A Real-time Distributed Routing and Admission Control Algorithm for ATM Networks

Frank Y.S. Lin
Bellcore
3 Corporate Place, PYA2J318
Piscataway, NJ 08854
lin1@cc.bellcore.com

James R. Yee
Department of Electrical Engineering
University of Hawaii at Manoa
Honolulu, HI 96822
jyee@wiliki.eng.hawaii.edu

Abstract

In this paper, we consider the problem of determining admission controls and a path for each admitted user pair (session) to *satisfice* user quality of service (QOS) requirements (required throughput, tolerable average cell delay and tolerable cell loss probability). This problem is formulated as a nonlinear combinatorial optimization problem. The objective is the maximization of the total reward for the admitted sessions where the reward for each session reflects its priority. The constraints require that the QOS requirements be satisfied for each admitted session.

The emphasis of this work is to develop a real-time distributed algorithm to determine path assignments and admission controls. In computational experiments, we compared the proposed distributed algorithm with the minimum hop algorithm on test networks with up to 61 nodes and 10,000 macro sessions. For the test networks, the proposed algorithm achieved a 36% (on the average) improvement in the total reward over a minimum hop routing algorithm and heuristic admission control scheme in less than 1.5 seconds of CPU time using distributed computation.

1 Introduction

Broadband ISDN (BISDN) will carry multimedia traffic which includes voice, video and data. Due to its effective and flexible utilization of network resources, Asynchronous Transfer Mode (ATM) has been adopted by the CCITT as the switching technique for future high-speed networks. An area of intense research interest today is the development of congestion control methods to ensure quality of service (QOS) requirements (throughput, delay and cell loss) for each traffic type.

Traditional congestion control mechanisms are reactive. A reactive congestion control scheme responds to the reporting of congestion within the network and then makes decisions to alleviate the congestion. For high-speed networks, these feedback type, reactive control algorithms are not suitable. The typical argument is that by the time congestion is detected by a node and reported to upstream nodes, the upstream nodes would have already transmitted a very large number of cells. For this reason, there has been increasing research interest in preventive congestion control. Preventive congestion control is divided into two levels – admission control and flow enforcement (policing). Admission control is the acceptance or blocking of calls or sessions. At the cell level, flow enforcement mechanisms police a source to ensure that its transmission rate does not exceed the negiotiated limit.

In this paper, we focus on call level admission control. In particular, we consider the problem of determining admission controls and a path for each admitted user pair (session) to *satisfice* each user's QOS requirements (required throughput, tolerable average cell delay and tolerable cell loss probability). The emphasis of this work is to develop a real-time distributed algorithm to determine path assignments and admission controls. The algorithm is source-based so that routing computations are avoided at intermediate nodes. Another design philosophy behind the construction of the algorithm is that the speed of the decisions is more important than how close the solution is to an optimal solution.

The remainder of this paper is organized as follows. In section 2, the problem is formulated as a nonlinear combinatorial optimization problem. The objective is the maximization of the total reward for the admitted sessions where the reward for each session reflects its priority. The constraints require that the QOS requirements be satisfied for each admitted session. The feasible region (with the integrality constraints relaxed) is nonconvex. In section 3, we describe an approach to circumventing the nonconvexity. In section 4, we construct a restricted problem where the feasible region is smaller and convex. The construction is done by employing a simple cutting method. In section 5, we present an algorithm to find a feasible solution to the restricted problem (and to the original problem). In order to evaluate the quality of the heuristic solutions, a procedure is developed in section 6 to obtain true upper bounds on the optimal objective function value of the original problem. This procedure is based upon two types of relaxations. In section 7, a distributed implementation of the solution procedure is presented. In section 8, we report computational experience.

2 Problem Formulation

An ATM network is modeled as a graph where the ATM switches are represented by the node set $V = \{1, 2, ..., N\}$ and the optical trunks are represented by the set of arcs L. Each ATM switch consists of an interconnection network (switch fabric) and a set of buffers. Several ATM switch architectures (input queueing, output queueing and shared buffering) have been proposed [11, 22]. In [10], it was shown that output queueing results in a better throughput-delay tradeoff than input queueing and input smoothing. We will

[1]Supported in part by NSF grant NCR-9016348.

assume output queueing.

An ATM switch with output queueing is illustrated in Figure 1. In this type of architecture, there is a queue associated with each output port. Let C_l be the capacity of optical trunk $l \in L$. According to CCITT recommendation I.121 [12], each ATM cell is 53 bytes and is transmitted at a speed of 150 Mbps. The trunk capacity C_l is divided into k_l channels each with a capacity 150 Mbps. An ATM switch is a slotted system where each time slot is the transmission time of an ATM cell. In Figure 1, whenever multiple ATM cells arriving at an ATM switch (from different input ports) in the same time slot are addressed to the same output port, they contend in the interconnection network. If the number of such ATM cells exceeds c (a parameter of the interconnection network), some of them will be dropped. Cells addressed to the same output port and are not dropped in the interconnection network will be stored in a buffer of size b (a parameter of the ATM switch). When the buffer is full, some incoming cells will be dropped. For further details on the architecture and an analysis of the delay and loss probabilities, the reader is referred to [22] and [10], respectively.

Let W be the set of user pairs (sessions) in the network. The average arrival rate of new traffic for user pair w is γ_w (cells/sec). As in [10], we assume that the cell arrivals to each input port can be modeled as a Bernoulli process. It is also assumed that the Bernoulli (arrival) processes at the input ports of an ATM switch are independent. Let P_w be a given set of simple directed paths from the origin to the destination node of user pair w. We assume virtual circuit service so that the traffic for each user pair w is transmitted over exactly one path in the set P_w. Let $P = \bigcup_{w \in W} P_w$. For each user pair $w \in W$, an artificial path p'_w is introduced to carry the rejected sessions. A session is rejected whenever (i) its QOS requirements can not be satisfied or (ii) its admittance into the network causes the QOS of other sessions (of possibly higher priority) to become unacceptable. Let $P'_w = P_w \bigcup \{p'_w\}$.

For each user pair $w \in W$, let x_{wp} be 1 if the traffic of user pair w is assigned to path $p \in P'_w$ and 0 otherwise. For each path $p \in P$ and link $l \in L$, let δ_{pl} be 1 if link l is on path p and 0 otherwise. For each link l, let $\mathcal{L}_l$ be the probability of a loss of a cell (i) in the output buffer, (ii) in the switch fabric or (iii) due to a transmission error. The cell loss probability $\mathcal{L}_l$ is a function of the aggregate flow of cells over link l. For real-time traffic, such as voice and video, a small percentage of cell losses is acceptable. Since long delays are unacceptable for real-time traffic, lost or damaged video and voice cells are not retransmitted. In contrast, data traffic is less time-sensitive and its transmission is usually required to be 100% accurate. Consequently, the flow of traffic on a path for a data user pair will include flow due to retransmissions. Let R_{wp} be the probability that a *data* cell is retransmitted when path p is chosen for user pair w. Assume that the number of transmissions of a cell is geometrically distributed. Then the rate that the sending node of user pair w transmits data traffic (including retransmissions) is $\gamma_w/(1 - R_{wp})$. Due to cell losses along path p, the rate of flow of data traffic on link l attributed to user pair w is $[\gamma_w/(1 - R_{wp})]x_{wp}\delta_{lp}\prod_{j \in h_p \setminus h^l_p}(1 - \mathcal{L}_l)$ where h_p is the set of the links on path p and h^l_p is the set containing link l and its downstream links on path p.

For user pairs that use path p, let L_p and $\mathcal{D}_p$ be the end-to-end cell loss probability and delay, respectively. Let M be a large constant which is larger than the largest end-to-end delay in the network. Let σ_w be 1 if the traffic for user pair w is data and 0 otherwise. For user pair w, let d_w and l_w be the acceptable end-to-end cell delay and the acceptable end-to-end cell loss probability *per transmission*, respectively. Note that if user w is transmitting data, the acceptable end-to-end cell loss probability is 0. However, $l_w = 1$ since lost data cells and data cells received in error at the destination are retransmitted. Alternatively, one can simply ignore these cell loss constraints for data traffic. Let c_w be the reward for servicing user pair w. This value reflects the priority class of user pair w.

The problem of determining path assignments and admission controls is formulated as the following nonlinear combinatorial optimization problem.

$$Z_{IP} = \max \sum_{w \in W} c_w \sum_{p \in P_w} x_{wp} \qquad (IP)$$

subject to

$$x_{wp} = 0 \text{ or } 1 \quad \forall p \in P'_w, w \in W \qquad (1)$$

$$\sum_{p \in P'_w} x_{wp} = 1 \quad \forall w \in W \qquad (2)$$

$$\sum_{w \in W} \sum_{p \in P_w} \frac{\gamma_w x_{wp} \delta_{lp}}{1 - R_{wp}\sigma_w} \prod_{j \in h_p \setminus h^l_p} (1 - \mathcal{L}_j) = f_l \quad \forall l \in L \qquad (3)$$

$$\mathcal{D}_p \le M - (M - d_w)x_{wp} \quad \forall p \in P_w, w \in W \qquad (4)$$

$$L_p \le 1 - (1 - l_w)x_{wp} \quad \forall p \in P_w, w \in W. \qquad (5)$$

Constraints (1) and (2) require that a session be either admitted or rejected. If a session is admitted, all of the traffic of the session is transmitted over exactly one path (virtual circuit service). Constraint (3) relates f_l, the aggregate link flow of cells on link l, to the path assignment variables and the cell loss probabilities. Constraint (4) requires that the delay requirement of user pair w be satisfied. The left hand side is the end-to-end delay. If path p is used for session w ($x_{wp} = 1$), then the right hand side becomes d_w. Otherwise ($x_{wp} = 0$), the right hand side becomes M, a very large number. This makes the constraint inactive for those paths which are not used. Constraint (5) requires that the cell loss probability requirement of each user pair be satisfied. This constraint has a similar structure to (4). Note that for a data user pair w, l_w is set to 1 so that the constraint is inactive. The objective function is the total reward from servicing users.

The problem (IP) is a very difficult problem. First, the left hand sides of (3), (4) and (5) are rather complicated nonlinear functions. The cell loss probabilities ($\mathcal{L}_j$ and L_p), the retransmission probabilities (R_{wp}) and the delays (D_p) are nonlinear functions of the aggregate link flows. These functions are such that the feasible region where the integrality constraints are relaxed is not convex [16]. Second, closed-form expressions for these quantities are not known at this time. For a given set of parameters, numerical values for these quantities are determined from steady state probabilities of a discrete time Markov chain. In the next section, we discuss a solution approach which circumvents these difficulties.

3 Solution Approach

In the design of an algorithm to determine path assignments and admission controls, we placed importance on several

characteristics. First, the algorithm should be distributed for the purposes of reliability. Second, the routing decisions for a session should be made at its source. In high speed networks, routing computations should be avoided at intermediate nodes. Third, emphasis is placed on the development of an algorithm to find a feasible solution in real-time.

Let Ω_{IP} be the feasible region of (IP) with the integrality constraints relaxed. As pointed out earlier, Ω_{IP} is not a convex set. Our approach is illustrated in Figure 2. Within the nonconvex set Ω_{IP}, we inscribe the convex set Ω_{RP}. Then any integer point in Ω_{RP} is a feasible solution to (IP). In the next section, we discuss how (IP) may be modified to become a problem which we call the restricted problem (RP). The feasible region of (RP) with the integrality constraints relaxed is Ω_{RP}. In section 5, we present a distributed, source based algorithm that quickly finds an integer point in Ω_{RP}. In the computational experiments which are reported in section 8, the difference between Ω_{IP} and Ω_{RP} is very small.

A true upper bound on the optimal objective function value of (IP) (rather than (RP)) is desired to evaluate the quality of the heuristic solutions obtained. Our approach is illustrated in Figure 3. Outside the nonconvex set Ω_{IP}, we circumscribe the convex set Ω_{UBP}. Then any feasible solution to (IP) is an integer point in Ω_{UBP}. In section 6, we discuss how (IP) may be modified to become a problem which we call the upper-bounding problem (UBP). The feasible region of (UBP) with the integrality constraints relaxed is Ω_{UBP}. We apply Lagrangean Relaxation to find an upper bound on the optimal objective function value of (UBP). This bound is a true upper bound on the optimal objective function value of (IP). In the computational experiments, the difference between Ω_{IP} and Ω_{UBP} is also very small.

We next describe a quasi-static (or quasi-dynamic) implementation of an algorithm that finds a feasible solution to (IP). The time horizon is divided into intervals. The length of each interval corresponds to one-half of the upper limit on the set-up time (a given design parameter) for a call. The set-up time includes the time for the algorithm to compute path assignments and admission controls. The upper limit on the computation time used in our computational experiments was $0.05 \leq \bar{t} \leq 1$ second. In Figure 4, a typical sequence of events is illustrated. The time axis is divided into intervals. Path assignments and admission control decisions are made at times $t_0, t_1, t_2, \ldots$ (marked by "×"). In the interval $[t_0, t_1)$, sessions c, d, e and f request a call set-up and sessions a and b are released. In the interval $[t_1, t_2)$, the algorithm computes which of the four sessions are admitted and determines a path assignment for each of the admitted sessions. The decision is implemented at time t_2. The call setup delay for these 4 calls is upper bounded by $t_2 - t_0$ plus the delay of the route set-up packet, which is about $\bar{t}$ *msec* when the delay of the route set-up packet is small.

4 Restricted Problem

In this section, we consider the problem (RP) which is a more tightly constrained version of (IP). As illustrated in Figure 2, we construct the feasible region Ω_{RP} which is a convex subset of the nonconvex set Ω_{IP}. As a preliminary to the construction we develop expressions for the exact aggregate flow of data cells on each link and the retransmission probabilities for each path. We then show that by replacing the retransmission probabilities in (IP) by an upper bound the resulting constraints correspond to a convex set.

In high speed networks, lost or damaged video and voice cells are not retransmitted. A data cell may be retransmitted if (i) there is an error in transmission, (ii) the cell is lost in the switch fabric or output buffer at an intermediate switch or (iii) a long delay causes a timeout. Since the end to end cell delay will be very short, we will ignore the third possibility. Recall that R_{wp} is the probability that a *data* cell is retransmitted when path p is chosen for user pair w. Due to cell losses, R_{wp} is dependent on the cell loss probability $L_l(f_l)$ for each link l on path p. Therefore, R_{wp} is a rather complicated nonlinear function of the aggregate flow f_l for each link l on path p. We will show that by using a simple upper bound on R_{wp} results in a much simpler version of (IP).

The exact aggregate flow of data on each link l is dependent on the particular cell error-recovery scheme used. For optical networks, end-to-end error control schemes have been shown to be more effective than link-by-link error control schemes [2]. Consequently, we will only consider end-to-end error control. However, the best retransmission strategy is unclear at this time. In [3], ARQ with go-back-n retransmission (default mode) with the option of selective retransmission is recommended. In [1], selective-repeat with a block acknowledgment scheme is suggested. In the formulation below, we will assume go-back-n. This formulation can be easily modified for selective repeat.

In ATM networks, data cells may be discarded at intermediate nodes due to transmission errors. The header of an ATM cell includes a HEC byte which is used for detecting errors in the header [12]. Whenever an error is detected in the header, an intermediate node discards the cell. [An option is to use the HEC byte to correct single-bit errors. We will ignore this option so that our analysis is more conservative.] We assume that only the destination node checks for errors in the data portion of a cell. Thus, a cell with an error in its data portion is discarded at the destination. Any cell which is discarded due to errors causes a timeout. Let q_l^d and q_l^h be the probabilities that there is an error in the data or header portions of the cell in its transmission over link l.

Let $L_l(f_l)$ be the cell loss probability associated with link l (lost in the concentrator or due to buffer overflow) where f_l is the aggregate flow on link l. Then the total cell loss probability associated with link l, $\mathcal{L}_l$ is equal to $L_l + q_l^h - L_l q_l^h$. It is assumed that the loss of successive data cells routed over link l are independent and that the probability of each event occurs with probability $L_l(f_l)$. Although this assumption is not valid, it was shown by simulation in [2] to be a reasonable approximate model. It is also assumed that $L_l(f_l)$ is convex and monotone increasing for f_l in the range of interest. In Figure 5, the loss probability is plotted as a function of the offered traffic. It is assumed that the cell losses on different links are independent. When the number of inputs to each queue is large, the dependency of cell losses is weak. With the above independence assumptions, the end-to-end cell loss probability for path p is

$$L_p = \sum_{l \in h_p} \{ \prod_{j \in h_p \setminus h_p^l} (1 - L_j - q_j^h + L_j q_j^h)(L_l + q_l^h - L_l q_l^h) \}$$

$$+ \prod_{l \in h_p} (1 - L_l - q_l^h + L_l q_l^h)(1 - \prod_{l \in h_p} (1 - q_l^d)). \quad (6)$$

We next determine $\bar{R}_{wp}$, an upper bound on R_{wp}. Let $\bar{L}_p$ be an upper bound on L_p. Let B_p be the probability that the ACK for a cell transmitted over path p is lost and let $\bar{B}_p$ be an upper bound on B_p. Suppose a tagged cell is transmitted at time T_{wp}. The tagged cell is retransmitted if any of the cells transmitted in the interval $[0, T_{wp})$ or any of their corresponding ACKs are lost or discarded. In other words, the tagged cell is not retransmitted if all of the cells transmitted in the interval $[0, T_{wp})$ and all of the corresponding ACKs are correctly received. Then, the probability that the tagged cell must be retransmitted is $R_{wp} = 1 - (1 - L_p - B_p + L_p B_p)^{K+1}$ where K is a random variable which represents the number of cells transmitted in the interval $[0, T_{wp})$. Let E equal the timeout period divided by the cell transmission time. Since $K \leq E$,

$$R_{wp} \leq \bar{R}_{wp} = 1 - (1 - \bar{L}_p - \bar{B}_p + \bar{L}_p \bar{B}_p)^{E+1}. \quad (7)$$

In the model, we have assumed that data and video cells have the same priority at the ATM switches. Consequently, the cell loss probability for video and data at each switch is the same. For video traffic, the end-to-end cell loss probability is required to be at most 10^{-8} [21]. Thus, $L_p \leq 10^{-8}$ and $B_p \leq 10^{-8}$. We next provide a conservative (high) estimate of the end-to-end delay for a path of length 6000 km consisting of 100 ATM switches. Suppose the buffer of each output queue at each switch is $b = 100$ cells. By assuming each buffer contains 99 cells, the sum of the propagation delay (assuming a propagation speed of 2×10^5 km/sec) and the queueing delay at the 100 intermediate ATM switches is less than 58.3 *msec*. Then $T_{wp} \leq 116.6$ *msec*. Therefore, $R_{wp} \leq 4.125 \times 10^{-4}$. Note by assuming that each of the buffers is full, we avoid the need to explicitly consider the random nature of the queue length at each ATM switch. This is valid since we are seeking to find an overestimate of the retransmission probability.

We next provide a high estimate of the average end-to-end delay for a data cell. Let $D_l(f_l)$ be the average cell delay (queueing and transmission) on link l. For the range of operation, we assume that $D_l(f_l)$ is convex and monotone increasing. Figure 6 shows the convexity of the average cell delay (for carried traffic only) for the ATM switch shown in Figure 1 where $c = 10$ and $b = 20, 40, 80, 200$. Let a_l be the propagation delay on link l. The end-to-end cell delay for a successful transmission over path p is $D_p = \sum_{l \in h_p} (D_l + a_l)$. Let E_{wp}^{GBN} be the average end-to-end cell delay (including retransmissions) for user pair w when using path p. Then,

$$\mathcal{D}_p = E_{wp}^{GBN} = T_{wp} \frac{R_{wp}}{1 - R_{wp}} + D_p. \quad (8)$$

We use the upper bounds on R_{wp} and $\mathcal{D}_p$ to formulate a problem which we call the restricted problem (RP). With the integrality constraints relaxed, the feasible region of (RP) is a convex subset of the feasible region of (IP).

$$Z_{RP} = \max \sum_{w \in W} c_w \sum_{p \in P_w} x_{wp} \quad (RP)$$

subject to (1), (2) and

$$\sum_{w \in W} \sum_{p \in P_w} \frac{\gamma_w x_{wp} \delta_{lp}}{1 - \bar{R}_{wp} \sigma_w} \leq f_l \quad \forall l \in L \quad (9)$$

$$0 \leq f_l \leq \alpha_l C_l \quad \forall l \in L \quad (10)$$

$$\sum_{l \in L} \delta_{lp} (D_l(f_l) + a_l) + \frac{T_{wp} \bar{R}_{wp} \sigma_w}{1 - \bar{R}_{wp} \sigma_w} \leq M - (M - d_w) x_{wp} \quad \forall p \in P_w, w \in W \quad (11)$$

$$\sum_{l \in L} \delta_{lp} (L_l(f_l) + q_l^h - L_l(f_l) q_l^h + q_l^d) \leq 1 - (1 - l_w) x_{wp} \quad \forall p \in P_w, w \in W. \quad (12)$$

The LHS of (9) is the LHS of (3) where (i) R_{wp} is replaced by $\bar{R}_{wp}$ and (i) cell losses at intermediate switches are ignored. Consequently, the LHS of (9) is an overestimate of the aggregate flow on link l. Constraint (9) should be stated as an equality and thus (9) is a relaxation. Using arguments similar to the ones in [5, 18], it can be shown that in an optimal solution to (RP), equality will hold. In order for the network to provide a satisfactory QOS to every user pair, network saturation (congestion) should be avoided and the aggregate flow on link l would be limited above by $\alpha_l C_l$ where $\alpha_l > 0$ and is usually less than 1 (constraint (10). For example, in an ATM network where video traffic (one of the anticipated major services [15]) is carried, L_p and thus L_l for each link on a path p which is used must be limited above by 10^{-8} [21]. Using These restrictions on the link flows impose upper bounds on L_l and D_l since both functions are monotone increasing. Let $\bar{L}_l$ and $\bar{D}_l$ be the upper bounds, respectively. If user pair w sends data traffic, then $\sigma_w = 1$ and the LHS of (11) reduces to (8) where R_{wp} is replaced by $\bar{R}_{wp}$. Thus, the LHS of (11) overestimates the end-to-end delay for data traffic. For video and voice traffic, $\sigma_w = 0$ and the LHS of (11) becomes the end-to-end delay on path p where there are no retransmissions. The LHS of (12) is an upper bound on the end-to-end cell lost probability given in (6). Since the left hand sides of (11) and (12) are convex, the feasible solutions satisfying (10)-(12) form a convex set. The property of (RP) facilitates the development of the algorithm given in the next section.

5 A Solution Procedure

In this section, we propose an algorithm to solve (RP). The basic approach is Lagrangean Relaxation. Lagrangean relaxation is a method for obtaining good primal solutions as well as upper bounds (for maximization problems) in integer programming problems. 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 dualizing the complicating constraints.

We dualize constraints (9), (11), and (12) to obtain the following relaxation

$$Z_D(u, v, s) =$$

$$\max \sum_{w \in W} c_w \sum_{p \in P_w} x_{wp} + \sum_{l \in L} u_l [f_l - \sum_{w \in W} \sum_{p \in P_w} \frac{\gamma_w x_{wp} \delta_{lp}}{1 - \bar{R}_{wp} \sigma_w}] + \sum_{w \in W} \sum_{p \in P_w} v_{wp} [M - (M - d_w) x_{wp} - \sum_{l \in L} \delta_{lp} (D_l(f_l) + a_l) + \frac{T_{wp} \bar{R}_{wp} \sigma_w}{1 - \bar{R}_{wp} \sigma_w}]$$

$$+\sum_{w\in W}\sum_{p\in P_w} s_{wp}[1-(1-l_w)x_{wp}$$
$$-\sum_{l\in L}\delta_{lp}[L_l(f_l)+q_l^h-L_l(f_l)q_l^h+q_l^d]] \quad (RPLR)$$

subject to (1), (2) and (10).
Note that the constraints are dualized in a way such that the corresponding multipliers are nonnegative.

(RPLR) can be decomposed into two independent subproblems. Note that the constant terms, e.g. the product terms involving T_{wp} or q_l^d, were omitted in the objective function in the subproblems.

Subproblem 1:

$$\max \sum_{l\in L}[u_l f_l - D_l(f_l)\sum_{w\in W}\sum_{p\in P_w} v_{wp}\delta_{lp}$$
$$-L_l(f_l)(1-q_l^h)\sum_{w\in W}\sum_{p\in P_w}\delta_{lp}s_{wp}]$$

subject to (10) and

Subproblem 2:

$$\max \sum_{w\in W}\sum_{p\in P_w}[c_w+\sum_{l\in L}\frac{\gamma_w\delta_{lp}u_l}{1-\bar{R}_{wp}\sigma_w}$$
$$+(M-d_w)v_{wp}+(1-l_w)s_{wp})]x_{wp}$$

subject to (1) and (2).

Subproblem 1 is composed of $|L|$ (one for each link) independent problems. Each of these problems is the maximization of a concave univariate function over a simple interval. Note that $D_l(f_l)$ and $L_l(f_l)$ are convex functions and all of the multipliers are nonnegative. Subproblem 1 is solved by a line search procedure on an interval for f_l.

Subproblem 2 consists of $|W|$ (one for each user pair) independent shortest path problems. For each user pair w, find a path in P_w with the smallest path cost $[\sum_{l\in L}\gamma_w\delta_{lp}u_l(1-\bar{R}_{wp}\sigma_w)^{-1}+(M-d_w)v_{wp}+(1-l_w)s_{wp})]$. If c_w is greater or equal to the length of the shortest path, assign the session to the shortest path found. Otherwise, reject the session ($x_{p_w'}=1$).

From the weak Lagrangean duality theorem, for any $(u,v,s)\geq 0$, $Z_D(u,v,s)$ is an upper bound on Z_{RP} [6]. Naturally, one wants to determine the smallest upper bound by

$$Z_D = \min_{u,v,s\geq 0} Z_D(u,v,s). \quad (D)$$

An upper bound on Z_{RP} can be determined as $\min\{Z_D, \sum_{w\in W} c_w\}$.

In the computational experiments discussed in section 8, three cases are considered. In the first case, the network parameters are such that it is feasible to admit all user pairs. For this case, we will develop the solution procedure SAT. In the second case, the network capacity is not sufficient to accommodate all of the offered sessions and satisfy their QOS requirements. For this case, we will develop PA&AC which is a variation of the basic algorithm SAT. In the third case, we consider path assignments and admission control in an operational network. We develop QD-PA&AC which makes admission control decisions and path assignments for newly arriving sessions.

There are several methods for solving the dual problem (D). The most popular method is the subgradient method [4, 6, 9]. Let an $|L|+2\sum_{w\in W}|P_w|$ vector y be a subgradient of $Z_D(u,v,s)$. In iteration k of the subgradient optimization procedure, the multiplier vector $b^k=(u^k,v^k,s^k)$ is updated by

$$b^{k+1} = b^k + t^k y^k. \quad (13)$$

The step size t^k is determined by

$$t^k = \delta\frac{Z_D(b^k) - Z_{RP}^h}{\|y^k\|^2} \quad (14)$$

where Z_{RP}^h is the objective function value for a heuristic solution (lower bound on Z_{RP}) and δ is a constant, $0<\delta\leq 2$. For Case I, Z_{RP} is known *a priori*. Using $Z_{RP}^h = Z_{RP}$ guarantees that the sequence of multipliers converges to an optimal solution [7].

The multiplier updating rule given by (13) and (14) is not suitable since too many overhead messages must be passed around the network [18]. For a distributed algorithm, the following multiplier updating rule [8] is adopted.

$$b^{k+1} = b^k + t^k y^k. \quad (15)$$

This multiplier updating rule is better suited for distributed computation since it does not involve the calculation of $\|y^k\|$. If the step sizes t^k are chosen to satisfy the two conditions: (i) $\lim_{k\to\infty} t^k = 0$ and (ii) $\sum_{k=1}^{\infty} t^k \to \infty$, then (D) is solved optimally [20].

For Case I, we propose the following heuristic to find good solutions to (RP).

SAT:

0. **Initialization**
   - 0.a Generate a candidate route set for each user pair.
   - 0.b Assign a nonnegative value to each multiplier.
   - 0.c Set the iteration counter, k, to zero.
1. **Solve the Lagrangean Relaxation**
   - 1.a Solve subproblem 1.
   - 1.b Solve subproblem 2.
2. **Adjust the multipliers**

   Use (15) to adjust the multipliers.
3. **Calculate heuristic routing assignments**

   Use $\{x_{wp}\}$ obtained in Step 1.b as the heuristic path assignments.
4. **Test stopping criteria**
   - 4.a If the number of iterations has reached the predetermined limit or the QOS requirements of each user pair are satisfied, stop.
   - 4.b $k \leftarrow k+1$.
   - 4.c Go to Step 1.

In steps 1.a and 1.b, subproblems 1 and 2 are solved using distributed computation. In the solution to subproblem 2, each user pair is assigned a path in the network. In step 4.a, the QOS is checked for each user pair. If one or more users' requirements are not satisfied, the multipliers are adjusted and a new path assignment is found (by solving (RPLR)).

Over time the number of users and their utilization of the network will increase so that it would be infeasible to

admit all users and still satisfy their QOS requirements. We refer to this case as moderately loaded and unbalanced or heavily loaded (Case II). For this case, we (i) delete the QOS requirement criterion in step 4.a of SAT and (ii) replace step 3 of SAT with the procedure DROP which is given below. We refer to the modified version of SAT as **PA&AC**.

Procedure **DROP:**

3.a Use $\{x_{wp}\}$ obtained in Step 1.b as the tentative path assignments and admission controls.

3.b If the QOS requirements of every admitted session is satisfied, go to Step 4.

3.c Remove (reject) one tentatively admitted session at a time whose QOS requirements are not satisfied.

3.d For each dropped session, adjust the aggregate flow, delay and loss probability for each link on the assigned path.

3.e Go to Step 3.b.

The procedure DROP is very primitive. This is consistent with our approach where the efficiency of the algorithm is emphasized rather than its effectiveness in finding near-optimal solutions. The computational results reported in section 8 show that PA&AC finds solutions that are approximately 5% from an optimal solution very quickly. A more sophisticated procedure can be developed if higher quality solutions are needed.

In Case III, we consider the modified model where there are existing and new sessions. For this case, we propose QD-PA&AC, which is a *quasi-dynamic* version of PA&AC. QD-PA&AC differs from PA&AC in three ways. First, in step 1.b of PA&AC (solve subproblem 2), the path assignments for the existing sessions are fixed. Here new sessions are not permitted to preempt the existing sessions. Second, the procedure DROP is only applied to new sessions. Third, another procedure is added after DROP. After applying DROP, the QOS requirements of all admitted new sessions are satisfied, but the QOS requirements of some of the existing sessions may not. If this occurs, an amount Δ (in the experiments $\Delta = 0.0002$) is added to the multiplier u_l for each link on the path corresponding to the existing session whose QOS is not satisfied. This has the effect of causing new sessions to be rerouted away from links on this path.

6 Getting True Upper Bounds

By using the approach discussed in the previous section, one can obtain upper bounds on Z_{RP}. But those bounds may not be true upper bounds on Z_{IP}. In this section, we propose a way to find true upper bounds on Z_{IP} to evaluate the quality of the heuristic solutions by applying PA&AC and QD-PA&AC.

As illustrated in Figure 3, we construct the following upper-bounding problem (UBP) whose feasible set Ω_{UBP} is convex (without considering the integrality constraints) and contains the nonconvex set Ω_{IP}. Let H_p be the number of hops of path p. Let $\bar{L}$ be an upper bound on L_l for all $l \in L$.

$$Z_{UBP} = \max \sum_{w \in W} c_w \sum_{p \in P_w} x_{wp} \qquad (UBP)$$

subject to (1), (2), (10) and

$$\sum_{w \in W} \sum_{p \in P_w} \gamma_w x_{wp} \delta_{lp} \prod_{l \in h_p} (1 - \bar{L}_l - q_l^h + \bar{L}_l q_l^h) \le f_l \quad \forall l \in L \quad (16)$$

$$\sum_{l \in L} \delta_{lp}(D_l(f_l) + a_l) \le M - (M - d_w)x_{wp} \quad \forall p \in P_w, w \in W \quad (17)$$

$$\underline{L}_p \le 1 - (1 - l_w)x_{wp} \quad \forall p \in P_w, w \in W \quad (18)$$

where

$$\underline{L}_p = \sum_{l \in L} \delta_{lp}(L_l(f_l) + q_l^h - L_l(f_l)q_l^h) + \prod_{l \in h_p} (1 - \bar{L}_l - q_l^h + \bar{L}_l q_l^h)(1 - \prod_{l \in h_p} (1 - q_l^d)) - \frac{H_p(H_p - 1)}{2} \bar{L}^2.$$

Comparing the left hand side of (16) with the left hand side of (3), it can be seen that the aggregate link flows are underestimated. In (17), the average end-to-end cell delays are underestimated [cf. (4) and (8)]. In (18), the average end-to-end cell loss probabilities are underestimated [cf. (5) and (6)]. Let $\{E_i\}$ be a set of mutually independent events and $\{p_i\}$ be the corresponding probabilities. The above statement can be proven by applying the following result: $Pr\{\bigcup_i E_i\} \ge \sum_i p_i - \sum_{i \ne j} p_i p_j$.

We next apply Lagrangean relaxation to obtain an upper bound on Z_{UBP}, which is a true upper bound on Z_{IP}.

7 Implementation

In this section, several implementation issues are discussed. We first propose a session aggregation procedure which substantially reduces the size (the number of variables) of both the primal and dual problems. Then, an analysis of the computation time needed in a distributed implementation of the algorithm is given.

The size of (RPLR) and (D) is approximately proportional to the total number of sessions which can be very large in a practical application. To reduce the size of the problem, we take advantage of the property that single-path routing performs very well when the load on the network is not heavy [18]. In single-path routing, all of the sessions of an origin-destination node pair are routed over the same path. However, for sessions with different traffic types and the same origin and destination, different paths may be used. We aggregate the sessions of the same traffic type and the same origin and destination into a single macro session. All sessions aggregated into a macro session must have the same QOS requirements. For example, there may be 100 voice users with the same origin and destination. For this O-D pair, the total voice traffic would be 6.4 Mbps (100×64 Kbps) which is far less than the traffic requirement of a single video user.

The aggregation of sessions corresponds to the imposition of additional constraints. Let (UBP1) denote (UBP) with additional constraints corresponding to the aggregation of sessions. Consider (LRUBP1) the Lagrangian Relaxation of (UBP1) obtained by dualizing (16), (17) and (18). The optimal objective function value of (LRUBP1), Z_{LRUBP1}, is an upper bound on Z_{IP}. This follows from the fact that the solution to the second subproblem of (LRUBP1) and the Lagrangean relaxation of (UBP) are the same. Let (RP1)

be (RP) with the additional session aggregation constraints. Then any feasible solution to (RP1) is a feasible solution to (RP) and (IP).

We next analyze the computation time of the algorithm needed in a distributed implementation. As in [19], we can show that the speedup of the algorithm using distributed computation over a centralized algorithm is approximately $|L|/G$ where G is the maximum outdegree in the network. (This result is based upon the assumption that the communication delays are negligible compared with the computation time. This assumption is valid since we are considering ATM (high-speed) networks.)

8 Computational Results and Discussion

The algorithms SAT, PA&AC and QD-PA&AC described in Section 5 were coded in FORTRAN 77 and run on a SUN 4/490 workstation[2]. The golden section search procedure was used to solve subproblem 1. The number of iterations used in the line search was 20.

The topologies of the 10 networks used in the computational experiments can be found in [17]. For each of the 10 networks, the capacity of each optical trunk is assumed to be 3.6 Gbps which corresponds to 24-150Mbps channels. The assumed architecture of each ATM switch is shown in Figure 1 where $c = 10$ and $b = 100$.

It was assumed that there were 3 types of traffic – video, voice and data. The cell length (for each traffic type) was assumed to be 53 bytes. The average rates of traffic generated for each established session of traffic types 1, 2 and 3 were 50 $Mbps$, 64 $Kbps$ and 1 $Mbps$, respectively. For each O-D node pair and each traffic type, the number of sessions were generated by rounding down to an integer the value generated from a uniform distribution.

As discussed in section 7, we aggregated into a single macro session the sessions of the same traffic type and the same O-D node pair. For each O-D node pair, since there are at most three traffic types, there are at most three macro sessions. For each macro session, there are at most three candidate paths. We chose the first candidate path to be a minimum hop path. The other candidate paths are shortest paths with respect to two sets of randomly generated arc weights.

As given in [21], the acceptable end-to-end cell loss probabilities for the video and voice traffic were assumed to be 10^{-8} and 10^{-3}, respectively. Since the bit error rates for optical fiber is extremely small (10^{-12} to 10^{-13}) compared to the cell loss probability at the ATM switches, we ignored the discarding of cells due to errors in the header or data.

The acceptable end-to-end cell delays for video, voice and data traffic were assumed to be 0.1, 0.1, and 1 second [13], respectively. We chose M to be 1 second. For a typical ATM network, the constraint on the end-to-end delay will not be binding. Consider an ATM network where the longest candidate path is less than 6000 km in length. Suppose this path has 100 ATM switches. Suppose the buffer for each output queue is 100. By assuming that there are 99 cells in each buffer, the end-to-end delay (assuming a propagation speed of 2×10^5 km/sec) is less than 58.3 $msec$. This is quite a bit less than the acceptable end-to-end cell delay for each traffic type. Therefore, we expect that the delay constraints in (RP) will not be binding and the corresponding multipliers to be 0 in an optimal dual solution.

For each link, the maximum utilization factor α_l was chosen so that constraint (11) would not be binding. That is, the link utilization would be limited by the constraint on the end-to-end cell loss probability rather than by constraint (11). For each link, we set $\alpha_l = 0.93$ which corresponds to a cell loss probability of 3.77×10^{-8}. A primary impetus for the development of high-speed networks is to provide home entertainment such as HDTV [15]. Thus, it is very likely that every optical trunk in an ATM network will carry video traffic. Therefore, the utilization of each link must be less than 0.93 so the link cell loss probability is less than 10^{-8} (the acceptable end-to-end cell loss probability for video).

8.1 Case I : Lightly Loaded Network

In Case I, we assumed network parameters so that the network is lightly loaded. In this case, it is feasible to admit all sessions and satisfy the QOS requirements of each session. For this case, only SAT is needed. Due to a limit on the paper length, the results corresponding to this set of experiments are not reported. The interested reader is referred to [17].

8.2 Case II : Heavily Loaded Network

In Case II, we assumed network parameters such that the admission controls had to be applied. As a reference, we consider the path assignment and admission control algorithm M&D, which is the minimum hop routing (MHR) in conjunction with DROP. In this set of experiments, we compare the relative effectiveness of PA&AC with M&D. We also investigate the relative sizes of Ω_{RP} and Ω_{UBP}.

Three networks (ARPA1, OCT, and GTE) were tested. The objective function used was the total network throughput ($c_w = \gamma_w$). For PA&AC, the distributed multiplier updating rule was used where the step sizes were $t^k = 2^{-32}(k+1)^{-1}$. The initial multipliers were $(u^0, v^0, s^0) = (0, 0, 0)$.

The computational results are reported in Table 1. For each O-D pair and each traffic type, a sample value was generated from a uniform distribution between 0 and an upper limit. The upper limit for each of the three traffic types is in the second column. The sample value was rounded down to obtain the number of sessions. The propagation delay on each link was chosen to be a sample value generated from a uniform distribution with a range of $[\theta, 2\theta]$. The fourth column is the smallest upper bound on Z_{RP}. Recall that this is the best objective function value of the dual problem. The fifth column gives the smallest upper bound on Z_{UBP}. The numbers in the fifth column are also true upper bounds on Z_{IP}, the optimal objective function value of the original problem (IP). Column 6 gives $Z^{PA\&AC}$, the best objective function value (throughput) found for (RP). In column 7, the percentage difference between columns 5 and 6 is given. Since true upper bounds given in column 5 are used in these calculations, the numbers in column 7 overestimate the true percentage error. The eighth column reports $Z^{M\&D}$, the throughput found by applying M&D. The

[2]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.

ninth column gives the percentage improvement in throughput of PA&AC over M&D. Column 10 gives the number of iterations PA&AC performed. Column 11 reports the required CPU time when the implementation of PA&AC is distributed. The reported CPU times were the times to obtain heuristic solutions. The programs were then run longer to find better upper bounds.

From an inspection of Table 1, PA&AC provides a 36% improvement in the throughput over M&D. PA&AC required less than 1.5 seconds of CPU time for networks with up to 61 nodes and 10,000 macro sessions. The error bounds are within 6.1%. We believe that this gap is primarily due to the simplicity of DROP. At the expense of additional CPU time, we could have easily developed an algorithm that is more effective than DROP. However, one of our algorithm design objectives was speed of decision making. Another observation from Table 1 is that the upper bounds on Z_{RP} is very close to those on Z_{UBP}. This suggests that the difference between Ω_{RP} and Ω_{UBP} is small. This is due to the extremely small cell loss probabilities.

8.3 Case III : Quasi-Dynamic Implementation

In this set of experiments, we tested QD-PA&AC under a set of conditions that resembles the actual operational environment of a network. In Figure 4, new sessions arrive and existing sessions terminate in the interval $[t_0, t_1)$. Path assignments and admission controls for the new sessions are computed in the interval $[t_1, t_2)$ and then implemented at time t_2. In our discussion, we will consider $[t_0, t_1)$ and $[t_1, t_2)$ to be a typical cycle in steady state. Let W_E be the set of existing sessions at time t_0. Let W_N and W_T be the set of sessions to arrive and terminate, respectively, in the interval $[t_0, t_1)$.

As before, we used the total network throughput as the objective function ($c_w = \gamma_w$). In the experiments, we used the 3 sets of problem parameters for OCT in Case II. We intentionally chose the most difficult test problems from Case II. That is, the three test problems for OCT required the most CPU time (see Table 3). This is due to the topological structure of OCT and the loading. The determination of W_E was as follows. The number of offered sessions for each O-D pair and each traffic type was generated randomly as in Case II. PA&AC was applied to determine the admitted sessions and their path assignments. W_E is the resulting set of admitted sessions. The final set of multipliers in PA&AC were used as the initial set of multipliers for QD-PA&AC.

We next discuss the sets W_T and W_N. Let $\bar{t}$ be the length of each of the intervals $[t_0, t_1)$ and $[t_1, t_2)$. In the experiments, $\bar{t}$ was varied between 0.05 and 1 second so that the session *setup* time was at most 2 seconds.

We assumed video, voice and data sessions arrive to the network as independent Poisson processes with respective average arrival rates $\lambda_1 = (180)^{-1}$, $\lambda_2 = (12)^{-1}$ and $\lambda_3 = (12)^{-1}$ (sessions/second). We also assumed that the holding times for video, voice and data sessions were independent and exponentially distributed with respective means $\mu_1^{-1} =$ 1 hour, $\mu_2^{-1} = 20$ minutes and $\mu_3^{-1} = 20$ minutes.

We next show that W_T, the subset (of W_E) of sessions that terminate in $[t_0, t_1)$, is small. The probability that a particular video session terminates in $[t_0, t_1)$ is $1 - e^{-\frac{\bar{t}}{3600}}$. For $\bar{t} = 1$ second, the expected number of video sessions that terminate in $[t_0, t_1)$ is $20(1 - e^{-\frac{1}{3600}}) = 0.0056$. In a similar way, it can be shown that the expected number of voice and data sessions that terminate in 1 second is 0.0833. In the experiments, W_T was ignored so that the conditions were more difficult (the load is heavier) for QD-PA&AC to find a solution.

We next characterize W_N. Recall that sessions of the same traffic type and with the same O-D pair are aggregated. The probability that there will be 1 new macro video session is the probability that there will be at least 1 new video session arrival in $[t_0, t_1)$ or $1 - e^{-\frac{\bar{t}}{180}} = 0.0055$ (when $\bar{t} = 1$). Similarly, the probability of 1 new macro voice (data) session arrival in $[t_0, t_1)$ or $1 - e^{-\frac{\bar{t}}{12}} = 0.08$. In the experiments, we used the above probabilities to randomly generate the number of new macro sessions for each O-D pair. For example, for a particular O-D pair, we generated 1 new macro video session with probability 0.0055 and 0 macro video sessions with probability 0.9945.

The computational results are reported in Table 2. The first three columns report the same information as the corresponding columns in Table 1. Note that the upper limits given in the second column were used to generate W_E. For each W_E, $\bar{t}$ was varied from 50 msec to 1 second. In other words, the session set up times were varied from 100 msec to 2 seconds. Note as $\bar{t}$ was varied, W_N varied and the new offered load varied. The fourth column gives the various values of $\bar{t}$. The values of $\bar{t}$ are also the maximum allowed computation times for QD-PA&AC. These times were converted into maximum number of iterations allowed which are given column 5. Also in column 5, the number of iterations needed to obtain the best heuristic solutions are given in parentheses. In the sixth column, the smallest true upper bound on the optimal objective function value is given for each test problem. This true upper bound was obtained by applying Lagrangean relaxation to solve the upper-bounding problem of the modified model. The smaller of this value and $\sum_{w \in W_E \cup W_N} c_w$ is given in column 6. In the experiments, we found that $\sum_{w \in W_E \cup W_N} c_w$ always gave a tighter upper bound. The error bound is [(upper-bound − lower-bound)×100/lower-bound].

From an inspection of Table 2 we find that QD-PA&AC solved 79% of the test problems optimally and found near-optimal (within 0.15%) in the other 21% of the test problems. Furthermore, the number of iterations needed is far less than the number of iterations allowed for Case II (highly loaded networks). There are two reasons for the additional efficiency and effectiveness of QD-PA&AC over PA&AC. First, the number of primal decision variables in the modified model is much smaller since most of the path assignments were fixed. The CPU time required for each iteration of QD-PA&AC is approximately the same as the CPU time needed for each iteration of PA&AC. For the modified model, there is no need to solve subproblem 2 for those $w \in W_E$. This results in fewer iterations needed to find a near-optimal solution for the modified model. Second, each of the test problems in Case III can be considered as a perturbation of the corresponding test problem in Case II. Recall that a good set of multipliers were calculated for a test problem (Case II) and then used as the initial set of multipliers for the corresponding perturbed problem (Case III). Then only a few iterations were needed to calculate a good set of multipliers for the perturbed problem. These multipliers were then used to calculate good path assignments and admission controls for the new sessions. Another observation from Table 2 is for the third set of existing sessions W_E, the error bounds increased as $\bar{t}$ was increased. As $\bar{t}$ was increased, $|W_N|$ and the offered load increased. As $\bar{t}$

approached 0.4 sec (from below), the network became saturated and some sessions in W_N were rejected. For $\bar{t} \geq 0.4$ seconds, the total throughput remained the same. However, the upper bound $\sum_{w \in W_E \cup W_N} c_w$ increased as $\bar{t}$ was increased. Thus, the error bound increased as $\bar{t}$ (and $|W_N|$) was increased.

References

[1] J.J. Bae and T. Suda. Survey of traffic control schemes and protocols in ATM networks. *Proceedings of the IEEE*, 79(2):170–189, February 1991.

[2] A. Bhargava, J.F. Kurose, D. Towsley, and G. Vanleemput. Performance comparison of error control schemes in high-speed computer communication networks. *IEEE JSAC*, 6(9):1565–1575, December 1988.

[3] W. Doeringer, D. Dykeman, M. Kaiserswerth, B. Meister, H. Rudin, and R. Williamson. A survey of lightweight transport protocols for high-speed networks. *IEEE Transactions on Communications*, 38(11):2025–2039, November 1990.

[4] M.L. Fisher. The Lagrangian Relaxation method for solving integer programming problems. *Management Science*, 27(1):1–18, January 1981.

[5] B. Gavish and S.L. Hantler. An algorithm for optimal route selection in SNA networks. *IEEE Transactions on Communications*, COM-31(10):1154–1160, October 1983.

[6] A.M. Geoffrion. Lagrangean relaxation and its uses in integer programming. *Math. Programming Study*, 2:82–114, 1974.

[7] J.L. Goffin. On the convergence rates of subgradient optimization methods. *Math. Programming*, 13:329–347, 1977.

[8] M. Held and R.M. Karp. The traveling salesman problem and minimum spanning trees: Part II. *Mathematical Programming*, 1:6–25, 1971.

[9] M. Held, P. Wolfe, and H.D. Crowder. Validation of subgradient optimization. *Math. Programming*, 6:62–88, 1974.

[10] M.G. Hluchyj and M.J. Karol. Queueing in high-performance packet switching. *IEEE JSAC*, SAC-6(9):1587–1597, December 1988.

[11] J.Y. Hui and E. Arthurs. A broadband packet switch for integrated transport. *IEEE JSAC*, SAC-5(8):1264–1273, October 1987.

[12] CCITT Recommendations I.121. Broadband aspects of ISDN. Technical report, CCITT, Melbourne, 1988.

[13] P. Kirton, J. Ellershaw, and M. Littlewood. Fast packet switching for integrated network evolution. In *Proc. ISS87*, 1987.

[14] L. Kleinrock. *Queueing Systems*, volume 1 and 2. New York: Wiley-Interscience, 1975 and 1976.

[15] L. Kleinrock. ISDN – the path to broadband networks. *Proceedings of the IEEE*, 79(2):112–117, February 1991.

[16] M.J. Lee and James R. Yee. Optimal minimax routing in ATM networks. In *Proceedings IEEE Globecom'90*, pages 505.5.1–505.5.5, 1990.

[17] F. Y.S. Lin. *Routing and Flow Control in Virtual Circuit Networks*. PhD thesis, Electrical Engineering Dept., University of Southern California, Loa Angeles, California, 1991.

[18] F. Y.S. Lin and J.R. Yee. A distributed routing algorithm for virtual circuit data networks. In *Proc. 1989 INFOCOM*, pages 200–207, 1989.

[19] 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 J. on Computing, Summer 1992.

[20] B.T. Polyak. A general method for solving extremal problems. *Soviet Mathematics Doklady*, 8:593–597, 1967.

[21] CCITT Study Group XVIII. CCITT Report R4. Technical report, CCITT, February 1989.

[22] Y.S. Yeh, M.G. Hluchyj, and A.S. Acampora. The knockout switch: A simple architecture for high performance packet switching. *IEEE JSAC*, SAC-5(8):1274–1283, October 1987.

Table 1 : Comparison of PA&AC and M&D

Network ID	Range of No. of Sessions	θ (msec)	Upper Bounds on Z_{RP}	Upper Bounds on Z_{UBP}	$Z^{PA\&AC}$ (× 10 Gbps)	Error Bounds (%)	$Z^{M\&D}$ (× 10 Gbps)	$\frac{Z^{PA\&AC} - Z^{M\&D}}{Z^{M\&D}}$ (%)	No. of Iter.	CPU Time (sec)
ARPA1	[2,100,20]	2	8.49200	8.49784	8.01473	6.0278	5.55584	44.26	50	1.159
ARPA1	[1.5,300,40]	2	9.38116	9.38563	8.93334	5.0629	6.06901	47.20	50	1.116
ARPA1	[1.3,200,20]	2	7.42470	7.42677	7.07028	5.0421	5.36732	31.77	50	1.135
OCT	[3,1000,100]	2	5.58488	5.58776	5.29469	5.5352	3.02334	75.13	200	1.393
OCT	[4,100,50]	2	5.00647	5.00880	4.72336	6.0431	2.95323	59.94	200	1.367
OCT	[2,500,80]	2	4.47751	4.47938	4.30526	4.0444	3.49292	23.26	200	1.447
GTE	[10,3000,500]	8	7.46751	7.46850	7.14304	4.5563	6.17185	15.74	200	0.714
GTE	[8,3500,700]	8	7.96348	7.96588	7.56520	5.2964	6.64087	13.92	200	0.714
GTE	[9,5000,600]	8	8.15144	8.15395	7.78526	4.7357	6.67069	16.71	200	0.714

Table 2 : Summary of computational results for QD-PA&AC

Network ID	Range of No. of Sessions	θ	Computation Time Allowed (sec)	No. of Iter. Allowed	Lower Bounds (×10 Gbps)	Upper Bounds (×10 Gbps)	Error Bounds (%)
OCT	[3,1000,100]	2	0.05	7 (1)*	5.29491	5.29491	0
OCT	[3,1000,100]	2	0.1	14 (1)	5.29512	5.29512	0
OCT	[3,1000,100]	2	0.2	28 (20)	5.29565	5.29565	0
OCT	[3,1000,100]	2	0.3	42 (20)	5.30102	5.30102	0
OCT	[3,1000,100]	2	0.4	57 (20)	5.30674	5.30674	0
OCT	[3,1000,100]	2	0.5	71 (20)	5.30697	5.30697	0
OCT	[3,1000,100]	2	0.6	85 (20)	5.30709	5.30709	0
OCT	[3,1000,100]	2	0.7	100 (20)	5.30765	5.30765	0
OCT	[3,1000,100]	2	0.8	114 (20)	5.31318	5.31318	0
OCT	[3,1000,100]	2	0.9	128 (20)	5.31841	5.31841	0
OCT	[3,1000,100]	2	1.0	142 (20)	5.31877	5.31877	0
OCT	[4,100,50]	2	0.05	7 (1)	4.72358	4.72358	0
OCT	[4,100,50]	2	0.1	14 (1)	4.72379	4.72379	0
OCT	[4,100,50]	2	0.2	28 (4)	4.72432	4.72432	0
OCT	[4,100,50]	2	0.3	42 (7)	4.72969	4.72969	0
OCT	[4,100,50]	2	0.4	57 (7)	4.73541	4.73541	0
OCT	[4,100,50]	2	0.5	71 (10)	4.73565	4.73565	0
OCT	[4,100,50]	2	0.6	85 (10)	4.73576	4.73576	0
OCT	[4,100,50]	2	0.7	100 (10)	4.73632	4.73632	0
OCT	[4,100,50]	2	0.8	114 (31)	4.74185	4.74185	0
OCT	[4,100,50]	2	0.9	128 (31)	4.74708	4.74708	0
OCT	[4,100,50]	2	1.0	142 (31)	4.74744	4.74744	0
OCT	[2,500,80]	2	0.05	7 (1)	4.30548	4.30548	0
OCT	[2,500,80]	2	0.1	14 (1)	4.30569	4.30569	0
OCT	[2,500,80]	2	0.2	28 (17)	4.30622	4.30622	0
OCT	[2,500,80]	2	0.3	42 (17)	4.31159	4.31159	0
OCT	[2,500,80]	2	0.4	57 (17)	4.31220	4.31731	0.119
OCT	[2,500,80]	2	0.5	71 (2)	4.31242	4.31755	0.119
OCT	[2,500,80]	2	0.6	85 (15)	4.31254	4.31766	0.119
OCT	[2,500,80]	2	0.7	100 (17)	4.31309	4.31821	0.119
OCT	[2,500,80]	2	0.8	114 (17)	4.31853	4.32375	0.121
OCT	[2,500,80]	2	0.9	128 (75)	4.32353	4.32898	0.126
OCT	[2,500,80]	2	1.0	142 (79)	4.32376	4.32934	0.129

* Numbers in parentheses are the iterations in which the best primal feasible solution was first found.

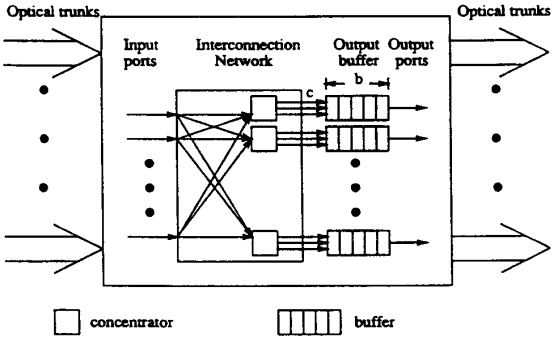

Fig. 1. An ATM switch model where output queueing is used

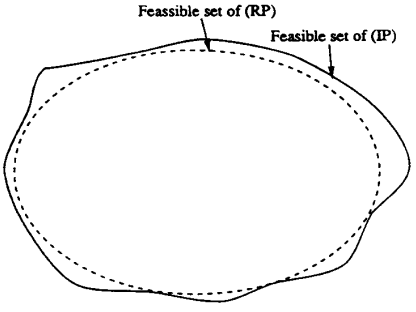

Fig. 2. Feasible Sets of (IP) and (RP)

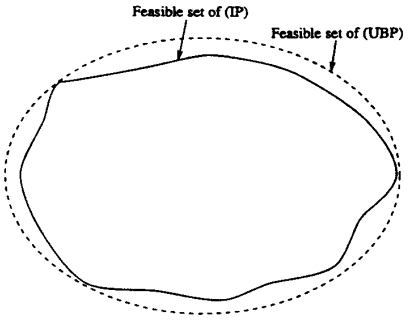

Fig. 3. Feasible Sets of (IP) and (UBP)

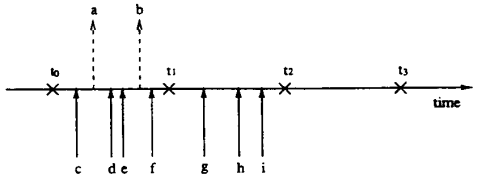

Fig. 4. A Schematic Illustraction of the Implementation of the Proposed Algorithm

Fig. 5. Cell loss probabilities of an output-queueing ATM switch

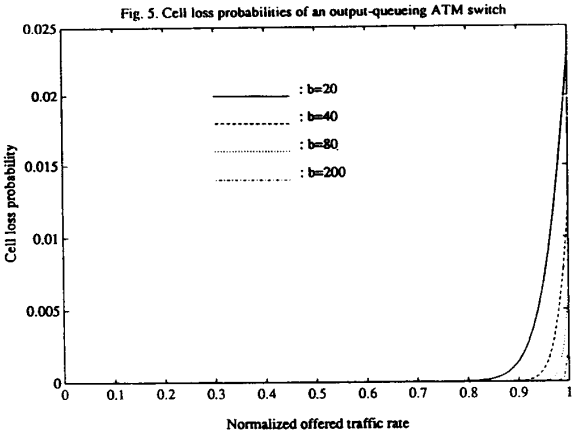

Fig. 6. Average cell delays of an output-queueing ATM switch