# A Routing Algorithm for Virtual Circuit Data Networks with Multiple Sessions Per O–D Pair

**James R. Yee***
*Department of Electrical Engineering*
*University of Hawaii at Manoa*
*Honolulu, Hawaii 96822*

**Frank Y. S. Lin†**
*Bellcore*
*3 Corporate Place, PYA2J318*
*Piscataway, New Jersey 08854*

In virtual circuit networks, all the packets in a session are transmitted over exactly one path established between the origin and the destination. For each origin–destination pair, it is assumed that there are multiple sessions. We consider the problem of choosing a path for each session so as to minimize the average packet delay in the network. We formulate this problem as a nonlinear multicommodity flow problem with integer decision variables. An iterative scheme that is similar to local search is developed to solve this problem. In each iteration, we apply Lagrangean relaxation and a multiplier adjustment procedure to solve a restricted problem. We show that the Lagrangean dual problem can be solved exactly by solving a convex program. In computational experiments, our algorithm determines solutions that are within 1% of an optimal solution in minutes of CPU time for networks with 26–61 nodes. In addition, we show that our proposed algorithm is better both theoretically and computationally than K(0)-ordering, single-path routing, or round-off Frank–Wolfe heuristics.

## 1. INTRODUCTION

The ARPANET inspired a great deal of research on computer communication networks with datagram service. In datagram networks, the packets sent for a particular user pair may be routed over different paths. The routing problem in datagram networks has been extensively studied. Centralized [4, 7] and distributed [3, 8, 21] routing algorithms have been proposed. More recently, networks such as SNA [11, 14], TYMNET [17, 20], TELENET [12],

*Supported by a contract from AT&T, by Contract DAAL 03-88-K0059 from the Army Research Office, and by a grant NCR-9016348 from NSF.
†Supported by a contract from AT&T.

NETWORKS, Vol. 22 (1992) 185–208

CCC 0028-3045/92/020185-24$04.00

and TRANSPAC [6] have been designed to provide virtual circuit service. In virtual circuit service, all the packets transmitted for a particular user pair (a session) are sent over exactly one path. The primary motivation for using this type of service is that all the packets arrive at the destination in the order in which they were sent. Note that in datagram networks packets can arrive out of order and thus must be reordered at the destination. Furthermore, most of the future networks such as ISDN will provide virtual circuit service [10].

The routing problem in datagram networks is usually formulated as a convex multicommodity flow problem. In contrast, the combinatorial nature of the routing problem in virtual circuit networks results from the fact that a session must be assigned to exactly one path. Courtois and Semal [5] modified the flow deviation method [7] to develop a heuristic routing algorithm for virtual circuit networks. The flow deviation method is the Frank–Wolfe method specifically tailored to solve an uncapacitated multicommodity flow problem with a convex objective function. They obtained solutions that are within 3% (on the average) of the optimal solutions for lightly loaded networks. However, no results were reported for heavily loaded networks. Segall [18] formulated the routing problem in virtual circuit networks as a convex programming problem and modified Gallager's algorithm [8] to develop a distributed routing algorithm. However, in his work, the routing decision variables were assumed to be continuous. Therefore, the combinatorial nature of the virtual circuit routing problem is not captured. Tsai [19] studied the convergence of gradient projection methods [2] in an asynchronous stochastic quasi-static virtual circuit network. He showed that the algorithms converge to a neighborhood of a long-term optimal routing. In addition, the neighborhood converges to zero if the number of virtual circuits approaches infinity while the total traffic requirement remains the same. In other words, the traffic requirement of each session needs to approach zero in order to obtain an optimal routing. With the "many small users" assumption, the inherent discrete structure of the virtual circuit routing problem was ignored.

Gavish and Hantler [9] considered the problem of selecting the first route from a candidate route set for each origin–destination pair in virtual circuit networks to minimize the average packet delay. They formulated the problem as a nonlinear multicommodity flow problem with 0-1 decision variables. They applied Lagrangean relaxation and the subgradient optimization method to develop a centralized algorithm to solve the problem. Their computational results showed that their heuristic algorithm is effective in finding good feasible solutions and determining tight lower bounds on the minimal expected delay.

Lin and Yee [16] slightly modified the formulation in [9] and developed a new multiplier adjustment procedure so that the problem can be solved by distributed computation. A common weakness of [9] and [16] is that all the traffic for each origin–destination (O–D) pair is routed over one path. For the case of "large and balanced networks" [15], this single-path routing (they called it fixed routing) is effective. A network is referred to as balanced if the traffic requirements of the O–D pairs do not differ by very much. However, in general, the minimal achievable average delay is smaller when the routing can

route sessions over a number of paths than when the algorithm is constrained to use one path for each session for each O–D pair.

In this paper, we consider the virtual circuit routing problem where (i) there are a number of sessions for each O–D pair and (ii) different sessions for an O–D pair may be assigned to different paths. A formulation of this problem is given and an algorithm is developed based upon Lagrangean relaxation.

The remainder of this paper is organized as follows: In Section 2, the routing problem is formulated as a nonlinear combinatorial optimization problem. In Section 3, a Lagrangean relaxation approach to the problem is presented. In Section 4, a solution procedure to the routing problem is presented. In Section 5, the computational results are reported.

## 2. PROBLEM FORMULATION

In this section, we generalize the models in [9] and [16] where there is one session for each O–D pair. A virtual circuit communications network is modeled as a graph where the processors are represented by nodes and the communication channels are represented by arcs. Let $V = \{1, 2, \ldots, N\}$ be the set of nodes in the graph, and let $L$ denote the set of communication links in the network. Let $W$ be the set of O–D pairs in the network. For each O–D pair $w \in W$, multiple sessions may be established and the arrival of new traffic for each session is modeled as a Poisson process with rate $\gamma$ (packets/second). Let $n_w$ be the total number of sessions for O–D pair $w$. Then, the arrival of new traffic to the network is a Poisson process with rate $\Gamma = \Sigma_w n_w \gamma$. For O–D pair $w$, the traffic for a particular session is transmitted over one path in the set $P_w$, a given set of simple directed paths from the origin to the destination of O–D pair $w$. Let $P$ be the set of all simple directed paths in the network. For each link $l \in L$, the capacity is $C_l$ packets/second.

For each O–D pair $w \in W$, let $x_p$ be the number of sessions assigned to path $p \in P_w$. For each path $p$ and link $l \in L$, let $\delta_{pl}$ denote an indicator function that is one if link $l$ is on path $p$ and zero otherwise. Then, the aggregate flow of packets over link $l$ is given by the left-hand side of (1).

In the network, there is a buffer for each outbound link. Using Kleinrock's independence assumption [15], the arrival of packets to each buffer is a Poisson process where the rate is the aggregate flow over the outbound link. It is assumed that the transmission time for each packet is exponentially distributed with mean $C_l^{-1}$. Thus, each buffer can be modeled as an M/M/1 queue.

The problem of determining a path for each session to minimize the average delay is formulated below as a nonlinear combinatorial optimization problem:

$$Z_{\overline{IP}} = \min \frac{1}{\Gamma} \sum_{l \in L} \frac{\Sigma_{w \in W} \Sigma_{p \in P_w} x_p \gamma \delta_{pl}}{C_l - \Sigma_{w \in W} \Sigma_{p \in P_w} x_p \gamma \delta_{pl}}, \qquad (\overline{IP})$$

subject to

$$\sum_{w \in W} \sum_{p \in P_w} x_p \gamma \delta_{pl} \leq C_l \qquad \forall\, l \in L \qquad (1)$$

$$\sum_{p \in P_w} x_p = n_w \qquad \forall\, w \in W \tag{2}$$

$$x_p = \text{nonnegative integer} \qquad \forall\, p \in P_w,\ w \in W. \tag{3}$$

The objective function represents the average packet delay in the network. Constraint (1) requires that the aggregate flow not exceed the capacity for each link. Note that this constraint should be written as a strict inequality and (1) is a relaxation. Using this relaxation results in an equivalent problem since the objective function serves as a barrier function that restricts the aggregate link flow to be less than the link capacity. Constraint (2) ensures that for each O–D pair all the sessions are serviced. Constraint (3) requires that the number of sessions assigned to a path be a nonnegative integer.

In the above formulation, sessions for an O–D pair may be established over several different paths. We refer to this as multiple-path routing. In [9] and [16], all the traffic for an O–D pair is routed over a single path. We refer to this type of routing assignment as single-path routing. It can be shown that ($\overline{IP}$) with the additional constraint

$$x_p = 0 \text{ or } n_w \qquad \forall\, p \in P_w,\ w \in W$$

is equivalent to the single-path routing formulation given in [9] and [16]. This additional constraint requires that for each O–D pair $w$, a path $p \in P_w$ carries either none or all of the traffic for O–D pair $w$. Let $Z_{IPSP}$ be the optimal objective function value for the single-path routing problem. Note that the set of feasible solutions for single-path routing is a proper subset of the set of feasible solutions for multiple-path routing. From this observation, $Z_{\overline{IP}} \leq Z_{IPSP}$.

A variation on the above formulation is $P_w$ to be the set of all simple paths for O–D pair $w$. By doing so, the set of feasible solutions would be enlarged and thus the minimal achievable average delay would be smaller. The solution procedure described in Section 4 can be easily modified to handle this change in the formulation. However, in our computational experiments, we found that this change in implementation resulted in a reduction of the minimal average delay of less than 5% while the computation time increased by 50–100% compared with the implementation where $|P_w|$ is at most 3.

An equivalent formulation of the above problem is given by ($IP$) below. Let $\{x_p^0\}$ be a feasible solution to Problem ($\overline{IP}$). As a result, ($\overline{IP}$) can be rewritten in the following form:

$$Z_{IP} = \min \sum_{l \in L} \frac{f_l}{C_l - f_l}, \tag{IP}$$

subject to

$$\sum_{w \in W} \sum_{p \in P_w} (x_p^0 + \Delta_p)\gamma\delta_{pl} \leq f_l \qquad \forall\, l \in L \tag{4}$$

$$0 \le f_l \le C_l \qquad \forall\, l \in L \tag{5}$$

$$\sum_{p \in P_w} (x_p^0 + \Delta_p) = n_w \qquad \forall\, w \in W \tag{6}$$

$$x_p^0 + \Delta_p \ge 0 \qquad \forall\, p \in P_w,\ w \in W \tag{7}$$

$$\Delta p = \text{integer} \qquad \forall\, p \in P_w,\ w \in W. \tag{8}$$

Since $\Gamma$ is a constant, it can be eliminated from the objective function. For each link $l$, a variable $f_l$ is introduced. We interpret these variables to be "estimates" of the aggregate flows. Since the objective function is strictly increasing with $f_l$ and $(IP)$ is a minimization problem, each $f_l$ will equal the aggregate flow in an optimal solution. In the Lagrangean relaxation, the introduction of $\{f_l\}$ has the effect of decoupling the problem into two separate subproblems. In $(IP)$, $\{x_p^0\}$ is the current routing assignment and is fixed. $\{\Delta_p\}$ represents the change from the current routing assignment. The set of feasible routing assignments of $(IP)$, $\{x_p^0 + \Delta_p\}$, is the same as the set of feasible assignments for $(\overline{IP})$. Thus, $(IP)$ is equivalent to $(\overline{IP})$. In the next section it will be shown that the introduction of $\{\Delta_p\}$ facilitates the development of an iterative scheme to find near-optimal multiple-path routing assignments.

At each iteration $k_1 \ge 1$, we solve the following "restricted" problem:

$$Z_{RP} = \min \sum_{l \in L} \frac{f_l}{C_l - f_l}, \tag{RP}$$

subject to

$$\sum_{w \in W} \sum_{p \in P_w} (x_p^{k_1 - 1} + \Delta_p) \gamma \delta_{pl} \le f_l \qquad \forall\, l \in L \tag{9}$$

$$0 \le f_l \le C_l \qquad \forall\, l \in L \tag{10}$$

$$\sum_{p \in P_w} (x_p^{k_1 - 1} + \Delta_p) = n_w \qquad \forall\, w \in W \tag{11}$$

$$x_p^{k_1 - 1} + \Delta_p \ge 0 \qquad \forall\, p \in P_w,\ w \in W \tag{12}$$

$$\Delta_p = 0,\ 1,\ or\ -1 \qquad \forall\, p \in P_w,\ w \in W, \tag{13}$$

where $\{x_p^{k_1 - 1}\}$ is the routing decision obtained by solving $(RP)$ at the $(k_1 - 1)$th iteration. It should thus be noted that $\{x_p^{k_1 - 1}\}$ is fixed in the course of solving $(RP)$ and the decision variables are $\{\Delta_p, f_l\}$. The last constraint in $(RP)$ restricts the change of traffic on a particular candidate route not to exceed one session at

each iteration. The motivation for choosing to solve (*IP*) by solving a sequence of (*RP*)'s is that better routing decisions may be found especially for a heavily loaded or unbalanced network where the bifurcation of traffic is needed. The introduction of $\{\Delta_p\}$ permits the splitting of sessions for each O–D pair. In contrast, the solution of the Lagrangean relaxation in [9] and [16] results in all of the sessions for a particular O–D pair to be assigned to the same path.

## 3. LAGRANGEAN RELAXATION AND DUAL PROBLEM

In this section, Lagrangean relaxation is applied to solve (*RP*). As in [9] and [16], we dualize constraints (9) to obtain the following relaxation:

$$Z_D(u) = \min \sum_{l \in L} \frac{f_l}{C_l - f_l} + \sum_{l \in L} u_l \left\{ \sum_{w \in W} \sum_{p \in P_w} (x_p^{k-1} + \Delta_p)\gamma\delta_{pl} - f_l \right\}, \qquad (RPLR)$$

subject to

$$0 \le f_l \le C_l \qquad \forall\, l \in L \qquad (14)$$

$$\sum_{p \in P_w} (x_p^{k_1-1} + \Delta_p) = n_w \qquad \forall\, w \in W \qquad (15)$$

$$x_p^{k_1-1} + \Delta_p \ge 0 \qquad \forall\, p \in P_w,\ w \in W \qquad (16)$$

$$\Delta_p = 0,\ 1,\ or\ -1 \qquad \forall\, p \in P_w,\ w \in W. \qquad (17)$$

Note that (*RPLR*) is composed of the following two independent subproblems:

Subproblem 1:

$$Z_D^1(u) = \min \sum_{l \in L} \left\{ \frac{f_l}{C_l - f_l} - u_l f_l \right\},$$

subject to

$$0 \le f_l \le C_l \qquad \forall\, l \in L \qquad (18)$$

and

Subproblem 2:

$$Z_D^2(u) = \min \left\{ \sum_{w \in W} \sum_{p \in P_w} \Delta_p \gamma \left\{ \sum_{l \in L} u_l \delta_{pl} \right\} + \text{constant} \right\},$$

subject to

$$\sum_{p \in P_w} (x_p^{k_1-1} + \Delta_p) = n_w \qquad \forall\, w \in W \tag{19}$$

$$x_p^{k_1-1} + \Delta_p \geq 0 \qquad \forall\, p \in P_w,\ w \in W \tag{20}$$

$$\Delta_p = 0,\ 1,\ or\ -1 \qquad \forall\, p \in P_w,\ w \in W. \tag{21}$$

Subproblem 1 is composed of $|L|$ (one for each link) simple problems. Each of these simple problems is the minimization of a convex univariate function over a simple interval. For each link $l \in L$, the solution is

$$f_l = \begin{cases} C_l\left(1 - \sqrt{\dfrac{1}{u_l C_l}}\right) & \text{if } u_l > \dfrac{1}{C_l} \\ 0 & \text{otherwise.} \end{cases} \tag{22}$$

Subproblem 2 consists of $|W|$ (one for each O–D pair) independent problems. For O–D pair $w$, the subproblem involves rearranging the distribution of the $n_w$ sessions among the candidate route set $P_w$ under the constraint that the change in the number of sessions on each path may not exceed one. Define the cost of path $p$ to be $\Sigma_{l \in h_p} u_l$, where $h_p$ represents the set of the links on path $p$. A solution to this problem is the following. For each O–D pair $w$, find a pair of paths that has the largest difference in cost. Then, shift one session from the more expensive path to the other so that the nonnegativity constraint (20) and the session-exchange constraint (21) are not violated. This procedure is repeated until no more such pair of paths can be found. However, a more systematic way to solve subproblem 2 is presented below. For each O–D pair $w$, sort the candidate routes in increasing order according to the path costs. If the number of sessions on the most expensive path is positive, shift one session to the least expensive path; otherwise, check the second most expensive path. Delete those paths that have been checked or have had traffic changes from the path set and repeat this procedure until the path set becomes empty.

Other forms of constraint (17) may be used. For example, constraint (17) can be replaced by

$$\Delta_p = -1,\ 0,\ 1,\ 2,\ \ldots \qquad \forall\, w \in W,\ p \in P_w. \tag{23}$$

Then, the solution to each of the $|W|$ subproblems is simply to shift one session from every path that carries at least one session to the shortest path and, therefore, we only need to keep track of those paths with nonzero flow but not all of the possible paths as in the above algorithm. The choice of constraint (17) should be such that (i) the resulting subproblem should be easy to solve and (ii) the number of sessions that are rerouted be limited in each iteration.

Our approach in solving (*IP*) is to solve a sequence of (*RP*)'s where (17) is

used to limit the rerouting of sessions in each iteration. The motivation for this approach can be appreciated by considering the solution of (*IP*) directly as in [9] and [16]. Consider the problem (*IP*) after dualizing constraint (4). Then, subproblem 2 of this Lagrangean relaxation problem is a shortest-path problem and the solution is to route *all* of the sessions for each O–D pair over the shortest path (whenever it is unique). In other words, the solution is a single-path routing solution. In contrast, our approach results in multiple-path solutions. Another motivation for solving a sequence of (*RP*)'s is to assure that the multipliers change gradually.

We then solve the dual problem:

$$Z_D = \max_{u \geq 0} Z_D(u) = \max_{u \geq 0} Z_D^1(u) + Z_D^2(u), \qquad (D)$$

and use the solution to subproblem 2 as a heuristic routing decision. Note that the solution to (*RPLR*) is a lower bound on $Z_{RP}$, but may not be a lower bound on $Z_{IP}$. A lower bound on $Z_{IP}$ is obtained by solving (*IPLR*). Note that subproblem 1 of (*IPLR*) is the same as subproblem 1 of (*RPLR*) and subproblem 2 of (*IPLR*) is $|W|$ shortest-path problems. Thus, the computation of a lower bound is essentially a byproduct of solving (*D*).

There are several methods for solving the dual problem (*D*). For example, we may apply the subgradient optimization procedure [9, 13] or the multiplier adjustment procedure [16]. In Section 4, the overall algorithm for solving the virtual circuit routing problem is presented.

## 4. A SOLUTION PROCEDURE

The overall routing algorithm consists of solving a sequence of restricted problems (*RP*)'s. $k_1$ is the iteration counter for the number of (*RP*)'s solved. The algorithm stops when the best heuristic solution is within a specified tolerance of an optimal solution or the number of restricted problems solved is $K_1$. For each (*RP*), (*RPLR*) is solved (step 1) and the multipliers are adjusted (step 2) $K_2$ times. $k_2$ is the iteration counter for the number of times (*RPLR*) has been solved and the number of times the multipliers have been adjusted. In step 2.a, we employed the multiplier adjustment procedure developed in [16] rather than the commonly used subgradient method. In [16], it was shown that this multiplier adjustment procedure is very effective and efficient and better suited for distributed computation. The control sequence in the multiplier adjustment procedure was chosen to be $k = (k_1 - 1)K_2 + k_2$.

*Let* $g_l^k$ denote the aggregate flow on link $l$ resulting from the solution of subproblem 2 of (*RPLR*) at the $k$th iteration. Then, an upper bound on $Z_{IP}$ is determined by

$$\sum_{l \in L} \frac{g_l^k}{C_l - g_l^k}.$$

Let $UB$ and $LB$ be the smallest upper bound and the largest lower bound obtained so far, respectively. Then, $(UB - LB)/LB$ is an upper bound on how far the best heuristic solution is from an optimal solution. Let $\varepsilon$ be a prespecified upper bound on $(UB - LB)/LB$. The overall routing algorithm is given below:

**0. Initialize**
- 0.a Find an initial primal feasible solution $\{x_p^0\}$.
- 0.b Find an initial dual feasible solution $\{u_l^0\}$.
- 0.c Set $k_1$ and $k_2$ to 1.

**1. Solve (*RPLR*)**
- 1.a $k \leftarrow (k_1 - 1)K_2 + k_2$.
- 1.b $x_p^k \leftarrow x_p^{k-1}$.
- 1.c For each link $l$, calculate $f_l^k$ by using (22).
- 1.d Solve subproblem 2 for each O–D pair.
- 1.e For each link $l$, calculate $g_l^k$.
- 1.f Update $UB$ and $LB$.

**2. Adjust the multipliers**
- 2.a $u_l^{k+1} \leftarrow C_l^{-1}\left[1 - \dfrac{kf_l^k + g_l^k}{(k+1)C_l}\right]^{-2} \quad \forall\, l \in L.$
- 2.b $k_2 \leftarrow k_2 + 1$.
- 2.c If $k_2 \le K_2$, go to step 1. Otherwise, $x_p^k \leftarrow (x_p^k + \Delta_p)$, $k_2 \leftarrow 1$, and $k_1 \leftarrow k_1 + 1$.
- 2.d If $(UB - LB)/\mathrm{LB} < \varepsilon$, stop. If $k_1 \le K_1$, go to step 1. Otherwise, stop.

In the initialization step, we need to find an initial primal feasible solution $\{x_p^0\}$ and an initial dual feasible solution $\{u_l^0\}$. A number of methods can be used to determine $\{x_p^0\}$. The K(0)-ordering heuristic [5] or the single-path routing method [9, 16] may be used. The method we used was to relax the integer constraints of (*IP*). The resulting relaxed problem is the datagram routing problem that can be solved by a number of convex programming techniques. For simplicity, we used the Frank–Wolfe method. Then, we used a round-off procedure to transform the continuous solution to the relaxed problem to find an initial primal feasible integer solution. The round-off scheme that we used is as follows: For each O–D pair, the paths are ordered according to the fractional part of the number of sessions assigned in the continuous solution. Then, the paths with larger fractional parts are rounded up and the paths with smaller fractional parts are rounded down. More precisely, the sum (an integer) of the fractional parts is computed and distributed to those paths with larger fractional parts.

To illustrate the round-off procedure, consider the following example for an O–D pair with three candidate paths. Suppose in the continuous solution that the number of sessions assigned to each of these paths is 4.8, 3.7, and 0.5, respectively. The fractional parts are 0.8, 0.7, and 0.5. Then, after applying the round-off procedure, the number of sessions assigned to these paths are 5, 4, and 0, respectively.

To find a good set of initial multipliers $\{x_l^0\}$, we apply a theorem from Lagrangean duality. This result states that if the primal problem is a convex programming problem then the optimal dual variables for the Lagrangean dual problem are precisely the Lagrangean multipliers in the Kuhn–Tucker conditions [1].

Consider a variation of (*IP*) with the decision variables $x_p = x_p^0 + \Delta_p$. Consider this problem where the integrality constraints are relaxed. The resulting problem is the datagram routing problem that we will call (*P*). We next show that the Lagrangean duals of (*P*) and (*IP*) are equivalent.

A Lagrangean relaxation of (*IP*) is

$$Z_{DIP}(u) = \min \sum_{l \in L} \frac{f_l}{C_l - f_l} + \sum_{l \in L} u_l \left\{ \sum_{w \in W} \sum_{p \in P_w} x_p \gamma \delta_{pl} - f_l \right\}, \qquad (IPLR)$$

subject to

$$0 \le f_l \le C_l \qquad \forall\, l \in L \tag{24}$$

$$\sum_{p \in P_w} x_p = n_w \qquad \forall\, w \in W \tag{25}$$

$$x_p \ge 0 \qquad \forall\, p \in P_w,\ w \in W \tag{26}$$

$$x_p = \text{integer} \qquad \forall\, p \in P_w,\ w \in W. \tag{27}$$

Let (*PLR*) be the Lagrangean relaxation of (*P*) where the flow constraints are dualized. Let $Z_{DIP}(u)$ and $Z_{DP}(u)$ be the optimal objective function values for (*IPLR*) and (*PLR*), respectively. Let (*DIP*) and (*DP*) denote the respective dual problems of maximizing $Z_{DIP}(u)$ and $Z_{DP}(u)$ over $u \ge 0$. Then,

**Lemma 1.**

(i) $Z_{DIP}(u) = Z_{DP}(u)$ for all $u \ge 0$.
(ii) (*DP*) and (*DIP*) are equivalent problems.

*Proof.* Note that the only difference between (*IPLR*) and (*PLR*) is the integer constraints on $\{x_p\}$. As in (*RPLR*), both (*IPLR*) and (*PLR*) consist of two subproblems. In (*RPLR*), (*IPLR*), and (*PLR*), the first subproblem is exactly the same. The optimal solution to the second subproblem for both (*IPLR*) and (*PLR*) is to assign $n_w$ sessions to a shortest path. This shows that $Z_{DIP}(u) = Z_{DP}(u)$ for any $u$. ■

To clarify the proof of Lemma 1, a few remarks are needed. First, there may be alternative shortest paths in the second subproblem so that the optimal assignment of sessions to paths may not be unique. However, the optimal

objective function value of subproblem 2 is unique. Second, by writing constraint (25) in the form $Ax = b$, $A$ is unimodular. This implies that every extreme point of $Ax = b$ is integer so that the integrality constraints in (*IPLR*) are redundant.

The following result is proven in Appendix I.

**Lemma 2.** A set of multipliers that satisfies the Kuhn–Tucker conditions for (*P*) is given by

$$u_l^* = \frac{C_l}{(C_l - f_l^*)^2}, \tag{28}$$

where $\{f_l^*\}$ is an optimal flow for link $l$.

Let $Z_{IP}$ and $Z_P$ denote the optimal objective function values of (*IP*) and (*P*), respectively. The following result is now almost immediate.

**Theorem 1.**

(i) $Z_{DIP} = Z_{DP} = Z_P \leq Z_{IP}$.
(ii) *The Lagrangean dual problem* (*DIP*) *is solved exactly by* (28).

*Proof.*

(i) By Lemma 1, $Z_{DIP} = Z_{DP}$. Since (*P*) is a convex programming problem, $Z_{DP} = Z_P$ by the strong Lagrangean duality theorem. By the weak Lagrangean duality theorem, $Z_{DIP} \leq Z_{IP}$.
(ii) Since (*P*) is a convex programming problem, the multipliers given in (28) are optimal dual variables for (*DP*). Since (*DP*) and (*DIP*) are equivalent problems (Lemma 1), the multipliers given in (28) are optimal dual variables for (*DIP*).

The significance of the above results is twofold. First, it was shown that the Lagrangean dual problem can be solved exactly by solving a convex programming problem (the datagram routing problem). Finding an exact solution to the Lagrangean dual problem of combinatorial optimization problems occurs rarely in the literature. Second, from Theorem 1, we would expect the lower bounds to be tight. This did indeed occur in our computational experiments where the difference between the best heuristic solution and the best lower bound was less than 1% for problems with 9,000 integer variables.

## 5. COMPUTATIONAL RESULTS AND DISCUSSION

The routing algorithm for virtual circuit networks described in Section 4 was coded in FORTRAN 77 and run on a SUN 4/260 workstation. One of the objectives in the computational experiments was to compare the quality of the solutions of the single-path and multiple-path routing assignments. To fairly compare the two solutions, the same number of iterations were used for both

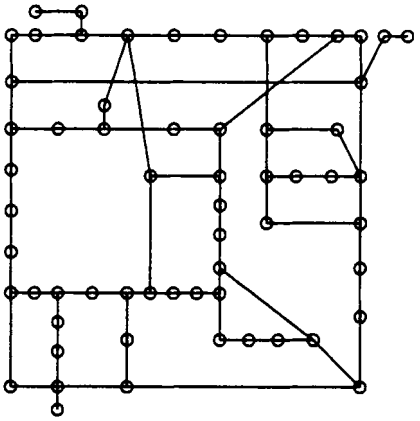

FIG. 1. 61-node 148-link ARPA net.

schemes. To do this, only the iteration counters were considered in the termination criterion ($\varepsilon$ was set to zero). In addition, in the experiments in [16], the accuracy of the single-path routing scheme (and therefore also multiple-path routing) was extremely high. The maximum number of restricted problems solved was $K_1 = 30$. For each restricted problems, the number of multiplier adjustments was $K_2 = 10$. Thus, the total number of iterations in terms of the number of times that (*RPLR*) was solved was 300. The control parameter $m$ in the multiplier adjustment procedure was chosen to be $(k_1 - 1)K_2 + k_2 + 1$.

An initial primal feasible solution was determined by (i) applying the Frank–Wolfe method for 20 iterations to the datagram routing problem and (ii) then using the round-off scheme described earlier. The method for determining an initial feasible flow is given in Appendix II. The initial set of multipliers was determined by Eq. (28) where the link flows correspond to the optimal routing assignments for the datagram routing problem. This initial set of multipliers was used as the initial feasible solution for the dual problem of the first (*RP*). In each iteration, the last set of multipliers found was used as the initial set of multipliers for the next dual problem. In the experiments, the last primal feasible solution found for the current (*RP*) was used as the initial primal solution to the next (*RP*). This could be modified by using the best primal feasible solution found instead of the last one. By best primal feasible solution, we mean the best heuristic solution of the 10 obtained by solving the 10 (*RPLR*)'s associated with the current (*RP*). However, the experiments show that this variation does not result in better solutions.

The algorithm was tested on three networks: ARPA, RING, and OCT with 61, 32, and 26 nodes, respectively. Their topologies are shown in Figures 1–3. All the link capacities were assumed to be 150 Mbps and the mean packet length was assumed to be 500 bits. For each O–D pair, the number of sessions was generated from rounding-down the value generated from a uniform distribution. The third column of Table I gives the ranges of the uniform distribu-

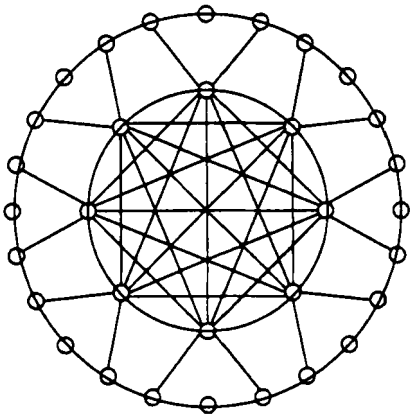

FIG. 2. 32-node 120-link RING net.

tions. The average packet arrival rate for each session is given in the fourth column. For each O–D pair, there were at most three candidate paths. Each candidate path was chosen as a shortest path with respect to a set of randomly generated arc weights.

The implementation of the multiple-path routing algorithm could be modified to consider all simple paths for each O–D pair. By replacing the session-exchange constraint (13) with (23), sessions would be shifted from expensive paths to a shortest path. In this variation, all possible simple paths are implicitly considered. This variation would likely result in better heuristic solutions. However, from the results reported in [9], the minimal average packet delay remains about the same when $|P_w|$ is three or more. In our early computational experiments, we tested an implementation where all simple paths were considered. This implementation resulted in a reduction of the minimal average delay of less than 5% while the computation time increased by 50–100% compared with the implementation where $|P_w|$ is at most 3.

Table I summarizes the results of our computational experiments on the performance of the multiple-path routing scheme. The fifth column is the larg-

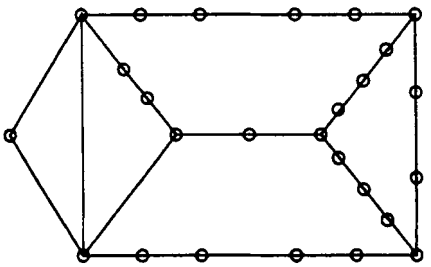

FIG. 3. 26-node 60-link OCT net.

TABLE I. Computational results of the multiple-path routing scheme.

| Case no. | Network ID | Range no. of sessions | Packets/second per session | Lower bounds ($\mu$s) | Upper bounds ($\mu$s) | Bound difference (%) | Max. link utilization | Degree of bifurcation (%) | CPU time (seconds) |
|---|---|---|---|---|---|---|---|---|---|
| 1 | ARPA | [0, 3.1] | 1000 | 65.2216 | 65.2506 | 0.045 | 0.9367 | 0.109 | 125.8 |
| 2 | " | [0, 4.7] | 600 | 94.6141 | 94.6578 | 0.046 | 0.9820 | 0.628 | 127.6 |
| 3 | " | [0, 8.5] | 300 | 106.002 | 106.007 | 0.005 | 0.9860 | 0.792 | 126.6 |
| 4 | " | [0, 18.5] | 128 | 97.7368 | 97.7387 | 0.002 | 0.9860 | 0.355 | 125.9 |
| 5 | RING | [0, 5.3] | 3000 | 31.6369 | 31.6835 | 0.147 | 0.9400 | 2.822 | 33.5 |
| 6 | " | [0, 7.8] | 2000 | 36.8814 | 36.9948 | 0.308 | 0.9467 | 4.132 | 34.7 |
| 7 | " | [0, 15] | 1000 | 38.3651 | 38.3946 | 0.077 | 0.9633 | 3.831 | 32.2 |
| 8 | " | [0, 24.5] | 600 | 38.6581 | 38.6759 | 0.046 | 0.9420 | 4.536 | 34.5 |
| 9 | " | [0, 48] | 300 | 37.5960 | 37.6035 | 0.020 | 0.9370 | 4.536 | 33.4 |
| 10 | " | [0, 113] | 128 | 40.5799 | 40.7348 | 0.382 | 0.9451 | 6.754 | 36.5 |
| 11 | OCT | [0, 4.18] | 3000 | 133.280 | 134.542 | 0.947 | 0.9800 | 3.692 | 19.2 |
| 12 | " | [0, 5.79] | 2000 | 154.766 | 155.821 | 0.682 | 0.9867 | 4.000 | 20.5 |
| 13 | " | [0, 10.7] | 1000 | 178.169 | 178.950 | 0.438 | 0.9867 | 6.769 | 19.5 |
| 14 | " | [0, 17] | 600 | 130.397 | 130.487 | 0.069 | 0.9700 | 7.538 | 20.7 |
| 15 | " | [0, 33] | 300 | 128.611 | 128.669 | 0.045 | 0.9720 | 5.692 | 19.4 |
| 16 | " | [0, 76] | 128 | 128.643 | 128.833 | 0.147 | 0.9715 | 7.846 | 20.1 |

est lower bound on the optimal objective function value found in 300 iterations. The sixth column gives the objective function value, denoted by $Z^{MPR}$, for the best primal feasible solution found in 300 iterations. The seventh column gives an upper bound [(upper-bound − lower-bound) · 100/lower-bound] on the percentage difference between the best feasible solution found and an optimal solution.

The eighth column gives the maximum link utilization factor in the final solution. The ninth column provides the percentage of session groups that are routed through multiple paths, where a session group refers to those sessions with the same O–D pair. This value is referred to as the degree of session bifurcation. The tenth column shows the CPU time, which consists of the time to compute a solution and the time used to input the problem parameters.

From an inspection of Table I, it is clear that the routing algorithm is efficient and very effective in finding near-optimal solutions. For every test problem (networks with up to 61 nodes), the routing algorithm determines a solution that is within 1% of an optimal solution in minutes of CPU time on a SUN 4/260 workstation. Also, our routing algorithm worked well for heavily loaded networks. Another observation is that the degree of session bifurcation is small for a large network (cases 1–4). Another observation that is not shown in Table I is that the initial solutions have little effect on the final solutions. The initial solutions obtained by using minimum hop paths for each session are almost as effective as those obtained by applying the Frank–Wolfe method and the round-off scheme. This property suggests a way to reduce the complexity and computation time of our algorithm. Another result that is also not reported in Table I is that if the link capacities are increased or the basic traffic requirement unit (the traffic rate of each session) is decreased the performance of the round-off heuristic improves.

A comparison of multiple-path routing, single-path routing, the round-off Frank–Wolfe heuristic, and the $K(0)$-ordering heuristic methods is reported in Table II. The third column presents the value of $Z^{SPR}$, the best average packet delay found in 300 iterations by using single-path routing. The fourth column shows the percentage difference between $Z^{SPR}$ and $Z^{MPR}$. The fifth column gives the value of $Z^{RO}$, the average packet delay obtained by using the round-off Frank–Wolfe heuristic. The sixth column shows the percentage difference between $Z^{RO}$ and $Z^{MPR}$. The seventh column gives the value of $Z^{K0}$, the average packet delay obtained by using the $K(0)$-ordering heuristic. The eighth column shows the percentage difference between $Z^{K0}$ and $Z^{MPR}$.

An observation from Table II is that the multiple-path routing scheme provides a lower average packet delay than do all the other schemes compared. For many cases, there is significant difference in the performance. For example, in case 13, the single-path routing method found a solution that is 57.18% worse, the round-off Frank–Wolfe heuristic determined a solution that is 24.53% worse, and the $K(0)$-ordering heuristic failed to obtain a feasible solution. It should be noted that in single-path routing and the $K(0)$-ordering heuristics all the sessions for each O–D pair are routed over the same path. In contrast, the solutions obtained by the Frank–Wolfe heuristic or the multiple-

TABLE II. A comparison of the multiple-path routing scheme with the single-path routing scheme, the round-off Frank–Wolfe algorithm, and the $K(0)$-ordering heuristic.

| Case no. | $Z^{MPR}$ ($\mu$s) | $Z^{SPR}$ ($\mu$s) | $\frac{(Z^{SPR} - Z^{MPR})}{Z^{MPR}}$ (%) | $Z^{RO}$ ($\mu$s) | $\frac{(Z^{RO} - Z^{MPR})}{Z^{MPR}}$ (%) | $Z^{K0}$ ($\mu$s) | $\frac{(Z^{K0} - Z^{MPR})}{Z^{MPR}}$ (%) |
|---|---|---|---|---|---|---|---|
| 1 | 65.2506 | 65.3537 | 0.158 | 65.2801 | 0.045 | 68.7421 | 5.351 |
| 2 | 94.6578 | 94.7050 | 0.050 | 94.6798 | 0.023 | 102.339 | 8.115 |
| 3 | 106.007 | 106.025 | 0.017 | 106.056 | 0.046 | 114.115 | 7.649 |
| 4 | 97.7387 | 97.7969 | 0.060 | 97.7511 | 0.013 | 105.209 | 7.643 |
| 5 | 31.6835 | 31.8830 | 0.630 | 33.1809 | 4.726 | 36.6828 | 15.78 |
| 6 | 36.9948 | 38.6193 | 4.391 | 38.1605 | 3.151 | 45.0372 | 21.74 |
| 7 | 38.3946 | ***[a] | $\infty$ | 46.0505 | 19.94 | 45.3283 | 18.06 |
| 8 | 38.6759 | ***[a] | $\infty$ | 39.9167 | 3.208 | 47.3715 | 22.48 |
| 9 | 37.6035 | 53.2143 | 41.51 | 38.0946 | 1.306 | 44.1007 | 17.28 |
| 10 | 40.7348 | 115.315 | 183.1 | 41.3041 | 1.398 | 49.7611 | 22.16 |
| 11 | 134.542 | 138.453 | 2.921 | 154.873 | 15.11 | 185.666 | 38.00 |
| 12 | 155.821 | 178.725 | 1.470 | 194.244 | 24.66 | 209.695 | 34.57 |
| 13 | 178.950 | 281.267 | 57.18 | 222.839 | 24.53 | ***[b] | $\infty$ |
| 14 | 130.487 | 150.400 | 15.26 | 134.456 | 3.042 | 145.962 | 11.86 |
| 15 | 128.669 | 152.144 | 18.24 | 132.380 | 2.884 | 140.960 | 9.552 |
| 16 | 128.833 | 129.770 | 0.727 | 133.080 | 3.297 | 159.211 | 23.58 |

[a] No feasible solution is found in 300 iterations.
[b] The solution obtained is infeasible.

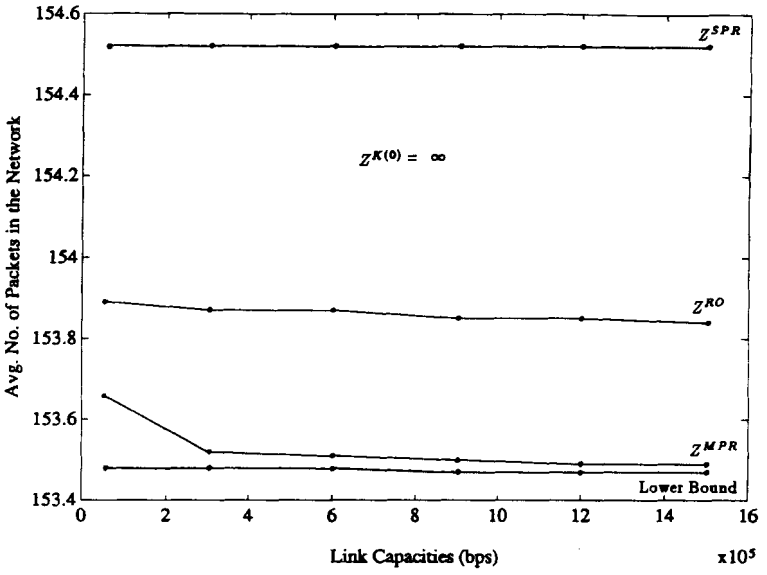


FIG. 4.

path routing method may result in the splitting of the sessions for an O–D pair over a number of different paths. Case 13 accentuates the importance of splitting sessions when there is heavy traffic. Another observation is that for a large network, e.g., the ARPANET, our routing algorithm performs only marginally better than does the single-path routing scheme and the round-off Frank–Wolfe heuristic.

In the next set of experiments, we investigated the effect of changing the link capacities on the performance of the $K(0)$-ordering heuristic [5], the rounded-off optimal continuous solution, the single-path routing scheme, and the multiple-path routing scheme. In all the test problems, the traffic rate of each session was the same. For each of the three topologies (ARPA, OCT, and RING) considered, the load on the network was kept constant. For example, in the first test problem for ARPA, all the link capacities were 50 Kbps and the number of sessions for each O–D pair was generated from some pmf. In the second test problem, the link capacities and the number of sessions for each O–D pair in the first problem were multiplied by some integer factor. In each of the subsequent test problems, the capacities and number of sessions were multiplied by other integer factors. Thus, the loading of the network (total traffic relative to the network capacity) was kept constant. The objective function used was the average number of packets in the network. Figures 4–6 provide the best objective function value found by each of the four methods for the ARPA, RING, and OCT networks, respectively. In addition, the best lower bound found for the optimal objective function value is also shown in the figures.

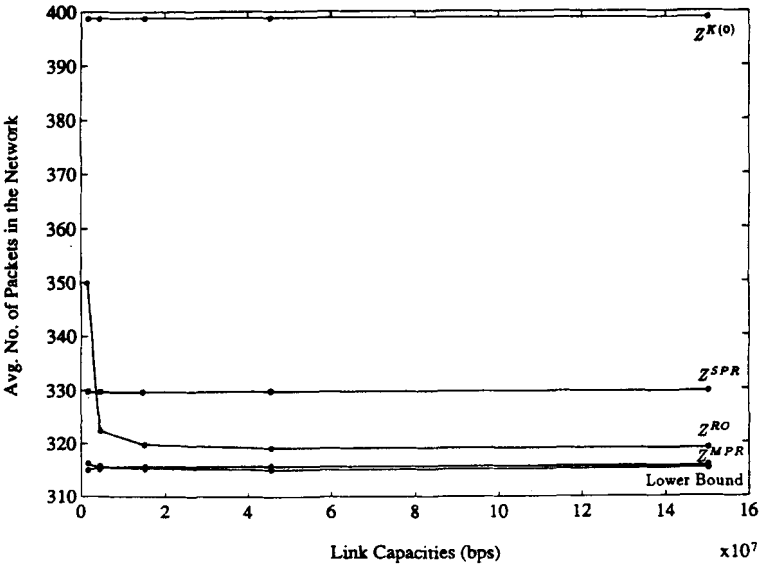


FIG. 5.

From these figures, the performance of the rounded-off optimal continuous solution and the multiple-path routing scheme improve as the link capacity increases. Another observation is that the best objective function values found for the $K(0)$-ordering and single-path routing heuristics are invariant with increases in the capacities. This observation from our computational experiments

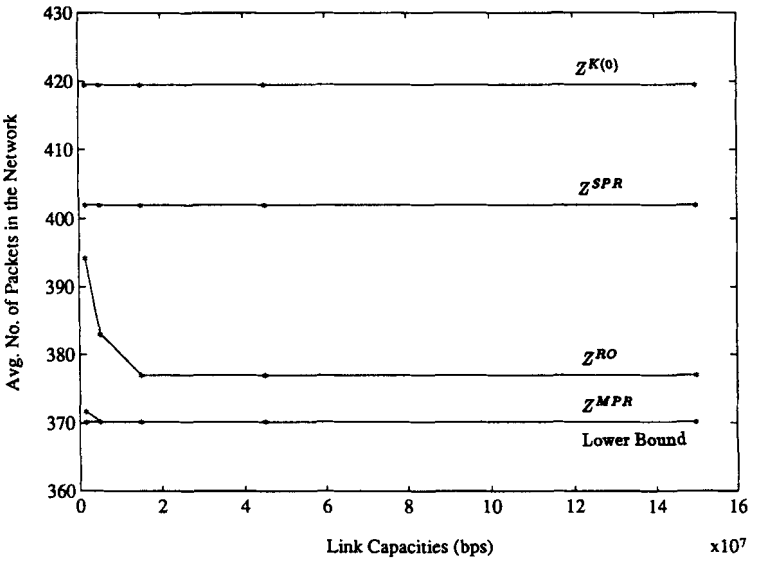


FIG. 6.

can be proven to be generally true. The proofs of the following results are given in Appendix III.

**Proposition 1.** If the number of sessions and the link capacities are changed by the same factor, then $Z^{K(0)}$ remains the same.

**Proposition 2.** If the number of sessions and the link capacities are changed by the same factor, then $Z^{SPR}$ will remain the same when the initialization procedure described in Section 4 is used. This result holds whenever the subgradient method or the multiplier adjustment procedure is used.

Other results are (i) $Z^{MPR} \leq Z^{RO}$ and (ii) $Z^{SPR} \leq Z^{K(0)}$. These relationships are true since initial feasible solutions for the multiple-path routing method and the single-path routing method are the rounded-off continuous solution and the $K(0)$-ordering heuristics, respectively. The performance of each of the four methods for the objective of minimizing the average delay can easily be obtained from Figures 4–6 by dividing each point by the total input rate. Recall that the capacities and the total input rate are multiplied by the same factor. Thus, as the capacities are increased, the decrease in the average delay is more dramatic than is the decrease in the average number of packets in the network.

## 6. SUMMARY AND CONCLUSIONS

In this article, a multiple-path routing algorithm has been proposed for virtual circuit networks with multiple sessions for each O–D pair. In contrast to the single-path routing algorithm [9, 16], the sessions of an O–D pair in a multiple-path routing scheme may be routed through different paths. The problem was formulated as a nonlinear combinatorial optimization problem. To solve this problem, we developed an iterative scheme where at each iteration we solved a restricted problem. For each of the restricted problems, the number of sessions shifted from one path to another is limited. The idea was to prevent the algorithm from finding single-path routing assignments. Another motivation for doing this is to assure that the multipliers change gradually. We then applied Lagrangean relaxation and the multiplier adjustment procedure [16] to solve each restricted problem.

In the computational experiments, our algorithm was shown to be very effective and efficient in finding near-optimal solutions. Our algorithm determines solutions that are within 1% of an optimal solution in minutes of CPU time for networks with 26–61 nodes.

Our algorithm is also compared with the round-off Frank–Wolfe method, the single-path routing scheme, and the $K(0)$-ordering heuristic. Both theoretical and computational results show that the multiple-path routing scheme is better than the others. We also found that when the network load is high or unbalanced, the single-path routing algorithm and the $K(0)$-ordering may perform far worse than the multiple-path routing algorithm. In addition, we found that when the ratio of session rate and link capacity becomes very small the round-

off Frank–Wolfe method performs almost as well as does the multiple-path routing scheme.

Since a distributed algorithm is more desirable than is a centralized one due to higher reliability, a continuation of this work would be to develop a distributed version of the multiple-path routing algorithm. Our formulation can be generalized to include priority traffic and input rate flow controls.

## APPENDIX I

In this appendix, we prove that the multipliers given by (28) satisfies the Kuhn–Tucker conditions for $(P)$. Let $\{u_l\}$, $\{v_l\}$, $\{s_w\}$, and $\{t_p\}$ be the multipliers associated with constraints (4)–(7). Note that $f_l < C_l$ in any optimal solution due to the implicit barrier function in the objective function. Thus, the right-hand inequality in (5) can be ignored.

*Proof of Lemma 2.* The Kuhn–Tucker conditions for $(P)$ are

$$\frac{C_l}{(C_l - f_l)^2} = u_l + v_l \qquad \forall\, l \in L \tag{29}$$

$$\gamma \sum_{l \in h_p} u_l = s_w + t_p \qquad \forall\, p \in P_w,\ w \in W \tag{30}$$

$$u_l \left( \sum_{w \in W} \sum_{p \in P_w} (x_p^0 + \Delta_p)\gamma\delta_{pl} - f_l \right) = 0 \qquad \forall\, l \in L \tag{31}$$

$$v_l f_l = 0 \qquad \forall\, l \in L \tag{32}$$

$$t_p(x_p^0 + \Delta_p) = 0 \qquad \forall\, p \in P_w,\ w \in W \tag{33}$$

$$u_l \geq 0 \qquad \forall\, l \in L \tag{34}$$

$$v_l \geq 0 \qquad \forall\, l \in L \tag{35}$$

$$t_p \geq 0 \qquad \forall\, p \in P_w,\ w \in W. \tag{36}$$

We now show that there exists a set of multipliers $\{u_l, v_l, s_w, t_p\}$, where $u_l = C_l/[(C_l - f_l)^2]$, that satisfies the Kuhn–Tucker conditions. In every optimal solution, $\Sigma_{w \in W} \Sigma_{p \in P_w} (x_p^0 + \Delta_p)\gamma\delta_{pl} = f_l$, $\forall\, l \in L$, so (31) is always satisfied. Consider any arbitrary path $p$. There are two cases to consider: (i) Suppose $x_p^0 + \Delta_p > 0$. Then, $f_l > 0$ for every link $l$ on path $p$. Then, the complementary slackness condition (32) implies $v_l = 0$. Then, from (29), $u_l = C_l/[(C_l - f_l)^2]$. (ii) Suppose $x_p^0 + \Delta_p = 0$. There are two subcases to consider: (a) Consider any link $l$ on path $p$ where $f_l > 0$ (due to other path flows). Then, from case (i), $u_l = C_l/[(C_l - f_l)^2]$. (b) Consider any link $l$ on path $p$ where $f_l = 0$. We now show that we can choose $v_l = 0$ and $u_l = C_l/[(C_l - f_l)^2]$ and find corresponding $\{s_w\}$ and $\{t_p\}$ that

satisfy the above conditions. To satisfy (33) and (36), $t_p$ can be any nonnegative number. Interpret $u_l$ as arc weights. Let $s_w = \min_{p \in P_w} \gamma \sum_{l \in h_p} u_l$ (the length of a shortest path). Then, $t_p$ can be chosen to be $\gamma \sum_{l \in h_p} u_l - \min_{p \in P_w} \gamma \sum_{l \in h_p} u_l$. ■

## APPENDIX II

In this appendix, the technique for finding an initial feasible solution to (*IP*) where the integer constraints are relaxed is summarized. This method is given in [4]. The idea is to relax the capacity constraints and to replace the original objective function $D(g)$ with an approximate function $\tilde{D}(g)$. The approximate function is convex and continuously differentiable. It is defined for all nonnegative flows and has a high penalty whenever any capacity constraint is violated.

The approximate function used is

$$\tilde{D}(g) = \sum_{l \in L} \tilde{D}_l(g_l),$$

where

$$\tilde{D}_l(g_l) = \begin{cases} \dfrac{g_l}{C_l - g_l} & \text{if } g_l \leq (1 - \alpha)C_l, \\ \dfrac{1 - \alpha}{\alpha} + \dfrac{g_l - (1 - \alpha)C_l}{\alpha^2 C_l} & \text{otherwise} \end{cases}$$

and $(0 < \alpha \ll 1)$. In Figure 7, the relationship between the original function and the approximate function is illustrated.

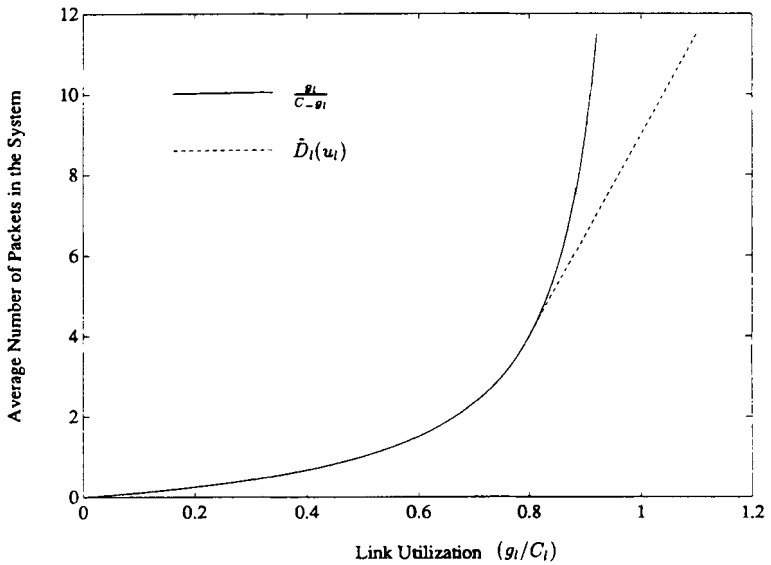


FIG. 7. The approximate function.

## APPENDIX III

In this appendix, we prove Propositions 1 and 2. Let $I$ denote the original problem instance with capacities $\{C_l\}$ and number of sessions $\{n_w\}$. Let $I_\alpha$ denote the problem instance where each $C_l$ and $n_w$ are multiplied by $\alpha$. Let $Z^{K(0),\alpha}$ be the objective function value found by the $K(0)$-ordering heuristic for $I_\alpha$. Before reading the proof of Proposition 1, the reader may want to review the $K(0)$-ordering heuristic in [5].

*Proof of Proposition 1.* The proof is by induction. We first show that the objective function values in the initial solutions of $I$ and $I_\alpha$ are the same. The initial solution is determined by assigning the traffic for each O–D pair to a shortest path where the arc weights are $\{C_l^{-1}\}$ for $I$ and $\{(\alpha C_l)^{-1}\}$ for $I_\alpha$. Since the arcs weights for the two problem instances differ by the same scale factor, the shortest path chosen for each O–D pair in both problems are identical. Here, we assume that the same shortest path algorithm is used on both problems so that ties are broken in the same way. Consequently, the aggregate flows in the two problems will differ by the same scale factor, i.e., $g_l^1(\alpha) = \alpha g_l^1$. Then, $\Sigma_l (\alpha g_l^1)/[(\alpha C_l - \alpha g_l^1)] = \Sigma_l (g_l^1)/[(C_l - g_l^1)]$, showing that the objective function values are the same in the two problems. In addition, the ordering of the O–D pairs is the same for each problem instance.

Assume that at iteration $k$, $g_l^k(\alpha) = \alpha g_l^k$. Thus, $\Sigma_l (\alpha g_l^k)/[(\alpha C_l - \alpha g_l^k)] = \Sigma_l (g_l^k)/[(C_l - g_l^k)]$. Then, at iteration $k + 1$, the sets of arc weights for the two problems are $\{(\alpha C_l)/(\alpha C_l - \alpha g_l^k)^2\}$ and $\{C_l/(C_l - g_l^k)^2\}$. Since the corresponding arc weights in the two problems differ by the same factor, the assignment of routes for each O–D in both problems will be the same in iteration $(k + 1)$. Consequently, $g_l^{k+1}(\alpha) = \alpha g_l^{k+1}$ and $\Sigma_l(\alpha g_l^{k+1})/(\alpha C_l - \alpha g_l^{k+1}) = \Sigma_l (g_l^{k+1})/(C_l - g_l^{k+1})$. ■

Let (*IPSP*) be the single-path routing problem. Let $Z_{DIPSP}(u)$ and $Z_{DIPSP,\alpha}(v)$ denote the optimal objective function value of the Lagrangean relaxation of (*IP*) and the objective function of (*DIPSP*) for problems $I$ and $I_\alpha$, respectively. Note that the vector of multipliers for $I_\alpha$ is $v$.

*Proof of Proposition 2.* We prove by induction that $v^k = u^k/\alpha$ for all $k$. It then follows that $g_l^k(\alpha) = \alpha g_l^k$, resulting in the same objective function value for both problem instances. We first show the basis of the induction. The initial set of aggregate flows for the two problem instance, $g_l^0$ and $g_l^0(\alpha)$, are determined by the $K(0)$-ordering heuristic. Then, $g_l^0(\alpha) = \alpha g_l^0$. From Eq. (28), $v_l^0 = u_l^0/\alpha$.

Assume that at iteration $k$, $v^j = u^j/\alpha$ for $j = 1, \ldots, k$. Then, the estimate of the flows for $I_\alpha$ at iteration $k + 1$ is given by

$$f_l^k(\alpha) = \alpha C_l \left(1 - \sqrt{\frac{1}{\frac{u_l^k}{\alpha} \alpha C_l}}\right) = \alpha f_l^k, \tag{37}$$

where $f_l^k$ represents the estimate flow on link $l$. In the algorithm, the multipliers

may be updated by (i) the multiplier adjustment procedure or (ii) the subgradient method. These two cases are considered.

**Case (i)—Multiplier Adjustment Procedure**

From Eq. (37) and step 2.a in the algorithm of Section 4,

$$v_l^{k+1} = \frac{1}{\alpha C_l\left(1 - \dfrac{(m_k - 1)\alpha f_l^k + \alpha g_l^k}{m_k \alpha C_l}\right)^2} = \frac{u_l^{k+1}}{\alpha}. \tag{38}$$

**Case (ii)–The Subgradient Method**

The update rule in the subgradient method is

$$v_l^{k+1} = v_l^k + t^k(\alpha) y_l^k(\alpha), \tag{39}$$

where $y^k(\alpha)$ is a subgradient of $Z_{DIPSP,\alpha}(v)$ at iteration $k$. The step size $t^k(\alpha)$ is determined by

$$t^k(\alpha) = \delta \frac{Z^{h,SPR} - Z_{DIPSP,\alpha}(v^k)}{\|y^k(\alpha)\|^2}, \tag{40}$$

where $Z^{h,SPR}$ is the best objective function value found in the first $k$ iterations and $\delta$ is a constant, $0 < \delta \leq 2$. Since $v^j = u^j/\alpha$ for iterations $j = 1, \ldots, k$, $g_l^k(\alpha) = \alpha g_l^k$ and $Z^{h,SPR}$ are the same for both problem instances.

When $v^k = u^k/\alpha$, $f_l^k(\alpha) = \alpha f_l^k$ and $g_l^k(\alpha) = \alpha g_l^k$, so that $Z_{DIPSP,\alpha}(v^k) = Z_{DIPSP}(u^k)$. From (39) and (40),

$$\begin{aligned} v_l^{k+1} &= v_1^k + \delta \frac{Z^{h,SPR} - Z_{DIPSP,\alpha}(v^k)}{\|y^k(\alpha)\|^2}(\alpha g_l^k - \alpha f_l^k) \\ &= \frac{u_l^k}{\alpha} + \delta \frac{Z^{h,SPR} - Z_{DIPSP}(u^k)}{\alpha^2 \|y^k\|^2} \alpha(g_l^k - f_l^k) \\ &= \frac{u_l^k}{\alpha} + \delta \frac{Z^{h,SPR} - Z_{DIPSP}(u^k)}{\alpha \|y^k\|^2}(g_l^k - f_l^k) \\ &= \frac{u_l^{k+1}}{\alpha}. \end{aligned}$$

■

## REFERENCES


[1] M. S. Bazaraa and C. M. Shetty, *Nonlinear Programming: Theory and Algorithm*. Wiley, New York (1979).

[2] D. P. Bertsekas, A class of optimal routing algorithms for communications networks. *Proceedings of the 1980 International Conference on Circuits and Computers*, Atlanta, GA (November 1980).

[3] D. P. Bertsekas, E. M. Gafni, and R. G. Gallager. Second derivative algorithms for minimum delay distributed routing in networks. *IEEE Trans. Commun.* **COM-32(8)** August (1984) 911–919.
[4] D. G. Cantor and M. Gerla, Optimal routing in a packet switched computer network. *IEEE Trans. Comput.* **C-23** (1974) 1062–1069.
[5] P. J. Courtois and P. Semal, An algorithm for the optimization of nonbifurcated flows in computer communication networks. *Performance Evaluation* **1** (1981) 139–152.
[6] A. Danet, R. Despres, A. L. Rest, G. Pichon, and S. Ritzenthaler, The French public packet switching service: The TRANSPAC network. *Proceeding of the Third International Computer Communication Conference* (1976) 251–260.
[7] L. Fratta, M. Gerla, and L. Kleinrock, The flow deviation method: An approach to store-and-forward communication network design. *Networks* **3** (1973) 97–133.
[8] R. G. Gallager, A minimum delay routing algorithm using distributed computation. *IEEE Trans. Commun.* **COM-25(1)** (1977) 73–85.
[9] B. Gavish and S. L. Hantler, An algorithm for optimal route selection in SNA networks. *IEEE Trans. Commun.* **COM-31(10)** (1983) 1154–1160.
[10] M. Gerla, Routing and flow control in ISDN's. *Proceedings of the 1986 ICCC* (1986) 643–647.
[11] J. P. Gray and T. B. McNeill, SNA multiple-system networking. *IBM Syst. J.* **18** (1979) 263–297.
[12] *Functional Description of GTE Telenet Packet Switching Networks*. GTE Telenet Communications Corp., Vienna, VA. (1982).
[13] M. Held, P. Wolfe, and H. D. Crowder. Validation of subgradient optimization. *Math. Program.* **6** (1974) 62–88.
[14] V. L. Hoberecht, SNA function management. *IEEE Trans. Commun.* **COM-28** (1980) 594–603.
[15] L. Kleinrock, *Queueing Systems*, Vols. 1 and 2. Wiley-Interscience, New York (1975, 1976).
[16] Y. S. Lin and J. R. Yee, A distributed routing algorithm for virtual circuit data networks. *Proceedings 1989 INFOCOM* (1989) 200–207.
[17] A. Rajaraman, Routing in TYMNET. *Proceedings of the European Computer Conference* (1978).
[18] A. Segall, Optimal routing for virtual line-switched data networks. *IEEE Trans. Commun.* **COM-27** (1979) 201–209.
[19] W. K. Tsai, Convergence of gradient projection routing methods in an asychronous stochastic quasi-static virtual circuit network. *IEEE Trans. Automatic Control* **AC-34(1)** (1989) 20–33.
[20] L. R. Tymes. Routing and flow control in TYMNET. *IEEE Trans. Commun.* **COM-29** (1981) 392–398.
[21] J. Y. Yee, *Distributed Routing and Flow Control Algorithms for Communication Networks*. PhD Thesis, Massachusetts Institute of Technology (December 1985).

Received December 1989
Accepted August 1991