Circuit-Switched "Network Capacity" under QoS Constraints

Jeffrey E. Wieselthier, Gam D. Nguyen, and Anthony Ephremides

Abstract: **Usually the network-throughput maximization problem for constant-bit-rate (CBR) circuit-switched traffic is posed for a fixed offered load profile. Then choices of routes and of admission control policies are sought to achieve maximum throughput (usually under QoS constraints). However, similarly to the notion of channel "capacity," it is also of interest to determine the "network capacity;" i.e., for a given network we would like to know the maximum throughput it can deliver (again subject to specified QoS constraints) if the appropriate traffic load is supplied. Thus, in addition to determining routes and admission controls, we would like to specify the vector of offered loads between each source/destination pair that "achieves capacity."**

Since the combined problem of choosing all three parameters (i.e., offered load, admission control, and routing) is too complex to address, we consider here only the optimal determination of offered load for given routing and admission control policies. We provide an off-line algorithm, which is based on Lagrangian techniques that perform robustly in this rigorously formulated nonlinear optimization problem with nonlinear constraints. We demonstrate that significant improvement is obtained, as compared with simple uniform loading schemes, and that fairness mechanisms can be incorporated with little loss in overall throughput.

Index Terms: **Communication network, performance evaluation, optimization, throughput, circuit-switched, quality of service (QoS).**

I. INTRODUCTION

A comprehensive approach to network optimization and control would address jointly the highly interdependent issues of admission control, routing, offered load, and (for wireless networks) channel access. However, each of these individual problems is extremely complex, making a complete solution of the joint problem unattainable. Therefore, it is necessary to simplify the problem by addressing these issues separately in a manner that will, hopefully, lead to useful insights for their joint solution.

In this paper, we consider circuit-switched (i.e., session- or connection-oriented) networks, for which the primary performance criteria are throughput and blocking probability. Our focus is on the off-line determination of the offered load profile that provides optimal performance (i.e., maximum throughput) for the case in which both the routing and admission-control policy are fixed; traffic is generated by Constant Bit Rate (CBR) sources where session arrivals are characterized by Poisson processes. We consider a sufficiently general model that includes classes of both "wired" and "wireless" networks. A fixed quantity of network resources (i.e., link bandwidth in wired networks and transceivers in wireless networks) is allocated to a session throughout its duration.

The problem we address here entails determining the load vector that results in the highest possible value of overall network throughput, subject to a quality-of-service (QoS) constraint on session blocking probability and a fixed set of routes for each source-destination node pair. This load vector consists of the offered loads to each of several pre-established circuits. At the optimal point, the loads may vary greatly on the different circuits, and we show that such a nonuniform loading may provide considerably increased throughput, as compared to the case of uniform loading. The mathematical formulation is that of a constrained optimization problem with a nonlinear objective function and multiple nonlinear constraint functions. This problem is of interest for "sizing" the network capability, and thereby determining a benchmark level of "network capacity."[1] This is an important quantity because, in practice, it is generally difficult to estimate the traffic loads that a network can support or the resulting throughput. Once the network capacity is determined, it is straightforward to determine whether any particular (typically user-specified) offered load profile achieves a level of throughput that is sufficiently close to this benchmark capacity. If it is not, then pricing mechanisms can be introduced to steer the offered load profile in a direction that maximizes overall revenue.

We present an iterative algorithm for the optimization of offered load, subject to QoS constraints. Our approach is based on the use of Lagrangian optimization techniques, which have been enhanced by means of heuristics that improve the reliability and quality of convergence. Preliminary versions of this approach can be found in [1]–[4]. Two versions of the QoS constraint are considered. In the first, we require that blocking probability satisfy this constraint on all circuits; in the second, we require only that the average blocking probability in the network satisfy this constraint. We demonstrate that when the constraint is relaxed to its average form, convergence is considerably faster and the true optimal solution is reached more rapidly and reliably than in examples in which each circuit must satisfy the constraint.

Manuscript received September 14, 2001; approved for publication by Erik A. van Doorn, Division III editor, August 13, 2002.

J. E. Wieselthier and G. D. Nguyen are with the Information Technology Division, Naval Research Laboratory, Washington, DC 20375, USA, e-mail: {wieselthier, nguyen}@itd.nrl.navy.mil.

A. Ephremides is with the Electrical & Computer Eng. Dept. and Institute for Systems Research, University of Maryland College Park, MD 20742, USA, e-mail: tony@eng.umd.edu.

This work was supported by the Office of Naval Research.

[1] The notion of network capacity referred to here has nothing to do with Shannon channel capacity concepts.

We have observed that a wide range of load profiles can typically provide nearly optimal performance; thus, acceptable performance can generally be obtained even when the offered loads are not optimal. Additionally, we address the fairness considerations associated with guaranteeing load levels to particular services, and we show that such service guarantees often result in only modest reductions in overall throughput.

The primary contributions of this paper are the novel formulation of the network "sizing" problem for session-based traffic, which has not previously been addressed in the literature, and the development of a heuristic that facilitates its solution. We show the degree of improvement that can be achieved, as compared to uniform offered loads, and we develop insight into several aspects of network operation including: the dependence of performance on offered loads, the interaction among traffic on different multihop circuits, and the differing impacts of blocking probability constraints that apply to each individual circuit versus the use of a single constraint on average blocking probability.

Our algorithm can serve as the basis for a design tool for network optimization. We demonstrate its robustness over a wide variety of network and algorithm parameters, and show that an easily applied stopping rule can be implemented, thereby providing reliable convergence to nearly optimal solutions. Although all of our examples represent the case of wireless networks, our formulation is also applicable to wired networks.

In Section II, we present our basic network model and define the optimization problem. In Section III, we discuss the projection heuristic that we have developed for this problem. In Section IV, we present the results of numerous test cases, which verify the robustness and effectiveness of the projection heuristic. In Section V, we quantify the improvement achieved by our approach, as compared to the case of uniform offered loads. In Section VI, we demonstrate the effect of admission-control thresholds on network performance. In Section VII, we define the concept of "underloaded circuits," and illustrate their impact on network performance; additionally, we address fairness issues. Thus far in the paper, all discussion addressed the case in which the constraint on blocking probability must be satisfied on each circuit. In Section VIII, we introduce an alternative form of the constraint in which only the average blocking probability (over all sessions and all circuits) must satisfy such a constraint. We show that overall performance is improved and that convergence is much faster; the projection heuristic is not needed in this case because only a single constraint on blocking probability needs to be satisfied. Finally, in Section IX, we present our conclusions from this research.

II. THE NETWORK MODEL AND OPTIMIZATION PROBLEM

We consider a network with fixed topology and fixed routes between each pair of source and destination nodes throughout the duration of each accepted session (e.g., voice call). Network topology can be described in terms of the communication resources available at each node and the connectivities between nodes. We assume a "blocked calls cleared" mode of operation, i.e., unless sessions are accepted for immediate transmission, they are "blocked" and lost from the system. Appropriate performance measures for this mode of operation include blocking probability and throughput.

Our objective is to maximize network throughput, subject to constraints on blocking probability. Since the routes and admission-control policy are fixed a priori, we achieve our objective by choosing the offered loads on each of circuits. This is a useful problem because it permits us to determine the maximum throughput that can be achieved, subject to constraints on blocking probability, when routing and admission-control policy are fixed. The optimization problem is defined in Section II-A. First, we describe the network model.

We consider J source-destination pairs, each of which is assigned a fixed route (circuit) through the graph of the network that interconnects them. That is, we do not consider the problem of optimally choosing these routes, although we fully recognize the impact of this choice on throughput. We let x_j denote the number of sessions that are ongoing on circuit j, and we assume that each accepted session consumes a fixed amount of resource throughout its duration, i.e., a fixed unit of bandwidth is required over each link in the circuit to support each session. Thus (using the terminology associated with broadband networks), we consider the case of CBR traffic sources. The state of the system is the J-dimensional vector $\mathbf{x} = (x_1, \cdots, x_J)$.

Our model can accommodate either wired or wireless networks. The capacity of network element (link or node) i is denoted by T_i. In the wired case, T_i is the number of channels supported by link $i, i = 1, \cdots, M$, where M is the number of links in the network. In the wireless case, T_i is the number of transceivers at node i, and M is the number of nodes in the network. Our wireless network model corresponds to the case of "ad hoc" (or, equivalently, "flat" or "infrastructureless," rather than cellular) networks. We assume that each node has several transceivers, and that each session requires the use of one transceiver at every node in its route; FDMA can then be conveniently assumed for channel access, provided that there is sufficient bandwidth for all transceivers to operate simultaneously at non-interfering frequencies. The state variables $x_1, x_2, \cdots, x_J$ satisfy sets of linear constraints of the form.

$$\sum_{j \in I_i} x_j \leq T_i, \quad i =, 1 \cdots, M, \tag{1}$$

where I_i is the set of circuits that share network element i.

We assume Poisson arrival statistics, and denote the offered load vector as $\lambda = (\lambda_1, \lambda_2, \cdots, \lambda_J)$, where λ_i is the arrival rate to circuit j. The expected call duration on circuit j is $1/\mu_j$, and the corresponding load on circuit j is $\rho_j = \lambda_j/\mu_j$. Admission control is implemented on a source-destination basis (i.e., not simply at the link level). We assume the use of "threshold-based" admission-control policies [5], in which calls are accepted as long as $x_j < X_j$, (where the X_j's are the admission-control parameters), and provided, of course, that at least one transceiver is available (i.e., not currently being used by another session) at each node along the route. Such threshold-based policies are a subset of the class of coordinate-convex policies [6]. Under the assumptions of Poisson arrivals and coordinate-convex admission-control policies (and for any distribution of call duration), the distribution of the system state has the prod-

uct form (see e.g., [7]). The admission-control policy is defined by the specification of the admissible state space Ω, which in our problem corresponds to specifying the X_j threshold values. The resulting stationary state distribution is then:

$$\pi_\Omega(\mathbf{x}) = \pi_\Omega(0) \prod_{j=1}^{J} \frac{\rho_j^{x_j}}{x_j!}, \tag{2}$$

where $\pi_\Omega(0)$ is the normalization constant given by

$$\pi_\Omega(0) = \left\{ \sum_{\mathbf{x}\in\Omega} \prod_{j=1}^{J} \frac{\rho_j^{x_j}}{x_j!} \right\}^{-1}. \tag{3}$$

For any state space Ω, it is straightforward (although time consuming) to evaluate $\pi_\Omega(0)$, which in turn permits the evaluation of performance measures such as throughput and blocking probability.

The use of threshold-based policies for this problem is reasonable, not only because of their simple implementation and their mathematical tractability by means of the product form[2], but also because they provide nearly optimal solutions [5]. In this paper we fix the admission-control policy Ω and maximize the throughput over all input rates that do not violate the QoS constraints.

A. The Optimization Problem

Our goal is to find the offered load vector λ that maximizes the total network throughput $S = S(\lambda)$ for a fixed threshold-based admission-control policy, subject to a circuit blocking probability $P_j(\lambda) \leq Q_j = QoS$ constraint for circuit j[3]. Note that such a vector λ is an extremal point of the Erlang capacity region for the network [8]. The equilibrium state distribution, and therefore throughput and blocking probability as well, are determined by means of the product-form solution discussed above.

The total network throughput S is the sum of the throughputs on each of the J circuits, i.e.,

$$S = S(\lambda) = \sum_{j=1}^{J} S_j(\lambda), \tag{4}$$

where

$$S_j(\lambda) = \lambda_j(1 - P_j(\lambda)) \tag{5}$$

is the throughput of circuit j, and $P_j(\lambda)$ is the probability that an incoming call to circuit j is blocked. The circuit blocking probabilities $P_j(\lambda)$, and hence the throughput $S(\lambda)$ as well, are obtained by means of the product-form solution associated with the particular network (i.e., the topology, number of transceivers at the nodes, and choice of circuits), the admission-control policy, and the offered load vector λ.

The introduction of constraints of the form $P_j(\lambda) \leq Q_j$ complicates the solution process considerably. To facilitate the discussion, we introduce the following definitions.

Definitions:

- An offered load vector λ is *admissible* if the constraints on blocking probability are satisfied.
- The *admissible region* contains all offered load vectors that are admissible.
- Corresponding to each admissible vector λ is a value of *admissible throughput.*

Constrained Optimization Problem:

Our goal is to choose the offered load vector λ to maximize the admissible throughput. This problem can be formulated as

$$\max_\lambda \{S(\lambda)\}, \tag{6}$$

$$\text{subject to: } P_j(\lambda) \leq Q_j. \tag{7}$$

The mathematical formulation is that of a constrained optimization problem with nonlinear objective and constraint functions of the variable λ. This is, in general, a difficult problem, for which the available mathematical theory provides the basic principles for solution, but no guarantee of convergence to the optimal point [9]. In this paper we choose to use the augmented Lagrangian approach as the method of solution, and we demonstrate its effectiveness and robustness when tested in a wide variety of network examples. From the theoretical point of view, this result constitutes already a contribution to nonlinear optimization theory. However, the focus of our paper is rather the application of this method to network design.

We first convert the constrained optimization problem to an unconstrained one by creating a new objective function that increases with $S(\lambda)$, but is decreased (penalized) when the QoS constraint is violated. For this new objective function, we use the augmented Lagrangian function

$$L(\lambda,\gamma) = S(\lambda) - \sum_{i=1}^{J} [\gamma_i \max\{0, P_i - Q_i\} + \tfrac{d}{2} (\max\{0, P_i - Q_i\})^2], \tag{8}$$

where $S(\lambda)$ is a function of the circuit blocking probabilities (see (4) and (5)), $d > 0$ and $\gamma = (\gamma_1, \gamma_2, \cdots, \gamma_J)$ is the Lagrange multiplier vector. Our goal is to maximize $L(\bullet)$ over λ. To do this, we use the iterative algorithm defined by

$$\lambda_j(k+1) = \max\left\{ \lambda_{\min}, \lambda_j(k) + \theta \frac{\partial L}{\partial \lambda_j} \right\}, \tag{9}$$

(for $j = 1, \cdots, J; k = 1, \cdots, k_{max}$) where the stepsize parameter θ and the Lagrange multipliers are updated using standard techniques (see [1], [9]).[4] The partial derivatives used in the update (9) are

$$\frac{\partial L}{\partial \lambda_i} = \frac{\partial S}{\partial \lambda_i} - \sum_{j=1}^{J} \frac{\partial P_j}{\partial \lambda_i} [\gamma_j + d(P_j - Q_j)] 1(P_j > Q_j), \tag{10}$$

[2] If the product-form solution did not apply, it would have been necessary to solve the balance equations to determine the equilibrium distribution.

[3] In Section VIII, we consider the use of an alternative QoS constraint, under which the average circuit blocking probability (weighted by the offered load to each circuit) must not exceed a specified value.

[4] In most of our examples, we have set $\lambda_{min} = 0$. If fairness is also an issue, a non-zero value of λ_{min} can be used to guarantee a specified throughput level on each circuit. Doing so typically results in only modest decrease in throughput for moderate values of λ_{min}, as is demonstrated in Section VII.

where

$$\frac{\partial S}{\partial \lambda_i} = \sum_{j=1}^{J} \frac{\partial S_j}{\partial \lambda_i}. \tag{11}$$

Thus the search proceeds along the direction of the throughput gradient ∇S when the QoS constraints are not violated, and in a direction influenced by both the throughput and blocking probability gradients when the constraints are violated.

The product-form solution is used to calculate the equilibrium state occupancy distribution, from which we can obtain the circuit blocking probabilities $P_i(\lambda)$, the circuit throughput values $S_i(\lambda)$, and the partial derivatives $\partial P_j(\lambda)/\partial \lambda_i$ and $\partial S_j(\lambda)/\partial \lambda_i$. In [10], Jordan and Varaiya have shown that

$$\frac{\partial P_j(\lambda)}{\partial \lambda_i} = \left\{ \begin{array}{ll} -\frac{\mu_j}{\lambda_i \lambda_j}\mathrm{cov}(x_i, x_j), & i \neq j \\ \frac{\mu_i}{\lambda_i^2}\left(E\{x_i\} - \mathrm{var}(x_i)\right), & i = j \end{array} \right\} \text{and}$$
$$\frac{\partial S_j(\lambda)}{\partial \lambda_i} = \frac{\mu_j}{\lambda_i}\mathrm{cov}(x_i, x_j). \tag{12}$$

We refer to the straightforward application of the updating rule defined by (9) and (10) as the "basic search technique."

B. Complexity and Computation Time

The complexity of our algorithm is dominated by the computation of the normalization constant $\pi_\Omega(0)$, given in (3). This computation requires the summation of quantities that are defined over the entire state space Ω; thus, the size of Ω dominates the complexity of our algorithm. The number of states in Ω increases rapidly as the following increase: the number of circuits J, the number of transceivers T_i at each node, and the threshold values X_j imposed on each circuit. Additionally, the set of circuit paths determines which circuits pass through any specific node, and thus which circuits must share the finite number of transceivers at that node, as expressed by the constraint of (1); as the number of circuits sharing a node increases, the state space becomes more constrained and the size of Ω tends to decrease. Thus, the overall run time depends on many factors, and varies drastically from network to network. For example, when run on a 100 MHz workstation with $T_i = 6$ and $X_j = 4$, we have observed run times per iteration of the order of seconds for Network 3 (for which $J = 8$), and minutes for Network 2 (for which $J = 10$); these networks are shown in Fig. 4. Clearly, these calculations are computationally intensive (although computation time could be reduced significantly by using a faster machine or perhaps through an improved software implementation). Nevertheless, since our algorithm is intended for off-line execution, computational efficiency is not of crucial importance.

III. DEVELOPMENT OF THE "PROJECTION HEURISTIC"

We have tested numerous versions of our algorithm on a variety of networks, including the example of "Network 1" shown in Fig. 1. In this example, the network supports $J = 10$ circuits, whose paths are listed in the caption. The speed of convergence obtained by using the basic search technique of (9) and (10) was typically slow, and strongly dependent on the parameter values

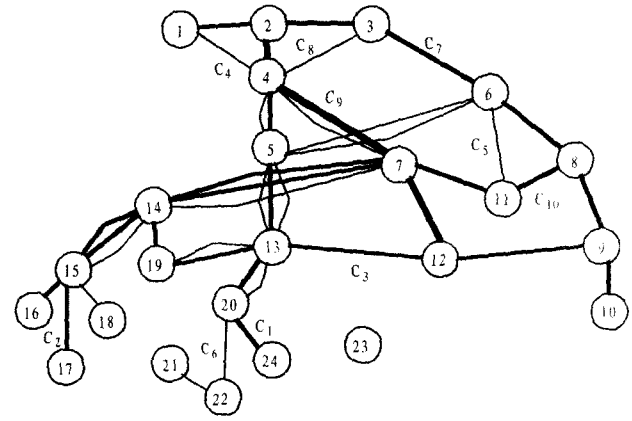


Fig. 1. "Network 1," a 24-node, 10-circuit network example.

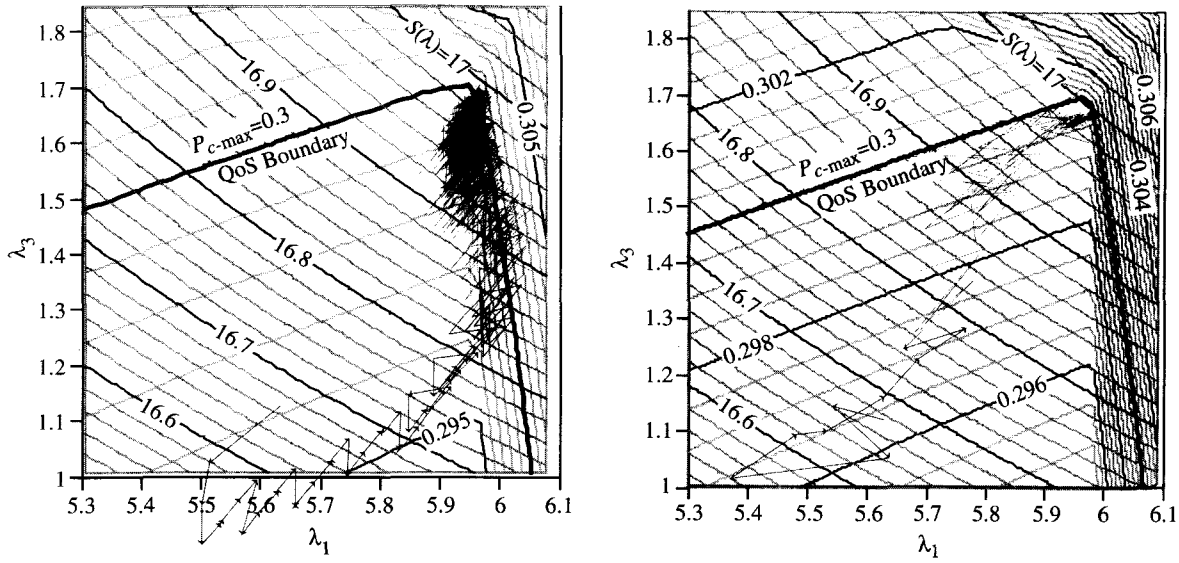
$C_1 = (4, 5, 13, 20, 24)$; $C_2 = (7, 14, 15, 17)$; $C_3 = (9, 12, 13, 19, 14, 15, 16)$; $C_4 = (1, 4, 5, 13, 19)$; $C_5 = (5, 6, 11)$; $C_6 = (21, 22, 20, 13, 5, 6)$; $C_7 = (1, 2, 3, 6, 8, 9, 10)$; $C_8 = (3, 4, 7, 14, 15, 18)$; $C_9 = (2, 4, 7, 12)$; $C_{10} = (14, 7, 11, 8)$.

used in the search. The most interesting, and troublesome, behavior occurred when the search trajectory passed near the QoS constraint contour. A typical example for the basic search technique, is shown in Fig. 2(a). In this figure a two-dimensional representation of the trajectory is shown, in which the values of the (λ_1, λ_3) pair are plotted as the search proceeds from its initial point to the final solution. All ten offered load values actually vary throughout the search; however, the contours of constant throughput $S(\lambda)$ and P_{c-max} (i.e., the largest blocking probability among the J circuits) shown in the figure are based on the values of λ_2 and $\lambda_4 - \lambda_{10}$ at the end of the search. Fig. 2(b) represents the use of the of the guided search technique, which is discussed in Section III-A.

The search trajectory shown in Fig. 2(a) is typical of results obtained using the basic search technique in that significant (although non-monotonic) progress is made in the early stage of the search, whereas considerably less-productive oscillatory behavior is observed as the search progresses. A common difficulty in constrained optimization problems arises because the optimum lies on the search boundary (i.e., one or more the circuit blocking probabilities is at the maximum-permitted QoS value). In such cases, typical gradient search techniques rely on damping of the stepsize θ to cause the search to slow and home in on the optimum. However, an overly rapid decrease in θ results in failure to reach the optimal solution; a less rapid decrease in θ can result in unacceptably slow convergence.

A. Guided Search Techniques

We have attempted to mitigate the oscillatory behavior of the basic search technique by using our knowledge of the throughput and blocking probability gradients to guide the search more efficiently. Ideally we would like the search to proceed along a direction of increasing throughput so that, at the same time, the blocking probabilities of circuits that are close to the QoS constraint do not increase. The basic approach to guiding the search was illustrated in [1], [2] by assuming that the blocking probability of a *single* circuit (circuit c) is close to the QoS constraint value. A heuristic was developed, based on a projection



(a) Basic search (b) Guided search

Fig. 2. Trajectory in λ_1 and λ_3 superimposed on throughput and P_{c-max} contours; λ_2, λ_4–λ_{10} are fixed at their final values.

operation in which the search is guided by removing the component of ∇S that is parallel to ∇P_c, the gradient of the blocking probability of circuit c. Fig. 2(b) shows a typical example of the trajectory of the guided search. Note that there is considerably less oscillation, and that a higher value of throughput is reached (the QoS constraint boundary now includes points that have an admissible throughput that exceeds $S(\lambda) = 17$).

The original version of the heuristic did not take into consideration the fact that, at a typical point in the search, it is common for *several* circuits to violate the QoS constraint or to be sufficiently close to the QoS boundary that the QoS constraint is in danger of being violated. We have observed behavior in which the chosen circuit for the projection alternates among two or three of the circuits, resulting in oscillatory behavior in which little progress is made toward the optimal solution. To mitigate this behavior, we have considered a generalized form of the projection operation in which several circuits are included in the projection, thereby taking into consideration the fact that *several* circuits are in danger of violating the QoS constraints, and should be discouraged from doing so.

B. The Generalized Projection Operation

To incorporate the QoS constraints associated with some or all of the circuits into the search-guiding mechanism, we introduce the quantity P_Σ, which is a function of the circuit-blocking probabilities $P_1, P_2, \cdots, P_J$. In this study we have used the following simple, linear form for P_Σ:

$$P_\Sigma = \frac{1}{N_\Sigma} \sum\nolimits_{i \in \Sigma} P_i, \tag{13}$$

where $\sum$ is a subset of $1, 2, \cdots, J$, and N_Σ is the number of circuits included in Σ. Thus P_Σ is the *average* blocking probability of the circuits included in Σ. The objective of the projection operation is then to increase S, while not increasing P_Σ. The desirable component of ∇S is a vector $\mathbf{D} = (D_1, D_2, \cdots, D_J)$,

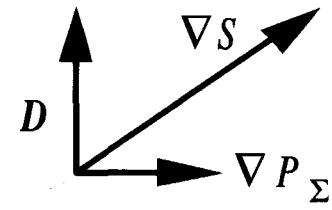


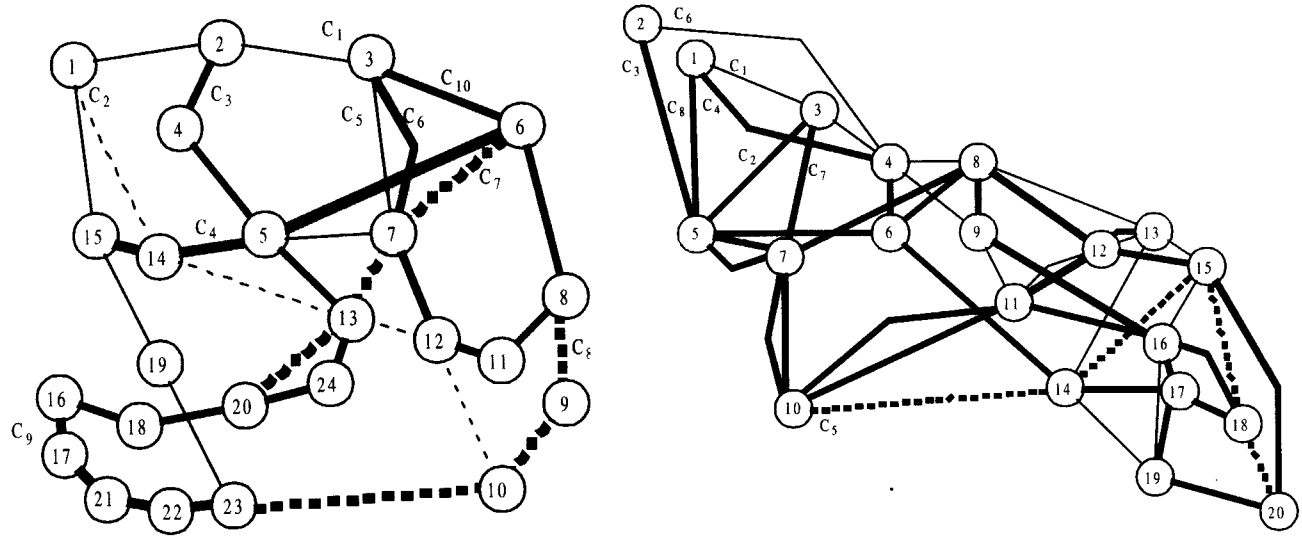
Fig. 3. $\mathbf{D}$ = component of ∇S that is orthogonal to ∇P_Σ.

which is obtained by projecting ∇S on the plane that is orthogonal to ∇P_Σ as illustrated in Fig. 3, and is mathematically given by:

$$\mathbf{D} = \begin{cases} \nabla S - \frac{\nabla S \cdot \nabla P_\Sigma}{\|\nabla P_\Sigma\|^2} \nabla P_\Sigma \\ \qquad \text{if } \|\nabla S - \frac{\nabla S \cdot \nabla P_\Sigma}{\|\nabla P_\Sigma\|^2} \nabla P_\Sigma\| > \tau \|\nabla S\|; \\ \nabla S \quad \text{otherwise,} \end{cases} \tag{14}$$

(10) is modified by replacing $\partial S / \partial \lambda_i$ by D_i. Therefore, λ is updated in a direction that tends to increase throughput without increasing the average blocking probability of the circuits in Σ. For further geometrical interpretation and the rationale behind of the projection formulation, see [1], [2].

The reason for using the projection operation only when it provides a sufficiently large value of $\|\mathbf{D}\|$ is based on our experimental observation that (in some cases) the trajectory can reach a point at which $\|\mathbf{D}\|$ is quite small. This behavior results in slow progress toward the optimal point, or even virtually total stopping of the trajectory, resulting in premature convergence; in fact, the trajectory can converge to a point interior to the admissible region (thus none of the circuit blocking probabilities are at the specified value, a condition not characteristic of the optimal point). This behavior is especially prevalent when the set Σ is large (e.g., we have considered cases in which Σ contains all J circuits). It occurs when the gradients of S and P_Σ are nearly parallel to each other. Turning off the projection oper-



a) Network 2 b) Network 3

Fig. 4. Networks 2 and 3.

Network 2: C_1 = (3, 2, 1, 15, 19, 23); C_2 = (1, 14, 13, 12, 10); C_3 = (2, 4, 5, 13, 24, 20, 18, 16); C_4 = (6, 5, 14, 15); C_5 = (3, 7, 5); C_6 = (3, 7, 12, 11, 8); C_7 = (6, 7, 13, 20); C_8 = (8, 9, 10, 23); C_9 = (16, 17, 21, 22, 23); C_{10} = (3, 6, 8)

Network 3: C_1 = (1, 3, 4, 8, 13, 14, 19); C_2 = (3, 5, 7, 10, 11, 16, 18); C_3 = (2, 5, 6, 8, 12, 15, 20); C_4 = (1, 4, 6, 14, 17, 19, 20); C_5 = (10, 14, 15, 18, 20); C_6 = (2, 4, 9, 11, 12, 13, 15, 16, 19); C_7 = (3, 7, 8, 9, 16, 17, 18); C_8 = (1, 5, 7, 10, 11, 12, 13)

ation (typically for just a single iteration) permits the trajectory to escape from such undesirable points. We have found that a value of $\tau = 0.1$ works well.

In this paper we consider a version of the projection rule, in which Σ is defined as follows:

$$\Sigma = \{i : P_i \geq P_{min} + \nu(P_{max} - P_{min})\}, \qquad (15)$$

where $P_{min} = \min\{P_i, i = 1, 2, \cdots, J\}$, $P_{max} = \max\{P_i, i = 1, 2, \cdots, J\}$, and $\nu \in [0, 1]$. The parameter ν can be chosen to include either few or many circuits, as desired. For example, for the networks discussed in this paper, the choice of $\nu = 0.2$ causes, on the average, about 8 (out of 10) circuits to be included in Σ (thus Σ is a large set). We also provide some results for the case in which only the "dominant circuit" (i.e., the circuit with the largest blocking probability) is included in the projection. Alternative versions of the projection rule are considered in [11].

We have observed that the use of a large set Σ tends to keep the trajectory well inside the admissible region during the early phase of the search, and discourages the trajectory from straying too far into the inadmissible region once the QoS-constraint boundary has been crossed.[5] However, although the neighborhood of the optimal point is reached rapidly, it is common for the trajectory to proceed past it, eventually converging to a point relatively far from the optimal. Apparently, the distortion introduced by the use the use of $\mathbf{D}$ rather than ∇S results in failure to converge to the true optimal point.

Based on these observations, which have been supported by extensive numerical results, we have concluded that it is often best to use a large set Σ during the early phase of the search, and then to turn off the projection term (i.e., set $\Sigma = \phi$, the empty set) at some point during the search. When the projection is turned off, the final approach to the optimal solution can be made without the presence of distortion.

[5] Although our original intent was to apply the projection operation only when the trajectory is close to the QoS-constraint boundary [1]–[3], we have found that it is often beneficial to apply it throughout the early stages of the search.

C. Other Algorithmic Considerations

The choice of stepsize (θ) damping rule is also critical to the performance of the algorithm, as it is in most iterative algorithms. The parameter θ must be large enough at the beginning of the search to make significant progress toward the region of the optimal solution. Typically, we have chosen the initial stepsize θ_0 on the basis of a short pilot run (that does not use the projection) so that, starting at $\lambda_j = 0$, the trajectory exits the admissible region for the first time after about five to ten iterations. Equally important is the rate of decrease of θ. Too fast a decrease can cause premature convergence of the algorithm, and hence failure to reach the optimal solution, whereas too slow a decrease can prevent convergence within the desired time constraints.

Care must also be taken in the choice of several other parameters used in the algorithm, in particular d, which weights the penalty term in (8). The use of $d = 50$ worked well for high values of QoS (e.g., ≥ 0.2), but not for more realistic values. The relatively poor performance for low values of QoS was observed because the gradient terms $\partial P_j / \partial \lambda_i$ were too small to drive the trajectory back into the admissible region at the low offered loads that are characteristic of low values of QoS. This problem has been mitigated by weighting the constraint-violation terms by ${Q_j}^{-0.5}$ as follows

Table 1. Offered load table for networks 1, 2, and 3 with $T_i = 6$, $X_j = 4$; $Q_j = 0.001$, and 0.3.

	λ_1 $\hat{P}_1$	λ_2 $\hat{P}_2$	λ_3 $\hat{P}_3$	λ_4 $\hat{P}_4$	λ_5 $\hat{P}_5$	λ_6 $\hat{P}_6$	λ_7 $\hat{P}_7$	λ_8 $\hat{P}_8$	λ_9 $\hat{P}_9$	λ_{10} $\hat{P}_{10}$	S
Network 1;											
$Q_j =$	0.2512	0.3230	0.3075	0.2321	0.3395	0.1603	0.3941	0.0229	0.3379	0.2987	2.6646
0.001	1.0000	1.0000	1.0000	1.0000	1.0000	0.9910	1.0000	0.9270	1.0000	1.0000	
$Q_j =$	3.3160	3.5230	1.9599	0.0009	2.0194	0.0008	3.3179	0.0005	1.9675	0.0036	
0.3	0.9997	0.9999	0.8975	0.8862	0.7656	0.9999	1.0000	1.0000	0.9011	0.7542	11.5380
Network 2;											
$Q_j =$	0.2326	0.3724	0.3129	0.3571	0.2460	0.2819	0.2941	0.3546	0.3856	0.2357	3.0700
0.001	1.0000	1.0000	0.9990	0.9990	1.0000	1.0000	0.9990	0.9990	1.0000	1.0000	
$Q_j =$	0.0002	3.2507	1.1397	2.0615	2.3653	1.3159	1.2968	2.5453	3.4367	1.7296	13.4129
0.3	0.9999	0.9999	0.9999	1.0000	0.9819	1.0000	0.9999	0.9999	0.9999	0.9989	
Network 3;											
$Q_j =$	0.2481	0.2529	0.2852	0.3242	0.2808	0.2777	0.3122	0.2646			2.2436
0.001	0.9990	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000			
$Q_j =$	1.1174	1.2697	1.5078	2.2527	1.6666	1.7254	2.1911	1.7158			9.4128
0.3	1.0000	0.9998	0.9999	1.0000	0.9999	1.0000	1.0000	0.9999			

upper entries: λ_j

lower entries: $\hat{P}_j$ (normalized blocking probability)

$$\frac{\partial L}{\partial \lambda_i} = D_i - \alpha \sum_{j=1}^{J} \frac{\partial P_j}{\partial \lambda_i} \left[\frac{\gamma_j + d(P_j - Q_j)}{\sqrt{Q_j}} \right] 1(P_j > Q_j), \tag{16}$$

where α is a "kick-up" factor that can be updated (increased from an initial value of 1) as necessary: e.g., α can be increased if too many consecutive inadmissible solutions are observed. We refer to (16) as the "projection algorithm." In the "no-projection algorithm" we use $\partial S/\partial \lambda_i$ instead of D_i; thus, it is similar to the basic search, but it incorporates the coefficients α and ${Q_j}^{-0.5}$.

IV. THROUGHPUT OPTIMIZATION EXAMPLES: UNIFORM QOS REQUIREMENTS

To validate our algorithm, and in particular to verify the robustness and effectiveness of our heuristics, we have chosen to run numerous test cases. It is assumed that the durations of calls on all circuits have the same mean; without further loss of generality, we set $\mu_j = 1$ $(1 \leq j \leq J)$. We have performed extensive testing of our algorithm on three networks, namely Network 1 of Fig. 1 and Networks 2 and 3 shown in Fig. 4. Network 2, like Network 1, supports ten circuits, whereas Network 3 supports eight. Our basic test consisted of the "core" runs, in which the 18 versions of the algorithm (which differ in their particular stepsize rules and projection rules, but which all have the same length of 1000 iterations–see [11] for full descriptions) were tested for these three networks and a variety of parameter values. The core runs are characterized by "uniform" parameter sets, as defined below. Results are presented in this paper for a blocking-probability constraint of QoS = 0.001; results for QoS = 0.3 are provided in [11], thus illustrating the effectiveness of our algorithm over a wide range of values of this parameter. In the core runs, all 18 versions of the algorithm provided nearly identical throughput, although there were significant differences in the speed of convergence. The results presented in this paper demonstrate the robustness of our algorithm (in its many versions) in a variety of network examples. The large body of additional results provided in [11] provides further evidence of its robustness and effectiveness.

In the uniform parameter sets used in the core runs, all nodes have the same number of transceivers ($T_1 = \cdots = T_N$), all circuits have the same threshold ($X_1 = \cdots = X_J$) on the permitted number of calls and all circuits must satisfy the same QoS constraint on blocking probability ($Q_1 = \cdots = Q_J$). In our discussion of particular network examples, the shorthand notation "$T_i = 6$" means $T_i = 6, i = 1, 2, \cdots, N$ (i.e., there are six transceivers at each node). Similarly, "$X_j = 4$" means $X_j = 4, j = 1, 2, \cdots, J$ (i.e., at most four calls are permitted on any circuit at any time), and "Q_j = 0.001" means that $Q_j = 0.001, j = 1, 2, \cdots, J$. The subscript i normally refers to node number, and the subscript j normally refers to circuit number.

Furthermore, there are no restrictions on the offered loads; thus, $\lambda_{min} = 0$ (which means that the offered load values are not prevented from decreasing to zero). Finally, the same rule is used to pick the initial stepsize value θ_0 in all of the core runs, namely that θ_0 is chosen so that the first inadmissible solution is obtained at the fifth iteration for the case in which the projection is not used (see Section III-C); the value of θ_0 is the same for all 18 versions of the algorithm for each network example (i.e., for a specified network topology, set of circuits, number of transceivers, admission-control policy, and QoS value), but different for different network examples. The versions have been evaluated based on their ability to find the optimal solution, as well as on their speed.

Our primary conclusion, based on extensive numerical evaluation, is that virtually all versions of the algorithm perform well, based on the criterion of providing optimal (or nearly optimal) throughput within 1000 iterations. The evolution of the offered load vector may vary greatly among the set of algorithms, but the final throughput value is typically close to the optimal value for all versions.

A. Blocking Probability

One characteristic property of the optimal solution in constrained optimization problems such as ours is that at least one of the circuit blocking probabilities must be at the maximum

permissible value of QoS. To measure how close the individual circuits approach this value, we introduce the *normalized circuit blocking probabilities*

$$\hat{P}_j = P_j/Q_j, \quad j = 1, \cdots, J. \tag{17}$$

Thus $\hat{P}_j = 1$ when $P_j = Q_j$.

Table 1, which we refer to as an "offered load" table, summarizes the solutions obtained for Networks 1, 2, and 3 with $T_i = 6, X_j = 4$, and $Q_j = 0.001$ and 0.3. For each of these six cases, we provide the solution that provided the best (i.e., highest admissible throughput) performance among the 18 versions of the algorithm. The entries shown are the offered load values (λ_j's), the corresponding normalized circuit blocking probabilities ($\hat{P}_j$'s) and the throughput (S) at the best point obtained for each run. The ten columns, with double entries in each cell, show the offered loads and normalized circuit blocking probabilities. For example, consider the top row, which shows the solution obtained for Network 1 with $Q_j = 0.001$. The cell in the $\{\lambda_1, \hat{P}_1\}$ column, with entries 0.2512 and 1.0000, indicates that $\lambda_1 = 0.2512$ and $P_1 = 1.0000$ at the best result for that run. The column at the far right shows the throughput achieved for that run, which is the same as the benchmark throughput in Table 6 in the APPENDIX.

With the exception of Network 1 with $Q_j = 0.3$, there is little difference in the offered loads found by the different versions of the algorithm (full details are provided in [11]). The fact that all of the circuit blocking probabilities are close to the maximum permitted value suggests strongly (although does not prove) that our solution is, indeed, close to the true optimal point. Furthermore, the fact that the solutions produced by the 18 versions of the algorithm are very similar to each other, despite the fact that the 18 versions produced very different trajectories, offers further support for this conclusion.

For $Q_j = 0.3$ there is a considerably wider variation in the solutions produced by different versions of the algorithm, both in terms of the offered loads (and the resulting normalized blocking probabilities) and in the throughput. Typically, the normalized blocking probability is 0.999 or greater on at most five of the ten circuits (although not always on the same set of five circuits); on the other circuits it is significantly lower than the maximum permitted QoS value (as low as 0.7542). This behavior is in marked contrast to that for $Q_j = 0.001$, in which all circuits were close to the maximum permitted QoS value.

The fact that not all blocking probabilities are near the specified QoS level when $Q_j = 0.3$ is not surprising. It is not a failure of the algorithm, but rather reflects the fact that the level of interaction among the circuits increases as offered load increases. Thus, there does not exist a set of offered load values for which all blocking probabilities are at the maximum permitted QoS value when that value is relatively high (e.g., 0.3). Thus, we see that in examples where there does not exist a solution for which the QoS value is reached on all circuits, the best solution may include some relatively extreme (i.e., small) values of normalized circuit blocking probability.

The optimal solutions for Networks 2 and 3 are quite different from that of Network 1 in the sense that all of the blocking probabilities are, in fact, extremely close to the QoS constraint value, even when $Q_j = 0.3$. In all cases for these two networks, the solutions produced by all versions of the algorithm (i.e., the offered load values) were virtually identical.

On the basis of these observations, it appears that whenever the optimal solution does, in fact, lie very close to the QoS contour in all dimensions, there is very little difference in the quality of the solutions produced by the various versions of the algorithm. Also, it appears that our algorithm is more robust in such cases; typically, fewer iterations are needed, and more-aggressive stepsize rules (resulting in faster attainment of milestones) are usually successful. Furthermore, we believe that one can have more confidence in the quality of the solution if the blocking probabilities are all close to the QoS constraint value.

B. Discussion

Based on the core runs, we are able to make a number of observations on both the sensitivity of the throughput to offered load and on the performance of the various versions of the algorithm that are designed to find that load.

We have observed that throughput, when the blocking probability on each circuit is constrained to the QoS value, is a relatively flat function of the offered load in the region of the optimal solution. Thus, rather large deviations in several of the λ_j values may be observed as the throughput increases from the 95% to the 98% and higher milestone values.

One of our principal conclusions is that all versions perform well in the sense of providing nearly optimal throughput. However, there are some differences in performance, both in terms of the speed of achieving "good" (although not necessarily optimal) solutions and in terms of reaching the optimal point. Although [11] discusses some examples in which some versions of the algorithm provided somewhat less than 98% of the benchmark throughput values, our basic conclusion remains valid, namely that all versions perform well when performed for a limited number of iterations (1000 in most of our examples).

V. THROUGHPUT GAIN ACHIEVED BY OPTIMIZATION OF OFFERED LOADS

From the perspective of the network user/manager, it is more important to determine the improvement obtained by means of the optimization of the offered loads than to evaluate the relatively subtle differences in performance among the algorithms. Thus, we compare the performance obtained by using our algorithms to that which is obtained under "uniform loading" (i.e., the case in which the λ_j's are all equal). In the latter case, the search for the optimal solution is a simple one-dimensional search, which constitutes a standard "quick-and-dirty" approach to network sizing.

A. Baseline Examples

To assess the benefits achievable through optimal loading, we first consider Networks 1, 2, and 3, again for the case of $T_i = 6$, and $X_j = 4$. Fig. 5 shows throughput in Network 1 as a function of QoS (i.e., of the maximum permitted blocking probability on any individual circuit) for both uniform and optimal loading. The "optimal loading" curve represents the higher throughput

Table 2. Throughput improvement achieved by optimization of offered load in network examples with different values of the QoS constraint Q_j ($T_i = 6$, and $X_j = 4$); case 1: $Q_1 = \cdots = Q_5 = 0.001, Q_6 = \cdots = Q_{10} = 0.3$; case 2: $Q_1 = \cdots = Q_5 = 0.3, Q_6 = \cdots = Q_{10} = 0.001$; case 1a: $Q_1 = \cdots = Q_4 = 0.001, Q_5 = \cdots = Q_8 = 0.3$; case 2a: $Q_1 = \cdots = Q_4 = 0.3, Q_5 = \cdots = Q_8 = 0.001$.

Network example	Network 1		Network 2		Network 3	
	case 1	case 2	case 1	case 2	case 1a	case 2a
Throughput (all λ_j's equal)	2.34	2.34	2.63	2.63	2.17	2.17
Throughput (2 equal groups)	2.45	2.36	2.70	2.78	2.17	2.25
Throughput (optimal load)	5.22	5.68	5.59	7.63	2.35	5.61
Improvement	123%	142%	112%	190%	8%	158%

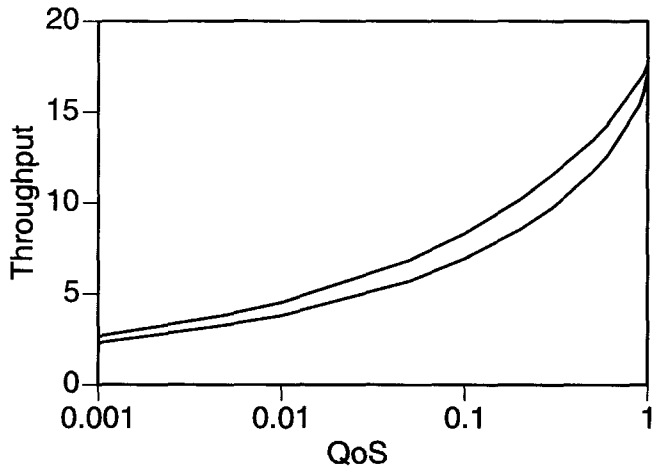


Fig. 5. Throughput achievable in Network 1 using uniform (lower curve) and optimal loading (upper curve).

obtained by our optimization algorithm, and demonstrates the degree of improvement that has been achieved by means of our multidimensional search. The percentage improvement is greatest at about $Q_j = 0.1$, where it is 19.6%. In [11] it is shown that similar improvement is achieved for Network 2. However, little improvement is obtained for Network 3, apparently because the offered loads at the optimal point are more nearly symmetrical than they were for the other two network examples.

In view of the highly asymmetrical nature of the optimal offered load (particularly for the case of Network 1), it is perhaps surprising that only "modest" increase in throughput was obtained by means of the optimization process. The inability to produce more substantial gains can be explained, at least in part, by the highly coupled nature of the circuits in the network. We can view this as a form of "balanced coupling," in which a relatively large region of the offered load space produces similar values of throughput.

The characteristics of these networks that cause the insensitivity to loading appear to include the following:

- interaction (i.e., contention for resources) among many circuits.
- equal QoS requirements for each circuit.
- equal number of transceivers at each node.

We now explore some network examples that do not possess all of these characteristics, and demonstrate the capability of our optimization techniques to provide significant increase in throughput

Table 3. Throughput improvement achieved by optimization of offered load in example with non-uniform number of transceivers; Network 1 ($T_i = 3, i = 1, 2, \cdots, 12$, and $T_i = 6, i = 13, 14, \cdots, 24$, and $X_j = 6$).

Q_1	0.001	0.3	0.9
Throughput (all λ_j's equal)	0.37	3.82	7.88
Throughput (optimal load)	0.51	4.92	8.52
Improvement	37.8%	28.8%	8.1%

B. Networks 1, 2, and 3 with Different QoS Values

We assume that the blocking probability of half of the circuits must satisfy a QoS constraint of 0.001 and half must satisfy 0.3. In all cases, $T_i = 6$ and $X_j = 4$. Two examples were considered for each case, as summarized in Table 2. These results are based on the use of the basic search technique; we have found that the use of the projection operation is not helpful in examples in which there is great disparity among the QoS constraint values.

The first row shows the maximum throughput that can be obtained when all of the λ_j's are constrained to be equal. We also considered an alternative form of loading in which all of the circuits in the QoS = 0.001 group are constrained to have the same offered load value (call it $\lambda_{0.001}$), and all those in the QoS = 0.3 group are constrained to have the same offered load value (call it $\lambda_{0.3}$). The throughput associated with the best values of $\lambda_{0.001}$ and $\lambda_{0.3}$ are shown in the second row. Little improvement over the case of uniform offered load is observed. Apparently, this is because even small increases of the offered load on some circuits in the QoS = 0.3 group (above the optimum uniform value) result in the violation of the QoS constraint on one or more circuits in the QoS = 0.001 group. The third row shows the throughput achieved using our optimization algorithm. The fourth row shows the improvement when the throughput under optimal loading is compared to that under uniform loading. The degree of improvement ranges from modest to dramatic, and depends strongly on network characteristics and on which circuits are in the two QoS groups.

C. Unequal Number of Transceivers

We examined the degree of improvement that can be achieved in networks in which the number of transceivers is not the same at all nodes. In particular, we considered Network 1 with

Table 4. Optimal throughput values for Network 1 with $T_i = 6$.

	$Q_j = 0.001$	$Q_j = 0.3$
$X_j = 3$	1.8391	11.2686
$X_j = 4$	2.6645	11.5380
$X_j = 6$	2.9095	11.6147

$T_i = 3, i = 1, 2, \cdots, 12$, and $T_i = 6, i = 13, 14, \cdots, 24$. We considered a system without admission control, i.e., one with $X_j = 6$. Table 3 shows the throughput achieved using uniform and optimal loading for this example, for $Q_j = 0.001, 0.3$, and 0.9. A significant degree of improvement is achieved, except for very high values of Q_j.

VI. THE EFFECT OF THE ADMISSION-CONTROL THRESHOLDS ON THE OPTIMIZATION OF OFFERED LOAD

We pointed out in Section I that network performance depends strongly on the highly interdependent issues of admission control, routing, offered load, and channel access. In this paper, we have focused on the optimization of offered load under fixed threshold-based admission control and fixed routing schemes. In this section we address the impact of adjusting the admission control thresholds on achievable throughput, as well as on the performance of our optimization algorithms.

We first address the impact of the imposition of threshold-based admission control (i.e., setting $X_j \leq T_i$) on achievable throughput. Table 4 shows the optimal throughput values for the case of Network 1 and $T_i = 6$ for $X_j = 3, 4$, and 6. When $Q_j = 0.3$, there is little difference in optimal throughput as X_j is varied from 3 to 6 (an increase of only 2.4% as it is increased from 3 to 4, and only 0.66% as it is increased from 4 to 6). However, the increase is much more substantial when $Q_j = 0.001$ (where the increases are 44.9% and 9.2%, respectively).

The case of $X_j = T_i$ corresponds to an "uncontrolled" network, in the sense that calls are admitted as long as a resources are available at every node along the route. This form of admission control is often referred to as "complete sharing." Such complete sharing provides increased opportunities to increase admissible throughput, since the region of admissible load vectors (λ) increases in size as the control threshold values X_j increase. A consequence of complete sharing is that the optimal offered load profile tends to be more asymmetrical as the thresholds X_j are increased.

VII. UNDERLOADED CIRCUITS AND FAIRNESS CONSIDERATIONS

In this section we observe that the presence of "underloaded circuits" (defined below) can reduce overall throughput even though they offer negligible load to the network. In addition, we address the issue of fairness, which can be implemented by guaranteeing each circuit at least a specified level of offered load.

In several of our examples, the optimal offered load vector contains one or more entries that are either zero or close to it. We refer to circuits whose offered load at the optimal point is less than 1% of the average offered load as "underloaded circuits." For example, consider Network 1 with $T_i = 6$, $X_j = 4$, $\lambda_{min} = 0$, and $Q_j = 0.3$. The optimal throughput value for this network is 11.538. It is interesting to note that this value was achieved when three of the offered load values (λ_4, λ_6, and λ_8) were less than 10^{-3}, while the average value of offered load on the other seven circuits was 2.3. To assess the impact of these underloaded circuits on overall network performance, the algorithm was re-run by setting $\lambda_4 = \lambda_6 = \lambda_8 = 0$, and ignoring the requirement that these underloaded circuits satisfy the QoS constraint.[6] The resulting optimal throughput value increased to 11.83. Thus, a modest (2.5%) increase in throughput was achieved by ignoring the requirement to satisfy the QoS constraint on circuits that make a negligible impact on the total network throughput. One way to view this situation is to say that the non-underloaded circuits are being penalized by the need to satisfy the QoS constraint on underloaded circuits. They are forced to reduce their offered load to accommodate the QoS requirements of unlikely events (the occurrence of arrivals on underloaded circuits).

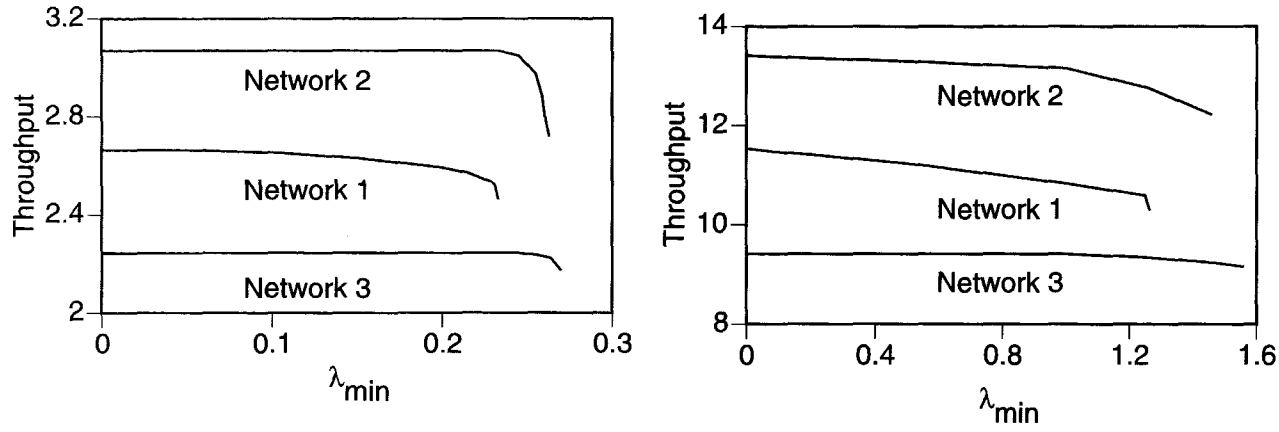
The issue of whether the QoS constraint should be guaranteed for all circuits, or for only those with significant offered loads, is a topic for future study. There are implications on pricing, e.g., whether the underloaded circuits should be required to pay more than non-underloaded circuits to receive QoS guarantees. Here, the requirement that the QoS constraint be satisfied by the underloaded circuits has a greater deleterious impact on the other circuits in the network than does the blocking caused by traffic on the underloaded circuits (which is negligible). Thus, trade-offs must be made between the often-conflicting goals of maximizing overall revenue and providing uniform QoS to all users.

A. *Fairness Considerations: A Guaranteed Offered Load on Each Circuit*

Another way to address the issue of underloaded circuits is to prevent their occurrence by guaranteeing a non-negligible offered load, λ_{min}, to all circuits. The goal here is to provide some degree of fairness to all users. Of course, to maintain satisfaction of the QoS constraint, some of the offered load values will have to be decreased, resulting in decreased overall throughput. We again consider the same network example of Network 1 with $T_i = 6$, $X_j = 4$, and $Q_j = 0.001$ and 0.3 (and return to our original requirement that the QoS constraint must be satisfied on all circuits, including the underloaded ones). Fig. 6 shows that the optimal admissible throughput decreases when λ_{min} increases, as is certainly expected.

For example, consider first the case of $Q_j = 0.001$. Beginning at about $\lambda_{min} = 0.1$, Network 1 at first experiences a gradual perceptible decrease in admissible throughput as λ_{min} is increased. However, at $\lambda_{min} = 0.23$ the decrease in throughput becomes precipitous, and the largest value of λ_{min} for which an admissible solution can be found is about 0.234; the curves stop at this point. This value of λ_{min} is equal to the value obtained by assuming that all of the λ_j are equal, and solving for the largest value of λ_j that provides an admissible solution. The relative

[6]The blocking probability of a circuit can be high (relative to the specified QoS constraint value) even when the offered load to it is zero. The blocking probability for circuit j is the probability that the network is in a state in which a call to circuit j would be blocked, should it arrive.



a) $Q_j = 0.001$ b) $Q_j = 0.3$

Fig. 6. Maximum throughput achievable on Networks 1, 2, and 3 when each circuit is guaranteed an offered load of at least λ_{min}.

insensitivity of Networks 2 and 3 to λ_{min} is a consequence of the more nearly symmetrical nature of these networks, as is evidenced by the more nearly equal values of the λ_j's at the optimal point.

Results are qualitatively similar when $Q_j = 0.3$, although the effects of λ_{min} are felt at values of λ_{min} that are a smaller fraction of the maximum value it can take on. Whether or not the decrease in overall throughput, for the sake of providing each of the users a guaranteed level of offered load, is an acceptable tradeoff is, of course, the decision of the network designer/manager.

VIII. AN ALTERNATIVE QOS CONSTRAINT: AVERAGE BLOCKING PROBABILITY

Instead of requiring that each circuit satisfy the QoS constraint on blocking probability, we may consider an alternative version of the QoS constraint in which we require only that the average blocking probability in the network must satisfy this constraint. Whether the QoS constraint should be applied to the average blocking probability or to each individual circuit is a decision to be made by the network designer/administrator. We demonstrate in this section that relaxing the QoS constraint in this manner results in not only higher throughput values, but also in considerably faster convergence. To distinguish these two forms of the QoS constraint, we henceforth refer to them as P_{av} and P_{max} constraints, respectively. In addition, we demonstrate that in some cases the use of a QoS constraint based on the average blocking probability can provide a good initial point for a search based on the satisfaction of the QoS constraint on each individual circuit.

A. Mathematical Model

The overall blocking probability P_{av} (i.e., the fraction of calls arriving to the network that are blocked, regardless of the circuit to which they arrive) is defined as the ratio of the expected number of blocked calls per unit time (summed over all circuits) to the expected total number of call arrivals per unit time:

$$P_{av} = \frac{1}{\Lambda}\sum_{j=1}^{J}\lambda_j P_j, \tag{18}$$

where $\Lambda = \lambda_1 + ... + \lambda_J$. Thus, some circuits may be permitted relatively large blocking probabilities provided that the overall blocking probability satisfies the single QoS constraint, i.e., that $P_{av} \leq Q$.

Following the approach of Section II, the augmented Lagrangian function can now be written as

$$L(\lambda,\gamma) = S(\lambda) - \gamma\max(0, P_{av} - Q) - \frac{d}{2}[\max(0, P_{av} - Q)]^2. \tag{19}$$

Note that this expression has the same form as (8), except that there is a single constraint on P_{av} rather than individual constraints on each of the J blocking probabilities P_j. The resulting partial derivatives are

$$\frac{\partial L}{\partial \lambda_i} = \frac{\partial S}{\partial \lambda_i} - [\frac{\gamma + d(P_{av} - Q)}{\sqrt{Q}}]\frac{\partial P_{av}}{\partial \lambda_i}1(P_{av} > Q), \tag{20}$$

where

$$\frac{\partial P_{av}}{\partial \lambda_i} = \frac{(\lambda_1 + ... + \lambda_J)[P_i + \sum_{j=1}^{J}\lambda_j\frac{\partial P_j}{\partial \lambda_i}] - \sum_{j=1}^{J}\lambda_j P_j}{(\lambda_1 + ... + \lambda_J)^2}. \tag{21}$$

The iterative algorithm proceeds similarly to the case in which the QoS constraint must be satisfied on each individual circuit.

In [11] we studied the use of the projection to guide the search in examples involving the P_{av} form of the QoS constraint. In this case, since only the average blocking probability P_{av} is of interest (and not the individual values of the P_j's), P_{av} is used instead of P_Σ in the definition of $\mathbf{D}$. The use of the projection made little difference in performance, in terms of either quality of solution or speed of convergence, for the P_{av} QoS criterion. The examples discussed here do not use the projection.

Table 5. Offered loads and normalized circuit blocking probabilities for Network 1 under both P_{av} and P_{max} forms of the QoS constraint on average blocking probability; $T_i = 6, X_j = 4$.

QoS	form of constr	λ_1 $\hat{P}_1$	λ_2 $\hat{P}_2$	λ_3 $\hat{P}_3$	λ_4 $\hat{P}_4$	λ_5 $\hat{P}_5$	λ_6 $\hat{P}_6$	λ_7 $\hat{P}_7$	λ_8 $\hat{P}_8$	λ_9 $\hat{P}_9$	λ_10 $\hat{P}_10$	S
0.001	Average	0.2557	0.3248	0.3126	0.2255	0.3530	0.1360	0.4087	0.0289	0.3385	0.2863	2.6674
		0.9530	0.9990	0.9850	0.9410	1.0260	0.9280	1.0990	0.9260	1.0070	0.9670	
	Max	0.2512	0.3230	0.3075	0.2321	0.3395	0.1603	0.3941	0.0229	0.3379	0.2987	2.6646
		1.0000	1.0000	1.0000	1.0000	1.0000	0.9910	1.0000	0.9270	1.0000	1.0000	
0.3	Average	2.7669	2.8347	2.0771	0.5338	2.5085	0.0000	3.2890	0.0000	2.0771	0.8449	11.8524
		1.0131	0.9921	1.0014	0.9823	0.9669	1.1590	1.0353	1.1484	1.0014	0.9486	
	Average	3.3160	3.5230	1.9599	0.0009	2.0194	0.0008	3.3179	0.0005	1.9675	0.0036	11.5380
		0.9997	0.9999	0.8975	0.8862	0.7656	0.9999	1.0000	1.0000	0.9011	0.7542	

column 2: form of constraint indicates whether the P_{av} or P_{max} case applies to the following columns
columns 3–12: λ_j shows offered load to circuit j at best solution, $\hat{P}_j$ shows normalized circuit blocking probability at best solution

B. Performance Results

We tested this formulation on Networks 1, 2, and 3, and obtained rapid convergence to optimal solutions when the stepsize was reduced exponentially to 10 percent of its initial value in 400 iterations. Such an aggressive stepsize rule did generally not perform well in the P_{max} case; it typically resulted in failure to reach the neighborhood of the optimal solution. In all of our examples the milestones are reached much more rapidly than they were in the P_{max} case, and the evolution of the offered loads is much smoother than in the P_{max} case. Both of these characteristics are a consequence of the need to satisfy only a single average QoS constraint, which permits the set of offered loads to trade off among themselves more easily than the case in which the QoS constraint must be satisfied on each individual circuit. Convergence properties of the P_{av} model are discussed briefly in the APPENDIX, and in more detail in [11].

We now compare the results obtained from the P_{max} and P_{av} models to see the impact of the form of the QoS constraint on throughput. Table 5 shows the optimal offered loads, the corresponding normalized blocking probabilities, and the throughput for the case of Network 1 with $T_i = 6$ and $X_j = 4$, for $Q = 0.001$ and 0.3. The results for the P_{max} case are based on the best run among the 18 versions of the algorithm (i.e., the one that provided the highest throughput).

For the case of $Q = 0.001$, the relaxation of the QoS constraint to the P_{av} form has had a negligible impact on the optimal offered load vector and the overall throughput. By contrast, for the case of $Q = 0.3$, the use of the average QoS constraint results in significant changes in some of the offered load values, as well as a 2.7% increase in throughput. For Networks 2 and 3 there was little difference between the P_{av} and P_{max} solutions for either QoS value [11].

C. Combined Use of Average and Individual QoS Constraints

The results of Table 5 indicate that the use of the average blocking probability P_{av} as the QoS constraint typically provides a solution that is "similar" to that obtained using the individual circuit blocking probability constraint P_{max}, especially for small QoS values (i.e., 0.001). Thus, it brings us close to the optimal solution of our original problem. This similarity has led us to consider the use of an alternative approach during the first phase of the algorithm; instead of using the projection in conjunction with QoS constraints on each individual circuit, we have considered using the P_{av} form of the QoS constraint (without use of the projection operation). We use the P_{av} constraint for 100 iterations, in conjunction with an aggressive stepsize rule in which the stepsize is reduced linearly to 5 percent of its initial value during this interval. Then, at the beginning of the second phase the QoS constraint is applied to each individual circuit as in the original formulation. Thus, the formulation based on the average blocking probability is used to determine an initial condition for the problem in which the QoS constraint must be satisfied on every circuit.

For the case of $Q_j = 0.3$, the aggressive stepsize rule that worked well for $Q_j = 0.001$ produced only 98% of the optimal throughput value. It was possible to reach the 99.5% milestone only when more-conservative stepsize rules were used; hence many more iterations were needed to reach comparable performance, and little difference was observed in the speed of the algorithm as compared to the basic search technique. The combined P_{av}/P_{max} algorithm was also used for Networks 2 and 3 for both values of the QoS constraint parameter. In these cases, the 99.5% milestones were reached very rapidly (by approximately iteration 150) when the stepsize rule discussed here was used.

For five of the six examples discussed here (i.e., two QoS values for each of the three networks), the performance obtained by using the combined P_{av}/P_{max} approach is comparable to, or better than (in terms of speed), that of the better algorithms that are based on the use of the projection in one or more phases. It appears that this approach would be most effective when the solution based on the P_{av} QoS criterion is most similar to that based on the P_{max} QoS criterion. Of course, it is not possible to determine with certainty that the solutions are similar without solving both problems. However, an examination of the circuit blocking probabilities shown in Table 5 may provide a helpful clue. In cases where the maximum normalized blocking probability found under the P_{av} constraint is only slightly greater than 1.0, little has to be done to coax the offered load values to a point that maximizes throughput while satisfying the P_{max} QoS criterion.

IX. CONCLUSIONS

In this paper, we have used a heuristic mix of sophisticated mathematical methods and intuition to develop a tool for the off-line determination of the offered load vector λ that provides

the maximum value of throughput, subject to QoS constraints on blocking probability, for a specified routing and admission-control policy. We have considered circuit-switched CBR service in both wireless and non-wireless networks, with the usual Markovian assumptions on the traffic characteristics. An easily implemented stopping rule provides convergence to nearly optimal solutions. Our approach provides a basis for "sizing" the network capability, i.e., for determining a benchmark level of "network capacity." If the original, user-specified, offered load profile does not achieve a level of throughput that is sufficiently close to this benchmark capacity, then pricing mechanisms can be introduced to steer the offered load profile in a direction that maximizes overall revenue. We have demonstrated that optimizing throughput over all circuit input rates often yields a significant improvement in achievable throughput, as compared to ad hoc techniques (such as uniform loading).

We have investigated two forms of the QoS constraint. In the first, the QoS constraint must be satisfied on each circuit; in the second, it is sufficient that the average blocking probability satisfy this constraint. We have found that, in the former case, it is often helpful to use the "projection" version of our algorithm, which provides faster and more reliable convergence to nearly optimal solutions, with little need to manually adjust parameters. Nevertheless, a high level of robustness was displayed by all versions of the algorithm (including those that do not use the projection heuristic) in that they all provided at least 98% of the optimal throughput in almost all examples. Relaxing the QoS constraint to its average form results in somewhat higher throughput (depending on the particular example), as well as faster and more reliable convergence (whether or not the projection heuristic is used).

The decision on which form of the QoS constraint is appropriate can be made by the network manager, based on the preference for individual vs social optimization. Economic issues can be incorporated by introducing a pricing structure that charges more for services that are provided to circuits that block an inordinate number of other circuits; alternatively, one can charge less for services for which less-stringent QoS levels are guaranteed. However, we leave a detailed consideration of this issue to future work.

It is interesting to note that, particularly for large values of QoS, the throughput is a rather flat function of the offered load in the sense that large changes in the offered load on several circuits yield only a small change in throughput. Therefore, when the convergence criterion is satisfied the throughput is close to its optimal value, even though the individual offered loads may be far from their optimal values. Consequently, it is often possible to incorporate fairness mechanisms that guarantee a specified level of throughput to each circuit, without significantly decreasing the overall throughput.

An obvious limitation of our algorithm (common to all solutions that focus on a single aspect of network control) is that, in its current form, it must be implemented for each routing scheme and for each admission-control policy. Future research is needed to develop a tool that jointly addresses the issues of offered load profile, routing, and admission control.

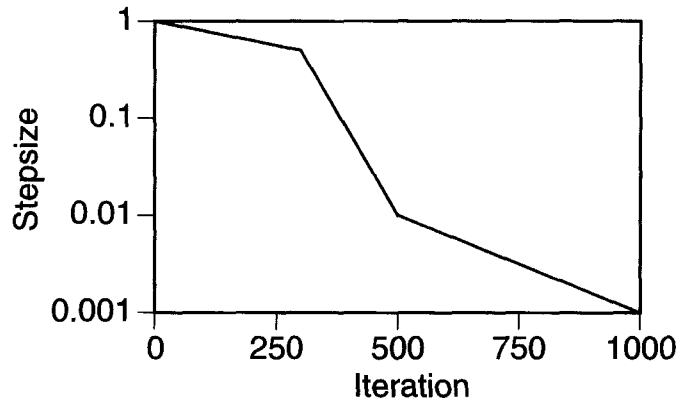


Fig. 7. Normalized stepsize as a function of iteration number - Stepsize Rule #3.

APPENDIX: CONVERGENCE PROPERTIES OF THE ALGORITHM

The focus of this paper has been on the "network capacity" and the offered load that achieves it, as well as on the properties of the solution. In this appendix we discuss some of the details of the various versions of the algorithm, as well as their convergence properties. Our primary focus is on the P_{max} case, for which 18 versions were studied. We also discuss briefly the P_{av} case.

The P_{max} Case

Table 6 provides a summary of the important "milestones" of each run for Network 1 with $T_i = 6$, $X_j = 4$, and $Q_j = 0.001$. We have included results for five versions of our algorithm, which make use of Stepsize Rule #3, which is shown in Fig. 7. The three "phases" are the three regions of constant slope (on the logarithmic scale). Additional stepsize rules, and the corresponding performance results, are discussed in [11].

The different versions of the algorithm make use of the following projection rules:

B = 0: no projection used at any time during the search;
B = 1: the projection based on only the "dominant" circuit (i.e., the circuit with the largest blocking probability) is used throughout the entire search;
B = 2: the projection with $\nu = 0.2$ is used during phase 1; no projection is used during phases 2 and 3;
B = 3: the projection with $\nu = 0.8$ is used during phase 1; no projection is used during phases 2 and 3;
B = 4: the projection with $\nu = 0.2$ is used during phase 1, $\nu = 0.8$ is used during phase 2, and no projection is used during phase 3.

In our evaluations, the initial values of the offered loads are zero for all circuits. The "benchmark throughput" value (listed in the figure caption) is the highest value of admissible throughput that was obtained for any of the 18 versions of the algorithm. Results for this network example but with $Q_j = 0.3$, as well as numerous other examples, are provided in [11]. We explain now the labels of the nine columns in Table 6.

1) algo: the version of the algorithm, denoted as A.B (e.g., 3.2, etc.), where A represents alternative "stepsize rules" and B represents alternative "projection rules";

Table 6. Milestone table for Network 1 with $T_i = 6, X_j = 4$, and $Q_j = 0.001$; Stepsize Rule #3 (benchmark throughput: 2.6646).

algo	adm thruput at first exit	best adm thruput	last adm thruput	95.0%	98.0%	99.0%	99.5%	99.9%
3.0	8 78.22%	975 99.99%	991 99.98%	62	177	382	382	472
3.1	14 92.89%	899 100.00%	998 99.99%	28	235	372	400	456
3.2	29 94.11%	981 99.99%	1000 99.98%	36	91	380	405	477
3.3	18 94.66%	990 99.99%	999 99.99%	41	292	332	408	476
3.4	29 94.11%	914 100.00%	995 99.98%	36	91	341	354	517

first column: version of algorithm in A.B notation
upper entries (columns 2–9): iteration at which milestone is reached
lower entries (columns 2–4): percentage of benchmark throughput at this iteration

2) adm thruput at first exit: the upper entry indicates the last iteration before the trajectory leaves the admissible region for the first time; the lower entry shows the percentage of the benchmark throughput value achieved at this iteration;
3) best adm thruput: the iteration at which the highest admissible throughput is found, and the percentage of benchmark throughput achieved at this iteration;
4) last adm thruput: the last iteration for which the solution is in the admissible region, and the percentage of benchmark throughput achieved at this iteration;
5) 95.0%: the first iteration at which the solution is admissible and equal to at least 95% of the benchmark throughput value;
6) - 9) 98.0%, etc.: first iteration at which the corresponding milestone is achieved.

We first consider $Q_j = 0.001$, as shown in Table 6. Virtually identical throughput values were obtained for all 18 versions of the algorithm, including the five summarized in this table as well as the additional versions discussed in [11]; in all 18 cases, the throughput was at least 99.9% of the benchmark value. For $Q_j = 0.3$ all 18 versions provided at least 99% of the benchmark throughput value, 15 versions provided at least 99.5%, and 13 versions provided at least 99.9%.

Such convergence to nearly optimal solutions is typical behavior over a wide variety of network scenarios, as is evident from the milestone tables for the nine other core runs, which are provided in [11]. In fact, for nine of the eleven network examples in the core runs, all 18 versions of the algorithm were able to provide at least 99.9% of the optimal throughput value; for ten of the eleven network examples all 18 versions were able to provide at least 99.5% of the optimal throughput value; and in all eleven network examples, all 18 versions were able to provide at least 99% of the optimal throughput value.

The purpose of the "adm thruput at first exit" column is to indicate the percentage of the benchmark throughput that is achieved prior to exiting the admissible region for the first time. This can be viewed as a measure of how well the algorithm behaves in the early stages in the sense of the directness of the approach to the optimal solution. Generally, the versions of the algorithm that do not use the projection (i.e., A.0) exit the admissible region sooner than the versions that do use the projection when the same stepsize is used. One reason for this behavior is that, since $\| \mathbf{D} \| \leq \| \nabla S \|$, the offered loads (and hence the resulting throughput) typically change by a smaller amount at each iteration than for examples in which the projection is not used. Another (related) reason is that, when the projection is not used, the trajectory is not affected by the QoS constraint until the admissible region is exited for the first time.

Moreover, in most cases the percentage of the benchmark throughput that is achieved just prior to exiting the admissible region for the first time is generally considerably higher for the versions that use the projection with a large number of circuits. For example, when $Q_j = 0.3$, the A.2 and A.4 versions of the algorithm provide about 96% of the benchmark throughput prior to exiting the admissible region for the first time. By contrast, the A.1 and A.3 versions provide only about 85% to 88% of the benchmark throughput. It thus appears that the versions of the algorithm that use a large number of circuits in the projection provide a more-direct ascent to an early "good" solution (in which the offered loads still may be far from the optimal solution). The results for $Q_j = 0.001$ shown in Table 6 are not as conclusive, although it is clear that the A.0 versions provide the lowest percentage of benchmark throughput prior to the first exit of the admissible region.

Our primary conclusion is that virtually all versions of the algorithm perform well, based on the criterion of providing optimal (or nearly optimal) throughput within 1000 iterations. The evolution of the offered load vector may vary greatly among the set of algorithms, but the final throughput value is typically close to the optimal value for all versions.

The Stopping-rule

As observed in Section I, the objective addressed in this paper is the determination of the load profile that maximizes throughput, subject to QoS constraints on blocking probability. Thus, for a given set of source-destination pairs, a given routing scheme, and a given admission-control policy, one can view our algorithm as a "black box" that determines the optimal load profile and the corresponding "network capacity." Once the optimal load profile and throughput have been determined, it is straightforward to determine whether any user-specified load profile provides performance that is sufficiently close to the optimal. Pricing mechanisms can then be introduced that encourage the users to steer their loads toward the neighborhood of the op-

Table 7. Stopping-Rule table for Network 1 with $T_i = 6, X_j = 4$, and $Q_j = 0.001$; Stepsize Rule #3 (benchmark throughput: 2.6646.)

max it	300			600			1000		
δ	0.1	0.01	0.001	0.1	0.01	0.001	0.01	0.001	0.0001
3.0	54	300	300	54	372	571	372	571	997
	93.612	98.099	98.099	93.612	98.374	99.966	98.374	99.966	99.990
3.1	15	300	300	15	380	498	380	498	908
	92.890	98.380	98.380	92.890	99.454	99.958	99.454	99.958	99.997
3.2	14	143	300	14	143	524	143	524	913
	56.247	98.237	98.434	56.247	98.237	99.949	98.237	99.949	99.991
3.3	15	300	300	15	389	510	389	510	1000
	79.320	98.501	98.501	79.320	99.208	99.961	99.208	99.961	99.992
3.4	14	143	300	14	143	407	143	407	441
	56.247	98.237	98.434	56.247	98.237	99.772	98.237	99.772	99.862

max it = maximum number of iterations (= 300, 600, or 1000)

δ = tolerance level used in stopping rule (= 0.1, 0.01, 0.001, or 0.0001)

upper entries: point at which iteration is stopped (either because convergence criterion is satisfied, or because maximum number of iterations has been reached)

lower entries: percentage of benchmark throughput at stopping point

timal solution. It is in this sense that we view our algorithm as a design tool.

The practical use of the algorithm as such a tool requires the specification of a stopping rule. The milestone tables (e.g., Table 6) are based on running of the different versions of the algorithm for a fixed number of iterations, in this case 1000. However, in most applications, iterative algorithms such as those developed in this paper would be terminated when a suitable convergence criterion is satisfied. The use of appropriate stopping rules can save considerable computational time if convergence is obtained early; on the other hand, failure to converge indicates that additional iterations are needed. The effectiveness of algorithms with particular stopping rules can be evaluated in terms of speed (number of iterations until convergence is declared) and quality of the resulting solution (in our case, percentage of the benchmark throughput that is obtained when the algorithm is stopped).

To evaluate the stopping rules, we did not perform additional runs, but instead examined the data from the runs with duration equal to 1000 iterations, and determined the outcome that would have occurred had these stopping rules been used. The convergence criterion used in our studies is

$$\frac{|S_{k+1} - S_k|}{\max\{S_k, S_{k+1}\}} < \delta, \quad k = m, m+1, ..., m+4 \quad \text{for some } m, \tag{22}$$

i.e., that five consecutive throughput values (whether or not they are admissible) should not differ from the previous throughput value by more than a specified fraction, which we denote by δ. Table 7 shows the effect of using different stopping rules for the determination of convergence for the same example for which the milestone table was just discussed, i.e., for $T_i = 6$, $X_j = 4$, $Q_j = 0.001$, and for Stepsize Rule #3. For example, three columns are headed by the number 300, and show the effect of stopping the run when convergence to the specified tolerance (δ = 0.1, 0.01, or 0.001) has been achieved, or at iteration 300 if convergence to the specified tolerance is not achieved.

When the search is stopped (either because the convergence criterion is satisfied, or because the specified number of iterations has been reached), the solution to the problem is declared to be the set of offered loads that has provided the highest admissible throughput thus far during the course of the run; the best solution is not necessarily the offered load at the stopping point because of the nonmonotonic nature of the algorithm. Specifically, consider the case of Version 3.0 in Table 7. The entry of 54 in the column headed by 300 and 0.1 indicates that convergence to within a tolerance of δ = 0.1 is obtained at iteration number 54. The entry of 93.612 under the 54 indicates that use of this stopping criterion provides 93.612% of the benchmark throughput (2.6646), i.e., the best admissible throughput observed thus far (not necessarily at this iteration–it may have occurred earlier) is 93.612% of the benchmark throughput. Similarly the entries in the next column indicate that convergence to within 0.01 is not obtained during the first 300 iterations, and that the best solution in the first 300 iterations provides 98.099% of the benchmark throughput; the criterion of δ = 0.01 is satisfied at iteration 372, at which point 98.374% of the benchmark throughput is obtained. The criterion of δ = 0.001 is satisfied at iteration 571.

It is impossible to anticipate the quality of the solution that will be determined by a particular level of convergence. For the network example of Table 7, when comparing the five versions of the algorithm that use Stepsize Rule #3, we see that a convergence criterion of δ = 0.1 (see the column for 600 iterations) provides a solution with throughput that is somewhere between 56% and 93.6% of the benchmark value. A criterion of δ = 0.01 (see the column for 1000 iterations) provides a solution between 98.2% and 99.5% of the benchmark value. Finally, a convergence criterion of δ = 0.001 (see the column for 1000 iterations) provides a solution that is better than 99.86% of the benchmark value in all cases. These numbers vary significantly for the different network examples we have considered, but are similar for the various stepsize rules we have studied.

Several factors affect the rate of convergence. For example, small stepsizes result in small changes in the offered load (and hence throughput) at each iteration; thus a rule that reduces stepsize rapidly would result in rapid satisfaction of the convergence condition, although not necessarily convergence to the optimal point. Also, we have observed that use of the projection operation with a large number of circuits tends to result in a smoother trajectory (not only a more-direct path, but also a smaller "effective stepsize" for a given θ than the use of no projection), and thus the increased possibility of declaring convergence prematurely.

The P_{av} Case:

Convergence to the 98% and higher milestones was usually

much faster for the P_{av} version of the constraint than for the P_{max} version. For example, the 99.5% milestone was typically achieved within the first 50 iterations, and the 99.9% milestone was typically achieved in about 100 to 150 iterations. Furthermore, the trajectories of admissible throughput and of the offered loads (the λ_j's) are considerably smoother in the P_{av} case. Although the admissible throughput doesn't necessarily increase monotonically, the decreases observed before convergence is achieved are much smaller than those observed for the P_{max} case. The faster convergence and smoother trajectories are a consequence of the need to satisfy only one constraint on overall blocking probability, rather than one for each individual circuit. Further details are provided in [11].

ACKNOWLEDGMENT

The authors acknowledge the contribution of Craig M. Barnhart, formerly of the Information Technology Division, Naval Research Laboratory, to the development of the algorithm proposed in this study. Mr. Barnhart is currently with TRW Data Technologies Division, Aurora, CO.

REFERENCES

[1] C. M. Barnhart, J. E. Wieselthier, and A. Ephremides, "Constrained optimization methods for admission control and offered load in communication networks," in *Proc. 30th Annual Conf. Inform. Sciences and Systems (CISS)*, Princeton University, Princeton, NJ, Mar. 1996, pp. 686–691.

[2] J. E. Wieselthier, *et al.*, "A problem of constrained optimization for bandwidth allocation in high-speed and wireless communication networks," in *Proc. 35th IEEE Conf. Decision and Control*, Kobe, Japan, Dec. 11-13, 1996, pp. 1347–1348.

[3] G. D. Nguyen, J. E. Wieselthier, and A. Ephremides, "Lagrangian techniques for optimizing throughput in wireless communication networks subject to QoS constraints," in *Proc. 31st Conf. Inform. Sciences and Systems (CISS)*, Johns Hopkins University, Baltimore, MD, Mar. 1997, pp. 405–410.

[4] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, "Algorithms for finding optimal offered load in wireless communication networks," in *Proc. IEEE MILCOM'97*, Monterey, CA, Nov. 1997, pp. 1570–1574

[5] C. M. Barnhart, J. E. Wieselthier, and A. Ephremides, "Admission-control policies for multihop wireless networks," Wireless Networks, 1–4, pp. 373–387, Dec. 1995.

[6] J. M. Aein, "A multi-user-class, blocked-calls-cleared, demand access model," *IEEE Trans. Commun.*, COM-26-3, pp. 378–385, Mar. 1978.

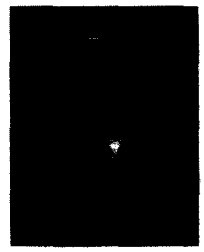
[7] K. W. Ross, *Multiservice Loss Models for Broadband Telecommunication Networks*, London: Springer-Verlag, 1995.

[8] D. Mitra and J. A. Morrison, "Erlang capacity and uniform approximations for shared unbuffered resources," *IEEE/ACM Trans. Networking*, 2-6, pp. 558-570, Dec. 1994.

[9] D. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*, Belmont, MA: Athena Scientific (originally published by Academic Press in 1982), 1996.

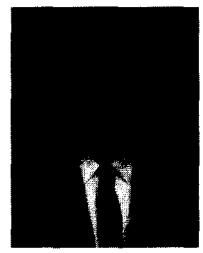
[10] S. Jordan and P. Varaiya, "Throughput in multiple service, multiple resource communication networks," *IEEE Trans. Commun.*, 39-8, pp. 1216–1222, Aug. 1991.

[11] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, "Throughput maximization under quality of service constraints," NRL Formal Report 5520–00-9922, Naval Research Laboratory, June 2000.



Jeffrey E. Wieselthier was born in Brooklyn, NY, on March 14, 1949. He received the S.B. degree from the Massachusetts Institute of Technology, Cambridge, in 1969, the M.S. degree from the Johns Hopkins University, Baltimore, MD, in 1971, and the Ph.D. degree from the University of Maryland, College Park, in 1979, all in electrical engineering. He was employed at the Naval Surface Warfare Center, White Oak, Silver Spring, MD, from 1969 to 1979. Since 1979 he has been with the Information Technology Division of the Naval Research Laboratory, Washington, DC. He was Technical Program Co-Chair of the Third IEEE Symposium on Computers and Communications in Athens, Greece, in 1998 and Treasurer of the 1991 IEEE International Symposium on Information Theory in Budapest, Hungary. He won the IEEE Fred W. Ellersick Award for the best unclassified paper at MILCOM 2000. He has studied a variety of communication networking problems, including multiple access and routing in spread-spectrum networks and the use of neural networks and other approaches for network performance evaluation, optimization and control. His current interests include wireless communication networks, with an emphasis on issues relating to energy-efficient operation. Dr. Wieselthier is a member of Eta Kappa Nu and Sigma Xi, and is a Senior Member of IEEE

Gam D. Nguyen received the Ph.D. in electrical engineering from the University of Maryland, College Park, MD in 1990. He has been at the Naval Research Laboratory, Washington, DC, since 1991. He won the IEEE Fred W. Ellersick Award for the best unclassified paper at MILCOM 2000. His research interests include communication systems, computer communication networks, and information processing.



Anthony Ephremides received his B.S. degree from the National Technical University of Athens (1967), and M.S. (1969) and Ph.D. (1971) degrees from Princeton University, all in Electrical Engineering. He has been at the University of Maryland since 1971, and currently holds a joint appointment as Professor in the Electrical and Computer Engineering Department and the Institute for Systems Research (ISR). He is co-founder of the NASA Center for Commercial Development of Space on Hybrid and Satellite Communications Networks established in 1991 at Maryland as an off-shoot of the ISR. He was a Visiting Professor in 1978 at the National Technical University in Athens, Greece, and in 1979 at the EECS Department of the University of California, Berkeley, and at INRIA, France. During 1985 - 1986 he was on leave at MIT and ETH in Zurich, Switzerland. He was the General Chairman of the 1986 IEEE Conference on Decision and Control in Athens, Greece and of the 1991 IEEE International Symposium on Information Theory in Budapest, Hungary. He was the Technical Program Co-Chair of the IEEE INFOCOM in New York City in 1999 and of the IEEE International Symposium on Information theory in Sorrento, Italy in 2000. He has also been the Director of the Fairchild Scholars and Doctoral Fellows Program, an academic and research partnership program in Satellite Communications between Fairchild Industries and the University of Maryland. He has received several awards, including the IEEE Donald E. Fink Prize Paper Award (1992) and the Sigmobile Award of the ACM for contributions to wireless communications (1997). He has been the President of the Information Theory Society of the IEEE (1987), and has served on the Board of the IEEE (1989 and 1990). He is a Fellow of IEEE. Dr. Ephremides' interests are in the areas of communication theory, communication systems and networks, queueing systems, signal processing, and satellite communications.