USE OF A TRUNK STATUS MAP FOR REAL-TIME DNHR

Gerald R. Ash
AT&T Bell Laboratories
Holmdel, New Jersey

ABSTRACT

Dynamic nonhierarchical routing (DNHR) is a new routing technique that is currently being implemented in the AT&T Communications network. DNHR incorporates a preplanned time-of-day routing strategy that enables a significant increase in the efficiency of the AT&T-C network. Trunk status map routing (TSMR) is an extension of the DNHR concept to a centralized trunk status map (TSM) that provides real-time routing decisions in the DNHR network. Various implementations of TSMR are investigated, and the comparative advantages of several real-time dynamic routing algorithms are illustrated. A simple implementation of TSMR which is a hybrid of time variable routing and fully dynamic routing is shown to yield benefits comparable to more complicated schemes.

1. INTRODUCTION AND SUMMARY

Earlier work has described the design, optimization, and control of networks with dynamic routing [1-3]. The term dynamic routing frequently suggests a real-time search for the optimal routing patterns based on current network loads. In its initial implementation, dynamic nonhierarchical routing (DNHR) uses a preplanned time-variable routing strategy with a limited amount of real-time control. Real-time, traffic sensitive routing is indeed the limiting case of preplanned time-variable routing, and in this paper we investigate the feasibility and benefits of such "true dynamic routing." Here we investigate a central control capability called trunk status map routing (TSMR), which greatly extends the real-time routing capability of DNHR. TSMR uses a real-time network status map to implement a dynamic routing strategy, and we examine the interaction of TSMR with network operation. The trunk status map (TSM) concept involves having an update of the number of idle trunks in each DNHR trunk group sent to a network data base every T seconds. These updates are sent by each stored program control (SPC) switch only when the trunk group status has changed. In return, the TSM, which is located within the common channel signaling network, periodically sends to the SPC switches ordered routing sequences to be used until the next update in T seconds. These routing sequences are determined by the TSM in real time using the TSMR dynamic routing strategy. As T approaches zero seconds, these updates effectively are sent for every call arrival.

In this paper we investigate through simulation techniques a number of approaches to TSMR. These TSMR strategies range from schemes that simply determine the single most idle routing path (which is a one or two-link connection) between each node pair in the network to more complex schemes that periodically reoptimize the entire network routing pattern by solving a large mathematical program. The scheme that appears to be the most attractive operates as follows. The first choice path determined by the design algorithm (the unified algorithm or UA [1], as modified for TSMR [4]) is used if a circuit is available. This first choice path is updated once every routing interval or load-set period (LSP), which is typically one hour to several hours in duration. If the first path is busy, the second path is selected from the list of other paths determined by the UA on the basis of having the greatest number of idle circuits at the time. Hence the TSMR approach is a hybrid of time variable routing and fully dynamic routing. A detailed implementation strategy is described that stabilizes the routing patterns through trunk reservation techniques, and also allows automatic routing controls to augment the TSMR strategy during network overloads and failures.

TSMR provides uniformly better blocking performance than the current implementation of DNHR and a significantly reduced number of crankback messages; both network blocking and crankback messages can decrease as the number of idle trunks increases. TSMR updates effectively are sent for every call arrival. DNHR network operations are more centralized with time-of-day routing and fully dynamic routing. That DNHR call has been blocked on the second link of a two-link path. An initial evaluation of TSMR provides performance under overloads and failures that is significantly better than the performance of the current implementation of DNHR.

2. REVIEW OF DNHR CONCEPTS

DNHR brings three principal changes to the network plan. First, there is a new network configuration. With DNHR, the AT&T Communications (AT&T-C) network will evolve from its present multiple level structure to a structure consisting of DNHR switches at the upper level with subsetting hierarchical switches at the lower level. There is also a new routing technique. DNHR allows the choice of traffic paths to change with time of day and is not constrained by a hierarchical ranking of the switches. This approach greatly expands the flexibility of network routing and thereby permits a large increase in the efficient use of the network. That efficiency reduces the need to construct transmission facilities in the future. Finally, there is a new way of operating and designing the network. DNHR network operations are more centralized and more automated than they are today, and that increases operating efficiency and improves network performance. Several technical feasibility studies have shown that these changes brought about by DNHR are technically feasible, that network performance is comparable to that of the hierarchy, and, with a proper level of automation, that operation of a DNHR network is quite tractable [3].

Figure 1 illustrates the network structure which incorporates DNHR into the AT&T-C intercity network. The DNHR network has only one class of switching system, and end-offices are home on DNHR tandem offices by way of exchange access networks. The DNHR portion of the AT&T-C intercity network consists of 4ESS™ switches (and possibly successor
The real-time, traffic-sensitive component of DNHR uses "real-time" paths for possible completion of calls that overflow the "engineered" paths (see Figure 2). The engineered paths are designed to provide the objective blocking performance.

The real-time paths, which are also determined by the central forecasting system, can be used only if the number of idle trunks in a group is greater than a specified number of trunks—the reservation level—before the connection is made. This prevents calls that normally use a trunk group from being swamped by real-time routed calls.

The third principal change brought about by DNHR is the way the network is designed and operated. Several operations systems provide centralized functions such as switch planning, trunk forecasting, trunk servicing, routing administration, and network management. An overview of the operations systems used to support DNHR is given in Reference 3. Embedded within the forecasting and servicing systems is the UA, which simultaneously determines the trunking and routing for the entire DNHR network [1].

3. TRUNK STATUS MAP ROUTING METHOD

The real-time routing method discussed in connection with the present implementation of DNHR is used to improve network service. Service improvement with real-time routing is significant even with relatively simple procedures. As an alternative to using real-time routing to improve service, we could hold the node-to-node blocking (service) level fixed and use the real-time routing scheme to reduce the number of trunks required to provide that level of service. This alternative approach can produce additional network savings. Furthermore, real-time dynamic routing can improve network performance in the event of network failures, especially when some amount of reserve capacity is available for redirecting traffic flows from their usual patterns. Hence an increased level of real-time decision making might be warranted in the DNHR network.

Trunk status map routing (TSMR) is an extension of the DNHR concept that uses a centralized trunk status map (TSM) to provide real-time routing decisions in the DNHR network. As described in the introduction, the TSM receives, every T seconds from each SPC switch, updates of the number of idle trunks in each DNHR trunk group. These updates are sent only when the number of idle trunks in the trunk group has changed. In return, the TSM periodically sends to the SPC switches ordered routing sequences to be used until the next update in T seconds. These routing sequences are determined by the TSM in real time using the TSM dynamic routing strategy (TSMR). TSMR therefore represents a much more dynamic routing method than the current implementation of DNHR.

The real-time routing method can improve network performance, especially in the event of network failures, by dynamically routing calls to routes that are less congested. This method is especially useful in situations where the network is already heavily loaded and additional traffic is being added.

The TSMR method is an extension of the DNHR concept that uses a centralized trunk status map (TSM) to provide real-time routing decisions in the DNHR network. As described in the introduction, the TSM receives, every T seconds, updates of the number of idle trunks in each DNHR trunk group. These updates are sent only when the number of idle trunks in the trunk group has changed. In return, the TSM periodically sends to the SPC switches ordered routing sequences to be used until the next update in T seconds. These routing sequences are determined by the TSM in real time using the TSM dynamic routing strategy (TSMR). TSMR therefore represents a much more dynamic routing method than the current implementation of DNHR.
In this section we investigate alternative approaches to TSMR as well as the appropriate value of the status and routing update interval (T). We also study an automatic network congestion control strategy that could be implemented by the TSM.

A call-by-call simulation model is used to measure the performance of an engineered network under various routing strategies: DNHR, various alternative TSMR strategies, and TSMR combined with automatic congestion control strategies. A 25-node DNHR network model projected for 1986 is used for the simulation studies. The 25-node model is designed by the DNHR and TSMR design algorithms [1,4] for 16 hours throughout the day (from 8 a.m. through 11 p.m.). The behavior of the network is investigated over a typical two-week period consisting of 10 average business days. Low daily variations are applied to the load for each node-to-node pair in the network simulation. In addition, a systematic daily variation of the total network load is superimposed on the random load variation according to typical load patterns over an average business week.

3.1 Alternative Approaches to TSMR

Various techniques are investigated to determine the most efficient TSMR method. These methods include the following:

1) route each call on the least loaded path (the path having the greatest number of idle circuits) among all candidate paths;
2) route each call first on the direct path, if it exists and is available, or else select the least loaded path;
3) route each call on the first path assigned by the UA, if available, or else select the least loaded path;
4) compute the UA routing sequences that maximize carried traffic for a short-term estimate of the network loads and then apply Method 3 using the new routing sequences.

Methods 1 and 2 do not provide adequate performance for the following reasons. Method 1 favors the path having the largest available capacity, not necessarily the first path, and usually a two-link path. This tendency to favor two-link paths results in a significant redistribution of flows and relatively poor network performance. Method 2 performs considerably better than Method 1 but falls short of the performance of Method 3, which we describe shortly. Method 2 has a problem similar to that of Method 1: since it always selects the direct path as the first choice, and since the UA does not always design the direct path to be the first choice, Method 2 makes the path order too different from that designed by the UA. Hence Method 1 and Method 2 performance is degraded because the actual realization of network flows deviates too far from the network flow patterns designed by the UA.

Method 3, however, performs quite well. It reflects sufficiently accurately the design of the UA by assigning first path traffic to the design first path. Flow on the first choice path accounts for about 80 to 90 percent of the total network traffic flow and hence the routing of this flow must correspond well to the placement of trunks for the network to behave properly. Flows on the second and higher-numbered paths are better assigned by a least-loaded selection method than by a preplanned sequential selection method. This statement is supported by the simulation results shown in Figures 3 and 4.

Figure 3 shows the 10-day average hourly blocking for TSMR Method 3 (for status and routing update interval T equal to five seconds) and also for the current implementation of DNHR. Figure 4 shows the 99th percentile hourly node pair blocking for TSMR and DNHR. These results were obtained with a network designed for DNHR and clearly demonstrate the benefits of TSMR in comparison to the current implementation of DNHR. The details of the TSMR strategy used in these simulations will be described in Section 3.2, 3.3, and 3.4.

A simple intuitive explanation of why TSMR Method 3 might complete more calls than the current implementation of DNHR is illustrated by the four-node example shown in Figure 5. The current number of idle circuits in each trunk group is shown. If the DNHR routing sequence for A-D calls is A-D - A-C-D - A-B-D, then the next A-D call arrival will block link AC and link CD, and either an A-C or C-D call arrival will then be blocked. However, the TSMR routing sequence for A-D calls in this network state is A-D - A-B-D - A-C-D. Therefore the next A-D call arrival under the TSMR strategy will not block any additional links. As illustrated by this simple example, TSMR tends to leave capacity on links throughout the network, distributed as uniformly as possible. Because of this property of TSMR, calls arriving in various parts of the network will have a greater chance of being completed than under the hypothesized DNHR routing sequence. Bulfer [5] has established the optimality of the least loaded routing strategy for a class of two stage concentrators.

![Figure 4](image-url)
TSMR Method 4 has greater adaptivity to load shifts than Method 3 but is more complex to implement and as such must have better performance to be justified. The details of Method 4 are now discussed. To calculate optimum routing sequences with the UA, as required by Method 4, we assume that these routing patterns are computed in advance of their actual use and that the average load in the future LSP is known precisely. This second assumption of course is an idealization of reality and represents an upper limit on the possible performance of Method 4. We used two methods based on the UA to determine the routing sequences that maximize network flow in the existing network. For each method we find the minimum incremental network capacity required to carry the future LSP load given the existing trunks as available network capacity [2]. That is we minimize

$$\sum_{j=1}^{L} \Delta a_j,$$

where $\Delta a_j$ are the augmentations above the existing link capacities $a_j$, and $L$ is the number of links in the network. In this formulation of the UA, we set all incremental link costs to one in order to transform the objective function from incremental network cost to incremental network capacity.

In the first method (which we will call Method 4A), the maximum node-to-node blocking grade of service (GOS) is held at one percent, and the routing and capacity augmentations $\Delta a_j$ are determined which minimize the objective function. We implement the routing solution, but the capacity augmentations $\Delta a_j$ produced by the optimization cannot actually be added to the network. Hence the traffic that would have been carried on these augmentations will actually be blocked. But since we have minimized these hypothetical capacity augmentations, this routing solution approximates the minimization of total blocked traffic. In the second method (Method 4B) the node-to-node blocking GOS is raised until there are zero capacity augmentations $\Delta a_j$ produced by the optimization. This second solution approximates the minimization of the maximum node-to-node blocking.

A comparison of the average network blocking performance of Method 3, Method 4A, and Method 4B is shown in Figure 6. For these results the update interval $T$ is five seconds, and the network is designed for TSMR, as described in Reference 6. As can be seen there is no apparent improvement gained from the more complex methods over Method 3, which suggests that Method 3 may achieve nearly the maximum flow performance. TSMR Method 3 was therefore selected for further study to determine the best switching control logic and TSM logic to implement TSMR.

### 3.2 Implementation Strategy for TSMR

Extensive simulation studies of TSMR Method 3 were used to determine the implementation strategy discussed in this section. We describe the TSMR strategy found to perform the best from these simulation studies.

The SPC switch maintains a TSMR route sequence for each destination. The route sequence stored in the SPC switch is made up of two functional parts. The first part consists of a single path, called the "first-choice path," which is updated by the TSM once each LSP and is made to correspond to the UA selected first-choice path. The second and subsequent paths of the route sequence consist of the "least loaded routing (LLR) paths." These LLR paths are updated every $T$ seconds according to a least loaded criterion applied to the current network status. For each route sequence that needs to be updated, the TSM determines the paths in the current route sequence that need to be changed and transmits these changes to the SPC switch. If the least loaded path differs from the second-choice path currently in the SPC switch, the TSM sends an appropriate message to the SPC switch that changes the contents of the LLR paths to reflect the new least loaded path and also changes the position of the other paths in the route sequence. We found from the simulations that it is sufficient for small values of $T$ to have the TSM compute only the least loaded path and put it second in a route rather than sort the entire route in least loaded order.

For each node pair for which the direct trunk group exists and is the first choice, the TSM applies a thresholding scheme to determine whether a routing update is needed, as follows. Consider the typical case in which the direct path is the first choice and a two-link path is the second choice (case 1 in Figure 7). If the total number of idle circuits on these two choices is greater than a threshold number which is sufficient to...
permit completion of all calls likely to arrive over the T-second update interval, then there is no need to reorder the route sequence since the calls can be completed without generating any crankbacks. A threshold that works reasonably well is $N/36$, where $N$ is the number of trunks on the direct trunk group to the terminating switch. (If the direct group does not exist or is not the first choice, the thresholding scheme is not used). For case 1 in Figure 7, this means that if the number of trunks on the direct group is less than 200, then no routing update message will be sent to the SPC switch. For case 2 in Figure 7, this thresholding scheme is not used. Use of this logic concentrates the TSM processing capacity on those node pairs for which a routing update to the SPC switch is most beneficial. This logic also limits the number of node pairs requiring updated LLR paths.

If there is not a sufficient number of idle circuits according to the thresholding scheme discussed above, and a routing update is potentially required, the next step for the TSM is to determine the current least loaded path. It does this by first retrieving the LLR path choice candidates, for this node pair, from the routing data base memory. These candidates are typically a subset of the one- and two-link paths between the two nodes. In the simulations we find that the number of idle circuits by a small fraction of the trunk group size protects large groups from being selected by a disproportionately large number of node pairs as a via-path candidates and these large groups have temporarily idle trunks. Large groups tend to have short periods with a relatively large number of idle trunks, and overselection of those groups for via traffic during these short periods can be detrimental to the direct traffic routed on the large groups. The simulation studies indicate that discounting the number of idle trunks by N/36 trunks, where N is the trunk group size, works reasonably well in protecting the direct traffic on the large groups. (Groups with fewer than 36 trunks are not affected). Using the discounted idle trunk values, the TSM then determines the number of idle circuits on each path choice and selects the path having the greatest number of idle circuits (which we call the least loaded path). This discounting procedure is applied only in the TSM path ordering logic and is not applied in the SPC switch path selection logic.

If the new least loaded path is the same as the old path or if there are no idle circuits on any candidate path, then a routing update is not sent to the SPC switch because there is no advantage in doing so. Simulations of this logic in combination with the idle circuit thresholds described above predict that almost all the node pairs need routing updates. When the least loaded path does change, then a message is sent to the SPC switch changing the contents of the routing sequence, in the SPC switch. As an example, consider case 2 in Figure 7. In the figure the idle trunk quantities are the quantities discounted by N/36 trunks. The current route to switch B has a maximum of four path choices, of which three are LLR path candidates - A-C-B, A-D-B, and A-B - in addition to the two-link, first-choice path A-E-B determined by the UA. The TSM first determines that the second LLR path candidate, A-D-B, has become about 10-15 percent lower in the TSM workload, which changes the first-choice path to route via the second path A-D-B, and then the TSM sends LLR path choices one and two to the SPC switch in the order A-D-B and A-C-B. Path choice A-B is not sent to the SPC switch since it remains the third LLR path choice. We call this a "push-down" logic since the new least loaded path replaces the old least loaded path, and the remaining path choices are pushed down one slot. The SPC switch replaces its current path with the LLR path choices one and two with the new paths and leaves the current third LLR path choice unchanged. Hence this logic places the least loaded path in the second-choice position but does not necessarily leave the remaining paths in the order of their available capacity. According to the simulation results, for an update interval $T$ of five seconds, the TSM transmits an average of about four entries in the SPC path sequence with each routing update.

With this implementation of TSMR, the SPC switch does not require any time-varying routing capability of its own, as now implemented for DNHR [6]. All such dynamic routing capabilities are controlled by the TSM, which changes the first-choice path every LSP and also controls the LLR paths in a fully dynamic manner. The SPC switch, however, is the only place where individual trunks are selected and assigned to a particular call. The SPC switch sets up all calls over the selected paths using the present DNHR two-link routing procedure with crankback. The number of crankbacks, however, is greatly reduced by the use of TSMR.

### 3.3 Status and Routing Update Interval $T$

Four 10-day simulations were made with values of the status and routing update interval, $T$, equal to 2, 5, 10, and 30 seconds. For these four values of $T$, Figure 8 compares the TSMR results on the basis of three performance measures. The first measure is average network blocking, which is the 10-day total number of blocked calls divided by the 10-day total number of originating attempts; the second measure is average number of crankbacks per originating attempt, which is the 10-day total number of simulated crankbacks divided by the 10-day total number of originating attempts; and the third measure is the TSM workload, which is the daily total number of least loaded path searches averaged over the 10 days. We can see from the results that the status and routing update interval should be as short as possible within the limitations of SPC switch and TSM processing if we wish to maximize completions and minimize crankbacks.

There is approximately a two percent penalty in SPC switch capacity due to the real time needed to process crankback messages associated with the implementation of DNHR in the 4ESS switch [3]. If we apply this penalty to the current forecast of switches in the mid-1990's, and if we assume that the switch cost penalty for the TSMR strategy can be allocated in direct proportion to the total number of crankbacks generated, then we obtain the crankback cost as a function of $T$, which is shown in Figure 9. Also shown in Figure 9 is the total cost of processors needed to support the TSMR processing level shown in Figure 8, under the assumption that two processors are needed to support a five-second update interval for a 180-node TSMR network. Here we have assumed that each processor costs $100,000. As shown in Figure 9, no fewer than four processors are required since the TSM is duplicated, because of reliability considerations, as are signal transfer points in the CCS network, and at least two processors are required at each TSM, also for reliability purposes. The results shown in Figure 9 suggest that an update interval $T$ in the range of two to eight seconds will minimize the total cost of processors needed to support the TSMR processing level.
idle in the process of determining the least loaded path. In no case is trunk reservation applied at the TSM or SPC switch for a one-link (direct) path.

BPR controls are triggered on total network blocking, and, once triggered, BPR removes all busy two-link paths from all route sequences. That is, when the average network blocking over a five minute period exceeds three percent, those two-link paths (excluding the first path) that have zero free circuits are removed from the SPC switch route patterns for the next five-second interval. This strategy results in a uniformly applied restrictive control on the network; all node pairs are triggered at once and are prevented from routing calls through congested parts of the network. BPR controls are also used when an SPC switch signals the TSM that it is in congestion or has failed. In the case of SPC switch congestion, the TSM removes the node as a via point in all routing patterns, and in the case of SPC switch failure the TSM removes both direct and via routing to the failed node.

With the ERL logic, extended searches for idle capacity are triggered for a node-to-node pair whenever the blocking for the pair exceeds one percent over a five-minute period. When the total network blocking threshold used for BPR is triggered, however, then the ERL logic is used only when the node-to-node blocking over a five-minute interval exceeds five percent, with a minimum of two blocked calls. When ERL is triggered, the search for the least loaded path for the triggered node pair is extended to include a larger set of candidate paths.

We simulated the ATR, BPR, and ERL controls over the three hour morning busy period for an average daily load, as well as for 10, 20, and 30 percent general overloads in which each node-to-node traffic load was increased by the overload percentage. We also considered a focused overload on the White Plains switch in which each load to White Plains was increased by a factor of three. Finally, we considered two failure situations which involved a Dallas-Wayne (DILLS­WAYN) trunk group failure and an Anaheim-White Plains (ANHM­WHPL) trunk group failure. The results are given in Table 1, in which the measures used reflect averages over the

### TABLE 1

<table>
<thead>
<tr>
<th>ROUTING</th>
<th>% OVERLOAD</th>
<th>AVG PERCENTILE</th>
<th>99th PERCENTILE</th>
<th>CRKBS CALL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNHR</td>
<td>0</td>
<td>.00044</td>
<td>.00215</td>
<td>.01131</td>
</tr>
<tr>
<td>TSMR</td>
<td>0</td>
<td>.00002</td>
<td>.00010</td>
<td>.00257</td>
</tr>
<tr>
<td>DNHR</td>
<td>10</td>
<td>.01382</td>
<td>.03330</td>
<td>.03462</td>
</tr>
<tr>
<td>TSMR</td>
<td>10</td>
<td>.00479</td>
<td>.01230</td>
<td>.00931</td>
</tr>
<tr>
<td>DNHR</td>
<td>20</td>
<td>.05753</td>
<td>.11558</td>
<td>.08038</td>
</tr>
<tr>
<td>TSMR</td>
<td>20</td>
<td>.03016</td>
<td>.13793</td>
<td>.01855</td>
</tr>
<tr>
<td>DNHR</td>
<td>30</td>
<td>.10745</td>
<td>.18454</td>
<td>.12002</td>
</tr>
<tr>
<td>TSMR</td>
<td>30</td>
<td>.06524</td>
<td>.22876</td>
<td>.05332</td>
</tr>
<tr>
<td>DNHR</td>
<td>FOCUS</td>
<td>.01860</td>
<td>.20361</td>
<td>.04878</td>
</tr>
<tr>
<td>TSMR</td>
<td>ON WHPL</td>
<td>.00808</td>
<td>.12265</td>
<td>.02655</td>
</tr>
<tr>
<td>DNHR</td>
<td>DILLS­WAYN</td>
<td>.00101</td>
<td>.00806</td>
<td>.01556</td>
</tr>
<tr>
<td>TSMR</td>
<td>FAILURE</td>
<td>.00004</td>
<td>.00055</td>
<td>.00314</td>
</tr>
<tr>
<td>DNHR</td>
<td>ANHM­WHPL</td>
<td>.00187</td>
<td>.01651</td>
<td>.03012</td>
</tr>
<tr>
<td>TSMR</td>
<td>FAILURE</td>
<td>.00001</td>
<td>.00000</td>
<td>.00167</td>
</tr>
</tbody>
</table>
three hours of the simulation: the average network blocking is the total number of blocked attempts in three hours divided by the total number of originating attempts; the 99th percentile blocking is the 99th percentile node pair blocking, where each node pair blocking is averaged over the three simulation hours; and the average number of crankbacks per originating attempt is the total simulated number of crankbacks over the three simulation hours divided by the total number of originating attempts. For comparison, we have included results for the current implementation of DNHR. The results indicate that TSMR (with automatic congestion controls) provides improved average blocking performance and comparable or better 99th percentile blocking performance for all the cases studied. The number of crankback messages is also significantly lower with TSMR.

As another indication of peak load performance, Figure 10 shows the hourly blocking performance for TSMR and the current implementation of DNHR under an average Monday load pattern (these loads are normally the highest loads of the week). These results also demonstrate the ability of TSMR congestion controls to increase network flow in comparison to DNHR. Both DNHR and TSMR performance would be improved further by automatic network management controls present in the SPC switch and in the network management support system.

4. CONCLUSION
Our conclusions from these studies are that 1) TSMR provides uniformly better blocking performance than the current implementation of DNHR and a significantly reduced number of crankback messages; both network blocking and the number of crankback messages decrease as the status and routing update interval, T, decreases, 2) an initial evaluation of TSM processing cost and crankback cost as a function of the update interval T places the optimum value of T in the range of two to eight seconds; and 3) initial studies show that the TSM can implement effective automatic congestion control strategies.

5. ACKNOWLEDGEMENTS
B. B. Oliver of AT&T Communications first suggested the use of a trunk status map for DNHR. I also benefited greatly from the insights of A. F. Bulfer of AT&T Communications who first suggested investigation of LLR strategies for TSMR. A. H. Kafker of AT&T Bell Laboratories was instrumental in the design of the TSM implementation strategy described in Section 3.2.

REFERENCES
4. Ash, G. R., forthcoming BSTJ paper on TSMR.