Routing and Admission Control of Virtual Circuits in General Topology Networks

Rainer Gawlick[1] Anil Kamath[2,3] Serge Plotkin[2] K. G. Ramakrishnan[3]

[1] MIT Laboratory for Computer Science
[2] Stanford University
[3] AT&T Bell Laboratories

January 16, 1995

Abstract

Emerging high speed Broadband Integrated Services Digital Networks (B-ISDN) are expected to carry traffic for services like video-on-demand and video teleconferencing which will require resource reservation along the path on which the traffic is sent. As a result, such networks will need efficient *routing* and *admission control* algorithms. The simplest approach is to use fixed paths and no admission control. More sophisticated approaches which use state dependent routing and a form of admission control called trunk reservation can be found in the circuit switching literature. However, the circuit switching literature has generally focused on fully connected (complete) networks.

This paper suggests a new routing and admission control algorithm for *general topology* networks. Our algorithm is an adaptation of a recently discovered theoretical algorithm that is asymptotically best possible with respect to the worst case performance. The main idea behind our algorithm was to improve the performance in the expected instead of the worst case. Although the original theoretically optimum algorithm behaves quite badly in practice, we prove that our modifications dramatically improve the performance under several classes of traffic loads.

The paper considers both a centralized setting, where routing and admission control decisions are made by a centralized processor, and a *distributed* setting, where decisions are made at the source node, using potentially inaccurate state information. We show that our algorithm can be easily adapted to the distributed setting.

We have evaluated the performance of our algorithms using extensive simulations on an existing commercial network topology and the NSFNet T3 backbone topology. The simulations show that our modified algorithm outperforms several existing algorithms. The increase in performance is most dramatic in the distributed setting, indicating that our algorithm is able to use outdated information more effectively.

1 Introduction

The wide spectrum of new consumer services, like video-on-demand, video teleconferencing, etc. expected to be offered on emerging high speed Broadband Integrated Services Digital Networks (B-ISDN), will tax network resources despite rapid technology advances. As a result, the allocation of network resources is of critical importance. The industry consensus is that B-ISDN will be based on the Asynchronous Transfer Mode (ATM) protocol. For ATM, each communication stream between a source and a destination uses a virtual circuit, which is a fixed path through the network. Most applications like voice and video need quality-of-service (QOS) guarantees. As a result, such applications must reserve resources along their virtual circuit paths. Resource reservation, in turn, implies the need for admission control to ensure that resources are not reserved beyond those that are physically available. Thus B-ISDN requires fast and efficient on-line *routing* and *admission control* algorithms.

Generally, the objective of a routing and admission control algorithm is to maximize the throughput of the network without violating resource constraints. Many factors complicate routing and admission control decisions. First, the decisions must be made *on-line* without knowledge of future requests. Second, the current state of the network may not be available, thus the routing decision may be based on static or inaccurate state information. Third, the decision must be made in real time i.e., it has to adhere to call setup time requirements.

In this paper we propose a distributed routing and admission control algorithm for Switched Virtual Circuits (SVCs) in general topology networks. Our algorithm can be viewed as a *practical* adaptation of the theoretical framework developed in [3, 2]. The main advantage of the algorithms in [3, 2] is that they achieve provably good performance without relying on probabilistic assumptions on the offered traffic. For example, for any sequence of requests, the algorithm in [3] achieves at least $O(1/\log nT)$ fraction of the throughput achievable by the optimum off-line algorithm that is presented with all the requests in advance, where n is the number of nodes in the network and T is the maximum holding time of a circuit.

Unfortunately, the fact that algorithms in [2, 3] do not rely on any probablilistic assumptions has the effect that they are tuned to deal with worst case situations, thus making these algorithms impractical. The algorithm, EXP, developed in this paper, is an adaptation of the centralized worst-case algorithm in [3] to the more realistic distributed average-case situation. The algorithm works by choosing an appropriate state dependent metric to determine the "cost" of each link, based on the *local knowledge* about the global state of the network. Admission control is done by rejecting virtual circuits for which there is no path with sufficiently low cost and sufficient available capacity (bandwidth). If an appropriate path exists, it is used to route the circuit. Otherwise, the request is rejected. The EXP algorithm is described for the case of SVCs with known holding times. The issue of using statistical information about the holding times instead is addressed in a forthcoming paper.

We have conducted extensive simulation experiments on a realistic topology to compare the EXP algorithm to several other techniques. The simulations indicate the following.

- Our EXP algorithm performs significantly better than techniques like greedy minimum hop and minimum hop with trunk reservation (e.g. the algorithm of Sibal and DeSimone [12]).
- Unlike minimum hop based techniques, the performance of our EXP algorithm is not dramatically affected by outdated and inaccurate state information. Thus, its relative performance advantage is even greater in the distributed setting.

Routing and admission control algorithms, for both centralized and distributed environments have been studied in the past. A class of *cost-based* routing algorithms have been studied for general networks in theoretical framework by Kelly [7] in the context of state-independent routing, and by Ott and Krishnan [11] in the context of state-dependent routing. Roughly speaking, these algorithms are based on the concept of costs that reflect the effect of a routing decision on the system performance. These costs are used to assign probabilities to different alternate routes. Since the costs are computed using an a priori known traffic matrix, these algorithms are in general not dynamic in the sense that they do not seek to exploit fluctuations in the traffic rates.

Another important class of routing schemes is based on the concept of *trunk reservation* [4, 10, 9, 8]. These schemes were designed for circuit switched routing in *fully connected* networks with primary direct paths and alternate paths for overflow traffic. Roughly speaking, the idea is to reserve some of the capacity on the direct paths, and allow an alternative path to use only edges with available capacity that exceeds the reservation parameter. This is in contrast to the greedy admission control policy that would admit any virtual circuit as long as there is sufficient capacity to accommodate it. Intuitively, the reservation is beneficial since (when properly tuned) it will reject a virtual circuit request when the only available path for the virtual circuit uses resources that might be used more efficiently by future virtual circuits.

An example of an existing admission control scheme based on trunk reservation is the Real Time Network Routing (RTNR) [1] used in the AT&T long distance network. Logically, this network is a complete graph. All calls are initially routed along the direct link and the overflow traffic uses the least busy 2-hop alternate path. The trunk reservation in RTNR is implemented by forbidding the use of a 2-hop alternate path if one of the links is loaded close to capacity. This least busy alternate routing with trunk reservation has been found to work well in practice in fully connected networks. In fact, Hunt and Laws [5] prove asymptotic optimality of essentially this scheme for direct and 2-link paths in fully connected networks.

Recently, Sibal and DeSimone [12] have proposed a simple and practical alternate routing and trunk reservation strategy for general topologies. In their approach, each node pair is pre-assigned a primary route and a set of alternate routes. If feasible, a call is routed on its primary path. Otherwise, an alternate route is chosen subject to pre-calculated trunk reservation. The trunk reservation parameters are calculated based on the traffic matrix and statistical characteristics of the traffic such as arrival rate and holding times.

The main problem with designing trunk reservation algorithms for general networks is the definition of the "direct path". This is in contrast to the fully connected networks, where the notion of direct vs. alternative path is defined in a natural way. The disadvantage of both shadow-cost and the reservation-based algorithms mentioned above is that they require knowledge of the traffic matrix in advance; this matrix is used to compute the reservation parameters or the shadow costs. Thus, it is not clear how to make these algorithms dynamically exploit the fluctuations in the traffic rate.

Our EXP algorithm can be viewed as having the advantages of both trunk reservation and shadow cost based algorithms. The EXP algorithm routes along the shortest paths, where the costs are *dynamically* assigned based on the *instantaneous* state of the system. The reservation is implemented by rejecting circuits for which sufficiently inexpensive paths were not found. Note that the EXP algorithm does not restrict routing to a prespecified set of paths. Instead, it provides a dynamic decision policy that chooses between all the available paths according to the currently known information on the load, while giving preference to the shorter ones. Two of its main advantages are that it neither requires a priori knowledge of the traffic matrix, nor computation of "direct" vs. "alternative" paths. As we have mentioned above, our simulations indicate that it outperforms the reservation scheme of [12].

In Section 2, we describe the algorithm in [3] and discuss its theoretical performance guarantees. We show how the algorithm can be adapted to work well in practice and provide simulation results for the adapted algorithm. Section 3 shows how to modify our algorithm for a distributed setting and provides simulations results for that setting. Section 4 discusses several implementation issues and offer plans for the future.

2 Centralized On-line Algorithm for SVC Routing

2.1 Exponential-cost based routing

In this section we describe our routing algorithm, which we refer to as the "EXP" algorithm, and analyze its performance in the centralized setting. The EXP algorithm can be viewed as a modified version of the exponential cost based algorithm of [3], which we will refer to as "AAP", and thus we start by describing the AAP algorithm and then show that the EXP algorithm significantly outperforms it under realistic assumptions on the offered traffic.

The network is represented by a capacitated (directed or undirected) graph $G(V, E, u)$. Let n denote the number of nodes in the graph. The capacity $u(e)$ assigned to each edge $e \in E$ represents bandwidth available on this edge.

ROUTE$(s_j, t_j, T_j^s, T_j^f, r_j)$:
$\forall \tau, e \in E: \; c_e(\tau, j) \leftarrow u(e)(\mu^{\lambda_e(\tau,j)} - 1);$ (*)
if $\exists$ path P in $G(V,E)$ from s_j to r_j s.t. the cost of P

$$\sum_{e \in P} \sum_{\tau} \frac{r_j}{u(e)} c_e(\tau, j) \leq nrT$$

then route the requested circuit on P with the minimum cost, and set:
$\forall e \in P, T_j^s \leq \tau \leq T_j^f,$
$\lambda_e(\tau, j+1) \leftarrow \lambda_e(\tau, j) + \frac{r_j}{u(e)}$
else reject the connection

Figure 1: The original AAP routing algorithm

The exponential-cost based algorithm of [3] is shown in Figure 1. Upon receiving request i, represented by $(s_i, t_i, r_i, T_i^s, T_i^f)$, the AAP algorithm tries to allocate a route of capacity r_i from originating node s_i to the destination node t_i starting at time T_i^s and ending at time T_i^f. For simplicity, we assume that the routing is done at time T_i^s. The algorithm either accepts the request, allocating an appropriate route, or rejects the request. The goal of the algorithm is to maximize the total number of routed requests.[1] Let $T_i = T_i^f - T_i^s$ denote the "holding time" of the circuit, T denote the maximum possible T_i, and r denote the maximum request bandwidth (rate) r_i.

The routing decision is based on the current information about the current and future *load* on the edges of the network. The load is measured relative to the edge capacity $u(e)$. Let P_i denote the route used to satisfy the ith request. The load on edge e at time τ as seen by the routing algorithm when routing the kth circuit is defined as follows:

$$\lambda_e(\tau, k) = \sum_{\substack{e \in P_i, i<k \\ \tau \in [T_i^s, T_i^f]}} \frac{r_i}{u(e)}.$$

After computing the load, the next step is to compute the *exponential cost*. The cost of edge e at time τ as seen by the routing algorithm when routing the kth circuit is defined by

$$c_e(\tau, j) = u(e)(\mu^{\lambda_e(\tau,j)} - 1),$$

where μ is a parameter.

The main result in [3] is that choosing $\mu = 2nTr/r_{\min} + 1$ guarantees that for any sequence of requests where for all i we have $r_i \leq \min_e u(e)/\log \mu$, the total number of requests accepted by the AAP algorithm is within $O(\log \mu) = O(\log nTr/r_{\min})$ of the maximum total requests that can be routed by the best *off-line* algorithm, i.e. an algorithm that knows all of the requests in advance. In fact, it is trivial to see that if the maximum allowed hop count is L, then the performance bound is improved to $O(\log LTr/r_{\min})$

[1]It is easy to modify the algorithm to optimize a general "profit" measure, where each routed request bring a predefined profit. For example, such profit can be given by the sum of $\rho_i = r_i T_i$ over all the accepted circuits.

2.2 Adding probabilistic assumptions

In [3], it is shown that the performance achieved by the AAP algorithm is within a constant factor of the best possible performance achievable by any online algorithm, i.e. any online algorithm can be presented with a network and a request sequence such that the number of requests accepted by this algorithm will be below $O(1/\log(LTr/r_{\min})$ fraction of the total requests accepted by the best off-line algorithm. In this section, we show that under realistic assumptions about the input sequence (and without any assumptions about the network) it is possible to get a much better performance.

Intuitively, the performance of the AAP algorithm is low due to the fact that it uses high value of μ. Unfortunately, simply reducing μ causes the AAP algorithm to accept circuits in excess of the available capacity, which is not allowed. In other words, the AAP algorithm is correct only for sufficiently large μ.

A natural approach is to reduce μ and to modify the AAP algorithm by explicitly forbidding the use of saturated edges. More precisely, after the line marked (*) in Figure 1, we compute a *modified cost function* c'

$$(1) \quad \forall \tau, e \in E : \textbf{if } (\lambda_e(\tau, j) + r_j/u(e)) \geq 1 \qquad \textbf{then } c'_e(\tau, j) \leftarrow \infty$$
$$\textbf{else } c'_e(\tau, j) \leftarrow c_e(\tau, j)$$

and then use c' instead of c in the rest of the algorithm.

We will refer to the modified algorithm as the EXP algorithm. Observe that the EXP algorithm produces legal routing decisions for any value of μ. Moreover, it is easy to see that by keeping μ the same as for AAP algorithm, the two algorithms behave identically.

Unfortunately, reducing μ invalidates the proof of the AAP algorithm. In particular, the corner-stone assumption that all the rejected circuits connect distant (in terms of the cost c_e) nodes is no longer valid. In what follows we will show a modified proof that is correct under certain assumptions on the offered traffic.

In order to analyze the performance of the EXP algorithm, consider some set of requests and divide them into sets according to the way they were treated by the algorithm. Let P_i^* denote the path chosen by the optimum off-line algorithm for routing the ith request. The requests R that were rejected by EXP but accepted by the optimum off-line algorithm fall into two types: R_1 consists of requests i where the cost (when routing i) of the path P_i^* with respect to c exceeds LrT and $R_2 = R - R_1$. Observe that $i \in R_2$ implies that the cost of P_i^* does not exceed LrT but P_i^* passes through at least one saturated edge.

Intuitively, one can view the EXP algorithm as if it consists of two parts: one part proposes a low-cost path, and the other part checks whether there is available capacity on this path. Using this description, R_1 corresponds to the requests for which sufficiently low-cost path was not found, and R_2 corresponds to the requests that were rejected because every low-cost path has a saturated edge.

The proof in [3] can be easily adapted to show a bound on $|R_1|$.

Lemma 2.1 ([3]) *Let A be the set of circuits accepted by the EXP algorithm then $|R_1| \leq O(\log \mu)|A|$.*

The main difficulty lies in bounding the value of $|R_2|$. Since the cost is exponential in the load, paths through saturated (or close to saturated) edges will, in general, have high cost. Observe, that unless μ is chosen to be at least $\Omega(LT)$, R_2 is not empty, since the cost of a saturated edge is not high enough to prevent us from trying to use it.

Consider a circuit that was rejected because there was no sufficiently short path with respect to costs c', while there was a short path with respect to costs c. This can happen only if there is an edge on this path that was close to saturation. Let $\{C_e(\delta), 0 \leq \delta \leq T\}$ be a random vector that denotes the costs as computed by the algorithm when trying to route a circuit, where δ corresponds to the number of time steps into the future with respect to the arrival time. We will use $\Lambda_e(\delta)$ to represent the corresponding load on edge e. Observe that, by construction, $\delta > \delta'$ implies that $C_e(\delta) \leq C_e(\delta')$. Let $\tilde{C}_e(\delta)$ be the random vector $C_e(\delta)$ conditioned on the event that $C_e(0) = u(e)(\mu - 1)$, i.e. the event that the arriving circuit sees that e is saturated.

Consider the requests in R_2. As we have mentioned above, request i is in R_2 if and only if the cost of the corresponding P_i^* computed when routing i does not exceed LrT and at the same time, P_i^* contains a saturated edge. Thus, the probability of a request in $R_2 \cup R_1$ ending up in R_2 is bounded by:

$$\mathcal{P} = \max_e \Pr[\sum_{0 \leq \delta < T_i} \tilde{C}_e(\delta) \leq nrT].$$

Let OPT denote the set of requests satisfied by the optimum off-line algorithm.

Lemma 2.2

$$\frac{E[|OPT|]}{E[|A|]} \leq O(\frac{\log \mu}{1 - \mathcal{P}}).$$

Proof: From the definition of $\mathcal{P}$, we have:

$$E[|R_2|] \leq \mathcal{P} \cdot (E[|R_1| + |R_2|])$$

This implies:

$$E[|R_2|] \leq \frac{\mathcal{P}}{1 - \mathcal{P}} E[|R_1|].$$

The claim follows from the fact that $|R_1| \leq O(\log \mu)|A|$ and the fact that $|OPT| \leq |A| + |R_1| + |R_2|$. ∎

When is $\mathcal{P}$ small ? One family of input traffic sequences for which $\mathcal{P}$ is small is when the expected cost of an edge over the duration of T_i of the ith circuit is sufficiently larger than LrT. More precisely, using Chebyshev inequality, we get the following lemma:

Lemma 2.3 *If for all i, we have $E[\sum_{\delta \leq T_i} \tilde{C}_e(\delta) - LrT] \geq 2\sqrt{VAR[\sum_{\delta \leq T_i} \tilde{C}_e(\delta)]}$ then $\mathcal{P}$ is a constant bounded away from 1.*

There is a large general class of cases for which lemma 2.3 gives $O(\log \mu)$ competitive performance for significantly smaller values of μ. For example, consider the following case:

- All circuits have uniform traffic rate $r \leq \min_e \frac{u(e)}{16 \log^2 L}$.
- The conditional probability of a circuit that is alive at time $\delta = 0$ finishing at time $\delta \in [0, T-1]$ is at most $\frac{\alpha}{T}$, where $\alpha > 0$ is a constant, independent of all the other circuits that were alive at $\delta = 0$.

The proof of the following lemma will be included in the full version of the paper.

Lemma 2.4 *Under conditions outlined above, there exists μ, where $\log \mu = O(\log L)$ is independent of T, such that the number of requests accepted by the EXP algorithm $E[|A|]$ is within $O(\log \mu)$ factor of the expected number of requests $E[|OPT|]$ accepted by the optimum off-line algorithm.*

Instead of proving the above lemma, we show analysis of a simple special case of this lemma case that corresponds to the simulations discussed in the next section. Consider the following case: arrival times of routed circuits are Poisson-distributed, r_i and T_i are the same for all i. We will refer to this as the "simplified case" and will show that a significantly smaller $\mu = O(L)$ (independent of T) is sufficient for this case.

Since T_i and r_i are the same for all i, we can simplify notation by assuming that $r_i = 1, T_i = T$, and denoting $\tilde{C}_e = \sum_{0 \leq \delta \leq T} \tilde{C}_e(\delta)$. We also assume $\log^2 L \leq \min_e u(e)/2$.

Lemma 2.5 *The mean and variance of C_e are bounded as follows:*

$$E[C_e] \simeq Tu(e)\left(\frac{\mu^{\frac{u(e)+1}{u(e)}} - 1}{(\mu^{\frac{1}{u(e)}} - 1)(u(e)+1)} - 1\right),$$

$$\sigma^2(C_e) \le T^2 u(e)^2 \left(\frac{\mu^{\frac{2(u(e)+1)}{u(e)}} - 1}{(\mu^{\frac{2}{u(e)}} - 1)(u(e)+1))} - \frac{\mu^{\frac{2u(e)+1}{u(e)}} - 1}{(\mu^{\frac{1}{u(e)}} - 1)(2u(e)+1))} \right).$$

Proof: Since we assume Poisson arrival of circuits, the number of circuits still alive at time $\delta \in [0, T-1]$ given that they were alive at $\delta = 0$ is binomially distributed between 0 and $u(e)$ with probability $(1 - \delta/T)$ for each $\delta \in [0, T-1]$ i.e. for $k = 0, \ldots, u(e)$ and $t \in [0, T-1]$

$$\Pr\left(\Lambda_e(t) = \frac{k}{u(e)}\right) = \binom{u(e)}{k}(1 - \frac{t}{T})^k (\frac{t}{T})^{u(e)-k}.$$

The results for mean and variance stated in the lemma follow from this observation. ∎

Now we can prove the lemma that states that the smaller value of μ is sufficient.

Lemma 2.6 *There exists μ where $\log \mu = O(\log L)$, such that in the "simplified case" defined above, the number of requests accepted by the EXP algorithm $E[|A|]$ is within $O(\log \mu)$ factor of the expected number of requests $E[|OPT|]$ accepted by the optimum off-line algorithm.*

Proof: Immediate from Lemmas 2.5, 2.3, and 2.2.

2.3 Simulation Results

In the previous section we present the theoretical background underlying on-line routing and admission control algorithms based on exponential costs in general, and the EXP algorithm in particular. Since it is impossible to analyze the performance of such algorithms analytically for realistic networks and traffic patters, we conducted extensive simulation experiments. The main goal of our simulations was to compare the EXP algorithm to several algorithms which are based on minimum hop ideas. We have used two different realistic network topologies - An existing commercial topology, and the NSFNet backbone T3 topology (as used in [12]).

2.3.1 An Existing Commercial Topology

Our simulations use an existing commercial network that consists of 25 nodes and 61 links. The topology is pictured in Figure 2. The capacities of the links are all chosen to be 155 Mbps, which corresponds to SONET OC-3 service. The virtual circuits all require 1 Mbps in both directions and have a holding time of 30 minutes. Virtual circuit requests arrive as a Poisson process. The proportion of the aggregate requests that goes between every source destination pair is determined by a traffic matrix that corresponds to the actual current traffic on the network whose topology we are using.

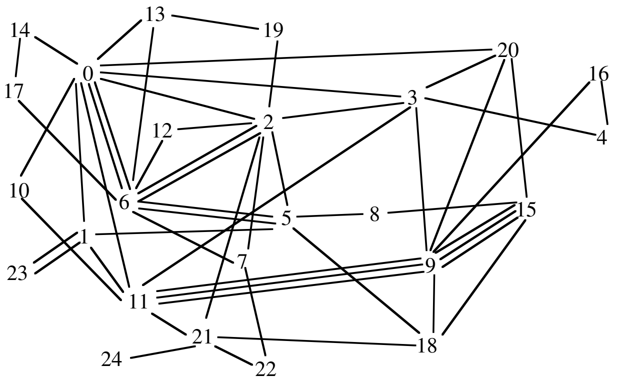

Figure 2: Topology for an existing commercial network.

In order to put the performance advantage of the EXP algorithm over minimum hop based algorithms into perspective we wish to compute the performance of the optimum off-line algorithm. Unfortunately, this computation does not seem to be tractable. Instead, we compute a lower bound on the optimum rejection rate by solving a multi-commodity flow problem in which the objective function is to satisfy the maximum demand between node pairs without violating the capacity constraints, where the demand between node pairs is determined by the traffic matrix. In particular, the demand between nodes i and j is set to the average number of bits per second that are expected to be requested with i and j as the source and destination. It is easy to see that the solution of this optimization problem is indeed a lower bound on the rejection rate. However, this lower bound may be far off from the true optimum since it does not take into account variations in arrival rate due to statistical behaviour. Furthermore, the lower bound can split the traffic from a virtual circuit over multiple paths. Unfortunately, it is not clear how far off our lower bound is nor how to compute a better approximation to the optimum rejection rate.

Figure 3 compares the performance of the EXP algorithm with $\mu = 2.4$ to the performance of a simple minimum-hop algorithm and our lower bound on the performance of the optimum algorithm[2]. The X-axis gives the aggregate arrival rate in virtual circuit per second and the Y-axis gives the percentage virtual circuits rejected for that aggregate arrival rate. It can be seen that the EXP algorithm has a significant performance advantage over the minimum-hop algorithm for a wide range of arrival rates. The EXP algorithm can maintain a much higher arrival rate given a target rejection (loss) rate. For a target maximum rejection rate of 2%, the EXP algorithm can sustain an arrival rate that is 9.2% higher than the arrival rate that can be sustained by the minimum-hop algorithm. The relative performance advantage of the EXP algorithm increases with arrival rate. For example, the improvement for target maximum rejection rate of 20% is 17%.

[2]The graphs do not include confidence intervals. Our results are very consistent over different seeds so we expect the intervals to be tight. We will include the intervals in a published version of the paper

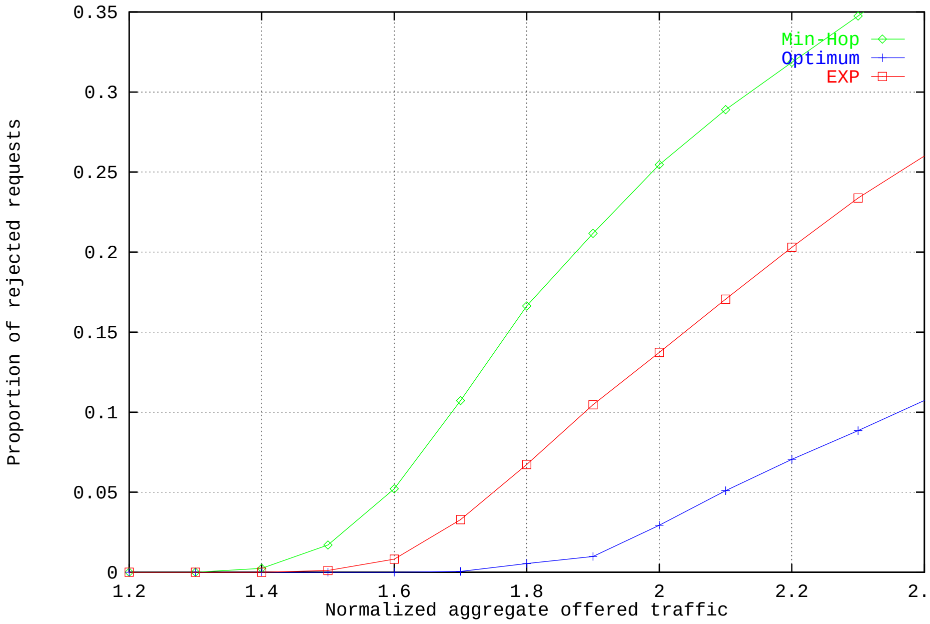

Figure 3: Simulation results for the centralized algorithms

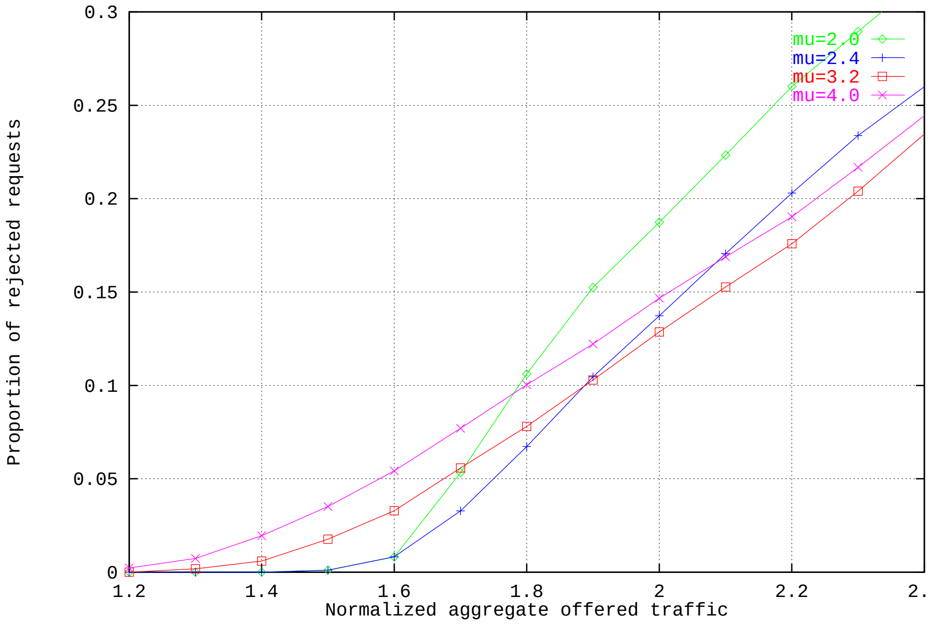

Figure 4: Determining μ

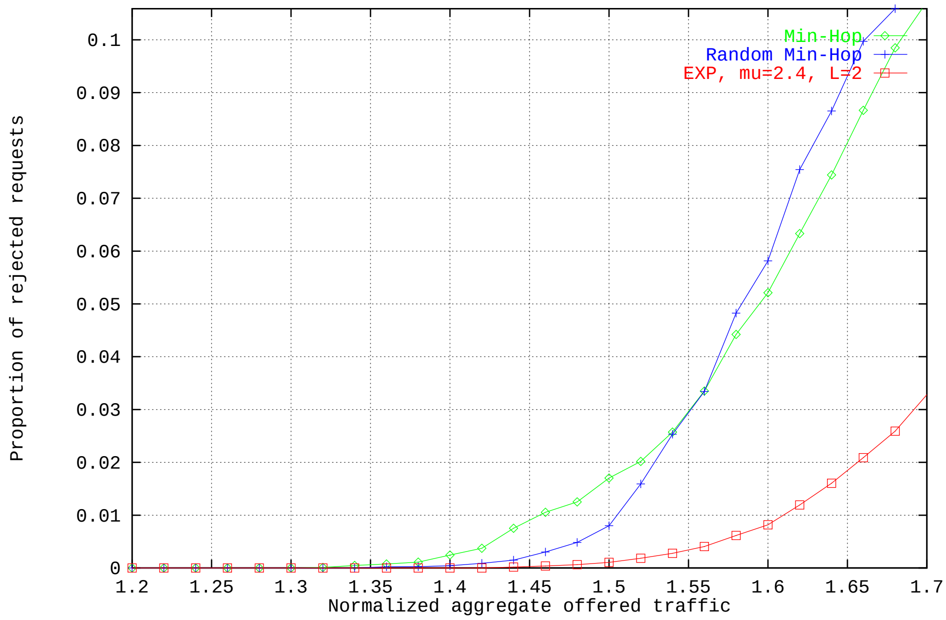

Figure 5: Comparing EXP with randomized minimum-hop

The main parameter in the EXP algorithm is the base μ used to compute the exponential costs. Figure 4 explores the effect of various values of μ on the performance. Essentially, a larger value of μ implies a more conservative admission control policy in the sense that a virtual circuit requiring a long path is less likely to be accepted when EXP uses a higher μ. Not surprisingly, EXP with a higher μ value performs better than EXP with a lower μ value at high arrival rates and worse at low arrival rates. In our simulations, the average path length is approximately two. Thus, the value for μ calculated by the analysis for "simplified case" in Section 2 is ≈ 4.5. Based on the results in Figure 4, this μ value is clearly too high. The reason that the analysis for the "simplified case" in Section 2 choose a μ that is too large is that it attempts to be competitive with the optimal algorithm for *all* arrival rates. However, in practice we are only interested in arrival rates that lead to relatively low rejection rates. Therefore, the value of $\mu = 2.4$, which works well for arrival rates that lead to low rejection rates, is not directly predicted by the "simplified case" in Section 2. It is interesting to note that the original AAP algorithm suggests μ in the thousands for the network considered in our simulations. It is clear that this would not lead to acceptable performance. (In fact, we have simulated the algorithm with such μ and found that it works best for arrival rates that cause rejection rates that exceed 70

A natural variation of the minimum-hop algorithm is the *randomized* minimum-hop algorithm, where we choose one of the available minimum-hop paths uniformly at random. Not surprisingly, this algorithm works better than the simple minimum-hop algorithm for small

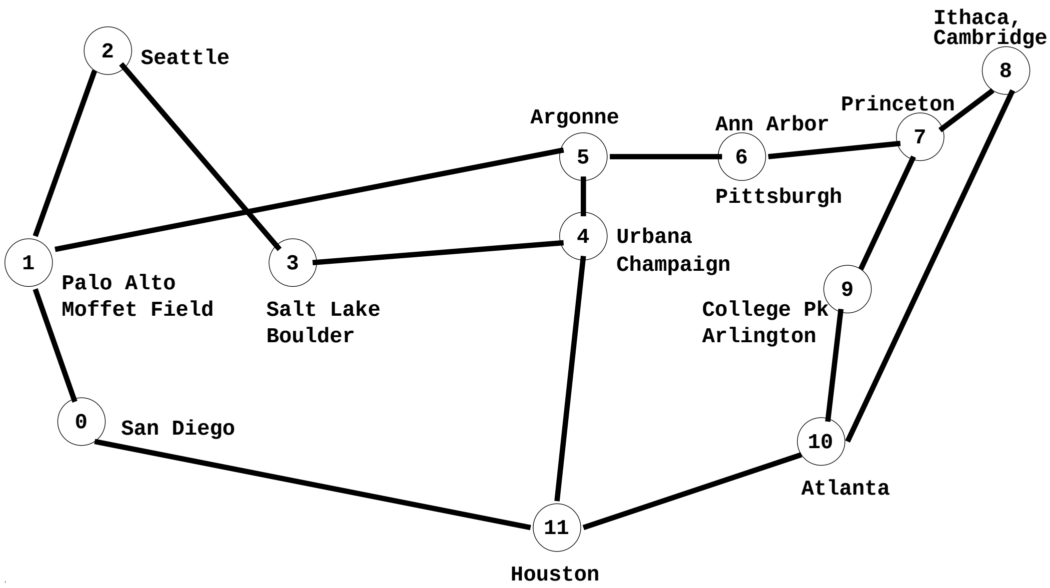

Figure 6: Topology for NSFNet T3 Backbone.

loads. However, the randomized minimum-hop algorithm is still significantly outperformed by the EXP algorithm. This can be observed in Figure 5. Note that this figure covers only a small portion of the range of the aggregate offered traffic shown in Figure 3.

2.3.2 NSFNet T3 Topology

In this section we describe results of simulating our algorithms on the NSFNet T3 backbone network. The definition of the network and of the traffic matrix was taken from the paper by Sibal and DeSimone [12]. For completeness, we give the topology and the traffic profile here as well.

The topology of NSFNet is depicted in Figure 6. This NSFNet configuration was extant in 1992. Each edge in the topology represents a 100Mbits bidirectional link. All arrivals were assumed to be medium picture quality video calls requiring 1Mb/sec.

The traffic matrix is shown in 1. Each entry denotes the traffic in Erlang. For our experiments, we generated exponentially distributed holding times with a mean of 100 seconds. This, together with the traffic profile in 1 defines the call arrival rate.

The simulation results are presented in Figure 7. The figure shows comparative performance of the EXP, min-hop, and the reservation-based scheme used by Sibal and DeSimone (marked [SD] in the figure) on the internet topology. The X-axis corresponds to aggregate normalized traffic and Y-axis corresponds to the fraction of rejected circuits. The results for "controlled

$$
\begin{pmatrix}
0 & 4 & 2 & 3 & 6 & 0 & 7 & 1 & 9 & 5 & 2 & 3 \\
4 & 0 & 3 & 6 & 13 & 0 & 15 & 2 & 19 & 9 & 4 & 6 \\
2 & 3 & 0 & 2 & 5 & 0 & 6 & 1 & 7 & 3 & 1 & 2 \\
3 & 6 & 2 & 0 & 8 & 0 & 10 & 1 & 12 & 6 & 2 & 4 \\
7 & 14 & 5 & 9 & 0 & 0 & 24 & 3 & 31 & 15 & 6 & 9 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
8 & 17 & 6 & 11 & 25 & 0 & 0 & 4 & 37 & 18 & 7 & 11 \\
1 & 2 & 1 & 1 & 3 & 0 & 3 & 0 & 4 & 2 & 1 & 1 \\
11 & 22 & 9 & 15 & 33 & 0 & 39 & 5 & 0 & 24 & 9 & 15 \\
5 & 9 & 4 & 6 & 14 & 0 & 16 & 2 & 21 & 0 & 4 & 6 \\
2 & 4 & 1 & 2 & 5 & 0 & 6 & 1 & 8 & 4 & 0 & 2 \\
3 & 6 & 2 & 4 & 8 & 0 & 10 & 1 & 12 & 6 & 2 & 0
\end{pmatrix}
$$

Table 1: NSFNet Traffic Profile

alternate routing" were copied from [12]. The figure show that the "controlled alternate routing scheme" proposed in [12] significantly outperforms the min-hop algorithm, while the EXP algorithm outperforms this scheme for rejection rates above 2%. For example, if the cutoff point is taken as 5% rejection rate, the EXP algorithm with $\mu = 3.0$ can sustain about 11% higher aggregate offered load as compared to the SD algorithm.

As for the Frame-Relay topology discussed in the previous section, the EXP algorithm behaves differently for different values of parameter μ. In general, higher μ produce better results for heavy traffic conditions, while low values of μ are best for lightly loaded networks. Figure 7 shows behavior of the EXP algorithm for two values of $\mu = 2.4$ and $\mu = 3.0$. It can be seen that $\mu = 3$ becomes a better alternative starting from about 3% rejection rate. An interesting open problem is how to choose μ depending on the traffic conditions. In particular, whether it is possible to change μ dynamically. We are currently investigating these questions.

3 Distributed Algorithm

Large scale networks cannot expect to have accurate state information available at every source node. As a consequence, such a large scale network must either make all routing decisions at a centralized site, or must make routing decisions locally based on static or inaccurate state information. In this section, we describe how the EXP routing scheme of Section 2 can be effectively adapted to a setting where only inaccurate state information is available.

Our focus on local routing decisions rather than on routing decisions at a centralized site is motivated by several systems considerations. In particular, a centralized solution is more

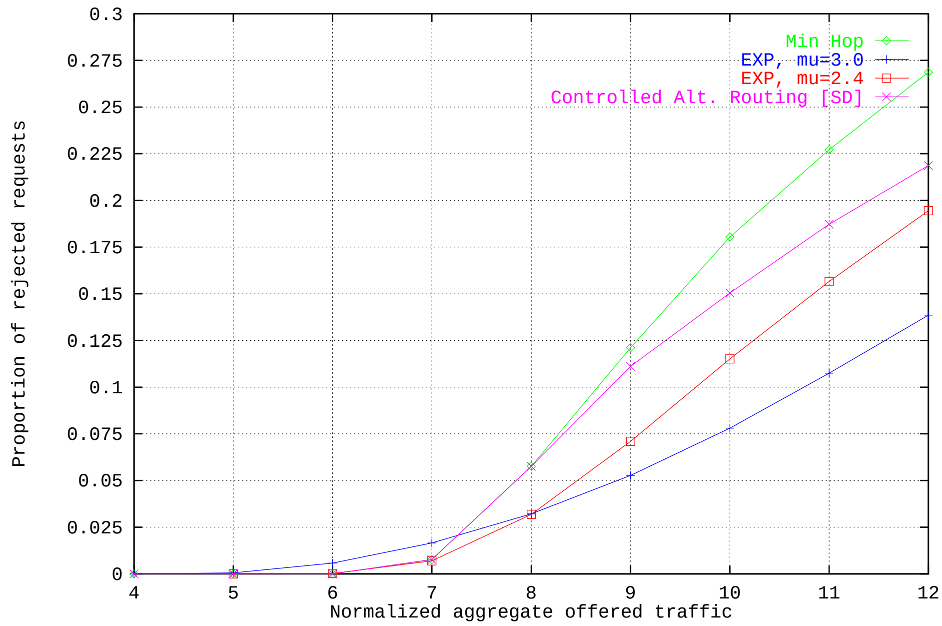

Figure 7: Comparative performance for Internet topology

likely to suffer from reliability problems since it has a single point of failure and cannot function properly in the presence of network partitions. Furthermore, fast virtual circuit setup is not possible since round trip communication with a potentially distant central node must precede the establishment of every circuit.

A key factor in distributed algorithms is the degree of inaccuracy in local state information. This is largely a function of the system constraints. In order to ensure the applicability of our work to a broad range of system environments, we have adopted a very conservative system model that, for example, does not include any type of periodic state information exchange. Specifically, we permit a node to collect state information only for the links along the path on which it attempts to establish a virtual circuit. Thus, the state information gathering task can be carried by the control messages used during a virtual circuit setup. The overhead of this scheme is clearly very low.

One of the motivations for making local routing decisions is the need for fast virtual circuit setup. In discussions with several switch manufacturers we have learned that it is not uncommon for requirements on setup times to be of the order of several milliseconds. In a national network, setup times of that order of magnitude can only be achieved by pipelining the data packets immediately behind the setup packet. In such environments, multiple setup attempts may not be possible. Fortunately, our algorithm, in contrast to a minimum hop based algorithm, achieves good performance even if it is restricted to a single setup attempt.

To convert the centralized EXP algorithm described in Section 2 to a distributed setting, we proceed as follows. Each source maintains a local view of the network state. That view is constructed from the state of the incident links and from information about other links gathered during circuit setup. When a virtual circuit request arrives, the EXP algorithm is invoked. It either rejects the request outright, or it returns a path, in which case we attempt to use this path to route the circuit.

If the path has sufficient capacity to accommodate the virtual circuit, the circuit is routed[3] The acknowledgement packet that informs the source node whether or not the circuit was indeed routed collects the current state information of each link along the circuit's path. That state information is used to update the local view of the network at the source node. If the acknowledgement packet indicated that the circuit was not routed due to insufficient available capacity, the algorithm tries to route the circuit again as if it was a new request.

We distinguish the distributed algorithms according to the number of allowed tries. The next section presents results for a single try and 3-tries variants. Note that the state information returned by the acknowledgement packet(s) may involve many bits since it includes information about the future state of the links. In Section 4 we show how to reduce the number of required bits to encode the state information in the acknowledgement packet(s).

3.1 Simulation Results

The results of our simulations for the existing commercial topology are summarized in Figure 8. This figure compares the performance of several variants of the EXP algorithm (centralized, distributed with 1 attempt, distributed with 3 attempts) with several variants of min-hop algorithm (centralized and distributed with 1 or 3 attempts). The conversion of centralized min-hop into a distributed one was done exactly as for the EXP algorithm. The reason we have chosen to present the "3 attempts" case is that our simulations indicated that increasing the number of attempts beyond three does not increase the performance.

The simulations show that the performance advantage of the EXP algorithm increases in the distributed setting. When comparing the distributed exponential strategies to the distributed minimum hop strategies we observe a significant performance advantage for the exponential strategies over a wide range of arrival rates. It seems that the performance penalty suffered due to the inaccurate state information is much less pronounced for the variants of the EXP algorithm. The EXP-based algorithms can sustain a much higher arrival rate given a target rejection (loss) rate. For a target rejection rate of 2% and a limit of three setup attempts, the distributed EXP algorithm can maintain an arrival rate that is 8% higher than the arrival rate that the minimum hop based algorithm can maintain. When we restrict the algorithms to a single attempt, the advantage of the distributed EXP algorithm increases to 38%.

[3] Issues arising from multiple circuits concurrently sending setup packets are not modeled by our simulations. We will address this issue in the full version of the paper.

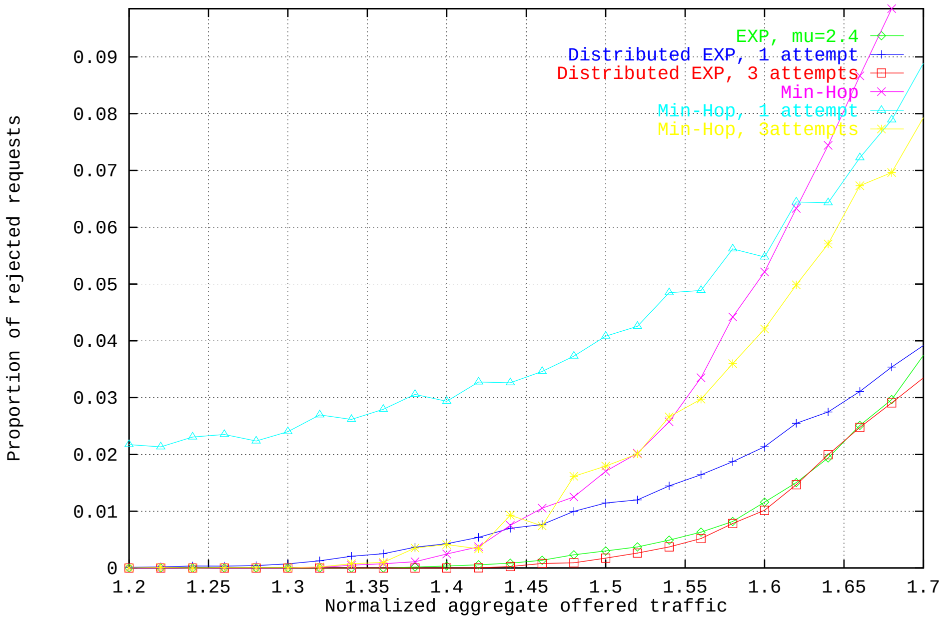

Figure 8: Performance of distributed vs centralized schemes.

The sensitivity of the minimum hop based algorithms to the number of permitted attempts is particularly dramatic. The simulations show that a minimum hop based strategy would not be appropriate in a network requiring fast setup. In contrast, our exponential function based strategy still achieves good performance when fast setup, and consequently at most one setup attempt, is required. This is due to the fact that the exponential algorithm tends to pick paths that use edges with lower perceived load, thus reducing the probability of picking a saturated edge.

Another interesting phenomenon is the fact that the distributed minimum hop algorithm performs better than the centralized minimum hop algorithm at high arrival rates. The phenomenon provides a good illustration of the importance of trunk reservation. Since the centralized minimum hop algorithm is greedy, it will attempt to route a virtual circuit even if the only available path is very long. However, routing a virtual circuit on a long path will lead to inefficiencies at high arrival rates since it is likely that two or more future virtual circuit requests, each requiring shorter paths, can be routed on the long path. Since the distributed minimum hop algorithms only make a few routing attempts, they are unlikely to try long paths. Thus they effectively are implementing a simple trunk reservation policy. Unfortunately, this simple trunk reservation policy is too aggressive for low arrival rates. In particular, an aggressive trunk reservation policy will lead to inefficiencies at low arrival rates since future, shorter (in terms of hops) virtual circuit requests are less likely to arrive to replace a long virtual circuit request that was rejected due to trunk reservation. Thus, the greedy policy of the centralized minimum hop algorithm is superior to distributed single attempt minimum hop algorithm at low arrival rates.

One can view our distributed EXP algorithm as implementing a trunk reservation policy that is adjusted with the arrival rate. At low arrival rates, the link cost is low and so the admission control policy is greedy. When the arrival rate is very high, the link cost is high, thus implementing an aggressive trunk reservation policy. By dynamically adjusting its trunk reservation policy with the arrival rate, the EXP algorithm is able to outperform both minimum hop based algorithms at high and low arrival rates.

4 Discussion

There are several implementation issues that are not completely addressed by our simulations. This section suggests some approaches to these issues. In future work we hope to explore our approaches in greater detail.

Both the centralized and the distributed EXP algorithms assume that the duration of the virtual circuit is known in advance. While this is a reasonable assumption for many applications, e.g. movies, there are a large class of applications, e.g. phone calls, for which this is not a reasonable assumption. However, a statistical description of the duration is available for many applications for which the duration is not known in advance. Again, phone calls are a good example. We envision using the statistical duration information about the virtual circuits to make routing and admission control decisions. In fact, it was shown that it is possible to modify the EXP algorithm to give theoretically good competitive ratio for the case where statistical information on the duration is known [6]. If the majority of virtual circuits either have holding times that are known in advance or belong to a class of connections for which statistics on the holding times are known, we expect to see performance similar to that presented in this paper.

Recall that our distributed EXP algorithm collects the entire state of each link on the path on which a circuit setup is attempted. Since our algorithm makes use of circuit duration information, the link state includes not only the currently allocated load, but also the load allocated by the current virtual circuits for all future points in time. In general, a large number of bits will be needed to transmit this state information. The following mechanism can be used to decrease the amount of transmitted state information to approximately 60 bits per link. For each edge, we transmit the current load and the times at which the load allocated by the current circuits reaches 95%, 85%, 65%, and 25% of the current load. Notice that the sample points are spaced so as to provide the greatest amount of detail at high load levels. Preliminary simulations of this approach suggest that such a decrease in accuracy of state information will not have a dramatic impact on the performance of our algorithm. With this approach, link information for paths with up to 9 links can be accommodated in a single ATM cell.

Acknowledgments

We would like to thank Debasis Mitra for his insight and support. We would also like to thank Flavio Bonomi, Bala Rajagopalan, and Vikram Saksena for helpful discussions.

References

[1] G.R. Ash, J.Chen, A.E. Frey, B.D. Huang, C.Lee, and G.L. McDonald. Real-time network routing in the AT&T network-improved service quality at lower cost. In *Proceedings of the IEEE Global Telecommunicationsi Conference, Orlando, Florida*, pages 802–813, Dec. 1992.

[2] J. Aspnes, Y. Azar, A. Fiat, S. Plotkin, and O. Waarts. On-line machine scheduling with applications to load balancing and virtual circuit routing. In *Proc. 25th Annual ACM Symposium on Theory of Computing*, pages 623–631, May 1993.

[3] B. Awerbuch, Y. Azar, and S. Plotkin. Throughput competitive on-line routing. In *Proc. 34th IEEE Annual Symposium on Foundations of Computer Science*, pages 32–40, November 1993.

[4] R.J. Gibbons and F.P. Kelly. Dynamic routing in fully connected networks. *IMA journal of Mathematical Control and Information*, 7:77–111, 1990.

[5] P.J. Hunt and C.J. Laws. Asymptotically optimal loss network control. *Mathematics of Operations Research*, 18(4):880–900, 1993.

[6] A. Kamath, S. Plotkin, and K.G. Ramakrishnan. Online routing of atm circuits with known distribution of holding times. Unpublished manuscript, May 1994.

[7] F.P. Kelly. Routing in circuit-switched networks: optimization, shadow prices and decentralization. *Advances in Applied Probability*, 20:112–144, 1988.

[8] D. Mitra, R. Gibbens, and B. Huang. State-dependent routing on symmetric loss netowkrs with trunk reservations – I. *IEEE Trans. on Communications*, 41(2), February 1993.

[9] D. Mitra and R.J. Gibbons. State Ddependent Routing on Symmetric Loss Networks with Trunk Reservations ii: Asymptotics, Optimal Design. *Annals of Operations Research*, 35:3–30, 1992.

[10] D. Mitra and J.B. Seery. Comparative evaluations of randomized and dynamic routing strategies for circuit switched networks. *IEEE Trans. on Communications*, 39(1):102–116, jan 1991.

[11] T. Ott and K. Krishnan. State dependent routing of telephone traffic and the use of separable routing schemes. In *Proceedings of the 11th international teletraffic congress, Kyoto, Japan*, 1985.

[12] S. Sibal and A. DeSimone. Controlling alternate routing in general-mesh packet flow networks. In *Proceedings of SIGCOMM 94, London, England*, 1994.