Competitive Routing of Virtual Circuits in ATM networks

Serge Plotkin[*]
Stanford University

Abstract

Classical routing and admission control strategies achieve provably good performance by relying on an assumption that the virtual circuits arrival pattern can be described by some a priori known probabilistic model. Recently a new online routing framework, based on the notion of *competitive analysis*, was proposed. This framework is geared towards design of strategies that have provably good performance even in the case where there are *no statistical assumptions* on the arrival pattern and parameters of the virtual circuits. The online strategies motivated by this framework are quite different from the min-hop and reservation-based strategies. This paper surveys the online routing framework, the proposed routing and admission control strategies, and discusses some of the implementation issues.

[*] Research supported by NSF CCR-9304971, ARO DAAH04-95-1-0121, and by Terman Fellowship.
E-Mail: plotkin@cs.stanford.edu, URL: http://theory.stanford.edu/people/plotkin/plotkin.html.

1 Introduction

Future Broadband Integrated Services Digital Networks (B-ISDN) will carry a wide spectrum of new consumer services, such as video-on-demand, video teleconferencing, etc. A key characteristic of these services is that they require quality-of-service (QoS) guarantees. Assuring QoS requires reservation of resources. As a result, B-ISDN will likely allocate resources in terms of virtual circuits (or virutal paths). In particular, creating a virtual circuit will require reservation of bandwidth on some path between the endpoints of the connection. Since network resources are limited, some requests for establishment of virtual circuits will be denied.

Thus, there is a need for an *admission control* and *routing* strategy that addresses the following two questions:

- What path should be used to route a given circuit ?
- Which circuits should be routed and which ones should be rejected ?

The admission control and routing strategy has to take into account the limitations of the underlying network. In particular, due to the high bandwidth-dely product, each circuit has to be routed on a single path. Moreover, rerouting circuits is either heavily discouraged or outright forbidden. For some applications it is more efficient to use *multicast* circuits, where instead of a single destination there are multiple destinations. Examples include teleconferencing, video on demand, database updates, etc. In this case the routing strategy has to choose a *tree* that spans the nodes participating in the multicast.

The problem of bandwidth reservation and management in the context of circuit-switched networks has been studied extensively. Until recently, most of the analysis and design effort concentrated on achieving provably good performance under the assumption that the arrival pattern of virtual circuit requests can be described by a simple probabilistic model with known parameters. This approach, starting from the work of Kelly [26], lead to the development of many practical routing strategies. Examples include the Dynamic Non-Hierarchical Routing (DNHR) [2, 4, 1] and Real Time Network Routing (RTNR) [3] strategies developed in AT&T, as well as Dynamically Controlled Routing (DCR) developed at Bell Northern Research [16, 17, 24]. Dynamic state-dependent routing strategies based on reservation were thoroughly analyzed by Mitra et. al. in [34, 35].

Recently, a new framework was proposed that leads to routing strategies that can achieve provably good performance without relying on any assumptions about the probabilistic behavior of the traffic [20, 5, 19, 8, 9]. The first step in developing such strategies was to develop an appropriate way to compare these strategies one to another. The difficulty lies in the fact that traditionally, the performance of routing strategies is measured as a function of the parameters characterizing the input traffic; this approach is not appropriate when there is no a priori knowledge of these parameters. Thus, it was necessary to develop a measure that will allow us to compare different strategies and that is meaningful independent of whether or not there are assumptions about the input traffic.

The idea, proposed in [20, 5], is to use the concept of the *competitive ratio*. This concept was introduced in [39] and further developed in [25, 15, 33]. (Some of the earlier papers on approximation algorithms, e.g. [40], can be viewed as optimizing the competitive ratio as well.) A competitive ratio is defined with respect to two algorithms; both algorithms have to deal with the incoming requests, which in our case

are requests for establishing virtual circuits. One is the *online* algorithm whose performance we want to measure. The online algorithm has to deal with the requests one-by-one and can not use any knowledge of the future requests. The other is the best possible (in terms of performance) *offline* algorithm; this algorithm is omniscient in the sense that it is assumed to have a *complete a priori knowledge* of the entire request sequence, including future requests. Informally, the competitive ratio measures how much the online algorithm loses in performance as a result of not making optimum decisions. In particular, it measures how much is lost due to lack of knowledge of future requests.

Formally, the competitive ratio of a given online algorithm is defined as the supremum over all input sequences of the performance achieved by the optimum offline algorithm and the performance achieved by this online algorithm. Roughly speaking, the statement that a particular strategy has competitive ratio α means that its performance is at most a factor of $1/\alpha$ of the performance of the best possible offline algorithm.

Clearly, the "best" strategy with respect to the competitive ratio depends on the parameter chosen to measure the performance. Recently, two measures were proposed for computing the competitive ratio of the online routing and admission control algorithms – the *throughput* and the *congestion* measure. In the first, we measure the proportion of the routed (accepted) circuits. A variation on this measure is the total routed bandwidth-duration product, i.e. total throughput. In the second, we measure the maximum link congestion, i.e. maximum ratio of the allocated bandwidth on a link to its total bandwidth. Clearly, to use this measure we have to disallow rejections. Consequently, disallowing rejections implies that the online algorithm should be allowed to use more than 100% of the edge capacity.

A competitive strategy for the *congestion-minimization* model was developed by Aspnes, Azar, Fiat, Plotkin, and Waarts in [5]. This strategy achieves a competitive ratio of $O(\log n)$ for permanent (i.e. infinite holding time) virtual circuits (PVCs), where n is the number of nodes in the network. It was extended to the case of finite holding time circuits (SVCs) where the holding time of a circuit becomes known upon its arrival in [13]. For this case, the strategy achieves $O(\log nT)$ competitive ratio, where T is the maximum holding time.

Competitive strategies for the *throughput-maximization* model were given by Garay and Gopal for the case of a single link [20], by Garay, Gopal, Kutten, Mansour and Yung [19] for a line network, and by Awerbuch, Azar, and Plotkin [8] for general network topologies. In particular, if the bandwidth requested by a single circuit never exceeds $O(1/\log L)$ fraction of link capacity, the strategy in [8] achieves a competitive ratio of $O(\log L)$ for PVC routing, where L is the maximum number of hops taken by a virtual circuit. In the SVC case where the duration of a circuit becomes known upon its arrival and the requested bandwidth never exceeds $O(1/\log LT)$ fraction, this strategy achieves $O(\log LT)$ ratio. It was shown that polylogarithmic competitive ratio is not achievable if the bandwidth of a single circuit exceeds $O(1/\log L)$ fraction of single link bandwidth for the PVC case and $O(1/\log LT)$ fraction for the SVC case [8].

Roughly speaking, the main idea behind the new competitive strategies is to assign each link a "length" that is a highly nonlinear (e.g. exponential) function of its current congestion. The new circuit is routed along the *shortest path* with respect to this length. In the throughput-maximization model, the circuit is rejected if the length of the shortest path exceeds some predefined threshold that depends on the profit (e.g., bandwidth-duration product) associated with this circuit.

The intuition of routing along shortest paths with respect to a length function that is exponential in the

congestion is based on viewing the routing problem as an instance of *multicommodity flow problem*. (See [14] for a thorough treatment of this relationship.) The multicommodity flow problem involves simultaneously shipping of several different commodities from their respective sources to their sinks in a single network so that the total amount of flow going through each edge is no more than its capacity. Associated with each commodity is a demand, which is the amount of that commodity that we wish to ship. In the routing context, commodities correspond to virtual circuits, demands correspond to the requested bandwidth, and capacity corresponds to the link bandwidth.

Roughly speaking, the multicommodity flow solution is optimal if all flows are routed along shortest paths, where the length is defined to be an appropriate function of the link congestion. Recently, several multicommodity flow algorithms that are based on repeated rerouting of flow onto shortest paths were developed [37, 28, 31]. By defining the length function to be exponential in the current congestion, these algorithms produce an approximate multicommodity flow in polynomial time.

It is important to note that although these (offline) multicommodity flow algorithms provide the intuition behind the idea of routing along shortest paths with respect to an exponential function of congestion, they can not be used directly to construct dynamic online routing strategies. The main reason is that these algorithms rely on repetitive rerouting. Moreover, they work only if we have a complete knowledge of the entire request sequence.

While the $O(\log LT)$ competitive ratio does not seem sufficiently small from a practical point of view, it is important to observe that this ratio is achieved without any a priori knowledge about the input. The guarantee on competitive ratio can be greatly improved if we can introduce some probabilistic assumptions. The resulting strategies work well if the assumptions are satisfied, and do not "break down" if the assumptions turn out to be wrong.

We discuss several possible classes of assumptions in Section 4. For example, to achieve an $O(\log LT)$ competitive ratio in the throughput-maximization case, it is sufficient to know the distribution on the holding time of each arriving circuit rather than having the specific holding time. In particular, if the holding times are exponentially distributed, then it is possible to further improve the competitive ratio to $O(\log L \log\log L)$ [22]. This bound does not rely on the frequently made assumption that the arrival patterns to different links are independent. It is interesting to note that there can be no strategy (deterministic or randomized) that achieves a polylogarithmic competitive ratio in case there is no a priori information (and hence no probabilistic assumptions) about the holding times [12, 32].

Several practical routing and admission control strategies derived from the competitive strategies were studied in [21] for the PVC case and in [22] for the SVC case. Simulations show that these strategies outperform the minimum-hop and reservation-based strategies for sparse networks. Reservation based strategies are usually based on the notion of "primary paths" that are preferred paths for routing; other, "alternate" paths, are used only if the primary paths are totally congested. Selection of primary paths is a non-trivial problem and usually has to involve knowledge of the traffic matrix for general topologies. Since the strategies based on the competitive analysis do not rely on the notion of primary vs. alternate traffic, they seem more suitable for sparse networks with non-symmetric traffic matrices.

Section 2 introduces the general framework. Section 3 presents competitive strategies for PVC routing in the throughput-maximization and congestion-minimization models and Section 4 addresses the issues involved in extending these strategies to the SVC case. Section 5 considers the problem of routing multicast

circuits. Section 6 discusses some results of applying these theoretical algorithms to realistic networks.

2 Model and Definitions

The network is represented as a capacitated (directed or undirected) graph $G(V, E, u)$ with n nodes and m edges, where $u(e)$ represents the capacity of the edge $e \in E$. The input sequence is a collection of k requests: $\beta_1, \beta_2, ..., \beta_k$, where each request β_i is described by the tuple:

$$\beta_i = (s_i, t_i, r(i,\tau), T^s(i), T^f(i), \rho(i))$$

Nodes s_i and t_i are the source and destination of the request, $r(i,\tau)$ is the bandwidth required by β_i as a function of time; $T^s(i)$ and $T^f(i)$ are the start and finish times for the request. Let $T(i) = T^f(i) - T^s(i)$ denote the *holding time* (i.e. duration) of the requested circuit, and let T denote the maximum possible value of $T(i)$. We use $r(i)$ instead of $r(i,\tau)$ if the requested bandwidth does not depend on τ. $\rho(i)$ denotes the "profit" that we get if the request is routed (accepted). For example, $\rho(i)$ might be defined to be proportional to the bandwidth-duration product $r(i)T(i)$.

For convenience $r(i,\tau)$ is defined to be 0 for $\tau \notin [T^s(i), T^f(i)]$ and also if i is rejected. In most cases, the requested bandwidth $r(i,\tau)$ remains fixed over the duration of the circuit. The *relative load* at time τ on an edge e just before considering the jth request is defined by

$$\lambda_e(j,\tau) = \sum_{e \in P_i, i<j} \frac{r(i,\tau)}{u(e)}$$

where P_i is the path along which the ith connection is routed. Let $\lambda(j) = \max_{e \in E, \tau} \lambda_e(j,\tau)$. Similarly, define $\lambda^*_e(j,\tau)$ and $\lambda^*(j)$ to be the corresponding quantities for the routes produced by the offline algorithm. For simplicity we will abbreviate $\lambda(k)$ as λ and $\lambda^*(k)$ as λ^*, where k is the index of the last request.

In this paper we concentrate on two related models. In the *congestion-minimization* model, the routing strategy is required to accept all of the requests. The goal is to minimize the maximum congestion λ. The competitive ratio used to measure the performance of strategies in this model is defined as the supremum over all request sequences of λ/λ^*, where λ^* is the congestion produced the the optimum offline algorithm.

In the *throughput-maximization* model, the routing strategy is allowed to reject some of the requests. The goal in this case is to maximize the total profit $\sum_i \rho(i)$ associated with the accepted requests. The competitive ratio in this model is the supremum over all possible request sequences σ of the ratio between the profit of the optimum offline algorithm on σ and the profit of the online algorithm on the same σ. Note that the optimization goal depends on the exact definition of the profit. For example, if the profit is constant per request, then the goal is to maximize the total number of satisfied requests. If the profit is proportional to the bandwidth-duration product, then the goal is to maximize the total routed bandwidth.

3 Routing Permanent Virtual Circuits

3.1 Competitive PVC Routing in Throughput-Maximization Model

In this section we describe a competitive routing and admission control strategy for the throughput-maximization model [8]. We start by identifying some additional restrictions that have to be imposed on the type of traffic, and then present a simplified strategy for routing PVC circuits. The discussion of the more general routing and admission control strategy for the SVC case is deferred to Section 4.

Assumption and Restrictions Let L be the maximum number of hops allowed for a single circuit. First, we assume that the profit is scaled such that the following condition is satisfied.

$$1 \leq \frac{1}{L} \cdot \frac{\rho(i)}{r(i)} \leq F \tag{1}$$

Note that it is always possible to satisfy the above condition if we choose large enough F. For example, if the profit is defined to be proportional to the bandwidth, than we can take $\rho(j) = Lr(j)$ and $F = 1$.

We assume that the rates $r(i)$ of the requests are small compared to the link capacities. More precisely, we assume that for every request

$$r(i) \leq \frac{\min_e\{u(e)\}}{\log(2LF+1)}. \tag{2}$$

It was shown in [8] that such restriction is necessary. More precisely, they showed that if we allow requests with rates $r(j)$ larger than $\min\{u(e)\}/k$, then there can be no online strategy that achieves a competitive ratio smaller than $\Omega(L^{1/k} + F^{1/k})$. In other words, without assumption (2) it is impossible to design a strategy that achieves a polylogarithmic competitive ratio in the general case. If we restrict the underlying network topology to a tree or a hypercube, it is possible to achieve logarithmic competitive ratio without the above assumption [11, 10].

PVC Routing Algorithm In the PVC case, each arriving circuit has infinite holding time. The idea is to assign each link a *length* that is *exponential* in the current congestion of this link. More precisely, define the base $\mu = 2LF + 1$. (This choice will become clear after the proof of Lemma 3.3.) The *length* of an edge e just before job j is considered, is defined by:

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

The online routing strategy, proposed in [8], is as follows:

- When request j arrives, check if there exists a path P in the graph from s_j to t_j satisfying the following condition.

$$\sum_{e \in P} \frac{r(j)}{u(e)} c_e(j) \leq \rho(j)$$

- If such a path P exists, accept the connection and use P to route the connection; otherwise reject the request.

The proof of competitiveness of the above strategy is based on three lemmas. Lemma 3.1 shows that the sum of the lengths of all the edges at any point in time is within $O(\log \mu)$ factor of the already accrued profit. Lemma 3.2 shows that the total profit of circuits that were rejected by the online algorithm but accepted by the offline algorithm is bounded by the total length. Finally, Lemma 3.3 shows that appropriate choice of μ implies that the routing strategy never violates capacity constraints. In other words, if the algorithm decides to route a circuit, then there is always sufficient available capacity.

Lemma 3.1 Let $\mathcal{A}$ be the set of accepted requests and k be the index of the last connection. Then

$$2 \log \mu \sum_{j \in \mathcal{A}} \rho(j) \geq \sum_{e} c_e(k+1)$$

Proof: By induction on k, the number of arrived requests. For $k = 0$, both sides are 0. Rejected connections do not matter since they neither affect the length nor the profit. Therefore we are done if we show the following for an *accepted* connection β_j.

$$\sum_{e} \left(c_e(j+1) - c_e(j)\right) \leq 2\rho(j) \log \mu$$

Consider an edge $e \in P_j$. From the definition of the length we have

$$\begin{aligned} c_e(j+1) - c_e(j) &= u(e)\left(\mu^{\lambda_e(j) + \frac{r(j)}{u(e)}} - \mu^{\lambda_e(j)}\right) \\ &= u(e)\mu^{\lambda_e(j)}\left(\mu^{\frac{r(j)}{u(e)}} - 1\right) \\ &= u(e)\mu^{\lambda_e(j)}\left(2^{\frac{r(j)}{u(e)}\log \mu} - 1\right) \end{aligned}$$

Assumption (2) implies that $r(j) \leq \frac{u(e)}{\log \mu}$. Since $2^x - 1 \leq x$ for $x \in [0,1]$, we get

$$c_e(j+1) - c_e(j) \leq \mu^{\lambda_e(j)} r(j) \log \mu = \log \mu \left(\frac{r(j)}{u(e)} c_e(j) + r(j)\right).$$

Summing up over all the links and using the fact that the request β_j was accepted, we get:

$$\sum_{e} \left(c_e(j+1) - c_e(j)\right) \leq \log \mu \left(\rho(j) + |P_j| \cdot r(j)\right).$$

Using the fact that the number of hops in the path is less than L, and the assumption (1) that the profit was scaled appropriately, implies

$$\sum_{e} \left(c_e(j+1) - c_e(j)\right) \leq 2\rho(j) \log \mu.$$

The next step is to show that the profit due to requests that were routed by the offline algorithm but were rejected by the online algorithm is bounded by the sum of the lengths of all the edges.

Lemma 3.2 Let $\mathcal{Q}$ be the set of indices of the requests that were admitted (routed) by the offline algorithm but not by the online algorithm, and denote $\ell = \max\{\mathcal{Q}\}$. Then $\sum_{j \in \mathcal{Q}} \rho(j) \leq \sum_e c_e(\ell)$.

Proof: Let P'_j be the path used by the offline algorithm to route β_j, for $j \in \mathcal{Q}$. The fact that β_j was not admitted and monotonicity in j of the lengths $c_e(j)$ with respect to j imply

$$\rho(j) \leq \sum_{e \in P'_j} r(j) \frac{c_e(j)}{u(e)} \leq \sum_{e \in P'_j} r(j) \frac{c_e(\ell)}{u(e)}.$$

Summing over all $j \in \mathcal{Q}$, we get:

$$\sum_{j \in \mathcal{Q}} \rho(j) \leq \sum_{j \in \mathcal{Q}} \sum_{e \in P'_j} \frac{r(j)}{u(e)} c_e(\ell) \leq \sum_e c_e(\ell) \sum_{j \in \mathcal{Q}: e \in P'_j} \frac{r(j)}{u(e)} \leq \sum_e c_e(\ell).$$

The last inequality follows from the fact that the offline algorithm cannot exceed unit relative load at any instance in time. ∎

Lemmas 3.1 and 3.2 imply a polylogarithmic bound on the maximum possible ratio between the profits of the offline algorithm and the profit due to requests that were accepted by the online algorithm. It remains to show that if the online strategy decides to accept a request and route it along some path P, then there is indeed sufficient available capacity along P. The intuition is that the length of an edge when it is close to saturation is so large that it will never be chosen by the online algorithm. In the following lemma $\mathcal{A}$ denotes the set of requests that were *routed* by the online algorithm.

Lemma 3.3 For all edges $e \in E$, $\sum_{j \in \mathcal{A}, e \in P_j} r(j) \leq u(e)$.

Proof: Let β_j be the first request that caused the capacity violation on an edge e. This implies that $\lambda_e(j) > 1 - \frac{r(j)}{u(e)}$. Using assumption (2) and the definition of length, we have

$$c_e(j)/u(e) = \mu^{\lambda_e(j)} - 1 > \mu^{1 - \frac{1}{\log \mu}} - 1 = \mu/2 - 1 = LF$$

Assumption (1) gives

$$\frac{r(j)}{u(e)} c_e(j) > LF \cdot r(j) \geq \rho(j)$$

which means that request β_j could not have been routed through edge e. ∎

Note that the total profit of the offline algorithm can be bounded by the profit of the online algorithm plus the profit due to circuits that were accepted by the offline but rejected by the online. This fact, together with lemmas 3.1, 3.2, and 3.3 implies that the competitive ratio of the algorithm is bounded by $O(\log \mu)$. Note that if profit is proportional to the bandwidth, we have $O(\log \mu) = O(\log L)$, matching the lower bound in [8].

Theorem 3.4 ([8]) The PVC routing and admission control strategy shown in Figure 1 never violates the capacity constraints and accrues at least $O(1/\log \mu)$-fraction of the profit accrued by the optimal offline algorithm.

3.2 PVC Routing in Congestion-Minimization Model

In this section we will describe an online strategy for the PVC case that is competitive in the congestion-minimization model. This strategy was developed in [5]; we will present the analysis from [9]. As before, we defer the discussion of the SVC case to Section 4.

As in the throughput-maximization model, we define the length of a link to be exponential in the current congestion on this link. More precisely, we set

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

where $\mu = 1 + \gamma/2$ for some $0 < \gamma < 1$. The online strategy routes on the *shortest path* with respect to this length. (Recall that, in contrast to the throughput-maximization model, no rejections are allowed.)

For simplicity, we will assume that the congestion of the offline algorithm is 1. If it is not, one can use a binary doubling procedure to guess an approximation to this congestion, and scale the input appropriately [5]. The key to the proof of competitiveness of the above strategy is the following "stability condition":

Definition 3.5 ([9]) Let P be some existing $s-t$ route satisfying request β_j, and let P' be any $s-t$ path in G. We say that the algorithm is in a *stable state* if, for any P, P' and $k \geq j$, we have:

$$\sum_{e \in P} \mu^{\lambda_e(j)}(\mu^{r(j)/u(e)} - 1) \leq 2 \sum_{e \in P'} \mu^{\lambda_e(k)}(\mu^{r(j)/u(e)} - 1).$$

We will show that as long as the stability condition is satisfied, the congestion is small. First, observe that this condition is inductively satisfied since the relative loads do not decrease. Moreover, applying the condition with P' being the path P_j^* used by the optimum offline algorithm to route connection β_j, and using the fact that $\forall x \in [0,1]: \; 2(\mu^x - 1) \leq \gamma x$, we get:

$$\begin{aligned} \sum_{e \in P_j} \mu^{\lambda_e(j)}(\mu^{r(j)/u(e)} - 1) &\leq 2 \sum_{e \in P^*} \mu^{\lambda_e(k)}(\mu^{r(j)/u(e)} - 1) \\ &\leq \gamma \sum_{e \in P_j^*} \mu^{\lambda_e(k)} r(j)/u(e). \end{aligned}$$

Let $\mathcal{P}, \mathcal{P}^*$ be the set of paths used by the online and the offline algorithms to route the currently active connections, respectively. Summing over all currently active connections, and exchanging the order of summation yields

$$\sum_{e \in E} \sum_{P_j \in \mathcal{P} | e \in P_j} \mu^{\lambda_e(j)}(\mu^{r(j)/u(e)} - 1) \leq \gamma \sum_{e \in E} \mu^{\lambda_e(k)} \sum_{P_i^* \in \mathcal{P}^* | e \in P_i^*} r(i)/u(e).$$

Notice that the left hand-side is a telescopic sum for each edge e. Since the normalized load of the offline algorithm never exceeds 1, we have $\sum_{e \in E} \mu^{\lambda_e} \leq m/(1-\gamma)$. Thus, the ratio between the congestion of the online algorithm and the congestion of the offline algorithm (which we assumed to be 1) never exceeds $O(\log m)$, which implies the following claim.

Theorem 3.6 ([5]) The congestion-minimization routing algorithm is $O(\log n)$-competitive.

4 Switched Virtual Circuits Routing

The strategies presented in Sections 3.1 and 3.2 work only for routing PVC circuits. In this section we describe how to adapt these strategies routing Switched Virtual Circuits (SVC).

4.1 Known Holding Times

The PVC routing and admission control strategies described above can be easily extended to the SVC case when the holding time of a circuit becomes known upon the arrival of this circuit. We will refer to this case as the *known holding times* case. The idea, proposed in [13], is to "flatten" the time, i.e. duplicate the network for each time unit.

More precisely, let $\lambda_e(j, \tau)$ denote the congestion on edge e at time τ as measured upon arrival of request β_j. In other words, this is the congestion on e at time τ due to the load created by requests $\beta_1, \ldots, \beta_{j-1}$. Observe that $\lambda_e(j, \tau)$ is non-decreasing with j. For the throughput-maximization case, the length associated with edge e and time instance τ is defined by

$$c_e(j, \tau) = u(e)(\mu^{\lambda_e(j,\tau)} - 1), \tag{3}$$

where $\mu = 2LTF + 1$, and F is defined by (1). If $e \in P_j$, then e's contribution to the length of the path P_j is computed as:

$$\sum_{\tau} \frac{r(j, \tau)}{u(e)} c_e(j, \tau) = \sum_{\tau} r(j, \tau)(\mu^{\lambda_e(j,\tau)} - 1), \tag{4}$$

If there exists a path whose length is bounded by the profit $\rho(j)$, then this path is used to route the connection β_j. Otherwise, the connection is rejected. The modified algorithm is presented in Figure 1.

It is straightforward to adapt the proofs of Lemmas 3.1, 3.2, and 3.3 to the SVC case. The additional requirement needed to satisfy Lemma 3.1 is that $\rho(j) \geq Lr(j)T(j)$, and the condition that is sufficient for

SVC_ROUTE($\beta_j = (s, t, r(\tau), T^s, T^f, \rho)$):

$\forall \tau, e \in E:\ c_e(j,\tau) \leftarrow u(e)(\mu^{\lambda_e(j,\tau)} - 1)$;

if $\exists$ path P in $G(V,E)$ from s to t s.t.

$$\sum_{e \in P} \sum_{T^s \leq \tau < T^f} \frac{r(\tau)}{u(e)} c_e(j,\tau) \leq \rho$$

then route the connection on P, and set:

$\forall e \in P, T^s \leq \tau \leq T^f$,

$\lambda_e(j+1,\tau) \leftarrow \lambda_e(j,\tau) + \frac{r(\tau)}{u(e)}$

else reject the connection

Figure 1: The SVC routing and admission control strategy for the throughput-maximization model.

Lemma 3.3 is that $\mu/2 - 1 \geq \rho(j)$. Thus, instead of the assumptions (1) and (2) we have to assume that:

$$1 \leq \frac{1}{L} \cdot \frac{\rho(j)}{r(j,\tau)T(j)} \leq F \tag{5}$$

$$r(j,\tau) \leq \frac{\min_e\{u(e)\}}{\log(2LTF+1)} \tag{6}$$

As in the PVC case, (5) can be always satisfied by taking a large enough value of F. Also, without assumption (6), it is impossible to achieve a polylogarithmic competitive ratio [8]. For example, setting $\rho(j) = LT(j)r(j)$ and $\mu = 2LT$ satisfies the above assumption and leads to $O(\log \mu) = O(\log LT)$ competitive ratio. Maximizing $\sum_j \rho(j)$ in this case corresponds to maximizing the total bandwidth-duration product, i.e. throughput. If the goal is to maximize the total acceptance rate, then the profit should be constant per request. For example, defining $\rho(j) = LT_{max}r_{max}$ leads to competitive ratio of $O(\log(L(T_{\max}/T_{\min})(r_{\max}/r_{\min})))$.

The same idea of "flattening" the time scale can be applied to design competitive algorithms for congestion-minimization case. More precisely, define the length of an edge as follows:

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

where μ is defined as in Section 3.2. The strategy is to route request β_j on path P_j that minimizes

$$\sum_{\substack{e \in P_j \\ \tau \in [T^s, T^f]}} c_e(j,\tau)$$

The stability condition (Definition 3.5) can be adapted to the SVC case in a natural way by adding a summation over time. See [13] for a complete proof that the above strategy achieves $O(\log nT)$ competitive ratio.

4.2 Unknown Holding Times

The SVC routing strategies described above rely on a priori knowledge of holding times. In many real applications holding times become known only after termination of the call, and hence it is important to develop strategies for this case, which we refer to as the *unknown holding times* case.

If we have no information whatsoever about the holding times, it is easy to show that there can be no competitive throughput-maximization algorithm. Roughly speaking, the reason is as follows. First, observe that the online algorithm has to reject a request at some point. Say, it rejects the kth request in some sequence. Now assign arbitrary long holding time to this rejected request and make it the last request in the sequence. The offline algorithm will reject all the requests except this last one, and thus will get arbitrary larger throughput than the online algorithm. Similar argument shows that there can be no competitive algorithm that maximizes the number of routed requests [9].

The situation is not much better for the congestion-minimization case. It is easy to adapt the lower bound of Azar, Broder, and Karlin [12] to show $\Omega(n^{1/4})$ lower bound for the competitive ratio in this case,[1] assuming the underlying graph is directed. This adaptation results in a sequence where holding times of some of the requests are exponential in n, and hence does not preclude an $O(\log T)$-competitive algorithm. Ma and Plotkin [32] showed how to construct a different sequence where all holding times are polynomial, and improved this lower bound to $\Omega(\min\{n^{1/4}, T^{1/3}\})$. The lower bounds in [12, 32] apply even if the online algorithm is allowed to use randomization. It is interesting to note that no nontrivial lower bounds are known for the case where the underlying graph is undirected.

These lower bounds indicate that we should consider alternate models. One natural model, considered in [23], is to assume that the *distribution* on the holding time becomes known with the arrival of the request. Observe that in this case we have to compare to an offline algorithm that does not have a complete knowledge of the request sequence. More precisely, it should have a complete knowledge of the request parameters except the holding time. In this case it is possible to modify the throughput-maximization algorithm so that the expected throughput achieved by this strategy will be within $O(\log LT)$ factor of the expected throughput achieved by the optimum offline algorithm. Roughly speaking, the idea is to route a circuit with constant rate $r(j)$ as if its rate depends on time and is equal to $r(j,\tau) = r(j) \cdot \Pr\{j \text{ alive at } \tau\}$.

In case the holding times are distributed exponentially, the algorithm in [23] becomes significantly simpler and achieves $O(\log L \log\log L)$ competitive ratio with respect to expected throughput. This improvement is based on two ideas. First, the fact that the holding times are distributed according to an exponential distribution implies that the result of summing the cost of an edge over time is a function of the current load only, and hence can be written in a closed form. Second, μ can be taken to be independent of T. Intuitively, the reason that μ was taken to be proportional to T in the throughput-competitive SVC routing algorithm described in the previous section was to make sure that even if a link is currently saturated but will be free from the next time step on, the cost of this link (summed up over time) will exceed the maximum profit, preventing the algorithm from using this link. The fact that the holding times are distributed exponentially implies that the probability of such even is very low.

An alternative model is to allow rerouting. Observe that if the number of reroutings per connection is

[1] The result in [12] is an $\Omega(\sqrt{n})$ lower bound in the context of load-balancing. Adaptation of this lower bound to the routing model results in an $\Omega(\sqrt[4]{n})$ lower bound.

not limited, it is trivial to maintain optimum relative load, i.e. competitive ratio of 1. Thus, we should allow only *limited rerouting*. In [9], it was shown that by allowing each circuit to be rerouted at most $O(\log n)$ times during its lifetime, we can maintain $O(\log n)$ competitiveness with respect to maximum congestion. Roughly speaking, the idea is to make sure that the stability condition (Definition 3.5) is always satisfied. If one of the routed circuits does not satisfy the condition, it is rerouted onto the shortest possible path. Since this causes its length to go down by at least a factor of 2, one can show that it will not be rerouted more than $O(\log n)$ times.

5 Routing Multicast Circuits

Similar to a virtual circuit request, a *multicast* request is specified by the required bandwidth, holding time, and the set of participating nodes. In order to satisfy a multicast request, we need to allocate the required bandwidth along a *tree* connecting the *set* of participating nodes. This is in contrast to the virtual circuit routing problem, where we needed to allocate bandwidth along an appropriate path.

The multicast case can be treated in exactly the same way as we have treated the virtual circuit routing problem in the previous sections, leading to strategies with exactly the same bounds. The only difference is that instead of finding a shortest path connecting the source and the sink nodes, we need to find a *minimum-cost Steiner tree* spanning the set of participating nodes. For example, to see that this is true for the throughput-maximization case, consider the proof of Theorem 3.4. Observe that the fact that we route along a path rather than some other subgraph of the network is used only in one place: in equation 3 we used the fact that the path can include at most L links. Thus, the same approach will work for multicast routing, with L replaced by L', the maximum number of links in the tree.

Although finding minimum-cost Steiner trees is NP-hard, it is sufficient to use a simple polynomial-time 2-approximation algorithm. One interesting variant of multicast routing is when we are allowed to choose a subset of the nodes participating in the multicast group, and allocate a tree spanning only these nodes. Naturally, the profit in this case should be proportional to the number of spanned (satisfied) nodes. A profit-competitive strategy for this case is to find the largest subset of the multicast group that can be spanned by a steiner tree whose cost is below the profit scaled by the size of this subset [8]. This is an NP-hard problem. Heuristics for solving this problem were given in [18]; an $O(\log^2 n)$-approximation algorithm is described in [7].

6 Implementation Issues

The previous sections presented the competitive routing and admission control strategies from a purely theoretical point of view. In this section we will discuss some of the issues involved in applying these ideas to realistic scenarios.

Should we implement the throughput-maximization strategy analyzed in Section 4 "as is" ? This question was studied in [22], where it was observed that using this strategy "as-is" does not lead to sufficiently good results. In fact, the resulting performance was observed to be always worse than the performance of a simple

greedy min-hop strategy.

The main problem is that the analysis suggests a huge value of $\mu = 2LTF + 1$, proportional to the largest holding time T, which leads to a very conservative admission control strategy. Indeed, consider a single link network ($L = 1$) where at time t_0 we are trying to route request β_j with unit holding time. Consider the case where the current congestion of the link is $\lambda(j, t_0) = 1/\log T$. Observe that if we follow the throughput-maximization strategy exactly, then we will not be able to route this request, even though the link is essentially unused.

Recall that μ was determined in order to satisfy Lemma 3.3. Roughly speaking, the analysis suggested that μ should be large enough so that the cost of any path that has a saturated link will exceed the maximum profit. In [23] it was observed that in many realistic situations the cases where the link is saturated at t and is totally unused at $t + 1$ are rare. Using this observation they showed that if one assumes exponential distribution of the holding times, then a much smaller $\mu = O(L \log L)$ is sufficient if the strategy is modified to reject a request that has a saturated link on each one of the shortest paths connecting its endpoints.

Simulation studies of this modified strategy (referred to as the "EXP" strategy in [22]) show that it outperforms many natural algorithms, including several variants of min-hop and reservation-based algorithms on several commercial topologies. Somewhat surprisingly, the EXP strategy leads to shorter average hop-count than the min-hop based algorithms.

An important question is how to implement shortest-path-based strategies in a distributed environment. Namely, in the environment where there is no central node that makes all the routing decisions and that has a complete knowledge of the state of the system. There are two main issues that have to be addressed. First, the routing should be *concurrent*, i.e. several nodes might be trying to create routes at the same time. Second, the strategy has to work even with *outdated data* about the global state of the system.

Concurrent routing was considered from a theoretical point of view by Awerbuch and Azar in [6]. They showed that by *iterative flooding* over all of the possible routes, one can route all of the requests in a logarithmic number of "flooding rounds" if several nodes are trying to route concurrently. Their strategy leads to a polylogarithmic competitive ratio if the number of different possible routes between any two points is very small. Moreover, their strategy is resilient against non-malicious failures of the nodes. It is not known how to treat the case where there are node pairs that are connected by a large number of different possible routes.

In order to make sure that each node in the network has a recent view of the global state of the network, one can periodically flood the network with state updates. An alternative approach, suggested in [22], is to allow each node to update its view of the network only when it decides on a route. More precisely, during the creation of the route, the node should update its information about all the edges on the route. If during the creation of the route a saturated edge is encountered, the strategy is to try again, up to some parameter k tries, and then reject the request.

The advantage of this approach is that has relatively low overhead and does not require an external "update procedure". Simulation results described in [22] indicate that although for $k = 1$ (i.e. reject if the path chosen by looking only at the locally available information has a saturated link) the performance is worse than the performance of the centralized strategy, it is significantly better than the performance of min-hop based strategies. Simulations also indicate that already for small values of k, the performance

essentially reaches that of the centralized algorithms.

Intuitively, the distributed strategy based on the exponential length function outperforms the min-hop strategy because it tries to choose paths that are the least saturated according to its (outdated) information. This is in contrast to min-hop based strategy, that tries to route over arbitrary paths that had sufficient available capacity according to the (same) outdated data. Thus, the paths chosen by the exponential-length based strategy are more likely to have sufficient available capacity at present.

7 Conclusions

In this paper we have surveyed a class of routing and admission control strategies based on assigning each link a length that is an exponential function of the current congestion on this link, and routing along the shortest path with respect to this length; if no sufficiently short path exists, the request is rejected. From the theoretical point of view, this approach is attractive because it unifies the routing and admission control decisions. Moreover, it was shown that the appropriate choice of the length function results in algorithms that outperform min-hop based and reservation-based algorithms on sparse network topologies.

The algorithms were motivated by using *competitive ratio* to compare performance of various strategies. The main advantage of this measure is that it allows us to compare strategies without any assumptions on the statistical behavior of the offered traffic. By incorporating statistical assumptions into algorithms motivated by competitive analysis we get algorithms that have good performance under these assumptions, and at the same time do not "break down" when these assumptions are not satisfied.

Although all of the strategies presented in this paper are dealing with managing the available bandwidth, similar competitive approaches can be applied to design strategies for managing other resources in the network, such as the available buffer space, processing load, etc.

Much work remains to be done. One potential problem with the strategies described in this paper is that they do not address the fairness issues. For example, it is easy to see that the admission control strategy described in Section 4 will be much more inclined to reject high-bandwidth requests than low-bandwidth ones. Another problem stems from the fact that the competitive ratio is too pessimistic a measure, since it compares to a very powerful omniscient offline algorithm that can not be implemented. Several new measures that are more refined than competitive ratio were recently proposed in [29]. An interesting research direction is to try to design strategies based on these new measures. Another promising research direction is to combine the ideas of the competitive online routing with the more traditional reservation-based approaches [3, 27, 34, 35, 36, 30, 38] in order to derive strategies that have good performance both for dense and for sparse topologies.

Acknowledgments

I would like to thank Baruch Awerbuch, Yossi Azar, Rainer Gawlick, Anil Kamath, Debasis Mitra, Omri Palmon, Ram Ramakrishnan, and Orli Waarts for helpful discussions. Alan Borodin's comments helped in improving the clarity of the presentation. I would also like to thanks Jeffrey Oldham for his numerous

comments on the early draft of this paper.

References

[1] G.R. Ash. Use of trunk status map for real-time DNHR. In *Proc. of ITC-12, Torino, Italy*, 1988.

[2] G.R. Ash, R.H. Cardwell, and R.P. Murray. Design and optimization of networks with dynamic routing. *Bell Syst. Tech. J.*, 60:1787–1820, 1981.

[3] G.R. Ash, J.S. Chen, and A.E. Frey andB.D. Huang. Real-time network routing in an integrated network. In *Proc. of ITC-13, Copenhagen, Denmark*, 1991.

[4] G.R. Ash, A.H. Kafker, and K.R. Krishnana. Servicing and real-time control of networks with dynamic routing. *Bell Syst. Tech. J.*, 60:1821–1845, 1981.

[5] 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.

[6] B. Awerbuch and Y. Azar. Local optimization of global objectives: Competitive distributed deadlock resolution and resource allocation. In *Proc. 35th IEEE Annual Symposium on Foundations of Computer Science*, pages 240–249, November 1994.

[7] B. Awerbuch, Y. Azar, A. Blum, and S. Vempala. Improved approximation guarantees for minimum-weight k-trees, prize-collecting salesmen, and bank robbers. Unpublished manuscript, 1994.

[8] 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.

[9] B. Awerbuch, Y. Azar, S. Plotkin, and O. Waarts. Competitive routing of virtual circuits with unknown duration. In *Proc. 5th ACM-SIAM Symposium on Discrete Algorithms*, pages 321–327, January 1994.

[10] B. Awerbuch, R. Gawlick, T. Leighton, and Y. Rabani. On-line admission control and circuit routing for high-performance computing and communication. In *Proc. 35th IEEE Annual Symposium on Foundations of Computer Science*, pages 412–423, November 1994.

[11] B. Awerbuch, Y.Bartal, A. Fiat, and A. Rosen. Competitive non-preemptive call control. In *Proc. 5th ACM-SIAM Symposium on Discrete Algorithms*, 1994.

[12] Y. Azar, A. Broder, and A. Karlin. On-line load balancing. In *Proc. 33rd IEEE Annual Symposium on Foundations of Computer Science*, pages 218–225, 1992.

[13] Y. Azar, B. Kalyanasundaram, S. Plotkin, K. Pruhs, and O. Waarts. On-line load balancing of temporary tasks. In *Proc. Workshop on Algorithms and Data Structures*, pages 119–130, August 1993.

[14] D. Bertssekas and R. Gallager. *Data Networks, 2nd edition*. Prentice Hall, 1992.

[15] A. Borodin, N. Linial, and M. Saks. An optimal online algorithm for metrical task systems. *J. ACM*, (39):745–763, 1992.

[16] W.H. Cameron, J. Regnier, P. Galloy, and A.M. Savoie. Dynamic routing for intercity telephone networks. In *Proc. of ITC-10, Montreal, Canada*, 1983.

[17] F. Caron. Results of the Telecom Canada high performance routing trial. In *Proc. of ITC-12, Torino, Italy*, 1988.

[18] S.Y Cheung and A. Khumar. Efficient quorumcast routing algorithms. In *Proc. INFOCOM '94, Volume 2*, 1994.

[19] J. Garay, I. Gopal, S. Kutten, Y. Mansour, and M. Yung. Efficient on-line call control algorithms. In *Proc. of 2nd Annual Israel Conference on Theory of Computing and Systems*, 1993.

[20] J.A. Garay and I.S. Gopal. Call preemption in communication networks. In *Proc. INFOCOM '92*, volume 44, pages 1043–1050, Florence, Italy, 1992.

[21] R. Gawlick, C. Kalmanek, and K. Ramakrishnan. On-line permanent virtual circuit routing. In *Proceedings of IEEE Infocom*, April 1995.

[22] R. Gawlick, A. Kamath, S. Plotkin, and K. Ramakrishnan. Routing and admission control of virtual circuits in general topology networks. Technical Report BL011212-940819-19TM, AT&T Bell Laboratories, 1994.

[23] R. Gawlick, A. Kamath, S. Plotkin, and K. Ramakrishnan. Routing of virtual circuits with unknown holding times. Unpublished manuscript, 1994.

[24] A. Girard and M.A. Bell. Blocking evaluation for netowkrs with residual capacity adaptive routing. *IEEE Trans. Commun.*, 37:1372–1380, 1989.

[25] A.R. Karlin, M.S. Manasse, L.Rudolph, and D.D. Sleator. Competitive snoopy caching. *Algorithmica*, 1(3):70–119, 1988.

[26] F. P. Kelly. Blocking probabilities in large circuit-switched networks. *Adv. Appl. Prob.*, 18:473–505, 1986.

[27] F. P. Kelly. Network routing. *Phil. Trans. R. Soc. Lond.*, pages 343–367, 1991.

[28] P. Klein, S. Plotkin, C. Stein, and É. Tardos. Faster approximation algorithms for the unit capacity concurrent flow problem with applications to routing and finding sparse cuts. *SIAM Journal on Computing*, 23(3):466–487, June 1994.

[29] E. Koutsoupias and C.H. Papadimitriou. Beyond competitive analysis. In *Proc. 35th IEEE Annual Symposium on Foundations of Computer Science*, pages 394–400, November 1994.

[30] K.R. Krishnan and T.J. Ott. Forward-looking routing: A new state-dependent routing scheme. In *Proc. of ITC-12, Torino, Italy*, 1988.

[31] T. Leighton, F. Makedon, S. Plotkin, C. Stein, É. Tardos, and S. Tragoudas. Fast approximation algorithms for multicommodity flow problem. In *Proc. 23th ACM Symposium on the Theory of Computing*, pages 101–111, May 1991.

[32] Y. Ma and S. Plotkin. Improved lower bounds for load balancing of tasks with unknown duration. Unpublished manuscript, 1995.

[33] M.S. Manasse, L.A. McGeoch, and D.D. Sleator. Competitive algorithms for online problems. In *Proc. 20th Annual ACM Symposium on Theory of Computing*, pages 322–332, 1988.

[34] D. Mitra, R.J. Gibbens, and B.D. Huang. State-dependent routing on symmetric loss networks with trunk reservations – I. *Annals of Operations Research*, (35):3–30, 1992.

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

[36] T.J. Ott and K.R. Krishnan. State-dependent routing of telephone traffic and the use of separable routing schemes. In *Proc. of ITC-11, Kyoto, Japan*, 1985.

[37] F. Shahrokhi and D. Matula. The maximum concurrent flow problem. *J. Assoc. Comput. Mach.*, 37:318–334, 1990.

[38] S. Sibal and A. DeSimone. Controlling alternate routing in general-mesh packet flow networks. Technical Report BL045370F-940603-01TM, AT&T, June 1994.

[39] D.D. Sleator and R.E. Tarjan. Amortized efficiency of list update and paging rules. *Comm. ACM*, 28(2):202–208, 1985.

[40] A.C. Yao. New algorithms for bin packing. *Journal of the ACM*, 27:207–227, 1980.