Adaptive Call Admission Control in Circuit-Switched Networks

Kagan Gokbayrak and Christos G. Cassandras, *Fellow, IEEE*

***Abstract*—We consider threshold-based admission control policies for traffic in fixed-route circuit-switched networks, and develop a scheme for adjusting the threshold parameters *online* so that, as operating conditions in the network change, the thresholds "adapt" with the objective of minimizing a weighted sum of call blocking probabilities. Instrumental in this scheme is an algorithm for estimating *online* the sensitivity of the call blocking metric with respect to thresholds. The formal optimization problem over the set of discrete threshold parameters is solved by means of a conversion to an optimization problem over a set of auxiliary real-valued parameters. Such threshold-based policies, though conservative at low traffic rates, have the advantage of being simple to implement, distributed in nature, adaptive, and not requiring explicit distributional modeling assumptions. Numerical results included in the paper indicate that at higher traffic rates these simple policies yield the same performance as more complex and less flexible call admission schemes.**

***Index Terms*—Call admission control, perturbation analysis (PA), sensitivity estimation, stochastic optimization.**

I. Introduction

QUALITY-OF-SERVICE (QoS) requirements for traffic (especially voice and video) such as low variance and short delay motivate establishing and maintaining, for the duration of the call, a circuit-switched path between the communicating nodes. Circuit-switching entails the reservation of limited resources (i.e., bandwidth) at each node along the circuit-switched path or circuit. If, upon arrival of a call, the desired resources are unavailable at any of the intermediate nodes, the call is said to be *blocked*. Blocked calls are assumed to be lost from the system, a mode of operation known as "blocked calls cleared" (in some models, calls that are denied immediate access can be queued until network resources are available). Common performance measures for this mode of operation include blocking probability and throughput. Circuit-switching is ubiquitous in telephony and there exists a vast body of literature on the performance analysis of circuit-switched voice in communication networks (see [21], [28], [29], and the references therein). Typically, blocking models are developed for the network in an *uncontrolled* mode of operation, i.e., a call is always accepted provided network resources are available. A fundamental issue arising in networks supporting QoS requirements is that of *call admission*: the decision to accept or reject a new call. The need for admission control, even when network resources are available, is due to the fact that the acceptance of certain calls can have a detrimental effect on the performance of the entire network, as well as the individual performance of currently active calls. The call admission problem has attracted considerable interest in recent years and is most often placed in the context of high-speed integrated services networks (see, e.g., [13] and [25]).

The problem of determining an optimal call admission policy is further complicated by its interdependence with the call routing problem. Marbach *et al.* [25], for example, consider the problem of call admission control (CAC) and routing in an integrated services network that handles several classes of calls of different value and with different resource requirements. Even when the call arrivals are assumed to be independent Poisson processes with known rates, the problem is too complex to be solved exactly, therefore an approximation is necessary. In [25], neurodynamic programming (NDP) methods are used along with a decomposition approach to tackle this problem.

In this paper, we consider the call admission problem in general-topology circuit-switched networks with fixed routing and homogeneous traffic. Fixed routing implies that all routes between source and destination (S/D) nodes have been predetermined, so that each S/D pair can be viewed as a circuit with the ith such circuit denoted by $\mathbf{c}_i$. A call routed on circuit i will be termed as a *type* i *call*. Circuit switching is generally implemented using frequency division multiple access (FDMA) or time division multiple access (TDMA) over the shared transmission channel (see, e.g., [4]). In the case of fixed networks, the multiplexing can be achieved using frequency division multiplexing (FDM), time division multiplexing (TDM), or wavelength division multiplexing (WDM) techniques. Conceptually, multiplexing can be viewed as partitioning the shared channel with capacity C bits/s, into n logical channels each with capacity C/n bits/s. Associated with each logical channel is a transceiver (transmitter/receiver pair). Equivalently, the network resources can be thought of as consisting of n transceivers at each node. A type i voice call in this model must reserve, for the duration of the call, the desired bandwidth (number of transceivers), at each node along circuit i. For simplicity, we assume that the bandwidth requirements are the same for all call types and a call needs only one transceiver at each node in the circuit. We point out, however, that if multiple traffic classes are to be explicitly modeled, this assumption is readily relaxed and *different* bandwidth requirements may be associated with different call types.

Manuscript received October 17, 2000; revised May 29, 2001. Recommended by the Guest Editors W. Gong and T. Başar. This work was supported in part by the National Science Foundation under Grants ACI-9873339 and EEC-00-88073, the Air Force Office of Scientific Research under Grant F49620-01-0056, the AFRL under Contract F30603-99-C-0057, and the EPRI/DOD under Contract WO8333-03.

The authors are with the Department of Manufacturing Engineering, Boston University, Brookline, MA 02446 USA (e-mail: kgokbayr@bu.edu; cgc@bu.edu).

Publisher Item Identifier S 0018-9286(02)05505-8.

The call admission problem in circuit-switched networks with fixed routes was considered in [2] and [31] (in the context of multihop radio networks) using a multiple-service–multiple-resource (MSMR) framework in conjunction with a coordinate convex policy [20]. Although coordinate convex policies are not necessarily optimal, Jordan and Varaiya [20] have considered several examples where such a policy performs as well as dynamic programming solutions. However, determination of the optimal coordinate convex policy (under exponentially distributed call interarrival times and call durations) requires an *offline* computationally intensive evaluation thereby limiting its applicability as the network size increases (see [31]). In [2] and [31], it has been shown that optimizing only the circuit thresholds results in control policies that are almost as good as the *optimal* coordinate convex policy which is found by optimizing both the circuit thresholds and the linear-combination constraints. Therefore, in this paper we consider a state-dependent but simpler *threshold-based* call admission policy where a type i call is accepted if and only if there are less than T_i calls currently active over circuit i. These thresholds can then be "tuned" *online* so as to optimize network performance (usually, minimization of the call blocking probability or maximization of call throughput).

Threshold-based call admission control policies have three attractive features. First and foremost, they are distributed in nature. That is, call admission is done at the source node using *local information*; this is in contrast to the uncontrolled case (i.e., no admission control) which employs call setup packets, or the coordinate convex policy in [2] and [31] which entails centralized admission control. Second, the proposed approach for determining the threshold values requires *no distributional information* on the arrival process and minimal assumptions on the nature of the call service processes. Third, it is an *adaptive* policy in the sense that the optimal threshold values are automatically adjusted as operating conditions (e.g., traffic loads or bandwidth) in the network change. The price to pay for these features is that threshold-based policies belong to the class of *complete partitioning* policies, which may make inefficient use of resources at lower network utilizations.

The main contributions of this paper are the following. First, we formulate the optimal threshold-based admission control problem as a stochastic resource allocation problem and develop a specific methodology for the *online optimization* of the thresholds so as to minimize the weighted network call blocking probability. Central to our approach is the use of sensitivity information, i.e., knowledge of the effect of changing a threshold value T_i by $+1$ or -1 on the type i call blocking probability. Thus, a second contribution is an algorithm for estimating this quantity for networks with isochronous traffic. Even with this information available, determining the optimal threshold parameters remains a hard discrete stochastic optimization problem. We present a specific algorithm toward this goal, based on transforming the original discrete optimization problem into a continuous optimization problem. Finally, we present a performance comparison of our threshold-based approach with the coordinate convex policy considered in [2] and [31], as well as the uncontrolled network. Extensive simulation results indicate that when the network is not lightly loaded, the performance of this simple distributed threshold-based policy is in fact comparable to that of the coordinate convex policy.

Note that, with minor modifications, the applicability of our algorithm can be extended to other networking environments where the bandwidth is allocated to different traffic classes over permanent or semi-permanent virtual paths (e.g., as in the asynchronous transfer mode (ATM) setting [3]).

The paper is organized as follows. In Section II, we pose the optimal threshold assignment problem $(\mathbf{P1})$. In Section III, we consider the case where the performance measure is the blocking probability over a fixed number of call arrivals and address the issue of estimating the sensitivity of this measure with respect to the threshold parameters. We show that a sample-path-based algorithm can be used to estimate such sensitivities *online*. Next, in Section IV, we return to problem $(\mathbf{P1})$. Rather than solving it directly, we use the approach recently proposed in [15], whereby we transform $(\mathbf{P1})$ into a *continuous* "surrogate" optimization problem $(\mathbf{P2})$. The latter is then solved through a stochastic approximation type algorithm which is driven by gradient estimates of the cost function with respect to the "surrogate" variables. Gradient estimation makes use of the algorithms developed in Section III. Finally, in Section V we illustrate the proposed methodology by considering several circuit-switched networks and compare the performance of the threshold-based policy with that of the coordinate convex policy considered in [2] and [31] to the uncontrolled network case.

II. Call Admission Control Problem Formulation

We consider an N-node network with fixed routing specified by C circuits. Let the ith circuit be denoted by the vector $\mathbf{c}_i = [c_{i1}, c_{i2}, \ldots, c_{iN}]$ where $c_{ij} = 1$ if circuit i traverses node j, and $c_{ij} = 0$ otherwise. For simplicity, we assume bandwidth homogeneity, i.e., the bandwidth is such that we can multiplex n voice calls on any node. Circuit-switching requires that for a call to be accepted adequate resources (i.e., a transceiver) at each node along the circuit should be available. The thresholds in our admission control policy should be chosen so as to ensure that circuit-switching is emulated. Finally, the policy is work-conserving, i.e., a type i call is always admitted if the threshold is not exceeded.

Thresholds are viewed as *partitioning* the total resources (n transceivers) at each network node. Therefore, determining the optimal thresholds is equivalent to finding the optimal partitioning of the transceivers at each node. With this in mind, let $t_i(j)$ be the number of transceivers assigned to circuit i at node j with the *capacity constraints* $\sum_{i=1}^{C} t_i(j)c_{ij} \le n$ for all $j = 1, \ldots, N$. When establishing a circuit, a call needs to reserve a transceiver at each node along the circuit which imposes a *circuit constraint* such that $t_i(j) = t_i(k) = T_i$ for all nodes j, k which belong to circuit i. We can then define a *threshold vector* $\mathbf{T} = [T_1, \ldots, T_C]$, which inherits the circuit constraints.

Consider the probability space $(\Omega, \Im, \mathbb{P})$ where $\Omega = [0,1]^\infty$ is the sample space, $\Im = 2^\Omega$ is the σ-field and P is a probability measure (depending on the properties of the system) on $\Im$. Note that $\omega \in \Omega$ is a sequence of numbers from $[0,1]$ used to create the interarrival times, the service times, and the call types and characterizes a sample path. Let $L_i(\mathbf{T})$ be the cost of circuit i observed along a sample path associated with the threshold vector $\mathbf{T}$. The resource partitioning problem is now formulated as a discrete optimization problem where the objective is to determine the vector $\mathbf{T}$ minimizing a weighted sum of expected costs $E[L_i(\mathbf{T})]$ over all circuits. In our problem, $L_i(\mathbf{T})$ is the fraction of blocked type i calls over some given time interval (or at steady state, if appropriate) and depends only on T_i, so that the performance objective becomes the minimization of a weighted blocking probability over all call types

$$(\mathbf{P1}) \quad \min_{\mathbf{T}} \sum_{i=1}^{C} \beta_i E[L_i(T_i)]$$

subject to the *resource capacity* constraints

$$\sum_{i\in D_j} T_i \le n \quad \text{for all nodes } j = 1,\ldots,N:$$
$$T_i \in \{0,1,\ldots\} \quad \text{for all circuits } i = 1,\ldots,C$$

In this formulation, β_i is the weight associated with type i calls, and

$$D_j = \{i : c_{ij} = 1; i = 1,\ldots,C\}$$

is the set of all circuits that traverse node j.

Example: To illustrate this problem formulation, consider the 6-node tandem network with five circuits shown in Fig. 1. For this network, letting $\mathbb{Z}_+ = \{0,1,\ldots\}$, the partitioning problem $(\mathbf{P1})$ is given by

$$\min_{T\in\mathbb{Z}_+^N} \sum_{i=1}^{5} \beta_i E[L_i(T_i)]$$

subject to

$$T_1 + T_4 \le n$$
$$T_2 + T_4 \le n$$
$$T_2 + T_5 \le n$$
$$T_3 + T_5 \le n$$

where the first inequality above applies to nodes 1 and 2, both of which share the call 1 and call 4 traffic. Similarly, the second inequality applies to node 3 which shares call 2 and call 4 traffic; the third inequality applies to node 4 which shares call 2 and call 5 traffic, and the last inequality applies to nodes 5 and 6 which share call 3 and call 5 traffic. In Fig. 1, the acceptance of a type 4 call $(\mathbf{c}4)$ will subsequently block resources from type 1 and type 2 calls. Thus, there exists an inherent performance trade-off between rejecting a type 4 call and freeing up resources for type 1 and type 2 calls. This tradeoff is captured by the selection of the threshold T_4.

Assuming that $E[L_i(T_i)]$ is a priori known for all feasible $\mathbf{T}, (\mathbf{P1})$ is a special case of a deterministic nonlinear integer programming problem (see [19], [26], and the references therein), and is in general NP-hard [19]. Aside from this computational difficulty, it is generally the case that no closed-form expression for $E[L_i(T_i)]$ is available, so that this cost is estimated through Monte Carlo simulation or by direct measurements made on the actual system. Thus, $(\mathbf{P1})$ involves

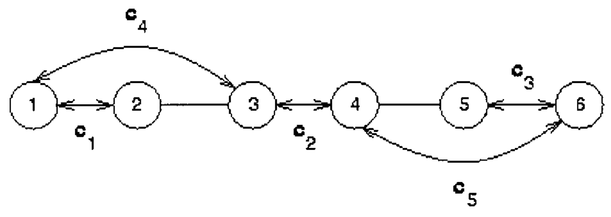


Fig. 1. Tandem network with six nodes.

both *optimization* over a discrete set and *estimation* of the cost under all feasible threshold vectors $\mathbf{T}$. It therefore becomes a *stochastic* discrete optimization problem.

For solving $(\mathbf{P1})$ we will use the approach proposed in [15]. We consider transceivers at each node as discrete resources to be allocated to circuits. By relaxing the integrality constraints on the thresholds, we transform $(\mathbf{P1})$ into a "surrogate" continuous optimization problem $(\mathbf{P2})$, which is then solved *online* through a stochastic approximation type algorithm updating the actual system as the surrogate system is updated. The solution of this "surrogate" problem can be used to recover the solution of the original problem as shown in [15]. The problem $(\mathbf{P1})$ is separable, therefore the sensitivity estimation required in the algorithm is simplified in that we will use perturbation analysis (PA) (see [9] and [17]) for the sensitivity estimation of the cost criterion with respect to the thresholds $T_i, i = 1,\ldots,C$. This estimation is based on the observed data, hence, no specific assumptions on the distributions of call arrival and holding processes will be needed.

III. Sensitivity Estimation

In this section, we focus on processing packetized traffic in a circuit-switched network. We begin by specifying the model we will adopt and then present an online sensitivity estimation algorithm for the effect of threshold parameters on the cost criterion, taken to be a weighted sum of call blocking probabilities over all call types (circuits).

A. Frame-Based Modeling for Isochronous Traffic

Circuit-switching is most commonly implemented using *TDMA*, where multiplexing is achieved by defining, for *each* network node, an n-slot *frame* such that each slot in the frame is uniquely assigned to a call, and such that the slot size is determined by the bandwidth requirements of each call type (for simplicity, we assume uniform bandwidth requirements). A call which is assigned a slot in a given frame retains, for the duration of the call, that same slot in subsequent frames. In this TDMA model, an *individual slot* within a frame corresponds to a logical channel or, equivalently, a *transceiver*.

The threshold-based policy described in Section II implies that we need to specify a frame for each network node; moreover, the respective frames must be designed so that circuit switching is emulated. For example, for the tandem network shown in Fig. 1, if a type 4 call is assigned the jth slot in the frame at the source node (node 1), then the corresponding jth slot in the frame design at node 2, and node 3 must also be reserved for that call. This slot assignment results in allocating T_i slots to type i calls in each frame for each node which belongs to circuit i. For example, in Fig. 1, the frames associated with node 1, node 2, and node 3 are designed such that T_4 slots are allocated to type 4 calls. In short, the threshold parameter T_i in our admission policy is *equivalent to the number of slots in a frame allocated to type i calls*.

The QoS associated with the traffic is such that a call is blocked if it cannot be assigned a slot in the frame. We will not be concerned here with any specific model for the traffic carried by the network; rather, we simply assume that sampling and encoding are such that at most one packet is submitted for transmission per frame. For this reason, it is referred to as *isochronous* or *continuous* traffic, in the sense that, for the duration of the call, fixed-length packets are generated at a uniform rate of 1 packet/frame.

With this frame-based model in mind, we now proceed as follows. First, at each *source* node there is a process of call arrivals for each call type that happens to originate at that node, which we assume to be characterized by an arbitrary interarrival distribution, possibly including correlations between individual call arrivals of a given call type, as well as between arrivals of different call types. The jth type i call is characterized by the pair (A_j^i, ξ_j^i), where A_j^i is the call arrival epoch and ξ_j^i is the call duration, expressed as an integer number of frames. The discrete random variable ξ_j^i has an arbitrary distribution.

Next, we describe the operation of the system from the point of view of some call type i at node q (the operation is the same from the point of view of any other call type and node). Recall that each slot in a frame is allocated to some call type; thus, the node q frame consists of (a) slots allocated to type $i (i \in D_q)$ calls, and (b) slots allocated to other call type(s). Using the terminology introduced in [7], a slot allocated to type i calls is referred to as a *transmission slot*, whereas all remaining slots are termed *vacation slots* as far as type i calls are concerned. Suppose the node q frame is designed so that T_i slots are allocated to type i calls. At any time instant, not all of these slots need to be currently utilized by such a call; in particular, let "free" slots $f_i, 0 \leq f_i \leq T_i$, be the number of transmission slots that are not presently used by any ongoing type i call (thus, there are T_i slots *allocated* to type i calls, but currently f_i of them are unused, i.e., *free*). Then, when a call (A_j^i, ξ_j^i) is submitted to the system, it is admitted if $f_i > 0$, in which case f_i is decremented by 1; otherwise, if $f_i = 0$ the call is blocked and considered lost. Note that the call admission decision is made using only local information, i.e., the number of free transmission slots allocated to i in the current node q frame. Once a slot is assigned to this particular call it remains assigned for the duration of the call, i.e., if the jth type i call is accepted and begins using the mth slot of the kth frame on the time line, then it maintains possession of the same mth slot for the $\{(k+1), \ldots, (k+\xi_j^i - 1)\}$ frames. These slots are now considered *unavailable* to future type i calls. Note that the call duration ξ_j^i *need not be known in advance*; the effect of a call terminating is simply to increment the variable f_i by 1, when this event takes place.

The model is complete once we specify how long an arriving call can wait for an accept/reject decision. We assume that the decision to accept calls is made at the beginning of each frame; all calls that arrive during a frame are therefore queued up to the beginning of the next frame. At that time, those calls that cannot be assigned an available transmission slot are blocked. A similar model was considered in [7] and [24]. Note that this model allows for an arbitrary selection of which calls at the beginning of any frame will be blocked.

Let us now concentrate on the specific performance measure of interest to our problem, i.e., the blocking probability for each type i call. Suppose a sample path of this system is observed and let b_i be the total number of type i calls that are blocked over this sample path. If K type i call arrivals were observed, then an estimate of the blocking probability is given by $(b_i(K)/K)$. Assuming that a stationary blocking probability exists under standard ergodicity conditions, we denote it by P_i and observe that

$$P_i = \lim_{K \to \infty} \frac{b_i(K)}{K}. \tag{1}$$

As already pointed out, in this model the threshold parameter T_i represents the maximum number of slots in a frame that can be assigned to type i calls. Clearly, a change in T_i generally affects the number of blocked calls b_i observed in a sample path. We now pose the following question, typical of the PA approach used in similar problems (e.g., see [8], [14], and [17]). Given that a slot is allocated to type i calls in a frame, can we predict the effect of removing that slot from the allocation to type i calls? Similarly: Given that a slot is not allocated to type i calls in a frame, can we predict the effect of adding that slot to the allocation to type i calls? Our goal is to answer these questions based *only on data directly available from an observed sample path under the current threshold parameter T_i*.

B. The "Marked/Phantom" Sensitivity Estimation Algorithm

In the observed sample path, where the system operates under a threshold T_i for type i calls, a frame at some node q contains a total number of n slots (same as the number of transceivers available) of which T_i are allocated to type i calls $(i \in D_q)$. Let us assume that K arrivals of type i call are observed over F_i frames. We then ask the question "What would happen to the number of blocked calls b_i if one less (or more) slot were allocated to these calls?" Equivalently, using the terminology introduced in the previous section, "What would happen to b_i if a transmission (or vacation) slot was converted into a vacation (or transmission) slot for type i calls?"

Our first approach in answering these questions is to view a slot allocated to the type i call as a *marked* slot and attempt to evaluate the number of blocked calls that would have resulted had this been a vacation slot instead. We emphasize that we are interested in accomplishing this based on data directly obtained from the sample path we are observing (or perhaps, simulating). Our second approach, which is the dual of the first approach, is to consider a vacation slot for type i calls and view it as a *phantom* slot, in which case our objective is to evaluate the number of blocked calls that would have resulted had this been a transmission slot instead.

We will refer to the observed system as the *nominal* one and the system that would have resulted from marking (or phantomizing) a slot as the *marked* (or *phantomized*) system. We also point out that, given the model presented in the previous section, the marking (or phantomizing) a slot within the frame at node q

is equivalent to the marking (or phantomizing) of a transceiver at that node.

Let us define the following sample path quantities.

b_i:	Total number of type i blocked calls in the nominal system.
b_i^m:	Total number of type i blocked calls in the marked system.
b_i^p:	Total number of type i blocked calls in the phantomized system.
$a_i(k)$:	Number of type i call arrivals during the kth frame of the nominal system.
$d_i(k)$:	Number of type i call completions during the kth frame of the nominal system.
$d_i^m(k)$:	Number of type i call completions during the kth frame of the marked system.
$d_i^p(k)$:	Number of type i call completions during the kth frame of the phantomized system.
$f_i(k)$:	Number of transmission slots available to type i calls in the kth frame of the nominal system (i.e., slots allocated to type i calls which are currently free). Clearly, $0 \leq f_i(k) \leq T_i$.
$f_i^m(k)$:	Number of transmission slots available to type i calls in the kth frame of the marked system. Clearly, $0 \leq f_i^m(k) \leq T_i - 1$.
$f_i^p(k)$:	Number of transmission slots available to type i calls in the kth frame of the phantomized system. Clearly, $0 \leq f_i^p(k) \leq T_i + 1$.

Then, our objective is to evaluate b_i^m (or b_i^p) using only quantities observed along a sample path of the nominal system. The value will be used in evaluating the sensitivity of the blocking probability for type i calls (with respect to the threshold parameter T_i) defined as

$$\frac{\Delta b_i^m}{K} = \frac{b_i^m - b_i}{K} \tag{2}$$

or

$$\frac{\Delta b_i^p}{K} = \frac{b_i - b_i^p}{K}. \tag{3}$$

Note that the dependence on the threshold parameter T_i is not shown for notational convenience.

Before proceeding with the construction of the sample paths for marked and phantomized systems let us analyze the nominal system sample path. Our analysis begins with the following simple recursive equation which $f_i(k)$ satisfies

$$f_i(k+1) = [f_i(k) + d_i(k) - a_i(k)]^+, f_i(0) = T_i \tag{4}$$

where $[x]^+ = \max(0, x)$. Thus, the number of free slots in a frame is initially given by the threshold parameter T_i for type i calls. Subsequently, $f_i(k)$ is incremented by the number of the call completions $d_i(k)$ and decremented by the number of new calls $a_i(k)$, with the obvious constraint that $f_i(k)$ must remain nonnegative for all k.

1) Construction of the Marked System Sample Path: The recursive equation for the free slots of the marked system can be written as follows:

$$f_i^m(k+1) = [f_i^m(k) + d_i^m(k) - a_i(k)]^+, f_i^m(0) = T_i - 1. \tag{5}$$

In the marked system $a_i(k)$ is directly observable from the nominal system, whereas $d_i^m(k)$ will be obtained as explained below.

Let us now introduce the concept of a *tagged* call. This is a call that is accepted in the nominal but not in the marked system at the end of some frame. We define an additional binary variable $z_i(k)$ (indicator of a tagged call at the beginning of kth frame) initialized as $z_i(0) = 0$.

Now let

$$u = \min\{k : f_i(k) + d_i(k) \leq a_i(k), k = 0, 1, \ldots\}.$$

Note that if such u does not exist, $f_i(k) + d_i(k) > a_i(k)$ for all k, which means removing a slot does not affect the cost, i.e., the sensitivity is zero on the observed sample path.

Assuming existence of u, we have

$$f_i^m(k) = f_i(k) - 1 \quad \text{for all } k = 0, \ldots, u$$
$$d_i^m(k) = d_i(k) \quad \text{for all } k = 0, \ldots, u.$$

A call arriving in the uth frame will be tagged. In the nominal system $f_i(u) + d_i(u)$, new calls are accepted, while in the marked system only $f_i^m(u) + d_i^m(u) = f_i(u) + d_i(u) - 1$ new calls are accepted. Equivalently, $a_i(u) - f_i(u) - d_i(u)$ calls are blocked in the nominal, while $a_i(u) - f_i(u) - d_i(u) + 1$ calls are blocked in the marked system. This additional call that is blocked only in the marked system is precisely what we have defined above as 'tagged'. We then set $z_i(u+1) = 1$. It also follows from (4) and (5) that

$$f_i(u+1) = 0 = f_i^m(u+1). \tag{6}$$

Let the termination of this tagged call occur at some frame $l > u$. For all $k = u+1, \ldots, l-1$, note that (4) and (5) apply to the nominal and marked system respectively with the new initial condition (6). Note that

$$d_i^m(k) = d_i(k) \quad \text{for all } k = u+1, \ldots, l-1$$

therefore

$$f_i^m(k) = f_i(k) \quad \text{for all } k = u+1, \ldots, l.$$

That is, for the duration of this tagged call, both the nominal and the marked system see the same number of blocked calls. The fact that the marked system has one less transmission slot for type i calls is compensated by the fact that the extra slot of the nominal system is used by the tagged call. Note that $d_i^m(k)$ and $d_i(k)$ differ by one at the completion of a tagged call, when $k = l$, and are equal at other times.

Finally in the lth frame, the nominal system and the marked system will have $f_i(l) + d_i(l)$ and $f_i^m(l) + d_i^m(l) = f_i(l) + d_i(l) - 1$ available slots, respectively. Two cases need to be considered.

- $f_i(l) + d_i(l) \le a_i(l)$ In this case the slot freed up by the completion of the tagged call will be used by a new tagged call, therefore
$$z_i(l+1) = 1.$$
The process above repeats with the initial condition
$$f_i(l+1) = 0 = f_i^m(l+1).$$
- $f_i(l) + d_i(l) > a_i(l)$ In this case, the slot freed up by the completion of the tagged call is not used by a new one, therefore
$$z_i(l+1) = 0.$$
The process above repeats with the initial condition
$$f_i(l+1) > 0 \quad \text{and} \quad f_i^m(l+1) = f_i(l+1) - 1.$$

The tagging process $\{z_i(k)\}, k = 0, 1, 2, \ldots$ therefore consists of cycles as described above. In order to formally define the dynamics of $z_i(k)$, let us introduce one more binary variable $y_i(k)$ as the indicator of completion of a tagged call of type i within the kth frame. We then have

$$z_i(k+1) = \begin{cases} 1 & \text{if } f_i(k) + d_i(k) \le a_i(k) \\ 0 & \text{if } f_i(k) + d_i(k) > a_i(k) \text{ and } y_i(k) = 1 \\ z_i(k) & \text{otherwise} \end{cases}$$

with the initial condition $z_i(0) = 0$. Note that $z_i(k)$ is completely determined from observable quantities along the nominal sample path, $a_i(k), d_i(k), f_i(k)$, and the observable events corresponding to tagged call completions. It should be clear that there can be at most one tagged call in the nominal system at any instant, since only one transmission slot is removed in the marked system.

The final step is the evaluation of the sensitivity of the blocking probability for type i calls, $(\Delta b_i^m / K)$, defined in (2). By definition, this is the ratio of the total number of tagged calls over the observed sample path to the total number of observed arrivals. Let $\mathbf{1}[\cdot]$ be the usual indicator function. We then get

$$\frac{\Delta b_i^m}{K} = \frac{1}{K} \sum_{k=0}^{F_i - 1} \mathbf{1}[f_i(k) + d_i(k) \le a_i(k)] \times \mathbf{1}[z_i(k) = 0 \text{ or } y_i(k) = 1]. \quad (7)$$

In practice, Δb_i^m is simply incremented by 1 with every transition of $z_i(k)$ from 0 to 1, including the case where it becomes 0 in some frame k and then immediately reset to 1 $(y_i(k) = 1)$, i.e., a tagged call is terminated in that frame and another tagged call is accepted.

2) Construction of the Phantomized System Sample Path: The recursive equation for the free slots of the phantomized system can be written as follows:

$$f_i^p(k+1) = \left[f_i^p(k) + d_i^p(k) - a_i(k)\right]^+, \quad f_i^p(0) = T_i + 1. \quad (8)$$

Let

$$u = \min\{k: f_i(k) + d_i(k) < a_i(k), k = 0, 1, \ldots\}.$$

If u does not exist, then the nominal system does not block any calls of type i. Adding another slot will not improve the performance, i.e., the sensitivity is zero on the observed sample path.

Assuming existence of u, the phantomized and the nominal system will accept every call until the uth frame. From (8) and (4)

$$f_i^p(k) = f_i(k) + 1 \quad \text{for all } k = 0, \ldots, u$$
$$d_i^p(k) = d_i(k) \quad \text{for all } k = 0, \ldots, u.$$

In the uth frame, a call, referred to as the *phantom call*, is blocked in the nominal system but will be accepted in the perturbed one. The service time ξ (number of frames a call will use a slot) needed for this call will be unavailable from the nominal system. We will address the issue of assigning a service time to the phantom call in Section III-B3.

In the uth frame, the nominal system accepted $f_i(u) + d_i(u)$ new calls, while $f_i^p(k) + d_i^p(k) = f_i(u) + d_i(u) + 1$ calls are accepted by the phantomized system. Equivalently, $a_i(u) - (f_i(u) + d_i(u))$ calls are blocked in the nominal system, while $a_i(u) - (f_i(u) + d_i(u)) - 1$ calls are blocked in the phantomized system. The additional call that is accepted is what we defined above as “phantom.” We then set $z_i(u+1) = 1$ where $z_i(k)$ is defined as the indicator of presence of a phantom call of type i at the beginning of the kth frame and initialized as $z_i(0) = 0$. It also follows from (4) and (8) that:

$$f_i(u+1) = 0 = f_i^p(u+1). \quad (9)$$

Let the termination of the phantom call occur at $l = u + \xi$. For all $k = u+1, \ldots, l-1$, (4) and (8) apply to the nominal and the marked system respectively with the new initial condition (9). Note that

$$d_i^p(k) = d_i(k) \quad \text{for all } k = u+1, \ldots, l-1$$

therefore

$$f_i^p(k) = f_i(k) \quad \text{for all } k = u+1, \ldots, l.$$

That is, for the duration of the phantom call, both the nominal and the phantomized systems see the same number of blocked calls. The fact that the phantom system has an additional transmission slot for type i calls is compensated by the fact that the extra slot is being used by the phantom call. Note that the $d_i^p(k)$ and $d_i(k)$ differ by one at the completion of a phantom call, when $k = l$, and are equal at other times.

Finally in the lth frame, the nominal system and the phantomized system will have $f_i(l) + d_i(l)$ and $f_i^p(l) + d_i^p(l) = f_i(l) + d_i(l) + 1$ available slots, respectively. Two cases need to be considered.

- $f_i(l) + d_i(l) < a_i(l)$ In this case, the slot freed up by the termination of the phantom call is filled by a new phantom call, therefore
$$z_i(l+1) = 1.$$
The process above starts with the initial condition
$$f_i(l+1) = 0 = f_i^p(l+1).$$
- $f_i(l) + d_i(l) \geq a_i(l)$ In this case, the slot freed up by the termination of the phantom call is not used by a new call, therefore
$$z_i(l+1) = 0.$$
The aforementioned process starts with the initial condition
$$f_i(l+1) + 1 = f_i^p(l+1) > 0.$$

The sequence $\{z_i(k)\}, k = 0, 1, \ldots$ therefore consists of cycles as described above. In order to formally define the dynamics of $z_i(k)$, let us introduce one more binary variable $y_i(k)$ as the indicator of termination of a phantom call of type i within the kth frame. We then have

$$z_i(k+1) = \begin{cases} 1 & \text{if } f_i(k) + d_i(k) < a_i(k) \\ 0 & \text{if } f_i(k) + d_i(k) \geq a_i(k) \text{ and } y_i(k) = 1 \\ z_i(k) & \text{otherwise} \end{cases}$$

with the initial condition $z_i(0) = 0$. Note that $z_i(k)$ is completely determined from observable quantities along the nominal sample path, $a_i(k), d_i(k), f_i(k)$, and the observable events corresponding to phantom call completions with the added difficulty of injecting a service time. It should be clear that there can be at most one phantom call of type i in the phantomized system at any instant, since only one transmission slot is added in the phantomized system.

The final step is evaluation of the sensitivity of the blocking probability for type i calls, $(\Delta b_i^p/K)$, defined in (3). By definition, this is the ratio of the total number of tagged calls over the observed sample path to the total number of observed arrivals. Let $\mathbf{1}(\cdot)$ be the usual indicator function. We then get:

$$\frac{\Delta b_i^p}{K} = \frac{1}{K} \sum_{k=0}^{F_i-1} \mathbf{1}[f_i(k) + d_i(k) < a_i(k)] \times \mathbf{1}[z_i(k) = 0 \text{ or } y_i(k) = 1] \quad (10)$$

In practice, Δb_i^p is simply incremented by 1 with every transition of $z_i(k)$ from 0 to 1, including the case where it becomes 0 in some frame k and then immediately reset to 1 ($y_i(k) = 1$), i.e., a phantom call is terminated in that frame and another phantom call is accepted.

3) Service Time Assignment to the Phantom Calls: In the phantomized system, $a_i(k)$ is directly observable from the nominal system (because the arrivals are never disabled). However, if a call that arrived in the kth frame is blocked by the nominal system and is accepted in the phantomized system, the service duration for that call, which is essential in constructing the phantomized system, will be unobservable. Assuming that the service times are independent and identically distributed, we will consider two cases.

Case 1) The service time distribution for ξ^i is available
If the service time distribution is available (e.g., from previous data), one can assign a service time to this phantom call by sampling from the distribution.

Case 2) The service time distribution for ξ^i is not available
If the service time distribution is not available, we can apply the *time warping algorithm* (*TWA*) proposed in [10].

While constructing the marked sample path we assume that the calls come with their service times. However, in order to construct the phantomized sample path, we assume that the transceiver assigns the service times. These two assumptions are statistically equivalent because of the i.i.d. property.

The TWA keeps track of the service times that are observed in the nominal system. When the service time for a call is not available, the construction of the phantomized system is paused. As soon as the service time becomes available, the construction of the phantomized system starts and proceeds in a *time warping* fashion. Let us define v_j^i as the service time of the jth accepted call of type i to the nominal system. Note that $v_j^i = \xi_{j+b^i}^i$ where b^i is the number of blocked calls of type i in the nominal system up to jth accepted call. The phantomized system sample path will be constructed using v_j^i value of the nominal system as the service time for the jth accepted call in the phantomized system. Note that since the phantomized system may have more accepted arrivals than the nominal system, one may need to observe more arrivals than K to the nominal system in order to get enough service time data for the phantomized system.

After the phantomized system is constructed for K arrivals (possibly corresponding to more arrivals in the nominal system), the sensitivity is calculated as in (3) where b_i is the number of the blocked type i calls in the "first" K arrivals to the nominal system.

IV. Optimal Call Admission Threshold Determination

Let us now return to problem $(\mathbf{P1})$, where a threshold vector $\mathbf{T}$ which satisfies the capacity constraints at the nodes is sought to minimize a weighted sum of blocking probabilities.

While the area of stochastic optimization over *continuous* decision spaces is rich and usually involves gradient-based techniques as in several well-known stochastic approximation algorithms [22], [27], the literature in the area of *discrete* stochastic optimization is relatively limited. Most known approaches are based on some form of random search, with the added difficulty of having to estimate the cost function at every step. Such algorithms have been recently proposed by Yan and Mukai [32], Gong *et al.* [16], and Shi and Olafsson [30]. Although such random search techniques have general applicability, they do not fully exploit any inherent structure of the problem. They also tend to be exceedingly slow in reaching a good solution, since these algorithms typically have to visit several poor allocations. This is a particularly undesirable feature for online algorithms we are interested in here, where speed is essential and operating at an arbitrarily selected point can lead to extremely poor performance.

Another recent contribution to this area involves the *ordinal* optimization approach presented in [18]. The basic premise of ordinal optimization is that it requires fewer resources to identify the best solution among several candidates than identifying how much better one is from the others. One can obtain sensitivity estimates $(\Delta b_i^m/K)$ (or $(\Delta b_i^p/K)$) in order to identify the "least" and the "most" sensitive call types and transfer a slot (i.e., a transceiver) from the former to the latter. Among other features, this approach is intended to exploit the fact that ordinal estimates are particularly robust with respect to estimation noise compared to cardinal estimates (see also [11]). The implication is that convergence of such algorithms is substantially faster. One such algorithm is applied to resource allocation problems with total capacity constraints in [5]. This algorithm was shown to converge in probability (and a.s. under certain added conditions [12]). Even though the approach in [5] yields a fast resource allocation algorithm, it is still constrained to iterate so that every step involves the transfer of no more than a single resource from one user to some other user. One can expect, however, that much faster improvements can be realized in a scheme allowed to reallocate multiple resources from users whose cost-sensitivities are small to users whose sensitivities are much larger. This is precisely the rationale of most gradient-based continuous optimization schemes, where the gradient is a measure of this sensitivity.

With this motivation in mind, we apply here the algorithm that was introduced in [15]. In particular, we transform the original discrete feasible set into a continuous feasible set over which a "surrogate" optimization problem is defined and subsequently solved. As in earlier work in [5] and [6], and unlike algorithms presented in [19], an important feature of our approach is that every state $\mathbf{T}$ in the optimization process remains feasible, so that our scheme can be used *online* to adjust the decision vector as operating conditions (e.g., system parameters) change over time. Thus, at every step of the continuous optimization process, the obtained continuous state is mapped back into a feasible discrete state using a specific transformation; based on a realization under this feasible state, new sensitivity estimates are obtained that drive the continuous optimization process to yield the next continuous state. Therefore, the proposed scheme involves an interplay of sensitivity-driven iterations and continuous-to-discrete state transformations. It is shown in [15] that when an optimal threshold vector $\boldsymbol{\tau}^*$ is obtained in the continuous state space, it is transformed to the optimal threshold vector $\mathbf{T}^*$. In other words, if a solution to the surrogate optimization problem is found, then a solution to the original discrete stochastic optimization problem is also obtained through the transformation used to map continuous into discrete states. Convergence of the algorithm is also established in [15] under standard assumptions common to the stochastic approximation framework.

Let us now apply this approach to solving problem $(\mathbf{P1})$. We will start with relaxing the constraint that T_i is integer for $i = 1, \ldots, C$. The relaxed problem can be formulated as

$$(\mathbf{P2}) \quad \min_{\boldsymbol{\tau}} \sum_{i=1}^{C} \beta_i E[L_i(\tau_i)]$$

subject to the *resource capacity* constraints

$$\sum_{i \in D_j} \tau_i \le n \quad \text{for all nodes } j = 1, \ldots, N:$$
$$\tau_i \ge 0 \quad \text{for all circuits } i = 1, \ldots, C.$$

In this formulation, τ_i is the "surrogate" variable (real-valued threshold) for type i calls, β_i is the weight associated with type i calls, and

$$D_j = \{i : c_{ij} = 1; i = 1, \ldots, C\}$$

is the set of all circuits that traverse node j.

The algorithm to solve this minimization problem is as follows (see also [15]).

- Start with some (initial) threshold vector $\mathbf{T}_0$, and set $\boldsymbol{\tau}_0 = \mathbf{T}_0$.
- For any iteration $n = 0, 1, \ldots$:
    1) Perturb $\boldsymbol{\tau}_n$ (if necessary) so that all components are noninteger.
    2) Select $\mathbf{T}_n = \arg\min_{r \in A_d} ||\mathbf{T} - \boldsymbol{\tau}_n$ where A_d is the discrete-feasible set.
    3) Operate at $\mathbf{T}_n$ to evaluate sensitivity estimate H_n using the PA techniques in Section III.
    4) Update the continuous threshold vector: $\boldsymbol{\tau}_{n+1} = \arg\min_{\boldsymbol{\tau} \in A_c} ||\boldsymbol{\tau} - (\boldsymbol{\tau}_n - \eta_n H_n)||$ where A_c is the convex hull of A_d.
    5) If some stopping condition is not satisfied, repeat steps for $n+1$. Else, set $\boldsymbol{\tau}^* = \boldsymbol{\tau}_{n+1}$.
- Obtain $\mathbf{T}^*$ as one of the neighboring feasible states in the set.

Note that $||\cdot||$ is the standard Euclidean norm. Since the cost function is defined as the summation of weighted blocking probabilities, the derivative estimates H_n are calculated as follows: Depending on the value of the ith component of $(\mathbf{T}_{m+1} - \boldsymbol{\tau}_{m+1})$, the marked or the phantom slot approach is used, corresponding to the left or right derivative, respectively. Then

$$(H_n)_i = \begin{cases} -\beta_i \frac{\Delta b_i^p}{K} & (\mathbf{T}_{m+1})_i < (\boldsymbol{\tau}_{m+1})_i \\ -\beta_i \frac{\Delta b_i^m}{K} & (\mathbf{T}_{m+1})_i > (\boldsymbol{\tau}_{m+1})_i \end{cases} \tag{11}$$

where $(\Delta b_i^m/K)$and $(\Delta b_i^p/K)$ are given by (7) and (10) respectively. For details on the derivation of the gradient of the "surrogate" cost function in terms of the aforementioned finite differences, see [15].

This algorithm is a standard stochastic approximation scheme driven by the derivative estimates H_n with $\{\eta_n\}$ an appropriately selected step size sequence (e.g., see [15], [22], [23], and [27]). It is worth pointing out that the conditions required for convergence [15] are not dependent on the stochastic characteristics of the call arrival process, except for standard ergodicity assumptions mentioned in Section III-A. In particular, there are six technical conditions required to establish convergence to a global optimum (for details, see [15]). Two of them involve the choice of the step size sequence $\{\eta_n\}$. Two more pertain to the noise process and are satisfied by unbiased and consistent sensitivity estimators in (11). These estimators are not dependent on the distribution of the arrival process; in fact, because of the simple finite-difference nature of the estimators, these

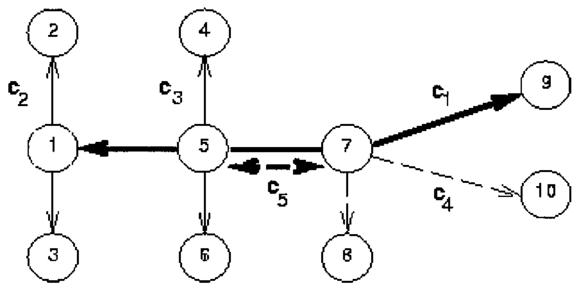


Fig. 2. 10-node circuit-switched network with five circuits.

properties are tantamount to the ergodicity assumptions made in Section 3.1 (for similar types of estimators, see [8], [9], and [17]). The fifth technical condition is imposed on the cost function in order to guarantee a unique global optimum and is independent of the arrival process. The final condition is the requirement $\sup_{\tau\in A_c}\|H(\tau)\|<\infty$. Looking at (11), it is easy to see that blocking probability finite differences are bounded by 0 and 1, thus satisfying this condition (recalling that in (7) and (10) K is a fixed number of observed arrivals over F_i frames).

V. Numerical Examples

In this section, we illustrate our call admission control approach and its features by considering several circuit-switched networks and making comparisons with alternative approaches.

Example 1: Consider the 10-node circuit-switched network with five circuits (S/D pairs) shown in Fig. 2. For the purpose of analytically determining blocking probabilities, we assume Poisson call arrivals for circuit i with rate λ_i and exponential call durations with mean $(1/\mu_i)$. Note that call duration in this example is not an integer which does not satisfy our frame-based model given in Section III-A.

Our objective is to determine the optimal thresholds T_i^* (in the call admission policy) so as to minimize the weighted network call blocking probability

$$P_b=\frac{\sum_{i=1}^{5}\lambda_iP_i(T_i)}{\sum_{j=1}^{5}\lambda_j}$$

where $P_i(T_i)$ is the expected circuit i call blocking probability with assigned threshold T_i. Under the previously mentioned Markovian modeling assumptions

$$P_i(T_i)=\frac{\rho_i^{T_i}/T_i!}{\sum_{j=0}^{T_i}\rho_i^j/j!} \tag{12}$$

where $\rho_i=(\lambda_i/\mu_i)$.

In the simulation experiment, we assume $n=15$ transceivers per node and circuit loads

$$\rho_1=\rho_3=\rho_4=1 \quad \text{and} \quad \rho_2=\rho_5=2.$$

The corresponding separable optimization problem is

$$\min_{T\in\mathbb{Z}_+^N}\sum_{i=1}^{5}\beta_iP_i(T_i)$$

TABLE I
Single Run Optimization of 10 Node-5 Circuit Network

k	**T**	P_b^*
0	$(9,6,2,2,4)$	0.0878
1	$(6,9,5,5,4)$	0.0282
2	$(4,11,2,2,9)$	0.0594
3	$(3,12,7,7,5)$	0.0194
4	$(4,11,6,6,5)$	0.0128
5	$(3,12,5,5,7)$	0.0108
6	$(4,11,5,5,6)$	0.0065
7	$(4,11,5,5,6)$	0.0065
8	$(4,11,5,5,6)$	0.0065

subject to

$$T_1+T_2\le 15$$
$$T_1+T_3+T_5\le 15$$
$$T_1+T_4+T_5\le 15$$

where $\beta_i=\lambda_i/\sum_{j=1}^{5}\lambda_j$ and all call types are of equal importance. With the aforementioned system parameters, through an exhaustive search the optimal thresholds are determined to be

$$T_1^*=4, T_2^*=11, T_3^*=5, T_4^*=5, T_5^*=6$$

and the corresponding blocking probability is $P_b^*=0.0065$.

To illustrate our approach, we perform a single-run optimization of the thresholds in the call admission policy. We employ a gradient projection method, (gradients in this case are directly calculated using (11) and (12) so that there is no estimation noise) with a constant step size $\eta_k=300$.

In Table I, we observe that starting from an initial threshold assignment of $T_1=9, T_2=6, T_3=2, T_3=2, T_5=4$, we converge to the optimal threshold vector in only six steps.

Example 2: In this set of experiments we again consider the 10-node network shown in Fig. 2, but now the objective is to compare the performance of the optimal threshold based policy with the optimal coordinate-convex policy proposed in [2] and [31] (subsequently referred to as CC) as well as the uncontrolled system. As before, we assume Poisson call arrivals and exponential call durations for each circuit. The network has $n=3$ transceivers per node and each circuit has uniform load $\rho_i=\rho$. This example was used in [1] and chosen here so as to make comparisons with the results for the uncontrolled network and under the coordinate convex policy reported in [1]. Table II shows a comparison of the weighted network call blocking under the optimal thresholds (P_b^*), the uncontrolled network, and the optimal coordinate-convex policy $(P_{\rm CC}^*)$ (The blocking probabilities for the latter two cases have been reproduced from [1]).

There are a number of interesting observations pertaining to Table II. One observation is that under low network loads, the threshold-based policy is overly conservative, and the resulting network blocking is worse than even the uncontrolled system. However, as the network load increases, the relative performance of the threshold-based policy improves, and at high loads it results in the same performance as the coordinate-convex policy (as already mentioned, the threshold-based

TABLE II
COMPARISON OF BLOCKING PROBABILITIES UNDER DIFFERENT CALL ADMISSION POLICIES

ρ	$\mathbf{T}$	P_b^*	Uncontrolled	P_{CC}^*
0.5	$(1,2,1,1,1)$	0.2821	0.1353	0.1353
1.0	$(0,3,2,2,1)$	0.3925	0.3222	0.3220
2.0	$(0,3,3,3,0)$	0.5263	0.5297	0.5122
3.0	$(0,3,3,3,0)$	0.6077	0.6348	0.6077
10.0	$(0,3,3,3,0)$	0.8392	0.8507	0.8392

TABLE III
COMPARISON OF THRESHOLD-BASED POLICY AND SELF-REGULATION (FOR UNCONTROLLED SYSTEM) UNDER PRIORITIZED TRAFFIC

ρ	w_1	w_j^+	$\mathbf{T}$	P_b^w	Uncontrolled
3	1.5	1.0	$(0,3,3,3,0)$	0.7077	0.7213
3	2.0	1.0	$(0,3,3,3,0)$	0.8077	0.8077
3	5.0	1.0	$(3,0,0,0,0)$	1.1462	1.3264
3	10.0	1.0	$(3,0,0,0,0)$	1.4923	2.1908

policy has the advantage of requiring a simple online distributed implementation). In particular, at high loads, the optimal policy is never to accept a type 1 call (which blocks all calls on the remaining circuits) or a type 5 call (which blocks calls on circuits 1, 3, and 4). Another observation is that there is a marginal improvement (at most 5% reduction) in the blocking probability resulting from exercising admission control in this example. This observation is attributed to *self-regulation* properties of such circuit-switched networks pointed out in [1], wherein the network tends to accept more calls on circuits that compete for resources with the fewest other circuits, and block calls that tend to interfere with many other circuits. Thus, when all call types are of the *same weight*, the effect of network self-regulation in an uncontrolled network achieves nearly the same performance as that of optimal call admission policy. Therefore, in Fig. 2, self-regulation forces the network, in a majority of the cases in Table II, to reject type 1 and type 5 calls and in doing so achieves near-optimal control.

However, when call types are assigned different weights, exercising a call admission policy can result in a significant performance improvement. For example, if we associate a weight w_i with type i calls, then the performance measure of interest is now of the form

$$P_b^w = \frac{\sum_{i=1}^5 w_i \lambda_i P_i(T_i)}{\sum_{i=1}^5 \lambda_i}.$$

In Table III optimal P_b^w values (from threshold-based policy) are compared to the uncontrolled system blocking probabilities for the network shown in Fig. 2. As before, we assume $n = 3$ transceivers per node, uniform load $\rho_i = \rho$ for all $i = 1, \ldots, 5$ and calls have assigned weights w_i as shown (w_j^+ denotes the equal weight assigned to call type j for $j = 2, \ldots, 5$). In Table III, we observe that as the weight for call type 1 is increased, the optimal decision becomes to always accept type 1 calls, whereas in the uncontrolled network, because of self-regulation, the decision is to restrict type 1 calls.

Example 3: The ability of our approach to adjust the thresholds in response to changing operating conditions, is seen in Table IV, where now the circuit load profile is varied. In this example we assume $n = 8$ transceivers and a load profile (ρ_1, ρ_j^+) where $\rho_j^+ = \rho_2 = \rho_3 = \rho_4 = \rho_5$. We can see that as the parameter ρ_j^+ changes, the algorithm adjusts the threshold values accordingly.

TABLE IV
COMPARISON OF BLOCKING PROBABILITIES WITH VARYING CIRCUIT LOAD PROFILE

ρ_1	ρ_j^+	$\mathbf{T}$	P_b	Uncontrolled
9.0	0.1	$(8,0,0,0,0)$	0.3194	0.3059
9.0	1.0	$(6,2,2,2,0)$	0.4280	0.3864
9.0	2.0	$(4,4,3,3,1)$	0.4641	0.4284
9.0	3.0	$(3,5,4,4,1)$	0.4845	0.4680
9.0	5.0	$(0,8,6,6,2)$	0.5051	0.5339
9.0	7.0	$(0,8,8,8,0)$	0.5339	0.5917
9.0	10.0	$(0,8,8,8,0)$	0.5949	0.6629

Example 4: In Examples 1–3, on account of the Markovian modeling assumptions, we had analytical expressions for the gradients needed to drive the *online* optimization algorithm. If such expressions are not available, the gradients in question must be estimated through simulation or direct measurements made on the actual system. In this example, we perform *online* optimization with gradients estimated via the Marked/Phantom Slot estimator. Consider the 6-node tandem network with five circuits shown in Fig. 1. We assume *isochronous* traffic (see Section III-A) where the jth type i call is characterized by the pair (A_j^i, ξ_j^i). Recall that A_j^i is the call arrival epoch and ξ_j^i is the call duration, specified as the number of packets. The optimization problem is formulated as follows:

$$P = \min_{T \in \mathbb{Z}_+^N} \sum_{i=1}^{5} \beta_i E[L_i(\mathbf{T})]$$

subject to

$$\begin{aligned} T_1 + T_4 &\le 24 \\ T_2 + T_4 &\le 24 \\ T_2 + T_5 &\le 24 \\ T_3 + T_5 &\le 24 \end{aligned}$$

where $E[L_i(\mathbf{T})]$ is the expected blocking probability associated with the threshold vector $\mathbf{T}$.

We perform a single-run optimization experiment, where we assume $n = 24$ transceivers per node. The initial threshold vector is selected to be $\mathbf{T}^{(0)} = [1, 1, 1, 23, 23]$. Call arrivals are assumed to be Poisson with rate $\lambda_i = 0.4$ arrivals per second for all $i = 1, \ldots, 5$. Call duration ξ_i is uniformly distributed over $\{1, 2, \ldots, 9\}$ for all $i = 1, \ldots, 5$. Each packet is transmitted in 1 s (a frame is 24 s long). We employ a gradient-based algorithm with projection where we keep the step size $\eta_k = 10\,000$ for all update steps k. The observation intervals over which the estimation is carried out before an update occurs are chosen to gradually increase in length. In particular, the kth interval length is defined through $I_k = I_{k-1} + R$ with $R = 200$ call arrivals and $I_0 = 2000$ call arrivals. Each call is given the same weight $\beta_i = 1$, for $i = 1, \ldots, 5$. The algorithm (explained in Section IV) performs as shown in Table V.

TABLE V
OPTIMIZATION OF 6-NODE TANDEM NETWORK WITH SYMMETRIC TRAFFIC

k	$\mathbf{T}$	$P^{(k)}$
0	$(1, 1, 1, 23, 23)$	3.994000
1	$(24, 24, 24, 0, 0)$	3.490909
2	$(24, 24, 24, 0, 0)$	3.506250
3	$(24, 24, 24, 0, 0)$	3.535385

TABLE VI
OPTIMIZATION OF 6-NODE TANDEM NETWORK WITH SYMMETRIC TRAFFIC (SHORT OBSERVATION INTERVALS)

k	$\mathbf{T}$	$P^{(k)}$
0	$(1, 1, 1, 23, 23)$	3.840000
1	$(13, 13, 13, 11, 11)$	3.950000
2	$(24, 24, 24, 0, 0)$	3.528571
3	$(24, 24, 24, 0, 0)$	3.537500

Note that the algorithm converged to the optimal in just one step. Since the circuits have identical loads and β_i weights, $(24, 24, 24, 0, 0)$ is the optimal threshold vector for this example because the transceivers reserved for call type 4 can be used to accommodate both call types 1 and 2. Similarly, the transceivers reserved for call type 5 can be used to accommodate both call types 2 and 3.

One can observe the effect of noise in Table V, where threshold vector $\mathbf{T} = (24, 24, 24, 0, 0)$ yields different $P^{(k)}$ values. In order to see the effect of having shorter observation intervals, which would increase the effect of noise, let us perform the same experiment where the kth interval length is defined through $I_k = I_{k-1} + R$ with $R = 10$ call arrivals and $I_0 = 50$ call arrivals. We will use a smaller step size $\eta_k = 1000$ because the derivative estimation is not as reliable as in the previous case. The results of this experiment are given in Table VI.

Note that even though the network is still in the transient state (initially the network is assumed to be empty), the threshold vector reached the optimal value at the second update.

VI. CONCLUSION

We have considered threshold-based call admission policies for circuit-switched networks and developed a scheme for adjusting the threshold parameters *online*, the objective being to minimize a weighted sum of call blocking probabilities. Such threshold-based policies are conservative at low traffic rates (i.e., they may reject more calls than necessary), but, as the numerical results of Section V also indicate, at higher traffic rates they yield the same performance as more complex call admission policies. In addition to this, the main advantages of this threshold-based admission control scheme lie in its implementational simplicity, and the facts that: it is completely distributed in nature; it is adaptive in the sense that it can automatically adjust the thresholds as the operating conditions change; and it does not require any explicit distributional modeling assumptions.

Central to this admission control scheme is the Marked/Phantom Slot algorithm developed in Section III for *online* estimation of the sensitivity of the call blocking metric defined above with respect to the thresholds. It is because this algorithm is based on directly observable network data (e.g., call termination events, number of call arrivals per frame) that no special distributional modeling assumptions are required. Our approach for optimizing over the set of feasible thresholds is based on the recently proposed "surrogate" problem method [15], as described in Section IV. This requires estimating gradients with respect to the surrogate control parameters, which is accomplished by making use of the Marked/Phantom Slot algorithm.

Lastly, note that the approach we have presented need not be limited to the call blocking probability metric. Similar admission control problems can be formulated with more general cost functions or with multiple objectives if several traffic classes are to be explicitly modeled.

REFERENCES

[1] C. M. Barnhart, J. E. Wieselthier, and A. Ephremides, "Admission Control in Integrated Voice/Data Multihop Radio Networks," Naval Research Laboratory, Tech. Rep., NRL/MR/5521-93-7196.

[2] ——, "Admission control policies for multihop wireless networks," *Wireless Networks*, vol. 1, no. 4, pp. 373–387, 1995.

[3] J. Boudec, "The asynchronous transfer mode: A tutorial," *Comp. Networks ISDN Syst.*, vol. 24, pp. 279–309, 1992.

[4] A. B. Carlson, *Communication Systems*. New York: McGraw-Hill, 1986.

[5] C. G. Cassandras, L. Dai, and C. G. Panayiotou, "Ordinal optimization for deterministic and stochastic resource allocation," *IEEE Trans. Automat. Contr.*, vol. 43, pp. 881–900, July 1998.

[6] C. G. Cassandras and V. Julka, "A new approach for some combinatorially hard stochastic optimization problems," in *Proc. 31st Annual Allerton Conf. Communication, Control, Computing*, 1993, pp. 667–676.

[7] ——, "Optimal scheduling in systems with delay-sensitive traffic," in *Proc. 32st IEEE Conf. Decision Control*, 1993, pp. 1685–1691.

[8] ——, "Scheduling policies using marked/phantom slot algorithms," *Queuing Syst.: Theory Appl.*, vol. 20, pp. 207–254, 1995.

[9] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. Norwell, MA: Kluwer, 1999.

[10] C. G. Cassandras and C. G. Panayiotou, "Concurrent sample path analysis of discrete event systems," *J. Discrete Event Dyna. Syst.: Theory Appl.*, vol. 9, pp. 171–195, 1999.

[11] L. Dai, "Convergence properties of ordinal comparison in the simulation of discrete event dynamic systems," *J. Optimiz. Theory Appl.*, vol. 91, pp. 363–388, 1996.

[12] L. Dai, C. G. Cassandras, and C. G. Panayiotou, "On the convergence rate of ordinal optimization for a class of stochastic discrete resource allocation problems," *IEEE Trans. Automat. Contr.*, vol. 45, pp. 588–591, Mar. 2000.

[13] F. Davoli and P. Maryni, "A two-level stochastic approximation for admission control and bandwidth allocation," *IEEE J. Select. Areas Commun.*, vol. 18, pp. 222–233, Feb. 2000.

[14] P. Glasserman, *Gradient Estimation via Perturbation Analysis*. Norwell, MA: Kluwer, 1991.

[15] K. Gokbayrak and C. G. Cassandras, "An online 'surrogate problem' methodology for stochastic discrete resource allocation problems," *J. Optimiz. Theory Appl.*, vol. 108, no. 2, pp. 349–376, 2001.

[16] W. B. Goug, Y. C. Ho, and W. Zhai, "Stochastic comparison algorithm for discrete optimization with estimation," in *Proc. 31st IEEE Conf. Decision Control*, 1992, pp. 795–800.

[17] Y. C. Ho and X. Cao, *Perturbation Analysis of Discrete Event Dynamic Systems*. Norwell, MA: Kluwer, 1991.

[18] Y. C. Ho, R. S. Sreenivas, and P. Vakili, "Ordinal optimization in DEDS," *J. Discrete Event Dyna. Syst.: Theory Appl.*, vol. 2, pp. 61–88, 1992.

[19] T. Ibaraki and N. Katoh, *Resource Allocation Problems: Algorithmic Approaches*. Cambridge, MA: MIT Press, 1988.

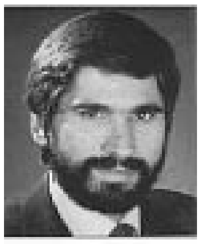
[20] S. Jordan and P. Varaiya, "Throughput in multiple service, multiple resource communication networks," *IEEE Trans. Commun.*, vol. 39, pp. 1216–1222, Aug. 1991.

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

[22] J. Kiefer and J. Wolfowitz, "Stochastic estimation of the maximum of a regression function," *Ann. Math. Stat.*, vol. 23, pp. 462–466, 1952.
[23] H. J. Kushner and G. G. Yin, *Stochastic Approximation Algorithms and Applications*. New York: Springer-Verlag, 1997.
[24] R. H. Kwong and A. Leon-Garcia, "Multiplexer performance for integrated line and packet-switched traffic," *Perform. Eval.*, vol. 4, pp. 81–91, 1984.
[25] P. Marbach, O. Mihatsch, and J. N. Tsitsiklis, "Call admission control and routing in integrated services networks using neuro-dynamic programming," *IEEE J. Select. Areas Commun.*, vol. 18, pp. 197–208, Feb. 2000.
[26] R. G. Parker and R. L. Rardin, *Discrete Optimization*. New York: Academic, 1988.
[27] H. Robbins and S. Monro, "A stochastic approximation method," *Ann. Math. Stat.*, vol. 22, pp. 400–407, 1951.
[28] K. W. Ross and D. H. K. Tsang, "Teletraffic engineering for product-form circuit-switched networks," *Adv. Appl. Prob.*, vol. 22, pp. 657–675, 1990.
[29] M. Schwartz, *Telecommunication Networks: Protocols, Modeling and Analysis*. Reading, MA: Addison-Wesley, 1987.
[30] L. Shi and S. Olafsson, "Nested partitions method for global optimization," *Oper. Res.*, vol. 48, pp. 390–407, 2000.
[31] J. E. Wieselthier, C. M. Barnhart, and A. Ephremides, "An approach to voice admission control in multihop wireless networks," in *Proc. INFOCOM*, 1993.
[32] D. Yan and H. Mukai, "Stochastic discrete optimization," *SIAM J. Control Optimiz.*, vol. 30, pp. 549–612, 1992.

Kagan Gokbayrak was born in Istanbul, Turkey, in 1972. He received the B.S. degrees in mathematics and in electrical engineering from Bogazici University, Istanbul, the M.S. degree in electrical and computer engineering from the University of Massachusetts, Amherst, and the Ph.D. degree in manufacturing engineering from Boston University, Boston, MA, in 1995, 1995, 1997, and 2001, respectively.

Since 2001, he has been working as a Network Planning Engineer at Genuity, Inc., Burlington, MA. His research interests are in the fields of systems, optimization, control, and operations research.



Christos G. Cassandras (S'82–M'82–SM'91–F'96) received the B.S. degree from Yale University, New Haven, CT, the M.S.E.E. degree from Stanford University, Stanford, CA, and the S.M. and Ph.D. degrees from Harvard University, Cambridge, MA, in 1977, 1978, 1979, and 1982, respectively.

From 1982 to 1984, he was with ITP Boston, Inc. where he worked on the design of automated manufacturing systems. From 1984 to 1996, he was a Faculty Member in the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst. Currently, he is Professor of Manufacturing Engineering and Professor of Electrical and Computer Engineering at Boston University, Boston, MA. He specializes in the areas of discrete-event and hybrid systems, stochastic optimization, and computer simulation, with applications to computer networks, manufacturing systems, transportation systems, and command-control systems. He has published over 180 papers in these areas, and two textbooks, one of which was awarded the 1999 Harold Chestnut Prize by the IFAC. He has served on several editorial boards and as Guest Editor for various journals, and is a member of Phi Beta Kappa and Tau Beta Pi.

Dr. Cassandras is currently Editor-in-Chief of the IEEE TRANSACTIONS ON AUTOMATIC CONTROL. He is a member of the IEEE Control Systems Society Board of Governors, and was awarded a 1991 Lilly Fellowship.