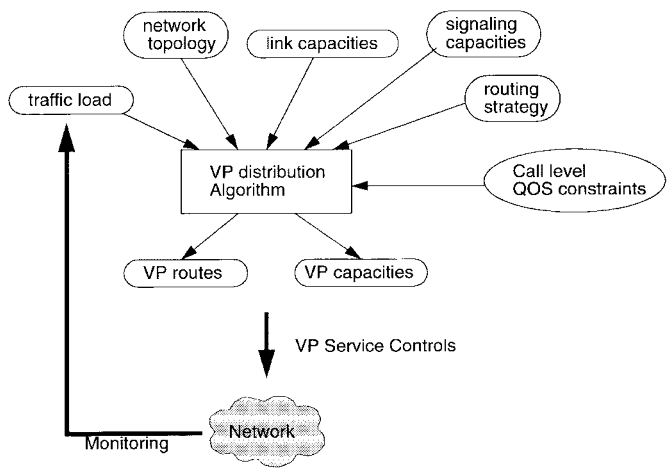

Fig. 2. The algorithm for VP distribution.

change in the offered load, the manager does not need to take any further action. If the load has changed, or QoS is not guaranteed, the manager can use the algorithm to provide a new VP distribution that will satisfy the QoS requirements. The algorithm is given the current estimates of the traffic load, the desired QoS constraints, and other configuration information, and produces the VP distribution. The latter is installed in the network by using the control mechanisms provided by the service management architecture.

The VP distribution problem has appeared in many disguises, especially in the design of circuit-switched networks and, more generally, in the configuration of logical networks given a fixed topology of physical links.

The requirement described in constraint 2) is justified by our experiments with the Xunet III ATM testbed. According to these experiments, the transport network can be saturated even with small call arrival rates of wide-band (video) calls. In order to achieve the same effect with narrow-band (e.g., voice) calls, much higher call arrival rates are needed (the capacity of one video call in our experiments was roughly equivalent to 70 voice calls). In this case, however, the capacity of the signaling system was reached *before* the actual transport capacity was exhausted. During signaling congestion, only a small percentage of the total number of offered calls could be established (the remaining calls were rejected due to signaling message losses, excessive call setup times, etc.) and, as a result, the transport network was operating well below its capacity. In other words, even if the total capacity demand is the same, a small call arrival rate with a high capacity demand per call puts pressure on the transport network, whereas for greater call arrival rates with small capacity demands the pressure is shifted to the signaling system.

These experiments further indicated that an uncontrolled overload of the signaling system can render inoperable most of the backbone switches that receive the highest signaling load and reduce the throughput of the transport network dramatically. In general, an ATM network supporting a switched virtual circuit (SVC) connection service can overcome this problem in two ways: by blocking a portion of the incoming call setup requests at the source node, thereby preventing congestion at the downstream nodes, or by setting up VP's between the SD pairs that contribute the bulk of the signaling load on the intermediate switches. If the first approach is followed, calls might be blocked even if there is enough capacity available in the network. Therefore, the second approach is superior but at the expense of a reduced network throughput due to end-to-end bandwidth reservation.

The proposed algorithmic framework is part of a network management architecture that applies network controls from a centralized location in time scales of the order of seconds, or even minutes. Therefore, the algorithm for VP distribution that will be discussed here is not applicable for real-time network control but rather for short- to mid-term network capacity and throughput management. It is a centralized nonlinear programming algorithm that uses as input the global state of the network and provides a solution that maximizes the total network revenue while satisfying the blocking and signaling capacity constraints. In this context it is conceivable that the algorithm is most appropriate for use by an ATM network service provider. The algorithm is run when the manager observes significant changes in the traffic patterns. In current telephone networks this period ranges from one half-hour to one hour, and is expected to be similar for broadband networks. The resulting VP distribution is installed in the network and held constant until the next run.

Rather than trying to maximize the overall network throughput, the VP distribution problem can also be considered from the viewpoint of noncooperative networks, where each VP controller is trying to (selfishly) maximize its own performance by requesting bandwidth for its end-to-end VP's. This leads to a problem that can be formalized as a noncooperative game and was explored in [20].

This paper is organized as follows. Section II presents an overview of related work on VP distribution algorithms. Section III presents an algorithm for VP distribution together with the concepts that provide our QoS framework. Section IV applies the algorithm to three network topologies and makes observations on its performance characteristics. Finally, Section V summarizes our conclusions and proposes directions for further study.

II. Statement of the Problem and Review of Related Work

A. Notation

Before starting the review of the individual approaches to the VP distribution problem it is worthwhile to define a common problem setting based on which all approaches can be compared. For this purpose, we introduce the following canonical model.

1) Topology Information:

- The network topology is represented as a directional graph $\mathbf{G}(V, E)$, where V is the set of vertices (network nodes) and E is the set of network links (edges).
- $\mathbf{W}$ is the set of SD pairs $\mathbf{W} = \{w = (u, v) \mid\mid u, v \in V\}$.
- The network supports $k = 1, 2, \cdots, K$ traffic classes, each with its own QoS requirements.

- $\mathbf{P}$ is the set of VP's.
- For each SD pair $w, R_w = \{r_{wi}, i = 1, 2, \cdots, N_w\}$ is a set of routes of cardinality N_w. Each route consists of an arbitrary combination of VP's and links between the source and the destination node.
- We define a logical graph $G'(V, E')$, where the set of edges E' is obtained from the set of edges of the original graph G by adding one edge between every VP source and termination point. In this way, VP's can be represented as logical network links.

2) Capacity Information:

- The networking capacity for link l is denoted by S_l and is described by the schedulable region (SR) [12].

3) Loading Information:

- The call arrival rate and the inverse of the holding time of calls for each SD pair w and class k is denoted by λ_w^k and μ_w^k, respectively. The traffic intensity (in Erlangs) is denoted by ρ_w^k.
- For each switching node $v \in V$, μ_v denotes the processing capacity of the node signaling processor in requests per unit of time. Finally, λ_v denotes the total arrival rate of call setup requests at node v.

4) Control Policy:

- For each p in $\mathbf{P}$, we denote as C_p the networking capacity assigned to p. The networking capacity is in general given by the CR [14]. For every link l we must have that the sum of VP capacities traveling over that link is less or equal to the link capacity.
- At every outgoing link and VP operates an admission controller. Encoded in the admission controller is the admission control policy, denoted by $\mathbf{u}$. The admission controller accepts or rejects calls based on their traffic class and the number of currently active calls in the link or VP.
- For each set R_w with $N_w > 1$ there exists a routing policy $\Re_w$ which specifies the schedule for finding available routes. For example, an incoming call might try the first route in the set and, if no capacity is available, the second route and so on.
- In order to guarantee that every signaling processor is operating normally, an overload control rejects the surplus of call requests whenever $\lambda_v > \mu_v$.

5) Constraints:

- The blocking constraint for each SD pair and traffic class is denoted by β_w^k. The blocking constraint enforces QoS at the call level. The blocking probability P_w^k is the percentage of call attempts of class k for the above SD pair that are denied service due to the unavailability of resources. We must always have that $P_w^k \leq \beta_w^k$.
- At every signaling processor, the arrival rate of call setup requests must be less than the node's processing capacity, i.e., $\lambda_v < \mu_v$.
- For every route $r \in R_w$, the call setup time t_r on route r must be bounded, i.e., $t_r < T_r$.

For example, consider the network of Fig. 3. We have two SD pairs (A,E) and (B,F). There are two VP's configured in the network: VP1 uses the links 1, 3, and 4 and VP2 uses the links 2, 3, and 5. Each VP can be also considered as a logical link directly connecting nodes A and E and B and F, and is represented by the dashed links 6 and 7, respectively. For each SD pair, there exist two routes: the first route (direct) consists only of the VP that links the source and the destination node and the second route consists of the single hop-by-hop route between the nodes (which happens to follow the same path as the VP). The routing policy attempts first to establish a call on the VP route, and if no capacity is available, the second route is tried. If the second attempt is also unsuccessful, the call is blocked.

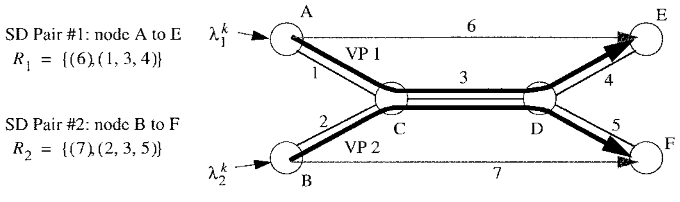

Fig. 3. An example.

Our objective is to compute the capacities of the VP's such that the revenue of the network is maximized under the appropriate constraints. One way to compute the revenue is by multiplying the calls of each traffic class k with a constant representing the network gain achieved by accepting a call of class k per unit of time. The VP capacities must be such that the following constraints are satisfied: the sum of VP capacities must not be greater than the capacity of link 3, the capacity of VP1 must be less or equal to the capacity of links 1 and 4, etc., the blocking probability for each SD pair must be less than the blocking constraint, and the arrival rate of signaling messages at every node must be less than the corresponding processing capacity.

The reader might have already noted the following tradeoff: by increasing the capacity of VP1 we can reduce the blocking probability for SD pair 1 but, at the same time, increase it for SD pair 2. It can be easily verified for this particular example that by not allocating any capacity to the VP's, the network throughput (and consequently the revenue) is maximized. However, in that case, node C experiences the combined arrival rate of call setup messages from both SD pairs, since all call setups take place only on a hop-by-hop basis. If this arrival rate is greater than the processing capacity of that node, the solution is unacceptable, and, therefore, part of the traffic must be routed over the VP's. Therefore, the solution lies in routing only part of the traffic over VP's to prevent congestion of the signaling system and maintain at the same time an acceptable blocking probability for all SD pairs.

The VP capacity allocation problem in ATM networks can also be regarded in a more general context: given a network topology, a portion of the capacity of the physical links is reserved on an end-to-end fashion for creating logical links connecting nonneighboring nodes. This problem rises in circuit-switched networks, in the configuration of leased lines and virtual trunks (VT's), and also in the configuration of logical channels in SONET-based networks [8]. For this reason, we will review a variety of approaches for capacity allocation that are similar in nature to the VP distribution problem.

B. Taxonomy of Algorithms

1) Synchronous versus Asynchronous: All algorithms fall into two major categories: synchronous and asynchronous. Synchronous algorithms update the VP capacity in real time based on the observed demand for call establishment. In this context the bandwidth of VP's can be expanded or contracted at call arrival or departure times. Such is the scheme presented in [24] and [22].

On the other hand, asynchronous algorithms maintain a fixed VP distribution for a time period T (also referred to as the *update interval*). These algorithms are called asynchronous, because the modifications of VP capacity are not driven by the call arrival or departure process associated with each VP. The VP distribution policy is computed at the beginning of the period and remains fixed during that period. The decision on the most appropriate VP distribution policy is based on estimates of the offered load during the coming period. For this purpose, a load estimator is usually needed to *predict* the offered load.

2) Centralized versus Decentralized: Asynchronous algorithms can be further distinguished in two major categories: centralized and decentralized. Centralized algorithms are executed in one location (typically at the network manager's site) and require the collection of up-to-date information from all network nodes, while decentralized algorithms are executed in every network switch and information is passed along between neighboring nodes.

3) Form of the Cost Function: Algorithms can be also categorized based on the cost (objective) function employed in the decision-making process for VP capacity assignment. The scheme of [24] and [22] does not use a cost function. For selecting the VP that will receive a bandwidth increase, [25] uses a cost function based on the VP blocking probability. In its other variation a linear combination of the carried traffic in every VP is employed. In [21] the total call blocking rate is used. Multiple cost functions have been selected in [8]: the link residual capacity and the amount of bandwidth demand that needs to be blocked overall among SD pairs. A weighted sum of the total rejected bandwidth demand and the total load assigned to each link are defined in [7]. A linear combination of the reserved bandwidth on each link, weighted by a cost factor, is employed in [11]. In [18] a function of the reserved capacity on every link in combination with a switching cost for every VP is also used. The AT&T capacity design system [5] uses a weighted combination of the residual link capacities. In [10] the revenue for each iteration of the algorithm, expressed as the difference of the expected blocking probabilities before and after increasing the capacity of a path times the expected load on the path, is used. Essentially, the same cost function is used by [3], slightly modified to incorporate the cost observed by other SD pairs when the capacity of every VP is increased, while the capacity of all others is held constant.

Since the objective of the VP distribution problem is to achieve a satisfactory network throughput, it is logical to include the call blocking rates in the cost function. This approach is taken by [25], [21], [10], and [3]. The total rejected bandwidth demand in the cost function is incorporated in [8] and [7]. The solution in this case maximizes the total carried capacity in the network. This has some advantages because the call blocking rates (which are nonlinear functions of the VP capacities) do not participate in the cost function, and the optimization problem becomes more tractable. However, in either case, in order to guarantee QoS at the call level the call blocking constraints must be introduced as constraints into the optimization problem. Only [3] addresses this problem. In the other cases, even if the solution maximizes the network throughput (or the carried capacity), the call blocking rates might be acceptable only for some SD pairs and violating the blocking constraints for others.

4) Tradeoff Between Signaling Costs and Capacity Allocation: Most algorithms except [5] and [18] assume that one or more VP's are established between every SD pair and, thus, the destination can be reached from the source in one hop only. In our opinion this approach has two major flaws. First, it is not scalable: a network with hundreds of nodes will need a very large number of VP's and, consequently, the VP distribution task will be overwhelming. Second, there is a substantial cost associated with loss of throughput due to the rigid capacity reservation for each VP.

A VP capacity allocation algorithm should not distribute all network capacity between VP's, but rather only a portion of it. This can be achieved by using a hybrid routing scheme, where calls first attempt to be established over a route using a direct VP, and, if unsuccessful, over routes using one or more intermediate nodes and VP's or physical links in between. Such a scheme can maintain a natural balance between the processing costs and the cost due to the reservation of resources. A lesson can be learned here from the design of the AT&T real-time network routing system [5]. This system achieves very low call blocking probabilities because it allows, in addition to a direct route, a large number of nondirect alternate routes to be followed (the equivalent of a call setup using two VP's and an intermediate node in an ATM environment) and it has been shown to absorb well traffic load fluctuations. This would be very difficult to achieve by using a single direct route for every SD pair.

From all of the capacity allocation algorithms reviewed, only [22] investigates the tradeoff between capacity utilization and signaling costs. The context is slightly different because there is only one route available to each destination (using a VP), and the signaling cost is associated with the frequency of messages needed to expand or contract the capacity of the VP. These messages must travel along the route of the VP in the same way as the hop-by-hop call setup messages that would be used to establish a VC on the same route. Thus, this work models the cost for allocating capacity to VP's and we believe that this must be fully taken into account in a flexible capacity allocation algorithm.

C. Discussion

Table I presents a comparison of the capabilities of several VP capacity allocation algorithms. The last column corre-

TABLE I
COMPARISON OF CAPACITY ALLOCATION ALGORITHMS

Capability	[22]	[25]	[21]	[8]	[7]	[11]	[10]	[3]	[5]	ours
Sync/Async	Sync	Async	Async	Async	Async	Async	Async	Async	Async	Async
Cetr./De-centralized	Decent	Decent	Centr	Centr	Centr	Centr	Centr	Centr	Centr	Centr
Route Set per SD pair	One VP	One VP	One VP or Link-only	Many Paths (VPs)	Many VPs	Many VPs	Many Paths (VPs)	One VP	1 and 2-hop paths	Mixed VP/Link paths
Route Selection Policy[a]	Static	Static	Static	Static	Static	Static	Static	Optimal Static	Dynamic	Static or Dynamic
# tr. classes	one	one	one	N/A	any	one	N/A	any	any	any
Admission Control Policy	per TC[b]	per TC	per TC	N/A	per TC	per TC	N/A	per TC	N/A	Flexible
Signaling Constraints	no	no	no	no	no	no	no	no	no	yes
per SD Pair Blocking Constraints	no	no	no	no[c]	no[c]	no	no	yes	yes	yes
Cost Function	None	Call Blocking Rate	Call Blocking Rate	Residual cap., Rej. BW demand	Link load, Rej. BW demand	Resrvd Bandwidth	Revenue	Revenue/ Loss	Residual capacity	Revenue

[a] Static implies that the routing policy is predetermined and given as input to the algorithm. Optimal applies to [3] only and implies that the single VP route is determined by the algorithm based on some optimization criterion. Dynamic implies that routing is determined in real-time from the available set of routes.

[b] Implies that a separate VP carries every traffic class and, hence, capacity cannot be shared between traffic classes.

[c] [7] and [8] minimize the maximum percentage of rejected capacity among all SD pairs.

sponds to the VP allocation scheme that will be presented in Section III. All algorithms fall into two major categories, based on the time scale they operate on, the information they use, and the level of optimality they provide. Scalable algorithms that are designed to operate on a fast time scale must be decentralized. They provide, however, a solution which might not be the global optimum. On the other hand, centralized algorithms run on a much slower time scale and can provide a solution closer to the optimum. However, they have limitations with regard to scalability.

Even in its simplest setting the VP distribution problem contains nonlinearities in the objective function, and is very difficult to solve analytically. It is a nonlinear programming problem whose objective function is neither convex nor concave. Standard nonlinear programming techniques can be used, however, to provide an efficient solution. The solution is not always the global optimum and becomes even more complex to determine if optimization is done for the joint problem of VP routing and VP capacity allocation. As a result, most algorithms prefer to fix the routing part. Others provide a fixed set of VP's but with more than one route from source to destination, while others [3] provide multiple routing paths by running a shortest path algorithm jointly with the VP capacity allocation algorithm. A simple methodology for routing VP's with known capacities was given in [6] but is not useful in our case because the VP capacities are not known in advance.

III. THE VP DISTRIBUTION PROBLEM

By observing the limitations of previous approaches to the VP distribution problem, we propose an algorithm that satisfies the following basic requirements:

- supports any number of traffic classes;
- explicitly guarantees QoS at the call level by introducing hard bounds for the call blocking rates for each SD pair and bounds for the time to establish calls through the signaling system;
- works with any combination of admission control and routing scheme (i.e., static priority or adaptive routing).

In addition, the algorithm has the following desirable properties:

- is independent from the abstraction used to describe the networking capacity of links and VP's;
- is independent from the admission control policy used at every link and VP.

The algorithm tries to maximize the network *revenue* per unit of time (defined as the weighted sum of the throughputs for each traffic class and SD pair multiplied with the call holding time) while satisfying the node processing and SD blocking constraints. The algorithm is presented in Section III-A. Sections III-B and III-C provide a methodology for computing the quantities needed by the algorithm for a simple routing scheme with prioritized alternate routes. For every SD pair, every route consists of one or more logical links, where each logical link can be either a physical link or a VP.

A. An Algorithm for VP Distribution

Initialization: The algorithm begins with all VP's at zero capacity. Traffic between all SD pairs follows a hop-by-hop call setup procedure, except for the SD pairs which are served only by routes comprised of VP's (in which case all traffic is blocked). Compute blocking probabilities for all SD pairs for the given call arrival rates.

Step 1: Find the SD pairs for which the blocking constraints are not satisfied. If none are found, proceed to Step 2. Else, consider every VP whose capacity can be increased as a bandwidth increase candidate (BIC). The capacity of a VP is increased by removing the necessary resources from all the links on the path of the VP. If the BIC set is empty (i.e., there is no spare capacity available in the network) then exit. Else, compute the *blocking drift* D_b for the current vector of VP capacities (also referred to as the current solution) from

$$D_b = \sum_{w,k} \max\bigl(0, P_w^k - \beta_w^k\bigr), \qquad w \in \mathbf{W}. \tag{1}$$

This quantity represents "how far" we are from satisfying the blocking constraints. For each BIC, create an *alternative solution* by assigning one unit of capacity to the BIC while holding all other VP's to their current capacity. For each alternative solution, compute the blocking drift. If a larger number is obtained for all alternatives, then there appears to be no way of satisfying the blocking constraints for the given network load; **exit**. Else, select the alternative with the lowest blocking drift and make it the current solution.

Step 2: Check for nodes that violate signaling capacity constraints. If none are found, and there are no blocking constraint violations, proceed to Step 3, else return to Step 1. Else (there exist capacity violations), compute the *capacity drift* D_c from

$$D_c = \sum_{v} \max(0, \lambda_v - \mu_v), \qquad v \in V. \tag{2}$$

Similarly, this quantity represents our "distance" from satisfying signaling capacity constraints. Every VP whose capacity can be increased by one unit is considered a BIC and an alternative solution is constructed in the same way as above by adding capacity to the BIC while holding all other VP's at their present capacity. If the BIC set is empty, then exit. For each alternative solution, compute the capacity drift. If all alternatives have a higher capacity drift from the current solution, there appears to be no way of satisfying the signaling capacity constraints; **exit**. Else, select the alternative with the lowest capacity drift, make it the current solution, and go to Step 1.

Step 3: Now we have satisfied all constraints. Every VP whose capacity can be increased by one unit is considered a BIC. First compute the throughput for each SD pair

$$\gamma_w^k = \bigl(1 - P_w^k\bigr)\lambda_w^k. \tag{3}$$

The network revenue per unit of time is then given by

$$J = \sum_w \sum_k \gamma_w^k F_w^k \tag{4}$$

where F_w^k denotes the revenue per unit of time obtained by accepting one call of SD pair w and class k. For each BIC, create an alternative solution and compute the resulting network revenue. Drop the alternatives for which blocking or signaling capacity constraints are violated. If there is no remaining alternative that produced a higher revenue, then **exit**. Else, select the alternative with the highest revenue increase and repeat Step 3.

Intuitively, the algorithm works as follows: In Steps 1 and 2 capacity is added to VP's until a solution is reached that satisfies the problem's constraints. The objective used in each of these steps is representative of the corresponding set of constraints that need to be satisfied. Step 3 attempts to further improve the solution by adding capacity to the VP's that promise a higher increase in revenue while satisfying all constraints.

The algorithm is a hill-climbing procedure, selecting at every step the BIC that promises the best improvement from the current solution. The way the SD blocking probabilities and the call arrival rates at the intermediate nodes are computed is independent from the algorithm itself. This implies that the algorithm can be used unchanged with any other representation of networking resources, admission control policies, or routing schemes.

B. General Assumptions

VP's are established hop-by-hop, using a signaling protocol or a management function activated by a central network management facility. Assuming nonblocking switches with output buffering, at every node along the route of the VP, the necessary networking capacity must be secured from the outgoing link that the VP is traversing. In order to represent link and VP capacities we use the methodology introduced in [12] and [14]. The networking capacity of the output links is described by the SR and of the VP's by the CR [14]. Informally, the SR is a surface in a K-dimensional space (where K is the number of traffic classes) that describes the allowable combinations of calls from each class that can be accepted on the link and guaranteed QoS. The advantage of this representation is that once the SR is measured for the link multiplexer, all admission control decisions can be made by simply checking if the operating state of the link upon a new call arrival lies above or below this region. The CR is a subregion of the SR reserved for exclusive use by a VP. An admission controller is installed at the source node of the VP and accepts a new call if the new operating state of the VP falls within the CR. QoS is guaranteed for all calls on a link (regardless if they are part of a VP or not) if the sum of CR's of all VP's traversing the link is a region strictly below the link's SR. The remaining capacity on a link after allocating CR's for the VP's that traverse it is obtained by subtracting the sum of all CR's from the link's SR. In [14] a calculus for region addition and subtraction operations is introduced.

We also make the following modeling assumptions: the call arrival process for all SD pairs is Poisson and each call has an exponentially distributed holding time. All logical links (VP's or physical links) block independently, and the overflow traffic is also Poisson.[1]

The representation of a capacity region increases in complexity with the number of traffic classes. As a result, in a typical broadband system which will offer between four and ten different traffic classes, both the storage of capacity regions and related operations (addition, subtraction, etc.) will become increasingly complex. For this reason, we present two methodologies for the representation of capacity regions.

1) Equivalent Scalar Capacity Approximation: According to the equivalent scalar capacity approximation (ESCA) method, an SR $\mathbf{S}_l$ can be approximated by a scalar quantity representing the capacity of the link, denoted by C_l. This is accomplished by bounding the SR of link l from below by a hyperplane given by the combinations of calls from different traffic classes that occupy an amount (mathematically just below or) equal to the link capacity. Let $\mathbf{c}(C_l)$ represent the vector of equivalent capacities where $c^k(C_l)$ is the "equivalent capacity" of a call of class k over a link of capacity C_l [14]. Let also $\mathbf{x} = (x^1, x^2, \cdots, x^K) \in \mathbf{S}_l$ be the state of the link. The bounding hyperplane is given by

$$H(C_l, C_l) = \left\{ \mathbf{x} \,\middle\|\, \sum_{k=1}^{K} x^k c^k(C_l) \le C_l \right\}. \tag{5}$$

The advantage of this approach is that, given the vector $\mathbf{c}$, an approximation of the SR can be constructed from the scalar C_l. Similarly, a VP traversing link l can also be characterized by a scalar capacity C_p. The CR in this case can be approximated by a hyperplane

$$H(C_p, C_l) = \left\{ \mathbf{x} \,\middle\|\, \sum_{k=1}^{K} x^k c^k(C_l) \le C_p \right\}. \tag{6}$$

In this way, the calculus of regions can be reduced to scalar operations. A sufficient condition to satisfy the capacity constraints for every link is that the sum of the VP capacities traversing the link be less or equal to the link capacity

$$\sum_p C_p \le C_l. \tag{7}$$

2) Generic Hyperplanar Partitioning: According to the generic hyperplanar partitioning (GHP) approach, the SR is represented as an arbitrarily shaped region in the K-dimensional space. The CR's, however, are represented as hyperplanes. This simplifies the implementation of the VP admission controller. Every hyperplanar region can be uniquely represented by a tuple $(m^1, \cdots, m^K)$, where m^k is the number of calls of class k that can be admitted into the system when the number of calls from every other traffic class is 0. The admission control test for VP's is then given by

$$\frac{x^1}{m^1} + \cdots + \frac{x^K}{m^K} \le 1. \tag{8}$$

The advantage of this representation is that higher accuracy can be achieved during addition and subtraction operations (which are performed in the K-dimensional space) while maintaining simplicity for the admission control test, at least for VP's. In reality, the SR is irregularly shaped, and by using this representation we are reducing the "quantization errors" that appear in the ESCA method. There is, however, a complexity penalty for the calculus of regions compared to scalar operations.

[1]The Poisson model is widely used to model the call arrival process in the circuit-switched networks, and we believe it wll be adequate for modeling the call-level behavior in broad-band networks as well.

3) Selection of the Alternative Solution Set: An important aspect in the execution of the algorithm is the construction of the alternative solution set. When using the ESCA approach, each alternative is constructed by increasing the capacity of a BIC by a scalar quantity while holding all other VP's at their present capacity. If the BIC set contains n BIC's, the algorithm constructs n alternative solutions. The value of the objective function is examined for each alternative and the alternative that provides the best improvement is chosen.

When adding capacity to a VP in the GHP approach, the shape of the new hyperplanar region has to be specified as well. In order to determine what is the most appropriate shape for the CR, the following technique is employed. For each BIC, K alternative solutions are constructed. The CR of the BIC for each alternative is derived from the original by expanding into one of the K possible directions by a certain amount (which can differ between classes). Thus, the alternative solution set consists of possibly more than one alternative for each BIC. The algorithm will then compute the objective function for each alternative and will select the VP and the CR shape that provides the best improvement in the objective function. In this way, after several iterations of the algorithm, CR's can take an arbitrary hyperplanar shape.

A special case of the above is when the complete partitioning (CP) admission control policy is used. In the GHP approach, both the SR's and the CR's have a hyper-rectangular shape. The calculus of hyper-rectangular regions is straightforward and the alternative solution set is derived in a similar way, by producing K alternatives for each BIC, each one of them constructed by expanding the CR in one of the K possible dimensions. The ESCA approach requires some additional calculations since we need to determine the capacity increments for each traffic class from the capacity increase step (which, in this case, is a scalar).

The capacity of every link l is characterized by the tuple

$$\left(C_l^1, \cdots, C_l^K\right) \quad \text{where} \quad \sum_k C_l^k = C_l. \tag{9}$$

The CR of a VP can be characterized in a similar way. In order to compute the capacity tuple of every BIC, we compute the maximum capacity for each traffic class $c_{\max}^k$ which can be added to the VP. The capacity of the BIC is then given by

$$\left(C_p^1 + y^1, \cdots, C_p^K + y^K\right) \quad \text{where} \quad y^k = s\frac{c_{\max}^k}{\sum_k c_{\max}^k} \tag{10}$$

and s is the total added capacity to the VP. Thus, if available, the capacity s is proportionally partitioned between traffic classes based on the maximum capacity available to the VP for every class. In this way, capacity assignment is not "blocked" for other classes when there is no capacity left for a specific class. It can be easily verified that the sum of VP capacities

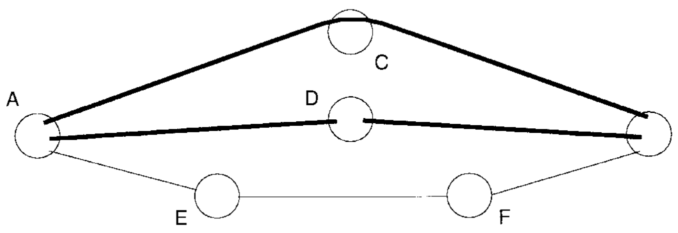

Fig. 4. Example of route selection.

on the link is less than or equal to the link capacity, and also the sum of capacities allocated for class k in every VP is less than or equal to the capacity available for class k in the link. Further, for each VP p, the sum of capacities assigned to every traffic class is equal to the VP capacity C_p

$$\sum_p C_p \le C_l \qquad \sum_p C_p^k \le C_l^k \qquad \sum_k C_p^k = C_p. \tag{11}$$

4) The Route Selection Policy: We assume a static priority route selection scheme similar to dynamic hierarchical routing in circuit-switched networks. This choice was made to simplify the analysis. The VP distribution algorithm, however, is not tied to a particular routing scheme. Other schemes such as adaptive routing (where the route for a call is dynamically determined based on real-time resource utilization information) can be applied as well.

According to the static priority routing model, the route set R_w consists of N_w routes. An incoming call first attempts the first route in the set. If the call is blocked, the second route is attempted, and so on. The call is blocked if no available capacity was found on all the routes in the set. As mentioned previously, every route consists of one or more logical links. Every logical link is either a physical link or a VP.

Fig. 4 shows an example of a route selection policy. For the SD pair (A,B), the first attempted route is comprised of the direct VP from A to B. The second route consists of two VP's and the intermediate node D. The third route does not contain any VP's. In that case the call request goes through the intermediate nodes E and F.

C. Finding the Blocking Probabilities

Let us now focus on a single logical link. We denote by λ_l^k and μ_l^k the arrival rate and the inverse of the holding time of class $\mathbf{k}$ calls arriving on link l. Let also $\mathbf{x} = (x^1, x^2, \cdots, x^K) \in \mathbf{S}_l$ be the state of the link. $u_l^k(\mathbf{x})$ denotes the state dependent admission control policy for class k calls [13]. We also define some set macros that will be used in the sequel:

$\mathrm{Links}(r)$	set of (logical) links composing route r;
$\mathrm{Nodes}(r)$	set of nodes on route r;
$\mathrm{First}(r,l)$	set of links of route r prior to link l;
$\mathrm{Routes}(l)$	set of routes traveling over link l;
$\mathrm{Out}(v)$	set of outgoing links from node v.

Let $\pi_l(\mathbf{x})$ be the equilibrium probabilities of the corresponding Markov chain. The global balance equations can be written as

$$\begin{gathered}\pi_l(\mathbf{x}) \sum_{\mathbf{y}\in\mathbf{S}_l} q_l(\mathbf{x},\mathbf{y}) = \sum_{\mathbf{y}\in\mathbf{S}_l} \pi_l(\mathbf{y}) q_l(\mathbf{y},\mathbf{x}) \\ \sum_{\mathbf{y}\in\mathbf{S}_l} \pi_l(\mathbf{y}) = 1\end{gathered} \tag{12}$$

where

$$q_l(\mathbf{x},\mathbf{y}) = \begin{cases} u_l^k(\mathbf{x})\lambda_l^k, & \text{if } \mathbf{y} = \mathbf{x} + \mathbf{e}^k \\ x^k \mu_l^k, & \text{if } \mathbf{y} = \mathbf{x} - \mathbf{e}^k \end{cases} \tag{13}$$

and $\mathbf{e}^k$ is the elementary vector with a one at position k and zeros everywhere. After solving the above system of equations, we can compute the blocking probability of the link for each class as

$$P_l^k = 1 - \sum_{\mathbf{x}\in\mathbf{S}_l} u_l^k(\mathbf{x})\pi_l(\mathbf{x}) = \sum_{\mathbf{x}\in\mathbf{S}_l : u_l^k(\mathbf{x})=0} \pi_l(\mathbf{x}). \tag{14}$$

Let us now focus on the route level. The probability that a call is blocked on route r_{wi} is

$$P_{wi}^k = 1 - \prod_{l\in\mathrm{Links}(r_{wi})} \left(1 - P_l^k\right). \tag{15}$$

Therefore the call load offered to route r_{wi} is

$$\lambda_{wi}^k = \begin{cases} \lambda_w^k, & i = 1 \\ \lambda_w^k \prod_{j=1}^{i-1} P_{wj}^k, & 1 < i \le N_w. \end{cases} \tag{16}$$

The blocking probability for SD pair w is given by

$$P_w^k = \prod_{j=1}^{N_w} P_{wj}^k \tag{17}$$

and thus the total load offered to link l is

$$\lambda_l^k = \sum_{r\in\mathrm{Routes}(l)} \lambda_r^k \prod_{q\in\mathrm{First}(r,l)} \left(1 - P_q^k\right). \tag{18}$$

Finally, the load on the link due to route r is

$$\lambda_{l,r}^k = \lambda_r^k \prod_{q\in\mathrm{First}(r,l)} \left(1 - P_q^k\right). \tag{19}$$

It is usually realistic to assume that the time between the arrival of the call setup message on a link (when the resources are actually reserved) and the time the resources are freed because the call setup request was not successful is negligible compared to the holding time of an accepted call. This is true for example when the signaling protocol supports timeouts during call setups. Otherwise, it is necessary to compute the contribution of the call setup time in the call holding time as in [15]. For example, in the Xunet ATM testbed a call setup request is timed out after 2 s if no reply has been received, an interval significantly smaller than the average holding time of a call.

In this case, we have that

$$\frac{1}{\mu_l^k} = \frac{1}{\lambda_l^k} \cdot \sum_{r\in\mathrm{Routes}(l)} \left(\lambda_{l,r}^k \frac{1}{\mu_r^k}\right). \tag{20}$$

where $\mu_r^k = \mu_w^k$ for all routes r that belong to SD pair w. The P_l^k can be determined using a fixed point approximation.

Starting with an arbitrary initial value in [0, 1] for the P_l^k, the λ_l^k and μ_l^k can be evaluated from (18) and (20), respectively. Using the obtained values, the P_l^k can be determined from (14). The new values are then used in the next step. The procedure stops when the improvement to the solution has satisfied a convergence criterion. In our experiments the procedure stops when the improvement for every P_l^k has dropped below 10^{-6}.

We can rewrite (14) as a function of the offered load, and the capacity region of the link in the following form:

$$P_l^k = F\left(\frac{\lambda_l^k}{\mu_l^k}, \mathbf{S}_l, \mathbf{u}_l\right), \qquad l = 1, \cdots, |E'|. \tag{21}$$

The above set of equations defines a continuous mapping from the compact set $[0, 1]^L$ into itself and thus, by Brouwer's fixed point theorem, there exists a solution. In [16] and [17] it is proven that the solution is unique, and the loss probabilities calculated by this solution converge to the exact loss probabilities.

Note that the above algorithm can be easily parallelized. At every step, a separate processor can compute the P_l^k for every link only with the knowledge of the P_l^k in the previous computation step. As an example, suppose that the routing scheme for each SD pair contains a direct route using one VP and a link-only route. The direct route r_{w0} is tried first, and then the link-only route r_{w1}. The throughput of calls for SD pair w is then given by

$$\gamma_w^k = \lambda_w^k(1 - P_{w0}^k) + \lambda_w^k P_{w0}^k \prod_{l \in r_{w1}} (1 - P_l^k). \tag{22}$$

D. Networks with Adaptive Routing

The algorithm is also capable of working with more complex routing schemes than the static priority scheme used in the previous section. The main deficiency of this scheme is that for every call setup, the routes are always attempted in a certain order, which leads to a waste of signaling resources in case of blocking. Adaptive routing networks such as [5] can select the most appropriate route in which to establish the call, thus making more efficient use of the signaling system. An analytical model for the adaptive routing scheme used in the AT&T long distance network was presented in [4].

The difficulty which arises in the analysis of adaptive routing networks is the estimation of the offered load on every route. In order to compute this quantity, the probabilities of selecting an alternate route have to be derived. The latter are obtained from the link state probabilities and (in the case of [4]) also from the node-to-node blocking probabilities. The Erlang fixed point approximation can be used again in this case to provide a solution.

E. Complexity Analysis

This section provides an evaluation of the complexity of the algorithm. The number of steps until the termination is bounded by the total capacity in the network is divided by the capacity increase step. For every step, the algorithm computes a number of alternatives that, in the worst case, equals the number of VP's times the number of traffic classes in the network. In order to evaluate every alternative the algorithm must calculate blocking probabilities for every link and VP. When using the approximation technique, the calculation of the blocking probabilities is reduced to a simple Erlang blocking formula whose complexity is proportional to the capacity of the link. However, when an accurate computation of the blocking probabilities is necessary [by solving the system of (12)], the complexity of this operation is $O(N^3)$, where N is the number of states in the capacity region.

Overall, the algorithm executes in polynomial time. As the following section will reveal, the calculation of the blocking probabilities is the highest contribution to the execution time and in order for the algorithm to complete in a few seconds (the desirable time frame for a network management application) it is necessary to apply approximation techniques.

IV. Experimental Results

In this section we present exploratory results obtained by running our VP capacity allocation algorithm on networks of different sizes and evaluating its complexity. We also evaluate the performance of the algorithm parameterized by the network offered load. Unfortunately, since our algorithm introduces new considerations into the VP distribution problem, such as flexible routing and admission control policies, signaling capacity constraints, and the support of multiple traffic classes on the same VP, it has not been possible to make a straightforward comparison with other existing algorithms for VP distribution because the latter regards the problem in a simpler setting.

A. Experimenting with a Small Network

We first study a small network with two SD pairs and two VP's that share a common link. The network model is shown in Fig. 3. For each SD pair, there is a direct route with a direct VP to the destination, and one alternative route (also referred to as the link-only route) traveling over links (1, 3, 4) and (2, 3, 5) for each SD pair, respectively. The holding time of all calls is assumed to be 1 min. The number of traffic classes is $K = 2$. The first traffic class corresponds to a video service and the second to a voice service. Every link has a capacity of 45 Mb/s, approximating the DS3 transmission rate. We used the ESCA approach to characterize the SR and CR. We assumed a capacity vector $\mathbf{c} = (4, 0.064)$ to characterize the networking capacity occupied by every traffic class. The arrival rates for the two SD pairs (in calls per minute) are

$$\lambda_1^1 = 2.0, \quad \lambda_1^2 = 100.0, \quad \lambda_2^1 = 1.0, \quad \lambda_2^2 = 400.0.$$

We employed three different admission control policies: the first was complete sharing (CS), as described in [13]; the second was a variation of CS that admits calls of class 2 (narrow-band traffic) only if it is possible to accommodate an additional arriving call of class 1. We will refer to this scheme as CS with wide-band reservation (CS_WB) because capacity is always reserved for a wide-band (class 1) call. Reservation results in reduced blocking probabilities for class 1 traffic compared to the CS policy. The third scheme uses CS for links 1–5 and CS_WB for the two VP's (logical links 6 and 7) only. We will call this scheme CS_VPWB.

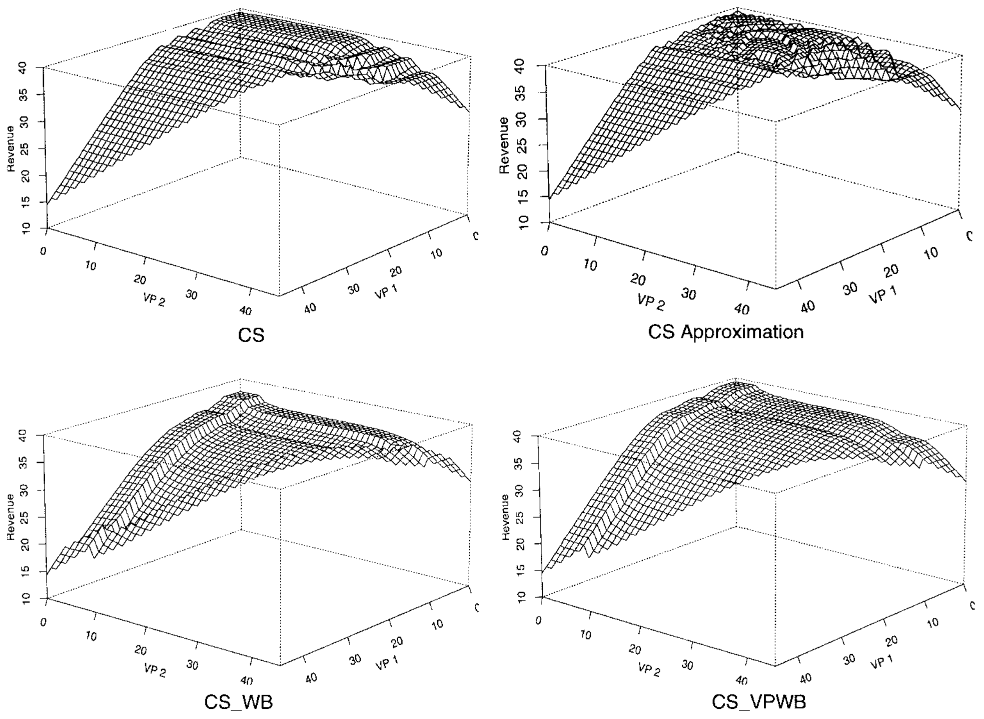

Fig. 5. Network revenue versus allocated capacity to VP's.

We defined the revenue for each call to be proportional to its capacity and holding time. In order to evaluate the problem we temporarily removed the SD blocking and signaling capacity constraints and plotted the total network revenue versus the size of the CR's of the two VP's. Note that the sum of the VP capacities is always less or equal to the capacity of link 3. We used (22) to compute the throughput for each SD pair and (14) to compute the link blocking probabilities. Because this involves an intense numerical computation that provides the solution to a large system of linear equations (of the order of $O(N^3)$, where N is the number of states in the capacity region), we also provide the results for the CS case by using the approximation technique presented in [19]. According to this technique, the blocking probabilities of each traffic class in the CS case can be approximated by simple Erlang formulas. We have found this technique to be very useful for network links with capacities greater than 45 Mb/s, when the solution to the system of global balance equations becomes prohibitively expensive. The network revenue for every admission control policy used is plotted in Fig. 5.

We immediately observe that the results obtained by employing the CS approximation are almost identical to the ones obtained by using the CS admission control policy. The CS_WB policy produces less network revenue compared to the CS policy, because class 2 calls are rejected to reserve space for an anticipated class 1 call. The CS_VPWB is almost identical to the CS_WB. In every case the revenue starts dropping quickly when a large percentage of the bandwidth of link 3 is reserved to a single VP. The drop in revenue is higher for VP1, because the first SD pair has an overall lower bandwidth demand.

An HP 755 workstation was used to compute the results shown above. The solution to (14) for each link requires approximately 6 s of CPU time. This time increases rapidly with the capacity of the link. The entire plots were computed in approximately 65 min. When the CS approximation is employed, the time to obtain the solution to (14) is negligible and the corresponding plot is completed in 5 s.

We applied the VP distribution algorithm starting from the initial solution, where no bandwidth is allocated to VP's, and assuming that the signaling processing capacities of nodes C and D are 200 calls/min (or 400 signaling messages per minute since every call request involves two signaling messages—one in each direction). The blocking constraints for each SD pair were set to 0.6 for class 1 and 0.1 for class 2. We conducted three experiments, using the CS, CS approximation, and CS_WB policies, respectively. Table II shows the computed blocking probabilities for each SD pair and traffic class, and the VP capacities at the end of each phase.

The first set of results was obtained for the CS policy. Note that the initial solution (Phase 1) does not satisfy the node signaling capacity constraints, because the arrival rates at node C and D (which are 503 and 498 calls/min—the arrivals at node D are lower due to blocking at link 3) are higher than the signaling capacity of these nodes. The algorithm then enters the second phase (Steps 1 and 2), where it increases the VP capacities until the node signaling capacity constraints are satisfied. The second phase terminates with the VP capacities set at 3 and 17 Mb/s, respectively. The arrival rate at node C is then below 200 calls/min and the blocking probabilities for the two SD pairs are 0.37 and 0.38 for class 1 and 0.004 and 0.003 for class 2, respectively. The algorithm then enters the third phase (Step 3), where it allocates bandwidth to the VP's in the direction of the highest revenue increase while the blocking constraints and the signaling capacity constraints are satisfied. The algorithm then reaches the final solution where the VP

TABLE II
RESULTS FOR THE SMALL NETWORK.

Policy	CS				CS Approximation				CS_WB			
	SD 1	SD 2	VP1	VP 2	SD 1	SD 2	VP 1	VP 2	SD 1	SD 2	VP 1	VP 2
Phase 1	0.37 0.005	0.38 0.006	0.0	0.0	0.37 0.006	0.39 0.006	0.0	0.0	0.11 0.158	0.11 0.162	0.0	0.0
Phase 2	0.37 0.004	0.38 0.003	3.0	17.0	0.39 0.005	0.41 0.002	1.0	19.0	0.32 0.17	0.05 0.16	4.0	24.0
Phase 3	0.37 0.004	0.38 0.002	3.0	18.0	0.39 0.005	0.41 0.002	1.0	19.0	0.34 0.18	0.05 0.15	4.0	25.0

TABLE III
CONFIGURATION AND BLOCKING PROBABILITIES AT THE END OF EACH PHASE

	SD 0	SD 1	SD 2	SD 3	SD 4	SD 5	SD 6	SD 7	SD 8	SD 9
Nodes	0 - 4	1 - 8	2 - 3	3 - 6	4 - 5	5 - 0	6 - 5	7 - 6	8 - 1	9 - 5
Arrivals	10.0 600.0	8.0 700.0	8.0 300.0	8.0 400.0	10.0 200.0	10.0 600.0	5.0 400.0	12.0 400.0	8.0 300.0	10.0 200.0
Phase 1	1.4e-01 2.2e-03	6.4e-02 8.9e-04	2.5e-05 3.8e-09	3.7e-03 4.5e-05	8.4e-04 9.7e-06	2.0e-06 1.8e-08	8.4e-04 9.7e-06	1.4e-01 2.2e-03	5.8e-03 7.0e-05	3.8e-02 5.2e-04
Phase 2	8.6e-03 1.1e-05	3.0e-05 1.5e-08	2.3e-10 1.3e-15	1.0e-06 6.0e-10	2.3e-04 1.6e-06	2.5e-10 4.5e-13	2.3e-04 1.4e-07	8.0e-03 4.2e-06	5.7e-04 3.3e-07	6.0e-04 1.6e-06
Phase 3	6.0e-17 0.0e+00	2.8e-16 0.0e+00	3.1e-31 0.0e+00	1.0e-16 0.0e+00	2.3e-16 0.0e+00	0.0e+00 0.0e+00	2.7e-15 0.0e+00	7.6e-17 0.0e+00	2.8e-17 0.0e+00	6.4e-17 0.0e+00

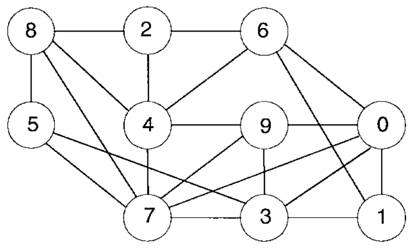

Fig. 6. Network topology.

capacities are 3 and 18 Mb/s, and the SD blocking probabilities are 0.37 and 0.38 for class 1 and 0.004 and 0.002 for class 2.

Using the approximation technique with the CS admission control policy, we obtain a slightly different solution for the VP capacities: 1 and 19 Mb/s, respectively. The blocking probabilities for the two SD pairs are 0.39 and 0.41 for class 1 and 0.005 and 0.002 for class 2. Computing the exact value of the probabilities gives 0.37 and 0.38 for class 1 and 0.006 and 0.002 for class 2. This shows that the approximation provides a very satisfactory value for the blocking probabilities.

We also evaluated the VP distribution under the CS_WB policy. In this case we slightly raised the blocking constraints for both SD pairs for class 2 to 0.3. Phase 2 concludes with VP capacities 4 and 24 Mb/s, respectively. The blocking probabilities are 0.32 and 0.05 for class 1 and 0.17 and 0.16 for class 2. Step 3 concludes with the VP capacities set at 4 and 25 Mb/s. The blocking probabilities are 0.34 and 0.05 for class 1 and 0.18 and 0.15 for class 2.

B. Experimenting with a Larger Network

We now apply the algorithm to a larger network (also used in [3]) with a significantly larger number of SD pairs and VP's. The topology of the network is shown in Fig. 6. All links are bidirectional and carry 155 Mb/s. The SR's and CR's are characterized using the ESCA method. There is a total of ten SD pairs (see Table III). For each pair, we define one direct VP route (chosen to be the shortest path route) and one route that consists of links only and is the second shortest path route. We have used the CS admission control policy and the approximation of [19] to compute the blocking probabilities due to the large size of the problem (a total of 52 links including the VP's). The algorithm provided the final solution in less than 5 s.

The row titled "Nodes" shows the source and destination node of each SD pair. The "Arrivals" row shows the arrival rates for class 1 and class 2 traffic in calls per unit of time. The holding time is assumed the same for all calls and is equal to one unit of time. The call blocking constraints were set at 0.01 for class 1 and 0.001 for class 2. The remaining rows show the blocking probabilities for each class at the end of each of the three phases of the algorithm. Phase 1 again represents the initial solution with all VP's still at zero capacity. Phase 2 shows the result after running Steps 1 and 2, and Phase 3 the final solution (after completing Step 3). The signaling capacity of every processor was set to 200 calls/min. Table IV presents the arrival rates at every node at the end of every phase.

The initial solution (Phase 1) violates the signaling capacity constraints for all nodes. Blocking constraints are also violated for class 1 and SD pairs 0, 1, 7, and 9. At the end of Phase 2, the blocking constraints and signaling capacity constraints are satisfied by assigning capacity to VP's. Phase 3 continues assigning extra capacities to VP's while both the blocking and capacity constraints are satisfied. Table V shows the VP capacities and the total network revenue at the end of each phase.

We note that at the end of Phase 3 all SD pairs have significantly reduced blocking probabilities and the revenue

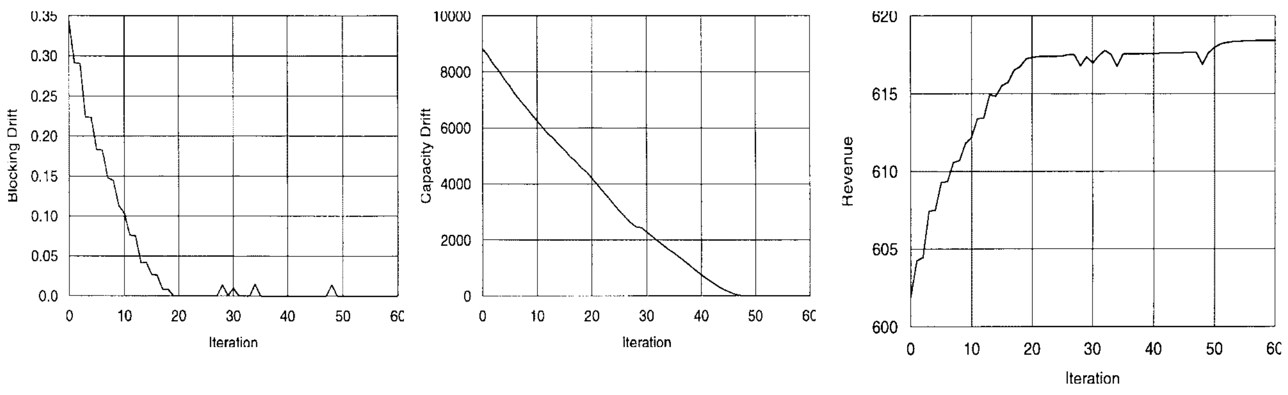

Fig. 7. Algorithm parameters versus number of iterations.

TABLE IV
CALL ARRIVAL RATES

Node	0	1	2	3	4	5	6	7	8	9
Phase 1	1022.0	1424.0	713.4	1633.4	518.0	1317.2	1320.3	1022.0	923.4	930.0
Phase 2	92.4	93.8	49.5	148.0	154.8	182.0	116.1	168.4	181.7	102.7
Phase 3	4.5	22.8	10.0	23.4	3.5	30.6	12.0	18.2	9.9	6.6

TABLE V
VP CAPACITIES

	VP 0	VP 1	VP 2	VP 3	VP 4	VP 5	VP 6	VP 7	VP 8	VP 9	Revenue
Nodes	0 - 4	1 - 8	2 - 3	3 - 6	4 - 5	5 - 0	6 - 5	7 - 6	8 - 1	9 - 5	
Phase 1	0	0	0	0	0	0	0	0	0	0	601924
Phase 2	35	45	20	25	5	30	25	30	20	10	617611
Phase 3	90	60	50	55	60	55	40	80	60	55	618400

is higher compared to the end of Phase 1. This happens because the capacity that was added to the VP's is essentially increasing the end-to-end capacity available for the SD pairs. This behavior was not observed in the network of Fig. 3, because increasing the capacity of the VP's resulted in decreased capacity available on the link-only routes and, consequently, decreased throughput.

The plots of Fig. 7 show the sample path of three important quantities during the execution of the algorithm. The first plot shows the blocking drift versus the number of iterations. An iteration here is considered to be any VP capacity increase that takes place at any of the Steps 1, 2, and 3 of the algorithm. The blocking constraints are initially satisfied after 20 iterations. The second plot shows the capacity drift, which is satisfied in 48 iterations. Notice that between iterations 20 and 48, when the algorithm tries to satisfy the capacity constraints, the blocking constraints are occasionally violated and satisfied again in the following iteration. The last violation occurs at the 49th iteration. The third plot shows the revenue at every iteration. Note how the revenue fluctuates while in Phase 2 and then becomes strictly increasing in Phase 3, until it levels off after the 60th iteration.

In order to examine the robustness of the algorithm when the load increases we rerun the algorithm by increasing the load homogeneously up to 100% of the original load. Beyond that point, the algorithm failed to satisfy the blocking constraints for class 1 traffic. In order to accept a wider range of offered loads we increased our blocking constraints to 0.05 for class 1 and 0.01 for class 2.

The graphs of Fig. 8 show that an increased load for the same network requires additional iterations for the algorithm in order to complete Phase 2 successfully, while the total number of iterations remains about the same. These iterations are "stolen" from the third phase.

Another experiment involved computing the VP distribution policy for the initial configuration and then evaluating the performance of the network by increasing the offered load while keeping the VP distribution fixed. Fig. 9 demonstrates the results: blocking constraint violations occur only after the load increases by 60%. This is due to the low blocking probabilities achieved at the end of Phase 3 for the initial load. Signaling capacity violations, however, start already at a 50% load increase. This implies that the VP distribution for this example will still satisfy all constraints, even if the load increases by 50%. The solid revenue curve shows the increase in revenue as the load increases. The dotted curve corresponds to the revenue that would be obtained by recalculating the distribution policy. Recomputing the VP distribution policy results both in satisfying the blocking and signaling capacity constraints and increasing the network revenue.

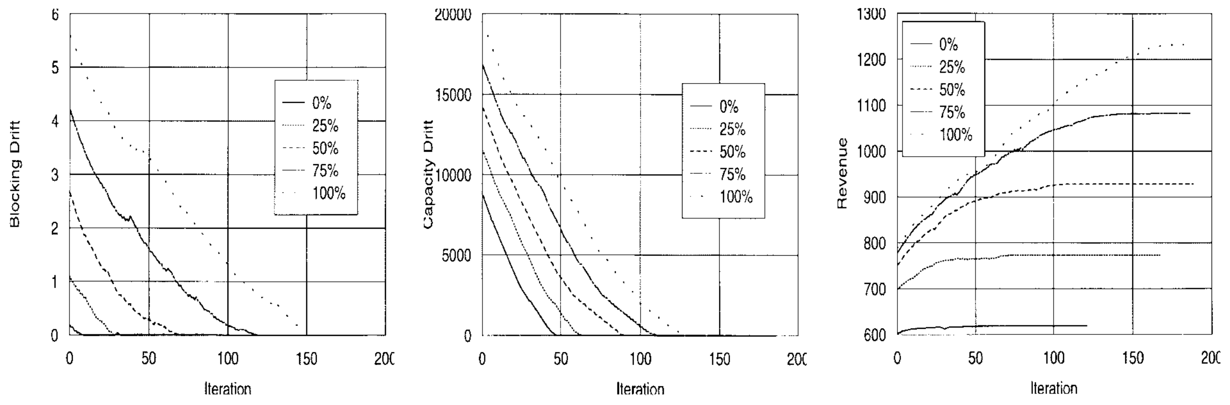

Fig. 8. Algorithm behavior under different loads.

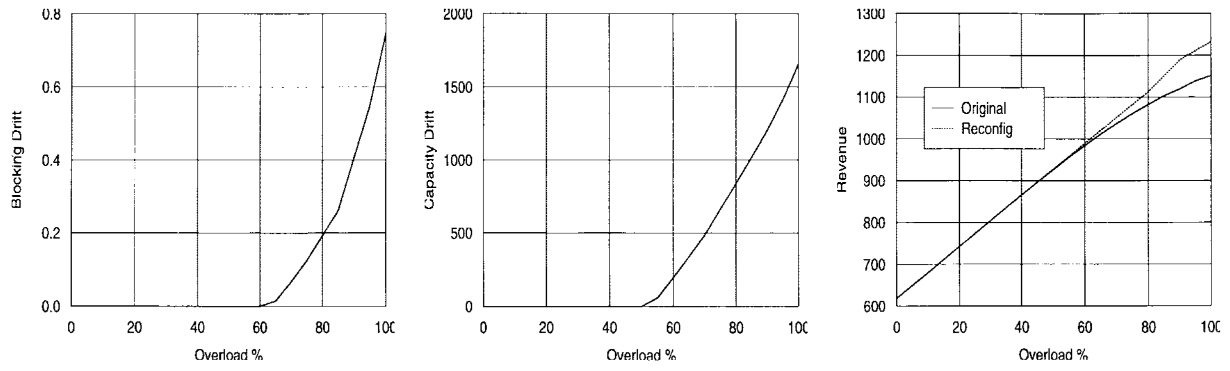

Fig. 9. Resilience of the solution to overload.

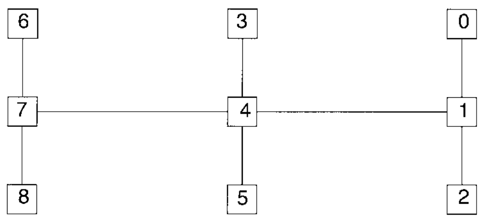

Fig. 10. Xunet topology.

C. *Experimenting on the Xunet Testbed*

In this example we applied the VP distribution algorithm on the Xunet ATM testbed. The topology of the network is shown in Fig. 10. All the links are bidirectional with a capacity of 45 Mb/s. For this experiment, we have used a CP admission control policy and the GHP method to characterize the SR's and CR's. Two-thirds of the capacity of every link was allocated exclusively to class 1 traffic, and the remaining one-third to class 2 traffic. We used a blocking constraint of 0.6 for class 1 and 0.3 for class 2. The capacity of the Xunet signaling processors is 400 calls/min. Our routing scheme provided a direct VP for each SD pair and a link-only route for the overflow traffic of the VP.

After applying the VP distribution algorithm, we observe that the initial solution (Phase 1) does not satisfy the signaling capacity constraints at nodes 1, 4, and 7. In particular, the arrival rate at node 4 is 766 calls/min. The blocking constraints are satisfied for all SD pairs. Table VI shows the arrival rates for the SD pairs and the observed blocking probabilities at the end of each phase.

TABLE VI
SD Call Arrival Rates and Blocking Prbabilities for Complete Partitioning

	SD 0	SD 1	SD 2	SD 3	SD 4	SD 5	SD 6	SD 7
Nodes	0 - 5	0 - 8	6 - 5	8 - 3	5 - 0	3 - 6	6 - 0	3 - 8
Arrivals	1.0 100.0	0.8 80.0	1.0 100.0	0.8 80.0	1.8 180.0	1.0 100.0	0.5 50.0	0.8 80.0
Phase 1	0.296 0.001	0.409 0.123	0.349 0.041	0.224 0.041	0.361 0.055	0.332 0.123	0.466 0.093	0.346 0.123
Phase 2	0.285 0.000	0.387 0.111	0.303 0.024	0.255 0.047	0.302 0.031	0.355 0.144	0.534 0.113	0.341 0.111
Phase 3	0.236 0.000	0.408 0.111	0.341 0.030	0.208 0.023	0.273 0.018	0.350 0.144	0.590 0.147	0.336 0.111

Table VII shows the configuration of VP's in the system and the capacity (in Mb/s) allocated to each traffic class. Note that the capacity tuple for each VP can uniquely identify a rectangular CR.

The algorithm completes in only ten iterations. The reader will quickly observe that the VP's that carry the traffic of the most loaded SD pairs have been assigned more capacity. Also, SD 6 that is not assigned any VP capacity experiences increased blocking. On the other hand, SD's 0 and 4 that have been assigned the highest VP capacities have significantly reduced blocking probabilities.

We conducted the same experiment using a CS admission control policy for every link and VP. In this case, the CR of every VP is represented as a hyperplane, and the VP capacities are given at the points where the hyperplane meets the axes.

TABLE VII
VP CAPACITIES AND REVENUE FOR COMPLETE PARTITIONING

	VP 0	VP 1	VP 2	VP 3	VP 4	VP 5	VP 6	VP 7	Revenue
Nodes	0 - 5	0 - 8	6 - 5	8 - 3	5 - 0	3 - 6	6 - 0	3 - 8	
Phase 1	0.0 0.0	0.0 0.0	0.0 0.0	0.0 0.0	0.0 0.0	0.0 0.0	0.0 0.0	0.0 0.0	106498
Phase 2	4.0 6.4	4.0 3.2	4.0 3.2	0.0 0.0	8.0 6.4	4.0 3.2	0.0 0.0	4.0 3.2	108036
Phase 3	8.0 6.4	4.0 3.2	4.0 3.2	4.0 3.2	12.0 9.6	4.0 3.2	0.0 0.0	4.0 3.2	108936

TABLE VIII
SD CALL ARRIVAL RATES AND BLOCKING PROBABILITIES FOR COMPLETE SHARING

	SD 0	SD 1	SD 2	SD 3	SD 4	SD 5	SD 6	SD 7
Nodes	0 - 5	0 - 8	6 - 5	8 - 3	5 - 0	3 - 6	6 - 0	3 - 8
Arrivals	1.0 100.0	0.8 80.0	1.0 100.0	0.8 80.0	1.8 180.0	1.0 100.0	0.5 50.0	0.8 80.0
Phase 1	0.218 0.003	0.387 0.007	0.318 0.004	0.228 0.001	0.343 0.002	0.337 0.007	0.465 0.004	0.342 0.007
Phase 2	0.260 0.001	0.442 0.005	0.351 0.003	0.245 0.003	0.368 0.005	0.384 0.002	0.482 0.013	0.388 0.005
Phase 3	0.262 0.001	0.442 0.005	0.342 0.000	0.234 0.001	0.343 0.000	0.378 0.000	0.473 0.003	0.382 0.005
Xunet Experiment	0.242 0.001	0.433 0.014	0.323 0.002	0.190 0.004	0.335 0.000	0.392 0.005	0.447 0.010	0.367 0.025

TABLE IX
VP CAPACITIES AND REVENUE FOR COMPLETE SHARING

	VP 0	VP 1	VP 2	VP 3	VP 4	VP 5	VP 6	VP 7	Revenue
Nodes	0 - 5	0 - 8	6 - 5	8 - 3	5 - 0	3 - 6	6 - 0	3 - 8	
Phase 1	0.0 0.0	0.0 0.0	0.0 0.0	0.0 0.0	0.0 0.0	0.0 0.0	0.0 0.0	0.0 0.0	111628
Phase 2	4.0 6.4	4.0 3.2	4.0 3.2	4.0 3.2	8.0 6.4	4.0 3.2	0.0 0.0	0.0 0.0	108377
Phase 3	4.0 6.4	4.0 3.2	4.0 6.4	4.0 3.2	8.0 9.6	4.0 6.4	0.0 0.0	0.0 0.0	109417

The results are shown in Table VIII. The last row of the table provides experimental results after installing the configuration obtained at the end of Phase 3 on the Xunet testbed. We used the management system of [2] to configure the VP's and the call generators and to obtain the blocking probabilities. The measurements were taken over a period of 6 h, during which a total of approximately 300 000 calls were generated. Due to limited network access we were not able to run the experiment many times in order to obtain a confidence interval for the above measurements. The experimental results show that the actual blocking probabilities approximate very closely the figures predicted by the VP distribution algorithm. The obtained figures when compared to the prediction (Phase 3) appear, in general, to be lower for class 1 and higher for class 2. In either case, they are well below the SD blocking constraints. The reader may have noticed that we use rather high numbers for the blocking constraints. The reason is that when verifying the VP distribution on the testbed, the experiment requires a smaller number of generated calls to provide accurate blocking measurements. If we were using blocking constraints of the order of, e.g., 10^{-3}, we would need to run the experiment for significantly longer times to obtain accurate measurements.

Table IX shows the VP capacities and resulting network revenue.

A comparison of CS and CP reveals the following: although the total attained revenue is almost the same, the blocking for class 1 appears to be increased for CS. On the other hand, blocking for class 2 in the CS case is significantly reduced for most SD pairs. By setting lower blocking constraints for class 1, we found that it was not possible to reach a feasible solution in both the CS and CP cases. That was only possible when the initial solution did not violate the blocking constraints. This result is intuitively understandable for topologies where the VP's available for every SD pair share the link-only routes. By contrast, in the network of Section IV-B it is possible to further reduce the blocking probabilities of the initial solution if VP's follow different routes.

V. CONCLUSIONS AND FURTHER STUDY

This paper addressed the VP distribution problem for ATM networks. We presented previous approaches to the problem and identified their strengths and weaknesses. We showed how VP's can be used to tune the fundamental tradeoff between call throughput and overall performance of the signaling system. We provided an algorithm for the VP distribution problem that, for the first time, satisfies the combination of nodal constraints on the processing of signaling messages and constraints on blocking for each SD pair. The algorithm maximizes the network revenue under the above set of constraints and works independently of the number of traffic classes in the network, the admission control policy used in every link, and the network routing scheme. We provided a solution methodology for a static priority routing scheme. The methodology can be easily extended to networks with adaptive routing policies.

We finally applied the algorithm to three sample networks and studied various performance characteristics.

We have intentionally left open so far the topic of determining what VP's to establish between every SD pair. Future research in the area of VP control will need to address the joint problem of VP capacity allocation and routing. In the present setting, the VP topology and routing scheme is given as input to the algorithm. In reality, however, the choice of the VP topology has obvious implications on the optimality of the final solution. The main difficulty that arises in the joint VP capacity allocation and routing problem is that the optimal routing component is NP-complete. From the experiments we have conducted, we are led to believe that even if our solution is suboptimal by predetermining the set of available routes for the VP distribution problem, the end result can be nearly optimal if many link-disjoint routes are available for every SD pair. This increases the (shared) end-to-end available capacity for each SD pair and, as a result, the end-to-end blocking probability is reduced. This is also why adaptive routing schemes are performing well when given a large number of alternative routes without recomputing an optimal route set every time the network loading matrix has changed.

References

[1] N. Anerousis and A. A. Lazar, "Managing virtual paths on Xunet III: Architecture, experimental platform and performance," in *Proc. 1995 IFIP/IEEE Int. Symp. Integrated Network Management*, Santa Barbara, CA, May 1–5, 1995.

[2] N. Anerousis, A. A. Lazar, and D. E. Pendarakis, "Taming Xunet III," *ACM Comput. Commun. Rev.*, vol. 25, no. 3, pp. 44–65, Oct. 1995.

[3] A. Ardvisson, "Management of reconfigurable virtual path networks," in *Proc. 1994 Int. Teletraffic Congress*, J. Labetoulle and J. W. Roberts, Eds. Amsterdam, The Netherlands: Elsevier, 1994.

[4] G. R. Ash and B. D. Huang, "An analytical model for adaptive routing networks," in *IEEE Trans. Commun.*, vol. 41, Nov. 1993.

[5] G. R. Ash, J. S. Chen, A. E. Frey, and B. D. Huang, "Real-time network routing in a dynamic class-of-service network," in *Proc. 13th Int. Teletraffic Congress*, Copenhagen, Denmark, June 1991.

[6] I. Chlamtac, A. Farago, and T. Zhang, "Optimizing the system of virtual paths," *IEEE/ACM Trans. Networking*, vol. 2, pp. 581–587, Dec. 1994.

[7] A. Gersht and A. Shulman, "Optimal dynamic virtual path bandwidth allocation and restoration in ATM networks," in *Proc. GLOBECOM'94*, San Francisco, CA, Nov. 1994.

[8] A. Gersht and S. Kheradpir, "Integrated traffic management in SONET-based multi-service networks," in *Proc. 13th Int. Teletraffic Congress*, Copenhagen, Denmark, June 1991.

[9] A. Gersht and A. Shulman, "Optimal routing in circuit switched communication networks," *IEEE Trans. Commun.*, vol. 37, Nov. 1989.

[10] G. Gopal, C. Kim, and A. Weinrib, "Algorithms for reconfigurable networks," in *Proc. 13th Int. Teletraffic Congress*, Copenhagen, Denmark, June 1991.

[11] J. Y. Hui, M. B. Gursoy, N. Moayeri, and R. D. Yates, "A layered broadband switching architecture with physical or virtual path configurations," *IEEE J. Select. Areas Commun.*, vol. 9, Dec. 1991.

[12] J. M. Hyman, A. A. Lazar, and G. Pacifici, "Real-time scheduling with quality of service constraints," *IEEE J. Select. Areas Commun.*, vol. 9, pp. 1052–1063, Sept. 1991. Available URL: ftp://ftp.ctr.columbia.edu/CTR-Research/comet/public/papers/91/HYM91a.ps.gz.

[13] ——, "A separation principle between scheduling and admission control for broadband switching," *IEEE J. Select. Areas Commun.*, vol. 11, pp. 605–616, May 1993. Available URL: ftp://ftp.ctr.columbia.edu/CTR-Research/comet/public/papers/93/HYM93.ps.gz.

[14] ——, "A methodology for virtual path and virtual network bandwidth assignment in broadband networks with quality of service guarantees," in *Proc. IEEE Int. Conf. Communications*, New Orleans, LA, May 1994, pp. 165–169. Available URL: ftp://ftp.ctr.columbia.edu/CTR-Research/comet/public/papers/94/HYM94.ps.gz.

[15] R. H. Hwang, J. F. Kurose, and D. Towsley, "On call processing delay in high speed networks," *IEEE/ACM Trans. Networking*, vol. 3, pp. 628–639, Dec. 1995.

[16] F. P. Kelly, "Routing in circuit-switched networks: Optimization, shadow prices and decentralization," *Adv. Appl. Prob.*, vol. 20, pp. 112–144, 1988.

[17] ——, "Blocking probabilities in large circuit-switched networks," *Adv. Appl. Prob.*, vol. 18, pp. 473–505, 1986.

[18] S.-B. Kim, "An optimal establishment of virtual path connections for ATM networks," in *Proc. 1995 IEEE INFOCOM*, Boston, MA, Apr. 2–6, 1995, pp. 72–79.

[19] J. F. Labourdette and G. W. Hart, "Blocking probabilities in multitraffic loss systems: Insensitivity, asymptotic behavior, and approximations," *IEEE Trans. Commun.*, vol. 40, Aug. 1992.

[20] A. A. Lazar, A. Orda, and D. Pendarakis, "Virtual path allocation in multi-user networks: A noncooperative approach," in *Proc. 1995 IEEE INFOCOM*, Boston, MA, Apr. 2–6, 1995.

[21] F. Lin and K.-T. Cheng, "Virtual path assignment and virtual circuit routing in ATM networks," in *Proc. GLOBECOM'93*, Houston, TX, Dec. 1993, pp. 436–441.

[22] S. Ohta and K.-I. Sato, "Dynamic bandwidth control of the virtual path in an asynchronous transfer mode network," *IEEE Trans. Commun.*, vol. 40, July 1992.

[23] H. Saito, K. Kawashima, and K.-I. Sato, "Traffic control technologies in ATM networks," *IEICE Trans.*, vol. E74, no. 4, Apr. 1991.

[24] K.-I. Sato, S. Ohta, and I. Tokizawa, "Broad-band ATM network architecture based on virtual paths," *IEEE Trans. Commun.*, vol. 38, pp. 1212–1222, Aug. 1990.

[25] S. Shioda and H. Uose, "Virtual path bandwidth control method for ATM networks: Successive modification method," *IECIE Trans.*, vol. E74, no. 12, Dec. 1991.

Nikolaos Anerousis (S'89–M'96) received the Dipl. Eng. degree from the National Technical University of Athens, and the M.S., M.Phil. and Ph.D. degrees from Columbia University, New York, in 1990, 1991, 1994 and 1995 respectively, all in electrical engineering.

He is a Senior Technical Staff Member at AT&T Labs— Research (formerly AT&T Bell Laboratories), Florham Park, NJ. His research interests include network dimensioning and optimization, service pricing, software architectures for management and control of broadband networks, and multimedia services. In 1998 he was also an Adjunct Assistant Professor of electrical engineering at Columbia University. Most of his research work has been in the area of broadband network management. In 1992 he designed and implemented the first operational performance management system using TMN standards on the AT&T Xunet gigabit platform. He is currently investigating scalable lightweight management architectures using mobile agent technology and the World Wide Web.

Dr. Anerousis is a member of the Technical Chamber of Greece.

Aurel A. Lazar (S'77–M'80–SM'90–F'93) is with the Department of Electrical Engineering and the Center for Telecommunications Research, Columbia University, New York, NY, as a Professor of electrical engineering. His research interests span both theoretical and experimental studies of telecommunication networks and multimedia systems.

His current research in broadband networking with QoS guarantees focuses on modeling of video streams and analyzing their multiplexing behavior, with emphasis on multiple time scales and subexponentiality. He is also leading investigations into multimedia networking architectures supporting interoperable exchange mechanisms for interactive and on demand multimedia applications with QoS requirements. The main focus of this work is on building a broadband kernel (also known as xbind) that enables the rapid creation, deployment and management of multimedia services. Resource allocation and networking games algorithms form the core of this architecture. The xbind work lead to the establishment of the IEEE standards working group on Programming Interfaces for Networks. He was instrumental in establishing the OPENSIG (http://comet.columbia.edu/opensig) international working group with the goal of exploring network programability and next generation signaling technology.

Professor Lazar was the Program Chair of the Fall and Spring OPENSIG'96 workshops and of the First IEEE Conference on Open Architectures and Network Programming (OPENARCH'98) (http://comet.columbia.edu/openarch).