An Analytical Model for Adaptive Routing Networks

Gerald R. Ash, *Fellow, IEEE,* and BaoSheng D. Huang

Abstract—**Real-time network routing (RTNR) is a new adaptive routing method which replaced dynamic nonhierarchical routing (DNHR) in the AT&T network starting in 1991. RTNR is being introduced to extend dynamic routing to all new and existing services, and to increase network robustness. With RTNR, switches have a simple way of exchanging link status bit map information, thereby determining the availability and load conditions of the direct and all two-link paths to the destination. Link busy-idle status is exchanged between the network nodes using a bit map data exchange through the common channel signaling (CCS) network, and calls are set up where there is the most available capacity in the network. To date the analysis of RTNR networks has been limited to simulation models, in part because of the lack of analytical models for such networks. In this paper, an analytical model is developed for the AT&T network under RTNR, and is shown to provide good agreement with simulation models.**

The analytical model for RTNR networks uses an erlang fixed point method to solve the nonlinear equations describing dynamical network behavior. The equations include the link state probability, network flows, link arrival rates, adaptive trunk reservation level, and adaptive path selection depth. The link state model provides the aggregate link state probabilities through solution of the birth–death equations, and models the adaptive nature of trunk reservation. The network flow model provides a method to calculate the traffic flow using the least busy concept employed in RTNR, and also models the adaptive nature of the path selection depth. The analytical model addresses asymmetrical networks, and computational examples show the differences from the simulation model to be small. We also use the analytical model to examine key RTNR parameters over a range of values, and the model provides validation of some of the parameter values selected for initial RTNR implementation.

I. INTRODUCTION

DYNAMIC routing in telecommunications networks has been the subject of worldwide study and interest [1]–[28]. During the 1980s, dynamic nonhierarchical routing (DNHR) was fully deployed in the AT&T long distance network. DNHR has provided considerable benefits in improved performance quality and reduced costs [25], and has motivated the extension of dynamic routing to all classes-of-service. Real-time network routing (RTNR) is a new adaptive routing method which replaced DNHR in the AT&T network starting in 1991. RTNR provides the platform for a class-of-service dynamic routing strategy which supports dynamic routing on an integrated transport network for all new and existing voice, data, and wideband services. RTNR also provides a multiple ingress/egress routing arrangement to ensure reliability and flexibility for international and access networks. See [27] for a discussion of dynamic class-of-service routing and multiple ingress/egress routing.

With RTNR, switches have a simple way of exchanging link status information, and thereby determining the availability and load conditions of the direct and all two-link paths to the destination [27]. Link busy-idle status is exchanged between the network nodes using a bit map data exchange through the common channel signaling (CCS) network, and calls are set up where there is the most available capacity in the network. To date the analysis of RTNR networks has been limited to simulation models [27], in part because of the lack of analytical models for such networks. In this paper, an analytical model is developed for the AT&T network under RTNR, and is shown to provide good agreement with simulation models. We have used the model to examine parameters used in the design of RTNR networks, and the model provides validation of some of the parameter values selected for initial RTNR implementation.

Analytical modeling of dynamic routing networks has received considerable attention in the literature [29]–[37], with several of the models treating adaptive routing networks [31]–[34]. The analytical model for adaptive RTNR networks developed in this paper uses an erlang fixed point method [29]–[33] to solve the nonlinear equations describing network behavior, and makes two modeling assumptions: link independence and poisson overflow traffic. These two assumptions have been shown to give accurate results for modeling symmetrical nonhierarchical networks with constant trunk reservation strategies [31]–[33]. The model described in this paper addresses the asymmetrical network with aggregate link states, as well as adaptive trunk reservation and adaptive path selection depth. The equations solved by the model include the link state probability, which is computed iteratively by an analytical link state model, the network flows and link arrival rates, which are computed iteratively by an analytical network flow model, and the adaptive trunk reservation level and adaptive path selection depth, which are computed iteratively by an analytical periodic update model.

Section II briefly reviews adaptive routing systems to place the RTNR strategy in context. Section III discusses the RTNR strategy that is relevant to the analytical model. Section IV describes the analytical model in which an erlang fixed point method is used to solve the nonlinear equations for network behavior. Section V presents numerical results and comparisons with simulation models. Section VI summarizes the paper and identifies future work.

II. BRIEF REVIEW OF ADAPTIVE ROUTING SYSTEMS

A number of adaptive routing methods have been proposed which make real-time routing decisions. These include real-

---

Paper approved by E. G. Sable, the Editor for Communications Switching of the IEEE Communications Society. Manuscript received April 17, 1991; revised October 7, 1991.

The authors are with AT&T Bell Laboratories, Holmdel, NJ 07733.

IEEE Log Number 9211385.

time routing within DNHR, DNHR-based trunk status map routing, dynamically controlled routing, and dynamic alternative routing. DNHR [4]–[8], [25] uses a hybrid time varying and real-time routing system to respond to network load variations, and incorporates one and two-link path routing between originating and terminating switches. Time varying routing allows pre-specified routing patterns to change as frequently as every hour to respond to expected traffic patterns, and supplementary real-time routing, which searches for idle capacity on a call-by-call basis, if needed. The real-time routing method appends to each sequence of two-link paths engineered by the DNHR design algorithm [4] to meet the expected load, several additional two-link (real-time) paths to be used only when idle capacity is available. Dynamic trunk reservation is used to recognize idle network capacity on real-time paths. Access to trunks on a particular trunk group is allowed only when a specified number of trunks—the reservation level—is available which guarantees minimal interference with engineered traffic. A Network management operations system (NEMOS), under conditions of network stress, is capable of generating up to seven additional paths to append to the 14 preplanned DNHR engineered and real-time paths generated by the design. Network blocking performance is improved by both real-time routing and NEMOS reroutes under a variety of overload and failure conditions as demonstrated by network simulation studies [5] and network performance measurements [25].

Trunk status map routing (TSMR) is a proposed extension of the DNHR concept to a centralized trunk status map (TSM), which provides real-time routing decisions in the DNHR network [10]. The TSM concept involves having an update of the number of idle trunks in each DNHR trunk group sent to a network data vase every T seconds. In return, the TSM periodically sends the switches ordered routing sequences to be used until the next update in T seconds. These routing decisions are determined by the TSM every T seconds using the TSMR dynamic routing strategy. Dynamically controlled routing (DCR), a routing system developed by Bell Northern Research, also uses a central processor to track the busy-idle status of network trunks and determine the best alternate route choices based on status data every 10 seconds [3], [14], [20], [21], [28]. DCR is a feature of Northern Telecom's dynamic network controller system [14], and is now deployed in the Trans Canadian network. Dynamic alternative routing (DAR) was developed by British Telecom and uses a simple decentralized learning approach to adaptive routing [12], [13], [18]. When the direct trunk group is busy, the alternate two-link path last successfully used is used once again for the next call overflow from the direct trunk group. If the alternate path is busy the call is blocked and a new alternate path is selected at random. DAR is planned for the British Telecom network.

In comparison to these methods, RTNR has the advantage of being a simple decentralized strategy that attains or surpasses the performance gains demonstrated for the above approaches, but does not need the development of a central controller. It also avoids the risks associated with a central processor controlling the routing of the entire network. Hence RTNR is a promising strategy on which the AT&T dynamic class-of-service network has been fully implemented in 1991 [27].

III. RTNR STRATEGY FOR DYNAMIC NETWORKS

Switches using RTNR [27] first select the direct link between the originating switch, denoted here as OS_j, and the terminating switch, denoted here as TS_k. When no direct capacity is available, the originating switch checks the availability and load conditions of all of the two-link paths to TS_k on a per call basis. If any of these two-link paths is available, the call is set up over the least loaded two-link path. Traffic loads are dynamically balanced across links throughout the network to maximize the call throughput of the network.

In this paper, the term "link" refers to a trunk group connecting two switches, and the term "path" denotes the direct link or two links concatenated in tandem to connect two switches. A "route" refers to a collection of paths for routing calls between the originating switch and terminating switch. Finally, the terms "node" and "switch" are used interchangeably. In a dynamic class-of-service network, calls for a particular class-of-service are assumed to consume an average bandwidth equal to r^i, using a single unit of capacity denoted as one virtual trunk (VT). For example, each VT would have r^i equal to 64 kbps of bandwidth for voice calls or 64 kbps switched digital service calls, r^i equal to 384 kbps of bandwidth for 384 kbps switched digital service calls, and r^i equal to 1536 kbps of bandwidth for 1536 switched digital service calls. Here we focus on a single service network since that is the starting point for an analytical model for RTNR networks.

A. Exchanging Network Status Bit Maps

As illustrated in Fig. 1, an available two-link path from OS_j to TS_k goes through a via switch to which both OS_j and TS_k have idle capacity, that is, neither link is busy. An available two-link path is considered to be lightly loaded if the number of idle virtual trunks on both links exceeds a threshold level. In order to determine all of the switches in the network which satisfy this criterion, the originating switch sends a message to TS_k over the CCS network, requesting TS_k to send a list of the switches to which it has lightly loaded links. Upon receiving this list of switches from TS_k, the originating switch compares this list with its own list of switches to which it has lightly loaded links. Any switch that appears in both lists currently has lightly loaded links to both OS_j and TS_k, and therefore can be used as the via switch for a two-link connection for this call. In Fig. 1 there are two lightly loaded paths found between OS_j and TS_k.

The switch identifiers used in the switch list sent by TS_k must be recognized by the originating switch. Each switch in the network is assigned a unique network switch number (NSN); these NSNs are used as switch identifiers. In the example depicted in Fig. 1, there are five switches in a network which have been arbitrarily assigned NSNs. With these NSN assignments, a list of switches can be represented by a bit map that has a 1-bit entry for each NSN in the network. In Fig. 1, a "1" entry is made in the bit map for each NSN having a lightly

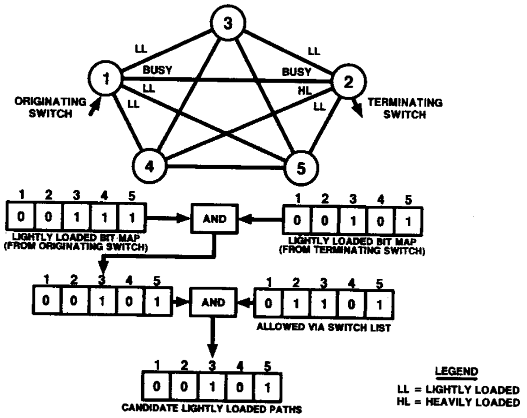


Fig. 1. Determining least loaded paths.

loaded link to TS_k. OS_j also maintains its own bit map listing each NSN having a lightly loaded link to OS_j. Using bit maps makes it very easy and efficient for the originating switch to find all lightly loaded two-link paths. The originating switch simply ANDs the bit map it receives from TS_k, which lists all of the lightly loaded links out of TS_k, with its own bit map to produce a new bit map which identifies all the via switches with lightly loaded links to both OS_j and TS_k. Bit maps also are a very compact way to store a list of switches. This is an important consideration since the list is sent in a CCS message. In the simplest case, only 16 bytes of data are needed to encode a given load state for a network with 128 switches.

Since some of the available two-link paths may not provide good voice transmission quality, network administrators can restrict path selection to the two-link paths which provide good transmission quality through use of another bit map, the allowed via switch list, which specifies the acceptable via switches from OS_j to TS_k. As illustrated in Fig. 1, ANDing this bit map with the bit map containing the via switches of all of the available two-link paths removes the via switches of paths with unacceptable transmission quality. When two or more available paths have the same load status, as in Fig. 1 where two lightly loaded paths are found, the originating switch picks one of these paths to use for the call. To pick a path for this particular call to TS_k, the originating switch starts a circular search through the bit map list of paths with the same load status, beginning with the entry immediately following the via switch it last used for a call to TS_k.

B. Determining Link Load States

In order to dynamically balance traffic loads across the network, it is necessary to know the actual load conditions of links throughout the network, and use two-link paths that have the least loaded links. RTNR uses six discrete load states for links; lightly loaded 1 (LL1), LL2, LL3, heavily loaded (HL), reserved (R), and all busy (B). The number of idle virtual trunks on a link is compared with the load state thresholds for the link to determine the load condition of the link. This determination is made every time a virtual trunk on the link is either seized or released. The terminating switch identifies all the switches to which it has idle capacity, and the load condition of the links to each of these switches, in the reply message sent to the originating switch. The terminating switch does this by sending multiple bit map lists in this reply message; a bit map list of the switches to which it has a LL1 link, a bit map list of the switches to which it has a LL2 (or less loaded) link, etc. The originating switch maintains similar bit map lists of the switches to which it has idle capacity, ordered by the load condition of the links to these switches. The originating switch first compares the list of switches to which it has a LL1 link with the list of switches to which the terminating switch has a LL1 link to determine if any switch appears in both lists. If not, the originating switch successively compares the other load status lists to find the via switch with the least loaded available links.

The load state thresholds used for a particular link are based on the current estimates of four quantities maintained by switch OS_j, for each switch TS_k:

1) the current number of calls-in-progress CIP_k which is the number of active calls from OS_j to TS_k. Calls-in-progress is measured on-line by the switch and includes calls completed on both direct and two-link connections.
2) the current node-to-node blocking level NN_k which is the rate of blocking for calls from OS_j to TS_k. Here

   $$NN_k = OV_k / PC_k$$

   where OV_k is the overflow count and PC_k is the peg count to destination TS_k over the periodic update interval of 3 min. Blocking level is estimated on-line by the switch.
3) the offered traffic load TL_k to each of the other switches in the network. This approximation is based on the number of calls-in-progress CIP_k and the call blocking to each switch.
4) $VTtraf_k^i$, which is the number of virtual trunks required to meet the grade-of-service (GOS) objective for the current offered traffic load TL_k from OS_j to TS_k. Here $VTtraf_k$ is given by

   $$VTtraf_k = 1.1 \times TL_k$$

   $VTtraf_k^i$ is estimated on-line by the switch and can exceed the total number of direct virtual trunks between OS_j and TS_k; this condition means that virtual trunks from two-link alternate paths are needed to meet the GOS objective for the current offered traffic load.

The load state thresholds for the lightly loaded and heavily loaded states are set to fixed percentages of the $VTtraf_k$ estimate. As such, the load state thresholds rise as the $VTtraf_k$ estimate to that switch increases. Higher load state thresholds reduce the chances that the link is used for two-link connections for calls to or from other switches; this enables the link to carry more direct traffic and therefore better handle the call load between the switches connected by the link. The reserved state threshold is based on the reservation level R_k calculated on each link, which in turn is based on the blocking level NN_k, as shown in Table I.

TABLE I
Trunk Reservation Level

NN_k Blk. Threshold	Reservation Level	R_k (VTs)
[0, .01]	0	0
(.01, .05]	1	$.05 \times VTtraf_k$
(.05, .15]	2	$.1 \times VTtraf_k$
(.15, .5]	3	$.15 \times VTtraf_k$
(.5, 1]	4	$.2 \times VTtraf_k$

Note that appropriate upper and lower bounds are placed on the value of R_k determined through simulation analysis. If the number of blocked calls to TS_k exceeds the NN_k thresholds, which is computed each periodic 3 min update interval, then the switch immediately triggers that next level of trunk reservation. For example, if the current trunk reservation level is level 0, as given in Table I, and if the number of blocked calls reaches the number needed for trunk reservation level 1, then level 1 is triggered immediately. In this way reservation protection is applied immediately following a load surge or facility failure, which greatly aids in improving the switch and network response to load variation and failure.

The reserved virtual trunks $Rtraf_k$ are at most R_k, as defined above, and are bounded above by $VTtraf_k - CIP_k$, as follows

$$Rtraf_k = \min[R_k, \max(0, VTtraf_k - CIP_k)] .$$

Fig. 2 summarizes the above mechanisms for RTNR trunk reservation. Each switch continuously checks the blocking rate for call attempts to each of the other switches in the network. If the originating switch finds that the GOS objective for its call attempts to TS_k is not met, where the GOS objective is taken as .01 for illustration in Fig. 2, the originating switch sets $Rtraf_k$. As shown in Fig. 2, the higher the node-to-node blocking, the greater is the number of virtual trunks reserved $Rtraf_k$ for direct traffic versus alternate routed traffic. $Rtraf_k$ is set to the difference between the $VTtraf_k$ estimate and the current number of calls-in-progress CIP_k to TS_k, with an upper bound equal to R_k, given in Table I. This difference is the number of additional calls from the originating switch to TS_k that need to be completed to reach the current $VTtraf_k$ estimate. The originating switch adjusts $Rtraf_k$ as the number of completed calls to TS_k (CIP_k) changes. Every time the originating switch completes a new call attempt to TS_k, either over the direct link or a two-link path, the originating switch checks to see whether the reserved state threshold should be decremented for the link to TS_k. Likewise, when a call to TS_k disconnects, the originating switch checks to see if the $Rtraf_k$ threshold should be incremented. By following these actions, the originating switch turns the reservation control for this link off and on as the number of calls-in-progress to TS_k oscillates around the $VTtraf_k$ estimate. Also note that RTNR allows for a number n of trunks continuously reserved. In this case, the quantity n is added to the right-hand-side of the above expression for $Rtraf_k$. Illustrative values of the VT thresholds to determine the different states in the 6-state model are summarized in Table II.

TABLE II
Link Load State Thresholds

Threshold Name	Threshold Level (VTs)
$TK1_k$	$.05 \times VTtraf_k$
$TK2_k$	$.1 \times VTtraf_k$
$TK3_k$	$.2 \times VTtraf_k$

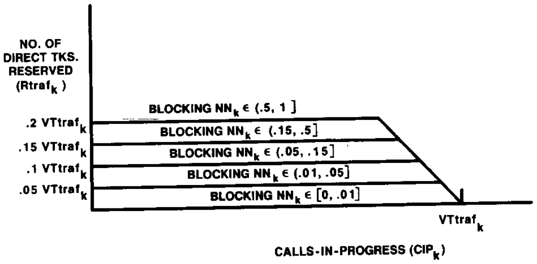


Fig. 2. Operation of RTNR trunk reservation.

TABLE III
Definition of Link Load States

Idle Link Virtual Trunks	Load State
$ILVT_k = 0$	busy
$1 \le ILVT_k \le Rtraf_k$	reserved
$Rtraf_k < ILVT_k \le TK1_k + Rtraf_k$	HL
$TK1_k + Rtraf_k < ILVT_k \le TK2_k + Rtraf_k$	LL3
$TK2_k + Rtraf_k < ILVT_k \le TK3_k + Rtraf_k$	LL2
$TK3_k + Rtraf_k < ILVT_k$	LL1

TABLE IV
Path Selection Order

Paths Selected	Path Sequence
direct	1
via LL1	2
via LL2	3
via LL3	4
via HL	5 (controlled use)
via reserved	6 (controlled use)

Here again, appropriate minimum and maximum values are applied to the load state thresholds as determined through simulation analysis. If the idle virtual trunks for the link from OS_j to TS_k is $ILVT_k$, then the link load states are given in Table III.

Therefore the reservation level and state boundary thresholds are proportional to the estimated offered-traffic-load level, which means that the number of virtual trunks reserved and the number of idle virtual trunks required to constitute a lightly loaded link rises and falls with the traffic load, as intuitively it should. Under normal nonblocking network conditions, all traffic fully shares all available capacity. As discussed above, when blocking occurs RTNR trunk reservation is activated to prohibit alternate routed traffic from seizing direct link capacity designed. This is accomplished by setting the reserved state threshold equal to the number of virtual trunks reserved $Rtraf_k$ for this link, and placing the link in the reserved state

TABLE V
PATH SELECTION DEPTH

	Traffic Conditions				Depth Factor	
Item	$VTtraf_k$ (VT's)	Dir. Tks. (VTs)	Tot. Off. Blk. (TO_j)	Node-to-Node Blk. (NN_k)	Dhl_k	Dr_k
A	> 15	> 0	[0, .03]	[0, .01]	0	0
				(.01, 1]	1.0	0
B	≤ 15	> 0	[0, .03]	[0, .01]	1.0	0
				(.01, 1]	1.0	1.0
C	all	> 0	(.03, .1]	[0, .5]	0	0
				(.5, 1]	1.0	0
D	all	> 0	(.1, 1]	[0, .5]	0	0
				(.5, 1]	0	0
E	all	$= 0$	[0, .03]	[0, 1]	8.0	8.0
F	all	$= 0$	(.03, 1]	[0, 1]	8.0	0

when the number of idle virtual trunks in the link is less than this reserved state threshold. As virtual trunks in the link become idle when old calls disconnect, the capacity is reserved for new call attempts between the originating switch and TS_k. The lightly loaded and heavily loaded state thresholds for the link are adjusted upward from their base values by the amount of the reserved state threshold as given in Table III. Once the originating switch finds that the GOS objective for its call attempts to TS_k is again being met, the reserved state threshold for the link between these two switches is reset to zero, and the lightly loaded and heavily loaded state thresholds are adjusted accordingly.

C. Allocating Traffic to Direct and Two-Link Via Paths

At this point we discuss the process of selecting the most desired path through the network from all the available candidates, based on network status information discussed in the above sections. The rules for selecting direct and two-link via paths for a call are depicted in Table IV.

The path selection is governed by the availability of the direct link $VTtraf_k$ total office blocking TO_j, and node-to-node blocking NN_k. The path sequence consists of the direct path, if it exists, lightly loaded 1 via paths, lightly loaded 2 via paths, lightly loaded 3 via paths, heavily loaded via paths, and reserved via paths. Two thresholds, Dhl_k and Dr_k, are defined to be associated with the path selection depth. These thresholds allow a "controlled use" of the associated path, as follows. If

$$CIP_k < Dhl_k \times VTtraf_k$$

then calls to TS_k can be routed using heavily loaded via paths. Also if

$$CIP_k < Dr_k \times VTtraf_k$$

then calls to TS_k can use reserved via paths. Hence, the factors Dhl_k and Dr_k are factors controlling the depth of selection for a via path on a given call.

Hence, the path sequence selection as well as the depth factors Dhl_k and Dr_k are controlled by node-to-node and total office blocking thresholds, the existence of direct link capacity, and the level of offered node-to-node (direct) traffic $VTtraf_k$, as illustrated in Table V.

The selection of paths is given by the congestion state based on the total-office blocking TO_j for node j over a periodic 3 min. update interval, node-to-node blocking NN_k to destinating TS_k over the 3 min. update interval, and the existence of direct link capacity or not. In general, greater path selection depth is allowed if blocking is detected to TS_k, since more alternate route choices serves to reduce the blocking to TS_k. This greater selection depth is inhibited, however, if the total office blocking reaches a high level, indicating that a general overload condition exists, in which case it is more advantageous to reduce alternate routing. Also, if there is no direct link capacity, as in items E and F in Table V, or the node-to-node traffic is small, say with $VTtraf_k$ less than 15 VT's, as in item B in Table V, then the selection depth again is increased since trunk reservation becomes ineffective or even impossible, and greater dependence on alternate routing is needed to meet network blocking objectives. The use of heavily loaded and reserved links for two-link connections is relaxed for calls between two switches which are not connected by a direct link, or when the link capacity is greatly reduced by a network failure event such as a fiber cut. In this manner the blocking performance can be controlled by the depth of access to via paths in the heavily loaded and reserved state.

The above mechanisms are illustrated in Fig. 3, where we focus on the controlled use of heavily loaded paths. The objective of accessing heavily loaded via paths is to increase the call completions from the originating switch to TS_k such that the required virtual trunk capacity $VTtraf_k$ needed to meet the GOS objective is provided. As shown in Fig. 3, lightly loaded RTNR paths can always be selected. However, heavily loaded RTNR paths are selected only under the conditions that A) a NN_k blocking rate in excess of the GOS objective of .01 is detected, and b) the calls-in-progress CIP_k is less than $VTtraf_k$. When calls-in-progress CIP_k exceeds $VTtraf_k$, then heavily loaded paths should not be required to meet the GOS objective, and thereby heavily loaded capacity can be used by other traffic still in need of additional capacity. By controlling the use of heavily loaded two-link connections, the completion rate for calls between switches that have not been meeting their GOS objective is improved, while possibly increasing the blocking rate for calls between pairs of switches that are within their GOS

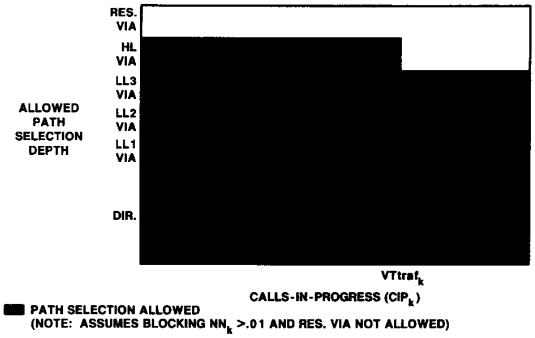


Fig. 3. Operation of RTNR path selection.

objective. Links in the reserved state are normally used only for direct traffic, but may be used for via routing when higher priority calls are placed under heavy network congestion. In this manner the blocking performance can be controlled by the depth of access to via paths in the heavily loaded and reserved state. Simulations have shown that the values of these parameters illustrated in Table V provide a good allocation of available network capacity for node pairs experiencing blocking.

D. Periodic Update of Adaptive Trunk Reservations & Adaptive Path Selection Depth

The update process is executed each periodic update interval of three minutes duration to determine the node-to-node blocking level, $NN_k = OV_k/PC_k$, node-to-node reservation level, R_k, node-to-node estimated traffic-load, TL_k, virtual trunk requirement, $VTtraf_k$, node-to-node path sequence with the Dhl_k and Dr_k values, and total office blocking level TO_j. The virtual trunks required to complete the estimated traffic within the grade-of-service objective are estimated by the formula given above, that is

$$VTtraf_k = 1.1 \times TL_k$$

where TL_k is the estimated offered traffic load, for which an approximation is incorporated for the blocking term. The reservation levels R_k and the path sequence for each node-to-node pair together with the Dhl_k and Dr_k values are determined by applying NN_k, $VTtraf_k$, and TO_j to Tables I and V, which are presented above.

IV. Analytical Model for RTNR Networks

The analytical model computes network flows as well as node-to-node blocking probabilities for networks with RTNR. An assumption of the analytical model is that the stationary behavior of RTNR networks can be computed based on the stationary, independent link state probabilities, and on via switch selection probabilities, which are based on the link state probabilities and node-to-node blocking probabilities. An iterative erlang fixed point approach [29]–[33] is used and all overflow traffic is assumed to be poisson.

Three steps are performed iteratively by the algorithm. The link state probability model derives the link aggregate state probabilities from the link arrival rates, the link arrival rates are the outcome of the network flow model. The network flow model uses the link aggregate state probabilities to derive the path state probabilities and the route state probabilities. With these state probabilities, the flow model determines the route flows, path flows, path arrival rates, and the link arrival rates, where the latter are used in the link state probability model to compute the link state probabilities. Finally the adaptive trunk reservation and adaptive path selection depth model computes the node-to-node blocking probability distribution, and from that models the path selection depth and the link reservation levels. A model of calls-in-progress is used to determine the probability that reservation and path selection depth are enabled. Fig. 4 illustrates the above three steps in the analytical model for RTNR networks. The convergence criterion is the sum of the squared differences in link state probability levels from the previous to the current iteration, summed over all links in the network. Note that the Erlang fixed point method guarantees the existence of a solution, but may not converge to a unique solution if trunk reservation is not sufficient [35], [36]. Our numerical experience shows that, when RTNR trunk reservation is applied, the approximation method converges quickly to the fixed point solution.

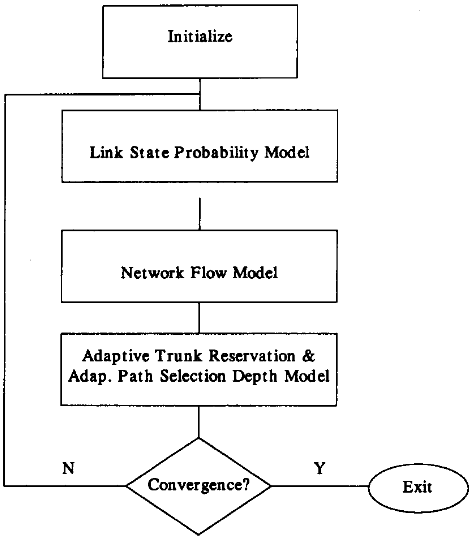


Fig. 4. Analytical model for RTNR networks: algorithm steps.

A. Link State Probability Model

As discussed in Section II, RTNR uses six discrete load states for links; lightly loaded 1 (LL1), LL2, LL3, heavily loaded (HL), reserved (R), and all busy (B). The number of idle virtual trunks on the link is compared with the load state thresholds for the link to determine the load condition of the link. This determination is made every time a virtual trunk on the link is either seized or released.

The boundary of the six states on each link is defined in Tables I–III, and is illustrated in Fig. 5. As discussed in Section II, the link state thresholds depend on the following:

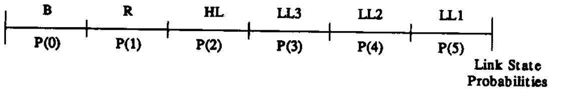


Fig. 5. Definition of link state probabilities.

1) node-to-node blocking level (NN_k) for node pair k, which determines the reservation level, and
2) node-to-node traffic load estimate TL_k and estimated virtual trunk requirements $VTtraf_k$ to meet the GOS requirement for node pair k.

Then, the link aggregate state probability for state S is defined as

$$P_k(S) = \sum_{i=L(S)_k}^{U(s)_k} P_{ki} \tag{1}$$

where $S = 0, 1, 2, 3, 4, 5$ and $L(S)_k$ and $U(S)_k$ are the lower and upper bounds of the aggregate link state S, for link k, as determined from Tables I–III. As illustrated in Fig. 5, $S = 0$ denotes the busy state B, $S = 1$ denotes the reserved state R, $S = 2$ denotes the heavily loaded state HL, $S = 3$ denotes the lightly loaded 3 state LL3, $S = 4$ denotes the lightly loaded 2 state LL2, and $S = 5$ denotes the lightly loaded 1 state LL1. The quantity P_{ki} denotes the probability of i busy circuits on link k, which is computed from the birth–death equations for the link state. These equations are given as follows. The call termination rate on link k is

$$d_{ki} = i\mu$$

for $i = 1, 2, \ldots, C_k$ where C_k is the number of virtual trunks on link k and the call holding time is exponentially distributed with mean $1/\mu$. The call arrival rate is

$$b_{ki} = (D_k + A_{ki})\mu$$

for $i = 0, 1, \ldots, C_k - 1$ where D_k is the direct offered load for link k and A_{ki} is the overflow load offered to link k when it has i busy circuits. Note that time indexing is omitted here for simplicity. Then the balance equation

$$b_{ki}P_{ki} = d_{ki+1}P_{ki+1}$$

is solved for each link k.

To avoid numerical instability, the following procedure is used to compute the link state probability. This approach has been used by Mitra [33] in studies of symmetric networks. First, the initial point n^* is set to the integer part of $\min[D_k, C_k]$. Here, the choice of the initial point n^* will assure that the probability density function for the link state is near the maximum value, and underflow problems are avoided. With the initial point n^* for forward and backward computation, the algorithm below provides the P_{ki} values as follows.

1) Initial step: Let $P_{kn^*} = 1$, and SUM $= 1$, be the initial values for computing link state probabilities.
2) Forward computation: For $i = n^* + 1, \ldots, C_k$, let $P_{ki} = P_{ki-1} \times b_{ki-1}/d_{ki}$, and SUM $=$ SUM $+P_{ki}$,
3) Backward computation: For $i = n^* - 1, \ldots, 0$, let $P_{ki} = P_{ki+1} \times d_{ki+1}/b_{ki}$, and SUM $=$ SUM $+P_{ki}$,
4) Normalization computation: For $i = 0, \ldots, C_k, P_{ki} = P_{ki}/\text{SUM}$.

The aggregate state probabilities $P_k(S)$ are then computed from the P_{ki} values, as given in equation (1).

B. Network Flow Model

This section describes the network flow model for the analytical model for RTNR networks. We first describe the procedures to compute the path state probabilities and the route state probabilities. We then compute link offered loads for each state S due to the overflow traffic. For a path consisting of two links, by the independence assumption, the path state probabilities $P_{PH}(S)$ are given by

$$\begin{aligned}
P_{PH}(S) &= \Pr(\text{path is in state } S)\\
&= \Pr(\text{Both links are in state } S)\\
&\quad + \Pr(\text{1st link L1 is in state } S)\\
&\quad \times \Pr(\text{2nd link L2 is in state better than } S)\\
&\quad + \Pr(\text{2nd link L2 is in state } S)\\
&\quad \times \Pr(\text{1st link L1 is in state better than } S)\\
&= \Pr(S_{L1} = S)\Pr(S_{L2} = S) + \Pr(S_{L1} = S)\\
&\quad \times \Pr(S_{L2} > S) + \Pr(S_{L1} > S)\Pr(S_{L2} = S)\\
&= \Pr(S_{L1} = S)\Pr(S_{L2} \geq S) + \Pr(S_{L1} > S)\\
&\quad \times \Pr(S_{L2} = S),\\
S &= 0, 1, 2, 3, 4, 5.
\end{aligned}$$

Here "state better than S" means a state with greater idle capacity than state S; for example, state LL1 has more idle capacity, or is "better than," state LL2. Fig. 6 illustrates the computation of the path states probabilities from the link state probabilities. With the path probabilities defined, the route state probability is defined as

$$\begin{aligned}
P_{RT}(S) &= \Pr(\text{route is in state } S)\\
&= \Pr(\text{at least one of its paths is in state } S\\
&\qquad \text{and no path is in a state better than } S).
\end{aligned}$$

To derive the route state probability, let $G_{PH}(S)$ be the cumulative distribution function of the path state probabilities, that is, for path p (PHp),

$$G_{PHp}(S) = \Pr(S_{PHp} \leq S).$$

Then the route state probabilities $R_{RT}(S)$ are given as follows:

$$\begin{aligned}
P_{RT}(S) &= \Pr(\text{route is in state } S)\\
&= \begin{cases} \prod\limits_{PHp} \Pr(S_{PHp} = 0), & S = 0\\ \prod\limits_{PHp} G_{PHp}(S) - \prod\limits_{PHp} G_{PHp}(S-1), & S = 1, 2, 3, 4, 5 \end{cases}
\end{aligned}$$

where the product is over all paths in the route for node pair k. Fig. 7 illustrates the computation of $P_{RT}(S)$.

In Section II we described the call by call adaptive routing procedure for RTNR. Here, in the analytical model, the steady state behavior of RTNR is modeled to capture this adaptive routing procedure in which the direct overflow is allocated to

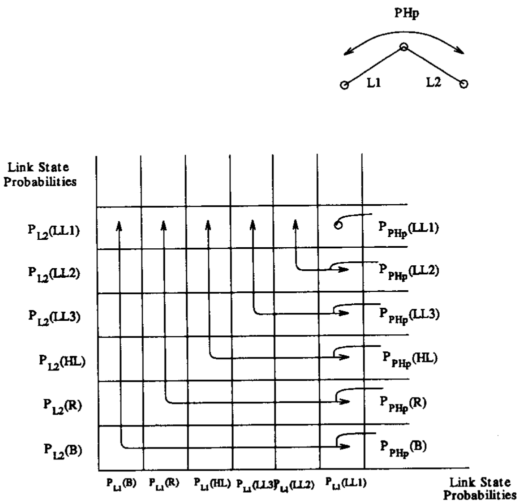


Fig. 6. Computation of path state probabilities.

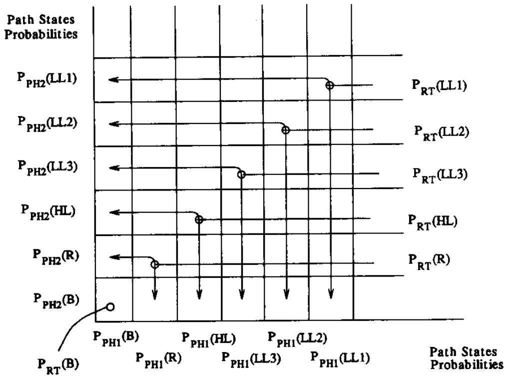


Fig. 7. Computation of route state probabilities.

via paths according to the route state probabilities, path state probabilities, and link state probabilities. That is, the via path flow is based on the following:

1) overflow load α_k from link k,
2) route state probabilities $P_{RT}(S)$,
3) path state probabilities $P_{PH}(S)$, and
4) path sequence depth $DEPTH_{RTk}(S)$ for the route for node pair k in state S.

The overflow load α_k from link k is given by

$$\alpha_k = D_k \times P_k(S = 0)$$

where D_k is the direct offered load for link k and $P_k(S = 0)$ is the probability of the busy state for link k. Let $DEPTH_{RTk}(S)$ equal the probability that the route for node pair k is allowed, according to Tables IV and V, to overflow load to paths in state S. This quantity is computed in the Adaptive Trunk Reservation and Adaptive Path Selection Depth Model described below. Then the flow carried by the route for node pair k, in state S, is given by

$$f_{RTk}(S) = \alpha_k \times P_{RTk}(S) \times DEPTH_{RTk}(S).$$

Now define the probability sum for state S for paths in route k to be

$$PSUM_{RTk}(S) = \sum_{p=1}^{P_k} P_{PHp}(S)$$

where the summation is over all paths P_k in the route for node pair k. Then the proportion of this flow carried on path p is approximated by the ratio of the probability that path p is in state S relative to the probability sum that the other paths in route k are in state S. That is

$$f_{PHp}(S) = f_{RTk}(S) \times P_{PHp}(S)/PSUM_{RTk}(S).$$

Then the arrival rate to path p on the route for node pair k when path p is in state S is given by the path carried flow divided by the probability the path is in state S, that is

$$y_{PHp}(S) = f_{PHp}(S)/P_{PHp}(S).$$

This expression is similar to that obtained by Wong and Yum [31], [32] for a symmetric network model. Following Wong's and Yum's approach, we let link L1 of path PHp be in state S_{L1}, and link L2 be in state S_{L2}. Then the state of path PHp is

$$S_{PHp} = \min(S_{L1}, S_{L2})$$

and the load offered to link L1 on path PHp, given that link L1 is in state S_{L1} and link L2 is in state S_{L2}, is given by

$$a_{L1}^{PHp}(S_{L1}) = \sum_{S_{L2}=0}^{5} y_{PHp}[\min(S_{L1}, S_{L2})] \times P_{L2}(S_{L2}).$$

Then the arrival rate to link 1 from all overflow loads in the network, given that link L1 is in state S_{L1}, is given by

$$A_{L1}(S_{L1}) = \sum_{PHp} a_{L1}^{PHp}(S_{L1})$$

where the summation is over all paths PHp which contain link L1. These link arrival rates are used in the link state probability model to recompute the link state probabilities, as described above.

C. Adaptive Trunk Reservation & Adaptive Path Selection Depth Model

As discussed in Section II, RTNR uses an adaptive trunk reservation algorithm in which the number of reserved trunks depends on the node-to-node blocking detected over a periodic update interval of 3 mins. In this manner, traffic attempting to alternate route over the direct link of a triggered node pair is subject to trunk reservation, and the direct traffic is favored for the triggered node pair.

A binomial distribution is used to approximate the node-to-node blocking in the 3 min. periodic update interval. That is, the expected 3-minute peg counts for node pair k are given by

$$\overline{PC}_k = TL_k \times 180 \times \mu$$

where $1/\mu$ is the mean holding time, in seconds. If the node-to-node blocking probability is NN_k for node pair k, then the overflow counts are approximately binomially distributed in the 3 min. update interval with mean overflow count equal to

$$\overline{OV}_k = \overline{PC}_k \times NN_k\,.$$

Here the stationary value of NN_k is computed from the results of the network flow model. Hence, with this estimate for $\overline{PC}_k$ and the binomial distribution of OV_k, the probability of each reservation level $i, P_k(RLi)$ for each node pair k, as given in Table I, is computed.

Recall that reservation is turned off when the calls-in-progress for node pair k, CIP_k, exceed $VTtraf_k$. These dynamics of reservation turn-off are modeled by computing the probability that reservation is turned on for node pair k, Pon_k. Pon_k is computed as the total probability mass for the condition that the calls-in-progress is less than or equal to $VTtraf_k$, that is when $CIP \leq VTtraf_k$. Pon_k is computed in this way under the assumptions that the calls-in-progress is poisson distributed, with variance equal to the mean, and that the reservation turn-off process is independent of the reservation level triggering process.

With these estimates for $P_k(RLi)$ and Pon_k, we compute the mean reservation value according to

$$\overline{R}_k = \sum_{i=0}^{4} P_k(RLi) \times i \times .05 \times VTtraf_k \times Pon_k\,.$$

We next compute the path sequence depth probabilities, $DEPTH_{RTk}(S)$, for the route associated with node pair k, according to Tables IV and V. We compute the total office blocking for node j, TO_j, as

$$TO_j = \sum OV_k \Big/ \sum PC_k$$

where the summations are over all node pairs k originating at node j. Then, with the probability distribution of each node-to-node blocking and reservation level determined, as described above, the probability is computed that the route uses the depth parameters given in Items A, B, D, E, and F of Table V. Here again, because access to paths in the heavily loaded state and reserved states are regulated by a turn-off procedure when $CIP_k > VTtraf_k$, as described in Section III-C, the depth probabilities for the reserved state $DEPTH_{RTk}(S = 1)$, and for the heavily loaded state $DEPTH_{RTk}(S = 2)$, as obtained from Table V, are multiplied by Pon_k. Here the computation of Pon_k is the same as described above.

V. NUMERICAL RESULTS

In this section we give numerical examples of the use of the analytical model for RTNR networks. A call-by-call simulation model that has been used for earlier RTNR studies [27] is used here for comparison.

We used a six node network model given by Mitra and Seery in [38]. Table VI gives the trunk and load information for the six node model.

Results are given in Table VII for the engineered load, and for general overloads of 10, 20, 30, and 40%, respectively, in which we compare total network blocking for the simulation model and for the analytical model. The node-to-node blocking comparisons are given in Table VIII for the 40% overload case. The node-to-node blocking comparisons are given in Table IX for the 30% overload case. It is noted that good agreement is achieved between the analytical model and simulation model, and that node-to-node blocking estimates are usually within 1% of each other. In addition, the estimated blocking in the analytical model is almost always in the range of variability of the simulated blocking. We note also that good agreement is reported by Mitra in [33] between analytical methods and simulation, for the case of symmetric least-busy-alternative routing networks with fixed trunk reservation.

TABLE VI
SIX NODE NETWORK DATA

orig. switch	term. switch	trunks	load (erlangs)
1	2	36	27.5
1	3	24	7.0
1	4	324	257.8
1	5	48	20.5
1	6	48	29.1
2	3	96	25.1
2	4	96	101.6
2	5	108	76.8
2	6	96	82.6
3	4	12	11.9
3	5	48	6.9
3	6	24	13.2
4	5	192	79.4
4	6	84	83.0
5	6	336	127.1

TABLE VII
TOTAL NETWORK BLOCKING COMPARISON FOR SIX NODE NETWORK

overload factor	simulation model	analytical model
1.0	0.000	0.000
1.1	0.001	0.000
1.2	0.005	0.004
1.3	0.025	0.025
1.4	0.058	0.059

We have used the analytical model to provide a preliminary examination of the performance of the six node network under variation of some of the RTNR parameters. First, we examined the reservation level threshold factor, which multiplies $VTtraf_k$ to yield the reservation level threshold, and which is given as 0.05 for reservation level 1 in Table I. We examined a range of values for the cases of 20, 30, and 40% overload in the six node network. The results are given in Table X.

These results indicate that the exact level of trunk reservation does not strongly influence network performance, although reservation is extremely important for the stabilization of network throughput. These results from the analytical model are consistent with the results of the simulation model for various reservation level factors.

We also examined the link load state threshold factor, which multiplies $VTtraf_k$ to yield the load state threshold, and which is given as 0.05 for $TK1_k$ in Table II. We examined a range of values for the cases of 20, 30, and 40% overload in the six node network. The results are given in Table XI.

TABLE VIII
NODE TO NODE BLOCKING COMPARISON FOR SIX NODE NETWORK (40% OVERLOAD)

orig. switch	term. switch	trunks	load (erlangs)	simulation model	analytical model
1	2	36	27.5	0.046	0.042
1	3	24	7.0	0.016	0.008
1	4	324	257.8	0.066	0.071
1	5	48	20.5	0.013	0.012
1	6	48	29.1	0.033	0.042
2	3	96	25.1	0.001	0.000
2	4	96	101.6	0.222	0.229
2	5	108	76.8	0.021	0.009
2	6	96	82.6	0.091	0.093
3	4	12	11.9	0.067	0.046
3	5	48	6.9	0.003	0.000
3	6	24	13.2	0.012	0.004
4	5	192	79.4	0.010	0.008
4	6	84	83.0	0.031	0.032
5	6	336	127.1	0.000	0.000

TABLE IX
NODE TO NODE BLOCKING COMPARISON FOR SIX NODE NETWORK (30% OVERLOAD)

orig. switch	term. switch	trunks	load (erlangs)	simulation model	analytical model
1	2	36	27.5	0.012	0.017
1	3	24	7.0	0.002	0.001
1	4	324	257.8	0.027	0.017
1	5	48	20.5	0.003	0.001
1	6	48	29.1	0.013	0.020
2	3	96	25.1	0.000	0.000
2	4	96	101.6	0.114	0.137
2	5	108	76.8	0.003	0.000
2	6	96	82.6	0.036	0.044
3	4	12	11.9	0.017	0.001
3	5	48	6.9	0.001	0.000
3	6	24	13.2	0.001	0.000
4	5	192	79.4	0.008	0.000
4	6	84	83.0	0.014	0.001
5	6	336	127.1	0.000	0.000

TABLE X
TOTAL NETWORK BLOCKING VERSUS RESERVATION LEVEL THRESHOLD R_k FACTOR

res. level thres. factor (R_k = factor × $VTtraf_k$)	20% overload	30% overload	40% overload
.02	0.0038	0.0250	0.0606
.03	0.0038	0.0250	0.0602
.04	0.0038	0.0250	0.0598
.05	0.0038	0.0248	0.0595
.06	0.0038	0.0245	0.0594
.07	0.0038	0.0245	0.0593
.08	0.0038	0.0245	0.0592
2.0	0.0038	0.0242	0.0603

These results indicate that the load state threshold factor influences network performance rather strongly. The results in Table XI suggest that a threshold factor of .05 is a good choice across a range of conditions, which is the value implemented for RTNR operation in the AT&T network [27].

Finally, we examined different $VTtraf_k$ factors, which multiplies the node-to-node traffic load estimate TL_k to yield $VTtraf_k$, and which is given as 1.1 in Section III-B. We examined a range of values for the cases of 20, 30, and 40% overload in the six node model. The results are given in Table XII. These results indicate that the $VTtraf_k$ factor has a substantial influence on network performance. The results in Table XII suggest that a factor of 1.1 is a good choice across a range of conditions, which is the value implemented for RTNR operation in the AT&T network [27]. Hence the analytical model for RTNR networks provides validation of some of the parameter values selected for initial RTNR implementation.

VI. SUMMARY AND FUTURE WORK

This paper discusses an analytical model for RTNR networks which uses an erlang fixed point approximation to solve the nonlinear equations describing network dynamical behavior. The link state model provides the aggregate link state

TABLE XI
TOTAL NETWORK BLOCKING VERSUS LOAD STATE THRESHOLD $TK1_k$ FACTOR

load state thres. factor ($TK1_k$ = factor $\times$ $VTtraf_k$)	20% overload	30% overload	40% overload
.02	0.0038	0.0253	0.0599
.03	0.0039	0.0252	0.0597
.04	0.0039	0.0250	0.0595
.05	0.0038	0.0248	0.0595
.06	0.0048	0.0263	0.0605
.07	0.0048	0.0264	0.0615
.08	0.0055	0.0277	0.0615
2.0	0.0226	0.0456	0.0783

TABLE XII
TOTAL NETWORK BLOCKING VERSUS VIRTUAL TRUNK REQUIREMENTS $VTtraf_k$ FACTOR

$VTtraf_k$ factor ($VTtraf_k$ = factor $\times$ TL_k)	20% overload	30% overload	40% overload
.8	0.0050	0.0270	0.0646
.9	0.0045	0.0265	0.0615
1.0	0.0041	0.0252	0.0602
1.1	0.0038	0.0248	0.0595
1.2	0.0036	0.0247	0.0601
1.3	0.0035	0.0258	0.0601
1.4	0.0041	0.0258	0.0601
2.0	0.0078	0.0274	0.0620

probabilities. The link state probability computation solves the birth–death equations and models the adaptive nature of trunk reservation. The network flow model provides a method to calculate the traffic flow using the least busy concept employed in RTNR, and also models the adaptive nature of the path selection depth. The analytical model output for nonsymmetrical networks is provided and the differences from a simulation model are presented. Good agreement is found between the analytical model and the simulation model.

We also used the analytical model to examine key RTNR parameters over a range of values. The model provides validation of some of the parameter values selected for initial RTNR implementation. Further use of the analytical model to examine RTNR parameters and for RTNR network design are warranted. Future work also could include models for the dynamic class-of-service aspect of RTNR networks, in which bandwidth sharing is performed among services and a dynamic trunk reservation algorithm is implemented by class-of-service [27].

ACKNOWLEDGMENT

We are indebted to S. D. Schwartz of AT&T Bell Laboratories for his contributions to the early phase of this study. We also are thankful for stimulating discussions and collaborations with our colleagues at Bell Labs; D. Mitra, A. E. Frey, and J.-S. Chen.

REFERENCES

[1] K. S. Narendra, E. A. Wright, and L. G. Mason, "Application of learning automata to telephone traffic routing and control," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-7, Nov. 1977.

[2] K. S. Narendra and M. A. L. Thathachar, "On the behavior of a learning automaton in a changing environment with application to telephone traffic routing," *IEEE Trans. Syst., Man Cybern.*, vol. SMC-10, May 1980.

[3] W. H. Cameron, P. Galloy, and W. J. Graham, "Report on the Toronto advance routing concept trail," in *Proc. Networks Con.*, Paris, France, 1980.

[4] G. R. Ash, R. H. Cardwell, and R. P. Murray, "Design and optimization of networks with dynamic routing," *Bell Syst. Tech. J.*, vol. 60, no. 8, Oct. 1981.

[5] G. R. Ash, A. H. Kafker, and K. R. Krishnan, "Servicing and real-time control of networks with dynamic routing," *Bell Syst. Tech. J.*, vol. 60, no. 8, Oct. 1981.

[6] F. A. Field, "The benefits of dynamic nonhierarchical routing in metropolitan traffic networks," in *Proc. Tenth Int. Teletraff. Congr.*, Montreal, P.Q., Canada, June 1983.

[7] J. J. Carroll, M. J. DiCarlo-Cottone, and A. H. Kafker, "4ESS™ switch implementation of dynamic nonhierarchical routing," in *Proc. IEEE Global Telecommun. Conf.*, San Diego, CA, Dec. 1983.

[8] G. R. Ash, A. H. Kafker, and K. R. Krishnan, "Intercity dynamic routing architecture and feasibility," in *Proc. Tenth Int. Teletraff. Cong.*, Montreal, P.Q., Canada, June 1983.

[9] J. M. Garcia, F. Le Gall, C., Castel, P. Chemouil, P. Gauthier, and G. Lechermeier, "Comparative evaluation of centralized/decentralized traffic routing policies in telephone networks," in *Proc. Eleventh Int. Teletraff. Congr.*, Kyoto, Japan, Sept. 1985.

[10] G. R. Ash, "Use of a trunk status map for real-time DNHR," in *Proc. Eleventh Int. Teletraff. Congr.*, Kyoto, Japan, Sept. 1985.

[11] P. Chemouil, J. Filipiak, and P. Gauthier, "Analysis and control of traffic routing in circuit-switched networks," *Comput. Networks ISDN Systs.*, vol. 11, no. 3, Mar. 1986.

[12] R. J. Gibbens, "Some aspects of dynamic routing in circuit-switched telecommunications networks," *Statist. Lab.*, Univ. Cambridge, Jan. 1986.

[13] A. Mees, "Simple is best for dynamic routing in circuit-switched telecommunications networks," *Nature*, vol. 323, p. 108, Sept. 11, 1986.

[14] BNR, Special Issue: Dynamic Network Controller Family, *Telesis Mag.*, 1986.

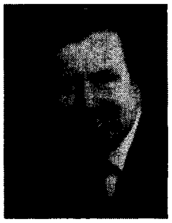
[15] B. R. Hurley, C. J. R. Seidl, and W. F. Sewell, "A survey of dynamic routing methods for circuit-switched traffic," *IEEE Commun. Mag.*, vol. 25, 1987.

[16] Y. Watanabe, and H. Mori, "Dynamic routing schemes for international ISDNs," in *Proc. Fifth ITC Specialists Sem: Traffic Eng. ISDN Design and Planning*, Lake Como, Italy, May 1987.

[17] P. Gauthier, P. Chemouil, and M. Klein, "STAR: A system to test adaptive routing in France," in *Proc. IEEE Global Telecommun. Conf.*, Tokyo, Japan, 1987.

[18] R. R. Stacey and D. J. Songhurst, "Dynamic alternative routing in the British telecom trunk network," in *Proc. Int. Switching Symp.*, Phoenix, AZ, Mar. 1987.

[19] S. S. Katz, G. R. Ash, and A. N. Kashper, "Robust design of a worldwide intelligent network in the digital era," in *Int. Sem. Teletraffic Networking,* Beijing, China, Sept. 1988.
[20] D. M. Wanamaker and D. A. Dorrance, "Dynamically controlled routing field trail experience," in *Proc. Network Oper. Management Syst. Conf.,* New Orleans, LA, Mar. 1988.
[21] F. Caron, "Results of the telecom Canada high performance routing trail," in *Proc. Twelfth Int. Teletraffic Congr.,* Torino, Italy, June 1988.
[22] K. R. Krishnan and T. J. Ott, "State-dependent routing for telephone traffic: Theory and results," in *Proc. 25th Conf. Decision Contr.,* Athens, Greece, Dec. 1988.
[23] G. R. Ash, B. M. Blake, and S. D. Schwartz, "Integrated network routing and design," in *Proc. Twelfth Int. Teletraff. Congr.,* Torino, Italy, June 1988.
[24] G. R. Ash, P. Chemouil, A. N. Kashper, S. S. Katz, K. Yamazaki, and Y. Wantanabe, "Robust design and planning of a worldwide intelligent network," *IEEE J. Select. Areas Commun.,* vol. 7, Oct. 1989.
[25] G. R. Ash and E. Oberer, "Dynamic routing in the AT&T network—Improved service quality at lower cost," in *Proc. IEEE Global Telecommun. Conf.,* Dallas, TX, Nov. 1989.
[26] G. R. Ash and S. D. Schwartz, "Network routing evolution," *Network Management Contr. Workshop,* Tarrytown, NY, Sept. 1989.
[27] G. R. Ash, J.-S. Chen, A. E. Frey, and B. D. Huang, "Real-time network routing in a dynamic class-of-service network," in *Proc. Thirteenth Int. Teletraff. Congr.,* Copenhagen, Denmark, June 1991.
[28] F. Langlois and J. Regnier, "Dynamic congestion control in circuit-switched telecommunications networks," in *Proc. Thirtenth Int. Teletraff. Congr.,* Copenhagen, Denmark, June 1991.
[29] S. S. Katz, "Statistical performance analysis of a switched communication network," in *Proc. Fifth Int. Teletraff. Congr.,* NY, 1967.
[30] F. P. Kelly, "Blocking probabilities in large circuit-switched networks," in *Advances in Applied Probability,* vol. 18, 1986.
[31] E. W. M. Wong and T.-S. Yum, "Selective alternate routing for nonhierarchical circuit-switched networks," Dep. Electron., The Chinese University of Hong Kong, Shatin, Hong Kong.
[32] ——, "Maximum free circuit routing in circuit-switched networks," in *Proc. IEEE INFOCOM '90,* 1990.
[33] D. Mitra, R.J. Gibbens, and B.D. Huang, "Analysis and optimal design of aggregated-least-busy-alternative routing on symmetric loss networks with trunk reservation," in *Proc. Thirteenth Int. Teletraff. Congr.,* Copenhagen, Denmark, June 1991.
[34] J. Regnier, P. Blondeau, and W.H. Cameron, "Grade of service of a dynamic call-routing system," in *Proc. Tenth Int. Teletraff. Congr.,* Montreal, P.Q., Canada, June 1983.
[35] J. M. Akinpelu, "The overload performance of engineered networks with nonhierarchical and hierarchical routing," *Bell Syst. Tech. J.,* vol. 63, 1984.
[36] W. Whitt, "Blocking when service is required from several facilities simultaneously," *AT&T Tech. J.,* vol. 64, 1985.
[37] S. P. Chung and K. W. Ross, "Reduced load approximations for multirate loss networks," Tech. Rep., Univ. Pennsylvania, 1990.
[38] D. Mitra and J. B. Seery, "Comparative evaluations of randomized and dynamic routing strategies for circuit-switched networks," *IEEE Trans. Comm.,* Jan. 1991.



Gerald R. Ash (F'93) received the B.S. degree from Rutgers University in 1964, and the M.S. and Ph.D. degrees at the California Institute of Technology in 1965 and 1969, all in electrical engineering.

He is Technical Manager of Routing Planning at AT&T Bell Laboratories in Holmdel, NJ. He joined AT&T Bell Laboratories in 1972, and in 1976 became supervisor of the Routing Studies Group: a group responsible for studies and analyses which led to the development of dynamic nonhierarchical routing (DNHR). In 1981 he took on his present responsibility for AT&T network evolution which includes traffic engineering for integrated services digital networks, real-time network control, and self-healing network design. He has been closely involved with the implementation of DNHR in the AT&T long distance network, and in 1984 he was named an AT&T Bell Laboratories Fellow for his contributions in this area. More recently he has been involved in the implementation of class-of-service real time network routing (RTNR), which replaced DNHR in 1991.

Dr. Ash is author of over 50 articles and technical papers, and holds six patents. He was awarded the Bronze Star Medal for his Vietnam Service, and is the recipient of the 1989 Alexander Graham Bell Medal awarded by the IEEE. He is a member of Eta Kappa Nu, Tau Beta Pi, and Sigma Xi.

BaoSheng D. Huang received the B.Sc. degree in mathematics from Fu-Jen Catholic University, the M.Sc. degree in computer science and electrical engineering from Stevens Institute of Technology, and the Ph.D. degee in operations research from the University of North Carolina at Chapel Hill.

He is a Distinguished Member of Technical Staff of Network Utilization at AT&T Bell Laboratories in Middletown, NJ. He joined Bell Labs. in 1981. From 1981 to 1986, he worked on loop technologies planning and digital loop carrier development. Since 1987, he has been working on switched network planning and design, including state dependent routing, integrated voice/data network planning and design, and network reliability planning. He has been working on the analysis and design of AT&T's Real Time Network Routing (RTNR) network, and his main interests are in network modeling and performance evaluation for new network technologies.