A Minimax Utilization Routing Algorithm in Networks with Single-path Routing

Frank Y.S. Lin
Bellcore
444 Hoes Lane
Piscataway, NJ 08854
linl@cc.bellcore.com

Jonathan L. Wang
Bellcore
331 Newman Springs Road
Red Bank, NJ 07701
jwang@cc.bellcore.com

Abstract

In this paper, we consider the routing problem in networks with single-path routing and multicasting services. In such networks, all the single-destination traffic is transmitted over exactly one path between the origin and the destination and the multiple-destination traffic (if applicable) is transmitted over exactly one tree rooted at the origin. Examples of such networks are the conventional circuit-switched telephone networks, virtual-circuit based packet networks such as the X.25 networks, networks supporting Switched Multi-megabit Data Service (SMDS), networks supporting Frame Relay Service (FRS), and the recently proposed Broadband Integrated Services Digital Networks (B-ISDNs) based on Asynchronous Transfer Mode (ATM). We consider the problem of choosing a route/tree between every origin and its single/multiple destination(s) in a network so as to minimize the maximum link utilization. We consider the formulation of this problem as a linear integer programming problem.

The emphasis of this paper is (i) to consider routing for single-destination and multiple-destination (multicast) traffic in a uniform way, (ii) to develop a near-optimal quasi-static algorithm to solve the problem, and (iii) to evaluate the efficiency and effectiveness of using the minimax criterion as a surrogate objective of other performance measures, e.g. the average packet delay. The basic approach to the algorithm development is Lagrangean relaxation. In computational experiments, the proposed algorithm determines solutions that are within a few percent of an optimal solution for networks with up to 61 nodes in 2 minutes of CPU time. Compared with the minimum hop routing scheme, the minimax utilization routing algorithm results in a 16.67% to 75.00% improvement in the maximum link utilization factor for 10 test networks. We also show that the routing decisions made by the minimax utilization routing (MUR) algorithm are within 2% of an optimal solution where the objective is to minimize the average packet delay. Finally, we discuss issues of implementing the MUR algorithm.

1. Introduction

In this paper, we consider the routing problem for networks with single-path routing and multicasting services. In a network with single-path routing, all of the single-destination traffic is transmitted over exactly one path between the origin and the destination. If the network also provides multicasting service, the multiple-destination traffic from an origin is transmitted over exactly one tree rooted at the origin where one copy of the packets is transmitted over each link in the tree (packets are not duplicated unless necessary). Due to ease of management and planning, many networks that provide single-path routing service have been implemented or proposed. These include the conventional circuit-switched telephone networks, the virtual-circuit based packet networks (such as Public Packet Switched Networks (PPSN)[1], SNA[2], Telenet[3], TYMNET[4], TRANSPAC[5]) , networks supporting Switched Multi-megabit Data Service (SMDS)[6], , networks supporting Frame Relay Service (FRS)[7], and Broadband Integrated Services Digital Networks (B-ISDNs)[8] based on the Asynchronous Transfer Mode (ATM) technology. The routing algorithms used in these networks depend on the specific switching equipment manufacturers and are mostly either fixed or adaptive only when major events occur (e.g., topology change) instead of being adaptive to the dynamic traffic conditions in the networks. This is due to the fact that static routing is simple in terms of network management and capacity expansion, and that static routing does not generate extra network routing messages as in the case of dynamic routing. However, routing without considerations of network traffic conditions usually gives poor performance especially when the traffic condition changes a lot. We therefore, in this paper, consider a "quasi-static" routing algorithm that not only is simple in terms of network planning but also takes (longer term) traffic conditions into account.

The objective of the algorithm is to find a route/tree between every origin and its destination(s) in a network so as to minimize the maximum link utilization. The algorithm is referred as the *minimax utilization routing (MUR) algorithm*. The major advantages of using the minimum of the maximum link utilization as the performance objective include:

- A single performance indicator (the maximum link utilization factor) is provided. This single value can be used to derive upper bounds on other performance measures, e.g. end-to-end delay, call blocking rate and cell loss probability.
- For engineering tractability, end-to-end performance objectives are converted to link utilization constraints. The MUR then provides the most efficient utilization of the network capacity and avoids unnecessary capacity expansion.
- The minimax criterion can be treated as a goal of the system to provide a balanced and robust operating point.
- It is clear that an optimal routing assignment (with respect to the minimax criterion) remains optimal if the traffic requirements grow uniformly.
- The routing decisions made by the MUR algorithm usually do very well with respect to other major performance criteria in various networks such as the call blocking probability in circuit-switched networks, the packet delay in virtual-circuit based packet networks and the cell loss probability in B-ISDNs. For example, the maxmini residual capacity routing, which is a variation of the MUR, has been proposed and evaluated[9] [10]. Yee and Lee [11] compared 4 routing algorithms for ATM networks and concluded that the minimax criterion achieves the best trade-off between average/largest cell delay and average/largest cell loss probability. Moreover, as will be shown in Section 4, the routing decisions made by the MUR algorithm are within 2% of an optimal solution where the objective function is the minimization of the average packet delay.
- The problem can be formulated as a linear integer programming problem as compared to non-linear integer programming problems resulting from using other performance objectives such as delay or blocking probability.

Due to the discrete nature of the MUR problem in networks with single-path routing and multicasting services, to our knowledge, no published research attempted to solve the problem optimally. Tcha and Maruyama[12] developed a straightforward heuristic scheme, which is conceptually similar to the simplex method, to solve the MUR problem for virtual circuit networks. They tested the heuristic on relatively small test problems with 15 to 95 origin-destination (O-D) pairs (the test problems we considered are with 90 to 3660 O-D pairs). The error bounds in their experiments were up to 35.98%.

In this paper, the MUR problem is solved by the Lagrangean relaxation technique. We find that the dual problem

0-7803-0917-0/93$03.00 © 1993 IEEE

(transformed from the original problem) becomes two independent and easily solvable subproblems. In computational experiments, the MUR algorithm determines solutions that are within a few percent of an optimal solution for networks with up to 61 nodes in two minutes of CPU time on a SUN SPARC 490[1].

In the following sections, we first present a formal definition of the routing problem addressed in this paper. We then propose a solution approach in Section 3. Some computational results are reported in Section 4. Section 5 discusses a number of implementational issues. Section 6 summarizes this paper.

2. Problem Formulation

We model a communication network as a graph $G(V,L)$ where the switches are represented by nodes and the communication channels are represented by links. $V = \{1,2,...,N\}$ is the set of nodes and L is the set of links in the graph (network). Let W be the set of O-D pairs (single destination) in the network and M be the set of multicast groups (multiple destinations) in the network. For the sake of discussions, the terminologies used in the rest of the paper are based on those used in virtual-circuit based packet networks. However, the analysis applies to all other types of networks providing single-path routing and multicasting services. For each O-D pair $w \in W$, the mean arrival rate of new traffic is γ_w (packets/sec), while the mean arrival rate of a multicast group m is α_m (packets/sec). Let P_w be a given set of all possible simple directed paths from the origin to the destination for O-D pair w. The overall traffic for O-D pair w is transmitted over exactly one path in the set P_w. Let P be the set of all simple directed paths in the network, that is, $P = \cup_{w \in W} P_w$. For a multicast group m, let T_m be a given set of trees rooted at the origin and spanning all the destinations. The traffic is transmitted over one tree in the set T_m. Let T be the set of all trees in the network, that is, $T = \cup_{m \in M} T_m$. For each link $l \in L$, the capacity is C_l packets per second.

For each O-D pair $w \in W$, let

$$x_p = \begin{cases} 1 & \text{path } p \in P_w \text{ is used to transmit the packets for } w \\ 0 & \text{otherwise.} \end{cases}$$

Since all of the packets in a session are transmitted over a single path from the origin to the destination in a virtual circuit network, we have

$$\sum_{p \in P_w} x_p = 1.$$

For each path $p \in P$ and link $l \in L$, let

$$\delta_{pl} = \begin{cases} 1 & \text{if link } l \text{ is on path } p \\ 0 & \text{otherwise.} \end{cases}$$

For each multicast group $m \in M$, let

$$y_t = \begin{cases} 1 & \text{tree } t \in T_m \text{ is used to transmit the packets for } m \\ 0 & \text{otherwise.} \end{cases}$$

The switches in the virtual circuit network are assumed to have the capability of duplicating packets for multiple downstream destinations residing in different branches of the tree. When a packet is multicast from the root to the destinations using tree t, it is assumed that exactly one copy of the packet is transmitted over each link in tree t. Similar to the single-destination case, we have

$$\sum_{t \in T_m} y_t = 1.$$

For each tree $t \in T$ and link $l \in L$, let

$$\sigma_{tl} = \begin{cases} 1 & \text{if link } l \text{ is on tree } t \\ 0 & \text{otherwise.} \end{cases}$$

The problem of determining a path for each O-D pair and a tree for each multicast group to minimize the maximum link utilization in the network can then be formulated as the following linear combinatorial optimization problem:

$$Z_{IP'} = \min \max_{l \in L} \frac{\sum_{w \in W} \sum_{p \in P_w} x_p \gamma_w \delta_{pl} + \sum_{m \in M} \sum_{t \in T_m} y_t \alpha_m \sigma_{tl}}{C_l} \qquad \text{(IP')}$$

subject to:

$$\sum_{w \in W} \sum_{p \in P_w} x_p \gamma_w \delta_{pl} + \sum_{m \in M} \sum_{t \in T_m} y_t \alpha_m \sigma_{tl} \le C_l \qquad \forall l \in L \quad (1)$$

$$\sum_{p \in P_w} x_p = 1 \qquad \forall w \in W \quad (2)$$

$$\sum_{t \in T_m} y_t = 1 \qquad \forall m \in M \quad (3)$$

$$x_p = 0 \text{ or } 1 \qquad \forall p \in P_w,\ w \in W \quad (4)$$

$$y_t = 0 \text{ or } 1 \qquad \forall t \in T_m,\ m \in M. \quad (5)$$

The objective function represents the minimax link utilization in the network. Terms in the left hand side of Constraint (1) denote the aggregate flow of packets over link l. Constraint (1) requires that the aggregate flow not exceed the capacity of each link. Constraints (2) and (4) require that all of the traffic for one O-D pair is transmitted over exactly one path. Similarly, Constraints (3) and (5) require that all of the traffic for one multicast group is transmitted over exactly one tree.

Let

$$s = \max_{l \in L} \frac{\sum_{w \in W} \sum_{p \in P_w} x_p \gamma_w \delta_{pl} + \sum_{m \in M} \sum_{t \in T_m} y_t \alpha_m \sigma_{tl}}{C_l}.$$

An equivalent formulation of IP' is

$$Z_{IP} = \min s \qquad \text{(IP)}$$

subject to:

$$\sum_{w \in W} \sum_{p \in P_w} x_p \gamma_w \delta_{pl} + \sum_{m \in M} \sum_{t \in T_m} y_t \alpha_m \sigma_{tl} \le C_l\, s \qquad \forall l \in L \quad (6)$$

$$0 \le s \le 1 \quad (7)$$

$$\sum_{p \in P_w} x_p = 1 \qquad \forall w \in W \quad (8)$$

$$\sum_{t \in T_m} y_t = 1 \qquad \forall m \in M \quad (9)$$

$$x_p = 0 \text{ or } 1 \qquad \forall p \in P_w,\ w \in W \quad (10)$$

$$y_t = 0 \text{ or } 1 \qquad \forall t \in T_m,\ m \in M. \quad (11)$$

Constraints (8)-(11) are the same as Constraints (2)-(5). Constraints (6) and (7) require that the utilization of each link not exceed s (and unity). Constraint (1) is therefore redundant and can be eliminated.

3. Solution Procedure

Lagrangean Relaxation (LR):

The basic approach to solve the linear programming problem formulated in the previous section is Lagrangean Relaxation. A Lagrangean relaxation is obtained by identifying in the primal problem a set of complicating constraints whose removal will simplify the solution of the primal problem. Each of the complicating constraints is multiplied by a multiplier and added to the objective function. This mechanism is referred to as

1. Bellcore does not recommend or endorse products or vendors. Any mention of a product or vendor in this paper is to indicate the computing environment for the computational experiments discussed or to provide an example of technology for illustrative purposes; it is not intended to be a recommendation or endorsement of any product or vendor. Neither the inclusion of a product or a vendor in a computing environment or in this paper, nor the omission of a product or vendor, should be interpreted as indicating a position or opinion of that product or vendor on the part of the authors or Bellcore.

dualizing the complicating constraints.

In our solution approach to (IP), we dualize Constraint (6) of (IP) to obtain the following relaxation:

$$Z_D(u) = \min \left\{ s + \sum_{l \in L} u_l \, G \right\} \qquad \text{(LR)}$$

where

$$G = \sum_{w \in W} \sum_{p \in P_w} x_p \gamma_w \delta_{pl} + \sum_{m \in M} \sum_{t \in T_m} y_t \alpha_m \sigma_{tl} - C_l \, s$$

subject to:

$$0 \leq s \leq 1 \qquad (12)$$

$$\sum_{p \in P_w} x_p = 1 \quad \forall w \in W \qquad (13)$$

$$\sum_{t \in T_m} y_t = 1 \quad \forall m \in M \qquad (14)$$

$$x_p = 0 \text{ or } 1 \quad \forall p \in P_w, \ w \in W \qquad (15)$$

$$y_t = 0 \text{ or } 1 \quad \forall t \in T_m, \ m \in M. \qquad (16)$$

The solution to (LR) is for every O-D pair w and every multicast group m, to route all of the required traffic over a shortest path and a minimum cost tree, respectively, where the arc weight of link l is u_l. To determine s, there are two cases:

1. If $\sum_{l \in L} u_l \, C_l \geq 1$, then $s = 1$.
2. If $\sum_{l \in L} u_l \, C_l < 1$, then $s = 0$.

Dual (D):

For any $u \geq 0$, by the weak Lagrangean duality theorem[13], the optimal objective function value of (LR), $Z_D(u)$, is a lower bound on Z_{IP}. The dual problem (D) is

$$Z_D = \max_{u \geq 0} Z_D(u). \qquad \text{(D)}$$

Therefore, in order to obtain the greatest lower bound, we solve the dual problem (D). There are several methods for solving the dual problem (D), of which the subgradient method[14] is the most popular and is employed here. Let an $|L|$-tuple vector b be a subgradient of $Z_D(u)$. Then, in iteration k of the subgradient optimization procedure, the multiplier for each link $l \in L$ is updated by[14]

$$u_l^{k+1} = u_l^k + t^k b_l^k. \qquad (17)$$

The step size t^k is determined by[14]

$$t^k = \delta \, \frac{Z_{IP}^h - Z_D(u^k)}{\|b^k\|^2} \qquad (18)$$

where Z_{IP}^h is the objective function value for a heuristic solution (upper bound on Z_{IP}) and δ is a constant, $0 < \delta \leq 2$. In our implementation (results of which are described in the next section), Z_{IP}^h was initially chosen as 1 and updated to the best upper bound found so far in each iteration. In addition, δ was initially set to 2 and halved whenever the objective function value did not improve in 25 iterations. The initial values of the multipliers were chosen to be 0.

In order to find better lower bounds, we further employ the following mechanism: First, we temporarily choose the multiplier $u_l = u_l \, (\sum_{l \in L} u_l \, C_l)^{-1}$ for link l so that $\sum_{l \in L} u_l \, C_l = 1$. It can be verified that this temporary multiplier is a break point of $Z_D(u)$ (since s has a discrete jump at this point) and potentially corresponds to a higher dual objective function value than at u. Note that using the temporary multiplier u_l results in the same choice of paths as using the original u_l. We also note that when all of the arc weights (multipliers) are multiplied by a positive scalar κ, the shortest paths found in solving (LR) do not change. It is only for the purpose of calculating $Z_D(u)$, we temporarily use the multiplier to attempt to find a higher dual objective function value by a simple calculation. The original value of the multiplier u_l will still be used in the subgradient method to update the multiplier for the next iteration. The above procedure is for solving the dual problem and obtaining good lower bounds on the optimal objective function value. We next describe a procedure for finding good primal solutions.

Primal Solutions:

In each iteration of solving (D), where an (LR) is solved, a shortest path for each O-D pair and a minimum cost tree for each multicast group are found. If these routing assignments also satisfy the capacity constraints (for each link the aggregate flow does not exceed the link capacity), then the routing assignments are primal feasible and is considered as a heuristic solution. The maximum link utilization factor is calculated and the best heuristic solution found in the course of solving (D) is reported.

4. Computational Results

Since the mechanism to determine the routing assignments for multicast traffic is similar to that for the individually addressed traffic, we only consider the individually addressed traffic in our current computational experiments. Three sets of computational experiments are performed:

1. In the first set of experiments, we test the MUR algorithm with respect to its (i) computational efficiency and (ii) effectiveness in determining good solutions.
2. In the second set of experiments, we quantify how much the maximal link utilization can be reduced by the MUR algorithm compared with the minimum hop routing (MHR) algorithm.
3. In the third set of experiments, we evaluate the effectiveness of applying the MUR algorithm on the virtual circuit routing problem where the objective is to minimize the average packet delay.

In the following, we describe the experimental results:

1. The MUR algorithm in networks with single-path routing and multicasting services described in Section 3 was coded in FORTRAN 77 and run on a SUN SPARC 490 server. For the first set of experiments, the algorithm was tested on three networks: ARPA1 (61 nodes), RING (32 nodes), and OCT (26 nodes) whose topologies are shown in [15]. For each of the three networks, it was assumed that for each O-D pair there were at most three candidate routes. For each O-D pair, shortest paths were found with respect to 3 sets of randomly generated utilization factors (arc weights) and the distinct paths were used as candidate routes. Our previous research[15] showed that considering all possible simple paths will improve the objective function value for a few percent at the cost of doubling the computation time. For each of the three networks, it was assumed that for each O-D pair the total traffic rate at which packets are generated is 1 packet per second.

   Table 1 summarizes the results of our computational experiments with the MUR algorithm. The second column specifies the capacity of each link (in packets per second) in each network. The third column is the largest lower bound on the optimal objective function value found in the number of iterations specified in the seventh column. Recall that this is the best objective function value of the dual problem. In addition to the mechanism described in Section 3 to improve the lower bound, we attempt another mechanism based upon the fact that the set of possible objective function values for (IP) is discrete (equal to the number of O-D pairs using the most congested link divided by the link capacity) in the experiments. The fourth column gives the best objective function value for (IP) in the number of iterations specified in the seventh column. The percentage difference (|upper-bound − lower-bound| × 100 / lower-bound) is an upper bound on how far the best feasible solution found is from an optimal solution. The sixth column provides the CPU times (in seconds) which include the time to input the problem parameters. Table 1 shows that the MUR algorithm is efficient and effective in finding near-optimal solutions. For every test problem (networks with up to 61 nodes), the algorithm determines a solution that is within a few percent of an optimal solution in less than 2 minutes of CPU time on a SUN SPARC 490.

2. Another set of experiments were performed to compare the MUR algorithm with the minimum hop routing (MHR) algorithm. In the MHR, for each O-D pair a path was found with the fewest number of links (ties were broken at random). In order to make the experiments realistic, we chose many real network topologies. The ten network topologies used in the test problems are shown in [15]. With the exceptions of the RING and the OCT being fictitious, the other eight topologies have been practically implemented. The link capacities for each test network are assumed to be the same (homogeneous networks). The mean traffic demand for each O-D pair in each of the test networks is assumed to be 1 packet per second. For the purpose of illustration, we use the maximal aggregate link flow instead of the maximal link utilization factor as the performance measure (these two measures are equivalent for homogeneous networks).

   A comparison of the performance of the MUR algorithm with the performance of the MHR algorithm is reported in Table 2. The second column specifies the average node degree $|L|/|V|$ for each of the networks. This is a measure of the richness of the network topology or the number of paths between each user pair. The third column reports Z^{MUR} (in packets per second), the best primal objective function value found by the MUR algorithm. The fourth column reports Z^{MHR} (in packets per second) which is the objective function value found by applying a MHR algorithm. The fifth column gives the percentage improvement of the MUR over the MHR.

   The results in Table 2 show that using the MUR algorithm results in an improvement in the maximum link utilization. The range of improvements is from 16.67% to 75.00%. In addition, the improvement in the maximum link utilization factor is highly correlated with the richness of the network topology. In comparing columns 2 and 5, one observes that the improvement is greater when the average nodal degree is high. This is due to the fact that for a network with a high nodal degree (and more paths with a small number of hops) the MUR algorithm has more flexibility in choosing good paths. Another observation from Table 2 is that 5 out of 10 test problems were solved optimally by the MUR algorithm. The optimality was verified by the coincidence of the upper and lower bounds.

3. We next show by an example that the minimax criterion (a linear performance measure) can serve as a good surrogate for other performance criteria which may be complicated, if known, functions of aggregate flows. Lin and Yee[15] considered the virtual circuit routing problem where the objective is to minimize the average packet delay. In their formulation, each link was modeled as an $M/M/1$ queue. We repeated the set of experiments reported in Table 1 and used the final routing assignments (obtained by applying the MUR algorithm) to calculate the average packet delay, using the performance measure in Lin and Yee[15]. We then implement Lin and Yee's algorithm to calculate lower bounds on the minimum average packet delay (their algorithm provides both upper and lower bounds), which were used to evaluate the quality of the minimax criterion.

   We report the result in Table 3. The third column shows the average packet delay D_{MUR} which is obtained by using the routing assignment determined by the MUR algorithm. The fourth column gives lower bounds on the minimum average packet delay D_{LB} (using the lower bounding scheme in Lin and Yee[15]). The fifth column shows the percentage difference between columns three and four.

   From an inspection of Table 3, the minimax utilization criterion is a very good surrogate for the objective of minimizing the average packet delay. The routing decisions made by the MUR algorithm are within 2% of an optimal solution where the objective function is the minimization of the average packet delay.

5. Implementation Issues

Network routing protocols are either left to the specific vendor implementation (e.g., X.25) or specified in standards (e.g., SS7[16] and SMDS ISSI[17]). Therefore, once the routing protocol has been defined and implemented, it is difficult to improve the network performance by changing the underlying routing procedure (i.e., routing procedure change requires vendor software and possibly hardware adjustments). In addition, as we have mentioned, the networks with single-path routing service implemented today largely use "static" routing algorithms. The routing tables are not updated according to current network traffic levels which simplifies the design, however, at the expense of performance. As described below, the "quasi-static" MUR algorithm discussed in this paper can be used to improve the performance of any existing "static" routing procedure without the costly vendor equipment changes.

The MUR computation described in Section 3 can be performed centrally, for example, in a network operations support system (OSS) such as the INPLANS® system[18]. Raw network traffic measurements are collected and sent to the OSS for data processing (validating and aggregating) and performing the routing computation. The traffic data required for the MUR algorithm is the point-to-point traffic demand for each O-D pair in the network. However, most network switching equipment does not collect the point-to-point traffic loads. Therefore, estimation of the point-to-point traffic loads[19] needs to be performed based on the available switch and link measurements. The principal idea of the estimation method for point-to-point traffic loads from link measurements is the Moore-Penrose pseudo-inverse[20]. The pseudo-inverse scheme has been proven to provide the optimal estimate in term of the variance of estimation errors. However, it is possible that some of the aggregate link flows are underestimated when the above optimal estimate of end-to-end traffic requirements are used and the routing assignments are changed. One conservative estimation scheme is proposed below. The basic idea is to calculate the maximum value of each point-to-point traffic requirement subject to the aggregate link flow information (and perhaps other information to increase the estimation accuracy, e.g. the total external traffic requirement to each switch). Then the aggregate link flows, given any routing assignment, will not be underestimated. For each O-D pair, the problem is formulated as a linear programming problem. Unfortunately, no special structure of the linear programming problem has been identified so that more efficient algorithms than the simplex method can be applied. Nevertheless, since the basic idea is to obtain worst case estimates, one may apply the Lagrangean relaxation technique to efficiently calculate tight upper bounds on the optimal objective function value.

The estimated point-to-point demands are used as input to the MUR computation for obtaining the near-optimal routing decisions. These decisions are then used to update the routing tables maintained in the switches. Depending on the desirable performance, the whole process described above can be implemented on different time scale such as every 5 minutes, 30 minutes, hourly, etc. This, of course, also depends on the traffic data collection schedules. For longer term operations such as daily or weekly, the point-to-point demands may need to be trended before they are used for routing computation to account for the demand forecasts. Figure 1 depicts a process for implementing the MUR procedure.

6. Summary

In this paper, we propose a near-optimal quasi-static MUR algorithm in networks with single-path routing and multicasting

INPLANS is a trademark of Bellcore.

services. We formulate the problem as an integer programming problem where the objective is to minimize the maximum link utilization factor. A number of advantages of using the minimax criterion are discussed.

The basic approach to the algorithm development is Lagrangean relaxation. Since the proposed algorithm does not require to be performed in real time, the computational complexity is not a major concern as long as it is reasonable. In the computational experiments, the proposed algorithm was able to determine solutions that are within a few percent of an optimal solution for networks with up to 61 nodes in 2 minutes of CPU time of a SUN SPARC 490 server.

The proposed MUR algorithm is compared with the MHR algorithm. The performance measure is the maximum link utilization factor. Ten network topologies, most of which have been practically implemented, are used in this set of experiments. The results show that using the MUR algorithm results in significant improvement in the maximum link utilization over the minimum hop routing scheme. The range of improvements is from 16.67% to 75.00%. In addition, the improvement in the maximum link utilization factor is highly correlated with the richness of the network topology. We also show that the routing decisions made by the MUR are within 2% of an optimal solution where the objec is to minimize the average packet delay.

A number of issues involved in implementing the MUR algorithm are discussed. The routing computation and the eventual routing table updates of the proposed algorithm can be realized through centralized OSS. The major issue in implementing the algorithm is to obtain point-to-point traffic requirements. A method for calculating conservative point-to-point load estimates from available switch traffic measurements is proposed and currently under study.

REFERENCES

1. "PPSN Generic Requirements", Bellcore Technical Reference, TR-TSY-000301, Issue 1, September 1986.
2. J. P. Gray and T. B. McNeill, "SNA Multiple-System Networking", *IBM System Journal*, vol. 18, pp. 263-297, 1979.
3. "Functional Description of GTE Telenet Packet Switching Networks", GTE Telenet Communications Corp., Vienna, VA, May 1982.
4. L. R. Tymes, "Routing and Flow Control in TYMNET", *IEEE Trans. Commun.*, COM-29, pp. 392-398, 1981.
5. A. Danet, *et. al.*, "The French Public Packet Switching Service: The TRANSPAC Network", in *Proc. Third ICCC*, pp. 251-260, 1976.
6. "*Generic System Requirements in Support of Switched Multi-megabit Data Service*", Bellcore Technical Reference, TR-TSV-000772, Issue 1, May 1991.
7. Special Issue on Wide Area Private Networks, *IEEE Commun. Mag.*, Vol. 30, No. 3, March 1992.
8. CCITT Draft Recommendation I.150, "B-ISDN ATM Functional Characteristics", Geneva, May 1990.
9. A. Gersht and A. Shulman, "Optimal Routing in Circuit Switched Communication Networks", *IEEE Trans. Commun.*, Vol. 37, No. 11, pp. 1203-1211, November 1989.
10. A. Girard and M. Bell, "Blocking Evaluation for Networks with Residual Capacity Adaptive Routing", *IEEE Trans. Commun.*, Vol. 37, No. 11, pp. 1372-1380, December 1989.
11. J. R. Yee and M. J. Lee, "Convergence of an Iterative Method for ATM Networks", the *Proceedings of INFOCOM*, 1992.
12. D. Tcha and K. Maruyama, "On the Selection of Primary Paths for a Communication Network", *Computer Networks and ISDN Systems*, Vol. 9, pp. 257-265, 1985.
13. A. M. Geoffrion, "Lagrangean Relaxation and Its Use in Integer Programming", *Math. Programming Study*, Vol. 2, pp. 82-114, 1974.
14. M. Held, P. Wolfe and H. D. Crowder, "Validation of Subgradient Optimization", *Math. Programming*, Vol. 6, pp. 62-88, 1974.
15. F. Y.S. Lin and J. R. Yee, "A New Multiplier Adjustment Procedure for the Distributed Computation of Routing Assignments in Virtual Circuit Data Networks", *ORSA Journal on Computing*, Vol. 4, No. 3, Summer 1992.
16. "Bell Communications Research Specification of Signaling System Number 7," Bellcore Technical Reference, TR-NWT-000246, Issue 2, June 1991.
17. "Inter-Switching System Interface Generic Requirements in Support of SMDS Service," Bellcore Technical Advisory, TA-TSV-001059, Issue 2, September 1992.
18. J. L. Wang, "INPLANS - An Integrated Planning and Engineering System for Telecommunication Networks", in *Proc. 13th International Teletraffic Congress*, pp. 137-142, Copenhagen, Denmark, June 1991.
19. M. Tu, "Estimation of Point-to-Point Traffic Demands in Dynamic Routing Networks", in *Proc. 13th International Teletraffic Congress*, Copenhagen, Denmark, June 1991.
20. A. Albert, *Regression and the Moore-Penrose Pseudo-Inverse*, Academic Press, New York, 1972.

Network ID	Cap.	Lower Bounds	Upper Bounds	% Diff.	CPU Time	No. Iter.
ARPA1	500	0.516	0.528	2.326	64.6	500
RING	60	0.500	0.517	3.333	21.5	700
OCT	100	0.600	0.600	0.000	18.9	1000

Table 1. Summary of computational results of the MUR algorithm

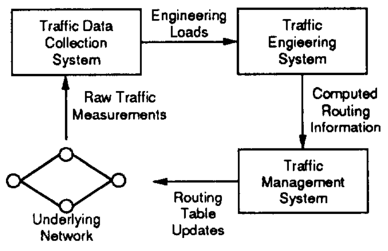
Network ID	$\frac{\vert L \vert}{\vert V \vert}$	Z^{MUR}	Z^{MHR}	$\frac{Z^{MHR} - Z^{MUR}}{Z^{MUR}}$ (%)
ARPA1	2.43	264	356	34.85
RING	3.75	31	44	41.94
OCT	2.31	60†	70	16.67
ARPA2	2.48	41	48	17.07
NORDIC	3.69	12	17	41.67
SWIFT	2.53	22†	26	18.18
SITA	5.60	4	7	75.00
PSS	3.00	13†	18	38.46
GTE	4.17	7†	12	71.43
TRANSPAC	3.67	13†	16	23.08

† Optimal solution was found.

Table 2. Comparison of the maximal link utilization obtained by the MUR algorithm and by the MHR algorithm

Network ID	D_{MUR} (msec)	D_{LB} (msec)	Percentage Difference (%)
ARPA1	17.5205	17.3970	0.710
RING	90.1949	88.8735	1.487
OCT	87.2603	86.1749	1.260

Table 3. Average packet delay obtained by using the routing assignments determined by the MUR algorithm



Traffic Data Collection System
Engineering Loads
Traffic Engieering System
Raw Traffic Measurements
Computed Routing Information
Traffic Management System
Routing Table Updates
Underlying Network

Figure 1. An implementation process for the proposed routing algorithm